Project report submitted in partial fulfillment for the Degree of B. Tech in Applied Electronics & Instrumentation Engineering under Maulana Abul Kalam Azad University of Technology

DIGITAL MULTIMETER

By
Debajyoti Chakraborty (Roll No. 11705515012)
Mainak Sarkar (Roll No. 11705515025)
Sangik Dutta (Roll No. 11705515039)
Susmit Bhattacharya (Roll No. 11705515050)

Under Supervision of
Mr. Avishek Paul
Assistant Professor, Dept. of AEIE



DEPARTMENT OF APPLIED ELECTRONICS &
INSTRUMENTATION ENGINEERING,
RCC INSTITUTE OF INFORMATION TECHNOLOGY,
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700015,
May, 2019

ACKNOWLEDGEMENT

It is a great privilege for us to express our profound gratitude to our respected teacher
Mr. Avishek Paul, Applied Electronics & Instrumentation Engineering, RCC Institute of Information Technology, for his/her constant guidance, valuable suggestions, supervision and inspiration throughout the course work without which it would have been difficult to complete the work within scheduled time.

We would like to express our gratitude towards all the faculty members for his/her kind co-operation and encouragement which helped us in completion of this project.

We are also indebted to the Head of the Department, Applied Electronics & Instrumentation Engineering, RCC Institute of Information Technology for permitting us to pursue the project.

We would like to take this opportunity to thank all the respected teachers of this department for being a perennial source of inspiration and showing the right path at the time of necessity.

Debajyoti Chakraborty
(11705515012)

Mainak Sarkar
(11705515025)

Sangik Dutta
(11705515039)

Susmit Bhattacharya
(11705515050)

RCC INSTITUTE OF INFORMATION TECHNOLOGY
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700 015
PHONE : 2323 2463 FAX : (033)2323 4668
E-mail : campus@rcciit.in Website : www.rcciit.org

CERTIFICATE OF APPROVAL

The project report titled "Digital Multimeter" prepared by Debajyoti Chakraborty(11705515012); Mainak Sarkar(11705515024); Sangik Dutta(11705515039); Susmit Bhattacharya(11705515050) is hereby approved and certified as a creditable study in technological subjects performed in a way sufficient for its acceptance for partial fulfilment of the degree for which it issubmitted.

It is to be understood that by this approval, the undersigned do not, necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it is submitted.

Mr. Avishek Paul
Project Supervisor
Applied Electronics & Instrumentation Engineering

Mr. Arijit Ghosh
Head of The Dept.
Applied Electronics & Instrumentation Engineering

Examiner

Examiner

RCC INSTITUTE OF INFORMATION TECHNOLOGY
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700 015
PHONE : 2323 2463 FAX : (033)2323 4668
E-mail : campus@rcciit.in Website : www.rcciit.org

RECOMMENDATION

I hereby recommend that the project report titled "Digital Multimeter" prepared by Debajyoti Chakraborty(11705515012); Mainak Sarkar(11705515025); Sangik Dutta(11705515039); Susmit Bhattacharya(11705515050) be accepted in partial fulfillment of the requirement for the Degree of Bachelor of Technology in Applied Electronics & Instrumentation Engineering, RCC Institute of Information Technology.

Mr. Avishek Paul
Project Supervisor
Applied
Electronics & Instrumentation
Engineering

Mr. Arijit Ghosh

Head of the Dept .
Applied Electronics & Instrumentation
Engineering

RCC INSTITUTE OF INFORMATION TECHNOLOGY

CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700 015

PHONE : 2323 2463 FAX : (033)2323 4668

E-mail : campus@rcciit.in Website : www.rcciit.org

Abstract

In the recent years, a Multimeter has seen rapid changes due to introduction of various measurement technologies. The Digital Multimeter is supposed to be implemented in between the other existing measuring instruments, without making any changes in basic structure. This system is most suitable for basic fault finding and field service work. Multimeter is intended as a bench instrument which can measure to a very high degree of accuracy. So in this paperwork our aim is to design a digital multimeter for measurement purpose.

LIST OF FIGURES

Figure 1: Analog Multimeter 09

Figure 2: Digital Multimeter 09

Figure 3: Basic Block Diagram Of The Project 10

Figure 4: Analog to Digital Converter 12

Figure 5: Basic Circuit Diagram of +5V Power Supply Using Multisimulator 15

Figure 6: LCD Display 17

Figure 7: ATMega328P Microcontroller Pinout 20

Figure 8: Continuity Test Circuit 25

Figure 9: Simulated Continuity Test Circuit by Proteus Software 25

Figure 10: DC Voltage Measurement Circuit 28

Figure 11: Simulated DC Voltage Measurement Circuit 28

Figure 12: AC Voltage Measurement Circuit 31

Figure 13: Simulated AC Voltage Measurement Circuit 31

Figure 14: Hardware implementation 33

Figure 15: Hardware implementation of digital multimeter 33

Figure 16: Continuity Check Circuit 35

Figure 17: DC Voltage Check Circuit Result 1 35

Figure 18: DC Voltage Check Circuit Result 2 35

Figure 19: DC Voltage Check Circuit Result 3 35

Figure 20: DC Voltage Check Circuit Result 4 35

Figure 21: Continuity Test Circuit 36

Figure 22: AC Voltage Measurement Circuit 36

Figure 23: DC Voltage Measurement Circuit 36

LIST OF TABLE

Table 1 : Pin Configuration of LCD Display 17

Table 2 : Key Parameter of Microcontroller 19

Contents

Chapter 1 Literature Review ……..06

Chapter 2 Introduction ……...09

2.1 Block Diagram ……..10
2.1.1 Power Supply……10
2.1.2 Current to Voltage Converter……...10
2.1.3 Rectifier Circuit ……11
2.1.4 Analog to Voltage Converter ……...12

Chapter 3

3.1 Power Supply……14

3.1.1 Circuit Diagram ……15

3.2 LCD Display ……16

3.2.1 16x2 LCD Display Figure with Pin Configuration…17

Chapter 4 Microcontroller ……...18

4.1 Specification ...……...19

4.2 Pin Configuration of ATmega328P ……..20

Chapter 5 Interfacing of ATmega328P with LCD Display ……21

5.1 Circuit Diagram of Interfacing ……..22

Chapter 6 Continuity Test Circuit ……24

6.1 Circuit Diagram of Continuity Test Circuit ……25

6.2 Simulated Circuit Diagram of Continuity Test Circuit ……25

Chapter 7 DC Voltage Measurement Circuit ……27

7.1 Circuit Diagram of DC Voltage Measurement Circuit ……28

7.2 Simulated Circuit Diagram of DC Voltage Measurement Circuit …...28

Chapter 8 AC Voltage Measurement Circuit ……29

8.1 Circuit Diagram of AC Voltage Measurement Circuit ……..30

8.2 Simulated Circuit Diagram of AC Voltage Measurement Circuit …...30

Contents

Chapter 9 Hardware Implementation …….32

Chapter 10 Result ……..34

Chapter 11 Conclusion ……..37

Chapter 12 Reference …….39

Appendix A Functional Flowchart or Programs ……...41

Appendix B Datasheet ……43

CHAPTER 1

Literature Review

Literature Review:

1. The electrical industries consist of many electrical equipment, which are installed for different applications. Whenever there would be any fault or problem occur in any machine or a part of system, complete system is being turned off by isolating the power supply. In this procedure, the equipment who is working without fault also stops. This project is all about dealing with these problem, we are designing & making a device which is in the form of a Advanced Multi-meter with additional feature such as locating of switch in the panel Whenever there is any fault in any part of system, our Device can be used as a detector by swapping the multi-meter across equipment which has a problem and immediately a LED light will glow near the switch which will help to detect the switches belonging to faulty equipment.
2. Measurement of electrical parameters like current and voltage are crucial in field of measurement. Frequency indicate the state of rhythmic behavior of signal and proper change in system with respect to time. Multimeter are used in measuring various electrical parameters including resistance, capacitance and inductance. This paper shed lights on the development of a low cost Multimeter using digital electronic circuitry. The proposed device is able to measure various electrical parameter in the certain specified range. A simple, but highly accurate, sampling method for the measurement of frequency in combination of other parameters are corporate here by means of a digital multimeter(DMM). In modern electronics laboratory a digital multimeter with high precision and digital displaying mechanism is an essential device. This paper describes the mechanism of a digital multimeter that can measure electrical parameters in under the particular range with specified resolution. accuracy of digital multimeter discussed in this paper is quite good ,it can be used where precise measurement at low cost is required.
3. The major goal for designing any measuring instrument is to obtain a device that can measure with high precision the quality with which it was design to measure and obtain a minimal error.
4. Over the years, different types of digital multimeters have been designed to measure different parameters like voltage, current, resistance etc.
5. There are digital multimeters that can measure different types of parameters in a very precise manner. We can measure AC voltage, DC voltage, AC current, DC current, resistance etc.
6. In this paper, we have first reviewed the recently proposed using different parameters through them we are getting different values. But those are not accurate. So in this we are implementing digital multimeter which is more precise.
7. Multimeter is very important instrument in case of measureing purpose. It is very helpful to implementing any project or any industry.
8. It is very vital step to understand the basics, i.e. the advantages , disadvantages, requirements and flexibility of the instrument.

CHAPTER 2

Introduction

Introduction

A Multimeter is an electronic instrument or device which can measure several electronic components or electrical properties. It is also known as Multi-tester or Volt-Ohm Milliammeter. By using a multimeter we can measure voltage, current and resistance. There are basically two types of multimeter.

In case of electronic measurement devices there are different types of instruments or devices like voltmeter, ammeter etc. Multimeter is a modern device which is basically the combination of ammeter and voltmeter. There are analog multimeters as well as digital multimeters. Analog multimeters uses a microammeter with a moving pointer to display readings. But in case of Digital Multimeters, they consist a numeric display and may also show the graphical bar representing the measured value. Digital Multimeters are more presised than analog multimeters. But in some cases analog multimeters are more useful when we have to measure any rapidly changed value.

This device is also very useful for fault finding and field service work. It is a bench device which can measure to a very high degree of accuracy. By using a multimeter we can check the continuity of a circuit. It can measure AC & DC voltages , resistance as well as AC & DC current. Today's advance multimeters come with additional features and measures frequency, temperature and capacitance also. Some multimeters have the function of testing diode and lets the current flow in one direction in the circuit.



Figure 1: Analog multimeter



Figure 2: Digital multimeter

Block Diagram

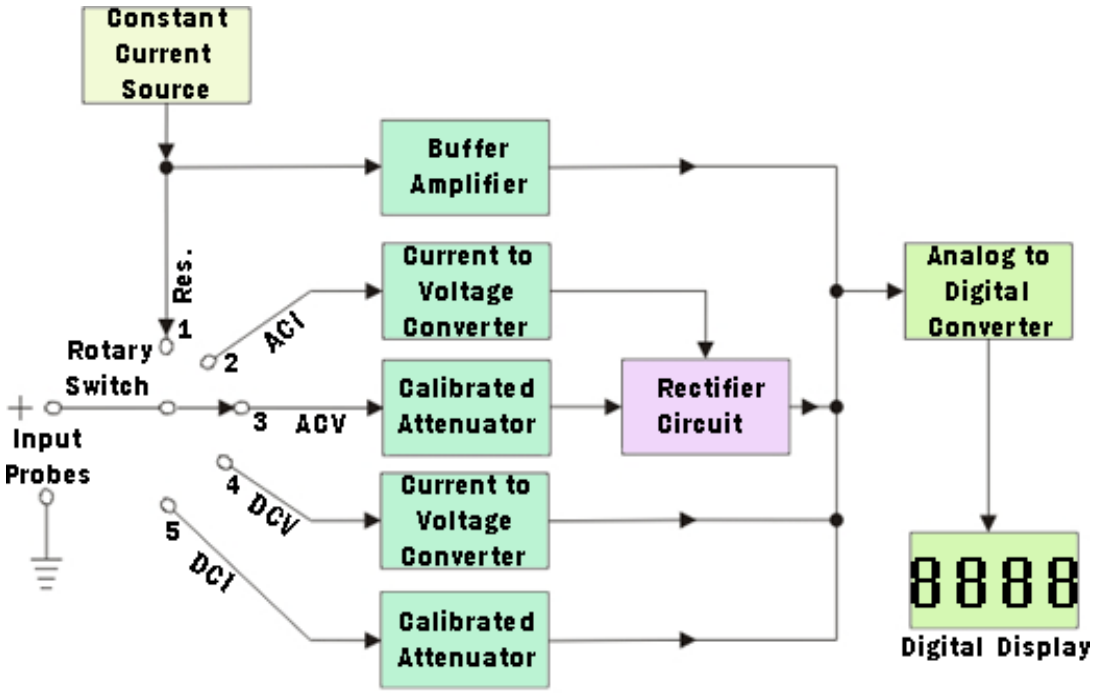


Figure 3: Block Diagram of the Project

2.1.1 Power Supply Unit

In this project we have made a regulated power supply unit with a two various types of outputs voltage options. For that purpose we used the voltage regulator of series 78XX and 79XX for positive and negative output voltage respectively.

2.1.2 Current to voltage converter

This circuit converts an input current to a proportional amount of voltage. Use the two switches to select an input current. This current flows across a 1k resistor. The op-amp outputs a voltage equal to the voltage drop across the resistor in order to ensure that the – terminal is at ground, which means that the output voltage is proportional to the resistor voltage (and therefore to the input current).

2.1.3 Rectifier Circuit

One of the important applications of a semiconductor diode is in rectification of AC signals to DC. Diodes are very commonly used for obtaining DC voltage supplies from the readily available AC voltage.

There are many possible ways to construct rectifier circuits using diodes. The three basic types of rectifier circuits are:

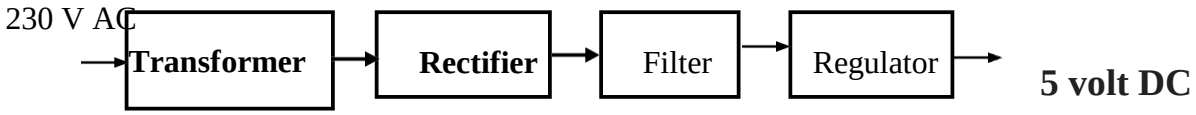
- The Half Wave Rectifier
- The Full Wave Rectifier
- The Bridge Rectifier

Diode rectifier for power supply

The purpose of a power supply is to take electrical energy in one form and convert it into another. There are many types of power supply. Most are designed to convert high voltage AC mains electricity to a suitable low voltage supply for electronics circuits and other devices such as computers, fax machines and telecommunication equipment. In Singapore, supply from 230V, 50Hz AC mains is converted into smooth DC using AC-DC power supply.

A power supply can by broken down into a series of blocks, each of which performs a particular function. A transformer first steps down high voltage AC to low voltage AC. A rectifier circuit is then used to convert AC to DC. This DC, however, contains ripples, which can be smoothened by a filter circuit. Power supplies can be 'regulated' or 'unregulated'. A regulated power supply maintains a constant DC output voltage through 'feedback action'. The output voltage of an unregulated supply, on the other hand, will not remain constant. It will vary depending on varying operating conditions, for example when the magnitude of input AC voltage changes.

Main components of a regulated supply to convert 230V AC voltage to 5V DC are shown below.



2.1.4 Analog to digital converter:

When selecting an Analog to Digital converter (ADC) chip for an application, three characteristics must be considered. Similar to selecting a DAC, these characteristics are; number of bits, accuracy, and speed. There are three different methods used for constructing Analog to Digital converters. These ADC types are; Parallel Comparator or Flash type, Dual-Slope type, and Successive Approximation type.

Figure 6 displays a 3-bit, flash type ADC circuit. The voltage divider provides eight reference voltages on the negative poles of the operational amplifiers. The unknown input voltage is compared to each one of the reference voltages. If the input voltage is greater than any one of the reference voltages, the associated Op-Amp will be saturated to its positive supply voltage level. The eight-to-three encoder will generate an appropriate three bit binary word according to the inputs A0 through A7. For example, assume A4 is the only input to the encoder that is set. That would mean that the input voltage is 2.5V, and the output binary number will be 100.

The advantage of a Flash type ADC is its instantaneous speed. The Flash type ADC is expensive due to the large numbers of OP-Amp devices contained therein. For example, an 8-bit flash type ADC requires 2^8-1 = 255 Op-Amps. Flash type ADCs are used in special applications such as systems used by the military or NASA.

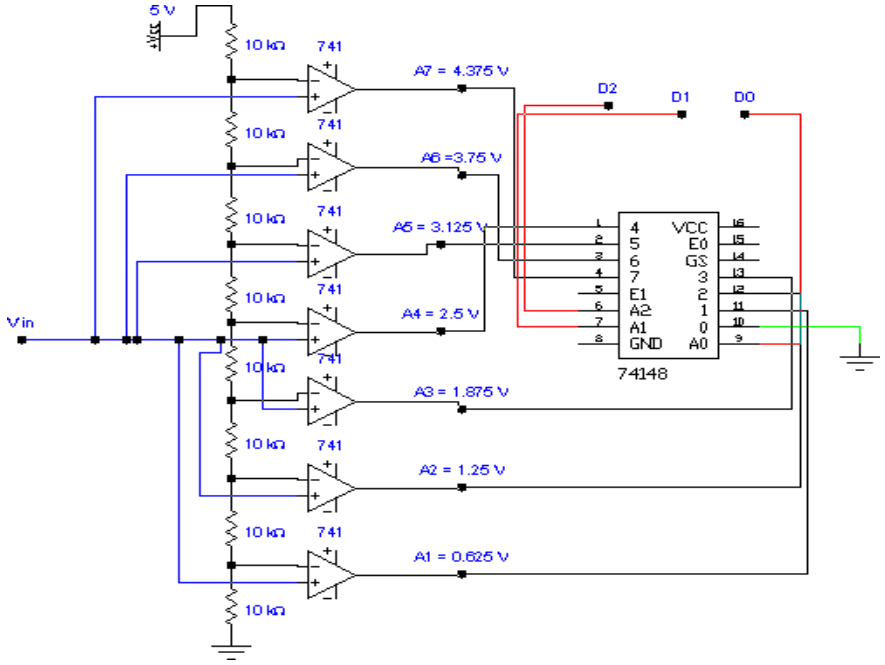


Fig : Analog to Digital Converter

CHAPTER 3

Power Supply

Power Supply

A regulated power supply is an embedded circuit. It converts unregulated AC into a constant DC. With the help of a rectifier it converts AC supply into DC. In this project we are using voltage regulator. A voltage regulator is designed to automatically maintain a constant voltage level. It may use an electromechanical mechanism, or electronic components. Depending on the design, it may be used to regulate one or more AC or DC voltages.

For serving our purpose we want +5V and -5V. So for that reason we use linear voltage regulator. In electronics, a linear regulator is a system used to maintain a steady voltage. The resistance of the regulator varies in accordance with the load resulting in a constant output voltage. The regulating device is made to act like a variable resistor, continuously adjusting a voltage divider network to maintain a constant output voltage, and continually dissipating the difference between the input and regulated voltages as waste heat. By constant, a switching regulator uses an active device that switches on and off to maintain an average value of output. Because the regulated voltage of a linear regulator must always be lower than input voltage, efficiency is limited and the input voltage must be high enough to always allow the active device to drop some voltage.

The 78XX(sometimes LM78XX, MC78XX…) is a family of self-contained fixed linear voltage regulator integrated circuits. The 78XX family is commonly used in electronic circuits requiring a regulated power supply due to their ease-of-use and low cost. For ICs within the family, the xx is replaced with two digits, indicating the output voltage(for example, the 7805 has a 5-volt output, while the 7812 produces 12 volts). The 78XX line are positive voltage regulators; they produce a voltage that is positive relative to a common ground. There is a related line of 79XX devices which complementary negative voltage regulators. 78XX and 79XX ICs have three terminals and are commonly found in the TO-220 form factor, although they are available in surface-mount, TO-92, and TO-3 packages. These devices support an input voltage anywhere from around 2.5 volts over the intended output voltage up to a maximum of 35 to 40 volts depending on the model, and typically provide 1 or 1.5 amperes of current(though smaller or larger packages may have a lower or higher current rating).

3.1.1 Circuit Diagram

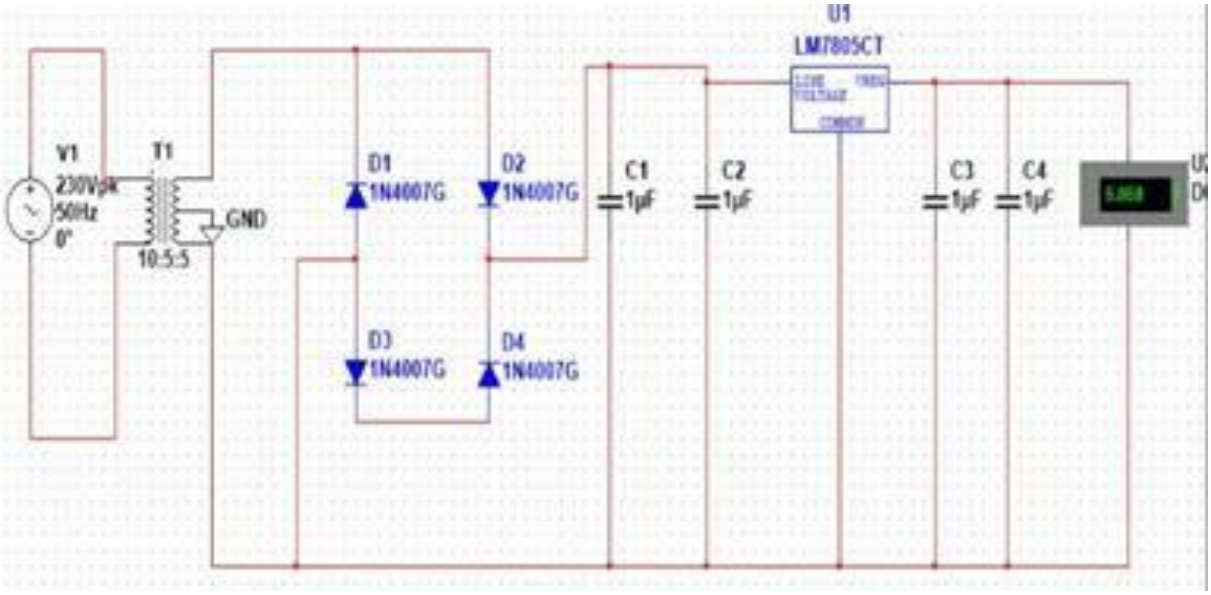


Figure 5: Circuit Diagram of +5V Power Supply Using Multisimulator

3.2 LCD Display

LCD (liquid crystal display) is the technology used for displays in notebook and other smaller computers. Like light-emitting diode(LED) and gas-plasma technologies, LCDs allow displays to be much thinner than cathode ray tube(CRT) technology. LCDs consume much less power than LED and gas-display displays because they work on the principle of blocking light rather than emitting it.

An LCD is made with either a passive matrix display grid. The active matrix LCD is also known as a thin film transistor(TFT) display. The passive matrix LCD has a grid of conductors with pixels located at each intersection in the grid. A current is sent across two conductors on the grid to control the light for any pixel. An active matrix has a transistor located at each pixel intersection requiring less current to control the luminance of a pixel. For this reason, the current in an active matrix display can be switched on and off more frequently, improving the screen refresh time.

Some passive matrix LCD's have dual scanning, meaning that they scan the grid twice with current in the same time that it look for one scan in the original technology. However, active matrix is still a superior technology.

16x2 LCD display figure with pin configurations

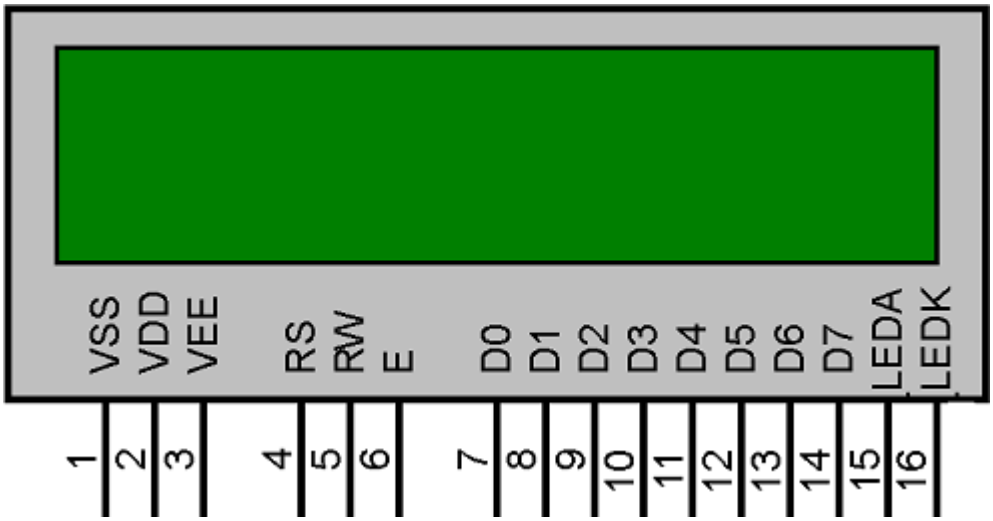


Figure 6: LCD Display

PIN NO	PIN FUNCTION
1	GROUND
2	+5 VOLT
3	CONTRAST
4	REGISTER SELECT
5	READ/WRITE
6	ENABLE
7	DATA PIN 0
8	DATA PIN 1
9	DATA PIN 2
10	DATA PIN 3
11	DATA PIN 4
12	DATA PIN 5
13	DATA PIN 6
14	DATA PIN 7
15	LED POSITIVE
16	LED NEGATIVE

Table 1 : Pin Configuration of LCD Display

CHAPTER 4

Microcontroller

Microcontroller

The ATmega328 is a single-chip microcontroller created by Atmel in the mega AVR family (later Microchip Technology acquired Atmel in 2016). It has a modified Harvard architecture 8- bit RISC processor core.

The Atmel 8-bit AVR RISC - based microcontroller combines 32 kB ISP flash memory with read- while-write capabilities, 1 kB EEPROM, 2kB SRAM 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, 6-channel 10-bit A/D converter, programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts. The device achieves throughput approaching 1MIPS per MHz.

4.1 Specification

PARAMETER	VALUE
CPU Type	8-bit AVR
Performance	20MIPS at 20MHz
Flash memory	32Kb
SRAM	2kB
EEPROM	1kB
Pin Count	28
Maximum operating frequency	20MHz
Number of touch channels	16
Maximum I/O pins	NO
External interrupts	23
USB Interface	02
USB Speed	-

Table 2: Specification of Microcontroller

Pin Configuration of ATMega328P

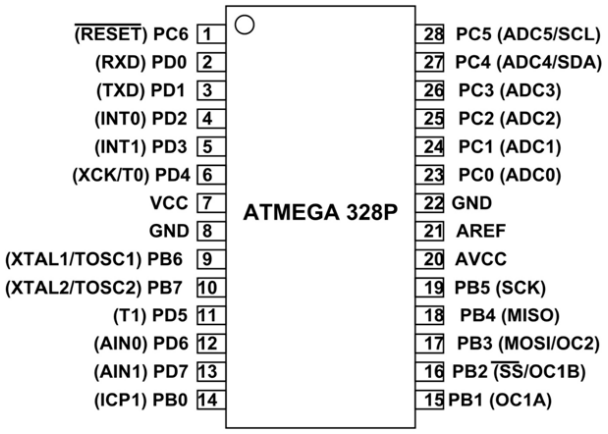


Figure 7: ATMega328P Microcontroller Pinout

CHAPTER 5

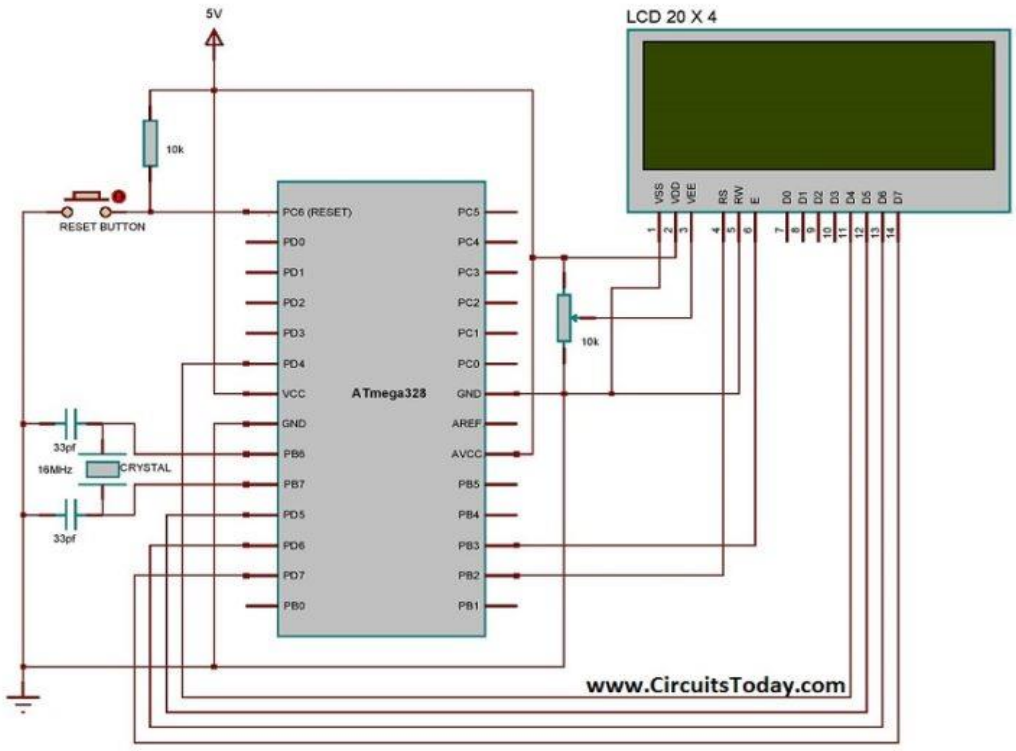
Pin layout of ATmega328P with LCD Display

Pin layout of ATmega328P with LCD Display

Circuit Diagram

As shown in the circuit diagram, port B and port D of the controller is used for interfacing it with LCD module. In 4 bit mode only 4 lines D4-D7, along with RS, R/W and E pins are used. This will save us 4 pins of our controller which we can use it for other purpose. Here we only need to write to the LCD module. So the R/W pin can be ground it as shown in the schematic diagram.

In this way the total number of pins can be reduced to 6. In 4 Bit mode the data bytes are split into two four bits and are transferred in the form of a nibble. The data transmission to a LCD is performed by assigning logic states to the control pins RS and E. The reset circuit, oscillator circuit and power supply need to be provided for the proper working of the circuit.



INTERFACING LCD MODULE WITH AVR IN 4-BIT MODE

CHAPTER 6

Continuity Test Circuit

Continuity Test Circuit

In electronics, a continuity test is the checking of an electric circuit to see if current flows. A continuity test is performed by placing a small voltage across the chosen path. If electron flow is inhibited by broken conductors, damaged components, or excessive resistance, the circuit is "open".

Devices that can be used to perform continuity tests include multimeters which measure current and specialized continuity testers which are cheaper, more basic devices, generally with a simple light bulb that lights up when current flows.

An important application is the continuity test of a bundle of wires so as to find the two ends belonging to a particular one of these wires; there will be a negligible resistance between the "right" ends, and only between the "right" ends.

A continuity test is mostly performed by providing a small voltage between the selected paths. If the flows of electrons are not continuous due to damaged components, broken conductors or high resistance, the circuit is considered as "open circuit". Multimeters and ohmmeters are normally used for continuity tests. Specialized continuity testers are also available which are more basic in nature, inexpensive and have a lightbulb which glows in the case of current flow. It can also help in determining if the soldering is good.

In case of a continuity test circuit of a multimeter, a buzzer is used instead of a LED light to detect the fault. If there are any short circuit connection or and damaged capacitor or resistance, the buzzer will give us the indication.

Figure shows a diagram of the continuity measurement function. The circuit uses two opamps and a piezoelectric buzzer. When the leads are connected across a circuit and the resistance less than 20Ω the circuit oscillates and the buzzer sounds.

6.1 Circuit Diagram of Continuity Test Circuit

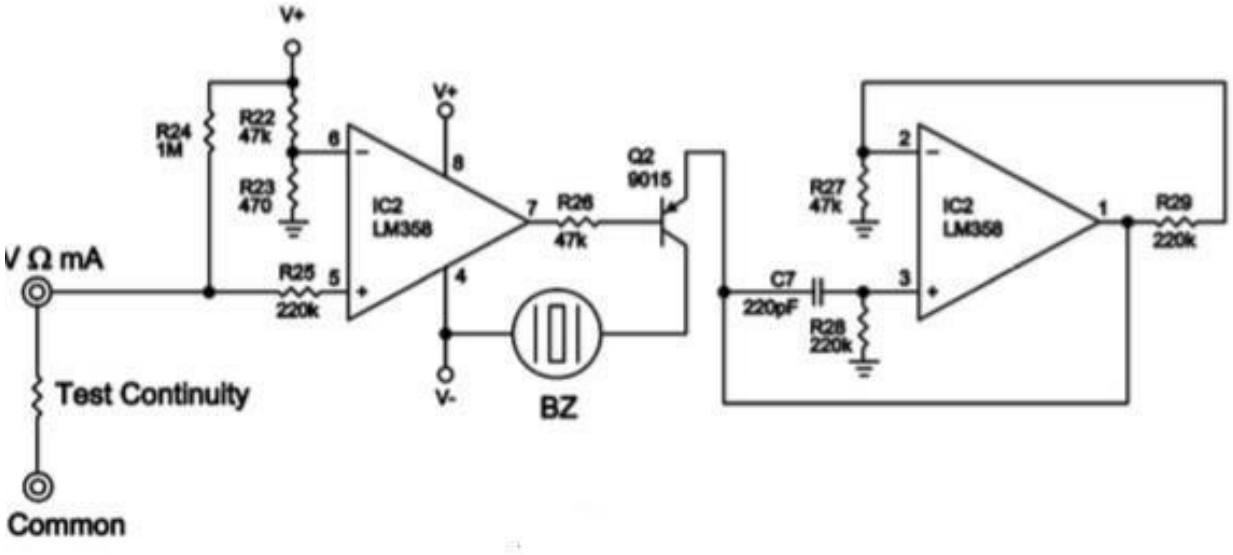


Figure 8 : Continuity Test Circuit

6.2 Simulated Continuity Test Circuit

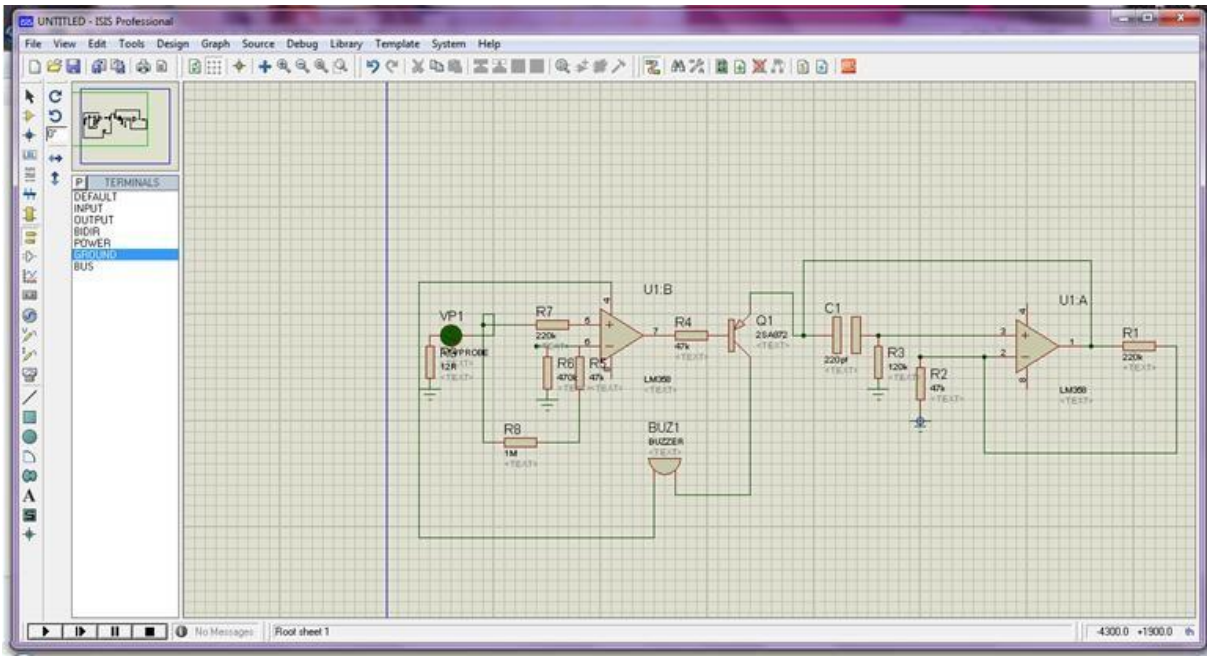


Figure 9 : Simulated Continuity Test Circuit by Proteus Software

CHAPTER 7

DC Voltage Measurement Circuit

DC Voltage Measurement Circuit

Instruments for the measurement of electric voltage are called voltmeters. Correct insertion of a voltmeter requires the connection of its terminals to the points of an electric circuit across which the voltage has to be measured. To a first approximation, the electric equivalent circuit of a voltmeter can be represented by a resistive impedance Z. This means that any voltmeter, once connected to an electric circuit, draws a current I given by:

$$= \frac{U}{Z}$$

Where U is the measured voltage. The higher the value of the international impedance, the higher the quantity of the voltmeter, since it does not significantly modify the status of the electric circuit under rest.

Different operating principles are used to measure an electric voltage. The mechanical interaction between currents, between a current and a magnetic field, or between electrified conductors was widely adopted in the past to generate a mechanical torque proportional to the voltage or the squared voltage to be measured. This torque, balanced by a restarting torque, usually generated by a spring, causes the instrument pointer.

Basically there are two types of voltmeter. One is analog voltmeter and another one is digital type voltmeter. The analog type voltmeter generally uses for measuring AC voltage. It displays the reading through the pointer which is fixed on the calibrated scale. The deflection of the t pointer depends on the torque acting on it. The digital voltmeter which displays the reading in the numeric form is known as the digital voltmeter. The digital voltmeter gives the accurate result.

In this project we have tried to measure the DC voltage by using voltage divider rule circuit. We have made a DC measurement instrument with a measuring range of 0V to 50V. We have made a circuit whose output is always in between 5V. Then we have feeded it to the microcontroller and showed the result on a LCD display.

7.1 Circuit Diagram of DC Measurement Circuit

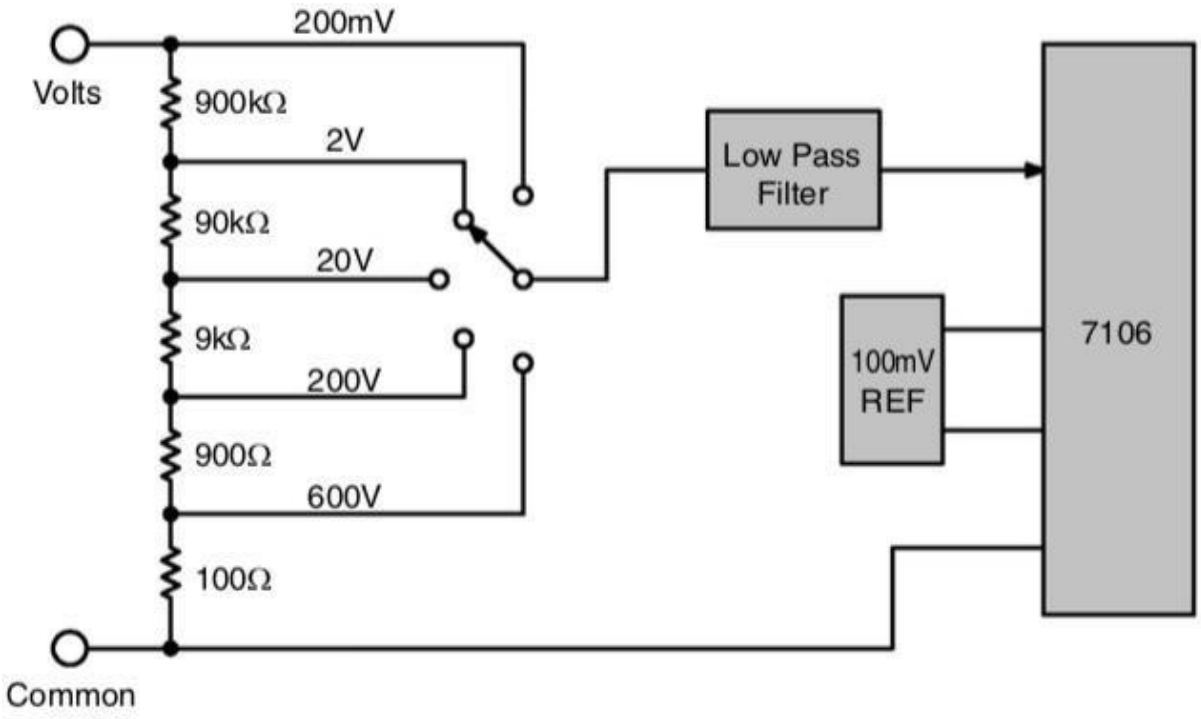


Figure 10: DC Voltage Measurement Circuit

7.2 Simulated Circuit Diagram

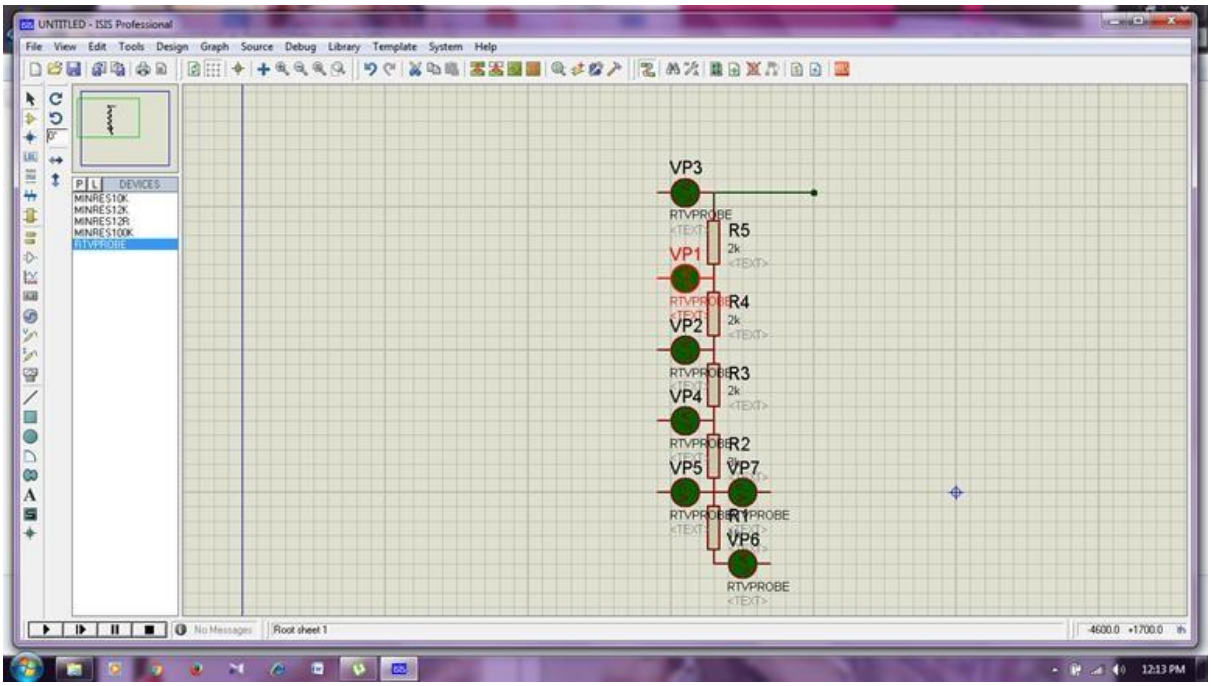


Figure 11: Simulated DC Voltage Measurement Circuit

CHAPTER 8

AC Voltage Measurement Circuit

AC Voltage Measurement

An alternating function or AC Waveform on the other hand is defined as one that varies in both magnitude and direction in more or less an even manner with respect to time making it a "Bi-directional" waveform. An AC function can represent either a power source or a signal source with the shape of an *AC waveform* generally following that of a mathematical sinusoid being defined as: $A(t) = A_{max}*\sin(2\pi ft)$.

The term AC or to give it its full description of Alternating Current, generally refers to a time-varying waveform with the most common of all being called a Sinusoid better known as a Sinusoidal Waveform. Sinusoidal waveforms are more generally called by their short description as Sine Waves. Sine waves are by far one of the most important types of AC waveform used in electrical engineering.

The shape obtained by plotting the instantaneous ordinate values of either voltage or current against time is called an AC Waveform. An AC waveform is constantly changing its polarity every half cycle alternating between a positive maximum value and a negative maximum value respectively with regards to time with a common example of this being the domestic mains voltage supply we use in our homes.

This means then that the *AC Waveform* is a "time-dependent signal" with the most common type of time-dependant signal being that of the Periodic Waveform. The periodic or AC waveform is the resulting product of a rotating electrical generator. Generally, the shape of any periodic waveform can be generated using a fundamental frequency and superimposing it with harmonic signals of varying frequencies and amplitudes but that's for another tutorial.

Alternating voltages and currents cannot be stored in batteries or cells like direct current (DC) can, it is much easier and cheaper to generate these quantities using alternators or waveform generators when they are needed. The type and shape of an AC waveform depends upon the generator or device producing them, but all AC waveforms consist of a zero voltage line that divides the waveform into two symmetrical halves. The main characteristics of an AC Waveform are defined as:

AC Waveform Characteristics

- • The Period, (T) is the length of time in seconds that the waveform takes to repeat itself from start to finish. This can also be called the *Periodic Time* of the waveform for sine waves, or the *Pulse Width* for square waves.
- • The Frequency, (f) is the number of times the waveform repeats itself within a one second time period. Frequency is the reciprocal of the time period, ($f = 1/T$) with the unit of frequency being the *Hertz*, (Hz).
- • The Amplitude (A) is the magnitude or intensity of the signal waveform measured in volts or amps.

8.1 Circuit Diagram of AC Voltage Measurement Circuit

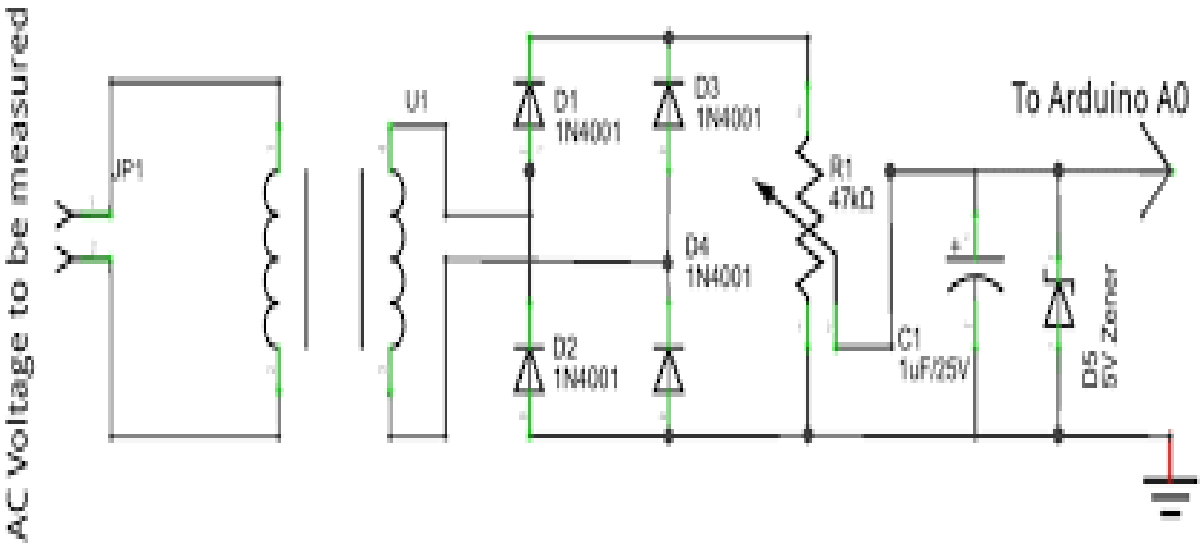


Figure 12: Circuit Diagram of AC measuring Circuit

8.2 Simulated Circuit Diagram of AC Voltage Measurement Circuit

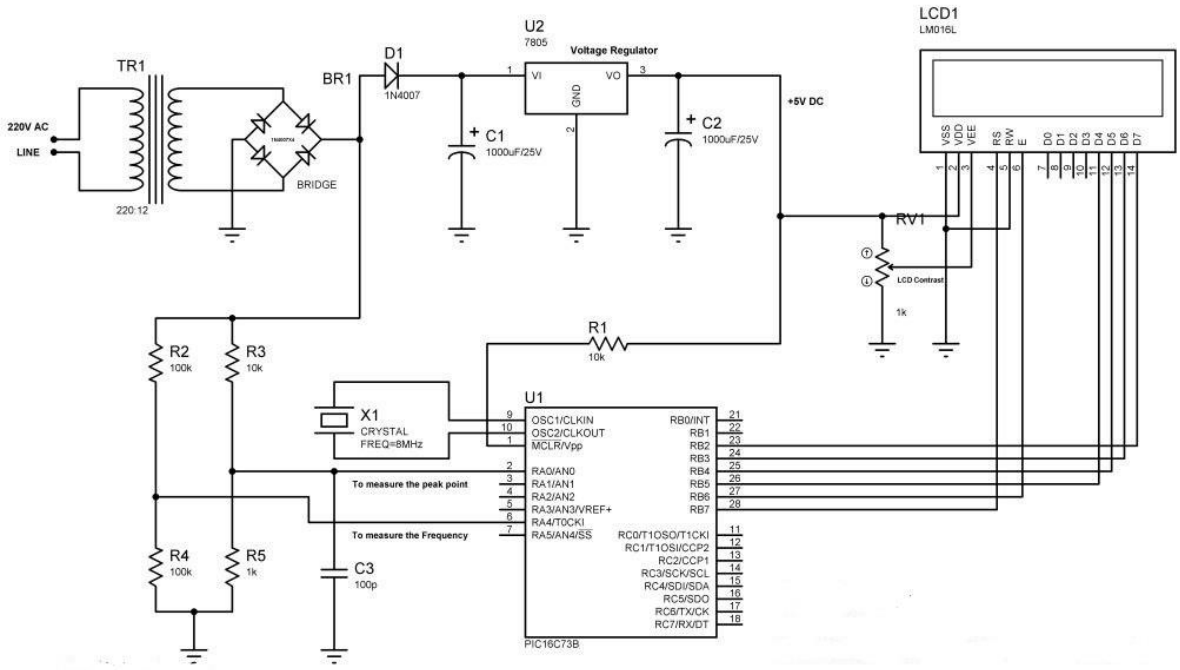


Figure 13: AC Voltage Measurement Circuit

CHAPTER 9

Hardware Implementation

Hardware Implementation

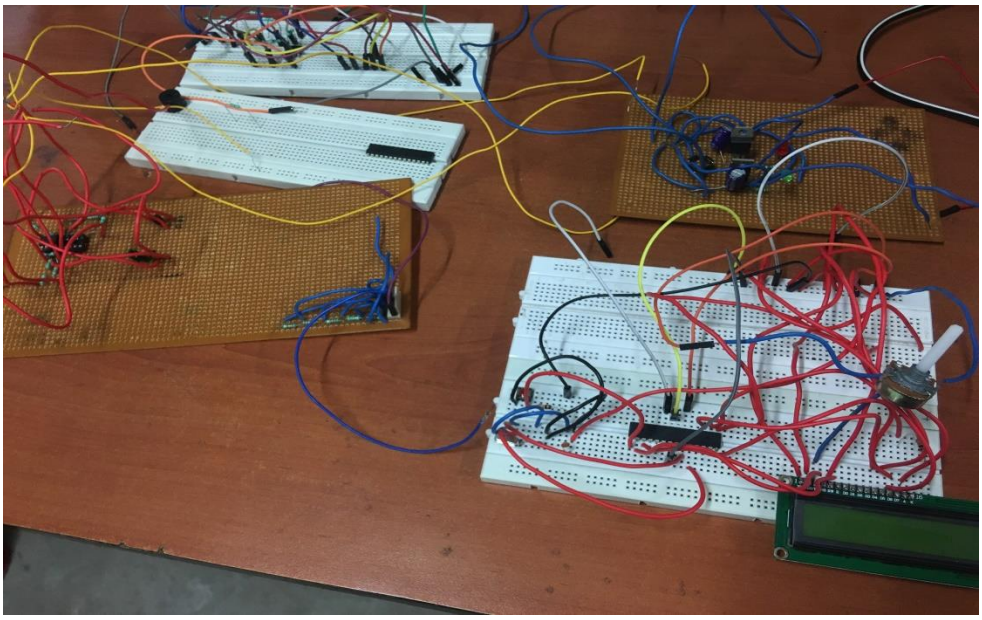


Figure14: Hardware Implementation

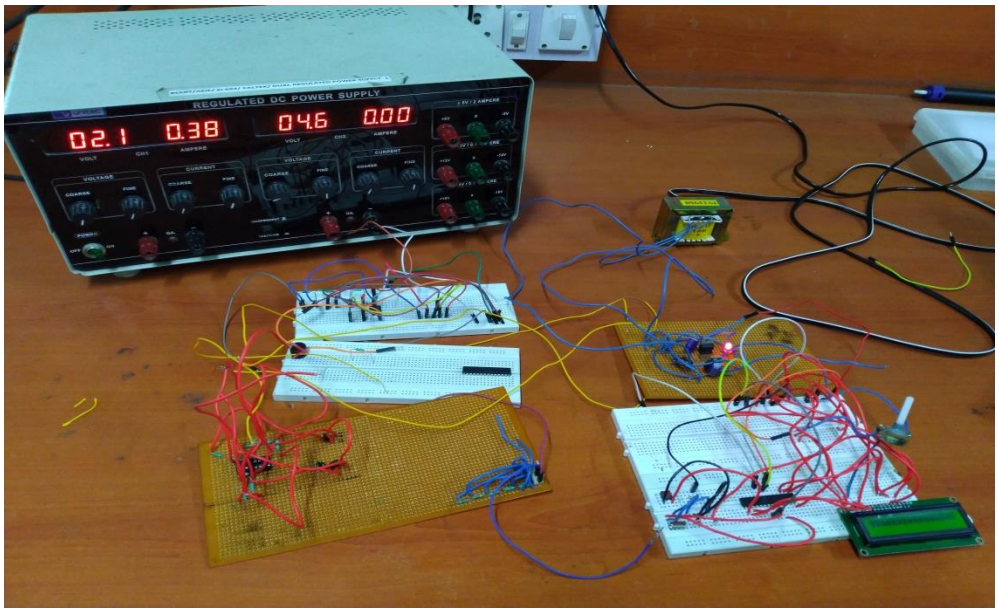


Figure 15: Hardware implementation of Digital multimeter

CHAPTER 10

Result

Result

Testing Phase:

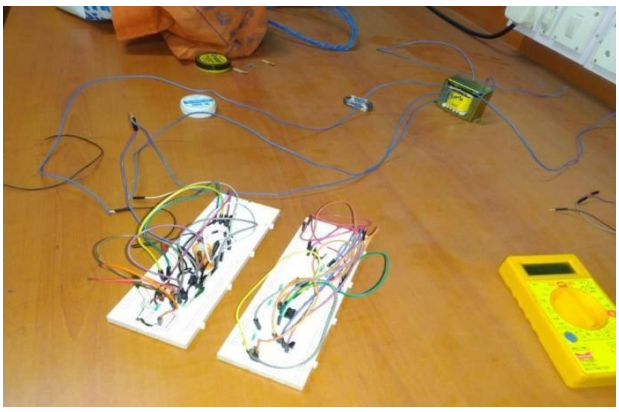


Figure 16: Continuity Check Circuit



Figure 17: DC Voltage Check Circuit Result 1



Figure 18: DC Voltage Check Circuit Result 2



Figure 19: DC Voltage Check Circuit Result 3



Figure 20: DC Voltage Check Circuit Result 4

Final Circuit

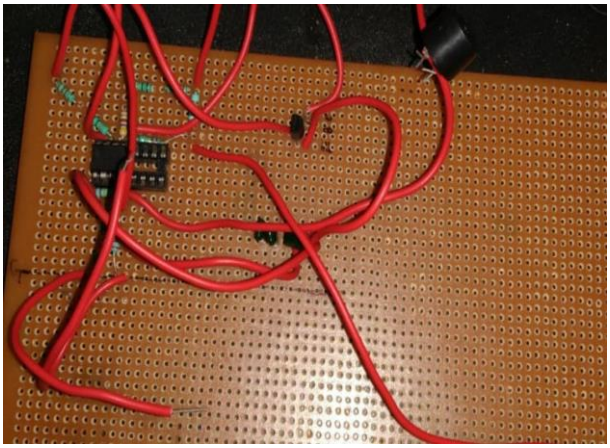


Figure 21:Continuity Test Circuit

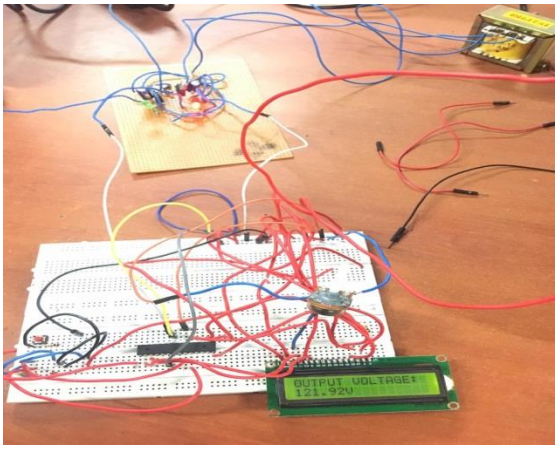


Figure 22: AC Voltage Measurement Circuit

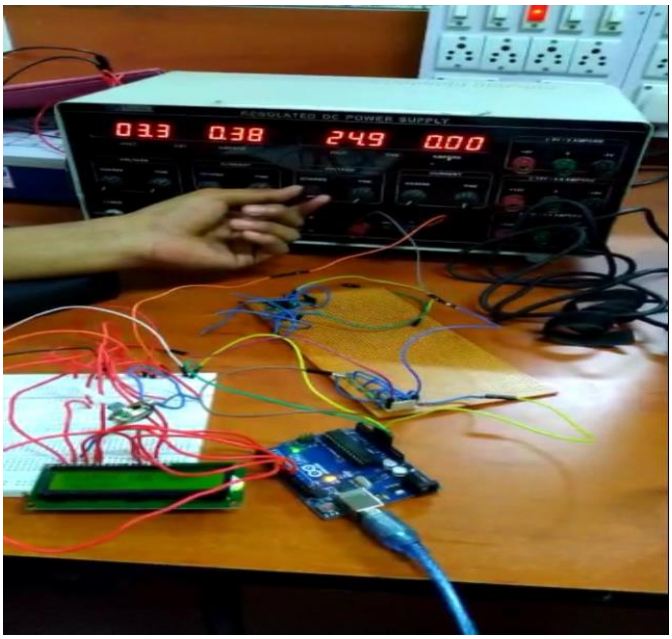


Figure 23: DC Voltage Measurement Circuit

CHAPTER 11

Conclusion

Conclusion

We have made the power supply circuit of +5v,- 5v and +12 v and -12v and we have designed the continuity circuit. We successfully complete our continuity test. Using LM358 we have made continuity test circuit for the multimeter to check the fault or continuity of a circuit. We have made a DC voltage measurement circuit and AC voltage measurement circuit to measure the required voltage. We have done this circuit on the vero board and we have used a LCD display to show this values.

CHAPTER 12

References

References

1. P. Petrovic, S. Marjanovic, and M. Stevanovic, "Digital method for power frequency measurement using synchronous sampling," Proc. Inst. Elect. Eng., vol. 146, no. 4, pp. 383–390,July 1999

2. E.R. Hnatek ,AUser'sHandbookofD/AandA/DConverters. New York: Wiley,1976

3. J. XiandJ. F. Chicharo, "Anew algorithmforimprovingthe accuracy of periodic signal analysis," IEEE Trans. Instrum. Meas., vol. 45, pp. 827–831, Aug. 1996.

4. L. Toivonen and J. Morsky, "Digital multirate algorithms for measurement of voltage, current, power and flicker," IEEE Trans. Power Delivery, vol. 10, Jan. 1995

5. P. Petrovic, S. Marjanovic, and M. Stevanovic, "Measuring of slowly changing AC signals without sample and hold circuit," IEEETrans.Instrum.Meas.,vol.49,pp.1245–1248,Dec.2000

6. H. Germer, "High-precision AC measurements using the Monte Carlo method," IEEE Trans. Instrum. Meas., vol. 50, pp. 457–460, Apr. 2001

7. B. C. Waltrip and N. M. Oldham, "Wideband wattmeter based on RMS voltage measurements," IEEE Trans. Instrum. Meas., vol. 46, pp. 781–783, Aug. 1997

8. T.M.Sounders,B.C.Waltrip,and O.B. Laug,"Awideband samplingvoltmeter,"IEEE Trans. Instrum. Meas., vol. 46,pp. 947–953, Aug. 1997

APPENDIX A

Flow Chart of Programs

Program of AC voltage measurement

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins

LiquidCrystal lcd(0, 1, 8, 9, 10, 11); /// REGISTER SELECT PIN,ENABLE PIN,D4 PIN,D5 PIN, D6 PIN, D7 PIN

int analogPin = A0;

int analogValue = 0;

void setup()

{

// set up the LCD's number of columns and rows:

lcd.begin(16, 2);

}

void loop()

{

// set the cursor to column 0, line 1

lcd.setCursor(0, 0);

lcd.print("OUTPUT VOLTAGE: ");//print name

lcd.setCursor(0, 1); // set the cursor to column 0, line 2

analogValue = analogRead(analogPin);

analogValue = analogValue * 55;

lcd.print(analogValue);

lcd.print("mV");

delay(500);//delay of 0.50sec

lcd.scrollDisplayLeft();//shifting data on LCD

lcd.setCursor(0, 0);// set the cursor to column 0, line1

}
```

APPENDIX B

Datasheet

Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash progam memory (ATmega48P/88P/168P/328P)
  - 256/512/512/1K Bytes EEPROM (ATmega48P/88P/168P/328P)
  - 512/1K/1K/2K Bytes Internal SRAM (ATmega48P/88P/168P/328P)
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
    Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
    Temperature Measurement
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Byte-oriented 2-wire Serial Interface (Philips I^2C compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
  - 1.8 - 5.5V for ATmega48P/88P/168PV
  - 2.7 - 5.5V for ATmega48P/88P/168P
  - 1.8 - 5.5V for ATmega328P
- Temperature Range:
  - -40°C to 85°C
- Speed Grade:
  - ATmega48P/88P/168PV: 0 - 4 MHz @ 1.8 - 5.5V, 0 - 10 MHz @ 2.7 - 5.5V
  - ATmega48P/88P/168P: 0 - 10 MHz @ 2.7 - 5.5V, 0 - 20 MHz @ 4.5 - 5.5V
  - ATmega328P: 0 - 4 MHz @ 1.8 - 5.5V, 0 - 10 MHz @ 2.7 - 5.5V, 0 - 20 MHz @ 4.5 - 5.5V
- Low Power Consumption at 1 MHz, 1.8V, 25°C for ATmega48P/88P/168P:
  - Active Mode: 0.3 mA
  - Power-down Mode: 0.1 µA
  - Power-save Mode: 0.8 µA (Including 32 kHz RTC)

Note: 1. See "Data Retention" on page 7 for details.

8-bit AVR® Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash

ATmega48P/V*
ATmega88P/V*
ATmega168P/V
ATmega328P**

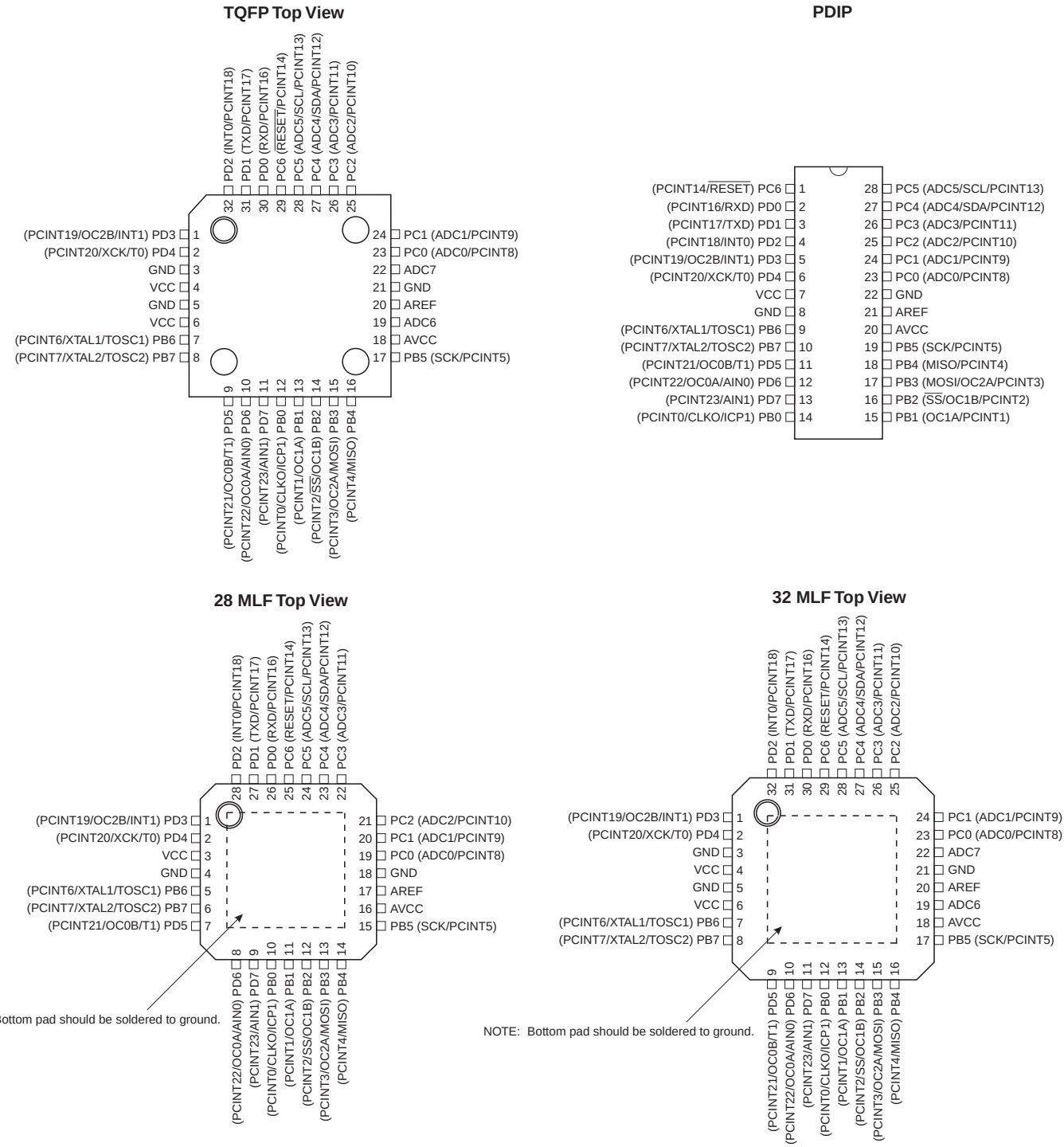
**Preliminary

* Not recommended for new designs.

Rev. 8025I–AVR–02/09

1. Pin Configurations

Figure 1-1. Pinout ATmega48P/88P/168P/328P



1.1 Pin Descriptions

1.1.1 VCC

Digital supply voltage.

1.1.2 GND

Ground.

1.1.3 Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7..6 is used as TOSC2..1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

The various special features of Port B are elaborated in ”Alternate Functions of Port B” on page 82 and ”System Clock and Clock Options” on page 26.

1.1.4 Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5..0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

1.1.5 PC6/$\overline{\text{RESET}}$

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. The minimum pulse length is given in Table 26-3 on page 320. Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated in ”Alternate Functions of Port C” on page 85.

1.1.6 Port D (PD7:0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

The various special features of Port D are elaborated in ”Alternate Functions of Port D” on page 88.

1.1.7 AV_{CC}

AV_{CC} is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter. Note that PC6..4 use digital supply voltage, V_{CC}.

1.1.8 AREF

AREF is the analog reference pin for the A/D Converter.

1.1.9 ADC7:6 (TQFP and QFN/MLF Package Only)

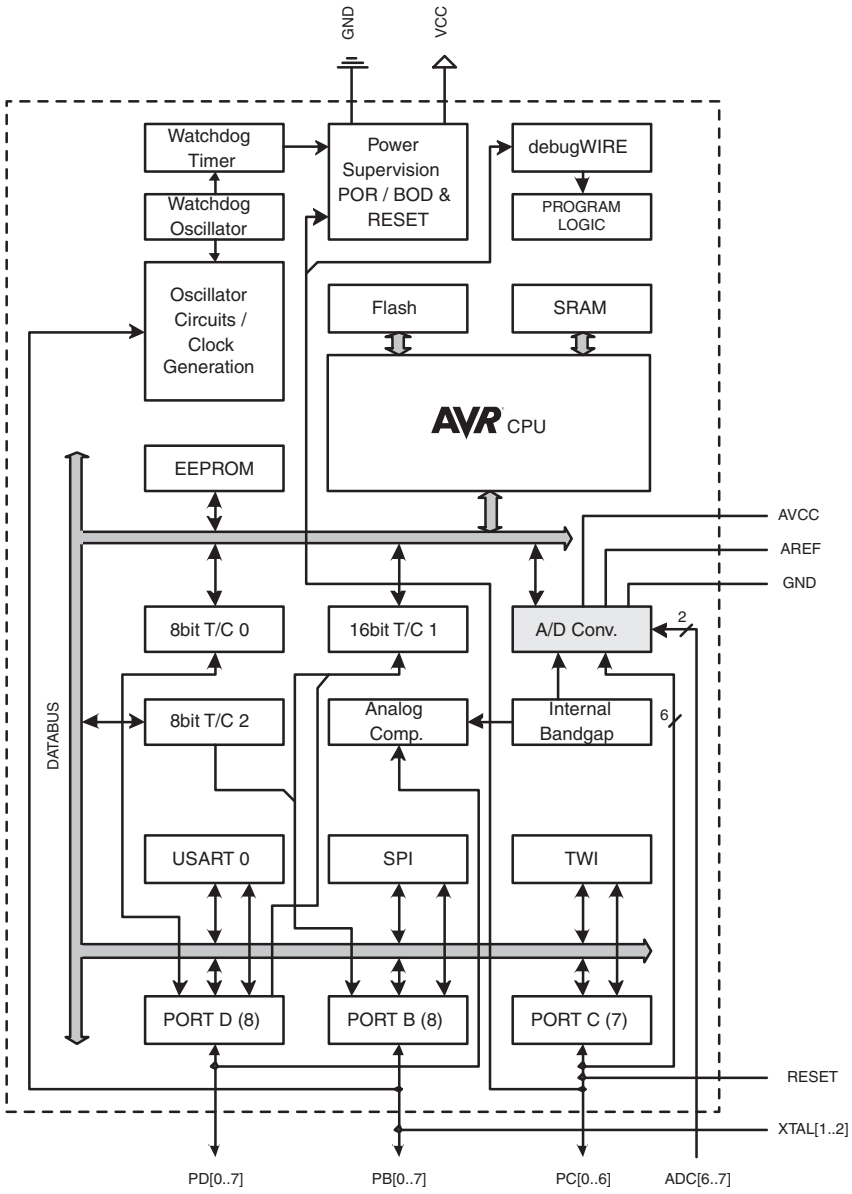
In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

2. Overview

The ATmega48P/88P/168P/328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48P/88P/168P/328P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega48P/88P/168P/328P provides the following features: 4K/8K/16K/32K bytes of In-System Programmable Flash with Read-While-Write capabilities, 256/512/512/1K bytes EEPROM, 512/1K/1K/2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable

Watchdog Timer with internal Oscillator, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega48P/88P/168P/328P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega48P/88P/168P/328P AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

2.2 Comparison Between ATmega48P, ATmega88P, ATmega168P, and ATmega328P

The ATmega48P, ATmega88P, ATmega168P, and ATmega328P differ only in memory sizes, boot loader support, and interrupt vector sizes. Table 2-1 summarizes the different memory and interrupt vector sizes for the three devices.

Table 2-1. Memory Size Summary

Device	Flash	EEPROM	RAM	Interrupt Vector Size
ATmega48P	4K Bytes	256 Bytes	512 Bytes	1 instruction word/vector
ATmega88P	8K Bytes	512 Bytes	1K Bytes	1 instruction word/vector
ATmega168P	16K Bytes	512 Bytes	1K Bytes	2 instruction words/vector
ATmega328P	32K Bytes	1K Bytes	2K Bytes	2 instructions words/vector

ATmega88P, ATmega168P, and ATmega328P support a real Read-While-Write Self-Programming mechanism. There is a separate Boot Loader Section, and the SPM instruction can only execute from there. In ATmega48P, there is no Read-While-Write support and no separate Boot Loader Section. The SPM instruction can execute from the entire Flash.

3. About

3.1 Disclaimer

Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

3.2 Resources

A comprehensive set of development tools, application notes and datasheets are available for download on http://www.atmel.com/avr.

3.3 Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C.

3.4 Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

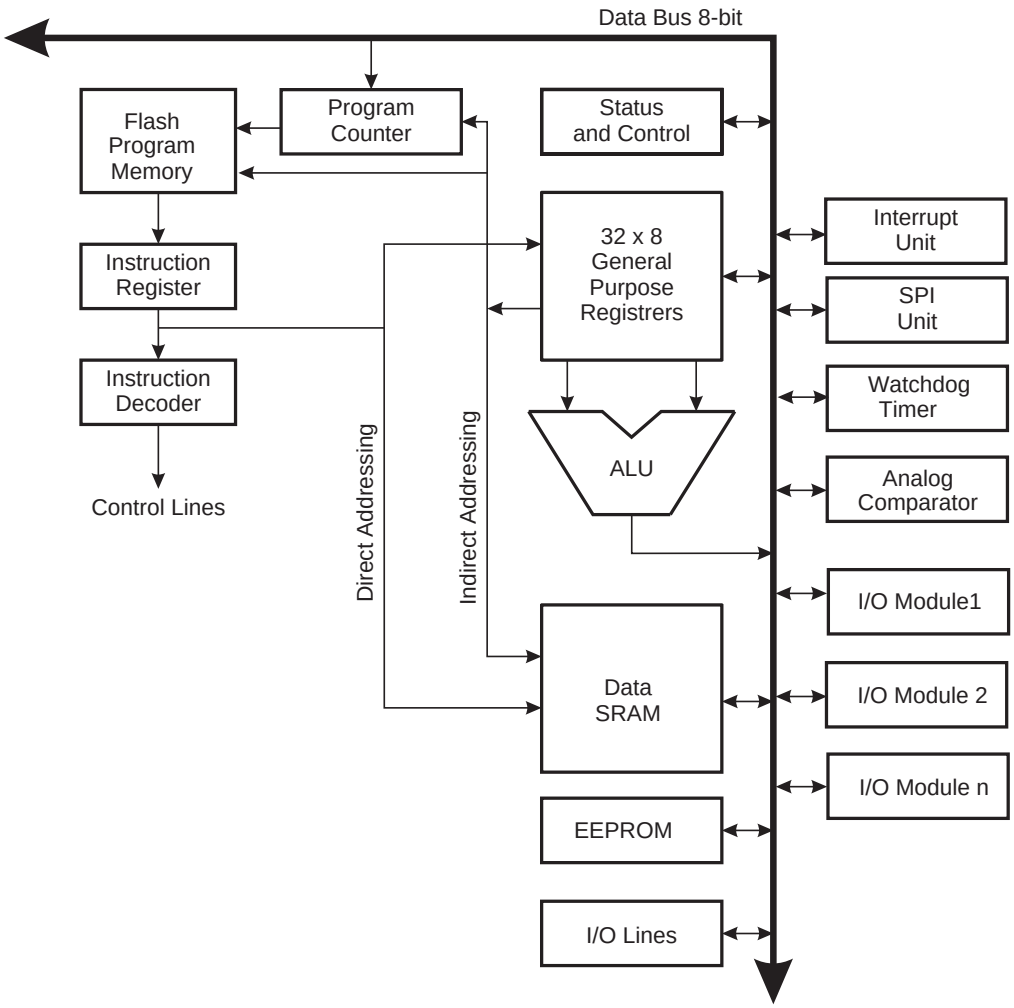
For I/O Registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

4. AVR CPU Core

4.1 Overview

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

Figure 4-1. Block Diagram of the AVR Architecture



In order to maximize performance and parallelism, the AVR uses a Harvard architecture – with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory.

The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typ-

ical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

Program Flash memory space is divided in two sections, the Boot Program section and the Application Program section. Both sections have dedicated Lock bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot Program section.

During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the Reset routine (before subroutines or interrupts are executed). The Stack Pointer (SP) is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional Global Interrupt Enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the Register File, 0x20 - 0x5F. In addition, the ATmega48P/88P/168P/328P has Extended I/O space from 0x60 - 0xFF in SRAM where only the ST/STS/STD and LD/LDS/LDD instructions can be used.

4.2 ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the “Instruction Set” section for a detailed description.

4.3 Status Register

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the Status Register is updated after all ALU operations, as

specified in the Instruction Set Reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code.

The Status Register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

4.3.1 SREG – AVR Status Register

The AVR Status Register – SREG – is defined as:

Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – I: Global Interrupt Enable**

The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the Global Interrupt Enable Register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the instruction set reference.

- **Bit 6 – T: Bit Copy Storage**

The Bit Copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T-bit as source or destination for the operated bit. A bit from a register in the Register File can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the Register File by the BLD instruction.

- **Bit 5 – H: Half Carry Flag**

The Half Carry Flag H indicates a Half Carry in some arithmetic operations. Half Carry Is useful in BCD arithmetic. See the “Instruction Set Description” for detailed information.

- **Bit 4 – S: Sign Bit, S = N ⊕ V**

The S-bit is always an exclusive or between the Negative Flag N and the Two’s Complement Overflow Flag V. See the “Instruction Set Description” for detailed information.

- **Bit 3 – V: Two’s Complement Overflow Flag**

The Two’s Complement Overflow Flag V supports two’s complement arithmetics. See the “Instruction Set Description” for detailed information.

- **Bit 2 – N: Negative Flag**

The Negative Flag N indicates a negative result in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

- **Bit 1 – Z: Zero Flag**

The Zero Flag Z indicates a zero result in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

- **Bit 0 – C: Carry Flag**

The Carry Flag C indicates a carry in an arithmetic or logic operation. See the “Instruction Set Description” for detailed information.

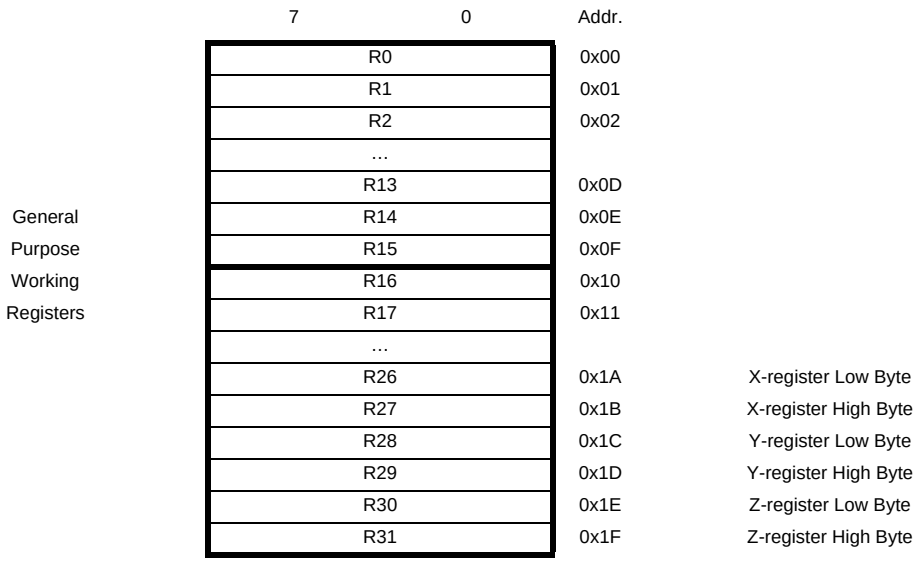
4.4 General Purpose Register File

The Register File is optimized for the AVR Enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the Register File:

- One 8-bit output operand and one 8-bit result input
- Two 8-bit output operands and one 8-bit result input
- Two 8-bit output operands and one 16-bit result input
- One 16-bit output operand and one 16-bit result input

Figure 4-2 shows the structure of the 32 general purpose working registers in the CPU.

Figure 4-2. AVR CPU General Purpose Working Registers



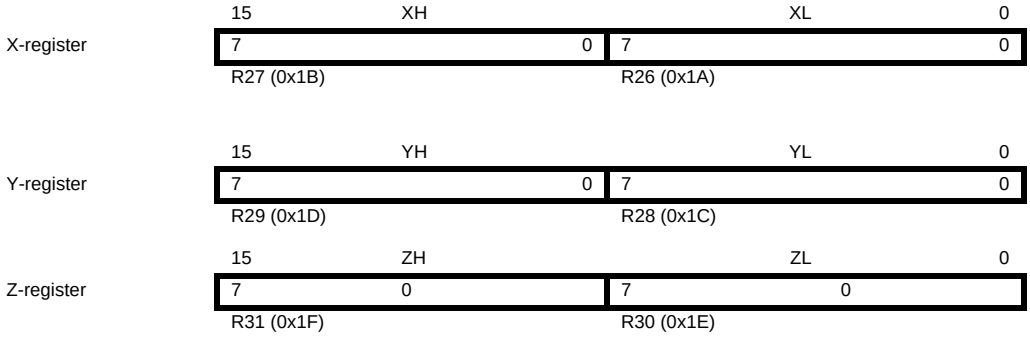
Most of the instructions operating on the Register File have direct access to all registers, and most of them are single cycle instructions.

As shown in Figure 4-2, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y- and Z-pointer registers can be set to index any register in the file.

4.4.1 The X-register, Y-register, and Z-register

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the data space. The three indirect address registers X, Y, and Z are defined as described in Figure 4-3.

Figure 4-3. The X-, Y-, and Z-registers



In the different addressing modes these address registers have functions as fixed displacement, automatic increment, and automatic decrement (see the instruction set reference for details).

4.5 Stack Pointer

The Stack is mainly used for storing temporary data, for storing local variables and for storing return addresses after interrupts and subroutine calls. Note that the Stack is implemented as growing from higher to lower memory locations. The Stack Pointer Register always points to the top of the Stack. The Stack Pointer points to the data SRAM Stack area where the Subroutine and Interrupt Stacks are located. A Stack PUSH command will decrease the Stack Pointer.

The Stack in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. Initial Stack Pointer value equals the last address of the internal SRAM and the Stack Pointer must be set to point above start of the SRAM, see Figure 5-3 on page 18.

See Table 4-1 for Stack Pointer details.

Table 4-1. Stack Pointer instructions

Instruction	Stack pointer	Description
PUSH	Decremented by 1	Data is pushed onto the stack
CALL ICALL RCALL	Decremented by 2	Return address is pushed onto the stack with a subroutine call or interrupt
POP	Incremented by 1	Data is popped from the stack
RET RETI	Incremented by 2	Return address is popped from the stack with return from subroutine or return from interrupt

The AVR Stack Pointer is implemented as two 8-bit registers in the I/O space. The number of bits actually used is implementation dependent. Note that the data space in some implementations of the AVR architecture is so small that only SPL is needed. In this case, the SPH Register will not be present.

4.5.1 SPH and SPL – Stack Pointer High and Stack Pointer Low Register

Bit	15	14	13	12	11	10	9	8	
0x3E (0x5E)	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
0x3D (0x5D)	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	
	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	RAMEND	

4.6 Instruction Execution Timing

This section describes the general access timing concepts for instruction execution. The AVR CPU is driven by the CPU clock clk_{CPU}, directly generated from the selected clock source for the chip. No internal clock division is used.

Figure 4-4 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access Register File concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 4-4. The Parallel Instruction Fetches and Instruction Executions

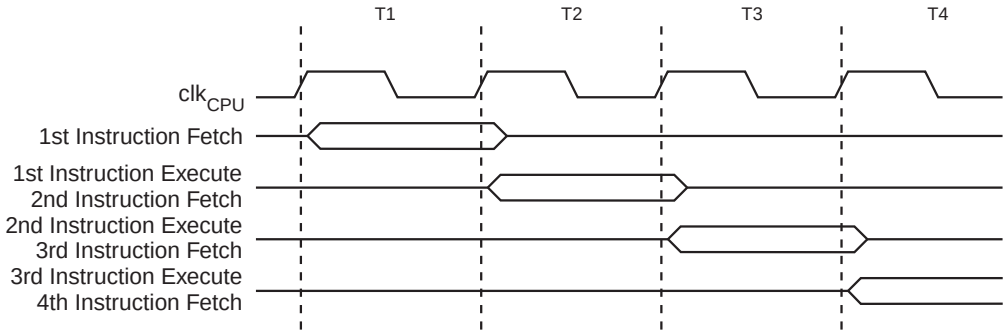
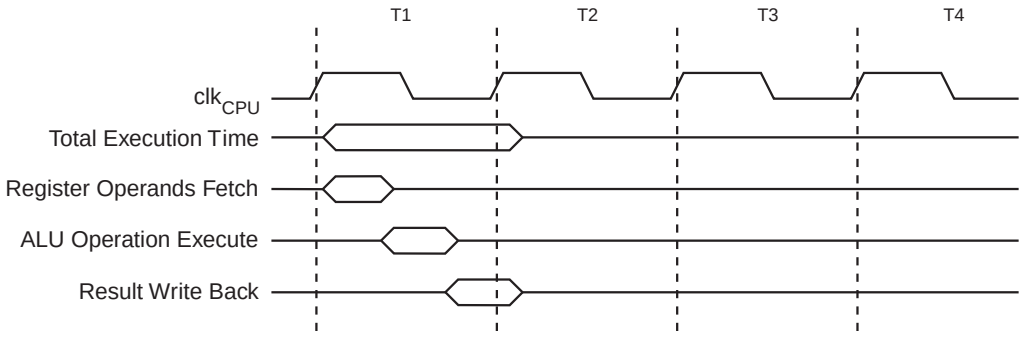


Figure 4-5 shows the internal timing concept for the Register File. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

Figure 4-5. Single Cycle ALU Operation



4.7 Reset and Interrupt Handling

The AVR provides several different interrupt sources. These interrupts and the separate Reset Vector each have a separate program vector in the program memory space. All interrupts are assigned individual enable bits which must be written logic one together with the Global Interrupt Enable bit in the Status Register in order to enable the interrupt. Depending on the Program Counter value, interrupts may be automatically disabled when Boot Lock bits BLB02 or BLB12 are programmed. This feature improves software security. See the section ”Memory Programming” on page 294 for details.

The lowest addresses in the program memory space are by default defined as the Reset and Interrupt Vectors. The complete list of vectors is shown in ”Interrupts” on page 57. The list also determines the priority levels of the different interrupts. The lower the address the higher is the priority level. RESET has the highest priority, and next is INT0 – the External Interrupt Request 0. The Interrupt Vectors can be moved to the start of the Boot Flash section by setting the IVSEL bit in the MCU Control Register (MCUCR). Refer to ”Interrupts” on page 57 for more information. The Reset Vector can also be moved to the start of the Boot Flash section by programming the BOOTRST Fuse, see ”Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P” on page 277.

When an interrupt occurs, the Global Interrupt Enable I-bit is cleared and all interrupts are disabled. The user software can write logic one to the I-bit to enable nested interrupts. All enabled interrupts can then interrupt the current interrupt routine. The I-bit is automatically set when a Return from Interrupt instruction – RETI – is executed.

There are basically two types of interrupts. The first type is triggered by an event that sets the Interrupt Flag. For these interrupts, the Program Counter is vectored to the actual Interrupt Vector in order to execute the interrupt handling routine, and hardware clears the corresponding Interrupt Flag. Interrupt Flags can also be cleared by writing a logic one to the flag bit position(s) to be cleared. If an interrupt condition occurs while the corresponding interrupt enable bit is cleared, the Interrupt Flag will be set and remembered until the interrupt is enabled, or the flag is cleared by software. Similarly, if one or more interrupt conditions occur while the Global Interrupt Enable bit is cleared, the corresponding Interrupt Flag(s) will be set and remembered until the Global Interrupt Enable bit is set, and will then be executed by order of priority.

The second type of interrupts will trigger as long as the interrupt condition is present. These interrupts do not necessarily have Interrupt Flags. If the interrupt condition disappears before the interrupt is enabled, the interrupt will not be triggered.

When the AVR exits from an interrupt, it will always return to the main program and execute one more instruction before any pending interrupt is served.

Note that the Status Register is not automatically stored when entering an interrupt routine, nor restored when returning from an interrupt routine. This must be handled by software.

When using the CLI instruction to disable interrupts, the interrupts will be immediately disabled. No interrupt will be executed after the CLI instruction, even if it occurs simultaneously with the CLI instruction. The following example shows how this can be used to avoid interrupts during the timed EEPROM write sequence.

Assembly Code Example

```
in  r16, SREG     ; store SREG value
cli    ; disable interrupts during timed sequence
sbi EECR, EEMPE   ; start EEPROM write
sbi EECR, EEPE
out SREG, r16     ; restore SREG value (I-bit)
```

C Code Example

```
char cSREG;
cSREG = SREG; /* store SREG value */
/* disable interrupts during timed sequence */
_CLI();
EECR |= (1<<EEMPE); /* start EEPROM write */
EECR |= (1<<EEPE);
SREG = cSREG; /* restore SREG value (I-bit) */
```

When using the SEI instruction to enable interrupts, the instruction following SEI will be executed before any pending interrupts, as shown in this example.

Assembly Code Example

```
sei  ; set Global Interrupt Enable
sleep; enter sleep, waiting for interrupt
; note: will enter sleep before any pending interrupt(s)
```

C Code Example

```
__enable_interrupt(); /* set Global Interrupt Enable */
__sleep(); /* enter sleep, waiting for interrupt */
/* note: will enter sleep before any pending interrupt(s) */
```

4.7.1 Interrupt Response Time

The interrupt execution response for all the enabled AVR interrupts is four clock cycles minimum. After four clock cycles the program vector address for the actual interrupt handling routine is executed. During this four clock cycle period, the Program Counter is pushed onto the Stack. The vector is normally a jump to the interrupt routine, and this jump takes three clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served. If an interrupt occurs when the MCU is in sleep mode, the interrupt execution response time is increased by four clock cycles. This increase comes in addition to the start-up time from the selected sleep mode.

A return from an interrupt handling routine takes four clock cycles. During these four clock cycles, the Program Counter (two bytes) is popped back from the Stack, the Stack Pointer is incremented by two, and the I-bit in SREG is set.

5. AVR Memories

5.1 Overview

This section describes the different memories in the ATmega48P/88P/168P/328P. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, the ATmega48P/88P/168P/328P features an EEPROM Memory for data storage. All three memory spaces are linear and regular.

5.2 In-System Reprogrammable Flash Program Memory

The ATmega48P/88P/168P/328P contains 4/8/16/32K bytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 2/4/8/16K x 16. For software security, the Flash Program memory space is divided into two sections, Boot Loader Section and Application Program Section in ATmega88P and ATmega168P. ATmega48P does not have separate Boot Loader and Application Program sections, and the SPM instruction can be executed from the entire Flash. See SELFPRGEN description in section ”SPMCSR – Store Program Memory Control and Status Register” on page 275 and page 292for more details.

The Flash memory has an endurance of at least 10,000 write/erase cycles. The ATmega48P/88P/168P/328P Program Counter (PC) is 11/12/13/14 bits wide, thus addressing the 2/4/8/16K program memory locations. The operation of Boot Program section and associated Boot Lock bits for software protection are described in detail in ”Self-Programming the Flash, ATmega48P” on page 269 and ”Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P” on page 277. ”Memory Programming” on page 294 contains a detailed description on Flash Programming in SPI- or Parallel Programming mode.

Constant tables can be allocated within the entire program memory address space (see the LPM – Load Program Memory instruction description).

Timing diagrams for instruction fetch and execution are presented in ”Instruction Execution Timing” on page 13.

Figure 5-1. Program Memory Map, ATmega48P

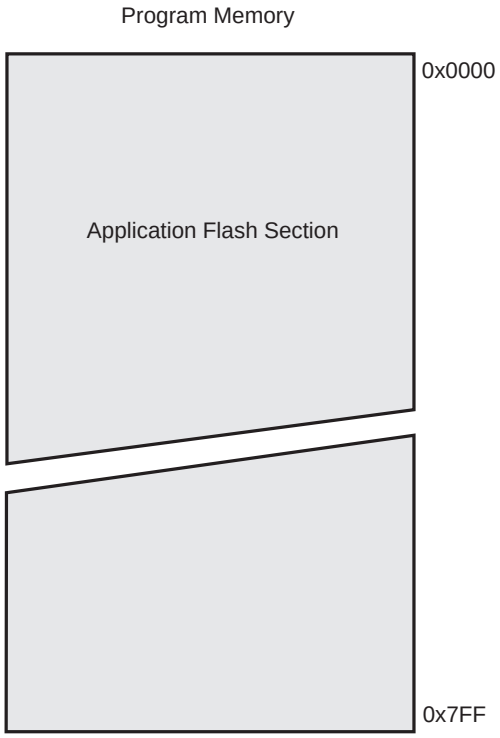
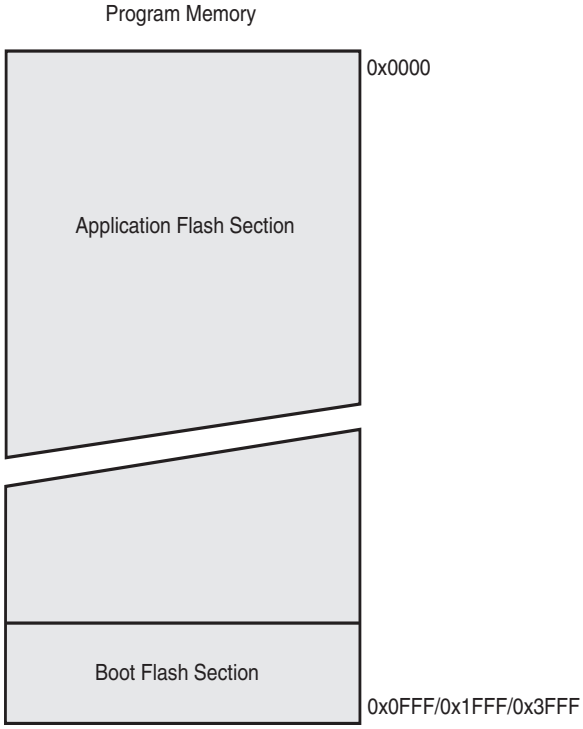


Figure 5-2. Program Memory Map, ATmega88P, ATmega168P, and ATmega328P



5.3 SRAM Data Memory

Figure 5-3 shows how the ATmega48P/88P/168P/328P SRAM Memory is organized.

The ATmega48P/88P/168P/328P is a complex microcontroller with more peripheral units than can be supported within the 64 locations reserved in the Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

The lower 768/1280/1280/2303 data memory locations address both the Register File, the I/O memory, Extended I/O memory, and the internal data SRAM. The first 32 locations address the Register File, the next 64 location the standard I/O memory, then 160 locations of Extended I/O memory, and the next 512/1024/1024/2048 locations address the internal data SRAM.

The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect, Indirect with Pre-decrement, and Indirect with Post-increment. In the Register File, registers R26 to R31 feature the indirect addressing pointer registers.

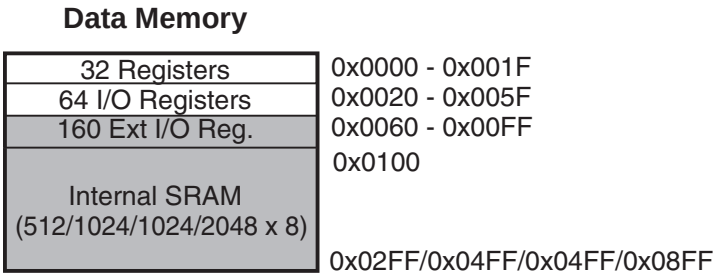
The direct addressing reaches the entire data space.

The Indirect with Displacement mode reaches 63 address locations from the base address given by the Y- or Z-register.

When using register indirect addressing modes with automatic pre-decrement and post-increment, the address registers X, Y, and Z are decremented or incremented.

The 32 general purpose working registers, 64 I/O Registers, 160 Extended I/O Registers, and the 512/1024/1024/2048 bytes of internal data SRAM in the ATmega48P/88P/168P/328P are all accessible through all these addressing modes. The Register File is described in ”General Purpose Register File” on page 11.

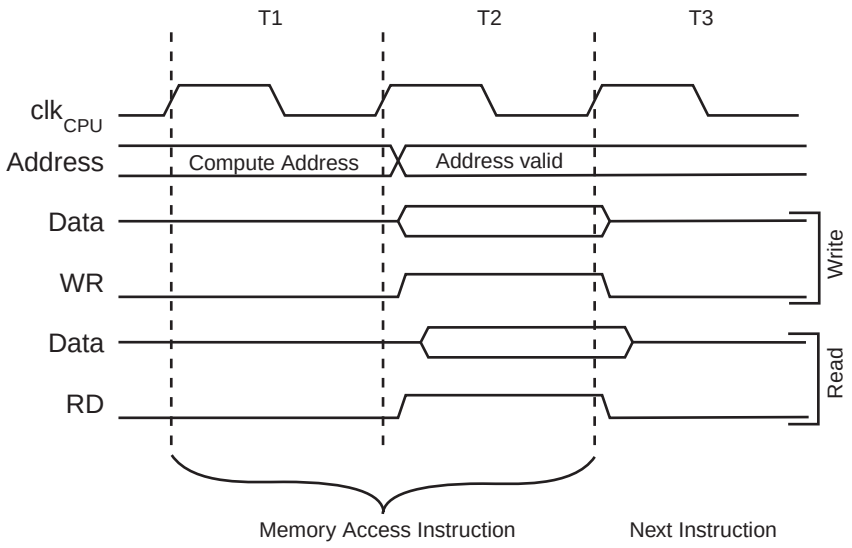
Figure 5-3. Data Memory Map



5.3.1 Data Memory Access Times

This section describes the general access timing concepts for internal memory access. The internal data SRAM access is performed in two clk_{CPU} cycles as described in Figure 5-4.

Figure 5-4. On-chip Data SRAM Access Cycles



5.4 EEPROM Data Memory

The ATmega48P/88P/168P/328P contains 256/512/512/1K bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described in the following, specifying the EEPROM Address Registers, the EEPROM Data Register, and the EEPROM Control Register.

"Memory Programming" on page 294 contains a detailed description on EEPROM Programming in SPI or Parallel Programming mode.

5.4.1 EEPROM Read/Write Access

The EEPROM Access Registers are accessible in the I/O space.

The write access time for the EEPROM is given in Table 5-2. A self-timing function, however, lets the user software detect when the next byte can be written. If the user code contains instructions that write the EEPROM, some precautions must be taken. In heavily filtered power supplies, V_{CC} is likely to rise or fall slowly on power-up/down. This causes the device for some period of time to run at a voltage lower than specified as minimum for the clock frequency used. See "Preventing EEPROM Corruption" on page 20 for details on how to avoid problems in these situations.

In order to prevent unintentional EEPROM writes, a specific write procedure must be followed. Refer to the description of the EEPROM Control Register for details on this.

When the EEPROM is read, the CPU is halted for four clock cycles before the next instruction is executed. When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed.

5.4.2 Preventing EEPROM Corruption

During periods of low $V_{CC,}$ the EEPROM data can be corrupted because the supply voltage is too low for the CPU and the EEPROM to operate properly. These issues are the same as for board level systems using EEPROM, and the same design solutions should be applied.

An EEPROM data corruption can be caused by two situations when the voltage is too low. First, a regular write sequence to the EEPROM requires a minimum voltage to operate correctly. Secondly, the CPU itself can execute instructions incorrectly, if the supply voltage is too low.

EEPROM data corruption can easily be avoided by following this design recommendation:

Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal Brown-out Detector (BOD). If the detection level of the internal BOD does not match the needed detection level, an external low V_{CC} reset Protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided that the power supply voltage is sufficient.

5.5 I/O Memory

The I/O space definition of the ATmega48P/88P/168P/328P is shown in ”Register Summary” on page 425.

All ATmega48P/88P/168P/328P I/Os and peripherals are placed in the I/O space. All I/O locations may be accessed by the LD/LDS/LDD and ST/STS/STD instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O Registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set section for more details. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these addresses. The ATmega48P/88P/168P/328P is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.

Some of the Status Flags are cleared by writing a logical one to them. Note that, unlike most other AVRs, the CBI and SBI instructions will only operate on the specified bit, and can therefore be used on registers containing such Status Flags. The CBI and SBI instructions work with registers 0x00 to 0x1F only.

The I/O and peripherals control registers are explained in later sections.

5.5.1 General Purpose I/O Registers

The ATmega48P/88P/168P/328P contains three General Purpose I/O Registers. These registers can be used for storing any information, and they are particularly useful for storing global variables and Status Flags. General Purpose I/O Registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI, CBI, SBIS, and SBIC instructions.

5.6 Register Description

5.6.1 EEARH and EEARL – The EEPROM Address Register

Bit	15	14	13	12	11	10	9	8	
0x22 (0x42)	–	–	–	–	–	–	–	EEAR8	EEARH
0x21 (0x41)	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	X	
	X	X	X	X	X	X	X	X	

- **Bits 15..9 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bits 8..0 – EEAR8..0: EEPROM Address**

The EEPROM Address Registers – EEARH and EEARL specify the EEPROM address in the 256/512/512/1K bytes EEPROM space. The EEPROM data bytes are addressed linearly between 0 and 255/511/511/1023. The initial value of EEAR is undefined. A proper value must be written before the EEPROM may be accessed.

EEAR8 is an unused bit in ATmega48P and must always be written to zero.

5.6.2 EEDR – The EEPROM Data Register

Bit	7	6	5	4	3	2	1	0	
0x20 (0x40)	MSB							LSB	EEDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..0 – EEDR7.0: EEPROM Data**

For the EEPROM write operation, the EEDR Register contains the data to be written to the EEPROM in the address given by the EEAR Register. For the EEPROM read operation, the EEDR contains the data read out from the EEPROM at the address given by EEAR.

5.6.3 EECR – The EEPROM Control Register

Bit	7	6	5	4	3	2	1	0	
0x1F (0x3F)	–	–	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE	EECR
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	X	X	0	0	X	0	

- **Bits 7..6 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bits 5, 4 – EEPM1 and EEPM0: EEPROM Programming Mode Bits**

The EEPROM Programming mode bit setting defines which programming action that will be triggered when writing EEPE. It is possible to program data in one atomic operation (erase the old value and program the new value) or to split the Erase and Write operations in two different operations. The Programming times for the different modes are shown in Table 5-1. While EEPE

is set, any write to EEPMn will be ignored. During reset, the EEPMn bits will be reset to 0b00 unless the EEPROM is busy programming.

Table 5-1. EEPROM Mode Bits

EEPM1	EEPM0	Programming Time	Operation
0	0	3.4 ms	Erase and Write in one operation (Atomic Operation)
0	1	1.8 ms	Erase Only
1	0	1.8 ms	Write Only
1	1	–	Reserved for future use

- **Bit 3 – EERIE: EEPROM Ready Interrupt Enable**

Writing EERIE to one enables the EEPROM Ready Interrupt if the I bit in SREG is set. Writing EERIE to zero disables the interrupt. The EEPROM Ready interrupt generates a constant interrupt when EEPE is cleared. The interrupt will not be generated during EEPROM write or SPM.

- **Bit 2 – EEMPE: EEPROM Master Write Enable**

The EEMPE bit determines whether setting EEPE to one causes the EEPROM to be written. When EEMPE is set, setting EEPE within four clock cycles will write data to the EEPROM at the selected address If EEMPE is zero, setting EEPE will have no effect. When EEMPE has been written to one by software, hardware clears the bit to zero after four clock cycles. See the description of the EEPE bit for an EEPROM write procedure.

- **Bit 1 – EEPE: EEPROM Write Enable**

The EEPROM Write Enable Signal EEPE is the write strobe to the EEPROM. When address and data are correctly set up, the EEPE bit must be written to one to write the value into the EEPROM. The EEMPE bit must be written to one before a logical one is written to EEPE, otherwise no EEPROM write takes place. The following procedure should be followed when writing the EEPROM (the order of steps 3 and 4 is not essential):

1. Wait until EEPE becomes zero.
2. Wait until SELFPRGEN in SPMCSR becomes zero.
3. Write new EEPROM address to EEAR (optional).
4. Write new EEPROM data to EEDR (optional).
5. Write a logical one to the EEMPE bit while writing a zero to EEPE in EECR.
6. Within four clock cycles after setting EEMPE, write a logical one to EEPE.

The EEPROM can not be programmed during a CPU write to the Flash memory. The software must check that the Flash programming is completed before initiating a new EEPROM write. Step 2 is only relevant if the software contains a Boot Loader allowing the CPU to program the Flash. If the Flash is never being updated by the CPU, step 2 can be omitted. See ”Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P” on page 277 for details about Boot programming.

Caution: An interrupt between step 5 and step 6 will make the write cycle fail, since the EEPROM Master Write Enable will time-out. If an interrupt routine accessing the EEPROM is interrupting another EEPROM access, the EEAR or EEDR Register will be modified, causing the interrupted EEPROM access to fail. It is recommended to have the Global Interrupt Flag cleared during all the steps to avoid these problems.

When the write access time has elapsed, the EEPE bit is cleared by hardware. The user software can poll this bit and wait for a zero before writing the next byte. When EEPE has been set, the CPU is halted for two cycles before the next instruction is executed.

- **Bit 0 – EERE: EEPROM Read Enable**

The EEPROM Read Enable Signal EERE is the read strobe to the EEPROM. When the correct address is set up in the EEAR Register, the EERE bit must be written to a logic one to trigger the EEPROM read. The EEPROM read access takes one instruction, and the requested data is available immediately. When the EEPROM is read, the CPU is halted for four cycles before the next instruction is executed.

The user should poll the EEPE bit before starting the read operation. If a write operation is in progress, it is neither possible to read the EEPROM, nor to change the EEAR Register.

The calibrated Oscillator is used to time the EEPROM accesses. Table 5-2 lists the typical programming time for EEPROM access from the CPU.

Table 5-2. EEPROM Programming Time

Symbol	Number of Calibrated RC Oscillator Cycles	Typ Programming Time
EEPROM write (from CPU)	26,368	3.3 ms

The following code examples show one assembly and one C function for writing to the EEPROM. The examples assume that interrupts are controlled (e.g. by disabling interrupts globally) so that no interrupts will occur during execution of these functions. The examples also assume that no Flash Boot Loader is present in the software. If such code is present, the EEPROM write function must also wait for any ongoing SPM command to finish.

Assembly Code Example

```
EEPROM_write:
  ; Wait for completion of previous write
  sbic EECR,EEPE
  rjmp EEPROM_write
  ; Set up address (r18:r17) in address register
  out  EEARH, r18
  out  EEARL, r17
  ; Write data (r16) to Data Register
  out  EEDR,r16
  ; Write logical one to EEMPE
  sbi  EECR,EEMPE
  ; Start eeprom write by setting EEPE
  sbi  EECR,EEPE
  ret
```

C Code Example

```
void EEPROM_write(unsigned int uiAddress, unsigned char ucData)
{
  /* Wait for completion of previous write */
  while(EECR & (1<<EEPE))
    ;
  /* Set up address and Data Registers */
  EEAR = uiAddress;
  EEDR = ucData;
  /* Write logical one to EEMPE */
  EECR |= (1<<EEMPE);
  /* Start eeprom write by setting EEPE */
  EECR |= (1<<EEPE);
}
```

The next code examples show assembly and C functions for reading the EEPROM. The examples assume that interrupts are controlled so that no interrupts will occur during execution of these functions.

Assembly Code Example

```
EEPROM_read:
  ; Wait for completion of previous write
  sbic EECR,EEPE
  rjmp EEPROM_read
  ; Set up address (r18:r17) in address register
  out  EEARH, r18
  out  EEARL, r17
  ; Start eeprom read by writing EERE
  sbi  EECR,EERE
  ; Read data from Data Register
  in   r16,EEDR
  ret
```

C Code Example

```
unsigned char EEPROM_read(unsigned int uiAddress)
{
  /* Wait for completion of previous write */
  while(EECR & (1<<EEPE))
    ;
  /* Set up address register */
  EEAR = uiAddress;
  /* Start eeprom read by writing EERE */
  EECR |= (1<<EERE);
  /* Return data from Data Register */
  return EEDR;
}
```

5.6.4 GPIOR2 – General Purpose I/O Register 2

Bit	7	6	5	4	3	2	1	0	
0x2B (0x4B)	MSB							LSB	GPIOR2
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

5.6.5 GPIOR1 – General Purpose I/O Register 1

Bit	7	6	5	4	3	2	1	0	
0x2A (0x4A)	MSB							LSB	GPIOR1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

5.6.6 GPIOR0 – General Purpose I/O Register 0

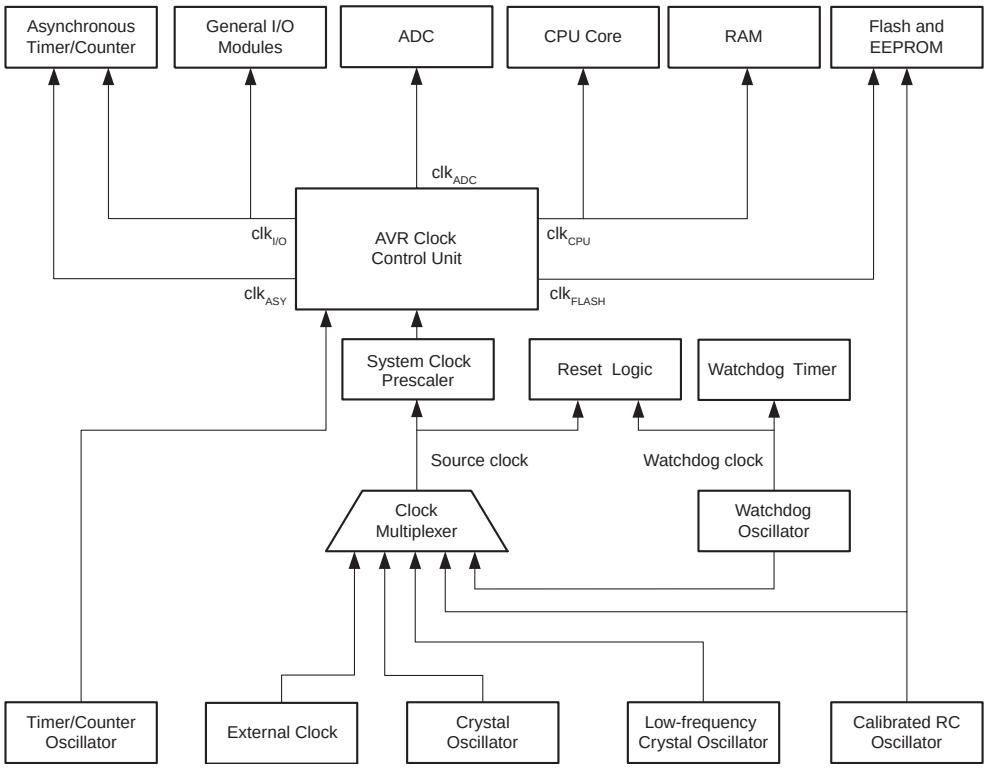
Bit	7	6	5	4	3	2	1	0	
0x1E (0x3E)	MSB							LSB	GPIOR0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

6. System Clock and Clock Options

6.1 Clock Systems and their Distribution

Figure 6-1 presents the principal clock systems in the AVR and their distribution. All of the clocks need not be active at a given time. In order to reduce power consumption, the clocks to modules not being used can be halted by using different sleep modes, as described in ”Power Management and Sleep Modes” on page 39. The clock systems are detailed below.

Figure 6-1. Clock Distribution



6.1.1 CPU Clock – clk_{CPU}

The CPU clock is routed to parts of the system concerned with operation of the AVR core. Examples of such modules are the General Purpose Register File, the Status Register and the data memory holding the Stack Pointer. Halting the CPU clock inhibits the core from performing general operations and calculations.

6.1.2 I/O Clock – $clk_{I/O}$

The I/O clock is used by the majority of the I/O modules, like Timer/Counters, SPI, and USART. The I/O clock is also used by the External Interrupt module, but note that some external interrupts are detected by asynchronous logic, allowing such interrupts to be detected even if the I/O clock is halted. Also note that start condition detection in the USI module is carried out asynchronously when $clk_{I/O}$ is halted, TWI address recognition in all sleep modes.

6.1.3 Flash Clock – clk_{FLASH}

The Flash clock controls operation of the Flash interface. The Flash clock is usually active simultaneously with the CPU clock.

6.1.4 Asynchronous Timer Clock – clk_{ASY}

The Asynchronous Timer clock allows the Asynchronous Timer/Counter to be clocked directly from an external clock or an external 32 kHz clock crystal. The dedicated clock domain allows using this Timer/Counter as a real-time counter even when the device is in sleep mode.

6.1.5 ADC Clock – clk_{ADC}

The ADC is provided with a dedicated clock domain. This allows halting the CPU and I/O clocks in order to reduce noise generated by digital circuitry. This gives more accurate ADC conversion results.

6.2 Clock Sources

The device has the following clock source options, selectable by Flash Fuse bits as shown below. The clock from the selected source is input to the AVR clock generator, and routed to the appropriate modules.

Table 6-1. Device Clocking Options Select(1)

Device Clocking Option	CKSEL3..0
Low Power Crystal Oscillator	1111 - 1000
Full Swing Crystal Oscillator	0111 - 0110
Low Frequency Crystal Oscillator	0101 - 0100
Internal 128 kHz RC Oscillator	0011
Calibrated Internal RC Oscillator	0010
External Clock	0000
Reserved	0001

Note: 1. For all fuses “1” means unprogrammed while “0” means programmed.

6.2.1 Default Clock Source

The device is shipped with internal RC oscillator at 8.0MHz and with the fuse CKDIV8 programmed, resulting in 1.0MHz system clock. The startup time is set to maximum and time-out period enabled. (CKSEL = "0010", SUT = "10", CKDIV8 = "0"). The default setting ensures that all users can make their desired clock source setting using any available programming interface.

6.2.2 Clock Startup Sequence

Any clock source needs a sufficient V_{CC} to start oscillating and a minimum number of oscillating cycles before it can be considered stable.

To ensure sufficient V_{CC}, the device issues an internal reset with a time-out delay (t_{TOUT}) after the device reset is released by all other reset sources. ”System Control and Reset” on page 46 describes the start conditions for the internal reset. The delay (t_{TOUT}) is timed from the Watchdog Oscillator and the number of cycles in the delay is set by the SUTx and CKSELx fuse bits. The

selectable delays are shown in Table 6-2. The frequency of the Watchdog Oscillator is voltage dependent as shown in "Typical Characteristics" on page 328.

Table 6-2. Number of Watchdog Oscillator Cycles

Typ Time-out (V_{CC} = 5.0V)	Typ Time-out (V_{CC} = 3.0V)	Number of Cycles
0 ms	0 ms	0
4.1 ms	4.3 ms	512
65 ms	69 ms	8K (8,192)

Main purpose of the delay is to keep the AVR in reset until it is supplied with minimum V_{CC}. The delay will not monitor the actual voltage and it will be required to select a delay longer than the V_{CC} rise time. If this is not possible, an internal or external Brown-Out Detection circuit should be used. A BOD circuit will ensure sufficient V_{CC} before it releases the reset, and the time-out delay can be disabled. Disabling the time-out delay without utilizing a Brown-Out Detection circuit is not recommended.

The oscillator is required to oscillate for a minimum number of cycles before the clock is considered stable. An internal ripple counter monitors the oscillator output clock, and keeps the internal reset active for a given number of clock cycles. The reset is then released and the device will start to execute. The recommended oscillator start-up time is dependent on the clock type, and varies from 6 cycles for an externally applied clock to 32K cycles for a low frequency crystal.

The start-up sequence for the clock includes both the time-out delay and the start-up time when the device starts up from reset. When starting up from Power-save or Power-down mode, V_{CC} is assumed to be at a sufficient level and only the start-up time is included.

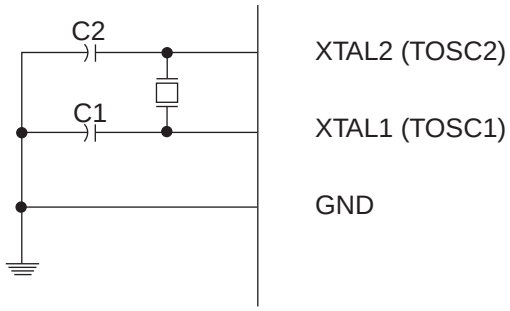
6.3 Low Power Crystal Oscillator

Pins XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an On-chip Oscillator, as shown in Figure 6-2 on page 29. Either a quartz crystal or a ceramic resonator may be used.

This Crystal Oscillator is a low power oscillator, with reduced voltage swing on the XTAL2 output. It gives the lowest power consumption, but is not capable of driving other clock inputs, and may be more susceptible to noise in noisy environments. In these cases, refer to the "Full Swing Crystal Oscillator" on page 30.

C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. Some initial guidelines for choosing capacitors for use with crystals are given in Table 6-3 on page 29. For ceramic resonators, the capacitor values given by the manufacturer should be used.

Figure 6-2. Crystal Oscillator Connections



The Low Power Oscillator can operate in three different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3..1 as shown in Table 6-3 on page 29.

Table 6-3. Low Power Crystal Oscillator Operating Modes[3]

Frequency Range[1] (MHz)	Recommended Range for Capacitors C1 and C2 (pF)	CKSEL3..1
0.4 - 0.9	–	100[2]
0.9 - 3.0	12 - 22	101
3.0 - 8.0	12 - 22	110
8.0 - 16.0	12 - 22	111

Notes:
1. This is the recommended CKSEL settings for the different frequency ranges.
2. This option should not be used with crystals, only with ceramic resonators.
3. If 8 MHz frequency exceeds the specification of the device (depends on V_{CC}), the CKDIV8 Fuse can be programmed in order to divide the internal frequency by 8. It must be ensured that the resulting divided clock meets the frequency specification of the device.

The CKSEL0 Fuse together with the SUT1..0 Fuses select the start-up times as shown in Table 6-4.

Table 6-4. Start-up Times for the Low Power Crystal Oscillator Clock Selection

Oscillator Source / Power Conditions	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	CKSEL0	SUT1..0
Ceramic resonator, fast rising power	258 CK	14CK + 4.1 ms[1]	0	00
Ceramic resonator, slowly rising power	258 CK	14CK + 65 ms[1]	0	01
Ceramic resonator, BOD enabled	1K CK	14CK[2]	0	10
Ceramic resonator, fast rising power	1K CK	14CK + 4.1 ms[2]	0	11
Ceramic resonator, slowly rising power	1K CK	14CK + 65 ms[2]	1	00

Table 6-4. Start-up Times for the Low Power Crystal Oscillator Clock Selection (Continued)

Oscillator Source / Power Conditions	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	CKSEL0	SUT1..0
Crystal Oscillator, BOD enabled	16K CK	14CK	1	01
Crystal Oscillator, fast rising power	16K CK	14CK + 4.1 ms	1	10
Crystal Oscillator, slowly rising power	16K CK	14CK + 65 ms	1	11

Notes: 1. These options should only be used when not operating close to the maximum frequency of the device, and only if frequency stability at start-up is not important for the application. These options are not suitable for crystals.
2. These options are intended for use with ceramic resonators and will ensure frequency stability at start-up. They can also be used with crystals when not operating close to the maximum frequency of the device, and if frequency stability at start-up is not important for the application.

6.4 Full Swing Crystal Oscillator

Pins XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an On-chip Oscillator, as shown in Figure 6-2 on page 29. Either a quartz crystal or a ceramic resonator may be used.

This Crystal Oscillator is a full swing oscillator, with rail-to-rail swing on the XTAL2 output. This is useful for driving other clock inputs and in noisy environments. The current consumption is higher than the "Low Power Crystal Oscillator" on page 28. Note that the Full Swing Crystal Oscillator will only operate for V_{CC} = 2.7 - 5.5 volts.

C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. Some initial guidelines for choosing capacitors for use with crystals are given in Table 6-6 on page 31. For ceramic resonators, the capacitor values given by the manufacturer should be used.

The operating mode is selected by the fuses CKSEL3..1 as shown in Table 6-5.

Table 6-5. Full Swing Crystal Oscillator operating modes(1)

Frequency Range (MHz)	Recommended Range for Capacitors C1 and C2 (pF)	CKSEL3..1
0.4 - 20	12 - 22	011

Notes: 1. If 8 MHz frequency exceeds the specification of the device (depends on V_{CC}), the CKDIV8 Fuse can be programmed in order to divide the internal frequency by 8. It must be ensured that the resulting divided clock meets the frequency specification of the device.

Figure 6-3. Crystal Oscillator Connections

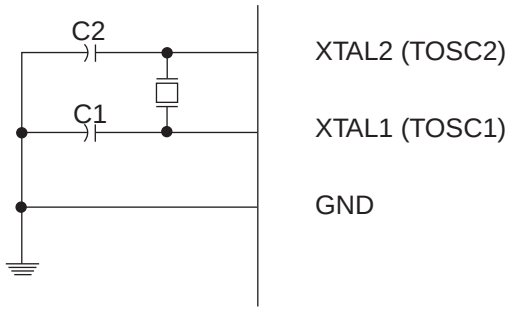


Table 6-6. Start-up Times for the Full Swing Crystal Oscillator Clock Selection

Oscillator Source / Power Conditions	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	CKSEL0	SUT1..0
Ceramic resonator, fast rising power	258 CK	14CK + 4.1 ms[1]	0	00
Ceramic resonator, slowly rising power	258 CK	14CK + 65 ms[1]	0	01
Ceramic resonator, BOD enabled	1K CK	14CK[2]	0	10
Ceramic resonator, fast rising power	1K CK	14CK + 4.1 ms[2]	0	11
Ceramic resonator, slowly rising power	1K CK	14CK + 65 ms[2]	1	00
Crystal Oscillator, BOD enabled	16K CK	14CK	1	01
Crystal Oscillator, fast rising power	16K CK	14CK + 4.1 ms	1	10
Crystal Oscillator, slowly rising power	16K CK	14CK + 65 ms	1	11

Notes:
1. These options should only be used when not operating close to the maximum frequency of the device, and only if frequency stability at start-up is not important for the application. These options are not suitable for crystals.
2. These options are intended for use with ceramic resonators and will ensure frequency stability at start-up. They can also be used with crystals when not operating close to the maximum frequency of the device, and if frequency stability at start-up is not important for the application.

6.5 Low Frequency Crystal Oscillator

The Low-frequency Crystal Oscillator is optimized for use with a 32.768 kHz watch crystal. When selecting crystals, load capasitance and crystal's Equivalent Series Resistance, ESR must be taken into consideration. Both values are specified by the crystal vendor. ATmega48P/88P/168P/328P oscillator is optimized for very low power consumption, and thus when selecting crystals, see Table 6-7 on page 32 for maximum ESR recommendations on 6.5 pF, 9.0 pF and 12.5 pF crystals

Table 6-7. Maximum ESR Recommendation for 32.768 kHz Crystal

Crystal CL (pF)	Max ESR [kΩ](1)
6.5	75
9.0	65
12.5	30

Note: 1. Maximum ESR is typical value based on characterization

The Low-frequency Crystal Oscillator provides an internal load capacitance of typical 6 pF at each TOSC pin. The external capacitance (C) needed at each TOSC pin can be calculated by using:

$$C = 2 \cdot CL - C_s$$

where CL is the load capacitance for a 32.768 kHz crystal specified by the crystal vendor and C_S is the total stray capacitance for one TOSC pin.

Crystals specifying load capacitance (CL) higher than 6 pF, require external capacitors applied as described in Figure 6-2 on page 29.

The Low-frequency Crystal Oscillator must be selected by setting the CKSEL Fuses to "0110" or "0111", as shown in Table 6-9. Start-up times are determined by the SUT Fuses as shown in Table 6-8.

Table 6-8. Start-up Times for the Low-frequency Crystal Oscillator Clock Selection

SUT1..0	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00	4 CK	Fast rising power or BOD enabled
01	4 CK + 4.1 ms	Slowly rising power
10	4 CK + 65 ms	Stable frequency at start-up
11	Reserved	

Table 6-9. Start-up Times for the Low-frequency Crystal Oscillator Clock Selection

CKSEL3..0	Start-up Time from Power-down and Power-save	Recommended Usage
0100(1)	1K CK	
0101	32K CK	Stable frequency at start-up

Note: 1. This option should only be used if frequency stability at start-up is not important for the application

6.6 Calibrated Internal RC Oscillator

By default, the Internal RC Oscillator provides an approximate 8.0 MHz clock. Though voltage and temperature dependent, this clock can be very accurately calibrated by the user. See Table 26-1 on page 319 for more details. The device is shipped with the CKDIV8 Fuse programmed. See "System Clock Prescaler" on page 35 for more details.

This clock may be selected as the system clock by programming the CKSEL Fuses as shown in Table 6-10. If selected, it will operate with no external components. During reset, hardware loads the pre-programmed calibration value into the OSCCAL Register and thereby automatically calibrates the RC Oscillator. The accuracy of this calibration is shown as Factory calibration in Table 26-1 on page 319.

By changing the OSCCAL register from SW, see "OSCCAL – Oscillator Calibration Register" on page 37, it is possible to get a higher calibration accuracy than by using the factory calibration. The accuracy of this calibration is shown as User calibration in Table 26-1 on page 319.

When this Oscillator is used as the chip clock, the Watchdog Oscillator will still be used for the Watchdog Timer and for the Reset Time-out. For more information on the pre-programmed calibration value, see the section "Calibration Byte" on page 299.

Table 6-10. Internal Calibrated RC Oscillator Operating Modes(2)

Frequency Range(1) (MHz)	CKSEL3..0
7.3 - 8.1	0010

Notes: 1. This is the recommended CKSEL settings for the different frequency ranges.
2. If 8 MHz frequency exceeds the specification of the device (depends on V_{CC}), the CKDIV8 Fuse can be programmed in order to divide the internal frequency by 8.

When this Oscillator is selected, start-up times are determined by the SUT Fuses as shown in Table 6-11 on page 33.

Table 6-11. Start-up times for the internal calibrated RC Oscillator clock selection

Power Conditions	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	SUT1..0
BOD enabled	6 CK	14CK(1)	00
Fast rising power	6 CK	14CK + 4.1 ms	01
Slowly rising power	6 CK	14CK + 65 ms(2)	10
Reserved			11

Note: 1. If the RSTDISBL fuse is programmed, this start-up time will be increased to 14CK + 4.1 ms to ensure programming mode can be entered.
2. The device is shipped with this option selected.

6.7 128 kHz Internal Oscillator

The 128 kHz internal Oscillator is a low power Oscillator providing a clock of 128 kHz. The frequency is nominal at 3V and 25°C. This clock may be select as the system clock by programming the CKSEL Fuses to "11" as shown in Table 6-12.

Table 6-12. 128 kHz Internal Oscillator Operating Modes

Nominal Frequency[1]	CKSEL3..0
128 kHz	0011

Note: 1. Note that the 128 kHz oscillator is a very low power clock source, and is not designed for a high accuracy.

When this clock source is selected, start-up times are determined by the SUT Fuses as shown in Table 6-13.

Table 6-13. Start-up Times for the 128 kHz Internal Oscillator

Power Conditions	Start-up Time from Power-down and Power-save	Additional Delay from Reset	SUT1..0
BOD enabled	6 CK	14CK[1]	00
Fast rising power	6 CK	14CK + 4 ms	01
Slowly rising power	6 CK	14CK + 64 ms	10
Reserved			11

Note: 1. If the RSTDISBL fuse is programmed, this start-up time will be increased to 14CK + 4.1 ms to ensure programming mode can be entered.

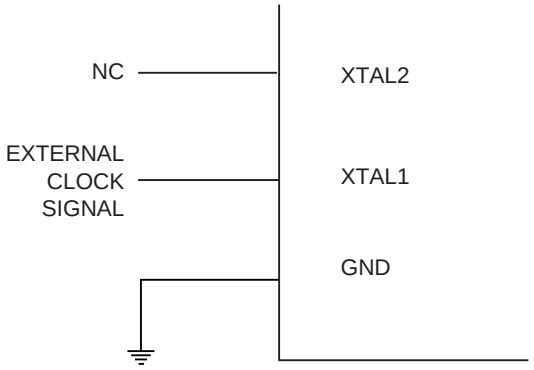
6.8 External Clock

To drive the device from an external clock source, XTAL1 should be driven as shown in Figure 6-4 on page 34. To run the device on an external clock, the CKSEL Fuses must be programmed to “0000” (see Table 6-14).

Table 6-14. Crystal Oscillator Clock Frequency

Frequency	CKSEL3..0
0 - 20 MHz	0000

Figure 6-4. External Clock Drive Configuration



When this clock source is selected, start-up times are determined by the SUT Fuses as shown in Table 6-15.

Table 6-15. Start-up Times for the External Clock Selection

Power Conditions	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	SUT1..0
BOD enabled	6 CK	14CK	00
Fast rising power	6 CK	14CK + 4.1 ms	01
Slowly rising power	6 CK	14CK + 65 ms	10
Reserved			11

When applying an external clock, it is required to avoid sudden changes in the applied clock frequency to ensure stable operation of the MCU. A variation in frequency of more than 2% from one clock cycle to the next can lead to unpredictable behavior. If changes of more than 2% is required, ensure that the MCU is kept in Reset during the changes.

Note that the System Clock Prescaler can be used to implement run-time changes of the internal clock frequency while still ensuring stable operation. Refer to ”System Clock Prescaler” on page 35 for details.

6.9 Clock Output Buffer

The device can output the system clock on the CLKO pin. To enable the output, the CKOUT Fuse has to be programmed. This mode is suitable when the chip clock is used to drive other circuits on the system. The clock also will be output during reset, and the normal operation of I/O pin will be overridden when the fuse is programmed. Any clock source, including the internal RC Oscillator, can be selected when the clock is output on CLKO. If the System Clock Prescaler is used, it is the divided system clock that is output.

6.10 Timer/Counter Oscillator

ATmega48P/88P/168P/328P uses the same crystal oscillator for Low-frequency Oscillator and Timer/Counter Oscillator. See ”Low Frequency Crystal Oscillator” on page 32 for details on the oscillator and crystal requirements.

ATmega48P/88P/168P/328P share the Timer/Counter Oscillator Pins (TOSC1 and TOSC2) with XTAL1 and XTAL2. When using the Timer/Counter Oscillator, the system clock needs to be four times the oscillator frequency. Due to this and the pin sharing, the Timer/Counter Oscillator can only be used when the Calibrated Internal RC Oscillator is selected as system clock source.

Applying an external clock source to TOSC1 can be done if EXTCLK in the ASSR Register is written to logic one. See ”Asynchronous Operation of Timer/Counter2” on page 155 for further description on selecting external clock as input instead of a 32.768 kHz watch crystal.

6.11 System Clock Prescaler

The ATmega48P/88P/168P/328P has a system clock prescaler, and the system clock can be divided by setting the ”CLKPR – Clock Prescale Register” on page 377. This feature can be used to decrease the system clock frequency and the power consumption when the requirement for processing power is low. This can be used with all clock source options, and it will affect the clock frequency of the CPU and all synchronous peripherals. $clk_{I/O}$, clk_{ADC}, clk_{CPU}, and clk_{FLASH} are divided by a factor as shown in Table 26-3 on page 320.

When switching between prescaler settings, the System Clock Prescaler ensures that no glitches occurs in the clock system. It also ensures that no intermediate frequency is higher than

neither the clock frequency corresponding to the previous setting, nor the clock frequency corresponding to the new setting. The ripple counter that implements the prescaler runs at the frequency of the undivided clock, which may be faster than the CPU's clock frequency. Hence, it is not possible to determine the state of the prescaler - even if it were readable, and the exact time it takes to switch from one clock division to the other cannot be exactly predicted. From the time the CLKPS values are written, it takes between T1 + T2 and T1 + 2 * T2 before the new clock frequency is active. In this interval, 2 active clock edges are produced. Here, T1 is the previous clock period, and T2 is the period corresponding to the new prescaler setting.

To avoid unintentional changes of clock frequency, a special write procedure must befollowed to change the CLKPS bits:

1. Write the Clock Prescaler Change Enable (CLKPCE) bit to one and all other bitsin CLKPR to zero.
2. Within four cycles, write the desired value to CLKPS while writing a zero to CLKPCE.

Interrupts must be disabled when changing prescaler setting to make sure the write procedure is not interrupted.

6.12 Register Description

6.12.1 OSCCAL – Oscillator Calibration Register

Bit	7	6	5	4	3	2	1	0	
(0x66)	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	OSCCAL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	Device Specific Calibration Value								

- **Bits 7..0 – CAL7..0: Oscillator Calibration Value**

The Oscillator Calibration Register is used to trim the Calibrated Internal RC Oscillator to remove process variations from the oscillator frequency. A pre-programmed calibration value is automatically written to this register during chip reset, giving the Factory calibrated frequency as specified in Table 26-1 on page 319. The application software can write this register to change the oscillator frequency. The oscillator can be calibrated to frequencies as specified in Table 26-1 on page 319. Calibration outside that range is not guaranteed.

Note that this oscillator is used to time EEPROM and Flash write accesses, and these write times will be affected accordingly. If the EEPROM or Flash are written, do not calibrate to more than 8.8 MHz. Otherwise, the EEPROM or Flash write may fail.

The CAL7 bit determines the range of operation for the oscillator. Setting this bit to 0 gives the lowest frequency range, setting this bit to 1 gives the highest frequency range. The two frequency ranges are overlapping, in other words a setting of OSCCAL = 0x7F gives a higher frequency than OSCCAL = 0x80.

The CAL6..0 bits are used to tune the frequency within the selected range. A setting of 0x00 gives the lowest frequency in that range, and a setting of 0x7F gives the highest frequency in the range.

6.12.2 CLKPR – Clock Prescale Register

Bit	7	6	5	4	3	2	1	0	
(0x61)	CLKPCE	–	–	–	CLKPS3	CLKPS2	CLKPS1	CLKPS0	CLKPR
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	See Bit Description				

- **Bit 7 – CLKPCE: Clock Prescaler Change Enable**

The CLKPCE bit must be written to logic one to enable change of the CLKPS bits. The CLKPCE bit is only updated when the other bits in CLKPR are simultaneously written to zero. CLKPCE is cleared by hardware four cycles after it is written or when CLKPS bits are written. Rewriting the CLKPCE bit within this time-out period does neither extend the time-out period, nor clear the CLKPCE bit.

- **Bits 3..0 – CLKPS3..0: Clock Prescaler Select Bits 3 - 0**

These bits define the division factor between the selected clock source and the internal system clock. These bits can be written run-time to vary the clock frequency to suit the application requirements. As the divider divides the master clock input to the MCU, the speed of all synchronous peripherals is reduced when a division factor is used. The division factors are given in Table 6-16 on page 38.

The CKDIV8 Fuse determines the initial value of the CLKPS bits. If CKDIV8 is unprogrammed, the CLKPS bits will be reset to “0000”. If CKDIV8 is programmed, CLKPS bits are reset to “0011”, giving a division factor of 8 at start up. This feature should be used if the selected clock source has a higher frequency than the maximum frequency of the device at the present operating conditions. Note that any value can be written to the CLKPS bits regardless of the CKDIV8 Fuse setting. The Application software must ensure that a sufficient division factor is chosen if the selected clock source has a higher frequency than the maximum frequency of the device at the present operating conditions. The device is shipped with the CKDIV8 Fuse programmed.

Table 6-16. Clock Prescaler Select

CLKPS3	CLKPS2	CLKPS1	CLKPS0	Clock Division Factor
0	0	0	0	1
0	0	0	1	2
0	0	1	0	4
0	0	1	1	8
0	1	0	0	16
0	1	0	1	32
0	1	1	0	64
0	1	1	1	128
1	0	0	0	256
1	0	0	1	Reserved
1	0	1	0	Reserved
1	0	1	1	Reserved
1	1	0	0	Reserved
1	1	0	1	Reserved
1	1	1	0	Reserved
1	1	1	1	Reserved

7. Power Management and Sleep Modes

Sleep modes enable the application to shut down unused modules in the MCU, thereby saving power. The AVR provides various sleep modes allowing the user to tailor the power consumption to the application's requirements.

When enabled, the Brown-out Detector (BOD) actively monitors the power supply voltage during the sleep periods. To further save power, it is possible to disable the BOD in some sleep modes. See "BOD Disable" on page 40 for more details.

7.1 Sleep Modes

Figure 6-1 on page 26 presents the different clock systems in the ATmega48P/88P/168P/328P, and their distribution. The figure is helpful in selecting an appropriate sleep mode. Table 7-1 shows the different sleep modes, their wake up sources BOD disable ability.

Table 7-1. Active Clock Domains and Wake-up Sources in the Different Sleep Modes.

	Active Clock Domains					Oscillators		Wake-up Sources							
Sleep Mode	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{ASY}	Main Clock Source Enabled	Timer Oscillator Enabled	INT1, INT0 and Pin Change	TWI Address Match	Timer2	SPM/EEPROM Ready	ADC	WDT	Other/O	Software BOD Disable
Idle			X	X	X	X	X(2)	X	X	X	X	X	X	X	
ADC Noise Reduction				X	X	X	X(2)	X(3)	X	X(2)	X	X	X		
Power-down								X(3)	X				X		X
Power-save					X		X(2)	X(3)	X	X			X		X
Standby(1)						X		X(3)	X				X		X
Extended Standby					X(2)	X	X(2)	X(3)	X	X			X		X

Notes: 1. Only recommended with external crystal or resonator selected as clock source.
2. If Timer/Counter2 is running in asynchronous mode.
3. For INT1 and INT0, only level interrupt.

To enter any of the six sleep modes, the SE bit in SMCR must be written to logic one and a SLEEP instruction must be executed. The SM2, SM1, and SM0 bits in the SMCR Register select which sleep mode (Idle, ADC Noise Reduction, Power-down, Power-save, Standby, or Extended Standby) will be activated by the SLEEP instruction. See Table 7-2 on page 44 for a summary.

If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up. The MCU is then halted for four cycles in addition to the start-up time, executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the Register File and SRAM are unaltered when the device wakes up from sleep. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset Vector.

7.2 BOD Disable

When the Brown-out Detector (BOD) is enabled by BODLEVEL fuses, Table 25-7 on page 297, the BOD is actively monitoring the power supply voltage during a sleep period. To save power, it is possible to disable the BOD by software for some of the sleep modes, see Table 7-1 on page 39. The sleep mode power consumption will then be at the same level as when BOD is globally disabled by fuses. If BOD is disabled in software, the BOD function is turned off immediately after entering the sleep mode. Upon wake-up from sleep, BOD is automatically enabled again. This ensures safe operation in case the V_{CC} level has dropped during the sleep period.

When the BOD has been disabled, the wake-up time from sleep mode will be approximately 60 µs to ensure that the BOD is working correctly before the MCU continues executing code.

BOD disable is controlled by bit 6, BODS (BOD Sleep) in the control register MCUCR, see "MCUCR – MCU Control Register" on page 44. Writing this bit to one turns off the BOD in relevant sleep modes, while a zero in this bit keeps BOD active. Default setting keeps BOD active, i.e. BODS set to zero.

Writing to the BODS bit is controlled by a timed sequence and an enable bit, see "MCUCR – MCU Control Register" on page 44.

7.3 Idle Mode

When the SM2..0 bits are written to 000, the SLEEP instruction makes the MCU enter Idle mode, stopping the CPU but allowing the SPI, USART, Analog Comparator, ADC, 2-wire Serial Interface, Timer/Counters, Watchdog, and the interrupt system to continue operating. This sleep mode basically halts clk_{CPU} and clk_{FLASH}, while allowing the other clocks to run.

Idle mode enables the MCU to wake up from external triggered interrupts as well as internal ones like the Timer Overflow and USART Transmit Complete interrupts. If wake-up from the Analog Comparator interrupt is not required, the Analog Comparator can be powered down by setting the ACD bit in the Analog Comparator Control and Status Register – ACSR. This will reduce power consumption in Idle mode. If the ADC is enabled, a conversion starts automatically when this mode is entered.

7.4 ADC Noise Reduction Mode

When the SM2..0 bits are written to 001, the SLEEP instruction makes the MCU enter ADC Noise Reduction mode, stopping the CPU but allowing the ADC, the external interrupts, the 2-wire Serial Interface address watch, Timer/Counter2[1], and the Watchdog to continue operating (if enabled). This sleep mode basically halts $clk_{I/O}$, clk_{CPU}, and clk_{FLASH}, while allowing the other clocks to run.

This improves the noise environment for the ADC, enabling higher resolution measurements. If the ADC is enabled, a conversion starts automatically when this mode is entered. Apart from the ADC Conversion Complete interrupt, only an External Reset, a Watchdog System Reset, a Watchdog Interrupt, a Brown-out Reset, a 2-wire Serial Interface address match, a Timer/Counter2 interrupt, an SPM/EEPROM ready interrupt, an external level interrupt on INT0 or INT1 or a pin change interrupt can wake up the MCU from ADC Noise Reduction mode.

Note: 1. Timer/Counter2 will only keep running in asynchronous mode, see "8-bit Timer/Counter2 with PWM and Asynchronous Operation" on page 144 for details.

7.5 Power-down Mode

When the SM2..0 bits are written to 010, the SLEEP instruction makes the MCU enter Power-down mode. In this mode, the external Oscillator is stopped, while the external interrupts, the 2-wire Serial Interface address watch, and the Watchdog continue operating (if enabled). Only an External Reset, a Watchdog System Reset, a Watchdog Interrupt, a Brown-out Reset, a 2-wire Serial Interface address match, an external level interrupt on INT0 or INT1, or a pin change interrupt can wake up the MCU. This sleep mode basically halts all generated clocks, allowing operation of asynchronous modules only.

Note that if a level triggered interrupt is used for wake-up from Power-down mode, the changed level must be held for some time to wake up the MCU. Refer to ”External Interrupts” on page 70 for details.

When waking up from Power-down mode, there is a delay from the wake-up condition occurs until the wake-up becomes effective. This allows the clock to restart and become stable after having been stopped. The wake-up period is defined by the same CKSEL Fuses that define the Reset Time-out period, as described in ”Clock Sources” on page 27.

7.6 Power-save Mode

When the SM2..0 bits are written to 011, the SLEEP instruction makes the MCU enter Power-save mode. This mode is identical to Power-down, with one exception:

If Timer/Counter2 is enabled, it will keep running during sleep. The device can wake up from either Timer Overflow or Output Compare event from Timer/Counter2 if the corresponding Timer/Counter2 interrupt enable bits are set in TIMSK2, and the Global Interrupt Enable bit in SREG is set.

If Timer/Counter2 is not running, Power-down mode is recommended instead of Power-save mode.

The Timer/Counter2 can be clocked both synchronously and asynchronously in Power-save mode. If Timer/Counter2 is not using the asynchronous clock, the Timer/Counter Oscillator is stopped during sleep. If Timer/Counter2 is not using the synchronous clock, the clock source is stopped during sleep. Note that even if the synchronous clock is running in Power-save, this clock is only available for Timer/Counter2.

7.7 Standby Mode

When the SM2..0 bits are 110 and an external crystal/resonator clock option is selected, the SLEEP instruction makes the MCU enter Standby mode. This mode is identical to Power-down with the exception that the Oscillator is kept running. From Standby mode, the device wakes up in six clock cycles.

7.8 Extended Standby Mode

When the SM2..0 bits are 111 and an external crystal/resonator clock option is selected, the SLEEP instruction makes the MCU enter Extended Standby mode. This mode is identical to Power-save with the exception that the Oscillator is kept running. From Extended Standby mode, the device wakes up in six clock cycles.

7.9 Power Reduction Register

The Power Reduction Register (PRR), see ”PRR – Power Reduction Register” on page 45, provides a method to stop the clock to individual peripherals to reduce power consumption. The current state of the peripheral is frozen and the I/O registers can not be read or written. Resources used by the peripheral when stopping the clock will remain occupied, hence the peripheral should in most cases be disabled before stopping the clock. Waking up a module, which is done by clearing the bit in PRR, puts the module in the same state as before shutdown.

Module shutdown can be used in Idle mode and Active mode to significantly reduce the overall power consumption. In all other sleep modes, the clock is already stopped.

7.10 Minimizing Power Consumption

There are several possibilities to consider when trying to minimize the power consumption in an AVR controlled system. In general, sleep modes should be used as much as possible, and the sleep mode should be selected so that as few as possible of the device’s functions are operating. All functions not needed should be disabled. In particular, the following modules may need special consideration when trying to achieve the lowest possible power consumption.

7.10.1 Analog to Digital Converter

If enabled, the ADC will be enabled in all sleep modes. To save power, the ADC should be disabled before entering any sleep mode. When the ADC is turned off and on again, the next conversion will be an extended conversion. Refer to ”Analog-to-Digital Converter” on page 250 for details on ADC operation.

7.10.2 Analog Comparator

When entering Idle mode, the Analog Comparator should be disabled if not used. When entering ADC Noise Reduction mode, the Analog Comparator should be disabled. In other sleep modes, the Analog Comparator is automatically disabled. However, if the Analog Comparator is set up to use the Internal Voltage Reference as input, the Analog Comparator should be disabled in all sleep modes. Otherwise, the Internal Voltage Reference will be enabled, independent of sleep mode. Refer to ”Analog Comparator” on page 246 for details on how to configure the Analog Comparator.

7.10.3 Brown-out Detector

If the Brown-out Detector is not needed by the application, this module should be turned off. If the Brown-out Detector is enabled by the BODLEVEL Fuses, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to ”Brown-out Detection” on page 48 for details on how to configure the Brown-out Detector.

7.10.4 Internal Voltage Reference

The Internal Voltage Reference will be enabled when needed by the Brown-out Detection, the Analog Comparator or the ADC. If these modules are disabled as described in the sections above, the internal voltage reference will be disabled and it will not be consuming power. When turned on again, the user must allow the reference to start up before the output is used. If the reference is kept on in sleep mode, the output can be used immediately. Refer to ”Internal Voltage Reference” on page 49 for details on the start-up time.

7.10.5 Watchdog Timer

If the Watchdog Timer is not needed in the application, the module should be turned off. If the Watchdog Timer is enabled, it will be enabled in all sleep modes and hence always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to ”Watchdog Timer” on page 50 for details on how to configure the Watchdog Timer.

7.10.6 Port Pins

When entering a sleep mode, all port pins should be configured to use minimum power. The most important is then to ensure that no pins drive resistive loads. In sleep modes where both the I/O clock ($clk_{I/O}$) and the ADC clock (clk_{ADC}) are stopped, the input buffers of the device will be disabled. This ensures that no power is consumed by the input logic when not needed. In some cases, the input logic is needed for detecting wake-up conditions, and it will then be enabled. Refer to the section ”Digital Input Enable and Sleep Modes” on page 79 for details on which pins are enabled. If the input buffer is enabled and the input signal is left floating or have an analog signal level close to $V_{CC}/2$, the input buffer will use excessive power.

For analog input pins, the digital input buffer should be disabled at all times. An analog signal level close to $V_{CC}/2$ on an input pin can cause significant current even in active mode. Digital input buffers can be disabled by writing to the Digital Input Disable Registers (DIDR1 and DIDR0). Refer to ”DIDR1 – Digital Input Disable Register 1” on page 249 and ”DIDR0 – Digital Input Disable Register 0” on page 266 for details.

7.10.7 On-chip Debug System

If the On-chip debug system is enabled by the DWEN Fuse and the chip enters sleep mode, the main clock source is enabled and hence always consumes power. In the deeper sleep modes, this will contribute significantly to the total current consumption.

7.11 Register Description

7.11.1 SMCR – Sleep Mode Control Register

The Sleep Mode Control Register contains control bits for power management.

Bit	7	6	5	4	3	2	1	0	
0x33 (0x53)	–	–	–	–	SM2	SM1	SM0	SE	SMCR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..4 Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bits 3..1 – SM2..0: Sleep Mode Select Bits 2, 1, and 0**

These bits select between the five available sleep modes as shown in Table 7-2.

Table 7-2. Sleep Mode Select

SM2	SM1	SM0	Sleep Mode
0	0	0	Idle
0	0	1	ADC Noise Reduction
0	1	0	Power-down
0	1	1	Power-save
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Standby(1)
1	1	1	External Standby(1)

Note: 1. Standby mode is only recommended for use with external crystals or resonators.

- **Bit 0 – SE: Sleep Enable**

The SE bit must be written to logic one to make the MCU enter the sleep mode when the SLEEP instruction is executed. To avoid the MCU entering the sleep mode unless it is the programmer's purpose, it is recommended to write the Sleep Enable (SE) bit to one just before the execution of the SLEEP instruction and to clear it immediately after waking up.

7.11.2 MCUCR – MCU Control Register

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	–	BODS	BODSE	PUD	–	–	IVSEL	IVCE	MCUCR
Read/Write	R	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 6 – BODS: BOD Sleep**

The BODS bit must be written to logic one in order to turn off BOD during sleep, see Table 7-1 on page 39. Writing to the BODS bit is controlled by a timed sequence and an enable bit, BODSE in MCUCR. To disable BOD in relevant sleep modes, both BODS and BODSE must first

be set to one. Then, to set the BODS bit, BODS must be set to one and BODSE must be set to zero within four clock cycles.

The BODS bit is active three clock cycles after it is set. A sleep instruction must be executed while BODS is active in order to turn off the BOD for the actual sleep mode. The BODS bit is automatically cleared after three clock cycles.

- **Bit 5 – BODSE: BOD Sleep Enable**

BODSE enables setting of BODS control bit, as explained in BODS bit description. BOD disable is controlled by a timed sequence.

7.11.3 PRR – Power Reduction Register

Bit	7	6	5	4	3	2	1	0	
(0x64)	**PRTWI**	**PRTIM2**	**PRTIM0**	**–**	**PRTIM1**	**PRSPI**	**PRUSART0**	**PRADC**	**PRR**
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 - PRTWI: Power Reduction TWI**

Writing a logic one to this bit shuts down the TWI by stopping the clock to the module. When waking up the TWI again, the TWI should be re initialized to ensure proper operation.

- **Bit 6 - PRTIM2: Power Reduction Timer/Counter2**

Writing a logic one to this bit shuts down the Timer/Counter2 module in synchronous mode (AS2 is 0). When the Timer/Counter2 is enabled, operation will continue like before the shutdown.

- **Bit 5 - PRTIM0: Power Reduction Timer/Counter0**

Writing a logic one to this bit shuts down the Timer/Counter0 module. When the Timer/Counter0 is enabled, operation will continue like before the shutdown.

- **Bit 4 - Res: Reserved bit**

This bit is reserved in ATmega48P/88P/168P/328P and will always read as zero.

- **Bit 3 - PRTIM1: Power Reduction Timer/Counter1**

Writing a logic one to this bit shuts down the Timer/Counter1 module. When the Timer/Counter1 is enabled, operation will continue like before the shutdown.

- **Bit 2 - PRSPI: Power Reduction Serial Peripheral Interface**

If using debugWIRE On-chip Debug System, this bit should not be written to one.

Writing a logic one to this bit shuts down the Serial Peripheral Interface by stopping the clock to the module. When waking up the SPI again, the SPI should be re initialized to ensure proper operation.

- **Bit 1 - PRUSART0: Power Reduction USART0**

Writing a logic one to this bit shuts down the USART by stopping the clock to the module. When waking up the USART again, the USART should be re initialized to ensure proper operation.

- **Bit 0 - PRADC: Power Reduction ADC**

Writing a logic one to this bit shuts down the ADC. The ADC must be disabled before shut down. The analog comparator cannot use the ADC input MUX when the ADC is shut down.

8. System Control and Reset

8.1 Resetting the AVR

During reset, all I/O Registers are set to their initial values, and the program starts execution from the Reset Vector. For the ATmega168P, the instruction placed at the Reset Vector must be a JMP – Absolute Jump – instruction to the reset handling routine. For the ATmega48P and ATmega88P, the instruction placed at the Reset Vector must be an RJMP – Relative Jump – instruction to the reset handling routine. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa (ATmega88P/168P only). The circuit diagram in Figure 8-1 on page 47 shows the reset logic. Table 26-3 on page 320 defines the electrical parameters of the reset circuitry.

The I/O ports of the AVR are immediately reset to their initial state when a reset source goes active. This does not require any clock source to be running.

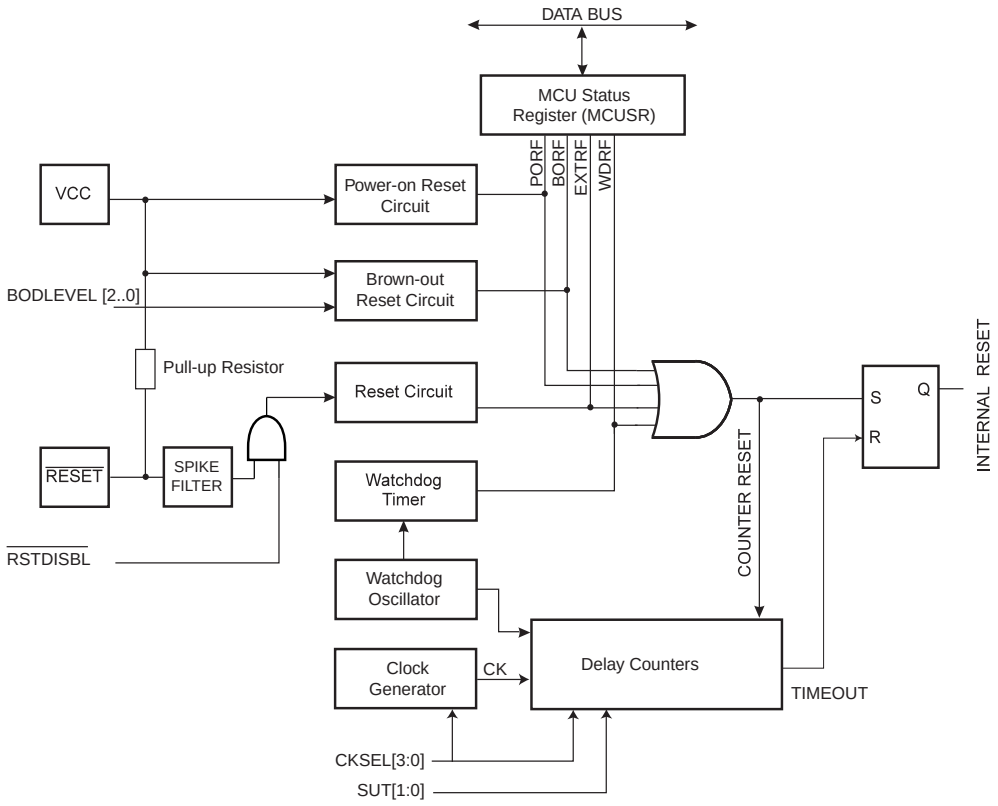
After all reset sources have gone inactive, a delay counter is invoked, stretching the internal reset. This allows the power to reach a stable level before normal operation starts. The time-out period of the delay counter is defined by the user through the SUT and CKSEL Fuses. The different selections for the delay period are presented in ”Clock Sources” on page 27.

8.2 Reset Sources

The ATmega48P/88P/168P/328P has four sources of reset:

- Power-on Reset. The MCU is reset when the supply voltage is below the Power-on Reset threshold (V_{POT}).
- External Reset. The MCU is reset when a low level is present on the $\overline{\text{RESET}}$ pin for longer than the minimum pulse length.
- Watchdog System Reset. The MCU is reset when the Watchdog Timer period expires and the Watchdog System Reset mode is enabled.
- Brown-out Reset. The MCU is reset when the supply voltage V_{CC} is below the Brown-out Reset threshold (V_{BOT}) and the Brown-out Detector is enabled.

Figure 8-1. Reset Logic



8.3 Power-on Reset

A Power-on Reset (POR) pulse is generated by an On-chip detection circuit. The detection level is defined in "System and Reset Characteristics" on page 320. The POR is activated whenever V_{CC} is below the detection level. The POR circuit can be used to trigger the start-up Reset, as well as to detect a failure in supply voltage.

A Power-on Reset (POR) circuit ensures that the device is reset from Power-on. Reaching the Power-on Reset threshold voltage invokes the delay counter, which determines how long the device is kept in RESET after V_{CC} rise. The RESET signal is activated again, without any delay, when V_{CC} decreases below the detection level.

Figure 8-2. MCU Start-up, $\overline{\text{RESET}}$ Tied to V_{CC}

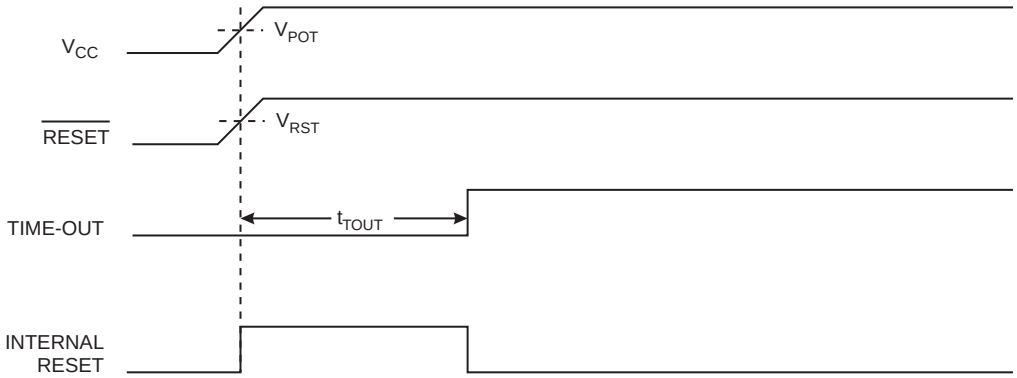
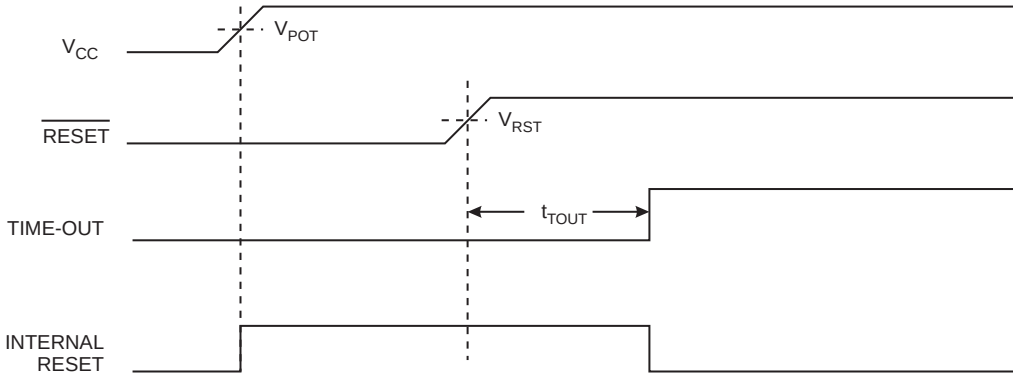


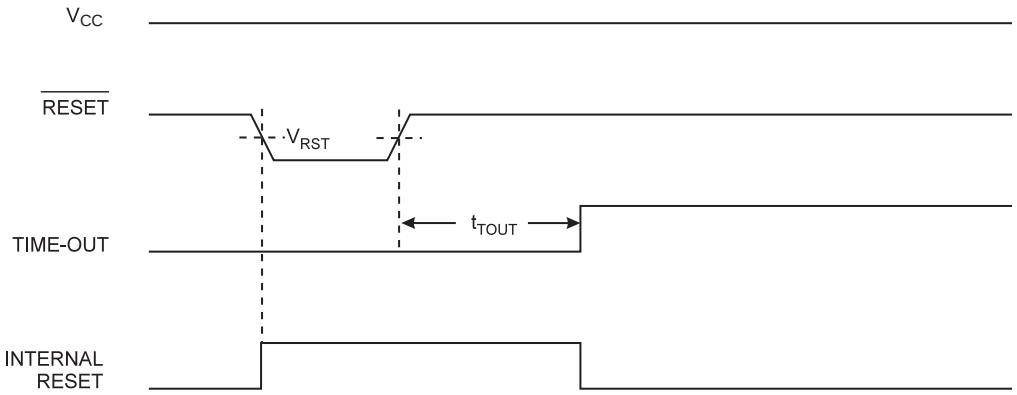
Figure 8-3. MCU Start-up, $\overline{\text{RESET}}$ Extended Externally



8.4 External Reset

An External Reset is generated by a low level on the $\overline{\text{RESET}}$ pin. Reset pulses longer than the minimum pulse width (see "System and Reset Characteristics" on page 320) will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage – V_{RST} – on its positive edge, the delay counter starts the MCU after the Time-out period – t_{TOUT} – has expired. The External Reset can be disabled by the RSTDISBL fuse, see Table 25-7 on page 297.

Figure 8-4. External Reset During Operation

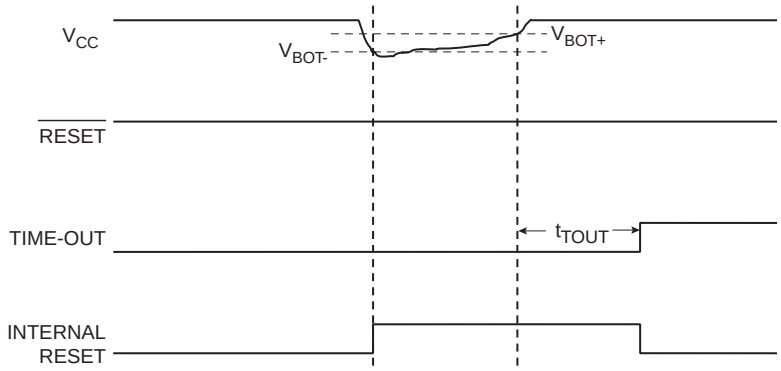


8.5 Brown-out Detection

ATmega48P/88P/168P/328P has an On-chip Brown-out Detection (BOD) circuit for monitoring the V_{CC} level during operation by comparing it to a fixed trigger level. The trigger level for the BOD can be selected by the BODLEVEL Fuses. The trigger level has a hysteresis to ensure spike free Brown-out Detection. The hysteresis on the detection level should be interpreted as $V_{BOT+} = V_{BOT} + V_{HYST}/2$ and $V_{BOT-} = V_{BOT} - V_{HYST}/2$.When the BOD is enabled, and V_{CC} decreases to a value below the trigger level (V_{BOT-} in Figure 8-5 on page 49), the Brown-out Reset is immediately activated. When V_{CC} increases above the trigger level (V_{BOT+} in Figure 8-5 on page 49), the delay counter starts the MCU after the Time-out period t_{TOUT} has expired.

The BOD circuit will only detect a drop in V_{CC} if the voltage stays below the trigger level for longer than t_{BOD} given in "System and Reset Characteristics" on page 320.

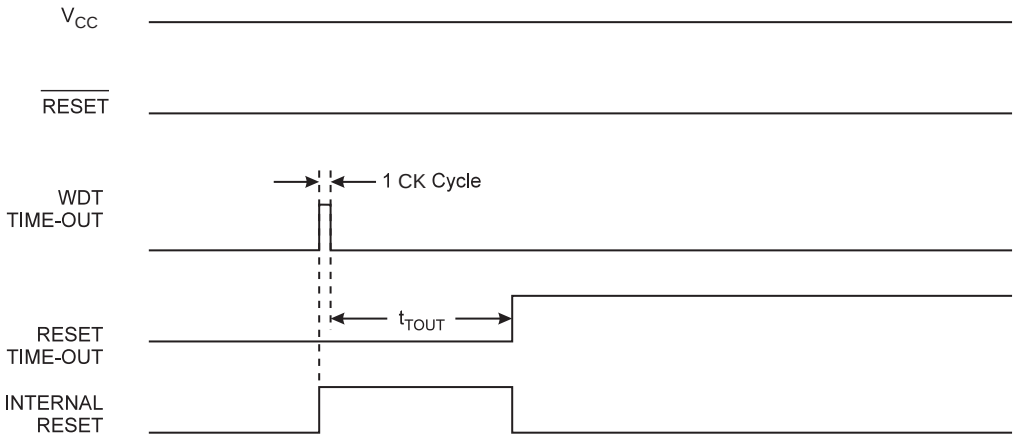
Figure 8-5. Brown-out Reset During Operation



8.6 Watchdog System Reset

When the Watchdog times out, it will generate a short reset pulse of one CK cycle duration. On the falling edge of this pulse, the delay timer starts counting the Time-out period t_{TOUT}. Refer to page 50 for details on operation of the Watchdog Timer.

Figure 8-6. Watchdog System Reset During Operation



8.7 Internal Voltage Reference

ATmega48P/88P/168P/328P features an internal bandgap reference. This reference is used for Brown-out Detection, and it can be used as an input to the Analog Comparator or the ADC.

8.7.1 Voltage Reference Enable Signals and Start-up Time

The voltage reference has a start-up time that may influence the way it should be used. The start-up time is given in "System and Reset Characteristics" on page 320. To save power, the reference is not always turned on. The reference is on during the following situations:

1. When the BOD is enabled (by programming the BODLEVEL [2:0] Fuses).
2. When the bandgap reference is connected to the Analog Comparator (by setting the ACBG bit in ACSR).
3. When the ADC is enabled.

Thus, when the BOD is not enabled, after setting the ACBG bit or enabling the ADC, the user must always allow the reference to start up before the output from the Analog Comparator or

ADC is used. To reduce power consumption in Power-down mode, the user can avoid the three conditions above to ensure that the reference is turned off before entering Power-down mode.

8.8 Watchdog Timer

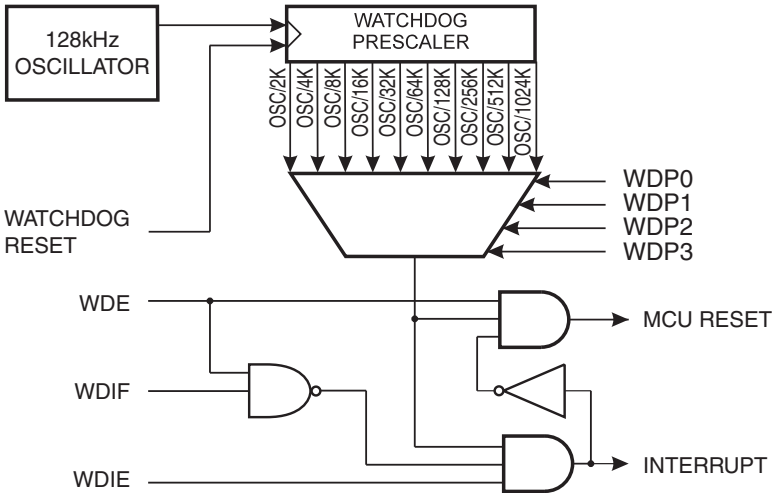
8.8.1 Features

- **Clocked from separate On-chip Oscillator**
- **3 Operating modes**
  - **Interrupt**
  - **System Reset**
  - **Interrupt and System Reset**
- **Selectable Time-out period from 16ms to 8s**
- **Possible Hardware fuse Watchdog always on (WDTON) for fail-safe mode**

8.8.2 Overview

ATmega48P/88P/168P/328P has an Enhanced Watchdog Timer (WDT). The WDT is a timer counting cycles of a separate on-chip 128 kHz oscillator. The WDT gives an interrupt or a system reset when the counter reaches a given time-out value. In normal operation mode, it is required that the system uses the WDR - Watchdog Timer Reset - instruction to restart the counter before the time-out value is reached. If the system doesn't restart the counter, an interrupt or system reset will be issued.

Figure 8-7. Watchdog Timer



In Interrupt mode, the WDT gives an interrupt when the timer expires. This interrupt can be used to wake the device from sleep-modes, and also as a general system timer. One example is to limit the maximum time allowed for certain operations, giving an interrupt when the operation has run longer than expected. In System Reset mode, the WDT gives a reset when the timer expires. This is typically used to prevent system hang-up in case of runaway code. The third mode, Interrupt and System Reset mode, combines the other two modes by first giving an interrupt and then switch to System Reset mode. This mode will for instance allow a safe shutdown by saving critical parameters before a system reset.

The Watchdog always on (WDTON) fuse, if programmed, will force the Watchdog Timer to System Reset mode. With the fuse programmed the System Reset mode bit (WDE) and Interrupt

mode bit (WDIE) are locked to 1 and 0 respectively. To further ensure program security, alterations to the Watchdog set-up must follow timed sequences. The sequence for clearing WDE and changing time-out configuration is as follows:

1. In the same operation, write a logic one to the Watchdog change enable bit (WDCE) and WDE. A logic one must be written to WDE regardless of the previous value of the WDE bit.
2. Within the next four clock cycles, write the WDE and Watchdog prescaler bits (WDP) as desired, but with the WDCE bit cleared. This must be done in one operation.

The following code example shows one assembly and one C function for turning off the Watchdog Timer. The example assumes that interrupts are controlled (e.g. by disabling interrupts globally) so that no interrupts will occur during the execution of these functions.

Assembly Code Example[1]

```
WDT_off:
  ; Turn off global interrupt
  cli
  ; Reset Watchdog Timer
  wdr
  ; Clear WDRF in MCUSR
  in    r16, MCUSR
  andi  r16, (0xff & (0<<WDRF))
  out   MCUSR, r16
  ; Write logical one to WDCE and WDE
  ; Keep old prescaler setting to prevent unintentional time-out
  lds r16, WDTCSR
  ori   r16, (1<<WDCE) | (1<<WDE)
  sts WDTCSR, r16
  ; Turn off WDT
  ldi   r16, (0<<WDE)
  sts WDTCSR, r16
  ; Turn on global interrupt
  sei
  ret
```

C Code Example[1]

```
void WDT_off(void)
{
  __disable_interrupt();
  __watchdog_reset();
  /* Clear WDRF in MCUSR */
  MCUSR &= ~(1<<WDRF);
  /* Write logical one to WDCE and WDE */
  /* Keep old prescaler setting to prevent unintentional time-out */
  WDTCSR |= (1<<WDCE) | (1<<WDE);
  /* Turn off WDT */
  WDTCSR = 0x00;
  __enable_interrupt();
}
```

Note: 1. See ”Code Examples” on page 7.

Note: If the Watchdog is accidentally enabled, for example by a runaway pointer or brown-out condition, the device will be reset and the Watchdog Timer will stay enabled. If the code is not set up to handle the Watchdog, this might lead to an eternal loop of time-out resets. To avoid this situation, the application software should always clear the Watchdog System Reset Flag (WDRF) and the WDE control bit in the initialisation routine, even if the Watchdog is not in use.

The following code example shows one assembly and one C function for changing the time-out value of the Watchdog Timer.

Assembly Code Example[1]

```
WDT_Prescaler_Change:
  ; Turn off global interrupt
  cli
  ; Reset Watchdog Timer
  wdr
  ; Start timed sequence
  lds r16, WDTCSR
  ori   r16, (1<<WDCE) | (1<<WDE)
  sts WDTCSR, r16
  ; --  Got four cycles to set the new values from here -
  ; Set new prescaler(time-out) value = 64K cycles (~0.5 s)
  ldi   r16, (1<<WDE) | (1<<WDP2) | (1<<WDP0)
  sts WDTCSR, r16
  ; --  Finished setting new values, used 2 cycles -
  ; Turn on global interrupt
  sei
  ret
```

C Code Example[1]

```
void WDT_Prescaler_Change(void)
{
  __disable_interrupt();
  __watchdog_reset();
  /* Start timed  equence */
  WDTCSR |= (1<<WDCE) | (1<<WDE);
  /* Set new prescaler(time-out) value = 64K cycles (~0.5 s) */
  WDTCSR  = (1<<WDE) | (1<<WDP2) | (1<<WDP0);
  __enable_interrupt();
}
```

Note: 1. See "Code Examples" on page 7.

Note: The Watchdog Timer should be reset before any change of the WDP bits, since a change in the WDP bits can result in a time-out when switching to a shorter time-out period.

8.9 Register Description

8.9.1 MCUSR – MCU Status Register

The MCU Status Register provides information on which reset source caused an MCU reset.

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	–	–	–	–	WDRF	BORF	EXTRF	PORF	MCUSR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	See Bit Description				

- **Bit 7..4: Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 3 – WDRF: Watchdog System Reset Flag**

This bit is set if a Watchdog System Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 2 – BORF: Brown-out Reset Flag**

This bit is set if a Brown-out Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 1 – EXTRF: External Reset Flag**

This bit is set if an External Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 0 – PORF: Power-on Reset Flag**

This bit is set if a Power-on Reset occurs. The bit is reset only by writing a logic zero to the flag.

To make use of the Reset Flags to identify a reset condition, the user should read and then Reset the MCUSR as early as possible in the program. If the register is cleared before another reset occurs, the source of the reset can be found by examining the Reset Flags.

8.9.2 WDTCSR – Watchdog Timer Control Register

Bit	7	6	5	4	3	2	1	0	
(0x60)	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0	WDTCSR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	X	0	0	0	

- **Bit 7 - WDIF: Watchdog Interrupt Flag**

This bit is set when a time-out occurs in the Watchdog Timer and the Watchdog Timer is configured for interrupt. WDIF is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, WDIF is cleared by writing a logic one to the flag. When the I-bit in SREG and WDIE are set, the Watchdog Time-out Interrupt is executed.

- **Bit 6 - WDIE: Watchdog Interrupt Enable**

When this bit is written to one and the I-bit in the Status Register is set, the Watchdog Interrupt is enabled. If WDE is cleared in combination with this setting, the Watchdog Timer is in Interrupt Mode, and the corresponding interrupt is executed if time-out in the Watchdog Timer occurs. If WDE is set, the Watchdog Timer is in Interrupt and System Reset Mode. The first time-out in the Watchdog Timer will set WDIF. Executing the corresponding interrupt vector will clear WDIE and

WDIF automatically by hardware (the Watchdog goes to System Reset Mode). This is useful for keeping the Watchdog Timer security while using the interrupt. To stay in Interrupt and System Reset Mode, WDIE must be set after each interrupt. This should however not be done within the interrupt service routine itself, as this might compromise the safety-function of the Watchdog System Reset mode. If the interrupt is not executed before the next time-out, a System Reset will be applied.

Table 8-1. Watchdog Timer Configuration

WDTON[1]	WDE	WDIE	Mode	Action on Time-out
1	0	0	Stopped	None
1	0	1	Interrupt Mode	Interrupt
1	1	0	System Reset Mode	Reset
1	1	1	Interrupt and System Reset Mode	Interrupt, then go to System Reset Mode
0	x	x	System Reset Mode	Reset

Note: 1. WDTON Fuse set to “0” means programmed and “1” means unprogrammed.

- **Bit 4 - WDCE: Watchdog Change Enable**

This bit is used in timed sequences for changing WDE and prescaler bits. To clear the WDE bit, and/or change the prescaler bits, WDCE must be set.

Once written to one, hardware will clear WDCE after four clock cycles.

- **Bit 3 - WDE: Watchdog System Reset Enable**

WDE is overridden by WDRF in MCUSR. This means that WDE is always set when WDRF is set. To clear WDE, WDRF must be cleared first. This feature ensures multiple resets during conditions causing failure, and a safe start-up after the failure.

- **Bit 5, 2..0 - WDP3..0: Watchdog Timer Prescaler 3, 2, 1 and 0**

The WDP3..0 bits determine the Watchdog Timer prescaling when the Watchdog Timer is running. The different prescaling values and their corresponding time-out periods are shown in Table 8-2 on page 55.

Table 8-2. Watchdog Timer Prescale Select

WDP3	WDP2	WDP1	WDP0	Number of WDT Oscillator Cycles	Typical Time-out at V_{CC} = 5.0V
0	0	0	0	2K (2048) cycles	16 ms
0	0	0	1	4K (4096) cycles	32 ms
0	0	1	0	8K (8192) cycles	64 ms
0	0	1	1	16K (16384) cycles	0.125 s
0	1	0	0	32K (32768) cycles	0.25 s
0	1	0	1	64K (65536) cycles	0.5 s
0	1	1	0	128K (131072) cycles	1.0 s
0	1	1	1	256K (262144) cycles	2.0 s

Table 8-2. Watchdog Timer Prescale Select (Continued)

WDP3	WDP2	WDP1	WDP0	Number of WDT Oscillator Cycles	Typical Time-out at V_{CC} = 5.0V
1	0	0	0	512K (524288) cycles	4.0 s
1	0	0	1	1024K (1048576) cycles	8.0 s
1	0	1	0	Reserved	
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

9. Interrupts

This section describes the specifics of the interrupt handling as performed in ATmega48P/88P/168P/328P. For a general explanation of the AVR interrupt handling, refer to ”Reset and Interrupt Handling” on page 14.

The interrupt vectors in ATmega48P, ATmega88P, ATmega168P, and ATmega328P are generally the same, with the following differences:

- Each Interrupt Vector occupies two instruction words in ATmega168P and ATmega328P, and one instruction word in ATmega48P and ATmega88P.
- ATmega48P does not have a separate Boot Loader Section. In ATmega88P, ATmega168P, and ATmega328P, the Reset Vector is affected by the BOOTRST fuse, and the Interrupt Vector start address is affected by the IVSEL bit in MCUCR.

9.1 Interrupt Vectors in ATmega48P

Table 9-1. Reset and Interrupt Vectors in ATmega48P

Vector No.	Program Address	Source	Interrupt Definition
1	0x000	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x001	INT0	External Interrupt Request 0
3	0x002	INT1	External Interrupt Request 1
4	0x003	PCINT0	Pin Change Interrupt Request 0
5	0x004	PCINT1	Pin Change Interrupt Request 1
6	0x005	PCINT2	Pin Change Interrupt Request 2
7	0x006	WDT	Watchdog Time-out Interrupt
8	0x007	TIMER2 COMPA	Timer/Counter2 Compare Match A
9	0x008	TIMER2 COMPB	Timer/Counter2 Compare Match B
10	0x009	TIMER2 OVF	Timer/Counter2 Overflow
11	0x00A	TIMER1 CAPT	Timer/Counter1 Capture Event
12	0x00B	TIMER1 COMPA	Timer/Counter1 Compare Match A
13	0x00C	TIMER1 COMPB	Timer/Coutner1 Compare Match B
14	0x00D	TIMER1 OVF	Timer/Counter1 Overflow
15	0x00E	TIMER0 COMPA	Timer/Counter0 Compare Match A
16	0x00F	TIMER0 COMPB	Timer/Counter0 Compare Match B
17	0x010	TIMER0 OVF	Timer/Counter0 Overflow
18	0x011	SPI, STC	SPI Serial Transfer Complete
19	0x012	USART, RX	USART Rx Complete
20	0x013	USART, UDRE	USART, Data Register Empty
21	0x014	USART, TX	USART, Tx Complete
22	0x015	ADC	ADC Conversion Complete
23	0x016	EE READY	EEPROM Ready

Table 9-1. Reset and Interrupt Vectors in ATmega48P (Continued)

Vector No.	Program Address	Source	Interrupt Definition
24	0x017	ANALOG COMP	Analog Comparator
25	0x018	TWI	2-wire Serial Interface
26	0x019	SPM READY	Store Program Memory Ready

The most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega48P is:

```
Address  Labels Code                 Comments
0x000           rjmp   RESET          ; Reset Handler
0x001           rjmp   EXT_INT0       ; IRQ0 Handler
0x002           rjmp   EXT_INT1       ; IRQ1 Handler
0x003           rjmp   PCINT0         ; PCINT0 Handler
0x004           rjmp   PCINT1         ; PCINT1 Handler
0x005           rjmp   PCINT2         ; PCINT2 Handler
0x006           rjmp   WDT            ; Watchdog Timer Handler
0x007           rjmp   TIM2_COMPA     ; Timer2 Compare A Handler
0x008           rjmp   TIM2_COMPB     ; Timer2 Compare B Handler
0x009           rjmp   TIM2_OVF       ; Timer2 Overflow Handler
0x00A           rjmp   TIM1_CAPT      ; Timer1 Capture Handler
0x00B           rjmp   TIM1_COMPA     ; Timer1 Compare A Handler
0x00C           rjmp   TIM1_COMPB     ; Timer1 Compare B Handler
0x00D           rjmp   TIM1_OVF       ; Timer1 Overflow Handler
0x00E           rjmp   TIM0_COMPA     ; Timer0 Compare A Handler
0x00F           rjmp   TIM0_COMPB     ; Timer0 Compare B Handler
0x010           rjmp   TIM0_OVF       ; Timer0 Overflow Handler
0x011           rjmp   SPI_STC        ; SPI Transfer Complete Handler
0x012           rjmp   USART_RXC      ; USART, RX Complete Handler
0x013           rjmp   USART_UDRE     ; USART, UDR Empty Handler
0x014           rjmp   USART_TXC      ; USART, TX Complete Handler
0x015           rjmp   ADC            ; ADC Conversion Complete Handler
0x016           rjmp   EE_RDY         ; EEPROM Ready Handler
0x017           rjmp   ANA_COMP       ; Analog Comparator Handler
0x018           rjmp   TWI            ; 2-wire Serial Interface Handler
0x019           rjmp   SPM_RDY        ; Store Program Memory Ready Handler
;
0x01ARESET:     ldi    r16, high(RAMEND); Main program start
0x01B           out    SPH,r16        ; Set Stack Pointer to top of RAM
0x01C           ldi    r16, low(RAMEND)
0x01D           out    SPL,r16
0x01E           sei                   ; Enable interrupts
0x01F           <instr>  xxx
  ...     ...     ...  ...
```

9.2 Interrupt Vectors in ATmega88P

Table 9-2. Reset and Interrupt Vectors in ATmega88P

Vector No.	Program Address(2)	Source	Interrupt Definition
1	0x000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x001	INT0	External Interrupt Request 0
3	0x002	INT1	External Interrupt Request 1
4	0x003	PCINT0	Pin Change Interrupt Request 0
5	0x004	PCINT1	Pin Change Interrupt Request 1
6	0x005	PCINT2	Pin Change Interrupt Request 2
7	0x006	WDT	Watchdog Time-out Interrupt
8	0x007	TIMER2 COMPA	Timer/Counter2 Compare Match A
9	0x008	TIMER2 COMPB	Timer/Counter2 Compare Match B
10	0x009	TIMER2 OVF	Timer/Counter2 Overflow
11	0x00A	TIMER1 CAPT	Timer/Counter1 Capture Event
12	0x00B	TIMER1 COMPA	Timer/Counter1 Compare Match A
13	0x00C	TIMER1 COMPB	Timer/Coutner1 Compare Match B
14	0x00D	TIMER1 OVF	Timer/Counter1 Overflow
15	0x00E	TIMER0 COMPA	Timer/Counter0 Compare Match A
16	0x00F	TIMER0 COMPB	Timer/Counter0 Compare Match B
17	0x010	TIMER0 OVF	Timer/Counter0 Overflow
18	0x011	SPI, STC	SPI Serial Transfer Complete
19	0x012	USART, RX	USART Rx Complete
20	0x013	USART, UDRE	USART, Data Register Empty
21	0x014	USART, TX	USART, Tx Complete
22	0x015	ADC	ADC Conversion Complete
23	0x016	EE READY	EEPROM Ready
24	0x017	ANALOG COMP	Analog Comparator
25	0x018	TWI	2-wire Serial Interface
26	0x019	SPM READY	Store Program Memory Ready

Notes: 1. When the BOOTRST Fuse is programmed, the device will jump to the Boot Loader address at reset, see "Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P" on page 277.
2. When the IVSEL bit in MCUCR is set, Interrupt Vectors will be moved to the start of the Boot Flash Section. The address of each Interrupt Vector will then be the address in this table added to the start address of the Boot Flash Section.

Table 9-3 on page 60 shows reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa.

Table 9-3. Reset and Interrupt Vectors Placement in ATmega88P[1]

BOOTRST	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	0x000	0x001
1	1	0x000	Boot Reset Address + 0x001
0	0	Boot Reset Address	0x001
0	1	Boot Reset Address	Boot Reset Address + 0x001

Note: 1. The Boot Reset Address is shown in Table 24-7 on page 289. For the BOOTRST Fuse “1” means unprogrammed while “0” means programmed.

The most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega88P is:

```
Address  Labels Code                 Comments
0x000           rjmp   RESET          ; Reset Handler
0x001           rjmp   EXT_INT0       ; IRQ0 Handler
0x002           rjmp   EXT_INT1       ; IRQ1 Handler
0x003           rjmp   PCINT0         ; PCINT0 Handler
0x004           rjmp   PCINT1         ; PCINT1 Handler
0x005           rjmp   PCINT2         ; PCINT2 Handler
0x006           rjmp   WDT            ; Watchdog Timer Handler
0x007           rjmp   TIM2_COMPA     ; Timer2 Compare A Handler
0X008           rjmp   TIM2_COMPB     ; Timer2 Compare B Handler
0x009           rjmp   TIM2_OVF       ; Timer2 Overflow Handler
0x00A           rjmp   TIM1_CAPT      ; Timer1 Capture Handler
0x00B           rjmp   TIM1_COMPA     ; Timer1 Compare A Handler
0x00C           rjmp   TIM1_COMPB     ; Timer1 Compare B Handler
0x00D           rjmp   TIM1_OVF       ; Timer1 Overflow Handler
0x00E           rjmp   TIM0_COMPA     ; Timer0 Compare A Handler
0x00F           rjmp   TIM0_COMPB     ; Timer0 Compare B Handler
0x010           rjmp   TIM0_OVF       ; Timer0 Overflow Handler
0x011           rjmp   SPI_STC        ; SPI Transfer Complete Handler
0x012           rjmp   USART_RXC      ; USART, RX Complete Handler
0x013           rjmp   USART_UDRE     ; USART, UDR Empty Handler
0x014           rjmp   USART_TXC      ; USART, TX Complete Handler
0x015           rjmp   ADC            ; ADC Conversion Complete Handler
0x016           rjmp   EE_RDY         ; EEPROM Ready Handler
0x017           rjmp   ANA_COMP       ; Analog Comparator Handler
0x018           rjmp   TWI            ; 2-wire Serial Interface Handler
0x019           rjmp   SPM_RDY        ; Store Program Memory Ready Handler
;
0x01ARESET:     ldi    r16, high(RAMEND); Main program start
0x01B           out    SPH,r16        ; Set Stack Pointer to top of RAM
0x01C           ldi    r16, low(RAMEND)
0x01D           out    SPL,r16
0x01E           sei                   ; Enable interrupts
0x01F           <instr>  xxx
```

When the BOOTRST Fuse is unprogrammed, the Boot section size set to 2K bytes and the IVSEL bit in the MCUCR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega88P is:

```
Address  Labels Code                 Comments
0x000    RESET: ldi    r16,high(RAMEND); Main program start
0x001           out    SPH,r16        ; Set Stack Pointer to top of RAM
0x002           ldi    r16,low(RAMEND)
0x003           out    SPL,r16
0x004           sei                   ; Enable interrupts
0x005           <instr>  xxx
;
.org 0xC01
0x0C01          rjmp   EXT_INT0       ; IRQ0 Handler
0x0C02          rjmp   EXT_INT1       ; IRQ1 Handler
...             ...    ...            ;
0x0C19          rjmp   SPM_RDY        ; Store Program Memory Ready Handler
```

When the BOOTRST Fuse is programmed and the Boot section size set to 2K bytes, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega88P is:

```
Address  Labels Code            Comments
.org 0x001
0x001           rjmp   EXT_INT0       ; IRQ0 Handler
0x002           rjmp   EXT_INT1       ; IRQ1 Handler
...             ...    ...            ;
0x019           rjmp   SPM_RDY        ; Store Program Memory Ready Handler
;
.org 0xC00
0xC00    RESET: ldi    r16,high(RAMEND); Main program start
0xC01           out    SPH,r16        ; Set Stack Pointer to top of RAM
0xC02           ldi    r16,low(RAMEND)
0xC03           out    SPL,r16
0xC04           sei                   ; Enable interrupts
0xC05           <instr>  xxx
```

When the BOOTRST Fuse is programmed, the Boot section size set to 2K bytes and the IVSEL bit in the MCUCR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega88P is:

```
Address  Labels Code                 Comments
;
.org 0xC00
0xC00           rjmp   RESET          ; Reset handler
0xC01           rjmp   EXT_INT0       ; IRQ0 Handler
0xC02           rjmp   EXT_INT1       ; IRQ1 Handler
...             ...    ...            ;
0xC19           rjmp   SPM_RDY        ; Store Program Memory Ready Handler
;
0xC1A    RESET: ldi    r16,high(RAMEND); Main program start
```

```
0xC1B          out    SPH,r16        ; Set Stack Pointer to top of RAM
0xC1C          ldi    r16,low(RAMEND)
0xC1D          out    SPL,r16
0xC1E          sei                   ; Enable interrupts
0xC1F          <instr>  xxx
```

9.3 Interrupt Vectors in ATmega168P

Table 9-4. Reset and Interrupt Vectors in ATmega168P

VectorNo.	Program Address(2)	Source	Interrupt Definition
1	0x0000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 1
4	0x0006	PCINT0	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2
7	0x000C	WDT	Watchdog Time-out Interrupt
8	0x000E	TIMER2 COMPA	Timer/Counter2 Compare Match A
9	0x0010	TIMER2 COMPB	Timer/Counter2 Compare Match B
10	0x0012	TIMER2 OVF	Timer/Counter2 Overflow
11	0x0014	TIMER1 CAPT	Timer/Counter1 Capture Event
12	0x0016	TIMER1 COMPA	Timer/Counter1 Compare Match A
13	0x0018	TIMER1 COMPB	Timer/Coutner1 Compare Match B
14	0x001A	TIMER1 OVF	Timer/Counter1 Overflow
15	0x001C	TIMER0 COMPA	Timer/Counter0 Compare Match A
16	0x001E	TIMER0 COMPB	Timer/Counter0 Compare Match B
17	0x0020	TIMER0 OVF	Timer/Counter0 Overflow
18	0x0022	SPI, STC	SPI Serial Transfer Complete
19	0x0024	USART, RX	USART Rx Complete
20	0x0026	USART, UDRE	USART, Data Register Empty
21	0x0028	USART, TX	USART, Tx Complete
22	0x002A	ADC	ADC Conversion Complete
23	0x002C	EE READY	EEPROM Ready
24	0x002E	ANALOG COMP	Analog Comparator
25	0x0030	TWI	2-wire Serial Interface
26	0x0032	SPM READY	Store Program Memory Ready

Notes: 1. When the BOOTRST Fuse is programmed, the device will jump to the Boot Loader address at reset, see ”Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P” on page 277.
2. When the IVSEL bit in MCUCR is set, Interrupt Vectors will be moved to the start of the Boot Flash Section. The address of each Interrupt Vector will then be the address in this table added to the start address of the Boot Flash Section.

Table 9-5 on page 63 shows reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa.

Table 9-5. Reset and Interrupt Vectors Placement in ATmega168P(1)

BOOTRST	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	0x000	0x002
1	1	0x000	Boot Reset Address + 0x0002
0	0	Boot Reset Address	0x002
0	1	Boot Reset Address	Boot Reset Address + 0x0002

Note: 1. The Boot Reset Address is shown in Table 24-7 on page 289. For the BOOTRST Fuse “1” means unprogrammed while “0” means programmed.

The most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega168P is:

```
Address  Labels Code                 Comments
0x0000          jmp    RESET         ; Reset Handler
0x0002          jmp    EXT_INT0      ; IRQ0 Handler
0x0004          jmp    EXT_INT1      ; IRQ1 Handler
0x0006          jmp    PCINT0        ; PCINT0 Handler
0x0008          jmp    PCINT1        ; PCINT1 Handler
0x000A          jmp    PCINT2        ; PCINT2 Handler
0x000C          jmp    WDT           ; Watchdog Timer Handler
0x000E          jmp    TIM2_COMPA    ; Timer2 Compare A Handler
0x0010          jmp    TIM2_COMPB    ; Timer2 Compare B Handler
0x0012          jmp    TIM2_OVF      ; Timer2 Overflow Handler
0x0014          jmp    TIM1_CAPT     ; Timer1 Capture Handler
0x0016          jmp    TIM1_COMPA    ; Timer1 Compare A Handler
0x0018          jmp    TIM1_COMPB    ; Timer1 Compare B Handler
0x001A          jmp    TIM1_OVF      ; Timer1 Overflow Handler
0x001C          jmp    TIM0_COMPA    ; Timer0 Compare A Handler
0x001E          jmp    TIM0_COMPB    ; Timer0 Compare B Handler
0x0020          jmp    TIM0_OVF      ; Timer0 Overflow Handler
0x0022          jmp    SPI_STC       ; SPI Transfer Complete Handler
0x0024          jmp    USART_RXC     ; USART, RX Complete Handler
0x0026          jmp    USART_UDRE    ; USART, UDR Empty Handler
0x0028          jmp    USART_TXC     ; USART, TX Complete Handler
0x002A          jmp    ADC           ; ADC Conversion Complete Handler
0x002C          jmp    EE_RDY        ; EEPROM Ready Handler
0x002E          jmp    ANA_COMP      ; Analog Comparator Handler
0x0030          jmp    TWI           ; 2-wire Serial Interface Handler
0x0032          jmp    SPM_RDY       ; Store Program Memory Ready Handler
;
0x0033RESET:    ldi    r16, high(RAMEND); Main program start
```

```
0x0034          out    SPH,r16        ; Set Stack Pointer to top of RAM
0x0035          ldi    r16, low(RAMEND)
0x0036          out    SPL,r16
0x0037          sei                   ; Enable interrupts
0x0038          <instr>  xxx
  ...     ...     ...  ...
```

When the BOOTRST Fuse is unprogrammed, the Boot section size set to 2K bytes and the IVSEL bit in the MCUCR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega168P is:

```
Address  Labels Code                 Comments
0x0000   RESET: ldi    r16,high(RAMEND); Main program start
0x0001          out    SPH,r16        ; Set Stack Pointer to top of RAM
0x0002          ldi    r16,low(RAMEND)
0x0003          out    SPL,r16
0x0004          sei                   ; Enable interrupts
0x0005          <instr>  xxx
;
.org 0x1C02
0x1C02          jmp    EXT_INT0       ; IRQ0 Handler
0x1C04          jmp    EXT_INT1       ; IRQ1 Handler
...             ...    ...            ;
0x1C32          jmp    SPM_RDY        ; Store Program Memory Ready Handler
```

When the BOOTRST Fuse is programmed and the Boot section size set to 2K bytes, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega168P is:

```
Address  Labels Code            Comments
.org 0x0002
0x0002          jmp    EXT_INT0       ; IRQ0 Handler
0x0004          jmp    EXT_INT1       ; IRQ1 Handler
...             ...    ...            ;
0x0032          jmp    SPM_RDY        ; Store Program Memory Ready Handler
;
.org 0x1C00
0x1C00   RESET: ldi    r16,high(RAMEND); Main program start
0x1C01          out    SPH,r16        ; Set Stack Pointer to top of RAM
0x1C02          ldi    r16,low(RAMEND)
0x1C03          out    SPL,r16
0x1C04          sei                   ; Enable interrupts
0x1C05          <instr>  xxx
```

When the BOOTRST Fuse is programmed, the Boot section size set to 2K bytes and the IVSEL bit in the MCUCR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega168P is:

```
Address  Labels Code                  Comments
;
.org 0x1C00
0x1C00          jmp    RESET          ; Reset handler
0x1C02          jmp    EXT_INT0       ; IRQ0 Handler
0x1C04          jmp    EXT_INT1       ; IRQ1 Handler
...             ...    ...            ;
0x1C32          jmp    SPM_RDY        ; Store Program Memory Ready Handler
;
0x1C33  RESET: ldi     r16,high(RAMEND); Main program start
0x1C34          out    SPH,r16        ; Set Stack Pointer to top of RAM
0x1C35          ldi    r16,low(RAMEND)
0x1C36          out    SPL,r16
0x1C37          sei                   ; Enable interrupts
0x1C38          <instr>  xxx
```

9.4 Interrupt Vectors in ATmega328P

Table 9-6. Reset and Interrupt Vectors in ATmega328P

VectorNo.	Program Address[2]	Source	Interrupt Definition
1	0x0000[1]	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 1
4	0x0006	PCINT0	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2
7	0x000C	WDT	Watchdog Time-out Interrupt
8	0x000E	TIMER2 COMPA	Timer/Counter2 Compare Match A
9	0x0010	TIMER2 COMPB	Timer/Counter2 Compare Match B
10	0x0012	TIMER2 OVF	Timer/Counter2 Overflow
11	0x0014	TIMER1 CAPT	Timer/Counter1 Capture Event
12	0x0016	TIMER1 COMPA	Timer/Counter1 Compare Match A
13	0x0018	TIMER1 COMPB	Timer/Coutner1 Compare Match B
14	0x001A	TIMER1 OVF	Timer/Counter1 Overflow
15	0x001C	TIMER0 COMPA	Timer/Counter0 Compare Match A
16	0x001E	TIMER0 COMPB	Timer/Counter0 Compare Match B
17	0x0020	TIMER0 OVF	Timer/Counter0 Overflow
18	0x0022	SPI, STC	SPI Serial Transfer Complete
19	0x0024	USART, RX	USART Rx Complete
20	0x0026	USART, UDRE	USART, Data Register Empty
21	0x0028	USART, TX	USART, Tx Complete

Table 9-6. Reset and Interrupt Vectors in ATmega328P (Continued)

VectorNo.	Program Address[2]	Source	Interrupt Definition
22	0x002A	ADC	ADC Conversion Complete
23	0x002C	EE READY	EEPROM Ready
24	0x002E	ANALOG COMP	Analog Comparator
25	0x0030	TWI	2-wire Serial Interface
26	0x0032	SPM READY	Store Program Memory Ready

Notes: 1. When the BOOTRST Fuse is programmed, the device will jump to the Boot Loader address at reset, see "Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P" on page 277.
2. When the IVSEL bit in MCUCR is set, Interrupt Vectors will be moved to the start of the Boot Flash Section. The address of each Interrupt Vector will then be the address in this table added to the start address of the Boot Flash Section.

Table 9-7 on page 66 shows reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa.

Table 9-7. Reset and Interrupt Vectors Placement in ATmega328P[1]

BOOTRST	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	0x000	0x002
1	1	0x000	Boot Reset Address + 0x0002
0	0	Boot Reset Address	0x002
0	1	Boot Reset Address	Boot Reset Address + 0x0002

Note: 1. The Boot Reset Address is shown in Table 24-7 on page 289. For the BOOTRST Fuse "1" means unprogrammed while "0" means programmed.

The most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega328P is:

```
Address  Labels Code                 Comments
0x0000         jmp    RESET          ; Reset Handler
0x0002         jmp    EXT_INT0       ; IRQ0 Handler
0x0004         jmp    EXT_INT1       ; IRQ1 Handler
0x0006         jmp    PCINT0         ; PCINT0 Handler
0x0008         jmp    PCINT1         ; PCINT1 Handler
0x000A         jmp    PCINT2         ; PCINT2 Handler
0x000C         jmp    WDT            ; Watchdog Timer Handler
0x000E         jmp    TIM2_COMPA     ; Timer2 Compare A Handler
0x0010         jmp    TIM2_COMPB     ; Timer2 Compare B Handler
0x0012         jmp    TIM2_OVF       ; Timer2 Overflow Handler
0x0014         jmp    TIM1_CAPT      ; Timer1 Capture Handler
0x0016         jmp    TIM1_COMPA     ; Timer1 Compare A Handler
0x0018         jmp    TIM1_COMPB     ; Timer1 Compare B Handler
0x001A         jmp    TIM1_OVF       ; Timer1 Overflow Handler
```

```
0x001C          jmp     TIM0_COMPA      ; Timer0 Compare A Handler
0x001E          jmp     TIM0_COMPB      ; Timer0 Compare B Handler
0x0020          jmp     TIM0_OVF        ; Timer0 Overflow Handler
0x0022          jmp     SPI_STC         ; SPI Transfer Complete Handler
0x0024          jmp     USART_RXC       ; USART, RX Complete Handler
0x0026          jmp     USART_UDRE      ; USART, UDR Empty Handler
0x0028          jmp     USART_TXC       ; USART, TX Complete Handler
0x002A          jmp     ADC             ; ADC Conversion Complete Handler
0x002C          jmp     EE_RDY          ; EEPROM Ready Handler
0x002E          jmp     ANA_COMP        ; Analog Comparator Handler
0x0030          jmp     TWI             ; 2-wire Serial Interface Handler
0x0032          jmp     SPM_RDY         ; Store Program Memory Ready Handler
;
0x0033RESET:    ldi     r16, high(RAMEND); Main program start
0x0034          out     SPH,r16         ; Set Stack Pointer to top of RAM
0x0035          ldi     r16, low(RAMEND)
0x0036          out     SPL,r16
0x0037          sei                     ; Enable interrupts
0x0038          <instr>  xxx
  ...     ...     ...  ...
```

When the BOOTRST Fuse is unprogrammed, the Boot section size set to 2K bytes and the IVSEL bit in the MCUCR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega328P is:

```
Address  Labels Code                    Comments
0x0000   RESET: ldi     r16,high(RAMEND); Main program start
0x0001          out     SPH,r16         ; Set Stack Pointer to top of RAM
0x0002          ldi     r16,low(RAMEND)
0x0003          out     SPL,r16
0x0004          sei                     ; Enable interrupts
0x0005          <instr>  xxx
;
.org 0x3C02
0x3C02          jmp     EXT_INT0        ; IRQ0 Handler
0x3C04          jmp     EXT_INT1        ; IRQ1 Handler
...             ...     ...             ;
0x3C32          jmp     SPM_RDY         ; Store Program Memory Ready Handler
```

When the BOOTRST Fuse is programmed and the Boot section size set to 2K bytes, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega328P is:

```
Address  Labels Code             Comments
.org 0x0002
0x0002          jmp     EXT_INT0        ; IRQ0 Handler
0x0004          jmp     EXT_INT1        ; IRQ1 Handler
...             ...     ...             ;
0x0032          jmp     SPM_RDY         ; Store Program Memory Ready Handler
;
```

```
.org 0x3C00
0x3C00   RESET: ldi    r16,high(RAMEND); Main program start
0x3C01          out    SPH,r16        ; Set Stack Pointer to top of RAM
0x3C02          ldi    r16,low(RAMEND)
0x3C03          out    SPL,r16
0x3C04          sei                   ; Enable interrupts
0x3C05          <instr>  xxx
```

When the BOOTRST Fuse is programmed, the Boot section size set to 2K bytes and the IVSEL bit in the MCUCR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega328P is:

```
Address  Labels Code                 Comments
;
.org 0x3C00
0x3C00          jmp    RESET          ; Reset handler
0x3C02          jmp    EXT_INT0       ; IRQ0 Handler
0x3C04          jmp    EXT_INT1       ; IRQ1 Handler
...             ...    ...            ;
0x3C32          jmp    SPM_RDY        ; Store Program Memory Ready Handler
;
0x3C33   RESET: ldi    r16,high(RAMEND); Main program start
0x3C34          out    SPH,r16        ; Set Stack Pointer to top of RAM
0x3C35          ldi    r16,low(RAMEND)
0x3C36          out    SPL,r16
0x3C37          sei                   ; Enable interrupts
0x3C38          <instr>  xxx
```

9.5 Register Description

9.5.1 Moving Interrupts Between Application and Boot Space, ATmega88P, ATmega168P and ATmega328P

The MCU Control Register controls the placement of the Interrupt Vector table.

9.5.2 MCUCR – MCU Control Register

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	–	BODS	BODSE	PUD	–	–	IVSEL	IVCE	MCUCR
Read/Write	R	R	R	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 1 – IVSEL: Interrupt Vector Select**

When the IVSEL bit is cleared (zero), the Interrupt Vectors are placed at the start of the Flash memory. When this bit is set (one), the Interrupt Vectors are moved to the beginning of the Boot Loader section of the Flash. The actual address of the start of the Boot Flash Section is determined by the BOOTSZ Fuses. Refer to the section ”Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P” on page 277 for details. To avoid unintentional changes of Interrupt Vector tables, a special write procedure must be followed to change the IVSEL bit:

a. Write the Interrupt Vector Change Enable (IVCE) bit to one.

b. Within four cycles, write the desired value to IVSEL while writing a zero to IVCE.

Interrupts will automatically be disabled while this sequence is executed. Interrupts are disabled in the cycle IVCE is set, and they remain disabled until after the instruction following the write to IVSEL. If IVSEL is not written, interrupts remain disabled for four cycles. The I-bit in the Status Register is unaffected by the automatic disabling.

Note: If Interrupt Vectors are placed in the Boot Loader section and Boot Lock bit BLB02 is programmed, interrupts are disabled while executing from the Application section. If Interrupt Vectors are placed in the Application section and Boot Lock bit BLB12 is programed, interrupts are disabled while executing from the Boot Loader section. Refer to the section "Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P" on page 277 for details on Boot Lock bits.

This bit is not available in ATmega48P.

- **Bit 0 – IVCE: Interrupt Vector Change Enable**

The IVCE bit must be written to logic one to enable change of the IVSEL bit. IVCE is cleared by hardware four cycles after it is written or when IVSEL is written. Setting the IVCE bit will disable interrupts, as explained in the IVSEL description above. See Code Example below.

Assembly Code Example

```
Move_interrupts:
  ; Enable change of Interrupt Vectors
  ldi  r16, (1<<IVCE)
  out  MCUCR, r16
  ; Move interrupts to Boot Flash section
  ldi  r16, (1<<IVSEL)
  out  MCUCR, r16
  ret
```

C Code Example

```
void Move_interrupts(void)
{
  /* Enable change of Interrupt Vectors */
  MCUCR = (1<<IVCE);
  /* Move interrupts to Boot Flash section */
  MCUCR = (1<<IVSEL);
}
```

This bit is not available in ATmega48P.

10. External Interrupts

The External Interrupts are triggered by the INT0 and INT1 pins or any of the PCINT23..0 pins. Observe that, if enabled, the interrupts will trigger even if the INT0 and INT1 or PCINT23..0 pins are configured as outputs. This feature provides a way of generating a software interrupt. The pin change interrupt PCI2 will trigger if any enabled PCINT23..16 pin toggles. The pin change interrupt PCI1 will trigger if any enabled PCINT14..8 pin toggles. The pin change interrupt PCI0 will trigger if any enabled PCINT7..0 pin toggles. The PCMSK2, PCMSK1 and PCMSK0 Registers control which pins contribute to the pin change interrupts. Pin change interrupts on PCINT23..0 are detected asynchronously. This implies that these interrupts can be used for waking the part also from sleep modes other than Idle mode.

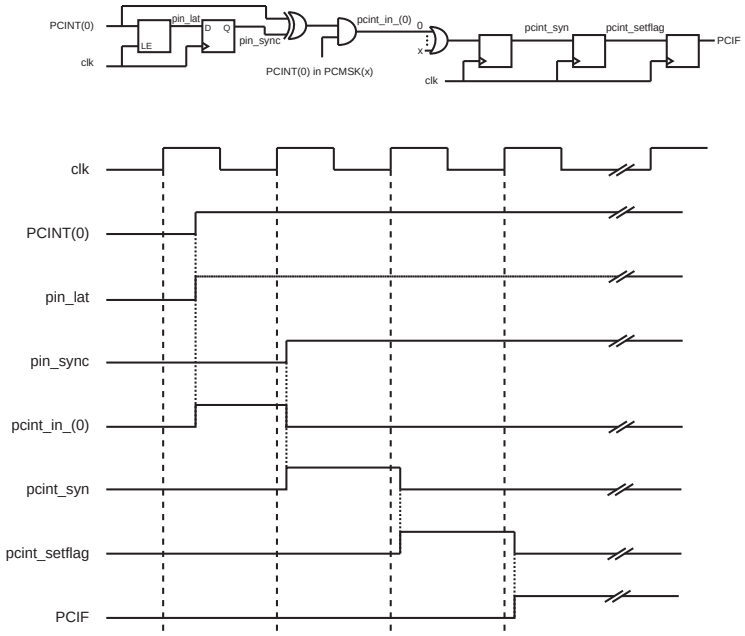
The INT0 and INT1 interrupts can be triggered by a falling or rising edge or a low level. This is set up as indicated in the specification for the External Interrupt Control Register A – EICRA. When the INT0 or INT1 interrupts are enabled and are configured as level triggered, the interrupts will trigger as long as the pin is held low. Note that recognition of falling or rising edge interrupts on INT0 or INT1 requires the presence of an I/O clock, described in ”Clock Systems and their Distribution” on page 26. Low level interrupt on INT0 and INT1 is detected asynchronously. This implies that this interrupt can be used for waking the part also from sleep modes other than Idle mode. The I/O clock is halted in all sleep modes except Idle mode.

Note that if a level triggered interrupt is used for wake-up from Power-down, the required level must be held long enough for the MCU to complete the wake-up to trigger the level interrupt. If the level disappears before the end of the Start-up Time, the MCU will still wake up, but no interrupt will be generated. The start-up time is defined by the SUT and CKSEL Fuses as described in ”System Clock and Clock Options” on page 26.

10.1 Pin Change Interrupt Timing

An example of timing of a pin change interrupt is shown in Figure 10-1.

Figure 10-1. Timing of pin change interrupts



10.2 Register Description

10.2.1 EICRA – External Interrupt Control Register A

The External Interrupt Control Register A contains control bits for interrupt sense control.

Bit	7	6	5	4	3	2	1	0	
(0x69)	–	–	–	–	ISC11	ISC10	ISC01	ISC00	EICRA
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..4 – Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 3, 2 – ISC11, ISC10: Interrupt Sense Control 1 Bit 1 and Bit 0**

The External Interrupt 1 is activated by the external pin INT1 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT1 pin that activate the interrupt are defined in Table 10-1. The value on the INT1 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 10-1. Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request.
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

- **Bit 1, 0 – ISC01, ISC00: Interrupt Sense Control 0 Bit 1 and Bit 0**

The External Interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT0 pin that activate the interrupt are defined in Table 10-2. The value on the INT0 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 10-2. Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

10.2.2 EIMSK – External Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
0x1D (0x3D)	–	–	–	–	–	–	INT1	INT0	EIMSK
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..2 – Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 1 – INT1: External Interrupt Request 1 Enable**

When the INT1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control1 bits 1/0 (ISC11 and ISC10) in the External Interrupt Control Register A (EICRA) define whether the external interrupt is activated on rising and/or falling edge of the INT1 pin or level sensed. Activity on the pin will cause an interrupt request even if INT1 is configured as an output. The corresponding interrupt of External Interrupt Request 1 is executed from the INT1 Interrupt Vector.

- **Bit 0 – INT0: External Interrupt Request 0 Enable**

When the INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control0 bits 1/0 (ISC01 and ISC00) in the External Interrupt Control Register A (EICRA) define whether the external interrupt is activated on rising and/or falling edge of the INT0 pin or level sensed. Activity on the pin will cause an interrupt request even if INT0 is configured as an output. The corresponding interrupt of External Interrupt Request 0 is executed from the INT0 Interrupt Vector.

10.2.3 EIFR – External Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x1C (0x3C)	–	–	–	–	–	–	INTF1	INTF0	EIFR
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..2 – Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 1 – INTF1: External Interrupt Flag 1**

When an edge or logic change on the INT1 pin triggers an interrupt request, INTF1 becomes set (one). If the I-bit in SREG and the INT1 bit in EIMSK are set (one), the MCU will jump to the corresponding Interrupt Vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it. This flag is always cleared when INT1 is configured as a level interrupt.

- **Bit 0 – INTF0: External Interrupt Flag 0**

When an edge or logic change on the INT0 pin triggers an interrupt request, INTF0 becomes set (one). If the I-bit in SREG and the INT0 bit in EIMSK are set (one), the MCU will jump to the corresponding Interrupt Vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it. This flag is always cleared when INT0 is configured as a level interrupt.

10.2.4 PCICR – Pin Change Interrupt Control Register

Bit	7	6	5	4	3	2	1	0	
(0x68)	–	–	–	–	–	PCIE2	PCIE1	PCIE0	PCICR
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..3 - Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 2 - PCIE2: Pin Change Interrupt Enable 2**

When the PCIE2 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), pin change interrupt 2 is enabled. Any change on any enabled PCINT23..16 pin will cause an interrupt. The corresponding interrupt of Pin Change Interrupt Request is executed from the PCI2 Interrupt Vector. PCINT23..16 pins are enabled individually by the PCMSK2 Register.

- **Bit 1 - PCIE1: Pin Change Interrupt Enable 1**

When the PCIE1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), pin change interrupt 1 is enabled. Any change on any enabled PCINT14..8 pin will cause an interrupt. The corresponding interrupt of Pin Change Interrupt Request is executed from the PCI1 Interrupt Vector. PCINT14..8 pins are enabled individually by the PCMSK1 Register.

- **Bit 0 - PCIE0: Pin Change Interrupt Enable 0**

When the PCIE0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), pin change interrupt 0 is enabled. Any change on any enabled PCINT7..0 pin will cause an interrupt. The corresponding interrupt of Pin Change Interrupt Request is executed from the PCI0 Interrupt Vector. PCINT7..0 pins are enabled individually by the PCMSK0 Register.

10.2.5 PCIFR – Pin Change Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x1B (0x3B)	–	–	–	–	–	PCIF2	PCIF1	PCIF0	PCIFR
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..3 - Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 2 - PCIF2: Pin Change Interrupt Flag 2**

When a logic change on any PCINT23..16 pin triggers an interrupt request, PCIF2 becomes set (one). If the I-bit in SREG and the PCIE2 bit in PCICR are set (one), the MCU will jump to the corresponding Interrupt Vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

- **Bit 1 - PCIF1: Pin Change Interrupt Flag 1**

When a logic change on any PCINT14..8 pin triggers an interrupt request, PCIF1 becomes set (one). If the I-bit in SREG and the PCIE1 bit in PCICR are set (one), the MCU will jump to the corresponding Interrupt Vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

- **Bit 0 - PCIF0: Pin Change Interrupt Flag 0**

When a logic change on any PCINT7..0 pin triggers an interrupt request, PCIF0 becomes set (one). If the I-bit in SREG and the PCIE0 bit in PCICR are set (one), the MCU will jump to the corresponding Interrupt Vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it.

10.2.6 PCMSK2 – Pin Change Mask Register 2

Bit	7	6	5	4	3	2	1	0	
(0x6D)	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16	PCMSK2
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..0 – PCINT23..16: Pin Change Enable Mask 23..16**

Each PCINT23..16-bit selects whether pin change interrupt is enabled on the corresponding I/O pin. If PCINT23..16 is set and the PCIE2 bit in PCICR is set, pin change interrupt is enabled on the corresponding I/O pin. If PCINT23..16 is cleared, pin change interrupt on the corresponding I/O pin is disabled.

10.2.7 PCMSK1 – Pin Change Mask Register 1

Bit	7	6	5	4	3	2	1	0	
(0x6C)	–	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8	PCMSK1
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – Res: Reserved Bit**

This bit is an unused bit in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 6..0 – PCINT14..8: Pin Change Enable Mask 14..8**

Each PCINT14..8-bit selects whether pin change interrupt is enabled on the corresponding I/O pin. If PCINT14..8 is set and the PCIE1 bit in PCICR is set, pin change interrupt is enabled on the corresponding I/O pin. If PCINT14..8 is cleared, pin change interrupt on the corresponding I/O pin is disabled.

10.2.8 PCMSK0 – Pin Change Mask Register 0

Bit	7	6	5	4	3	2	1	0	
(0x6B)	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	PCMSK0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..0 – PCINT7..0: Pin Change Enable Mask 7..0**

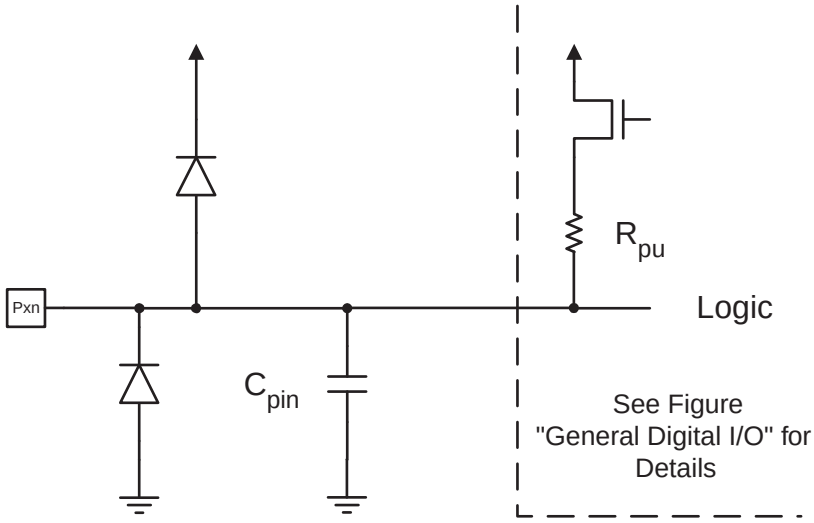
Each PCINT7..0 bit selects whether pin change interrupt is enabled on the corresponding I/O pin. If PCINT7..0 is set and the PCIE0 bit in PCICR is set, pin change interrupt is enabled on the corresponding I/O pin. If PCINT7..0 is cleared, pin change interrupt on the corresponding I/O pin is disabled.

11. I/O-Ports

11.1 Overview

All AVR ports have true Read-Modify-Write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies when changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input). Each output buffer has symmetrical drive characteristics with both high sink and source capability. The pin driver is strong enough to drive LED displays directly. All port pins have individually selectable pull-up resistors with a supply-voltage invariant resistance. All I/O pins have protection diodes to both V_{CC} and Ground as indicated in Figure 11-1. Refer to ”Electrical Characteristics” on page 313 for a complete list of parameters.

Figure 11-1. I/O Pin Equivalent Schematic



All registers and bit references in this section are written in general form. A lower case “x” represents the numbering letter for the port, and a lower case “n” represents the bit number. However, when using the register or bit defines in a program, the precise form must be used. For example, PORTB3 for bit no. 3 in Port B, here documented generally as PORTxn. The physical I/O Registers and bit locations are listed in ”Register Description” on page 92.

Three I/O memory address locations are allocated for each port, one each for the Data Register – PORTx, Data Direction Register – DDRx, and the Port Input Pins – PINx. The Port Input Pins I/O location is read only, while the Data Register and the Data Direction Register are read/write. However, writing a logic one to a bit in the PINx Register, will result in a toggle in the corresponding bit in the Data Register. In addition, the Pull-up Disable – PUD bit in MCUCR disables the pull-up function for all pins in all ports when set.

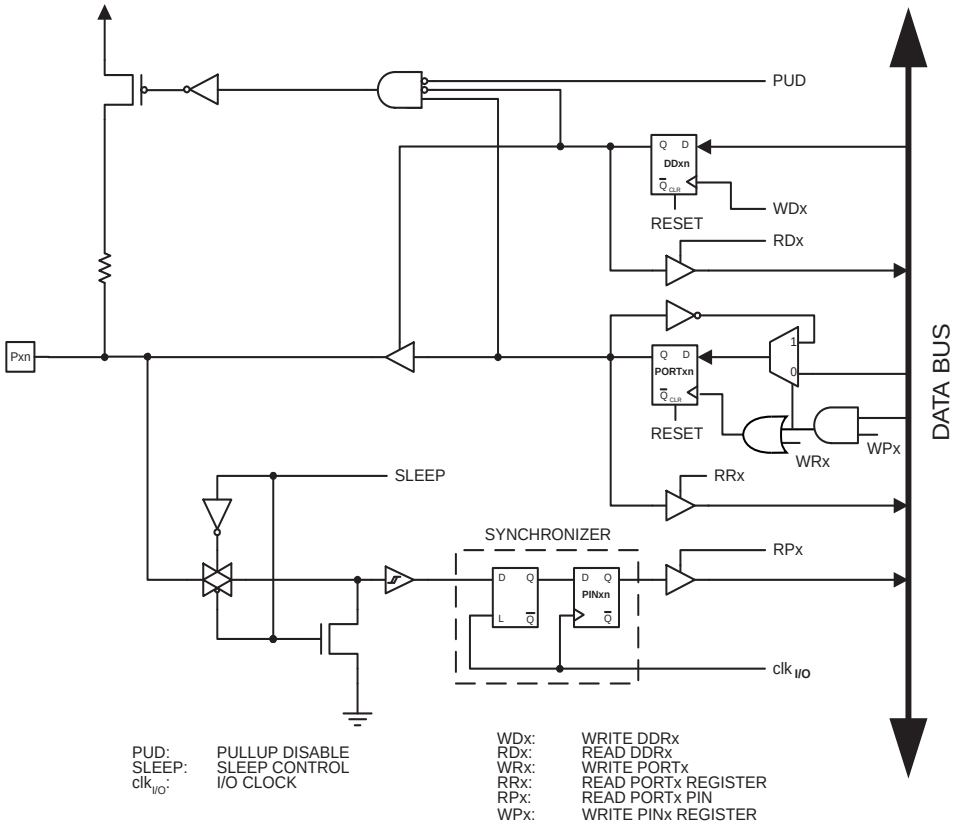
Using the I/O port as General Digital I/O is described in ”Ports as General Digital I/O” on page 76. Most port pins are multiplexed with alternate functions for the peripheral features on the device. How each alternate function interferes with the port pin is described in ”Alternate Port Functions” on page 80. Refer to the individual module sections for a full description of the alternate functions.

Note that enabling the alternate function of some of the port pins does not affect the use of the other pins in the port as general digital I/O.

11.2 Ports as General Digital I/O

The ports are bi-directional I/O ports with optional internal pull-ups. Figure 11-2 shows a functional description of one I/O-port pin, here generically called Pxn.

Figure 11-2. General Digital I/O(1)



Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

11.2.1 Configuring the Pin

Each port pin consists of three register bits: DDxn, PORTxn, and PINxn. As shown in "Register Description" on page 92, the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx Register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activated. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin. The port pins are tri-stated when reset condition becomes active, even if no clocks are running.

If PORTxn is written logic one when the pin is configured as an output pin, the port pin is driven high (one). If PORTxn is written logic zero when the pin is configured as an output pin, the port pin is driven low (zero).

11.2.2 Toggling the Pin

Writing a logic one to PINxn toggles the value of PORTxn, independent on the value of DDRxn. Note that the SBI instruction can be used to toggle one single bit in a port.

11.2.3 Switching Between Input and Output

When switching between tri-state ({DDxn, PORTxn} = 0b00) and output high ({DDxn, PORTxn} = 0b11), an intermediate state with either pull-up enabled {DDxn, PORTxn} = 0b01) or output low ({DDxn, PORTxn} = 0b10) must occur. Normally, the pull-up enabled state is fully acceptable, as a high-impedance environment will not notice the difference between a strong high driver and a pull-up. If this is not the case, the PUD bit in the MCUCR Register can be set to disable all pull-ups in all ports.

Switching between input with pull-up and output low generates the same problem. The user must use either the tri-state ({DDxn, PORTxn} = 0b00) or the output high state ({DDxn, PORTxn} = 0b11) as an intermediate step.

Table 11-1 summarizes the control signals for the pin value.

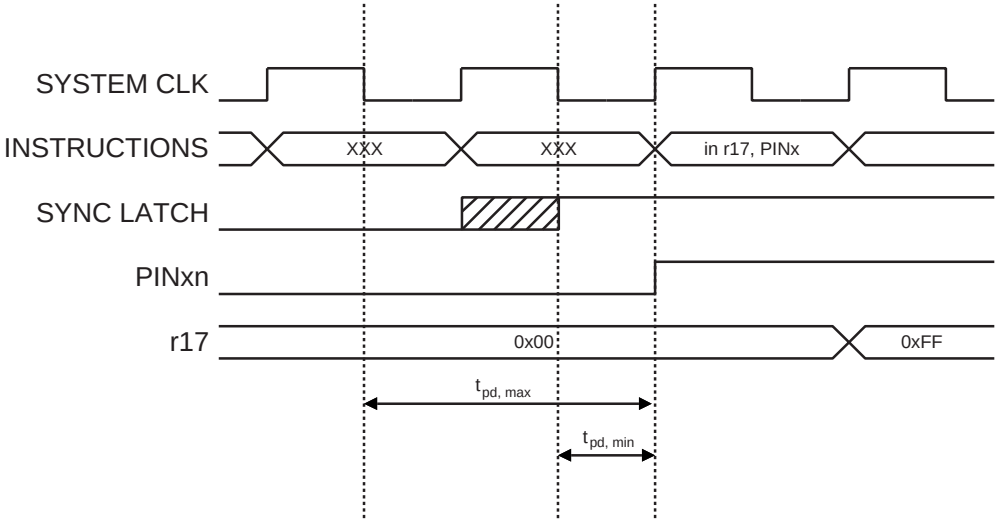
Table 11-1. Port Pin Configurations

DDxn	PORTxn	PUD (in MCUCR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

11.2.4 Reading the Pin Value

Independent of the setting of Data Direction bit DDxn, the port pin can be read through the PINxn Register bit. As shown in Figure 11-2, the PINxn Register bit and the preceding latch constitute a synchronizer. This is needed to avoid metastability if the physical pin changes value near the edge of the internal clock, but it also introduces a delay. Figure 11-3 shows a timing diagram of the synchronization when reading an externally applied pin value. The maximum and minimum propagation delays are denoted $t_{pd,max}$ and $t_{pd,min}$ respectively.

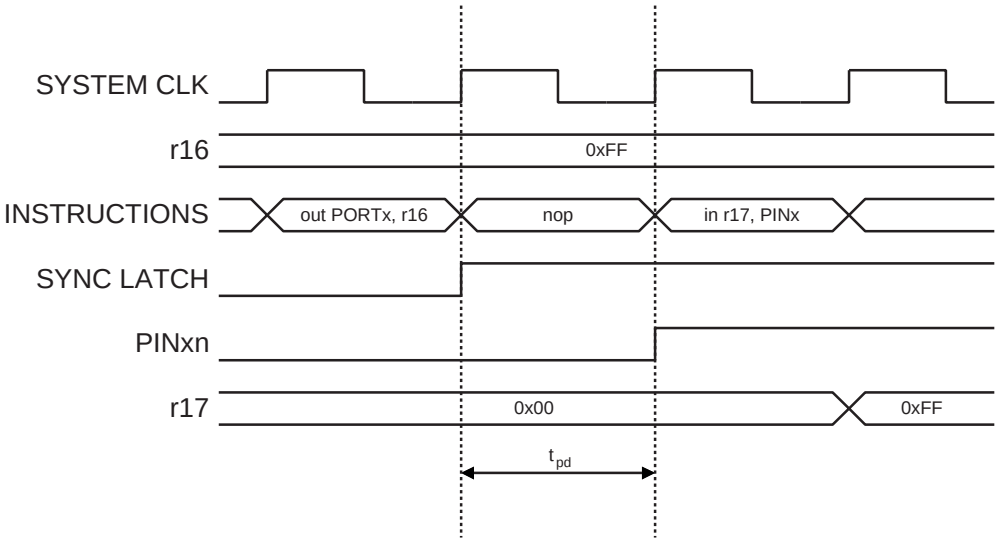
Figure 11-3. Synchronization when Reading an Externally Applied Pin value



Consider the clock period starting shortly after the first falling edge of the system clock. The latch is closed when the clock is low, and goes transparent when the clock is high, as indicated by the shaded region of the “SYNC LATCH” signal. The signal value is latched when the system clock goes low. It is clocked into the PINxn Register at the succeeding positive clock edge. As indicated by the two arrows tpd,max and tpd,min, a single signal transition on the pin will be delayed between ½ and 1½ system clock period depending upon the time of assertion.

When reading back a software assigned pin value, a nop instruction must be inserted as indicated in Figure 11-4. The out instruction sets the “SYNC LATCH” signal at the positive edge of the clock. In this case, the delay tpd through the synchronizer is 1 system clock period.

Figure 11-4. Synchronization when Reading a Software Assigned Pin Value



The following code example shows how to set port B pins 0 and 1 high, 2 and 3 low, and define the port pins from 4 to 7 as input with pull-ups assigned to port pins 6 and 7. The resulting pin values are read back again, but as previously discussed, a nop instruction is included to be able to read back the value recently assigned to some of the pins.

Assembly Code Example[1]

```
...
; Define pull-ups and set outputs high
; Define directions for port pins
ldi  r16,(1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0)
ldi  r17,(1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0)
out  PORTB,r16
out  DDRB,r17
; Insert nop for synchronization
nop
; Read port pins
in   r16,PINB
...
```

C Code Example

```
unsigned char i;
...
/* Define pull-ups and set outputs high */
/* Define directions for port pins */
PORTB = (1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0);
DDRB = (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0);
/* Insert nop for synchronization*/
__no_operation();
/* Read port pins */
i = PINB;
...
```

Note: 1. For the assembly program, two temporary registers are used to minimize the time from pull-ups are set on pins 0, 1, 6, and 7, until the direction bits are correctly set, defining bit 2 and 3 as low and redefining bits 0 and 1 as strong high drivers.

11.2.5 Digital Input Enable and Sleep Modes

As shown in Figure 11-2, the digital input signal can be clamped to ground at the input of the Schmitt Trigger. The signal denoted SLEEP in the figure, is set by the MCU Sleep Controller in Power-down mode, Power-save mode, and Standby mode to avoid high power consumption if some input signals are left floating, or have an analog signal level close to $V_{CC}/2$.

SLEEP is overridden for port pins enabled as external interrupt pins. If the external interrupt request is not enabled, SLEEP is active also for these pins. SLEEP is also overridden by various other alternate functions as described in ”Alternate Port Functions” on page 80.

If a logic high level (“one”) is present on an asynchronous external interrupt pin configured as “Interrupt on Rising Edge, Falling Edge, or Any Logic Change on Pin” while the external interrupt is *not* enabled, the corresponding External Interrupt Flag will be set when resuming from the above mentioned Sleep mode, as the clamping in these sleep mode produces the requested logic change.

11.2.6 Unconnected Pins

If some pins are unused, it is recommended to ensure that these pins have a defined level. Even though most of the digital inputs are disabled in the deep sleep modes as described above, float-

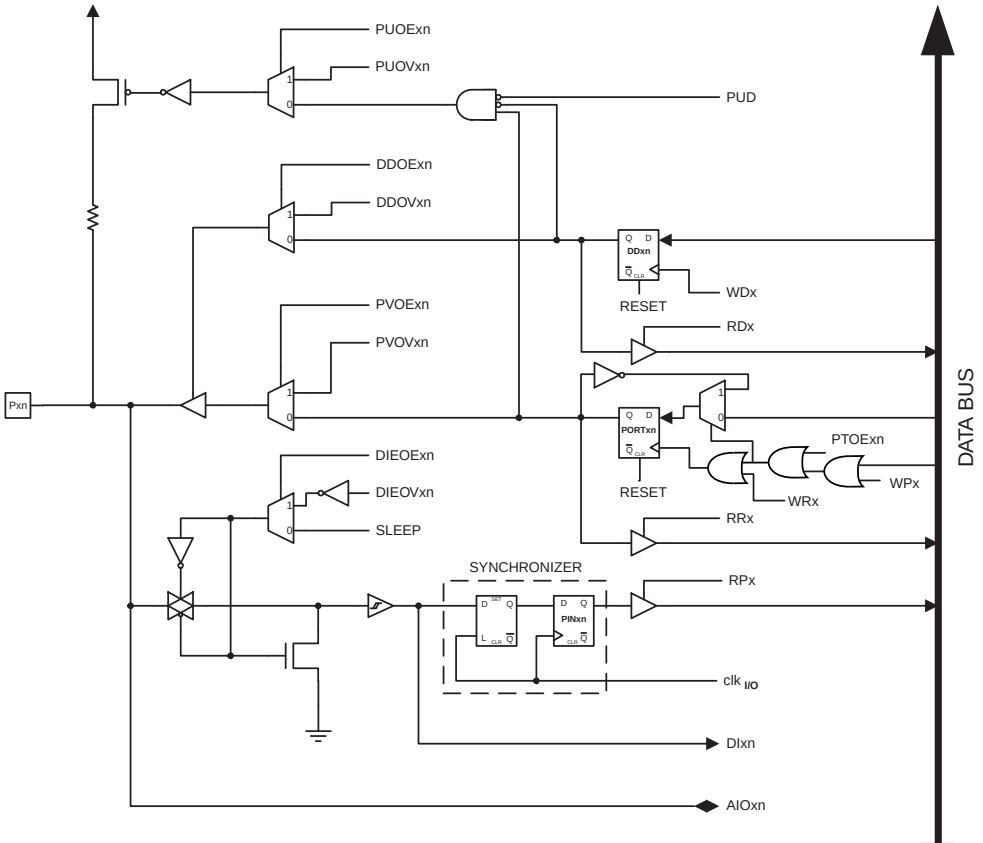
ing inputs should be avoided to reduce current consumption in all other modes where the digital inputs are enabled (Reset, Active mode and Idle mode).

The simplest method to ensure a defined level of an unused pin, is to enable the internal pull-up. In this case, the pull-up will be disabled during reset. If low power consumption during reset is important, it is recommended to use an external pull-up or pull-down. Connecting unused pins directly to V_{CC} or GND is not recommended, since this may cause excessive currents if the pin is accidentally configured as an output.

11.3 Alternate Port Functions

Most port pins have alternate functions in addition to being general digital I/Os. Figure 11-5 shows how the port pin control signals from the simplified Figure 11-2 on page 76 can be overridden by alternate functions. The overriding signals may not be present in all port pins, but the figure serves as a generic description applicable to all port pins in the AVR microcontroller family.

Figure 11-5. Alternate Port Functions[1]



PUOExn:	Pxn PULL-UP OVERRIDE ENABLE	PUD:	PULLUP DISABLE
PUOVxn:	Pxn PULL-UP OVERRIDE VALUE	WDx:	WRITE DDRx
DDOExn:	Pxn DATA DIRECTION OVERRIDE ENABLE	RDx:	READ DDRx
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE	RRx:	READ PORTx REGISTER
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE	WRx:	WRITE PORTx
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE	RPx:	READ PORTx PIN
DIEOExn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE ENABLE	WPx:	WRITE PINx
DIEOVxn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE VALUE	$clk_{I/O}$:	I/O CLOCK
SLEEP:	SLEEP CONTROL	DIxn:	DIGITAL INPUT PIN n ON PORTx
PTOExn:	Pxn, PORT TOGGLE OVERRIDE ENABLE	AIOxn:	ANALOG INPUT/OUTPUT PIN n ON PORTx

Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports. All other signals are unique for each pin.

Table 11-2 summarizes the function of the overriding signals. The pin and port indexes from Figure 11-5 on page 80 are not shown in the succeeding tables. The overriding signals are generated internally in the modules having the alternate function.

Table 11-2. Generic Description of Overriding Signals for Alternate Functions

Signal Name	Full Name	Description
PUOE	Pull-up Override Enable	If this signal is set, the pull-up enable is controlled by the PUOV signal. If this signal is cleared, the pull-up is enabled when {DDxn, PORTxn, PUD} = 0b010.
PUOV	Pull-up Override Value	If PUOE is set, the pull-up is enabled/disabled when PUOV is set/cleared, regardless of the setting of the DDxn, PORTxn, and PUD Register bits.
DDOE	Data Direction Override Enable	If this signal is set, the Output Driver Enable is controlled by the DDOV signal. If this signal is cleared, the Output driver is enabled by the DDxn Register bit.
DDOV	Data Direction Override Value	If DDOE is set, the Output Driver is enabled/disabled when DDOV is set/cleared, regardless of the setting of the DDxn Register bit.
PVOE	Port Value Override Enable	If this signal is set and the Output Driver is enabled, the port value is controlled by the PVOV signal. If PVOE is cleared, and the Output Driver is enabled, the port Value is controlled by the PORTxn Register bit.
PVOV	Port Value Override Value	If PVOE is set, the port value is set to PVOV, regardless of the setting of the PORTxn Register bit.
PTOE	Port Toggle Override Enable	If PTOE is set, the PORTxn Register bit is inverted.
DIEOE	Digital Input Enable Override Enable	If this bit is set, the Digital Input Enable is controlled by the DIEOV signal. If this signal is cleared, the Digital Input Enable is determined by MCU state (Normal mode, sleep mode).
DIEOV	Digital Input Enable Override Value	If DIEOE is set, the Digital Input is enabled/disabled when DIEOV is set/cleared, regardless of the MCU state (Normal mode, sleep mode).
DI	Digital Input	This is the Digital Input to alternate functions. In the figure, the signal is connected to the output of the Schmitt Trigger but before the synchronizer. Unless the Digital Input is used as a clock source, the module with the alternate function will use its own synchronizer.
AIO	Analog Input/Output	This is the Analog Input/output to/from alternate functions. The signal is connected directly to the pad, and can be used bi-directionally.

The following subsections shortly describe the alternate functions for each port, and relate the overriding signals to the alternate function. Refer to the alternate function description for further details.

11.3.1 Alternate Functions of Port B

The Port B pins with alternate functions are shown in Table 11-3.

Table 11-3. Port B Pins Alternate Functions

Port Pin	Alternate Functions
PB7	XTAL2 (Chip Clock Oscillator pin 2) TOSC2 (Timer Oscillator pin 2) PCINT7 (Pin Change Interrupt 7)
PB6	XTAL1 (Chip Clock Oscillator pin 1 or External clock input) TOSC1 (Timer Oscillator pin 1) PCINT6 (Pin Change Interrupt 6)
PB5	SCK (SPI Bus Master clock Input) PCINT5 (Pin Change Interrupt 5)
PB4	MISO (SPI Bus Master Input/Slave Output) PCINT4 (Pin Change Interrupt 4)
PB3	MOSI (SPI Bus Master Output/Slave Input) OC2A (Timer/Counter2 Output Compare Match A Output) PCINT3 (Pin Change Interrupt 3)
PB2	$\overline{SS}$ (SPI Bus Master Slave select) OC1B (Timer/Counter1 Output Compare Match B Output) PCINT2 (Pin Change Interrupt 2)
PB1	OC1A (Timer/Counter1 Output Compare Match A Output) PCINT1 (Pin Change Interrupt 1)
PB0	ICP1 (Timer/Counter1 Input Capture Input) CLKO (Divided System Clock Output) PCINT0 (Pin Change Interrupt 0)

The alternate pin configuration is as follows:

- **XTAL2/TOSC2/PCINT7 – Port B, Bit 7**

XTAL2: Chip clock Oscillator pin 2. Used as clock pin for crystal Oscillator or Low-frequency crystal Oscillator. When used as a clock pin, the pin can not be used as an I/O pin.

TOSC2: Timer Oscillator pin 2. Used only if internal calibrated RC Oscillator is selected as chip clock source, and the asynchronous timer is enabled by the correct setting in ASSR. When the AS2 bit in ASSR is set (one) and the EXCLK bit is cleared (zero) to enable asynchronous clocking of Timer/Counter2 using the Crystal Oscillator, pin PB7 is disconnected from the port, and becomes the inverting output of the Oscillator amplifier. In this mode, a crystal Oscillator is connected to this pin, and the pin cannot be used as an I/O pin.

PCINT7: Pin Change Interrupt source 7. The PB7 pin can serve as an external interrupt source.

If PB7 is used as a clock pin, DDB7, PORTB7 and PINB7 will all read 0.

- **XTAL1/TOSC1/PCINT6 – Port B, Bit 6**

XTAL1: Chip clock Oscillator pin 1. Used for all chip clock sources except internal calibrated RC Oscillator. When used as a clock pin, the pin can not be used as an I/O pin.

TOSC1: Timer Oscillator pin 1. Used only if internal calibrated RC Oscillator is selected as chip clock source, and the asynchronous timer is enabled by the correct setting in ASSR. When the

AS2 bit in ASSR is set (one) to enable asynchronous clocking of Timer/Counter2, pin PB6 is disconnected from the port, and becomes the input of the inverting Oscillator amplifier. In this mode, a crystal Oscillator is connected to this pin, and the pin can not be used as an I/O pin.

PCINT6: Pin Change Interrupt source 6. The PB6 pin can serve as an external interrupt source.

If PB6 is used as a clock pin, DDB6, PORTB6 and PINB6 will all read 0.

- **SCK/PCINT5 – Port B, Bit 5**

SCK: Master Clock output, Slave Clock input pin for SPI channel. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB5. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB5. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB5 bit.

PCINT5: Pin Change Interrupt source 5. The PB5 pin can serve as an external interrupt source.

- **MISO/PCINT4 – Port B, Bit 4**

MISO: Master Data input, Slave Data output pin for SPI channel. When the SPI is enabled as a Master, this pin is configured as an input regardless of the setting of DDB4. When the SPI is enabled as a Slave, the data direction of this pin is controlled by DDB4. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB4 bit.

PCINT4: Pin Change Interrupt source 4. The PB4 pin can serve as an external interrupt source.

- **MOSI/OC2/PCINT3 – Port B, Bit 3**

MOSI: SPI Master Data output, Slave Data input for SPI channel. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB3. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB3. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB3 bit.

OC2, Output Compare Match Output: The PB3 pin can serve as an external output for the Timer/Counter2 Compare Match. The PB3 pin has to be configured as an output (DDB3 set (one)) to serve this function. The OC2 pin is also the output pin for the PWM mode timer function.

PCINT3: Pin Change Interrupt source 3. The PB3 pin can serve as an external interrupt source.

- **$\overline{SS}$/OC1B/PCINT2 – Port B, Bit 2**

$\overline{SS}$: Slave Select input. When the SPI is enabled as a Slave, this pin is configured as an input regardless of the setting of DDB2. As a Slave, the SPI is activated when this pin is driven low. When the SPI is enabled as a Master, the data direction of this pin is controlled by DDB2. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB2 bit.

OC1B, Output Compare Match output: The PB2 pin can serve as an external output for the Timer/Counter1 Compare Match B. The PB2 pin has to be configured as an output (DDB2 set (one)) to serve this function. The OC1B pin is also the output pin for the PWM mode timer function.

PCINT2: Pin Change Interrupt source 2. The PB2 pin can serve as an external interrupt source.

- **OC1A/PCINT1 – Port B, Bit 1**

OC1A, Output Compare Match output: The PB1 pin can serve as an external output for the Timer/Counter1 Compare Match A. The PB1 pin has to be configured as an output (DDB1 set

(one)) to serve this function. The OC1A pin is also the output pin for the PWM mode timer function.

PCINT1: Pin Change Interrupt source 1. The PB1 pin can serve as an external interrupt source.

- **ICP1/CLKO/PCINT0 – Port B, Bit 0**

ICP1, Input Capture Pin: The PB0 pin can act as an Input Capture Pin for Timer/Counter1.

CLKO, Divided System Clock: The divided system clock can be output on the PB0 pin. The divided system clock will be output if the CKOUT Fuse is programmed, regardless of the PORTB0 and DDB0 settings. It will also be output during reset.

PCINT0: Pin Change Interrupt source 0. The PB0 pin can serve as an external interrupt source.

Table 11-4 and Table 11-5 on page 85 relate the alternate functions of Port B to the overriding signals shown in Figure 11-5 on page 80. SPI MSTR INPUT and SPI SLAVE OUTPUT constitute the MISO signal, while MOSI is divided into SPI MSTR OUTPUT and SPI SLAVE INPUT.

Table 11-4. Overriding Signals for Alternate Functions in PB7..PB4

Signal Name	PB7/XTAL2/ TOSC2/PCINT7(1)	PB6/XTAL1/ TOSC1/PCINT6(1)	PB5/SCK/ PCINT5	PB4/MISO/ PCINT4
PUOE	$\overline{\text{INTRC}}$ • $\overline{\text{EXTCK}}$+ AS2	$\overline{\text{INTRC}}$ + AS2	SPE • $\overline{\text{MSTR}}$	SPE • MSTR
PUOV	0	0	PORTB5 • $\overline{\text{PUD}}$	PORTB4 • $\overline{\text{PUD}}$
DDOE	$\overline{\text{INTRC}}$ • $\overline{\text{EXTCK}}$+ AS2	$\overline{\text{INTRC}}$ + AS2	SPE • $\overline{\text{MSTR}}$	SPE • MSTR
DDOV	0	0	0	0
PVOE	0	0	SPE • MSTR	SPE • $\overline{\text{MSTR}}$
PVOV	0	0	SCK OUTPUT	SPI SLAVE OUTPUT
DIEOE	$\overline{\text{INTRC}}$ • $\overline{\text{EXTCK}}$ + AS2 + PCINT7 • PCIE0	$\overline{\text{INTRC}}$ + AS2 + PCINT6 • PCIE0	PCINT5 • PCIE0	PCINT4 • PCIE0
DIEOV	(INTRC + EXTCK) • $\overline{\text{AS2}}$	INTRC • $\overline{\text{AS2}}$	1	1
DI	PCINT7 INPUT	PCINT6 INPUT	PCINT5 INPUT SCK INPUT	PCINT4 INPUT SPI MSTR INPUT
AIO	Oscillator Output	Oscillator/Clock Input	–	–

Notes: 1. INTRC means that one of the internal RC Oscillators are selected (by the CKSEL fuses), EXTCK means that external clock is selected (by the CKSEL fuses)

Table 11-5. Overriding Signals for Alternate Functions in PB3..PB0

Signal Name	PB3/MOSI/ OC2/PCINT3	PB2/$\overline{SS}$/ OC1B/PCINT2	PB1/OC1A/ PCINT1	PB0/ICP1/ PCINT0
PUOE	SPE • $\overline{MSTR}$	SPE • $\overline{MSTR}$	0	0
PUOV	PORTB3 • $\overline{PUD}$	PORTB2 • $\overline{PUD}$	0	0
DDOE	SPE • $\overline{MSTR}$	SPE • $\overline{MSTR}$	0	0
DDOV	0	0	0	0
PVOE	SPE • MSTR + OC2A ENABLE	OC1B ENABLE	OC1A ENABLE	0
PVOV	SPI MSTR OUTPUT + OC2A	OC1B	OC1A	0
DIEOE	PCINT3 • PCIE0	PCINT2 • PCIE0	PCINT1 • PCIE0	PCINT0 • PCIE0
DIEOV	1	1	1	1
DI	PCINT3 INPUT SPI SLAVE INPUT	PCINT2 INPUT SPI $\overline{SS}$	PCINT1 INPUT	PCINT0 INPUT ICP1 INPUT
AIO	–	–	–	–

11.3.2 Alternate Functions of Port C

The Port C pins with alternate functions are shown in Table 11-6.

Table 11-6. Port C Pins Alternate Functions

Port Pin	Alternate Function
PC6	$\overline{RESET}$ (Reset pin) PCINT14 (Pin Change Interrupt 14)
PC5	ADC5 (ADC Input Channel 5) SCL (2-wire Serial Bus Clock Line) PCINT13 (Pin Change Interrupt 13)
PC4	ADC4 (ADC Input Channel 4) SDA (2-wire Serial Bus Data Input/Output Line) PCINT12 (Pin Change Interrupt 12)
PC3	ADC3 (ADC Input Channel 3) PCINT11 (Pin Change Interrupt 11)
PC2	ADC2 (ADC Input Channel 2) PCINT10 (Pin Change Interrupt 10)
PC1	ADC1 (ADC Input Channel 1) PCINT9 (Pin Change Interrupt 9)
PC0	ADC0 (ADC Input Channel 0) PCINT8 (Pin Change Interrupt 8)

The alternate pin configuration is as follows:

- **$\overline{\text{RESET}}$/PCINT14 – Port C, Bit 6**

$\overline{\text{RESET}}$, Reset pin: When the RSTDISBL Fuse is programmed, this pin functions as a normal I/O pin, and the part will have to rely on Power-on Reset and Brown-out Reset as its reset sources. When the RSTDISBL Fuse is unprogrammed, the reset circuitry is connected to the pin, and the pin can not be used as an I/O pin.

If PC6 is used as a reset pin, DDC6, PORTC6 and PINC6 will all read 0.

PCINT14: Pin Change Interrupt source 14. The PC6 pin can serve as an external interrupt source.

- **SCL/ADC5/PCINT13 – Port C, Bit 5**

SCL, 2-wire Serial Interface Clock: When the TWEN bit in TWCR is set (one) to enable the 2-wire Serial Interface, pin PC5 is disconnected from the port and becomes the Serial Clock I/O pin for the 2-wire Serial Interface. In this mode, there is a spike filter on the pin to suppress spikes shorter than 50 ns on the input signal, and the pin is driven by an open drain driver with slew-rate limitation.

PC5 can also be used as ADC input Channel 5. Note that ADC input channel 5 uses digital power.

PCINT13: Pin Change Interrupt source 13. The PC5 pin can serve as an external interrupt source.

- **SDA/ADC4/PCINT12 – Port C, Bit 4**

SDA, 2-wire Serial Interface Data: When the TWEN bit in TWCR is set (one) to enable the 2-wire Serial Interface, pin PC4 is disconnected from the port and becomes the Serial Data I/O pin for the 2-wire Serial Interface. In this mode, there is a spike filter on the pin to suppress spikes shorter than 50 ns on the input signal, and the pin is driven by an open drain driver with slew-rate limitation.

PC4 can also be used as ADC input Channel 4. Note that ADC input channel 4 uses digital power.

PCINT12: Pin Change Interrupt source 12. The PC4 pin can serve as an external interrupt source.

- **ADC3/PCINT11 – Port C, Bit 3**

PC3 can also be used as ADC input Channel 3. Note that ADC input channel 3 uses analog power.

PCINT11: Pin Change Interrupt source 11. The PC3 pin can serve as an external interrupt source.

- **ADC2/PCINT10 – Port C, Bit 2**

PC2 can also be used as ADC input Channel 2. Note that ADC input channel 2 uses analog power.

PCINT10: Pin Change Interrupt source 10. The PC2 pin can serve as an external interrupt source.

- **ADC1/PCINT9 – Port C, Bit 1**

PC1 can also be used as ADC input Channel 1. Note that ADC input channel 1 uses analog power.

PCINT9: Pin Change Interrupt source 9. The PC1 pin can serve as an external interrupt source.

- **ADC0/PCINT8 – Port C, Bit 0**

PC0 can also be used as ADC input Channel 0. Note that ADC input channel 0 uses analog power.

PCINT8: Pin Change Interrupt source 8. The PC0 pin can serve as an external interrupt source.

Table 11-7 and Table 11-8 relate the alternate functions of Port C to the overriding signals shown in Figure 11-5 on page 80.

Table 11-7. Overriding Signals for Alternate Functions in PC6..PC4[1]

Signal Name	PC6/$\overline{\text{RESET}}$/PCINT14	PC5/SCL/ADC5/PCINT13	PC4/SDA/ADC4/PCINT12
PUOE	RSTDISBL	TWEN	TWEN
PUOV	1	PORTC5 • $\overline{\text{PUD}}$	PORTC4 • $\overline{\text{PUD}}$
DDOE	RSTDISBL	TWEN	TWEN
DDOV	0	SCL_OUT	SDA_OUT
PVOE	0	TWEN	TWEN
PVOV	0	0	0
DIEOE	RSTDISBL + PCINT14 • PCIE1	PCINT13 • PCIE1 + ADC5D	PCINT12 • PCIE1 + ADC4D
DIEOV	RSTDISBL	PCINT13 • PCIE1	PCINT12 • PCIE1
DI	PCINT14 INPUT	PCINT13 INPUT	PCINT12 INPUT
AIO	RESET INPUT	ADC5 INPUT / SCL INPUT	ADC4 INPUT / SDA INPUT

Note: 1. When enabled, the 2-wire Serial Interface enables slew-rate controls on the output pins PC4 and PC5. This is not shown in the figure. In addition, spike filters are connected between the AIO outputs shown in the port figure and the digital logic of the TWI module.

Table 11-8. Overriding Signals for Alternate Functions in PC3..PC0

Signal Name	PC3/ADC3/ PCINT11	PC2/ADC2/ PCINT10	PC1/ADC1/ PCINT9	PC0/ADC0/ PCINT8
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	0	0
PVOV	0	0	0	0
DIEOE	PCINT11 • PCIE1 + ADC3D	PCINT10 • PCIE1 + ADC2D	PCINT9 • PCIE1 + ADC1D	PCINT8 • PCIE1 + ADC0D
DIEOV	PCINT11 • PCIE1	PCINT10 • PCIE1	PCINT9 • PCIE1	PCINT8 • PCIE1
DI	PCINT11 INPUT	PCINT10 INPUT	PCINT9 INPUT	PCINT8 INPUT
AIO	ADC3 INPUT	ADC2 INPUT	ADC1 INPUT	ADC0 INPUT

11.3.3 Alternate Functions of Port D

The Port D pins with alternate functions are shown in Table 11-9.

Table 11-9. Port D Pins Alternate Functions

Port Pin	Alternate Function
PD7	AIN1 (Analog Comparator Negative Input) PCINT23 (Pin Change Interrupt 23)
PD6	AIN0 (Analog Comparator Positive Input) OC0A (Timer/Counter0 Output Compare Match A Output) PCINT22 (Pin Change Interrupt 22)
PD5	T1 (Timer/Counter 1 External Counter Input) OC0B (Timer/Counter0 Output Compare Match B Output) PCINT21 (Pin Change Interrupt 21)
PD4	XCK (USART External Clock Input/Output) T0 (Timer/Counter 0 External Counter Input) PCINT20 (Pin Change Interrupt 20)
PD3	INT1 (External Interrupt 1 Input) OC2B (Timer/Counter2 Output Compare Match B Output) PCINT19 (Pin Change Interrupt 19)
PD2	INT0 (External Interrupt 0 Input) PCINT18 (Pin Change Interrupt 18)
PD1	TXD (USART Output Pin) PCINT17 (Pin Change Interrupt 17)
PD0	RXD (USART Input Pin) PCINT16 (Pin Change Interrupt 16)

The alternate pin configuration is as follows:

- **AIN1/OC2B/PCINT23 – Port D, Bit 7**

AIN1, Analog Comparator Negative Input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the Analog Comparator.

PCINT23: Pin Change Interrupt source 23. The PD7 pin can serve as an external interrupt source.

- **AIN0/OC0A/PCINT22 – Port D, Bit 6**

AIN0, Analog Comparator Positive Input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the Analog Comparator.

OC0A, Output Compare Match output: The PD6 pin can serve as an external output for the Timer/Counter0 Compare Match A. The PD6 pin has to be configured as an output (DDD6 set (one)) to serve this function. The OC0A pin is also the output pin for the PWM mode timer function.

PCINT22: Pin Change Interrupt source 22. The PD6 pin can serve as an external interrupt source.

- **T1/OC0B/PCINT21 – Port D, Bit 5**

T1, Timer/Counter1 counter source.

OC0B, Output Compare Match output: The PD5 pin can serve as an external output for the Timer/Counter0 Compare Match B. The PD5 pin has to be configured as an output (DDD5 set (one)) to serve this function. The OC0B pin is also the output pin for the PWM mode timer function.

PCINT21: Pin Change Interrupt source 21. The PD5 pin can serve as an external interrupt source.

- **XCK/T0/PCINT20 – Port D, Bit 4**

XCK, USART external clock.

T0, Timer/Counter0 counter source.

PCINT20: Pin Change Interrupt source 20. The PD4 pin can serve as an external interrupt source.

- **INT1/OC2B/PCINT19 – Port D, Bit 3**

INT1, External Interrupt source 1: The PD3 pin can serve as an external interrupt source.

OC2B, Output Compare Match output: The PD3 pin can serve as an external output for the Timer/Counter0 Compare Match B. The PD3 pin has to be configured as an output (DDD3 set (one)) to serve this function. The OC2B pin is also the output pin for the PWM mode timer function.

PCINT19: Pin Change Interrupt source 19. The PD3 pin can serve as an external interrupt source.

- **INT0/PCINT18 – Port D, Bit 2**

INT0, External Interrupt source 0: The PD2 pin can serve as an external interrupt source.

PCINT18: Pin Change Interrupt source 18. The PD2 pin can serve as an external interrupt source.

- **TXD/PCINT17 – Port D, Bit 1**

TXD, Transmit Data (Data output pin for the USART). When the USART Transmitter is enabled, this pin is configured as an output regardless of the value of DDD1.

PCINT17: Pin Change Interrupt source 17. The PD1 pin can serve as an external interrupt source.

- **RXD/PCINT16 – Port D, Bit 0**

RXD, Receive Data (Data input pin for the USART). When the USART Receiver is enabled this pin is configured as an input regardless of the value of DDD0. When the USART forces this pin to be an input, the pull-up can still be controlled by the PORTD0 bit.

PCINT16: Pin Change Interrupt source 16. The PD0 pin can serve as an external interrupt source.

Table 11-10 and Table 11-11 relate the alternate functions of Port D to the overriding signals shown in Figure 11-5 on page 80.

Table 11-10. Overriding Signals for Alternate Functions PD7..PD4

Signal Name	PD7/AIN1 /PCINT23	PD6/AIN0/ OC0A/PCINT22	PD5/T1/OC0B/ PCINT21	PD4/XCK/ T0/PCINT20
PUOE	0	0	0	0
PUO	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	OC0A ENABLE	OC0B ENABLE	UMSEL
PVOV	0	OC0A	OC0B	XCK OUTPUT
DIEOE	PCINT23 • PCIE2	PCINT22 • PCIE2	PCINT21 • PCIE2	PCINT20 • PCIE2
DIEOV	1	1	1	1
DI	PCINT23 INPUT	PCINT22 INPUT	PCINT21 INPUT T1 INPUT	PCINT20 INPUT XCK INPUT T0 INPUT
AIO	AIN1 INPUT	AIN0 INPUT	–	–

Table 11-11. Overriding Signals for Alternate Functions in PD3..PD0

Signal Name	PD3/OC2B/INT1/ PCINT19	PD2/INT0/ PCINT18	PD1/TXD/ PCINT17	PD0/RXD/ PCINT16
PUOE	0	0	TXEN	RXEN
PUO	0	0	0	PORTD0 • $\overline{\text{PUD}}$
DDOE	0	0	TXEN	RXEN
DDOV	0	0	1	0
PVOE	OC2B ENABLE	0	TXEN	0
PVOV	OC2B	0	TXD	0
DIEOE	INT1 ENABLE + PCINT19 • PCIE2	INT0 ENABLE + PCINT18 • PCIE1	PCINT17 • PCIE2	PCINT16 • PCIE2
DIEOV	1	1	1	1
DI	PCINT19 INPUT INT1 INPUT	PCINT18 INPUT INT0 INPUT	PCINT17 INPUT	PCINT16 INPUT RXD
AIO	–	–	–	–

11.4 Register Description

11.4.1 MCUCR – MCU Control Register

Bit	7	6	5	4	3	2	1	0	
0x35 (0x55)	–	BODS	BODSE	PUD	–	–	IVSEL	IVCE	MCUCR
Read/Write	R	R	R	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 4 – PUD: Pull-up Disable**

When this bit is written to one, the pull-ups in the I/O ports are disabled even if the DDxn and PORTxn Registers are configured to enable the pull-ups ({DDxn, PORTxn} = 0b01). See ”Configuring the Pin” on page 76 for more details about this feature.

11.4.2 PORTB – The Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

11.4.3 DDRB – The Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

11.4.4 PINB – The Port B Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

11.4.5 PORTC – The Port C Data Register

Bit	7	6	5	4	3	2	1	0	
0x08 (0x28)	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	PORTC
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

11.4.6 DDRC – The Port C Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x07 (0x27)	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

11.4.7 PINC – The Port C Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x06 (0x26)	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	PINC
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

11.4.8 PORTD – The Port D Data Register

Bit	7	6	5	4	3	2	1	0	
0x0B (0x2B)	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

11.4.9 DDRD – The Port D Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x0A (0x2A)	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

11.4.10 PIND – The Port D Input Pins Address

Bit	7	6	5	4	3	2	1	0	
0x09 (0x29)	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

12. 8-bit Timer/Counter0 with PWM

12.1 Features

- **Two Independent Output Compare Units**
- **Double Buffered Output Compare Registers**
- **Clear Timer on Compare Match (Auto Reload)**
- **Glitch Free, Phase Correct Pulse Width Modulator (PWM)**
- **Variable PWM Period**
- **Frequency Generator**
- **Three Independent Interrupt Sources (TOV0, OCF0A, and OCF0B)**

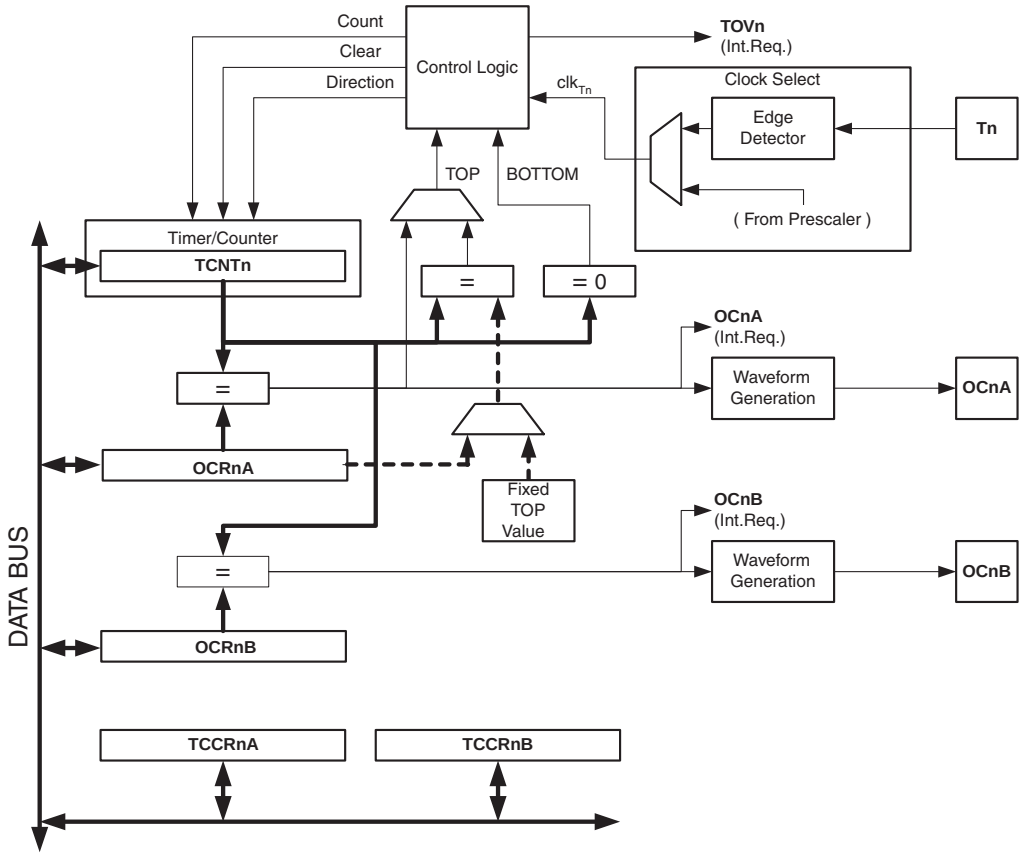
12.2 Overview

Timer/Counter0 is a general purpose 8-bit Timer/Counter module, with two independent Output Compare Units, and with PWM support. It allows accurate program execution timing (event management) and wave generation.

A simplified block diagram of the 8-bit Timer/Counter is shown in Figure 12-1. For the actual placement of I/O pins, refer to ”Pinout ATmega48P/88P/168P/328P” on page 2. CPU accessible I/O Registers, including I/O bits and I/O pins, are shown in bold. The device-specific I/O Register and bit locations are listed in the ”Register Description” on page 106.

The PRTIM0 bit in ”Minimizing Power Consumption” on page 42 must be written to zero to enable Timer/Counter0 module.

Figure 12-1. 8-bit Timer/Counter Block Diagram



12.2.1 Definitions

Many register and bit references in this section are written in general form. A lower case “n” replaces the Timer/Counter number, in this case 0. A lower case “x” replaces the Output Compare Unit, in this case Compare Unit A or Compare Unit B. However, when using the register or bit defines in a program, the precise form must be used, i.e., TCNT0 for accessing Timer/Counter0 counter value and so on.

The definitions in Table 12-1 are also used extensively throughout the document.

Table 12-1. Definitions

BOTTOM	The counter reaches the BOTTOM when it becomes 0x00.
MAX	The counter reaches its MAXimum when it becomes 0xFF (decimal 255).
TOP	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be the fixed value 0xFF (MAX) or the value stored in the OCR0A Register. The assignment is dependent on the mode of operation.

12.2.2 Registers

The Timer/Counter (TCNT0) and Output Compare Registers (OCR0A and OCR0B) are 8-bit registers. Interrupt request (abbreviated to Int.Req. in the figure) signals are all visible in the Timer Interrupt Flag Register (TIFR0). All interrupts are individually masked with the Timer Interrupt Mask Register (TIMSK0). TIFR0 and TIMSK0 are not shown in the figure.

The Timer/Counter can be clocked internally, via the prescaler, or by an external clock source on the T0 pin. The Clock Select logic block controls which clock source and edge the Timer/Counter uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the Clock Select logic is referred to as the timer clock (clk_{T0}).

The double buffered Output Compare Registers (OCR0A and OCR0B) are compared with the Timer/Counter value at all times. The result of the compare can be used by the Waveform Generator to generate a PWM or variable frequency output on the Output Compare pins (OC0A and OC0B). See Section “13.7.3” on page 123. for details. The compare match event will also set the Compare Flag (OCF0A or OCF0B) which can be used to generate an Output Compare interrupt request.

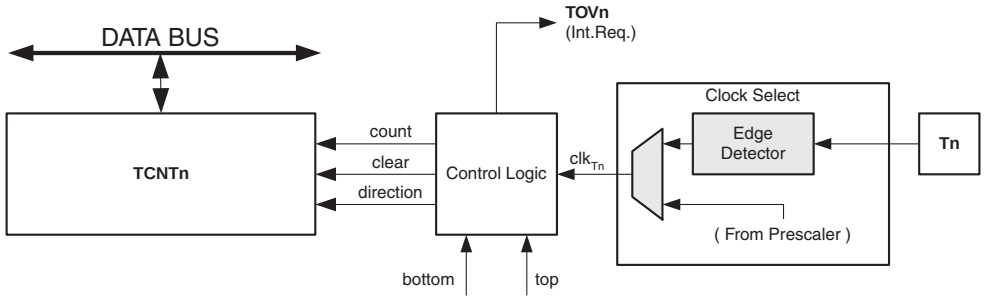
12.3 Timer/Counter Clock Sources

The Timer/Counter can be clocked by an internal or an external clock source. The clock source is selected by the Clock Select logic which is controlled by the Clock Select (CS02:0) bits located in the Timer/Counter Control Register (TCCR0B). For details on clock sources and prescaler, see ”Timer/Counter0 and Timer/Counter1 Prescalers” on page 141.

12.4 Counter Unit

The main part of the 8-bit Timer/Counter is the programmable bi-directional counter unit. Figure 12-2 shows a block diagram of the counter and its surroundings.

Figure 12-2. Counter Unit Block Diagram



Signal description (internal signals):

count	Increment or decrement TCNT0 by 1.
direction	Select between increment and decrement.
clear	Clear TCNT0 (set all bits to zero).
clk_{Tn}	Timer/Counter clock, referred to as clk_{T0} in the following.
top	Signalize that TCNT0 has reached maximum value.
bottom	Signalize that TCNT0 has reached minimum value (zero).

Depending of the mode of operation used, the counter is cleared, incremented, or decremented at each timer clock (clk_{T0}). clk_{T0} can be generated from an external or internal clock source, selected by the Clock Select bits (CS02:0). When no clock source is selected (CS02:0 = 0) the timer is stopped. However, the TCNT0 value can be accessed by the CPU, regardless of whether clk_{T0} is present or not. A CPU write overrides (has priority over) all counter clear or count operations.

The counting sequence is determined by the setting of the WGM01 and WGM00 bits located in the Timer/Counter Control Register (TCCR0A) and the WGM02 bit located in the Timer/Counter Control Register B (TCCR0B). There are close connections between how the counter behaves (counts) and how waveforms are generated on the Output Compare outputs OC0A and OC0B. For more details about advanced counting sequences and waveform generation, see ”Modes of Operation” on page 99.

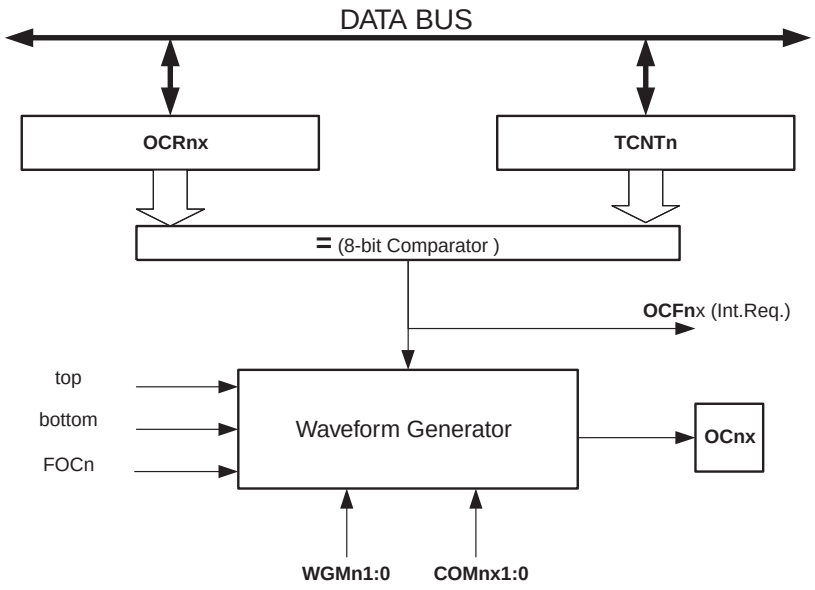
The Timer/Counter Overflow Flag (TOV0) is set according to the mode of operation selected by the WGM02:0 bits. TOV0 can be used for generating a CPU interrupt.

12.5 Output Compare Unit

The 8-bit comparator continuously compares TCNT0 with the Output Compare Registers (OCR0A and OCR0B). Whenever TCNT0 equals OCR0A or OCR0B, the comparator signals a match. A match will set the Output Compare Flag (OCF0A or OCF0B) at the next timer clock cycle. If the corresponding interrupt is enabled, the Output Compare Flag generates an Output Compare interrupt. The Output Compare Flag is automatically cleared when the interrupt is executed. Alternatively, the flag can be cleared by software by writing a logical one to its I/O bit location. The Waveform Generator uses the match signal to generate an output according to operating mode set by the WGM02:0 bits and Compare Output mode (COM0x1:0) bits. The max and bottom signals are used by the Waveform Generator for handling the special cases of the extreme values in some modes of operation (”Modes of Operation” on page 99).

Figure 12-3 shows a block diagram of the Output Compare unit.

Figure 12-3. Output Compare Unit, Block Diagram



The OCR0x Registers are double buffered when using any of the Pulse Width Modulation (PWM) modes. For the normal and Clear Timer on Compare (CTC) modes of operation, the double buffering is disabled. The double buffering synchronizes the update of the OCR0x Compare Registers to either top or bottom of the counting sequence. The synchronization prevents the occurrence of odd-length, non-symmetrical PWM pulses, thereby making the output glitch-free.

The OCR0x Register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCR0x Buffer Register, and if double buffering is disabled the CPU will access the OCR0x directly.

12.5.1 Force Output Compare

In non-PWM waveform generation modes, the match output of the comparator can be forced by writing a one to the Force Output Compare (FOC0x) bit. Forcing compare match will not set the OCF0x Flag or reload/clear the timer, but the OC0x pin will be updated as if a real compare match had occurred (the COM0x1:0 bits settings define whether the OC0x pin is set, cleared or toggled).

12.5.2 Compare Match Blocking by TCNT0 Write

All CPU write operations to the TCNT0 Register will block any compare match that occur in the next timer clock cycle, even when the timer is stopped. This feature allows OCR0x to be initialized to the same value as TCNT0 without triggering an interrupt when the Timer/Counter clock is enabled.

12.5.3 Using the Output Compare Unit

Since writing TCNT0 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT0 when using the Output Compare Unit, independently of whether the Timer/Counter is running or not. If the value written to TCNT0 equals the OCR0x value, the compare match will be missed, resulting in incorrect waveform generation. Similarly, do not write the TCNT0 value equal to BOTTOM when the counter is downcounting.

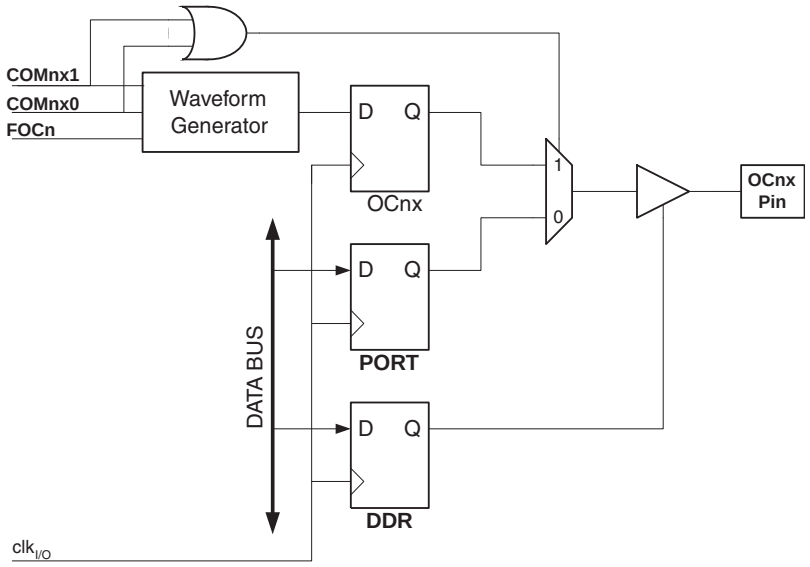
The setup of the OC0x should be performed before setting the Data Direction Register for the port pin to output. The easiest way of setting the OC0x value is to use the Force Output Compare (FOC0x) strobe bits in Normal mode. The OC0x Registers keep their values even when changing between Waveform Generation modes.

Be aware that the COM0x1:0 bits are not double buffered together with the compare value. Changing the COM0x1:0 bits will take effect immediately.

12.6 Compare Match Output Unit

The Compare Output mode (COM0x1:0) bits have two functions. The Waveform Generator uses the COM0x1:0 bits for defining the Output Compare (OC0x) state at the next compare match. Also, the COM0x1:0 bits control the OC0x pin output source. Figure 12-4 shows a simplified schematic of the logic affected by the COM0x1:0 bit setting. The I/O Registers, I/O bits, and I/O pins in the figure are shown in bold. Only the parts of the general I/O port control registers (DDR and PORT) that are affected by the COM0x1:0 bits are shown. When referring to the OC0x state, the reference is for the internal OC0x Register, not the OC0x pin. If a system reset occur, the OC0x Register is reset to “0”.

Figure 12-4. Compare Match Output Unit, Schematic



The general I/O port function is overridden by the Output Compare (OC0x) from the Waveform Generator if either of the COM0x1:0 bits are set. However, the OC0x pin direction (input or output) is still controlled by the Data Direction Register (DDR) for the port pin. The Data Direction Register bit for the OC0x pin (DDR_OC0x) must be set as output before the OC0x value is visible on the pin. The port override function is independent of the Waveform Generation mode.

The design of the Output Compare pin logic allows initialization of the OC0x state before the output is enabled. Note that some COM0x1:0 bit settings are reserved for certain modes of operation. See Section “12.9” on page 106.

12.6.1 Compare Output Mode and Waveform Generation

The Waveform Generator uses the COM0x1:0 bits differently in Normal, CTC, and PWM modes. For all modes, setting the COM0x1:0 = 0 tells the Waveform Generator that no action on the OC0x Register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 12-2 on page 106. For fast PWM mode, refer to Table 12-3 on page 106, and for phase correct PWM refer to Table 12-4 on page 107.

A change of the COM0x1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC0x strobe bits.

12.7 Modes of Operation

The mode of operation, i.e., the behavior of the Timer/Counter and the Output Compare pins, is defined by the combination of the Waveform Generation mode (WGM02:0) and Compare Output mode (COM0x1:0) bits. The Compare Output mode bits do not affect the counting sequence, while the Waveform Generation mode bits do. The COM0x1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM0x1:0 bits control whether the output should be set, cleared, or toggled at a compare match (See Section “12.6” on page 98.).

For detailed timing information refer to ”Timer/Counter Timing Diagrams” on page 104.

12.7.1 Normal Mode

The simplest mode of operation is the Normal mode (WGM02:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 8-bit value (TOP = 0xFF) and then restarts from the bottom (0x00). In normal operation the Timer/Counter Overflow Flag (TOV0) will be set in the same timer clock cycle as the TCNT0 becomes zero. The TOV0 Flag in this case behaves like a ninth bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV0 Flag, the timer resolution can be increased by software. There are no special cases to consider in the Normal mode, a new counter value can be written anytime.

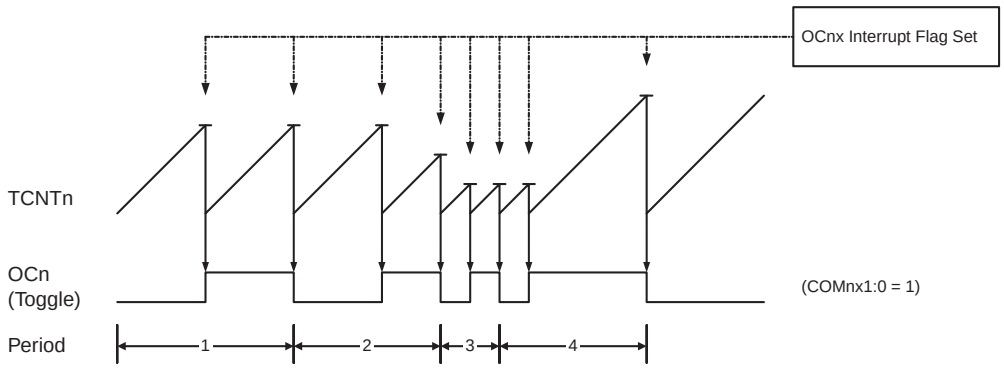
The Output Compare unit can be used to generate interrupts at some given time. Using the Output Compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

12.7.2 Clear Timer on Compare Match (CTC) Mode

In Clear Timer on Compare or CTC mode (WGM02:0 = 2), the OCR0A Register is used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT0) matches the OCR0A. The OCR0A defines the top value for the counter, hence also its resolution. This mode allows greater control of the compare match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 12-5. The counter value (TCNT0) increases until a compare match occurs between TCNT0 and OCR0A, and then counter (TCNT0) is cleared.

Figure 12-5. CTC Mode, Timing Diagram



An interrupt can be generated each time the counter value reaches the TOP value by using the OCF0A Flag. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing TOP to a value close to BOTTOM when the counter is running with none or a low prescaler value must be done with care since the CTC mode does not have the double buffering feature. If the new value written to OCR0A is lower than the current value of TCNT0, the counter will miss the compare match. The counter will then have to count to its maximum value (0xFF) and wrap around starting at 0x00 before the compare match can occur.

For generating a waveform output in CTC mode, the OC0A output can be set to toggle its logical level on each compare match by setting the Compare Output mode bits to toggle mode (COM0A1:0 = 1). The OC0A value will not be visible on the port pin unless the data direction for

the pin is set to output. The waveform generated will have a maximum frequency of $f_{OC0} = f_{clk_I/O}/2$ when OCR0A is set to zero (0x00). The waveform frequency is defined by the following equation:

$$f_{OCnx} = \frac{f_{clk_I/O}}{2 \cdot N \cdot (1 + OCRnx)}$$

The *N* variable represents the prescale factor (1, 8, 64, 256, or 1024).

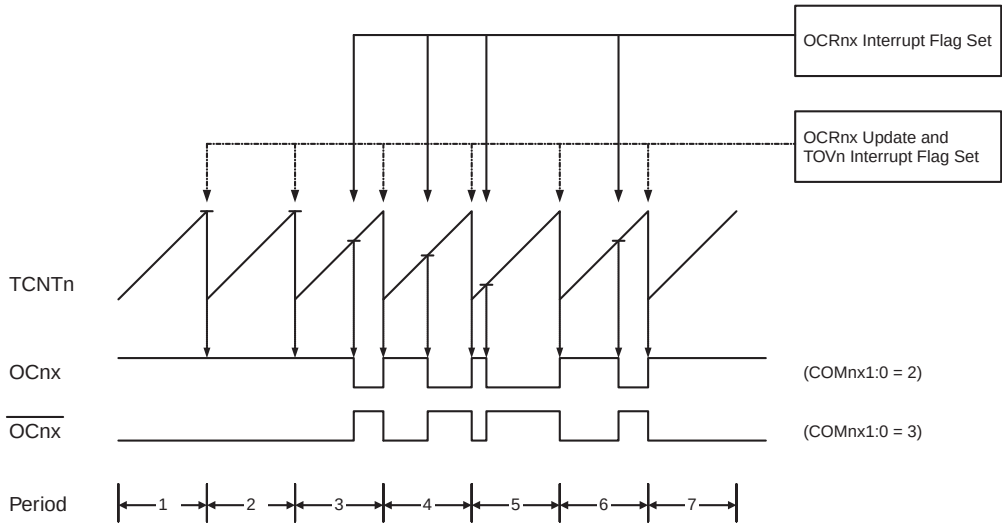
As for the Normal mode of operation, the TOV0 Flag is set in the same timer clock cycle that the counter counts from MAX to 0x00.

12.7.3 Fast PWM Mode

The fast Pulse Width Modulation or fast PWM mode (WGM02:0 = 3 or 7) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM option by its single-slope operation. The counter counts from BOTTOM to TOP then restarts from BOTTOM. TOP is defined as 0xFF when WGM2:0 = 3, and OCR0A when WGM2:0 = 7. In non-inverting Compare Output mode, the Output Compare (OC0x) is cleared on the compare match between TCNT0 and OCR0x, and set at BOTTOM. In inverting Compare Output mode, the output is set on compare match and cleared at BOTTOM. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as high as the phase correct PWM mode that use dual-slope operation. This high frequency makes the fast PWM mode well suited for power regulation, rectification, and DAC applications. High frequency allows physically small sized external components (coils, capacitors), and therefore reduces total system cost.

In fast PWM mode, the counter is incremented until the counter value matches the TOP value. The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in Figure 12-6. The TCNT0 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT0 slopes represent compare matches between OCR0x and TCNT0.

Figure 12-6. Fast PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV0) is set each time the counter reaches TOP. If the interrupt is enabled, the interrupt handler routine can be used for updating the compare value.

In fast PWM mode, the compare unit allows generation of PWM waveforms on the OC0x pins. Setting the COM0x1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM0x1:0 to three: Setting the COM0A1:0 bits to one allows the OC0A pin to toggle on Compare Matches if the WGM02 bit is set. This option is not available for the OC0B pin (see Table 12-6 on page 107). The actual OC0x value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by setting (or clearing) the OC0x Register at the compare match between OCR0x and TCNT0, and clearing (or setting) the OC0x Register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

The PWM frequency for the output can be calculated by the following equation:

$$f_{OCnxPWM} = \frac{f_{clk_I/O}}{N \cdot 256}$$

The *N* variable represents the prescale factor (1, 8, 64, 256, or 1024).

The extreme values for the OCR0A Register represents special cases when generating a PWM waveform output in the fast PWM mode. If the OCR0A is set equal to BOTTOM, the output will be a narrow spike for each MAX+1 timer clock cycle. Setting the OCR0A equal to MAX will result in a constantly high or low output (depending on the polarity of the output set by the COM0A1:0 bits.)

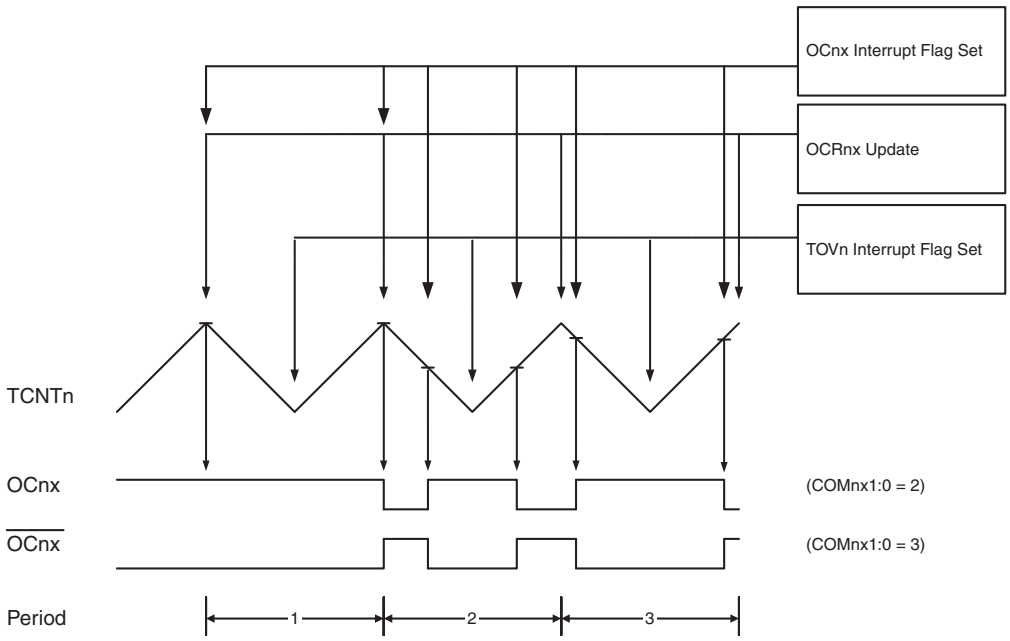
A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC0x to toggle its logical level on each compare match (COM0x1:0 = 1). The waveform generated will have a maximum frequency of $f_{OC0} = f_{clk_I/O}/2$ when OCR0A is set to zero. This feature is similar to the OC0A toggle in CTC mode, except the double buffer feature of the Output Compare unit is enabled in the fast PWM mode.

12.7.4 Phase Correct PWM Mode

The phase correct PWM mode (WGM02:0 = 1 or 5) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is based on a dual-slope operation. The counter counts repeatedly from BOTTOM to TOP and then from TOP to BOTTOM. TOP is defined as 0xFF when WGM2:0 = 1, and OCR0A when WGM2:0 = 5. In non-inverting Compare Output mode, the Output Compare (OC0x) is cleared on the compare match between TCNT0 and OCR0x while upcounting, and set on the compare match while downcounting. In inverting Output Compare mode, the operation is inverted. The dual-slope operation has lower maximum operation frequency than single slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

In phase correct PWM mode the counter is incremented until the counter value matches TOP. When the counter reaches TOP, it changes the count direction. The TCNT0 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 12-7. The TCNT0 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT0 slopes represent compare matches between OCR0x and TCNT0.

Figure 12-7. Phase Correct PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV0) is set each time the counter reaches BOTTOM. The Interrupt Flag can be used to generate an interrupt each time the counter reaches the BOTTOM value.

In phase correct PWM mode, the compare unit allows generation of PWM waveforms on the OC0x pins. Setting the COM0x1:0 bits to two will produce a non-inverted PWM. An inverted PWM output can be generated by setting the COM0x1:0 to three: Setting the COM0A0 bits to one allows the OC0A pin to toggle on Compare Matches if the WGM02 bit is set. This option is not available for the OC0B pin (see Table 12-7 on page 108). The actual OC0x value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by clearing (or setting) the OC0x Register at the compare match between OCR0x and TCNT0 when the counter increments, and setting (or clearing) the OC0x Register at compare match between OCR0x and TCNT0 when the counter decrements. The PWM frequency for the output when using phase correct PWM can be calculated by the following equation:

$$f_{OCnxPCPWM} = \frac{f_{clk_I/O}}{N \cdot 510}$$

The N variable represents the prescale factor (1, 8, 64, 256, or 1024).

The extreme values for the OCR0A Register represent special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR0A is set equal to BOTTOM, the output will be continuously low and if set equal to MAX the output will be continuously high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

At the very start of period 2 in Figure 12-7 OCnx has a transition from high to low even though there is no Compare Match. The point of this transition is to guarantee symmetry around BOTTOM. There are two cases that give a transition without Compare Match.

- OCRnx changes its value from MAX, like in Figure 12-7. When the OCR0A value is MAX the OCn pin value is the same as the result of a down-counting Compare Match. To ensure

symmetry around BOTTOM the OCnx value at MAX must correspond to the result of an up-counting Compare Match.

- The timer starts counting from a value higher than the one in OCRnx, and for that reason misses the Compare Match and hence the OCnx change that would have happened on the way up.

12.8 Timer/Counter Timing Diagrams

The Timer/Counter is a synchronous design and the timer clock (clk_{T0}) is therefore shown as a clock enable signal in the following figures. The figures include information on when interrupt flags are set. Figure 12-8 contains timing data for basic Timer/Counter operation. The figure shows the count sequence close to the MAX value in all modes other than phase correct PWM mode.

Figure 12-8. Timer/Counter Timing Diagram, no Prescaling

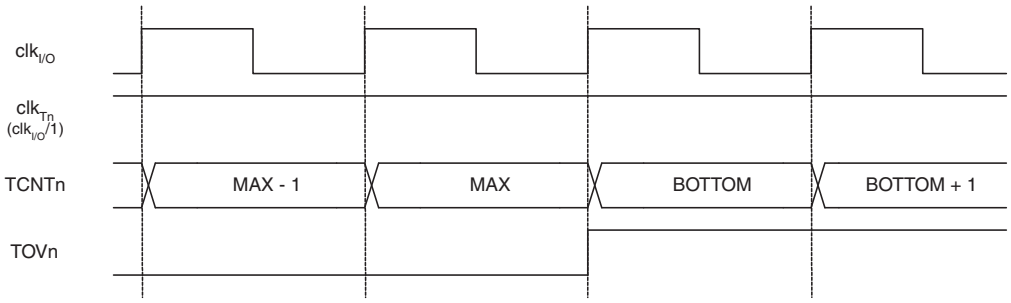


Figure 12-9 shows the same timing data, but with the prescaler enabled.

Figure 12-9. Timer/Counter Timing Diagram, with Prescaler ($f_{clk_I/O}/8$)

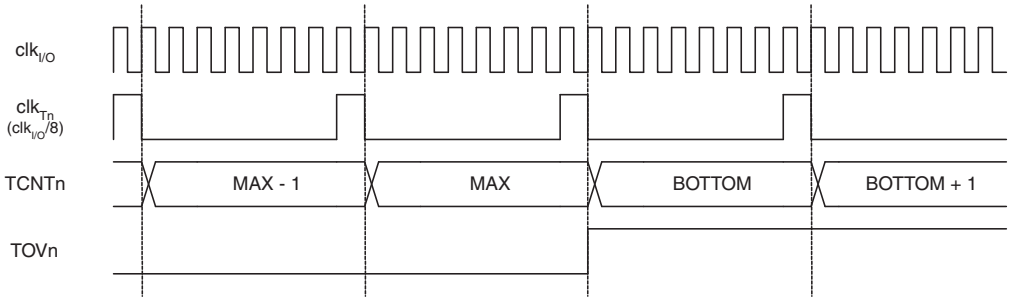


Figure 12-10 shows the setting of OCF0B in all modes and OCF0A in all modes except CTC mode and PWM mode, where OCR0A is TOP.

Figure 12-10. Timer/Counter Timing Diagram, Setting of OCF0x, with Prescaler ($f_{clk_I/O}/8$)

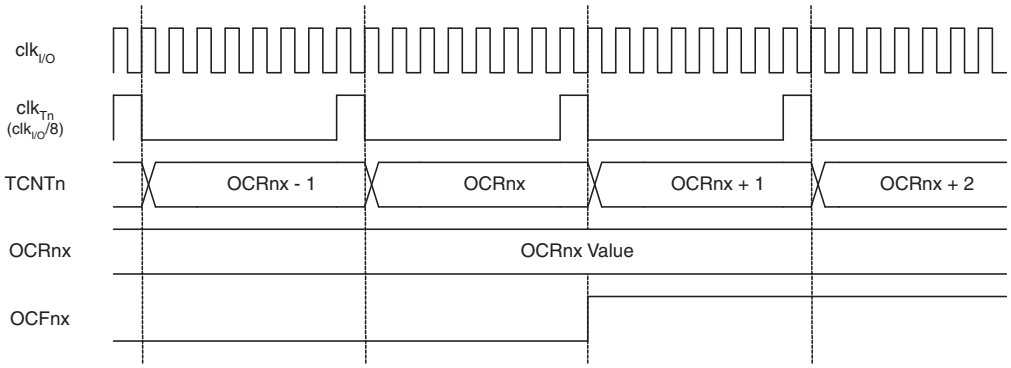
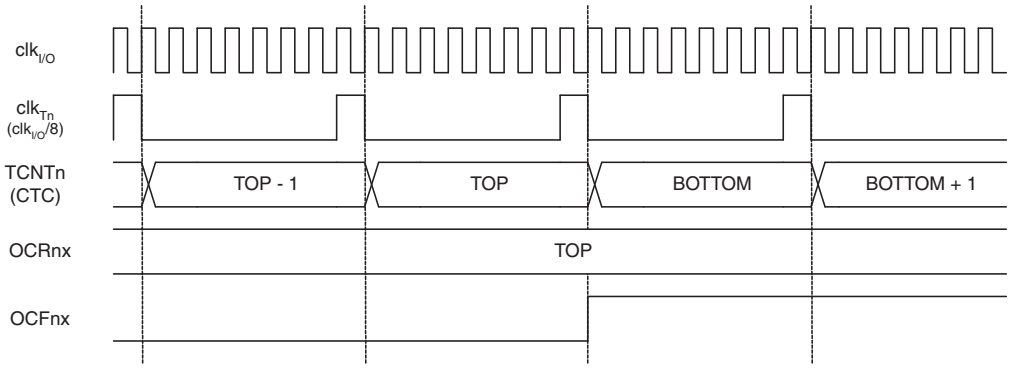


Figure 12-11 shows the setting of OCF0A and the clearing of TCNT0 in CTC mode and fast PWM mode where OCR0A is TOP.

Figure 12-11. Timer/Counter Timing Diagram, Clear Timer on Compare Match mode, with Prescaler ($f_{clk_I/O}/8$)



12.9 Register Description

12.9.1 TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:6 – COM0A1:0: Compare Match Output A Mode**

These bits control the Output Compare pin (OC0A) behavior. If one or both of the COM0A1:0 bits are set, the OC0A output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC0A pin must be set in order to enable the output driver.

When OC0A is connected to the pin, the function of the COM0A1:0 bits depends on the WGM02:0 bit setting. Table 12-2 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 12-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match
1	1	Set OC0A on Compare Match

Table 12-3 shows the COM0A1:0 bit functionality when the WGM01:0 bits are set to fast PWM mode.

Table 12-3. Compare Output Mode, Fast PWM Mode(1)

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected. WGM02 = 1: Toggle OC0A on Compare Match.
1	0	Clear OC0A on Compare Match, set OC0A at BOTTOM, (non-inverting mode).
1	1	Set OC0A on Compare Match, clear OC0A at BOTTOM, (inverting mode).

Note: 1. A special case occurs when OCR0A equals TOP and COM0A1 is set. In this case, the Compare Match is ignored, but the set or clear is done at BOTTOM. See ”Fast PWM Mode” on page 101 for more details.

Table 12-4 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 12-4. Compare Output Mode, Phase Correct PWM Mode(1)

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected. WGM02 = 1: Toggle OC0A on Compare Match.
1	0	Clear OC0A on Compare Match when up-counting. Set OC0A on Compare Match when down-counting.
1	1	Set OC0A on Compare Match when up-counting. Clear OC0A on Compare Match when down-counting.

Note: 1. A special case occurs when OCR0A equals TOP and COM0A1 is set. In this case, the Compare Match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 128 for more details.

- **Bits 5:4 – COM0B1:0: Compare Match Output B Mode**

These bits control the Output Compare pin (OC0B) behavior. If one or both of the COM0B1:0 bits are set, the OC0B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC0B pin must be set in order to enable the output driver.

When OC0B is connected to the pin, the function of the COM0B1:0 bits depends on the WGM02:0 bit setting. Table 12-5 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 12-5. Compare Output Mode, non-PWM Mode

COM0B1	COM0B0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Toggle OC0B on Compare Match
1	0	Clear OC0B on Compare Match
1	1	Set OC0B on Compare Match

Table 12-6 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to fast PWM mode.

Table 12-6. Compare Output Mode, Fast PWM Mode(1)

COM0B1	COM0B0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Reserved
1	0	Clear OC0B on Compare Match, set OC0B at BOTTOM, (non-inverting mode)
1	1	Set OC0B on Compare Match, clear OC0B at BOTTOM, (inverting mode).

Note: 1. A special case occurs when OCR0B equals TOP and COM0B1 is set. In this case, the Compare Match is ignored, but the set or clear is done at TOP. See "Fast PWM Mode" on page 101 for more details.

Table 12-7 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 12-7. Compare Output Mode, Phase Correct PWM Mode(1)

COM0B1	COM0B0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Reserved
1	0	Clear OC0B on Compare Match when up-counting. Set OC0B on Compare Match when down-counting.
1	1	Set OC0B on Compare Match when up-counting. Clear OC0B on Compare Match when down-counting.

Note: 1. A special case occurs when OCR0B equals TOP and COM0B1 is set. In this case, the Compare Match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 102 for more details.

- **Bits 3, 2 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bits 1:0 – WGM01:0: Waveform Generation Mode**

Combined with the WGM02 bit found in the TCCR0B Register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used, see Table 12-8. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), Clear Timer on Compare Match (CTC) mode, and two types of Pulse Width Modulation (PWM) modes (see "Modes of Operation" on page 99).

Table 12-8. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Notes: 1. MAX = 0xFF
2. BOTTOM = 0x00

12.9.2 TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – FOC0A: Force Output Compare A**

The FOC0A bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0A bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0A output is changed according to its COM0A1:0 bits setting. Note that the FOC0A bit is implemented as a strobe. Therefore it is the value present in the COM0A1:0 bits that determines the effect of the forced compare.

A FOC0A strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0A as TOP.

The FOC0A bit is always read as zero.

- **Bit 6 – FOC0B: Force Output Compare B**

The FOC0B bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0B bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0B output is changed according to its COM0B1:0 bits setting. Note that the FOC0B bit is implemented as a strobe. Therefore it is the value present in the COM0B1:0 bits that determines the effect of the forced compare.

A FOC0B strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0B as TOP.

The FOC0B bit is always read as zero.

- **Bits 5:4 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bit 3 – WGM02: Waveform Generation Mode**

See the description in the ”TCCR0A – Timer/Counter Control Register A” on page 106.

- **Bits 2:0 – CS02:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter.

Table 12-9. Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter0, transitions on the T0 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

12.9.3 TCNT0 – Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
0x26 (0x46)	TCNT0[7:0]								TCNT0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Timer/Counter Register gives direct access, both for read and write operations, to the Timer/Counter unit 8-bit counter. Writing to the TCNT0 Register blocks (removes) the Compare Match on the following timer clock. Modifying the counter (TCNT0) while the counter is running, introduces a risk of missing a Compare Match between TCNT0 and the OCR0x Registers.

12.9.4 OCR0A – Output Compare Register A

Bit	7	6	5	4	3	2	1	0	
0x27 (0x47)	OCR0A[7:0]								OCR0A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register A contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0A pin.

12.9.5 OCR0B – Output Compare Register B

Bit	7	6	5	4	3	2	1	0	
0x28 (0x48)	OCR0B[7:0]								OCR0B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register B contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0B pin.

12.9.6 TIMSK0 – Timer/Counter Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bit 2 – OCIE0B: Timer/Counter Output Compare Match B Interrupt Enable**

When the OCIE0B bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter Compare Match B interrupt is enabled. The corresponding interrupt is executed if a Compare Match in Timer/Counter occurs, i.e., when the OCF0B bit is set in the Timer/Counter Interrupt Flag Register – TIFR0.

- **Bit 1 – OCIE0A: Timer/Counter0 Output Compare Match A Interrupt Enable**

When the OCIE0A bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter0 Compare Match A interrupt is enabled. The corresponding interrupt is executed if a Compare Match in Timer/Counter0 occurs, i.e., when the OCF0A bit is set in the Timer/Counter 0 Interrupt Flag Register – TIFR0.

- **Bit 0 – TOIE0: Timer/Counter0 Overflow Interrupt Enable**

When the TOIE0 bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter0 occurs, i.e., when the TOV0 bit is set in the Timer/Counter 0 Interrupt Flag Register – TIFR0.

12.9.7 TIFR0 – Timer/Counter 0 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x15 (0x35)	–	–	–	–	–	OCF0B	OCF0A	TOV0	TIFR0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..3 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bit 2 – OCF0B: Timer/Counter 0 Output Compare B Match Flag**

The OCF0B bit is set when a Compare Match occurs between the Timer/Counter and the data in OCR0B – Output Compare Register0 B. OCF0B is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0B is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0B (Timer/Counter Compare B Match Interrupt Enable), and OCF0B are set, the Timer/Counter Compare Match Interrupt is executed.

- **Bit 1 – OCF0A: Timer/Counter 0 Output Compare A Match Flag**

The OCF0A bit is set when a Compare Match occurs between the Timer/Counter0 and the data in OCR0A – Output Compare Register0. OCF0A is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0A is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0A (Timer/Counter0 Compare Match Interrupt Enable), and OCF0A are set, the Timer/Counter0 Compare Match Interrupt is executed.

- **Bit 0 – TOV0: Timer/Counter0 Overflow Flag**

The bit TOV0 is set when an overflow occurs in Timer/Counter0. TOV0 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV0 is cleared by writing a logic one to the flag. When the SREG I-bit, TOIE0 (Timer/Counter0 Overflow Interrupt Enable), and TOV0 are set, the Timer/Counter0 Overflow interrupt is executed.

The setting of this flag is dependent of the WGM02:0 bit setting. Refer to Table 12-8, ”Waveform Generation Mode Bit Description” on page 108.

13. 16-bit Timer/Counter1 with PWM

13.1 Features

- **True 16-bit Design (i.e., Allows 16-bit PWM)**
- **Two independent Output Compare Units**
- **Double Buffered Output Compare Registers**
- **One Input Capture Unit**
- **Input Capture Noise Canceler**
- **Clear Timer on Compare Match (Auto Reload)**
- **Glitch-free, Phase Correct Pulse Width Modulator (PWM)**
- **Variable PWM Period**
- **Frequency Generator**
- **External Event Counter**
- **Four independent interrupt Sources (TOV1, OCF1A, OCF1B, and ICF1)**

13.2 Overview

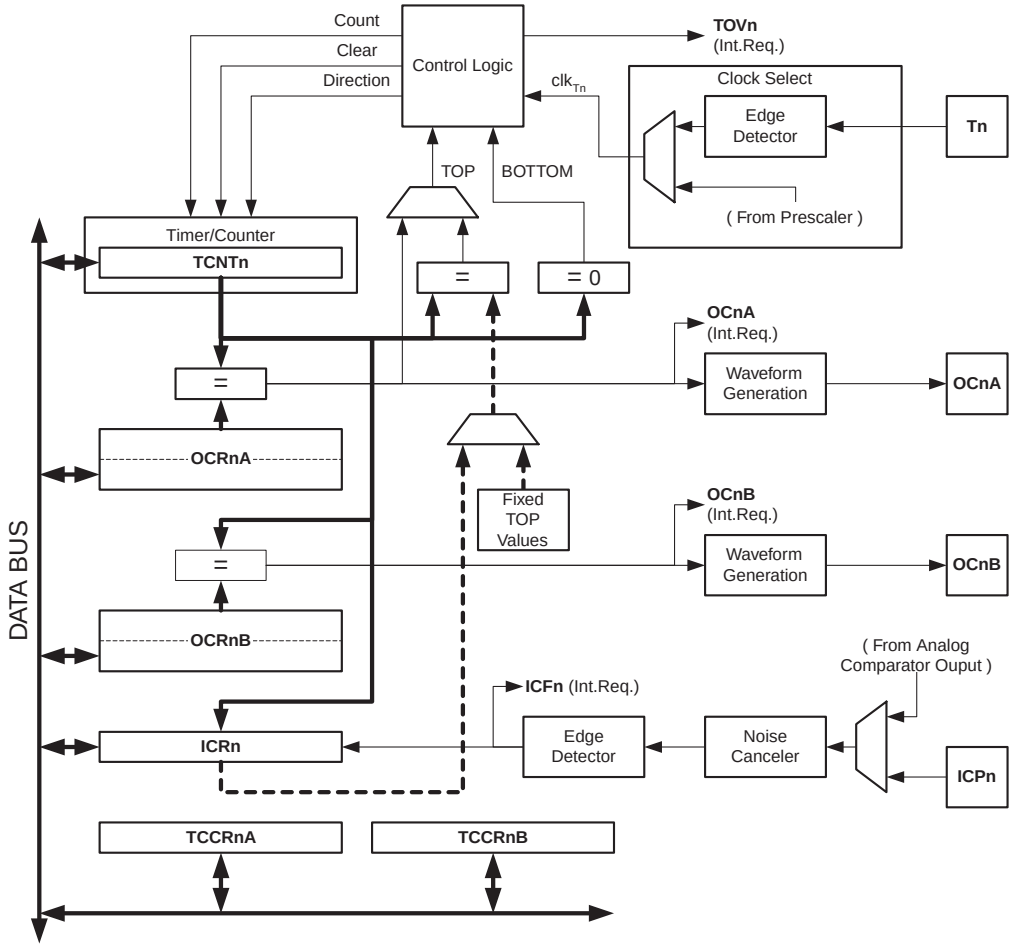
The 16-bit Timer/Counter unit allows accurate program execution timing (event management), wave generation, and signal timing measurement.

Most register and bit references in this section are written in general form. A lower case “n” replaces the Timer/Counter number, and a lower case “x” replaces the Output Compare unit channel. However, when using the register or bit defines in a program, the precise form must be used, i.e., TCNT1 for accessing Timer/Counter1 counter value and so on.

A simplified block diagram of the 16-bit Timer/Counter is shown in Figure 13-1. For the actual placement of I/O pins, refer to ”Pinout ATmega48P/88P/168P/328P” on page 2. CPU accessible I/O Registers, including I/O bits and I/O pins, are shown in bold. The device-specific I/O Register and bit locations are listed in the ”Register Description” on page 134.

The PRTIM1 bit in ”PRR – Power Reduction Register” on page 45 must be written to zero to enable Timer/Counter1 module.

Figure 13-1. 16-bit Timer/Counter Block Diagram[1]



Note: 1. Refer to Figure 1-1 on page 2, Table 11-3 on page 82 and Table 11-9 on page 88 for Timer/Counter1 pin placement and description.

13.2.1 Registers

The *Timer/Counter* (TCNT1), *Output Compare Registers* (OCR1A/B), and *Input Capture Register* (ICR1) are all 16-bit registers. Special procedures must be followed when accessing the 16-bit registers. These procedures are described in the section "Accessing 16-bit Registers" on page 115. The *Timer/Counter Control Registers* (TCCR1A/B) are 8-bit registers and have no CPU access restrictions. Interrupt requests (abbreviated to Int.Req. in the figure) signals are all visible in the *Timer Interrupt Flag Register* (TIFR1). All interrupts are individually masked with the *Timer Interrupt Mask Register* (TIMSK1). TIFR1 and TIMSK1 are not shown in the figure.

The Timer/Counter can be clocked internally, via the prescaler, or by an external clock source on the T1 pin. The Clock Select logic block controls which clock source and edge the Timer/Counter uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the Clock Select logic is referred to as the timer clock (clk_{T1}).

The double buffered Output Compare Registers (OCR1A/B) are compared with the Timer/Counter value at all time. The result of the compare can be used by the Waveform Generator to generate a PWM or variable frequency output on the Output Compare pin (OC1A/B). See "Out-

put Compare Units" on page 122. The compare match event will also set the Compare Match Flag (OCF1A/B) which can be used to generate an Output Compare interrupt request.

The Input Capture Register can capture the Timer/Counter value at a given external (edge triggered) event on either the Input Capture pin (ICP1) or on the Analog Comparator pins (See "Analog Comparator" on page 246) The Input Capture unit includes a digital filtering unit (Noise Canceler) for reducing the chance of capturing noise spikes.

The TOP value, or maximum Timer/Counter value, can in some modes of operation be defined by either the OCR1A Register, the ICR1 Register, or by a set of fixed values. When using OCR1A as TOP value in a PWM mode, the OCR1A Register can not be used for generating a PWM output. However, the TOP value will in this case be double buffered allowing the TOP value to be changed in run time. If a fixed TOP value is required, the ICR1 Register can be used as an alternative, freeing the OCR1A to be used as PWM output.

13.2.2 Definitions

The following definitions are used extensively throughout the section:

BOTTOM	The counter reaches the *BOTTOM* when it becomes 0x0000.
MAX	The counter reaches its *MAX*imum when it becomes 0xFFFF (decimal 65535).
TOP	The counter reaches the *TOP* when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be one of the fixed values: 0x00FF, 0x01FF, or 0x03FF, or to the value stored in the OCR1A or ICR1 Register. The assignment is dependent of the mode of operation.

13.3 Accessing 16-bit Registers

The TCNT1, OCR1A/B, and ICR1 are 16-bit registers that can be accessed by the AVR CPU via the 8-bit data bus. The 16-bit register must be byte accessed using two read or write operations. Each 16-bit timer has a single 8-bit register for temporary storing of the high byte of the 16-bit access. The same temporary register is shared between all 16-bit registers within each 16-bit timer. Accessing the low byte triggers the 16-bit read or write operation. When the low byte of a 16-bit register is written by the CPU, the high byte stored in the temporary register, and the low byte written are both copied into the 16-bit register in the same clock cycle. When the low byte of a 16-bit register is read by the CPU, the high byte of the 16-bit register is copied into the temporary register in the same clock cycle as the low byte is read.

Not all 16-bit accesses uses the temporary register for the high byte. Reading the OCR1A/B 16-bit registers does not involve using the temporary register.

To do a 16-bit write, the high byte must be written before the low byte. For a 16-bit read, the low byte must be read before the high byte.

The following code examples show how to access the 16-bit Timer Registers assuming that no interrupts updates the temporary register. The same principle can be used directly for accessing the OCR1A/B and ICR1 Registers. Note that when using "C", the compiler handles the 16-bit access.

Assembly Code Examples[1]

```
...
; Set TCNT1 to 0x01FF
ldi r17,0x01
ldi r16,0xFF
out TCNT1H,r17
out TCNT1L,r16
; Read TCNT1 into r17:r16
in  r16,TCNT1L
in  r17,TCNT1H
...
```

C Code Examples[1]

```
unsigned int i;
...
/* Set TCNT1 to 0x01FF */
TCNT1 = 0x1FF;
/* Read TCNT1 into i */
i = TCNT1;
...
```

Note: 1. See ”Code Examples” on page 7.
For I/O Registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

The assembly code example returns the TCNT1 value in the r17:r16 register pair.

It is important to notice that accessing 16-bit registers are atomic operations. If an interrupt occurs between the two instructions accessing the 16-bit register, and the interrupt code updates the temporary register by accessing the same or any other of the 16-bit Timer Registers, then the result of the access outside the interrupt will be corrupted. Therefore, when both the main code and the interrupt code update the temporary register, the main code must disable the interrupts during the 16-bit access.

The following code examples show how to do an atomic read of the TCNT1 Register contents. Reading any of the OCR1A/B or ICR1 Registers can be done by using the same principle.

Assembly Code Example[1]

```
TIM16_ReadTCNT1:
  ; Save global interrupt flag
  in  r18,SREG
  ; Disable interrupts
  cli
  ; Read TCNT1 into r17:r16
  in  r16,TCNT1L
  in  r17,TCNT1H
  ; Restore global interrupt flag
  out SREG,r18
  ret
```

C Code Example[1]

```
unsigned int TIM16_ReadTCNT1( void )
{
  unsigned char sreg;
  unsigned int i;
  /* Save global interrupt flag */
  sreg = SREG;
  /* Disable interrupts */
  _CLI();
  /* Read TCNT1 into i */
  i = TCNT1;
  /* Restore global interrupt flag */
  SREG = sreg;
  return i;
}
```

Note: 1. See "Code Examples" on page 7.
For I/O Registers located in extended I/O map, "IN", "OUT", "SBIS", "SBIC", "CBI", and "SBI" instructions must be replaced with instructions that allow access to extended I/O. Typically "LDS" and "STS" combined with "SBRS", "SBRC", "SBR", and "CBR".

The assembly code example returns the TCNT1 value in the r17:r16 register pair.

The following code examples show how to do an atomic write of the TCNT1 Register contents. Writing any of the OCR1A/B or ICR1 Registers can be done by using the same principle.

Assembly Code Example[1]

```
TIM16_WriteTCNT1:
  ; Save global interrupt flag
  in  r18,SREG
  ; Disable interrupts
  cli
  ; Set TCNT1 to r17:r16
  out TCNT1H,r17
  out TCNT1L,r16
  ; Restore global interrupt flag
  out SREG,r18
  ret
```

C Code Example[1]

```
void TIM16_WriteTCNT1( unsigned int i )
{
  unsigned char sreg;
  unsigned int i;
  /* Save global interrupt flag */
  sreg = SREG;
  /* Disable interrupts */
  _CLI();
  /* Set TCNT1 to i */
  TCNT1 = i;
  /* Restore global interrupt flag */
  SREG = sreg;
}
```

Note: 1. See ”Code Examples” on page 7.
For I/O Registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

The assembly code example requires that the r17:r16 register pair contains the value to be written to TCNT1.

13.3.1 Reusing the Temporary High Byte Register

If writing to more than one 16-bit register where the high byte is the same for all registers written, then the high byte only needs to be written once. However, note that the same rule of atomic operation described previously also applies in this case.

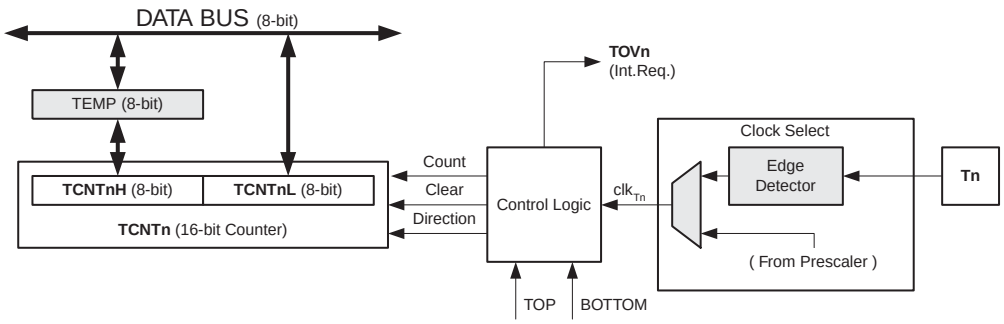
13.4 Timer/Counter Clock Sources

The Timer/Counter can be clocked by an internal or an external clock source. The clock source is selected by the Clock Select logic which is controlled by the *Clock Select* (CS12:0) bits located in the *Timer/Counter control Register B* (TCCR1B). For details on clock sources and prescaler, see ”Timer/Counter0 and Timer/Counter1 Prescalers” on page 141.

13.5 Counter Unit

The main part of the 16-bit Timer/Counter is the programmable 16-bit bi-directional counter unit. Figure 13-2 shows a block diagram of the counter and its surroundings.

Figure 13-2. Counter Unit Block Diagram



Signal description (internal signals):

Count	Increment or decrement TCNT1 by 1.
Direction	Select between increment and decrement.
Clear	Clear TCNT1 (set all bits to zero).
clk_{T1}	Timer/Counter clock.
TOP	Signalize that TCNT1 has reached maximum value.
BOTTOM	Signalize that TCNT1 has reached minimum value (zero).

The 16-bit counter is mapped into two 8-bit I/O memory locations: *Counter High* (TCNT1H) containing the upper eight bits of the counter, and *Counter Low* (TCNT1L) containing the lower eight bits. The TCNT1H Register can only be indirectly accessed by the CPU. When the CPU does an access to the TCNT1H I/O location, the CPU accesses the high byte temporary register (TEMP). The temporary register is updated with the TCNT1H value when the TCNT1L is read, and TCNT1H is updated with the temporary register value when TCNT1L is written. This allows the CPU to read or write the entire 16-bit counter value within one clock cycle via the 8-bit data bus. It is important to notice that there are special cases of writing to the TCNT1 Register when the counter is counting that will give unpredictable results. The special cases are described in the sections where they are of importance.

Depending on the mode of operation used, the counter is cleared, incremented, or decremented at each *timer clock* (clk_{T1}). The clk_{T1} can be generated from an external or internal clock source, selected by the *Clock Select* bits (CS12:0). When no clock source is selected (CS12:0 = 0) the timer is stopped. However, the TCNT1 value can be accessed by the CPU, independent of whether clk_{T1} is present or not. A CPU write overrides (has priority over) all counter clear or count operations.

The counting sequence is determined by the setting of the *Waveform Generation mode* bits (WGM13:0) located in the *Timer/Counter Control Registers* A and B (TCCR1A and TCCR1B). There are close connections between how the counter behaves (counts) and how waveforms are generated on the Output Compare outputs OC1x. For more details about advanced counting sequences and waveform generation, see ”Modes of Operation” on page 125.

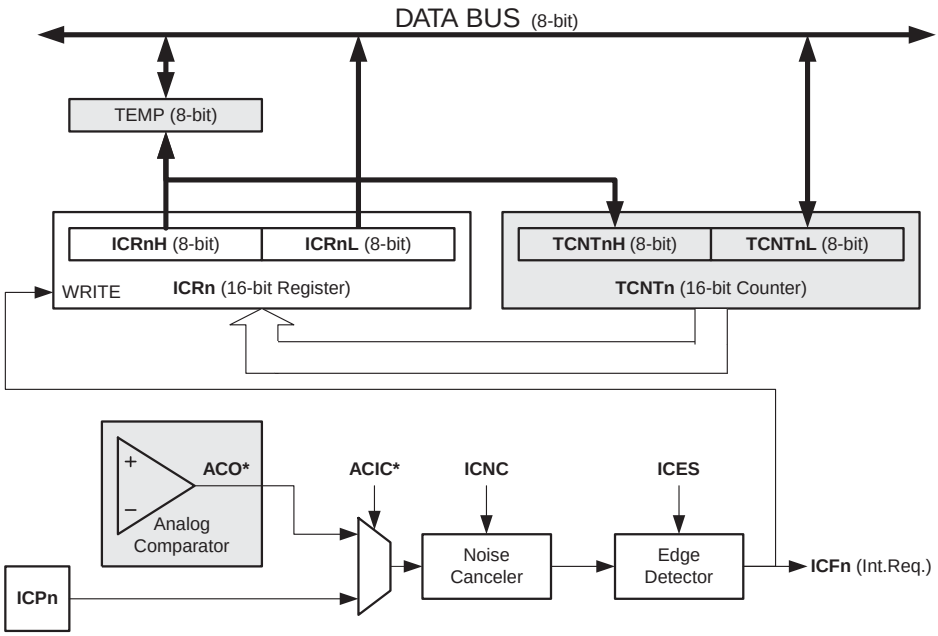
The Timer/Counter Overflow Flag (TOV1) is set according to the mode of operation selected by the WGM13:0 bits. TOV1 can be used for generating a CPU interrupt.

13.6 Input Capture Unit

The Timer/Counter incorporates an Input Capture unit that can capture external events and give them a time-stamp indicating time of occurrence. The external signal indicating an event, or multiple events, can be applied via the ICP1 pin or alternatively, via the analog-comparator unit. The time-stamps can then be used to calculate frequency, duty-cycle, and other features of the signal applied. Alternatively the time-stamps can be used for creating a log of the events.

The Input Capture unit is illustrated by the block diagram shown in Figure 13-3. The elements of the block diagram that are not directly a part of the Input Capture unit are gray shaded. The small “n” in register and bit names indicates the Timer/Counter number.

Figure 13-3. Input Capture Unit Block Diagram



When a change of the logic level (an event) occurs on the *Input Capture pin* (ICP1), alternatively on the *Analog Comparator output* (ACO), and this change confirms to the setting of the edge detector, a capture will be triggered. When a capture is triggered, the 16-bit value of the counter (TCNT1) is written to the *Input Capture Register* (ICR1). The *Input Capture Flag* (ICF1) is set at the same system clock as the TCNT1 value is copied into ICR1 Register. If enabled (ICIE1 = 1), the Input Capture Flag generates an Input Capture interrupt. The ICF1 Flag is automatically cleared when the interrupt is executed. Alternatively the ICF1 Flag can be cleared by software by writing a logical one to its I/O bit location.

Reading the 16-bit value in the *Input Capture Register* (ICR1) is done by first reading the low byte (ICR1L) and then the high byte (ICR1H). When the low byte is read the high byte is copied into the high byte temporary register (TEMP). When the CPU reads the ICR1H I/O location it will access the TEMP Register.

The ICR1 Register can only be written when using a Waveform Generation mode that utilizes the ICR1 Register for defining the counter’s TOP value. In these cases the *Waveform Genera-*

tion mode (WGM13:0) bits must be set before the TOP value can be written to the ICR1 Register. When writing the ICR1 Register the high byte must be written to the ICR1H I/O location before the low byte is written to ICR1L.

For more information on how to access the 16-bit registers refer to "Accessing 16-bit Registers" on page 115.

13.6.1 Input Capture Trigger Source

The main trigger source for the Input Capture unit is the *Input Capture pin* (ICP1). Timer/Counter1 can alternatively use the Analog Comparator output as trigger source for the Input Capture unit. The Analog Comparator is selected as trigger source by setting the *Analog Comparator Input Capture* (ACIC) bit in the *Analog Comparator Control and Status Register* (ACSR). Be aware that changing trigger source can trigger a capture. The Input Capture Flag must therefore be cleared after the change.

Both the *Input Capture pin* (ICP1) and the *Analog Comparator output* (ACO) inputs are sampled using the same technique as for the T1 pin (Figure 14-1 on page 141). The edge detector is also identical. However, when the noise canceler is enabled, additional logic is inserted before the edge detector, which increases the delay by four system clock cycles. Note that the input of the noise canceler and edge detector is always enabled unless the Timer/Counter is set in a Waveform Generation mode that uses ICR1 to define TOP.

An Input Capture can be triggered by software by controlling the port of the ICP1 pin.

13.6.2 Noise Canceler

The noise canceler improves noise immunity by using a simple digital filtering scheme. The noise canceler input is monitored over four samples, and all four must be equal for changing the output that in turn is used by the edge detector.

The noise canceler is enabled by setting the *Input Capture Noise Canceler* (ICNC1) bit in *Timer/Counter Control Register B* (TCCR1B). When enabled the noise canceler introduces additional four system clock cycles of delay from a change applied to the input, to the update of the ICR1 Register. The noise canceler uses the system clock and is therefore not affected by the prescaler.

13.6.3 Using the Input Capture Unit

The main challenge when using the Input Capture unit is to assign enough processor capacity for handling the incoming events. The time between two events is critical. If the processor has not read the captured value in the ICR1 Register before the next event occurs, the ICR1 will be overwritten with a new value. In this case the result of the capture will be incorrect.

When using the Input Capture interrupt, the ICR1 Register should be read as early in the interrupt handler routine as possible. Even though the Input Capture interrupt has relatively high priority, the maximum interrupt response time is dependent on the maximum number of clock cycles it takes to handle any of the other interrupt requests.

Using the Input Capture unit in any mode of operation when the TOP value (resolution) is actively changed during operation, is not recommended.

Measurement of an external signal's duty cycle requires that the trigger edge is changed after each capture. Changing the edge sensing must be done as early as possible after the ICR1 Register has been read. After a change of the edge, the Input Capture Flag (ICF1) must be

cleared by software (writing a logical one to the I/O bit location). For measuring frequency only, the clearing of the ICF1 Flag is not required (if an interrupt handler is used).

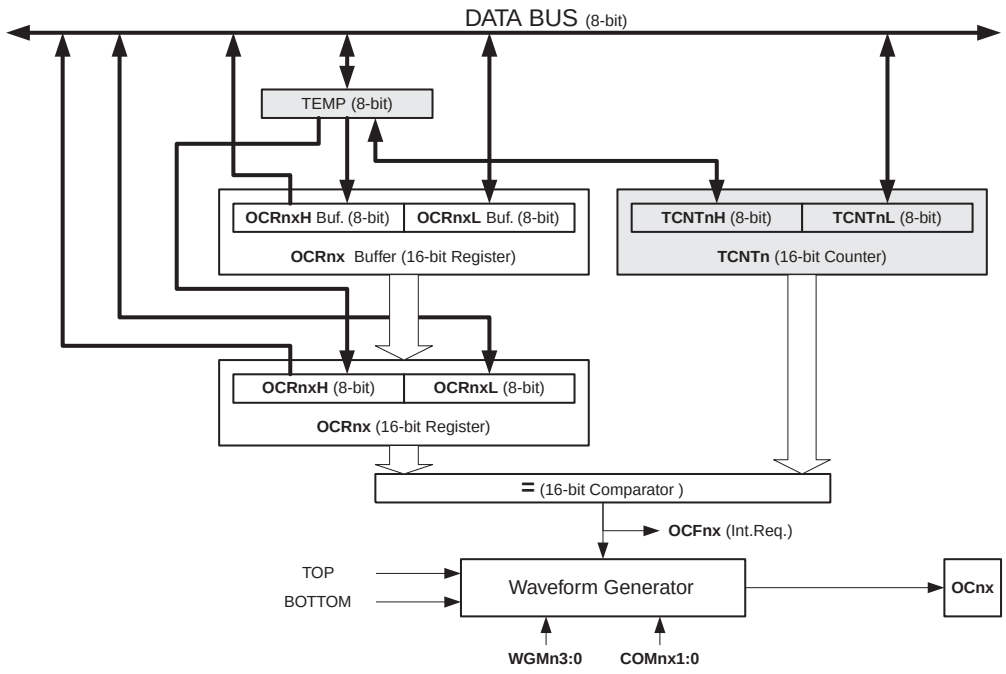
13.7 Output Compare Units

The 16-bit comparator continuously compares TCNT1 with the *Output Compare Register* (OCR1x). If TCNT equals OCR1x the comparator signals a match. A match will set the *Output Compare Flag* (OCF1x) at the next timer clock cycle. If enabled (OCIE1x = 1), the Output Compare Flag generates an Output Compare interrupt. The OCF1x Flag is automatically cleared when the interrupt is executed. Alternatively the OCF1x Flag can be cleared by software by writing a logical one to its I/O bit location. The Waveform Generator uses the match signal to generate an output according to operating mode set by the *Waveform Generation mode* (WGM13:0) bits and *Compare Output mode* (COM1x1:0) bits. The TOP and BOTTOM signals are used by the Waveform Generator for handling the special cases of the extreme values in some modes of operation (See Section “13.9” on page 125.)

A special feature of Output Compare unit A allows it to define the Timer/Counter TOP value (i.e., counter resolution). In addition to the counter resolution, the TOP value defines the period time for waveforms generated by the Waveform Generator.

Figure 13-4 shows a block diagram of the Output Compare unit. The small “n” in the register and bit names indicates the device number (n = 1 for Timer/Counter 1), and the “x” indicates Output Compare unit (A/B). The elements of the block diagram that are not directly a part of the Output Compare unit are gray shaded.

Figure 13-4. Output Compare Unit, Block Diagram



The OCR1x Register is double buffered when using any of the twelve *Pulse Width Modulation* (PWM) modes. For the Normal and *Clear Timer on Compare* (CTC) modes of operation, the double buffering is disabled. The double buffering synchronizes the update of the OCR1x Compare Register to either TOP or BOTTOM of the counting sequence. The synchronization

prevents the occurrence of odd-length, non-symmetrical PWM pulses, thereby making the output glitch-free.

The OCR1x Register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCR1x Buffer Register, and if double buffering is disabled the CPU will access the OCR1x directly. The content of the OCR1x (Buffer or Compare) Register is only changed by a write operation (the Timer/Counter does not update this register automatically as the TCNT1 and ICR1 Register). Therefore OCR1x is not read via the high byte temporary register (TEMP). However, it is a good practice to read the low byte first as when accessing other 16-bit registers. Writing the OCR1x Registers must be done via the TEMP Register since the compare of all 16 bits is done continuously. The high byte (OCR1xH) has to be written first. When the high byte I/O location is written by the CPU, the TEMP Register will be updated by the value written. Then when the low byte (OCR1xL) is written to the lower eight bits, the high byte will be copied into the upper 8-bits of either the OCR1x buffer or OCR1x Compare Register in the same system clock cycle.

For more information of how to access the 16-bit registers refer to ”Accessing 16-bit Registers” on page 115.

13.7.1 Force Output Compare

In non-PWM Waveform Generation modes, the match output of the comparator can be forced by writing a one to the *Force Output Compare* (FOC1x) bit. Forcing compare match will not set the OCF1x Flag or reload/clear the timer, but the OC1x pin will be updated as if a real compare match had occurred (the COM11:0 bits settings define whether the OC1x pin is set, cleared or toggled).

13.7.2 Compare Match Blocking by TCNT1 Write

All CPU writes to the TCNT1 Register will block any compare match that occurs in the next timer clock cycle, even when the timer is stopped. This feature allows OCR1x to be initialized to the same value as TCNT1 without triggering an interrupt when the Timer/Counter clock is enabled.

13.7.3 Using the Output Compare Unit

Since writing TCNT1 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT1 when using any of the Output Compare channels, independent of whether the Timer/Counter is running or not. If the value written to TCNT1 equals the OCR1x value, the compare match will be missed, resulting in incorrect waveform generation. Do not write the TCNT1 equal to TOP in PWM modes with variable TOP values. The compare match for the TOP will be ignored and the counter will continue to 0xFFFF. Similarly, do not write the TCNT1 value equal to BOTTOM when the counter is downcounting.

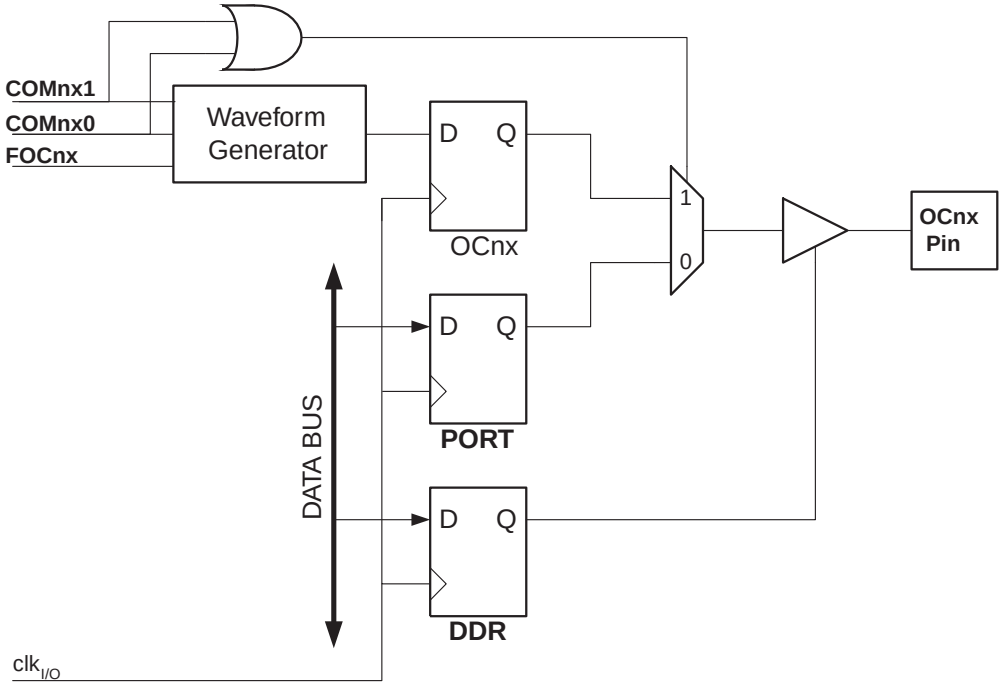
The setup of the OC1x should be performed before setting the Data Direction Register for the port pin to output. The easiest way of setting the OC1x value is to use the Force Output Compare (FOC1x) strobe bits in Normal mode. The OC1x Register keeps its value even when changing between Waveform Generation modes.

Be aware that the COM1x1:0 bits are not double buffered together with the compare value. Changing the COM1x1:0 bits will take effect immediately.

13.8 Compare Match Output Unit

The *Compare Output mode* (COM1x1:0) bits have two functions. The Waveform Generator uses the COM1x1:0 bits for defining the Output Compare (OC1x) state at the next compare match. Secondly the COM1x1:0 bits control the OC1x pin output source. Figure 13-5 shows a simplified schematic of the logic affected by the COM1x1:0 bit setting. The I/O Registers, I/O bits, and I/O pins in the figure are shown in bold. Only the parts of the general I/O Port Control Registers (DDR and PORT) that are affected by the COM1x1:0 bits are shown. When referring to the OC1x state, the reference is for the internal OC1x Register, not the OC1x pin. If a system reset occur, the OC1x Register is reset to “0”.

Figure 13-5. Compare Match Output Unit, Schematic



The general I/O port function is overridden by the Output Compare (OC1x) from the Waveform Generator if either of the COM1x1:0 bits are set. However, the OC1x pin direction (input or output) is still controlled by the *Data Direction Register* (DDR) for the port pin. The Data Direction Register bit for the OC1x pin (DDR_OC1x) must be set as output before the OC1x value is visible on the pin. The port override function is generally independent of the Waveform Generation mode, but there are some exceptions. Refer to Table 13-1, Table 13-2 and Table 13-3 for details.

The design of the Output Compare pin logic allows initialization of the OC1x state before the output is enabled. Note that some COM1x1:0 bit settings are reserved for certain modes of operation. See Section “13.11” on page 134.

The COM1x1:0 bits have no effect on the Input Capture unit.

13.8.1 Compare Output Mode and Waveform Generation

The Waveform Generator uses the COM1x1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM1x1:0 = 0 tells the Waveform Generator that no action on the OC1x Register is to be performed on the next compare match. For compare output actions in the

non-PWM modes refer to Table 13-1 on page 134. For fast PWM mode refer to Table 13-2 on page 135, and for phase correct and phase and frequency correct PWM refer to Table 13-3 on page 135.

A change of the COM1x1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC1x strobe bits.

13.9 Modes of Operation

The mode of operation, i.e., the behavior of the Timer/Counter and the Output Compare pins, is defined by the combination of the *Waveform Generation mode* (WGM13:0) and *Compare Output mode* (COM1x1:0) bits. The Compare Output mode bits do not affect the counting sequence, while the Waveform Generation mode bits do. The COM1x1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM1x1:0 bits control whether the output should be set, cleared or toggle at a compare match (See Section “13.8” on page 124.)

For detailed timing information refer to "Timer/Counter Timing Diagrams” on page 132.

13.9.1 Normal Mode

The simplest mode of operation is the *Normal mode* (WGM13:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 16-bit value (MAX = 0xFFFF) and then restarts from the BOTTOM (0x0000). In normal operation the *Timer/Counter Overflow Flag* (TOV1) will be set in the same timer clock cycle as the TCNT1 becomes zero. The TOV1 Flag in this case behaves like a 17th bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV1 Flag, the timer resolution can be increased by software. There are no special cases to consider in the Normal mode, a new counter value can be written anytime.

The Input Capture unit is easy to use in Normal mode. However, observe that the maximum interval between the external events must not exceed the resolution of the counter. If the interval between events are too long, the timer overflow interrupt or the prescaler must be used to extend the resolution for the capture unit.

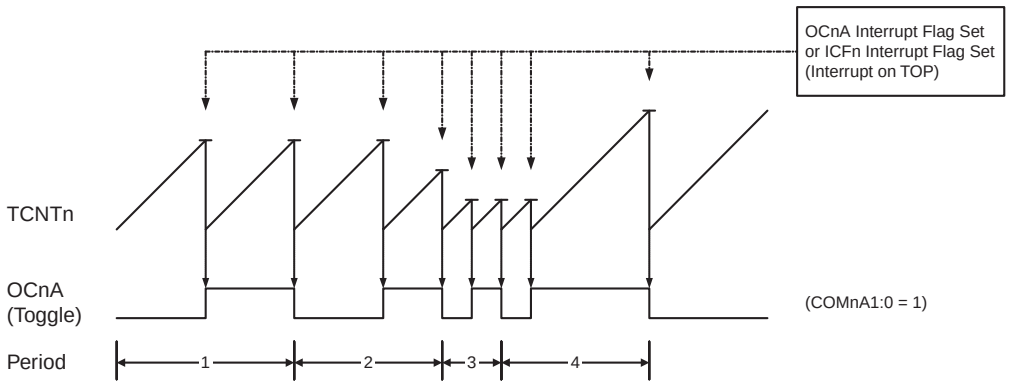
The Output Compare units can be used to generate interrupts at some given time. Using the Output Compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

13.9.2 Clear Timer on Compare Match (CTC) Mode

In *Clear Timer on Compare* or CTC mode (WGM13:0 = 4 or 12), the OCR1A or ICR1 Register are used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT1) matches either the OCR1A (WGM13:0 = 4) or the ICR1 (WGM13:0 = 12). The OCR1A or ICR1 define the top value for the counter, hence also its resolution. This mode allows greater control of the compare match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 13-6. The counter value (TCNT1) increases until a compare match occurs with either OCR1A or ICR1, and then counter (TCNT1) is cleared.

Figure 13-6. CTC Mode, Timing Diagram



An interrupt can be generated at each time the counter value reaches the TOP value by either using the OCF1A or ICF1 Flag according to the register used to define the TOP value. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing the TOP to a value close to BOTTOM when the counter is running with none or a low prescaler value must be done with care since the CTC mode does not have the double buffering feature. If the new value written to OCR1A or ICR1 is lower than the current value of TCNT1, the counter will miss the compare match. The counter will then have to count to its maximum value (0xFFFF) and wrap around starting at 0x0000 before the compare match can occur. In many cases this feature is not desirable. An alternative will then be to use the fast PWM mode using OCR1A for defining TOP (WGM13:0 = 15) since the OCR1A then will be double buffered.

For generating a waveform output in CTC mode, the OC1A output can be set to toggle its logical level on each compare match by setting the Compare Output mode bits to toggle mode (COM1A1:0 = 1). The OC1A value will not be visible on the port pin unless the data direction for the pin is set to output (DDR_OC1A = 1). The waveform generated will have a maximum frequency of $f_{OC1A} = f_{clk_I/O}/2$ when OCR1A is set to zero (0x0000). The waveform frequency is defined by the following equation:

$$f_{OCnA} = \frac{f_{clk_I/O}}{2 \cdot N \cdot (1 + OCRnA)}$$

The *N* variable represents the prescaler factor (1, 8, 64, 256, or 1024).

As for the Normal mode of operation, the TOV1 Flag is set in the same timer clock cycle that the counter counts from MAX to 0x0000.

13.9.3 Fast PWM Mode

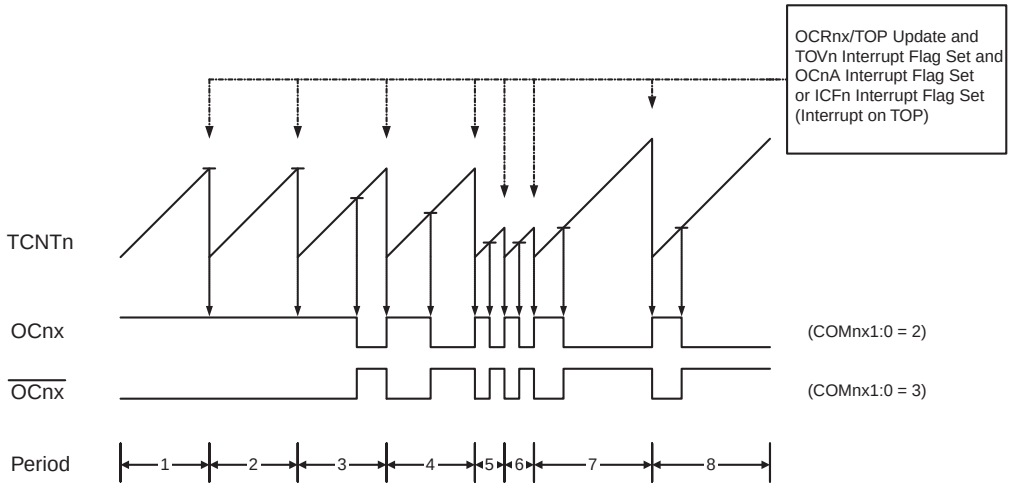
The *fast Pulse Width Modulation* or fast PWM mode (WGM13:0 = 5, 6, 7, 14, or 15) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM options by its single-slope operation. The counter counts from BOTTOM to TOP then restarts from BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x, and set at BOTTOM. In inverting Compare Output mode output is set on compare match and cleared at BOTTOM. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as high as the phase correct and phase and frequency correct PWM modes that use dual-slope operation. This high frequency makes the fast PWM mode well suited for power regulation, rectification, and DAC applications. High frequency allows physically small sized external components (coils, capacitors), hence reduces total system cost.

The PWM resolution for fast PWM can be fixed to 8-, 9-, or 10-bit, or defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated by using the following equation:

$$R_{FPWM} = \frac{\log(TOP+1)}{\log(2)}$$

In fast PWM mode the counter is incremented until the counter value matches either one of the fixed values 0x00FF, 0x01FF, or 0x03FF (WGM13:0 = 5, 6, or 7), the value in ICR1 (WGM13:0 = 14), or the value in OCR1A (WGM13:0 = 15). The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in Figure 13-7. The figure shows fast PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x Interrupt Flag will be set when a compare match occurs.

Figure 13-7. Fast PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV1) is set each time the counter reaches TOP. In addition the OC1A or ICF1 Flag is set at the same timer clock cycle as TOV1 is set when either OCR1A or ICR1 is used for defining the TOP value. If one of the interrupts are enabled, the interrupt handler routine can be used for updating the TOP and compare values.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the Compare Registers. If the TOP value is lower than any of the Compare Registers, a compare match will never occur between the TCNT1 and the OCR1x. Note that when using fixed TOP values the unused bits are masked to zero when any of the OCR1x Registers are written.

The procedure for updating ICR1 differs from updating OCR1A when used for defining the TOP value. The ICR1 Register is not double buffered. This means that if ICR1 is changed to a low value when the counter is running with none or a low prescaler value, there is a risk that the new ICR1 value written is lower than the current value of TCNT1. The result will then be that the counter will miss the compare match at the TOP value. The counter will then have to count to the MAX value (0xFFFF) and wrap around starting at 0x0000 before the compare match can occur. The OCR1A Register however, is double buffered. This feature allows the OCR1A I/O location

to be written anytime. When the OCR1A I/O location is written the value written will be put into the OCR1A Buffer Register. The OCR1A Compare Register will then be updated with the value in the Buffer Register at the next timer clock cycle the TCNT1 matches TOP. The update is done at the same timer clock cycle as the TCNT1 is cleared and the TOV1 Flag is set.

Using the ICR1 Register for defining TOP works well when using fixed TOP values. By using ICR1, the OCR1A Register is free to be used for generating a PWM output on OC1A. However, if the base PWM frequency is actively changed (by changing the TOP value), using the OCR1A as TOP is clearly a better choice due to its double buffer feature.

In fast PWM mode, the compare units allow generation of PWM waveforms on the OC1x pins. Setting the COM1x1:0 bits to two will produce a inverted PWM and an non-inverted PWM output can be generated by setting the COM1x1:0 to three (see Table on page 135). The actual OC1x value will only be visible on the port pin if the data direction for the port pin is set as output (DDR_OC1x). The PWM waveform is generated by setting (or clearing) the OC1x Register at the compare match between OCR1x and TCNT1, and clearing (or setting) the OC1x Register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

The PWM frequency for the output can be calculated by the following equation:

$$f_{OCnxPWM} = \frac{f_{clk_I/O}}{N \cdot (1 + TOP)}$$

The N variable represents the prescaler divider (1, 8, 64, 256, or 1024).

The extreme values for the OCR1x Register represents special cases when generating a PWM waveform output in the fast PWM mode. If the OCR1x is set equal to BOTTOM (0x0000) the output will be a narrow spike for each TOP+1 timer clock cycle. Setting the OCR1x equal to TOP will result in a constant high or low output (depending on the polarity of the output set by the COM1x1:0 bits.)

A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC1A to toggle its logical level on each compare match (COM1A1:0 = 1). This applies only if OCR1A is used to define the TOP value (WGM13:0 = 15). The waveform generated will have a maximum frequency of $f_{OC1A} = f_{clk_I/O}/2$ when OCR1A is set to zero (0x0000). This feature is similar to the OC1A toggle in CTC mode, except the double buffer feature of the Output Compare unit is enabled in the fast PWM mode.

13.9.4 Phase Correct PWM Mode

The *phase correct Pulse Width Modulation* or phase correct PWM mode (WGM13:0 = 1, 2, 3, 10, or 11) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is, like the phase and frequency correct PWM mode, based on a dual-slope operation. The counter counts repeatedly from BOTTOM (0x0000) to TOP and then from TOP to BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x while upcounting, and set on the compare match while downcounting. In inverting Output Compare mode, the operation is inverted. The dual-slope operation has lower maximum operation frequency than single slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

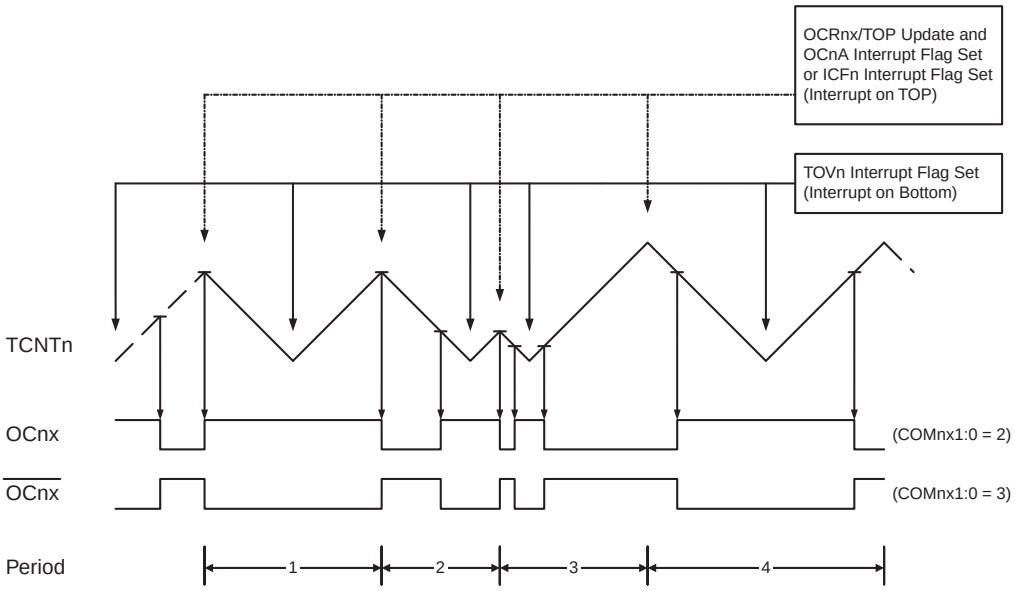
The PWM resolution for the phase correct PWM mode can be fixed to 8-, 9-, or 10-bit, or defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to

0x0003), and the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated by using the following equation:

$$R_{PCPWM} = \frac{\log(TOP + 1)}{\log(2)}$$

In phase correct PWM mode the counter is incremented until the counter value matches either one of the fixed values 0x00FF, 0x01FF, or 0x03FF (WGM13:0 = 1, 2, or 3), the value in ICR1 (WGM13:0 = 10), or the value in OCR1A (WGM13:0 = 11). The counter has then reached the TOP and changes the count direction. The TCNT1 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 13-8. The figure shows phase correct PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x Interrupt Flag will be set when a compare match occurs.

Figure 13-8. Phase Correct PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV1) is set each time the counter reaches BOTTOM. When either OCR1A or ICR1 is used for defining the TOP value, the OC1A or ICF1 Flag is set accordingly at the same timer clock cycle as the OCR1x Registers are updated with the double buffer value (at TOP). The Interrupt Flags can be used to generate an interrupt each time the counter reaches the TOP or BOTTOM value.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the Compare Registers. If the TOP value is lower than any of the Compare Registers, a compare match will never occur between the TCNT1 and the OCR1x. Note that when using fixed TOP values, the unused bits are masked to zero when any of the OCR1x Registers are written. As the third period shown in Figure 13-8 illustrates, changing the TOP actively while the Timer/Counter is running in the phase correct mode can result in an unsymmetrical output. The reason for this can be found in the time of update of the OCR1x Register. Since the OCR1x update occurs at TOP, the PWM period starts and ends at TOP. This

implies that the length of the falling slope is determined by the previous TOP value, while the length of the rising slope is determined by the new TOP value. When these two values differ the two slopes of the period will differ in length. The difference in length gives the unsymmetrical result on the output.

It is recommended to use the phase and frequency correct mode instead of the phase correct mode when changing the TOP value while the Timer/Counter is running. When using a static TOP value there are practically no differences between the two modes of operation.

In phase correct PWM mode, the compare units allow generation of PWM waveforms on the OC1x pins. Setting the COM1x1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM1x1:0 to three (See Table on page 135). The actual OC1x value will only be visible on the port pin if the data direction for the port pin is set as output (DDR_OC1x). The PWM waveform is generated by setting (or clearing) the OC1x Register at the compare match between OCR1x and TCNT1 when the counter increments, and clearing (or setting) the OC1x Register at compare match between OCR1x and TCNT1 when the counter decrements. The PWM frequency for the output when using phase correct PWM can be calculated by the following equation:

$$f_{OCnxPCPWM} = \frac{f_{clk_I/O}}{2 \cdot N \cdot TOP}$$

The N variable represents the prescaler divider (1, 8, 64, 256, or 1024).

The extreme values for the OCR1x Register represent special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR1x is set equal to BOTTOM the output will be continuously low and if set equal to TOP the output will be continuously high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values. If OCR1A is used to define the TOP value (WGM13:0 = 11) and COM1A1:0 = 1, the OC1A output will toggle with a 50% duty cycle.

13.9.5 Phase and Frequency Correct PWM Mode

The *phase and frequency correct Pulse Width Modulation,* or phase and frequency correct PWM mode (WGM13:0 = 8 or 9) provides a high resolution phase and frequency correct PWM waveform generation option. The phase and frequency correct PWM mode is, like the phase correct PWM mode, based on a dual-slope operation. The counter counts repeatedly from BOTTOM (0x0000) to TOP and then from TOP to BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x while upcounting, and set on the compare match while downcounting. In inverting Compare Output mode, the operation is inverted. The dual-slope operation gives a lower maximum operation frequency compared to the single-slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

The main difference between the phase correct, and the phase and frequency correct PWM mode is the time the OCR1x Register is updated by the OCR1x Buffer Register, (see Figure 13-8 and Figure 13-9).

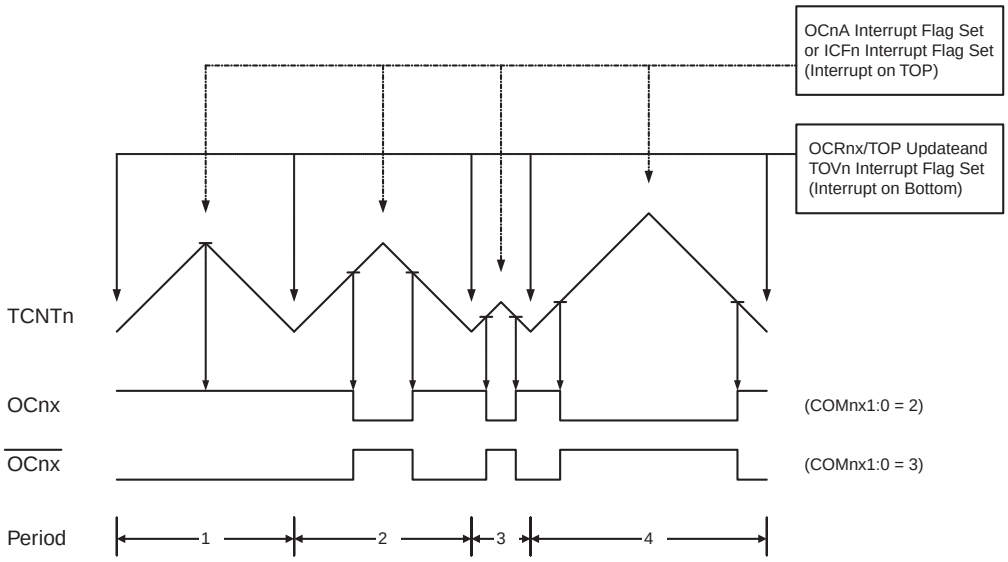
The PWM resolution for the phase and frequency correct PWM mode can be defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and

the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated using the following equation:

$$R_{PFCPWM} = \frac{\log(TOP+1)}{\log(2)}$$

In phase and frequency correct PWM mode the counter is incremented until the counter value matches either the value in ICR1 (WGM13:0 = 8), or the value in OCR1A (WGM13:0 = 9). The counter has then reached the TOP and changes the count direction. The TCNT1 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct and frequency correct PWM mode is shown on Figure 13-9. The figure shows phase and frequency correct PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x Interrupt Flag will be set when a compare match occurs.

Figure 13-9. Phase and Frequency Correct PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV1) is set at the same timer clock cycle as the OCR1x Registers are updated with the double buffer value (at BOTTOM). When either OCR1A or ICR1 is used for defining the TOP value, the OC1A or ICF1 Flag set when TCNT1 has reached TOP. The Interrupt Flags can then be used to generate an interrupt each time the counter reaches the TOP or BOTTOM value.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the Compare Registers. If the TOP value is lower than any of the Compare Registers, a compare match will never occur between the TCNT1 and the OCR1x.

As Figure 13-9 shows the output generated is, in contrast to the phase correct mode, symmetrical in all periods. Since the OCR1x Registers are updated at BOTTOM, the length of the rising and the falling slopes will always be equal. This gives symmetrical output pulses and is therefore frequency correct.

Using the ICR1 Register for defining TOP works well when using fixed TOP values. By using ICR1, the OCR1A Register is free to be used for generating a PWM output on OC1A. However, if the base PWM frequency is actively changed by changing the TOP value, using the OCR1A as TOP is clearly a better choice due to its double buffer feature.

In phase and frequency correct PWM mode, the compare units allow generation of PWM waveforms on the OC1x pins. Setting the COM1x1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM1x1:0 to three (See Table on page 135). The actual OC1x value will only be visible on the port pin if the data direction for the port pin is set as output (DDR_OC1x). The PWM waveform is generated by setting (or clearing) the OC1x Register at the compare match between OCR1x and TCNT1 when the counter increments, and clearing (or setting) the OC1x Register at compare match between OCR1x and TCNT1 when the counter decrements. The PWM frequency for the output when using phase and frequency correct PWM can be calculated by the following equation:

$$f_{OCnxPFCPWM} = \frac{f_{clk_I/O}}{2 \cdot N \cdot TOP}$$

The N variable represents the prescaler divider (1, 8, 64, 256, or 1024).

The extreme values for the OCR1x Register represents special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR1x is set equal to BOTTOM the output will be continuously low and if set equal to TOP the output will be set to high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values. If OCR1A is used to define the TOP value (WGM13:0 = 9) and COM1A1:0 = 1, the OC1A output will toggle with a 50% duty cycle.

13.10 Timer/Counter Timing Diagrams

The Timer/Counter is a synchronous design and the timer clock (clk_{T1}) is therefore shown as a clock enable signal in the following figures. The figures include information on when Interrupt Flags are set, and when the OCR1x Register is updated with the OCR1x buffer value (only for modes utilizing double buffering). Figure 13-10 shows a timing diagram for the setting of OCF1x.

Figure 13-10. Timer/Counter Timing Diagram, Setting of OCF1x, no Prescaling

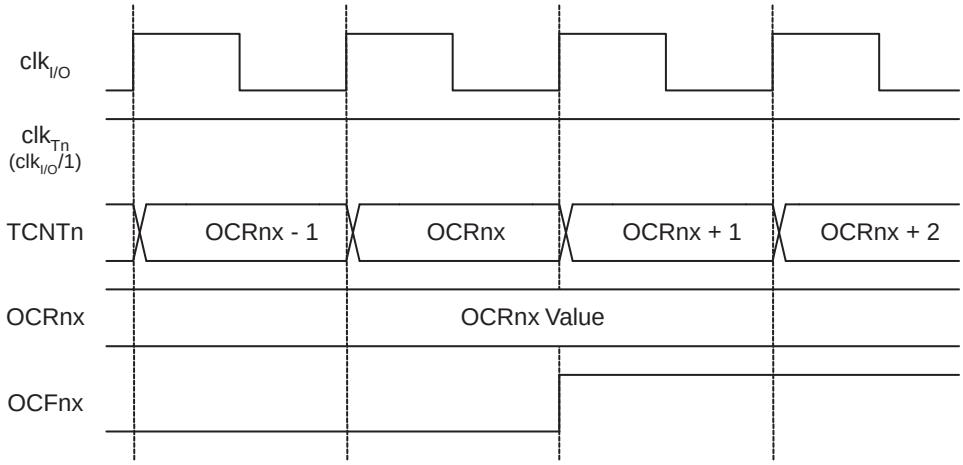


Figure 13-11 shows the same timing data, but with the prescaler enabled.

Figure 13-11. Timer/Counter Timing Diagram, Setting of OCF1x, with Prescaler ($f_{clk_I/O}/8$)

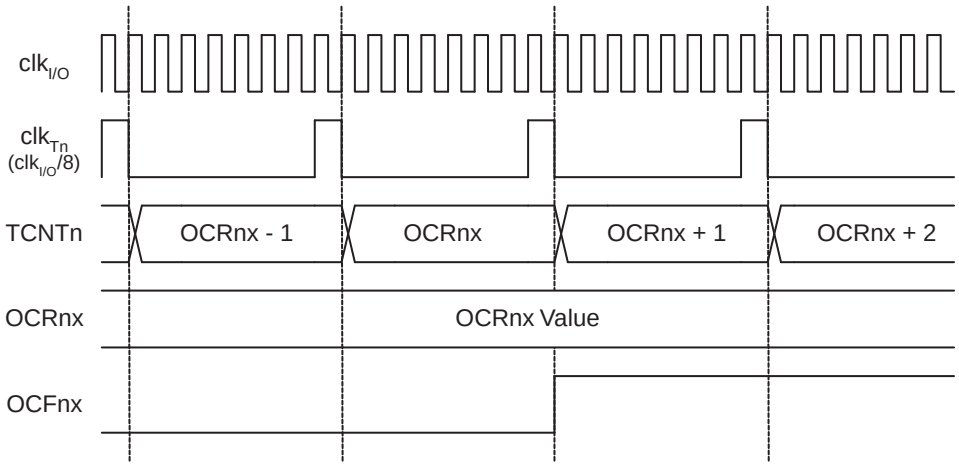


Figure 13-12 shows the count sequence close to TOP in various modes. When using phase and frequency correct PWM mode the OCR1x Register is updated at BOTTOM. The timing diagrams will be the same, but TOP should be replaced by BOTTOM, TOP-1 by BOTTOM+1 and so on. The same renaming applies for modes that set the TOV1 Flag at BOTTOM.

Figure 13-12. Timer/Counter Timing Diagram, no Prescaling

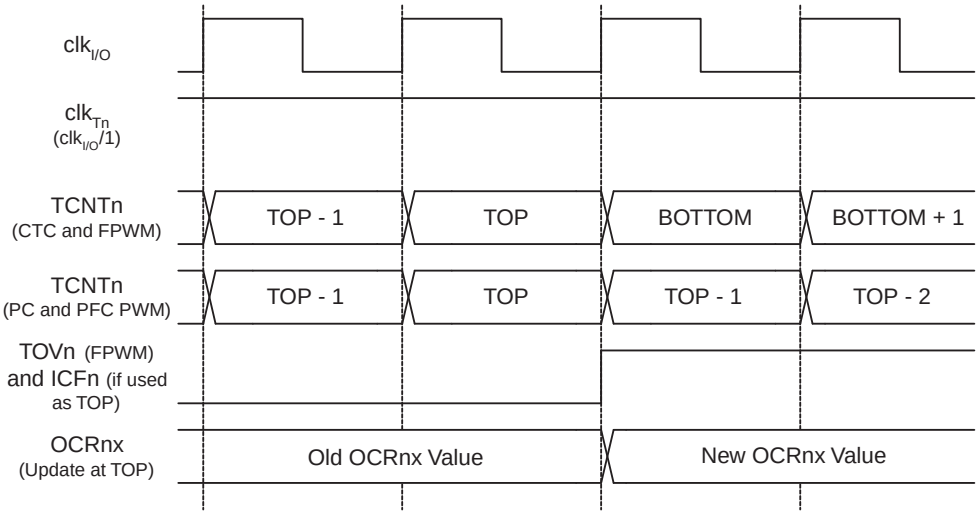
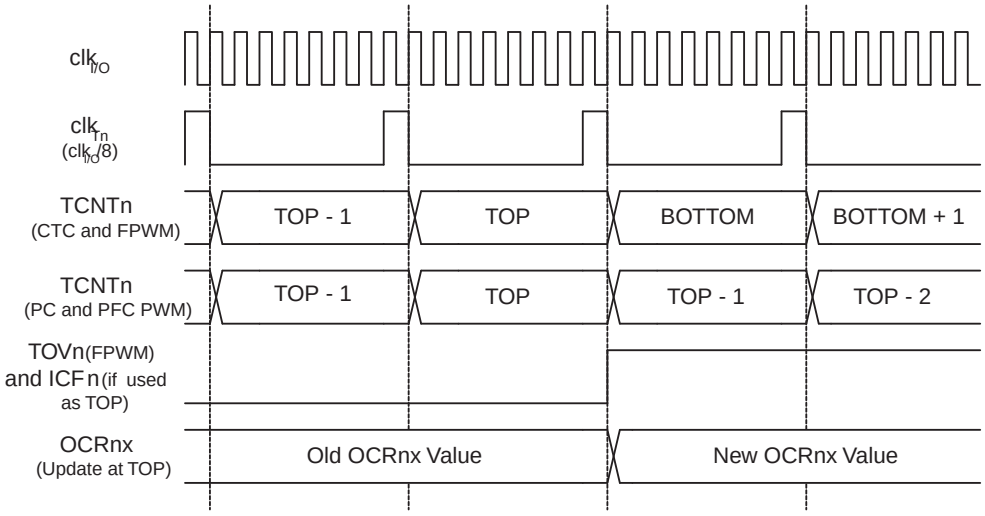


Figure 13-13 shows the same timing data, but with the prescaler enabled.

Figure 13-13. Timer/Counter Timing Diagram, with Prescaler ($f_{clk_I/O}/8$)



13.11 Register Description

13.11.1 TCCR1A – Timer/Counter1 Control Register A

Bit	7	6	5	4	3	2	1	0	
(0x80)	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:6 – COM1A1:0: Compare Output Mode for Channel A**

- **Bit 5:4 – COM1B1:0: Compare Output Mode for Channel B**

The COM1A1:0 and COM1B1:0 control the Output Compare pins (OC1A and OC1B respectively) behavior. If one or both of the COM1A1:0 bits are written to one, the OC1A output overrides the normal port functionality of the I/O pin it is connected to. If one or both of the COM1B1:0 bit are written to one, the OC1B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the *Data Direction Register* (DDR) bit corresponding to the OC1A or OC1B pin must be set in order to enable the output driver.

When the OC1A or OC1B is connected to the pin, the function of the COM1x1:0 bits is dependent of the WGM13:0 bits setting. Table 13-1 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to a Normal or a CTC mode (non-PWM).

Table 13-1. Compare Output Mode, non-PWM

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	Toggle OC1A/OC1B on Compare Match.
1	0	Clear OC1A/OC1B on Compare Match (Set output to low level).
1	1	Set OC1A/OC1B on Compare Match (Set output to high level).

Table 13-2 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the fast PWM mode.

Table 13-2. Compare Output Mode, Fast PWM[1]

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 14 or 15: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on Compare Match, set OC1A/OC1B at BOTTOM (non-inverting mode)
1	1	Set OC1A/OC1B on Compare Match, clear OC1A/OC1B at BOTTOM (inverting mode)

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. In this case the compare match is ignored, but the set or clear is done at BOTTOM. See Section “13.9.3” on page 126. for more details.

Table 13-3 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the phase correct or the phase and frequency correct, PWM mode.

Table 13-3. Compare Output Mode, Phase Correct and Phase and Frequency Correct PWM[1]

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 9 or 11: Toggle OC1A on Compare Match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on Compare Match when up-counting. Set OC1A/OC1B on Compare Match when downcounting.
1	1	Set OC1A/OC1B on Compare Match when up-counting. Clear OC1A/OC1B on Compare Match when downcounting.

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. See Section “13.9.4” on page 128. for more details.

- **Bit 1:0 – WGM11:0: Waveform Generation Mode**

Combined with the WGM13:2 bits found in the TCCR1B Register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used, see Table 13-4. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), Clear Timer on Compare match (CTC) mode, and three types of Pulse Width Modulation (PWM) modes. (See Section “13.9” on page 125.).

Table 13-4. Waveform Generation Mode Bit Description[1]

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	–	–	–
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

Note: 1. The CTC1 and PWM11:0 bit definition names are obsolete. Use the WGM12:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

13.11.2 TCCR1B – Timer/Counter1 Control Register B

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – ICNC1: Input Capture Noise Canceler**

Setting this bit (to one) activates the Input Capture Noise Canceler. When the noise canceler is activated, the input from the Input Capture pin (ICP1) is filtered. The filter function requires four successive equal valued samples of the ICP1 pin for changing its output. The Input Capture is therefore delayed by four Oscillator cycles when the noise canceler is enabled.

- **Bit 6 – ICES1: Input Capture Edge Select**

This bit selects which edge on the Input Capture pin (ICP1) that is used to trigger a capture event. When the ICES1 bit is written to zero, a falling (negative) edge is used as trigger, and when the ICES1 bit is written to one, a rising (positive) edge will trigger the capture.

When a capture is triggered according to the ICES1 setting, the counter value is copied into the Input Capture Register (ICR1). The event will also set the Input Capture Flag (ICF1), and this can be used to cause an Input Capture Interrupt, if this interrupt is enabled.

When the ICR1 is used as TOP value (see description of the WGM13:0 bits located in the TCCR1A and the TCCR1B Register), the ICP1 is disconnected and consequently the Input Capture function is disabled.

- **Bit 5 – Reserved Bit**

This bit is reserved for future use. For ensuring compatibility with future devices, this bit must be written to zero when TCCR1B is written.

- **Bit 4:3 – WGM13:2: Waveform Generation Mode**

See TCCR1A Register description.

- **Bit 2:0 – CS12:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter, see Figure 13-10 and Figure 13-11.

Table 13-5. Clock Select Bit Description

CS12	CS11	CS10	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter1, transitions on the T1 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

13.11.3 TCCR1C – Timer/Counter1 Control Register C

Bit	7	6	5	4	3	2	1	0	
(0x82)	FOC1A	FOC1B	–	–	–	–	–	–	TCCR1C
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – FOC1A: Force Output Compare for Channel A**

- **Bit 6 – FOC1B: Force Output Compare for Channel B**

The FOC1A/FOC1B bits are only active when the WGM13:0 bits specifies a non-PWM mode. When writing a logical one to the FOC1A/FOC1B bit, an immediate compare match is forced on the Waveform Generation unit. The OC1A/OC1B output is changed according to its COM1x1:0 bits setting. Note that the FOC1A/FOC1B bits are implemented as strobes. Therefore it is the value present in the COM1x1:0 bits that determine the effect of the forced compare.

A FOC1A/FOC1B strobe will not generate any interrupt nor will it clear the timer in Clear Timer on Compare match (CTC) mode using OCR1A as TOP. The FOC1A/FOC1B bits are always read as zero.

13.11.4 TCNT1H and TCNT1L – Timer/Counter1

Bit	7	6	5	4	3	2	1	0	
(0x85)	TCNT1[15:8]								TCNT1H
(0x84)	TCNT1[7:0]								TCNT1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The two *Timer/Counter* I/O locations (TCNT1H and TCNT1L, combined TCNT1) give direct access, both for read and for write operations, to the Timer/Counter unit 16-bit counter. To ensure that both the high and low bytes are read and written simultaneously when the CPU accesses these registers, the access is performed using an 8-bit temporary High Byte Register (TEMP). This temporary register is shared by all the other 16-bit registers. See Section “13.3” on page 115.

Modifying the counter (TCNT1) while the counter is running introduces a risk of missing a compare match between TCNT1 and one of the OCR1x Registers.

Writing to the TCNT1 Register blocks (removes) the compare match on the following timer clock for all compare units.

13.11.5 OCR1AH and OCR1AL – Output Compare Register 1 A

Bit	7	6	5	4	3	2	1	0	
(0x89)	OCR1A[15:8]								OCR1AH
(0x88)	OCR1A[7:0]								OCR1AL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

13.11.6 OCR1BH and OCR1BL – Output Compare Register 1 B

Bit	7	6	5	4	3	2	1	0	
(0x8B)	OCR1B[15:8]								OCR1BH
(0x8A)	OCR1B[7:0]								OCR1BL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Registers contain a 16-bit value that is continuously compared with the counter value (TCNT1). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC1x pin.

The Output Compare Registers are 16-bit in size. To ensure that both the high and low bytes are written simultaneously when the CPU writes to these registers, the access is performed using an 8-bit temporary High Byte Register (TEMP). This temporary register is shared by all the other 16-bit registers. See Section “13.3” on page 115.

13.11.7 ICR1H and ICR1L – Input Capture Register 1

Bit	7	6	5	4	3	2	1	0	
(0x87)	ICR1[15:8]								ICR1H
(0x86)	ICR1[7:0]								ICR1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Input Capture is updated with the counter (TCNT1) value each time an event occurs on the ICP1 pin (or optionally on the Analog Comparator output for Timer/Counter1). The Input Capture can be used for defining the counter TOP value.

The Input Capture Register is 16-bit in size. To ensure that both the high and low bytes are read simultaneously when the CPU accesses these registers, the access is performed using an 8-bit temporary High Byte Register (TEMP). This temporary register is shared by all the other 16-bit registers. See Section “13.3” on page 115.

13.11.8 TIMSK1 – Timer/Counter1 Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6F)	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	TIMSK1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7, 6 – Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 5 – ICIE1: Timer/Counter1, Input Capture Interrupt Enable**

When this bit is written to one, and the I-flag in the Status Register is set (interrupts globally enabled), the Timer/Counter1 Input Capture interrupt is enabled. The corresponding Interrupt Vector (see “Interrupts” on page 57) is executed when the ICF1 Flag, located in TIFR1, is set.

- **Bit 4, 3 – Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 2 – OCIE1B: Timer/Counter1, Output Compare B Match Interrupt Enable**

When this bit is written to one, and the I-flag in the Status Register is set (interrupts globally enabled), the Timer/Counter1 Output Compare B Match interrupt is enabled. The corresponding Interrupt Vector (see “Interrupts” on page 57) is executed when the OCF1B Flag, located in TIFR1, is set.

- **Bit 1 – OCIE1A: Timer/Counter1, Output Compare A Match Interrupt Enable**

When this bit is written to one, and the I-flag in the Status Register is set (interrupts globally enabled), the Timer/Counter1 Output Compare A Match interrupt is enabled. The corresponding Interrupt Vector (see “Interrupts” on page 57) is executed when the OCF1A Flag, located in TIFR1, is set.

- **Bit 0 – TOIE1: Timer/Counter1, Overflow Interrupt Enable**

When this bit is written to one, and the I-flag in the Status Register is set (interrupts globally enabled), the Timer/Counter1 Overflow interrupt is enabled. The corresponding Interrupt Vector (See ”Interrupts” on page 57) is executed when the TOV1 Flag, located in TIFR1, is set.

13.11.9 TIFR1 – Timer/Counter1 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x16 (0x36)	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1	TIFR1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7, 6 – Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 5 – ICF1: Timer/Counter1, Input Capture Flag**

This flag is set when a capture event occurs on the ICP1 pin. When the Input Capture Register (ICR1) is set by the WGM13:0 to be used as the TOP value, the ICF1 Flag is set when the counter reaches the TOP value.

ICF1 is automatically cleared when the Input Capture Interrupt Vector is executed. Alternatively, ICF1 can be cleared by writing a logic one to its bit location.

- **Bit 4, 3 – Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 2 – OCF1B: Timer/Counter1, Output Compare B Match Flag**

This flag is set in the timer clock cycle after the counter (TCNT1) value matches the Output Compare Register B (OCR1B).

Note that a Forced Output Compare (FOC1B) strobe will not set the OCF1B Flag.

OCF1B is automatically cleared when the Output Compare Match B Interrupt Vector is executed. Alternatively, OCF1B can be cleared by writing a logic one to its bit location.

- **Bit 1 – OCF1A: Timer/Counter1, Output Compare A Match Flag**

This flag is set in the timer clock cycle after the counter (TCNT1) value matches the Output Compare Register A (OCR1A).

Note that a Forced Output Compare (FOC1A) strobe will not set the OCF1A Flag.

OCF1A is automatically cleared when the Output Compare Match A Interrupt Vector is executed. Alternatively, OCF1A can be cleared by writing a logic one to its bit location.

- **Bit 0 – TOV1: Timer/Counter1, Overflow Flag**

The setting of this flag is dependent of the WGM13:0 bits setting. In Normal and CTC modes, the TOV1 Flag is set when the timer overflows. Refer to Table 13-4 on page 136 for the TOV1 Flag behavior when using another WGM13:0 bit setting.

TOV1 is automatically cleared when the Timer/Counter1 Overflow Interrupt Vector is executed. Alternatively, TOV1 can be cleared by writing a logic one to its bit location.

14. Timer/Counter0 and Timer/Counter1 Prescalers

"8-bit Timer/Counter0 with PWM" on page 94 and "16-bit Timer/Counter1 with PWM" on page 113 share the same prescaler module, but the Timer/Counters can have different prescaler settings. The description below applies to both Timer/Counter1 and Timer/Counter0.

14.1 Internal Clock Source

The Timer/Counter can be clocked directly by the system clock (by setting the CSn2:0 = 1). This provides the fastest operation, with a maximum Timer/Counter clock frequency equal to system clock frequency ($f_{CLK_I/O}$). Alternatively, one of four taps from the prescaler can be used as a clock source. The prescaled clock has a frequency of either $f_{CLK_I/O}/8$, $f_{CLK_I/O}/64$, $f_{CLK_I/O}/256$, or $f_{CLK_I/O}/1024$.

14.2 Prescaler Reset

The prescaler is free running, i.e., operates independently of the Clock Select logic of the Timer/Counter, and it is shared by Timer/Counter1 and Timer/Counter0. Since the prescaler is not affected by the Timer/Counter's clock select, the state of the prescaler will have implications for situations where a prescaled clock is used. One example of prescaling artifacts occurs when the timer is enabled and clocked by the prescaler (6 > CSn2:0 > 1). The number of system clock cycles from when the timer is enabled to the first count occurs can be from 1 to N+1 system clock cycles, where N equals the prescaler divisor (8, 64, 256, or 1024).

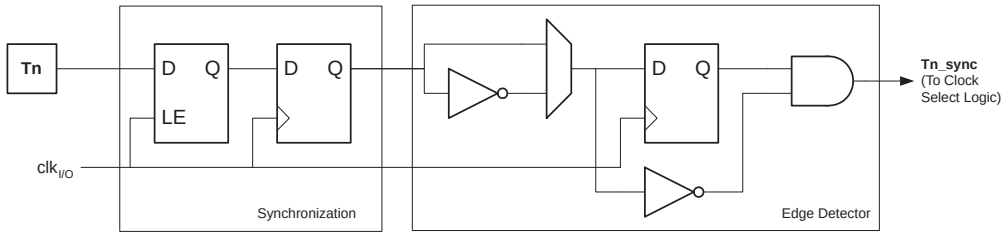
It is possible to use the prescaler reset for synchronizing the Timer/Counter to program execution. However, care must be taken if the other Timer/Counter that shares the same prescaler also uses prescaling. A prescaler reset will affect the prescaler period for all Timer/Counters it is connected to.

14.3 External Clock Source

An external clock source applied to the T1/T0 pin can be used as Timer/Counter clock (clk_{T1}/clk_{T0}). The T1/T0 pin is sampled once every system clock cycle by the pin synchronization logic. The synchronized (sampled) signal is then passed through the edge detector. Figure 14-1 shows a functional equivalent block diagram of the T1/T0 synchronization and edge detector logic. The registers are clocked at the positive edge of the internal system clock ($clk_{I/O}$). The latch is transparent in the high period of the internal system clock.

The edge detector generates one clk_{T1}/clk_{T0} pulse for each positive (CSn2:0 = 7) or negative (CSn2:0 = 6) edge it detects.

Figure 14-1. T1/T0 Pin Sampling



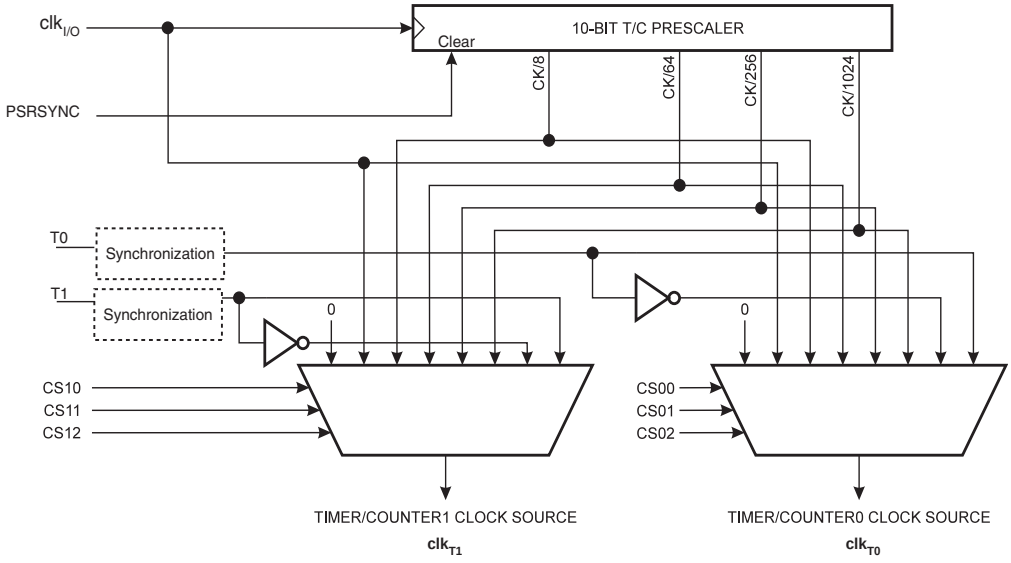
The synchronization and edge detector logic introduces a delay of 2.5 to 3.5 system clock cycles from an edge has been applied to the T1/T0 pin to the counter is updated.

Enabling and disabling of the clock input must be done when T1/T0 has been stable for at least one system clock cycle, otherwise it is a risk that a false Timer/Counter clock pulse is generated.

Each half period of the external clock applied must be longer than one system clock cycle to ensure correct sampling. The external clock must be guaranteed to have less than half the system clock frequency ($f_{ExtClk} < f_{clk_I/O}/2$) given a 50/50% duty cycle. Since the edge detector uses sampling, the maximum frequency of an external clock it can detect is half the sampling frequency (Nyquist sampling theorem). However, due to variation of the system clock frequency and duty cycle caused by Oscillator source (crystal, resonator, and capacitors) tolerances, it is recommended that maximum frequency of an external clock source is less than $f_{clk_I/O}/2.5$.

An external clock source can not be prescaled.

Figure 14-2. Prescaler for Timer/Counter0 and Timer/Counter1[(1)]



Note: 1. The synchronization logic on the input pins (T1/T0) is shown in Figure 14-1.

14.4 Register Description

14.4.1 GTCCR – General Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
0x23 (0x43)	TSM	–	–	–	–	–	PSRASY	PSRSYNC	GTCCR
Read/Write	R/W	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – TSM: Timer/Counter Synchronization Mode**

Writing the TSM bit to one activates the Timer/Counter Synchronization mode. In this mode, the value that is written to the PSRASY and PSRSYNC bits is kept, hence keeping the corresponding prescaler reset signals asserted. This ensures that the corresponding Timer/Counters are halted and can be configured to the same value without the risk of one of them advancing during configuration. When the TSM bit is written to zero, the PSRASY and PSRSYNC bits are cleared by hardware, and the Timer/Counters start counting simultaneously.

- **Bit 0 – PSRSYNC: Prescaler Reset**

When this bit is one, Timer/Counter1 and Timer/Counter0 prescaler will be Reset. This bit is normally cleared immediately by hardware, except if the TSM bit is set. Note that Timer/Counter1 and Timer/Counter0 share the same prescaler and a reset of this prescaler will affect both timers.

15. 8-bit Timer/Counter2 with PWM and Asynchronous Operation

15.1 Features

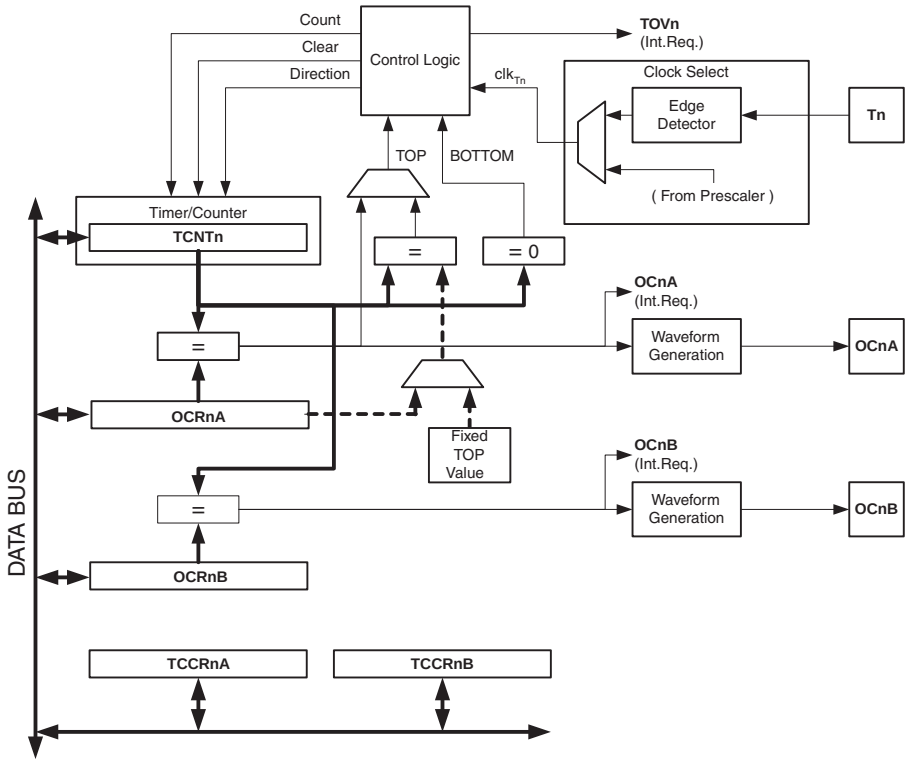
- **Single Channel Counter**
- **Clear Timer on Compare Match (Auto Reload)**
- **Glitch-free, Phase Correct Pulse Width Modulator (PWM)**
- **Frequency Generator**
- **10-bit Clock Prescaler**
- **Overflow and Compare Match Interrupt Sources (TOV2, OCF2A and OCF2B)**
- **Allows Clocking from External 32 kHz Watch Crystal Independent of the I/O Clock**

15.2 Overview

Timer/Counter2 is a general purpose, single channel, 8-bit Timer/Counter module. A simplified block diagram of the 8-bit Timer/Counter is shown in Figure 15-1. For the actual placement of I/O pins, refer to "Pinout ATmega48P/88P/168P/328P" on page 2. CPU accessible I/O Registers, including I/O bits and I/O pins, are shown in bold. The device-specific I/O Register and bit locations are listed in the "Register Description" on page 158.

The PRTIM2 bit in "Minimizing Power Consumption" on page 42 must be written to zero to enable Timer/Counter2 module.

Figure 15-1. 8-bit Timer/Counter Block Diagram



15.2.1 Registers

The Timer/Counter (TCNT2) and Output Compare Register (OCR2A and OCR2B) are 8-bit registers. Interrupt request (shorten as Int.Req.) signals are all visible in the Timer Interrupt Flag Register (TIFR2). All interrupts are individually masked with the Timer Interrupt Mask Register (TIMSK2). TIFR2 and TIMSK2 are not shown in the figure.

The Timer/Counter can be clocked internally, via the prescaler, or asynchronously clocked from the TOSC1/2 pins, as detailed later in this section. The asynchronous operation is controlled by the Asynchronous Status Register (ASSR). The Clock Select logic block controls which clock source he Timer/Counter uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the Clock Select logic is referred to as the timer clock (clk_{T2}).

The double buffered Output Compare Register (OCR2A and OCR2B) are compared with the Timer/Counter value at all times. The result of the compare can be used by the Waveform Generator to generate a PWM or variable frequency output on the Output Compare pins (OC2A and OC2B). See Section “15.5” on page 146. for details. The compare match event will also set the Compare Flag (OCF2A or OCF2B) which can be used to generate an Output Compare interrupt request.

15.2.2 Definitions

Many register and bit references in this document are written in general form. A lower case “n” replaces the Timer/Counter number, in this case 2. However, when using the register or bit defines in a program, the precise form must be used, i.e., TCNT2 for accessing Timer/Counter2 counter value and so on.

The definitions in Table 15-1 are also used extensively throughout the section.

Table 15-1. Definitions

BOTTOM	The counter reaches the BOTTOM when it becomes zero (0x00).
MAX	The counter reaches its MAXimum when it becomes 0xFF (decimal 255).
TOP	The counter reaches the TOP when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be the fixed value 0xFF (MAX) or the value stored in the OCR2A Register. The assignment is dependent on the mode of operation.

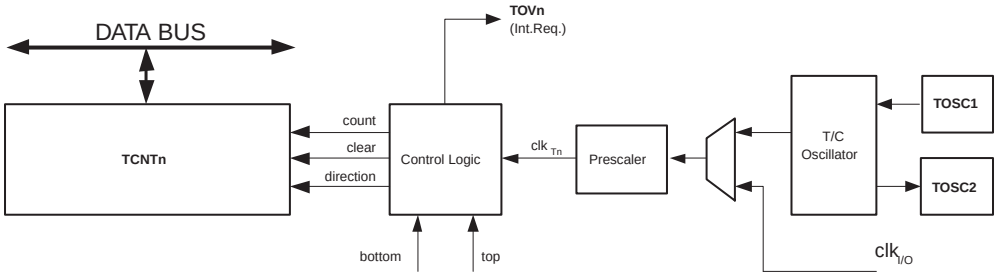
15.3 Timer/Counter Clock Sources

The Timer/Counter can be clocked by an internal synchronous or an external asynchronous clock source. The clock source clk_{T2} is by default equal to the MCU clock, $clk_{I/O}$. When the AS2 bit in the ASSR Register is written to logic one, the clock source is taken from the Timer/Counter Oscillator connected to TOSC1 and TOSC2. For details on asynchronous operation, see ”ASSR – Asynchronous Status Register” on page 164. For details on clock sources and prescaler, see ”Timer/Counter Prescaler” on page 156.

15.4 Counter Unit

The main part of the 8-bit Timer/Counter is the programmable bi-directional counter unit. Figure 15-2 on page 146 shows a block diagram of the counter and its surrounding environment.

Figure 15-2. Counter Unit Block Diagram



Signal description (internal signals):

count	Increment or decrement TCNT2 by 1.
direction	Selects between increment and decrement.
clear	Clear TCNT2 (set all bits to zero).
clk_{Tn}	Timer/Counter clock, referred to as clk_{T2} in the following.
top	Signalizes that TCNT2 has reached maximum value.
bottom	Signalizes that TCNT2 has reached minimum value (zero).

Depending on the mode of operation used, the counter is cleared, incremented, or decremented at each timer clock (clk_{T2}). clk_{T2} can be generated from an external or internal clock source, selected by the Clock Select bits (CS22:0). When no clock source is selected (CS22:0 = 0) the timer is stopped. However, the TCNT2 value can be accessed by the CPU, regardless of whether clk_{T2} is present or not. A CPU write overrides (has priority over) all counter clear or count operations.

The counting sequence is determined by the setting of the WGM21 and WGM20 bits located in the Timer/Counter Control Register (TCCR2A) and the WGM22 located in the Timer/Counter Control Register B (TCCR2B). There are close connections between how the counter behaves (counts) and how waveforms are generated on the Output Compare outputs OC2A and OC2B. For more details about advanced counting sequences and waveform generation, see ”Modes of Operation” on page 149.

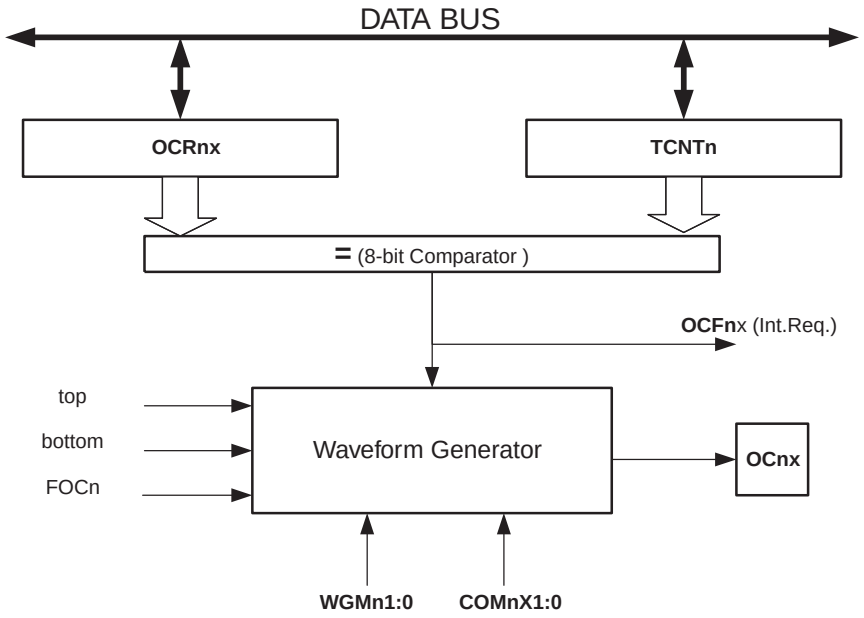
The Timer/Counter Overflow Flag (TOV2) is set according to the mode of operation selected by the WGM22:0 bits. TOV2 can be used for generating a CPU interrupt.

15.5 Output Compare Unit

The 8-bit comparator continuously compares TCNT2 with the Output Compare Register (OCR2A and OCR2B). Whenever TCNT2 equals OCR2A or OCR2B, the comparator signals a match. A match will set the Output Compare Flag (OCF2A or OCF2B) at the next timer clock cycle. If the corresponding interrupt is enabled, the Output Compare Flag generates an Output Compare interrupt. The Output Compare Flag is automatically cleared when the interrupt is executed. Alternatively, the Output Compare Flag can be cleared by software by writing a logical one to its I/O bit location. The Waveform Generator uses the match signal to generate an output according to operating mode set by the WGM22:0 bits and Compare Output mode (COM2x1:0) bits. The max and bottom signals are used by the Waveform Generator for handling the special cases of the extreme values in some modes of operation (”Modes of Operation” on page 149).

Figure 15-3 shows a block diagram of the Output Compare unit.

Figure 15-3. Output Compare Unit, Block Diagram



The OCR2x Register is double buffered when using any of the Pulse Width Modulation (PWM) modes. For the Normal and Clear Timer on Compare (CTC) modes of operation, the double buffering is disabled. The double buffering synchronizes the update of the OCR2x Compare Register to either top or bottom of the counting sequence. The synchronization prevents the occurrence of odd-length, non-symmetrical PWM pulses, thereby making the output glitch-free.

The OCR2x Register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCR2x Buffer Register, and if double buffering is disabled the CPU will access the OCR2x directly.

15.5.1 Force Output Compare

In non-PWM waveform generation modes, the match output of the comparator can be forced by writing a one to the Force Output Compare (FOC2x) bit. Forcing compare match will not set the OCF2x Flag or reload/clear the timer, but the OC2x pin will be updated as if a real compare match had occurred (the COM2x1:0 bits settings define whether the OC2x pin is set, cleared or toggled).

15.5.2 Compare Match Blocking by TCNT2 Write

All CPU write operations to the TCNT2 Register will block any compare match that occurs in the next timer clock cycle, even when the timer is stopped. This feature allows OCR2x to be initialized to the same value as TCNT2 without triggering an interrupt when the Timer/Counter clock is enabled.

15.5.3 Using the Output Compare Unit

Since writing TCNT2 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT2 when using the Output Compare channel, independently of whether the Timer/Counter is running or not. If the value written to TCNT2 equals the OCR2x value, the compare match will be missed, resulting in incorrect waveform generation. Similarly, do not write the TCNT2 value equal to BOTTOM when the counter is downcounting.

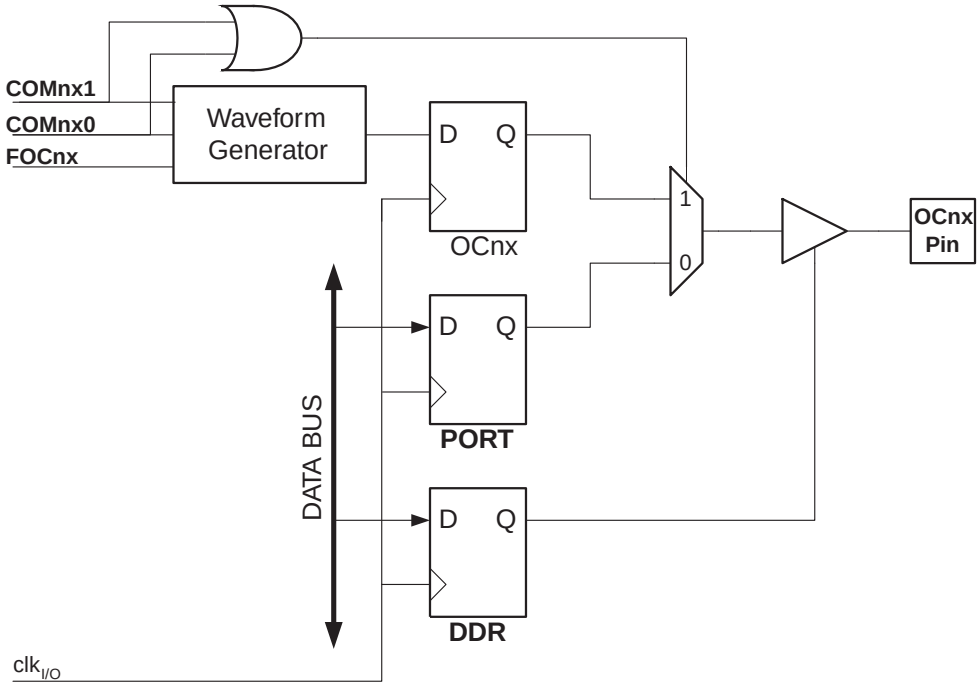
The setup of the OC2x should be performed before setting the Data Direction Register for the port pin to output. The easiest way of setting the OC2x value is to use the Force Output Compare (FOC2x) strobe bit in Normal mode. The OC2x Register keeps its value even when changing between Waveform Generation modes.

Be aware that the COM2x1:0 bits are not double buffered together with the compare value. Changing the COM2x1:0 bits will take effect immediately.

15.6 Compare Match Output Unit

The Compare Output mode (COM2x1:0) bits have two functions. The Waveform Generator uses the COM2x1:0 bits for defining the Output Compare (OC2x) state at the next compare match. Also, the COM2x1:0 bits control the OC2x pin output source. Figure 15-4 shows a simplified schematic of the logic affected by the COM2x1:0 bit setting. The I/O Registers, I/O bits, and I/O pins in the figure are shown in bold. Only the parts of the general I/O Port Control Registers (DDR and PORT) that are affected by the COM2x1:0 bits are shown. When referring to the OC2x state, the reference is for the internal OC2x Register, not the OC2x pin.

Figure 15-4. Compare Match Output Unit, Schematic



The general I/O port function is overridden by the Output Compare (OC2x) from the Waveform Generator if either of the COM2x1:0 bits are set. However, the OC2x pin direction (input or output) is still controlled by the Data Direction Register (DDR) for the port pin. The Data Direction Register bit for the OC2x pin (DDR_OC2x) must be set as output before the OC2x value is visible on the pin. The port override function is independent of the Waveform Generation mode.

The design of the Output Compare pin logic allows initialization of the OC2x state before the output is enabled. Note that some COM2x1:0 bit settings are reserved for certain modes of operation. See Section “15.11” on page 158.

15.6.1 Compare Output Mode and Waveform Generation

The Waveform Generator uses the COM2x1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM2x1:0 = 0 tells the Waveform Generator that no action on the OC2x Register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 15-5 on page 159. For fast PWM mode, refer to Table 15-6 on page 159, and for phase correct PWM refer to Table 15-7 on page 160.

A change of the COM2x1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC2x strobe bits.

15.7 Modes of Operation

The mode of operation, i.e., the behavior of the Timer/Counter and the Output Compare pins, is defined by the combination of the Waveform Generation mode (WGM22:0) and Compare Output mode (COM2x1:0) bits. The Compare Output mode bits do not affect the counting sequence, while the Waveform Generation mode bits do. The COM2x1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM2x1:0 bits control whether the output should be set, cleared, or toggled at a compare match (See Section “15.6” on page 148.).

For detailed timing information refer to ”Timer/Counter Timing Diagrams” on page 153.

15.7.1 Normal Mode

The simplest mode of operation is the Normal mode (WGM22:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 8-bit value (TOP = 0xFF) and then restarts from the bottom (0x00). In normal operation the Timer/Counter Overflow Flag (TOV2) will be set in the same timer clock cycle as the TCNT2 becomes zero. The TOV2 Flag in this case behaves like a ninth bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV2 Flag, the timer resolution can be increased by software. There are no special cases to consider in the Normal mode, a new counter value can be written anytime.

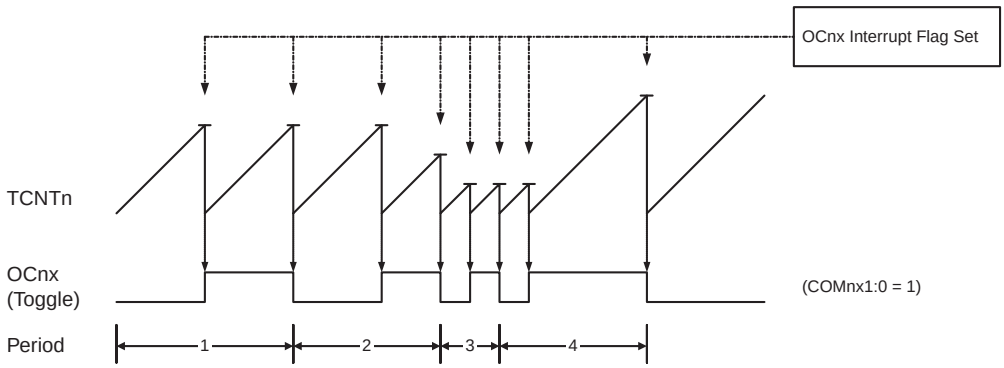
The Output Compare unit can be used to generate interrupts at some given time. Using the Output Compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

15.7.2 Clear Timer on Compare Match (CTC) Mode

In Clear Timer on Compare or CTC mode (WGM22:0 = 2), the OCR2A Register is used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT2) matches the OCR2A. The OCR2A defines the top value for the counter, hence also its resolution. This mode allows greater control of the compare match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 15-5. The counter value (TCNT2) increases until a compare match occurs between TCNT2 and OCR2A, and then counter (TCNT2) is cleared.

Figure 15-5. CTC Mode, Timing Diagram



An interrupt can be generated each time the counter value reaches the TOP value by using the OCF2A Flag. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing TOP to a value close to BOTTOM when the counter is running with none or a low prescaler value must be done with care since the CTC mode does not have the double buffering feature. If the new value written to OCR2A is lower than the current value of TCNT2, the counter will miss the compare match. The counter will then have to count to its maximum value (0xFF) and wrap around starting at 0x00 before the compare match can occur.

For generating a waveform output in CTC mode, the OC2A output can be set to toggle its logical level on each compare match by setting the Compare Output mode bits to toggle mode (COM2A1:0 = 1). The OC2A value will not be visible on the port pin unless the data direction for the pin is set to output. The waveform generated will have a maximum frequency of $f_{OC2A} = f_{clk_I/O}/2$ when OCR2A is set to zero (0x00). The waveform frequency is defined by the following equation:

$$f_{OCnx} = \frac{f_{clk_I/O}}{2 \cdot N \cdot (1 + OCRnx)}$$

The *N* variable represents the prescale factor (1, 8, 32, 64, 128, 256, or 1024).

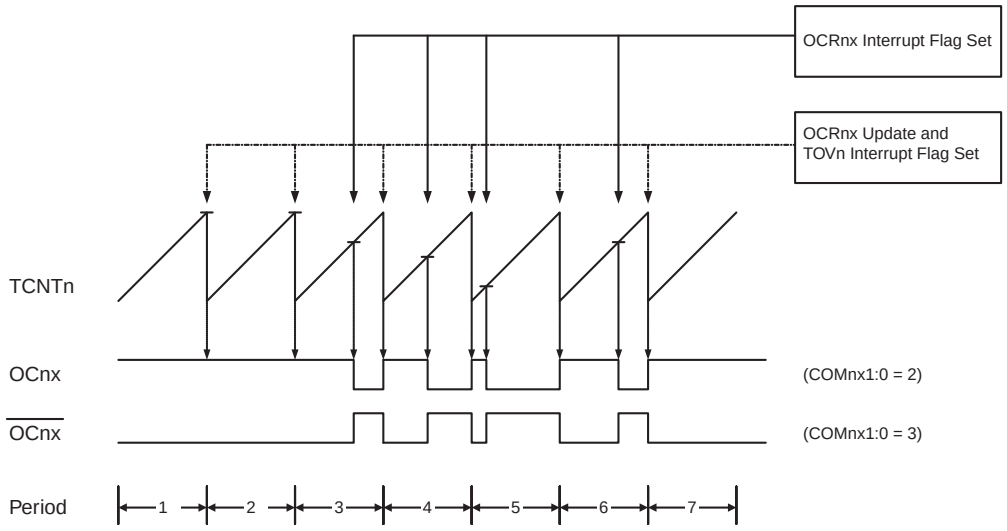
As for the Normal mode of operation, the TOV2 Flag is set in the same timer clock cycle that the counter counts from MAX to 0x00.

15.7.3 Fast PWM Mode

The fast Pulse Width Modulation or fast PWM mode (WGM22:0 = 3 or 7) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM option by its single-slope operation. The counter counts from BOTTOM to TOP then restarts from BOTTOM. TOP is defined as 0xFF when WGM2:0 = 3, and OCR2A when MGM2:0 = 7. In non-inverting Compare Output mode, the Output Compare (OC2x) is cleared on the compare match between TCNT2 and OCR2x, and set at BOTTOM. In inverting Compare Output mode, the output is set on compare match and cleared at BOTTOM. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as high as the phase correct PWM mode that uses dual-slope operation. This high frequency makes the fast PWM mode well suited for power regulation, rectification, and DAC applications. High frequency allows physically small sized external components (coils, capacitors), and therefore reduces total system cost.

In fast PWM mode, the counter is incremented until the counter value matches the TOP value. The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in Figure 15-6. The TCNT2 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT2 slopes represent compare matches between OCR2x and TCNT2.

Figure 15-6. Fast PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV2) is set each time the counter reaches TOP. If the interrupt is enabled, the interrupt handler routine can be used for updating the compare value.

In fast PWM mode, the compare unit allows generation of PWM waveforms on the OC2x pin. Setting the COM2x1:0 bits to two will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM2x1:0 to three. TOP is defined as 0xFF when WGM2:0 = 3, and OCR2A when MGM2:0 = 7. (See Table 15-3 on page 158). The actual OC2x value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by setting (or clearing) the OC2x Register at the compare match between OCR2x and TCNT2, and clearing (or setting) the OC2x Register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

The PWM frequency for the output can be calculated by the following equation:

$$f_{OCnxPWM} = \frac{f_{clk_I/O}}{N \cdot 256}$$

The *N* variable represents the prescale factor (1, 8, 32, 64, 128, 256, or 1024).

The extreme values for the OCR2A Register represent special cases when generating a PWM waveform output in the fast PWM mode. If the OCR2A is set equal to BOTTOM, the output will be a narrow spike for each MAX+1 timer clock cycle. Setting the OCR2A equal to MAX will result in a constantly high or low output (depending on the polarity of the output set by the COM2A1:0 bits.)

A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC2x to toggle its logical level on each compare match (COM2x1:0 = 1). The waveform

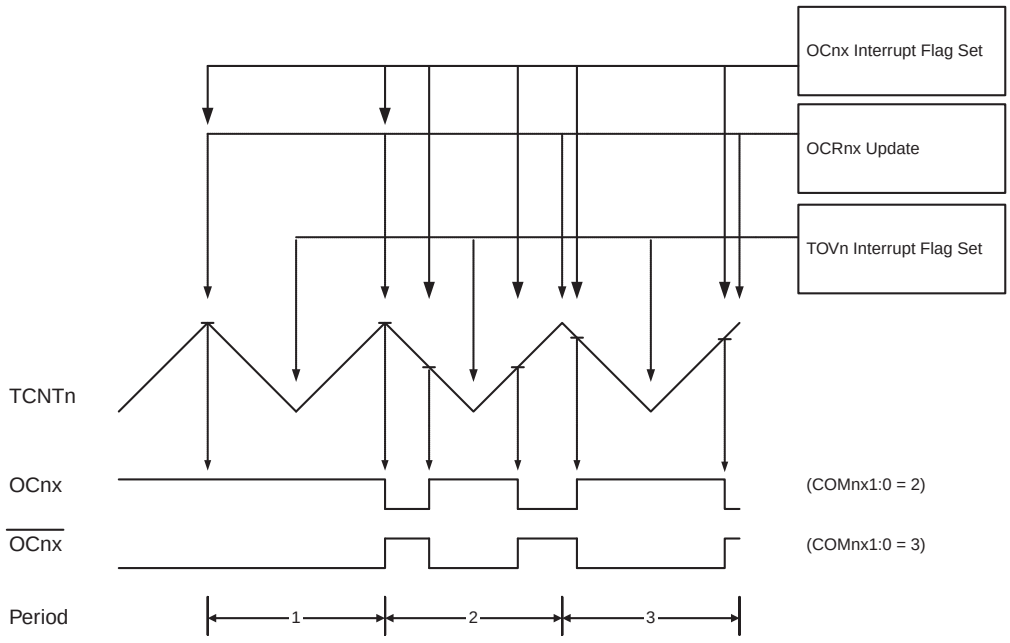
generated will have a maximum frequency of $f_{oc2} = f_{clk_I/O}/2$ when OCR2A is set to zero. This feature is similar to the OC2A toggle in CTC mode, except the double buffer feature of the Output Compare unit is enabled in the fast PWM mode.

15.7.4 Phase Correct PWM Mode

The phase correct PWM mode (WGM22:0 = 1 or 5) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is based on a dual-slope operation. The counter counts repeatedly from BOTTOM to TOP and then from TOP to BOTTOM. TOP is defined as 0xFF when WGM2:0 = 3, and OCR2A when MGM2:0 = 7. In non-inverting Compare Output mode, the Output Compare (OC2x) is cleared on the compare match between TCNT2 and OCR2x while upcounting, and set on the compare match while downcounting. In inverting Output Compare mode, the operation is inverted. The dual-slope operation has lower maximum operation frequency than single slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

In phase correct PWM mode the counter is incremented until the counter value matches TOP. When the counter reaches TOP, it changes the count direction. The TCNT2 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 15-7. The TCNT2 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT2 slopes represent compare matches between OCR2x and TCNT2.

Figure 15-7. Phase Correct PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV2) is set each time the counter reaches BOTTOM. The Interrupt Flag can be used to generate an interrupt each time the counter reaches the BOTTOM value.

In phase correct PWM mode, the compare unit allows generation of PWM waveforms on the OC2x pin. Setting the COM2x1:0 bits to two will produce a non-inverted PWM. An inverted PWM

output can be generated by setting the COM2x1:0 to three. TOP is defined as 0xFF when WGM2:0 = 3, and OCR2A when MGM2:0 = 7 (See Table 15-4 on page 159). The actual OC2x value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by clearing (or setting) the OC2x Register at the compare match between OCR2x and TCNT2 when the counter increments, and setting (or clearing) the OC2x Register at compare match between OCR2x and TCNT2 when the counter decrements. The PWM frequency for the output when using phase correct PWM can be calculated by the following equation:

$$f_{OCnxPCPWM} = \frac{f_{clk_I/O}}{N \cdot 510}$$

The *N* variable represents the prescale factor (1, 8, 32, 64, 128, 256, or 1024).

The extreme values for the OCR2A Register represent special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR2A is set equal to BOTTOM, the output will be continuously low and if set equal to MAX the output will be continuously high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

At the very start of period 2 in Figure 15-7 OCnx has a transition from high to low even though there is no Compare Match. The point of this transition is to guarantee symmetry around BOTTOM. There are two cases that give a transition without Compare Match.

- OCR2A changes its value from MAX, like in Figure 15-7. When the OCR2A value is MAX the OCn pin value is the same as the result of a down-counting compare match. To ensure symmetry around BOTTOM the OCn value at MAX must correspond to the result of an up-counting Compare Match.
- The timer starts counting from a value higher than the one in OCR2A, and for that reason misses the Compare Match and hence the OCn change that would have happened on the way up.

15.8 Timer/Counter Timing Diagrams

The following figures show the Timer/Counter in synchronous mode, and the timer clock (clk_{T2}) is therefore shown as a clock enable signal. In asynchronous mode, $clk_{I/O}$ should be replaced by the Timer/Counter Oscillator clock. The figures include information on when Interrupt Flags are set. Figure 15-8 contains timing data for basic Timer/Counter operation. The figure shows the count sequence close to the MAX value in all modes other than phase correct PWM mode.

Figure 15-8. Timer/Counter Timing Diagram, no Prescaling

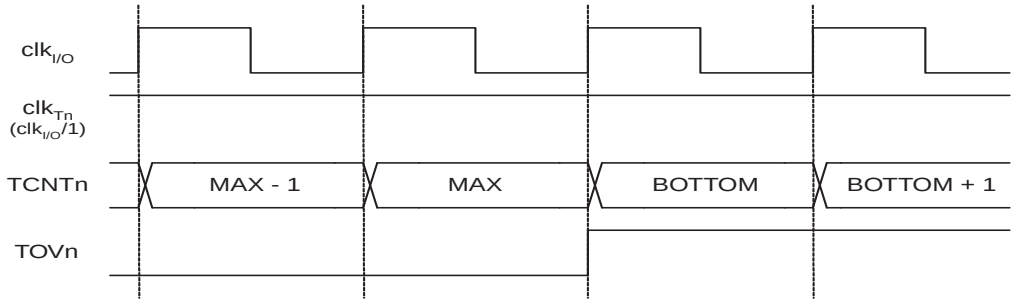


Figure 15-9 shows the same timing data, but with the prescaler enabled.

Figure 15-9. Timer/Counter Timing Diagram, with Prescaler ($f_{clk_I/O}/8$)

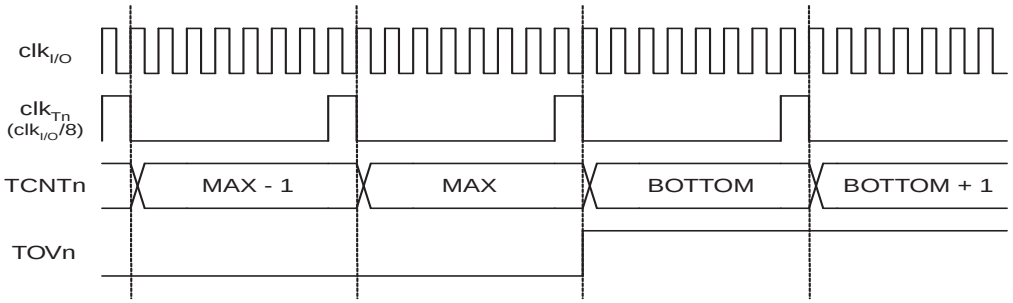


Figure 15-10 shows the setting of OCF2A in all modes except CTC mode.

Figure 15-10. Timer/Counter Timing Diagram, Setting of OCF2A, with Prescaler ($f_{clk_I/O}/8$)

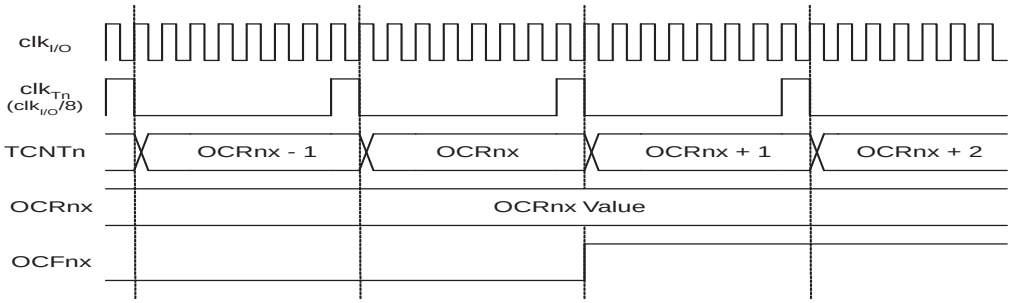
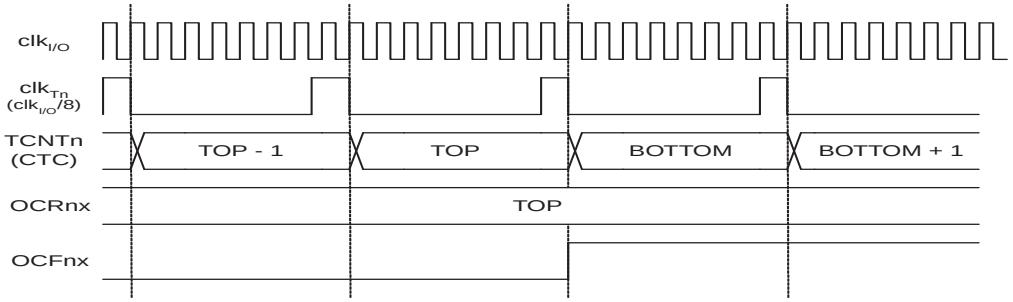


Figure 15-11 shows the setting of OCF2A and the clearing of TCNT2 in CTC mode.

Figure 15-11. Timer/Counter Timing Diagram, Clear Timer on Compare Match mode, with Prescaler ($f_{clk_I/O}/8$)



15.9 Asynchronous Operation of Timer/Counter2

When Timer/Counter2 operates asynchronously, some considerations must be taken.

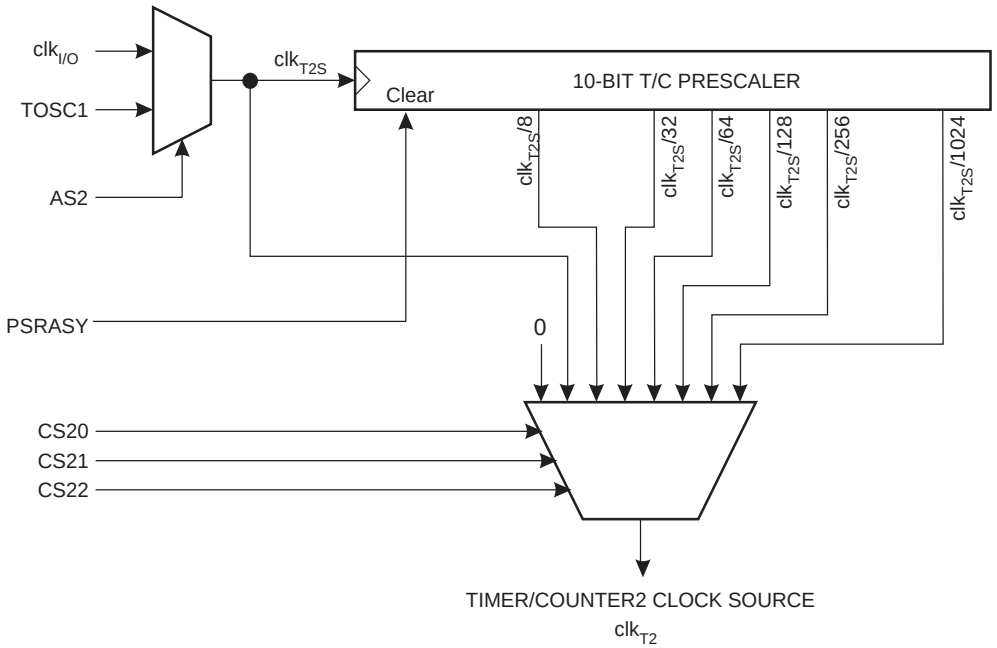
- Warning: When switching between asynchronous and synchronous clocking of Timer/Counter2, the Timer Registers TCNT2, OCR2x, and TCCR2x might be corrupted. A safe procedure for switching clock source is:
  a. Disable the Timer/Counter2 interrupts by clearing OCIE2x and TOIE2.
  b. Select clock source by setting AS2 as appropriate.
  c. Write new values to TCNT2, OCR2x, and TCCR2x.
  d. To switch to asynchronous operation: Wait for TCN2xUB, OCR2xUB, and TCR2xUB.
  e. Clear the Timer/Counter2 Interrupt Flags.
  f. Enable interrupts, if needed.
- The CPU main clock frequency must be more than four times the Oscillator frequency.
- When writing to one of the registers TCNT2, OCR2x, or TCCR2x, the value is transferred to a temporary register, and latched after two positive edges on TOSC1. The user should not write a new value before the contents of the temporary register have been transferred to its destination. Each of the five mentioned registers have their individual temporary register, which means that e.g. writing to TCNT2 does not disturb an OCR2x write in progress. To detect that a transfer to the destination register has taken place, the Asynchronous Status Register – ASSR has been implemented.
- When entering Power-save or ADC Noise Reduction mode after having written to TCNT2, OCR2x, or TCCR2x, the user must wait until the written register has been updated if Timer/Counter2 is used to wake up the device. Otherwise, the MCU will enter sleep mode before the changes are effective. This is particularly important if any of the Output Compare2 interrupt is used to wake up the device, since the Output Compare function is disabled during writing to OCR2x or TCNT2. If the write cycle is not finished, and the MCU enters sleep mode before the corresponding OCR2xUB bit returns to zero, the device will never receive a compare match interrupt, and the MCU will not wake up.
- If Timer/Counter2 is used to wake the device up from Power-save or ADC Noise Reduction mode, precautions must be taken if the user wants to re-enter one of these modes: If re-entering sleep mode within the TOSC1 cycle, the interrupt will immidiately occur and the device wake up again. The result is multiple interrupts and wake-ups within one TOSC1 cycle from the first interrupt. If the user is in doubt whether the time before re-entering Power-save or ADC Noise Reduction mode is sufficient, the following algorithm can be used to ensure that one TOSC1 cycle has elapsed:
  a. Write a value to TCCR2x, TCNT2, or OCR2x.
  b. Wait until the corresponding Update Busy Flag in ASSR returns to zero.
  c. Enter Power-save or ADC Noise Reduction mode.
- When the asynchronous operation is selected, the 32.768 kHz Oscillator for Timer/Counter2 is always running, except in Power-down and Standby modes. After a Power-up Reset or wake-up from Power-down or Standby mode, the user should be aware of the fact that this Oscillator might take as long as one second to stabilize. The user is advised to wait for at least one second before using Timer/Counter2 after power-up or wake-up from Power-down or Standby mode. The contents of all Timer/Counter2 Registers must be considered lost after a wake-up from Power-down or Standby mode due to unstable clock signal upon start-up, no matter whether the Oscillator is in use or a clock signal is applied to the TOSC1 pin.

- Description of wake up from Power-save or ADC Noise Reduction mode when the timer is clocked asynchronously: When the interrupt condition is met, the wake up process is started on the following cycle of the timer clock, that is, the timer is always advanced by at least one before the processor can read the counter value. After wake-up, the MCU is halted for four cycles, it executes the interrupt routine, and resumes execution from the instruction following SLEEP.
- Reading of the TCNT2 Register shortly after wake-up from Power-save may give an incorrect result. Since TCNT2 is clocked on the asynchronous TOSC clock, reading TCNT2 must be done through a register synchronized to the internal I/O clock domain. Synchronization takes place for every rising TOSC1 edge. When waking up from Power-save mode, and the I/O clock ($clk_{I/O}$) again becomes active, TCNT2 will read as the previous value (before entering sleep) until the next rising TOSC1 edge. The phase of the TOSC clock after waking up from Power-save mode is essentially unpredictable, as it depends on the wake-up time. The recommended procedure for reading TCNT2 is thus as follows:
    a. Write any value to either of the registers OCR2x or TCCR2x.
    b. Wait for the corresponding Update Busy Flag to be cleared.
    c. Read TCNT2.

During asynchronous operation, the synchronization of the Interrupt Flags for the asynchronous timer takes 3 processor cycles plus one timer cycle. The timer is therefore advanced by at least one before the processor can read the timer value causing the setting of the Interrupt Flag. The Output Compare pin is changed on the timer clock and is not synchronized to the processor clock.

15.10 Timer/Counter Prescaler

Figure 15-12. Prescaler for Timer/Counter2



The clock source for Timer/Counter2 is named clk_{T2S}. clk_{T2S} is by default connected to the main system I/O clock clk_{IO}. By setting the AS2 bit in ASSR, Timer/Counter2 is asynchronously clocked from the TOSC1 pin. This enables use of Timer/Counter2 as a Real Time Counter

(RTC). When AS2 is set, pins TOSC1 and TOSC2 are disconnected from Port C. A crystal can then be connected between the TOSC1 and TOSC2 pins to serve as an independent clock source for Timer/Counter2. The Oscillator is optimized for use with a 32.768 kHz crystal.

For Timer/Counter2, the possible prescaled selections are: $clk_{T2S}/8$, $clk_{T2S}/32$, $clk_{T2S}/64$, $clk_{T2S}/128$, $clk_{T2S}/256$, and $clk_{T2S}/1024$. Additionally, clk_{T2S} as well as 0 (stop) may be selected. Setting the PSRASY bit in GTCCR resets the prescaler. This allows the user to operate with a predictable prescaler.

15.11 Register Description

15.11.1 TCCR2A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
(0xB0)	COM2A1	COM2A0	COM2B1	COM2B0	–	–	WGM21	WGM20	TCCR2A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:6 – COM2A1:0: Compare Match Output A Mode**

These bits control the Output Compare pin (OC2A) behavior. If one or both of the COM2A1:0 bits are set, the OC2A output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC2A pin must be set in order to enable the output driver.

When OC2A is connected to the pin, the function of the COM2A1:0 bits depends on the WGM22:0 bit setting. Table 15-2 shows the COM2A1:0 bit functionality when the WGM22:0 bits are set to a normal or CTC mode (non-PWM).

Table 15-2. Compare Output Mode, non-PWM Mode

COM2A1	COM2A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC2A on Compare Match
1	0	Clear OC2A on Compare Match
1	1	Set OC2A on Compare Match

Table 15-3 shows the COM2A1:0 bit functionality when the WGM21:0 bits are set to fast PWM mode.

Table 15-3. Compare Output Mode, Fast PWM Mode(1)

COM2A1	COM2A0	Description
0	0	Normal port operation, OC2A disconnected.
0	1	WGM22 = 0: Normal Port Operation, OC0A Disconnected. WGM22 = 1: Toggle OC2A on Compare Match.
1	0	Clear OC2A on Compare Match, set OC2A at BOTTOM, (non-inverting mode).
1	1	Set OC2A on Compare Match, clear OC2A at BOTTOM, (inverting mode).

Note: 1. A special case occurs when OCR2A equals TOP and COM2A1 is set. In this case, the Compare Match is ignored, but the set or clear is done at BOTTOM. See ”Fast PWM Mode” on page 150 for more details.

Table 15-4 shows the COM2A1:0 bit functionality when the WGM22:0 bits are set to phase correct PWM mode.

Table 15-4. Compare Output Mode, Phase Correct PWM Mode(1)

COM2A1	COM2A0	Description
0	0	Normal port operation, OC2A disconnected.
0	1	WGM22 = 0: Normal Port Operation, OC2A Disconnected. WGM22 = 1: Toggle OC2A on Compare Match.
1	0	Clear OC2A on Compare Match when up-counting. Set OC2A on Compare Match when down-counting.
1	1	Set OC2A on Compare Match when up-counting. Clear OC2A on Compare Match when down-counting.

Note: 1. A special case occurs when OCR2A equals TOP and COM2A1 is set. In this case, the Compare Match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 152 for more details.

- **Bits 5:4 – COM2B1:0: Compare Match Output B Mode**

These bits control the Output Compare pin (OC2B) behavior. If one or both of the COM2B1:0 bits are set, the OC2B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC2B pin must be set in order to enable the output driver.

When OC2B is connected to the pin, the function of the COM2B1:0 bits depends on the WGM22:0 bit setting. Table 15-5 shows the COM2B1:0 bit functionality when the WGM22:0 bits are set to a normal or CTC mode (non-PWM).

Table 15-5. Compare Output Mode, non-PWM Mode

COM2B1	COM2B0	Description
0	0	Normal port operation, OC2B disconnected.
0	1	Toggle OC2B on Compare Match
1	0	Clear OC2B on Compare Match
1	1	Set OC2B on Compare Match

Table 15-6 shows the COM2B1:0 bit functionality when the WGM22:0 bits are set to fast PWM mode.

Table 15-6. Compare Output Mode, Fast PWM Mode(1)

COM2B1	COM2B0	Description
0	0	Normal port operation, OC2B disconnected.
0	1	Reserved
1	0	Clear OC2B on Compare Match, set OC2B at BOTTOM, (non-inverting mode).
1	1	Set OC2B on Compare Match, clear OC2B at BOTTOM, (inverting mode).

Note: 1. A special case occurs when OCR2B equals TOP and COM2B1 is set. In this case, the Compare Match is ignored, but the set or clear is done at BOTTOM. See "Phase Correct PWM Mode" on page 152 for more details.

Table 15-7 shows the COM2B1:0 bit functionality when the WGM22:0 bits are set to phase correct PWM mode.

Table 15-7. Compare Output Mode, Phase Correct PWM Mode(1)

COM2B1	COM2B0	Description
0	0	Normal port operation, OC2B disconnected.
0	1	Reserved
1	0	Clear OC2B on Compare Match when up-counting. Set OC2B on Compare Match when down-counting.
1	1	Set OC2B on Compare Match when up-counting. Clear OC2B on Compare Match when down-counting.

Note: 1. A special case occurs when OCR2B equals TOP and COM2B1 is set. In this case, the Compare Match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 152 for more details.

- **Bits 3, 2 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bits 1:0 – WGM21:0: Waveform Generation Mode**

Combined with the WGM22 bit found in the TCCR2B Register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used, see Table 15-8. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), Clear Timer on Compare Match (CTC) mode, and two types of Pulse Width Modulation (PWM) modes (see "Modes of Operation" on page 149).

Table 15-8. Waveform Generation Mode Bit Description

Mode	WGM2	WGM1	WGM0	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Notes: 1. MAX= 0xFF
2. BOTTOM= 0x00

15.11.2 TCCR2B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
(0xB1)	FOC2A	FOC2B	–	–	WGM22	CS22	CS21	CS20	TCCR2B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – FOC2A: Force Output Compare A**

The FOC2A bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR2B is written when operating in PWM mode. When writing a logical one to the FOC2A bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC2A output is changed according to its COM2A1:0 bits setting. Note that the FOC2A bit is implemented as a strobe. Therefore it is the value present in the COM2A1:0 bits that determines the effect of the forced compare.

A FOC2A strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR2A as TOP.

The FOC2A bit is always read as zero.

- **Bit 6 – FOC2B: Force Output Compare B**

The FOC2B bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR2B is written when operating in PWM mode. When writing a logical one to the FOC2B bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC2B output is changed according to its COM2B1:0 bits setting. Note that the FOC2B bit is implemented as a strobe. Therefore it is the value present in the COM2B1:0 bits that determines the effect of the forced compare.

A FOC2B strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR2B as TOP.

The FOC2B bit is always read as zero.

- **Bits 5:4 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bit 3 – WGM22: Waveform Generation Mode**

See the description in the ”TCCR2A – Timer/Counter Control Register A” on page 158.

- **Bit 2:0 – CS22:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter, see Table 15-9 on page 162.

Table 15-9. Clock Select Bit Description

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk_{T2S}/(No prescaling)
0	1	0	clk_{T2S}/8 (From prescaler)
0	1	1	clk_{T2S}/32 (From prescaler)
1	0	0	clk_{T2S}/64 (From prescaler)
1	0	1	clk_{T2S}/128 (From prescaler)
1	1	0	clk_{T2S}/256 (From prescaler)
1	1	1	clk_{T2S}/1024 (From prescaler)

If external pin modes are used for the Timer/Counter0, transitions on the T0 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

15.11.3 TCNT2 – Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
(0xB2)	TCNT2[7:0]								TCNT2
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Timer/Counter Register gives direct access, both for read and write operations, to the Timer/Counter unit 8-bit counter. Writing to the TCNT2 Register blocks (removes) the Compare Match on the following timer clock. Modifying the counter (TCNT2) while the counter is running, introduces a risk of missing a Compare Match between TCNT2 and the OCR2x Registers.

15.11.4 OCR2A – Output Compare Register A

Bit	7	6	5	4	3	2	1	0	
(0xB3)	OCR2A[7:0]								OCR2A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register A contains an 8-bit value that is continuously compared with the counter value (TCNT2). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC2A pin.

15.11.5 OCR2B – Output Compare Register B

Bit	7	6	5	4	3	2	1	0	
(0xB4)	OCR2B[7:0]								OCR2B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register B contains an 8-bit value that is continuously compared with the counter value (TCNT2). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC2B pin.

15.11.6 TIMSK2 – Timer/Counter2 Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x70)	–	–	–	–	–	OCIE2B	OCIE2A	TOIE2	TIMSK2
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 2 – OCIE2B: Timer/Counter2 Output Compare Match B Interrupt Enable**

When the OCIE2B bit is written to one and the I-bit in the Status Register is set (one), the Timer/Counter2 Compare Match B interrupt is enabled. The corresponding interrupt is executed if a compare match in Timer/Counter2 occurs, i.e., when the OCF2B bit is set in the Timer/Counter 2 Interrupt Flag Register – TIFR2.

- **Bit 1 – OCIE2A: Timer/Counter2 Output Compare Match A Interrupt Enable**

When the OCIE2A bit is written to one and the I-bit in the Status Register is set (one), the Timer/Counter2 Compare Match A interrupt is enabled. The corresponding interrupt is executed if a compare match in Timer/Counter2 occurs, i.e., when the OCF2A bit is set in the Timer/Counter 2 Interrupt Flag Register – TIFR2.

- **Bit 0 – TOIE2: Timer/Counter2 Overflow Interrupt Enable**

When the TOIE2 bit is written to one and the I-bit in the Status Register is set (one), the Timer/Counter2 Overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter2 occurs, i.e., when the TOV2 bit is set in the Timer/Counter2 Interrupt Flag Register – TIFR2.

15.11.7 TIFR2 – Timer/Counter2 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x17 (0x37)	–	–	–	–	–	OCF2B	OCF2A	TOV2	TIFR2
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 2 – OCF2B: Output Compare Flag 2 B**

The OCF2B bit is set (one) when a compare match occurs between the Timer/Counter2 and the data in OCR2B – Output Compare Register2. OCF2B is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF2B is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE2B (Timer/Counter2 Compare match Interrupt Enable), and OCF2B are set (one), the Timer/Counter2 Compare match Interrupt is executed.

- **Bit 1 – OCF2A: Output Compare Flag 2 A**

The OCF2A bit is set (one) when a compare match occurs between the Timer/Counter2 and the data in OCR2A – Output Compare Register2. OCF2A is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF2A is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE2A (Timer/Counter2 Compare match Interrupt Enable), and OCF2A are set (one), the Timer/Counter2 Compare match Interrupt is executed.

- **Bit 0 – TOV2: Timer/Counter2 Overflow Flag**

The TOV2 bit is set (one) when an overflow occurs in Timer/Counter2. TOV2 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV2 is cleared by writing a logic one to the flag. When the SREG I-bit, TOIE2A (Timer/Counter2 Overflow Interrupt Enable), and TOV2 are set (one), the Timer/Counter2 Overflow interrupt is executed. In PWM mode, this bit is set when Timer/Counter2 changes counting direction at 0x00.

15.11.8 ASSR – Asynchronous Status Register

Bit	7	6	5	4	3	2	1	0	
(0xB6)	–	EXCLK	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB	ASSR
Read/Write	R	R/W	R/W	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – RES: Reserved bit**

This bit is reserved and will always read as zero.

- **Bit 6 – EXCLK: Enable External Clock Input**

When EXCLK is written to one, and asynchronous clock is selected, the external clock input buffer is enabled and an external clock can be input on Timer Oscillator 1 (TOSC1) pin instead of a 32 kHz crystal. Writing to EXCLK should be done before asynchronous operation is selected. Note that the crystal Oscillator will only run when this bit is zero.

- **Bit 5 – AS2: Asynchronous Timer/Counter2**

When AS2 is written to zero, Timer/Counter2 is clocked from the I/O clock, $clk_{I/O}$. When AS2 is written to one, Timer/Counter2 is clocked from a crystal Oscillator connected to the Timer Oscillator 1 (TOSC1) pin. When the value of AS2 is changed, the contents of TCNT2, OCR2A, OCR2B, TCCR2A and TCCR2B might be corrupted.

- **Bit 4 – TCN2UB: Timer/Counter2 Update Busy**

When Timer/Counter2 operates asynchronously and TCNT2 is written, this bit becomes set. When TCNT2 has been updated from the temporary storage register, this bit is cleared by hardware. A logical zero in this bit indicates that TCNT2 is ready to be updated with a new value.

- **Bit 3 – OCR2AUB: Output Compare Register2 Update Busy**

When Timer/Counter2 operates asynchronously and OCR2A is written, this bit becomes set. When OCR2A has been updated from the temporary storage register, this bit is cleared by hardware. A logical zero in this bit indicates that OCR2A is ready to be updated with a new value.

- **Bit 2 – OCR2BUB: Output Compare Register2 Update Busy**

When Timer/Counter2 operates asynchronously and OCR2B is written, this bit becomes set. When OCR2B has been updated from the temporary storage register, this bit is cleared by hardware. A logical zero in this bit indicates that OCR2B is ready to be updated with a new value.

- **Bit 1 – TCR2AUB: Timer/Counter Control Register2 Update Busy**

When Timer/Counter2 operates asynchronously and TCCR2A is written, this bit becomes set. When TCCR2A has been updated from the temporary storage register, this bit is cleared by hardware. A logical zero in this bit indicates that TCCR2A is ready to be updated with a new value.

- **Bit 0 – TCR2BUB: Timer/Counter Control Register2 Update Busy**

When Timer/Counter2 operates asynchronously and TCCR2B is written, this bit becomes set. When TCCR2B has been updated from the temporary storage register, this bit is cleared by hardware. A logical zero in this bit indicates that TCCR2B is ready to be updated with a new value.

If a write is performed to any of the five Timer/Counter2 Registers while its update busy flag is set, the updated value might get corrupted and cause an unintentional interrupt to occur.

The mechanisms for reading TCNT2, OCR2A, OCR2B, TCCR2A and TCCR2B are different. When reading TCNT2, the actual timer value is read. When reading OCR2A, OCR2B, TCCR2A and TCCR2B the value in the temporary storage register is read.

15.11.9 GTCCR – General Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
0x23 (0x43)	TSM	–	–	–	–	–	PSRASY	PSRSYNC	GTCCR
Read/Write	R/W	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 1 – PSRASY: Prescaler Reset Timer/Counter2**

When this bit is one, the Timer/Counter2 prescaler will be reset. This bit is normally cleared immediately by hardware. If the bit is written when Timer/Counter2 is operating in asynchronous mode, the bit will remain one until the prescaler has been reset. The bit will not be cleared by hardware if the TSM bit is set. Refer to the description of the ”Bit 7 – TSM: Timer/Counter Synchronization Mode” on page 143 for a description of the Timer/Counter Synchronization mode.

16. SPI – Serial Peripheral Interface

16.1 Features

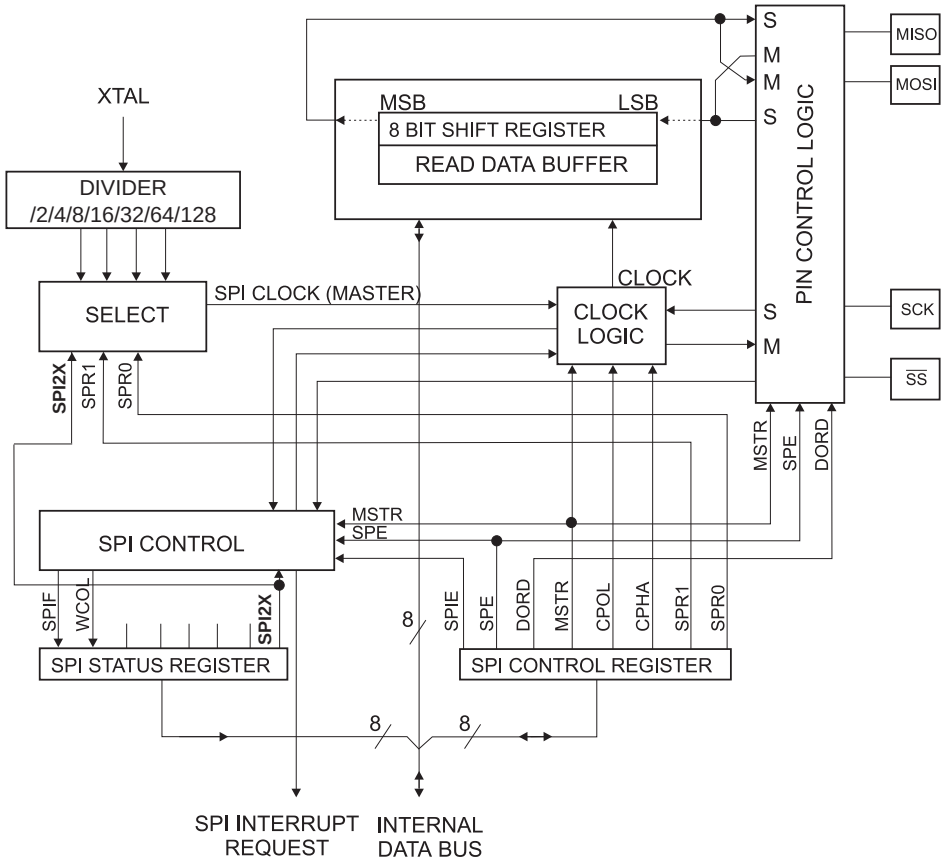
- **Full-duplex, Three-wire Synchronous Data Transfer**
- **Master or Slave Operation**
- **LSB First or MSB First Data Transfer**
- **Seven Programmable Bit Rates**
- **End of Transmission Interrupt Flag**
- **Write Collision Flag Protection**
- **Wake-up from Idle Mode**
- **Double Speed (CK/2) Master SPI Mode**

16.2 Overview

The Serial Peripheral Interface (SPI) allows high-speed synchronous data transfer between the ATmega48P/88P/168P/328P and peripheral devices or between several AVR devices.

The USART can also be used in Master SPI mode, see “USART in SPI Mode” on page 204. The PRSPI bit in ”Minimizing Power Consumption” on page 42 must be written to zero to enable SPI module.

Figure 16-1. SPI Block Diagram[1]



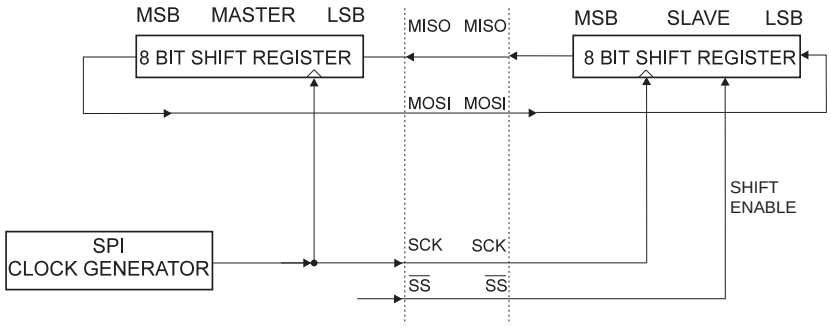
Note: 1. Refer to Figure 1-1 on page 2, and Table 11-3 on page 82 for SPI pin placement.

The interconnection between Master and Slave CPUs with SPI is shown in Figure 16-2 on page 167. The system consists of two shift Registers, and a Master clock generator. The SPI Master initiates the communication cycle when pulling low the Slave Select $\overline{SS}$ pin of the desired Slave. Master and Slave prepare the data to be sent in their respective shift Registers, and the Master generates the required clock pulses on the SCK line to interchange data. Data is always shifted from Master to Slave on the Master Out – Slave In, MOSI, line, and from Slave to Master on the Master In – Slave Out, MISO, line. After each data packet, the Master will synchronize the Slave by pulling high the Slave Select, $\overline{SS}$, line.

When configured as a Master, the SPI interface has no automatic control of the $\overline{SS}$ line. This must be handled by user software before communication can start. When this is done, writing a byte to the SPI Data Register starts the SPI clock generator, and the hardware shifts the eight bits into the Slave. After shifting one byte, the SPI clock generator stops, setting the end of Transmission Flag (SPIF). If the SPI Interrupt Enable bit (SPIE) in the SPCR Register is set, an interrupt is requested. The Master may continue to shift the next byte by writing it into SPDR, or signal the end of packet by pulling high the Slave Select, $\overline{SS}$ line. The last incoming byte will be kept in the Buffer Register for later use.

When configured as a Slave, the SPI interface will remain sleeping with MISO tri-stated as long as the $\overline{SS}$ pin is driven high. In this state, software may update the contents of the SPI Data Register, SPDR, but the data will not be shifted out by incoming clock pulses on the SCK pin until the $\overline{SS}$ pin is driven low. As one byte has been completely shifted, the end of Transmission Flag, SPIF is set. If the SPI Interrupt Enable bit, SPIE, in the SPCR Register is set, an interrupt is requested. The Slave may continue to place new data to be sent into SPDR before reading the incoming data. The last incoming byte will be kept in the Buffer Register for later use.

Figure 16-2. SPI Master-slave Interconnection



The system is single buffered in the transmit direction and double buffered in the receive direction. This means that bytes to be transmitted cannot be written to the SPI Data Register before the entire shift cycle is completed. When receiving data, however, a received character must be read from the SPI Data Register before the next character has been completely shifted in. Otherwise, the first byte is lost.

In SPI Slave mode, the control logic will sample the incoming signal of the SCK pin. To ensure correct sampling of the clock signal, the minimum low and high periods should be:

Low periods: Longer than 2 CPU clock cycles.

High periods: Longer than 2 CPU clock cycles.

When the SPI is enabled, the data direction of the MOSI, MISO, SCK, and $\overline{SS}$ pins is overridden according to Table 16-1 on page 168. For more details on automatic port overrides, refer to "Alternate Port Functions" on page 80.

Table 16-1. SPI Pin Overrides(Note:)

Pin	Direction, Master SPI	Direction, Slave SPI
MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
$\overline{SS}$	User Defined	Input

Note: See "Alternate Functions of Port B" on page 82 for a detailed description of how to define the direction of the user defined SPI pins.

The following code examples show how to initialize the SPI as a Master and how to perform a simple transmission. DDR_SPI in the examples must be replaced by the actual Data Direction Register controlling the SPI pins. DD_MOSI, DD_MISO and DD_SCK must be replaced by the actual data direction bits for these pins. E.g. if MOSI is placed on pin PB5, replace DD_MOSI with DDB5 and DDR_SPI with DDRB.

Assembly Code Example[1]

```
SPI_MasterInit:
  ; Set MOSI and SCK output, all others input
  ldi   r17,(1<<DD_MOSI)|(1<<DD_SCK)
  out   DDR_SPI,r17
  ; Enable SPI, Master, set clock rate fck/16
  ldi   r17,(1<<SPE)|(1<<MSTR)|(1<<SPR0)
  out   SPCR,r17
  ret

SPI_MasterTransmit:
  ; Start transmission of data (r16)
  out   SPDR,r16
Wait_Transmit:
  ; Wait for transmission complete
  in    r16, SPSR
  sbrsr16, SPIF
  rjmp Wait_Transmit
  ret
```

C Code Example[1]

```
void SPI_MasterInit(void)
{
  /* Set MOSI and SCK output, all others input */
  DDR_SPI = (1<<DD_MOSI)|(1<<DD_SCK);
  /* Enable SPI, Master, set clock rate fck/16 */
  SPCR = (1<<SPE)|(1<<MSTR)|(1<<SPR0);
}

void SPI_MasterTransmit(char cData)
{
  /* Start transmission */
  SPDR = cData;
  /* Wait for transmission complete */
  while(!(SPSR & (1<<SPIF)))
    ;
}
```

Note: 1. See "Code Examples" on page 7.

The following code examples show how to initialize the SPI as a Slave and how to perform a simple reception.

Assembly Code Example[1]

```
SPI_SlaveInit:
  ; Set MISO output, all others input
  ldi  r17,(1<<DD_MISO)
  out  DDR_SPI,r17
  ; Enable SPI
  ldi  r17,(1<<SPE)
  out  SPCR,r17
  ret

SPI_SlaveReceive:
  ; Wait for reception complete
  sbis SPSR,SPIF
  rjmp SPI_SlaveReceive
  ; Read received data and return
  in   r16,SPDR
  ret
```

C Code Example[1]

```
void SPI_SlaveInit(void)
{
  /* Set MISO output, all others input */
  DDR_SPI = (1<<DD_MISO);
  /* Enable SPI */
  SPCR = (1<<SPE);
}

char SPI_SlaveReceive(void)
{
  /* Wait for reception complete */
  while(!(SPSR & (1<<SPIF)))
    ;
  /* Return Data Register */
  return SPDR;
}
```

Note: 1. See "Code Examples" on page 7.

16.3 $\overline{SS}$ Pin Functionality

16.3.1 Slave Mode

When the SPI is configured as a Slave, the Slave Select ($\overline{SS}$) pin is always input. When $\overline{SS}$ is held low, the SPI is activated, and MISO becomes an output if configured so by the user. All other pins are inputs. When $\overline{SS}$ is driven high, all pins are inputs, and the SPI is passive, which means that it will not receive incoming data. Note that the SPI logic will be reset once the $\overline{SS}$ pin is driven high.

The $\overline{SS}$ pin is useful for packet/byte synchronization to keep the slave bit counter synchronous with the master clock generator. When the $\overline{SS}$ pin is driven high, the SPI slave will immediately reset the send and receive logic, and drop any partially received data in the Shift Register.

16.3.2 Master Mode

When the SPI is configured as a Master (MSTR in SPCR is set), the user can determine the direction of the $\overline{SS}$ pin.

If $\overline{SS}$ is configured as an output, the pin is a general output pin which does not affect the SPI system. Typically, the pin will be driving the $\overline{SS}$ pin of the SPI Slave.

If $\overline{SS}$ is configured as an input, it must be held high to ensure Master SPI operation. If the $\overline{SS}$ pin is driven low by peripheral circuitry when the SPI is configured as a Master with the $\overline{SS}$ pin defined as an input, the SPI system interprets this as another master selecting the SPI as a slave and starting to send data to it. To avoid bus contention, the SPI system takes the following actions:

1. The MSTR bit in SPCR is cleared and the SPI system becomes a Slave. As a result of the SPI becoming a Slave, the MOSI and SCK pins become inputs.
2. The SPIF Flag in SPSR is set, and if the SPI interrupt is enabled, and the I-bit in SREG is set, the interrupt routine will be executed.

Thus, when interrupt-driven SPI transmission is used in Master mode, and there exists a possibility that $\overline{SS}$ is driven low, the interrupt should always check that the MSTR bit is still set. If the MSTR bit has been cleared by a slave select, it must be set by the user to re-enable SPI Master mode.

16.4 Data Modes

There are four combinations of SCK phase and polarity with respect to serial data, which are determined by control bits CPHA and CPOL. The SPI data transfer formats are shown in Figure 16-3 and Figure 16-4 on page 172. Data bits are shifted out and latched in on opposite edges of the SCK signal, ensuring sufficient time for data signals to stabilize. This is clearly seen by summarizing Table 16-3 on page 173 and Table 16-4 on page 173, as done in Table 16-2.

Table 16-2. SPI Modes

SPI Mode	Conditions	Leading Edge	Trailing eDge
0	CPOL=0, CPHA=0	Sample (Rising)	Setup (Falling)
1	CPOL=0, CPHA=1	Setup (Rising)	Sample (Falling)
2	CPOL=1, CPHA=0	Sample (Falling)	Setup (Rising)
3	CPOL=1, CPHA=1	Setup (Falling)	Sample (Rising)

Figure 16-3. SPI Transfer Format with CPHA = 0

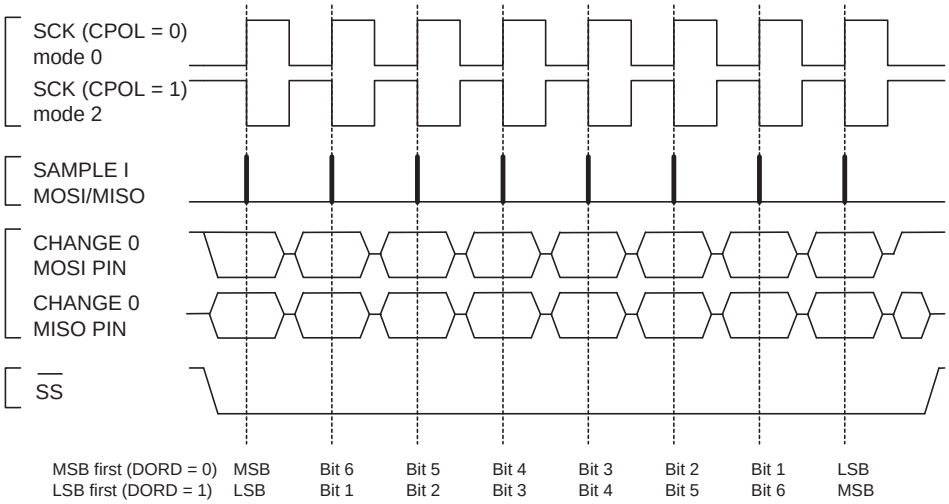
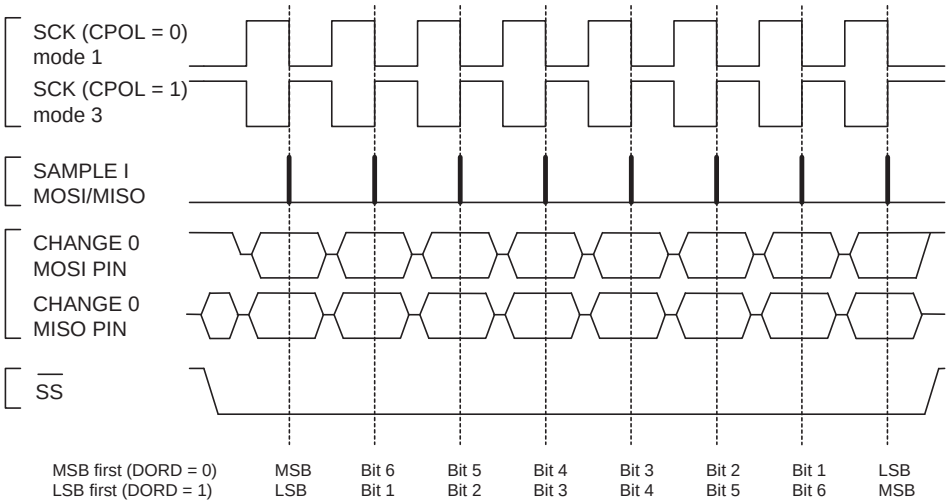


Figure 16-4. SPI Transfer Format with CPHA = 1



16.5 Register Description

16.5.1 SPCR – SPI Control Register

Bit	7	6	5	4	3	2	1	0	
0x2C (0x4C)	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – SPIE: SPI Interrupt Enable**

This bit causes the SPI interrupt to be executed if SPIF bit in the SPSR Register is set and the if the Global Interrupt Enable bit in SREG is set.

- **Bit 6 – SPE: SPI Enable**

When the SPE bit is written to one, the SPI is enabled. This bit must be set to enable any SPI operations.

- **Bit 5 – DORD: Data Order**

When the DORD bit is written to one, the LSB of the data word is transmitted first.

When the DORD bit is written to zero, the MSB of the data word is transmitted first.

- **Bit 4 – MSTR: Master/Slave Select**

This bit selects Master SPI mode when written to one, and Slave SPI mode when written logic zero. If $\overline{SS}$ is configured as an input and is driven low while MSTR is set, MSTR will be cleared, and SPIF in SPSR will become set. The user will then have to set MSTR to re-enable SPI Master mode.

- **Bit 3 – CPOL: Clock Polarity**

When this bit is written to one, SCK is high when idle. When CPOL is written to zero, SCK is low when idle. Refer to Figure 16-3 and Figure 16-4 for an example. The CPOL functionality is summarized below:

Table 16-3. CPOL Functionality

CPOL	Leading Edge	Trailing Edge
0	Rising	Falling
1	Falling	Rising

- **Bit 2 – CPHA: Clock Phase**

The settings of the Clock Phase bit (CPHA) determine if data is sampled on the leading (first) or trailing (last) edge of SCK. Refer to Figure 16-3 and Figure 16-4 for an example. The CPOL functionality is summarized below:

Table 16-4. CPHA Functionality

CPHA	Leading Edge	Trailing Edge
0	Sample	Setup
1	Setup	Sample

- **Bits 1, 0 – SPR1, SPR0: SPI Clock Rate Select 1 and 0**

These two bits control the SCK rate of the device configured as a Master. SPR1 and SPR0 have no effect on the Slave. The relationship between SCK and the Oscillator Clock frequency f_{osc} is shown in the following table:

Table 16-5. Relationship Between SCK and the Oscillator Frequency

SPI2X	SPR1	SPR0	SCK Frequency
0	0	0	$f_{osc}/4$
0	0	1	$f_{osc}/16$
0	1	0	$f_{osc}/64$
0	1	1	$f_{osc}/128$
1	0	0	$f_{osc}/2$
1	0	1	$f_{osc}/8$
1	1	0	$f_{osc}/32$
1	1	1	$f_{osc}/64$

16.5.2 SPSR – SPI Status Register

Bit	7	6	5	4	3	2	1	0	
0x2D (0x4D)	SPIF	WCOL	–	–	–	–	–	SPI2X	SPSR
Read/Write	R	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – SPIF: SPI Interrupt Flag**

When a serial transfer is complete, the SPIF Flag is set. An interrupt is generated if SPIE in SPCR is set and global interrupts are enabled. If $\overline{SS}$ is an input and is driven low when the SPI is in Master mode, this will also set the SPIF Flag. SPIF is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, the SPIF bit is cleared by first reading the SPI Status Register with SPIF set, then accessing the SPI Data Register (SPDR).

- **Bit 6 – WCOL: Write COLlision Flag**

The WCOL bit is set if the SPI Data Register (SPDR) is written during a data transfer. The WCOL bit (and the SPIF bit) are cleared by first reading the SPI Status Register with WCOL set, and then accessing the SPI Data Register.

- **Bit 5..1 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bit 0 – SPI2X: Double SPI Speed Bit**

When this bit is written logic one the SPI speed (SCK Frequency) will be doubled when the SPI is in Master mode (see Table 16-5). This means that the minimum SCK period will be two CPU clock periods. When the SPI is configured as Slave, the SPI is only guaranteed to work at $f_{osc}/4$ or lower.

The SPI interface on the ATmega48P/88P/168P/328P is also used for program memory and EEPROM downloading or uploading. See page 308 for serial programming and verification.

16.5.3 SPDR – SPI Data Register

Bit	7	6	5	4	3	2	1	0	
0x2E (0x4E)	**MSB**							**LSB**	**SPDR**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	X	X	X	X	X	X	X	X	Undefined

The SPI Data Register is a read/write register used for data transfer between the Register File and the SPI Shift Register. Writing to the register initiates data transmission. Reading the register causes the Shift Register Receive buffer to be read.

17. USART0

17.1 Features

- **Full Duplex Operation (Independent Serial Receive and Transmit Registers)**
- **Asynchronous or Synchronous Operation**
- **Master or Slave Clocked Synchronous Operation**
- **High Resolution Baud Rate Generator**
- **Supports Serial Frames with 5, 6, 7, 8, or 9 Data Bits and 1 or 2 Stop Bits**
- **Odd or Even Parity Generation and Parity Check Supported by Hardware**
- **Data OverRun Detection**
- **Framing Error Detection**
- **Noise Filtering Includes False Start Bit Detection and Digital Low Pass Filter**
- **Three Separate Interrupts on TX Complete, TX Data Register Empty and RX Complete**
- **Multi-processor Communication Mode**
- **Double Speed Asynchronous Communication Mode**

17.2 Overview

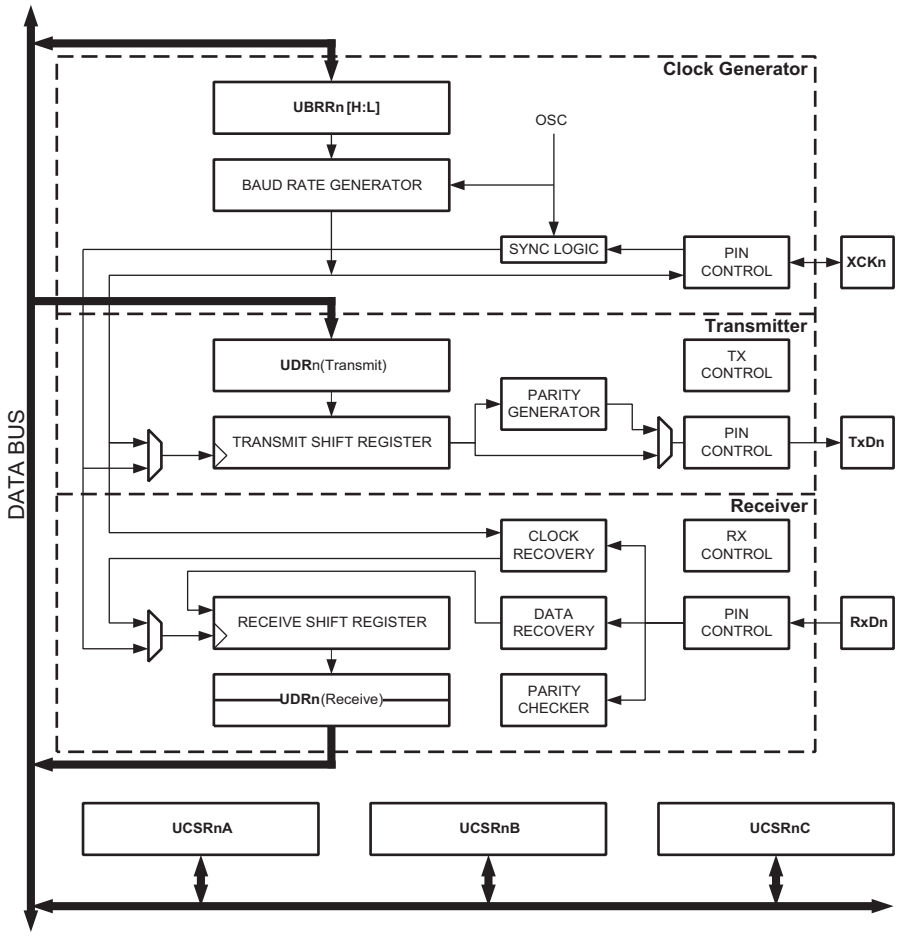
The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a highly flexible serial communication device.

The USART0 can also be used in Master SPI mode, see “USART in SPI Mode” on page 204. The Power Reduction USART bit, PRUSART0, in ”Minimizing Power Consumption” on page 42 must be disabled by writing a logical zero to it.

A simplified block diagram of the USART Transmitter is shown in Figure 17-1 on page 177. CPU accessible I/O Registers and I/O pins are shown in bold.

The dashed boxes in the block diagram separate the three main parts of the USART (listed from the top): Clock Generator, Transmitter and Receiver. Control Registers are shared by all units. The Clock Generation logic consists of synchronization logic for external clock input used by synchronous slave operation, and the baud rate generator. The XCKn (Transfer Clock) pin is only used by synchronous transfer mode. The Transmitter consists of a single write buffer, a serial Shift Register, Parity Generator and Control logic for handling different serial frame formats. The write buffer allows a continuous transfer of data without any delay between frames. The Receiver is the most complex part of the USART module due to its clock and data recovery units. The recovery units are used for asynchronous data reception. In addition to the recovery units, the Receiver includes a Parity Checker, Control logic, a Shift Register and a two level receive buffer (UDRn). The Receiver supports the same frame formats as the Transmitter, and can detect Frame Error, Data OverRun and Parity Errors.

Figure 17-1. USART Block Diagram[1]



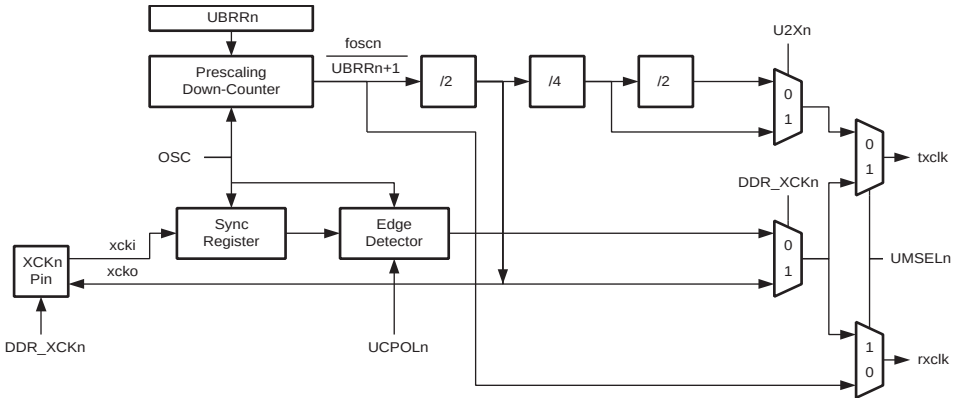
Note: 1. Refer to Figure 1-1 on page 2 and Table 11-9 on page 88 for USART0 pin placement.

17.3 Clock Generation

The Clock Generation logic generates the base clock for the Transmitter and Receiver. The USART supports four modes of clock operation: Normal asynchronous, Double Speed asynchronous, Master synchronous and Slave synchronous mode. The UMSELn bit in USART Control and Status Register C (UCSRnC) selects between asynchronous and synchronous operation. Double Speed (asynchronous mode only) is controlled by the U2Xn found in the UCSRnA Register. When using synchronous mode (UMSELn = 1), the Data Direction Register for the XCKn pin (DDR_XCKn) controls whether the clock source is internal (Master mode) or external (Slave mode). The XCKn pin is only active when using synchronous mode.

Figure 17-2 shows a block diagram of the clock generation logic.

Figure 17-2. Clock Generation Logic, Block Diagram



Signal description:

txclk Transmitter clock (Internal Signal).

rxclk Receiver base clock (Internal Signal).

xcki Input from XCK pin (internal Signal). Used for synchronous slave operation.

xcko Clock output to XCK pin (Internal Signal). Used for synchronous master operation.

fosc XTAL pin frequency (System Clock).

17.3.1 Internal Clock Generation – The Baud Rate Generator

Internal clock generation is used for the asynchronous and the synchronous master modes of operation. The description in this section refers to Figure 17-2.

The USART Baud Rate Register (UBRRn) and the down-counter connected to it function as a programmable prescaler or baud rate generator. The down-counter, running at system clock (f_{osc}), is loaded with the UBRRn value each time the counter has counted down to zero or when the UBRRnL Register is written. A clock is generated each time the counter reaches zero. This clock is the baud rate generator clock output (= f_{osc}/(UBRRn+1)). The Transmitter divides the baud rate generator clock output by 2, 8 or 16 depending on mode. The baud rate generator output is used directly by the Receiver's clock and data recovery units. However, the recovery units use a state machine that uses 2, 8 or 16 states depending on mode set by the state of the UMSELn, U2Xn and DDR_XCKn bits.

Table 17-1 contains equations for calculating the baud rate (in bits per second) and for calculating the UBRRn value for each mode of operation using an internally generated clock source.

Table 17-1. Equations for Calculating Baud Rate Register Setting

Operating Mode	Equation for Calculating Baud Rate[1]	Equation for Calculating UBRRn Value
Asynchronous Normal mode (U2Xn = 0)	$BAUD = \frac{f_{OSC}}{16(\boldsymbol{UBRR}n + 1)}$	$\boldsymbol{UBRR}n = \frac{f_{OSC}}{16BAUD} - 1$
Asynchronous Double Speed mode (U2Xn = 1)	$BAUD = \frac{f_{OSC}}{8(\boldsymbol{UBRR}n + 1)}$	$\boldsymbol{UBRR}n = \frac{f_{OSC}}{8BAUD} - 1$
Synchronous Master mode	$BAUD = \frac{f_{OSC}}{2(\boldsymbol{UBRR}n + 1)}$	$\boldsymbol{UBRR}n = \frac{f_{OSC}}{2BAUD} - 1$

Note: 1. The baud rate is defined to be the transfer rate in bit per second (bps)

BAUD Baud rate (in bits per second, bps)

f_{osc} System Oscillator clock frequency

UBRRn Contents of the UBRRnH and UBRRnL Registers, (0-4095)

Some examples of UBRRn values for some system clock frequencies are found in Table 17-9 (see page 200).

17.3.2 Double Speed Operation (U2Xn)

The transfer rate can be doubled by setting the U2Xn bit in UCSRnA. Setting this bit only has effect for the asynchronous operation. Set this bit to zero when using synchronous operation.

Setting this bit will reduce the divisor of the baud rate divider from 16 to 8, effectively doubling the transfer rate for asynchronous communication. Note however that the Receiver will in this case only use half the number of samples (reduced from 16 to 8) for data sampling and clock recovery, and therefore a more accurate baud rate setting and system clock are required when this mode is used. For the Transmitter, there are no downsides.

17.3.3 External Clock

External clocking is used by the synchronous slave modes of operation. The description in this section refers to Figure 17-2 for details.

External clock input from the XCKn pin is sampled by a synchronization register to minimize the chance of meta-stability. The output from the synchronization register must then pass through an edge detector before it can be used by the Transmitter and Receiver. This process introduces a two CPU clock period delay and therefore the maximum external XCKn clock frequency is limited by the following equation:

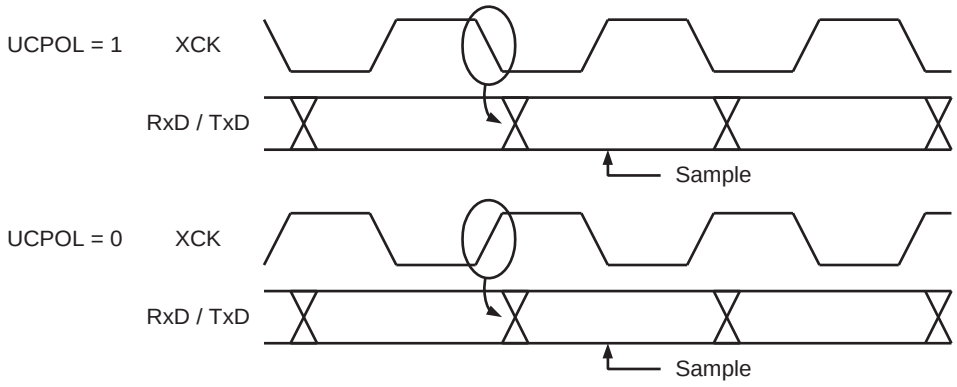
$$f_{XCK} < \frac{f_{OSC}}{4}$$

Note that f_{osc} depends on the stability of the system clock source. It is therefore recommended to add some margin to avoid possible loss of data due to frequency variations.

17.3.4 Synchronous Clock Operation

When synchronous mode is used (UMSELn = 1), the XCKn pin will be used as either clock input (Slave) or clock output (Master). The dependency between the clock edges and data sampling or data change is the same. The basic principle is that data input (on RxDn) is sampled at the opposite XCKn clock edge of the edge the data output (TxDn) is changed.

Figure 17-3. Synchronous Mode XCKn Timing.



The UCPOLn bit UCRSC selects which XCKn clock edge is used for data sampling and which is used for data change. As Figure 17-3 shows, when UCPOLn is zero the data will be changed at rising XCKn edge and sampled at falling XCKn edge. If UCPOLn is set, the data will be changed at falling XCKn edge and sampled at rising XCKn edge.

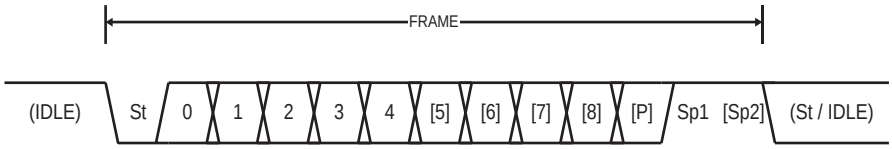
17.4 Frame Formats

A serial frame is defined to be one character of data bits with synchronization bits (start and stop bits), and optionally a parity bit for error checking. The USART accepts all 30 combinations of the following as valid frame formats:

- 1 start bit
- 5, 6, 7, 8, or 9 data bits
- no, even or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit followed by the least significant data bit. Then the next data bits, up to a total of nine, are succeeding, ending with the most significant bit. If enabled, the parity bit is inserted after the data bits, before the stop bits. When a complete frame is transmitted, it can be directly followed by a new frame, or the communication line can be set to an idle (high) state. Figure 17-4 illustrates the possible combinations of the frame formats. Bits inside brackets are optional.

Figure 17-4. Frame Formats



St Start bit, always low.

(n) Data bits (0 to 8).

P Parity bit. Can be odd or even.

Sp Stop bit, always high.

IDLE No transfers on the communication line (RxDn or TxDn). An IDLE line must be high.

The frame format used by the USART is set by the UCSZn2:0, UPMn1:0 and USBSn bits in UCSRnB and UCSRnC. The Receiver and Transmitter use the same setting. Note that changing the setting of any of these bits will corrupt all ongoing communication for both the Receiver and Transmitter.

The USART Character SiZe (UCSZn2:0) bits select the number of data bits in the frame. The USART Parity mode (UPMn1:0) bits enable and set the type of parity bit. The selection between one or two stop bits is done by the USART Stop Bit Select (USBSn) bit. The Receiver ignores the second stop bit. An FE (Frame Error) will therefore only be detected in the cases where the first stop bit is zero.

17.4.1 Parity Bit Calculation

The parity bit is calculated by doing an exclusive-or of all the data bits. If odd parity is used, the result of the exclusive or is inverted. The relation between the parity bit and data bits is as follows:

$$P_{even} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0$$
$$P_{odd} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 1$$

P_{even} Parity bit using even parity

P^{odd} Parity bit using odd parity

d_n Data bit n of the character

If used, the parity bit is located between the last data bit and first stop bit of a serial frame.

17.5 USART Initialization

The USART has to be initialized before any communication can take place. The initialization process normally consists of setting the baud rate, setting frame format and enabling the Transmitter or the Receiver depending on the usage. For interrupt driven USART operation, the Global Interrupt Flag should be cleared (and interrupts globally disabled) when doing the initialization.

Before doing a re-initialization with changed baud rate or frame format, be sure that there are no ongoing transmissions during the period the registers are changed. The TXCn Flag can be used to check that the Transmitter has completed all transfers, and the RXC Flag can be used to check that there are no unread data in the receive buffer. Note that the TXCn Flag must be cleared before each transmission (before UDRn is written) if it is used for this purpose.

The following simple USART initialization code examples show one assembly and one C function that are equal in functionality. The examples assume asynchronous operation using polling (no interrupts enabled) and a fixed frame format. The baud rate is given as a function parameter.

For the assembly code, the baud rate parameter is assumed to be stored in the r17:r16 Registers.

Assembly Code Example[1]

```
USART_Init:
  ; Set baud rate
  out   UBRRnH, r17
  out   UBRRnL, r16
  ; Enable receiver and transmitter
  ldi   r16, (1<<RXENn)|(1<<TXENn)
  out   UCSRnB,r16
  ; Set frame format: 8data, 2stop bit
  ldi   r16, (1<<USBSn)|(3<<UCSZn0)
  out   UCSRnC,r16
  ret
```

C Code Example[1]

```
#define FOSC 1843200 // Clock Speed
#define BAUD 9600
#define MYUBRR FOSC/16/BAUD-1
void main( void )
{
...
  USART_Init(MYUBRR)
...
}
void USART_Init( unsigned int ubrr)
{
  /*Set baud rate */
  UBRR0H = (unsigned char)(ubrr>>8);
  UBRR0L = (unsigned char)ubrr;
  Enable receiver and transmitter */
  UCSR0B = (1<<RXEN0)|(1<<TXEN0);
  /* Set frame format: 8data, 2stop bit */
  UCSR0C = (1<<USBS0)|(3<<UCSZ00);
}
```

Note: 1. See "Code Examples" on page 7.

More advanced initialization routines can be made that include frame format as parameters, disable interrupts and so on. However, many applications use a fixed setting of the baud and control registers, and for these types of applications the initialization code can be placed directly in the main routine, or be combined with initialization code for other I/O modules.

17.6 Data Transmission – The USART Transmitter

The USART Transmitter is enabled by setting the *Transmit Enable* (TXEN) bit in the UCSRnB Register. When the Transmitter is enabled, the normal port operation of the TxDn pin is overridden by the USART and given the function as the Transmitter's serial output. The baud rate, mode of operation and frame format must be set up once before doing any transmissions. If syn-

chronous operation is used, the clock on the XCKn pin will be overridden and used as transmission clock.

17.6.1 Sending Frames with 5 to 8 Data Bit

A data transmission is initiated by loading the transmit buffer with the data to be transmitted. The CPU can load the transmit buffer by writing to the UDRn I/O location. The buffered data in the transmit buffer will be moved to the Shift Register when the Shift Register is ready to send a new frame. The Shift Register is loaded with new data if it is in idle state (no ongoing transmission) or immediately after the last stop bit of the previous frame is transmitted. When the Shift Register is loaded with new data, it will transfer one complete frame at the rate given by the Baud Register, U2Xn bit or by XCKn depending on mode of operation.

The following code examples show a simple USART transmit function based on polling of the *Data Register Empty* (UDREn) Flag. When using frames with less than eight bits, the most significant bits written to the UDRn are ignored. The USART has to be initialized before the function can be used. For the assembly code, the data to be sent is assumed to be stored in Register R16

Assembly Code Example[1]

```
USART_Transmit:
  ; Wait for empty transmit buffer
  sbis UCSRnA,UDREn
  rjmp USART_Transmit
  ; Put data (r16) into buffer, sends the data
  out  UDRn,r16
  ret
```

C Code Example[1]

```
void USART_Transmit( unsigned char data )
{
  /* Wait for empty transmit buffer */
  while ( !( UCSRnA & (1<<UDREn)) )
       ;
  /* Put data into buffer, sends the data */
  UDRn = data;
}
```

Note: 1. See ”Code Examples” on page 7.

The function simply waits for the transmit buffer to be empty by checking the UDREn Flag, before loading it with new data to be transmitted. If the Data Register Empty interrupt is utilized, the interrupt routine writes the data into the buffer.

17.6.2 Sending Frames with 9 Data Bit

If 9-bit characters are used (UCSZn = 7), the ninth bit must be written to the TXB8 bit in UCSRnB before the low byte of the character is written to UDRn. The following code examples

show a transmit function that handles 9-bit characters. For the assembly code, the data to be sent is assumed to be stored in registers R17:R16.

Assembly Code Example[1][2]

```
USART_Transmit:
  ; Wait for empty transmit buffer
  sbis UCSRnA,UDREn
  rjmp USART_Transmit
  ; Copy 9th bit from r17 to TXB8
  cbi  UCSRnB,TXB8
  sbrc r17,0
  sbi  UCSRnB,TXB8
  ; Put LSB data (r16) into buffer, sends the data
  out  UDRn,r16
  ret
```

C Code Example[1][2]

```
void USART_Transmit( unsigned int data )
{
  /* Wait for empty transmit buffer */
  while ( !( UCSRnA & (1<<UDREn)) )
       ;
  /* Copy 9th bit to TXB8 */
  UCSRnB &= ~(1<<TXB8);
  if ( data & 0x0100 )
    UCSRnB |= (1<<TXB8);
  /* Put data into buffer, sends the data */
  UDRn = data;
}
```

Notes: 1. These transmit functions are written to be general functions. They can be optimized if the contents of the UCSRnB is static. For example, only the TXB8 bit of the UCSRnB Register is used after initialization.
2. See ”Code Examples” on page 7.

The ninth bit can be used for indicating an address frame when using multi processor communication mode or for other protocol handling as for example synchronization.

17.6.3 Transmitter Flags and Interrupts

The USART Transmitter has two flags that indicate its state: USART Data Register Empty (UDREn) and Transmit Complete (TXCn). Both flags can be used for generating interrupts.

The Data Register Empty (UDREn) Flag indicates whether the transmit buffer is ready to receive new data. This bit is set when the transmit buffer is empty, and cleared when the transmit buffer contains data to be transmitted that has not yet been moved into the Shift Register. For compatibility with future devices, always write this bit to zero when writing the UCSRnA Register.

When the Data Register Empty Interrupt Enable (UDRIEn) bit in UCSRnB is written to one, the USART Data Register Empty Interrupt will be executed as long as UDREn is set (provided that global interrupts are enabled). UDREn is cleared by writing UDRn. When interrupt-driven data transmission is used, the Data Register Empty interrupt routine must either write new data to

UDRn in order to clear UDREn or disable the Data Register Empty interrupt, otherwise a new interrupt will occur once the interrupt routine terminates.

The Transmit Complete (TXCn) Flag bit is set one when the entire frame in the Transmit Shift Register has been shifted out and there are no new data currently present in the transmit buffer. The TXCn Flag bit is automatically cleared when a transmit complete interrupt is executed, or it can be cleared by writing a one to its bit location. The TXCn Flag is useful in half-duplex communication interfaces (like the RS-485 standard), where a transmitting application must enter receive mode and free the communication bus immediately after completing the transmission.

When the Transmit Compete Interrupt Enable (TXCIEn) bit in UCSRnB is set, the USART Transmit Complete Interrupt will be executed when the TXCn Flag becomes set (provided that global interrupts are enabled). When the transmit complete interrupt is used, the interrupt handling routine does not have to clear the TXCn Flag, this is done automatically when the interrupt is executed.

17.6.4 Parity Generator

The Parity Generator calculates the parity bit for the serial frame data. When parity bit is enabled (UPMn1 = 1), the transmitter control logic inserts the parity bit between the last data bit and the first stop bit of the frame that is sent.

17.6.5 Disabling the Transmitter

The disabling of the Transmitter (setting the TXEN to zero) will not become effective until ongoing and pending transmissions are completed, i.e., when the Transmit Shift Register and Transmit Buffer Register do not contain data to be transmitted. When disabled, the Transmitter will no longer override the TxDn pin.

17.7 Data Reception – The USART Receiver

The USART Receiver is enabled by writing the Receive Enable (RXENn) bit in the UCSRnB Register to one. When the Receiver is enabled, the normal pin operation of the RxDn pin is overridden by the USART and given the function as the Receiver's serial input. The baud rate, mode of operation and frame format must be set up once before any serial reception can be done. If synchronous operation is used, the clock on the XCKn pin will be used as transfer clock.

17.7.1 Receiving Frames with 5 to 8 Data Bits

The Receiver starts data reception when it detects a valid start bit. Each bit that follows the start bit will be sampled at the baud rate or XCKn clock, and shifted into the Receive Shift Register until the first stop bit of a frame is received. A second stop bit will be ignored by the Receiver. When the first stop bit is received, i.e., a complete serial frame is present in the Receive Shift Register, the contents of the Shift Register will be moved into the receive buffer. The receive buffer can then be read by reading the UDRn I/O location.

The following code example shows a simple USART receive function based on polling of the Receive Complete (RXCn) Flag. When using frames with less than eight bits the most significant

bits of the data read from the UDRn will be masked to zero. The USART has to be initialized before the function can be used.

Assembly Code Example[1]

```
USART_Receive:
  ; Wait for data to be received
  sbis UCSRnA, RXCn
  rjmp USART_Receive
  ; Get and return received data from buffer
  in   r16, UDRn
  ret
```

C Code Example[1]

```
unsigned char USART_Receive( void )
{
  /* Wait for data to be received */
  while ( !(UCSRnA & (1<<RXCn)) )
       ;
  /* Get and return received data from buffer */
  return UDRn;
}
```

Note: 1. See "Code Examples" on page 7.
For I/O Registers located in extended I/O map, "IN", "OUT", "SBIS", "SBIC", "CBI", and "SBI" instructions must be replaced with instructions that allow access to extended I/O. Typically "LDS" and "STS" combined with "SBRS", "SBRC", "SBR", and "CBR".

The function simply waits for data to be present in the receive buffer by checking the RXCn Flag, before reading the buffer and returning the value.

17.7.2 Receiving Frames with 9 Data Bits

If 9-bit characters are used (UCSZn=7) the ninth bit must be read from the RXB8n bit in UCSRnB **before** reading the low bits from the UDRn. This rule applies to the FEn, DORn and UPEn Status Flags as well. Read status from UCSRnA, then data from UDRn. Reading the UDRn I/O location will change the state of the receive buffer FIFO and consequently the TXB8n, FEn, DORn and UPEn bits, which all are stored in the FIFO, will change.

The following code example shows a simple USART receive function that handles both nine bit characters and the status bits.

Assembly Code Example[1]

```
USART_Receive:
  ; Wait for data to be received
  sbis UCSRnA, RXCn
  rjmp USART_Receive
  ; Get status and 9th bit, then data from buffer
  in   r18, UCSRnA
  in   r17, UCSRnB
  in   r16, UDRn
  ; If error, return -1
  andi r18,(1<<FEn)|(1<<DORn)|(1<<UPEn)
  breq USART_ReceiveNoError
  ldi  r17, HIGH(-1)
  ldi  r16, LOW(-1)
USART_ReceiveNoError:
  ; Filter the 9th bit, then return
  lsr  r17
  andi r17, 0x01
  ret
```

C Code Example[1]

```
unsigned int USART_Receive( void )
{
  unsigned char status, resh, resl;
  /* Wait for data to be received */
  while ( !(UCSRnA & (1<<RXCn)) )
       ;
  /* Get status and 9th bit, then data */
  /* from buffer */
  status = UCSRnA;
  resh = UCSRnB;
  resl = UDRn;
  /* If error, return -1 */
  if ( status & (1<<FEn)|(1<<DORn)|(1<<UPEn) )
    return -1;
  /* Filter the 9th bit, then return */
  resh = (resh >> 1) & 0x01;
  return ((resh << 8) | resl);
}
```

Note: 1. See ”Code Examples” on page 7.
For I/O Registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

The receive function example reads all the I/O Registers into the Register File before any computation is done. This gives an optimal receive buffer utilization since the buffer location read will be free to accept new data as early as possible.

17.7.3 Receive Compete Flag and Interrupt

The USART Receiver has one flag that indicates the Receiver state.

The Receive Complete (RXCn) Flag indicates if there are unread data present in the receive buffer. This flag is one when unread data exist in the receive buffer, and zero when the receive buffer is empty (i.e., does not contain any unread data). If the Receiver is disabled (RXENn = 0), the receive buffer will be flushed and consequently the RXCn bit will become zero.

When the Receive Complete Interrupt Enable (RXCIEn) in UCSRnB is set, the USART Receive Complete interrupt will be executed as long as the RXCn Flag is set (provided that global interrupts are enabled). When interrupt-driven data reception is used, the receive complete routine must read the received data from UDRn in order to clear the RXCn Flag, otherwise a new interrupt will occur once the interrupt routine terminates.

17.7.4 Receiver Error Flags

The USART Receiver has three Error Flags: Frame Error (FEn), Data OverRun (DORn) and Parity Error (UPEn). All can be accessed by reading UCSRnA. Common for the Error Flags is that they are located in the receive buffer together with the frame for which they indicate the error status. Due to the buffering of the Error Flags, the UCSRnA must be read before the receive buffer (UDRn), since reading the UDRn I/O location changes the buffer read location. Another equality for the Error Flags is that they can not be altered by software doing a write to the flag location. However, all flags must be set to zero when the UCSRnA is written for upward compatibility of future USART implementations. None of the Error Flags can generate interrupts.

The Frame Error (FEn) Flag indicates the state of the first stop bit of the next readable frame stored in the receive buffer. The FEn Flag is zero when the stop bit was correctly read (as one), and the FEn Flag will be one when the stop bit was incorrect (zero). This flag can be used for detecting out-of-sync conditions, detecting break conditions and protocol handling. The FEn Flag is not affected by the setting of the USBSn bit in UCSRnC since the Receiver ignores all, except for the first, stop bits. For compatibility with future devices, always set this bit to zero when writing to UCSRnA.

The Data OverRun (DORn) Flag indicates data loss due to a receiver buffer full condition. A Data OverRun occurs when the receive buffer is full (two characters), it is a new character waiting in the Receive Shift Register, and a new start bit is detected. If the DORn Flag is set there was one or more serial frame lost between the frame last read from UDRn, and the next frame read from UDRn. For compatibility with future devices, always write this bit to zero when writing to UCSRnA. The DORn Flag is cleared when the frame received was successfully moved from the Shift Register to the receive buffer.

The Parity Error (UPEn) Flag indicates that the next frame in the receive buffer had a Parity Error when received. If Parity Check is not enabled the UPEn bit will always be read zero. For compatibility with future devices, always set this bit to zero when writing to UCSRnA. For more details see ”Parity Bit Calculation” on page 181 and ”Parity Checker” on page 189.

17.7.5 Parity Checker

The Parity Checker is active when the high USART Parity mode (UPMn1) bit is set. Type of Parity Check to be performed (odd or even) is selected by the UPMn0 bit. When enabled, the Parity Checker calculates the parity of the data bits in incoming frames and compares the result with the parity bit from the serial frame. The result of the check is stored in the receive buffer together with the received data and stop bits. The Parity Error (UPEn) Flag can then be read by software to check if the frame had a Parity Error.

The UPEn bit is set if the next character that can be read from the receive buffer had a Parity Error when received and the Parity Checking was enabled at that point (UPMn1 = 1). This bit is valid until the receive buffer (UDRn) is read.

17.7.6 Disabling the Receiver

In contrast to the Transmitter, disabling of the Receiver will be immediate. Data from ongoing receptions will therefore be lost. When disabled (i.e., the RXENn is set to zero) the Receiver will no longer override the normal function of the RxDn port pin. The Receiver buffer FIFO will be flushed when the Receiver is disabled. Remaining data in the buffer will be lost

17.7.7 Flushing the Receive Buffer

The receiver buffer FIFO will be flushed when the Receiver is disabled, i.e., the buffer will be emptied of its contents. Unread data will be lost. If the buffer has to be flushed during normal operation, due to for instance an error condition, read the UDRn I/O location until the RXCn Flag is cleared. The following code example shows how to flush the receive buffer.

Assembly Code Example[1]

```
USART_Flush:
  sbis UCSRnA, RXCn
  ret
  in   r16, UDRn
  rjmp USART_Flush
```

C Code Example[1]

```
void USART_Flush( void )
{
  unsigned char dummy;
  while ( UCSRnA & (1<<RXCn) ) dummy = UDRn;
}
```

Note: 1. See ”Code Examples” on page 7.
For I/O Registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

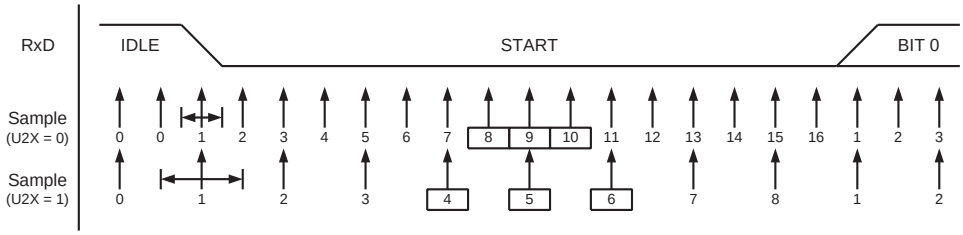
17.8 Asynchronous Data Reception

The USART includes a clock recovery and a data recovery unit for handling asynchronous data reception. The clock recovery logic is used for synchronizing the internally generated baud rate clock to the incoming asynchronous serial frames at the RxDn pin. The data recovery logic samples and low pass filters each incoming bit, thereby improving the noise immunity of the Receiver. The asynchronous reception operational range depends on the accuracy of the internal baud rate clock, the rate of the incoming frames, and the frame size in number of bits.

17.8.1 Asynchronous Clock Recovery

The clock recovery logic synchronizes internal clock to the incoming serial frames. Figure 17-5 illustrates the sampling process of the start bit of an incoming frame. The sample rate is 16 times the baud rate for Normal mode, and eight times the baud rate for Double Speed mode. The horizontal arrows illustrate the synchronization variation due to the sampling process. Note the larger time variation when using the Double Speed mode (U2Xn = 1) of operation. Samples denoted zero are samples done when the RxDn line is idle (i.e., no communication activity).

Figure 17-5. Start Bit Sampling

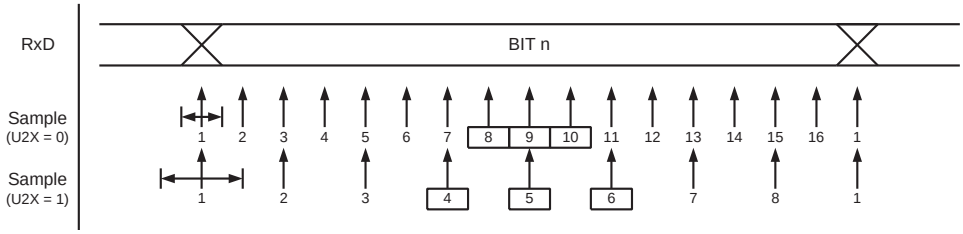


When the clock recovery logic detects a high (idle) to low (start) transition on the RxDn line, the start bit detection sequence is initiated. Let sample 1 denote the first zero-sample as shown in the figure. The clock recovery logic then uses samples 8, 9, and 10 for Normal mode, and samples 4, 5, and 6 for Double Speed mode (indicated with sample numbers inside boxes on the figure), to decide if a valid start bit is received. If two or more of these three samples have logical high levels (the majority wins), the start bit is rejected as a noise spike and the Receiver starts looking for the next high to low-transition. If however, a valid start bit is detected, the clock recovery logic is synchronized and the data recovery can begin. The synchronization process is repeated for each start bit.

17.8.2 Asynchronous Data Recovery

When the receiver clock is synchronized to the start bit, the data recovery can begin. The data recovery unit uses a state machine that has 16 states for each bit in Normal mode and eight states for each bit in Double Speed mode. Figure 17-6 shows the sampling of the data bits and the parity bit. Each of the samples is given a number that is equal to the state of the recovery unit.

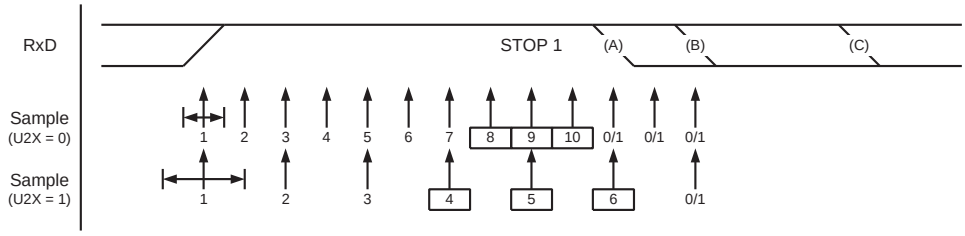
Figure 17-6. Sampling of Data and Parity Bit



The decision of the logic level of the received bit is taken by doing a majority voting of the logic value to the three samples in the center of the received bit. The center samples are emphasized on the figure by having the sample number inside boxes. The majority voting process is done as follows: If two or all three samples have high levels, the received bit is registered to be a logic 1. If two or all three samples have low levels, the received bit is registered to be a logic 0. This majority voting process acts as a low pass filter for the incoming signal on the RxDn pin. The recovery process is then repeated until a complete frame is received. Including the first stop bit. Note that the Receiver only uses the first stop bit of a frame.

Figure 17-7 on page 192 shows the sampling of the stop bit and the earliest possible beginning of the start bit of the next frame.

Figure 17-7. Stop Bit Sampling and Next Start Bit Sampling



The same majority voting is done to the stop bit as done for the other bits in the frame. If the stop bit is registered to have a logic 0 value, the Frame Error (FEn) Flag will be set.

A new high to low transition indicating the start bit of a new frame can come right after the last of the bits used for majority voting. For Normal Speed mode, the first low level sample can be at point marked (A) in Figure 17-7. For Double Speed mode the first low level must be delayed to (B). (C) marks a stop bit of full length. The early start bit detection influences the operational range of the Receiver.

17.8.3 Asynchronous Operational Range

The operational range of the Receiver is dependent on the mismatch between the received bit rate and the internally generated baud rate. If the Transmitter is sending frames at too fast or too slow bit rates, or the internally generated baud rate of the Receiver does not have a similar (see Table 17-2 on page 193) base frequency, the Receiver will not be able to synchronize the frames to the start bit.

The following equations can be used to calculate the ratio of the incoming data rate and internal receiver baud rate.

$$R_{slow} = \frac{(D+1)S}{S-1+D\cdot S+S_F} \qquad R_{fast} = \frac{(D+2)S}{(D+1)S+S_M}$$

D Sum of character size and parity size (D = 5 to 10 bit)

S Samples per bit. S = 16 for Normal Speed mode and S = 8 for Double Speed mode.

S_F First sample number used for majority voting. S_F = 8 for normal speed and S_F = 4 for Double Speed mode.

S_M Middle sample number used for majority voting. S_M = 9 for normal speed and S_M = 5 for Double Speed mode.

R_{slow} is the ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate. R_{fast} is the ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate.

Table 17-2 on page 193 and Table 17-3 on page 193 list the maximum receiver baud rate error that can be tolerated. Note that Normal Speed mode has higher toleration of baud rate variations.

Table 17-2. Recommended Maximum Receiver Baud Rate Error for Normal Speed Mode (U2Xn = 0)

D # (Data+Parity Bit)	R_{slow} (%)	R_{fast} (%)	Max Total Error (%)	Recommended Max Receiver Error (%)
5	93.20	106.67	+6.67/-6.8	± 3.0
6	94.12	105.79	+5.79/-5.88	± 2.5
7	94.81	105.11	+5.11/-5.19	± 2.0
8	95.36	104.58	+4.58/-4.54	± 2.0
9	95.81	104.14	+4.14/-4.19	± 1.5
10	96.17	103.78	+3.78/-3.83	± 1.5

Table 17-3. Recommended Maximum Receiver Baud Rate Error for Double Speed Mode (U2Xn = 1)

D # (Data+Parity Bit)	R_{slow} (%)	R_{fast} (%)	Max Total Error (%)	Recommended Max Receiver Error (%)
5	94.12	105.66	+5.66/-5.88	± 2.5
6	94.92	104.92	+4.92/-5.08	± 2.0
7	95.52	104,35	+4.35/-4.48	± 1.5
8	96.00	103.90	+3.90/-4.00	± 1.5
9	96.39	103.53	+3.53/-3.61	± 1.5
10	96.70	103.23	+3.23/-3.30	± 1.0

The recommendations of the maximum receiver baud rate error was made under the assumption that the Receiver and Transmitter equally divides the maximum total error.

There are two possible sources for the receivers baud rate error. The Receiver's system clock (XTAL) will always have some minor instability over the supply voltage range and the temperature range. When using a crystal to generate the system clock, this is rarely a problem, but for a resonator the system clock may differ more than 2% depending of the resonators tolerance. The second source for the error is more controllable. The baud rate generator can not always do an exact division of the system frequency to get the baud rate wanted. In this case an UBRRn value that gives an acceptable low error can be used if possible.

17.9 Multi-processor Communication Mode

Setting the Multi-processor Communication mode (MPCMn) bit in UCSRnA enables a filtering function of incoming frames received by the USART Receiver. Frames that do not contain address information will be ignored and not put into the receive buffer. This effectively reduces the number of incoming frames that has to be handled by the CPU, in a system with multiple MCUs that communicate via the same serial bus. The Transmitter is unaffected by the MPCMn setting, but has to be used differently when it is a part of a system utilizing the Multi-processor Communication mode.

If the Receiver is set up to receive frames that contain 5 to 8 data bits, then the first stop bit indicates if the frame contains data or address information. If the Receiver is set up for frames with

nine data bits, then the ninth bit (RXB8n) is used for identifying address and data frames. When the frame type bit (the first stop or the ninth bit) is one, the frame contains an address. When the frame type bit is zero the frame is a data frame.

The Multi-processor Communication mode enables several slave MCUs to receive data from a master MCU. This is done by first decoding an address frame to find out which MCU has been addressed. If a particular slave MCU has been addressed, it will receive the following data frames as normal, while the other slave MCUs will ignore the received frames until another address frame is received.

17.9.1 Using MPCMn

For an MCU to act as a master MCU, it can use a 9-bit character frame format (UCSZn = 7). The ninth bit (TXB8n) must be set when an address frame (TXB8n = 1) or cleared when a data frame (TXB = 0) is being transmitted. The slave MCUs must in this case be set to use a 9-bit character frame format.

The following procedure should be used to exchange data in Multi-processor Communication mode:

1. All Slave MCUs are in Multi-processor Communication mode (MPCMn in UCSRnA is set).
2. The Master MCU sends an address frame, and all slaves receive and read this frame. In the Slave MCUs, the RXCn Flag in UCSRnA will be set as normal.
3. Each Slave MCU reads the UDRn Register and determines if it has been selected. If so, it clears the MPCMn bit in UCSRnA, otherwise it waits for the next address byte and keeps the MPCMn setting.
4. The addressed MCU will receive all data frames until a new address frame is received. The other Slave MCUs, which still have the MPCMn bit set, will ignore the data frames.
5. When the last data frame is received by the addressed MCU, the addressed MCU sets the MPCMn bit and waits for a new address frame from master. The process then repeats from 2.

Using any of the 5- to 8-bit character frame formats is possible, but impractical since the Receiver must change between using n and n+1 character frame formats. This makes full-duplex operation difficult since the Transmitter and Receiver uses the same character size setting. If 5- to 8-bit character frames are used, the Transmitter must be set to use two stop bit (USBSn = 1) since the first stop bit is used for indicating the frame type.

Do not use Read-Modify-Write instructions (SBI and CBI) to set or clear the MPCMn bit. The MPCMn bit shares the same I/O location as the TXCn Flag and this might accidentally be cleared when using SBI or CBI instructions.

17.10 Register Description

17.10.1 UDRn – USART I/O Data Register n

Bit	7	6	5	4	3	2	1	0	
	RXB[7:0]								UDRn (Read)
	TXB[7:0]								UDRn (Write)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The USART Transmit Data Buffer Register and USART Receive Data Buffer Registers share the same I/O address referred to as USART Data Register or UDRn. The Transmit Data Buffer Register (TXB) will be the destination for data written to the UDRn Register location. Reading the UDRn Register location will return the contents of the Receive Data Buffer Register (RXB).

For 5-, 6-, or 7-bit characters the upper unused bits will be ignored by the Transmitter and set to zero by the Receiver.

The transmit buffer can only be written when the UDREn Flag in the UCSRnA Register is set. Data written to UDRn when the UDREn Flag is not set, will be ignored by the USART Transmitter. When data is written to the transmit buffer, and the Transmitter is enabled, the Transmitter will load the data into the Transmit Shift Register when the Shift Register is empty. Then the data will be serially transmitted on the TxDn pin.

The receive buffer consists of a two level FIFO. The FIFO will change its state whenever the receive buffer is accessed. Due to this behavior of the receive buffer, do not use Read-Modify-Write instructions (SBI and CBI) on this location. Be careful when using bit test instructions (SBIC and SBIS), since these also will change the state of the FIFO.

17.10.2 UCSRnA – USART Control and Status Register n A

Bit	7	6	5	4	3	2	1	0	
	RXCn	TXCn	UDREn	FEn	DORn	UPEn	U2Xn	MPCMn	UCSRnA
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

- **Bit 7 – RXCn: USART Receive Complete**

This flag bit is set when there are unread data in the receive buffer and cleared when the receive buffer is empty (i.e., does not contain any unread data). If the Receiver is disabled, the receive buffer will be flushed and consequently the RXCn bit will become zero. The RXCn Flag can be used to generate a Receive Complete interrupt (see description of the RXCIEn bit).

- **Bit 6 – TXCn: USART Transmit Complete**

This flag bit is set when the entire frame in the Transmit Shift Register has been shifted out and there are no new data currently present in the transmit buffer (UDRn). The TXCn Flag bit is automatically cleared when a transmit complete interrupt is executed, or it can be cleared by writing a one to its bit location. The TXCn Flag can generate a Transmit Complete interrupt (see description of the TXCIEn bit).

- **Bit 5 – UDREn: USART Data Register Empty**

The UDREn Flag indicates if the transmit buffer (UDRn) is ready to receive new data. If UDREn is one, the buffer is empty, and therefore ready to be written. The UDREn Flag can generate a

Data Register Empty interrupt (see description of the UDRIEn bit). UDREn is set after a reset to indicate that the Transmitter is ready.

- **Bit 4 – FEn: Frame Error**

This bit is set if the next character in the receive buffer had a Frame Error when received. I.e., when the first stop bit of the next character in the receive buffer is zero. This bit is valid until the receive buffer (UDRn) is read. The FEn bit is zero when the stop bit of received data is one. Always set this bit to zero when writing to UCSRnA.

- **Bit 3 – DORn: Data OverRun**

This bit is set if a Data OverRun condition is detected. A Data OverRun occurs when the receive buffer is full (two characters), it is a new character waiting in the Receive Shift Register, and a new start bit is detected. This bit is valid until the receive buffer (UDRn) is read. Always set this bit to zero when writing to UCSRnA.

- **Bit 2 – UPEn: USART Parity Error**

This bit is set if the next character in the receive buffer had a Parity Error when received and the Parity Checking was enabled at that point (UPMn1 = 1). This bit is valid until the receive buffer (UDRn) is read. Always set this bit to zero when writing to UCSRnA.

- **Bit 1 – U2Xn: Double the USART Transmission Speed**

This bit only has effect for the asynchronous operation. Write this bit to zero when using synchronous operation.

Writing this bit to one will reduce the divisor of the baud rate divider from 16 to 8 effectively doubling the transfer rate for asynchronous communication.

- **Bit 0 – MPCMn: Multi-processor Communication Mode**

This bit enables the Multi-processor Communication mode. When the MPCMn bit is written to one, all the incoming frames received by the USART Receiver that do not contain address information will be ignored. The Transmitter is unaffected by the MPCMn setting. For more detailed information see ”Multi-processor Communication Mode” on page 193.

17.10.3 UCSRnB – USART Control and Status Register n B

Bit	7	6	5	4	3	2	1	0	
	RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n	UCSRnB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – RXCIEn: RX Complete Interrupt Enable n**

Writing this bit to one enables interrupt on the RXCn Flag. A USART Receive Complete interrupt will be generated only if the RXCIEn bit is written to one, the Global Interrupt Flag in SREG is written to one and the RXCn bit in UCSRnA is set.

- **Bit 6 – TXCIEn: TX Complete Interrupt Enable n**

Writing this bit to one enables interrupt on the TXCn Flag. A USART Transmit Complete interrupt will be generated only if the TXCIEn bit is written to one, the Global Interrupt Flag in SREG is written to one and the TXCn bit in UCSRnA is set.

- **Bit 5 – UDRIEn: USART Data Register Empty Interrupt Enable n**

Writing this bit to one enables interrupt on the UDREn Flag. A Data Register Empty interrupt will be generated only if the UDRIEn bit is written to one, the Global Interrupt Flag in SREG is written to one and the UDREn bit in UCSRnA is set.

- **Bit 4 – RXENn: Receiver Enable n**

Writing this bit to one enables the USART Receiver. The Receiver will override normal port operation for the RxDn pin when enabled. Disabling the Receiver will flush the receive buffer invalidating the FEn, DORn, and UPEn Flags.

- **Bit 3 – TXENn: Transmitter Enable n**

Writing this bit to one enables the USART Transmitter. The Transmitter will override normal port operation for the TxDn pin when enabled. The disabling of the Transmitter (writing TXENn to zero) will not become effective until ongoing and pending transmissions are completed, i.e., when the Transmit Shift Register and Transmit Buffer Register do not contain data to be transmitted. When disabled, the Transmitter will no longer override the TxDn port.

- **Bit 2 – UCSZn2: Character Size n**

The UCSZn2 bits combined with the UCSZn1:0 bit in UCSRnC sets the number of data bits (Character SiZe) in a frame the Receiver and Transmitter use.

- **Bit 1 – RXB8n: Receive Data Bit 8 n**

RXB8n is the ninth data bit of the received character when operating with serial frames with nine data bits. Must be read before reading the low bits from UDRn.

- **Bit 0 – TXB8n: Transmit Data Bit 8 n**

TXB8n is the ninth data bit in the character to be transmitted when operating with serial frames with nine data bits. Must be written before writing the low bits to UDRn.

17.10.4 UCSRnC – USART Control and Status Register n C

Bit	7	6	5	4	3	2	1	0	
	UMSELn1	**UMSELn0**	**UPMn1**	**UPMn0**	**USBSn**	**UCSZn1**	**UCSZn0**	**UCPOLn**	**UCSRnC**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	1	1	0	

- **Bits 7:6 – UMSELn1:0 USART Mode Select**

These bits select the mode of operation of the USARTn as shown in Table 17-4.

Table 17-4. UMSELn Bits Settings

UMSELn1	UMSELn0	Mode
0	0	Asynchronous USART
0	1	Synchronous USART
1	0	(Reserved)
1	1	Master SPI (MSPIM)[1]

Note: 1. See "USART in SPI Mode" on page 204 for full description of the Master SPI Mode (MSPIM) operation

- **Bits 5:4 – UPMn1:0: Parity Mode**

These bits enable and set type of parity generation and check. If enabled, the Transmitter will automatically generate and send the parity of the transmitted data bits within each frame. The Receiver will generate a parity value for the incoming data and compare it to the UPMn setting. If a mismatch is detected, the UPEn Flag in UCSRnA will be set.

Table 17-5. UPMn Bits Settings

UPMn1	UPMn0	Parity Mode
0	0	Disabled
0	1	Reserved
1	0	Enabled, Even Parity
1	1	Enabled, Odd Parity

- **Bit 3 – USBSn: Stop Bit Select**

This bit selects the number of stop bits to be inserted by the Transmitter. The Receiver ignores this setting.

Table 17-6. USBS Bit Settings

USBSn	Stop Bit(s)
0	1-bit
1	2-bit

- **Bit 2:1 – UCSZn1:0: Character Size**

The UCSZn1:0 bits combined with the UCSZn2 bit in UCSRnB sets the number of data bits (Character SiZe) in a frame the Receiver and Transmitter use.

Table 17-7. UCSZn Bits Settings

UCSZn2	UCSZn1	UCSZn0	Character Size
0	0	0	5-bit
0	0	1	6-bit
0	1	0	7-bit
0	1	1	8-bit
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	9-bit

- **Bit 0 – UCPOLn: Clock Polarity**

This bit is used for synchronous mode only. Write this bit to zero when asynchronous mode is used. The UCPOLn bit sets the relationship between data output change and data input sample, and the synchronous clock (XCKn).

Table 17-8. UCPOLn Bit Settings

UCPOLn	Transmitted Data Changed (Output of TxDn Pin)	Received Data Sampled (Input on RxDn Pin)
0	Rising XCKn Edge	Falling XCKn Edge
1	Falling XCKn Edge	Rising XCKn Edge

17.10.5 UBRRnL and UBRRnH – USART Baud Rate Registers

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	UBRRn[11:8]				UBRRnH
	UBRRn[7:0]								UBRRnL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

- **Bit 15:12 – Reserved Bits**

These bits are reserved for future use. For compatibility with future devices, these bit must be written to zero when UBRRnH is written.

- **Bit 11:0 – UBRR11:0: USART Baud Rate Register**

This is a 12-bit register which contains the USART baud rate. The UBRRnH contains the four most significant bits, and the UBRRnL contains the eight least significant bits of the USART baud rate. Ongoing transmissions by the Transmitter and Receiver will be corrupted if the baud rate is changed. Writing UBRRnL will trigger an immediate update of the baud rate prescaler.

17.11 Examples of Baud Rate Setting

For standard crystal and resonator frequencies, the most commonly used baud rates for asynchronous operation can be generated by using the UBRRn settings in Table 17-9. UBRRn values which yield an actual baud rate differing less than 0.5% from the target baud rate, are bold in the table. Higher error ratings are acceptable, but the Receiver will have less noise resistance when the error ratings are high, especially for large serial frames (see ”Asynchronous Operational Range” on page 192). The error values are calculated using the following equation:

$$\text{Error}[\%] = \left(\frac{\text{BaudRate}_{\text{Closest Match}}}{\text{BaudRate}} - 1\right) \bullet 100\%$$

Table 17-9. Examples of UBRRn Settings for Commonly Used Oscillator Frequencies

Baud Rate (bps)	f_{osc} = 1.0000 MHz				f_{osc} = 1.8432 MHz				f_{osc} = 2.0000 MHz			
	U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1	
	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error
2400	25	0.2%	51	0.2%	47	0.0%	95	0.0%	51	0.2%	103	0.2%
4800	12	0.2%	25	0.2%	23	0.0%	47	0.0%	25	0.2%	51	0.2%
9600	6	-7.0%	12	0.2%	11	0.0%	23	0.0%	12	0.2%	25	0.2%
14.4k	3	8.5%	8	-3.5%	7	0.0%	15	0.0%	8	-3.5%	16	2.1%
19.2k	2	8.5%	6	-7.0%	5	0.0%	11	0.0%	6	-7.0%	12	0.2%
28.8k	1	8.5%	3	8.5%	3	0.0%	7	0.0%	3	8.5%	8	-3.5%
38.4k	1	-18.6%	2	8.5%	2	0.0%	5	0.0%	2	8.5%	6	-7.0%
57.6k	0	8.5%	1	8.5%	1	0.0%	3	0.0%	1	8.5%	3	8.5%
76.8k	–	–	1	-18.6%	1	-25.0%	2	0.0%	1	-18.6%	2	8.5%
115.2k	–	–	0	8.5%	0	0.0%	1	0.0%	0	8.5%	1	8.5%
230.4k	–	–	–	–	–	–	0	0.0%	–	–	–	–
250k	–	–	–	–	–	–	–	–	–	–	0	0.0%
Max.[1]	62.5 kbps		125 kbps		115.2 kbps		230.4 kbps		125 kbps		250 kbps	

Note: 1. UBRRn = 0, Error = 0.0%

Table 17-10. Examples of UBRRn Settings for Commonly Used Oscillator Frequencies (Continued)

Baud Rate (bps)	f_{osc} = 3.6864 MHz				f_{osc} = 4.0000 MHz				f_{osc} = 7.3728 MHz			
	U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1	
	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error
2400	95	0.0%	191	0.0%	103	0.2%	207	0.2%	191	0.0%	383	0.0%
4800	47	0.0%	95	0.0%	51	0.2%	103	0.2%	95	0.0%	191	0.0%
9600	23	0.0%	47	0.0%	25	0.2%	51	0.2%	47	0.0%	95	0.0%
14.4k	15	0.0%	31	0.0%	16	2.1%	34	-0.8%	31	0.0%	63	0.0%
19.2k	11	0.0%	23	0.0%	12	0.2%	25	0.2%	23	0.0%	47	0.0%
28.8k	7	0.0%	15	0.0%	8	-3.5%	16	2.1%	15	0.0%	31	0.0%
38.4k	5	0.0%	11	0.0%	6	-7.0%	12	0.2%	11	0.0%	23	0.0%
57.6k	3	0.0%	7	0.0%	3	8.5%	8	-3.5%	7	0.0%	15	0.0%
76.8k	2	0.0%	5	0.0%	2	8.5%	6	-7.0%	5	0.0%	11	0.0%
115.2k	1	0.0%	3	0.0%	1	8.5%	3	8.5%	3	0.0%	7	0.0%
230.4k	0	0.0%	1	0.0%	0	8.5%	1	8.5%	1	0.0%	3	0.0%
250k	0	-7.8%	1	-7.8%	0	0.0%	1	0.0%	1	-7.8%	3	-7.8%
0.5M	–	–	0	-7.8%	–	–	0	0.0%	0	-7.8%	1	-7.8%
1M	–	–	–	–	–	–	–	–	–	–	0	-7.8%
Max. (1)	230.4 kbps		460.8 kbps		250 kbps		0.5 Mbps		460.8 kbps		921.6 kbps	

1. UBRRn = 0, Error = 0.0%

Table 17-11. Examples of UBRRn Settings for Commonly Used Oscillator Frequencies (Continued)

Baud Rate (bps)	f_{osc} = 8.0000 MHz				f_{osc} = 11.0592 MHz				f_{osc} = 14.7456 MHz			
	U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1	
	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error
2400	207	0.2%	416	-0.1%	287	0.0%	575	0.0%	383	0.0%	767	0.0%
4800	103	0.2%	207	0.2%	143	0.0%	287	0.0%	191	0.0%	383	0.0%
9600	51	0.2%	103	0.2%	71	0.0%	143	0.0%	95	0.0%	191	0.0%
14.4k	34	-0.8%	68	0.6%	47	0.0%	95	0.0%	63	0.0%	127	0.0%
19.2k	25	0.2%	51	0.2%	35	0.0%	71	0.0%	47	0.0%	95	0.0%
28.8k	16	2.1%	34	-0.8%	23	0.0%	47	0.0%	31	0.0%	63	0.0%
38.4k	12	0.2%	25	0.2%	17	0.0%	35	0.0%	23	0.0%	47	0.0%
57.6k	8	-3.5%	16	2.1%	11	0.0%	23	0.0%	15	0.0%	31	0.0%
76.8k	6	-7.0%	12	0.2%	8	0.0%	17	0.0%	11	0.0%	23	0.0%
115.2k	3	8.5%	8	-3.5%	5	0.0%	11	0.0%	7	0.0%	15	0.0%
230.4k	1	8.5%	3	8.5%	2	0.0%	5	0.0%	3	0.0%	7	0.0%
250k	1	0.0%	3	0.0%	2	-7.8%	5	-7.8%	3	-7.8%	6	5.3%
0.5M	0	0.0%	1	0.0%	–	–	2	-7.8%	1	-7.8%	3	-7.8%
1M	–	–	0	0.0%	–	–	–	–	0	-7.8%	1	-7.8%
Max. (1)	0.5 Mbps		1 Mbps		691.2 kbps		1.3824 Mbps		921.6 kbps		1.8432 Mbps	

1. UBRRn = 0, Error = 0.0%

Table 17-12. Examples of UBRRn Settings for Commonly Used Oscillator Frequencies (Continued)

Baud Rate (bps)	f_{osc} = 16.0000 MHz				f_{osc} = 18.4320 MHz				f_{osc} = 20.0000 MHz			
	U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1	
	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error
2400	416	-0.1%	832	0.0%	479	0.0%	959	0.0%	520	0.0%	1041	0.0%
4800	207	0.2%	416	-0.1%	239	0.0%	479	0.0%	259	0.2%	520	0.0%
9600	103	0.2%	207	0.2%	119	0.0%	239	0.0%	129	0.2%	259	0.2%
14.4k	68	0.6%	138	-0.1%	79	0.0%	159	0.0%	86	-0.2%	173	-0.2%
19.2k	51	0.2%	103	0.2%	59	0.0%	119	0.0%	64	0.2%	129	0.2%
28.8k	34	-0.8%	68	0.6%	39	0.0%	79	0.0%	42	0.9%	86	-0.2%
38.4k	25	0.2%	51	0.2%	29	0.0%	59	0.0%	32	-1.4%	64	0.2%
57.6k	16	2.1%	34	-0.8%	19	0.0%	39	0.0%	21	-1.4%	42	0.9%
76.8k	12	0.2%	25	0.2%	14	0.0%	29	0.0%	15	1.7%	32	-1.4%
115.2k	8	-3.5%	16	2.1%	9	0.0%	19	0.0%	10	-1.4%	21	-1.4%
230.4k	3	8.5%	8	-3.5%	4	0.0%	9	0.0%	4	8.5%	10	-1.4%
250k	3	0.0%	7	0.0%	4	-7.8%	8	2.4%	4	0.0%	9	0.0%
0.5M	1	0.0%	3	0.0%	–	–	4	-7.8%	–	–	4	0.0%
1M	0	0.0%	1	0.0%	–	–	–	–	–	–	–	–
Max. (1)	1 Mbps		2 Mbps		1.152 Mbps		2.304 Mbps		1.25 Mbps		2.5 Mbps	

1. UBRRn = 0, Error = 0.0%

18. USART in SPI Mode

18.1 Features

- **Full Duplex, Three-wire Synchronous Data Transfer**
- **Master Operation**
- **Supports all four SPI Modes of Operation (Mode 0, 1, 2, and 3)**
- **LSB First or MSB First Data Transfer (Configurable Data Order)**
- **Queued Operation (Double Buffered)**
- **High Resolution Baud Rate Generator**
- **High Speed Operation (fXCKmax = f_{CK}/2)**
- **Flexible Interrupt Generation**

18.2 Overview

The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) can be set to a master SPI compliant mode of operation.

Setting both UMSELn1:0 bits to one enables the USART in MSPIM logic. In this mode of operation the SPI master control logic takes direct control over the USART resources. These resources include the transmitter and receiver shift register and buffers, and the baud rate generator. The parity generator and checker, the data and clock recovery logic, and the RX and TX control logic is disabled. The USART RX and TX control logic is replaced by a common SPI transfer control logic. However, the pin control logic and interrupt generation logic is identical in both modes of operation.

The I/O register locations are the same in both modes. However, some of the functionality of the control registers changes when using MSPIM.

18.3 Clock Generation

The Clock Generation logic generates the base clock for the Transmitter and Receiver. For USART MSPIM mode of operation only internal clock generation (i.e. master operation) is supported. The Data Direction Register for the XCKn pin (DDR_XCKn) must therefore be set to one (i.e. as output) for the USART in MSPIM to operate correctly. Preferably the DDR_XCKn should be set up before the USART in MSPIM is enabled (i.e. TXENn and RXENn bit set to one).

The internal clock generation used in MSPIM mode is identical to the USART synchronous master mode. The baud rate or UBRRn setting can therefore be calculated using the same equations, see Table 18-1:

Table 18-1. Equations for Calculating Baud Rate Register Setting

Operating Mode	Equation for Calculating Baud Rate[1]	Equation for Calculating UBRRn Value
Synchronous Master mode	$BAUD = \frac{f_{OSC}}{2(\boldsymbol{UBRR}n + 1)}$	$\boldsymbol{UBRR}n = \frac{f_{OSC}}{2BAUD} - 1$

Note: 1. The baud rate is defined to be the transfer rate in bit per second (bps)

BAUD Baud rate (in bits per second, bps)

f_{osc} System Oscillator clock frequency

UBRRn Contents of the UBRRnH and UBRRnL Registers, (0-4095)

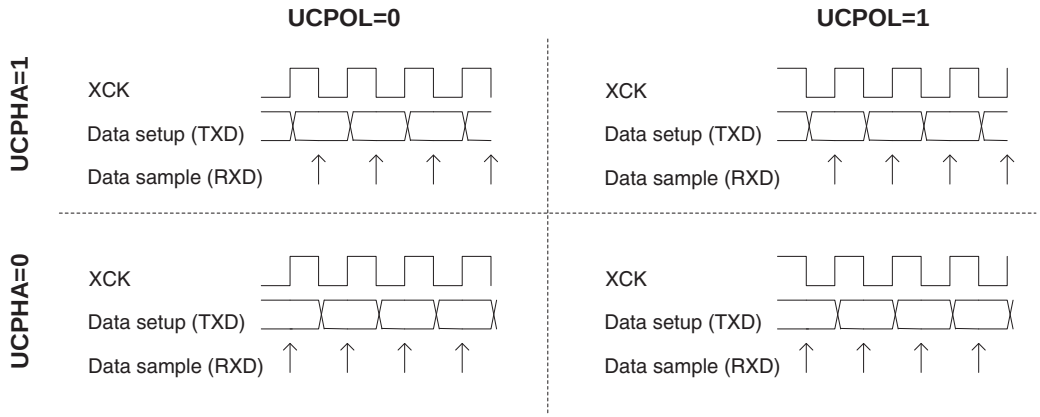
18.4 SPI Data Modes and Timing

There are four combinations of XCKn (SCK) phase and polarity with respect to serial data, which are determined by control bits UCPHAn and UCPOLn. The data transfer timing diagrams are shown in Figure 18-1. Data bits are shifted out and latched in on opposite edges of the XCKn signal, ensuring sufficient time for data signals to stabilize. The UCPOLn and UCPHAn functionality is summarized in Table 18-2. Note that changing the setting of any of these bits will corrupt all ongoing communication for both the Receiver and Transmitter.

Table 18-2. UCPOLn and UCPHAn Functionality-

UCPOLn	UCPHAn	SPI Mode	Leading Edge	Trailing Edge
0	0	0	Sample (Rising)	Setup (Falling)
0	1	1	Setup (Rising)	Sample (Falling)
1	0	2	Sample (Falling)	Setup (Rising)
1	1	3	Setup (Falling)	Sample (Rising)

Figure 18-1. UCPHAn and UCPOLn data transfer timing diagrams.



18.5 Frame Formats

A serial frame for the MSPIM is defined to be one character of 8 data bits. The USART in MSPIM mode has two valid frame formats:

- 8-bit data with MSB first
- 8-bit data with LSB first

A frame starts with the least or most significant data bit. Then the next data bits, up to a total of eight, are succeeding, ending with the most or least significant bit accordingly. When a complete frame is transmitted, a new frame can directly follow it, or the communication line can be set to an idle (high) state.

The UDORDn bit in UCSRnC sets the frame format used by the USART in MSPIM mode. The Receiver and Transmitter use the same setting. Note that changing the setting of any of these bits will corrupt all ongoing communication for both the Receiver and Transmitter.

16-bit data transfer can be achieved by writing two data bytes to UDRn. A UART transmit complete interrupt will then signal that the 16-bit value has been shifted out.

18.5.1 USART MSPIM Initialization

The USART in MSPIM mode has to be initialized before any communication can take place. The initialization process normally consists of setting the baud rate, setting master mode of operation (by setting DDR_XCKn to one), setting frame format and enabling the Transmitter and the Receiver. Only the transmitter can operate independently. For interrupt driven USART operation, the Global Interrupt Flag should be cleared (and thus interrupts globally disabled) when doing the initialization.

Note: To ensure immediate initialization of the XCKn output the baud-rate register (UBRRn) must be zero at the time the transmitter is enabled. Contrary to the normal mode USART operation the UBRRn must then be written to the desired value after the transmitter is enabled, but before the first transmission is started. Setting UBRRn to zero before enabling the transmitter is not necessary if the initialization is done immediately after a reset since UBRRn is reset to zero.

Before doing a re-initialization with changed baud rate, data mode, or frame format, be sure that there is no ongoing transmissions during the period the registers are changed. The TXCn Flag can be used to check that the Transmitter has completed all transfers, and the RXCn Flag can

be used to check that there are no unread data in the receive buffer. Note that the TXCn Flag must be cleared before each transmission (before UDRn is written) if it is used for this purpose.

The following simple USART initialization code examples show one assembly and one C function that are equal in functionality. The examples assume polling (no interrupts enabled). The baud rate is given as a function parameter. For the assembly code, the baud rate parameter is assumed to be stored in the r17:r16 registers.

Assembly Code Example[1]

```
USART_Init:
  clr r18
  out UBRRnH,r18
  out UBRRnL,r18
  ; Setting the XCKn port pin as output, enables master mode.
  sbi XCKn_DDR, XCKn
  ; Set MSPI mode of operation and SPI data mode 0.
  ldi r18, (1<<UMSELn1)|(1<<UMSELn0)|(0<<UCPHAn)|(0<<UCPOLn)
  out UCSRnC,r18
  ; Enable receiver and transmitter.
  ldi r18, (1<<RXENn)|(1<<TXENn)
  out UCSRnB,r18
  ; Set baud rate.
  ; IMPORTANT: The Baud Rate must be set after the transmitter is enabled!
  out UBRRnH, r17
  out UBRRnL, r18
  ret
```

C Code Example[1]

```
void USART_Init( unsigned int baud )
{
  UBRRn = 0;
  /* Setting the XCKn port pin as output, enables master mode. */
  XCKn_DDR |= (1<<XCKn);
  /* Set MSPI mode of operation and SPI data mode 0. */
  UCSRnC = (1<<UMSELn1)|(1<<UMSELn0)|(0<<UCPHAn)|(0<<UCPOLn);
  /* Enable receiver and transmitter. */
  UCSRnB = (1<<RXENn)|(1<<TXENn);
  /* Set baud rate. */
  /* IMPORTANT: The Baud Rate must be set after the transmitter is enabled
  */
  UBRRn = baud;
}
```

Note: 1. See "Code Examples" on page 7.

18.6 Data Transfer

Using the USART in MSPI mode requires the Transmitter to be enabled, i.e. the TXENn bit in the UCSRnB register is set to one. When the Transmitter is enabled, the normal port operation of the TxDn pin is overridden and given the function as the Transmitter's serial output. Enabling the receiver is optional and is done by setting the RXENn bit in the UCSRnB register to one. When the receiver is enabled, the normal pin operation of the RxDn pin is overridden and given the function as the Receiver's serial input. The XCKn will in both cases be used as the transfer clock.

After initialization the USART is ready for doing data transfers. A data transfer is initiated by writing to the UDRn I/O location. This is the case for both sending and receiving data since the transmitter controls the transfer clock. The data written to UDRn is moved from the transmit buffer to the shift register when the shift register is ready to send a new frame.

Note: To keep the input buffer in sync with the number of data bytes transmitted, the UDRn register must be read once for each byte transmitted. The input buffer operation is identical to normal USART mode, i.e. if an overflow occurs the character last received will be lost, not the first data in the buffer. This means that if four bytes are transferred, byte 1 first, then byte 2, 3, and 4, and the UDRn is not read before all transfers are completed, then byte 3 to be received will be lost, and not byte 1.

The following code examples show a simple USART in MSPIM mode transfer function based on polling of the Data Register Empty (UDREn) Flag and the Receive Complete (RXCn) Flag. The USART has to be initialized before the function can be used. For the assembly code, the data to be sent is assumed to be stored in Register R16 and the data received will be available in the same register (R16) after the function returns.

The function simply waits for the transmit buffer to be empty by checking the UDREn Flag, before loading it with new data to be transmitted. The function then waits for data to be present in the receive buffer by checking the RXCn Flag, before reading the buffer and returning the value.

Assembly Code Example[1]

```
USART_MSPIM_Transfer:
  ; Wait for empty transmit buffer
  sbis UCSRnA, UDREn
  rjmp USART_MSPIM_Transfer
  ; Put data (r16) into buffer, sends the data
  out UDRn,r16
  ; Wait for data to be received
USART_MSPIM_Wait_RXCn:
  sbis UCSRnA, RXCn
  rjmp USART_MSPIM_Wait_RXCn
  ; Get and return received data from buffer
  in r16, UDRn
  ret
```

C Code Example[1]

```
unsigned char USART_Receive( void )
{
  /* Wait for empty transmit buffer */
  while ( !( UCSRnA & (1<<UDREn)) );
  /* Put data into buffer, sends the data */
  UDRn = data;
  /* Wait for data to be received */
  while ( !(UCSRnA & (1<<RXCn)) );
  /* Get and return received data from buffer */
  return UDRn;
}
```

Note: 1. See ”Code Examples” on page 7.

18.6.1 Transmitter and Receiver Flags and Interrupts

The RXCn, TXCn, and UDREn flags and corresponding interrupts in USART in MSPIM mode are identical in function to the normal USART operation. However, the receiver error status flags (FE, DOR, and PE) are not in use and is always read as zero.

18.6.2 Disabling the Transmitter or Receiver

The disabling of the transmitter or receiver in USART in MSPIM mode is identical in function to the normal USART operation.

18.7 AVR USART MSPIM vs. AVR SPI

The USART in MSPIM mode is fully compatible with the AVR SPI regarding:

- Master mode timing diagram.
- The UCPOLn bit functionality is identical to the SPI CPOL bit.
- The UCPHAn bit functionality is identical to the SPI CPHA bit.
- The UDORDn bit functionality is identical to the SPI DORD bit.

However, since the USART in MSPIM mode reuses the USART resources, the use of the USART in MSPIM mode is somewhat different compared to the SPI. In addition to differences of the control register bits, and that only master operation is supported by the USART in MSPIM mode, the following features differ between the two modules:

- The USART in MSPIM mode includes (double) buffering of the transmitter. The SPI has no buffer.
- The USART in MSPIM mode receiver includes an additional buffer level.
- The SPI WCOL (Write Collision) bit is not included in USART in MSPIM mode.
- The SPI double speed mode (SPI2X) bit is not included. However, the same effect is achieved by setting UBRRn accordingly.
- Interrupt timing is not compatible.
- Pin control differs due to the master only operation of the USART in MSPIM mode.

A comparison of the USART in MSPIM mode and the SPI pins is shown in Table 18-3 on page 210.

Table 18-3. Comparison of USART in MSPIM mode and SPI pins.

USART_MSPIM	SPI	Comment
TxDn	MOSI	Master Out only
RxDn	MISO	Master In only
XCKn	SCK	(Functionally identical)
(N/A)	$\overline{SS}$	Not supported by USART in MSPIM

18.8 Register Description

The following section describes the registers used for SPI operation using the USART.

18.8.1 UDRn – USART MSPIM I/O Data Register

The function and bit description of the USART data register (UDRn) in MSPI mode is identical to normal USART operation. See “UDRn – USART I/O Data Register n” on page 195.

18.8.2 UCSRnA – USART MSPIM Control and Status Register n A

Bit	7	6	5	4	3	2	1	0	
	RXCn	TXCn	UDREn	-	-	-	-	-	UCSRnA
Read/Write	R	R/W	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	1	1	0	

- **Bit 7 - RXCn: USART Receive Complete**

This flag bit is set when there are unread data in the receive buffer and cleared when the receive buffer is empty (i.e., does not contain any unread data). If the Receiver is disabled, the receive buffer will be flushed and consequently the RXCn bit will become zero. The RXCn Flag can be used to generate a Receive Complete interrupt (see description of the RXCIEn bit).

- **Bit 6 - TXCn: USART Transmit Complete**

This flag bit is set when the entire frame in the Transmit Shift Register has been shifted out and there are no new data currently present in the transmit buffer (UDRn). The TXCn Flag bit is automatically cleared when a transmit complete interrupt is executed, or it can be cleared by writing a one to its bit location. The TXCn Flag can generate a Transmit Complete interrupt (see description of the TXCIEn bit).

- **Bit 5 - UDREn: USART Data Register Empty**

The UDREn Flag indicates if the transmit buffer (UDRn) is ready to receive new data. If UDREn is one, the buffer is empty, and therefore ready to be written. The UDREn Flag can generate a Data Register Empty interrupt (see description of the UDRIE bit). UDREn is set after a reset to indicate that the Transmitter is ready.

- **Bit 4:0 - Reserved Bits in MSPI mode**

When in MSPI mode, these bits are reserved for future use. For compatibility with future devices, these bits must be written to zero when UCSRnA is written.

18.8.3 UCSRnB – USART MSPIM Control and Status Register n B

Bit	7	6	5	4	3	2	1	0	
	RXCIEn	TXCIEn	UDRIE	RXENn	TXENn	-	-	-	UCSRnB
Read/Write	R/W	R/W	R/W	R/W	R/W	R	R	R	
Initial Value	0	0	0	0	0	1	1	0	

- **Bit 7 - RXCIEn: RX Complete Interrupt Enable**

Writing this bit to one enables interrupt on the RXCn Flag. A USART Receive Complete interrupt will be generated only if the RXCIEn bit is written to one, the Global Interrupt Flag in SREG is written to one and the RXCn bit in UCSRnA is set.

- **Bit 6 - TXCIEn: TX Complete Interrupt Enable**

Writing this bit to one enables interrupt on the TXCn Flag. A USART Transmit Complete interrupt will be generated only if the TXCIEn bit is written to one, the Global Interrupt Flag in SREG is written to one and the TXCn bit in UCSRnA is set.

- **Bit 5 - UDRIE: USART Data Register Empty Interrupt Enable**

Writing this bit to one enables interrupt on the UDREn Flag. A Data Register Empty interrupt will be generated only if the UDRIE bit is written to one, the Global Interrupt Flag in SREG is written to one and the UDREn bit in UCSRnA is set.

- **Bit 4 - RXENn: Receiver Enable**

Writing this bit to one enables the USART Receiver in MSPIM mode. The Receiver will override normal port operation for the RxDn pin when enabled. Disabling the Receiver will flush the receive buffer. Only enabling the receiver in MSPI mode (i.e. setting RXENn=1 and TXENn=0) has no meaning since it is the transmitter that controls the transfer clock and since only master mode is supported.

- **Bit 3 - TXENn: Transmitter Enable**

Writing this bit to one enables the USART Transmitter. The Transmitter will override normal port operation for the TxDn pin when enabled. The disabling of the Transmitter (writing TXENn to zero) will not become effective until ongoing and pending transmissions are completed, i.e., when the Transmit Shift Register and Transmit Buffer Register do not contain data to be transmitted. When disabled, the Transmitter will no longer override the TxDn port.

- **Bit 2:0 - Reserved Bits in MSPI mode**

When in MSPI mode, these bits are reserved for future use. For compatibility with future devices, these bits must be written to zero when UCSRnB is written.

18.8.4 UCSRnC – USART MSPIM Control and Status Register n C

Bit	7	6	5	4	3	2	1	0	
	UMSELn1	UMSELn0	-	-	-	UDORDn	UCPHAn	UCPOLn	UCSRnC
Read/Write	R/W	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	1	1	0	

- **Bit 7:6 - UMSELn1:0: USART Mode Select**

These bits select the mode of operation of the USART as shown in Table 18-4. See "UCSRnC – USART Control and Status Register n C" on page 197 for full description of the normal USART operation. The MSPIM is enabled when both UMSELn bits are set to one. The UDORDn, UCPHAn, and UCPOLn can be set in the same write operation where the MSPIM is enabled.

Table 18-4. UMSELn Bits Settings

UMSELn1	UMSELn0	Mode
0	0	Asynchronous USART
0	1	Synchronous USART
1	0	Reserved
1	1	Master SPI (MSPIM)

- **Bit 5:3 - Reserved Bits in MSPI mode**

When in MSPI mode, these bits are reserved for future use. For compatibility with future devices, these bits must be written to zero when UCSRnC is written.

- **Bit 2 - UDORDn: Data Order**

When set to one the LSB of the data word is transmitted first. When set to zero the MSB of the data word is transmitted first. Refer to the Frame Formats section page 4 for details.

- **Bit 1 - UCPHAn: Clock Phase**

The UCPHAn bit setting determine if data is sampled on the leasing edge (first) or tailing (last) edge of XCKn. Refer to the SPI Data Modes and Timing section page 4 for details.

- **Bit 0 - UCPOLn: Clock Polarity**

The UCPOLn bit sets the polarity of the XCKn clock. The combination of the UCPOLn and UCPHAn bit settings determine the timing of the data transfer. Refer to the SPI Data Modes and Timing section page 4 for details.

18.8.5 USART MSPIM Baud Rate Registers - UBRRnL and UBRRnH

The function and bit description of the baud rate registers in MSPI mode is identical to normal USART operation. See “UBRRnL and UBRRnH – USART Baud Rate Registers” on page 199.

19. 2-wire Serial Interface

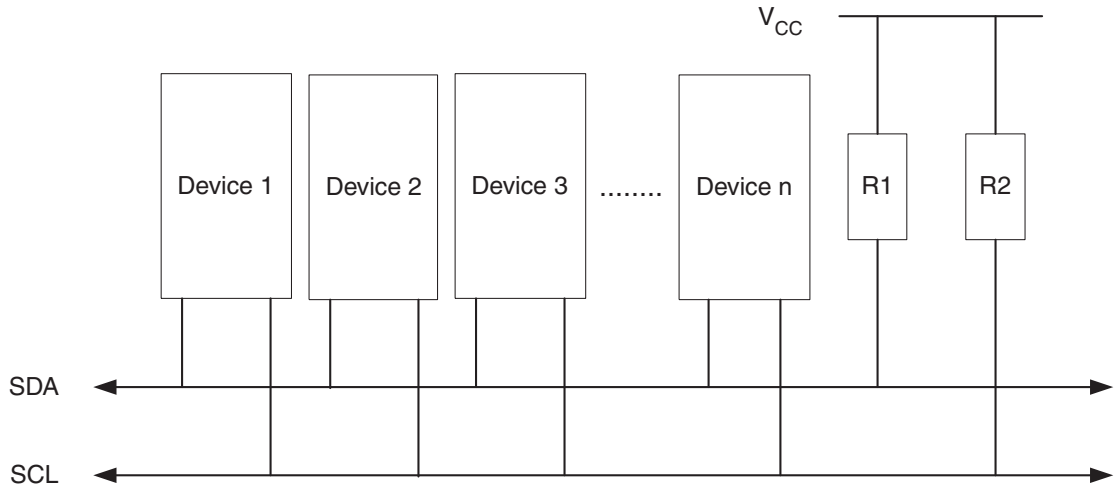
19.1 Features

- **Simple Yet Powerful and Flexible Communication Interface, only two Bus Lines Needed**
- **Both Master and Slave Operation Supported**
- **Device can Operate as Transmitter or Receiver**
- **7-bit Address Space Allows up to 128 Different Slave Addresses**
- **Multi-master Arbitration Support**
- **Up to 400 kHz Data Transfer Speed**
- **Slew-rate Limited Output Drivers**
- **Noise Suppression Circuitry Rejects Spikes on Bus Lines**
- **Fully Programmable Slave Address with General Call Support**
- **Address Recognition Causes Wake-up When AVR is in Sleep Mode**
- **Compatible with Philips' I^2C protocol**

19.2 2-wire Serial Interface Bus Definition

The 2-wire Serial Interface (TWI) is ideally suited for typical microcontroller applications. The TWI protocol allows the systems designer to interconnect up to 128 different devices using only two bi-directional bus lines, one for clock (SCL) and one for data (SDA). The only external hardware needed to implement the bus is a single pull-up resistor for each of the TWI bus lines. All devices connected to the bus have individual addresses, and mechanisms for resolving bus contention are inherent in the TWI protocol.

Figure 19-1. TWI Bus Interconnection



19.2.1 TWI Terminology

The following definitions are frequently encountered in this section.

Table 19-1. TWI Terminology

Term	Description
Master	The device that initiates and terminates a transmission. The Master also generates the SCL clock.
Slave	The device addressed by a Master.
Transmitter	The device placing data on the bus.
Receiver	The device reading data from the bus.

The PRTWI bit in ”Minimizing Power Consumption” on page 42 must be written to zero to enable the 2-wire Serial Interface.

19.2.2 Electrical Interconnection

As depicted in Figure 19-1, both bus lines are connected to the positive supply voltage through pull-up resistors. The bus drivers of all TWI-compliant devices are open-drain or open-collector. This implements a wired-AND function which is essential to the operation of the interface. A low level on a TWI bus line is generated when one or more TWI devices output a zero. A high level is output when all TWI devices tri-state their outputs, allowing the pull-up resistors to pull the line high. Note that all AVR devices connected to the TWI bus must be powered in order to allow any bus operation.

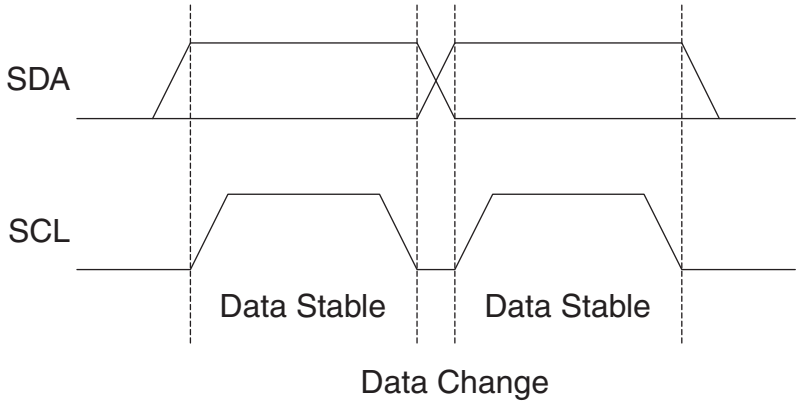
The number of devices that can be connected to the bus is only limited by the bus capacitance limit of 400 pF and the 7-bit slave address space. A detailed specification of the electrical characteristics of the TWI is given in ”2-wire Serial Interface Characteristics” on page 323. Two different sets of specifications are presented there, one relevant for bus speeds below 100 kHz, and one valid for bus speeds up to 400 kHz.

19.3 Data Transfer and Frame Format

19.3.1 Transferring Bits

Each data bit transferred on the TWI bus is accompanied by a pulse on the clock line. The level of the data line must be stable when the clock line is high. The only exception to this rule is for generating start and stop conditions.

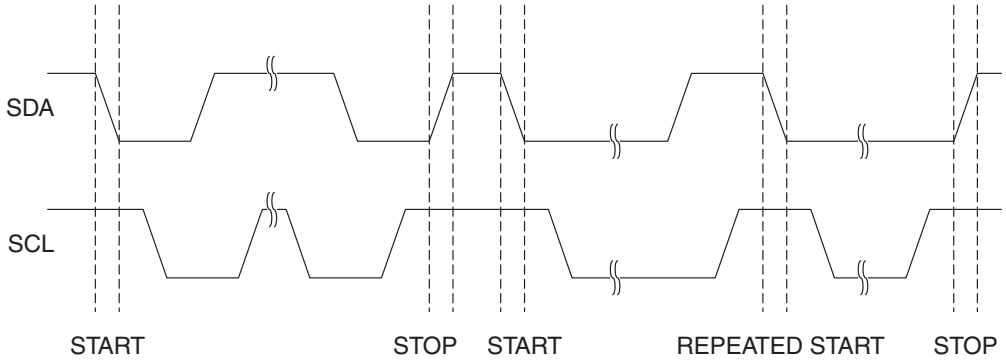
Figure 19-2. Data Validity



19.3.2 START and STOP Conditions

The Master initiates and terminates a data transmission. The transmission is initiated when the Master issues a START condition on the bus, and it is terminated when the Master issues a STOP condition. Between a START and a STOP condition, the bus is considered busy, and no other master should try to seize control of the bus. A special case occurs when a new START condition is issued between a START and STOP condition. This is referred to as a REPEATED START condition, and is used when the Master wishes to initiate a new transfer without relinquishing control of the bus. After a REPEATED START, the bus is considered busy until the next STOP. This is identical to the START behavior, and therefore START is used to describe both START and REPEATED START for the remainder of this datasheet, unless otherwise noted. As depicted below, START and STOP conditions are signalled by changing the level of the SDA line when the SCL line is high.

Figure 19-3. START, REPEATED START and STOP conditions



19.3.3 Address Packet Format

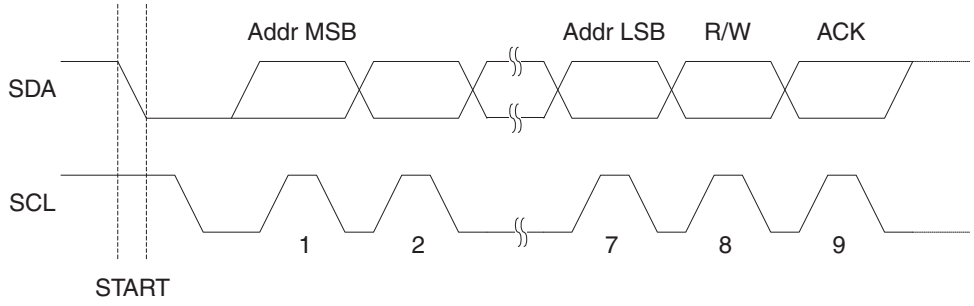
All address packets transmitted on the TWI bus are 9 bits long, consisting of 7 address bits, one READ/WRITE control bit and an acknowledge bit. If the READ/WRITE bit is set, a read operation is to be performed, otherwise a write operation should be performed. When a Slave recognizes that it is being addressed, it should acknowledge by pulling SDA low in the ninth SCL (ACK) cycle. If the addressed Slave is busy, or for some other reason can not service the Master's request, the SDA line should be left high in the ACK clock cycle. The Master can then transmit a STOP condition, or a REPEATED START condition to initiate a new transmission. An address packet consisting of a slave address and a READ or a WRITE bit is called SLA+R or SLA+W, respectively.

The MSB of the address byte is transmitted first. Slave addresses can freely be allocated by the designer, but the address 0000 000 is reserved for a general call.

When a general call is issued, all slaves should respond by pulling the SDA line low in the ACK cycle. A general call is used when a Master wishes to transmit the same message to several slaves in the system. When the general call address followed by a Write bit is transmitted on the bus, all slaves set up to acknowledge the general call will pull the SDA line low in the ack cycle. The following data packets will then be received by all the slaves that acknowledged the general call. Note that transmitting the general call address followed by a Read bit is meaningless, as this would cause contention if several slaves started transmitting different data.

All addresses of the format 1111 xxx should be reserved for future purposes.

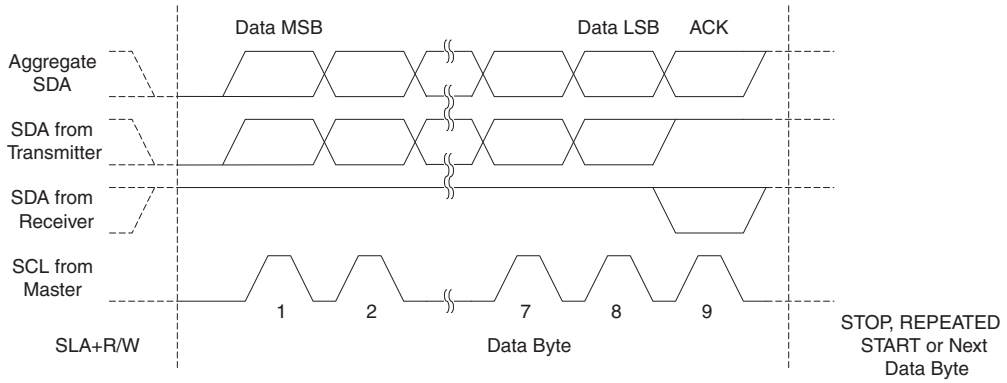
Figure 19-4. Address Packet Format



19.3.4 Data Packet Format

All data packets transmitted on the TWI bus are nine bits long, consisting of one data byte and an acknowledge bit. During a data transfer, the Master generates the clock and the START and STOP conditions, while the Receiver is responsible for acknowledging the reception. An Acknowledge (ACK) is signalled by the Receiver pulling the SDA line low during the ninth SCL cycle. If the Receiver leaves the SDA line high, a NACK is signalled. When the Receiver has received the last byte, or for some reason cannot receive any more bytes, it should inform the Transmitter by sending a NACK after the final byte. The MSB of the data byte is transmitted first.

Figure 19-5. Data Packet Format

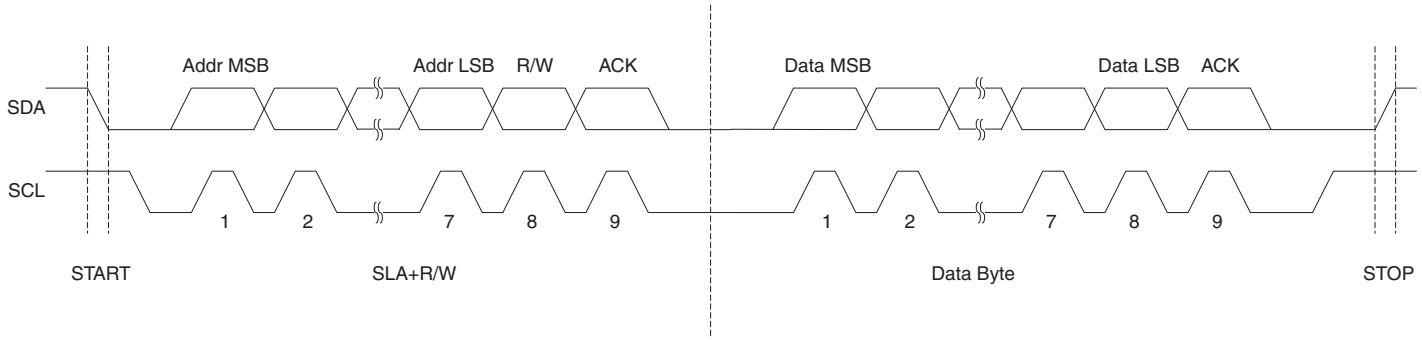


19.3.5 Combining Address and Data Packets into a Transmission

A transmission basically consists of a START condition, a SLA+R/W, one or more data packets and a STOP condition. An empty message, consisting of a START followed by a STOP condition, is illegal. Note that the Wired-ANDing of the SCL line can be used to implement handshaking between the Master and the Slave. The Slave can extend the SCL low period by pulling the SCL line low. This is useful if the clock speed set up by the Master is too fast for the Slave, or the Slave needs extra time for processing between the data transmissions. The Slave extending the SCL low period will not affect the SCL high period, which is determined by the Master. As a consequence, the Slave can reduce the TWI data transfer speed by prolonging the SCL duty cycle.

Figure 19-6 shows a typical data transmission. Note that several data bytes can be transmitted between the SLA+R/W and the STOP condition, depending on the software protocol implemented by the application software.

Figure 19-6. Typical Data Transmission



19.4 Multi-master Bus Systems, Arbitration and Synchronization

The TWI protocol allows bus systems with several masters. Special concerns have been taken in order to ensure that transmissions will proceed as normal, even if two or more masters initiate a transmission at the same time. Two problems arise in multi-master systems:

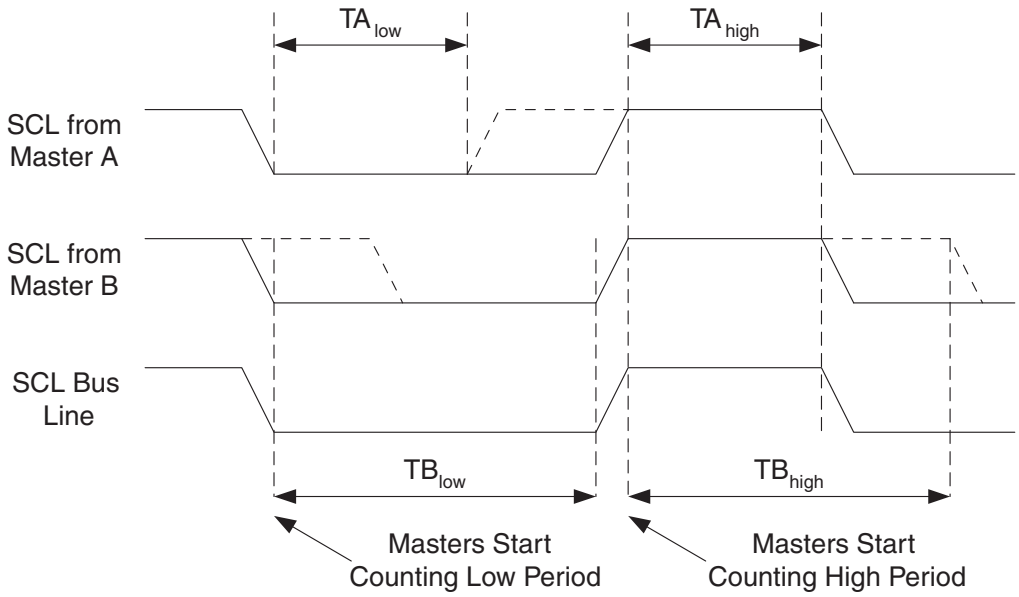
- An algorithm must be implemented allowing only one of the masters to complete the transmission. All other masters should cease transmission when they discover that they have lost the selection process. This selection process is called arbitration. When a contending master discovers that it has lost the arbitration process, it should immediately switch to Slave mode to check whether it is being addressed by the winning master. The fact that multiple

masters have started transmission at the same time should not be detectable to the slaves, i.e. the data being transferred on the bus must not be corrupted.

- Different masters may use different SCL frequencies. A scheme must be devised to synchronize the serial clocks from all masters, in order to let the transmission proceed in a lockstep fashion. This will facilitate the arbitration process.

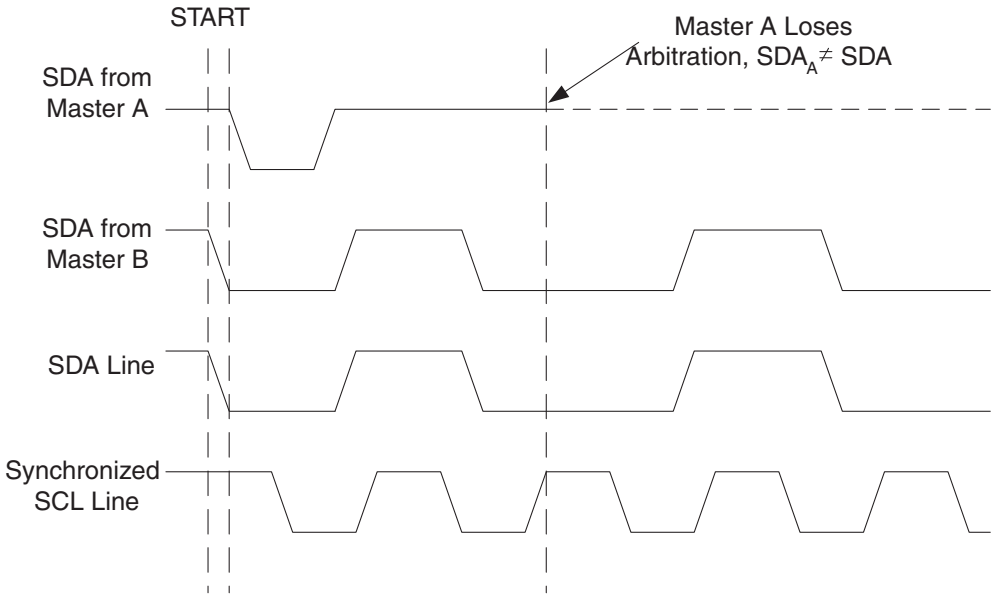
The wired-ANDing of the bus lines is used to solve both these problems. The serial clocks from all masters will be wired-ANDed, yielding a combined clock with a high period equal to the one from the Master with the shortest high period. The low period of the combined clock is equal to the low period of the Master with the longest low period. Note that all masters listen to the SCL line, effectively starting to count their SCL high and low time-out periods when the combined SCL line goes high or low, respectively.

Figure 19-7. SCL Synchronization Between Multiple Masters



Arbitration is carried out by all masters continuously monitoring the SDA line after outputting data. If the value read from the SDA line does not match the value the Master had output, it has lost the arbitration. Note that a Master can only lose arbitration when it outputs a high SDA value while another Master outputs a low value. The losing Master should immediately go to Slave mode, checking if it is being addressed by the winning Master. The SDA line should be left high, but losing masters are allowed to generate a clock signal until the end of the current data or address packet. Arbitration will continue until only one Master remains, and this may take many bits. If several masters are trying to address the same Slave, arbitration will continue into the data packet.

Figure 19-8. Arbitration Between Two Masters



Note that arbitration is not allowed between:

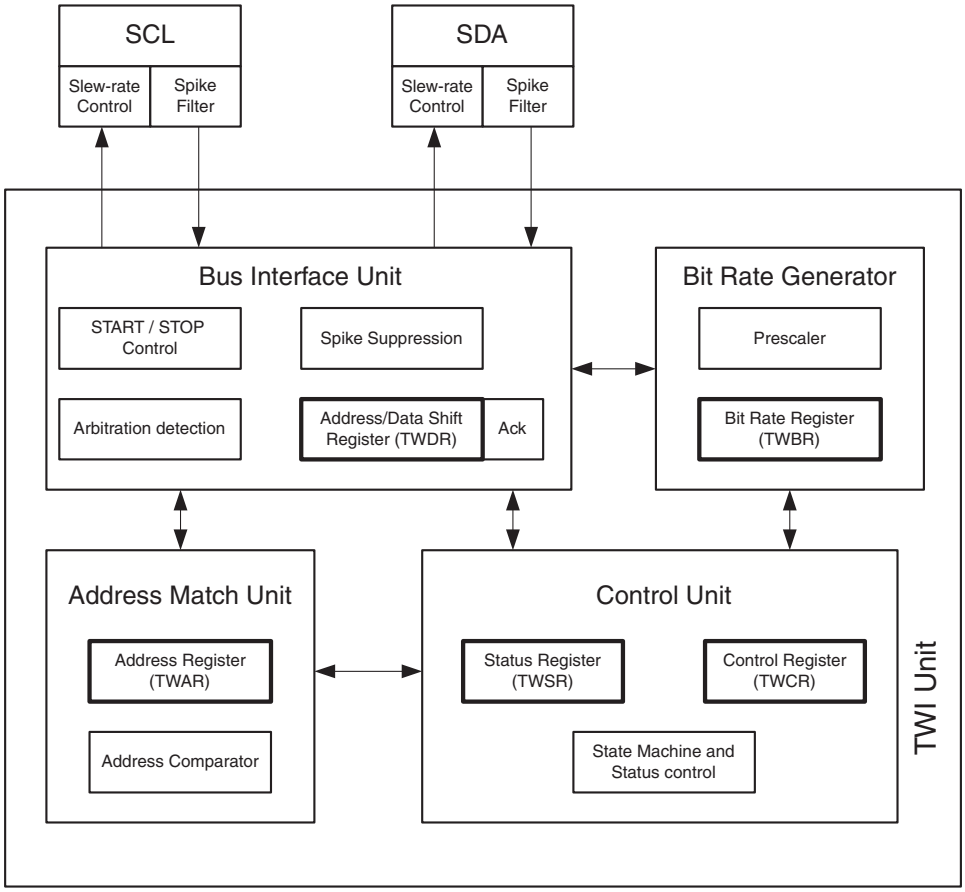
- A REPEATED START condition and a data bit.
- A STOP condition and a data bit.
- A REPEATED START and a STOP condition.

It is the user software's responsibility to ensure that these illegal arbitration conditions never occur. This implies that in multi-master systems, all data transfers must use the same composition of SLA+R/W and data packets. In other words: All transmissions must contain the same number of data packets, otherwise the result of the arbitration is undefined.

19.5 Overview of the TWI Module

The TWI module is comprised of several submodules, as shown in Figure 19-9. All registers drawn in a thick line are accessible through the AVR data bus.

Figure 19-9. Overview of the TWI Module



19.5.1 SCL and SDA Pins

These pins interface the AVR TWI with the rest of the MCU system. The output drivers contain a slew-rate limiter in order to conform to the TWI specification. The input stages contain a spike suppression unit removing spikes shorter than 50 ns. Note that the internal pull-ups in the AVR pads can be enabled by setting the PORT bits corresponding to the SCL and SDA pins, as explained in the I/O Port section. The internal pull-ups can in some systems eliminate the need for external ones.

19.5.2 Bit Rate Generator Unit

This unit controls the period of SCL when operating in a Master mode. The SCL period is controlled by settings in the TWI Bit Rate Register (TWBR) and the Prescaler bits in the TWI Status Register (TWSR). Slave operation does not depend on Bit Rate or Prescaler settings, but the CPU clock frequency in the Slave must be at least 16 times higher than the SCL frequency. Note

that slaves may prolong the SCL low period, thereby reducing the average TWI bus clock period. The SCL frequency is generated according to the following equation:

$$\text{SCL frequency} = \frac{\text{CPU Clock frequency}}{16 + 2(\text{TWBR}) \cdot (PrescalerValue)}$$

- TWBR = Value of the TWI Bit Rate Register.
- *PrescalerValue* = Value of the prescaler, see Table 19-7 on page 243.

Note: Pull-up resistor values should be selected according to the SCL frequency and the capacitive bus line load. See Table 26-6 on page 323 for value of pull-up resistor.

19.5.3 Bus Interface Unit

This unit contains the Data and Address Shift Register (TWDR), a START/STOP Controller and Arbitration detection hardware. The TWDR contains the address or data bytes to be transmitted, or the address or data bytes received. In addition to the 8-bit TWDR, the Bus Interface Unit also contains a register containing the (N)ACK bit to be transmitted or received. This (N)ACK Register is not directly accessible by the application software. However, when receiving, it can be set or cleared by manipulating the TWI Control Register (TWCR). When in Transmitter mode, the value of the received (N)ACK bit can be determined by the value in the TWSR.

The START/STOP Controller is responsible for generation and detection of START, REPEATED START, and STOP conditions. The START/STOP controller is able to detect START and STOP conditions even when the AVR MCU is in one of the sleep modes, enabling the MCU to wake up if addressed by a Master.

If the TWI has initiated a transmission as Master, the Arbitration Detection hardware continuously monitors the transmission trying to determine if arbitration is in process. If the TWI has lost an arbitration, the Control Unit is informed. Correct action can then be taken and appropriate status codes generated.

19.5.4 Address Match Unit

The Address Match unit checks if received address bytes match the seven-bit address in the TWI Address Register (TWAR). If the TWI General Call Recognition Enable (TWGCE) bit in the TWAR is written to one, all incoming address bits will also be compared against the General Call address. Upon an address match, the Control Unit is informed, allowing correct action to be taken. The TWI may or may not acknowledge its address, depending on settings in the TWCR. The Address Match unit is able to compare addresses even when the AVR MCU is in sleep mode, enabling the MCU to wake up if addressed by a Master.

If another interrupt (e.g., INT0) occurs during TWI Power-down address match and wakes up the CPU, the TWI aborts operation and return to it's idle state. If this cause any problems, ensure that TWI Address Match is the only enabled interrupt when entering Power-down[(1)].

Note: 1. This applies to all device revisions except ATmega88P revision C or newer.

19.5.5 Control Unit

The Control unit monitors the TWI bus and generates responses corresponding to settings in the TWI Control Register (TWCR). When an event requiring the attention of the application occurs on the TWI bus, the TWI Interrupt Flag (TWINT) is asserted. In the next clock cycle, the TWI Status Register (TWSR) is updated with a status code identifying the event. The TWSR only contains relevant status information when the TWI Interrupt Flag is asserted. At all other times, the TWSR contains a special status code indicating that no relevant status information is avail-

able. As long as the TWINT Flag is set, the SCL line is held low. This allows the application software to complete its tasks before allowing the TWI transmission to continue.

The TWINT Flag is set in the following situations:

- After the TWI has transmitted a START/REPEATED START condition.
- After the TWI has transmitted SLA+R/W.
- After the TWI has transmitted an address byte.
- After the TWI has lost arbitration.
- After the TWI has been addressed by own slave address or general call.
- After the TWI has received a data byte.
- After a STOP or REPEATED START has been received while still addressed as a Slave.
- When a bus error has occurred due to an illegal START or STOP condition.

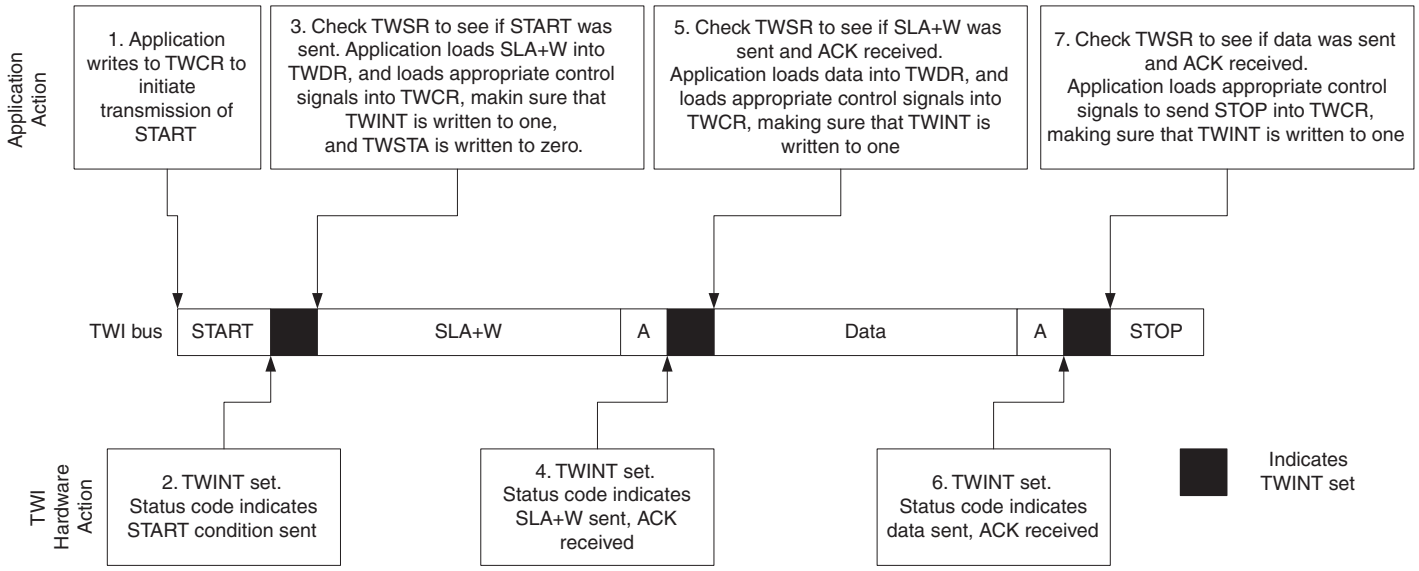
19.6 Using the TWI

The AVR TWI is byte-oriented and interrupt based. Interrupts are issued after all bus events, like reception of a byte or transmission of a START condition. Because the TWI is interrupt-based, the application software is free to carry on other operations during a TWI byte transfer. Note that the TWI Interrupt Enable (TWIE) bit in TWCR together with the Global Interrupt Enable bit in SREG allow the application to decide whether or not assertion of the TWINT Flag should generate an interrupt request. If the TWIE bit is cleared, the application must poll the TWINT Flag in order to detect actions on the TWI bus.

When the TWINT Flag is asserted, the TWI has finished an operation and awaits application response. In this case, the TWI Status Register (TWSR) contains a value indicating the current state of the TWI bus. The application software can then decide how the TWI should behave in the next TWI bus cycle by manipulating the TWCR and TWDR Registers.

Figure 19-10 is a simple example of how the application can interface to the TWI hardware. In this example, a Master wishes to transmit a single data byte to a Slave. This description is quite abstract, a more detailed explanation follows later in this section. A simple code example implementing the desired behavior is also presented.

Figure 19-10. Interfacing the Application to the TWI in a Typical Transmission



1. The first step in a TWI transmission is to transmit a START condition. This is done by writing a specific value into TWCR, instructing the TWI hardware to transmit a START condition. Which value to write is described later on. However, it is important that the TWINT bit is set in the value written. Writing a one to TWINT clears the flag. The TWI will not start any operation as long as the TWINT bit in TWCR is set. Immediately after the application has cleared TWINT, the TWI will initiate transmission of the START condition.
2. When the START condition has been transmitted, the TWINT Flag in TWCR is set, and TWSR is updated with a status code indicating that the START condition has successfully been sent.
3. The application software should now examine the value of TWSR, to make sure that the START condition was successfully transmitted. If TWSR indicates otherwise, the application software might take some special action, like calling an error routine. Assuming that the status code is as expected, the application must load SLA+W into TWDR. Remember that TWDR is used both for address and data. After TWDR has been loaded with the desired SLA+W, a specific value must be written to TWCR, instructing the TWI hardware to transmit the SLA+W present in TWDR. Which value to write is described later on. However, it is important that the TWINT bit is set in the value written. Writing a one to TWINT clears the flag. The TWI will not start any operation as long as the TWINT bit in TWCR is set. Immediately after the application has cleared TWINT, the TWI will initiate transmission of the address packet.
4. When the address packet has been transmitted, the TWINT Flag in TWCR is set, and TWSR is updated with a status code indicating that the address packet has successfully been sent. The status code will also reflect whether a Slave acknowledged the packet or not.
5. The application software should now examine the value of TWSR, to make sure that the address packet was successfully transmitted, and that the value of the ACK bit was as expected. If TWSR indicates otherwise, the application software might take some special action, like calling an error routine. Assuming that the status code is as expected, the application must load a data packet into TWDR. Subsequently, a specific value must be written to TWCR, instructing the TWI hardware to transmit the data packet present in TWDR. Which value to write is described later on. However, it is important that the TWINT bit is set in the value written. Writing a one to TWINT clears the flag. The TWI will

not start any operation as long as the TWINT bit in TWCR is set. Immediately after the application has cleared TWINT, the TWI will initiate transmission of the data packet.

6. When the data packet has been transmitted, the TWINT Flag in TWCR is set, and TWSR is updated with a status code indicating that the data packet has successfully been sent. The status code will also reflect whether a Slave acknowledged the packet or not.
7. The application software should now examine the value of TWSR, to make sure that the data packet was successfully transmitted, and that the value of the ACK bit was as expected. If TWSR indicates otherwise, the application software might take some special action, like calling an error routine. Assuming that the status code is as expected, the application must write a specific value to TWCR, instructing the TWI hardware to transmit a STOP condition. Which value to write is described later on. However, it is important that the TWINT bit is set in the value written. Writing a one to TWINT clears the flag. The TWI will not start any operation as long as the TWINT bit in TWCR is set. Immediately after the application has cleared TWINT, the TWI will initiate transmission of the STOP condition. Note that TWINT is NOT set after a STOP condition has been sent.

Even though this example is simple, it shows the principles involved in all TWI transmissions. These can be summarized as follows:

- When the TWI has finished an operation and expects application response, the TWINT Flag is set. The SCL line is pulled low until TWINT is cleared.
- When the TWINT Flag is set, the user must update all TWI Registers with the value relevant for the next TWI bus cycle. As an example, TWDR must be loaded with the value to be transmitted in the next bus cycle.
- After all TWI Register updates and other pending application software tasks have been completed, TWCR is written. When writing TWCR, the TWINT bit should be set. Writing a one to TWINT clears the flag. The TWI will then commence executing whatever operation was specified by the TWCR setting.

In the following an assembly and C implementation of the example is given. Note that the code below assumes that several definitions have been made, for example by using include-files.

	Assembly Code Example	C Example	Comments
1	`ldi r16, (1<<TWINT)|(1<<TWSTA)| (1<<TWEN)` `out TWCR, r16`	`TWCR = (1<<TWINT)|(1<<TWSTA)| (1<<TWEN)`	Send START condition

2	`wait1:`
	`in r16,TWCR`
	`sbrs r16,TWINT`
	`rjmp wait1`
		`while (!(TWCR & (1<<TWINT))) ;`	Wait for TWINT Flag set. This indicates that the START condition has been transmitted

3	`in r16,TWSR`
	`andi r16, 0xF8`
	`cpi r16, START`
	`brne ERROR`
		`if ((TWSR & 0xF8) != START) ERROR();`	Check value of TWI Status Register. Mask prescaler bits. If status different from START go to ERROR

	`ldi r16, SLA_W`
	`out TWDR, r16`
	`ldi r16, (1<<TWINT) | (1<<TWEN)`
	`out TWCR, r16`
		`TWDR = SLA_W;`
		`TWCR = (1<<TWINT) | (1<<TWEN);`
			Load SLA_W into TWDR Register. Clear TWINT bit in TWCR to start transmission of address

4	`wait2:`
	`in r16,TWCR`
	`sbrs r16,TWINT`
	`rjmp wait2`
		`while (!(TWCR & (1<<TWINT))) ;`	Wait for TWINT Flag set. This indicates that the SLA+W has been transmitted, and ACK/NACK has been received.

5	`in r16,TWSR`
	`andi r16, 0xF8`
	`cpi r16, MT_SLA_ACK`
	`brne ERROR`
		`if ((TWSR & 0xF8) != MT_SLA_ACK) ERROR();`	Check value of TWI Status Register. Mask prescaler bits. If status different from MT_SLA_ACK go to ERROR

	`ldi r16, DATA`
	`out TWDR, r16`
	`ldi r16, (1<<TWINT) | (1<<TWEN)`
	`out TWCR, r16`
		`TWDR = DATA;`
		`TWCR = (1<<TWINT) | (1<<TWEN);`
			Load DATA into TWDR Register. Clear TWINT bit in TWCR to start transmission of data

6	`wait3:`
	`in r16,TWCR`
	`sbrs r16,TWINT`
	`rjmp wait3`
		`while (!(TWCR & (1<<TWINT))) ;`	Wait for TWINT Flag set. This indicates that the DATA has been transmitted, and ACK/NACK has been received.

7	`in r16,TWSR`
	`andi r16, 0xF8`
	`cpi r16, MT_DATA_ACK`
	`brne ERROR`
		`if ((TWSR & 0xF8) != MT_DATA_ACK) ERROR();`	Check value of TWI Status Register. Mask prescaler bits. If status different from MT_DATA_ACK go to ERROR

	`ldi r16, (1<<TWINT)|(1<<TWEN)| (1<<TWSTO)` `out TWCR, r16`	`TWCR = (1<<TWINT)|(1<<TWEN)| (1<<TWSTO);`	Transmit STOP condition

19.7 Transmission Modes

The TWI can operate in one of four major modes. These are named Master Transmitter (MT), Master Receiver (MR), Slave Transmitter (ST) and Slave Receiver (SR). Several of these modes can be used in the same application. As an example, the TWI can use MT mode to write data into a TWI EEPROM, MR mode to read the data back from the EEPROM. If other masters are present in the system, some of these might transmit data to the TWI, and then SR mode would be used. It is the application software that decides which modes are legal.

The following sections describe each of these modes. Possible status codes are described along with figures detailing data transmission in each of the modes. These figures contain the following abbreviations:

S: START condition

Rs: REPEATED START condition

R: Read bit (high level at SDA)

W: Write bit (low level at SDA)

A: Acknowledge bit (low level at SDA)

$\overline{A}$: Not acknowledge bit (high level at SDA)

Data: 8-bit data byte

P: STOP condition

SLA: Slave Address

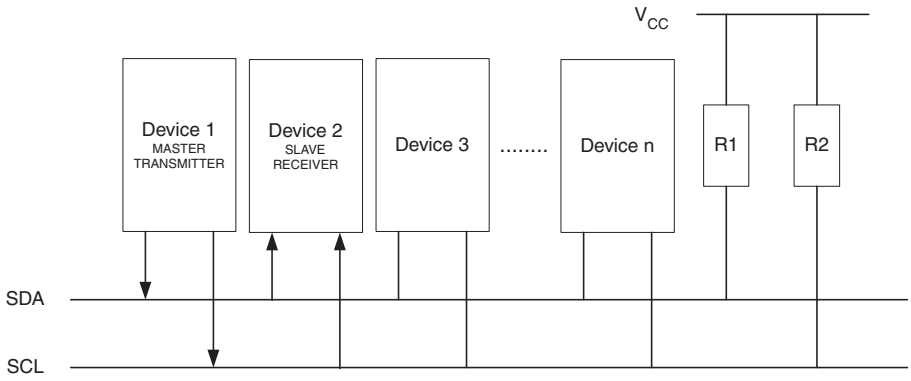
In Figure 19-12 to Figure 19-18, circles are used to indicate that the TWINT Flag is set. The numbers in the circles show the status code held in TWSR, with the prescaler bits masked to zero. At these points, actions must be taken by the application to continue or complete the TWI transfer. The TWI transfer is suspended until the TWINT Flag is cleared by software.

When the TWINT Flag is set, the status code in TWSR is used to determine the appropriate software action. For each status code, the required software action and details of the following serial transfer are given in Table 19-2 to Table 19-5. Note that the prescaler bits are masked to zero in these tables.

19.7.1 Master Transmitter Mode

In the Master Transmitter mode, a number of data bytes are transmitted to a Slave Receiver (see Figure 19-11). In order to enter a Master mode, a START condition must be transmitted. The format of the following address packet determines whether Master Transmitter or Master Receiver mode is to be entered. If SLA+W is transmitted, MT mode is entered, if SLA+R is transmitted, MR mode is entered. All the status codes mentioned in this section assume that the prescaler bits are zero or are masked to zero.

Figure 19-11. Data Transfer in Master Transmitter Mode



A START condition is sent by writing the following value to TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	1	0	X	1	0	X

TWEN must be set to enable the 2-wire Serial Interface, TWSTA must be written to one to transmit a START condition and TWINT must be written to one to clear the TWINT Flag. The TWI will then test the 2-wire Serial Bus and generate a START condition as soon as the bus becomes free. After a START condition has been transmitted, the TWINT Flag is set by hardware, and the status code in TWSR will be 0x08 (see Table 19-2). In order to enter MT mode, SLA+W must be transmitted. This is done by writing SLA+W to TWDR. Thereafter the TWINT bit should be cleared (by writing it to one) to continue the transfer. This is accomplished by writing the following value to TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	0	0	X	1	0	X

When SLA+W have been transmitted and an acknowledgement bit has been received, TWINT is set again and a number of status codes in TWSR are possible. Possible status codes in Master mode are 0x18, 0x20, or 0x38. The appropriate action to be taken for each of these status codes is detailed in Table 19-2.

When SLA+W has been successfully transmitted, a data packet should be transmitted. This is done by writing the data byte to TWDR. TWDR must only be written when TWINT is high. If not, the access will be discarded, and the Write Collision bit (TWWC) will be set in the TWCR Register. After updating TWDR, the TWINT bit should be cleared (by writing it to one) to continue the transfer. This is accomplished by writing the following value to TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	0	0	X	1	0	X

This scheme is repeated until the last byte has been sent and the transfer is ended by generating a STOP condition or a repeated START condition. A STOP condition is generated by writing the following value to TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	0	1	X	1	0	X

A REPEATED START condition is generated by writing the following value to TWCR:

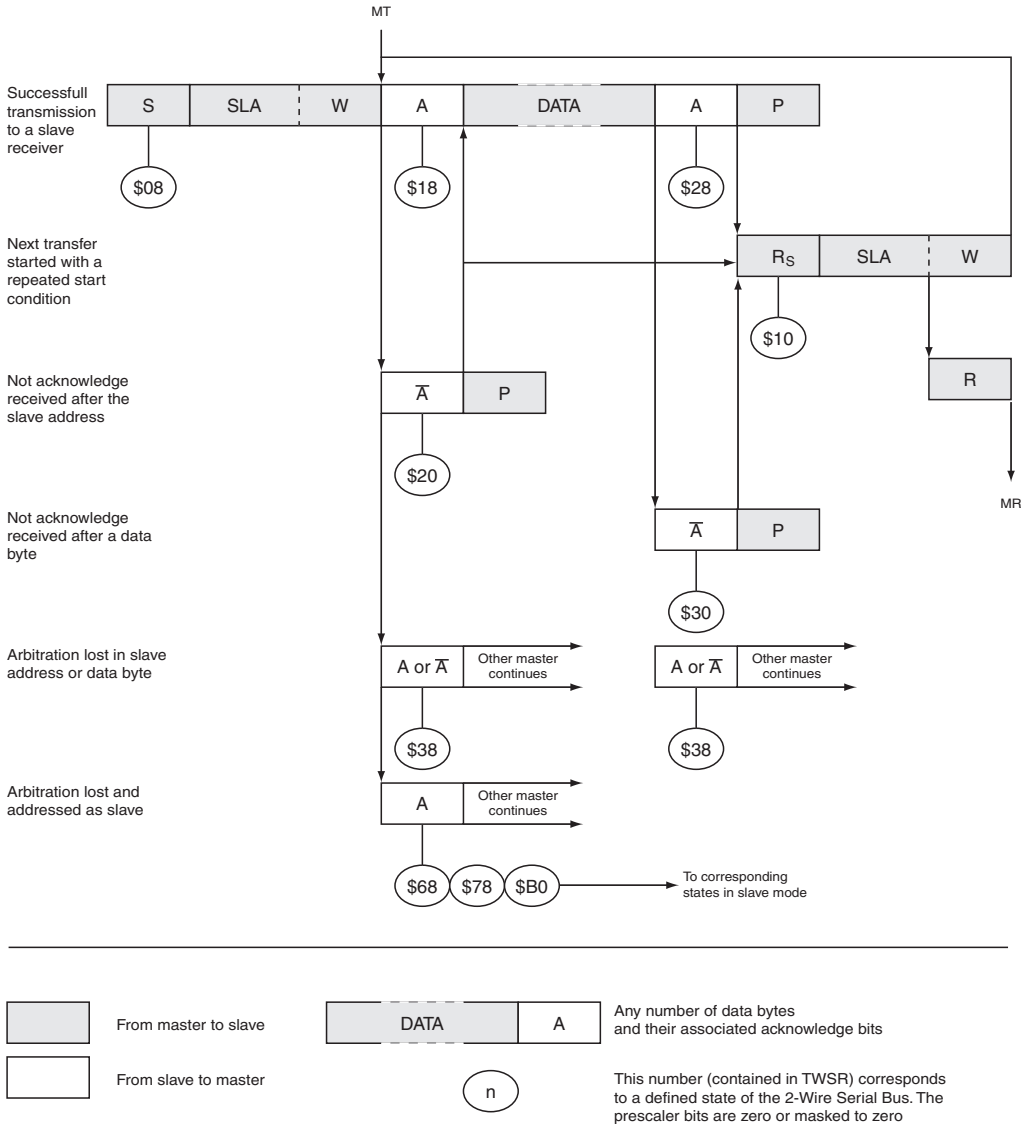
TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	1	0	X	1	0	X

After a repeated START condition (state 0x10) the 2-wire Serial Interface can access the same Slave again, or a new Slave without transmitting a STOP condition. Repeated START enables the Master to switch between Slaves, Master Transmitter mode and Master Receiver mode without losing control of the bus.

Table 19-2. Status codes for Master Transmitter Mode

Status Code (TWSR) Prescaler Bits are 0	Status of the 2-wire Serial Bus and 2-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware
		To/from TWDR	To TWCR				
			STA	STO	TWINT	TWEA	
0x08	A START condition has been transmitted	Load SLA+W	0	0	1	X	SLA+W will be transmitted; ACK or NOT ACK will be received
0x10	A repeated START condition has been transmitted	Load SLA+W or	0	0	1	X	SLA+W will be transmitted; ACK or NOT ACK will be received
		Load SLA+R	0	0	1	X	SLA+R will be transmitted; Logic will switch to Master Receiver mode
0x18	SLA+W has been transmitted; ACK has been received	Load data byte or	0	0	1	X	Data byte will be transmitted and ACK or NOT ACK will be received
		No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x20	SLA+W has been transmitted; NOT ACK has been received	Load data byte or	0	0	1	X	Data byte will be transmitted and ACK or NOT ACK will be received
		No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x28	Data byte has been transmitted; ACK has been received	Load data byte or	0	0	1	X	Data byte will be transmitted and ACK or NOT ACK will be received
		No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x30	Data byte has been transmitted; NOT ACK has been received	Load data byte or	0	0	1	X	Data byte will be transmitted and ACK or NOT ACK will be received
		No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x38	Arbitration lost in SLA+W or data bytes	No TWDR action or	0	0	1	X	2-wire Serial Bus will be released and not addressed Slave mode entered
		No TWDR action	1	0	1	X	A START condition will be transmitted when the bus becomes free

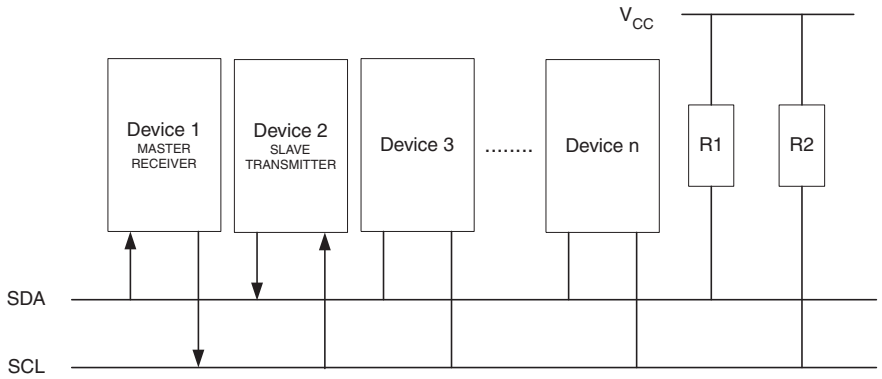
Figure 19-12. Formats and States in the Master Transmitter Mode



19.7.2 Master Receiver Mode

In the Master Receiver mode, a number of data bytes are received from a Slave Transmitter (Slave see Figure 19-13). In order to enter a Master mode, a START condition must be transmitted. The format of the following address packet determines whether Master Transmitter or Master Receiver mode is to be entered. If SLA+W is transmitted, MT mode is entered, if SLA+R is transmitted, MR mode is entered. All the status codes mentioned in this section assume that the prescaler bits are zero or are masked to zero.

Figure 19-13. Data Transfer in Master Receiver Mode



A START condition is sent by writing the following value to TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	1	0	X	1	0	X

TWEN must be written to one to enable the 2-wire Serial Interface, TWSTA must be written to one to transmit a START condition and TWINT must be set to clear the TWINT Flag. The TWI will then test the 2-wire Serial Bus and generate a START condition as soon as the bus becomes free. After a START condition has been transmitted, the TWINT Flag is set by hardware, and the status code in TWSR will be 0x08 (See Table 19-2). In order to enter MR mode, SLA+R must be transmitted. This is done by writing SLA+R to TWDR. Thereafter the TWINT bit should be cleared (by writing it to one) to continue the transfer. This is accomplished by writing the following value to TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	0	0	X	1	0	X

When SLA+R have been transmitted and an acknowledgement bit has been received, TWINT is set again and a number of status codes in TWSR are possible. Possible status codes in Master mode are 0x38, 0x40, or 0x48. The appropriate action to be taken for each of these status codes is detailed in Table 19-3. Received data can be read from the TWDR Register when the TWINT Flag is set high by hardware. This scheme is repeated until the last byte has been received. After the last byte has been received, the MR should inform the ST by sending a NACK after the last received data byte. The transfer is ended by generating a STOP condition or a repeated START condition. A STOP condition is generated by writing the following value to TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	0	1	X	1	0	X

A REPEATED START condition is generated by writing the following value to TWCR:

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	1	X	1	0	X	1	0	X

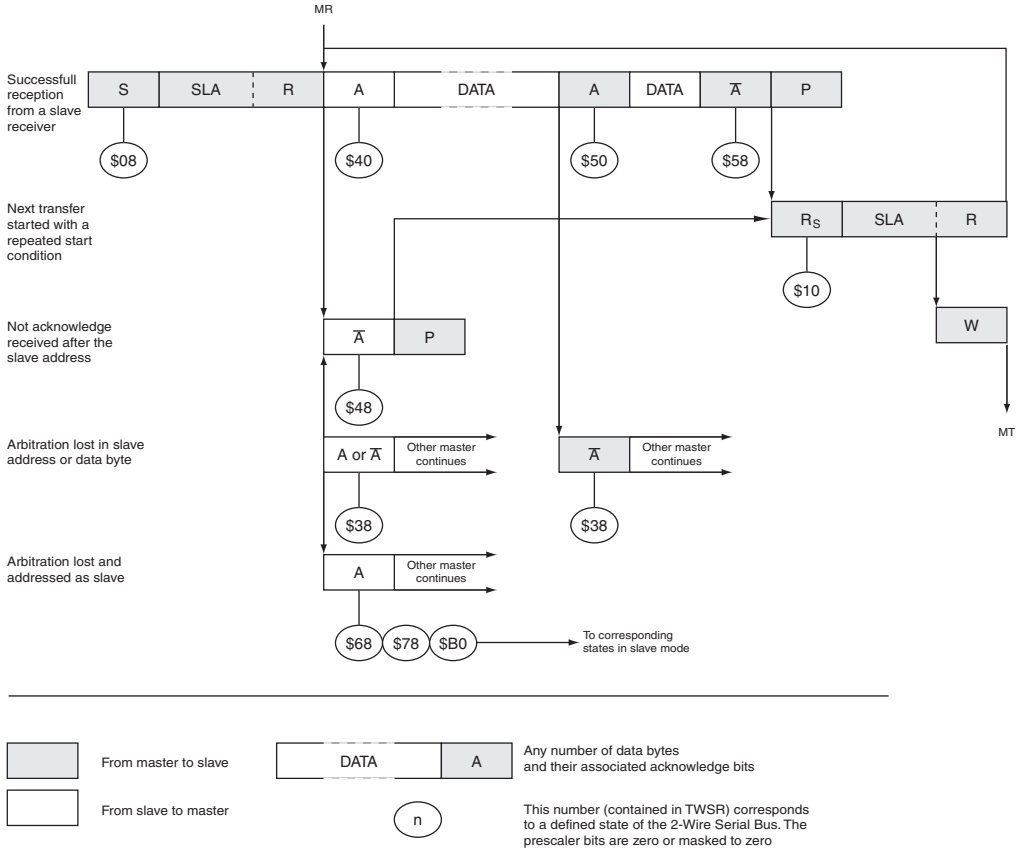
After a repeated START condition (state 0x10) the 2-wire Serial Interface can access the same Slave again, or a new Slave without transmitting a STOP condition. Repeated START enables

the Master to switch between Slaves, Master Transmitter mode and Master Receiver mode without losing control over the bus.

Table 19-3. Status codes for Master Receiver Mode

Status Code (TWSR) Prescaler Bits are 0	Status of the 2-wire Serial Bus and 2-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware
		To/from TWDR	To TWCR				
			STA	STO	TWINT	TWEA	
0x08	A START condition has been transmitted	Load SLA+R	0	0	1	X	SLA+R will be transmitted ACK or NOT ACK will be received
0x10	A repeated START condition has been transmitted	Load SLA+R or	0	0	1	X	SLA+R will be transmitted ACK or NOT ACK will be received
		Load SLA+W	0	0	1	X	SLA+W will be transmitted Logic will switch to Master Transmitter mode
0x38	Arbitration lost in SLA+R or NOT ACK bit	No TWDR action or	0	0	1	X	2-wire Serial Bus will be released and not addressed Slave mode will be entered
		No TWDR action	1	0	1	X	A START condition will be transmitted when the bus becomes free
0x40	SLA+R has been transmitted; ACK has been received	No TWDR action or	0	0	1	0	Data byte will be received and NOT ACK will be returned
		No TWDR action	0	0	1	1	Data byte will be received and ACK will be returned
0x48	SLA+R has been transmitted; NOT ACK has been received	No TWDR action or	1	0	1	X	Repeated START will be transmitted
		No TWDR action or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		No TWDR action	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset
0x50	Data byte has been received; ACK has been returned	Read data byte or	0	0	1	0	Data byte will be received and NOT ACK will be returned
		Read data byte	0	0	1	1	Data byte will be received and ACK will be returned
0x58	Data byte has been received; NOT ACK has been returned	Read data byte or	1	0	1	X	Repeated START will be transmitted
		Read data byte or	0	1	1	X	STOP condition will be transmitted and TWSTO Flag will be reset
		Read data byte	1	1	1	X	STOP condition followed by a START condition will be transmitted and TWSTO Flag will be reset

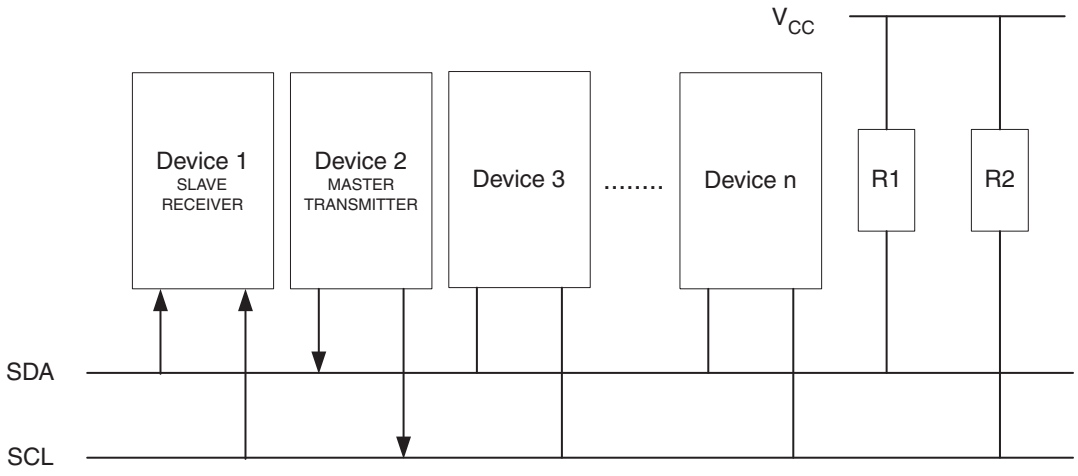
Figure 19-14. Formats and States in the Master Receiver Mode



19.7.3 Slave Receiver Mode

In the Slave Receiver mode, a number of data bytes are received from a Master Transmitter (see Figure 19-15). All the status codes mentioned in this section assume that the prescaler bits are zero or are masked to zero.

Figure 19-15. Data transfer in Slave Receiver mode



To initiate the Slave Receiver mode, TWAR and TWCR must be initialized as follows:

TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
value	Device's Own Slave Address							

The upper 7 bits are the address to which the 2-wire Serial Interface will respond when addressed by a Master. If the LSB is set, the TWI will respond to the general call address (0x00), otherwise it will ignore the general call address.

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	0	1	0	0	0	1	0	X

TWEN must be written to one to enable the TWI. The TWEA bit must be written to one to enable the acknowledgement of the device's own slave address or the general call address. TWSTA and TWSTO must be written to zero.

When TWAR and TWCR have been initialized, the TWI waits until it is addressed by its own slave address (or the general call address if enabled) followed by the data direction bit. If the direction bit is "0" (write), the TWI will operate in SR mode, otherwise ST mode is entered. After its own slave address and the write bit have been received, the TWINT Flag is set and a valid status code can be read from TWSR. The status code is used to determine the appropriate software action. The appropriate action to be taken for each status code is detailed in Table 19-4. The Slave Receiver mode may also be entered if arbitration is lost while the TWI is in the Master mode (see states 0x68 and 0x78).

If the TWEA bit is reset during a transfer, the TWI will return a "Not Acknowledge" ("1") to SDA after the next received data byte. This can be used to indicate that the Slave is not able to receive any more bytes. While TWEA is zero, the TWI does not acknowledge its own slave address. However, the 2-wire Serial Bus is still monitored and address recognition may resume at any time by setting TWEA. This implies that the TWEA bit may be used to temporarily isolate the TWI from the 2-wire Serial Bus.

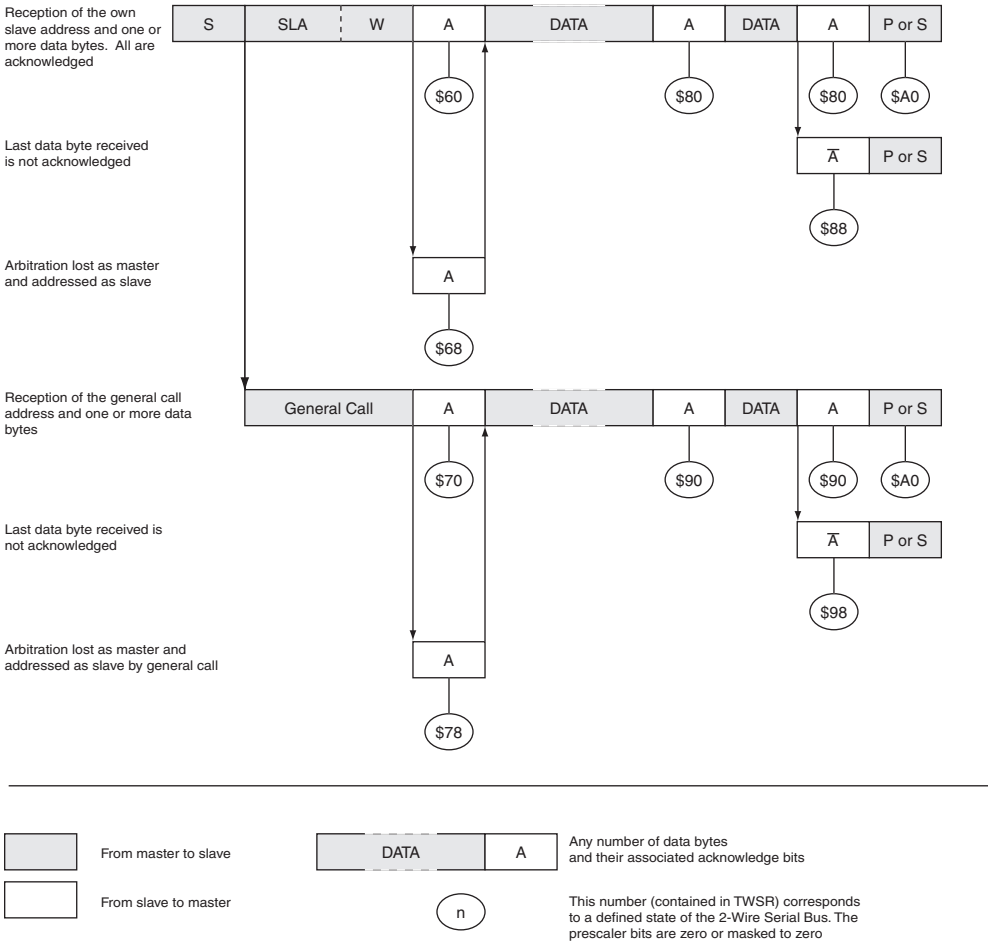
In all sleep modes other than Idle mode, the clock system to the TWI is turned off. If the TWEA bit is set, the interface can still acknowledge its own slave address or the general call address by using the 2-wire Serial Bus clock as a clock source. The part will then wake up from sleep and the TWI will hold the SCL clock low during the wake up and until the TWINT Flag is cleared (by writing it to one). Further data reception will be carried out as normal, with the AVR clocks running as normal. Observe that if the AVR is set up with a long start-up time, the SCL line may be held low for a long time, blocking other data transmissions.

Note that the 2-wire Serial Interface Data Register – TWDR does not reflect the last byte present on the bus when waking up from these Sleep modes.

Table 19-4. Status Codes for Slave Receiver Mode

Status Code (TWSR) Prescaler Bits are 0	Status of the 2-wire Serial Bus and 2-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware
		To/from TWDR	To TWCR				
			STA	STO	TWINT	TWEA	
0x60	Own SLA+W has been received; ACK has been returned	No TWDR action or	X	0	1	0	Data byte will be received and NOT ACK will be returned
		No TWDR action	X	0	1	1	Data byte will be received and ACK will be returned
0x68	Arbitration lost in SLA+R/W as Master; own SLA+W has been received; ACK has been returned	No TWDR action or	X	0	1	0	Data byte will be received and NOT ACK will be returned
		No TWDR action	X	0	1	1	Data byte will be received and ACK will be returned
0x70	General call address has been received; ACK has been returned	No TWDR action or	X	0	1	0	Data byte will be received and NOT ACK will be returned
		No TWDR action	X	0	1	1	Data byte will be received and ACK will be returned
0x78	Arbitration lost in SLA+R/W as Master; General call address has been received; ACK has been returned	No TWDR action or	X	0	1	0	Data byte will be received and NOT ACK will be returned
		No TWDR action	X	0	1	1	Data byte will be received and ACK will be returned
0x80	Previously addressed with own SLA+W; data has been received; ACK has been returned	Read data byte or	X	0	1	0	Data byte will be received and NOT ACK will be returned
		Read data byte	X	0	1	1	Data byte will be received and ACK will be returned
0x88	Previously addressed with own SLA+W; data has been received; NOT ACK has been returned	Read data byte or	0	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA
		Read data byte or	0	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”
		Read data byte or	1	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free
		Read data byte	1	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”; a START condition will be transmitted when the bus becomes free
0x90	Previously addressed with general call; data has been received; ACK has been returned	Read data byte or	X	0	1	0	Data byte will be received and NOT ACK will be returned
		Read data byte	X	0	1	1	Data byte will be received and ACK will be returned
0x98	Previously addressed with general call; data has been received; NOT ACK has been returned	Read data byte or	0	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA
		Read data byte or	0	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”
		Read data byte or	1	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free
		Read data byte	1	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”; a START condition will be transmitted when the bus becomes free
0xA0	A STOP condition or repeated START condition has been received while still addressed as Slave	No action	0	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA
			0	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”
			1	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free
			1	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”; a START condition will be transmitted when the bus becomes free

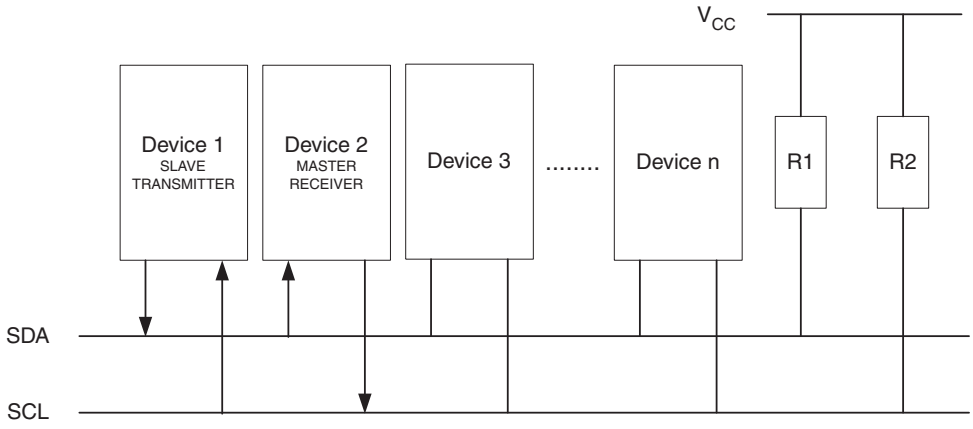
Figure 19-16. Formats and States in the Slave Receiver Mode



19.7.4 Slave Transmitter Mode

In the Slave Transmitter mode, a number of data bytes are transmitted to a Master Receiver (see Figure 19-17). All the status codes mentioned in this section assume that the prescaler bits are zero or are masked to zero.

Figure 19-17. Data Transfer in Slave Transmitter Mode



To initiate the Slave Transmitter mode, TWAR and TWCR must be initialized as follows:

TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
value	Device's Own Slave Address							

The upper seven bits are the address to which the 2-wire Serial Interface will respond when addressed by a Master. If the LSB is set, the TWI will respond to the general call address (0x00), otherwise it will ignore the general call address.

TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
value	0	1	0	0	0	1	0	X

TWEN must be written to one to enable the TWI. The TWEA bit must be written to one to enable the acknowledgement of the device's own slave address or the general call address. TWSTA and TWSTO must be written to zero.

When TWAR and TWCR have been initialized, the TWI waits until it is addressed by its own slave address (or the general call address if enabled) followed by the data direction bit. If the direction bit is "1" (read), the TWI will operate in ST mode, otherwise SR mode is entered. After its own slave address and the write bit have been received, the TWINT Flag is set and a valid status code can be read from TWSR. The status code is used to determine the appropriate software action. The appropriate action to be taken for each status code is detailed in Table 19-5. The Slave Transmitter mode may also be entered if arbitration is lost while the TWI is in the Master mode (see state 0xB0).

If the TWEA bit is written to zero during a transfer, the TWI will transmit the last byte of the transfer. State 0xC0 or state 0xC8 will be entered, depending on whether the Master Receiver transmits a NACK or ACK after the final byte. The TWI is switched to the not addressed Slave mode, and will ignore the Master if it continues the transfer. Thus the Master Receiver receives all "1" as serial data. State 0xC8 is entered if the Master demands additional data bytes (by transmitting ACK), even though the Slave has transmitted the last byte (TWEA zero and expecting NACK from the Master).

While TWEA is zero, the TWI does not respond to its own slave address. However, the 2-wire Serial Bus is still monitored and address recognition may resume at any time by setting TWEA. This implies that the TWEA bit may be used to temporarily isolate the TWI from the 2-wire Serial Bus.

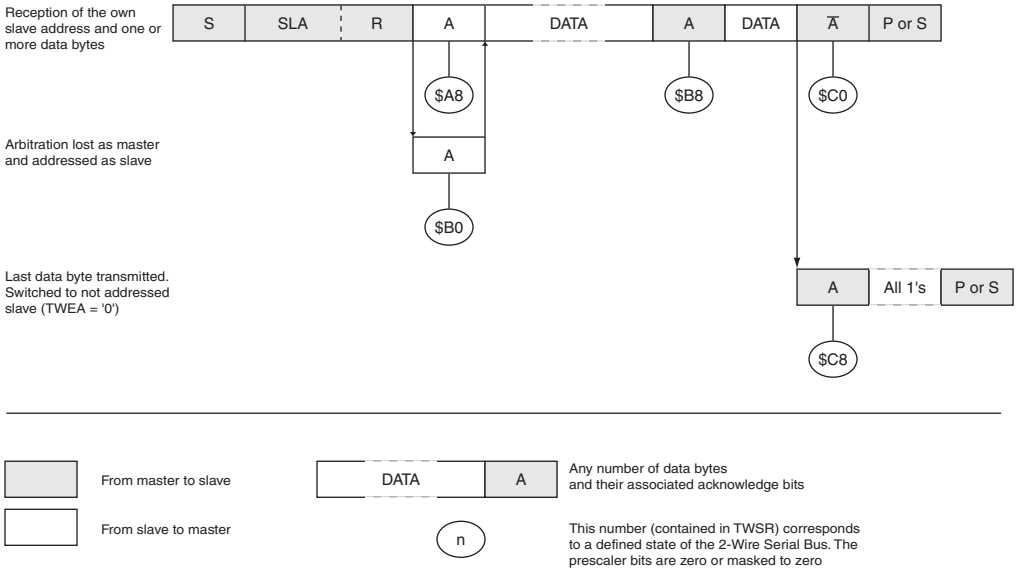
In all sleep modes other than Idle mode, the clock system to the TWI is turned off. If the TWEA bit is set, the interface can still acknowledge its own slave address or the general call address by using the 2-wire Serial Bus clock as a clock source. The part will then wake up from sleep and the TWI will hold the SCL clock will low during the wake up and until the TWINT Flag is cleared (by writing it to one). Further data transmission will be carried out as normal, with the AVR clocks running as normal. Observe that if the AVR is set up with a long start-up time, the SCL line may be held low for a long time, blocking other data transmissions.

Note that the 2-wire Serial Interface Data Register – TWDR does not reflect the last byte present on the bus when waking up from these sleep modes.

Table 19-5. Status Codes for Slave Transmitter Mode

Status Code (TWSR) Prescaler Bits are 0	Status of the 2-wire Serial Bus and 2-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware
		To/from TWDR	To TWCR				
			STA	STO	TWINT	TWEA	
0xA8	Own SLA+R has been received; ACK has been returned	Load data byte or	X	0	1	0	Last data byte will be transmitted and NOT ACK should be received
		Load data byte	X	0	1	1	Data byte will be transmitted and ACK should be received
0xB0	Arbitration lost in SLA+R/W as Master; own SLA+R has been received; ACK has been returned	Load data byte or	X	0	1	0	Last data byte will be transmitted and NOT ACK should be received
		Load data byte	X	0	1	1	Data byte will be transmitted and ACK should be received
0xB8	Data byte in TWDR has been transmitted; ACK has been received	Load data byte or	X	0	1	0	Last data byte will be transmitted and NOT ACK should be received
		Load data byte	X	0	1	1	Data byte will be transmitted and ACK should be received
0xC0	Data byte in TWDR has been transmitted; NOT ACK has been received	No TWDR action or	0	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA
		No TWDR action or	0	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”
		No TWDR action or	1	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free
		No TWDR action	1	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”; a START condition will be transmitted when the bus becomes free
0xC8	Last data byte in TWDR has been transmitted (TWEA = “0”); ACK has been received	No TWDR action or	0	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA
		No TWDR action or	0	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”
		No TWDR action or	1	0	1	0	Switched to the not addressed Slave mode; no recognition of own SLA or GCA; a START condition will be transmitted when the bus becomes free
		No TWDR action	1	0	1	1	Switched to the not addressed Slave mode; own SLA will be recognized; GCA will be recognized if TWGCE = “1”; a START condition will be transmitted when the bus becomes free

Figure 19-18. Formats and States in the Slave Transmitter Mode



19.7.5 Miscellaneous States

There are two status codes that do not correspond to a defined TWI state, see Table 19-6.

Status 0xF8 indicates that no relevant information is available because the TWINT Flag is not set. This occurs between other states, and when the TWI is not involved in a serial transfer.

Status 0x00 indicates that a bus error has occurred during a 2-wire Serial Bus transfer. A bus error occurs when a START or STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions are during the serial transfer of an address byte, a data byte, or an acknowledge bit. When a bus error occurs, TWINT is set. To recover from a bus error, the TWSTO Flag must set and TWINT must be cleared by writing a logic one to it. This causes the TWI to enter the not addressed Slave mode and to clear the TWSTO Flag (no other bits in TWCR are affected). The SDA and SCL lines are released, and no STOP condition is transmitted.

Table 19-6. Miscellaneous States

Status Code (TWSR) Prescaler Bits are 0	Status of the 2-wire Serial Bus and 2-wire Serial Interface Hardware	Application Software Response					Next Action Taken by TWI Hardware
		To/from TWDR	To TWCR				
			STA	STO	TWINT	TWEA	
0xF8	No relevant state information available; TWINT = “0”	No TWDR action	No TWCR action				Wait or proceed current transfer
0x00	Bus error due to an illegal START or STOP condition	No TWDR action	0	1	1	X	Only the internal hardware is affected, no STOP condition is sent on the bus. In all cases, the bus is released and TWSTO is cleared.

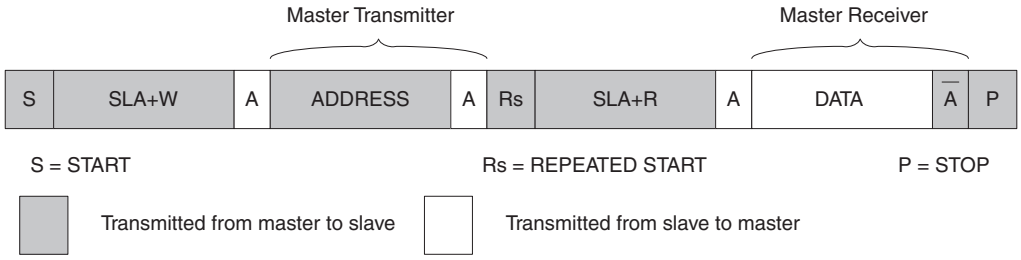
19.7.6 Combining Several TWI Modes

In some cases, several TWI modes must be combined in order to complete the desired action. Consider for example reading data from a serial EEPROM. Typically, such a transfer involves the following steps:

1. The transfer must be initiated.
2. The EEPROM must be instructed what location should be read.
3. The reading must be performed.
4. The transfer must be finished.

Note that data is transmitted both from Master to Slave and vice versa. The Master must instruct the Slave what location it wants to read, requiring the use of the MT mode. Subsequently, data must be read from the Slave, implying the use of the MR mode. Thus, the transfer direction must be changed. The Master must keep control of the bus during all these steps, and the steps should be carried out as an atomical operation. If this principle is violated in a multi master system, another Master can alter the data pointer in the EEPROM between steps 2 and 3, and the Master will read the wrong data location. Such a change in transfer direction is accomplished by transmitting a REPEATED START between the transmission of the address byte and reception of the data. After a REPEATED START, the Master keeps ownership of the bus. The following figure shows the flow in this transfer.

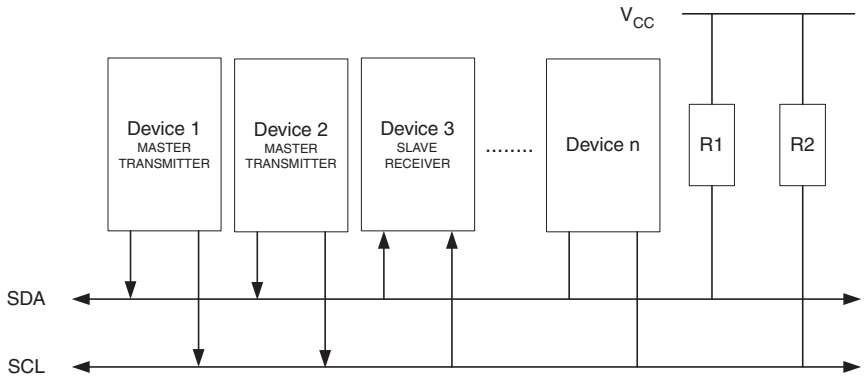
Figure 19-19. Combining Several TWI Modes to Access a Serial EEPROM



19.8 Multi-master Systems and Arbitration

If multiple masters are connected to the same bus, transmissions may be initiated simultaneously by one or more of them. The TWI standard ensures that such situations are handled in such a way that one of the masters will be allowed to proceed with the transfer, and that no data will be lost in the process. An example of an arbitration situation is depicted below, where two masters are trying to transmit data to a Slave Receiver.

Figure 19-20. An Arbitration Example



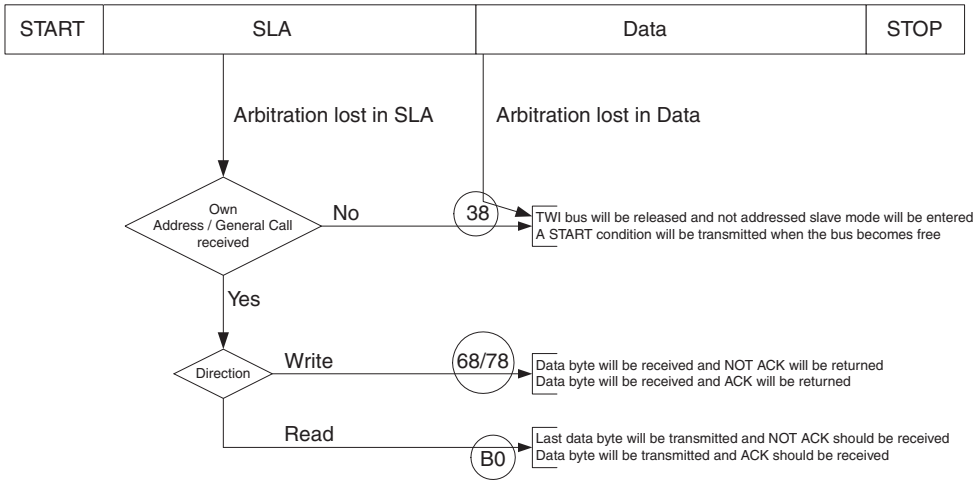
Several different scenarios may arise during arbitration, as described below:

- Two or more masters are performing identical communication with the same Slave. In this case, neither the Slave nor any of the masters will know about the bus contention.
- Two or more masters are accessing the same Slave with different data or direction bit. In this case, arbitration will occur, either in the READ/WRITE bit or in the data bits. The masters trying to output a one on SDA while another Master outputs a zero will lose the arbitration. Losing masters will switch to not addressed Slave mode or wait until the bus is free and transmit a new START condition, depending on application software action.

- Two or more masters are accessing different slaves. In this case, arbitration will occur in the SLA bits. Masters trying to output a one on SDA while another Master outputs a zero will lose the arbitration. Masters losing arbitration in SLA will switch to Slave mode to check if they are being addressed by the winning Master. If addressed, they will switch to SR or ST mode, depending on the value of the READ/WRITE bit. If they are not being addressed, they will switch to not addressed Slave mode or wait until the bus is free and transmit a new START condition, depending on application software action.

This is summarized in Figure 19-21. Possible status values are given in circles.

Figure 19-21. Possible Status Codes Caused by Arbitration



19.9 Register Description

19.9.1 TWBR – TWI Bit Rate Register

Bit	7	6	5	4	3	2	1	0	
(0xB8)	TWBR7	TWBR6	TWBR5	TWBR4	TWBR3	TWBR2	TWBR1	TWBR0	TWBR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..0 – TWI Bit Rate Register**

TWBR selects the division factor for the bit rate generator. The bit rate generator is a frequency divider which generates the SCL clock frequency in the Master modes. See ”Bit Rate Generator Unit” on page 221 for calculating bit rates.

19.9.2 TWCR – TWI Control Register

Bit	7	6	5	4	3	2	1	0	
(0xBC)	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE	TWCR
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The TWCR is used to control the operation of the TWI. It is used to enable the TWI, to initiate a Master access by applying a START condition to the bus, to generate a Receiver acknowledge, to generate a stop condition, and to control halting of the bus while the data to be written to the bus are written to the TWDR. It also indicates a write collision if data is attempted written to TWDR while the register is inaccessible.

- **Bit 7 – TWINT: TWI Interrupt Flag**

This bit is set by hardware when the TWI has finished its current job and expects application software response. If the I-bit in SREG and TWIE in TWCR are set, the MCU will jump to the TWI Interrupt Vector. While the TWINT Flag is set, the SCL low period is stretched. The TWINT Flag must be cleared by software by writing a logic one to it. Note that this flag is not automatically cleared by hardware when executing the interrupt routine. Also note that clearing this flag starts the operation of the TWI, so all accesses to the TWI Address Register (TWAR), TWI Status Register (TWSR), and TWI Data Register (TWDR) must be complete before clearing this flag.

- **Bit 6 – TWEA: TWI Enable Acknowledge Bit**

The TWEA bit controls the generation of the acknowledge pulse. If the TWEA bit is written to one, the ACK pulse is generated on the TWI bus if the following conditions are met:

1. The device's own slave address has been received.
2. A general call has been received, while the TWGCE bit in the TWAR is set.
3. A data byte has been received in Master Receiver or Slave Receiver mode.

By writing the TWEA bit to zero, the device can be virtually disconnected from the 2-wire Serial Bus temporarily. Address recognition can then be resumed by writing the TWEA bit to one again.

- **Bit 5 – TWSTA: TWI START Condition Bit**

The application writes the TWSTA bit to one when it desires to become a Master on the 2-wire Serial Bus. The TWI hardware checks if the bus is available, and generates a START condition on the bus if it is free. However, if the bus is not free, the TWI waits until a STOP condition is detected, and then generates a new START condition to claim the bus Master status. TWSTA must be cleared by software when the START condition has been transmitted.

- **Bit 4 – TWSTO: TWI STOP Condition Bit**

Writing the TWSTO bit to one in Master mode will generate a STOP condition on the 2-wire Serial Bus. When the STOP condition is executed on the bus, the TWSTO bit is cleared automatically. In Slave mode, setting the TWSTO bit can be used to recover from an error condition. This will not generate a STOP condition, but the TWI returns to a well-defined unaddressed Slave mode and releases the SCL and SDA lines to a high impedance state.

- **Bit 3 – TWWC: TWI Write Collision Flag**

The TWWC bit is set when attempting to write to the TWI Data Register – TWDR when TWINT is low. This flag is cleared by writing the TWDR Register when TWINT is high.

- **Bit 2 – TWEN: TWI Enable Bit**

The TWEN bit enables TWI operation and activates the TWI interface. When TWEN is written to one, the TWI takes control over the I/O pins connected to the SCL and SDA pins, enabling the slew-rate limiters and spike filters. If this bit is written to zero, the TWI is switched off and all TWI transmissions are terminated, regardless of any ongoing operation.

- **Bit 1 – Res: Reserved Bit**

This bit is a reserved bit and will always read as zero.

- **Bit 0 – TWIE: TWI Interrupt Enable**

When this bit is written to one, and the I-bit in SREG is set, the TWI interrupt request will be activated for as long as the TWINT Flag is high.

19.9.3 TWSR – TWI Status Register

Bit	7	6	5	4	3	2	1	0	
(0xB9)	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0	TWSR
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	1	1	1	1	1	0	0	0	

- **Bits 7..3 – TWS: TWI Status**

These 5 bits reflect the status of the TWI logic and the 2-wire Serial Bus. The different status codes are described later in this section. Note that the value read from TWSR contains both the 5-bit status value and the 2-bit prescaler value. The application designer should mask the prescaler bits to zero when checking the Status bits. This makes status checking independent of prescaler setting. This approach is used in this datasheet, unless otherwise noted.

- **Bit 2 – Res: Reserved Bit**

This bit is reserved and will always read as zero.

- **Bits 1..0 – TWPS: TWI Prescaler Bits**

These bits can be read and written, and control the bit rate prescaler.

Table 19-7. TWI Bit Rate Prescaler

TWPS1	TWPS0	Prescaler Value
0	0	1
0	1	4
1	0	16
1	1	64

To calculate bit rates, see ”Bit Rate Generator Unit” on page 221. The value of TWPS1..0 is used in the equation.

19.9.4 TWDR – TWI Data Register

Bit	7	6	5	4	3	2	1	0	
(0xBB)	TWD7	TWD6	TWD5	TWD4	TWD3	TWD2	TWD1	TWD0	TWDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	1	1	1	1	1	1	1	

In Transmit mode, TWDR contains the next byte to be transmitted. In Receive mode, the TWDR contains the last byte received. It is writable while the TWI is not in the process of shifting a byte. This occurs when the TWI Interrupt Flag (TWINT) is set by hardware. Note that the Data Register cannot be initialized by the user before the first interrupt occurs. The data in TWDR remains stable as long as TWINT is set. While data is shifted out, data on the bus is simultaneously shifted in. TWDR always contains the last byte present on the bus, except after a wake up from a sleep mode by the TWI interrupt. In this case, the contents of TWDR is undefined. In the case

of a lost bus arbitration, no data is lost in the transition from Master to Slave. Handling of the ACK bit is controlled automatically by the TWI logic, the CPU cannot access the ACK bit directly.

- **Bits 7..0 – TWD: TWI Data Register**

These eight bits constitute the next data byte to be transmitted, or the latest data byte received on the 2-wire Serial Bus.

19.9.5 TWAR – TWI (Slave) Address Register

Bit	7	6	5	4	3	2	1	0	
(0xBA)	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	TWAR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	1	1	1	1	1	1	0	

The TWAR should be loaded with the 7-bit Slave address (in the seven most significant bits of TWAR) to which the TWI will respond when programmed as a Slave Transmitter or Receiver, and not needed in the Master modes. In multi master systems, TWAR must be set in masters which can be addressed as Slaves by other Masters.

The LSB of TWAR is used to enable recognition of the general call address (0x00). There is an associated address comparator that looks for the slave address (or general call address if enabled) in the received serial address. If a match is found, an interrupt request is generated.

- **Bits 7..1 – TWA: TWI (Slave) Address Register**

These seven bits constitute the slave address of the TWI unit.

- **Bit 0 – TWGCE: TWI General Call Recognition Enable Bit**

If set, this bit enables the recognition of a General Call given over the 2-wire Serial Bus.

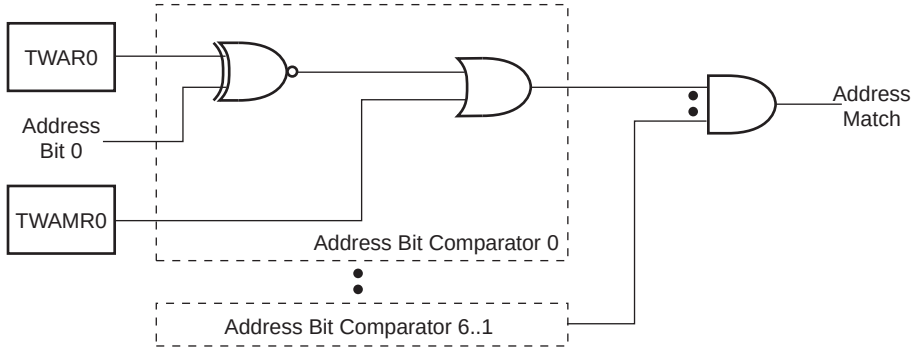
19.9.6 TWAMR – TWI (Slave) Address Mask Register

Bit	7	6	5	4	3	2	1	0	
(0xBD)	TWAM[6:0]							–	TWAMR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..1 – TWAM: TWI Address Mask**

The TWAMR can be loaded with a 7-bit Salve Address mask. Each of the bits in TWAMR can mask (disable) the corresponding address bits in the TWI Address Register (TWAR). If the mask bit is set to one then the address match logic ignores the compare between the incoming address bit and the corresponding bit in TWAR. Figure 19-22 shown the address match logic in detail.

Figure 19-22. TWI Address Match Logic, Block Diagram



- **Bit 0 – Res: Reserved Bit**

This bit is an unused bit in the ATmega48P/88P/168P/328P, and will always read as zero.

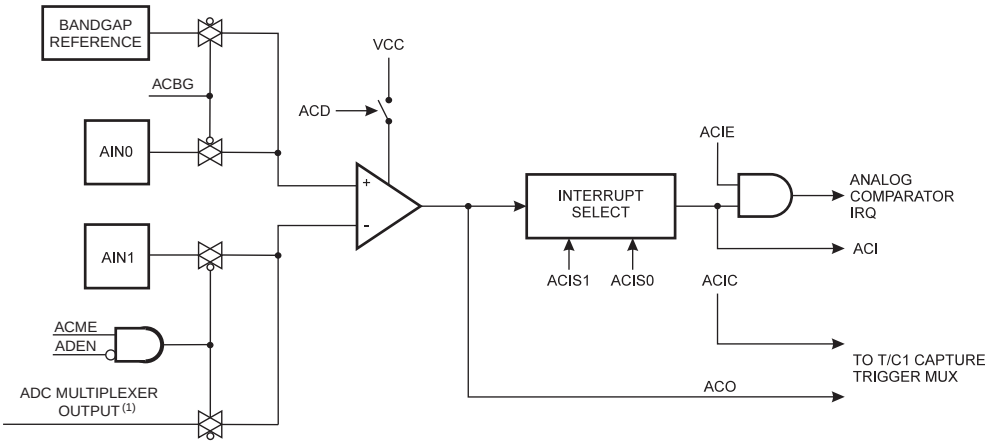
20. Analog Comparator

20.1 Overview

The Analog Comparator compares the input values on the positive pin AIN0 and negative pin AIN1. When the voltage on the positive pin AIN0 is higher than the voltage on the negative pin AIN1, the Analog Comparator output, ACO, is set. The comparator's output can be set to trigger the Timer/Counter1 Input Capture function. In addition, the comparator can trigger a separate interrupt, exclusive to the Analog Comparator. The user can select Interrupt triggering on comparator output rise, fall or toggle. A block diagram of the comparator and its surrounding logic is shown in Figure 20-1.

The Power Reduction ADC bit, PRADC, in "Minimizing Power Consumption" on page 42 must be disabled by writing a logical zero to be able to use the ADC input MUX.

Figure 20-1. Analog Comparator Block Diagram[2]



Notes: 1. See Table 20-1 on page 247.
2. Refer to Figure 1-1 on page 2 and Table 11-9 on page 88 for Analog Comparator pin placement.

20.2 Analog Comparator Multiplexed Input

It is possible to select any of the ADC7..0 pins to replace the negative input to the Analog Comparator. The ADC multiplexer is used to select this input, and consequently, the ADC must be switched off to utilize this feature. If the Analog Comparator Multiplexer Enable bit (ACME in ADCSRB) is set and the ADC is switched off (ADEN in ADCSRA is zero), MUX2..0 in ADMUX select the input pin to replace the negative input to the Analog Comparator, as shown in Table 20-1. If ACME is cleared or ADEN is set, AIN1 is applied to the negative input to the Analog Comparator

.

Table 20-1. Analog Comparator Multiplexed Input

ACME	ADEN	MUX2..0	Analog Comparator Negative Input
0	x	xxx	AIN1
1	1	xxx	AIN1
1	0	000	ADC0
1	0	001	ADC1
1	0	010	ADC2
1	0	011	ADC3
1	0	100	ADC4
1	0	101	ADC5
1	0	110	ADC6
1	0	111	ADC7

20.3 Register Description

20.3.1 ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
(0x7B)	–	ACME	–	–	–	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 6 – ACME: Analog Comparator Multiplexer Enable**

When this bit is written logic one and the ADC is switched off (ADEN in ADCSRA is zero), the ADC multiplexer selects the negative input to the Analog Comparator. When this bit is written logic zero, AIN1 is applied to the negative input of the Analog Comparator. For a detailed description of this bit, see ”Analog Comparator Multiplexed Input” on page 246.

20.3.2 ACSR – Analog Comparator Control and Status Register

Bit	7	6	5	4	3	2	1	0	
0x30 (0x50)	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	ACSR
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	N/A	0	0	0	0	0	

- **Bit 7 – ACD: Analog Comparator Disable**

When this bit is written logic one, the power to the Analog Comparator is switched off. This bit can be set at any time to turn off the Analog Comparator. This will reduce power consumption in Active and Idle mode. When changing the ACD bit, the Analog Comparator Interrupt must be disabled by clearing the ACIE bit in ACSR. Otherwise an interrupt can occur when the bit is changed.

- **Bit 6 – ACBG: Analog Comparator Bandgap Select**

When this bit is set, a fixed bandgap reference voltage replaces the positive input to the Analog Comparator. When this bit is cleared, AIN0 is applied to the positive input of the Analog Comparator. When the bandgap referance is used as input to the Analog Comparator, it will take a

certain time for the voltage to stabilize. If not stabilized, the first conversion may give a wrong value. See ”Internal Voltage Reference” on page 49

- **Bit 5 – ACO: Analog Comparator Output**

The output of the Analog Comparator is synchronized and then directly connected to ACO. The synchronization introduces a delay of 1 - 2 clock cycles.

- **Bit 4 – ACI: Analog Comparator Interrupt Flag**

This bit is set by hardware when a comparator output event triggers the interrupt mode defined by ACIS1 and ACIS0. The Analog Comparator interrupt routine is executed if the ACIE bit is set and the I-bit in SREG is set. ACI is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ACI is cleared by writing a logic one to the flag.

- **Bit 3 – ACIE: Analog Comparator Interrupt Enable**

When the ACIE bit is written logic one and the I-bit in the Status Register is set, the Analog Comparator interrupt is activated. When written logic zero, the interrupt is disabled.

- **Bit 2 – ACIC: Analog Comparator Input Capture Enable**

When written logic one, this bit enables the input capture function in Timer/Counter1 to be triggered by the Analog Comparator. The comparator output is in this case directly connected to the input capture front-end logic, making the comparator utilize the noise canceler and edge select features of the Timer/Counter1 Input Capture interrupt. When written logic zero, no connection between the Analog Comparator and the input capture function exists. To make the comparator trigger the Timer/Counter1 Input Capture interrupt, the ICIE1 bit in the Timer Interrupt Mask Register (TIMSK1) must be set.

- **Bits 1, 0 – ACIS1, ACIS0: Analog Comparator Interrupt Mode Select**

These bits determine which comparator events that trigger the Analog Comparator interrupt. The different settings are shown in Table 20-2.

Table 20-2. ACIS1/ACIS0 Settings

ACIS1	ACIS0	Interrupt Mode
0	0	Comparator Interrupt on Output Toggle.
0	1	Reserved
1	0	Comparator Interrupt on Falling Output Edge.
1	1	Comparator Interrupt on Rising Output Edge.

When changing the ACIS1/ACIS0 bits, the Analog Comparator Interrupt must be disabled by clearing its Interrupt Enable bit in the ACSR Register. Otherwise an interrupt can occur when the bits are changed.

20.3.3 DIDR1 – Digital Input Disable Register 1

Bit	7	6	5	4	3	2	1	0	
(0x7F)	–	–	–	–	–	–	AIN1D	AIN0D	DIDR1
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7..2 – Res: Reserved Bits**

These bits are unused bits in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bit 1, 0 – AIN1D, AIN0D: AIN1, AIN0 Digital Input Disable**

When this bit is written logic one, the digital input buffer on the AIN1/0 pin is disabled. The corresponding PIN Register bit will always read as zero when this bit is set. When an analog signal is applied to the AIN1/0 pin and the digital input from this pin is not needed, this bit should be written logic one to reduce power consumption in the digital input buffer.

21. Analog-to-Digital Converter

21.1 Features

- **10-bit Resolution**
- **0.5 LSB Integral Non-linearity**
- **± 2 LSB Absolute Accuracy**
- **13 - 260 µs Conversion Time**
- **Up to 76.9 kSPS (Up to 15 kSPS at Maximum Resolution)**
- **6 Multiplexed Single Ended Input Channels**
- **2 Additional Multiplexed Single Ended Input Channels (TQFP and QFN/MLF Package only)**
- **Temperature Sensor Input Channel**
- **Optional Left Adjustment for ADC Result Readout**
- **0 - V_{CC} ADC Input Voltage Range**
- **Selectable 1.1V ADC Reference Voltage**
- **Free Running or Single Conversion Mode**
- **Interrupt on ADC Conversion Complete**
- **Sleep Mode Noise Canceler**

21.2 Overview

The ATmega48P/88P/168P/328P features a 10-bit successive approximation ADC. The ADC is connected to an 8-channel Analog Multiplexer which allows eight single-ended voltage inputs constructed from the pins of Port A. The single-ended voltage inputs refer to 0V (GND).

The ADC contains a Sample and Hold circuit which ensures that the input voltage to the ADC is held at a constant level during conversion. A block diagram of the ADC is shown in Figure 21-1 on page 251.

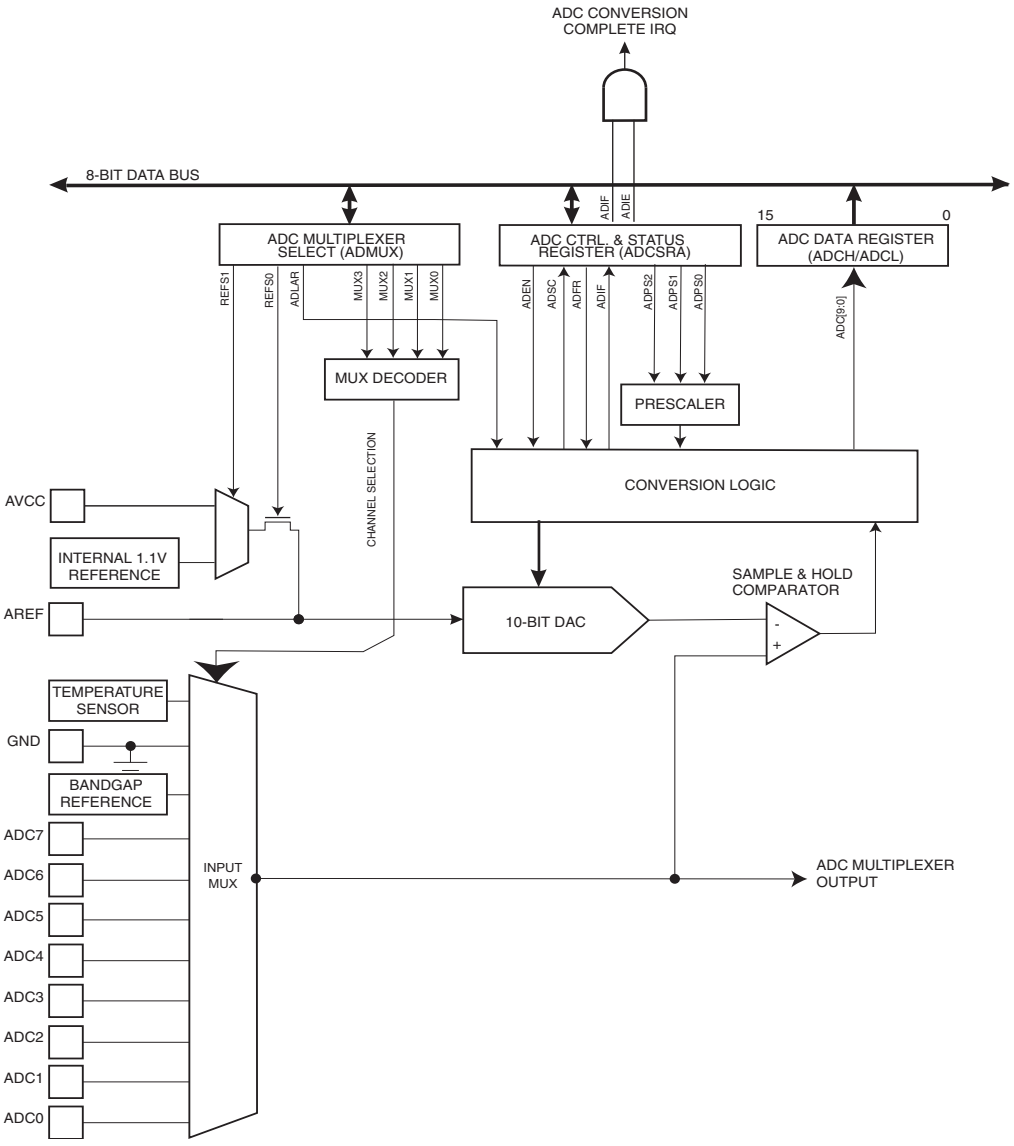
The ADC has a separate analog supply voltage pin, AV_{CC}. AV_{CC} must not differ more than ± 0.3V from V_{CC}. See the paragraph ”ADC Noise Canceler” on page 256 on how to connect this pin.

Internal reference voltages of nominally 1.1V or AV_{CC} are provided On-chip. The voltage reference may be externally decoupled at the AREF pin by a capacitor for better noise performance.

The Power Reduction ADC bit, PRADC, in ”Minimizing Power Consumption” on page 42 must be disabled by writing a logical zero to enable the ADC.

The ADC converts an analog input voltage to a 10-bit digital value through successive approximation. The minimum value represents GND and the maximum value represents the voltage on the AREF pin minus 1 LSB. Optionally, AV_{CC} or an internal 1.1V reference voltage may be connected to the AREF pin by writing to the REFSn bits in the ADMUX Register. The internal voltage reference may thus be decoupled by an external capacitor at the AREF pin to improve noise immunity.

Figure 21-1. Analog to Digital Converter Block Schematic Operation,



The analog input channel is selected by writing to the MUX bits in ADMUX. Any of the ADC input pins, as well as GND and a fixed bandgap voltage reference, can be selected as single ended inputs to the ADC. The ADC is enabled by setting the ADC Enable bit, ADEN in ADCSRA. Voltage reference and input channel selections will not go into effect until ADEN is set. The ADC does not consume power when ADEN is cleared, so it is recommended to switch off the ADC before entering power saving sleep modes.

The ADC generates a 10-bit result which is presented in the ADC Data Registers, ADCH and ADCL. By default, the result is presented right adjusted, but can optionally be presented left adjusted by setting the ADLAR bit in ADMUX.

If the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH, to ensure that the content of the Data Registers belongs to the same conversion. Once ADCL is read, ADC access to Data Registers is blocked. This means that if ADCL has been read, and a conversion completes before ADCH is

read, neither register is updated and the result from the conversion is lost. When ADCH is read, ADC access to the ADCH and ADCL Registers is re-enabled.

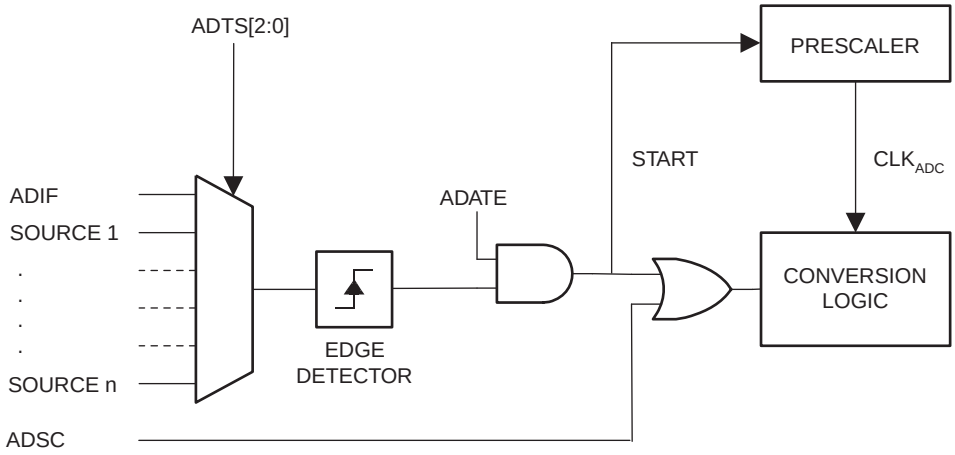
The ADC has its own interrupt which can be triggered when a conversion completes. When ADC access to the Data Registers is prohibited between reading of ADCH and ADCL, the interrupt will trigger even if the result is lost.

21.3 Starting a Conversion

A single conversion is started by disabling the Power Reduction ADC bit, PRADC, in "Minimizing Power Consumption" on page 42 by writing a logical zero to it and writing a logical one to the ADC Start Conversion bit, ADSC. This bit stays high as long as the conversion is in progress and will be cleared by hardware when the conversion is completed. If a different data channel is selected while a conversion is in progress, the ADC will finish the current conversion before performing the channel change.

Alternatively, a conversion can be triggered automatically by various sources. Auto Triggering is enabled by setting the ADC Auto Trigger Enable bit, ADATE in ADCSRA. The trigger source is selected by setting the ADC Trigger Select bits, ADTS in ADCSRB (See description of the ADTS bits for a list of the trigger sources). When a positive edge occurs on the selected trigger signal, the ADC prescaler is reset and a conversion is started. This provides a method of starting conversions at fixed intervals. If the trigger signal still is set when the conversion completes, a new conversion will not be started. If another positive edge occurs on the trigger signal during conversion, the edge will be ignored. Note that an Interrupt Flag will be set even if the specific interrupt is disabled or the Global Interrupt Enable bit in SREG is cleared. A conversion can thus be triggered without causing an interrupt. However, the Interrupt Flag must be cleared in order to trigger a new conversion at the next interrupt event.

Figure 21-2. ADC Auto Trigger Logic

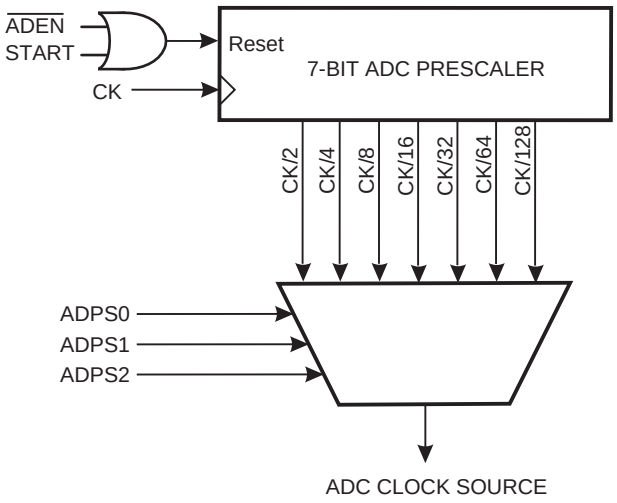


Using the ADC Interrupt Flag as a trigger source makes the ADC start a new conversion as soon as the ongoing conversion has finished. The ADC then operates in Free Running mode, constantly sampling and updating the ADC Data Register. The first conversion must be started by writing a logical one to the ADSC bit in ADCSRA. In this mode the ADC will perform successive conversions independently of whether the ADC Interrupt Flag, ADIF is cleared or not.

If Auto Triggering is enabled, single conversions can be started by writing ADSC in ADCSRA to one. ADSC can also be used to determine if a conversion is in progress. The ADSC bit will be read as one during a conversion, independently of how the conversion was started.

21.4 Prescaling and Conversion Timing

Figure 21-3. ADC Prescaler



By default, the successive approximation circuitry requires an input clock frequency between 50 kHz and 200 kHz to get maximum resolution. If a lower resolution than 10 bits is needed, the input clock frequency to the ADC can be higher than 200 kHz to get a higher sample rate.

The ADC module contains a prescaler, which generates an acceptable ADC clock frequency from any CPU frequency above 100 kHz. The prescaling is set by the ADPS bits in ADCSRA. The prescaler starts counting from the moment the ADC is switched on by setting the ADEN bit in ADCSRA. The prescaler keeps running for as long as the ADEN bit is set, and is continuously reset when ADEN is low.

When initiating a single ended conversion by setting the ADSC bit in ADCSRA, the conversion starts at the following rising edge of the ADC clock cycle.

A normal conversion takes 13 ADC clock cycles. The first conversion after the ADC is switched on (ADEN in ADCSRA is set) takes 25 ADC clock cycles in order to initialize the analog circuitry.

When the bandgap reference voltage is used as input to the ADC, it will take a certain time for the voltage to stabilize. If not stabilized, the first value read after the first conversion may be wrong.

The actual sample-and-hold takes place 1.5 ADC clock cycles after the start of a normal conversion and 13.5 ADC clock cycles after the start of an first conversion. When a conversion is complete, the result is written to the ADC Data Registers, and ADIF is set. In Single Conversion mode, ADSC is cleared simultaneously. The software may then set ADSC again, and a new conversion will be initiated on the first rising ADC clock edge.

When Auto Triggering is used, the prescaler is reset when the trigger event occurs. This assures a fixed delay from the trigger event to the start of conversion. In this mode, the sample-and-hold takes place two ADC clock cycles after the rising edge on the trigger source signal. Three additional CPU clock cycles are used for synchronization logic.

In Free Running mode, a new conversion will be started immediately after the conversion completes, while ADSC remains high. For a summary of conversion times, see Table 21-1 on page 255.

Figure 21-4. ADC Timing Diagram, First Conversion (Single Conversion Mode)

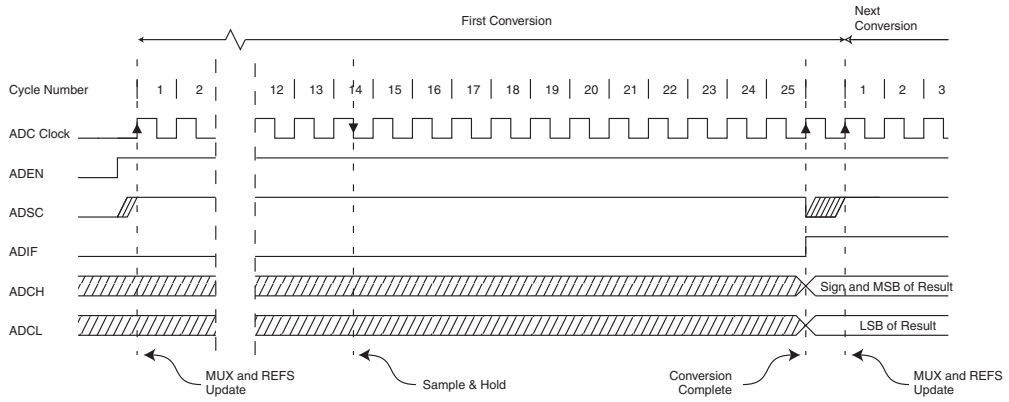


Figure 21-5. ADC Timing Diagram, Single Conversion

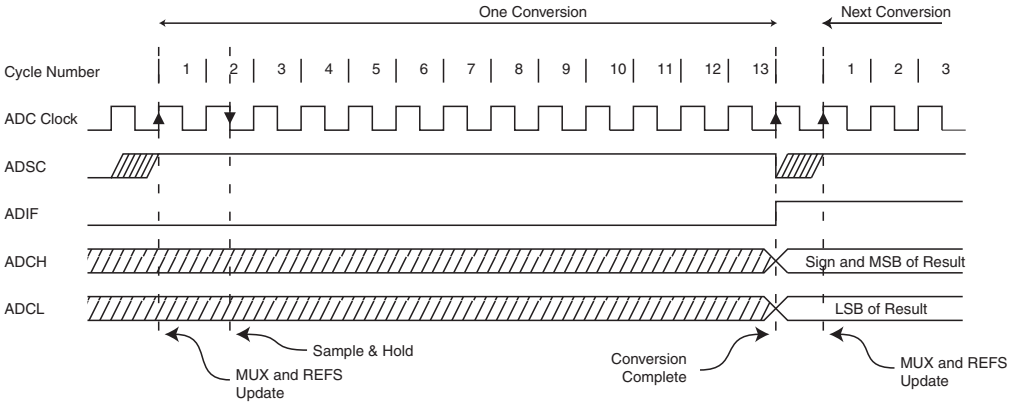


Figure 21-6. ADC Timing Diagram, Auto Triggered Conversion

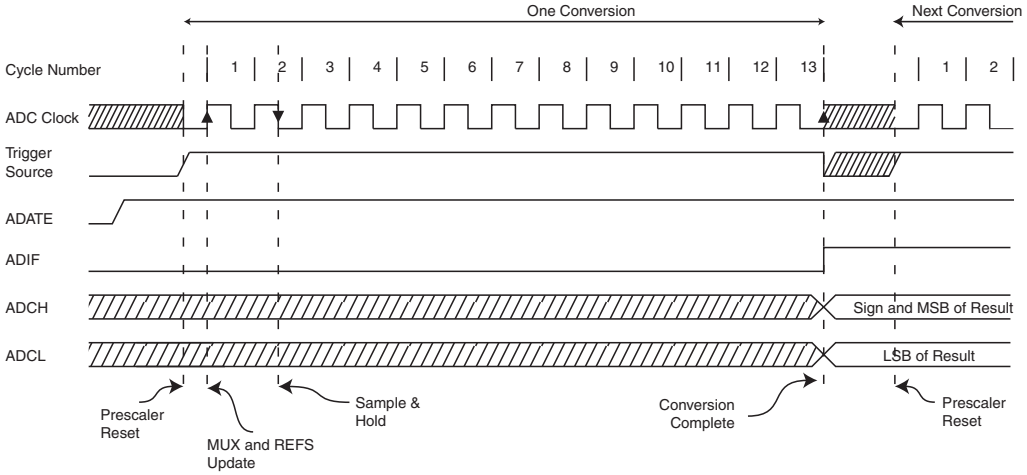


Figure 21-7. ADC Timing Diagram, Free Running Conversion

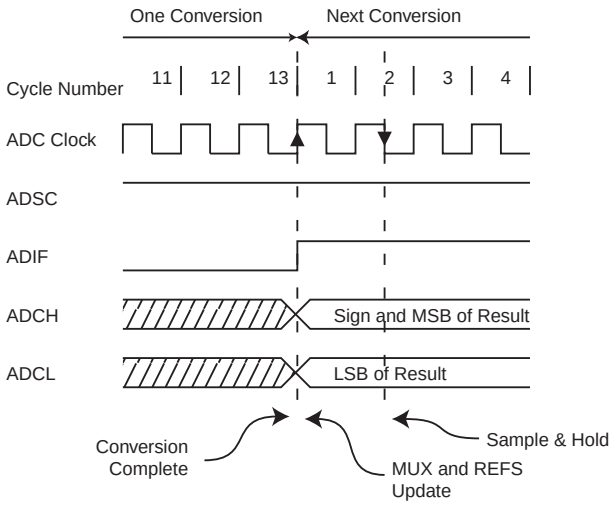


Table 21-1. ADC Conversion Time

Condition	Sample & Hold (Cycles from Start of Conversion)	Conversion Time (Cycles)
First conversion	13.5	25
Normal conversions, single ended	1.5	13
Auto Triggered conversions	2	13.5

21.5 Changing Channel or Reference Selection

The MUXn and REFS1:0 bits in the ADMUX Register are single buffered through a temporary register to which the CPU has random access. This ensures that the channels and reference selection only takes place at a safe point during the conversion. The channel and reference selection is continuously updated until a conversion is started. Once the conversion starts, the channel and reference selection is locked to ensure a sufficient sampling time for the ADC. Continuous updating resumes in the last ADC clock cycle before the conversion completes (ADIF in ADCSRA is set). Note that the conversion starts on the following rising ADC clock edge after ADSC is written. The user is thus advised not to write new channel or reference selection values to ADMUX until one ADC clock cycle after ADSC is written.

If Auto Triggering is used, the exact time of the triggering event can be indeterministic. Special care must be taken when updating the ADMUX Register, in order to control which conversion will be affected by the new settings.

If both ADATE and ADEN is written to one, an interrupt event can occur at any time. If the ADMUX Register is changed in this period, the user cannot tell if the next conversion is based on the old or the new settings. ADMUX can be safely updated in the following ways:

a. When ADATE or ADEN is cleared.
b. During conversion, minimum one ADC clock cycle after the trigger event.
c. After a conversion, before the Interrupt Flag used as trigger source is cleared.

When updating ADMUX in one of these conditions, the new settings will affect the next ADC conversion.

21.5.1 ADC Input Channels

When changing channel selections, the user should observe the following guidelines to ensure that the correct channel is selected:

In Single Conversion mode, always select the channel before starting the conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the conversion to complete before changing the channel selection.

In Free Running mode, always select the channel before starting the first conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the first conversion to complete, and then change the channel selection. Since the next conversion has already started automatically, the next result will reflect the previous channel selection. Subsequent conversions will reflect the new channel selection.

21.5.2 ADC Voltage Reference

The reference voltage for the ADC (V_{REF}) indicates the conversion range for the ADC. Single ended channels that exceed V_{REF} will result in codes close to 0x3FF. V_{REF} can be selected as either AV_{CC}, internal 1.1V reference, or external AREF pin.

AV_{CC} is connected to the ADC through a passive switch. The internal 1.1V reference is generated from the internal bandgap reference (V_{BG}) through an internal amplifier. In either case, the external AREF pin is directly connected to the ADC, and the reference voltage can be made more immune to noise by connecting a capacitor between the AREF pin and ground. V_{REF} can also be measured at the AREF pin with a high impedance voltmeter. Note that V_{REF} is a high impedance source, and only a capacitive load should be connected in a system.

If the user has a fixed voltage source connected to the AREF pin, the user may not use the other reference voltage options in the application, as they will be shorted to the external voltage. If no external voltage is applied to the AREF pin, the user may switch between AV_{CC} and 1.1V as reference selection. The first ADC conversion result after switching reference voltage source may be inaccurate, and the user is advised to discard this result.

21.6 ADC Noise Canceler

The ADC features a noise canceler that enables conversion during sleep mode to reduce noise induced from the CPU core and other I/O peripherals. The noise canceler can be used with ADC Noise Reduction and Idle mode. To make use of this feature, the following procedure should be used:

a. Make sure that the ADC is enabled and is not busy converting. Single Conversion mode must be selected and the ADC conversion complete interrupt must be enabled.
b. Enter ADC Noise Reduction mode (or Idle mode). The ADC will start a conversion once the CPU has been halted.
c. If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the CPU and execute the ADC Conversion Complete interrupt routine. If another interrupt wakes up the CPU before the ADC conversion is complete, that interrupt will be executed, and an ADC Conversion Complete interrupt request will be generated when the ADC conversion completes. The CPU will remain in active mode until a new sleep command is executed.

Note that the ADC will not be automatically turned off when entering other sleep modes than Idle mode and ADC Noise Reduction mode. The user is advised to write zero to ADEN before entering such sleep modes to avoid excessive power consumption.

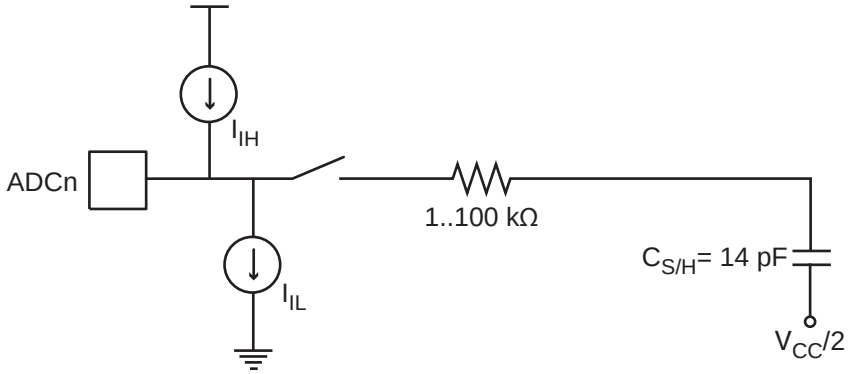
21.6.1 Analog Input Circuitry

The analog input circuitry for single ended channels is illustrated in Figure 21-8. An analog source applied to ADCn is subjected to the pin capacitance and input leakage of that pin, regardless of whether that channel is selected as input for the ADC. When the channel is selected, the source must drive the S/H capacitor through the series resistance (combined resistance in the input path).

The ADC is optimized for analog signals with an output impedance of approximately 10 kΩ or less. If such a source is used, the sampling time will be negligible. If a source with higher impedance is used, the sampling time will depend on how long time the source needs to charge the S/H capacitor, with can vary widely. The user is recommended to only use low impedance sources with slowly varying signals, since this minimizes the required charge transfer to the S/H capacitor.

Signal components higher than the Nyquist frequency ($f_{ADC}/2$) should not be present for either kind of channels, to avoid distortion from unpredictable signal convolution. The user is advised to remove high frequency components with a low-pass filter before applying the signals as inputs to the ADC.

Figure 21-8. Analog Input Circuitry



21.6.2 Analog Noise Canceling Techniques

Digital circuitry inside and outside the device generates EMI which might affect the accuracy of analog measurements. If conversion accuracy is critical, the noise level can be reduced by applying the following techniques:

a. Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground plane, and keep them well away from high-speed switching digital tracks.

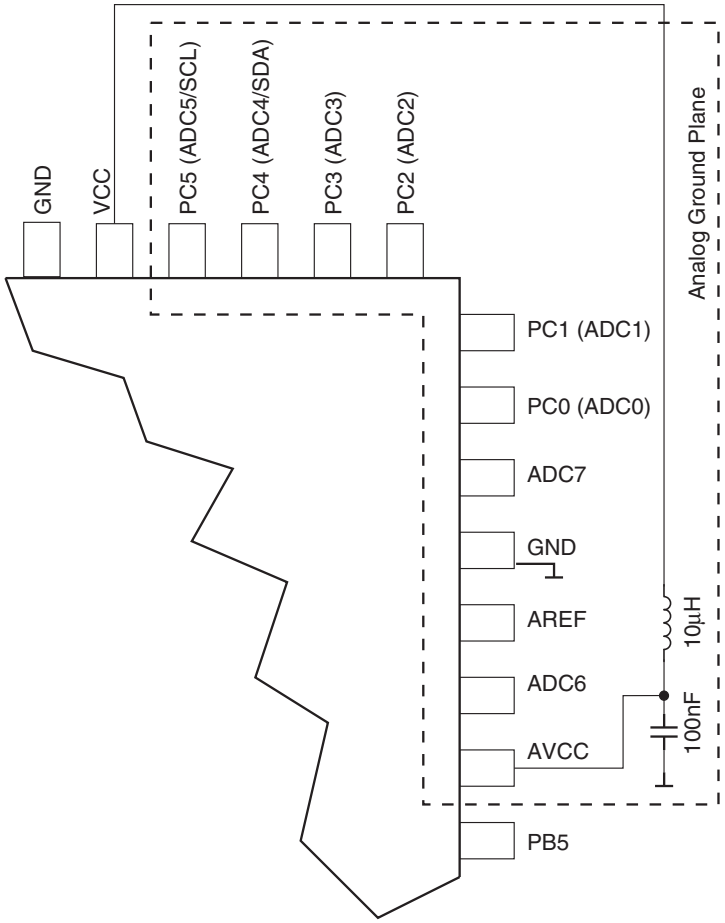
b. The AV_{CC} pin on the device should be connected to the digital V_{CC} supply voltage via an LC network as shown in Figure 21-9.

c. Use the ADC noise canceler function to reduce induced noise from the CPU.

d. If any ADC [3..0] port pins are used as digital outputs, it is essential that these do not switch while a conversion is in progress. However, using the 2-wire Interface (ADC4

and ADC5) will only affect the conversion on ADC4 and ADC5 and not the other ADC channels.

Figure 21-9. ADC Power Connections



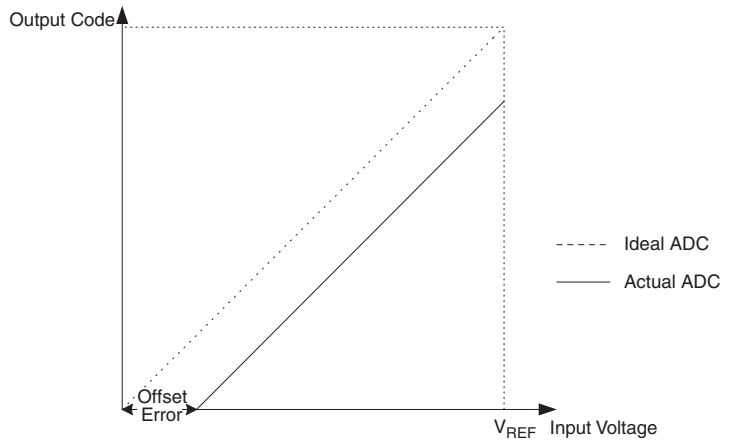
21.6.3 ADC Accuracy Definitions

An n-bit single-ended ADC converts a voltage linearly between GND and V_{REF} in 2^n steps (LSBs). The lowest code is read as 0, and the highest code is read as 2^n-1.

Several parameters describe the deviation from the ideal behavior:

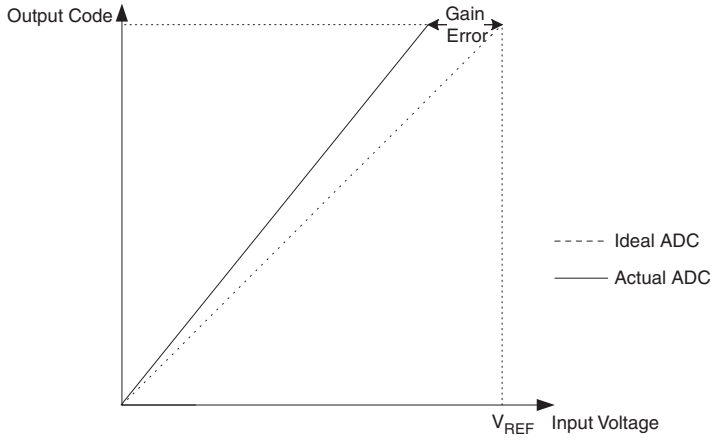
- Offset: The deviation of the first transition (0x000 to 0x001) compared to the ideal transition (at 0.5 LSB). Ideal value: 0 LSB.

Figure 21-10. Offset Error



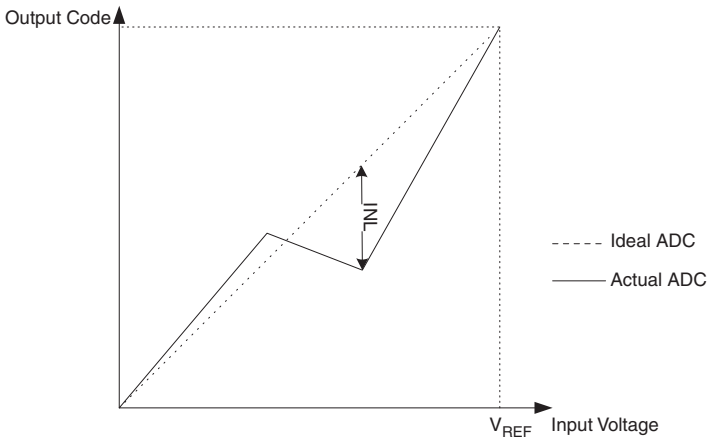
- Gain error: After adjusting for offset, the gain error is found as the deviation of the last transition (0x3FE to 0x3FF) compared to the ideal transition (at 1.5 LSB below maximum). Ideal value: 0 LSB

Figure 21-11. Gain Error



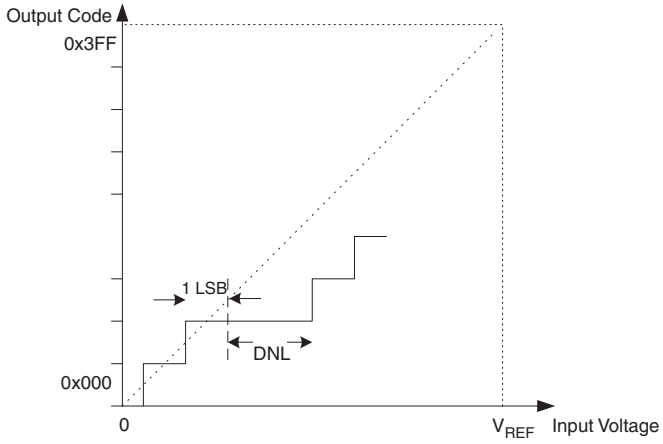
- Integral Non-linearity (INL): After adjusting for offset and gain error, the INL is the maximum deviation of an actual transition compared to an ideal transition for any code. Ideal value: 0 LSB.

Figure 21-12. Integral Non-linearity (INL)



- Differential Non-linearity (DNL): The maximum deviation of the actual code width (the interval between two adjacent transitions) from the ideal code width (1 LSB). Ideal value: 0 LSB.

Figure 21-13. Differential Non-linearity (DNL)



- Quantization Error: Due to the quantization of the input voltage into a finite number of codes, a range of input voltages (1 LSB wide) will code to the same value. Always ±0.5 LSB.
- Absolute accuracy: The maximum deviation of an actual (unadjusted) transition compared to an ideal transition for any code. This is the compound effect of offset, gain error, differential error, non-linearity, and quantization error. Ideal value: ±0.5 LSB.

21.7 ADC Conversion Result

After the conversion is complete (ADIF is high), the conversion result can be found in the ADC Result Registers (ADCL, ADCH).

For single ended conversion, the result is

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}$$

where V_{IN} is the voltage on the selected input pin and V_{REF} the selected voltage reference (see Table 21-3 on page 262 and Table 21-4 on page 263). 0x000 represents analog ground, and 0x3FF represents the selected reference voltage minus one LSB.

21.8 Temperature Measurement

The temperature measurement is based on an on-chip temperature sensor that is coupled to a single ended ADC8 channel. Selecting the ADC8 channel by writing the MUX3..0 bits in ADMUX register to "1000" enables the temperature sensor. The internal 1.1V voltage reference must also be selected for the ADC voltage reference source in the temperature sensor measurement. When the temperature sensor is enabled, the ADC converter can be used in single conversion mode to measure the voltage over the temperature sensor.

The measured voltage has a linear relationship to the temperature as described in Table 21-2. The voltage sensitivity is approximately 1 mV/°C and the accuracy of the temperature measurement is +/- 10°C.

Table 21-2. Temperature vs. Sensor Output Voltage (Typical Case)

Temperature / °C	**-45°C**	**+25°C**	**+85°C**
Voltage / mV	242 mV	314 mV	380 mV

The values described in Table 21-2 are typical values. However, due to the process variation the temperature sensor output voltage varies from one chip to another. To be capable of achieving more accurate results the temperature measurement can be calibrated in the application software. The software calibration requires that a calibration value is measured and stored in a register or EEPROM for each chip, as a part of the production test. The software calibration can be done utilizing the formula:

$$T = \{ [(ADCH << 8) \mid ADCL] - T_{OS} \} / k$$

where ADCn are the ADC data registers, k is a fixed coefficient and T_{OS} is the temperature sensor offset value determined and stored into EEPROM as a part of the production test.

21.9 Register Description

21.9.1 ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
(0x7C)	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:6 – REFS1:0: Reference Selection Bits**

These bits select the voltage reference for the ADC, as shown in Table 21-3. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSRA is set). The internal voltage reference options may not be used if an external reference voltage is being applied to the AREF pin.

Table 21-3. Voltage Reference Selections for ADC

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal V_{ref} turned off
0	1	AV_{CC} with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 1.1V Voltage Reference with external capacitor at AREF pin

- **Bit 5 – ADLAR: ADC Left Adjust Result**

The ADLAR bit affects the presentation of the ADC conversion result in the ADC Data Register. Write one to ADLAR to left adjust the result. Otherwise, the result is right adjusted. Changing the ADLAR bit will affect the ADC Data Register immediately, regardless of any ongoing conversions. For a complete description of this bit, see ”ADCL and ADCH – The ADC Data Register” on page 265.

- **Bit 4 – Res: Reserved Bit**

This bit is an unused bit in the ATmega48P/88P/168P/328P, and will always read as zero.

- **Bits 3:0 – MUX3:0: Analog Channel Selection Bits**

The value of these bits selects which analog inputs are connected to the ADC. See Table 21-4 for details. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSRA is set).

Table 21-4. Input Channel Selections

MUX3..0	Single Ended Input
0000	ADC0
0001	ADC1
0010	ADC2
0011	ADC3
0100	ADC4
0101	ADC5
0110	ADC6
0111	ADC7
1000	ADC8[1]
1001	(reserved)
1010	(reserved)
1011	(reserved)
1100	(reserved)
1101	(reserved)
1110	1.1V (V_{BG})
1111	0V (GND)

Note: 1. For Temperature Sensor.

21.9.2 ADCSRA – ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – ADEN: ADC Enable**

Writing this bit to one enables the ADC. By writing it to zero, the ADC is turned off. Turning the ADC off while a conversion is in progress, will terminate this conversion.

- **Bit 6 – ADSC: ADC Start Conversion**

In Single Conversion mode, write this bit to one to start each conversion. In Free Running mode, write this bit to one to start the first conversion. The first conversion after ADSC has been written after the ADC has been enabled, or if ADSC is written at the same time as the ADC is enabled, will take 25 ADC clock cycles instead of the normal 13. This first conversion performs initialization of the ADC.

ADSC will read as one as long as a conversion is in progress. When the conversion is complete, it returns to zero. Writing zero to this bit has no effect.

- **Bit 5 – ADATE: ADC Auto Trigger Enable**

When this bit is written to one, Auto Triggering of the ADC is enabled. The ADC will start a conversion on a positive edge of the selected trigger signal. The trigger source is selected by setting the ADC Trigger Select bits, ADTS in ADCSRB.

- **Bit 4 – ADIF: ADC Interrupt Flag**

This bit is set when an ADC conversion completes and the Data Registers are updated. The ADC Conversion Complete Interrupt is executed if the ADIE bit and the I-bit in SREG are set. ADIF is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ADIF is cleared by writing a logical one to the flag. Beware that if doing a Read-Modify-Write on ADCSRA, a pending interrupt can be disabled. This also applies if the SBI and CBI instructions are used.

- **Bit 3 – ADIE: ADC Interrupt Enable**

When this bit is written to one and the I-bit in SREG is set, the ADC Conversion Complete Interrupt is activated.

- **Bits 2:0 – ADPS2:0: ADC Prescaler Select Bits**

These bits determine the division factor between the system clock frequency and the input clock to the ADC.

Table 21-5. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

21.9.3 ADCL and ADCH – The ADC Data Register

21.9.3.1 ADLAR = 0

Bit	15	14	13	12	11	10	9	8	
(0x79)	–	–	–	–	–	–	ADC9	ADC8	ADCH
(0x78)	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

21.9.3.2 ADLAR = 1

Bit	15	14	13	12	11	10	9	8	
(0x79)	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
(0x78)	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

When an ADC conversion is complete, the result is found in these two registers.

When ADCL is read, the ADC Data Register is not updated until ADCH is read. Consequently, if the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH.

The ADLAR bit in ADMUX, and the MUXn bits in ADMUX affect the way the result is read from the registers. If ADLAR is set, the result is left adjusted. If ADLAR is cleared (default), the result is right adjusted.

- **ADC9:0: ADC Conversion Result**

These bits represent the result from the conversion, as detailed in ”ADC Conversion Result” on page 261.

21.9.4 ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
(0x7B)	–	ACME	–	–	–	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7, 5:3 – Res: Reserved Bits**

These bits are reserved for future use. To ensure compatibility with future devices, these bist must be written to zero when ADCSRB is written.

- **Bit 2:0 – ADTS2:0: ADC Auto Trigger Source**

If ADATE in ADCSRA is written to one, the value of these bits selects which source will trigger an ADC conversion. If ADATE is cleared, the ADTS2:0 settings will have no effect. A conversion will be triggered by the rising edge of the selected Interrupt Flag. Note that switching from a trigger source that is cleared to a trigger source that is set, will generate a positive edge on the

trigger signal. If ADEN in ADCSRA is set, this will start a conversion. Switching to Free Running mode (ADTS[2:0]=0) will not cause a trigger event, even if the ADC Interrupt Flag is set.

Table 21-6. ADC Auto Trigger Source Selections

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free Running mode
0	0	1	Analog Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match A
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter1 Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

21.9.5 DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
(0x7E)	–	–	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:6 – Res: Reserved Bits**

These bits are reserved for future use. To ensure compatibility with future devices, these bits must be written to zero when DIDR0 is written.

- **Bit 5:0 – ADC5D..ADC0D: ADC5..0 Digital Input Disable**

When this bit is written logic one, the digital input buffer on the corresponding ADC pin is disabled. The corresponding PIN Register bit will always read as zero when this bit is set. When an analog signal is applied to the ADC5..0 pin and the digital input from this pin is not needed, this bit should be written logic one to reduce power consumption in the digital input buffer.

Note that ADC pins ADC7 and ADC6 do not have digital input buffers, and therefore do not require Digital Input Disable bits.

22. debugWIRE On-chip Debug System

22.1 Features

- **Complete Program Flow Control**
- **Emulates All On-chip Functions, Both Digital and Analog, except RESET Pin**
- **Real-time Operation**
- **Symbolic Debugging Support (Both at C and Assembler Source Level, or for Other HLLs)**
- **Unlimited Number of Program Break Points (Using Software Break Points)**
- **Non-intrusive Operation**
- **Electrical Characteristics Identical to Real Device**
- **Automatic Configuration System**
- **High-Speed Operation**
- **Programming of Non-volatile Memories**

22.2 Overview

The debugWIRE On-chip debug system uses a One-wire, bi-directional interface to control the program flow, execute AVR instructions in the CPU and to program the different non-volatile memories.

22.3 Physical Interface

When the debugWIRE Enable (DWEN) Fuse is programmed and Lock bits are unprogrammed, the debugWIRE system within the target device is activated. The RESET port pin is configured as a wire-AND (open-drain) bi-directional I/O pin with pull-up enabled and becomes the communication gateway between target and emulator.

Figure 22-1. The debugWIRE Setup

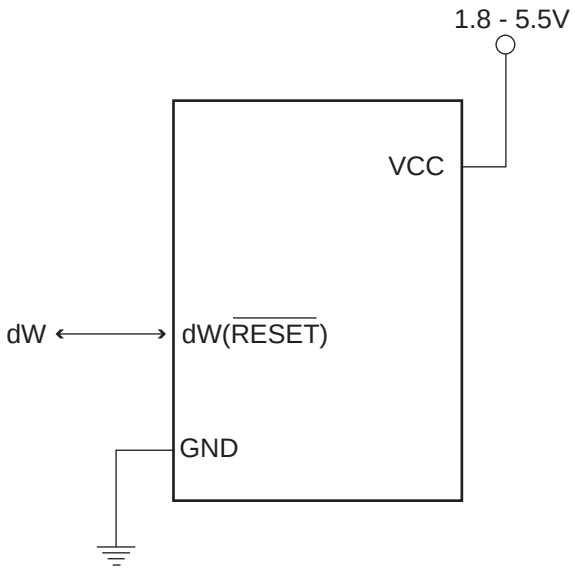


Figure 22-1 shows the schematic of a target MCU, with debugWIRE enabled, and the emulator connector. The system clock is not affected by debugWIRE and will always be the clock source selected by the CKSEL Fuses.

When designing a system where debugWIRE will be used, the following observations must be made for correct operation:

- Pull-up resistors on the dW/(RESET) line must not be smaller than 10kΩ. The pull-up resistor is not required for debugWIRE functionality.
- Connecting the RESET pin directly to V_{CC} will not work.
- Capacitors connected to the RESET pin must be disconnected when using debugWire.
- All external reset sources must be disconnected.

22.4 Software Break Points

debugWIRE supports Program memory Break Points by the AVR Break instruction. Setting a Break Point in AVR Studio® will insert a BREAK instruction in the Program memory. The instruction replaced by the BREAK instruction will be stored. When program execution is continued, the stored instruction will be executed before continuing from the Program memory. A break can be inserted manually by putting the BREAK instruction in the program.

The Flash must be re-programmed each time a Break Point is changed. This is automatically handled by AVR Studio through the debugWIRE interface. The use of Break Points will therefore reduce the Flash Data retention. Devices used for debugging purposes should not be shipped to end customers.

22.5 Limitations of debugWIRE

The debugWIRE communication pin (dW) is physically located on the same pin as External Reset (RESET). An External Reset source is therefore not supported when the debugWIRE is enabled.

A programmed DWEN Fuse enables some parts of the clock system to be running in all sleep modes. This will increase the power consumption while in sleep. Thus, the DWEN Fuse should be disabled when debugWire is not used.

22.6 Register Description

The following section describes the registers used with the debugWire.

22.6.1 DWDR – debugWire Data Register

Bit	7	6	5	4	3	2	1	0	
	DWDR[7:0]								DWDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The DWDR Register provides a communication channel from the running program in the MCU to the debugger. This register is only accessible by the debugWIRE and can therefore not be used as a general purpose register in the normal operations.

23. Self-Programming the Flash, ATmega48P

23.1 Overview

In ATmega48P, there is no Read-While-Write support, and no separate Boot Loader Section. The SPM instruction can be executed from the entire Flash.

The device provides a Self-Programming mechanism for downloading and uploading program code by the MCU itself. The Self-Programming can use any available data interface and associated protocol to read code and write (program) that code into the Program memory.

The Program memory is updated in a page by page fashion. Before programming a page with the data stored in the temporary page buffer, the page must be erased. The temporary page buffer is filled one word at a time using SPM and the buffer can be filled either before the Page Erase command or between a Page Erase and a Page Write operation:

Alternative 1, fill the buffer before a Page Erase

- Fill temporary page buffer
- Perform a Page Erase
- Perform a Page Write

Alternative 2, fill the buffer after Page Erase

- Perform a Page Erase
- Fill temporary page buffer
- Perform a Page Write

If only a part of the page needs to be changed, the rest of the page must be stored (for example in the temporary page buffer) before the erase, and then be re-written. When using alternative 1, the Boot Loader provides an effective Read-Modify-Write feature which allows the user software to first read the page, do the necessary changes, and then write back the modified data. If alternative 2 is used, it is not possible to read the old data while loading since the page is already erased. The temporary page buffer can be accessed in a random sequence. It is essential that the page address used in both the Page Erase and Page Write operation is addressing the same page.

23.1.1 Performing Page Erase by SPM

To execute Page Erase, set up the address in the Z-pointer, write “00000011” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE in the Z-register. Other bits in the Z-pointer will be ignored during this operation.

- The CPU is halted during the Page Erase operation.

23.1.2 Filling the Temporary Buffer (Page Loading)

To write an instruction word, set up the address in the Z-pointer and data in R1:R0, write “00000001” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The content of PCWORD in the Z-register is used to address the data in the temporary buffer. The temporary buffer will auto-erase after a Page Write operation or by writing the RWWSRE bit in SPMCSR. It is also erased after a system reset. Note that it is not possible to write more than one time to each address without erasing the temporary buffer.

If the EEPROM is written in the middle of an SPM Page Load operation, all data loaded will be lost.

23.1.3 Performing a Page Write

To execute Page Write, set up the address in the Z-pointer, write “00000101” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE. Other bits in the Z-pointer must be written to zero during this operation.

- The CPU is halted during the Page Write operation.

23.2 Addressing the Flash During Self-Programming

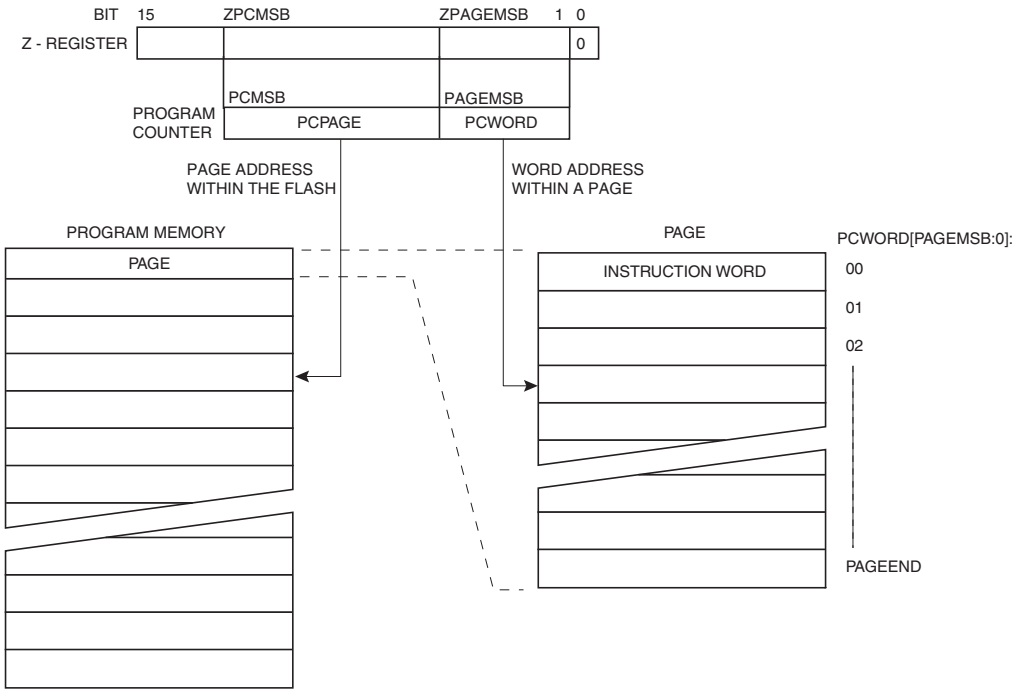
The Z-pointer is used to address the SPM commands.

Bit	15	14	13	12	11	10	9	8
ZH (R31)	**Z15**	**Z14**	**Z13**	**Z12**	**Z11**	**Z10**	**Z9**	**Z8**
ZL (R30)	**Z7**	**Z6**	**Z5**	**Z4**	**Z3**	**Z2**	**Z1**	**Z0**
	7	6	5	4	3	2	1	0

Since the Flash is organized in pages (see Table 25-11 on page 299), the Program Counter can be treated as having two different sections. One section, consisting of the least significant bits, is addressing the words within a page, while the most significant bits are addressing the pages. This is shown in Figure 24-3 on page 282. Note that the Page Erase and Page Write operations are addressed independently. Therefore it is of major importance that the software addresses the same page in both the Page Erase and Page Write operation.

The LPM instruction uses the Z-pointer to store the address. Since this instruction addresses the Flash byte-by-byte, also the LSB (bit Z0) of the Z-pointer is used.

Figure 23-1. Addressing the Flash During SPM[1]



Note: 1. The different variables used in Figure 24-3 are listed in Table 25-11 on page 299.

23.2.1 EEPROM Write Prevents Writing to SPMCSR

Note that an EEPROM write operation will block all software programming to Flash. Reading the Fuses and Lock bits from software will also be prevented during the EEPROM write operation. It is recommended that the user checks the status bit (EEPE) in the EECR Register and verifies that the bit is cleared before writing to the SPMCSR Register.

23.2.2 Reading the Fuse and Lock Bits from Software

It is possible to read both the Fuse and Lock bits from software. To read the Lock bits, load the Z-pointer with 0x0001 and set the BLBSET and SELFPRGEN bits in SPMCSR. When an LPM instruction is executed within three CPU cycles after the BLBSET and SELFPRGEN bits are set in SPMCSR, the value of the Lock bits will be loaded in the destination register. The BLBSET and SELFPRGEN bits will auto-clear upon completion of reading the Lock bits or if no LPM instruction is executed within three CPU cycles or no SPM instruction is executed within four CPU cycles. When BLBSET and SELFPRGEN are cleared, LPM will work as described in the Instruction set Manual.

Bit	7	6	5	4	3	2	1	0
Rd	–	–	–	–	–	–	**LB2**	**LB1**

The algorithm for reading the Fuse Low byte is similar to the one described above for reading the Lock bits. To read the Fuse Low byte, load the Z-pointer with 0x0000 and set the BLBSET and SELFPRGEN bits in SPMCSR. When an LPM instruction is executed within three cycles after the BLBSET and SELFPRGEN bits are set in the SPMCSR, the value of the Fuse Low byte (FLB) will be loaded in the destination register as shown below.See Table 25-5 on page 296 for a detailed description and mapping of the Fuse Low byte.

Bit	7	6	5	4	3	2	1	0
Rd	**FLB7**	**FLB6**	**FLB5**	**FLB4**	**FLB3**	**FLB2**	**FLB1**	**FLB0**

Similarly, when reading the Fuse High byte (FHB), load 0x0003 in the Z-pointer. When an LPM instruction is executed within three cycles after the BLBSET and SELFPRGEN bits are set in the SPMCSR, the value of the Fuse High byte will be loaded in the destination register as shown below. See Table 25-4 on page 295 for detailed description and mapping of the Extended Fuse byte.

Bit	7	6	5	4	3	2	1	0
Rd	**FHB7**	**FHB6**	**FHB5**	**FHB4**	**FHB3**	**FHB2**	**FHB1**	**FHB0**

Similarly, when reading the Extended Fuse byte (EFB), load 0x0002 in the Z-pointer. When an LPM instruction is executed within three cycles after the BLBSET and SELFPRGEN bits are set in the SPMCSR, the value of the Extended Fuse byte will be loaded in the destination register as shown below. See Table 25-5 on page 296 for detailed description and mapping of the Extended Fuse byte.

Bit	7	6	5	4	3	2	1	0
Rd	**FHB7**	**FHB6**	**FHB5**	**FHB4**	**FHB3**	**FHB2**	**FHB1**	**FHB0**

Fuse and Lock bits that are programmed, will be read as zero. Fuse and Lock bits that are unprogrammed, will be read as one.

23.2.3 Preventing Flash Corruption

During periods of low V_{CC}, the Flash program can be corrupted because the supply voltage is too low for the CPU and the Flash to operate properly. These issues are the same as for board level systems using the Flash, and the same design solutions should be applied.

A Flash program corruption can be caused by two situations when the voltage is too low. First, a regular write sequence to the Flash requires a minimum voltage to operate correctly. Secondly, the CPU itself can execute instructions incorrectly, if the supply voltage for executing instructions is too low.

Flash corruption can easily be avoided by following these design recommendations (one is sufficient):

1. Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal Brown-out Detector (BOD) if the operating voltage matches the detection level. If not, an external low V_{CC} reset protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided that the power supply voltage is sufficient.
2. Keep the AVR core in Power-down sleep mode during periods of low V_{CC}. This will prevent the CPU from attempting to decode and execute instructions, effectively protecting the SPMCSR Register and thus the Flash from unintentional writes.

23.2.4 Programming Time for Flash when Using SPM

The calibrated RC Oscillator is used to time Flash accesses. Table 24-6 shows the typical programming time for Flash accesses from the CPU.

Table 23-1. SPM Programming Time(1)

Symbol	Min Programming Time	Max Programming Time
Flash write (Page Erase, Page Write, and write Lock bits by SPM)	3.7 ms	4.5 ms

Note: 1. Minimum and maximum programming time is per individual operation.

23.2.5 Simple Assembly Code Example for a Boot Loader

Note that the RWWSB bit will always be read as zero in ATmega48P. Nevertheless, it is recommended to check this bit as shown in the code example, to ensure compatibility with devices supporting Read-While-Write.

```
  ;-the routine writes one page of data from RAM to Flash
  ; the first data location in RAM is pointed to by the Y pointer
  ; the first data location in Flash is pointed to by the Z-pointer
  ;-error handling is not included
  ;-the routine must be placed inside the Boot space
  ; (at least the Do_spm sub routine). Only code inside NRWW section can
  ; be read during Self-Programming (Page Erase and Page Write).
  ;-registers used: r0, r1, temp1 (r16), temp2 (r17), looplo (r24),
  ; loophi (r25), spmcrval (r20)
  ; storing and restoring of registers is not included in the routine
  ; register usage can be optimized at the expense of code size
  ;-It is assumed that either the interrupt table is moved to the Boot
  ; loader section or that the interrupts are disabled.
.equ PAGESIZEB = PAGESIZE*2   ;PAGESIZEB is page size in BYTES, not words
.org SMALLBOOTSTART
Write_page:
  ; Page Erase
  ldi  spmcrval, (1<<PGERS) | (1<<SELFPRGEN)
  rcallDo_spm

  ; re-enable the RWW section
  ldi  spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
  rcallDo_spm

  ; transfer data from RAM to Flash page buffer
  ldi  looplo, low(PAGESIZEB)   ;init loop variable
  ldi  loophi, high(PAGESIZEB)  ;not required for PAGESIZEB<=256
Wrloop:
  ld   r0, Y+
  ld   r1, Y+
  ldi  spmcrval, (1<<SELFPRGEN)
  rcallDo_spm
  adiw ZH:ZL, 2
  sbiw loophi:looplo, 2         ;use subi for PAGESIZEB<=256
  brne Wrloop

  ; execute Page Write
  subi ZL, low(PAGESIZEB)       ;restore pointer
  sbci ZH, high(PAGESIZEB)      ;not required for PAGESIZEB<=256
  ldi  spmcrval, (1<<PGWRT) | (1<<SELFPRGEN)
  rcallDo_spm

  ; re-enable the RWW section
  ldi  spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
  rcallDo_spm

  ; read back and check, optional
  ldi  looplo, low(PAGESIZEB)   ;init loop variable
  ldi  loophi, high(PAGESIZEB)  ;not required for PAGESIZEB<=256
  subi YL, low(PAGESIZEB)       ;restore pointer
```

```
  sbci YH, high(PAGESIZEB)
Rdloop:
  lpm  r0, Z+
  ld   r1, Y+
  cpse r0, r1
  rjmp Error
  sbiw loophi:looplo, 1         ;use subi for PAGESIZEB<=256
  brne Rdloop

  ; return to RWW section
  ; verify that RWW section is safe to read
Return:
  in   temp1, SPMCSR
  sbrs temp1, RWWSB     ; If RWWSB is set, the RWW section is not ready yet
  ret
  ; re-enable the RWW section
  ldi  spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
  rcallDo_spm
  rjmp Return

Do_spm:
  ; check for previous SPM complete
Wait_spm:
  in   temp1, SPMCSR
  sbrc temp1, SELFPRGEN
  rjmp Wait_spm
  ; input: spmcrval determines SPM action
  ; disable interrupts if enabled, store status
  in   temp2, SREG
  cli
  ; check that no EEPROM write access is present
Wait_ee:
  sbic EECR, EEPE
  rjmp Wait_ee
  ; SPM timed sequence
  out  SPMCSR, spmcrval
  spm
  ; restore SREG (to enable interrupts if originally enabled)
  out  SREG, temp2
  ret
```

23.3 Register Description’

23.3.1 SPMCSR – Store Program Memory Control and Status Register

The Store Program Memory Control and Status Register contains the control bits needed to control the Program memory operations.

Bit	7	6	5	4	3	2	1	0	
0x37 (0x57)	SPMIE	RWWSB	–	RWWSRE	BLBSET	PGWRT	PGERS	SELFPRGEN	SPMCSR
Read/Write	R/W	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – SPMIE: SPM Interrupt Enable**

When the SPMIE bit is written to one, and the I-bit in the Status Register is set (one), the SPM ready interrupt will be enabled. The SPM ready Interrupt will be executed as long as the SELFPRGEN bit in the SPMCSR Register is cleared. The interrupt will not be generated during EEPROM write or SPM.

- **Bit 6 – RWWSB: Read-While-Write Section Busy**

This bit is for compatibility with devices supporting Read-While-Write. It will always read as zero in ATmega48P.

- **Bit 5 – Res: Reserved Bit**

This bit is a reserved bit in the ATmega48P/88P/168P/328P and will always read as zero.

- **Bit 4 – RWWSRE: Read-While-Write Section Read Enable**

The functionality of this bit in ATmega48P is a subset of the functionality in ATmega88P/168P. If the RWWSRE bit is written while filling the temporary page buffer, the temporary page buffer will be cleared and the data will be lost.

- **Bit 3 – BLBSET: Boot Lock Bit Set**

The functionality of this bit in ATmega48P is a subset of the functionality in ATmega88P/168P. An LPM instruction within three cycles after BLBSET and SELFPRGEN are set in the SPMCSR Register, will read either the Lock bits or the Fuse bits (depending on Z0 in the Z-pointer) into the destination register. See ”Reading the Fuse and Lock Bits from Software” on page 271 for details.

- **Bit 2 – PGWRT: Page Write**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles executes Page Write, with the data stored in the temporary buffer. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The PGWRT bit will auto-clear upon completion of a Page Write, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire Page Write operation.

- **Bit 1 – PGERS: Page Erase**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles executes Page Erase. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The PGERS bit will auto-clear upon completion of a Page Erase, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire Page Write operation.

- **Bit 0 – SELFPRGEN: Self Programming Enable**

This bit enables the SPM instruction for the next four clock cycles. If written to one together with either RWWSRE, BLBSET, PGWRT, or PGERS, the following SPM instruction will have a special meaning, see description above. If only SELFPRGEN is written, the following SPM instruction will store the value in R1:R0 in the temporary page buffer addressed by the Z-pointer. The LSB of the Z-pointer is ignored. The SELFPRGEN bit will auto-clear upon completion of an SPM instruction, or if no SPM instruction is executed within four clock cycles. During Page Erase and Page Write, the SELFPRGEN bit remains high until the operation is completed.

Writing any other combination than “10001”, “01001”, “00101”, “00011” or “00001” in the lower five bits will have no effect.

24. Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P

24.1 Features

- **Read-While-Write Self-Programming**
- **Flexible Boot Memory Size**
- **High Security (Separate Boot Lock Bits for a Flexible Protection)**
- **Separate Fuse to Select Reset Vector**
- **Optimized Page[1] Size**
- **Code Efficient Algorithm**
- **Efficient Read-Modify-Write Support**

Note: 1. A page is a section in the Flash consisting of several bytes (see Table 25-11 on page 299) used during programming. The page organization does not affect normal operation.

24.2 Overview

In ATmega88P, ATmega168P and ATmega328P, the Boot Loader Support provides a real Read-While-Write Self-Programming mechanism for downloading and uploading program code by the MCU itself. This feature allows flexible application software updates controlled by the MCU using a Flash-resident Boot Loader program. The Boot Loader program can use any available data interface and associated protocol to read code and write (program) that code into the Flash memory, or read the code from the program memory. The program code within the Boot Loader section has the capability to write into the entire Flash, including the Boot Loader memory. The Boot Loader can thus even modify itself, and it can also erase itself from the code if the feature is not needed anymore. The size of the Boot Loader memory is configurable with fuses and the Boot Loader has two separate sets of Boot Lock bits which can be set independently. This gives the user a unique flexibility to select different levels of protection.

24.3 Application and Boot Loader Flash Sections

The Flash memory is organized in two main sections, the Application section and the Boot Loader section (see Figure 24-2). The size of the different sections is configured by the BOOTSZ Fuses as shown in Table 24-7 on page 289 and Figure 24-2. These two sections can have different level of protection since they have different sets of Lock bits.

24.3.1 Application Section

The Application section is the section of the Flash that is used for storing the application code. The protection level for the Application section can be selected by the application Boot Lock bits (Boot Lock bits 0), see Table 24-2 on page 281. The Application section can never store any Boot Loader code since the SPM instruction is disabled when executed from the Application section.

24.3.2 BLS – Boot Loader Section

While the Application section is used for storing the application code, the The Boot Loader software must be located in the BLS since the SPM instruction can initiate a programming when executing from the BLS only. The SPM instruction can access the entire Flash, including the BLS itself. The protection level for the Boot Loader section can be selected by the Boot Loader Lock bits (Boot Lock bits 1), see Table 24-3 on page 281.

24.4 Read-While-Write and No Read-While-Write Flash Sections

Whether the CPU supports Read-While-Write or if the CPU is halted during a Boot Loader software update is dependent on which address that is being programmed. In addition to the two sections that are configurable by the BOOTSZ Fuses as described above, the Flash is also divided into two fixed sections, the Read-While-Write (RWW) section and the No Read-While-Write (NRWW) section. The limit between the RWW- and NRWW sections is given in Table 24-8 on page 289 and Figure 24-2 on page 280. The main difference between the two sections is:

- When erasing or writing a page located inside the RWW section, the NRWW section can be read during the operation.
- When erasing or writing a page located inside the NRWW section, the CPU is halted during the entire operation.

Note that the user software can never read any code that is located inside the RWW section during a Boot Loader software operation. The syntax “Read-While-Write section” refers to which section that is being programmed (erased or written), not which section that actually is being read during a Boot Loader software update.

24.4.1 RWW – Read-While-Write Section

If a Boot Loader software update is programming a page inside the RWW section, it is possible to read code from the Flash, but only code that is located in the NRWW section. During an ongoing programming, the software must ensure that the RWW section never is being read. If the user software is trying to read code that is located inside the RWW section (i.e., by a call/jmp/lpm or an interrupt) during programming, the software might end up in an unknown state. To avoid this, the interrupts should either be disabled or moved to the Boot Loader section. The Boot Loader section is always located in the NRWW section. The RWW Section Busy bit (RWWSB) in the Store Program Memory Control and Status Register (SPMCSR) will be read as logical one as long as the RWW section is blocked for reading. After a programming is completed, the RWWSB must be cleared by software before reading code located in the RWW section. See Section “24.9.1” on page 292. for details on how to clear RWWSB.

24.4.2 NRWW – No Read-While-Write Section

The code located in the NRWW section can be read when the Boot Loader software is updating a page in the RWW section. When the Boot Loader code updates the NRWW section, the CPU is halted during the entire Page Erase or Page Write operation.

Table 24-1. Read-While-Write Features

Which Section does the Z-pointer Address during the Programming?	Which Section can be read during Programming?	CPU Halted?	Read-While-Write Supported?
RWW Section	NRWW Section	No	Yes
NRWW Section	None	Yes	No

Figure 24-1. Read-While-Write vs. No Read-While-Write

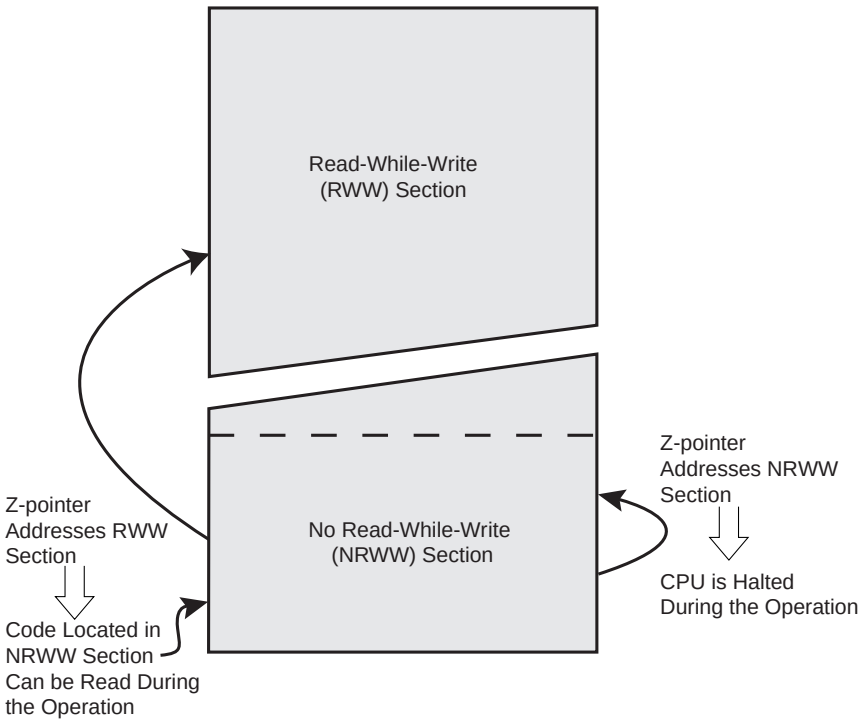
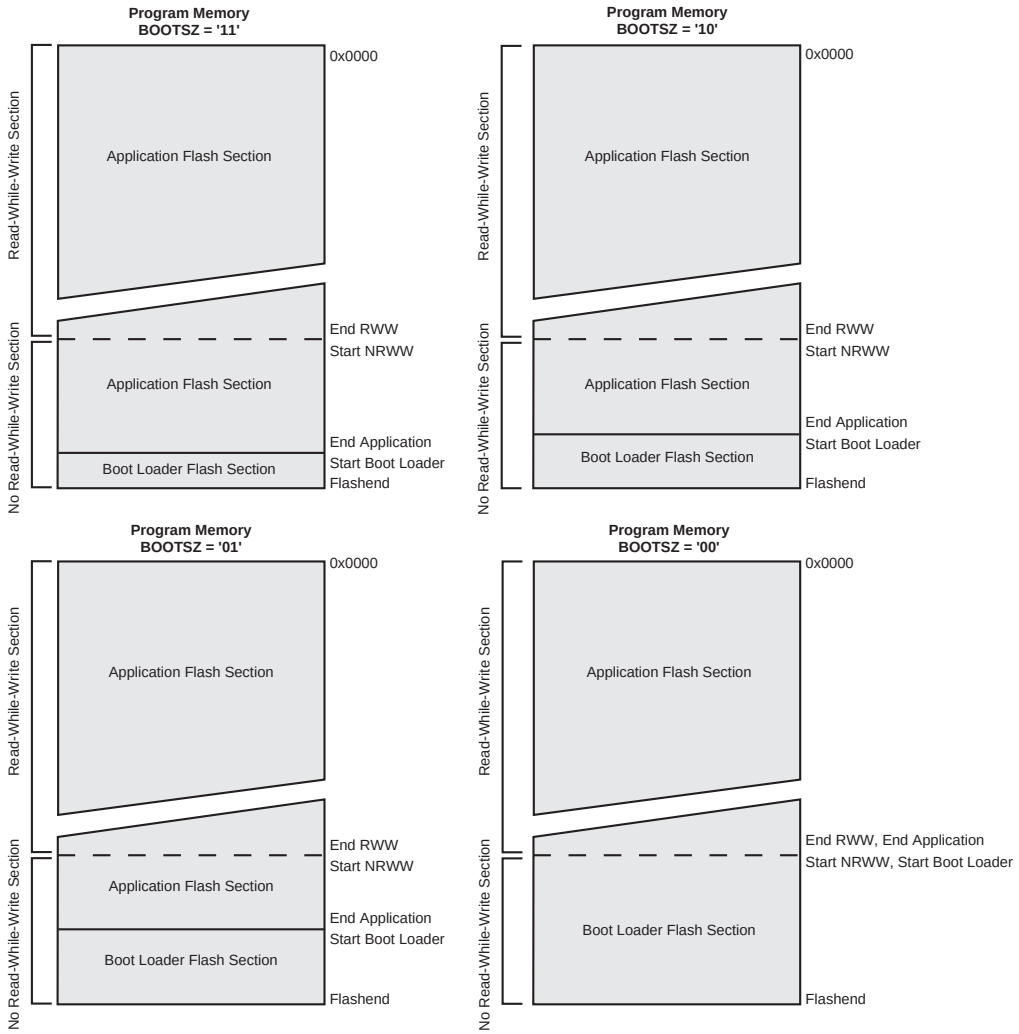


Figure 24-2. Memory Sections



Note: 1. The parameters in the figure above are given in Table 24-7 on page 289.

24.5 Boot Loader Lock Bits

If no Boot Loader capability is needed, the entire Flash is available for application code. The Boot Loader has two separate sets of Boot Lock bits which can be set independently. This gives the user a unique flexibility to select different levels of protection.

The user can select:

- To protect the entire Flash from a software update by the MCU.
- To protect only the Boot Loader Flash section from a software update by the MCU.
- To protect only the Application Flash section from a software update by the MCU.
- Allow software update in the entire Flash.

See Table 24-2 and Table 24-3 for further details. The Boot Lock bits can be set in software and in Serial or Parallel Programming mode, but they can be cleared by a Chip Erase command only. The general Write Lock (Lock Bit mode 2) does not control the programming of the Flash memory by SPM instruction. Similarly, the general Read/Write Lock (Lock Bit mode 1) does not control reading nor writing by LPM/SPM, if it is attempted.

Table 24-2. Boot Lock Bit0 Protection Modes (Application Section)[1]

BLB0 Mode	BLB02	BLB01	Protection
1	1	1	No restrictions for SPM or LPM accessing the Application section.
2	1	0	SPM is not allowed to write to the Application section.
3	0	0	SPM is not allowed to write to the Application section, and LPM executing from the Boot Loader section is not allowed to read from the Application section. If Interrupt Vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
4	0	1	LPM executing from the Boot Loader section is not allowed to read from the Application section. If Interrupt Vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.

Note: 1. “1” means unprogrammed, “0” means programmed

Table 24-3. Boot Lock Bit1 Protection Modes (Boot Loader Section)[1]

BLB1 Mode	BLB12	BLB11	Protection
1	1	1	No restrictions for SPM or LPM accessing the Boot Loader section.
2	1	0	SPM is not allowed to write to the Boot Loader section.
3	0	0	SPM is not allowed to write to the Boot Loader section, and LPM executing from the Application section is not allowed to read from the Boot Loader section. If Interrupt Vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.
4	0	1	LPM executing from the Application section is not allowed to read from the Boot Loader section. If Interrupt Vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.

Note: 1. “1” means unprogrammed, “0” means programmed

24.6 Entering the Boot Loader Program

Entering the Boot Loader takes place by a jump or call from the application program. This may be initiated by a trigger such as a command received via USART, or SPI interface. Alternatively, the Boot Reset Fuse can be programmed so that the Reset Vector is pointing to the Boot Flash start address after a reset. In this case, the Boot Loader is started after a reset. After the application code is loaded, the program can start executing the application code. Note that the fuses cannot be changed by the MCU itself. This means that once the Boot Reset Fuse is programmed, the Reset Vector will always point to the Boot Loader Reset and the fuse can only be changed through the serial or parallel programming interface.

Table 24-4. Boot Reset Fuse[1]

BOOTRST	Reset Address
1	Reset Vector = Application Reset (address 0x0000)
0	Reset Vector = Boot Loader Reset (see Table 24-7 on page 289)

Note: 1. “1” means unprogrammed, “0” means programmed

24.7 Addressing the Flash During Self-Programming

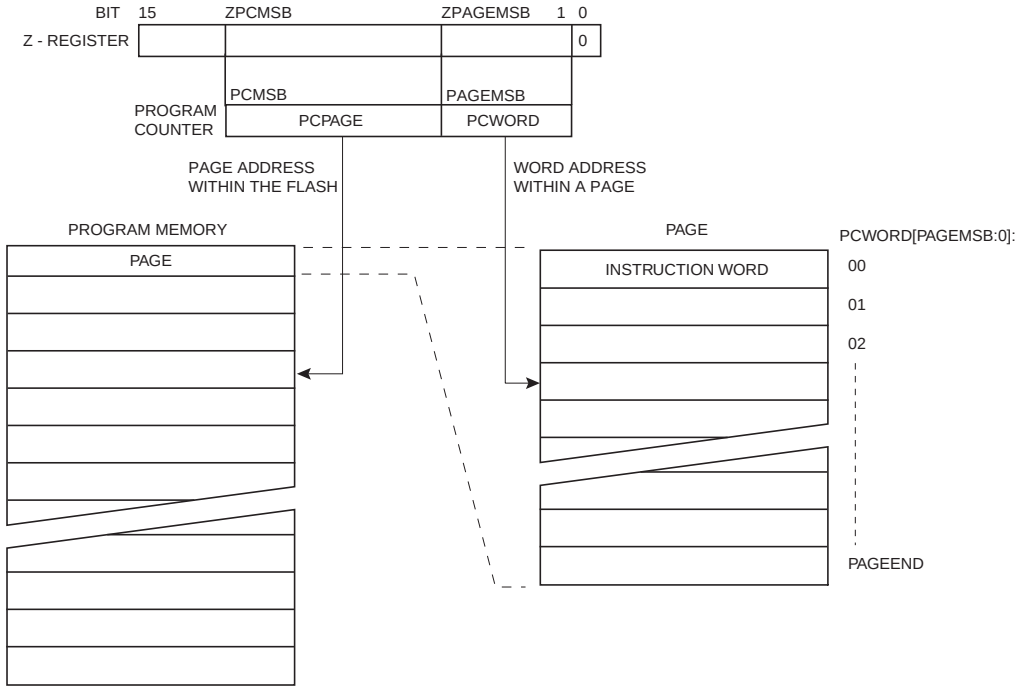
The Z-pointer is used to address the SPM commands.

Bit	15	14	13	12	11	10	9	8
ZH (R31)	Z15	Z14	Z13	Z12	Z11	Z10	Z9	Z8
ZL (R30)	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0
	7	6	5	4	3	2	1	0

Since the Flash is organized in pages (see Table 25-11 on page 299), the Program Counter can be treated as having two different sections. One section, consisting of the least significant bits, is addressing the words within a page, while the most significant bits are addressing the pages. This is1 shown in Figure 24-3. Note that the Page Erase and Page Write operations are addressed independently. Therefore it is of major importance that the Boot Loader software addresses the same page in both the Page Erase and Page Write operation. Once a programming operation is initiated, the address is latched and the Z-pointer can be used for other operations.

The only SPM operation that does not use the Z-pointer is Setting the Boot Loader Lock bits. The content of the Z-pointer is ignored and will have no effect on the operation. The LPM instruction does also use the Z-pointer to store the address. Since this instruction addresses the Flash byte-by-byte, also the LSB (bit Z0) of the Z-pointer is used.

Figure 24-3. Addressing the Flash During SPM(1)



Note: 1. The different variables used in Figure 24-3 are listed in Table 24-9 on page 289.

24.8 Self-Programming the Flash

The program memory is updated in a page by page fashion. Before programming a page with the data stored in the temporary page buffer, the page must be erased. The temporary page buffer is filled one word at a time using SPM and the buffer can be filled either before the Page Erase command or between a Page Erase and a Page Write operation:

Alternative 1, fill the buffer before a Page Erase

- Fill temporary page buffer
- Perform a Page Erase
- Perform a Page Write

Alternative 2, fill the buffer after Page Erase

- Perform a Page Erase
- Fill temporary page buffer
- Perform a Page Write

If only a part of the page needs to be changed, the rest of the page must be stored (for example in the temporary page buffer) before the erase, and then be rewritten. When using alternative 1, the Boot Loader provides an effective Read-Modify-Write feature which allows the user software to first read the page, do the necessary changes, and then write back the modified data. If alternative 2 is used, it is not possible to read the old data while loading since the page is already erased. The temporary page buffer can be accessed in a random sequence. It is essential that the page address used in both the Page Erase and Page Write operation is addressing the same page. See ”Simple Assembly Code Example for a Boot Loader” on page 286 for an assembly code example.

24.8.1 Performing Page Erase by SPM

To execute Page Erase, set up the address in the Z-pointer, write “X0000011” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE in the Z-register. Other bits in the Z-pointer will be ignored during this operation.

- Page Erase to the RWW section: The NRWW section can be read during the Page Erase.
- Page Erase to the NRWW section: The CPU is halted during the operation.

24.8.2 Filling the Temporary Buffer (Page Loading)

To write an instruction word, set up the address in the Z-pointer and data in R1:R0, write “00000001” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The content of PCWORD in the Z-register is used to address the data in the temporary buffer. The temporary buffer will auto-erase after a Page Write operation or by writing the RWWSRE bit in SPMCSR. It is also erased after a system reset. Note that it is not possible to write more than one time to each address without erasing the temporary buffer.

If the EEPROM is written in the middle of an SPM Page Load operation, all data loaded will be lost.

24.8.3 Performing a Page Write

To execute Page Write, set up the address in the Z-pointer, write “X0000101” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR. The data in R1 and R0 is ignored. The page address must be written to PCPAGE. Other bits in the Z-pointer must be written to zero during this operation.

- Page Write to the RWW section: The NRWW section can be read during the Page Write.
- Page Write to the NRWW section: The CPU is halted during the operation.

24.8.4 Using the SPM Interrupt

If the SPM interrupt is enabled, the SPM interrupt will generate a constant interrupt when the SELFPRGEN bit in SPMCSR is cleared. This means that the interrupt can be used instead of polling the SPMCSR Register in software. When using the SPM interrupt, the Interrupt Vectors should be moved to the BLS section to avoid that an interrupt is accessing the RWW section when it is blocked for reading. How to move the interrupts is described in ”Interrupts” on page 57.

24.8.5 Consideration While Updating BLS

Special care must be taken if the user allows the Boot Loader section to be updated by leaving Boot Lock bit11 unprogrammed. An accidental write to the Boot Loader itself can corrupt the entire Boot Loader, and further software updates might be impossible. If it is not necessary to change the Boot Loader software itself, it is recommended to program the Boot Lock bit11 to protect the Boot Loader software from any internal software changes.

24.8.6 Prevent Reading the RWW Section During Self-Programming

During Self-Programming (either Page Erase or Page Write), the RWW section is always blocked for reading. The user software itself must prevent that this section is addressed during the self programming operation. The RWWSB in the SPMCSR will be set as long as the RWW section is busy. During Self-Programming the Interrupt Vector table should be moved to the BLS as described in ”Watchdog Timer” on page 50, or the interrupts must be disabled. Before addressing the RWW section after the programming is completed, the user software must clear the RWWSB by writing the RWWSRE. See ”Simple Assembly Code Example for a Boot Loader” on page 286 for an example.

24.8.7 Setting the Boot Loader Lock Bits by SPM

To set the Boot Loader Lock bits and general Lock Bits, write the desired data to R0, write “X0001001” to SPMCSR and execute SPM within four clock cycles after writing SPMCSR.

Bit	7	6	5	4	3	2	1	0
R0	**1**	**1**	**BLB12**	**BLB11**	**BLB02**	**BLB01**	**LB2**	**LB1**

See Table 24-2 and Table 24-3 for how the different settings of the Boot Loader bits affect the Flash access.

If bits 5..0 in R0 are cleared (zero), the corresponding Lock bit will be programmed if an SPM instruction is executed within four cycles after BLBSET and SELFPRGEN are set in SPMCSR. The Z-pointer is don’t care during this operation, but for future compatibility it is recommended to load the Z-pointer with 0x0001 (same as used for reading the lO_{ck} bits). For future compatibility it is also recommended to set bits 7 and 6 in R0 to “1” when writing the Lock bits. When programming the Lock bits the entire Flash can be read during the operation.

24.8.8 EEPROM Write Prevents Writing to SPMCSR

Note that an EEPROM write operation will block all software programming to Flash. Reading the Fuses and Lock bits from software will also be prevented during the EEPROM write operation. It is recommended that the user checks the status bit (EEPE) in the EECR Register and verifies that the bit is cleared before writing to the SPMCSR Register.

24.8.9 Reading the Fuse and Lock Bits from Software

It is possible to read both the Fuse and Lock bits from software. To read the Lock bits, load the Z-pointer with 0x0001 and set the BLBSET and SELFPRGEN bits in SPMCSR. When an LPM

instruction is executed within three CPU cycles after the BLBSET and SELFPRGEN bits are set in SPMCSR, the value of the Lock bits will be loaded in the destination register. The BLBSET and SELFPRGEN bits will auto-clear upon completion of reading the Lock bits or if no LPM instruction is executed within three CPU cycles or no SPM instruction is executed within four CPU cycles. When BLBSET and SELFPRGEN are cleared, LPM will work as described in the Instruction set Manual.

Bit	7	6	5	4	3	2	1	0
Rd	–	–	BLB12	BLB11	BLB02	BLB01	LB2	LB1

The algorithm for reading the Fuse Low byte is similar to the one described above for reading the Lock bits. To read the Fuse Low byte, load the Z-pointer with 0x0000 and set the BLBSET and SELFPRGEN bits in SPMCSR. When an LPM instruction is executed within three cycles after the BLBSET and SELFPRGEN bits are set in the SPMCSR, the value of the Fuse Low byte (FLB) will be loaded in the destination register as shown below. Refer to Table 25-5 on page 296 for a detailed description and mapping of the Fuse Low byte.

Bit	7	6	5	4	3	2	1	0
Rd	FLB7	FLB6	FLB5	FLB4	FLB3	FLB2	FLB1	FLB0

Similarly, when reading the Fuse High byte, load 0x0003 in the Z-pointer. When an LPM instruction is executed within three cycles after the BLBSET and SELFPRGEN bits are set in the SPMCSR, the value of the Fuse High byte (FHB) will be loaded in the destination register as shown below. Refer to Table 25-7 on page 297 for detailed description and mapping of the Fuse High byte.

Bit	7	6	5	4	3	2	1	0
Rd	FHB7	FHB6	FHB5	FHB4	FHB3	FHB2	FHB1	FHB0

When reading the Extended Fuse byte, load 0x0002 in the Z-pointer. When an LPM instruction is executed within three cycles after the BLBSET and SELFPRGEN bits are set in the SPMCSR, the value of the Extended Fuse byte (EFB) will be loaded in the destination register as shown below. Refer to Table 25-4 on page 295 for detailed description and mapping of the Extended Fuse byte.

Bit	7	6	5	4	3	2	1	0
Rd	–	–	–	–	EFB3	EFB2	EFB1	EFB0

Fuse and Lock bits that are programmed, will be read as zero. Fuse and Lock bits that are unprogrammed, will be read as one.

24.8.10 Reading the Signature Row from Software

To read the Signature Row from software, load the Z-pointer with the signature byte address given in Table 24-5 on page 286 and set the SIGRD and SPMEN bits in SPMCSR. When an LPM instruction is executed within three CPU cycles after the SIGRD and SPMEN bits are set in SPMCSR, the signature byte value will be loaded in the destination register. The SIGRD and SPMEN bits will auto-clear upon completion of reading the Signature Row Lock bits or if no LPM instruction is executed within three CPU cycles. When SIGRD and SPMEN are cleared, LPM will work as described in the Instruction set Manual.

Table 24-5. Signature Row Addressing

Signature Byte	Z-Pointer Address
Device Signature Byte 1	0x0000
Device Signature Byte 2	0x0002
Device Signature Byte 3	0x0004
RC Oscillator Calibration Byte	0x0001

Note: All other addresses are reserved for future use.

24.8.11 Preventing Flash Corruption

During periods of low V_{CC}, the Flash program can be corrupted because the supply voltage is too low for the CPU and the Flash to operate properly. These issues are the same as for board level systems using the Flash, and the same design solutions should be applied.

A Flash program corruption can be caused by two situations when the voltage is too low. First, a regular write sequence to the Flash requires a minimum voltage to operate correctly. Secondly, the CPU itself can execute instructions incorrectly, if the supply voltage for executing instructions is too low.

Flash corruption can easily be avoided by following these design recommendations (one is sufficient):

1. If there is no need for a Boot Loader update in the system, program the Boot Loader Lock bits to prevent any Boot Loader software updates.
2. Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal Brown-out Detector (BOD) if the operating voltage matches the detection level. If not, an external low V_{CC} reset protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided that the power supply voltage is sufficient.
3. Keep the AVR core in Power-down sleep mode during periods of low V_{CC}. This will prevent the CPU from attempting to decode and execute instructions, effectively protecting the SPMCSR Register and thus the Flash from unintentional writes.

24.8.12 Programming Time for Flash when Using SPM

The calibrated RC Oscillator is used to time Flash accesses. Table 24-6 shows the typical programming time for Flash accesses from the CPU.

Table 24-6. SPM Programming Time(1)

Symbol	Min Programming Time	Max Programming Time
Flash write (Page Erase, Page Write, and write Lock bits by SPM)	3.7 ms	4.5 ms

Note: 1. Minimum and maximum programming time is per individual operation.

24.8.13 Simple Assembly Code Example for a Boot Loader

```
;-the routine writes one page of data from RAM to Flash
; the first data location in RAM is pointed to by the Y pointer
; the first data location in Flash is pointed to by the Z-pointer
;-error handling is not included
```

```
  ;-the routine must be placed inside the Boot space
  ; (at least the Do_spm sub routine). Only code inside NRWW section can
  ; be read during Self-Programming (Page Erase and Page Write).
  ;-registers used: r0, r1, temp1 (r16), temp2 (r17), looplo (r24),
  ; loophi (r25), spmcrval (r20)
  ; storing and restoring of registers is not included in the routine
  ; register usage can be optimized at the expense of code size
  ;-It is assumed that either the interrupt table is moved to the Boot
  ; loader section or that the interrupts are disabled.
.equ PAGESIZEB = PAGESIZE*2   ;PAGESIZEB is page size in BYTES, not words
.org SMALLBOOTSTART
Write_page:
  ; Page Erase
  ldi  spmcrval, (1<<PGERS) | (1<<SELFPRGEN)
  call Do_spm

  ; re-enable the RWW section
  ldi  spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
  call Do_spm

  ; transfer data from RAM to Flash page buffer
  ldi  looplo, low(PAGESIZEB)   ;init loop variable
  ldi  loophi, high(PAGESIZEB)  ;not required for PAGESIZEB<=256
Wrloop:
  ld   r0, Y+
  ld   r1, Y+
  ldi  spmcrval, (1<<SELFPRGEN)
  call Do_spm
  adiw ZH:ZL, 2
  sbiw loophi:looplo, 2         ;use subi for PAGESIZEB<=256
  brne Wrloop

  ; execute Page Write
  subi ZL, low(PAGESIZEB)       ;restore pointer
  sbci ZH, high(PAGESIZEB)      ;not required for PAGESIZEB<=256
  ldi  spmcrval, (1<<PGWRT) | (1<<SELFPRGEN)
  call Do_spm

  ; re-enable the RWW section
  ldi  spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
  call Do_spm

  ; read back and check, optional
  ldi  looplo, low(PAGESIZEB)   ;init loop variable
  ldi  loophi, high(PAGESIZEB)  ;not required for PAGESIZEB<=256
  subi YL, low(PAGESIZEB)       ;restore pointer
  sbci YH, high(PAGESIZEB)
Rdloop:
  lpm  r0, Z+
  ld   r1, Y+
  cpse r0, r1
  jmp  Error
  sbiw loophi:looplo, 1         ;use subi for PAGESIZEB<=256
  brne Rdloop

  ; return to RWW section
  ; verify that RWW section is safe to read
Return:
  in   temp1, SPMCSR
```

```
  sbrs temp1, RWWSB     ; If RWWSB is set, the RWW section is not ready yet
  ret
  ; re-enable the RWW section
  ldi  spmcrval, (1<<RWWSRE) | (1<<SELFPRGEN)
  call Do_spm
  rjmp Return

Do_spm:
  ; check for previous SPM complete
Wait_spm:
  in   temp1, SPMCSR
  sbrc temp1, SELFPRGEN
  rjmp Wait_spm
  ; input: spmcrval determines SPM action
  ; disable interrupts if enabled, store status
  in   temp2, SREG
  cli
  ; check that no EEPROM write access is present
Wait_ee:
  sbic EECR, EEPE
  rjmp Wait_ee
  ; SPM timed sequence
  out  SPMCSR, spmcrval
  spm
  ; restore SREG (to enable interrupts if originally enabled)
  out  SREG, temp2
  ret
```

24.8.14 ATmega88P Boot Loader Parameters

In Table 24-7 through Table 24-9, the parameters used in the description of the self programming are given.

Table 24-7. Boot Size Configuration, ATmega88P

BOOTSZ1	BOOTSZ0	Boot Size	Pages	Application Flash Section	Boot Loader Flash Section	End Application Section	Boot Reset Address (Start Boot Loader Section)
1	1	128 words	4	0x000 - 0xF7F	0xF80 - 0xFFF	0xF7F	0xF80
1	0	256 words	8	0x000 - 0xEFF	0xF00 - 0xFFF	0xEFF	0xF00
0	1	512 words	16	0x000 - 0xDFF	0xE00 - 0xFFF	0xDFF	0xE00
0	0	1024 words	32	0x000 - 0xBFF	0xC00 - 0xFFF	0xBFF	0xC00

Note: The different BOOTSZ Fuse configurations are shown in Figure 24-2 on page 280.

Table 24-8. Read-While-Write Limit, ATmega88P

Section	Pages	Address
Read-While-Write section (RWW)	96	0x000 - 0xBFF
No Read-While-Write section (NRWW)	32	0xC00 - 0xFFF

For details about these two section, see "NRWW – No Read-While-Write Section" on page 278 and "RWW – Read-While-Write Section" on page 278

Table 24-9. Explanation of Different Variables used in Figure 24-3 and the Mapping to the Z-pointer, ATmega88P

Variable		Corresponding Z-value[1]	Description
PCMSB	11		Most significant bit in the Program Counter. (The Program Counter is 12 bits PC[11:0])
PAGEMSB	4		Most significant bit which is used to address the words within one page (32 words in a page requires 5 bits PC [4:0]).
ZPCMSB		Z12	Bit in Z-register that is mapped to PCMSB. Because Z0 is not used, the ZPCMSB equals PCMSB + 1.
ZPAGEMSB		Z5	Bit in Z-register that is mapped to PAGEMSB. Because Z0 is not used, the ZPAGEMSB equals PAGEMSB + 1.
PCPAGE	PC[11:5]	Z12:Z6	Program counter page address: Page select, for page erase and page write
PCWORD	PC[4:0]	Z5:Z1	Program counter word address: Word select, for filling temporary buffer (must be zero during page write operation)

Note: 1. Z15:Z13: always ignored
Z0: should be zero for all SPM commands, byte select for the LPM instruction.
See "Addressing the Flash During Self-Programming" on page 282 for details about the use of Z-pointer during Self-Programming.

24.8.15 ATmega168P Boot Loader Parameters

In Table 24-10 through Table 24-12, the parameters used in the description of the self programming are given.

Table 24-10. Boot Size Configuration, ATmega168P

BOOTSZ1	BOOTSZ0	Boot Size	Pages	Application Flash Section	Boot Loader Flash Section	End Application Section	Boot Reset Address (Start Boot Loader Section)
1	1	128 words	2	0x0000 - 0x1F7F	0x1F80 - 0x1FFF	0x1F7F	0x1F80
1	0	256 words	4	0x0000 - 0x1EFF	0x1F00 - 0x1FFF	0x1EFF	0x1F00
0	1	512 words	8	0x0000 - 0x1DFF	0x1E00 - 0x1FFF	0x1DFF	0x1E00
0	0	1024 words	16	0x0000 - 0x1BFF	0x1C00 - 0x1FFF	0x1BFF	0x1C00

Note: The different BOOTSZ Fuse configurations are shown in Figure 24-2 on page 280.

Table 24-11. Read-While-Write Limit, ATmega168P

Section	Pages	Address
Read-While-Write section (RWW)	112	0x0000 - 0x1BFF
No Read-While-Write section (NRWW)	16	0x1C00 - 0x1FFF

For details about these two section, see "NRWW – No Read-While-Write Section" on page 278 and "RWW – Read-While-Write Section" on page 278

Table 24-12. Explanation of Different Variables used in Figure 24-3 and the Mapping to the Z-pointer, ATmega168P

Variable		Corresponding Z-value[1]	Description
PCMSB	12		Most significant bit in the Program Counter. (The Program Counter is 13 bits PC[12:0])
PAGEMSB	5		Most significant bit which is used to address the words within one page (64 words in a page requires 6 bits PC [5:0])
ZPCMSB		Z13	Bit in Z-register that is mapped to PCMSB. Because Z0 is not used, the ZPCMSB equals PCMSB + 1.
ZPAGEMSB		Z6	Bit in Z-register that is mapped to PAGEMSB. Because Z0 is not used, the ZPAGEMSB equals PAGEMSB + 1.
PCPAGE	PC[12:6]	Z13:Z7	Program counter page address: Page select, for page erase and page write
PCWORD	PC[5:0]	Z6:Z1	Program counter word address: Word select, for filling temporary buffer (must be zero during page write operation)

Note: 1. Z15:Z14: always ignored
Z0: should be zero for all SPM commands, byte select for the LPM instruction.
See "Addressing the Flash During Self-Programming" on page 282 for details about the use of Z-pointer during Self-Programming.

24.8.16 ATmega328P Boot Loader Parameters

In Table 24-13 through Table 24-15, the parameters used in the description of the self programming are given.

Table 24-13. Boot Size Configuration, ATmega328P

BOOTSZ1	BOOTSZ0	Boot Size	Pages	Application Flash Section	Boot Loader Flash Section	End Application Section	Boot Reset Address (Start Boot Loader Section)
1	1	256 words	4	0x0000 - 0x3EFF	0x3F00 - 0x3FFF	0x3EFF	0x3F00
1	0	512 words	8	0x0000 - 0x3DFF	0x3E00 - 0x3FFF	0x3DFF	0x3E00
0	1	1024 words	16	0x0000 - 0x3BFF	0x3C00 - 0x3FFF	0x3BFF	0x3C00
0	0	2048 words	32	0x0000 - 0x37FF	0x3800 - 0x3FFF	0x37FF	0x3800

Note: The different BOOTSZ Fuse configurations are shown in Figure 24-2 on page 280.

Table 24-14. Read-While-Write Limit, ATmega328P

Section	Pages	Address
Read-While-Write section (RWW)	224	0x0000 - 0x37FF
No Read-While-Write section (NRWW)	32	0x3800 - 0x3FFF

For details about these two section, see "NRWW – No Read-While-Write Section" on page 278 and "RWW – Read-While-Write Section" on page 278

Table 24-15. Explanation of Different Variables used in Figure 24-3 and the Mapping to the Z-pointer, ATmega328P

Variable		Corresponding Z-value[1]	Description
PCMSB	13		Most significant bit in the Program Counter. (The Program Counter is 14 bits PC[13:0])
PAGEMSB	5		Most significant bit which is used to address the words within one page (64 words in a page requires 6 bits PC [5:0])
ZPCMSB		Z14	Bit in Z-register that is mapped to PCMSB. Because Z0 is not used, the ZPCMSB equals PCMSB + 1.
ZPAGEMSB		Z6	Bit in Z-register that is mapped to PAGEMSB. Because Z0 is not used, the ZPAGEMSB equals PAGEMSB + 1.
PCPAGE	PC[13:6]	Z14:Z7	Program counter page address: Page select, for page erase and page write
PCWORD	PC[5:0]	Z6:Z1	Program counter word address: Word select, for filling temporary buffer (must be zero during page write operation)

Note: 1. Z15: always ignored
Z0: should be zero for all SPM commands, byte select for the LPM instruction.
See "Addressing the Flash During Self-Programming" on page 282 for details about the use of Z-pointer during Self-Programming.

24.9 Register Description

24.9.1 SPMCSR – Store Program Memory Control and Status Register

The Store Program Memory Control and Status Register contains the control bits needed to control the Boot Loader operations.

Bit	7	6	5	4	3	2	1	0	
0x37 (0x57)	SPMIE	RWWSB	–	RWWSRE	BLBSET	PGWRT	PGERS	SELFPRGEN	SPMCSR
Read/Write	R/W	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – SPMIE: SPM Interrupt Enable**

When the SPMIE bit is written to one, and the I-bit in the Status Register is set (one), the SPM ready interrupt will be enabled. The SPM ready Interrupt will be executed as long as the SELFPRGEN bit in the SPMCSR Register is cleared.

- **Bit 6 – RWWSB: Read-While-Write Section Busy**

When a Self-Programming (Page Erase or Page Write) operation to the RWW section is initiated, the RWWSB will be set (one) by hardware. When the RWWSB bit is set, the RWW section cannot be accessed. The RWWSB bit will be cleared if the RWWSRE bit is written to one after a Self-Programming operation is completed. Alternatively the RWWSB bit will automatically be cleared if a page load operation is initiated.

- **Bit 5 – Res: Reserved Bit**

This bit is a reserved bit in the ATmega48P/88P/168P/328P and always read as zero.

- **Bit 4 – RWWSRE: Read-While-Write Section Read Enable**

When programming (Page Erase or Page Write) to the RWW section, the RWW section is blocked for reading (the RWWSB will be set by hardware). To re-enable the RWW section, the user software must wait until the programming is completed (SELFPRGEN will be cleared). Then, if the RWWSRE bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles re-enables the RWW section. The RWW section cannot be re-enabled while the Flash is busy with a Page Erase or a Page Write (SELFPRGEN is set). If the RWWSRE bit is written while the Flash is being loaded, the Flash load operation will abort and the data loaded will be lost.

- **Bit 3 – BLBSET: Boot Lock Bit Set**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles sets Boot Lock bits and Memory Lock bits, according to the data in R0. The data in R1 and the address in the Z-pointer are ignored. The BLBSET bit will automatically be cleared upon completion of the Lock bit set, or if no SPM instruction is executed within four clock cycles.

An LPM instruction within three cycles after BLBSET and SELFPRGEN are set in the SPMCSR Register, will read either the Lock bits or the Fuse bits (depending on Z0 in the Z-pointer) into the destination register. See ”Reading the Fuse and Lock Bits from Software” on page 284 for details.

- **Bit 2 – PGWRT: Page Write**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles executes Page Write, with the data stored in the temporary buffer. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The

PGWRT bit will auto-clear upon completion of a Page Write, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire Page Write operation if the NRWW section is addressed.

- **Bit 1 – PGERS: Page Erase**

If this bit is written to one at the same time as SELFPRGEN, the next SPM instruction within four clock cycles executes Page Erase. The page address is taken from the high part of the Z-pointer. The data in R1 and R0 are ignored. The PGERS bit will auto-clear upon completion of a Page Erase, or if no SPM instruction is executed within four clock cycles. The CPU is halted during the entire Page Write operation if the NRWW section is addressed.

- **Bit 0 – SELFPRGEN: Self Programming Enable**

This bit enables the SPM instruction for the next four clock cycles. If written to one together with either RWWSRE, BLBSET, PGWRT or PGERS, the following SPM instruction will have a special meaning, see description above. If only SELFPRGEN is written, the following SPM instruction will store the value in R1:R0 in the temporary page buffer addressed by the Z-pointer. The LSB of the Z-pointer is ignored. The SELFPRGEN bit will auto-clear upon completion of an SPM instruction, or if no SPM instruction is executed within four clock cycles. During Page Erase and Page Write, the SELFPRGEN bit remains high until the operation is completed.

Writing any other combination than “10001”, “01001”, “00101”, “00011” or “00001” in the lower five bits will have no effect.

25. Memory Programming

25.1 Program And Data Memory Lock Bits

The ATmega88P/168P/328P provides six Lock bits which can be left unprogrammed (“1”) or can be programmed (“0”) to obtain the additional features listed in Table 25-2. The Lock bits can only be erased to “1” with the Chip Erase command.The ATmega48P has no separate Boot Loader section. The SPM instruction is enabled for the whole Flash if the SELFPRGEN fuse is programmed (“0”), otherwise it is disabled.

Table 25-1. Lock Bit Byte[1]

Lock Bit Byte	Bit No	Description	Default Value
	7	–	1 (unprogrammed)
	6	–	1 (unprogrammed)
BLB12[2]	5	Boot Lock bit	1 (unprogrammed)
BLB11[2]	4	Boot Lock bit	1 (unprogrammed)
BLB02[2]	3	Boot Lock bit	1 (unprogrammed)
BLB01[2]	2	Boot Lock bit	1 (unprogrammed)
LB2	1	Lock bit	1 (unprogrammed)
LB1	0	Lock bit	1 (unprogrammed)

Notes: 1. “1” means unprogrammed, “0” means programmed
2. Only on ATmega88P/168P/328P.

Table 25-2. Lock Bit Protection Modes[1][2]

Memory Lock Bits			Protection Type
LB Mode	LB2	LB1	
1	1	1	No memory lock features enabled.
2	1	0	Further programming of the Flash and EEPROM is disabled in Parallel and Serial Programming mode. The Fuse bits are locked in both Serial and Parallel Programming mode.[1]
3	0	0	Further programming and verification of the Flash and EEPROM is disabled in Parallel and Serial Programming mode. The Boot Lock bits and Fuse bits are locked in both Serial and Parallel Programming mode.[1]

Notes: 1. Program the Fuse bits and Boot Lock bits before programming the LB1 and LB2.
2. “1” means unprogrammed, “0” means programmed

Table 25-3. Lock Bit Protection Modes[1][2]. Only ATmega88P/168P/328P.

BLB0 Mode	BLB02	BLB01	
1	1	1	No restrictions for SPM or LPM accessing the Application section.
2	1	0	SPM is not allowed to write to the Application section.
3	0	0	SPM is not allowed to write to the Application section, and LPM executing from the Boot Loader section is not allowed to read from the Application section. If Interrupt Vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
4	0	1	LPM executing from the Boot Loader section is not allowed to read from the Application section. If Interrupt Vectors are placed in the Boot Loader section, interrupts are disabled while executing from the Application section.
BLB1 Mode	**BLB12**	**BLB11**	
1	1	1	No restrictions for SPM or LPM accessing the Boot Loader section.
2	1	0	SPM is not allowed to write to the Boot Loader section.
3	0	0	SPM is not allowed to write to the Boot Loader section, and LPM executing from the Application section is not allowed to read from the Boot Loader section. If Interrupt Vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.
4	0	1	LPM executing from the Application section is not allowed to read from the Boot Loader section. If Interrupt Vectors are placed in the Application section, interrupts are disabled while executing from the Boot Loader section.

Notes: 1. Program the Fuse bits and Boot Lock bits before programming the LB1 and LB2.
2. “1” means unprogrammed, “0” means programmed

25.2 Fuse Bits

The ATmega48P/88P/168P/328P has three Fuse bytes. Table 25-4 - Table 25-9 describe briefly the functionality of all the fuses and how they are mapped into the Fuse bytes. Note that the fuses are read as logical zero, “0”, if they are programmed.

Table 25-4. Extended Fuse Byte for ATmega48P

Extended Fuse Byte	Bit No	Description	Default Value
–	7	–	1
–	6	–	1
–	5	–	1
–	4	–	1
–	3	–	1
–	2	–	1
–	1	–	1
SELFPRGEN	0	Self Programming Enable	1 (unprogrammed)

Table 25-5. Extended Fuse Byte for ATmega88P/168P

Extended Fuse Byte	Bit No	Description	Default Value
–	7	–	1
–	6	–	1
–	5	–	1
–	4	–	1
–	3	–	1
BOOTSZ1	2	Select Boot Size (see Table 24-7 on page 289, Table 24-10 on page 290 and Table 24-13 on page 291 for details)	0 (programmed)(1)
BOOTSZ0	1	Select Boot Size (see Table 24-7 on page 289, Table 24-10 on page 290 and Table 24-13 on page 291 for details)	0 (programmed)(1)
BOOTRST	0	Select Reset Vector	1 (unprogrammed)

Note: 1. The default value of BOOTSZ[1:0] results in maximum Boot Size. See "Pin Name Mapping" on page 300.

Table 25-6. Extended Fuse Byte for ATmega328P

Extended Fuse Byte	Bit No	Description	Default Value
–	7	–	1
–	6	–	1
–	5	–	1
–	4	–	1
–	3	–	1
BODLEVEL2(1)	2	Brown-out Detector trigger level	1 (unprogrammed)
BODLEVEL1(1)	1	Brown-out Detector trigger level	1 (unprogrammed)
BODLEVEL0(1)	0	Brown-out Detector trigger level	1 (unprogrammed)

Note: 1. See Table 26-4 on page 320 for BODLEVEL Fuse decoding..

Table 25-7. Fuse High Byte for ATmega48P/88P/168P

High Fuse Byte	Bit No	Description	Default Value
RSTDISBL[1]	7	External Reset Disable	1 (unprogrammed)
DWEN	6	debugWIRE Enable	1 (unprogrammed)
SPIEN[2]	5	Enable Serial Program and Data Downloading	0 (programmed, SPI programming enabled)
WDTON[3]	4	Watchdog Timer Always On	1 (unprogrammed)
EESAVE	3	EEPROM memory is preserved through the Chip Erase	1 (unprogrammed), EEPROM not reserved
BODLEVEL2[4]	2	Brown-out Detector trigger level	1 (unprogrammed)
BODLEVEL1[4]	1	Brown-out Detector trigger level	1 (unprogrammed)
BODLEVEL0[4]	0	Brown-out Detector trigger level	1 (unprogrammed)

Notes: 1. See "Alternate Functions of Port C" on page 85 for description of RSTDISBL Fuse.
2. The SPIEN Fuse is not accessible in serial programming mode.
3. See "WDTCSR – Watchdog Timer Control Register" on page 54 for details.
4. See Table 26-4 on page 320 for BODLEVEL Fuse decoding.

Table 25-8. Fuse High Byte for ATmega328P

High Fuse Byte	Bit No	Description	Default Value
RSTDISBL[1]	7	External Reset Disable	1 (unprogrammed)
DWEN	6	debugWIRE Enable	1 (unprogrammed)
SPIEN[2]	5	Enable Serial Program and Data Downloading	0 (programmed, SPI programming enabled)
WDTON[3]	4	Watchdog Timer Always On	1 (unprogrammed)
EESAVE	3	EEPROM memory is preserved through the Chip Erase	1 (unprogrammed), EEPROM not reserved
BOOTSZ1	2	Select Boot Size (see Table 24-7 on page 289, Table 24-10 on page 290 and Table 24-13 on page 291 for details)	0 (programmed)[4]
BOOTSZ0	1	Select Boot Size (see Table 24-7 on page 289, Table 24-10 on page 290 and Table 24-13 on page 291 for details)	0 (programmed)[4]
BOOTRST	0	Select Reset Vector	1 (unprogrammed)

Notes: 1. See "Alternate Functions of Port C" on page 85 for description of RSTDISBL Fuse.

2. The SPIEN Fuse is not accessible in serial programming mode.
3. See "WDTCSR – Watchdog Timer Control Register" on page 54 for details.
4. The default value of BOOTSZ[1:0] results in maximum Boot Size. See "Pin Name Mapping" on page 300.

Table 25-9. Fuse Low Byte

Low Fuse Byte	Bit No	Description	Default Value
CKDIV8(4)	7	Divide clock by 8	0 (programmed)
CKOUT(3)	6	Clock output	1 (unprogrammed)
SUT1	5	Select start-up time	1 (unprogrammed)(1)
SUT0	4	Select start-up time	0 (programmed)(1)
CKSEL3	3	Select Clock source	0 (programmed)(2)
CKSEL2	2	Select Clock source	0 (programmed)(2)
CKSEL1	1	Select Clock source	1 (unprogrammed)(2)
CKSEL0	0	Select Clock source	0 (programmed)(2)

Note:
1. The default value of SUT1..0 results in maximum start-up time for the default clock source. See Table 6-11 on page 33 for details.
2. The default setting of CKSEL3..0 results in internal RC Oscillator @ 8 MHz. See Table 6-10 on page 33 for details.
3. The CKOUT Fuse allows the system clock to be output on PORTB0. See "Clock Output Buffer" on page 35 for details.
4. See "System Clock Prescaler" on page 35 for details.

The status of the Fuse bits is not affected by Chip Erase. Note that the Fuse bits are locked if Lock bit1 (LB1) is programmed. Program the Fuse bits before programming the Lock bits.

25.2.1 Latching of Fuses

The fuse values are latched when the device enters programming mode and changes of the fuse values will have no effect until the part leaves Programming mode. This does not apply to the EESAVE Fuse which will take effect once it is programmed. The fuses are also latched on Power-up in Normal mode.

25.3 Signature Bytes

All Atmel microcontrollers have a three-byte signature code which identifies the device. This code can be read in both serial and parallel mode, also when the device is locked. The three bytes reside in a separate address space. For the ATmega48P/88P/168P/328P the signature bytes are given in Table 25-10.

Table 25-10. Device ID

	Signature Bytes Address		
Part	**0x000**	**0x001**	**0x002**
ATmega48P	0x1E	0x92	0x0A
ATmega88P	0x1E	0x93	0x0F
ATmega168P	0x1E	0x94	0x0B
ATmega328P	0x1E	0x95	0x0F

25.4 Calibration Byte

The ATmega48P/88P/168P/328P has a byte calibration value for the Internal RC Oscillator. This byte resides in the high byte of address 0x000 in the signature address space. During reset, this byte is automatically written into the OSCCAL Register to ensure correct frequency of the calibrated RC Oscillator.

25.5 Page Size

Table 25-11. No. of Words in a Page and No. of Pages in the Flash

Device	Flash Size	Page Size	PCWORD	No. of Pages	PCPAGE	PCMSB
ATmega48P	2K words (4K bytes)	32 words	PC[4:0]	64	PC[10:5]	10
ATmega88P	4K words (8K bytes)	32 words	PC[4:0]	128	PC[11:5]	11
ATmega168P	8K words (16K bytes)	64 words	PC[5:0]	128	PC[12:6]	12
ATmega328P	16K words (32K bytes)	64 words	PC[5:0]	256	PC[13:6]	13

Table 25-12. No. of Words in a Page and No. of Pages in the EEPROM

Device	EEPROM Size	Page Size	PCWORD	No. of Pages	PCPAGE	EEAMSB
ATmega48P	256 bytes	4 bytes	EEA[1:0]	64	EEA[7:2]	7
ATmega88P	512 bytes	4 bytes	EEA[1:0]	128	EEA[8:2]	8
ATmega168P	512 bytes	4 bytes	EEA[1:0]	128	EEA[8:2]	8
ATmega328P	1K bytes	4 bytes	EEA[1:0]	256	EEA[9:2]	9

25.6 Parallel Programming Parameters, Pin Mapping, and Commands

This section describes how to parallel program and verify Flash Program memory, EEPROM Data memory, Memory Lock bits, and Fuse bits in the ATmega48P/88P/168P/328P. Pulses are assumed to be at least 250 ns unless otherwise noted.

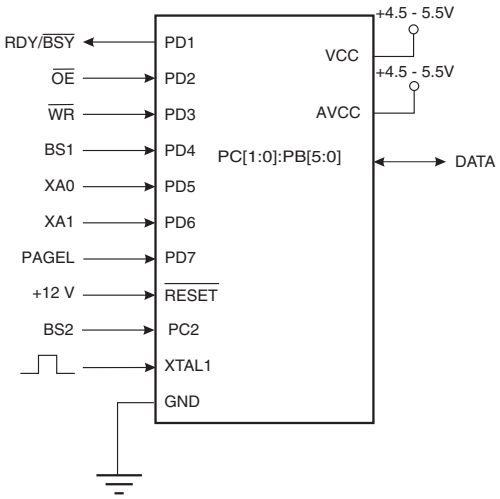
25.6.1 Signal Names

In this section, some pins of the ATmega48P/88P/168P/328P are referenced by signal names describing their functionality during parallel programming, see Figure 25-1 and Table 25-13. Pins not described in the following table are referenced by pin names.

The XA1/XA0 pins determine the action executed when the XTAL1 pin is given a positive pulse. The bit coding is shown in Table 25-15.

When pulsing $\overline{WR}$ or $\overline{OE}$, the command loaded determines the action executed. The different Commands are shown in Table 25-16.

Figure 25-1. Parallel Programming



Note: V_{CC} - 0.3V < AV_{CC} < V_{CC} + 0.3V, however, AV_{CC} should always be within 4.5 - 5.5V

Table 25-13. Pin Name Mapping

Signal Name in Programming Mode	Pin Name	I/O	Function
RDY/$\overline{BSY}$	PD1	O	0: Device is busy programming, 1: Device is ready for new command
$\overline{OE}$	PD2	I	Output Enable (Active low)
$\overline{WR}$	PD3	I	Write Pulse (Active low)
BS1	PD4	I	Byte Select 1 (“0” selects Low byte, “1” selects High byte)
XA0	PD5	I	XTAL Action Bit 0
XA1	PD6	I	XTAL Action Bit 1
PAGEL	PD7	I	Program memory and EEPROM Data Page Load
BS2	PC2	I	Byte Select 2 (“0” selects Low byte, “1” selects 2’nd High byte)
DATA	{PC[1:0]: PB[5:0]}	I/O	Bi-directional Data bus (Output when $\overline{OE}$ is low)

Table 25-14. Pin Values Used to Enter Programming Mode

Pin	Symbol	Value
PAGEL	Prog_enable[3]	0
XA1	Prog_enable[2]	0
XA0	Prog_enable[1]	0
BS1	Prog_enable[0]	0

Table 25-15. XA1 and XA0 Coding

XA1	XA0	Action when XTAL1 is Pulsed
0	0	Load Flash or EEPROM Address (High or low address byte determined by BS1).
0	1	Load Data (High or Low data byte for Flash determined by BS1).
1	0	Load Command
1	1	No Action, Idle

Table 25-16. Command Byte Bit Coding

Command Byte	Command Executed
1000 0000	Chip Erase
0100 0000	Write Fuse bits
0010 0000	Write Lock bits
0001 0000	Write Flash
0001 0001	Write EEPROM
0000 1000	Read Signature Bytes and Calibration byte
0000 0100	Read Fuse and Lock bits
0000 0010	Read Flash
0000 0011	Read EEPROM

25.7 Parallel Programming

25.7.1 Enter Programming Mode

The following algorithm puts the device in Parallel (High-voltage) Programming mode:

1. Set Prog_enable pins listed in Table 25-14 on page 300 to “0000”, RESET pin to 0V and V_{CC} to 0V.
2. Apply 4.5 - 5.5V between V_{CC} and GND.

Ensure that V_{CC} reaches at least 1.8V within the next 20 µs.

3. Wait 20 - 60 µs, and apply 11.5 - 12.5V to RESET.
4. Keep the Prog_enable pins unchanged for at least 10µs after the High-voltage has been applied to ensure the Prog_enable Signature has been latched.
5. Wait at least 300 µs before giving any parallel programming commands.
6. Exit Programming mode by power the device down or by bringing RESET pin to 0V.

If the rise time of the V_{CC} is unable to fulfill the requirements listed above, the following alternative algorithm can be used.

1. Set Prog_enable pins listed in Table 25-14 on page 300 to “0000”, RESET pin to 0V and V_{CC} to 0V.
2. Apply 4.5 - 5.5V between V_{CC} and GND.
3. Monitor V_{CC}, and as soon as V_{CC} reaches 0.9 - 1.1V, apply 11.5 - 12.5V to RESET.

4. Keep the Prog_enable pins unchanged for at least 10µs after the High-voltage has been applied to ensure the Prog_enable Signature has been latched.
5. Wait until V_{CC} actually reaches 4.5 -5.5V before giving any parallel programming commands.
6. Exit Programming mode by power the device down or by bringing RESET pin to 0V.

25.7.2 Considerations for Efficient Programming

The loaded command and address are retained in the device during programming. For efficient programming, the following should be considered.

- The command needs only be loaded once when writing or reading multiple memory locations.
- Skip writing the data value 0xFF, that is the contents of the entire EEPROM (unless the EESAVE Fuse is programmed) and Flash after a Chip Erase.
- Address high byte needs only be loaded before programming or reading a new 256 word window in Flash or 256 byte EEPROM. This consideration also applies to Signature bytes reading.

25.7.3 Chip Erase

The Chip Erase will erase the Flash and EEPROM[1] memories plus Lock bits. The Lock bits are not reset until the program memory has been completely erased. The Fuse bits are not changed. A Chip Erase must be performed before the Flash and/or EEPROM are reprogrammed.

Note: 1. The EEPRPOM memory is preserved during Chip Erase if the EESAVE Fuse is programmed.

Load Command “Chip Erase”

1. Set XA1, XA0 to “10”. This enables command loading.
2. Set BS1 to “0”.
3. Set DATA to “1000 0000”. This is the command for Chip Erase.
4. Give XTAL1 a positive pulse. This loads the command.
5. Give $\overline{WR}$ a negative pulse. This starts the Chip Erase. RDY/$\overline{BSY}$ goes low.
6. Wait until RDY/$\overline{BSY}$ goes high before loading a new command.

25.7.4 Programming the Flash

The Flash is organized in pages, see Table 25-11 on page 299. When programming the Flash, the program data is latched into a page buffer. This allows one page of program data to be programmed simultaneously. The following procedure describes how to program the entire Flash memory:

A. Load Command “Write Flash”

1. Set XA1, XA0 to “10”. This enables command loading.
2. Set BS1 to “0”.
3. Set DATA to “0001 0000”. This is the command for Write Flash.
4. Give XTAL1 a positive pulse. This loads the command.

B. Load Address Low byte

1. Set XA1, XA0 to “00”. This enables address loading.
2. Set BS1 to “0”. This selects low address.
3. Set DATA = Address low byte (0x00 - 0xFF).

4. Give XTAL1 a positive pulse. This loads the address low byte.

C. Load Data Low Byte

1. Set XA1, XA0 to “01”. This enables data loading.
2. Set DATA = Data low byte (0x00 - 0xFF).
3. Give XTAL1 a positive pulse. This loads the data byte.

D. Load Data High Byte

1. Set BS1 to “1”. This selects high data byte.
2. Set XA1, XA0 to “01”. This enables data loading.
3. Set DATA = Data high byte (0x00 - 0xFF).
4. Give XTAL1 a positive pulse. This loads the data byte.

E. Latch Data

1. Set BS1 to “1”. This selects high data byte.
2. Give PAGEL a positive pulse. This latches the data bytes. (See Figure 25-3 for signal waveforms)

F. Repeat B through E until the entire buffer is filled or until all data within the page is loaded.

While the lower bits in the address are mapped to words within the page, the higher bits address the pages within the FLASH. This is illustrated in Figure 25-2 on page 304. Note that if less than eight bits are required to address words in the page (pagesize < 256), the most significant bit(s) in the address low byte are used to address the page when performing a Page Write.

G. Load Address High byte

1. Set XA1, XA0 to “00”. This enables address loading.
2. Set BS1 to “1”. This selects high address.
3. Set DATA = Address high byte (0x00 - 0xFF).
4. Give XTAL1 a positive pulse. This loads the address high byte.

H. Program Page

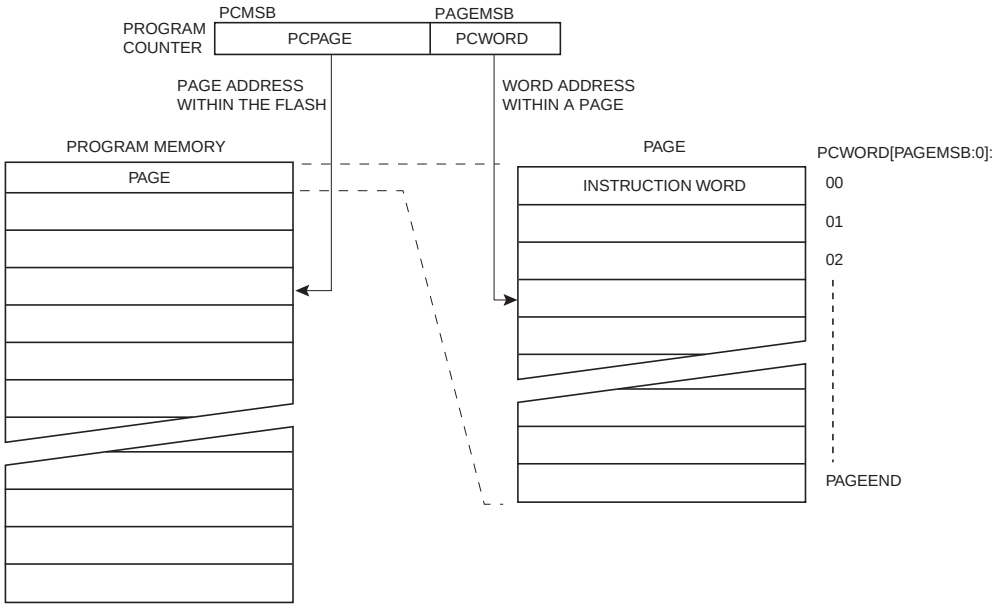
1. Give $\overline{WR}$ a negative pulse. This starts programming of the entire page of data. RDY/$\overline{BSY}$ goes low.
2. Wait until RDY/$\overline{BSY}$ goes high (See Figure 25-3 for signal waveforms).

I. Repeat B through H until the entire Flash is programmed or until all data has been programmed.

J. End Page Programming

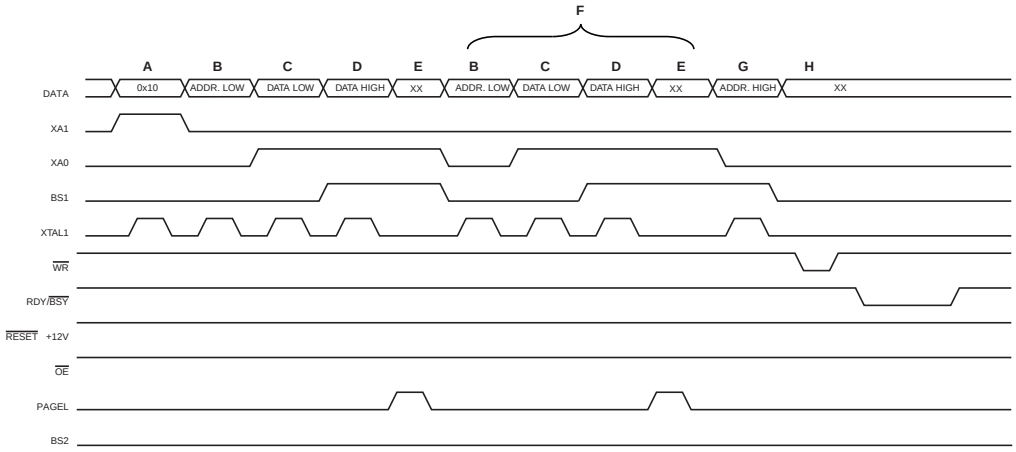
1. 1. Set XA1, XA0 to “10”. This enables command loading.
2. Set DATA to “0000 0000”. This is the command for No Operation.
3. Give XTAL1 a positive pulse. This loads the command, and the internal write signals are reset.

Figure 25-2. Addressing the Flash Which is Organized in Pages[1]



Note: 1. PCPAGE and PCWORD are listed in Table 25-11 on page 299.

Figure 25-3. Programming the Flash Waveforms[1]



Note: 1. “XX” is don’t care. The letters refer to the programming description above.

25.7.5 Programming the EEPROM

The EEPROM is organized in pages, see Table 25-12 on page 299. When programming the EEPROM, the program data is latched into a page buffer. This allows one page of data to be programmed simultaneously. The programming algorithm for the EEPROM data memory is as follows (refer to ”Programming the Flash” on page 302 for details on Command, Address and Data loading):

1. A: Load Command “0001 0001”.
2. G: Load Address High Byte (0x00 - 0xFF).
3. B: Load Address Low Byte (0x00 - 0xFF).
4. C: Load Data (0x00 - 0xFF).

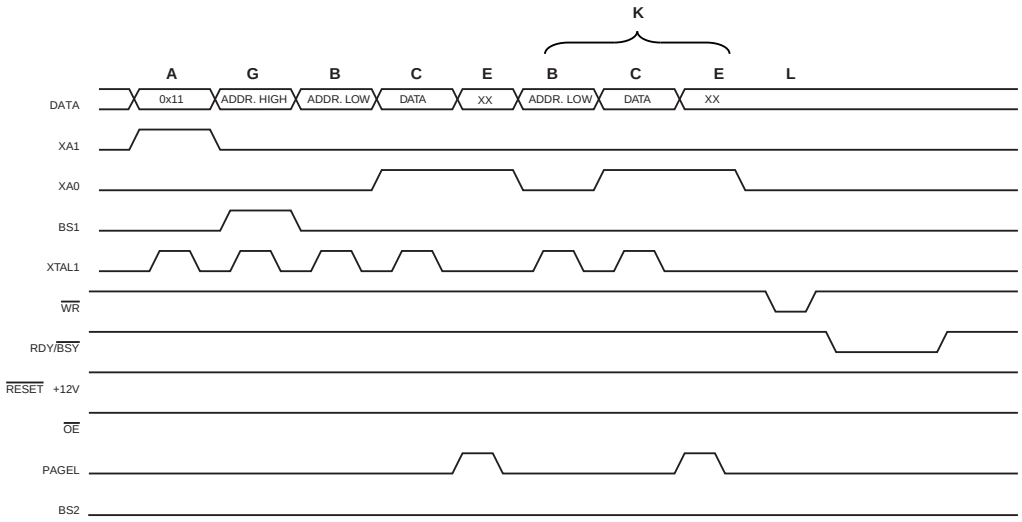
5. E: Latch data (give PAGEL a positive pulse).

K: Repeat 3 through 5 until the entire buffer is filled.

L: Program EEPROM page

1. Set BS1 to “0”.
2. Give $\overline{WR}$ a negative pulse. This starts programming of the EEPROM page. RDY/$\overline{BSY}$ goes low.
3. Wait until to RDY/$\overline{BSY}$ goes high before programming the next page (See Figure 25-4 for signal waveforms).

Figure 25-4. Programming the EEPROM Waveforms



25.7.6 Reading the Flash

The algorithm for reading the Flash memory is as follows (refer to ”Programming the Flash” on page 302 for details on Command and Address loading):

1. A: Load Command “0000 0010”.
2. G: Load Address High Byte (0x00 - 0xFF).
3. B: Load Address Low Byte (0x00 - 0xFF).
4. Set $\overline{OE}$ to “0”, and BS1 to “0”. The Flash word low byte can now be read at DATA.
5. Set BS1 to “1”. The Flash word high byte can now be read at DATA.
6. Set $\overline{OE}$ to “1”.

25.7.7 Reading the EEPROM

The algorithm for reading the EEPROM memory is as follows (refer to ”Programming the Flash” on page 302 for details on Command and Address loading):

1. A: Load Command “0000 0011”.
2. G: Load Address High Byte (0x00 - 0xFF).
3. B: Load Address Low Byte (0x00 - 0xFF).
4. Set $\overline{OE}$ to “0”, and BS1 to “0”. The EEPROM Data byte can now be read at DATA.
5. Set $\overline{OE}$ to “1”.

25.7.8 Programming the Fuse Low Bits

The algorithm for programming the Fuse Low bits is as follows (refer to ”Programming the Flash” on page 302 for details on Command and Data loading):

1. A: Load Command “0100 0000”.
2. C: Load Data Low Byte. Bit n = “0” programs and bit n = “1” erases the Fuse bit.
3. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.

25.7.9 Programming the Fuse High Bits

The algorithm for programming the Fuse High bits is as follows (refer to ”Programming the Flash” on page 302 for details on Command and Data loading):

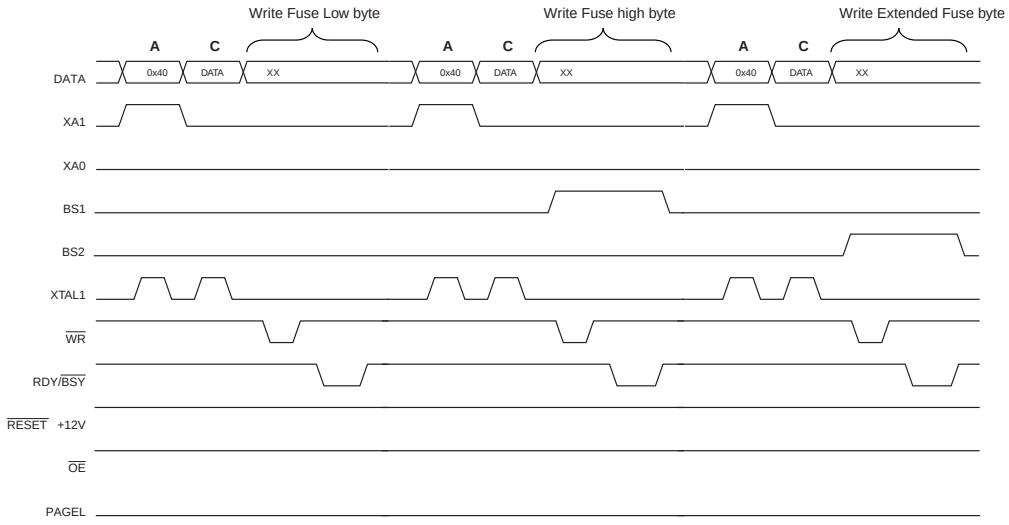
1. A: Load Command “0100 0000”.
2. C: Load Data Low Byte. Bit n = “0” programs and bit n = “1” erases the Fuse bit.
3. Set BS1 to “1” and BS2 to “0”. This selects high data byte.
4. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.
5. Set BS1 to “0”. This selects low data byte.

25.7.10 Programming the Extended Fuse Bits

The algorithm for programming the Extended Fuse bits is as follows (refer to ”Programming the Flash” on page 302 for details on Command and Data loading):

1. 1. A: Load Command “0100 0000”.
2. 2. C: Load Data Low Byte. Bit n = “0” programs and bit n = “1” erases the Fuse bit.
3. 3. Set BS1 to “0” and BS2 to “1”. This selects extended data byte.
4. 4. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.
5. 5. Set BS2 to “0”. This selects low data byte.

Figure 25-5. Programming the FUSES Waveforms



25.7.11 Programming the Lock Bits

The algorithm for programming the Lock bits is as follows (refer to ”Programming the Flash” on page 302 for details on Command and Data loading):

1. A: Load Command “0010 0000”.
2. C: Load Data Low Byte. Bit n = “0” programs the Lock bit. If LB mode 3 is programmed (LB1 and LB2 is programmed), it is not possible to program the Boot Lock bits by any External Programming mode.
3. Give $\overline{WR}$ a negative pulse and wait for RDY/$\overline{BSY}$ to go high.

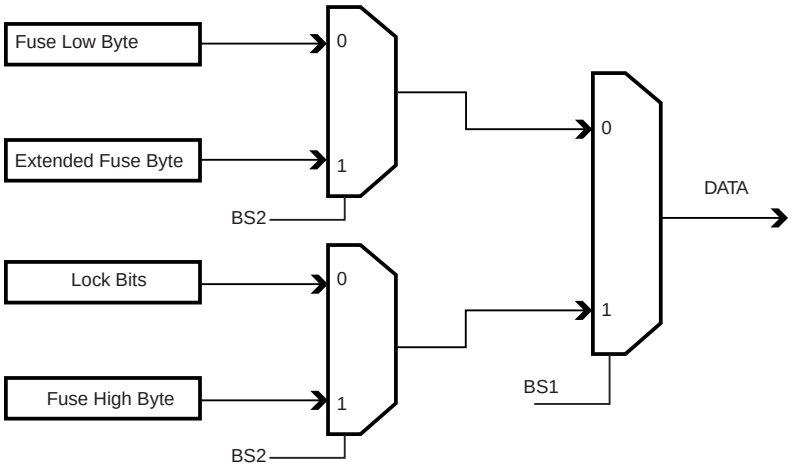
The Lock bits can only be cleared by executing Chip Erase.

25.7.12 Reading the Fuse and Lock Bits

The algorithm for reading the Fuse and Lock bits is as follows (refer to ”Programming the Flash” on page 302 for details on Command loading):

1. A: Load Command “0000 0100”.
2. Set $\overline{OE}$ to “0”, BS2 to “0” and BS1 to “0”. The status of the Fuse Low bits can now be read at DATA (“0” means programmed).
3. Set $\overline{OE}$ to “0”, BS2 to “1” and BS1 to “1”. The status of the Fuse High bits can now be read at DATA (“0” means programmed).
4. Set OE to “0”, BS2 to “1”, and BS1 to “0”. The status of the Extended Fuse bits can now be read at DATA (“0” means programmed).
5. Set $\overline{OE}$ to “0”, BS2 to “0” and BS1 to “1”. The status of the Lock bits can now be read at DATA (“0” means programmed).
6. Set $\overline{OE}$ to “1”.

Figure 25-6. Mapping Between BS1, BS2 and the Fuse and Lock Bits During Read



25.7.13 Reading the Signature Bytes

The algorithm for reading the Signature bytes is as follows (refer to ”Programming the Flash” on page 302 for details on Command and Address loading):

1. A: Load Command “0000 1000”.
2. B: Load Address Low Byte (0x00 - 0x02).
3. Set $\overline{OE}$ to “0”, and BS1 to “0”. The selected Signature byte can now be read at DATA.
4. Set $\overline{OE}$ to “1”.

25.7.14 Reading the Calibration Byte

The algorithm for reading the Calibration byte is as follows (refer to ”Programming the Flash” on page 302 for details on Command and Address loading):

1. A: Load Command “0000 1000”.
2. B: Load Address Low Byte, 0x00.
3. Set $\overline{OE}$ to “0”, and BS1 to “1”. The Calibration byte can now be read at DATA.
4. Set $\overline{OE}$ to “1”.

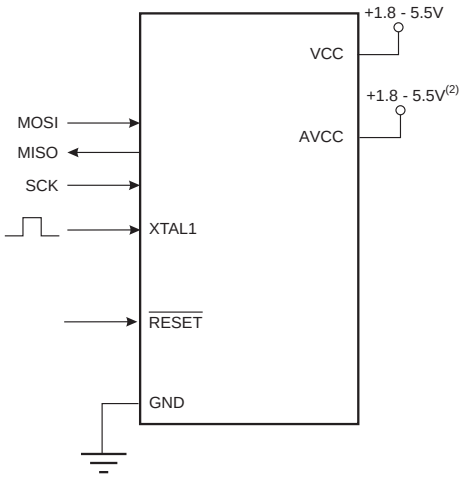
25.7.15 Parallel Programming Characteristics

For chracteristics of the Parallel Programming, see ”Parallel Programming Characteristics” on page 326.

25.8 Serial Downloading

Both the Flash and EEPROM memory arrays can be programmed using the serial SPI bus while $\overline{RESET}$ is pulled to GND. The serial interface consists of pins SCK, MOSI (input) and MISO (output). After $\overline{RESET}$ is set low, the Programming Enable instruction needs to be executed first before program/erase operations can be executed. NOTE, in Table 25-17 on page 309, the pin mapping for SPI programming is listed. Not all parts use the SPI pins dedicated for the internal SPI interface.

Figure 25-7. Serial Programming and Verify[1]



Notes: 1. If the device is clocked by the internal Oscillator, it is no need to connect a clock source to the XTAL1 pin.
2. V_{CC} - 0.3V < AV_{CC} < V_{CC} + 0.3V, however, AV_{CC} should always be within 1.8 - 5.5V

When programming the EEPROM, an auto-erase cycle is built into the self-timed programming operation (in the Serial mode ONLY) and there is no need to first execute the Chip Erase instruction. The Chip Erase operation turns the content of every memory location in both the Program and EEPROM arrays into 0xFF.

Depending on CKSEL Fuses, a valid clock must be present. The minimum low and high periods for the serial clock (SCK) input are defined as follows:

Low: > 2 CPU clock cycles for f_{ck} < 12 MHz, 3 CPU clock cycles for f_{ck} >= 12 MHz

High: > 2 CPU clock cycles for f_{ck} < 12 MHz, 3 CPU clock cycles for f_{ck} >= 12 MHz

25.8.1 Serial Programming Pin Mapping

Table 25-17. Pin Mapping Serial Programming

Symbol	Pins	I/O	Description
MOSI	PB3	I	Serial Data in
MISO	PB4	O	Serial Data out
SCK	PB5	I	Serial Clock

25.8.2 Serial Programming Algorithm

When writing serial data to the ATmega48P/88P/168P/328P, data is clocked on the rising edge of SCK.

When reading data from the ATmega48P/88P/168P/328P, data is clocked on the falling edge of SCK. See Figure 25-9 for timing details.

To program and verify the ATmega48P/88P/168P/328P in the serial programming mode, the following sequence is recommended (See Serial Programming Instruction set in Table 25-19 on page 310):

1. Power-up sequence:
   Apply power between V_{CC} and GND while $\overline{\text{RESET}}$ and SCK are set to “0”. In some systems, the programmer can not guarantee that SCK is held low during power-up. In this case, $\overline{\text{RESET}}$ must be given a positive pulse of at least two CPU clock cycles duration after SCK has been set to “0”.
2. Wait for at least 20 ms and enable serial programming by sending the Programming Enable serial instruction to pin MOSI.
3. The serial programming instructions will not work if the communication is out of synchronization. When in sync. the second byte (0x53), will echo back when issuing the third byte of the Programming Enable instruction. Whether the echo is correct or not, all four bytes of the instruction must be transmitted. If the 0x53 did not echo back, give $\overline{\text{RESET}}$ a positive pulse and issue a new Programming Enable command.
4. The Flash is programmed one page at a time. The memory page is loaded one byte at a time by supplying the 6 LSB of the address and data together with the Load Program Memory Page instruction. To ensure correct loading of the page, the data low byte must be loaded before data high byte is applied for a given address. The Program Memory Page is stored by loading the Write Program Memory Page instruction with the 7 MSB of the address. If polling (RDY/$\overline{\text{BSY}}$) is not used, the user must wait at least t_{WD_FLASH} before issuing the next page (See Table 25-18). Accessing the serial programming interface before the Flash write operation completes can result in incorrect programming.
5. **A**: The EEPROM array is programmed one byte at a time by supplying the address and data together with the appropriate Write instruction. An EEPROM memory location is first automatically erased before new data is written. If polling (RDY/$\overline{\text{BSY}}$) is not used, the user must wait at least t_{WD_EEPROM} before issuing the next byte (See Table 25-18). In a chip erased device, no 0xFFs in the data file(s) need to be programmed.
   B: The EEPROM array is programmed one page at a time. The Memory page is loaded one byte at a time by supplying the 6 LSB of the address and data together with the Load EEPROM Memory Page instruction. The EEPROM Memory Page is stored by loading the Write EEPROM Memory Page Instruction with the 7 MSB of the address. When using EEPROM page access only byte locations loaded with the Load EEPROM Memory Page instruction is altered. The remaining locations remain unchanged. If polling (RDY/$\overline{\text{BSY}}$) is

not used, the used must wait at least t_{WD_EEPROM} before issuing the next byte (See Table 25-18). In a chip erased device, no 0xFF in the data file(s) need to be programmed.

6. Any memory location can be verified by using the Read instruction which returns the content at the selected address at serial output MISO.
7. At the end of the programming session, $\overline{\text{RESET}}$ can be set high to commence normal operation.
8. Power-off sequence (if needed):
   Set $\overline{\text{RESET}}$ to “1”.
   Turn V_{CC} power off.

Table 25-18. Typical Wait Delay Before Writing the Next Flash or EEPROM Location

Symbol	Minimum Wait Delay
t_{WD_FLASH}	4.5 ms
t_{WD_EEPROM}	3.6 ms
t_{WD_ERASE}	9.0 ms

25.8.3 Serial Programming Instruction set

Table 25-19 on page 310 and Figure 25-8 on page 312 describes the Instruction set.

Table 25-19. Serial Programming Instruction Set (Hexadecimal values)

	Instruction Format			
Instruction/Operation	**Byte 1**	**Byte 2**	**Byte 3**	**Byte4**
Programming Enable	$AC	$53	$00	$00
Chip Erase (Program Memory/EEPROM)	$AC	$80	$00	$00
Poll RDY/$\overline{\text{BSY}}$	$F0	$00	$00	data byte out
Load Instructions				
Load Extended Address byte[1]	$4D	$00	Extended adr	$00
Load Program Memory Page, High byte	$48	$00	adr LSB	high data byte in
Load Program Memory Page, Low byte	$40	$00	adr LSB	low data byte in
Load EEPROM Memory Page (page access)	$C1	$00	0000 000aa	data byte in
Read Instructions				
Read Program Memory, High byte	$28	adr MSB	adr LSB	high data byte out
Read Program Memory, Low byte	$20	adr MSB	adr LSB	low data byte out
Read EEPROM Memory	$A0	0000 00aa	aaaa aaaa	data byte out
Read Lock bits	$58	$00	$00	data byte out
Read Signature Byte	$30	$00	0000 000aa	data byte out
Read Fuse bits	$50	$00	$00	data byte out
Read Fuse High bits	$58	$08	$00	data byte out
Read Extended Fuse Bits	$50	$08	$00	data byte out
Read Calibration Byte	$38	$00	$00	data byte out

Table 25-19. Serial Programming Instruction Set (Hexadecimal values) (Continued)

Instruction/Operation	Instruction Format			
	Byte 1	**Byte 2**	**Byte 3**	**Byte4**
Write Instructions[6]				
Write Program Memory Page	$4C	adr MSB	adr LSB	$00
Write EEPROM Memory	$C0	0000 00aa	aaaa aaaa	data byte in
Write EEPROM Memory Page (page access)	$C2	0000 00aa	aaaa aa00	$00
Write Lock bits	$AC	$E0	$00	data byte in
Write Fuse bits	$AC	$A0	$00	data byte in
Write Fuse High bits	$AC	$A8	$00	data byte in
Write Extended Fuse Bits	$AC	$A4	$00	data byte in

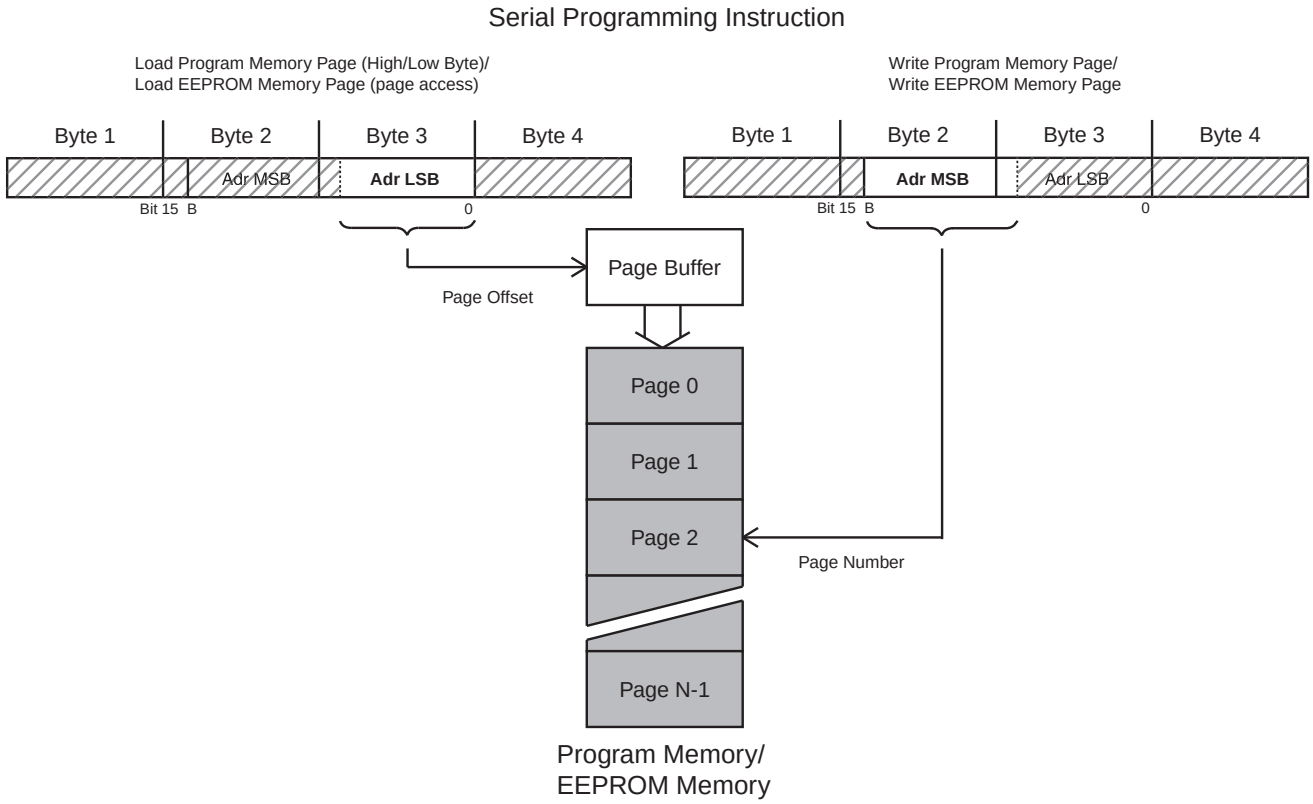
Notes:
1. Not all instructions are applicable for all parts.
2. a = address.
3. Bits are programmed ‘0’, unprogrammed ‘1’.
4. To ensure future compatibility, unused Fuses and Lock bits should be unprogrammed (‘1’) .
5. Refer to the correspondig section for Fuse and Lock bits, Calibration and Signature bytes and Page size.
6. Instructions accessing program memory use a word address. This address may be random within the page range.
7. See htt://www.atmel.com/avr for Application Notes regarding programming and programmers.

If the LSB in RDY/BSY data byte out is ‘1’, a programming operation is still pending. Wait until this bit returns ‘0’ before the next instruction is carried out.

Within the same page, the low data byte must be loaded prior to the high data byte.

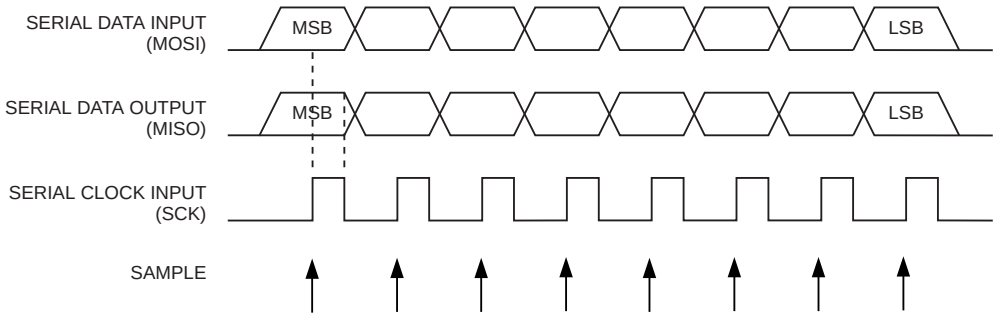
After data is loaded to the page buffer, program the EEPROM page, see Figure 25-8 on page 312.

Figure 25-8. Serial Programming Instruction example



25.8.4 SPI Serial Programming Characteristics

Figure 25-9. Serial Programming Waveforms



For characteristics of the SPI module see “SPI Timing Characteristics” on page 321.

26. Electrical Characteristics

26.1 Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0 mA
DC Current V_{CC} and GND Pins	200.0 mA

*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

26.2 DC Characteristics

.

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
V_{IL}	Input Low Voltage, except XTAL1 and $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5 -0.5		$0.2V_{CC}$[1] $0.3V_{CC}$[1]	V
V_{IH}	Input High Voltage, except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$[2] $0.6V_{CC}$[2]		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{IL1}	Input Low Voltage, XTAL1 pin	V_{CC} = 1.8V - 5.5V	-0.5		$0.1V_{CC}$[1]	V
V_{IH1}	Input High Voltage, XTAL1 pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.8V_{CC}$[2] $0.7V_{CC}$[2]		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{IL2}	Input Low Voltage, $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 5.5V	-0.5		$0.1V_{CC}$[1]	V
V_{IH2}	Input High Voltage, $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 5.5V	$0.9V_{CC}$[2]		V_{CC} + 0.5	V
V_{IL3}	Input Low Voltage, $\overline{\text{RESET}}$ pin as I/O	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5 -0.5		$0.2V_{CC}$[1] $0.3V_{CC}$[1]	V
V_{IH3}	Input High Voltage, $\overline{\text{RESET}}$ pin as I/O	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$[2] $0.6V_{CC}$[2]		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{OL}	Output Low Voltage[3] except RESET pin	I_{OL} = 20 mA, V_{CC} = 5V I_{OL} = 10 mA, V_{CC} = 3V			0.9 0.6	V
V_{OH}	Output High Voltage[4] except Reset pin	I_{OH} = -20 mA, V_{CC} = 5V I_{OH} = -10 mA, V_{CC} = 3V	4.2 2.3			V
I_{IL}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin low (absolute value)			1	µA
I_{IH}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin high (absolute value)			1	µA

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted) (Continued)

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
R_{RST}	Reset Pull-up Resistor		30		60	kΩ
R_{PU}	I/O Pin Pull-up Resistor		20		50	kΩ
V_{ACIO}	Analog Comparator Input Offset Voltage	V_{CC} = 5V V_{in} = V_{CC}/2		<10	40	mV
I_{ACLK}	Analog Comparator Input Leakage Current	V_{CC} = 5V V_{in} = V_{CC}/2	-50		50	nA
t_{ACID}	Analog Comparator Propagation Delay	V_{CC} = 2.7V V_{CC} = 4.0V		750 500		ns

Notes:
1. “Max” means the highest value where the pin is guaranteed to be read as low
2. “Min” means the lowest value where the pin is guaranteed to be read as high
3. Although each I/O port can sink more than the test conditions (20 mA at V_{CC} = 5V, 10 mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   ATmega48P/88P/168P/328P:
   1] The sum of all I_{OL}, for ports C0 - C5, ADC7, ADC6 should not exceed 100 mA.
   2] The sum of all I_{OL}, for ports B0 - B5, D5 - D7, XTAL1, XTAL2 should not exceed 100 mA.
   3] The sum of all I_{OL}, for ports D0 - D4, $\overline{\text{RESET}}$ should not exceed 100 mA.
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.
4. Although each I/O port can source more than the test conditions (20 mA at V_{CC} = 5V, 10 mA at V_{CC} = 3V) under steady state conditions (non-transient), the following must be observed:
   ATmega48P/88P/168P/328P:
   1] The sum of all I_{OH}, for ports C0 - C5, D0- D4, ADC7, $\overline{\text{RESET}}$ should not exceed 150 mA.
   2] The sum of all I_{OH}, for ports B0 - B5, D5 - D7, ADC6, XTAL1, XTAL2 should not exceed 150 mA.
   If II_{OH} exceeds the test condition, V_{OH} may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.

26.2.1 ATmega48P DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.	Typ.[2]	Max.	Units
I_{CC}	Power Supply Current[1]	Active 1 MHz, V_{CC} = 2V		0.3	0.5	mA
		Active 4 MHz, V_{CC} = 3V		1.9	2.5	mA
		Active 8 MHz, V_{CC} = 5V		6.8	9	mA
		Idle 1 MHz, V_{CC} = 2V		0.06	0.15	mA
		Idle 4 MHz, V_{CC} = 3V		0.4	0.7	mA
		Idle 8 MHz, V_{CC} = 5V		1.6	2.7	mA
	Power-save mode[3][4]	32 kHz TOSC enabled, V_{CC} = 1.8V		0.75	1.6	µA
		32 kHz TOSC enabled, V_{CC} = 3V		0.85	2.6	µA
	Power-down mode[3]	WDT enabled, V_{CC} = 3V		4.2	8	µA
		WDT disabled, V_{CC} = 3V		0.18	2	µA

Notes:
1. Values with “Minimizing Power Consumption” enabled (0xFF).
2. Typical values at 25°C. Maximum values are test limits in production.
3. The current consumption values include input leakage current.

4. Maximum values are characterized values and not test limits in production.

26.2.2 ATmega88P DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.	Typ.[2]	Max.	Units
I_{CC}	Power Supply Current[1]	Active 1 MHz, V_{CC} = 2V		0.3	0.5	mA
		Active 4 MHz, V_{CC} = 3V		1.7	2.5	mA
		Active 8 MHz, V_{CC} = 5V		6.3	9	mA
		Idle 1 MHz, V_{CC} = 2V		0.05	0.15	mA
		Idle 4 MHz, V_{CC} = 3V		0.3	0.7	mA
		Idle 8 MHz, V_{CC} = 5V		1.4	2.7	mA
	Power-save mode[3][4]	32 kHz TOSC enabled, V_{CC} = 1.8V		0.72	1.6	µA
		32 kHz TOSC enabled, V_{CC} = 3V		0.9	2.6	µA
	Power-down mode[3]	WDT enabled, V_{CC} = 3V		4.4	8	µA
		WDT disabled, V_{CC} = 3V		0.2	2	µA

Notes:
1. Values with “Minimizing Power Consumption” enabled (0xFF).
2. Typical values at 25°C. Maximum values are test limits in production.
3. The current consumption values include input leakage current.
4. Maximum values are characterized values and not test limits in production.

26.2.3 ATmega168P DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.	Typ.[2]	Max.	Units
I_{CC}	Power Supply Current[1]	Active 1 MHz, V_{CC} = 2V		0.3	0.5	mA
		Active 4 MHz, V_{CC} = 3V		1.8	2.5	mA
		Active 8 MHz, V_{CC} = 5V		6.7	9	mA
		Idle 1 MHz, V_{CC} = 2V		0.06	0.15	mA
		Idle 4 MHz, V_{CC} = 3V		0.4	0.7	mA
		Idle 8 MHz, V_{CC} = 5V		1.7	2.7	mA
	Power-save mode[3][4]	32 kHz TOSC enabled, V_{CC} = 1.8V		0.8	1.6	µA
		32 kHz TOSC enabled, V_{CC} = 3V		0.9	2.6	µA
	Power-down mode[3]	WDT enabled, V_{CC} = 3V		4.6	8	µA
		WDT disabled, V_{CC} = 3V		0.1	2	µA

Notes:
1. Values with “Minimizing Power Consumption” enabled (0xFF).
2. Typical values at 25°C. Maximum values are test limits in production.

3. The current consumption values include input leakage current.
4. Maximum values are characterized values and not test limits in production.

26.2.4 ATmega328P DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.	Typ.[2]	Max.	Units
I_{CC}	Power Supply Current[1]	Active 1 MHz, V_{CC} = 2V		0.3	0.5	mA
		Active 4 MHz, V_{CC} = 3V		1.7	2.5	mA
		Active 8 MHz, V_{CC} = 5V		5.2	9	mA
		Idle 1 MHz, V_{CC} = 2V		0.04	0.15	mA
		Idle 4 MHz, V_{CC} = 3V		0.3	0.7	mA
		Idle 8 MHz, V_{CC} = 5V		1.2	2.7	mA
	Power-save mode[3][4]	32 kHz TOSC enabled, V_{CC} = 1.8V		0.8	1.6	µA
		32 kHz TOSC enabled, V_{CC} = 3V		0.9	2.6	µA
	Power-down mode[3]	WDT enabled, V_{CC} = 3V		4.2	8	µA
		WDT disabled, V_{CC} = 3V		0.1	2	µA

Notes:
1. Values with “Minimizing Power Consumption” enabled (0xFF).
2. Typical values at 25°C. Maximum values are test limits in production.
3. The current consumption values include input leakage current.
4. Maximum values are characterized values and not test limits in production.

26.3 Speed Grades

Maximum frequency is dependent on $V_{CC.}$ As shown in Figure 26-1 and Figure 26-2, the Maximum Frequency vs. V_{CC} curve is linear between 1.8V < V_{CC} < 2.7V and between 2.7V < V_{CC} < 4.5V.

Figure 26-1. Maximum Frequency vs. V_{CC}, ATmega48P/88P/168PV

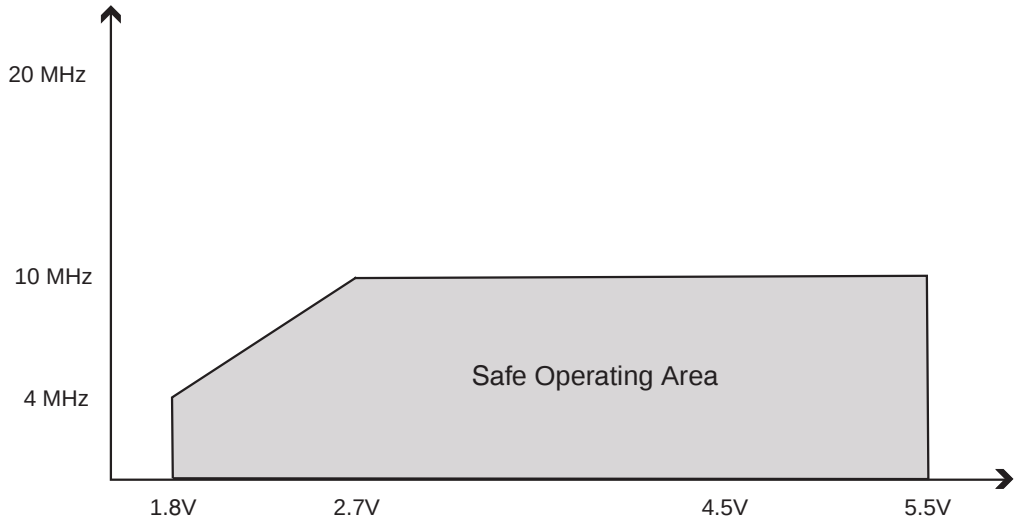


Figure 26-2. Maximum Frequency vs. V_{CC}, ATmega48P/88P/168P

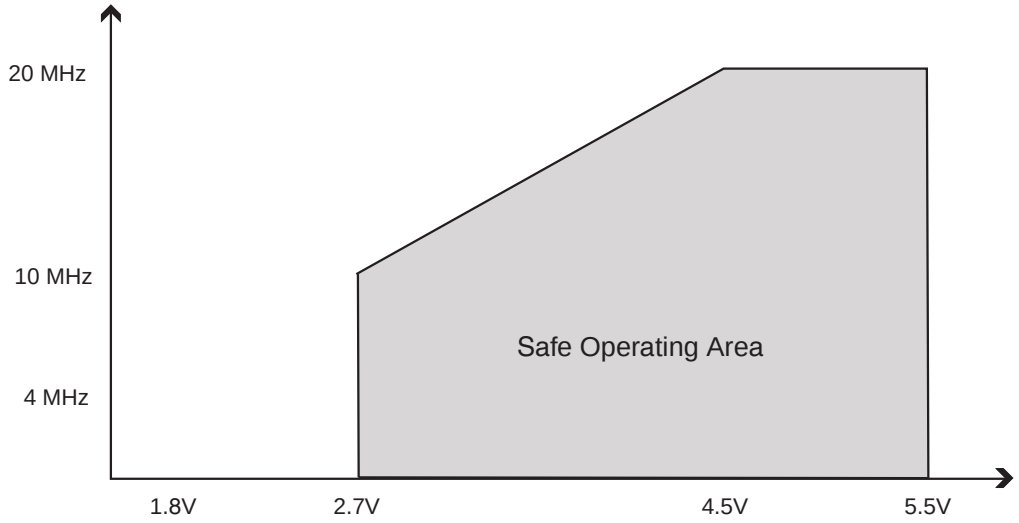
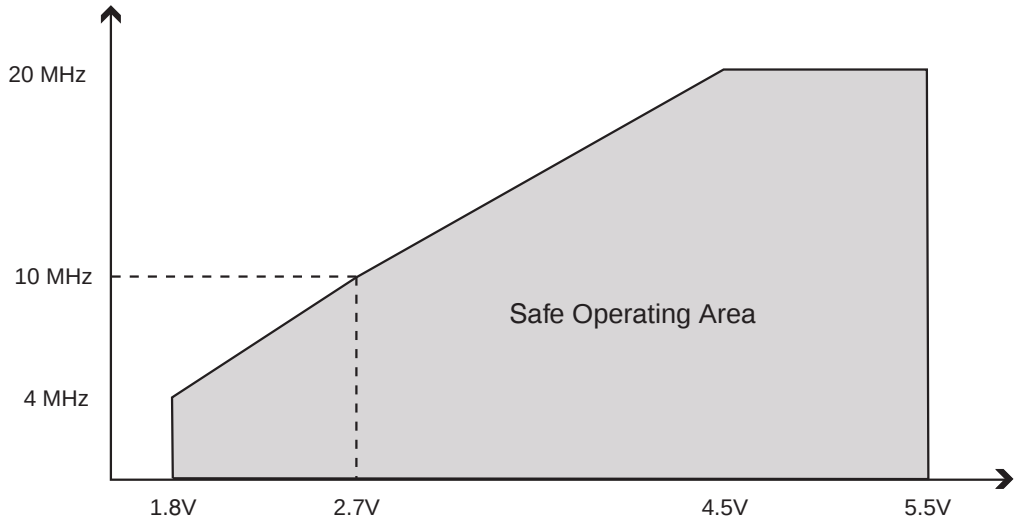


Figure 26-3. Maximum Frequency vs. V_{CC}, ATmega328P



26.4 Clock Characteristics

26.4.1 Calibrated Internal RC Oscillator Accuracy

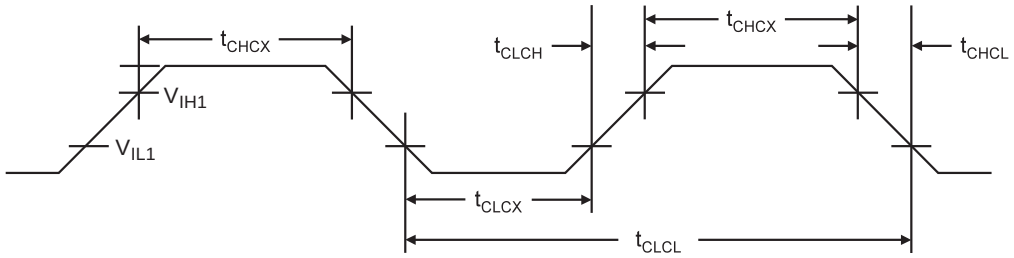
Table 26-1. Calibration Accuracy of Internal RC Oscillator

	Frequency	V_{CC}	Temperature	Calibration Accuracy
Factory Calibration	8.0 MHz	3V	25°C	±10%
User Calibration	7.3 - 8.1 MHz	1.8V - 5.5V(1) 2.7V - 5.5V(2)	-40°C - 85°C	±1%

Notes: 1. Voltage range for ATmega48PV/88PV/168PV/328PV.
2. Voltage range for ATmega48P/88P/168P/328P.

26.4.2 External Clock Drive Waveforms

Figure 26-4. External Clock Drive Waveforms



26.4.3 External Clock Drive

Table 26-2. External Clock Drive

Symbol	Parameter	V_{CC}=1.8-5.5V		V_{CC}=2.7-5.5V		V_{CC}=4.5-5.5V		Units
		Min.	Max.	Min.	Max.	Min.	Max.	
$1/t_{CLCL}$	Oscillator Frequency	0	4	0	10	0	20	MHz
t_{CLCL}	Clock Period	250		100		50		ns
t_{CHCX}	High Time	100		40		20		ns
t_{CLCX}	Low Time	100		40		20		ns
t_{CLCH}	Rise Time		2.0		1.6		0.5	µs
t_{CHCL}	Fall Time		2.0		1.6		0.5	µs
Δt_{CLCL}	Change in period from one clock cycle to the next		2		2		2	%

Note: All DC Characteristics contained in this datasheet are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are preliminary values representing design targets, and will be updated after characterization of actual silicon.

26.5 System and Reset Characteristics

Table 26-3. Reset, Brown-out and Internal Voltage Characteristics[1]

Symbol	Parameter		Min	Typ	Max	Units
V_{POT}	Power-on Reset Threshold Voltage (rising)		1.1	1.4	1.6	V
	Power-on Reset Threshold Voltage (falling)[2]		0.6	1.3	1.6	V
SR_{ON}	Power-on Slope Rate		0.01		10	V/ms
V_{RST}	$\overline{\text{RESET}}$ Pin Threshold Voltage		0.2 V_{CC}		0.9 V_{CC}	V
t_{RST}	Minimum pulse width on $\overline{\text{RESET}}$ Pin				2.5	µs
V_{HYST}	Brown-out Detector Hysteresis			50		mV
t_{BOD}	Min Pulse Width on Brown-out Reset			2		µs
V_{BG}	Bandgap reference voltage	V_{CC}=2.7 T_A=25°C	1.0	1.1	1.2	V
t_{BG}	Bandgap reference start-up time	V_{CC}=2.7 T_A=25°C		40	70	µs
I_{BG}	Bandgap reference current consumption	V_{CC}=2.7 T_A=25°C		10		µA

Notes: 1. Values are guidelines only.
2. The Power-on Reset will not work unless the supply voltage has been below V_{POT} (falling).

Table 26-4. BODLEVEL Fuse Coding[1]

BODLEVEL 2:0 Fuses	Min V_{BOT}	Typ V_{BOT}	Max V_{BOT}	Units
111	BOD Disabled			
110	1.7	1.8	2.0	V
101	2.5	2.7	2.9	
100	4.1	4.3	4.5	
011	Reserved			
010				
001				
000				

Notes: 1. V_{BOT} may be below nominal minimum operating voltage for some devices. For devices where this is the case, the device is tested down to V_{CC} = V_{BOT} during the production test. This guarantees that a Brown-Out Reset will occur before V_{CC} drops to a voltage where correct operation of the microcontroller is no longer guaranteed. The test is performed using BODLEVEL = 110 and BODLEVEL = 101 for ATmega48P/88P/168PV and ATmega328P, and BODLEVEL = 101 and BODLEVEL = 100 for ATmega48P/88P/168P/328P.

26.6 SPI Timing Characteristics

See Figure 26-5 and Figure 26-6 for details.

Table 26-5. SPI Timing Parameters

	Description	Mode	Min	Typ	Max	
1	SCK period	Master		See Table 16-5		ns
2	SCK high/low	Master		50% duty cycle		
3	Rise/Fall time	Master		3.6		
4	Setup	Master		10		
5	Hold	Master		10		
6	Out to SCK	Master		0.5 • t_{sck}		
7	SCK to out	Master		10		
8	SCK to out high	Master		10		
9	$\overline{SS}$ low to out	Slave		15		
10	SCK period	Slave	4 • t_{ck}			
11	SCK high/low(1)	Slave	2 • t_{ck}			
12	Rise/Fall time	Slave			1600	
13	Setup	Slave	10			
14	Hold	Slave	t_{ck}			
15	SCK to out	Slave		15		
16	SCK to $\overline{SS}$ high	Slave	20			
17	$\overline{SS}$ high to tri-state	Slave		10		
18	$\overline{SS}$ low to SCK	Slave	20			

Note: 1. In SPI Programming mode the minimum SCK high/low period is:
- 2 t_{CLCL} for f_{CK} < 12 MHz
- 3 t_{CLCL} for f_{CK} > 12 MHz

2. All DC Characteristics contained in this datasheet are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are preliminary values representing design targets, and will be updated after characterization of actual silicon.

Figure 26-5. SPI Interface Timing Requirements (Master Mode)

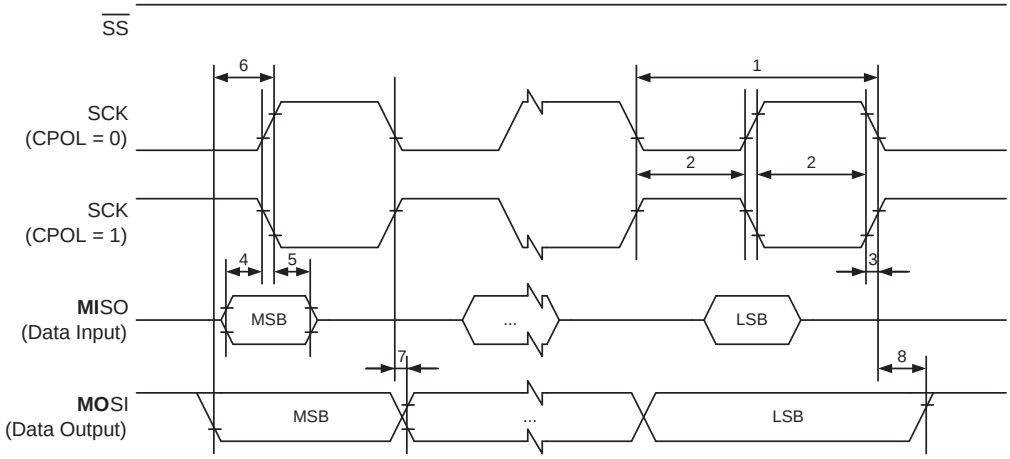
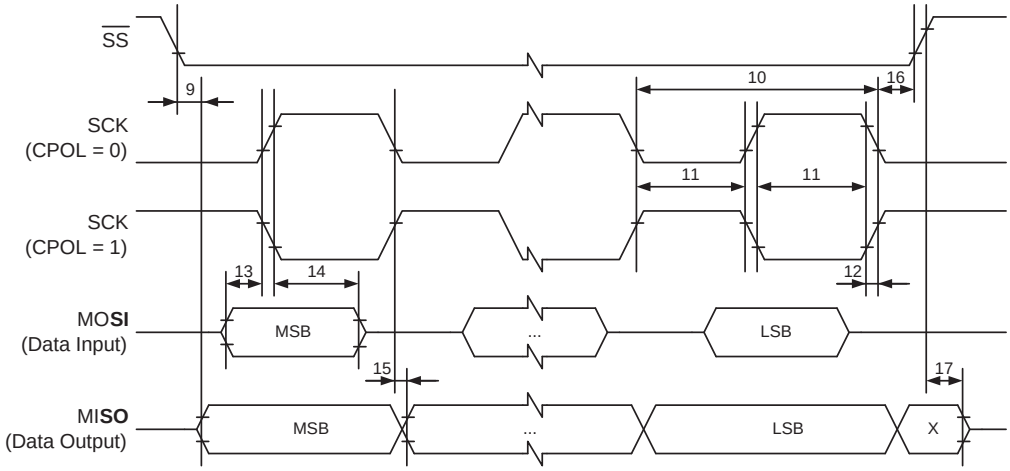


Figure 26-6. SPI Interface Timing Requirements (Slave Mode)



26.7 2-wire Serial Interface Characteristics

Table 26-6 describes the requirements for devices connected to the 2-wire Serial Bus. The ATmega48P/88P/168P/328P 2-wire Serial Interface meets or exceeds these requirements under the noted conditions.

Timing symbols refer to Figure 26-7.

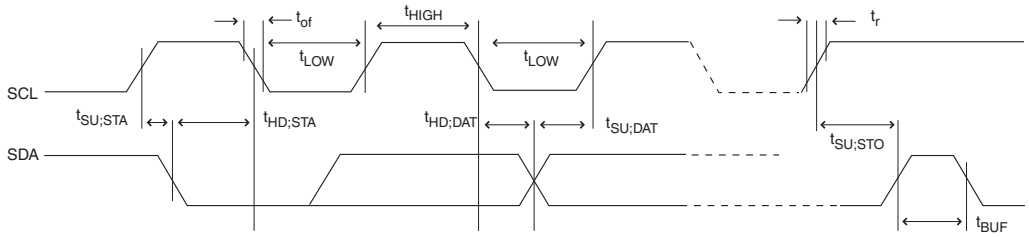
Table 26-6. 2-wire Serial Bus Requirements

Symbol	Parameter	Condition	Min	Max	Units
V_{IL}	Input Low-voltage		-0.5	0.3 V_{CC}	V
V_{IH}	Input High-voltage		0.7 V_{CC}	V_{CC} + 0.5	V
V_{hys} (1)	Hysteresis of Schmitt Trigger Inputs		0.05 V_{CC} (2)	–	V
V_{OL} (1)	Output Low-voltage	3 mA sink current	0	0.4	V
t_r (1)	Rise Time for both SDA and SCL		20 + 0.1C_b (3)(2)	300	ns
t_{of} (1)	Output Fall Time from V_{IHmin} to V_{ILmax}	10 pF < C_b < 400 pF (3)	20 + 0.1C_b (3)(2)	250	ns
t_{SP} (1)	Spikes Suppressed by Input Filter		0	50 (2)	ns
I_i	Input Current each I/O Pin	0.1V_{CC} < V_i < 0.9V_{CC}	-10	10	µA
C_i (1)	Capacitance for each I/O Pin		–	10	pF
f_{SCL}	SCL Clock Frequency	f_{CK} (4) > max(16f_{SCL}, 250kHz) (5)	0	400	kHz
Rp	Value of Pull-up resistor	f_{SCL} ≤ 100 kHz	$\frac{V_{CC} - 0{,}4V}{3mA}$	$\frac{1000ns}{C_b}$	Ω
		f_{SCL} > 100 kHz	$\frac{V_{CC} - 0{,}4V}{3mA}$	$\frac{300ns}{C_b}$	Ω
$t_{HD;STA}$	Hold Time (repeated) START Condition	f_{SCL} ≤ 100 kHz	4.0	–	µs
		f_{SCL} > 100 kHz	0.6	–	µs
t_{LOW}	Low Period of the SCL Clock	f_{SCL} ≤ 100 kHz	4.7	–	µs
		f_{SCL} > 100 kHz	1.3	–	µs
t_{HIGH}	High period of the SCL clock	f_{SCL} ≤ 100 kHz	4.0	–	µs
		f_{SCL} > 100 kHz	0.6	–	µs
$t_{SU;STA}$	Set-up time for a repeated START condition	f_{SCL} ≤ 100 kHz	4.7	–	µs
		f_{SCL} > 100 kHz	0.6	–	µs
$t_{HD;DAT}$	Data hold time	f_{SCL} ≤ 100 kHz	0	3.45	µs
		f_{SCL} > 100 kHz	0	0.9	µs
$t_{SU;DAT}$	Data setup time	f_{SCL} ≤ 100 kHz	250	–	ns
		f_{SCL} > 100 kHz	100	–	ns
$t_{SU;STO}$	Setup time for STOP condition	f_{SCL} ≤ 100 kHz	4.0	–	µs
		f_{SCL} > 100 kHz	0.6	–	µs
t_{BUF}	Bus free time between a STOP and START condition	f_{SCL} ≤ 100 kHz	4.7	–	µs
		f_{SCL} > 100 kHz	1.3	–	µs

Notes: 1. In ATmega48P/88P/168P/328P, this parameter is characterized and not 100% tested.
2. Required only for f_{SCL} > 100 kHz.

3. C_b = capacitance of one bus line in pF.
4. f_{CK} = CPU clock frequency
5. This requirement applies to all ATmega48P/88P/168P/328P 2-wire Serial Interface operation. Other devices connected to the 2-wire Serial Bus need only obey the general f_{SCL} requirement.

Figure 26-7. 2-wire Serial Bus Timing



26.8 ADC Characteristics – Preliminary Data

Table 26-7. ADC Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
	Resolution			10		Bits
	Absolute accuracy (Including INL, DNL, quantization error, gain and offset error)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		2		LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 1 MHz		4.5		LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz Noise Reduction Mode		2		LSB
		V_{REF} = 4V, V_{CC} = 4V, ADC clock = 1 MHz Noise Reduction Mode		4.5		LSB
	Integral Non-Linearity (INL)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		0.5		LSB
	Differential Non-Linearity (DNL)	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		0.25		LSB
	Gain Error	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		2		LSB
	Offset Error	V_{REF} = 4V, V_{CC} = 4V, ADC clock = 200 kHz		2		LSB
	Conversion Time	Free Running Conversion	13		260	µs
	Clock Frequency		50		1000	kHz
AV_{CC} (1)	Analog Supply Voltage		V_{CC} - 0.3		V_{CC} + 0.3	V
V_{REF}	Reference Voltage		1.0		AV_{CC}	V
V_{IN}	Input Voltage		GND		V_{REF}	V
	Input Bandwidth			38.5		kHz
V_{INT}	Internal Voltage Reference		1.0	1.1	1.2	V
R_{REF}	Reference Input Resistance			32		kΩ
R_{AIN}	Analog Input Resistance			100		MΩ

Note: 1. AV_{CC} absolute min/max: 1.8V/5.5V

26.9 Parallel Programming Characteristics

Table 26-8. Parallel Programming Characteristics, V_{CC} = 5V ± 10%

Symbol	Parameter	Min	Typ	Max	Units
V_{PP}	Programming Enable Voltage	11.5		12.5	V
I_{PP}	Programming Enable Current			250	µA
t_{DVXH}	Data and Control Valid before XTAL1 High	67			ns
t_{XLXH}	XTAL1 Low to XTAL1 High	200			ns
t_{XHXL}	XTAL1 Pulse Width High	150			ns
t_{XLDX}	Data and Control Hold after XTAL1 Low	67			ns
t_{XLWL}	XTAL1 Low to $\overline{WR}$ Low	0			ns
t_{XLPH}	XTAL1 Low to PAGEL high	0			ns
t_{PLXH}	PAGEL low to XTAL1 high	150			ns
t_{BVPH}	BS1 Valid before PAGEL High	67			ns
t_{PHPL}	PAGEL Pulse Width High	150			ns
t_{PLBX}	BS1 Hold after PAGEL Low	67			ns
t_{WLBX}	BS2/1 Hold after $\overline{WR}$ Low	67			ns
t_{PLWL}	PAGEL Low to $\overline{WR}$ Low	67			ns
t_{BVWL}	BS1 Valid to $\overline{WR}$ Low	67			ns
t_{WLWH}	$\overline{WR}$ Pulse Width Low	150			ns
t_{WLRL}	$\overline{WR}$ Low to RDY/$\overline{BSY}$ Low	0		1	µs
t_{WLRH}	$\overline{WR}$ Low to RDY/$\overline{BSY}$ High[1]	3.7		4.5	ms
t_{WLRH_CE}	$\overline{WR}$ Low to RDY/$\overline{BSY}$ High for Chip Erase[2]	7.5		9	ms
t_{XLOL}	XTAL1 Low to $\overline{OE}$ Low	0			ns
t_{BVDV}	BS1 Valid to DATA valid	0		250	ns
t_{OLDV}	$\overline{OE}$ Low to DATA Valid			250	ns
t_{OHDZ}	$\overline{OE}$ High to DATA Tri-stated			250	ns

Notes: 1. t_{WLRH} is valid for the Write Flash, Write EEPROM, Write Fuse bits and Write Lock bits commands.
2. t_{WLRH_CE} is valid for the Chip Erase command.

Figure 26-8. Parallel Programming Timing, Including some General Timing Requirements

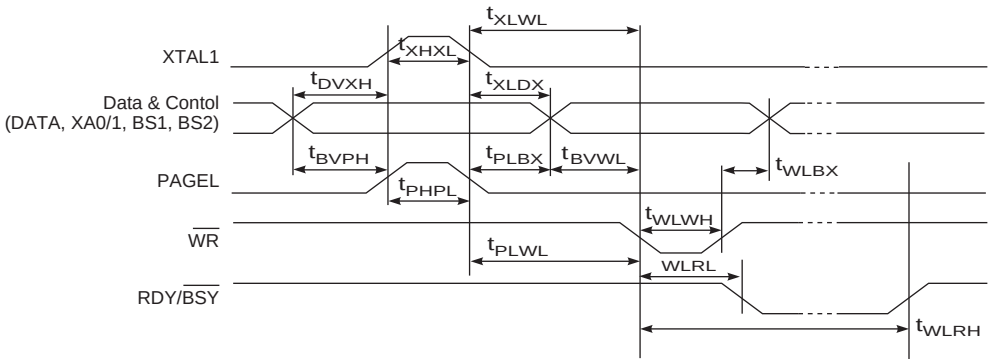
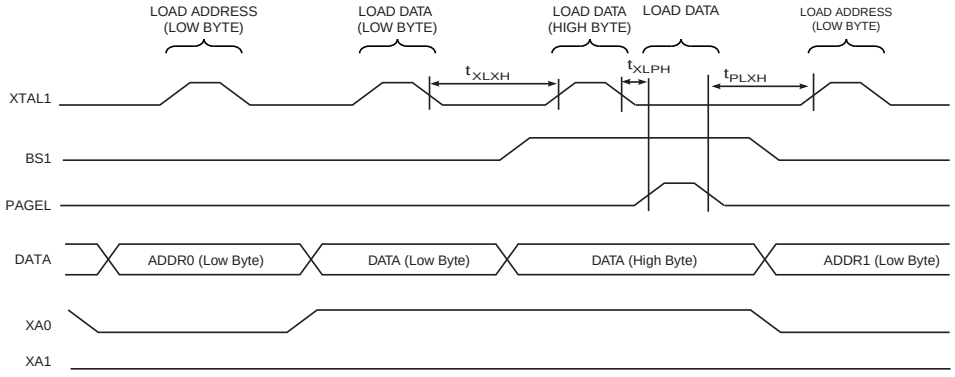
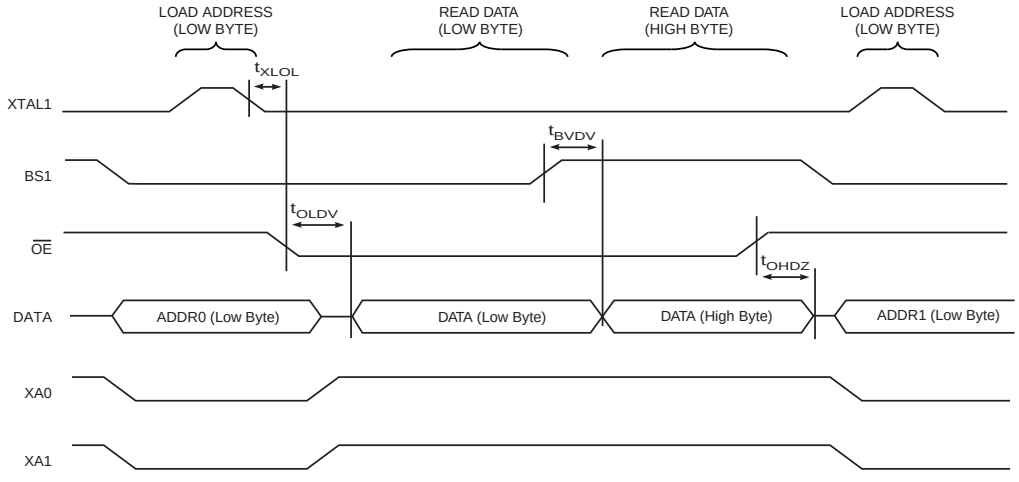


Figure 26-9. Parallel Programming Timing, Loading Sequence with Timing Requirements[1]



Note: 1. The timing requirements shown in Figure 26-8 (i.e., t_{DVXH}, t_{XHXL}, and t_{XLDX}) also apply to loading operation.

Figure 26-10. Parallel Programming Timing, Reading Sequence (within the Same Page) with Timing Requirements[1]



Note: 1. The timing requirements shown in Figure 26-8 (i.e., t_{DVXH}, t_{XHXL}, and t_{XLDX}) also apply to reading operation.

27. Typical Characteristics

The following charts show typical behavior. These figures are not tested during manufacturing. All current consumption measurements are performed with all I/O pins configured as inputs and with internal pull-ups enabled. A square wave generator with rail-to-rail output is used as clock source.

All Active- and Idle current consumption measurements are done with all bits in the PRR register set and thus, the corresponding I/O modules are turned off. Also the Analog Comparator is disabled during these measurements. The ”Supply Current of IO Modules” on page 334, page 358 and page 382 shows the additional current consumption compared to I_{CC} Active and I_{CC} Idle for every I/O module controlled by the Power Reduction Register. See ”Power Reduction Register” on page 42 for details.

The power consumption in Power-down mode is independent of clock selection.

The current consumption is a function of several factors such as: operating voltage, operating frequency, loading of I/O pins, switching rate of I/O pins, code executed and ambient temperature. The dominating factors are operating voltage and frequency.

The current drawn from capacitive loaded pins may be estimated (for one pin) as $C_L*V_{CC}*f$ where C_L = load capacitance, V_{CC} = operating voltage and f = average switching frequency of I/O pin.

The parts are characterized at frequencies higher than test limits. Parts are not guaranteed to function properly at frequencies higher than the ordering code indicates.

The difference between current consumption in Power-down mode with Watchdog Timer enabled and Power-down mode with Watchdog Timer disabled represents the differential current drawn by the Watchdog Timer.

27.1 ATmega48P Typical Characteristics

27.1.1 Active Supply Current

Figure 27-1. Active Supply Current vs. Low Frequency (0.1-1.0 MHz).

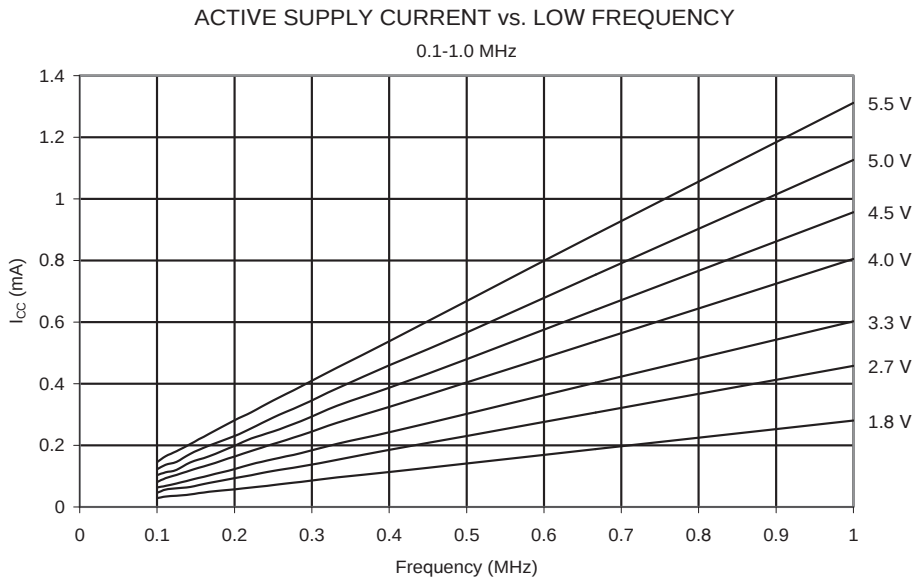


Figure 27-2. Active Supply Current vs. Frequency (1 - 20 MHz).

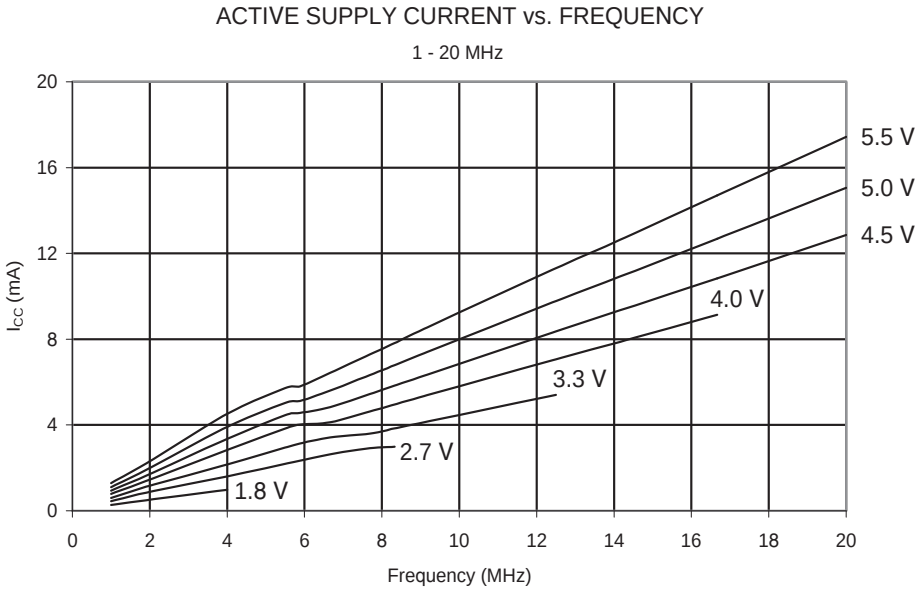


Figure 27-3. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz).

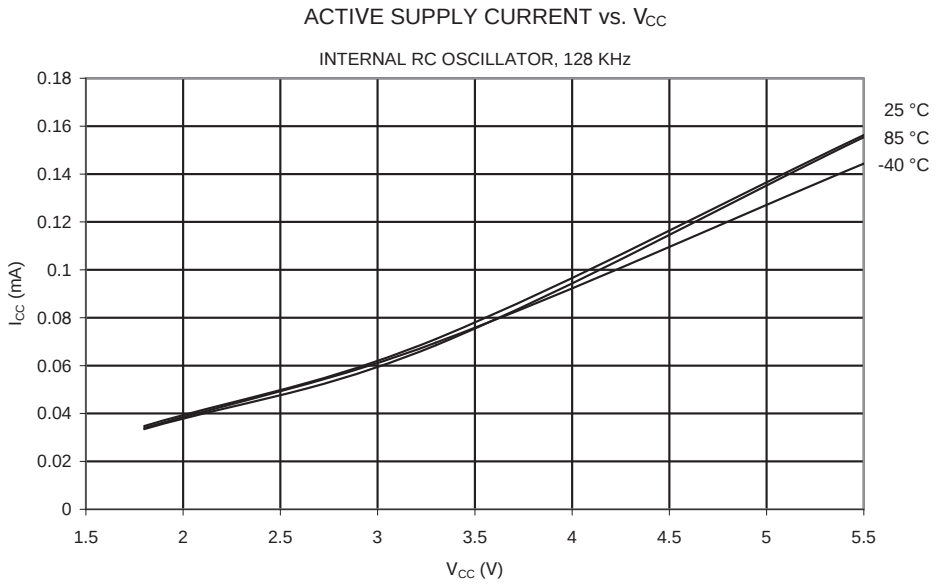


Figure 27-4. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz).

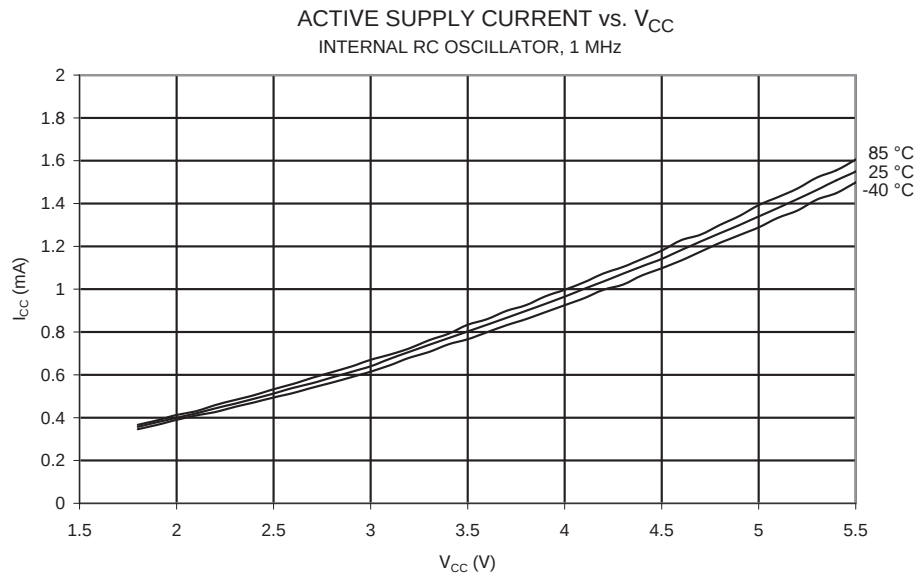
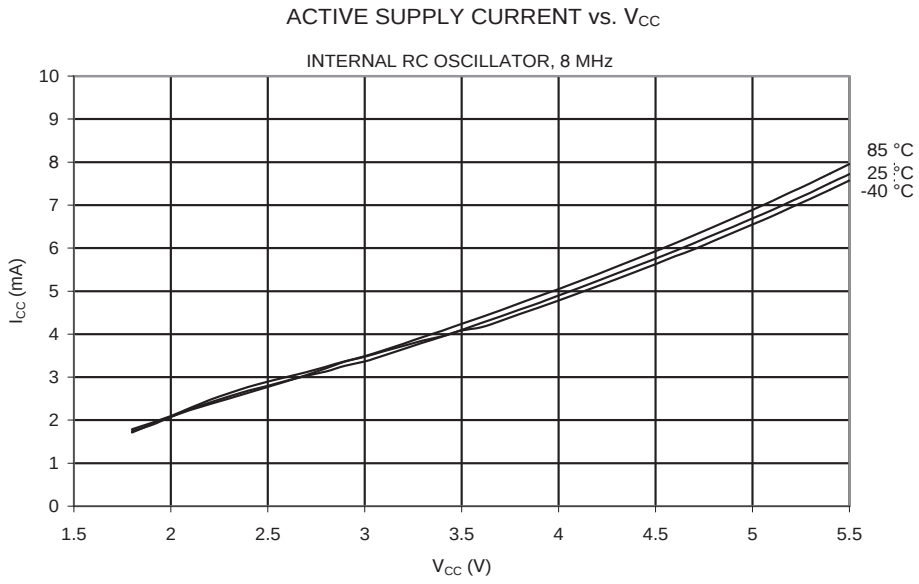


Figure 27-5. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 8 MHz).



27.1.2 Idle Supply Current

Figure 27-6. Idle Supply Current vs. Low Frequency (0.1-1.0 MHz).

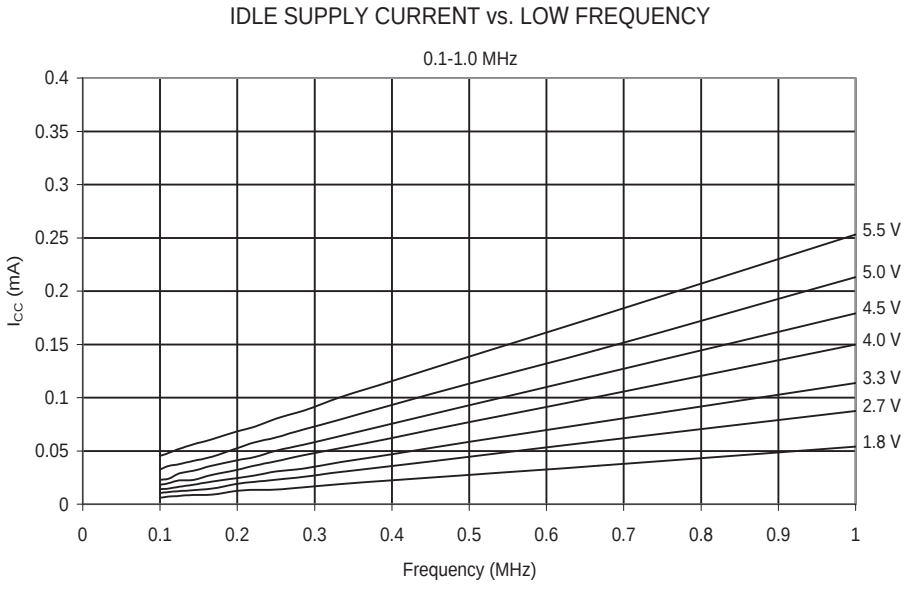


Figure 27-7. Idle Supply Current vs. Frequency (1-20 MHz).

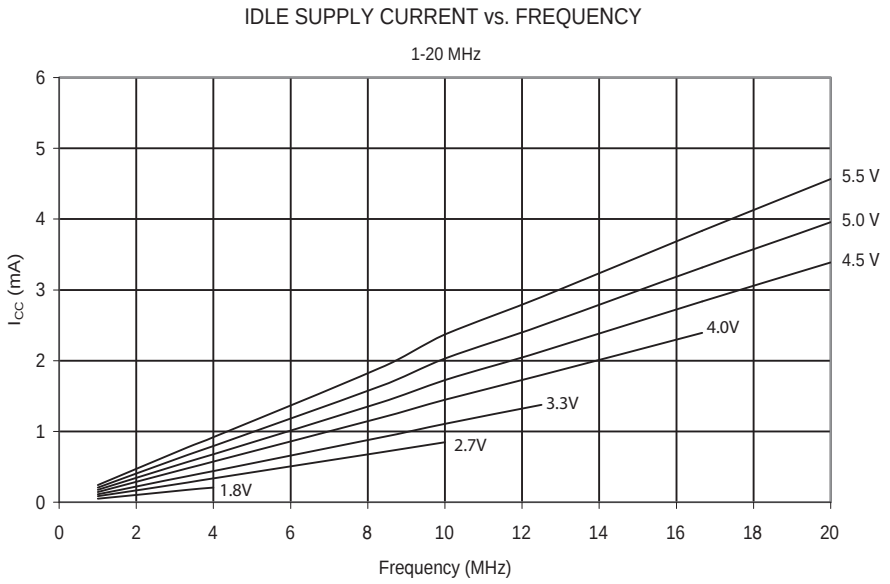


Figure 27-8. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz).

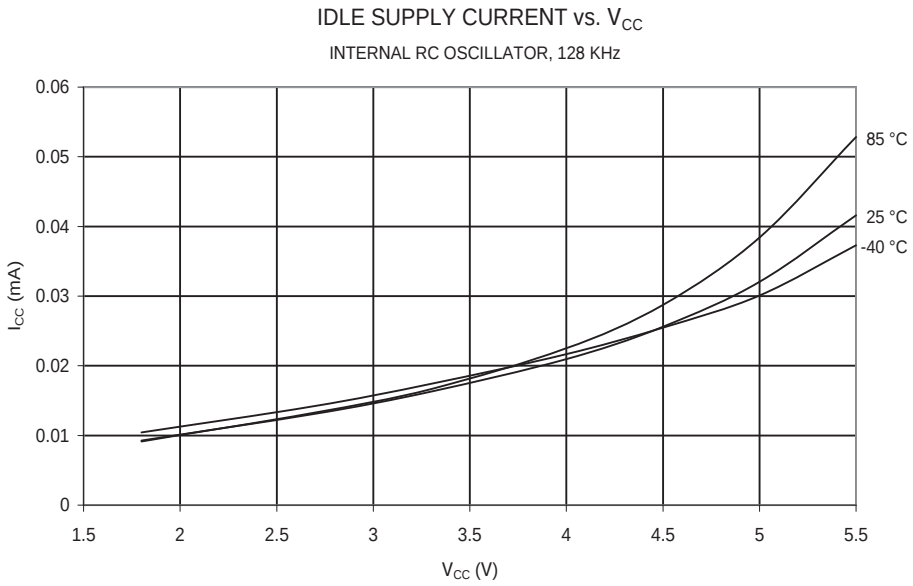


Figure 27-9. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz).

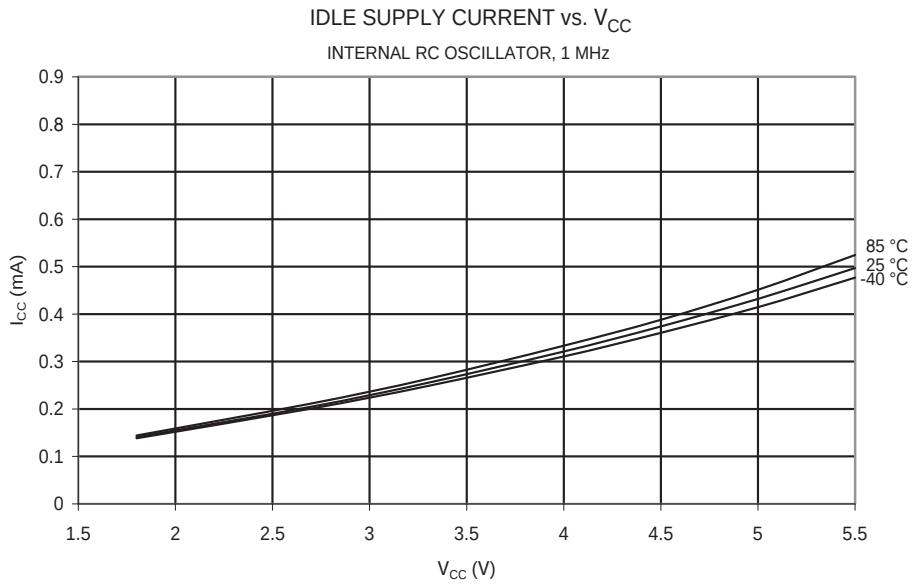
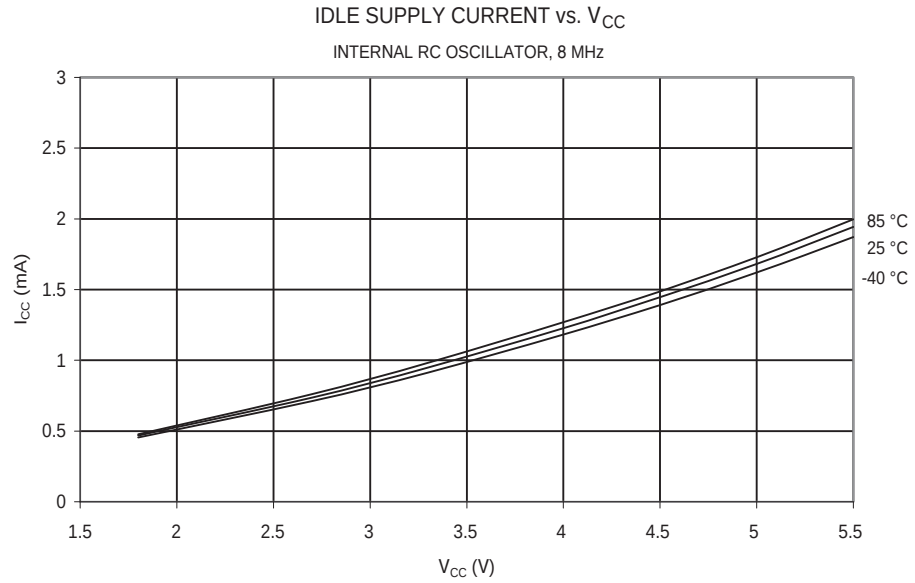


Figure 27-10. Idle Supply Current vs. Vcc (Internal RC Oscillator, 8 MHz).



27.1.3 Supply Current of IO Modules

The tables and formulas below can be used to calculate the additional current consumption for the different I/O modules in Active and Idle mode. The enabling or disabling of the I/O modules are controlled by the Power Reduction Register. See ”Power Reduction Register” on page 42 for details.

Table 27-1. Additional Current Consumption for the different I/O modules (absolute values)

PRR bit	Typical numbers		
	V_{CC} = 2V, F = 1MHz	V_{CC} = 3V, F = 4MHz	V_{CC} = 5V, F = 8MHz
PRUSART0	5.58 uA	35.6 uA	136.5 uA
PRTWI	8.97 uA	57.0 uA	231.5 uA
PRTIM2	9.84 uA	64.7 uA	263.5 uA
PRTIM1	9.80 uA	62.8 uA	247.9 uA
PRTIM0	2.31 uA	15.3 uA	60.9 uA
PRSPI	7.88 uA	51.9 uA	215.7 uA
PRADC	9.35 uA	60.5 uA	237.7 uA

Table 27-2. Additional Current Consumption (percentage) in Active and Idle mode

PRR bit	Additional Current consumption compared to Active with external clock (see Figure 27-1 on page 329 and Figure 27-2 on page 329)	Additional Current consumption compared to Idle with external clock (see Figure 27-6 on page 331 and Figure 27-7 on page 332)
PRUSART0	1.9%	9.1%
PRTWI	3.1%	14.8%
PRTIM2	3.5%	16.6%
PRTIM1	3.4%	16.1%
PRTIM0	0.8%	3.9%
PRSPI	2.8%	13.4%
PRADC	3.3%	15.5%

It is possible to calculate the typical current consumption based on the numbers from Table 27-2 on page 334 for other V_{CC} and frequency settings than listed in Table 27-1 on page 334.

Example

Calculate the expected current consumption in idle mode with TIMER1, ADC, and SPI enabled at V_{CC} = 2.0V and F = 1MHz. From Table 27-2 on page 334, third column, we see that we need to add 16.1% for the TIMER1, 15.5% for the ADC, and 13.4% for the SPI module. Reading from **Figure 27-6 on page 331**, we find that the idle current consumption is ~0.055 mA at V_{CC} = 2.0V and F = 1MHz. The total current consumption in idle mode with TIMER1, ADC, and SPI enabled, gives:

$$I_{CCtotal} \approx 0.055\ mA \cdot (1 + 0.161 + 0.155 + 0.134) \approx 0.080\ mA$$

27.1.4 Power-down Supply Current

Figure 27-11. Power-Down Supply Current vs. V_{CC} (Watchdog Timer Disabled).

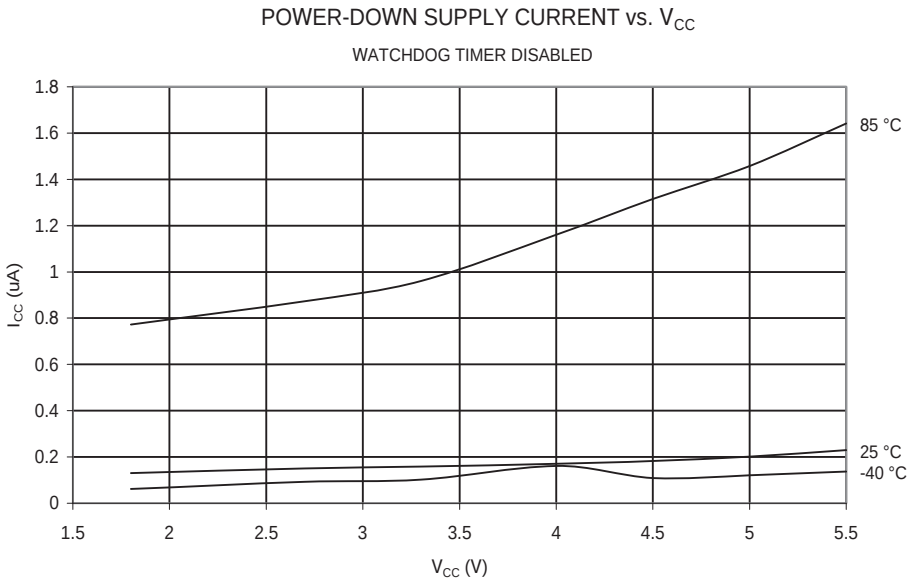
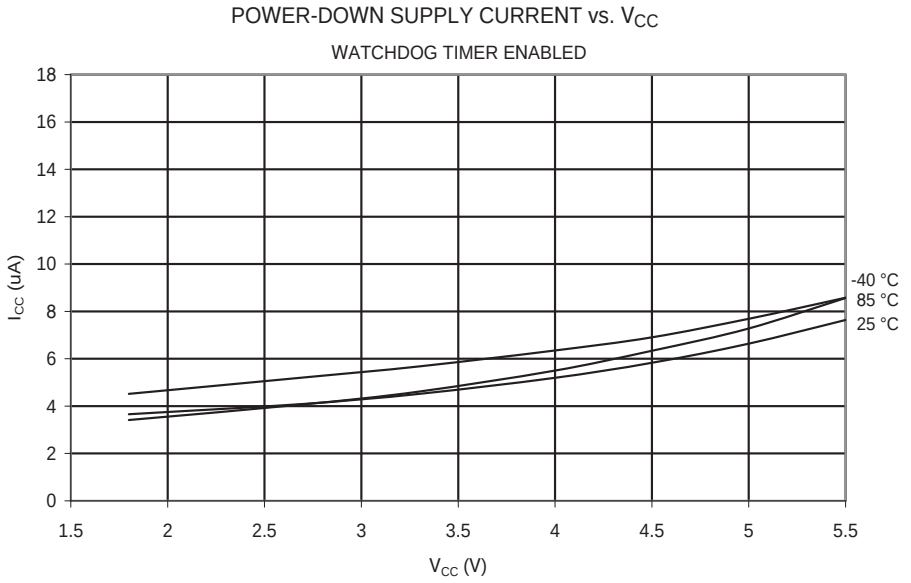
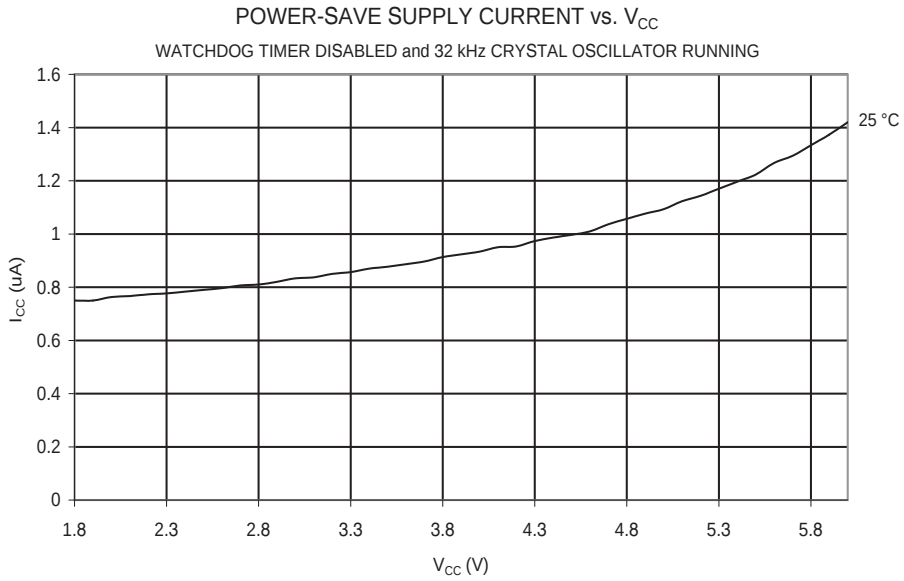


Figure 27-12. Power-Down Supply Current vs. V_{CC} (Watchdog Timer Enabled).



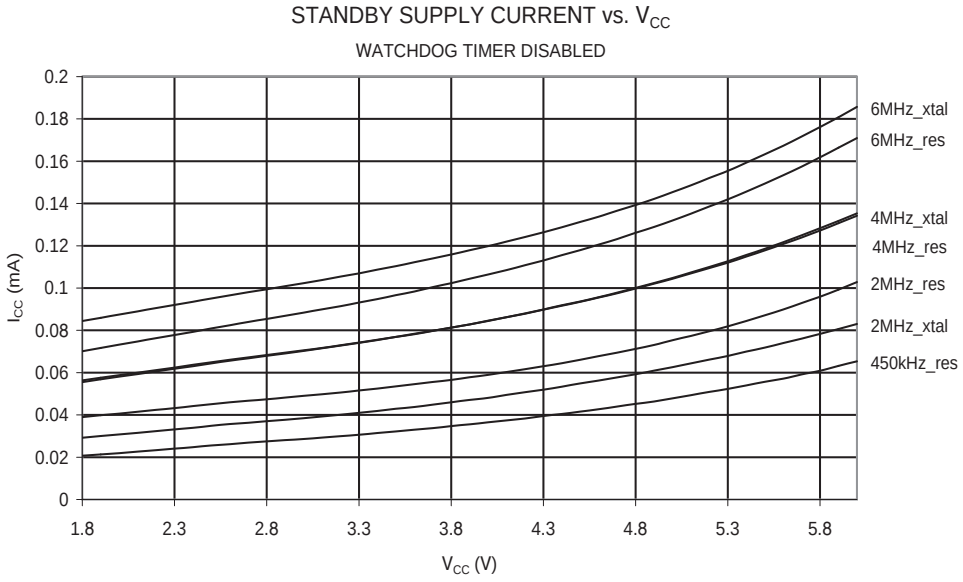
27.1.5 Power-save Supply Current

Figure 27-13. Power-Save Supply Current vs. V_{CC} (Watchdog Timer Disabled and 32 kHz Crystal Oscillator Running).



27.1.6 Standby Supply Current

Figure 27-14. Standby Supply Current vs. Vcc (Watchdog Timer Disabled).



27.1.7 Pin Pull-Up

Figure 27-15. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 1.8 V).

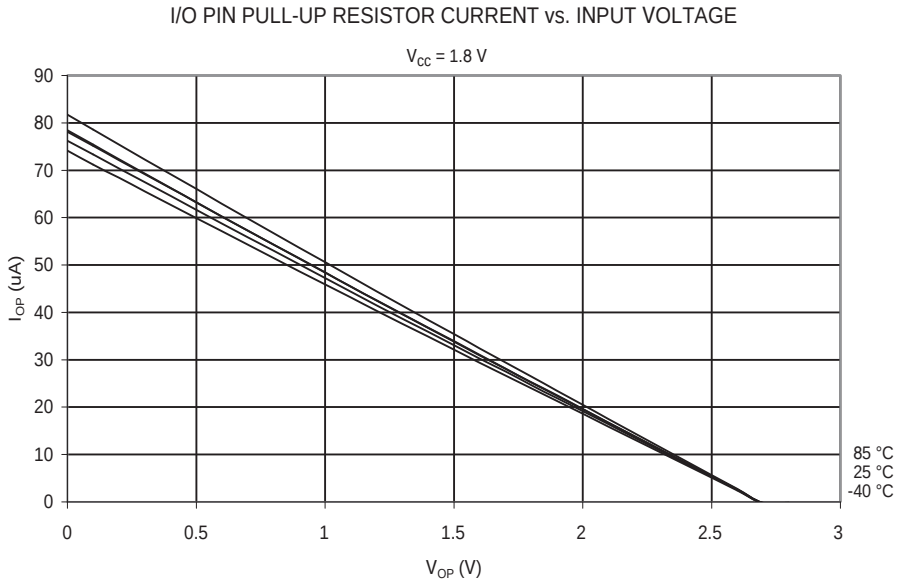


Figure 27-16. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 2.7 V).

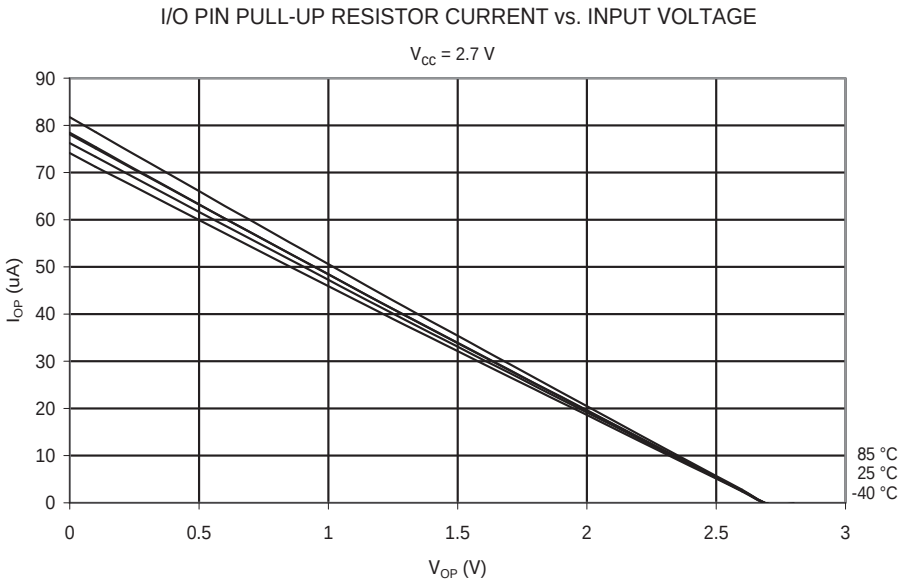


Figure 27-17. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 5 V).

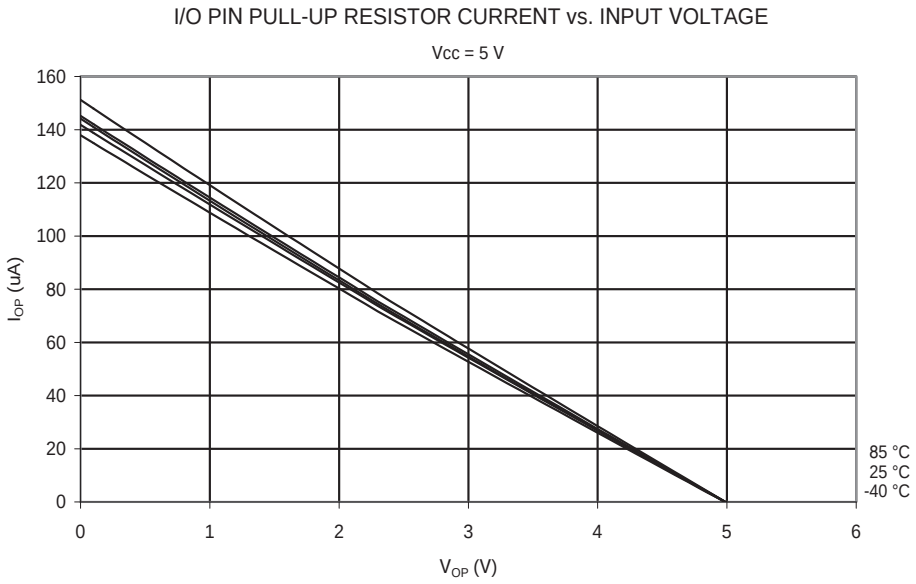


Figure 27-18. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 1.8 V).

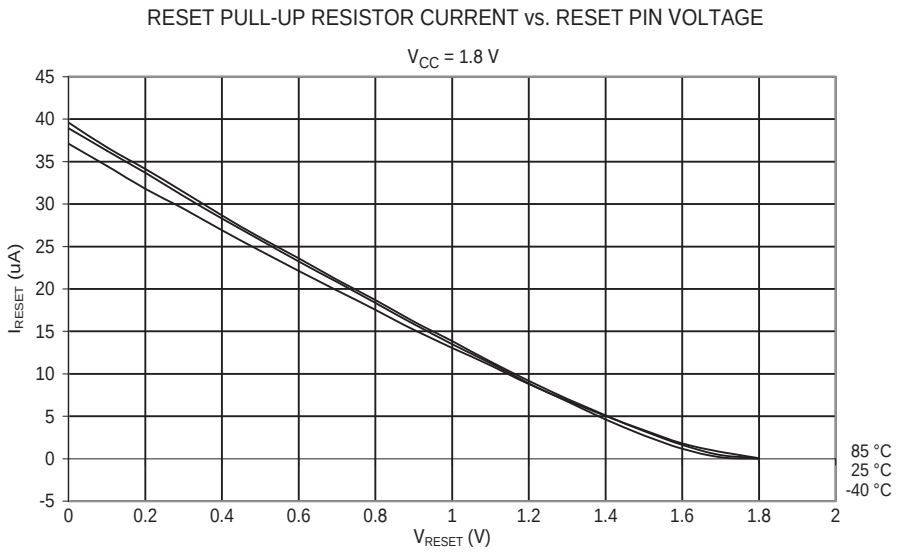


Figure 27-19. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 2.7 V).

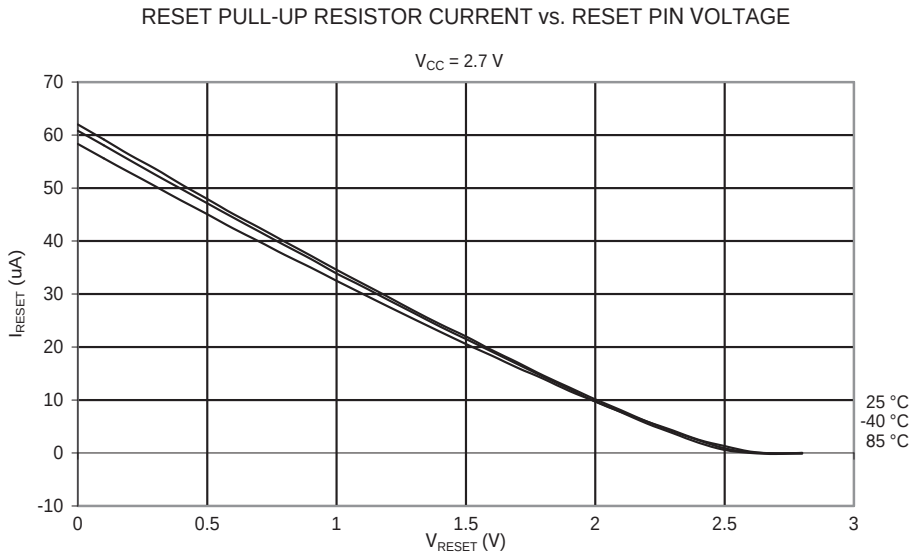
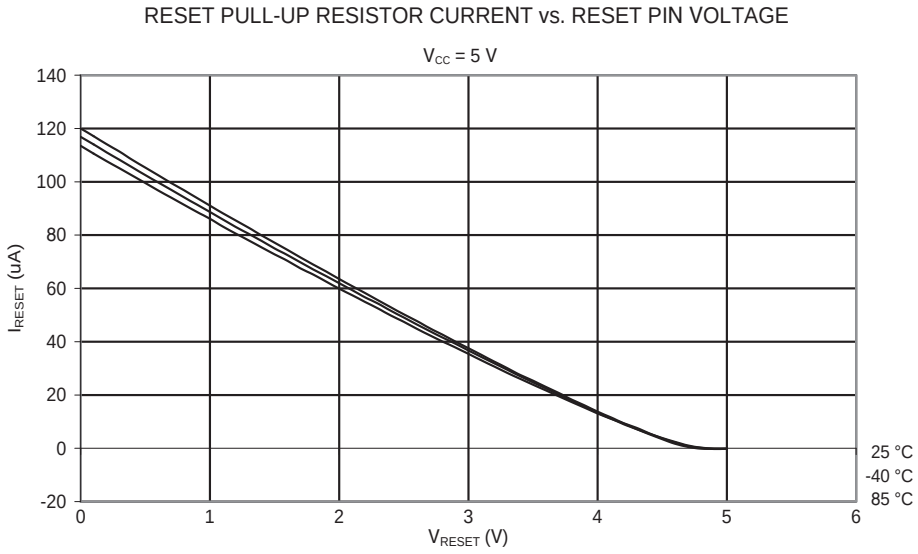


Figure 27-20. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 5 V).



27.1.8 Pin Driver Strength

Figure 27-21. I/O Pin Output Voltage vs. Sink Current (V_{CC} = 3 V).

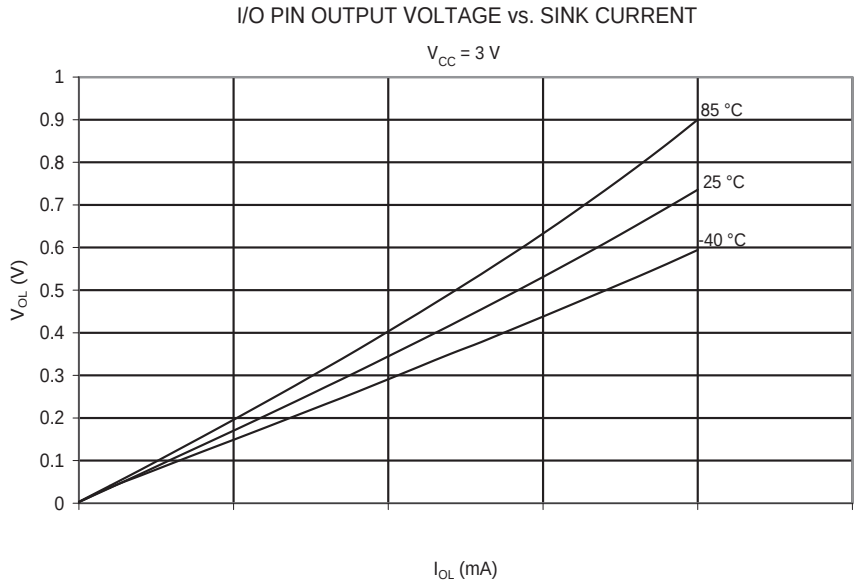


Figure 27-22. I/O Pin Output Voltage vs. Sink Current (V_{CC} = 5 V).

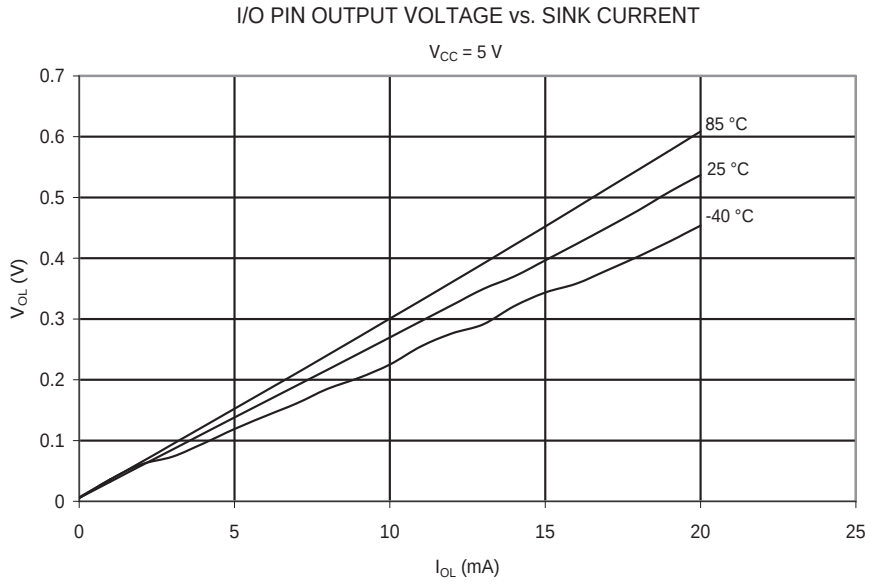


Figure 27-23. I/O Pin Output Voltage vs. Source Current (Vcc = 3 V).

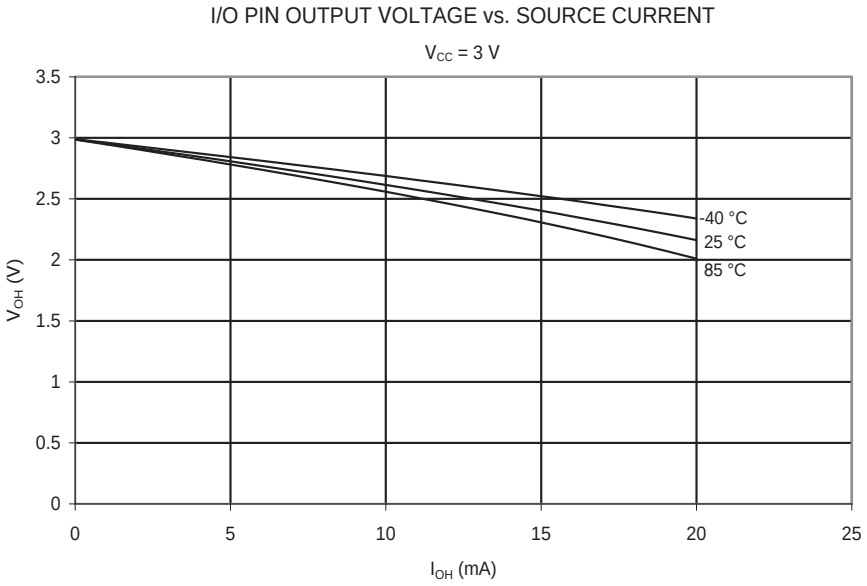
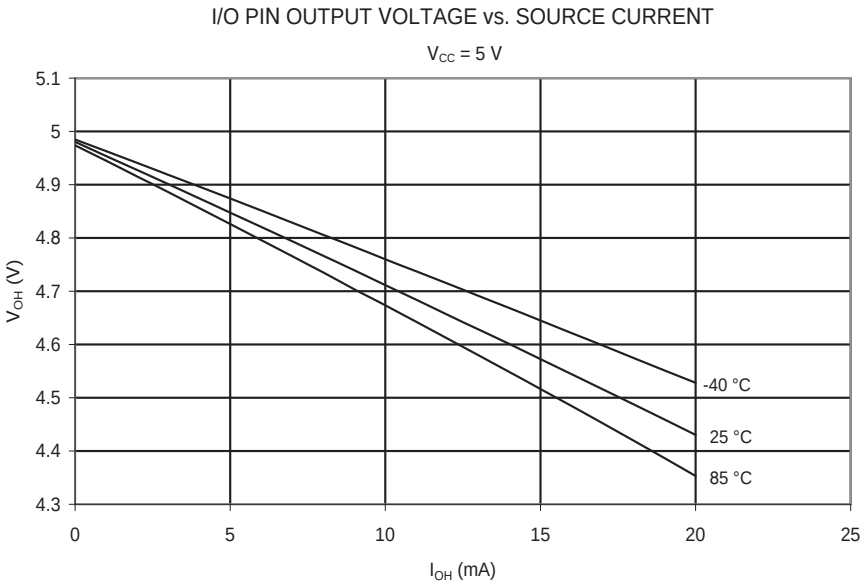


Figure 27-24. I/O Pin Output Voltage vs. Source Current(V_{CC} = 5 V).



27.1.9 Pin Threshold and Hysteresis

Figure 27-25. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin read as '1').

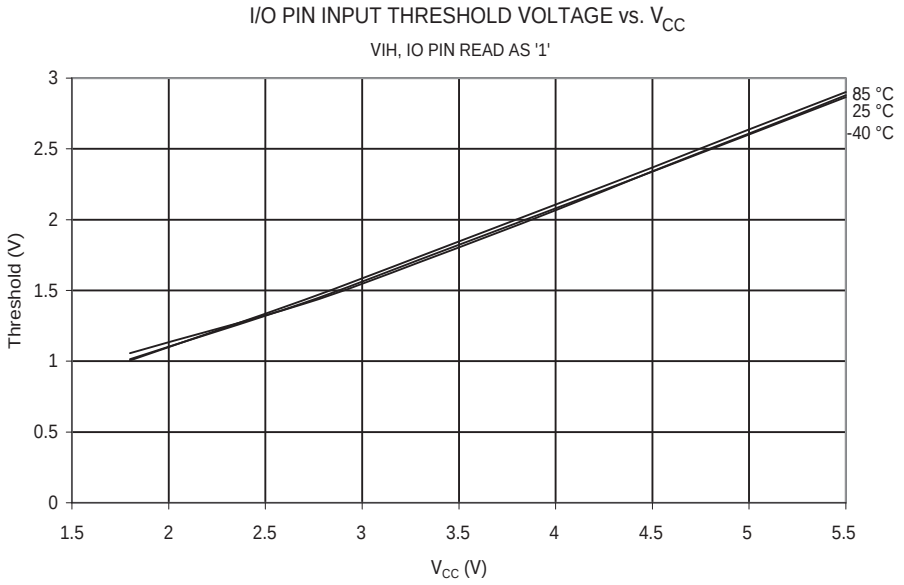


Figure 27-26. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O Pin read as '0').

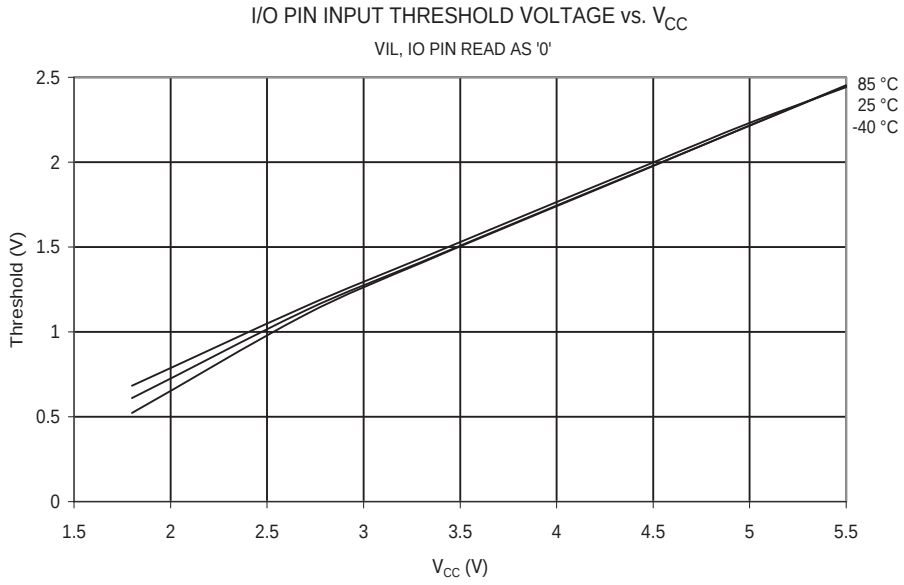


Figure 27-27. I/O Pin Input Hysteresis vs. V_{CC}.

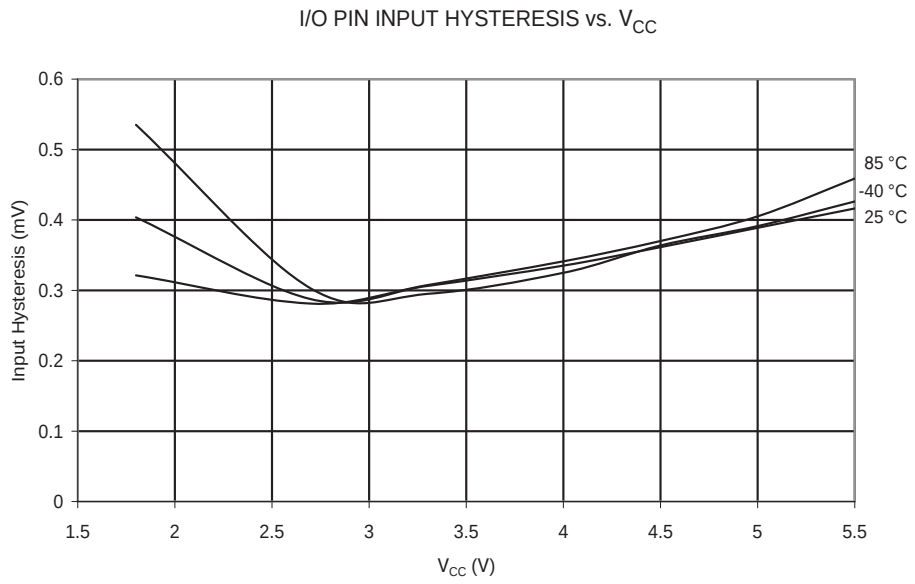


Figure 27-28. Reset Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin read as '1').

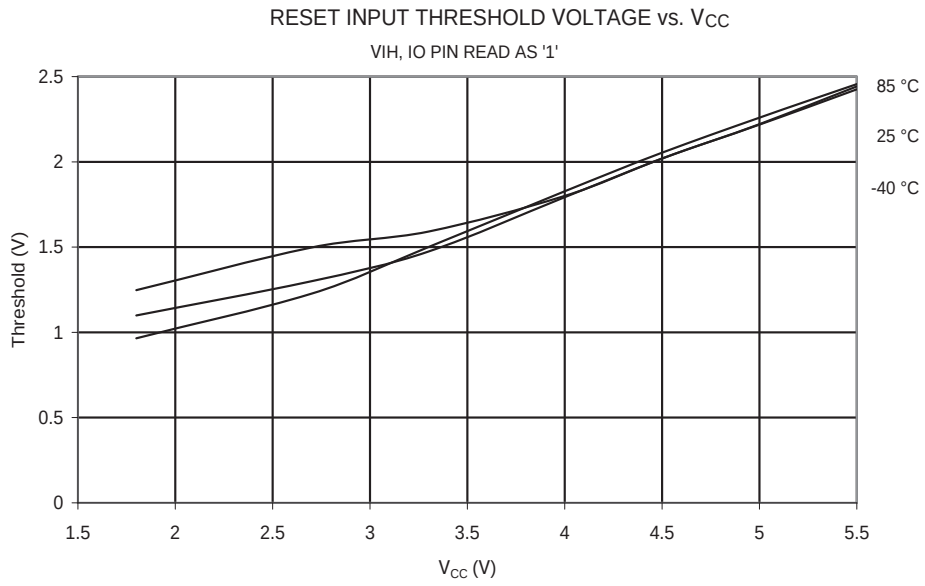


Figure 27-29. Reset Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O Pin read as '0').

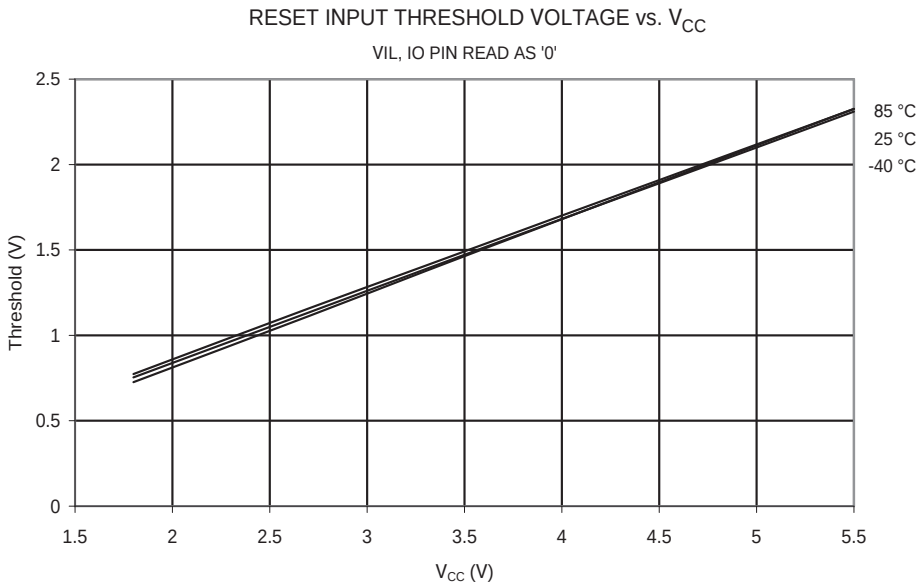
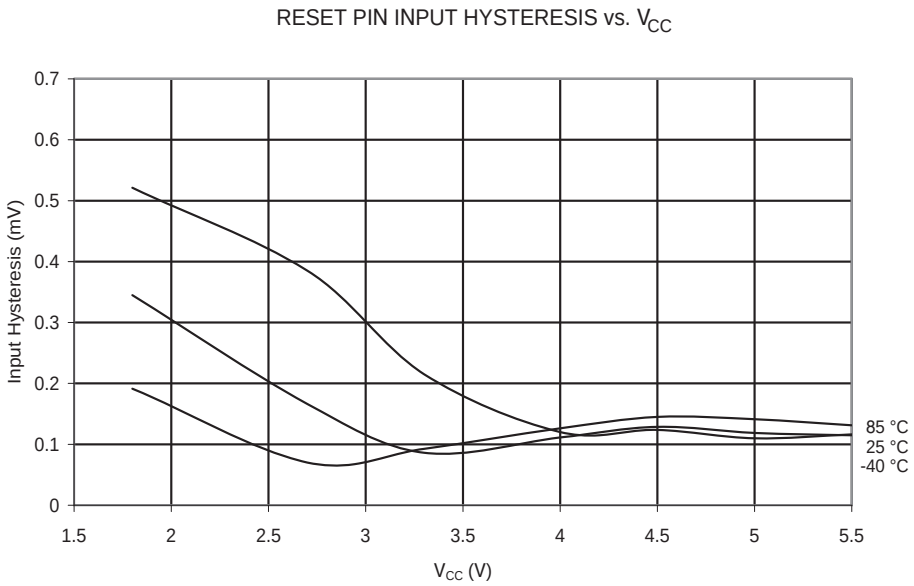


Figure 27-30. Reset Pin Input Hysteresis vs. V_{CC}.



27.1.10 BOD Threshold

Figure 27-31. BOD Thresholds vs. Temperature (BODLEVEL is 1.8 V).

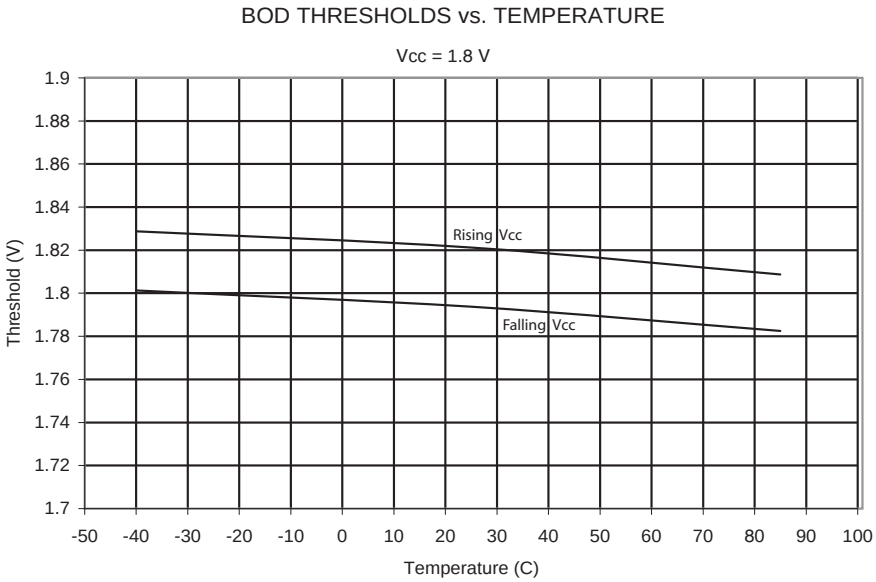


Figure 27-32. BOD Thresholds vs. Temperature (BODLEVEL is 2.7 V).

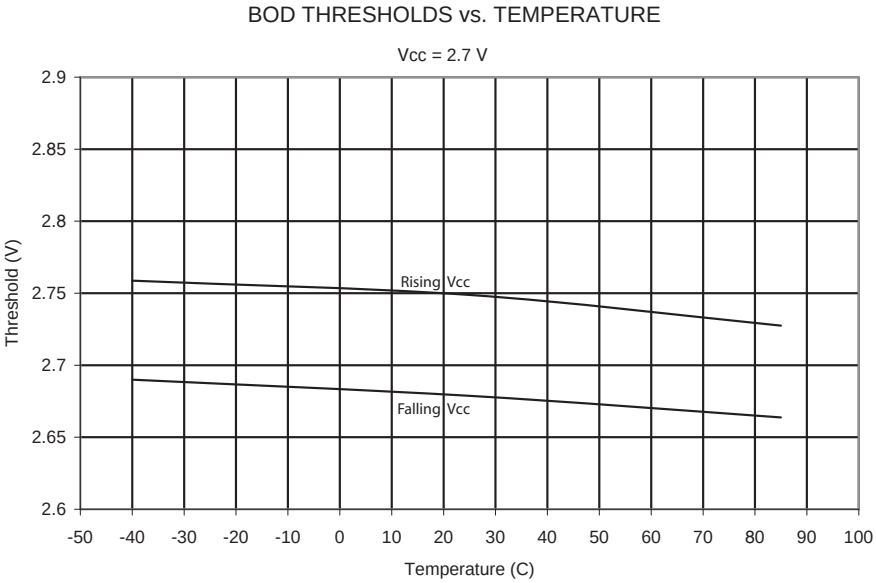
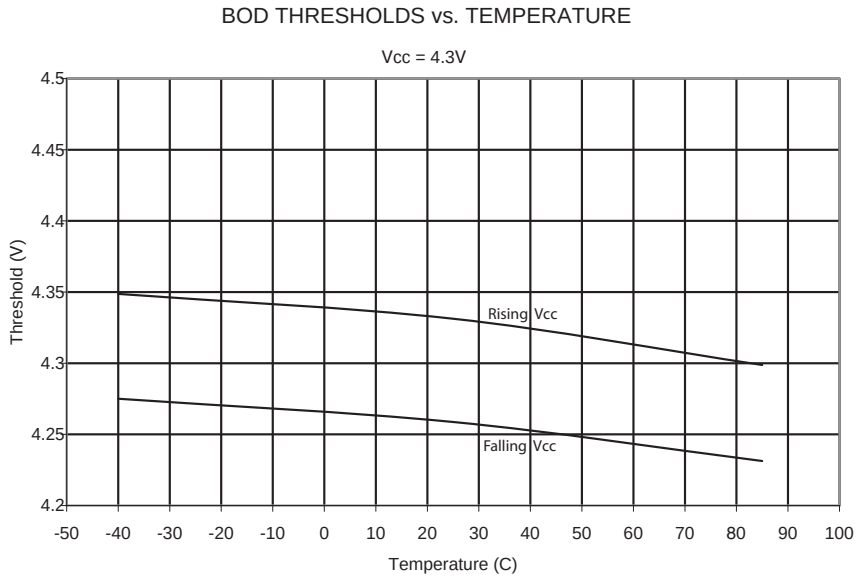


Figure 27-33. BOD Thresholds vs. Temperature (BODLEVEL is 4.3 V).



27.1.11 Internal Oscilllator Speed

Figure 27-34. Watchdog Oscillator Frequency vs. Temperature.

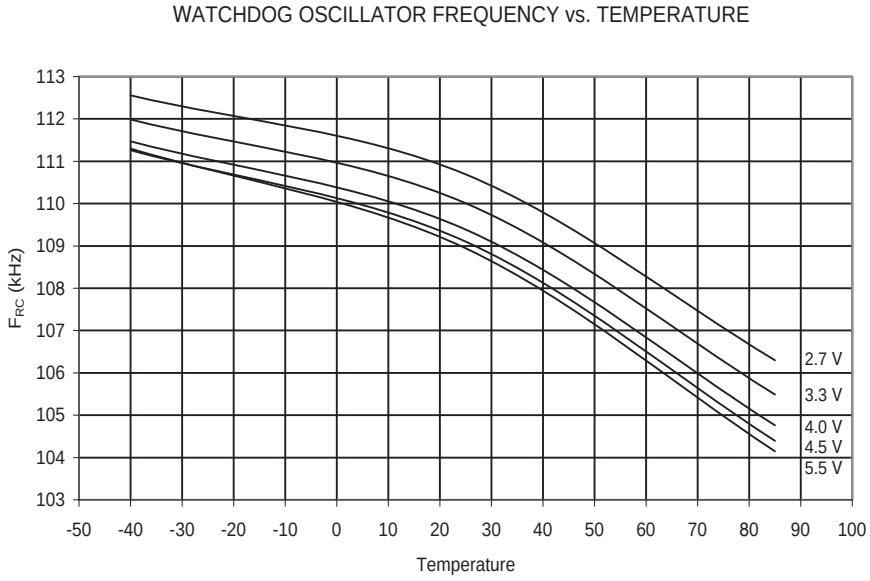


Figure 27-35. Watchdog Oscillator Frequency vs. V_{CC}.

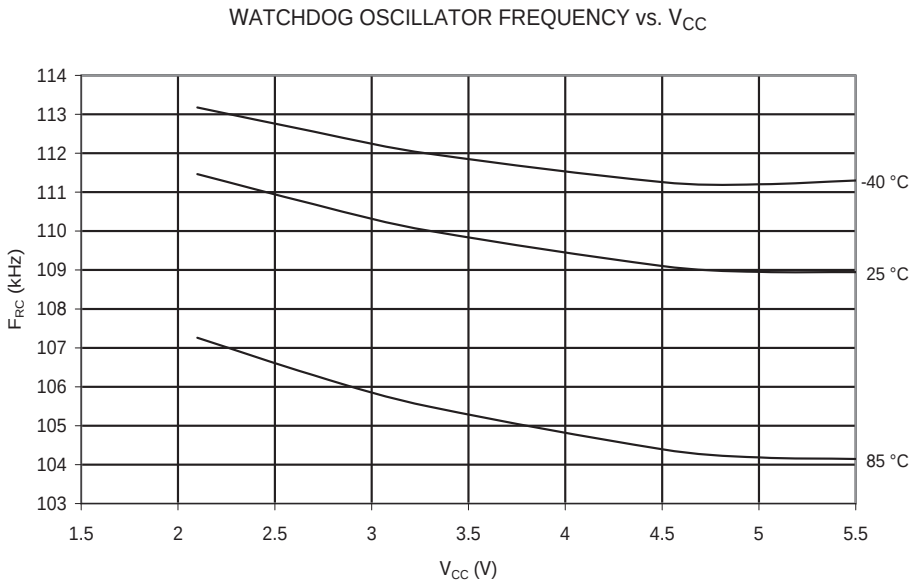


Figure 27-36. Calibrated 8 MHz RC Oscillator Frequency vs. V_{CC}.

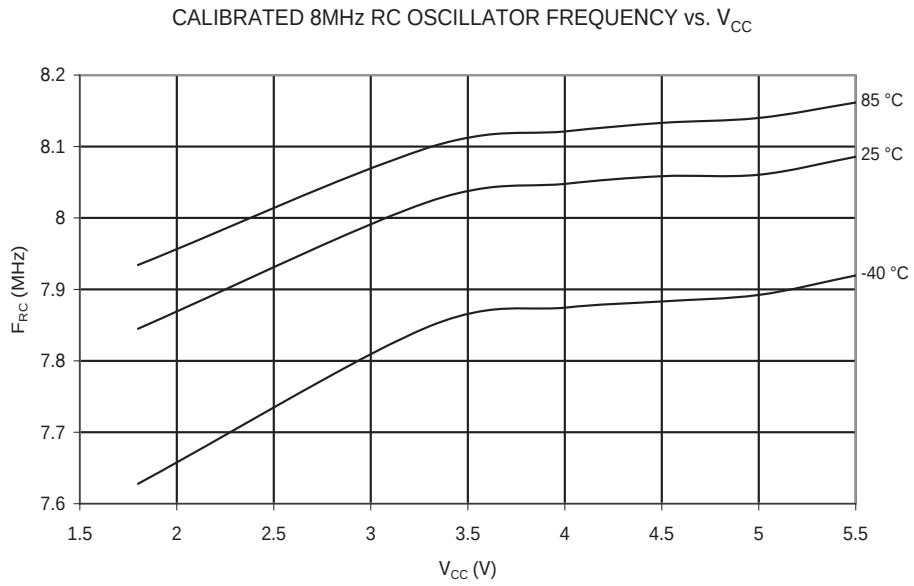


Figure 27-37. Calibrated 8 MHz RC Oscillator Frequency vs. Temperature.

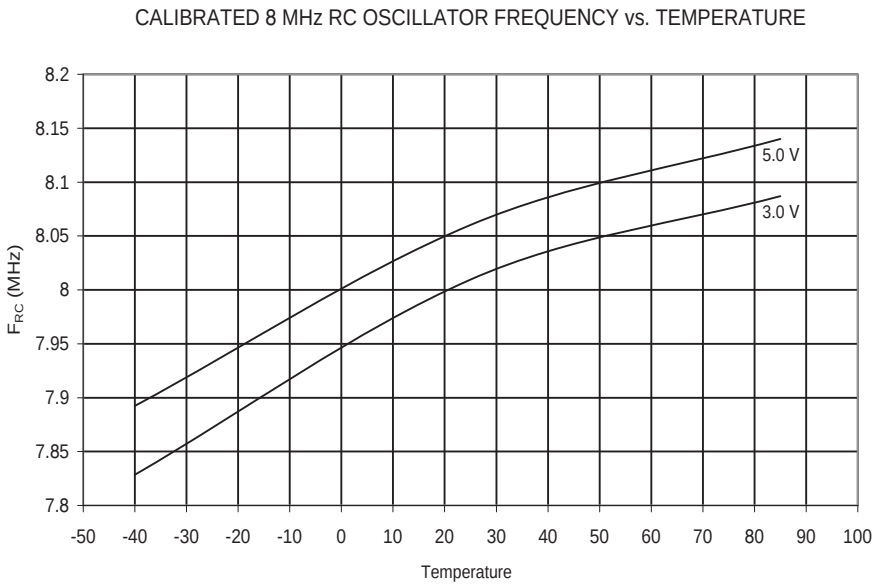
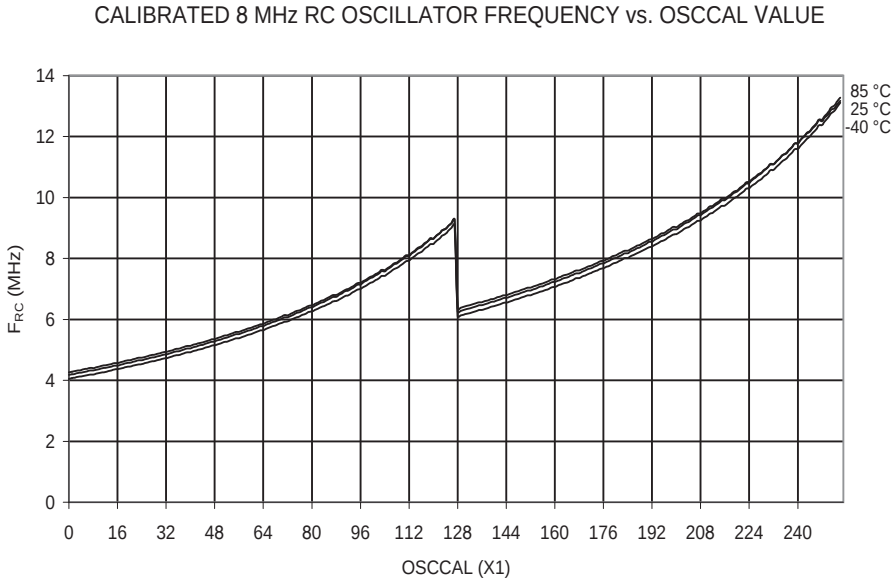


Figure 27-38. Calibrated 8 MHz RC Oscillator Frequency vs. OSCCAL Value.



27.1.12 Current Consumption of Peripheral Units

Figure 27-39. ADC Current vs. V_{CC} ($AREF = AV_{CC}$).

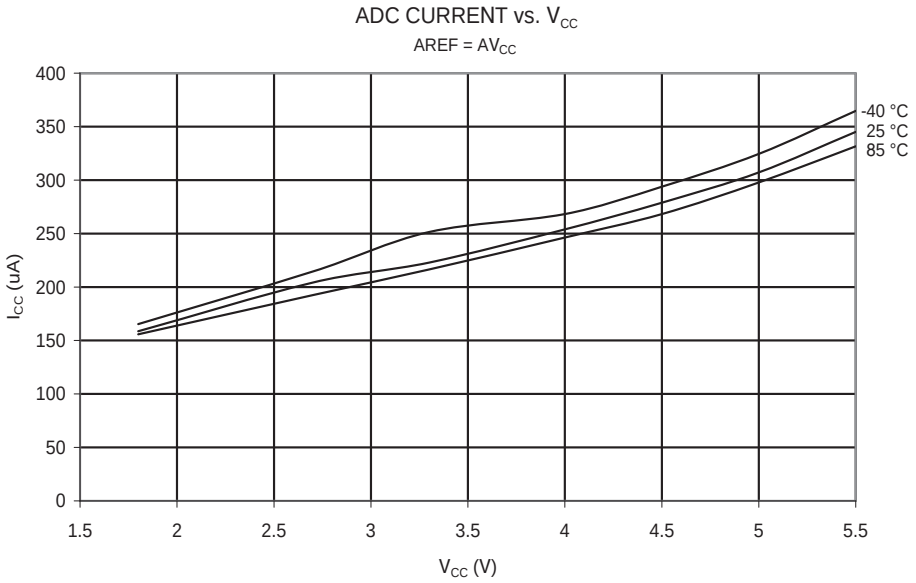


Figure 27-40. Analog Comparator Current vs. V_{CC}.

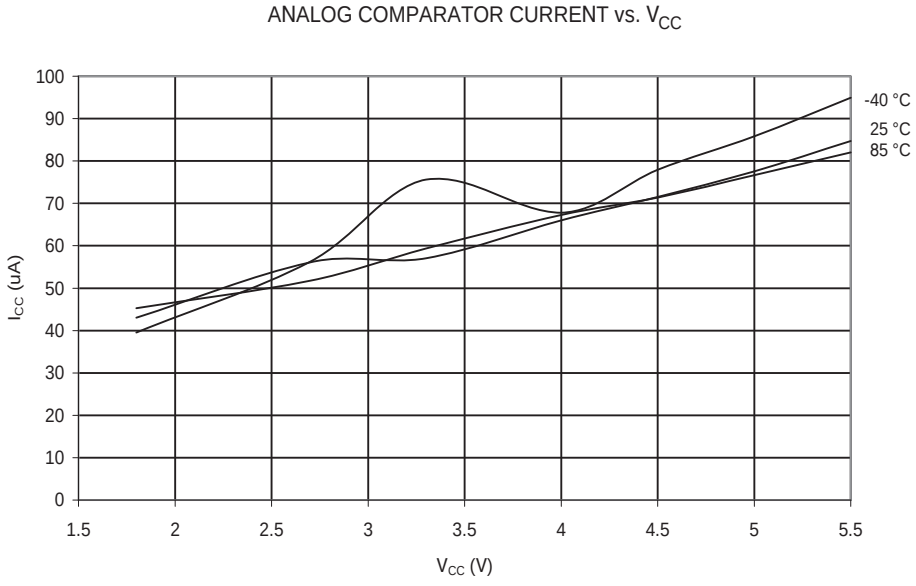


Figure 27-41. AREF External Reference Current vs. V_{CC}.

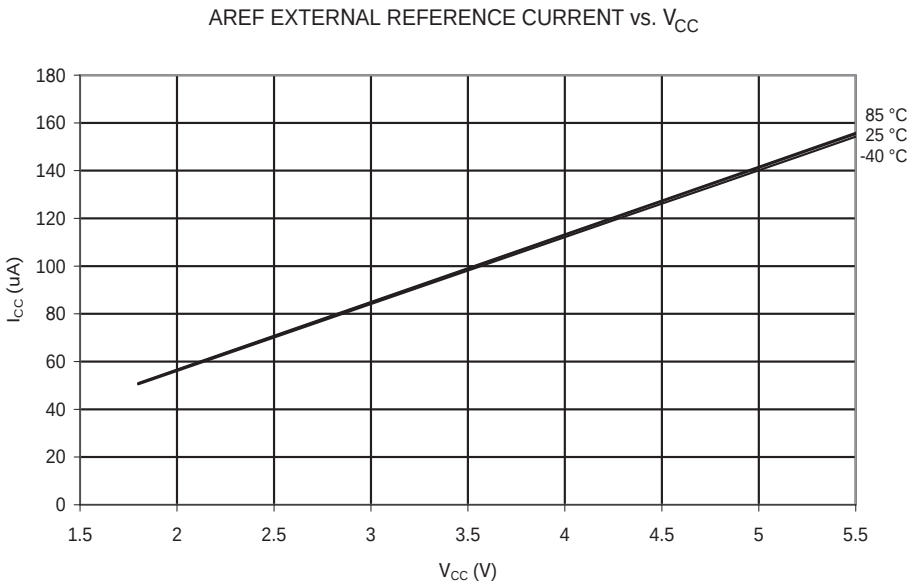


Figure 27-42. Brownout Detector Current vs. V_{CC}.

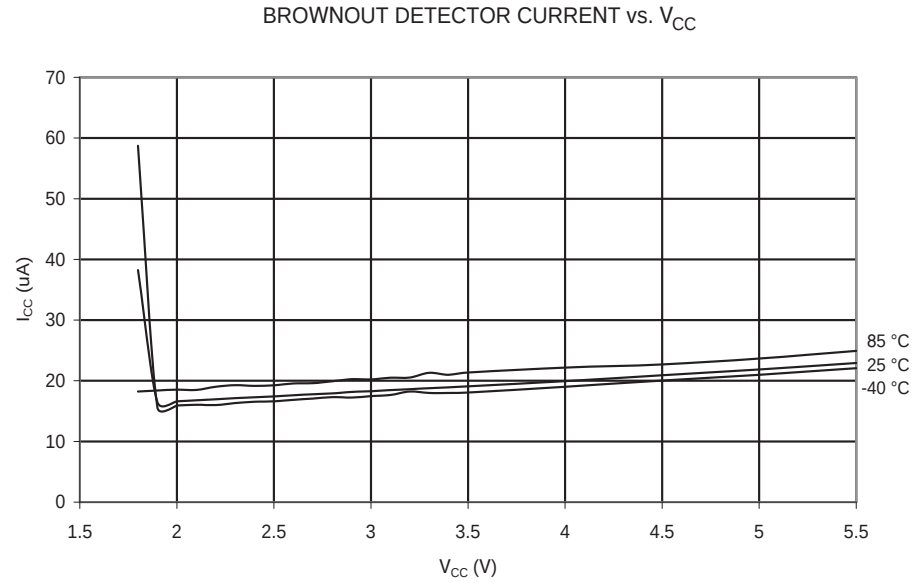
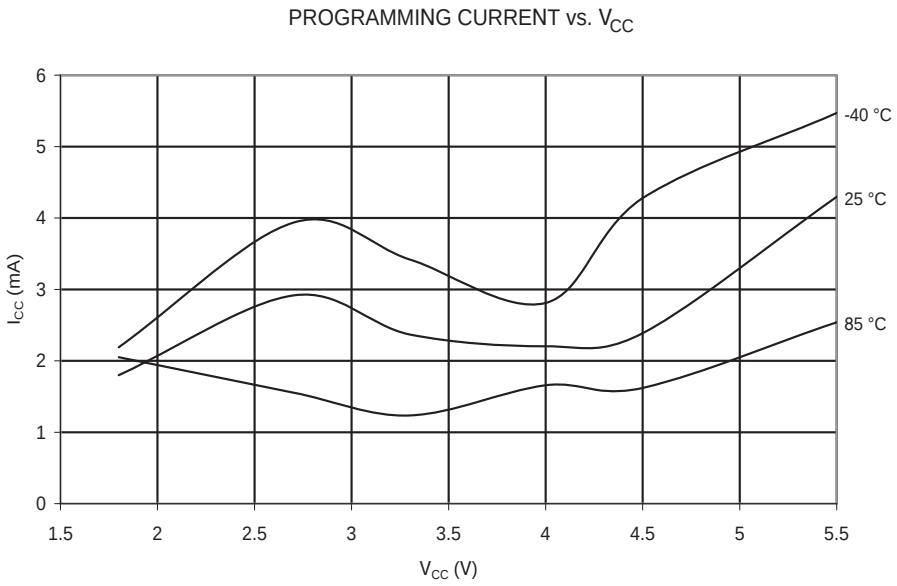


Figure 27-43. Programming Current vs. V_{CC}.



27.1.13 Current Consumption in Reset and Reset Pulsewidth

Figure 27-44. Reset Supply Current vs. Low Frequency (0.1 - 1.0 MHz).

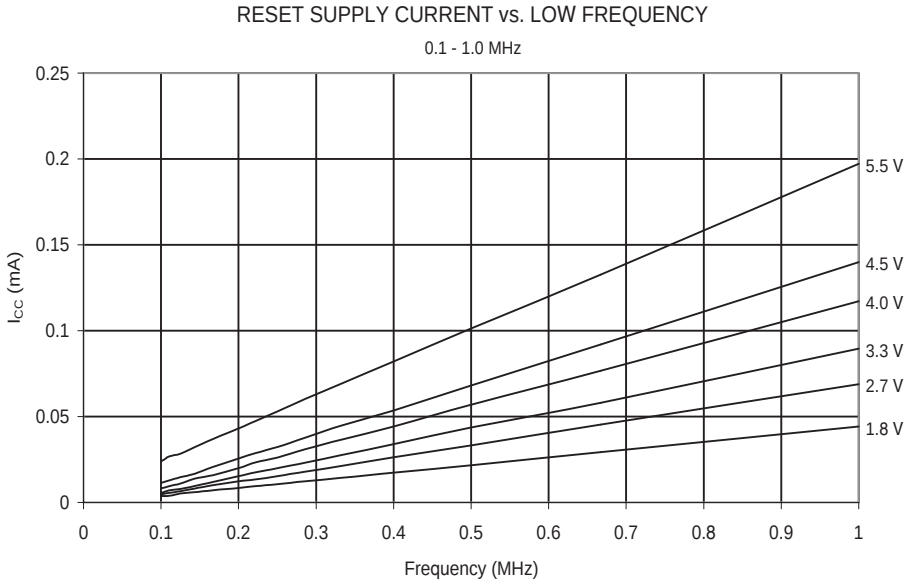


Figure 27-45. Reset Supply Current vs. Frequency (1 - 20 MHz).

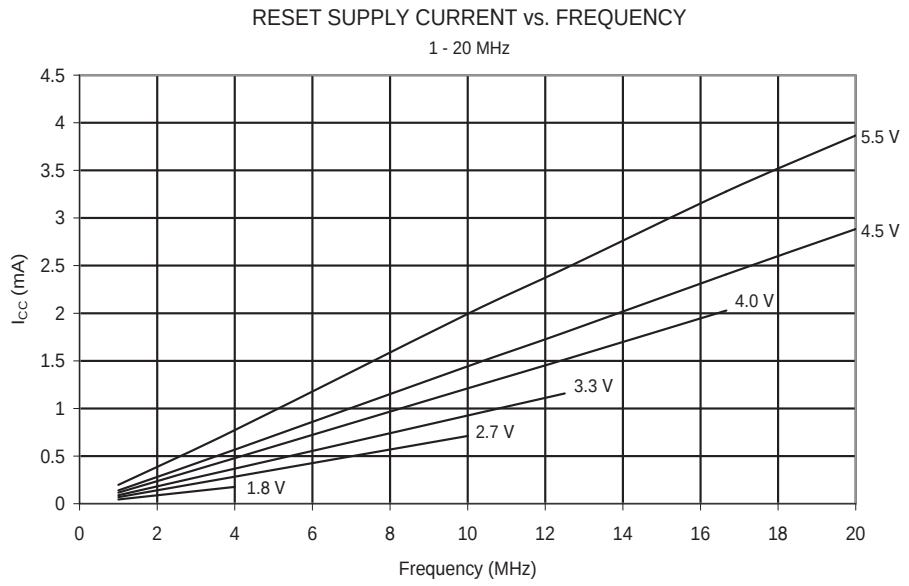
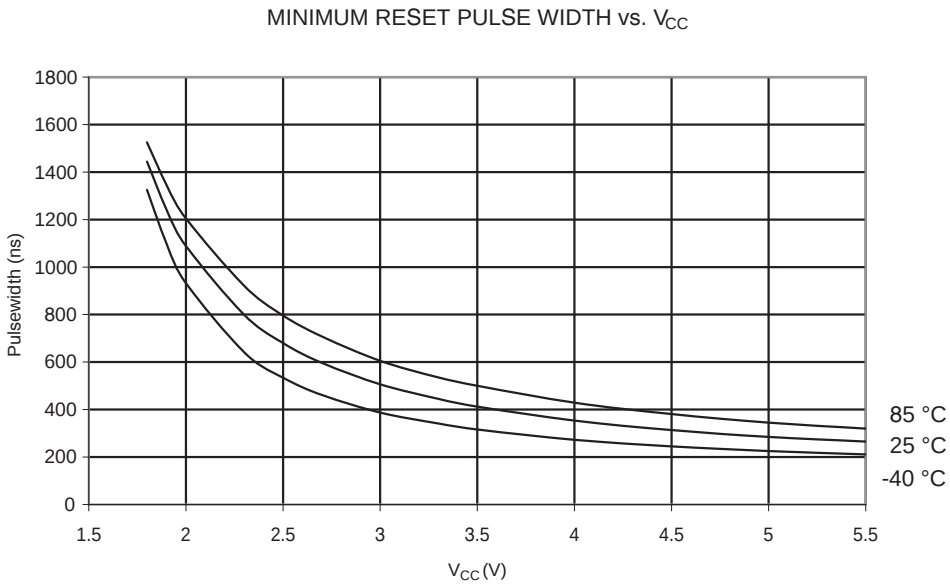


Figure 27-46. Minimum Reset Pulse width vs. V_{CC}.



27.2 ATmega88P Typical Characteristics

27.2.1 Active Supply Current

Figure 27-47. Active Supply Current vs. Low Frequency (0.1-1.0 MHz).

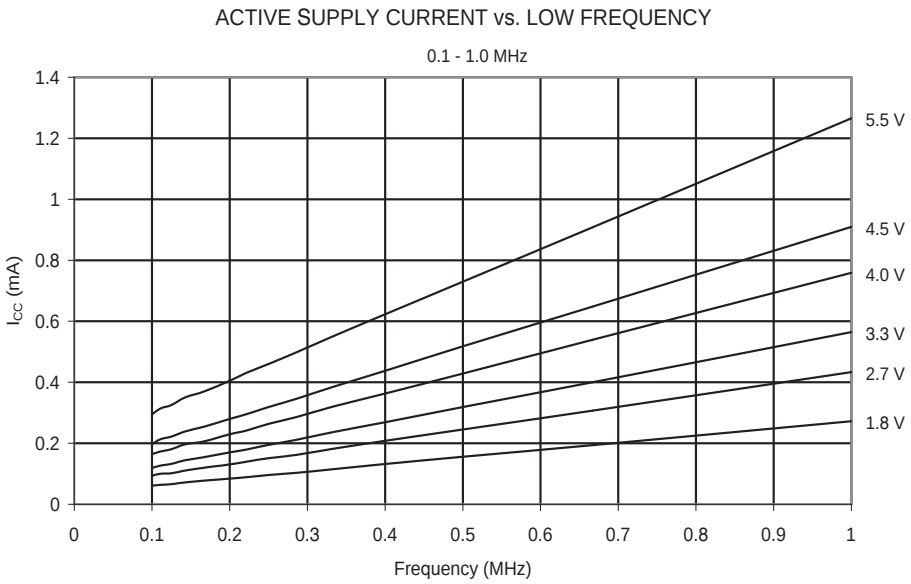


Figure 27-48. Active Supply Current vs. Frequency (1-20 MHz).

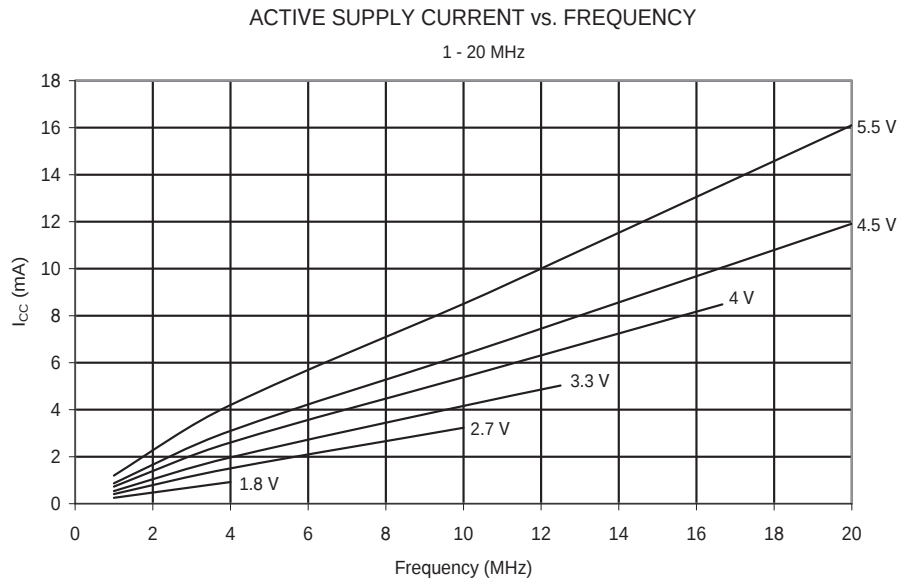


Figure 27-49. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz).

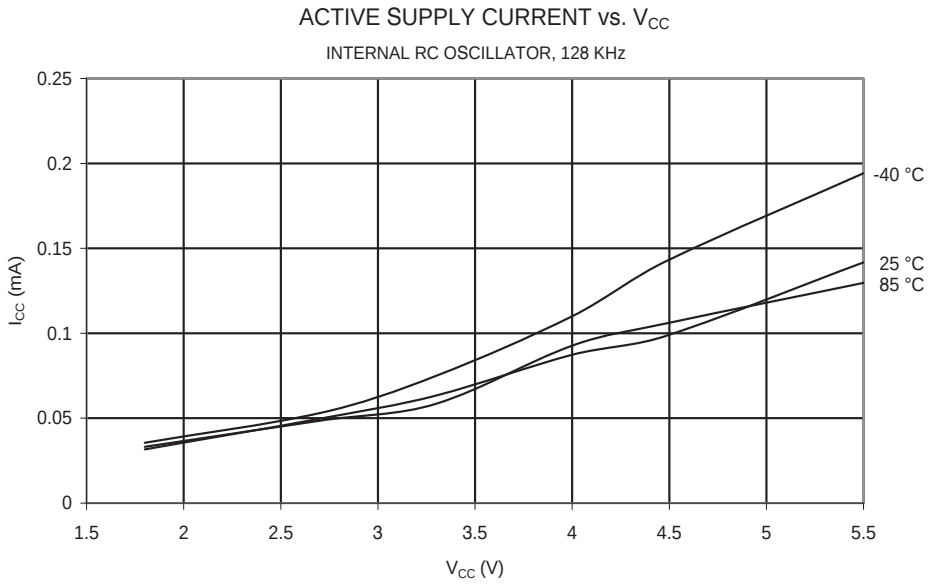


Figure 27-50. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz).

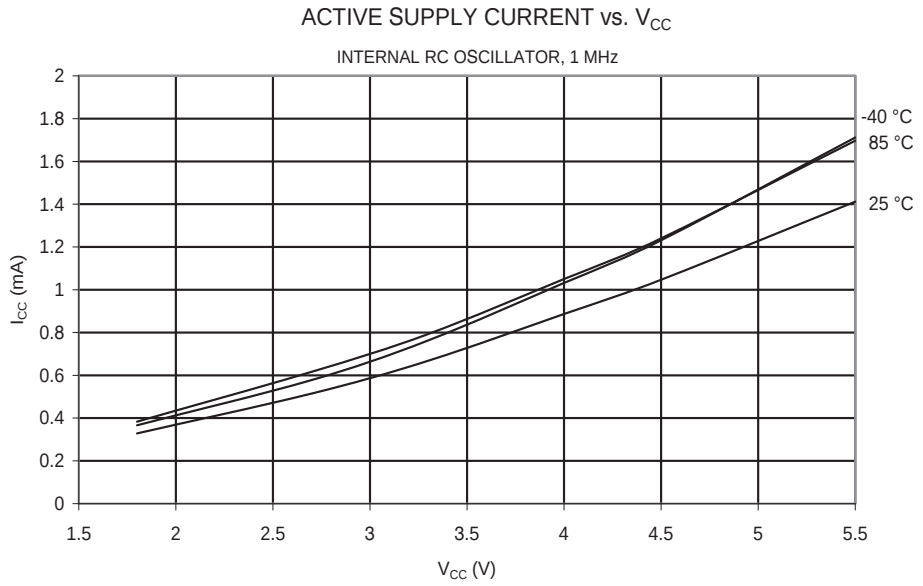
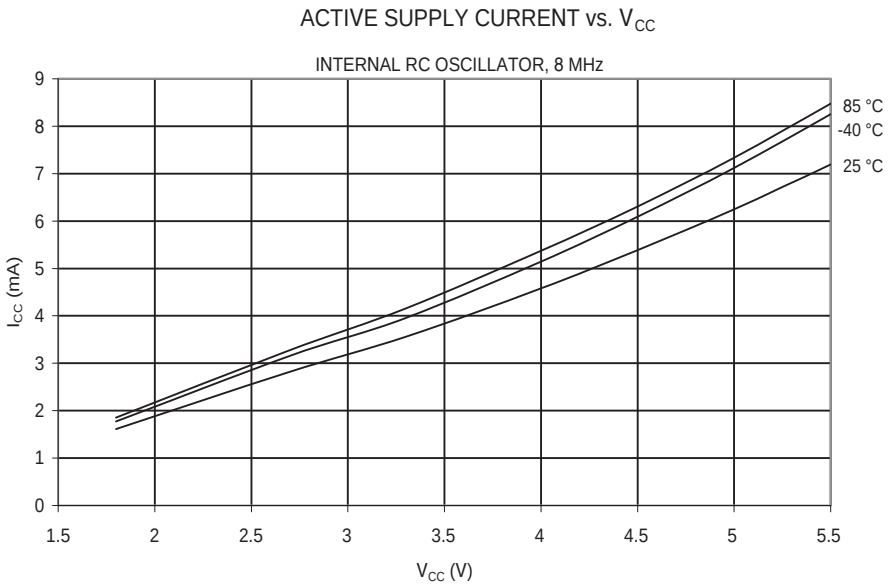


Figure 27-51. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 8 MHz).



27.2.2 Idle Supply Current

Figure 27-52. Idle Supply Current vs. Low Frequency (0.1-1.0 MHz).

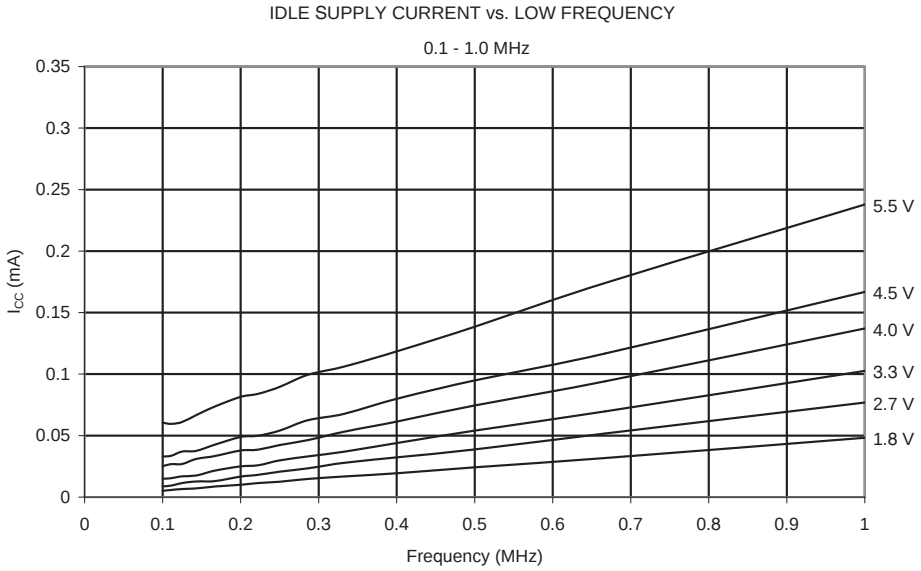


Figure 27-53. Idle Supply Current vs. Frequency (1-20 MHz).

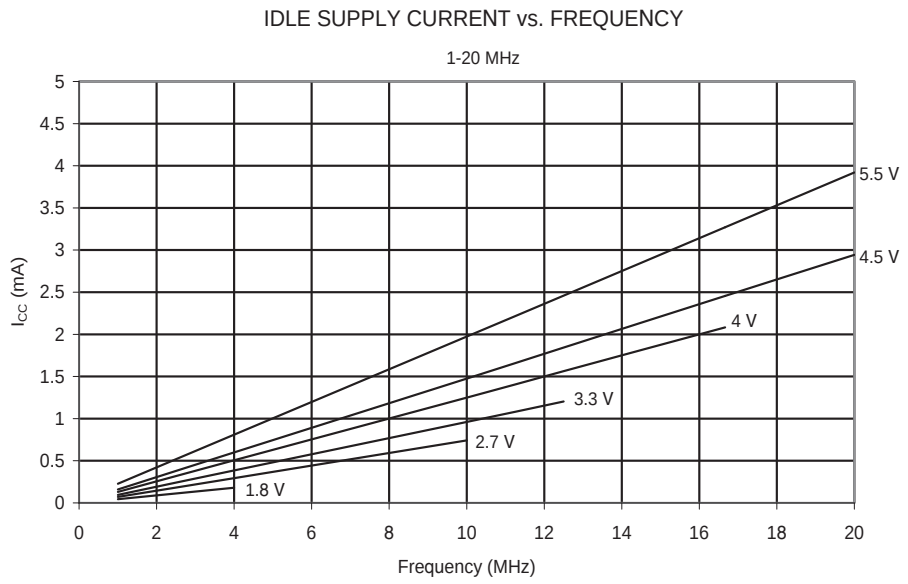


Figure 27-54. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz).

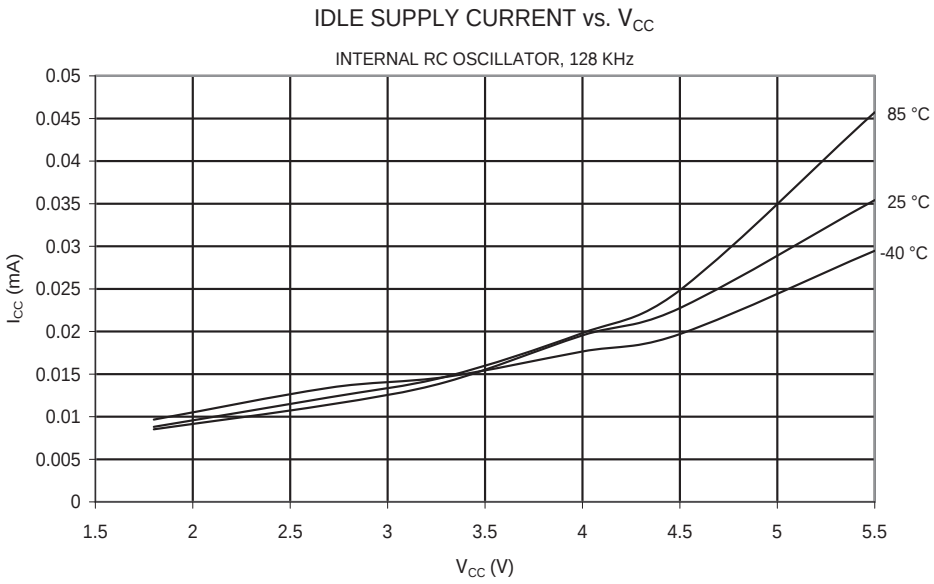


Figure 27-55. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz).

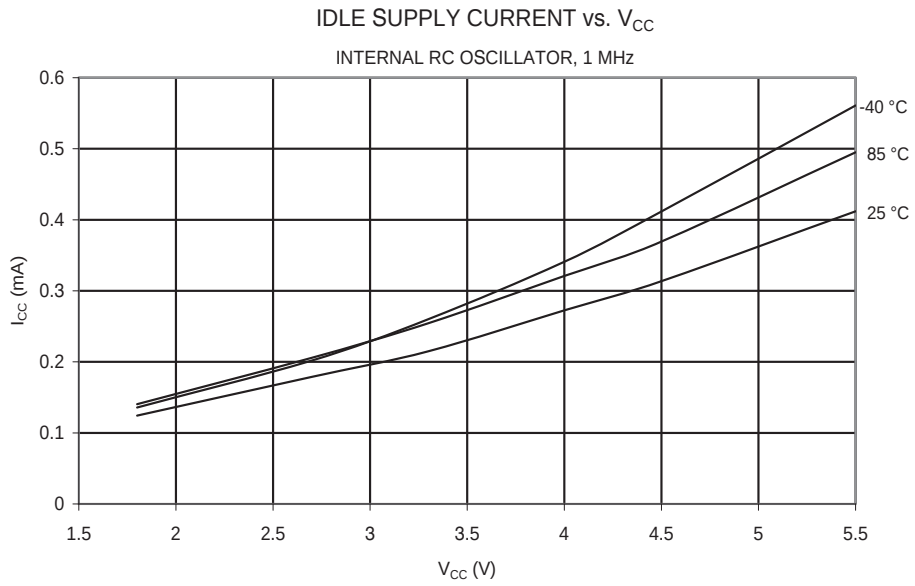
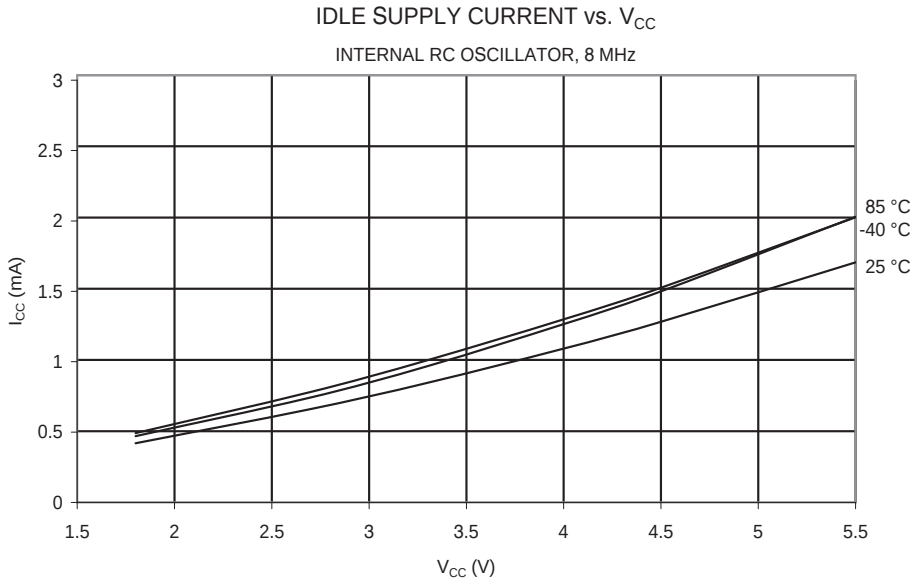


Figure 27-56. Idle Supply Current vs. Vcc (Internal RC Oscillator, 8 MHz).



27.2.3 Supply Current of IO Modules

The tables and formulas below can be used to calculate the additional current consumption for the different I/O modules in Active and Idle mode. The enabling or disabling of the I/O modules are controlled by the Power Reduction Register. See "Power Reduction Register" on page 42 for details.

Table 27-3. Additional Current Consumption for the different I/O modules (absolute values)

PRR bit	Typical numbers		
	V_{CC} = 2V, F = 1MHz	V_{CC} = 3V, F = 4MHz	V_{CC} = 5V, F = 8MHz
PRUSART0	4.12 uA	26.7 uA	108.3 uA
PRTWI	8.96 uA	58.6 uA	238.2 uA
PRTIM2	9.94 uA	64.1 uA	256.3 uA
PRTIM1	8.81 uA	56.9 uA	227.0 uA
PRTIM0	2.29 uA	15.5 uA	62.3 uA
PRSPI	8.31 uA	56.8 uA	260.4 uA
PRADC	9.27 uA	58.4 uA	230.8 uA

Table 27-4. Additional Current Consumption (percentage) in Active and Idle mode

PRR bit	Additional Current consumption compared to Active with external clock (see Figure 27-47 on page 353 and Figure 27-48 on page 353)	Additional Current consumption compared to Idle with external clock (see Figure 27-52 on page 355 and Figure 27-53 on page 356)
PRUSART0	1.3%	8.0%
PRTWI	2.9%	17.6%
PRTIM2	3.2%	19.2%
PRTIM1	2.8%	17.0%
PRTIM0	0.8%	4.6%
PRSPI	3.0%	17.5%
PRADC	2.9%	17.6%

It is possible to calculate the typical current consumption based on the numbers from Table 27-4 on page 358 for other V_{CC} and frequency settings than listed in Table 27-3 on page 358.

Example

Calculate the expected current consumption in idle mode with TIMER1, ADC, and SPI enabled at V_{CC} = 2.0V and F = 1MHz. From Table 27-4 on page 358, third column, we see that we need to add 17.0% for the TIMER1, 17.6% for the ADC, and 17.5% for the SPI module. Reading from **Figure 27-52 on page 355**, we find that the idle current consumption is ~0.055 mA at V_{CC} = 2.0V and F = 1MHz. The total current consumption in idle mode with TIMER1, ADC, and SPI enabled, gives:

$$I_{CCtotal} \approx 0.055\ mA \cdot (1 + 0.170 + 0.176 + 0.175) \approx 0.084\ mA$$

27.2.4 Power-down Supply Current

Figure 27-57. Power-Down Supply Current vs. V_{CC} (Watchdog Timer Disabled).

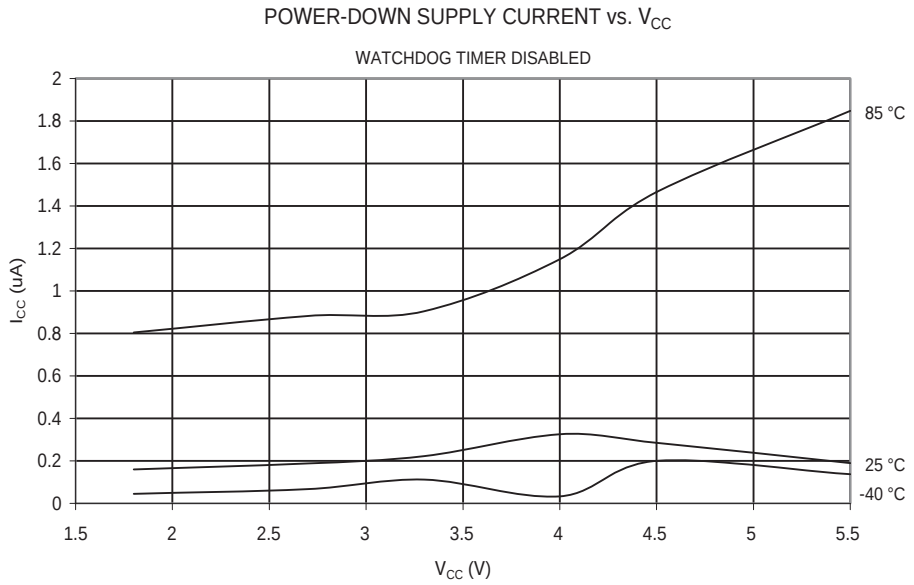
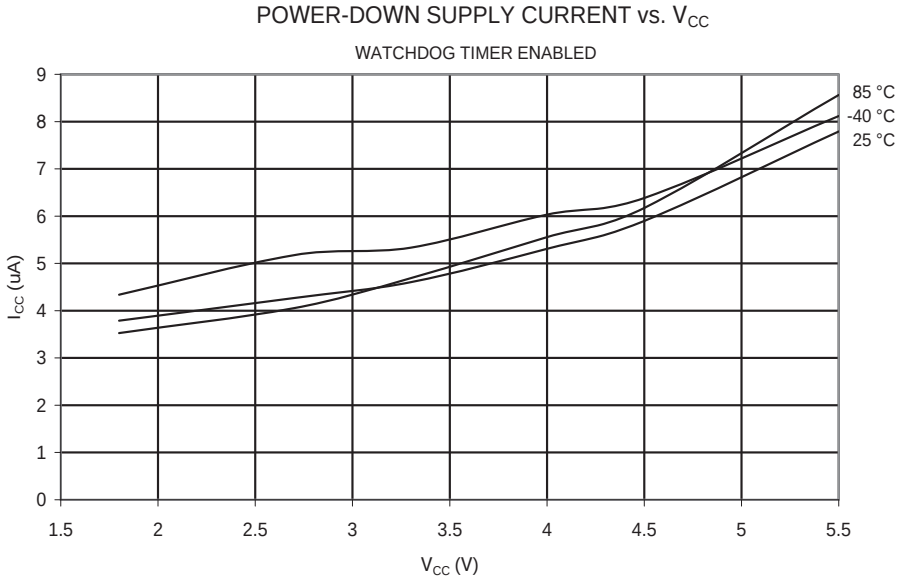
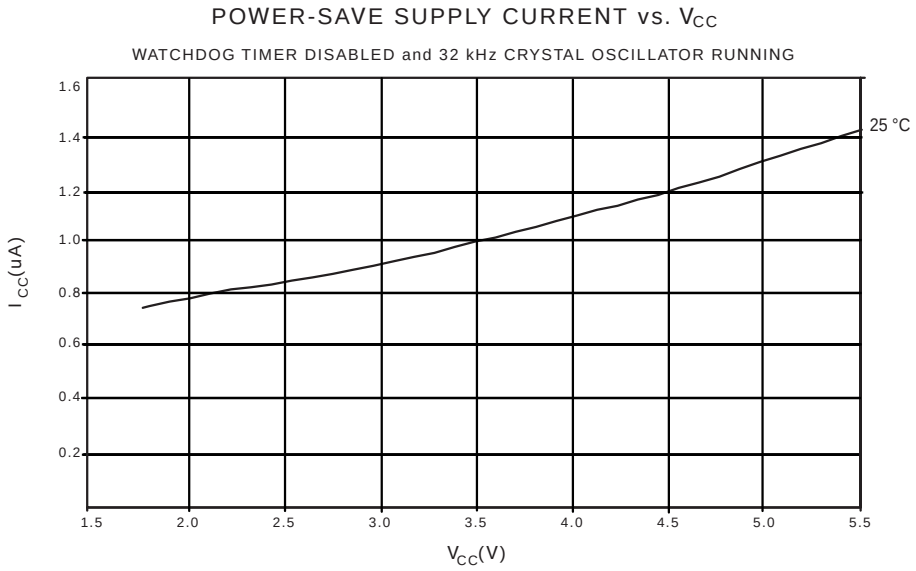


Figure 27-58. Power-Down Supply Current vs. V_{CC} (Watchdog Timer Enabled).



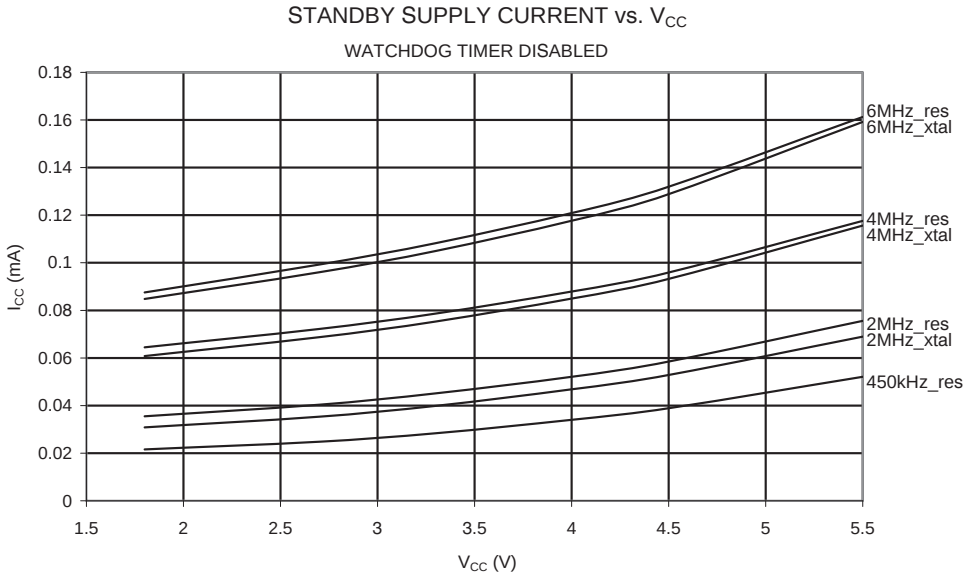
27.2.5 Power-save Supply Current

Figure 27-59. Power-Save Supply Current vs. V_{CC} (Watchdog Timer Disabled and 32 kHz Crystal Oscillator Running).



27.2.6 Standby Supply Current

Figure 27-60. Standby Supply Current vs. Vcc (Watchdog Timer Disabled).



27.2.7 Pin Pull-Up

Figure 27-61. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 1.8 V).

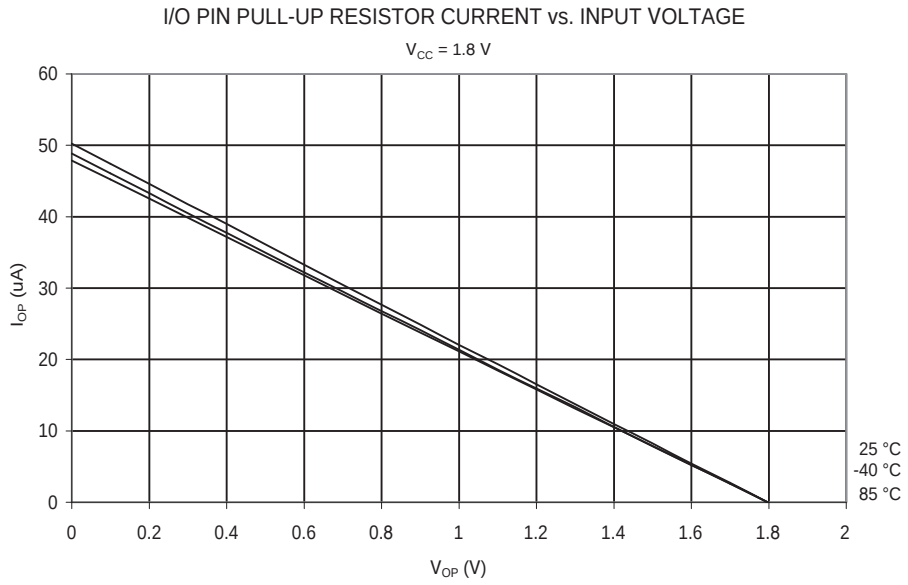


Figure 27-62. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 2.7 V).

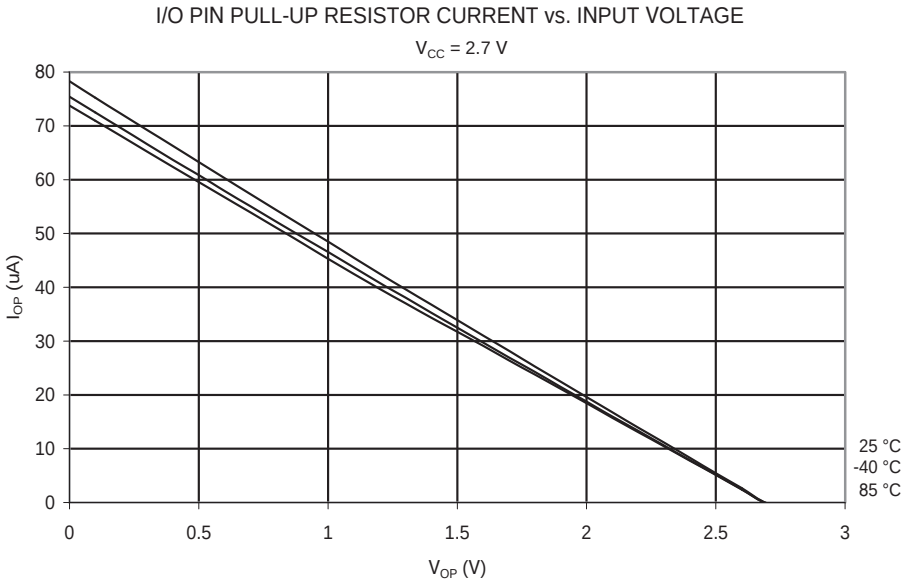


Figure 27-63. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 5 V).

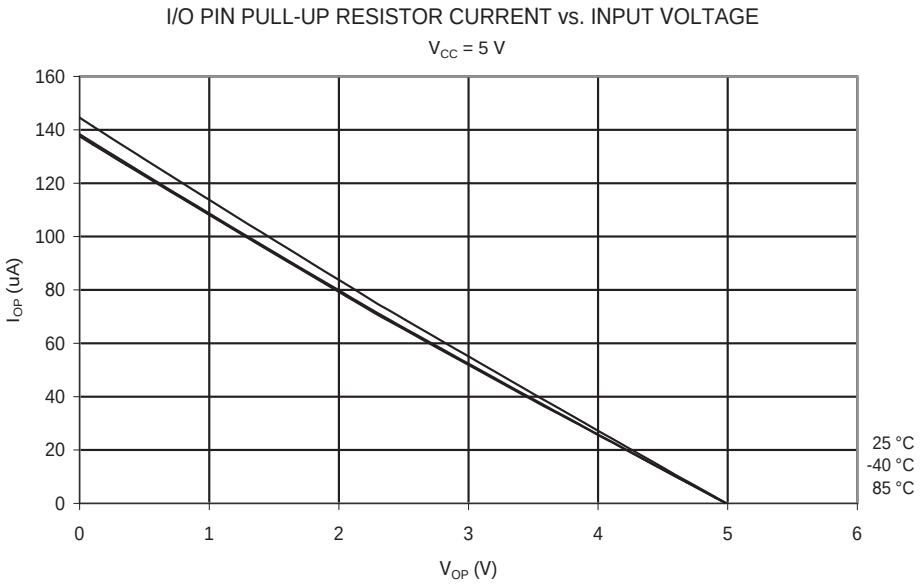


Figure 27-64. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 1.8 V).

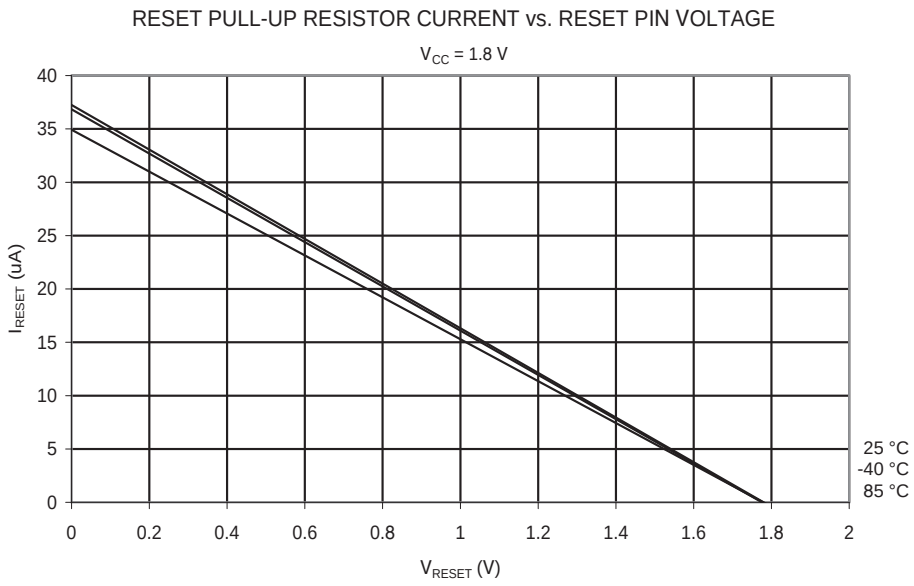


Figure 27-65. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 2.7 V).

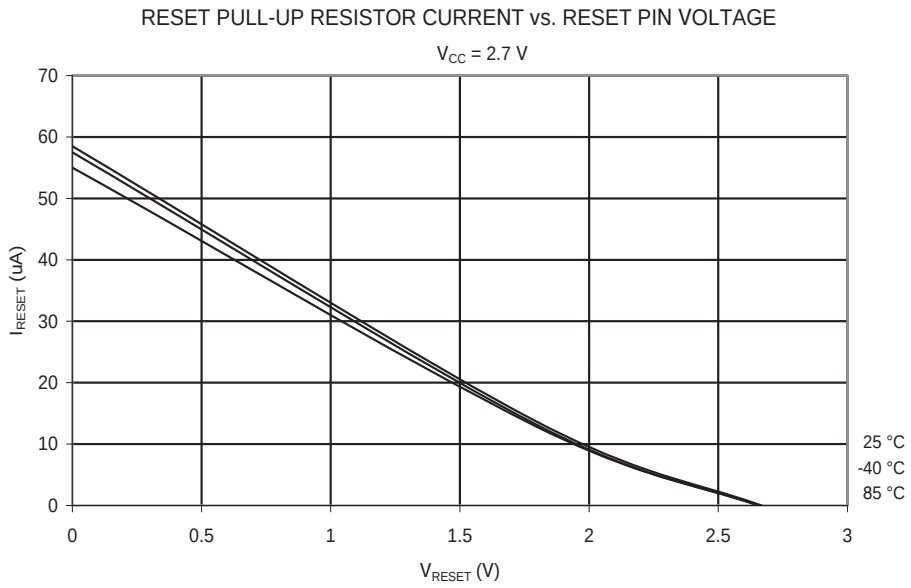
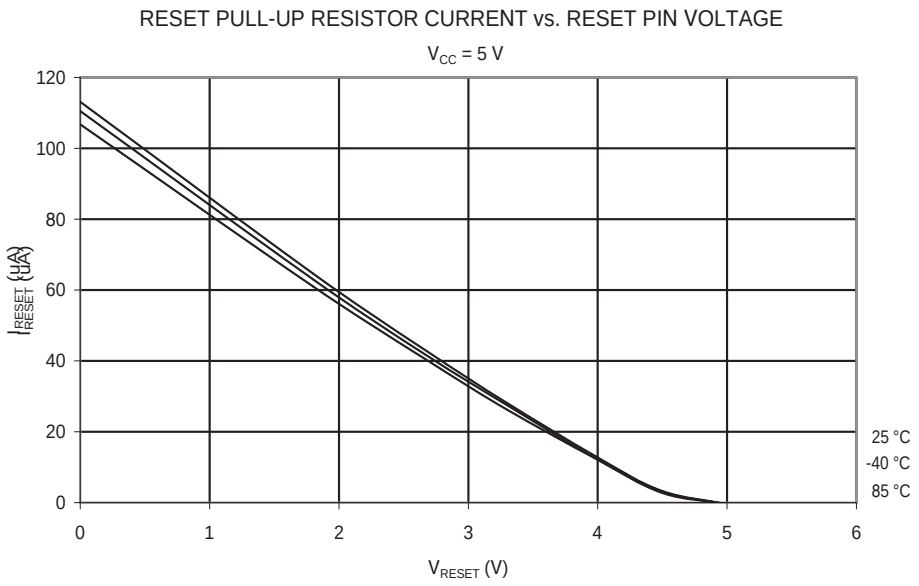


Figure 27-66. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 5 V).



27.2.8 Pin Driver Strength

Figure 27-67. I/O Pin Output Voltage vs. Sink Current(V_{CC} = 3 V).

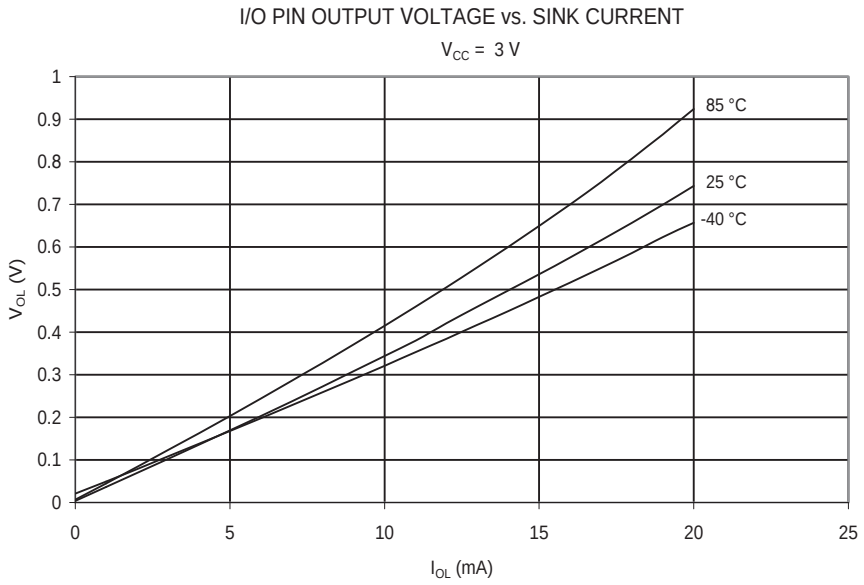


Figure 27-68. I/O Pin Output Voltage vs. Sink Current(V_{CC} = 5 V).

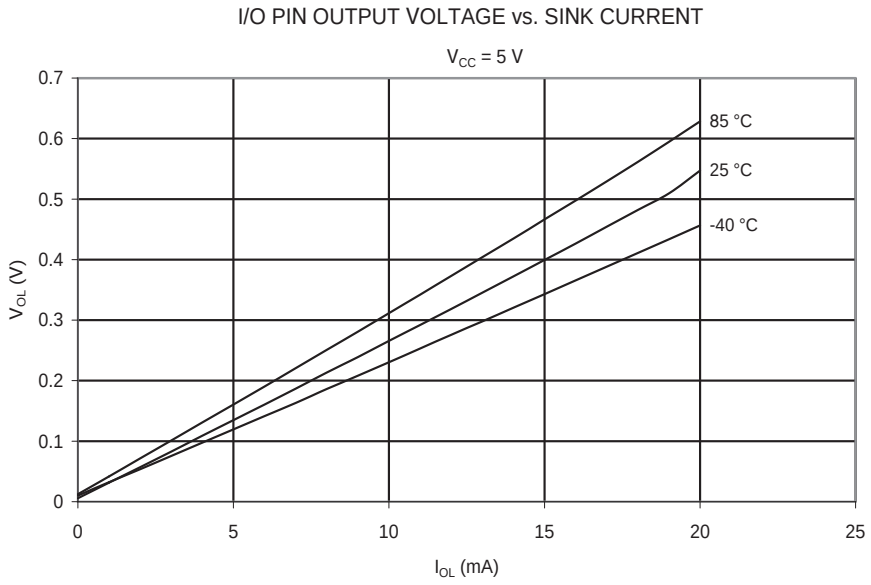


Figure 27-69. I/O Pin Output Voltage vs. Source Current(Vcc = 3 V).

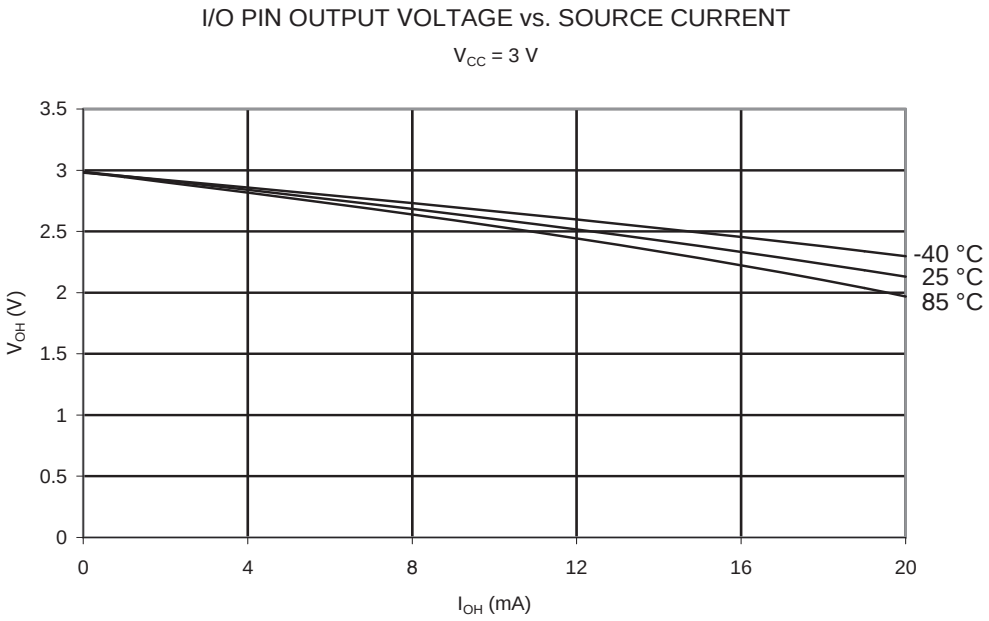
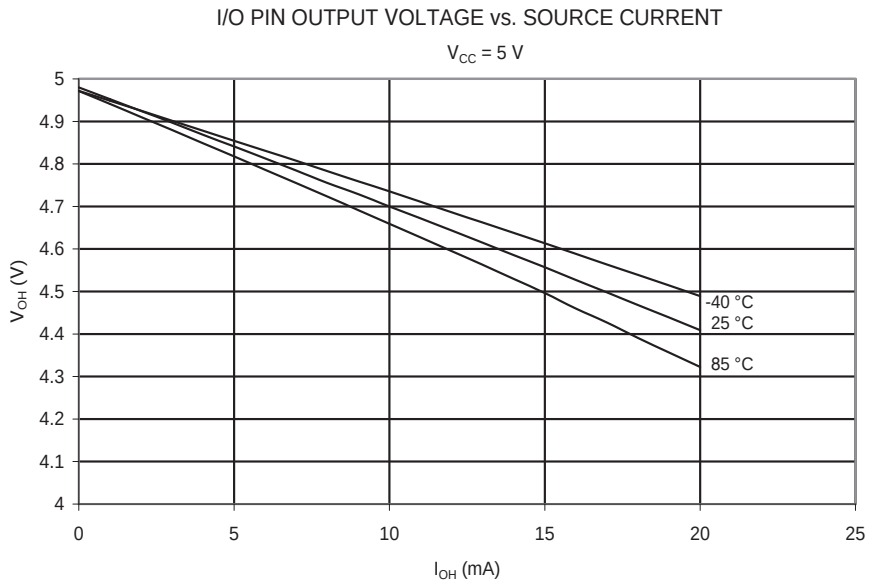


Figure 27-70. I/O Pin Output Voltage vs. Source Current(V_{CC} = 5 V).



27.2.9 Pin Threshold and Hysteresis

Figure 27-71. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin read as '1').

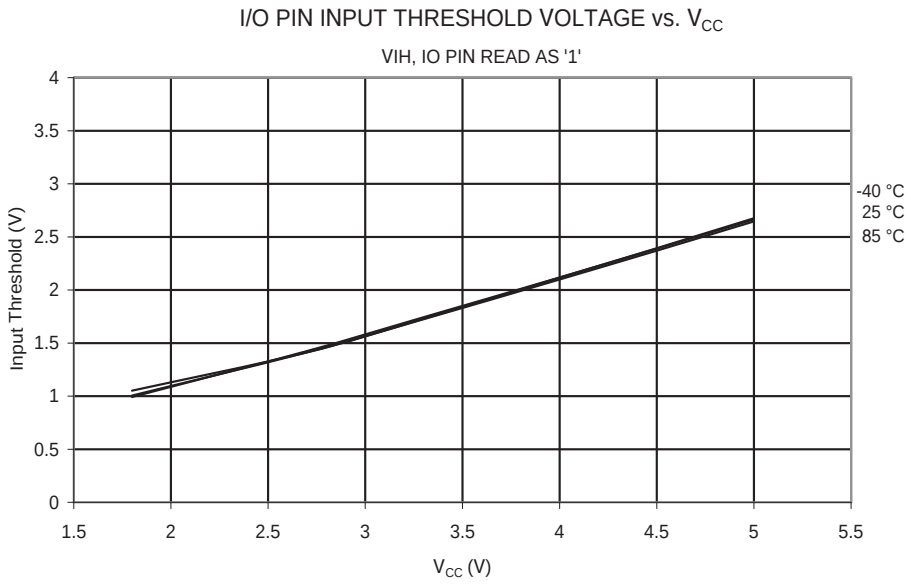


Figure 27-72. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O Pin read as '0').

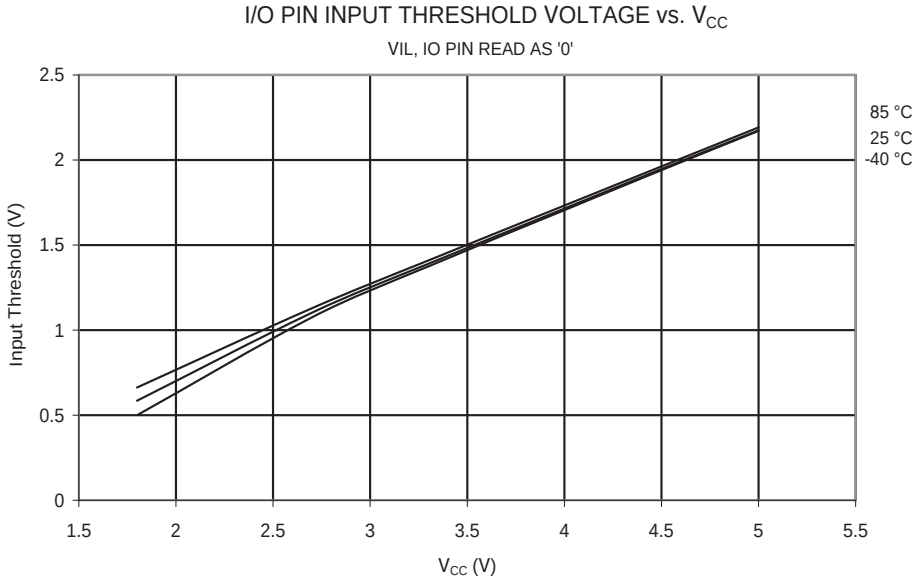


Figure 27-73. I/O Pin Input Hysteresis vs. V_{CC}.

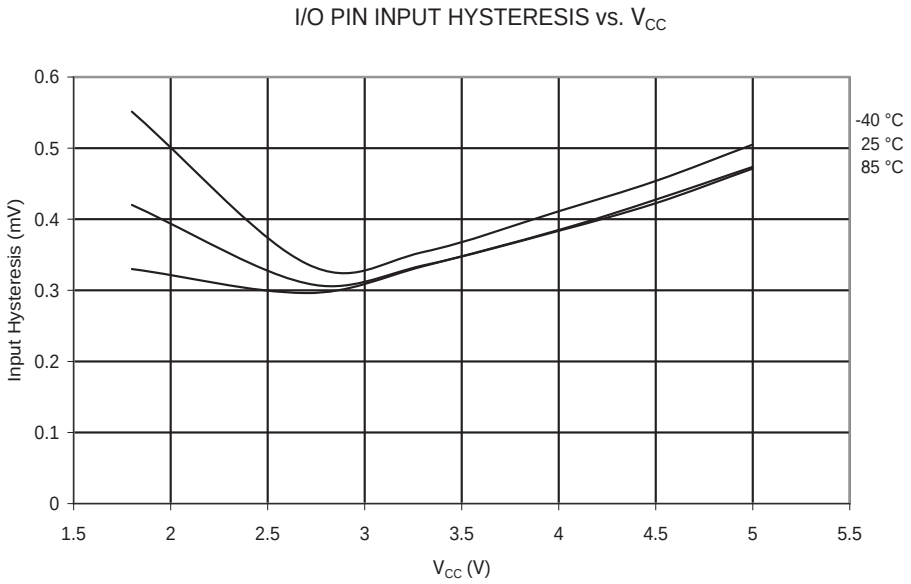


Figure 27-74. Reset Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin read as '1').

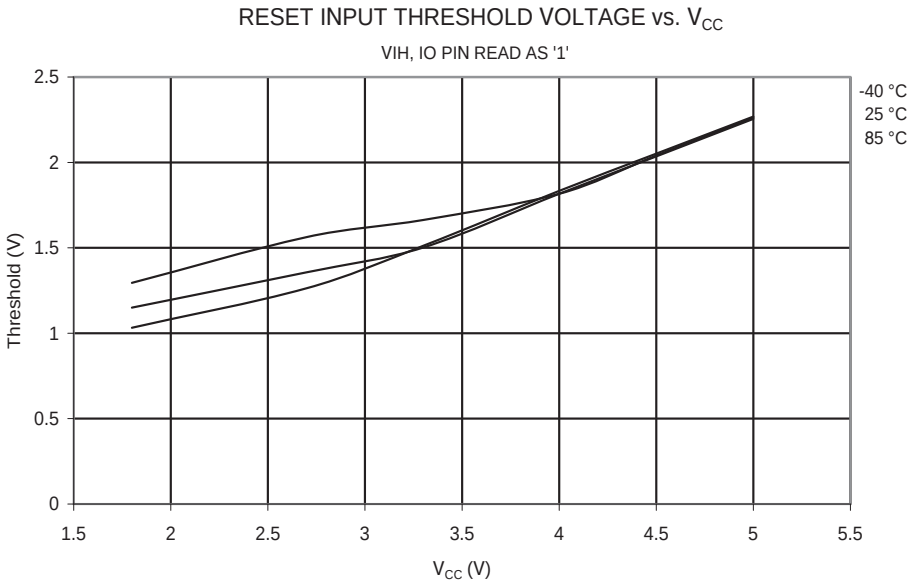


Figure 27-75. Reset Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O Pin read as '0').

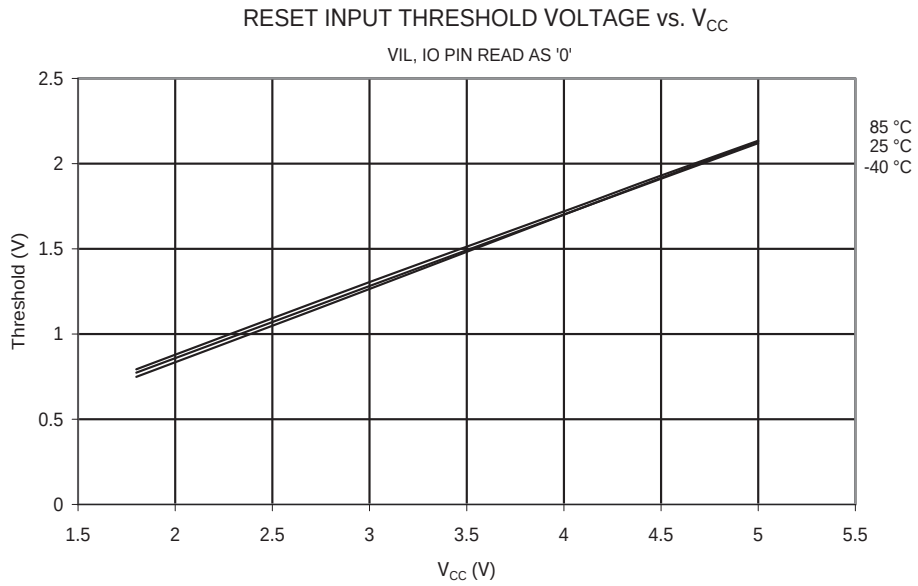
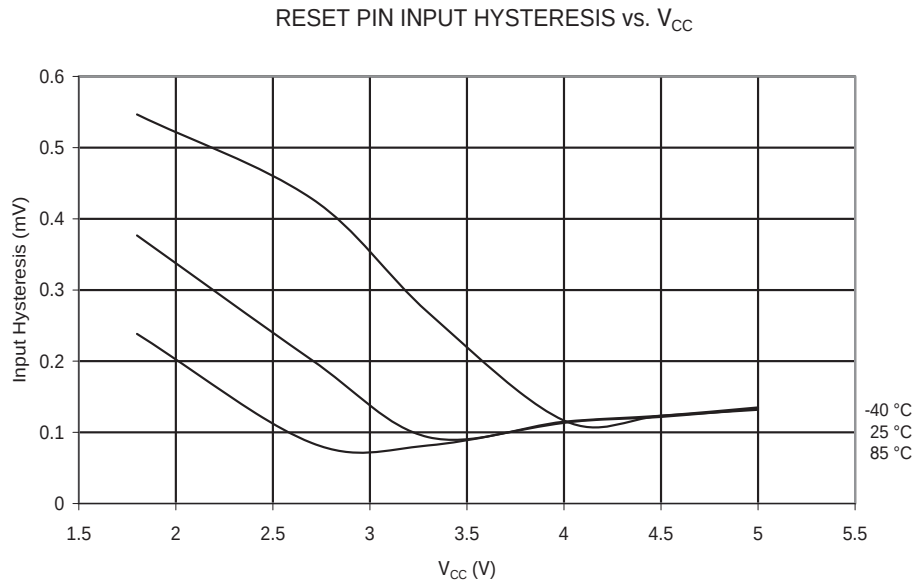


Figure 27-76. Reset Pin Input Hysteresis vs. V_{CC}.



27.2.10 BOD Threshold

Figure 27-77. BOD Thresholds vs. Temperature (BODLEVEL is 1.8 V).

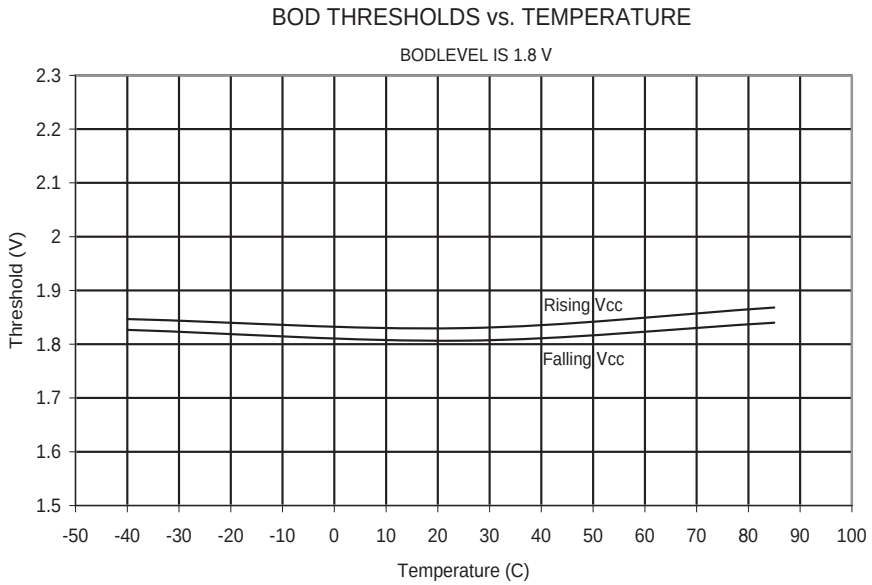


Figure 27-78. BOD Thresholds vs. Temperature (BODLEVEL is 2.7 V).

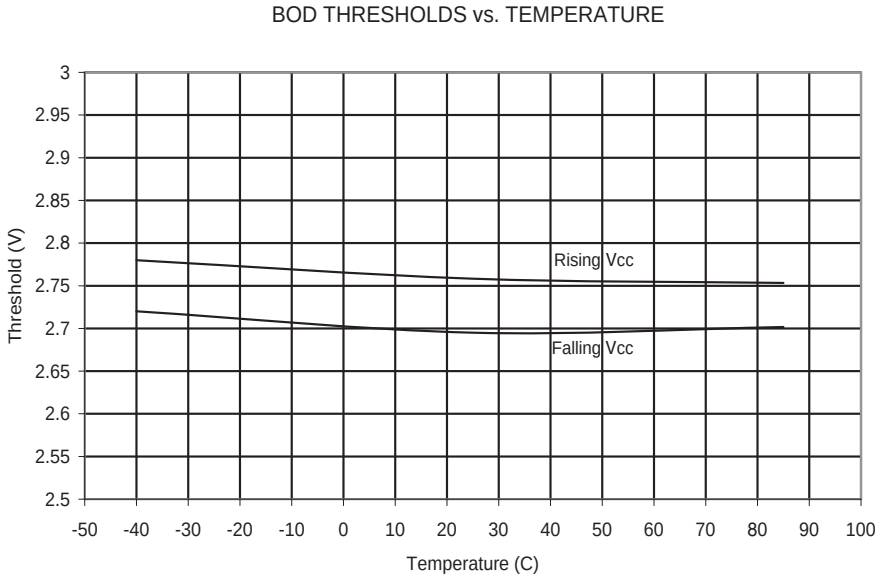
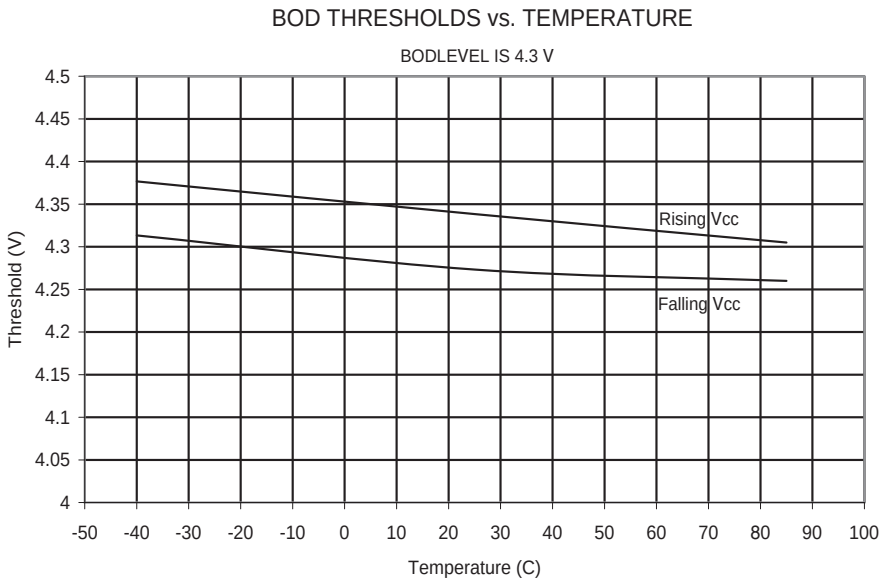


Figure 27-79. BOD Thresholds vs. Temperature (BODLEVEL is 4.3 V).



27.2.11 Internal Oscilllator Speed

Figure 27-80. Watchdog Oscillator Frequency vs. Temperature.

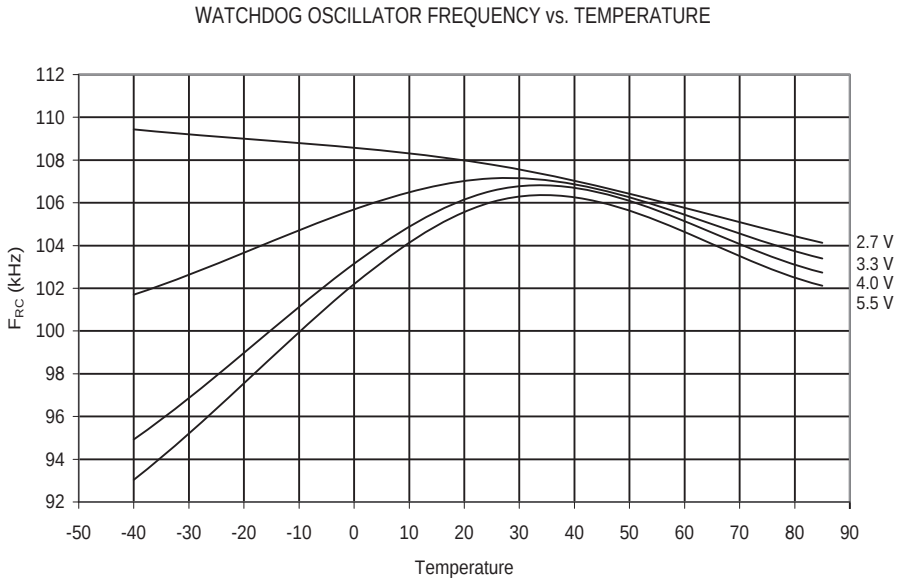


Figure 27-81. Watchdog Oscillator Frequency vs. V_{CC}.

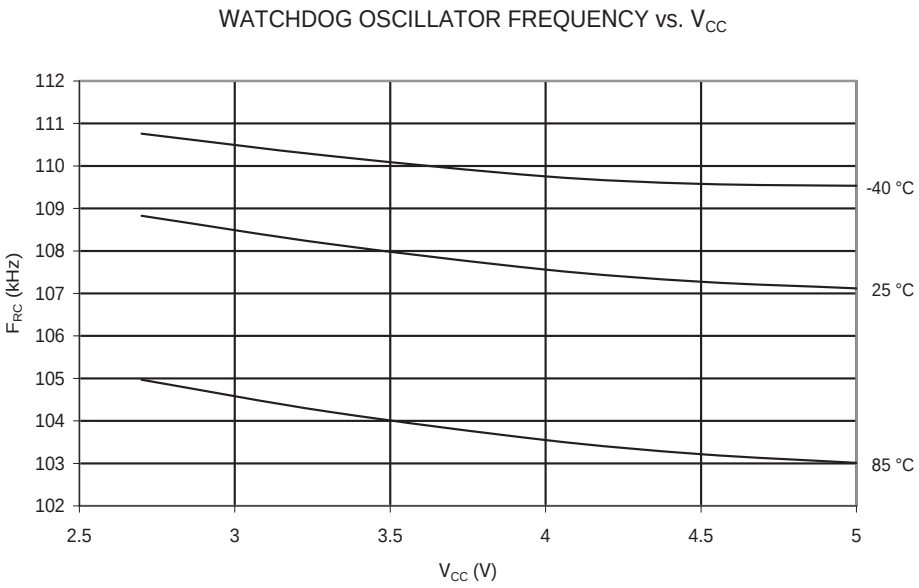


Figure 27-82. Calibrated 8 MHz RC Oscillator Frequency vs. V_{CC}.

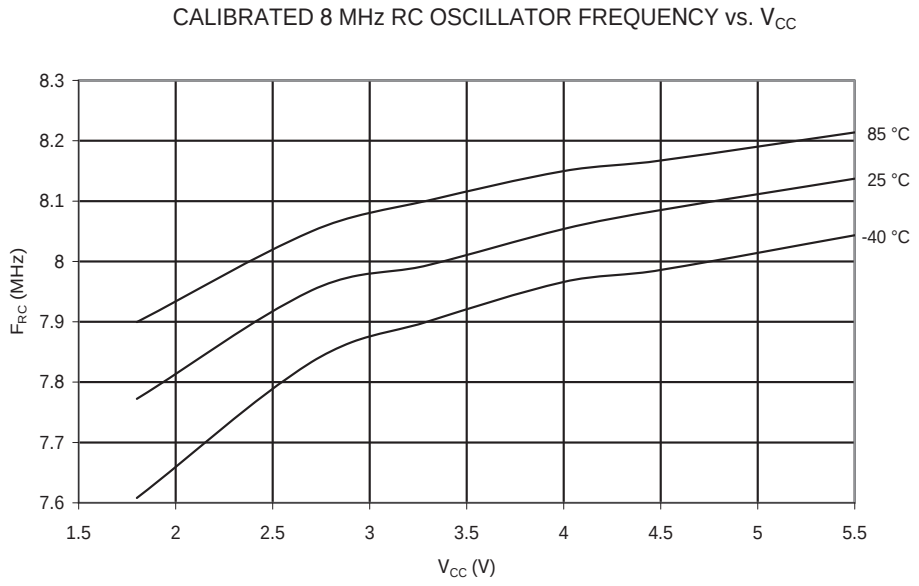


Figure 27-83. Calibrated 8 MHz RC Oscillator Frequency vs. Temperature.

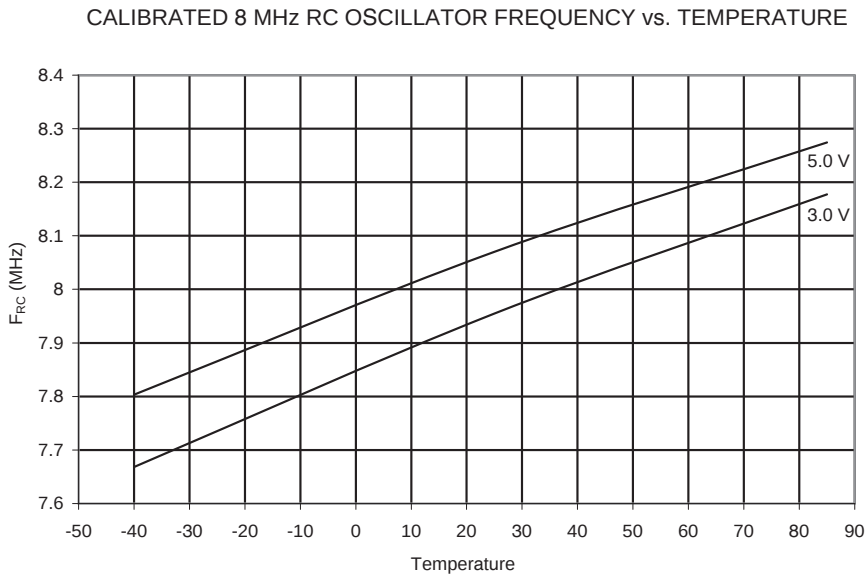
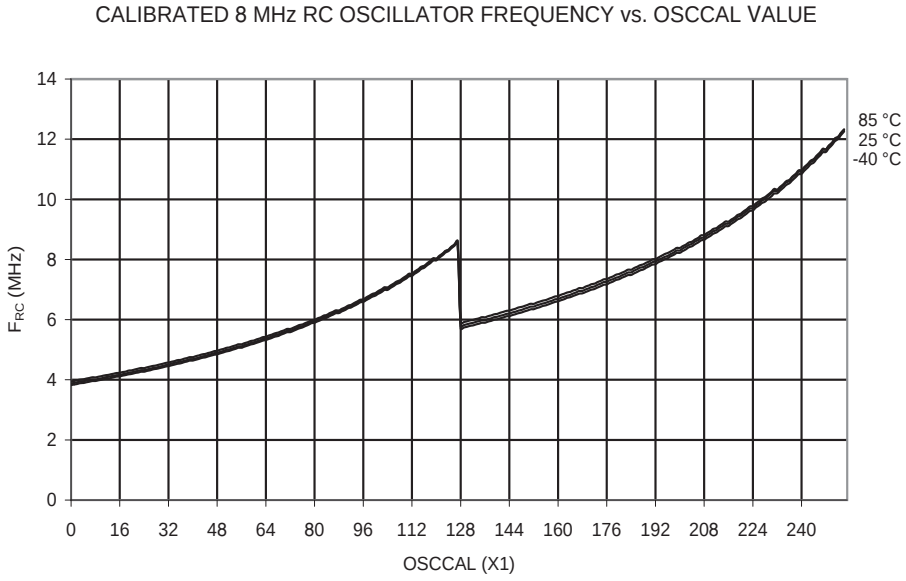


Figure 27-84. Calibrated 8 MHz RC Oscillator Frequency vs. OSCCAL Value.



27.2.12 Current Consumption of Peripheral Units

Figure 27-85. ADC Current vs. V_{CC} (AREF = AV_{CC}).

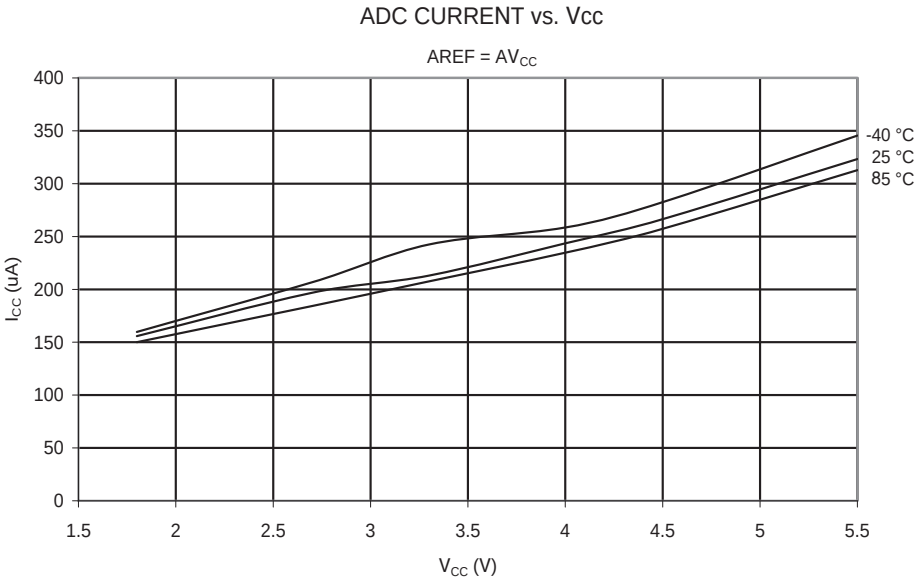


Figure 27-86. Analog Comparator Current vs. V_{CC}.

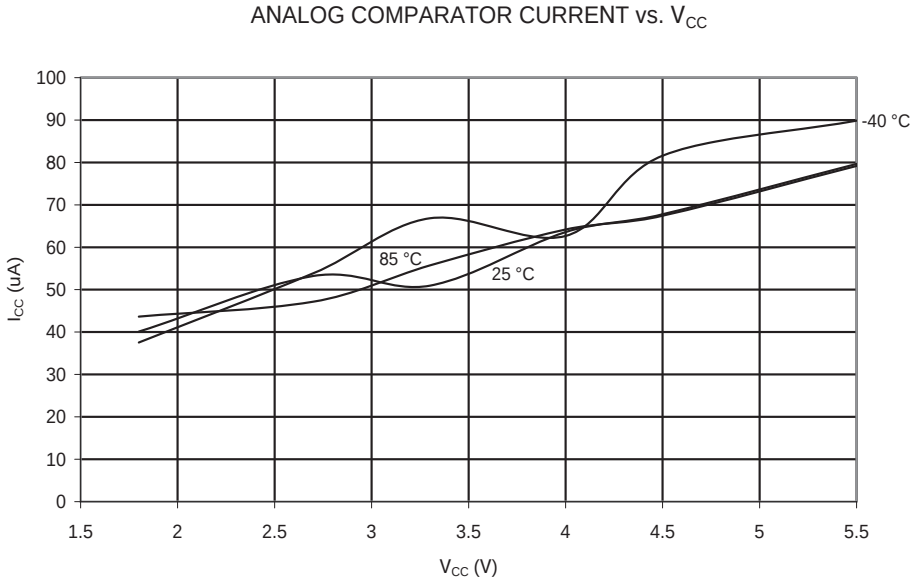


Figure 27-87. AREF External Reference Current vs. V_{CC}.

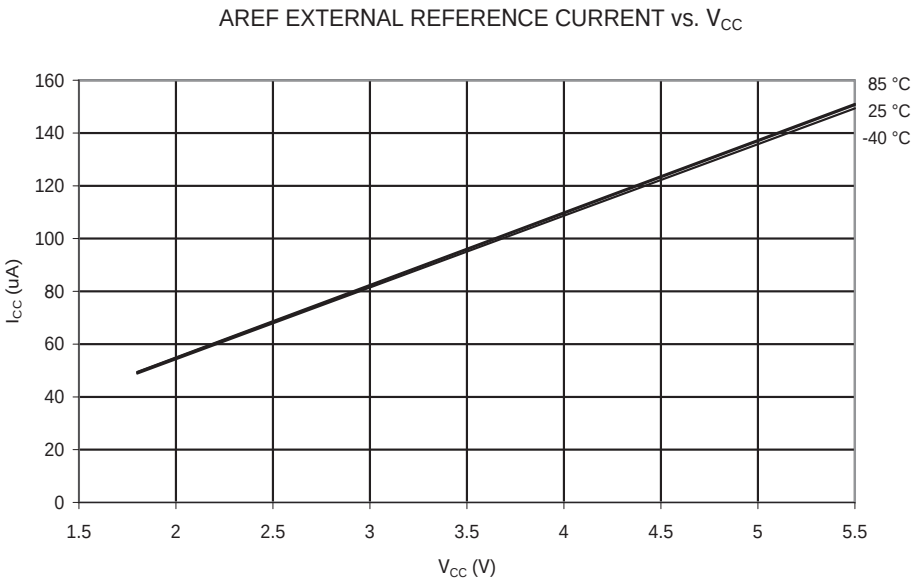


Figure 27-88. Brownout Detector Current vs. V_{CC}.

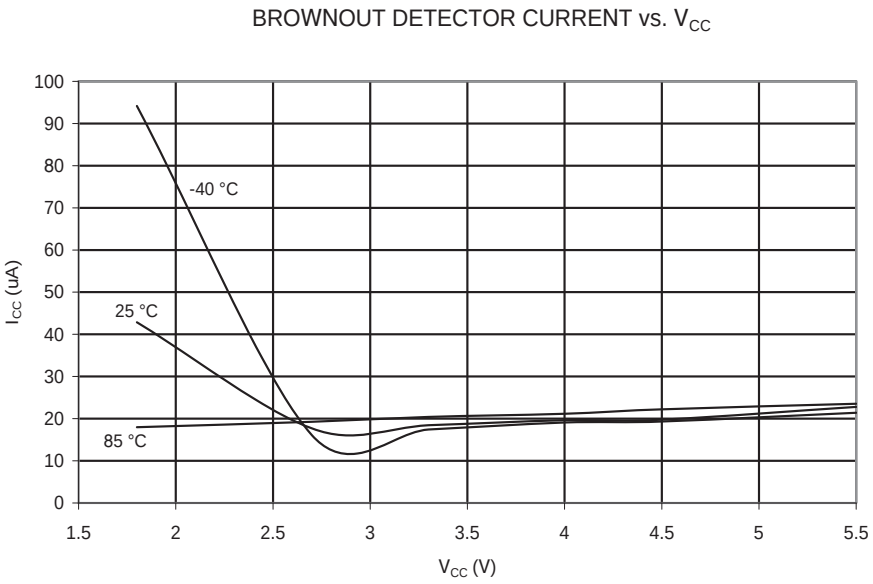
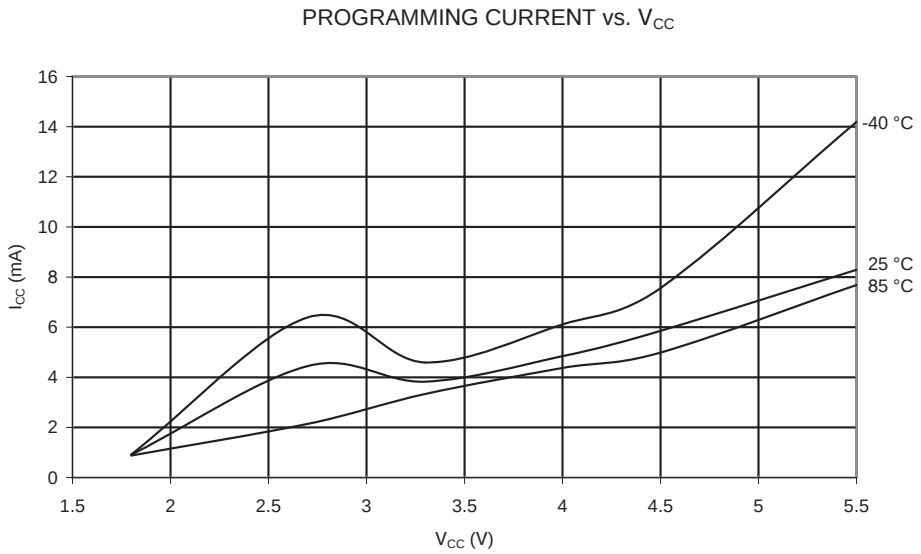


Figure 27-89. Programming Current vs. V_{CC}.



27.2.13 Current Consumption in Reset and Reset Pulsewidth

Figure 27-90. Reset Supply Current vs. Low Frequency (0.1 - 1.0 MHz).

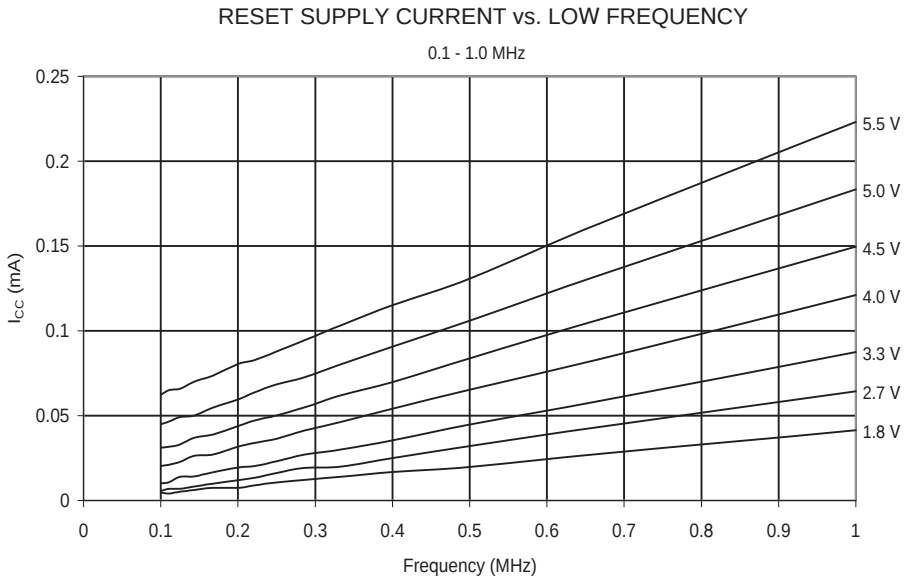


Figure 27-91. Reset Supply Current vs. Frequency (1 - 20 MHz).

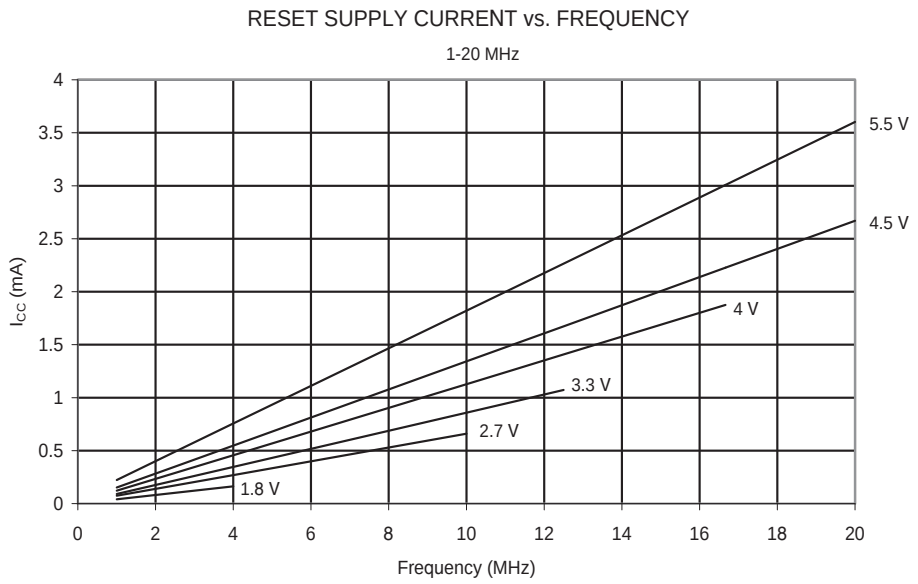
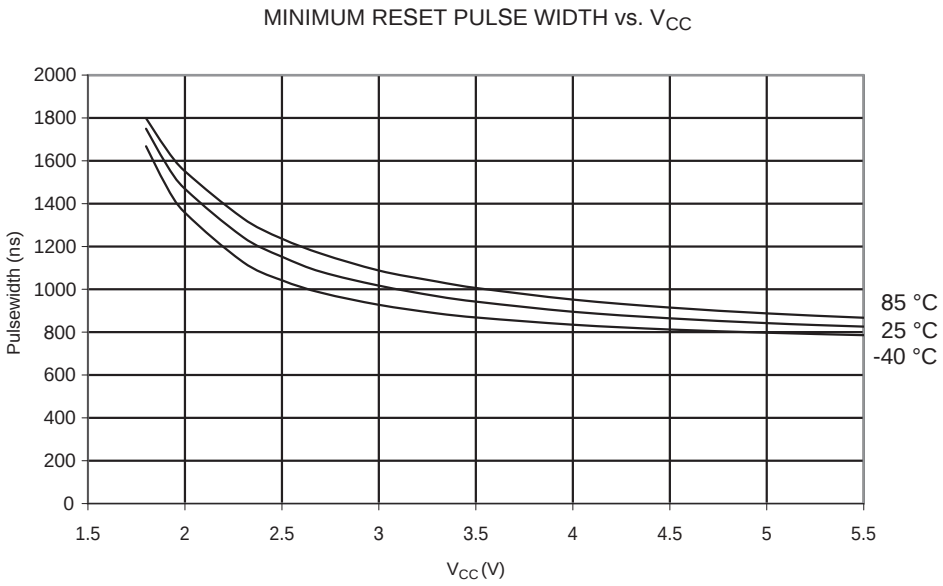


Figure 27-92. Minimum Reset Pulse width vs. V_{CC}.



27.3 ATmega168P Typical Characteristics

27.3.1 Active Supply Current

Figure 27-93. Active Supply Current vs. Low Frequency (0.1-1.0 MHz).

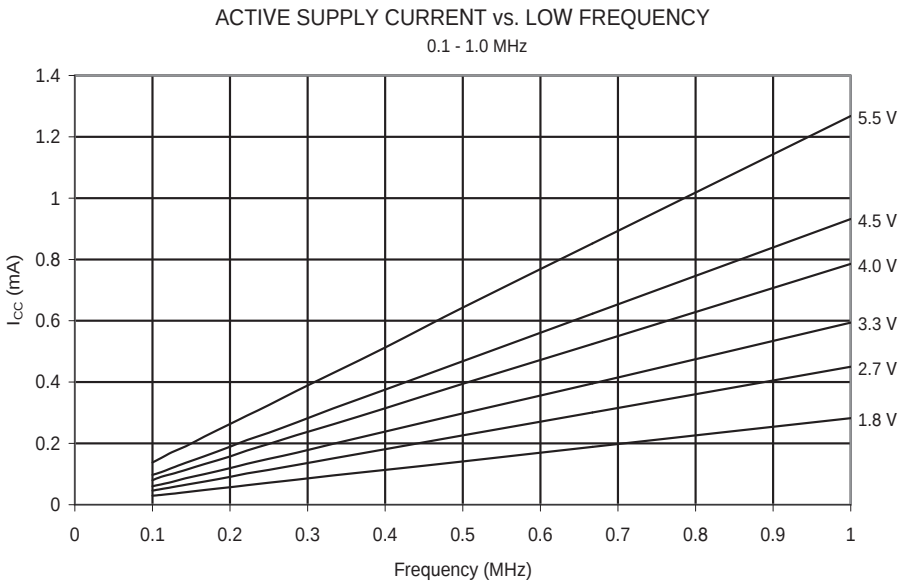


Figure 27-94. Active Supply Current vs. Frequency (1-20 MHz).

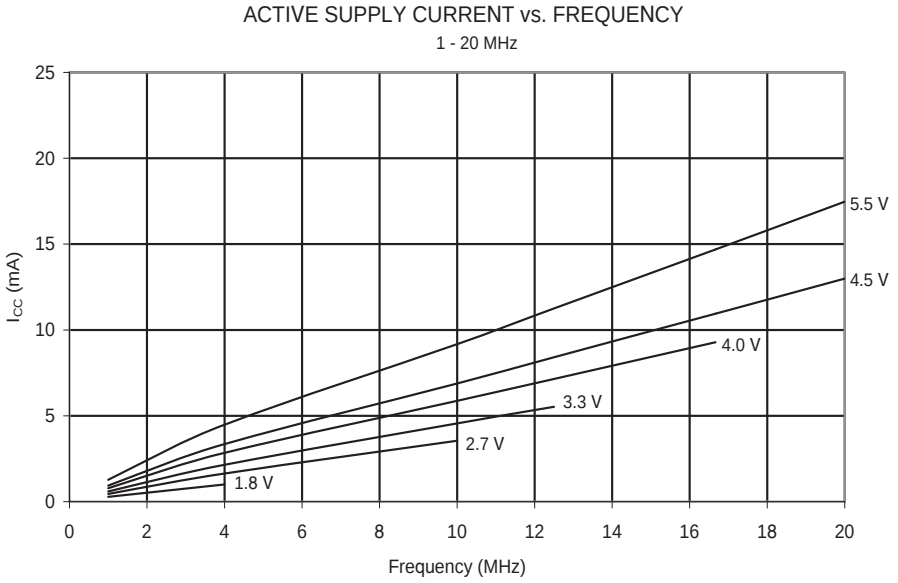


Figure 27-95. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz).

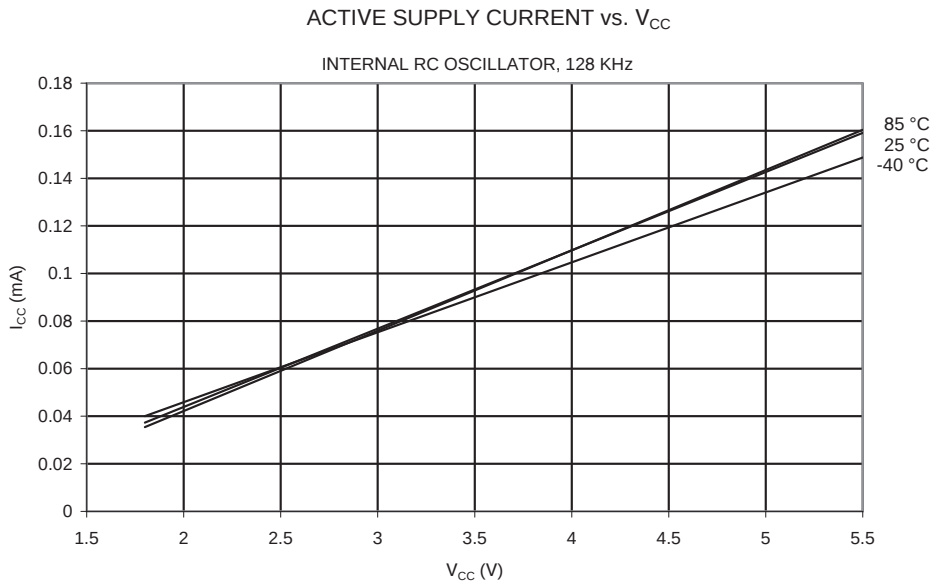


Figure 27-96. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz).

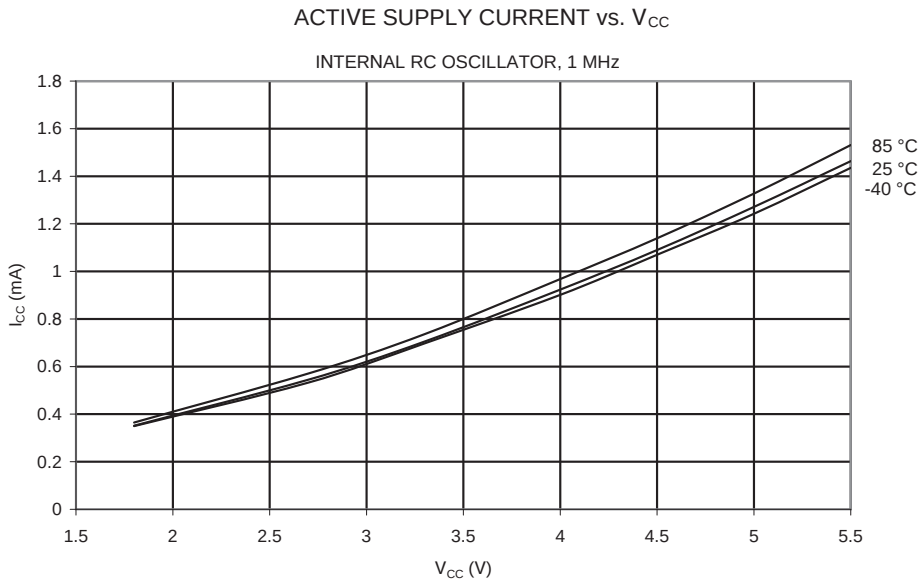
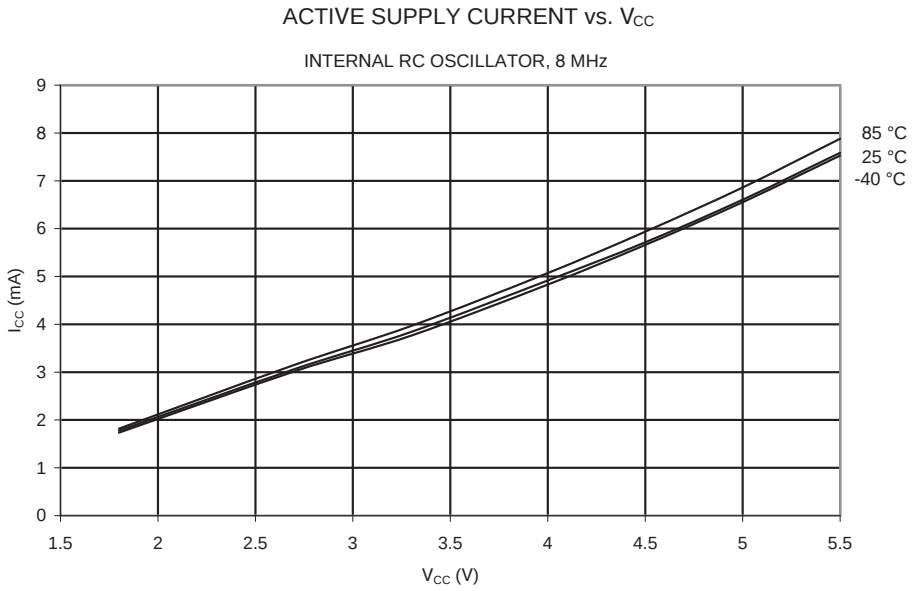


Figure 27-97. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 8 MHz).



27.3.2 Idle Supply Current

Figure 27-98. Idle Supply Current vs. Low Frequency (0.1-1.0 MHz).

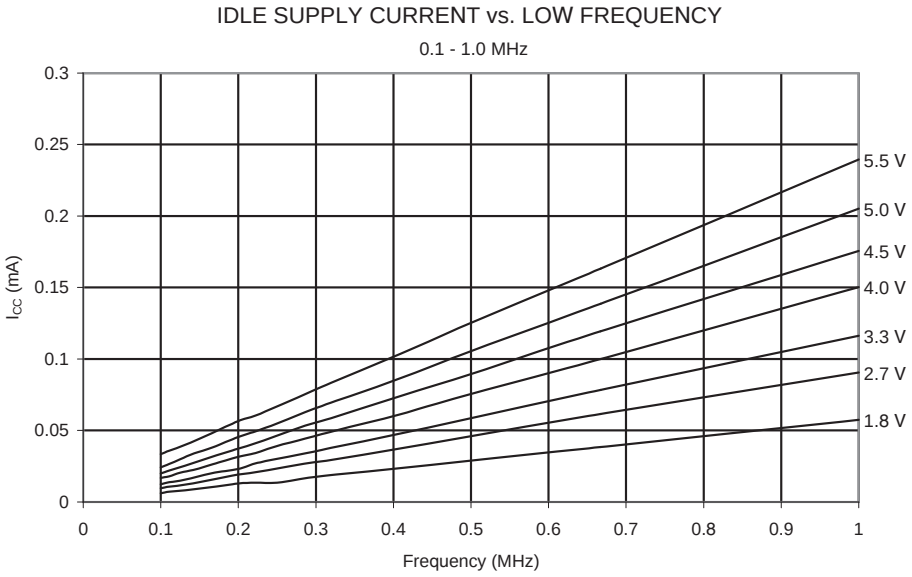


Figure 27-99. Idle Supply Current vs. Frequency (1-20 MHz).

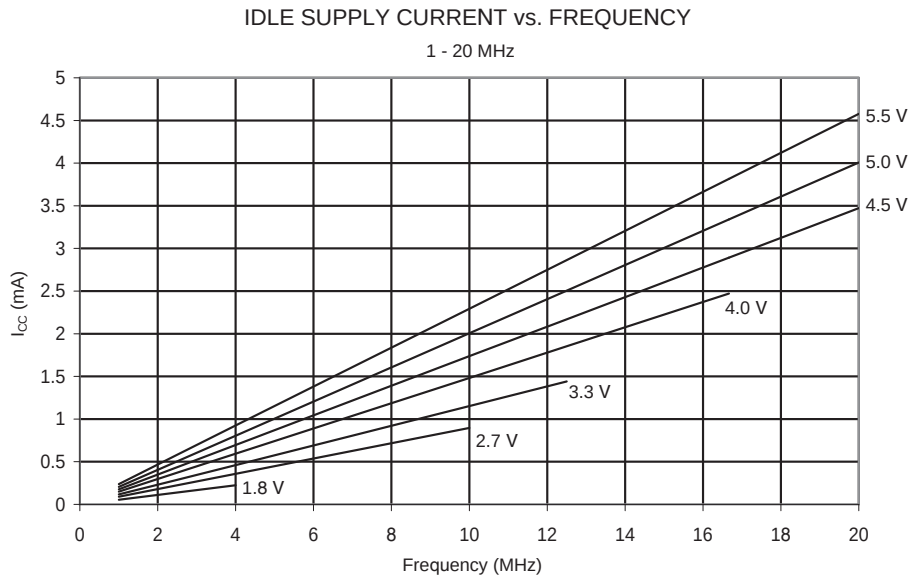


Figure 27-100. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz).

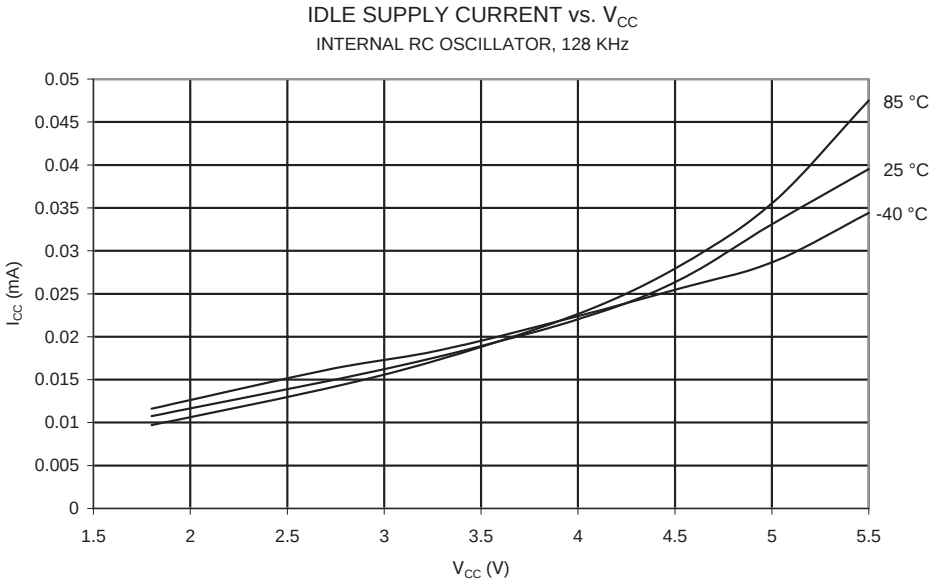


Figure 27-101. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz).

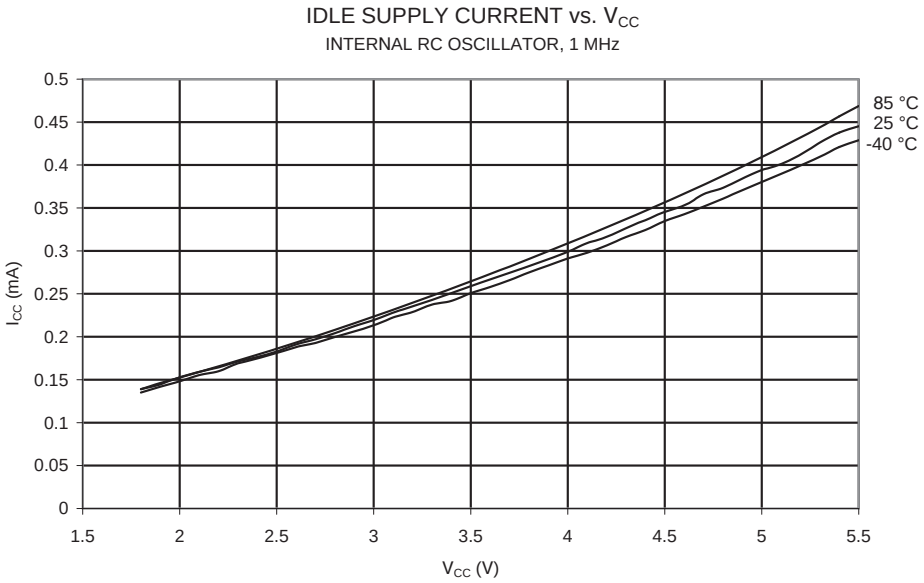
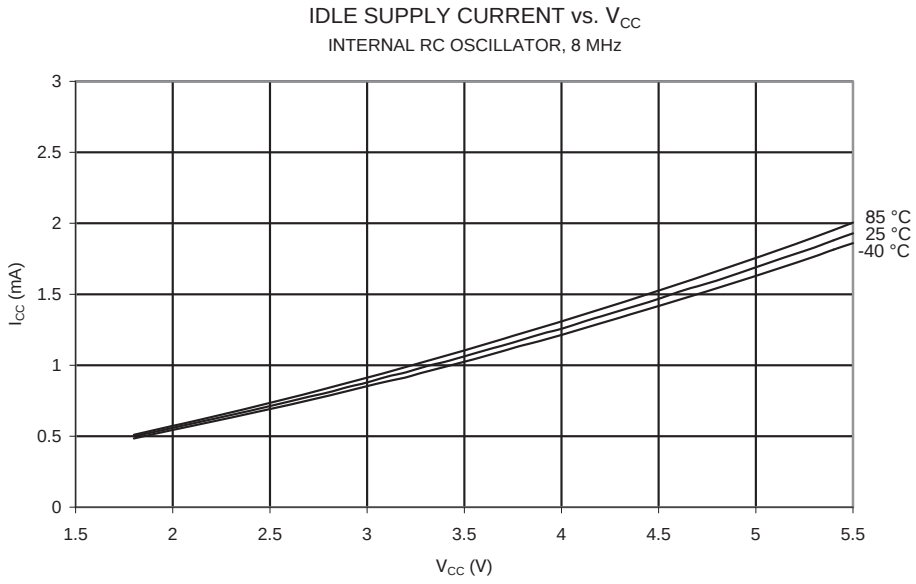


Figure 27-102. Idle Supply Current vs. Vcc (Internal RC Oscillator, 8 MHz).



27.3.3 Supply Current of IO Modules

The tables and formulas below can be used to calculate the additional current consumption for the different I/O modules in Active and Idle mode. The enabling or disabling of the I/O modules are controlled by the Power Reduction Register. See ”Power Reduction Register” on page 42 for details.

Table 27-5. Additional Current Consumption for the different I/O modules (absolute values)

PRR bit	Typical numbers		
	V_{CC} = 2V, F = 1MHz	V_{CC} = 3V, F = 4MHz	V_{CC} = 5V, F = 8MHz
PRUSART0	5.54 uA	34.8 uA	133.9 uA
PRTWI	10.1 uA	63.1 uA	250.6 uA
PRTIM2	10.4 uA	68.5 uA	269.4 uA
PRTIM1	9.66 uA	62.9 uA	248.1 uA
PRTIM0	2.45 uA	14.1 uA	60.8 uA
PRSPI	9.55 uA	63.8 uA	275.6 uA
PRADC	9.04 uA	58.6 uA	243.9 uA

Table 27-6. Additional Current Consumption (percentage) in Active and Idle mode

PRR bit	Additional Current consumption compared to Active with external clock (see Figure 27-93 on page 377 and Figure 27-94 on page 377)	Additional Current consumption compared to Idle with external clock (see Figure 27-98 on page 379 and Figure 27-99 on page 380)
PRUSART0	1.9%	8.5%
PRTWI	3.4%	15.6%
PRTIM2	3.6%	16.5%
PRTIM1	3.4%	15.2%
PRTIM0	0.8%	3.7%
PRSPI	3.5%	15.8%
PRADC	3.2%	14.5%

It is possible to calculate the typical current consumption based on the numbers from Table 27-6 on page 382 for other V_{CC} and frequency settings than listed in Table 27-5 on page 382.

Example

Calculate the expected current consumption in idle mode with TIMER1, ADC, and SPI enabled at V_{CC} = 2.0V and F = 1MHz. From Table 27-6 on page 382, third column, we see that we need to add 15.2% for the TIMER1, 14.5% for the ADC, and 15.8% for the SPI module. Reading from Figure 27-98 on page 379, we find that the idle current consumption is ~0.055 mA at V_{CC} = 2.0V and F = 1MHz. The total current consumption in idle mode with TIMER1, ADC, and SPI enabled, gives:

$$I_{CCtotal} \approx 0.065\ mA \cdot (1 + 0.152 + 0.145 + 0.158) \approx 0.095\ mA$$

27.3.4 Power-down Supply Current

Figure 27-103. Power-Down Supply Current vs. V_{CC} (Watchdog Timer Disabled).

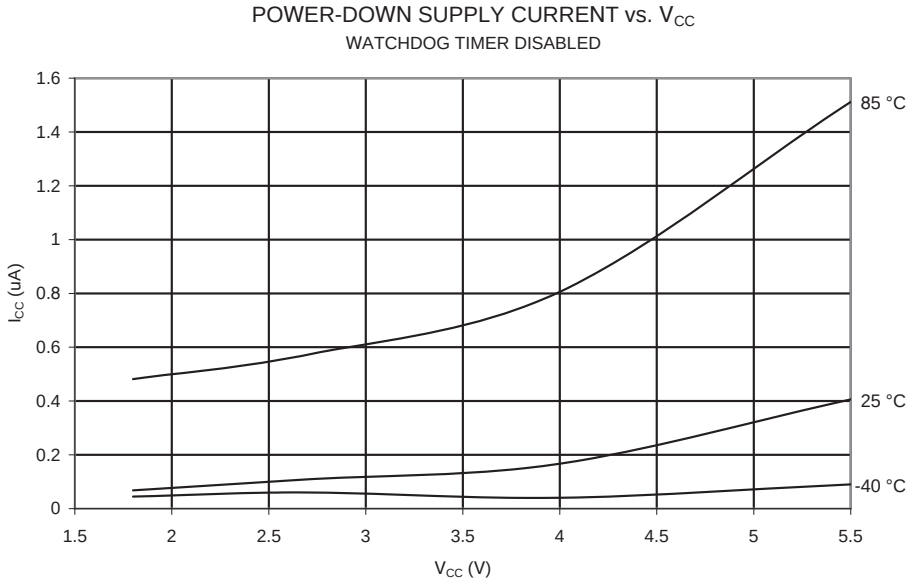
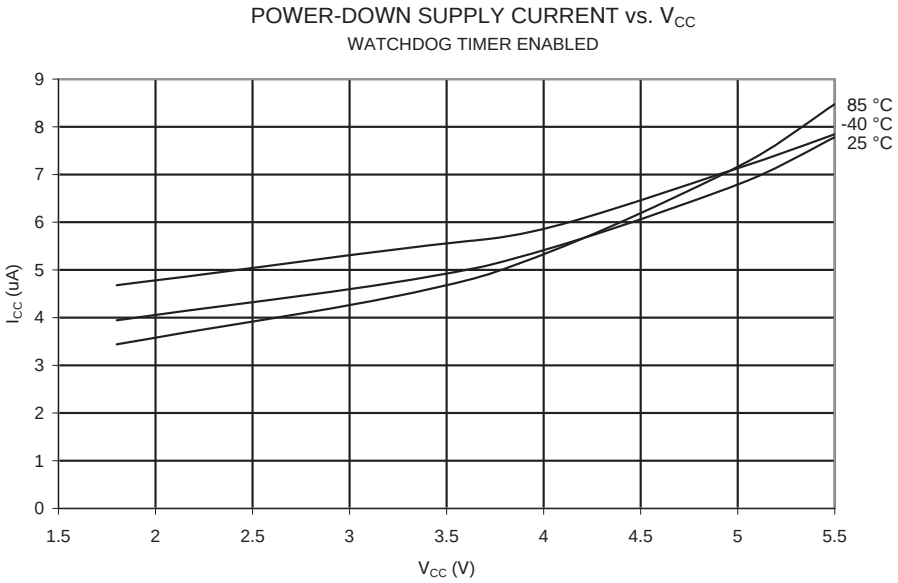
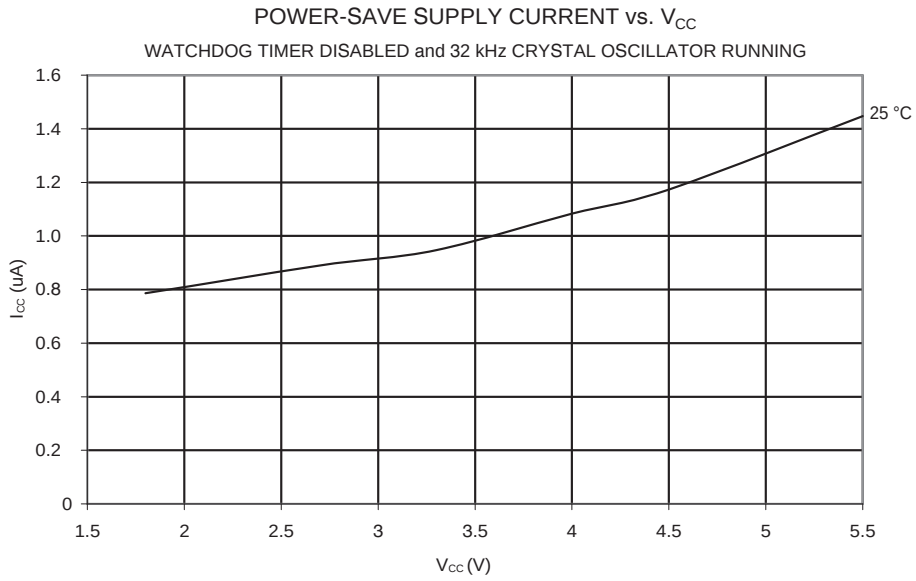


Figure 27-104. Power-Down Supply Current vs. V_{CC} (Watchdog Timer Enabled).



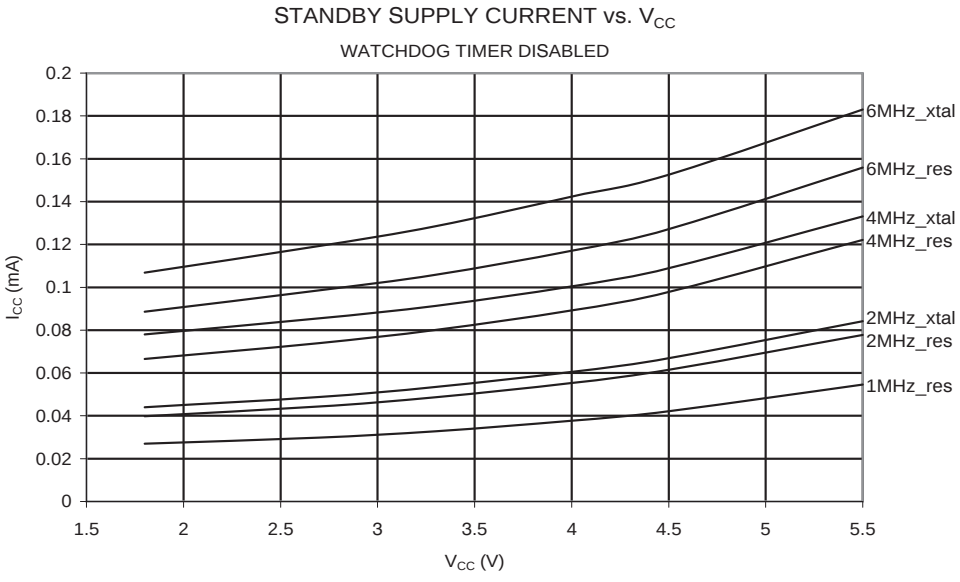
27.3.5 Power-save Supply Current

Figure 27-105. Power-Save Supply Current vs. V_{CC} (Watchdog Timer Disabled and 32 kHz Crystal Oscillator Running).



27.3.6 Standby Supply Current

Figure 27-106. Standby Supply Current vs. Vcc (Watchdog Timer Disabled).



27.3.7 Pin Pull-Up

Figure 27-107. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 1.8 V).

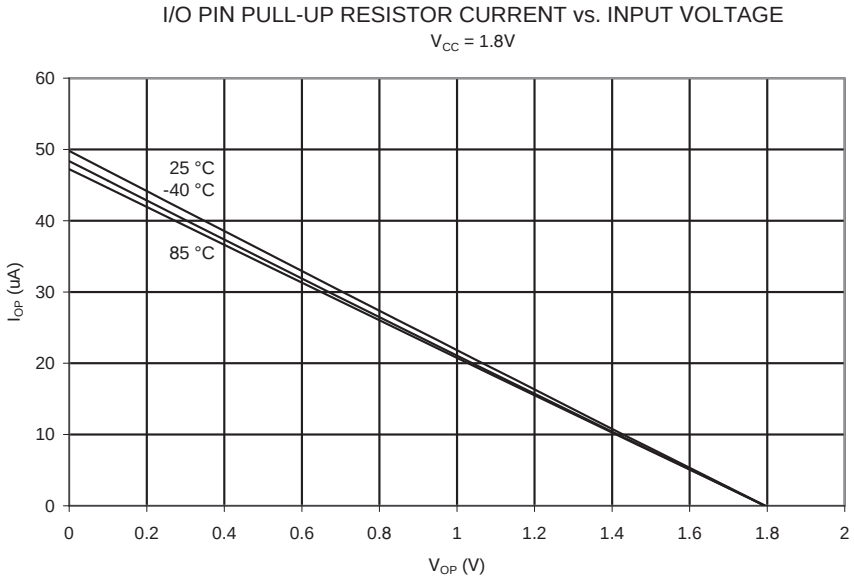


Figure 27-108. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 2.7 V).

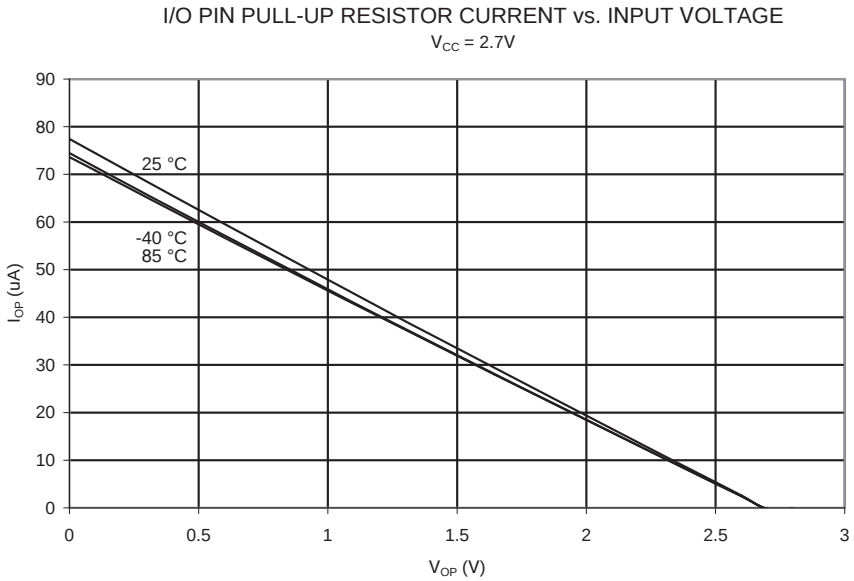


Figure 27-109. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 5 V).

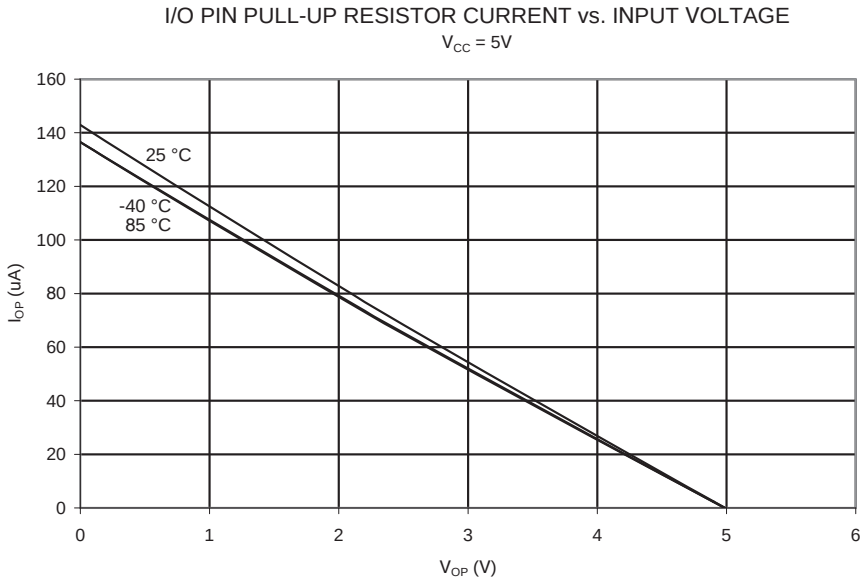


Figure 27-110. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 1.8 V).

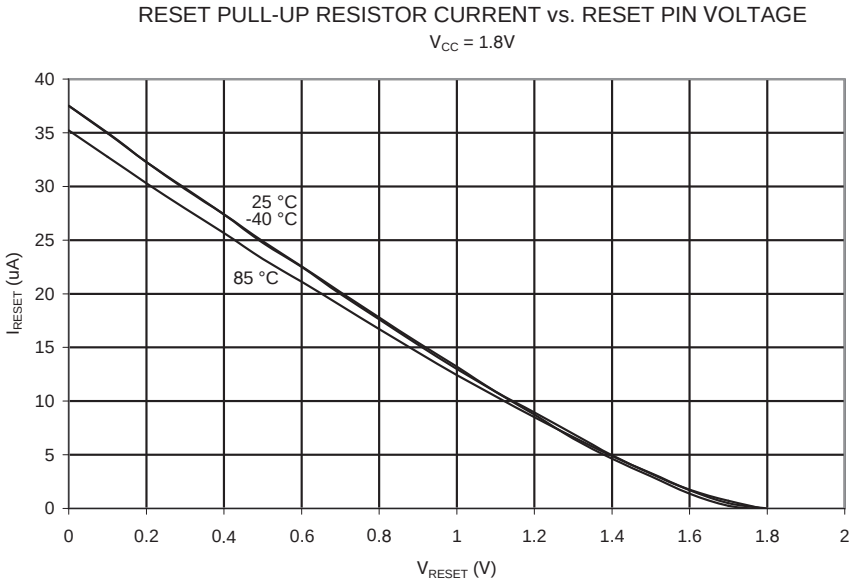


Figure 27-111. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 2.7 V).

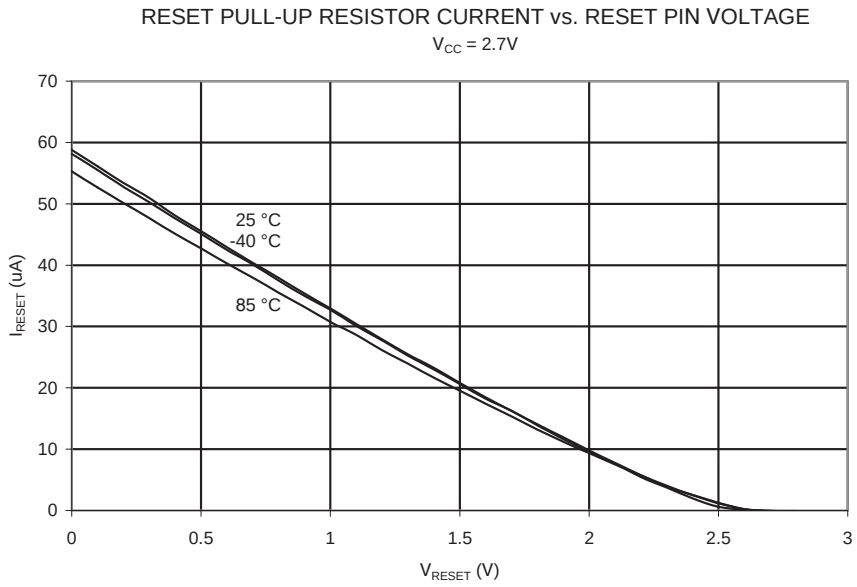
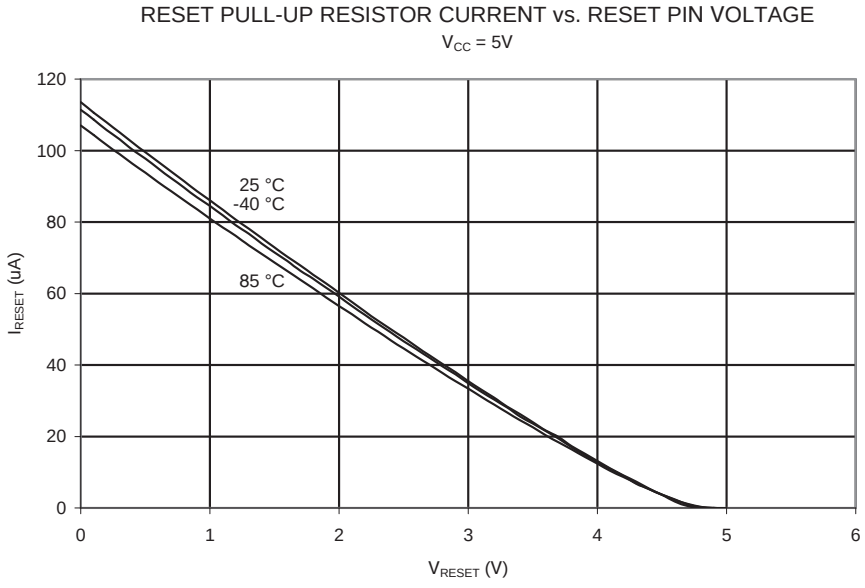


Figure 27-112. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 5 V).



27.3.8 Pin Driver Strength

Figure 27-113. I/O Pin Output Voltage vs. Sink Current(V_{CC} = 3 V).

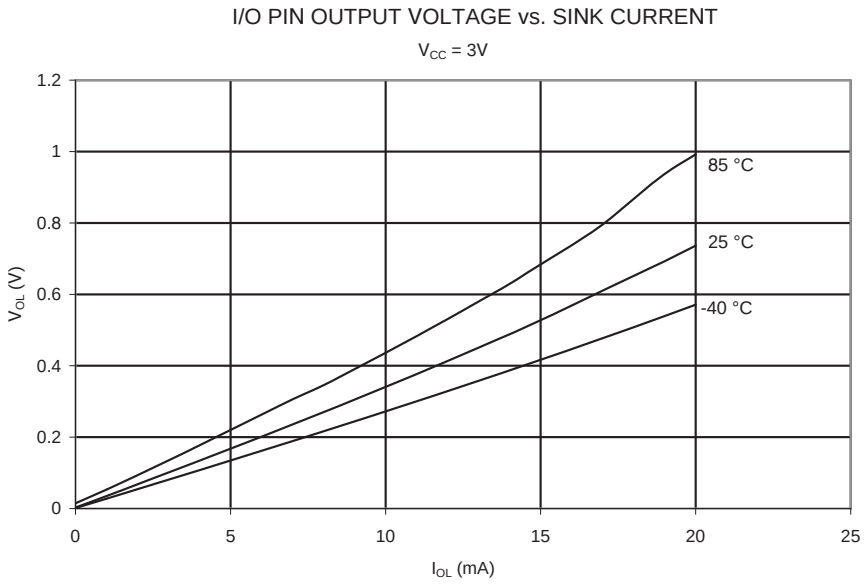


Figure 27-114. I/O Pin Output Voltage vs. Sink Current(V_{CC} = 5 V).

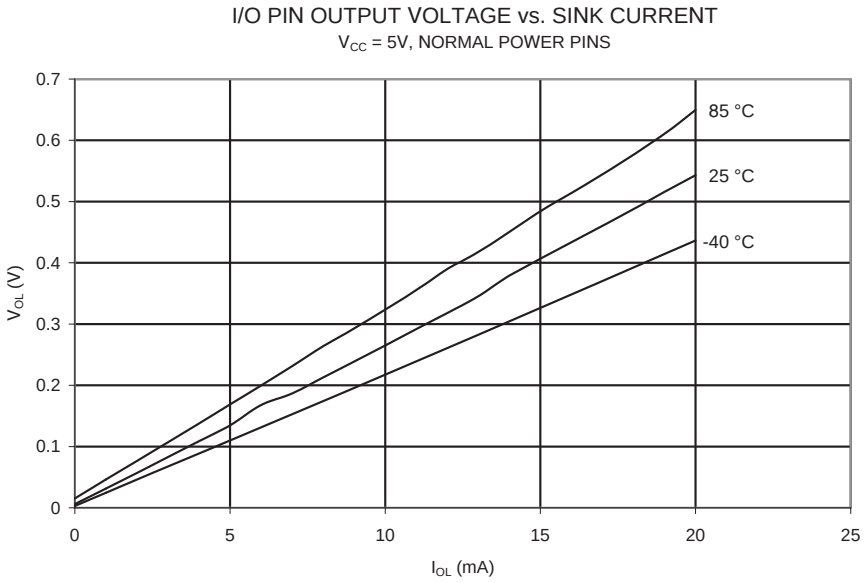


Figure 27-115. I/O Pin Output Voltage vs. Source Current(Vcc = 3 V).

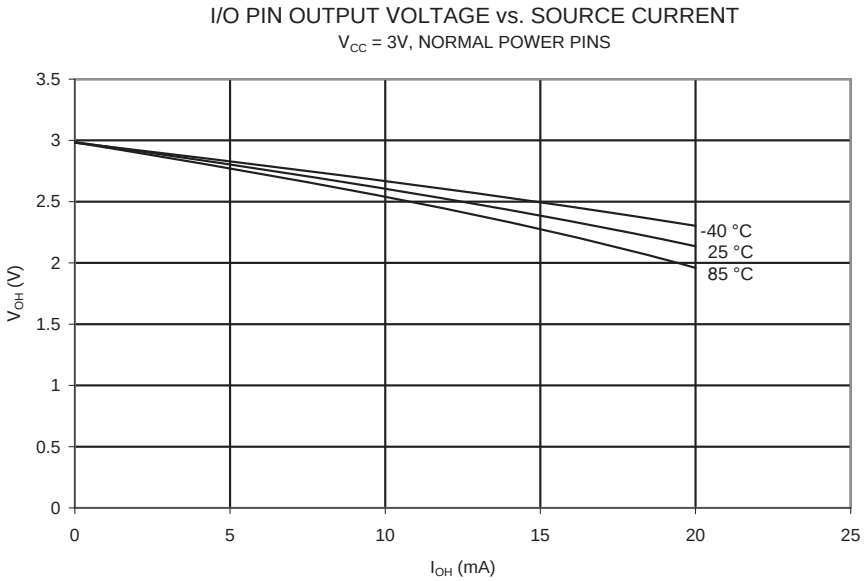
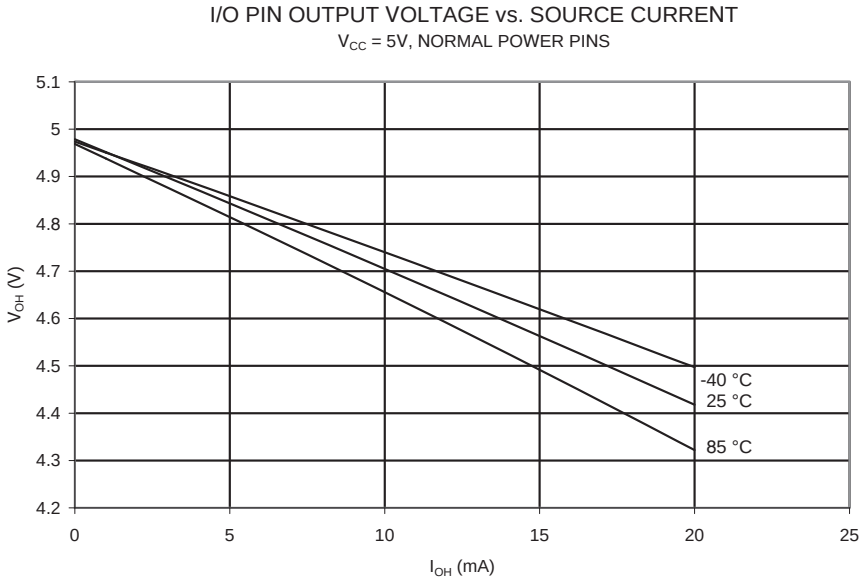


Figure 27-116. I/O Pin Output Voltage vs. Source Current(V_{CC} = 5 V).



27.3.9 Pin Threshold and Hysteresis

Figure 27-117. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin read as '1').

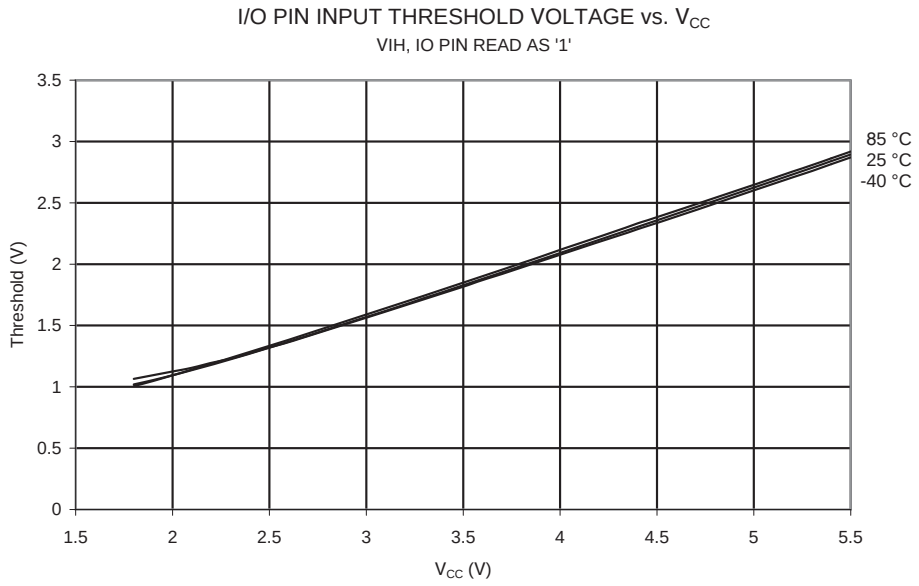


Figure 27-118. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O Pin read as '0').

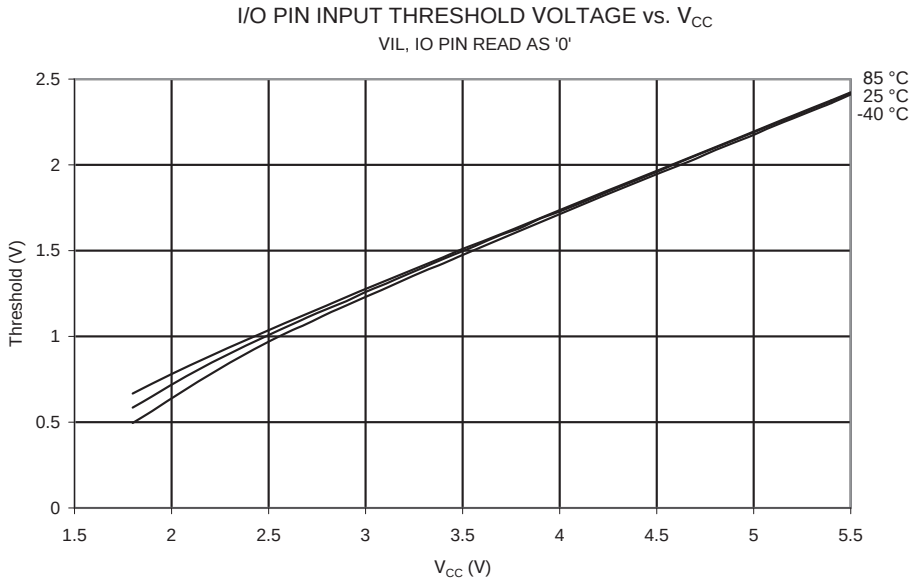


Figure 27-119. I/O Pin Input Hysteresis vs. V_{CC}.

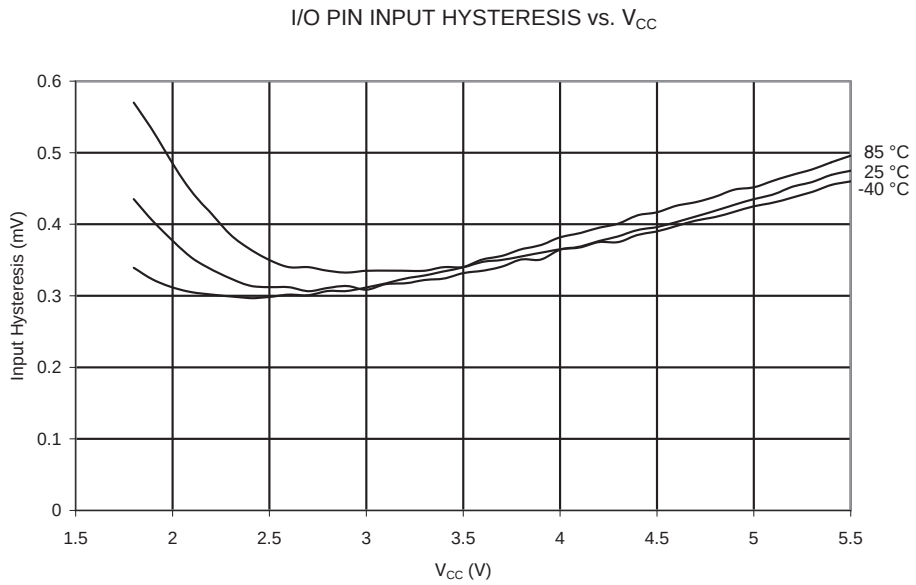


Figure 27-120. Reset Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin read as '1').

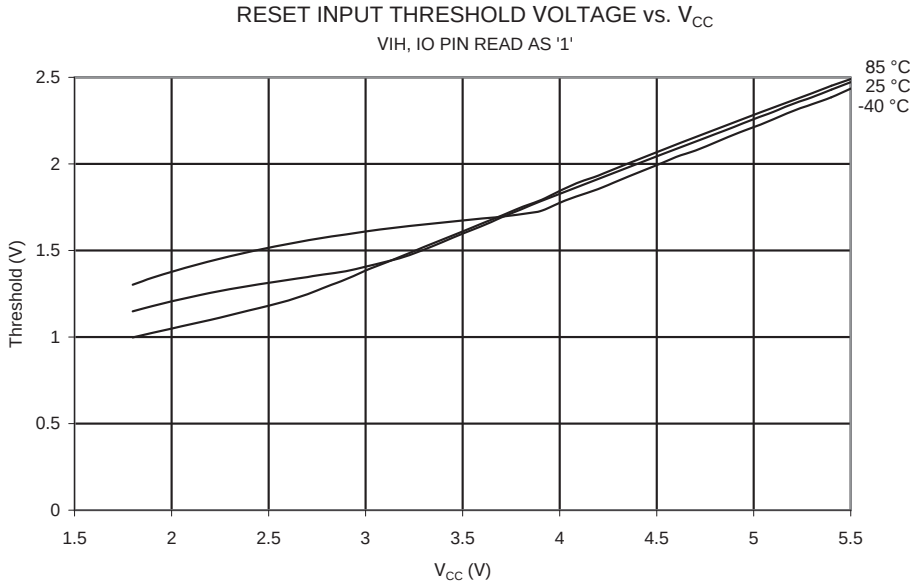


Figure 27-121. Reset Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O Pin read as '0').

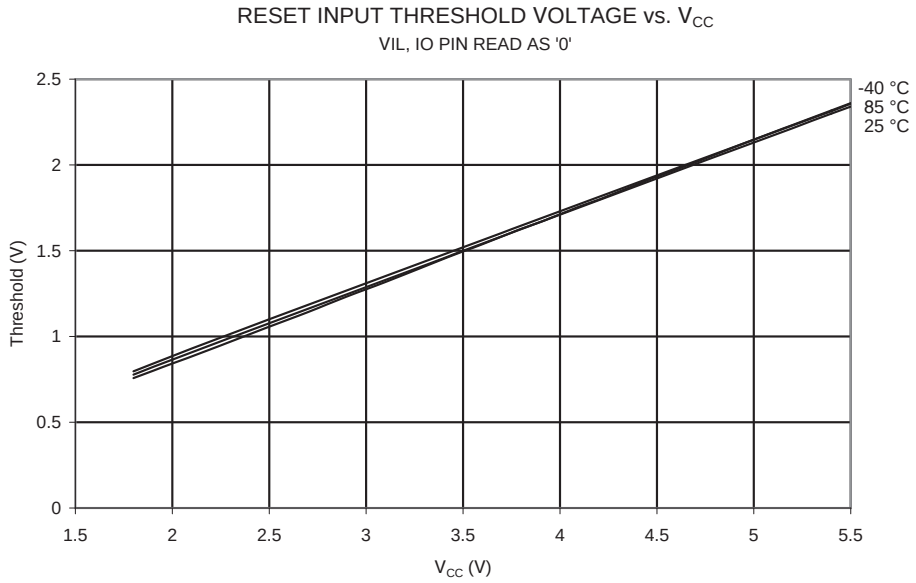
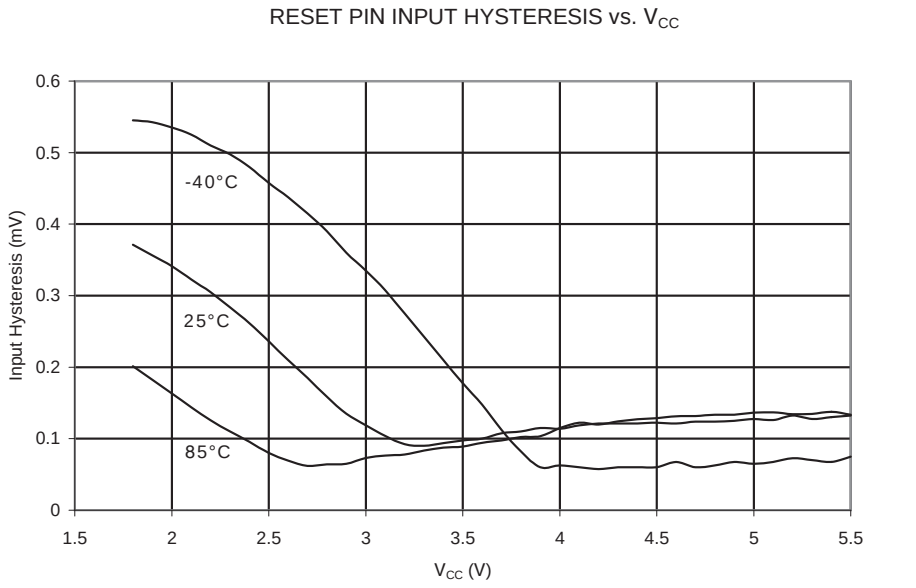


Figure 27-122. Reset Pin Input Hysteresis vs. V_{CC}.



27.3.10 BOD Threshold

Figure 27-123.BOD Thresholds vs. Temperature (BODLEVEL is 1.8 V).

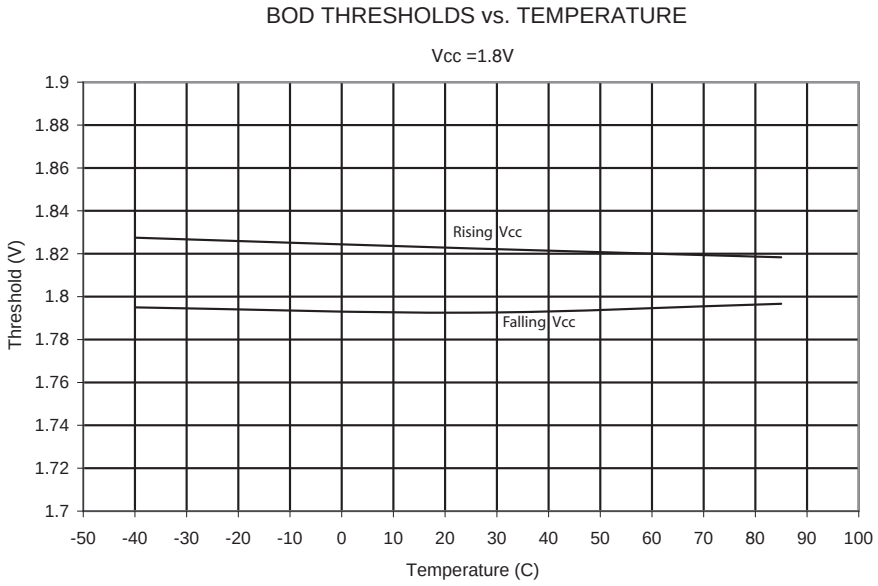


Figure 27-124.BOD Thresholds vs. Temperature (BODLEVEL is 2.7 V).

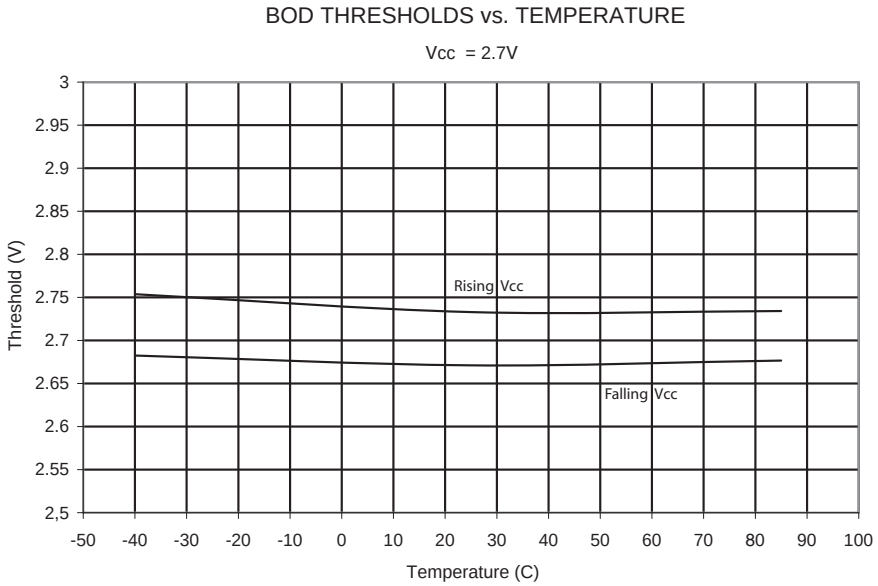
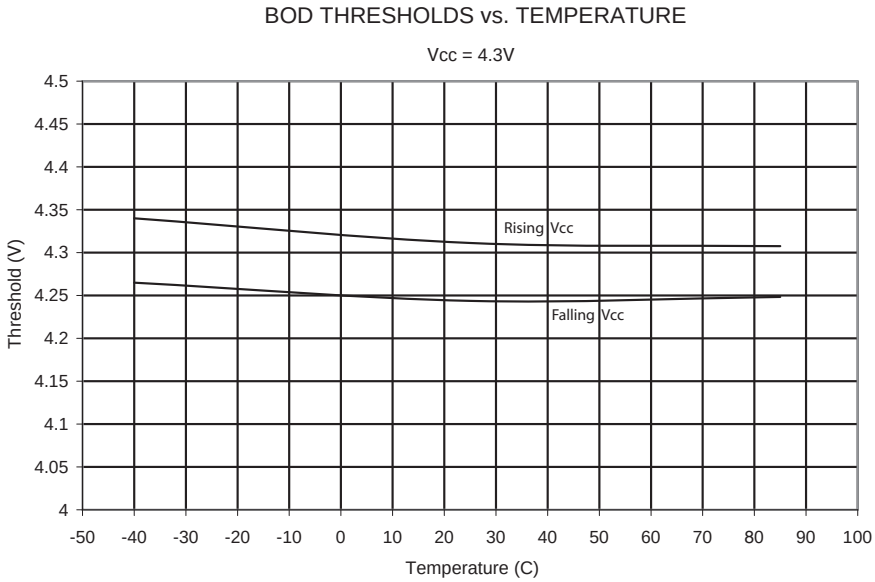


Figure 27-125.BOD Thresholds vs. Temperature (BODLEVEL is 4.3 V).



27.3.11 Internal Oscilllator Speed

Figure 27-126. Watchdog Oscillator Frequency vs. Temperature.

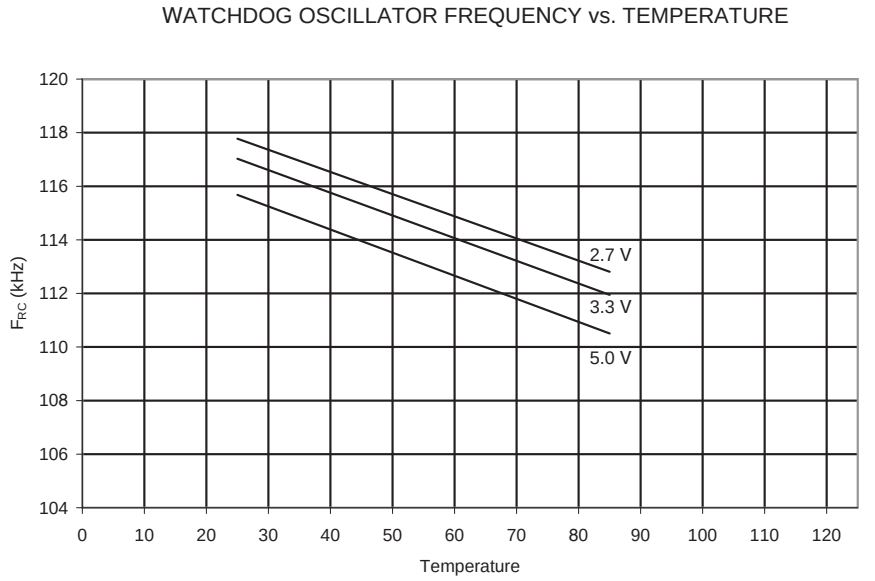


Figure 27-127. Watchdog Oscillator Frequency vs. V_{CC}.

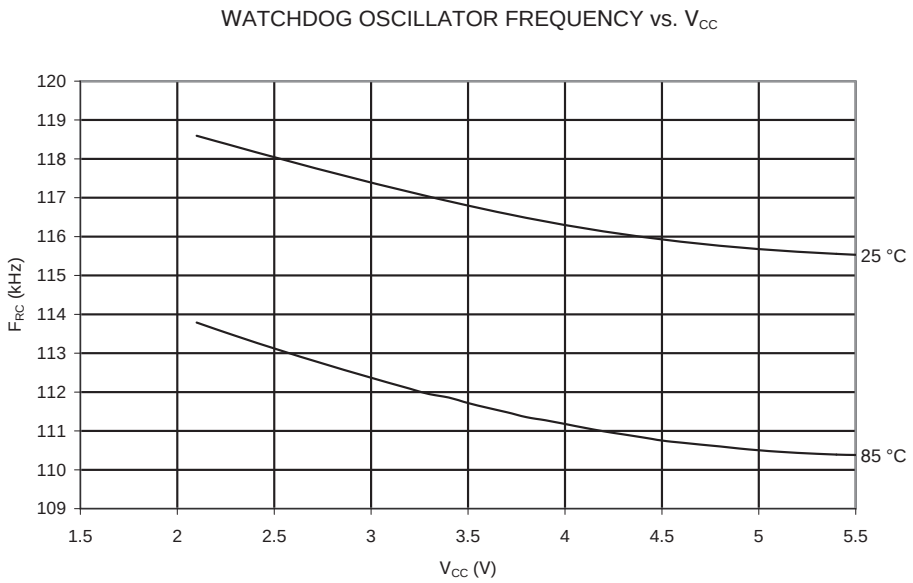


Figure 27-128. Calibrated 8 MHz RC Oscillator Frequency vs. V_{CC}.

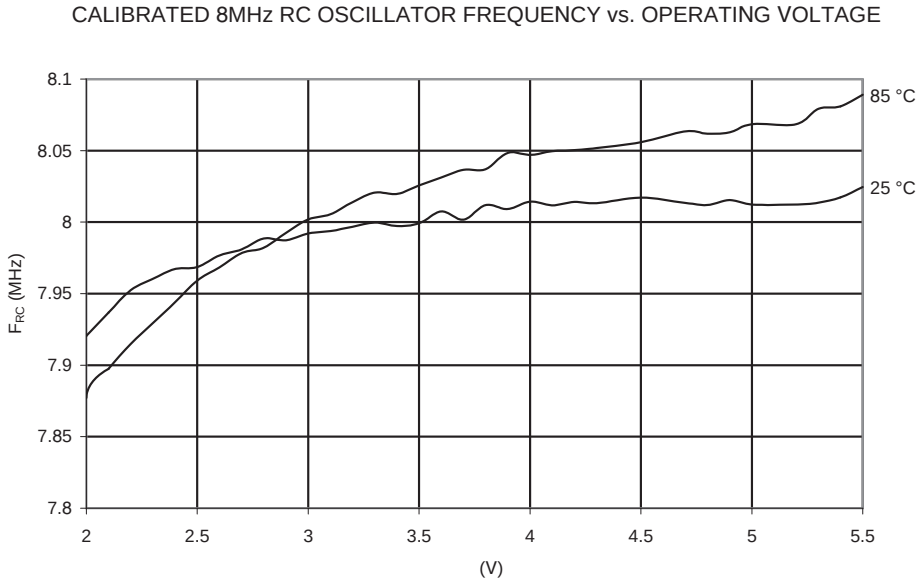


Figure 27-129. Calibrated 8 MHz RC Oscillator Frequency vs. Temperature.

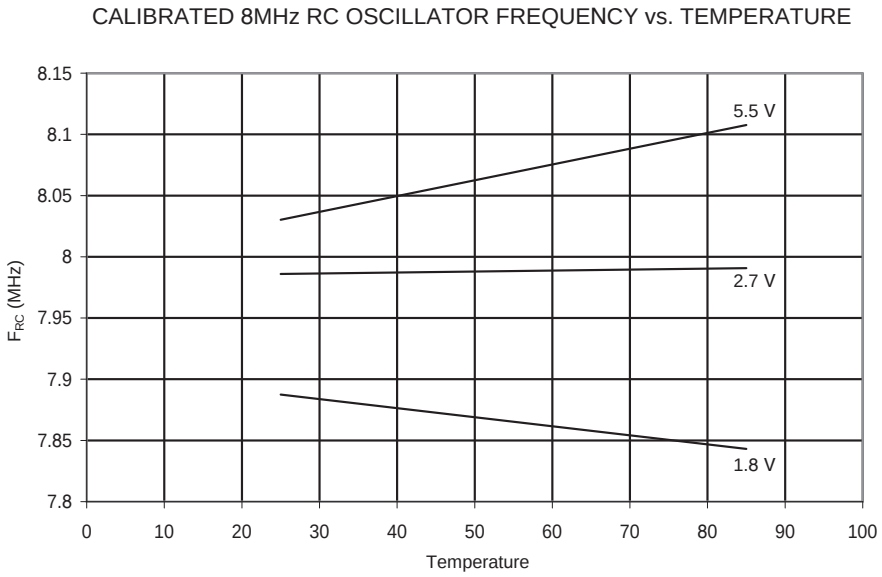
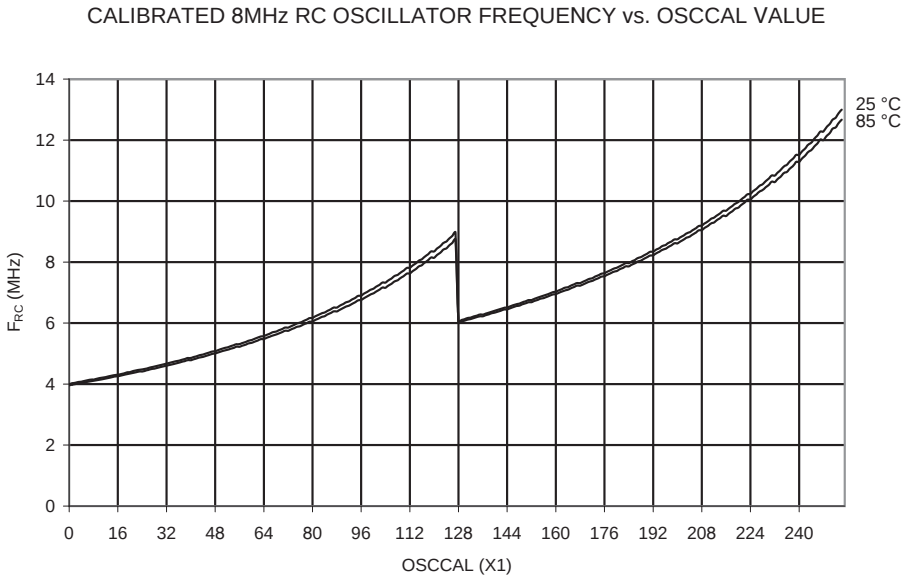


Figure 27-130. Calibrated 8 MHz RC Oscillator Frequency vs. OSCCAL Value.



27.3.12 Current Consumption of Peripheral Units

Figure 27-131. ADC Current vs. V_{CC} ($AREF = AV_{CC}$).

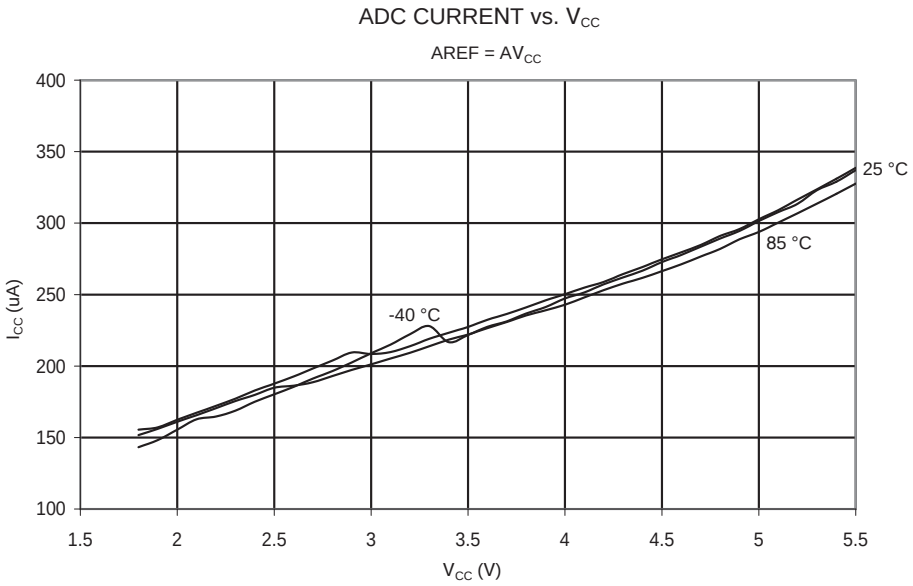


Figure 27-132. Analog Comparator Current vs. V_{CC}.

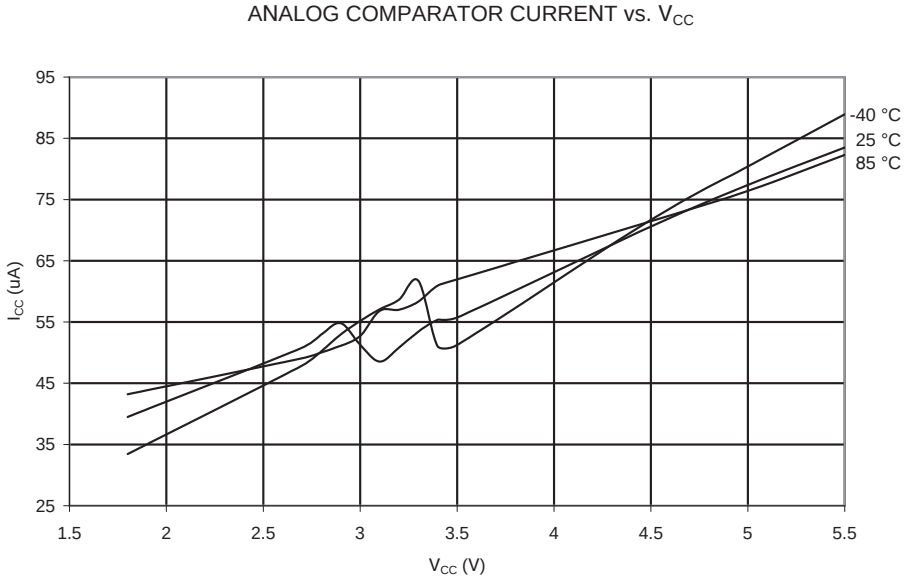


Figure 27-133.AREF External Reference Current vs. V_{CC}.

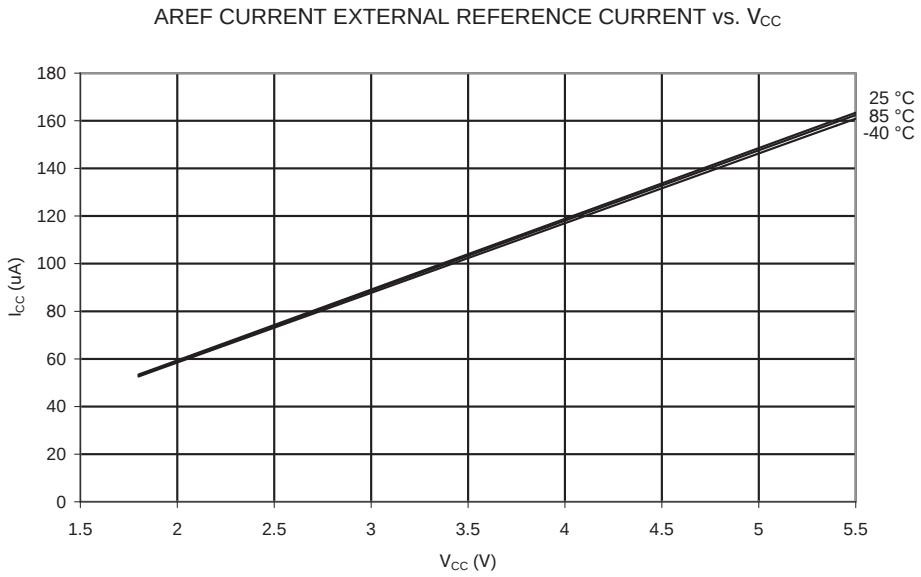


Figure 27-134. Brownout Detector Current vs. V_{CC}.

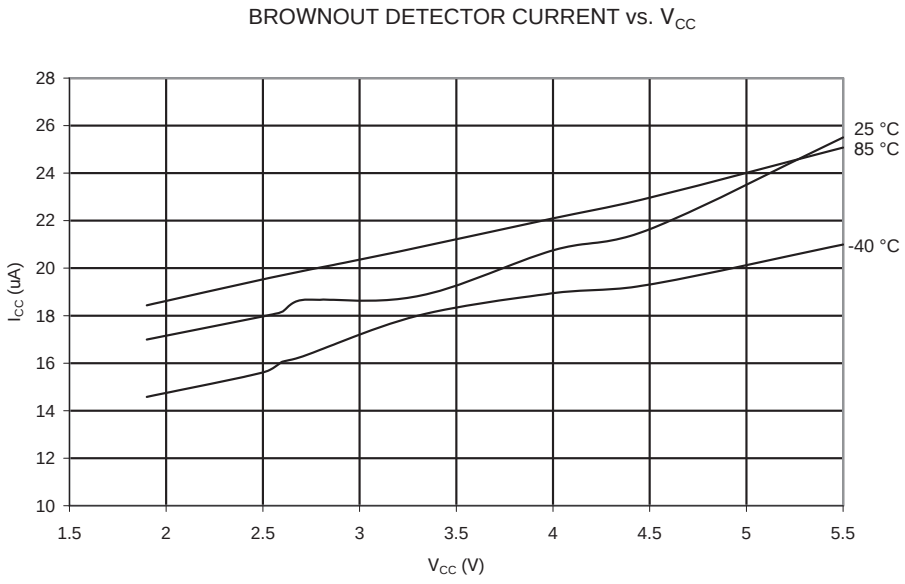
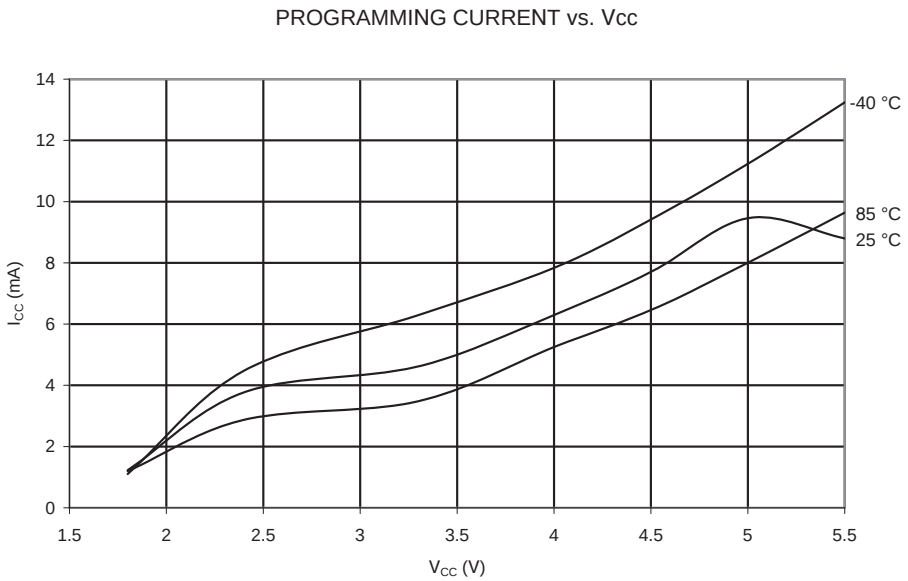


Figure 27-135. Programming Current vs. V_{CC}.



27.3.13 Current Consumption in Reset and Reset Pulsewidth

Figure 27-136. Reset Supply Current vs. Low Frequency (0.1 - 1.0 MHz).

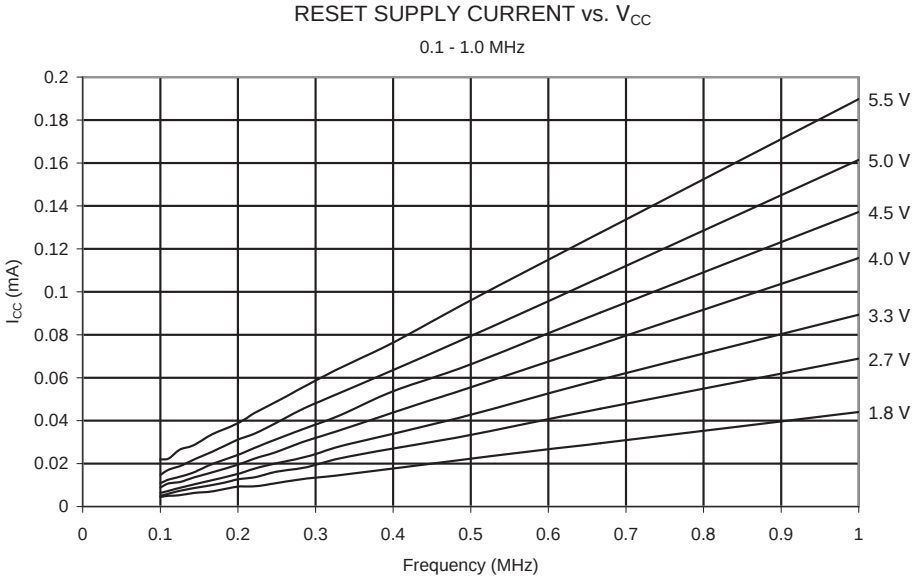


Figure 27-137. Reset Supply Current vs. Frequency (1 - 20 MHz).

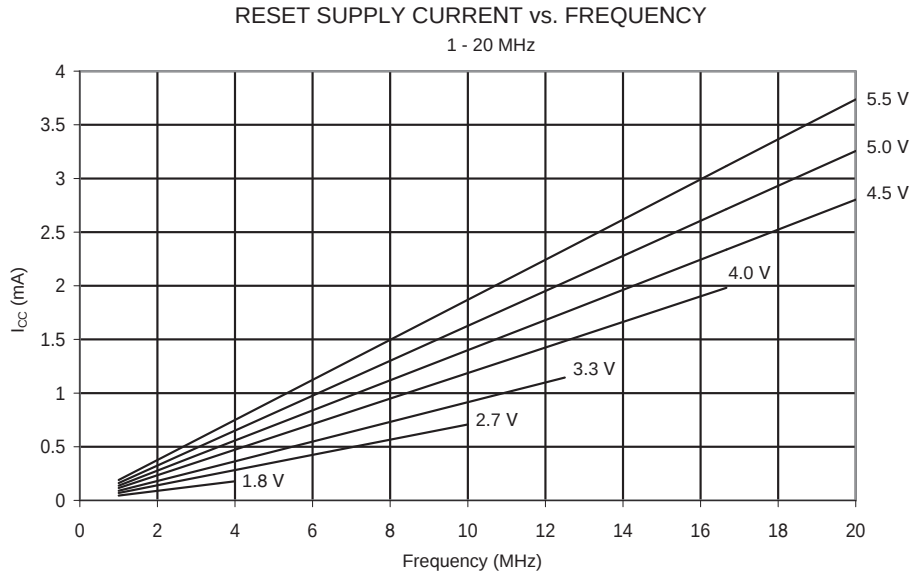
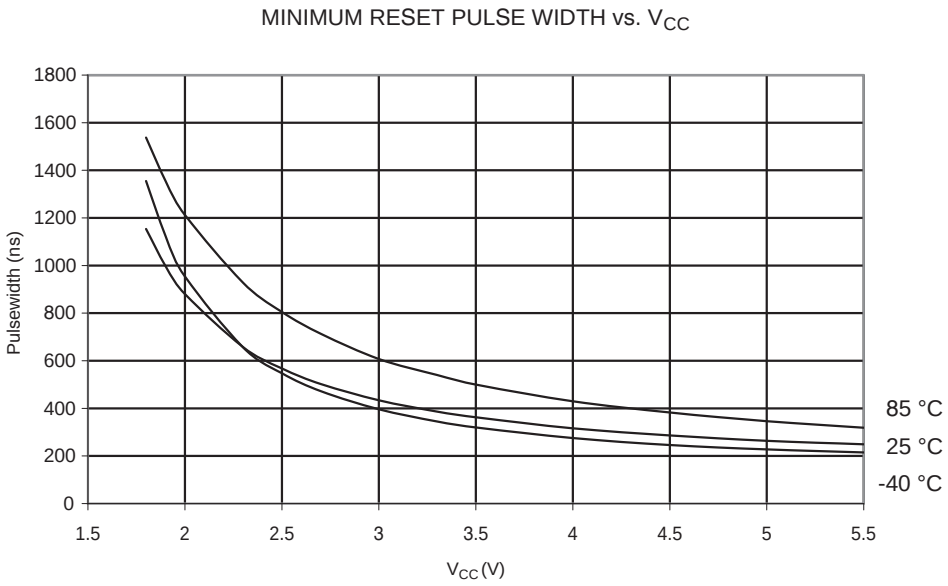


Figure 27-138. Minimum Reset Pulse width vs. V_{CC}.



27.4 ATmega328P Typical Characteristics

27.4.1 Active Supply Current

Figure 27-139. Active Supply Current vs. Low Frequency (0.1-1.0 MHz).

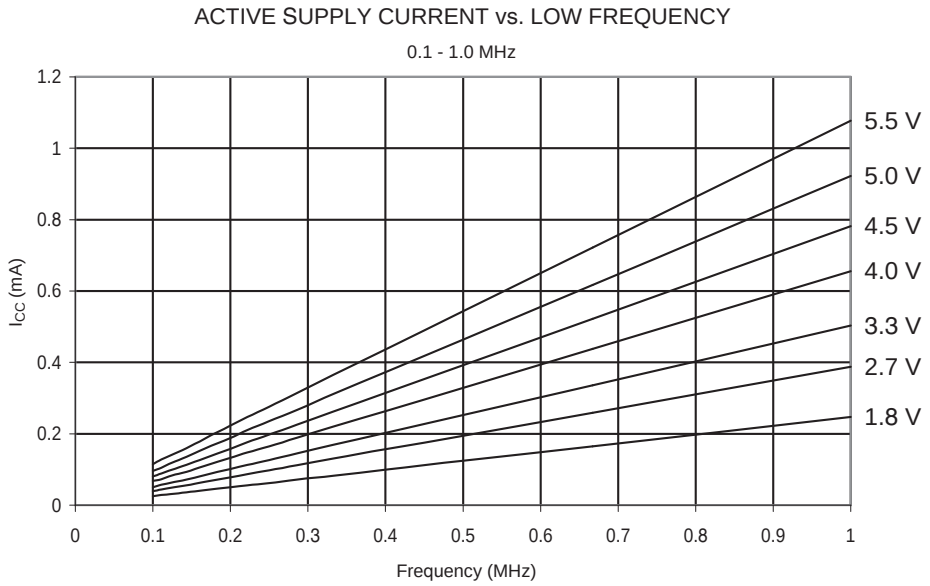


Figure 27-140. Active Supply Current vs. Frequency (1-20 MHz).

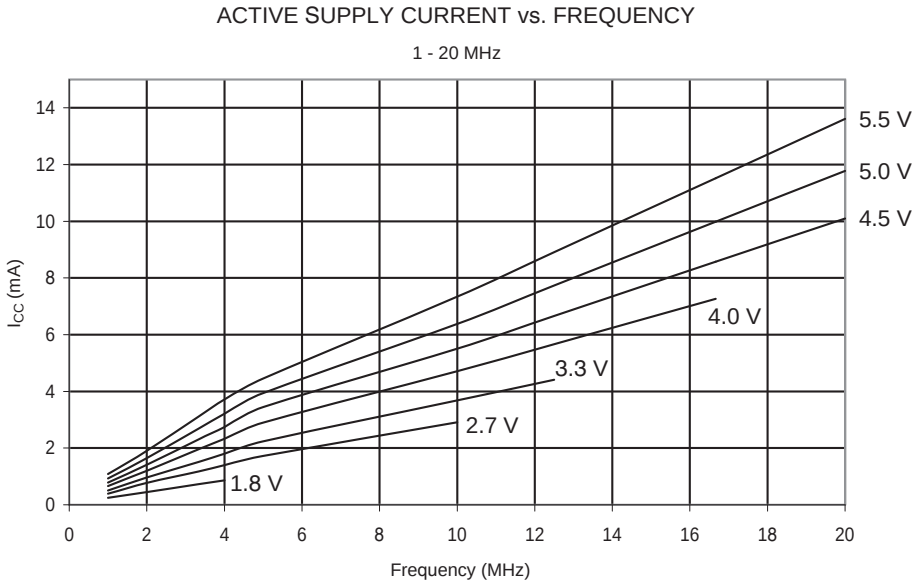


Figure 27-141. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz).

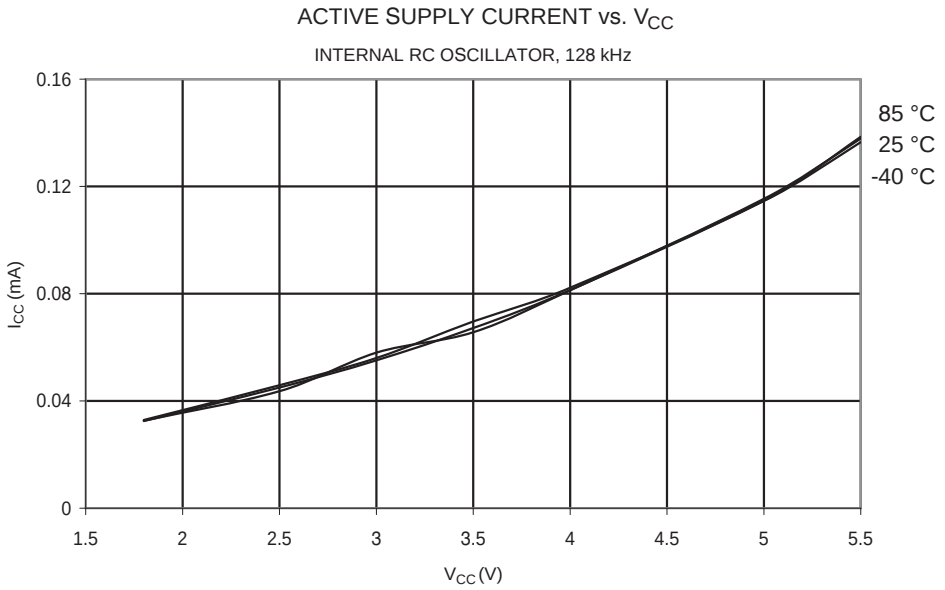


Figure 27-142. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz).

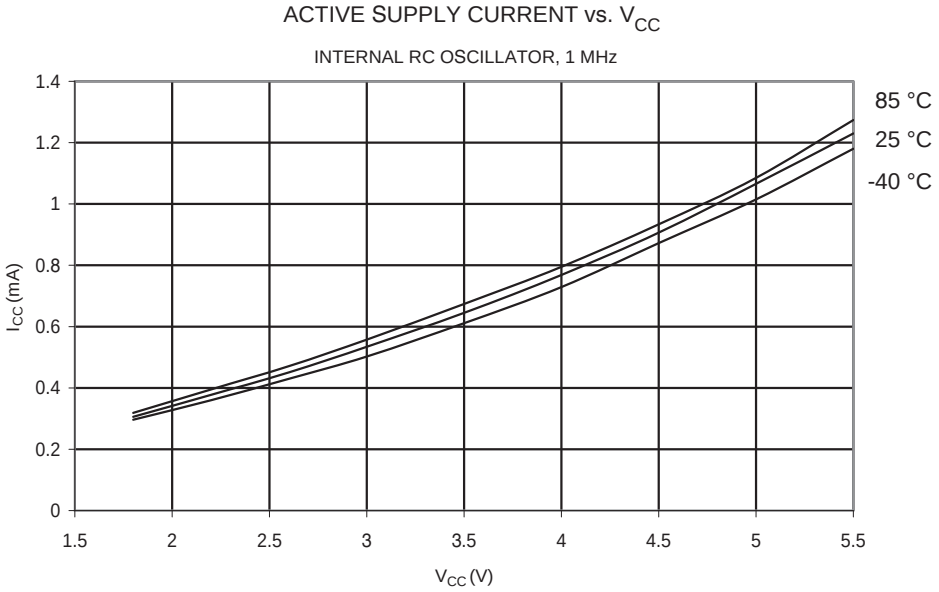
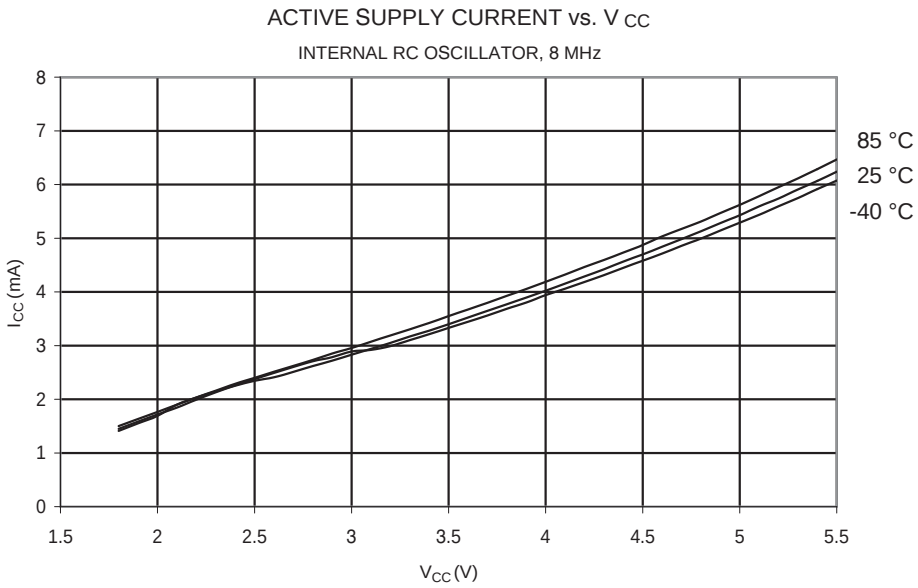


Figure 27-143. Active Supply Current vs. V_{CC} (Internal RC Oscillator, 8 MHz).



27.4.2 Idle Supply Current

Figure 27-144. Idle Supply Current vs. Low Frequency (0.1-1.0 MHz).

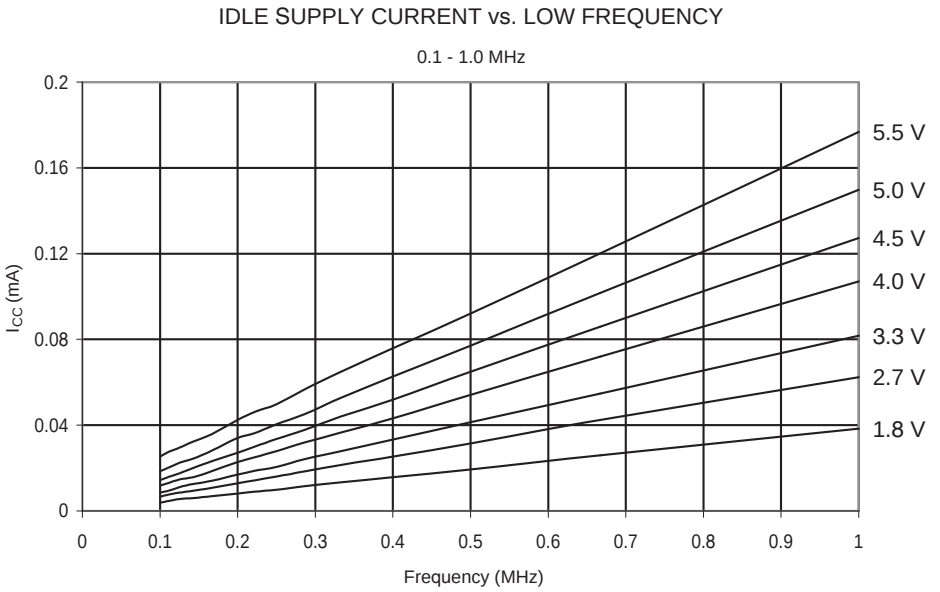


Figure 27-145. Idle Supply Current vs. Frequency (1-20 MHz).

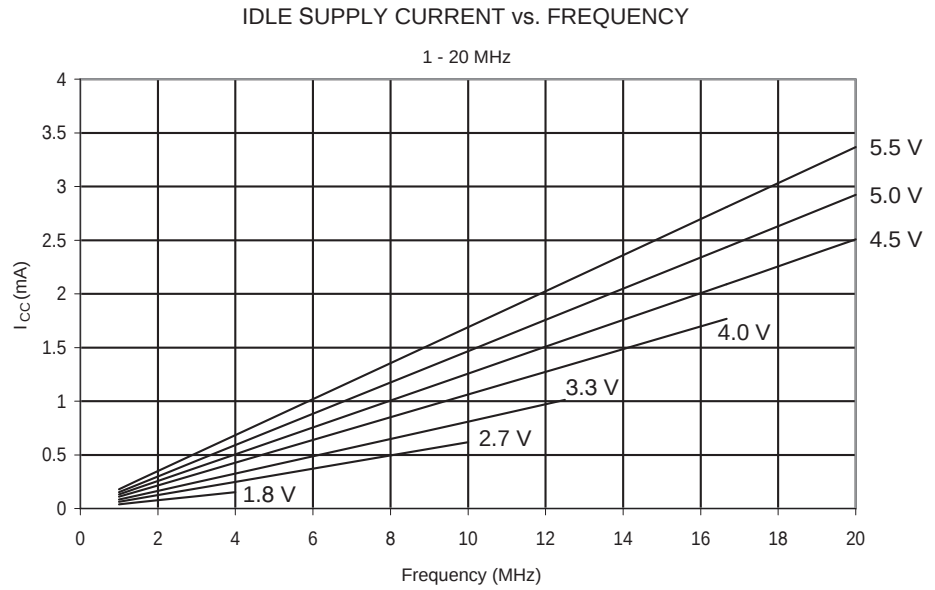


Figure 27-146. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 128 kHz).

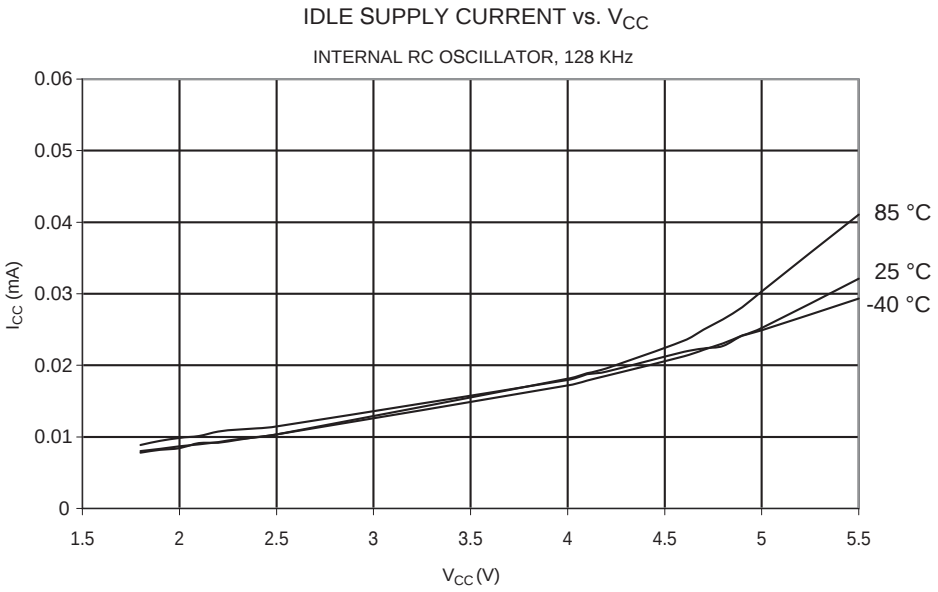


Figure 27-147. Idle Supply Current vs. V_{CC} (Internal RC Oscillator, 1 MHz).

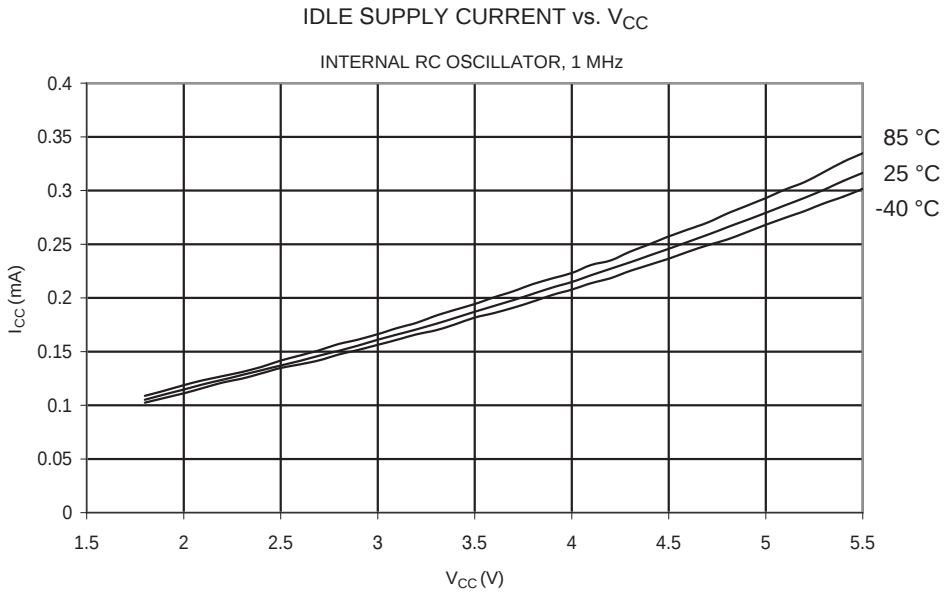
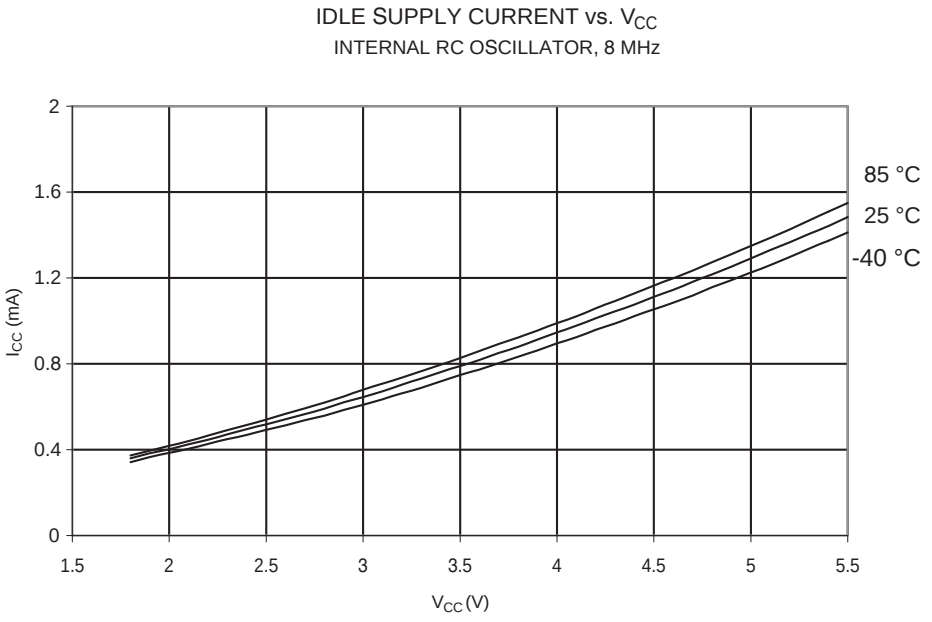


Figure 27-148. Idle Supply Current vs. Vcc (Internal RC Oscillator, 8 MHz).



27.4.3 Supply Current of IO Modules

The tables and formulas below can be used to calculate the additional current consumption for the different I/O modules in Active and Idle mode. The enabling or disabling of the I/O modules are controlled by the Power Reduction Register. See "Power Reduction Register" on page 42 for details.

Table 27-7. Additional Current Consumption for the different I/O modules (absolute values)

PRR bit	Typical numbers		
	V_{CC} = 2V, F = 1MHz	V_{CC} = 3V, F = 4MHz	V_{CC} = 5V, F = 8MHz
PRUSART0	3.20 µA	22.17 µA	100.25 µA
PRTWI	7.34 µA	46.55 µA	199.25 µA
PRTIM2	7.34 µA	50.79 µA	224.25 µA
PRTIM1	6.19 µA	41.25 µA	176.25 µA
PRTIM0	1.89 µA	14.28 µA	61.13 µA
PRSPI	6.94 µA	43.84 µA	186.50 µA
PRADC	8.66 µA	61.80 µA	295.38 µA

Table 27-8. Additional Current Consumption (percentage) in Active and Idle mode

PRR bit	Additional Current consumption compared to Active with external clock (see Figure 27-139 on page 401 and Figure 27-140 on page 401)	Additional Current consumption compared to Idle with external clock (see Figure 27-144 on page 403 and Figure 27-145 on page 404)
PRUSART0	1.4 %	7.8%
PRTWI	3.0 %	16.6 %
PRTIM2	3.3 %	17.8 %
PRTIM1	2.7 %	14.5 %
PRTIM0	0.9 %	4.8 %
PRSPI	2.9 %	15.7 %
PRADC	4.1 %	22.1 %

It is possible to calculate the typical current consumption based on the numbers from Table 27-8 on page 406 for other V_{CC} and frequency settings than listed in Table 27-7 on page 406.

Example

Calculate the expected current consumption in idle mode with TIMER1, ADC, and SPI enabled at V_{CC} = 2.0V and F = 1MHz. From Table 27-8 on page 406, third column, we see that we need to add 14.5% for the TIMER1, 22.1% for the ADC, and 15.7% for the SPI module. Reading from Figure 27-145 on page 404, we find that the idle current consumption is ~0.055 mA at V_{CC} = 2.0V and F = 1MHz. The total current consumption in idle mode with TIMER1, ADC, and SPI enabled, gives:

$$I_{CCtotal} \approx 0.045\ mA \cdot (1 + 0.145 + 0.221 + 0.157) \approx 0.069\ mA$$

27.4.4 Power-down Supply Current

Figure 27-149. Power-Down Supply Current vs. V_{CC} (Watchdog Timer Disabled).

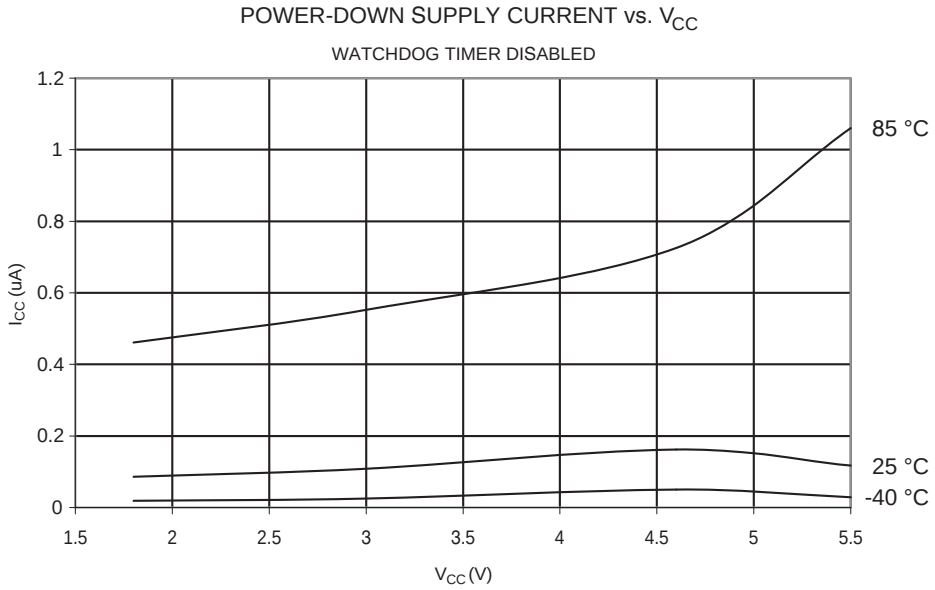
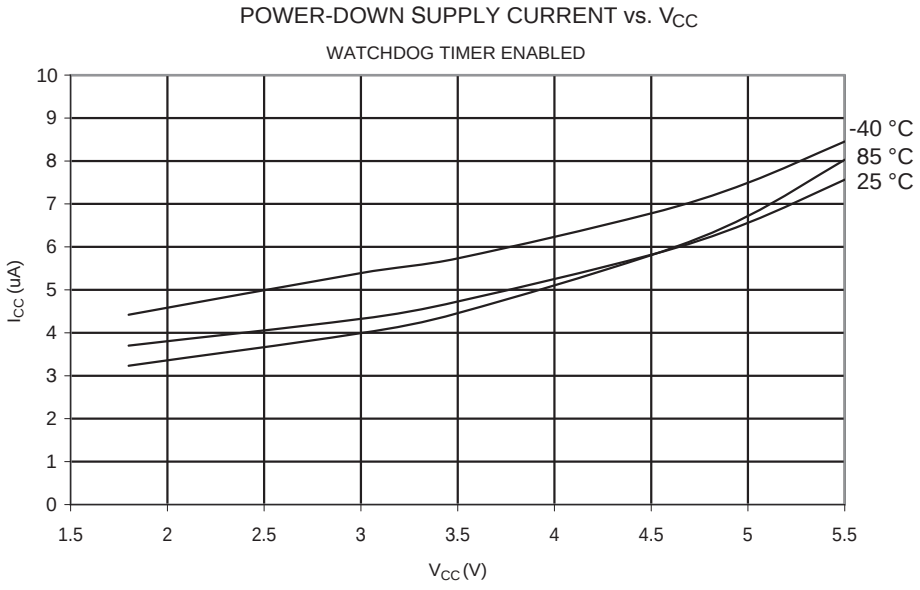
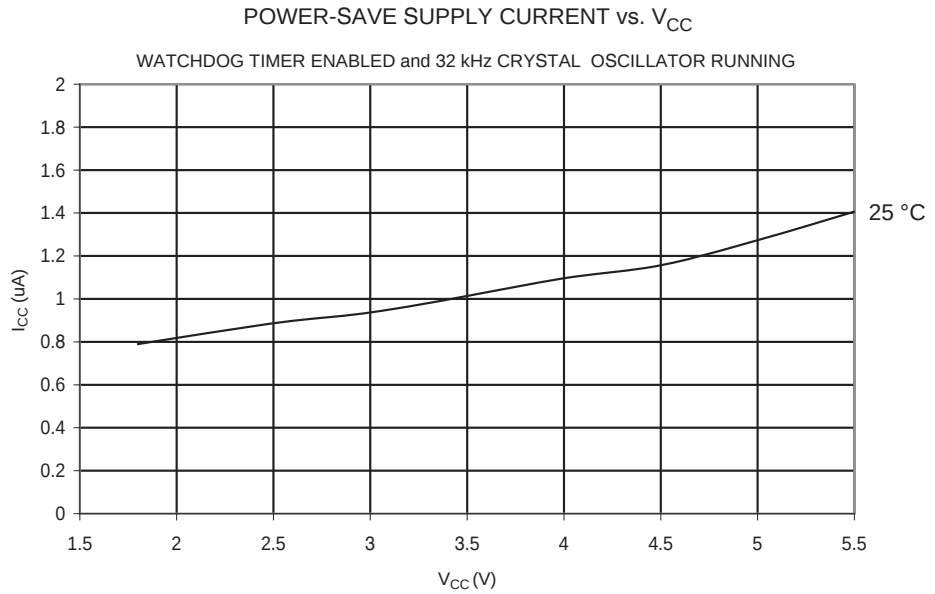


Figure 27-150. Power-Down Supply Current vs. V_{CC} (Watchdog Timer Enabled).



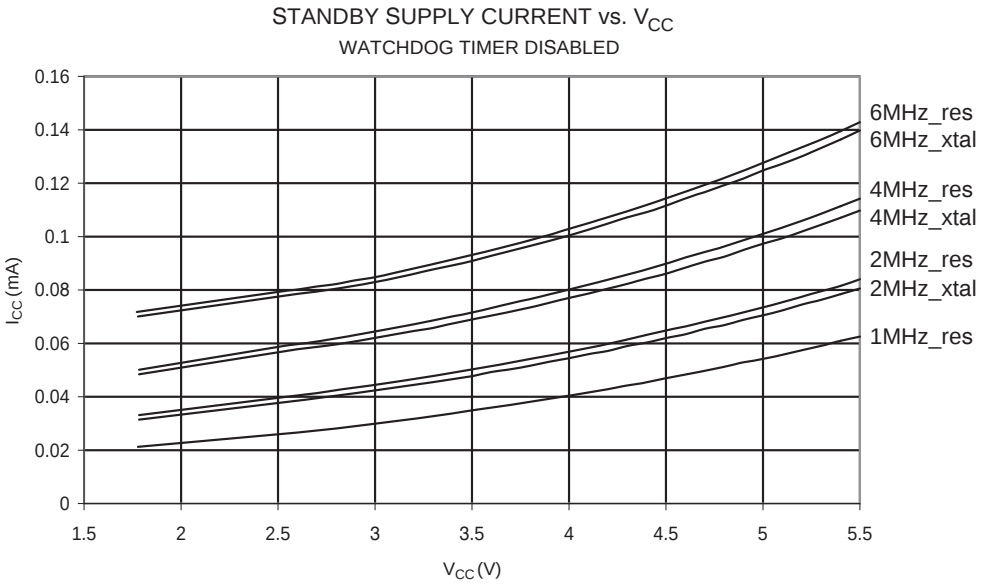
27.4.5 Power-save Supply Current

Figure 27-151. Power-Save Supply Current vs. V_{CC} (Watchdog Timer Disabled and 32 kHz Crystal Oscillator Running).



27.4.6 Standby Supply Current

Figure 27-152. Standby Supply Current vs. Vcc (Watchdog Timer Disabled).



27.4.7 Pin Pull-Up

Figure 27-153. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 1.8 V).

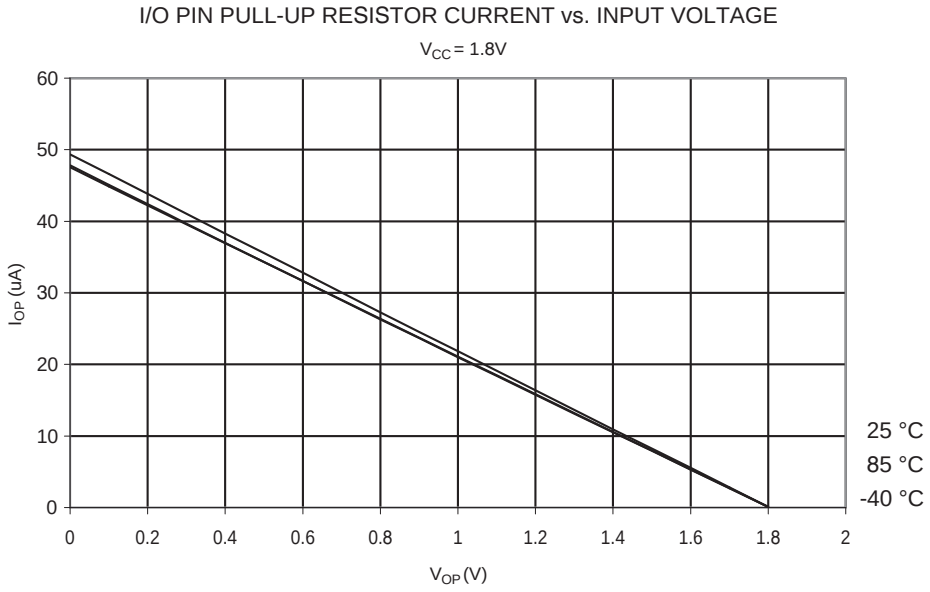


Figure 27-154. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 2.7 V).

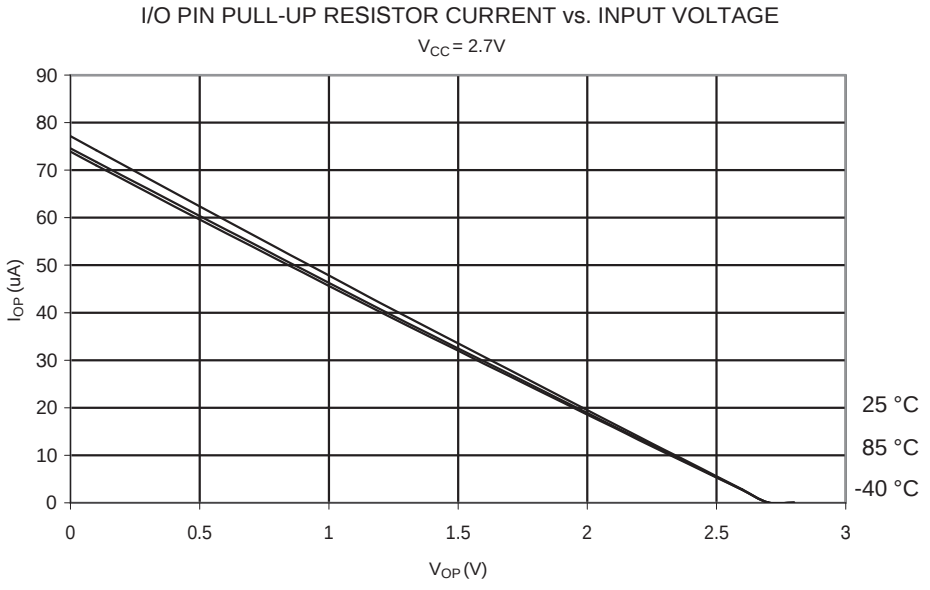


Figure 27-155. I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 5 V).

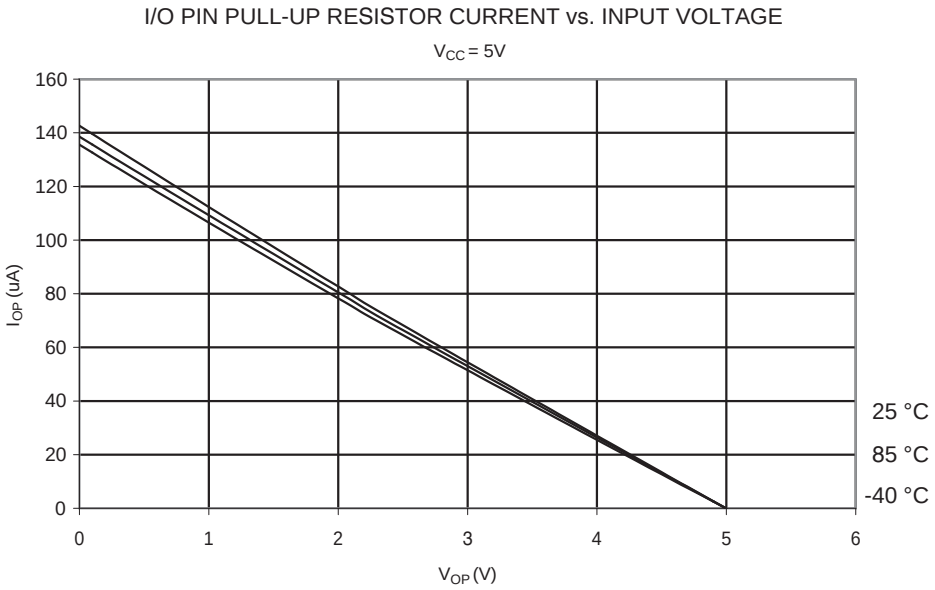


Figure 27-156. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 1.8 V).

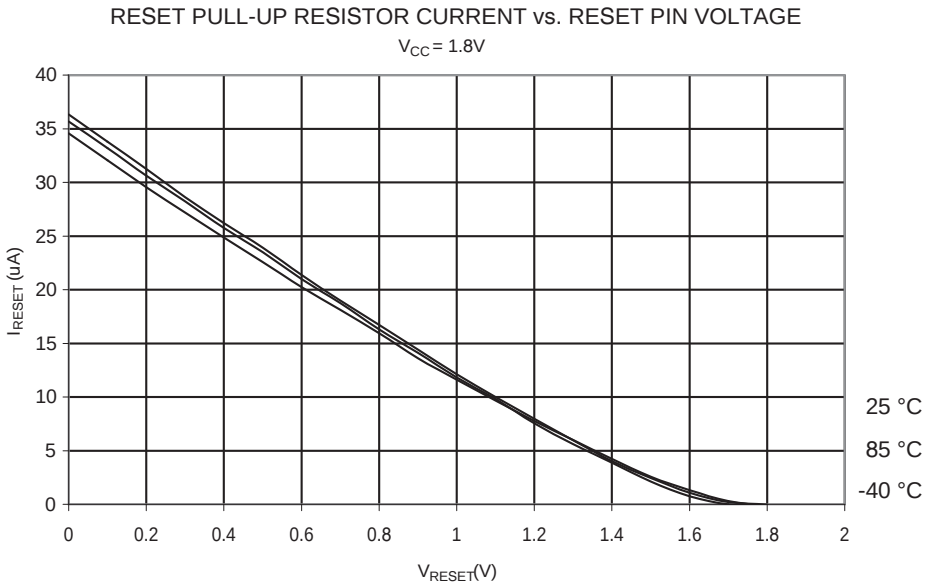


Figure 27-157. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 2.7 V).

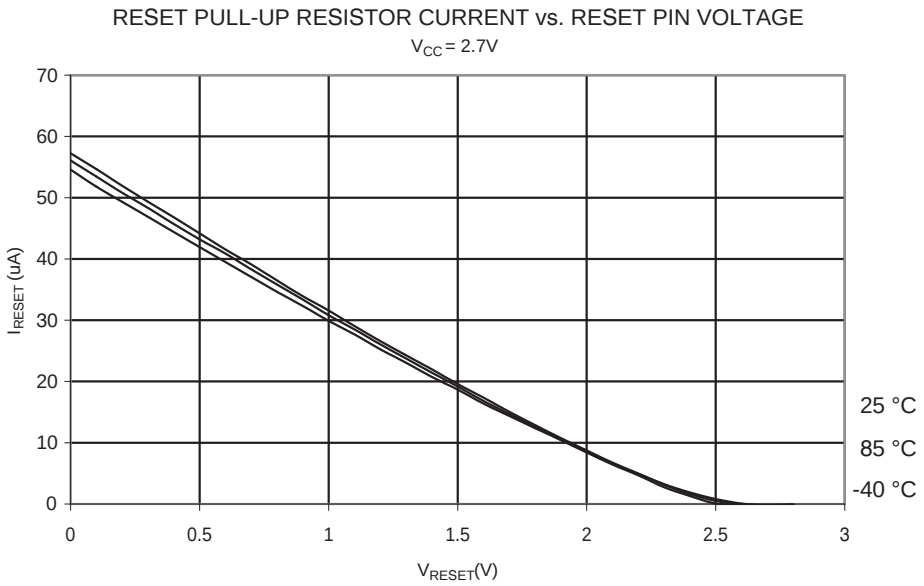
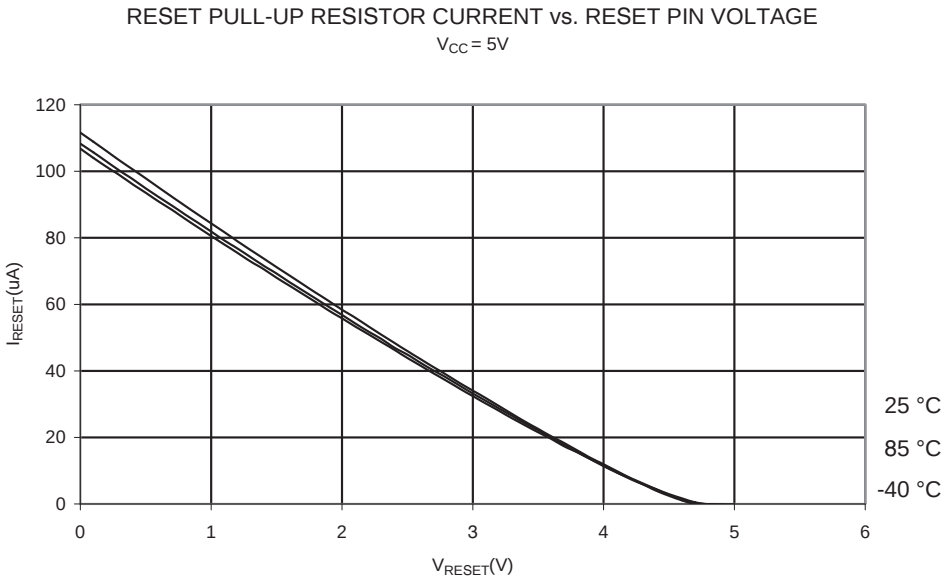


Figure 27-158. Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 5 V).



27.4.8 Pin Driver Strength

Figure 27-159. I/O Pin Output Voltage vs. Sink Current(V_{CC} = 3 V).

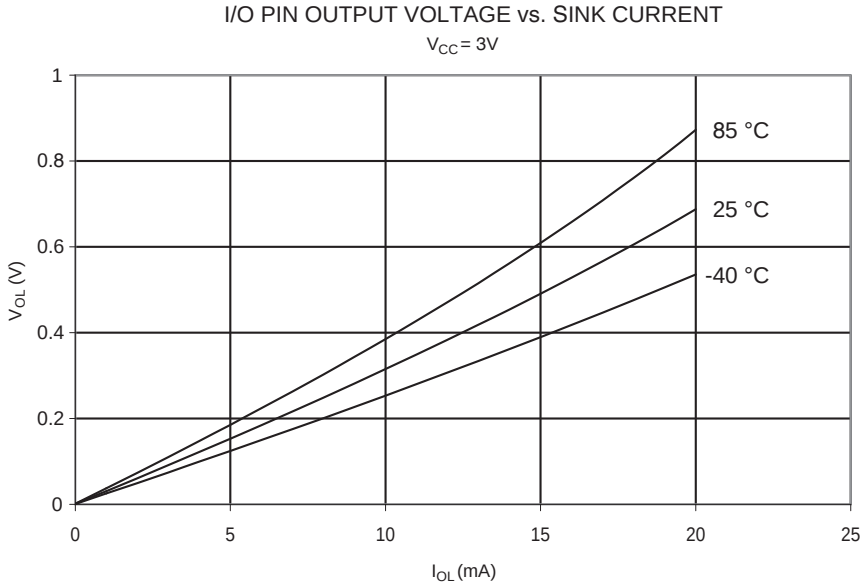


Figure 27-160. I/O Pin Output Voltage vs. Sink Current(V_{CC} = 5 V).

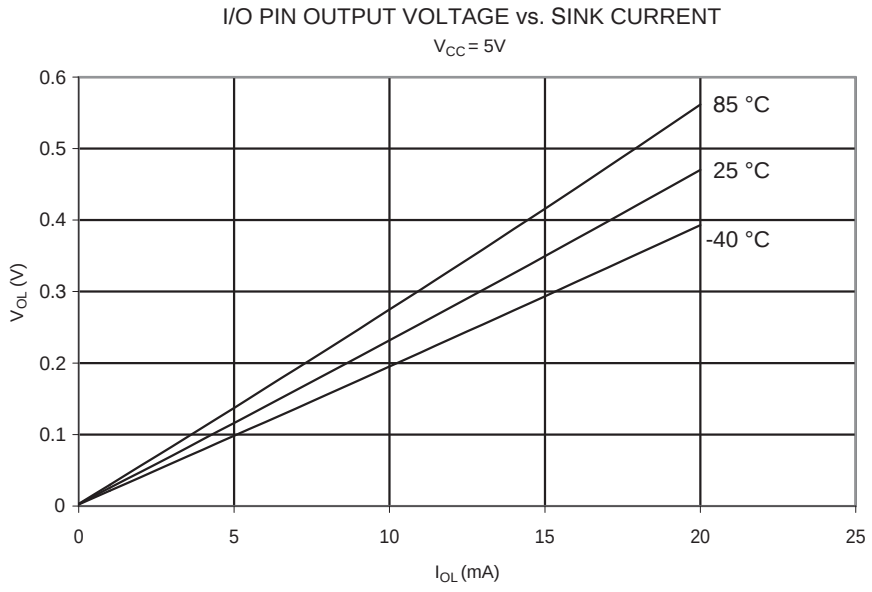


Figure 27-161. I/O Pin Output Voltage vs. Source Current(Vcc = 3 V).

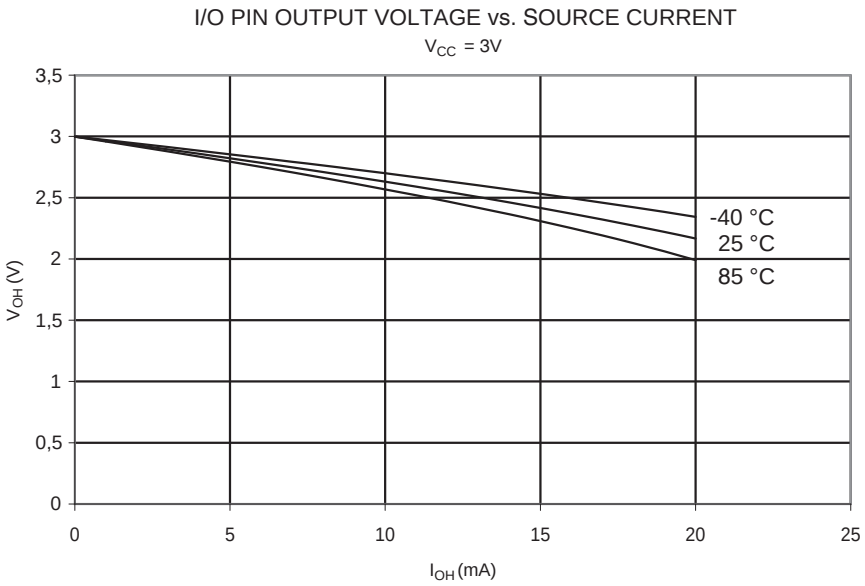
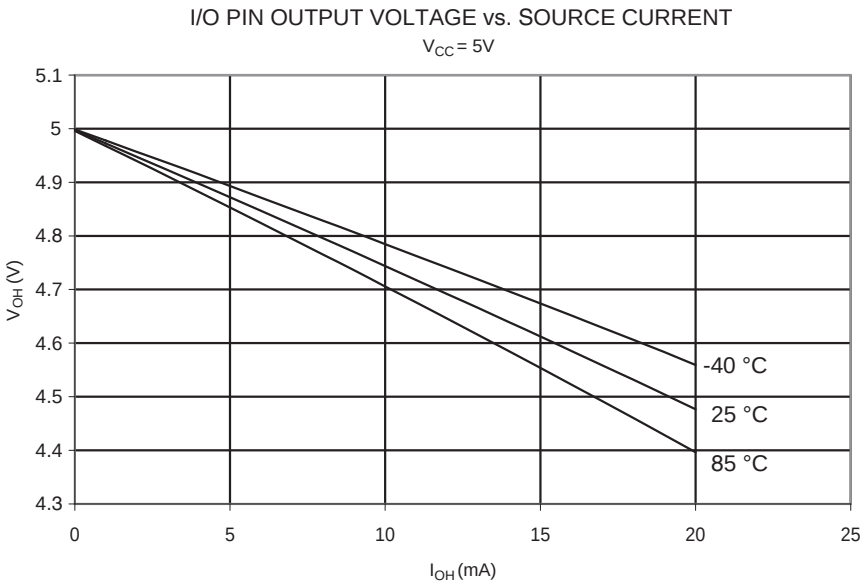


Figure 27-162. I/O Pin Output Voltage vs. Source Current(V_{CC} = 5 V).



27.4.9 Pin Threshold and Hysteresis

Figure 27-163. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin read as '1').

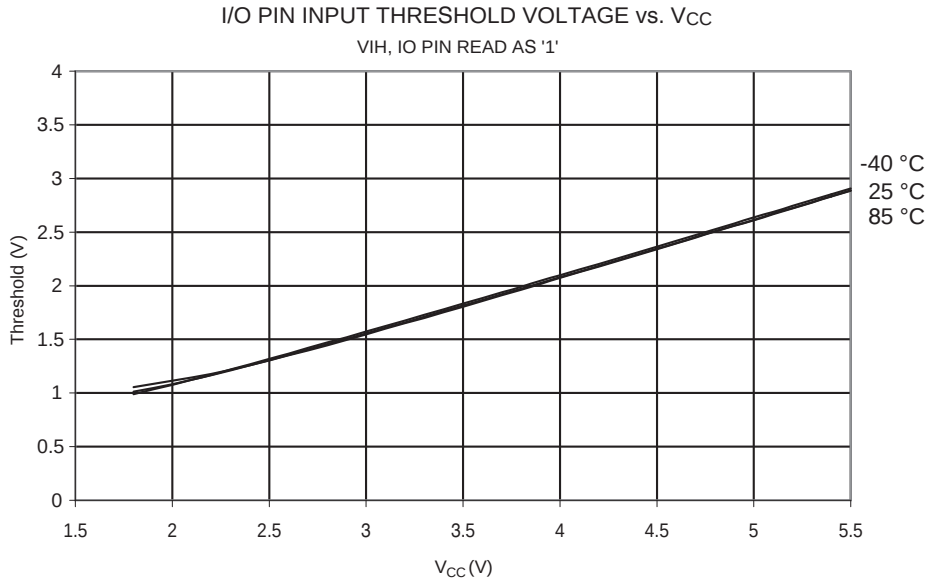


Figure 27-164. I/O Pin Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O Pin read as '0').

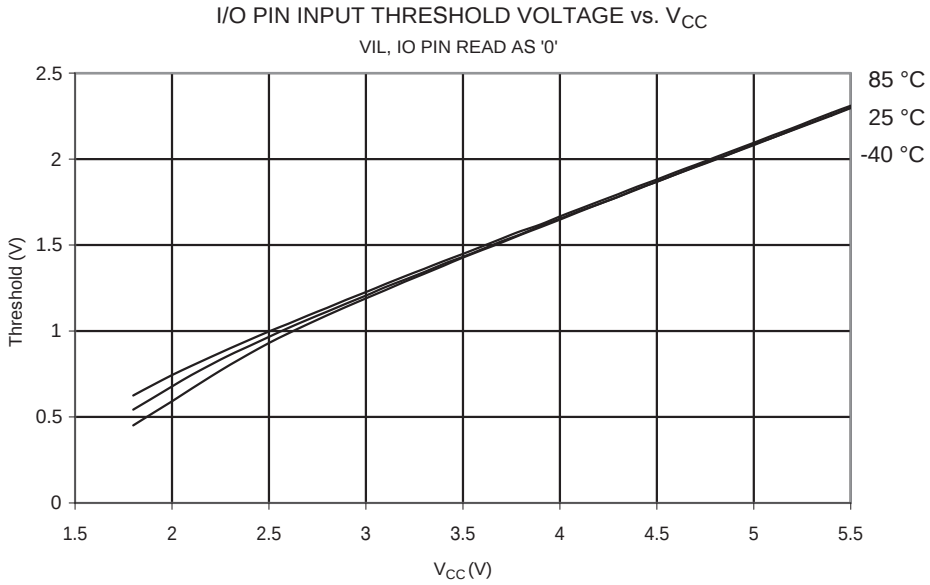


Figure 27-165. I/O Pin Input Hysteresis vs. V_{CC}.

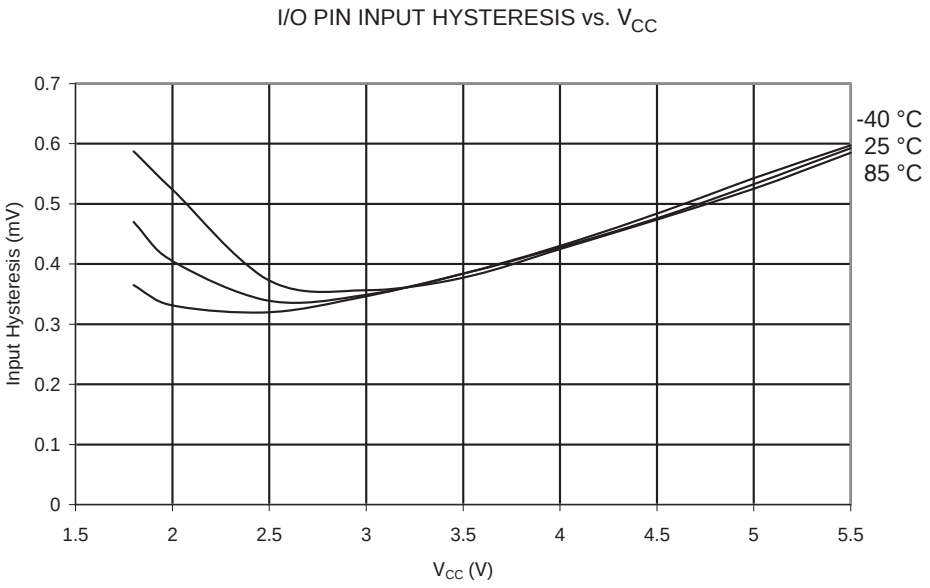


Figure 27-166. Reset Input Threshold Voltage vs. V_{CC} (V_{IH}, I/O Pin read as '1').

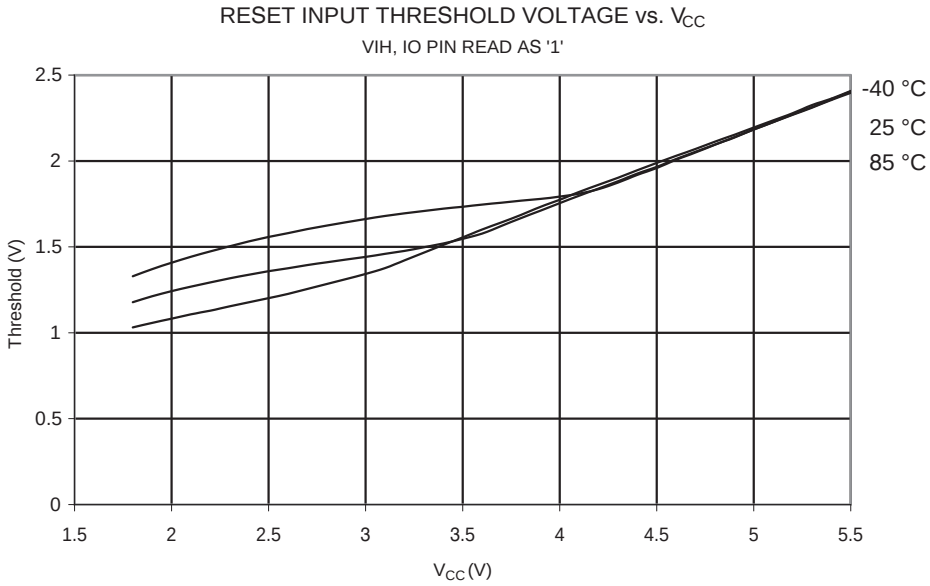


Figure 27-167. Reset Input Threshold Voltage vs. V_{CC} (V_{IL}, I/O Pin read as '0').

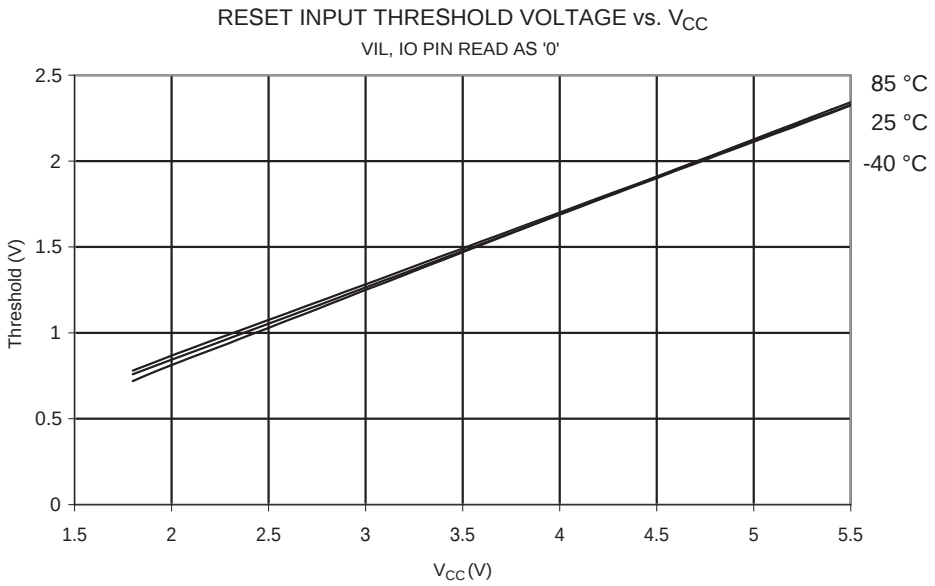
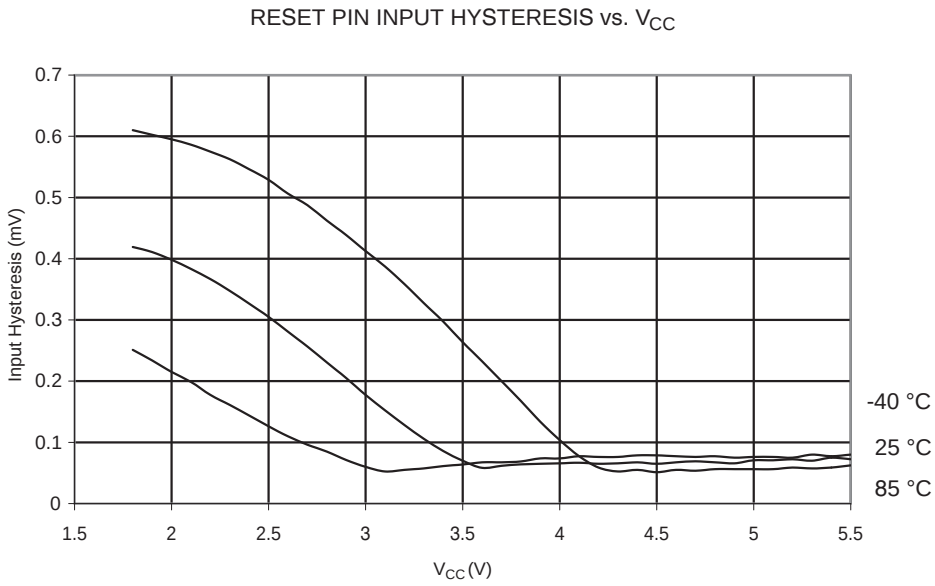


Figure 27-168. Reset Pin Input Hysteresis vs. V_{CC}.



27.4.10 BOD Threshold

Figure 27-169.BOD Thresholds vs. Temperature (BODLEVEL is 1.8 V).

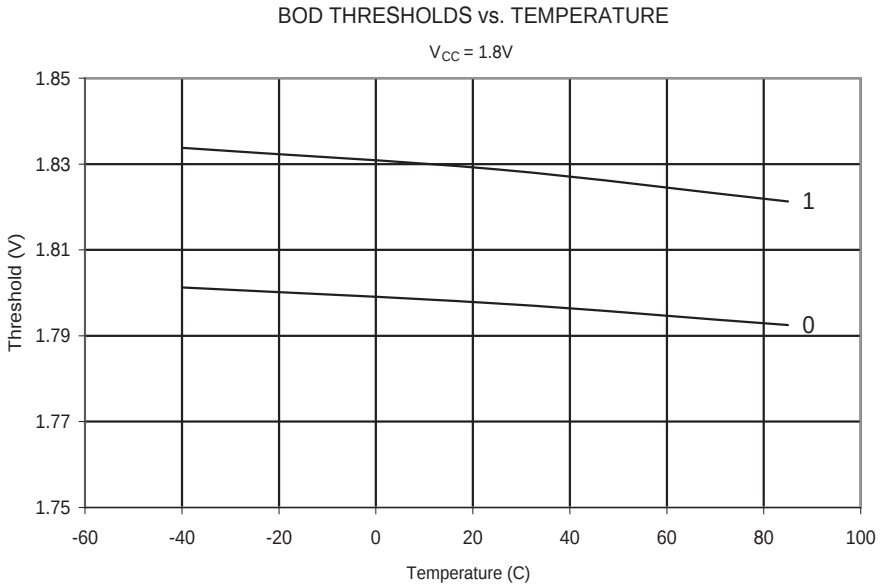


Figure 27-170.BOD Thresholds vs. Temperature (BODLEVEL is 2.7 V).

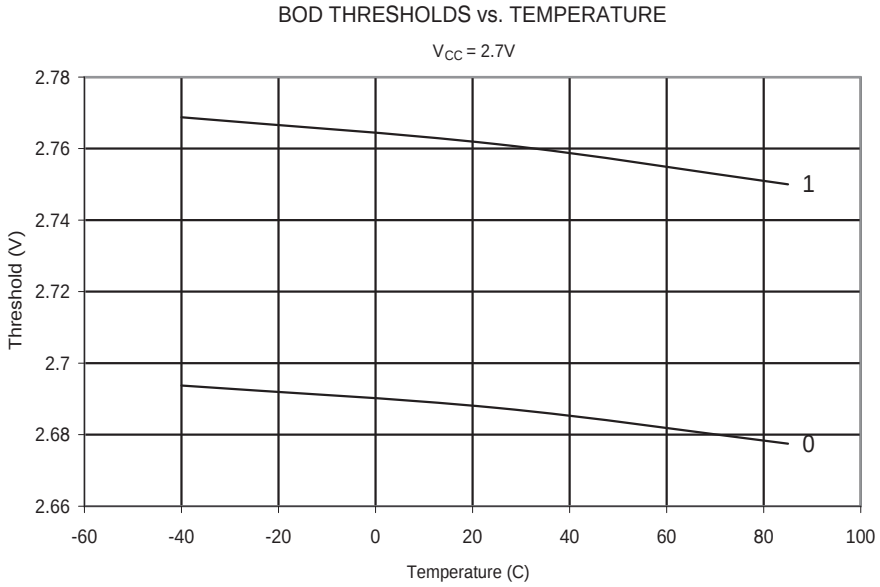
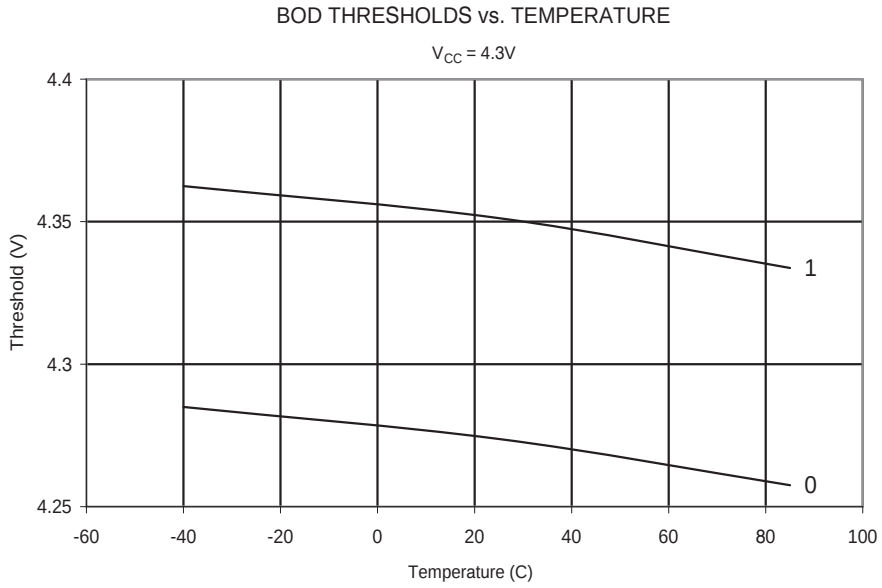


Figure 27-171.BOD Thresholds vs. Temperature (BODLEVEL is 4.3 V).



27.4.11 Internal Oscilllator Speed

Figure 27-172. Watchdog Oscillator Frequency vs. Temperature.

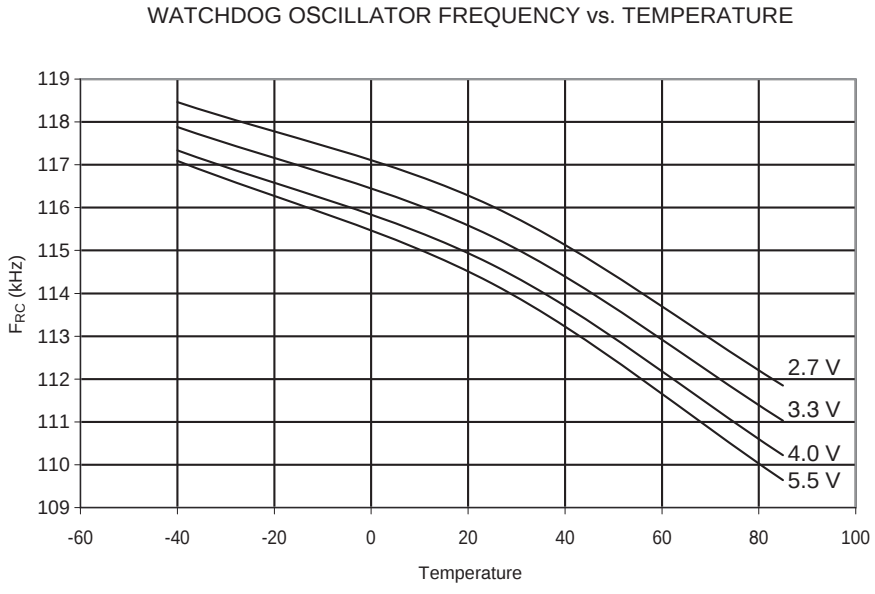


Figure 27-173. Watchdog Oscillator Frequency vs. V_{CC}.

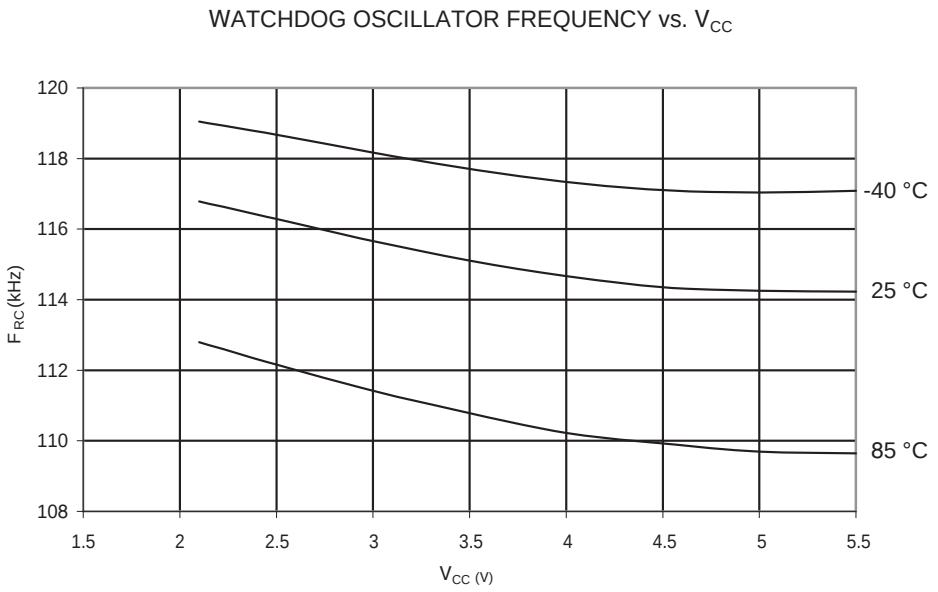


Figure 27-174. Calibrated 8 MHz RC Oscillator Frequency vs. V_{CC}.

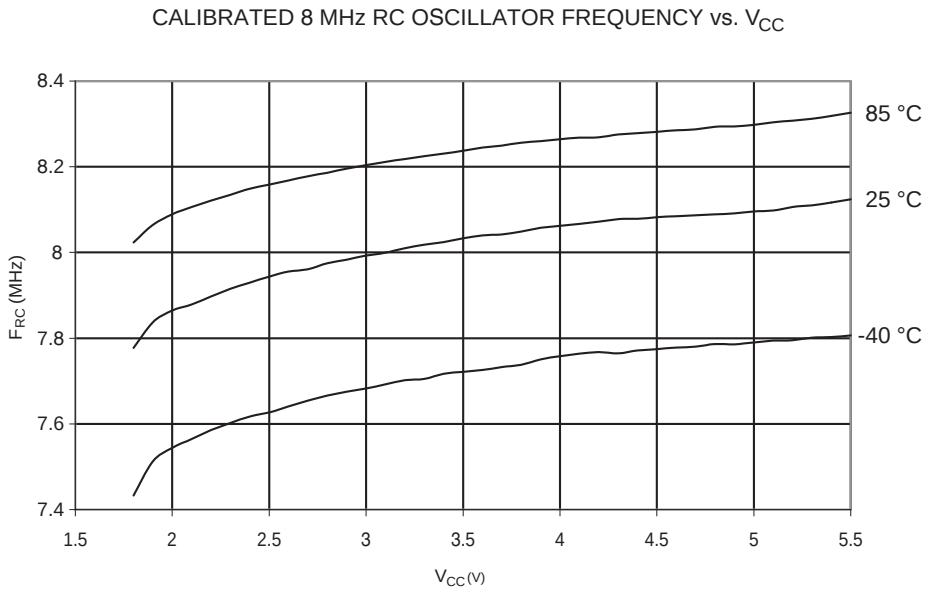


Figure 27-175. Calibrated 8 MHz RC Oscillator Frequency vs. Temperature.

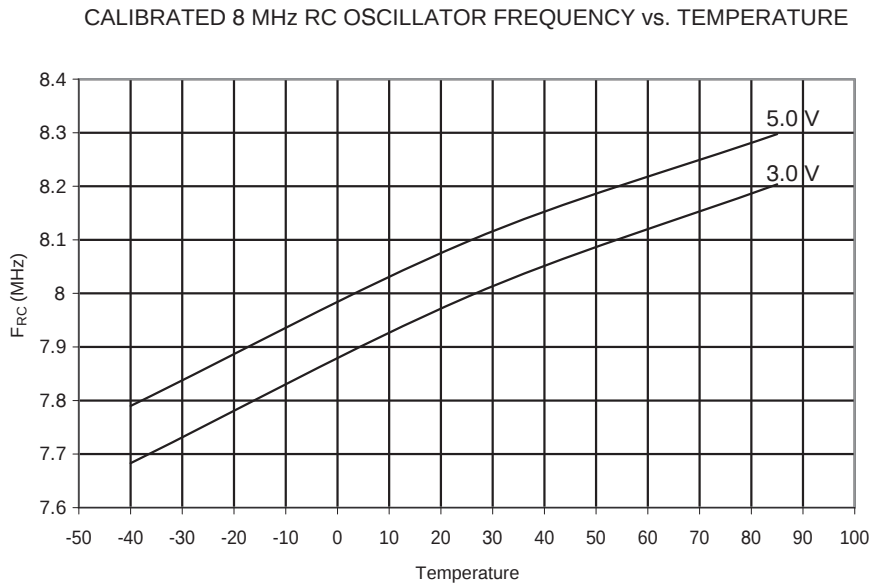
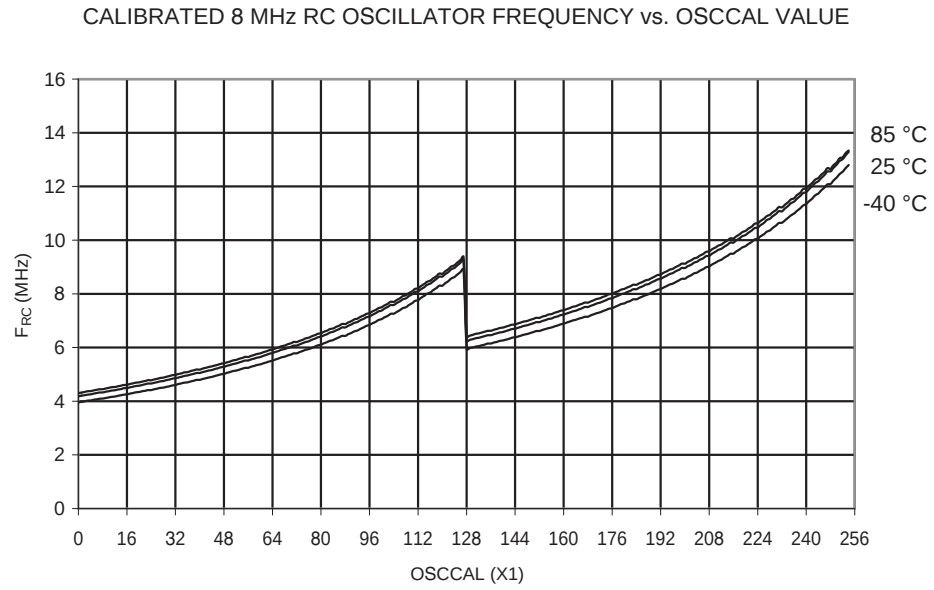


Figure 27-176. Calibrated 8 MHz RC Oscillator Frequency vs. OSCCAL Value.



27.4.12 Current Consumption of Peripheral Units

Figure 27-177. ADC Current vs. V_{CC} ($AREF = AV_{CC}$).

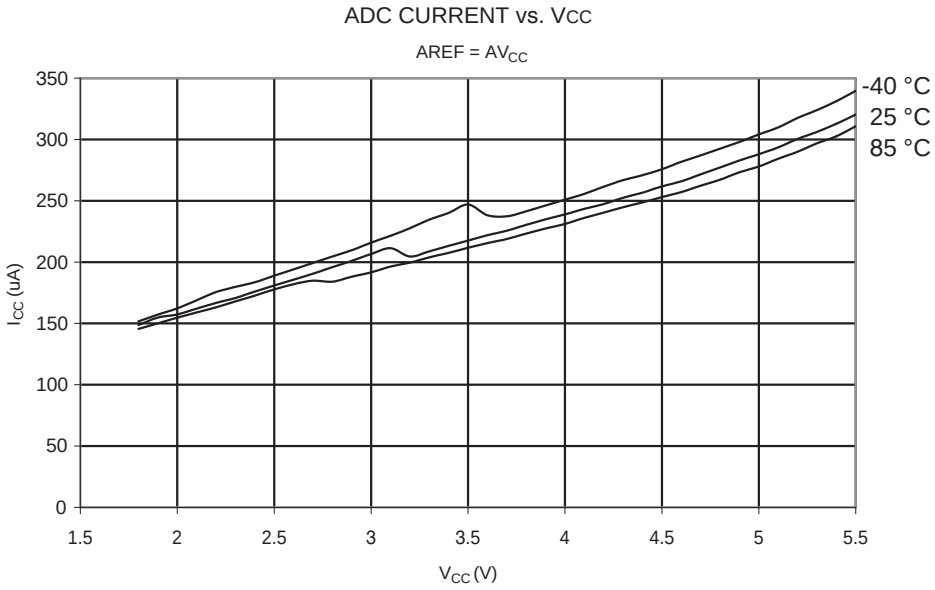


Figure 27-178. Analog Comparator Current vs. V_{CC}.

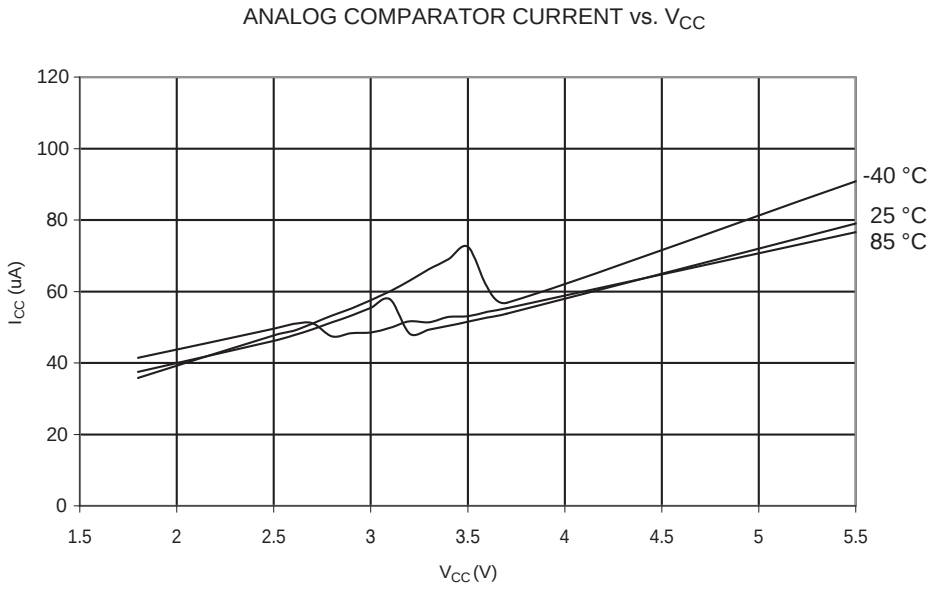


Figure 27-179. AREF External Reference Current vs. V_{CC}.

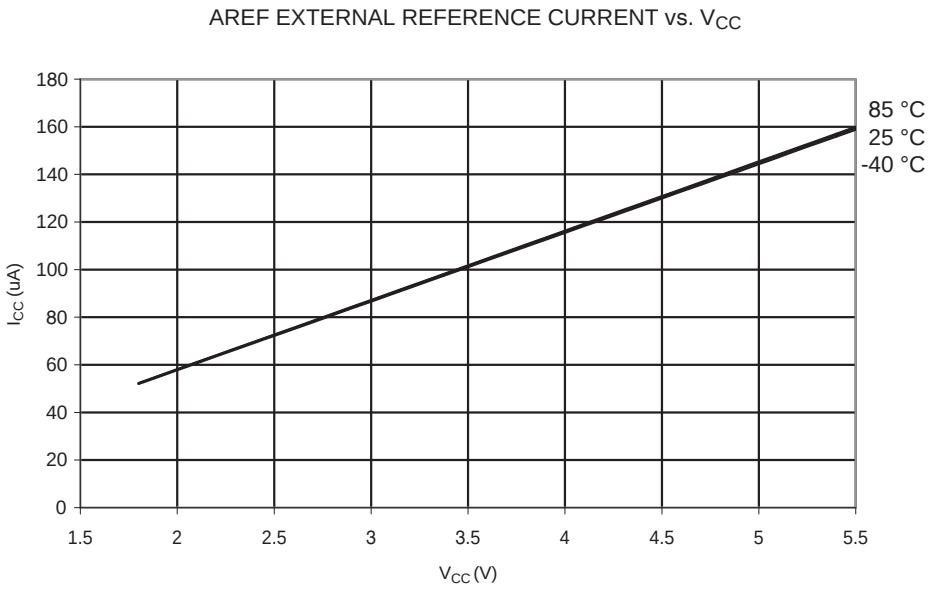


Figure 27-180. Brownout Detector Current vs. V_{CC}.

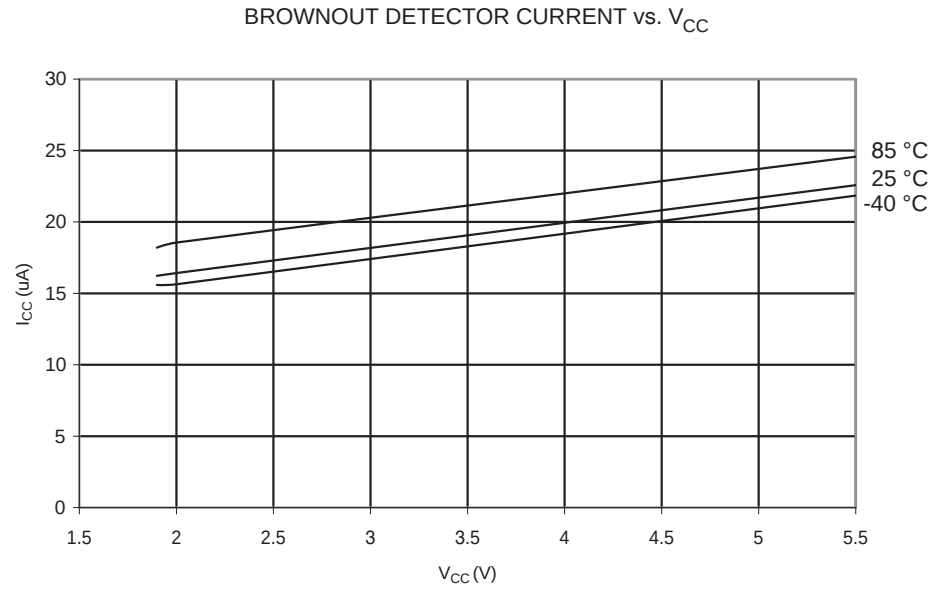
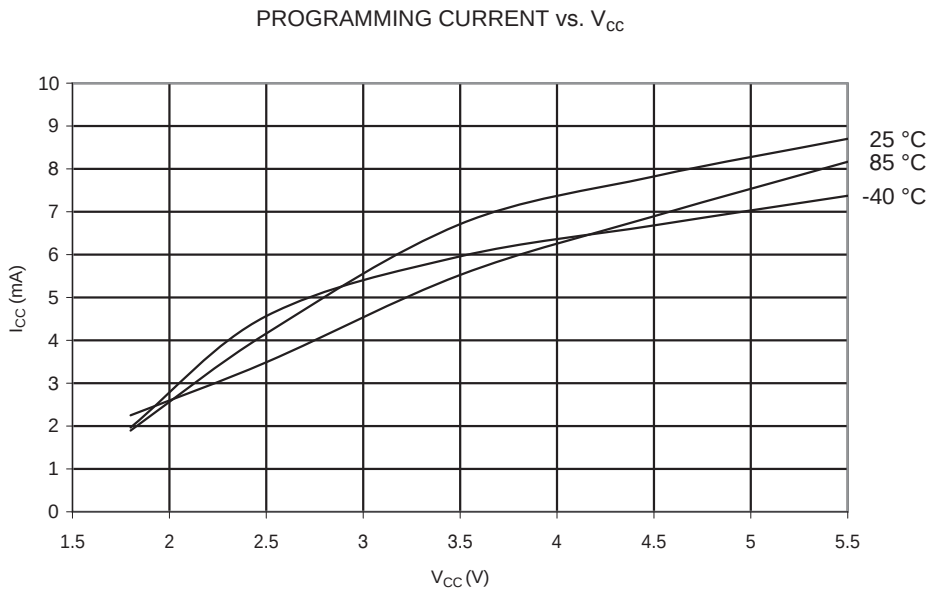


Figure 27-181. Programming Current vs. V_{CC}.



27.4.13 Current Consumption in Reset and Reset Pulsewidth

Figure 27-182. Reset Supply Current vs. Low Frequency (0.1 - 1.0 MHz).

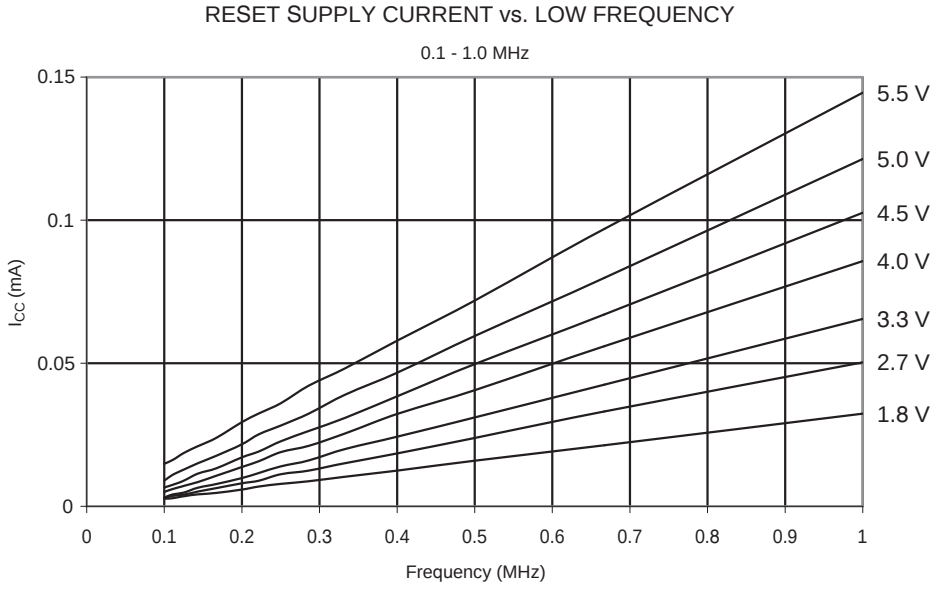


Figure 27-183. Reset Supply Current vs. Frequency (1 - 20 MHz).

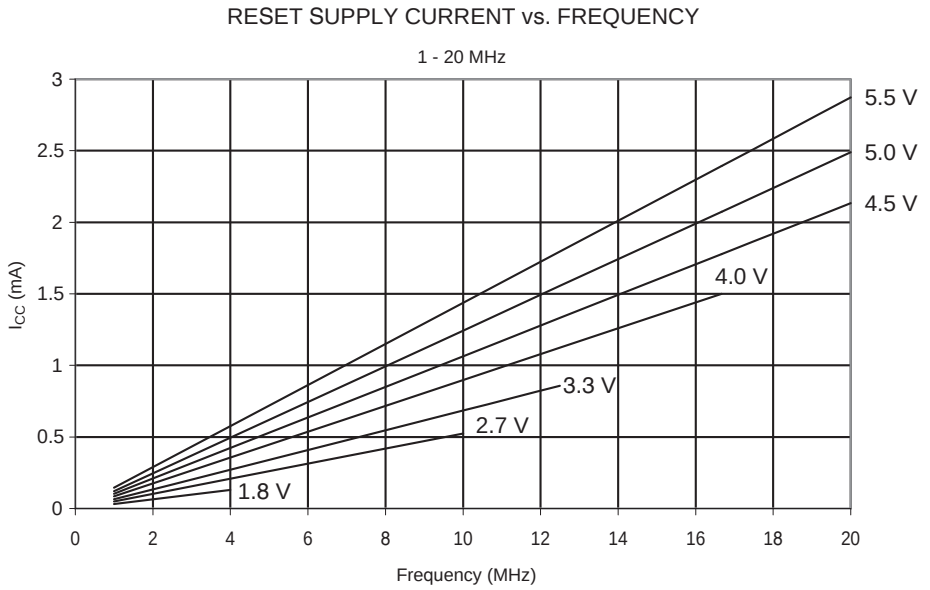
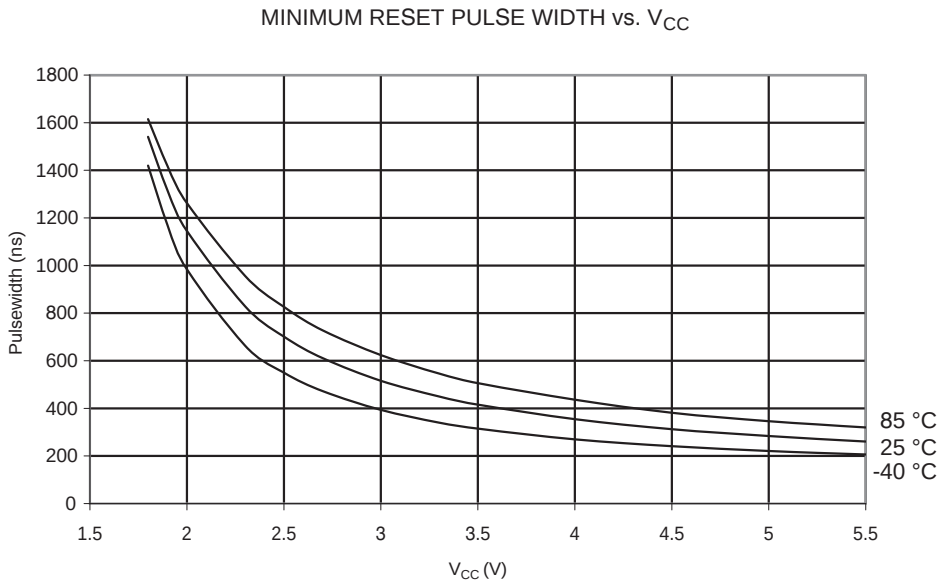


Figure 27-184. Minimum Reset Pulse width vs. V_{CC}.



28. Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0xFF)	Reserved	–	–	–	–	–	–	–	–	
(0xFE)	Reserved	–	–	–	–	–	–	–	–	
(0xFD)	Reserved	–	–	–	–	–	–	–	–	
(0xFC)	Reserved	–	–	–	–	–	–	–	–	
(0xFB)	Reserved	–	–	–	–	–	–	–	–	
(0xFA)	Reserved	–	–	–	–	–	–	–	–	
(0xF9)	Reserved	–	–	–	–	–	–	–	–	
(0xF8)	Reserved	–	–	–	–	–	–	–	–	
(0xF7)	Reserved	–	–	–	–	–	–	–	–	
(0xF6)	Reserved	–	–	–	–	–	–	–	–	
(0xF5)	Reserved	–	–	–	–	–	–	–	–	
(0xF4)	Reserved	–	–	–	–	–	–	–	–	
(0xF3)	Reserved	–	–	–	–	–	–	–	–	
(0xF2)	Reserved	–	–	–	–	–	–	–	–	
(0xF1)	Reserved	–	–	–	–	–	–	–	–	
(0xF0)	Reserved	–	–	–	–	–	–	–	–	
(0xEF)	Reserved	–	–	–	–	–	–	–	–	
(0xEE)	Reserved	–	–	–	–	–	–	–	–	
(0xED)	Reserved	–	–	–	–	–	–	–	–	
(0xEC)	Reserved	–	–	–	–	–	–	–	–	
(0xEB)	Reserved	–	–	–	–	–	–	–	–	
(0xEA)	Reserved	–	–	–	–	–	–	–	–	
(0xE9)	Reserved	–	–	–	–	–	–	–	–	
(0xE8)	Reserved	–	–	–	–	–	–	–	–	
(0xE7)	Reserved	–	–	–	–	–	–	–	–	
(0xE6)	Reserved	–	–	–	–	–	–	–	–	
(0xE5)	Reserved	–	–	–	–	–	–	–	–	
(0xE4)	Reserved	–	–	–	–	–	–	–	–	
(0xE3)	Reserved	–	–	–	–	–	–	–	–	
(0xE2)	Reserved	–	–	–	–	–	–	–	–	
(0xE1)	Reserved	–	–	–	–	–	–	–	–	
(0xE0)	Reserved	–	–	–	–	–	–	–	–	
(0xDF)	Reserved	–	–	–	–	–	–	–	–	
(0xDE)	Reserved	–	–	–	–	–	–	–	–	
(0xDD)	Reserved	–	–	–	–	–	–	–	–	
(0xDC)	Reserved	–	–	–	–	–	–	–	–	
(0xDB)	Reserved	–	–	–	–	–	–	–	–	
(0xDA)	Reserved	–	–	–	–	–	–	–	–	
(0xD9)	Reserved	–	–	–	–	–	–	–	–	
(0xD8)	Reserved	–	–	–	–	–	–	–	–	
(0xD7)	Reserved	–	–	–	–	–	–	–	–	
(0xD6)	Reserved	–	–	–	–	–	–	–	–	
(0xD5)	Reserved	–	–	–	–	–	–	–	–	
(0xD4)	Reserved	–	–	–	–	–	–	–	–	
(0xD3)	Reserved	–	–	–	–	–	–	–	–	
(0xD2)	Reserved	–	–	–	–	–	–	–	–	
(0xD1)	Reserved	–	–	–	–	–	–	–	–	
(0xD0)	Reserved	–	–	–	–	–	–	–	–	
(0xCF)	Reserved	–	–	–	–	–	–	–	–	
(0xCE)	Reserved	–	–	–	–	–	–	–	–	
(0xCD)	Reserved	–	–	–	–	–	–	–	–	
(0xCC)	Reserved	–	–	–	–	–	–	–	–	
(0xCB)	Reserved	–	–	–	–	–	–	–	–	
(0xCA)	Reserved	–	–	–	–	–	–	–	–	
(0xC9)	Reserved	–	–	–	–	–	–	–	–	
(0xC8)	Reserved	–	–	–	–	–	–	–	–	
(0xC7)	Reserved	–	–	–	–	–	–	–	–	
(0xC6)	UDR0	USART I/O Data Register								195
(0xC5)	UBRR0H					USART Baud Rate Register High				199
(0xC4)	UBRR0L	USART Baud Rate Register Low								199
(0xC3)	Reserved	–	–	–	–	–	–	–	–	
(0xC2)	UCSR0C	UMSEL01	UMSEL00	UPM01	UPM00	USBS0	UCSZ01 /UDORD0	UCSZ00 / UCPHA0	UCPOL0	197/212

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0xC1)	UCSR0B	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80	196
(0xC0)	UCSR0A	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0	195
(0xBF)	Reserved	–	–	–	–	–	–	–	–	
(0xBE)	Reserved	–	–	–	–	–	–	–	–	
(0xBD)	TWAMR	TWAM6	TWAM5	TWAM4	TWAM3	TWAM2	TWAM1	TWAM0	–	244
(0xBC)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE	241
(0xBB)	TWDR	2-wire Serial Interface Data Register								243
(0xBA)	TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	244
(0xB9)	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0	243
(0xB8)	TWBR	2-wire Serial Interface Bit Rate Register								241
(0xB7)	Reserved	–		–	–	–	–	–	–	
(0xB6)	ASSR	–	EXCLK	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB	164
(0xB5)	Reserved	–	–	–	–	–	–	–	–	
(0xB4)	OCR2B	Timer/Counter2 Output Compare Register B								162
(0xB3)	OCR2A	Timer/Counter2 Output Compare Register A								162
(0xB2)	TCNT2	Timer/Counter2 (8-bit)								162
(0xB1)	TCCR2B	FOC2A	FOC2B	–	–	WGM22	CS22	CS21	CS20	161
(0xB0)	TCCR2A	COM2A1	COM2A0	COM2B1	COM2B0	–	–	WGM21	WGM20	158
(0xAF)	Reserved	–	–	–	–	–	–	–	–	
(0xAE)	Reserved	–	–	–	–	–	–	–	–	
(0xAD)	Reserved	–	–	–	–	–	–	–	–	
(0xAC)	Reserved	–	–	–	–	–	–	–	–	
(0xAB)	Reserved	–	–	–	–	–	–	–	–	
(0xAA)	Reserved	–	–	–	–	–	–	–	–	
(0xA9)	Reserved	–	–	–	–	–	–	–	–	
(0xA8)	Reserved	–	–	–	–	–	–	–	–	
(0xA7)	Reserved	–	–	–	–	–	–	–	–	
(0xA6)	Reserved	–	–	–	–	–	–	–	–	
(0xA5)	Reserved	–	–	–	–	–	–	–	–	
(0xA4)	Reserved	–	–	–	–	–	–	–	–	
(0xA3)	Reserved	–	–	–	–	–	–	–	–	
(0xA2)	Reserved	–	–	–	–	–	–	–	–	
(0xA1)	Reserved	–	–	–	–	–	–	–	–	
(0xA0)	Reserved	–	–	–	–	–	–	–	–	
(0x9F)	Reserved	–	–	–	–	–	–	–	–	
(0x9E)	Reserved	–	–	–	–	–	–	–	–	
(0x9D)	Reserved	–	–	–	–	–	–	–	–	
(0x9C)	Reserved	–	–	–	–	–	–	–	–	
(0x9B)	Reserved	–	–	–	–	–	–	–	–	
(0x9A)	Reserved	–	–	–	–	–	–	–	–	
(0x99)	Reserved	–	–	–	–	–	–	–	–	
(0x98)	Reserved	–	–	–	–	–	–	–	–	
(0x97)	Reserved	–	–	–	–	–	–	–	–	
(0x96)	Reserved	–	–	–	–	–	–	–	–	
(0x95)	Reserved	–	–	–	–	–	–	–	–	
(0x94)	Reserved	–	–	–	–	–	–	–	–	
(0x93)	Reserved	–	–	–	–	–	–	–	–	
(0x92)	Reserved	–	–	–	–	–	–	–	–	
(0x91)	Reserved	–	–	–	–	–	–	–	–	
(0x90)	Reserved	–	–	–	–	–	–	–	–	
(0x8F)	Reserved	–	–	–	–	–	–	–	–	
(0x8E)	Reserved	–	–	–	–	–	–	–	–	
(0x8D)	Reserved	–	–	–	–	–	–	–	–	
(0x8C)	Reserved	–	–	–	–	–	–	–	–	
(0x8B)	OCR1BH	Timer/Counter1 - Output Compare Register B High Byte								138
(0x8A)	OCR1BL	Timer/Counter1 - Output Compare Register B Low Byte								138
(0x89)	OCR1AH	Timer/Counter1 - Output Compare Register A High Byte								138
(0x88)	OCR1AL	Timer/Counter1 - Output Compare Register A Low Byte								138
(0x87)	ICR1H	Timer/Counter1 - Input Capture Register High Byte								139
(0x86)	ICR1L	Timer/Counter1 - Input Capture Register Low Byte								139
(0x85)	TCNT1H	Timer/Counter1 - Counter Register High Byte								138
(0x84)	TCNT1L	Timer/Counter1 - Counter Register Low Byte								138
(0x83)	Reserved	–	–	–	–	–	–	–	–	
(0x82)	TCCR1C	FOC1A	FOC1B	–	–	–	–	–	–	137
(0x81)	TCCR1B	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	136
(0x80)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	134

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0x7F)	DIDR1	–	–	–	–	–	–	AIN1D	AIN0D	249
(0x7E)	DIDR0	–	–	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	266
(0x7D)	Reserved	–	–	–	–	–	–	–	–	
(0x7C)	ADMUX	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	262
(0x7B)	ADCSRB	–	ACME	–	–	–	ADTS2	ADTS1	ADTS0	265
(0x7A)	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	263
(0x79)	ADCH	ADC Data Register High byte								265
(0x78)	ADCL	ADC Data Register Low byte								265
(0x77)	Reserved	–	–	–	–	–	–	–	–	
(0x76)	Reserved	–	–	–	–	–	–	–	–	
(0x75)	Reserved	–	–	–	–	–	–	–	–	
(0x74)	Reserved	–	–	–	–	–	–	–	–	
(0x73)	Reserved	–	–	–	–	–	–	–	–	
(0x72)	Reserved	–	–	–	–	–	–	–	–	
(0x71)	Reserved	–	–	–	–	–	–	–	–	
(0x70)	TIMSK2	–	–	–	–	–	OCIE2B	OCIE2A	TOIE2	163
(0x6F)	TIMSK1	–	–	ICIE1	–	–	OCIE1B	OCIE1A	TOIE1	139
(0x6E)	TIMSK0	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	111
(0x6D)	PCMSK2	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16	74
(0x6C)	PCMSK1	–	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8	74
(0x6B)	PCMSK0	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	74
(0x6A)	Reserved	–	–	–	–	–	–	–	–	
(0x69)	EICRA	–	–	–	–	ISC11	ISC10	ISC01	ISC00	71
(0x68)	PCICR	–	–	–	–	–	PCIE2	PCIE1	PCIE0	
(0x67)	Reserved	–	–	–	–	–	–	–	–	
(0x66)	OSCCAL	Oscillator Calibration Register								37
(0x65)	Reserved	–	–	–	–	–	–	–	–	
(0x64)	PRR	PRTWI	PRTIM2	PRTIM0	–	PRTIM1	PRSPI	PRUSART0	PRADC	42
(0x63)	Reserved	–	–	–	–	–	–	–	–	
(0x62)	Reserved	–	–	–	–	–	–	–	–	
(0x61)	CLKPR	CLKPCE	–	–	–	CLKPS3	CLKPS2	CLKPS1	CLKPS0	37
(0x60)	WDTCSR	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0	54
0x3F (0x5F)	SREG	I	T	H	S	V	N	Z	C	9
0x3E (0x5E)	SPH	–	–	–	–	–	(SP10) [5.]	SP9	SP8	12
0x3D (0x5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	12
0x3C (0x5C)	Reserved	–	–	–	–	–	–	–	–	
0x3B (0x5B)	Reserved	–	–	–	–	–	–	–	–	
0x3A (0x5A)	Reserved	–	–	–	–	–	–	–	–	
0x39 (0x59)	Reserved	–	–	–	–	–	–	–	–	
0x38 (0x58)	Reserved	–	–	–	–	–	–	–	–	
0x37 (0x57)	SPMCSR	SPMIE	(RWWSB)[5.]	–	(RWWSRE)[5.]	BLBSET	PGWRT	PGERS	SELFPRGEN	292
0x36 (0x56)	Reserved	–	–	–	–	–	–	–	–	
0x35 (0x55)	MCUCR	–	BODS	BODSE	PUD	–	–	IVSEL	IVCE	44/68/92
0x34 (0x54)	MCUSR	–	–	–	–	WDRF	BORF	EXTRF	PORF	54
0x33 (0x53)	SMCR	–	–	–	–	SM2	SM1	SM0	SE	40
0x32 (0x52)	Reserved	–	–	–	–	–	–	–	–	
0x31 (0x51)	Reserved	–	–	–	–	–	–	–	–	
0x30 (0x50)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	247
0x2F (0x4F)	Reserved	–	–	–	–	–	–	–	–	
0x2E (0x4E)	SPDR	SPI Data Register								175
0x2D (0x4D)	SPSR	SPIF	WCOL	–	–	–	–	–	SPI2X	174
0x2C (0x4C)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	173
0x2B (0x4B)	GPIOR2	General Purpose I/O Register 2								25
0x2A (0x4A)	GPIOR1	General Purpose I/O Register 1								25
0x29 (0x49)	Reserved	–	–	–	–	–	–	–	–	
0x28 (0x48)	OCR0B	Timer/Counter0 Output Compare Register B								
0x27 (0x47)	OCR0A	Timer/Counter0 Output Compare Register A								
0x26 (0x46)	TCNT0	Timer/Counter0 (8-bit)								
0x25 (0x45)	TCCR0B	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	
0x24 (0x44)	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	
0x23 (0x43)	GTCCR	TSM	–	–	–	–	–	PSRASY	PSRSYNC	143/165
0x22 (0x42)	EEARH	(EEPROM Address Register High Byte) [5.]								21
0x21 (0x41)	EEARL	EEPROM Address Register Low Byte								21
0x20 (0x40)	EEDR	EEPROM Data Register								21
0x1F (0x3F)	EECR	–	–	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE	21
0x1E (0x3E)	GPIOR0	General Purpose I/O Register 0								25

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
0x1D (0x3D)	EIMSK	–	–	–	–	–	–	INT1	INT0	72
0x1C (0x3C)	EIFR	–	–	–	–	–	–	INTF1	INTF0	72
0x1B (0x3B)	PCIFR	–	–	–	–	–	PCIF2	PCIF1	PCIF0	
0x1A (0x3A)	Reserved	–	–	–	–	–	–	–	–	
0x19 (0x39)	Reserved	–	–	–	–	–	–	–	–	
0x18 (0x38)	Reserved	–	–	–	–	–	–	–	–	
0x17 (0x37)	TIFR2	–	–	–	–	–	OCF2B	OCF2A	TOV2	163
0x16 (0x36)	TIFR1	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1	140
0x15 (0x35)	TIFR0	–	–	–	–	–	OCF0B	OCF0A	TOV0	
0x14 (0x34)	Reserved	–	–	–	–	–	–	–	–	
0x13 (0x33)	Reserved	–	–	–	–	–	–	–	–	
0x12 (0x32)	Reserved	–	–	–	–	–	–	–	–	
0x11 (0x31)	Reserved	–	–	–	–	–	–	–	–	
0x10 (0x30)	Reserved	–	–	–	–	–	–	–	–	
0x0F (0x2F)	Reserved	–	–	–	–	–	–	–	–	
0x0E (0x2E)	Reserved	–	–	–	–	–	–	–	–	
0x0D (0x2D)	Reserved	–	–	–	–	–	–	–	–	
0x0C (0x2C)	Reserved	–	–	–	–	–	–	–	–	
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	93
0x0A (0x2A)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	93
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	93
0x08 (0x28)	PORTC	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	92
0x07 (0x27)	DDRC	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	92
0x06 (0x26)	PINC	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	92
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	92
0x04 (0x24)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	92
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	92
0x02 (0x22)	Reserved	–	–	–	–	–	–	–	–	
0x01 (0x21)	Reserved	–	–	–	–	–	–	–	–	
0x0 (0x20)	Reserved	–	–	–	–	–	–	–	–	

Note:
1. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.
2. I/O Registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions.
3. Some of the Status Flags are cleared by writing a logical one to them. Note that, unlike most other AVRs, the CBI and SBI instructions will only operate on the specified bit, and can therefore be used on registers containing such Status Flags. The CBI and SBI instructions work with registers 0x00 to 0x1F only.
4. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these addresses. The ATmega48P/88P/168P/328P is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.
5. Only valid for ATmega88P/168P.

29. Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS					
ADD	Rd, Rr	Add two Registers	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry two Registers	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl,K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	Rd ← Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry two Registers	Rd ← Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	Rd ← Rd - K - C	Z,C,N,V,H	1
SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	Rd ← Rd • Rr	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd • K	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	One's Complement	Rd ← 0xFF – Rd	Z,C,N,V	1
NEG	Rd	Two's Complement	Rd ← 0x00 – Rd	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • (0xFF - K)	Z,N,V	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd ← Rd – 1	Z,N,V	1
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z,N,V	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	Set Register	Rd ← 0xFF	None	1
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC ← Z	None	2
JMP[1]	k	Direct Jump	PC ← k	None	3
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None	3
ICALL		Indirect Call to (Z)	PC ← Z	None	3
CALL[1]	k	Direct Subroutine Call	PC ← k	None	4
RET		Subroutine Return	PC ← STACK	None	4
RETI		Interrupt Return	PC ← STACK	I	4
CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2/3
CP	Rd,Rr	Compare	Rd – Rr	Z, N,V,C,H	1
CPC	Rd,Rr	Compare with Carry	Rd – Rr – C	Z, N,V,C,H	1
CPI	Rd,K	Compare Register with Immediate	Rd – K	Z, N,V,C,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC←PC+k + 1	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC←PC+k + 1	None	1/2
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if (N ⊕ V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None	1/2

Mnemonics	Operands	Description	Operation	Flags	#Clocks
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
BIT AND BIT-TEST INSTRUCTIONS					
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	Rotate Left Through Carry	Rd(0)←C,Rd(n+1)← Rd(n),C←Rd(7)	Z,C,N,V	1
ROR	Rd	Rotate Right Through Carry	Rd(7)←C,Rd(n)← Rd(n+1),C←Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0..6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3..0)←Rd(7..4),Rd(7..4)←Rd(3..0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Twos Complement Overflow.	V ← 1	V	1
CLV		Clear Twos Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half Carry Flag in SREG	H ← 1	H	1
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, - X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, - Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	- X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	- Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q,Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2

Mnemonics	Operands	Description	Operation	Flags	#Clocks
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2
MCU CONTROL INSTRUCTIONS					
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	1
WDR		Watchdog Reset	(see specific descr. for WDR/timer)	None	1
BREAK		Break	For On-chip Debug Only	None	N/A

Note: 1. These instructions are only available in ATmega168P and ATmega328P.

30. Ordering Information

30.1 ATmega48P

Speed (MHz)	Power Supply	Ordering Code[2]	Package[1]	Operational Range
10[3]	1.8 - 5.5	ATmega48PV-10AU ATmega48PV-10MMU ATmega48PV-10MU ATmega48PV-10PU	32A 28M1 32M1-A 28P3	Industrial (-40°C to 85°C)
20[3]	2.7 - 5.5	ATmega48P-20AU ATmega48P-20MMU ATmega48P-20MU ATmega48P-20PU	32A 28M1 32M1-A 28P3	Industrial (-40°C to 85°C)

Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive).Also Halide free and fully Green.
3. See Figure 26-1 on page 317 and Figure 26-2 on page 317.

Package Type	
32A	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
28M1	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
28P3	28-lead, 0.300” Wide, Plastic Dual Inline Package (PDIP)

30.2 ATmega88P

Speed (MHz)	Power Supply	Ordering Code[2]	Package[1]	Operational Range
10[3]	1.8 - 5.5	ATmega88PV-10AU ATmega88PV-10MU ATmega88PV-10PU	32A 32M1-A 28P3	Industrial (-40°C to 85°C)
20[3]	2.7 - 5.5	ATmega88P-20AU ATmega88P-20MU ATmega88P-20PU	32A 32M1-A 28P3	Industrial (-40°C to 85°C)

Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive).Also Halide free and fully Green.
3. See Figure 26-1 on page 317 and Figure 26-2 on page 317.

Package Type	
32A	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
28P3	28-lead, 0.300” Wide, Plastic Dual Inline Package (PDIP)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)

30.3 ATmega168P

Speed (MHz)[3]	Power Supply	Ordering Code[2]	Package[1]	Operational Range
10	1.8 - 5.5	ATmega168PV-10AU ATmega168PV-10MU ATmega168PV-10PU	32A 32M1-A 28P3	Industrial (-40°C to 85°C)
20	2.7 - 5.5	ATmega168P-20AU ATmega168P-20MU ATmega168P-20PU	32A 32M1-A 28P3	Industrial (-40°C to 85°C)

Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive).Also Halide free and fully Green.
3. See Figure 26-1 on page 317 and Figure 26-2 on page 317.

Package Type	
32A	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
28P3	28-lead, 0.300” Wide, Plastic Dual Inline Package (PDIP)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)

30.4 ATmega328P

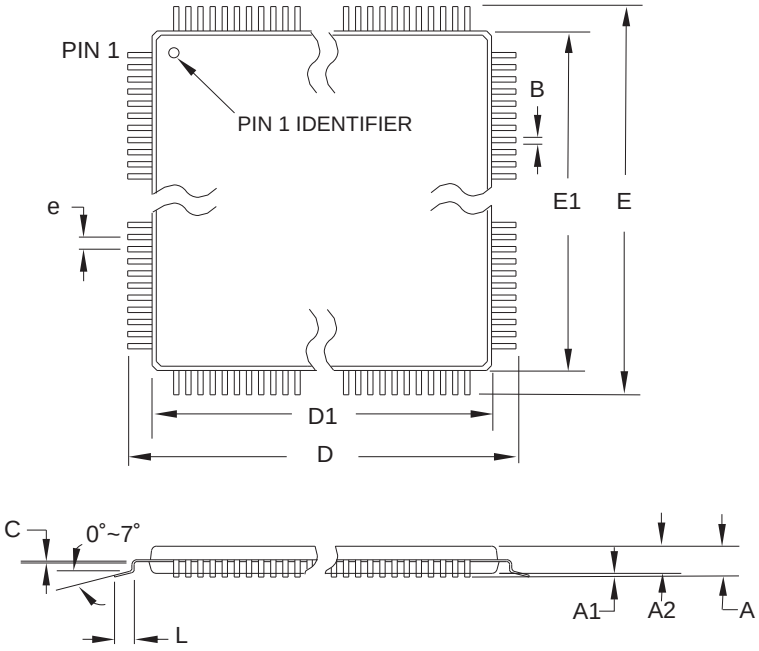
Speed (MHz)	Power Supply	Ordering Code[2]	Package[1]	Operational Range
20[3]	1.8 - 5.5	ATmega328P- AU ATmega328P- MU ATmega328P- PU	32A 32M1-A 28P3	Industrial (-40°C to 85°C)

Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive).Also Halide free and fully Green.
3. See Figure 26-3 on page 318.

Package Type	
32A	32-lead, Thin (1.0 mm) Plastic Quad Flat Package (TQFP)
28P3	28-lead, 0.300” Wide, Plastic Dual Inline Package (PDIP)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50 mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)

31. Packaging Information

31.1 32A



COMMON DIMENSIONS
(Unit of Measure = mm)

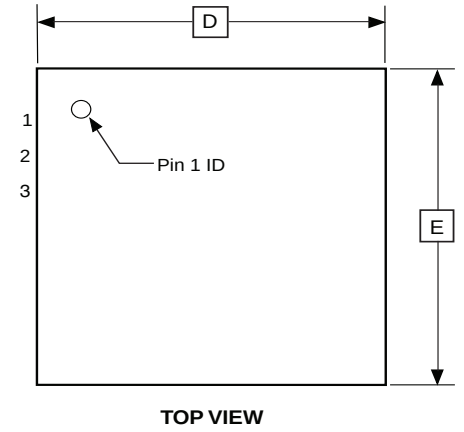
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	8.75	9.00	9.25	
D1	6.90	7.00	7.10	Note 2
E	8.75	9.00	9.25	
E1	6.90	7.00	7.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

Notes:
1. This package conforms to JEDEC reference MS-026, Variation ABA.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10 mm maximum.

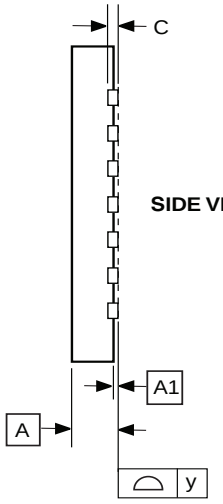
10/5/2001

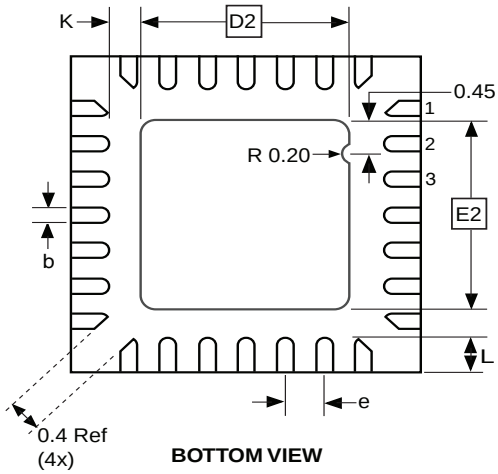
	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**32A,** 32-lead, 7 x 7 mm Body Size, 1.0 mm Body Thickness, 0.8 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)	32A	B

31.2 28M1



TOP VIEW





BOTTOM VIEW

Note: The terminal #1 ID is a Laser-marked Feature.

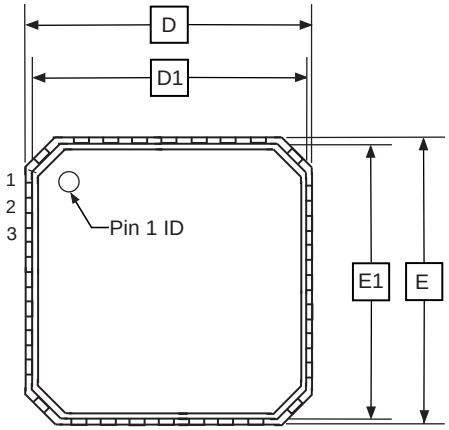
COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.90	1.00	
A1	0.00	0.02	0.05	
b	0.17	0.22	0.27	
C		0.20 REF		
D	3.95	4.00	4.05	
D2	2.35	2.40	2.45	
E	3.95	4.00	4.05	
E2	2.35	2.40	2.45	
e		0.45		
L	0.35	0.40	0.45	
y	0.00	–	0.08	
K	0.20	–	–	

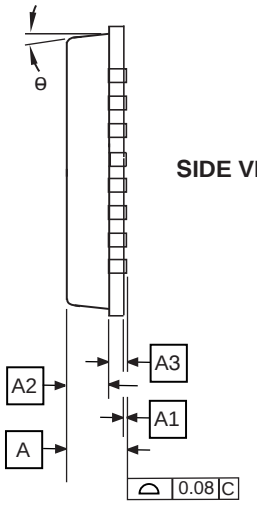
10/24/08

	TITLE	GPC	DRAWING NO.	REV.
Package Drawing Contact: packagedrawings@atmel.com	**28M1,** 28-pad, 4 x 4 x 1.0 mm Body, Lead Pitch 0.45 mm, 2.4 x 2.4 mm Exposed Pad, Thermally Enhanced Plastic Very Thin Quad Flat No Lead Package (VQFN)	ZBV	28M1	B

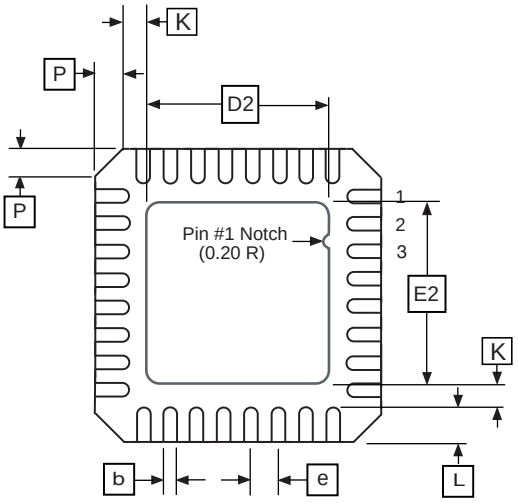
31.3 32M1-A



TOP VIEW



SIDE VIEW



BOTTOM VIEW

COMMON DIMENSIONS
(Unit of Measure = mm)

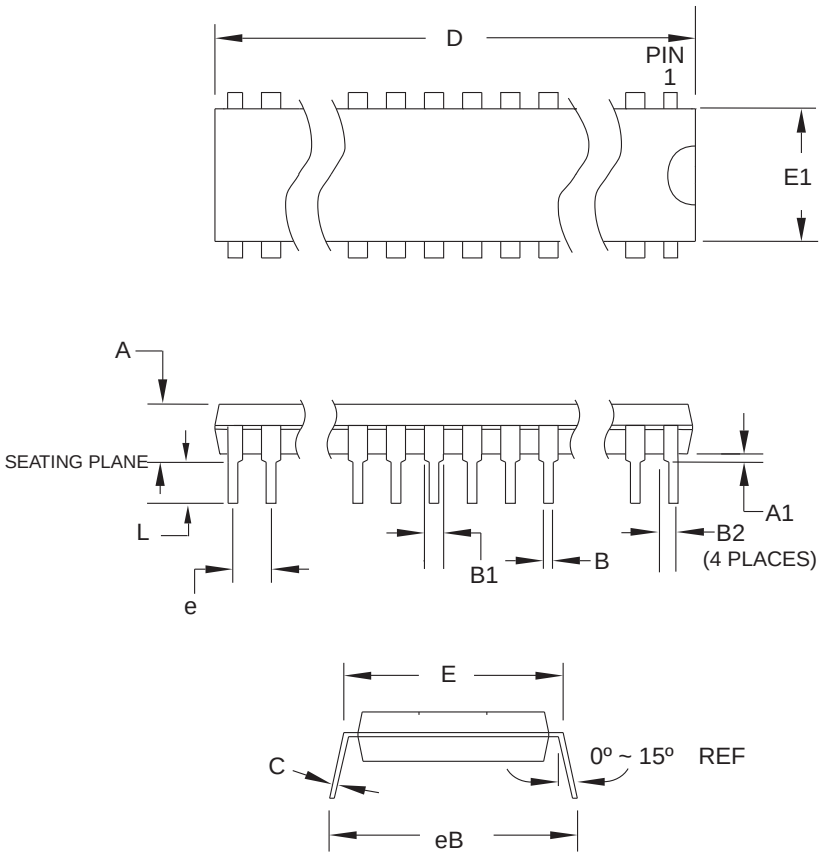
SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.90	1.00	
A1	–	0.02	0.05	
A2	–	0.65	1.00	
A3		0.20 REF		
b	0.18	0.23	0.30	
D	4.90	5.00	5.10	
D1	4.70	4.75	4.80	
D2	2.95	3.10	3.25	
E	4.90	5.00	5.10	
E1	4.70	4.75	4.80	
E2	2.95	3.10	3.25	
e		0.50 BSC		
L	0.30	0.40	0.50	
P	–	–	0.60	
θ	–	–	12^{o}	
K	0.20	–	–	

Note: JEDEC Standard MO-220, Fig. 2 (Anvil Singulation), VHHD-2.

5/25/06

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**32M1-A**, 32-pad, 5 x 5 x 1.0 mm Body, Lead Pitch 0.50 mm, 3.10 mm Exposed Pad, Micro Lead Frame Package (MLF)	32M1-A	E

31.4 28P3



COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.5724	
A1	0.508	–	–	
D	34.544	–	34.798	Note 1
E	7.620	–	8.255	
E1	7.112	–	7.493	Note 1
B	0.381	–	0.533	
B1	1.143	–	1.397	
B2	0.762	–	1.143	
L	3.175	–	3.429	
C	0.203	–	0.356	
eB	–	–	10.160	
e	2.540 TYP			

Note: 1. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

09/28/01

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**28P3**, 28-lead (0.300"/7.62 mm Wide) Plastic Dual Inline Package (PDIP)	28P3	B

32. Errata

32.1 Errata ATmega48P

The revision letter in this section refers to the revision of the ATmega48P device.

32.1.1 Rev. C

No known errata.

32.1.2 Rev. B

No known errata.

32.1.3 Rev. A

Not Sampled.

32.2 Errata ATmega88P

The revision letter in this section refers to the revision of the ATmega88P device.

32.2.1 Rev. C

Not sampled.

32.2.2 Rev. B

No known errata.

32.2.3 Rev. A

No known errata.

32.3 Errata ATmega168P

The revision letter in this section refers to the revision of the ATmega168P device.

32.3.1 Rev B

No known errata.

32.3.2 Rev A

No known errata.

32.4 Errata ATmega328P

The revision letter in this section refers to the revision of the ATmega328P device.

32.4.1 Rev C

No known errata.

32.4.2 Rev B

- **Unstable 32 kHz Oscillator**

1. **Unstable 32 kHz Oscillator**

   The 32 kHz oscillator does not work as system clock.

   The 32 kHz oscillator used as asynchronous timer is inaccurate.

Problem Fix/ Workaround

None

32.4.3 Rev A

- **Unstable 32 kHz Oscillator**

1. Unstable 32 kHz Oscillator

The 32 kHz oscillator does not work as system clock.

The 32 kHz oscillator used as asynchronous timer is inaccurate.

Problem Fix/ Workaround

None

33. Datasheet Revision History

Please note that the referring page numbers in this section are referred to this document. The referring revision in this section are referring to the document revision.

33.1 Rev. 2545I-02/09

1. Removed “preliminary” from ATmega48P/88P/168P.

33.2 Rev. 2545H-02/09

1. Added Power-save Maximum values and footnote to ”ATmega48P DC Characteristics” on page 314.
2. Added Power-save Maximum values and footnote to ”ATmega88P DC Characteristics” on page 315.
3. Added Power-save Maximum values and footnote to ”ATmega168P DC Characteristics” on page 315.
4. Added Power-save Maximum values and footnote to ”ATmega328P DC Characteristics” on page 316.
5. Added errata for revision A, ”Errata ATmega328P” on page 440.

33.3 Rev. 2545G-01/09

1 ATmega48P/88P not recommended for new designs.
2. Updated the footnote Note1 of the Table 6-3 on page 29.
3. Updated the Table 6-5 on page 30 by removing a footnote Note1.
4. Updated the Table 6-10 on page 33 by removing a footnote Note1.
5. Updated the footnote Note1 of the Table 6-12 on page 34.
6. Updated the footnote Note2 of the ”ATmega48P DC Characteristics” on page 314 and removed TBD from the table.
7. Updated the footnote Note2 of the ”ATmega88P DC Characteristics” on page 315 and removed TBD from the table.
8. Updated the footnote Note2 of the ”ATmega168P DC Characteristics” on page 315 and removed TBD from the table.
9. Updated the footnote Note2 of the ”ATmega328P DC Characteristics” on page 316 and removed TBD from the table.
10. Updated the footnote Note1 of the Table 26-4 on page 320.
11. Replaced the Figure 27-69 on page 365 by a correct one.
12. Replaced the Figure 27-173 on page 419 by a correct one.
13. Updated ”Errata” on page 440.
14. Updated ”MCUCR – MCU Control Register” on page 44.
15. Updated ”TCCR2B – Timer/Counter Control Register B” on page 161.

33.4 Rev. 2545F-08/08

1. Updated ”ATmega328P Typical Characteristics” on page 401 with Power-save numbers.
2. Added ATmega328P ”Standby Supply Current” on page 408.

33.5 Rev. 2545E-08/08

1. Updated description of ”Stack Pointer” on page 12.
2. Updated description of use of external capacitors in ”Low Frequency Crystal Oscillator” on page 32.
3. Updated Table 6-9 in ”Low Frequency Crystal Oscillator” on page 32.
4. Added note to ”Address Match Unit” on page 222.
5. Added section ”Reading the Signature Row from Software” on page 285.
6. Updated ”Program And Data Memory Lock Bits” on page 294 to include ATmega328P in the description.
7. Added ”ATmega328P DC Characteristics” on page 316.
8. Updated ”Speed Grades” on page 316 for ATmega328P.
9. Removed note 6 and 7 from the table ”2-wire Serial Interface Characteristics” on page 323.
10. Added figure ”Minimum Reset Pulse width vs. VCC.” on page 352 for ATmega48P.
11. Added figure ”Minimum Reset Pulse width vs. VCC.” on page 376 for ATmega88P.
12. Added figure ”Minimum Reset Pulse width vs. VCC.” on page 400 for ATmega168P.
13. Added ”ATmega328P Typical Characteristics” on page 401.
14. Updated Ordering Information for ”ATmega328P” on page 435.

33.6 Rev. 2545D-03/08

1. Updated figures in ”Speed Grades” on page 316.
2. Updated note in Table 26-4 in ”System and Reset Characteristics” on page 320.
3. Ordering codes for ”ATmega328P” on page 435 updated.
   - ATmega328P is offered in 20 MHz option only.
4. Added Errata for ATmega328P rev. B, ”Errata ATmega328P” on page 440.

33.7 Rev. 2545C-01/08

1. Power-save Maximum values removed form ”ATmega48P DC Characteristics” on page 314, ”ATmega88P DC Characteristics” on page 315, and ”ATmega168P DC Characteristics” on page 315.

33.8 Rev. 2545B-01/08

1. Updated ”Features” on page 1.
2. Added ”Data Retention” on page 7.
3. Updated Table 6-2 on page 28.
4. Removed “Low-frequency Crystal Oscillator Internal Load Capacitance“ table from”Low Frequency Crystal Oscillator” on page 32.
5. Removed JTD bit from ”MCUCR – MCU Control Register” on page 44.
6. Updated typical and general program setup for Reset and Interrupt Vector Addresses in ”Interrupt Vectors in ATmega168P” on page 62 and ”Interrupt Vectors in ATmega328P” on page 65.
7. Updated Interrupt Vectors Start Address in Table 9-5 on page 63 and Table 9-7 on page 66.
8. Updated ”Temperature Measurement” on page 261.
9. Updated ATmega328P ”Fuse Bits” on page 295.
10. Removed V_{OL3}/V_{OH3} rows from ”DC Characteristics” on page 313.
11. Updated condition for V_{OL} in ”DC Characteristics” on page 313.
    Updated max value for V_{IL2} in ”DC Characteristics” on page 313.
12. Added ”ATmega48P DC Characteristics” on page 314, ”ATmega88P DC Characteristics” on page 315, and ”ATmega168P DC Characteristics” on page 315.
13. Updated ”System and Reset Characteristics” on page 320.
14. Added ”ATmega48P Typical Characteristics” on page 329, ”ATmega88P Typical Characteristics” on page 353, and ”ATmega168P Typical Characteristics” on page 377.
15. Updated note in ”Instruction Set Summary” on page 429.

33.9 Rev. 2545A-07/07

1. Initial revision.

Table of Contents

Features *1*

1 Pin Configurations *2*

1.1 Pin Descriptions 3

2 Overview *4*

2.1 Block Diagram 5

2.2 Comparison Between ATmega48P, ATmega88P, ATmega168P, and ATmega328P 6

3 About *7*

3.1 Disclaimer 7

3.2 Resources 7

3.3 Data Retention 7

3.4 Code Examples 7

4 AVR CPU Core *8*

4.1 Overview 8

4.2 ALU – Arithmetic Logic Unit 9

4.3 Status Register 9

4.4 General Purpose Register File 11

4.5 Stack Pointer 12

4.6 Instruction Execution Timing 13

4.7 Reset and Interrupt Handling 14

5 AVR Memories *16*

5.1 Overview 16

5.2 In-System Reprogrammable Flash Program Memory 16

5.3 SRAM Data Memory 18

5.4 EEPROM Data Memory 19

5.5 I/O Memory 20

5.6 Register Description 21

6 System Clock and Clock Options *26*

6.1 Clock Systems and their Distribution 26

6.2 Clock Sources 27

6.3 Low Power Crystal Oscillator 28

6.4 Full Swing Crystal Oscillator 30

6.5 Low Frequency Crystal Oscillator 32

6.6Calibrated Internal RC Oscillator 33
6.7128 kHz Internal Oscillator 33
6.8External Clock 34
6.9Clock Output Buffer 35
6.10Timer/Counter Oscillator 35
6.11System Clock Prescaler 35
6.12Register Description 37

7 Power Management and Sleep Modes 39

7.1Sleep Modes 39
7.2BOD Disable 40
7.3Idle Mode 40
7.4ADC Noise Reduction Mode 40
7.5Power-down Mode 41
7.6Power-save Mode 41
7.7Standby Mode 41
7.8Extended Standby Mode 41
7.9Power Reduction Register 42
7.10Minimizing Power Consumption 42
7.11Register Description 44

8 System Control and Reset 46

8.1Resetting the AVR 46
8.2Reset Sources 46
8.3Power-on Reset 47
8.4External Reset 48
8.5Brown-out Detection 48
8.6Watchdog System Reset 49
8.7Internal Voltage Reference 49
8.8Watchdog Timer 50
8.9Register Description 54

9 Interrupts 57

9.1Interrupt Vectors in ATmega48P 57
9.2Interrupt Vectors in ATmega88P 59
9.3Interrupt Vectors in ATmega168P 62
9.4Interrupt Vectors in ATmega328P 65
9.5Register Description 68

10 External Interrupts 70

10.1Pin Change Interrupt Timing 70

10.2Register Description 71

11 I/O-Ports 75

11.1Overview 75

11.2Ports as General Digital I/O 76

11.3Alternate Port Functions 80

11.4Register Description 92

12 8-bit Timer/Counter0 with PWM 94

12.1Features 94

12.2Overview 94

12.3Timer/Counter Clock Sources 96

12.4Counter Unit 96

12.5Output Compare Unit 97

12.6Compare Match Output Unit 98

12.7Modes of Operation 99

12.8Timer/Counter Timing Diagrams 104

12.9Register Description 106

13 16-bit Timer/Counter1 with PWM 113

13.1Features 113

13.2Overview 113

13.3Accessing 16-bit Registers 115

13.4Timer/Counter Clock Sources 118

13.5Counter Unit 119

13.6Input Capture Unit 120

13.7Output Compare Units 122

13.8Compare Match Output Unit 124

13.9Modes of Operation 125

13.10Timer/Counter Timing Diagrams 132

13.11Register Description 134

14 Timer/Counter0 and Timer/Counter1 Prescalers 141

14.1Internal Clock Source 141

14.2Prescaler Reset 141

14.3External Clock Source 141

14.4Register Description 143

15 8-bit Timer/Counter2 with PWM and Asynchronous Operation 144

15.1Features 144

15.2Overview 144

15.3Timer/Counter Clock Sources 145

15.4Counter Unit 145

15.5Output Compare Unit 146

15.6Compare Match Output Unit 148

15.7Modes of Operation 149

15.8Timer/Counter Timing Diagrams 153

15.9Asynchronous Operation of Timer/Counter2 155

15.10Timer/Counter Prescaler 156

15.11Register Description 158

16 SPI – Serial Peripheral Interface 166

16.1Features 166

16.2Overview 166

16.3$\overline{SS}$ Pin Functionality 171

16.4Data Modes 171

16.5Register Description 173

17 USART0 176

17.1Features 176

17.2Overview 176

17.3Clock Generation 177

17.4Frame Formats 180

17.5USART Initialization 182

17.6Data Transmission – The USART Transmitter 183

17.7Data Reception – The USART Receiver 186

17.8Asynchronous Data Reception 190

17.9Multi-processor Communication Mode 193

17.10Register Description 195

17.11Examples of Baud Rate Setting 199

18 USART in SPI Mode 204

18.1Features 204

18.2Overview 204

18.3Clock Generation 204

18.4SPI Data Modes and Timing 205

18.5Frame Formats 206
18.6Data Transfer 208
18.7AVR USART MSPIM vs. AVR SPI 210
18.8Register Description 211

19 2-wire Serial Interface 214

19.1Features 214
19.22-wire Serial Interface Bus Definition 214
19.3Data Transfer and Frame Format 216
19.4Multi-master Bus Systems, Arbitration and Synchronization 218
19.5Overview of the TWI Module 221
19.6Using the TWI 223
19.7Transmission Modes 227
19.8Multi-master Systems and Arbitration 240
19.9Register Description 241

20 Analog Comparator 246

20.1Overview 246
20.2Analog Comparator Multiplexed Input 246
20.3Register Description 247

21 Analog-to-Digital Converter 250

21.1Features 250
21.2Overview 250
21.3Starting a Conversion 252
21.4Prescaling and Conversion Timing 253
21.5Changing Channel or Reference Selection 255
21.6ADC Noise Canceler 256
21.7ADC Conversion Result 261
21.8Temperature Measurement 261
21.9Register Description 262

22 debugWIRE On-chip Debug System 267

22.1Features 267
22.2Overview 267
22.3Physical Interface 267
22.4Software Break Points 268
22.5Limitations of debugWIRE 268
22.6Register Description 268

23 Self-Programming the Flash, ATmega48P 269

23.1Overview 269

23.2Addressing the Flash During Self-Programming 270

23.3Register Description' 275

24 Boot Loader Support – Read-While-Write Self-Programming, ATmega88P, ATmega168P and ATmega328P 277

24.1Features 277

24.2Overview 277

24.3Application and Boot Loader Flash Sections 277

24.4Read-While-Write and No Read-While-Write Flash Sections 278

24.5Boot Loader Lock Bits 280

24.6Entering the Boot Loader Program 281

24.7Addressing the Flash During Self-Programming 282

24.8Self-Programming the Flash 282

24.9Register Description 292

25 Memory Programming 294

25.1Program And Data Memory Lock Bits 294

25.2Fuse Bits 295

25.3Signature Bytes 298

25.4Calibration Byte 299

25.5Page Size 299

25.6Parallel Programming Parameters, Pin Mapping, and Commands 299

25.7Parallel Programming 301

25.8Serial Downloading 308

26 Electrical Characteristics 313

26.1Absolute Maximum Ratings* 313

26.2DC Characteristics 313

26.3Speed Grades 316

26.4Clock Characteristics 319

26.5System and Reset Characteristics 320

26.6SPI Timing Characteristics 321

26.72-wire Serial Interface Characteristics 323

26.8ADC Characteristics – Preliminary Data 325

26.9Parallel Programming Characteristics 326

27 Typical Characteristics 328

27.1ATmega48P Typical Characteristics 329
27.2ATmega88P Typical Characteristics 353
27.3ATmega168P Typical Characteristics 377
27.4ATmega328P Typical Characteristics 401

28 Register Summary 425

29 Instruction Set Summary 429

30 Ordering Information 432

30.1ATmega48P 432
30.2ATmega88P 433
30.3ATmega168P 434
30.4ATmega328P 435

31 Packaging Information 436

31.132A 436
31.228M1 437
31.332M1-A 438
31.428P3 439

32 Errata 440

32.1Errata ATmega48P 440
32.2Errata ATmega88P 440
32.3Errata ATmega168P 440
32.4Errata ATmega328P 440

33 Datasheet Revision History 442

33.1Rev. 2545I-02/09 442
33.2Rev. 2545H-02/09 442
33.3Rev. 2545G-01/09 442
33.4Rev. 2545F-08/08 443
33.5Rev. 2545E-08/08 443
33.6Rev. 2545D-03/08 443
33.7Rev. 2545C-01/08 443
33.8Rev. 2545B-01/08 444
33.9Rev. 2545A-07/07 444

Table of Contents..... i

Headquarters

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

International

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
Tel: (852) 2245-6100
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Product Contact

Web Site
www.atmel.com

Technical Support
avr@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Requests
www.atmel.com/literature

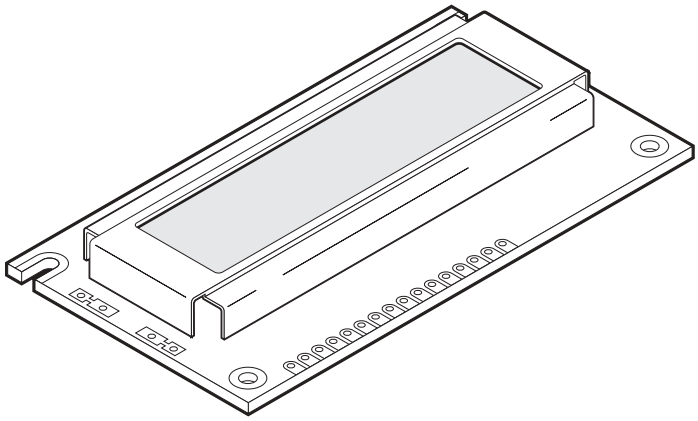
Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. **EXCEPT AS SET FORTH IN ATMEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATMEL'S WEB SITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.** Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel's products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

© 2009 Atmel Corporation. All rights reserved. Atmel®, logo and combinations thereof, AVR® and others are registered trademarks or trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others.

ALPHANUMERIC LCD DISPLAY (16 x 2)

Order Code

LED008 16 x 2 Alphanumeric Display
FRM010 Serial LCD Firmware (optional)



Contents

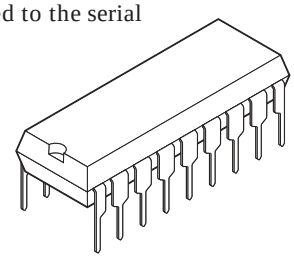
1 x 16x2 Alphanumeric Display
1 x data booklet

Introduction

Alphanumeric displays are used in a wide range of applications, including palmtop computers, word processors, photocopiers, point of sale terminals, medical instruments, cellular phones, etc. The 16 x 2 intelligent alphanumeric dot matrix display is capable of displaying 224 different characters and symbols. A full list of the characters and symbols is printed on pages 7/8 (note these symbols can vary between brand of LCD used). This booklet provides all the technical specifications for connecting the unit, which requires a single power supply (+5V).

Further Information

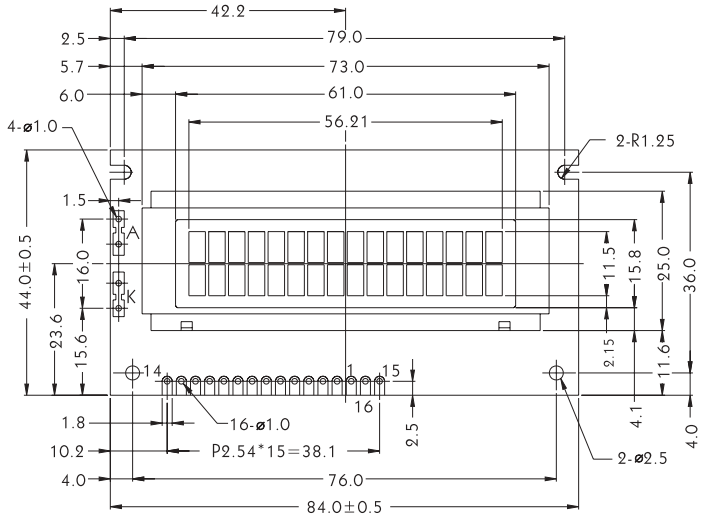
Available as an optional extra is the Serial LCD Firmware, which allows serial control of the display. This option provides much easier connection and use of the LCD module. The firmware enables microcontrollers (and microcontroller based systems such as the PICAXE) to visually output user instructions or readings onto an LCD module. All LCD commands are transmitted serially via a single microcontroller pin. The firmware can also be connected to the serial port of a computer.

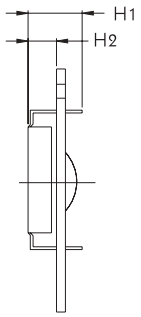


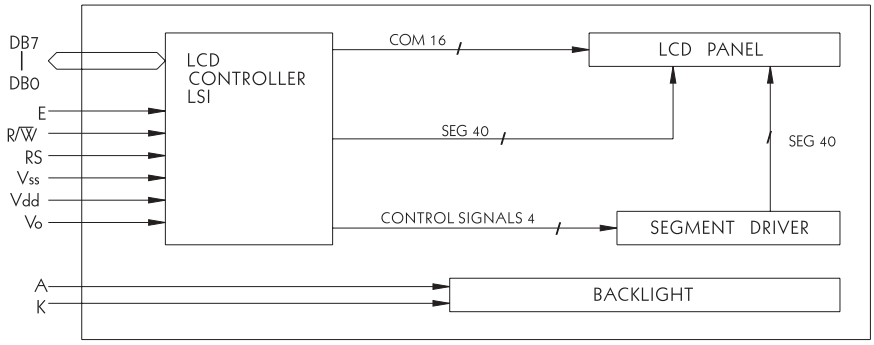
An example PICAXE instruction to print the text 'Hello' using the **serout** command is as follows:

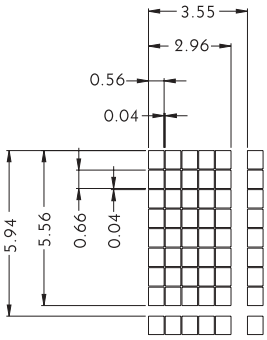
```
serout 7,T2400,("Hello")
```

Outline Dimension and Block Diagram









The tolerance unless classified ±0.3mm

MECHANICAL SPECIFICATION			
Overall Size	84.0 * 44.0	Module	H2 / H1
View Area	61.0 * 15.8	W/O B/L	5.1 / 9.7
Dot Size	0.56 * 0.66	EL B/L	5.1 / 9.7
Dot Pitch	0.60 * 0.70	LED B/L	9.4 / 14.0

PIN ASSIGNMENT		
Pin no.	Symbol	Function
1	Vss	Power supply (GND)
2	Vdd	Power supply (+5V)
3	Vo	Contrast Adjust
4	RS	Register select signal
5	R/W	Data read /write
6	E	Enable signal
7	DB0	Data bus line
8	DB1	Data bus line
9	DB2	Data bus line
10	DB3	Data bus line
11	DB4	Data bus line
12	DB5	Data bus line
13	DB6	Data bus line
14	DB7	Data bus line
15	A	Power supply for LED B/L (+)
16	K	Power supply for LED B/L (−)

ABSOLUTE MAXIMUM RATING					
Item	Symbol	Conditions	Min.	Max.	Unit
Power Supply Voltage	Vdd−Vss	−	0	7	V
LCD Driving Supply Voltage	Vdd−Vee	−	0	13	V
Input Voltage	Vin	−	−0.3	Vdd+0.3	V
Operating Temperature	Topr	Nor.	0	50	°C
Storage Temperature	Tstg	Nor.	−20	+70	°C

ELECTRICAL CHARACTERISTICS (Vdd =+5V, Ta =25°C)						
Item	Symbol	Conditions	Min.	Typ.	Max.	Unit
Logic Supply Voltage	Vdd	−	4.5	5	5.5	V
"H" Input Voltage	V_{IH}	−	2.2	−	−	V
"L" Input Voltage	V_{IL}	−	−	−	0.6	V
"H" Output Voltage	V_{OH}	−	2.4	−	−	V
"L" Output Voltage	V_{OL}	−	−	−	0.4	V
Supply Current	Idd	−	2	−	−	mA
LCD Driving Voltage	V_{LCD}	Vdd−Vo	4.3	−	4.8	V

Electrical Characteristics

Vdd = 5V±5%
Vss = 0V

Item	Symbol	Condition	Standard value			Unit	Applicable terminal
			Min.	Typ.	Max.		
Power voltage	Vdd		4.5	5.00	5.5	V	Vdd
Input H- level voltage	VIH		2.2	—	Vdd	V	RS, R/$\overline{W}$, E DB0~DB7
Input L - level voltage	VIL		−0.3	—	0.6	V	
Output H - level voltage	VOH	−IOH = 0.205mA	2.4	—	—	V	DB0~DB7
Output L - level voltage	VOL	IOL = 1.2mA	—	—	0.4	V	
I/O leakage current	IIL	Vin = 0~Vdd	−1	—	1.0	μA	RS, R/$\overline{W}$, E DB0~DB7
Supply current	Idd	Vdd = 5V	2	—	—	mA	Vdd
LCD operating voltage	VLCD	Vdd—V0	3.0	—	11.0	V	V0

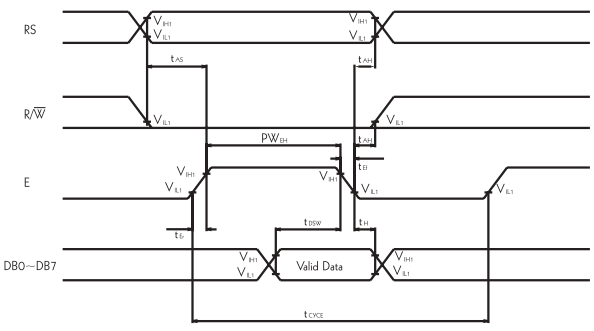
Timing Characteristics

Vdd = 5V±5%
Vss = 0V

Item		Symbol	Min.	Max.	Unit
Enable cycle time		TCYCE	500	—	ns
Enable pulse width	"High" level	PWEH	220	—	ns
Enable rise / fall time		TER, TEF	—	25	ns
Set-up time	RS, R/$\overline{W}$, E	TAS	40	—	ns
Address hold time		TAH	10	—	ns
Data set-up time		TDSH	60	—	ns
Data delay time		TDDR	60	120	ns
Data hold time (writing)		TH	10	—	ns
Data hold time (reading)		TDHR	20	—	ns
Clock oscillating frequency		TOSC	270(TYP.)		KHz

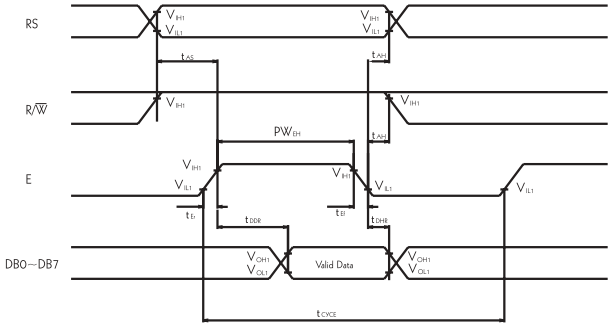
Timing Chart

◆ FIG.1 WRITE OPERATION



(Write Data from MPU to MODULE)

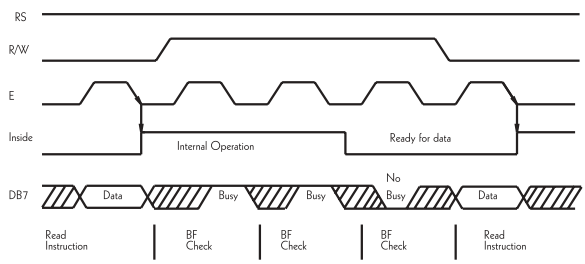
◆ FIG.2 READ OPERATION

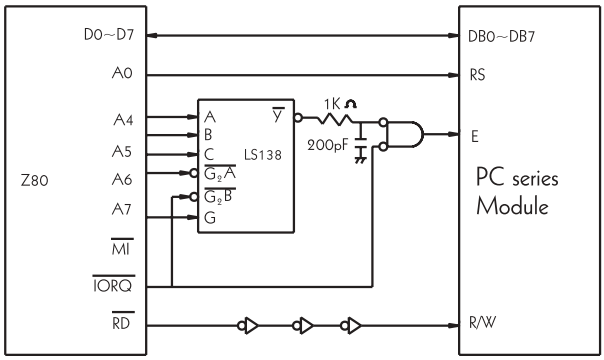


(Read Data from MODULE to MPU)

Interface with MPU

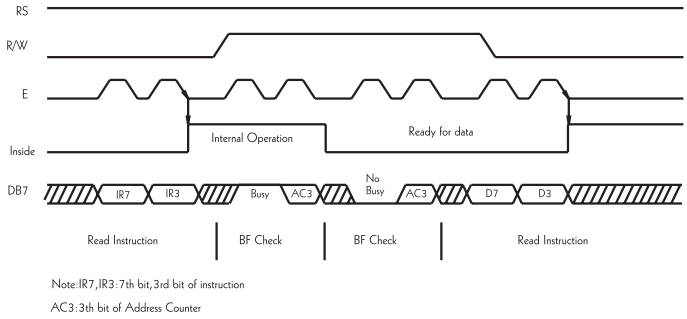
Example of Interface with 8-bit MPU (Z80)

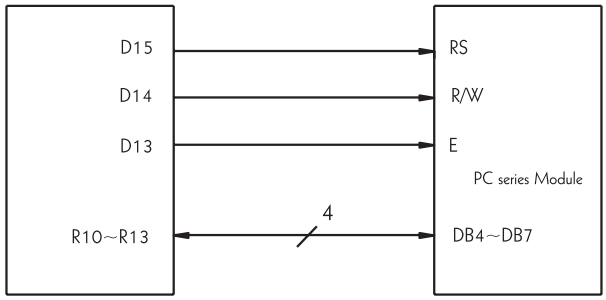




Example of interface with 4-bit MPU

Interface with 4-bit MPU can be made through I / O port of 4-bit MPU. If there are enough I / O ports, data can be transfered by 8-bit, however, if there are not data transfer can be done by 4-bit in twice (select interface is 4-bit long), and timing sequence will be complicated in this case. Please take into account that 2 cycles of BF check is necessary, while 2 cycles of data transfer are also necessary.





Features

(1) Interface with 8-bit or 4-bit MPU is available.
(2) 192 kind of alphabets, numerals, symbols and special characters can be displayed by built-in character generator (ROM).
(3) Other preferred characters can be displayed by character generator (RAM).
(4) Various functions of instruction are available by programming.
- Clear display • Cursor at home • On / off cursor
- Blink character • Shift display • Shift cursor
- Read / write display data.....etc.

(5) Compact and light weight design which can be easily assembled in devices.
(6) Single power supply +5V drive (except for extended temp. type).
(7) Low power consumption.

*Interface between data bus line and 4-bit or 8-bit MPU is available. Data transfer are made in twice in case of 4-bit MPU, and once in case of 8-bit MPU.

If interface data is 4-bit long

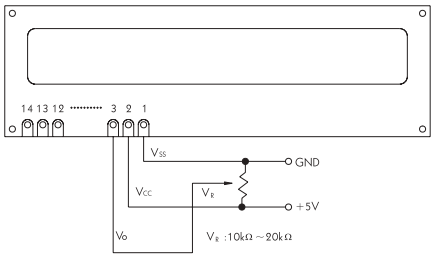
Data transfer are made through 4 bus lines from DB4 to DB7. (while the rest of 4 bus lines fromDB0 to DB3 are not used.) Data transfer with MPU are completed when 4-bit data are transfered in twice.
(first upper 4-bit data. then lower 4-bit data.)

If interface data is 8-bit long

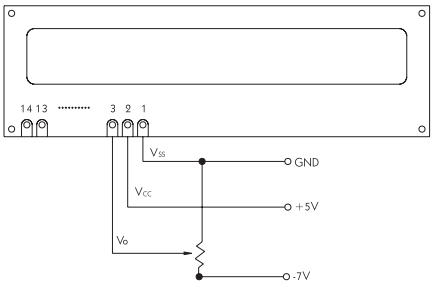
Data transfer are made through all of 8 bus lines from DB0 to DB7.

Example of Power Supply

Normal Temperature Type

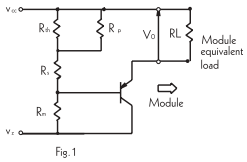


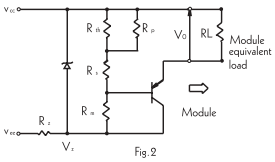
Extended Temperature Type



Examples of Temperature Compensation Circuits for Extended Temp Type. (Only for reference)

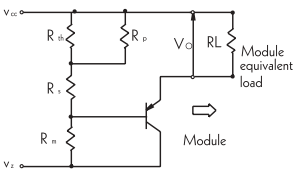
(A) 1/8Duty—1/4Bias

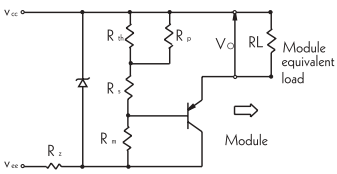




Thermistor: Rth(25°C)=15[k-ohm], B=4200[K]
Resistors: Rp=30[k-ohm]], Rs=6.8[k-ohm], Rm=3.3[k-ohm]
Transistor: PNP Type
Vcc: +5V , Vss: 0V (Logic Supply)
Vz: -8[V] (-7.8 to -8.2[V])
Vee<Vz[V]. Rz=(Vz-Vee) / 5[k-ohm]

(B) 1/16Duty—1/5Bias





Thermistor: Rth(25°C)=15[k-ohm], B=4200[K]
Resistors: Rp=510[k-ohm]], Rs=8.2[k-ohm], Rm=3.9[k-ohm]
Transistor: PNP Type
Vcc: +5V , Vss: 0V (Logic Supply)
Vz: -11[V] (-10.725 to -11.275[V])
Vee<Vz[V]. Rz=(Vz-Vee) / 5[k-ohm]

Instructions

Instruction	Code										Description	Executed Time(max.)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Clears all display and returns the cursor to the home position (Address 0)	1.64mS
Cursor At Home	0	0	0	0	0	0	0	0	1	*	Returns the cursor to the home position (Address 0). Also returns the display being shifted to the original position. DD RAM contents remain unchanged.	1.64mS
Entry Mode Set	0	0	0	0	0	0	0	1	1/D	S	Sets the cursor move direction and specifies or not to shift the display. These operations are performed during data write and read.	40µS
Display On / Off Control	0	0	0	0	0	0	1	D	C	B	Sets ON / OFF of all display (D), cursor NO / OFF (C), and blink of cursor position character (B).	40µS
Cursor / Display Shift	0	0	0	0	0	1	S/C	R/L	*	*	Moves the cursor and shifts the display without changing DD RAM contents.	40µS
Function Set	0	0	0	0	1	DL	N	F	*	*	Sets interface data length (DL) number of display lines (L) and character font (F)	40µS
CG RAM Address Set	0	0	0	1	ACG						Sets the CG RAM address. CG RAM data is sent and received after this setting.	40µS
DD RAM Address Set	0	0	1	ADD							Sets the DD RAM address. DD RAM data is sent and received after this setting.	40µS
Busy Flag / Address Read	0	1	BF	AC							Reads Busy flag (FB) indicating internal operation is being performed and reads address counter counts.	0µS
CG RAM / DD RAM Data Write	1	0	WRITE DATA								Writes data into DD RAM or CG RAM.	40µS
CG RAM / DD RAM Data Read	1	1	READ DATA								Reads data from DD RAM or CG RAM.	40µS

Code		Descripion	Executed Time (max)
I/D = 1:Increment I/D = 0:Decrement S= 1:With display shift S/C= 0:cursor movement R/L= 1: Shift to the right R/L= 0: Shift to the left DL = 1: 8-bit	DL = 0:4-bit N = 1:2lines N = 0:1line F = 1:5×10dots F = 0.5×7dots BF = 1:Internal operation is being performed BF = 0:Instruction acceptable	DD RAM:Display Data RAM CG RAM:Character Generator RAM ACG:CG RAM Address ADD:DD RAM Address Corresponds to cursor address. AC:Address Counter, used for both DD RAM and CG RAM *:Invalid	fcp or fosc = 250KHz However. when frequency changes. eecution time also changes Example if fcp or fosc is 270KHz, 70µS x 250 / 270 = 37µS

Power Supply Reset

The internal reset circuit will be operated properly when the following power supply conditions are satisfied. If it is not operated properly, please perform initial setting along with the instruction.

Item	Symbol	Measuring Condition	Standard Value			Unit
			Min.	Typ.	Max.	
Power Supply RISE Time	tree	—	0.1	—	10	mS
Power Supply CFF Time	toff	—	1	—	—	mS

Reset function

◆ **Initialization Made by Internal Reset Circuit**

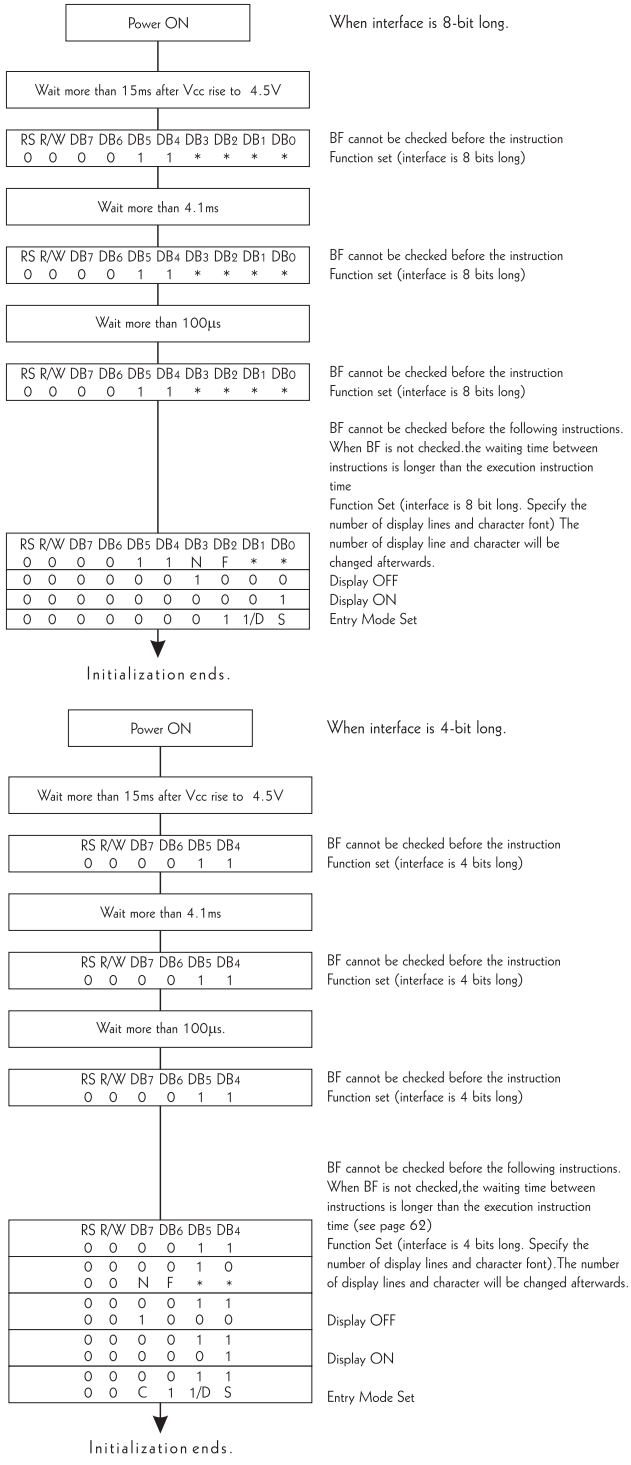
HD44780 automatically initializes (resets) when power is supplied (builtin internal reset circuit). The following instructions are executed in initialization. The busy flag (BF) is kept in busy state until initialization ends. (BF=1) The busy state is 10 ms after Vdd reachs to 4.5V.

(1) Display clear
(2) Function set
DL= 1:8 bit long interface data
DL= 0:4 bit F= 0:5 x 7dots character font
N= 1:2 lines
N= 0:1 line
(3) Display ON / OFF control
D= 0:Display OFF C= 0:Cursor OFF
B= 0:Blink OFF
(4) Entry mode set
1 / D= 1:+1(increment) S= 0:No shift

Note:When conditions stated in power supply conditions using internal reset circuit are not satisfied.The internal reset circuit will not operate properly and initialization will not be performed. Please make initialization using MPU along with instruction.

◆ **Initialization along with instruction**

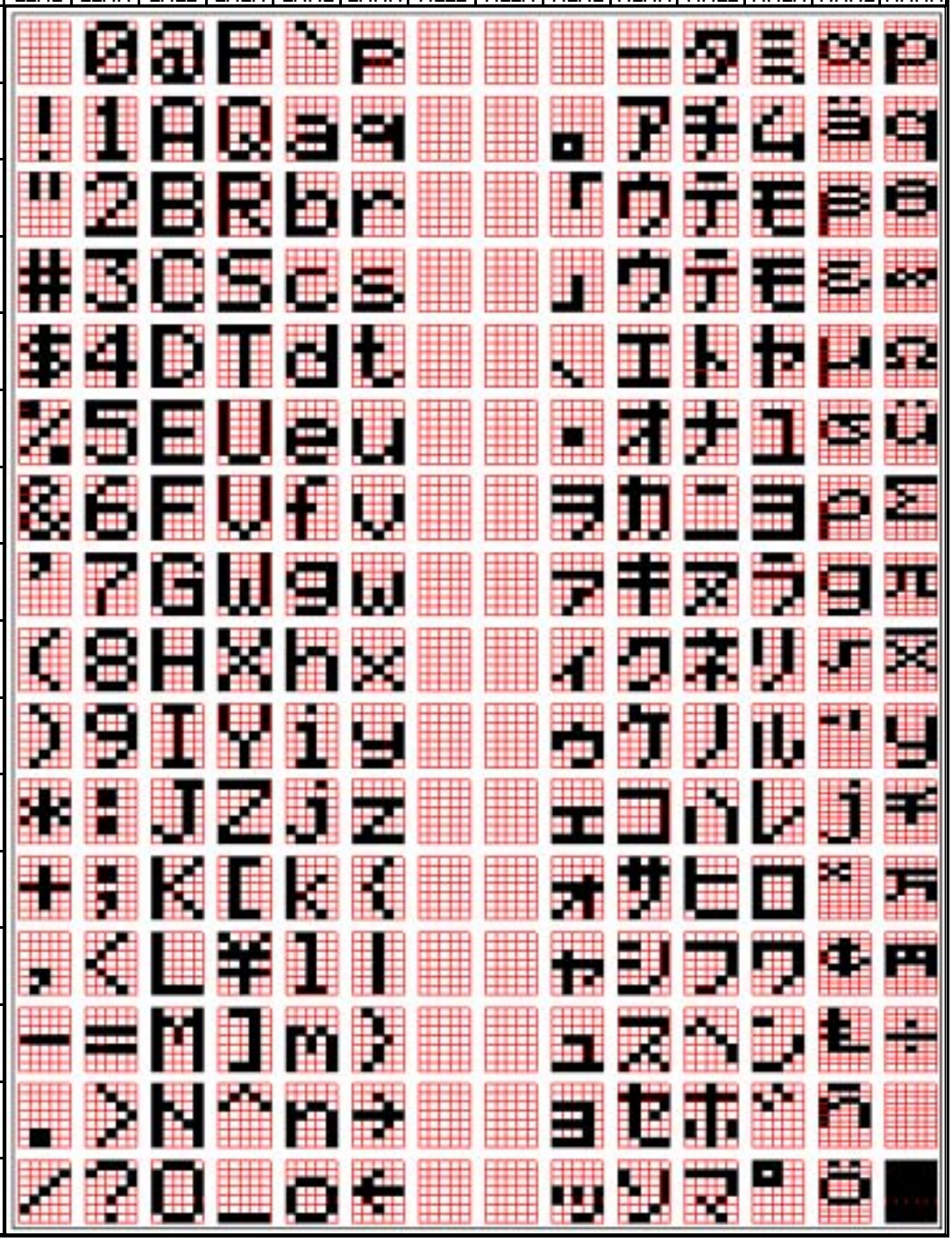
If power supply conditions are not satisfied, which for proper operation of internal rest circuit, it is required to make initialzation along with instruction. Please make following procedures.



Standard Character Pattern (Powertip Module)



Standard Character Pattern (Elec & Eltek Module)

Upper(4bit) / Lower(4bit)	LLLL	LLHL	LLHH	LHLL	LHLH	LHHL	LHHH	HLLL	HLLH	HLHL	HLHH	HHLL	HHLH	HHHL	HHHH
LLLL	CG RAM (1)														
LLLH	(2)														
LLHL	(3)														
LLHH	(4)														
LHLL	(5)														
LHLH	(6)														
LHHL	(7)														
LHHH	(8)														
HLLL	(1)														
HLLH	(2)														
HLHL	(3)														
HLHH	(4)														
HHLL	(5)														
HHLH	(6)														
HHHL	(7)														
HHHH	(8)														

LM79XX Series 3-Terminal Negative Regulators

Check for Samples: LM7905, LM7912, LM7915

FEATURES

- **Thermal, Short Circuit and Safe Area Protection**
- **High Ripple Rejection**
- **1.5A Output Current**
- **4% Tolerance on Preset Output Voltage**

DESCRIPTION

The LM79XX series of 3-terminal regulators is available with fixed output voltages of −5V, −12V, and −15V. These devices need only one external component—a compensation capacitor at the output. The LM79XX series is packaged in the TO-220 power package and is capable of supplying 1.5A of output current.

These regulators employ internal current limiting safe area protection and thermal shutdown for protection against virtually all overload conditions.

Low ground pin current of the LM79XX series allows output voltage to be easily boosted above the preset value with a resistor divider. The low quiescent current drain of these devices with a specified maximum change with line and load ensures good regulation in the voltage boosted mode.

For applications requiring other voltages, see LM137 datasheet.

Connection Diagram

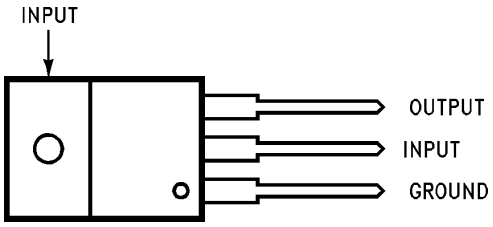
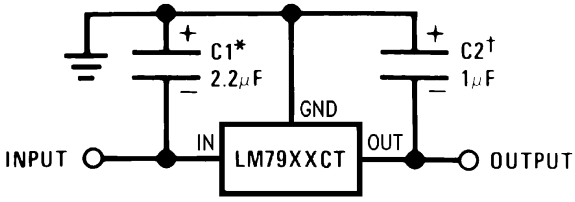


Figure 1. TO-220 Package
Front View
See Package Number NDE0003B

Typical Applications



*Required if regulator is separated from filter capacitor by more than 3″. For value given, capacitor must be solid tantalum. 25μF aluminum electrolytic may be substituted.

†Required for stability. For value given, capacitor must be solid tantalum. 25μF aluminum electrolytic may be substituted. Values given may be increased without limit.

For output capacitance in excess of 100μF, a high current diode from input to output (1N4001, etc.) will protect the regulator from momentary input shorts.

Figure 2. Fixed Regulator

Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

All trademarks are the property of their respective owners.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of the Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

Copyright © 1999–2013, Texas Instruments Incorporated

These devices have limited built-in ESD protection. The leads should be shorted together or the device placed in conductive foam during storage or handling to prevent electrostatic damage to the MOS gates.

ABSOLUTE MAXIMUM RATINGS(1)

Input Voltage	
(V_o = −5V)	−25V
(V_o = −12V and −15V)	−35V
Input-Output Differential	
(V_o = −5V)	25V
(V_o = −12V and −15V)	30V
Power Dissipation (2)	Internally Limited
Operating Junction Temperature Range	0°C to +125°C
Storage Temperature Range	−65°C to +150°C
Lead Temperature (Soldering, 10 sec.)	230°C

(1) Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. Operating Ratings indicate conditions for which the device is intended to be functional, but do not ensure Specific Performance limits. For ensured specifications and test conditions, see the Electrical Characteristics.
(2) Refer to DESIGN CONSIDERATIONS for details.

ELECTRICAL CHARACTERISTICS

Conditions unless otherwise noted: I_{OUT} = 500mA, C_{IN} = 2.2µF, C_{OUT} = 1µF, 0°C ≤ T_J ≤ +125°C, Power Dissipation ≤ 1.5W.

Part Number			**LM7905C**			**Units**
Output Voltage			**−5V**			
Input Voltage (unless otherwise specified)			**−10V**			
Symbol	**Parameter**	**Conditions**	**Min**	**Typ**	**Max**	
V_O	Output Voltage	T_J = 25°C	−4.8	−5.0	−5.2	V
		5mA ≤ I_{OUT} ≤ 1A,	−4.75		−5.25	V
		P ≤ 15W		(−20 ≤ V_{IN} ≤ −7)		V
ΔV_O	Line Regulation	T_J = 25°C, (1)		8	50	mV
				(−25 ≤ V_{IN} ≤ −7)		V
				2	15	mV
				(−12 ≤ V_{IN} ≤ −8)		V
ΔV_O	Load Regulation	T_J = 25°C, (1)				
		5mA ≤ I_{OUT} ≤ 1.5A		15	100	mV
		250mA ≤ I_{OUT} ≤ 750mA		5	50	mV
I_Q	Quiescent Current	T_J = 25°C		1	2	mA
ΔI_Q	Quiescent Current	With Line			0.5	mA
	Change			(−25 ≤ V_{IN} ≤ −7)		V
		With Load, 5mA ≤ I_{OUT} ≤ 1A			0.5	mA
V_n	Output Noise Voltage	T_A = 25°C, 10Hz ≤ f ≤ 100Hz		125		µV
	Ripple Rejection	f = 120Hz	54	66		dB
				(−18 ≤ V_{IN} ≤ −8)		V
	Dropout Voltage	T_J = 25°C, I_{OUT} = 1A		1.1		V
I_{OMAX}	Peak Output Current	T_J = 25°C		2.2		A
	Average Temperature	I_{OUT} = 5mA,		0.4		mV/°C
	Coefficient of	0 C ≤ T_J ≤ 100°C				
	Output Voltage					

(1) Regulation is measured at a constant junction temperature by pulse testing with a low duty cycle. Changes in output voltage due to heating effects must be taken into account.

Copyright © 1999–2013, Texas Instruments Incorporated

ELECTRICAL CHARACTERISTICS

Conditions unless otherwise noted: I_{OUT} = 500mA, C_{IN} = 2.2µF, C_{OUT} = 1µF, 0°C ≤ T_J ≤ +125°C, Power Dissipation ≤ 1.5W.

Part Number			LM7912C			LM7915C			Units
Output Voltage			−12V			−15V			
Input Voltage (unless otherwise specified)			−19V			−23V			
Symbol	Parameter	Conditions	Min	Typ	Max	Min	Typ	Max	
V_O	Output Voltage	T_J = 25°C	−11.5	−12.0	−12.5	−14.4	−15.0	−15.6	V
		5mA ≤ I_{OUT} ≤ 1A,	−11.4		−12.6	−14.25		−15.75	V
		P ≤ 15W	(−27 ≤ V_{IN} ≤ −14.5)			(−30 ≤ V_{IN} ≤ −17.5)			V
ΔV_O	Line Regulation	T_J = 25°C, (1)		5	80		5	100	mV
			(−30 ≤ V_{IN} ≤ −14.5)			(−30 ≤ V_{IN}≤ −17.5)			V
				3	30		3	50	mV
			(−22 ≤ V_{IN} ≤ −16)			(−26 ≤ V_{IN} ≤−20)			V
ΔV_O	Load Regulation	T_J = 25°C, (1)							
		5mA ≤ I_{OUT} ≤ 1.5A		15	200		15	200	mV
		250mA ≤ I_{OUT} ≤ 750mA		5	75		5	75	mV
I_Q	Quiescent Current	T_J = 25°C		1.5	3		1.5	3	mA
ΔI_Q	Quiescent Current	With Line			0.5			0.5	mA
	Change		(−30 ≤ V_{IN} ≤ −14.5)			(−30 ≤V_{IN} ≤ −17.5)			V
		With Load, 5mA ≤ I_{OUT} ≤ 1A			0.5			0.5	mA
V_n	Output Noise Voltage	T_A = 25°C, 10Hz ≤ f ≤ 100Hz		300			375		µV
	Ripple Rejection	f = 120 Hz	54	70		54	70		dB
			(−25 ≤ V_{IN} ≤ −15)			(−30 ≤ V_{IN}≤ −17.5)			V
	Dropout Voltage	T_J = 25°C, I_{OUT} = 1A		1.1			1.1		V
I_{OMAX}	Peak Output Current	T_J = 25°C		2.2			2.2		A
	Average Temperature	I_{OUT} = 5mA,		−0.8			−1.0		mV/°C
	Coefficient of	0 C ≤ T_J ≤ 100°C							
	Output Voltage								

(1) Regulation is measured at a constant junction temperature by pulse testing with a low duty cycle. Changes in output voltage due to heating effects must be taken into account.

Copyright © 1999–2013, Texas Instruments Incorporated

DESIGN CONSIDERATIONS

The LM79XX fixed voltage regulator series has thermal overload protection from excessive power dissipation, internal short circuit protection which limits the circuit's maximum current, and output transistor safe-area compensation for reducing the output current as the voltage across the pass transistor is increased.

Although the internal power dissipation is limited, the junction temperature must be kept below the maximum specified temperature (125°C) in order to meet data sheet specifications. To calculate the maximum junction temperature or heat sink required, the following thermal resistance values should be used:

Package	Typ θ_{JC} °C/W	Max θ_{JC} °C/W	Typ θ_{JA} °C/W	Max θ_{JA} °C/W
TO-220	3.0	5.0	60	40

$$P_{D\,MAX} = \frac{T_{J\,Max} - T_A}{\theta_{JC} + \theta_{CA}} \text{ or } \frac{T_{J\,Max}\ T_A}{\theta_{JA}}$$

$$\theta_{CA} = \theta_{CS} + \theta_{SA} \text{ (without heat sink)} \quad (1)$$

Solving for T_J:

$$T_J = T_A + P_D (\theta_{JC} + \theta_{CA})$$

or

$$= T_A + P_D\theta_{JA} \text{ (without heat sink)}$$

where

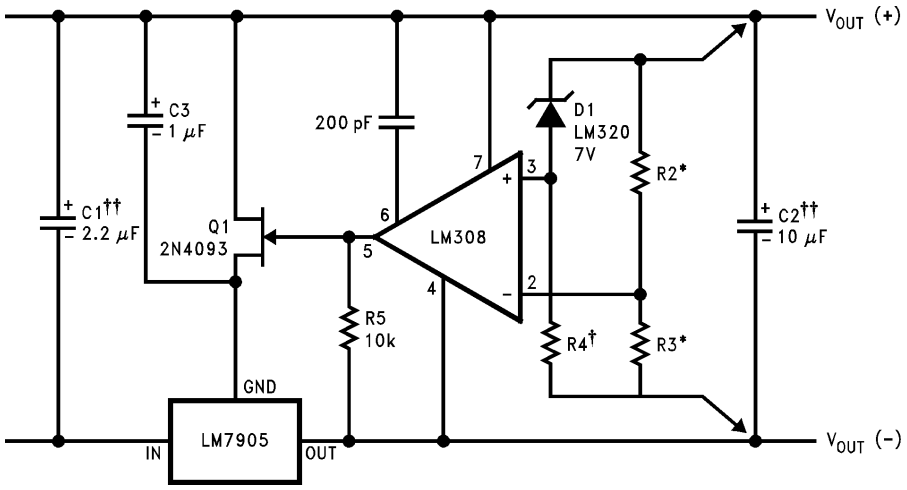
- T_J = Junction Temperature
- T_A = Ambient Temperature
- P_D = Power Dissipation
- θ_{JA} = Junction-to-Ambient Thermal Resistance
- θ_{JC} = Junction-to-Case Thermal Resistance
- θ_{CA} = Case-to-Ambient Thermal Resistance
- θ_{CS} = Case-to-Heat Sink Thermal Resistance
- θ_{SA} = Heat Sink-to-Ambient Thermal Resistance

Typical Applications

Bypass capacitors are necessary for stable operation of the LM79XX series of regulators over the input voltage and output current ranges. Output bypass capacitors will improve the transient response by the regulator.

The bypass capacitors, (2.2µF on the input, 1.0µF on the output) should be ceramic or solid tantalum which have good high frequency characteristics. If aluminum electrolytics are used, their values should be 10µF or larger. The bypass capacitors should be mounted with the shortest leads, and if possible, directly across the regulator terminals.

Copyright © 1999–2013, Texas Instruments Incorporated



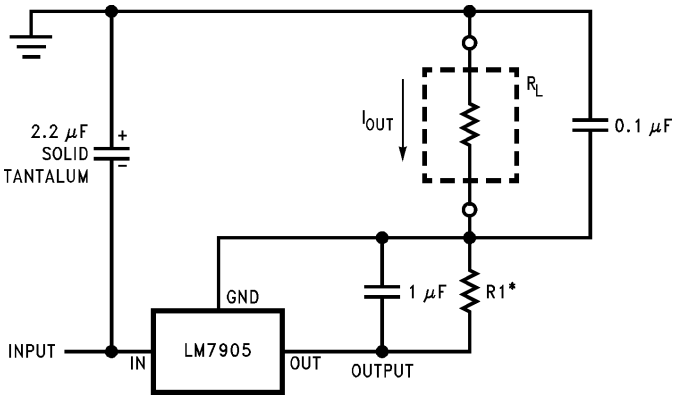
Load and line regulation < 0.01% temperature stability ≤ 0.2%

†Determine Zener current

††Solid tantalum

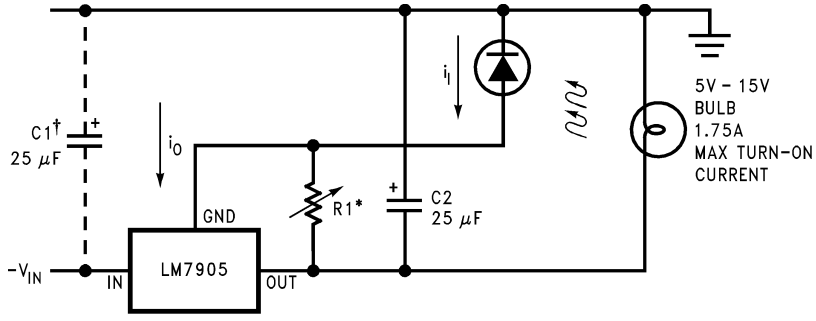
*Select resistors to set output voltage. 2 ppm/°C tracking suggested

Figure 3. High Stability 1 Amp Regulator



$*I_{OUT} = 1\ mA + \frac{5V}{R1}$

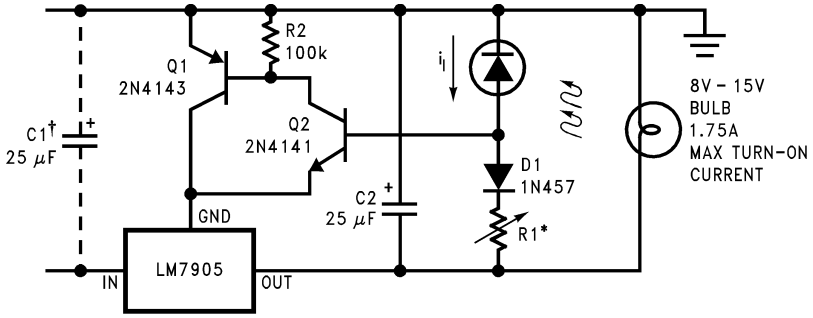
Figure 4. Current Source



*Lamp brightness increase until $i_I = i_Q$ (≈ 1 mA) + 5V/R1.

†Necessary only if raw supply filter capacitor is more that 2″ from LM7905CT

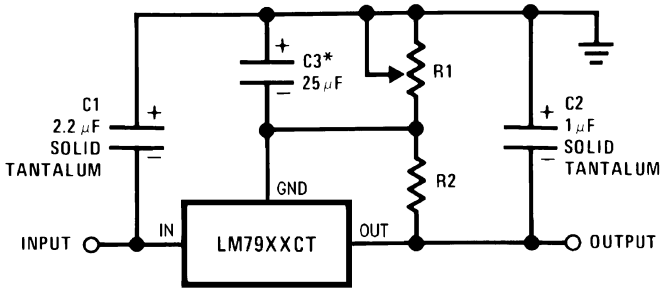
Figure 5. Light Controller Using Silicon Photo Cell



*Lamp brightness increases until i_i = 5V/R1 (I_i can be set as low as 1 μA)

†Necessary only if raw supply filter capacitor is more that 2″ from LM7905

Figure 6. High-Sensitivity Light Controller



*Improves transient response and ripple rejection. Do not increase beyond 50 μF.

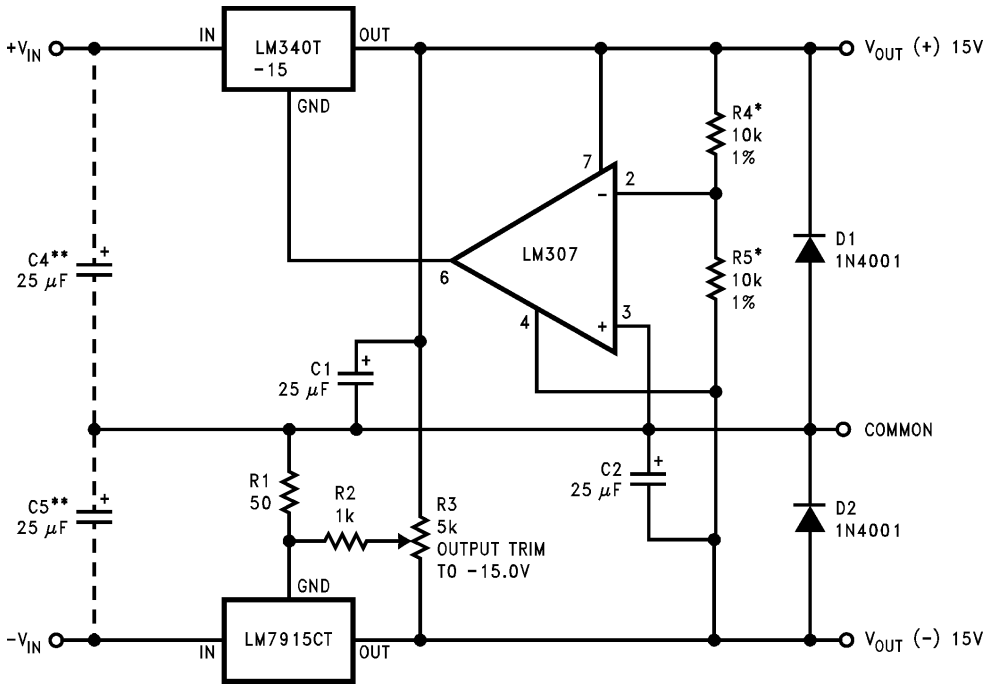
$V_{OUT} = V_{SET}\left(\frac{R1 + R2}{R2}\right)$

Select R2 as follows:

LM7905CT	300Ω
LM7912CT	750Ω
LM7915CT	1k

Figure 7. Variable Output

Copyright © 1999–2013, Texas Instruments Incorporated



	(-15)	(+15)
Load Regulation at ΔI_L = 1A	40mV	2mV
Output Ripple, C_{IN} = 3000µF, I_L = 1A	100 µVms	100 µVms
Temperature Stability	50mV	50mV
Output Noise 10Hz ≤ f ≤ 10kHz	150 µVms	150 µVms

*Resistor tolerance of R4 and R5 determine matching of (+) and (−) outputs.

**Necessary only if raw supply filter capacitors are more than 3″ from regulators.

Figure 8. ±15V, 1 Amp Tracking Regulators

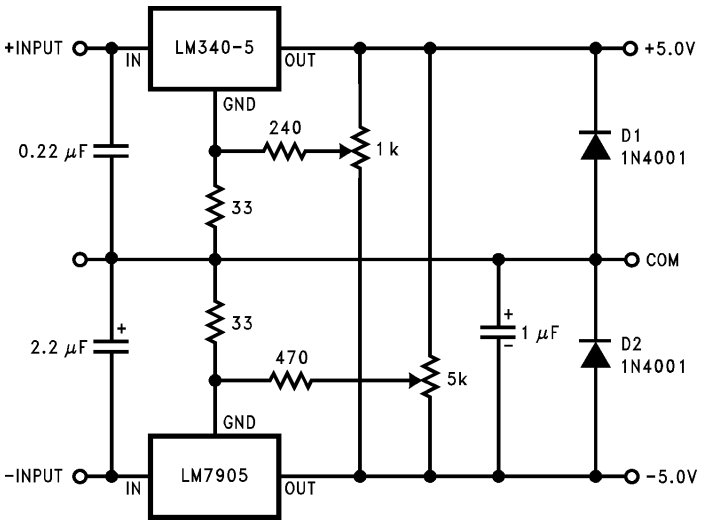


Figure 9. Dual Trimmed Supply

Schematic Diagrams

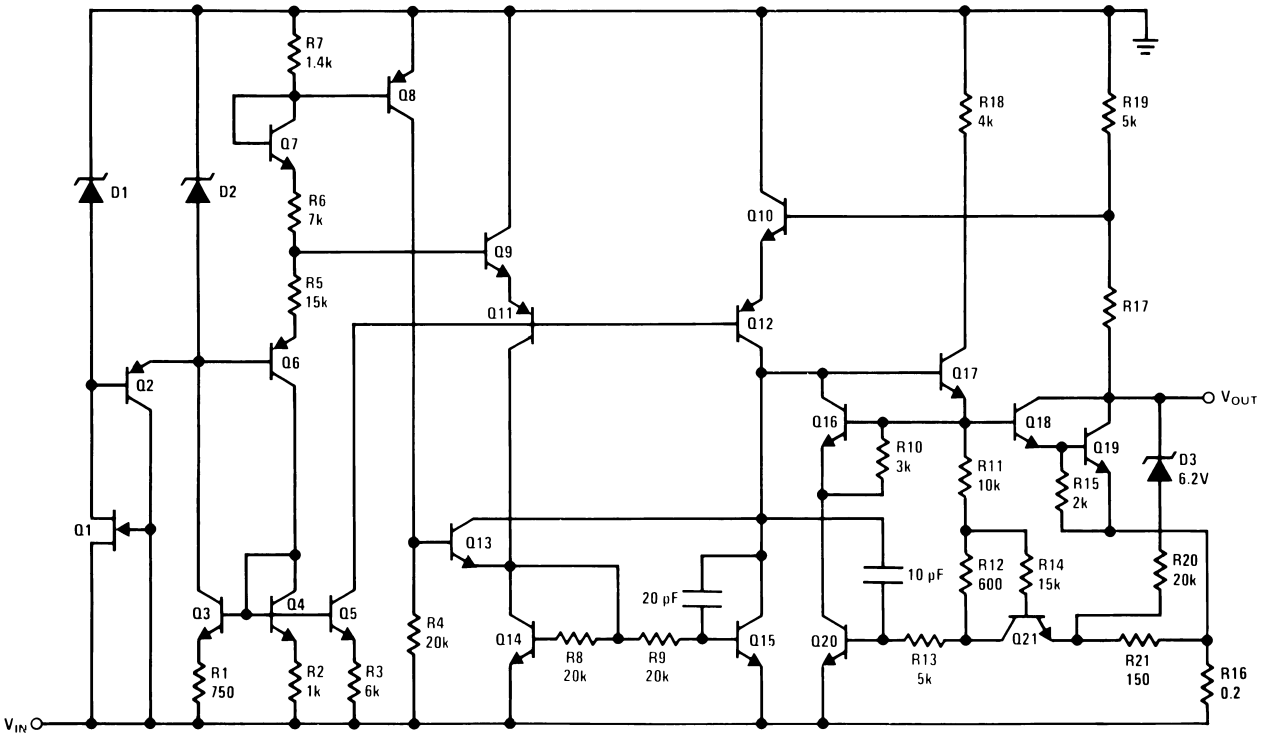


Figure 10. −5V

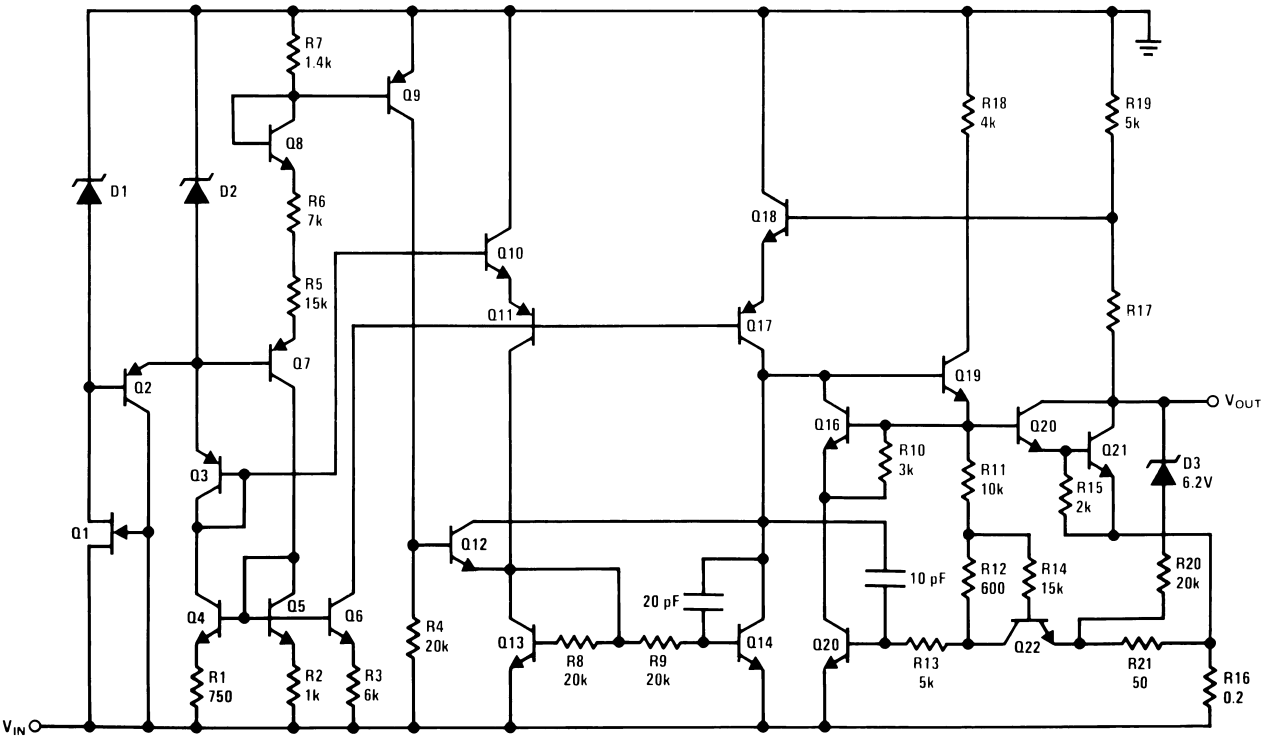


Figure 11. −12V and −15V

Copyright © 1999–2013, Texas Instruments Incorporated

REVISION HISTORY

Changes from Revision B (May 2013) to Revision C	Page
• Changed layout of National Data Sheet to TI format.	8

Copyright © 1999–2013, Texas Instruments Incorporated

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish (6)	MSL Peak Temp (3)	Op Temp (°C)	Device Marking (4/5)	Samples
LM7905CT	NRND	TO-220	NDE	3	45	TBD	Call TI	Call TI	0 to 125	LM7905CT	
LM7905CT/NOPB	ACTIVE	TO-220	NDE	3	45	Green (RoHS & no Sb/Br)	CU SN	Level-1-NA-UNLIM	0 to 125	LM7905CT	Samples
LM7912CT	NRND	TO-220	NDE	3	45	TBD	Call TI	Call TI	0 to 125	LM7912CT	
LM7912CT/NOPB	ACTIVE	TO-220	NDE	3	45	Green (RoHS & no Sb/Br)	CU SN	Level-1-NA-UNLIM	0 to 125	LM7912CT	Samples
LM7915CT	NRND	TO-220	NDE	3	45	TBD	Call TI	Call TI	0 to 125	LM7915CT	
LM7915CT/NOPB	ACTIVE	TO-220	NDE	3	45	Green (RoHS & no Sb/Br)	CU SN	Level-1-NA-UNLIM	0 to 125	LM7915CT	Samples

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

(2) **RoHS:** TI defines "RoHS" to mean semiconductor products that are compliant with the current EU RoHS requirements for all 10 RoHS substances, including the requirement that RoHS substance do not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, "RoHS" products are suitable for use in specified lead-free processes. TI may reference these types of products as "Pb-Free".
RoHS Exempt: TI defines "RoHS Exempt" to mean products that contain lead but are compliant with EU RoHS pursuant to a specific EU RoHS exemption.
Green: TI defines "Green" to mean the content of Chlorine (Cl) and Bromine (Br) based flame retardants meet JS709B low halogen requirements of <=1000ppm threshold. Antimony trioxide based flame retardants must also meet the <=1000ppm threshold requirement.

(3) MSL, Peak Temp. - The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

(4) There may be additional marking, which relates to the logo, the lot trace code information, or the environmental category on the device.

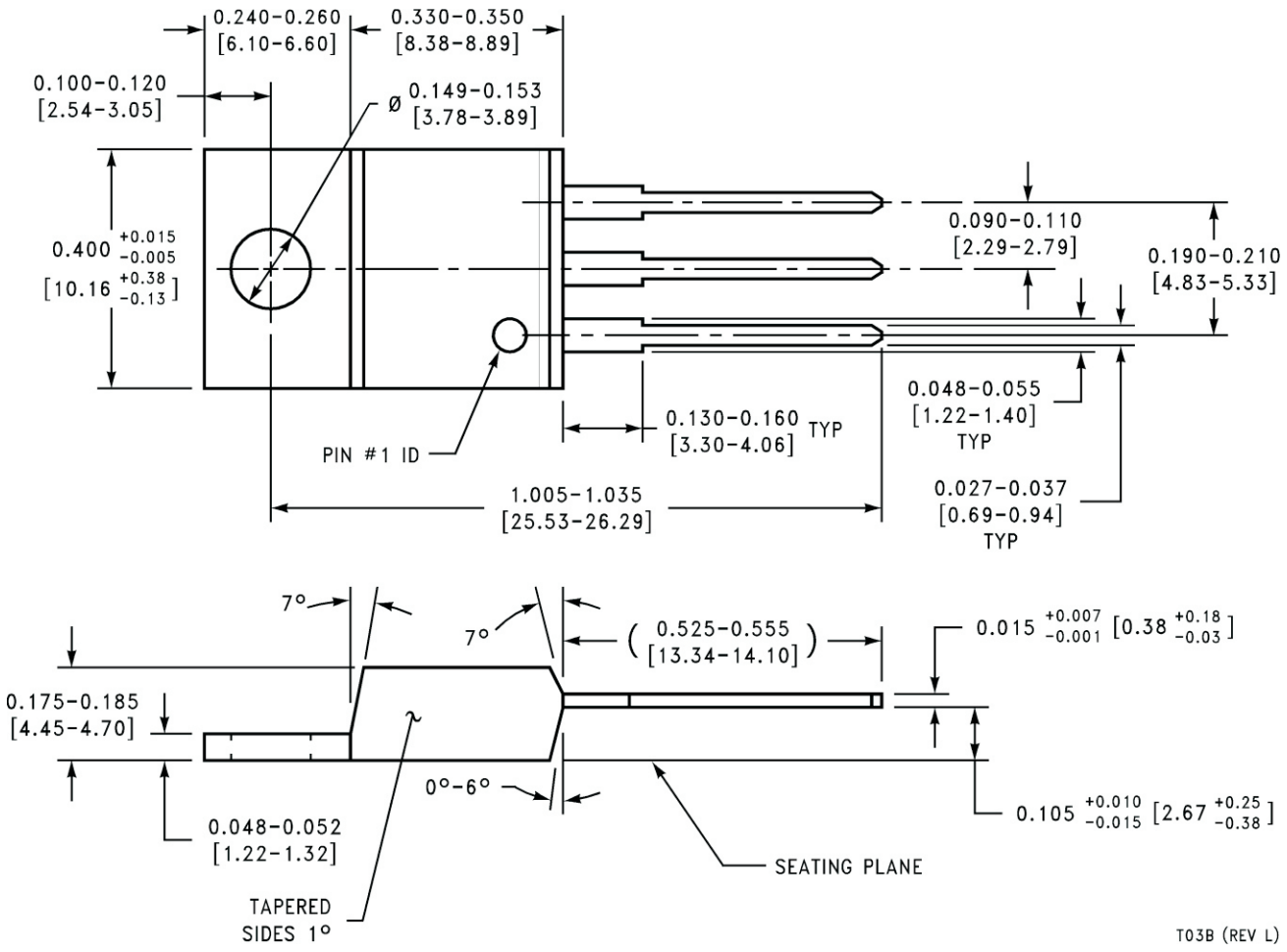
(5) Multiple Device Markings will be inside parentheses. Only one Device Marking contained in parentheses and separated by a "~" will appear on a device. If a line is indented then it is a continuation of the previous line and the two combined represent the entire Device Marking for that device.

(6) Lead/Ball Finish - Orderable Devices may have multiple material finish options. Finish options are separated by a vertical ruled line. Lead/Ball Finish values may wrap to two lines if the finish value exceeds the maximum column width.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

NDE0003B


0.240-0.260
[6.10-6.60]
0.330-0.350
[8.38-8.89]
0.100-0.120
[2.54-3.05]
Ø 0.149-0.153
[3.78-3.89]
0.400 +0.015 -0.005
[10.16 +0.38 -0.13]
0.090-0.110
[2.29-2.79]
0.190-0.210
[4.83-5.33]
PIN #1 ID
0.130-0.160 TYP
[3.30-4.06]
0.048-0.055
[1.22-1.40]
TYP
1.005-1.035
[25.53-26.29]
0.027-0.037
[0.69-0.94]
TYP
7°
7°
(0.525-0.555
[13.34-14.10])
0.015 +0.007 -0.001 [0.38 +0.18 -0.03]
0.175-0.185
[4.45-4.70]
0°-6°
0.105 +0.010 -0.015 [2.67 +0.25 -0.38]
0.048-0.052
[1.22-1.32]
SEATING PLANE
TAPERED
SIDES 1°
T03B (REV L)

IMPORTANT NOTICE

Texas Instruments Incorporated (TI) reserves the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

TI's published terms of sale for semiconductor products (http://www.ti.com/sc/docs/stdterms.htm) apply to the sale of packaged integrated circuit products that TI has qualified and released to market. Additional terms may apply to the use or sale of other types of TI products and services.

Reproduction of significant portions of TI information in TI data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such reproduced documentation. Information of third parties may be subject to additional restrictions. Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyers and others who are developing systems that incorporate TI products (collectively, "Designers") understand and agree that Designers remain responsible for using their independent analysis, evaluation and judgment in designing their applications and that Designers have full and exclusive responsibility to assure the safety of Designers' applications and compliance of their applications (and of all TI products used in or for Designers' applications) with all applicable regulations, laws and other applicable requirements. Designer represents that, with respect to their applications, Designer has all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. Designer agrees that prior to using or distributing any applications that include TI products, Designer will thoroughly test such applications and the functionality of such TI products as used in such applications.

TI's provision of technical, application or other design advice, quality characterization, reliability data or other services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using TI Resources in any way, Designer (individually or, if Designer is acting on behalf of a company, Designer's company) agrees to use any particular TI Resource solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

Designer is authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS. TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY DESIGNER AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Unless TI has explicitly designated an individual product as meeting the requirements of a particular industry standard (e.g., ISO/TS 16949 and ISO 26262), TI is not responsible for any failure to meet such industry standard requirements.

Where TI specifically promotes products as facilitating functional safety or as compliant with industry functional safety standards, such products are intended to help enable customers to design and create their own applications that meet applicable functional safety standards and requirements. Using products in an application does not by itself establish any safety features in the application. Designers must ensure compliance with safety-related requirements and standards applicable to their applications. Designer may not use any TI products in life-critical medical equipment unless authorized officers of the parties have executed a special contract specifically governing such use. Life-critical medical equipment is medical equipment where failure of such equipment would cause serious bodily injury or death (e.g., life support, pacemakers, defibrillators, heart pumps, neurostimulators, and implantables). Such equipment includes, without limitation, all medical devices identified by the U.S. Food and Drug Administration as Class III devices and equivalent classifications outside the U.S.

TI may expressly designate certain products as completing a particular qualification (e.g., Q100, Military Grade, or Enhanced Product). Designers agree that it has the necessary expertise to select the product with the appropriate qualification designation for their applications and that proper product selection is at Designers' own risk. Designers are solely responsible for compliance with all legal and regulatory requirements in connection with such selection.

Designer will fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of Designer's non-compliance with the terms and provisions of this Notice.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2017, Texas Instruments Incorporated

LMx58-N Low-Power, Dual-Operational Amplifiers

1 Features

- Available in 8-Bump DSBGA Chip-Sized Package, (See AN-1112, SNVA009)
- Internally Frequency Compensated for Unity Gain
- Large DC Voltage Gain: 100 dB
- Wide Bandwidth (Unity Gain): 1 MHz (Temperature Compensated)
- Wide Power Supply Range:
  - Single Supply: 3V to 32V
  - Or Dual Supplies: ±1.5V to ±16V
- Very Low Supply Current Drain (500 μA)—Essentially Independent of Supply Voltage
- Low Input Offset Voltage: 2 mV
- Input Common-Mode Voltage Range Includes Ground
- Differential Input Voltage Range Equal to the Power Supply Voltage
- Large Output Voltage Swing
- Unique Characteristics:
  - In the Linear Mode the Input Common-Mode Voltage Range Includes Ground and the Output Voltage Can Also Swing to Ground, even though Operated from Only a Single Power Supply Voltage.
  - The Unity Gain Cross Frequency is Temperature Compensated.
  - The Input Bias Current is also Temperature Compensated.
- Advantages:
  - Two Internally Compensated Op Amps
  - Eliminates Need for Dual Supplies
  - Allows Direct Sensing Near GND and V_{OUT} Also Goes to GND
  - Compatible with All Forms of Logic
  - Power Drain Suitable for Battery Operation

2 Applications

- Active Filters
- General Signal Conditioning and Amplification
- 4- to 20-mA Current Loop Transmitters

3 Description

The LM158 series consists of two independent, high gain, internally frequency compensated operational amplifiers which were designed specifically to operate from a single power supply over a wide range of voltages. Operation from split power supplies is also possible and the low power supply current drain is independent of the magnitude of the power supply voltage.

Application areas include transducer amplifiers, dc gain blocks and all the conventional op-amp circuits which now can be more easily implemented in single power supply systems. For example, the LM158 series can be directly operated off of the standard 3.3-V power supply voltage which is used in digital systems and will easily provide the required interface electronics without requiring the additional ±15V power supplies.

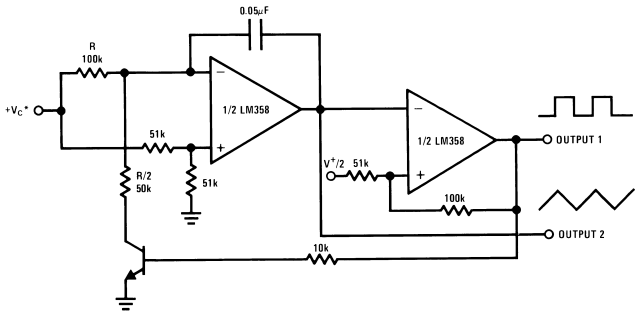
The LM358 and LM2904 are available in a chip sized package (8-Bump DSBGA) using TI's DSBGA package technology.

Device Information(1)

PART NUMBER	PACKAGE	BODY SIZE (NOM)
LM158-N	TO-CAN (8)	9.08 mm x 9.09 mm
	CDIP (8)	10.16 mm x 6.502 mm
LM258-N	TO-CAN (8)	9.08 mm x 9.09 mm
LM2904-N	DSBGA (8)	1.31 mm x 1.31 mm
	SOIC (8)	4.90 mm x 3.91 mm
	PDIP (8)	9.81 mm x 6.35 mm
LM358-N	TO-CAN (8)	9.08 mm x 9.09 mm
	DSBGA (8)	1.31 mm x 1.31 mm
	SOIC (8)	4.90 mm x 3.91 mm
	PDIP (8)	9.81 mm x 6.35 mm

(1) For all available packages, see the orderable addendum at the end of the datasheet.

Voltage Controlled Oscillator (VCO)



An IMPORTANT NOTICE at the end of this data sheet addresses availability, warranty, changes, use in safety-critical applications, intellectual property matters and other important disclaimers. PRODUCTION DATA.

Table of Contents

1 Features ... 1
2 Applications ... 1
3 Description ... 1
4 Revision History ... 2
5 Pin Configuration and Functions ... 3
6 Specifications ... 4
6.1 Absolute Maximum Ratings ... 4
6.2 ESD Ratings ... 4
6.3 Recommended Operating Conditions ... 5
6.4 Thermal Information ... 5
6.5 Electrical Characteristics: LM158A, LM358A, LM158, LM258 ... 5
6.6 Electrical Characteristics: LM358, LM2904 ... 7
6.7 Typical Characteristics ... 9
7 Detailed Description ... 12
7.1 Overview ... 12
7.2 Functional Block Diagram ... 12
7.3 Feature Description ... 12
7.4 Device Functional Modes ... 13
8 Application and Implementation ... 14
8.1 Application Information ... 14
8.2 Typical Applications ... 14
9 Power Supply Recommendations ... 24
10 Layout ... 24
10.1 Layout Guidelines ... 24
10.2 Layout Example ... 24
11 Device and Documentation Support ... 25
11.1 Related Links ... 25
11.2 Trademarks ... 25
11.3 Electrostatic Discharge Caution ... 25
11.4 Glossary ... 25
12 Mechanical, Packaging, and Orderable Information ... 25

4 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision H (March 2013) to Revision I — **Page**

- Added *Pin Configuration and Functions* section, *ESD Ratings* table, *Feature Description* section, *Device Functional Modes*, *Application and Implementation* section, *Power Supply Recommendations* section, *Layout* section, *Device and Documentation Support* section, and *Mechanical, Packaging, and Orderable Information* section ... 1

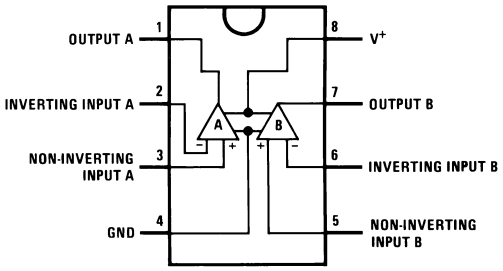
Changes from Revision G (March 2013) to Revision H — **Page**

- Changed layout of National Data Sheet to TI format ... 25

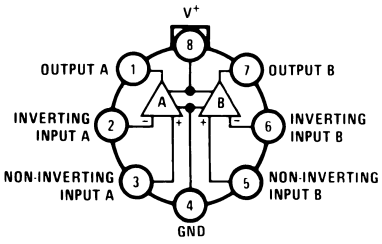
Copyright © 2000–2014, Texas Instruments Incorporated

5 Pin Configuration and Functions

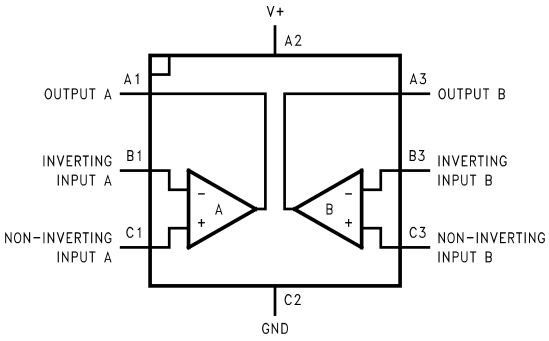
D, P, and NAB Package
8-Pin SOIC, PDIP, and CDIP
Top View



LMC Package
8-Pin TO-99
Top View



YPB Package
8-Pin DSBGA
Top View



Pin Functions

PIN			TYPE	DESCRIPTION
D/P/LMC NO.	DSBGA NO.	NAME		
1	A1	OUTA	O	Output , Channel A
2	B1	-INA	I	Inverting Input, Channel A
3	C1	+INA	I	Non-Inverting Input, Channel A
4	C2	GND / V-	P	Ground for Single supply configurations. negative supply for dual supply configurations
5	C3	+INB	I	Output, Channel B
6	B3	-INB	I	Inverting Input, Channel B
7	A3	OUTB	O	Non-Inverting Input, Channel B
8	A2	V+	P	Positive Supply

Copyright © 2000–2014, Texas Instruments Incorporated

6 Specifications

6.1 Absolute Maximum Ratings

See (1)(2)(3).

			LM158, LM258, LM358, LM158A, LM258A, LM358A		LM2904		UNIT
			MIN	MAX	MIN	MAX	
Supply Voltage, V^+				32		26	V
Differential Input Voltage				32		26	V
Input Voltage			−0.3	32	−0.3	26	V
Power Dissipation(4)	PDIP (P)			830		830	mW
	TO-99 (LMC)			550			mW
	SOIC (D)			530		530	mW
	DSBGA (YPB)			435			mW
Output Short-Circuit to GND (One Amplifier)(5)	$V^+ \le 15$ V and $T_A = 25°C$			Continuous		Continuous	
Input Current ($V_{IN} < -0.3V$)(6)				50		50	mA
Temperature			−55	125			°C
	PDIP Package (P): Soldering (10 seconds)			260		260	°C
	SOIC Package (D)	Vapor Phase (60 seconds)		215		215	°C
		Infrared (15 seconds)		220		220	°C
Lead Temperature	PDIP (P): (Soldering, 10 seconds)			260		260	°C
	TO-99 (LMC): (Soldering, 10 seconds)			300		300	°C
Storage temperature, T_{stg}			−65	150	−65	150	°C

(1) *Absolute Maximum Ratings* indicate limits beyond which damage to the device may occur. *Recommended Operating Conditions* indicate conditions for which the device is intended to be functional, but specific performance is not ensured. For ensured specifications and the test conditions, see the Electrical Characteristics.
(2) Refer to RETS158AX for LM158A military specifications and to RETS158X for LM158 military specifications.
(3) If Military/Aerospace specified devices are required, please contact the TI Sales Office/Distributors for availability and specifications.
(4) For operating at high temperatures, the LM358/LM358A, LM2904 must be derated based on a 125°C maximum junction temperature and a thermal resistance of 120°C/W for PDIP, 182°C/W for TO-99, 189°C/W for SOIC package, and 230°C/W for DSBGA, which applies for the device soldered in a printed circuit board, operating in a still air ambient. The LM258/LM258A and LM158/LM158A can be derated based on a +150°C maximum junction temperature. The dissipation is the total of both amplifiers—use external resistors, where possible, to allow the amplifier to saturate or to reduce the power which is dissipated in the integrated circuit.
(5) Short circuits from the output to V^+ can cause excessive heating and eventual destruction. When considering short circuits to ground, the maximum output current is approximately 40 mA independent of the magnitude of V^+. At values of supply voltage in excess of +15 V, continuous short-circuits can exceed the power dissipation ratings and cause eventual destruction. Destructive dissipation can result from simultaneous shorts on all amplifiers.
(6) This input current will only exist when the voltage at any of the input leads is driven negative. It is due to the collector-base junction of the input PNP transistors becoming forward biased and thereby acting as input diode clamps. In addition to this diode action, there is also lateral NPN parasitic transistor action on the IC chip. This transistor action can cause the output voltages of the op amps to go to the V^+voltage level (or to ground for a large overdrive) for the time duration that an input is driven negative. This is not destructive and normal output states will re-establish when the input voltage, which was negative, again returns to a value greater than −0.3 V (at 25°C).

6.2 ESD Ratings

			VALUE	UNIT
$V_{(ESD)}$	Electrostatic discharge	Human-body model (HBM), per ANSI/ESDA/JEDEC JS-001(1)	±250	V

(1) JEDEC document JEP155 states that 500-V HBM allows safe manufacturing with a standard ESD control process.

Copyright © 2000–2014, Texas Instruments Incorporated

6.3 Recommended Operating Conditions

over operating free-air temperature range (unless otherwise noted)

	MIN	MAX	UNIT
Supply Voltage (V+ - V-):LM158. LM258, LM358	3 (±1.5)	32 (±16)	V
Supply Voltage (V+ - V-):LM2904	3 (±1.5)	26 (±13)	V
Operating Temperature: LM158	-55	125	°C
Operating Temperature: LM258	-25	85	°C
Operating Temperature: LM358	0	70	°C
Operating Temperature: LM2904	-40	85	°C

6.4 Thermal Information

THERMAL METRIC(1)	LM158-N, LM258-N, LM358-N	LM158-N	LM2904-N, LM358-N			UNIT
	LMC	NAB	YPB	D	P	
	8 PINS					
$R_{\theta JA}$ Junction-to-ambient thermal resistance	155	132	230	189	120	°C/W

(1) For more information about traditional and new thermal metrics, see the *IC Package Thermal Metrics* application report, SPRA953.

6.5 Electrical Characteristics: LM158A, LM358A, LM158, LM258

V^+ = +5.0 V, See(1), unless otherwise stated

PARAMETER	TEST CONDITIONS	LM158A			LM358A			LM158, LM258			UNIT
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	
Input Offset Voltage	See(2), T_A = 25°C		1	2		2	3		2	5	mV
Input Bias Current	$I_{IN(+)}$ or $I_{IN(-)}$, T_A = 25°C,		20	50		45	100		45	150	nA
	V_{CM} = 0 V,(3)										
Input Offset Current	$I_{IN(+)} - I_{IN(-)}$, V_{CM} = 0V, T_A = 25°C		2	10		5	30		3	30	nA
Input Common-Mode Voltage Range	V^+ = 30 V,(4) (LM2904, V^+ = 26V), T_A = 25°C	0		V^+−1.5	0		V^+−1.5	0		V^+−1.5	V
Supply Current	Over Full Temperature Range R_L = ∞ on All Op Amps										
	V^+ = 30V (LM2904 V^+ = 26V)		1	2		1	2		1	2	mA
	V^+ = 5V		0.5	1.2		0.5	1.2		0.5	1.2	mA
Large Signal Voltage Gain	V^+ = 15 V, T_A = 25°C, R_L ≥ 2 kΩ, (For V_O = 1 V to 11 V)	50	100		25	100		50	100		V/mV
Common-Mode Rejection Ratio	T_A = 25°C, V_{CM} = 0 V to V^+−1.5 V	70	85		65	85		70	85		dB
Power Supply Rejection Ratio	V^+ = 5 V to 30 V (LM2904, V^+ = 5 V to 26 V), T_A = 25°C	65	100		65	100		65	100		dB

(1) These specifications are limited to –55°C ≤ T_A ≤ +125°C for the LM158/LM158A. With the LM258/LM258A, all temperature specifications are limited to −25°C ≤ T_A ≤ 85°C, the LM358/LM358A temperature specifications are limited to 0°C ≤ T_A ≤ 70°C, and the LM2904 specifications are limited to –40°C ≤ T_A ≤ 85°C.
(2) V_O ≅ 1.4 V, R_S = 0 Ω with V^+ from 5 V to 30 V; and over the full input common-mode range (0 V to V^+ −1.5 V) at 25°C. For LM2904, V^+ from 5 V to 26 V.
(3) The direction of the input current is out of the IC due to the PNP input stage. This current is essentially constant, independent of the state of the output so no loading change exists on the input lines.
(4) The input common-mode voltage of either input signal voltage should not be allowed to go negative by more than 0.3 V (at 25°C). The upper end of the common-mode voltage range is V^+ −1.5 V (at 25°C), but either or both inputs can go to 32 V without damage (26 V for LM2904), independent of the magnitude of V^+.

Copyright © 2000–2014, Texas Instruments Incorporated

Electrical Characteristics: LM158A, LM358A, LM158, LM258 (continued)

V^+ = +5.0 V, See(1), unless otherwise stated

PARAMETER		TEST CONDITIONS		LM158A			LM358A			LM158, LM258			UNIT
				MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	
Power Supply		V^+ = 5 V to 30 V		65	100		65	100		65	100		dB
Rejection Ratio		(LM2904, V^+ = 5 V to 26 V), T_A = 25°C											
Amplifier-to-Amplifier Coupling		f = 1 kHz to 20 kHz, T_A = 25°C (Input Referred), See(5)			−120			−120			−120		dB
Output Current	Source	V_{IN}^+ = 1 V,		20	40		20	40		20	40		mA
		V_{IN}^- = 0 V,											
		V^+ = 15 V,											
		V_O = 2 V, T_A = 25°C											
	Sink	V_{IN}^- = 1 V, V_{IN}^+ = 0 V		10	20		10	20		10	20		mA
		V^+ = 15 V, T_A = 25°C,											
		V_O = 2 V											
		V_{IN}^- = 1 V,		12	50		12	50		12	50		µA
		V_{IN}^+ = 0 V											
		T_A = 25°C, V_O = 200 mV,											
		V^+ = 15 V											
Short Circuit to Ground		T_A = 25°C, See(6), V^+ = 15 V			40	60		40	60		40	60	mA
Input Offset Voltage		See(2)				4			5			7	mV
Input Offset Voltage Drift		R_S = 0Ω			7	15		7	20		7		µV/°C
Input Offset Current		$I_{IN(+)} - I_{IN(-)}$				30			75			100	nA
Input Offset Current Drift		R_S = 0Ω			10	200		10	300		10		pA/°C
Input Bias Current		$I_{IN(+)}$ or $I_{IN(-)}$			40	100		40	200		40	300	nA
Input Common-Mode Voltage Range		V^+ = 30 V, See(4) (LM2904, V^+ = 26 V)		0		V^+−2	0		V^+−2	0		V^+−2	V
Large Signal Voltage Gain		V^+ = +15 V		25			15			25			V/mV
		(V_O = 1 V to 11 V)											
		R_L ≥ 2 kΩ											
Output Voltage	V_{OH}	V^+ = +30 V	R_L = 2 kΩ	26			26			26			V
		(LM2904, V^+ = 26 V)	R_L = 10 kΩ	27	28		27	28		27	28		V
Swing	V_{OL}	V^+ = 5V, R_L = 10 kΩ			5	20		5	20		5	20	mV
Output Current	Source	V_{IN}^+ = +1 V, V_{IN}^- = 0 V,		10	20		10	20		10	20		mA
		V^+ = 15 V, V_O = 2 V											
	Sink	V_{IN}^- = +1 V, V_{IN}^+ = 0 V,		10	15		5	8		5	8		mA
		V^+ = 15 V, V_O = 2 V											

(5) Due to proximity of external components, insure that coupling is not originating via stray capacitance between these external parts. This typically can be detected as this type of capacitance increases at higher frequencies.

(6) Short circuits from the output to V^+ can cause excessive heating and eventual destruction. When considering short circuits to ground, the maximum output current is approximately 40 mA independent of the magnitude of V^+. At values of supply voltage in excess of +15 V, continuous short-circuits can exceed the power dissipation ratings and cause eventual destruction. Destructive dissipation can result from simultaneous shorts on all amplifiers.

Copyright © 2000–2014, Texas Instruments Incorporated

6.6 Electrical Characteristics: LM358, LM2904

V^+ = +5.0 V, See(1), unless otherwise stated

PARAMETER		TEST CONDITIONS	LM358			LM2904			UNIT
			MIN	TYP	MAX	MIN	TYP	MAX	
Input Offset Voltage		See(2) , T_A = 25°C		2	7		2	7	mV
Input Bias Current		$I_{IN(+)}$ or $I_{IN(-)}$, T_A = 25°C, V_{CM} = 0 V, See(3)		45	250		45	250	nA
Input Offset Current		$I_{IN(+)} - I_{IN(-)}$, V_{CM} = 0 V, T_A = 25°C		5	50		5	50	nA
Input Common-Mode Voltage Range		V^+ = 30 V, See(4) (LM2904, V^+ = 26 V), T_A = 25°C	0		$V^+-1.5$	0		$V^+-1.5$	V
Supply Current		Over Full Temperature Range							
		R_L = ∞ on All Op Amps							
		V^+ = 30 V (LM2904 V^+ = 26 V)		1	2		1	2	mA
		V^+ = 5 V		0.5	1.2		0.5	1.2	mA
Large Signal Voltage		V^+ = 15V, T_A = 25°C,							
Gain		R_L ≥ 2 kΩ, (For V_O = 1 V to 11 V)	25	100		25	100		V/mV
Common-Mode Rejection Ratio		T_A = 25°C,	65	85		50	70		dB
		V_{CM} = 0 V to $V^+-1.5$ V							
Power Supply Rejection Ratio		V^+ = 5 V to 30 V	65	100		50	100		dB
		(LM2904, V^+ = 5 V to 26 V), T_A = 25°C							
Amplifier-to-Amplifier Coupling		f = 1 kHz to 20 kHz, T_A = 25°C (Input Referred), See(5)		−120			−120		dB
Output Current	Source	V_{IN}^+ = 1 V,	20	40		20	40		mA
		V_{IN}^- = 0 V,							
		V^+ = 15 V,							
		V_O = 2 V, T_A = 25°C							
	Sink	V_{IN}^- = 1 V, V_{IN}^+ = 0 V	10	20		10	20		mA
		V^+ = 15V, T_A = 25°C,							
		V_O = 2 V							
		V_{IN}^- = 1 V,	12	50		12	50		µA
		V_{IN}^+ = 0 V							
		T_A = 25°C, V_O = 200 mV,							
		V^+ = 15 V							
Short Circuit to Ground		T_A = 25°C, See(6), V^+ = 15 V		40	60		40	60	mA
Input Offset Voltage		See(2)			9			10	mV
Input Offset Voltage Drift		R_S = 0 Ω		7			7		µV/°C
Input Offset Current		$I_{IN(+)} - I_{IN(-)}$			150		45	200	nA
Input Offset Current Drift		R_S = 0 Ω		10			10		pA/°C
Input Bias Current		$I_{IN(+)}$ or $I_{IN(-)}$		40	500		40	500	nA

(1) These specifications are limited to –55°C ≤ T_A ≤ +125°C for the LM158/LM158A. With the LM258/LM258A, all temperature specifications are limited to −25°C ≤ T_A ≤ 85°C, the LM358/LM358A temperature specifications are limited to 0°C ≤ T_A ≤ 70°C, and the LM2904 specifications are limited to –40°C ≤ T_A ≤ 85°C.

(2) V_O ≅ 1.4 V, R_S = 0 Ω with V^+ from 5 V to 30 V; and over the full input common-mode range (0 V to V^+ −1.5 V) at 25°C. For LM2904, V^+ from 5 V to 26 V.

(3) The direction of the input current is out of the IC due to the PNP input stage. This current is essentially constant, independent of the state of the output so no loading change exists on the input lines.

(4) The input common-mode voltage of either input signal voltage should not be allowed to go negative by more than 0.3 V (at 25°C). The upper end of the common-mode voltage range is V^+ −1.5 V (at 25°C), but either or both inputs can go to 32 V without damage (26 V for LM2904), independent of the magnitude of V^+.

(5) Due to proximity of external components, insure that coupling is not originating via stray capacitance between these external parts. This typically can be detected as this type of capacitance increases at higher frequencies.

(6) Short circuits from the output to V^+ can cause excessive heating and eventual destruction. When considering short circuits to ground, the maximum output current is approximately 40 mA independent of the magnitude of V^+. At values of supply voltage in excess of +15 V, continuous short-circuits can exceed the power dissipation ratings and cause eventual destruction. Destructive dissipation can result from simultaneous shorts on all amplifiers.

Copyright © 2000–2014, Texas Instruments Incorporated

Electrical Characteristics: LM358, LM2904 (continued)

V^+ = +5.0 V, See(1), unless otherwise stated

PARAMETER		TEST CONDITIONS		LM358			LM2904			UNIT
				MIN	TYP	MAX	MIN	TYP	MAX	
Input Common-Mode Voltage Range		V^+ = 30 V, See(4) (LM2904, V^+ = 26 V)		0		V^+−2	0		V^+ −2	V
Large Signal Voltage Gain		V^+ = +15 V		15			15			V/mV
		(V_O = 1 V to 11 V)								
		R_L ≥ 2 kΩ								
Output	V_{OH}	V^+ = 30 V	R_L = 2 kΩ	26			22			V
Voltage		(LM2904, V^+ = 26 V)	R_L = 10 kΩ	27	28		23	24		V
Swing	V_{OL}	V^+ = 5 V, R_L = 10 kΩ			5	20		5	100	mV
Output Current	Source	V_{IN}^+ = 1 V, V_{IN}^- = 0 V,		10	20		10	20		mA
		V^+ = 15 V, V_O = 2 V								
	Sink	V_{IN}^- = 1 V, V_{IN}^+ = 0 V,		5	8		5	8		mA
		V^+ = 15 V, V_O = 2 V								

Copyright © 2000–2014, Texas Instruments Incorporated

6.7 Typical Characteristics

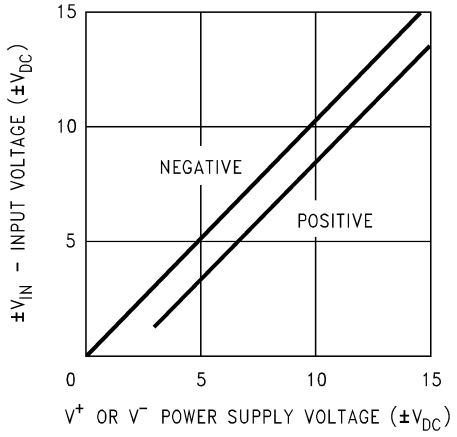


Figure 1. Input Voltage Range

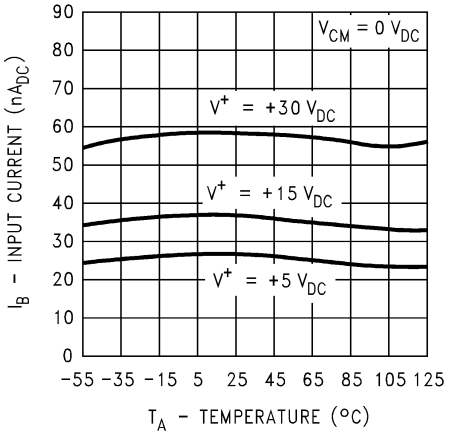


Figure 2. Input Current

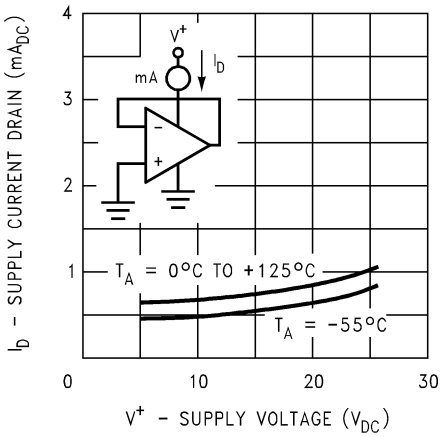


Figure 3. Supply Current

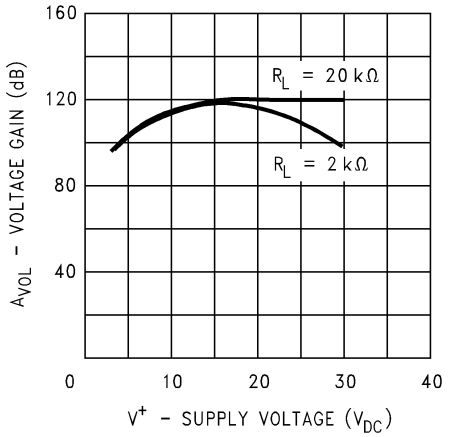


Figure 4. Voltage Gain

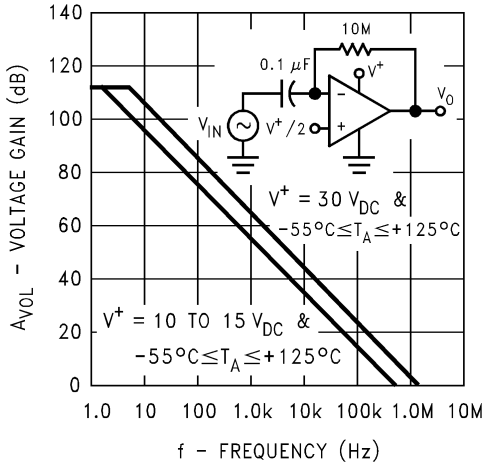


Figure 5. Open Loop Frequency Response

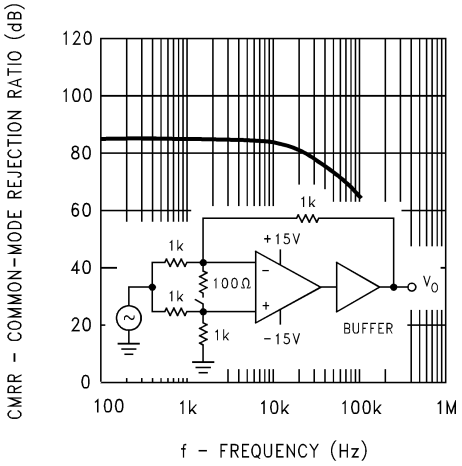


Figure 6. Common-Mode Rejection Ratio

Copyright © 2000–2014, Texas Instruments Incorporated

Typical Characteristics (continued)

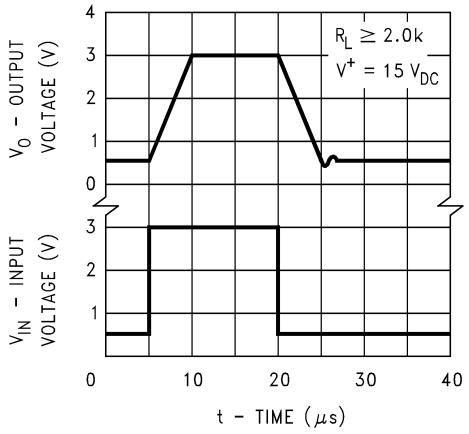


Figure 7. Voltage Follower Pulse Response

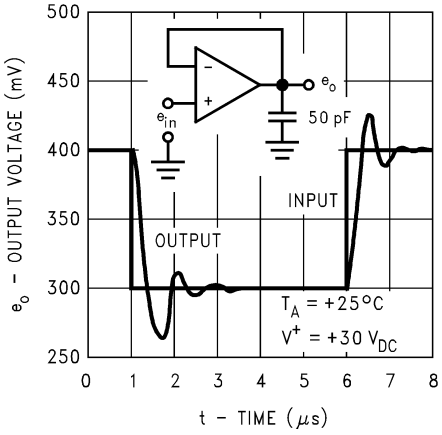


Figure 8. Voltage Follower Pulse Response (Small Signal)

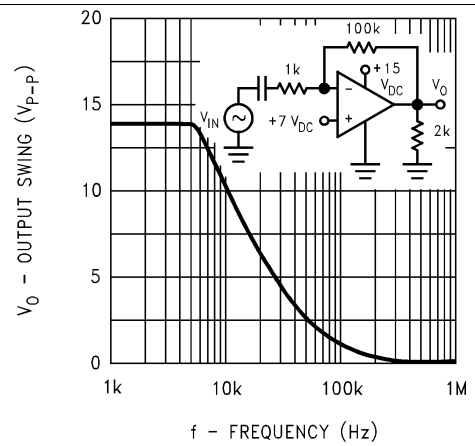


Figure 9. Large Signal Frequency Response

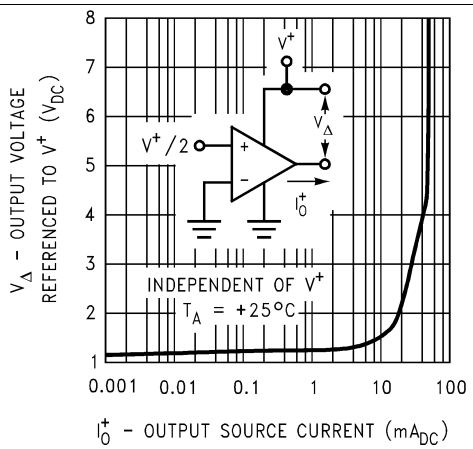


Figure 10. Output Characteristics Current Sourcing

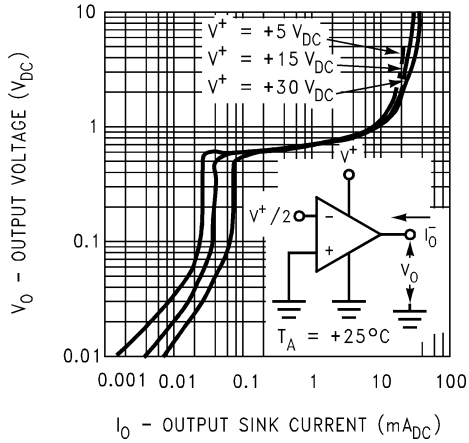


Figure 11. Output Characteristics Current Sinking

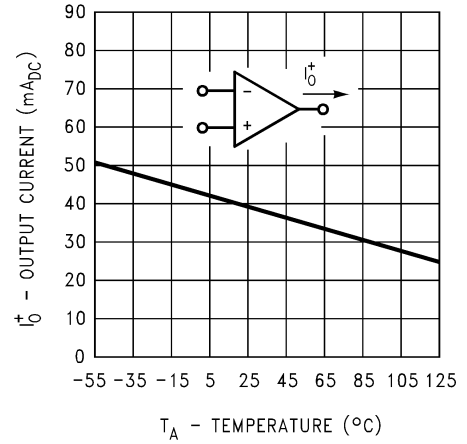


Figure 12. Current Limiting

Copyright © 2000–2014, Texas Instruments Incorporated

Typical Characteristics (continued)

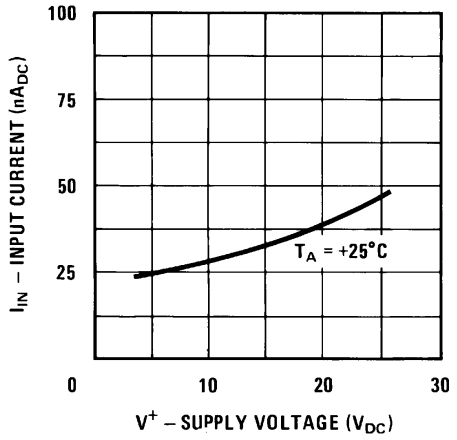


Figure 13. Input Current (LM2902 Only)

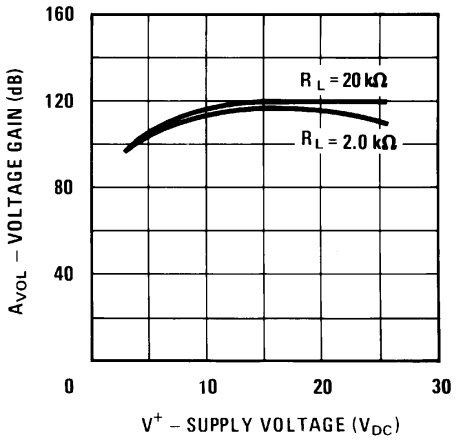


Figure 14. Voltage Gain (LM2902 Only)

Copyright © 2000–2014, Texas Instruments Incorporated

7 Detailed Description

7.1 Overview

The LM158 series are operational amplifiers which can operate with only a single power supply voltage, have true-differential inputs, and remain in the linear mode with an input common-mode voltage of 0 V_{DC}. These amplifiers operate over a wide range of power supply voltage with little change in performance characteristics. At 25°C amplifier operation is possible down to a minimum supply voltage of 2.3 V_{DC}.

Large differential input voltages can be easily accommodated and, as input differential voltage protection diodes are not needed, no large input currents result from large differential input voltages. The differential input voltage may be larger than V^+ without damaging the device. Protection should be provided to prevent the input voltages from going negative more than −0.3 V_{DC} (at 25°C). An input clamp diode with a resistor to the IC input terminal can be used.

7.2 Functional Block Diagram

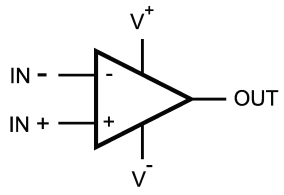


Figure 15. (Each Amplifier)

7.3 Feature Description

The amplifier's differential inputs consist of a non-inverting input (+IN) and an inverting input (–IN). The amplifer amplifies only the difference in voltage between the two inpus, which is called the differential input voltage. The output voltage of the op-amp Vout is given by Equation 1:

VOUT = AOL (IN+ - IN-)

where

- AOL is the open-loop gain of the amplifier, typically around 100dB (100,000x, or 10uV per Volt). (1)

To reduce the power supply current drain, the amplifiers have a class A output stage for small signal levels which converts to class B in a large signal mode. This allows the amplifiers to both source and sink large output currents. Therefore both NPN and PNP external current boost transistors can be used to extend the power capability of the basic amplifiers. The output voltage needs to raise approximately 1 diode drop above ground to bias the on-chip vertical PNP transistor for output current sinking applications.

For ac applications, where the load is capacitively coupled to the output of the amplifier, a resistor should be used, from the output of the amplifier to ground to increase the class A bias current and prevent crossover distortion. Where the load is directly coupled, as in dc applications, there is no crossover distortion.

Capacitive loads which are applied directly to the output of the amplifier reduce the loop stability margin. Values of 50 pF can be accommodated using the worst-case non-inverting unity gain connection. Large closed loop gains or resistive isolation should be used if larger load capacitance must be driven by the amplifier.

The bias network of the LM158 establishes a drain current which is independent of the magnitude of the power supply voltage over the range of 3 V_{DC} to 30 V_{DC}.

Output short circuits either to ground or to the positive power supply should be of short time duration. Units can be destroyed, not as a result of the short circuit current causing metal fusing, but rather due to the large increase in IC chip power dissipation which will cause eventual failure due to excessive junction temperatures. Putting direct short-circuits on more than one amplifier at a time will increase the total IC power dissipation to destructive levels, if not properly protected with external dissipation limiting resistors in series with the output leads of the amplifiers. The larger value of output source current which is available at 25°C provides a larger output current capability at elevated temperatures (see *Typical Characteristics*) than a standard IC op amp.

Copyright © 2000–2014, Texas Instruments Incorporated

7.4 Device Functional Modes

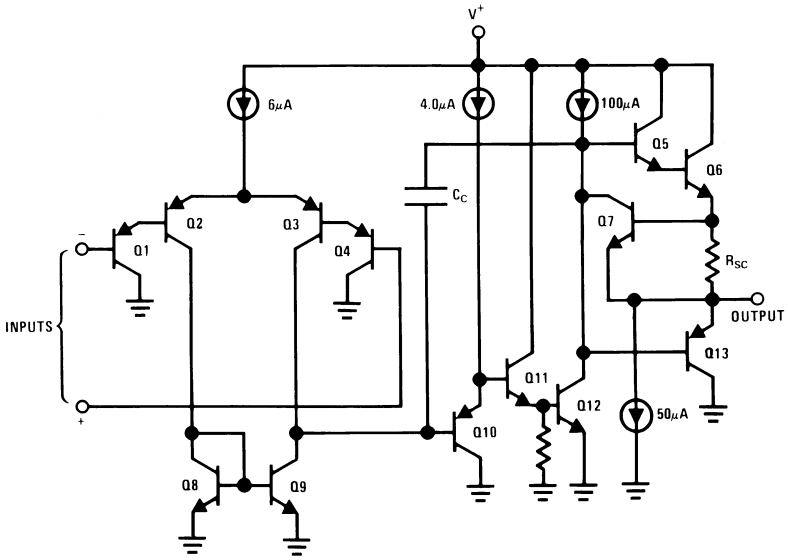


Figure 16. Schematic Diagram

The circuits presented in the *Typical Single-Supply Applications* emphasize operation on only a single power supply voltage. If complementary power supplies are available, all of the standard op-amp circuits can be used. In general, introducing a pseudo-ground (a bias voltage reference of $V^+/2$) will allow operation above and below this value in single power supply systems. Many application circuits are shown which take advantage of the wide input common-mode voltage range which includes ground. In most cases, input biasing is not required and input voltages which range to ground can easily be accommodated.

Copyright © 2000–2014, Texas Instruments Incorporated

8 Application and Implementation

NOTE

Information in the following applications sections is not part of the TI component specification, and TI does not warrant its accuracy or completeness. TI's customers are responsible for determining suitability of components for their purposes. Customers should validate and test their design implementation to confirm system functionality.

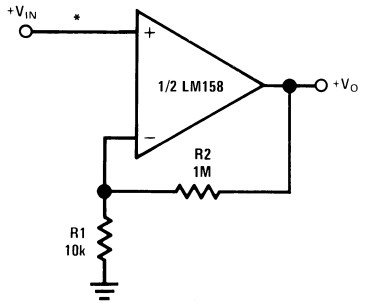
8.1 Application Information

The LM158 family bring performance, economy, and ease-of-use to a wide variety of op-amp applications.

8.2 Typical Applications

8.2.1 Noninverting DC Gain

Figure 17 shows a high input impedance non-inverting circuit. This circuit gives a closed-loop gain equal to the ratio of the sum of R1 and R2 to R1 and a closed-loop 3 dB bandwidth equal to the amplifier unity-gain frequency divided by the closed-loop gain. This design has the benefit of a very high input impedance, which is equal to the differential input impedance multiplied by loop gain. (Open loop gain/Closed loop gain.) In DC coupled applications, input impedance is not as important as input current and its voltage drop across the source resistance. Note that the amplifier output will go into saturation if the input is allowed to float. This may be important if the amplifier must be switched from source to source.



*R not needed due to temperature independent I_{IN}

Figure 17. Non-Inverting DC Gain (0-V Output)

8.2.1.1 Design Requirements

For this example application, the supply voltage is +5V, and 100x±5% of noninverting gain is necessary. Signal input impedance is approx 10kΩ.

8.2.1.2 Detailed Design Procedure

Using the equation for a non-inverting amplifier configuration ; G = 1+ R2/R1, set R1 to 10kΩ, and R2 to 99x the value of R1, which would be 990kΩ. Replacing the 990kΩ with a 1MΩ will result in a gain of 101, which is within the desired gain tolerance.

The gain-frequency characteristic of the amplifier and its feedback network must be such that oscillation does not occur. To meet this condition, the phase shift through amplifier and feedback network must never exceed 180° for any frequency where the gain of the amplifier and its feedback network is greater than unity. In practical applications, the phase shift should not approach 180° since this is the situation of conditional stability. Obviously the most critical case occurs when the attenuation of the feedback network is zero.

Copyright © 2000–2014, Texas Instruments Incorporated

Typical Applications (continued)

8.2.1.3 Application Curve

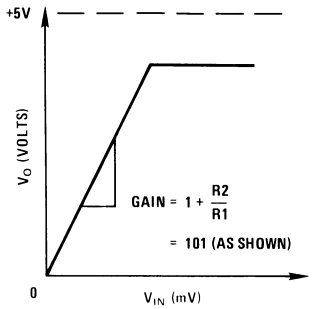
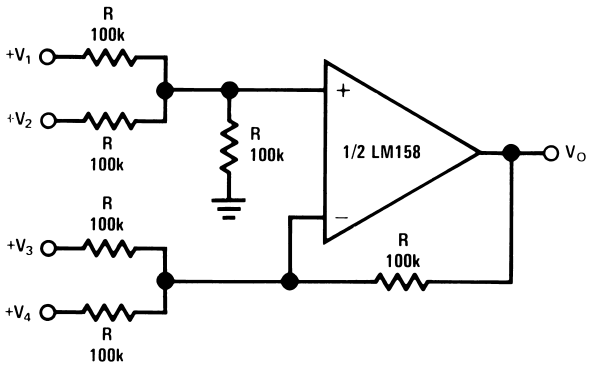


Figure 18. Transfer Curve for Non-Inverting Configuration

8.2.2 System Examples

8.2.2.1 Typical Single-Supply Applications

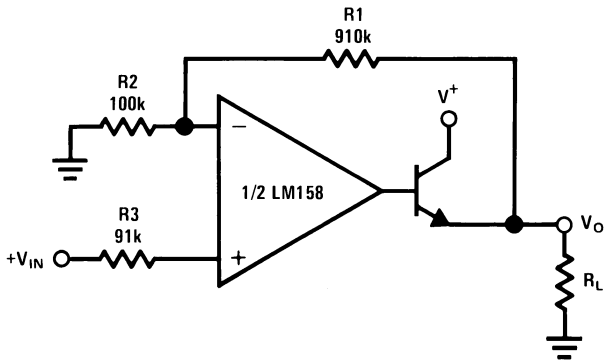
(V^+ = 5.0 V_{DC})



Where: $V_O = V_1 + V_2 - V_3 - V_4$

$(V_1 + V_2) \geq (V_3 + V_4)$ to keep $V_O > 0$ V_{DC}

Figure 19. DC Summing Amplifier ($V_{IN'S} \geq 0$ V_{DC} and $V_O \geq 0$ V_{DC})



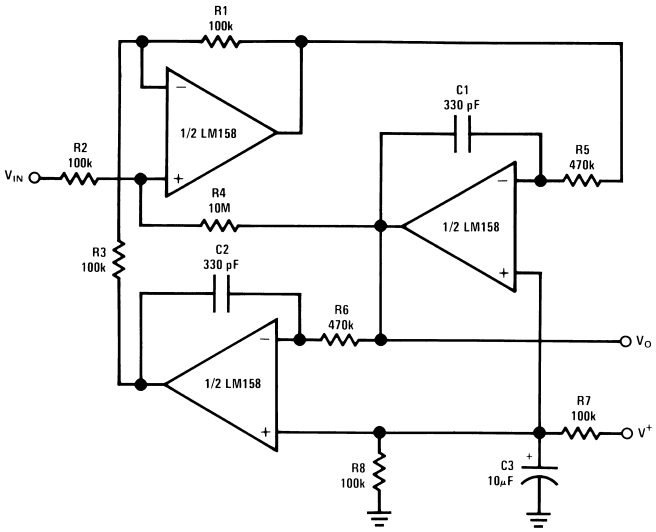
$V_O = 0$ V_{DC} for $V_{IN} = 0$ V_{DC}

$A_V = 10$

Figure 20. Power Amplifier

Typical Applications (continued)

(V^+ = 5.0 V_{DC})



f_o = 1 kHz

Q = 50

A_v = 100 (40 dB)

Figure 21. "BI-QUAD" RC Active Bandpass Filter

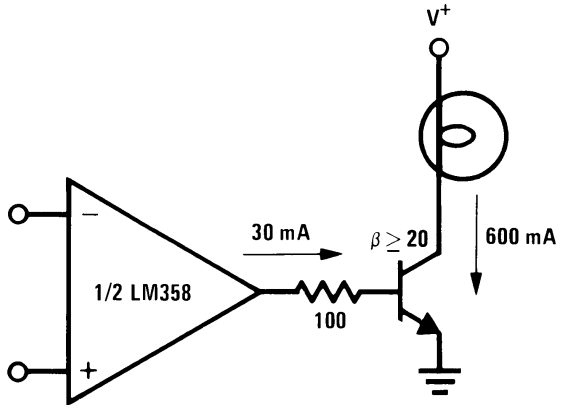


Figure 22. Lamp Driver

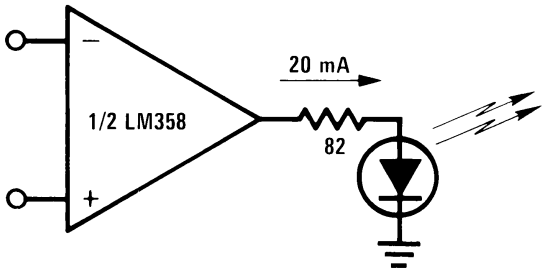


Figure 23. LED Driver

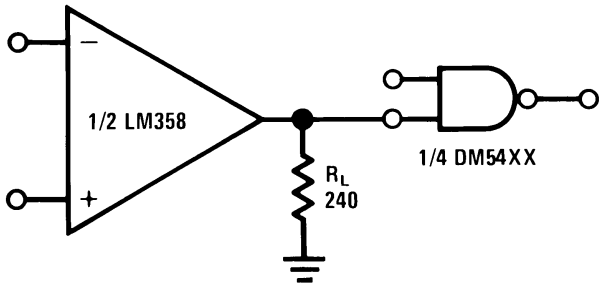
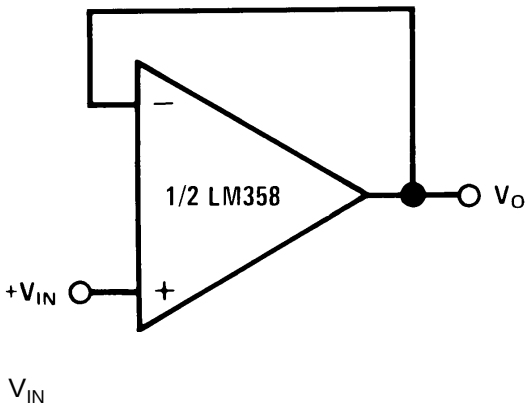


Figure 24. Driving TTL

Copyright © 2000–2014, Texas Instruments Incorporated

Typical Applications (continued)

(V^+ = 5.0 V_{DC})



$V_O = V_{IN}$

Figure 25. Voltage Follower

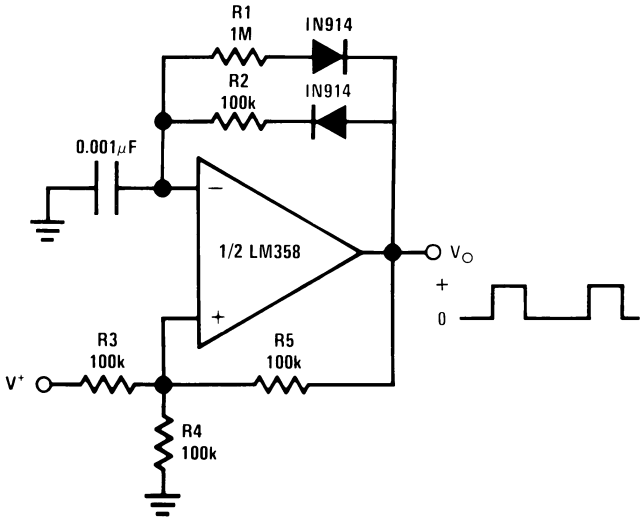


Figure 26. Pulse Generator

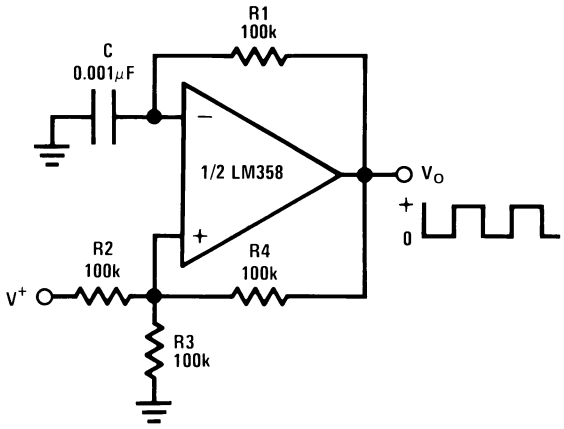


Figure 27. Squarewave Oscillator

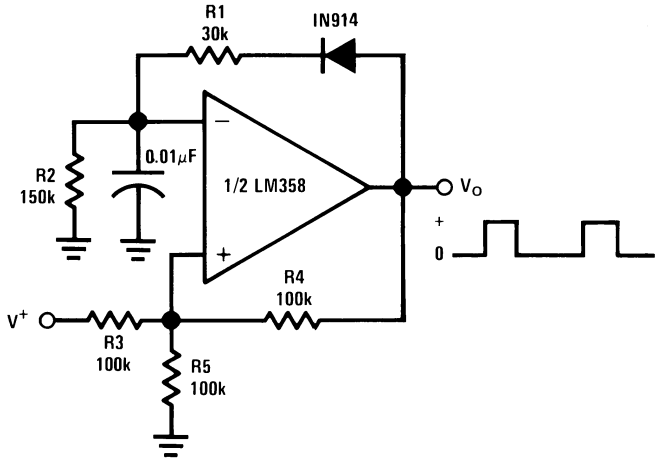
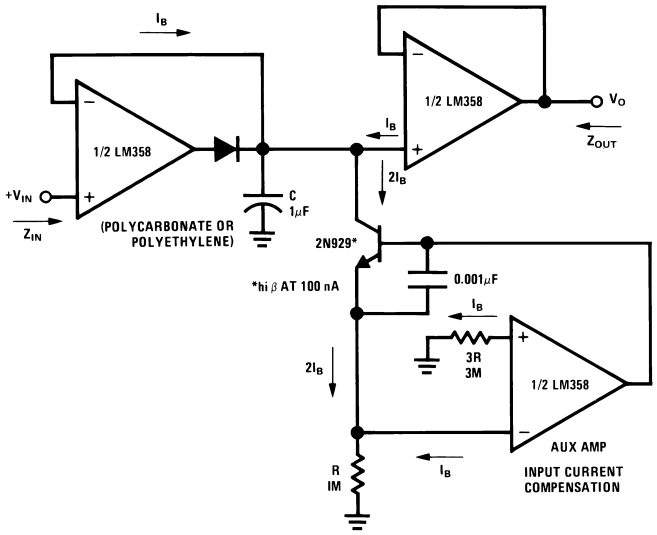


Figure 28. Pulse Generator

Typical Applications (continued)

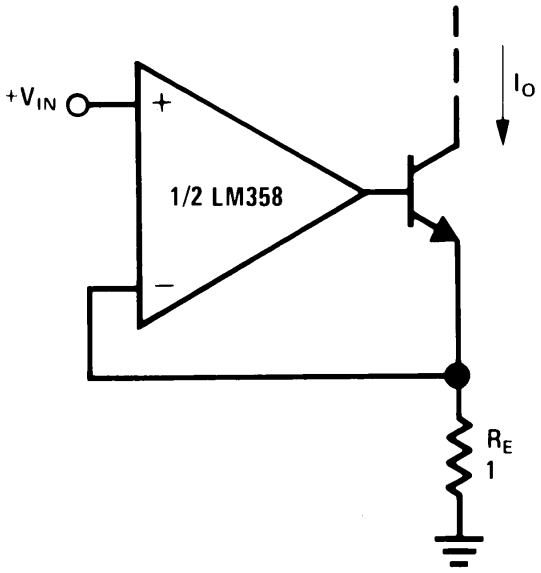
(V^+ = 5.0 V_{DC})



HIGH Z_{IN}

LOW Z_{OUT}

Figure 29. Low Drift Peak Detector



I_O = 1 amp/volt V_{IN}

(Increase R_E for I_O small)

Figure 30. High Compliance Current Sink

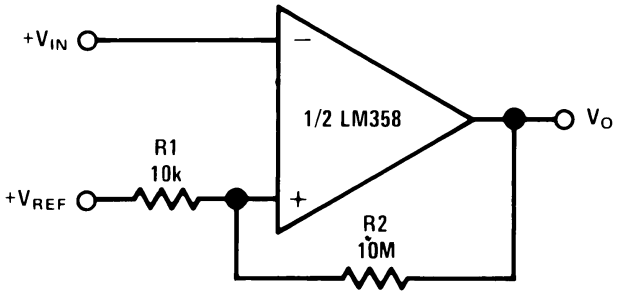
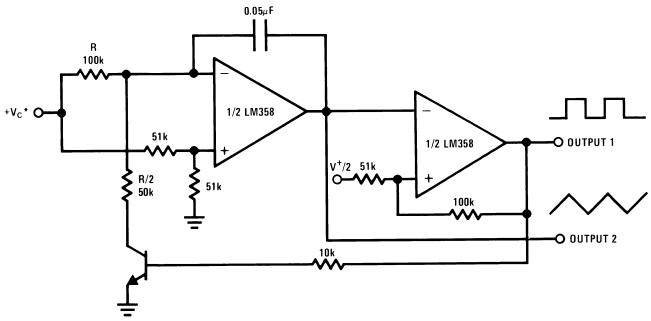


Figure 31. Comparator with Hysteresis



*WIDE CONTROL VOLTAGE RANGE: 0 V_{DC} ≤ V_C ≤ 2 (V^+ −1.5V_{DC})

Figure 32. Voltage Controlled Oscillator (VCO)

Copyright © 2000–2014, Texas Instruments Incorporated

Typical Applications (continued)

(V^+ = 5.0 V_{DC})

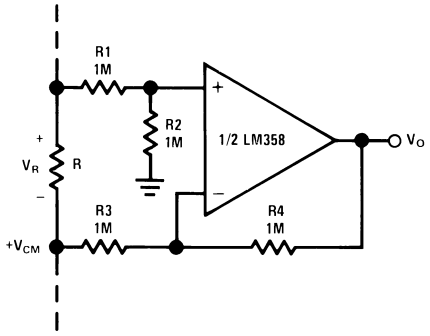
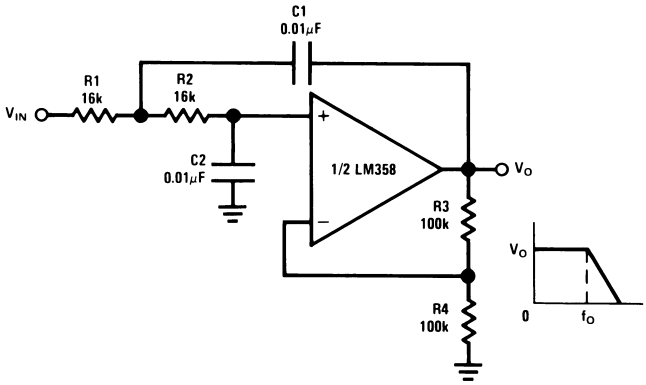


Figure 33. Ground Referencing a Differential Input Signal

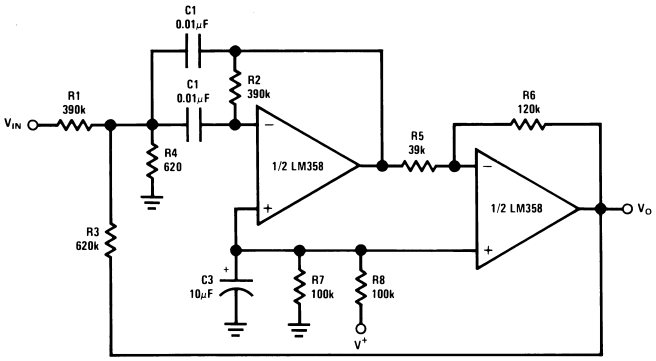


f_o = 1 kHz

Q = 1

A_V = 2

Figure 34. DC Coupled Low-Pass RC Active Filter



f_o = 1 kHz

Q = 25

Figure 35. Bandpass Active Filter

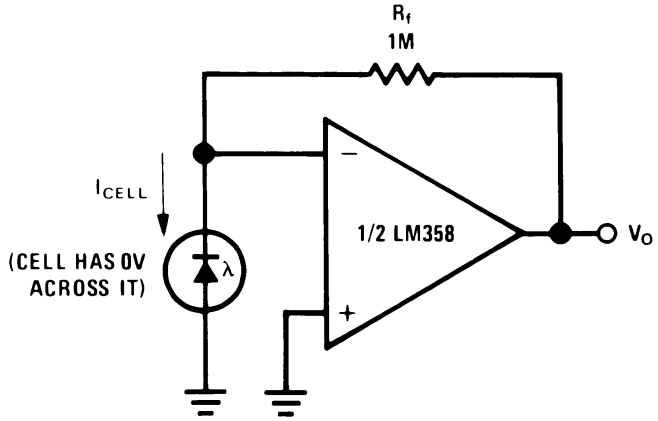


Figure 36. Photo Voltaic-Cell Amplifier

Typical Applications (continued)

(V^+ = 5.0 V_{DC})

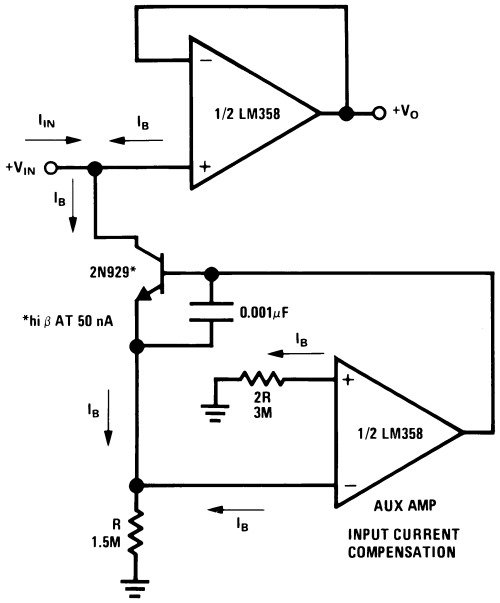
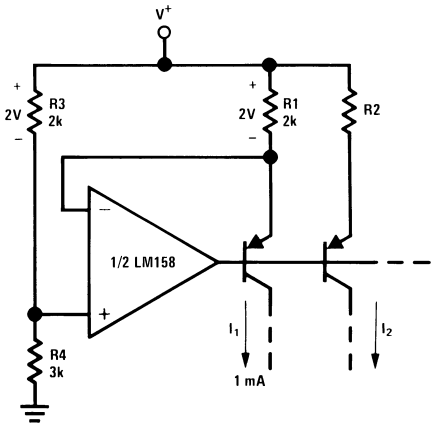


Figure 37. Using Symmetrical Amplifiers to Reduce Input Current (General Concept)



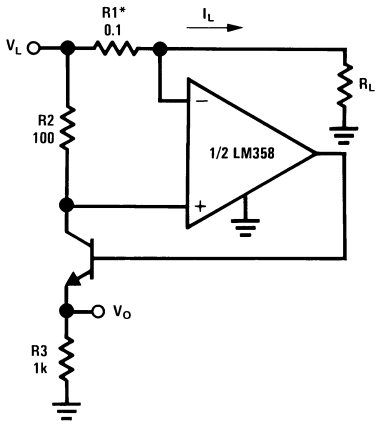
$$I_2 = \left(\frac{R1}{R2}\right) I_1$$

Figure 38. Fixed Current Sources

Copyright © 2000–2014, Texas Instruments Incorporated

Typical Applications (continued)

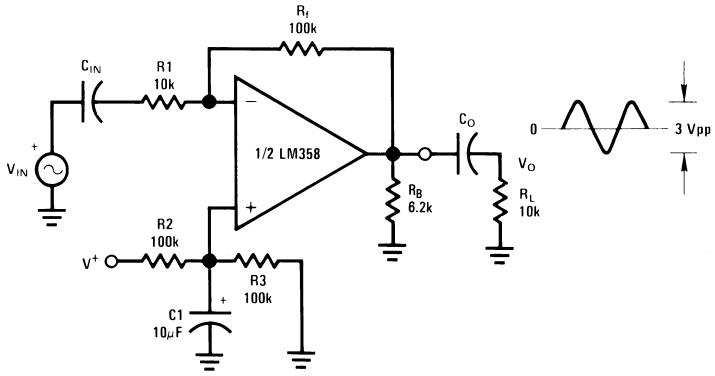
(V^+ = 5.0 V_{DC})



$$V_O = \frac{1V\ (I_L)}{1A}$$

*(Increase R1 for I_L small)
$V_L \leq V^+ - 2V$

Figure 39. Current Monitor

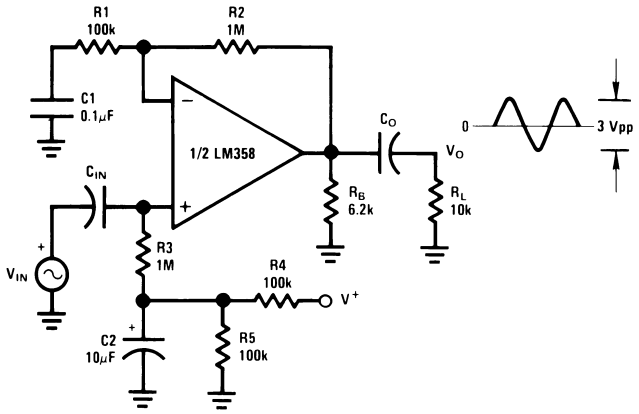


$$A_V = \frac{R_f}{R1} \text{ (As shown, } A_V = 10)$$

Figure 40. AC Coupled Inverting Amplifier

Typical Applications (continued)

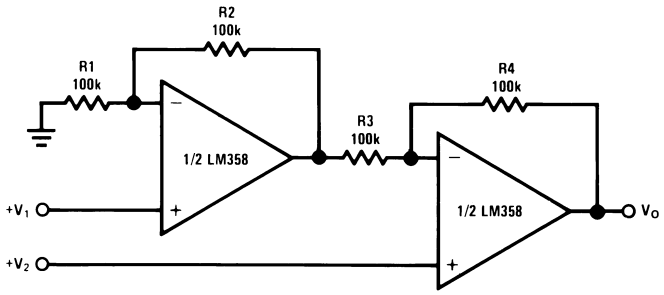
(V^+ = 5.0 V_{DC})



$$A_V = 1 + \frac{R2}{R1}$$

A_v = 11 (As Shown)

Figure 41. AC Coupled Non-Inverting Amplifier

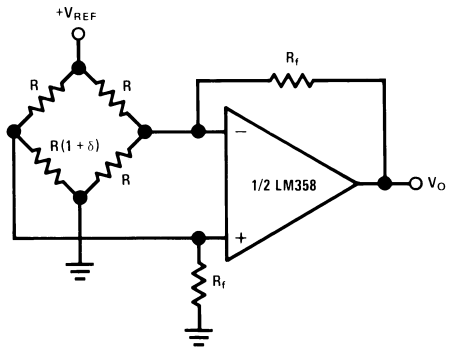


For $\frac{R1}{R2} = \frac{R4}{R3}$ (CMRR depends on this resistor ratio match)

$$V_O = 1 + \frac{R4}{R3}(V_2 - V_1)$$

As Shown: $V_O = 2(V_2 - V_1)$

Figure 42. High Input Z, DC Differential Amplifier



For $\delta << 1$ and $R_f >> R$

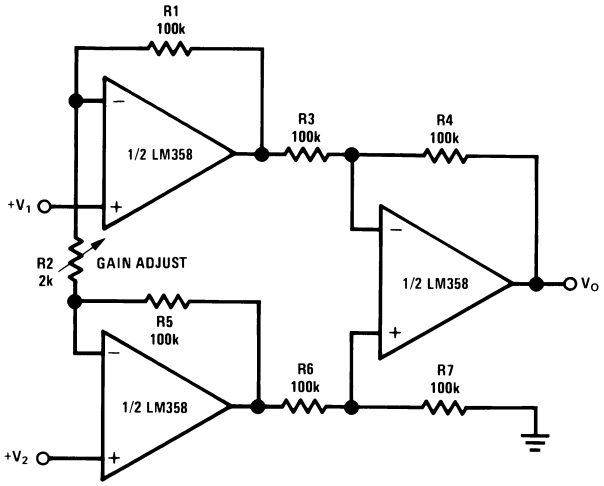
$$V_O \cong V_{REF}\left(\frac{\delta}{2}\right)\frac{R_f}{R}$$

Figure 43. Bridge Current Amplifier

Copyright © 2000–2014, Texas Instruments Incorporated

Typical Applications (continued)

(V^+ = 5.0 V_{DC})



If R1 = R5 & R3 = R4 = R6 = R7 (CMRR depends on match)

$$V_O = 1 + \frac{2R1}{R2}(V_2 - V_1)$$

As shown $V_O = 101 (V_2 - V_1)$

Figure 44. High Input Z Adjustable-Gain DC Instrumentation Amplifier

Copyright © 2000–2014, Texas Instruments Incorporated

9 Power Supply Recommendations

For proper operation, the power supplies must be properly decoupled. For decoupling the supply pins it is suggested that 10 nF capacitors be placed as close as possible to the op-amp power supply pins. For single supply, place a capacitor between V+ and V−supply leads. For dual supplies, place one capacitor between V+ and ground, and one capacitor between V- and ground.

Precautions should be taken to insure that the power supply for the integrated circuit never becomes reversed in polarity or that the unit is not inadvertently installed backwards in a test socket as an unlimited current surge through the resulting forward diode within the IC could cause fusing of the internal conductors and result in a destroyed unit.

10 Layout

10.1 Layout Guidelines

For single-ended supply configurations, the V+ pin should be bypassed to ground with a low ESR capacitor. The optimum placement is closest to the V+ pin. Care should be taken to minimize the loop area formed by the bypass capacitor connection between V+ and ground. The ground pin should be connected to the PCB ground plane at the pin of the device. The feedback components should be placed as close to the device as possible to minimize stray parasitics.

For dual supply configurations, both the V+ pin and V- pin should be bypassed to ground with a low ESR capacitor. The optimum placement is closest to the corresponding supply pin. Care should be taken to minimize the loop area formed by the bypass capacitor connection between V+ or V- and ground. The feedback components should be placed as close to the device as possible to minimize stray parasitics.

For both configurations, as ground plane underneath the device is recommended.

10.2 Layout Example

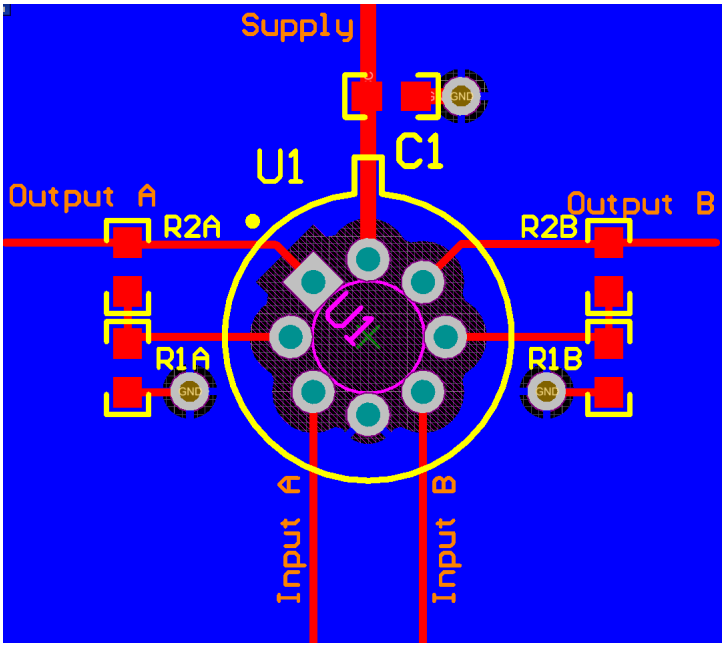


Figure 45. Layout Example

Copyright © 2000–2014, Texas Instruments Incorporated

11 Device and Documentation Support

11.1 Related Links

The table below lists quick access links. Categories include technical documents, support and community resources, tools and software, and quick access to sample or buy.

Table 1. Related Links

PARTS	PRODUCT FOLDER	SAMPLE & BUY	TECHNICAL DOCUMENTS	TOOLS & SOFTWARE	SUPPORT & COMMUNITY
LM158-N	Click here	Click here	Click here	Click here	Click here
LM258-N	Click here	Click here	Click here	Click here	Click here
LM2904-N	Click here	Click here	Click here	Click here	Click here
LM358-N	Click here	Click here	Click here	Click here	Click here

11.2 Trademarks

All trademarks are the property of their respective owners.

11.3 Electrostatic Discharge Caution



These devices have limited built-in ESD protection. The leads should be shorted together or the device placed in conductive foam during storage or handling to prevent electrostatic damage to the MOS gates.

11.4 Glossary

SLYZ022 — *TI Glossary*.

This glossary lists and explains terms, acronyms, and definitions.

12 Mechanical, Packaging, and Orderable Information

The following pages include mechanical, packaging, and orderable information. This information is the most current data available for the designated devices. This data is subject to change without notice and revision of this document. For browser-based versions of this data sheet, refer to the left-hand navigation.

Copyright © 2000–2014, Texas Instruments Incorporated

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish (6)	MSL Peak Temp (3)	Op Temp (°C)	Device Marking (4/5)	Samples
LM158AH	ACTIVE	TO-99	LMC	8	500	TBD	Call TI	Call TI	-55 to 125	(LM158AH, LM158AH)	Samples
LM158AH/NOPB	ACTIVE	TO-99	LMC	8	500	Green (RoHS & no Sb/Br)	Call TI	Level-1-NA-UNLIM	-55 to 125	(LM158AH, LM158AH)	Samples
LM158H	ACTIVE	TO-99	LMC	8	500	TBD	Call TI	Call TI	-55 to 125	(LM158H, LM158H)	Samples
LM158H/NOPB	ACTIVE	TO-99	LMC	8	500	Green (RoHS & no Sb/Br)	Call TI	Level-1-NA-UNLIM	-55 to 125	(LM158H, LM158H)	Samples
LM158J	ACTIVE	CDIP	NAB	8	40	TBD	Call TI	Call TI	-55 to 125	LM158J	Samples
LM258H	ACTIVE	TO-99	LMC	8	500	TBD	Call TI	Call TI	-25 to 85	(LM258H, LM258H)	Samples
LM258H/NOPB	ACTIVE	TO-99	LMC	8	500	Green (RoHS & no Sb/Br)	Call TI	Level-1-NA-UNLIM	-25 to 85	(LM258H, LM258H)	Samples
LM2904ITP/NOPB	ACTIVE	DSBGA	YPB	8	250	Green (RoHS & no Sb/Br)	SNAGCU	Level-1-260C-UNLIM	-40 to 85	A 09	Samples
LM2904ITPX/NOPB	ACTIVE	DSBGA	YPB	8	3000	Green (RoHS & no Sb/Br)	SNAGCU	Level-1-260C-UNLIM	-40 to 85	A 09	Samples
LM2904M	OBSOLETE	SOIC	D	8		TBD	Call TI	Call TI	-40 to 85	LM 2904M	
LM2904M/NOPB	ACTIVE	SOIC	D	8	95	Green (RoHS & no Sb/Br)	CU SN	Level-1-260C-UNLIM	-40 to 85	LM 2904M	Samples
LM2904MX	NRND	SOIC	D	8	2500	TBD	Call TI	Call TI	-40 to 85	LM 2904M	
LM2904MX/NOPB	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU SN	Level-1-260C-UNLIM	-40 to 85	LM 2904M	Samples
LM2904N/NOPB	ACTIVE	PDIP	P	8	40	Green (RoHS & no Sb/Br)	CU SN	Level-1-NA-UNLIM	-40 to 85	LM 2904N	Samples
LM358AM	NRND	SOIC	D	8	95	TBD	Call TI	Call TI	0 to 70	LM 358AM	
LM358AM/NOPB	ACTIVE	SOIC	D	8	95	Green (RoHS & no Sb/Br)	CU SN	Level-1-260C-UNLIM	0 to 70	LM 358AM	Samples
LM358AMX	NRND	SOIC	D	8	2500	TBD	Call TI	Call TI	0 to 70	LM 358AM	

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish (6)	MSL Peak Temp (3)	Op Temp (°C)	Device Marking (4/5)	Samples
LM358AMX/NOPB	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU SN	Level-1-260C-UNLIM	0 to 70	LM 358AM	Samples
LM358AN/NOPB	ACTIVE	PDIP	P	8	40	Green (RoHS & no Sb/Br)	CU SN	Level-1-NA-UNLIM	0 to 70	LM 358AN	Samples
LM358H/NOPB	ACTIVE	TO-99	LMC	8	500	Green (RoHS & no Sb/Br)	Call TI	Level-1-NA-UNLIM	0 to 70	(LM358H, LM358H)	Samples
LM358M	NRND	SOIC	D	8	95	TBD	Call TI	Call TI	0 to 70	LM 358M	
LM358M/NOPB	ACTIVE	SOIC	D	8	95	Green (RoHS & no Sb/Br)	CU SN	Level-1-260C-UNLIM	0 to 70	LM 358M	Samples
LM358MX	NRND	SOIC	D	8	2500	TBD	Call TI	Call TI	0 to 70	LM 358M	
LM358MX/NOPB	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU SN	Level-1-260C-UNLIM	0 to 70	LM 358M	Samples
LM358N/NOPB	ACTIVE	PDIP	P	8	40	Green (RoHS & no Sb/Br)	CU SN	Level-1-NA-UNLIM	0 to 70	LM 358N	Samples
LM358TP/NOPB	ACTIVE	DSBGA	YPB	8	250	Green (RoHS & no Sb/Br)	SNAGCU	Level-1-260C-UNLIM	0 to 70	A 07	Samples
LM358TPX/NOPB	ACTIVE	DSBGA	YPB	8	3000	Green (RoHS & no Sb/Br)	SNAGCU	Level-1-260C-UNLIM	0 to 70	A 07	Samples

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

(2) **RoHS:** TI defines "RoHS" to mean semiconductor products that are compliant with the current EU RoHS requirements for all 10 RoHS substances, including the requirement that RoHS substance do not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, "RoHS" products are suitable for use in specified lead-free processes. TI may reference these types of products as "Pb-Free".
RoHS Exempt: TI defines "RoHS Exempt" to mean products that contain lead but are compliant with EU RoHS pursuant to a specific EU RoHS exemption.
Green: TI defines "Green" to mean the content of Chlorine (Cl) and Bromine (Br) based flame retardants meet JS709B low halogen requirements of <=1000ppm threshold. Antimony trioxide based flame retardants must also meet the <=1000ppm threshold requirement.

(3) MSL, Peak Temp. - The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

(4) There may be additional marking, which relates to the logo, the lot trace code information, or the environmental category on the device.

(5) Multiple Device Markings will be inside parentheses. Only one Device Marking contained in parentheses and separated by a "~" will appear on a device. If a line is indented then it is a continuation of the previous line and the two combined represent the entire Device Marking for that device.

(6) Lead/Ball Finish - Orderable Devices may have multiple material finish options. Finish options are separated by a vertical ruled line. Lead/Ball Finish values may wrap to two lines if the finish value exceeds the maximum column width.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

OTHER QUALIFIED VERSIONS OF LM2904-N :

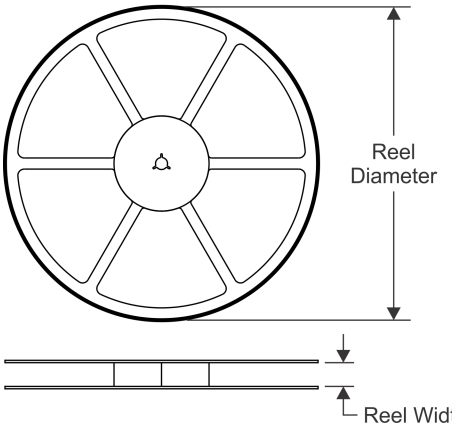
- Automotive: LM2904-Q1
- Enhanced Product: LM2904-EP

NOTE: Qualified Version Definitions:

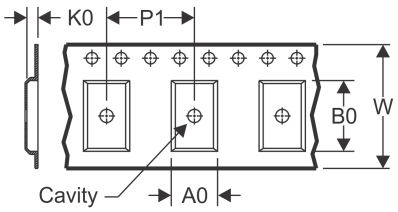
- Automotive - Q100 devices qualified for high-reliability automotive applications targeting zero defects
- Enhanced Product - Supports Defense, Aerospace and Medical Applications

TAPE AND REEL INFORMATION

REEL DIMENSIONS

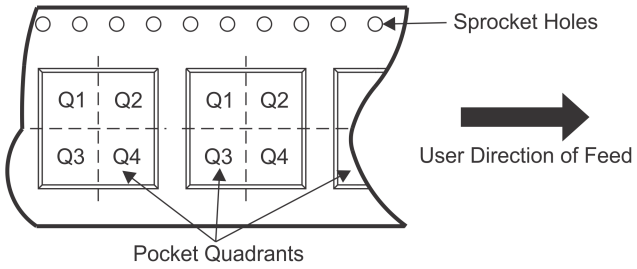


TAPE DIMENSIONS



A0	Dimension designed to accommodate the component width
B0	Dimension designed to accommodate the component length
K0	Dimension designed to accommodate the component thickness
W	Overall width of the carrier tape
P1	Pitch between successive cavity centers

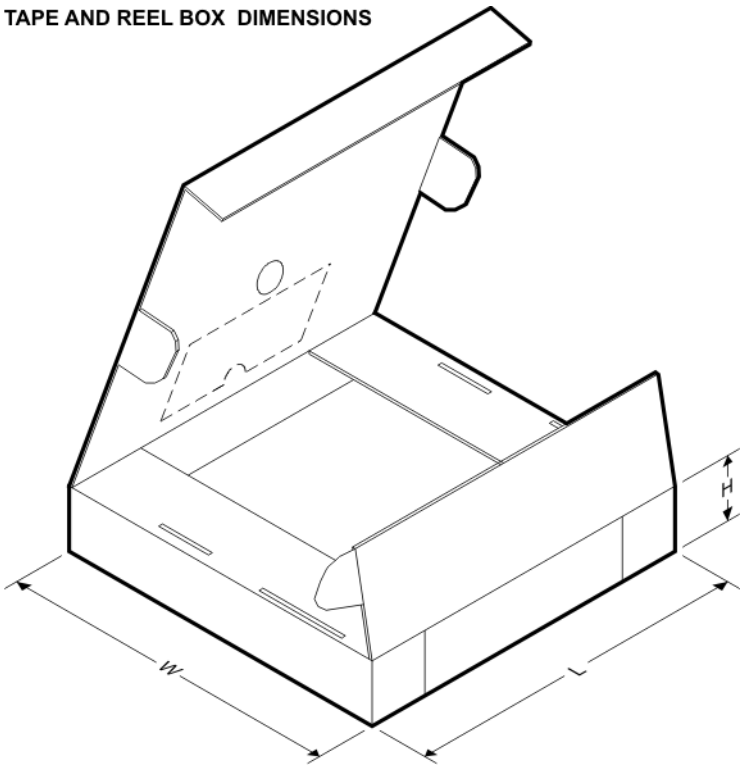
QUADRANT ASSIGNMENTS FOR PIN 1 ORIENTATION IN TAPE



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
LM2904ITP/NOPB	DSBGA	YPB	8	250	178.0	8.4	1.5	1.5	0.66	4.0	8.0	Q1
LM2904ITPX/NOPB	DSBGA	YPB	8	3000	178.0	8.4	1.5	1.5	0.66	4.0	8.0	Q1
LM2904MX	SOIC	D	8	2500	330.0	12.4	6.5	5.4	2.0	8.0	12.0	Q1
LM2904MX/NOPB	SOIC	D	8	2500	330.0	12.4	6.5	5.4	2.0	8.0	12.0	Q1
LM358AMX	SOIC	D	8	2500	330.0	12.4	6.5	5.4	2.0	8.0	12.0	Q1
LM358AMX/NOPB	SOIC	D	8	2500	330.0	12.4	6.5	5.4	2.0	8.0	12.0	Q1
LM358MX	SOIC	D	8	2500	330.0	12.4	6.5	5.4	2.0	8.0	12.0	Q1
LM358MX/NOPB	SOIC	D	8	2500	330.0	12.4	6.5	5.4	2.0	8.0	12.0	Q1
LM358TP/NOPB	DSBGA	YPB	8	250	178.0	8.4	1.5	1.5	0.66	4.0	8.0	Q1
LM358TPX/NOPB	DSBGA	YPB	8	3000	178.0	8.4	1.5	1.5	0.66	4.0	8.0	Q1

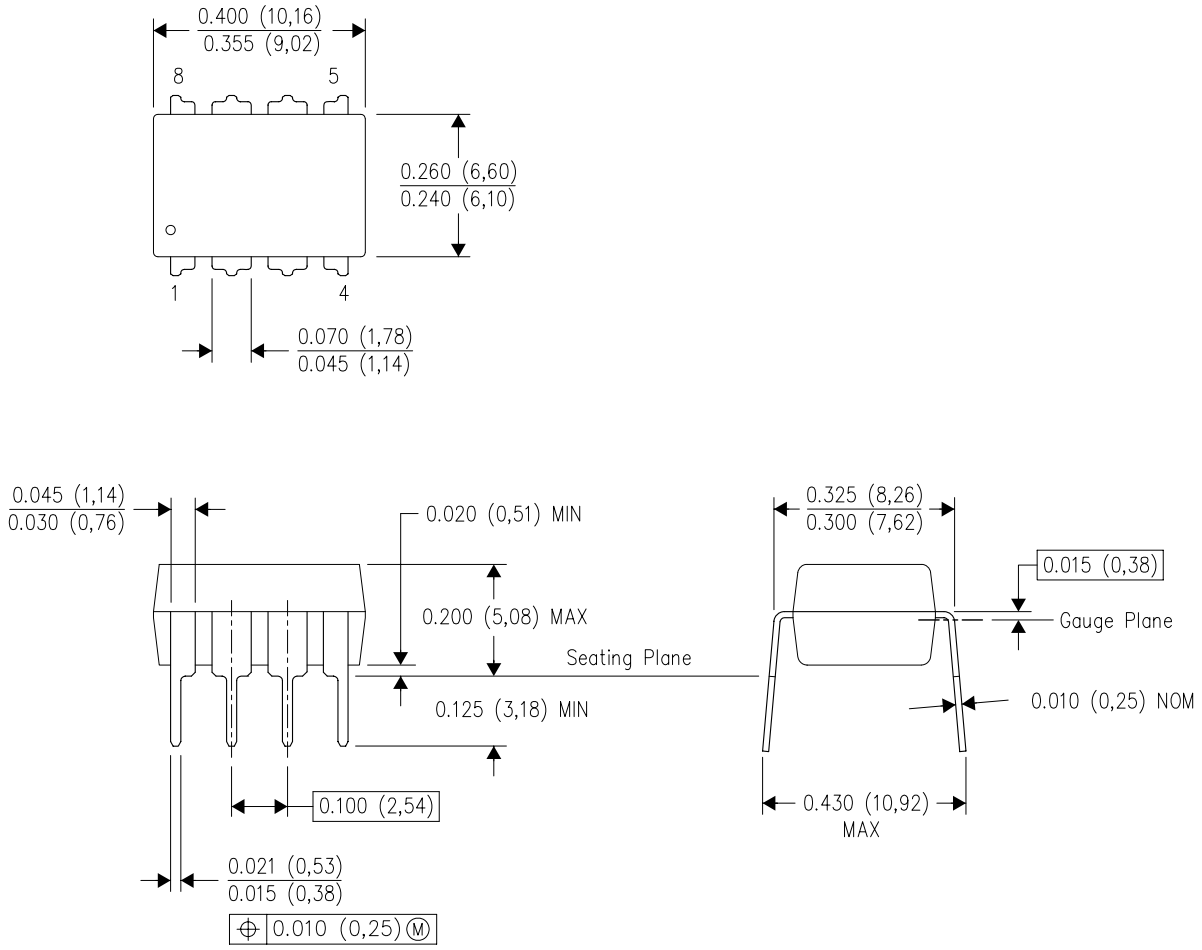
TAPE AND REEL BOX DIMENSIONS



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
LM2904ITP/NOPB	DSBGA	YPB	8	250	210.0	185.0	35.0
LM2904ITPX/NOPB	DSBGA	YPB	8	3000	210.0	185.0	35.0
LM2904MX	SOIC	D	8	2500	367.0	367.0	35.0
LM2904MX/NOPB	SOIC	D	8	2500	367.0	367.0	35.0
LM358AMX	SOIC	D	8	2500	367.0	367.0	35.0
LM358AMX/NOPB	SOIC	D	8	2500	367.0	367.0	35.0
LM358MX	SOIC	D	8	2500	367.0	367.0	35.0
LM358MX/NOPB	SOIC	D	8	2500	367.0	367.0	35.0
LM358TP/NOPB	DSBGA	YPB	8	250	210.0	185.0	35.0
LM358TPX/NOPB	DSBGA	YPB	8	3000	210.0	185.0	35.0

P (R–PDIP–T8) PLASTIC DUAL–IN–LINE PACKAGE



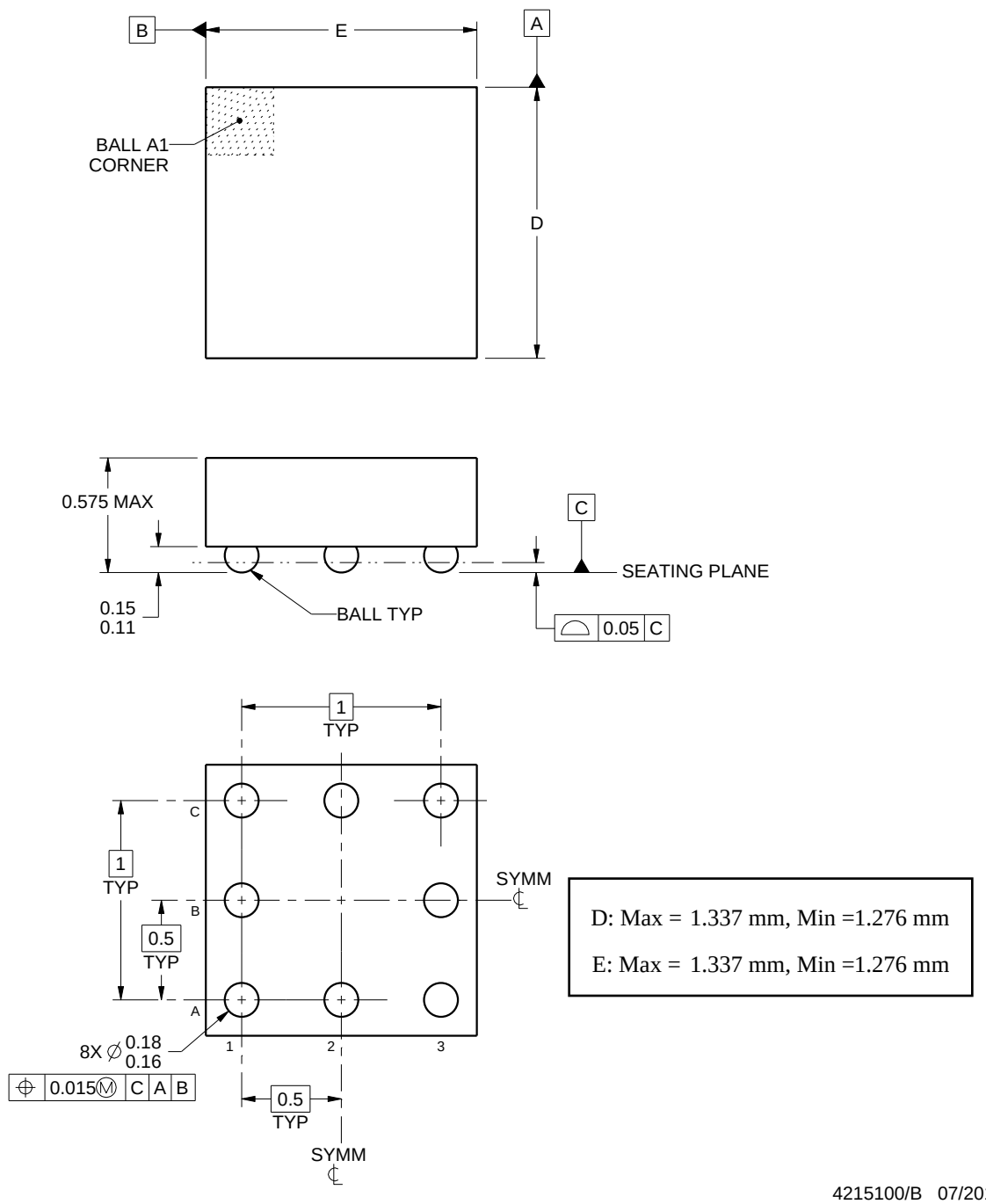
NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Falls within JEDEC MS–001 variation BA.

PACKAGE OUTLINE

YPB0008

DSBGA - 0.575 mm max height

DIE SIZE BALL GRID ARRAY



NOTES:

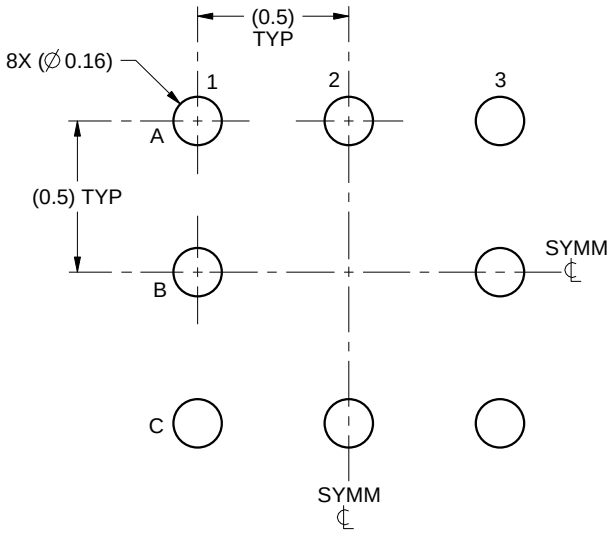
1. All linear dimensions are in millimeters. Any dimensions in parenthesis are for reference only. Dimensioning and tolerancing per ASME Y14.5M.
2. This drawing is subject to change without notice.

EXAMPLE BOARD LAYOUT

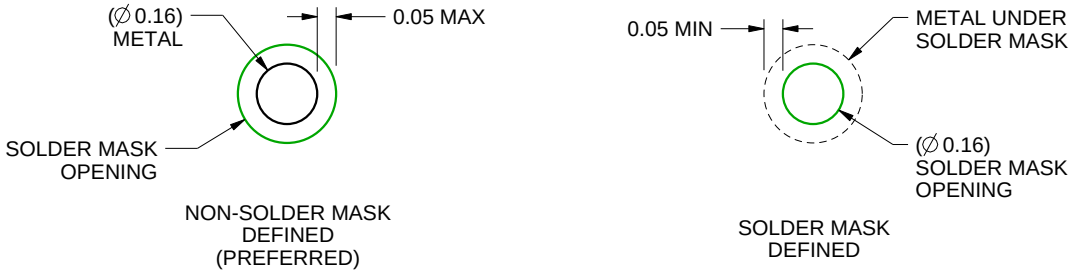
YPB0008

DSBGA - 0.575 mm max height

DIE SIZE BALL GRID ARRAY



LAND PATTERN EXAMPLE
SCALE:40X

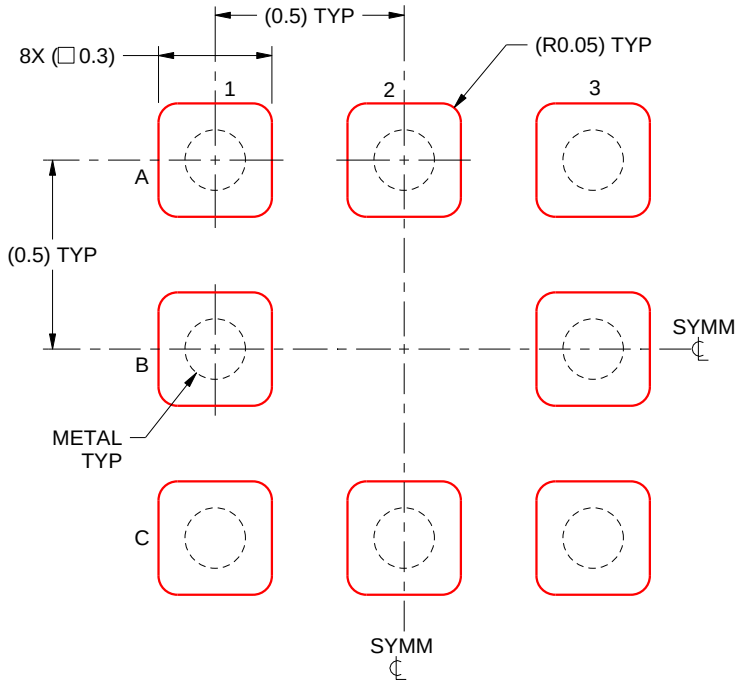


SOLDER MASK DETAILS
NOT TO SCALE

4215100/B 07/2016

NOTES: (continued)

3. Final dimensions may vary due to manufacturing tolerance considerations and also routing constraints.
   See Texas Instruments Literature No. SNVA009 (www.ti.com/lit/snva009).



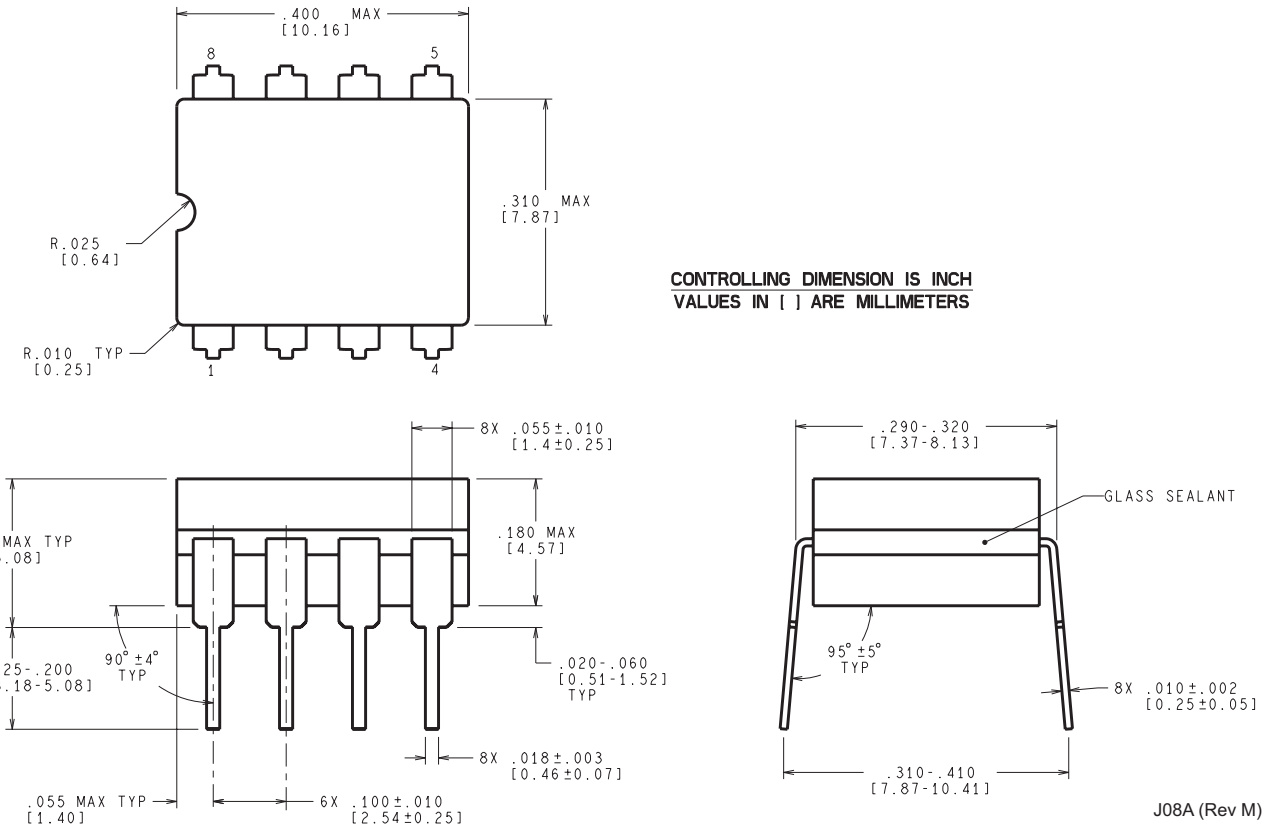
SOLDER PASTE EXAMPLE
BASED ON 0.125mm THICK STENCIL
SCALE:50X

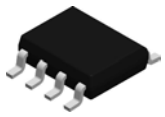
4215100/B 07/2016

NOTES: (continued)

4. Laser cutting apertures with trapezoidal walls and rounded corners may offer better paste release.

NAB0008A


.400 MAX
[10.16]
8
5
.310 MAX
[7.87]
R.025
[0.64]
R.010 TYP
[0.25]
1
4
CONTROLLING DIMENSION IS INCH
VALUES IN [ ] ARE MILLIMETERS
8X .055±.010
[1.4±0.25]
.200 MAX TYP
[5.08]
.180 MAX
[4.57]
8X .125-.200
[3.18-5.08]
90° ±4°
TYP
.020-.060
[0.51-1.52]
TYP
8X .018±.003
[0.46±0.07]
.055 MAX TYP
[1.40]
6X .100±.010
[2.54±0.25]
.290-.320
[7.37-8.13]
GLASS SEALANT
95° ±5°
TYP
8X .010±.002
[0.25±0.05]
.310-.410
[7.87-10.41]
J08A (Rev M)

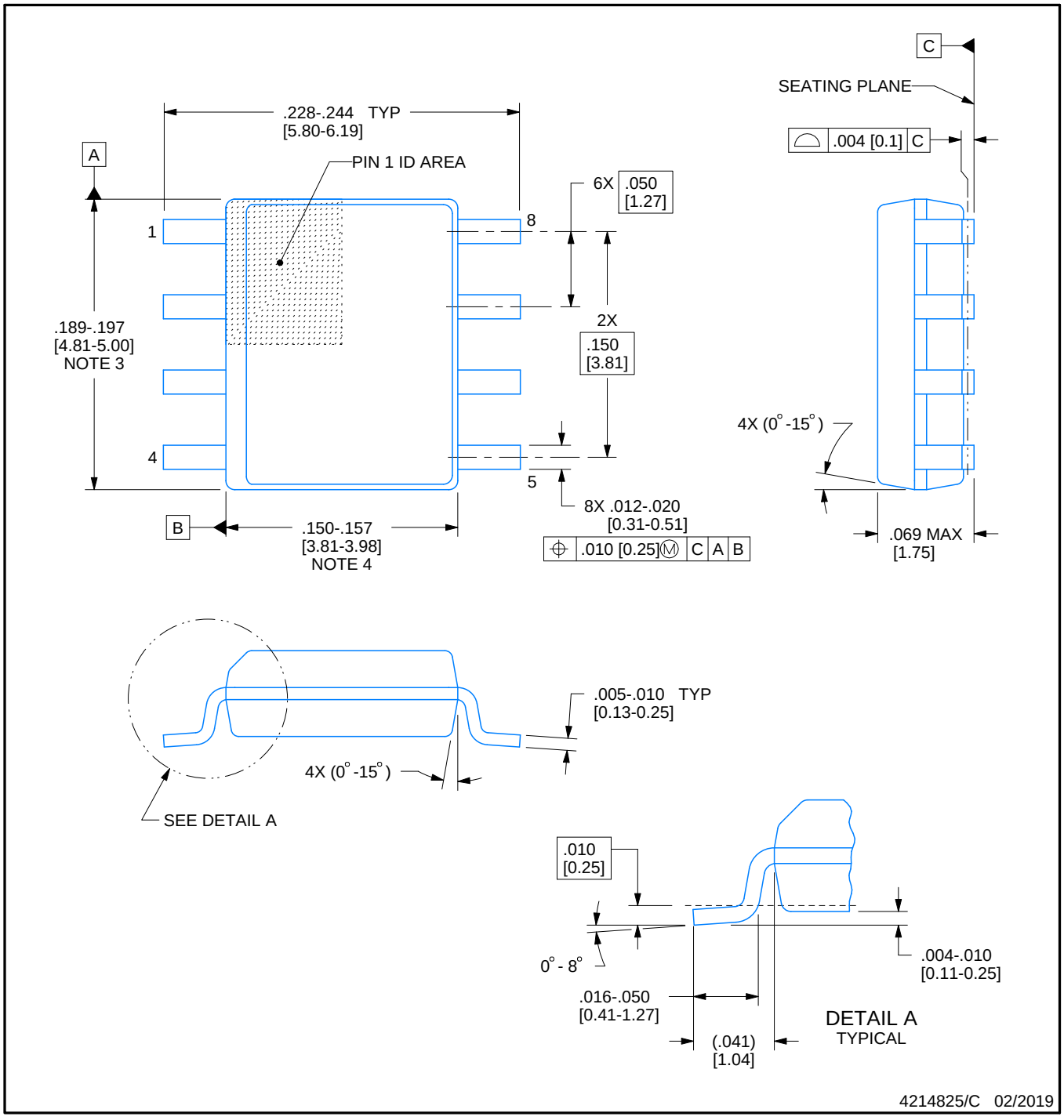


PACKAGE OUTLINE

D0008A

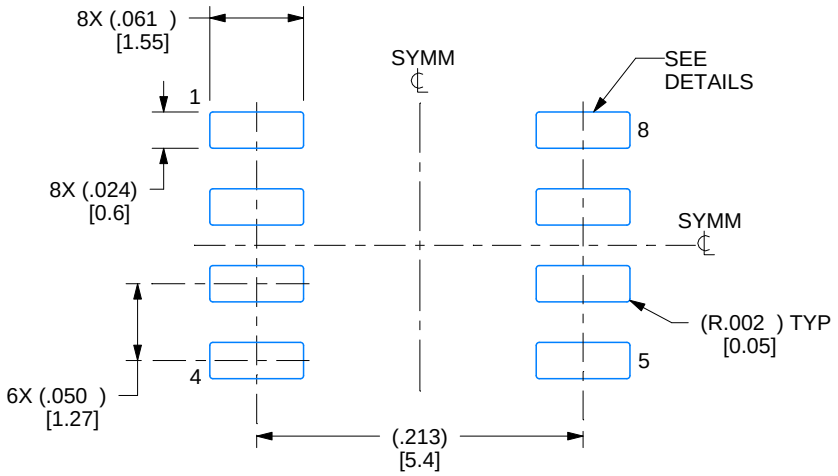
SOIC - 1.75 mm max height

SMALL OUTLINE INTEGRATED CIRCUIT

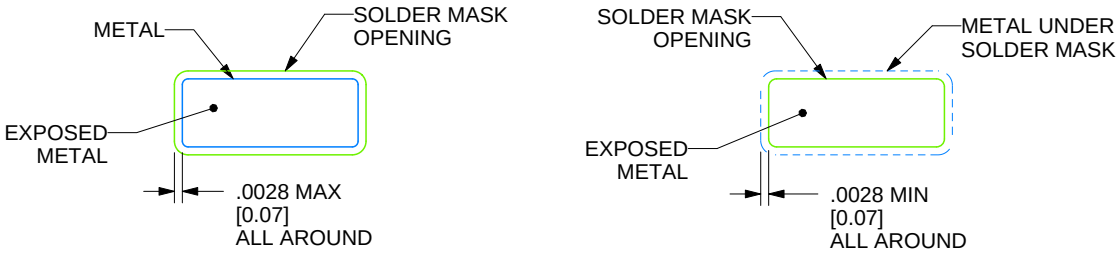


NOTES:

1. Linear dimensions are in inches [millimeters]. Dimensions in parenthesis are for reference only. Controlling dimensions are in inches. Dimensioning and tolerancing per ASME Y14.5M.
2. This drawing is subject to change without notice.
3. This dimension does not include mold flash, protrusions, or gate burrs. Mold flash, protrusions, or gate burrs shall not exceed .006 [0.15] per side.
4. This dimension does not include interlead flash.
5. Reference JEDEC registration MS-012, variation AA.



LAND PATTERN EXAMPLE
EXPOSED METAL SHOWN
SCALE:8X



SOLDER MASK DETAILS

4214825/C 02/2019

NOTES: (continued)

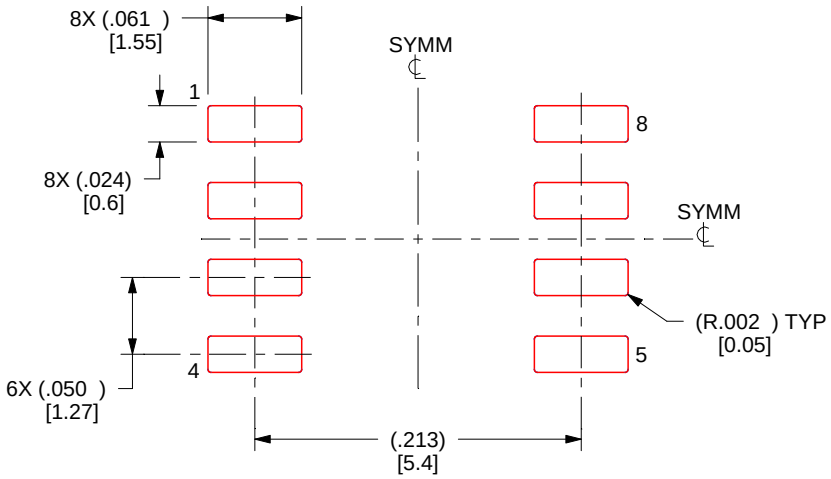
6. Publication IPC-7351 may have alternate designs.
7. Solder mask tolerances between and around signal pads can vary based on board fabrication site.

EXAMPLE STENCIL DESIGN

D0008A

SOIC - 1.75 mm max height

SMALL OUTLINE INTEGRATED CIRCUIT



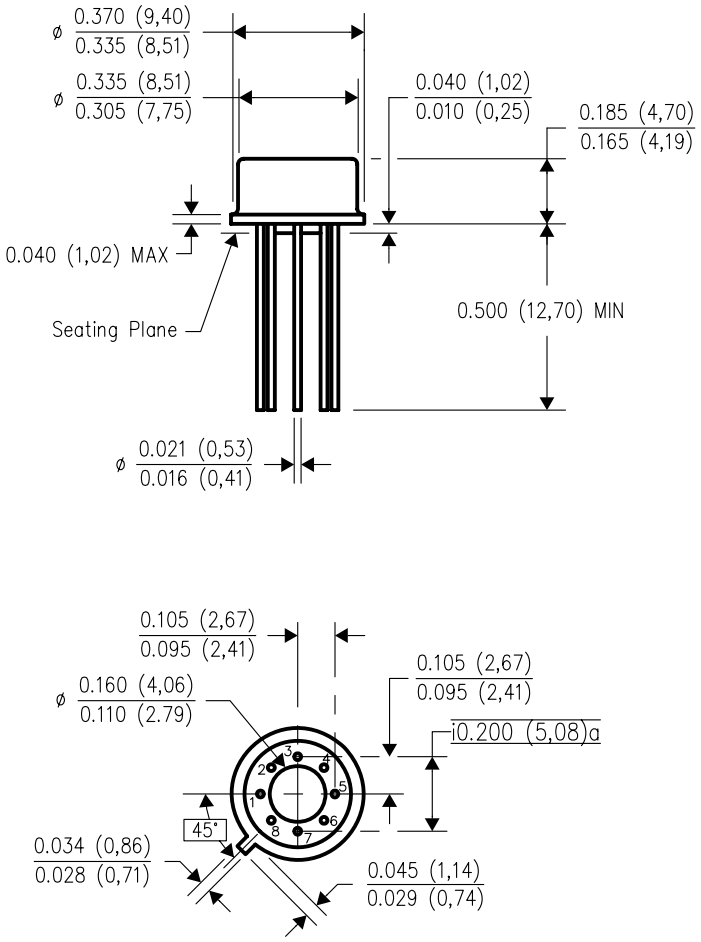
SOLDER PASTE EXAMPLE
BASED ON .005 INCH [0.125 MM] THICK STENCIL
SCALE:8X

4214825/C 02/2019

NOTES: (continued)

8. Laser cutting apertures with trapezoidal walls and rounded corners may offer better paste release. IPC-7525 may have alternate design recommendations.
9. Board assembly site may have different recommendations for stencil design.

LMC (O–MBCY–W8) METAL CYLINDRICAL PACKAGE



4202483/B 09/07

NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Leads in true position within 0.010 (0,25) R @ MMC at seating plane.
D. Pin numbers shown for reference only. Numbers may not be marked on package.
E. Falls within JEDEC MO–002/TO–99.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

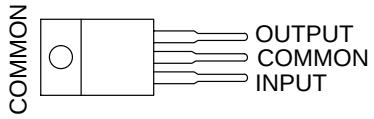
TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2019, Texas Instruments Incorporated

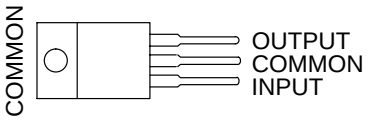
SLVS056J – MAY 1976 – REVISED MAY 2003

- **3-Terminal Regulators**
- **Output Current up to 1.5 A**
- **Internal Thermal-Overload Protection**
- **High Power-Dissipation Capability**
- **Internal Short-Circuit Current Limiting**
- **Output Transistor Safe-Area Compensation**

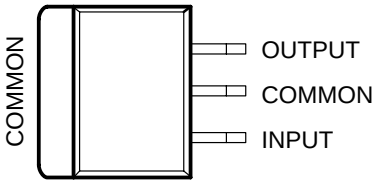
KC (TO-220) PACKAGE
(TOP VIEW)



KCS (TO-220) PACKAGE
(TOP VIEW)



KTE PACKAGE
(TOP VIEW)



description/ordering information

This series of fixed-voltage integrated-circuit voltage regulators is designed for a wide range of applications. These applications include on-card regulation for elimination of noise and distribution problems associated with single-point regulation. Each of these regulators can deliver up to 1.5 A of output current. The internal current-limiting and thermal-shutdown features of these regulators essentially make them immune to overload. In addition to use as fixed-voltage regulators, these devices can be used with external components to obtain adjustable output voltages and currents, and also can be used as the power-pass element in precision regulators.

ORDERING INFORMATION

T_J	$V_{O(NOM)}$ (V)	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 125°C	5	POWER-FLEX (KTE)	Reel of 2000	μA7805CKTER	μA7805C
		TO-220 (KC)	Tube of 50	μA7805CKC	μA7805C
		TO-220, short shoulder (KCS)	Tube of 20	μA7805CKCS	
	8	POWER-FLEX (KTE)	Reel of 2000	μA7808CKTER	μA7808C
		TO-220 (KC)	Tube of 50	μA7808CKC	μA7808C
		TO-220, short shoulder (KCS)	Tube of 20	μA7808CKCS	
	10	POWER-FLEX (KTE)	Reel of 2000	μA7810CKTER	μA7810C
		TO-220 (KC)	Tube of 50	μA7810CKC	μA7810C
	12	POWER-FLEX (KTE)	Reel of 2000	μA7812CKTER	μA7812C
		TO-220 (KC)	Tube of 50	μA7812CKC	μA7812C
		TO-220, short shoulder (KCS)	Tube of 20	μA7812CKCS	
	15	POWER-FLEX (KTE)	Reel of 2000	μA7815CKTER	μA7815C
		TO-220 (KC)	Tube of 50	μA7815CKC	μA7815C
		TO-220, short shoulder (KCS)	Tube of 20	μA7815CKCS	
	24	POWER-FLEX (KTE)	Reel of 2000	μA7824CKTER	μA7824C
		TO-220 (KC)	Tube of 50	μA7824CKC	μA7824C

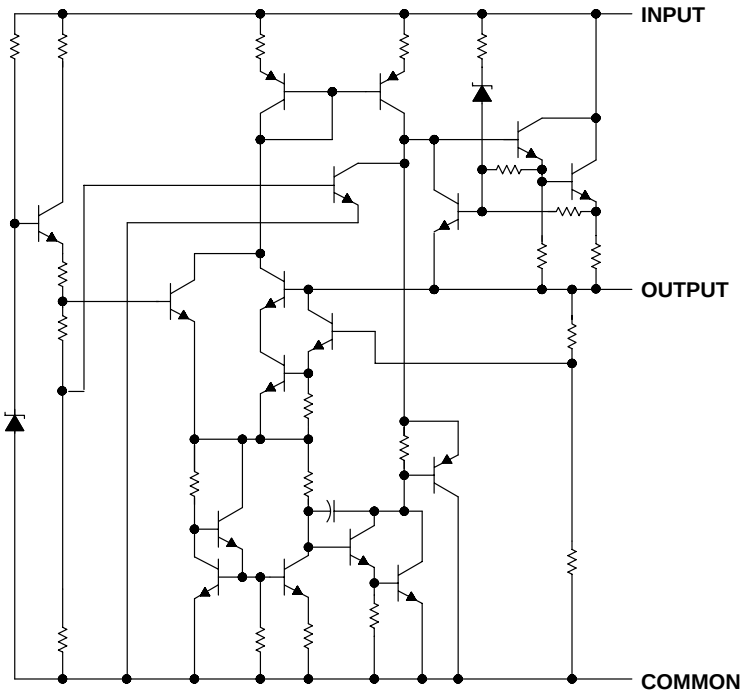
† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

Copyright © 2003, Texas Instruments Incorporated

schematic



absolute maximum ratings over virtual junction temperature range (unless otherwise noted)†

Input voltage, V_I: μA7824C . . . . . 40 V
All others . . . . . 35 V
Operating virtual junction temperature, T_J . . . . . 150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds . . . . . 260°C
Storage temperature range, T_{stg} . . . . . –65°C to 150°C

† Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

package thermal data (see Note 1)

PACKAGE	BOARD	θ_{JC}	θ_{JA}
POWER-FLEX (KTE)	High K, JESD 51-5	3°C/W	23°C/W
TO-220 (KC/KCS)	High K, JESD 51-5	3°C/W	19°C/W

NOTE 1: Maximum power dissipation is a function of $T_J(max)$, θ_{JA}, and T_A. The maximum allowable power dissipation at any allowable ambient temperature is $P_D = (T_J(max) – T_A)/\theta_{JA}$. Operating at the absolute maximum T_J of 150°C can affect reliability.

recommended operating conditions

			MIN	MAX	UNIT
V_I	Input voltage	μA7805C	7	25	V
		μA7808C	10.5	25	
		μA7810C	12.5	28	
		μA7812C	14.5	30	
		μA7815C	17.5	30	
		μA7824C	27	38	
I_O	Output current			1.5	A
T_J	Operating virtual junction temperature	μA7800C series	0	125	°C

electrical characteristics at specified virtual junction temperature, V_I = 10 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7805C			UNIT
			MIN	TYP	MAX	
Output voltage	I_O = 5 mA to 1 A, V_I = 7 V to 20 V, $P_D \le$ 15 W	25°C	4.8	5	5.2	V
		0°C to 125°C	4.75		5.25	
Input voltage regulation	V_I = 7 V to 25 V	25°C		3	100	mV
	V_I = 8 V to 12 V			1	50	
Ripple rejection	V_I = 8 V to 18 V, f = 120 Hz	0°C to 125°C	62	78		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		15	100	mV
	I_O = 250 mA to 750 mA			5	50	
Output resistance	f = 1 kHz	0°C to 125°C		0.017		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1.1		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		40		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.2	8	mA
Bias current change	V_I = 7 V to 25 V	0°C to 125°C			1.3	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		750		mA
Peak output current		25°C		2.2		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 14 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7808C			UNIT
			MIN	TYP	MAX	
Output voltage	I_O = 5 mA to 1 A, V_I = 10.5 V to 23 V, $P_D \le 15$ W	25°C	7.7	8	8.3	V
		0°C to 125°C	7.6		8.4	
Input voltage regulation	V_I = 10.5 V to 25 V	25°C		6	160	mV
	V_I = 11 V to 17 V			2	80	
Ripple rejection	V_I = 11.5 V to 21.5 V, f = 120 Hz	0°C to 125°C	55	72		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	160	mV
	I_O = 250 mA to 750 mA			4	80	
Output resistance	f = 1 kHz	0°C to 125°C		0.016		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–0.8		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		52		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.3	8	mA
Bias current change	V_I = 10.5 V to 25 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		450		mA
Peak output current		25°C		2.2		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 17 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7810C			UNIT
			MIN	TYP	MAX	
Output voltage	I_O = 5 mA to 1 A, V_I = 12.5 V to 25 V, $P_D \le 15$ W	25°C	9.6	10	10.4	V
		0°C to 125°C	9.5	10	10.5	
Input voltage regulation	V_I = 12.5 V to 28 V	25°C		7	200	mV
	V_I = 14 V to 20 V			2	100	
Ripple rejection	V_I = 13 V to 23 V, f = 120 Hz	0°C to 125°C	55	71		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	200	mV
	I_O = 250 mA to 750 mA			4	100	
Output resistance	f = 1 kHz	0°C to 125°C		0.018		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		70		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.3	8	mA
Bias current change	V_I = 12.5 V to 28 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		400		mA
Peak output current		25°C		2.2		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 19 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7812C			UNIT
			MIN	TYP	MAX	
Output voltage	I_O = 5 mA to 1 A, V_I = 14.5 V to 27 V, $P_D \le 15$ W	25°C	11.5	12	12.5	V
		0°C to 125°C	11.4		12.6	
Input voltage regulation	V_I = 14.5 V to 30 V	25°C		10	240	mV
	V_I = 16 V to 22 V			3	120	
Ripple rejection	V_I = 15 V to 25 V, f = 120 Hz	0°C to 125°C	55	71		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	240	mV
	I_O = 250 mA to 750 mA			4	120	
Output resistance	f = 1 kHz	0°C to 125°C		0.018		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		75		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.3	8	mA
Bias current change	V_I = 14.5 V to 30 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		350		mA
Peak output current		25°C		2.2		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 23 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7815C			UNIT
			MIN	TYP	MAX	
Output voltage	I_O = 5 mA to 1 A, V_I = 17.5 V to 30 V, $P_D \le 15$ W	25°C	14.4	15	15.6	V
		0°C to 125°C	14.25		15.75	
Input voltage regulation	V_I = 17.5 V to 30 V	25°C		11	300	mV
	V_I = 20 V to 26 V			3	150	
Ripple rejection	V_I = 18.5 V to 28.5 V, f = 120 Hz	0°C to 125°C	54	70		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	300	mV
	I_O = 250 mA to 750 mA			4	150	
Output resistance	f = 1 kHz	0°C to 125°C		0.019		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		90		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.4	8	mA
Bias current change	V_I = 17.5 V to 30 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		230		mA
Peak output current		25°C		2.1		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

electrical characteristics at specified virtual junction temperature, V_I = 33 V, I_O = 500 mA (unless otherwise noted)

PARAMETER	TEST CONDITIONS	T_J†	μA7824C			UNIT
			MIN	TYP	MAX	
Output voltage	I_O = 5 mA to 1 A, V_I = 27 V to 38 V, $P_D \le 15$ W	25°C	23	24	25	V
		0°C to 125°C	22.8		25.2	
Input voltage regulation	V_I = 27 V to 38 V	25°C		18	480	mV
	V_I = 30 V to 36 V			6	240	
Ripple rejection	V_I = 28 V to 38 V, f = 120 Hz	0°C to 125°C	50	66		dB
Output voltage regulation	I_O = 5 mA to 1.5 A	25°C		12	480	mV
	I_O = 250 mA to 750 mA			4	240	
Output resistance	f = 1 kHz	0°C to 125°C		0.028		Ω
Temperature coefficient of output voltage	I_O = 5 mA	0°C to 125°C		–1.5		mV/°C
Output noise voltage	f = 10 Hz to 100 kHz	25°C		170		μV
Dropout voltage	I_O = 1 A	25°C		2		V
Bias current		25°C		4.6	8	mA
Bias current change	V_I = 27 V to 38 V	0°C to 125°C			1	mA
	I_O = 5 mA to 1 A				0.5	
Short-circuit output current		25°C		150		mA
Peak output current		25°C		2.1		A

† Pulse-testing techniques maintain the junction temperature as close to the ambient temperature as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-μF capacitor across the input and a 0.1-μF capacitor across the output.

APPLICATION INFORMATION

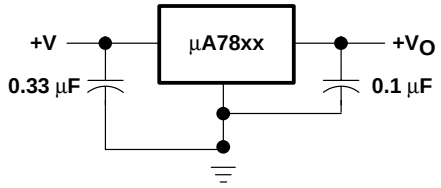


Figure 1. Fixed-Output Regulator

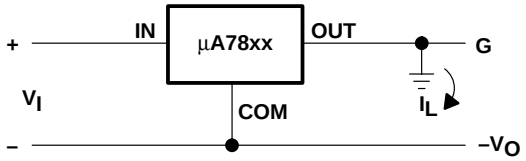
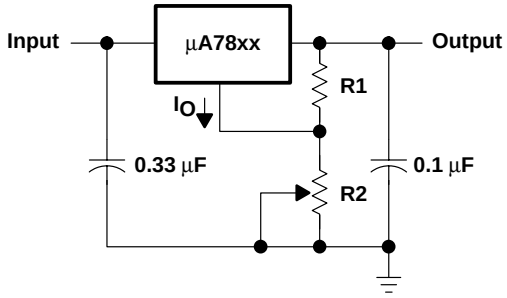


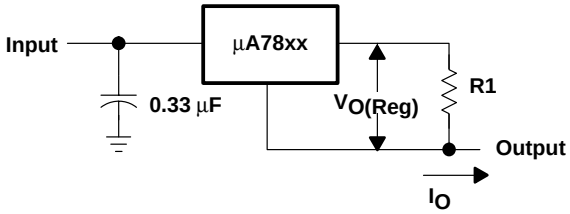
Figure 2. Positive Regulator in Negative Configuration (V_I Must Float)



NOTE A: The following formula is used when V_{XX} is the nominal output voltage (output to common) of the fixed regulator:

$$V_O = V_{xx} + \left(\frac{V_{xx}}{R1} + I_Q\right)R2$$

Figure 3. Adjustable-Output Regulator



$I_O = (V_O/R1) + I_O$ Bias Current

Figure 4. Current Regulator

APPLICATION INFORMATION

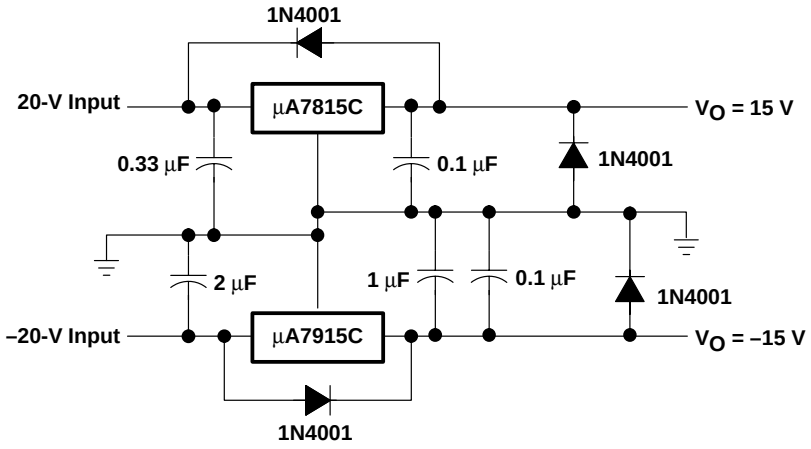


Figure 5. Regulated Dual Supply

operation with a load common to a voltage of opposite polarity

In many cases, a regulator powers a load that is not connected to ground but, instead, is connected to a voltage source of opposite polarity (e.g., operational amplifiers, level-shifting circuits, etc.). In these cases, a clamp diode should be connected to the regulator output as shown in Figure 6. This protects the regulator from output polarity reversals during startup and short-circuit operation.

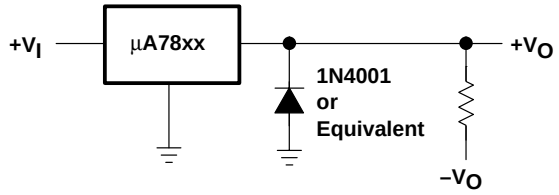


Figure 6. Output Polarity-Reversal-Protection Circuit

reverse-bias protection

Occasionally, the input voltage to the regulator can collapse faster than the output voltage. This can occur, for example, when the input supply is crowbarred during an output overvoltage condition. If the output voltage is greater than approximately 7 V, the emitter-base junction of the series-pass element (internal or external) could break down and be damaged. To prevent this, a diode shunt can be used as shown in Figure 7.

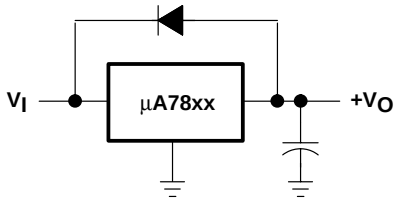
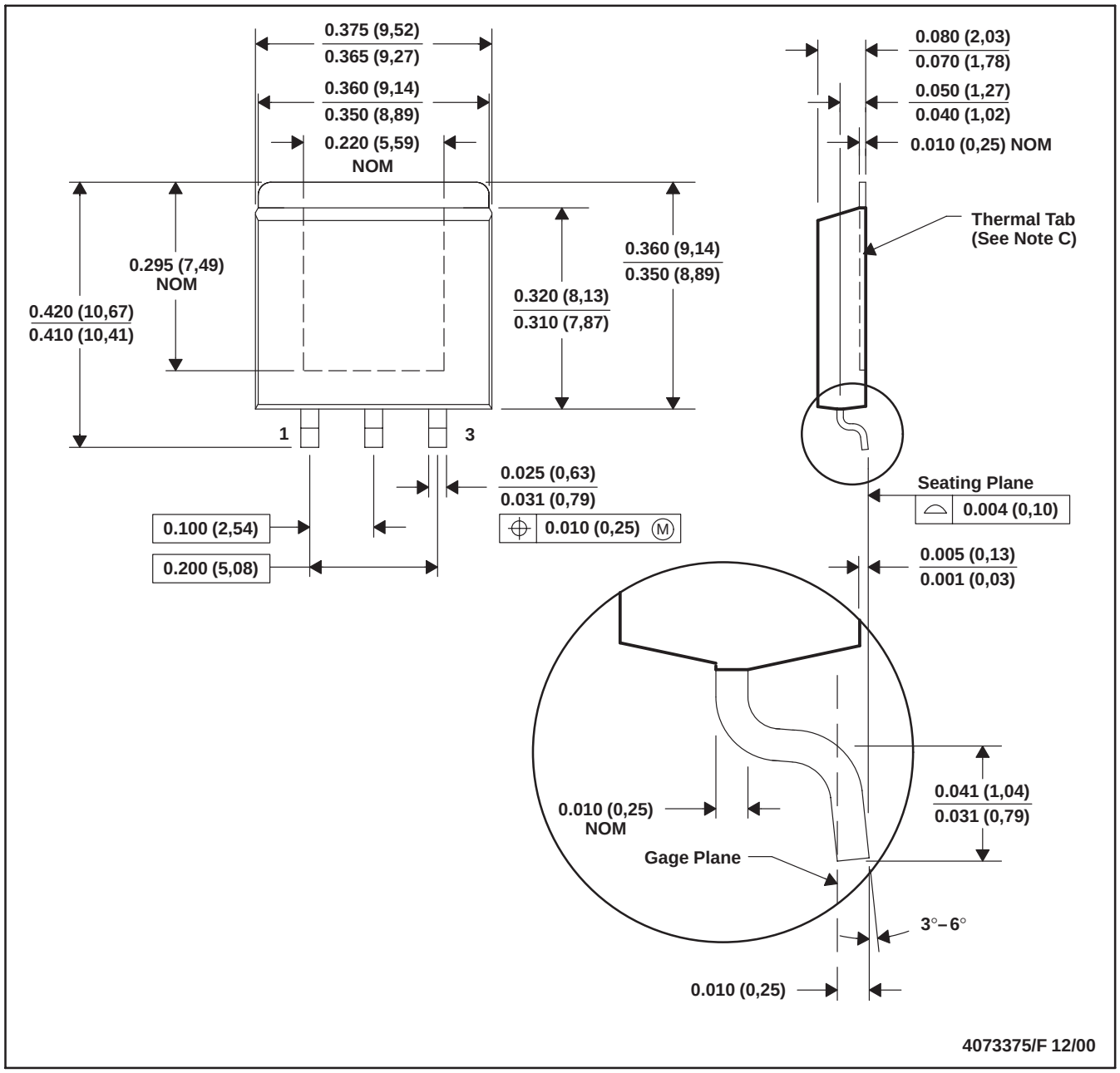


Figure 7. Reverse-Bias-Protection Circuit

MPFM001E – OCTOBER 1994 – REVISED JANUARY 2001

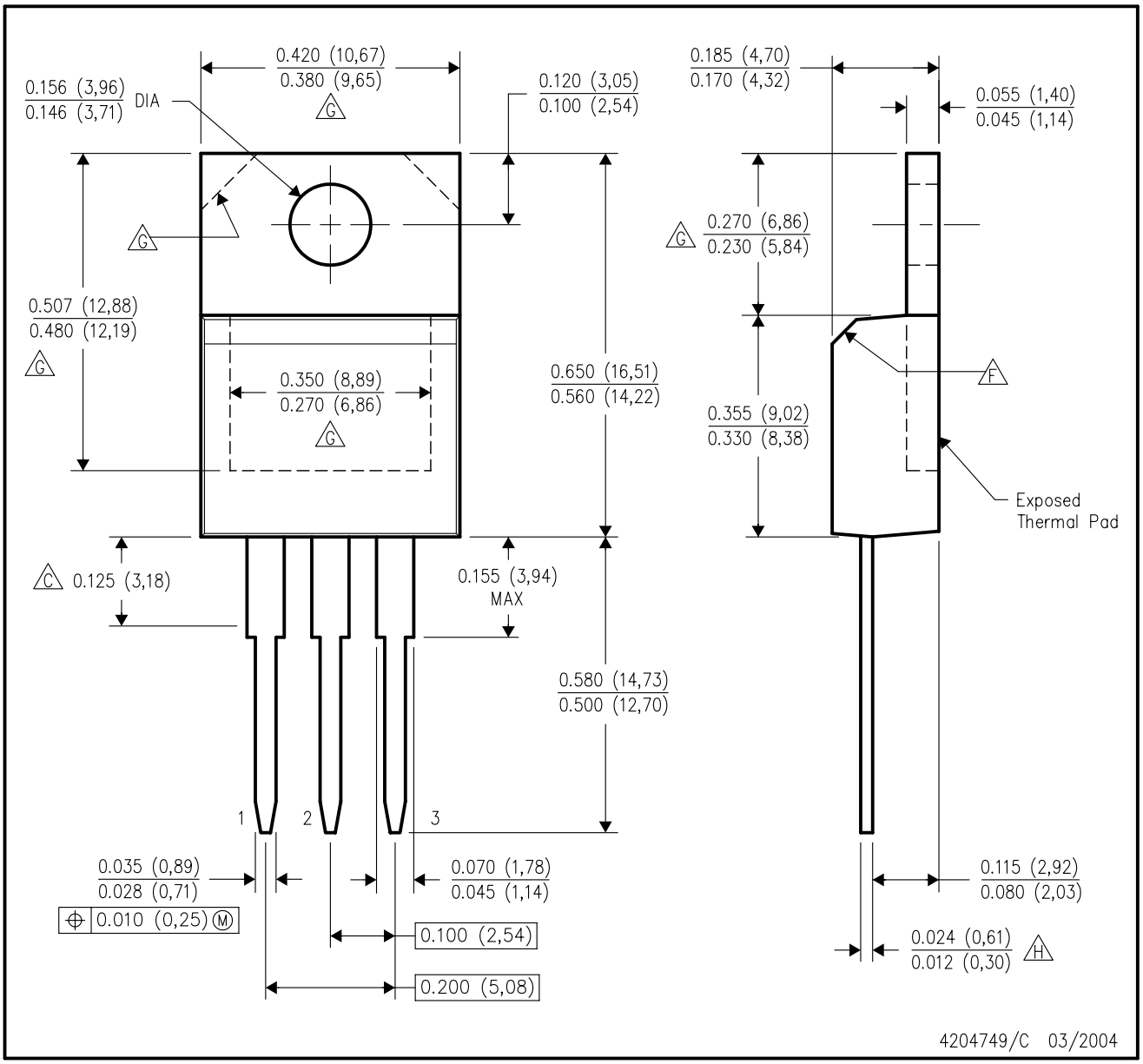
KTE (R-PSFM-G3) **PowerFLEX™ PLASTIC FLANGE-MOUNT**



NOTES: A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. The center lead is in electrical contact with the thermal tab.
D. Dimensions do not include mold protrusions, not to exceed 0.006 (0,15).
E. Falls within JEDEC MO-169

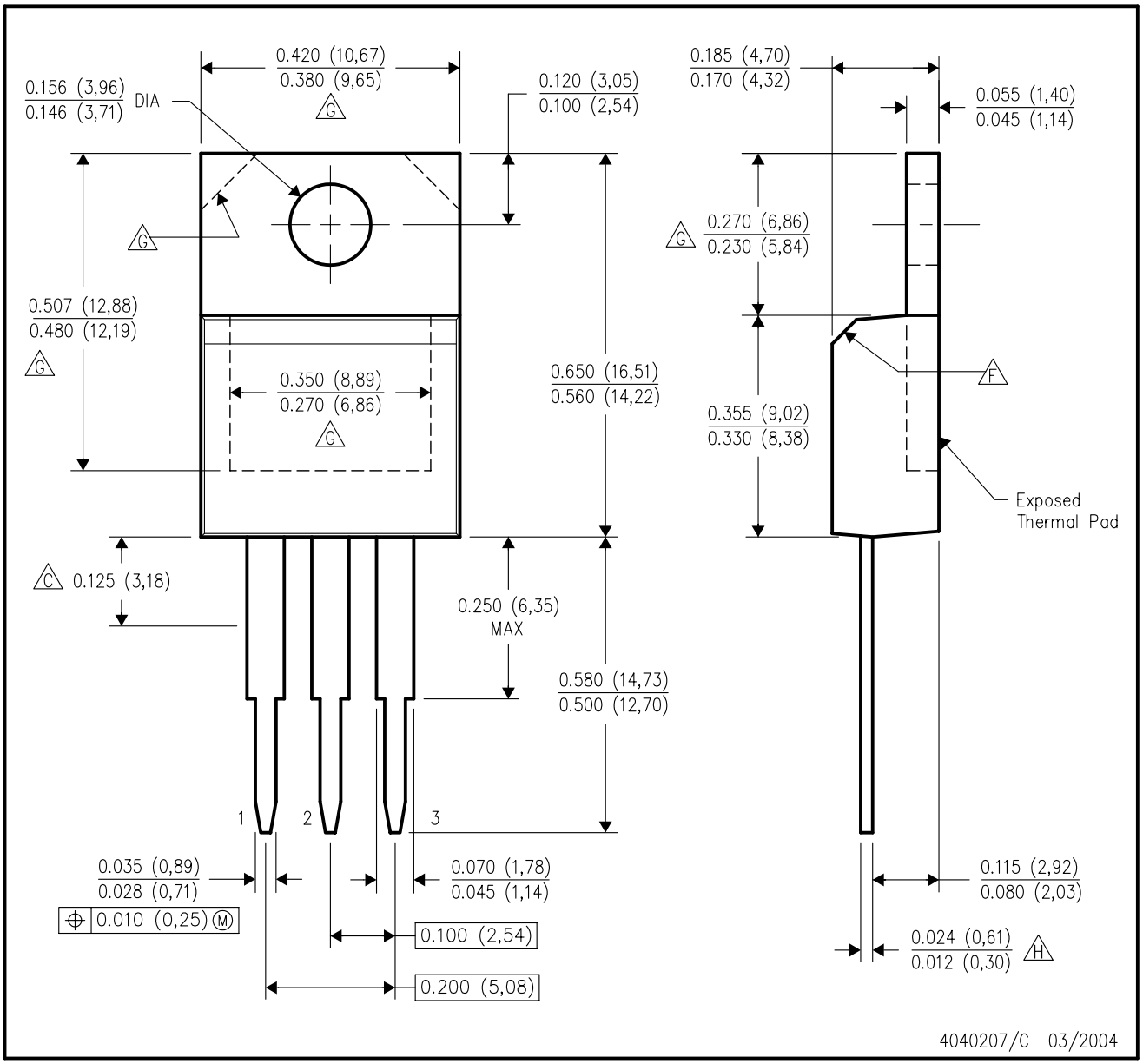
PowerFLEX is a trademark of Texas Instruments.

KCS (R–PSFM–T3) PLASTIC FLANGE–MOUNT PACKAGE



NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Lead dimensions are not controlled within this area.
D. All lead dimensions apply before solder dip.
E. The center lead is in electrical contact with the mounting tab.
F. The chamfer is optional.
G. Thermal pad contour optional within these dimensions.
H. Falls within JEDEC TO–220 variation AB, except minimum lead thickness.

KC (R–PSFM–T3) PLASTIC FLANGE–MOUNT PACKAGE



NOTES:
- A. All linear dimensions are in inches (millimeters).
- B. This drawing is subject to change without notice.
- C. Lead dimensions are not controlled within this area.
- D. All lead dimensions apply before solder dip.
- E. The center lead is in electrical contact with the mounting tab.
- F. The chamfer is optional.
- G. Thermal pad contour optional within these dimensions.
- H. Falls within JEDEC TO–220 variation AB, except minimum lead thickness.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

Copyright © 2004, Texas Instruments Incorporated