AVR BASICS

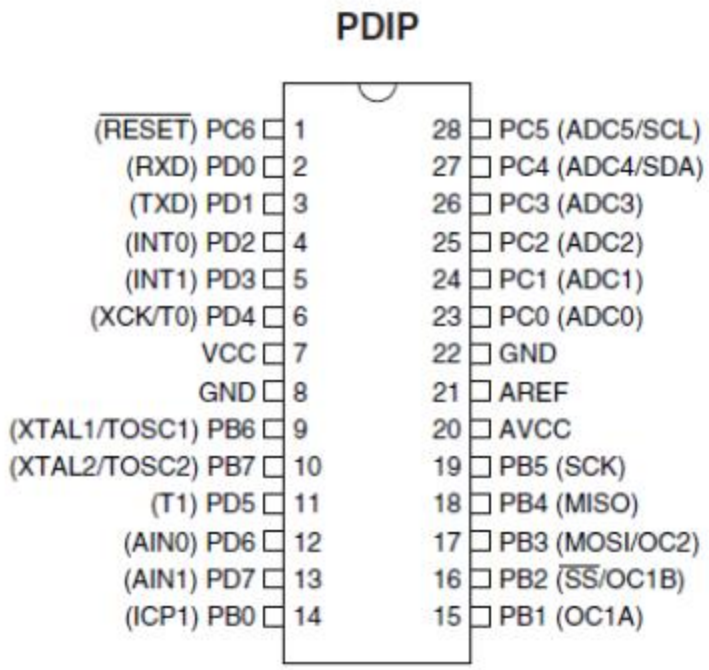
Datasheet

A datasheet of any electronic item/component/device is a document in which the manufacturer gives information regarding the product to its users. The data given in it is in detail. All the features, technical specs, design, register Summary, expected usage, troubleshooting, pin details, etc, everything is given in detail. It is the best source of info for that particular device!! All electronic components have a datasheet published by the manufacturers. You can Google them out!! They are free. Check out the datasheet of the following AVR MCUs:

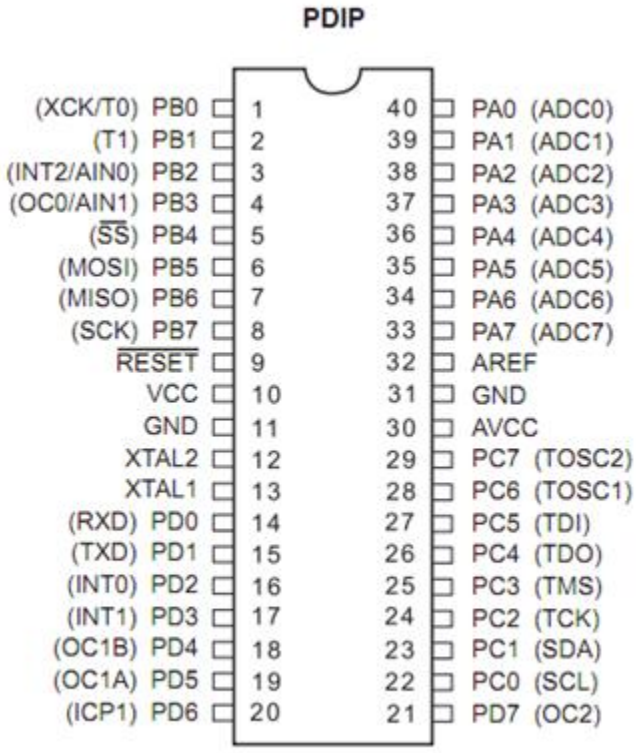
- ATTINY2313
- ATMEGA8
- ATMEGA16
- ATMEGA32
- ATMEGA168

Pin Configuration

The pin configurations of an ATMEGA8 and ATMEGA16/32 MCU are shown below.



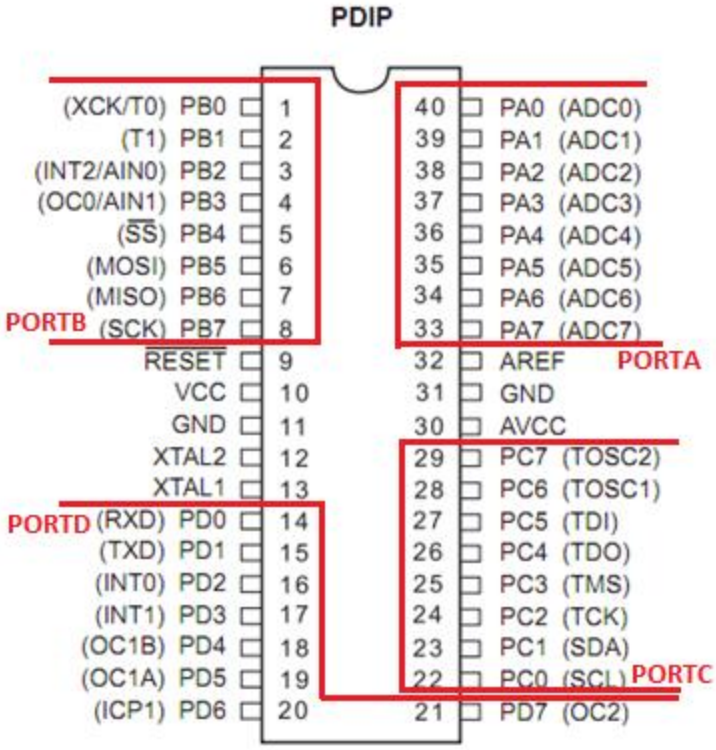
ATMEGA8 Pin Configuration



ATMEGA16/32 Pin Configuration

Please note that we will be dealing with DIP packages. For more information of types of IC Packages, visit this site. Now, let's analyze the pin configuration of ATMEGA16/32. First of all, note that the ATMEGA16 and ATMEGA32 MCUs are completely similar, except the fact that ATMEGA16 has 16KB Flash, 1KB RAM and 512B EEPROM whereas ATMEGA32 has 32KB Flash, 2KB RAM and 1KB EEPROM.

Hey, don't get scared after looking at the pin configurations. The MCU has 40 pins. It's not that complicated. Here, let me simplify it as follows.



ATMEGA32 Pin Configuration Simplified

So here, as you can see, I have split the pins into four major parts. Remember about ports? I said that ports contain the pins of a MCU. ATMEGA32 has four ports, PORTA, PORTB, PORTC and PORTD, each one having 8 pins. Don't worry about the other details, we will discuss them as and when needed. Right now, concentrate on PA0…PA7, PB0…PB7, PC0…PC7 and PD0…PD7. These are the four ports that make up the GPIO pins (General Purpose Input Output). These concepts are discusses in my previous post, Basics of Microcontroller.

So, how many pins over? $4 \times 8 = 32$. How many remain? $40 - 32 = 8$! The remaining 8 are mostly consumed by supply pins (VCC, AVCC), ground (GND), reset, XTAL pins, and other minor stuffs. So, all the 40 pins over, isn't it? ;)

P.S. Every MCU has an internal oscillator which determines its frequency of oscillator. But we need not stick to it. We can connect an external crystal oscillator to generate higher frequencies and clock pulses. This external oscillator is connected across the XTAL pins (XTAL1 and XTAL2).

AVR Peripherals

Okay, now have a look again at the pin configuration of ATMEGA32. Have a look at all the GPIO pins of all the ports. Can you see some things written in the brackets? Like PA0 (ADC0), PB5 (MOSI), PC2 (TCK), PD1 (TXD), etc. Well, these are the *extra* features that the MCU can offer you apart from GPIO. In other words, these pins show dual behavior. If nothing is specified, they act as GPIO pins. The secondary features of these pins become active only if you enable certain bits of some registers. These are called peripherals. There are several peripherals that AVR offers in ATMEGA32, some are as follows:

- **ADC** – Analog to Digital Converter – usually 10-12 bit
- **Timers** – 8 bit and 16 bit timers
- **JTAG** – Joint Test Action Group
- **TWI** – Two Wire Interface (or) **I2C** – Inter-Integrated Circuit
- **USART** – Universal Synchronous Asynchronous Receiver Transmitter
- **UART** – Universal Asynchronous Receiver Transmitter
- **SPI** – Serial Peripheral Interface
- **WDT**– Watchdog Timer …and many more!

You can get a complete list from the datasheet. Let me give you brief idea regarding them. Detailed information will be given in later posts. For now, the following is enough.

ADC stands for Analog to Digital Conversion. The term is self-explaining. This feature converts Analog signals into Digital signals.

Timers are something which are a consequence of clock frequency. We can manipulate the clock pulses in order to generate timers of required resolution.

JTAG corresponds to testing of the circuit. When we make a circuit and fix the MCU onto it, we use JTAG to verify whether our connection, soldering, circuit design, etc is correct or not.

TWI/I2C (its actually I-square-C) is a revolutionary technology by Philips, in which two devices are connected and communicate by using two wires! USART/UART is related to serial communication in which the MCU sends and receives signals from another device based on serial protocol. SPI is something which helps to interface I2C, UART, etc. The adjoining figure showing SPI Communication using UART is just for representation purpose.

WDT (Watch Dog Timer) is something interesting it seems. The name is also quite interesting. Just like a watchdog always keeps an eye on it's master to protect him from any harm, WDT keeps an eye on the execution of the code to protect the MCU from any harm. A WDT is a computer hardware or software timer that triggers a system reset or other corrective action if the main program, due to some fault condition, such as a hang, neglects to regularly service the watchdog (writing a "service pulse" to it, also referred to as "kicking the dog", "petting the dog", "feeding the watchdog" or "waking the watchdog"). The intention is to bring the system back from the unresponsive state into normal operation.

Source: http://maxembedded.wordpress.com/2011/06/09/avr-basics/