NextSapiens

**Study Material for Summer Workshop 2008**

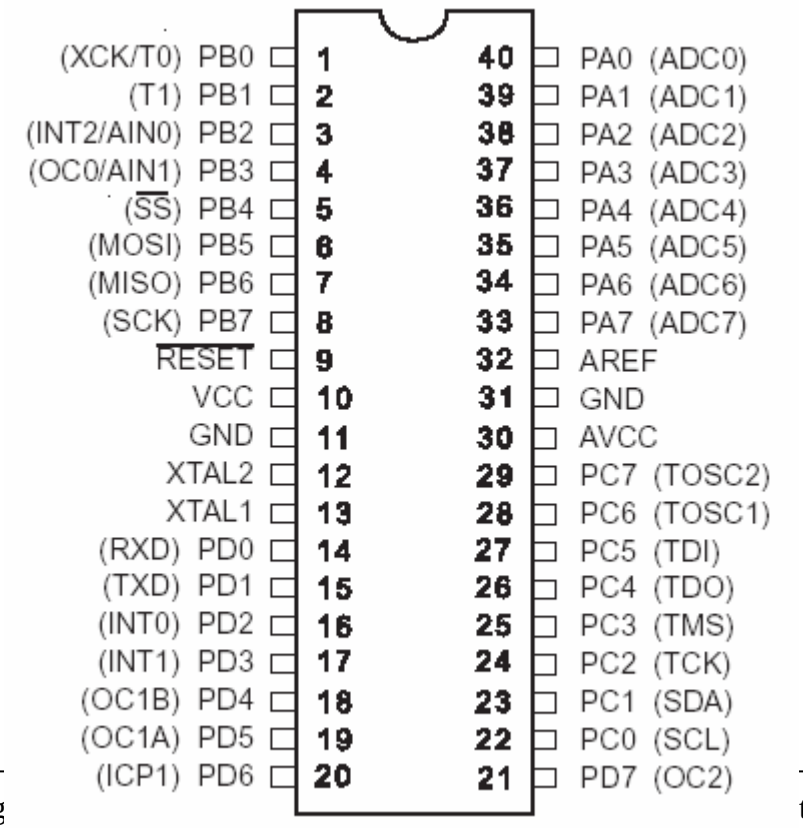
## Introduction to Microcontrollers:

A Microcontroller can be defined as a "Computer-on-Chip". Just as a Personal Computer has input devices like keyboards, mouse, etc, output devices like monitor, printer, etc and the Central Processing Unit, a Microcontroller also has Input/Output ports and a Processor embedded into a single chip. A Microcontroller is a complete self-sufficient system usually requiring no external components.

## Difference between Microprocessor and Microcontroller:

Often people misuse the terms "Microprocessor" and "Microcontroller", but both of them are different. A Microprocessor is just a processor similar to a CPU of a PC. A Microprocessor has no Input/Output ports or Memory on chip. A lot of peripheral components are required for the functioning of the Microprocessor. While a Microcontroller is a complete self-sufficient system with on chip I/O ports, Memory and various other features.

## The Atmega16 Microcontroller:



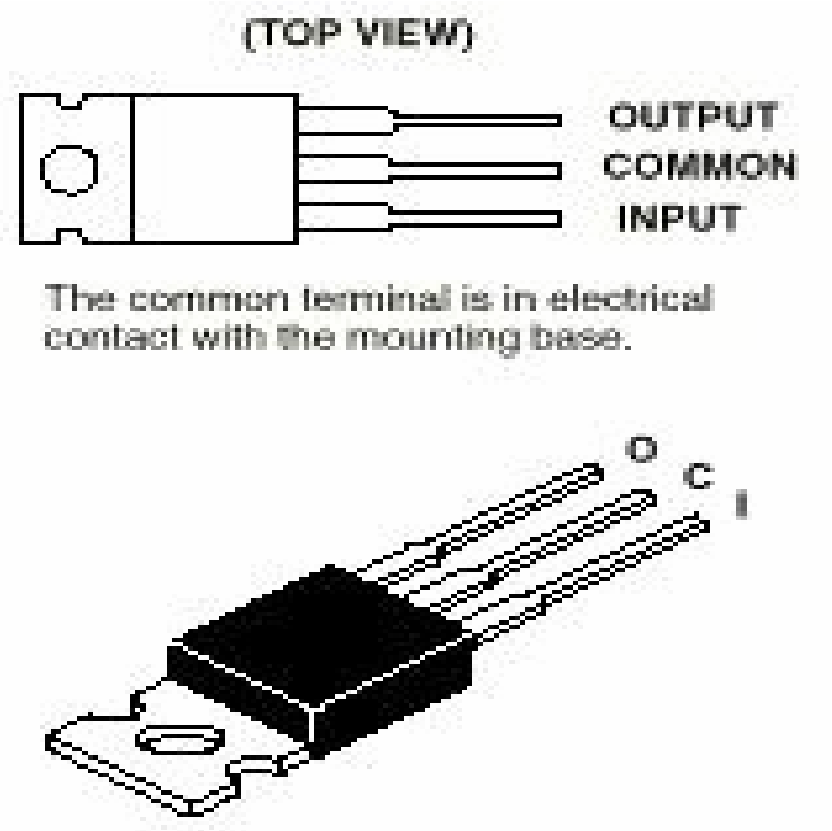
The Atmega16 is an 8-bit Microcontroller. 8-bit means the Arithmetic and Logic Unit of the controller can handle 8-bit data at a time. The figure shows the pin layout of Atmega16. It has four I/O ports: Port A, B, C and D each of which can be configured as either input or output. Note the numbering of the pins. The numbering starts from left side of the notch goes till the bottom, then again continues from the right side bottom and continues upwards up till the right side of the notch. Pins PA0-PA7 indicates Port A, PB0-PB7 indicate Port B and so on. The ports are "Byte-addressable" as well as "Bit-addressable". There are two pins for providing power supply: VCC & AVCC. Both of them have to be connected to a constant +5V DC supply. There are two pins providing Ground. Both of them have to be connected to the 0V of the supply. AREF is the reference voltage for the Analog to Digital Converter. The external crystal oscillator has to be connected to XTAL1 and XTAL2. The RESET pin is used to reset the Microcontroller. It is active-low and hence to provide reset the pin has to be taken to 0V.

## Voltage Regulator:

A voltage regulator is needed to obtain a fixed DC supply voltage to be supplied to the Microcontroller. The 78xx are positive voltage regulators where "xx" indicates the output voltage level. 7805 is a +5V DC voltage regulator.

### 7805:

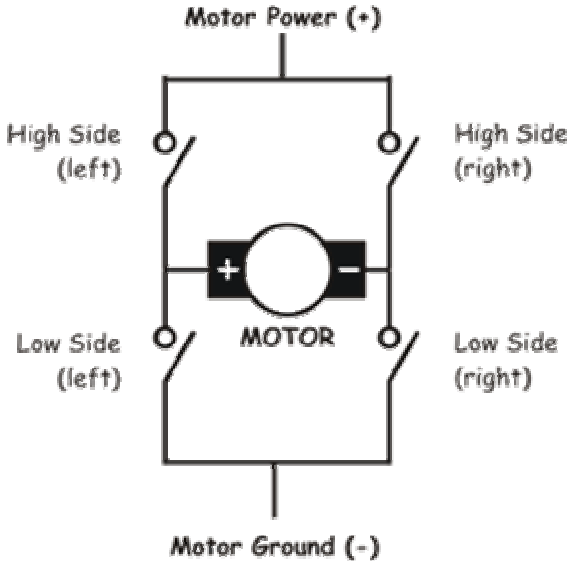
**7805** is an integrated three-terminal positive fixed linear voltage regulator. It supports an input voltage of 7 volts to 35 volts and output voltage of 5 volts. It typically has a current rating of 1 amp although both higher and lower current models are available. Its output voltage is fixed at 5.0V. The 7805 also has a built-in current limiter as a safety feature. The 7805 will automatically reduce output current if it gets too hot. The 7805 is one of the most common and well-known of the 78xx series regulators, as its small component count and medium-power regulated 5V make it useful for powering TTL devices.




## H-Bridge

Let's start with the name, H-bridge. Sometimes called a "full bridge" the H-bridge is so named because it has four switching elements at the "corners" of the H and the motor forms the cross bar.

The basic bridge is shown in the figure.

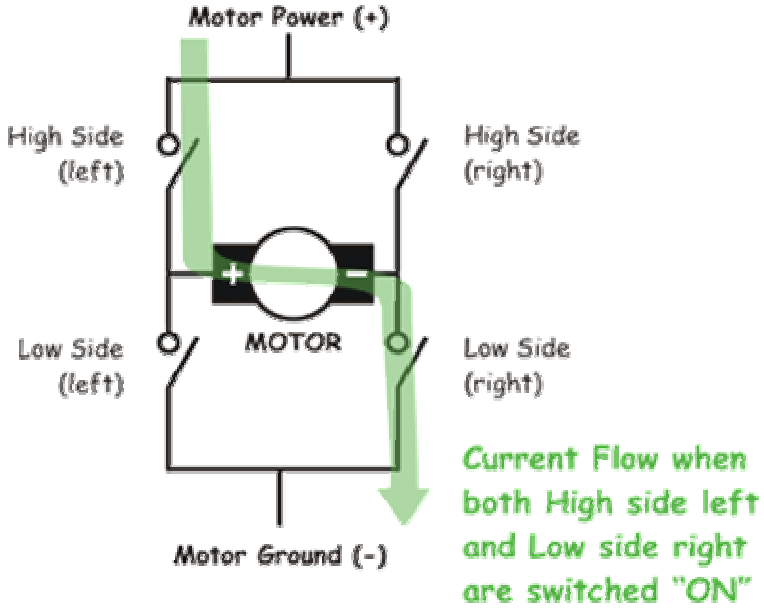



Of course the letter H doesn't have the top and bottom joined together, but hopefully the picture is clear. This is also something of a theme of this tutorial where I will state something, and then tell you it isn't really true :-).

The key fact to note is that there are, in theory, four switching elements within the bridge. These four elements are often called, high side left, high side right, low side right, and low side left (when traversing in clockwise order).

The switches are turned on in pairs, either high left and lower right, or lower left and high right, but never both switches on the same "side" of the bridge. If both switches on one side of a bridge are turned on it creates a short circuit between the battery plus and battery minus terminals. This phenomena is called shoot through in the Switch-Mode Power Supply (SMPS) literature. If the bridge is sufficiently powerful it will absorb that load and your batteries will simply drain quickly. Usually however the switches in question melt.

To power the motor, you turn on two switches that are diagonally opposed. In the picture to the right, imagine that the high side left and low side right switches are turned on. The current flow is shown in green.




The current flows and the motor begin to turn in a "positive" direction. What happens if you turn on the high side right and low side left switches? You guessed it, current flows the other direction through the motor and the motor turns in the opposite direction.

| High Side Left | High Side Right | Lower Left | Lower Right | Quadrant Description |
|---|---|---|---|---|
| On | Off | Off | On | Motor goes Clockwise |
| Off | On | On | Off | Motor goes Counter-clockwise |
| On | On | Off | Off | Motor "brakes" and decelerates |
| Off | Off | On | On | Motor "brakes" and decelerates |

The last two rows describe a maneuver where you "short circuit" the motor which causes the motors generator effect to work against itself. The turning motor generates a voltage, which tries to force the motor to turn the opposite direction. This causes the motor to rapidly stop spinning and is called "braking" on a lot of H-bridge designs. Of course there is also the state where all the transistors are turned off. In this case the motor coasts if it was spinning and does nothing if it was doing nothing.

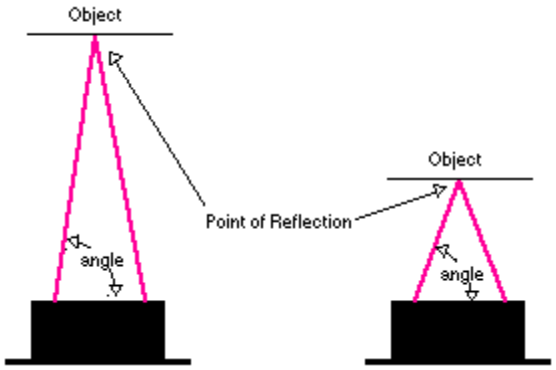
NextSapiens

## IR Sensors- Design and Working

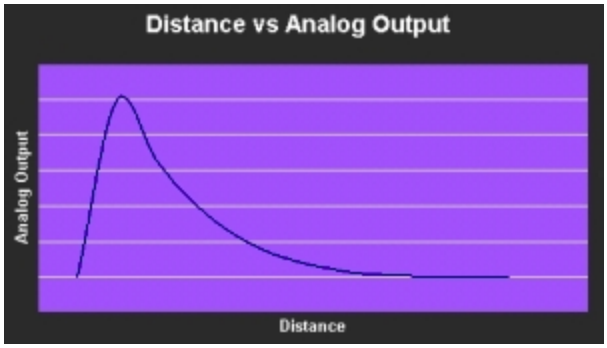
The **Sharp IR Range Finder** is probably the most powerful sensor available to the everyday robot hobbyist. It is extremely effective, easy to use, very affordable, very small, good range (inches to meters), and has low power consumption.

### How it Works

The Sharp IR Range Finder works by the process of **triangulation**. A pulse of light (wavelength range of 850nm +/-70nm) is emitted and then reflected back (or not reflected at all). When the light returns it comes back at an angle that is dependent on the distance of the reflecting object. Triangulation works by detecting this reflected beam angle - by knowing the angle, distance can then be determined.




The IR range finder receiver has a special precision lens that transmits the reflected light onto an enclosed linear CCD array based on the triangulation angle. The CCD array then determines the angle and causes the rangefinder to then give a corresponding *analog value to be read by your **microcontroller**. Additional to this, the Sharp IR Range Finder circuitry applies a modulated frequency to the emitted IR beam. This ranging method is almost immune to interference from ambient light, and offers amazing indifference to the color of the object being detected. In other words, the sensor is capable of detecting a black wall in full sunlight with almost zero noise.

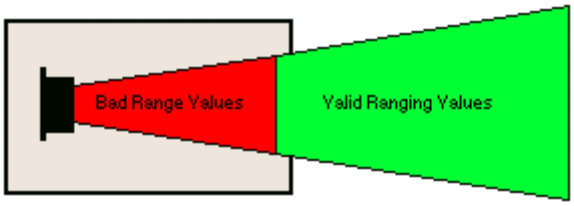



**Non-Linear Ouput**

The Sharp IR has a non-linear output. This means that as the distance increases linearly (by set increments), the analog output increases/decreases non-linearly. The image above is a typical expected output from your range finder. Notice the strange kink in the beginning of the graph. This is because the range finder is not capable of detecting very short distances. Refer to the particular range finder you are using to determine the range

**Disadvantages/Issues**

One major issue with the Sharp IR Range Finder and that is **going below the minimum sensor range**. This is when an object is so close the sensor cannot get an accurate reading, and it tells your robot that a really close object is really far. This is bad, as your robot then proceeds to ramp up in speed for a messy collision. Sonar also has this minimum range problem. The solution to this problem is to NOT put your sensor flush with the front of your robot, but to instead back the sensor into the robot so that the front of the robot is located before the minimum sensor range (refer to image).

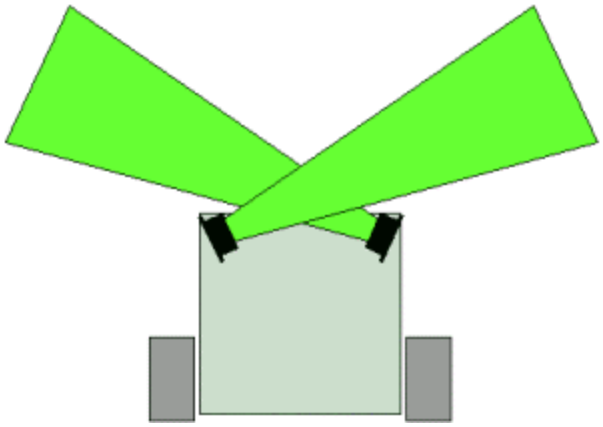



Another issue is the **high narrowness** of the IR beam. In reading sharp details and getting high accuracy, a thin beam is ideal. But the problem with a thin

beam is that if it is not pointed exactly at the object, the object is therefore invisible. A common joke in robotics is that a chair is the arch-rival of a small robot. Why? Because chair legs are really thin and easy to miss by a sensor. In contrast to the IR Range Finder would be the sonar. Sonar has really poor accuracy, but since it has a wide beam it can easily detect chair legs. Unfortunately you cannot tell the size or shape of an object with a cheap hobby sonar. Sonar also has a cone shaped beam (spreading out from the point of origin) and the Sharp IR Range Finder beam is more **football** shaped (the widest portion in the middle being about 16 cm wide).

An issue that these range finders have in common with sonar is **cross interference**. This means that the signal emitted by one sensor can potentially be read by another sensor and therefore give you bad readings. However, unlike sonar which have sound signals that can bounce off of multiple walls, you just need to make sure your IR beams do not cross in parallel (the wide parts of the football shaped beam not overlapping). This makes sense because you have over redundant sensors if the two beams cross.

### Techniques With the Sharp IR Range Finder



### Bumper Switch

The sharp IR can be used as a quick and easy front **non-contact robot bumper** on your robot. Just place two IR devices in front of your robot and cross beams as shown. Ideally you would prefer to use rangers that have wider beams. Note: A single sonar can do this job just as well.

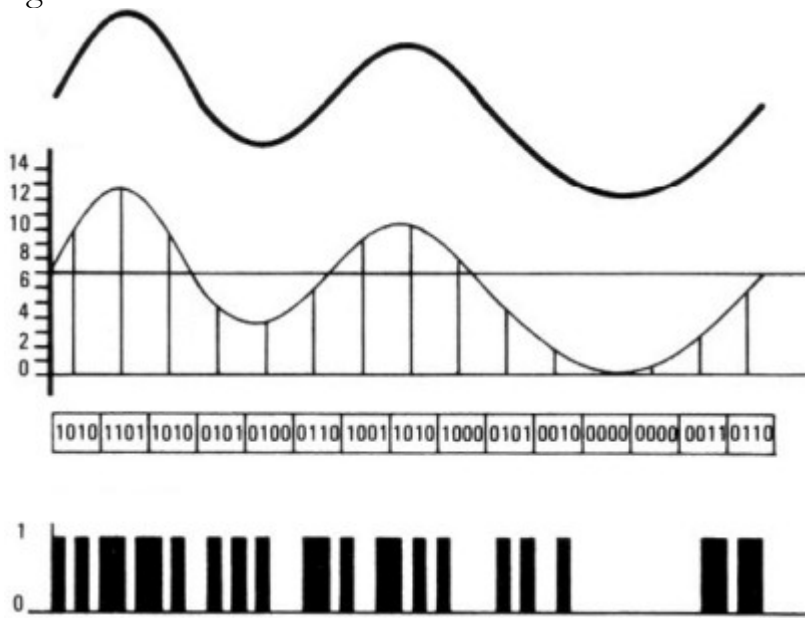
NextSapiens

## Analog Input Ports

Analog Ports are necessary to connect sensors to your robot. Also known as an analog to digital converter (ADC), they receive analog signals and convert them to a digital number within a certain numerical range.

So what is analog? Analog is a continuous voltage range and is typically found with sensors. However computers can only operate in the digital realm with 0's and 1's. So how does a microcontroller convert an analog signal to a digital signal?

First, the analog is measured after a predefined period of time passes. At each time period, the voltage is recorded as a number. This number then defines a signal of 0's and 1's as shown:




The advantage of digital over analog is that digital is much better at eliminating background noise. Cell phones are all digital today, and although the digital signal is less representative than an analog signal, it is much less likely to degrade since computers can restore damaged digital signals. This allows for a clearer output signal to talk to your mom or whoever. MP3's are all digital too, usually encoded in 128 bit. Higher bit rates obviously mean higher quality because they better represent the analog signal. But higher bit rates also require more memory and processing power.

Most Microcontrollers today are **8 bit**, meaning they have a range of 256 (2^8=256). There are a few that are 10 bit, 12 bit, and even 32 bit, but as you increase precision you also need a much faster processor.

What does this bit stuff mean for ADC? For example, suppose a sensor reads 0V to an 8 bit ADC. This would give you a digital output of 0. 5V would be 255. Now suppose a sensor gave an output of 2.9V, what would the ADC output be?

Doing the math:

2.9V/5V = X/255
X = 2.9*255/5 = **148**

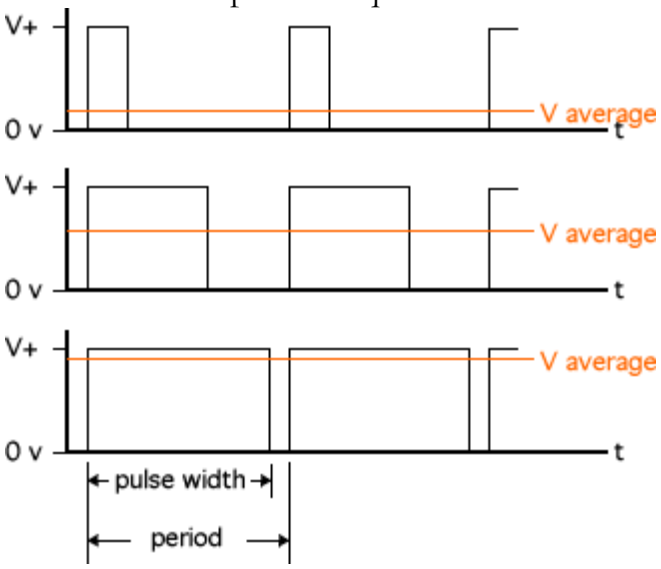
So how do you use an analog port? First make sure your sensor output does not exceed your digital logic voltage (usually 0V -> 5V). Then plug that output directly to the analog port.

This bit range could also be seen as a resolution. Higher resolutions mean higher accuracy, but occasionally can mean slower processing and more susceptibility to noise. For example, suppose you had a 3 bit controller which has a range of 2^3=8. Then you have a distance sensor that outputted a number 0->7 (a total of 8) that represents the distance between your robot and the wall. If your sensor can see only 8 feet, then you get a resolution of 1 bit per foot (8 resolution / 8 feet = 1). But then suppose you have an 8 bit controller, you would get 256/8=32 ~ 1 bit per centimeter - way more accurate and useful! With the 3-bit controller, you could not tell the difference between 1 inch and 11 inches.

## Digital I/O Ports

Digital ports are like analog ports, but with only 1 bit (2^1=2) hence a resolution of 2 - on and off. Digital ports obviously for that reason are rarely used for sensors; except for maybe on/off switches . . . What they are mostly used for is signal output. You can use them to control motors or LED's or just about anything. Send a high 5V signal to turn something on, or a low 0V to turn something off. Or if you want to have an LED at only half brightness, or a motor at half speed, send a square wave. Square waves are like turning something on and off so fast that its almost like sending out an analog voltage of your choice. Neat, huh?

This is an example of a square wave for **PWM**:




These squarewaves are called **PWM**, short for pulse width modulation. They are most often used for controlling servos or DC motor H-Bridges.

Also a quick side note, analog ports can be used as digital ports.

## Serial Communication, rs232

A serial connection on your microcontroller is very useful for communication. You can use it to program your controller from a computer, use it to output data from your controller to your computer (great for debugging), or even use it to operate other electronics such as digital video cameras. Usually the microcontroller would require an external IC to handle everything, such as an RS232.

I^2C (pronounced 'I-squared-C') is also useful for communicating, but I have never used it. Just make sure your controller has some method of communicating data to you for easy and effective debugging/testing of your robot programs. Its actually somewhat complicated, but usually the manufacturer has simplified it so all you have to do is plug-n-play and do a few print statements.