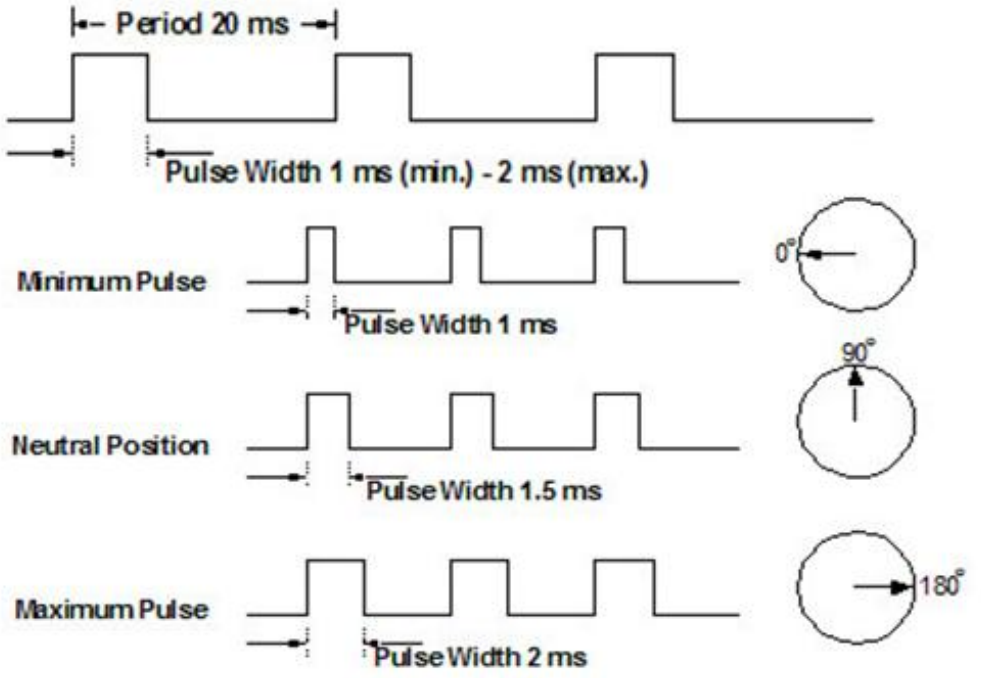
Servo Motor Control

Servos

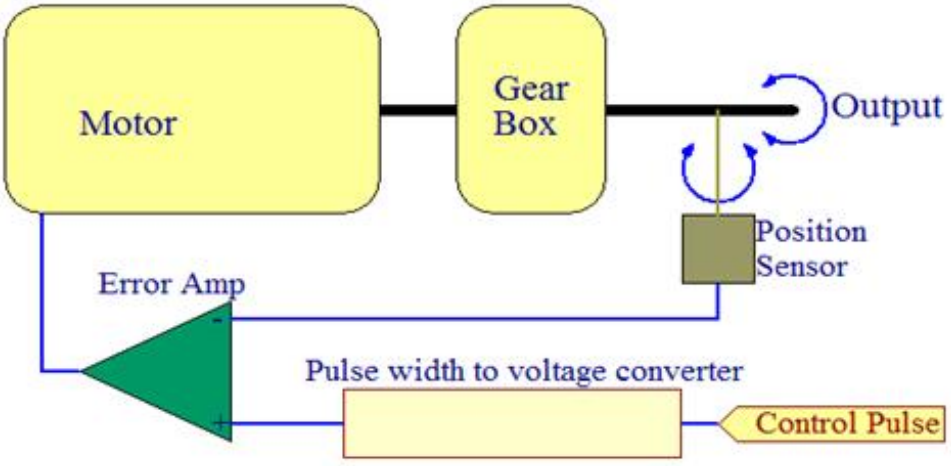
Servos are DC motors with built in gearing and feedback control loop circuitry. And no motor drivers required. They are extremely popular with robot, RC plane, and RC boat builders. Most servo motors can rotate about 90 to 180 degrees. Some rotate through a full 360 degrees or more.

However, servos are unable to continually rotate, means they can't be used for driving wheels, unless they are modified, but their precision positioning makes them ideal for robot legs and arms, rack and pinion steering etc. To use a servo, simply connect the black wire to ground, the red to a 4.8-6V source, and the yellow/white wire to a signal generator (such as from your microcontroller). Vary the square wave pulse width from 1-2 ms and your servo is now position/velocity controlled.

Pulse width modulation (PWM) is a powerful technique for controlling analog circuits with a processor's digital outputs. The general concept is to simply send an ordinary logic square wave to your servo at a specific wave length, and your servo goes to a particular angle . The wavelength directly maps to servo angle.



Basic working concept of servo motors -



How to derive your logic ???

>> we need a 20ms total pulse width (high+low).

>> Simultaneously we need to generate acurate pulse width to control angle of shaft.

>> In 8051 microcontroller world whenever we need to perform two task almost at same time like in this case we must use Interrupts. But here as we are using ATmega8 i.e. AVR MC there is no need to use interrupts at all as AVR has a very rich timer/counter feature. The inbuilt timer block can be used to generate highly accurate PWM signals.

Timer PWM

>> For a detailed tutorial on AVR Timers click here

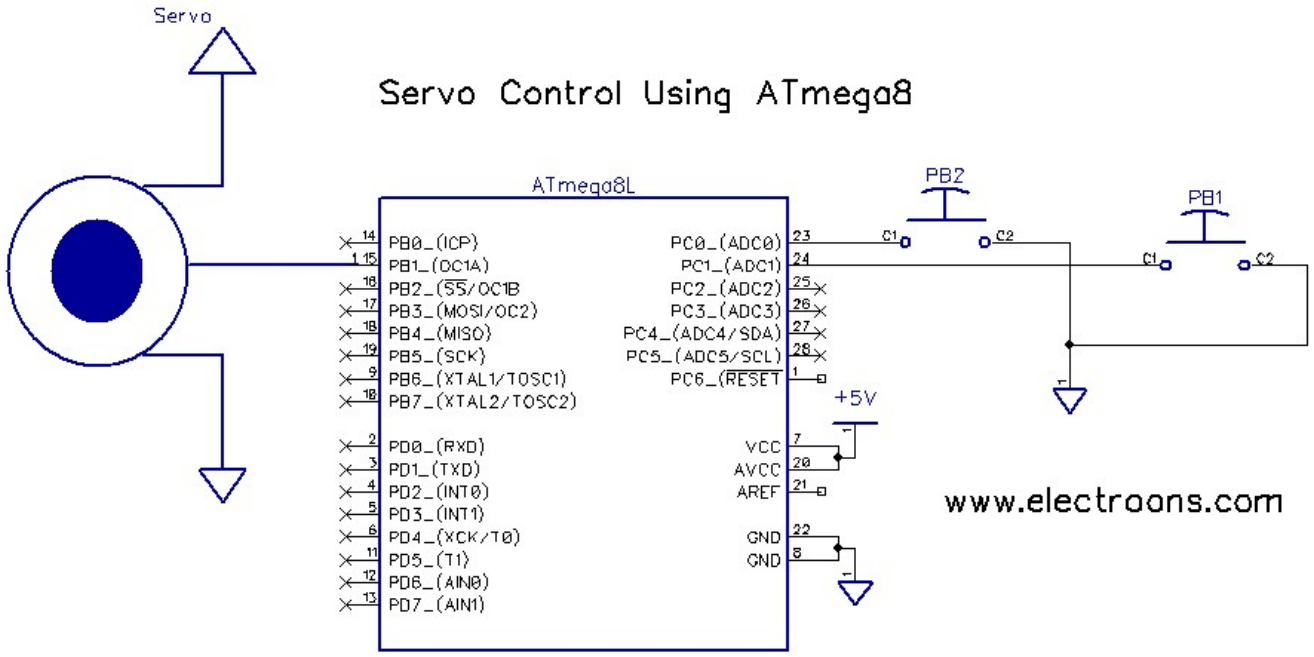
>> Here to control servo motor we are using "Fast PWM mode" of 16 bit Timer1 of ATmega8, with ICR1 as TOP and OCR1A update on BOTTOM.

>> It is kind of a complex logic, the desired 20msec pulse is generated when timer moves form BOTTOM to ICR1 value. We are setting ICR1 on a value which generates a total time of 20msec.

>> Once 20msec time period is generated now its time to generate different pulse widths within 20msec to rotate the servo shaft in different possible angles between 0-180 degrees.

>> we are varying the values in OCR1A which decides the ON time period of pulse, so by varying the values in OCR1A we can control the shaft position very easily.

Schematic Diagram -




```
/************************************************************
Example Servo Motor Control electroons.com AVR Tutorials
Control signal is connected to OCR1A i.e PIN15 in ATmega8
ATmega8 is running on 1 MHz internal RC oscillator

Now if ICR1A=20000;

Fpwm = Fosc/ n*(1+TOP);

Now we desire a PWM frequency of 50 Hz to generate 20msec time
period.

we have chosen values as follows -
```

```
F_osc = 1MHz

n (prescaling factor) = 1 ;

Top = ICR1 = 20000;

*************************************************************/

#include <avr\io.h>


int main(void)
{


        DDRC=0x00;      //PortC pins as input
        PORTC=0xFF;     //Enable internal pull ups
        DDRB=0xFF;      //Set PORTB1 pin as output

        //TOP=ICR1;
        //Output compare OC1A 8 bit non inverted PWM
        //Clear OC1A on Compare Match, set OC1A at TOP
        //Fast PWM
        //ICR1=20000 defines 50Hz PWM

        ICR1=20000;
        TCCR1A  = (0<<COM1A0)|(1<<COM1A1)|(0<<COM1B0)|
(0<<COM1B1)|(0<<FOC1A)|(0<<FOC1B)|(1<<WGM11)|(0<<WGM10);
        TCCR1B =  (0<<ICNC1)|(0<<ICES1)|(1<<WGM13)|(1<<WGM12)|
(0<<CS12)|(0<<CS11)|(1<<CS10);
        //start timer with prescaler 1 for 1 MHz

        while(1)
        {

                if(bit_is_clear(PINC, 0))
                {
                        //increase duty cycle
                        OCR1A+=10;
                        loop_until_bit_is_set(PINC, 0);
                }

                if(bit_is_clear(PINC, 1))
                {
                        //decease duty cycle
                        OCR1A-=10;
                        loop_until_bit_is_set(PINC, 1);
                }

        }//while(1) ends here

return 0;
} //main ends here
```

HOME PAGE

devesh@electroons.com
Best viewed at Firefox 1024x768 resolution