

'-----Title-----

' File.....servo2.pbp  
' Started....5/22/08  
' Microcontroller Used: Microchip Technology 16F88  
' microchip.com  
' PicBasic Pro Code: micro-Engineering Labs, Inc.  
' melabs.com

'-----Program Description-----

' Program makes servo rotate clockwise then counter-  
' clockwise using PAUSEUS command.  
' Discussion about basic servo pulse  
' control may be found at [www.seattlerobotics.org/guide/servos.html](http://www.seattlerobotics.org/guide/servos.html)  
' or [www.geocities.com/hobby\\_robotics/was.htm](http://www.geocities.com/hobby_robotics/was.htm)

'-----Related Lesson-----

' servo2.pbp is used in the lesson PIC PROGRAMMING 3 SERVOS at:  
' [http://cornerstonerobotics.org/curriculum/lessons\\_year2/erii13\\_pic\\_programming3\\_servos.pdf](http://cornerstonerobotics.org/curriculum/lessons_year2/erii13_pic_programming3_servos.pdf)

'-----Comments-----

' WITH THE PIC16F88, MAKE SURE TO HAVE SEPARATE POWER  
' SOURCES FOR THE PIC AND THE SERVO. MAKE SURE TO  
' HAVE A COMMON GROUND BETWEEN THE PIC AND SERVO. We use one 9V  
' battery and two 78L05 voltage regulators. See  
' discussion about voltage regulators at:  
' [http://cornerstonerobotics.org/curriculum/lessons\\_year2/erii3\\_diodes\\_power\\_supplies\\_voltage\\_reg.pdf](http://cornerstonerobotics.org/curriculum/lessons_year2/erii3_diodes_power_supplies_voltage_reg.pdf)

'-----New PicBasic Pro Command-----

' The PicBasic Pro Compiler Manual is on line at:  
' <http://www.microengineeringlabs.com/resources/index.htm#Manuals>  
'  
' PAUSEUS Period  
' Pause the program for Period microseconds  
' Look around page 113 in the PicBasic Pro Compiler Manual

'-----Variables-----

    i                  **VAR  BYTE**  '  BYTE to store counter, i  
    pulse\_width      **VAR  WORD**  '  WORD to store pulse\_width

'-----Initialization-----

    TRISB = 0                  '  Set all PORTB pins as outputs  
    PORTB = 0                  '  Sets all PORTB pins to LOW(0 volts)  
                                '  Make certain to include this  
                                '  initialization as it sets the

```
' proper polarity of pulses in
' the PULSOUT command.
' To set just one pin such as RB0, to
' LOW, enter PORTB.0 = 0.

OSCCON = $60          ' Sets the internal oscillator in the
                     ' 16F88 to 4 MHz

'-----Main Code-----

loop:

' Servo clockwise position:

    FOR i = 1 TO 100    ' FOR..NEXT loop determines the number
                       ' of pulses sent to the servo, therefore
                       ' the time the servo remains in position.
                       ' Since each pulse period is 20 ms,
                       ' the time for the servo to move to this
                       ' position and remain there is 2 seconds:
                       ' 20 ms/pulse * 100 pulses = 2000 ms,
                       ' 2000 ms = 2 seconds

    pulse_width = 2000 ' Set pulse_width to 2000

    HIGH 0             ' Leading edge of pulse

    PAUSEUS pulse_width ' Length of pulse_width in microseconds
                       ' 2000 us = 2 ms
                       ' The pulse remains HIGH for 2 ms.

    LOW 0              ' Falling edge of pulse

    PAUSEUS 20000-pulse_width ' LOW for 20 ms period - pulse_width
                              ' This equation keeps the period of
                              ' the servo pulse a constant 20 ms.
                              ' In this case, HIGH for 2 ms and
                              ' LOW for 18 ms = 20 ms.

    NEXT i            ' Go back to the FOR statement and do
                     ' next count

' Servo counter-clockwise position:

    FOR i = 1 TO 25    ' Since each pulse period is 20 ms,
                       ' the time for the servo to move into
                       ' position and remain is 0.5 seconds.
                       ' 20 ms * 25 = 500 ms = 1/2 sec

    pulse_width = 1000 ' Set pulse_width to 1000

    HIGH 0             ' Leading edge of pulse into PWM input

    PAUSEUS pulse_width ' Length of pulse_width in microseconds
```

```
                                ' 1000 us = 1 ms
                                ' The pulse remains HIGH for 1 ms.

LOW 0                            ' Falling edge of pulse

PAUSEUS 20000-pulse_width      ' LOW for 20 ms period - pulse_width
                                ' This equation keeps the period of
                                ' the servo pulse a constant 20 ms.
                                ' In this case, HIGH for 1 ms and
                                ' LOW for 19 ms = 20 ms.

NEXT i                            ' Go back to the FOR statement and do
                                ' next count

GOTO loop                        ' Makes the program run forever.

END
```