

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

' File.....sonar_car_d.pbp
' Started....2/23/10
' Microcontroller used: Microchip Technology 16F88
' microchip.com
' PBPro Code, micro-Engineering Labs, Inc.
' melabs.com

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

' Final in a series of 4 programs that takes the class
' through the development of the final program sonar_car1.pbp.
' This program adds avoidance if the car bumps into an object.

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

' MAKE SURE that the servo power source is separate from the
' power source for the PIC16F88 microcontroller.

' Also, MAKE SURE the PULSOUT pin to drive the servo is set
' LOW to establish the correct polarity of the servo pulse.

' To activate RA6 and RA7 as digital I/O pins:
' * Push Compile and Program button
' * The meProg dialog box appears
' * Press the C command button
' * The meProg - Configuration dialog box appears
' * On the Oscillator row, press the drop-down list box
' * Highlight the INTRC option
' * Now program the chip

'-----New PicBasic Pro Commands-----

' SELECT CASE
' See around page 129 at: <http://www.melabs.com/downloads/pbpm304.pdf>

'-----PIC Connections-----

16F88 Pin	Wiring
RA0	LCD pin 11(DB4)
RA1	LCD pin 12(DB5)
RA2	LCD pin 13(DB6)
RA3	LCD pin 14(DB7)
RA4	LCD Register Select(RS)
RA6	Left Bumper Switch
RA7	Right Bumper Switch
RB0	PWM Motor 2 Input into SN755410
RB1	Direction Motor 2 Input into SN754410
RB2	PWM Motor 1 Input into SN755410
RB3	Direction Motor 1 Input into SN754410
RB4	Emitter Pin on Sharp SRF04 Ultrasonic Range Finder
RB5	Echo Pin on Sharp SRF04 Ultrasonic

```
'
                                Range Finder
'   RB6                         Control Pin to Hobby Servo
'   RB7                         LCD Enable(E)
'   Vdd                         +5 V
'   Vss                         Ground
'   MCLR                        4.7K Resistor to +5 V

'-----LCD Connections-----
'
'   LCD Pin                      Wiring
'   -----                      -----
'   1                            Ground(Vss)
'   2                            + 5v(Vdd)
'   3                            Center of 20K Pot(Contrast)
'   4                            RA4(Register Select,RS)
'   5                            Ground(Read/Write,R/W)
'   6                            RB7(Enable)
'   7                            No Connection(DB0)
'   8                            No Connection(DB1)
'   9                            No Connection(DB2)
'  10                            No Connection(DB3)
'  11                            RA0(DB4)
'  12                            RA1(DB5)
'  13                            RA2(DB6)
'  14                            RA3(DB7)

'-----Defines-----
'
' To make room at PIC16F88 pin RB3 for the Direction Motor 1
' operation, the LCD enable function (default at RB3)
' must be moved to RB7. The following DEFINES activate the move.

DEFINE LCD_EREG    PORTB    ' Set LCD Enable PORT to PORTB
DEFINE LCD_EBIT    7        ' Set LCD Enable pin to RB7

'-----Constants-----

conv_to_in CON  15          ' Conversion factor to convert
                                ' sonar readings to inches

left_time  CON  2100        ' Set maximum left turn time
                                ' Adjust these time settings
                                ' to correspond to the maximum
                                ' angle position of the ultrasonic
                                ' range finder

right_time CON  2100        ' Set maximum right turn time

'-----Switch Input Pins-----

left_switch VAR PORTA.6    ' Labels PORTA.6 as left_switch
right_switch VAR PORTA.7    ' Labels PORTA.7 as right_switch

'--SN754410 H-Bridge Control Pins--
```

```
pwm_motor2    VAR PORTB.0    ' Labels PORTB.0 as pwm_motor2
dx_motor2     VAR PORTB.1    ' Labels PORTB.1 as dx_motor2
pwm_motor1    VAR PORTB.2    ' Labels PORTB.2 as pwm_motor1
dx_motor1     VAR PORTB.3    ' Labels PORTB.3 as dx_motor1

'-----Sonar Control Pins-----

emit_pin      VAR PORTB.4    ' Labels PORTB.4 as emit_pin
echo_pin      VAR PORTB.5    ' Labels PORTB.5 as echo_pin

'-----Servo Control Pin-----

servo_pin     VAR PORTB.6    ' Labels PORTB.6 as servo_pin

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

p0            VAR BYTE      ' BYTE to store servo pulse period
c0            VAR WORD      ' WORD for counter
num           VAR BYTE      ' BYTE for array numbers
position      VAR BYTE[7]   ' BYTE for angle array
position_max  VAR BYTE      ' BYTE for angle of maximum reading
dx            VAR WORD      ' WORD for sonar input
dx_in        VAR WORD[7]    ' WORD for distance converted to
              ' inches array
dx_in_max     VAR WORD      ' WORD for maximum distance reading

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

ANSEL = 0     ' Configure all ADC pins to digital
              ' operation since not using ADC
              ' (Analog to Digital Converter)

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

'-----PORT Configurations-----

PORTA = %00000000 ' Set all PORTA pins to LOW
PORTB = %00000000 ' MAKE SURE THE PULSOUT PIN TO THE SERVO
                  ' IS SET LOW TO ESTABLISH THE PROPER
                  ' POLARITY OF THE SERVO PULSE. The PULSOUT
                  ' pin in this program is PORTB.6,(RB6).

TRISA = %11000000 ' Set switch pins,(RA6 & RA7), as inputs
TRISB = %00100000 ' Set echo input pin,(RB5), as an input

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

loop:

    num = 0      ' Set num to 0

    dx_in_max = 0 ' Set dx_in_max to 0

    ' Move servo to starting position:
```

```
FOR c0 = 1 TO 20           ' Send out PULSOUT command 20 times

PULSOUT servo_pin,70      ' Send servo pulse signal to servo_pin
                          ' (PORTB.6) for 0.7 ms. Pulse out time
                          ' The period,(70) is multiplied by the
                          ' increment for a 4 MHz oscillator
                          ' (10 usec) to get a pulse out time
                          ' of 700 us or 0.7 ms.

PAUSE 20                  ' Pause 20 msec

NEXT c0

' Pan servo across front of car:

FOR p0 = 70 TO 208 STEP 23 ' Rotate servo counter-clockwise
                          ' through 7 positions, 1 starting
                          ' position (p0 = 70) + 6 steps of 23.

FOR c0 = 1 TO 15          ' Send out PULSOUT command 15 times

PULSOUT servo_pin,p0      ' Send servo pulse signal to servo_pin
                          ' (PORTB.6). Pulse out time varies from
                          ' 0.7 msec to 2.08 ms.

PAUSE 20                  ' Pause 20 msec

NEXT c0

' Take readings at each servo position; record data in dx_in &
' position arrays:

PULSOUT emit_pin,1        ' Emit sonar pulse

PULSIN echo_pin,1, dx     ' Receive sonar reflected pulse back

dx_in[num] = dx/conv_to_in ' Fill dx_in array (dx_in[0] to
                          ' dx_in[6] with distances converted
                          ' to inches

position[num] = num       ' Assign numbers to position array

' Select maximum distance recorded and corresponding position:

IF dx_in[num] > dx_in_max THEN ' Make comparison of current dx_in[num]
                              ' value to dx_in_max. If the
                              ' comparison is true, dx_in_max

dx_in_max = dx_in[num]      ' If the comparison is true, dx_in_max
                              ' is assigned the value of dx_in[num].

position_max = position[num] ' If the comparison is true,
                              ' position_max is assigned the
                              ' value of position[num].
```

```
ELSE                                     ' If the comparison in the IF..THEN
                                         ' command is false, continue to
                                         ' next command.

ENDIF

' Display maximum distance recorded and corresponding position:

LCDOUT $FE,1,"dx_in_max = ", #dx_in_max   ' Display dx_in_max in
                                         ' inches

LCDOUT $FE,$C0,"position_max = ", #position_max
                                         ' Display position_max number
num = num + 1

NEXT p0                                  ' Go to next value of p0

'Rotate car to position corresponding to position_max:

SELECT CASE position_max                 ' Use SELECT CASE command to select
                                         ' direction of wheel rotation and
                                         ' rotate car to direction that
                                         ' corresponds to the direction
                                         ' of position_max.

CASE 0                                   ' If CASE 0,(angle_max = 0) is true,
                                         ' then execute statements that
                                         ' follow CASE 0.

    LOW dx_motor1                        ' Direction input into H-bridge driver
                                         ' (SN754410) for Motor 1.
                                         ' LOW makes the motor rotate in reverse

    HIGH pwm_motor1                      ' Set PWM input into H-bridge driver
                                         ' (SN754410) for Motor 1. Set PWM
                                         ' for 100% duty cycle (HIGH).

    PAUSE right_time                     ' Pause value of right_time constant.
                                         ' Right_time is the time of maximum
                                         ' rotation to the right.
                                         ' The PAUSE values must be determined
                                         ' experimentally for each car and sonar
                                         ' mounting.

CASE 1                                   ' Other cases have similar comments as
                                         ' CASE 0.

    LOW dx_motor1
    HIGH pwm_motor1
    PAUSE 2 * right_time /3

CASE 2
    LOW dx_motor1
    HIGH pwm_motor1
    PAUSE right_time /3

CASE 3
    GOTO forward                          ' Do not rotate if position_max faces
```

```
                                ' forward
CASE 4
    LOW dx_motor2
    HIGH pwm_motor2
    PAUSE left_time /3
CASE 5
    LOW dx_motor2
    HIGH pwm_motor2
    PAUSE 2 * left_time /3
CASE 6
    LOW dx_motor2
    HIGH pwm_motor2
    PAUSE left_time

END SELECT                                ' End SELECT CASE command

forward:

HIGH dx_motor2 : HIGH dx_motor1          ' Direction input into H-bridge
                                           ' driver (SN754410) for Motors
                                           ' 1 and 2. HIGH makes the motors
                                           ' rotate in the forward direction

HIGH pwm_motor2 : HIGH pwm_motor1        ' Set PWM input into H-bridge
                                           ' driver (SN754410) for Motors
                                           ' 1 and 2. Set PWM for 100%
                                           ' duty cycle (HIGH).

FOR c0 = 1 TO 400                          ' The forward time is in 10 ms
                                           ' segments to constantly poll the
                                           ' left and right switches.

IF left_switch = 1 THEN GOTO right        ' If left switch is pressed, go
                                           ' to "right" label

IF right_switch = 1 THEN GOTO left        ' If right switch is pressed, go
                                           ' to "left" label

PAUSE 10                                    ' Go forward for 10ms

NEXT c0

LOW pwm_motor2 : LOW pwm_motor1          ' Turn Motors 1 and 2 off.
                                           ' Set PWM input into H-bridge
                                           ' driver (SN754410) for Motors
                                           ' 1 and 2.
                                           ' Set PWM for 0% duty cycle (LOW).

GOTO loop                                  ' Jump to loop label

' When the right bumper switch is pushed, left will
' make the car backup and turn to the left.

left:                                       ' Backup and turn left subroutine
```

```
' Backup
  LOW dx_motor2 : LOW dx_motor1      ' LOW makes the motor rotate in
                                      ' the reverse direction

  HIGH pwm_motor2 : HIGH pwm_motor1  ' Set PWM input into H-bridge
                                      ' driver (SN754410) for Motors
                                      ' 1 and 2. Set PWM for 100%
                                      ' duty cycle (HIGH).

  PAUSE 500                          ' Backup for 500 ms or 0.5 seconds

' Turn left

  LOW pwm_motor2: PAUSE 1             ' Turn off Motor 2

  LOW dx_motor1                       ' Direction input into H-bridge driver
                                      ' (SN754410) for Motor 1.
                                      ' LOW makes the motor rotate in reverse

  HIGH pwm_motor1                     ' Set PWM input into H-bridge driver
                                      ' (SN754410) for Motor 1. Set PWM
                                      ' for 100% duty cycle.

  PAUSE 1000                          ' Pause 1000 ms or 1 second

  LOW pwm_motor1 : PAUSE 1           ' Turn off Motor 1

  GOTO loop

' When the left bumper switch is pushed, right will
' make the car backup and turn to the right.

right:                                ' Backup and turn right subroutine

  LOW dx_motor2 : LOW dx_motor1      ' Comments similiar to left
                                      ' subroutine

  HIGH pwm_motor2 : HIGH pwm_motor1

  PAUSE 500
  LOW pwm_motor1 : PAUSE 1
  LOW dx_motor2
  HIGH pwm_motor2
  PAUSE 1000
  LOW pwm_motor2 : PAUSE 1
  GOTO loop

END
```