define OSC 8
OSCCON.0 = 1 : OSCCON.4 = 1 : OSCCON.5 = 1 : OSCCON.6 = 1
'Set processor speed for PicBasicPro

ANSEL = 0
'Make ports digital.

PORTB = %00000000
'Set portb to all low.

TRISA = %01000000
'Traces inputs and outputs on all pins.

TRISB = %00000000
TRISC = %00001111
TRISD = %00000011

LCDCON = %00000000
'Special config registers for pic16f917.

LCDS1 = %00000000
'View datasheets for specifics. Will make certain pins

LCDS2 = %00000000
'work when they previously didn't. Not associated with

the LCD controls defined below.

xaxis var PORTD.0
'y variable resistor within xaxis of joystick

yaxis var PORTD.1
'yvariable resistor within yaxis of joystick

trans var PORTD.2
'pin connected to transmitter

left var PORTC.0
'Left on 5 position button

right var PORTC.1
'right on 5 position button

up var PORTC.2
'up on 5 position button

down var PORTC.3
'down on 5 position button

press var PORTA.6
'press on 5 position button

buzz var PORTD.3
'buzzer

xval var word
'Xaxis value from rctime from joystick

yval var word
'yaxis value from rctime from joystick

xtran var byte
'value to be transmitted calculated from xval

ytran var byte
'value to be transmitted calculated from yval

xscale var byte
'scale values set to be used later in equations

yscale var byte

xrest var byte
'the center x/ytran values, usually 275 are adjusted by the

yrest var byte
'trim values

i var byte
'Used for counting

j var byte
'Traces menu number

u var byte
'Manages lcd updates

k var bit
'Up ----positive edge trigger detect

l var bit
'Left --positive edge trigger detect

m var bit
'Down --positive edge trigger detect

n var bit
'Right -positive edge trigger detect

b var bit
'Press -positive edge trigger detect

b var bit
'Used with debug menu j=4

time var byte
'time to pause to charge capacitors.

ptime var byte
'Additional time to pause to make total pause time=20ms.

speed var byte
'Speed value for transmission.

ballast var byte
'Ballast value for trans.

accel var bit
'Accelerometer state bit (on/off) for trans.

trimvert var byte
'Vertical trim adjustment.
trimhorz var byte 'Horizontal trim adjustment.

xpls var word 'Calculates pulses to servos for debug purposes.
ypls var word

define LCD_DREG PORTB 'Sets operational bits to portb and
#define LCD_DBIT 0 'starting bit is pin 0 (pins 0-3).
define LCD_RSREG PORTB 'Set LCD RS pin to portb and
define LCD_RSBIT 4 'pin 4.
define LCD_EREG PORTB 'Set LCD E pin to portb and
define LCD_EBIT 5 'pins 5.
define LCD_BITS 4 'There are 4 operational bits.
define LCD_LINES 2 'There are 2 lines on the display.
define LCD_COMMANDUS 1500 'Pause 1500us between commands.
define LCD_DATAUS 44 'Pause 44us between bits of data.
define PULSIN_MAX 5000 'Why is this still here??? From earlier debugging.

time=5 'Set initial Values.
ptime=9
i=0
j=0
e=0
speed=5
accel=1
ballast=0
trimvert=6
trimhorz=6

low buzz 'Make sure buzzer is off.

pause 1000 'calibration block
lcdout $FE,1, "The Cake's a lie" 'LCDOUT means give a command to the screen.
lcdout $FE,$C0,"Mech307",200,210 '"$FE" means send the following command.
pause 2000 'Command "1" means clear screen.
lcdout $FE,1, "Don't Touch" 'Command "$c0 means goto second line.
lcdout $FE,$C0,"joystick " 'text in quotations is then displayed.
pause 500
i=4 'perform an onscreen countdown before calibration.
repeat
i=i-1
lcdout $FE,1, "Don't Touch"
lcdout $FE,$C0,"joystick ",dec i 'dec i displays the decimal version of i.
pause 200
until i=0
high xaxis 'makes input high
pause time 'charges capacitor for time
rctime xaxis,1,xscale 'measures amount of time to discharge capacitor
high yaxis 'value is stored in xscale and used later
pause time 'in the calculations.
rctime yaxis,1,yscale

lcdout $FE,1, "Calibrated"
lcdout $FE,$C0,"Press to begin" ' "
pause 500
lcdout $FE,1, "Press center to"
lcdout $FE,$C0,"advance menu"

repeat 'waits for key to be pressed
until press==0 'key is normally high
pause 50
repeat 'this initiates the main part of the prog.
until press==1
top
' Top of the program.
if up=1 then k=0
' When a key is not being pressed, its corresponding
if down=1 then m=0
' bit will go to 0.
if left=1 then l=0
if right=1 then n=0
if press=1 then b=0
if e=1 then j=0
' If e=1 then menu j=4 is active but set j=0 now.
if left=0 and press=0 and j=0 then
ballast=0
pause 1000
goto jump
endif
if right=0 and press=0 and j=0 then
' These are conditions to go into
ballast=2
pause 1000
goto jump
endif
if press=0 and b=0 then
' Remember, all buttons I use are normally high.
j=j+1
' Advances menu.
b=1
' Set b bit to 1 so that this command is not
if e=1 then j=0
' repeated until after press is released.
e=0
' Special conditions for e=1.
if j>2 then j=0
' If menu gets to 3 then go back to 0.
endif
if up=0 and k=0 then
select case j
' Says "Do this if up goes low and wasn't low already".
case 0
speed=speed+1
' First menu, increase speed.
case 1
trimvert=trimvert+1
' Second menu, increase vertical trim.
end select
k=1
endif
if down=0 and m=0 then
select case j
' The below are similar to the "up" key above
case 0
speed=speed-1
case 1
trimvert=trimvert-1
case 2
j=4
e=1
end select
m=1
endif
if left=0 and l=0 then
select case j
' Say's "As this if left goes low and wasn't low already".
case 1
trimhorz=trimhorz-1
case 2
j=3
end select
l=1
endif
if right==0 and n==0 then
  select case j
  case 1
    trimhorz=trimhorz+1
  case 2
    accel=accel+1
  end select
  n=1
endif

if ballast==0 then jump  'If in surface mode, skip the following commands.
if left==0 and j==0 then
  ballast=1
  goto jump  'While left is pressed in first menu, 'set ballast to empty
else
  ballast=2
endif

if right==0 and j==0 then
  ballast=3  'set ballast to fill
  goto jump
else
  ballast=2
endif

jump  'a place to jump to.

if speed==3 then speed=4  'Sets upper and lower limits on speed and trim.
if speed==7 then speed=6
if trimvert==0 then trimvert=1
if trimvert==12 then trimvert=11
if trimhorz==0 then trimhorz=1
if trimhorz==12 then trimhorz=11

if ballast == 0 then i=0  'while in surface mode, set i=0
if ballast != 0 then  'in submerge mode, increase i by 1.
i=i+1
  if i==250 then i=240
select case i
  case 1
    high buzz
  case 30
    low buzz
  case 60
    high buzz
  case 90
    low buzz
  case 120
    high buzz
  case 150
    low buzz
end select

if u==3 then  'this makes the lcd screen update once for every
  u=0
else
  u=u+1
  goto main
endif
if e==1 then  'Here is the special e condition for menu j=4.

if j==0 and speed==4 then   'Below are all the conditions for display.
lcdout $FE,1, " Reverse   
endif
if j==0 and speed==5 then
lcdout $FE,1, " Full Stop  
endif
if j==0 and speed==6 then
lcdout $FE,1, " Forward  
endif

if j==0 then
if ballast==0 then
lcdout $FE,$C0, " Surface   
endif
if ballast==1 then
lcdout $FE,$C0, " Submerged RISE 
endif
if ballast==2 then
lcdout $FE,$C0, " Submerged     
endif
if ballast==3 then
lcdout $FE,$C0, " Submerged SINK 
endif
endif

if j==1 and trimvert==1 then
lcdout $FE,1, "vert:",248,"-----   
endif
if j==1 and trimvert==2 then
lcdout $FE,1, "vert: ",127,"-----   
endif
if j==1 and trimvert==3 then
lcdout $FE,1, "vert: ",127,"--   
endif
if j==1 and trimvert==4 then
lcdout $FE,1, "vert:  ",127,"-   
endif
if j==1 and trimvert==5 then
lcdout $FE,1, "vert:  ",127,"   
endif
if j==1 and trimvert==6 then
lcdout $FE,1, "vert:   ",127,"   
endif
if j==1 and trimvert==7 then
lcdout $FE,1, "vert:    ",127,"   
endif
if j==1 and trimvert==8 then
lcdout $FE,1, "vert:     ",127,"   
endif
if j==1 and trimvert==9 then
lcdout $FE,1, "vert:      ",127,"   
endif
if j==1 and trimvert==10 then
lcdout $FE,1, "vert:       ",127,"   
endif
if j==1 and trimvert==11 then
lcdout $FE,1, "vert:         ",127,"   
endif

if j==1 and trimhorz==1 then
lcdout $FE,$C0, "horz:",248,"-----   
endif
endif
if j==1 and trimhorz==2 then
    lcdout $FE,$C0, "horz: ",127,"----
endif
if j==1 and trimhorz==3 then
    lcdout $FE,$C0, "horz: ",127,"--
endif
if j==1 and trimhorz==4 then
    lcdout $FE,$C0, "horz: ",127,"- 
endif
if j==1 and trimhorz==5 then
    lcdout $FE,$C0, "horz: ",-",126,
endif
if j==1 and trimhorz==6 then
    lcdout $FE,$C0, "horz: --",126,
endif
if j==1 and trimhorz==7 then
    lcdout $FE,$C0, "horz: ---",126,
endif
if j==1 and trimhorz==8 then
    lcdout $FE,$C0, "horz: -----",248
endif

if j==2 then
    lcdout $FE,1,127, "Disp x-y values"
    if accel==1 then
        lcdout $FE,$C0,"Disable Accelr?",126
    else
        lcdout $FE,$C0," Enable Accelr?",126
    endif
endif

if j==3 then
    'the first debug menu
    lcdout $FE,1,"xpls:", dec xpls, " y:",dec ypls
    lcdout $FE,$C0,"xtran:",dec xtran," y:",dec ytran
endif

if j==4 then
    'the second debug menu
    lcdout $FE,1,"Ballast:", dec ballast
    lcdout $FE,$C0,"Speed:",dec speed
endif

main
    'here is the main part of the program

high xaxis
    'Makes input high.
pause time
    'Charges capaciter for time.
rctime xaxis,1,xval
    'Measures amount of time to discharge, store in xval.
high yaxis
    'Same with yaxis.
pause time
rctime yaxis,1,yval

xtran=(xval*125/xscale)+((trimhorz-6)*5) 'Calculates trans value from val,
ytran=(yval*125/yscale)+((trimvert-6)*5) 'scale, and trim.
xrest=(125+(trimhorz-6)*5) 'Calcs center tran position from trim.
yrest=(125+(trimvert-6)*5)

' the below makes sure that the trim isn't putting the tran value
' outside of the limits of 0 and 250.
if ((xval*125/xscale)+((trimhorz-6)*5))>250 then xtran=250
if ((xval*125/xscale)+((trimhorz-6)*5)+50)<50 then xtran=0
if ((yval*125/yscale)+((trimvert-6)*5))>250 then ytran=250
if ((yval*125/yscale)+((trimvert-6)*5)+50)<50 then ytran=0

'makes a small dead zone in the tran using rest.
if (xtran<(xrest+3) and xtran>(xrest-3)) then xtran=xrest
if (ytran<(yrest+3) and ytran>(yrest-3)) then ytran=yrest

xpls=xtran*7/5+100  ' calcs pulses for display in first debug menu.
ypls=ytran*7/5+100

serout trans,0,"["sync",xtran,ytran,]speed,ballast,accel]"
' output serial data on pin trans in mode 0 or T2400. The first
' piece of data sent is the word "sync". The pic recieving won't
' store any data until it sees this word. Each additional byte or bit
' is then transmitted. Make sure that bytes and bits are defined the
' same in both transmitting and recieving pics.

pause ptime
' this pause is to ensure that my servos on the other pic are not
' being sent more than 50 pulses per second. The total pause is 20ms
' between each transmission.

goto top  ' Goto the top of the program.

' By Brandon McDowall