Codewarrior:

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"  /* derivative information */
#include "SCI.h"

char string[20];
unsigned short val;
int Interrupt = 0;
float analog;
unsigned short angle;
int test;

//---------------------OutCRLF---------------------
// Output a CR,LF to SCI to move cursor to a new line
// Input: none
// Output: none
// Toggle LED each time through the loop

void OutCRLF(void){
  SCI_OutChar(CR);
  SCI_OutChar(LF);
  PTJ ^= 0x20;          // toggle LED D2
}

void setCLK6(void){
  CPMUPROT = 0;       //disable clock write protection
  CPMUCLKS = 0x80;      //set PLLSEL=1
  CPMUOSC = 0x80;       // set OSCE=1
  CPMUREFDIV = 0x41;    //set reference frequency to 8/2=4MHZ
  CPMUSYNR = 0x05;      //set VCOCLK frequency to 2*4*(5+1)=48MHZ
  CPMUPOSTDIV = 0x03;   //set pll frequency to 48/(3+1)=12MHZ
  while(CPMUFLG == 0);  //wait for pll to engage
  CPMUPROT = 1;         //enable clock write protection
}

void delay1ms(unsigned int multiple){
  int i;                //loop contorl variable
  TSCR1 = 0x90;         //enable timer
  TSCR2 = 0x00;         //prescaler=1
  TIOS |= 0x01;
  TC0 = TCNT + 6000;
  TIE = 0x00;
  for(i=0;i<multiple;i++){
    while(!(TFLG1_C0F));
    TC0 += 6000;
  }
  TIOS &= ~0x01;
  TIE = 0x03;
}
void main(void) {
    setCLK6();
    DDRJ = 0xFF;
    //interrupt configuration
    TSCR1 = 0x90;
```

```
    TSCR2 = 0x00;
    TIOS = 0xFC;
    PERT = 0x03;
    TCTL3 = 0x00;
    TCTL4 = 0x0A;
    TIE = 0x03;
    //analog to digital configuration
    ATDCTL1 = 0x4F;       //set for 12 bit resolution
    ATDCTL3 = 0x88;       //right justified,one sample per sequence
    ATDCTL4 = 0x02;       //ATD clock = 6MHZ/2*(2+1)=1MHZ
    ATDCTL5 = 0x24;       //continuous conversion for one channel(channel 4)

    //enable interrupt
    EnableInterrupts;
    SCI_Init(9600);
  DDR0AD = 0b00001111; // angle / 10
  DDR1AD = 0b01001111; // angle % 10
  ATDDIEN = 0x0000;
    PER1AD = 0x00;

    DDRT = 0b00000000;

  //mode 1 ()

  Interrupt=0;
    for(;;){
      if(Interrupt % 2 == 1){

          PTJ ^= 0x01;
          val=ATDDR0;
          SCI_OutUDec (val);
          OutCRLF();
          delay1ms(100);
          analog= ((val * 3.3/4095)-1.65)/0.3;

angle=(analog+(analog*analog*analog)/6+3*(analog*analog*analog*analog*analog)
/40+5*(analog*analog*analog*analog*analog*analog*analog)/112)*(180/3);
          // mode 1
          if(PTT == 0xC7){

          if(angle>=0 && angle<5){
           PT1AD=0b00000000;
           PT0AD=0b00000000;
          }
          if(angle>=5 && angle<15){
           PT1AD=0b00000001;
           PT0AD=0b00000000;
          }
          if(angle>=15 && angle<25){
           PT1AD=0b00000011;
           PT0AD=0b00000000;
          }
          if(angle>=25 && angle<35){
           PT1AD=0b00000111;
           PT0AD=0b00000000;
          }
          if(angle>=35 && angle<45){
```

```
            PT1AD=0b00001111;
            PT0AD=0b00000000;
           }
           if(angle>=45 && angle<55){
            PT1AD = 0b01001111;
            PT0AD=0b00000000;
           }
           if(angle>=55 && angle<65){
            PT1AD = 0b01001111;
            PT0AD=0b00000001;
           }
           if(angle>=65 && angle<75){
            PT1AD = 0b01001111;
            PT0AD=0b00000011;
           }
           if(angle>=75 && angle<85){
            PT1AD = 0b01001111;
            PT0AD=0b00000111;
           }
           if(angle>=85 && angle<90){
            PT1AD = 0b01001111;
            PT0AD=0b00001111;
           }
           }
           //mode 0
           if(PTT == 0xC3){

           PT0AD = angle / 10;
           PT1AD = angle % 10;
           }

      }
    }
}

interrupt  VectorNumber_Vtimch1 void ISR_Vtimch1(void){
    unsigned int temp;
    Interrupt++;
    temp = TC1;
}
```

MATLAB:

```
delete(instrfindall);
s = serial('COM1');
s.baudrate = 9600;
s.terminator = 'CR';
signal = animatedline('Color','b');
fopen(s);
time = 0;
while(1)
```

```
    digital = str2num(fgetl(s));
    disp(digital);
    analog = ((digital * 3.3 / 1023) - 1.65) / 0.3;
    angle = real(asin(analog)*180/pi);
    addpoints(signal, time, angle);
    drawnow
    time = time + 1;
end
fclose(s)
delete(s)
```