A Quick Programming Guide for Students Using the MSP430X2XX in Laboratory Projects

E. J. Seykora
Department of Physics
East Carolina University
Greenville, NC

Introduction

This guide is intended to serve as a fast introduction to the use of the msp430x2xx embedded microcontroller for use by students in Electronics and Advanced Laboratory environments. The intent is to allow an easy access to calls for I/O applications and serial communications transmitting data to external computers. It is assumed that the student has some understanding of c-programming and has been through the introduction to using the Code Composer Studio or IAR Embedded Workbench Kickstart, IDE's.

The IDE's and a discussion of the MSP430LaunchPad is available at:
http://processors.wiki.ti.com/index.php/MSP430_LaunchPad_(MSP-EXP430G2)

Getting started

All the I/O functions used here are calls to an "include" file MSPEZIO.h, and can be used with the normal c-program commands used with the msp430x2xx. Many of the calls to MSPEZIO are set to the default mode or modes for a given application. This can be change by changing the register calls in the given application and to this end it is important that the student uses the **MSP430x2xx Family User's Guide:** http://focus.ti.com/lit/ug/slau144h/slau144h.pdf to see how changes to registers can be made for more advanced applications. The following section gives references to the Function calls and example statements, MSPEZIO.h Include File, and Example programs. The final section includes a short c program, for use on a Windows PC, which can be used as Terminal to receive serial data transmitted by some of the program examples. For programs that require both TX and RX any terminal program should work provided it is set-up for a Baud rate of 9600 with 8 data bits, no parity and 1 stop bit. Tera Term version 4.68: http://ttssh2.sourceforge.jp/index.html.en is one example of a terminal program which can be used.

Functions included in MSPEZIO.h

Digital I/O

- DigitalRead()
- DigitalWrite()

- Toggle()
- Time
- Delay()
- Delayus()

Reset/Timer Reset

- Reset()
- StopWDTimer()
- SetWDTimer()
- StartWDTimer()

Serial I/O

- SerialPrintChar()
- SerialPrint()
- SerialWaitChar()

Analog I/O

- SetADC10()
- StartADC10Conversion()
- SetSD16()
- StartSD16Conversion()
- StartPWM()
- Temp()

Functions calls, Syntax, and Example statements

Digital I/O

DigitalWrite()

Syntax - DigitalWrite(Pin,Value);
Parameters - Pin = IC Pin# or BIT#
Value = High or 1
Value = Low or 0
Return - None
Example statement 1 - DigitalWrite(Pin2,High); or DigitalWrite(BIT0, 1);
This will turn on, High, IC Pin #2 which is the same as BIT0
Example statement 2 - DigitalWrite(BIT0 + BIT6, High);
This will turn on BIT0 and BIT6. BIT0 and BIT6 are connected to LED's on the Launchpad Board

Digital Read()

Syntax – int val = DigitalRead(Pin);
Parameters – Pin = BIT# or IC Pin#
Return - val = 0 or 1
Example statement - int val=DigitalRead(BIT0); This will return with val equal to 1 or 0, the state of BIT0

Toggle()

Syntax – Toggle(Pin);
Parameters _ Pin= BIT# or IC Pin#
Return – Toggle output state of Pin
Example statement – Toggle(BIT0); This will toggle BIT0 or p1.0

Time

Delay()

Syntax – Delay(int number);
Parameters – number of milliseconds
Return - None
Example statement – Delay(1000); Processor delay of 1000 milliseconds.

Delayus()

Syntax – Delay(int number);
Parameters – number of microseconds
Return - None

Example statement – Delayus(100); Processor delay of 100 microseconds.

Reset/Timer Reset

Reset()

Syntax – Reset();
Parameters – None
Return - None
Example statement –Reset(); Software reset to beginning of program.

StopWDTimer()

Syntax – StopWDTimer();
Parameters – None
Return - None
Example statement – StopWDTimer(); Stops the WDTimer ***NOTE: This should be set at the beginning of all programs that do not use the Watch Dog Timer***

SetWDTimer()

Syntax – SetWDTimer(value);
Parameters – value   The value is from the tabulation below. For the 32KHz clock the 32 KHz Xtal must be attached.

WDT-interval times [1ms] coded with Bits 0-2	
WDT is clocked by fSMCLK (assumed 1MHz)	
WDT_MDLY_32	32ms interval (default)
WDT_MDLY_8	8ms
WDT_MDLY_0_5	0.5ms
WDT_MDLY_0_064	0.064ms
WDT is clocked by fACLK (assumed 32KHz)	
WDT_ADLY_1000	1000ms
WDT_ADLY_250	250ms
WDT_ADLY_16	16ms
WDT_ADLY_1_9	1.9ms
Watchdog mode -> reset after expired time	
WDT is clocked by fSMCLK (assumed 1MHz)	
WDT_MRST_32	32ms interval (default)
WDT_MRST_8	8ms
WDT_MRST_0_5	0.5ms
WDT_MRST_0_064	0.064ms
WDT is clocked by fACLK (assumed 32KHz)	

WDT_ARST_1000	1000ms
WDT_ARST_250	250ms
WDT_ARST_16	16ms
WDT_ARST_1_9	1.9ms

Return – None

Example statement – SetWDTimer(WDT_ARST_1000); Set the Watch Dog Timer for 1000 ms, or 1 second.

StartWDTimer()

Syntax – StartWDTimer(); Starts the Watch Dog Timer with the time set by the command SetWDTimer(value). This will reset the program after the timer value.
Parameters – None
Return – None
Example statement – Start WDTimer();

Serial I/O

SerialPrintChar()

Syntax – SerialPrintChar('H'); Sends the character 'H' out the serial port.
Parameters – Single character to be sent, TXD through, serial port.
Return – None
Example statement – SerialPrintChar('a'); TXD the character a

SerialPrint()

Syntax – SerialPrint(int value);
Parameters – int value, any integer value up to 65535
Return – None
Example statement – SerialPrint(12345); TXD the number 12345

SerialWaitChar()

Syntax – SerialWaitChar();
Parameters – A wait state until any key strike from a serial keyboard RXD by the device
Return – Starts program from a wait state
Example statement – SerialWaitChar();

Analog I/O

Set ADC10()

Syntax – SetADC10(INCH1); Set-up the ADC10 ADC for a conversion.

Parameters – INCH_# Where # is the input channel number . Note: INCH_4 is p1.4 or BIT4 on
LaunchPad or msp430f2012 IC pin # 6. NOTE: INCH_10 or INCH_0x0A is the
temperature sensor.
The default reference voltage is used in MSPEZIO.h with VRef+= Vcc, ~ 3.3V and
VRef-= Vss.
Return – None
Example statement – SetADC10(INCH_3); Attach the ADC10 to Input channel 3 or p1.3

StartADC10Conversion()

Syntax – StartADC10Conversion(); Convert the analog voltage at INCH_# set in
SetADC10(INCH_#);
Parameters – None
Return - The ADC10 conversion result(0-1023) is stored in register ADC10MEM;
Example statement – StartADC10Conversion();
int ADCValue= ADC10MEM;

SetSD16()

Syntax – SetSD16(INCH_# , GAIN_# , Polarity);
Parameters - INCH_#, Where # is the input channel number . This has been set-up in this file

for single ended inputs where INCH_0 is P1.0, INCH_1 is P1.2, INCH_2 is P1.4,
INCH_3 is P1.6 and INCH_6 is the temperature sensor.

GAIN_#, Where # represents amplifier gains of 1, 2, 4, 8, 16, 32.

Polarity, Set as UNIPOLAR (output from 0 to 65535) or as BIPOLAR (output from
0 to 65535 with 0 volts at 32767. The default reference voltage is used in
MSPEZIO.h with VRef=1.2 volts.
Return – None
Example statement – SetSD16(INCH_2 , GAIN_1 , UNIPOLAR);

StartSD16Conversion()

Syntax - StartSD16Conversion(); Convert the analog voltage at INCH# , GAIN, and Polarity set by SetSD16().
Parameters - None
Return - The SD16 conversion result is (0-65535) is stored in register SD16MEM0.
Example statement – StartSD16Conversion();
int ADCValue=SD16MEM0;

StartPWM()

Syntax – StartPWM(Period, Cycle, Pout);
Parameters - Period, PWM period.
Cycle, Duty cycle on out pin.
Pout, Output pin number or BIT#.
Example statement – StartPWM(1023,ADCValue,BIT6); or
StartPWM(1023,ADCValue,0X40)

Temp()

Syntax – Temp();
Parameters – None
Return – The **int** Temperature Value is saved as DegF or DegC.
Example statement – Temp();
SerialPrint(DegF);

MSPEZIO.h Include file

Copy, paste and save this include file as MSPEZIO.h in your IDE along with your program.

```
////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////
// MSPEZIO.h
// Some EZ I/O commands for the MSP430f2012, MSP430f2013, MSP430G2231
// E. J. Seykora
// Physics Department East Carolina University
// Feb. 2011
////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////
void StopWDTimer()
{
 WDTCTL = WDTPW + WDTHOLD;       // Stop WDT
}
////////////////////////////////////////////////////////////////////////////////////
void Reset()
{
    WDTCTL=0;
}
////////////////////////////////////////////////////////////////////////////////////
void SetWDTimer(int Time)
{
  WDTCTL = Time;
}
////////////////////////////////////////////////////////////////////////////////////
void StartWDTimer()
{
  IE1 |= WDTIE; //enable interrupt
 _BIS_SR(LPM0_bits + GIE);      // Enter LPM3 w/interrupt
}
////////////////////////////////////////////////////////////////////////////////////
#define High 0x01
#define Low  0x00
////////////////////////////////////////////////////////////////////////////////////
int DigitalRead (int Pin)
{
 P1DIR |= 0x01;                // Set P1.0 to output direction
 Pin & P1IN;
 return (Pin & P1IN);
```

```
}
////////////////////////////////////////////////////////////////////////////////
void DigitalWrite(int Pin,int Value)
{
       P1DIR=Pin;
    if  (Value==High){
       P1OUT=Pin;}
    if  (Value==Low){
       P1OUT &=~Pin;}
}
////////////////////////////////////////////////////////////////////////////////
void Toggle(int Pin)
{
    P1DIR |=Pin;
    P1OUT ^= Pin;
}
///////////////////////////////////////////////////////////////////////////////
void SerialWaitChar()
{
while ((BIT2 & P1IN)!=0){}   //Wait for RX-BIT2 to be High from Serial in
}
///////////////////////////////////////////////////////////////////////////////

 // Function Transmits Character from TXByte

#define Bitime 104 //9600 Baud, SMCLK=1MHz (1MHz/9600)=104
#define TXD BIT1 // TXD on P1.1
#define RXD BIT2 // RXD on P1.2
unsigned char BitCnt; // Bit count, used when transmitting byte
int TXByte; // Value sent when Transmit() is called
void transmit()
{
  BCSCTL1 = CALBC1_1MHZ; // Set range
  DCOCTL = CALDCO_1MHZ; // SMCLK = DCO = 1MHz
  TACTL = TASSEL_2 + MC_2; // SMCLK, continuous mode
  TXByte |= 0x100; // Add stop bit to TXByte
  TXByte = TXByte << 1; // Add start bit  0
  BitCnt =0XA;//Load Bit counter, 8 data + ST/SP
  CCTL0 = OUT; // TXD Idle as Mark
  CCR0 = TAR;
  CCR0 += Bitime; // Time to first bit
  CCTL0 = CCIS0 + OUTMOD0 + CCIE; // Set signal, intial value, enable interrupts
  while ( CCTL0 & CCIE ); // Wait for TX completion
 // TACTL = TASSEL_1; // SMCLK, timer off (for power consumption)
```

```
}

// Timer A0 interrupt service routine
#pragma vector=TIMERA0_VECTOR
__interrupt void Timer_A (void)
{
  CCR0 += Bitime; // Add Offset to CCR0
  if ( BitCnt == 0)
  {
    CCTL0 &= ~ CCIE ;  //All bits TXed, disable interrupt
  }
  else
  {
    CCTL0 |= OUTMOD2; // TX Space
    if (TXByte & 0x01)
    CCTL0 &= ~ OUTMOD2; // TX Mark
    TXByte = TXByte >> 1;
    BitCnt --;
  }
}
////////////////////////////////////////////////////////////////////////////////
int val,v;
void SerialPrintChar (int val)
{
  P1SEL |= TXD + RXD;
  P1DIR |= TXD;
  v=val;
  TXByte=v;
  transmit();
}
////////////////////////////////////////////////////////////////////////////////
int i;
unsigned int a,b,c,d,e,f;
//unsigned int ADC;
void SerialPrint(unsigned int ADC)
{
 P1SEL |= TXD + RXD;
 P1DIR |= TXD;
 a=(ADC & 0xFF00)/10000;
 b=(ADC-a*10000)/1000;///0
 c=(ADC-a*10000-b*1000)/100;
 d=(ADC-a*10000-b*1000-c*100)/10;
 e=ADC-a*10000-b*1000-c*100-d*10;
 if (a==0) TXByte=0x20;
 else TXByte=a+0x30;
 transmit();
 if (a+b==0) TXByte=0x20;
 else TXByte=b+0x30;
 transmit();
 if (a+b+c==0) TXByte=0x20;
 else TXByte=c+0x30;
 transmit();
 if(a+b+c+d==0) TXByte=0x20;
 else TXByte=d+0x30;
 transmit();
 TXByte=e+0x30;
 transmit();
 TXByte=0x0D;
 transmit();
 TXByte=0x0A;
 transmit();
}
////////////////////////////////////////////////////////////////////////////////
#ifdef __msp430x20x3
#define INCH_0   SD16INCH_0
#define INCH_1   SD16INCH_1
#define INCH_2   SD16INCH_2
#define INCH_3   SD16INCH_3
#define INCH_6   SD16INCH_6
#define GAIN_1   SD16GAIN_1
#define GAIN_2   SD16GAIN_2
#define GAIN_4   SD16GAIN_4
#define GAIN_8   SD16GAIN_8
#define GAIN_16  SD16GAIN_16
#define GAIN_32  SD16GAIN_32
#define UNIPOLAR SD16UNI
#define BIPOLAR  0
// Pin1 is Vcc, Pin14 is Ground, Pin3 & Pin4 TXD RXD
#define Pin2   BIT0
#define Pin3   BIT1
#define Pin4   BIT2
#define Pin5   BIT3
#define Pin6   BIT4
#define Pin7   BIT5
void SetSD16(int INCH,int GAIN,int POLAR)
{
 SD16CTL = SD16REFON + SD16SSEL_1; // 1.2V ref, SMCLK
 SD16INCTL0 = GAIN+INCH;
 SD16CCTL0 =  POLAR;
  __bis_SR_register(GIE);   // interrupts enabled
}
#endif
////////////////////////////////////////////////////////////////////////////////
#ifdef __msp430x20x2
void SetADC10(long int INCH)
{
  ADC10CTL0 =ADC10SHT_0 + ADC10ON;
  if (INCH==INCH_10) ADC10CTL0 = SREF_1 + ADC10SHT_3 + REFON + ADC10ON;
  ADC10CTL1=CONSEQ_0 + INCH;
  ADC10AE0 |=INCH;
  __bis_SR_register(GIE);  // interrupts enabled
}
#endif
///////////////////////////////////////////////////////////////////////////////
#ifdef __MSP430G2231
void SetADC10(long int INCH)
{
  ADC10CTL0 =ADC10SHT_0  + ADC10ON;
  if (INCH==INCH_10) ADC10CTL0 = SREF_1 + ADC10SHT_3 + REFON + ADC10ON;
  ADC10CTL1=CONSEQ_0 + INCH;
  ADC10AE0 |=INCH;
  __bis_SR_register(GIE);  // interrupts enabled
}
#endif
///////////////////////////////////////////////////////////////////////////////
void StartPWM(int Period,int Cycle,int Pout)
{
  WDTCTL = WDTPW + WDTHOLD;       // Stop WDT
  P1DIR |= Pout;                  // P1.2 and P1.3 output
  P1SEL |= Pout;                  // P1.2 and P1.3 TA1/2 options
  CCR0 = Period;                  // PWM Period/2
  CCTL1 = OUTMOD_2;               // CCR1 toggle/set
  CCR1 = Cycle;                   // CCR1 PWM duty cycle
  TACTL = TASSEL_2 + MC_3;        // SMCLK-TSSEL_2,ACLK-TASSEL_1,
                              //  MC_3,up-down mode up to TACCR0 and down
                              //  MC_2,Continuous count up to 0XFFFF
                              //  MC_1,up mode count up to TACCR0
  //_BIS_SR(LPM0_bits+GIE);       // Enter LPM0
}
/////////////////////////////////////////////////////////////////////////////////
#ifdef __msp430x20x3
void StartSD16Conversion()
```

```
{
  SD16CCTL0 |= SD16SC;
  while(!(SD16CCTL0 & SD16IFG));
}
#endif
////////////////////////////////////////////////////////////////////////////////
#ifdef __msp430x20x2
void StartADC10Conversion()
{
  ADC10CTL0 &= ~ADC10SC;
  ADC10CTL0 &= ~ENC;
  ADC10CTL0 &= ~ADC10IFG;
  ADC10CTL0 |=ENC + ADC10SC;
  while(!(ADC10IFG & ADC10CTL0));
}
#endif
////////////////////////////////////////////////////////////////////////////////
#ifdef __MSP430G2231
void StartADC10Conversion()
{
  ADC10CTL0 &= ~ADC10SC;
  ADC10CTL0 &= ~ENC;
  ADC10CTL0 &= ~ADC10IFG;
  ADC10CTL0 |=ENC + ADC10SC;
  while(!(ADC10IFG & ADC10CTL0));
}
#endif
/////////////////////////////////////////////////////////////////////////////////
void Delay(unsigned int ms)
{
  unsigned int i;
  for (i=0;i<ms;i++)
    __delay_cycles(1000);
}
/////////////////////////////////////////////////////////////////////////////////
void Delayus(unsigned int us)
{
  unsigned int i;
  for (i=0;i<us;i++)
    __delay_cycles(1);
}
/////////////////////////////////////////////////////////////////////////////////
#ifdef __msp430x20x2
long int DegF,DegC;
long temp;
void Temp()
{
SetADC10(INCH_10);
StartADC10Conversion();
temp=ADC10MEM;
DegF = ((temp - 630) * 761) / 1024;
DegC = ((temp - 673) * 423) / 1024;
}
#endif
////////////////////////////////////////////////////////////////////////////////
 #ifdef  __msp430x20x3
long int DegF,DegC;
void Temp()
{
SetSD16(INCH_6,GAIN_1,BIPOLAR);
StartSD16Conversion();
DegF=SD16MEM0;
DegF=DegF*(24969)/1000000-1275;
DegC=(DegF-32)*5/9;
}
#endif
////////////////////////////////////////////////////////////////////////////////
#ifdef __MSP430G2231
long int DegF,DegC;
long temp;
void Temp()
{
SetADC10(INCH_10);
StartADC10Conversion();
temp=ADC10MEM;
DegF = ((temp - 630) * 761) / 1024;
DegC = ((temp - 673) * 423) / 1024;
}
#endif
////////////////////////////////////////////////////////////////////////////////
```

Example programs using msp430f2012, msp430f2013, and msp430g2231

```
/////////////////////////////////////////////////////////////////////////////////
// Example 1.  ADC conversion(using ADC10 on msp430f2012 or msp430g2231) on
// INCH_5 (BIT5) and output result sent to TXT p1.1 on Launchpad board.
// Baud rate - 9600, Data bits - 8, Parity - None, Stop Bits 1.
// Program is in an endless "for" loop. View output with a terminal program.
/////////////////////////////////////////////////////////////////////////////////
#include "msp430f2012.h"  //or #include "msp430g2231.h"
#include "MSPEZIO.h"
int ADCValue;
void main(void)
{
StopWDTimer();
for(;;)
{
SetADC10(INCH_5);
StartADC10Conversion();
ADCValue=ADC10MEM;
SerialPrint(ADCValue);
}
}
```

```
/////////////////////////////////////////////////////////////////////////////////
// Example 2.  ADC conversion(using ADC10 on msp430f2012 or msp430g2231) on
// INCH_5 (BIT5) and output result sent to TXT p1.1 on Launchpad board.
// Baud rate - 9600, Data bits - 8, Parity - None, Stop Bits 1.
// Program is in an endless loop by the Reset() as the last statement. View
// output with a terminal program.
/////////////////////////////////////////////////////////////////////////////////
#include "msp430f2012.h"  //or #include "msp430g2231.h"
#include "MSPEZIO.h"
int ADCValue;
void main(void)
{
StopWDTimer();
SetADC10(INCH_5);
StartADC10Conversion();
ADCValue=ADC10MEM;
SerialPrint(ADCValue);
Reset();
}
```

```
////////////////////////////////////////////////////////////////////////////////
// Example 3.  ADC conversion(using ADC10 on msp430f2012 or msp430g2231) on
// INCH_5 (BIT5) and output result to TXT p1.1 on Launchpad board.
// Baud rate - 9600, Data bits - 8, Parity - None, Stop Bits 1.
// Program is reset after ~1 second by WDT. View output with a terminal program.
////////////////////////////////////////////////////////////////////////////////
#include "msp430f2012.h"  //or #include "msp430g2231.h"
#include "MSPEZIO.h"
unsigned int ADCValue;
void main(void)
{
SetWDTimer(WDT_ADLY_1000);
SetADC10(INCH_5);
StartADC10Conversion();
ADCValue=ADC10MEM;
SerialPrint(ADCValue);
StartWDTimer();
Reset();
}
```

```
////////////////////////////////////////////////////////////////////////////////
//  Example 4.  ADC conversion(using ADC10 on msp430f2012 or msp430g2231) on
//  INCH_4 (BIT4) (p1.4) and output result to TXT p1.1 on Launchpad board.
//  Baud rate - 9600, Data bits - 8, Parity - None Stop Bits 1. In an endless
//  "for" loop with a delay of 1.0 sec between readings. Set LED's BIT0 and
//  BIT6 in Bunary code during delay, View output with a terminal program.
////////////////////////////////////////////////////////////////////////////////
#include "msp430g2231.h"  //or #include "msp430f2012.h"
#include "MSPEZIO.h"
int ADCValue;
void main(void)
{
StopWDTimer();
  for(;;)
{
SetADC10(INCH_4);
StartADC10Conversion();
ADCValue=ADC10MEM;
SerialPrint(ADCValue);
DigitalWrite(BIT0+BIT6,Low);
Delay(250);
DigitalWrite(BIT0,High);
Delay(250);
DigitalWrite(BIT0,Low);
DigitalWrite(BIT6,High);
Delay(250);
DigitalWrite(BIT0+BIT6,High);
Delay(250);
}
}

////////////////////////////////////////////////////////////////////////////////
// Example 5.  ADC conversion(using SD16 on msp430f2013) for INCH_0,GAIN_1,
// UNIPOLAR mode with result sent to TXT p1.1 on Launchpad board. Baud rate
// 9600, Data bits - 8, Parity - None, Stop Bits 1. Program is in an endless
// "for" loop. View output with a terminal program.
// Note: MSPEZIO.h sets SD16 to single ended inputs with:
// INCH_0 on BIT0 or p1.0
// INCH_1 on BIT2 or p1.2
// INCH_2 on BIT4 or p1.4
// INCH_3 on Bit6 or p1.6
// INCH_6 Temperature sensor
// Also note that p1.1 and p1.2 are set set for TXT and RXD for USB virtual com
// port on Launchpad board.
////////////////////////////////////////////////////////////////////////////////
#include "msp430f2013.h"
#include "MSPEZIO.h"
int ADCValue;
void main(void)
{
 StopWDTimer();
 SetSD16(INCH_0,GAIN_1,UNIPOLAR);
 for(;;)
 {
 StartSD16Conversion();
 ADCValue=SD16MEM0;
 SerialPrint(ADCValue);
 }
}

////////////////////////////////////////////////////////////////////////////////
// Example 6.  ADC conversion(using SD16 on msp430f2013) for INCH_0,GAIN_1,
// UNIPOLAR mode with result sent to TXT p1.1 on Launchpad board. Baud rate
// 9600, Data bits - 8, Parity - None, Stop Bits 1. Program is in an endless
// "for" loop with a delay of 0.5 sec.  Read (BIT6) or P1.6 and transmit
// state high 'H' or low 'L'. Connect a wire from BIT6 to ground or Vcc to
// change state.
// View output with a terminal program.
// Note: MSPEZIO.h sets SD16 to single ended inputs with:
// INCH_0 on BIT0 or p1.0
// INCH_1 on BIT2 or p1.2
// INCH_2 on BIT4 or p1.4
// INCH_3 on Bit6 or p1.6
// INCH_6 Temperature sensor
// Also note that p1.1 and p1.2 are set set for TXT and RXD for USB virtual com
// port on Launchpad board.
///////////////////////////////////////////////////////////////////////////
#include "msp430f2013.h"
#include "MSPEZIO.h"
int ADCValue;
 int k;
void main(void)
{
  StopWDTimer();
  SetSD16(INCH_2,GAIN_1,UNIPOLAR);
  for(;;)
  {
  StartSD16Conversion();
  ADCValue=SD16MEM0;
  k=DigitalRead(BIT6);
  if(k==0){
    SerialPrintChar('L');
  }
  else
    SerialPrintChar('H');
  SerialPrintChar(0xA);     //print line feed
  SerialPrintChar(0xD);     //print carriage return
  SerialPrint(ADCValue);
  Delay(500);
  }
}

///////////////////////////////////////////////////////////////////////////
// Example 7.  ADC conversion(using SD16 on msp430f2013) for INCH_0,GAIN_1,
// BIPOLAR mode with result sent to TXT p1.1 on Launchpad board. Baud rate
// 9600, Data bits - 8, Parity - None, Stop Bits 1. Program is in an endless
// "for" loop. The program is in a wait state for any char to be sent. After
// a char is sent from keyboard the ADC value is transmitted.  Use a terminal
// program to sent each char and display each output.
// Note: MSPEZIO.h sets SD16 to single ended inputs with:
// INCH_0 on BIT0 or p1.0
// INCH_1 on BIT2 or p1.2
// INCH_2 on BIT4 or p1.4
// INCH_3 on Bit6 or p1.6
// INCH_6 Temperature sensor
// Also note that p1.1 and p1.2 are set set for TXT and RXD for USB virtual com
// port on Launchpad board.
////////////////////////////////////////////////////////////////////////////////

#include "msp430f2013.h"
#include "MSPEZIO.h"
int ADCValue;
void main(void)
{
 StopWDTimer();
 SetSD16(INCH_0,GAIN_1,BIPOLAR);
 for(;;)
 {
 SerialWaitChar();
 StartSD16Conversion();
 ADCValue=SD16MEM0;
 SerialPrint(ADCValue);
 }
}

////////////////////////////////////////////////////////////////////////////////
// Example 8.  ADC conversion(using ADC10 on msp430f2012 or msp430g2231) on
// INCH_5 (BIT5). The ADCValue is Pulse Width Modulated and sent to BIT6
// "LED" output on Launchpad board for display.  Program is in an endless
// "for" loop.
////////////////////////////////////////////////////////////////////////////////
#include "msp430f2012.h"  //or #include "msp430g2231.h"
#include "MSPEZIO.h"
int ADCValue;
void main(void)
{
StopWDTimer();
for(;;)
{
SetADC10(INCH_5);
StartADC10Conversion();
ADCValue=ADC10MEM;
StartPWM(1023/2,ADC10MEM/2,BIT6);
}
}
```

Serial terminal program

```
////////////////////////////////////////////////////////////////////////////////
// SerialTerm.c is a program used to receive(only) data transmitted on a com or
// virtual com port.  The program calls TXRX.h where the baud rate and com
// number must be set. SerialTerm.c and TXRX.h use standard win32 API calls.
// The received data, text file, is saved at c:\\chan1.dat and may be plotted
// using gnuplot or wgnuplot.
// SerialTerm.c + TXRX.h may be compiled using Dev-C++ from
// http://www.bloodshed.net
////////////////////////////////////////////////////////////////////////////////

#include "TXRX.h"
main()
{
     TXRX();
FILE*fp;
fp=fopen("c:\\chan1.dat","w");
 for(;;)
{
          ReadFile(hCom, &c1, 1,  &iBytesRead,NULL);
          printf("%c",c1);
          fprintf(fp,"%c",c1);
          c1=0;
}

 CloseHandle(hCom);
 hCom=0;
 fclose(fp);
}



////////////////////////////////////////////////////////////////////////////////
//  TXRX.h header file used to open and close a virtual com (serial) port.
//  Com# must be set for com port used.
//  BaudRate must be set for baud rate used.
//  TXRX.h use standard win32 API calls
////////////////////////////////////////////////////////////////////////////////
#include <windows.h>
#include <stdio.h>
int iBytesRead;
```

```
char a;
int c1;
int iBytesRead=0;
HANDLE hCom;
void TXRX(void);
void TXRX(void)
{
  DCB dcb;
  char *pcCommPort = "COM8";// Change to COM# used by your device
  hCom = CreateFile( pcCommPort,
                GENERIC_READ | GENERIC_WRITE,
                0,    // must be opened with exclusive-access
                NULL, // no security attributes
                OPEN_EXISTING, // must use OPEN_EXISTING
                0,    // not overlapped I/O
                NULL  // hTemplate must be NULL for comm devices
                );

  GetCommState(hCom, &dcb);
  // Fill in DCB: 9,600 bps, 8 data bits, no parity, and 1 stop bit.
  dcb.BaudRate = CBR_9600;     // set the baud rate
  dcb.ByteSize = 8;            // data size, xmit, and rcv
  dcb.Parity = NOPARITY;       // no parity bit
  dcb.StopBits = ONESTOPBIT;    // one stop bit
  SetCommState(hCom, &dcb);
}
```