Assignment 8

Kyle Rosenthal 5/12/17 S17

Extra Credit done.

# Youtube Link

<https://www.youtube.com/watch?v=qDJF_OJ52iA>

# Code

## main.c

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

// MSP432 main.c template - Empty main

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**#include** "msp.h"

**#include** "adc.h"

**#include** "uart.h"

**void** **main**(**void**)

{

WDTCTL = WDTPW | WDTHOLD; // Stop watchdog timer

\_\_enable\_irq();

Setup\_UART();

Setup\_ADC(33,0);

**char** arr[8];

**int** i;

**while** (1) {

ADC\_RequestNextSample();

**while** (!ADC\_CheckReady()) {}

ADC\_GetFormatedValue(arr);

**for** (i = 0; i < 6; i++) {

UART0Tx(arr[i]);

}

UART0Tx(' ');

UART0Tx('V');

UART0Tx(0x1b);

UART0Tx('[');

UART0Tx('H');

**for** (i = 0; i < 20000; i++) {}

}

}

## uart.h

/\*

\* uart.h

\*

\* Created on: May 8, 2017

\* Author: kmrosent

\*/

**#ifndef** UART\_H\_

**#define** UART\_H\_

**int** statusFlag;

**int** val;

**void** **Setup\_UART**();

**int** **readFlag**();

**void** **setFlag**();

**void** **clearFlag**();

**int** **readVal**();

**unsigned** **char** **UART0Rx**(**void**);

**unsigned** **char** **UART0Tx**(**unsigned** **char** c);

**void** **EUSCIA0\_IRQHandler**(**void**);

**#endif** /\* UART\_H\_ \*/

## uart.c

/\*

\* uart.c

\*

\* Created on: May 8, 2017

\* Author: kmrosent

\*/

**#include** "uart.h"

**#include** "msp.h"

**void** **Setup\_UART**() {

val = 0;

statusFlag = 0;

\_\_disable\_irq();

EUSCI\_A0->CTLW0 |= BIT0;

EUSCI\_A0->MCTLW = 0;

EUSCI\_A0->CTLW0 = 0x0081;

EUSCI\_A0->BRW = 26;

P1SEL0 |= (BIT2 + BIT3);

P1SEL1 &= ~(BIT2 + BIT3);

EUSCI\_A0->CTLW0 &= ~BIT0;

EUSCI\_A0->IFG |= EUSCI\_A\_IFG\_RXIFG;

EUSCI\_A0->IE |= EUSCI\_A\_IE\_RXIE;

//NVIC\_SetPriority(EUSCIA0\_IRQn, 4);

NVIC\_EnableIRQ(*EUSCIA0\_IRQn*);

\_\_enable\_irq();

}

**int** **readFlag**() {

**return** statusFlag;

}

**void** **setFlag**() {

statusFlag = 1;

}

**void** **clearFlag**() {

statusFlag = 0;

}

**int** **readVal**() {

**int** temp = val;

val = -1;

**return** temp;

}

/\* read a character from UART0 \*/

**unsigned** **char** **UART0Rx**(**void**) {

**char** c;

**while**(!(EUSCI\_A0->IFG & 0x01)) ;

c = EUSCI\_A0->RXBUF;

**return** c;

}

/\* write a character to UART \*/

**unsigned** **char** **UART0Tx**(**unsigned** **char** c) {

**while**(!(EUSCI\_A0->IFG&0x02)) ;

EUSCI\_A0->TXBUF = c;

**return** c;

}

**void** **EUSCIA0\_IRQHandler**(**void**) {

**char** c = EUSCI\_A0->RXBUF;

**if** (c == '\r') {

UART0Tx(c);

c = '\n';

}

**while**(!(EUSCI\_A0->IFG & 0x02)) {}

EUSCI\_A0->TXBUF = c;

}

## adc.h

/\*

\* adc.h

\*

\* Created on: May 10, 2017

\* Author: kmrosent

\*/

**#ifndef** ADC\_H\_

**#define** ADC\_H\_

**#define** F\_ADC\_READ\_ME 0

**#define** F\_ADC\_REQUEST 1

**#define** F\_ADC\_NO\_OP 2

**#define** CAL 79 / 428 + 30

**int** vL, vH, adcflag;

**unsigned** **long** lastRead;

**void** **Setup\_ADC**(**int** v\_h, **int** v\_l);

**void** **ADC\_RequestNextSample**();

**void** **ADC14\_IRQHandler**();

**int** **ADC\_CheckReady**();

**unsigned** **int** **ADC\_GetRawValue**();

**void** **ADC\_GetFormatedValue**(**char**\* value);

**#endif** /\* ADC\_H\_ \*/

## adc.c

/\*

\* adc.c

\*

\* Created on: May 10, 2017

\* Author: kmrosent

\*/

**#include** "adc.h"

**#include** "msp.h"

**void** **Setup\_ADC**(**int** v\_h, **int** v\_l) {

vL = v\_l;

vH = v\_h;

adcflag = F\_ADC\_NO\_OP;

P5->SEL0 |= BIT5;

P5->SEL1 |= BIT5;

//sample speed, sample and hold = 16, on

ADC14->CTL0 = ADC14\_CTL0\_SHT0\_2 | ADC14\_CTL0\_SHP | ADC14\_CTL0\_ON;

//sample res = 14 bit

ADC14->CTL1 = ADC14\_CTL1\_RES\_3;

ADC14->MCTL[0] |= ADC14\_MCTLN\_INCH\_0;

// int enable

ADC14->IER0 |= ADC14\_IER0\_IE0;

NVIC->ISER[0] = 1 << ((*ADC14\_IRQn*) & 31);

//wake on isr exit

SCB->SCR &= ~SCB\_SCR\_SLEEPONEXIT\_Msk;

}

**void** **ADC\_RequestNextSample**() {

//start sample

**if** (adcflag != F\_ADC\_REQUEST) {

ADC14->CTL0 |= ADC14\_CTL0\_ENC | ADC14\_CTL0\_SC;

}

adcflag = F\_ADC\_REQUEST;

}

**void** **ADC14\_IRQHandler**() {

lastRead = ADC14->MEM[0]; //output

adcflag =F\_ADC\_READ\_ME;

}

**int** **ADC\_CheckReady**() {

**return** adcflag == F\_ADC\_READ\_ME;

}

**unsigned** **int** **ADC\_GetRawValue**() {

adcflag = F\_ADC\_NO\_OP;

**return** lastRead;

}

**void** **ADC\_GetFormatedValue**(**char**\* value) {

adcflag = F\_ADC\_NO\_OP;

**unsigned** **long** **long** conversion = lastRead \* CAL;

**int** loc = 5;

value[2] = '.';

**while** (loc >= 0) {

value[loc] = '0' + (conversion % 10);

conversion /= 10;

**if** (loc == 3) loc--;

loc--;

}

}