**Appendix B – Code for Part 1**

// Lab 4 Part 1.c

// Chris Pybus and Jeff Pistacchio

//

//This program polls an Analog input and outputs the voltage in hex and dec

//-------------------------------------------------------------------------------------------

// Includes

//-------------------------------------------------------------------------------------------

#include <c8051f120.h> // SFR declarations.

#include <stdio.h> // Necessary for printf.

#include "putget.h" // Necessary for printf

//-------------------------------------------------------------------------------------------

// Global Variables

//-------------------------------------------------------------------------------------------

unsigned int count **=** 0**;** //used to count the number of MAC iterations

//for the average value calculation

\_\_sbit \_\_at **(**0x90**)** PB**;**

unsigned char n **=** 0**;** //for for loop in calc average

//-------------------------------------------------------------------------------------------

// MAIN Routine

//-------------------------------------------------------------------------------------------

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

**{**

unsigned short int volt**;**

float print**;**

float voltL **=** 2.5**;** //store low voltage value

float voltH **=** 0**;** //store high voltage value

unsigned long int avgV **=** 0**;** //store average voltage running sum

unsigned short int value**;**

unsigned char sum1**;**

unsigned char sum2**;**

unsigned char sum3**;**

unsigned char sum4**;**

WDTCN **=** 0xDE**;** // Disable watchdog timer.

WDTCN **=** 0xAD**;**

SFRPAGE **=** CONFIG\_PAGE**;**

SYSCLK\_Init**();** // Initialize the oscillator.

TIMER\_Init**();** // Initialize timers

UART\_Init**();** // Initialize UARTs.

PORT\_Init**();** // Configure the Crossbar and GPIO.

ADC\_Init**();**

printf**(**"\033[2J"**);** //Erase screen and move cursor to the home position.

printf**(**"Press Push Button\r\n"**);**

**while(**PB**);**

printf**(**"\033[2J"**);**

**while(**1**)**

**{**

//printf("\033[2J");

//Read Analog Input

volt **=** read\_AD\_input**();**

value **=** volt**;**

//Check for low Value

**if** **(**volt **<** voltL**)**

**{**

voltL **=** volt**;**

**}**

//Check for high value

**if** **(**volt **>** voltH**)**

**{**

voltH **=** volt**;**

**}**

//MAC\_\_\_\_\_\_\_\_\_\_\_\_\_

SFRPAGE **=** MAC0\_PAGE**;**

//Clear Accumulator

MAC0CF **=** 0x08**;**

//set Mac A value

MAC0AH **=** 0x00**;**

MAC0AL **=** 0x01**;**

//Set Mac B value

value **=** volt**>>**8**;**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//Set Mac B value

value **=** **(**volt**>>**8**);**

MAC0BH **=** value**;**

value **=** **(**char**)**volt**%**16**;**

MAC0BL **=** value**;**

SFRPAGE **=** MAC0\_PAGE**;**

SFRPAGE **=** MAC0\_PAGE**;**

//Multiply, Accumulate

//put accumulator registors onto a long int

avgV **=** MAC0ACC3**;**

avgV **=** **(**avgV**<<**8**)+** MAC0ACC2**;**

avgV **=** **(**avgV**<<**8**)+** MAC0ACC1**;**

avgV **=** **(**avgV**<<**8**)+** MAC0ACC0**;**

sum1 **=** MAC0ACC0**;** sum2 **=** MAC0ACC1**;** sum3 **=** MAC0ACC2**;** sum4 **=** MAC0ACC3**;**

SFRPAGE **=** UART0\_PAGE**;**

//Print Digital voltage value to six decimal places

print **=** volt**;** //store ADC value on float.

printf\_fast\_f**(**" Voltage: %.6f V | "**,** **(**print **\*** 2.4**/**4096**));**

printf**(**"%x \r"**,**volt**);**

printf\_fast\_f**(**"\n Low Voltage: %.6f V | "**,** **(**voltL **\*** 2.4**/**4096**));**

printf**(**"%x \r"**,**voltL**);**

printf\_fast\_f**(**"\n High Voltage: %.6f V | "**,** **(**voltH **\*** 2.4**/**4096**));**

printf**(**"%x \r"**,**voltH**);**

print **=** avgV**;**

printf\_fast\_f**(**"\nAverage Voltage: %.6f V | "**,** **(**2**\***print**\***2.4**/(**77100**)));**

printf**(**"%llx \r"**,**avgV**/**10**);**

//printf("\n BHigh %x BLow %x \r\n Byte 4 %x \r\n Byte 3 %x \r\n Byte 2 %x \r\n Byte 1 %x \r", volt%16, volt/16, sum4, sum3, sum2,sum1);

printf**(**"\033[1;0H"**);**

**}**

**}**

//-------------------------------------------------------------------------------------------

// Timer\_Init

//-------------------------------------------------------------------------------------------

void TIMER\_Init**(**void**)**

**{**

//Timer2, used for UART0, baud 115200

SFRPAGE **=** TMR2\_PAGE**;**

TMR2CN **=** 0x04**;**

TMR2CF **=** 0x08**;**

RCAP2L **=** 0xF4**;**

RCAP2H **=** 0xFF**;**

**}**

//-------------------------------------------------------------------------------------------

// PORT\_Init

//-------------------------------------------------------------------------------------------

void PORT\_Init**()**

**{**

SFRPAGE **=** CONFIG\_PAGE**;**

XBR0 **=** 0x04**;** //eanble UART0

XBR1 **=** 0x00**;**

XBR2 **=** 0x40**;**

P0MDOUT **=** 0x05**;**

P0 **=** **~**0x05**;**

P1MDOUT **&=** 0xFE**;**

P1 **|=** 0x01**;** //PB input on p1.0

//P1MDIN = 0xFE;

**}**

//-------------------------------------------------------------------------------------------

// SYSCLK\_Init

//-------------------------------------------------------------------------------------------

void SYSCLK\_Init**(**void**)**

**{**

int i **=** 0**;**

SFRPAGE **=** CONFIG\_PAGE**;**

OSCXCN **=** 0x67**;**

**for** **(**i **=** 0**;** i **<** 3000**;** i**++);** // Wait 1ms for initialization

**while** **((**OSCXCN **&** 0x80**)** **==** 0**);**

CLKSEL **=** 0x01**;**

OSCICN **&=** **~**0x80**;**

**}**

//-------------------------------------------------------------------------------------------

// UART0\_Init

//-------------------------------------------------------------------------------------------

void UART\_Init**()**

**{**

//UART0

SFRPAGE **=** UART0\_PAGE**;**

SCON0 **=** 0x50**;**

SSTA0 **=** 0x05**;**

TI0 **=** 1**;**

**}**

void ADC\_Init**(**void**)**

**{**

SFRPAGE **=** 0**;**

REF0CN **=** 0x03**;** // Set Vref to use internal reference voltage (2.4 V)

AMX0CF **&=** 0xF0**;** // Configure A/D pins

AMX0SL **&=** 0xF0**;** // Set AIN0.0 to single ended input

ADC0CF **|=** 0x28**;** // Set A/D converter gain to 1 and define SAR clock speed 2Mhz

ADC0CN **=** 0x80**;** // Enable A/D converter (ADC2)

**}**

unsigned short int read\_AD\_input**(**void**)**

**{**

unsigned short int volt**;**

SFRPAGE **=** 0x00**;**

AD0BUSY **=** 1**;** //Start conversion

**while** **(!**AD0INT**);** //wait for ADC to finish

volt **=** **(**ADC0H **<<** 8**)** **+** ADC0L**;**

**return** volt**;**

**}**

void MAC\_Init**(**void**)**

**{**

MAC0CF **=** 0x0A**;**

**}**

**Appendix C – Code for Part 3**

// Lab 4 Part 3.c

// Chris Pybus and Jeff Pistacchio

//

//This program polls an Analog input and outputs the voltage in hex and dec

//-------------------------------------------------------------------------------------------

// Includes

//-------------------------------------------------------------------------------------------

#include <c8051f120.h> // SFR declarations.

#include <stdio.h>

#include <math.h>

#include "putget.h" // Necessary for printf

//-------------------------------------------------------------------------------------------

// Function PROTOTYPES

//-------------------------------------------------------------------------------------------

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

void TIMER\_Init**(**void**);**

void PORT\_Init**(**void**);**

void SYSCLK\_Init**(**void**);**

void UART\_Init**(**void**);**

void ADC\_Init**(**void**);**

void DAC\_Init**();**

unsigned short int read\_AD\_input**(**void**);**

void dac\_output**(**unsigned short value**);**

void dac\_output\_sinwave**(**void**);**

void dac\_output\_sawtooth**();**

//-------------------------------------------------------------------------------------------

// Global Variables

//-------------------------------------------------------------------------------------------

unsigned int count **=** 0**;** //used to count the number of MAC iteration

\_\_sbit \_\_at **(**0x90**)** PB**;**

//-------------------------------------------------------------------------------------------

// MAIN Routine

//-------------------------------------------------------------------------------------------

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

**{**

unsigned short int volt **=** 0**;**

int index **=** 0**;**

//float volt2 = 0;

WDTCN **=** 0xDE**;** // Disable watchdog timer.

WDTCN **=** 0xAD**;**

SFRPAGE **=** CONFIG\_PAGE**;**

SYSCLK\_Init**();** // Initialize the oscillator.

TIMER\_Init**();** // Initialize timers

UART\_Init**();** // Initialize UARTs.

PORT\_Init**();** // Configure the Crossbar and GPIO.

ADC\_Init**();**

DAC\_Init**();**

printf**(**"\033[2J"**);** //Erase screen and move cursor to the home position.

printf**(**"Press push button to start part 3"**);**

**while(**PB**);**

printf**(**"\rRunning Part 3 \n\n\r"**);**

**while(**1**)**

**{**

volt **=** read\_AD\_input**();**

//volt2 = volt;

//printf\_fast\_f("Voltage: %.2f V \r", (volt \* 2.4/4096));

dac\_output**(**volt**);**

**}**

**}**

//-------------------------------------------------------------------------------------------

// Timer\_Init

//-------------------------------------------------------------------------------------------

void TIMER\_Init**(**void**)**

**{**

//Timer2, used for UART0, baud 115200

SFRPAGE **=** TMR2\_PAGE**;**

TMR2CN **=** 0x04**;**

TMR2CF **=** 0x08**;**

RCAP2L **=** 0xF4**;**

RCAP2H **=** 0xFF**;**

**}**

//-------------------------------------------------------------------------------------------

// PORT\_Init

//-------------------------------------------------------------------------------------------

void PORT\_Init**()**

**{**

SFRPAGE **=** CONFIG\_PAGE**;**

XBR0 **=** 0x04**;** //eanble UART0

XBR1 **=** 0x00**;**

XBR2 **=** 0x40**;**

P0MDOUT **=** 0x05**;**

P0 **=** **~**0x05**;**

P1MDOUT **&=** 0xFE**;**

P1 **|=** 0x01**;** //PB input on p1.0

//P1MDIN = 0xFE;

**}**

//-------------------------------------------------------------------------------------------

// SYSCLK\_Init

//-------------------------------------------------------------------------------------------

void SYSCLK\_Init**(**void**)**

**{**

int i **=** 0**;**

SFRPAGE **=** CONFIG\_PAGE**;**

OSCXCN **=** 0x67**;**

**for** **(**i **=** 0**;** i **<** 3000**;** i**++);** // Wait 1ms for initialization

**while** **((**OSCXCN **&** 0x80**)** **==** 0**);**

CLKSEL **=** 0x01**;**

OSCICN **&=** **~**0x80**;**

**}**

//-------------------------------------------------------------------------------------------

// UART0\_Init

//-------------------------------------------------------------------------------------------

void UART\_Init**()**

**{**

//UART0

SFRPAGE **=** UART0\_PAGE**;**

SCON0 **=** 0x50**;**

SSTA0 **=** 0x05**;**

TI0 **=** 1**;**

**}**

void ADC\_Init**(**void**)**

**{**

SFRPAGE **=** 0**;**

REF0CN **=** 0x03**;** // Set Vref to use internal reference voltage (2.4 V)

AMX0CF **&=** 0xF0**;** // Configure A/D pins

AMX0SL **&=** 0xF0**;** // Set AIN0.0 to single ended input

ADC0CF **|=** 0x28**;** // Set A/D converter gain to 1 and define SAR clock speed 2Mhz

ADC0CN **=** 0x80**;** // Enable A/D converter (ADC2)

**}**

void DAC\_Init**()**

**{**

SFRPAGE **=** DAC0\_PAGE**;**

DAC0CN **=** 0x80**;**

**}**

unsigned short int read\_AD\_input**(**void**)**

**{**

unsigned short int volt**;**

SFRPAGE **=** 0x00**;**

AD0BUSY **=** 1**;** //Start conversion

**while** **(!**AD0INT**);** //wait for ADC to finish

volt **=** ADC0H**<<**8 **|** ADC0L**;**

//printf("currently: %i \r\n", volt);

**return** volt**;**

**}**

void dac\_output**(**unsigned short val**)**

**{**

DAC0L **=** val**;**

DAC0H **=** val **>>** 8**;**

**}**

**Appendix D – Code for Part 4**

// Lab 4 Part 4.c

// Chris Pybus and Jeff Pistacchio

//

//This program polls an Analog input and outputs the voltage in hex and dec

//-------------------------------------------------------------------------------------------

// Includes

//-------------------------------------------------------------------------------------------

#include <c8051f120.h> // SFR declarations.

#include <stdio.h>

#include "putget.h" // Necessary for printf

\_\_sbit \_\_at **(**0x90**)** PB**;**

void SYSCLK\_Init**(**void**)**

**{**

int i **=** 0**;**

SFRPAGE **=** CONFIG\_PAGE**;**

OSCXCN **=** 0x67**;**

**for** **(**i **=** 0**;** i **<** 3000**;** i**++);** // Wait 1ms for initialization

**while** **((**OSCXCN **&** 0x80**)** **==** 0**);**

CLKSEL **=** 0x01**;**

OSCICN **&=** **~**0x80**;**

**}**

void TIMER\_Init**(**void**)**

**{**

//Timer2, used for UART0, baud 115200

SFRPAGE **=** TMR2\_PAGE**;**

TMR2CN **=** 0x04**;**

TMR2CF **=** 0x08**;**

RCAP2L **=** 0xF4**;**

RCAP2H **=** 0xFF**;**

**}**

void UART\_Init**()**

**{**

//UART0

SFRPAGE **=** UART0\_PAGE**;**

SCON0 **=** 0x50**;**

SSTA0 **=** 0x05**;**

TI0 **=** 1**;**

**}**

void PORT\_Init**()**

**{**

SFRPAGE **=** CONFIG\_PAGE**;**

XBR0 **=** 0x04**;** //eanble UART0

XBR1 **=** 0x00**;**

XBR2 **=** 0x40**;**

P0MDOUT **=** 0x05**;**

P0 **=** **~**0x05**;**

P1MDOUT **&=** 0xFE**;**

P1 **|=** 0x01**;** //PB input on p1.0

**}**

void ADC\_Init**(**void**)**

**{**

SFRPAGE **=** ADC0\_PAGE**;**

REF0CN **=** 0x03**;** // Set Vref to use internal reference voltage (2.4 V)

AMX0CF **&=** 0xF0**;** // Configure A/D pins

AMX0SL **&=** 0xF0**;** // Set AIN0.0 to single ended input

ADC0CF **=** 0x28**;** // Set A/D converter gain to 1 and define SAR clock speed 2Mhz

ADC0CN **=** 0x81**;** // Enable A/D converter (ADC) and left justify output

**}**

void DAC\_Init**()**

**{**

SFRPAGE **=** DAC0\_PAGE**;**

DAC0CN **=** 0x84**;** //left aligned

**}**

void MAC\_Init**(**void**)**

**{**

MAC0CF **=** 0x0A**;**

**}**

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

**{**

unsigned short int x2 **=** 0**;** //2 inputs ago

unsigned short int x1 **=** 0**;** //previous input

unsigned short int x **=** 0**;** //current input

unsigned short int y1 **=** 0**;** //previous output

unsigned short int y **=** 0**;** //current output.

WDTCN **=** 0xDE**;** // Disable watchdog timer.

WDTCN **=** 0xAD**;**

SFRPAGE **=** CONFIG\_PAGE**;**

SYSCLK\_Init**();** // Initialize the oscillator.

TIMER\_Init**();** // Initialize timers

UART\_Init**();** // Initialize UARTs.

PORT\_Init**();** // Configure the Crossbar and GPIO.

ADC\_Init**();**

DAC\_Init**();**

printf**(**"\033[2J"**);** //Erase screen and move cursor to the home position.

printf**(**"Press push button to start part 4"**);**

**while(**PB**);**

printf**(**"\rRunning Part 4 \n\n\r"**);**

**while(**1**)**

**{**

SFRPAGE **=** ADC0\_PAGE**;**

AD0BUSY **=** 1**;** //Start conversion

**while** **(!**AD0INT**);** //wait for ADC to finish

x **=** **((**ADC0H **<<** 8**)** **|** ADC0L**);** //Turn 12 bit adc output to 16 bit fractional value

//MSB must be zero (if positive) bc 2'c compliment

//X will become a number between 0 and 1 that's relative to 0 to 4096

//ie, if ADC measures 2048, x will be .5

//MAC Operations///////////////////////////

SFRPAGE **=** MAC0\_PAGE**;**

MAC0CF **=** 0x0A**;** //clear accumulator and operate in fractional mode

// Operation: 10/32 \* x(k)

// set Mac A value: Coefficient

// 10/32 = .3125 => 0|010 1000 0000 0000 => 0x2800

MAC0AH **=** 0x28**;**

MAC0AL **=** 0x00**;**

//Set Mac B value

MAC0BH **=** x**>>**8**;**

MAC0BL **=** x**;**

// Operation: 25/104 \* x(k-1)

// set Mac A value: Coefficient

// 25/104 = 0.24038 => 0|001 1110 1100 0100 => 0x1EC4

MAC0AH **=** 0x1E**;**

MAC0AL **=** 0xC4**;**

//Set Mac B value

MAC0BH **=** x1**>>**8**;**

MAC0BL **=** x1**;**

// Operation: 10/32 \* x(k-2)

// set Mac A value: Coefficient

// 10/32 = .3125 => 0|010 1000 0000 0000 => 0x2800

MAC0AH **=** 0x28**;**

MAC0AL **=** 0x00**;**

//Set Mac B value

MAC0BH **=** x2**>>**8**;**

MAC0BL **=** x2**;**

// Operation: 19/64 \* y(k-1)

// set Mac A value: Coefficient

// 19/64 = .296875 => 0|010 0110 0000 0000 => 0x2600

MAC0AH **=** 0x26**;**

MAC0AL **=** 0x00**;**

//Set Mac B value

MAC0BH **=** y1**>>**8**;**

MAC0BL **=** y1**;**

SFRPAGE **=** MAC0\_PAGE**;** //wait for operation

y **=** **(**MAC0ACC3**<<**8**)** **|** MAC0ACC2**;**

//reset stored values

y1 **=** y**;**

x2 **=** x1**;**

x1 **=** x**;**

//output y to DAC

SFRPAGE **=** DAC0\_PAGE**;**

DAC0H **=** y **>>** 8**;**

DAC0L **=** y**;**

**}**

**}**