## Introduction

This introductory lab is meant to familiarize the lab participants with the software and hardware used in Microprocessor Systems. The SiLabs IDE using a SDCC compiler are used to program the 8051 microprocessor. This lab is broken down into 3 parts. The first reads a keyboard input and returns the key pressed via the serial output. This loop is repeated until <ESC> is pressed. The second is an expansion of the first, but involves interfacing with an ANSI terminal and setting its various display options with escape codes. The last part involves setting certain port as input and a separate one as output. The output, represented by LEDs, is controlled by the input, represented by potentiometers.

## Materials and Methods

For the first part of the lab we used getchar() to read a keyboard input into a char, and then printf(..) with %s to display it on the terminal.  
For the second part, we added the features one by one and tested them to make sure they worked correctly before moving onto the next. The program accepts ASCII characters 32 to 126 as printable; other ASCII characters (besides 27) are deemed unprintable and their hex codes printed instead. Escape characters such as ESC or the arrow keys exit the program.  
For the third part, pins on Port 2 were set to output, and pins on Port 1 to input and high-impedance. The outputs led to 1k ohm resistors connected LEDs, whose other ends were connected to ground, and the inputs were connected to potentiometers with their other ends at +5V. The output was set equal to the input, so that when the resistance of the potentiometer is increased such that the input goes LOW, the output also goes LOW and the LED it is connected to turns off.

## Results / Analysis

For part 1, we successfully displayed the input character. For part 2, we successfully added the effects. For part three, we successfully mirrored the input on the output for four LEDs.

## Conclusion

Our results from this lab were within expectations. The first part of the lab, ensuring the configuration of the serial output was correct went without a hitch. The ANSI terminal presented a more difficult challenge, particularly with the aging machines running the ANSI terminals. The escape codes required major focus, as the distributed attributes often had to be reset and would result in undesired attributes in incorrect locations. The last portion of the lab resulted in our discovery of a faulty buffer or port on the board, most likely saving a few headaches in the future.

# Appendices

Lab 1-1

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

// Lab1-1.c

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

// Receives an input from the keyboard and outputs the key pressed onto the screen

// Program is terminated by the <ESC> key

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

// Includes

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

#include <c8051f120.h>

#include <stdio.h>

#include "putget.h"

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

// Global Constants

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

#define EXTCLK 22118400 // External oscillator frequency in Hz

#define SYSCLK 22118400 // Output of crystal oscillator

#define BAUDRATE 28800 // UART baud rate in bps

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

// Function Prototypes

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

void main(void);

void SYSCLK\_INIT(void);

void PORT\_INIT(void);

void UART0\_INIT(void);

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

// MAIN Routine

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

void main(void)

{

char choice;

WDTCN = 0xDE; // Disable the watchdog timer

WDTCN = 0xAD; // Note: = "DEAD"!

SYSCLK\_INIT(); // Initialize the oscillator

PORT\_INIT(); // Initialize the Crossbar and GPIO

UART0\_INIT(); // Initialize UART0

SFRPAGE = UART0\_PAGE; // Direct output to UART0

//print the program info

printf("Returns input char. To exit hit ESC.\n\n\r");

while(1)

{

choice=getchar();

if (choice >= 32 && choice <= 126)

{

printf("\n\rThe keyboard character is %c.\n\r", choice);

}

else

{

if(choice ==27){

//exitprogram

printf("Program exit \n\r");

return;

}

}

}

}

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

// SYSCLK\_Init

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

//

// Initialize the system clock to use a 22.1184MHz crystal as its clock source

//

void SYSCLK\_INIT(void)

{

int i;

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page SFRPAGE = CONFIG\_PAGE;

SFRPAGE = CONFIG\_PAGE;

OSCXCN = 0x67; // Start ext osc with 22.1184MHz crystal

for(i=0; i < 3000; i++); // Wait for the oscillator to start up

while(!(OSCXCN & 0x80));

CLKSEL = 0x01; // Switch to the external crystal oscillator

OSCICN = 0x00; // Disable the internal oscillator

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}

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

// PORT\_Init

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

//

// Configure the Crossbar and GPIO ports

//

void PORT\_INIT(void)

{

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page

SFRPAGE = CONFIG\_PAGE;

XBR0 = 0x04; // Enable UART0

XBR1 = 0x00;

XBR2 = 0x40; // Enable Crossbar and weak pull-up

P0MDOUT |= 0x01; // Set TX0 on P0.0 pin to push-pull

P1MDOUT |= 0x40; // Set green LED ooutput P1.6 to push-pull

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}

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

// UART0\_Init

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

//

// Configure the UART0 using Timer1, for <baudrate> and 8-N-1

//

void UART0\_INIT(void)

{

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page

SFRPAGE = TIMER01\_PAGE;

TCON = 0x40;

TMOD &= 0x0F;

TMOD |= 0x20; // Timer1, Mode 2, 8-bit reload

CKCON |= 0x10; // Timer1 uses SYSCLK as time base

// TH1 = 256 - SYSCLK/(BAUDRATE\*32) Set Timer1 reload baudrate value T1 Hi Byte

TH1 = 0xE8; // 0xE8 = 232

TR1 = 1; // Start Timer1

SFRPAGE = UART0\_PAGE;

SCON0 = 0x50; // Mode 1, 8-bit UART, enable RX

SSTA0 = 0x00; // SMOD0 = 0, in this mode

// TH1 = 256 - SYSCLK/(baud rate \* 32)

TI0 = 1; // Indicate TX0 ready

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}

Lab 1-2

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

// Lab1-2

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

// This program communicates with an ANSI terminal. A keyboard input is expected.

// Printable items are displayed on an overwriting line. Non-printable items are

// displayed on a scrolling panel with blinking text.

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

// Includes

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

#include <c8051f120.h>

#include <stdio.h>

#include "putget.h"

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

// Global Constants

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

#define EXTCLK 22118400 // External oscillator frequency in Hz

#define SYSCLK 22118400 // Output of crystal oscillator

#define BAUDRATE 28800 // UART baud rate in bps

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

// Function Prototypes

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

void main(void);

void SYSCLK\_INIT(void);

void PORT\_INIT(void);

void UART0\_INIT(void);

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

// MAIN Routine

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

void main(void)

{

char choice;

char choice2;

int printOnThisLine = 11;

WDTCN = 0xDE; // Disable the watchdog timer

WDTCN = 0xAD; // Note: = "DEAD"!

SYSCLK\_INIT(); // Initialize the oscillator

PORT\_INIT(); // Initialize the Crossbar and GPIO

UART0\_INIT(); // Initialize UART0

SFRPAGE = UART0\_PAGE; // Direct output to UART0

printf("\033[2J"); // Erase ANSI terminal & move cursor to home position

printf("\033[33m\033[1B\033[21C"); // makes text yellow, pushes cursor down a line

//print the program info

printf("Returns input char. To exit hit ESC.\n\n\r");

//define scrollable area

printf("\033[12;24r");

while(1)

{

choice=getchar();

if (choice >= 32 && choice <= 126)

{ //reset attributes

printf("\033[0m");

//reapply attributes

printf("\033[44;33m");

//put cursor over line 6

printf("\033[6;0H");

//print

printf("\n\rThe keyboard character is \033[37m%c\033[33m.\n\r", choice);

}

else

{

if(choice ==27){

//exitprogram

printf("\033[m");

printf("\033[24;0H");

printf("\033[44;33m");

printf("Program exit \n\r");

return;

}

//check which line we want to print to

if(printOnThisLine == 24)

{

//scroll up

printf("\033[m");

//setbackground color

printf("\033[44;33m");

}

else

{

printOnThisLine++;

}

//go to scrollable area

printf("\033[%d;0H", printOnThisLine);

//enable blinking text

printf("\033[5m");

//print

printf("The character $%X is '\033[4mnot printable'\033[0m\033[5;44;33m.\n\r", choice);

//beep

putchar(7);

}

}

}

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

// SYSCLK\_Init

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

//

// Initialize the system clock to use a 22.1184MHz crystal as its clock source

//

void SYSCLK\_INIT(void)

{

int i;

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page SFRPAGE = CONFIG\_PAGE;

SFRPAGE = CONFIG\_PAGE;

OSCXCN = 0x67; // Start ext osc with 22.1184MHz crystal

for(i=0; i < 3000; i++); // Wait for the oscillator to start up

while(!(OSCXCN & 0x80));

CLKSEL = 0x01; // Switch to the external crystal oscillator

OSCICN = 0x00; // Disable the internal oscillator

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}

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

// PORT\_Init

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

//

// Configure the Crossbar and GPIO ports

//

void PORT\_INIT(void)

{

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page

SFRPAGE = CONFIG\_PAGE;

XBR0 = 0x04; // Enable UART0

XBR1 = 0x00;

XBR2 = 0x40; // Enable Crossbar and weak pull-up

P0MDOUT |= 0x01; // Set TX0 on P0.0 pin to push-pull

P1MDOUT |= 0x40; // Set green LED ooutput P1.6 to push-pull

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}

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

// UART0\_Init

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

//

// Configure the UART0 using Timer1, for <baudrate> and 8-N-1

//

void UART0\_INIT(void)

{

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page

SFRPAGE = TIMER01\_PAGE;

TCON = 0x40;

TMOD &= 0x0F;

TMOD |= 0x20; // Timer1, Mode 2, 8-bit reload

CKCON |= 0x10; // Timer1 uses SYSCLK as time base

// TH1 = 256 - SYSCLK/(BAUDRATE\*32) Set Timer1 reload baudrate value T1 Hi Byte

TH1 = 0xE8; // 0xE8 = 232

TR1 = 1; // Start Timer1

SFRPAGE = UART0\_PAGE;

SCON0 = 0x50; // Mode 1, 8-bit UART, enable RX

SSTA0 = 0x00; // SMOD0 = 0, in this mode

// TH1 = 256 - SYSCLK/(baud rate \* 32)

TI0 = 1; // Indicate TX0 ready

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}

Lab1-3

// Lab1-3.c

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

// This program takes inputs from port 1 and outputs them to port 2

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

// Includes

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

#include <c8051f120.h>

#include <stdio.h>

#include "putget.h"

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

// Global Constants

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

#define EXTCLK 22118400 // External oscillator frequency in Hz

#define SYSCLK 22118400 // Output of crystal oscillator

#define BAUDRATE 28800 // UART baud rate in bps

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

// Function Prototypes

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

void main(void);

void SYSCLK\_INIT(void);

void PORT\_INIT(void);

void UART0\_INIT(void);

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

// MAIN Routine

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

void main(void)

{

char choice;

WDTCN = 0xDE; // Disable the watchdog timer

WDTCN = 0xAD; // Note: = "DEAD"!

SYSCLK\_INIT(); // Initialize the oscillator

PORT\_INIT(); // Initialize the Crossbar and GPIO

UART0\_INIT(); // Initialize UART0

SFRPAGE = UART0\_PAGE; // Direct output to UART0

printf("Reading inputs to P1\n\r");

while(1)

{

P2 = (P1 & 0x1E); //set P2 outputs equal to P1 inputs; ignore pins 0, 5-7

printf("P2: %x P1 : %x\n\r", P2, P1); //print current values for each port (in hex)

}

}

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

// SYSCLK\_Init

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

//

// Initialize the system clock to use a 22.1184MHz crystal as its clock source

//

void SYSCLK\_INIT(void)

{

int i;

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page SFRPAGE = CONFIG\_PAGE;

SFRPAGE = CONFIG\_PAGE;

OSCXCN = 0x67; // Start ext osc with 22.1184MHz crystal

for(i=0; i < 3000; i++); // Wait for the oscillator to start up

while(!(OSCXCN & 0x80));

CLKSEL = 0x01; // Switch to the external crystal oscillator

OSCICN = 0x00; // Disable the internal oscillator

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}

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

// PORT\_Init

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

//

// Configure the Crossbar and GPIO ports

//

void PORT\_INIT(void)

{

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page

SFRPAGE = CONFIG\_PAGE;

XBR0 = 0x04; // Enable UART0

XBR1 = 0x00;

XBR2 = 0x40; // Enable Crossbar and weak pull-up

P0MDOUT |= 0x01; // Set TX0 on P0.0 pin to push-pull

P1MDOUT &= 0xE0; // Port 1 pins 0-4 to open-drain; pin 0 does not work.

P1 |= ~0xE0; // set pins 0-4 to high-impedence

P2MDOUT |= 0x1F; // Port 2 pins 0-4 to output

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}

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

// UART0\_Init

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

//

// Configure the UART0 using Timer1, for <baudrate> and 8-N-1

//

void UART0\_INIT(void)

{

char SFRPAGE\_SAVE;

SFRPAGE\_SAVE = SFRPAGE; // Save Current SFR page

SFRPAGE = TIMER01\_PAGE;

TCON = 0x40;

TMOD &= 0x0F;

TMOD |= 0x20; // Timer1, Mode 2, 8-bit reload

CKCON |= 0x10; // Timer1 uses SYSCLK as time base

// TH1 = 256 - SYSCLK/(BAUDRATE\*32) Set Timer1 reload baudrate value T1 Hi Byte

TH1 = 0xE8; // 0xE8 = 232

TR1 = 1; // Start Timer1

SFRPAGE = UART0\_PAGE;

SCON0 = 0x50; // Mode 1, 8-bit UART, enable RX

SSTA0 = 0x00; // SMOD0 = 0, in this mode

// TH1 = 256 - SYSCLK/(baud rate \* 32)

TI0 = 1; // Indicate TX0 ready

SFRPAGE = SFRPAGE\_SAVE; // Restore SFR page

}