Lab Report 5

This lab provides exercises towards communication systems with emphasis on I2C communication method being carried on PIC24f MCU to The Sitronix ST7032 Switch Science Small LCD. The demonstration of the lab is to print strings of words on the LCD using I2C protocols. Below is the set of functions that achieve this purpose.

```
void lcd_cmd(char Package){
   I2C2CONbits.SEN=1;
   while(I2C2CONbits.SEN==1);
   IFS3bits.MI2C2IF=0;
   I2C2TRN = 0b01111100; // 8-bits consisting of the slave address and the R/nW bit
   while(IFS3bits.MI2C2IF == 0 );// *Refer to NOTE below*
   IFS3bits.MI2C2IF=0;
   I2C2TRN = 0b00000000; // 8-bits consisting of control byte
   while(IFS3bits.MI2C2IF == 0 );
   IFS3bits.MI2C2IF=0;
   I2C2TRN = Package; // 8-bits consisting of the data byte
   while(IFS3bits.MI2C2IF == 0 );
   IFS3bits.MI2C2IF=0;
   I2C2CONbits.PEN = 1;
   while(I2C2CONbits.PEN==1); // PEN will clear when Stop bit is complete
}
```

This function is the main method that a user uses to communicate with the LCD, it consists of several parts. The first part is the starting bit, and then the second part is the slave address of the LCD plus the Read or Write bit which in the case of this lab is always going to be to write. The next part is the control byte that is instructed by the data sheet. The last part is the data the user wants to transmit and at the end is the stop bit. Ignored here are the acknowledgement bits. Since an oscilloscope is not used there is no way the acknowledgement bit from the slave can be captured, however the use of polling is able to guarantee that each process is able to be finished before the next instruction.

```
void setup(){
   CLKDIVbits.RCDIV = 0;  //Set RCDIV=1:1 (default 2:1) 32MHz or FCY/2=16M
   AD1PCFG = 0x9fff;
   I2C2BRG=0x9d;
   I2C2CONbits.I2CEN=0;
   I2C2CONbits.I2CEN=1;
   IFS3bits.MI2C2IF=0;
```

```
}
```
The setup function turns the I2C function inside the MCU on and sets the calculated baudrate 157 under given frequency 16MHz and SCL 100kHz. It also clears the interrupt bit to make sure no process is being on.

```
void lcd_init(void){
   int i;
   for(i=0;i<50;i++){
      xie_wait_1ms();
   }
   lcd_cmd(0b00111000); // function set, normal instruction mode
   lcd_cmd(0b00111001); // function set, extended instruction mode
   lcd_cmd(0b00010100); // interval osc
   lcd_cmd(0b01110000); // contrast C3-C0
   lcd_cmd(0b01010110); // Ion, Bon, C5-C4
   lcd_cmd(0b01101100); // follower control
   for(i=0;i<200;i++){
      xie_wait_1ms();
   }
   lcd_cmd(0b00111000); // function set, normal instruction mode
   lcd_cmd(0b00001100); // Display On
   lcd_cmd(0b00000001); // Clear Display
   xie_wait_1ms();
   xie_wait_1ms();
}
```
This function is the setup of the Sitronix ST7032 LCD with given instructions in steps from the data sheet transmitted. It will be called along with the setup() function inside the main to make the LCD ready to operate.

```
void lcd_setCursor(char x, char y){
   lcd_cmd(0b10000000|(0x40*y+x));
}
```
This function follows the calculation pattern of inputs from the user to the position on the LCD display, noted here is that the first bit is required to always be 1.

```
void lcd_printChar(char Package) {
   I2C2CONbits.SEN= 1;    //Initiate Start condition
   while(I2C2CONbits.SEN==1);  // SEN will clear when Start Bit is complete
   IFS3bits.MI2C2IF = 0;
```

```
    I2C2TRN = 0b01111100; // 8-bits consisting of the slave address and the R/nW bit
    while(IFS3bits.MI2C2IF==0);
    IFS3bits.MI2C2IF = 0;
    I2C2TRN = 0b01000000; // 8-bits consisting of control byte /w RS=1
    while(IFS3bits.MI2C2IF==0);
    IFS3bits.MI2C2IF=0;
    I2C2TRN = Package; // 8-bits consisting of the data byte
    while(IFS3bits.MI2C2IF==0);
    IFS3bits.MI2C2IF = 0;
    I2C2CONbits.PEN = 1;
    while(I2C2CONbits.PEN==1); // PEN will clear when Stop bit is complete
}
```

This function achieves to print characters on the LCD display. It follows basically the same communication protocol as the lcd_cmd() function does with the only difference that in the control byte the RS bit has to be 1 in order to be able to print characters.

```
void lcd_printStr(const char s[]){
    int size=strlen(s);
    I2C2CONbits.SEN= 1;    //Initiate Start condition
    while(I2C2CONbits.SEN==1);  // SEN will clear when Start Bit is complete
    IFS3bits.MI2C2IF = 0;
    I2C2TRN = 0b01111100; // 8-bits consisting of the slave address and the R/nW bit
    while(IFS3bits.MI2C2IF==0);
    IFS3bits.MI2C2IF = 0;
    int i;
    for(i=0;i<size-1;i++){
        I2C2TRN = 0b11000000; // 8-bits consisting of control byte /w RS=1
        while (IFS3bits.MI2C2IF==0);
        IFS3bits.MI2C2IF = 0;
        I2C2TRN = s[i]; // 8-bits consisting of the data byte
        while (IFS3bits.MI2C2IF==0);
        IFS3bits.MI2C2IF = 0;
    }
    I2C2TRN = 0b01000000; // 8-bits consisting of control byte /w RS=1, Co=0
    while (IFS3bits.MI2C2IF==0);
    IFS3bits.MI2C2IF = 0;
    I2C2TRN = s[size-1]; // 8-bits consisting of the data byte
    while (IFS3bits.MI2C2IF==0);
    IFS3bits.MI2C2IF = 0;
```

```
    I2C2CONbits.PEN = 1;
    while(I2C2CONbits.PEN==1); // PEN will clear when Stop bit is complete
}
```

This function is able to let the user print strings of words on the display. It follows the same I2C communication protocol. However in the process of printing each character, if a character that is being printed is not the last one desired to print, the Co bit has to be set to 1, and in contrary if a character is the last one to be printed as desired, the Co bit has to be set 0 to notify the LCD. Here, a for loop that gets counted towards the length of the desired string is able to achieve the goal. When the last character is about to be printed, the loop ends and the Co bit becomes 0 before the printing of the last character. An extra function inside this function that is used is the strlen() function that returns the size of the string. It is under the “string.h” library that is included at the beginning.

```
int main(void) {
    setup();
    lcd_init();
    lcd_setCursor(0,0);
    lcd_printStr("Hello");
    lcd_setCursor(2,1);
    lcd_printStr("World!");
    while (1) {
    }
    return 0;
}
```

This is the main function that combines every function used to print the string “Hello World!” on the display. First the setup() and lcd_init() functions are called to make I2C and the LCD ready to operate. Then the Cursor is set at row 0 column 0 position for the word “Hello”, noted here is that as long as the initial position of a string is set the rest of the string will be printed one after each automatically until if the space in the line is used up. The string “World!” follows the same protocol and gets printed on the display starting at the position column 2 row 1. This whole method achieves the goal of the demonstration of printing strings on the display.