|  |
| --- |
| **EMBEDDED SYSTEM LABORATORY** |
| **LAB 3** |

**MULTITASK PROGRAMMING FOR ARM MICROCONTROLLER WITH ADC INTERFACE**

### I. LAB OBJECTIVES

### - In this Lab students will learn about ARM-CORTEX M3 (LPC1768) Microcontroller.

### - This Lab experiments are intended to implement basic ADC of ARM-CORTEX M3 Microcotroller to pheriperal devices in MB1700 Kit and write C code programming to control these devices.

### II. PRE-LAB : ADC Register Review

A screenshot of a computer program

Description automatically generated

A screenshot of a computer registration form

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer error

Description automatically generated

A screenshot of a computer

Description automatically generated

A black and white text on a white background

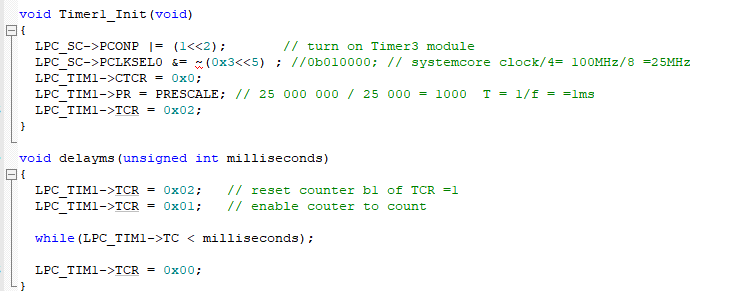
Description automatically generated

### III. LAB PROCERUCE

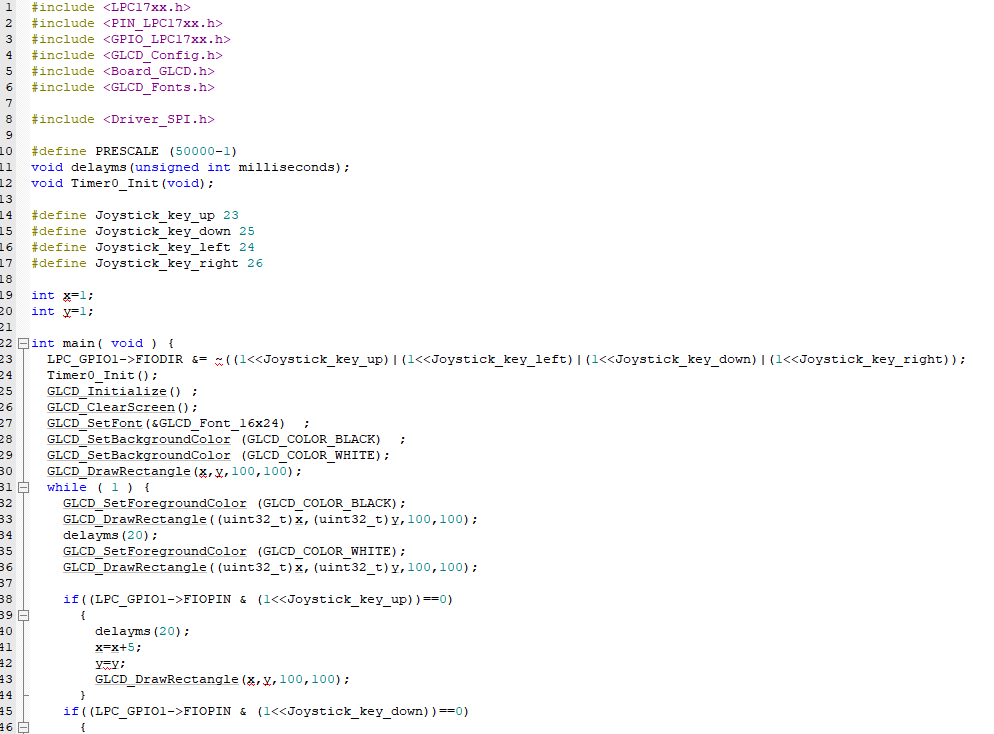
### The LPC1768 Microconttroler KIT using 100MHz system clock.

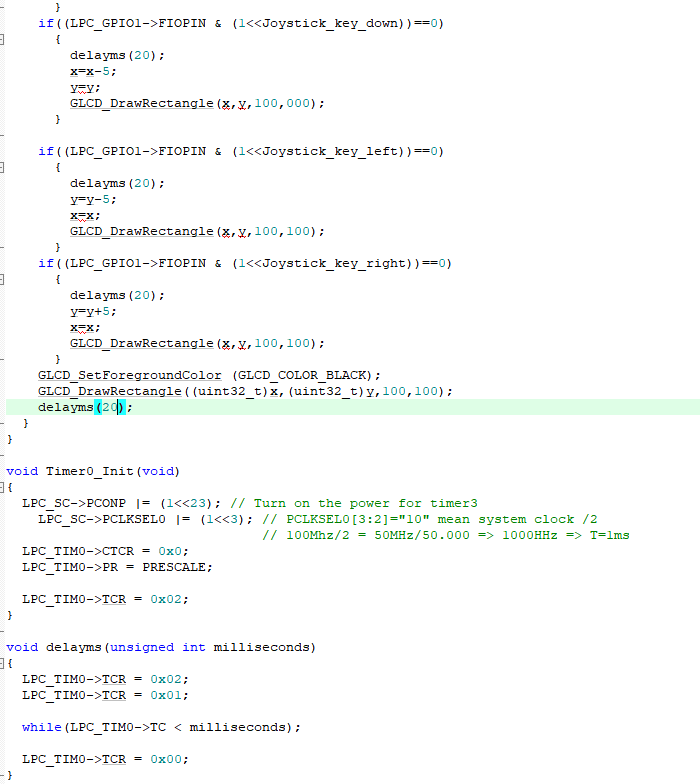
### III.1 Lab Experiment 1 : Write the C code to use GLCD Driver Library in GLCD to draw a rectangle 100x100 pixel at position x=50, y=50, then move the rectangle in different position of the LCD screen with moving delay 1 second using timer 1 polling method;

### 



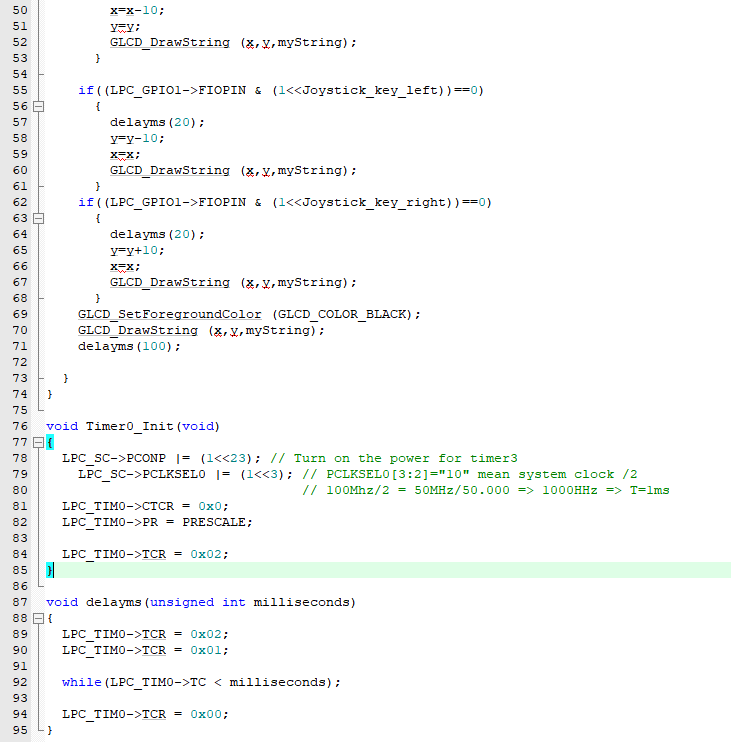
### III.2 Lab Experiment 2 : Write the C code to use GLCD Driver Library in GLCD to draw a bar graph 100x100 pixel at position x=150, y=5; then move the bar graph in different position of the LCD screen with joystick controlled by four buttons P1.23,P1.24,P1.25,P1.26;





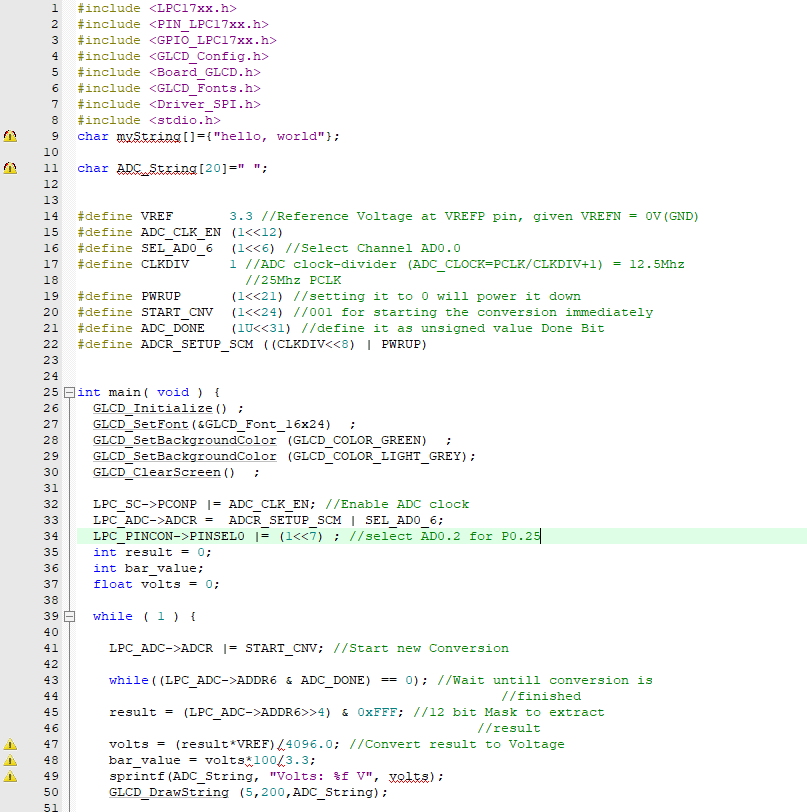
### III.3 Lab Experiment 3 : Write the C code to use GLCD Driver Library in GLCD to display your name at position x=10, y=160 ; then scroll up your name in the screen.

### 



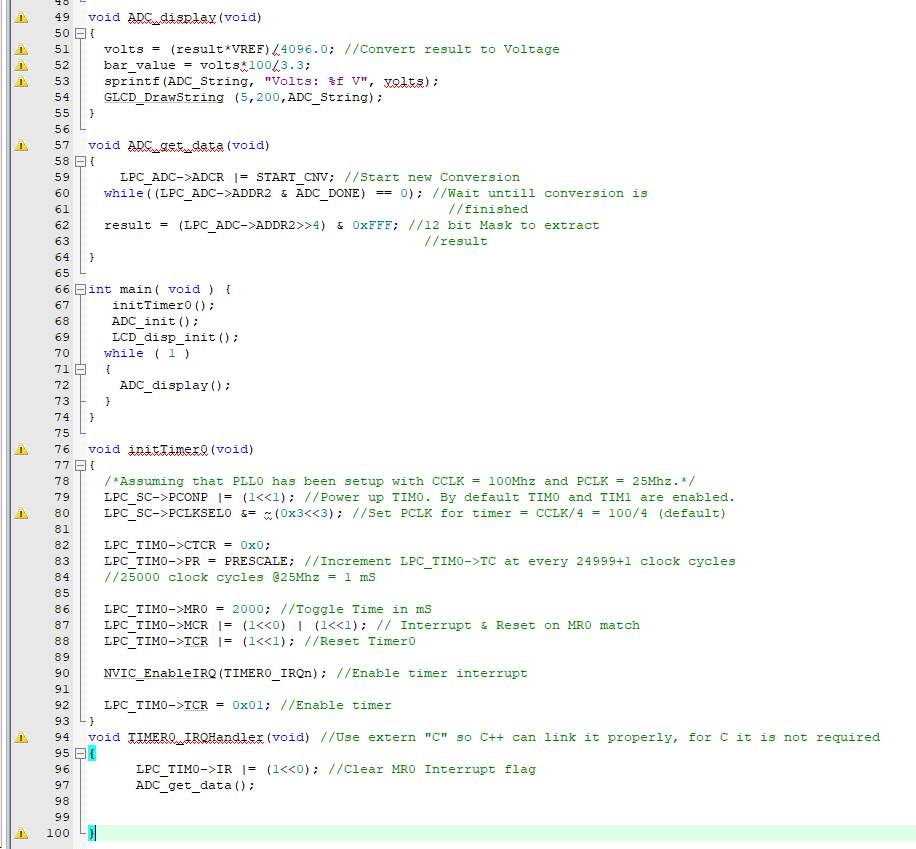
### III.4 Lab Experiment 4 : Write the C code to design the simple snake game. use GLCD Driver Library in GLCD to draw a rectangle 250x250 pixel at position x=1, y=1 ( considered as the wall). draw a rectangle 10x10 pixel at position x=20 y=20, withbgreen colors (considered as a snake); then move the snake to different position of the LCD screen with joystick controlled by four buttons P1.23,P1.24,P1.25,P1.26; When the snake hit the wall, stop the while loop and print out to the GLCD “you are lost”.

### III.5 Lab Experiment 5 : Write the C code to get data From ADC channel 2 using polling method then display 8-bit high value result to 8 bit LED in the Experiment KIT .



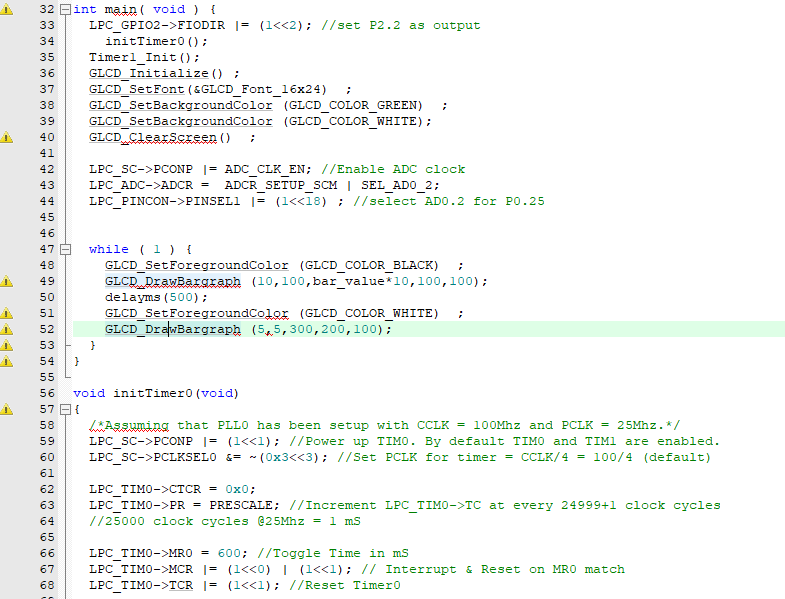
### III.6 Lab Experiment 6 : Write the code to using Timer1 Interrupt to get data From ADC channel 2 using polling method then display 8-bit high value result to 8 bit LED with sampling time 1 second/1 time.

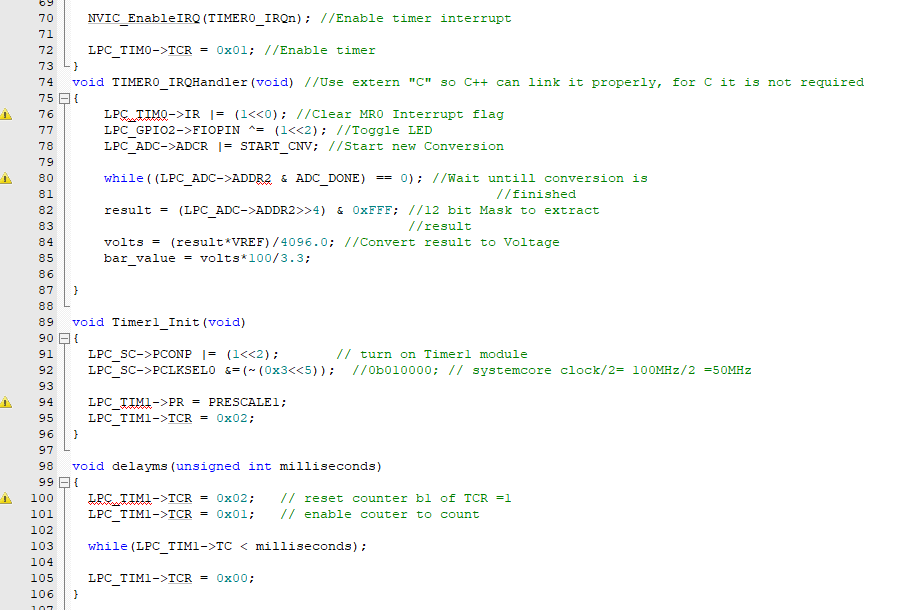
### 



### III.7 Lab Experiment 7 : Write the code to using Timer1 Interrupt to get data From ADC channel 2 using polling method then display 12-bit value result to GLCD in form of text with sampling time 1 second/1 time.

### 





### III.8 Lab Experiment 8 : Write the code to using Timer1 Interrupt to get data From ADC channel 2 using polling method then display 8-bit high value result to 8 bit LED with sampling time 1 second/1 time.

### #include "LPC17xx.h"

### #define POWER\_CONTROL (LPC\_SC->PCONP)

### #define PINCONSEL1 (LPC\_PINCON->PINSEL1)

### #define PINCONSEL2 (LPC\_PINCON->PINSEL2)

### #define IO1\_DIR (LPC\_GPIO1->FIODIR)

### #define IO2\_DIR (LPC\_GPIO2->FIODIR)

### #define IO1\_SET (LPC\_GPIO1->FIOSET)

### #define IO2\_SET (LPC\_GPIO2->FIOSET)

### #define IO1\_CLR (LPC\_GPIO1->FIOCLR)

### #define IO2\_CLR (LPC\_GPIO2->FIOCLR)

### #define IO1\_PIN (LPC\_GPIO1->FIOPIN)

### #define IO2\_PIN (LPC\_GPIO2->FIOPIN)

### #define LED1\_BIT (1 << 6)

### #define LED2\_BIT (1 << 5)

### #define LED3\_BIT (1 << 4)

### #define LED4\_BIT (1 << 3)

### #define LED5\_BIT (1 << 2)

### #define LED6\_BIT (1 << 31)

### #define LED7\_BIT (1 << 29)

### #define LED8\_BIT (1 << 28)

### #define VREF 3.3 //Reference Voltage at VREFP pin, given VREFN = 0V(GND)

### #define ADC\_CLK\_EN (1<<12)

### #define SEL\_AD0\_2 (1<<2) //Select Channel AD0.0

### #define CLKDIV 1 //ADC clock-divider (ADC\_CLOCK=PCLK/CLKDIV+1) = 12.5Mhz

### //25Mhz PCLK

### #define PWRUP (1<<21) //setting it to 0 will power it down

### #define START\_CNV (1<<24) //001 for starting the conversion immediately

### #define ADC\_DONE (1U<<31) //define it as unsigned value Done Bit

### #define ADCR\_SETUP\_SCM ((CLKDIV<<8) | PWRUP)

### void leds\_init(void)

### {

### POWER\_CONTROL |= (1 << 15);

### IO2\_DIR = (LED1\_BIT | LED2\_BIT | LED3\_BIT | LED4\_BIT | LED5\_BIT);

### IO1\_DIR = (LED6\_BIT | LED7\_BIT | LED8\_BIT);

### }

### void led\_on(unsigned int index)

### {

### switch(index)

### {

### case 1:

### IO2\_SET = LED1\_BIT;

### break;

### case 2:

### IO2\_SET = LED2\_BIT;

### break;

### case 3:

### IO2\_SET = LED3\_BIT;

### break;

### case 4:

### IO2\_SET = LED4\_BIT;

### break;

### case 5:

### IO2\_SET = LED5\_BIT;

### break;

### case 6:

### IO1\_SET = LED6\_BIT;

### break;

### case 7:

### IO1\_SET = LED7\_BIT;

### break;

### case 8:

### IO1\_SET = LED8\_BIT;

### break;

### default:

### break;

### }

### }

### void led\_off(unsigned int index)

### {

### switch(index)

### {

### case 1:

### IO2\_CLR = LED1\_BIT;

### break;

### case 2:

### IO2\_CLR = LED2\_BIT;

### break;

### case 3:

### IO2\_CLR = LED3\_BIT;

### break;

### case 4:

### IO2\_CLR = LED4\_BIT;

### break;

### case 5:

### IO2\_CLR = LED5\_BIT;

### break;

### case 6:

### IO1\_CLR = LED6\_BIT;

### break;

### case 7:

### IO1\_CLR = LED7\_BIT;

### break;

### case 8:

### IO1\_CLR = LED8\_BIT;

### break;

### default:

### break;

### }

### }

### void leds\_set\_value(char led\_Value)

### {

### int i;

### for(i = 0; i < 8; i++)

### {

### if(led\_Value & (1<<i))

### {

### led\_on(i+1);

### }

### else

### {

### led\_off(i+1);

### }

### }

### }

### void ADC\_Initialize(void)

### {

### LPC\_SC->PCONP |= ADC\_CLK\_EN; //Enable ADC clock

### LPC\_ADC->ADCR = ADCR\_SETUP\_SCM | SEL\_AD0\_2;

### LPC\_PINCON->PINSEL1 |= (1<<18) ; //select AD0.2 for P0.25

### 

### }

### void ADC\_StartConversion(void)

### {

### LPC\_ADC->ADCR |= START\_CNV; //Start new Conversion

### }

### int main()

### {

### leds\_init();

### ADC\_Initialize();

### int ADC\_Value;

### char led\_value;

### while(1)

### {

### ADC\_StartConversion();

### while((LPC\_ADC->ADDR2 & ADC\_DONE) == 0); //Wait untill conversion is

### //finished

### ADC\_Value = (LPC\_ADC->ADDR2>>8) & 0xFF; //12 bit Mask to extract

### //result

### led\_value = (char) ADC\_Value;

### leds\_set\_value(led\_value);

### }

### }

**IV. LAB PERFORMANCE GRADING AND LAB REPORT GUIDELINES**

For each Lab experiment Students show the successful running results to Lab Instructor for Lab Performance grading.

Students write a report which includes : Algorithm flowchart and C++ Code for each experiment. In each block of the code or line of code, give the comments for the meaning of this block of code.