**LPC EXPERIMENTS**

**EXP NO: 21**

WRITE AN ASSEMBLY LANGUAGE PROGRAM FOR LED BLINK USING LPC2148.

**AIM :**

To write an assembly language program for LED blink using LPC2148.

**SOFTWARE/HARDARE required :**

* Keil uvision 5 (software)
* Philips Flash utility (software)
* LPC2148 trainer kit (hardware)

**PROCEDURE :**

* Open the keil software, create new project file.
* Enter LEGACY DEVICE DRIVER – LPC2148.
* Click Yes.
* Open project and select “option for target”.
* Select output and linker then enable CREATE HEX file.
* Add target file to the source file.
* Click the Build target options.
* Check the error
* Open LPC2000 (Philips flash utility software)
* Set the Baud rate 38400
* Set XTAL frequency 12000.
* Import HEX file
* Dump the hex file into the LPC2148 trainer kit.
* Check the output and Note it.

**PROGRAM:**

#include "lpc214x.h"

void delay (unsigned int k);

void main(void)

{

IODIR0 = 0xFFFFFFFF; //Configure Port0 as output Port

PINSEL0 = 0; //Configure Port0 as General Purose IO

while(1)

{

IOSET0 = 0x0000ff00; //Set P0.15-P0.8 to '1'

delay(1000); //1 sec Delay

IOCLR0 = 0x0000ff00; //Set P0.15-P0.8 to '0'

delay(1000); //1 Sec Delay

}

}

//Delay Program

//Input - delay value in milli seconds

void delay(unsigned int k)

{

unsigned int i,j;

for (j=0; j<k; j++)

for(i = 0; i<=800; i++);

}

**RESULT :**

Thus the program has been successfully verified and executed.

**EXP NO: 22**

WRITE AN ASSEMBLY LANGUAGE PROGRAM FOR SEVEN SEGMENT DISPLAY USING LPC2148.

**AIM :**

To write an assembly language program for Seven segment display using LPC2148.

**SOFTWARE/HARDARE required :**

* Keil uvision 5 (software)
* Philips Flash utility (software)
* LPC2148 trainer kit (hardware)

**PROCEDURE :**

* Open the keil software, create new project file.
* Enter LEGACY DEVICE DRIVER – LPC2148.
* Click Yes.
* Open project and select “option for target”.
* Select output and linker then enable CREATE HEX file.
* Add target file to the source file.
* Click the Build target options.
* Check the error
* Open LPC2000 (Philips flash utility software)
* Set the Baud rate 38400
* Set XTAL frequency 12000.
* Import HEX file
* Dump the hex file into the LPC2148 trainer kit.
* Check the output and Note it.

**PROGRAM:**

#include <LPC214X.H>

#define DS3 1<<13 // P0.13

#define DS4 1<<12 // P0.12

#define SEG\_CODE 0xFF<<16 // Segment Data from P0.16 to P0.23

unsigned char const seg\_dat[]={0x3F, 0x6, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x7, 0x7F, 0x67};

void delayms(int n)

{

int i,j;

for(i=0;i<n;i++)

{for(j=0;j<5035;j++) //5035 for 60Mhz \*\* 1007 for 12Mhz

{;}

}

}

/\*-------------------------------------------------------\*/

int main (void)

{

unsigned char count;

PINSEL0 = 0; // Configure Port0 as General Purpose IO => P0.0 to P0.15

PINSEL1 = 0; // Configure Port0 as General Purpose IO => P0.16 to P0.31

IODIR0 = SEG\_CODE | DS3 | DS4; //Configure Segement data & Select signal as output

IOSET0 = SEG\_CODE | DS3 ; //Disable DS3 display

IOCLR0 = DS4; //Enable DS4 Display

count = 0; //Initialize Count

//Display Count value

IOCLR0 = SEG\_CODE;

IOSET0 = seg\_dat[count]<<16;

while(1){

delayms(1000); //1 sec delay

count++; //Increment count

if(count>9) count=0; //Limit 0-9

//Display Count value

IOCLR0 = SEG\_CODE;

IOSET0 = seg\_dat[count]<<16;

   }

}

**RESULT :**

Thus the program has been successfully verified and executed.

**EXP NO: 23**

WRITE AN ASSEMBLY LANGUAGE PROGRAM FOR LCD DISPLAY USING LPC2148.

**AIM :**

To write an assembly language program for LCD display using LPC2148.

**SOFTWARE/HARDARE required :**

* Keil uvision 5 (software)
* Philips Flash utility (software)
* LPC2148 trainer kit (hardware)

**PROCEDURE :**

* Open the keil software, create new project file.
* Enter LEGACY DEVICE DRIVER – LPC2148.
* Click Yes.
* Open project and select “option for target”.
* Select output and linker then enable CREATE HEX file.
* Add target file to the source file.
* Click the Build target options.
* Check the error
* Open LPC2000 (Philips flash utility software)
* Set the Baud rate 38400
* Set XTAL frequency 12000.
* Import HEX file
* Dump the hex file into the LPC2148 trainer kit.
* Check the output and Note it.

**PROGRAM:**

#include <LPC214X.H>

#define LCD\_RS 1<<24 //Port1.24

#define LCD\_RW 1<<16 //Port0.16

#define LCD\_EN 1<<17 //Port0.17

#define LCD\_DATA 0xFF<<16 //Port1.16 to 1.23

#define LCD\_STS 1<<23 //Port1.23

//Function declaration

int main(void);

void lcdini(void);

void lsts(void);

void lcdctl(unsigned char val1);

void lputc(unsigned char lcr);

void put\_s(char \*str);

void delay (unsigned int k);

int main(void)

{

IODIR0 = LCD\_RW | LCD\_EN; //configure LCD r/w & EN as o/p

IODIR1 = LCD\_RS; //Configure RS & DATA as o/p

PINSEL0 = 0; // Configure Port0 as General Purpose IO

PINSEL1 = 0; // Configure Port1 as General Purpose IO

IOCLR0=LCD\_EN; IOCLR1=LCD\_RS; IOSET0=LCD\_RW; /\* DISABLE LCD TEMPORALY\*/

delay(1000); //LCD Power-up Delay

lcdini(); //Initialise LCD

put\_s("TEST MESSAGE"); //Print msg

while(1); //Terminate Program

}

/\*----------------------------------------------------------------------\*/

void lcdini()

{

lcdctl(0x38); /\* Function Set 2 LINE 5 X 8 CHAR\*/

delay(5); /\* Waits for 5 Msec. \*/

lcdctl(0x38); /\* Sends Function Set - AGAIN \*/

delay(5); /\* Waits for 5 Msec. \*/

lcdctl(0x38); /\* Sends Function Set - AGAIN \*/

lsts(); /\* Wait Till BUSY=0 \*/

lcdctl(0x38); /\* Sends Function Set - AGAIN\*/

lsts(); /\* Wait Till BUSY=0 \*/

lcdctl(0x04); /\* Display off \*/

lsts(); /\* Wait Till BUSY=0 \*/

lcdctl(0x01); /\* Clear Display \*/

lsts(); /\* Wait Till BUSY=0 \*/

lcdctl(0x06); /\* Set Entry mode \*/

lsts(); /\* Wait Till BUSY=0 \*/

lcdctl(0x0c); /\* Set Display ON \*/

}

/\*--------------------------------------------\*/

/\*\*\*\*\*\*\* Checks the LCD Status for busy\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void lsts()

{unsigned long int tp1;

IOCLR1=LCD\_RS; IOSET0=LCD\_RW;

do{

IOSET0=LCD\_EN;

tp1 = IOPIN1 & LCD\_STS;

IOCLR0=LCD\_EN;

}while(tp1);

IOCLR0=LCD\_EN;

IOCLR0=LCD\_RW;

}

/\*-----------------------------------------------------------\*/

void lcdctl(unsigned char val1)

{ unsigned long int dat;

dat = ((unsigned long int)val1) << 16;

IODIR1 = LCD\_RS | LCD\_DATA; //Configure RS & DATA as o/p

/\* WRITE COMMAND TO CONTROL REGISTER\*/

IOCLR1 = LCD\_RS;

IOCLR0 = LCD\_RW;

IOCLR1 = LCD\_DATA;

IOSET1 = dat;

IOSET0=LCD\_EN;

IOCLR0=LCD\_EN;

IOSET0=LCD\_RW;

IODIR1 = LCD\_RS; //Configure RS as o/p & DATA as i/p

}

/\*----------------------------------------------------------\*/

/\*\*\*\*\*\*\* Displays a Character in the LCD \*\*\*\*\*\*/

void lputc(unsigned char lcr)

{ unsigned long int dat;

lsts();

dat = ((unsigned long int)lcr)<<16;

IODIR1 = LCD\_RS | LCD\_DATA; //Configure RS & DATA as o/p

IOSET1=LCD\_RS; IOCLR0=LCD\_RW;

IOCLR1 = LCD\_DATA;

IOSET1 = dat;

IOSET0=LCD\_EN;

IOCLR0=LCD\_EN;

IOSET0=LCD\_RW;

IODIR1 = LCD\_RS; //Configure RS as o/p & DATA as i/p

}

/\*----------------------------------------------------------\*/

void put\_s(char \*str)

{

while(\*str)

{ lputc(\*str);

str++;

}

}

/\*----------------------------------------\*/

//Delay Program

//Input - delay value in milli seconds

void delay(unsigned int k)

{

unsigned int i,j;

for (j=0; j<k; j++)

for(i = 0; i<=800; i++);

}

**RESULT :**

Thus the program has been successfully verified and executed.

**EXP NO: 24**

WRITE AN ASSEMBLY LANGUAGE PROGRAM FOR 4\*4 MATRIX KEYPAD USING LPC2148.

**AIM :**

To write an assembly language program for 4\*4 matrix keypad using LPC2148.

**SOFTWARE/HARDARE required :**

* Keil uvision 5 (software)
* Philips Flash utility (software)
* LPC2148 trainer kit (hardware)

**PROCEDURE :**

* Open the keil software, create new project file.
* Enter LEGACY DEVICE DRIVER – LPC2148.
* Click Yes.
* Open project and select “option for target”.
* Select output and linker then enable CREATE HEX file.
* Add target file to the source file.
* Click the Build target options.
* Check the error
* Open LPC2000 (Philips flash utility software)
* Set the Baud rate 38400
* Set XTAL frequency 12000.
* Import HEX file
* Dump the hex file into the LPC2148 trainer kit.
* Check the output and Note it.

**PROGRAM:**

#include "lpc214x.h"

#define SC0 1<< 12

#define SC1 1<< 13

#define SC2 1<< 14

#define SC3 1<< 15

#define RTN 0x0F << 8 //Port 0.8 to 0.11

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

/\*---- I2C-0 Signals ---------------\*/

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

#define SDA0 1<<6 // I2C Serial DATA

#define SCL0 1<<4 // I2C Serial Clock

#define I2EN 1<<6 // I2C Interface Enable

#define STA 1<<5 // Start Flag

#define STO 1<<4 // Stop Flag

#define SI 1<<3 // I2C Interrupt Flag

#define AA 1<<2 // Assert Acknowledge Flag

unsigned char getkey(void);

void i2c\_init(void);

void i2c\_write(unsigned char addr, unsigned char dat);

void delay (unsigned int k);

// 7-Segment data 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

unsigned char seg\_dat[]={0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80, 0x98,

0x88, 0x83, 0xC6, 0xA1, 0x86, 0x8E};

unsigned char scan;

int main(void)

{

unsigned char tp1;

IODIR0 = 0xFFFFF0FF; //Configure Port0.8 to Port0.11 as I/P, Port0.12 to Port0.15 as O/P

i2c\_init(); //Initialise I2C0

PINSEL1 = 0;

scan=0;

do{

tp1=getkey();

//Display 1 in DS1

i2c\_write(0x70, seg\_dat[tp1]); //PCF8574A => 0x70, PCF8574P => 0x40

}while(1);

//return 0;

}

/\*-------------------------------------------------------------\*/

unsigned char getkey()

{ unsigned long int tp1;

do{

/\* Generate Scan Logic \*/

switch(scan)

{ case 0: // 1st ROW

IOCLR0=SC0; IOSET0=SC1; IOSET0=SC2;IOSET0=SC3; //1110

//Check any Key is Pressed

tp1 = IOPIN0 & RTN;

if(tp1!=RTN) //Key is Pressed

{ switch(tp1)

{ case 0x0E00:

return(0);

break;

case 0x0D00:

return(1);

break;

case 0x0B00:

return(2);

break;

case 0x0700:

return(3);

break;

}

}

case 1: // 2nd ROW

IOSET0=SC0; IOCLR0=SC1; IOSET0=SC2;IOSET0=SC3; //1101

//Check any Key is Pressed

tp1 = IOPIN0 & RTN;

if(tp1!=RTN) //Key is Pressed

{ switch(tp1)

{ case 0x0E00:

return(4);

break;

case 0x0D00:

return(5);

break;

case 0x0B00:

return(6);

break;

case 0x0700:

return(7);

break;

}

}

case 2: // 3rd ROW

IOSET0=SC0; IOSET0=SC1; IOCLR0=SC2;IOSET0=SC3; //1011

//Check any Key is Pressed

tp1 = IOPIN0 & RTN;

if(tp1!=RTN) //Key is Pressed

{ switch(tp1)

{ case 0x0E00:

return(8);

break;

case 0x0D00:

return(9);

break;

case 0x0B00:

return(10);

break;

case 0x0700:

return(11);

break;

}

}

case 3: // 4th ROW

IOSET0=SC0; IOSET0=SC1; IOSET0=SC2;IOCLR0=SC3; //0111

//Check any Key is Pressed

tp1 = IOPIN0 & RTN;

if(tp1!=RTN) //Key is Pressed

{ switch(tp1)

{ case 0x0E00:

return(12);

break;

case 0x0D00:

return(13);

break;

case 0x0B00:

return(14);

break;

case 0x0700:

return(15);

break;

}

}

}

delay(2);

if (scan==4) scan = 0;

else scan ++;

}while(1);

}

/\*-------------------------------------------------------------\*/

void i2c\_init()

{

//Enable SDA0 & SCL0 Pins

PINSEL0 = 0 | SDA0 | SCL0;

//Set I2C0 Clock rate to 100Khz

I2C0SCLH = 75; I2C0SCLL = 75;

//Enable I2C

I2C0CONSET = I2EN; // Configure Master Mode

}

/\*------------------------------------------------------\*/

void i2c\_write(unsigned char addr, unsigned char dat)

{

unsigned char sts;

//Transmit Start Condition

I2C0CONSET = STA;

//Wait for tansmit to complete

do{

sts = I2C0CONSET & SI;

}while(sts!=SI);

I2C0CONCLR = STA; //Clear STA

//Transmit Slave Address

I2C0DAT = addr; //Slave Addr.+W

I2C0CONSET = AA; //Ack bit

I2C0CONCLR = SI; //Clear SI flag

//Wait for tansmit to complete

do{

sts = I2C0CONSET & SI;

}while(sts!=SI);

//Transmit data

I2C0DAT = dat; //data

I2C0CONSET = AA; //Ack bit

I2C0CONCLR = SI; //Clear SI flag

//Wait for tansmit to complete

do{

sts = I2C0CONSET & SI;

}while(sts!=SI);

//Stop Bit

I2C0CONSET = STO | AA; //Stop & Ack bit

I2C0CONCLR = SI; //Clear SI flag

delay(2);

I2C0CONCLR = SI; //Clear SI flag

}

/\*------------------------------------------------------\*/

//Delay Program

//Input - delay value in milli seconds

void delay(unsigned int k)

{

unsigned int i,j;

for (j=0; j<k; j++)

for(i = 0; i<=800; i++);

}

**RESULT :**

Thus the program has been successfully verified and executed.