GPIO (General Purpose Input/Output)

Write an embedded C program to turn ON and OFF LEDs connected to P0.11 – P0.4

```
#include <LPC17xx.h>
 unsigned int j;
 unsigned long LED = 0x00000FF0;

int main(void)
{
        SystemInit();
        SystemCoreClockUpdate();

        LPC_PINCON->PINSEL0 =  0x00000000; // P0.15-P0.0 GPIO
        LPC_GPIO0->FIODIR = 0x00000FF0; // P0.11-P0.4 as output

        while(1)
        {
                LPC_GPIO0->FIOSET = LED; // SET P0.11-P0.4
                for(j=0;j<10000;j++); // Delay

                LPC_GPIO0->FIOCLR = LED; // CLEAR P0.11-P0.4
                for(j=0;j<10000;j++); //Delay

        }
}
```

```
LPC_GPIO0->FIOPIN= ~(LPC_GPIO0->FIOPIN & 0x00000FF0);
for(j=0;j<10000;j++); //Delay
```

GPIO (General Purpose Input/Output)

8 bit Johnson Counter on LEDs

Twisted ring counter

Mod -16

2 x 8

P0.11 – P0.4

why

00000000
00000001
00000011
00000111
00001111
00011111
00111111
01111111
11111111
11111110
11111100
11111000
11110000
11100000
11000000
10000000
00000000

ON

OFF

16 states

```
#include <LPC17xx.h>

 unsigned int i,j;
 unsigned long LED = 0x00000010;

 int main(void)
 {
   SystemInit()

   SystemCoreClockUpdate();

   LPC_PINCON->PINSEL0 = 0
                    ;Configure  Port0 pins  P0.4-P0.11 ;as GPIO

   LPC_GPIO0->FIODIR = 0x00000FF0;
                                    ;Configure P0.4-P0.11 as output
```

GPIO (General Purpose Input/Output)

```
while(1)
        {
                LED = 0x00000010; Initial value on LED
                for(i=1;i<9;i++)          //ON the LED's serially
                {
                 LPC_GPIO0->FIOSET = LED;

                 for(j=0;j<10000;j++);
                  LED <<= 1;
                }


                LED = 0x00000010;

                   for(i=1;i<9;i++)        //OFF the LED's serially
                              {
                   LPC_GPIO0->FIOCLR = LED
                    for(j=0;j<10000;j++);
                   LED <<= 1;
                              }

            }
             }
```

GPIO (General Purpose Input/Output)

Write an embedded C program to turn ON LEDs connected to P0.11 – P0.4 when key connected to P2.12 pressed, else turn OFF.

```
#include <LPC17xx.h>
unsigned int j;
unsigned long LED = 0x0000FF00;

int main(void)
{
        SystemInit();
        SystemCoreClockUpdate();

        LPC_PINCON->PINSEL0 =  0x00000000; // P0.15-P0.0 GPIO
        LPC_GPIO0->FIODIR = 0x00000FF0; // P0.11-P0.4 as output

        while(1)
        {
            if ( !(LPC_GPIO2->FIOPIN & 1<<12))

                LPC_GPIO0->FIOSET = LED; // SET P0.11-P0.4
           else

                LPC_GPIO0->FIOCLR = LED; // CLEAR P0.11-P0.4

        }
}
```

Display digits 0-9 in seven segment display.

Seven Segment Display Programming

```
#include<lpc17xx.h>
unsigned char seven_seg[10]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};
unsigned int i,j;
void delay(void);
int main(void)
 {
            SystemInit();
            SystemCoreClockUpdate();

            LPC_PINCON->PINSEL0 = 0     //P0.4 to P0.11 GPIO data lines
            LPC_GPIO0->FIODIR |= 0x00000FF0;       //P0.4 to P0.11 output
            while (1)
              {
                  for(i=0; i<10; i++)
                   {
                    LPC_GPIO0->FIOPIN = seven_seg[i ] << 4;
                    delay();
                   }
              }
    }
void  delay(void)
{
 for(j=0;j<10000;j++);
}
```

Display a number in multiplexed seven segment display (number is 1,2,3,4).

Multiplexed Seven Segment Display Programming – Display a Number

```
#include <LPC17xx.h>
#include <stdio.h>

 #define   FIRST_SEG          0<<23
 #define   SECOND_SEG         1<<23
 #define   THIRD_SEG          2<<23
 #define   FOURTH_SEG         3<<23


  unsigned int dig_count;
  unsigned int digit_value = {0, 4, 3, 2, 1}
  unsigned int select_segment = {0, 0 << 23, 1<<23. 2<<23. 3<<23};
  unsigned char seven_seg[10]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};
  unsigned long int temp1,temp2 ,i=0;

 void Display(void);
 void delay(void);
 int main(void)
 {
            SystemInit();
            SystemCoreClockUpdate();

            LPC_PINCON->PINSEL0 = 0;    //P0.4 to P0.11 GPIO data lines
            LPC_PINCON->PINSEL3 = 0;    //P1.23 to P1.26 GPIO enable lines

            LPC_GPIO0->FIODIR = 0x00000FF0;       //P0.4 to P0.11 output
            LPC_GPIO1->FIODIR = 0x07800000;       //P1.23 to P1.26 output
```

Multiplexed Seven Segment Display Programming – Display a Number

```
        while(1)
        {
                delay();

                dig_count +=1;
                if(dig_count == 0x05)
                        dig_count = 0x01;

                Display();

        } //end of while(1)

}//end of main


 void Display(void)     //To Display on 7-segments
{

        LPC_GPIO1->FIOPIN = select_segment[dig_count];
        LPC_GPIO0->FIOPIN = seven_seg[digit_value[dig_count]] << 4;
        for(i=0;i<500;i++);
        LPC_GPIO0->FIOCLR = 0x00000FF0;
}
void delay(void)
{
for i=0;i<500;i++);
}
```

Seven segment up counter

Multiplexed Seven Segment Display Programming – 4 Digit BCD UP Counter

```
void delay(void)
{
for i=0;i<500;i++);

if(count ==N)
        {
                flag = 0xFF;
                count = 0;
        }
        else count += 1;
}
```

After one second, set Flag

```
if(flag == 0xFF)
        {
                flag = 0;
                digit_value[1] +=1;

                if(digit_value[1] == 0x0A)
                {
                        digit_value[1] = 0;
                        digit_value[2] +=1;

                        if(digit_value[2] == 0x0A)
                        {
                                digit_value[2]= 0;
                                digit_value[3] +=1;

                                if(digit_value[3] == 0x0A)
                                {
                                        digit_value[3] = 0;
                                        digit_value[4] += 1;

                                        if(digit_value[4]== 0x0A)
                                        {
                                                digit_value[4]= 0;
                                        } //end of dig4

                                } //end of dig3

                        } //end of dig2

                } //end of dig1

        } //end of one_sec if
```

For every second update the digits

Hex Keypad Programming

```
#include <LPC17xx.h>
unsigned char col,row,flag,key;
unsigned long var;
void scan();
int main(void)
 {
        SystemInit();
        SystemCoreClockUpdate();

        LPC_PINCON->PINSEL3 &= 0xFFC03FFF; //P1.23 to P1.26 MADE
                                                    //GPIO
        LPC_PINCON->PINSEL4 &= 0xF00FFFFF; //P2.10 t P2.13 made
                                                    //GPIO
        LPC_GPIO2->FIODIR |= 0x00003C00; //made output P2.10 to
                                                    //P2.13 (rows)
        LPC_GPIO1->FIODIR &= 0xF87FFFFF; //made input P1.23 to
                                                    //P1.26 (cols)
        while(1)
        {
         for (row=0;row<4;row++)
         {
          LPC_GPIO2->FIOPIN=1<<(row+10);
                flag=0;
                  scan();
                if (flag==1)
                { key= 4*row+col;
                }
         }}
 }
 void scan()
 {
           var=LPC_GPIO1->FIOPIN &(0xf<<23);
         if(var)
         { flag=0x1;
                var=var>>23;
                switch(var)
                {
                        case 1 : col =0;
                        break;
                        case 2: col=1;
                                break;
                        case 4: col=2;
                                 break;
                        case 8: col=3;
                                break;
                }
         }
 }
```

LCD print a message program:

```
#include <lpc17xx.h>
#define RS_CTRL  0x08000000  //P0.27
#define EN_CTRL  0x10000000  //P0.28
#define DT_CTRL  0x07800000  //P0.23 to P0.26 data lines

 unsigned long int temp1=0, temp2=0,i,j ;
 unsigned char flag1 =0, flag2 =0;
 unsigned char msg[] = {"MITn 01manipal"};

void lcd_write(void);
void port_write(void);
void delay_lcd(unsigned int);
unsigned long int init_command[] = {0x30,0x30,0x30,0x20,0x28,0x0c,0x06,0x01,0xC0};
 int main(void)
 {
       SystemInit();
       SystemCoreClockUpdate();
                LPC_GPIO0->FIODIR = DT_CTRL | RS_CTRL | EN_CTRL;
                 flag1 =0;
        for (i=0; i<9;i++)
                 {
           temp1 = init_command[i];
           lcd_write();
                  }
                 flag1 =1;
       i =0;
       while (msg[i++] != '\0')
                  {
                   temp1 = msg[i];
                   lcd_write();
                   i+= 1;
                  }
       while(1);
 }


  void lcd_write(void)
                {
                 flag2 =  (flag1 == 1) ? 0 :((temp1 == 0x30) || (temp1 == 0x20)) ? 1 : 0;
                 temp2 = temp1 & 0xf0;//move data (26-8+1) times : 26 - HN place, 4 - Bits
       temp2 = temp2 << 19;//data lines from 23 to 26
       port_write();
                 if (!flag2)
                   {
          temp2 = temp1 & 0x0f; //26-4+1
          temp2 = temp2 << 23;
          port_write();
                  }
                }


 void port_write(void)
 {
       LPC_GPIO0->FIOPIN = 0;
       LPC_GPIO0->FIOPIN = temp2;
         if (flag1 == 0)
                 LPC_GPIO0->FIOCLR = RS_CTRL;
         else
                       LPC_GPIO0->FIOSET = RS_CTRL;

       LPC_GPIO0->FIOSET = EN_CTRL;
       delay_lcd(25);
       LPC_GPIO0->FIOCLR = EN_CTRL;
                 delay_lcd(30000);

  }

void delay_lcd(unsigned int r1)
 {
       unsigned int r;
       for(r=0;r<r1;r++);

    return;
 }
```

Display message in two lines in LCD display program

```
#include <lpc17xx.h>
#define RS  27 //P0.27
#define EN  28  //P0.28
#define DT  23  //P0.23 to P0.26 data lines

 unsigned long int temp1=0, temp2=0,i,j ;
 unsigned char flag1 =0, flag2 =0;
 unsigned char msg[] = {" Department of ICT MIT manipal"}; //As message is written in codes they are stored in ASCII values

void lcd_write(void);
void port_write(void);
void delay_lcd(unsigned int);
unsigned long int init_command[] = {0x30,0x30,0x30,0x20,0x28,0x0c,0x06,0x01,0x80};
int main(void)
 {
        SystemInit();
        SystemCoreClockUpdate();
                  LPC_GPIO0->FIODIR = 1<<RS|1<<EN|0XF<<DT;  //used to make all pins output
                   flag1 =0;    // flag1 = 0 all are command and flag1 = 1 all are data
         for (i=0; i<9;i++)
                    {
            temp1 = init_command[i];
            lcd_write();
                    }
                   flag1 =1;
        i =0;
        while (msg[i] != '\0')
                     {
                      temp1 = msg[i]; // char by char
                      lcd_write();
                      i+= 1;
                      if(i==16)  //check for 1 charactres in first line
                       {
                         flag1=0;  //if yes
                         temp1=0xc0; //configure second line in command register
                         lcd_write();
                         flag1=1;
                       }
                     }
        while(1);
 }
```

```
 void lcd_write(void)
               {
                flag2 =  (flag1 == 1) ? 0 :((temp1 == 0x30) || (temp1 == 0x20)) ? 1 : 0;
                temp2 = temp1 & 0xf0;//move data (26-8+1) times : 26 - HN place, 4 - Bits to extract MSB and then LSB as nedd to send 4 bit at a time
                                                                  temp2=temp2>>4;
      temp2 = temp2 << DT;//data lines from 23 to 26
      port_write();
                if (!flag2)
                  {
         temp2 = temp1 & 0x0f; //26-4+1
         temp2 = temp2 << DT;
         port_write();
                 }
               }


void port_write(void)
{
      LPC_GPIO0->FIOPIN = 0;
      LPC_GPIO0->FIOPIN = temp2;
        if (flag1 == 0)
               LPC_GPIO0->FIOCLR = 1<<RS;
        else
                     LPC_GPIO0->FIOSET = 1<<RS;

      LPC_GPIO0->FIOSET = 1<<EN;      //this and below 3 lines are used go give pulse for enable and wait for some time interval
      delay_lcd(25);
      LPC_GPIO0->FIOCLR = 1<<EN;
                delay_lcd(30000);                           // 3 mili sec highest delay

 }

void delay_lcd(unsigned int r1)
{
      unsigned int r;
      for(r=0;r<r1;r++);

   return;
}
```

Generate square wave with period 2 second

```
#include<stdio.h>
#include<LPC17xx.h>
void delay(void)
{
        LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
        LPC_TIM0->EMR = 0X20;//Set match bit upon match
        LPC_TIM0->PR = 1000; /
        LPC_TIM0->MR0 = 3000;         //for 1 second
        LPC_TIM0->MCR = 0x00000004;    // stop PC and TC on MR0
        LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
        while ( !(LPC_TIM0->EMR & 0x01)); // wait until match

        return;
        }


int main(void)
{
        LPC_GPIO0->FIODIR=0x00000004;

                while(1)
        {
                                                LPC_GPIO0->FIOPIN=~(LPC_GPIO0->FIOPIN & 0x00000004);
                                                Delay();
                LPC_GPIO0->FIOSET=0x4;
                delay();

                LPC_GPIO0->FIOCLR=0x4;
                delay();
        }
}
```

Toggle LED connected to P0.2 every second.i.e generate square wave with period 2 seconds

Generate square wave when duty cycle is given

Generate square wave of period 2 seconds with 75% duty cycle on P0.2

```
#include<stdio.h>
#include<LPC17xx.h>
void delay(void)
{
        //LPC_SC->PCONP |= (1<<1);   //powers the T0
        LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
        LPC_TIM0->EMR = 0X20; //Set EM0 upon match
        LPC_TIM0->PR = 2999;
        LPC_TIM0->MCR = 0x00000004;    // stop PC and TC on MR0
        LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
        while ( !(LPC_TIM0->EMR & 0x01));// Wait until EM0 is set
        return;
        }
int main(void)
{
        LPC_GPIO0->FIODIR=0x00000004;
                while(1)
        {
                LPC_GPIO0->FIOPIN=0x00000004;
                LPC_TIM0->MR0 = 1500; //For 1.5 seconds
                delay();
                LPC_GPIO0->FIOPIN=0x00000000;
                LPC_TIM0->MR0 = 500; //For 0.5 seconds
                delay();
}
}
```

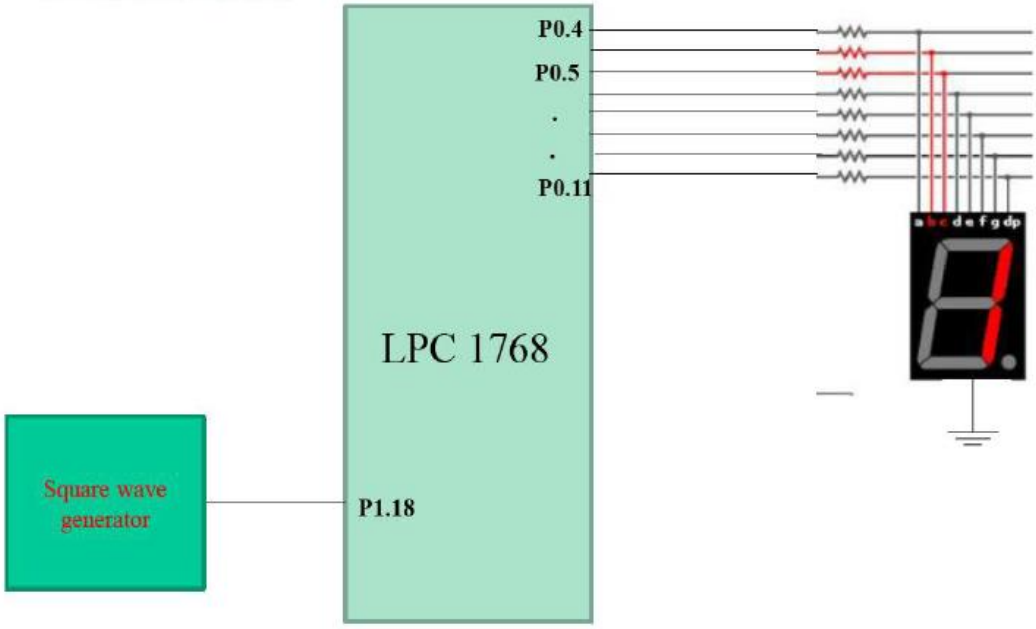
Square waveform on MAT 0.0 output line by taking EM0 on the output pin.

```
#include<stdio.h>
#include<LPC17xx.h>
void delay(void)
{
        LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
        LPC_TIM0->CTCR = 0x00000000;
        LPC_TIM0->EMR = 0X30;//Toggle bit upon match
        LPC_TIM0->PR = 0; //
        LPC_TIM0->MR0 = 3000000;       //
        LPC_TIM0->MCR = 0x00000002;    // Reset TC
        LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
        return;
        }
int main(void)
{
            LPC_PINCON->PINSEL3 |= (3<<24);//Get EM0 on MAT0.0 (P1.28) line
            delay();
            while(1);
}
```

MAT 1.1(P1.25) toggles whenever count reaches 3. CAP 1.0 (P1.18) is counter clock. i.e Divide the frequency of the square waveform input at P1.18 by a factor of 8 on P1.25

```
#include<stdio.h>
#include<LPC17xx.h>
void init_timer1(void)
{
        LPC_PINCON->PINSEL3 |=(3<<18 | 3<<4);//  MAT 1.1(P1.25) and CAP 1.0 (P1.18)
        LPC_TIM1->TCR=2;//Reset Counter1
        LPC_TIM1->CTCR = 0x2; // Counter at -ve edge of CAP1.0
        LPC_TIM1->MR1=0x03; //To count 4 clock pulses
        LPC_TIM1->MCR=0x10;//Clear TC upon Match1
        LPC_TIM1->EMR=0xC0;//Toggle EM1 upon Match
        LPC_TIM1->TCR=1;//Start Counter1
}
int  main(void)
{
        init_timer1();
        while(1);
}
```

Assume that output of a square wave generator (Frequency < 10Hz) is connected to P1.18 (CAP1.0, Function-3) , write aprogram to display the frequency of this square waveform on the seven segment connected to P011–P0.4.




```
#include<stdio.h>
#include<LPC17xx.h>
unsigned char seven_seg[10]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};
void delay(void)
{
        LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
        LPC_TIM0->EMR = 0X20;//Set match bit upon match
        LPC_TIM0->PR = 3000; //for 1 ms
        LPC_TIM0->MR0 = 1000;          //for 1 second
        LPC_TIM0->MCR = 0x00000004;    // stop PC and TC on MR0
        LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
        while ( !(LPC_TIM0->EMR & 0x01)); // wait until match
        }
void init_counter1(void)
{
        LPC_PINCON->PINSEL3 = (3<<4);//  cap 1.0 (P1.18)
        LPC_TIM1->CTCR = 0x01; // Counter at +ve edge of CAP1.0
}
```

```
int  main(void)
{
            LPC_PINCON->PINSEL0 = 0          //P0.4 to P0.11 GPIO data lines
            LPC_GPIO0->FIODIR = 0x00000FF0;            //P0.4 to P0.11 output
            init_counter1();
while(1)
        {
        LPC_TIM1->TCR=2;//Reset Counter1
        LPC_TIM1->TCR=1;//Start Counter1
        Delay(); // wait for 1 second
        LPC_GPIO0->FIOPIN = seven_seg[LPC_TIM1->TC ] << 4; Counter1 on the
seven segment
        }

}
```

Capture TC into TC when +ve edge is applied to CAP0.0 (P1.26) or CAP0.1(P1.27)

```
#include<stdio.h>
#include<LPC17xx.h>
void delay(void)
{
            //LPC_SC->PCONP |= (1<<1);   //powers the T0
            LPC_TIM0->CCR=9;//capture on positive edge
            LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
            LPC_TIM0->EMR = 0X20;//Set match bit upon match
            LPC_TIM0->PR = 3000; //for 1 ms
            LPC_TIM0->MR0 = 1000;              //for 1 second
            LPC_TIM0->MCR = 0x00000004;    // stop PC and TC on MR0
            LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
            while ( !(LPC_TIM0->EMR & 0x01)); // wait until match
            return;
            }

int main(void)
{
            LPC_GPIO0->FIODIR=0x00000004;
            LPC_PINCON->PINSEL3 |=(3<<20) | (3<<22);//select cap 0.0 and cap 0.1
                                    while(1)
            {
                        LPC_GPIO0->FIOPIN=~(LPC_GPIO0->FIOPIN & 0x00000004);//toggle p0.2
                        delay();

            }
}
```

Toggle LED connected to p0.2 every second while displaying the status of switch connected to P1.0 on the LED connected to P2.0

```
#include<stdio.h>
#include<LPC17xx.h>
unsigned int ticks=0,x;
void TIMER0_IRQHandler(void)
{
LPC_TIM0->IR = 1;
            ticks++;
            if(ticks==1000)
            {
                        ticks=0;
                        LPC_GPIO0->FIOPIN=~(LPC_GPIO0->FIOPIN & 0x00000004);
            }
                        }
void init_timer0(void)
{
            LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
            LPC_TIM0->CTCR =0x00;//Timer
            LPC_TIM0->MR0 = 2999; //  For 1ms
            LPC_TIM0->EMR = 0X00;//Do nothing for EM0
            LPC_TIM0->PR = 0;
            LPC_TIM0->MCR = 0x00000003;  //Reset TC upon Match-0 and generate INTR
            LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable

            return;


int main(void)
{
            LPC_GPIO0->FIODIR=0x00000004;
            LPC_GPIO2->FIODIR=0x00000001;
            init_timer0();
            NVIC_EnableIRQ(TIMER0_IRQn);//timer 0 intr enabled in NVIC
                        while(1)
                        {
                                    LPC_GPIO2->FIOPIN=(LPC_GPIO1->FIOPIN & 0x01) ;
                        }

}
```

Toggle P0.2 whenever counter value reaches 3. I. e for every 4 edges using counter interrupt.

```
#include<stdio.h>
#include<LPC17xx.h>
void TIMER0_IRQHandler(void)
{
                        LPC_TIM0->IR = 1; //Clear the interrupt
                        LPC_GPIO0->FIOPIN=~(LPC_GPIO0->FIOPIN & 0x00000004);

}
void init_timer0(void)
{       LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
        LPC_TIM0->CTCR =0x05; // Counter at +ve edge of CAP0.1
        LPC_TIM0->MR0 = 3;
        LPC_TIM0->EMR = 0X00;
        LPC_TIM0->PR = 0;
        LPC_TIM0->MCR = 0x00000003;
        LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
        return;
        }

int main(void)
{
        LPC_GPIO0->FIODIR=0x00000004;
        LPC_PINCON->PINSEL3 |=((3<<22)|(3<<24));
        init_timer0();
        NVIC_EnableIRQ(TIMER0_IRQn);
        while(1);
}
```

Timer interrupt for rectangular waveform generation (1.5 second HIGH and 0.5 second LOW)

```
include<stdio.h>
#include<LPC17xx.h>
unsigned char flag=1;
void TIMER0_IRQHandler(void)
{
LPC_TIM0->IR = 1;

if(flag)
        {
                flag=0;
                LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
                LPC_GPIO0->FIOCLR=0x00000004;
                LPC_TIM0->MR0 = 500;
                LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
        }
        else
                {
                flag=1;
                LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
                LPC_GPIO0->FIOSET=0x00000004;
                LPC_TIM0->MR0 = 1500;
                LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
                }

}
```

```
void init_timer0(void)
{
        LPC_TIM0->TCR = 0x00000002;    // Timer0 Reset
        LPC_TIM0->CTCR =0x00;
        LPC_TIM0->MR0 = 1500;
        LPC_TIM0->EMR = 0X00;
        LPC_TIM0->PR = 3000;
        LPC_TIM0->MCR = 0x00000005;
        LPC_TIM0->TCR = 0x00000001;    // Timer0 Enable
        LPC_GPIO0->FIOSET=0x00000004;
        return;
}
int main(void)
{
        LPC_GPIO0->FIODIR=0x00000004;
        init_timer0();
        NVIC_EnableIRQ(TIMER0_IRQn);
        while(1);
}
```

ADC software mode for 2-channel concurrent conversion

```
#include<LPC17xx.h>
 #include<stdio.h>
int main(void)
{
        unsigned long temp4, temp5;
        unsigned int i;

        SystemInit();
        SystemCoreClockUpdate();
        LPC_PINCON->PINSEL3 = (3<<28) | (3<<30);          //P1.30 as AD0.4 and P1.31 as AD0.5
        LPC_ADC->ADINTEN = 0;
        while(1)
        {
                LPC_ADC->ADCR = (1<<4)|(1<<21)|(1<<24);//;          //ADC0.4, start conversion and operational
                while(((temp4=LPC_ADC->ADDR4) & (1<<31)) == 0);   //wait till 'done' bit is 1, indicates conversion complete
                temp4 = LPC_ADC->ADDR4;
                temp4 >>= 4;
                temp4  &= 0x00000FFF;                             //12 bit ADC

                LPC_ADC->ADCR = (1<<5)|(1<<21)|(1<<24);//          //ADC0.5, start conversion and operational
                for(i=0;i<2000;i++);                              //delay for conversion
                while(((temp5=LPC_ADC->ADDR5) & (1<<31)) == 0);   //wait till 'done' bit is 1, indicates conversion complete
                temp5 = LPC_ADC->ADDR5;
                temp5 >>= 4;
                temp5 &= 0x00000FFF;                              //12 bit ADC
//Now you can use temp4 and temp5 for further processing based on your requirement
        }
}
```

ADC burst mode for 2-channel concurrent conversion

```
#include<LPC17xx.h>
#include<stdio.h>
int main(void)

{                       SystemInit();
                        SystemCoreClockUpdate();
                        LPC_PINCON->PINSEL3 =(3<<28)|(3<<30);       //P1.30 as AD0.4 and P1.31 as AD0.5
                        LPC_ADC->ADCR = (1<<4) | (1<<5)|(1<<16) | (1<<21); //Enable CH 4 and 5 for BURST mode with ADC power ON
                        LPC_ADC->ADINTEN =(1<<4)|(1<<5); // Enable  DONE for INTR
                        NVIC_EnableIRQ(ADC_IRQn);
                        while(1);
}
void ADC_IRQHandler(void)
{
        int channel,temp,result;
        channel=(LPC_ADC->ADGDR >>24) & 0x07;
        result= ( LPC_ADC->ADGDR >>4) & 0xFFF;
        if(channel == 4)
        {
         temp4 = ( LPC_ADC->ADDR4 >>4) & 0xFFF ; //Read to Clear Done flag
         }
        else if(channel == 5)
        {
         temp5 = ( LPC_ADC->ADDR5 >>4) & 0xFFF ; //Read to Clear Done flag
//Now you can use temp4 and temp5 for further processing based on your requirement
 }
```

Input Analog voltage and display its digital equivalent on LCD

```
include<LPC17xx.h>
 #include<stdio.h>
 #define      Ref_Vtg                    3.300
 #define      Full_Scale   0xFFF//12 bit ADC
int main(void)
{
              unsigned long adc_temp;
              unsigned int i;
              float in_vtg;
              unsigned char vtg[7], dval[7];
              unsigned char Msg3[] = {"ANALOG IP:"};
              unsigned char Msg4[] = {"ADC OUTPUT:"};
              SystemInit();
              SystemCoreClockUpdate();
              lcd_init();//Initialize LCD
               LPC_PINCON->PINSEL3 |= 3<<30;//P1.31 as AD0.5
              LPC_SC->PCONP |= (1<<12);//enable the peripheral ADC
              flag1=0;//Command
              temp1 = 0x80;//Cursor at beginning of first line
              lcd_write();
              flag1=1;//Data
               i =0;
               while (Msg3[i++] != '\0')
                 {
                  temp1 = Msg3[i];
                  lcd_write();//Send data bytes
                 }
```

```
flag1=0; //Command
    temp1 = 0xC0;//Cursor at beginning of second line
  lcd_write();
  flag1=1;
    i =0;
  while (Msg4[i++] != '\0')
   {
    temp1 = Msg4[i];
    lcd_write();//Send data bytes
    }
while(1)
{
        LPC_ADC->ADCR = (1<<5)|(1<<21)|(1<<24);//ADC0.5, start conversion and operational
          while(((adc_temp=LPC_ADC->ADGDR) & (1<<31)) == 0);
        adc_temp = LPC_ADC->ADGDR;
        adc_temp >>= 4;
        adc_temp &= 0x00000FFF;          //12 bit ADC
        in_vtg = (((float)adc_temp * (float)Ref_Vtg))/((float)Full_Scale);          //calculating input analog voltage
        sprintf(vtg,"%3.2fV",in_vtg);          //convert the readings into string to display on LCD
        sprintf(dval,"%x",adc_temp);
        flag1=0;;
        temp1 = 0x8A;
        lcd_write();
        flag1=1;
            i =0;
          while (vtg[i++] != '\0')
           {
            temp1 = vtg[i];
            lcd_write();//Send data bytes
            }
```

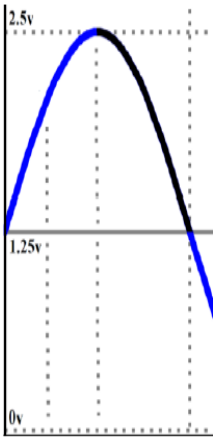
```
flag1=0;
temp1 = 0xCB;
lcd_write();
flag1=1;
  i =0;
 while (dval[i++] != '\0')
   {
    temp1 = dval[i];
    lcd_write();//Send data bytes
    }
   for(i=0;i<7;i++)
   vtg[i] = dval[i] = 0;
}
}
```

Generate a sawtooth waveform with peak to peak amplitude 3.3v at P0.26.

```
#include <lpc17xx.h>
 void DAC_Init(void);
 int main (void)
 {
            unsigned int m,i;
            SystemInit();
            SystemCoreClockUpdate();
            LPC_PINCON->PINSEL1 = 2<<20;// Analog Input P0.26
            LPC_DAC->DACCNTVAL = 0x0050; // DAC Counter for Double Buffering
            LPC_DAC->DACCTRL = (0x1<<1)|(0x1<<2); //Double buffering
            while ( 1 )
            {
                        LPC_DAC->DACR = (i << 6) ;
                        i++;
                        if ( i == 0x400 )                    //Maximum value is 0x3FF in 10 bit DAC
                        {
                        i = 0;
                        }
            }
 }
```

Generate a sinewave with peak to peak amplitude 2.5v at P0.26. (i.e. Vout = 1.25+1.25 sin θ)

```
#include <lpc17xx.h>
 void DAC_Init(void);
sinetable[]= { 388, 582,723, 776, 723,582, 388,194,52,0,52,194};
 int main (void)
 {
            unsigned int m,i;
            SystemInit();
            SystemCoreClockUpdate();
            LPC_PINCON->PINSEL1 = 2<<20;// Analog Input P0.26
            while ( 1 )
            {
                        for(i=0; i < 12; i++) // Assuming samples separated by 30 degrees
                         {
                          LPC_DAC->DACR = (sinetable[i % 12]<< 6) ;
                          delay(); // Call timer delay based on period. If period is 10 ms. Delay is  (10ms/12)
            }
 }
```



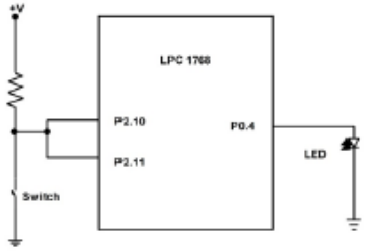
INSEM QUESTIONS:

Write an embedded C program using timer interrupt to generate a square waveform of frequency 100 kHz and duty cycle 75% on P2.3 using TIMER-0 (PCLK = 3 MHz)

```
#include<stdio.h>
#include<LPC17xx.h>
unsigned char flag=1;
void TIMER0_IRQHandler(void)
{

        if(flag)
        {
                flag=0;
                LPC_TIM0->TCR = 0x00000002;     // Timer0 Reset
                LPC_GPIO2->FIOCLR=0x00000008;
                LPC_TIM0->MR0 = 7;
                LPC_TIM0->TCR = 0x00000001;     // Timer0 Enable
        }
        else
                {
                flag=1;
                LPC_TIM0->TCR = 0x00000002;     // Timer0 Reset
                LPC_GPIO2->FIOSET=0x00000008;
                LPC_TIM0->MR0 = 22;
                LPC_TIM0->TCR = 0x00000001;     // Timer0 Enable
        }
                LPC_TIM0->IR = 1;
        }
void init_timer0(void)
{
        LPC_TIM0->TCR = 0x00000002;     // Timer0 Reset
        LPC_TIM0->CTCR =0x00;
        LPC_TIM0->MR0 = 22;
        LPC_TIM0->EMR = 0X30;
        LPC_TIM0->PR = 0;
        LPC_TIM0->MCR = 0x00000005;
        LPC_TIM0->TCR = 0x00000001;     // Timer0 Enable
        LPC_GPIO2->FIOSET=0x00000008;
        return;
        }


int main(void)
{
        LPC_GPIO2->FIODIR=0x00000008;
        init_timer0();
        NVIC_EnableIRQ(TIMER0_IRQn);
        while(1);}
```

Main-0.5, Timer init 1.5, ISS - 1

3. For the connections shown below, write an embedded C program using GPIO interrupt to turn ON the LED whenever the switch is pressed and turn OFF the LED whenever the switch is released.




```
#include<LPC17xx.h>

unsigned int x,y;
void EINT3_IRQHandler (void)
{
}
x = (LPC_GPIOINT->IO2IntStatR)>>10;
if (x== 0x01)
LPC_GPIO0->FIOCLR = 0x04;
y = (LPC_GPIOINT->IO2IntStatF)>>10;
if (y== 0x02)
LPC_GPIO0->FIOSET = 0x04;

LPC_GPIOINT->IO2IntClr = 0x03<<10;
}
void main(void)
{
LPC_PINCON -> PINSEL4 = (1<<20) | (1<<22) ;
LPC_GPIO0 ->FIODIR = 0x10;
LPC_GPIOINT->IO2IntEnR=0x01<<10; // P2.10 raising edge
LPC_GPIOINT->IO2IntEnF=0x01<<11; // P2.11 falling edge
NVIC_EnableIRQ(EINT3_IRQn);
while(1);
}
```

Functions – 1.5 each

5. Assume that output of a square wave generator (Frequency range 0-9 Hz) is connected to P2.12 (EINT-2, Function-1) input. Write an embedded C program using external hardware interrupt to display the frequency of this square waveform on the seven-segment display connected to P0.7-P0.0.

```
#include<LPC17xx.h>
unsigned int count =0;
unsigned char
seven_seg[10]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};
void EINT2_IRQHandler(void)
{
count++;
}
void delay(void)
{
        LPC_TIM0->TCR = 0x00000002;     // Timer0 Reset
        LPC_TIM0->EMR = 0X20;//Set match bit upon match
        LPC_TIM0->PR = 3000; //for 1 ms
        LPC_TIM0->MR0 = 1000;     //for 1 second
        LPC_TIM0->MCR = 0x00000004;     // stop PC and TC on MR0
        LPC_TIM0->TCR = 0x00000001;     // Timer0 Enable
        while ( !(LPC_TIM0->EMR & 0x01)); // wait until match

}
int  main(void)
{
LPC_GPIO0->FIODIR = 0x00000FF;
LPC_PINCON -> PINSEL4 = (1<<24);
LPC_SC ->EXTMODE =0x04;
LPC_SC ->EXTPOLAR = 0x04;
NVIC_EnableIRQ(EINT2_IRQn);
while(1)
        {
        LPC_TIM1->TCR=2;//Reset Counter1
         Delay(); // wait for 1 second
        LPC_GPIO0->FIOPIN = seven_seg[count ] << 4; Counter1 on the seven
segment
count=0;
        }
```

4

EINT ISS -1, Other functions – 1.5 each

6. With a neat diagram, explain how a 3-digit multiplexed 7 segment display can be interfaced to microcontroller. Write an embedded C program to display 123 on this

```
#include<LPC17xx.h>
#include<stdio.h>

#define FIRST_SEG 0<<23
#define SECOND_SEG 1<<23
#define THIRD_SEG 2<<23

unsigned int dig count;
unsigned int digit_value = (0, 3, 2, 1)
unsigned int select_segment = (0, 0 << 23. 1<<23. 2<<23);
unsigned char seven_seg[3]={0x06, 0x5B, 0x4F};
 unsigned long int temp1, temp2 ,i=0;

void Display(void);
void delay(void);
int main(void) ;
Systemlnit();
SystemCoreClockUpdate();

LPC _PINCON->PINSELO = 0; P0.4 to P0.11 GPIO data lines
```

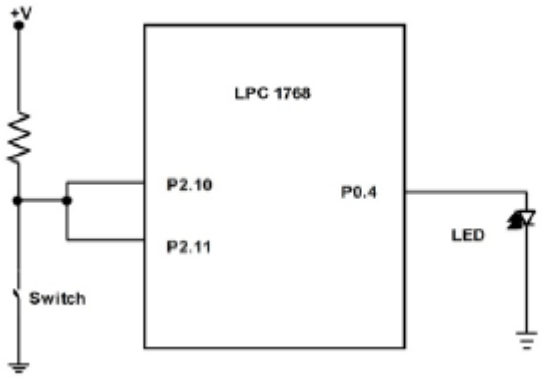
```
LPC PINCON•>PINSEL3 = 0;    P1.23 to P1.26 GPIO enable lines
LPC_GP100•>FIODIR = Ox00000FFO; P0.4 to P0.11 output
LPC_GP101->FIODIR = 0x07800000; HP1.23 to P1.26 output

while(1)
{
delay();
dig_count +=1;
if(dig_count == 0x04)
dig_count = Ox01:
Display()
} //end of while(1)
}//end of main

void Display(void) //To Display on 7•segmenb
{
LPC_GP101•>F1OPIN = select_segment[dig_count];
LPC_GP100•>FIOPIN = seven_seg_digit_value[dig_count]] << 4;
for(i=0;i<500;i++);
LPC_GP100->FIOCLR = Ox00000FFO;
}
void delay(void)
{
 for 1=0;i<500;i++);
}

void delay(void)
{
for i=0;i<500;i++);
if(count ==N)
{
flag = OxFF;
count = 0;
}
else count += 1;


  if(flag == 0XFF)
  {
   Flag = 0;
  Digit_value[1] ==3;
  Digit_value[2] ==2;
  Digit_value[3] ==1;

  }}
```

2. Assume that output of a square wave generator is connected to P1.29(CAP 1.1, Function-3). Write an embedded C program to generate a square waveform on the P1.25 (MAT 1.1, Function-3) whose frequency is one fourth of the frequency of the square wave input at P1.29.

```
#include<stdio.h>
#include<LPC17xx.h>
void init_timer1(void)
{

        LPC_PINCON->PINSEL3 |=(3<<18 | 3<<26);//  MAT 1.1(P1.25) and CAP 1.1 (P1.29)
        LPC_TIM1->TCR=2;//Reset Counter1
        LPC_TIM1->CTCR = 0x5; // Counter at +ve edge of CAP1.1
        LPC_TIM1->MR1=0x01; //To count 2 clock pulses in half cycle
        LPC_TIM1->MCR=0x10;//Clear TC upon Match1
        LPC_TIM1->EMR=0xC0;//Toggle EM1 upon Match
        LPC_TIM1->TCR=1;//Start Counter1
}
int  main(void)
{
        init_timer1();
        while(1);
}
```

(MR Value -1, Program 2)

3. For the connections shown below, write an embedded C program using external hardware interrupt to turn ON the LED whenever the switch is pressed and turn OFF the LED whenever the switch is released.



```
#include<LPC17xx.h>

void EINT0_IRQHandler(void)
{
LPC_GPIO0 ->FIOSET = 0x10;
LPC_SC -> EXTINT = 0x01;
}

void EINT1_IRQHandler(void)
{
LPC_GPIO0 ->FIOCLR = 0x10;
LPC_SC -> EXTINT = 0x2;

}


void main(void)
{
LPC_PINCON -> PINSEL4 = (1<<20) | (1<<22) ;
LPC_GPIO0 ->FIODIR = 0x10;
LPC_SC ->EXTMODE =0x03;
LPC_SC ->EXTPOLAR = 0x02;


NVIC_EnableIRQ(EINT0_IRQn);
NVIC_EnableIRQ(EINT1_IRQn);

while(1);}
```

Main -2, Functions – 1each

5. Assume that output of a square wave generator is connected to P2.12 input. Write an embedded C program using GPIO interrupt to generate a square waveform at P0.4 whose frequency is 0.125 times the frequency of the input square waveform at P2.12.

```
#include<LPC17xx.h>
unsigned int x;
void EINT3_IRQHandler (void)
{
}
x ++
if (x==4) // for frequency 1/8
{
x=0;
LPC_GPIO0->FIOPIN = ~ (LPC_GPIO0->FIOPIN & 1<<4);
}
LPC_GPIOINT->IO2IntClr = 1<<12;
}
void main(void)
{
LPC_GPIO0 ->FIODIR = 1<<4;
LPC_GPIOINT->IO2IntEnR=1<<12; // P2.12 raising edge
NVIC_EnableIRQ(EINT3_IRQn);
while(1);
}
```

Functions – 2each

4

6. With a neat diagram, explain how a 3x3 matrix keyboard can be interfaced to microcontroller. Write an embedded C program to display the keycode of the key pressed on the LEDs connected to P0.2-P0.0

4

Interfacing Diagram: 1 Mark
LPC Pin configuration : 0.5 Mark
Polling the key pressed with identification : 1+1.5 Mark

```
  Main(void)
{
Initialization for P1.0 to P1.5 (Key Pad:GPIO);
Initialization for P0.0 to P0.2 (LED's GPIO);

Setting Direction Port1 : Input (Key Pad);
Setting Direction Port0 : output (LED);

Int flag, row;

While(1)
{
 For row = 0; row<3; row++)
{
Making each row high one after other;
Flag = 0 ;
Scan();
If(flag = 1)
Break;
}

If(flag = 1)
{
Keypress = 3*row + col;
LEDdisplay(Keypress);
}}}


Voidscan()
{
X = LPC_GPIO1→ FIOPIN;
X = x&07;
If(x!=0)
{
Flag=1;

Using switch case finding the column;
}

Void LEDdisplay(Keypress)
{
Based on keypressvalues

Using if statement or switch and LPC_GPIO→FIOSET enable the LED's

}
```

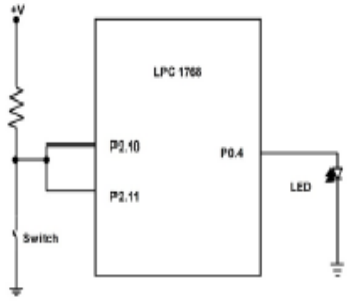
2. Write an embedded C program using timer interrupt to generate a square waveform of frequency 1 kHz and duty cycle 67% on P2.6 using TIMER-1 (PCLK = 6 MHz)

3

```
#include<stdio.h>
#include<LPC17xx.h>
unsigned char flag=1;
void TIMER1_IRQHandler(void)
{
        if(flag)
        {
                flag=0;
                LPC_TIM1->TCR = 0x00000002;     // Timer1 Reset
                LPC_GPIO2->FIOCLR=1<<6;
                LPC_TIM1->MR0 = 1980;
                LPC_TIM1->TCR = 0x00000001;     // Timer1 Enable

        }
        else
                {
                flag=1;
                LPC_TIM1->TCR = 0x00000002;     // Timer1 Reset
                LPC_GPIO2->FIOSET=1<<6;
                LPC_TIM1->MR0 = 4020;
                LPC_TIM1->TCR = 0x00000001;     // Timer1 Enable
        }
                LPC_TIM1->IR = 1;
        }
void init_timer1(void)
{

        LPC_TIM1->TCR = 0x00000002;     // Timer1 Reset
        LPC_TIM1->CTCR =0x00;
        LPC_TIM1->MR0 = 4020;
        LPC_TIM1->EMR = 0X30;
        LPC_TIM1->PR = 0;
        LPC_TIM1->MCR = 0x00000005;
        LPC_TIM1->TCR = 0x00000001;     // Timer1 Enable
        LPC_GPIO2->FIOSET=1<<6;
        return;
        }


int main(void)
{
        LPC_GPIO2->FIODIR=1<<6;
        init_timer1();
        NVIC_EnableIRQ(TIMER1_IRQn);
        while(1);}
```

Main-0.5, Timer init 1.5, ISS - 1

3. For the connections shown below, write an embedded C program using GPIO interrupt to turn ON the LED after pressing and releasing the Switch FOUR times.



3

```
#include<LPC17xx.h>
unsigned int x, count1, count2, y;
void EINT3_IRQHandler (void)
{
}
x = (LPC_GPIOINT->IO2IntStatR)>>10;
y = (LPC_GPIOINT->IO2IntStatF)>>10;
if (x== 0x01)
count1++;

if (y== 0x02)
count2++;
if (count1==4 && count2 ==4)
LPC_GPIO0->SET = 0x04;

LPC_GPIOINT->IO2IntClr = 0x03<<10;
}
void main(void)
{
LPC_PINCON -> PINSEL4 = (1<<20) | (1<<22) ;
LPC_GPIO0 ->FIODIR = 0x10;
LPC_GPIOINT->IO2IntEnR=0x01<<10; // P2.10 raising edge
LPC_GPIOINT->IO2IntEnF=0x01<<11; // P2.11 falling edge
NVIC_EnableIRQ(EINT3_IRQn);
while(1);
}
```

5. Assume that output of a square wave generator with 50% duty cycle is connected to P2.12 (EINT2, Function-1). Write an embedded C program using external hardware interrupt to generate a square waveform on P0.4 with frequency one eighth of the frequency of the input square waveform at P2.12 and duty cycle 75%.

```
#include<LPC17xx.h>
unsigned int count =0;
void EINT2_IRQHandler(void)
{
count++;
if (count==6)
LPC_GPIO0->FIOCLR = 1<<4;
if(count==8)
{
LPC_GPIO0->FIOSET = 1<<4;
count=0;
}
LPC_SC ->EXTINT = 0x04;
}
int  main(void)
{
LPC_GPIO0->FIODIR = 1<<4;
LPC_PINCON -> PINSEL4 = (1<<24);
LPC_SC ->EXTMODE =0x04;
LPC_SC ->EXTPOLAR = 0x04;
NVIC_EnableIRQ(EINT2_IRQn);
LPC_GPIO0->FIOSET = 1<<4;
while(1);
        }
```

EINT ISS -2, Main-2

4

6. With a neat diagram, explain how a 16x2 LCD can be interfaced to the microcontroller. Write an embedded C program to display the message "Best Wishes" 4

```
#include <lpc17xx.h>
#define RS  27 //P0.27
#define EN  28  //P0.28
#define DT  23  //P0.23 to P0.26 data lines

 unsigned long int temp1=0, temp2=0,i,j ;
 unsigned char flag1 =0, flag2 =0;
 unsigned char msg[] = {" Best Wishes "}; //As message is written in codes they are stored in
ASCII values

void lcd_write(void);
void port_write(void);
void delay_lcd(unsigned int);
unsigned long int init_command[] = {0x30,0x30,0x30,0x20,0x28,0x0c,0x06,0x01,0x80};
int main(void)
 {
        SystemInit();
        SystemCoreClockUpdate();
           LPC_GPIO0->FIODIR  =  1<<RS|1<<EN|0XF<<DT;   //used  to  make  all  pins
output
            flag1 =0;     // flag1 = 0 all are command and flag1 = 1 all are data
         for (i=0; i<9;i++)
             {
           temp1 = init_command[i];
           lcd_write();
             }
            flag1 =1;
        i =0;
        while (msg[i] != '\0')
              {
              temp1 = msg[i]; // char by char
              lcd_write();
                    i+= 1;

if(i==16)  //check for 1 charactres in first line
                                                                                {

         flag1=0;  //if yes

         temp1=0xc0; //configure second line in command register

         lcd_write();

         flag1=1;
                                                                                }
              }
        while(1);
```

```
}

 void lcd_write(void)
            {
                flag2 =  (flag1 == 1) ? 0 :((temp1 == 0x30) || (temp1 == 0x20)) ? 1 : 0;
              temp2 = temp1 & 0xf0;//move data (26-8+1) times : 26 - HN place, 4 - Bits to
extract MSB and then LSB as nedd to send 4 bit at a time
                                                                    temp2=temp2>>4;
          temp2 = temp2 << DT;//data lines from 23 to 26
          port_write();
             if (!flag2)
               {
            temp2 = temp1 & 0x0f; //26-4+1
            temp2 = temp2 << DT;
            port_write();
              }
             }


 void port_write(void)
 {
          LPC_GPIO0->FIOPIN = 0;
          LPC_GPIO0->FIOPIN = temp2;
        if (flag1 == 0)
             LPC_GPIO0->FIOCLR = 1<<RS;
        else
                   LPC_GPIO0->FIOSET = 1<<RS;

          LPC_GPIO0->FIOSET = 1<<EN;        //this and below 3 lines are used  give pulse
for enable and wait for some time interval
          delay_lcd(25);
          LPC_GPIO0->FIOCLR = 1<<EN;
             delay_lcd(30000);                              // 3 ms  highest delay

 }

void delay_lcd(unsigned int r1)
 {
          unsigned int r;
          for(r=0;r<r1;r++);

   return;
 }
```