SOLVED

Q. WAP to display message on LCD

CODE

```
#include <lpc17xx.h>

void lcd_init(void);
void write(int, int);
void delay_lcd(unsigned int);
void lcd_comdata(int, int);
void clear_ports(void);
void lcd_puts(unsigned char *);

int main(void) {
   unsigned char Msg1[7] = {"MIT"};
   unsigned char Msg2[19] = {"CSE"};

   SystemInit();
   SystemCoreClockUpdate();

   lcd_init();
   lcd_comdata(0x80, 0);
   delay_lcd(80000);
```

```
    lcd_puts(&Msg1[0]);

    lcd_comdata(0xC0, 0);
    delay_lcd(80000);
    lcd_puts(&Msg2[0]);

    return 0;
}

// LCD initialization
void lcd_init() {
    /* Ports initialized as GPIO */
    LPC_PINCON->PINSEL1 &= 0xFC003FFF; // P0.23 to P0.28

    /* Setting the directions as output */
    LPC_GPIO0->FIODIR |= 0x0F << 23 | 1 << 27 | 1 << 28;

    clear_ports();
    delay_lcd(3200);

    lcd_comdata(0x33, 0);
    delay_lcd(30000);

    lcd_comdata(0x32, 0);
```

```
    delay_lcd(30000);

    lcd_comdata(0x28, 0); // function set
    delay_lcd(30000);

    lcd_comdata(0x0c, 0); // display on cursor off
    delay_lcd(80000);

    lcd_comdata(0x06, 0); // entry mode set increment cursor right
    delay_lcd(80000);

    lcd_comdata(0x01, 0); // display clear
    delay_lcd(10000);
    return;
}

void lcd_comdata(int temp1, int type) {
    int temp2 = temp1 & 0xf0; // move data (26-8+1) times: 26 - HN place, 4 - Bits
    temp2 = temp2 << 19; // data lines from 23 to 26
    write(temp2, type);

    temp2 = temp1 & 0x0f; // 26-4+1
    temp2 = temp2 << 23;
    write(temp2, type);
```

```
    delay_lcd(1000);

    return;
}

void write(int temp2, int type) {
    // write to command/data reg
    clear_ports();
    LPC_GPIO0->FIOPIN = temp2; // Assign the value to the data lines

    if (type == 0) {
        LPC_GPIO0->FIOCLR = 1 << 27; // clear bit RS for Command
    } else {
        LPC_GPIO0->FIOSET = 1 << 27; // set bit RS for Data
    }

    LPC_GPIO0->FIOSET = 1 << 28; // EN = 1
    delay_lcd(25);
    LPC_GPIO0->FIOCLR = 1 << 28; // EN = 0
    return;
}

void delay_lcd(unsigned int r1) {
    unsigned int r;
```

```
    for (r = 0; r < r1; r++);
    return;
}

void clear_ports(void) {
    /* Clearing the lines at power on */
    LPC_GPIO0->FIOCLR = 0x0F << 23; // Clearing data lines
    LPC_GPIO0->FIOCLR = 1 << 27;    // Clearing RS line
    LPC_GPIO0->FIOCLR = 1 << 28;    // Clearing Enable line
    return;
}

void lcd_puts(unsigned char *buf1) {
    unsigned int i = 0;
    unsigned int temp3;

    while (buf1[i] != '\0') {
        temp3 = buf1[i];
        lcd_comdata(temp3, 1);
        i++;

        if (i == 16) {
            lcd_comdata(0xc0, 0);
        }
```

```
    }
    return;
}
```

OUTPUT

We can see the required text ‘MIT CSE’ displayed on the board.

Q1 Simulate DIE tossing on LCD. Hint: Program reads the external interrupt using the key SW2. A random number between 0-6 should be displayed on the LCD upon keypress.

CODE

```
#include<LPC17xx.h>

#include "lcdfn.h"

#include<stdlib.h>

unsigned char msg1[13]="Dice Result";

unsigned char key;

unsigned long int temp1 = 0;

int main()

{

        unsigned char k;

        lcd_init();

        temp1 = 0x80;

        lcd_comdata(temp1, 0);

        delay_lcd(20000);
```

```
lcd_puts(&msg1[0]);

while(1)

{

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

        {

                k = (rand()%6)+1;

                k=k+0x30;

                temp1 = 0xc0;

                lcd_comdata(temp1, 0);

                delay_lcd(20000);

                lcd_puts(&k);

        }
```

```
        }

}
```

Header file

```
#include <lpc17xx.h>

void lcd_init(void);

void write(int, int);

void delay_lcd(unsigned int);

void lcd_comdata(int, int);

void clear_ports(void);

void lcd_puts(unsigned char *);

void lcd_init() {

        /*Ports initialized as GPIO */

        LPC_PINCON->PINSEL1 &= 0xFC003FFF; //P0.23 to P0.28
```

```
/*Setting the directions as output */

LPC_GPIO0->FIODIR |= 0x0F<<23 | 1<<27 | 1<<28;

clear_ports();

delay_lcd(3200);

lcd_comdata(0x33, 0);

delay_lcd(30000);

lcd_comdata(0x32, 0);

delay_lcd(30000);

lcd_comdata(0x28, 0); //function set

delay_lcd(30000);

lcd_comdata(0x0c, 0);//display on cursor off

delay_lcd(800);

lcd_comdata(0x06, 0); //entry mode set increment cursor right
```

```
        delay_lcd(800);

        lcd_comdata(0x01, 0); //display clear

        delay_lcd(10000);

        return;

}

void lcd_comdata(int temp1, int type) {

        int temp2 = temp1 & 0xf0; //move data (26-8+1) times : 26 - HN place, 4 -
Bits

        temp2 = temp2 << 19; //data lines from 23 to 26

        write(temp2, type);

        temp2 = temp1 & 0x0f; //26-4+1

        temp2 = temp2 << 23;

        write(temp2, type);
```

```
	delay_lcd(1000);

	return;

}

void write(int temp2, int type) { /*write to command/data reg */

	clear_ports();

	LPC_GPIO0->FIOPIN = temp2; // Assign the value to the data lines

	if(type==0)

		LPC_GPIO0->FIOCLR = 1<<27; // clear bit RS for Command

	else

		LPC_GPIO0->FIOSET = 1<<27; // set bit RS for Data

	LPC_GPIO0->FIOSET = 1<<28; // EN=1

	delay_lcd(25);

	LPC_GPIO0->FIOCLR = 1<<28; // EN =0
```

```
	return;
}
void delay_lcd(unsigned int r1)
{
unsigned int r;
for(r=0;r<r1;r++);
return;
}
void clear_ports(void) { /* Clearing the lines at power on */
	LPC_GPIO0->FIOCLR = 0x0F<<23; //Clearing data lines
	LPC_GPIO0->FIOCLR = 1<<27; //Clearing RS line
	LPC_GPIO0->FIOCLR = 1<<28; //Clearing Enable line
```

```
        return;

}

void lcd_puts(unsigned char *buf1) {

        unsigned int i=0;

        unsigned int temp3;

        while(buf1[i]!='\0') {

                temp3 = buf1[i];

                lcd_comdata(temp3, 1);

                i++;

                if(i==16)

                        lcd_comdata(0xc0, 0);

        }

        return;
```

}

OUTPUT

Saw it on LPC board.

NOTE

Connect CNB to CNB1

Connect CNAD to CND

Connect CND1 to CNA