EE2016 LAB
EXPERIMENT 6

Abhinav I S
EE23B002

---

1 Introduction

In this experiment, using C programming, we will read the status of the DIP switch and use the LEDs to display the state (ON or OFF) of each of the 8 DIP switches.

Stepper motor control using LPC2148-based ARM development board.

2 Objectives

1. Read the status (binary position) of the DIP (Dual Inline Package) switch and use the LEDs (8 LEDs are provided) to display the state (ON or OFF) of each of the 8 DIP switches.

2. Stepper motor control using LPC2148 based ARM development board

3 Procedures

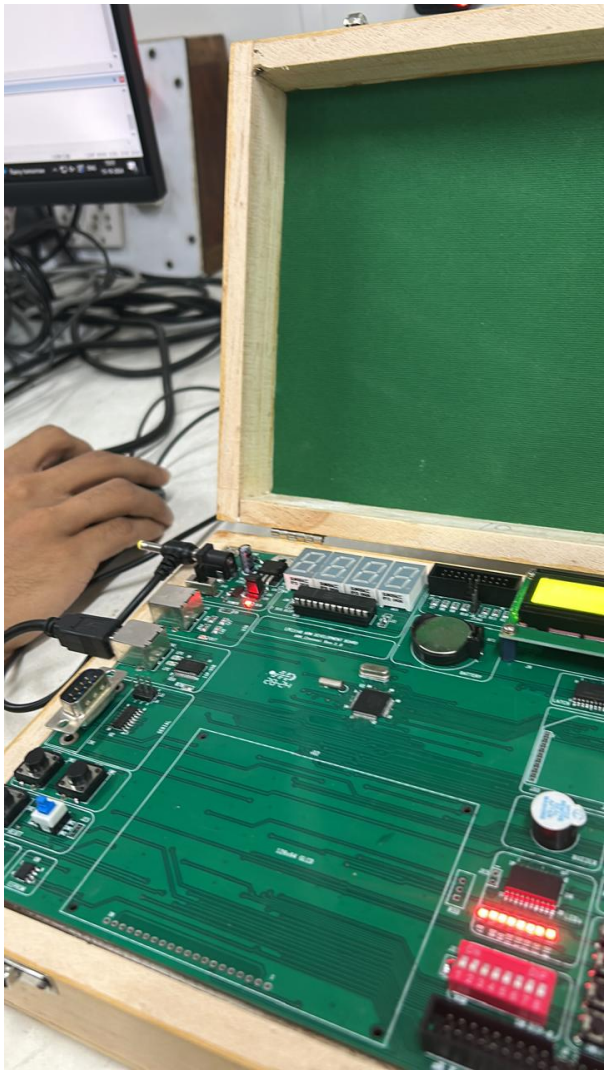
3.1 Only LEDs

This section demonstrates how to programmatically control LEDs on the development board. The code initializes each LED pin as an output and sets a specific value to display a pattern of ON/OFF states on the LEDs.

```
1 //LPC2148- C program to display a value on the LEDs
  #include "LPC214x.h"                          /* LPC21xx
     definitions */
3
   #define LED_IOPIN       IO0PIN
5 #define BIT(x)  (1 << x)

7 #define LED_D0  (1 << 10)       // P0.10
  #define LED_D1  (1 << 11)       // P0.11
9 #define LED_D2  (1 << 12)       // P0.12
  #define LED_D3  (1 << 13)       // P0.13
```

```
11
#define LED_D4  (1 << 15)        // P0.15
13 #define LED_D5  (1 << 16)        // P0.16
#define LED_D6  (1 << 17)        // P0.17
15 #define LED_D7  (1 << 18)        // P0.18
#define LED_DATA_MASK            ((unsigned long)((LED_D7 |
   LED_D6 | LED_D5 | LED_D4 | LED_D3 | LED_D2 | LED_D1 |
   LED_D0)))
17    #ifndef LED_DRIVER_OUTPUT_EN
#define LED_DRIVER_OUTPUT_EN (1 << 5)   // P0.5
19 #endif
#define LED1_ON     LED_IOPIN |= (unsigned long)(LED_D0);
       // LED1 ON
21 #define LED2_ON     LED_IOPIN |= (unsigned long)(LED_D1);
       // LED2 ON
#define LED3_ON     LED_IOPIN |= (unsigned long)(LED_D2);
       // LED3 ON
23 #define LED4_ON     LED_IOPIN |= (unsigned long)(LED_D3);
       // LED4 ON
#define LED5_ON     LED_IOPIN |= (unsigned long)(LED_D4);
       // LED5 ON
25 #define LED6_ON     LED_IOPIN |= (unsigned long)(LED_D5);
       // LED6 ON
#define LED7_ON     LED_IOPIN |= (unsigned long)(LED_D6);
       // LED7 ON
27 #define LED8_ON     LED_IOPIN |= (unsigned long)(LED_D7);
       // LED8 ON

29 int main (void)
{
31
  IO0DIR |= LED_DATA_MASK;          // GPIO Direction
     control -> pin is output
33    IO0DIR |= LED_DRIVER_OUTPUT_EN;     // GPIO Direction
       control -> pin is output
    IO0CLR |= LED_DRIVER_OUTPUT_EN;
35


37    while(1)
    {
39        int value=0x86;
```

```
41      if(value & BIT(0)) LED8_ON;
        if(value & BIT(1)) LED7_ON;
43      if(value & BIT(2)) LED6_ON;
        if(value & BIT(3)) LED5_ON;
45
        if(value & BIT(4)) LED4_ON;
47      if(value & BIT(5)) LED3_ON;
        if(value & BIT(6)) LED2_ON;
49      if(value & BIT(7)) LED1_ON;
        }
51
   return 0;
53 }
```



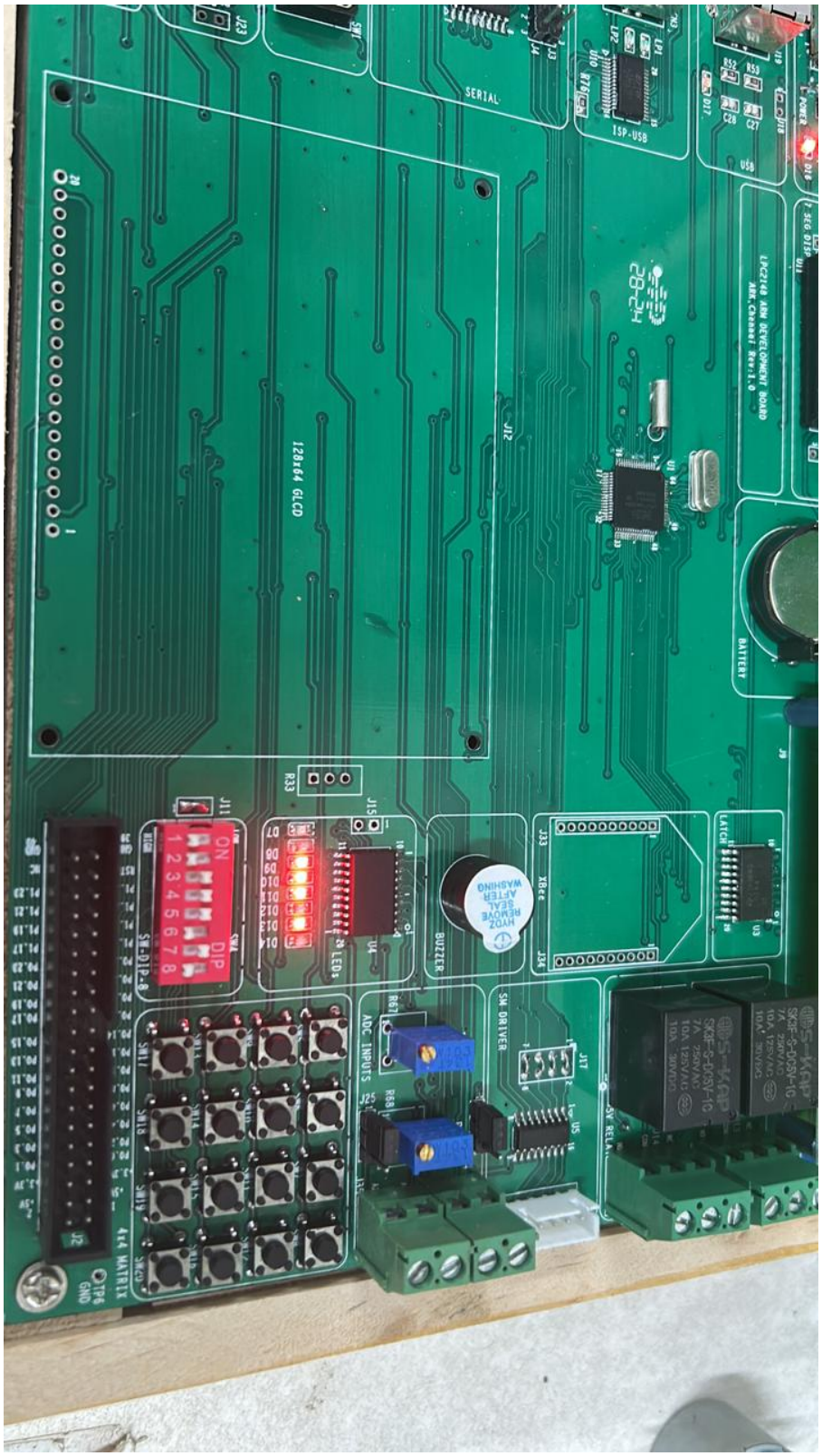
3.2 DIP Switch and LED

This code segment reads the state of each DIP switch and updates the corresponding LED to reflect whether the switch is ON or OFF. This setup helps understand input handling and conditional output control on the microcontroller.

```
2
  DEVELOPED BY:- ARK DEVELOPER
4 WHAT PROGRAM DO:- LED ON/OFF based on 8-PIN DIP SWITCH
     STATUS
6
  #include "LPC214x.H"                          /* LPC214x
     definitions */
8 #include "led.h"
  #include "delay.h"
10

12 #define DIP_SW_D0 (1 << 16)      // P0.16
  #define DIP_SW_D1 (1 << 17)      // P0.17
14 #define DIP_SW_D2 (1 << 18)      // P0.18
  #define DIP_SW_D3 (1 << 19)      // P0.19
16
  #define DIP_SW_D4 (1 << 22)      // P0.22
18 #define DIP_SW_D5 (1 << 23)      // P0.23
  #define DIP_SW_D6 (1 << 24)      // P0.24
20 #define DIP_SW_D7 (1 << 25)      // P0.25

22 #define DIP_SW_DIR       IO1DIR
  #define DIP_SW_PIN       IO1PIN
24
  #define DIP_SW_DATA_MASK    (DIP_SW_D7 | DIP_SW_D6 |
     DIP_SW_D5 | DIP_SW_D4 | DIP_SW_D3 | DIP_SW_D2 | DIP_SW_D1
     | DIP_SW_D0)
26

28 void set_dipswitch_port_input( void )
  {
30     DIP_SW_DIR &= ~(DIP_SW_DATA_MASK);
  }
32
  unsigned long read_dip_switch( void )
```

```
{
    return DIP_SW_PIN;
}


int main (void)
{
    unsigned long sw_status;

    set_led_port_output();
    set_dipswitch_port_input();

    while(1)
    {
        sw_status = read_dip_switch();
/*
        if(sw_status & DIP_SW_D0){ LED1_OFF;} else{ LED1_ON
           ;}
        if(sw_status & DIP_SW_D1){ LED2_OFF;} else{ LED2_ON
           ;}
        if(sw_status & DIP_SW_D2){ LED3_OFF;} else{ LED3_ON
           ;}
        if(sw_status & DIP_SW_D3){ LED4_OFF;} else{ LED4_ON
           ;}

        if(sw_status & DIP_SW_D4){ LED5_OFF;} else{ LED5_ON
           ;}
        if(sw_status & DIP_SW_D5){ LED6_OFF;} else{ LED6_ON
           ;}
        if(sw_status & DIP_SW_D6){ LED7_OFF;} else{ LED7_ON
           ;}
        if(sw_status & DIP_SW_D7){ LED8_OFF;} else{ LED8_ON
           ;}
*/

        if(sw_status & DIP_SW_D0){ LED1_OFF;} else{ LED1_ON
           ;}
              delay_mSec(10);
        if(sw_status & DIP_SW_D1){ LED2_OFF;} else{ LED2_ON
           ;}
              delay_mSec(10);
        if(sw_status & DIP_SW_D2){ LED3_OFF;} else{ LED3_ON
```

```
            ;}
66                  delay_mSec(10);
        if(sw_status & DIP_SW_D3){ LED4_OFF;} else{ LED4_ON
            ;}
68                  delay_mSec(10);

70      if(sw_status & DIP_SW_D4){ LED5_OFF;} else{ LED5_ON
            ;}
                    delay_mSec(10);
72      if(sw_status & DIP_SW_D5){ LED6_OFF;} else{ LED6_ON
            ;}
                    delay_mSec(10);
74      if(sw_status & DIP_SW_D6){ LED7_OFF;} else{ LED7_ON
            ;}
                    delay_mSec(10);
76      if(sw_status & DIP_SW_D7){ LED8_OFF;} else{ LED8_ON
            ;}
                    delay_mSec(10);
78
        delay_mSec(100);
80  }

82 //    return 0;
   }
```


SERIAL
ISP-USB
128x64 GLCD
LPC2148 ARM DEVELOPMENT BOARD
ARK Channel Rev:1.0
BATTERY
BUZZER
ADC INPUTS
SW DRIVER
LATCH
SE-DIP-8
4x4 MATRIX
TP6
GND
ON
DIP

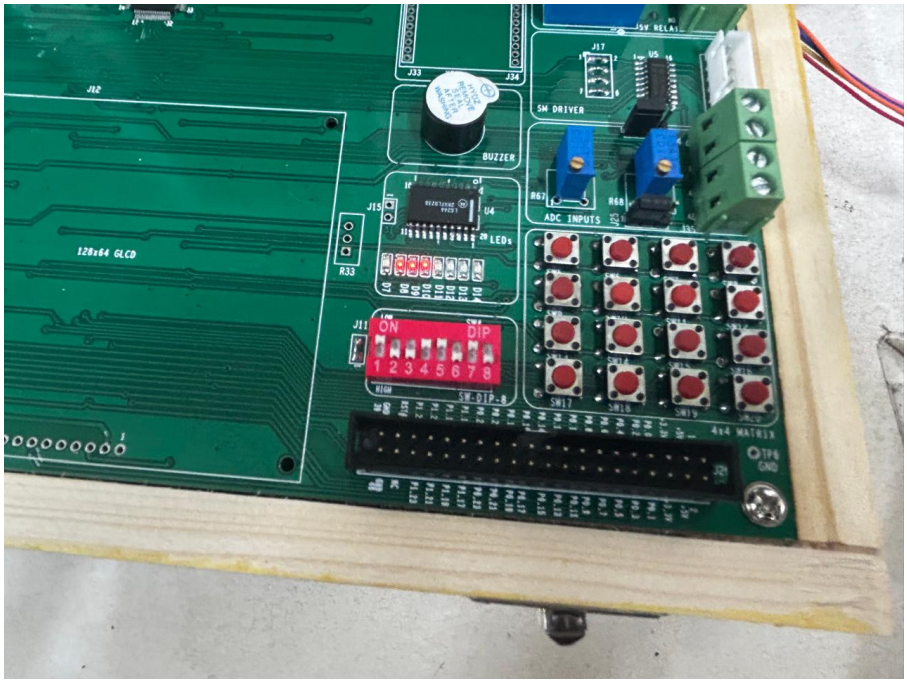
4 Sum

Here, two 4-bit inputs from DIP switches are read, and their product is calculated and displayed using LEDs. This section illustrates binary arithmetic operations and the display of results through GPIO.

```
    #include "LPC214x.H"                        /* LPC214x
       definitions */
#include "led.h"
#include "delay.h"


#define DIP_SW_D0 (1 << 16)     // P0.16
#define DIP_SW_D1 (1 << 17)     // P0.17
#define DIP_SW_D2 (1 << 18)     // P0.18
#define DIP_SW_D3 (1 << 19)     // P0.19

#define DIP_SW_D4 (1 << 22)     // P0.22
#define DIP_SW_D5 (1 << 23)     // P0.23
#define DIP_SW_D6 (1 << 24)     // P0.24
#define DIP_SW_D7 (1 << 25)     // P0.25

#define DIP_SW_DIR      IO1DIR
#define DIP_SW_PIN      IO1PIN

#define DIP_SW_DATA_MASK    (DIP_SW_D7 | DIP_SW_D6 |
   DIP_SW_D5 | DIP_SW_D4 | DIP_SW_D3 | DIP_SW_D2 | DIP_SW_D1
   | DIP_SW_D0)

void set_dipswitch_port_input( void )
{
    DIP_SW_DIR &= ~(DIP_SW_DATA_MASK);
}

unsigned long read_dip_switch( void )
{
    return DIP_SW_PIN;
}

int main (void)
{
    unsigned long sw_status;
```

```
35      set_led_port_output();
        set_dipswitch_port_input();
37
        while(1)
39      {
            sw_status = read_dip_switch();
41          unsigned long input1 = (~(sw_status >> 16))&0x0F;
            unsigned long input2 = (~(sw_status >> 22))&0x0F;
43
            unsigned long sum = input1 * input2;
45


47          sw_status = sum << 16;
            sw_status = ~sw_status;
49


51          if(sw_status & DIP_SW_D0){ LED1_OFF;} else{ LED1_ON
               ;}
                    delay_mSec(10);
53          if(sw_status & DIP_SW_D1){ LED2_OFF;} else{ LED2_ON
               ;}z
                    delay_mSec(10);
55          if(sw_status & DIP_SW_D2){ LED3_OFF;} else{ LED3_ON
               ;}
                    delay_mSec(10);
57          if(sw_status & DIP_SW_D3){ LED4_OFF;} else{ LED4_ON
               ;}
                    delay_mSec(10);
59
            if(sw_status & DIP_SW_D4){ LED5_OFF;} else{ LED5_ON
               ;}
61                  delay_mSec(10);
            if(sw_status & DIP_SW_D5){ LED6_OFF;} else{ LED6_ON
               ;}
63                  delay_mSec(10);
            if(sw_status & DIP_SW_D6){ LED7_OFF;} else{ LED7_ON
               ;}
65                  delay_mSec(10);
            if(sw_status & DIP_SW_D7){ LED8_OFF;} else{ LED8_ON
               ;}
67                  delay_mSec(10);
```

```
69          delay_mSec(100);
    }
71
//     return 0;
73  }
```



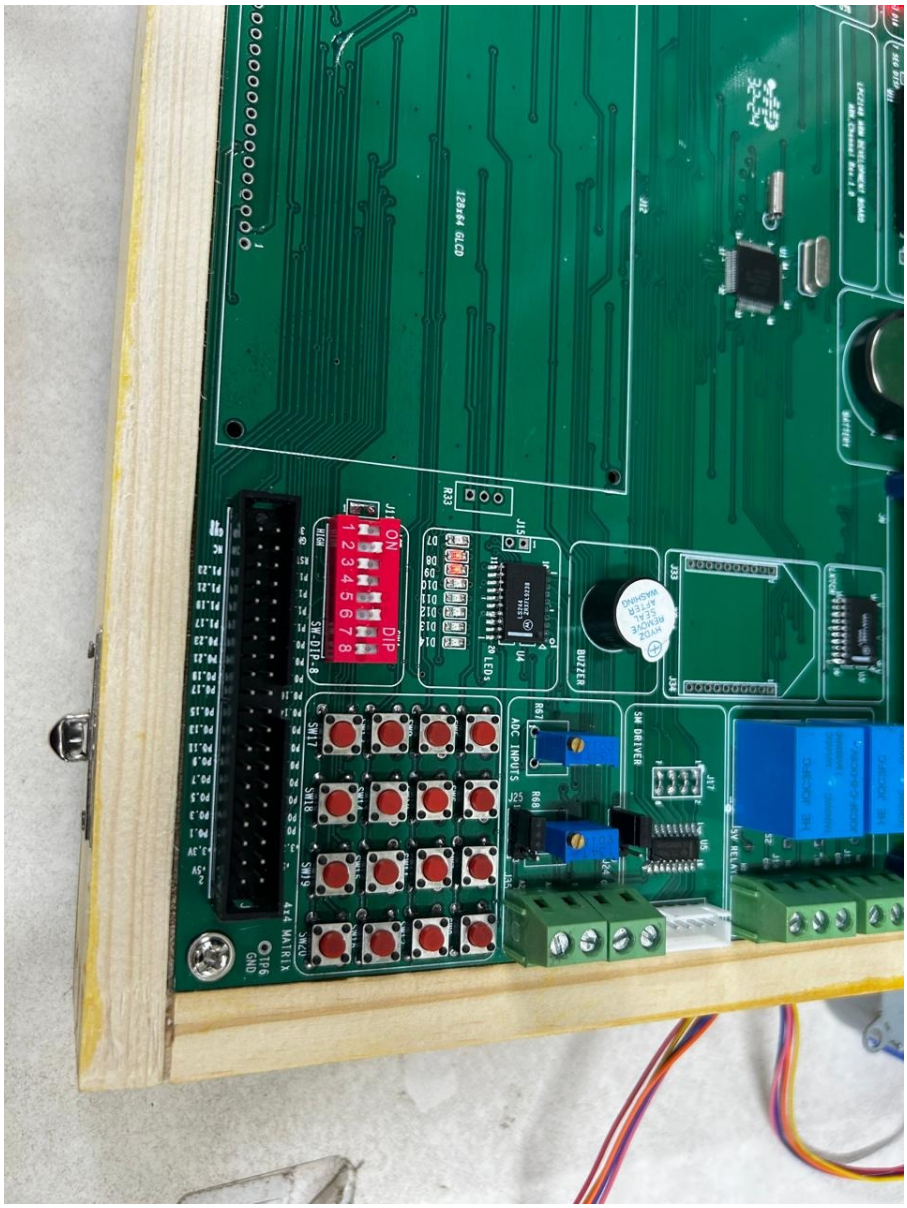
5 Product

In this section, we perform multiplication on two 4-bit binary inputs obtained from DIP switches and display the result using LEDs. The multiplication result helps demonstrate logical shifting and bit masking in C.

```
2
4   #include "LPC214x.H"                          /* LPC214x
       definitions */
    #include "led.h"
6   #include "delay.h"

8
    #define DIP_SW_D0 (1 << 16)      // P0.16
10  #define DIP_SW_D1 (1 << 17)      // P0.17
```

```
#define DIP_SW_D2 (1 << 18)     // P0.18
#define DIP_SW_D3 (1 << 19)     // P0.19

#define DIP_SW_D4 (1 << 22)     // P0.22
#define DIP_SW_D5 (1 << 23)     // P0.23
#define DIP_SW_D6 (1 << 24)     // P0.24
#define DIP_SW_D7 (1 << 25)     // P0.25

#define DIP_SW_DIR      IO1DIR
#define DIP_SW_PIN      IO1PIN

#define DIP_SW_DATA_MASK    (DIP_SW_D7 | DIP_SW_D6 |
   DIP_SW_D5 | DIP_SW_D4 | DIP_SW_D3 | DIP_SW_D2 | DIP_SW_D1
   | DIP_SW_D0)

void set_dipswitch_port_input( void )
{
    DIP_SW_DIR &= ~(DIP_SW_DATA_MASK);
}

unsigned long read_dip_switch( void )
{
    return DIP_SW_PIN;
}

int main (void)
{
    unsigned long sw_status;

    set_led_port_output();
    set_dipswitch_port_input();

    while(1)
    {
        sw_status = read_dip_switch();
        unsigned long input1 = (~(sw_status >> 16))&0x0F;
        unsigned long input2 = (~(sw_status >> 22))&0x0F;

        unsigned long sum = input1 * input2;

        unsigned long temp = (sum & 0xF0);
        temp = temp << 2;
```

```
        sum |= temp;
52
        sw_status = sum << 16;
54      sw_status = ~sw_status;

56
        if(sw_status & DIP_SW_D0){ LED1_OFF;} else{ LED1_ON
           ;}
58              delay_mSec(10);
        if(sw_status & DIP_SW_D1){ LED2_OFF;} else{ LED2_ON
           ;}z
60              delay_mSec(10);
        if(sw_status & DIP_SW_D2){ LED3_OFF;} else{ LED3_ON
           ;}
62              delay_mSec(10);
        if(sw_status & DIP_SW_D3){ LED4_OFF;} else{ LED4_ON
           ;}
64              delay_mSec(10);

66      if(sw_status & DIP_SW_D4){ LED5_OFF;} else{ LED5_ON
           ;}
                delay_mSec(10);
68      if(sw_status & DIP_SW_D5){ LED6_OFF;} else{ LED6_ON
           ;}
                delay_mSec(10);
70      if(sw_status & DIP_SW_D6){ LED7_OFF;} else{ LED7_ON
           ;}
                delay_mSec(10);
72      if(sw_status & DIP_SW_D7){ LED8_OFF;} else{ LED8_ON
           ;}
                delay_mSec(10);
74
        delay_mSec(100);
76    }

78 //    return 0;
   }
```



5.1 Motor

This final section demonstrates how to control a stepper motor by setting appropriate step sequences. Each step sequence is sent to the motor pins to control the motor's movement and rotation direction.

```
    /*5V Stepper Motor

Connector J16 connect with stepper motor  as per below
   mentioned configuration
```

```
Pin - 1  : BLUE
Pin - 2  : PINK
Pin - 3  : YELLOW
Pin - 4  : ORANGE
Pin - 5  : Red (Motor Vcc)

Motor Pins:
P0.4 to P0.7
*/

#include <LPC214x.h>                          /* LPC21xx definitions */

void delay_mSec(int);

int main (void)
{
int i;
unsigned char steps[4] = {0x09, 0x0c, 0x06, 0x03};  // standard step sequence for stepper motor
//unsigned char steps[4] = {0x03, 0x06, 0x0c, 0x09}; // all that i have done here is to reverse the order of step codes.
signed char x = 0;


PINSEL0 = 0x0; // Pin function Select -> P0.0 to P0.15 -> GPIO Port
IO0DIR |= 0xF0;                         // Set stepper motor pins as output in IO0 port
delay_mSec(10);

while(1)
{
for(i=0;i<2500;i++)
{
IO0PIN = (steps[x++] << 4); //send the 4-bit step value to stepper motor lines connected to IO0 port
if(x > 3)
x = 0;

delay_mSec(2);
```

```
}
}
    return 0;
}

void delay_mSec(int dCnt) // pr_note:~dCnt mSec
{
  int j=0,i=0;
  while(dCnt--)
  {
 for(j=0;j<30;j++)
 {
   /* At 60Mhz, the below loop introduces
   delay of 10 us */
   for(i=0;i<10;i++);
 }
  }
}
```

We just need to change the order of elements in the array for getting reverse rotation. Video was sent in group to TA.