ARM C-Interfacing

Aim :

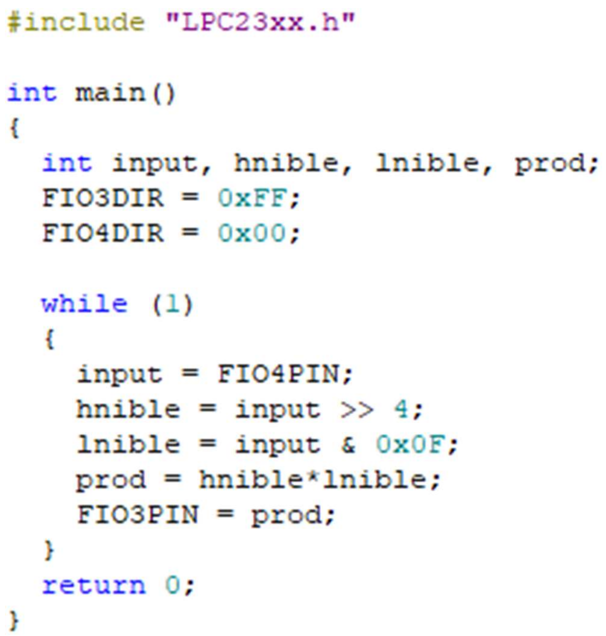
- Read status of switch and display the status using LED’s
- Stepper motor control

Tasks :

1) Program to dis-assemble a byte into two nibbles from the DIP switch states, multiply and display result using LED’s
2) Stepper motor to rotate in opposite direction of that given inn the code

Code :

1)



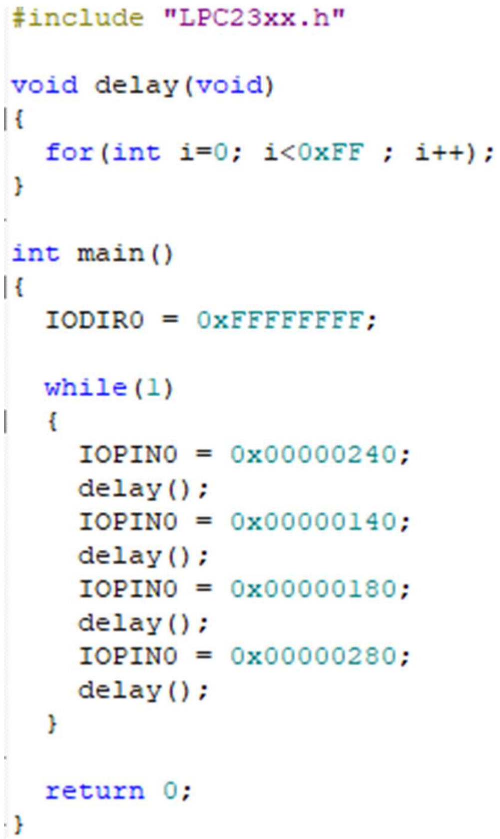
```
#include "LPC23xx.h"

int main()
{
  int input, hnible, lnible, prod;
  FIO3DIR = 0xFF;
  FIO4DIR = 0x00;

  while (1)
  {
    input = FIO4PIN;
    hnible = input >> 4;
    lnible = input & 0x0F;
    prod = hnible*lnible;
    FIO3PIN = prod;
  }
  return 0;
}
```

To obtain the highest nibble from 8-bit word, it is shifted 4 times to the right. And to obtain the lowest nibble it undergoes and operation with 0F(hex). The highest and the lowest nibble are multiplied and displayed with LED’s.

2)



```
#include "LPC23xx.h"

void delay(void)
{
  for(int i=0; i<0xFF ; i++);
}

int main()
{
  IODIR0 = 0xFFFFFFFF;

  while(1)
  {
    IOPIN0 = 0x00000240;
    delay();
    IOPIN0 = 0x00000140;
    delay();
    IOPIN0 = 0x00000180;
    delay();
    IOPIN0 = 0x00000280;
    delay();
  }

  return 0;
}
```

To make the stepper motor rotate in opposite direction the sequence in which the electro magnets in motor get magnetized is reversed. Therefore, the input into the pin which magnetizes the magnets is given in reverse order.

Conclusion :

Interfacing ARM with c makes the above code less complicated and more flexible.

Many different practical implementations can be done