Name: Narasimhan Srikanth Roll Number: EE20B087

EE2016 – Microprocessor Theory and Lab

Experiment 4: ARM C-Interfacing - Emulation of Switch LED and Stepper Motor Control

Target of the experiment:

The aim of this experiment is to use C-programming (C-interfacing) to do the following tasks:
a) Read the status (binary position) of the switch and use the LEDs (8 LEDs are provided) to display the status of each of the 8-bit DIP switch.
b) Stepper motor control using Vi Microsystem's ViARM 7238 development board.
Both the tasks are achieved through their emulator versions only.

Tasks:

Following are the tasks for the lab session:
1. Write a program (in C) to dis-assemble a byte into two nibbles from the DIP switch states, multiply and display the product in the LED.
2. Modify the demo code supplied to demonstrate the control of stepper motor to rotate in opposite direction (opposite to the direction shown in the demo code). The demo code was displayed in the lab session along with the desired output. The signals to the stepper motor are also to be identified.
3. Rotate the stepper through an angle of 80 degrees.

Flowchart and solutions:

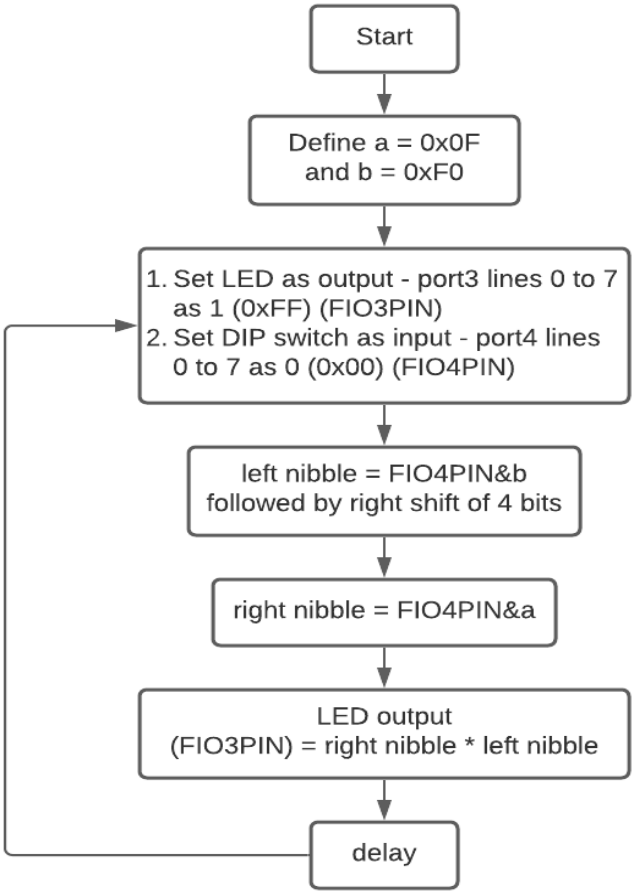
Problem 1: The DIP switch has 8 port lines and the ViARM 2378 development board has one 8-way DIP switch. The port lines used are port-4, pins 0 to 7. The board also has 8 LED pins in port-3 from pin 0 to pin 7.
LED pins are set to output and DIP lines are made input using PINxDIR register.

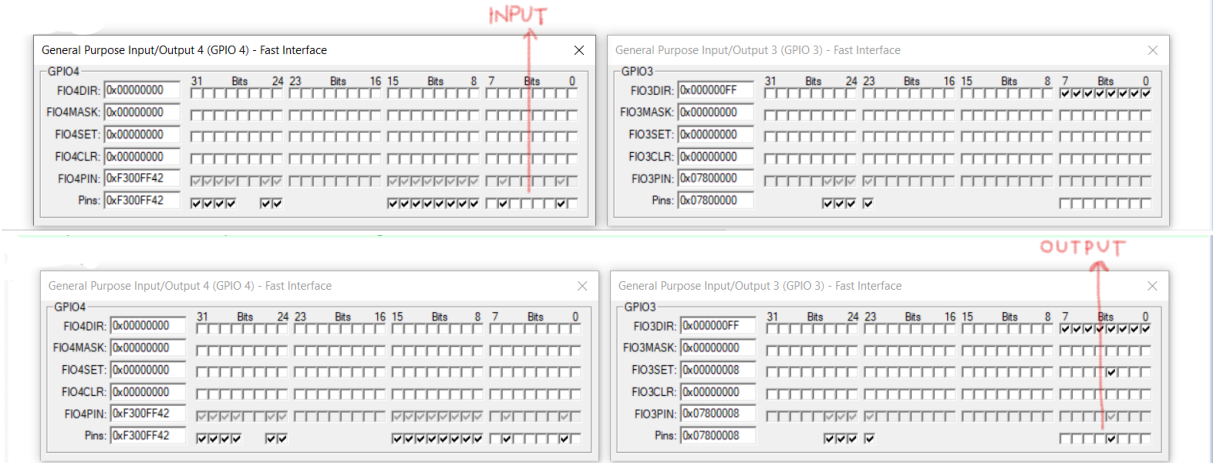
In this problem, the DIP pins are the inputs and the LEDs display the output. The lines 0 to 3 form one nibble and lines 4 to 7 form another nibble. The product of these two nibbles is displayed as output in the 8-bit long LED display line.

For example, if the DIP input is, (pin7) 01000010 (pin0), then the left nibble is 4 (0100) and right nibble is 2 (0010). Their product is 8 which is shown in the LED pins as, (pin7) 00001000 (pin0).
This process is repeated as long as the power supply is present. A delay is present at the end (in the form of a FOR loop) to make the output visible in the LED for a significant time duration. It also allows the user to change input in the switch by adjusting the lines of the switch (ON or OFF) before the next cycle starts.

To obtain the left nibble, I have applied the bit wise AND operation between FIO4PIN (register with values from line 31 to 0 in port 4) and 0xF0. Further, we have to right shift by 4 bits to make it a nibble (not a byte). For obtaining the right nibble, we don't have to do the extra step of right shift. Just performing an AND operation between FIO4PIN and 0x0F gives us the value. The result is stored in FIO3PIN (contains the LED lines). Click for C file

```c
#include "LPC23XX.h"
int a = 0x0F;
int b = 0xF0;
int main(void)
{
  while(1)
  {
    FIO3DIR = 0xFF;     //LED is output - pins 0 to 7
    FIO4DIR = 0x00;     //DIP switches are set as input - pins 0 to 7
    int leftNibble = (FIO4PIN&b)>>4;      //left nibble is FIO4PIN&0xF0 followed by right shift of 4 bits
    int rightNibble = (FIO4PIN&a);        //right nibble is FIO4PIN&0x0F
    FIO3PIN = rightNibble*leftNibble;     //multiplies the two nibbles to get the product and store it in LED
    for(int i = 0;i<0xFF;i++);            //delay
  }
  return 0;
}
```

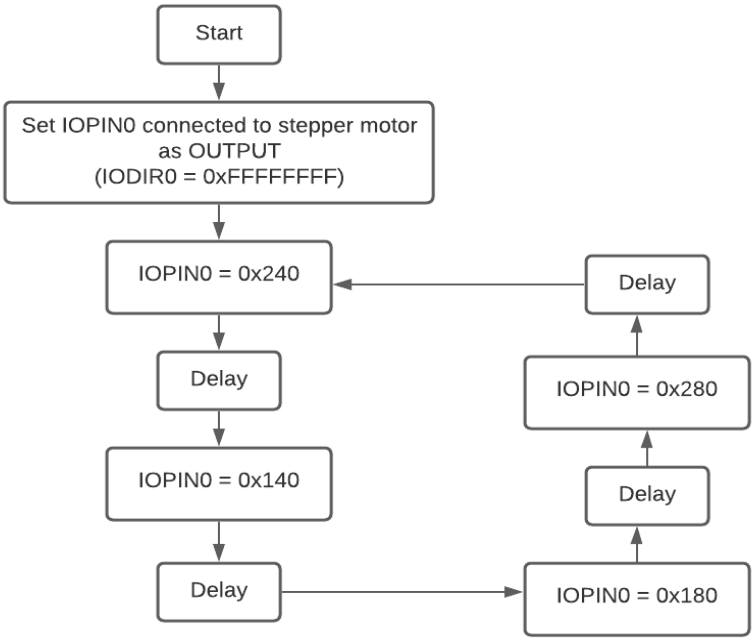




The above boxes show the sample input output (during debugging). For input 01000010 in FIO4PIN (last 8 lines) we are getting output as 00001000 in FIO3PIN.

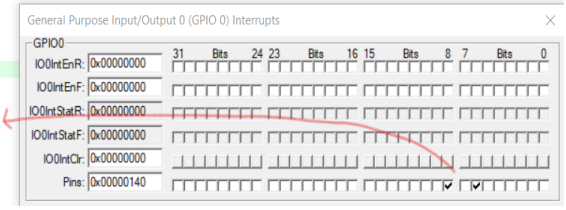
Problem 2: The IOPIN0 is connected to the motor and so, it is set to output mode by making IODIR0 1 for all lines. We control rotation by manipulating values of bit numbers 6 to 9 in IOPIN0.
In the demo code, the values written to IOPIN0 were ordered to rotate the stepper motor in clockwise sense. Hence to rotate it anti-clockwise sense, we have to reverse the order in the while loop. The speed of rotation is controlled by the delay() function. The longer the delay, the slower the rotation of the stepper motor. The smaller the delay, the quicker the steps are taken by the motor. Click for C file

```c
#include "LPC23xx.h"
void delay(void)
{ int i,j;
for(i=0; i<0xff;i++)
for(j=0; j<0XFF;j++);
}
int main(void)
{
  IODIR0 = 0XFFFFFFFF;
  // to rotate the loop by a specific angle we control this while statement.
  /* In the demo code, the values written to the IOPIN was in the order 0x280, 0x180, 0x140 and 0x240.
  This rotated the stepper motor in clockwise direction.
  Hence to rotate it opposite, we write the values to IOPIN in the reverse order.
  The speed of rotation is controlled by the values in delay function*/
  while(1)
  {
    IOPIN0=0X00000240;
    delay();
    IOPIN0=0X00000140;
    delay();
    IOPIN0=0X00000180;
    delay();
    IOPIN0=0X00000280;
    delay();
  }
  return 0;
}
```




```c
while(1)
{
  IOPIN0=0X00000240;
  delay();
  IOPIN0=0X00000140;
  delay();
  IOPIN0=0X00000180;
  delay();
  IOPIN0=0X00000280;
  delay();
}
return 0;
}
```

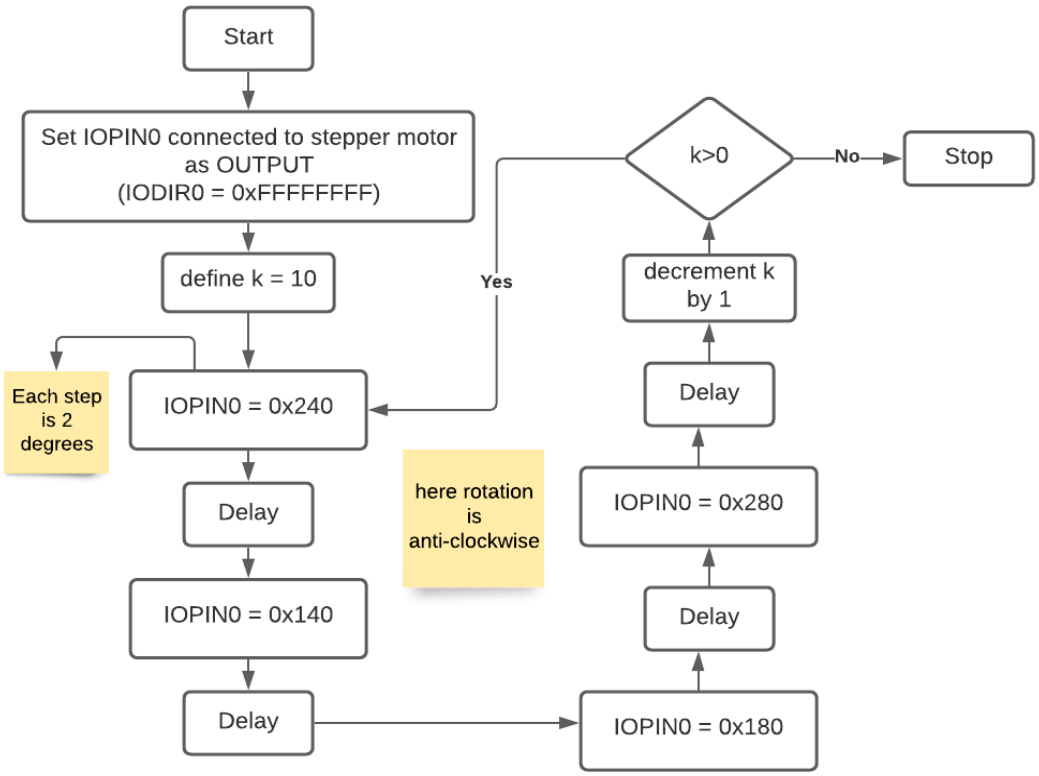
0x140 IN IOPIN0



Above is a picture of the IOPIN0 during debugging. The value is 0x140 in the lines connected to the motor. These are the *outputs signals transmitted to the stepper motor controlling its rotation*.

Problem 3: *We will make use of the assumption that every step of writing a value to IOPIN0 in the previous program, rotates the stepper motor by 2 degrees*. So, every time the entire while loop is executed, the stepper rotates by 8 degrees. Hence, to rotate it by 80 degrees, the while loop has to execute 10 times only. This is implemented by using a variable k within the while loop. So, k will go from 10 to 1 and loop will run as long as k>0. Click for C file

```c
#include "LPC23xx.h"
void delay(void)
{ int i,j;
for(i=0; i<0xff;i++)
for(j=0; j<0XFF;j++);
}
int main(void)
{
  IODIR0 = 0XFFFFFFFF;
  // to rotate the loop by a specific angle we control this while statement.
  /* In the demo code, the values written to the IOPIN was in the order 0x280, 0x180, 0x140 and 0x240.
  This rotated the stepper motor in clockwise direction.
  Hence to rotate it opposite, we write the values to IOPIN in the reverse order.
  The speed of rotation is controlled by the values in delay function*/
  // to rotate by 80 degrees, the while loop has to run 10 times.
  int k = 10;
  while(k>0)
  {
    IOPIN0=0X00000240;
    delay();
    IOPIN0=0X00000140;
    delay();
    IOPIN0=0X00000180;
    delay();
    IOPIN0=0X00000280;
    delay();
    k--;
  }
  return 0;
}
```



Inference:

From the above tasks under experiment-4, I got an introduction to C-interfacing in ARM processor. I was able to explore and manipulate the various GPIO and Fast GPIO registers in Keil IDE (in debugging mode). The demo videos and lab material gave a tour of the ViARM 2378 development board and a detailed explanation of the stepper motor and DIP switch interface.

Through the tasks, rotation of the stepper motor was controlled and data stored in the DIP switch was manipulated and displayed using the LEDs on the ViARM board.