California State University, Northridge

Department of Electrical & Computer Engineering

Lab 4

Lighting Up LEDs

September 29th, 2022

ECE 425L

Assigned by: Neda Khavari

Written By: Pathum Addarapathirana & Cristian Robles

Introduction:

Lab 4 furthers our understanding of the loading and storing instructions learned in previous labs by using them to learn how to output signals through microcontroller pins that control light emitting diodes (LEDs)

Procedure:

Equipment Used

- Keil uVision4
- Debugger

Description of Procedure

1. Set up startup code. This code is needed to startup the lab and it is pretty straightforward.

```
1                   GLOBAL Reset_Handler
2                   AREA    mycode, CODE, Readonly
3   Reset_Handler   ;this label is necessary for the first line of your code
```

2. In the next step we have to first set up GPIO. This means we have to set the pins as 'General Purpose Input and Output'. This is done by storing 0 into PINSEL0 address.

```
5   PINSEL0     EQU     0xE002C000      ;equate symbolic name PINSEL0 as address 0xE002C000
6
7                                       ;Selecting funcion as GPIO by writing all zeros to given address
8               MOV     r0, #0          ;moves #0 into register r0
9               LDR     r1, =PINSEL0    ;puts PINSEL0 in r1 register; CANNOT PUT MOV, outputs error in keil
10              STR     r0,[r1]         ;copies value stored in r0 to memory address of r1
```

3. Then we have to set the signal direction of pins to determine whether they are input pins or output pins. This is done by storing a '1' into each bit that we want to make an output pin of IO0DIR. We only needed to make the first 8 pins an output pin so we used the hex code 0x0000FF00.

```
13  IO0DIR      EQU     0xE0028008      ;IO0DIR assigned to 0xE0028008
14
15                                      ;Selecting signal direction of each port pin, we put '1' in each to bit to make it an output pin
16              MOV     r0, #0x0000FF00
17              LDR     r1, =IO0DIR     ;puts 0xE0028008 or IO0DIR to r1 register
18              STR     r0, [r1]        ;copies value stored in r0 (0x0000FF00) to memory address of r1
```

4. For our first task we are to turn all 8 LED's on. This is done by storing a '0' into each LED of IO0PIN.

```
21  IO0PIN      EQU     0xE0028000
22  ;task 1
23              MOV     r0, #0x00000000 ;moves #0x00000000 into register r0 (0 means ON)
24              LDR     r1, =IO0PIN     ;puts 0xE0028000 or IO0PIN to r1 register
25              STR     r0, [r1]        ;copies value stored in r0 (0x0000FF00) to memory address of r1 (binary #0000000000000000 - ALL LEDS ON)
```

5. For the second task we are to turn all the LED's off. Just like task 1 we have to set a '1' which to each LED of IO0PIN

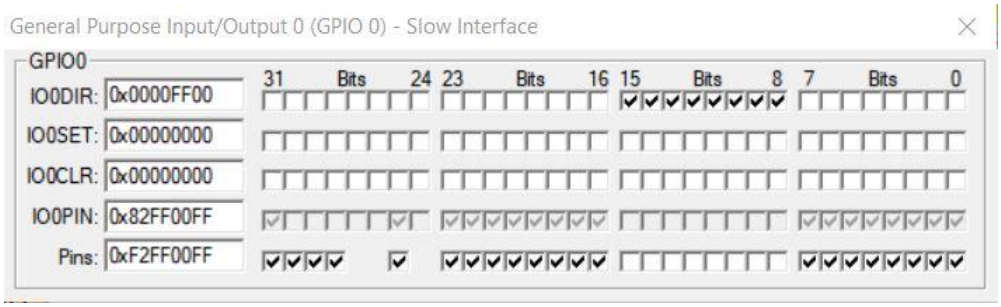
```
;task 2
            MOV     r3, #0x0000FF00 ;moves #0 into register r0
            LDR     r2, =IO0PIN     ;puts 0xE0028000 or IO0PIN to r1 register
            STR     r3, [r2]        ;copies value stored in r0 (0x00000000) to memory address of r1 (binary #1111111100000000 - ALL LEDS OFF)
```

6. For the third task we are to set the LED's in an alternating manner so that one LED is on and the next is off and so forth. We achieved this by storing IO0PIN to hexadecimal 0x00005500 which translates to '10101010'.

```
;task 3
            MOV     r0, #0x00005500 ;moves #0x00005500 into register r0
            LDR     r1, =IO0PIN     ;puts 0xE0028000 or IO0PIN to r1 register
            STR     r0, [r1]        ;copies value stored in r0 (0x00005500) to memory address of r1 (binary #101010100000000)
```

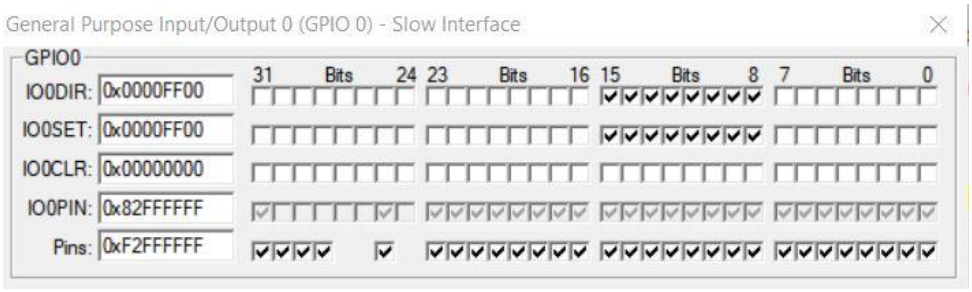
Results:

Task 1



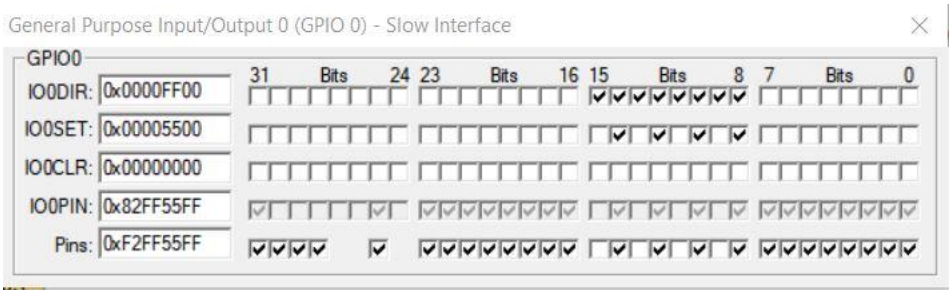
For task 1 the peripheral tab in the debugger confirmed that the code set bits 8-15 of the IO0DIR in order to make the pins outputs and set bits 8-15 of the IO0SET to 0 in order to get the all eight LEDs to turn on

Task 2



For task 2 the peripheral tab in the debugger confirmed that we set bits 8-15 of the IO0DIR in order to make the pins outputs and set bits 8-15 of the IO0SET to 1 in order to get the all eight LEDs to turn off

Task 3



For task 3 the peripheral tab in the debugger confirmed that we set bits 8-15 of the IO0DIR in order to make the pins outputs and set 1's and 0's alternated in pins 8-15 of the IO0SET in order to get the LEDs to alternate between on and off

Questions:

Question 1: What signal does the microcontroller need to output on pin P0.8 in order to light up LED5, in Figure 1?

The signal output to pin P0.8 needs to be 0 in order for LED5 to light up

Question 2: Is it ok to replace LDR by MOV in the second line of the above assembly code?

No, it is not ok to replace LDR by MOV because MOV can only be 8 bits and the value of PINSEL0 is greater than that

Question 3: You need to configure the pins connected to LEDs in Figure 1 as input pin or output pin?

We need to make them output pins

Question 4: You need to configure which bits of the direction register IO0DIR for the purpose of controlling these eight LEDs?

We need to configure bits 8-15 for controlling the 8 LEDs we will be working with

Question 5: Why can instruction MOV be used in the first line of the above assembly code?

Instruction MOV can be used because the value in the register can be represented as 8 bits and a rotation

Conclusion:

In conclusion, we were able to set GPIO and set each pin as inputs and outputs. We were also able to send signals in order to control the LEDs connected to these pins. As shown in the results we were able to verify our code through simulation but not on the actual board as we were not in class for this experiment. A way that this lab could be improved is to have a delay in between the tasks. During the simulation as we are stepping through

the code we can see the program at different points but on the board it would move so fast that we would only see the last task. This project furthered our knowledge of the store and load instructions and taught us how to apply them to control pins on the board.