Spring 2022

Haroutun Haroutunian
Andres Lopez

California State University Northridge
Department of Electrical and Computer Engineering

Experiment 5
Flashing LEDs

Professor Flynn
4 March, 2022

Introduction

Experiment 5 carries on from Experiment 4, where students develop an assembly program which flashs the LPC2148 Education board. The flash overwrites the existing code from Experiments 1-3 to toggle the LED pins (pins 8-15). Furthermore, students begin to implement previous understandings of mnemonics and labels to develop a proper running assembly program, as well as apply knowledge of oscilloscopes to measure the waveform of the LED pin outputs.

Procedure

The following symbols are used in this lab. These are used for determining the funcatinaly of the pins.

- PINSEL0 = assigns the pins P0.8-P0.15 as GPIO(General Purpose Input/Output) pins
- IO0DIR = used to set the direction of the pins as with inputs or outputs
- IO0PIN =sends a 0 or 1 to the port pins connected to the LEDs
- IO0SET = used to set a “1” to a given bit in the IO0PIN memory address
- IO0CLR = used to clear a bit in the IO0PIN register, by placing “1” in memory
- MAM = Memory Accelerator Module, which is utilized to enhance the performance of the ARM processor
- MAMCR = MAM Control Register determines the operating mode of the MAM, whether disabled or enable
- MAMTIM = MAM Timing Register determines how many clocks are used to access Flash Memory

First task requires us to make all of the 8 LEDs shown in Figure 1 illuminate. Like last week's lab, we need IO0DIR to set up the direction of the pins as shown in Figure 2.

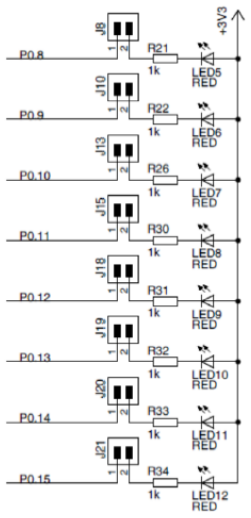
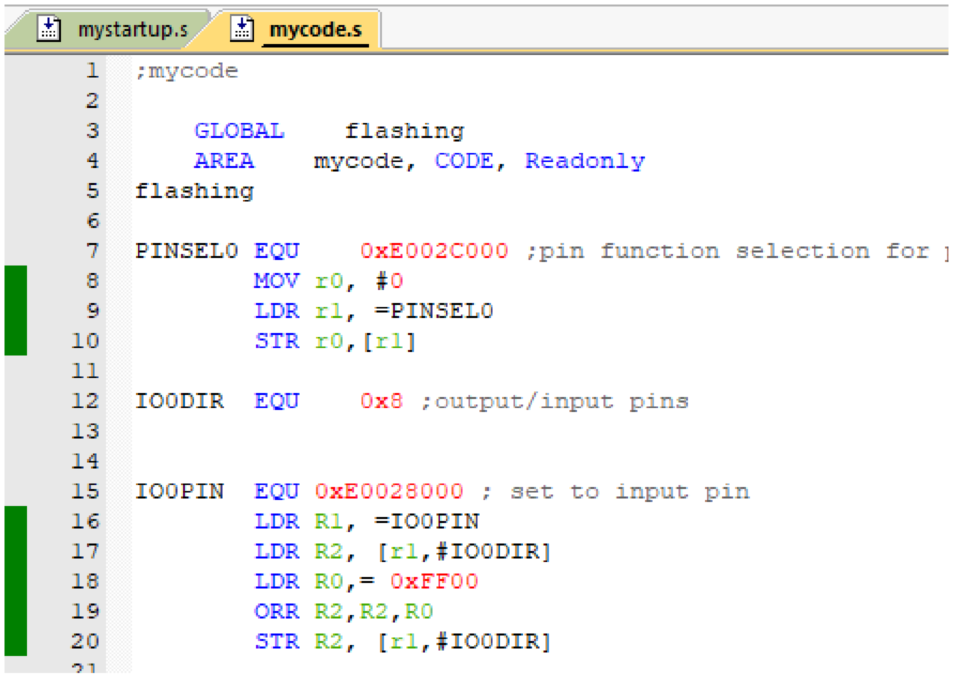


Figure 1. LPC2148 Education Board Schematic: LEDs

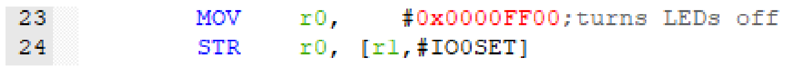


```
;mycode

    GLOBAL    flashing
    AREA    mycode, CODE, Readonly
flashing

PINSEL0 EQU     0xE002C000 ;pin function selection for
        MOV r0, #0
        LDR r1, =PINSEL0
        STR r0,[r1]

IO0DIR  EQU     0x8 ;output/input pins


IO0PIN  EQU 0xE0028000 ; set to input pin
        LDR R1, =IO0PIN
        LDR R2, [r1,#IO0DIR]
        LDR R0,= 0xFF00
        ORR R2,R2,R0
        STR R2, [r1,#IO0DIR]
```

Figure 2: PINSEL0 and IO0DIR function for the GPIO pins

PINSEL0 is the control register and it is configured to be GPIO. Inside the GPIO, there are 2 registers, IO0DIR and IO0PIN. IO0DIR sets the signal direction of GPIO for each pin. Configuring bits 8-15 of the IO0DIR register to “0” makes the pin an input (“1” for input) and allows us to write the next register which is IO0PIN. Finally, we are able to send a “0” or “1” to the port pins connected to the LEDs to the corresponding pin of the pin value register (IO0PIN), which uses the address 0xE0028000.

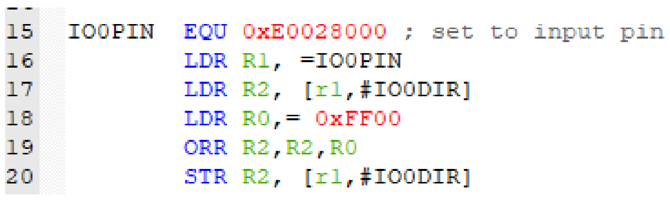
By writing a 32-bit number to the IO0PIN register, we modify all the output pins from P0.0 to P0.31 however, in order to control the eight LEDs, only pins P0.8- P0.15 need to be configured. We can do this by using IO0SET and IO0CLR. To set a certain bit in IO0PIN, we can select the corresponding bit positions in IO0SET with “1”s. Likewise, to clear a certain bit in the value register IO0PIN, we can select the corresponding bit positions in IO0CLR with “1”s.



```
        MOV     r0,     #0x0000FF00;turns LEDs off
        STR     r0,  [r1,#IO0SET]
```

Figure 3: Immediate move of mask into register 0, then storing into the (base) register 1 + (offset) IO0SET address

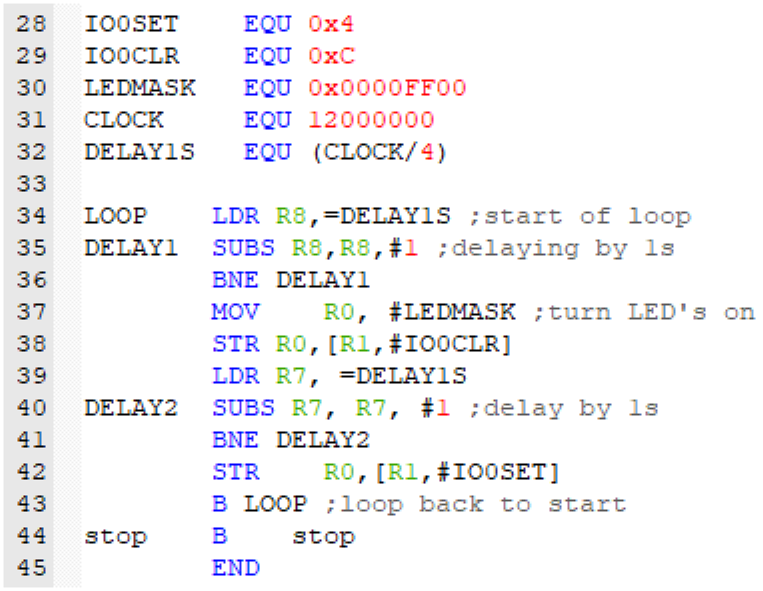
In this lab, we will be using pre-index addressing modes to construct the effective address of GPIO pins. Figure 4 demonstrates how this new addressing mode is created along with its offsets. IO0PIN is loaded into register 1, which reads the state of the pin. Register 2 is loaded with IO0DIR, which reads the last written bit of the pin. Register 0 is loaded with a mask to turn the LEDs off. The ORR mnemonic sets when used with a mask, “or-ing” register 2 with register 0 and immediately being loaded into register 2, which is then stored into IO0DIR and displayed its last written state.



```
IO0PIN  EQU 0xE0028000 ; set to input pin
        LDR R1, =IO0PIN
        LDR R2, [r1,#IO0DIR]
        LDR R0,= 0xFF00
        ORR R2,R2,R0
        STR R2, [r1,#IO0DIR]
```

Figure 4: Utilization of masks to turn the LEDs off

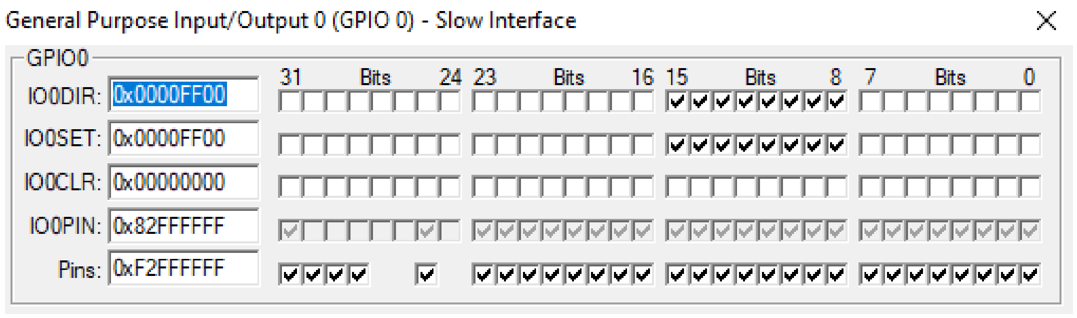
The delay function has been introduced from last week's lab. This function allows the LEDs to blink for a long period of time, which also prevents the LED’s from instantaneously toggling. This allows us to actually see the LEDs current state. In figure 6, students utilized their knowledge on loops and pre-indexing to turn the LEDs on. From lines 28-32, students created “find and replace” to begin the process of the loop of toggling the LEDs. Clock replaced with the value of 12000000 because of the 12MHz crystal quartz clock in the process, then is divided by 4 in Delay1s because the total number of instructions for the loop utilizes 4 clock cycles. In line 34,we created a loopback point and loaded register 8 with 3000000, which is subtracted by 1 and loaded immediately back in line 35. Line 36 utilized the “Branch if Not Equal” mnemonic, which continuously branches back to line 35 until register 8 is equivalent to 0. This process is repeated on lines 39-41 to delay the toggling process of the LED by 1 second. In line 38, students stored the contents of Register 0 into the IO0CLR by utilizing Register 1 as the base reference register and #IO0CLR as the offset, which adds R1 with 0xC, referring to the Clear address which turns the LEDs on.



```
IO0SET     EQU 0x4
IO0CLR     EQU 0xC
LEDMASK    EQU 0x0000FF00
CLOCK      EQU 12000000
DELAY1S    EQU (CLOCK/4)

LOOP    LDR R8,=DELAY1S ;start of loop
DELAY1  SUBS R8,R8,#1 ;delaying by 1s
        BNE DELAY1
        MOV    R0, #LEDMASK ;turn LED's on
        STR R0,[R1,#IO0CLR]
        LDR R7, =DELAY1S
DELAY2  SUBS R7, R7, #1 ;delay by 1s
        BNE DELAY2
        STR    R0,[R1,#IO0SET]
        B LOOP ;loop back to start
stop    B    stop
        END
```

Figure 5:Continuous loop of toggling the LEDs every 1 second

Figure 6 demonstrates the 2 states the LEDs are in; on and off.

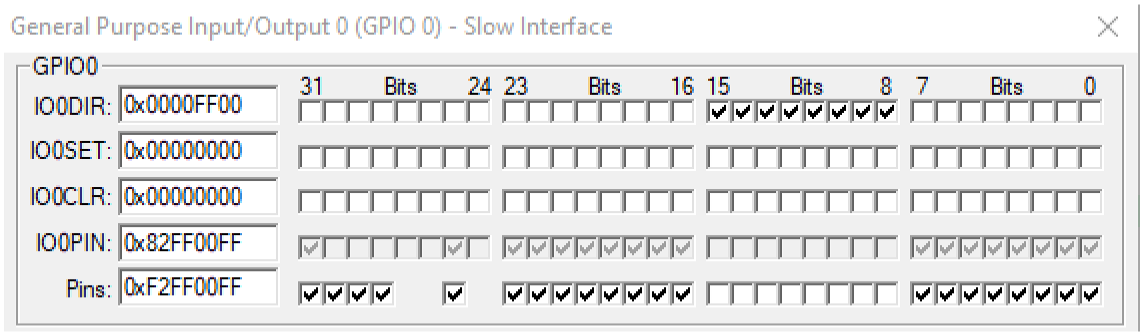


General Purpose Input/Output 0 (GPIO 0) - Slow Interface

GPIO0

IO0DIR:	0x0000FF00
IO0SET:	0x0000FF00
IO0CLR:	0x00000000
IO0PIN:	0x82FFFFFF
Pins:	0xF2FFFFFF

Figure 6: PINs 8-15 have been turned on from address 0xFF00 in IO0SET



General Purpose Input/Output 0 (GPIO 0) - Slow Interface

GPIO0

IO0DIR:	0x0000FF00
IO0SET:	0x00000000
IO0CLR:	0x00000000
IO0PIN:	0x82FF00FF
Pins:	0xF2FF00FF

Figure 7: PINs 8-15 have been turned off from address 0x00 in IO0SET

Students utilized an oscilloscope provided by the lab professor to measure the waveform of an LED being toggled. Utilizing the table from Figure 9, students connected a probe from the oscilloscope to Connector Pin 36 and Connector Pin 17, which are hardwired to P0.8 and Ground, respectively. As shown in Figure 8, the LED toggles every second due to the 1 second delay students implemented into the experiment.

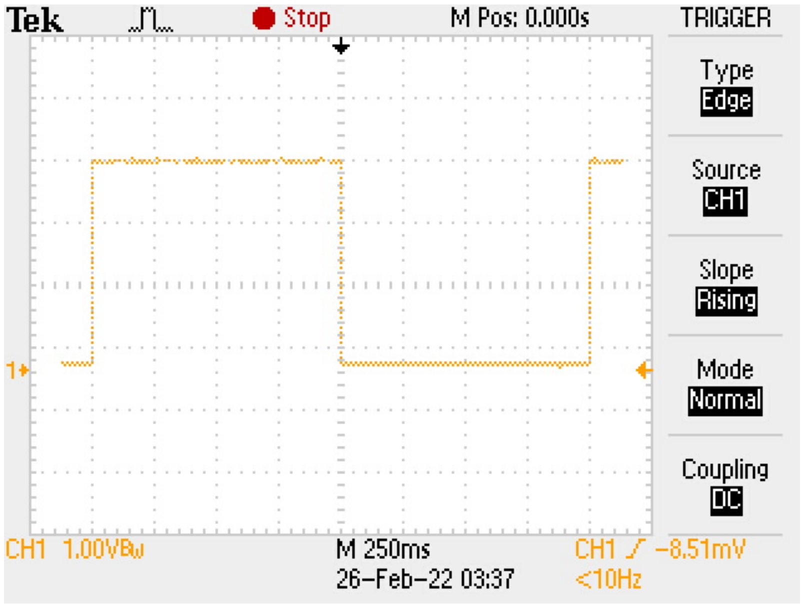
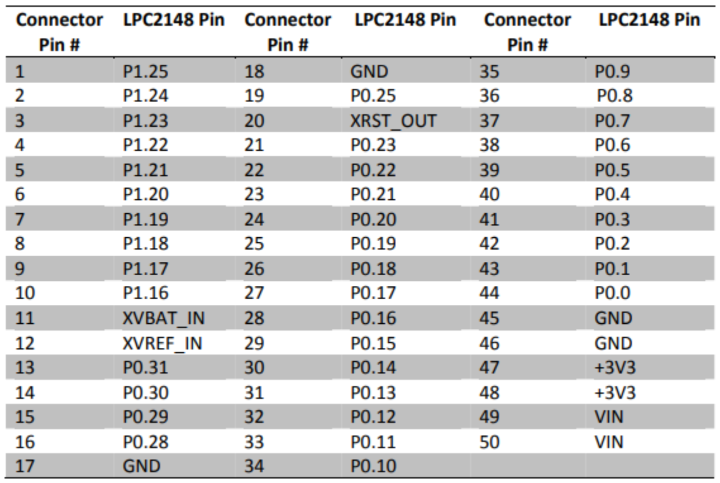


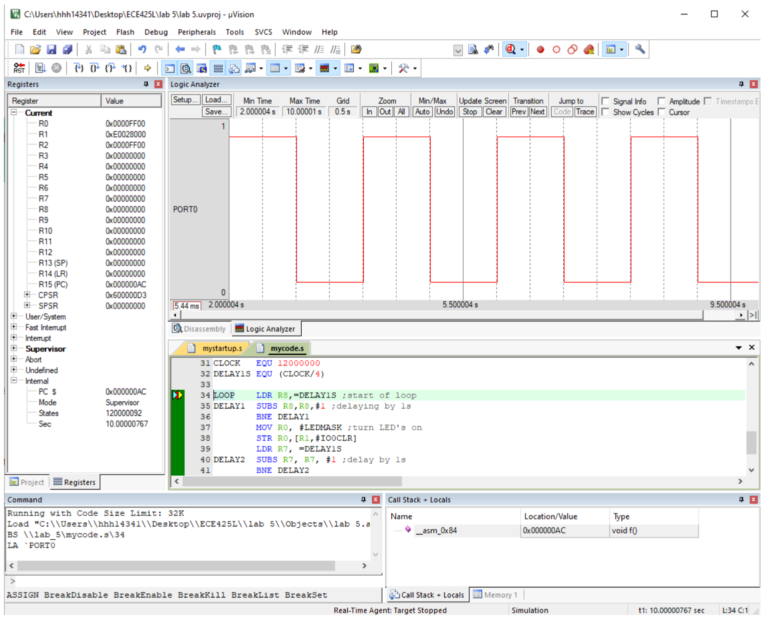
Figure 8: Oscilloscope waveform of LED Connector Pin 36 (pin0.8) and Connector Pin 17 (GND)



Connector Pin #	LPC2148 Pin	Connector Pin #	LPC2148 Pin	Connector Pin #	LPC2148 Pin
1	P1.25	18	GND	35	P0.9
2	P1.24	19	P0.25	36	P0.8
3	P1.23	20	XRST_OUT	37	P0.7
4	P1.22	21	P0.23	38	P0.6
5	P1.21	22	P0.22	39	P0.5
6	P1.20	23	P0.21	40	P0.4
7	P1.19	24	P0.20	41	P0.3
8	P1.18	25	P0.19	42	P0.2
9	P1.17	26	P0.18	43	P0.1
10	P1.16	27	P0.17	44	P0.0
11	XVBAT_IN	28	P0.16	45	GND
12	XVREF_IN	29	P0.15	46	GND
13	P0.31	30	P0.14	47	+3V3
14	P0.30	31	P0.13	48	+3V3
15	P0.29	32	P0.12	49	VIN
16	P0.28	33	P0.11	50	VIN
17	GND	34	P0.10		

Figure 9: Table of GPIO of EduBoard pins 1-50

Task 3 requires us to use the Keil's Logic Analyzer tool, under Peripherals, to display signal changes on the LED's. To access the Logic Analyzer, we go into debug and select "Logic Analyzer" on the toolbar. Next, a window shows up and we select Insert and enter PORT0 And Mask. Then, click close. Select Run and hit autoscale to adjust the graph.



Register	Value
Current	
R0	0x0000FF00
R1	0xE0028000
R2	0x0000FF00
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x000000AC
CPSR	0x600000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x000000AC
Mode	Supervisor
States	120000092
Sec	10.00000767

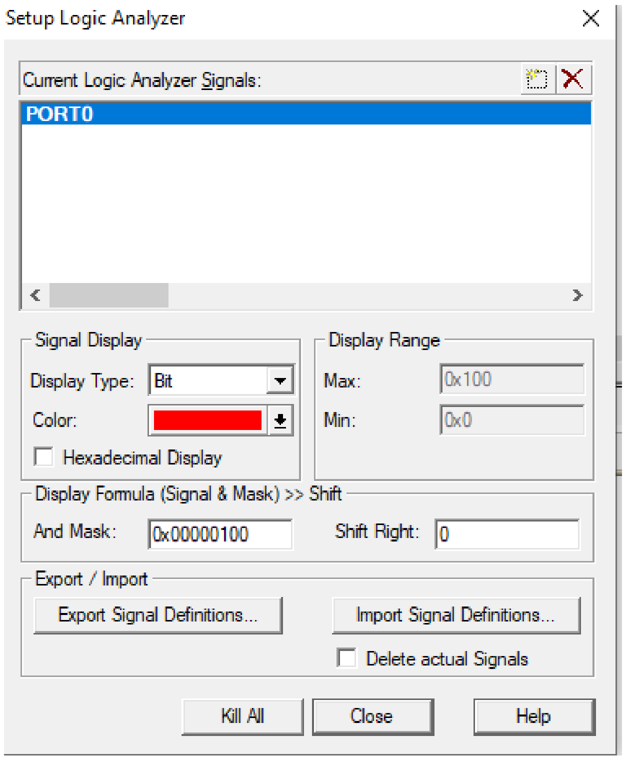
```
CLOCK    EQU 12000000
DELAY1S  EQU (CLOCK/4)

LOOP     LDR R8,=DELAY1S ;start of loop
DELAY1   SUBS R8,R8,#1 ;delaying by 1s
         BNE DELAY1
         MOV R0, #LEDMASK ;turn LED's on
         STR R0,[R1,#IO0CLR]
         LDR R7, =DELAY1S
DELAY2   SUBS R7, R7, #1 ;delay by 1s
         BNE DELAY2
```

```
Running with Code Size Limit: 32K
Load "C:\\Users\\hhh14341\\Desktop\\ECE425L\\lab 5\\Objects\\lab 5.a
BS \\lab_5\mycode.s\34
LA `PORT0
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

Name	Location/Value	Type
__asm_0x84	0x000000AC	void f()

Real-Time Agent: Target Stopped   Simulation   t1: 10.00000767 sec   L:34 C:1

Figure 10: Logic Analyzer which analyzes the toggling pin



Current Logic Analyzer Signals:
PORT0

Signal Display
Display Type: Bit
Color: 🟥
☐ Hexadecimal Display

Display Range
Max: 0x100
Min: 0x0

Display Formula (Signal & Mask) >> Shift
And Mask: 0x00000100
Shift Right: 0

Export / Import
☐ Delete actual Signals

Figure 11: Setup configuration of Logic Analyzer confirmed

Discussion

Students begin the experiment by creating the directional layout of the pins, as shown in figure 1. Then, “find and replace” mnemonics are utilized to set the necessary addresses of the pins. After all the initializing, the code ensures to set the LEDs off at the start of the program, before proceeding to the loop. Clock and Delay labels are set to specific calculations to delay the toggling of the LEDs by 1 second. A loopback point is then set to refer back to when the program reaches the end of the toggling to continuously run through itself. Register 7 and 8 are used as the delay registers and subtracted by 1, then immediately loaded back into itself until they reach 0. Once they reach 0, the “body” of each inner loop is run, toggling the LED with masks to their respective state by the IO0SET and IO0CLR. After the program is flashed onto the EduBoard, students view the GPIO through the Peripheral settings to view the state of the LEDs on pins 8-15. An oscilloscope is then connected to Connector Pins 36 and 17, Pin0.8 and GND, respectively, as shown in Figures 8 & 9.

Within the experiment, previous knowledge of pre-indexing, mnemonics and masks are utilized to create an efficient and proper running code. Loops were the main point of the program, because it allows the LEDs continuously toggle, switching their states every 1 second. Pre-indexing also cut the necessary clock cycles to a minimum due to the fact that only a single register is loaded with a base address, IO0PIN, to refer to IO0SET, IO0CLR and IO0DIR.

Conclusion

Throughout experiment 5, students utilized knowledge of loops, pre-indexing, masks, and oscilloscopes to create an established and proper running code to toggle the LEDs on the Education Board every 1 second, continuously. Assembly allows us to manipulate the states of pins to our desired state through the use of a mask and loading/storing to registers. IO0SET and IO0CLR are the primary addresses which are utilized as offsets within the register-loading pre-indexing to continuously toggle the LED. Rather than utilizing multiple registers to load individual addresses to toggle the LED, students only utilized the IO0SET at the base and indexed to the desired address, furthermore cutting the process time.