ECE 222 Fall 2024

Introduction to Lab 1

Flashing LED

UNIVERSITY OF WATERLOO
FACULTY OF ENGINEERING

Outline

- Objective
- General Notes
- Branching
- I/O port addressing and configuration
- Flashing LED flowchart
- Delay Calculation
- What to deliver

LAB 1 Objective

- Write some RISC-V assembly language instructions.
- Use different memory addressing modes.
- Test and debug the code on the RISC-V development board.

General Notes

- In LEARN, under Lab 1 Templates, you will download files:
    - Lab_1.h (DO NOT MODIFY!)
    - Lab_1_top.c (DO NOT MODIFY!)
    - Lab_1.S
- You will need to modify the C wrapper code for the Lab_1 to call the Lab_1 assembly code to get the LED to flash.
- You will need to flash an LED (Light Emitting Diode) at an approximate 1 Hz frequency

Branching

- BNEZ will branch if the result of the instruction is not equal 0.
  - BNEZ t0, label
- BEQZ will branch if the result of the instruction is equal to 0.
  - BNQZ t0, label

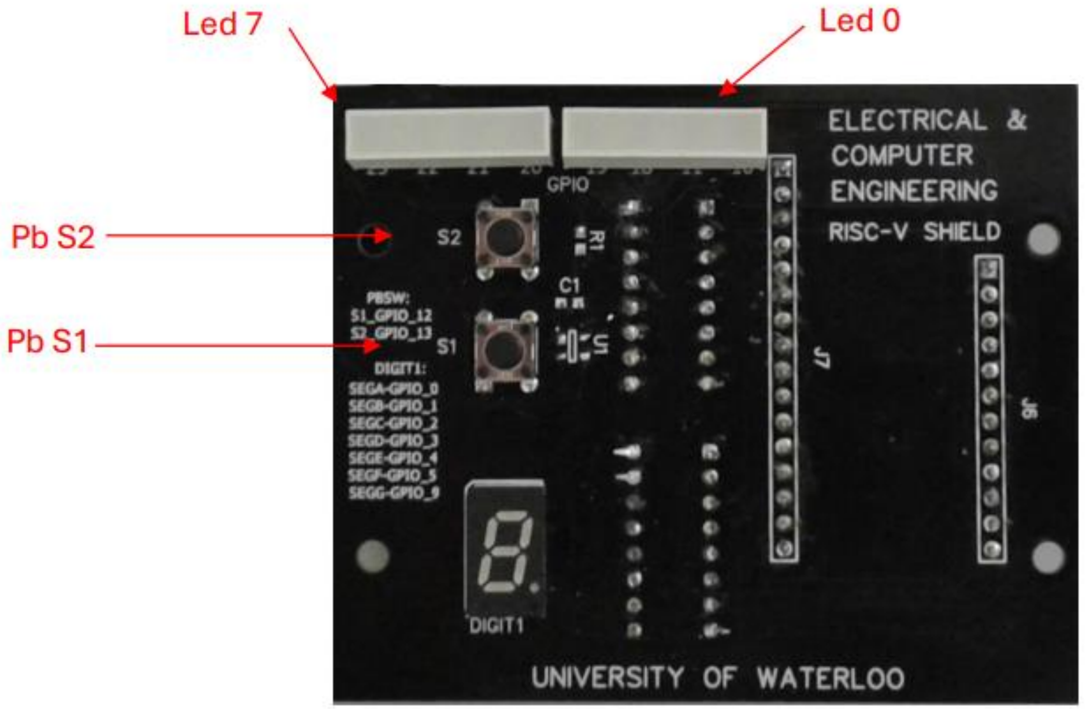
I/O port addressing and configuration

- GPIO_7SEGLED_PINS (0x0000023F): Pins used for the Seven Segment LED display. (pins GPIO 0 – 5, 9)
  - ✓ 0000 0000 0000 0000 0000 0010 0011 1111
- GPIO_LEDBAR_PINS (0x00FC0C00): Pins for an LED bar (8 LEDs). (pins GPIO 10, 11, 18 - 23)
  - ✓ 0000 0000 1111 1100 0000 1100 0000 0000
- GPIO_ALL_LED_PINS (0x00FC0E3F): All pins for both the Seven Segment display and the LED bar (a total of 15 pins).
  - ✓ 0000 0000 1111 1100 0000 1110 0011 1111
- GPIO_LEDBAR_LED_1 (0x00000800): The pin for controlling the first LED in the LED bar.
  - ✓ 0000 0000 0000 0000 0000 1000 0000 0000

I/O port addressing and configuration

- LED port addresses
  - Writing 0x00FC0E3F (GPIO_ALL_LED_PINS) into memory address 0x1001200C (GPIO_OUTPUT_VAL) to turn “off” the 8 LEDs on the LED Bar (pins GPIO 10, 11, 18 - 23) and to turn “off” the 7 LEDs of the digital display (pins GPIO 0 – 5, 9)
  - Toggle bit 11 (bit 1 of the LED bar) of address 0x00FC0E3F (GPIO_ALL_LED_PINS) and write to memory address 0x1001200C (GPIO_OUTPUT_VAL).
  - You will be toggling bit 11 between a 1 and a 0 with a 1 Hz frequency (500ms ON, 500ms OFF). Recall that the memory address is 32 bits wide (31 to 0)

RISC-V board

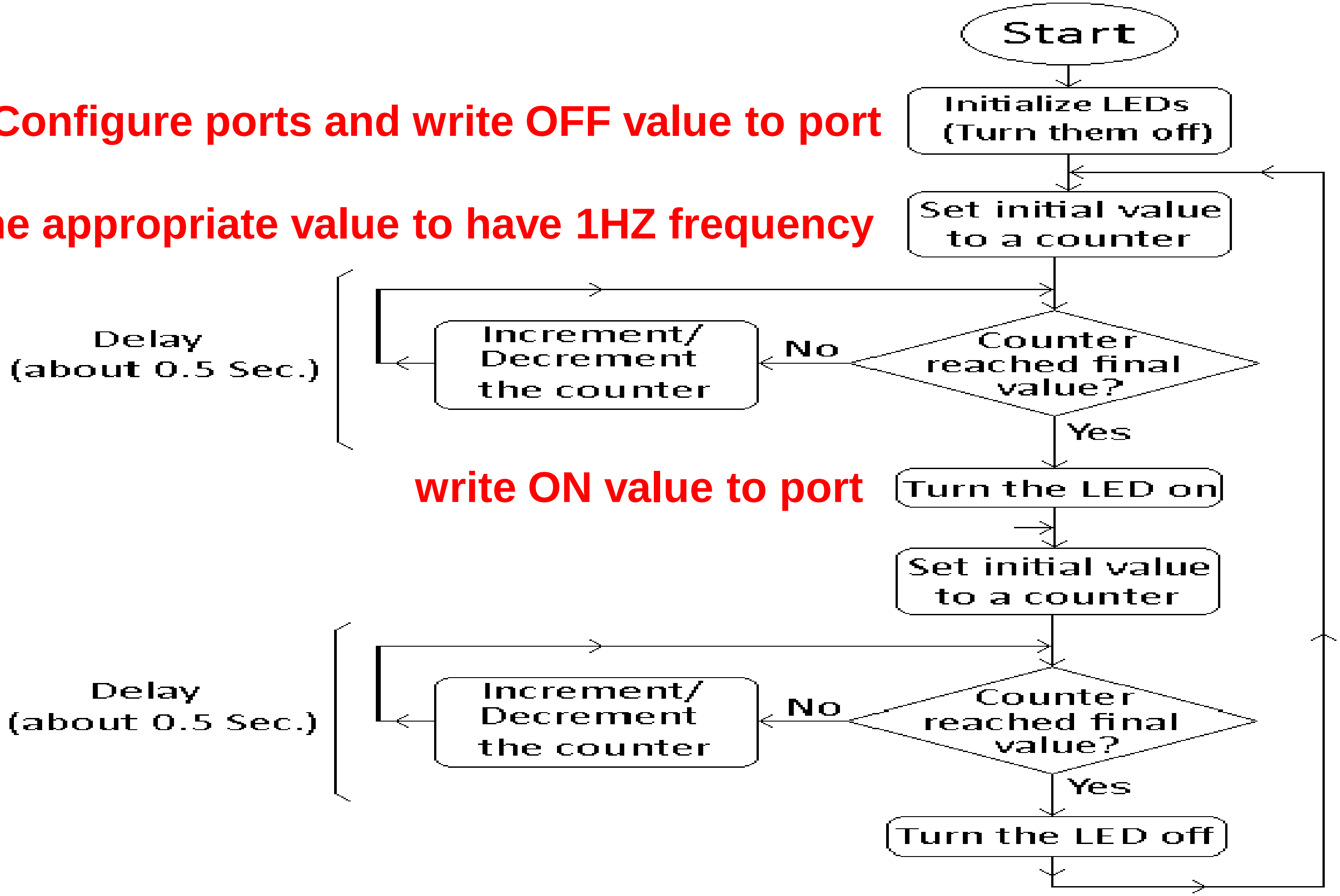

Led 7
Led 0
Pb S2
Pb S1
ELECTRICAL &
COMPUTER
ENGINEERING
RISC-V SHIELD
GPIO
S2
S1
R1
C1
U1
J7
J6
PBSW:
S1_GPIO_12
S2_GPIO_13
DIGIT1:
SEGA-GPIO_0
SEGB-GPIO_1
SEGC-GPIO_2
SEGD-GPIO_3
SEGE-GPIO_4
SEGF-GPIO_5
SEGG-GPIO_9
DIGIT1
UNIVERSITY OF WATERLOO

Flashing LED flowchart

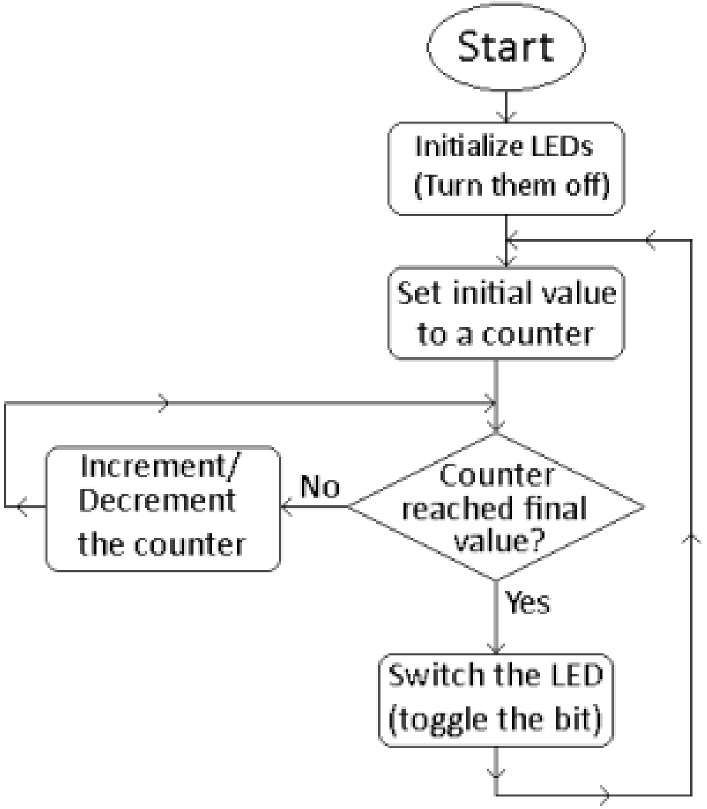
Configure ports and write OFF value to port

Use the appropriate value to have 1HZ frequency

write ON value to port


Start
Initialize LEDs (Turn them off)
Set initial value to a counter
Delay (about 0.5 Sec.)
Increment/ Decrement the counter
No
Counter reached final value?
Yes
Turn the LED on
Set initial value to a counter
Delay (about 0.5 Sec.)
Increment/ Decrement the counter
No
Counter reached final value?
Yes
Turn the LED off

Flashing LED flowchart

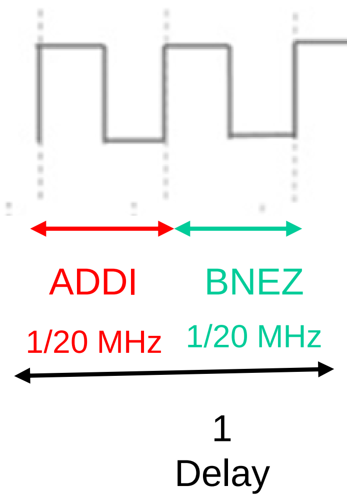

Start
Initialize LEDs (Turn them off)
Set initial value to a counter
Counter reached final value?
No
Increment/ Decrement the counter
Yes
Switch the LED (toggle the bit)

Information to Calculate Delay

- Clock frequency: 20 MHz
- Command ADDI: 1 clock cycle
- Command BNEZ: 1 clock cycle

Delay

```
Delay
        ADDI            1
        BNEZ Delay      1
```



Clock Frequency/Speed (F) = 20 MHz

F = 20 MHz

T=1/20 MHz

Here,

1 loop = 2/20 MHz

=>No. of loops x 2/20 MHz =0.5

1 Hz

ON for 0.5 Sec

OFF for 0.5 Sec

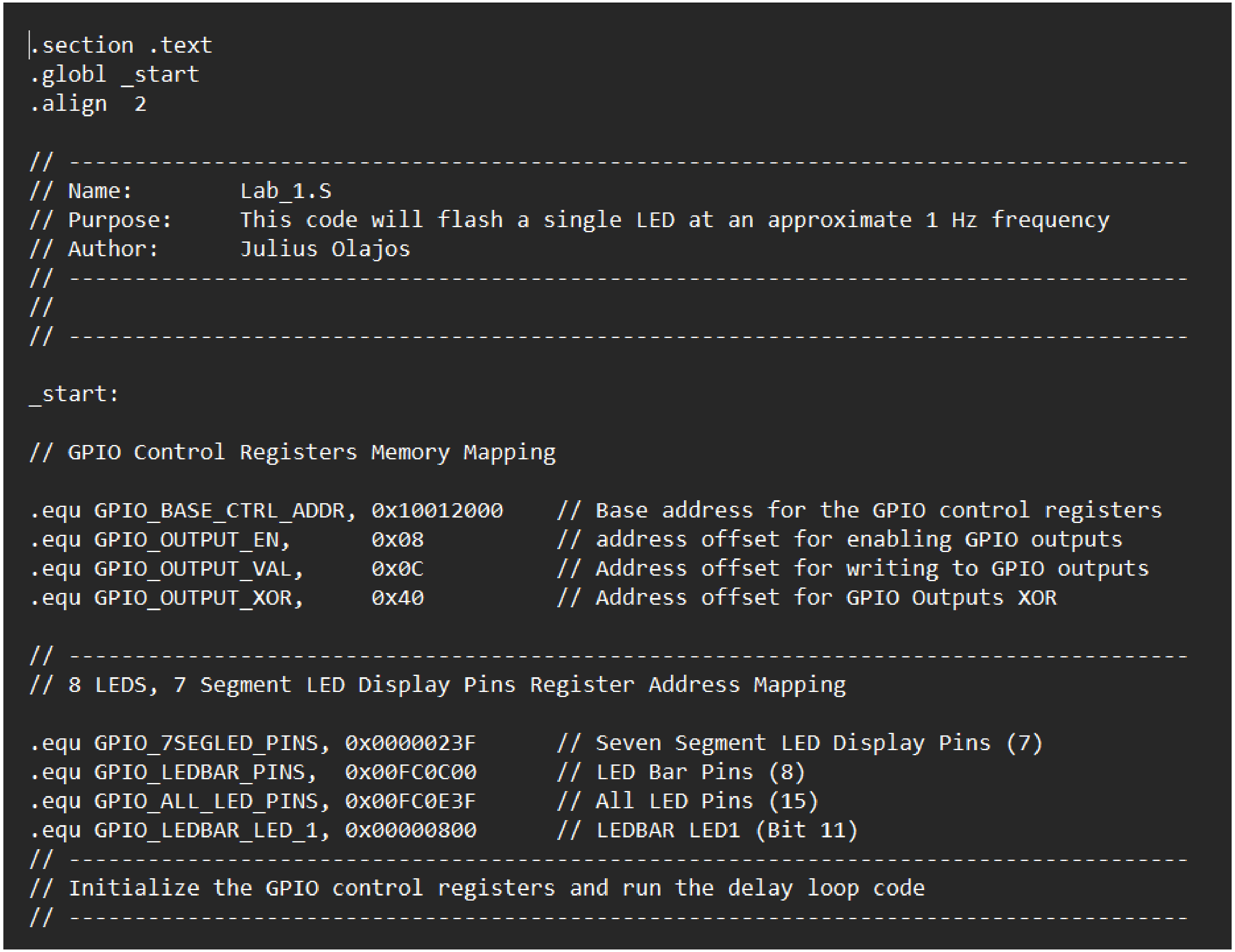
Formula to Calculate Delay

Clock Cycles in Loop= (Clock Speed x Time Delay)/Instruction Cycles

Clock Frequency/Speed (F) = 20 MHz

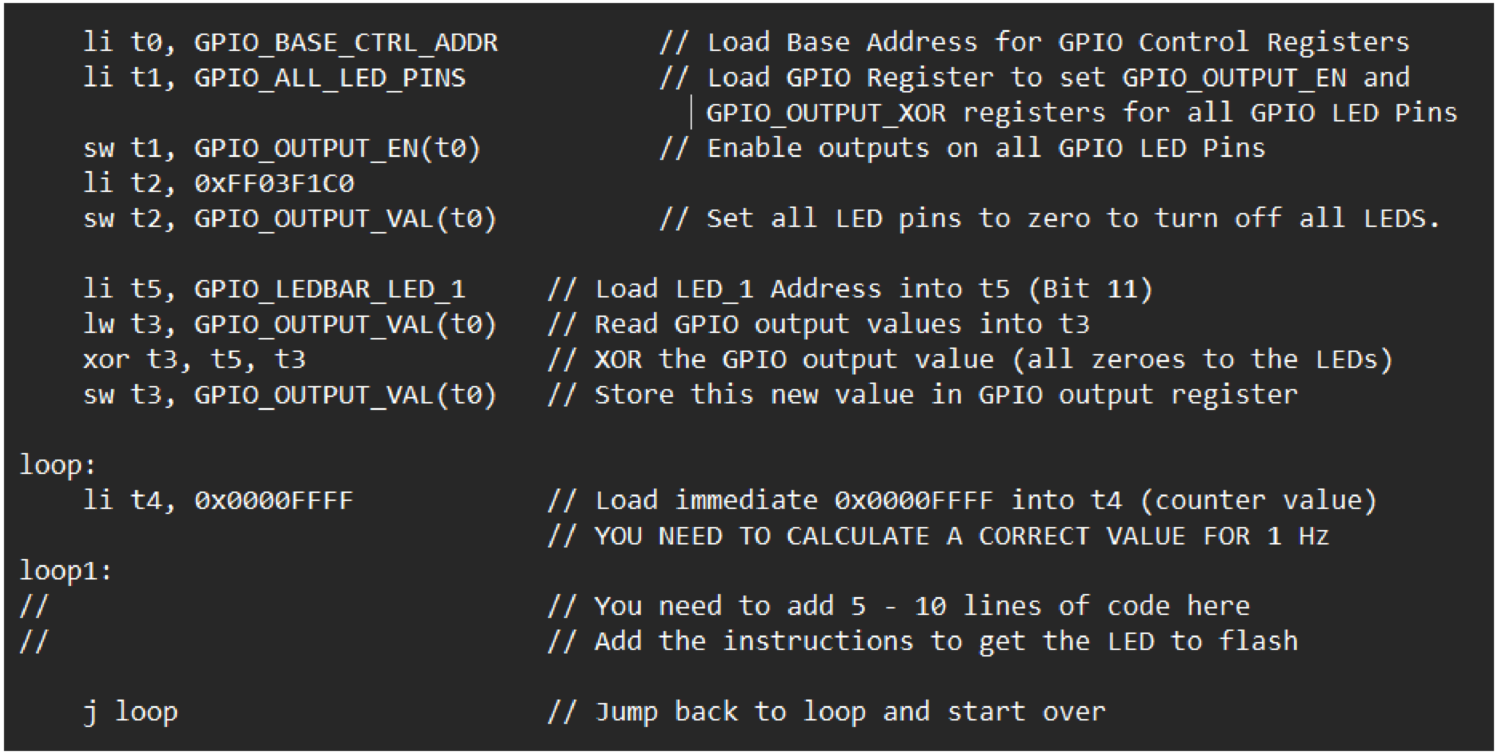
```
Delay
  1    ADDI
  1    BNEZ Delay
```

Template Code



```
.section .text
.globl _start
.align  2

// -------------------------------------------------------------------------------------
// Name:        Lab_1.S
// Purpose:     This code will flash a single LED at an approximate 1 Hz frequency
// Author:      Julius Olajos
// -------------------------------------------------------------------------------------
//
// -------------------------------------------------------------------------------------

_start:

// GPIO Control Registers Memory Mapping

.equ GPIO_BASE_CTRL_ADDR, 0x10012000    // Base address for the GPIO control registers
.equ GPIO_OUTPUT_EN,      0x08          // address offset for enabling GPIO outputs
.equ GPIO_OUTPUT_VAL,     0x0C          // Address offset for writing to GPIO outputs
.equ GPIO_OUTPUT_XOR,     0x40          // Address offset for GPIO Outputs XOR

// -------------------------------------------------------------------------------------
// 8 LEDS, 7 Segment LED Display Pins Register Address Mapping

.equ GPIO_7SEGLED_PINS, 0x0000023F      // Seven Segment LED Display Pins (7)
.equ GPIO_LEDBAR_PINS,  0x00FC0C00      // LED Bar Pins (8)
.equ GPIO_ALL_LED_PINS, 0x00FC0E3F      // All LED Pins (15)
.equ GPIO_LEDBAR_LED_1, 0x00000800      // LEDBAR LED1 (Bit 11)
// -------------------------------------------------------------------------------------
// Initialize the GPIO control registers and run the delay loop code
// -------------------------------------------------------------------------------------
```

Template Code



```
    li t0, GPIO_BASE_CTRL_ADDR          // Load Base Address for GPIO Control Registers
    li t1, GPIO_ALL_LED_PINS            // Load GPIO Register to set GPIO_OUTPUT_EN and
                                           GPIO_OUTPUT_XOR registers for all GPIO LED Pins
    sw t1, GPIO_OUTPUT_EN(t0)           // Enable outputs on all GPIO LED Pins
    li t2, 0xFF03F1C0
    sw t2, GPIO_OUTPUT_VAL(t0)          // Set all LED pins to zero to turn off all LEDS.

    li t5, GPIO_LEDBAR_LED_1     // Load LED_1 Address into t5 (Bit 11)
    lw t3, GPIO_OUTPUT_VAL(t0)   // Read GPIO output values into t3
    xor t3, t5, t3               // XOR the GPIO output value (all zeroes to the LEDs)
    sw t3, GPIO_OUTPUT_VAL(t0)   // Store this new value in GPIO output register

loop:
    li t4, 0x0000FFFF            // Load immediate 0x0000FFFF into t4 (counter value)
                                 // YOU NEED TO CALCULATE A CORRECT VALUE FOR 1 Hz
loop1:
//                               // You need to add 5 - 10 lines of code here
//                               // Add the instructions to get the LED to flash

    j loop                       // Jump back to loop and start over
```

Deliverables

- Lab report:
  - Your code, well documented.
  - Hand assembly of instruction: ADDI t4, t3, 0x100.
  - **Exact** 1 HZ frequency is not required.
  - Read your lab manual carefully and use the code skeleton supplied.
  - Report is submitted on Learn.
  - Make sure you understand your code for demo questions.