

Lab 1. Digital Logic on the TM4C123 (Spring 2021)

All students do Lab 1 by themselves (no partner for Lab 1)

[Preparation](#)

[Purpose](#)

[System Requirements](#)

[Procedure](#)

[Part a - Verify Keil Project for Lab1 is present and runs](#)

[Part b - Draw Flowchart](#)

[Part c - Write Pseudocode](#)

[Part d - Write Assembly](#)

[Demonstration](#)

[Deliverables](#)

[FAQ](#)

Preparation

Read Chapter 1 of the book

Step 0: Install and run Keil uVision 5 on your personal computer (TM4C123 not MSP432)

<http://edx-org-utaustinx.s3.amazonaws.com/UT601x/RTOSKeilInstall.htm>

Step 1: Download and run the installer for EE319K

http://users.ece.utexas.edu/~valvano/Volume1/EE319K_Install.exe (not ready yet)

Purpose

The general purpose of this laboratory is to familiarize you with the software development steps using the **uVision** simulator. Starting with Lab 3, we will use uVision for both simulation and debugging on the real board, but for Labs 1 and 2, we will just use just the simulator. You will learn how to perform digital input/output on parallel ports of the TM4C123. Software skills you will learn include port initialization, logic operations, and unconditional branching. Do not use any conditional branches in your solution. We want you to think of the solution in terms of logical and shift operations. Logical operations include AND ORR and EOR. Shift operations include LSL and LSR.

System Requirements

The objective of this system is to implement an even parity system. There are three bits of inputs and one bit of output. The output is in positive logic: outputting a 1 will turn on the LED, outputting a 0 will turn off the LED. Inputs are negative logic: meaning if the switch is not pressed the input is 1, if the switch is pressed the input is 0.

- PE0 is an input
- PE1 is an input
- PE2 is an input
- PE4 is the output

Even parity is an algorithm used in communication systems to detect errors during transmission. Consider the three inputs as a 3-bit data value, such that if the input switch is pressed, that data bit is 1. Your system will add one output bit, creating a 4-bit value, such that the number of ones, considered as one 4-bit value will always be even. The communication system (if there were one) sends the 4-bit value as a message (containing the 3-bit data plus parity), and the receiver could detect if one of the bits were to be flipped during transmission.

The specific operation of this system

- Initialize Port E to make PE0,PE1,PE2 inputs and PE4 an output
- Make the output 1 if there is an even number of switches pressed, otherwise make the output 0.
- Over and over, read the inputs, calculate the parity bit and set the parity bit at the output

The input data refers to the switch, not the input. The following table illustrates the expected behavior relative to output PE4 as a function of inputs PE0,PE1,PE2 (negative logic with respect to the switches).

PE2	PE1	PE0	PE4	
0	0	0	0	3 switches pressed, odd
0	0	1	1	2 switches pressed, even
0	1	0	1	2 switches pressed, even
0	1	1	0	1 switch pressed, odd
1	0	0	1	2 switches pressed, even
1	0	1	0	1 switch pressed, odd
1	1	0	0	1 switch pressed, odd
1	1	1	1	no switches pressed, even

There are 8 valid output values for Port E: 0x00,0x11,0x12,0x03,0x14,0x05,0x06, and 0x17. General rule, PE0,PE1,PE2,PE4 always have an even number of 1's (and an even number of 0's).

Procedure

The basic approach to this lab will be to develop and debug your system using the simulator. There is no automated grader for Lab 1. There is no hardware required for Lab 1.

Part a - Verify Keil Project for Lab1 is present and runs

To work on Lab 1, perform these tasks. Find a place on your hard drive to save all your TM4C123 software. In Figure 1 it is called **EE319KwareSpring2021**, created when you install the .exe. Download and unzip the starter projects. Notice the projects you use for the labs will be folders that begin with "Lab".

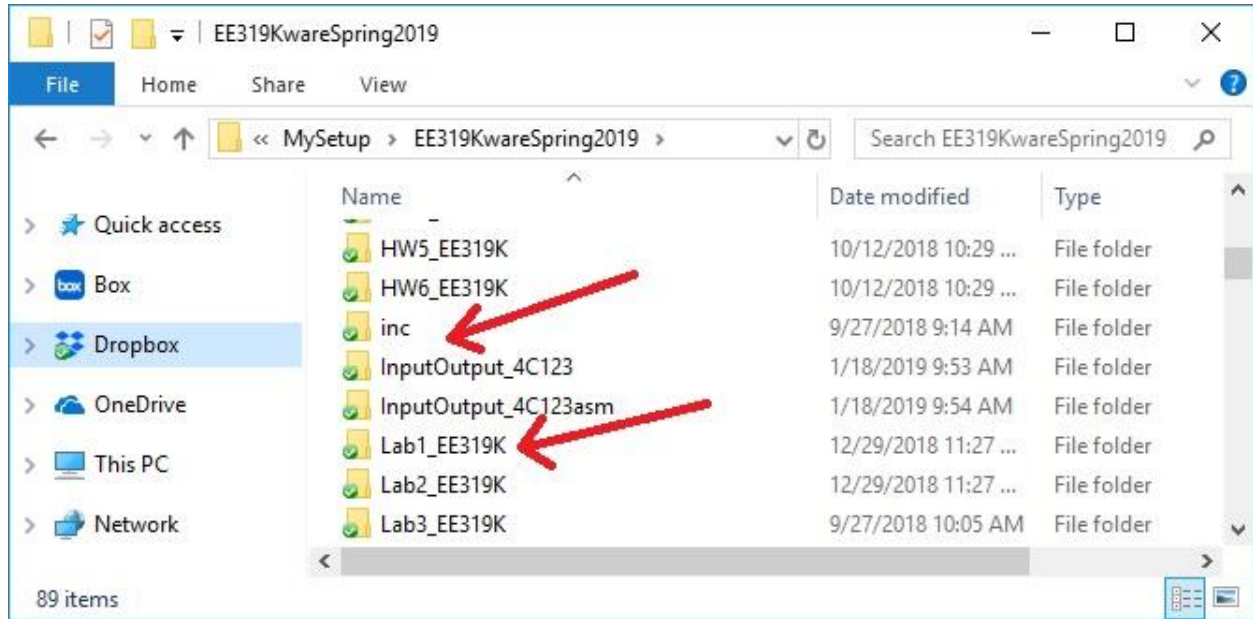


Figure 1. Directory structure with your Lab1.

It is important for the directory structure to look like Figure 1. Notice the directory relationship between the lab folders and the **inc** (include) folder. Begin with the **Lab1_EE319K** project in the folder **EE319KwareSpring2021**. Either double click the **uvprojx** file or open the project from within uVision. Make sure you can compile it and run on the simulator. Please contact your TA if the starter project does not compile or run on the simulator. **Startup.s** contains assembly source code to define the stack, reset vector, and interrupt vectors. All projects in this class will include this file, and you should not need to make any changes to the **Startup.s** file. **main.s** will contain your assembly source code for this lab. You will edit the **main.s** file.

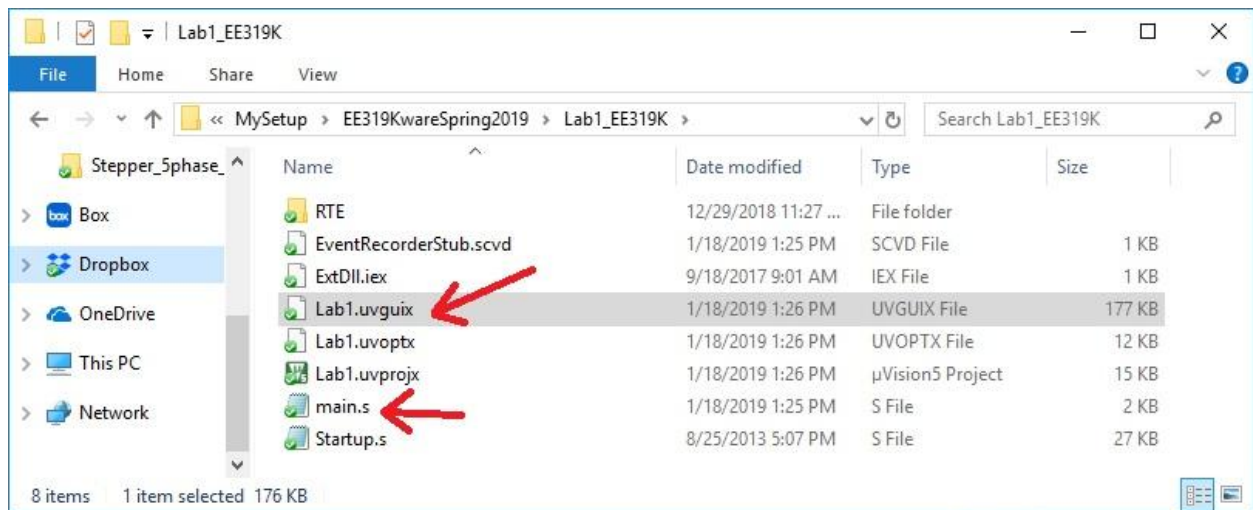


Figure 2. Start Keil by opening the **Lab1.uvprojx** file.

You should rename the Lab1 starter folder to include your EID. Add your names and the most recent date to the comments at the top of **main.s**. This code shows the basic template for the first few labs. You will not need global variables in lab 1.

```

        THUMB
        AREA    DATA, ALIGN=2
;global variables go here
        ALIGN
        AREA    |.text|, CODE, READONLY, ALIGN=2
        EXPORT  Start

Start
    ; initialization code goes here
loop
    ; put your main engine here
    B    loop
; put any subroutines here
        ALIGN    ; make sure the end of this section is aligned
        END      ; end of file

```

Program 1 Assembly language template.

To run the Lab 1 simulator, you must check two things. First, execute Project->Options and select the Debug tab. The debug parameter field must include **-dEE319KLab1**. Second, the **EE319KLab1.dll** file must be present in your Keil\ARM\BIN folder (the EE319K DLLs should have been put there by the installer).

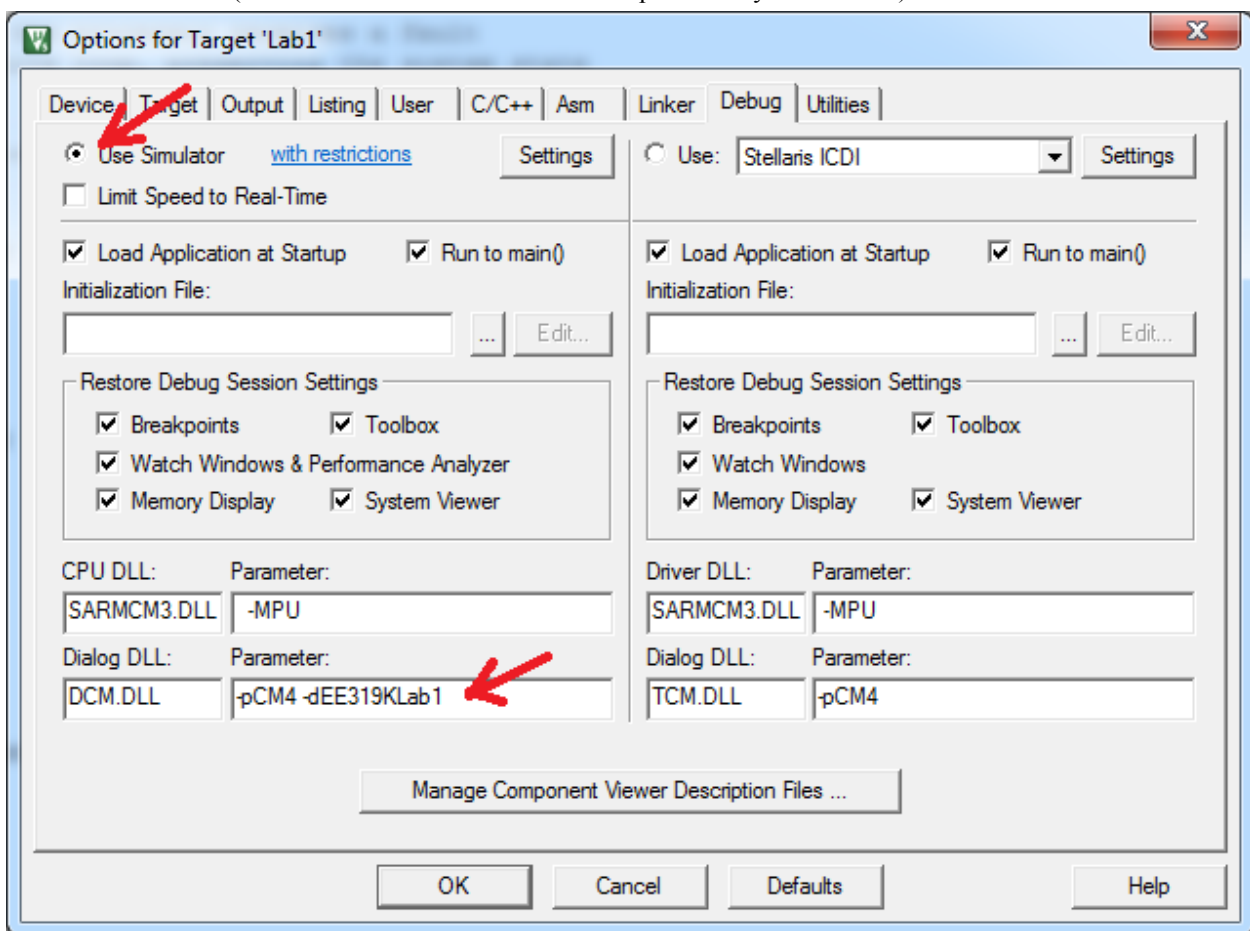


Figure 3. Debug the software using the simulator (DCM.DLL -pCM4 -dEE319KLab1).

Part b - Draw Flowchart

Write a flowchart for this program. We expect 5 to 15 symbols in the flowchart. A flowchart describes the algorithm used to solve the problem and is a visual equivalent of pseudocode. See Section 1.7 in the book for example flowcharts.

Part c - Write Pseudocode

Write pseudocode for this program. We expect 5 to 10 steps in the pseudocode. You may use any syntax you wish, but the algorithm should be clear. See Examples 1.14.1 and 1.14.2 in the book (Section 1.14) for an instance of what pseudocode ought to look like. Note, pseudocode ought to embody the algorithm and therefore be language blind. The same pseudocode can serve as an aid to writing the solution out in either assembly or C (or any other language).

Part d - Write Assembly

You will write assembly code that inputs from PE2, PE1, PE0 and outputs to PE4. The address definitions for Port E are listed below, and these are placed in the starter file **main.s**:

```
GPIO_PORTE_DATA_R EQU 0x400243FC
GPIO_PORTE_DIR_R EQU 0x40024400
GPIO_PORTE_DEN_R EQU 0x4002451C
SYSCTL_RCGCGPIO_R EQU 0x400FE608
```

The opening comments include: filename, overall objectives, hardware connections, specific functions, author name, TA, and date. The **equ** pseudo-op is used to define port addresses. Global variables are declared in RAM, and the main program is placed in EEPROM. The 32-bit contents at ROM address 0x00000004 define where the computer will begin execution after a power is turned on or after the reset button is pressed.

```
;***** main.s *****
; Program initially written by: Yerraballi and Valvano
; Author: Place your name here
; Date Created: 1/15/2018
; Last Modified: 12/23/2020
; Brief description of the program: Solution to Lab1
; The objective of this system is to implement odd-bit counting system
; Hardware connections:
; Output is positive logic, 1 turns on the LED, 0 turns off the LED
; Inputs are negative logic, meaning switch not pressed is 1, pressed is 0
;     PE0 is an input
;     PE1 is an input
;     PE2 is an input
;     PE4 is the output
; Overall goal:
;     Make the output 1 if there is an even number of switches pressed,
;     otherwise make the output 0
; The specific operation of this system
;     Initialize Port E to make PE0,PE1,PE2 inputs and PE4 an output
;     Over and over, read the inputs, calculate the result and set the output
; NOTE: Do not use any conditional branches in your solution.
;     We want you to think of the solution in terms of logical and shift operations
GPIO_PORTE_DATA_R EQU 0x400243FC
```

```

GPIO_PORTE_DIR_R    EQU 0x40024400
GPIO_PORTE_DEN_R    EQU 0x4002451C
SYSCTL_RCGCGPIO_R   EQU 0x400FE608
    THUMB
    AREA    DATA, ALIGN=2
;global variables go here
    ALIGN
    AREA    |.text|, CODE, READONLY, ALIGN=2
    EXPORT  Start

Start
    ;code to run once that initializes PE3,PE2,PE1,PE0

loop
    ;code that runs over and over
    B        loop
    ALIGN    ; make sure the end of this section is aligned
    END      ; end of file

```

Program 2. Required comments at the top of every file.

To interact with the I/O during simulation, **make sure that View->Periodic Window Update is checked or the simulator will not update!** Then, execute the **Peripherals->TEaS Port E** command. When running the simulator, we check and uncheck bits in the I/O Port box to change input pins. We observe the output pin in the window. You can also see the other registers, such as DIR DEN and RCGCGPIO.

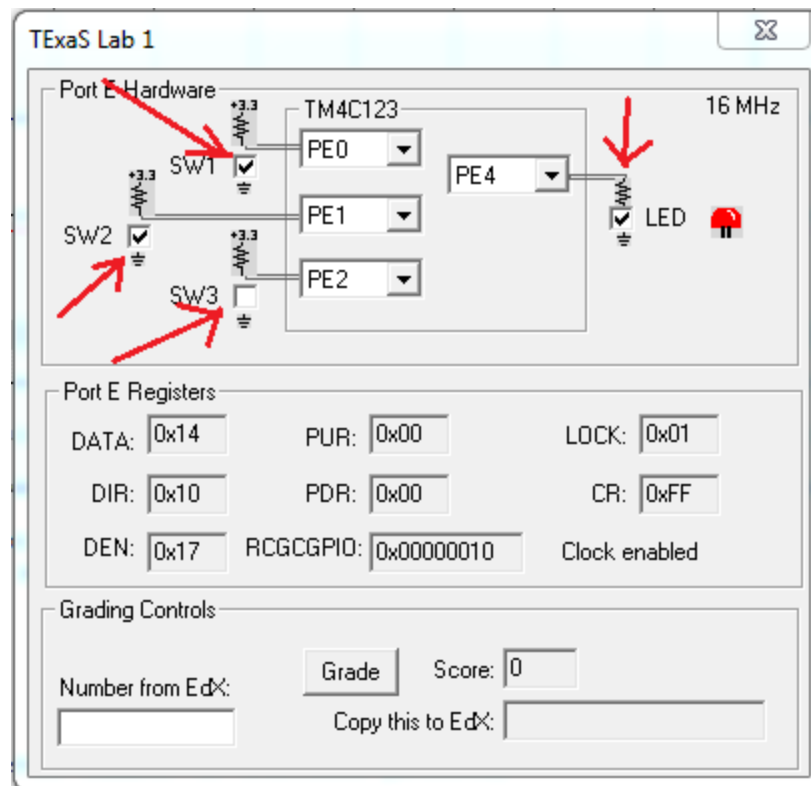


Figure 4. In simulation mode, we interact with virtual hardware. Notice SW1 and SW2 switches are pressed (i.e., two switches are pressed), so the output is high.

Demonstration

There are [grading sheets](#) for every lab so you know exactly how you will be evaluated. During the demonstration, you will be asked to run your program to verify proper operation. You should be able to single-step your program and explain what your program is doing and why. You need to know how to set and clear breakpoints. You should know how to visualize Port E input/output in the simulator.

Do all these well in advance of your checkout

1. **Signup for a Zoom time with a TA. All students do Lab 1 by themselves**
2. **If you cannot live-stream video, create a 60-sec YouTube video and upload it**
3. **Upload your software to canvas, make sure your name is on your software**
4. **Upload your one pdf with deliverables to Canvas**

Do all these during Zoom meeting

1. **Have your one pdf with deliverables open on your computer so it can be shared**
2. **Have Keil Lab 1 open so TA can ask about your code**
3. **Start promptly, because we are on a schedule.**
4. **Demonstrate lab to TA (YouTube video or livestream video)**
5. **Answer questions from TA in the usual manner**
6. **TA tells you your score (later the TA will upload scores to Canvas)**

Deliverables

Upload your **main.s** file to Canvas. Combine the following components into one pdf file and upload this file also to Canvas. Have the pdf file and Keil open on the computer during demonstration.

0. Your name, professor, and EID.
1. Flowchart of the system
2. Pseudocode for the algorithm
3. A screenshot of the Port E window, one showing the LED on (like Figure 4)
4. A screenshot of the Port E window, one showing the LED off (like Figure 4)

Optional Feedback : <http://goo.gl/forms/rBsP9NTxSy>

FAQ

The list of FAQ below are populated from Piazza over the semesters (thanks to the contributions of all past TAs and students). More questions may be posted so please check back regularly.

1. Should the program keep checking for inputs and update the LEDs continuously?
Your program should loop, so yes.
2. Our program works as expected when stepping through but when it is run through it does not. What could be causing this? What is an effective way to debug when our debugging method says that the program is working fine but the actual running of the program says otherwise?
First check if the "Periodic Window Update" under the "View" tab is on when you are in debugging mode. Also, some run-time errors can occur when setting up the clock register which don't appear when single stepping. Look over and ensure you are writing to the correct register and only affecting those bits required to activate Port E, as well as give enough time for it to start up.
3. What is the best way to include the source code for the main.s into the pdf file?

One way to include the source code for the main.s into the pdf file is to copy / paste the code into a Word document and then combine that with all of your other deliverables, depending on the lab, and then converting that file to a PDF. Please do not only include a screenshot.

4. For the flowchart and pseudocode, do we need to start at the very beginning with all the initialization tasks, or just at the actual logic for locking and unlocking the lock?
Your flowchart should cover the entire program that you write. Therefore, initialization should be included. How you represent initialization is up to you.
5. Are we allowed to branch?
Only unconditional branches are allowed in this lab. You are required to implement this lab using only Boolean logic and shift operations such as AND, OR, EOR, LSL and LSR.
6. Do both members of our lab group need to turn in a pdf, or do we just turn in one for the two of us?
Labs 1 and 2 are done individually. For Labs 3-10 both partners should submit the same pdf on Canvas.
7. When I try to run the debugger, I get: Error: Could not load file
'C:\Keil\EE319Kware\Lab1_EE319K\Lab1.axf'. Debugger aborted! What should I do?
You most likely forgot to build your project before running the debugger. The other possibility is the code has a syntax error. If that doesn't solve the problem, try running Keil as an administrator.
8. When I try to build, it gives the errors: Build target 'Lab1', error - cannot create command input file
'\startup_ia', error - cannot create command input file '\main_ia', Target not created! What should I do?
Try running Keil as an administrator
9. Where can I find the addresses for the different ports?
Register definitions for the microcontroller may be found here:
<http://users.ece.utexas.edu/~valvano/arm/tm4c123gh6pm.s>
10. I edited my code but nothing changes when I re-run it in the debugger!
If you made changes to your code "in debug mode", you most likely have not re-built your project and therefore your debugger is running the old version of your code. Exit out of debug mode and rebuild your code for the changes to take effect.
11. I cannot find the interactive UI simulator with switches and LEDs when I am in debugging mode!
Enable "TEaS Port E" under the "Peripherals" tab. If you do not see the Ports that you are expecting, you most likely do not have the correct DLL file listed in "Debug" tab in "Options for Target" under "Project" tab.
12. Keil tells me I have tons of build errors but nothing seems to be wrong with my assembly code.
Make sure you instructions are indented. Only the labels are aligned all the way to the left.