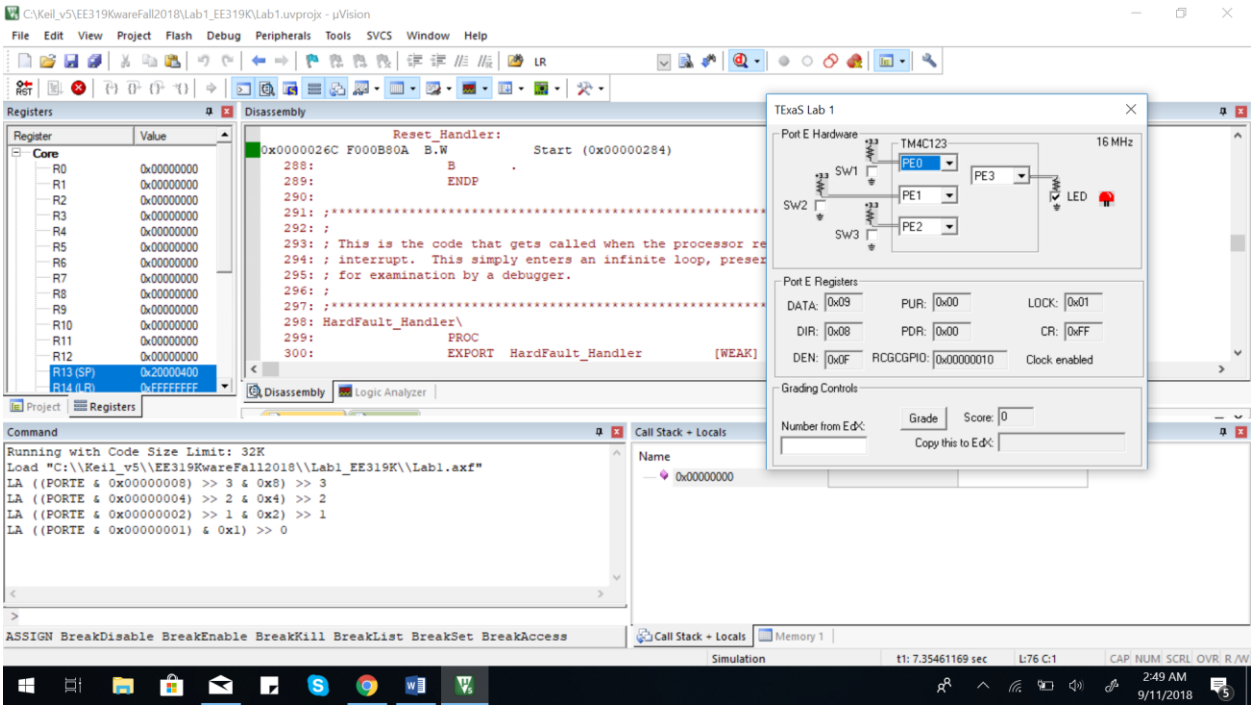
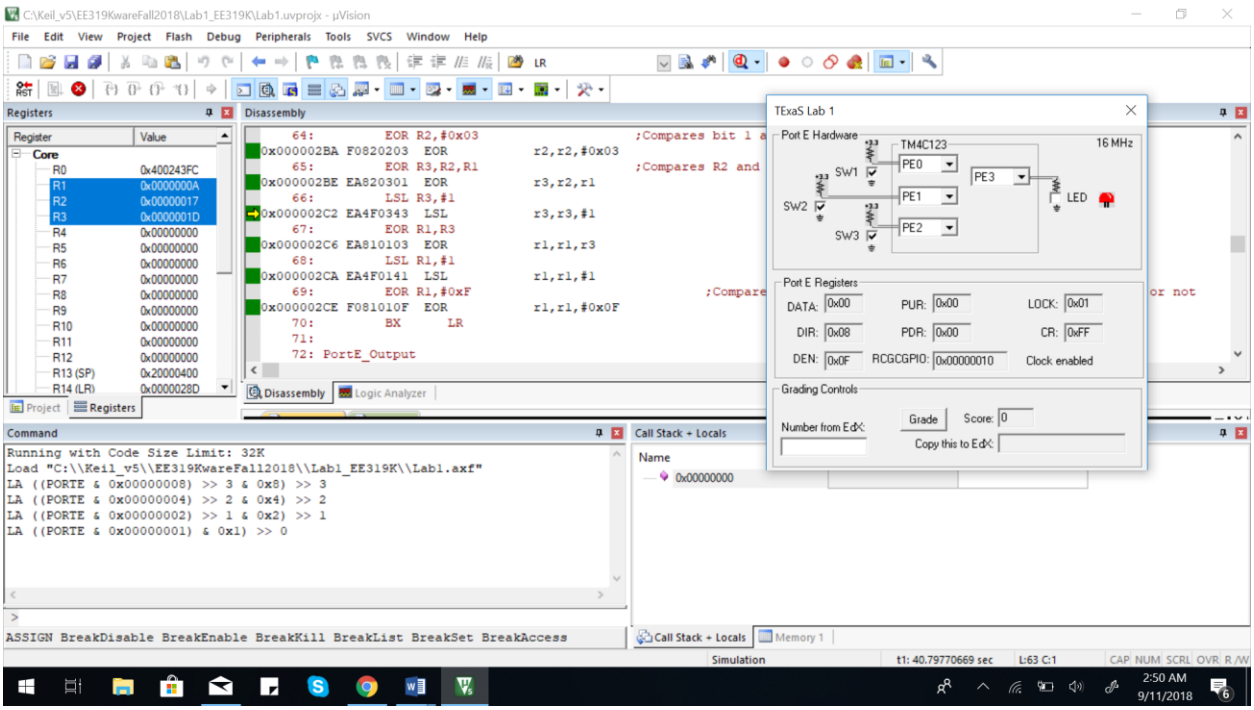
Screenshots





Assembly Code

```
;****************** main.s ***************
; Program initially written by: Yerraballi and Valvano
; Author: Eduardo Saul Ruiz
; Date Created: 1/15/2018
; Last Modified: 9/11/2018
; Brief description of the program: Fall 2018 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
;    PE3 is the output
; Overall goal:
;   Make the output 1 if there is an odd number of 1's at the inputs,
;     otherwise make the output 0

; The specific operation of this system
;   Initialize Port E to make PE0,PE1,PE2 inputs and PE3 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
   BL   PortE_Init
loop
         BL      PortE_Input

         BL      PortE_Output
   B    loop

PortE_Init
   LDR R0, =SYSCTL_RCGCGPIO_R          ;Initializes PTR
         LDR R1, [R0]                  ;Writes data inside PTR and loads it to R1
         ORR    R1, R1, #0x10          ;Set bit 4 to turn on clock 00010000
         STR    R1, [R0]                     ;Stores result into R0
         NOP                                       ;Stabalizing clock
         NOP
         LDR R0, =GPIO_PORTE_DIR_R     ;Initializes PTR
         MOV   R1,#0x08                       ;PE0 to PE2 are input, PE3 is output
         STR R1, [R0]                  ;Stores results into R0
```

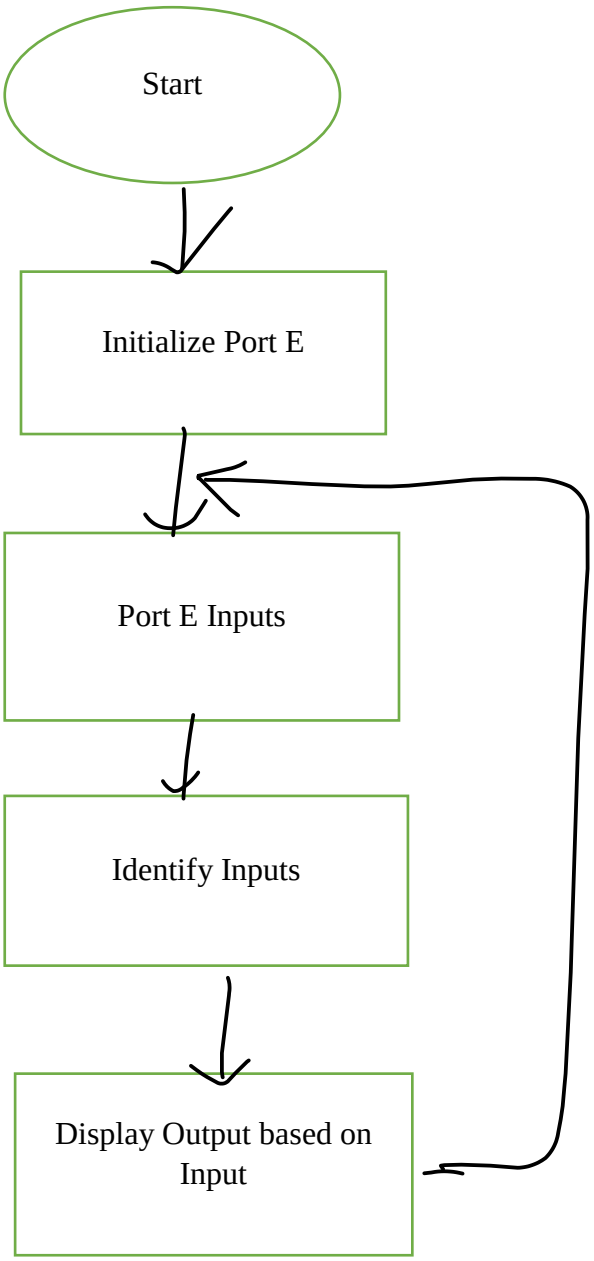
```
        LDR R0, =GPIO_PORTE_DEN_R       ;Enable Port E digital port
        MOV   R1, #0x0F                 ;Enable digital I/O R1<-00001111
        STR R1, [R0]                    ;Stores result into R0
        BX      LR                      ;BX means Return

PortE_Input
        LDR    R0, =GPIO_PORTE_DATA_R   ;Pointer to Port F data
        LDR R1, [R0]                    ;Read all of Port E
        LSL R2,R1,#1                    ;Moves bit 0 left one space in order to compare with
bit 1
        EOR R2,#0x03                    ;
        EOR R3,R2,R1                    ;Compares bit 1 and 0; 0000 0011
        LSL R3,#1                       ;Moves bit in order to compare
        EOR R1,R3                       ;Compares R1 and R3
        LSL R1,#1                       ;Moves bit in order to compare
        EOR R1,#0xF                     ;Compares R1 and 0000 1111 to tell
if LED should turn on or not
        BX      LR

PortE_Output
        STR R1,[R0]                     ;write Port E, sets PE3, enables LED on or
off depending on the truth table
        BX      LR                      ;Return

    ALIGN       ; make sure the end of this section is aligned
    END         ; end of file
```

Flowchart



Pseudocode