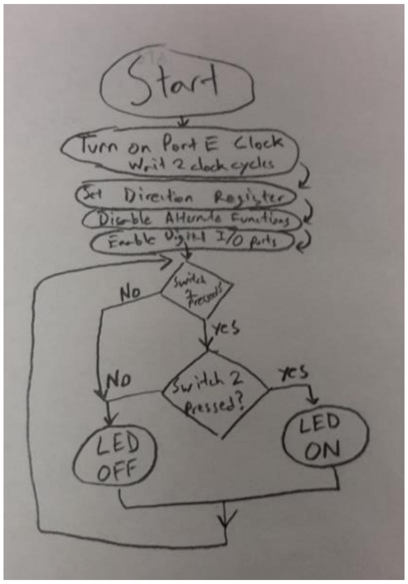
Flowchart



Pseudo Code

```
        Turn on Port E Clock and wait 2 clock cycles
        Set Direction Register (output PE2, input: PE3, PE4) 1=output 0=input
        Disable Alternate Functions of Registers
        Enable Digital I/o Port
Loop    Get PORTE DATA
        Test if PE3 is 0, else Branch to turnoff
        Test if PE4 is 0, else Branch to turnoff
        Turn ON PE2
        Branch to Loop
TurnOff Turn off PE2
        Branch to Loop
```

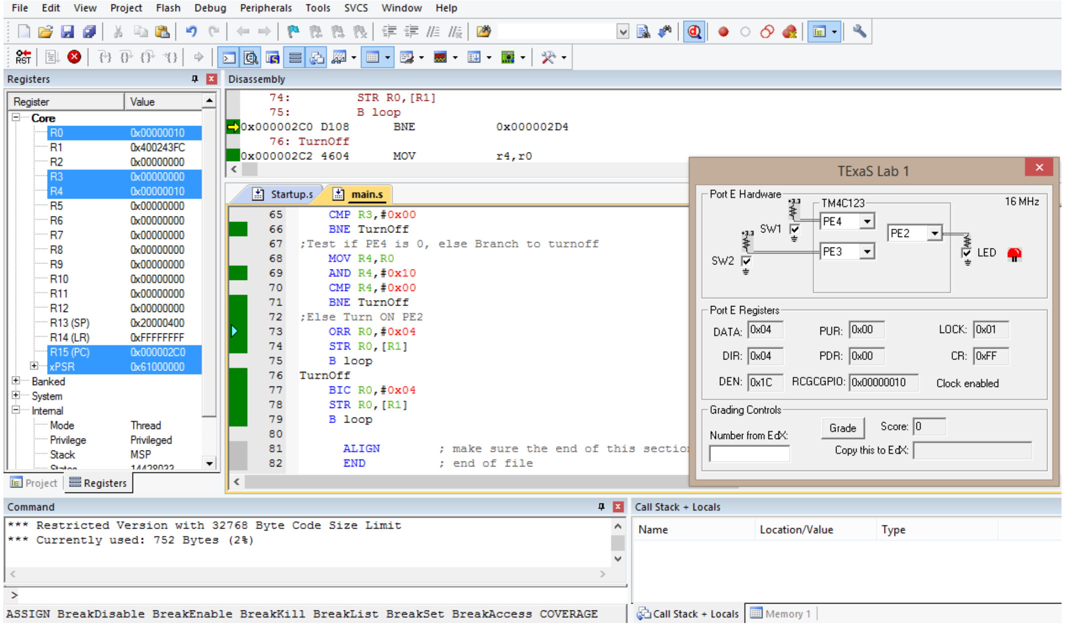
Main.S Program

```
;****************** main.s ***************
; Program written by: Kyle Sikora and Briar Coker
; Date Created: 1/24/2015
; Last Modified: 1/24/2015
; Section 1-2pm     TA: Wooseok Lee
; Lab number: 1
; Brief description of the program
; The overall objective of this system is a digital lock
; Hardware connections
;  PE3 is switch input  (1 means switch is not pressed, 0 means switch is pressed)
;  PE4 is switch input  (1 means switch is not pressed, 0 means switch is pressed)
;  PE2 is LED output (0 means door is locked, 1 means door is unlocked)
; The specific operation of this system is to
;   unlock if both switches are pressed

GPIO_PORTE_DATA_R       EQU   0x400243FC
GPIO_PORTE_DIR_R        EQU   0x40024400
```

```
GPIO_PORTE_AFSEL_R      EQU   0x40024420
GPIO_PORTE_DEN_R        EQU   0x4002451C
GPIO_PORTE_AMSEL_R      EQU   0x40024528
GPIO_PORTE_PCTL_R       EQU   0x4002452C
SYSCTL_RCGCGPIO_R       EQU   0x400FE608
SYSCTL_RCGC2_R          EQU   0x400FE108

    AREA    |.text|, CODE, READONLY, ALIGN=2
    THUMB
    EXPORT  Start
Start
        ;PE2 = 0 off, 1 ON
        ;PE3 = input switch = 1 notpress, 0 pressed
        ;PE4 = input switch = 1 notpress, 0 pressed
        ;PE2 = 1 if and only if PE3 = 0 && PE4 = 0
;Turn on Port E Clock and wait 2 clock cycles
        LDR R1,= SYSCTL_RCGCGPIO_R
        LDR R0,[R1]
        ORR R0,R0,#0x10
        STR R0,[R1]
        NOP
        NOP
;Set Direction Register (output PE2, input: PE3, PE4) 1=output 0=input
        LDR R1,=GPIO_PORTE_DIR_R
        LDR R0,[R1]
        ORR R0,#0x04
        BIC R0,#0x18
        STR R0,[R1]
;Disable Alternate Functions of Registers
        LDR R1,= GPIO_PORTE_AFSEL_R
        LDR R0, [R1]
        BIC R0,#0x1C
        STR R0,[R1]
;Enable Digital I/o Port
        LDR R1,=GPIO_PORTE_DEN_R
        LDR R0,[R1]
        ORR R0,#0x1C
        STR R0,[R1]

;R1 = PORTE data address location
        LDR R1,=GPIO_PORTE_DATA_R
loop
;Get PORTE DATA
        LDR R0,[R1]
;Test if PE3 is 0, else Branch to loop
        MOV R3,R0
        AND R3,#0x08
        CMP R3,#0x00
        BNE TurnOff
;Test if PE4 is 0, else Branch to loop
        MOV R4,R0
        AND R4,#0x10
        CMP R4,#0x00
        BNE TurnOff
;Else Turn ON PE2
        ORR R0,#0x04
        STR R0,[R1]
        B loop
```

```
TurnOff
        BIC R0,#0x04
        STR R0,[R1]
        B loop

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

Screenshot of Picture



Register	Value
Core	
R0	0x00000010
R1	0x400243FC
R2	0x00000000
R3	0x00000000
R4	0x00000010
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000400
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x000002C0
xPSR	0x61000000
Banked	
System	
Internal	
Mode	Thread
Privilege	Privileged
Stack	MSP

Disassembly

```
    74:          STR R0,[R1]
    75:          B loop
0x000002C0 D108      BNE          0x000002D4
    76: TurnOff
0x000002C2 4604      MOV          r4,r0
```

main.s

```
    CMP R3,#0x00
    BNE TurnOff
;Test if PE4 is 0, else Branch to turnoff
    MOV R4,R0
    AND R4,#0x10
    CMP R4,#0x00
    BNE TurnOff
;Else Turn ON PE2
    ORR R0,#0x04
    STR R0,[R1]
    B loop
TurnOff
    BIC R0,#0x04
    STR R0,[R1]
    B loop

      ALIGN       ; make sure the end of this sectio
      END         ; end of file
```

TExaS Lab 1

Port E Hardware: SW1 ☑ SW2 ☑ LED ☑

Port E Registers

DATA: 0x04　PUR: 0x00　LOCK: 0x01

DIR: 0x04　PDR: 0x00　CR: 0xFF

DEN: 0x1C　RCGCGPIO: 0x00000010　Clock enabled

Grading Controls

Number from EdX:

Score: 0

Copy this to EdX:

Command

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 752 Bytes (2%)
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE
```

Call Stack + Locals

Name	Location/Value	Type