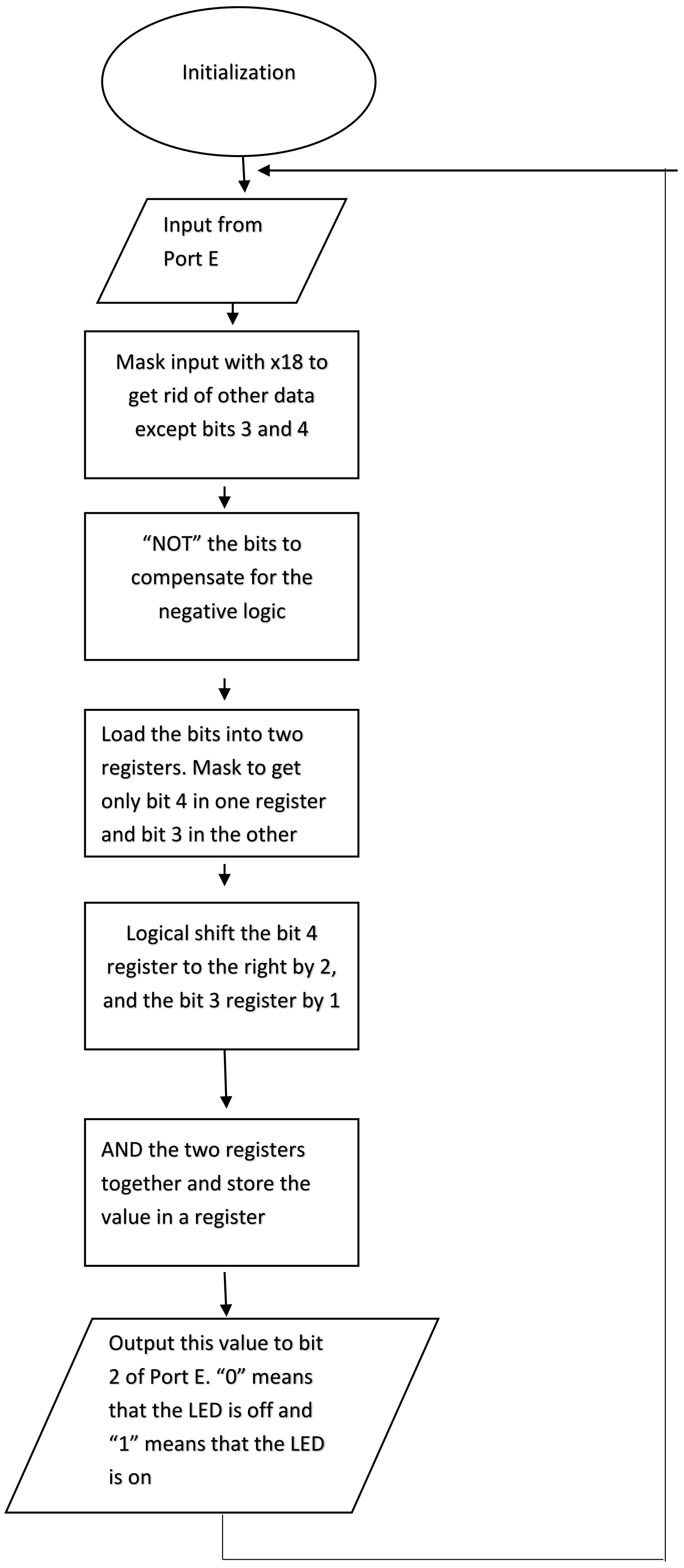
Lab 1 Flowchart


Initialization
Input from Port E
Mask input with x18 to get rid of other data except bits 3 and 4
"NOT" the bits to compensate for the negative logic
Load the bits into two registers. Mask to get only bit 4 in one register and bit 3 in the other
Logical shift the bit 4 register to the right by 2, and the bit 3 register by 1
AND the two registers together and store the value in a register
Output this value to bit 2 of Port E. "0" means that the LED is off and "1" means that the LED is on

Kassandra Smith and Madhumitha Venkataraman
Psuedo Code

```
start
      {
      Initialize the clock
      Stabilize
      Set Digital E port (DEN) so that all bits are 1, aka useable
      Set all bits in the Direction Register (DIR) to 0 except for the LED
bit (bit 3)
      Set special funtions to 0
      Disable analog functionality
      Configure Port E as GPIO
      }


while (true)
            {
      Read the switch data
      Not the switch data
      Get the two switch bits (3&4) individually
      Shift the bits so that their data are both in bit 2
      AND the two bits
      Put the result into the LED bit
            }
```

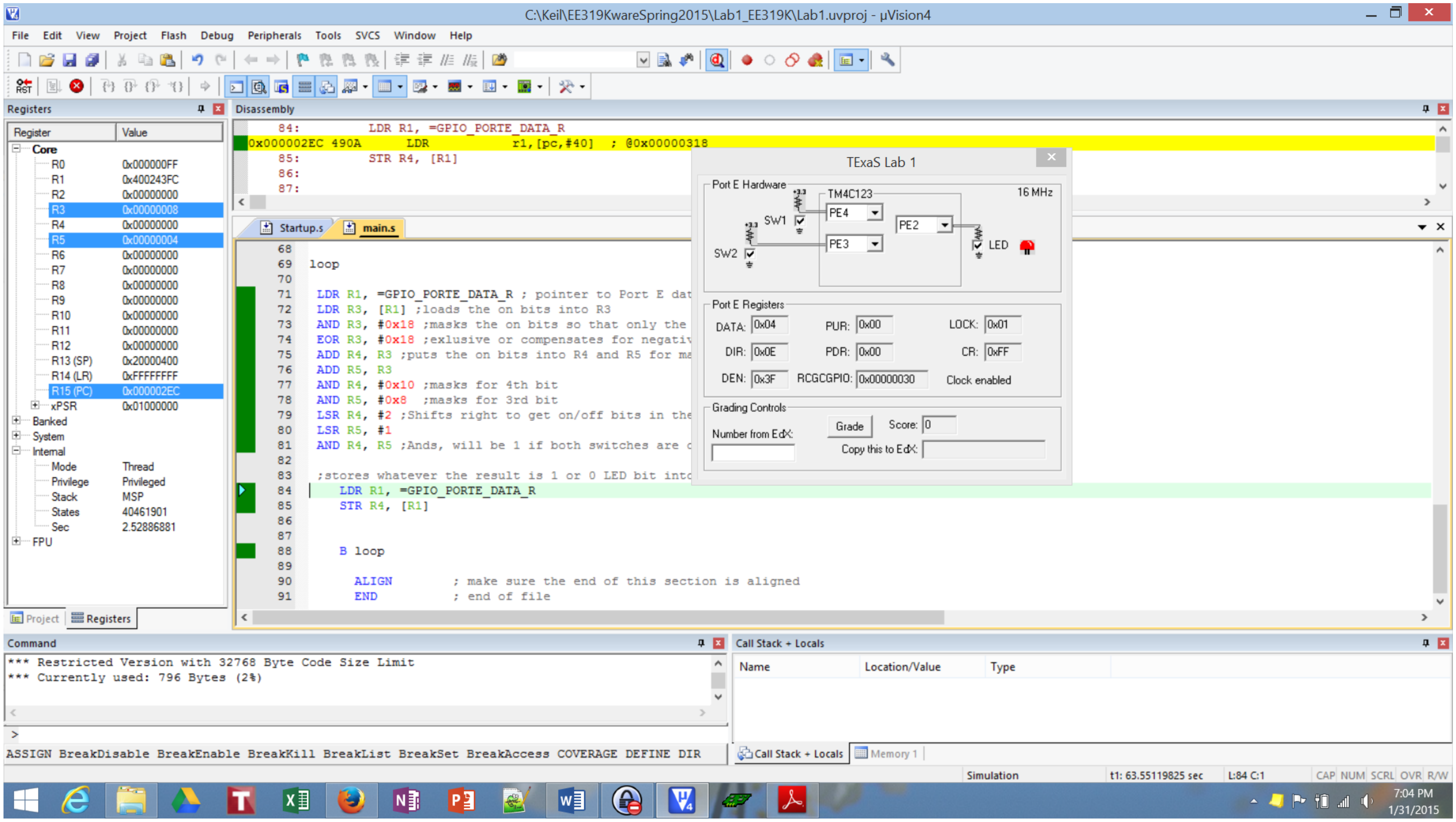
A C version of the code within the while loop

```
while (true)
            {
      int *portDataPointer = GPIO_PORTE_DATA_R;
      int portData = *portDataPointer;
      int Data= portData & 0x18 ; //mask
      int NotData= ~(Data)
      int BitFour = NotData & 0x10
      int BitThree = NotData & 0x08
      int ThreeShifted = BitThree >>1
      int FourShifted = BitFour >> 2
      int LedState = ThreeShifted & FourShifted
      int newData = (portData & mask) | LedState
      *portDataPointer = newData
            }
```

```
;****************** main.s ***************
; Program written by: Kassandra Smith and Madhumitha Venkataraman
; Date Created: 1/24/2015
; Last Modified: 1/31/2015
; Section 4-5pm    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


;EQU “#”, “#” is the location where the GPIO stuff is located

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

;LEDBit=0x04
;SwitchBits=0x18

;An AREA statement defines ARM-compatible segment
;ALIGN=2, 2^2=4, 4bytes=32 bits, makes the code word-aligned
;Thumb is the instruction set being used

        AREA      |.text|, CODE, READONLY, ALIGN=2
        THUMB
        EXPORT  Start
```

```
Start
      ;Initialize the clock,
      LDR R1, =SYSCTL_RCGCGPIO_R      ;activate clock
      LDR R0, [R1]
      ORR R0, R0, #0x10                ; set bit 5 to turn on clock
      STR R0, [R1]                              ; put it back
      NOP
      NOP                                        ;stabilize

      ;configure the rest of the port
      ;p. 354

      ;Set DEN so that the bits are useable
      LDR R1, =GPIO_PORTE_DEN_R        ;Port E digital port
      MOV R0, #0xFF                          ; 1 enables digital I/O
      STR R0, [R1]

      ;since we want the two switches to be inputs and the led to be an
output we make all bits 0 except the LEDbit
      LDR R1, =GPIO_PORTE_DIR_R        ;direction register
      MOV R0,#0x04                              ; PortE bit 2 is set to 1
      STR R0, [R1]

      ;We don't need the pins' special funtions so we set it to 0
      LDR R1, =GPIO_PORTE_AFSEL_R    ; alternate function select
      MOV R0, #0                            ; 0, alternate function is off
      STR R0, [R1]

      ;We want to disable analog functionality
      LDR R1, =GPIO_PORTE_AMSEL_R    ;analog functionality
      MOV R0, #0                            ; 0, analog is off
      STR R0, [R1]

      ;none of the pins (p.146 Table 4.5) need to be turned on
      LDR R1, =GPIO_PORTE_PCTL_R      ; configure as GPIO
      MOV R0, #0                ; 0 means configure Port E as GPIO
      STR R0, [R1]
```

```
loop
 AND R1, #0x0
 LDR R1, =GPIO_PORTE_DATA_R ; pointer to Port E data
 LDR R3, [R1] ;loads the on bits into R3
 AND R3, #0x18 ;masks the on bits so that only the two switches show
 EOR R3, #0x18 ;exlusive or compensates for negative logic
 ADD R4, R3 ;puts the on bits into R4 and R5 for mask
 ADD R5, R3
 AND R4, #0x10 ;masks for 4th bit
 AND R5, #0x8  ;masks for 3rd bit
 LSR R4, #2 ;Shifts right to get on/off bits in the same place as the LED bit
 LSR R5, #1
 AND R4, R5 ;Ands, will be 1 if both switches are on, 0 in any other instance

 ;stores whatever the result is 1 or 0 LED bit into the machine
      LDR R1, =GPIO_PORTE_DATA_R
      STR R4, [R1]

      B loop

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



Register	Value
Core	
R0	0x000000FF
R1	0x400243FC
R2	0x00000000
R3	0x00000008
R4	0x00000000
R5	0x00000004
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000400
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x000002EC
xPSR	0x01000000
Banked	
System	
Internal	
Mode	Thread
Privilege	Privileged
Stack	MSP
States	40461901
Sec	2.52886881
FPU	

```
    84:           LDR R1, =GPIO_PORTE_DATA_R
0x000002EC 490A      LDR          r1,[pc,#40]  ; @0x00000318
    85:           STR R4, [R1]
    86:
    87:
```

```
loop

 LDR R1, =GPIO_PORTE_DATA_R ; pointer to Port E dat
 LDR R3, [R1] ;loads the on bits into R3
 AND R3, #0x18 ;masks the on bits so that only the
 EOR R3, #0x18 ;exlusive or compensates for negativ
 ADD R4, R3 ;puts the on bits into R4 and R5 for ma
 ADD R5, R3
 AND R4, #0x10 ;masks for 4th bit
 AND R5, #0x8  ;masks for 3rd bit
 LSR R4, #2 ;Shifts right to get on/off bits in the
 LSR R5, #1
 AND R4, R5 ;Ands, will be 1 if both switches are c

 ;stores whatever the result is 1 or 0 LED bit into
    LDR R1, =GPIO_PORTE_DATA_R
    STR R4, [R1]


    B loop

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

DATA:	0x04	PUR:	0x00	LOCK:	0x01
DIR:	0x0E	PDR:	0x00	CR:	0xFF
DEN:	0x3F	RCGCGPIO:	0x00000030	Clock enabled	

Number from EdX:
Score: 0
Copy this to EdX:

```
*** Restricted Version with 32768 Byte Code Size Limit
*** Currently used: 796 Bytes (2%)
>
ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE DEFINE DIR
```

Name	Location/Value	Type

t1: 63.55119825 sec L:84 C:1