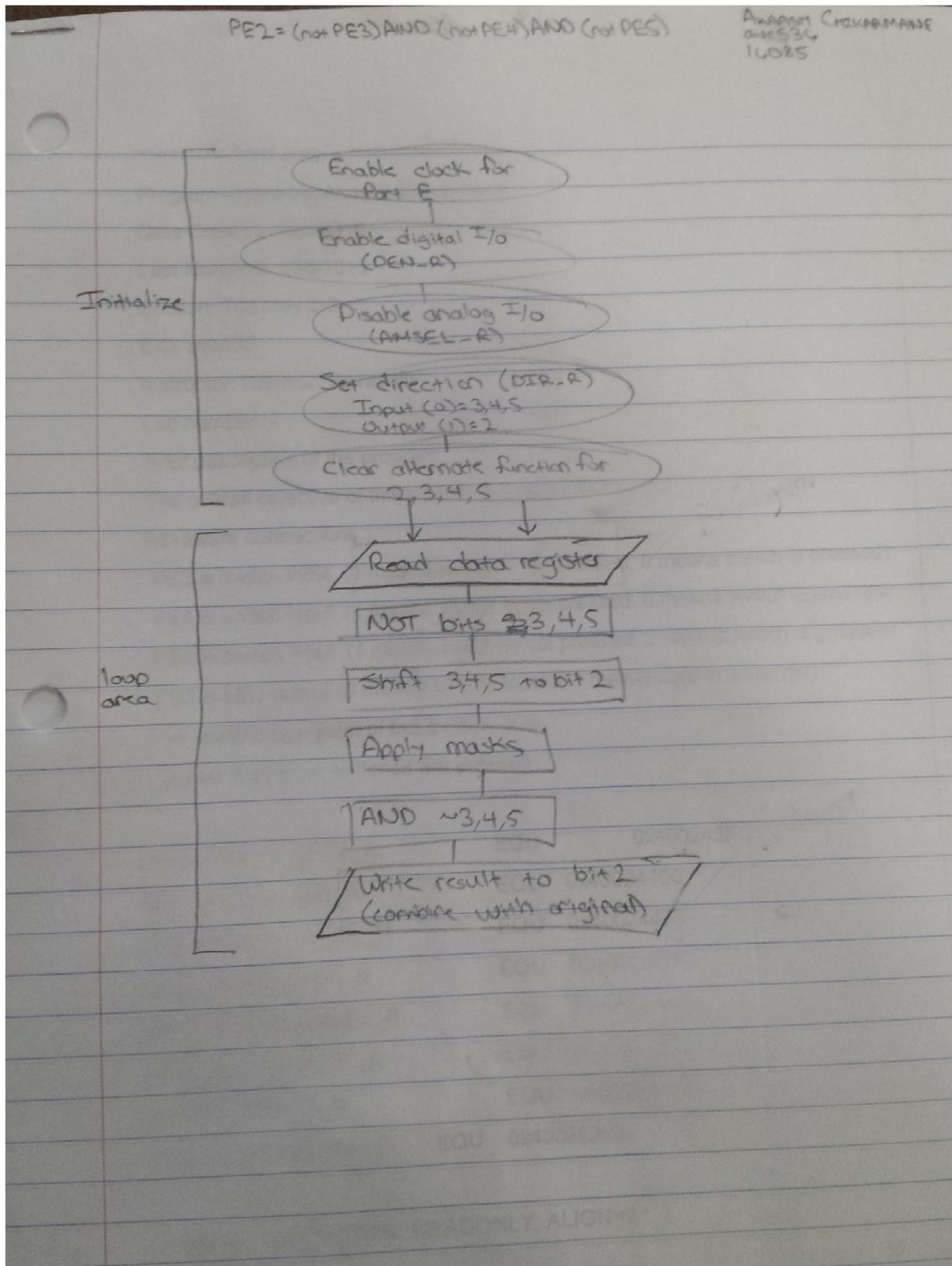


Flowchart



Pseudo-Code

```
;R0 = clock  
;read clock into R1  
;turn on clock E (0001 0000)  
;write back in  
;stabilize clock  
  
;enable digital I/O  
;digital for 2,3,4,5 (0011 1100)  
  
;disable analog I/O (not alternate functions)  
;complement of the above one (1100 0011)  
  
;changing the direction  
;PE2 is output (0000 0100)  
;PE3,4,5 is input (0011 1000)  
  
;alternate functions  
;clear AF for PE5,PE4 (0011 1100)  
  
;Copy the data. Modifying R1 then combining with original (R2).  
;R1 for bit 3  
;R3 for bit 4  
;R4 for bit 5  
  
;Move 3 to bit 2 and mask  
;Move 4 to bit 2 and mask  
;Move 5 to bit 2 and mask  
  
;Absorb(AND) bit 4  
;Absorb(AND) bit 5  
;Clear the original bit 2  
;Combine with original  
;write back in  
  
; make sure the end of this section is aligned  
; end of file
```

Main Program

```
;***** main.s *****
; Program written by: Akaash Chikarmane
; Date Created: 1/22/2016
; Last Modified: 1/30/2016
; Section: Tuesday 2-3
; EID: avc536
; Instructor: Ramesh Yerraballi
; 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)
; PE5 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 all three 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_RCGC2_R          EQU      0x400FE108
SYSCTL_RCGCGPIO_R     EQU      0x400FE608

AREA  [.text], CODE, READONLY, ALIGN=2
THUMB
EXPORT Start
Start
    LDR R0,=SYSCTL_RCGCGPIO_R ;R0 = clock
    LDR R1,[R0]                 ;read clock into R1
    ORR R1,#0x10                ;turn on clock E (0001 0000)
    STR R1,[R0]                 ;write back in
    NOP                         ;stabilize clock
    NOP

    LDR R0,=GPIO_PORTE_DEN_R   ;enable digital I/O
    LDR R1, [R0]
    ORR R1, #0x3C                ;digital for 2,3,4,5 (0011 1100)
    STR R1, [R0]

    LDR R0,=GPIO_PORTE_AMSEL_R ;disable analog I/O (not alternate functions)
    LDR R1, [R0]
```

```

AND R1, #0xC3          ;complement of the above one (1100 0011)
STR R1, [R0]

LDR R0,=GPIO_PORTE_DIR_R ;changing the direction
LDR R1,[R0]
ORR R1,#0x04           ;PE2 is output (0000 0100)
BIC R1,#0x38           ;PE3,4,5 is input (0011 1000)
STR R1, [R0]

LDR R0,=GPIO_PORTE_AFSEL_R ;alternate functions
LDR R1,[R0]
BIC R1,#0x3C           ;clear AF for PE5,PE4 (0011 1100)
STR R1,[R0]

loop
    LDR R0,=GPIO_PORTE_DATA_R
    LDR R1,[R0]
    MOV R2, R1             ;Copy the data. Modifying R1 then combining
with original (R2).
    EOR R1, R1, #0x3C      ;R1 for bit 3
    MOV R3, R1              ;R3 for bit 4
    MOV R4, R1              ;R4 for bit 5

    LSR R1, #1              ;Move 3 to bit 2 and mask
    AND R1, #0x04
    LSR R3, #2              ;Move 4 to bit 2 and mask
    AND R3, #0x04
    LSR R4, #3              ;Move 5 to bit 2 and mask
    AND R4, #0x04

    AND R1, R1, R3          ;Absorb(AND) bit 4
    AND R1, R1, R4          ;Absorb(AND) bit 5
    BIC R2, R2, #0x04      ;Clear the original bit 2
    ORR R1, R1, R2          ;Combine with original
    STR R1,[R0]             ;write back in
    B loop

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

```

Screenshot of LED on

File Edit View Project Flash Debug Peripherals Tools S/CS Window Help

Registers Core R0 0x400243FC R1 0x00000008 R2 0x00000008 R3 0x00000004 R4 0x00000004

Disassembly

```

60: LDR R0,=GPIO_PORTE_DATA_R
61: LDR R1,[R0]
62: MOV R2, R1 ;Copy the data. Modifying R1 then combining with original (R2).
      ;Copy the data. Modifying R1 then combining with original (R2).

52 BIC R1,#0x38 ; PE3,4,5 is input
53 STR R1, [R0]
54 LDR R0,=GPIO_PORTE_AFSEL_R ; alternate functions
55 LDR R1,[R0]
56 BIC R1,#0x3C ;clear AF for PE5,PE4
57 STR R1,[R0]
58 LDR R0,=GPIO_PORTE_DATA_R
59 loop
60 LDR R0,=GPIO_PORTE_DATA_R
61 LDR R1,[R0] ;Copy the data. Modifying R1 then combining with
62 MOV R2, R1 ;R1 for bit 3
63 EOR R1, R1, #0x3C ;R3 for bit 4
64 MOV R3, R1 ;R4 for bit 5
65 MOV R4, R1
66 LSR R1, #1 ;Move 3 to bit 2 and mask
67 AND R1, #0x04 ;Move 4 to bit 2 and mask
68 LSR R3, #2 ;Move 5 to bit 2 and mask
69 AND R1, #0x04 ;Clear the original bit 2
70 AND R3, #0x04 ;Combine with original
71 LSR R4, #3
72 AND R4, #0x04
73 AND R1, R1, R3 ;absorb (ANI) bit 4
74 AND R1, R1, R4 ;absorb (AND) bit 5
75 BIC R2, R2, #0x04 ;Clear the original bit 2
76 ORR R1, R1, R2 ;Combine with original
77 STM R1, R2, ! ;Move 3 to bit 2 and mask
      ;Copy the data. Modifying R1 then combining with original (R2).
      ;Copy the data. Modifying R1 then combining with original (R2).

```

TeaS Lab 1

Port E Hardware

Port E Registers

| | | | | |
|------|----|--------------------|---------------|------|
| Data | R0 | 0x00 | LOCK | 0x01 |
| Dir | R0 | 0x00 | CR | 0xFF |
| DEN | R0 | REGPORT 0x00000010 | Clock enabled | |

Grading Controls

| | | | |
|-------------------|-------|-------|---|
| Number from EDK | Grade | Score | 0 |
| Copy this to EDK: | | | |

Project Registers

Call Stack + Locals

Command

*** Restricted Version with 32768 Byte Code Size Limit

*** Currently used: 784 Bytes (2%)

Memory 1 | Memory 2 | Call Stack + Locals | Local

Assignment Breakable Breakpoint BreakList BreakSet Coverage Define DIR

Simulation t1:8.9094819 sec L60 C1 CAP NUM SERL OVR R/W