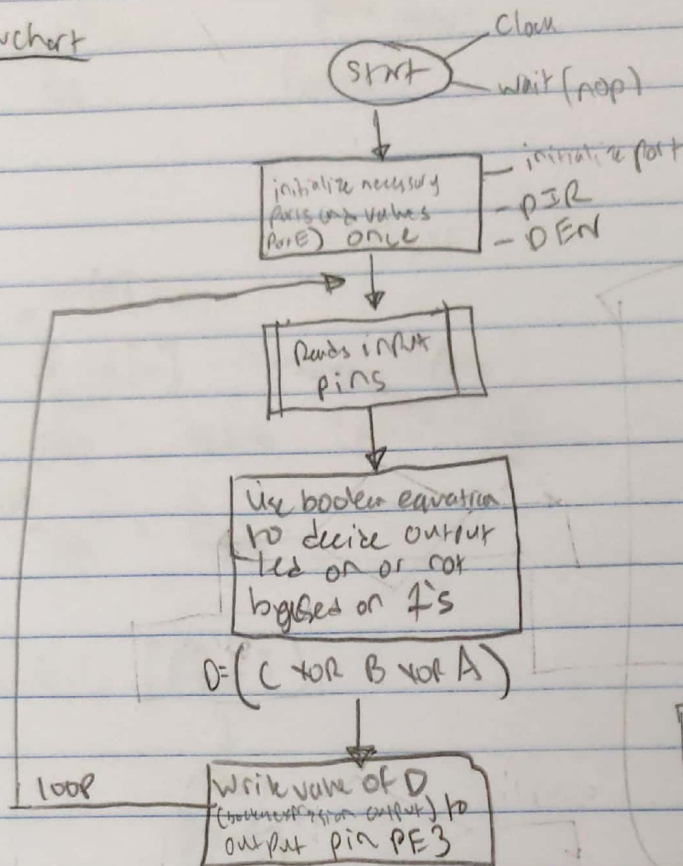


Lab 1

Flowchart



Boolean Algebra Equation

C	B	A	D
PE2	PE1	PE0	PE3

$$\bar{C}\bar{B}A + \bar{C}B\bar{A} + C\bar{B}\bar{A} + CBA = D$$

$$D = \bar{C}(\bar{B}A + B\bar{A}) + C(\bar{B}\bar{A} + BA)$$

$$= \bar{C}(B \text{ XOR } A) + C(B \text{ XOR } A)$$

$$D = C \text{ XOR } B \text{ XOR } A$$

Pseudocode

; First need to initialize ports

; Start Clock

; Load DIR to register

; Set DIR to 1000, to ^{Set} ~~input~~ PE3 as output and PE0-2 as input

; Set the DEN to 1 to enable input/output

; Now begin loop

; Use DATA register to get input values from PE0-2

; Use a mask on the DATA register to get the first four bits separately

; Use the boolean expression found above to compute the bit for the

; output (PE0 XOR PE1 XOR PE2 = PE3)

; Write to the output bit

; loop

```

1  ;***** main.s *****
2  ; Program initially written by: Yerraballi and Valvano
3  ; Author: Vikram Pandian
4  ; Date Created: 1/15/2018
5  ; Last Modified: 1/18/2019
6  ; Brief description of the program: Spring 2019 Lab1
7  ; The objective of this system is to implement odd-bit counting system
8  ; Hardware connections:
9  ; Output is positive logic, 1 turns on the LED, 0 turns off the LED
10 ; Inputs are negative logic, meaning switch not pressed is 1, pressed is 0
11 ; PE0 is an input
12 ; PE1 is an input
13 ; PE2 is an input
14 ; PE3 is the output
15 ; Overall goal:
16 ; Make the output 1 if there is an odd number of 1's at the inputs,
17 ; otherwise make the output 0
18 ; The specific operation of this system
19 ; Initialize Port E to make PE0,PE1,PE2 inputs and PE3 an output
20 ; Over and over, read the inputs, calculate the result and set the output
21
22 ; NOTE: Do not use any conditional branches in your solution.
23 ; We want you to think of the solution in terms of logical and shift operations
24
25 GPIO_PORTE_DATA_R EQU 0x400243FC
26 GPIO_PORTE_DIR_R EQU 0x40024400
27 GPIO_PORTE_DEN_R EQU 0x4002451C
28 SYSCCTL_RCGCGPIO_R EQU 0x400FE608
29
30 THUMB
31 AREA DATA, ALIGN=2
32 ;global variables go here
33 ALIGN
34 AREA |.text|, CODE, READONLY, ALIGN=2
35 EXPORT Start
36
37 Start
38 ; code to execute once at start goes here
39 LDR R1, =SYSCCTL_RCGCGPIO_R ;activates clock for port E
40 LDR R0, [R1]
41 ORR R0, R0, #0x10 ; set bit 5 to turn on clock
42 STR R0, [R1]
43 NOP
44 LDR R1, =GPIO_PORTE_DIR_R ;set DIR register
45 LDR R0, [R1] ;need to set a 1 for output and 0 for input
46 AND R0, R0, #0xF8
47 ORR R0, R0, #0x8 ;this should make PE0-2 inputs and PE3 an output
48 STR R0, [R1]
49
50 LDR R1, =GPIO_PORTE_DEN_R ;sets the enable register
51 LDR R0, [R1] ;loads to register R0
52 ORR R0, R0, #0x0F ; sets to 1 which means enable
53 STR R0, [R1]
54
55
56 loop
57 ; code that runs over and over goes here
58
59 LDR R2, =GPIO_PORTE_DATA_R ;R2 now has data register address
60 LDR R3, [R2] ;R3 now has the contents of the data register
61 ;Use masks/shifts to find the individual bits of PE2-0
62 ;Then XOR them to find the output PE3 bit value
63
64 ;Initialize the masks
65 AND R6, R6, #0
66 ADD R6, R6, #1 ;mask for PE0
67
68 AND R7, R7, #0
69 ADD R7, R7, #0x02 ;mask for PE1
70
71 AND R8, R8, #0
72 ADD R8, R8, #0x04 ;mask for PE2

```

```
73
74     ;Now use the masks
75     AND R6, R3, R6    ;has PE0 val
76     AND R7, R3, R7    ;has PE1 val
77     AND R8, R3, R8    ;has PE2 val
78
79     ;need to shift the PE1 and PE2 bits to LSB
80     LSR R7, R7, #1
81     LSR R8, R8, #2
82
83     EOR R6, R6, R7
84     EOR R6, R6, R8 ;R6 should now have the value PE0 xor PE1 xor PE2 at its LSB
85
86     LSL R6, R6, #3    ;shifts bit to the 4th bit from the right (PE3)
87
88     STR R6, [R2]      ;stores to Data register
89
90
91     B     loop
92
93     ALIGN          ; make sure the end of this section is aligned
94     END            ; end of file
95
```

Off

Registers

Register	Value
R0	0x0000000F
R1	0x4002451C
R2	0x400243FC
R3	0x0000000A
R4	0x00000000
R5	0x00000000
R6	0x00000001
R7	0x00000002
R8	0x00000004
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000400
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x000002C6
xPSR	0x01000000

Disassembly

```

0x000002C2 F1080804 ADD     r8,r8,#0x04
75:      AND R6, R3, R6 ;has PE0 val
0x000002C6 EA030606 AND     R6, R3, R6
76:      AND R7, R3, R7
0x000002CA EA030707 AND     R7, R3, R7
77:      AND R8, R3, R8
78:
79:      ;need to shift the
80:      LSR R7, R7, #1
81:      LSR R8, R8, #2

```

Port E Hardware

TM4C123 16 MHz

SW1 PE0
SW2 PE1
SW3 PE2

LED

Port E Registers

DATA: 0x06 PUR: 0x00 LOCK: 0x01
DIR: 0x08 PDR: 0x00 CR: 0xFF
DEN: 0x0F RCGCGPIO: 0x00000010 Clock enabled

Grading Controls

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

On

Registers

Register	Value
R0	0x0000000F
R1	0x4002451C
R2	0x400243FC
R3	0x00000006
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000002
R8	0x00000004
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000400
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x000002D2
xPSR	0x01000000

Disassembly

```

0x000002CE EA030808 AND     r8,r3,r8
80:      LSR R7, R7, #1
0x000002D2 EA4F0757 LSR     R7, R7, #1
81:      LSR R8, R8, #2
0x000002D6 EA4F0898 LSR     R8, R8, #2
82:
83:      ;need to shift the
84:      LSR R7, R7, #1
85:      LSR R8, R8, #2
86:      LSL R6, R6, #3

```

Port E Hardware

TM4C123 16 MHz

SW1 PE0
SW2 PE1
SW3 PE2

LED

Port E Registers

DATA: 0x0A PUR: 0x00 LOCK: 0x01
DIR: 0x08 PDR: 0x00 CR: 0xFF
DEN: 0x0F RCGCGPIO: 0x00000010 Clock enabled

Grading Controls

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