

Screenshots

Registers

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (GP)	0x20000400
R14 (LR)	0xFFFFFFFF

Disassembly

```
Reset_Handler:
0x0000026C F00B80A B.W      Start (0x00000284)
288:
289:      ENDP
290:
291: ;*****
292: ;
293: ; This is the code that gets called when the processor re
294: ; interrupt. This simply enters an infinite loop, prese
295: ; for examination by a debugger.
296: ;
297: ;*****
298: HardFault_Handler\
299:      PROC
300:      EXPORT HardFault_Handler [WEAK]
```

Command

```
Running with Code Size Limit: 32K
Load "C:\\Keil_v5\\EE319KwareFall2018\\Lab1_EE319K\\Lab1.axf"
LA ((PORTE & 0x00000008) >> 3 & 0x8) >> 3
LA ((PORTE & 0x00000004) >> 2 & 0x4) >> 2
LA ((PORTE & 0x00000002) >> 1 & 0x2) >> 1
LA ((PORTE & 0x00000001) & 0x1) >> 0
```

TExaS Lab 1

Port E Hardware

TM4C123 16 MHz

SW1 SW2 SW3

PE0 PE1 PE2

LED

Port E Registers

DATA: 0x09 PUR: 0x00 LOCK: 0x01

DIR: 0x08 PDR: 0x00 CR: 0xFF

DEN: 0x0F RCGCGPIO: 0x00000010 Clock enabled

Grading Controls

Number from Ed<: Grade Score: 0

Copy this to Ed<:

Registers

Register	Value
R0	0x400243FC
R1	0x0000000A
R2	0x00000017
R3	0x0000001D
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000400
R14 (LR)	0x0000028D

Disassembly

```
64:      EOR R2,#0x03
0x000002BA F0820203 EOR      r2,r2,#0x03 ;Compares bit 1
65:      EOR R3,R2,R1
0x000002BE EA820301 EOR      r3,r2,r1 ;Compares R2 and
66:      LSL R3,#1
0x000002C2 EA4F0343 LSL      r3,r3,#1
67:      EOR R1,R3
0x000002C6 EA810103 EOR      r1,r1,r3
68:      LSL R1,#1
0x000002CA EA4F0141 LSL      r1,r1,#1
69:      EOR R1,#0xF
0x000002CE F081010F EOR      r1,r1,#0x0F ;Compare
70:      BX LR
71:
72: PortE_Output
```

Command

```
Running with Code Size Limit: 32K
Load "C:\\Keil_v5\\EE319KwareFall2018\\Lab1_EE319K\\Lab1.axf"
LA ((PORTE & 0x00000008) >> 3 & 0x8) >> 3
LA ((PORTE & 0x00000004) >> 2 & 0x4) >> 2
LA ((PORTE & 0x00000002) >> 1 & 0x2) >> 1
LA ((PORTE & 0x00000001) & 0x1) >> 0
```

TExaS Lab 1

Port E Hardware

TM4C123 16 MHz

SW1 SW2 SW3

PE0 PE1 PE2

LED

Port E Registers

DATA: 0x00 PUR: 0x00 LOCK: 0x01

DIR: 0x08 PDR: 0x00 CR: 0xFF

DEN: 0x0F RCGCGPIO: 0x00000010 Clock enabled

Grading Controls

Number from Ed<: Grade Score: 0

Copy this to Ed<:

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

```

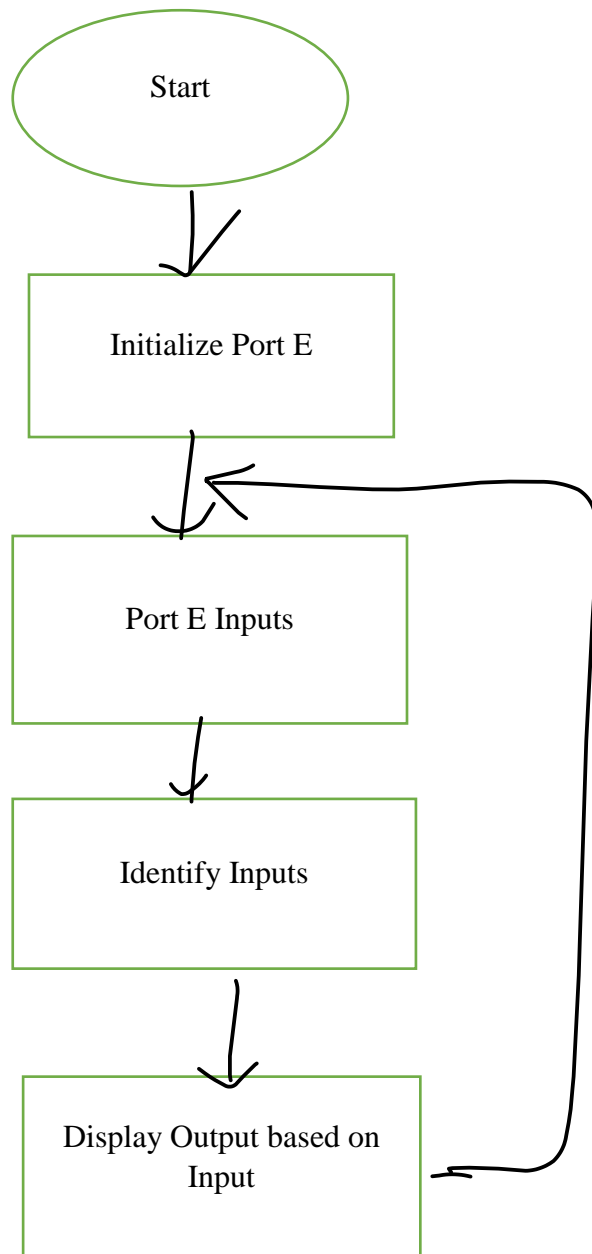
STR R1,[R0]                      ;write Port E, sets PE3, enables LED on or
off depending on the truth table
BX   LR                          ;Return

```

ALIGN ; make sure the end of this section is aligned

END ; end of file

Flowchart



Pseudocode

