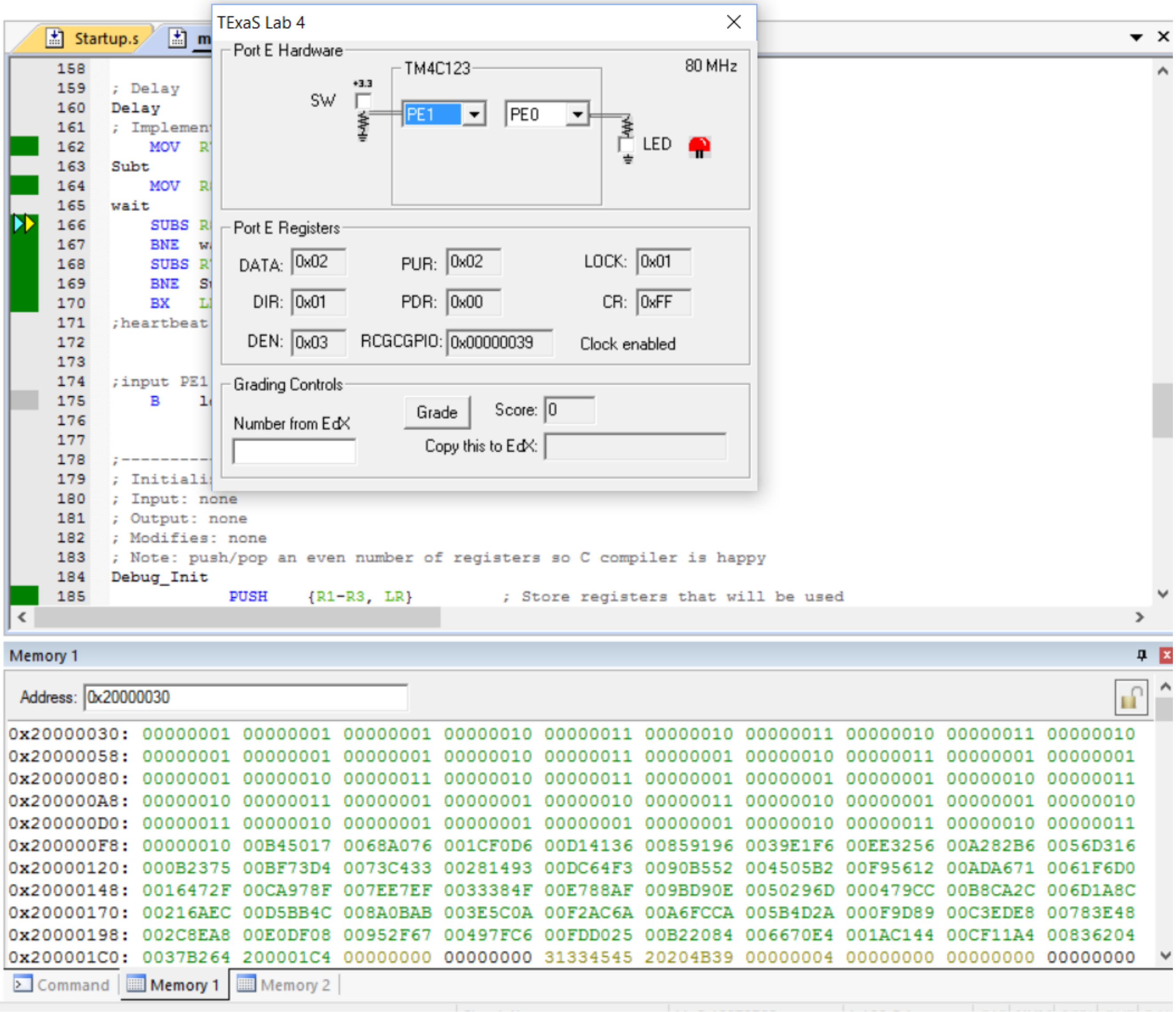
```
;****************** main.s ***************
; Program written by: Megan Cooper and Kaela Todd
; Date Created: 1/22/2016
; Last Modified: 1/22/2016
; Section Wednesday 3-4
; Instructor: Ramesh Yerraballi
; Lab number: 4
; Brief description of the program
;   If the switch is presses, the LED toggles at 8 Hz
; Hardware connections
;  PE1 is switch input  (1 means pressed, 0 means not pressed)
;  PE0 is LED output  (1 activates external LED on protoboard)
;Overall functionality of this system is the similar to Lab 3, with three changes:
;1-  initialize SysTick with RELOAD 0x00FFFFFF
;2-  add a heartbeat to PF2 that toggles every time through loop
;3-  add debugging dump of input, output, and time
; Operation
;   1) Make PE0 an output and make PE1 an input.
;   2) The system starts with the LED on (make PE0 =1).
;   3) Wait about 62 ms
;   4) If the switch is pressed (PE1 is 1), then toggle the LED once, else turn the LED on.
;   5) Steps 3 and 4 are repeated over and over


LED                     EQU 0x40024004   ;PE0
SWITCH                  EQU 0x40024008   ;PE1
SYSCTL_RCGCGPIO_R       EQU 0x400FE608
SYSCTL_RCGC2_GPIOE      EQU 0x00000010   ; port E Clock Gating Control
SYSCTL_RCGC2_GPIOF      EQU 0x00000020   ; port F Clock Gating Control
GPIO_PORTE_DATA_R       EQU 0x400243FC
GPIO_PORTE_DIR_R        EQU 0x40024400
GPIO_PORTE_AFSEL_R      EQU 0x40024420
GPIO_PORTE_PUR_R        EQU 0x40024510
GPIO_PORTE_DEN_R        EQU 0x4002451C
GPIO_PORTF_DATA_R       EQU 0x400253FC
GPIO_PORTF_DIR_R        EQU 0x40025400
GPIO_PORTF_AFSEL_R      EQU 0x40025420
GPIO_PORTF_DEN_R        EQU 0x4002551C
NVIC_ST_CTRL_R          EQU 0xE000E010
NVIC_ST_RELOAD_R        EQU 0xE000E014
NVIC_ST_CURRENT_R       EQU 0xE000E018
            THUMB
            AREA    DATA, ALIGN=4
SIZE        EQU    50
;You MUST use these two buffers and two variables
;You MUST not change their names
;These names MUST be exported
            EXPORT DataBuffer
            EXPORT TimeBuffer
            EXPORT DataPt [DATA,SIZE=4]
            EXPORT TimePt [DATA,SIZE=4]
DataBuffer  SPACE  SIZE*4
TimeBuffer  SPACE  SIZE*4
DataPt      SPACE  4
TimePt      SPACE  4


            ALIGN
            AREA    |.text|, CODE, READONLY, ALIGN=2
            THUMB
            EXPORT  Start
            IMPORT  TExaS_Init
            IMPORT  SysTick_Init


Start
; running at 80 MHz, scope voltmeter on PD3
            BL      TExaS_Init
; turn on clock for Ports E and F
            LDR     R0,= SYSCTL_RCGCGPIO_R
            LDR     R1,[R0]
            ORR     R1,#0x30
            STR     R1,[R0]

            NOP
            NOP
; initialize Port E
            LDR     R0,= GPIO_PORTE_DIR_R
            LDR     R1,[R0]
            BIC     R1,#0x02
            ORR     R1,#0x01
            STR     R1,[R0]

            LDR     R0,= GPIO_PORTE_AFSEL_R
            LDR     R1,[R0]
            BIC     R1,#0x03
            STR     R1,[R0]

            LDR     R0,= GPIO_PORTE_DEN_R
            LDR     R1,[R0]
            ORR     R1,#0x03
            STR     R1,[R0]

;           LDR     R0, =GPIO_PORTE_PUR_R
;           LDR     R1, [R0]
;           ORR     R1, #0x02
;           STR     R1,[R0]

; initialize Port F
            LDR     R0,= GPIO_PORTF_DIR_R
            LDR     R1,[R0]
            ORR     R1,#0x04
            STR     R1,[R0]

            LDR     R0,= GPIO_PORTF_AFSEL_R
            LDR     R1,[R0]
            BIC     R1,#0x04
            STR     R1,[R0]

            LDR     R0,= GPIO_PORTF_DEN_R
            LDR     R1,[R0]
            ORR     R1,#0x04
            STR     R1,[R0]

            BL      Debug_Init  ; initialize debugging dump, including SysTick



            CPSIE   I           ; TExaS voltmeter, scope runs on interrupts

            LDR     R0,= LED
            LDR     R1,[R0]
            ORR     R1,#0xFF    ; turns the LED on
            STR     R1,[R0]

            MOV     R5,#0xC8    ; Counter for Debug_Capture
loop

DC_loop     CMP     R5, #0x0
            BEQ     full
            BL      Debug_Capture
            SUB     R5, #0x04

full        BL      Delay       ; 2480062 instructions
; Heartbeat
            LDR     R3,= GPIO_PORTF_DATA_R
            LDR     R4, [R3]
            EOR     R4, #0xFF
            STR     R4, [R3]    ; 4 instructions

            LDR     R0,= SWITCH
            LDR     R2,= LED    ; 2 instructions

            LDR     R1,[R0]
            CMP     R1,#0       ; 2 instructions
            BNE     Toggle      ; Goes to Toggle if PE1 = 1
            BEQ     StayOn      ; Goes to StayOn if PE1 = 0
       ; 2480070 instructions
       ; 2480070*2*12.5ns= 62001750ns
       ; 725ns/62001750ns * 100% = 0.00117%

Toggle
; Flips PE0 if the switch is pressed
    LDR  R1,[R2]
    EOR  R1, R1, #0xFF
    STR  R1,[R2]
    B    loop; 4 instructions

StayOn
; Clears PE1 and returns to loop
    LDR  R1,[R2]
    ORR  R1,#0xFF
    STR  R1,[R2]
    B    loop; 4 instructions

; Delay
Delay
; Implements a 62ms long delay
    MOV  R7, #20
Subt
    MOV  R8, #62000
wait
    SUBS R8, #1
    BNE  wait        ; 2*62000=124000 instructions
    SUBS R7, #1
    BNE  Subt        ; (124000+3)*20 = 2480060 instructions
    BX   LR          ; 2480060 +2=2480062 instructions

    B    loop


;------------Debug_Init------------
; Initializes the debugging instrument
; Input: none
; Output: none
; Modifies: none
; Note: push/pop an even number of registers so C compiler is happy
Debug_Init
            PUSH    {R1-R3, LR}     ; Store registers that will be used
            LDR     R1, =DataBuffer
            MOV     R3, R1
            ADD     R3, #0xC8
            MOV     R2, #0xFFFFFFFF
notDone1    STR     R2, [R1]        ; Store 0xFFFFFFFF as the first element of DataBuffer
            ADD     R1, #0x04
            CMP     R1, R3
            BNE     notDone1

            LDR     R1, =DataBuffer
            LDR     R2, =DataPt
            STR     R1, [R2]        ; Make DataPt point to the start of DataBuffer

            LDR     R1, =TimeBuffer
            MOV     R3, R1
            ADD     R3, #0xC8
            MOV     R2, #0xFFFFFFFF
notDone2    STR     R2, [R1]        ; Store 0xFFFFFFFF as the first element of TimeBuffer
            ADD     R1, #0x04
            CMP     R1, R3
            BNE     notDone2

            LDR     R1, =TimeBuffer
            LDR     R2, =TimePt
            STR     R1, [R2]        ; Make TimePt point to the start of TimeBuffer

            BL      SysTick_Init    ; Init SysTick
            POP     {R1-R3, LR}     ; Pop stored values back into registers
            BX      LR

;------------Debug_Capture------------
; Dump Port E and time into buffers
; Input: none
; Output: none
; Modifies: none
; Note: push/pop an even number of registers so C compiler is happy
Debug_Capture
            PUSH    {R0-R8,LR}   ; Save used registers to the Stack

            LDR     R0, =DataPt
            LDR     R2, [R0]     ; R2 = pointer to DataBuffer


            LDR     R5, =TimePt
            LDR     R7, [R5]     ; R7 = pointer to TimeBuffer


            LDR     R3, =SWITCH
            LDR     R3, [R3]     ; R3 = SWITCH value
            LSL     R3, #3
            LDR     R4, =LED
            LDR     R4, [R4]     ; R4 = LED value
            ADD     R4, R3, R4   ; combine LED and SWITCH into one word
            STR     R4, [R2]     ; Store in DataBuffer

            ADD     R2, #0x04
            STR     R2, [R0]     ; Increment DataPt
            LDR     R8, =NVIC_ST_CURRENT_R
            LDR     R8, [R8]
            STR     R8, [R7]     ; Store time in TimeBuffer
            ADD     R7, #0x04
            STR     R7, [R5]     ; Increment TimePt

done        POP     {R0-R8, LR}
            BX      LR           ; 29 cycles




                                  ;29*2*12.5ns = 725ns

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

```

```
; SysTick.s
; Runs on LM4F120/TM4C123
; Provide functions that initialize the SysTick module, wait at least a
; designated number of clock cycles, and wait approximately a multiple
; of 10 milliseconds using busy wait.  After a power-on-reset, the
; LM4F120 gets its clock from the 16 MHz precision internal oscillator,
; which can vary by +/- 1% at room temperature and +/- 3% across all
; temperature ranges.  If you are using this module, you may need more
; precise timing, so it is assumed that you are using the PLL to set
; the system clock to 50 MHz.  This matters for the function
; SysTick_Wait10ms(), which will wait longer than 10 ms if the clock is
; slower.
; Daniel Valvano
; September 12, 2013

;  This example accompanies the book
;  "Embedded Systems: Introduction to ARM Cortex M Microcontrollers",
;  ISBN: 978-1469998749, Jonathan Valvano, copyright (c) 2014
;  Program 2.11, Section 2.6
;
;Copyright 2014 by Jonathan W. Valvano, valvano@mail.utexas.edu
;   You may use, edit, run or distribute this file
;   as long as the above copyright notice remains
;THIS SOFTWARE IS PROVIDED "AS IS".  NO WARRANTIES, WHETHER EXPRESS, IMPLIED
;OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF
;MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE.
;VALVANO SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL,
;OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.
;For more information about my classes, my research, and my books, see
;http://users.ece.utexas.edu/~valvano/

NVIC_ST_CTRL_R        EQU 0xE000E010
NVIC_ST_RELOAD_R      EQU 0xE000E014
NVIC_ST_CURRENT_R     EQU 0xE000E018
NVIC_ST_CTRL_COUNT    EQU 0x00010000  ; Count flag
NVIC_ST_CTRL_CLK_SRC  EQU 0x00000004  ; Clock Source
NVIC_ST_CTRL_INTEN    EQU 0x00000002  ; Interrupt enable
NVIC_ST_CTRL_ENABLE   EQU 0x00000001  ; Counter mode
NVIC_ST_RELOAD_M      EQU 0x00FFFFFF  ; Counter load value

        AREA    |.text|, CODE, READONLY, ALIGN=2
        THUMB
        EXPORT   SysTick_Init
        EXPORT   SysTick_Wait
        EXPORT   SysTick_Wait10ms

;------------SysTick_Init------------
; Initialize SysTick with busy wait running at bus clock.
; Input: none
; Output: none
; Modifies: R0, R1
SysTick_Init
    ; disable SysTick during setup
    LDR R1, =NVIC_ST_CTRL_R         ; R1 = &NVIC_ST_CTRL_R
    MOV R0, #0                      ; R0 = 0
    STR R0, [R1]                    ; [R1] = R0 = 0
    ; maximum reload value
    LDR R1, =NVIC_ST_RELOAD_R       ; R1 = &NVIC_ST_RELOAD_R
    LDR R0, =NVIC_ST_RELOAD_M;      ; R0 = NVIC_ST_RELOAD_M
    STR R0, [R1]                    ; [R1] = R0 = NVIC_ST_RELOAD_M
    ; any write to current clears it
    LDR R1, =NVIC_ST_CURRENT_R      ; R1 = &NVIC_ST_CURRENT_R
    MOV R0, #0                      ; R0 = 0
    STR R0, [R1]                    ; [R1] = R0 = 0
    ; enable SysTick with core clock
    LDR R1, =NVIC_ST_CTRL_R         ; R1 = &NVIC_ST_CTRL_R
                                    ; R0 = ENABLE and CLK_SRC bits set
    MOV R0, #(NVIC_ST_CTRL_ENABLE+NVIC_ST_CTRL_CLK_SRC)
    STR R0, [R1]                    ; [R1] = R0 = (NVIC_ST_CTRL_ENABLE|NVIC_ST_CTRL_CLK_SRC)
    BX  LR                          ; return

;------------SysTick_Wait------------
```

```
; Time delay using busy wait.
; Input: R0  delay parameter in units of the core clock (units of 12.5 nsec for 80 MHz clock)
; Output: none
; Modifies: R0, R1, R3
SysTick_Wait
    LDR  R1, =NVIC_ST_RELOAD_R      ; R1 = &NVIC_ST_RELOAD_R
    SUB  R0, #1
    STR  R0, [R1]                   ;delay-1;  // number of counts to wait
    LDR  R1, =NVIC_ST_CTRL_R        ; R1 = &NVIC_ST_CTRL_R
SysTick_Wait_loop
    LDR  R3, [R1]                   ; R3 = NVIC_ST_CTRL_R
    ANDS R3, R3, #0x00010000       ; Count set?
    BEQ  SysTick_Wait_loop
    BX   LR                         ; return

;------------SysTick_Wait10ms------------
; Time delay using busy wait.  This assumes 50 MHz clock
; Input: R0  number of times to wait 10 ms before returning
; Output: none
; Modifies: R0
DELAY10MS             EQU 800000    ; clock cycles in 10 ms (assumes 80 MHz clock)
SysTick_Wait10ms
    PUSH {R4, LR}                   ; save current value of R4 and LR
    MOVS R4, R0                     ; R4 = R0 = remainingWaits
    BEQ SysTick_Wait10ms_done       ; R4 == 0, done
SysTick_Wait10ms_loop
    LDR R0, =DELAY10MS              ; R0 = DELAY10MS
    BL  SysTick_Wait                ; wait 10 ms
    SUBS R4, R4, #1                 ; R4 = R4 - 1; remainingWaits--
    BHI SysTick_Wait10ms_loop       ; if(R4 > 0), wait another 10 ms
SysTick_Wait10ms_done
    POP {R4, LR}                    ; restore previous value of R4 and LR
    BX  LR                          ; return

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



```
; Delay
Delay
; Implemen
    MOV  R
Subt
    MOV  R
wait
    SUBS R
    BNE  w
    SUBS R
    BNE  S
    BX   L
;heartbeat


;input PE1
    B    l


;---------
; Initiali
; Input: none
; Output: none
; Modifies: none
; Note: push/pop an even number of registers so C compiler is happy
Debug_Init
            PUSH    {R1-R3, LR}          ; Store registers that will be used
```

TExaS Lab 4

Port E Hardware

Port E Registers

DATA: 0x02 PUR: 0x02 LOCK: 0x01

DIR: 0x01 PDR: 0x00 CR: 0xFF

DEN: 0x03 RCGCGPIO: 0x00000039 Clock enabled

Grading Controls

Number from EdX

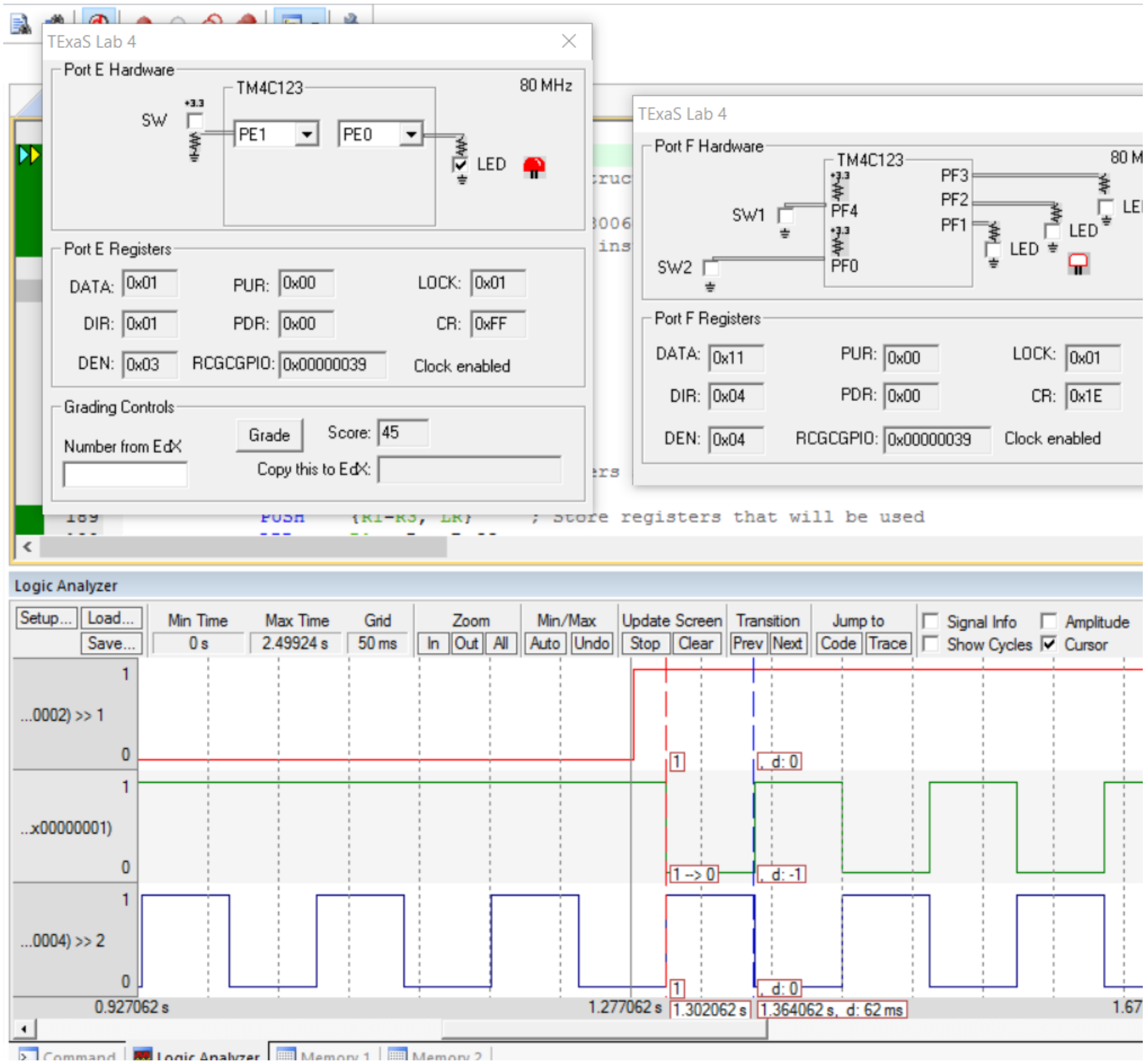
Score: 0

Copy this to EdX:

Memory 1

Address: 0x20000030

```
0x20000030: 00000001 00000001 00000001 00000010 00000011 00000010 00000011 00000010 00000011 00000010
0x20000058: 00000001 00000001 00000001 00000010 00000011 00000001 00000010 00000011 00000001 00000001
0x20000080: 00000001 00000010 00000011 00000010 00000011 00000001 00000001 00000001 00000010 00000011
0x200000A8: 00000010 00000011 00000001 00000001 00000010 00000011 00000010 00000001 00000001 00000010
0x200000D0: 00000011 00000010 00000001 00000001 00000001 00000001 00000010 00000011 00000010 00000011
0x200000F8: 00000010 00B45017 0068A076 001CF0D6 00D14136 00859196 0039E1F6 00EE3256 00A282B6 0056D316
0x20000120: 000B2375 00BF73D4 0073C433 00281493 00DC64F3 0090B552 004505B2 00F95612 00ADA671 0061F6D0
0x20000148: 0016472F 00CA978F 007EE7EF 0033384F 00E788AF 009BD90E 0050296D 000479CC 00B8CA2C 006D1A8C
0x20000170: 00216AEC 00D5BB4C 008A0BAB 003E5C0A 00F2AC6A 00A6FCCA 005B4D2A 000F9D89 00C3EDE8 00783E48
0x20000198: 002C8EA8 00E0DF08 00952F67 00497FC6 00FDD025 00B22084 006670E4 001AC144 00CF11A4 00836204
0x200001C0: 0037B264 200001C4 00000000 00000000 31334545 20204B39 00000004 00000000 00000000 00000000
```



TExaS Lab 4

Port E Hardware

TM4C123 80 MHz

SW □ PE1 PE0 ☑ LED

Port E Registers

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

DIR: 0x01 PDR: 0x00 CR: 0xFF

DEN: 0x03 RCGCGPIO: 0x00000039 Clock enabled

Grading Controls

Number from EdX Grade Score: 45

Copy this to EdX:

TExaS Lab 4

Port F Hardware

TM4C123 80 M

SW1 PF4 SW2 PF0 PF3 PF2 PF1 LED LED LE

Port F Registers

DATA: 0x11 PUR: 0x00 LOCK: 0x01

DIR: 0x04 PDR: 0x00 CR: 0x1E

DEN: 0x04 RCGCGPIO: 0x00000039 Clock enabled

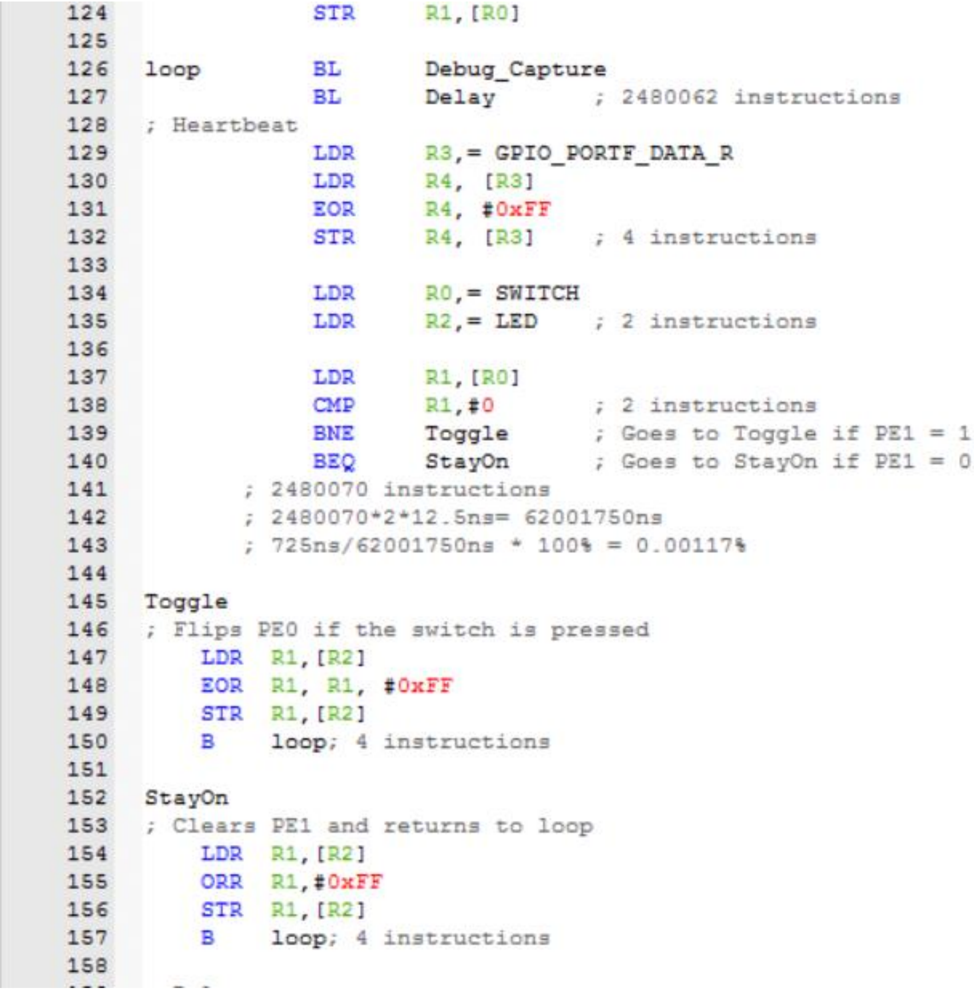
```
189            PUSH     {R1-R3, LR}       ; Store registers that will be used
```

Logic Analyzer

Min Time 0 s Max Time 2.49924 s Grid 50 ms

□ Signal Info □ Amplitude □ Show Cycles ☑ Cursor

Estimation of Intrusiveness:



```
            STR     R1,[R0]

loop        BL      Debug_Capture
            BL      Delay       ; 2480062 instructions
; Heartbeat
            LDR     R3,= GPIO_PORTF_DATA_R
            LDR     R4, [R3]
            EOR     R4, #0xFF
            STR     R4, [R3]    ; 4 instructions

            LDR     R0,= SWITCH
            LDR     R2,= LED    ; 2 instructions

            LDR     R1,[R0]
            CMP     R1,#0       ; 2 instructions
            BNE     Toggle      ; Goes to Toggle if PE1 = 1
            BEQ     StayOn      ; Goes to StayOn if PE1 = 0
       ; 2480070 instructions
       ; 2480070*2*12.5ns= 62001750ns
       ; 725ns/62001750ns * 100% = 0.00117%

Toggle
; Flips PE0 if the switch is pressed
    LDR  R1,[R2]
    EOR  R1, R1, #0xFF
    STR  R1,[R2]
    B    loop; 4 instructions

StayOn
; Clears PE1 and returns to loop
    LDR  R1,[R2]
    ORR  R1,#0xFF
    STR  R1,[R2]
    B    loop; 4 instructions
```

Calculations show about 0.00117 %

Results of Debugging Instrument:

```
:1000000010000000100000010000000
:1000000100000001100000010000000
:1000000100000000100000001000000
:1000000100000001100000001000000
:0000000110000000100000001000000
:1000000100000001100000010000000
:1000000010000000100000001000000
:0000000110000001000000011000000
:1000000010000001000000011000000
:0000000010000000100000010000000
:1000000100000000100000001000000
:1000000010000001000000011000000
:000000011000000100000001750B400
:6A06800D6F01C003641D10096918500
:6E139005632EE00B682A20016D35600
:5230B00D473BF0033C4730093142800
:364DC0052B59000B20545001256F900
:1A6AD00D0F661002F4716008F97CA00
:FE77E004F383300AF88E7000ED99B00
:D295000CC7904002CCAB8008C1A6D00
:C6A21004CBBD500AB0B8A000A5C3E00
:AACF200CAFCA6002A4D5B00899D0F00
:EDC300483E7800A8
```

Calculation of LED Period in msec =

First Calculation

@ 0000 0010 -> 0x0068A077

@ 0000 0011 -> 0x001CF0D7

Time = 0x4BAFA0 (difference between two pts) = 4960160 (in decimal)

Period = 4960160 * 12.5 ns = 62,002,000 ns = 62.002 ms

Second Calculation

@ 0000 0010 -> 0x00EE3255

@ 0000 0011 -> 0x00A282B5

Time = 0x4BAFA0 (diff. btw two pts) = 4960160 (in decimal)

Period = 4960160 * 12.5ns = 62,002,000 ns = 62.002ms

Third Calculation

@ 0000 0010 -> 0x00D14137

@ 0000 0001 -> 0x00859196

Time = 0x4BAFA1 (diff. btw two pts) = 4960161 (in decimal)

Period = 4960161 * 12.5ns = 62,002,012.5 ns = 62.0020125 ms

Average Calculation

$$\frac{(62002000) + (62002000) + (62002012.5\ )}{3} = 62002004.17\,ns = 62.002\,ms$$