```
;****************** main.s ***************
; Program written by: HYDER SHAD, MIRA SEHGAL
; Date Created: 1/22/2016
; Last Modified: 3/2/2016
; Section: THURSDAY 2-3 PM
; Instructor: V. JANAPA
; 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


SWITCH                  EQU 0x40024004  ;PE0
LED                     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
heartcount EQU 992000   ;calculated for lab 4 heart beat (62ms delay
running on 80MHz clock frequency)
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_PCTL_R       EQU   0x4002452C
GPIO_PORTF_DATA_R       EQU 0x400253FC
GPIO_PORTF_DIR_R        EQU 0x40025400
GPIO_PORTF_AFSEL_R      EQU 0x40025420
GPIO_PORTF_DEN_R        EQU 0x4002551C
GPIO_PORTF_PCTL_R               EQU 0x4002552C
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

INITPORTS

;1) activate clock for Port E
    LDR R0, =SYSCTL_RCGCGPIO_R
    LDR R1, [R0]
    ORR R1, #0X10                                           ;SET
BIT 4 HIGH TO ENABLE PORT E CLOCK
    STR R1, [R0]
    NOP
    NOP                                  ;ALLOW TIME TO FINISH
ACTIVATING (2+ CYCLES)
    NOP

; 2) set direction register

        LDR R0, =GPIO_PORTE_DIR_R               ;SET BIT 3 HIGH FOR
TO BE OUTPUT
        LDR R1, [R0]
        ORR R1, #0X01                                       ;using
pin PE0 as output, PE1 as input
        STR R1, [R0]

; 4) configure as GPIO

        LDR R0, =GPIO_PORTE_PCTL_R
    MOV R1, #0X0                                ;CLEAR PORT CONTROL
FIELD TO SET UP PINS FOR GPIO
    STR R1, [R0]

; 5) regular port function

        LDR R0, =GPIO_PORTE_AFSEL_R             ;DISABLE ALT FUNCTIONS
FOR BIN BY SETTING BITS TO ZERO
```

```
    MOV R1, #0
    STR R1, [R0]

; 6) enable digital port

        LDR R0, =GPIO_PORTE_DEN_R               ;R1 =
&GPIO_PORTD_DEN_R
    ORR R1, #0X03                               ;ENABLE DIGITAL I/O ON
PINS 0 and 1
    STR R1, [R0]
;-------------------------------------------------------------------
------------------
; 1) activate clock for Port F
    LDR R0, =SYSCTL_RCGCGPIO_R
    LDR R1, [R0]
    ORR R1, #0X20                                                   ;SET
BIT 5 HIGH TO ENABLE PORT F CLOCK
    STR R1, [R0]
    NOP
    NOP                                 ;ALLOW TIME TO FINISH
ACTIVATING (2+ CYCLES)
    NOP

; 3) set direction register

        LDR R0, =GPIO_PORTF_DIR_R               ;SET BIT 2 HIGH FOR IT
TO BE OUTPUT
        LDR R1, [R0]
        ORR R1, #0X04
        STR R1, [R0]

; 5) configure as GPIO

        LDR R0, =GPIO_PORTF_PCTL_R
    MOV R1, #0X0                                ;CLEAR PORT CONTROL
FIELD TO SET UP PINS FOR GPIO
    STR R1, [R0]

; 6) regular port function

        LDR R0, =GPIO_PORTF_AFSEL_R             ;DISABLE ALT FUNCTIONS
FOR PIN BY SETTING BITS TO ZERO
    MOV R1, #0
    STR R1, [R0]

; 7) enable digital port

        LDR R0, =GPIO_PORTF_DEN_R               ;R1 =
&GPIO_PORTD_DEN_R
    ORR R1, #0X04                               ;ENABLE DIGITAL I/O ON
```

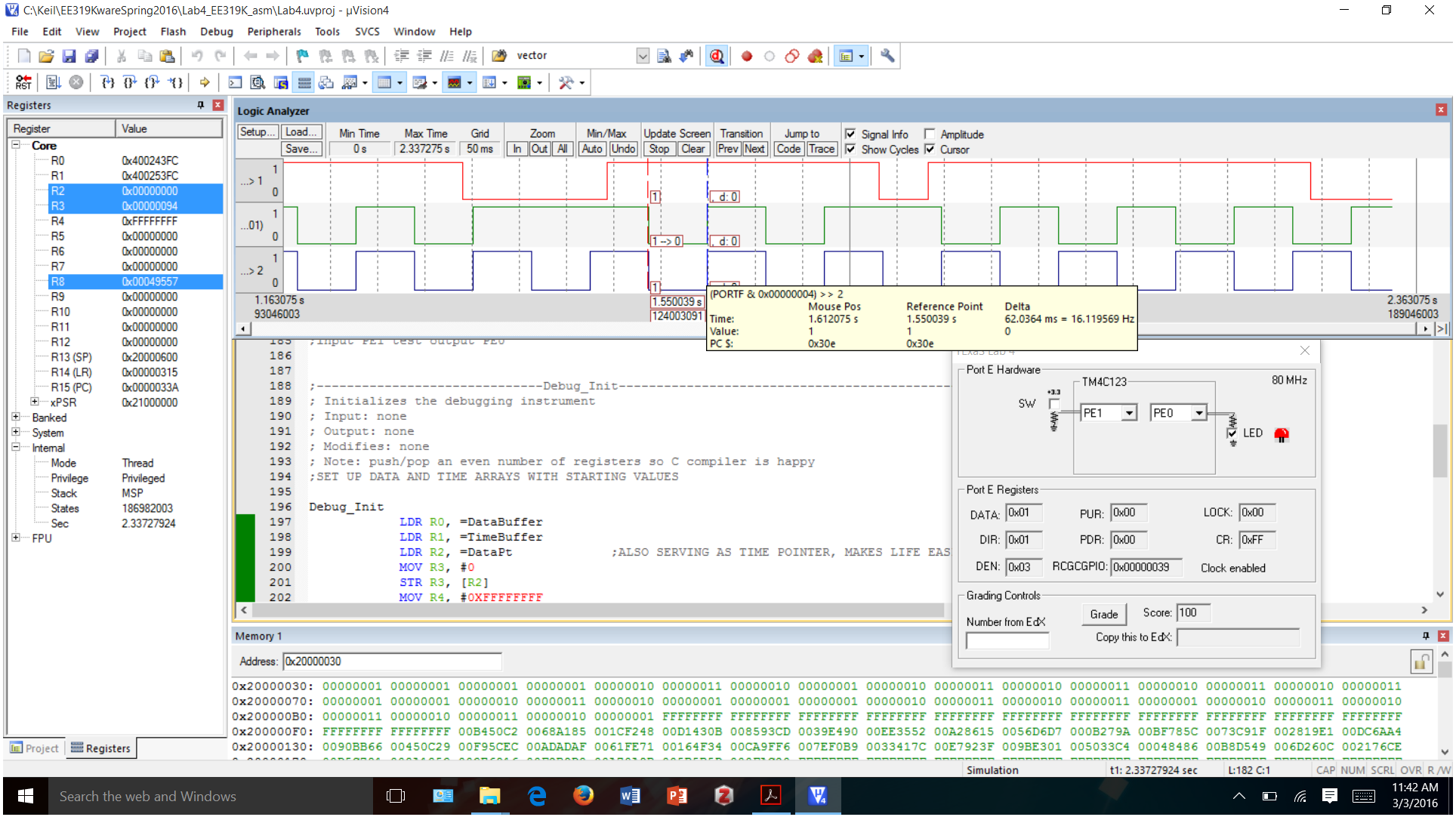
```
PIN 2
    STR R1, [R0]

; exit initialization process
        BX LR

Start BL   TExaS_Init  ; running at 80 MHz, scope voltmeter on PD3
; initialize Port E and F
        BL INITPORTS
; initialize debugging dump, including SysTick
        BL Debug_Init

    CPSIE  I    ; TExaS voltmeter, scope runs on interrupts
;------------------------END
INITIALIZATION-------------------------------------------------------
-------
;LAB 3 CODE WITH CAPTURE

                MOV R3, #0
        ;RESET ARRAY POINTERS
                LDR R0, =GPIO_PORTE_DATA_R              ;load R0 with
address of Port E data (memory mapped I/O)
                LDR R1, [R0]                                        ;load
Port E data into R1
                ORR R1, #0X01                                       ;set
bit 0 (PE0) HIGH to turn on LED (positive logic)
                STR R1, [R0]                                        ;write
modified data back to memory mapped I/O

                ;HEARTBEAT
loop    LDR R1, =GPIO_PORTF_DATA_R      ;HEARTBEAT TOGGLE CODE
                LDR R2, [R1]
                EOR R2, #0X04
                STR R2, [R1]
                BL heartdelay

                LDR R1, [R0]                                        ;load
data into R0
                CMP R1, #0X03                                       ;check
to see if the switch has been pressed, bit 1 HIGH means pressed,
toggle led
                BNE noswitch
        ;if switch pressed
                EOR R1, #0X01                                       ;EOR
to turn TOGGLE LED
                STR R1, [R0]                                        ;write
modified data back to memory mapped I/O to turn LED off
                B done
noswitch ORR R1, #0X01                                  ;SWITCH NOT
PRESSED LED ON
```

```
                STR R1, [R0]

                ;CAPTURE SECTION
done    CMP R3, #200
                BEQ FULL
        ;IF ARRAY FULL, SKIP DATA CAPTURE
                BL
Debug_Capture                           ;RECORD DATA
FULL    B loop
                                                        ;loop to check
for switch press

;----------------------------Delay
Functions-------------------------------------------------------------

                                                ;exit delay
subroutine, use link register to return to main program code

heartdelay LDR R8, =heartcount                  ;time delay, count
down from calculated value for 80 MHz Clock, with delay taking 5
cycles each time
count2  ADD R8,
#-1                                             ;subtract 1 from
countdown
                CMP R8, #0
        ;compare to see if countdown has reached 0
                BNE count2
        ;countdown = 0, then exit delay subroutine, otherwise loop to
count
                BX LR
;input PE1 test output PE0


;------------------------------
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

;SET UP DATA AND TIME ARRAYS WITH STARTING VALUES

Debug_Init
                        LDR R0, =DataBuffer
                        LDR R1, =TimeBuffer
                        LDR R2, =DataPt                         ;ALSO
SERVING AS TIME POINTER, MAKES LIFE EASIER
                        MOV R3, #0
```

```
                        STR R3, [R2]
                        MOV R4, #0XFFFFFFFF
ARRAYINIT       STR R4, [R0, R3]
                        STR R4, [R1, R3]
                        ADD R3, #4
                        STR R3, [R2]
                        CMP R3, #200
                        BNE ARRAYINIT

                        ; init SysTick
                        LDR R0, =NVIC_ST_CTRL_R
                        LDR R1, [R0]
                        AND R1, #~0X01
                        STR R1, [R0]

                        LDR R0, =NVIC_ST_RELOAD_R
                        LDR R1, [R0]
                        ORR R1, #0X00FFFFFF
                        STR R1, [R0]

                        LDR R0, =NVIC_ST_CURRENT_R
                        MOV R1, #0
                        STR R1, [R0]

                        LDR R0, =NVIC_ST_CTRL_R
                        LDR R1, [R0]
                        ORR R1, #0X05 ; ENABLE AND SET CLOCK SOURCE
BIT HIGH
                        STR R1, [R0]

                        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 INTRUSIVENESS CALCULATIONS:
;debug capture is 13 instructions, which using approximation of 2
cycles per instruction means subroutine is 26 cycles or .325us long
;there are 5000061 cycles executed between calls, which at 12.5ns/
cycle is 0.0625007625s in between debug capture subroutine calls
;percentage overhead is 100x(0.325us/0.0625007625s) = 0.000519993656%,
which by our definitions is considered a minimally intrusive debugging
method
```

```
Debug_Capture
          ;DO NOT CHANGE R0, R3 IS A PASS PARAMETER THAAT HOLDS
POINTER VALUE
          ;NO LDR DATA PORT ADDRESS SINCE R0 FROM MAIN CODE ALREADY
HAS IT, DO NOT MODIFY R0
          PUSH {R4-R6, LR}                              ;SAVE
REGISTERS TO STACK
          LDR R4, =DataBuffer
          LDR R5, =TimeBuffer
          LDR R1, [R0]
          AND R2, R1, #0X02
          AND R1, #~0X02
          LSL R2, #3
          ADD R1, R2, R1
          STR R1, [R4, R3]                              ;STORE DATA IN
DATA ARRAY
          LDR R1, =NVIC_ST_CURRENT_R
          LDR R1, [R1]
          STR R1, [R5, R3]                              ;STORE TIME IN
TIME ARRAY
      ADD R3, #4                                  ;INCREMENT POINTER FOR
BOTH ARRAYS
          POP {R4-R6, PC}

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

00350C0094080000BC08000000000000
00000000000000000000000000000000
00000000000000001000000000000000
01000000100000001000000010000000
10000000110000001000000011000000
10000000110000001000000001000000
01000000100000001000000011000000
10000000110000001000000001000000
01000000100000001100000001000000
01000000100000001000000001000000
01000000100000001000000001000000
01000000100000001000000011000000
10000000110000001000000011000000
01000000100000001000000011000000
10000000110000001000000011000000
1000000011000000C250B40085A16800
48F21C000B43D100CD93850090E43900
5235EE001586A200D7D656009A270B00
5C78BF001FC97300E2192800A56ADC00
67BB90002A0C4500EC5CF900AFADAD00
71FE6100344F1600F79FCA00B9F07E00
7C4133003F92E70002E39B00C5335000
888404004BD5B8000E266D00D1762100
94C7D50057188A001A693E00DDB9F200
9F0AA700625B5B0024AC0F00E7FCC300


C:\Keil\EE319KwareSpring2016\Lab4_EE319K_asm\Lab4.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Register Value
Core
R0 0x400243FC
R1 0x400253FC
R2 0x00000000
R3 0x00000094
R4 0xFFFFFFFF
R5 0x00000000
R6 0x00000000
R7 0x00000000
R8 0x00049557
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x20000600
R14 (LR) 0x00000315
R15 (PC) 0x0000033A
xPSR 0x21000000
Banked
System
Internal
Mode Thread
Privilege Privileged
Stack MSP
States 186982003
Sec 2.33727924
FPU
Logic Analyzer
Setup... Load... Save...
Min Time 0 s
Max Time 2.337275 s
Grid 50 ms
Zoom In Out All
Min/Max Auto Undo
Update Screen Stop Clear
Transition Prev Next
Jump to Code Trace
Signal Info
Show Cycles
Amplitude
Cursor
1.163075 s
93046003
1.550039 s
124003091
2.363075 s
189046003
(PORTF & 0x00000004) >> 2
Mouse Pos Reference Point Delta
Time: 1.612075 s 1.550039 s 62.0364 ms = 16.119569 Hz
Value: 1 1 0
PC $: 0x30e 0x30e
;Input PE1 test Output PE0
;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
;SET UP DATA AND TIME ARRAYS WITH STARTING VALUES
Debug_Init
LDR R0, =DataBuffer
LDR R1, =TimeBuffer
LDR R2, =DataPt ;ALSO SERVING AS TIME POINTER, MAKES LIFE EAS
MOV R3, #0
STR R3, [R2]
MOV R4, #0XFFFFFFFF
Port E Hardware
TM4C123
80 MHz
SW
PE1
PE0
LED
Port E Registers
DATA: 0x01
PUR: 0x00
LOCK: 0x00
DIR: 0x01
PDR: 0x00
CR: 0xFF
DEN: 0x03
RCGCGPIO: 0x00000039
Clock enabled
Grading Controls
Number from EdX
Grade
Score: 100
Copy this to EdX:
Memory 1
Address: 0x20000030
0x20000030: 00000001 00000001 00000001 00000001 00000010 00000011 00000010 00000001 00000010 00000011 00000010 00000011 00000010 00000011
0x20000070: 00000001 00000001 00000010 00000011 00000010 00000001 00000001 00000001 00000010 00000011 00000010 00000011 00000001 00000010
0x200000B0: 00000011 00000010 00000011 00000010 00000001 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF
0x200000F0: FFFFFFFF FFFFFFFF 00B450C2 0068A185 001CF248 00D1430B 008593CD 0039E490 00EE3552 00A28615 0056D6D7 000B279A 00BF785C 0073C91F
0x20000130: 0090BB66 00450C29 00F95CEC 00ADADAF 0061FE71 00164F34 00CA9FF6 007EF0B9 0033417C 00E7923F 009BE301 005033C4 00048486 00B8D549
Project Registers
Simulation t1: 2.33727924 sec L:182 C:1
Search the web and Windows
11:42 AM
3/3/2016

HYDER SHAD
MIRA SEHGAL
3/3/16

EE 319K LAB 4

Estimate of Intrusiveness for Debug_Capture Subroutine:

13 instructions x 2 cycles/instruction estimate = 26 cycles

1 cycle = 12.5 nS → 26 cycles = 0.325 uS to execute Debug_Capture.

Percentage overhead:

5000061 cycles between calls → 0.0625007625 seconds between calls

$$\text{Percentage overhead} = 100 \times \left(\frac{0.325\,\text{uS}}{0.0625007625\ \text{seconds}}\right)$$

(time between debug executions with delay.)

Percentage overhead = 0.0005199936561%

∴ minimally intrusive.

Capture timing:
Random samples

1. |x5B5B00 − x0AA700| ⇒ #5228960 (12.5nS) = .06611 2s
2. |xFLL300 − xALOF00| ⇒ #5228960 (12.5nS) = .06611 2s
3. |xB9F200 − x693E00| ⇒ #5228960 (12.5nS) = .06612s

Capture time average is = .06612 seconds