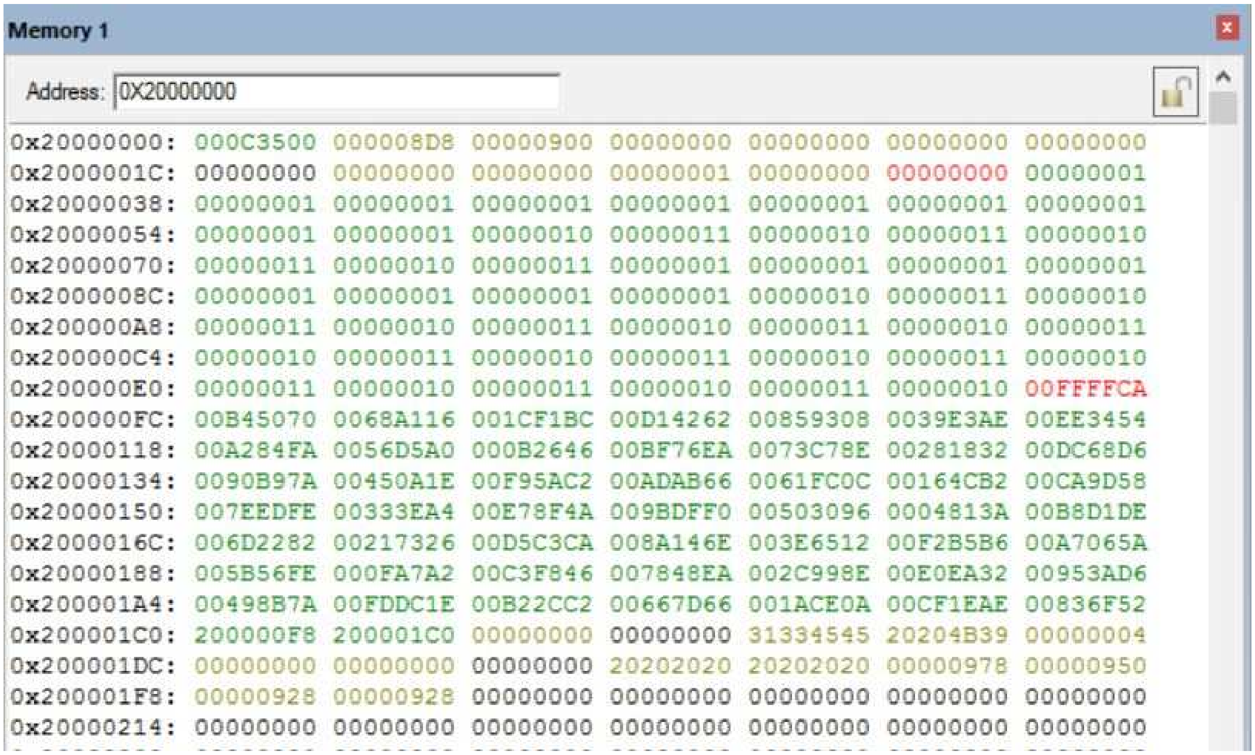
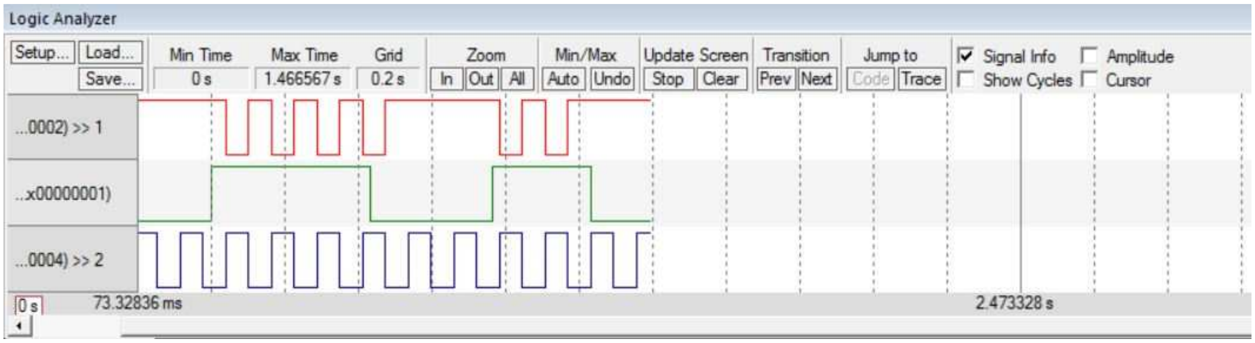


Memory 1

Address: 0X20000000

```
0x20000000: 000C3500 000008D8 00000900 00000000 00000000 00000000 00000000
0x2000001C: 00000000 00000000 00000000 00000001 00000000 00000000 00000001
0x20000038: 00000001 00000001 00000001 00000001 00000001 00000001 00000001
0x20000054: 00000001 00000001 00000010 00000011 00000010 00000011 00000010
0x20000070: 00000011 00000010 00000011 00000001 00000001 00000001 00000001
0x2000008C: 00000001 00000001 00000001 00000001 00000010 00000011 00000010
0x200000A8: 00000011 00000010 00000011 00000010 00000011 00000010 00000011
0x200000C4: 00000010 00000011 00000010 00000011 00000010 00000011 00000010
0x200000E0: 00000011 00000010 00000011 00000010 00000011 00000010 00FFFFCA
0x200000FC: 00B45070 0068A116 001CF1BC 00D14262 00859308 0039E3AE 00EE3454
0x20000118: 00A284FA 0056D5A0 000B2646 00BF76EA 0073C78E 00281832 00DC68D6
0x20000134: 0090B97A 00450A1E 00F95AC2 00ADAB66 0061FC0C 00164CB2 00CA9D58
0x20000150: 007EEDFE 00333EA4 00E78F4A 009BDFF0 00503096 0004813A 00B8D1DE
0x2000016C: 006D2282 00217326 00D5C3CA 008A146E 003E6512 00F2B5B6 00A7065A
0x20000188: 005B56FE 000FA7A2 00C3F846 007848EA 002C998E 00E0EA32 00953AD6
0x200001A4: 00498B7A 00FDDC1E 00B22CC2 00667D66 001ACE0A 00CF1EAE 00836F52
0x200001C0: 200000F8 200001C0 00000000 00000000 31334545 20204B39 00000004
0x200001DC: 00000000 00000000 00000000 20202020 20202020 00000978 00000950
0x200001F8: 00000928 00000928 00000000 00000000 00000000 00000000 00000000
0x20000214: 00000000 00000000 00000000 00000000 00000000 00000000 00000000
```



Logic Analyzer

Min Time: 0 s

Max Time: 1.466567 s

Grid: 0.2 s

☑ Signal Info ☐ Amplitude

☐ Show Cycles ☐ Cursor

...0002) >> 1

...x00000001)

...0004) >> 2

0 s

73.32836 ms

2.473328 s

```
;****************** main.s ***************
; Program written by: Vincent Nguyen, Atreya Misra
; Date Created: 1/24/2015
; Last Modified: 1/24/2015
; Section 4-5pm     TA: Jenny
; Lab number: 4
; Brief description of the program
; If the switch is presses, the LED toggles at 8 Hz
; Hardware connections
; PE0 is switch input (1 means pressed, 0 means not pressed)
; PE1 is LED output (1 activates external LED on protoboard)
; Overall functionality of this system is the similar to Lab 3, with four changes:
; 1-  activate the PLL to run at 80 MHz (12.5ns bus cycle time)
; 2-  initialize SysTick with RELOAD 0x00FFFFFF
; 3-  add a heartbeat to PF2 that toggles every time through loop
; 4-  add debugging dump of input, output, and time
;  Operation
; 1) Make PE1 an output and make PE0 an input.
; 2) The system starts with the LED on (make PE1 =1).
; 3) Wait about 62 ms
; 4) If the switch is pressed (PE0 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
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
X                           EQU     0x0012EBC0
SIZE                        EQU    50
             THUMB
```

```
        EXPORT DataBuffer
        EXPORT TimeBuffer
        EXPORT DataPt [DATA,SIZE=4]
        EXPORT TimePt [DATA,SIZE=4]
        AREA    DATA, ALIGN=4
; You MUST use these two buffers and two variables
; You MUST not change their names
; These names MUST be exported
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


Start
                BL   TExaS_Init                        ; running at 80 MHz, scope voltmeter
                                                        ; on PD3

                LDR R1, =SYSCTL_RCGCGPIO_R             ; initialize Port E
                LDR R0, [R1]
                ORR R0, R0, #0x10                      ; set bit 4 to 1 (Port E)
                STR R0, [R1]
                NOP                                     ; wait for clock
                NOP
                LDR R1, =GPIO_PORTE_DIR_R
                LDR R0, [R1]
                ORR R0, R0, #0x02                      ; set PE1 to output
                BIC R0, R0, #0x01                      ; set PE0 to input
                STR R0, [R1]
                LDR R1, =GPIO_PORTE_AFSEL_R
                LDR R0, [R1]
                BIC R0, R0, #0x03                      ; clear out bits 0 and 1
                STR R0, [R1]
                LDR R1, =GPIO_PORTE_DEN_R
                LDR R0, [R1]
                ORR R0, R0, #0x03                      ; enable bits 0 and 1
                STR R0, [R1]
                LDR R1, =SYSCTL_RCGCGPIO_R  ; initialize Port F
                LDR R0, [R1]
                ORR R0, R0, #0x20                       ; set bit 5 to 1 (Port F)
                STR R0, [R1]
                NOP                                     ; wait for clock
                NOP
                LDR R1, =GPIO_PORTF_DIR_R
                LDR R0, [R1]
                ORR R0, R0, #0x04                       ; set PF2 to output (heartbeat)
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_AFSEL_R
                LDR R0, [R1]
                BIC R0, R0, #0x04                       ; clear out bit 2
                STR R0, [R1]
                LDR R1, =GPIO_PORTF_DEN_R
                LDR R0, [R1]
                ORR R0, R0, #0x04                       ; enable bit 2
                STR R0, [R1]

                BL Debug_Init


                CPSIE  I                                ; TExaS voltmeter, scope runs on
interrupts
loop            BL   Debug_Capture

SUBDELAY
                LDR R10, =X
subbranch
                SUBS R10, #1                            ; Subtract 1 from R10
                BPL subbranch                           ; Subtract 1239999 more times

                LDR R1,=GPIO_PORTF_DATA_R               ; heartbeat
                LDR R0, [R1]
                EOR R0, R0, #0x04                       ; toggle PF2
                STR R0, [R1]

                LDR R1,=GPIO_PORTE_DATA_R
                LDR R0, [R1]
                AND R9, R9, #0                          ; Sets R9 to zero
                ADDS R9, R9, #0x01                      ; Puts bit 0 for into R9 (for checking)
                AND R8, R9, R0                          ; Check to see if bit 0 is one (pressed)
                SUBS R8, R8, #0x01                      ;
                BPL one                                   ; If the value is negative, that means bit
                                                          ; 0 is zero and the switch is pressed
                LDR R1,=GPIO_PORTE_DATA_R                 ; R1 points to GPIO_PORTF_DATA_R
                LDR R0, [R1]                              ; Read GPIO_PORTF_DATA_R into R0
                ORR R0, R0, #0x02                         ; Set bit 1 to 1 (PF1 is output, turns on
                                                          ;the LED)
                STR R0, [R1]
                B       loop
one             LDR R1,=GPIO_PORTE_DATA_R                 ; R1 points to GPIO_PORTF_DATA_R
                LDR R0, [R1]
                EOR R0, R0, #0x02                         ; Toggle (NOT) bit 1
                STR R0, [R1]; Delay
                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 {R0-R4, LR}                          ; preserves registers (debugging)
                LDR R1, =DataBuffer
                LDR R0, [R1]
                MOV R0, #0xFFFFFFFF                       ; no data yet saved
                MOV R2, #200                              ; 50 elements each bit 4 bytes
                MOV R3, #0
Loop1           STR R0, [R1, R3]
                ADD R3, R3, #4                            ; incrementing bit by 4 bytes
                CMP R3, R2

                BNE Loop1                                 ; keeps on checking until you hit the
                                                          ; bottom of the array

                LDR R1, =TimeBuffer
                LDR R0, [R1]
                MOV R0, #0xFFFFFFFF                       ; no data yet saved
                MOV R2, #200                              ; 50 elements each but 4 bytes
                MOV R3, #0
Loop2           STR R0, [R1, R3]
                ADD R3, R3, #4                            ; incrementing but by 4 bytes
                CMP R3, R2
```

```
        BNE Loop2                               ; keeps on checking until you hit the
                                                ; bottom of the array

        LDR R0, =DataBuffer
        LDR R1, =DataPt
        STR R0, [R1]                            ; puts data pointer at the beginning of
                                                ; data buffer
        LDR R2, =TimeBuffer
        LDR R3, =TimePt
        STR R2, [R3]                            ; puts time pointer at the beginning of
                                                ; time buffer
        LDR R1, =NVIC_ST_CTRL_R                 ; init SysTick
        MOV R0, #0                              ; clear enable
        STR R0, [R1]
        LDR R1, =NVIC_ST_RELOAD_R
        LDR R0, =0x00FFFFFF                     ; maximum reload value
        STR R0, [R1]                            ; reload at maximum
        LDR R1, =NVIC_ST_CURRENT_R
        MOV R0, #0                              ; any write to current clears it
        STR R0, [R1]
        LDR R1, =NVIC_ST_CTRL_R                 ; enable SysTick with core clock (no
                                                ; interrupts)
        MOV R0, #0x05
        STR R0, [R1]                            ; ENABLE bits set
        POP {R0-R4, PC}

        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-R3, R12, LR}
        LDR R1, =DataBuffer
        ADDS R1, R1, #200                       ; end of the buffer
        LDR R2, =DataPt
        LDR R0, [R2]
        CMP R0, R1                              ; If the data pointer is at the end of the
                                                ; buffer
```

```
                BEQ Full                                        ; exit subroutine (buffers are full)

                LDR R1, =GPIO_PORTE_DATA_R
                LDR R0, [R1]
                BIC R0, R0, #0xFC                               ; mask data only capturing buts 1 and 0
                LSL R2, R0, #4                                  ; shift bit 0 into bit 4 position
                LSR R3, R0, #1                                  ; shift bit 1 into bit 0 position
                AND R2, R2, #0x10
                AND R3, R3, #0x01
                ORR R0, R2, R3
                LDR R1, =DataPt
                LDR R12, [R1]
                STR R0, [R12]                                   ; dump shifted port info into DataBuffer
                ADD R12, #4                                     ; increment DataPt to next address
                STR R12, [R1]

                LDR R1, =NVIC_ST_CURRENT_R
                LDR R0, [R1]
                LDR R1, =TimePt
                LDR R12, [R1]
                STR R0, [R12]                                   ; dump time into TimeBuffer
                ADD R12, #4                                     ; increment TimePt to next address
                STR R12, [R1]

Full            POP {R0-R3, R12, PC}
                BX LR


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

28 instructions in Debug-Capture

62 mS (time estimation between calls, logic analyzer)

28x2 = 56 $\quad$ 56 x 12.5 nS = 700 nS

$$\frac{700\text{ nS} \times 100}{62\text{ mS}} = .00112903 \simeq \boxed{.001\%}$$

0x 00FFFFCA - 0x 00B45070 = 16777162 - 11817072 = 4960090

$$4960090\,(12.5\text{ ns}) = \boxed{62\text{ ms}}$$