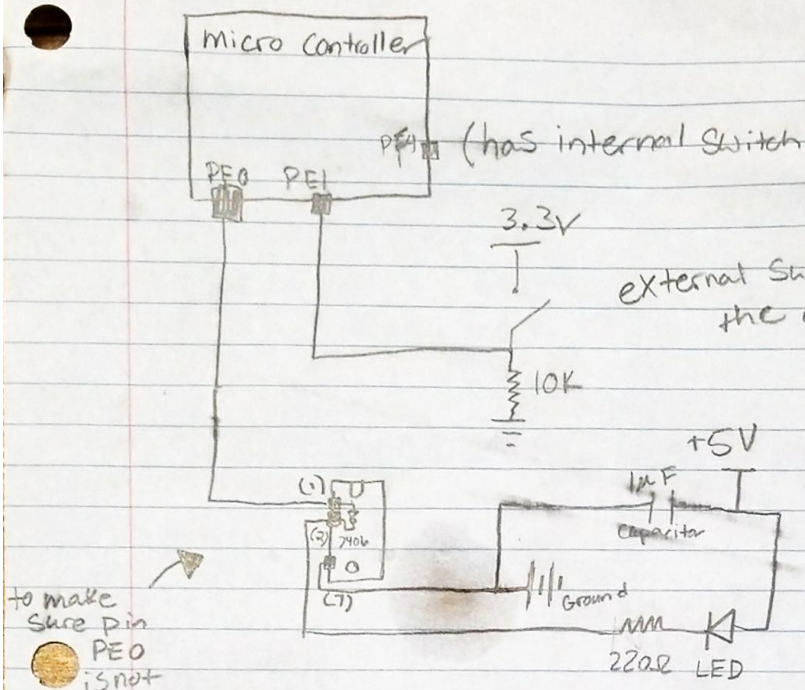
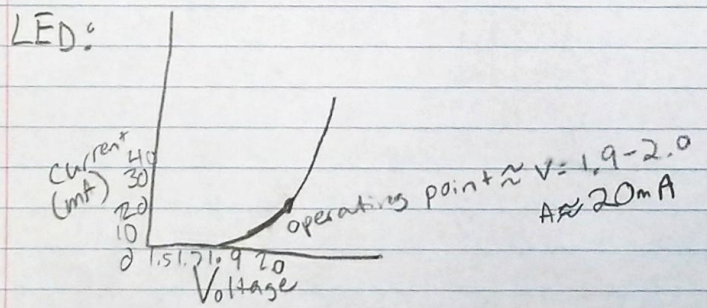
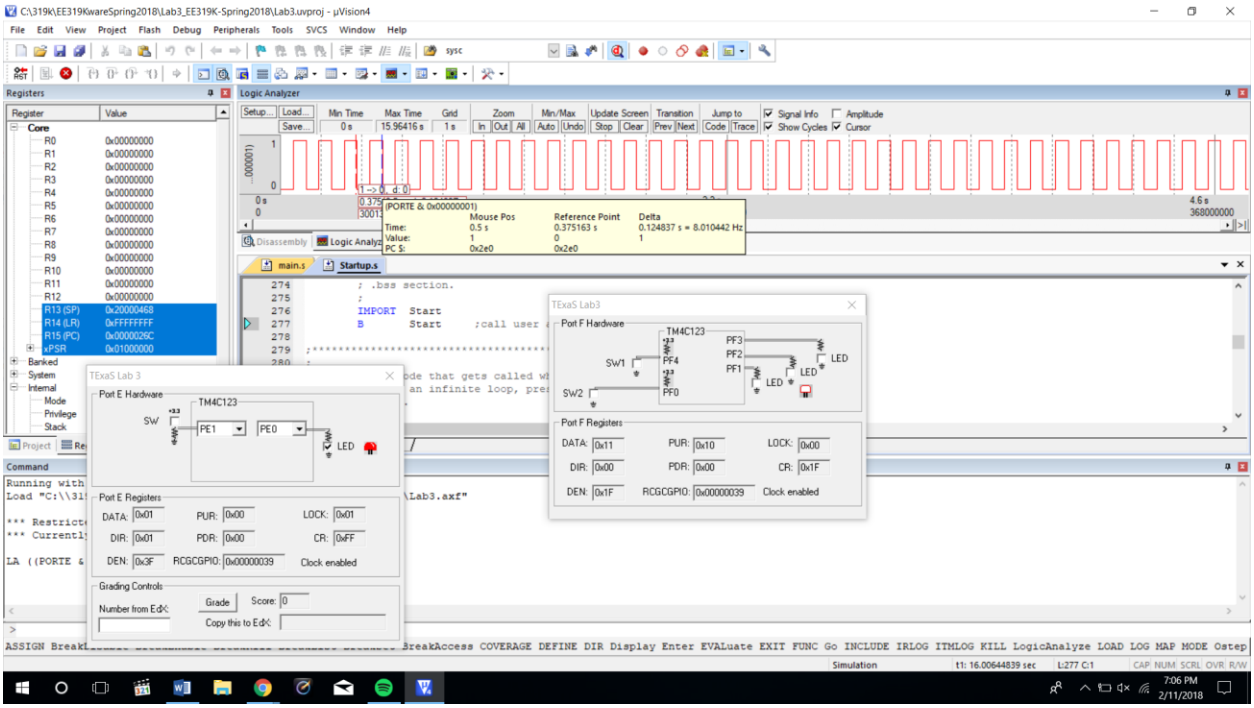
Rachel Clinger
Andrew Kirk



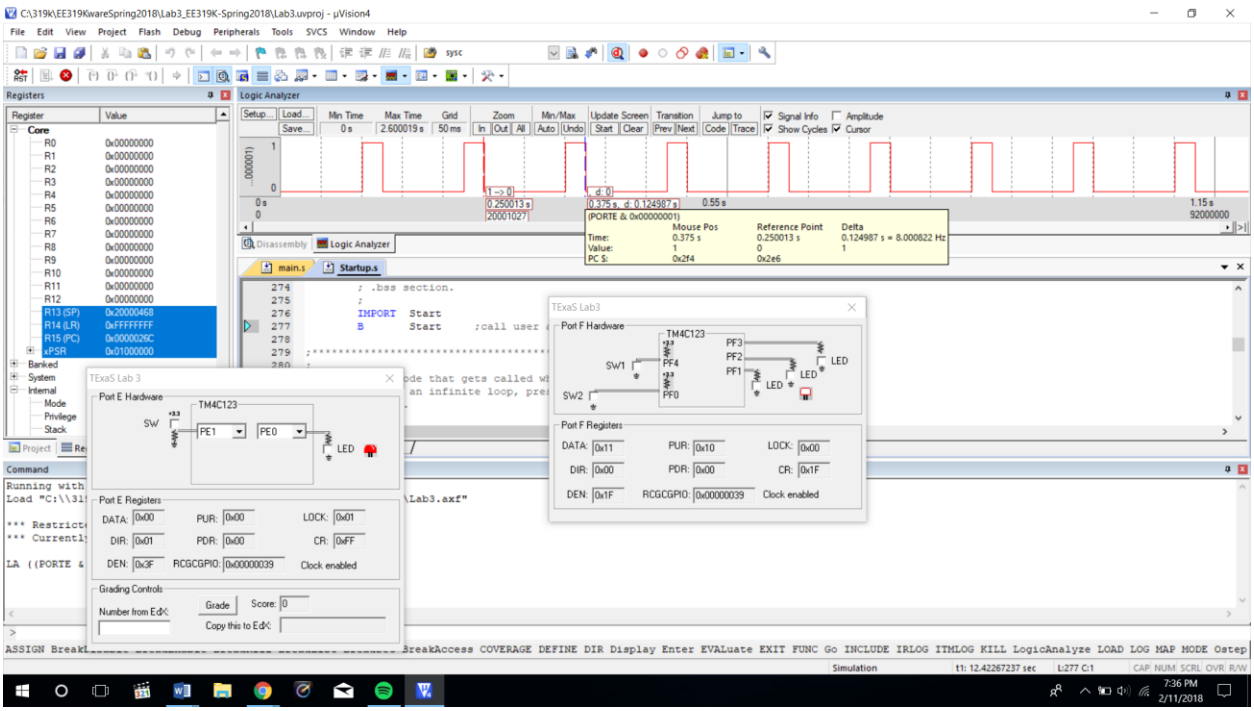
1.26



136



50% duty cycle



20% duty cycle

Rachel Clinger
Andrew Kirk

Switch measurments Table 3.1

parameter	value	units	conditions
10K Ω Resistor	9.82KΩ	ohms	w/ power off disconnected from circuit
Supply Voltage	3.2V	Volts	powered
Input Voltage	0V	Volts	powered, not pressed
resistor current (10KΩ)	0mA	mA	$I = V_{en} / R_1$
Input voltage	3.3V	V	powered, pressed
Resistor Current	0.32mA Calculated: 0.33mA	mA	powered, pressed

LED measurements

Row	Parameter	Value	units	conditions
1)	Resistance of 220Ω resistor	217Ω	Ω	power off & disconnected
2)	+5V power supply	5.06V	Volts	
3)	TM4C123 output, input to 7406	3.53V	volts	with PE0 = 0
4)	7406 output, V_K	5.07V	volts	PE = 0
5)	Led a^+	3.67V	volts	PE = 0
6)	Led voltage	1.5V V	volts	$V_{a+} - V_{k-}$
7.)	LED current	calculated: 6.8mA measured 6.75mA	mA mA	$(V_{+5} - V_{a+})/R19$
8)	TM4C123 Out VPE0 in 7406	0.13V	V	PE0 = 1
9)	7406 Out V_K	2.03V	V	PE0 = 1
10)	LED a^+	0.13V	V	PE0 = 1
11)	LED Voltage	1.9V	V	$V_{a+} - V_{k-}$
12)	LED current	22.4mA 22.2	mA	$(V_{+5} - V_{a+})/R19$

ASSEMBLY:

```
;****************** main.s ***************
; Program written by: Rachel Clinger, Andrew Kirk
; Date Created: 2/4/2017
; Last Modified: 1/15/2018
; Brief description of the program
;   The LED toggles at 8 Hz and a varying duty-cycle
; Hardware connections (External: One button and one LED)
;  PE1 is Button input  (1 means pressed, 0 means not pressed)
;  PE0 is LED output (1 activates external LED on protoboard)
;  PF4 is builtin button SW1 on Launchpad (Internal)
;       Negative Logic (0 means pressed, 1 means not pressed)
; Overall functionality of this system is to operate like this
;   1) Make PE0 an output and make PE1 and PF4 inputs.
;   2) The system starts with the the LED toggling at 8Hz,
;      which is 8 times per second with a duty-cycle of 20%.
;      Therefore, the LED is ON for (0.2*1/8)th of a second
;      and OFF for (0.8*1/8)th of a second.
;   3) When the button on (PE1) is pressed-and-released increase
;      the duty cycle by 20% (modulo 100%). Therefore for each
;      press-and-release the duty cycle changes from 20% to 40% to 60%
;      to 80% to 100%(ON) to 0%(Off) to 20% to 40% so on
;   4) Implement a "breathing LED" when SW1 (PF4) on the Launchpad is pressed:
;      a) Be creative and play around with what "breathing" means.
;         An example of "breathing" is most computers power LED in sleep mode
;         (e.g., https://www.youtube.com/watch?v=ZT6siXyIjvQ).
;      b) When (PF4) is released while in breathing mode, resume blinking at 8Hz.
;         The duty cycle can either match the most recent duty-
;         cycle or reset to 20%.
```

```
;      TIP: debugging the breathing LED algorithm and feel on the simulator is impossible.
; PortE device registers
GPIO_PORTE_DATA_R  EQU 0x400243FC
GPIO_PORTE_DIR_R   EQU 0x40024400
GPIO_PORTE_AFSEL_R EQU 0x40024420
GPIO_PORTE_DEN_R   EQU 0x4002451C
; PortF device registers
GPIO_PORTF_DATA_R  EQU 0x400253FC
GPIO_PORTF_DIR_R   EQU 0x40025400
GPIO_PORTF_AFSEL_R EQU 0x40025420
GPIO_PORTF_PUR_R   EQU 0x40025510
GPIO_PORTF_DEN_R   EQU 0x4002551C
GPIO_PORTF_LOCK_R  EQU 0x40025520
GPIO_PORTF_CR_R    EQU 0x40025524
GPIO_LOCK_KEY      EQU 0x4C4F434B  ; Unlocks the GPIO_CR register
SYSCTL_RCGCGPIO_R  EQU 0x400FE608


    IMPORT  TExaS_Init
    THUMB
    AREA    DATA, ALIGN=2
;global variables go here

    AREA    |.text|, CODE, READONLY, ALIGN=2
    THUMB
    EXPORT  Start
Start
 ; TExaS_Init sets bus clock at 80 MHz
    BL  TExaS_Init ; voltmeter, scope on PD3
```

```
; Initialization goes here

LDR R1, =SYSCTL_RCGCGPIO_R          ;activate clock for Port F and E
  LDR R0, [R1]
  ORR R0, R0, #0x30          ; set bit 5 and 4 to turn on clock
  STR R0, [R1]
  NOP
       NOP
       NOP
  NOP                    ; allow time for clock to finish
  LDR R1, =GPIO_PORTE_DIR_R    ; set direction register
  MOV R0,#0x01               ; PE1 is input, PE0 is output
  STR R0, [R1]

  LDR R1, =GPIO_PORTE_DEN_R    ; enable Port E digital port
  MOV R0, #0xFF              ; 1 means enable digital I/O
  STR R0, [R1]

  LDR R1, =GPIO_PORTF_LOCK_R   ; unlock the lock register
  LDR R0, =0x4C4F434B          ; unlock GPIO Port F Commit Register
  STR R0, [R1]
  LDR R1, =GPIO_PORTF_CR_R     ; enable commit for Port F
  MOV R0, #0xFF              ; 1 means allow access
  STR R0, [R1]
  LDR R1, =GPIO_PORTF_DIR_R    ; set direction register
  MOV R0,#0x00               ; PF4 input
  STR R0, [R1]
  LDR R1, =GPIO_PORTF_PUR_R    ; pull-up resistors for PF4,PF0
  MOV R0, #0x10              ; enable weak pull-up PF4
```

```
   STR R0, [R1]
   LDR R1, =GPIO_PORTF_DEN_R      ; enable Port F digital port
   MOV R0, #0xFF                  ; 1 means enable digital I/O
   STR R0, [R1]
         AND R5, R5, #0
         AND R6, R6, #0
         LDR R6, =500000
         AND R7, R7, #0
         LDR R7, =2500001
    CPSIE  I    ; TExaS voltmeter, scope runs on interrupts
loop
                  LDR R1, =GPIO_PORTF_DATA_R ;check if breathing switch is pressed
                  LDR R0, [R1]
                  ANDS R3, R0, #0x10
                  BEQ breathing
                  LDR R1, =GPIO_PORTE_DATA_R ; check if regular switch is pressed
                  LDR R0, [R1]
                  ANDS R3, R0, #2
                  BNE conston
                  ADDS R5, R5, #0                  ; if its the first run, start with everything off
                  BEQ constoff
                  SUBS R9, R5, #6                  ; if it has gotten to 100% on, reset to 0%
                  BEQ setR5
                  B continue                       ; continue to duty cycle code



breathing                                          ;breathing code
```

```
                LDR R1, =GPIO_PORTE_DATA_R
                LDR R0, [R1]
                LDR R10, =2500          ;time is small so the blinking is faster than the eye can
see
                LDR R11, =250000
                MOV R4, #0                      ; counter
breathestart                                    ;increasing breathe
                LDR R1, =GPIO_PORTE_DATA_R
                LDR R0, [R1]
                SUBS R3, R4, #100       ;if it is on 100%, go to decrease
                BEQ bdecrease
                ANDS R3, R0, #1                 ; if LED is currently off, go to second wait
                BEQ wait2B



                MUL R2, R10, R4
breathloop      SUBS R2,R2,#0x01        ;smaller to larger wait time
                        BNE  breathloop
                EOR R0, R0, #0x01
                STR R0, [R1]

                B contB

wait2B  ADD R2, R11, #0                         ;2500000 into R2 FOR BREATHING ONLY
                MUL R8, R10, R4                         ;multiply by counter
                SUB R2, R2, R8                  ;subract by counter - larger to smaller wait time
waitloop2B              SUBS R2,R2,#0x01        ;FOR BREATHING ONLY
                                BNE  waitloop2B
                EOR R0, R0, #0x01
```

```
                STR R0, [R1]

contB   ADD R4, R4, #1                  ;check if PF4 is released to go back to main function
                LDR R1, =GPIO_PORTF_DATA_R
                LDR R0, [R1]
                ANDS R3, R0, #0x10
                BNE loop
                B breathestart



bdecrease
breathestartD   ;decreasing breathe
                LDR R1, =GPIO_PORTE_DATA_R      ;same as breathing increase, but instead
decrement counter and go from larger to smaller wait vs smaller to larger wait
                LDR R0, [R1]
                ADDS R3, R4, #0
                BEQ breathestart
                ANDS R3, R0, #1
                BEQ wait2BD
                MUL R2, R10, R4
breathloopD     SUBS R2,R2,#0x01                ;same as breathlooop for increase
                        BNE  breathloopD
                EOR R0, R0, #0x01
                STR R0, [R1]

                B contBD

wait2BD         ADD R2, R11, #0                 ;2500000 into R2 FOR BREATHING ONLY
```

```
                MUL R8, R10, R4                         ;multiply by counter
                SUB R2, R2, R8                  ;subract by counter
                ADD R2, R2, #1                  ;since it is 1, when R2 approaches R8, we must
add 1 so that R2 does not become negative
waitloop2BD     SUBS R2,R2,#0x01        ;FOR BREATHING ONLY
                        BNE  waitloop2BD
                EOR R0, R0, #0x01
                STR R0, [R1]

contBD  SUB R4, R4, #1                  ;decrement, and continue
                LDR R1, =GPIO_PORTF_DATA_R
                LDR R0, [R1]
                ANDS R3, R0, #0x10              ;check if the switch is released
                BNE loop
                B breathestartD

setR5   SUB R5, R5, #6
                B continue


constonLDR R1, =GPIO_PORTE_DATA_R       ;stays constanly on when pressed and held
                LDR R0, [R1]
                ORR R0, R0, #1
                STR R0, [R1]
                ANDS R3, R0, #2                                 ;checks if button is released
                BEQ     incr
                B conston
constoff        LDR R1, =GPIO_PORTE_DATA_R
```

```
                LDR R0, [R1]                    ;keeps the led cnstantly off - initially
and after it has been on 100% of the time
                BIC R0, R0, #1
                STR R0, [R1]
                ANDS R3, R0, #2                 ;tests if button has been
pressed, if so, go to conston
                BNE     conston
                LDR R1, =GPIO_PORTF_DATA_R
                LDR R0, [R1]
                ANDS R3, R0, #0x10              ;check if PF4 is pressed, if so, go to
breating
                BEQ breathing
                B constoff
incr    ADD R5, R5, #1                  ;increment counter
                B continue

continue
                LDR R1, =GPIO_PORTE_DATA_R
                LDR R0, [R1]

                ANDS R3, R0, #1                 ;if LED is currently off, go to second wait
time
                BEQ wait2

                MUL R2, R6, R5                  ; since R5 is 20% of the 0.125 seconds, it
controls the duty cycle
waitloop        SUBS R2,R2,#0x01                ;how long on
                        BNE  waitloop
                EOR R0, R0, #0x01
```

```
                STR R0, [R1]
                B loop

wait2   ADD R2, R7, #0                  ;2500000 into R2
                MUL R8, R6, R5                  ;multiply by counter
                SUB R2, R2, R8                  ;subract by counter
waitloop2               SUBS R2,R2,#0x01                ;how long off
                                BNE  waitloop2
                EOR R0, R0, #0x01
                STR R0, [R1]
   B    loop

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