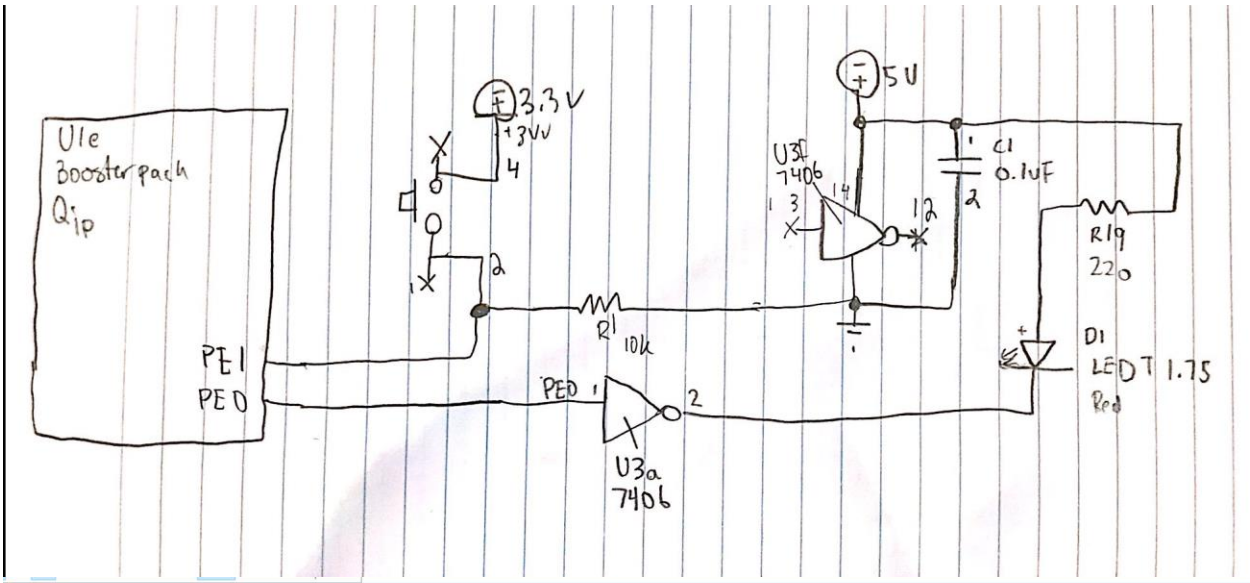
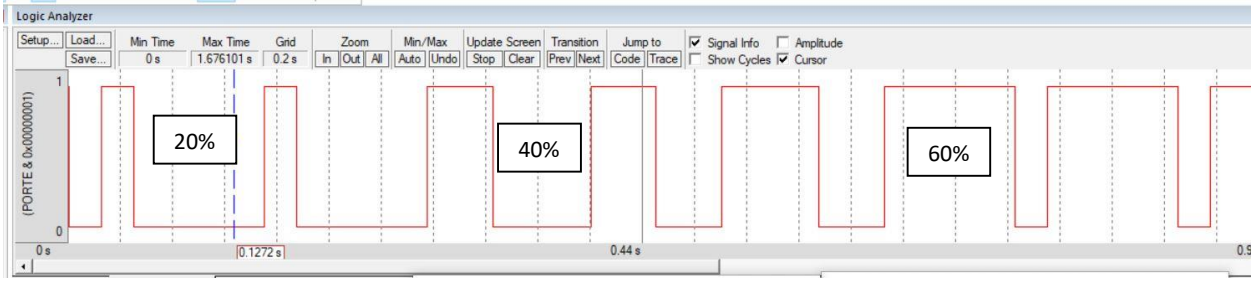
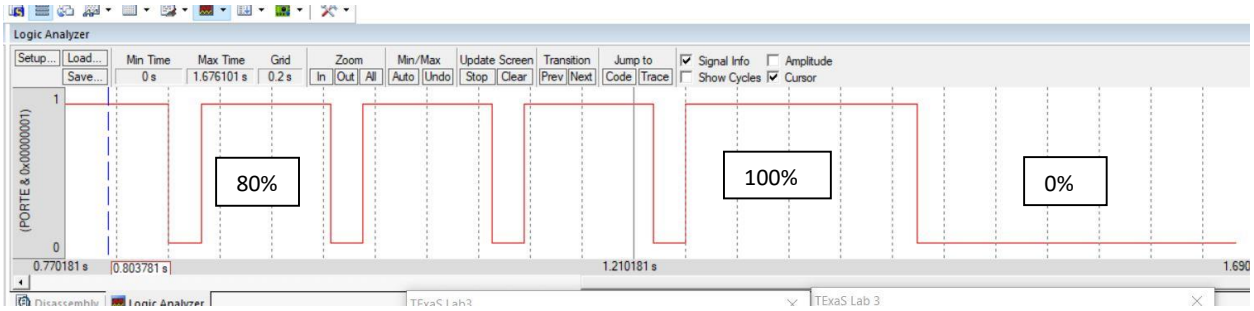
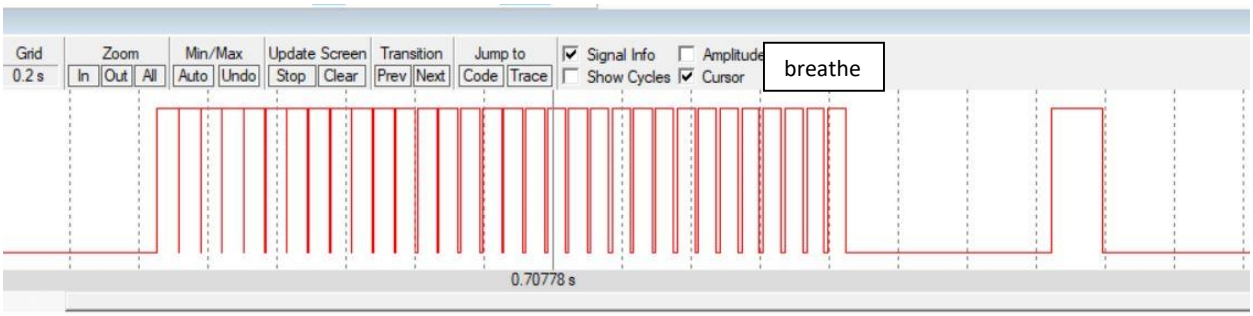

U1e
Booster pack
Qip
PE1
PE0
3.3V
+3VV
5V
U3b
7406
C1
0.1uF
R19
220
D1
LED T 1.75
Red
R1
10k
PE0
U3a
7406


Logic Analyzer
Setup...
Load...
Save...
Min Time
0 s
Max Time
1.676101 s
Grid
0.2 s
Zoom
In
Out
All
Min/Max
Auto
Undo
Update Screen
Stop
Clear
Transition
Prev
Next
Jump to
Code
Trace
Signal Info
Amplitude
Show Cycles
Cursor
(PORTE & 0x00000001)
1
0
20%
40%
60%
0 s
0.1272 s
0.44 s


Logic Analyzer
Setup...
Load...
Save...
Min Time
0 s
Max Time
1.676101 s
Grid
0.2 s
Zoom
In
Out
All
Min/Max
Auto
Undo
Update Screen
Stop
Clear
Transition
Prev
Next
Jump to
Code
Trace
Signal Info
Amplitude
Show Cycles
Cursor
(PORTE & 0x00000001)
1
0
80%
100%
0%
0.770181 s
0.803781 s
1.210181 s
1.690
Disassembly
Logic Analyzer
TExaS Lab3
TExaS Lab 3



Switch Measurments

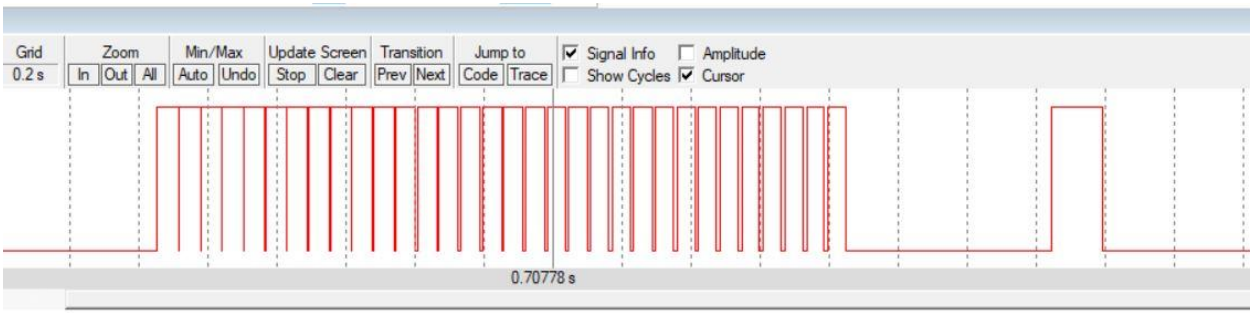
10kΩ R1	9.5kΩ	Ohms	with power off and disconnected from circuit
$V_{+3.3}$	2.975v	Volts	Powerd
V_{PE1}	37mV	Volts	Powerd, but with switch not pressed
Resistor Current	0mA AND 0mA	mA	Powerd but with switch not pressed
V_{PE1}	2.93v	Volts	Powerd and with switch pressed
Resistor Current	calculated: .37mA measured: 0.30mA	mA	Powered and switch pressed

LED Measurements

Row	Parameter	Value	Units	Conditions

1	Resistance of the 220Ω resistor, R19	219.6	ohms	with power off and disconnected from circuit (measured with ohmmeter)
2	+5 V power supply V_{+5}	4.95 V	volts	(measured with voltmeter relative to ground, *notice that the +5V power is not exactly +5 volts*)
3	TM4C123 Output, V_{PE0} input to 7406	3.266 V	volts	with **PE0** = 0 (measured with voltmeter relative to ground)
4	7406 Output, V_{k-} LED k-	0.59 V	volts	with **PE0** = 0 (measured with voltmeter relative to ground)
5	LED a+, V_{a+} Bottom side of R19 (anode side of LED)	2.43 V	volts	with **PE0** = 0 (measured with voltmeter relative to ground)
6	LED voltage	1.86 V	volts	calculated as $V_{a+} - V_{k-}$
7	LED current	Calculated:11.48 mA Measured: 9.7 mA	mA	calculated as $(V_{+5} - V_{a+})$/R19 and measured with an ammeter
8	TM4C123 Output, V_{PE0} input to 7406	1.598 V	volts	with **PE0** = 1 (measured with voltmeter relative to ground)
9	7406 Output, V_{k-} LED k-	2.378 V	volts	with **PE0** = 1 (measured with voltmeter relative to ground)
10	LED a+, V_{a+} Bottom side of R19 (anode side of LED)	3.695 V	volts	with **PE0** = 1 (measured with voltmeter relative to ground)
11	LED voltage	1.32 V	volts	calculated as $V_{a+} - V_{k-}$

12	LED current	Calculated:11.45 mA Measured: 10.67 mA	mA	calculated as $(V_{+5} - V_{a+})$/R19 and measured with an ammeter




```
;****************** main.s ***************
; Program written by: Joshua Kall and Sid Singh
; 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
COUNT0                          EQU 0x00000000
```

```
COUNT2                  EQU 0x00092046
COUNT2HIGH              EQU 0x00247698
COUNT4                  EQU 0x000A21CB
COUNT4HIGH              EQU 0x000F3279
COUNT6                  EQU 0x001B55A9
COUNT6HIGH              EQU 0x001249F0
COUNT8                  EQU 0x00144024
COUNT8HIGH              EQU 0x00050F0D
COUNT10                 EQU 0x002625A0
TWO50THOUSAND           EQU 0x0003D090
THOUSAND                EQU 0x000003E8
FIFTYTHOUSAND           EQU 0x0000C350

    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     R0, = SYSCTL_RCGCGPIO_R       ;Turn on clock for Port E
        MOV     R1, #0x30
        STR     R1, [R0]
        NOP
```

```
        NOP
        NOP
        NOP
        LDR     R1, = GPIO_LOCK_KEY             ;Unlock Port F
        LDR     R0, = GPIO_PORTF_LOCK_R
        STR     R1, [R0]
        LDR     R0, = GPIO_PORTE_DIR_R          ;Enable directions for Port E and F
        MOV     R1, #0x01
        STR     R1, [R0]
        LDR     R0, = GPIO_PORTF_PUR_R          ;Enable pull-up resistor for Port F
        MOV     R1, #0x10
        STR     R1, [R0]
        LDR     R0, = GPIO_PORTE_DEN_R          ;Enable digital logic for Port E and F
        MOV     R1, #0x03
        STR     R1, [R0]
        LDR     R0, = GPIO_PORTF_DEN_R
        MOV     R1, #0x10
        STR     R1, [R0]
        MOV     R9, #0x04
        MOV     R7, #0x01
        MOV     R6, #0x01
    CPSIE  I    ; TExaS voltmeter, scope runs on interrupts
loop
; main engine goes here
        LDR     R0, = GPIO_PORTE_DATA_R
        LDR     R8, [R0]
        BIC     R8, #0xFFFFFFFD
        SUBS    R8, #0x02
        BEQ     inc9link
```

```
        ADDS   R7, #0x00
        BEQ    inc9link
        LDR    R0, = GPIO_PORTF_DATA_R
        LDR    R8, [R0]
        BIC    R8, #0xFFFFFFEF
        ADDS   R8, #0x00
        BEQ    breathelink
return
        MOV    R6, #0x01
        ADDS   R10, R9, #0x00
        SUBS   R10, #0x02
        BEQ    delay0
        SUBS   R10, #0x02
        BEQ    delay2
        SUBS   R10, #0x02
        BEQ    delay4
        SUBS   R10, #0x02
        BEQ    delay6
        SUBS   R10, #0x02
        BEQ    delay8
        SUBS   R10, #0x02
        BEQ    delay10
toggle
        LDR    R0, = GPIO_PORTE_DATA_R
        LDR    R1, [R0]
        BIC    R1, #0xFFFFFFFE
        EOR    R1, #0xFFFFFFFF
        BIC    R1, #0xFFFFFFFE
        STR    R1, [R0]
```

```
    B               loop

breathelink
        B               breathe

inc9link
        B               inc9

delay0
        LDR     R0, = GPIO_PORTE_DATA_R
        LDR     R2, [R0]
        BIC     R2, #0xFFFFFFFF
        STR     R2, [R0]
delaydelay
        LDR     R2, = THOUSAND
delaydelay1
        SUBS    R2, #0x01
        BNE     delaydelay1
delay0next
        LDR     R5, [R0]
        AND     R5, #0x2
        CMP     R5, #2
        BEQ     delay0next1
        B               delay0next
delay0next1
        LDR     R2, [R0]
        BIC     R2, #0xFFFFFFFD
        ADDS    R2, #0x00
        BEQ     delay0next11
```

```
        B               delay0next1
delay0next11
        MOV   R9, #0x04
        B               loop

delay2
        ADDS  R1, #0x00
        BEQ   loop22
        LDR   R2, = COUNT2
loop2
        SUBS  R2, #0x01
        BNE   loop2
        B               toggle
loop22
        LDR   R2, = COUNT2HIGH
loop222
        SUBS  R2, #0x01
        BNE   loop222
        B               toggle

delay4
        ADDS  R1, #0x00
        BEQ   loop44
        LDR   R2, = COUNT4
loop4
        SUBS  R2, #0x01
        BNE   loop4
        B               toggle
loop44
```

```
        LDR     R2, = COUNT4HIGH
loop444
        SUBS    R2, #0x01
        BNE     loop444
        B               toggle

delay6
        ADDS    R1, #0x00
        BEQ     loop66
        LDR     R2, = COUNT6
loop6
        SUBS    R2, #0x01
        BNE     loop6
        B               toggle
loop66
        LDR     R2, = COUNT6HIGH
loop666
        SUBS    R2, #0x01
        BNE     loop666
        B               toggle

delay8
        ADDS    R1, #0x00
        BEQ     loop88
        LDR     R2, = COUNT8
loop8
        SUBS    R2, #0x01
        BNE     loop8
        B               toggle
```

```
loop88
        LDR     R2, = COUNT8HIGH
loop888
        SUBS    R2, #0x01
        BNE     loop888
        B               toggle

delay10
        LDR     R0, = GPIO_PORTE_DATA_R
        MOV     R2, #0x01
        STR     R2, [R0]
        LDR     R2, [R0]
        BIC     R2, #0xFFFFFFFD
        SUBS    R2, #0x02
        BEQ     delay10next
        B               delay10
delay10next
        MOV     R9, #0x02
        LDR     R2, [R0]
        BIC     R2, #0xFFFFFFFD
        ADDS    R2, #0x00
        BEQ     delay10next1
        B               delay10next
delay10next1
        MOV     R2, #0x00
        STR     R2, [R0]
        B               loop

inc9
```

```
        MOV   R7, #0x00
        LDR   R2, [R0]
        BIC   R2, #0xFFFFFFFD
        ADDS  R2, #0x00
        BEQ   inc9_1
        B             return
inc9_1
        MOV   R7, #0x01
        SUBS  R9, #0x0C
        BEQ   init9
        ADDS  R9, #0x0E
        B             loop
init9
        MOV   R9, #0x02
        B             loop

breathe
        ADDS  R6, #0x00                     ;Checks to see if we're in the breathe loop
        BEQ   posorneg
        LDR   R2, = COUNT0                  ;R2 initialized at 0
        B             breathenext
posorneg
        ADDS  R11, #0x00                    ;Checks to see if we were increasing or
decreasing the counter
        BEQ   negbreathe
breathenext
        MOV   R11, #0x01                    ;1 in R11 means were increasing counter
        MOV   R6, #0x00                     ;0 in R6 means were in the breathe loop
        LDR   R0, = GPIO_PORTF_DATA_R
```

```
        LDR     R8, [R0]                ;checking to see if switch was released ----> exit breathe
loop
        BIC     R8, #0xFFFFFFEF
        SUBS    R8, #0x10
        BEQ     return
loopbreathe
        MOV     R1, #850                ;R1 contains value we increment and
decrement by
        ADDS    R2, R1, R2              ;Increment counter
        ADDS    R3, R2, #0x00
        LDR     R12, = TWO50THOUSAND    ;Check to see if counter is at top limit ---> go to negative
section
        SUBS    R3, R3, R12
        BHS     negbreathe
delaybreathe
        LDR     R0, = GPIO_PORTE_DATA_R
        LDR     R1, [R0]                ;check to see if light is on or off
        ADD     R1, #0x00               ;if on ---> delaybreathe1
        SUBS    R1, #1
        BEQ     delaybreathe1
        ADDS    R3, R2, #0x00           ;make a copy of counter so that we can use it without
destroying it
delaybreathenext
        SUBS    R3, #0x01               ;Delay using the current counter for the light
when off
        BNE     delaybreathenext
        B               toggle
delaybreathe1
        ADDS    R3, R2, #0x00           ;make a copy of counter so that we can use it without
destroying it
```

```
        SUBS    R3, R12, R3                     ;Use the difference between top cap and
counter as the counter in order to keep frequency constant

delaybreathe11

        SUBS    R3, #0x01                       ;Delay using the makeshift counter for the light
when on

        BNE     delaybreathe11

        B               toggle


negbreathe

        MOV     R11, #0x00                      ;0 in R11 means were decreasing counter

        MOV     R6, #0x00                       ;0 in R6 means were in the breathe loop

        LDR     R0, = GPIO_PORTF_DATA_R

        LDR     R8, [R0]                ;check to see if switch was released ---> leave breathe
loop

        BIC     R8, #0xFFFFFFEF

        SUBS    R8, #0x10

        BEQ     return

loopnegbreathe

        MOV     R1, #850                        ;R1 contains the value we increment and
decrement by

        SUBS    R2, R2, R1                      ;decrement R2

        BEQ     breathenext                     ;go back to positive loop if we hit the low bound

delaynegbreathe

        LDR     R0, = GPIO_PORTE_DATA_R ;check to see if light is on or off

        LDR     R1, [R0]

        SUBS    R1, #1

        BEQ     delaynegbreathe1        ;if light is on ---> delaynegbreathe1

        ADDS    R3, R2, #0x00           ;make a copy of the counter so we can use it without
destroying it

delaynegbreathenext
```

```
        SUBS    R3, #0x01                       ;delay using the current counter when light is
on
        BNE     delaynegbreathenext
        B               toggle
delaynegbreathe1
        ADDS    R3, R2, #0x00           ;make a copy of the counter so we can use it without
destroying it
        SUBS    R3, R12, R3                     ;Use the difference between top cap and
current counter as new counter
delaynegbreathe11
        SUBS    R3, #0x01                       ;delay using the makeshift counter
        BNE     delaynegbreathe11
        B               toggle

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