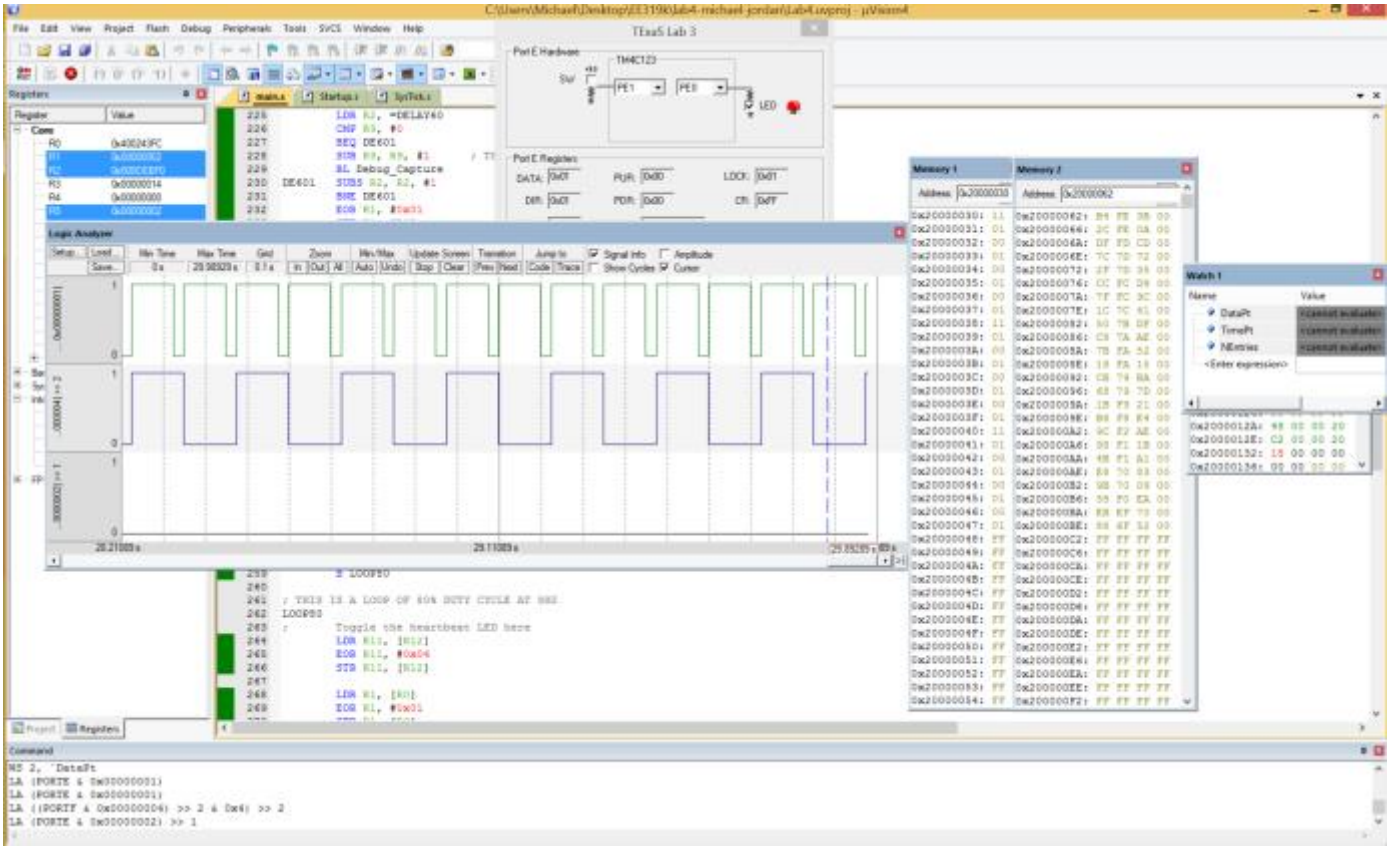

```
;****************** main.s ***************
;
; Program written by: Jordan and Michael Blume
; Date Created: 2/14/2017
; Last Modified: 2/27/2017
; Brief description of the program
;   The LED toggles at 8 Hz and a varying duty-cycle
;   Repeat the functionality from Lab2-3 but now we want you to
;   insert debugging instruments which gather data (state and timing)
;   to verify that the system is functioning as expected.
; 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)
;  PF2 is Blue LED on Launchpad used as a heartbeat
; Instrumentation data to be gathered is as follows:
; After Button(PE1) press collect one state and time entry.
; After Buttin(PE1) release, collect 7 state and
; time entries on each change in state of the LED(PE0):
; An entry is one 8-bit entry in the Data Buffer and one
; 32-bit entry in the Time Buffer
;  The Data Buffer entry (byte) content has:
;    Lower nibble is state of LED (PE0)
;    Higher nibble is state of Button (PE1)
;  The Time Buffer entry (32-bit) has:
;    24-bit value of the SysTick's Current register (NVIC_ST_CURRENT_R)
; Note: The size of both buffers is 50 entries. Once you fill these
;       entries you should stop collecting data
```

```
; The heartbeat is an indicator of the running of the program.
; On each iteration of the main loop of your program toggle the
; LED to indicate that your code(system) is live (not stuck or dead).

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_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
SYSCTL_RCGCGPIO_R  EQU 0x400FE608
NVIC_ST_CURRENT_R     EQU 0xE000E018
DELAY10   EQU  0x00003D000 ; this is 12.5ms which is 10% of a 8HZ frequency
DELAY80   EQU  0x0001E8000
DELAY20   EQU  0x00007A000 ; this is 25ms which is 20% of a 8HZ frequency
DELAY40   EQU  0X0000F4000
DELAY60   EQU  0x00016E000
DELAY5   EQU  0x0000030CC  ; this is 0.625ms which is 5% of a 80HZ frequency
DELAY100  EQU  0x000262000
DELAY1    EQU     0x0000009C3
DELAY1001  EQU  0x00003D02D



; RAM Area
      AREA    DATA, ALIGN=2
DataBuffer   SPACE  50
TimeBuffer   SPACE  50*4
DataPt   SPACE     4
TimePt   SPACE   4
NEntries  DCB       0

;-UUU-Declare  and allocate space for your Buffers
;    and any variables (like pointers and counters) here

; ROM Area
      IMPORT  TExaS_Init
      IMPORT  SysTick_Init
;-UUU-Import routine(s) from other assembly files (like SysTick.s) here
      AREA    |.text|, CODE, READONLY, ALIGN=2
      THUMB
      EXPORT  Start

Start
  BL  TExaS_Init ; voltmeter, scope on PD3

  LDR R0, =SYSCTL_RCGCGPIO_R  ; Turn on the clock for Port E and Port F
  LDR R1, [R0]
  ORR R1, #0x30
  STR R1, [R0]
  NOP
  NOP
```

```
  LDR R0, =GPIO_PORTE_DIR_R
  LDR R1, [R0]
  BIC R1, #0x02   ; Make PE1 an input = 0
  ORR R1, #0x01 ; Make PE0 an output = 1
  STR R1, [R0]

  LDR R0, =GPIO_PORTF_DIR_R
  LDR R1, [R0]
  ORR R1, #0x04 ; Make PF2 an output = 1
  STR R1, [R0]

  LDR R0, =GPIO_PORTE_AFSEL_R
  LDR R1, [R0]
  BIC R1, #0x03  ; Turn off alternate functions for Port E
  STR R1, [R0]

  LDR R0, =GPIO_PORTF_AFSEL_R
  LDR R1, [R0]
  BIC R1, #0x04  ; Turn off alternate functions for Port F
  STR R1, [R0]

  LDR R0, =GPIO_PORTE_DEN_R
  LDR R1, [R0]
  ORR R1, #0x03   ; Enable PE1,PE0
  STR R1, [R0]

  LDR R0, =GPIO_PORTF_DEN_R
  LDR R1, [R0]
  ORR R1, #0x04   ; Enable PF4
  STR R1, [R0]

  LDR R0, =GPIO_PORTE_DATA_R
  LDR R12, =GPIO_PORTF_DATA_R

  BL Debug_Init

   CPSIE  I    ; TExaS voltmeter, scope runs on interrupts

;R7, R9, AND R10
; THIS IS THE LOOP THAT OF A 20% DUTY CYCLE AT 8HZ
LOOP20
;  Toggle the heartbeat LED here
  LDR R11, [R12]
  EOR R11, #0x04
  STR R11, [R12]

  LDR R1, [R0]
  EOR R1, #0x01
  STR R1, [R0]
  LDR R2, =DELAY20
  CMP R9, #0
  BEQ DE201
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture ; (54 instructions * 12.5ns) = 675ns/125ms * 100 = 0.00108% overhead
DE201  SUBS R2, R2, #1
  BNE DE201
  EOR R1, #0x01
  STR R1, [R0]
```

```
  LDR R2, =DELAY80
  CMP R9, #0
  BEQ DE202
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture
DE202 SUBS R2, R2, #1
  BNE DE202


  LDR R1, [R0]
  AND R5, R1, #0x02 ; R5 contains the current state of the switch (positive logic)
  CMP R6, #0x02 ; Check to see if it has been pressed so we only dump once
  BEQ NEXT20    ; Skip the next few lines if it has been pressed before

  ORR R6, R6, R5 ; If it is pressed put 1 in R6
    CMP R6, #0x02  ; Check to see if it has been pressed for the first time
      BNE NEXT20    ; If not pressed yet, just loop back
  MOV R9, #1    ; If it is pressed for the first time, we want to dump only once, R9 = 1
  B LOOP20

NEXT20 EOR R8, R5, R6 ; If turned on (R6=1) but then is released, this logic will turn R8 into a 1
  CMP R8, #0x02
  BNE LOOP20     ; Keep looping without a dump if it has not been released
  MOV R9, #7     ; Once it is released, we want to dump 7 times, R9 = 7
  AND R6, R6, #0
  B LOOP40


; THIS IS A LOOP OF 40% DUTY CYCLE AT 8HZ
LOOP40
;  Toggle the heartbeat LED here
  LDR R11, [R12]
  EOR R11, #0x04
  STR R11, [R12]

  LDR R1, [R0]
  EOR R1, #0x01
  STR R1, [R0]
  LDR R2, =DELAY40
  CMP R9, #0
  BEQ DE401
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture
DE401  SUBS R2, R2, #1
  BNE DE401
  EOR R1, #0x01
  STR R1, [R0]
  LDR R2, =DELAY60
  CMP R9, #0
  BEQ DE402
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture
DE402 SUBS R2, R2, #1
  BNE DE402

  LDR R1, [R0]
  AND R5, R1, #0x02 ; R5 contains the current state of the switch (positive logic)
  CMP R6, #0x02 ; Check to see if it has been pressed so we only dump once
```

```
  BEQ NEXT40    ; Skip the next few lines if it has been pressed before

  ORR R6, R6, R5 ; If it is pressed put 1 in R6
    CMP R6, #0x02  ; Check to see if it has been pressed for the first time
      BNE NEXT40    ; If not pressed yet, just loop back
  MOV R9, #1     ; If it is pressed for the first time, we want to dump only once, R9 = 1
  B LOOP40

NEXT40 EOR R8, R5, R6 ; If turned on (R6=1) but then is released, this logic will turn R8 into a 1
  CMP R8, #0x02
  BNE LOOP40
  MOV R9, #7     ; Once it is released, we want to dump 7 times, R9 = 7
  AND R6, R6, #0
  B LOOP60

; THIS IS A LOOP OF 60% DUTY CYCLE AT 8HZ
LOOP60
;  Toggle the heartbeat LED here
  LDR R11, [R12]
  EOR R11, #0x04
  STR R11, [R12]


  LDR R1, [R0]
  EOR R1, #0x01
  STR R1, [R0]
  LDR R2, =DELAY60
  CMP R9, #0
  BEQ DE601
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture
DE601  SUBS R2, R2, #1
  BNE DE601
  EOR R1, #0x01
  STR R1, [R0]
  LDR R2, =DELAY40
    CMP R9, #0
  BEQ DE602
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture
DE602 SUBS R2, R2, #1
  BNE DE602


  LDR R1, [R0]
  AND R5, R1, #0x02 ; R5 contains the current state of the switch (positive logic)
  CMP R6, #0x02 ; Check to see if it has been pressed so we only dump once
  BEQ NEXT60    ; Skip the next few lines if it has been pressed before

  ORR R6, R6, R5 ; If it is pressed put 1 in R6
    CMP R6, #0x02  ; Check to see if it has been pressed for the first time
      BNE NEXT60    ; If not pressed yet, just loop back
  MOV R9, #1     ; If it is pressed for the first time, we want to dump only once, R9 = 1
  B LOOP60

NEXT60 EOR R8, R5, R6 ; If turned on (R6=1) but then is released, this logic will turn R8 into a 1
  CMP R8, #0x02
  BNE LOOP60
  MOV R9, #7     ; Once it is released, we want to dump 7 times, R9 = 7
```

```
  AND R6, R6, #0
  B LOOP80

; THIS IS A LOOP OF 80% DUTY CYCLE AT 8HZ
LOOP80
;  Toggle the heartbeat LED here
  LDR R11, [R12]
  EOR R11, #0x04
  STR R11, [R12]

  LDR R1, [R0]
  EOR R1, #0x01
  STR R1, [R0]
  LDR R2, =DELAY80
  CMP R9, #0
  BEQ DE801
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture
DE801  SUBS R2, R2, #1
  BNE DE801
  EOR R1, #0x01
  STR R1, [R0]
  LDR R2, =DELAY20
  CMP R9, #0
  BEQ DE802
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture
DE802 SUBS R2, R2, #1
  BNE DE802

  LDR R1, [R0]
  AND R5, R1, #0x02 ; R5 contains the current state of the switch (positive logic)
  CMP R6, #0x02 ; Check to see if it has been pressed so we only dump once
  BEQ NEXT80    ; Skip the next few lines if it has been pressed before

  ORR R6, R6, R5 ; If it is pressed put 1 in R6
    CMP R6, #0x02  ; Check to see if it has been pressed for the first time
      BNE NEXT80    ; If not pressed yet, just loop back
  MOV R9, #1    ; If it is pressed for the first time, we want to dump only once, R9 = 1
  B LOOP80

NEXT80 EOR R8, R5, R6 ; If turned on (R6=1) but then is released, this logic will turn R8 into a 1
  CMP R8, #0x02
  BNE LOOP80
  MOV R9, #7    ; Once it is released, we want to dump 7 times, R9 = 7
  AND R6, R6, #0
  B LOOP100

; THIS IS A LOOP OF 100% DUTY CYCLE AT 8HZ
LOOP100
;  Toggle the heartbeat LED here
  LDR R11, [R12]
  EOR R11, #0x04
  STR R11, [R12]

  LDR R1, [R0]
  ORR R1, #0x01
  STR R1, [R0]
```

```
  LDR R2, =DELAY100
  CMP R9, #0
  BEQ DE1001
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture

DE1001  SUBS R2, R2, #1
  BNE DE1001

  LDR R1, [R0]
  AND R5, R1, #0x02 ; R5 contains the current state of the switch (positive logic)
  CMP R6, #0x02 ; Check to see if it has been pressed so we only dump once
  BEQ NEXT100    ; Skip the next few lines if it has been pressed before

  ORR R6, R6, R5 ; If it is pressed put 1 in R6
    CMP R6, #0x02  ; Check to see if it has been pressed for the first time
      BNE NEXT100    ; If not pressed yet, just loop back
  MOV R9, #1     ; If it is pressed for the first time, we want to dump only once, R9 = 1
  B LOOP100

NEXT100 EOR R8, R5, R6 ; If turned on (R6=1) but then is released, this logic will turn R8 into a 1
  CMP R8, #0x02
  BNE LOOP100
  MOV R9, #7     ; Once it is released, we want to dump 7 times, R9 = 7
  AND R6, R6, #0
  B LOOP0

; THIS IS A LOOP OF 0% DUTY CYCLE AT 8HZ
LOOP0
;  Toggle the heartbeat LED here
  LDR R11, [R12]
  EOR R11, #0x04
  STR R11, [R12]

  LDR R1, [R0]
  BIC R1, #0x01
  STR R1, [R0]

  LDR R2, =DELAY100
  CMP R9, #0
  BEQ DE01
  SUB R9, R9, #1  ; This would dump R9 times
  BL Debug_Capture

DE01 SUBS R2, R2, #1
  BNE DE01

  LDR R1, [R0]
  AND R5, R1, #0x02 ; R5 contains the current state of the switch (positive logic)
  CMP R6, #0x02 ; Check to see if it has been pressed so we only dump once
  BEQ NEXT0    ; Skip the next few lines if it has been pressed before


  ORR R6, R6, R5 ; If it is pressed put 1 in R6
    CMP R6, #0x02  ; Check to see if it has been pressed for the first time
      BNE NEXT0    ; If not pressed yet, just loop back
  MOV R9, #1     ; If it is pressed for the first time, we want to dump only once, R9 = 1
```

```
  B LOOP0

NEXT0 EOR R8, R5, R6 ; If turned on (R6=1) but then is released, this logic will turn R8 into a 1
  CMP R8, #0x02
  BNE LOOP0
  MOV R9, #7     ; Once it is released, we want to dump 7 times, R9 = 7
  AND R6, R6, #0
  B LOOP20

Debug_Init

  PUSH {R0,R1,R2,R3,R4,LR}
  LDR R1, =DataBuffer
  LDR R2, =DataPt
  STR R1, [R2]
  LDR R1, =TimeBuffer
  LDR R2, =TimePt
  STR R1, [R2]

  MOV R0, #50
  LDR R1, =DataBuffer
  LDR R2, =TimeBuffer
  LDR R3, =0xFFFFFFFF
  MOV R4, #0xFF
Fill_Array
  STRB R4, [R1]
  STR R3, [R2]
  ADD R1, R1, #1
  ADD R2, R2, #4
  SUBS R0, R0, #1
  BNE Fill_Array

  BL SysTick_Init

  POP {R0,R1,R2,R3,R4,LR}
  BX LR


Debug_Capture

  PUSH {R0, R1, R2, R3, R4, R5, R6, R7, R8, LR}
  LDR R0, =NEntries
  LDRB R1, [R0]
  CMP R1, #50
  BEQ Done_Capture

  LDR R2, =GPIO_PORTE_DATA_R
  LDR R3, [R2]
  LDR R4, =NVIC_ST_CURRENT_R
  LDR R5, [R4]

  AND R6, R3, #0x01
  AND R7, R3, #0x02
  LSL R7, #3
  ORR R3, R6, R7

  LDR R6, =DataPt ; Store into DataBuffer, increment pointer by 1
  LDR R7, [R6]
```

```
  STRB R3, [R7]
  ADD R7, R7, #1
  STR R7, [R6]

  LDR R6, =TimePt ; Store into TimeBuffer, increment pointer by 4
  LDR R7, [R6]
  STR R5, [R7]
  ADD R7, R7, #4
  STR R7, [R6]

  ADD R1, R1, #1 ; Update NEntries
  STRB R1, [R0]
Done_Capture
  POP {R0, R1, R2, R3, R4, R5, R6, R7, R8, LR}
  BX LR

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

Estimation of Intrusiveness:

(27 instructions * 2) *12.5ns/instruction = 675ns/125ms * 100 = 0.00108%

Paste from the saved File (50 entries)				count:	50						
:020000042000DA								12.5	<- Time per tick		
:0E0062000E7A62009677A90049776C00E6F6E8	0E7A6200	9677A900	49776C00		Adjust-endia	Data	Differences	Time(ms)			
:1000700010009 9F6D30036767800E9753B0086F5D6	E6F61000	99F6D300	36767800	E9753B00	00627A0E	6453774					
:10008000DF0042E49F009B63D6004EE37A00EBE280	86F5DF00	42E49F00	9B63D600	4EE37A0(	00A97796	11106198	12124792	151.5599	<-time from press to release		
:100090003D009E62E2003B62A500EEE149008BE17B	EBE23D00	9E62E200	3B62A500	EEE14900	006C7749	7108425	3997773	49.9721625	<- first 6 time differences		
:1000A0000C00BFE0AA00F9DF4800ACDFCE00495FI	8BE10C00	BFE0AA00	F9DF4800	ACDFCE(	0010F6E6	1111782	5996643	74.9580375			
:1000B000B000FC5E360099DE17004CDE9D00E95D6	495FB000	FC5E3600	99DE1700	4CDE9D0	00D3F699	13891225	3997773	49.9721625			
:1000C0007F003FDDB50078DC5300145CBB00B0DB8	E95D7F00	3FDDB500	78DC5300	145CBB0(	00787636	7894582	5996643	74.9580375	ON	50/125=40%	
:1000D00022004C5B8A00E8DAF100845A590020DAE	B0DB2200	4C5B8A00	E8DAF10(	845A5900	003B75E9	3896809	3997773	49.9721625	OFF	75/125=60%	
:1000E000C0008559F700D35895006FD8FC000B5815	20DAC000	8559F700	D3589500	6FD8FC0(	00DFF586	14677382	5996643	74.9580375			
:1000F0006400A7D7CB0043573300DFD69A007B5666	0B586400	A7D7CB00	43573300	DFD69A0	009FE442	10478658	4198724				
:100100000200A6D50700DB540D008ED4EE002BD4E(	7B560200	A6D50700	DB540D0(	8ED4EE0(	00D6639B	14050203	13205671	165.070888	<-time from press to release		
:10011000740 0DE5356007B53DC002ED3BD00CBD2D	2BD47400	DE535600	7B53DC0(	2ED3BD0	007AE34E	8053582	5996621	74.9577625	<- next 6 time differences		
:0A012000430033D14E006DD0EC0017	CBD24300	33D14E00	6DD0EC00		003DE2EB	4055787	3997795	49.9724375			
:00000001FF					00E2629E	14836382	5996621	74.9577625			
					00A5623B	10838587	3997795	49.9724375	ON	75/125=60%	
					0049E1EE	4841966	5996621	74.9577625	OFF	50/125=40%	
					000CE18B	844171	3997795	49.9724375			
					00AAE0BF	11198655	6422732				
					0048DFF9	4775929	6422726	80.284075	<-time from press to release		
					00CEDFAC	13557676	7995469	99.9433625	<- next 6 time differences		
					00B05F49	11558729	1998947	24.9868375			
					00365EFC	3563260	7995469	99.9433625			
					0017DE99	1564313	1998947	24.9868375	ON	100/125=80%	
					009DDE4C	10346060	7995469	99.9433625	OFF	25/125=20%	
					007F5DE9	8347113	1998947	24.9868375			
					00B5DD3F	11918655	13205674				
					0053DC78	5495928	6422727	80.2840875	<-time from press to release		
					00BB5C14	12278804	9994340	124.92925	<- next 6 time differences		
					0022DBB0	2284464	9994340	124.92925			
					008A5B4C	9067340	9994340	124.92925			
					00F1DAE8	15850216	9994340	124.92925	ON	100%	
					00595A84	5855876	9994340	124.92925			
					00C0DA20	12638752	9994340	124.92925			
					00F75985	16210309	13205659				
					009558D3	9787603	6422706	80.283825	<-time from press to release		
					00FCD86F	16570479	9994340	124.92925	<- next 6 time differences		
					0064580B	6576139	9994340	124.92925			
					00CBD7A7	13359015	9994340	124.92925			
					00335743	3364675	9994340	124.92925	OFF	100%	
					009AD6DF	10147551	9994340	124.92925			
					0002567B	153211	9994340	124.92925			
					0007D5A6	513446	16416981				
					000D54DB	873691	16416971	205.212138	<-time from press to release		
					00EED48E	15651982	1998925	24.9865625	<- next 6 time differences		
					0074D42B	7656491	7995491	99.9436375			
					005653DE	5657566	1998925	24.9865625			
					00DC537B	14439291	7995491	99.9436375	ON	25/125=20%	
					00BDD32E	12440366	1998925	24.9865625	OFF	100/125=80%	
					0043D2CB	4444875	7995491	99.9436375			
					004ED133	5165363	16056728				
					00ECD06D	15519853	6422726	80.284075			