Lab 2

Michael Arboleda
Lab Section: 7F34

June 14, 2017

b. Answers to all pre-lab questions

1) How many TC0 channels are necessary to control all three LEDs in the uPADs RGB LED

ANS: 3. One for red, one for green, one for blue

2) What would happen if the RGB period was FFFF instead of FF?

ANS: The amount of cycles need restart is higher. Thus the time for a LED to be ion is bigger

c. Problems Encountered

My overflow interrupt would not work, even tho the Flag did go off. I changed the location of the .ORG and it worked

d. Future Work/Applications

IT is clear why interrupts and timer/counters are important. In the future I can learn about the different types of interrupts, not just external and overflow.

e. Schematics

N/A

g. Pseudocode/Flowcharts

Pseudocode for lab3a.asm:

```
MAIN:

    * Equate numbers
    * Set registers to hold constants

    * Call Change_CLK_32HZ subroutine
    * Remap Pin Ports
    * Set PORT D/LED to output
    * Invert Port D
    * Set TOP of PWM
    * Set up Control D for port D
    * Set up Control B for port D
    * Set up Compare Chanel
    * Set up Control D for port D

WHILE(TRUE){}
END


SUBROUTINE Change_CLK_32HZ
    * Enable the new oscillator

    WHILE(OSC FLAG not set){}

    * Write the IOREG signature to the CPU_CCP reg
    * Select the new clock source in the CLK_CTRL reg
    * Return to program
```

Pseudocode for lab3b.asm:

```
MAIN:

* Equate numbers
* Set registers to hold constants
* Call Change_CLK_32HZ subroutine

* Set up LEDS and Switches

* Set up External interrupt

SUBROUTINE BLUE_PWM:
* Remap Pin Ports
* Set PORT D/LED to output
* Invert Port D
* Set TOP of PWM
* Set up Control D for port D
* Set up Control B for port D
* Set up Compare Chanel
* Set up Control D for port D


ISR inc_display:
* INCREMENT counter
* Display on LED
```

Pseudocode for lab3c.asm:

```
MAIN:

* Equate numbers
* Set registers to hold constants
* Call Change_CLK_32HZ subroutine

* Set up LEDS and Switches

* Set up External interrupt

WHILE(TRUE){
* Toggle LED
}

END

ISR overflow_logic:
IF(BUTTON PRESSED){
* INCREMENT COUNT
* DISPLAY COUNT
}
* Disable Timer interrupt
* Disable Timer
* Enable external interrupt
* Clear Interupts flags

ISR ext_int_logic
* Disable external interrupt
* Initialize Timer and Timer interrupt
* Clear Flags

SUBROUTINE SET_COUNTER:
* Set TOP(PER) for counter

SUBROUTINE SET_COUNTER_INT:
* Set interrupt for counter
* Run Counter
```

Pseudocode for lab3d.asm:

```
MAIN:

* Equate numbers
* Set registers to hold constants
* Call Change_CLK_32HZ subroutine

* Set up LEDS and Switches

* call PWM_setup

* Set up PORT F interupt

While(TRUE){
* Toggle LED

** Wait for external interupt **
** Run ext_int_logic ISR
*** Wait for overflow interupt
*** Run overflow_logic ISR
*** Wait for 2nd overflow interupt
*** Run overflow_05_logic ISR
}

END

SUBROUTINE PWM_setup:
* Remap Ports
* Set CTRLs

ISR ext_int_logic:
* Disable external interrupt
* call STOP_05_COUNTER Subroutine
* Initialize Timer and Timer interrupt
* Clear Interupts flags

SUBROUTINE STOP_05_COUNTER:
* Disable Timer interrupt
* Disable Timer

SUBROUTINE SET_COUNTER:
* Set up TOP (PER)
```

```
SUBROUTINE SET_COUNTER_INT:
* Set Interupt config
* Start counter

ISR overflow_logic:
If(button is still held){
* increment external button counter
* Display erternal button on LEDs
}

* Call subroutine STOP_COUNTER
* Reset switcher counter
* Enable external interupt
* Set up .05sec timer and interrupt
* Set up compare chanel
* Clear interrupt flags

SUBROUTINE STOP_COUNTER
* Disable Timer interrupt
* Disable Timer

SUBROUTINE SET_05_COUNTER:
* Set up TOP (PER)

SUBROUTINE SET_05_COUNTER_INT:
* Set Interupt config
* Start counter

ISR overflow_05_logic
* Call SET_COMPARE_CH

SUBROUTINE SET_COMPARE_CH:
* Restart counter for PWM_setup
* Set Color Reg
IF(Switches counter = 0){
* Use first color of pattern
* Set Switches counter to 1
}
ELSE{
* Use 2nd color of pattern
* Set Switches counter to 0
}
* Set up Compare Chanel A, B, C
* Run Counter
```

```
SUBROUTINE Set_Pattern:
IF(Counter mod 4 = 0){
* Set all colors to off
}
ELSE IF(Counter mod 4 = 1){
* Set UF colors
}
ELSE IF(Counter mod 4 = 2){
* Set Holiday Colors
}
ELSE IF(Counter mod 4 = 3){
* Set Hulk Colors
}
```

h. Program Code

Code for lab3a.asm:

```
; Lab 3 Part A
; Name:          Michael Arboleda
; Section:       7F34
; TA Name:       Wesley Piard
; Description: Use interuptx for LED
;
; lab3a.asm
; Created: 6/11/2017 1:48:59 AM

.include "ATxmega128A1Udef.inc"

; address equates

; Constant equates
.equ BIT456 = 0x70
.equ new_clock_freq = 0b00000010
.equ port_map_config = 0b00000100
.equ ctrlb_config        = 0b01000011
.equ PORTD_PIN_CTRL = 0b01000000
.equ MAX_PERIOD = 0xFF
.equ BLUE_PERIOD = 0x0F
.equ clk_div = 0b00000111

.ORG 0x0000                     ;Code starts running from address 0x0000.
        rjmp MAIN               ;Relative jump to start of program.

MAIN:
        ; Run at 32MHz
        call Change_CLK_32HZ    ; Change Clk to 32MHz

        ; Remap ports
        ldi R16, port_map_config        ; Load port map config into R16
        sts PORTD_REMAP, R16            ; LOAD port map config

        ; Set PORT D/LED to output
        ldi R16, BIT456  ;load a four bit value (PORTD is only four bits)
        sts PORTD_DIRSET, R16   ;set all the GPIO's in the four bit
                                ; PORTD as outputs

        ; Invert Port D
        ldi R16, PORTD_PIN_CTRL ; LOAD PIN CTRL config
        sts PORTD_PIN0CTRL, R16 ; Invert Pin 0 of Port D
        sts PORTD_PIN1CTRL, R16 ; Invert Pin 1 of Port D
```

```
        sts PORTD_PIN2CTRL, R16 ; Invert Pin 2 of Port D
        sts PORTD_PIN3CTRL, R16 ; Invert Pin 3 of Port D
        sts PORTD_PIN4CTRL, R16 ; Invert Pin 4 of Port D
        sts PORTD_PIN5CTRL, R16 ; Invert Pin 5 of Port D
        sts PORTD_PIN6CTRL, R16 ; Invert Pin 6 of Port D
        sts PORTD_PIN7CTRL, R16 ; Invert Pin 7 of Port D

        ; Set TOP of PWM
        ldi R16, MAX_PERIOD
        ldi R17, 0x00                   ; LOAD R17 with 0x00
        sts TCD0_PER, R16               ; Set Lower Bits of TOP
        sts (TCD0_PER + 1), R17 ; Set Higher Bits of TOP

        ; Set up Control D
        sts TCD0_CTRLD, R17             ; LOAD CTRLD with 0x00

        ; Set up Control B
        ldi R17, ctrlb_config    ; LOAD 0b01000011 into R17
        sts TCD0_CTRLB, R17             ; Store contrl b config

        ; Set up Compare Chanel
        ldi R16, BLUE_PERIOD    ; LOAD r16 with blue time
        ldi R17, 0x00                   ; LOAD R17 with 0x00
        sts TCD0_CCC, R16               ; LOAD compare chanel (lower)
        sts (TCD0_CCC + 1), R17 ; LOAD compare chanel (higher)


        ;control a
        ldi R16, clk_div                ; LOAD 0b00000111
        sts TCD0_CTRLA, R16             ; store clk prescaler in ctrl A

Never_End:
        rjmp Never_End          ; Jump to restart output loop


;********************SUBROUTINES************************************
; Subroutine Name: Change_CLK_32HZ
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
Change_CLK_32HZ:
        ;Push Values
        push R17 ; PUSH r17 to stack
        push R18 ; PUSH r18 to stack
        push R19 ; PUSH r19 to stack
        push R20 ; PUSH r20 to stack
```

```
; Enable the new oscillator
ldi R16, new_clock_freq          ; Load R16 with the clk-freq config 0x02
sts OSC_CTRL, r16                ; Set the clk config

;Wait for the right flag to be set in the OSC_STATUS reg
; While flag is not set
While_32_flag:
        lds R17, OSC_STATUS      ; Load Status Flag
        and R17, R16             ; Bit-mask with 00000010
        cp R17, R16              ; Compare Mask and Value
        brne While_32_flag       ; Restart loop if flag not set

; Write the IOREG signature to the CPU_CCP reg
        ldi R17, CCP_IOREG_gc    ; Load IOREG into R17
        sts CPU_CCP, R17         ; Store IOREG into CPU CCP

;Select the new clock source in the CLK_CTRL reg
        ldi R17, CLK_SCLKSEL_RC32M_gc; load 32 MHz internal osc config
        sts CLK_CTRL, R17        ; Store config in clk control

        ;Pop Values
        pop R20 ; POP r20 from stack
        pop R19 ; POP r19 from stack
        pop R18 ; POP r18 from stack
        pop R17 ; POP r17 from stack
        ret
```

Code for lab3b.asm:

```
; Lab 3 Part B
; Name:          Michael Arboleda
; Section:       7F34
; TA Name:       Wesley Piard
; Description:
;
; lab3b.asm



.include "ATxmega128A1Udef.inc"

; address equates

; Constant equates
.equ BIT456 = 0x70
.equ new_clock_freq = 0b00000010
.equ port_map_config = 0b00000100
.equ ctrlb_config        = 0b01000011
.equ PORTD_PIN_CTRL = 0b01000000
.equ MAX_PERIOD = 0xFF
.equ BLUE_PERIOD = 0x0F
.equ clk_div = 0b00000111
;--------------------------------;
.equ intr_crtl_lvl_config = 0b00000011   ; 0x03
.equ button_PF2 = 0b00000100                        ; 0x04
.equ PORTF_PIN2_CONFIG = 0b00010000                 ; 0x10
.equ PMIC_crtl_lvl_config = 0b00000100   ;

;
.def external_counter = R20

.ORG PORTF_INT0_vect
        rjmp inc_display
.ORG 0x0000                     ;Code starts running from address 0x0000.
        rjmp MAIN               ;Relative jump to start of program.

MAIN:
        ;
        ldi external_counter, 0x00

        ; Run at 32MHz
        call Change_CLK_32HZ     ; Change Clk to 32MHz
```

```
        call BLUE_PWM                   ; Run bluw pulse

        ; Set PORT F
        ldi R16, 0xFF                   ; LOAD r16 with FF
        sts PORTF_DIRCLR, r16    ; Set to write

        ; Set Port C
        ldi R21, 0xFF                   ; LOAD FF to r21
        sts PORTC_DIRSET, R21    ; Set to write
        sts PORTC_OUT, R21

        ; Set PORTF interupt control
        ldi R16, intr_crtl_lvl_config    ; LOAD config into R16
        sts PORTF_INTCTRL, R16                  ; Set interupt control lvl config

        ; Set PORT F MASK
        ldi R16, button_PF2                     ; LOAD 0b00000100 into R16
        sts PORTF_INT0MASK, R16          ; Set intrerupt mask

        ;
        ldi R16, PORT_ISC_FALLING_gc     ; LOAD PFP2 config
        sts PORTF_PIN2CTRL, R16                 ; Set PFP2 config

        ldi R16, PMIC_crtl_lvl_config    ; LOAD PMIC lvl config
        sts PMIC_CTRL, R16                              ; Set PMIC lvl config

        ;
        sei


; infinite loop
Never_End:

        rjmp Never_End          ; Jump to restart output loop


;********************SUBROUTINES************************************
; Subroutine Name: Change_CLK_32HZ
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
Change_CLK_32HZ:
        ;Push Values
        push R17 ; PUSH r17 to stack
        push R18 ; PUSH r18 to stack
        push R19 ; PUSH r19 to stack
```

```
        push R20 ; PUSH r20 to stack

; Enable the new oscillator
ldi R16, new_clock_freq          ; Load R16 with the clk-freq config 0x02
sts OSC_CTRL, r16                ; Set the clk config

;Wait for the right flag to be set in the OSC_STATUS reg
; While flag is not set
While_32_flag:
        lds R17, OSC_STATUS      ; Load Status Flag
        and R17, R16             ; Bit-mask with 00000010
        cp R17, R16              ; Compare Mask and Value
        brne While_32_flag       ; Restart loop if flag not set

; Write the IOREG signature to the CPU_CCP reg
        ldi R17, CCP_IOREG_gc    ; Load IOREG into R17
        sts CPU_CCP, R17         ; Store IOREG into CPU CCP

;Select the new clock source in the CLK_CTRL reg
        ldi R17, CLK_SCLKSEL_RC32M_gc; load 32 MHz internal osc config
        sts CLK_CTRL, R17        ; Store config in clk control

        ;Pop Values
        pop R20 ; POP r20 from stack
        pop R19 ; POP r19 from stack
        pop R18 ; POP r18 from stack
        pop R17 ; POP r17 from stack
        ret




;*********************SUBROUTINES************************************
; Subroutine Name:  BLUE_PWM
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
BLUE_PWM:
        ;Push Values
        push R16 ; PUSH r17 to stack
        push R17 ; PUSH r17 to stack

        ; Remap ports
        ldi R16, port_map_config         ; Load port map config into R16
        sts PORTD_REMAP, R16             ; LOAD port map config
```

```
; Set PORT D/LED to output
ldi R16, BIT456                 ;load a four bit value (PORTD is only four
sts PORTD_DIRSET, R16   ;set all the GPIO's in the four bit
                                                ; PORTD as outputs

; Invert Port D
ldi R16, PORTD_PIN_CTRL ; LOAD PIN CTRL config
sts PORTD_PIN0CTRL, R16 ; Invert Pin 0 of Port D
sts PORTD_PIN1CTRL, R16 ; Invert Pin 1 of Port D
sts PORTD_PIN2CTRL, R16 ; Invert Pin 2 of Port D
sts PORTD_PIN3CTRL, R16 ; Invert Pin 3 of Port D
sts PORTD_PIN4CTRL, R16 ; Invert Pin 4 of Port D
sts PORTD_PIN5CTRL, R16 ; Invert Pin 5 of Port D
sts PORTD_PIN6CTRL, R16 ; Invert Pin 6 of Port D
sts PORTD_PIN7CTRL, R16 ; Invert Pin 7 of Port D

; Set TOP of PWM
ldi R16, MAX_PERIOD
ldi R17, 0x00                   ; LOAD R17 with 0x00
sts TCD0_PER, R16               ; Set Lower Bits of TOP
sts (TCD0_PER + 1), R17 ; Set Higher Bits of TOP

; Set up Control D
sts TCD0_CTRLD, R17             ; LOAD CTRLD with 0x00

; Set up Control B
ldi R17, ctrlb_config   ; LOAD 0b01000011 into R17
sts TCD0_CTRLB, R17             ; Store contrl b config

; Set up Compare Chanel
ldi R16, BLUE_PERIOD    ; LOAD r16 with blue time
ldi R17, 0x00                   ; LOAD R17 with 0x00
sts TCD0_CCC, R16               ; LOAD compare chanel (lower)
sts (TCD0_CCC + 1), R17 ; LOAD compare chanel (higher)


;control a
ldi R16, clk_div                ; LOAD 0b00000111
sts TCD0_CTRLA, R16             ; store clk prescaler in ctrl A


;Pop Values
pop R17 ; POP r17 from stack
pop R16 ; POP r16 from stack
ret
```

```
;********************ISR*************************************
; ISR Name: inc_display
; Inputs: No direct input (from stack)
; Outputs: R20
; Affected: R21
inc_display:
        inc external_counter            ; counter = counter + 1
        mov R21, external_counter       ; LOAD FF to r21
        com R21                                 ; Compliment R21
        sts PORTC_OUT, R21                      ; Display LED config
        ldi R21,  0b00000001            ; LOAD 0x01 into R21
        sts PORTF_INTFLAGS, R21         ; Clear Interupts flags
        reti                                    ; return
```

Code for lab3c.asm:

```
; Lab 3 Part C
; Name:          Michael Arboleda
; Section:       7F34
; TA Name:       Wesley Piard
; Description:
;
; lab3c.asm
; Created: 6/12/2017 5:03:28 AM

.include "ATxmega128A1Udef.inc"

; address equates

; Constant equates
.equ BIT456 = 0x70
.equ new_clock_freq = 0b00000010
.equ port_map_config = 0b00000100
.equ ctrlb_config       = 0b01000011
.equ PORTD_PIN_CTRL = 0b01000000
.equ MAX_PERIOD = 0xFF
.equ BLUE_PERIOD = 0x0F
.equ clk_div = 0b00000111
;-------------------------------------------------;
.equ intr_crtl_lvl_config = 0b00000011  ; 0x03
.equ button_PF2 = 0b00000100                    ; 0x04
.equ PORTF_PIN2_CONFIG = 0b00010000             ; 0x10
.equ PMIC_crtl_lvl_config = 0b00000111  ;
;-------------------------------------------------;
.equ clk_div_timer = 0b00000110
.equ delay_cycles = 0xFF
.equ delay_cycles_top = 0x00
.equ INTCTRLA_config = 0b00000011

; Reg Defs
.def external_counter = R20


.ORG PORTF_INT0_vect
        rjmp ext_int_logic

.ORG  TCC0_OVF_vect
        rjmp overflow_logic

.ORG 0x0000                     ;Code starts running from address 0x0000.
```

```
        rjmp MAIN               ;Relative jump to start of program.


.org 0x0200
MAIN:

        ldi external_counter, 0x00 ; LOAD R20 with 0x00

        ; Run at 32MHz
        call Change_CLK_32HZ    ; Change Clk to 32MHz

        call BLUE_PWM                   ; Run bluw pulse

        ; Set PORT F
        ldi R16, 0xFF                   ; LOAD r16 with FF
        sts PORTF_DIRCLR, r16   ; Set to write

        ; Set Port C
        ldi R21, 0xFF                   ; LOAD FF to r21
        sts PORTC_DIRSET, R21   ; Set to write
        sts PORTC_OUT, R21              ; Turn off LEDS



        ; Set PORTF interupt control
        ldi R16, intr_crtl_lvl_config   ; LOAD config into R16
        sts PORTF_INTCTRL, R16                  ; Set interupt control lvl config

        ; Set PORT F MASK
        ldi R16, button_PF2                     ; LOAD 0b00000100 into R16
        sts PORTF_INT0MASK, R16         ; Set intrerupt mask

        ;
        ldi R16, PORT_ISC_FALLING_gc    ; LOAD PFP2 config
        sts PORTF_PIN2CTRL, R16                 ; Set PFP2 config

        ldi R16, PMIC_crtl_lvl_config   ; LOAD PMIC lvl config
        sts PMIC_CTRL, R16                              ; Set PMIC lvl config

        ;
        sei


; infinite loop
Never_End:
        ldi R16, 0b01000000
```

```
        sts PORTD_OUTTGL, R16
        rjmp Never_End          ; Jump to restart output loop


;*********************SUBROUTINES************************************
; Subroutine Name:  SET_COUNTER
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None

SET_COUNTER:
        push R16        ; PUSH r16 to stack
        push R17        ; PUSH r17 to stack

        ldi r17, delay_cycles_top
        ldi r16, delay_cycles

        sts TCC0_PER, R16               ; Set Lower Bits of TOP
        sts (TCC0_PER + 1), R17 ; Set Higher Bits of TOP

        pop R17
        pop R16
        ret

;*********************SUBROUTINES************************************
; Subroutine Name:  SET_COUNTER_INT
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
SET_COUNTER_INT:
        push R16        ; PUSH r16 to stack
        push R18        ; PUSH r18 to stack

        ldi R16, INTCTRLA_config        ; LOAD CTRLA config into R16
        sts TCC0_INTCTRLA, R16          ; Set COUNTER for CTRLA

        ldi R18, clk_div        ; LOAD Clk prescaler into R18
        sts TCC0_CTRLA, R18             ; Set Prescalar for Counter

        pop R18 ; Pop r18 from stack
        pop R16 ; POP r16 from stack
        ret


;*********************ISR*****************************************
; ISR Name: overflow_logic
```

```
; Inputs: No direct input (from stack)
; Outputs: R20
; Affected: R21
overflow_logic:
        PUSH R16        ; PUSH r16 to stack
        PUSH R17        ; PUSH r17 to stack
        PUSH R19        ; PUSH r19 to stack
        PUSH R21        ; PUSH r21 to stack
;IF
        lds r19, PORTF_IN               ; LOAD Port F into r19
        ldi r17, button_PF2             ; Set bitmask 0000 0100 in r17
        and r19, r17                    ; Isolate bit 2 in r19
        cp R19, R17
        breq ENDIF
;IF_BODY
        inc external_counter            ; counter = counter + 1
        mov R21, external_counter       ; LOAD FF to r21
        com R21                                 ; Compliment R21
        sts PORTC_OUT, R21                      ; Display LED config

ENDIF:

        ;Disable Timer interrupt
        ldi R16, 0x00                   ; LOAD 0 into R16
        sts TCC0_INTCTRLA, R16  ; Store 0 into INT CTRL A

        ;Disable Timer
        sts TCC0_CTRLA, R16     ; Store 0 into INT CTRL A

        ;Enable external interrupt
        ldi R16, intr_crtl_lvl_config   ; LOAD config into R16
        sts PORTF_INTCTRL, R16                  ; Set interupt control lvl config

        ;Clear Interupts flags
        ldi R21,  0b00000001            ; LOAD 0x01 into R21
        sts PORTF_INTFLAGS, R21         ; Clear Interupts flags

        POP R21 ; POP r21 from stack
        POP R19 ; POP r19 from stack
        POP     R17     ; POP r17 from stack
        POP R16 ; POP r16 from stack
        reti    ; Return from interrupt


;********************ISR*************************************
; ISR Name: ext_int_logic
```

```
; Inputs: No direct input (from stack)
; Outputs: R20
; Affected: R21
ext_int_logic:

        push R16
        push R21

        ; Disable external interrupt
        ldi R16, 0x00
        sts PORTF_INTCTRL, R16

        ; Initialize Timer and Timer interrupt;
        call SET_COUNTER          ; Set the counter
        call SET_COUNTER_INT ; Set coinfig for overflow interrupt


        ldi R21,  0b00000001              ; LOAD 0x01 into R21
        sts PORTF_INTFLAGS, R21           ; Clear Interupts flags

        pop R21 ; POP r21 from stack
        pop R16 ; POP r16 from stack
        reti
```

```
;-----------------------------------------------------------
;-----------------------------------------------------------
; Reused from previous parts of lab
;-----------------------------------------------------------
;-----------------------------------------------------------

;*********************SUBROUTINES************************************
; Subroutine Name: Change_CLK_32HZ
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
Change_CLK_32HZ:
        ;Push Values
        push R17 ; PUSH r17 to stack
        push R18 ; PUSH r18 to stack
        push R19 ; PUSH r19 to stack
        push R20 ; PUSH r20 to stack

; Enable the new oscillator
ldi R16, new_clock_freq         ; Load R16 with the clk-freq config 0x02
sts OSC_CTRL, r16                       ; Set the clk config

;Wait for the right flag to be set in the OSC_STATUS reg
; While flag is not set
While_32_flag:
        lds R17, OSC_STATUS             ; Load Status Flag
        and R17, R16                    ; Bit-mask with 00000010
        cp R17, R16                     ; Compare Mask and Value
        brne While_32_flag              ; Restart loop if flag not set

; Write the IOREG signature to the CPU_CCP reg
        ldi R17, CCP_IOREG_gc    ; Load IOREG into R17
        sts CPU_CCP, R17                ; Store IOREG into CPU CCP

;Select the new clock source in the CLK_CTRL reg
        ldi R17, CLK_SCLKSEL_RC32M_gc; load 32 MHz internal osc config
        sts CLK_CTRL, R17               ; Store config in clk control

        ;Pop Values
        pop R20 ; POP r20 from stack
        pop R19 ; POP r19 from stack
```

```
        pop R18 ; POP r18 from stack
        pop R17 ; POP r17 from stack
        ret



;********************SUBROUTINES************************************
; Subroutine Name:  BLUE_PWM
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
BLUE_PWM:
        ;Push Values
        push R16 ; PUSH r17 to stack
        push R17 ; PUSH r17 to stack

        ; Remap ports
        ldi R16, port_map_config        ; Load port map config into R16
        sts PORTD_REMAP, R16            ; LOAD port map config

        ; Set PORT D/LED to output
        ldi R16, BIT456                 ;load a four bit value (PORTD is only four
        sts PORTD_DIRSET, R16   ;set all the GPIO's in the four bit
                                                        ; PORTD as outputs

        ; Invert Port D
        ldi R16, PORTD_PIN_CTRL ; LOAD PIN CTRL config
        sts PORTD_PIN0CTRL, R16 ; Invert Pin 0 of Port D
        sts PORTD_PIN1CTRL, R16 ; Invert Pin 1 of Port D
        sts PORTD_PIN2CTRL, R16 ; Invert Pin 2 of Port D
        sts PORTD_PIN3CTRL, R16 ; Invert Pin 3 of Port D
        sts PORTD_PIN4CTRL, R16 ; Invert Pin 4 of Port D
        sts PORTD_PIN5CTRL, R16 ; Invert Pin 5 of Port D
        sts PORTD_PIN6CTRL, R16 ; Invert Pin 6 of Port D
        sts PORTD_PIN7CTRL, R16 ; Invert Pin 7 of Port D

        ; Set TOP of PWM
        ldi R16, MAX_PERIOD
        ldi R17, 0x00                   ; LOAD R17 with 0x00
        sts TCD0_PER, R16               ; Set Lower Bits of TOP
        sts (TCD0_PER + 1), R17 ; Set Higher Bits of TOP

        ; Set up Control D
        sts TCD0_CTRLD, R17             ; LOAD CTRLD with 0x00

        ; Set up Control B
```

```
        ldi R17, ctrlb_config    ; LOAD 0b01000011 into R17
        sts TCD0_CTRLB, R17              ; Store contrl b config

        ; Set up Compare Chanel
        ldi R16, BLUE_PERIOD     ; LOAD r16 with blue time
        ldi R17, 0x00                    ; LOAD R17 with 0x00
        sts TCD0_CCC, R16                ; LOAD compare chanel (lower)
        sts (TCD0_CCC + 1), R17 ; LOAD compare chanel (higher)


        ;control a
        ldi R16, clk_div                 ; LOAD 0b00000111
        sts TCD0_CTRLA, R16              ; store clk prescaler in ctrl A


        ;Pop Values
        pop R17 ; POP r17 from stack
        pop R16 ; POP r16 from stack
        ret             ; Return
```

Code for lab3d.asm:

```
; Lab 3 Part D
; Name:          Michael Arboleda
; Section:       7F34
; TA Name:       Wesley Piard
; Description: Use interupts to display patterns
;
; lab3d.asm
; Created: 6/14/2017 1:21:50 AM

.include "ATxmega128A1Udef.inc"

; address equates

; Constant equates
.equ BIT456 = 0x70
.equ new_clock_freq = 0b00000010
.equ PORTD_PIN_CTRL = 0b01000000
.equ MAX_PERIOD = 0xFF
.equ clk_div = 0b00000111
;-------------------------------------------------------;
.equ intr_crtl_lvl_config = 0b00000011   ; 0x03
.equ button_PF2 = 0b00000100                          ; 0x04
.equ PORTF_PIN2_CONFIG = 0b00010000                   ; 0x10
.equ PMIC_crtl_lvl_config = 0b00000111   ;
;-------------------------------------------------------;
.equ clk_div_timer = 0b00000111
.equ delay_cycles = 0xFF
.equ delay_cycles_top = 0x00
.equ INTCTRLA_config = 0b00000011
;-------------------------------------------------------;
.equ delay_cycles_05 = 0x00
.equ delay_cycles_top_05 = 0xFF
.equ mod4 = 0b00000011
.equ port_map_config = 0b00000111
.equ ctrlb_config        = 0b01110011


.equ UFO_RED = 0xFA
.equ UFO_GREEN = 0x46
.equ UFO_BLUE = 0x16
.equ UFB_RED = 0x00
.equ UFB_GREEN = 0x21
.equ UFB_BLUE = 0xA5
```

```
.equ HOLIDAYR_RED = 0xC2
.equ HOLIDAYR_GREEN = 0x1F
.equ HOLIDAYR_BLUE = 0x1F
.equ HOLIDAYG_RED = 0x3C
.equ HOLIDAYG_GREEN = 0x8D
.equ HOLIDAYG_BLUE = 0x0D

.equ HULKP_RED = 0x8A
.equ HULKP_GREEN = 0x2C
.equ HULKP_BLUE = 0x9A
.equ HULKG_RED = 0x49
.equ HULKG_GREEN = 0xFF
.equ HULKG_BLUE = 0x07


; Reg Defs
.def external_counter = R4
.def switcher_counter = R5
.def RED1_PERIOD = R23
.def GREEN1_PERIOD = R24
.def BLUE1_PERIOD = R25
.def RED2_PERIOD = R20
.def GREEN2_PERIOD = R21
.def BLUE2_PERIOD = R22

;ORG defs
.ORG PORTF_INT0_vect
        rjmp ext_int_logic

.ORG TCC0_OVF_vect
        rjmp overflow_logic

.ORG TCE0_OVF_vect
        rjmp overflow_05_logic

.ORG 0x0000                     ;Code starts running from address 0x0000.
        rjmp MAIN               ;Relative jump to start of program.


.org 0x0200
MAIN:

        ; Set button counter
        ldi R16, 0x00                       ; LOAD R16 with 0x00
        mov external_counter, R16 ; LOAD R4 with 0x00
        mov switcher_counter, R16
```

```
        ; Run at 32MHz
        call Change_CLK_32HZ     ; Change Clk to 32MHz

        ; Set PORT F
        ldi R16, 0xFF                   ; LOAD r16 with FF
        sts PORTF_DIRCLR, r16    ; Set to write

        ; Set Port C
        ldi R21, 0xFF                   ; LOAD FF to r21
        sts PORTC_DIRSET, R21    ; Set to write
        sts PORTC_OUT, R21               ; Turn off LEDS

        ; SET PWM
        call PWM_setup

        ; Set PORTF interupt control
        ldi R16, intr_crtl_lvl_config    ; LOAD config into R16
        sts PORTF_INTCTRL, R16                  ; Set interupt control lvl config

        ; Set PORT F MASK
        ldi R16, button_PF2                     ; LOAD 0b00000100 into R16
        sts PORTF_INT0MASK, R16          ; Set intrerupt mask

        ;
        ldi R16, PORT_ISC_FALLING_gc     ; LOAD PFP2 config
        sts PORTF_PIN2CTRL, R16                 ; Set PFP2 config

        ldi R16, PMIC_crtl_lvl_config    ; LOAD PMIC lvl config
        sts PMIC_CTRL, R16                              ; Set PMIC lvl config

        sei


; infinite loop
Never_End:

        rjmp Never_End           ; Jump to restart output loop




;*********************SUBROUTINES*************************************
; Subroutine Name:  SET_COUNTER
```

```
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None

SET_COUNTER:
        push R16        ; PUSH r16 to stack
        push R17        ; PUSH r17 to stack

        ldi r17, delay_cycles_top       ; LOAD higher bits of top
        ldi r16, delay_cycles           ; LOAD lower bits of bottom

        sts TCC0_PER, R16               ; Set Lower Bits of TOP
        sts (TCC0_PER + 1), R17 ; Set Higher Bits of TOP

        pop R17 ; POP R17 from stack
        pop R16 ; POP R16 from stack
        ret             ; Return from subroutine



;*********************SUBROUTINES*************************************
; Subroutine Name:  SET_COUNTER_INT
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
SET_COUNTER_INT:
        push R16        ; PUSH r16 to stack
        push R18        ; PUSH r18 to stack

        ldi R16, INTCTRLA_config        ; LOAD CTRLA config into R16
        sts TCC0_INTCTRLA, R16          ; Set COUNTER for CTRLA

        ldi R18, clk_div        ; LOAD Clk prescaler into R18
        sts TCC0_CTRLA, R18             ; Set Prescalar for Counter

        pop R18 ; Pop r18 from stack
        pop R16 ; POP r16 from stack
        ret



;*********************SUBROUTINES*************************************
; Subroutine Name:  STOP_COUNTER
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
STOP_COUNTER:
```

```
        push R16

        ;Disable Timer interrupt
        ldi R16, 0x00                   ; LOAD 0 into R16
        sts TCC0_INTCTRLA, R16  ; Store 0 into INT CTRL A

        ;Disable Timer
        sts TCC0_CTRLA, R16     ; Store 0 into INT CTRL A

        ;Reset Timer
        ldi R16, TC_CMD_RESTART_gc
        sts TCC0_CTRLFCLR, R16

        pop R16
        ret


;*********************SUBROUTINES************************************
; Subroutine Name:  set_05_counter
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
SET_05_COUNTER:
        push R16        ; PUSH r16 to stack
        push R17        ; PUSH r17 to stack

        ldi r17, 0x3d;delay_cycles_top_05
        ldi r16, 0x09;delay_cycles_05

        sts TCE0_PER, R16               ; Set Lower Bits of TOP
        sts (TCE0_PER + 1), R17 ; Set Higher Bits of TOP

        pop R17
        pop R16
        ret



;*********************SUBROUTINES************************************
; Subroutine Name:  SET_05_COUNTER_INT
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
SET_05_COUNTER_INT:
        push R16        ; PUSH r16 to stack
        push R18        ; PUSH r18 to stack
```

```
        ldi R16, INTCTRLA_config        ; LOAD CTRLA config into R16
        sts TCE0_INTCTRLA, R16          ; Set COUNTER for CTRLA

        ldi R18, clk_div        ; LOAD Clk prescaler into R18
        sts TCE0_CTRLA, R18             ; Set Prescalar for Counter

        pop R18 ; Pop r18 from stack
        pop R16 ; POP r16 from stack
        ret



;*********************SUBROUTINES************************************
; Subroutine Name:  STOP_05_COUNTER
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
STOP_05_COUNTER:
        push R16

        ;Disable Timer interrupt
        ldi R16, 0x00                   ; LOAD 0 into R16
        sts TCE0_INTCTRLA, R16  ; Store 0 into INT CTRL A

        ;Disable Timer
        sts TCE0_CTRLA, R16     ; Store 0 into INT CTRL A

        ;3d09

        ;Reset Timer
        ldi R16, TC_CMD_RESTART_gc
        sts TCE0_CTRLFCLR, R16

        pop R16
        ret


;*********************SUBROUTINES************************************
; Subroutine Name:  Set_Pattern
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
Set_Pattern:
        push R18
```

```
        ldi r18, mod4           ; LOAD R18 with bitmask 0b00000011
        and r18, external_counter       ; And counter with bit mask
; IF counter mod 4 = 0
        cpi R18, 0x00           ; Check if 0
        brne ELSEIF1            ; brench if not equal
        ; MAKE no LED is on
        ldi RED1_PERIOD, 0x00           ; LOAD RED for 1st part
        ldi GREEN1_PERIOD,  0x00        ; LOAD GREEN for 1st part
        ldi BLUE1_PERIOD, 0x00          ; LOAD BLUE for 1st part
        ldi RED2_PERIOD, 0x00           ; LOAD RED for 2nd part
        ldi GREEN2_PERIOD, 0x00         ; LOAD GREEN for 2nd part
        ldi BLUE2_PERIOD, 0x00          ; LOAD BLUE for 2nd part
        jmp ENDELSE                                     ; JMP to end of if-elses
; ELSE IF counter mod 4 = 1
ELSEIF1:
        cpi R18, 0x01           ; Check if 0
        brne ELSEIF2            ; brench if not equal
        ; MAKE no LED is on
        ldi RED1_PERIOD, UFO_RED                ; LOAD RED for 1st part
        ldi GREEN1_PERIOD, UFO_GREEN    ; LOAD GREEN for 1st part
        ldi BLUE1_PERIOD, UFO_BLUE              ; LOAD BLUE for 1st part
        ldi RED2_PERIOD, UFB_RED                ; LOAD RED for 2nd part
        ldi GREEN2_PERIOD, UFB_GREEN            ; LOAD GREEN for 2nd part
        ldi BLUE2_PERIOD, UFB_BLUE      ; LOAD BLUE for 2nd part
        jmp ENDELSE                                     ; JMP to end of if-elses
; ELSE IF counter mod 4 = 2
ELSEIF2:
        cpi R18, 0x02           ; Check if 0
        brne ELSEIF3            ; brench if not equal
        ; MAKE no LED is on
        ldi RED1_PERIOD, HOLIDAYR_RED           ; LOAD RED for 1st part
        ldi GREEN1_PERIOD,  HOLIDAYR_GREEN      ; LOAD GREEN for 1st part
        ldi BLUE1_PERIOD, HOLIDAYR_BLUE         ; LOAD BLUE for 1st part
        ldi RED2_PERIOD, HOLIDAYG_RED           ; LOAD RED for 2nd part
        ldi GREEN2_PERIOD, HOLIDAYG_GREEN               ; LOAD GREEN for 2nd part
        ldi BLUE2_PERIOD, HOLIDAYG_BLUE ; LOAD BLUE for 2nd part
        jmp ENDELSE                                     ; JMP to end of if-elses

; ELSE IF counter mod 4 = 3
ELSEIF3:
        cpi R18, 0x03           ; Check if 0
        brne ENDELSE            ; brench if not equal
        ; MAKE no LED is on
        ldi RED1_PERIOD, HULKP_RED              ; LOAD RED for 1st part
        ldi GREEN1_PERIOD,  HULKP_GREEN ; LOAD GREEN for 1st part
        ldi BLUE1_PERIOD, HULKP_BLUE    ; LOAD BLUE for 1st part
```

```
        ldi RED2_PERIOD, HULKG_RED              ; LOAD RED for 2nd part
        ldi GREEN2_PERIOD, HULKG_GREEN  ; LOAD GREEN for 2nd part
        ldi BLUE2_PERIOD, HULKG_BLUE    ; LOAD BLUE for 2nd part

ENDELSE:

        pop R18
        ret



;********************ISR************************************
; ISR Name: overflow_logic
; Inputs: No direct input (from stack)
; Outputs: R20
; Affected: R21
overflow_logic:
        PUSH R16        ; PUSH r16 to stack
        PUSH R17        ; PUSH r17 to stack
        PUSH R19        ; PUSH r19 to stack
        PUSH R21        ; PUSH r21 to stack
;IF
        lds r19, PORTF_IN               ; LOAD Port F into r19
        ldi r17, button_PF2             ; Set bitmask 0000 0100 in r17
        and r19, r17                    ; Isolate bit 2 in r19
        cp R19, R17                             ; Compare R19, R17
        breq ENDIF                              ; branch if button not pressed
;IF_BODY
        inc external_counter            ; counter = counter + 1
        mov R21, external_counter       ; LOAD FF to r21
        com R21                                         ; Compliment R21
        sts PORTC_OUT, R21                      ; Display LED config

ENDIF:

        call STOP_COUNTER

        ;Reset switcher counter
        ldi R16, 0x00                   ; LOAD 0 into R16
        mov switcher_counter, R16

        ; Enable external interrupt
        ldi R16, intr_crtl_lvl_config   ; LOAD config into R16
        sts PORTF_INTCTRL, R16                  ; Set interupt control lvl config

        ; Initialize Timer and Timer interrupt for .05sec;
```

```
        call SET_05_COUNTER                ; set  counter
        call SET_05_COUNTER_INT ; set counter and start
        call SET_COMPARE_CH                ; set compare channels

        ;Clear Interupts flags
        ldi R21,  0b00000001               ; LOAD 0x01 into R21
        sts PORTF_INTFLAGS, R21            ; Clear Interupts flags

        POP R21 ; POP r21 from stack
        POP R19 ; POP r19 from stack
        POP     R17     ; POP r17 from stack
        POP R16 ; POP r16 from stack
        reti    ; Return from interrupt



;*********************ISR*************************************
; ISR Name: overflow_05_logic
; Inputs: No direct input (from stack)
; Outputs: R20
; Affected: R21
overflow_05_logic:
        push R21

        ;if overflow, change color
        call SET_COMPARE_CH

        ;Clear Interupts flags
        ldi R21,  0b00000001               ; LOAD 0x01 into R21
        sts PORTF_INTFLAGS, R21            ; Clear Interupts flags

        pop R21
        reti

;*********************ISR*************************************
; ISR Name: ext_int_logic
; Inputs: No direct input (from stack)
; Outputs: R20
; Affected: R21
ext_int_logic:
        push R16
        push R21

        ; Disable external interrupt
        ldi R16, 0x00
        sts PORTF_INTCTRL, R16
```

```
        ; STOP .5s counter
        call STOP_05_COUNTER

        ; Initialize Timer and Timer interrupt;
        call SET_COUNTER          ;
        call SET_COUNTER_INT ;
        call SET_COMPARE_CH       ;

        ldi R21,  0b00000001              ; LOAD 0x01 into R21
        sts PORTF_INTFLAGS, R21           ; Clear Interupts flags

        pop R21 ; POP r21 from stack
        pop R16 ; POP r16 from stack
        reti



;*********************SUBROUTINES************************************
; Subroutine Name:  PWM_setup
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
PWM_setup:
        ;Push Values
        push R16 ; PUSH r17 to stack
        push R17 ; PUSH r17 to stack

        ; Remap ports
        ldi R16, port_map_config          ; Load port map config into R16
        sts PORTD_REMAP, R16              ; LOAD port map config

        ; Set PORT D/LED to output
        ldi R16, BIT456                   ;load a four bit value (PORTD is only four
        sts PORTD_DIRSET, R16    ;set all the GPIO's in the four bit
                                                  ; PORTD as outputs

        ; Invert Port D
        ldi R16, PORTD_PIN_CTRL ; LOAD PIN CTRL config
        sts PORTD_PIN0CTRL, R16 ; Invert Pin 0 of Port D
        sts PORTD_PIN1CTRL, R16 ; Invert Pin 1 of Port D
        sts PORTD_PIN2CTRL, R16 ; Invert Pin 2 of Port D
        sts PORTD_PIN3CTRL, R16 ; Invert Pin 3 of Port D
        sts PORTD_PIN4CTRL, R16 ; Invert Pin 4 of Port D
        sts PORTD_PIN5CTRL, R16 ; Invert Pin 5 of Port D
        sts PORTD_PIN6CTRL, R16 ; Invert Pin 6 of Port D
```

```
        sts PORTD_PIN7CTRL, R16 ; Invert Pin 7 of Port D

        ; Set TOP of PWM
        ldi R16, MAX_PERIOD
        ldi R17, 0x00                   ; LOAD R17 with 0x00
        sts TCD0_PER, R16               ; Set Lower Bits of TOP
        sts (TCD0_PER + 1), R17 ; Set Higher Bits of TOP

        ; Set up Control D
        sts TCD0_CTRLD, R17             ; LOAD CTRLD with 0x00

        ; Set up Control B
        ldi R17, ctrlb_config    ; LOAD 0b01110011 into R17
        sts TCD0_CTRLB, R17             ; Store contrl b config

        ;Pop Values
        pop R17 ; POP r17 from stack
        pop R16 ; POP r16 from stack
        ret             ; Return


;*********************SUBROUTINES************************************
; Subroutine Name:  SET_COMPARE_CH
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
SET_COMPARE_CH:
        push R16
        push R17
        push R18
        push R19

        ; LOAD CNT with 0
        ldi R16, 0x0
        sts (TCD0_CNT + 1), r16
        ldi R16, 0x0
        sts (TCD0_CNT), r16

        ; Restart Counter
        ldi R19,TC_CMD_RESTART_gc
        sts TCD0_CTRLFSET, R19

        call SET_PATTERN
        ; if counter is even, display first color
;no switch
        ldi r19, 0
```

```
        cp switcher_counter, r19
;with switch
        ;ldi R19, 0x0b00000001                  ; LOAD R17 with 0x00
        ;and R19, switcher_counter
        ;cpi R19, 0
        ;
        brne ELSE
        mov R16, RED1_PERIOD     ; USE first red pattern
        mov R17, GREEN1_PERIOD   ; USE first Green pattern
        mov R18, BLUE1_PERIOD    ; USE first Blue pattern
        inc switcher_counter     ;
        jmp ENDIF_2
ELSE:
        mov R16, RED2_PERIOD     ; USE 2nd red pattern
        mov R17, GREEN2_PERIOD   ; USE 2nd Green pattern
        mov R18, BLUE2_PERIOD    ; USE 2nd Blue pattern
        mov switcher_counter, R19
ENDIF_2:

        ; Set up Compare Chanel C
        sts TCD0_CCC, R18               ; LOAD compare chanel (lower)
        sts (TCD0_CCC + 1), R19 ; LOAD compare chanel (higher)

        ; Set up Compare Chanel B
        sts TCD0_CCB, R17               ; LOAD compare chanel (lower)
        sts (TCD0_CCB + 1), R19 ; LOAD compare chanel (higher)

        ; Set up Compare Chanel A
        sts TCD0_CCA, R16               ; LOAD compare chanel (lower)
        sts (TCD0_CCA + 1), R19 ; LOAD compare chanel (higher)

        ;control a
        ldi R16, clk_div                ; LOAD 0b00000111
        sts TCD0_CTRLA, R16             ; store clk prescaler in ctrl A

        pop R19
        pop r18
        pop r17
        pop r16
        ret


;-----------------------------------------------------------
;-----------------------------------------------------------
; Reused from previous parts of lab
;-----------------------------------------------------------
```

```
;-----------------------------------------------------------
;*********************SUBROUTINES************************************
; Subroutine Name: Change_CLK_32HZ
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
Change_CLK_32HZ:
        ;Push Values
        push R17 ; PUSH r17 to stack
        push R18 ; PUSH r18 to stack
        push R19 ; PUSH r19 to stack
        push R20 ; PUSH r20 to stack

; Enable the new oscillator
ldi R16, new_clock_freq         ; Load R16 with the clk-freq config 0x02
sts OSC_CTRL, r16                       ; Set the clk config

;Wait for the right flag to be set in the OSC_STATUS reg
; While flag is not set
While_32_flag:
        lds R17, OSC_STATUS             ; Load Status Flag
        and R17, R16                    ; Bit-mask with 00000010
        cp R17, R16                     ; Compare Mask and Value
        brne While_32_flag              ; Restart loop if flag not set

; Write the IOREG signature to the CPU_CCP reg
        ldi R17, CCP_IOREG_gc    ; Load IOREG into R17
        sts CPU_CCP, R17                ; Store IOREG into CPU CCP

;Select the new clock source in the CLK_CTRL reg
        ldi R17, CLK_SCLKSEL_RC32M_gc; load 32 MHz internal osc config
        sts CLK_CTRL, R17               ; Store config in clk control

        ;Pop Values
        pop R20 ; POP r20 from stack
        pop R19 ; POP r19 from stack
        pop R18 ; POP r18 from stack
        pop R17 ; POP r17 from stack
        ret
```

i. Appendix

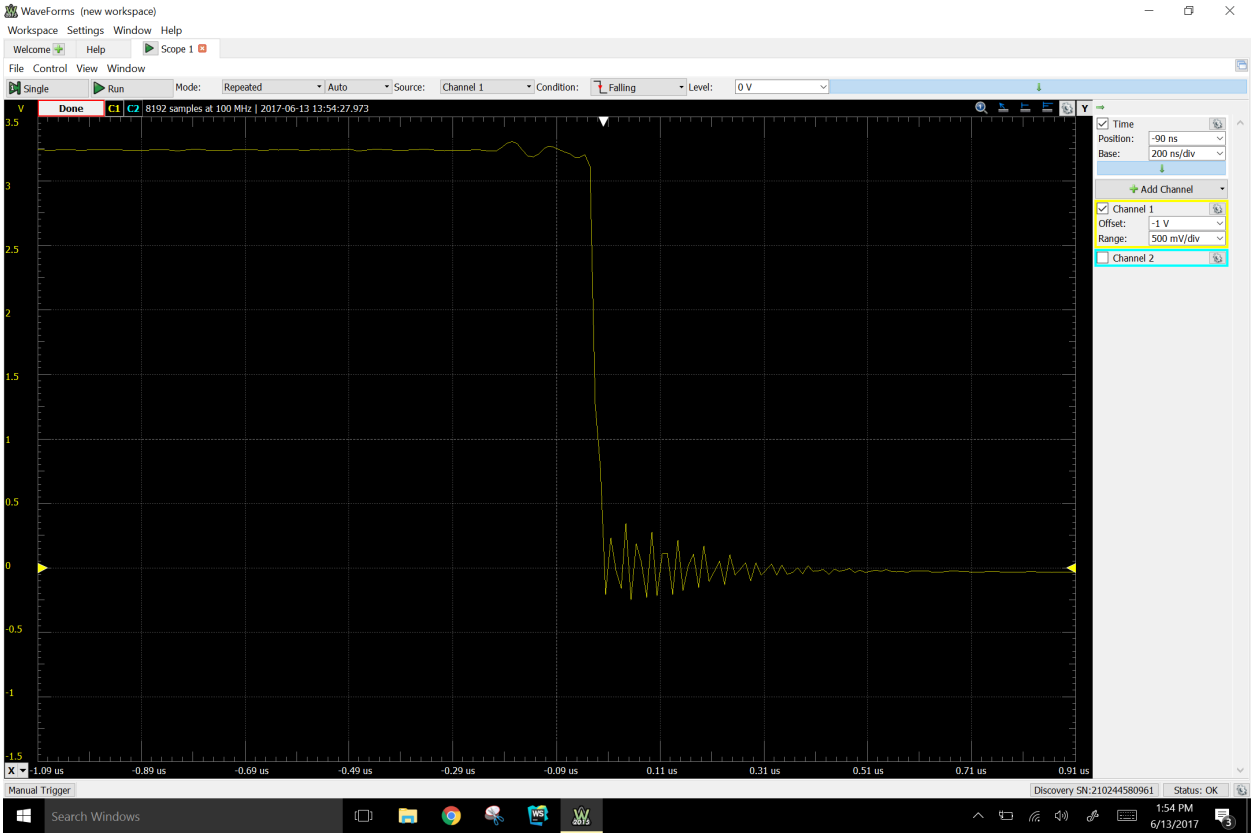

Figure 1: DEBOUNCE FALLING EDGE 1

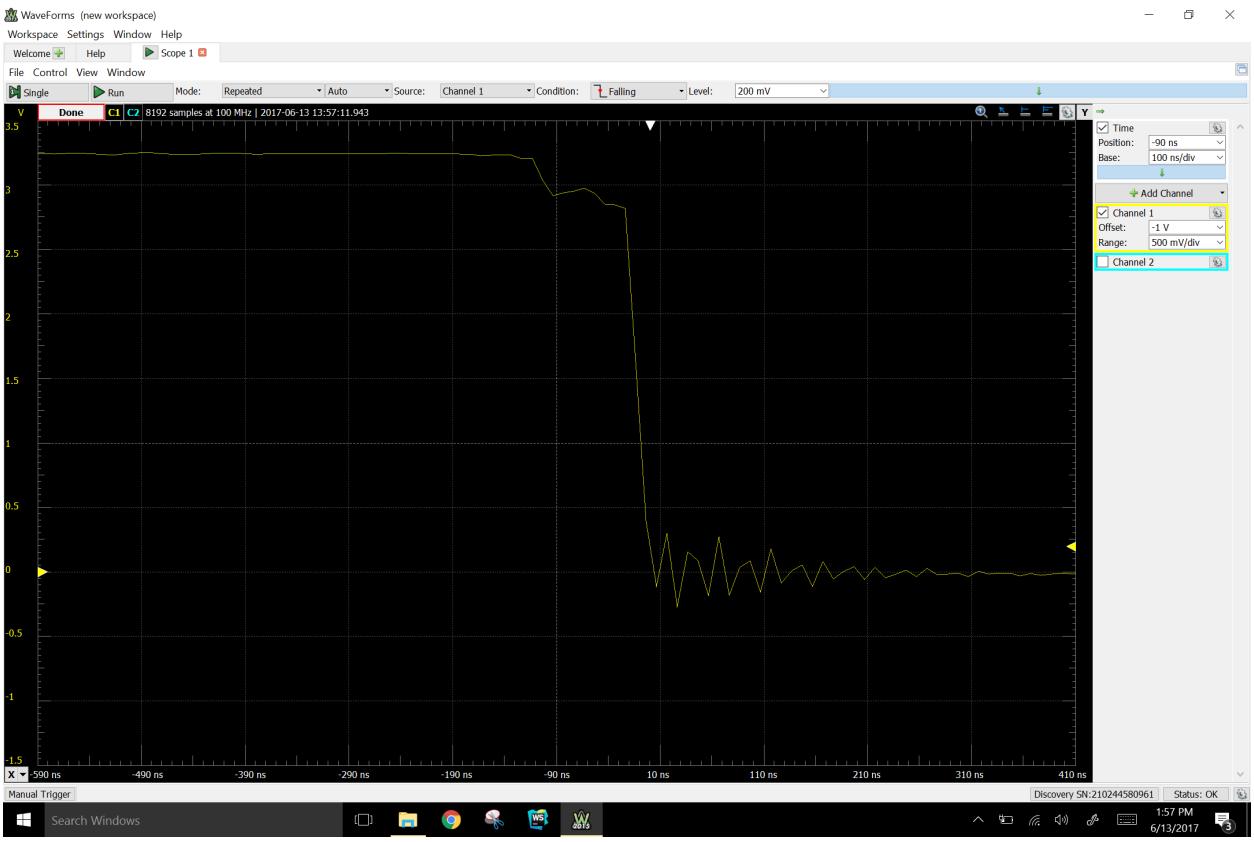

Figure 2: DEBOUNCE FALLING EDGE 2

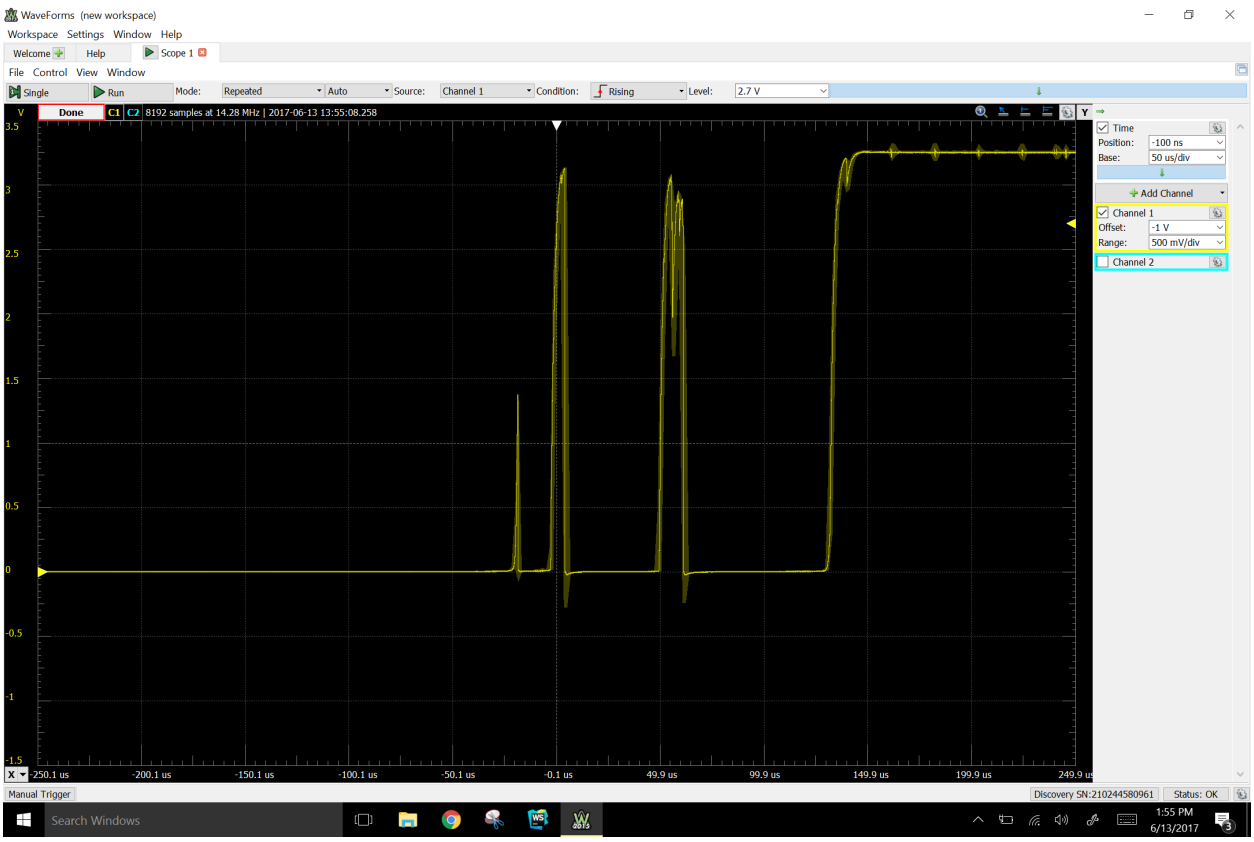

Figure 3: DEBOUNCE RISING EDGE 1

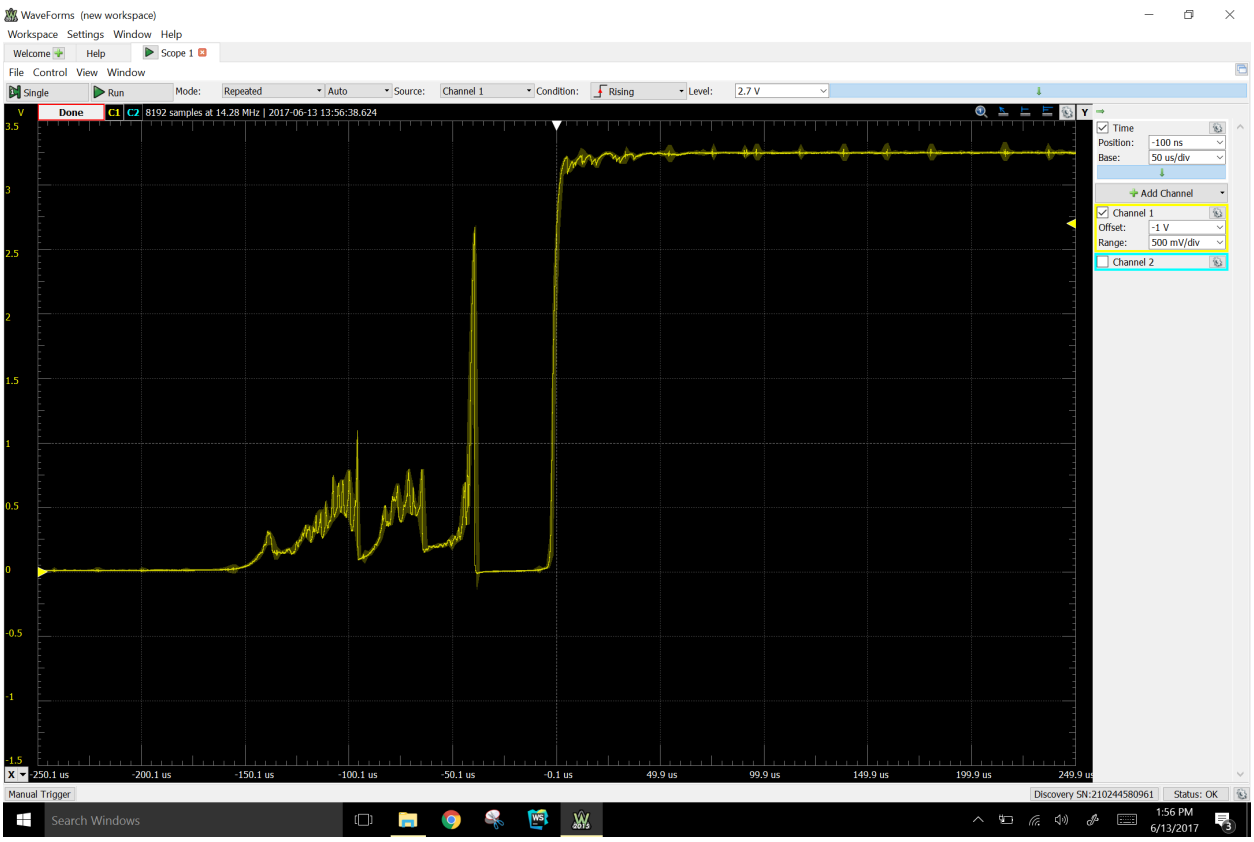

Figure 4: DEBOUNCE FALLING EDGE 1

THIS WAS ABOUT 100 u seconds. Thus the counter was at least 13