REQUIREMENTS NOT MET

N/A. All requirements are met in this lab.

PROBLEMS ENCOUNTERED

The main problems encountered were understanding nature of pulse-width modulation. The functions needed and implementation method with the timer/counter of the RGB LEDs required extensive research and review of examples to get a grasp. Though watching the supporting video establish some PWM code and reading the microprocessor manual to be better equipped with the understanding and structure nature of the ports and how the RBG LEDs works.

HOMEWORK EXERCISES

i. How many TC0 channels are necessary to control all three of the LEDs within the on-board RGB LED package?

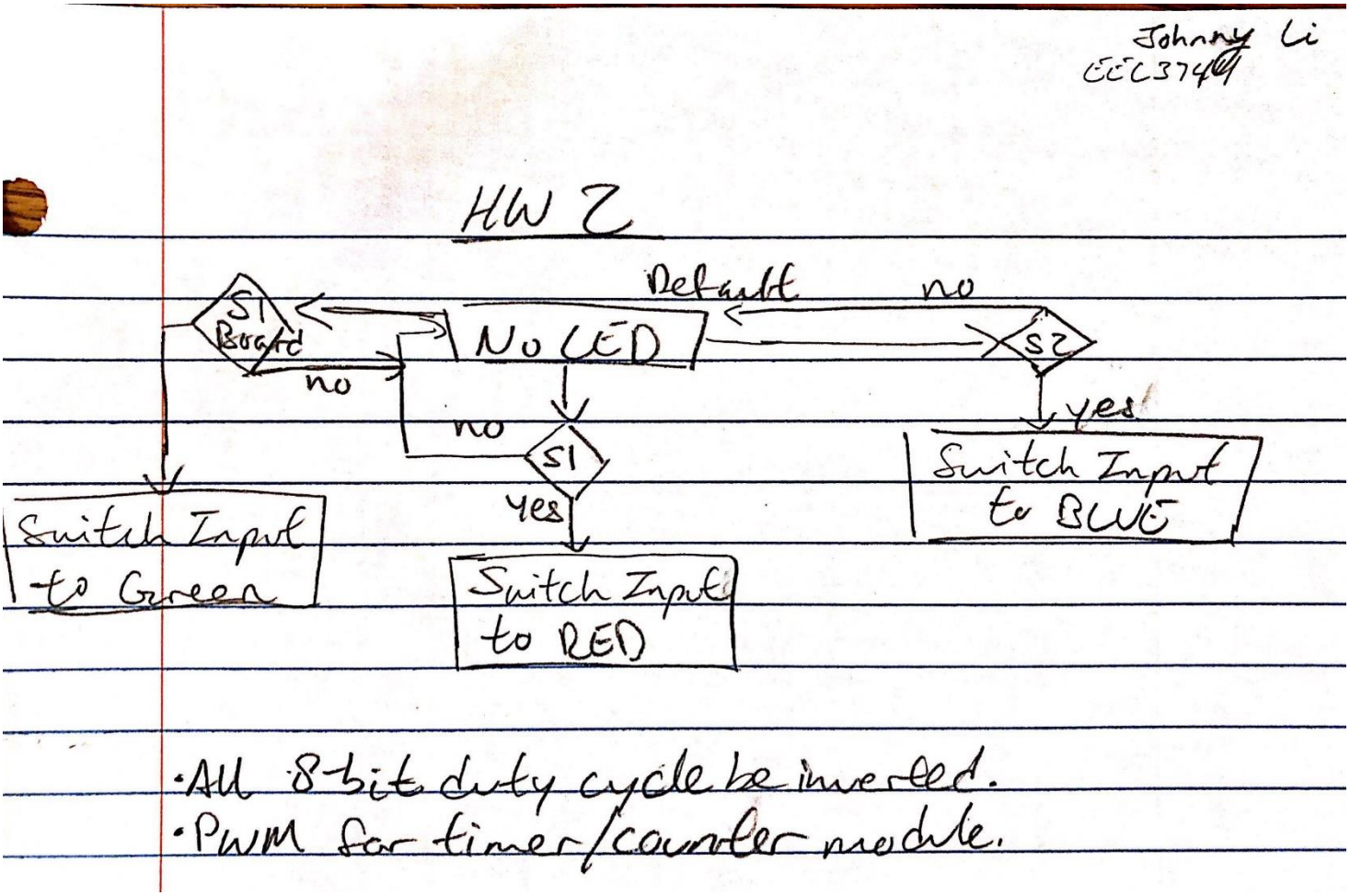
Three TC0 channels are necessary to control all three of the LEDs within the on-board RGB LED package, one for the red, one for the green, and another one for the blue LED.

ii. In the context of the program specified above, would any difference (theoretically) result from setting the RGB period to be $FFFF, instead of $FF?

Setting the RGB period to be $FFFF, instead of $FF means that the period gets bigger resulting in an increase amount of cycles to be ran though before returning to 0. This larger cycle number will lead to the LEDs to be on for a longer time then if it were when the period is $FF.

PSEUDOCODE/FLOWCHARTS

SECTION X (1, 2, etc.)



- All 8-bit duty cycle be inverted.
- PWM for timer/counter module.

PROGRAM CODE

SECTION X (1, 2, etc.)

```
;*********************************MAIN PROGRAM*********************************
	.cseg

	.org 0x0000
	rjmp MAIN

	.org 0x100
MAIN:
	;Stack initialization
	ldi YL, low(stackaddress)
	out CPU_SPL,YL
	LDI YL, high(stackaddress)
	out    CPU_SPH,YL

	; initialize relevant I/O modules (switches and LEDs)
	rcall IO_INIT

	; initialize (but do not start) the relevant timer/counter module(s)
	rcall TC_INIT

	; begin main program loop
	; "EDIT" mode
Default:
	;read S2
	lds r16, PORTF_IN
	sbrs r16, 3   ;check if 3rd bit is set thus S2 is pressed
	rjmp Timer

	;read S1
	lds r16, PORTF_IN
	sbrs r16, 2   ;check if 2nd bit is set thus S1 is not pressed
	rjmp Timer

	;read S1
	lds r16, PORTF_IN
	sbrs r16, 4   ;check if 2nd bit is set thus S1 is not pressed
	rjmp Timer

	rjmp Default  ;return to Default

;Debounce
Timer:
	ldi r18, 6
	sts TCC0_CTRLA, r18         ;increment and set prescaler 256

	lds r17, TCC0_INTFLAGS      ;load flag
	sbrs r17, 0   ;check if flag is triggered
	rjmp TimerRUN ;continue delay

TimerRUN:
	ldi r18, 0    ;disable timer
	sts TCC0_CTRLA, r18

	ldi r18,0     ;reset count
	sts TCC0_CNT, r18
	sts TCC0_CNT+1, r18
```

```
        ldi r18,1     ;reset flag
        sts TCC0_INTFLAGS, r18

        lds r16, PORTF_IN
        sbrs r16, 3   ;check if 3rd bit is set thus S2 is pressed
        rjmp BLUE_PWM ;go to BLUE

        ;read S1
        lds r16, PORTF_IN
        sbrs r16, 2   ;check if 2nd bit is set thus S1 is not pressed
        rjmp RED_PWM

        ;read S1
        lds r16, PORTQ_IN
        sbrs r16, 2   ;check if 2nd bit is set thus S1 is not pressed
        rjmp GREEN_PWM

        rjmp Default  ;return to Default

RED_PWM:
        rcall RED
        lds r16, PORTF_IN
        sbrs r16, 2   ;check if 3rd bit is set thus S2 is pressed
        rjmp Default  ;go to BLUE

        rjmp RED_PWM

BLUE_PWM:
        rcall BLUE
        lds r16, PORTF_IN
        sbrs r16, 3   ;check if 3rd bit is set thus S2 is pressed
        rjmp Default  ;go to BLUE

        rjmp BLUE_PWM

GREEN_PWM:
        rcall GREEN
        lds r16, PORTD_IN
        sbrs r16, 2   ;check if 2nd bit is set thus S1 is not pressed
        rjmp Default  ;go to BLUE

        rjmp GREEN_PWM

        ; end of program (never reached)
DONE:
        rjmp DONE
;*****************************END OF MAIN PROGRAM ****************************
;*******************************SUBROUTINES**********************************
; Name: IO_INIT
; Purpose: To initialize the relevant input/output modules, as pertains to the
;                application.
; Input(s): N/A
; Output: N/A
;****************************************************************************
IO_INIT:
        ; protect relevant registers
        push r16
        ; initialize the relevant I/O
        ; Switch
        ldi r16, inlow      ;load 0 to register
    sts PORTA_DIR,r16       ;set portA as input
```

```
	sts PORTQ_DIR,r16
	; LED
	ldi r16, outhigh     ;load 1 to register
	sts PORTD_OUT,r16    ;set portC initally off
	sts PORTD_DIR,r16    ;set portC as output
	; S1
	ldi r16, 0x01<<2
	sts PORTF_DIRCLR, r16
	; S2
	ldi r16, 0x01<<3
	sts PORTF_DIRCLR, r16
	; recover relevant registers
	pop r16
	; return from subroutine
	ret
;***************************************************************************
; Name: TC_INIT
; Purpose: To initialize the relevant timer/counter modules, as pertains to
;               application.
; Input(s): N/A
; Output: N/A
;***************************************************************************
TC_INIT:
	; protect relevant registers

	; initialize the relevant TC modules
	ldi ZL, 0x0F
	ldi ZH, 0x00
	sts TCD0_PER, ZL            ;load period tick
	sts TCD0_PER+1, ZH

	; recover relevant registers

	; return from subroutine
	ret
;***************************************************************************
; Subroutine Name:  BLUE
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
BLUE:
	;Save Values
	push R16
	push R17

	; Remap ports
	ldi R16, port_map_config
	sts PORTD_REMAP, R16

	; Set PORT D/LED to output
	ldi R16, 0x70
	sts PORTD_DIRSET, R16

	; Invert Port D
	ldi R16, PORTD_PIN_CTRL
	sts PORTD_PIN0CTRL, R16
	sts PORTD_PIN1CTRL, R16
	sts PORTD_PIN2CTRL, R16
	sts PORTD_PIN3CTRL, R16
	sts PORTD_PIN4CTRL, R16
	sts PORTD_PIN5CTRL, R16
	sts PORTD_PIN6CTRL, R16
```

```
       sts PORTD_PIN7CTRL, R16

       ; Set TOP of PWM
       ldi R16, 0xFF
       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

       ; Set up Control B
       ldi R17, ctrlb_config
       sts TCD0_CTRLB, R17         ;

       ; Set up Compare Chanel
       ldi R16, 0x0F ; 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)

       lds r16, PORTA_IN    ;load port A value to r16
    sts PORTD_OUT, r16      ;store switch value to LED portout

       ;Pop Values
       pop R17 ; POP r17 from stack
       pop R16 ; POP r16 from stack
       ret
       ;***************************************************************************
; Subroutine Name:  BLUE
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
RED:
       ;Push Values
       push R16
       push R17

       ; Remap ports
       ldi R16, port_map_config1
       sts PORTD_REMAP, R16

       ; Set PORT D/LED to output
       ldi R16, 0x70
       sts PORTD_DIRSET, R16

       ; Invert Port D
       ldi R16, PORTD_PIN_CTRL
       sts PORTD_PIN0CTRL, R16
       sts PORTD_PIN1CTRL, R16
       sts PORTD_PIN2CTRL, R16
       sts PORTD_PIN3CTRL, R16
       sts PORTD_PIN4CTRL, R16
       sts PORTD_PIN5CTRL, R16
       sts PORTD_PIN6CTRL, R16
       sts PORTD_PIN7CTRL, R16

       ; Set TOP of PWM
       ldi R16, 0xFF
       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

        ; Set up Control B
        ldi R17, ctrlb_config
        sts TCD0_CTRLB, R17         ;

        ; Set up Compare Chanel
        ldi R16, 0x0F ; 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)

        lds r16, PORTA_IN    ;load port A value to r16
    sts PORTD_OUT, r16       ;store switch value to LED portout

        ;Pop Values
        pop R17 ; POP r17 from stack
        pop R16 ; POP r16 from stack
        ret
;***************************************************************************
; Subroutine Name:  GREEN
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
; Affected: None
GREEN:
        ;Save Values
        push R16
        push R17

        ; Remap ports
        ldi R16, port_map_config2
        sts PORTD_REMAP, R16

        ; Set PORT D/LED to output
        ldi R16, 0x70
        sts PORTD_DIRSET, R16

        ; Invert Port D
        ldi R16, PORTD_PIN_CTRL
        sts PORTD_PIN0CTRL, R16
        sts PORTD_PIN1CTRL, R16
        sts PORTD_PIN2CTRL, R16
        sts PORTD_PIN3CTRL, R16
        sts PORTD_PIN4CTRL, R16
        sts PORTD_PIN5CTRL, R16
        sts PORTD_PIN6CTRL, R16
        sts PORTD_PIN7CTRL, R16

        ; Set TOP of PWM
        ldi R16, 0xFF
        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

        ; Set up Control B
        ldi R17, ctrlb_config
        sts TCD0_CTRLB, R17         ;
```

```
       ; Set up Compare Chanel
       ldi R16, 0x0F ; 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)

       lds r16, PORTA_IN    ;load port A value to r16
    sts PORTD_OUT, r16      ;store switch value to LED portout

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

;*****************************END OF SUBROUTINES*****************************
;*****************************END OF "lab2_4.asm"*****************************
```

APPENDIX

```
1  ;HW2
2  ;Section #: 1823
3  ;Name: Johnny Li
4  ;Class #: 12378
5  ;PI Name: Jared Holley
6  ;Description: Pulse-width Modulation
7  ;**********************************INCLUDES*************************************
8  .include "ATxmega128a1udef.inc"
9  ;********************************END OF INCLUDES********************************
10 ;********************************DEFINED SYMBOLS********************************
11 .equ outhigh=0xff   ;set as input or high value
12 .equ inlow=0x00 ;set as output or low value
13 .equ stackaddress=0x3FFF    ;stack starting address
14 .equ port_map_config=0b00000100 ;new map
15 .equ port_map_config2=0b00000101    ;new map
16 .equ port_map_config1=0b00000110    ;new map
17 .equ ctrlb_config=0b01000011    ;ctrl config
18 .equ PORTD_PIN_CTRL=0b01000000  ;pin control
19 ;***************************END OF DEFINED SYMBOLS*****************************
20 ;********************************MEMORY CONSTANTS********************************
21     ; data memory allocation
22     .dseg
23
24 ;***************************END OF MEMORY CONSTANTS*****************************
```

Screenshot 1: HW2 Memory Configuration