REQUIREMENTS NOT MET

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

PROBLEMS ENCOUNTERED

For part 1, the main problems encountered were understanding the functions and implementation of the interrupt, the way for the initialization of the correct enables and priority level placing of interrupts. This was eventually solved by watching the intensive research in the microprocessor manuals and lectures to be better equipped with the understanding and structure nature of the interrupts.

For part 2, the main problems encountered were trying to implement the debounce when the interrupt has occurred. Often, when the tactile S1 is pressed it should trip the interrupt that goes to the debounce but instead the debounce in a ran solely by itself, the blue LED would not stay on. This issue was solved by more research into the manuals. In addition, the structure for the subroutines were listed but never given an example so it became a makeshift process of making one to fit its requirements. There was another issue being that the delay on successfully run when adding an additional counter that branch to repeat the previous counter but would logically in my perspective do nothing to aid it.

FUTURE WORK/APPLICATIONS

This lab was a good introduction into the implementation and function of interrupt. This lab is to be the expansion of more complex assembly programs, able to give the users' another way to code, enabling the running of separate functions from the main program. Like the subroutine, the way I program is now changed to be inclusive of interrupts for more capability of my programs. If given more time, the code of the lab could have been more organized and have a much neater layout to further reduce the likelihood of mistakes and further enhance the understanding of the program. With more time a more extensive denounce would have been implemented and many more different LEDs could have been placed into to either continuously run a unique pattern and still show the indecencies of the interrupt. Additionally, I could have used better instructions to make the code run more efficiently or learn to write more complex programs

PRE-LAB EXERCISES

Part 1: Introduction to Interrupts

i. Describe the purpose of an interrupt vector.

The interrupt vector is the the memory location of an interrupt handler, which allows for the determination of priority level for the interrupts and stores them in queue if more than one interrupt is waiting to be handled.

ii. Devise and describe a generalized series of steps for configuring any interrupt within the ATxmega128A1U.

The initial steps for configuring any interrupt within the ATxmega128A1U is to start with the memory configuration along with initializing the stack since the interrupt is stored in it. This include setting up the address for the interrupt vector to exist in. Afterward, is to initialize the interrupt by setting global interrupt, the interrupt level, and selecting a priority level for them. When building the interrupt, push the CPU status and other important registers and code in the interrupt function, then pop out the saved information and use reti to return back to the program code.

PSEUDOCODE/FLOWCHARTS

SECTION X (1, 2, etc.)

Part 1: Introduction to Interrupts

Johnny Li
EEL3744

Lab 3: Part 1

Toggle LED every 100ms with Timer by interrupt

Main: Initialize stack.
Setup port C as output.
-Set range -load period -set prescaler

Loop: Interrupt Delay
Toggle LED on/off
rjmp Loop

Delay: Save CPU SREG
Timer/Counter
Handle Event
reti

Part 2a: Interrupts, Continued

Lab 3 Part 2a

Main:
- Memory Configure
- Initialize Stack
  - I/O Initialize
  - Interrupt Initialize
  - Blue_PWM

IIO:
- Setup port F, C, and D
- Set BLUE LED to off.

ret

Int:
- Set port F IntOMASK = 0x04
- Set PMI_CTRL to low level interrupts.
- Set port F to level low.
- Read falling edge on switch.
- sei

ret

BLue:
- Turn on/off BLUE LED

rjmp BLUE ; endless loop

ret

Interrupt:
- Save CPU SREG
- Increment counter and complement it.

Part 2b: (Debounced) Interrupts, Continued

Lab 3 Part 2b

Main:
- Memory Configure
- Initialize Stack
  - I/O Initialize
  - Interrupt Initialize
  - Blue_PWM
  - Interrupt LED counter
  - Interrupt timer ↙ = check/debounce before continuing

I/O:
- Setup port F, C, and D
- Set BLUE LED to off.

ret

Int:
- set port F IntOMASK = 0x04
- Set PMI_CTRL to low level interrupt.
- Set port F to level low.
- Read falling edge on switch.
- sei

ret

Blue:
- Turn on/off BLUE LED

rjmp BLUE ; endless loop

ret

Interrupt: LED counter
- Save CPU SREG
- Increment counter and complement it.

- reti

Timer:
- Setup the timer period, prescaler, and
- Disable Timer Interrupt
- reti

PROGRAM CODE

SECTION X (1, 2, etc.)

Part 1: Introduction to Interrupts

```
;*****************************************************************************
;Lab 3 Part 1
;Section #: 1823
;Name: Johnny Li
;Class #: 12378
;PI Name: Jared Holley
;Description: Introduction to Interrupts
;*********************************MAIN PROGRAM*********************************
       .cseg
.org 0x0000
       rjmp MAIN

;Interrupt Vector
.org TCC0_OVF_VECT
       rjmp Interrtupt

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

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

       ldi ZL, 0x0E
       ldi ZH, 0x03
       sts TCC0_PER, ZL            ;load period tick
       sts TCC0_PER+1, ZH

       ;call our subroutine to initialize our interrupt
       rcall INIT_INTERRUPT

       ldi r16,6
       sts TCC0_CTRLA, r16

Loop:
       rjmp Loop

;*****************************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
; Affected: N/A
;*****************************************************************************
IO_INIT:
       ; protect relevant registers

       ; initialize the relevant I/O
```

```
       ; LED
       ldi r16, outhigh     ;load 1 to register
       sts PORTC_OUT,r16    ;set portC initally off
       sts PORTC_DIR,r16    ;set portC as output

       ; recover relevant registers

       ; return from subroutine
       ret
;***************************************************************************
; Name:     INIT_INTERRUPT
; Purpose:  Subroutine to initialize the interrupt
; Inputs:   None
; Outputs:  None
; Affected:
;***************************************************************************
INIT_INTERRUPT:
       ;turn on low level interrupts
       ldi r16, 0x01
       sts PMIC_CTRL, r16
       sts TCC0_INTCTRLA, r16

       sei           ;turn on the global interrupt flag
       ret
;*****************************END OF SUBROUTINES*****************************
;*********************************INTERRTUPT*********************************
; Name: Interrtupt
; Purpose:  Toggle LED
; Inputs:   None
; Outputs:  None
; Affected:
;***************************************************************************
Interrtupt:
       lds r16,PORTC_IN     ;load value
       com r16
    sts PORTC_OUT, r16      ;store toggle LED portout

       ; Clear Interupts flags
       ldi r16,  0x01
       sts PORTC_INTFLAGS, r16

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

       ldi r18,1     ;reset flag
       sts TCC0_INTFLAGS, r18

       reti          ;return from the interrupt routine
;****************************END OF INTERRTUPT******************************
;****************************END OF "lab2_4.asm"****************************
```

Part 2a: Interrupts, Continued

```
;***************************************************************************
;Lab 3 Part 2a
;Section #: 1823
;Name: Johnny Li
;Class #: 12378
;PI Name: Jared Holley
;Description: Interrupts, Continued
;*****************************MAIN PROGRAM**********************************
```

```
	.cseg
.org 0x0000
	rjmp MAIN

;Interrupt Vector
.org PORTF_INT0_VECT
	rjmp Interrtupt

.org 0x100
MAIN:
	ldi r21,0	;counter

	;Stack initialization
	ldi YL, low(stackaddress)
	out CPU_SPL,YL
	LDI YL, high(stackaddress)
	out	CPU_SPH,YL
	;initialization of I/O
	rcall IO_INIT
	;call our subroutine to initialize our interrupt
	rcall INIT_INTERRUPT
	;intialization blue led
	rcall BLUE_PWM

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
; Affected: N/A
;******************************************************************************
IO_INIT:
	; protect relevant registers

	; initialize the relevant I/O
	;switch
	; S1
	ldi r16, 0x01<<2
	sts PORTF_DIRCLR, r16
	; LED
	;8-bit LED
	ldi r16, outhigh	;load 1 to register
	sts PORTC_OUT,r16	;set portC initally off
	sts PORTC_DIR,r16	;set portC as output
	;BLUE_PWM
	ldi r16, 0xFF
	sts PORTD_OUT, r16	;set portC initally off

	ldi r16, 0x40 ;load 1 to register LED
	sts PORTD_DIRSET,r16 ;set portC as output

	; recover relevant registers

	; return from subroutine
	ret
;******************************************************************************
; Name:	INIT_INTERRUPT
; Purpose:  Subroutine to initialize the interrupt
```

```
; Inputs:   None
; Outputs:  None
; Affected:
;*************************************************************************
INIT_INTERRUPT:
       ;select s1 as interrupt source
       ldi r16, 0x04
       sts PORTF_INT0MASK, r16
       ;turn on low level interrupts
       sts PMIC_CTRL, r16

       ;select low level pin for external interrupt
       ldi    r16, 0x03
       sts PORTF_INTCTRL, r16
       ;port falling config
       ldi r16, 0x02
       sts PORTF_PIN2CTRL, r16

       sei           ;turn on the global interrupt flag
       ret
;*************************************************************************
; Name:     BLUE_PWM
; Purpose:  Toggle BLUE LED
; Inputs:   None
; Outputs:  None
; Affected:
;*************************************************************************
 BLUE_PWM:
       ;Turn BLUE off
       ldi r16, 0xFF
       sts PORTD_OUT, r16
       ;Turn BLUE on
       ldi r16, 0x00
       sts PORTD_OUT, r16
       ;Loop endless
       rjmp BLUE_PWM

       ret
;****************************END OF SUBROUTINES*****************************
;*********************************INTERRTUPT*********************************
; Name: Interrtupt
; Purpose:  Toggle LED
; Inputs:   None
; Outputs:  None
; Affected:
;*************************************************************************
Interrtupt:
       lds r20,CPU_SREG
       push r20

       inc r21       ;increment counter
       mov r17,r21
       com r17       ;active high

       sts PORTC_OUT, r17   ;store counter value to LED portout

       ; Clear Interupts flags
       ldi r16,  0x01
       sts PORTF_INTFLAGS, r16

       sts CPU_SREG, r20
       pop r20
```

```
       reti          ;return from the interrupt routine
;*****************************END OF INTERRTUPT*********************************
;*****************************END OF "lab2_4.asm"*******************************
```

Part 2b: (Debounced) Interrupts, Continued

```
;*******************************************************************************
;Lab 3 Part 2b
;Section #: 1823
;Name: Johnny Li
;Class #: 12378
;PI Name: Jared Holley
;Description: Interrupts, Continued (Debounced)
;*********************************MAIN PROGRAM**********************************
       .cseg
.org 0x0000
       rjmp MAIN

;Interrupt Vector
.org PORTF_INT0_VECT
       rjmp Delay    ;Debounce
.org TCC0_OVF_vect
       rjmp Interrtupt      ;LED

.org 0x100
MAIN:
       ldi r21,0     ;counter

       ;Stack initialization
       ldi YL, low(stackaddress)
       out CPU_SPL,YL
       LDI YL, high(stackaddress)
       out    CPU_SPH,YL
       ;initialization of I/O
       rcall IO_INIT

       ldi ZL, low(20000)
       ldi ZH, high(20000)
       sts TCC0_PER, ZL            ;load period tick
       sts TCC0_PER+1, ZH

       ;call our subroutine to initialize our interrupt
       rcall INIT_INTERRUPT
       ;intialization blue led
       rcall BLUE_PWM

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
; Affected: N/A
;*******************************************************************************
IO_INIT:
       ; protect relevant registers

       ; initialize the relevant I/O
       ;switch
```

```
	; S1
	ldi r16, 0x01<<2
	sts PORTF_DIRCLR, r16
	; LED
	;8-bit LED
	ldi r16, outhigh     ;load 1 to register
	sts PORTC_OUT,r16    ;set portC initally off
	sts PORTC_DIR,r16    ;set portC as output
	;BLUE_PWM
	ldi r16, 0x04
	sts PORTD_DIRCLR, r16       ;set portC initally off

	ldi r16, 0x40 ;load 1 to register LED
	sts PORTD_DIRSET,r16 ;set portC as output

	; recover relevant registers

	; return from subroutine
	ret
;***************************************************************************
; Name:     INIT_INTERRUPT
; Purpose:  Subroutine to initialize the interrupt
; Inputs:   None
; Outputs:  None
; Affected:
;***************************************************************************
INIT_INTERRUPT:
	;select s1 as interrupt source
	ldi r16, 0x04
	sts PORTF_INT0MASK, r16

	;select low level pin for external interrupt
	ldi    r16, 0x01
	sts PORTF_INTCTRL, r16
	;port falling config
	ldi r16, 0x02
	sts PORTF_PIN2CTRL, r16

	;Timer setup
	ldi r16, 1
	sts TCC0_INTCTRLA, r16

	;turn on low level interrupts
	ldi r16,1
	sts PMIC_CTRL, r16

	sei           ;turn on the global interrupt flag
	ret
;***************************************************************************
; Name:     BLUE_PWM
; Purpose:  Toggle BLUE LED
; Inputs:   None
; Outputs:  None
; Affected:
;***************************************************************************
 BLUE_PWM:
	;Turn BLUE off
	ldi r16, 0xFF
	sts PORTD_OUT, r16
	;Turn BLUE on
	ldi r16, 0x00
	sts PORTD_OUT, r16
```

```
	;Loop endless
	rjmp BLUE_PWM

	ret
;*****************************END OF SUBROUTINES*******************************
;*********************************INTERRTUPT***********************************
; Name: Interrtupt
; Purpose:  Toggle LED
; Inputs:   None
; Outputs:  None
; Affected:
;******************************************************************************
Interrtupt:
	ldi r16, 1    ;reset flag
	sts TCC0_INTFLAGS, r16

	ldi r18,0     ;reset count
	sts TCC0_CNT, r18
	sts TCC0_CNT+1, r18
	;disable timer
	sts TCC0_CTRLA, r18

	lds r16, PORTF_IN
	sbrc r16, 2   ;check if 2nd bit is clear set thus S1 is not pressed
	rjmp Wait     ;return to edit

	inc r21       ;increment counter
	mov r17,r21
	com r17       ;active high

    sts PORTC_OUT, r17      ;store counter value to LED portout

	; Clear Interupts flags
	ldi r16,  0x01
	sts PORTF_INTFLAGS, r16

	rjmp Wait             ;return from the interrupt routine
;******************************************************************************
; Name: Delay
; Purpose:  Delay function
; Inputs:   None
; Outputs:  None
; Affected:
;******************************************************************************
Delay:
	ldi r16,2 ;prescaler
	sts TCC0_CTRLA,r16

	ldi r16, 4    ;trigger flag
	sts PORTF_INTFLAGS, r16

	ldi r16, 0    ;Disable port F
	sts PORTF_INTCTRL, r16
	reti          ;return from the interrupt routine
;******************************************************************************
; Name: Wait
; Purpose:  Delay function
; Inputs:   None
; Outputs:  None
; Affected:
;******************************************************************************
Wait:
```

```
	ldi r16, 1	;reset
	sts PORTF_INTCTRL, r16
	reti		;return from the interrupt routine
;*****************************END OF INTERRTUPT*******************************
;*****************************END OF "lab2_4.asm"*****************************
```

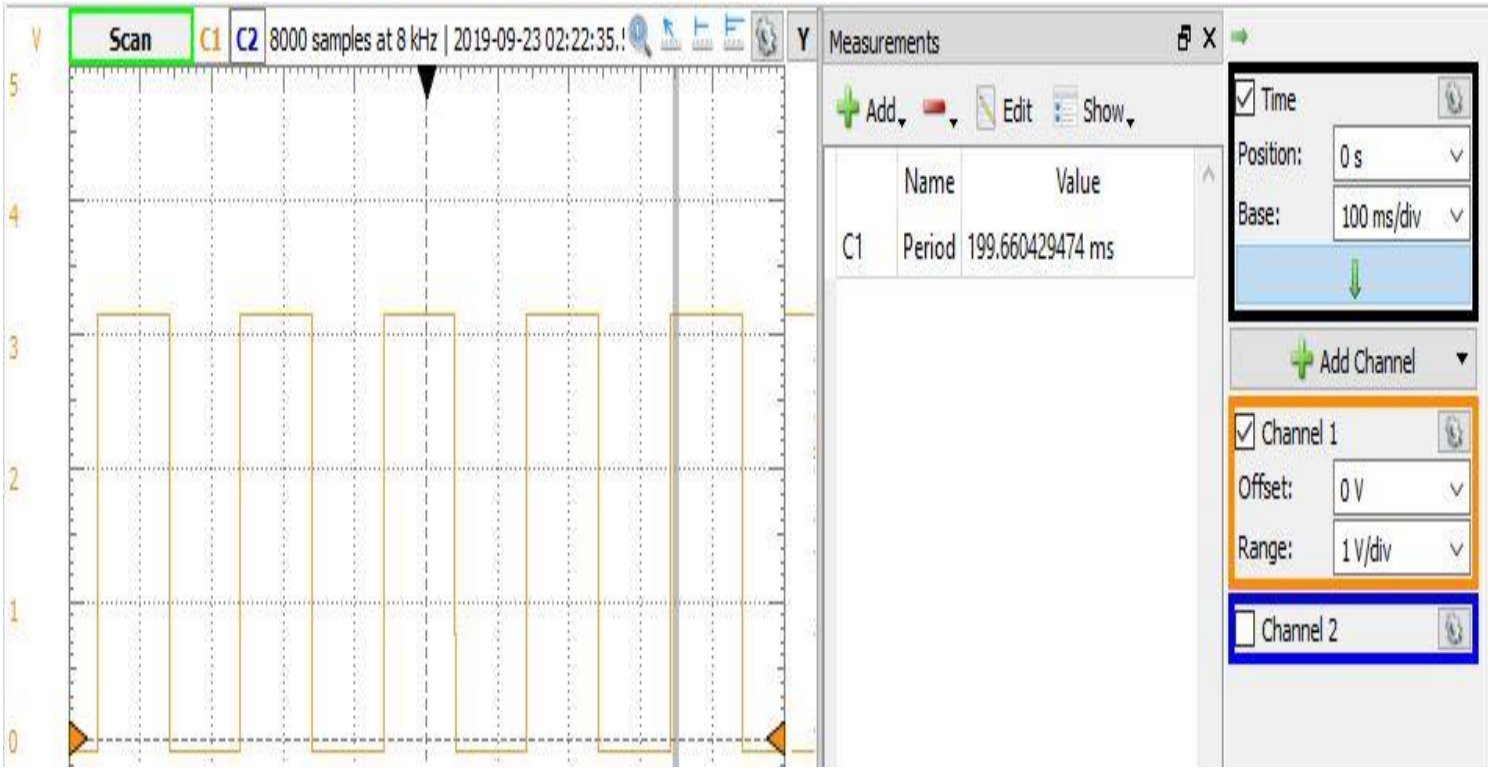
APPENDIX

Part 1: Introduction to Interrupts

```
;***********************************INCLUDES*****************************************
.include "ATxmega128a1udef.inc"
;*********************************END OF INCLUDES************************************
;*********************************DEFINED SYMBOLS************************************
.equ outhigh=0xff   ;set as input or high value
.equ inlow=0x00 ;set as output or low value
.equ stackaddress=0x3FFF    ;stack starting address
;***************************END OF DEFINED SYMBOLS***********************************
;*********************************MEMORY CONSTANTS***********************************
    ; data memory allocation
    .dseg

;***************************END OF MEMORY CONSTANTS**********************************
```

Screenshot 1: Memory Configuration of part 1.



Screenshot 2: Timer Delay with a prescaler of 256 of part 1, about 100ms. Half the period is on with a delay of 100ms till it becomes off and delay again for 1000ms to be on again, totaling a 200ms period.

Part 2a: Interrupts, Continued

```
1  ;*****************************************************************************
2  ;Lab 3 Part 2
3  ;Section #: 1823
4  ;Name: Johnny Li
5  ;Class #: 12378
6  ;PI Name: Jared Holley
7  ;Description: Interrupts, Continued
8  ;*****************************************************************************
9  ;*********************************INCLUDES************************************
10 .include "ATxmega128a1udef.inc"
11 ;*******************************END OF INCLUDES*******************************
12 ;*******************************DEFINED SYMBOLS*******************************
13 .equ outhigh=0xff   ;set as input or high value
14 .equ inlow=0x00 ;set as output or low value
15 .equ stackaddress=0x3FFF    ;stack starting address
16 ;***************************END OF DEFINED SYMBOLS*****************************
17 ;*******************************MEMORY CONSTANTS******************************
18     ; data memory allocation
19     .dseg
20
21 ;***************************END OF MEMORY CONSTANTS****************************
```

Screenshot 3: Memory Configuration of part 2a.

Part 2b: (Debounced) Interrupts, Continued

```
9  ;*********************************INCLUDES************************************
10 .include "ATxmega128a1udef.inc"
11 ;*******************************END OF INCLUDES*******************************
12 ;*******************************DEFINED SYMBOLS*******************************
13 .equ outhigh=0xff   ;set as input or high value
14 .equ inlow=0x00 ;set as output or low value
15 .equ stackaddress=0x3FFF    ;stack starting address
16 ;***************************END OF DEFINED SYMBOLS*****************************
17 ;*******************************MEMORY CONSTANTS******************************
18     ; data memory allocation
19     .dseg
20
21 ;***************************END OF MEMORY CONSTANTS****************************
```

Screenshot 4: Memory Configuration of part 2b.