**REQUIREMENTS NOT MET**

N/A

**PROBLEMS ENCOUNTERED**

N/A

**FUTURE WORK/APPLICATIONS**

The content in this lab can be used in future applications to eliminate switch bouncing in the most efficient way possible, implementing interrupts, and using bit masks.

**PRE-LAB EXERCISES**

**i. Assuming that no interrupt has been previously configured, devise and describe a generalized series of steps for configuring any interrupt within the ATxmega128A1U, i.e., not just an interrupt within the TC system.**

**Configure an interrupt source**

**Set interrupt condition and interrupt level(low, medium, or high)**

**Set the PMIC**

**Enable global interrupts**

**ii. Explain what happens in hardware (in other words, without the programmer’s intervention) when the processor detects and then services (and returns from) an interrupt. Be as specific as possible, referencing certain registers when appropriate. You can assume that the reti instruction does not count as programmer intervention. You may provide a flowchart as a response, if desired.**

**When an interrupt is detected, a flag is raised in the PMIC. When this occurs, the program memory address for the most recent instruction is added to the stack. The PC is loaded with the program memory address of the interrupt vector. Assuming the programmer loaded the vector with the ISR, the ISR is executed.   
 When the ISR is finished executing, the PMIC EX flags are cleared, and the PC is loaded with the most recent 2 bytes on the stack (our program instruction when the interrupt was called).**

**PSEUDOCODE/FLOWCHARTS**

**SECTION 1**

**A picture containing text, screenshot, diagram, font

Description automatically generated  
Figure 1: Flowchart for “lab3\_1.asm”  
The TC interrupt is enabled first, then the TC.   
Some useless code is then executed, waiting for an interrupt.**

**Section 2a**

**A picture containing text, screenshot, diagram, font

Description automatically generated  
Figure 2: Flowcharts for “lab3\_2a.asm”**

**Section 2b**

**A picture containing text, screenshot, diagram, parallel

Description automatically generated  
Figure 3: Flowchart for “lab3\_2b.asm”**

**A picture containing text, diagram, screenshot, font

Description automatically generated  
Figure 4: Second flowchart for “lab3\_2b.asm”**

**PROGRAM CODE**

**SECTION 1**

;Lab 3, Section 1

;Name: Steven Miller

;Class #: 11318

;PI Name: Anthony Stross

;Description: triggers an overflow interrupt every 84ms

.include "ATxmega128a1udef.inc"

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.EQU input = 0b00000000

.EQU output = 0b11111111

.EQU prescalar = 1024

.EQU sysclk = 2000000

.EQU reciprocal = 1/.084 ;idk how to spell reciprocal

.EQU offset =-11 ;correcting for imprecision

.equ stack\_init = 0x3FFF

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.CSEG

.org 0x0000

rjmp main

.CSEG

.org TCC0\_OVF\_vect

rjmp TC\_INT

.CSEG

.org 0x0200

MAIN:

;initialize stack pointer

ldi r16, low(stack\_init)

out CPU\_SPL, r16

ldi r16, high(stack\_init)

out CPU\_SPH, r16

rcall init\_tc\_int

rcall init\_tc

;toggle output port

loop:

;wait for interrupt

nop

rjmp loop

rjmp loop

end:

rjmp end

;\*\*\*\*\*\*\*\*\*\*\*END MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: INIT\_TC

; Purpose: To initialize the relevant timer/counter modules, as pertains to

; application.

; Input(s): N/A

; Output: N/A

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

INIT\_TC:

;initialize count register

ldi r16,0

sts TCC0\_CNT, r16

sts TCC0\_CNT+1,r16

;initialize period register

ldi r16,low(((sysclk/prescalar)/reciprocal)+offset)

sts TCC0\_PER, r16

ldi r16,high(((sysclk/prescalar)/reciprocal)+offset)

sts TCC0\_PER+1,r16

;initialize clksel

ldi r16, TC\_CLKSEL\_DIV1024\_gc

sts TCC0\_CTRLA,r16

ldi r16, input

sts TCC0\_CTRLB,r16

ret

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: INIT\_TC\_INT

; Purpose: To initialize the OVF interrupt

; Input(s): N/A

; Output: N/A

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

INIT\_TC\_INT:

;store registers

push r16

;initialize port c for output

ldi r16, output

sts PORTC\_DIR, r16

;enable tcc0 ovf interrupts, set priority to medium

ldi r16, 0b00000010

sts TCC0\_INTCTRLA,r16

;enable global interrupts

sts PMIC\_CTRL,r16

sei

pop r16

ret

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: TC\_INT

; Purpose: The TC interrupt service routine

; Input(s): TCC0\_INTFLAGS, CPU\_SREG

; Output: PORTC\_OUTTGL

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TC\_INT:

;push cpu sreg to stack

push r20

lds r20,CPU\_SREG

push r20

;push registers to stack

push r16

;toggle port c output

ldi r16,0b11111111

sts PORTC\_OUTTGL,r16

;pop registers and sreg from stack

pop r16

pop r20

sts CPU\_SREG, r20

pop r20

reti

**Section 2a**

;Lab 3, Section 2a

;Name: Steven Miller

;Class #: 11318

;PI Name: Anthony Stross

;Description: triggers an interrupt every time s2 on the SLB is pressed

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.include "ATxmega128a1udef.inc"

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.EQU input = 0b00000000

.EQU bit3 = 0b00001000

.EQU output = 0b11111111

.equ stack\_init = 0x3FFF

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.def global\_r20 = r20

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.CSEG

.org 0x0000

rjmp main

.CSEG

.org PORTF\_INT0\_vect

rjmp S2\_INT

.CSEG

.ORG 0x0200

main:

;set stack pointer

ldi r16, low(stack\_init)

out CPU\_SPL, r16

ldi r16, high(stack\_init)

out CPU\_SPH, r16

;initialization subroutines

rcall port\_init

rcall s2\_int\_init

ldi r16, 0b00100000

loop:

sts PORTD\_OUTTGL,r16

rjmp loop

end:

rjmp end

;INPUTS: SWITCHES

;OUTPUTS: LEDS

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: PORT\_INIT

; Purpose: TO INITIALIZE INPUT AND OUTPUT PORTS

; Input(s): S2\_SLB (PORTF\_PIN3)

; Output: SLB\_LEDS (PORTC)

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PORT\_INIT:

;save registers

push r16

;set s2 slb as input

ldi r16, bit3

sts PORTF\_DIRCLR,r16

;set slb\_leds as outputs

ldi r16,output

sts PORTC\_DIRSET,r16

;invert SLB LEDS by using a mask

ldi r16,0xff

sts PORTCFG\_MPCMASK,r16

ldi r16,0b01000000

sts PORTC\_PIN0CTRL,r16

;set green led as output

ldi r16,0b00100000

sts PORTD\_DIRSET,r16

ldi r16, 0b11011111

sts PORTD\_OUT,r16

;restore from stack

pop r16

RET

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: S2\_INT\_INIT

; Purpose: TO INITIALIZE S2 INTERRUPTS

; Input(s): N/A

; Output: N/A

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

S2\_INT\_INIT:

;save registers

push r16

;set s2 on slb as interrupt source

;sets interrupt level as medium

ldi r16, 0b00000010

sts PORTF\_INTCTRL,r16

;sets s2 slb as interrupt source

ldi r16, bit3

sts PORTF\_INT0MASK,r16

;set interrupt trigger to rising edge

ldi r16, 0b00000001

sts PORTF\_PIN3CTRL,r16

;set PMIC

ldi r16, 0b00000010

sts PMIC\_CTRL, r16

;enable global interrupts

sei

;restore from stack

pop r16

RET

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: S2\_INT

; Purpose: THE S2 ISR

; Input(s): N/A

; Output: SLB\_LEDS (PORTC)

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

S2\_INT:

;save registers from stack

push r16

lds r16, CPU\_SREG

push r16

;increment global count register

inc global\_r20

;display count on leds

sts PORTC\_OUT,global\_r20

;restore registers

pop r16

sts CPU\_SREG, r16

pop r16

RETI

**Section 2b**

;Lab 3, Section 2b

;Name: Steven Miller

;Class #: 11318

;PI Name: Anthony Stross

;Description: displays binary number in register 21 on the leds

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.include "ATxmega128a1udef.inc"

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.EQU input = 0b00000000

.EQU bit3 = 0b00001000

.EQU output = 0b11111111

.EQU stack\_init = 0x3FFF

.EQU prescalar = 1024

.EQU sysclk = 2000000

.EQU reciprocal = 1/.01 ;idk how to spell reciprocal

.EQU offset =0 ;correcting for imprecision

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.def global\_r21 = r21

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.CSEG

.org 0x0000

rjmp main

;set interrupt vectors

.CSEG

.org TCC0\_OVF\_vect

rjmp TC\_INT

.ORG PORTF\_INT0\_vect

rjmp S2\_INT

.CSEG

.ORG 0x0200

main:

;set stack pointer

ldi r16, low(stack\_init)

out CPU\_SPL, r16

ldi r16, high(stack\_init)

out CPU\_SPH, r16

;initialization subroutines

rcall port\_init

rcall s2\_int\_init

ldi r16, 0b00100000

loop:

sts PORTD\_OUTTGL,r16

rjmp loop

end:

rjmp end

;\*\*\*\*\*\*\*\*\*\*\*END MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: PORT\_INIT

; Purpose: TO INITIALIZE INPUT AND OUTPUT PORTS

; Input(s): S2\_SLB (PORTF\_PIN3)

; Output: SLB\_LEDS (PORTC)

; Registers affected: PORTF\_DIR,PORTCFG\_MPCMASK,PORTC\_DIR,PORTC\_PIN0CTRL,PORTD\_DIR

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PORT\_INIT:

;save registers

push r16

;set s2 slb as input

ldi r16, bit3

sts PORTF\_DIRCLR,r16

;set slb\_leds as outputs

ldi r16,output

sts PORTC\_DIRSET,r16

;invert SLB LEDS by using a mask

ldi r16,0xff

sts PORTCFG\_MPCMASK,r16

ldi r16,0b01000000

sts PORTC\_PIN0CTRL,r16

;set green led as output

ldi r16,0b00100000

sts PORTD\_DIRSET,r16

ldi r16, 0b11011111

sts PORTD\_OUT,r16

;restore from stack

pop r16

RET

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: S2\_INT\_INIT

; Purpose: TO INITIALIZE S2 INTERRUPTS

; Input(s): N/A

; Output: N/A

; Registers affected: PORTF\_INTCTRL, PORTF\_INT0MASK,PMIC\_CTRL,PORTF\_PIN3CTRL

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

S2\_INT\_INIT:

;save registers

push r16

;set s2 on slb as interrupt source

;sets interrupt level as medium

ldi r16, 0b00000010

sts PORTF\_INTCTRL,r16

;sets s2 slb as interrupt source

ldi r16, bit3

sts PORTF\_INT0MASK,r16

;set interrupt trigger to either edge

ldi r16, 0b00000000

sts PORTF\_PIN3CTRL,r16

;set PMIC

ldi r16, 0b00000010

sts PMIC\_CTRL, r16

;enable global interrupts

sei

;restore from stack

pop r16

RET

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: S2\_INT

; Purpose: THE S2 ISR

; Input(s): N/A

; Output: SLB\_LEDS (PORTC)

; Registers affected: PORTF\_INTCTRL

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

S2\_INT:

;save registers from stack

push r20

lds r20, CPU\_SREG

push r20

push r16

;disable io interrupts

ldi r16, 0b00000000

sts PORTF\_INTCTRL,r16

rcall init\_tc\_int

rcall init\_tc

pop r16

pop r20

sts CPU\_SREG,r20

pop r20

RETI

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: INIT\_TC

; Purpose: To initialize the relevant timer/counter modules, as pertains to

; application.

; Input(s): N/A

; Output: N/A

; Registers affected: TCC0\_CNT,TCC0\_PER,TCC0\_CTRLA,TCC0\_CTRLB

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

INIT\_TC:

push r16

;initialize count register

ldi r16,0

sts TCC0\_CNT, r16

sts TCC0\_CNT+1,r16

;initialize period register

ldi r16,low(((sysclk/prescalar)/reciprocal)+offset)

sts TCC0\_PER, r16

ldi r16,high(((sysclk/prescalar)/reciprocal)+offset)

sts TCC0\_PER+1,r16

;initialize clksel

ldi r16, TC\_CLKSEL\_DIV1024\_gc

sts TCC0\_CTRLA,r16

ldi r16, input

sts TCC0\_CTRLB,r16

pop r16

ret

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: INIT\_TC\_INT

; Purpose: To initialize the OVF interrupt

; Input(s): N/A

; Output: N/A

; Registers affected: TCC0\_CTRLA

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

INIT\_TC\_INT:

;store registers

push r16

;enable tcc0 ovf interrupts, set priority to medium

ldi r16, 0b00000010

sts TCC0\_INTCTRLA,r16

pop r16

ret

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; Name: TC\_INT

; Purpose: The TC interrupt service routine

; Input(s): TCC0\_INTFLAGS, CPU\_SREG

; Output: PORTC\_OUTTGL

; Registers affected: TCC0\_CNT,TCC0\_PER,TCC0\_CTRLA,TCC0\_CTRLB,TCC0\_INTCTRLA

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TC\_INT:

;push cpu sreg to stack

push r20

lds r20,CPU\_SREG

push r20

;push registers to stack

push r16

;disable TC

ldi r16, 0b00000000

sts TCC0\_CTRLA, r16

;disable tc interrupt

ldi r16, 0b00000000

sts TCC0\_INTCTRLA,r16

;reset period and counter

sts TCC0\_PER, r16

sts TCC0\_PER+1, r16

sts TCC0\_CNT,r16

sts TCC0\_CNT+1,r16

;get s2 slb switch status

lds r16, PORTF\_IN

;increment if on

sbrc r16,3

rjmp enable

;increment global count register

inc global\_r21

;display count on leds

sts PORTC\_OUT,global\_r21

;enable io interrupt

enable:

ldi r16, 0b00000010

sts PORTF\_INTCTRL,r16

ldi r16, 0B00000001

sts PORTF\_INTFLAGS,r16

;pop registers and sreg from stack

pop r16

pop r20

sts CPU\_SREG, r20

pop r20

reti

**APPENDIX**

**A screen shot of a graph

Description automatically generated with medium confidence  
Figure 5: Waveform using interrupt enabled TC**