

Bryan Takemoto

CPE301 – SPRING 2018

Design Assignment 2

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

The student understands that all required components should be submitted in complete for grading of this assignment.

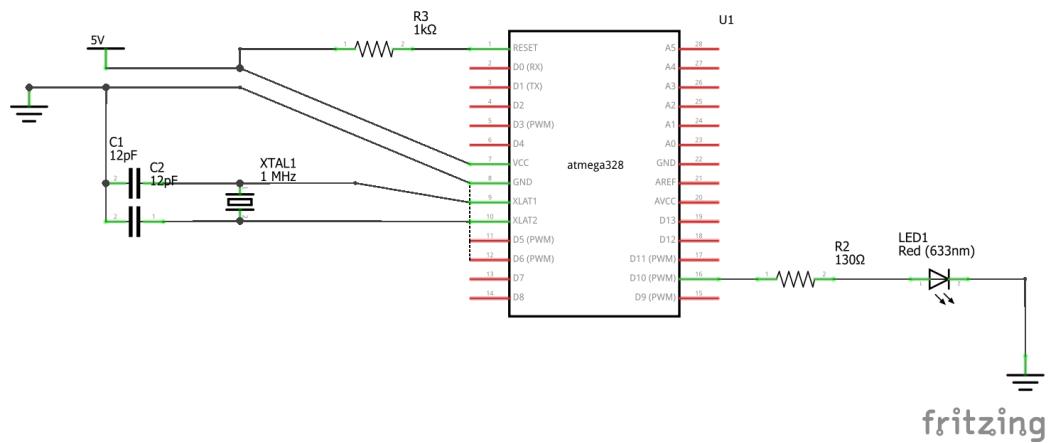
NO	SUBMISSION ITEM	COMPLETED (Y/N)	MARKS (/MAX)
1	COMPONENTS LIST, SCHEMATICS, AND BREADBOARD		
2.	TASK #1 ASSEMBLY CODE		
3.	TASK #1 C CODE		
4.	TASK #1 SIMULATION PICTURES		
5.	TASK #2 ASSEMBLY CODE		
6.	TASK #2 C CODE		
7.	TASK #2 SIMULATION PICTURES		
8.	TASK #3 ASSEMBLY CODE		
9.	TASK #3 C CODE		
10.	TASK #3 SIMULATION PICTURES		
11.	TASK #4 ASSEMBLY CODE		
12.	TASK #4 C CODE		
13.	TASK #4 SIMULATION PICTURES		
14.	TASK #5 ASSEMBLY CODE		
15.	TASK #5 C CODE		
16.	TASK #5 SIMULATION PICTURES		
17.	YouTube Links		

1. COMPONENTS LIST AND SCHEMATICS

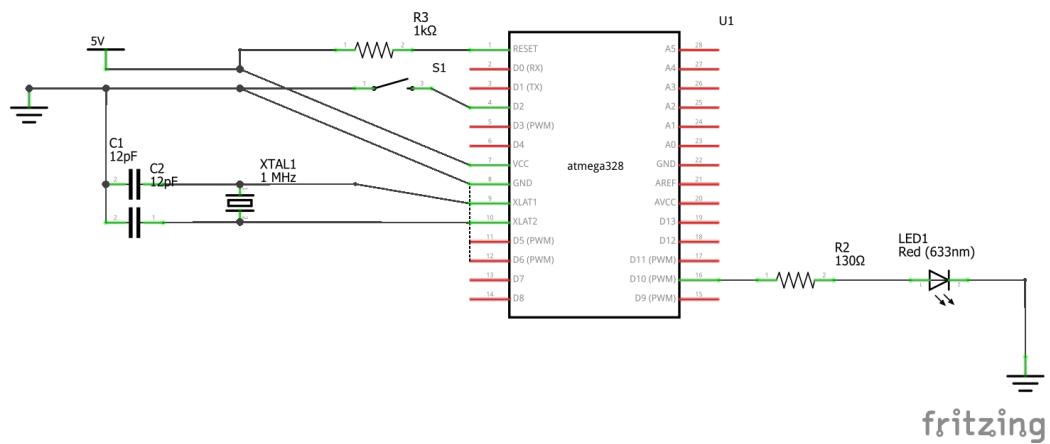
Components:

Red LED

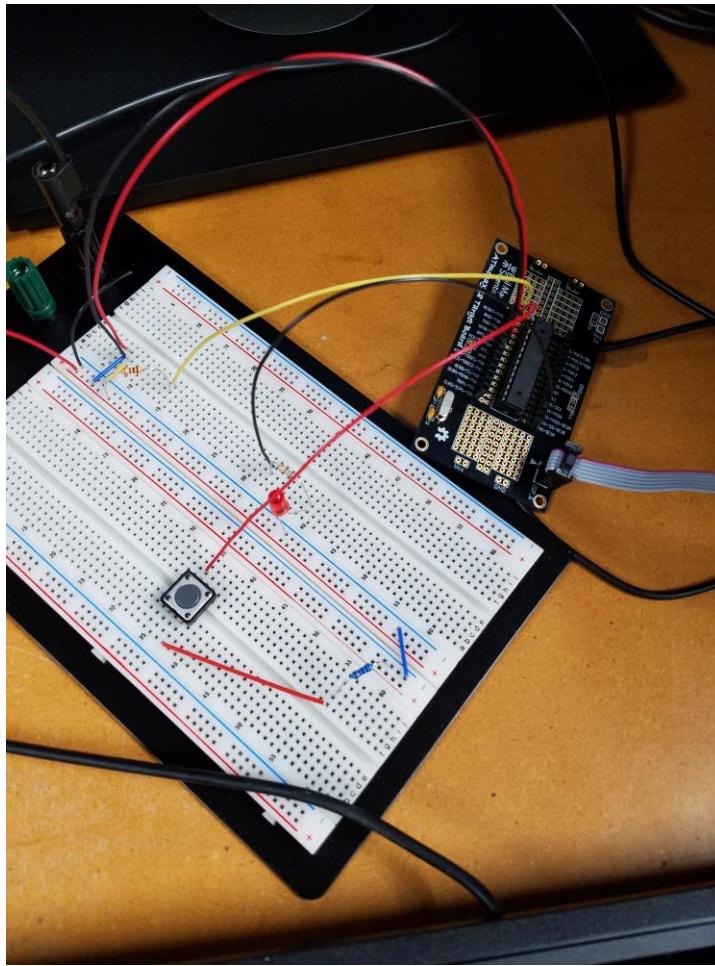
Switch



Schematic for Task #1, #3, and #4



Schematic for Task #2 and #5



Breadboard overview

2. Task #1 Assembly Code

```
.ORG 0
    ; Initialize the stack
LDI      R16, HIGH(RAMEND)
OUT     SPH, R16
LDI      R16, LOW(RAMEND)
OUT    SPL, R16

SBI    DDRB, 2      ; Set PORTB as output
LDI      R16, (1<<2) ; Output signal of PORTB
LDI      R17, 0x0
OUT    PORTB, R17   ; Clear PORTB

    ; Toggle PB2 port
BEGIN:
RCALL  DELAY
EOR      R17, R16    ; Toggle logic
OUT    PORTB, R17    ; Output the toggled signal
RJMP  BEGIN
```

```

; Delay for 0.25 second
; Delay subroutine using Timer1 (F = 0.5 MHz)
DELAY:
    ; Set the TCNT1 = 65536-15625 = 49911
    LDI      R20, HIGH(-15625)
    STS      TCNT1H, R20
    LDI      R20, LOW(-15625)
    STS      TCNT1L, R20
    ; Start the timer with a prescalar of 8
    LDI      R20, 0x00
    STS      TCCR1A, R20
    LDI      R20, 0x02
    STS      TCCR1B, R20
    ; Delay for 31250 cycles
LOOP:
    IN      R20, TIFR1
    SBRS   R20, TOV1
    RJMP   LOOP
    ; Stop timer and reset TOV flag
    LDI      R20, 0x0
    STS      TCCR1B, R20
    LDI      R20, (1<<TOV1)
    OUT     TIFR1, R20
    RET

```

3. Task #1 C Code

```

#include <avr/io.h>

void T1Delay();

int main(void)
{
    DDRB |= 0x04;           // Set PORTB.2 as output
    PORTB = 0x00;           // Clear PORTB
    while(1)
    {
        T1Delay();          // Call delay
        PORTB ^= 0x04;       // Toggle the LED
    }
    return 0;
}

// Function that uses Timer1 [Delay for 0.25 second]
// F = 0.5 MHz
void T1Delay()
{
    // Set the TCNT1 = 0xC2F7 [65536-15625 = 49911]
    TCNT1H = 0xC2;
    TCNT1L = 0xF7;
    // Start clock using prescalar 8
    TCCR1A = 0x00;
    TCCR1B = 0x02;
    while((TIFR1&0x01) == 0); // Loop until TOV flag is set
    TCCR1B = 0x00;           // Stop the timer
    TIFR1 |= 0x01;           // Reset TOV flag
}

```

4. Task #1 Simulation Pictures

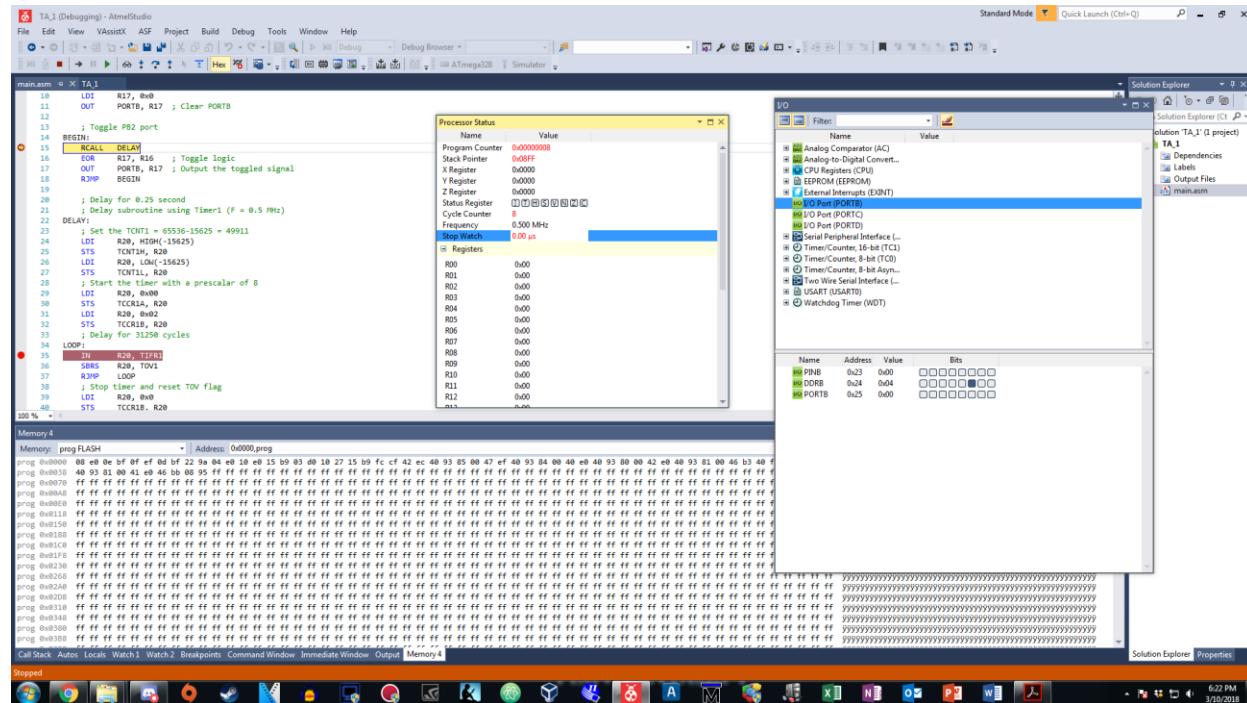


Figure 1: Before the call of delay, PB.2 is cleared and stopwatch ready (ASM)

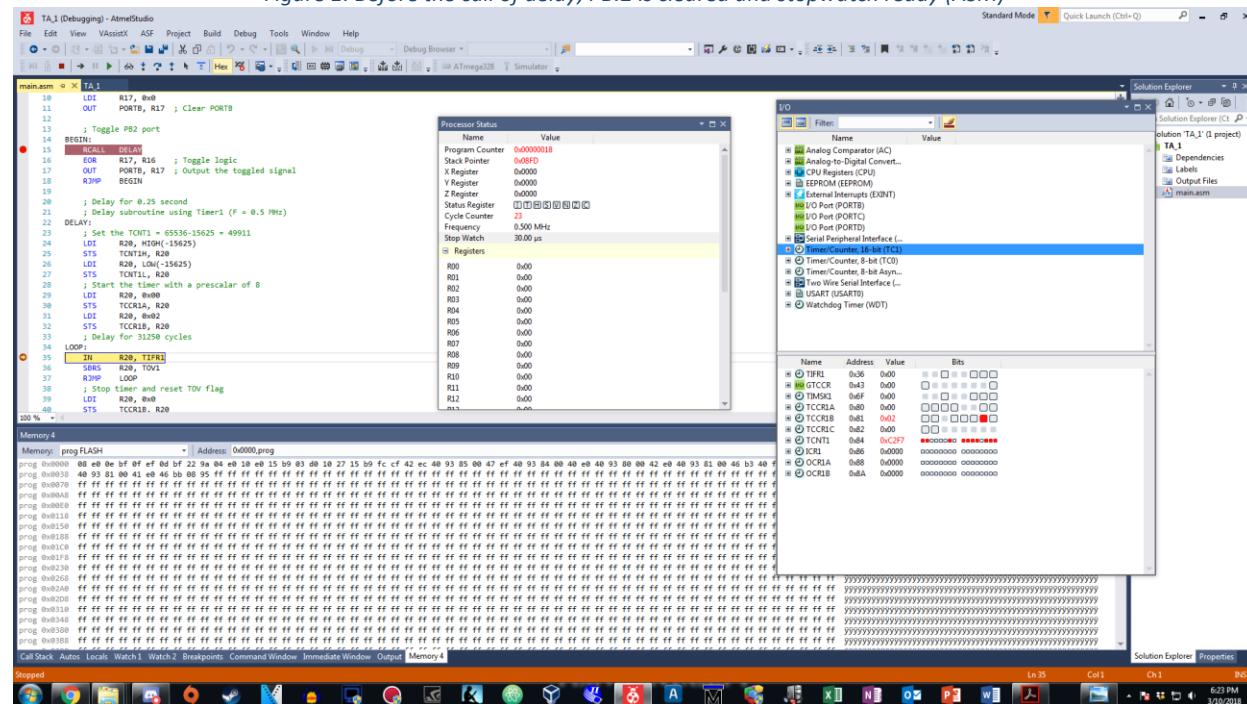


Figure 2: TIMER1 registers are set to count in the delay subroutine (ASM)

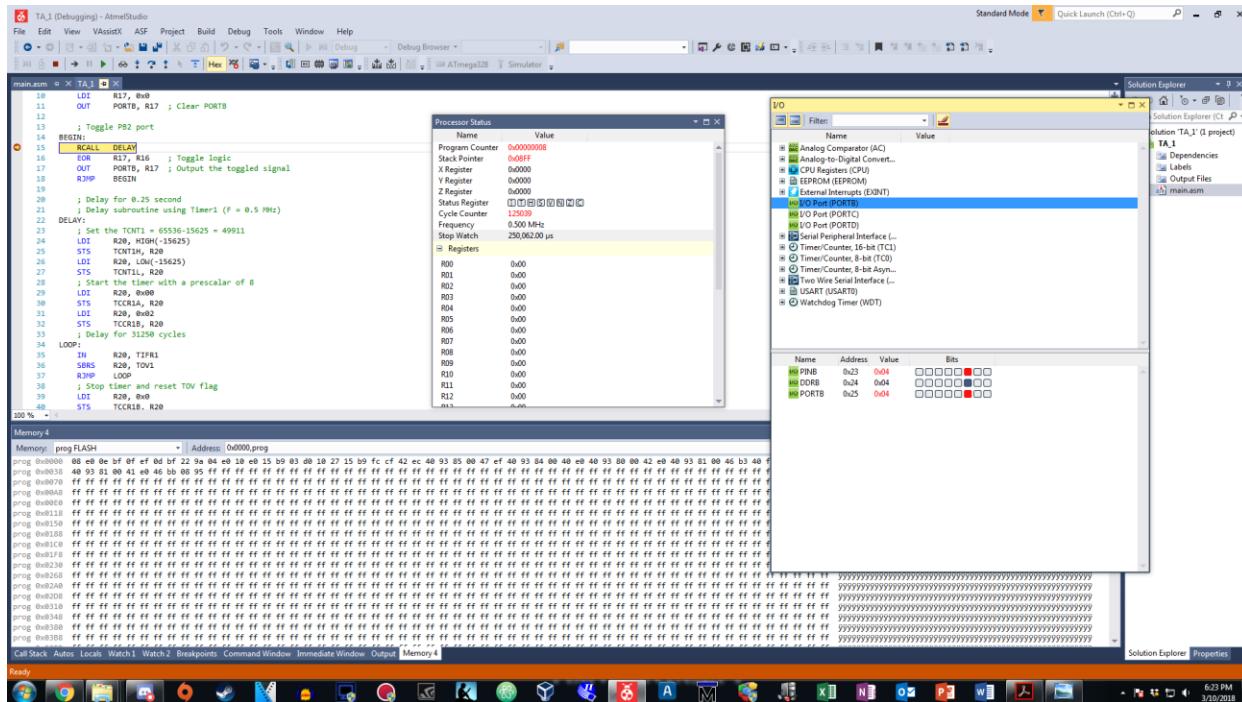


Figure 3: After 0.25 second delay, PB.2 toggles high (ASM)

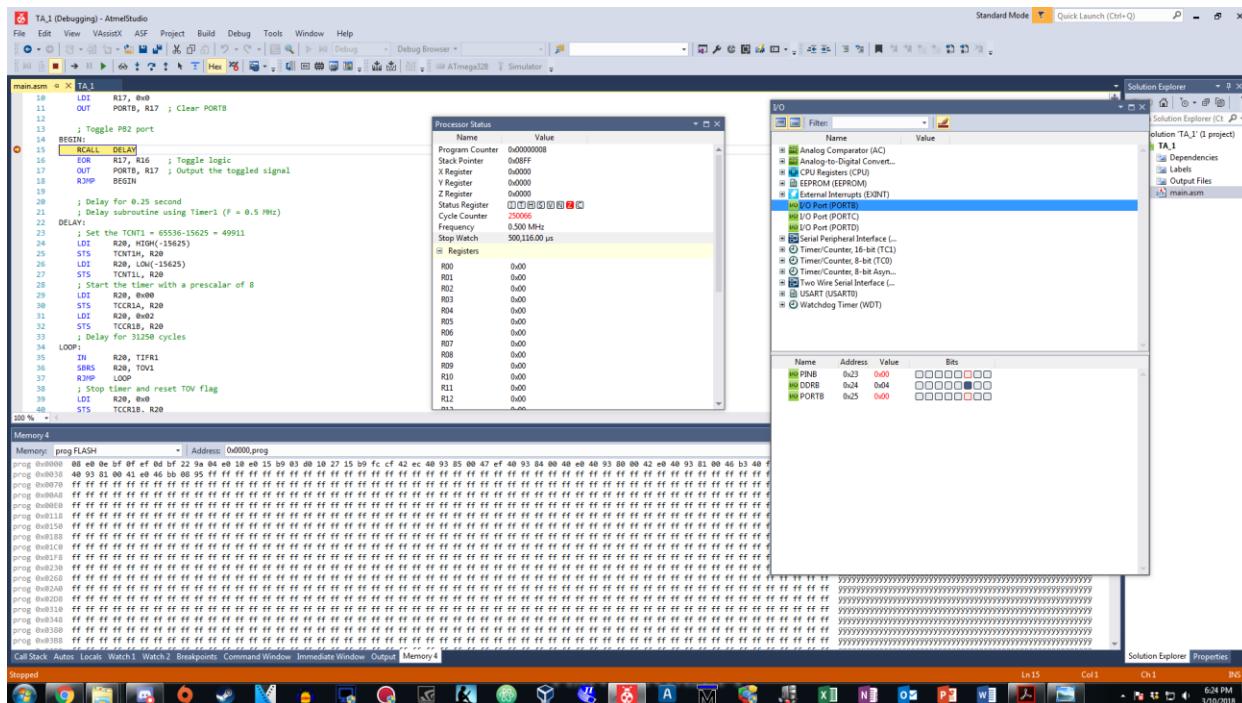


Figure 4: Another 0.25 second of delay, PB.2 toggles to low again (ASM)

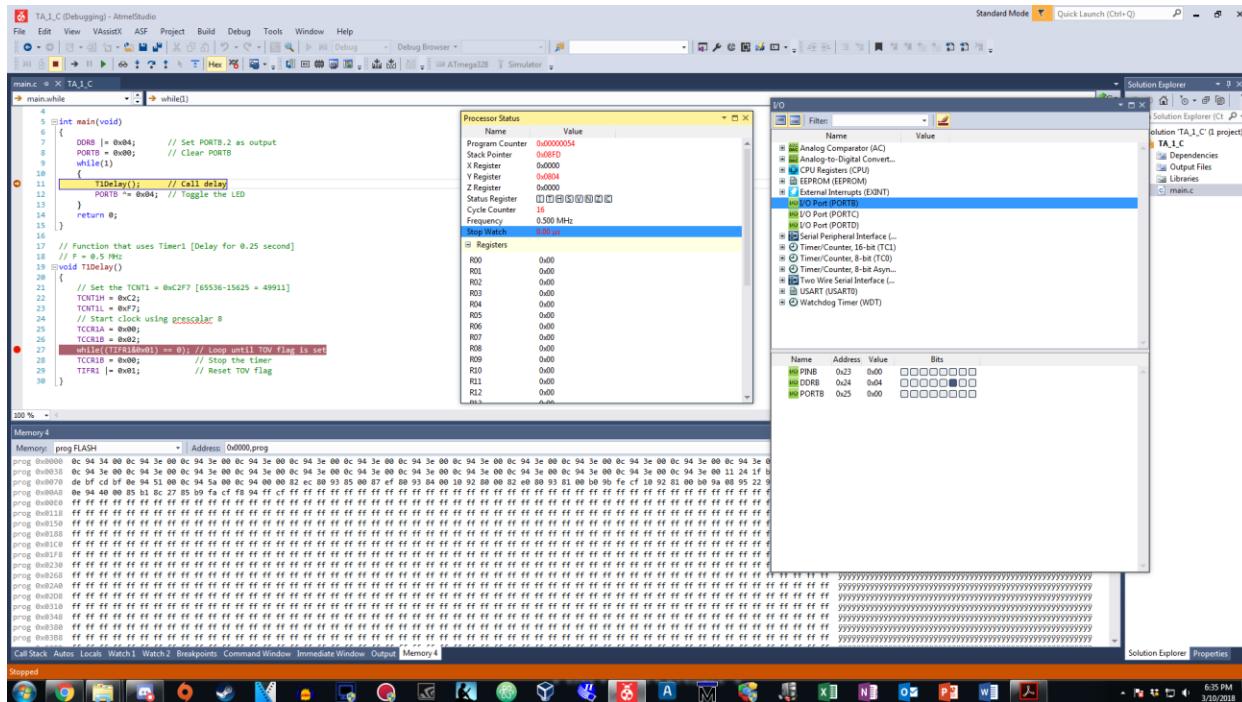


Figure 5: Before the call of delay, PB.2 is cleared and stopwatch ready (C)

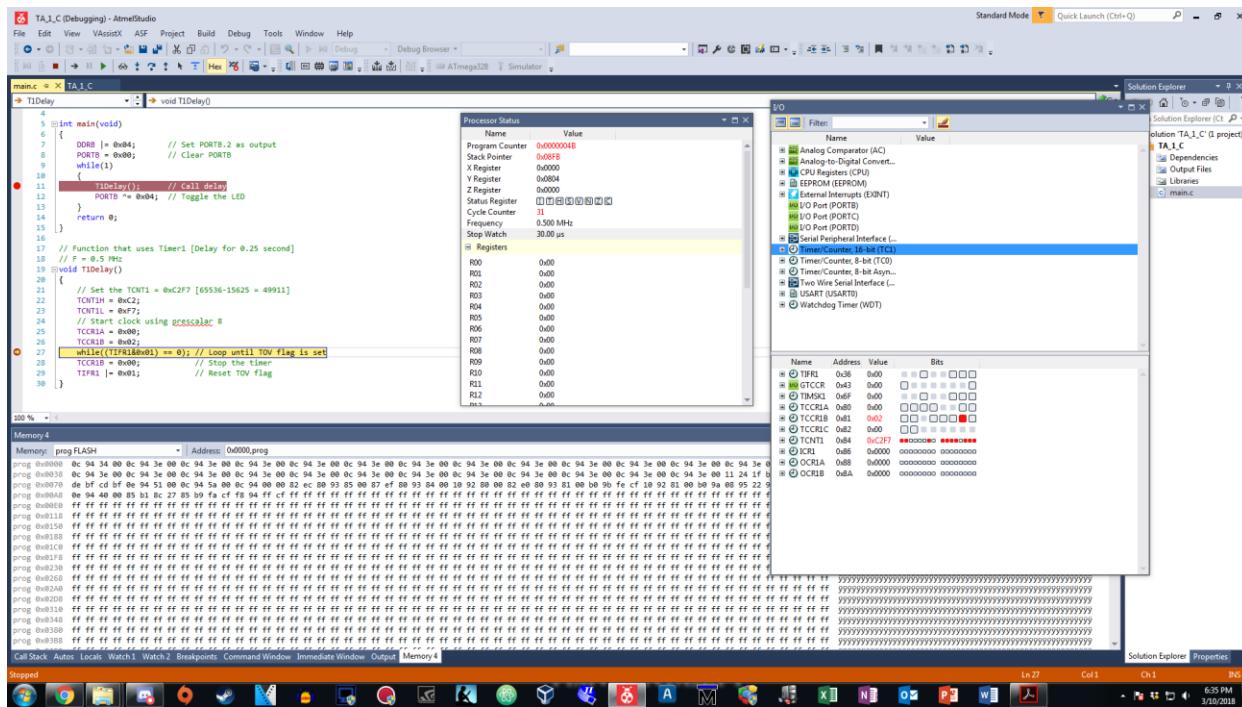


Figure 6: TIMER1 registers are set to count in the delay subroutine (C)

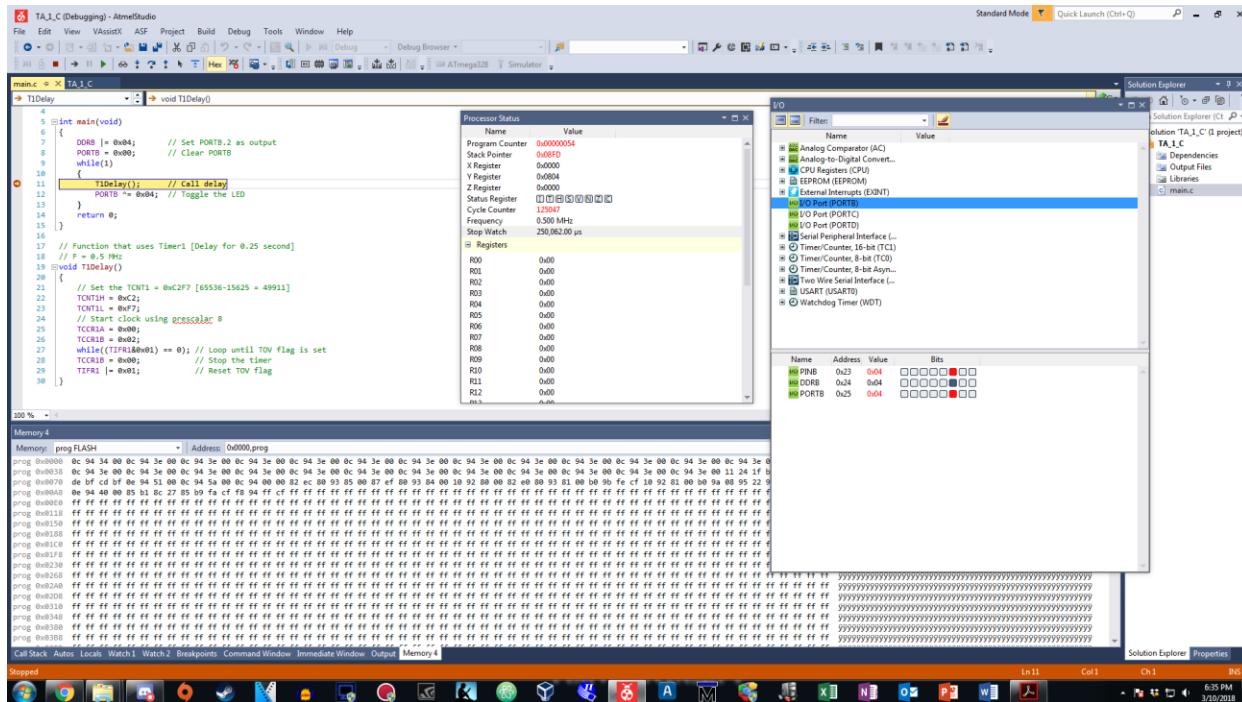


Figure 7: After 0.25 second delay, PB.2 toggles high (C)

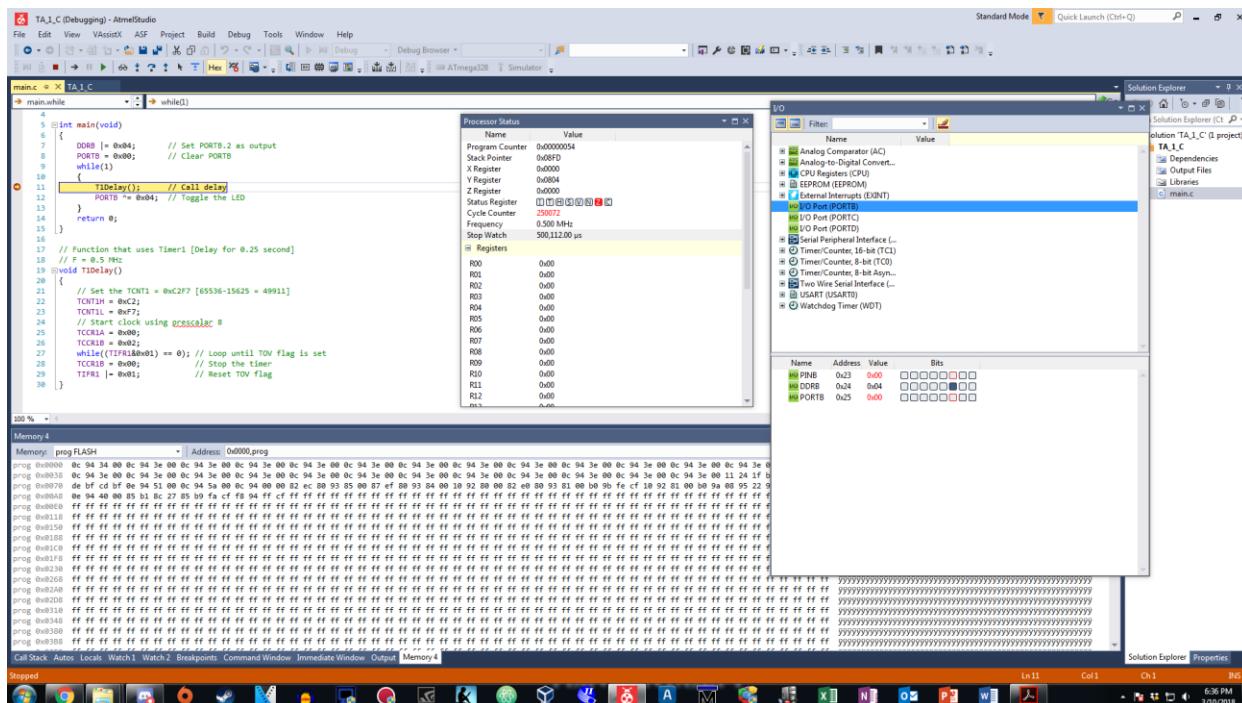


Figure 8: Another 0.25 second of delay, PB.2 toggles to low again (C)

5. Task #2 Assembly Code

```
.ORG 0
; Initialize stack
```

```

LDI      R16, HIGH(RAMEND)
OUT     SPH, R16
LDI      R16, LOW(RAMEND)
OUT    SPL, R16

; Port initialization
CBI     DDRD, 2      ; Set PORTD.2 as an input
SBI     PORTD, 2     ; Pull up the resistor of PORTD.2
SBI     DDRB, 2      ; Set PORTB.2 as an output

POLL:
SBIC   PIND, 2      ; Poll until PIND.2 is low
RJMP   POLL
SBI     PORTB, 2     ; Light up the LED

HOLD:
SBIS   PIND, 2      ; Poll until the user lets go of the switch
RJMP   HOLD
CALL   DELAY        ; Delay for 1 second after switch is off
CBI     PORTB, 2     ; Turn off the LED
RJMP   POLL

; Delay subroutine (F = 0.5 MHz) [Delay for 1 second]
DELAY:
; Set the TCNT1 = 65536-62500 = 3036
LDI    R20, HIGH(-62500)
STS    TCNT1H, R20
LDI    R20, LOW(-62500)
STS    TCNT1L, R20
; Set timer control register
LDI    R20, 0x00
STS    TCCR1A, R20
LDI    R20, 0x02
STS    TCCR1B, R20 ; Start the timer with a prescalar of 8

LOOP:
IN     R20, TIFR1    ; Check for the TOV flag
SBRS  R20, TOV0
RJMP  LOOP
; Stop timer and clear TOV flag
LDI    R20, 0x0
STS    TCCR1B, R20
LDI    R20, (1<<TOV0)
OUT   TIFR1, R20
RET

```

6. Task #2 C Code

```

#include <avr/io.h>

void T1Delay();

int main(void)
{
    DDRD = 0x00;          // Set PORTD.2 as input
    PORTD |= (1<<2);    // Pull up the resistor of PORTD.2
    DDRB = 0xFF;          // Set PORTB.2 as output
    PORTB = 0x00;          // Clear PORTB
}

```

```

// Poll until switch is pressed
while(1)
{
    if((PIND&0x04) == 0x00)
    {
        PORTB = (1<<2);           // Turn on LED
        while((PIND&0x04) == 0x00); // Poll until switch is turned off
        T1Delay();                 // Call the delay after switch is turned off
    }
    PORTB = 0x00;                  // Turn off LED
}
return 0;
}

// Delay subroutine for Timer1 [Delay for 1 second]
// F = 0.5 MHz
void T1Delay()
{
    // Set the TCNT1 = 65536-62500 = 3036 [0xBDC]
    TCNT1H = 0x0B;
    TCNT1L = 0xDC;
    // Start Timer1 with prescalar 8
    TCCR1A = 0x00;
    TCCR1B = 0x02;
    while((TIFR1&(1<<TOV1)) == 0x00);      // Check TOV flag
    TCCR1B = 0x00;                          // Stop Timer1
    TIFR1 |= (1<<TOV1);                   // Reset TOV flag in TIFR1 register
}

```

7. Task #2 Simulation Pictures

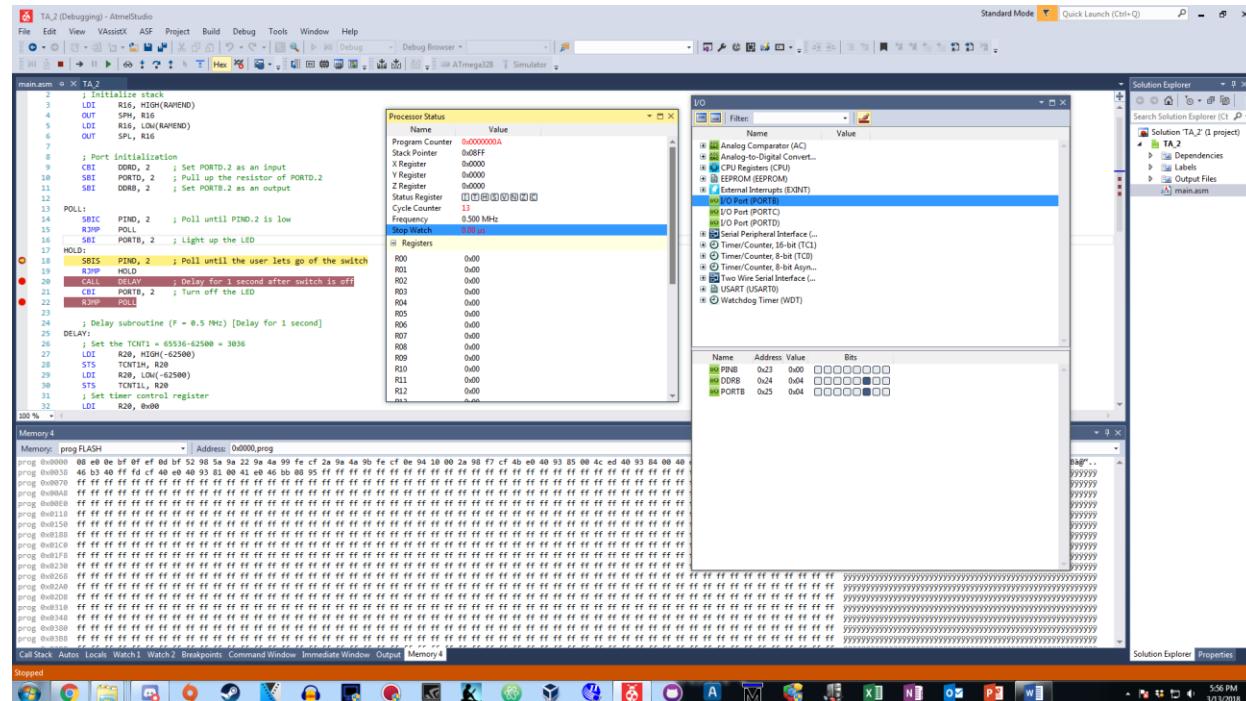


Figure 1: LED turns on at PORTB.2 and stopwatch set to measure the 1 second delay (ASM)

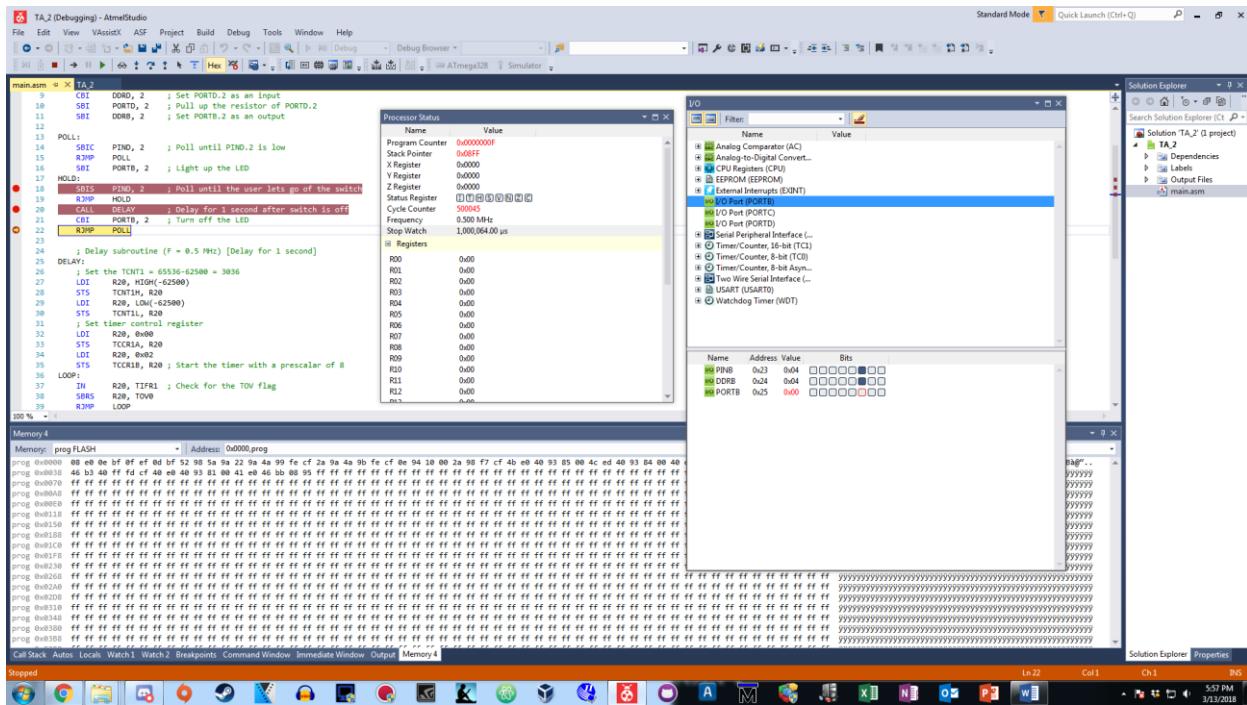


Figure 2: After 1 second delay, LED turns off at PORTB.2 (ASM)

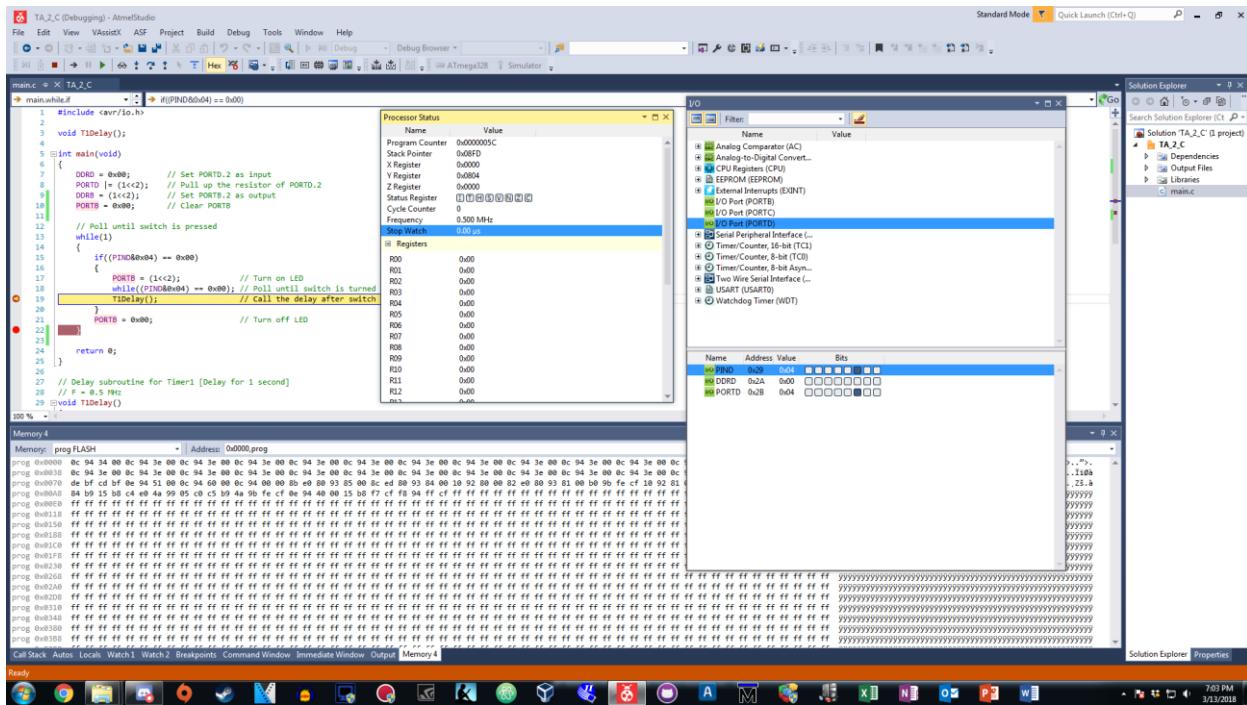


Figure 3: LED turns on at PORTB.2 and stopwatch set to measure the 1 second delay (C)

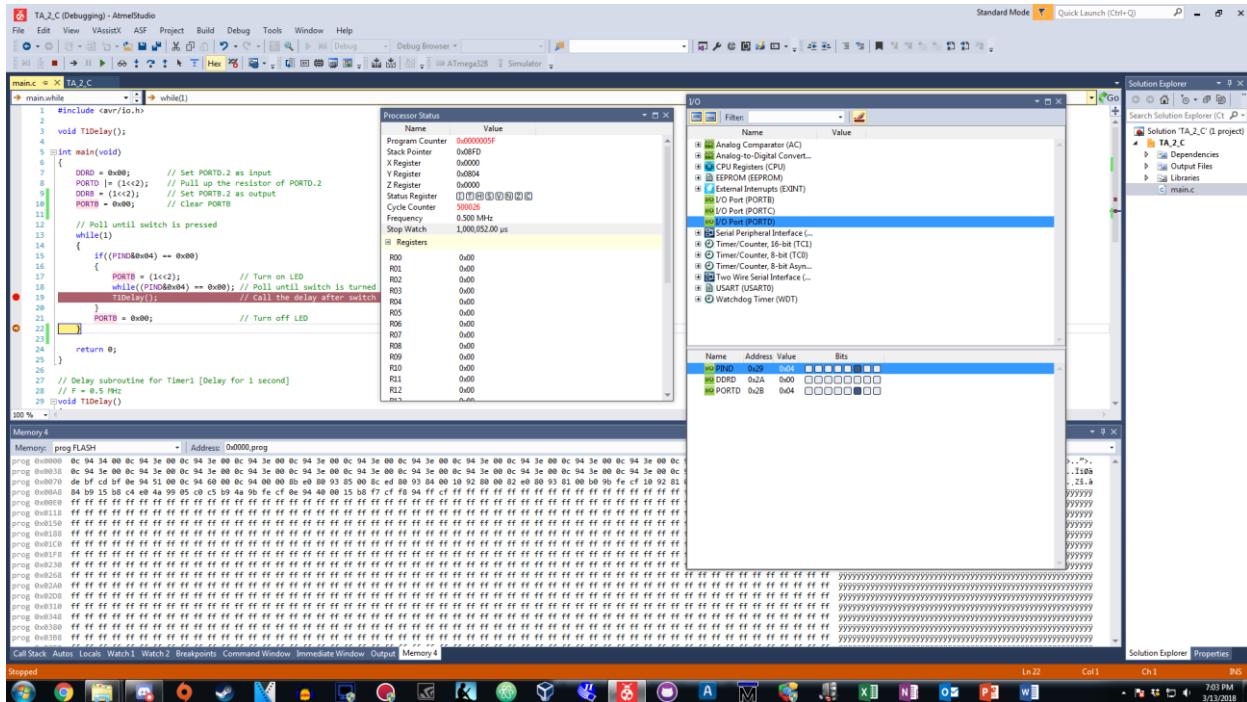


Figure 4: After 1 second delay, LED turns off at PORTB.2 (C)

8. Task #3 Assembly Code

```
.ORG 0
; Initialize the stack
LDI R16, HIGH(RAMEND)
OUT SPH, R16
LDI R16, LOW(RAMEND)
OUT SPL, R16

SBI DDRB, 2 ; Set PORTB as output
LDI R16, (1<<2) ; Output signal of PORTB
LDI R17, 0x00
OUT PORTB, R17 ; Clear PORTB

; Toggle PB2
BEGIN:
RCALL DELAY
EOR R17, R16 ; Toggle logic
OUT PORTB, R17 ; Output the toggled signal
RJMP BEGIN

; Delay for 0.25 second
; Delay subroutine using Timer0 (F = 0.5 MHz)
DELAY:
; Set TCNT0 = 256-122 = 134
LDI R20, -122
OUT TCNT0, R20
; Start the timer with a prescalar of 1024
LDI R20, 0x00
OUT TCCR0A, R20
```

```

LDI      R20, 0x05
OUT     TCCR0B, R20
; Delay for 134 cycles by checking the TOV0 flag
LOOP:
IN      R20, TIFR0
SBRS    R20, TOV0
RJMP   LOOP
; Stop timer and reset TOV flag
LDI      R20, 0x00
OUT     TCCR0B, R20
LDI      R20, (1<<TOV0)
OUT     TIFR0, R20
RET

```

9. Task #3 C Code

```

#include <avr/io.h>

void T0Delay();

int main(void)
{
    DDRB |= 0x04;           // Set PORTB.2 as output
    PORTB = 0x00;           // Clear PORTB
    while(1)
    {
        T0Delay();          // Call delay
        PORTB ^= 0x04;       // Toggle the LED
    }
    return 0;
}

// Delay for 0.25 second
// Delay subroutine using Timer0 (F = 0.5 MHz)
void T0Delay()
{
    TCNT0 = 0x86;           // Set TCNT0 = 256-122 = 134
    TCCR0A = 0x00;
    TCCR0B = 0x05;           // Start the timer0 with a prescalar of 1024
    while((TIFR0&0x01) == 0);
    TCCR0B = 0x00;           // Stop timer0
    TIFR1 |= (1<<TOV0);    // Reset TOV flag
}

```

10. Task #3 Simulation Pictures

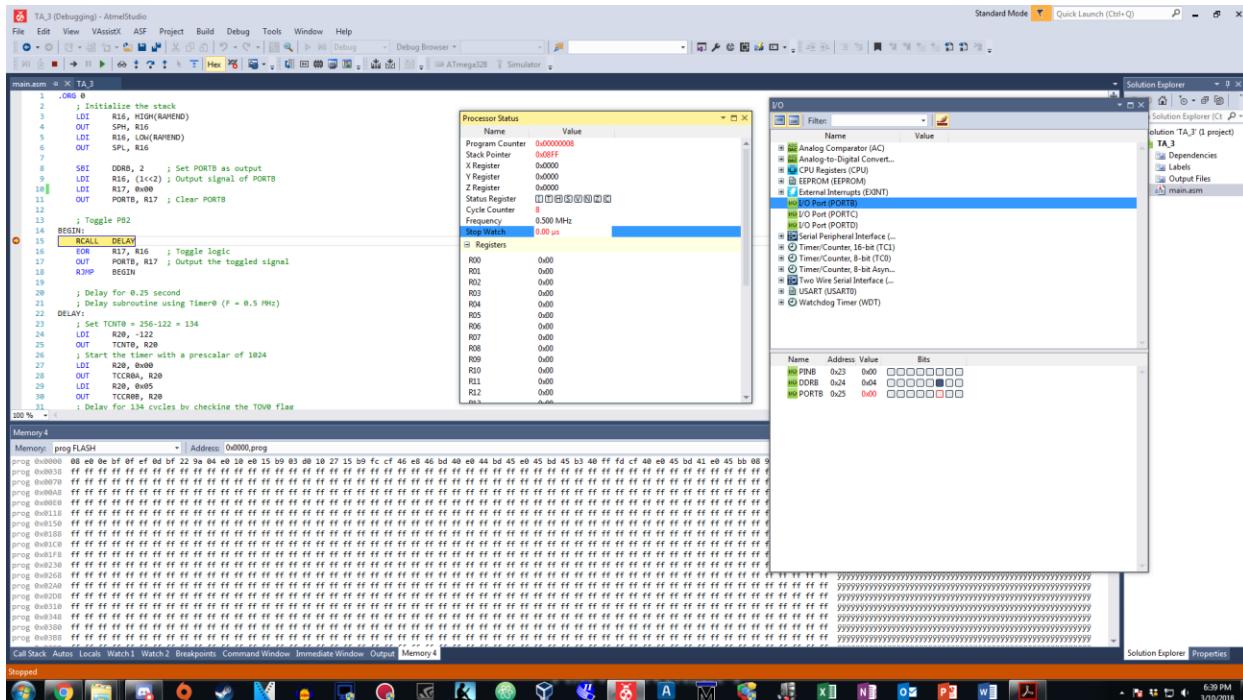


Figure 1: Before the call of delay, PB.2 is cleared and stopwatch ready (ASM)

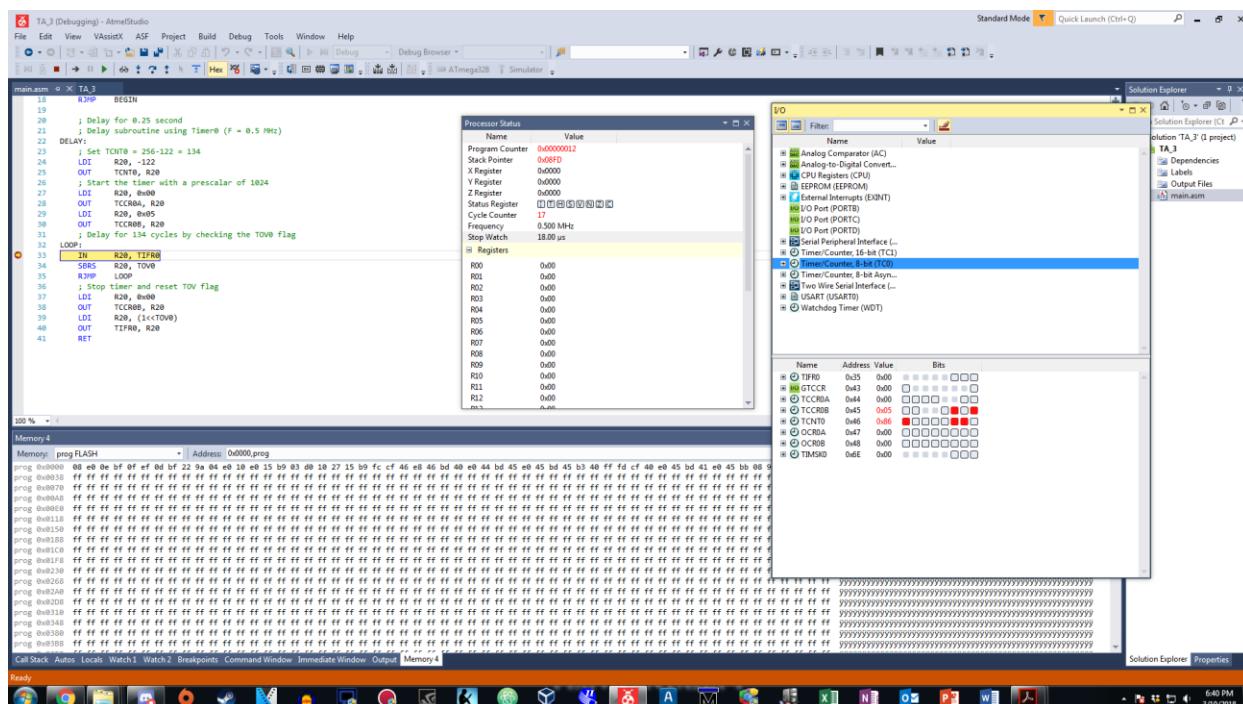


Figure 2: TIMERO registers are set to count in the delay subroutine (ASM)

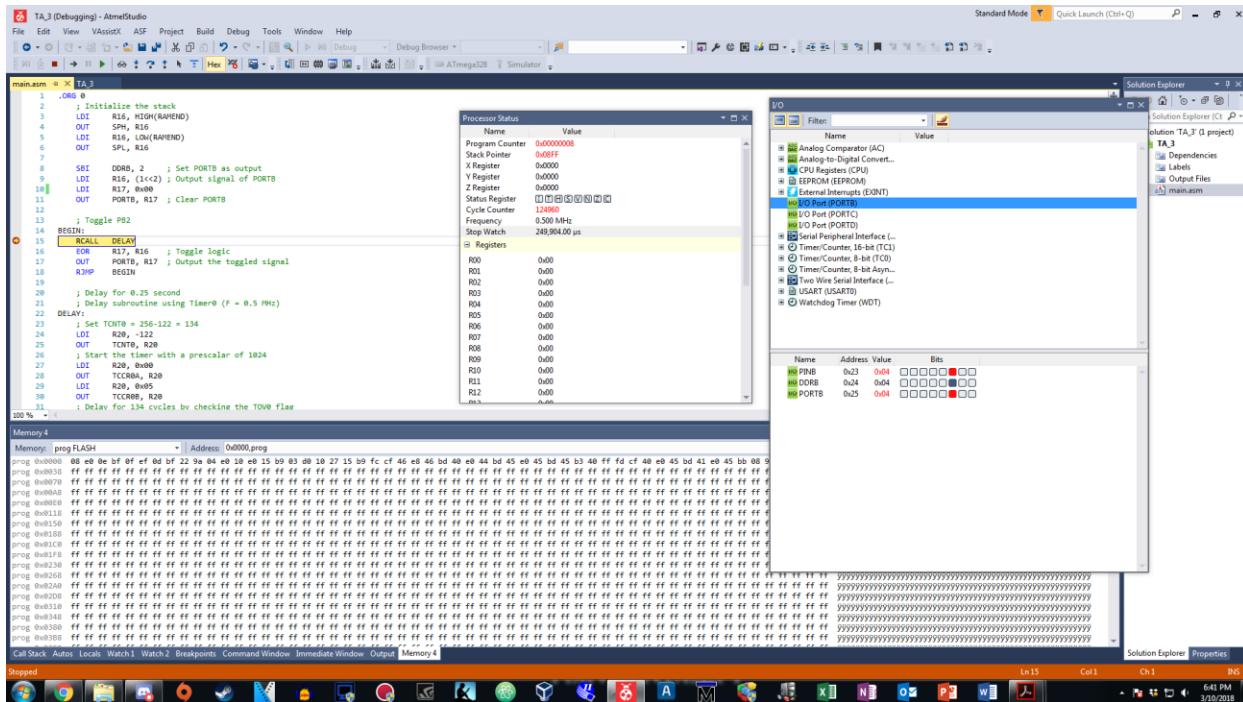


Figure 3: After 0.25 second delay, PB.2 toggles high (ASM)

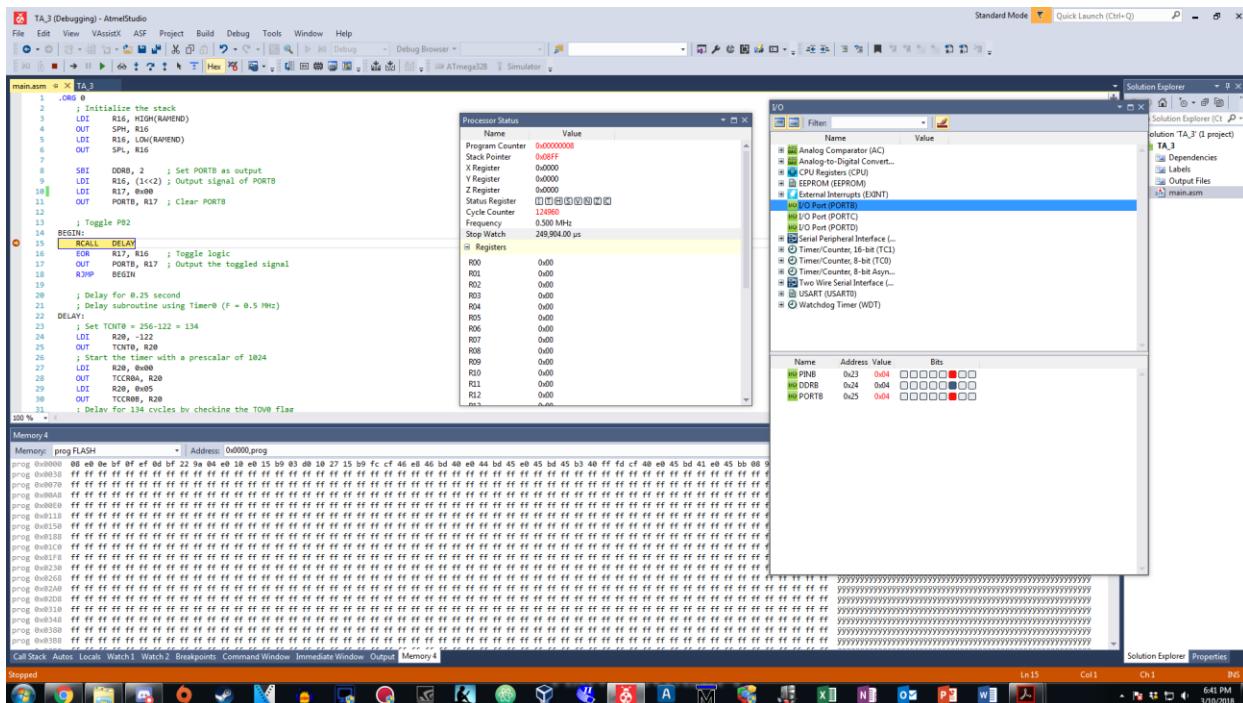


Figure 4: Another 0.25 second of delay, PB.2 toggles to low again (ASM)

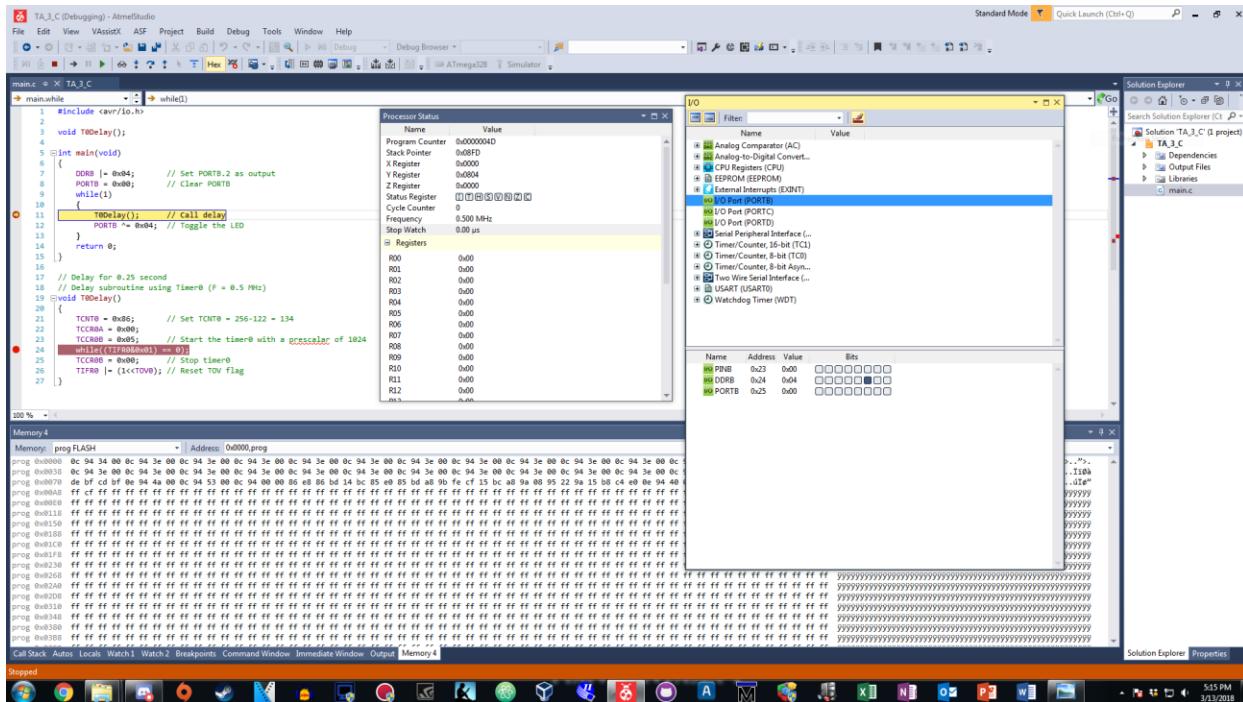


Figure 5: Before the call of delay, PB.2 is cleared and stopwatch ready (C)

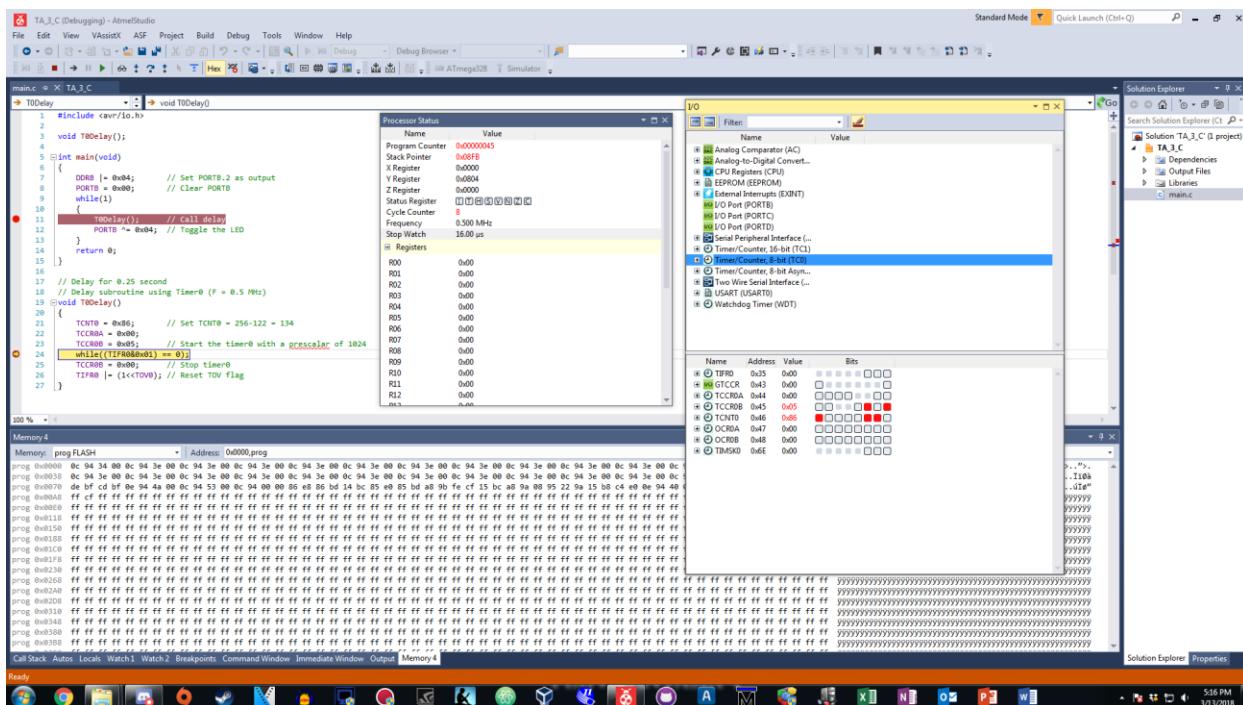


Figure 6: TIMER0 registers are set to count in the delay subroutine (C)

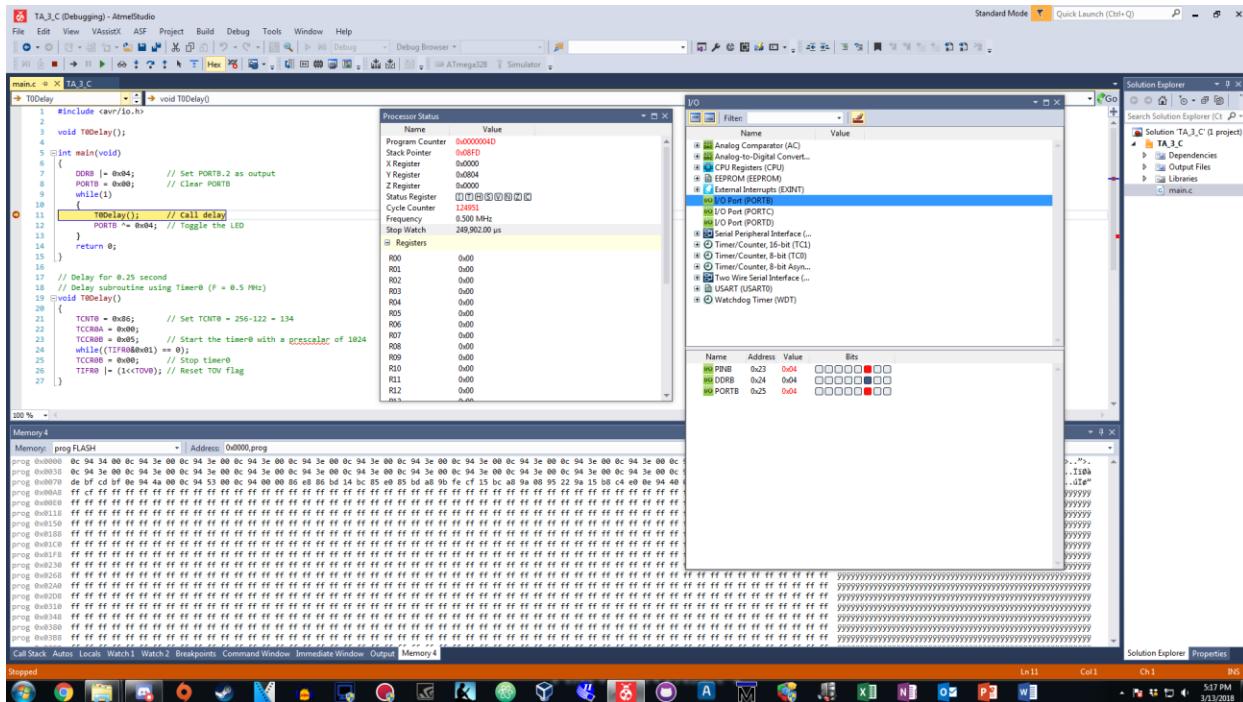


Figure 7: After 0.25 second delay, PB.2 toggles high (C)

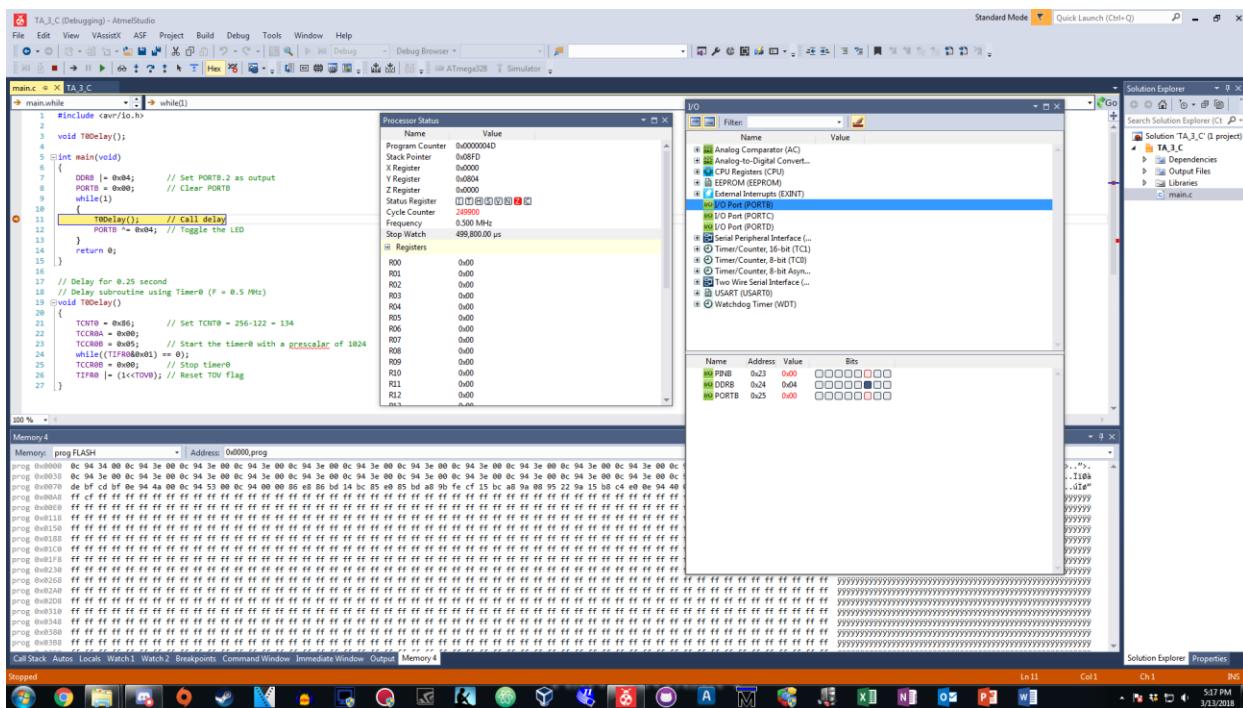


Figure 8: Another 0.25 second of delay, PB.2 toggles to low again (C)

11. Task #4 Assembly Code

.ORG 0x0000

```

        JMP      MAIN
.ORG 0x0020
        JMP      TIMER0_OVF_ISR

.ORG 0x0100
MAIN:
; Initialize the stack
LDI      R16, HIGH(RAMEND)
OUT     SPH, R16
LDI      R16, LOW(RAMEND)
OUT    SPL, R16

SBI      DDRB, 2      ; Set PORTB.2 as output
LDI      R16, (1<<2) ; Value for toggling the LED
LDI      R17, 0x00
OUT     PORTB, R17   ; Clear PORTB register
LDI      R20, -122    ; 256-122 = 134 cycles (DELAY)
OUT     TCNT0, R20    ; Load 134 into TCNT0
LDI      R20, (1<<TOIE0)
STS      TIMSK0, R20  ; Enable TIMER0 interrupt for OVF
SEI          ; Enable interrupts
LDI      R20, 0x00
OUT     TCCR0A, R20
LDI      R20, 0x05
OUT     TCCR0B, R20  ; Start TIMER0 with a prescalar of 1024 (F = 0.5 MHz)

LOOP:
RJMP    LOOP          ; Infinite loop (Wait until TIMER0_OVF interrupts)

.ORG 0x0200
; ISR for toggling PORTB.2
TIMER0_OVF_ISR:
LDI      R20, (1<<TOV0)      ; Clear TOV flag
OUT     TIFR0, R20
EOR      R17, R16           ; Toggle PORTB.2
OUT     PORTB, R17
LDI      R20, -122
OUT     TCNT0, R20          ; Reset TCNT0
RETI

```

12. Task #4 C Code

```

#include <avr/io.h>
#include <avr/interrupt.h>

int main(void)
{
    DDRB = (1<<2);      // Set PORTB.2 as output
    TCNT0 = 0x86;        // Set TCNT0 = 134 (DELAY)
    TIMSK0 = (1<<TOIE0); // Enable TIMER0 interrupt for OVF
    sei();               // Enable global interrupts
    TCCR0A = 0x00;
    TCCR0B = 0x05;       // Start TIMER0 with a prescalar of 1024 (F = 0.5 MHz)
    PORTB = 0x00;         // Clear PORTB
    // Poll until the interrupt occurs
    while(1)
    {

```

```

    }

    return 0;
}

// Subroutine to handle the TIMER0 OVF interrupt
// Clear TOV, toggle PB.2, and reset TCNT0
ISR(TIMER0_OVF_vect)
{
    TIFR0 = (1<<TOV0); // Clear TOV flag
    PORTB ^= (1<<2); // Toggle PORTB.2
    TCNT0 = 0x86; // Reset TCNT0 count back to 134
}

```

13. Task #4 Simulation Pictures

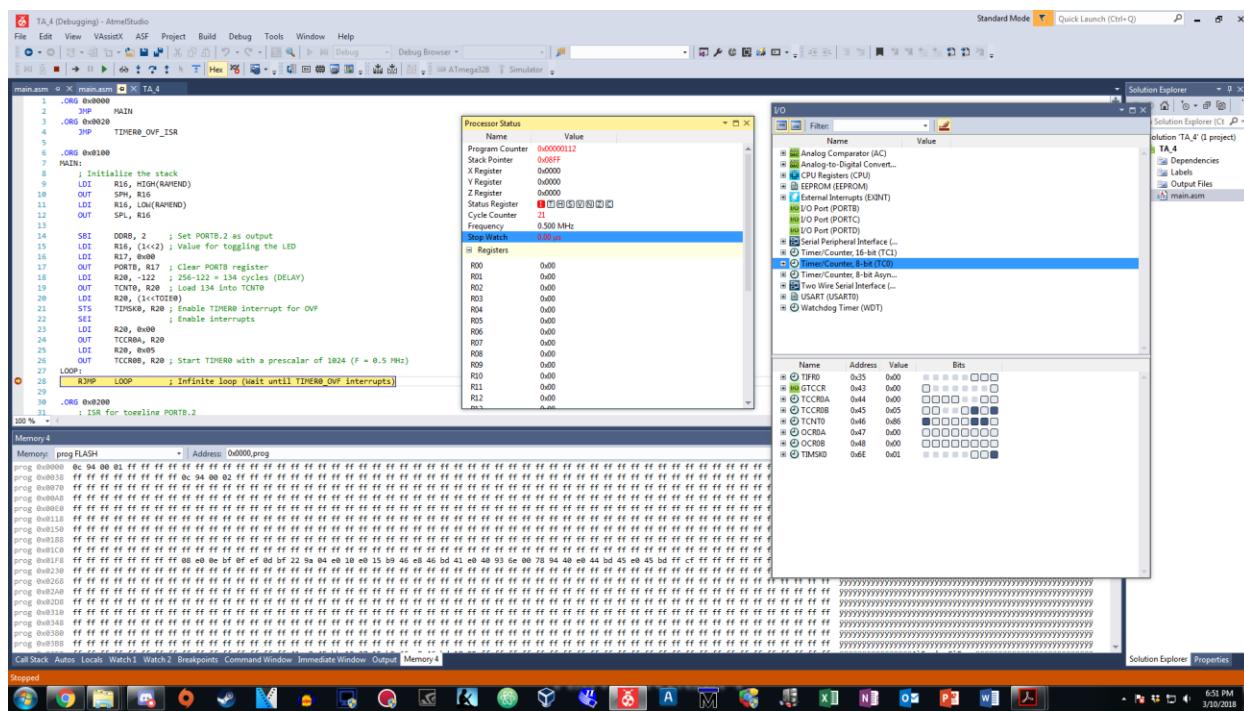


Figure 1: Before the interrupt, Timer0 interrupt registers are set (ASM)

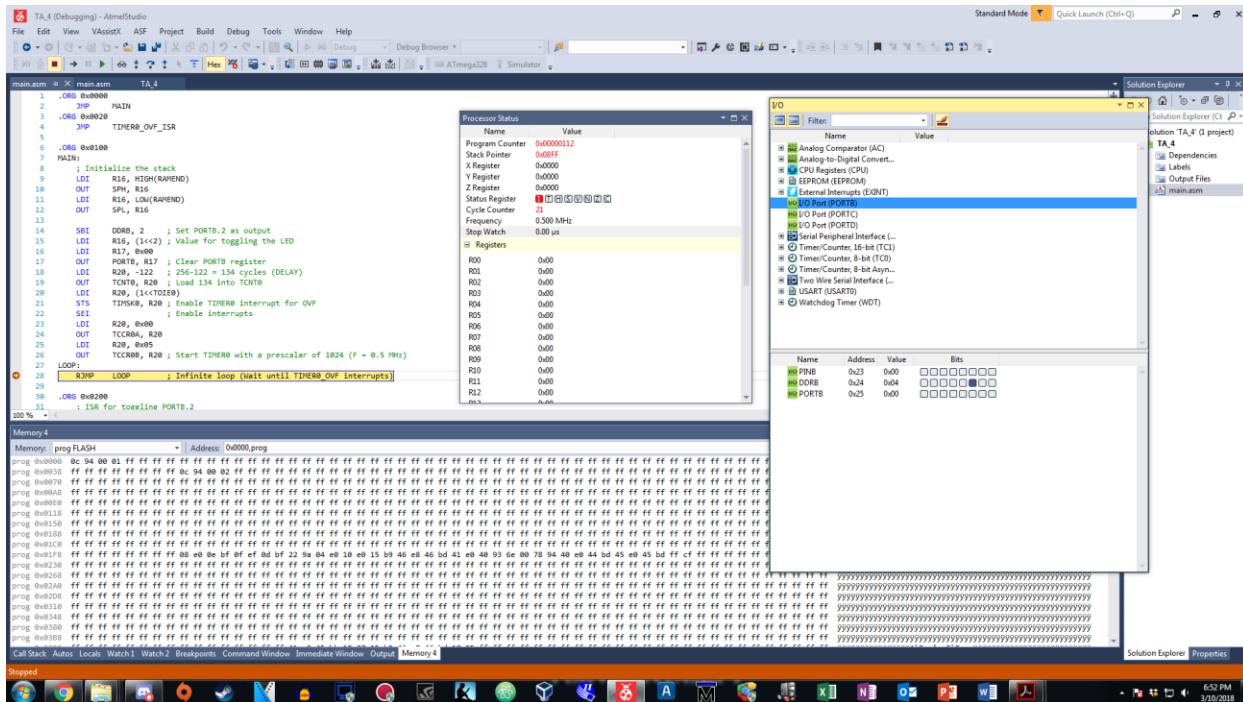


Figure 2: Before the interrupt, stopwatch is set and PB.2 is cleared (ASM)

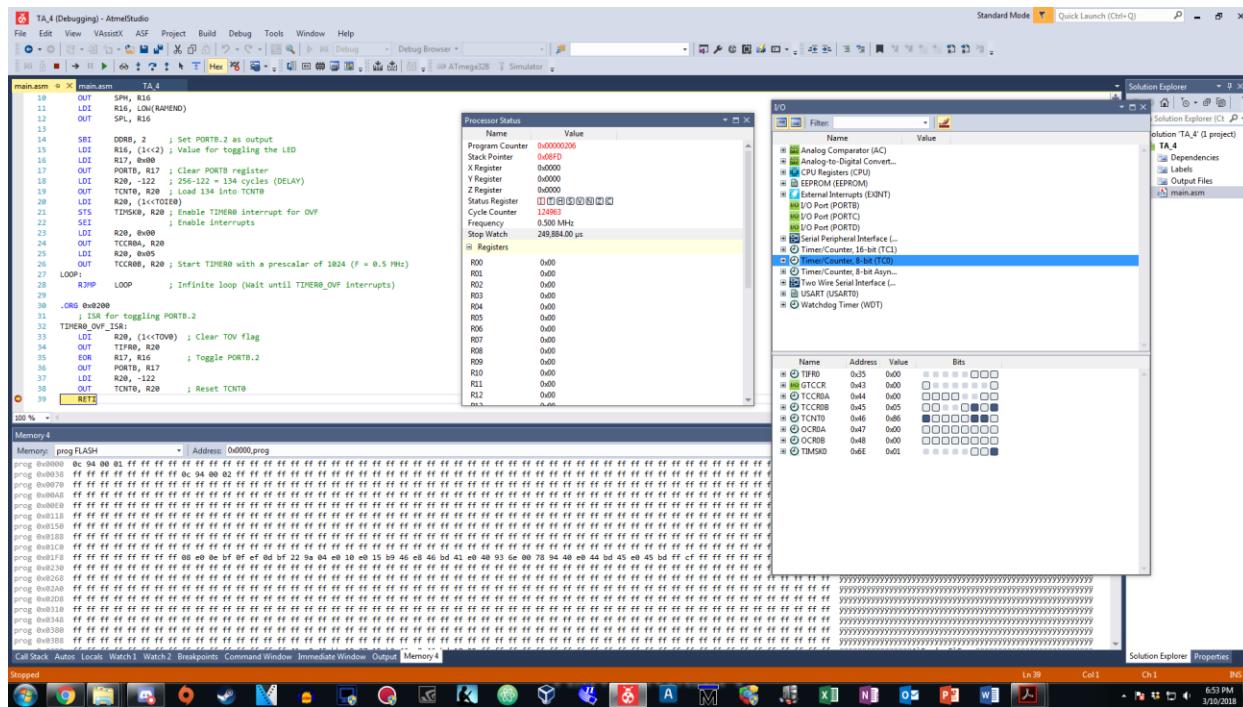


Figure 3: Interrupt occurred and serviced in the ISR by clearing the flag (ASM)

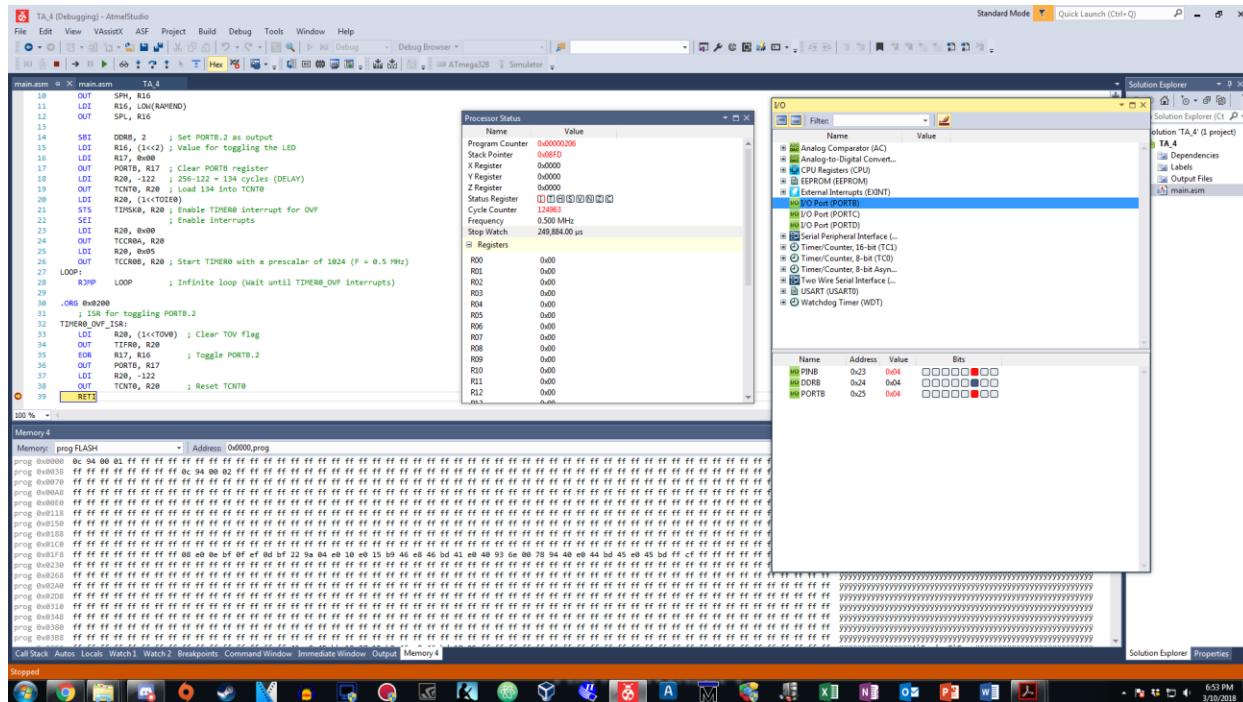


Figure 4: PB.2 is toggled in the ISR after 0.25 second of delay (ASM)

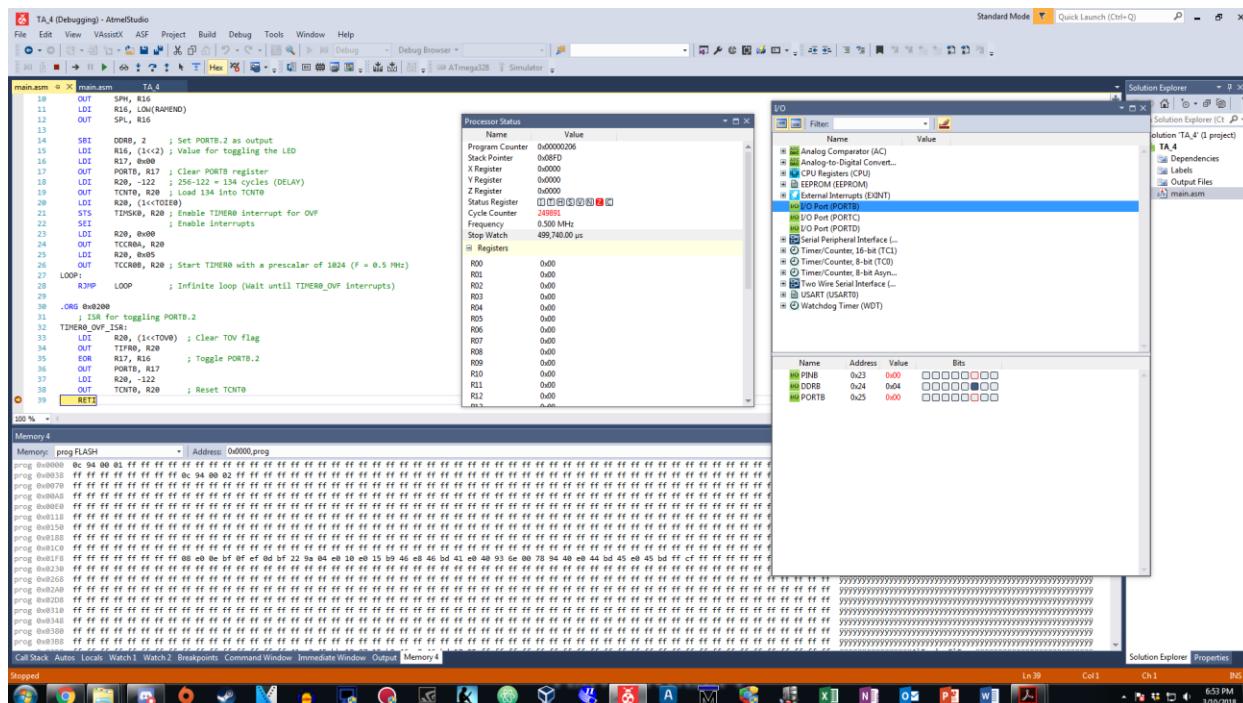


Figure 5: PB.2 is toggled in the ISR again after 0.25 second of delay (ASM)

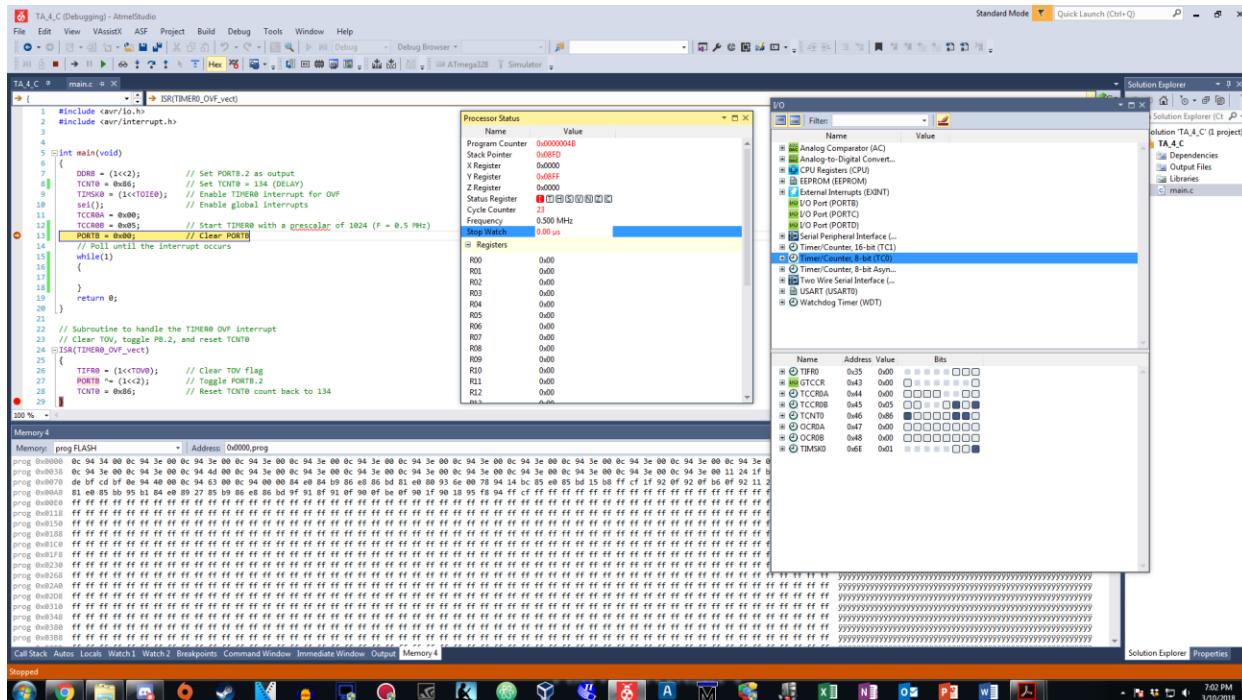


Figure 6: Before the interrupt, Timer0 interrupt registers are set (C)

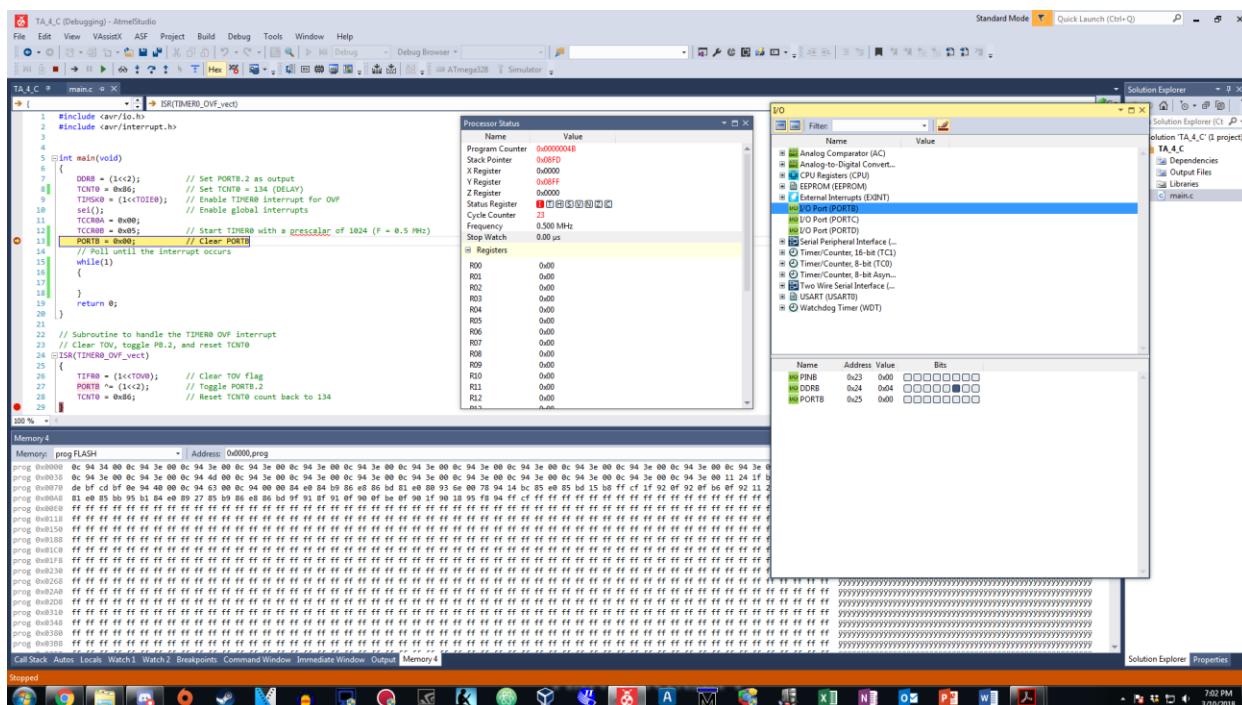


Figure 7: Before the interrupt, stopwatch is set and PB.2 is cleared (C)

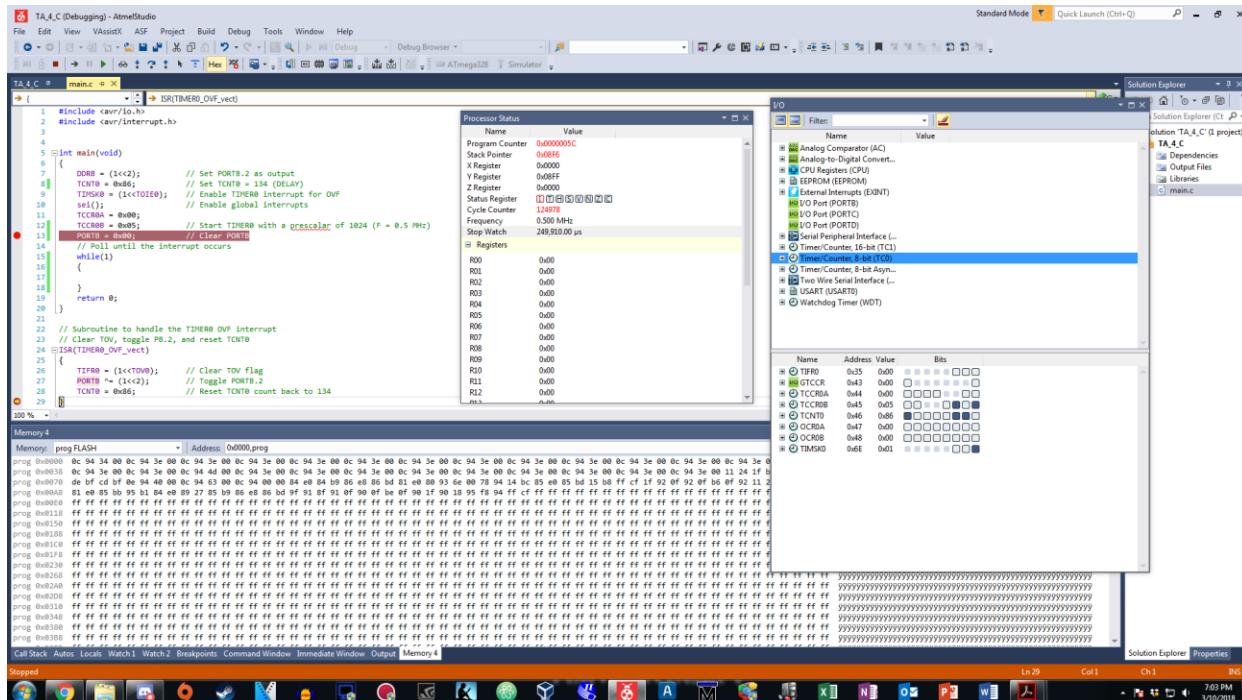


Figure 8: Interrupt occurred and serviced in the ISR by clearing the flag (C)

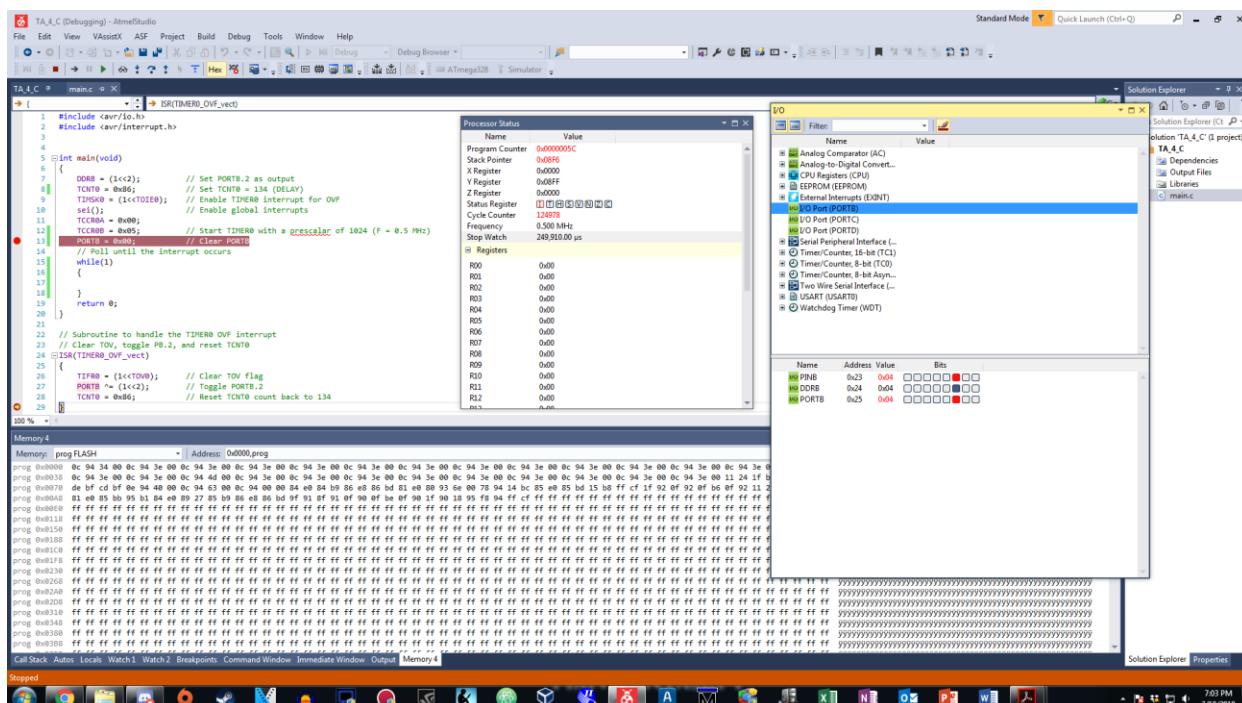


Figure 9: PB.2 is toggled in the ISR after 0.25 second of delay (C)

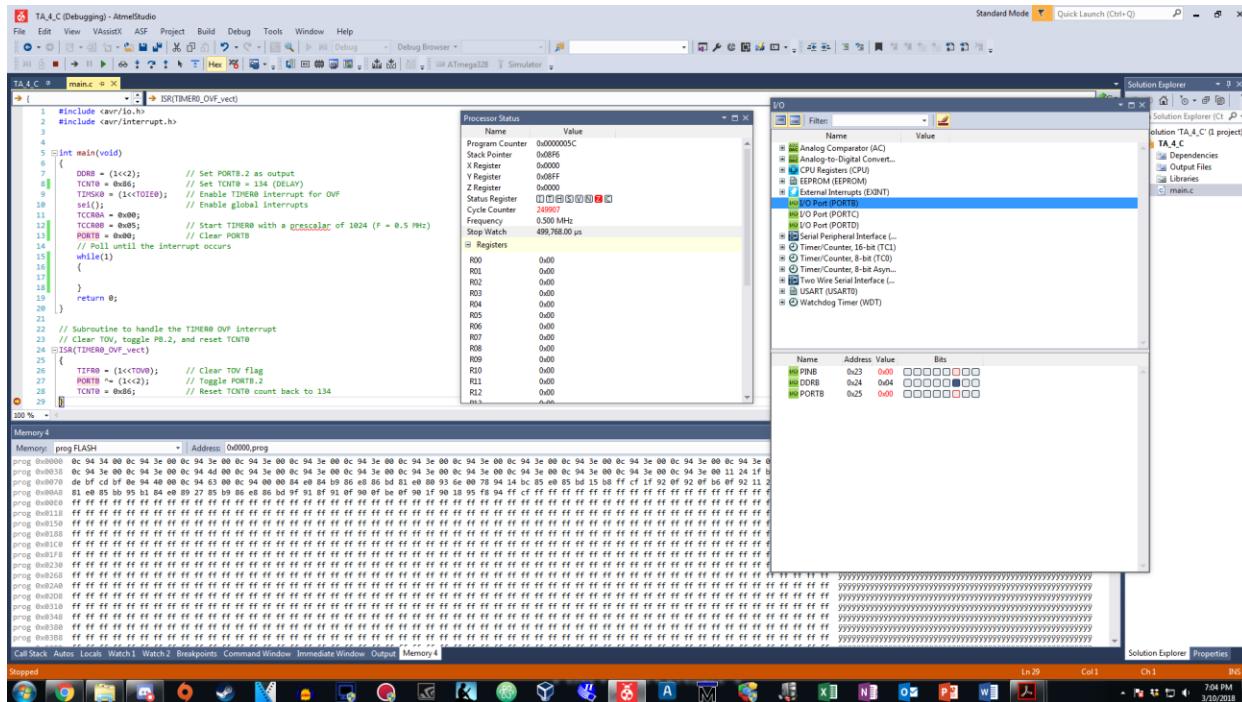


Figure 5: PB.2 is toggled in the ISR again after 0.25 second of delay (C)

14. Task #5 Assembly Code

```
.ORG 0x0000
    JMP MAIN
.ORG 0x0006
    JMP ISR_INT0
.ORG 0x0100
MAIN:
    ; Initialize stack
    LDI      R16, HIGH(RAMEND)
    OUT     SPH, R16
    LDI      R16, LOW(RAMEND)
    OUT     SPL, R16
    ; PORT initialization
    CBI      DDRD, 2           ; Set PORTD.2 as input
    SBI      PORTD, 2          ; Pull the resistor of PORTD.2
    SBI      DDRB, 2           ; Set PORTB.2 as output
    CBI      PORTB, 2          ; Clear PORTB.2
    ; Interrupt register initialization
    LDI      R16, (1<<INT0)
    OUT     EIMSK, R16         ; Enable INT0 interrupt
    LDI      R16, 0x02
    STS     EICRA, R16         ; INT0 interrupt occurs on the falling edge
    SEI
LOOP:
    RJMP    LOOP              ; Infinite loop until INT0 interrupt occurs

.ORG 0x0200
ISR_INT0:
    SBI      PORTB, 2          ; Light up the LED
HOLD:
```

```

SBIS  PIND, 2           ; Poll until the user lets go of the switch
RJMP  HOLD
CALL  DELAY             ; Delay for 1 second
CBI   PORTB, 2          ; Turn off the LED
LDI   R21, (1<<INTF0)
STS   EIFR, R21         ; Clear interrupt flag
RETI

; Delay subroutine (F = 0.5 MHz) [Delay for 1 second]
DELAY:
LDI   R20, HIGH(-62500)
STS   TCNT1H, R20
LDI   R20, LOW(-62500)
STS   TCNT1L, R20
; Set timer control register
LDI   R20, 0x00
STS   TCCR1A, R20
LDI   R20, 0x02
STS   TCCR1B, R20      ; Start the timer with a prescalar of 8
DLOOP:
IN    R20, TIFR1        ; Check for the TOV flag
SBRS R20, TOV0
RJMP DLOOP
; Stop timer and clear TOV flag
LDI   R20, 0x0
STS   TCCR1B, R20
LDI   R20, (1<<TOV0)
OUT  TIFR1, R20
RET

```

15. Task 5 C Code

```

#include <avr/io.h>
#include <avr/interrupt.h>

void T1Delay();

int main(void)
{
    // Port initialization
    DDRD = 0x00;           // Set PORTD as inputs
    PORTD |= (1<<2);     // Pull up the resistor of PORTD.2
    DDRB |= (1<<2);     // Set PORTB.2 as output
    PORTB = 0x00;          // Clear PORTB
    // Interrupt register initialization
    EIMSK |= (1<<INT0); // Enable INT0 interrupt
    EICRA |= 0x02;         // INT0 interrupt occurs on the falling edge
    sei();                 // Enable global interrupt
    while (1)
    {
    }
    return 0;
}

ISR(INT0_vect)
{

```

```

PORTB |= (1<<2);      // Light up the LED
while((PIND&(1<<2))&0x00);
T1Delay();
PORTB = 0x00;          // Turn off LED
EIFR = (1<<INTFO);   // Clear interrupt flag
}

// Delay subroutine for Timer1 [Delay for 1 second]
// F = 0.5 MHz
void T1Delay()
{
    // Set the TCNT1 = 65536-62500 = 3036 [0xBDC]
    TCNT1H = 0x0B;
    TCNT1L = 0xDC;
    // Start Timer1 with prescalar 8
    TCCR1A = 0x00;
    TCCR1B = 0x02;
    while((TIFR1&(1<<TOV1)) == 0x00); // Check TOV flag
    TCCR1B = 0x00;                      // Stop Timer1
    TIFR1 |= (1<<TOV1);                // Reset TOV flag in TIFR1 register
}

```

16. Task #5 Simulation Pictures

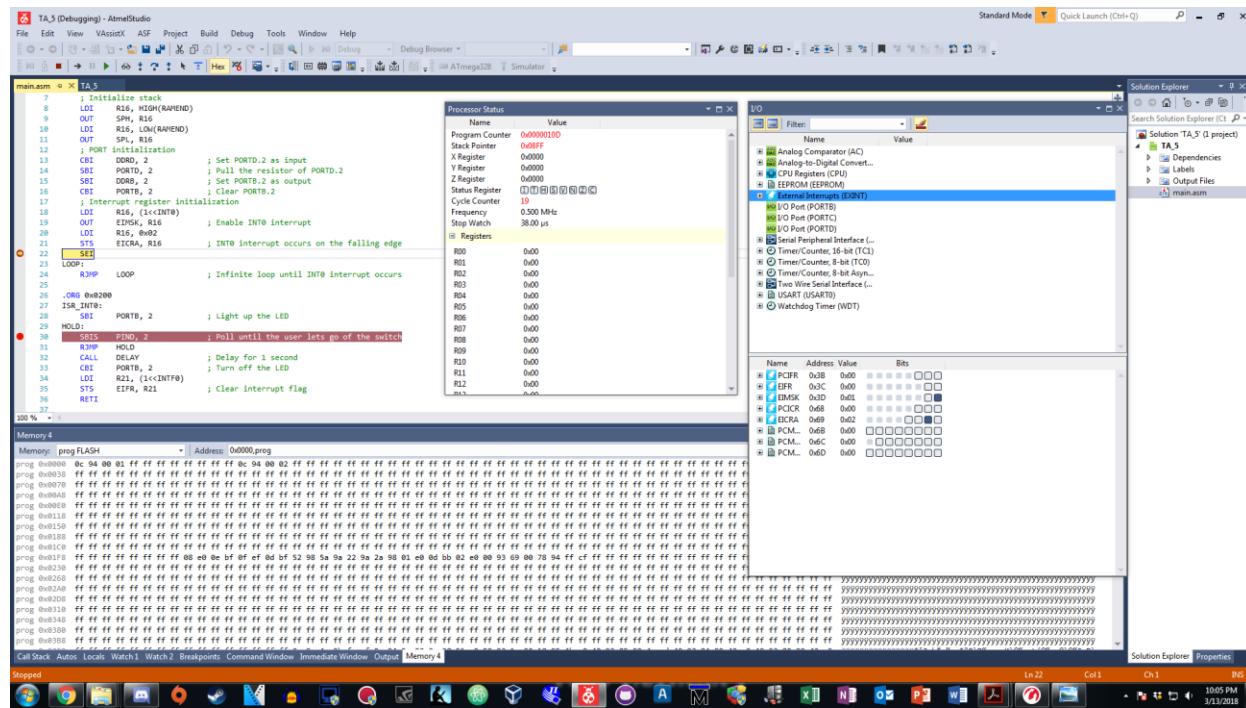


Figure 1: Appropriate interrupt registers are set: INT0 interrupt on falling edge (ASM)

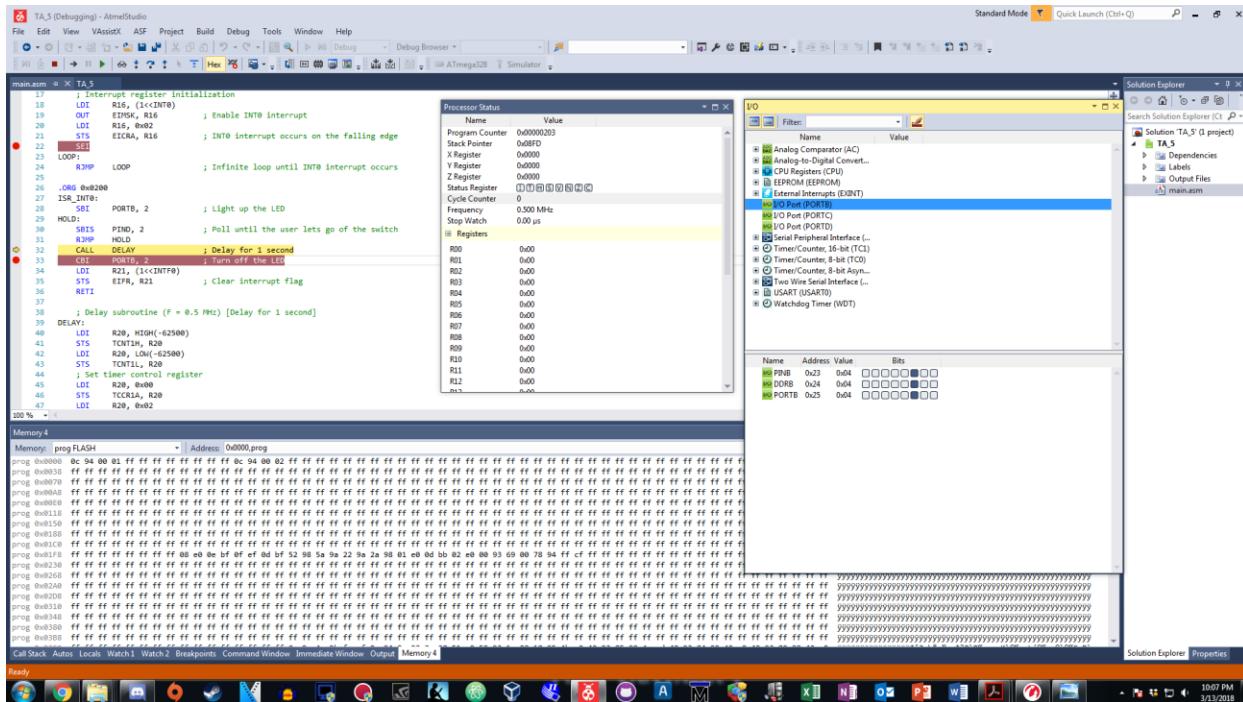


Figure 2: LED turns on at PORTB.2 and stopwatch set to measure the 1 second delay (ASM)

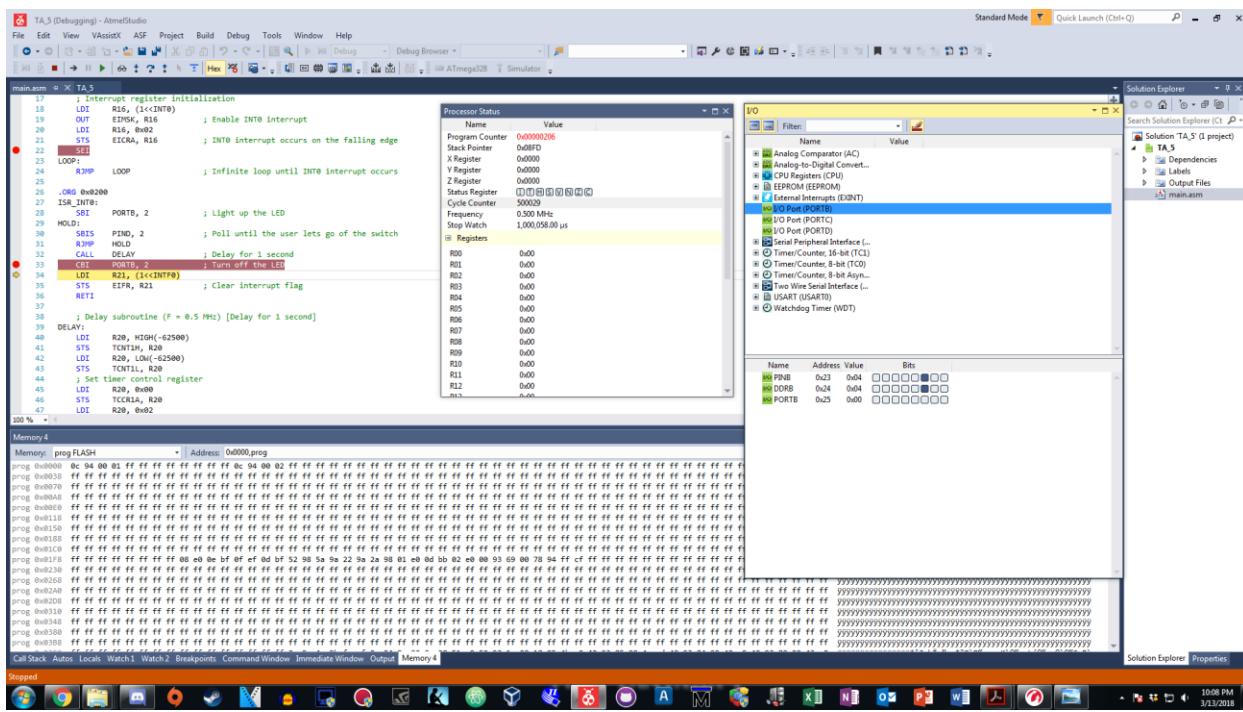


Figure 3: After 1 second delay, LED turns off at PORTB.2 (ASM)

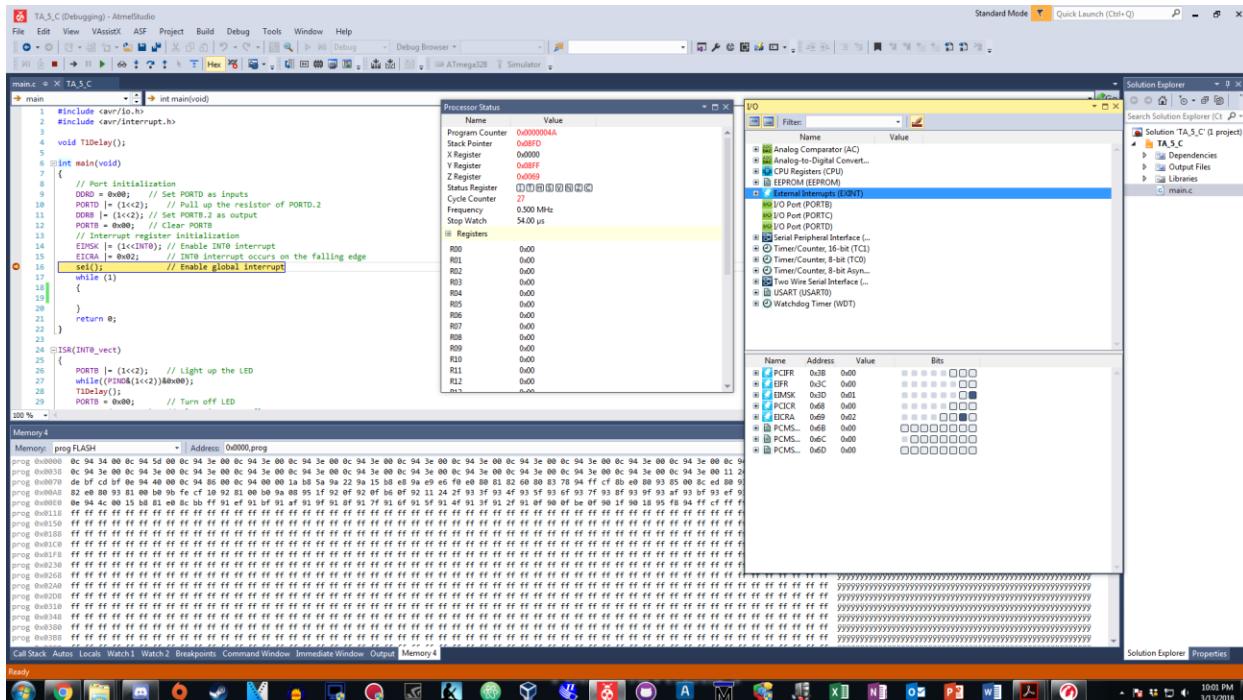


Figure 4: Appropriate interrupt registers are set: INTO interrupt on falling edge (C)

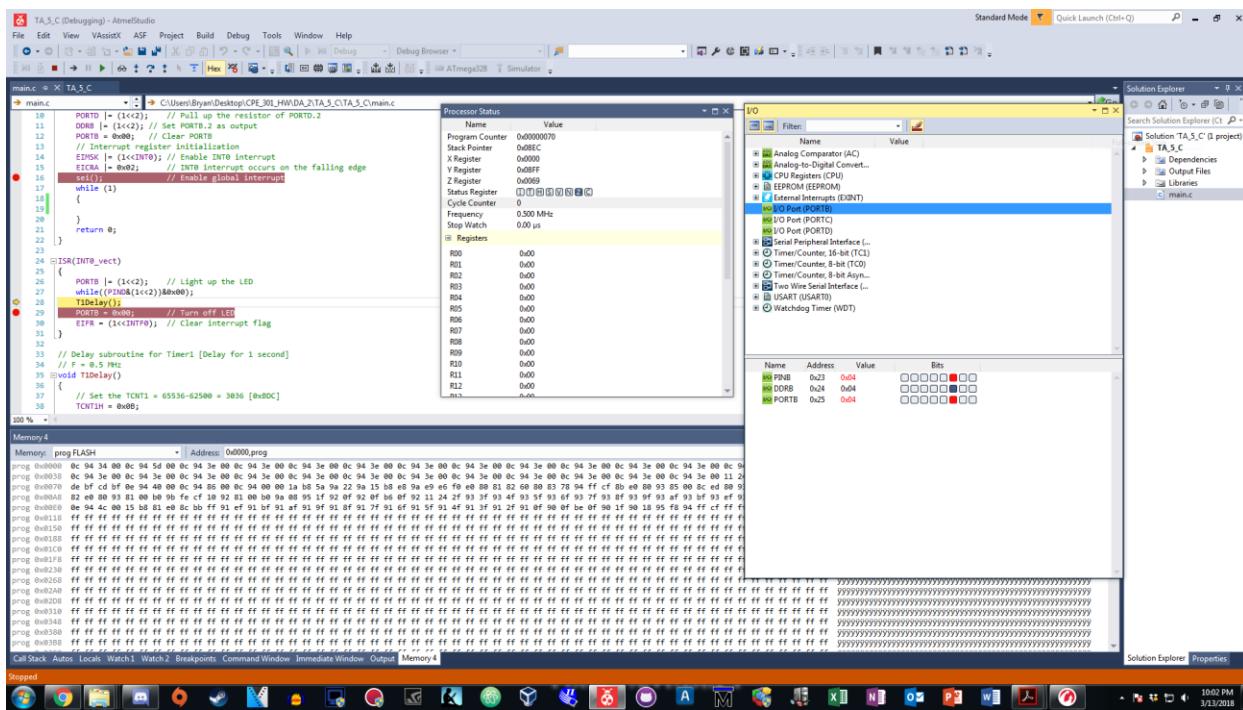


Figure 5: LED turns on at PORTB.2 and stopwatch set to measure the 1 second delay (C)

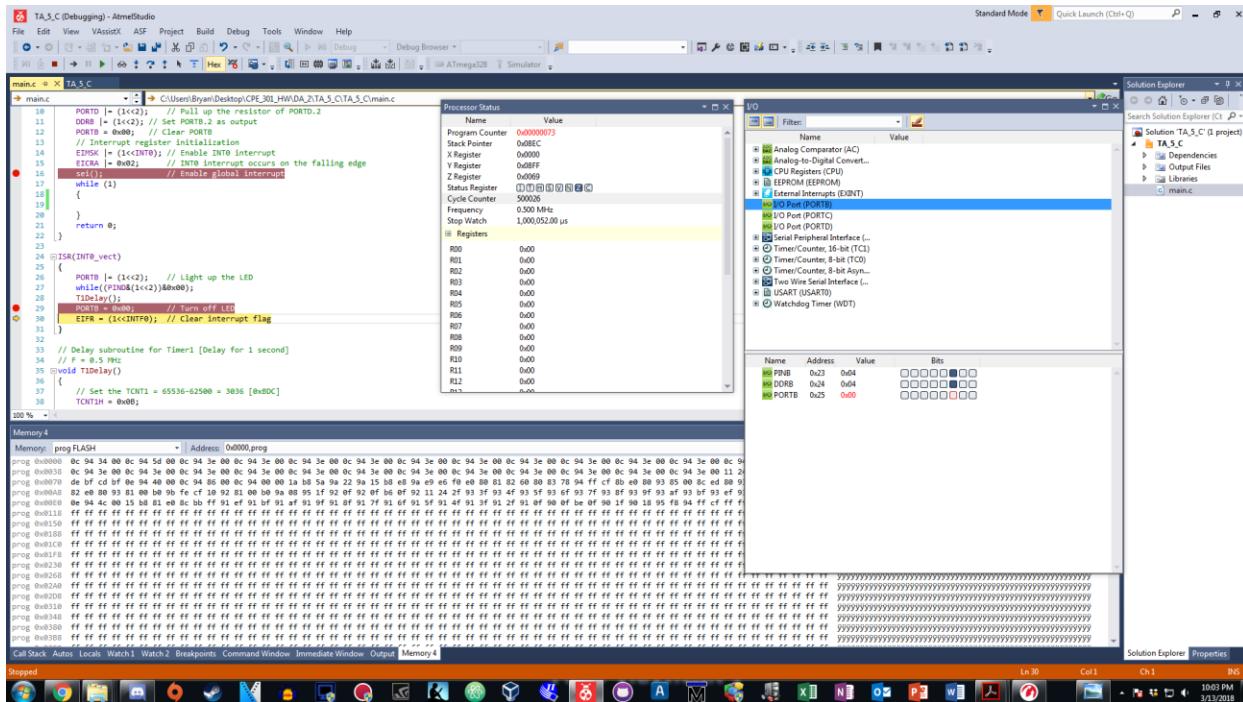


Figure 6: After 1 second delay, LED turns off at PORTB.2 (C)

17. YouTube Links

Task #1: <https://youtu.be/kM3ICCPzmo>

Task #2: <https://youtu.be/AVlxueKxSqc>

Student Academic Misconduct Policy

<http://studentconduct.unlv.edu/misconduct/policy.html>

"This assignment submission is my own, original work".

Bryan Takemoto