CPE301 – SPRING 2019

Design Assignment 2B

Student Name: Clayton Higbee

Student #: 5001189864

Student Email: higbee@unlv.nevada.edu

Primary Github address: <https://github.com/claytonjhigbee/CODSWORTH_MAIN>

Directory:

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

Components used:

Atmega328P Xplained Mini Board

Female to Male Wires (For logic analyzer Connections)

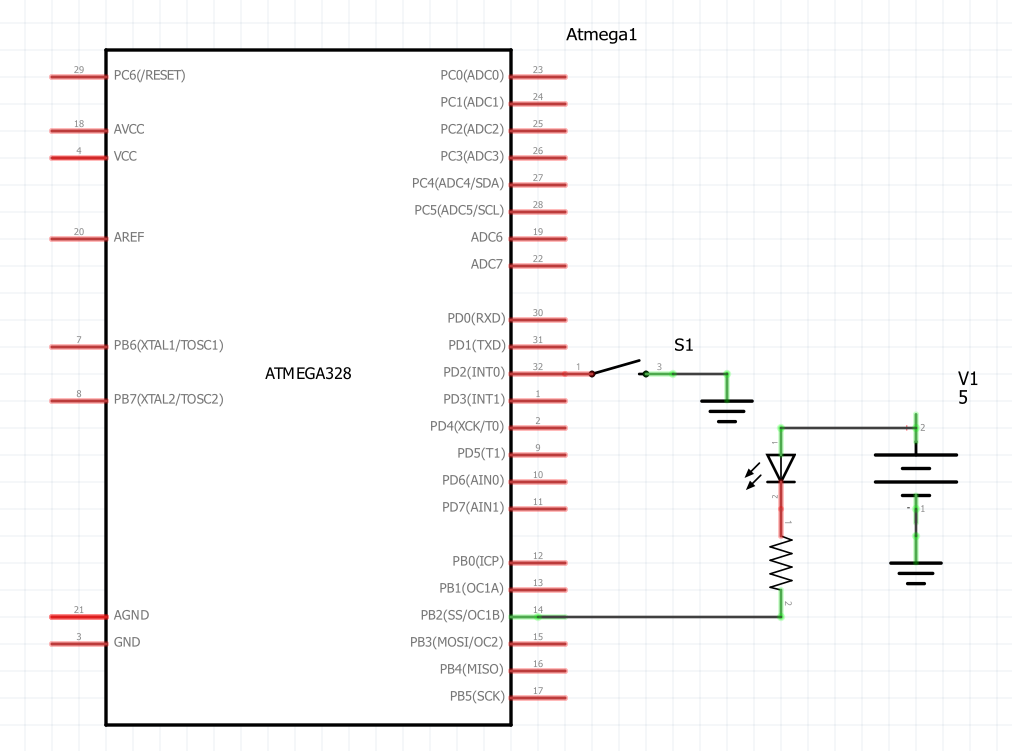
Logic Analyzer

Multi-Function Development Board Shield

Block diagram with pins used in the Atmega328P

PD2 – T0 – Interrupt Pin

PB2 – LED Pin



1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A**

Assembly Code:

;

; InterruptEnabledPushButton.asm

;

; Created: 3/6/2019 9:22:47 PM

; Author : clayt

;

; Task 1: Connect a switch to PORTD.2 (active high - turn on the pull up transistor) to poll for an event to turn on the led at PORTB.2 for 1.250 sec after the event.

.ORG 0; LOCATION FOR RESET

JMP START

.ORG 0x02 ; Location for external interrupt stack location

JMP EXO\_ISR

START:

; Initialize Stackpointer

LDI R20, HIGH(RAMEND)

OUT SPH, R20 ; LOAD HIGH BYTE OF STACKPOINTER TO INITIALIZE

LDI R20, LOW(RAMEND)

OUT SPL, R20

;Initialize I/O Pins

LDI R22, 0 ; Load immediate 0 into R22

LDI R16, 0xFF ; Load immediate 255 into R16

OUT DDRB, R16 ; Set DDRB as output

OUT DDRD, R22 ; Set DDRD as input

SBI PORTD, 2 ; Set PortD.2 Pull up resistors for input for interrupt

OUT PORTB, R16 ; Set Port B and related Pins HIGH to set LEDs to LOW (Reverse Logic)

;Initialize Interrupt

LDI R20, 0x2 ; CONFIGURE INTERRUPT TO OCCUR INT0 FOR FALLING EDGE TRIGGER

STS EICRA, R20 ; LOAD CONFIGURE INTO REGISTER

LDI R20, 1<< INT0 ; ENABLE EXTERNAL INTERRUPT INT0(PD2)

OUT EIMSK, R20 ; LOAD ENABLE INTO REGISTER

SEI ; GLOBAL INTERRUPT ENABLE COMMAND

HERE: JMP HERE ; LOOP HERE UNTIL THE INTERRUPT OCCURS

; INTERRUPT SUBROUTINE ---------------------------------------

EXO\_ISR:

LDI R16, 0xFF

LDI R19, 0b11111011 ; Load immediate into R19 to Set PortB2 LED ON

OUT PORTB, R19 ; Set LED Port B to Low (REVERSE LOGIC)

RCALL DELAY\_1250ms

OUT PORTB, R16 ; Set LED Port B to High (REVERSE LOGIC) to turn LED off

RETI ; RETURN FROM INTERRUPT

DELAY\_1250ms:

RCALL DELAY\_1000ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RET

DELAY\_1000ms:

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RCALL DELAY\_100ms

RET

DELAY\_100ms:

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RCALL DELAY\_10ms

RET

DELAY\_10ms:

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RCALL DELAY\_1ms

RET

DELAY\_1ms:

PUSH R16

PUSH R18

LDI R16, 172

LDI R18, 6

DELAY\_1A:

DELAY\_1B:

NOP

NOP

NOP

NOP

NOP

NOP

NOP

NOP

DEC R16

BRNE DELAY\_1B

DEC R18

BRNE DELAY\_1B

POP R16

POP R18

RET

C Code:

/\*

\* InterruptEnabledPushButtonCCode.c

\*

\* Created: 3/6/2019 9:23:38 PM

\* Author : clayt

\*/

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

int main()

{

DDRB = 0xFF; // Set PortB as an output

PORTB = 0xFF; // Set all PinB HIGH (REVERSE LOGIC LOW), turn all LEDs off

PORTD = 1<<2; // Set pull up resistor in PortD pin 2

EICRA = 0x2; // Set INT0 to be falling edge triggered

EIMSK = (1<<INT0); // Enable External Interrupt 0

sei(); // Global Enable for interrupts, notice that the command here is cap sensitive

while (1) // Wait here and do nothing until the interrupt occurs

{

}

}

ISR (INT0\_vect) // ISR for External Interrupt 0

{

PORTB = 0b11111011; // Set PinB 2 Low (REVERSE LOGIC HIGH), Turn PB2 LED on

*\_delay\_ms*(1250);

PORTB = 0xFF; // Set all PinB HIGH (REVERSE LOGIC LOW), Turn all LEDs off

}

1. **DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A**

There was no modified code between tasks, but this assignment did rely on code from DA\_2A Part 2 for implementing the LED and a delay. Those codes are below and DA\_2A has been included in this assignment’s repository for reference:

Assembly Code Delay:

DELAY\_1ms:

PUSH R16

PUSH R18

LDI R16, 172

LDI R18, 6

DELAY\_1A:

DELAY\_1B:

NOP

NOP

NOP

NOP

NOP

NOP

NOP

NOP

DEC R16

BRNE DELAY\_1B

DEC R18

BRNE DELAY\_1B

POP R16

POP R18

RET

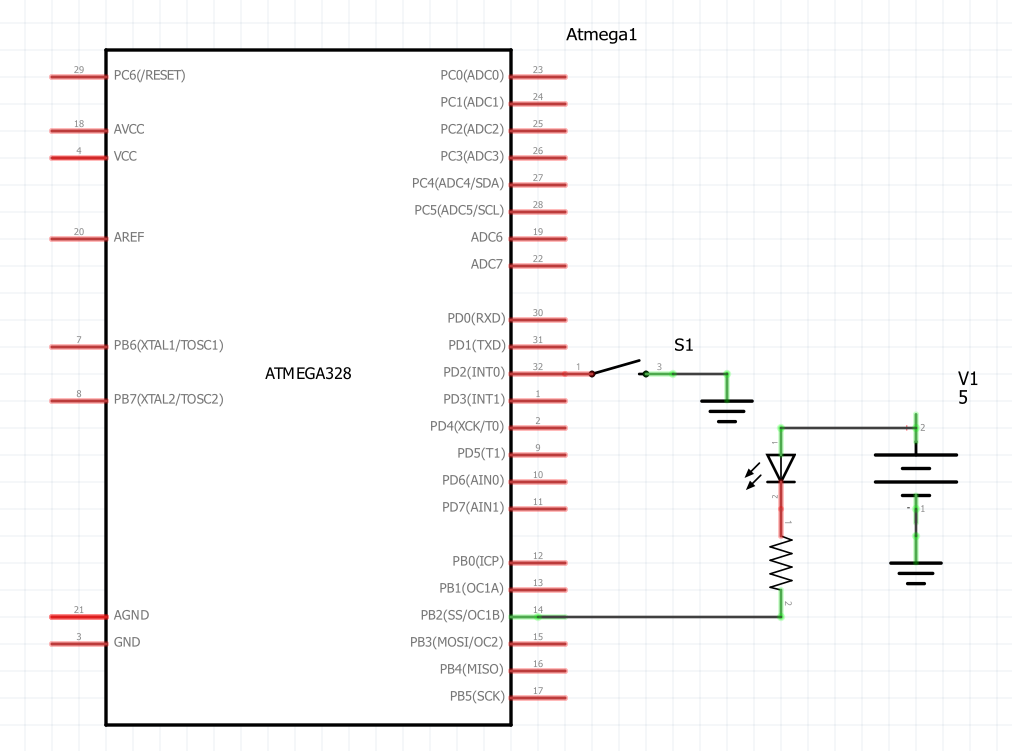
C Code Delay and LED Commands:

PORTB = 0b11111011; // Set PinB 2 Low (REVERSE LOGIC HIGH), Turn PB2 LED on

*\_delay\_ms*(1250);

PORTB = 0xFF; // Set all PinB HIGH (REVERSE LOGIC LOW), Turn all LEDs off

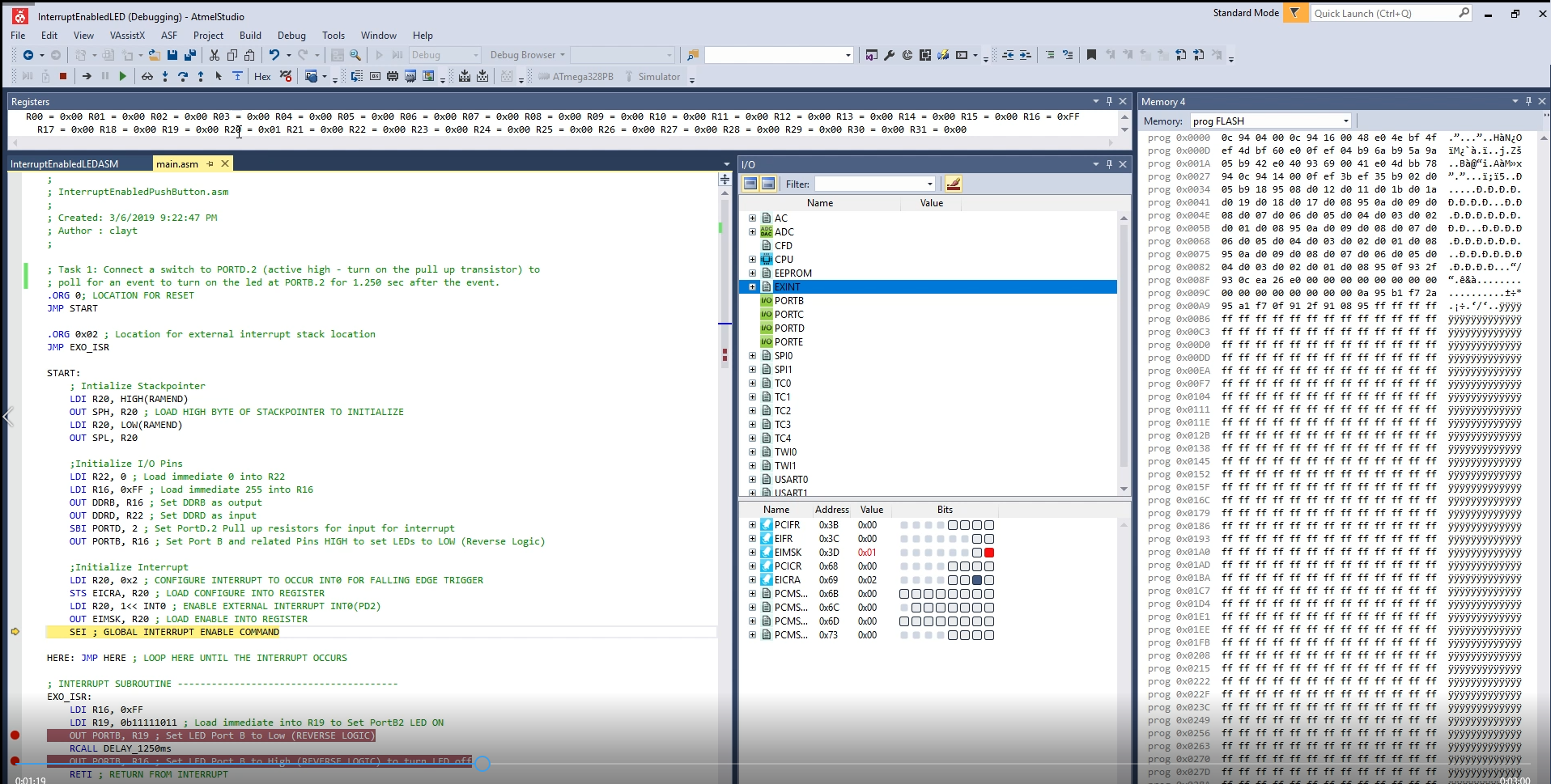
1. **SCHEMATICS**



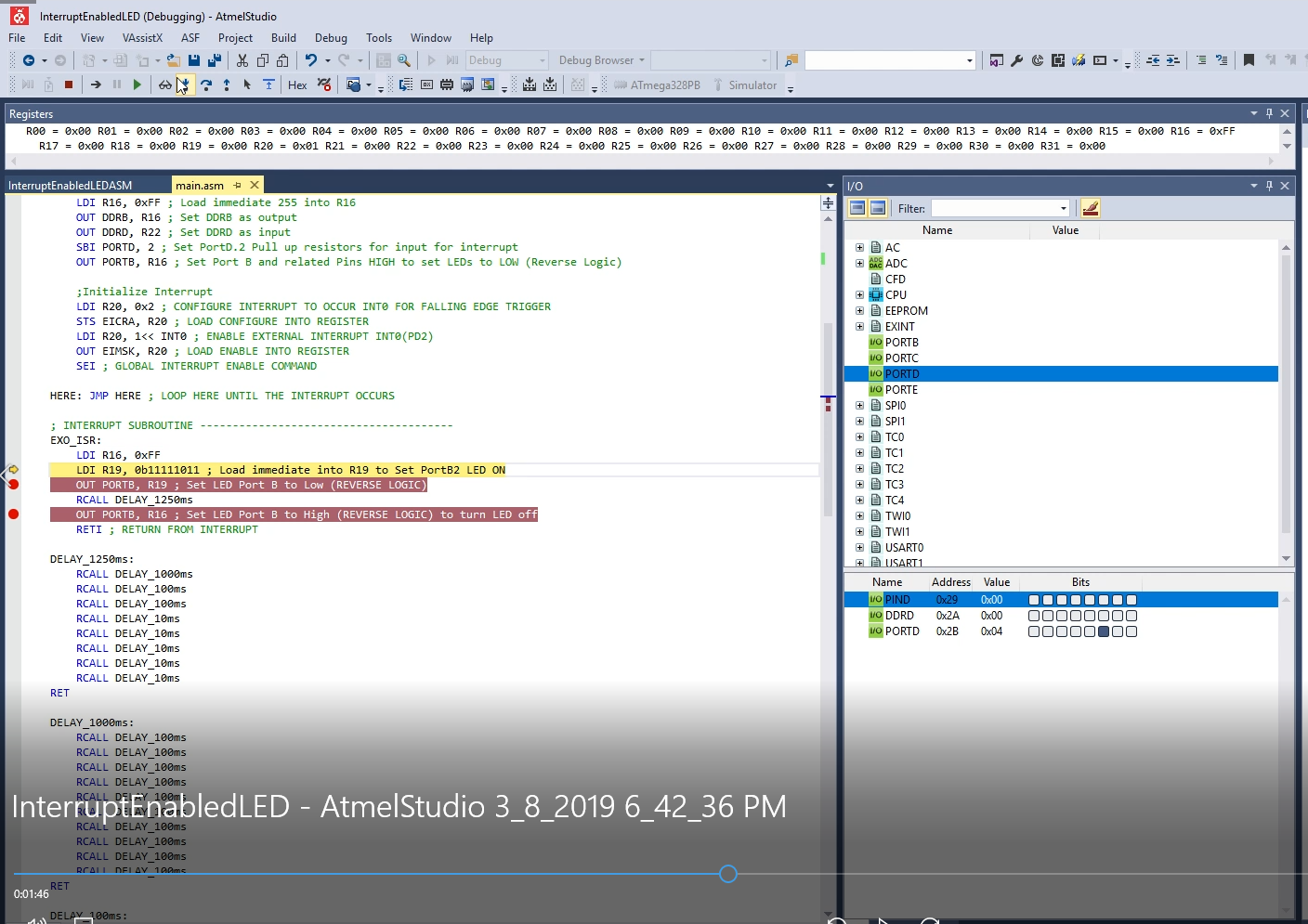
1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

Assembly

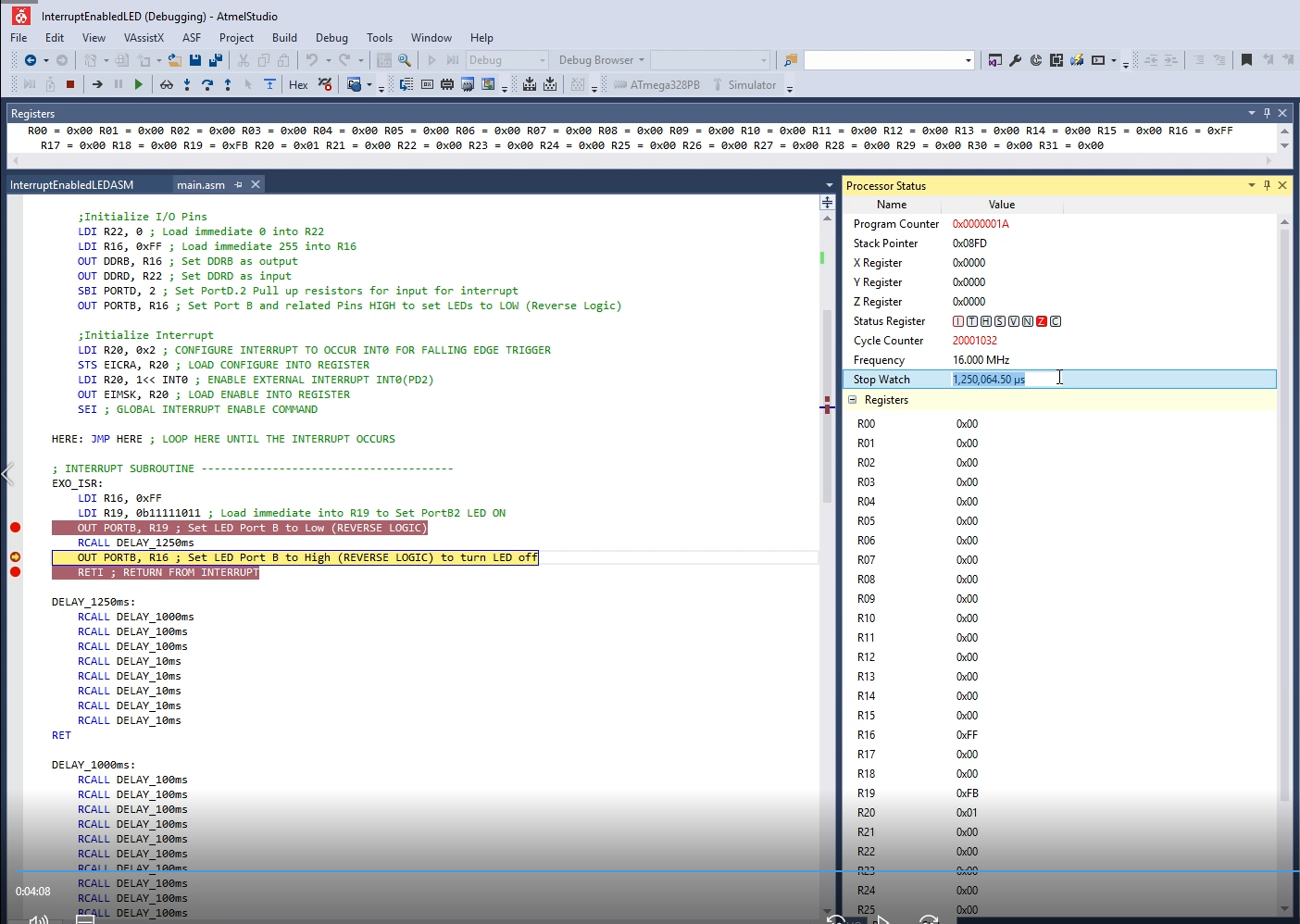
Initializing Interrupt in Registers



Falling Edge Triggers the Interrupt:

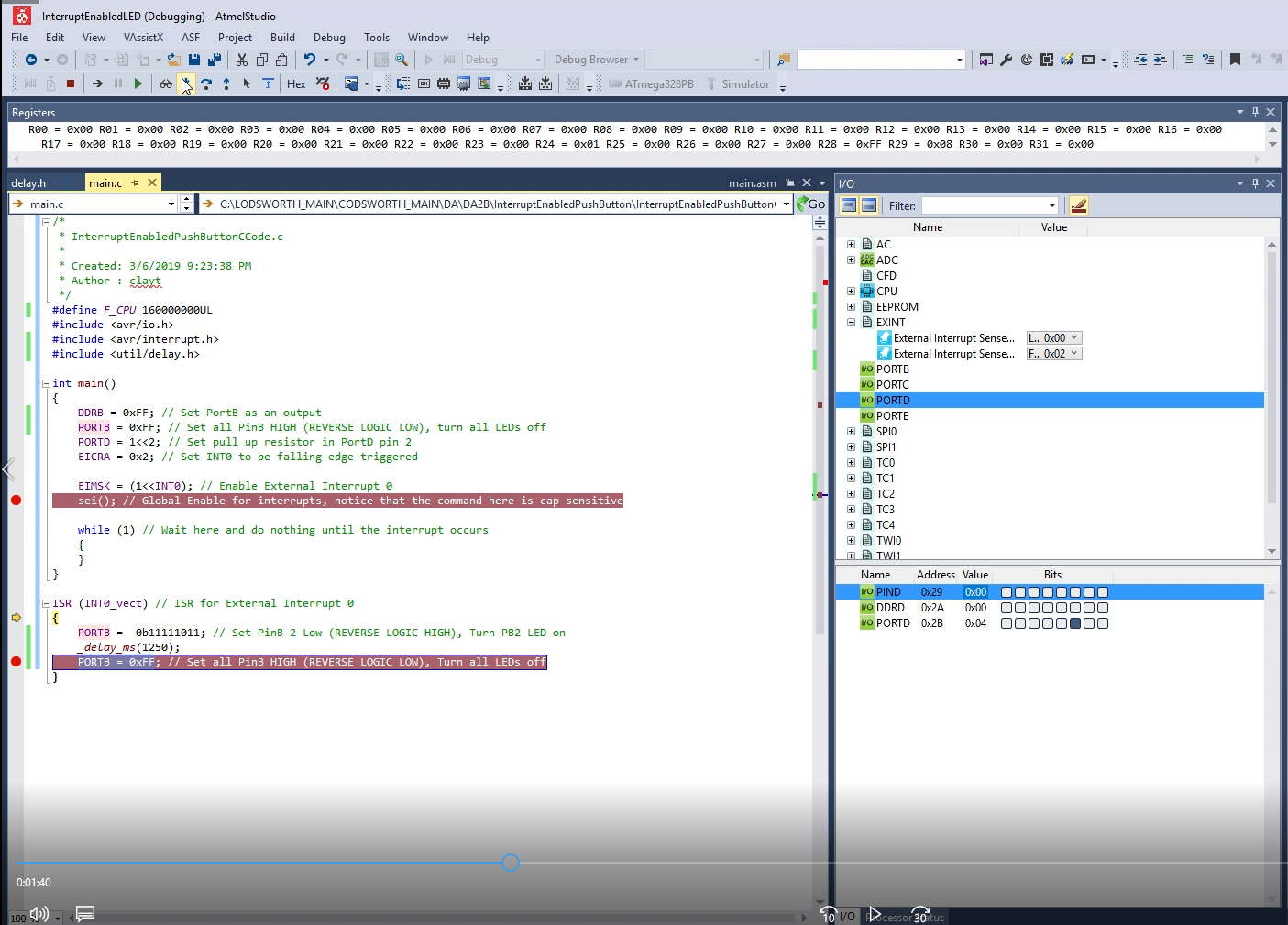


Final Delay Time:

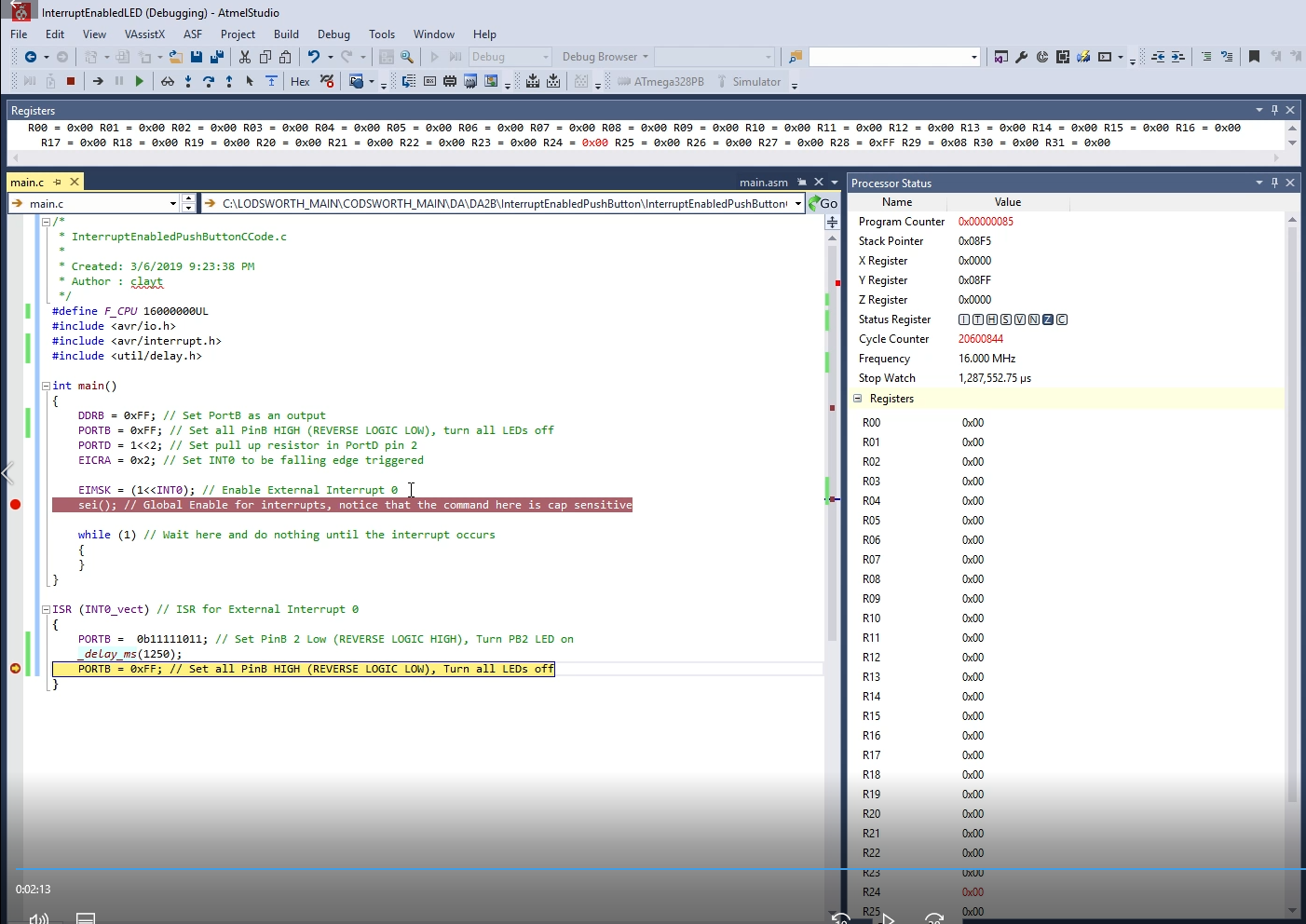


C Code

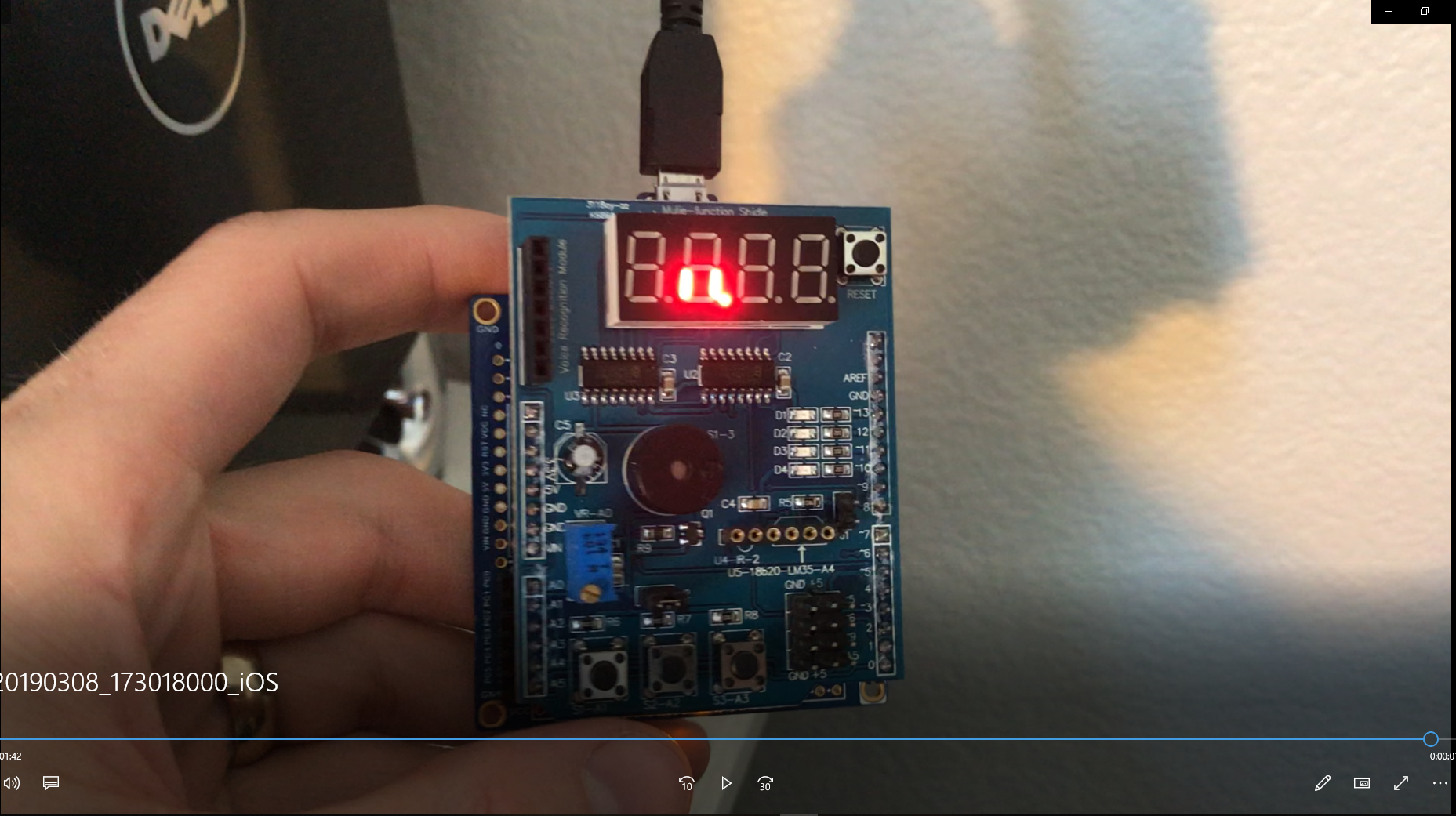
Interrupt Initializes entry into subroutine:



Final Stop watch time for delay, has additional 40us from attempting to catch interrupt



1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**



Board setup is same for Assembly and C Code

1. **VIDEO LINKS OF EACH DEMO**

<https://youtu.be/1V32edh0Z_8>

1. **GITHUB LINK OF THIS DA**

**Student Academic Misconduct Policy**

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

“This assignment submission is my own, original work”.

Clayton Higbee