Due on 11/03, submit a single file as doc, docx, or pdf in Canvas.

Please follow the example code attached in the package.

1. (10)

Atmega128 and Pic24e have the reset interrupt at the program address 0x0. Write a function reset() that works for both chips to reset a program. Copy and paste your code in report.

#include <avr/io.h>

#include <avr/interrupt.h>

#include <stdbool.h>

#define soft\_Reset()

#define F\_CPU 4000000UL //4mhz

volatile bool \_\_ran\_interupt = 0;

void \_\_reset\_Interupt(void){

do {

wdt\_enable(WDTO\_15MS);

for(;;) {

}

} while(0)

PORTB ^= 0x00;

\_\_ran\_interupt = true;

}

int main(void) {

PORTB ^= 0xFF;

OCR1A = 100;

OCR1B = 10;

TCCR1A = 0;

TCCR18 |= (1 << WGM12) | (1 << CS12);

// enable timer compare interrupt

TIMSK1 |= (1 << OCIE1A);

sei(); // Turns interupts on

while(!\_\_ran\_interupt)

{

\_\_reset\_Interupt();

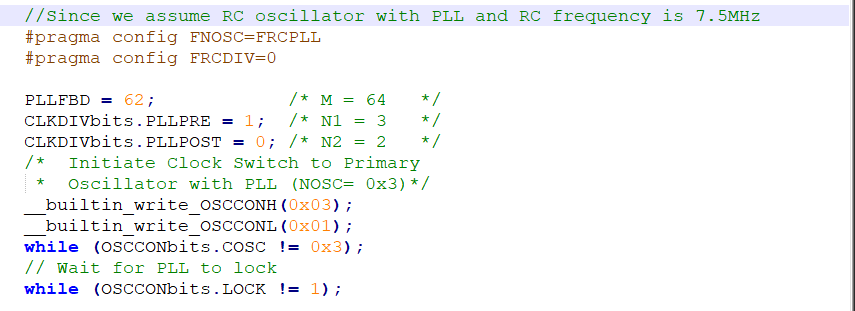
}

}

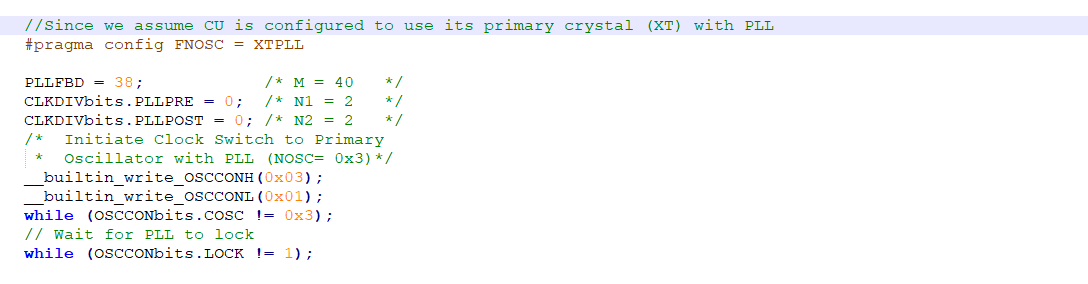
2. (10)

Program the clock for PIC24E. Refer to the example code included in "case10.clock.pic24e". Modify the code and copy and paste the code in your report (excluding the comments).

(a) Assume the MCU is configured to use its fast RC oscillator with PLL, and its RC's frequency is 7.5MHz. Program the clock to run at 40MHz. Show the code.



(b) Assume the MCU is configured to use its primary crystal (XT) with PLL, and the external crystal oscillator is 8MHz. Program the clock to run at 40MHz. Show the code.



3. (30)

Assume four LEDs are connected to Pins A4..7. Implement the following functions with bit operators only (|, &, ^, ~, <<, |=, &=, ^=).

You may refer to case12.led in example.code.

(1) Initialize the LEDs.

(2) Turn one selected LED on.

(3) Turn one selected LED off.

(4) Toggle one selected LED.

(5) Set a value to the LEDs.

(a) Show the code for Atmega128.

void initLeds() {

DDRA |= 0xF0;

}

void ledOn(uint8\_t sel) {

PORTA |= 1<<(sel+4);

}

void ledOff(uint8\_t sel) {

PORTA &= ~(1<<(sel+4));

}

void ledToggle(uint8\_t sel) {

PORTA ^= (1<<(sel+4));

}

void ledSet(uint8\_t val) {

PORTA = (PORTA & 0x0F) | (val<<4);

}

(b) Show the code for Pic24ep512gu810.

void initLeds() {

ANSELA &= 0xFF0F;

TRISA &= 0xFF0F;

}

void ledOn(uint8\_t sel) {

LATA |= 1<<(sel+4);

}

void ledOff(uint8\_t sel) {

LATA &= ~(1<<(sel+4));

}

void ledToggle(uint8\_t sel) {

LATA ^= (1<<(sel+4));

}

void ledSet(uint8\_t val) {

LATA = (LATA & 0xFF0F) | (val<<4);

}

4. (30)

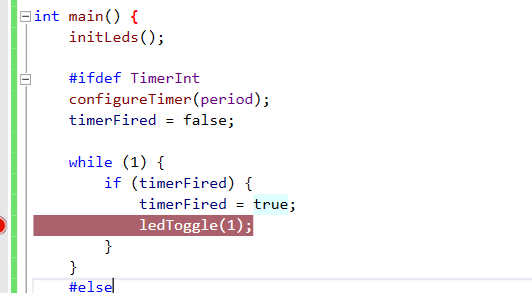
Make a program to blink/toggle a LED every second with the Atmega128 chip.

You may call the toggle function in Question 3 to toggle the led and do not need to show the toggle function again.

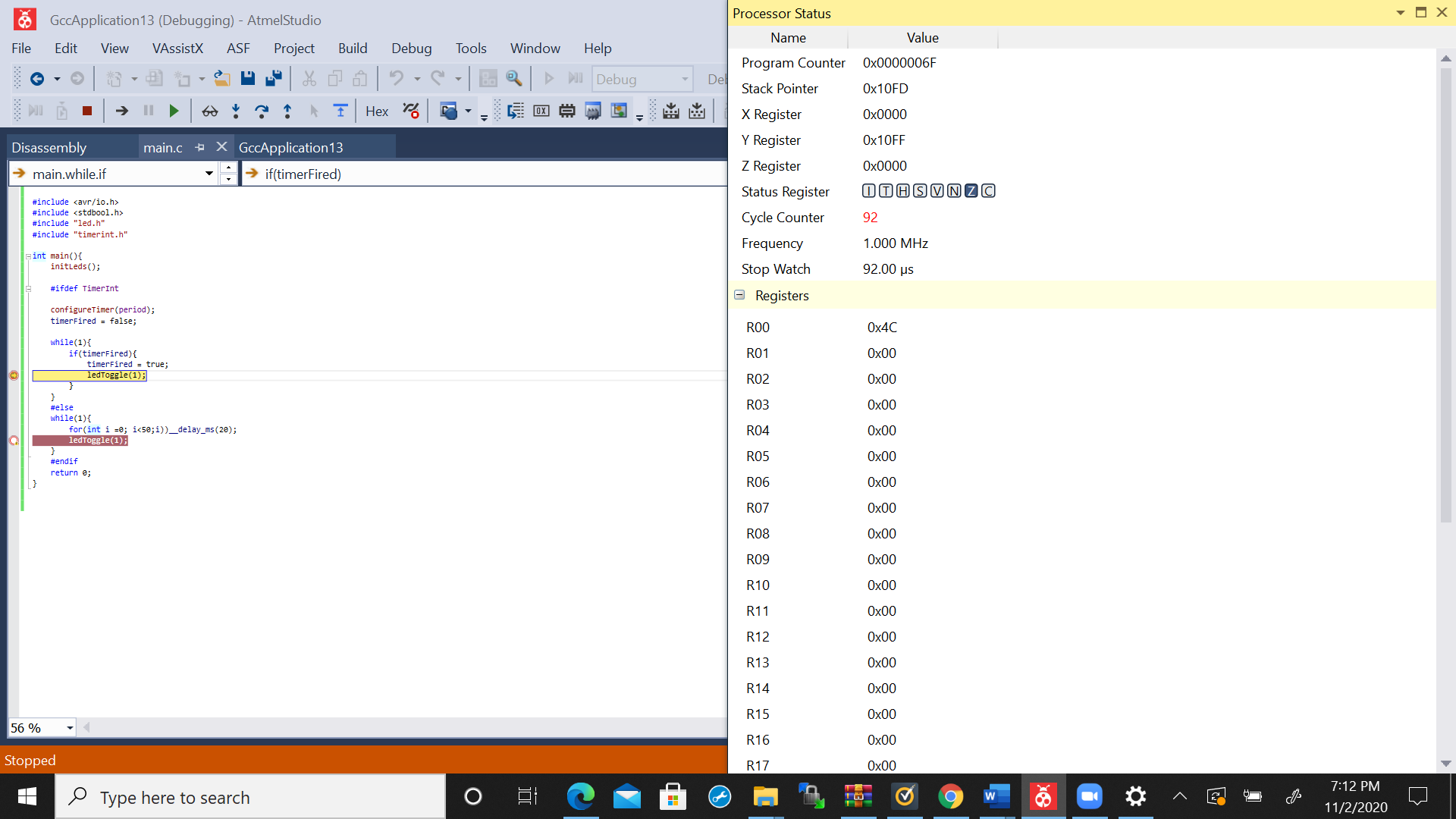
Hint: Because the delay function has a maximum delay limit, you will need an internal counter to accumulate delays to one second.

You may refer to case14.blink in example.code.

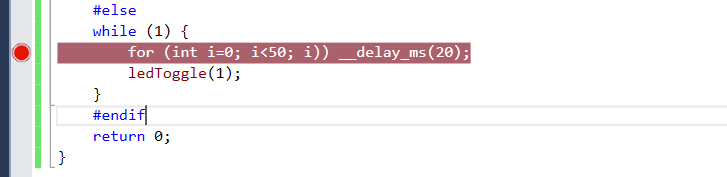
(a.1) Show the code to use the timer interrupt for implementation.



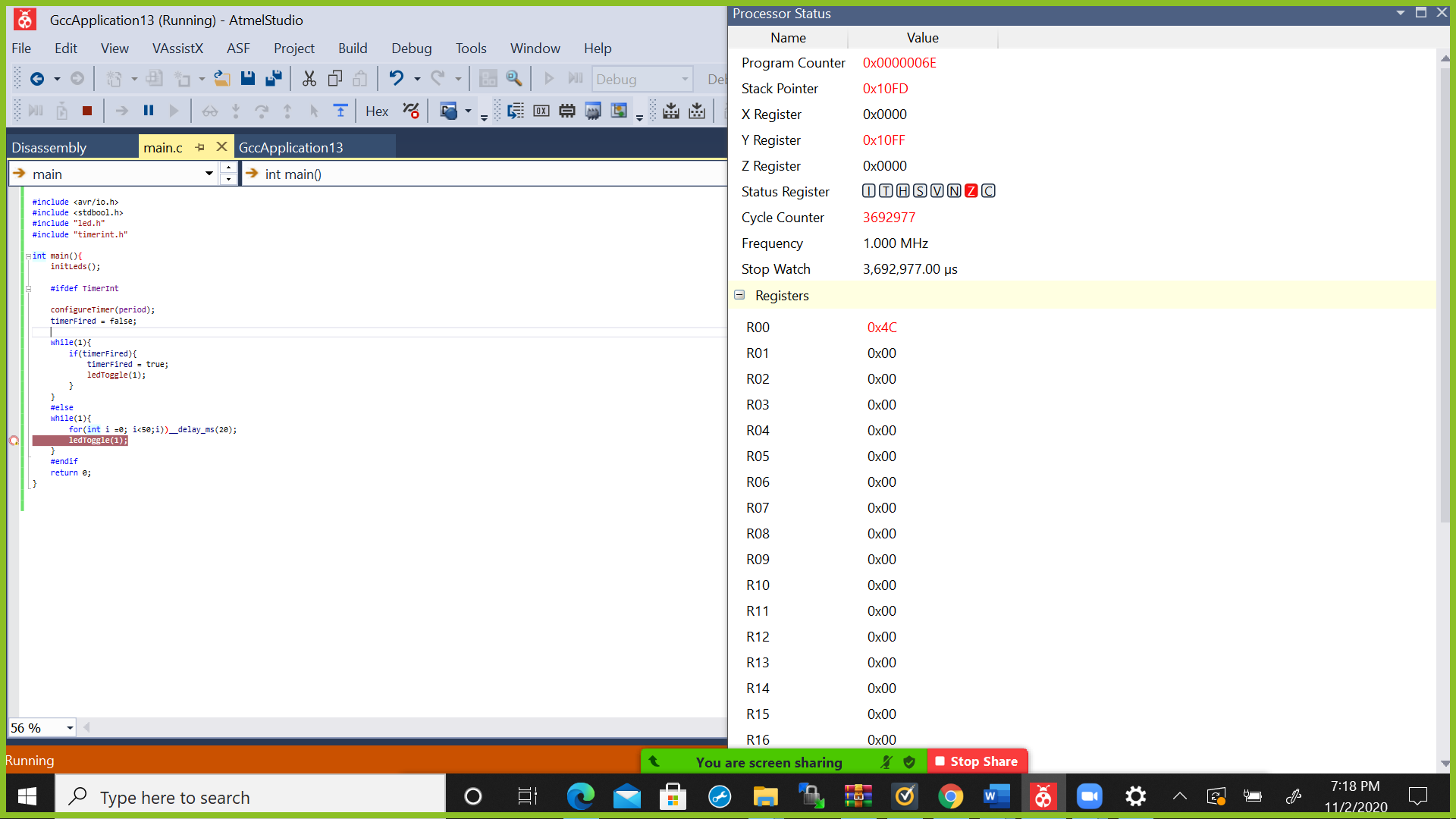
(a.2) Show the screen shot of the stopwatch on the second toggle in Atmel Studio.



(b.1) Show the code to use the delay function for implementation.



(b.2) Show the screen shot of the stopwatch on the third toggle in Atmel Studio.



5. (20)

Make a program to use the input interrupt to detect if a button is pressed with the PIC24E chip.

Assume the button is connected to Pin A0. If the button is pressed, turn the LED on. Otherwise, turn the LED off.

Use a state machine to handle the bouncing button issue. The state machine should toggle the LED only if the button changes its state for at least 10ms.

A part of code is in hw4.q5. Complete the code of led.c and button.c in hw4.q5.

(a) Draw the diagram of the state machine.



 `  Realase



(b) Complete the code of main.c that uses the state-centric approach to implement the diagram. Show the code of main.c.

#include <stdbool.h>

#include "Compiler.h"

#include "ConfigurationBits.h"

#include "led.h"

#include "delay.h"

#include "button.h"

#include "event.h"

#include "state.h"

#include "macros.h"

Event\_Button volatile evbutton;

Event\_Delay volatile evdelay;

int main() {

initLeds();

initButton();

State st = S0\_Released;

evbutton = Released;

evdelay = Expired;

while (1) {

#ifdef Polling

pollButton();

#endif

switch (st) {

case S0\_Released:

if (evbutton == Event\_Button) {

ledOn(1);

startDelay();

st = S1\_Pressed;

}

break;

case S1\_Pressed:

if (evdelay == Event\_Delay) {

st = S2\_Pressed;

}

break;

case S2\_Pressed:

if (evbutton == Event\_Button) {

ledOff(1);

startDelay();

st = S3\_Released;

}

break;

case S3\_Released:

if (evdelay == Event\_Delay) {

st = S0\_Released;

}

break;

default:

st = S0\_Released;

}

}

return 0;

}