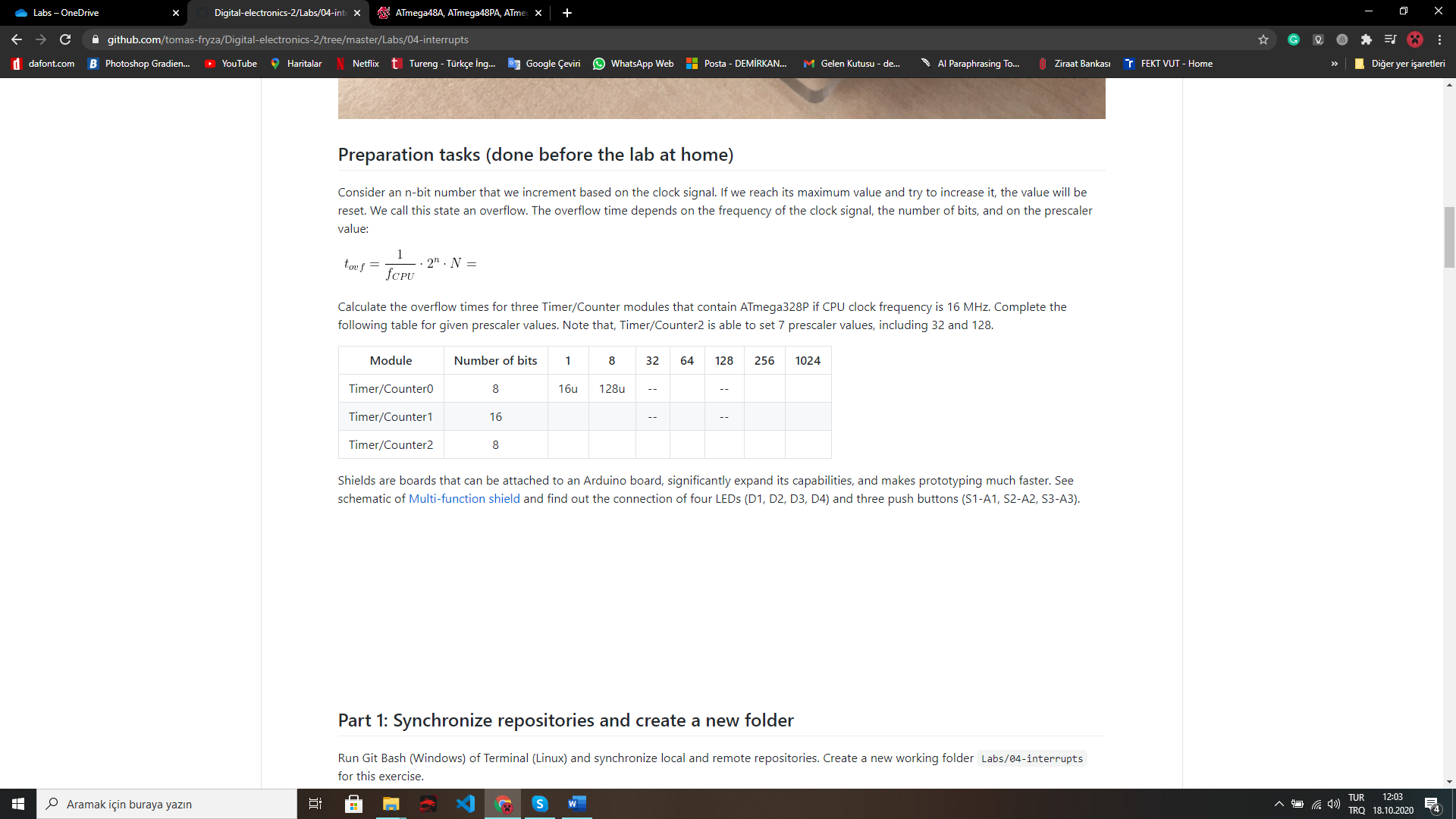
**DIGITAL ELECTRONICS 2 LAB ASSIGNMENT 4**

Name: Demirkan Korbey Baglamac



16ms

1ms

4ms

262ms

33ms

1s

4s

4ms

16u

128u

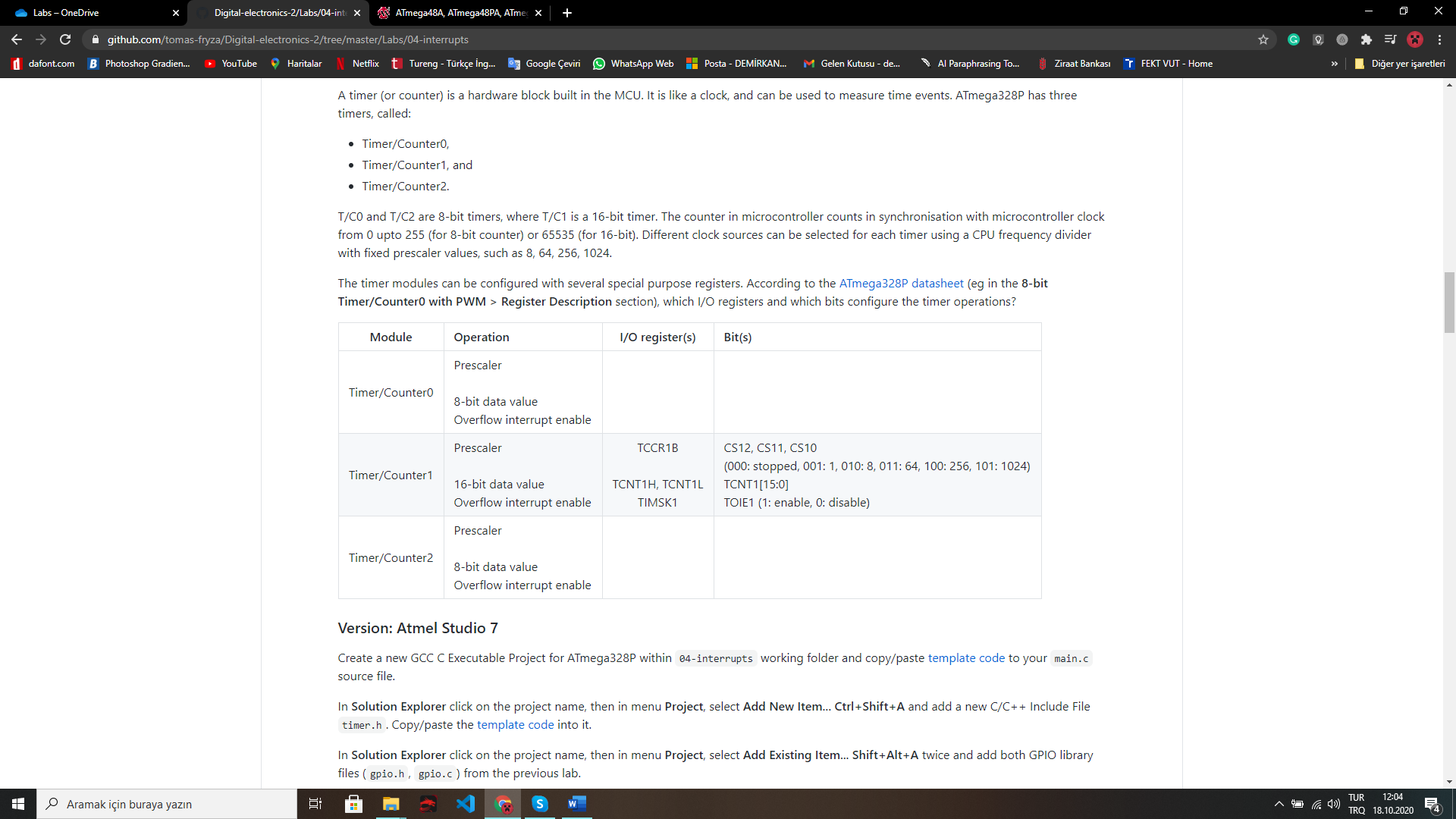
1ms

512u

16ms

4ms

2ms

1. 

TCCR0B

CS02, CS01, CS00 (000: stopped, 001: 1, 010: 8, 011: 64, 100: 256, 101: 1024)

TCNT0

TIMSK0

TCNT0[7:0]

TOIE0 (1: enable, 0:disable)

TCNT2

TCNT2[7:0]

TIMSK2

CS22, CS21, CS20 (000: stopped, 001: 1, 010: 8, 011: 32, 100: 64, 101:128, 110: 256, 111: 1024)

TOIE2 (1: enable, 0:disable)

TCCR2B

Time.h file:

#ifndef TIMER\_H

#define TIMER\_H

/\* Includes ----------------------------------------------------------\*/

#include <avr/io.h>

/\* Defines -----------------------------------------------------------\*/

/\*\*

\* @brief Defines prescaler CPU frequency values for Timer/Counter0.

\* @note F\_CPU = 16 MHz

\*/

#define TIM0\_stop() TCCR0B &= ~((1<<CS02) | (1<<CS01) | (1<<CS00));

#define TIM1\_overflow\_16u() TCCR0B &= ~((1<<CS02) | (1<<CS01)); TCCR0B |= (1<<CS00);

#define TIM1\_overflow\_128u() TCCR0B &= ~((1<<CS02) | (1<<CS00)); TCCR0B |= (1<<CS01);

#define TIM1\_overflow\_1ms() TCCR0B &= ~(1<<CS02); TCCR0B |= (1<<CS01) | (1<<CS00);

#define TIM1\_overflow\_4ms() TCCR0B &= ~(1<<CS01) | (1<<CS00); TCCR0B |= (1<<CS02);

#define TIM1\_overflow\_16ms() TCCR0B &= ~(1<<CS01); TCCR0B |= (1<<CS02) | (1<<CS00);

/\*\*

\* @brief Defines interrupt enable/disable modes for Timer/Counter0.

\*/

#define TIM0\_overflow\_interrupt\_enable() TIMSK0 |= (1<<TOIE0);

#define TIM0\_overflow\_interrupt\_disable() TIMSK0 &= ~(1<<TOIE0);

/\*\*

\* @brief Defines prescaler CPU frequency values for Timer/Counter1.

\* @note F\_CPU = 16 MHz

\*/

#define TIM1\_stop() TCCR1B &= ~((1<<CS12) | (1<<CS11) | (1<<CS10));

#define TIM1\_overflow\_4ms() TCCR1B &= ~((1<<CS12) | (1<<CS11)); TCCR1B |= (1<<CS10);

#define TIM1\_overflow\_33ms() TCCR1B &= ~((1<<CS12) | (1<<CS10)); TCCR1B |= (1<<CS11);

#define TIM1\_overflow\_262ms() TCCR1B &= ~(1<<CS12); TCCR1B |= (1<<CS11) | (1<<CS10);

#define TIM1\_overflow\_1s() TCCR1B &= ~((1<<CS11) | (1<<CS10)); TCCR1B |= (1<<CS12);

#define TIM1\_overflow\_4s() TCCR1B &= ~(1<<CS11); TCCR1B |= (1<<CS12) | (1<<CS10);

/\*\*

\* @brief Defines interrupt enable/disable modes for Timer/Counter1.

\*/

#define TIM1\_overflow\_interrupt\_enable() TIMSK1 |= (1<<TOIE1);

#define TIM1\_overflow\_interrupt\_disable() TIMSK1 &= ~(1<<TOIE1);

/\*\*

\* @brief Defines prescaler CPU frequency values for Timer/Counter2.

\* @note F\_CPU = 16 MHz

\*/

#define TIM2\_stop() TCCR2B &= ~((1<<CS22) | (1<<CS21) | (1<<CS20));

#define TIM2\_overflow\_16u() TCCR2B &= ~((1<<CS22) | (1<<CS21)); TCCR2B |= (1<<CS20);

#define TIM2\_overflow\_128u() TCCR2B &= ~((1<<CS22) | (1<<CS20)); TCCR2B |= (1<<CS21);

#define TIM2\_overflow\_512u() TCCR2B &= ~(1<<CS22); TCCR2B |= (1<<CS21) | (1<<CS20);

#define TIM2\_overflow\_1ms() TCCR2B &= ~((1<<CS21) | (1<<CS20)); TCCR2B |= (1<<CS22);

#define TIM2\_overflow\_2ms() TCCR2B &= ~(1<<CS21); TCCR2B |= (1<<CS22) | (1<<CS20);

#define TIM2\_overflow\_4ms() TCCR2B &= ~(1<<CS20); TCCR2B |= (1<<CS21) | (1<<CS22);

#define TIM2\_overflow\_16ms() TCCR2B |= (1<<CS22) | (1<<CS21) | (1<<CS20);

/\*\*

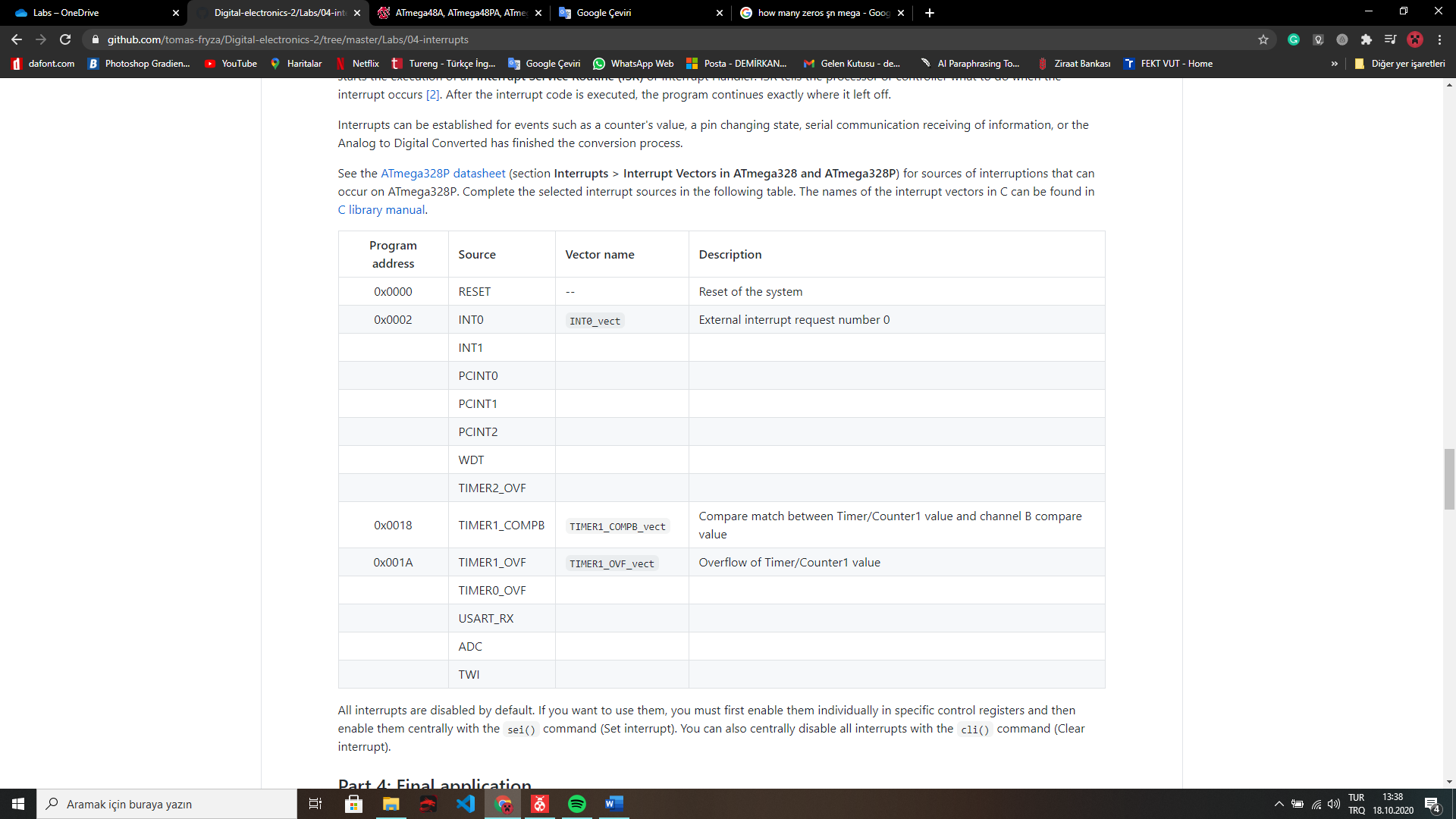
\* @brief Defines interrupt enable/disable modes for Timer/Counter2.

\*/

#define TIM2\_overflow\_interrupt\_enable() TIMSK2 |= (1<<TOIE2);

#define TIM2\_overflow\_interrupt\_disable() TIMSK2 &= ~(1<<TOIE2);

#endif



Pin change interrupt request 2

TWI\_vect

ADC\_ vect

USART\_RX\_vect

TIMER2\_OVF\_vect

WDT\_vect

PCINT2\_vect

PCINT1\_vect

PCINT0\_vect

INT1\_ vect

TIMER0\_OVF\_vect

2-wire Serial Interface

ADC Conversion Complete

USART, Tx Complete

Overflow of Timer/Counter0 value

Overflow of Timer/Counter2 value

öe

Watchdog Time-out Interrupt

Pin change interrupt request 1

Pin change interrupt request 0

External interrupt request number 1

0x0030

0x002A

0x0024

0x0020

0x0012

0x000C

0x000A

0x0008

0x0006

0x0004

Main.c file:

/\* Defines -----------------------------------------------------------\*/

#define LED\_D1 PB5

#define LED\_D2 PB4

#define LED\_D3 PB3

#define LED\_D4 PB2

#define BTN PD0

/\* Includes ----------------------------------------------------------\*/

#include <avr/io.h> // AVR device-specific IO definitions

#include <avr/interrupt.h> // Interrupts standard C library for AVR-GCC

#include "gpio.h" // GPIO library for AVR-GCC

#include "timer.h" // Timer library for AVR-GCC

/\* Function definitions ----------------------------------------------\*/

/\*\*

\* Main function where the program execution begins. Toggle three LEDs

\* on Multi-function shield with internal 8- and 16-bit timer modules.

\*/

// Global Variables for leds

*uint8\_t* leds[4] = {LED\_D1, LED\_D2, LED\_D3, LED\_D4};

int led\_count = 0;

int back = 0;

int main(void)

{

/\* Configuration of three LEDs \*/

GPIO\_config\_output(&DDRB, leds[0]);

GPIO\_write\_low(&PORTB, leds[0]);

GPIO\_config\_output(&DDRB, leds[1]);

GPIO\_write\_high(&PORTB, leds[1]);

GPIO\_config\_output(&DDRB, leds[2]);

GPIO\_write\_high(&PORTB, leds[2]);

GPIO\_config\_output(&DDRB, leds[3]);

GPIO\_write\_high(&PORTB, leds[3]);

/\*Setting up the push button\*/

GPIO\_config\_input\_pullup(&DDRD, BTN);

// Enables interrupts by setting the global interrupt mask

sei();

// Infinite loop

while (1)

{

//Determining if the button is pressed or not and adjusting the speed according to that

if(GPIO\_read(&PIND, BTN))

{

/\* Configuration of 16-bit Timer/Counter1

\* Set prescaler and enable overflow interrupt \*/

TIM1\_overflow\_1s();

TIM1\_overflow\_interrupt\_enable();

}

else

{

/\* Configuration of 16-bit Timer/Counter1

\* Set prescaler and enable overflow interrupt \*/

TIM1\_overflow\_262ms();

TIM1\_overflow\_interrupt\_enable();

}

}

// Will never reach this

return 0;

}

/\* Interrupt service routines ----------------------------------------\*/

ISR(TIMER1\_OVF\_vect)

{

//Toggling off the previous led

GPIO\_toggle(&PORTB, leds[led\_count]);

//Changing the direction

if(led\_count == 3) {

back = 1;

}

if(led\_count == 0) {

back = 0;

}

//Adjusting the led\_count value

if(back == 0){

led\_count = led\_count + 1;

}

else if(back == 1){

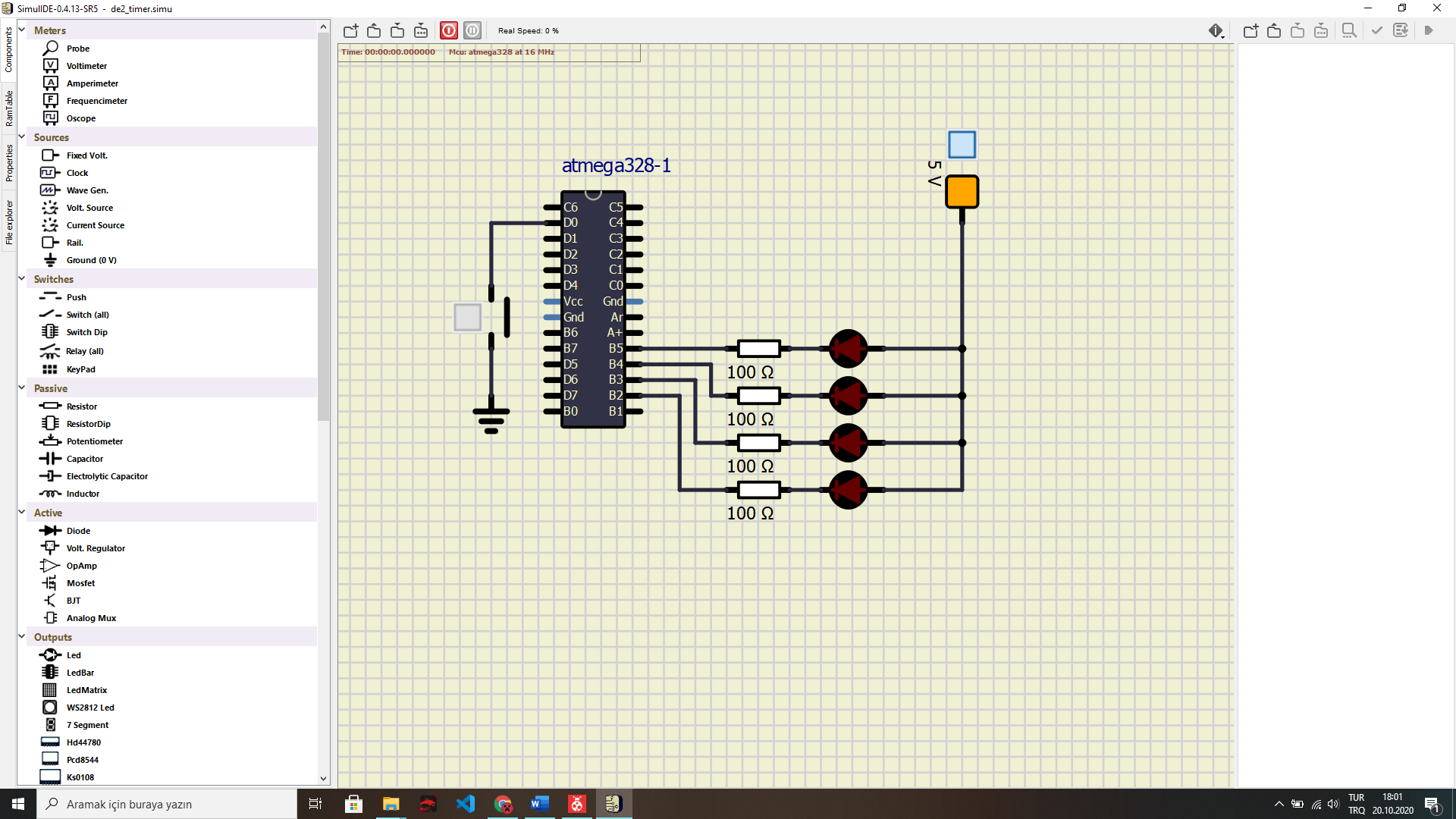
led\_count = led\_count - 1;

}

//Toggling on the new led

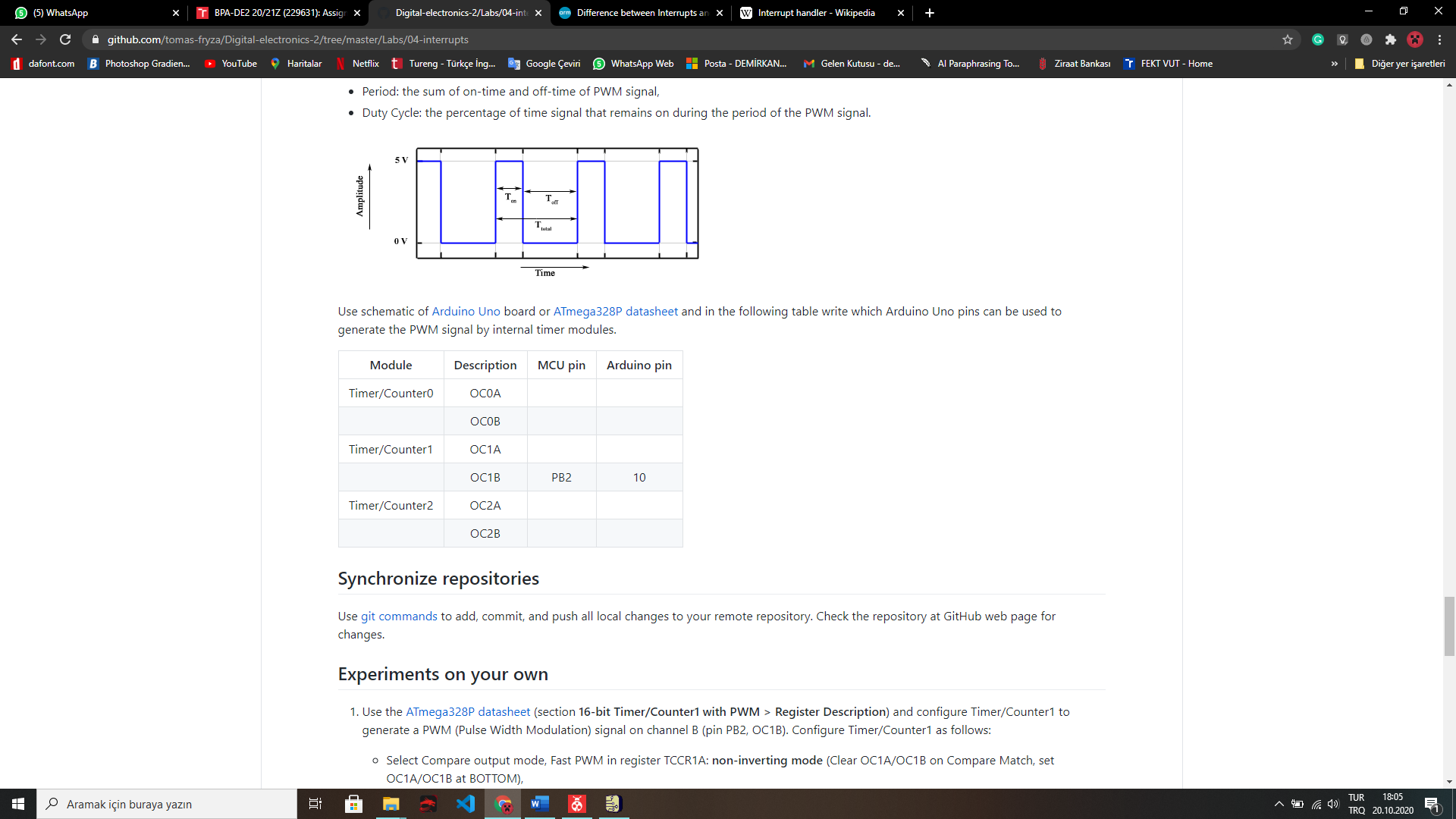
GPIO\_toggle(&PORTB, leds[led\_count]);

}

Screenshot of SimulIDE circuit:

In your words, describe the difference between a common C function and interrupt service routine:

Interrupt service routine is a speficitation of the microcontroller hardware but C function is just a executable code.



PD6

~6

PD5

~5

PB1

~9

PD3

PB3

~3

~11

**Clear timer on Compare Mode:** This mode gives us, more control on the Compare Match output frequency. Also it simplifies external event counting.

**Fast PWM Mode:** This mode provides high frequency PWM waveforms. This high frequency can be used in rectification, power regulation and more.