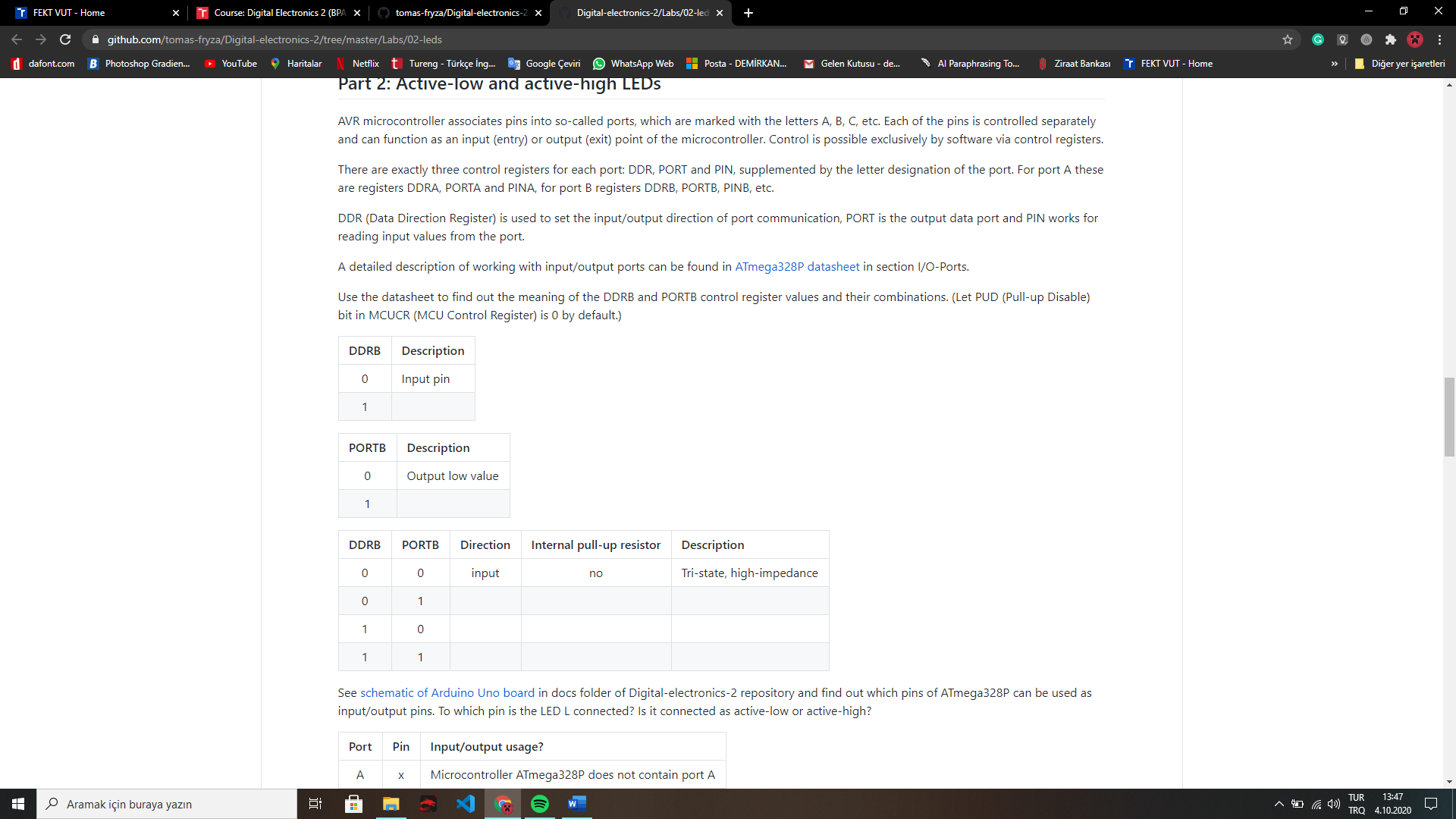
**DIGITAL ELECTRONICS 2 LAB ASSIGNMENT 2**

Name: Demirkan Korbey Baglamac

1. 

Pxn will source current

Output high

Output low

no

no

yes

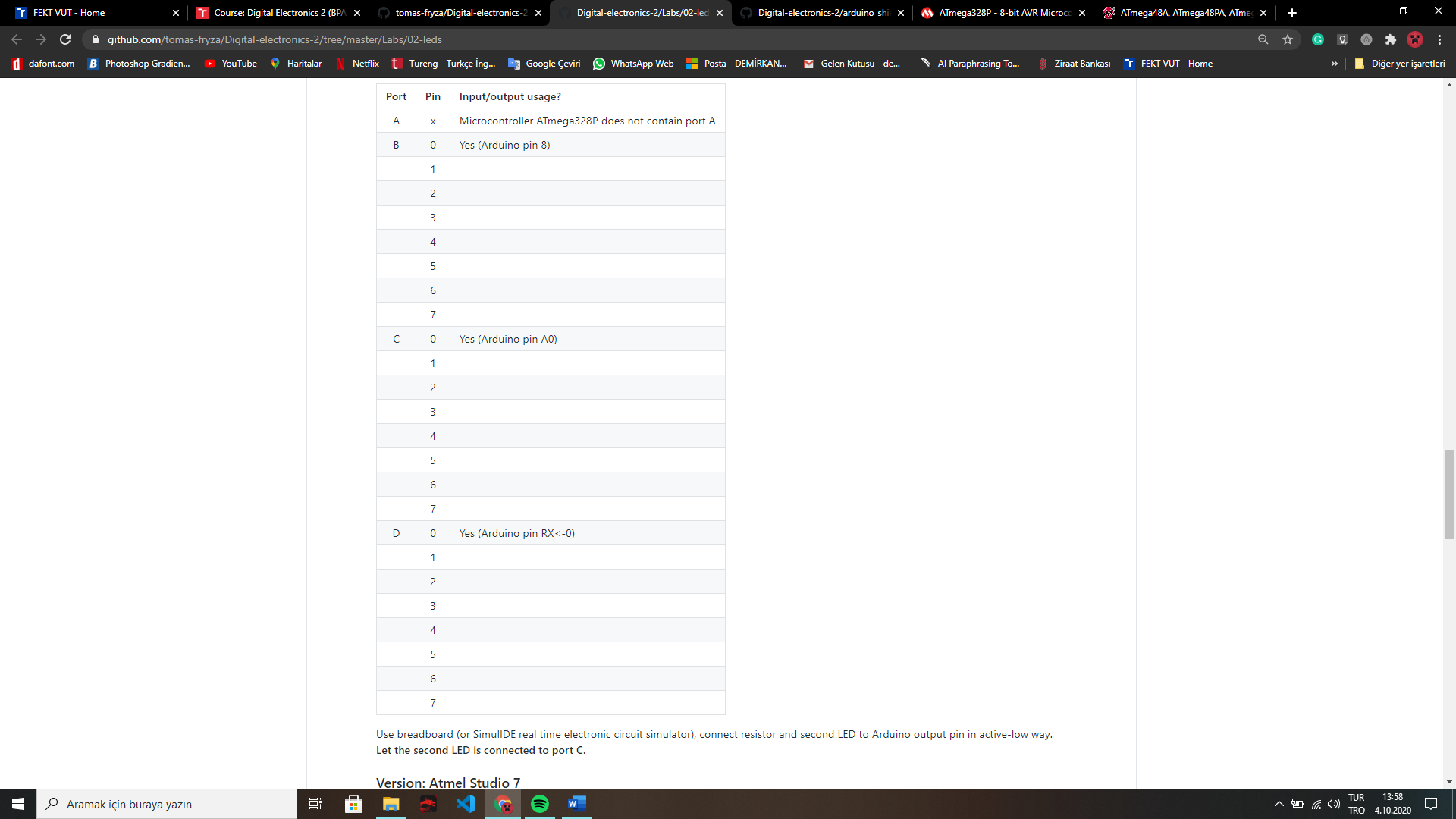
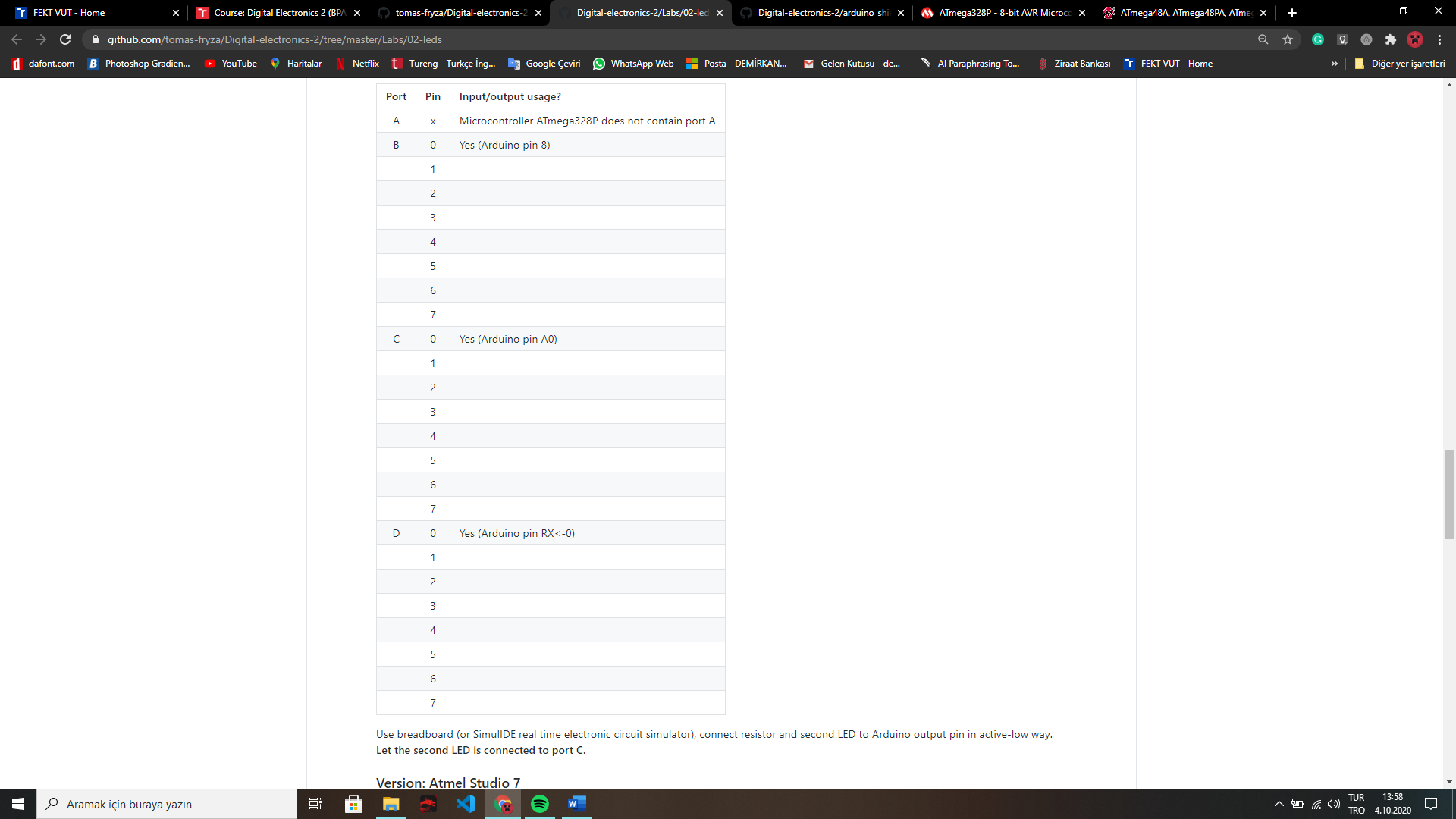
output

output

input

Output high value

Output pin



Yes (Arduino pin 7)

Yes (Arduino pin ~6)

Yes (Arduino pin ~5)

Yes (Arduino pin 4)

Yes (Arduino pin ~3)

Yes (Arduino pin 2)

Yes (Arduino pin TX ->1)

Yes (Arduino pin 13)

Yes (Arduino pin 12)

Yes (Arduino pin ~11)

Yes (Arduino pin ~10)

Yes (Arduino pin ~9)

Yes (Arduino pin A2)

Yes (Arduino pin A1)

Yes (Arduino pin A3)

Yes (Arduino pin A4)

Yes (Arduino pin A5)

C Code with two leds and a push button:

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

#define LED\_GREEN   PB5     // AVR pin where green LED is connected

#define LED\_RED     PC0

#define BLINK\_DELAY 500

#define BTN         PD0

#ifndef F\_CPU

#define F\_CPU 16000000      // CPU frequency in Hz required for delay

#endif

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

#include <util/delay.h>     // Functions for busy-wait delay loops

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

/\* Functions ---------------------------------------------------------\*/

/\*\*

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

 \* when a push button is pressed.

 \*/

int main(void)

{

    /\* GREEN LED \*/

    // Set pin as output in Data Direction Register...

    DDRB = DDRB | (1<<LED\_GREEN);

    // ...and turn LED off in Data Register

    PORTB = PORTB & ~(1<<LED\_GREEN);  //Turn OFF

    /\* second LED \*/

    DDRC = DDRC | (1<<LED\_RED);

    PORTC = PORTC & ~(1<<LED\_RED); // Turn ON

    /\* PUSH BUTTON\*/

    DDRD = DDRD & ~(1<<BTN); //input

    PORTD = PORTD | (1<<BTN); //enable internal pull up

    // Infinite loop

    while (1)

    {

        // Pause several milliseconds

        \_delay\_ms(BLINK\_DELAY);

        // WRITE YOUR CODE HERE

        if(bit\_is\_clear(PIND, BTN))

        {

            PORTB = PORTB ^ (1<<LED\_GREEN); //Invert

            PORTC = PORTC ^ (1<<LED\_RED); //Invert

        }

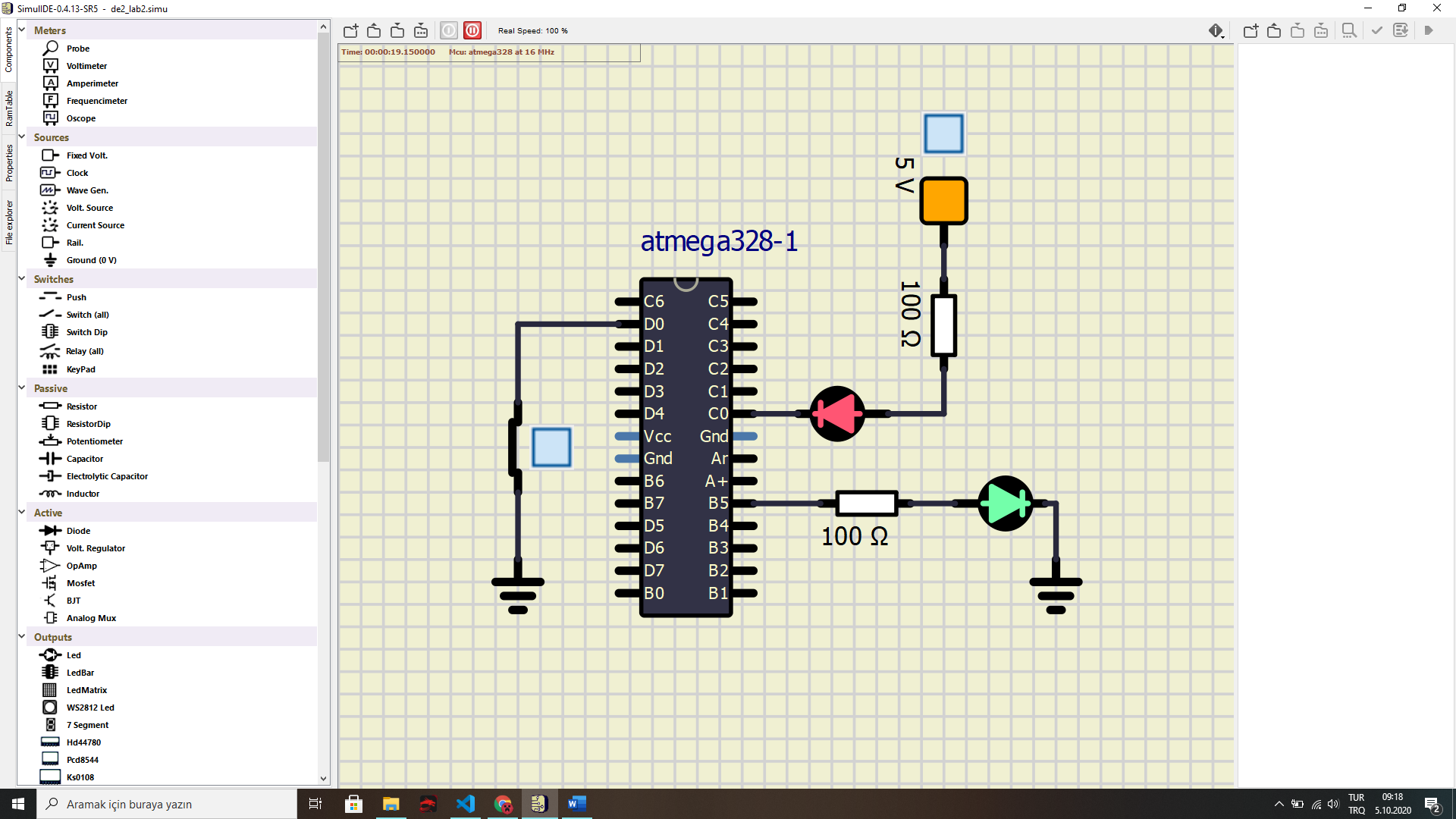
    }

    // Will never reach this

    return 0;

}

Screenshot of SimulIDE Ciruit:



1. Knight Rider Application Code:

/\*

 \* knight\_rider.c

 \*

 \* Created: 5.10.2020 09:29:44

 \* Author : dkorb

 \*/

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

#define LED1 PB1

#define LED2 PB2

#define LED3 PB3

#define LED4 PB4

#define LED5 PB5

#define BLINK\_DELAY 500

#define BTN         PD0

#ifndef F\_CPU

#define F\_CPU 16000000      // CPU frequency in Hz required for delay

#endif

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

#include <util/delay.h>     // Functions for busy-wait delay loops

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

/\* Functions ---------------------------------------------------------\*/

void clear();

/\*\*

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

 \* when a push button is pressed.

 \*/

int main(void)

{

    /\* Setting the DDRx as output for LEDs \*/

    DDRB = DDRB | (1<<LED1);

    DDRB = DDRB | (1<<LED2);

    DDRB = DDRB | (1<<LED3);

    DDRB = DDRB | (1<<LED4);

    DDRB = DDRB | (1<<LED5);

    /\* PUSH BUTTON\*/

    DDRD = DDRD & ~(1<<BTN); //input

    PORTD = PORTD | (1<<BTN); //enable internal pull up

    // Infinite loop

    while (1)

    {

            clear();

            loop\_until\_bit\_is\_clear(PIND, BTN);

            PORTB = PORTB | (1<<LED1);

            loop\_until\_bit\_is\_clear(PIND, BTN);

            clear();

            PORTB = PORTB | (1<<LED2);

            loop\_until\_bit\_is\_clear(PIND, BTN);

            clear();

            PORTB = PORTB | (1<<LED3);

            loop\_until\_bit\_is\_clear(PIND, BTN);

            clear();

            PORTB = PORTB | (1<<LED4);

            loop\_until\_bit\_is\_clear(PIND, BTN);

            clear();

            PORTB = PORTB | (1<<LED5);

            loop\_until\_bit\_is\_clear(PIND, BTN);

    }

    // Will never reach this

    return 0;

}

void clear() {

    // Pause several milliseconds

    \_delay\_ms(BLINK\_DELAY);

    //Making all outputs low for the LEDs

    PORTB = PORTB & ~(1<<LED1);

    PORTB = PORTB & ~(1<<LED2);

    PORTB = PORTB & ~(1<<LED3);

    PORTB = PORTB & ~(1<<LED4);

    PORTB = PORTB & ~(1<<LED5);

}

Knight Rider Application Circuit:

