**Kamil Káčer 211777 Repository: https://github.com/kamilio14/Digital-electronics-2.git**

**Work Done in a lab**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Alternately toggle two LEDs when a push button is pressed.

\* ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2

\*

\* Copyright (c) 2018-2020 Tomas Fryza

\* Dept. of Radio Electronics, Brno University of Technology, Czechia

\* This work is licensed under the terms of the MIT license.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

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

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

#define LED\_RED PC0

#define BTN PD0

#define BLINK\_DELAY 250

#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);

/\* second LED \*/

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

DDRC = DDRC | (1<<LED\_RED);

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

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

// Set pin as input

DDRD = DDRD & ~(1<<BTN);

// BTN Set to 1 because is BTN active low

PORTD = PORTD | (1<<BTN);

// Infinite loop

while (1)

{

// Pause several milliseconds

*\_delay\_ms*(BLINK\_DELAY);

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

{

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

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

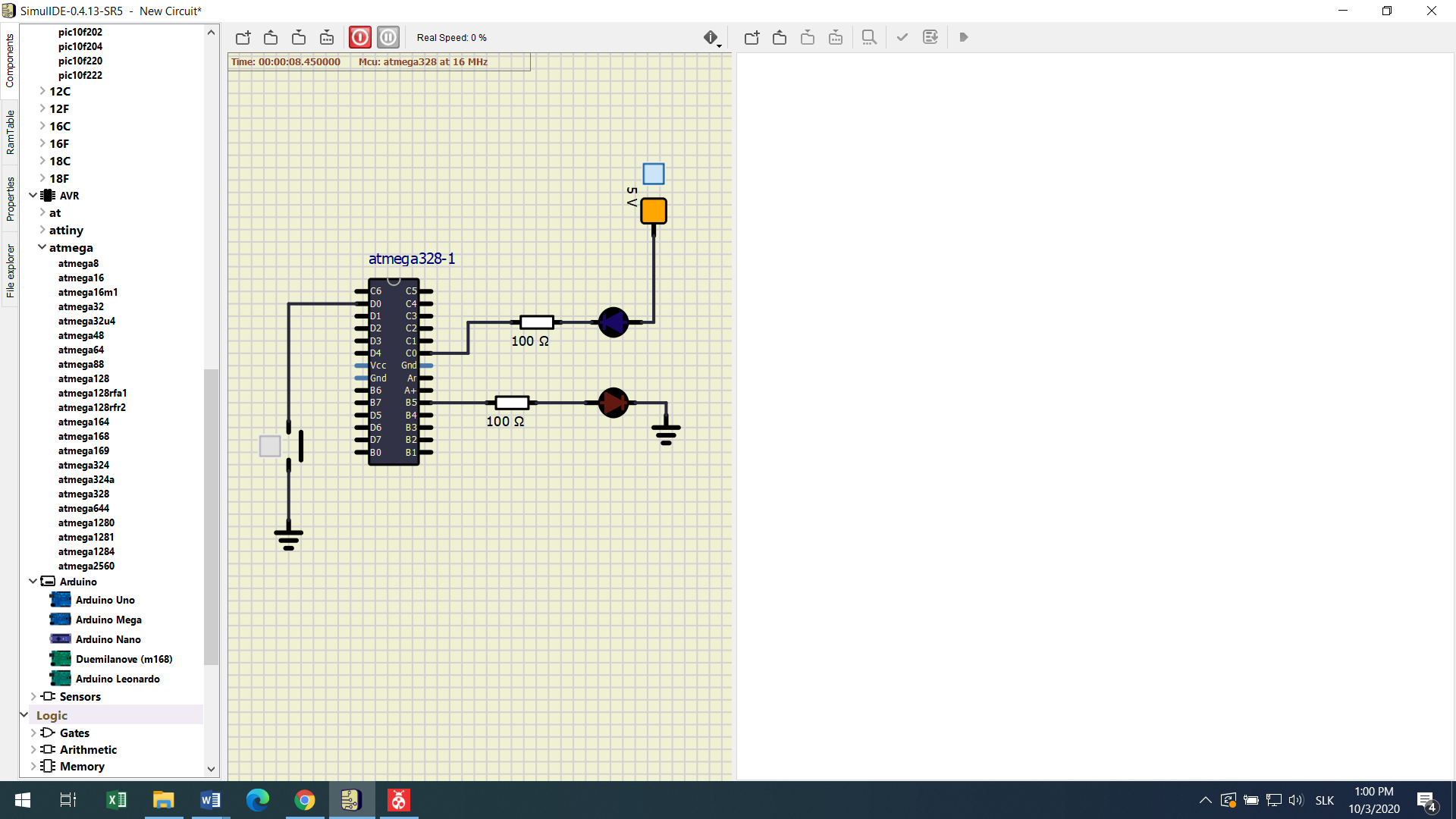
}

}

// Will never reach this

return 0;

}



**Home assignment**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Alternately toggle two LEDs when a push button is pressed.

\* ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2

\*

\* Copyright (c) 2018-2020 Tomas Fryza

\* Dept. of Radio Electronics, Brno University of Technology, Czechia

\* This work is licensed under the terms of the MIT license.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

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

#define LED\_GREEN0 PB0 // AVR pin where green LED is connected

#define LED\_GREEN1 PB1

#define LED\_GREEN2 PB2

#define LED\_GREEN3 PB3

#define LED\_GREEN4 PB4

#define BUTTON\_LED PC1

#define BTN PD0

#define BLINK\_DELAY 500

#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)

{

int btn\_memory = 0; // turnig off and on leds

// LED 0

DDRB = DDRB | (1<<LED\_GREEN0);

PORTB = PORTB & ~(1<<LED\_GREEN0);

// LED 1

DDRB = DDRB | (1<<LED\_GREEN1);

PORTB = PORTB & ~(1<<LED\_GREEN1);

// LED 2

DDRB = DDRB | (1<<LED\_GREEN2);

PORTB = PORTB & ~(1<<LED\_GREEN2);

// LED 3

DDRB = DDRB | (1<<LED\_GREEN3);

PORTB = PORTB & ~(1<<LED\_GREEN3);

// LED 4

DDRB = DDRB | (1<<LED\_GREEN4);

PORTB = PORTB & ~(1<<LED\_GREEN4);

//BUTTON

DDRD = DDRD & ~(1<<BTN);

PORTD = PORTD | (1<<BTN);

//button led

DDRC = DDRC | (1<<BUTTON\_LED);

PORTC = PORTC & ~(1<<BUTTON\_LED);

// Infinite loop

while (1)

{

if (bit\_is\_clear(PIND, BTN))

{

if (btn\_memory == 0)

{

btn\_memory = 1; // turning on

PORTC = PORTC ^ (1<<BUTTON\_LED); // Turning on button led to see if it works

}else{

btn\_memory = 0;

PORTC = PORTC ^ (1<<BUTTON\_LED); // Turning off button led

*\_delay\_ms*(5000); // Time for turning off button

}

}

if (btn\_memory == 1)

{

PORTB = PORTB ^ (1<<LED\_GREEN0); // Turning on Led0

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN0); // Turning off Led0

PORTB = PORTB ^ (1<<LED\_GREEN1); // Turning on Led1

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN1); // Turning off Led1

PORTB = PORTB ^ (1<<LED\_GREEN2); // Turning on Led2

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN2); // Turning off Led2

PORTB = PORTB ^ (1<<LED\_GREEN3); // Turning on Led3

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN3); // Turning off Led3

PORTB = PORTB ^ (1<<LED\_GREEN4); // Turning on Led4

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN4); // Turning off Led4

PORTB = PORTB ^ (1<<LED\_GREEN3); // Turning on Led3

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN3); // Turning off Led3

PORTB = PORTB ^ (1<<LED\_GREEN2); // Turning on Led2

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN2); // Turning off Led2

PORTB = PORTB ^ (1<<LED\_GREEN1); // Turning on Led1

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN1); // Turning off Led1

PORTB = PORTB ^ (1<<LED\_GREEN0); // Turning on Led0

*\_delay\_ms*(BLINK\_DELAY);

PORTB = PORTB ^ (1<<LED\_GREEN0); // Turning on Led0

*\_delay\_ms*(BLINK\_DELAY);

}else

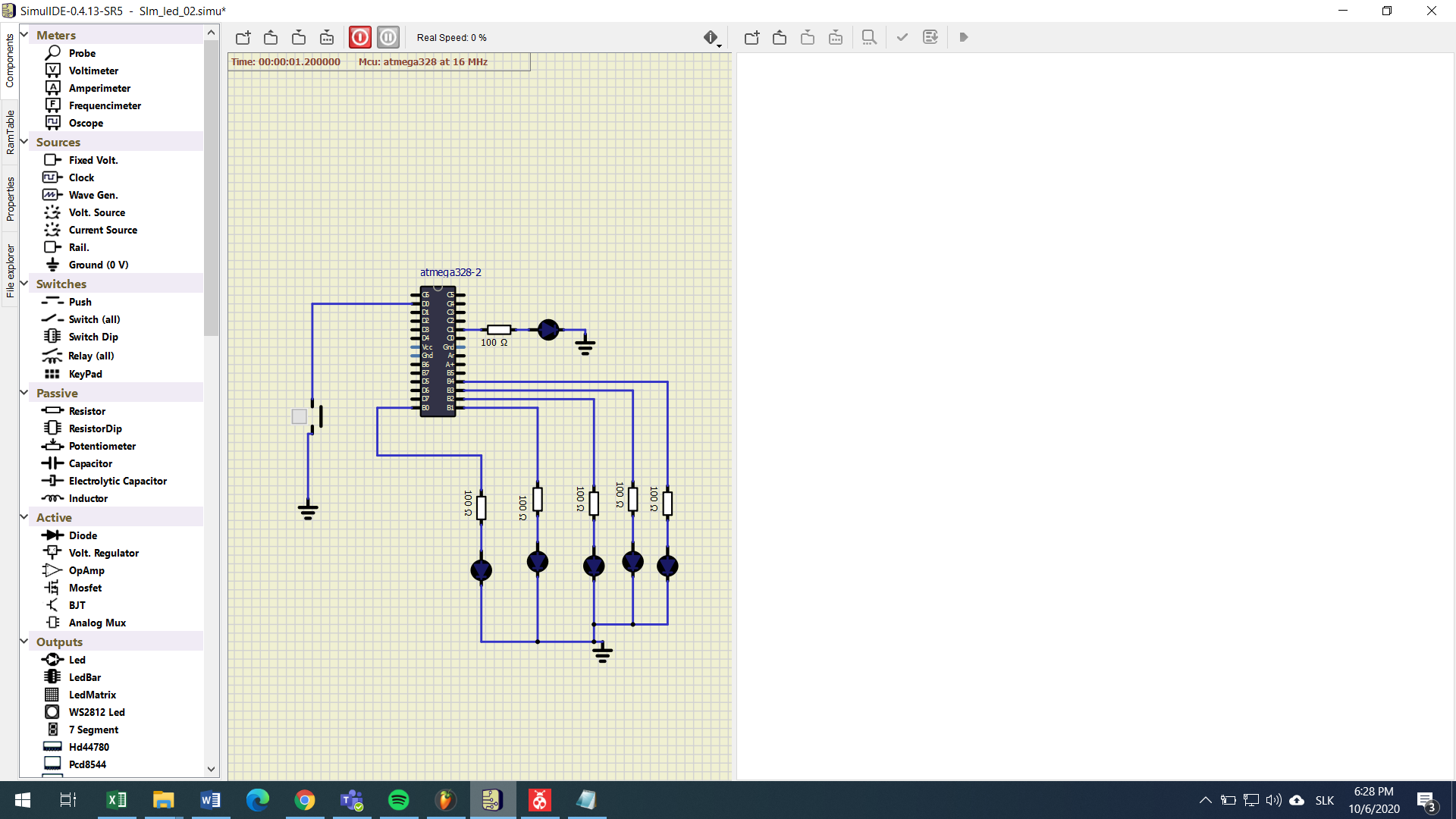
{

}

}

return 0;

}



**Charts**







