**Цель работы:** измерение длительности интервала времени.

**Задание:** разработать алгоритм, измеряющий время, которое определяется нажатием кнопки.

Вариант 5

Необходимо измерять время между двумя нажатиями кнопки.

Результат работы.

Код.

#define F\_CPU 8000000L

// Работа с таймером AVR. Подсчёт количества секунд между двумя нажатиями кнопки в 16 системе счисления. Точность таймера 0.1 (s)

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

void initDisplay(){

DDRB = 0xFF; // Порт B - выход

PORTB = 0b10000000;

DDRD = 0xFF; // Порт D - выход

PORTD = 0x00;

}

void initTimer\_0() {

// Настройка таймера 0 для прерываний каждые \_\_\_ миллисекунд

TCCR0A = 0x00;

TCCR0B = (1 << CS01) | (1 << CS00); // Предделитель 64

TIMSK0 = (1 << TOIE0); // Разрешить прерывание по переполнению таймера

}

void initTimer\_1() {

// Настройка таймера 1 для прерываний каждые \_\_\_ миллисекунд

// инициализация Timer1

cli(); // отключить глобальные прерывания

TCCR1A = 0; // установить регистры в 0

TCCR1B = 0;

OCR1A = 3125; // установка регистра совпадения

TCCR1B |= (1 << WGM12); // включение в CTC режим

// Установка битов CS12 на коэффициент деления 256

TCCR1B |= (1 << CS12);

TIMSK1 |= (1 << OCIE1A); // включение прерываний по совпадению

sei(); // включить глобальные прерывания

}

void initButton()

{

DDRC = 0x00;// инициализируем порт на вход

PORTC = 0xFF; // включаем подтягивающие резистры (на пинах будет высокий уровень, кнопка будет притягивать к земле и таким оброзом фиксироваться нажатие)

}

//---------------------------0-----1-----2-----3-----4-----5-----6-----7-----8------9-----10----11----12----13---14----15

unsigned char SEGMENTE[] = { 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F, 0xF7, 0xFC, 0x39, 0x5E, 0x76, 0x71 };

unsigned long display = 0;

unsigned char segCounter = 0;

bool pushButton = false;

void outNumberOnDisplay(long display )

{

PORTD = 0xFF; // Гасим все разряды

PORTB = (1 << segCounter); // Выбираем следующий разряд. Запись (1 << segCounter) устанавливает нужный порт в 1, а все остальные в 0

switch (segCounter)

{

case 0:

PORTD = ~(SEGMENTE[display % 65536 / 4096]); // Раскладываем число на разряды

break;

case 1:

PORTD = ~(SEGMENTE[display % 4096 / 256]);

break;

case 2:

PORTD = ~(SEGMENTE[display % 256 / 16]);

break;

case 3:

PORTD = ~(SEGMENTE[display % 16]);

break;

}

if (segCounter++ > 2) segCounter = 0;

}

ISR(TIMER0\_OVF\_vect) {

outNumberOnDisplay(display);

}

ISR(TIMER1\_COMPA\_vect)

{

if (pushButton) {

display++;

if (display == 65535)

pushButton = false;

}

}

// Главная функция

int main(void)

{

initDisplay();

initTimer\_0();

initTimer\_1();

bool resetDisplay = false;

while (1)

{

if (!(PINC & (1 << PC1))) //если на первом пине низкий уровень (кнопка прижала контакт к земле) выаолняем условие

{

pushButton = !pushButton;

if (pushButton == false && resetDisplay == true) {

\_delay\_ms(5000);

display = 0;

}

resetDisplay = true;

\_delay\_ms(300);

}

}

return 0;

}

Схема.

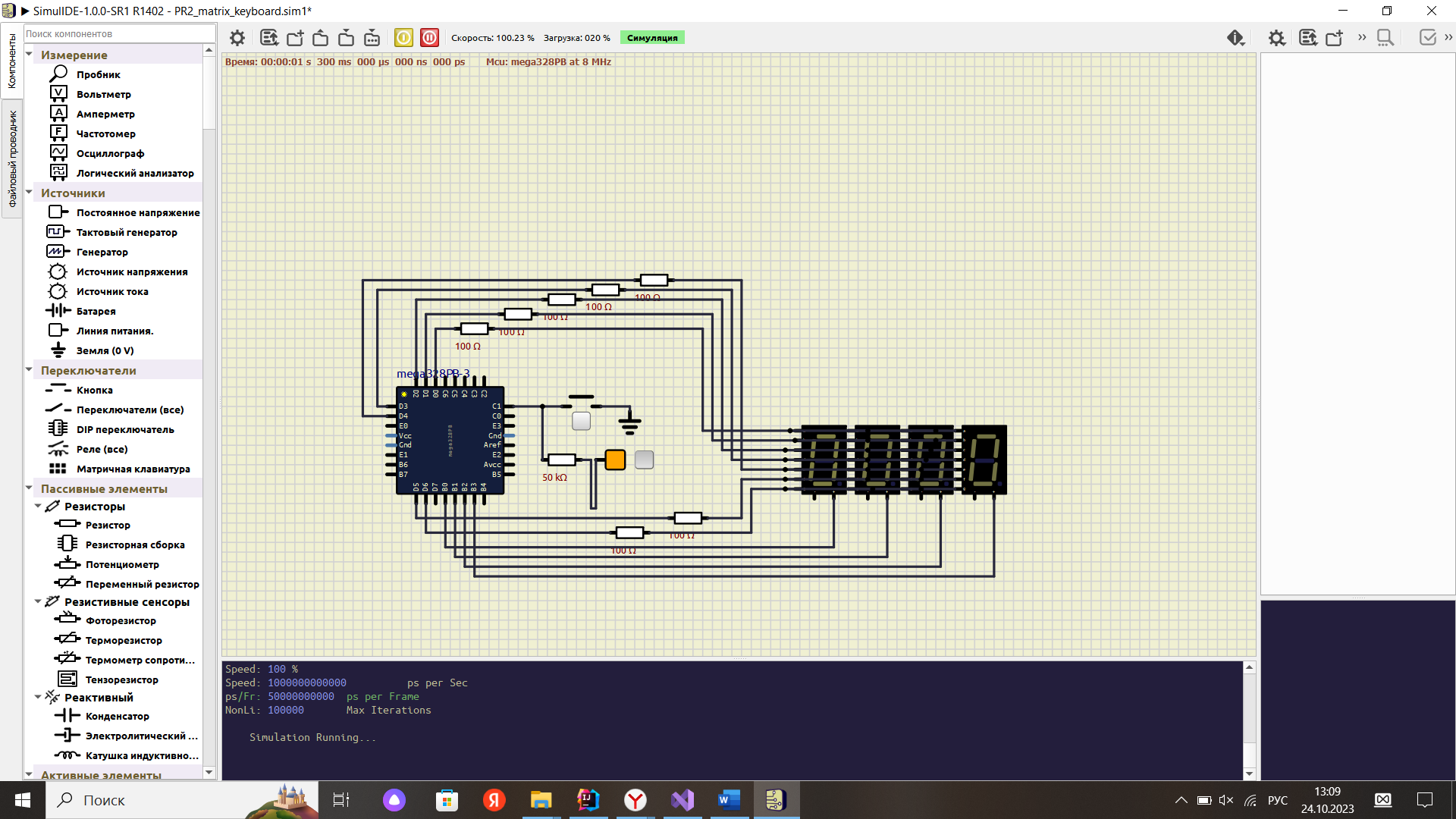


Рисунок 1 – Схема.