**Цель работы:** считывание с клавиатуры и вывод на индикаторы.

**Задание:** сформировать алгоритм цифрового замка.

Вариант 5

Результат работы.

Код.

#define F\_CPU 8000000L

// Подключение семисегментных индикаторов к AVR. Динамическая индикация. Опрос матричной клавиатуры. Использование таймера.

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

unsigned char key;

unsigned int pin7=0;

// Функция для инициализации порта DDRC

void initKeypad() {

DDRC = 0b0000111; // Настройка нижних 4 бит порта как выходы и верхних 4 бит как входы

PORTC =0b1111111; // Подключение подтягивающих резисторов к верхним 4 битам

}

void initDisplay(){

DDRB = 0xFF; // Порт B - выход

PORTB = 0b10000000;

DDRD = 0xFF; // Порт D - выход

PORTD = 0x00;

}

void initTimer() {

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

TCCR0A = 0x00;

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

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

}

// Функция для опроса клавиатуры

char readKeypad() {

PORTC |= 0b1111111; // Установим нижние 3 бита в высокий уровень и верхние 4 в режим чтения

char col, row;

for (col = 0; col < 3; col++) {

// Устанавливаем нижние 3 бита поочередно в 0

PORTC &= ~(1 << col);

for (row = 3; row < 7; row++) {

if (!(PINC & (1 << row))) {

// Клавиша нажата

// Вернем символ в зависимости от положения клавиши

if (col == 0 && row == 3) return '1';

if (col == 1 && row == 3) return '2';

if (col == 2 && row == 3) return '3';

if (col == 0 && row == 4) return '4';

if (col == 1 && row == 4) return '5';

if (col == 2 && row == 4) return '6';

if (col == 0 && row == 5) return '7';

if (col == 1 && row == 5) return '8';

if (col == 2 && row == 5) return '9';

if (col == 1 && row == 6) return '0';

if (col == 2 && row == 6) return '#';

}

}

PORTC |= 0b1111111; // Установим нижние 3 бита в высокий уровень и верхние 4 в режим чтения

}

return 0; // Ни одна клавиша не нажата

}

//---------------------------0-----1-----2-----3-----4-----5-----6-----7-----8------9

unsigned char SEGMENTE[] = { 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F };

unsigned char segCounter = 0;

struct segment

{

int number;

bool saveNumber = false;

};

segment display\_1;

segment display\_2;

segment display\_3;

segment display\_4;

bool codeRight;

void outNumberOnDisplay(char key)

{

if (key != 0) {

if (!display\_1.saveNumber) {

display\_1.number = (key - '0');

display\_1.saveNumber = true;

\_delay\_ms(300);

}

else if (!display\_2.saveNumber) {

display\_2.number = (key - '0');

display\_2.saveNumber = true;

\_delay\_ms(300);

}

else if (!display\_3.saveNumber) {

display\_3.number = (key - '0');

display\_3.saveNumber = true;

\_delay\_ms(300);

}

else if (!display\_4.saveNumber) {

display\_4.number = (key - '0');

display\_4.saveNumber = true;

\_delay\_ms(300);

}

else if (key == '#') {

if (display\_1.number == 0 && display\_2.number == 5 && display\_3.number == 0 && display\_4.number == 5)

//codeRight = true;

pin7 = 1;

display\_1.number = 0; display\_2.number = 0; display\_3.number = 0; display\_4.number = 0;

display\_1.saveNumber = 0; display\_2.saveNumber = 0; display\_3.saveNumber = 0; display\_4.saveNumber = 0;

\_delay\_ms(300);

}

}

PORTD = 0xFF; // Гасим все разряды

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

//PORTB &= ~(1 << PB0) | ~(1 << PB1) | ~(1 << PB2) | ~(1 << PB3) | ~(1 << PB4) | ~(1 << PB5) | ~(1 << PB6) | ~(1<<PB7);

PORTB |= pin7 << PB7;

switch (segCounter)

{

case 0:

PORTD = ~(SEGMENTE[display\_1.number % 10]); // Раскладываем число на разряды

break;

case 1:

PORTD = ~(SEGMENTE[display\_2.number % 10]);

break;

case 2:

PORTD = ~(SEGMENTE[display\_3.number % 10]);

break;

case 3:

PORTD = ~(SEGMENTE[display\_4.number % 10]);

break;

}

//PORTB &= ~(1 << segCounter);

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

}

ISR(TIMER0\_OVF\_vect) {

key = readKeypad();

outNumberOnDisplay(key);

}

// Главная функция

int main(void)

{

initKeypad();

initDisplay();

initTimer();

sei(); // Разрешить глобальные прерывания

codeRight = false;

while (1)

{

}

return 0;

}

Схема.

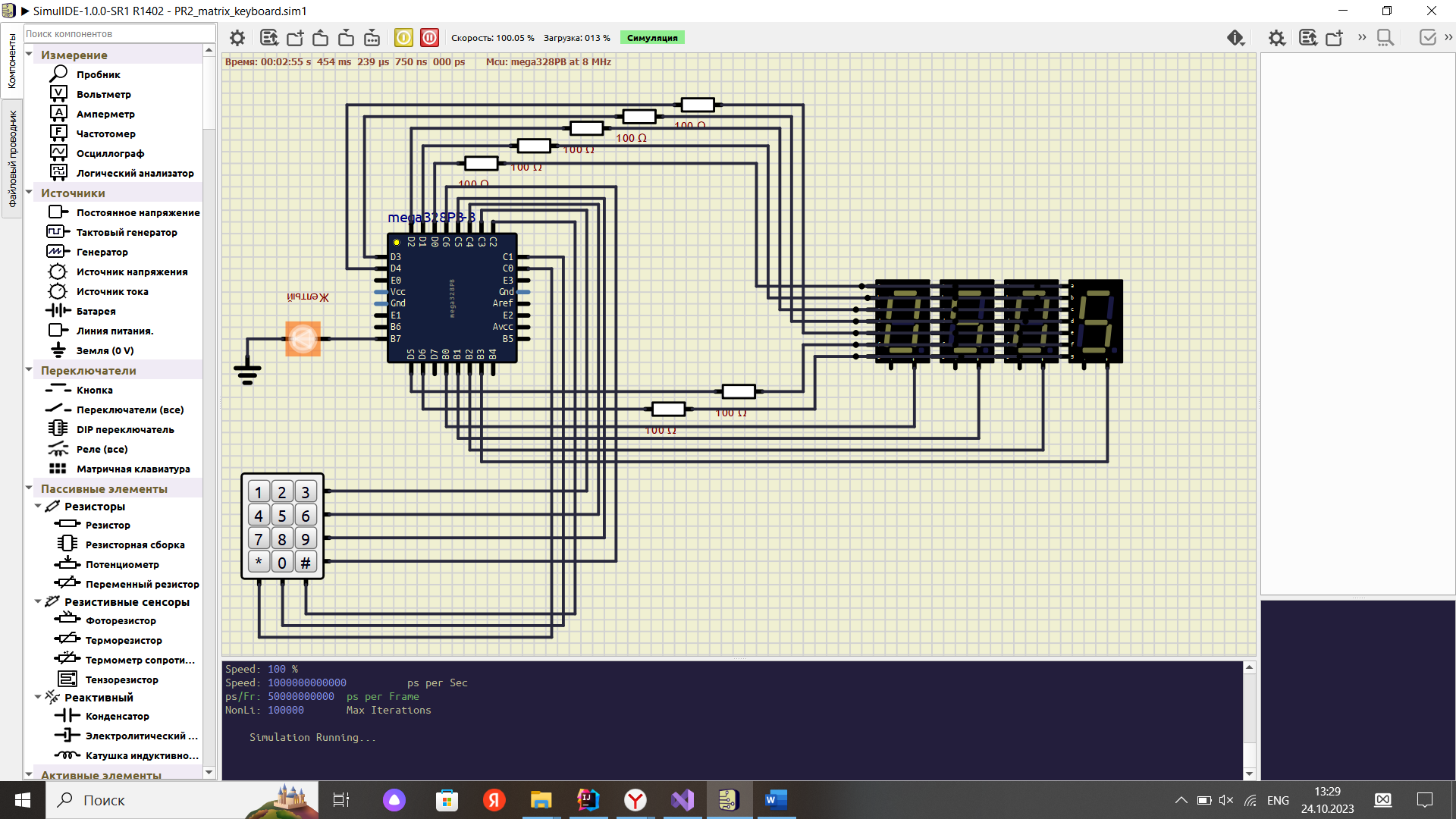


Рисунок 1 – Схема.