Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации Сибирский Государственный Университет
Телекоммуникаций и Информатики СибГУТИ

Кафедра вычислительных систем

Расчетно-графическая работа
по дисциплине «Микропроцессорные системы»
на тему:
«Реализация сигнального таймера/счетчика»

Выполнил:
ст. гр. ИВ-921
Ярошев Р.А.

Проверил:
старший преподаватель
Гонцова А.В.

Новосибирск 2022

Оглавление

Задание

Используя изученные схемы, реализовать сигнальный таймер/счётчик с выводом на семисегментный индикатор.

Данный таймер должен отсчитать введённое с клавиатуры время и по окончании счёта издать звуковой сигнал. Каждый отсчёт на семисегментном индикаторе должно изменяться показание (уменьшаться или увеличиваться).

Для этого:

1. Выбрать необходимые компоненты
2. Разработать схему устройства
3. Разработать алгоритм работы программы
4. Собрать схему
5. Написать программу
6. Провести отладку и тестирование разработанного устройства

Описание компонентов

В данной работе были использованы следующие компоненты:

- Плата «Arduino Uno».
- Мембранная матричная клавиатура 4*4.
- Шлейф для мембранной клавиатуры восьми канальный.
- 2 резистора на 1 кОм (коричневый, черный, красный), 2 резистора на 10 кОм (коричневый, черный, оранжевый), 8 резисторов на 100 Ом (коричневый, черный, коричневый).
- Пьезоизлучатель.
- Семисегментный индикатор.
- Макетная плата (5*17) x 2.
- Соединительные провода типа «папа-папа».

Схема принципиальная

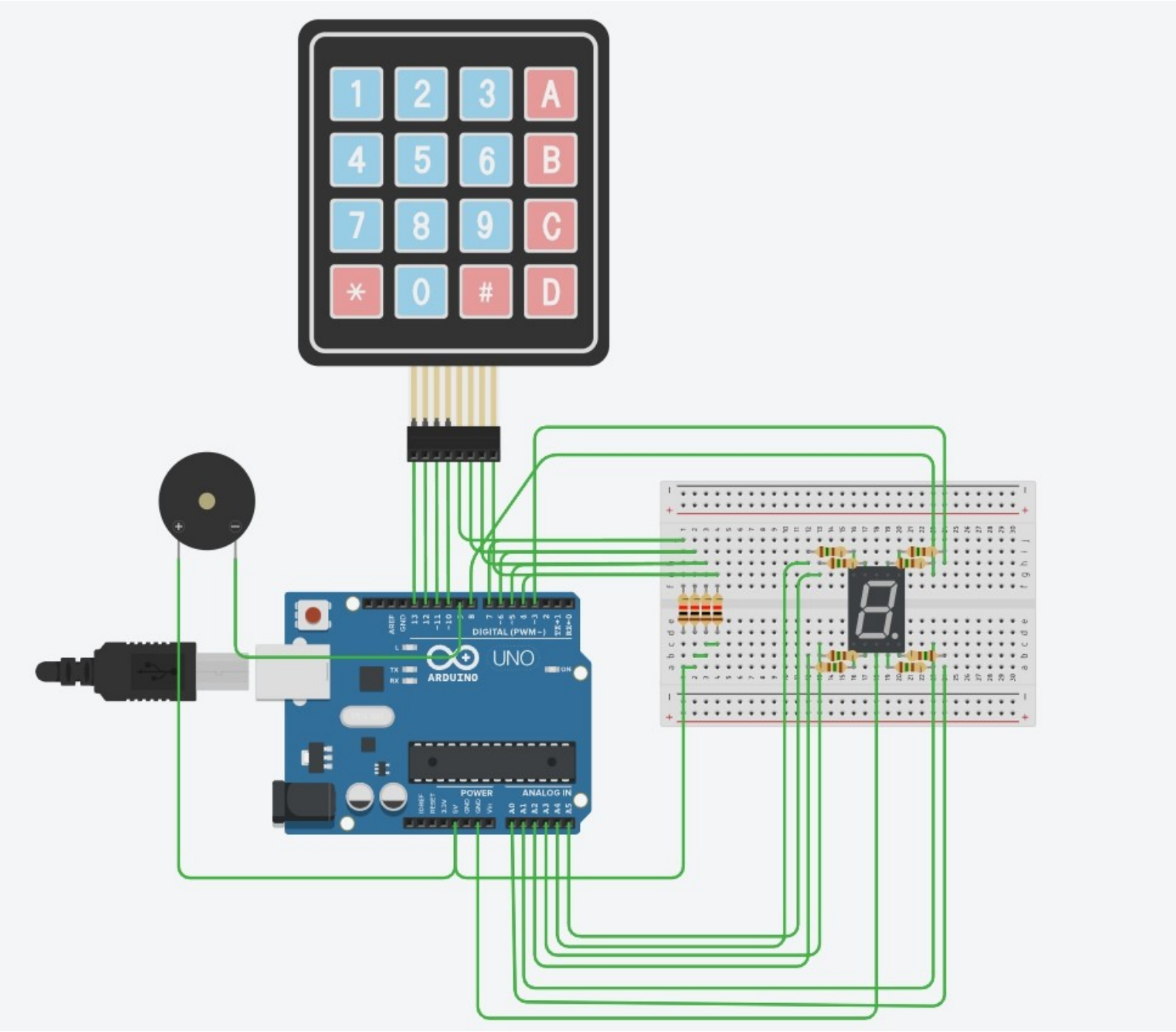

Рис 1. Схема принципиальная.

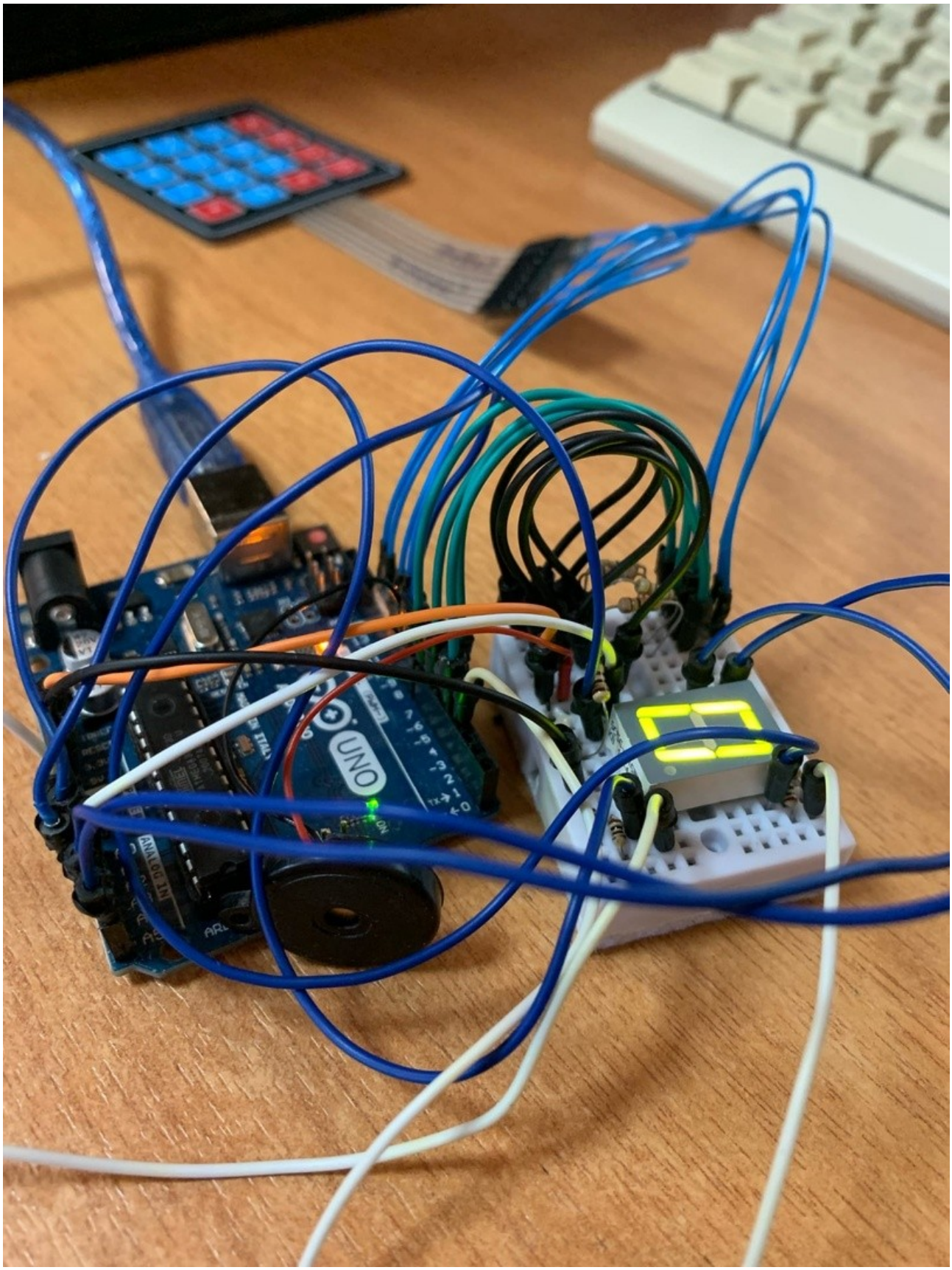

Рис 2. Схема принципиальная физическая.

Описание алгоритма программы

1. Инициализация и функционал клавиатуры

В данной работе используется матричная клавиатура, поэтому проинициализируем ее:

```
const char k4x4 [4][4] = {
 {'1', '2', '3', 'A'},
 {'4', '5', '6', 'B'},
 {'7', '8', '9', 'C'},
 {'*', '0', '#', 'D'}
};
```

Опишем функцию получения значения с клавиатуры:

```
char GetKey()
{
  static char old_a;
  char a = 0;
  for (int p = 0; p <= 3; p++)
  {
    PORTB &= ~(1<<(5-p));
    for (int m = 0; m <= 3; m++)
      if ((PIND & (1<<(7-m))) == 0)
      {
        a = k4x4[p][m];
      }

    PORTB |= 1<<(5-p);
  }

  if (a == old_a)
    return 0;
  old_a = a;

  return a;
}
```

Здесь:

- обходим (опрашиваем) строки и столбцы — цикл от 0 до 4.
- Устанавливается и в дальнейшем считывается сигнал (логическая 1) в выводах порта B .
- PIND — адрес входных выводов порта D.
- В выводах порта D также устанавливаем сигнал.
- После считывания и установки сигналов на строках и столбцах матрицы (на соответствующих выводах портов B и D) записываем выбранный символ.
- Чтобы использовать символ заново, перезаписываем ячейку памяти, где храниться наш символ.

По заданию ввод значений для индикатора происходит вручную.

Значения вводятся в основном цикле с помощью клавиатуры в пределах от 0 до 9, исключая символы A, B, C, *, #:

```
void loop() {
            <код>
    Serial.println(time);
    Serial.flush();
    print_number(time);
  }

  if (key == 'D') {
    start_timer();
    key = 0;
  }

  key = 0;
}
```

Для запуска процесса отсчета жмем клавишу «D».

2. Инициализация и функционал семисегментного индикатора

Ключевым компонентом в схеме является сам индикатор. Также проинициализируем его:

```
int numbers[10][7] = { 1, 1, 1, 1, 1, 0, 1,
                       0, 1, 1, 0, 0, 0, 0,
                       1, 1, 0, 1, 1, 1, 0,
                       1, 1, 1, 0, 1, 1, 0,
                       0, 1, 1, 0, 0, 1, 1,
                       1, 0, 1, 0, 1, 1, 1,
                       1, 0, 1, 1, 1, 1, 1,
                       1, 1, 1, 0, 0, 0, 0,
                       1, 1, 1, 1, 1, 1, 1,
                       1, 1, 1, 0, 1 ,1 ,1
      };
```

Каждая строка данного массива — есть соответствующая цифра на индикаторе, от 9 до 0, сверху — вниз.

Как и для клавиатуры, выделим порты под индикатор:

```
int indicator_port[7] = {8, 3, 15, 16, 17, 18, 19};
```

Данные порты настроены как выходные, на которые подан 0:

```
 for (int i = 0; i < 7; i++)
    pinMode(indicator_port[i], OUTPUT);

  for(int i=0;i<7;i++)
    digitalWrite(indicator_port[i], 0);
}
```

3. Инициализация и функционал пьезоизлучателя

Каждый раз при изменении показаний на индикаторе, «пищалка» издает звук определенной частоты и продолжительности.

```
void beep(int freq)
{
  OCR1A = int(16000000 / freq);
  TCCR1A = 0x40;
  _delay_ms(100);
  TCCR1A = 0x00;
}
```

Частоту определим исходя из номинала кварцевого генератора (16 МГц), при чем на каждое изменение показаний индикатора частота сигнала увеличивается на 50:

```
ISR (TIMER0_COMPA_vect) {
            <код>
    beep(freque);
    freque += 50;
            <код>
  }
```

Продолжительность сигнала на изменение показаний индикатора будет составлять 100 мс.

По окончании счета таймера до указанного значения, таймер сбрасывается, в последовательный порт выводится сообщение об окончании счета и производится звуковой сигнал частотой 400 и длительностью 1000 мс:

```
ISR (TIMER0_COMPA_vect) {
            <код>
    if (time == 0){
     stop_timer();
  }


void stop_timer()
{
  freque = 400;
  TCCR0B = 0b00000000;
  Serial.println("stop");
  Serial.flush();
  for(int i = 0; i < 3; i++)
    beep(1000);
  print_number(0);
}
```

Листинг

RGR.ino

```
int indicator_port[7] = {8, 3, 15, 16, 17, 18, 19};

    int numbers[10][7] = { 1, 1, 1, 1, 1, 0, 1,
                           0, 1, 1, 0, 0, 0, 0,
                           1, 1, 0, 1, 1, 1, 0,
                           1, 1, 1, 0, 1, 1, 0,
                           0, 1, 1, 0, 0, 1, 1,
                           1, 0, 1, 0, 1, 1, 1,
                           1, 0, 1, 1, 1, 1, 1,
                           1, 1, 1, 0, 0, 0, 0,
                           1, 1, 1, 1, 1, 1, 1,
                            1, 1, 1, 0, 1 ,1 ,1
                                   };

          volatile int interruptCount = 0;
                   int time = 1;
                    int freque;

                const char k4x4 [4][4] = {
                   {'1', '2', '3', 'A'},
                   {'4', '5', '6', 'B'},
                   {'7', '8', '9', 'C'},
                    {'*', '0', '#', 'D'}
                            };

                ISR (TIMER0_COMPA_vect) {
                     interruptCount++;
                if (interruptCount == 100) {
                          time -= 1;
                         beep(freque);
                         freque += 50;
                          deprint();
                     interruptCount = 0;
                    Serial.println(time);
                       Serial.flush();
                       if (time == 0){
                         stop_timer();
                               }
                             }
                    print_number(time);
                            }

                  void setup() {
                 Serial.begin(9600);


                 DDRB = 0b00111101;
                    DDRD = 0x00;
                   PORTB = 0xff;
                 DDRB |= (1 << 1);
                   TCCR1A = 0x00;
```

```
TCCR1B = 0b00001001;
TCCR0A = 0;
TCCR0B = 0;
OCR0A = 155;
TCCR0A = (1 << WGM01);
TIMSK0 |= (1 << OCIE0A);

for (int i = 0; i < 7; i++)
pinMode(indicator_port[i], OUTPUT);

for(int i=0;i<7;i++)
digitalWrite(indicator_port[i], 0);
}

void loop() {
char key = GetKey();

if (((key - 48) > 0) && (key < 'A')) {
time = int(key) - 48;
Serial.println(time);
Serial.flush();
print_number(time);
}

if (key == 'D') {
start_timer();
key = 0;
}

key = 0;
}

void print_number(int n)
{
for(int i = 0; i < 7; i++) {
digitalWrite(indicator_port[i], numbers[n][i]);
}
}

void deprint()
{
for(int i = 0; i < 7; i++) {
digitalWrite(indicator_port[i], 0);
}
}

void start_timer()
{
freque = 400;
TCCR0B = (1 << CS02) | (1 << CS00);
Serial.println("start");
Serial.flush();
}

void stop_timer()
{
freque = 400;
TCCR0B = 0b00000000;
```

```
   Serial.println("stop");
       Serial.flush();
  for(int i = 0; i < 3; i++)
           beep(1000);
        print_number(0);
              }

    void beep(int freq)
              {
OCR1A = int(16000000 / freq);
       TCCR1A = 0x40;
       _delay_ms(100);
       TCCR1A = 0x00;
              }

       char GetKey()
              {
     static char old_a;
        char a = 0;
for (int p = 0; p <= 3; p++)
              {
     PORTB &= ~(1<<(5-p));
 for (int m = 0; m <= 3; m++)
  if ((PIND & (1<<(7-m))) == 0)
                {
          a = k4x4[p][m];
                }

       PORTB |= 1<<(5-p);
              }

       if (a == old_a)
            return 0;
          old_a = a;

          return a;
              }
```