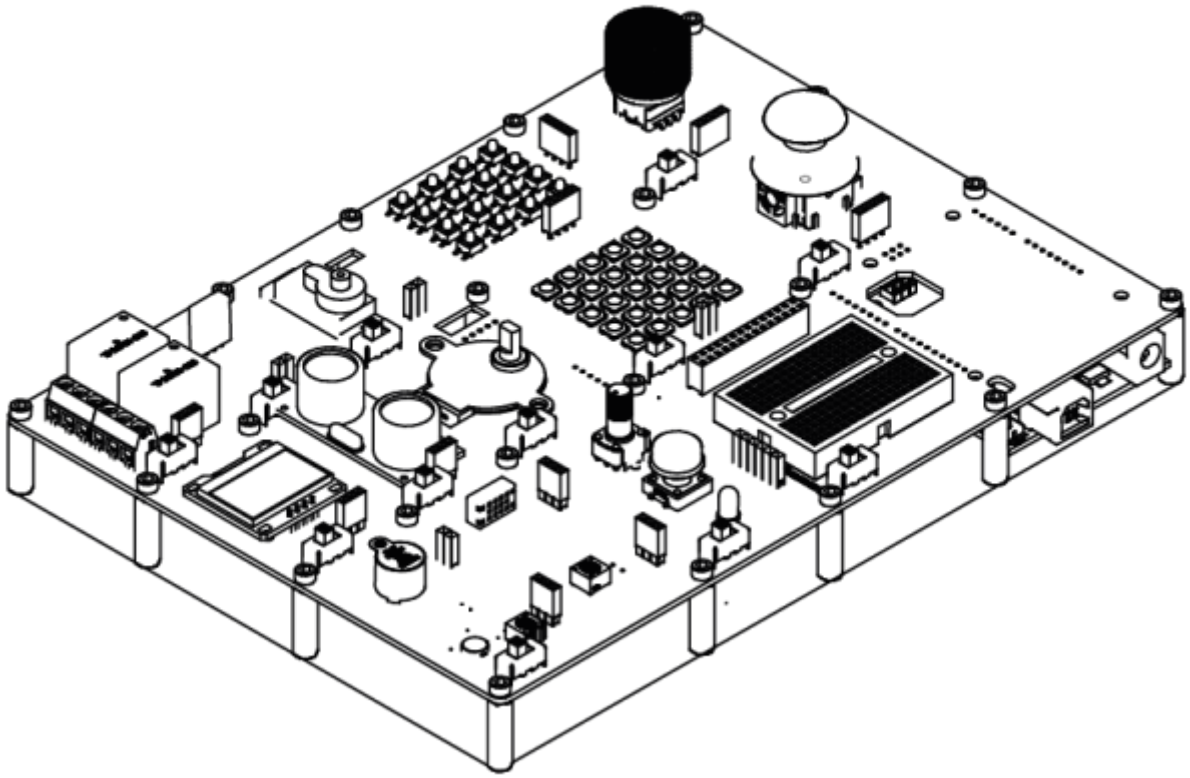
KM DEV BOARD: Инструкция, примеры использования и документация

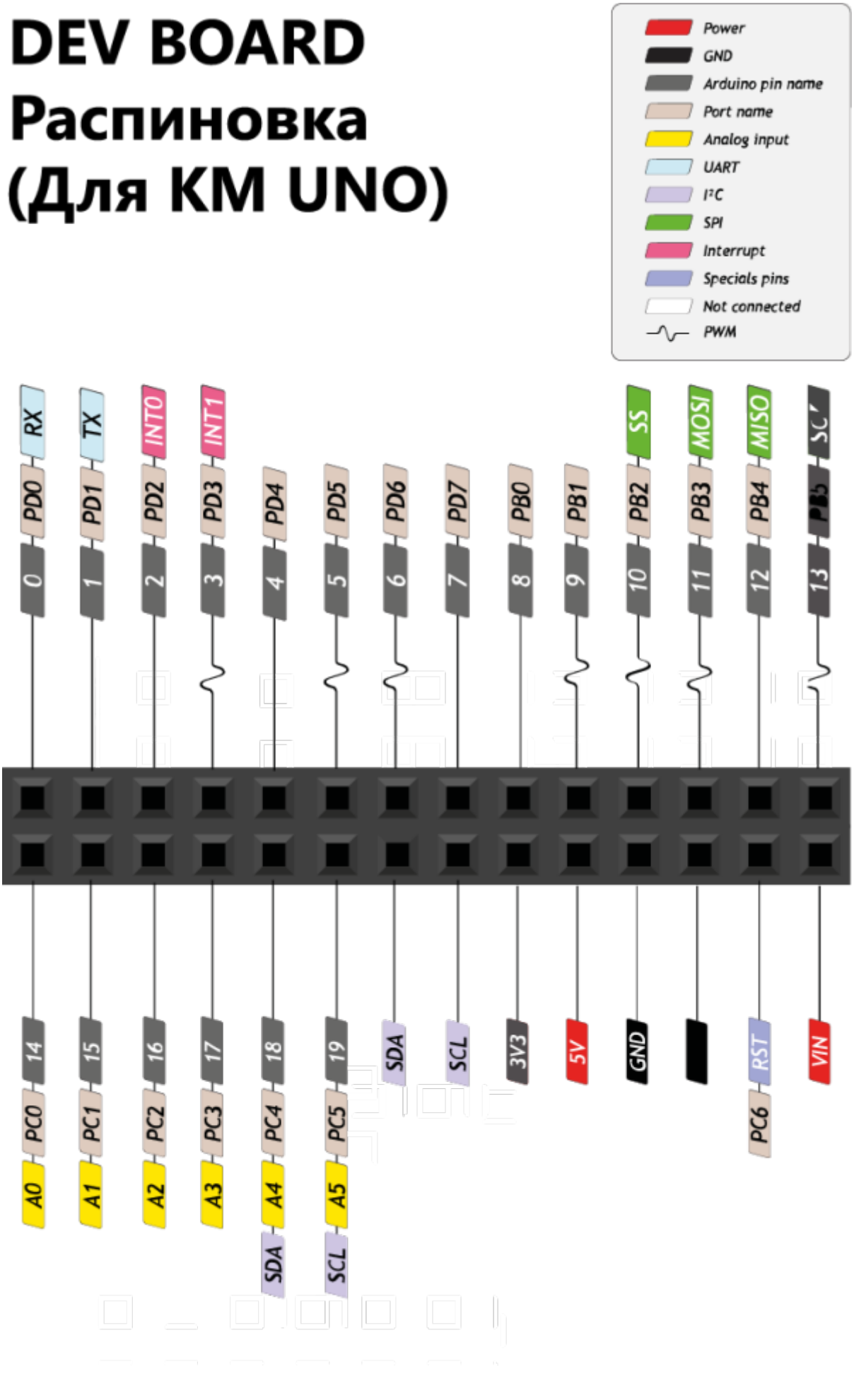
KM DEV BOARD - Плата содержащая 13 групп различных модулей, каждый из которых выполняет свою задачу. Логический уровень модулей 5В.



Для использования платы KM DEV BOARD необходимо:

1. На обратной стороне платы KM DEV BOARD необходимо найти разъём и подключить плату Arduino UNO или любую другую подобную, имеющую идентичные разъёмы.
2. Для обеспечения работоспособности платы необходимо подключить в разъём DC источник питания 9В.
3. Для загрузки прошивок в микроконтроллер используется кабель с разъёмом формата USB-B.
4. После переключаем движковый переключатель на модуле PROTO влево (сост. вкл.), если все шаги были выполнены правильно, то на этом модуле загорятся 3 светодиода синего цвета.
5. Далее можно прошить саму плату Arduino uno используя среду разработки Arduino_IDE, в которой можно написать собственный скетч, либо использовать готовые скетчи для каждого модуля, примеры которых приведены в данной статье.
6. С помощью комплекта перемычек подключить модуль Proto к остальным модулям. Все пины для подключения описаны в разделе с соответствующим модулем.

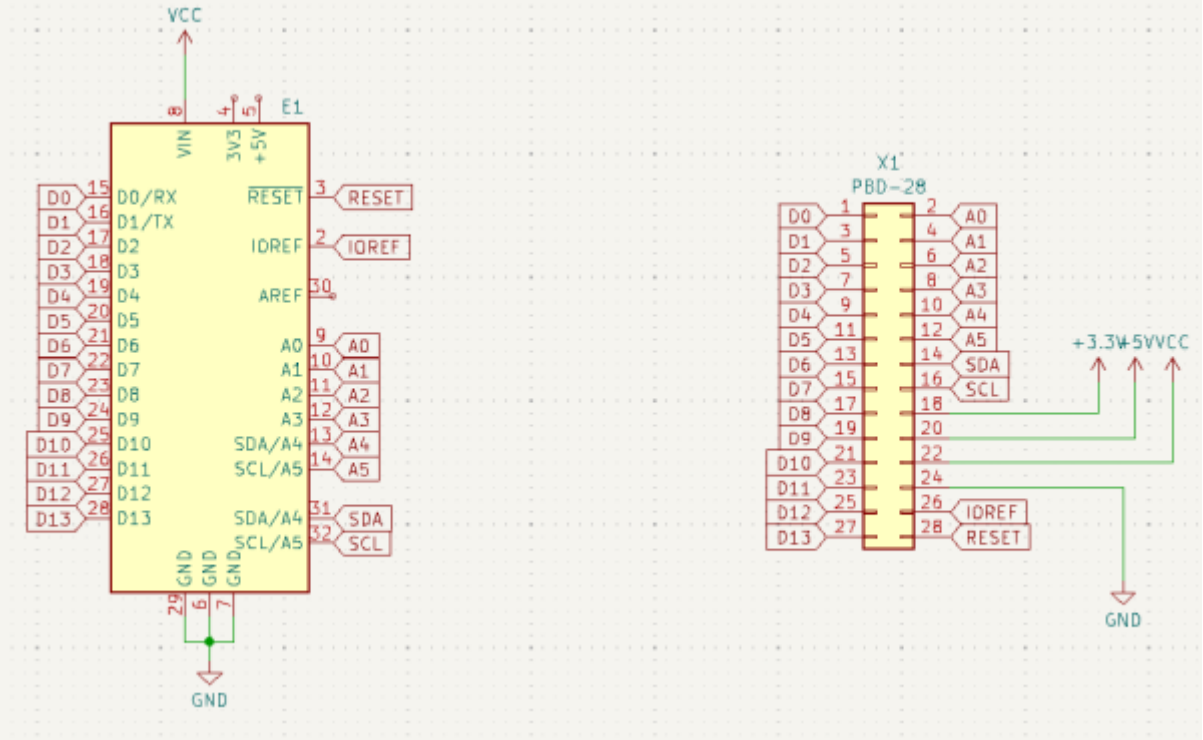
DEV BOARD
Распиновка
(Для KM UNO)



Модуль PROTO

PROTO – представляет собой модуль-макетную плату с аналоговыми (A0-A5) и цифровыми (0-13) выводами. Имеет переключатель питания. Имеет выводы: Питания – 3.3V и 5V. Этот модуль используется для подключения остальных модулей.

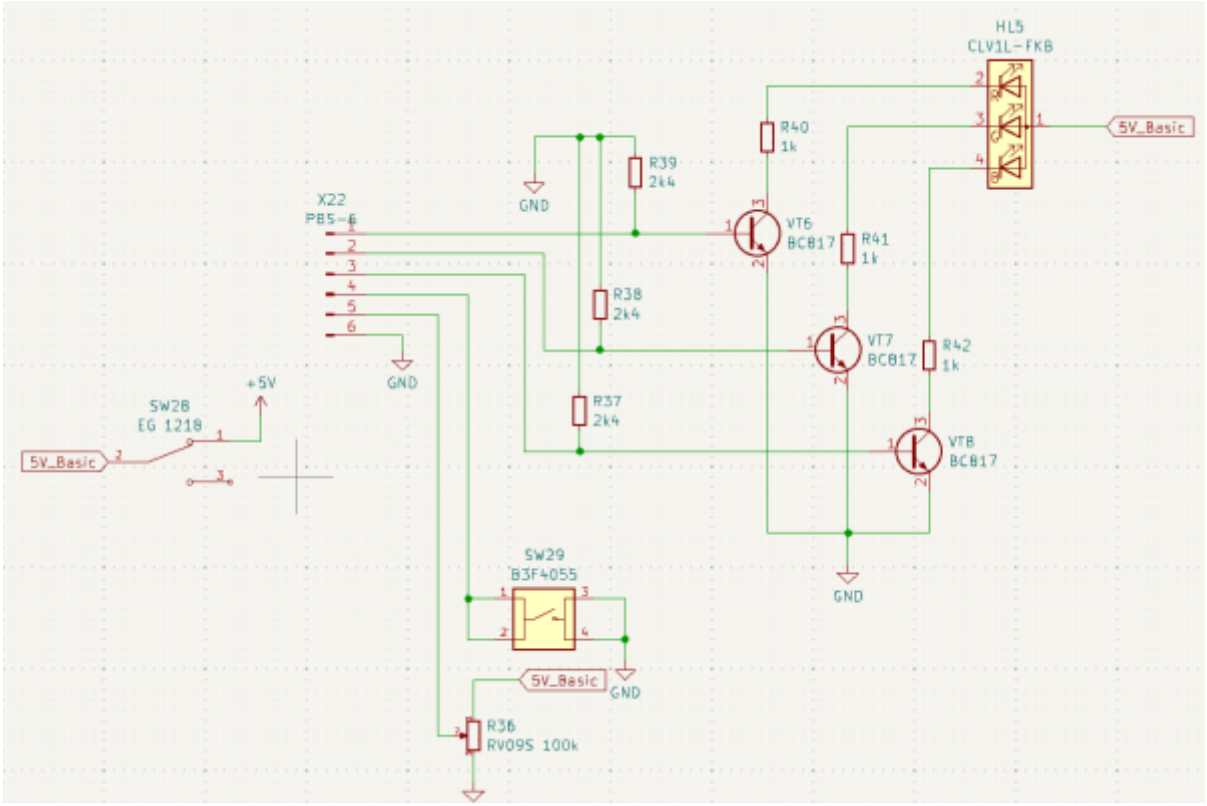
- SDA – линия данных
- SCL – линия синхронизации
- RESET – перезагрузка
- Vin – питание от внешнего источника



Модуль START

START – представляет собой модуль с потенциометром, кнопкой и светодиодом для изучения аналоговых и цифровых портов. Имеет переключатель питания. Имеет выводы:

- BLUE, GREEN, RED – вывод катода
- BUTTON – вывод кнопки
- POT – вывод потенциометра
- G – земля, для запуска модуля не требуется



Пример прошивки

Start.ino

```
const int PIN_POTENTIOMETER = A1; //Потенциометр
const int PIN_BUTTON = 3; // Кнопка
const int PIN_LIGHTDIODE = 4; // Светодиод

char messageBuffer[255];
bool flag = false;
int button = 0;

void setup() {
  Serial.begin(9600);
  pinMode(PIN_BUTTON, INPUT_PULLUP);
  pinMode(PIN_POTENTIOMETER, INPUT);
  pinMode(PIN_LIGHTDIODE, OUTPUT);
}

void loop() {
  digitalWrite(PIN_LIGHTDIODE, HIGH);    // зажигаем светодиод
  int val = analogRead(PIN_POTENTIOMETER);
  bool btnState = !digitalRead(PIN_BUTTON);
  if (btnState && !flag) {   // обработчик нажатия
    flag = true;
    button = 1;
  }
  if (!btnState && flag) {   // обработчик отпускания
```

```
    flag = false;
    button = 0;
  }
  sprintf(messageBuffer, "Potentiometer: = %d, Button: = %d", val,
button);
  Serial.println(messageBuffer);
  delay(50);
}
```

Модуль Sensor

Sensor – представляет собой модуль с датчиками DHT11 и BME280, фоторезистор и звуковой излучатель. Имеет выводы:

Для датчика DHT11:

- SDA – линия данных I2C
- SCL – линия синхронизации I2C
- G – земля, для подключения не требуется

Для датчика BME280:

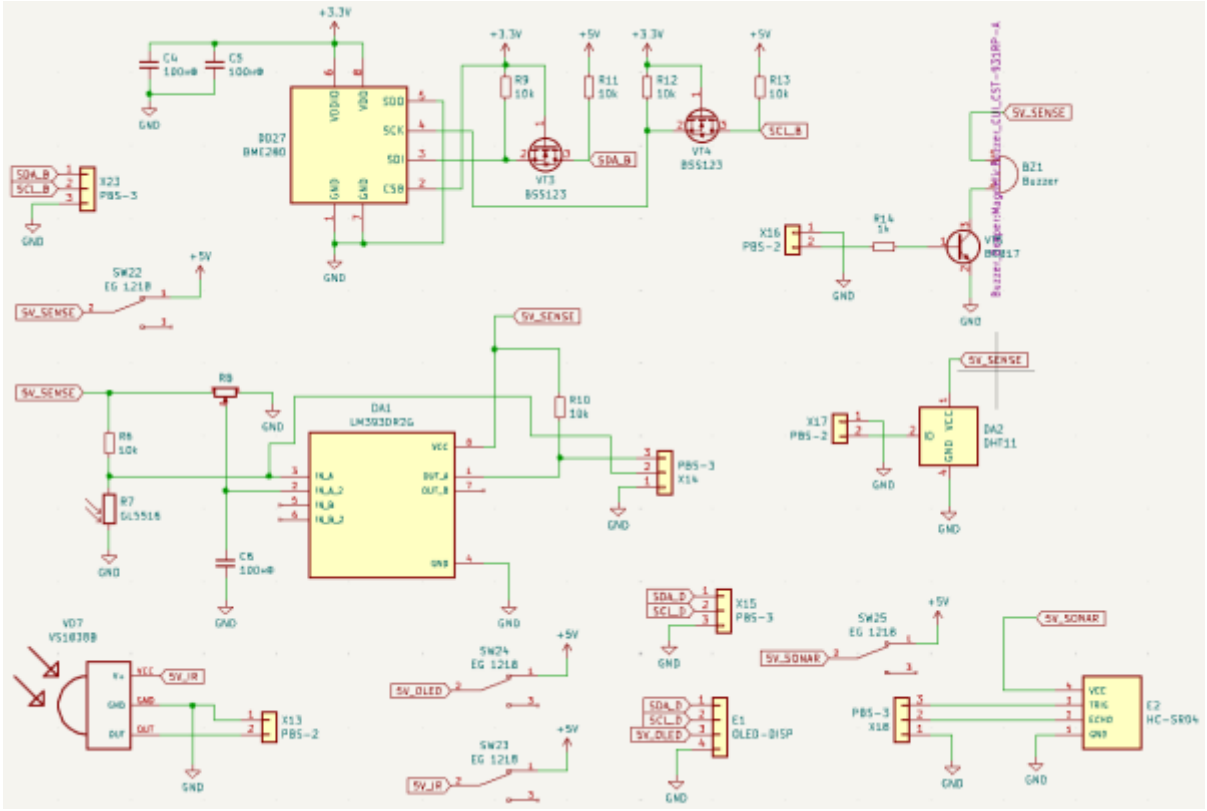
- SDA – линия данных I2C
- SCL – линия синхронизации I2C
- G – земля, для подключения не требуется

Для фоторезистора:

- D – линия передачи цифровых данных
- A – линия передачи аналоговых данных
- G – земля, для подключения не требуется

Для звукового излучателя:

- S – вывод для управляющего сигнала
- G – земля, для подключения не требуется



Пример прошивки

sensor.ino

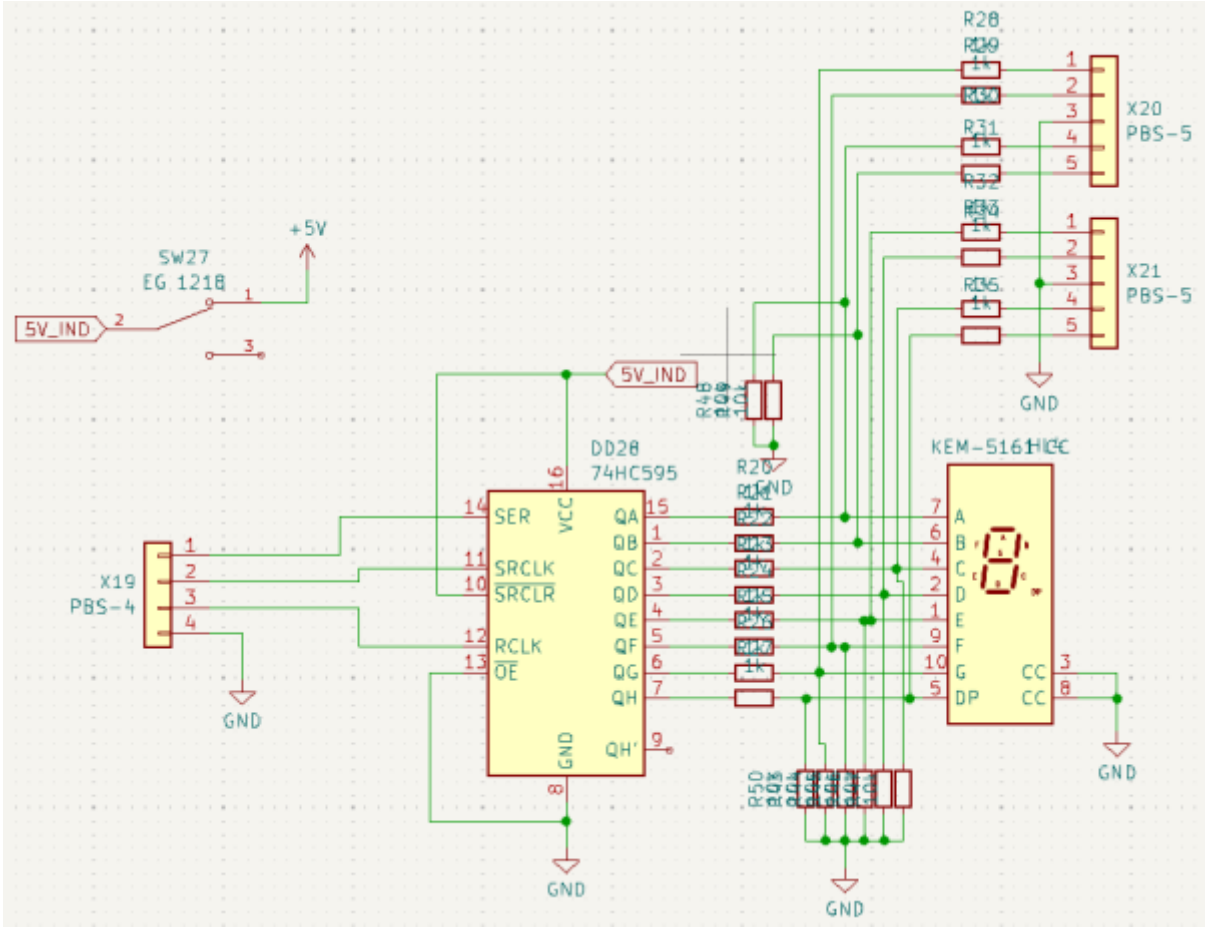
```
#include "DHT.h" // библиотека датчика влажности

#define PHOTO_PIN 11
#define ZoomerPIN 12 // пин для зумера

DHT dht(DHTPIN, DHT11); //Инициация датчика (текущий)

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  dht.begin(); // инициализация DH11
}

void loop() {

  float h = dht.readHumidity(); //Измеряем влажность
  float t = dht.readTemperature(); //Измеряем температуру

    if (isnan(h) == false || isnan(t) == false) // Проверка. Если не удается считать показания, выводится «Ошибка считывания», и программа завершает работу
    {
      Serial.print("Влажность: ");
      Serial.print(h);
```

```
      Serial.print(" %\t");
      Serial.print("Температура: ");
      Serial.print(t);
      Serial.println(" *C "); //Вывод показателей на экран
    }

    //Zoomer
    if (!digitalRead(BTTN_PIN))
    {
      tone(ZoomerPIN, 1500);
      delay(100);
    }

    //Zoomer
    noTone(ZoomerPIN);

  int photo = analogRead(PHOTO_PIN);
  Serial.println(photo );      // выводим показания фотодиода

  delay(1000);
}
```

Модуль 7-DIGIT

7-DIGIT – представляет собой семисегментный индикатор с прямым доступам к выводам индикатора и сдвиговым регистром. Имеет выводы:

- 8 выводов вокруг семисегментного индикатора для прямого подключения
- DS – вход данных сдвигового регистра
- SHCP – тактовый вход сдвигового регистра
- STCP – тактовый вход регистра хранения
- G – земля, для запуска модуля не требуется



Пример прошивки

7-DIGIT.ino

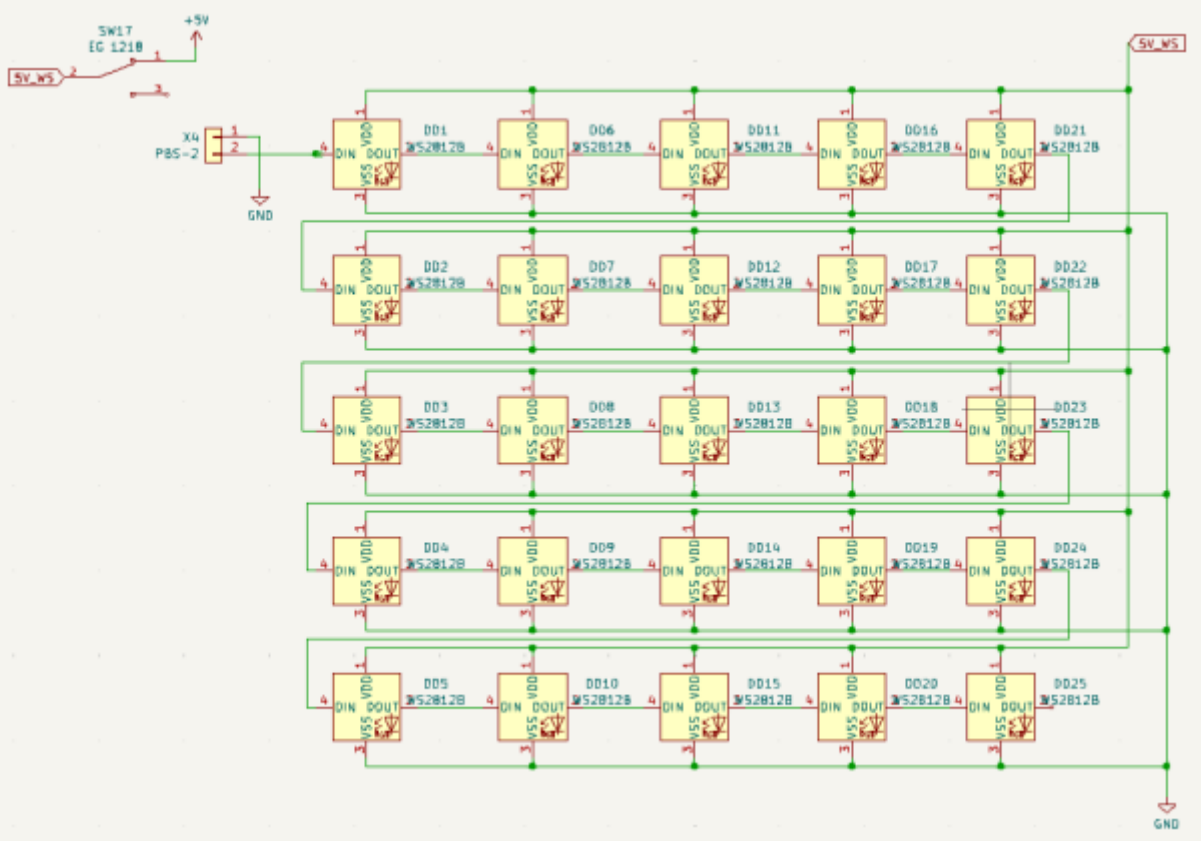
```
#define DATA 2 // DS(7-DIGIT)
#define CLOCK 3 //SHCP(7-DIGIT)
#define LATCH 4 // STCP(7-DIGIT)

int Number;// цифра семисегментика(.)
/*
 * . | 0b00000001
 * 0 | 0b11111100
 * 1 | 0b01100000
 * 2 | 0b11011010
 * 3 | 0b11110010
 * 4 | 0b01100110
 * 5 | 0b10110110
 * 6 | 0b10111110
 * 7 | 0b11100000
 * 8 | 0b11111110
 * 9 | 0b11110110
 */
void setup() {
  Serial.begin(115200);
   //7-DIGIT
```

```
   pinMode(CLOCK, OUTPUT);
   pinMode(DATA, OUTPUT);
   pinMode(LATCH, OUTPUT);
   digitalWrite(LATCH, HIGH);
}

void loop() {
  while (true)
  {
    Number = 0b00000001; // .
    shiftOut(DATA, CLOCK, LSBFIRST, Number); //7-DIGIT
    delay(1000);
    Number = 0b11111100; // 0
    shiftOut(DATA, CLOCK, LSBFIRST, Number); //7-DIGIT
    delay(1000);
    Number = 0b01100000; // 1
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
    Number = 0b11011010; // 2
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
    Number = 0b11110010; // 3
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
    Number = 0b01100110; // 4
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
    Number = 0b10110110; // 5
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
    Number = 0b10111110; // 6
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
    Number = 0b11100000; // 7
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
    Number = 0b11111110; // 8
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
    Number = 0b11110110; // 9
    shiftOut(DATA, CLOCK, LSBFIRST, Number);
    delay(1000);
  }
}
```

Модуль LED MATRIX

LED MATRIX – представляет собой светодиодную матрицу с адресными светодиодами, позволяющую зажигать отдельно каждый светодиод. Имеет выводы:

- S – сигнальный вывод
- G – земля, для запуска модуля не требуется



Пример прошивки

LedMatrix.ino

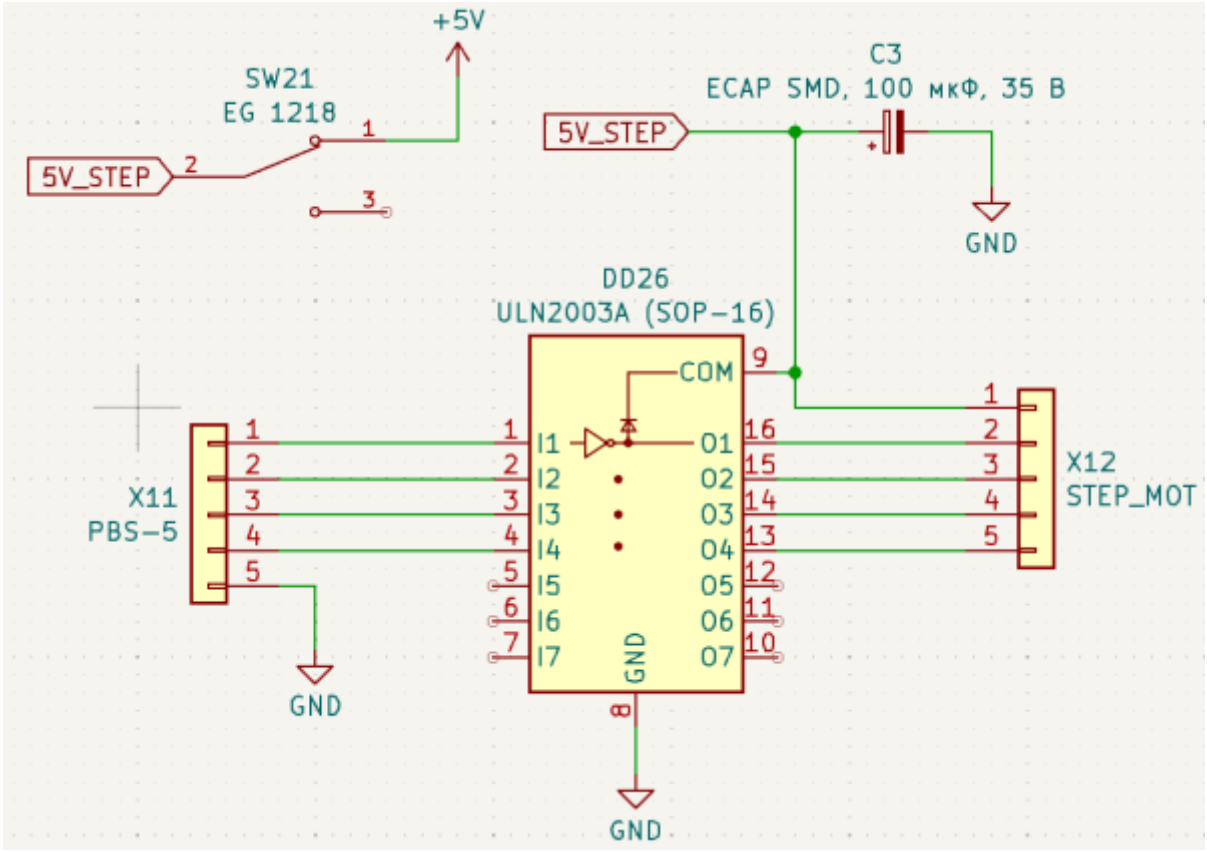
```
#include "Adafruit_NeoPixel.h" // подключаем библиотеку для светодиодной ленты

#define LENTAPIN  5// указываем пин для подключения ленты
#define NUMPIXELS 25  // указываем количество светодиодов в ленте

// создаем объект strip с нужными характеристиками
Adafruit_NeoPixel strip (NUMPIXELS, LENTAPIN, NEO_GRB + NEO_KHZ800);

void setup() {
   strip.begin();             // инициализируем ленту
   strip.setBrightness(50);   // указываем яркость светодиодов (максимум 255)
}
```

```
void loop() {
   for (int i = 0; i < NUMPIXELS; i++)
   {
   strip.setPixelColor(i, strip.Color(255, 0, 0));    // включаем красный цвет на светодиоде
   strip.show();    // отправляем сигнал на ленту
  }
}
```

Модуль STEPPER

STEPPER – представляет собой модуль шагового двигателя. Имеет выводы:

- Белый 5 пин – для подключения самого шагового двигателя
- S1 – сигнальный вывод 1
- S2 – сигнальный вывод 2
- S3 – сигнальный вывод 3
- S4 – сигнальный вывод 4
- G – земля, для запуска не требуется



Пример прошивки

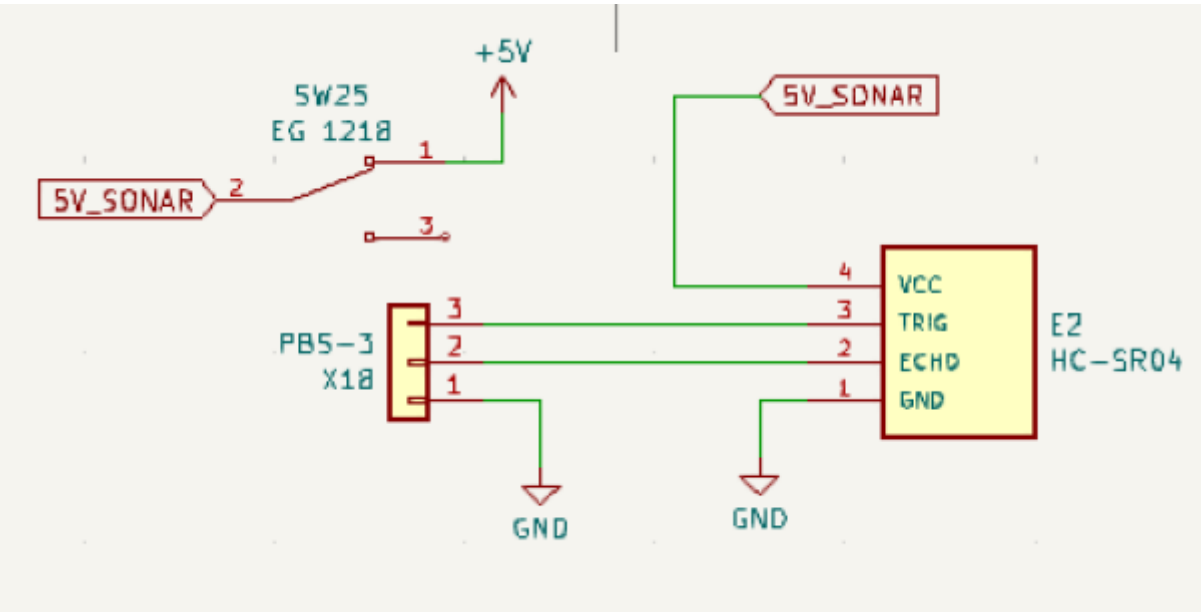
Stepper.ino

```
#include <Stepper.h>

const int stepsPerRevolution = 200;  // change this to fit the number
of steps per revolution
// for your motor

// initialize the stepper library on pins 8 through 11:
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

void setup() {
  // set the speed at 60 rpm:
  myStepper.setSpeed(60);
  // initialize the serial port:
  Serial.begin(9600);
}

void loop() {
  // step one revolution  in one direction:
  Serial.println("clockwise");
  myStepper.step(stepsPerRevolution);
  delay(500);

  // step one revolution in the other direction:
  Serial.println("counterclockwise");
  myStepper.step(-stepsPerRevolution);
  delay(500);
}
```

Модуль US

US – представляет собой модуль-ультразвуковой датчик расстояния. Имеет выводы:

- Trig – выход сигнала входа
- Echo – вывод сигнала выхода
- G – земля, для запуска не требуется



Пример прошивки

US.ino

```
#define PIN_TRIG 12
#define PIN_ECHO 11

long duration, cm;

void setup() {

  // Инициализируем взаимодействие по последовательному порту

  Serial.begin (9600);
  //Определяем вводы и выводы
  pinMode(PIN_TRIG, OUTPUT);
  pinMode(PIN_ECHO, INPUT);
}

void loop() {

  // Сначала генерируем короткий импульс длительностью 2-5 микросекунд.

  digitalWrite(PIN_TRIG, LOW);
  delayMicroseconds(5);
  digitalWrite(PIN_TRIG, HIGH);

  // Выставив высокий уровень сигнала, ждем около 10 микросекунд. В этот момент датчик будет посылать сигналы с частотой 40 КГц.
  delayMicroseconds(10);
  digitalWrite(PIN_TRIG, LOW);

  //  Время задержки акустического сигнала на эхолокаторе.
```

```
  duration = pulseIn(PIN_ECHO, HIGH);

  // Теперь осталось преобразовать время в расстояние
  cm = (duration / 2) / 29.1;

  Serial.print("Расстояние до объекта: ");
  Serial.print(cm);
  Serial.println(" см.");

  // Задержка между измерениями для корректной работы скеча
  delay(250);
}
```

Модуль OLED

OLED – представляет собой модуль oLed дисплея для вывода информации. Имеет выводы:

- SDA – вывод данных
- SCA – вывод синхронизации
- G – земля, для подключения не требуется

Пример прошивки

Oled.ino

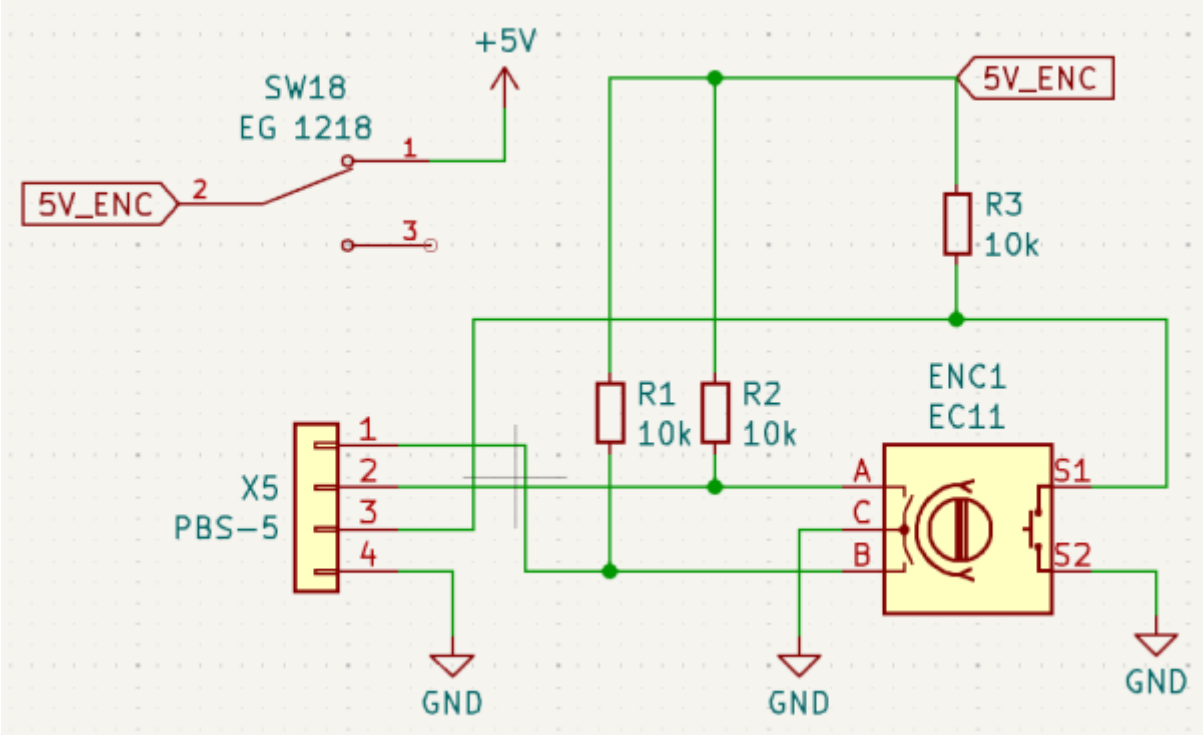
```
// библиотеки для работы с OLED экраном Arduino IDE
#include "Wire.h"
#include "Adafruit_GFX.h"
#include "Adafruit_SSD1306.h"

Adafruit_SSD1306 display(128, 64, &Wire, 4); // указываем размер экрана в пикселях

void setup() {
   display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // указываем адрес устройства на шине
   display.clearDisplay();
   display.setTextSize(1, 2); // указываем размер шрифта
   display.setTextColor(SSD1306_WHITE); // указываем цвет надписи

   display.setCursor(30, 10);
   display.println("ARDUINO");
   display.display();
   delay(3000);
```

```
   display.clearDisplay(); // очищаем экран
}

void loop() {
   display.setTextSize(3); // указываем размер шрифта
   display.setTextColor(SSD1306_WHITE); // указываем цвет надписи

   for (int i = 0; i <= 100; i++) {
        display.setCursor(20, 10);
      display.println(i);
      display.display();
      delay(100);
      display.clearDisplay(); // очищаем экран
   }

}
```

Модуль ENCODER

ENCODER – представляет собой модуль преобразователь угловых перемещений. Имеет выводы:

- DT – вывод Дата
- CLK – вывод синхронизации
- B – вывод кнопки
- G – земля, для подключения не требуется



Пример прошивки

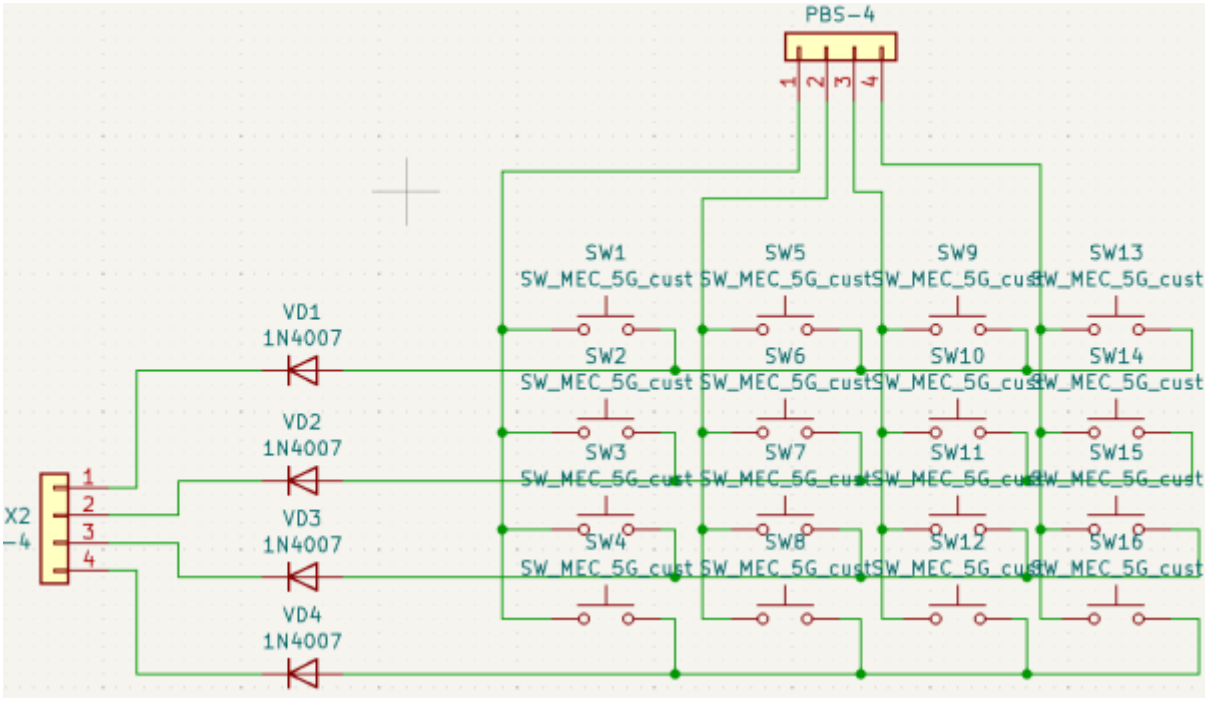
Encoder.ino

```
#define btn_long_push 1000   // Длительность долинного нажатия кнопки
(encoder)
  volatile uint8_t lastcomb=7, enc_state, btn_push=0;
  volatile int enc_rotation=0, btn_enc_rotate=0;
  volatile boolean btn_press=0;
  volatile uint32_t timer;

void setup() {
  Serial.begin(115200);
   //Encoder
   pinMode(A1,INPUT_PULLUP); // ENC-A(DT)
   pinMode(A2,INPUT_PULLUP); // ENC-B(CLK)
   pinMode(A3,INPUT_PULLUP); // BUTTON
   PCICR =  0b00000010; // PCICR |= (1<<PCIE1); Включить прерывание
PCINT1
   PCMSK1 = 0b00001110; // Разрешить прерывание для  A1, A2, A3
}

void loop() {
  //Encoder
  switch (enc_state)
    {
  case 1:   {
            Serial.print("Вращение без нажатия ");
            Serial.println(enc_rotation);
            }
   break;

  case 2:  {
            Serial.print("Вращение с нажатием ");
            Serial.println(btn_enc_rotate);
           }
    break;

  case 3: Serial.println("Нажатие кнопки ");
  break;

  case 4: Serial.println("Длинное нажатие кнопки ");
  break;
    }
  enc_state=0; //обнуляем статус энкодера
}
//------------------------------------------------
ISR (PCINT1_vect) //Обработчик прерывания от пинов  A1, A2, A3
{
  {
```

```
  uint8_t comb = bitRead(PINC, 3) << 2 | bitRead( PINC, 2)<<1 |
bitRead(PINC, 1); //считываем состояние пинов энкодера и кнопки

 if (comb == 3 && lastcomb == 7) btn_press=1; //Если было нажатие кнопки,
то меняем статус

 if (comb == 4)                              //Если было промежуточное
положение энкодера, то проверяем его предыдущее состояние
 {
    if (lastcomb == 5) --enc_rotation; //вращение по часовой стрелке
    if (lastcomb == 6) ++enc_rotation; //вращение против часовой
    enc_state=1;                            // был поворот энкодера
    btn_enc_rotate=0;                       //обнулить показания вращения с
нажатием
  }

   if (comb == 0)                           //Если было промежуточное положение
энкодера и нажатие, то проверяем его предыдущее состояние
   {
    if (lastcomb == 1) --btn_enc_rotate; //вращение по часовой стрелке
    if (lastcomb == 2) ++btn_enc_rotate; //вращение против частовой
    enc_state=2;                           // был поворот энкодера с нажатием
    enc_rotation=0;                        //обнулить показания вращения без
нажатия
    btn_press=0;                            //обнулить показания кнопки
   }

   if (comb == 7 && lastcomb == 3 && btn_press) //Если было отпускание
кнопки, то проверяем ее предыдущее состояние
   {
    if (millis() - timer > btn_long_push)          // проверяем сколько
прошло миллисекунд
    {
      enc_state=4;                                  // было длинное нажатие
    } else {
            enc_state=3;                      // было нажатие
           }
      btn_press=0;                          //обнулить статус кнопки
    }

  timer = millis();                         //сброс таймера
  lastcomb = comb;                          //сохраняем текущее состояние
энкодера
//----------------------------------------------
  }
}
```

Модуль BUTTON MATRIX

BUTTON MATRIX – представляет собой кнопочный модуль 4x4. Подписанные выводы образуют сетку.



Имеет выводы:

- 1 – вывод матрицы кнопок
- 2 – вывод матрицы кнопок
- 3 – вывод матрицы кнопок
- 4 – вывод матрицы кнопок
- A – вывод матрицы кнопок
- B – вывод матрицы кнопок
- C – вывод матрицы кнопок
- D – вывод матрицы кнопок

	A	B	C	D
1	1A	1B	1C	1D
2	2A	2B	2C	2D
3	3A	3B	3C	3D
4	4A	4B	4C	4D

Пример прошивки

ButtonMatrix.ino

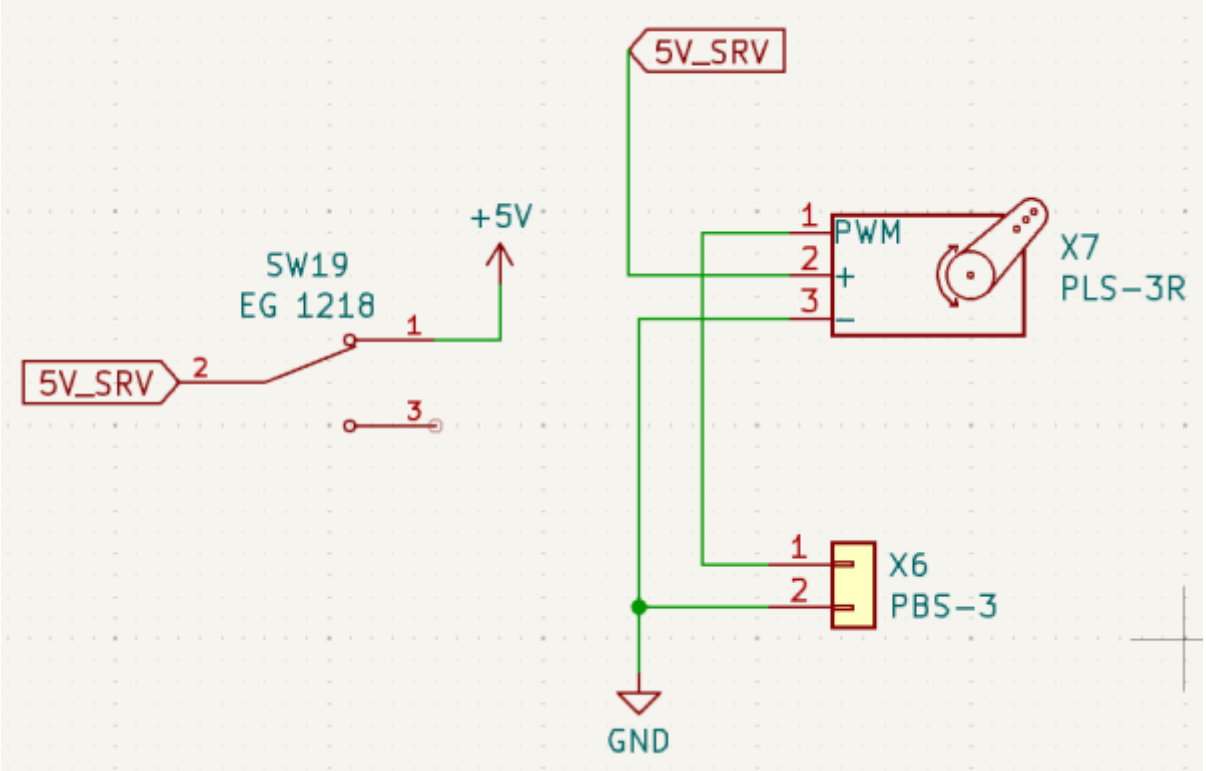
```
#include <Keypad.h>
```

```
const byte ROWS = 4; //four rows
const byte COLS = 4; //four columns
char keys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
byte rowPins[ROWS] = {9, 8, 7, 6}; //connect to the row pinouts of the
keypad
byte colPins[COLS] = {5, 4, 3, 2}; //connect to the column pinouts of
the keypad

Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS
);

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

void loop() {
  char key = keypad.getKey();

  if (key) {
    Serial.println(key);
  }
}
```

Модуль SERVO

SERVO – представляет из себя модуль сервопривода. Имеет выводы:

- S – сигнальный вывод
- G – земля, для подключения не требуется



Пример прошивки

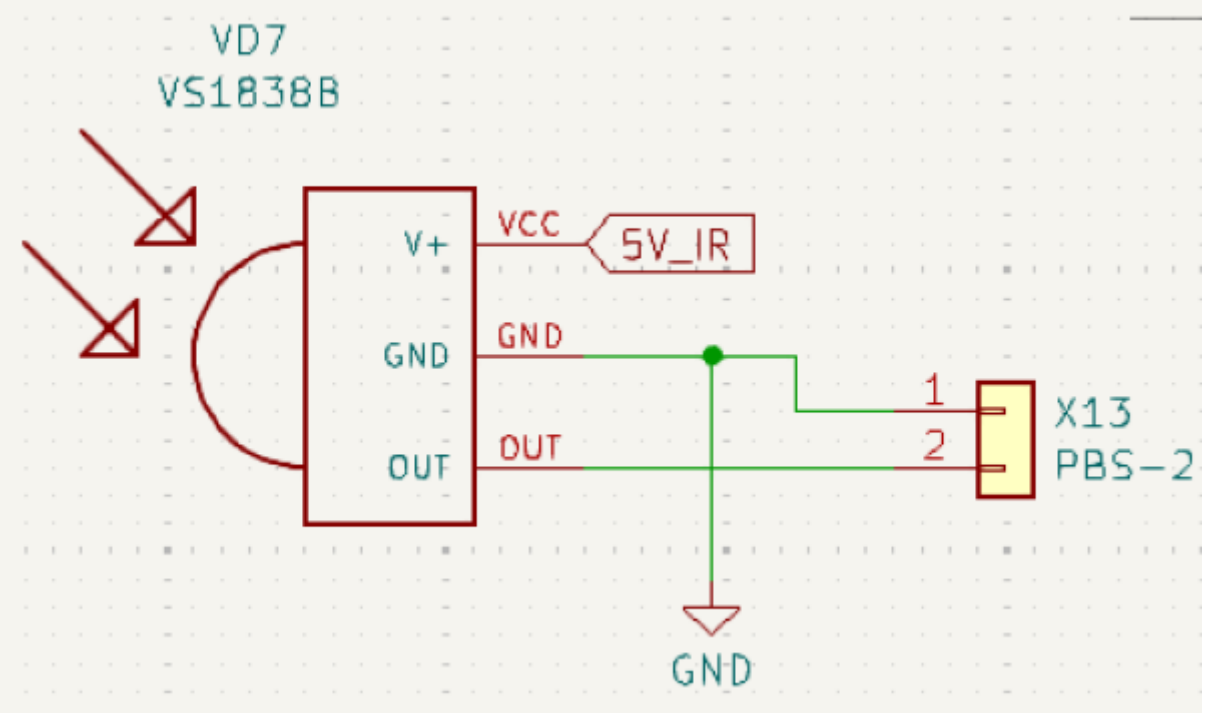
Servo.ino

```
#include <Servo.h>

Servo servo;  // Создаем объект
void setup() {
  servo.attach(9);    // Указываем объекту класса Servo, что серво присоединен к пину 9
  servo.write(0);    // Выставляем начальное положение
  }

void loop() {
  servo.write(0); // Поворачиваем серво на 90 градусов
  delay(1000);
  servo.write(180);
  delay(1000);
  servo.write(360);
  delay(1000);
}
```

Модуль IR

IR – представляет собой инфракрасный модуль. Имеет выводы:

- S – сигнальный вывод
- G – земля, для подключения не требуется



Пример прошивки

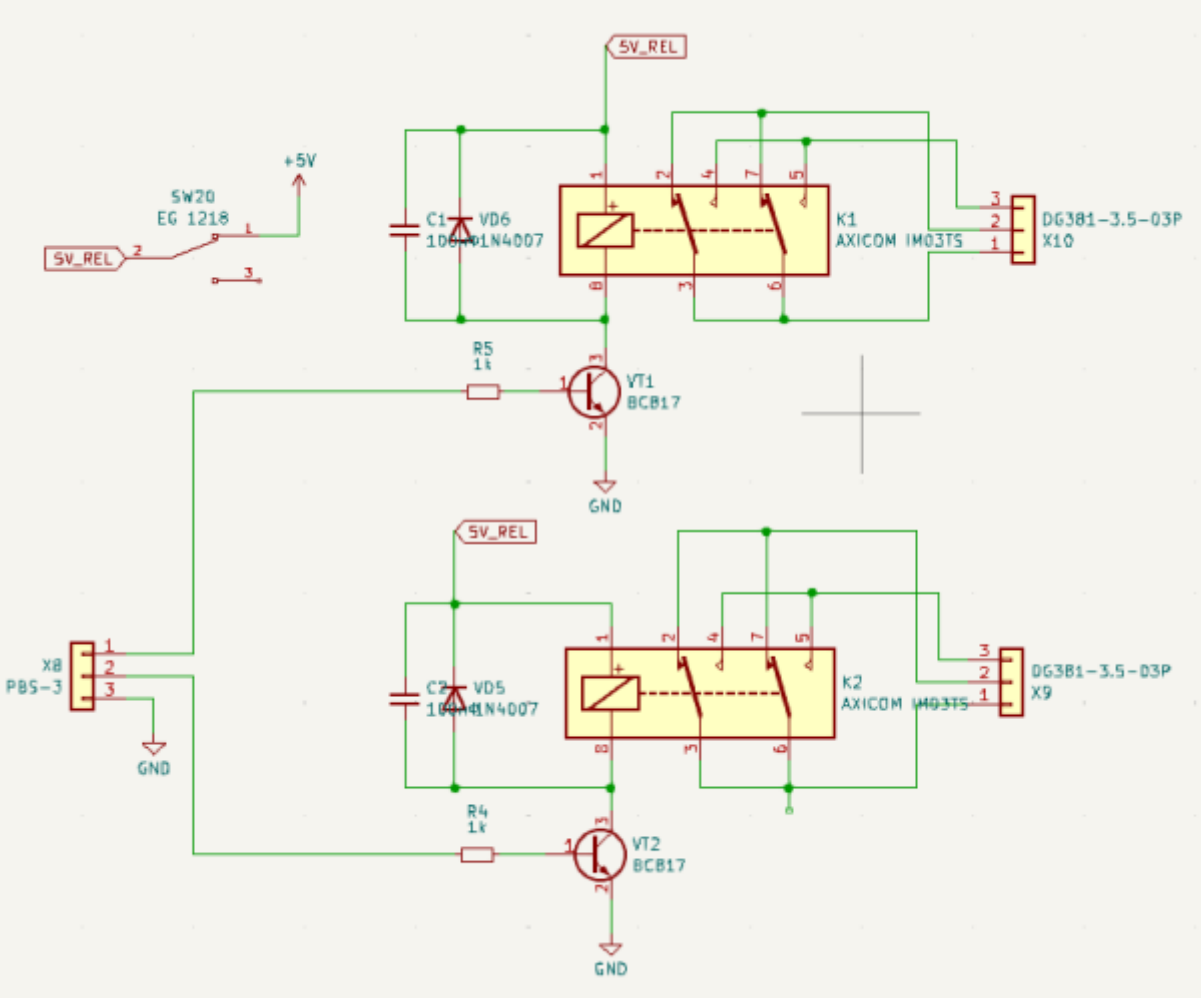
IR.ino

```
#include <IRremote.h> //подключение библиотеки
const int IR_pin = 8; //пин подключения инфракрасного приемника
decode_results results; //переменная для хранения результата приема

IRrecv irreceiver(IR_pin); //создание объекта приемника

void setup()
{
  Serial.begin(9600); //инициализация Serial - порта
  irreceiver.enableIRIn(); //инициализация приемника
}

void loop()
{
  if (irreceiver.decode(&results)) { //если что-то пришло
    Serial.println(results.value); //сообщить значение приема в монитор
    irreceiver.resume(); //возобновление работы ИК приемника
  }
}
```

Модуль RELAY

RELAY – представляет собой модуль с реле. Имеет выводы:

- S1 – вывод первого реле
- S2 – вывод второго реле
- G – земля, для подключения не требуется



Пример прошивки

Relay.ino

```
int relay_1 = 4;
int relay_2 = 7;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  pinMode(relay_1, OUTPUT);
```

```
  pinMode(relay_2, OUTPUT);

}

void loop() {

  digitalWrite(relay_1, HIGH);
  digitalWrite(relay_2, HIGH);

  Serial.println("All relays ON");

  delay(1000);

  digitalWrite(relay_1, LOW);
  digitalWrite(relay_2, LOW);
  Serial.println("All relays OFF");

  delay(1000);
}
```