МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ВГТУ)

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ

Робототехника в автоматизированном производстве

Воронеж 2022

СОДЕРЖАНИЕ

1. Управление сервоприводами манипулятора с помощью джойстика

Цель работы: разработать программное обеспечение для системы управления сервоприводами манипулятора

1.1. Краткие теоретические сведения

Сервопривод

Сервопривод (Рис. 1) является важным элементом при конструировании различных роботов и механизмов. Это точный исполнитель, который имеет обратную связь, позволяющую точно управлять движениями механизмов. Другими словами, получая на входе значение управляющего сигнала, сервомотор стремится поддерживать это значение на выходе своего исполнительного элемента.

Рис. 1 Устройство сервопривода (Arduino)

Как видно из Рис. 1 от сервопривода идут три провода:

- + U (красный),
- GND (черный или коричневый)
- Управляющий вход (желтый, оранжевый или белый).

На управляющий вход сервопривода подается импульсный сигнал с периодом T=20mc. Угол поворота задается временем импульса: при 1 mc он составляет 0^0 градусов, а при 2 mc – 180^0, а в промежутке значение от 0 до 180. (Рис. 2)

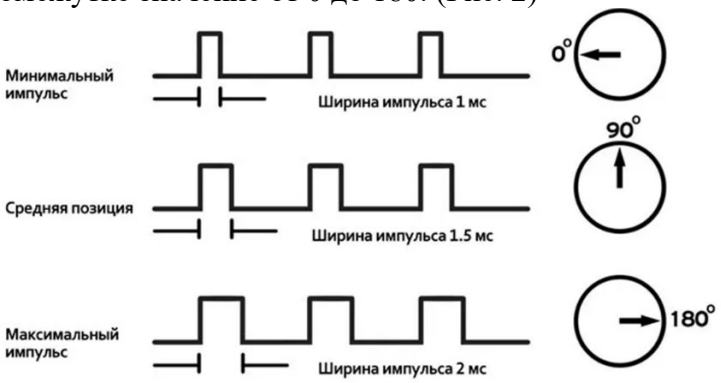

Рис. 2 Регулирование угла поворота вала сервопривода

Ниже приведен пример кода с комментариями, иллюстрирующий управление сервоприводом.

```
#include <Servo.h> // подключаем библиотеку для работы с сервоприводом

Servo servo1; // объявляем переменную servo типа "servo1"

void setup() {
  servo1.attach(11); // привязываем сервопривод к аналоговому выходу 11
}

void loop() {
servo1.write(0); // ставим угол поворота под 0
delay(2000); // ждем 2 секунды

servo1.write(90); // ставим угол поворота под 90
delay(2000); // ждем 2 секунды

servo1.write(180); // ставим угол поворота под 180
delay(2000); // ждем 2 секунды
}
```

Джойстик

Джойстик (Рис. 3) содержит два потенциометра и кнопку.

Рис. 3 Джойстик (внешний вид)

Потенциометры представляют собой переменные резисторы — своего рода датчики, выдающие напряжение в зависимости от поворота устройства вокруг своей оси. По мере перемещения джойстика вокруг своего центра его сопротивление — и, следовательно, вывод — изменяется. Выходы потенциометров являются аналоговыми, поэтому могут передавать значение только в диапазоне от 0 до 1023 при считывании аналоговым контактом платы Arduino. Обычно джойстик имеет пять штырьковых контактов:

- VRx (сигнал по оси x);
- VRy (сигнал по оси y);
- SW (кнопка);
- GND (заземление);
- 5V (питание);

Ниже приведен пример кода с комментариями, иллюстрирующий подключение джойстика.

```
#define pinX    A2  // ось X джойстика
#define pinY    A1  // ось Y джойстика
#define swPin    2  // кнопка джойстика
#define ledPin  13  // светодиод на Pin 13

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

  pinMode(ledPin, OUTPUT);
  pinMode(pinX, INPUT);
  pinMode(pinY, INPUT);

  pinMode(swPin, INPUT);
  digitalWrite(swPin, HIGH);
}

void loop() {
  boolean ledState = digitalRead(swPin); // считываем состояние кнопки
  digitalWrite(ledPin, ledState);        // вкл./выкл. светодиод

  int X = analogRead(pinX);              // считываем значение оси X
  int Y = analogRead(pinY);              // считываем значение оси Y

  Serial.print(X);                       // выводим в Serial Monitor
  Serial.print("\t");                    // табуляция
  Serial.println(Y);
}
```

1.2. Порядок выполнения работы

– В соответствии с данными Табл. 1 и Табл. 2 Собрать схему установки.

Табл. 1 Данные для подключения сервоприводов к плате ARDUINO

СЕРВОПРИВОДЫ	ARDUINO
Красные провода	Контакт 5V
Коричневые провода	Контакт GND
Желтый провод первого сервопривода	Контакт 9
Желтый провод второго сервопривода	Контакт 10

Табл. 2 Данные для подключения джойстика к плате ARDUINO

ДЖОЙСТИК	ARDUINO
Вывод 5V	Контакт 5V (через макетную плату)
Вывод GND	Контакт GND (через макетную плату)
Вывод VRx	Контакт A0
Вывод VRy	Контакт A1
Вывод SW	Не используется

– Разработать программу управления сервоприводами с учетом ограничений на углы поворота.

– Запустить установку и провести отладку разработанной программы.
– Оформить отчет по работе.

2. Импульсное регулирование скорости перемещения манипулятора

Цель работы: изучить методику импульсного регулирования скорости вращения сервопривода

2.1. Краткие теоретические сведения

Аппаратное прерывание от таймера.

Аппаратное прерывание это сигнал, сообщающий о каком-то событии. По его приходу выполнение программы приостанавливается, и управление переходит на обработчика прерываний. После обработки управление возвращается в прерванный код программы.

С точки зрения программы прерывание это вызов функции по внешнему, не связанному напрямую с программным кодом, событию.

Сигнал прерывания от таймера вырабатывается циклически, с заданным периодом. Формирует его аппаратный таймер – счетчик с логикой, сбрасывающий его код при достижении определенного значения, что позволяет программно задать время периода прерывания от таймера, установив код для логики сброса.

В микроконтроллере ATmega328 4 таймера. Timer0, Timer1, Timer2 и watchdog.

Функция ISR в программе для Aruino определяет/переопределяет стандартную функцию (или вектор) обработчик прерываний. Имя функции (или вектор) обработчика прерываний зарезервировано и изменить нельзя. Каждый таймер может генерировать несколько типов прерываний. Тип прерывания и его временные параметры зависят от значений служебных битов в специальных 8-ми битных регистрах управления (портах ввода/вывода) микроконтроллера (Табл. 1).

Табл. 3.
Прерывания по таймерам микроконтроллера ATmega328.

№	Vector	Описание
6	WDT_vect	Таймаут сторожевого таймера
7	TIMER2_COMPA_vect	Совпадение A таймера/счетчика T2
8	TIMER2_COMPB_vect	Совпадение B таймера/счетчика T2
9	TIMER2_OVF_vect	Переполнение таймера/счетчика T2
10	TIMER1_CAPT_vect	Захват таймера/счетчика T1
11	TIMER1_COMPA_vect	Совпадение A таймера/счетчика T1
12	TIMER1_COMPB_vect	Совпадение B таймера/счетчика T1
13	TIMER1_OVF_vect	Переполнение таймера/счетчика T1
14	TIMER0_COMPA_vect	Совпадение A таймера/счетчика T0
15	TIMER0_COMPB_vect	Совпадение B таймера/счетчика T0
16	TIMER0_OVF_vect	Переполнение таймера/счетчика T0

В нижеследующих примерах приведен код с комментариями, реализующий мигание светодиода на частоте 1 Гц. В первом примере используется 8-ми битный Timer2 (Обычно он используется в функции tone()), а во втором 16-ти битный Timer1 (обычно он используется библиотекой Servo)

Пример 1.

```
volatile int interruptCount = 0;


ISR (TIMER2_COMPA_vect) {                              // или так ISR(_VECTOR(7))
```

```
  interruptCount++;
  if (interruptCount == 50) {
    digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
    interruptCount = 0;
  }
}

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);

  // инициализация Timer2
  TCCR2A = 0;
  TCCR2B = 0;
  OCR2A = 155;                                  // установка регистра совпадения 500 мс
  TCCR2A = (1 << WGM21);                        // CTC режим
  TIMSK2 |= (1 << OCIE2A);                      // включение прерываний по совпадению
  TCCR2B = (1 << CS22) | (1 << CS21) | (1 << CS20); // запуск с таймера с делителем на
1024
}

void loop() {
}
```

Пример 2.

```
ISR(TIMER1_COMPA_vect) {                          // или так ISR(_VECTOR(11))
  digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
}

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);

  // инициализация Timer1
  TCCR1A = 0;
  TCCR1B = 0;
  OCR1A = 7750;                               // установка регистра совпадения 500 мс
  TCCR1B |= (1 << WGM12);                           // CTC режим
  TIMSK1 |= (1 << OCIE1A);                        // включение прерываний по совпадению
  TCCR1B |= (1 << CS10) | (1 << CS12);       // запуск с таймера с делителем на 1024
}

void loop() {
```

```
}
```

Использование библиотеки MsTimer2, предназначенной для конфигурирования аппаратного прерывания от **Timer2**, позволяет существенно упростить работу с этим таймером (http://mypractic.ru/downloads/arduino/MsTimer2.zip).

Библиотека имеет всего три функции:

MsTimer2::set(unsigned long ms, void (*f)())

Эта функция устанавливает время периода прерывания в мс. С таким периодом будет вызываться обработчик прерывания f. Он должен быть объявлен как void (не возвращает ничего) и не иметь аргументов. * f – это указатель на функцию. Вместо него надо написать имя функции.

MsTimer2::start()

Функция разрешает прерывания от таймера.

MsTimer2::stop()

Функция запрещает прерывания от таймера.

Перед именем функций надо писать MsTimer2::, т.к. библиотека написана с использованием директивы пространства имен namespace.

Ниже приведен пример использования библиотеки **MsTimer2**

Пример 3.

```
#include <MsTimer2.h>
#define LED_1_PIN 13 // светодиод подключен к выводу 13
volatile int count=0; // квалификатор volatile указывает, что переменная будет изменяться
в подпрограмме обработки прерывания
void setup() {
pinMode(LED_1_PIN, OUTPUT);      // определяем вывод светодиода как выход
MsTimer2::set(500, timerInterupt); // задаем период прерывания по таймеру 500 мс
MsTimer2::start();              // разрешаем прерывание по таймеру
}
void loop() {
  while (true) {
    if ( count != 0 ) break;
  }
  count= 0;
  digitalWrite(LED_1_PIN, ! digitalRead(LED_1_PIN));  // инверсия состояния светодиода
}
// обработчик прерывания
void  timerInterupt() {
  count++;
}
```

2.2. Порядок выполнения работы

– В соответствии с данными Табл. 1 и Табл. 2 Собрать схему установки.

Табл. 4
Данные для подключения сервоприводов к плате ARDUINO

СЕРВОПРИВОДЫ	ARDUINO
Красные провода	Контакт 5V
Коричневые провода	Контакт GND
Желтый провод первого сервопривода	Контакт 9
Желтый провод второго сервопривода	Контакт 10

Табл. 5
Данные для подключения джойстика к плате ARDUINO

ДЖОЙСТИК	ARDUINO
Вывод 5V	Контакт 5V (через макетную плату)
Вывод GND	Контакт GND (через макетную плату)
Вывод VRx	Контакт A0
Вывод VRy	Контакт A1
Вывод SW	Не используется

– Разработать программу, обеспечивающую импульсное регулирование скорости вращения сервоприводов с учетом ограничений на углы поворота. **Указание:** для реализации импульсного регулирования использовать прерывания по таймеру.
– Запустить установку и провести отладку разработанной программы.
– Оформить отчет по работе.

3. Система беспроводного управления манипулятором

Цель работы: изучить методику построения беспроводных систем управления сервоприводами манипулятора

3.1. Краткие теоретические сведения

ИК-приемник

ИК-приемник воспринимает инфракрасный сигнал только на частоте 38 кГц (иногда 40кГц). Именно такое свойство позволяет датчику игнорировать много посторонних световых шумов от ламп освещения и солнца. В работе используется ИК-датчик VS1838B со следующими характеристиками:

– несущая частота: 38 кГц;
– напряжение питания: 2,7 — 5,5 В;
– потребляемый ток: 50 мкА.

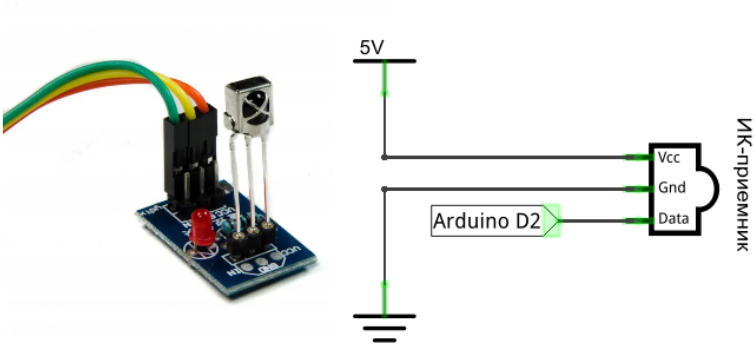

Рис. 4 Инфракрасный приемник

ИК-пульт

Внешний вид используемого в работе инфракрасного пульта управления приведен на Рис. 1. При нажатии на любую кнопку пульта, он передает соответствующее этой кнопке числовое значение, которое принимает ИК-датчик.

Рис. 5 Инфракрасный пульт управления

Настройка

1. Загрузить библиотеку IRremote (архив, содержащий библиотеку, доступен по адресу https://github.com/Arduino-IRremote/Arduino-IRremote) и установить ее в среде разработки Arduino.
2. Установить ИК-приемник на макетную плату. Подключить ИК- приемник к плате Arduino согласно данным Табл. 2 (у некоторых моделей ИК-приемников расположение выводов может отличаться).
3. Загрузить и выполнить нижеприведенный код

```
#include IRremote.h>
int RECV_PIN = 11;
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{
  Serial.begin(9600);
  // Подсказка пользователю в случае, если драйвер прерывания аварийно завершает
  //работу
  Serial.println("Enabling IRin");
  irrecv.enableIRIn(); // Запуск приемника
  Serial.println("Enabled IRin");
}
void loop() {
  if (irrecv.decode(&results)) {
```

```
    Serial.println(results.value, HEX);
    irrecv.resume(); // Получить следующее значение
  }
  delay(100);
      }
```

Вначале код обращается к библиотеке IRremote, с помощью которой перехватывается сигнал с ИК-датчика и полученные данные передаются на плату Arduino. ИК-приемнику назначается контакт 11 платы Arduino, и код формирует канал связи со средой разработки Arduino. Теперь при нажатии той или иной кнопки на пульте ее значение сразу отображается в окне монитора порта. Код продолжает работать в зацикленном режиме, ожидая нажатия кнопок и передавая соответствующие значения в среду разработки Arduino.

4. Откройте окно монитора порта в среде разработки Arduino.
5. Направьте пульт дистанционного управления на ИК-приемник и нажмите несколько кнопок. Их значения отобразятся в окне монитора порта в шестнадцатеричном виде, как показано на Рис. 3. Для достижения наилучшего результата, нажимайте кнопки быстро, сразу отпуская их. Если удерживать кнопку нажатой, монитор порта выведет значение Fs и будет отображать его до тех пор, пока вы не отпустите кнопку.

Рис. 6 Монитор порта

6. Запишите появляющиеся цифры и названия кнопок, которым они соответствуют.

3.2. Порядок выполнения работы

– В соответствии с данными Табл. 1 и собрать схему установки.

Табл. 6

Данные для подключения сервоприводов к плате ARDUINO

СЕРВОПРИВОДЫ	ARDUINO
Красные провода	Контакт 5V
Коричневые провода	Контакт GND
Желтый провод первого сервопривода	Контакт 9
Желтый провод второго сервопривода	Контакт 10

Табл. 7

Данные для подключения ИК-приемника к плате ARDUINO

ИК-приемник	ARDUINO
Вывод OUT	Контакт 11

Вывод GND	Контакт GND
Вывод VCC	Контакт 5V

– Разработать программу, обеспечивающую беспроводное управление сервоприводами манипулятора.
– Запустить установку и провести отладку разработанной программы.
– Оформить отчет по работе.

4. Вывод сообщений на ЖК-дисплей

Цель работы: изучить методику вывода сообщений о положении сервоприводов манипулятора на ЖК-дисплей.

4.1. Краткие теоретические сведения

В настоящей работе предполагается вывод информации о пложении сервоприводов манипулятора на жидкокристаллический дисплей (Liquid Crystal Display) LCD 1602.

Рис. 7 Двухстрочный ЖК-дисплей

Назначение выводов ЖК-дисплея приведено в Табл. 1.

Табл. 8
Назначение выводов ЖК-дисплея

Номер вывода	Назначение	Номер вывода	Назначение
1	Земля GND	6	Enable
2	Питание 5 В	7 – 14	Линии данных
3	Контрастность монитора	15	Плюс подсветки
4	Команда, данные	16	Минус подсветки
5	Запись и чтение данных		

Стандартная схема подключения ЖК-дисплея к плате Arduino приведена на Рис. 2. Из-за большого количества подключаемых контактов может не хватить места для присоединения нужных элементов. Использование интерфейса I2C уменьшает количество проводов до 4, а занятых выводов платы Arduino - до 2.

Протокол I2C

I2C / IIC (Inter-Integrated Circuit) – это протокол, изначально создававшийся для связи интегральных микросхем внутри электронного устройства. Разработка принадлежит фирме

Philips. В основе I2C протокола лежит использование 8-битной шины, которая нужна для связи блоков в управляющей электронике, и системе адресации, благодаря которой можно общаться по одним и тем же проводам с несколькими устройствами.

Самая простая схема I2C может содержать одно ведущее устройство (чаще всего это микроконтроллер Ардуино) и несколько ведомых (например, дисплей LCD). Каждое устройство имеет адрес в диапазоне от 7 до 127. Двух устройств с одинаковым адресом в одной схеме быть не должно.

Плата Arduino поддерживает I2C на аппаратном уровне, для чего используются выводы A4 и A5.

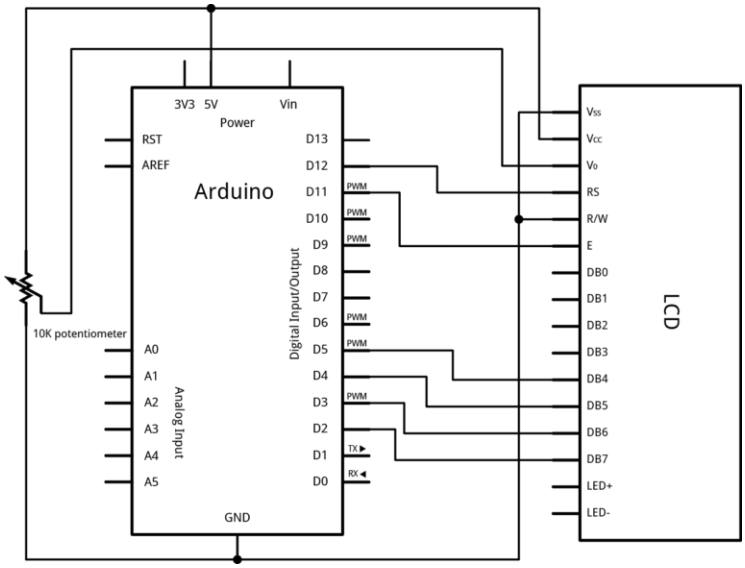

Рис. 8 Стандартная схема подключения ЖК-дисплея к плате Arduino

Модуль i2c для ЖК-дисплея 1602 Arduino

Внешний вид модуля I2C для ЖК-дисплея 1602 приведен на Рис. 3.

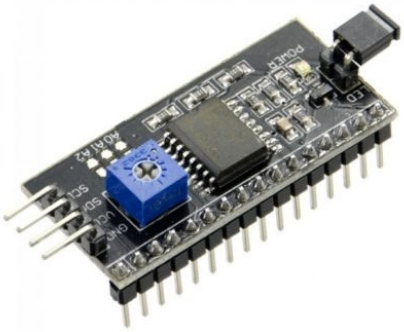

Рис. 9 Модуль i2c для ЖК-дисплея 1602

С одной стороны модуля расположены выводы I2C (см. Табл. 2), а с другой стороны расположен разъем внешнего питания. Кроме того, на плате модуля расположены выводы для подключения к ЖК-дисплею.

Табл. 9

Подключение модуля I2C к плате Arduino

Модуль I2C	ARDUINO
GND	GND
VCC	+5V
SCL	A5
SDA	A4

Библиотеки для работы с I2C LCD дисплеем

Для взаимодействия Arduino с LCD 1602 по шине I2C необходимы две библиотеки:

– Wire.h для работы с I2C уже имеется в стандартной программе Arduino IDE.
– LiquidCrystal_I2C.h, которая включает в себя большое разнообразие команд для управления монитором по шине I2C и позволяет сделать код проще и короче. Данную библиотеку нужно установить дополнительно (https://arduinomaster.ru/wp-content/uploads/2018/12/LiquidCrystal_I2Cv1-1.zip).

Пример использования библиотеки LiquidCrystal_I2C.h

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h> // Подключение библиотеки

LiquidCrystal_I2C lcd(0x27,16,2); // Указываем I2C адрес (наиболее
// распространенное значение), а также параметры экрана (в случае LCD 1602 - 2
// строки по 16 символов в каждой

void setup()
{
  lcd.init();                    // Инициализация дисплея
  lcd.backlight();               // Подключение подсветки
  lcd.setCursor(0,0);            // Установка курсора в начало первой строки
  lcd.print("Hello");       // Набор текста на первой строке
  lcd.setCursor(0,1);            // Установка курсора в начало второй строки
  lcd.print("ArduinoMaster");       // Набор текста на второй строке
}
void loop()
{
}
```

Также библиотека LiquidCrystal_I2C включает следующие функции и методы:

– **home()** и **clear()** – первая функция позволяет вернуть курсор в начало экрана, вторая тоже, но при этом удаляет все, что было на мониторе до этого.
– **write(ch)** – позволяет вывести одиночный символ ch на экран.
– **cursor()** и **noCursor()** – показывает/скрывает курсор на экране.
– **blink()** и **noBlink()** – курсор мигает/не мигает (если до этого было включено его отображение).
– **display()** и **noDisplay()** – позволяет подключить/отключить дисплей.
– **scrollDisplayLeft()** и **scrollDisplayRight()** – прокручивает экран на один знак влево/вправо.
– **autoscroll()** и **noAutoscroll()** – позволяет включить/выключить режим автопрокручивания. В этом режиме каждый новый символ записывается в одном и том же месте, вытесняя ранее написанное на экране.
– **leftToRight()** и **rightToLeft()** – Установка направление выводимого текста – слева направо или справа налево.

– **createChar(ch, bitmap)** – создает символ с кодом ch (0 – 7), используя массив битовых масок bitmap для создания черных и белых точек.

4.2. Порядок выполнения работы

– В соответствии с данными Табл. 2, Табл. 3 и Табл. 4 собрать схему установки.

Табл. 10

Данные для подключения сервоприводов к плате ARDUINO

СЕРВОПРИВОДЫ	ARDUINO
Красные провода	Контакт 5V
Коричневые провода	Контакт GND
Желтый провод первого сервопривода	Контакт 9
Желтый провод второго сервопривода	Контакт 10

Табл. 11

Данные для подключения ИК-приемника к плате ARDUINO

ИК-приемник	ARDUINO
Вывод OUT	Контакт 11
Вывод GND	Контакт GND
Вывод VCC	Контакт 5V

– Разработать программу, обеспечивающую беспроводное управление сервоприводами манипулятора и вывод информации о положении сервоприводов на ЖК-дисплей.

– Запустить установку и провести отладку разработанной программы.

– Оформить отчет по работе.