МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра «Вычислительные системы и технологии»

ОТЧЁТ

По лабораторной работе №1

по дисциплине «Аппаратное и программное обеспечение   
роботизированных систем»

ПРОВЕРИЛ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гай В.Е.

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Львутина Т.В.

**17-В-1**

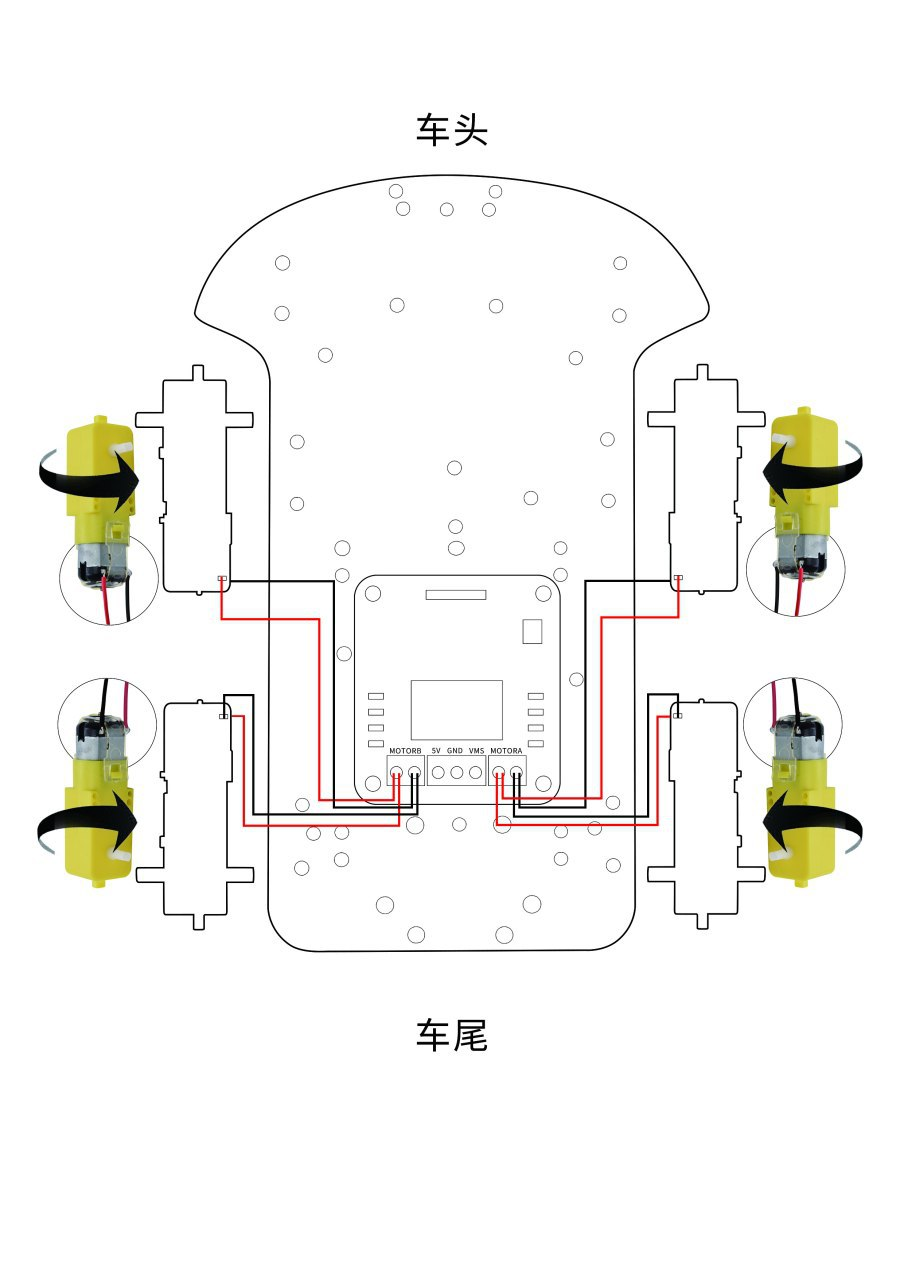
Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород

2020

**Часть 1. Построение робота.**



Описание распиновки:

GND: Земля 5v: выход 5V, если джампер 12V стоит на месте, идеально для питания Arduino (и т.п.)

IN1: включает мотор A1

IN2: включает мотор A0

IN3: включает мотор А3

IN4: включает моторА4

EnA: D6

EnB: D5

**Часть 2. Теоретическая справка.**

Структура программы Ардуино.

Структура программы Ардуино  достаточно проста и в минимальном варианте состоит из двух частей setup() и loop().

void setup()  {

//  код выполняется один раз при запуске программы

}

void loop()  {

// основной код, выполняется в цикле

}

Функция setup() выполняется один раз, при включении питания или сбросе контроллера. Обычно в ней происходят начальные установки переменных, регистров. Функция должна присутствовать в программе, даже если в ней ничего нет.

После завершения setup() управление переходит к функции loop().  Она в бесконечном цикле выполняет команды, записанные в ее теле (между фигурными скобками). Собственно эти команды и совершают все алгоритмические действия контроллера.

Для чего нужна функция void setup()

Загрузив программу, Arduino дает нашему коду возможность поучаствовать в инициализации системы. Для этого мы должны указать микроконтроллеру команды, которые он выполнит в момент загрузки и потом забудет про них (т.е. эти команды выполнятся только один раз при старте системы). И именно с этой целью в нашей с вами программе  мы должны выделить блок, в котором будут храниться эти команды. void setup(), а верней  пространство внутри фигурных скобок этой функции, и является таким местом внутри Arduino скетча.

Синтаксис void setup()

Синтаксис функции setup очень прост:

void setup(){

// тут располагаются команды подпрограммы

}

В этом примере слово setup – это название функции. Нужно писать именно так, именно в таком регистре. Слово перед названием описывает  тип возвращаемых данных. В данном случае никаких данных подпрограмма не возвращает, поэтому мы должны указать слово void. Все команды должны размещаться внутри фигурных скобок {}.

## Примеры void setup()

Чаще всего внутри void  setup указываются следующие инструкции:

* pinMode с указанием номера и типа пина. Это инструкция определяет  режим работы пинов ардуино.
* Serial.begin с указанием скорости (чаще всего, 9600). Эта инструкция инициализирует работы с последовательным портом на указанной скорости.
* Инструкции подключения и инициализации различных объектов библиотек ардуино. Например, servo.atach(6) укажет библиотеке, что сервопривод мы подключили к 9 пину, и все последующие действия код библиотеки будет совершать именно с этим портом.
* Инициализацию глобальных переменных, если мы по каким-то причинам не можем сделать это при определении переменных в глобальной области видимости.
* Выполнение других настроек и начальных значений для переменных и объектов.

## Функция void loop()

Функция loop это то место, куда мы должны поместить команды, которые  будут выполняться все время, пока включена плата Arduino. Начав выполнение с первой команды, микроконтроллер дойдет до конца и сразу же перепрыгнет в начало, чтобы повторить ту же последовательность. И так бесконечное число раз (до тех пор, пока на плату будет подан электричество). Наиболее уместный перевод английского слова loop в даном случае – это цикл (петля). По своей сути, void loop – это главная функция, точка входа в нашу программу.

С помощью функций arduino void loop и void setup мы передаем микроконтроллеру  инструкции нашего скетча. Все то, что содержится внутри блока setup выполнится один раз. Содержимое блока loop будет выполняться в цикле до тех пор, пока останется включенным Arduino-контроллер.

**Часть 3. Программа для робота.**

#include <IRremote.h> // подключаем библиотеку для работы с ИК приемником

int receiverpin = 11;//Define the infrared receiving signal pin

int pinI1=A1;//Define I1 interface

int pinI2=A0;//Define I2 interface

int speedpin1=6;//The definition of EA (PWM speed control) interface

int pinI3=A3;//Define the I3 interface

int pinI4=A2;//Define the I4 interface

int speedpin2=5;//The definition of EB (PWM speed control) interface

#define ADVAN 0xFF629D //The remote control >> key 7 forward

#define BAC 0xFFA857 //The remote control << key 8 back

#define STO 0xFF02FD //The remote control > | key 6 stop

#define left 0xFF22DD //The remote control - key 5

#define right 0xFFC23D //The remote control + key 4

IRrecv irrecv(receiverpin);//инициализируем порт 11 как порт ИК приемника

// Код в блоке setup выполняется один раз при каждом запуске микроконтроллера

void setup()

{

Serial.begin(9600);// Задаем скорость передачи данных в 9600 бод

irrecv.enableIRIn();// запускаем ИК приемник

pinMode(pinI1,OUTPUT);//определяем пин А1 как интерфейс вывода

pinMode(pinI2,OUTPUT); //определяем пин А0 как интерфейс вывода

pinMode(speedpin1,OUTPUT); //определяем пин 6 как интерфейс вывода

pinMode(pinI3,OUTPUT);// //определяем пин А2 как интерфейс вывода

pinMode(pinI4,OUTPUT); //определяем пин А3 как интерфейс вывода

pinMode(speedpin2,OUTPUT); //определяем пин 5 как интерфейс вывода

pinMode(receiverpin,INPUT); //определяем пин 11 как интерфейс ввода

irrecv.enableIRIn();// запускаем ИК приемник

}

// Код в блоке loop выполняется “по кругу” на всём протяжении работы микроконтроллера, начиная с момента завершения выполнения setup.

void loop()

{

if (irrecv.decode(&results)) // Если кнопка была нажата

{

Serial.println(results.value, HEX); // выводим сигнал полученный с порта при нажатии кнопки в консоль в шестнадцатеричном значении

switch(results.value) // применяем оператор switch к значению считанному с ИК приемника

{

case ADVAN://если нажата кнопка «вперед»

goForward(1000); //едем вперед со скорость 1000 единиц

break; //завершаем выполнение кода и передаем управление следующему оператору после switch

case BAC:// если нажата кнопка «назад»

goBackward(1000); // едем назад со скорость 1000 единиц

break; //завершаем выполнение кода и передаем управление следующему оператору после switch

case left:// если нажата кнопка «влево»

goLeft(1000); // крутимся влево со скорость 1000 единиц

break; //завершаем выполнение кода и передаем управление следующему оператору после switch

case right:// если нажата кнопка «вправо»

goRight(1000); //// крутимся вправо со скорость 1000 единиц

break; //завершаем выполнение кода и передаем управление следующему оператору после switch

case STO:// если нажата кнопка «stop»

stopa(); //останавливаемся

break; //завершаем выполнение кода и передаем управление следующему оператору после switch

default: // если значение не совпало с кейсами

delay(100); // делаем задержку в 100 мс

Serial.println(results.value); //выводим значение с ИК приемника в консоль

}

irrecv.resume(); // Считываем следующее значение с ИК приемника

}

delay(100); //задержка в 100 мс (после оператора switch )

}

// Далее идут пользовательские функции, выполняются один раз при каждом вызове из блоков setup или loop

void goForward(int a)// функция езды вперед

{

analogWrite(speedpin1,a);//подаем на аналоговый порт «6» значение «скорости»

analogWrite(speedpin2,a);//подаем на аналоговый порт «5» значение «скорости»

digitalWrite(pinI1,LOW);// 0 В на выходе

digitalWrite(pinI2,HIGH);// 5 В на выходе

digitalWrite(pinI3,LOW);// 0 В на выходе

digitalWrite(pinI4,HIGH);// 5 В на выходе

delay(100);//задержка в 100мс

}

void goBackward(int b)// функция езды назад

{

analogWrite(speedpin1,b); //подаем на аналоговый порт «6» значение «скорости»

analogWrite(speedpin2,b); //подаем на аналоговый порт «5» значение «скорости»

digitalWrite(pinI1,HIGH); // 0 В на выходе

digitalWrite(pinI2,LOW); // 0 В на выходе

digitalWrite(pinI3,HIGH); // 0 В на выходе

digitalWrite(pinI4,LOW); // 0 В на выходе

delay(100); //задержка в 100мс

}

void goLeft(int c)//функция кручения влево

{

analogWrite(speedpin1,c); //подаем на аналоговый порт «6» значение «скорости»

analogWrite(speedpin2,c); //подаем на аналоговый порт «5» значение «скорости»

digitalWrite(pinI1,LOW); // 0 В на выходе

digitalWrite(pinI2,HIGH); // 0 В на выходе

digitalWrite(pinI3,HIGH); // 0 В на выходе

digitalWrite(pinI4,LOW); // 0 В на выходе

delay(100); //задержка в 100мс

}

void goRight(int d)// фцнкция кручения вправо

{

analogWrite(speedpin1,d); //подаем на аналоговый порт «6» значение «скорости»

analogWrite(speedpin2,d); //подаем на аналоговый порт «5» значение «скорости»

digitalWrite(pinI1,HIGH); // 0 В на выходе

digitalWrite(pinI2,LOW); // 0 В на выходе

digitalWrite(pinI3,LOW); // 0 В на выходе

digitalWrite(pinI4,HIGH); // 0 В на выходе

delay(100); //задержка в 100мс

}

void stopa()// функция остановки

{

digitalWrite(pinI1,LOW); // 0 В на выходе

digitalWrite(pinI2,LOW); // 0 В на выходе

digitalWrite(pinI3,LOW); // 0 В на выходе

digitalWrite(pinI4,LOW); // 0 В на выходе

delay(100); //задержка в 100мс

}