Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра: электронных вычислительных машин

Факультет: компьютерных систем и сетей

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине
«Схемотехника»
на тему:
«Модуль автономного управления моделью автомобиля»

Выполнил:
студент гр. 350502
Зайцев Ю. В.

Руководитель проекта:
Марченко В.В.

Минск 2015

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире информационные технологии играют неотъемлимую роль в жизни человечества. На сегодняшний день практически не осталось сфер человеческой деятельности, не охваченных оптимизацией с использованием различных вычислительных средств. Это является результатом развития интегральной технологии, внедрение которой позволило наладить массовый выпуск дешёвых, высококачественных, не требующих настройки и наладки микроэлектронных функциональных узлов различного назначения. Если раньше значительная часть времени разработчиков уходила на расчёты режимов отдельных каскадов и определение их параметров, то в настоящее время большая часть усилий тратится на согласование работы микросхем.

Целью данной курсовой работы является разработка модуля управления, позволяющего добавить «интеллектуальные» функции к уже имеющемуся схемотехническому решению – радиоуправляемой модели автомобиля (МА). Следуя современным тенденциям, при разработке используется готовый универсальный компонент – плата Arduino Uno R3 на базе микроконтроллера ATmega328 (далее по тексту Arduino). При проектировании были поставлены следующие задачи:

- изучить на практике возможности программно-аппаратной платформы Arduino;
- разработать устройство управления на базе платы Arduino, разделяя разработку на структурный, функциональный и принципиальный уровни абстракции;
- разработать алгоритм работы модуля управления;
- правильно подключить модуль к модели автомобиля и устранить возможные неполадки;
- в процессе разработки сохранить существующий функционал МА.

Готовый модуль должен реализовывать какое-либо «интеллектуальное» поведение. В качестве такового автором рассматривались возможности реализации алгоритмов:

- передвижение в направлении светового пятна;
- передвижение в направлении цветного маркера;
- передвижение в направлении радио-маячка;
- передвижение по нарисованной линии;
- объезд препятствий, обнаруживаемых на расстоянии;
- объезд препятствий, обнаруживаемых контактно (при столкновении);
- режимы программирования и езды по заданной программе;
- управление положением солнечной батареи, располагаемой на крыше МА, для достижения ею максимального КПД.

В итоге для реализации был выбран алгоритм движения по линии, достоинствами которого являются простота аппаратной и программной реализации, а также наличие у автора необходимых аппаратных компонентов.

Возможными областями применения готового продукта являются:

- игрушка для детей и домашних животных;
- автоматизированная система доставки с легко изменяемым маршрутом.

Так как проектирование ведется не с нуля, а с использованием готовой МА, имеющейся в распоряжении автора, необходимо не только синтезировать устройство управления, но и проанализировать принцип работы печатной платы МА, определить способы внесения в ее логику необходимых изменений.

1 ОБЗОР ЛИТЕРАТУРЫ

Как было сказано выше, целью данной курсовой работы является разработка модуля управления, позволяющего добавить «интеллектуальное» поведение к уже имеющемуся МА. Для успешного проектирования необходимо понимание устройства МА, а также знание основ использования программно-аппаратной платформы Arduino. Ниже приведены основные источники информации, использованные при проектировании.

Для того чтобы выбрать максимально подходящий как по цене, так и по возможностям продукт из линейки Arduino, был использован официальный сайт разработчиков платформы[1]. Здесь можно найти все поддерживаемые на данный момент платы с указанием их характеристик, особенностей и области применения. Для использования в проекте была выбрана плата Arduino Uno R3 (см. рисунок 1.1), как наиболее универсальная и максимально подходящая для начинающих.

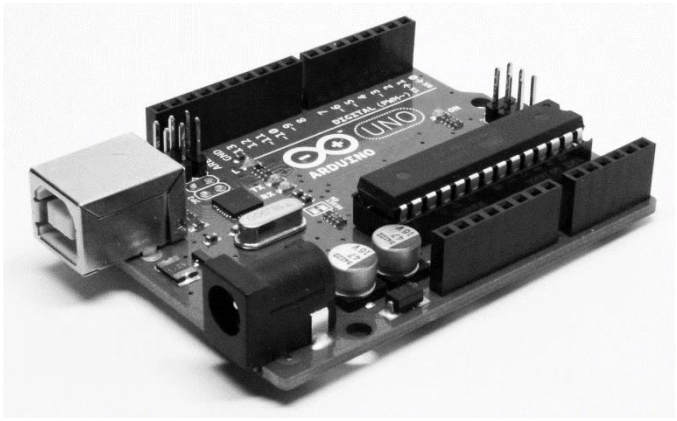


Рисунок 1.1. Arduino Uno R3

Инструкции для первичного подключения микроконтроллерной платы к компьютеру также могут быть найдены на официальном сайте в разделе для начинающих[2]. Этот источник, в частности, помог решить проблему, связанную с невозможностью использования виртуального серийного порта IDE Arduino, связанную с нехваткой прав, при использовании ОС GNU/Linux.

Для изучения основ программирования в среде Arduino IDE (см. рисунок 1.2) и практического ознакомления с аппаратными возможностями платформы была использована книга Тома Иго[3]. Данное издание нацелено, в первую очередь, на построение сетевого взаимодействия между Arduino и другими устройствами, а также на изучении концепции интернета вещей. Однако основы разработки устройств с использованием Arduino Uno описаны здесь достаточно подробно и доходчиво, даже для незнакомого со схемотехникой и программированием читателя.

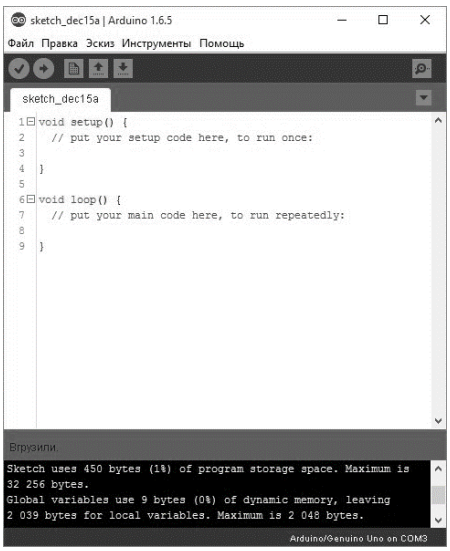
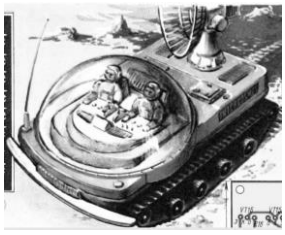
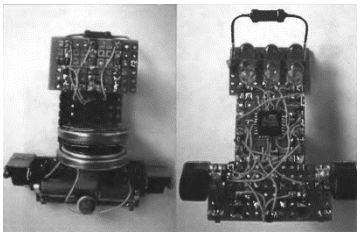
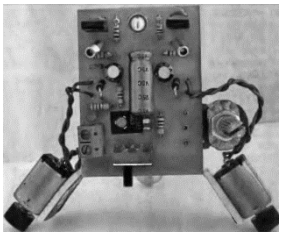


Рисунок 1.2. Внешний вид окна Arduino IDE в ОС Windows

Перед началом проектирования необходимо рассмотреть аналоги. Это позволяет избежать дублирования уже существующих решений и почерпнуть удачные идеи у других авторов. При разработке данного проекта были проанализированы выпуски журнала «Радио» - начинающим» за последние 25 лет на предмет наличия информации о любительском роботостроении. Наиболее любопытными являются статьи о кибернетическом планетоходе[4] (объезд препятствий при столкновении), роботе идущем по линии[5] (передвижение по темной линии на светлом фоне), роботе «Следопыт»[6] (движение следом за световым пятном на полу). Устройства, описанные в них, представлены на рисунке 1.3. Данные статьи повлияли на выбор алгоритма, реализованного в блоке управления МА.

а  б  в 

а – планетоход; б – робот идущий по линии; в – «Следопыт»
Рисунок 1.3

При составлении и оформлении данной пояснительной записки были использованы рекомендации учебно-методического пособия по дипломному проектированию кафедры ЭВМ БГУИР[7].

2 РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ

Структурная схема устройства представлена в Приложении А. Она состоит из следующих обособленных блоков: Arduino Uno, блок управления МА, оптический датчик, блок двигателей МА, блок индикации работы двигателей, проблесковый маячок. Ниже присутствует описание для каждого из блоков.

2.1 Arduino Uno

Устройство управления. Принимает данные, приходящие с блока управления МА и оптического сенсора, обрабатывает их согласно заданному алгоритму и, исходя из результатов, отдает команды остальным блокам. Представлен как аппаратно, так и програмно.

2.2 Блок управления МА

Предназначен для приема и расшифровки команд, приходящих с пульта дистанционного управления. Блок уже имеется в печатной плате МА, синтез не требуется. Необходима лишь небольшая доработка.

2.3 Оптический датчик

Контролирует положение МА относительно черной линии. Для уверенной работы требуется, чтобы линия была очень темной и находилась на ярком светлом фоне. Синтезирован самостоятельно.

2.4 Блок двигателей МА

Содержит маршевый и поворотный двигатели, а также схему для их управления. Так же, как и блок [2.2], присутствует в МА. Полностью функционален, доработки не требуются.

2.5 Блок индикации работы двигателей

Состоит из четырех индикаторов. Отображает направление, в котором пытается двигаться МА.

2.6 Проблесковый маячок

Индикатор работы МА в автономном режиме.

Устройство способно работать в двух режимах:

- ручной;
- автономный.

В ручном режиме Arduino беспрепятственно пропускает управляющие сигналы, приходящие с пульта управления через блок управления на блок двигателей. Таким образом сохранен исходный функционал МА.

В автономном режиме команды передвижения, приходящие с пульта игнорируются Arduino. Направление движения выбирается согласно данным, приходящим с оптического сенсора. Если МА находится ровно на линии, она продолжает двигаться вперед. Если же МА начинает сходить с линии, Arduino подает команду на поворот в противоположном направлении.

Режимы переключаются удержанием рычага поворота пульта в положении «влево» в течении двух секунд. При этом рычаг движения должен находиться в нейтральном положении.

3 РАЗРАБОТКА ФУНКЦИОНАЛЬНОЙ СХЕМЫ

Функциональная схема представлена в приложении Б. Она состоит из следующих блоков:

- Arduino Uno;
- блок управления МА (Car Model Controller);
- оптический датчик (OS);
- драйвер двигателя (Driver);
- проблескового маячка (Beacon);
- индикатора работы двигателей (MI).

Рассмотрим эти блоки подробнее.

3.1 Arduino Uno

Arduino Uno – это набор смонтированных печатных плат. Неотъемлемой частью является микроконтроллер. В данной модели это представитель линейки AVR ATmega328 производства компании Atmel. Микроконтроллеры для Arduino отличаются наличием предварительно прошитого в них загрузчика (bootloader). С помощью этого загрузчика пользователь загружает свою программу в микроконтроллер без использования традиционных отдельных аппаратных программаторов. Загрузчик соединяется с компьютером через интерфейс USB. Поддержка загрузчика встроена в Arduino IDE и выполняется в один щелчок мыши.

На случай затирания загрузчика или покупки микроконтроллера без загрузчика разработчики предоставляют возможность прошить загрузчик в микроконтроллер самостоятельно. Для этого в Arduino IDE встроена поддержка нескольких популярных дешевых программаторов, а большинство плат Arduino имеет штыревой разъем для внутрисхемного программирования (ICSP для AVR, JTAG для ARM).

Порты ввода-вывода микроконтроллеров оформлены в виде штыревых линеек. Никакого буферизирования, защиты, конвертации уровней или подтяжек, как правило, нет. Микроконтроллеры питаются от 5В или 3,3В, в зависимости от модели платы. Соответственно порты имеют такой же размах допустимых входных и выходных напряжений. Программисту доступны некоторые специальные возможности портов ввода-вывода микроконтроллеров, например широтно-импульсная модуляция (ШИМ), аналогово-цифровой преобразователь (АЦП), интерфейсы UART, SPI, I2C.

Программирование ведется целиком через собственную программную оболочку (IDE), бесплатно доступную на сайте Arduino[2]. Язык программирования Arduino является стандартным C++ (используется компилятор AVR-GCC) с некоторыми особенностями, облегчающими новичкам написание первой работающей программы.

Arduino является open-source продуктом. Принципиальная схема устройства представлена в Приложении Е.

Данный проект задействует пины 2, 4-11, A0, A1, а также пины 5В и GND. Назначение пинов:

- 2,4,7,8 – входы для приема команд с блока управления МА;
- 5,6 – выходы на драйвер маршевого двигателя. Используют ШИМ;
- 9,10 – выходы на драйвер поворотного двигателя;
- 11 – выход на проблесковый маячок;
- A0 – вход для приема сигналов с оптического датчика. Использует АЦП;
- A1 – вход для замера напряжения в системе подсистеме передвижения (двигатели и драйвера);
- 5V – питание. Используется всеми остальными блоками, кроме подсистемы передвижения и индикатора работы двигателей;
- GND – подключается к общему проводу.

Также в данном блоке реализована вся логика работы устройства при помощи программирования.

3.2 Блок управления МА

Принимает и анализирует сигналы с антенны. В оригинальной схеме МА был запитан от $U_{ип}$, однако в процессе работы был замечено следующее некорректное поведение.

1. При буксовании двигателей система перегружается, напряжение падает.
2. Блок управления отключается. Соответственно команды управления драйверами пропадают
3. Двигатели отключаются.
4. Напряжение растет, так как пропала нагрузка.
5. Блок управления включается и подает команды на драйверы.
6. Двигатели включаются и начинают буксовать.

Для исправления данного поведения цепь питания блока управления была разомкнута. Блок управления был заново запитан, но уже от пина 5V Arduino. Arduino имеет независимый стабильный источник питания, а потому проблема устранена.

Также блок имеет входы для общего провода (Gnd) и антенны (A).

Имеются выходы F («Forward» - вперед), B («Backward» - назад), L («Left» - влево), R («Right» - вправо), подключенные к входам драйверов двигателей.

3.3 Оптический датчик

Имеет входы для питания от источника Arduino (Vcc) и общего провода (Gnd). На выход подает данные о положении на линии.

3.4 Драйвер двигателя

Блок позволяет управлять большой мощностью двигателя, используя небольшую мощность Arduino. Имеет входы F и B, задающие полярность разности потенциалов между выходами + и -, что позволяет управлять направлением вращения двигателя. Блок реализован на изначальной схеме МА. Полностью функционален, никаких доработок выполнять не требуется.

3.5 Проблесковый маячок

Элементарный индикатор режима работы МА. Если светится – включен автономный режим. Имеет вход In включения с активным уровнем сигнала логической единицей. Также имеет вход Gnd на общий провод.

3.6 Индикатор работы двигателей

Входы F, B, L и R подключаются к выходам драйверов, параллельно с двигателями.

4 РАЗРАБОТКА ПРИНЦИПИАЛЬНОЙ СХЕМЫ

Разработка велась на основе имеющейся схемы МА (Приложение Д). В связи с этим в перечне отсутствуют компоненты изначальной схемы, так как их параметры не известны. Итоговый вариант схемы представлен в приложениях В и Г.

4.1 Расчет резисторов R1..R4

Для подключения модуля потребовалось изменить схему включения микросхемы D1. Фрагмент исходной схемы представлен на рисунке 4.1. Для простоты, на рисунке показано только подключения вывода 10. Выводы 11, 6 и 7 подключались аналогично.

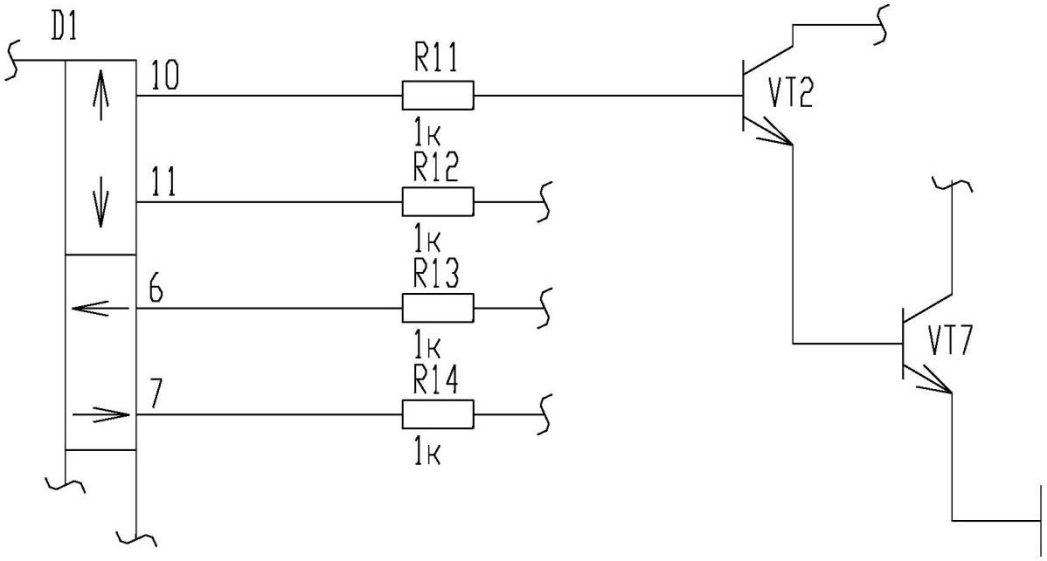


Рисунок 4.1. Исходная схема

Для подключения модуля схема была изменена, как показано на рисунке 4.2.

Требуется рассчитать величины резисторов R1…R4 так, что бы обеспечить работоспособность полученного устройства и не превысить нагрузочную способность выводов микросхемы D1.

Технические характеристики D_1 неизвестны, поэтому рассчитаем максимальный ток вывода 10 в исходной схеме. Ток вывода 10 будет равен току через резистор R_{11} (см. рисунок 4.3), и равен:

$I_{10} = (Uвых_{10} – Uбэ_2 – Uбэ_7)/R_{11}$, где

- $Uвых_{10}$ – напряжение на выводе 10 микросхемы D_1;
- $Uбэ_2$ – напряжение на переходе база – эмиттер транзистора VT_2;
- $Uбэ_7$ – напряжение на переходе база – эмиттер транзистора VT_7;

- R_{11} – сопротивление резистора R_{11}.

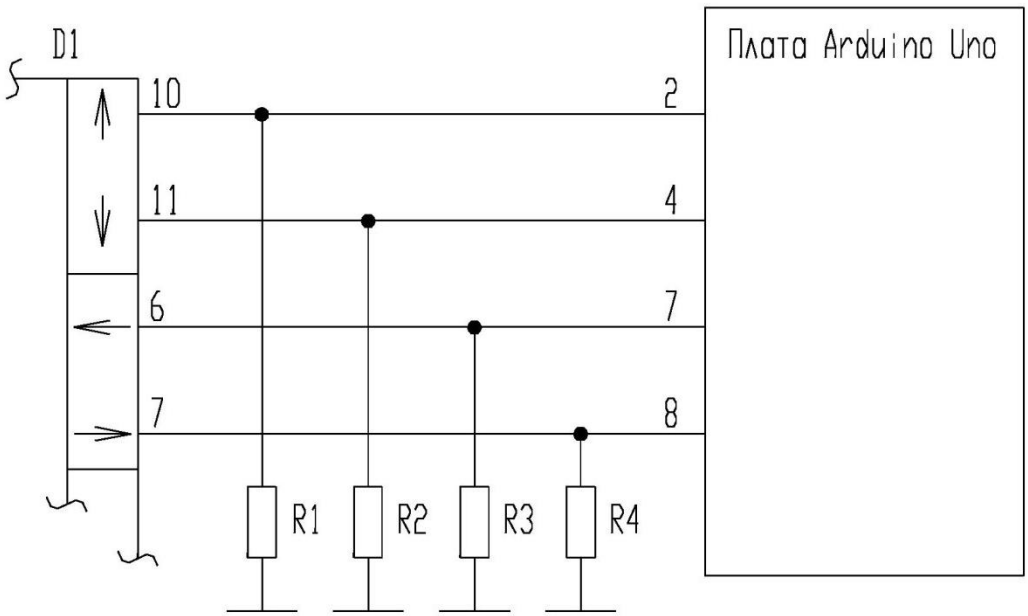


Рисунок 4.2. Подключение модуля управления

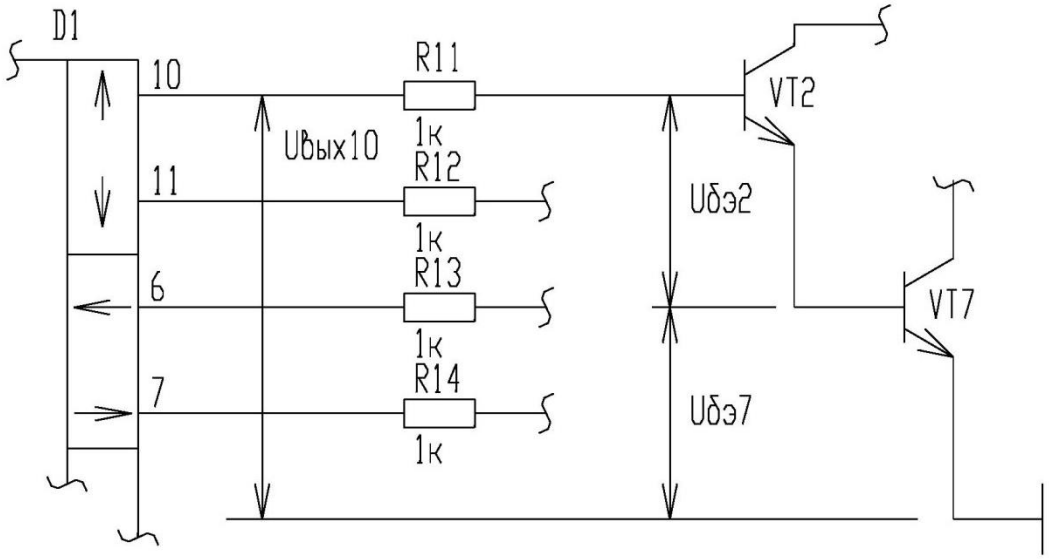


Рисунок 4.3. Пояснения к расчету тока вывода 10 D_1

$Uвых_{10}$ = 4,2 В - напряжение максимально заряженной аккумуляторной батареи (при этом пренебрегаем падением напряжения на выходном каскаде микросхемы D1);

-для кремневых транзисторов в открытом состоянии $Uбэ_2$ = 0,6 В и $Uбэ_7$ = 0,6 В [8];

R_{11} = 1000 Ом.

Для принятых значений

I_{10} = (4,2 В – 0,6 В – 0,6В)/1000 = 0,003 А.

В исходной схеме такие токи требовались для управления ключами на биполярных транзисторах. Для передачи сигнала на входы микроконтроллера не требуются, однако, при применении малых токов сильно возрастает вероятность паразитных наводок от ключевых элементов схемы. С учетом этого принимаем I_{10} = 0,001 А.

Если пренебречь входным сопротивлением микроконтроллера, то $R_1 = Uвых_{10}/ I_{10}$.

Для принятых значений $Uвых_{10}$ = 4,2 В и I_{10} = 0,001 А получаем R_1 = 4,2 В / 0,001А =4200 Ом.

Постоянные резисторы с таким номинальным сопротивлением выпускаются промышленностью, поэтому дополнительно уточнять номинальное сопротивление не требуется.

Для R_1 максимальная мощность, рассеиваемая на резисторе $PR_1 = Uвых_{10} * I_{10}$.

Для значений $Uвых_{10}$ = 4,2 В и I_{10} = 0,001 А получаем PR_1 = 4,2 В * 0,001А = 0,0042 Вт.

Следовательно, можно применить резистор с номинальной рассеиваемой мощностью 0,125 Вт.

В качестве R2 … R4 применим такие резисторы как R1 (их применение аналогично и дополнительные расчеты не требуются).

4.2 Анализ работы драйвера двигателя

С первого взгляда достаточно непросто понять принцип работы драйвера. Для упрощения восприятия была построена эквивалентная схема (см. рисунок 4.4).

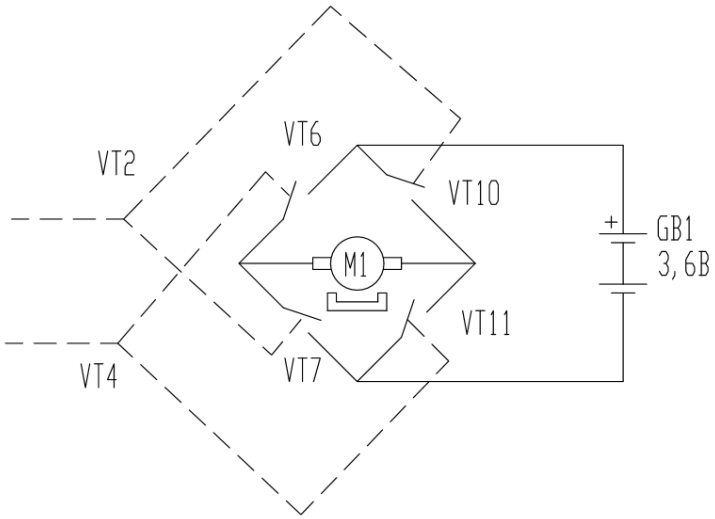


Рисунок 4.4. Эквивалентная схема работы драйвера двигателя

Стоит заметить, что ни в коем случае нельзя подавать логическую единицу одновременно на базы транзисторов VT_2 и VT_4, так как в этом случае клемы аккумулятора GB_1 замыкаются накоротко. В изначальной схемы это невозможно ввиду того, что команды на драйверы приходили с пульта дистанционного управления, конструкция которого исключает возможность выбрать одновременно движение вперед и назад, или поворот одновременно направо и налево. При использовании Arduino в качестве устройства управления это возможно, а потому необходимо быть очень внимательным при составлении алгоритма его работы.

4.3 Алгоритм работы

Алгоритм написан на языке Arduino (C++ подобный). Никаких сторонних библиотек использовано не было. Ниже приведен исходный код алгоритма.

```
#define SENSOR_PIN A0
#define BATTERY_VOLTAGE_PIN A1
#define COMMAND_F_PIN 2
#define COMMAND_B_PIN 4
#define COMMAND_R_PIN 8
#define COMMAND_L_PIN 7
#define MOVE_F_PIN 5
#define MOVE_B_PIN 6
#define MOVE_R_PIN 10
#define MOVE_L_PIN 9
#define BEACON_PIN 11

#define CYCLE_TIMEOUT 10
#define MODE_CHANGE_TIMEOUT 2000

#define L_BIAS 160
#define R_BIAS 230
#define NO_BIAS 195

#define MANUAL_MODE 0
#define AUTO_MODE 0

int ControllerVAxisState = 1;
int ControllerHAxisState = 1;
bool UpdateRequested = true;
int ModeChangeTimer = 0;
byte Mode = MANUAL_MODE;
const String states[3][3] = {{"< /\\  ", "  /\\  ", "  /\\ >"},
                             {"< ||  ",  "  ||  ",  "  || >"},
                             {"< \\/  ", "  \\/  ", "  \\/ >"}
};

int command = 0;
int prevcommand = 0;
```

```
float getPinVoltage(int pin);
void controlModeChange();
void updateRequestedCarState();
void setCarEnginesState();

void setup() {
  pinMode(SENSOR_PIN, INPUT);
  pinMode(BATTERY_VOLTAGE_PIN, INPUT);

  pinMode(COMMAND_F_PIN, INPUT);
  pinMode(COMMAND_B_PIN, INPUT);
  pinMode(COMMAND_R_PIN, INPUT);
  pinMode(COMMAND_L_PIN, INPUT);

  pinMode(MOVE_F_PIN, OUTPUT);
  pinMode(MOVE_B_PIN, OUTPUT);
  pinMode(MOVE_R_PIN, OUTPUT);
  pinMode(MOVE_L_PIN, OUTPUT);

  setCarEnginesState(1, 1);
  Serial.begin(9600);
}

void loop() {
  updateRequestedCarState();

  if (Mode == MANUAL_MODE) {
    if (UpdateRequested) {

Serial.println(states[ControllerVAxisState][ControllerHAxisState]);
      setCarEnginesState(ControllerVAxisState, ControllerHAxisState);
      UpdateRequested = false;
      ModeChangeTimer = 0;
    }
    else {
      if (ModeChangeTimer * CYCLE_TIMEOUT < MODE_CHANGE_TIMEOUT)
        ModeChangeTimer += 1;
      else if (ControllerVAxisState == 1 && ControllerHAxisState == 0 &&
ModeChangeTimer * CYCLE_TIMEOUT == MODE_CHANGE_TIMEOUT) {
        ModeChangeTimer += 1;
        controlModeChange();
      }
    }
  } else {
    if (UpdateRequested) {
      UpdateRequested = false;
      ModeChangeTimer = 0;
    } else {
      if (ModeChangeTimer * CYCLE_TIMEOUT < MODE_CHANGE_TIMEOUT)
        ModeChangeTimer += 1;
      else if (ControllerVAxisState == 1 && ControllerHAxisState == 0 &&
ModeChangeTimer * CYCLE_TIMEOUT == MODE_CHANGE_TIMEOUT) {
        ModeChangeTimer += 1;
        controlModeChange();
      }
    }
    int sensorValue = analogRead(SENSOR_PIN);
    if (!prevcommand) {
      if (sensorValue >= R_BIAS) {
        command = -1;
```

```
      }
      else if (sensorValue <= L_BIAS) {
        command = 1;
      }
    } else {
      if (prevcommand == 1) {
        if (sensorValue >= NO_BIAS) {
          command = 0;
        }
      } else if (prevcommand == -1) {
        if (sensorValue <= NO_BIAS) {
          command = 0;
        }
      }
    }
    if (prevcommand != command)
      switch (command) {
        case - 1:
          Serial.println("left");
          setCarEnginesState(0, 0);
          break;
        case 0:
          Serial.println("neutral");
          setCarEnginesState(0, 1);
          break;
        case 1:
          Serial.println("right");
          setCarEnginesState(0, 2);
          break;
      }
    prevcommand = command;
  }

  delay(CYCLE_TIMEOUT);
}

float getPinVoltage(int pin) {
  return analogRead(pin) * (5.0 / 1023.0);
}

void controlModeChange() {
  Serial.println("Info: Mode changed.");
  Mode = !Mode;
  digitalWrite(BEACON_PIN, Mode);
}

void updateRequestedCarState() {
  if (digitalRead(COMMAND_F_PIN) == HIGH) {
    if (ControllerVAxisState != 0)
      UpdateRequested = true;
    ControllerVAxisState = 0;
  }
  else if (digitalRead(COMMAND_B_PIN) == HIGH) {
    if (ControllerVAxisState != 2)
      UpdateRequested = true;
    ControllerVAxisState = 2;
  }
  else {
    if (ControllerVAxisState != 1)
      UpdateRequested = true;
    ControllerVAxisState = 1;
```

```
  }

  if (digitalRead(COMMAND_L_PIN) == HIGH) {
    if (ControllerHAxisState != 0)
      UpdateRequested = true;
    ControllerHAxisState = 0;
  }
  else if (digitalRead(COMMAND_R_PIN) == HIGH) {
    if (ControllerHAxisState != 2)
      UpdateRequested = true;
    ControllerHAxisState = 2;
  }
  else {
    if (ControllerHAxisState != 1)
      UpdateRequested = true;
    ControllerHAxisState = 1;
  }
}

void setCarEnginesState(int fb, int lr) {
  byte engineFState = fb == 0 ? 1 : 0;
  byte engineBState = fb == 2 ? 1 : 0;
  byte engineLState = lr == 0 ? 1 : 0;
  byte engineRState = lr == 2 ? 1 : 0;

  digitalWrite(MOVE_L_PIN, engineLState);
  digitalWrite(MOVE_R_PIN, engineRState);
  analogWrite(MOVE_F_PIN, engineFState * 230);
  analogWrite(MOVE_B_PIN, engineBState * 230);
}
```

ЗАКЛЮЧЕНИЕ

Во время выполнения курсового проекта все поставленные задачи были достигнуты, цель выполнена. Полученное устройство сохранило возможность дистанционного управления с пульта и приобрело функционал, позволяющий ездить вдоль черной линии.

Программная и аппаратные части были реализованы максимально абстрактно. За счет этого получившееся устройство легко модернизировать, как в аппаратном плане (установленный разъем позволяет легко заменить Arduino Uno на что-либо другое), так и в программном.

Недостатками проекта являются:

- низкая защищенность электронных компонентов, ввиду отсутствия корпуса;
- потери мощности, машина часто застревает на ровном месте

В будущем планируется исправить найденные недостатки. Также возможно добавление к устройству функционала, который рассматривался в качестве задач проекта, но был отвергнут (см. введение).

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Список поддерживаемых продуктов линейки Arduino [Электронный ресурс]. – 2015. – Режим доступа : https://www.arduino.cc/en/Main/Products.

2 Подключение платы Arduino к компьютеру и настройка операционной системы для работы с ней [Электронный ресурс]. – 2015. – Режим доступа : https://www.arduino.cc/en/ en/Guide/HomePage.

3 Иго, Т. Arduino, датчики и сети для связи устройств. Т. Иго / – СПб. : БХВ-Петербург, 2015. – 544 с.

4 Алешин, П. Кибернетический планетоход / П. Алешин // «Радио» - начинающим. – 1987. – №2. – С. 48-50.

5 Свита, С. Робот, идущий по линии / С. Свита // «Радио» - начинающим. – 2008. – №3. – С. 48-49.

6 Лечкин, А. Робот «Следопыт» / А. Лечкин // «Радио» - начинающим. – 2010. – №8. – С. 53-54.

7 Рожнова, Н. Г. Вычислительные машины, системы и сети. Дипломное проектирование : учеб.-метод. пособие / Н. Г. Рожнова, Н. А. Искра, И. И. Глецевич. – Минск : БГУИР, 2014. -96 с.

8 Хоровиц, П. Искусство схемотехники в двух томах, том 1, 3-е издание. / П. Хоровиц, У. Хилл. – М : Мир, 1986.