МИНИСТЕРСТВО ПО РАЗВИТИЮ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

САМАРКАНДСКИЙ ФИЛИАЛ ТАШКЕНТСКОГО УНИВЕРСИТЕТА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
имени МУХАММАДА АЛ-ХОРАЗМИ

Факультет: Компьютерный инжиниринг

Кафедра: Компьютерные системы

Направление: 5330500 – «Компьютерный инжиниринг (ИТ Сервис)»

ВЫПУСКНАЯ
КВАЛИФИКАЦИОННАЯ РАБОТА
для получения академической степени бакалавра

на тему: Формирование модели и разработка программы автоматизации управления роботом

Выпускница: _____ Рашидова Ш.

Научный руководитель: _____ Абдукаримов А.

Рецкнзент: _____Абдурашидов А.

Консультант по БЖД: _____Курбанниязов А.

С А М А Р К А Н Д – 2018

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

За последние годы в республике Узбекистан реализован комплекс мер, направленный на всестороннюю поддержку и развитие научно-инновационной деятельности, выработку действенных механизмов создания благоприятных условий для формирования и дальнейшего развития инновационного потенциала страны.

В целях создания современной инновационной инфраструктуры и благоприятных условий для овладения молодыми учеными и студентами передовыми в мировой практике методами научных исследований и современными технологиями создано Государственное предприятие «Учебно-экспериментальный Центр высоких технологий».

За истекший период принимаются действенные меры по укреплению материально-технического потенциала и повышению эффективности научно-исследовательской деятельности Центра высоких технологий. При нем организованы два производственных участка по выпуску диагностических наборов и производству рассады ценных растений методом микроклонального размножения.

Вместе с тем текущее состояние и основные тенденции развития в стране инновационной деятельности требуют всесторонней поддержки высокотехнологичных предприятий наукоемких отраслей экономики, имеющих разработки, связанные с применением эффективных инновационных технологий и позволяющие выпускать продукцию на уровне мировых стандартов.

Каждый год производители изменяют многие функции автомобилей: улучшают, упрощают, минимизируют. Этот процесс не обходит стороной и мультимедийную часть автомобиля. Обычным цветным дисплеем вместо головного устройства или блока управления климат контроля уже никого не удивишь. Но эти дисплеи встроены в автомобиль стационарно, то есть их нельзя снять в любой момент. Эти факторы и навели меня на мысли о создании продукта под управлением планшета, подключаемого к автомобилю. Данный

планшет можно будет использоваться как стандартный стационарный экран, даже с большим функционалом.

Конечно можно приобрести дорогой автомобиль или дорогой модуль, со встроенными функциями внутри, но не всем это позволяет финансовое положение. Благодаря Arduino и RaspberryPi можно собрать недорогой, но в то же в время функциональный продукт, конкурирующий с дорогими аналогами.

Работа состоит из семи глав, изложенных в 67 страницах. В первой главе дается аналитические обзор

1. Аналитический обзор

1.1 Актуальность

Данная система - это централизованное управление такими механизмами как:

1. Система запуска и остановки двигателя.
2. Система управления стеклоподъемниками.
3. Система закрывания и открывания центрального замка дверей.

Стоимость таких систем не по карману многим потребителям, да и потребность во всех функциях системы возникает не у всех. Одной из наиболее интересных функций данной системы является «Система запуска и остановки двигателя», позволяющий заводить и глушить двигатель автомобиля используя только одну кнопку.

1.2 Обзорно-аналитическая часть

1.2.1 Оценка требований

Для разработчика требуется иметь в наличие компьютер или ноутбук с ОС не ниже Windows7,иметь 4 Гб ОЗУ, двухядерный процессор частотой выше 2 Ггц, видео карта, имеющая 1024 Мб видео памяти, устройство на базе Android с ОС v.4.0 и выше, минимальная разрешающая способность сенсорного экрана 640x480 и наличие Bluetooth версии 2.0

Также необходимо скачать приложение для платформы Android-RemoteXY и программное обеспечение для написания кода под микроконтроллер - Arduino IDE, которые на момент выполнения проекта бесплатны.

Для избегания проблем с пайкой и программированием микроконтроллера проще всего использовать одну из плат Arduino или ее аналогов. А для сборки и тестирования схемы необходимо использовать

breadboard. Он позволяет подключать все элементы продукта, не прибегая к пайке.

Для обмена информацией с устройством Android по Bluetooth требуется приобрести необходимый модуль.

Также для отлаженной работы необходимо использовать реле. Самым оптимальным решением будет использовать готовый блок с восьмью реле.

1.2.2 Общее представление о системе Arduino

Это инструмент для проектирования электронных устройств (электронный конструктор) более плотно взаимодействующих с окружающей физической средой, чем стандартные персональные компьютеры, которые фактически не выходят за рамки виртуальности. Программная часть состоит из бесплатной программной оболочки (IDE) для написания программ, их компиляции и программирования аппаратуры. Аппаратная часть представляет собой набор смонтированных печатных плат, продающихся как официальным производителем, так и сторонними производителями. Полностью открытая архитектура системы позволяет свободно копировать или дополнять линейку продукции Ардуино.

Arduino может использоваться как для создания автономных объектов автоматики, так и подключаться к программному обеспечению на компьютере через стандартные проводные и беспроводные интерфейсы.

В концепцию Ардуино не входит корпусной или монтажный конструктив. Разработчик выбирает метод установки и механической защиты плат самостоятельно. Сторонними производителями выпускаются наборы робототехнической электромеханики, ориентированной на работу совместно с платами Ардуино.

Аппаратная часть.

Под торговой маркой Arduino выпускается несколько плат с микроконтроллером и платы. Большинство плат с микроконтроллером снабжены минимально необходимым набором обвязки для нормальной работы

микроконтроллера (стабилизатор питания, кварцевый резонатор, цепочки сброса ит.п.).

Внешний вид Arduino с платами расширения приведен в соответствии с рисунком 1.1.

Рисунок 1.1 - Стандартный конструктив Ардуино с платами расширения.

Существует несколько версий платформ Arduino. Последняя версия Leonardo базируется на микроконтроллере ATmega32u4.Uno, как и предыдущая версия Duemilanove построены на микроконтроллере Atmel ATmega328 (техническое описание). Старые версии платформы Diecimila и первая рабочая Duemilanoves были разработаны на основе Atmel ATmega168 (техническое описание), более ранние версии использовали ATmega8 (техническое описание). Arduino Mega2560, в свою очередь, построена на микроконтроллере ATmega2560 (техническое описание).

Конструктив.

Ардуино и Ардуино-совместимые платы спроектированы таким образом, чтобы их можно было при необходимости расширять, добавляя в устройство новые компоненты. Эти платы расширений подключаются к Ардуино посредством установленных на них штыревых разъёмов. Существует ряд плат с унифицированным конструктивом, допускающим конструктивно жесткое

соединение процессорной платы и плат расширения в стопку через штыревые линейки. Кроме того, выпускаются платы уменьшенных габаритов (например, Nano, Lilypad) и специальных конструктивов для задач робототехники. Независимыми производителями также выпускается большая гамма всевозможных датчиков и исполнительных устройств, в той или иной степени совместимых с базовым конструктивом Ардуино [1].

Установка Arduino IDE.

Arduino IDE является кроссплатформенным приложением разработанным на Java, поэтому он поддерживает такие операционные системы как Linux, Windows, Mac и другие. В данном программном продукте установка будет производиться на операционную систему Windows.

После установки Arduino IDE необходимо подключить плату и компьютер через usb провод и установить драйвера. После чего настроить подключение с платой. Для этого во вкладке Интсрументы-Плата выберите вашу модель Arduino, а так же во вкладке Инструменты-Порт выберите порт на компьютере, к которому подключена плата.

1.2.3 Среда разработки Arduino

Вид основного интерфейса системы представлен ниже в соответствии с рисунком 1.2.

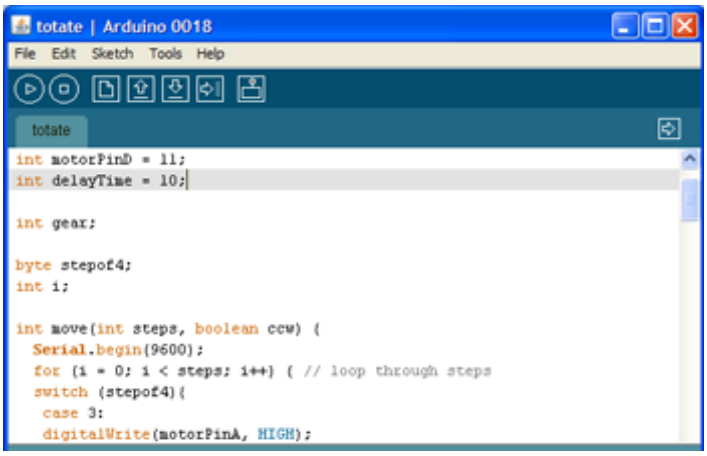

Рисунок 1.2 - Встроенный текстовый редактор программного кода.

Среда разработки Arduino состоит из встроенного текстового редактора программного кода, области сообщений, окна вывода текста(консоли), панели инструментов с кнопками часто используемых команд и нескольких меню. Для загрузки программ и связи среда разработки подключается к аппаратной части Arduino.

Программа, написанная в среде Arduino, называется скетч. Скетч пишется в текстовом редакторе, имеющем инструменты вырезки/вставки, поиска/замены текста. Во время сохранения и экспорта проекта в области сообщений появляются пояснения, также могут отображаться возникшие ошибки. Окно вывода текста(консоль) показывает сообщения Arduino, включающие полные отчеты об ошибках и другую информацию. Кнопки панели инструментов позволяют проверить и записать программу, создать, открыть и сохранить скетч, открыть мониторинг последовательной шины

Блокнот (Sketchbook).

Средой Arduino используется принцип блокнота: стандартное место для хранения программ (скетчей). Скетчи из блокнота открываются через меню File >Sketchbook или кнопкой Open на панели инструментов. При первом запуске программы Arduino автоматически создается директория для блокнота. Расположение блокнота меняется через диалоговое окно Preferences.

Загрузка скетча в Arduino. Перед загрузкой скетча требуется задать необходимые параметры в меню Tools > Board и Tools > Serial Port. Платформы описываются далее по тексту. В ОС Mac последовательный порт может обозначаться как dev/tty.usbserial-1B1 (для платы USB) или /dev/tty.USA19QW1b1P1.1 (для платы последовательной шины, подключенной через адаптер Keyspan USB-to-Serial). В ОС Windows порты могут обозначаться как COM1 или COM2 (для платы последовательной шины) или COM4, COM5, COM7 и выше (для платы USB). Определение порта USB производится в поле Последовательной шины USB Диспетчера устройств Windows. В ОС Linux порты могут обозначаться как /dev/ttyUSB0, /dev/ttyUSB1.

После выбора порта и платформы необходимо нажать кнопку загрузки на

панели инструментов или выбрать пункт меню File > Upload to I/O Board. Современные платформы Arduino перезагружаются автоматически перед загрузкой. На старых платформах необходимо нажать кнопку перезагрузки. На большинстве плат во время процесса будут мигать светодиоды RX и TX. Среда разработки Arduino выведет сообщение об окончании загрузки или об ошибках.

При загрузке скетча используется Загрузчик (Bootloader) Arduino, небольшая программа, загружаемая в микроконтроллер на плате. Она позволяет загружать программный код без использования дополнительных аппаратных средств. Загрузчик (Bootloader) активен в течении нескольких секунд при перезагрузке платформы и при загрузке любого из скетчей в микроконтроллер. Работа Загрузчика (Bootloader) распознается по миганию светодиода (13 пин) (напр.: при перезагрузке платы).

1.2.4 Выбор платформы для разработки приложения

Каждая из платформ для мобильных приложений имеет интегрированную среду разработки, предоставляющую инструменты, позволяющие разработчику программировать, тестировать и внедрять приложения на целевую платформу. Для сравнения мобильных приложений на Java с приложениями, написанными на других языках, сравним платформу Android с iOS и Windows Phone. Статистика продажи смартфонов за 2015 год представлен ниже в соответствии с рисунком 1.3.

Исходя из данной статистики, оптимальной платформой для разработки управляющего приложения является Android, так как она занимает наибольший процент мобильного рынка.

Worldwide Smartphone Sales to End Users by Operating System in 2Q15 (Thousands of Units)

Operating System	2Q15 Units	2Q15 Market Share (%)	2Q14 Units	2Q14 Market Share (%)
Android	271,010	82.2	243,484	83.8
iOS	48,086	14.6	35,345	12.2
Windows	8,198	2.5	8,095	2.8
BlackBerry	1,153	0.3	2,044	0.7
Others	1,229.0	0.4	1,416.8	0.5
Total	**329,676.4**	**100.0**	**290,384.4**	**100.0**

Source: Gartner (August 2015)

Рисунок 1.3 - Статистика продажи смартфонов.

1.2.5 Общее представление о системе Android

Структура платформы Android.

Платформа Android представляет собой программный стек операционной системы на основе Linux, предназначенный для управления устройством (аппаратурой), памятью и процессами. Различные компоненты этого стека представляют собой несколько уровней иерархии и, в целом, обеспечивают функционирование мобильного устройства. Одни из этих компонентов необходимы для работы непосредственно с оборудованием устройства, другие обеспечивают функции связанные с телефонией, позиционированием, мультимедиа и так далее, а третьи предоставляют приложениям фреймворк для реализации многочисленных возможностей мобильного устройства. Другими словами, платформа Android включает в себя непосредственно операционную систему, программное обеспечение промежуточного уровня (middleware) и набор приложений (как встроенных, так и сторонних).

Как мы уже выяснили, Android – это программный стек, то есть целый набор различных программных компонентов. С другой стороны, устройство на платформе Android представляет собой набор аппаратных средств, таких как экран, клавиши, батарея, процессор, память всевозможные датчики и сенсоры и так далее. Все это позволяет нам говорить о платформе Android как о едином аппаратно-программном комплексе, который имеет следующую структуру:

Аппаратура – набор элементов, обеспечивающих функционирование

устройства.

Linux Kernel (+Drivers) – ядро операционной системы Linux и набор драйверов, которые обеспечивают базовые механизмы управления оборудованием и распределения памяти, управления задачами, обеспечения безопасности и так далее.

Библиотеки (C/C++), используемые различными компонентами операционной системы.

Dalvik Virtual Machine (DVM) – виртуальная машина Dalvik, которая обеспечивает среду выполнения Android приложений и компонентов операционной системы (ОС).

Core Library – основные Android библиотеки, содержат большинство функций доступных в ядре библиотеки языка Java (Java API), а так же специальные функции операционной системы Android.

Application Framework (каркас приложений) – набор Java классов (API), предоставляющий приложениям интерфейс к функциям операционной системы и библиотекам.

Разработка приложений для платформы Android ведется преимущественно на языке Java. Для создания программ на языке Java необходимо специальное программное обеспечение. Самые последние версии этого ПО можно загрузить с официального сайта разработчика, Oracle Corporation. К этому программному комплексу относятся такие инструменты как JRE (Java Runtime Environment) и JDK (Java Development Kit). Первый инструмент представляет собой среду выполнения – минимальную реализацию виртуальной машины, в которой запускается и выполняется программный код на Java. Второй инструмент – это в свою очередь целый набор инструментов, комплект разработчика приложений на языке Java. На самом деле, JRE также входит в состав JDK, равно как и различные стандартные библиотеки классов Java, компилятор javac, документация, примеры кода и разнообразные служебные утилиты. Весь этот набор распространяется свободно и имеет версии для различных ОС, поэтому любой может его скачать и использовать. В

JDK не входит интегрированная среда разработки, предполагается, что её разработчик будет устанавливать отдельно.

2. Проектирование программного продукта

2.1 Определение функциональности программного продукта

Определяя функциональность программного продукта, первым делом выбирались функции, наиболее востребованные для управления по Bluetooth. Самая главная среди этих функций - это запуск зажигания и запуск двигателя, используя всего лишь одну кнопку, не прибегая к ключам.

2.2 Выбор и обоснование средств разработки

2.2.1 Выбор модели платы Arduino

Рассматривая оригинальные платы Arduino можно столкнуться с проблемой выбора. Главные отличия между ними – используемый микроконтроллер, а значит функциональность, внешние интерфейсы, а также объем памяти и физические размеры. Приоритетными критериями в выборе платы для данного проекта становятся:

1) Наличие достаточного числа цифровых выходов

2) Габариты

3) Цена

Сравнительная характеристика моделей Arduino представлена ниже в таблице 1.

Таблица 1 - Модели Arduino

Модель	Цена, руб	Кол-во выходов	Габариты, мм	Микро контроллер	Флеш память, кб	Дополнительно
Mini	990	14	43x18	ATmega328	32	Является аналогом Uno в компактном варианте. Для использования нужна пайка.

Uno	1150	14	68.6 x 53.3	ATmega328	32	Самая популярная версия базовой платформы Arduino
Leonardo	1150	20	68.6 x 53.3	Atmega32u4	32	Аналог Uno, но на микроконтроллере Atmega32u4
Micro	1190	20	48 x 17.7	Atmega32u4	32	Является аналогом Leonardo в компактном варианте. Нет гнезда для питания.
Nano	1990	14	43 x 18	ATmega328	32	Является аналогом Uno в компактном варианте. Нет гнезда для питания.
Ethernet	2290	14	68.6 x 53.3	ATmega328	32	Uno со встроенным Ethernet адаптером
Mega 2560	2290	54	101.6 x 53.3	ATmega256	256	Расширенная версия Arduino Uno. Больше контактов и аппаратных serial-портов.
Due	2590	54	101.6 x 53.3	AT91SAM3X8E	256	Мощная плата с новым процессором и большим числом контактов и входов

В ходе сравнения различных моделей Arduino была выбрана Arduino Uno, т. к. она обладает малыми размерами, приемлемой ценой и при этом тем же

функционалом, что и другие платы той же ценовой категории, но больших габаритов.

2.2.2 Выбор среды разработки Android

Существуют многочисленные IDE для Java-разработки, например, Android Studio, NetBeans, IntelliJ IDEA, Borland JBuilder и другие. Таким образом, прежде чем приступить к разработке приложения на базе ОС Android, необходимо подготовить инструментарий. Для разработки приложения на базе ОС Android я решил использовать готовое приложение Remote XY, так как она является наиболее оптимальной и удобной.

Преимущества Remote XY перед аналогами:

1) Привлекательная внешность.

Благодаря встроенному интерфейсу Remote XY приложения получаются очень привлекательными. Вид основного интерфейса системы представлен ниже в соответствии с рисунком 2.1.

Рисунок 2.1 - Интерфейс RemoteXY.

2) Простота использования.

В Remote XY можно разработать любой графический интерфейс управления, используя элементы управления, индикации и оформления в любой их комбинации. Размещая элементы на экране при помощи онлайн-

редактора можно разработать интерфейс под любую задачу.Вид основного рабочего стола системы представлен ниже в соответствии с рисунком 2.2.

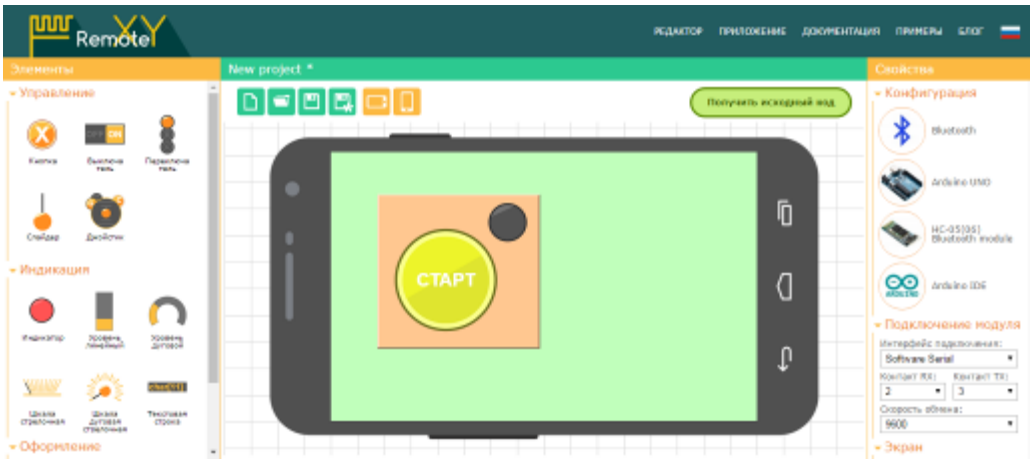

Рисунок 2.2 - Рабочий стол RemoteXY.

3) Отсутствие необходимости установки дополнительных сред разработки

При помощи одного мобильного приложения можно управлять большой гаммой устройств с разными графическими интерфейсами управления. Так как описание интерфейса хранится на борту микроконтроллерного устройства.

4)Собственная библиотека

У приложения RemoteXY есть собственная библиотека, упрощающая создание приложения. При написании кода программы, необходимо скачать и подключить библиотеку. Далее в коде программы просто пишем определенные команды, которые приложение будет распознавать. Например проверка нажатия кнопки в приложение:

```
if (RemoteXY.button_1!=0){
/*кнопка нажата*/
}
else{
/*кнопка не нажата*/}
```
[2]

3. Описание процессов

Перед тем, как программа Arduino (скетч) попадет в память микроконтроллера платы и начнет работать, должно произойти несколько скрытых от глаз пользователя процессов. Сначала среда разработки Arduino IDE выполняет небольшие преобразования кода скетча, чтобы он стал текстом, полностью совместимым со стандартом языка C или C++ (это два наиболее известных языка программирования в мире микроконтроллеров). Затем полученный текст программы передается компилятору avr-gcc, который переводит человеко-читаемый код программы в объектные машинные коды, пригодные (после дальнейшего преобразования - линковки) для выполнения ядром используемого микроконтроллера (обычно это Atmel AVR ATmega328). Затем объектный машинный код скетча комбинируется (этот процесс называется линковкой) с кодом функций из стандартных библиотек Arduino (эти библиотеки предоставляют множество функций, таких как базовые digitalWrite() или Serial.print()).

В результате получается один файл в формате Intel HEX, который должен быть записан в память микроконтроллера макетной платы Arduino. Обычно функцию программирования кода в память микроконтроллера берет на себя стандартный UART-загрузчик Arduino: код передается загрузчику через USB-соединение с компьютером (через специальную микросхему USB-UART, обычно компании FTDI). UART-загрузчик Arduino был изначально записан в память микроконтроллера платы специальным ISP-программатором.

Для обмена информации между платой Arduino и Android устройством необходимо использовать Bluetooth модуль. В данной работе применен модуль HC-06. Обмен информацией происходи по цифровым выводам RX и TX. Для подключения к приложению существует два метода подключения: через Software Serial и через Hardware Serial.

Software Serial позволяет подключить модуль к произвольным контактам микроконтроллера. Какие контакты использовать, необходимо указать в панели

настроек подключения модуля. Так же в панели настроек указывается скорость передачи данных для порта. Модуль HC-05(06) по умолчанию настроен на скорость 9600 бит/сек. Скорость работы модуля может быть изменена при помощи AT команд (требует специальных знаний), но если вы не изменяли скорость, установите значение по умолчанию, т.е. 9600 бит/сек.

Есть некоторые ограничения на использование контакта RX для плат Arduino. Ограничения связаны с поддержкой прерываний на соответствующих контактах микроконтроллера.

Arduino UNO и Nanодля RX нельзя использовать контакт 13(LED);

Arduino Mega и Mega 2560 для RX можно использовать только следующие контакты: 10, 11, 12, 13, 14, 15, 50, 51, 52, 53, A8(62), A9(63), A10(64), A11(65), A12(66), A13(67), A14(68), A15(69);

Arduino Leonardo и Micro для RX можно использовать только следующие контакты: 8, 9, 10, 11, 14, 15, 16;

Так же при использовании Software Serial вы должны принять следующие ограничения:

Нет возможности работы на больших скоростях передачи данных. Не рекомендуем использовать скорость соединения более 38400 бит/сек.

Некоторые библиотеки, которые так же используют прерывания, могут работать не корректно, или же их использование может сделать неработоспособным данный способ подключения. Например библиотека Servo будет подергивать сервоприводы.

Пример подключения модуля HC-06 для Software Serial к контактам 2(RX) и 3(TX) на рисунке. Обратите внимание, что необходимо контакты подключить перекрестием, т.е. контакт Arduino 2(RX) к контакту TX модуля, и контакт 3(TX) к контакту RX модуля.

Подключение через Hardware Serial позволяет подключить модуль к контактам микроконтроллера, поддерживающим один из аппаратных портов последовательного интерфейса. Для разных плат Arduino это разные порты и контакты.

Arduino UNO и Nano: Serial (RX-0, TX-1);

Arduino MEGA и MEGA2560: Serial (RX-0, TX-1), Serial1 (RX-19 и TX-18), Serial2 (RX-17 и TX-16), Serial3 (RX-15 и TX-14);

Какой порт использовать, а следовательно к каким контактам следует подключать модуль, необходимо указать в панели настроек подключения модуля. Следует так же иметь в виду, что для плат Arduino порт Serial задействован для программирования микроконтроллера, и если вы приняли решение использовать этот порт, при программировании необходимо отсоединять модуль HC-05(06). Хорошим решением будет использовать на платах Arduino MEGA любой порт, отличный от Serial.

Так же в панели настроек указывается скорость передачи данных для порта. Модуль HC-05(06) по умолчанию настроен на скорость 9600 бит/сек. Скорость работы модуля может быть изменена при помощи AT команд (требует специальных знаний), но если вы не изменяли скорость, установите значение по умолчанию, т.е. 9600 бит/сек.

Пример подключения модуля HC-05(06) для аппаратного Serial к контактам 0(RX) и 1(TX) показан в соответствии с рисунком 3.1.

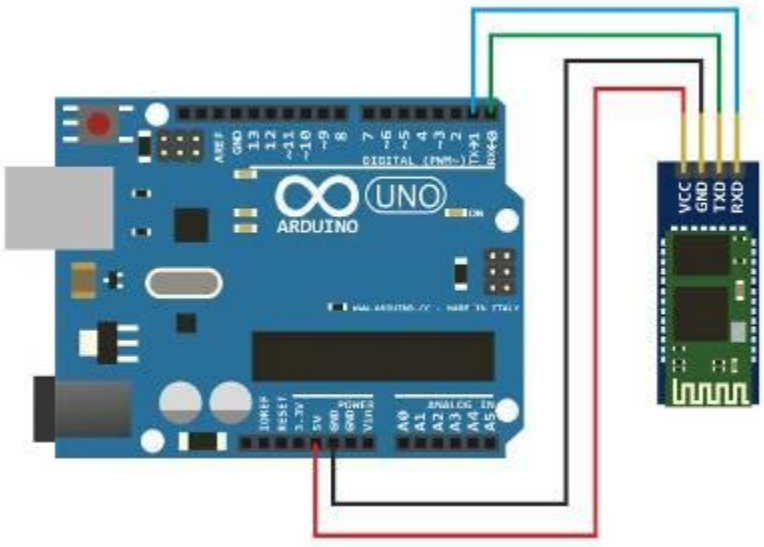

Рисунок 3.1 - Подключение HC-06.

Обратите внимание, что необходимо контакты подключить перекрестием, т.е. контакт Arduino 0(RX) к контакту TX модуля, и контакт 1(TX) к контакту RX модуля.

4. Разработка структур данных и алгоритмов

При разработке алгоритма запуска двигателя необходимо учесть ряд факторов, влияющих на правильную работу как приложения, так и автомобиля. Блок-схема алгоритма кнопки старт/стоп приведена в соответствии с рисунком в приложении 1. В алгоритме учтены многие обязательные факторы, например нажата ли педаль тормоза. Главным образом это необходимо для безопасности, а так же, чтобы не происходило случайного запуска двигателя. Так же в алгоритме учтено время накачки бензонасоса и автоматическое отключение стартера, при заведении двигателя. Автоматическое отключение стартера необходимо для того, чтобы стартер не крутил в пустую, тем самым мы продлеваем ему время работоспособности.

Алгоритм работы кнопки старт/стоп. При запуске программы в самом начале идет проверка, а не запущен ли двигатель? Если двигатель не запущен, то идет следующая проверка - нажата ли кнопка старт/стоп, если она не нажата, то мы переходим в режим ожидания и ждем нажатия. В том случае если сигнал пришел, то запускаем первое положение ключа и зажигание, а так же смотрим первый ли это у нас запуск. Если это не первый запуск и время между двумя нажатиями меньше 1,5 секунды, то глушим питание. Это означает, что мы не хотим запускать двигатель. Допустим у нас это первый запуск, тогда надо проверить нажата ли кнопка тормоза и горит ли индикатор заряда аккумулятора, дабы убедиться, что мы все-таки хотим завести двигатель. Если кнопка тормоза не нажата, то ничего не происходит.

Убедились, что тормоз нажат, тогда переходим к режиму запуска двигатель. Мы отключаем первое положение ключа, чтобы не потреблять большой ток, и запускаем стартер, до момента, когда погаснет индикатор заряда аккумулятора, означающий, что двигатель завелся. Если после трех

секунд индикатор не погас, то отключается все питание в качестве защиты. Допустим наш индикатор погас, тогда включаем первое положение ключа обратно и переходим в режим ожидания, где мы проверяем работает ли двигатель и ждем когда придет очередное нажатие с кнопки или загорится индикатор заряда аккумулятора. Когда, приходит нажатие, то глушим всю систему.

5. Используемые элементы

5.1 Микропроцессор Arduino

Существует большое количество вариантов микропроцессора Arduino и у каждого есть свои плюсы и минусы. В качестве оптимального микропроцессора для разработки программного продукта была выбрана плата Arduino UNO. Она обладает малыми размерами, приемлемой ценой и при этом тем же функционалом, что и другие платы той же ценовой категории, но больших габаритов.

Основной вид микропроцессора представлен ниже в соответствии с рисунком 5.1.

Рисунок 5.1 -Микропроцессор Arduino UNO.

5.2 Bluetooth модуль

С помощью Bluetooth модуля можно осуществлять радиосвязь между смартфонами или смартфоном и адаптером, встроенном в ноутбук или компьютер. Bluetooth-модули часто применяются для передачи данных в принтеры, считывания сигналов с GPS и для работы прочей высокотехнологичной аппаратуры. Для нас же, электронщиков, особый интерес вызывает возможность обеспечения связи двух микроконтроллеров по воздуху, когда один подключается к Bluetooth-мастеру, а другой соединяется с ведомым устройством. Такая связь по Bluetooth соответствует обычному последовательному интерфейсу UART с RXD и TXD линиями данных. Или же интересна возможность контролирования чего-нибудь, получая информацию об этом на терминальную программу своего карманного гаджета.

Существует некоторое количество типов модулей:

HC-03, HC-04(HC-04-M, HC-04-S) на чипе BC417143 – для промышленного применения;

- HC-05, HC-06(HC-06-M, HC-06-S) на чипе BC417143 – для коммерческого применения;

- HC-05-D, HC-06-D (с отладочной платой для оценки и тестирования);

- HC-07 – модуль с чипом CSR 41C6, предназначен для замены HC-06 (полностью с ним совместимый);

- HC-08 – модуль с ультранизким энергопотреблением и протоколом Bluetooth 4.0;

- HC-09 – самый новый модуль, предназначенный для замены HC-06 и HC-07.

Краткие характеристики модулей HC-03, HC-04, HC-05, HC-06

- чип Bluetooth – BC417143 производства CSR company (Cambridge Silicon Radio);

- протокол связи – Bluetooth Specification v2.0+EDR;

- радиус действия – до 10 метров (уровень мощности 2);

- совместимость со всеми Bluetooth-адаптерами, которые поддерживают SPP;

- объем flash-памяти (для хранения прошивки и настроек) – 8 Мбит;

- частота радиосигнала – 2.40 .. 2.48 ГГц;

- хост-интерфейс – USB 1.1/2.0 или UART;

- энергопотребление – ток в течение связи составляет 30-40 мА. Среднее значение тока около 25 мА. После установки связи потребляемый ток 8 мА Режим сна отсутствует.

Все модули HC-03, HC-04, HC-05, HC-06 выполнены на базе одного микроконтроллера BC417143, поэтому их (модулей) функционал отличается только его прошивкой.

Bluetooth-модули могут иметь два режима работы – master (ведущий) и slave (ведомый), причём для модулей HC-04 и HC-06 определённый режим уже установлен на заводе-изготовителе и меняться не может, разве что перепрошивкой (например HC-04-M – master или HC-06-S – slave). А вот модули HC-03 и HC-05 позволяют выбрать нужный режим работы с помощью AT-команд, причём изначально в этих модулях установлен режим slave. Bluetooth-модули HC-04 и HC-06, и, соответственно, HC-03 и HC-05 взаимно совместимы между собой по функциям. HC-04 и HC-06 - это ранние версии модулей, в которых, помимо невозможности изменения режима работы, имеются всего несколько рабочих AT-команд: установка имени Bluetooth-модуля (только для slave), пароля, скорости передачи данных и проверка номера версии. Набор команд в HC-03 и HC-05 является более широким, поэтому они более предпочтительны для радиолюбительского применения.

Модуль HC-07

Модуль HC-07 предназначен для замены (правда, уже устарел и сам) модуля HC-06 и практически полностью совместим с ним по функционалу и характеристикам, но всё же из-за небольшого отличия в AT-командах (они ещё более урезаны) я решил описать его отдельно, а не объединять с HC-04/HC-06. Характеристики модулей практически идентичны, за исключением того, что в

HC-07 применён другой чип Bluetooth – CSR BC04 производства CSR company.

Назначение и работа выводов модуля HC-07 соответствует модулям HC-03…HC-06, поэтому повторно приводить их описание не буду.

Модуль может иметь как режим – master (ведущий) так и slave (ведомый), изначально запрограммированный на заводе-изготовителе. Модули HC-07 изначально имеют такие настройки: скорость UART - 9600 (8 бит данных, без бита чётности, стоп бит (8N1)); пароль – 1234 или 0000.

Модуль HC-07, как и HC-04/HC-06, в режиме AT-команд находится до установки связи с другим Bluetooth-устройством (о чём свидетельствует мигающий LED). Непрерывное горение LED будет свидетельствовать о входе в режим передачи данных. Аналогично, команды в модулях HC-07 не имеют окончания CRLF, и в течение секунды позволяется отправка только одной AT-команды.

Модуль HC-08

Краткая характеристика модуля:

- чип Bluetooth – CC2540 производства Texas Instruments;
- протокол связи – Bluetooth v4.0 BLE;
- радиус действия – до 80 метров;
- совместимость со всеми Bluetooth-адаптерами, которые поддерживают SPP;
- частота радиосигнала 2.40 .. 2.48 ГГц;
- хост-интерфейс – USB 1.1/2.0 или UART;
- энергопотребление – в зависимости от режима работы и энергопотребления – от 0,4 мкА до 21 мА.

Это вообще очень хитрый модуль, т.к. за особенность ультранизкого потребления энергии при использовании нового протокола связи - Bluetooth v4.0 BLE (Bluetooth Low Energy), приходится расплачиваться и особым алгоритмом работы. Как сообщает Википедия, передатчик с таким протоколом включается только на время отправки данных, что обеспечивает возможность работы от одной батарейки типа CR2032 в течение нескольких лет, что

позволяет его активно применять для датчиков с батарейным питанием, домашних медицинских приборов, спортивных тренажеров.

Распиновка и назначение выводов модуля HC-08 совпадает с модулями HC-04/HC-06:

Модуль HC-09

Данный модуль полностью совместим с HC-06 и HC-07, однако, заводом-изготовителем он поставляется только в режиме slave-устройства. Модули HC-09 изначально имеют такие настройки: скорость UART - 9600 (8 бит данных, без бита чётности, стоп бит (8N1)); пароль – 1234. Список AT-команд незначительно отличается от модулей HC-06 и HC-07.

Для данного программного продукта был выбран модуль HC-06, так как он является оптимальным и удовлетворяем всем желаемым условиям для разработки.

Основной вид модуля представлен ниже в соответствии с рисунком 5.2.

Рисунок 5.2 -Модуль HC-06.

5.3 Блок реле

Для управления электроприборами которые питаются от бытовой электросети, люди пользуются различными клавишными выключателями и тумблерами. Чтобы управлять такими электроприборами с помощью микроконтроллера существует специальный тип выключателей — электромеханические реле. Relay Shield содержит четыре таких реле и позволяет Arduino управлять четырьмя электроприборами.

На Relay Shield устанавливается разное количество электромеханических реле, имеющих нормально замкнутый (normal closed, NC) и нормально разомкнутый (normal open, NO) контакты. Если на управляющей обмотке реле отсутствует напряжение, то между нормально замкнутым и коммутируемым контактами есть электрическая связь, а между нормально разомкнутым и коммутируемым — нет. При подаче напряжения на управляющую обмотку нормально разомкнутый контакт замыкается, а нормально замкнутый — размыкается.

Характеристики используемых реле:

Ток обмотки: 80 мА

Максимальное коммутируемое напряжение: 30 В постоянного тока; 250 В переменного тока

Максимальный коммутируемый ток: 5 А (NO), 3 А (NC)

Рекомендованная частота переключения: до 1 Гц

Время жизни: не менее 50000 переключений

Нагрузка к реле подключается через колодки под винт. Контакт от источника напряжения подключается к выводу COM, а нагрузка — к контакту NO или NC, в зависимости от задачи которую должно выполнять реле. Чаще всего реле используется для замыкания внешней цепи при подаче напряжения на управляющую обмотку. При таком способе даже если напряжение на Arduino по какой-то причине пропадёт, управляемая нагрузка будет автоматически отключена.

Для данного программного продукта был выбран Relay Shield с восьмью реле, так как это количество полностью удовлетворяет необходимые условия, для подключения.

Основной вид блока реле представлен ниже в соответствии с рис. 5.3.

Рисунок 5.3 - Relay Shield с восьмью реле.

5.4 Резисторы

Резисторы позволяют контролировать значения токов и напряжений в электрической цепи. Резисторы, например, обеспечивают режим смещения транзистора в усилителе электрических сигналов. Измеряя напряжение на резисторе, можно регулировать токи эмиттера и коллектора транзистора. С помощью резисторов выполняются делители токов и напряжений в измерительных приборах.

Электрические характеристики резистора взначительной мере определяются материалом, из которого он изготовлен, и его конструкцией.

При выборе типа резистора для конкретного применения обычно учитываются следующие параметры:

а) требуемое значение сопротивления (Ом, кОм, МОм),

б) точность (возможное отклонение, %, сопротивления от значения, .обозначенного на резисторе),

в) мощность, которую может рассеять резистор,

г)температурный коэффициент сопротивления резистора

$$R_T = R20[1 + \alpha(T - 20^o)],$$

где α — температурный коэффициент сопротивления.

Высокостабильные, малошумящие и высокоточные резисторы требуются лишь в специальных случаях. Например, они используются во входных каскадах измерительных усилителей малых сигналов. Широкое их применение ограничивается лишь высокой стоимостью этих приборов. Резисторы на основе угольного композита используются только в источниках электропитания и усилителях мощности.

Резисторы в керамических корпусах используются только в источниках электропитания и усилителях мощности. Резисторы в остеклованных корпусах находят широкие области применения, а резисторы в алюминиевой оболочке используются лишь в малосигнальных усилителях и измерительных приборах.

Номинальное значение сопротивления и точность резистора. На корпусе резистора всегда наносится ориентировочное значение его сопротивления. Так, резистор с маркировкой 100 Ом ± 10% может иметь любое сопротивление в пределах от 90 до 110 Ом. Сопротивление резистора, имеющего маркировку 100 Ом ± 1%, находится в интервале от 99 до 101 Ом.

Как правило, все выпускаемые промышленностью резисторы объединяются в серии. Количество номинальных значений сопротивлений в пределаходной̆ серии определяется принятой точностью. Например, для того чтобы перекрыть весь возможный̆ диапазон значений сопротивлений от 1 до 10 с помощью резисторов, имеющих точность ±20%, достаточно иметь набор из шести базовых значений (серия E6).

Серия E12 содержит 12 базовых значений сопротивлений с точностью ±10%. Серия E24 содержит 24 базовых значения сопротивлений резисторов с точностью ±5%.

В пределах каждой серии содержится 6 или 7 групп резисторов, сопротивления которых различаются в 10 раз. Это означает, что соответствующая группа сопротивлений получается умножением базового значения на 1, 10, 100, 1 кОм, 10 кОм, 100 кОм, 1 МОм [2].

В данном программном продукте используются резисторы с сопротивлением 1кОм и 10кОм. Резисторы с сопротивление 1кОм необходимы

для корректной работы диодов, предотвращая их перегорание. А резисторы с сопротивлением 10кОм подключаются к ноге кнопки и используются в качестве стягивающего резистора, создавая при этом нагрузку на цепи, которое необходимо для предотвращения короткого замыкания.

5.5 Диоды

Светодиод - далее - "СИД" (светоизлучающий диод), представляет собой полупроводниковый прибор, который при определенных условиях начинает светиться. Никакой связи с лампой накаливания он не имеет, в нем нет нити накаливания, нетвакуумной колбы. СИД устроен так: два прижатых друг к другу кристалла с определенными добавками, к ним приделаны две контактные ножки и все это залито оргстеклом. При подаче напряжения на ножки, частицы из двух кристаллов устремляются друг к другу и при соударении выделяют фотоны, т.е. выделяют свет.

У современных ярких СИД место вокруг кристалла покрывают люминесцентным веществом, которое тоже добавляет яркости. СИД, прежде всего, является диодом (главное свойство диода - пропускать ток только в одном направлении), поэтому зажигается он только при правильном подключении полярности. При ошибке СИД просто не включится и на его здоровье это не отразится. Чтобы не перепутать полярность, необходимо знать, что ножки у него разной длины. Это сделано специально - самая длинная ножка является плюсом. Если вы вдруг обрезали ножки, то если внимательно вглядеться внутрь конструкции, можно увидеть, что на одной ножке сделано подобие кроватки для кристалла, а от второй ножки к кристаллу подходит лишь тончайший проводок. Так вот, ножка с кроваткой является минусом. Еще один признак полярности - у минусовой ножки немного спилена юбочка.

СИД почти во всех случаях выгодно заменяет лампы накаливания. Основные плюсы - ничтожное потребление тока, более 60.000 часов гарантированной работы, не выделяет тепла, нечувствителен к вибрации,

выдерживает небольшие механические повреждения, отсутствует излучение в инфракрасном и ультрафиолетовом спектре (за исключением специальных).

Цвета светодиодов.

Светодиоды бывают почти всех цветов: красный, оранжевый, желтый, желтый, зеленый, синий и белый. Синего и белого светодиода немного дороже, чем другие цвета.

Цвет светодиодов определяется типом полупроводникового материала, из которого он сделан, а не цветом пластика его корпуса. Светодиоды любых цветов бывают в бесцветном корпусе, в таком случае цвет можно узнать только включив его.

Основной вид светодиодов представлен ниже в соответствии с рисунком 5.4.

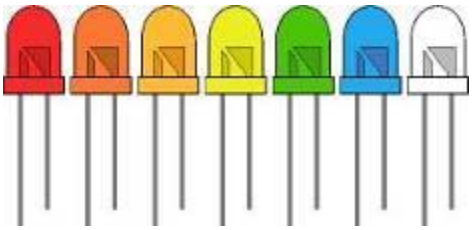

Рисунок 5.4 –Светодиоды.

Многоцветные светодиоды.

Устроен многоцветный светодиод просто, как правило это красный и зеленый объединенные в один корпус с тремя ножками. Путём изменения яркости или количества импульсов на каждом из кристаллов можно добиваться разных цветов свечения.

Основной вид многоцветного светодиода представлен ниже в соответствии с рисунком 5.5.

Рисунок 5.5 - Многоцветный светодиод.

Светодиод должен иметь резистор последовательно соединенный в его цепи, для ограничения тока, проходящего через светодиод, иначе он сгорит практически мгновенно

Схема подключения светодиода представлена ниже в соответствии с рисунком 5.6.

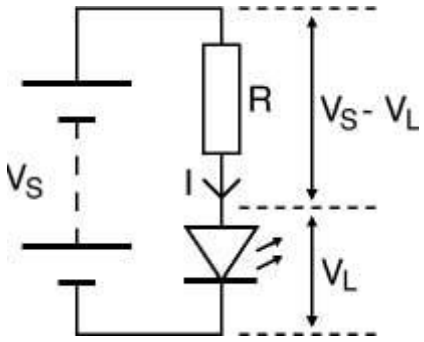

Рисунок 5.6 - Схема подключения светодиода.

Если размер сопротивления не получается подобрать точно, тогда возьмите резистор большего номинала. На самом деле разница не заметна. Яркость свечения уменьшится совсем незначительно.

В данном программном продукте используется восемь светодиодов, разных цветов. Они являются своего рода индикаторами того, что сигнал пришел в данную область схемы и служат лишь для тестирования работоспособности программного продукта. При дальнейшей установке программного продукта в автомобиль, светодиоды использоваться не будут, а будут заменены реальными электрическими механизмами.

5.6 Кнопки

Тактовая кнопка — простой, всем известный механизм, замыкающий цепь пока есть давление на толкатель.

Основной вид кнопки представлен ниже в соответствии с рисунком 5.7.

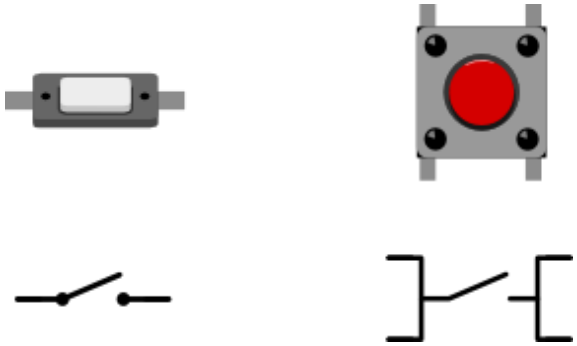

Рисунок 5.7 – Кнопка.

Кнопки с 4 контактами стоит рассматривать, как 2 пары рельс, которые соединяются при нажатии.

Эффект дребезга.

Эффект дребезга представлен ниже в соответствии с рисунком 5.8.

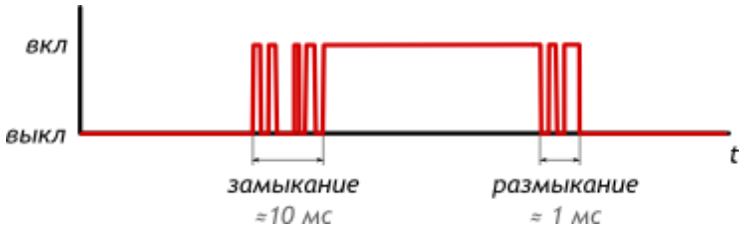

Рисунок 5.8 - Эффект дребезга.

При замыкании и размыкании между пластинами кнопки возникают микроискры, провоцирующие до десятка переключений за несколько миллисекунд. Явление называется дребезгом(англ. bounce). Это нужно учитывать, если необходимо фиксировать «клики».

Схема подключения.

Напрашивается подключение напрямую. Но это наивный, неверный способ.

Неправильная схема подключения кнопки представлен ниже в соответствии с рисунком 5.9.

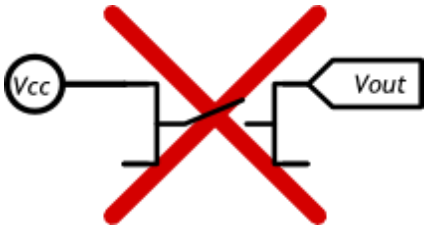

Рисунок 5.9 - Неправильная схема подключения кнопки.

Пока кнопка нажата, выходное напряжение Vout = Vcc, но пока она отпущена, Vout ≠ 0. Кнопка и провода в этом случае работают как антенна, и Vout будет «шуметь», принимая случайные значения «из воздуха».

Пока соединения нет, необходимо дать резервный, слабый путь, делающий напряжение определённым. Для этого используют один из двух вариантов.

Схема со стягивающим резистором.

Схема со стягивающим резистором представлена ниже в соответствии с рисунком 5.10.

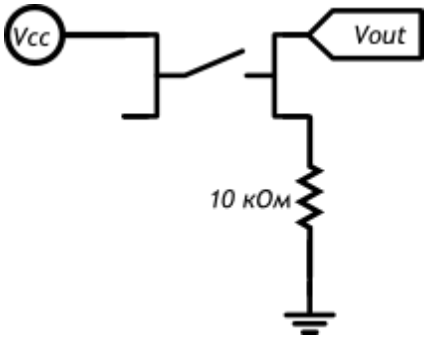

Рисунок 5.10 - Схема со стягивающим резистором.

Есть нажатие: Vout = Vcc.

Нет нажатия: Vout = 0.

Схема с подтягивающим резистором.

Схема с подтягивающим резистором представлена ниже в соответствии с рисунком 5.11.

Есть нажатие: Vout = 0.

Нет нажатия: Vout = Vcc.

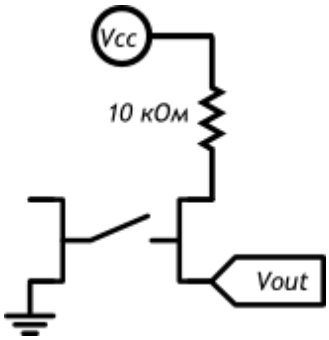

Рисунок 5.11 - Схема с подтягивающим резистором.

5.7 Макетная плата

Доска для прототипирования (так называемая breadboard или макетная плата) — незаменимая вещь для экспериментов с электроникой. Она позволяет быстро, удобно, без паяльника собирать электрические схемы.

При создании чего-то нового, в процессе проб и ошибок почти всегда приходится несколько раз корректировать схему. Если все детали сразу соединять пайкой, изменения становятся проблемой. Breadboard позволяет не думать об этом и вносить сколько угодно изменений.

На этой доске доступно 830 контактов. Четыре пары рельс по бокам предназначены для подключения питания и земли. Между ними — 126 групп соединённых между собой контактов.

Макетная плата представлена ниже в соответствии с рисунком 5.12.

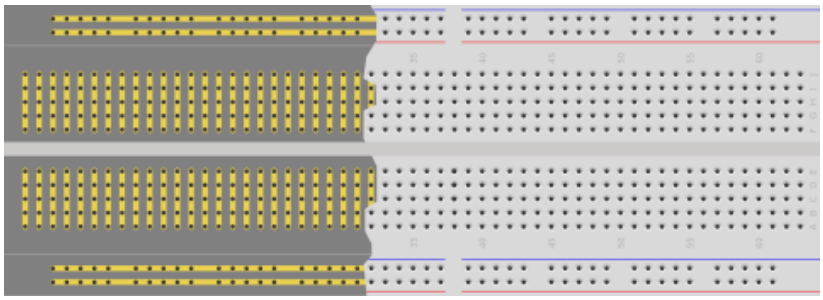

Рисунок 5.12 - Макетная плата.

Контакты можно соединять проводами с зачищенными концами, но значительно удобнее воспользоваться подготовленным набором перемычек или соединительными проводами.

По бокам breadboard’a расположены пазы, которые позволяют сцепить несколько макетных досок для увеличения рабочей площади. Основание доски сделано из самоклеящегося материала: если оторвать защитную плёнку, breadboard можно таким образом закрепить в устройстве.

Так же существует другие версии макетной платы такие как Breadboard mini или Proto Shield. Breadboard Mini представлена ниже в соответствии с рисунком 5.13, а Proto Shield представлена ниже в соответствии с рис. 5.14.

Рисунок 5.13 -Breadboard Mini.

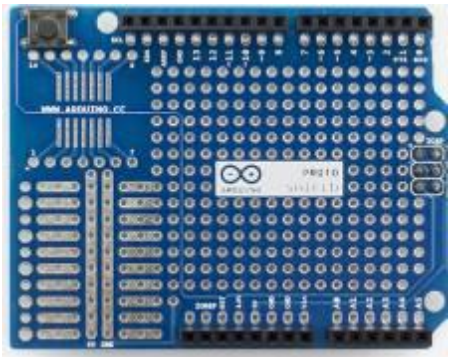

Рисунок 5.14 Proto Shield.

Для данного программного продукта используется стандартная макетная плата. Ее достаточно хватает для построения данной схемы [4].

6. Разработка программного продукта

6.1 Интерфейс приложения

Графический интерфейс приложения создается на официальном сайте приложения. Он позволяет разработать любой графический интерфейс управления, используя элементы управления, индикации и оформления в любой их комбинации. Размещая элементы на экране при помощи онлайн-редактора можно разработать интерфейс под любую задачу. Онлайн редактор размещен на сайтеremotexy.com.

Попав на страницу редактора первым делом необходимо настроить конфигурацию соединения. На момент разработки программного продукта были возможны три варианта соединения микропроцессора и аппарата на базе Android, предоставленные в соответствии с рисунком 6.1: соединение по Bluetooth, через Wi-Fi и через Ethernet. В данном программном продукте используется метод соединения через Bluetooth.

Рисунок 6.1 - Методы соединения

Следующим необходимым условием является выбор микропроцессора, под который будет написан код приложения. Весь возможный список поддерживаемых устройств предоставлен в соответствии с рисунком 6.2. В данном программном продукте используется микропроцессор Arduino UNO.

Рисунок 6.2 - Выбор микропроцессора.

Также во вкладке "Модуль" необходимо выбрать используемый модуль выбранного ранее способа соединения. Возможные модули предоставлены в соответствии с рисунком 6.3. Для данного программного продукта используется модуль HC-06.

Рисунок 6.3 - Выбор модуля соединения.

Завершающим этапом настройки конфигурации для разработки интерфейса приложения является выбор среды, под которую будет написан код программного продукта. На момент разработки были возможны два варианта Arduino IDE и FLProg IDE. Arduino IDE является стандартным официальным программным продуктом, предназначенным для разработки код под микропроцессоры Arduino. FLProg представляет собой Arduino IDE для визуального программирования одноименных плат Arduino. Основной целью работы является включение в круг пользователей плат Arduino людей незнакомых с программированием. Программа FLProg позволяет создавать прошивки для плат Arduino с помощью графических языков FBD и LAD, которые являются стандартом в области программирования промышленных контроллеров. При создании программы постарались максимально использовать наработки программистов Siemens, ABB, Schneider Electric в их средах программирования.

Данный программный продукт написан используя только Arduino IDE.

Вид основного интерфейса выбора среды разработки системы представлен ниже в соответствии с рисунком 6.4

Рисунок 6.4 -Выбор среды разработки.

После настройки конфигурации необходимо настроить подключение по Bluetooth. Определить через какой интерфейс Hardware Serial или Software Serial и через какие порты будет осуществляться обмен информацией. В данной работе используется Hardware Serial интерфейс и стандартные порты RX и TX.

Также необходимо настроить цвет фона экрана и его ориентация: горизонтальная, вертикальная или оба варианта.

Вид основного интерфейса настройки подключения модуля представлен ниже в соответствии с рисунком 6.5.

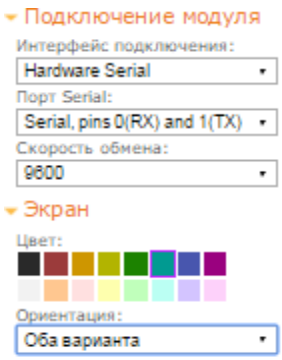

Рисунок 6.5 - Настройка подключения модуля и экрана.

Когда вся настройка будет закончена необходимо приступить к самому интерфейсу приложения и определить, какие элементы будут использоваться. Все доступные элементы на момент написания проекта предоставлены в соответствии с рисунком 6.6.

Рисунок 6.6 - Элементы интерфейса.

Из левого окна выбираем желаемые элементы и переносим их на основной экран. В данной работе используются пять кнопок, один индикатор, один выключатель и два лейбла. Также необходимо перенести все эти элементы на другую ориентацию экрана. Получившиеся интерфейсы можно наблюдать в соответствии с рисунком 6.7.

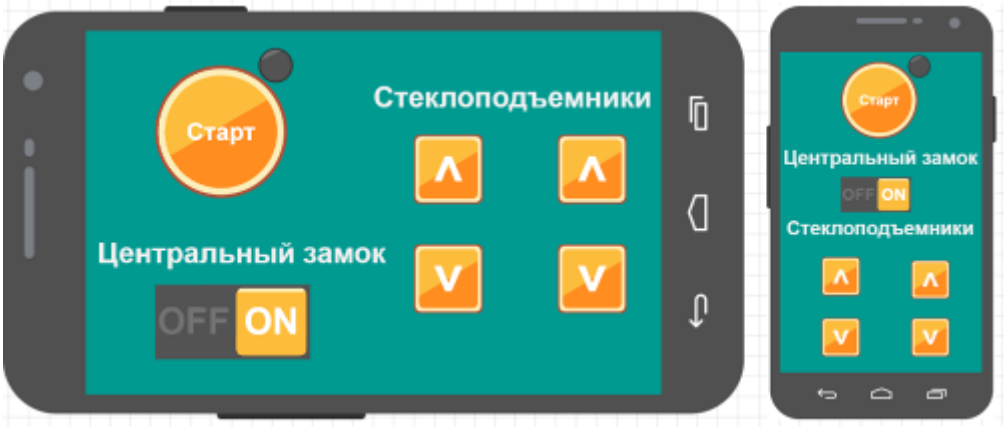

Рисунок 6.7 - Интерфейс приложения.

Завершив все операции с интерфейсом необходимо нажать кнопку "Получить исходный код", чтобы получить код, который необходимо вставить в основную программу Arduino.

6.2 Разработка схемы

Первым делом необходимо подключить Bluetooth модуль к микропроцессору Arduino. Выходы VCC и GND являются плюсом и минусом, соответственно их подключаем к основной плюсовой и минусовой шине. Выход TX на модуле подключаем ко входу RX микропроцессора, а выход RX модуля к TX входу микропроцессора. На этом подключение модуля завершено и можно проверить происходит ли подключение.

Следующим шагом необходимо подключить блок реле. У него так же есть порты VCC и GND. Их также подключаем к основным плюсовой и минусовой шинам соответственно. На блоке еще находятся порты IN1-IN8. Каждый порт связан со своим реле. Если на него приходит 1, то реле замыкается, а если 0, то размыкается. Все эти каналы модуля необходимо подключить к цифровым выходам микропроцессора. В данном программном продукте порты IN1-IN8 блока реле подключены к портам D11-D4 соответственно.

Подключение к самим реле может происходить в двух вариантах. Нормально замкнуто и нормально разомкнуто. Нормально замкнутый контакт соединен с общим контактом когда на реле не подается управляющее напряжение. А нормально разомкнутый контакт в этот момент не соединен с общим контактом. В данном проекте все реле подключены нормально разомкнуто. Далее к выходной каналу реле должен быть подключен соответствующий потребитель, но для проверки того, что сигнал пришел в данной работе служат светодиоды. Их подключение происходит обязательно через резистор с номинальным значением 1кОм. Подключаем резистор к выходу реле, к резистору в свою очередь подключаем анод (обычно длинная

ножка) светодиода. Катод подсоединяем к земле (Grd). И так необходимо повторить для всех восьми резисторов.

Следующий шаг это подключение кнопок. Вставляем в центре макетной платы кнопку таким образом, чтобы между парными ножками проходил желоб макетной платы. Далее соединяем цифровой вывод на Arduino с одной ножкой кнопки на плате. В данном проекте две кнопки подключены к пинам 3 и 12 на микропроцессоре. Эту же ножку кнопки, но с другой стороны соединяем с резистором 10кОм. После чего сам резистор соединяем с землей. Третью ножку кнопки соединяем к плюсовой шине. При таком подключении когда кнопка отключена, пин будет подключен к земле через резистор, сопротивление которого заведомо меньше внутреннего сопротивления пина. Поэтому наводка, попавшая на пин, стечет в землю. Если же подключен полезный сигнал (+5в) то он будет стекать в пин (незначительная часть сигнала стечет в землю через подтяжку).

Принципиальная схема подключения представлена в соответствии с рисунком 6.8.

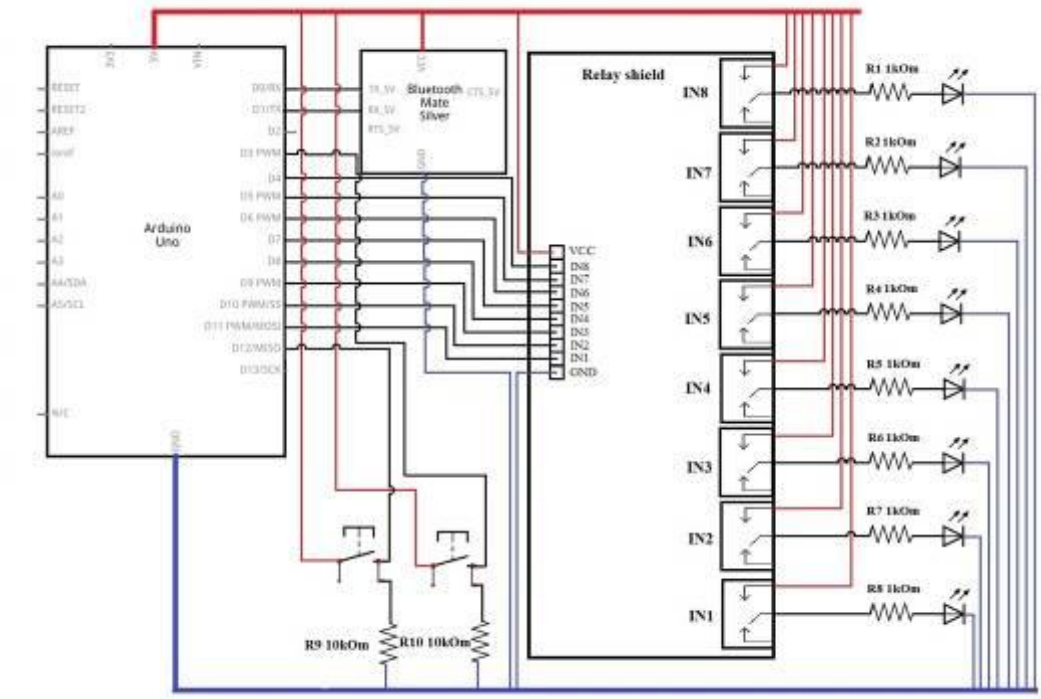

Рисунок 6.8 - Принципиальная схема.

6.3 Написание кода

Собрав всю схему и настроив интерфейс необходимо переходить к написанию самого кода приложения. Весь код приложения предоставлен в Приложении 2.

Первым делом нам необходимо скачать и подключить библиотеку RemoteXY так как без не будет происходить обмен информацией между микропроцессором и аппаратом на базе Android. Потом скопировать и вставить уже готовый код с сайта приложения RemoteXY. В этом коде в начале идет определения режима подключения модуля и микропроцессора, а также настройка соединения. Сам интерфейс приложения, который будет отображаться при подключении к модулю Bluetooth описан в переменной unsigned char RemoteXY_CONF[10]. Эта часть содержит массив байтов, который расскажет мобильному приложению о том, как необходимо построить графический интерфейс. Она содержит описание всех элементов, размещенных на интерфейсе, их позицию, цвет, другие настройки. Эти данные имеют шифрованный вид, который трогать не рекомендуется.

Все элементы интерфейса управления предназначенные для взаимодействия с интерфейсом описаны в struct{}. Структура содержит определение данных каждого элемента интерфейса.

Функция setup () запускается один раз, после каждого включения питания или сброса платы Arduino. В теле данной функции пишется код для инициализации переменных, установки режима работы цифровых портов, и т.д. Функция setup() обязательно должна содержать код для запуска инициализации библиотеки RemoteXY. Это вызов конструктора RemoteXY_Init (). В функции setup() можно произвести начальную инициализацию всех элементов управления, в том числе установить начальные положения переключателей, выключателей, слайдеров, джойстиков, и элементов отображения, если это необходимо.

Функция loop() в бесконечном цикле последовательно раз за разом

исполняет команды, которые описаны в её теле. Т.е. после завершения функции снова произойдет её вызов. Функция loop() имеет вызов обработчика RemoteXY_Handler (). Обработчик RemoteXY_Handler должен вызываться в каждом цикле программы. Вызов необходим для того, что бы библиотека RemoteXY могла обработать очередную порцию данных об элементах управления, переданных со смартфона и предать на смартфон новые данные о состоянии элементов индикации.

После основного кода в данной работе созданы вспомогательные функции:

void check_start() - основная функция проверки запуска двигателя

void do_start() - функция запуска двигателя

void check_for_shutdown() - функция запущенного двигателя

void do_shutdown() - функция отключения питания

7. Тестирование программного продукта

7.1 Функциональное тестирование

При функциональном тестировании исследуются работа всех функций программного продукта и проверяется соответствие полученных результатов ожидаемым, т.е. рассматриваются системные характеристики программ, но игнорируется их внутренняя логическая структура.

Проверка работоспособности кнопок. При нажатии на кнопку поднятия левого стеклоподъемника соответствующий диод загорается, а при отпускании диод перестает гореть. Также происходит с кнопкой опускания и кнопками правого стеклоподъемника. Работоспособность показана в соответствии с рисунком 7.1.

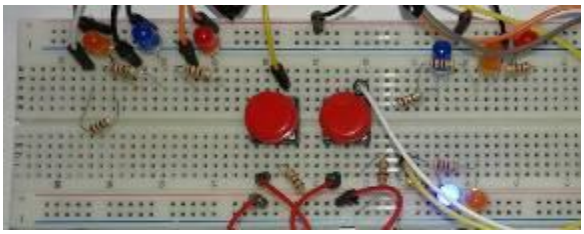

Рисунок 7.1 - Поднятие стеклоподъемника.

При включении переключателя центрального замка, соответствующий диод загорается, а при отключении перестает гореть. Работоспособность показана в соответствии с рисунком 7.2.

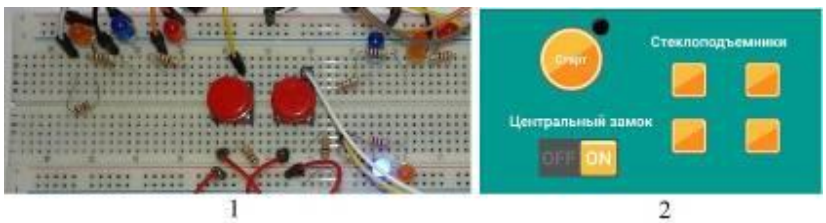

Рисунок 7.2 Включен центральный замок: 1 - на макетной плате; 2 - в приложении.

Нажав на кнопку "Старт" в приложении загорается красный индикатор, информирующий о том, что зажигание включено.Результат можно наблюдать в соответствии с рисунком 7.3. Соответствующие два диода на макетной плате тоже загораются, информируя о том, что пришли оба необходимых сигнала.

Рисунок 7.3 Включено зажигание: 1 - на макетной плате; 2 - в приложении.

Пробуем не нажав ни на одну кнопку на макетной плате запустить двигатель- изменений не происходит. Нажав два раза быстро на кнопку "Старт" красный индикатор перестает гореть и все соответствующие диоды на плате тоже перестают гореть. Опять запустив зажигание пробуем, нажав на макетной плате на кнопки отвечающие за педаль тормоза и индикации зарядка аккумулятора, запустить двигатель. После трех необходимых секунд видим в соответствии с рисунком 7.4, что сигнал на стартер пришел, соответственно машина пытается завестись.

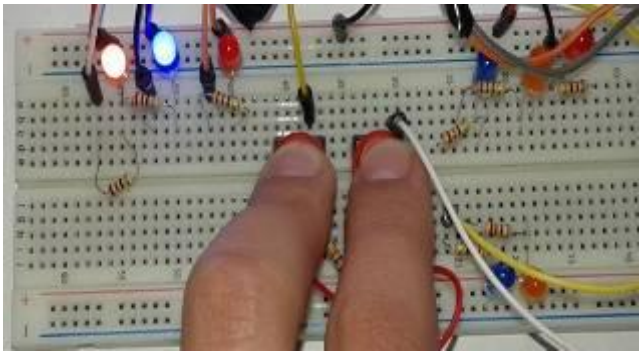

Рисунок 7.4 - Запущен стартер.

Не отпуская клавиш ждем пять секунд. Через пять секунд все питание отключается, значит защита от перекручивания стартера работает исправно и это не даст нам пытаться запустить двигатель впустую. Опять зажав все клавиши и нажав кнопку "Старт" пытаемся завести двигатель, только в этот раз в момент когда стартер начнет крутить, мы отпустим клавишу индикации заряда батарее на макетной плате, информируя о том что машина завелась. Проделав эти действия в приложении индикатор загорелся зеленым цветом, сообщая что машина заведена. Результат можно увидеть в соответствии с рисунком 7.5.

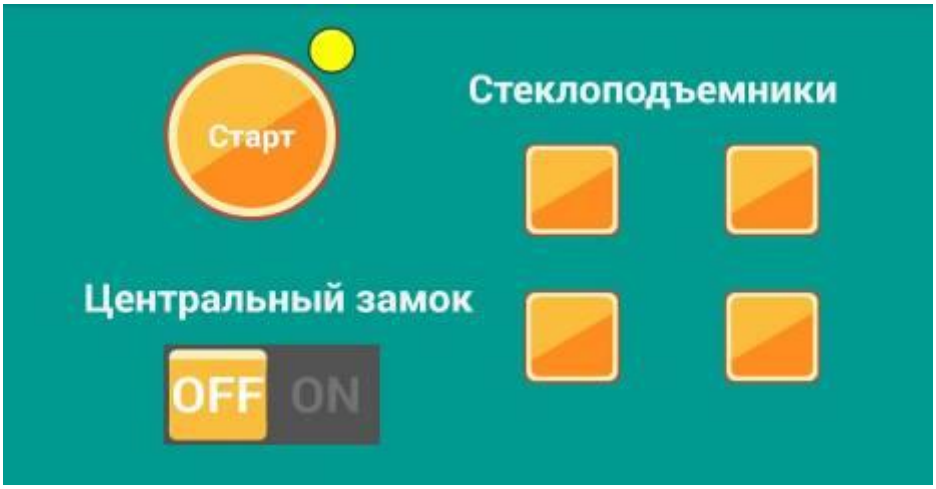

Рисунок 7.5 - Машина заведена.

Нажав еще раз на кнопку "Старт" происходит запланированное отключение питания.

Таким образом, проведя функциональное тестирование было проверено, что все заявленные функции работают так, как и должны работать.

7.2 Структурное тестирование

При структурном тестировании (тестировании «белого ящика») известна внутренняя структура программы, исследуются внутренние элементы программы и связи между ними. Здесь проверяется корректность построения

элементов программы и правильность их внутреннего взаимодействия друг с другом.

При выполнении структурного тестирования была проверена работоспособность всех функций и веток программного кода. Ошибок не обнаружено.

7.3 Альфа-тестирование

Альфа-тестирование - имитация реальной работы с системой штатными разработчиками, либо реальная работа с системой потенциальными пользователями/заказчиком.

При проведении некоторого количества раз альфа-тестирования была имитирована работа программного продукта. В результате ошибок не было обнаружено.

7.4. Охрана труда

Трудовой процесс осуществляется в определенных условиях производственной среды, которые характеризуются совокупностью элементов и факторов материально-производственной среды. Рассмотрим условия труда пользователя ПЭВМ, который является разработчиком программного продукта. Для работы используется следующее оборудование: Ноутбук Samsung RV-520-S01. 15 рабочих мест находятся в 15 помещениях, длина которых 12 м, ширина – 7 м, высота – 6 м. Уровень шума в помещениях 55 дБ, освещенность рабочего места составляет 400 лк. Воздух рабочей зоны имеет следующие параметры: температура – 19°С, скорость воздуха – 0,3 м/с, влажность – 76%. Продолжительность сосредоточенного наблюдения составляет 40%.

Специфика использования ПЭВМ состоит в том, что в процессе диалога человека и машины пользователь воспринимает интеллектуальную машину как

равноправного собеседника. Поэтому возникает много совершенно новых психологических и психофизиологических проблем, суть которых нужно учитывать при проектировании трудового процесса. Другой особенностью является значительная информационная нагрузка. Значительная нагрузка на центральную нервную и зрительную системы вызывает повышение нервно-эмоционального напряжения, и, как следствие, негативно влияет на сердечно-сосудистую систему. Важной стороной функционирования организма пользователя является влияние на него комплекса факторов трудовой среды, включающих действие электромагнитных волн разных частотных диапазонов, статического электричества, шума, микроклиматических факторов и др. Воздействие этого специфического комплекса может оказать на здоровье человека отрицательное влияние. При работах с использованием компьютеров возникает целый ряд эргономических проблем, решение которых может значительно снизить нагрузку. В этом случае имеются в виду только вопросы конструирования рабочего места пользователя и не охватываются вопросы формирования рационально построенных символов на экране и других, изменение которых возможно только при конструировании новой техники. Работа пользователя ЭВМ чаще всего проходит при активном взаимодействии с другими людьми. Поэтому возникают вопросы межличностных взаимоотношений, включающие как психологические, так и социально-психологические аспекты. Таким образом, на пользователя ЭВМ воздействуют 4 группы факторов трудовой среды: физические, эргономические, информационные и социально-психологические [25-26].

«Опасные и вредные производственные факторы. Классификация» все производственные факторы делятся на опасные и вредные факторы. Опасные и вредные производственные факторы в свою очередь делятся на физические, химические, биологические и психофизиологические факторы.

Опасный производственный фактор – фактор, воздействие которого может привести к травме или другому резкому внезапному ухудшению здоровья. Вредный производственный фактор – это фактор, воздействие

которого на работающего может привести к снижению работоспособности человека, заболеванию или профессиональному заболеванию.

Пользователи ПЭВМ в основном подвергаются воздействию физических и психофизиологических производственных факторов [26].

При работе с компьютером на человека могут воздействовать следующие опасные производственные факторы:

– поражение электрическим током;

– возникновение пожара;

– возможность механического травмирования;

– ожоги в результате случайного контакта с горячими поверхностями внутри лазерного принтера.

К вредным физическим производственным факторам относятся:

– повышенный уровень электромагнитного излучения;

– повышенный уровень статического электричества;

– повышенные уровни запыленности воздуха рабочей зоны;

– повышенное содержание положительных и отрицательных ионов в воздухе рабочей зоны;

– пониженная или повышенная влажность и подвижность воздуха рабочей зоны;

– повышенный уровень шума;

– нерациональная организация освещения рабочего места.

К психофизиологическим производственным факторам относятся:

– напряжение зрения;

– напряжение внимания;

– интеллектуальные и эмоциональные нагрузки;

– длительные статические нагрузки;

– монотонность труда;

– большие информационные нагрузки;

– нерациональная организация рабочего места (эргономические

факторы).

Вероятность воздействия химических и биологических факторов незначительная, но она значительно возрастает в переполненных и неправильно вентилируемых помещениях.

Важнейшими факторами являются электромагнитные поля в диапазоне от 3 Гц до 300 МГц, электростатические поля, напряжение зрения, большие нагрузки различного характера. Рассмотрим их более подробно.

ПЭВМ является источником нескольких видов электромагнитных полей и излучений: мягкого рентгеновского, ультрафиолетового, инфракрасного, видимого, низкочастотного, сверхнизкочастотного и высокочастотного. ЭМП негативно влияют на центральную нервную систему, вызывая головные боли, головокружения, тошноту, депрессию, бессонницу, отсутствие аппетита, возникновение синдрома стресса.

Низкочастотное ЭМП может явиться причиной кожных заболеваний (угревая сыпь, экзема, розовый лишай и др.), болезней сердечнососудистой системы и кишечно-желудочного тракта; оно воздействует на белые кровяные тельца, что приводит к возникновению опухолей, в том числе и злокачественных.

Основным источником электростатического поля (ЭСП) является положительный потенциал, подаваемый на внутреннюю поверхность экрана для ускорения электронного луча. ЭСП образуется за счет разности потенциалов экрана монитора и человека. На его величину оказывают существенное влияние потенциалы окружающих предметов и влажность воздуха (при влажности выше 50% ЭСП практически отсутствует). Напряженность поля может колебаться от 8 до 75 кВ/м. Заметный вклад в общее ЭСП вносят электризующиеся от трения поверхности клавиатуры и мыши. Электростатическое поле большой напряженности способно изменять и прерывать клеточное развитие, а также вызывать катаракту с последующим помутнением хрусталика.

Работа на ПЭВМ предполагает визуальное восприятие отображенной на

экране монитора информации, поэтому значительной нагрузке подвергается зрительный аппарат. Симптомы нарушения зрения можно условно разделить на две группы:

– глазные симптомы (боль, раздражение, жжение, краснота, зуд);

– зрительные симптомы (пелена перед глазами, двоение или мелькание).

По данным ВОЗ глазные и зрительные нарушения наблюдаются у 40–92 % пользователей ПЭВМ время от времени, а у 10–40 % – ежедневно.

Можно выделить следующие основные нарушения здоровья пользователей ПЭВМ [25-26]:

– зрительный дискомфорт и болезни органов зрения;

– перенапряжение опорно-двигательной системы;

– расстройства ЦНС и болезни сердечнососудистой системы;

– заболевания кожи;

– нарушение репродуктивной функции.

Кроме того, выявлено негативное влияние на другие системы организма – снижение иммунитета, атеросклероз, аритмия, гипертония, инфаркт миокарда, болезни органов пищеварения, застойные процессы в области малого таза и др. Нарушения здоровья и заболевания пользователей ПЭВМ являются, как правило, результатом воздействия не какого-либо отдельного фактора, а всего комплекса. Так, поражения кожи многие авторы связывают с наличием электростатического поля и воздействием психоэмоционального стресса, гинекологические нарушения – с комплексным влиянием электромагнитных полей, стресса, застойных явлений и других компонентов трудовой среды.

Представляет практический интерес комплексная оценка условий труда. Одним из широко используемых аналитических показателей условий труда является категория тяжести труда. Категория тяжести труда характеризует состояние организма человека, которое формируется под влиянием условий труда. Выполним количественную оценку условий труда на рассматриваемом рабочем месте. Каждый элемент условий труда оценим по шести бальной

шкале [27-29]. Результаты оценки приведены в таблице 18.

Таблица 18 – Балы оценки факторов тяжести труда

№	Элемент условий труда, единицы измерения	Обозначение	Значение	Оценка фактора, баллы
1	Температура, оС	X1	19	1
2	Скорость ветра, м/с	X2	0,3	2
3	Влажность воздуха, %	X3	76	4
4	Освещенность, лк	X4	400	1
5	Продолжительность сосредоточенного наблюдения, %	X5	40	2
6	Уровень шума, дБ А	X6	55	4

Интегральную балльную оценку тяжести труда определяем по формуле

$$И_т = 10\left(X_{оп} + \overline{X}\ \frac{6 - X_{оп}}{6} \right) \quad (31)$$

где $X_{оп}$ – элемент условий труда, который получил наибольшую оценку;

$\overline{X}$ - средний балл всех активных элементов условий труда кроме определяющего Хоп, который определяется по формуле 32:

$$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n-1} \quad (32)$$

где $\sum_{i=1}^{n} X_i$ - сумма всех элементов кроме определяющего $X_{оп}$;

n - количество учтенных элементов условий труда.

Согласно данным таблицы 17 элементы условий труда оцениваются, соответственно, X_1 =1, X_2=2, X_3=4, X_4=1, X_5=2 и X_6=4. Элементами условия труда, которых получили наибольшую оценку являются влажность воздуха и уровень шума: $X_{оп}$ = 4. Средний балл всех активных элементов условий труда кроме определяющих Хоп по формуле (32) составляет:

$$\overline{X} = \frac{1+2+1+2+4}{6-1} = 2$$

.

Интегральная балльная оценка тяжести труда соответственно равна:

$$И_Т = 10\left(4 + 2\ \frac{6-4}{6}\right) = 46{,}6$$

.

Интегральная оценка тяжести труда в 46,6 балла отвечает IV категории тяжести труда.

Степень утомления человека в условных единицах рассчитывают по формуле:

$$Y = \frac{Ит - 15{,}6}{0{,}64}, \qquad (33)$$

где Y – уровень утомления, условные единицы;

15,6 и 0,64 – коэффициенты регрессии.

Уровень утомления по формуле (33) составляет:

$$Y = \frac{46{,}6 - 15{,}6}{0{,}64} = 48{,}4$$

.

Трудоспособность человека определяется как величина противоположная утомлению по формуле 34:

$$R = 100 - Y, \qquad (34)$$

где R – трудоспособность человека, условные единицы.

Рассчитаем трудоспособность по формуле (35):

$$R = 100 - 48{,}4 = 51{,}6$$

Оценка условий труда показала, что они не являются комфортными (IV категория тяжести труда). Следовательно, необходимо разработать мероприятия по обеспечению безопасных и комфортных условий труда.

ЗАКЛЮЧЕНИЕ

Существует большое количество микропроцессоров и платформ осуществляющие схожие функции с Arduino: такие как Raspberry Pi, Iskra Neo, Strela, STM, Teensy, Beagly Bone и другие. Все эти устройства объединяет разрозненную информацию о программировании и заключает ее в легкую в использовании сборку. Однако Arduino имеет ряд преимуществ перед остальными для использования преподавателями, студентами и простыми пользователями:

Самое главное это низкая стоимость платы, по отношению к остальным.

Второе это кросс-платформенность. Arduino работает почти со всеми операционными системами Windows, Macintosh и Linux.

В третьих среда программирования проста и понятна. Она подходит как для начинающих пользователей, так и для опытных. Также эта среда очень полезна как для преподавателей, так и для студентов, потому что все программы пишутся на языке C/C++.

В ходе выполнения программного продукта был спроектировано и разработано приложение для взаимодействия с микропроцессором Arduino. Была разработана и собрана принципиальная схема, а так же написан необходимый код, удовлетворяющий потребностям выпускной квалификационной работы.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. ПостановлениеПрезидента Республики Узбекистаноб образовании центра передовых технологий при министерстве инновационного развития республики Узбекистан №ПП-3674 19.04.2018
2. Евстифеев А.В. «Микроконтроллеры AVR семейства Mega» – Москва – Издательский дом «Додэка - XXI», 2007.-595с.
3. Б.Ф. Бессарабов, В.Д. Федюк, Д.В. Федюк Справочник "Диоды, тиристоры, транзисторы и микросхемы широкого применения"- Изд. «Воронеж», 1994-320с.
4. Бабин А.И., Шипилов. Электрические машины и схемы управления: Методические указания для студентов очной формы обучения по дисцеплине автоматизированный электропривод. 2010. - 65 с.
5. Москаленко В.В. Автоматизированный электропривод: Учебник для вузов.- М.: Энергоатомиздат, 2008. - 416 с.
6. Чиликин М.Г.,Сандлер А.С. Общий курс электропривода: учебник для вузов. - М.:Энергоиздат, 2007. - 567 с.
7. Белов А.В. Самоучитель разработчика устройств на микроконтроллерах AVR. - Наука и техника, 2008г.
8. Фрунзе А. В. Микроконтроллеры? Это же просто. - Додэка-XXI, 2007г. Мартин Т. Микроконтроллеры ARM7. - Додэка-XXI, 2006г.
9. Таненбаум Э. Архитектура компьютера. -- СПб: Питер, 2007г.
10. Arduino документация // Официальный сайт приложения RemoteXY
11. Резисторы // Школа для электрика. Среда разработки Arduino // Официальный российский сайт Arduino [Электронный ресурс] - https://www.arduino.cc/
12. Роботы робототехника микроконтроллеры [Электронный ресурс] - http://www.myrobot.ru/articles
13. Руководство пользователя Micro Camp: инструкция по сборке и программированию.

14. Мельникова В.В. Защита информации в компьютерных системах. Финансы и статистика , 1997г. – 866 с.
15. Безопасность жизнедеятельности, А.Кудратов, Т.Ганиев и др. Тошкент, Алоқачи, 2005

ПРИЛОЖЕНИЕ

Основной код программы

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
#include <EEPROM.h>
#include <SoftwareSerial.h>
#include <Keypad.h>
#define SPEAKER 11
#define BEATTIME 200
unsigned long currentTime;  //Текущее время в милисекундах
unsigned long lastTime_Get, lastTime_Set, lastTime_NS ;
struct history{
 uint8_t pid;
 float litr;
 uint8_t davl;
};

void chageside(bool fl=true);
void cllcd(byte row=0, byte stcol=0, byte mlen=16);
void cmdSendLitr(int pid, unsigned long long int litr, long int price=0, unsigned short ind=0);
void cmdGetStatus(int pid, uint8_t ind=0, uint8_t command=0x01);
void cmdSetPrice(int pid, uint8_t ind, unsigned int price);
void cmdSetDencity(int pid, uint8_t ind, unsigned long int price);
void addhist(float val, uint8_t ival, uint8_t pid);

unsigned long int  Conv4ByteToInt(uint8_t *bpData);
unsigned long int  Conv3ByteToInt(uint8_t *bpData);
unsigned long int  Conv2ByteToInt(uint8_t *bpData);
String pname(uint8_t pnum=0);
void printLitr(float vl);
void getmenuval();
```

void setmenuval();

void savemenuval();

void setdefault(bool aside=true);

//Pist storona A

uint8_t stpid1=0x0A;

uint8_t pid1=stpid1;//stpid

bool ispause1=false;

bytemindex1=4; //1- Введите объем 2-Введите сумму, 3-

unsigned long int mval1=0; //Summa ili obyom

unsigned int mprice1[5];

bool iskeyboard1;

bool tiptopliva1;

bool isproc1=false;

bool ststop1=false;

bool stpause1=false;

history hist1[10];

bool comflag1=true;

uint8_t pr_ind1=pid1;

uint8_t counter1=0;

uint8_t menucode1=0;

unsigned int menuval1=0;

unsigned int iskey1=0;

uint8_t ival1=0;//Davlenie

bool netsvyazi1=false;

uint8_t svyazkol1=0;

uint8_t kolruk1=5;

//Pist storona B

uint8_t stpid2=0x0F;

uint8_t pid2=stpid2;//stpid

bool ispause2=false;

bytemindex2=4; //1- Введите объем 2-Введите сумму, 3-

unsigned long int mval2=0; //Summa ili obyom

```
unsigned int mprice2[5];
bool iskeyboard2;
bool tiptopliva2;
bool isproc2=false;
bool ststop2=false;
bool stpause2=false;
history hist2[10];
bool comflag2=true;
uint8_t pr_ind2=pid2;
uint8_t counter2=0;
uint8_t menucode2=0;
unsigned int menuval2=0;
unsigned int iskey2=0;
uint8_t ival2=0;//Davlenie
bool netsvyazi2=false;
uint8_t svyazkol2=0;
uint8_t kolruk2=5;

//Ukazateli
uint8_t *stpid=&stpid1;
uint8_t *pid=&pid1;//stpid
bool *ispause=&ispause1;
byte *mindex=&mindex1; //1- Введите объем 2-Введите сумму, 3-
unsigned long int *mval=&mval1; //Summa ili obyom
unsigned int *mprice=mprice1;
bool *iskeyboard=&iskeyboard1;
bool *tiptopliva=&tiptopliva1;
bool *isproc=&isproc1;
bool *ststop=&ststop1;
bool*stpause=&stpause1;
history *hist=hist1;
bool *comflag=&comflag1;
```

```
uint8_t *pr_ind=&pr_ind1;
uint8_t *counter=&counter1;
uint8_t *menucode=&menucode1;
unsigned int *menuval=&menuval1;
unsigned int *iskey=&iskey1;
uint8_t *ival=&ival1;//Davlenie
bool *netsvyazi=&netsvyazi1;
uint8_t *svyazkol=&svyazkol1;
uint8_t *kolruk=&kolruk1;

const int stmem=50;
bool sidea=true;
uint8_t port_otpr_tmp=0x0A;
uint8_t getport=0x0A;
const byte ROWS = 4;
const byte COLS = 4;
unsigned int ind=0;
bool flag=true;
byte cnt=0;

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

//byte rowPins[ROWS] = {9, 2, 3, 5};
//byte colPins[COLS] = {4, 6, 7, 8};

byte rowPins[ROWS] = {6, 7, 8, 9};
byte colPins[COLS] = {2, 3, 4, 5};
```

```
byte rowPins1[ROWS] = {10, 11, 12, 13}; //12->a6
byte colPins1[COLS] = {A3, A2, A1, A0}; //13->a7
Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
Keypad customKeypad1 = Keypad(makeKeymap(hexaKeys), rowPins1, colPins1, ROWS, COLS);

const int RX1 = 0;
const int TX1 = 1;
SoftwareSerial getSerial(RX1, TX1);
//LiquidCrystal lcd(A0, A1, A2, A3, A4, A5);
LiquidCrystal_I2C lcd1(0x27,16,2);
LiquidCrystal_I2C lcd2(0x23,16,2);
LiquidCrystal_I2C *lcd=&lcd1;

uint8_t mbytes[50];
uint8_t mlen=0;
uint8_t mbytesout[50];
uint8_t mlenout=0;

//===========Tekstovaya informatsiya nachalo==========
String spr_menu="======MEH\260======";
String spr_price="\341EHA: ";
String spr_netsvyazi="===HET CB\261\244\245!===";
String spr_lit="\247\245T:";


//==========Konec tekstovaya informatsiya============
void setup(){
 //pinMode (A3, OUTPUT);
 //digitalWrite(A3, LOW);
```

```
//lcd->begin(16, 2);
 //lcd->init();                    // Инициализация lcd
// lcd->backlight();
lcd2.init();
//lcd2.setBacklightPin(3,POSITIVE);
//lcd2.setBacklight(1);
lcd2.backlight();// Включаем подсветку дисплея

lcd1.init();
lcd1.backlight();// Включаем подсветку дисплея

tone(SPEKER, 262, BEATTIME);
for(int i=0;i<10;i++){
 hist1[i].litr=0;
 hist1[i].pid=0;
 hist1[i].davl=0;

 hist2[i].litr=0;
 hist2[i].pid=0;
 hist2[i].davl=0;
 }

for(int i=0;i<5;i++) EEPOM.get(i*2,mprice1[i]);
 EEPROM.get(11,iskeyboard1);
 EEPROM.get(12,tiptopliva1);
 EEPROM.get(13,stpid1);
 EEPROM.get(14,kolruk1);
 pid1=stpid1;
for(int i=0;i<5;i++) EEPROM.get(stmem+i*2,mprice2[i]);
 EEPROM.get(stmem+11,iskeyboard2);
 EEPROM.get(stmem+12,tiptopliva2);
 EEPROM.get(stmem+13,stpid2);
```

```
 EEPROM.get(stmem+14,kolruk2);
 pid2=stpid2;
delay(3000);
 chageside(true);
setdefault();
 chageside(false);
setdefault();
 getSerial.begin(9600);
}

//==========Nachalo tela programmi=================================
void loop(){
 char mkey = customKeypad.getKey();
 if(mkey){
  chageside(true);
 }
else{
  mkey = customKeypad1.getKey();
  if(mkey)chageside(false); //lcd=&lcd2;
 }
//--------Poluchenie ceni--------------
// if(pr_ind<(stpid+5)) cmdGetStatus(pr_++ind, ++ind, 0x02); //Poluchenie ceni

 if (mkey){
tone(SPEAKER, 494, BEATTIME);

  // ----------Menu/Pause----------
  if(mkey=='C'){
   if(*isproc){
    if(!*tiptopliva){
cmdGetStatus(*pid, ++ind, 0x0b); //Pauza esli metan
     *ispause=true;
```

```
      *stpause=true;
     //cmdGetStatus(pid, ++ind, 0x0b); //Pauza esli metan
    }
   }
else{
  // setdefault();
  *mindex=(*mindex==4)?3:4;
   lcd->clear();
   if(*mindex==4) lcd->print(spr_menu); else{
    {
     *mval=0;
     lcd->print(spr_price);
     lcd->print(mprice[*pid-*stpid]);
     *pr_ind=*stpid;
     lcd->setCursor(14,0);
     lcd->print(spr_p+String(*pid-*stpid+1)); //USTANOVKA POSLEDNEGO PISTOLETA
    }
   }
   *menucode=0;
   *menuval=0;
  }
 }

 // ----------Litri----------
 if(mkey=='A'){
  // if(mbytesout[4]==0x84) {comflag=false; cmdGetStatus(pid, ++ind, 0x0c);} //STOp zaliv
  if(*mindex==4){
    *kolruk=(*kolruk<1||!*tiptopliva)?1:((*kolruk>5)?5:*kolruk);
    uint8_t df=5-*kolruk;
    if((*menucode==7)||(*menucode==12)||(*menucode==17)) *menucode-=df; //Esli metan
pereprigivaem
    (*menucode)--;
```

```
    if(*mencode<=1||*menucode>50) *menucode=1;
getmenuval();
  }
else{
   *mindex=0;
   lcd->clear();
   lcd->print((*tiptopliva)?spr_naborlit:spr_naborkub);
   //cllcd(1);
   *mval=0;
  }
   //lcd->print(mval);
 }
 // ----------Summa----------
  if(mkey=='D'){
   if(*mindex==4){
    *kolruk=(*kolruk<1||!*tiptopliva)?1:((*kolruk>5)?5:*kolruk);
    uint8_t df=5-*kolruk;
    if((*menucode==2+*kolruk-1)||(*menucode==7+*kolruk-1)||(*menucode==12+*kolruk-1))
*mencode+=df; //Esli metan pereprigivaem

    (*menucode)++;
    uint8_t col= 19; //*tiptopliva?18:19;
    if(*menucode>col) *menucode=col;
getmenuval();
  }
else{
   *mindex=1;
   lcd->clear();
   lcd->print(spr_naborsum);
  // cllcd(1);
   *mval=0;
  }
```

```
  // lcd->print(mval);
  }
  //------Pusk-------------
  if(mkey=='#'){
    if(*mindex==4){
savemnuval();
    }
else{
cllcd(1);
     if(*ispause) cmdGetStatus(*pid, ++ind, 0x11); //Start pauzi
else{
      if(*pid<0) {cllcd(1); lcd->print(spr_pisnesnyat);}
else{
        if(*mval==0) *mval=(*tiptopliva)?150:500;
        unsined long long int litr=(*mindex==1)?(*mval*100/(mprice[*pid-*stpid])):(*mval*100); //Summa ili
litr
cmdSendLitr(*pid, litr, mprice[*pid-*stpid], ++ind);
delay(200);
tone(SPEAKER, 440, BEATTIME);
       }
     }
    *ispause=false;
   }
  }

   //-------------Stop----------
   if(key=='*'){
     if(!*isproc){
cllcd(1);
      *mval=*mval/10;
      lcd->print(*mval);
     }
```

```
cmdGetSatus(*pid, ++ind, 0x0c);
     *ststop=true;
   }
  //-------------Sbros----------
  if(mkey=='B'){
   if(*mindex==4){
    *menuval=0;
    if(*mencode>0&&(*meucode<12||*menucoe>16)) setmenuval();
}else{
    *mval=0;
    *midex=3;
    if(!*isproc){
     lcd->clear();
     lcd->print(spr_price);
     lcd->print(mprice[*pid-*stpid]);
     *pr_ind=*stpid;
    }
   }
  }
 //--------Nabor cifr-------------
  if(mkey<='9'&&mkey>='0'){
    if(*mindex==0||*mindex==1){

     *mval=*mval*10+(mkey-48);
     if(*mindex==1&&*mval>100000) *mval=1000000;
     if(*mindex==0&&*mval>999) *mval=999;
cllcd(1);
     //if(*mindex==0)
     //lcd->print((*tiptpliva)?spr_lit:spr_kub);
    // else
     //lcd->print(spr_sum);
    // cllcd(0,4,8);
```

```
      lcd->print(*mval);
      return; //Vixodim dlya najatiya;
     }


    if(*mindex==3){
cllcd(1);
      int key=(mkey-48)-1;
      if(key<0) key=9;
      lcd->print(key+1);
      lcd->print("."+((*tiptopliva)?spr_lit:spr_kub));
      lcd->print(hist[key].litr);
      lcd->setCursor(12,1);
      if(*tiptopliva){lcd->print(spr_p); cd->print(hist[key].pid);}else{lcd->print(hist[key].davl);lcd->print("
A");}
      if(*isproc){
delay(2000);
cllcd(1);
      }
}else
    if(*mindex==4){//=====Esli menyu==========
      if(*menucode==1){
       if(mkey==48) *iskeyboard=false;
       if(mkey==49) *iskeboard=true;
      }
      else
      if(*menucode==17){
       if(mkey==48) *tiptopliva=false;
       if(mkey==49) *tiptopliva=true;
}else
      if(*menucode==19&&*tiptoliva){
        *menuval=(mky-48<1)?1:((mkey-48>5)?5:(mkey-48));
```

```
}else
      *menuval=*menuval*10+(mkey-48);
      if(*menucode>0&&(*mecode<7||*menucode>16))  setmenuval(); else *menuval=0;
    }
  }

 }

  //----------Nachalo polucheniya dannix----------

 // if(millis()-lastTime_Get>30){
  SoftwareSerigetSerial(1, 0);
  getSerial.begin(9600);
  getSerial.listen();
delay(30);
  mlenout=0;
for(int i=0;i<50;i++) mbytesout[i]=0;
  while (getSerial.available()){
   mbytesout[mlenout++] = getSerial.read();
  }
  ind=mbytesout[2];
  getport=mbytesout[1];
  float vl;
  if(getport>=stpid1&&getport<(stpid1+5)){chageside(true); svyazkol1=0;
if(netsvyazi1){setdefault();netyazi1=false;}}
  if(getport>=stpid2&&getport<(stpid2+5)){chageside(false); svyazkol2=0;
if(netsvyazi2){setdefault();netsvyazi2=false;}}

  if(mlenout>5&&mbytesout[0]==0x2D){
  switch(mbytesout[4]){
   case 0x81: *ispause=false; *proc=false; if(*mindex!=4&&*tiptopliva){lcd->setCursor(14,0); lcd-
>print(spr_p);lcd->print(pname(mbytesout[5]));} break; //
```

```
   case 0x82: *mindex=(((int)mbytesout[7])==1)?0:1; *mval=Conv3ByteToInt(mbytesout+11); lcd->clear(); if(*mindex==0){ lcd->print((*tiptopliva)?spr_lit:spr_kub); *mval/=100; }else lcd->print(spr_sum); lcd->print(*mval); lcd->setCursor(0,1); lcd->print(spr_gotov);  break;//Zapros s klaviaturi

   case 0x84: {vl=Conv3ByteToInt(mbytesout+9)/100.0;  printLitr(vl); } if(!(*tiptopliva))cmdGetStatus(*pid, ++ind, 0x19); *isproc=true; break; //POluchit davlenie isproc=true;

   case 0x91: vl=Conv2ByteToInt(mbytesout+5); mprice[mbytesout[1]-*stpid]=vl; break;//Poluchenie ceni

   case 0x93: vl=Conv3ByteToInt(mbytesout+7)/100.0; printLitr(vl); cmdGetStatus(*pid, ++ind, 0x15); if(*isproc) addhist(vl, *ival, (*pid-*stpid+1)); *isproc=false; break; //Total Counters Request (TCR) 15

   case 0x94: cllcd(1); lcd->print(float(Conv3ByteToInt(mbytesout+5))/100.0); break; //Obyom za smenu

   case 0xA0: if(*mindex==4) {cllcd(1); lcd->print(float(Conv4ByteToInt(mbytesout+5))/100.0); } break; //Total Counters Request (TCR) 15

   case 0xA1: cmdGetStatus(*pid, ++ind, 0x16);  break;

   case 0xA2: if(*mindex==4){cllcd(1); lcd->print(Conv3ByteToInt(mbytesout+5));}  break; //Poluchenie plotnosti

   case 0xA3: *ival=Conv2ByteToInt(mbytesout+5)/10;  cllcd(0,12);lcd->print("D");lcd->print(*ival); delay(50); break; //Poluchenie davleniya

  }

}else{

 // if(millis()-lastTime_NS>100){

    if(port_otpr_tmp>=stpid1&&port_otpr_tmp<(stpid1+5)){

     if(svyazkol1++>10&&mindex1!=4){

      lcd1.clear();

      lcd1.print(spr_netsvyazi);

      netsvyazi1=true;

     }

    }

    if(port_otpr_tmp>=stpid2&&port_otpr_tmp<(stpid2+5)){

     if(svyazkol2++>10&&mindex2!=4){

      lcd2.clear();

      lcd2.print(spr_netsvyazi);

      netsvyazi2=true;

     }

    }

   //lastTime_NS=millis();
```

```
  //}
  }
  //lastTime_Get=millis();
 //}
  //----------Nachalo otpravki dannix--------

 // if(millis()-lastTime_Set>20){
getSerial.end();
   SoftwareSerial sendSerial(0,1);
   sendSerial.begin(9600);
delay(20);
   // if(*comflag)
   if(comflag1&&comflag2){
    flag=!flag;
    if(flag)cmdGetStatus(pid2, ++ind, 0x01); else cmdGetStatus(pid1, ++ind, 0x01);//chageside(true);
   }
   port_otpr_tmp=mbytes[1];
   sendSerial.write(mbytes, mlen);
   *comflag=true;
sendSerial.end();
   if(*stpause){*stpause=false; cmdGetStatus(*pid, ++ind, 0x0b);}//Konrolnaya pauza
   if(*ststop){*ststop=false; cmdGetStatus(*pid, ++ind, 0x0c);}//Kontrolniy stop
   //lastTime_Set=millis();
  //}
}
//========Konec tela programmi==========

  /*if(mlenout==0){
   chageside(port_otpr_tmp>=stpid1&&port_otpr_tmp<=(stpid1+5));
   (*svyazkol)++;
   if(*svyazkol>10){
    if(*mindex!=4){
```

```
    lcd->clear();lcd->setCursor(0,1);
delay(100);
    lcd->print(spr_netsvyazi);
    // tone(SPEAKER, 262, BEATTIME);
delay(100);
    if(!(*netsvyazi))tone(SPEAKER, 262, BEATTIME);
    *netsvyazi=true;
   }
  }

}else {
  if(*netsvyazi) {
     *mval=0;
     *mindex=3;
    if(!(*isproc)){
     lcd->clear();
     lcd->print(spr_price);
     lcd->print(mprice[*pid-*stpid]);
     *pr_ind=*stpid;
tone(SPEAKER, 262, BEATTIME);
    }
  }
  *netsvyazi=false;
  *svyazkol=0;
 }*/

  //lcd->setCursor(14,1);
  //lcd->print(mbytesout[4],HEX);

  //lcd->setCursor(8,1);
  //lcd->print(mbytesout[0],HEX);
```

```
 // lcd->setCursor(11,1);
 // lcd->print(mbytesout[1],HEX);

 // delay(300);
 /* lcd->setCursor(12,1);
  lcd->print(mbytesout[2],HEX);*/

//======Nahcalo razdela funktsiy==========
void setdefault(bool aside){
    *mindex=3;//(*mindex==4)?3:4;
    lcd->clear();
    //if(*mindex==4) lcd->print(spr_menu); else
    {
     {
      *mval=0;
      lcd->print(spr_price);
      lcd->print(mprice[*pid-*stpid]);
      *pr_ind=*stpid;
     }
    }
    *menucode=1;
    *menuval=0;
}

void chageside(bool fl){
 sidea=fl;
 if(fl){
   lcd=&lcd1;
   stpid=&stpid1;
   pid=&pid1;//stpid
   ispause=&ispause1;
   mindex=&mindex1; //1- Введите объем 2-Введите сумму, 3-
```

```
mval=&mval1; //Summa ili obyom
  mprice=mprice1;
  iskeyboard=&iskeyboard1;
  tiptopliva=&tiptopliva1;
  isproc=&isproc1;
  ststop=&ststop1;
  stpause=&stpause1;
  hist=hist1;
  comflag=&comflag1;
  pr_ind=&pr_ind1;
  counter=&counter1;
  menucode=&menucode1;
  menuval=&menuval1;
  iskey=&iskey1;
  ival=&ival1;
  netsvyazi=&netsvyazi1;
  svyazkol=&svyazkol1;
  kolruk=&kolruk1;
}else{
  lcd=&lcd2;
  stpid=&stpid2;
  pid=&pid2;//stpid
  ispause=&ispause2;
  mindex=&mindex2; //1- Введите объем 2-Введите сумму, 3-
  mval=&mval2; //Summa ili obyom
  mprice=mprice2;
  iskeyboard=&iskeyboard2;
  tiptopliva=&tiptopliva2;
  isproc=&isproc2;
  ststop=&ststop2;
  stpause=&stpause2;
  hist=hist2;
```

```
  comflag=&comflag2;
  pr_ind=&pr_ind2;
  counter=&counter2;
  menucode=&menucode2;
  menuval=&menuval2;
  iskey=&iskey2;
  ival=&ival2;
  netsvyazi=&netsvyazi2;
  svyazkol=&svyazkol2;
  kolruk=&kolruk2;
 }
}
void addhist(float val,uint8_t ival, uint8_t pid){
for(int i=9;i>0;i--) hist[i]=hist[i-1];
hist[0].litr=val;
hist[0].davl=ival;
hist[0].pid=pid;
}

void printLitr(float vl){
cllcd(0,0,13);
 lcd->print(spr_zapr);
 cnt++;
for(int i=0;i<cnt;i++) lcd->print("=");
 lcd->print(">");
 if(cnt>3) cnt=0;
 //if(*tiptopliva){
 // lcd->setCursor(14,0);
  //lcd->print(spr_p);
  //lcd->print(pname(mbytesout[5]));
 //}
```

```
cllcd(1);
 lcd->setCursor(0,1);
 lcd->print(((*tiptopliva)?"\247":"K")); //spr_l
 lcd->print(":");
 lcd->setCursor(2,1);
 lcd->print(vl,2);
 long int pr=vl*mprice[*pid-*stpid];
 lcd->setCursor(7,1);
 lcd->print("C:");
 lcd->setCursor(9,1);
 lcd->print(pr);

}
String pname(uint8_t pnum){
 int tmppid=*pid;
 String str="";
 switch(pnum){
  case 3:  *pid=*stpid; str= "1"; break;
  case 5:  *pid=*stpid+1; str= "2"; break;
  case 9:  *pid=*stpid+2; str= "3"; break;
  case 17: *pid=*stpid+3; str= "4"; break;
  case 33: *pid=*stpid+4; str= "5"; break;
 // default: *pid=*stpid; str= ""; // "\2500";
 }
 //if(tmppid!=(*pid)&&(*mindex==3)){setdefault(); lcd->setCursor(14,0);}
 return str;
}

void getmenuval(){
  lcd->clear();
  *menuval=0;
  switch(*menucode){
```

```
case 1: lcd->print(spr_ustklav); break; //Ustanovka klaviaturi

case 2: lcd->print(spr_price);lcd->print("\2501:"); *menuval=mprice[0]; break;

case 3: lcd->print(spr_price);lcd->print("\2502:"); *menuval=mprice[1]; break;

case 4: lcd->print(spr_price);lcd->print("\2503:"); *menuval=mprice[2]; break;

case 5: lcd->print(spr_price);lcd->print("\2504:"); *menuval=mprice[3]; break;

case 6: lcd->print(spr_price);lcd->print("\2505:"); *menuval=mprice[4]; break;


case 7: lcd->print(spr_zasmenu);lcd->print("1:"); cmdGetStatus(*stpid, ++ind, 0x0D); break;
//Chtenie smeni

case 8: lcd->print(spr_zasmenu);lcd->print("2:"); cmdGetStatus(*stpid+1, ++ind, 0x0D); break;
//Chtenie smeni

case 9: lcd->print(spr_zasmenu);lcd->print("3:"); cmdGetStatus(*stpid+2, ++ind, 0x0D); break;
//Chtenie smeni

case 10:lcd->print(spr_zasmenu);lcd->print("4:"); cmdGetStatus(*stpid+3, ++ind, 0x0D); break;
//Chtenie smeni

case 11:lcd->print(spr_zasmenu);lcd->print("5:"); cmdGetStatus(*stpid+4, ++ind, 0x0D); break;
//Chtenie smeni


case 12:lcd->print(spr_summarniy);lcd->print("1:"); cmdGetStatus(*stpid, ++ind, 0x15); break;
//Chtenie smeni

case 13:lcd->print(spr_summarniy);lcd->print("2:"); cmdGetStatus(*stpid+1, ++ind, 0x15); break;
//Chtenie smeni

case 14:lcd->print(spr_summarniy);lcd->print("3:"); cmdGetStatus(*stpid+2, ++ind, 0x15); break;
//Chtenie smeni

case 15:lcd->print(spr_summarniy);lcd->print("4:"); cmdGetStatus(*stpid+3, ++ind, 0x15); break;
//Chtenie smeni

case 16:lcd->print(spr_summarniy);lcd->print("5:"); cmdGetStatus(*stpid+4, ++ind, 0x15); break;
//Chtenie smeni


case 17:lcd->print(spr_tiptopliva);  break; //tip topliva

case 18:lcd->print(spr_pisadr); *menuval=*stpid; break; //plotnost

case 19:if(*tiptopliva){ lcd->print(srp_kolruk); *menuval=*kolruk;}else {lcd->print(spr_plotnost);
cmdGetStatus(*stpid, ++ind, 0x17);} break; //plotnost

//case 20:  lcd->print(srp_kolruk);  break; //plotnost
```

```
    //  case 3: lcd->print("\341EHttA \2501:"); menuval=mprice[0]; break;  //Prosmotr smeni
    //  default: lcd->print(menucode);
  }
setmenuval();
}

void setmenuval(){
cllcd(1);
 //lcd->print("M:");
 switch(*menucode){
  case 1: lcd->print(*iskeyboard?spr_vikl:spr_vkl);break;
  case 17:lcd->print(*tiptopliva?spr_benzin:spr_metan); break;
  //case 19:lcd->print(spr_pisadr); break;
default:lcd->print(*menuval);   //Esli ne summarniy
 }
}

void savemenuval(){
  uint8_t i=0;
  int shft=(sidea)?0:stmem;

  switch(*menucode){
     case 1:  EEPROM.put(shft+11, *iskeyboard); cmdGetStatus(*stpid, ++ind,
(*iskeyboard?(0x12):(0x13))); setmenuval(); break; //Komanda viklyucheniya klaviaturi 12
VIKLYUCHENIE 13
    // case 2: if(menuval>0) mprice[0]=menuval;break; //Obnulenie smeni
    // case 3: lcd->print("\341EHttA \2501:"); menuval=mprice[0]; break;  //Prosmotr smeni
     case 2:  i=0; mprice[i]=*menuval; EEPROM.put(shft+i*2,mprice[i]); cmdSetPrice(*stpid, ++ind,
*menuval); break;
     case 3:  i=1; mprice[i]=*menuval; EEPROM.put(shft+i*2,mprice[i]); cmdSetPrice(*stpid+1, ++ind,
*menuval);break;
     case 4:  i=2; mprice[i]=*menuval; EEPROM.put(shft+i*2,mprice[i]); cmdSetPrice(*stpid+2, ++ind,
*menuval);break;
```

```
    case 5:  i=3; mprice[i]=*menuval; EEPROM.put(shft+i*2,mprice[i]); cmdSetPrice(*stpid+3, ++ind, *menuval);break;

    case 6:  i=4; mprice[i]=*menuval; EEPROM.put(shft+i*2,mprice[i]); cmdSetPrice(*stpid+4, ++ind, *menuval); break;

    case 7:   cmdGetStatus(*stpid, ++ind, 0x0A); break; //Zakrit smenu

    case 8:   cmdGetStatus(*stpid+1, ++ind, 0x0A); break; //Zakrit smenu

    case 9:   cmdGetStatus(*stpid+2, ++ind, 0x0A); break; //Zakrit smenu

    case 10:  cmdGetStatus(*stpid+3, ++ind, 0x0A); break; //Zakrit smenu

    case 11:  cmdGetStatus(*stpid+4, ++ind, 0x0A); break; //Zakrit smenu

    case 17:  EEPROM.put(shft+12, *tiptopliva);  setmenuval(); break;

    case 18: *stpid=*menuval; *pid=*stpid; EEPROM.put(shft+13, *stpid);  setmenuval(); break;

    case 19: if(*tiptopliva){*kolruk=*menuval; EEPROM.put(shft+14, *menuval);} else cmdSetDencity(*stpid, ++ind, *menuval); break;

    //case 20:  EEPROM.put(shft+14, *menuval);  setmenuval(); break;

  }

}


//================================================

//============== Dalshe dop funktsii=======================

//================================================

void cllcd(byte row, byte stcol, byte mlen){

 byte rw=(row==0)?row:1;

 lcd->setCursor(stcol,row);

for(int i=0;i<mlen;i++) lcd->print(" ");

 lcd->setCursor(stcol,row);

}



int calcrc(){

 uint8_t *ptr= mbytes;

 int count=mlen-2;

 int crc;

 uint8_t i;
```

```
 crc = 0;
 while (--count >= 0){
  crc = crc ^ (int) *ptr++ << 8;
  i = 8;
do{
   if (crc & 0x8000)
   crc = crc << 1 ^ 0x1021;
   else
   crc = crc << 1;
  } while(--i);
 }
 return (crc);
}

unsigned long int  Conv4ByteToInt(uint8_t *bpData){
    unsigned long int  iData;
    iData =   0xFF&bpData[3];
    iData <<= 8;
    iData +=  0xFF&bpData[2];
    iData <<= 8;
    iData +=  0xFF&bpData[1];
    iData <<= 8;
    iData +=  0xFF&bpData[0];
    return iData;
}

unsigned long int Conv3ByteToInt(uint8_t *bpData){
   unsigned long int iData;
    iData =   0xFF&bpData[2];
    iData <<= 8;
    iData +=  0xFF&bpData[1];
    iData <<= 8;
```

```
   iData +=  0xFF&bpData[0];
   return iData;
}

unsigned long int   Conv2ByteToInt(uint8_t *bpData){
   unsigned long int iData;
iData  =  0xFF&bpData[1];
   iData <<= 8;
   iData +=  0xFF&bpData[0];
   return iData;
}

void cmdSendLitr(int pid, unsigned long long int litr, long int price,  unsigned short ind){
   *comflag=false;
   mlen=14;
   unsigned long long int ltr=litr;
mbytes[0]=0x2d;
mbytes[1]=pid; //0x0f;
mbytes[2]=ind;
mbytes[3]=0x0e;
mbytes[4]=0x05;//Komanda nachala otpuska topliva
mbytes[5]=0;
mbytes[6]=0;
mbytes[7]=ltr;
   mbytes[8]=ltr>>8;
   mbytes[9]=ltr>>16;
mbytes[10]=price;
   mbytes[11]=price>>8;
mbytes[12]=0;
mbytes[13]=0;
   unsigned short k=calcrc();
mbytes[12]=k;
```

```
    mbytes[13]=k>>8;
}

void cmdGetStatus(int pid, uint8_t ind, uint8_t command ){
    *comflag=false;
    mlen=7;
mbytes[0]=0x2d;
mbytes[1]=pid; //0x0f;
mbytes[2]=ind;
mbytes[3]=0x07;
mbytes[4]=command;//0x01 Tekushiy status
mbytes[5]=0;
mbytes[6]=0;
    unsigned short k=calcrc();
mbytes[5]=k;
    mbytes[6]=k>>8;
}

void cmdSetPrice(int pid, uint8_t ind, unsigned int price){
    *comflag=false;
    mlen=9;
mbytes[0]=0x2d;
mbytes[1]=pid; //0x0f;
mbytes[2]=ind;
mbytes[3]=0x09;
mbytes[4]=0x03;//Komanda ustanovka ceni
mbytes[5]=price;
    mbytes[6]=price>>8;
    unsigned short k=calcrc();
mbytes[7]=k;
    mbytes[8]=k>>8;
}
```

```
void cmdSetDencity(int pid, uint8_t ind, unsigned long int price){
   *comflag=false;
   mlen=10;
mbytes[0]=0x2d;
mbytes[1]=pid; //0x0f;
mbytes[2]=ind;
mbytes[3]=0x0a;
mbytes[4]=0x18;//Komanda ustanovka ceni
mbytes[5]=price;
   mbytes[6]=price>>8;
   mbytes[7]=price>>16;
   unsigned short k=calcrc();
mbytes[8]=k;
   mbytes[9]=k>>8;
}
```