

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМ. Р.Е. АЛЕКСЕЕВА»
(НГТУ)

Институт Радиоэлектроники и информационных технологий

Направление подготовки (специальность) 09.03.01 Информатика и вычислительная техника
(код и наименование)

Направленность (профиль) образовательной программы Вычислительные машины, комплексы, системы и сети
(наименование)

Кафедра Вычислительных систем и технологий


ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
бакалавра

(бакалавра, магистра, специалиста)

Студента Болотова Михаила Александровича группы 16-B-1
(Ф.И.О.)

на тему Аппаратно-программная система для помощи людям с ограниченными возможностями для ориентации в пространстве
(наименование темы работы)

СТУДЕНТ:

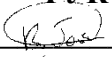

(подпись) Болотов М. А.
(фамилия, и., о.)
03.07.2020
(дата)

КОНСУЛЬТАНТЫ:

1. По _____
(подпись) _____ (фамилия, и., о.)

(дата)

РУКОВОДИТЕЛЬ:


(подпись) Гай В. Е.
(фамилия, и., о.)
03.07.2020
(дата)

2. По _____
(подпись) _____ (фамилия, и., о.)

(дата)

РЕЦЕНЗЕНТ:

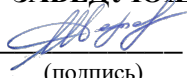
(подпись) _____ (фамилия, и., о.)

(дата)

3. По _____
(подпись) _____ (фамилия, и., о.)

(дата)

ЗАВЕДУЮЩИЙ КАФЕДРОЙ


(подпись) Жевнерчук Д. В.
(фамилия, и.о.)
03.07.2020
(дата)


ВКР защищена _____
(дата)

протокол № _____
с оценкой _____

**МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМ. Р.Е. АЛЕКСЕЕВА»
(НГТУ)**

Кафедра Вычислительных систем и технологий

УТВЕРЖДАЮ

 Зав. кафедрой ВСТ
Жевнерчук Д. В.

«12» мая 2020 г.

**ЗАДАНИЕ
на выполнение выпускной квалификационной работы**

по направлению подготовки (специальности) 09.03.01 Информатика и вычислительная техника

студенту Болотову Михаилу Александровичу группы 16-В-1
(Ф.И.О.)

1. Тема ВКР Аппаратно-программная система для помощи людям с ограниченными возможностями для ориентации в пространстве (утверждена приказом по вузу от 15.04.2020 №867/5)
2. Срок сдачи студентом законченной работы 02.06.2020
3. Исходные данные к работе однолучевой радар, платформа Arduino Mega 2560, языки разработки – Python, Arduino.
4. Содержание расчетно-пояснительной записки (перечень вопросов, подлежащих разработке)

Введение

1 Техническое задание

2 Анализ технического задания

3 Разработка системы на структурном уровне

4 Разработка аппаратной части системы

5 Разработка программной архитектуры системы

6 Тестирование системы

Заключение

Список литературы

Приложение А. Программа обработки данных

Приложение Б. Программа для Arduino Mega 2560

5. Перечень графического материала (с точным указанием обязательных чертежей)

1 Структурная схема аппаратно-программной системы

2 Принципиальная схема работы аппаратно-программной системы

3 Фрагменты чертежей механизма воздействия

4 Блок-схемы программных алгоритмов

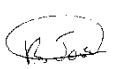
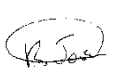
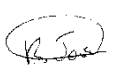
5 Результаты работы алгоритма обработки данных

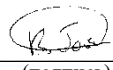
6 Изображения тестирования системы

6. Консультанты по ВКР (с указанием относящихся к ним разделов)

Нормоконтроль Гай В.Е. _____

7. Дата выдачи задания 12.05.2020

Код и содержание Компетенции	Задание	Проектируемый результат	Отметка о выполнении
ПК-1, Способность разрабатывать модели компонентов информационных систем, включая модели баз данных и модели интерфейсов «человек – электронно-вычислительная машина»	Разработать структурную схему системы	Структурная схема системы	
ПК-2, Способность разрабатывать компоненты аппаратно-программных комплексов и баз данных, используя современные инструментальные средства и технологии программирования	Разработать прототип системы для помощи людям с ограниченными возможностями в ориентации в пространстве; разработать алгоритмы работы системы	Прототип аппаратно-программной системы; блок-схемы алгоритмов работы системы; реализация алгоритмов на языках Python и Arduino	
ПК-3, Способность обосновывать принимаемые проектные решения, осуществлять постановку и выполнять эксперименты по проверке их корректности и эффективности	Выполнить анализ существующих систем и выбрать метод передачи информации; оценить эффективность выбранного метода	Анализ существующих систем проведен; выбор метода передачи информации протестирован и оценен	

Руководитель  В.Е. Гай
(подпись)

Задание принял к исполнению 12.05.2020
(дата)

Студент  М.А. Болотов
(подпись)

**МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМ. Р.Е. АЛЕКСЕЕВА»
(НГТУ)**

АННОТАЦИЯ

к выпускной квалификационной работе

**по направлению подготовки (специальности) 09.03.01 Информатика и
вычислительная техника**
(код и наименование)

студента Болотова Михаила Александровича группы 16-В-1
(Ф.И.О.)

по теме Аппаратно-программная система для помощи людям с ограниченными
возможностями для ориентации в пространстве.

Выпускная квалификационная работа выполнена на 47 страницах, содержит 20 рисунков, 0 таблиц, библиографический список из 14 источников, 2 приложения.

Актуальность: исследование методов передачи информации об окружающем пространстве
людям с ограниченными зрительными возможностями является актуальной задачей в
постоянно усложняющемся мире. Разработка прототипа аппаратно-программной системы для
помощи людям с ограниченными возможностями для ориентации в пространстве с
использованием современных технологий может способствовать облегчению передвижения
не зрячего человека.

Объект исследования: методы передачи информации об окружающем пространстве людям с
ограниченными зрительными возможностями.

Предмет исследования: аппаратно-программная система трансформирующая сигнал с радара
в механические движения иголок, тактильно воздействующих на руку человека.

Цель исследования: разработка прототипа аппаратно-программной системы для помощи
людям с ограниченными возможностями для ориентации в пространстве.

Задачи исследования: исследовать существующие методы передачи информации об
окружающем пространстве людям с ограниченными зрительными возможностями;
исследовать целесообразность использования метода тактильного воздействия в связке с
радаром в аппаратно-программной системе, предназначенной для помощи людям с
ограниченными зрительными возможностями для ориентации в пространстве; разработать

прототип аппаратно-программной системы для помощи не зрячим людям в ориентации в пространстве; выполнить тестирование разработанного прототипа.

Методы исследования: анализ существующих патентов устройств и методов для ориентации людей с ограниченными возможностями в пространстве; моделирование прототипа аппаратно-программной системы; наблюдение тактильного воздействия иголок интерактивной части системы на кожу человека; проведение экспериментов с прототипом аппаратно-программной системы.

Структура работы: выпускная квалификационная работа состоит из введения, шести глав, заключения, списка литературы и двух приложений.

Во введении дается описание проблемы, лежащей в основе выпускной квалификационной работы.

В 1 разделе «Техническое задание» составлено техническое задание на разработку.

Во 2 разделе «Анализ технического задания» произведен выбор метода передачи зрительной информации в тактильную, выбор датчика и аппаратной платформы для прототипа аппаратно-программной системы для помощи людям с ограниченными возможностями для ориентации в пространстве.

В 3 разделе «Разработка системы на структурном уровне» приводится общая структурная схема системы, описывается принцип работы прототипа аппаратно-программной системы, поясняется устройство механизма воздействия прототипа.

В 4 разделе «Разработка аппаратной части системы» описывается процесс изготовления катушек индуктивности, проектирования и изготовления корпуса для механизма воздействия.

В 5 разделе «Разработка программной архитектуры» выполняется обзор программной архитектуры и разрабатываются программные средства для прототипа аппаратно-программной системы.

В 6 разделе «Тестирование системы» описываются методы тестирования аппаратно-программной системы на разных этапах разработки и полученные результаты.

В заключении приводятся основные выводы по проделанной работе.

Выводы:

1. Разработан прототип аппаратно-программной системы для помощи людям с ограниченными возможностями для ориентации в пространстве.
2. Тестирование прототипа показало успешность в выборе метода передачи информации об окружающем пространстве человеку с ограниченными зрительными возможностями.

Рекомендации:

1. Дальнейшее развитие проекта.
2. Использование передовых технологий в области машинного обучения.







/Болотов М.А.

подпись студента /расшифровка подписи

«3» июля 2020 г.

Оглавление	
Введение	5
1 Техническое задание	6
1.1 Назначение разработки и область применения	6
1.2 Технические требования	6
2 Анализ технического задания.....	7
2.1 Выбор датчика	7
2.2 Выбор аппаратной платформы	9
2.3 Выбор метода передачи данных	11
3 Разработка системы на структурном уровне	12
3.1 Общая структурная схема	12
3.2 Принцип работы аппаратно-программной системы.....	13
3.3 Устройство механизма воздействия прототипа АПС.....	14
4 Разработка аппаратной части системы	16
4.1 Изготовление катушек индуктивности для механизма воздействия.....	16
4.2 Проектирование и изготовление корпуса для механизма воздействия.....	18
5 Разработка программной архитектуры системы	22
5.1 Обзор программной архитектуры.....	22
5.2 Разработка скрипта взаимодействия радара и микроконтроллера	22
5.3 Разработка программы для АПС	27
6 Тестирование системы	29
Заключение	36
Список литературы	37
Приложение А. Программа обработки данных	39
Приложение Б. Программа для Arduino Mega 2560	43

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)			
Изм.	Лист	№ докум.	Подпись	Дата	Аппаратно-программная Система для помощи людям с ограниченными возможностями для ориентации в пространстве	Лит.	Лист	Листов
Разраб.		Болотов М.А.		03.07				
Провер.		Гай В.Е.		03.07			4	51
Реценз.						НГТУ кафедра ВСТ		
Н. Контр.		Гай В.Е.		03.07				
Утверд.		Жевнерчук Д.В.		03.07				

Введение

Во всем мире идут разработки по созданию устройств, помогающих людям с ограниченными возможностями лучше ориентироваться в пространстве. Так, еще 50 лет назад было трудно представить компактную перчатку с радаром, проецирующую рельеф пространства на ладонь человека. С неуклонно растущим прогрессом увеличивается и спектр возможностей по созданию приспособлений для ориентации в пространстве людей с ограниченными возможностями.

Люди с ограниченными возможностями могут получать больше информации через другие органы чувств. Человек с ограниченным зрением лучше слышит и у него более развиты тактильные ощущения. Идея приспособления, помогающего с ориентацией в пространстве - заменить недостающие органы зрения датчиками-радаром, а информацию, в перекодированном виде человек «считывает» кожей рук, самыми чувствительными нервными окончаниями, например, на кончиках пальцев.

Всем известны классические методы ориентации в пространстве – трость, назначение которой сигнализировать о возможных препятствиях, собака-поводырь, двигаясь впереди идущего человека «подсказывает» ему путь. Теперь, возможности слабовидящих людей можно расширить компактными электронными приборами. Такие устройства могут быть универсальным «поводырем», который заменит трость и собаку одновременно. Так незрячий человек обучившись навыкам работы с прибором, получит возможность, с помощью тактильных ощущений представлять себе окружающее пространство, его форму, расстояние до объектов, размеры самих объектов.

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		5

1 Техническое задание

1.1 Назначение разработки и область применения

Аппаратно-программная система, далее АПС, предназначена для дополнительного восприятия слабовидящими и незрячими людьми окружающего пространства. Информация о рельефе поступает на микроконтроллер, который запрограммирован обрабатывать данные и проецировать их в механические движения. АПС можно разделить на функциональные части: приемник информации об окружающем мире, обработчик сигнала в удобочитаемую форму, тактильный интерфейс взаимодействия с человеком. Для взаимодействия этих частей необходимо разработать программное обеспечение.

Область применения АПС - носимое устройство, с помощью которого посредством тактильных ощущений слепой человек может получать информацию о предметах, находящихся перед ним.

1.2 Технические требования

Требования к разработке:

1. Устройство должно достоверно передавать информацию о пространстве в виде механических движений, интуитивно понятных для человека.

2. Устройство должно определять расстояние до препятствий и распределять их по 4 группам: цели до 1 метра, от 1 до 2 метров, от 2 до 3 и от 3 метров.

3. АПС должна иметь расширяемую архитектуру: как с аппаратной стороны, так и с программной.

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		6

2 Анализ технического задания

2.1 Выбор датчика

Идея разработки аппаратно-программной системы основывается на некоторых интернет ресурсах. Проанализировав множество приспособлений, помогающих людям с ограниченными возможностями ориентироваться в пространстве, мною было выявлено оптимальное решение на данный момент. АПС имеет расширяемую архитектуру, а это значит, что отдельные модули, такие как датчик определения препятствий, интерактивная часть и микроконтроллер, могут заменяться улучшенными версиями. На момент разработки АПС выбор датчиков был весьма ограничен, но тем не менее, мною был исследован ряд интересных решений, приведенный ниже.

1. Электронный поводырь для слепых «Электросонар» [1].

Это устройство имеет небольшие размеры, носится на руке, воздействуя на нее вибрацией разной длительности – в зависимости от расстояния до препятствия. Кроме тактильного воздействия имеет дополнительное звуковое сопровождение. Решение задействовать несколько органов восприятия уменьшает шанс пропуска препятствия, что снижает риски. В качестве датчиков используются ультразвуковые и инфракрасные радары. Дальность определения препятствий составляет около 7 метров. В статье утверждается, что предусмотрено несколько режимов работы прибора – в помещении и на улице. Такой подход позволяет детализировать комнату, и, наоборот, увеличить дальность действия радара на открытом пространстве. Для понимания рельефа местности необходимо направлять устройство в разные стороны, тем самым чувствуется расстояние до объектов.

2. Прибор для реабилитации слепых и слабовидящих [2].

Еще один прибор был создан на основе использования эхолокации. Данное устройство по форме напоминает очки – на уровне глаз находятся ультразвуковые сенсоры, а информация, полученная с них, перекодирована в звуковой сигнал. Аналогично предыдущему примеру интенсивность звукового сигнала увеличивается по мере приближения к препятствию.

3. Ультразвуковой локатор для слепых [3].

Данное устройство имеет форму трости с ультразвуковым локатором в рукоятке. Основным преимуществом данной конструкции является возможность определять не только наземные объекты, но и препятствия, не касающиеся земли. Информирование о препятствии основано на звуковом сигнале – по мере приближения к объекту тональность звука уменьшается. По заявлениям автора данного устройства, человек с хорошим слухом может определить препятствие с точностью до 15 сантиметров. Такое изобретение обладает неоспоримым преимуществом – в случае выхода из строя электронного модуля обнаружения, остается вариант использования прибора как обычной трости.

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		7

Мною был рассмотрен ряд запатентованных устройств – от простых механических, до электронных.

4. Поводырь для слепых [4].

Представляет собой специально изготовленную тележку с колёсами, которую толкает перед собой слепой. По вибрации, полученной через тележку от дороги, человек имеет некоторое представление о препятствиях на своем пути. Поводырь для слепых – механическое устройство, простое в изготовлении, помогающее людям с ограниченными возможностями лучше ориентироваться в пространстве. В настоящее время появилось множество возможностей и ресурсов для разработок продвинутых устройств, рассмотренных ниже.

5. Устройство для обнаружения препятствий слепыми [5].

Фактически это сложный оптический прибор, предназначенный для точного определения расстояния до объектов, но без дополнительных устройств не может использоваться слепым.

6. Устройство акустического представления пространственной информации для инвалидов по зрению [6].

В этом патенте содержится описание устройства преобразования ультразвуковых сигналов в звуковые сигналы, которые выводятся на стереонаушники.

7. Способ и система точной локализации слабовидящего или слепого человека [7].

Способ основан на использовании данных GPS – приёмника и вывода преобразованных данных на звуковые предупреждения. Информация о препятствиях и путях обхода с учетом скорости движения пользователя, двумерная и трехмерная карты передаются в систему. Создается карта глубины окружающего пространства, производится фильтрация, выделяются объекты, которые накладываются на двумерную карту. В результате вычисляется расстояние до потенциальных препятствий, информирование о которых происходит посредством звукового сигнала.

8. Трость для инвалида по зрению [8].

Это изобретение имеет форму трости с улучшенной рукояткой. Рукоять трости содержит приемоизлучатель, блок обработки данных и звуковой сигнализатор. Угол наклона приемоизлучателя может изменяться пользователем. Это устройство представляет собой усовершенствованную трость, что является несомненным преимуществом перед другими аналогами без механической части обнаружения препятствий.

9. Ультразвуковой локатор для слепых [9].

Данное устройство основано на последовательно соединенных мультивибраторе, передатчике, приемнике и электроакустическом преобразователе. Информирование о препятствиях также, как и в предыдущих примерах представлено в виде звукового сигнала.

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		

Все перечисленные приборы имеют свои преимущества и недостатки. Основными недостатками имеющихся приборов является или их очень высокая цена или несовершенство конструкции, а также неполное, частичное предоставление визуальной информации человеку с ограниченными возможностями зрения.

После тщательного изучения запатентованных устройств и приборов для помощи слепым, было принято решение создать прототип работающего устройства АПС с механизмом воздействия, основанном на тактильных ощущениях, для помощи слепым людям. В качестве датчика приемника хотелось бы использовать FMCW (Frequency-Modulated Continuous-Wave) радар с незатухающей гармонической волной непрерывно передающий одиночную частоту и прослушивающий эхо. Данный радар не может определить расстояние до цели из-за непрерывного прослушивания эхо, однако применив частотную модуляцию появляется возможность определять расстояние как до неподвижных, так и до движущихся целей. FMCW радар позволил бы извлекать информацию о скорости движения цели, что дало бы возможность спроецировать препятствие и информировать о нем пользователя. Радары данного типа позволяют получать спектрограмму местности, которую, не трудно спроецировать на интерактивную часть взаимодействия с пользователем.

Не имея возможности получить радар FMCW, по причине не дешевизны, мне было предложено использовать однолучевой радар, который был в наличии и достался мне для проекта бесплатно. Недостатком однолучевого радара является то, что он определяет количество целей – препятствий с шагом в глубину, равным 0,7 метра и без возможности определения с какой стороны находится объект. Не смотря на ограниченные возможности радара, АПС была спроектирована с возможностью использования FMCW радара в качестве датчика – приемника. Таким образом, мною предусмотрена расширяемая архитектура АПС для помощи людям с ограниченными возможностями.

2.2 Выбор аппаратной платформы

Аппаратная часть системы проектировалась с учетом получения на вход спектрограммы рельефа, поэтому было запланировано использование 16 катушек индуктивности с неодимовыми магнитами внутри. Часть АПС, предназначенная для тактильного взаимодействия с пользователем далее будет называться интерактивной. Получая с радара спектрограмму поверхности, предусматривалось разделение ее на матрицу 4 на 4. Расстояние до препятствия в каждой ячейке матрицы сопоставляется с определенной катушкой в интерактивной части. Частота движений магнита в катушке индуктивности зависит от расстояния до препятствия в соответствующей

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

секции матрицы. Чем дальше препятствие, тем меньше частота, и наоборот, с приближением к цели, частота движения магнита в катушке возрастает.

Интерактивная часть была спроектирована основываясь на возможности использования в системе радара типа FMCW. Поэтому, для обслуживания всех 16 катушек индуктивности необходимо подобрать микроконтроллер со следующими характеристиками:

1. Таймеры-счетчики для создания сигнала заданной частоты для радара.
2. Аналоговые входы для получения сигнала с радара.
3. Цифровые выходы с возможностью широтно-импульсной модуляции.
4. Количество цифровых входов – 16 и более.

Рассмотрено три модели контроллеров семейства Arduino – Arduino Nano, Arduino Uno и Arduino Mega 2560. Все имеют тактовую частоту процессора 16 МГц. Рассмотрение Arduino в качестве контроллера для АПС обусловлено несколькими факторами:

1. Простота программирования.
2. Доступная документация.
3. Дешевизна.

Arduino Nano имеет всего 14 цифровых входов/выходов, что недостаточно для обеспечения соединения с интерактивной частью, содержащей 16 катушек. Из преимуществ – микроконтроллер имеет малые размеры: 1.85 см на 4.2 см.

Arduino Uno также имеет 14 цифровых входов/выходов. Памяти, как оперативной, так и постоянной больше в два раза, по сравнению с Arduino Nano. Имеется недостаток – размер платы составляет 5.3 см на 6.9 см.

Arduino Mega 2560 содержит 54 цифровых входа/выхода, 14 из них могут работать в режиме широтно-импульсной модуляции. Аналоговых входов в количестве 16 будет достаточно для получения сигнала с радара. Неоспоримым преимуществом является наличие 6 таймеров-счетчиков. Памяти, как оперативной, так и постоянной в 2 раза больше, чем у Arduino Uno и в 4 раза больше по сравнению с Arduino Nano. Единственным минусом данного микроконтроллера являются его большие размеры, по сравнению с конкурентами - 10.16 см на 5.3 см.

Основываясь на вышеописанных преимуществах и критериях выбора, для АПС был выбран микроконтроллер Arduino Mega 2560. Прототип аппаратной системы для помощи людям с ограниченными возможностями для ориентации в пространстве допускает не малые размеры конструкции, для удешевления и упрощения разработки.

2.3 Выбор метода передачи данных

Текущая конфигурация аппаратной системы – однолучевой радар, микроконтроллер ATmega2560 на платформе Arduino и интерактивная часть в виде корпуса с 16 катушками индуктивности. Необходимо предусмотреть надежное взаимодействие вышеперечисленных частей.

Взаимодействие радара с Arduino Mega 2560 потребует дополнительной конфигурации с использованием модулей-переходников RJ45. Исходя из того, что радар имеет не малые размеры, прототип АПС будет не переносным, поэтому нет необходимости в дополнительных модулях, усложняющих конструкцию. Радар соединяется сетевым кабелем по интерфейсу RJ45 с компьютером. Компьютер с Arduino Mega2560 соединяется по UART интерфейсу.

Вышеописанный вариант соединения радара с Arduino Mega2560 имеет преимущество – ресурсы компьютера можно использовать для обработки данных, полученных с радара, тем самым снижается нагрузка на процессор микроконтроллера.

Соединение Arduino Mega 2560 с интерактивной частью выполняется в виде проводов с мягкой оболочкой диаметром 0,1 см. От каждой катушки выводится пара проводов – земля и питание. Земля катушек замыкается и соединяется одним проводом с платой. Для удобства размещения интерактивной части поверх платформы Arduino, провода с катушек объединяются в разъем плоского кабеля для подключения дисковод FDD от материнской платы. Разъем кабеля подсоединяется к задней части Arduino Mega 2560 в области нахождения цифровых входов/выходов с 20 по 55.

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

3 Разработка системы на структурном уровне

3.1 Общая структурная схема

Прототип аппаратно-программной системы для помощи людям с ограниченными зрительными возможностями условно можно разделить на три части – радар, контроллер Arduino Mega 2560 и интерактивная часть взаимодействия с пользователем. В связи с особенностями радара – появляется дополнительная промежуточная часть в виде скрипта на компьютере, функции которого – принимать данные, выделять значимую информацию, обрабатывать ее и передавать в удобочитаемую форму на микроконтроллер. На рисунке 1 представлена общая структурная схема прототипа АПС.

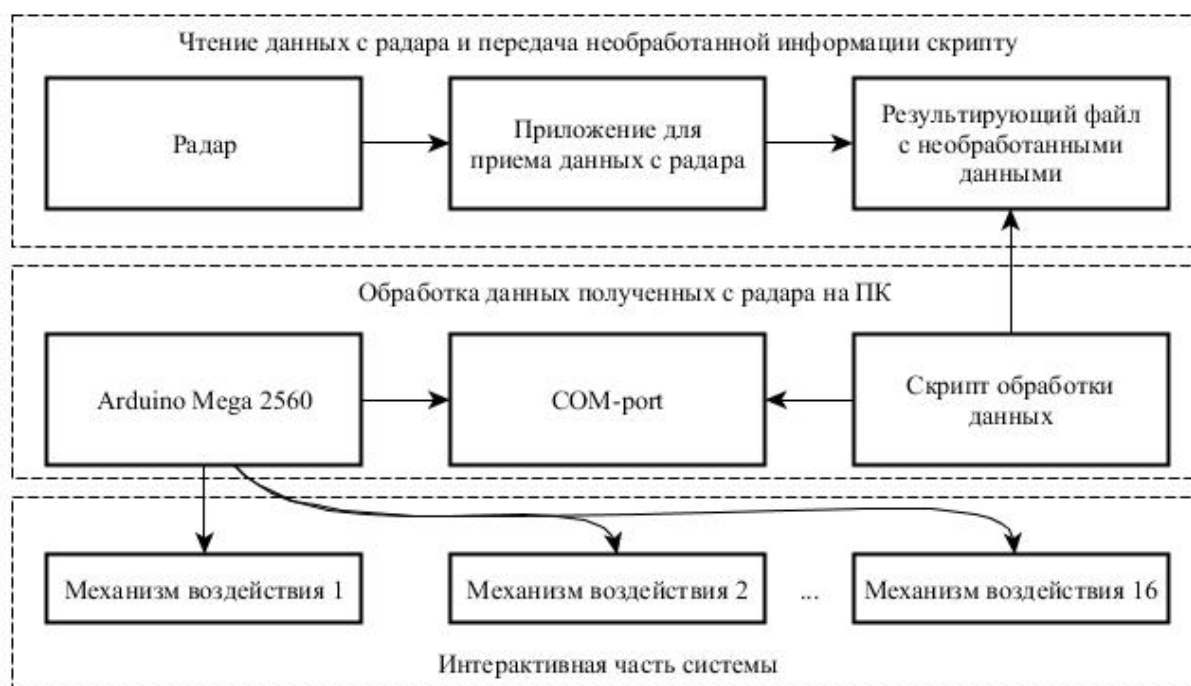


Рис. 1. Общая структурная схема аппаратно-программной системы

На схеме представлены точки взаимодействия модулей:

1. AutoRadarResultLog.txt – результирующий файл, создаваемый программой взаимодействия с радаром AutoRadarControl.exe. Содержит необработанные данные в формате «[номер цели] [номер зоны] [амплитуда]». Значительная часть файла – не информативные данные, заполненные нулями.

2. COM-port – один из COM портов, открытый микроконтроллером, предназначен для передачи обработанных скриптом данных в микроконтроллер по последовательному порту. Скрипт отсеивает не информативные данные, переводит амплитуду в метры и распределяет обнаруженные цели по группам. Данные передаются в формате «<start>aNbNcNdN<stop>», где a, b, c, d – номера групп, объединяющих по 4

катушки индуктивности, а N – количество объектов, принадлежащих к группе. Arduino Mega 2560 активирует ряд катушек индуктивности, соответствующий определенной группе с частотой равной количеству целей в ней.

3.2 Принцип работы аппаратно-программной системы

Принцип работы АПС заключается в преобразовании высокочастотного цифрового сигнала от радара в импульсы возвратно-поступательных движений иглоочек, «показывающих» незрячему человеку форму пространства перед радаром. По частоте и амплитуде движений шестнадцати иглоочек человек получает информацию и ориентируется в пространстве. Идеально было бы разместить радар, сам прибор и механизм воздействия компактно в виде надеваемой на руку перчатки, где роль механизма воздействия будет играть внутренний слой перчатки, обращенный к коже пальцев руки, а направление радара разместить по направлению пальцев человека, как если бы он показывал на незнакомый предмет рукой. Для решения поставленной задачи было принято решение первый образец выполнить в виде прототипа с целью экономии и отработки механизмов преобразования сигнала, программирования кода, а также основных параметров механизма воздействия.

Принципиальная схема прототипа АПС представлена на рисунке 2. Радар посылает сигнал, который отражается от предмета и возвращается в радар. Далее сигнал преобразуется в микроконтроллере и, через механизм воздействия в виде иглоочек, передает форму, размеры и расстояние до него незрячему человеку. По частоте и амплитуде воздействия на кожу человек определяет размеры предмета и расстояние до него.

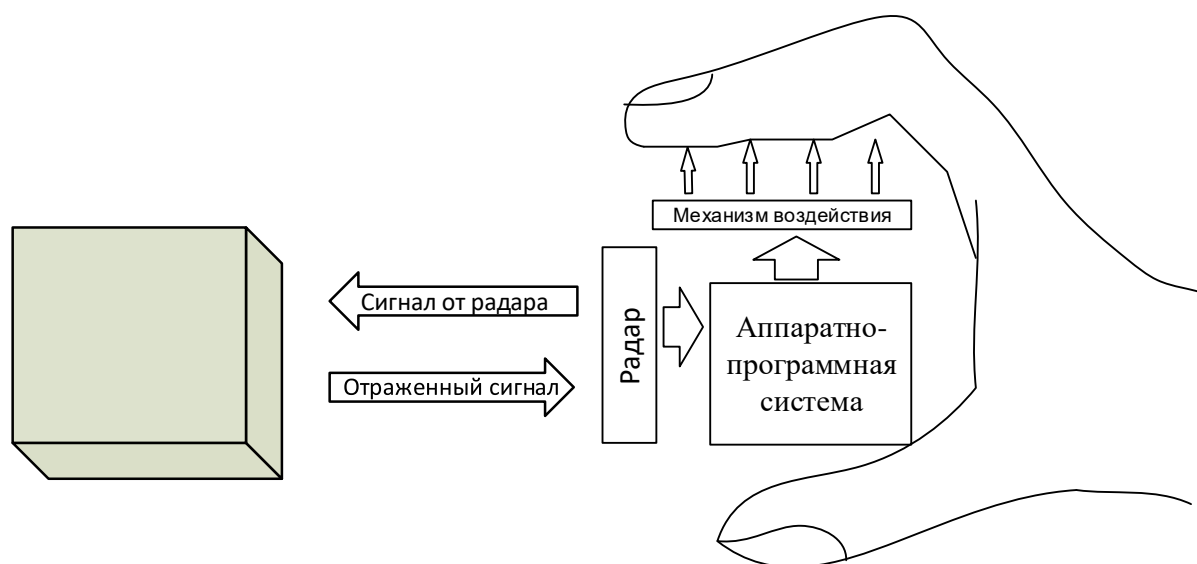


Рис. 2. Принципиальная схема работы АПС

Всего в прототипе предусмотрено шестнадцать иголочек. Матрица четыре на четыре. При этом внутренние иголки «показывают» середину, а наружные периметр «осматриваемого» пространства.

Беря во внимание текущую конфигурацию прототипа – использование однолучевого радара, без возможности определения позиции объектов и малой точностью, принцип работы АПС следующий. Радар обнаруживает все цели на дистанции до 30 метров, попавшие в угол обзора равный 14 градусам, объекты сортируются по 4 группам – до 1 метра, от 1 до 2 метров, от 2 до 3 и от 3 метров. Интерактивная часть делится на 4 группы – по 4 катушки на группу соответственно. Нижняя линия катушек индуктивности представляет группу объектов до 1 метра, вторая линия – от 1 до 2 метров и так далее. Частота движений иголочек по группам зависит от количества объектов, попавших в определенный диапазон. Если число объектов в какой-либо группе равно нулю, то соответствующая линия катушек индуктивности на интерактивной части активирована не будет. Таким образом, человек, использующий АПС понимает сколько препятствий перед ним находится и их примерное количество в каждой группе.

3.3 Устройство механизма воздействия прототипа АПС

Принцип работы устройства механизма воздействия, в дальнейшем МВ, состоит в следующем. МВ состоит из корпуса с 16-ю катушками с отверстиями в центре каждой катушки и из подвижных частей, представляющих собой иголки, которые воздействуют на кожу человека возвратно-поступательными движениями с определенной частотой. Корпус в совокупности с МВ определяет интерактивную часть системы. Для изготовления прототипа МВ созданы 3-D модели корпусов, катушек и деталей в программе Компас-3Д. Прототип МВ состоит из шестнадцати катушек индуктивности, намотанных внавалку проводом 125 мкм. Внутри катушки с зазором 1 мм помещён сердечник 3X10мм из неодимового магнита, к которому с помощью трубки подходящего диаметра прикреплена иголочка (обрезок от силовой кнопки) размером 1x5мм с одной стороны, и пружинка 2x5 мм с другой.

Трубка первоначально представляла собой кусок термоусадки, но в процессе испытаний была заменена на обрезок от «ватной» палочки с подходящим внутренним и наружным диаметром. Магнит помещен с «натягом» внутрь палочки, оставшееся сверху пространство заняла иголочка, которая была увеличена по толщине кольцом от изоляции провода подходящего сечения, а снизу пространства была установлена пружинка.

Общая длина подвижной части МВ, а именно магнита с иголочкой и пружинкой является минимальным расстоянием для определения толщины самого устройства МВ и составляет 22 мм. Схема сердечника показана на рисунке 3.

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						14
Изм.	Лист	№ докум.	Подпись	Дата		

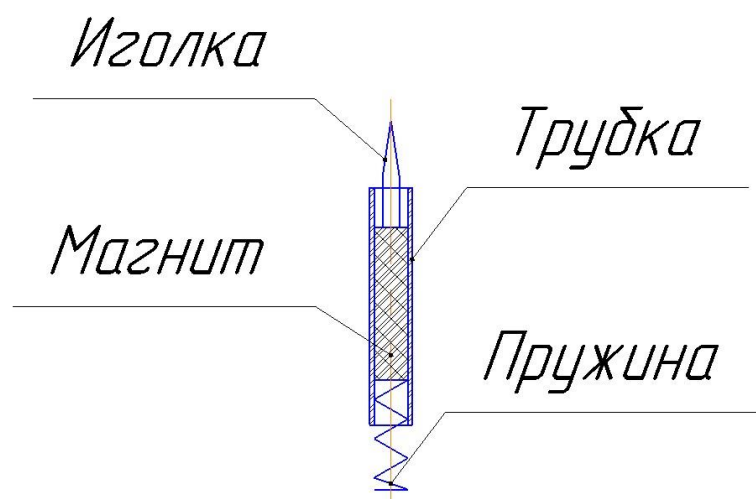


Рис. 3. Схема сердечника

При возвратно-поступательном движении иголки происходит тактильное воздействие на кожу человека с определенной и заданной частотой, которая зависит от расстояния до объекта. Таких иголок в МВ шестнадцать. Расположены эти иголки в четыре ряда по четыре строки, что представляет, как бы экран, окно в «видимый» мир для человека без зрения. По частоте воздействия он определяет удаленность от предмета, а по количеству одновременно срабатывающих иголок размеры предмета и его форму.

4 Разработка аппаратной части системы

4.1 Изготовление катушек индуктивности для механизма воздействия

Для изготовления механизма воздействия прототипа созданы 3-D модели корпусов, катушек и деталей в программе Компас-2Д. Прототип АПС состоит из шестнадцати катушек индуктивности, намотанных внавалку проводом 125 мкм. Внутри катушки с зазором 1 мм помещён сердечник 3X10мм из неодимового магнита, к которому с помощью термоусадки прикреплена иглочка (обрезок от силовой кнопки) размером 1x5мм с одной стороны, и пружинка 2x5 мм с другой.

Для получения максимальной добротности и индуктивности катушку выгоднее делать короткой, но большого диаметра, с отношением D/l порядка 2,5. Индуктивность таких катушек более точно рассчитывается по эмпирической (подобранной опытным путем) формуле:

$$L = \frac{0,1 \cdot D^2 \cdot N^2}{4 \cdot D + 11 \cdot l} \quad (1)$$

где размеры берутся в сантиметрах, а индуктивность получается в микрогенри.

В качестве D берут средний диаметр:

$$D = \frac{(D_{\max} + D_{\min})}{2} = \frac{(1,5 + 0,6)}{2} = 1,05 \text{ см} \quad (2)$$

а в качестве l - ширину намотки,

$$l = \frac{(D_{\max} - D_{\min})}{2} = \frac{(1,5 - 0,6)}{2} = 0,45 \text{ см} \quad (3)$$

Число витков катушки для такого сердечника было подобрано опытным путем. Так, начиная с 72 витков при которых наблюдались незначительные колебания, а при увеличении числа витков амплитуда колебаний возрастала, было доведено число витков до 2000. Больше увеличивать число витков было неэффективно, так как это увеличивало вес прототипа. Размеры каждой катушки (диаметр-высота) составляют 16x19 мм. Намотка осуществлялась на самодельном станке с электроприводом и счетчиком витков изготовленном из детского конструктора и калькулятора, показанном на рисунке 4.

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		16

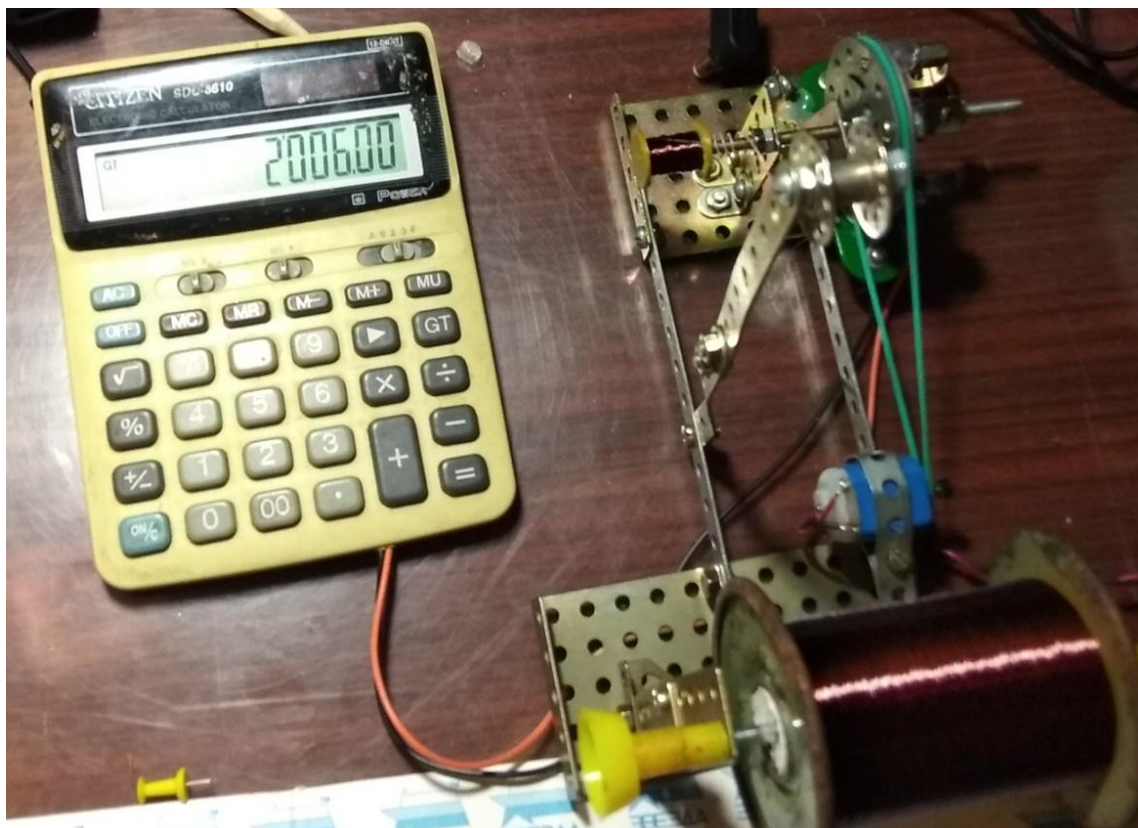


Рис. 4. Станок для намотки катушек с счетчиком витков.

Катушки и корпус, в котором они крепятся напечатаны на 3-D принтере из пластика PLA. Фрагмент чертежа представлен на рисунке 5.

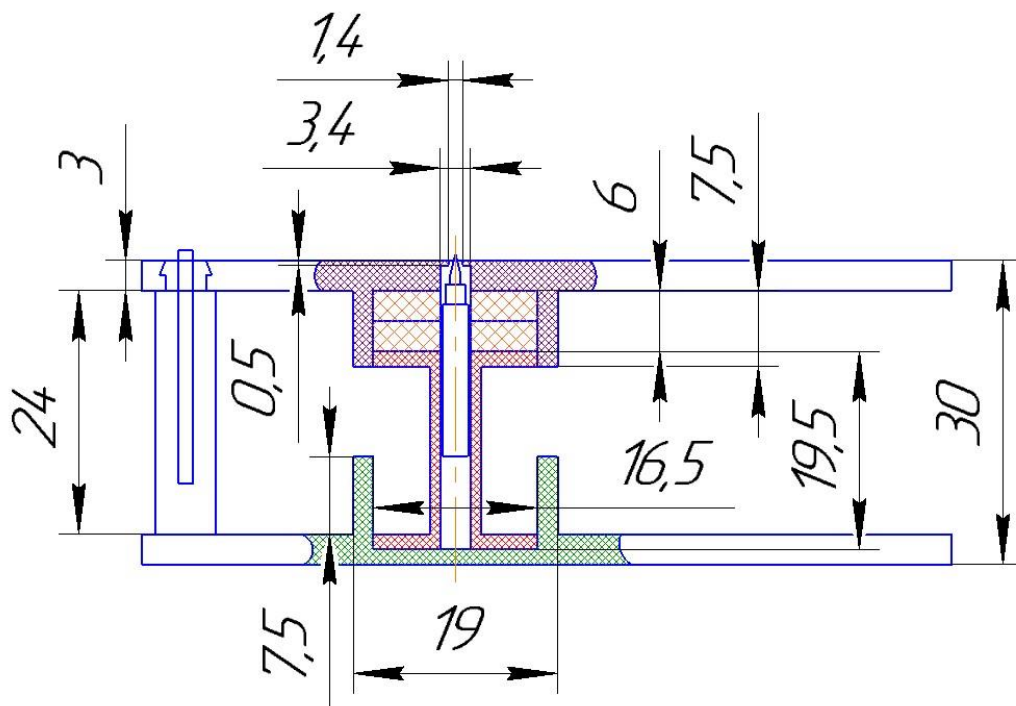


Рис. 5. Фрагмент чертежа с размерами катушки

Опытным путем было установлено, что наибольшая амплитуда достигалась при соединении катушки с цифровым выходом Arduino Mega 2560. Частота изменения напряжения на ножке контроллера задается программно. Было рассмотрено еще два варианта активации МВ – подача синусоидального напряжения через генератор и использование широтно-импульсной модуляции средствами Arduino Mega 2560. Как синусоидальное напряжение, так и ШИМ давали слабый эффект – МВ двигался почти неощутимо, в то время, как прямоугольный импульс, достигаемый изменением напряжения на цифровых выходах микроконтроллера показывал наибольшую амплитуду движения МВ.

4.2 Проектирование и изготовление корпуса для механизма воздействия

Корпус для механизма воздействия должен удовлетворять нескольким требованиям:

1. По размерам быть сопоставимым с ладонью взрослого человека.
2. Вмещать в себя 16 катушек индуктивности.
3. Быть разборным – состоять из нескольких деталей. Так, появляется возможность замены катушек индуктивности на этапе тестирования.

Для изготовления некоторых деталей для прототипа и уменьшения стоимости, был рассмотрен и принят вариант печати на 3-D принтере. Чтобы печать была успешной, необходимо знать не только геометрические размеры деталей из которых состоит прототип, но и способ посадки, необходимые зазоры и натяги, а эти параметры во многом зависят от выбранного материала для изделий, скорости печати, температуры и многих других параметров.

Первая версия корпуса получилась недостаточно прочной, так как прочностные параметры применяемого распространённого пластика не были известны. Для увеличения амплитуды движения сердечника общей длиной 20мм, высоту корпуса пришлось увеличивать дополнительно на 6мм. С учетом минимального воздействия соседних катушек, общий размер корпуса для катушек составил: 80x80x30 мм (ширина-глубина-высота).

Для обеспечения достаточной прочности корпуса принято решение основные плоскости делать не менее 3 мм, а размеры соединительных отверстий для винтов принять не менее 2 мм. После нескольких практических экспериментов с моделями, напечатанными на 3-D принтере, принято решение об окончательной форме и размерах корпусов.

В Компасе, также, как и в Автокаде и любой другой аналогичной программе довольно просто получить 3-D модель детали любой сложности. Не будем вдаваться в способы черчения, но определим основные моменты получения правильной 3-D детали.

Все изображения, строящиеся на плоскости, состоят из двух простейших фигур: отрезка и дуги. Остальные, более сложные виды фигур

состоят только из различных комбинаций простейших фигур. Первый этап разработки 3-D модели состоит в том, чтобы создать в эскизе замкнутый контур детали на плоскости, например, на рисунке 6 слева показан вид сверху модели корпуса механизма воздействия.

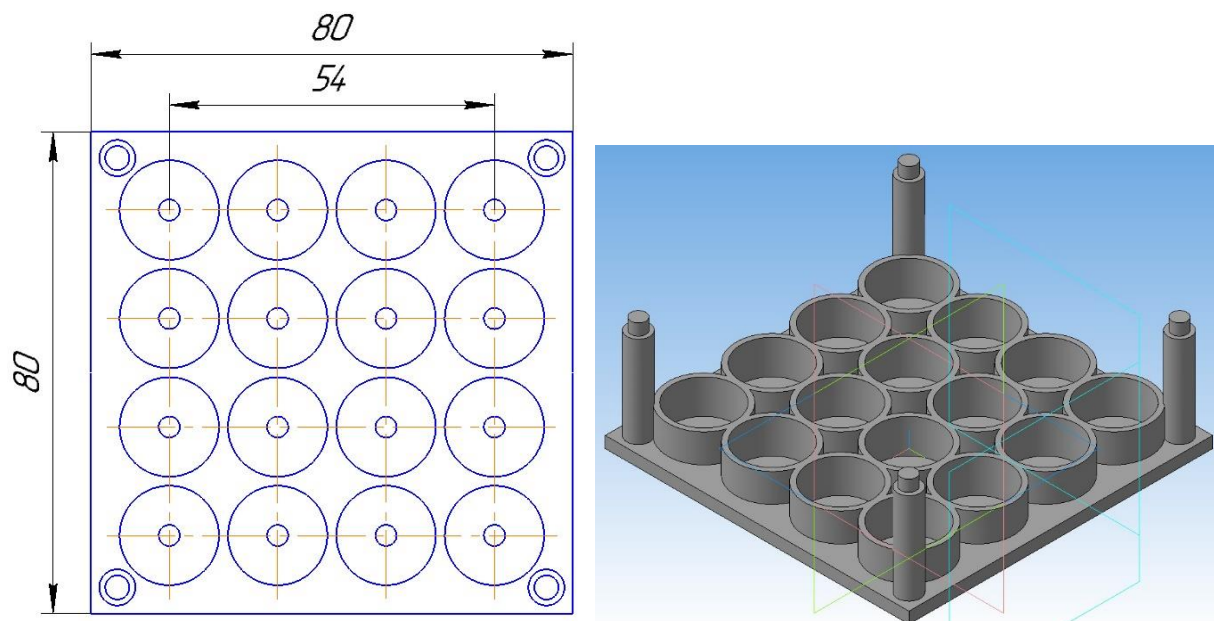


Рис. 6. Корпус для крепления катушек. Вид сверху. Вид 3-D модели

Вторым этапом этот контур превращаем в 3-D объект методом «выдавливания» на определенную величину. Дальнейшие действия по построению 3-D модели сводятся к повторению этих двух этапов, эскизы профилей строятся в нужных плоскостях, а затем выдавливаются или вырезаются на определенное расстояние. Существуют и многие другие способы получения 3-D фигур вращение, например, с помощью операций «вращения», «по сечениям» и т.д.

Крышка корпуса МВ показана на рисунке 7. Она представляет собой верхнюю часть повторяющую форму нижней части корпуса, но в середине каждой катушки сделаны отверстия для выхода игловок. Для предотвращения заклинивания игловок в корпусе, отверстия выполнены больше диаметра иглолки на 0,2 мм.

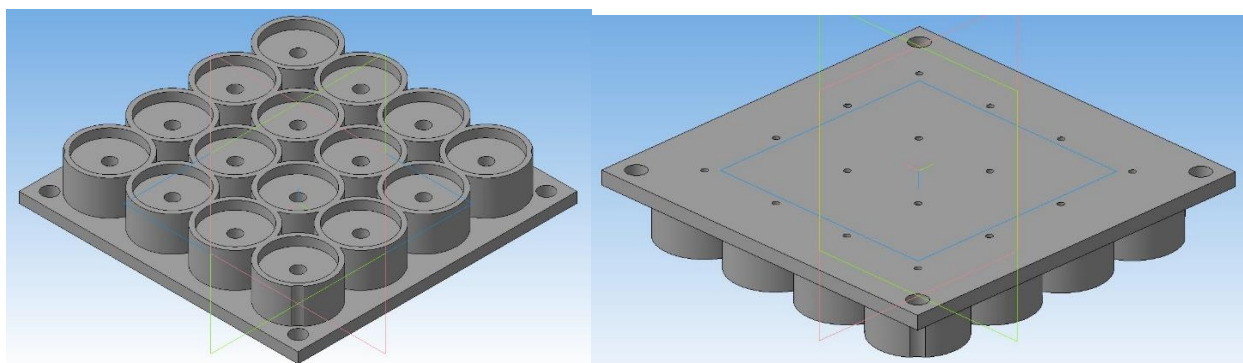


Рис. 7. Крышка корпуса. Два вида 3-D модели

Полученные файлы сохраняются с расширением «*.stl». Следующим шагом необходимо преобразовать файл 3-D модели в файл для печати на 3-D принтере. Некоторые принтеры позволяют осуществлять печать непосредственно из программы, в которой получены 3-D модели, то есть печатают файлы «*.stl».

Для печати непосредственно с подключенного к принтеру ПК можно запустить программу CURA, которая идет в наборе с принтером и запустить печать. Зачастую печатать из программы не удобно, так как время печати может достигать нескольких часов, а иногда и суток и все это время будет задействован ПК. Для печати с флеш-карты без участия компьютера можно подготовить специальный файл печати с расширением «*.gcode». Программа CURA, показана на рисунке 8, как раз и предназначена для создания такого файла из файла «*.stl».

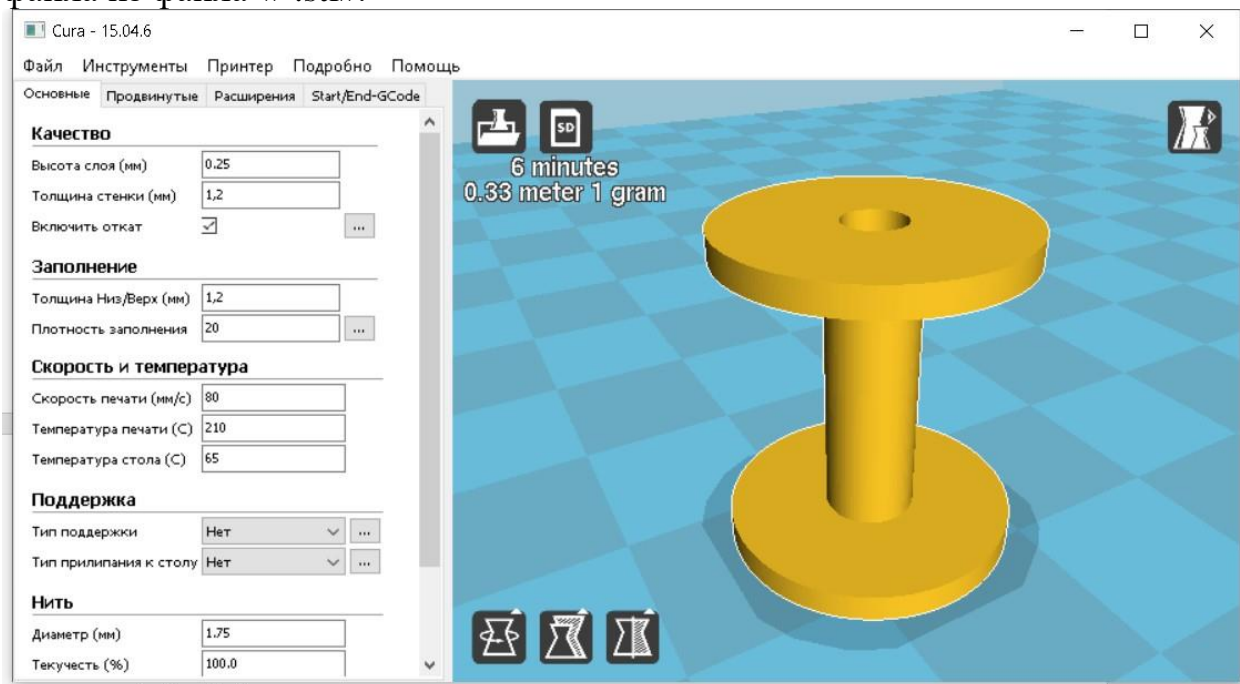


Рис. 8 Программа CURA – подготовка модели к печати

При соответствующих настройках программы, в которой нужно выбрать принтер можно получить готовый файл для печати.

Настройки для печати, а именно высота слоя, толщина стенки, плотность заполнения, скорость и температура печати, время печати отображаются сразу. На рисунке N видно, что время печати одной катушки составляет 6 минут, длина нити - 0,33 метра и вес готового изделия 1 грамм.

Если сохранить эти настройки в файл с расширением «*.gcode» на флеш-карту и открыть его из меню 3-D принтера, то принтер будет печатать с учетом всех настроек самостоятельно без участия ПК.

Корпус катушки, выполненный в программе Компас-3D, готовая катушка в сборе с обмоткой показаны на рисунке 9. После намотки провода,

припаивания выводов и защиты обмотки, получилось шестнадцать готовых катушек.

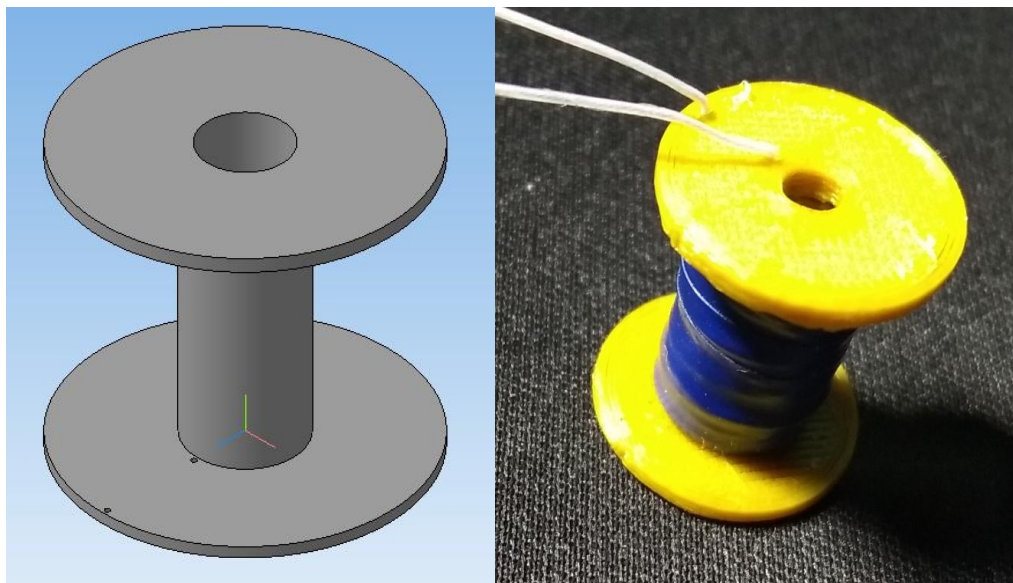


Рис. 9. 3D-Модель корпуса катушки. Готовая катушка в сборе

После сборки катушек в корпус и установки сердечников, каждая была соединена проводом с разъемом, проводники которого выведены на контакты платы Arduino Mega 2560. Разъем для этого использован от FDD с 40 контактами, которые по шагу совпадают с областью нахождения цифровых входов/выходов с 20 по 55 на Arduino Mega 2560. Интерактивная часть системы в сборе показана на рисунке 10.

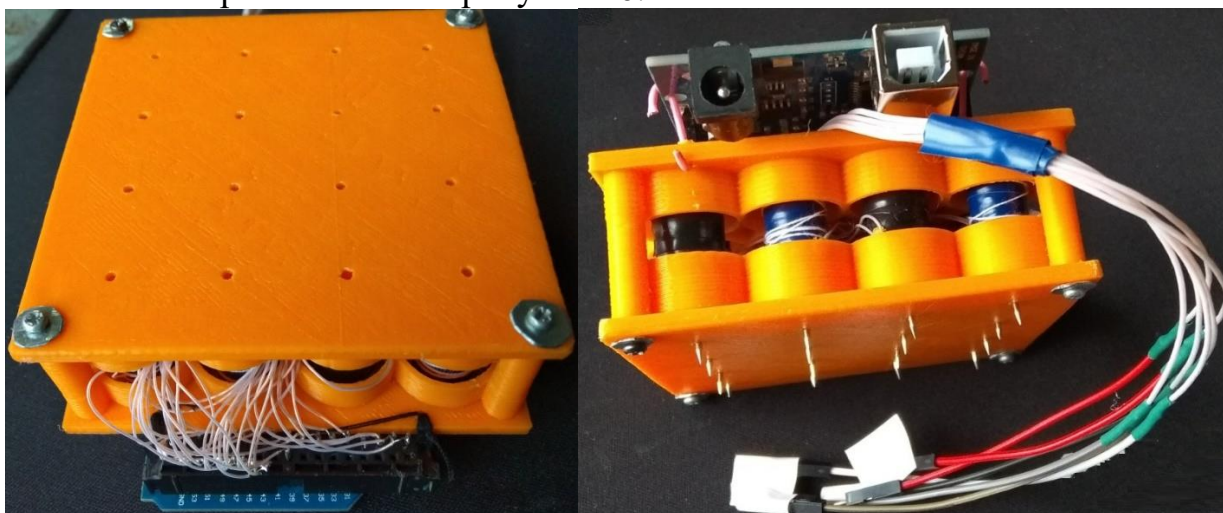


Рис. 10. Вывод проводников катушек на разъем Arduino Mega 2560

5 Разработка программной архитектуры системы

5.1 Обзор программной архитектуры

Программная часть состоит из нескольких модулей:

1. Программа AutoRadarControl.exe.

Считывает данные от радара с Ethernet порта, записывает информацию об обнаруженных целях в результирующий файл. Это приложение считается внешним, так как оно поставлялось с радаром. Не имея возможности переписать исходный код, возникает вопрос о передачи данных на микроконтроллер. Для этой задачи разработан скрипт взаимодействия внешней программы с микроконтроллером - conversionDataAPS.py.

2. Скрипт conversionDataAPS.py.

Данное инфраструктурное приложение представляет из себя скрипт, написанный на языке Python 3.x, основной функций которого является обработка и передача полученных от AutoRadarControl.exe данных к микроконтроллеру в удобочитаемой форме. Приложение имеет ряд дополнительных функций по автоматизации и обработке ошибок – проверка на существование и автоматический запуск приложения, создание и очищение файлов. Подробнее функциональность скрипта будет рассмотрена в 5 главе.

3. Скрипт interactivePartAPS.ino.

Фактически является прошивкой для микроконтроллера. Написан на языке Ардуино, который является надстройкой над C/C++. Микроконтроллер читает последовательный порт, который открывает инфраструктурный скрипт conversionDataAPS.py, на основе считанных данных распределяет количество целей по группам и активирует ряд катушек с частотой в соответствии с количеством целей, относящихся к данной группе.

Одновременная передача данных микроконтроллером на цифровые выходы, для активации МВ, может быть достигнута двумя путями – использование многоядерного микроконтроллера или использование псевдопараллелизма на одном ядре. Многоядерные микроконтроллеры дорогостоящие и, зачастую, громоздкие, поэтому был выбран вариант с использованием псевдопараллелизма. Частота 16 МГц микроконтроллера ATmega2560 позволяет использовать библиотеки для имитации параллельного выполнения программы. Быстрое переключение между потоками будет незаметно для пользователя.

5.2 Разработка скрипта взаимодействия радара и микроконтроллера

В связи с тем, что программное обеспечение для радара не содержало открытый код, а написать собственную программу по приему данных с радара не было возможным по причине отсутствия документации, мною было

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		22

принято решение разработать скрипт, автоматизирующий запуск стороннего приложения и обработку данных получаемых с радара.

Программа AutoRadarControl.exe имеет пользовательский графический интерфейс. Для приема данных с радара необходимо подключить его по сетевому кабелю к компьютеру. Запуск основного процесса программы открывает пользовательское меню, показанное на рисунке 11.

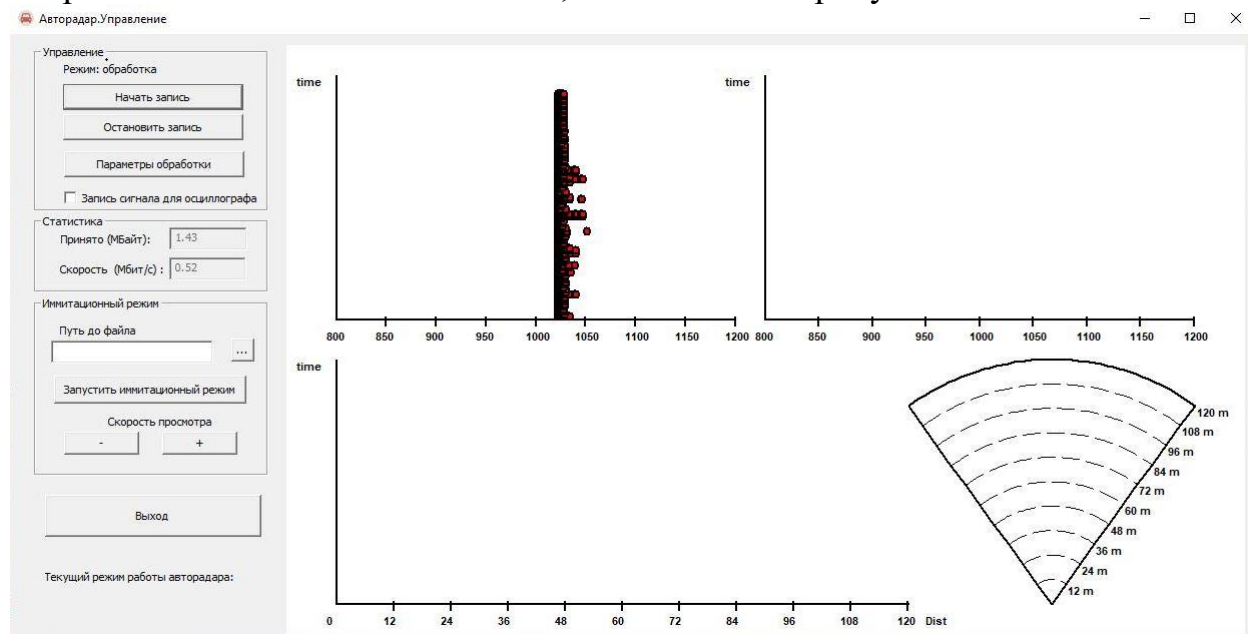


Рис. 11. Пользовательский интерфейс программы приема данных с радара

Для начала записи в результирующий файл принятых с радара данных, необходимо нажать кнопку «Начать запись», а для останова – «Остановить запись». Начало записи инициирует создание 4-ех логирующих файлов в подкаталоге нахождения выполняемой программы. Файлы непрерывно заполняются данными определенной структуры: «[номер цели] [номер зоны] [амплитуда]».

1. Номер цели – идентификатор обнаруженного объекта, целочисленное число от 0 до N.

2. Номер зоны – с текущей конфигурацией радара всегда имеет значение «0».

3. Амплитуда – величина, характеризующая объект. Для получения расстояния до цели в метрах, амплитуда умножается на коэффициент «0,527».

Исходя из исследования работы программы AutoRadarControl.exe мною был выявлен ряд трудностей, связанных с получением и обработкой данных:

1. Программа AutoRadarControl.exe открывает логирующий файл и непрерывно записывает в него данные. Это затрудняет открытие файла другим процессом и сужает список возможных функций для этого.

2. Непрерывная запись в логирующий файл затрудняет отбор значимых данных – помимо информативных данных, в файл также попадают и пустые значения.

3. Размер файла растет во время работы программы AutoRadarControl.exe. Указатель на позицию записи в логирующем файле находится в памяти сторонней программы, поэтому очищать файл не имеет смысла – запись продолжится с прежнего места, а все содержимое файла до указателя заполнится пробелами.

С учетом вышеперечисленных трудностей был разработан инфраструктурный скрипт conversionDataAPS.py, решающий как основные задачи, так и выполняющий сопутствующую автоматизацию. Основные задачи скрипта перечислены ниже:

1. Считывать данные, получаемые программой AutoRadarControl.exe от радара.

2. Отсеивать неинформативные, пустые строки.

3. Обработать данные – получать из амплитуды расстояние до объекта в метрах.

4. Передавать обработанные данные микроконтроллеру.

Такая функциональность является достаточной для работы АПС, но без должной автоматизации процесс включения системы усложняется и требует ручных манипуляций. Исходя из этого, скрипт conversionDataAPS.py дополнен соответствующими проверками и автоматизацией:

1. Проверка путей, каталогов и ключевых файлов на существование.

2. Сопровождение отображаемыми сообщениями о текущем состоянии выполнения скрипта – успешное или не успешное открытие каталогов, запуск процесса, результат выполнения проверок, обработанные данные.

3. Обработка исключений при выделении значимых данных.

4. Автоматический запуск приложения AutoRadarControl.exe и навигация по графическому интерфейсу пользователя с целью начала считывания данных с радара.

5. Автоматическое завершение процесса AutoRadarControl.exe, при инициации завершения работы скрипта.

6. Удаление логирующих файлов в начале работы. Новый процесс AutoRadarControl.exe создает уникальные логирующие файлы, поэтому, с целью освобождения места, старые файлы удаляются.

7. Представление обработанных данных в удобочитаемом формате. Структура данных для микроконтроллера имеет следующий формат: «[номер группы] [количество целей]».

Ниже приводится описание алгоритма получения и обработки данных.

Основной цикл считывания и преобразования данных начинается после успешно пройденных проверок на существование путей, каталогов и файлов, начала выполнения процесса AutoRadarControl.exe и создания логирующих файлов. Файл AutoRadarResultLog.txt, содержащий необработанные данные,

открывается средствами встроенной в Python 3.x библиотеки `os`, предназначенной для работы с операционной системой. Метод `os.read()` позволяет читать данные из файла, открытого другим процессом под запись. За один проход по циклу считывается 1 Кб данных. Условием начала обработки данных является успешная проверка на наличие считанной информации, иначе попытки прочесть файл будут продолжаться. После успешной попытки прочесть файл, данные разделяются на строки с попутным удалением сторонних символов. Далее из строк выделяются значимые цифры – номер цели и амплитуда. Амплитуда переводится в величину расстояния до объекта выраженной в метрах. Каждая цель определяется к одной из 4 групп: объекты до 1 метра, от 1 до 2 метров, от 2 до 3 и от 3 метров. Результирующие строки групп с количеством в них целей записываются в последовательный порт, который читается микроконтроллером. На рисунке 12 приведена блок-схема цикла чтения и обработки данных, полученных с радара.

В результате был разработан вспомогательный скрипт, выполняющий обработку строк, полученных от внешней программы. Такой подход – использование промежуточного скрипта на внешней ЭВМ снизит нагрузку на микроконтроллер, и облегчит разработку прототипа АПС. В перспективе, при замене однолучевого радара на FMCW радар, необходимость в скрипте пропадет.

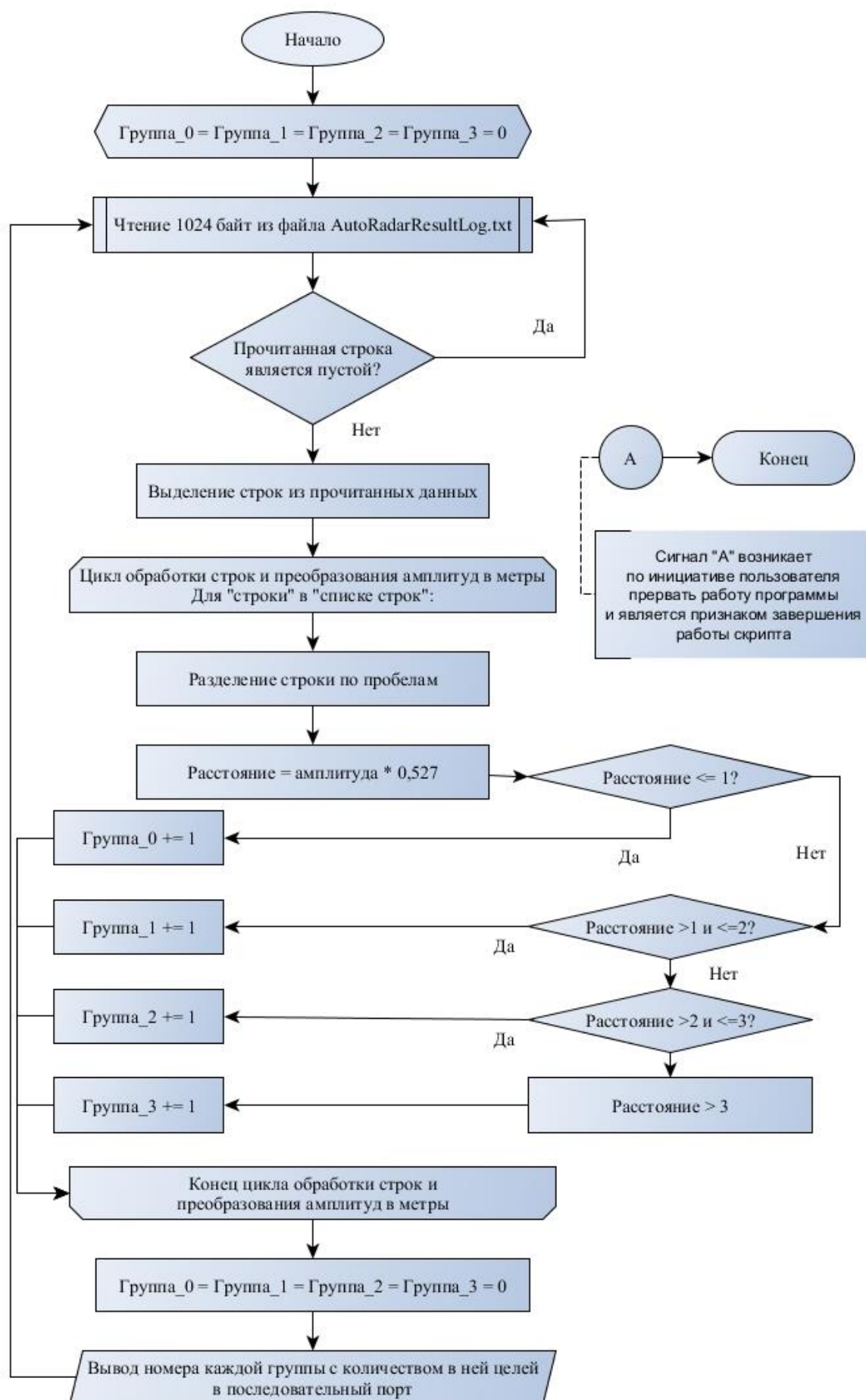


Рис. 12. Блок-схема алгоритма чтения и обработки данных

5.3 Разработка программы для АПС

Программа фактически является прошивкой для микроконтроллера ATmega2560, написанной на языке Ардуино. Назначение ее – устанавливать напряжение на катушках индуктивности в соответствии с данными, полученными от радара. Первоначально программа для микроконтроллера разрабатывалась независимо от радара, поэтому мною была предусмотрена возможность изменения частоты движения каждой иглолки в катушках, без привязки к группам. Arduino Mega 2560 поддерживает 6 аппаратных таймеров-счетчиков, способных генерировать прерывания независимо друг от друга. Библиотеки, реализующие псевдопараллелизм, позволяют задействовать ресурсы платформы для разделения процесса во времени на 16 потоков. Пользователь не сможет различить столь малые задержки в обслуживании МВ, поэтому такой подход вполне оправдан.

Основным назначением программы для микроконтроллера является непрерывное обслуживание МВ. Мною был сформулирован ряд требований к программе:

1. Непрерывное прослушивание последовательного порта и установка полученных данных в качестве частоты движения МВ.
2. Реализация механизма обработки ошибок – создание протокола взаимодействия по последовательному порту с проверкой старт и стоп битов.
3. Оптимальное использование ресурсов микроконтроллера для создания эффекта параллельной работы – задействованы библиотеки, реализующие псевдопараллелизм.

Для анализа наличия каких-либо данных на последовательном порту используется функция `serialEvent`, тело которой выполняется каждый раз после основного цикла программы, если на последовательном порту имеются данные. После конфигурации – инициализации переменных, установки соединенных с катушками ножек Arduino как выходов и привязки функций обработки катушек к потокам, начинается основной цикл программы. В основном цикле программы вызывается функция `frequency_matching()`, определяющая какому(им) из 16 потоков наступило время выполнения. Каждый поток выполняет простую задачу – изменяет напряжение на соответствующем выходе Arduino на противоположное. Перед следующим выполнением цикла функция `serialEvent` прослушивает последовательный порт и пытается считать из него данные с соответствующими проверками на старт и стоп биты. Когда данные успешно считаны – вызывается функция `setFreq()`, отвечающая за выделение из пакета идентификатора группы и количества целей с последующей установкой частоты на группах катушек (частоты выполнения потоков).

На рисунке 13 приводится блок схема вышеописанного алгоритма.

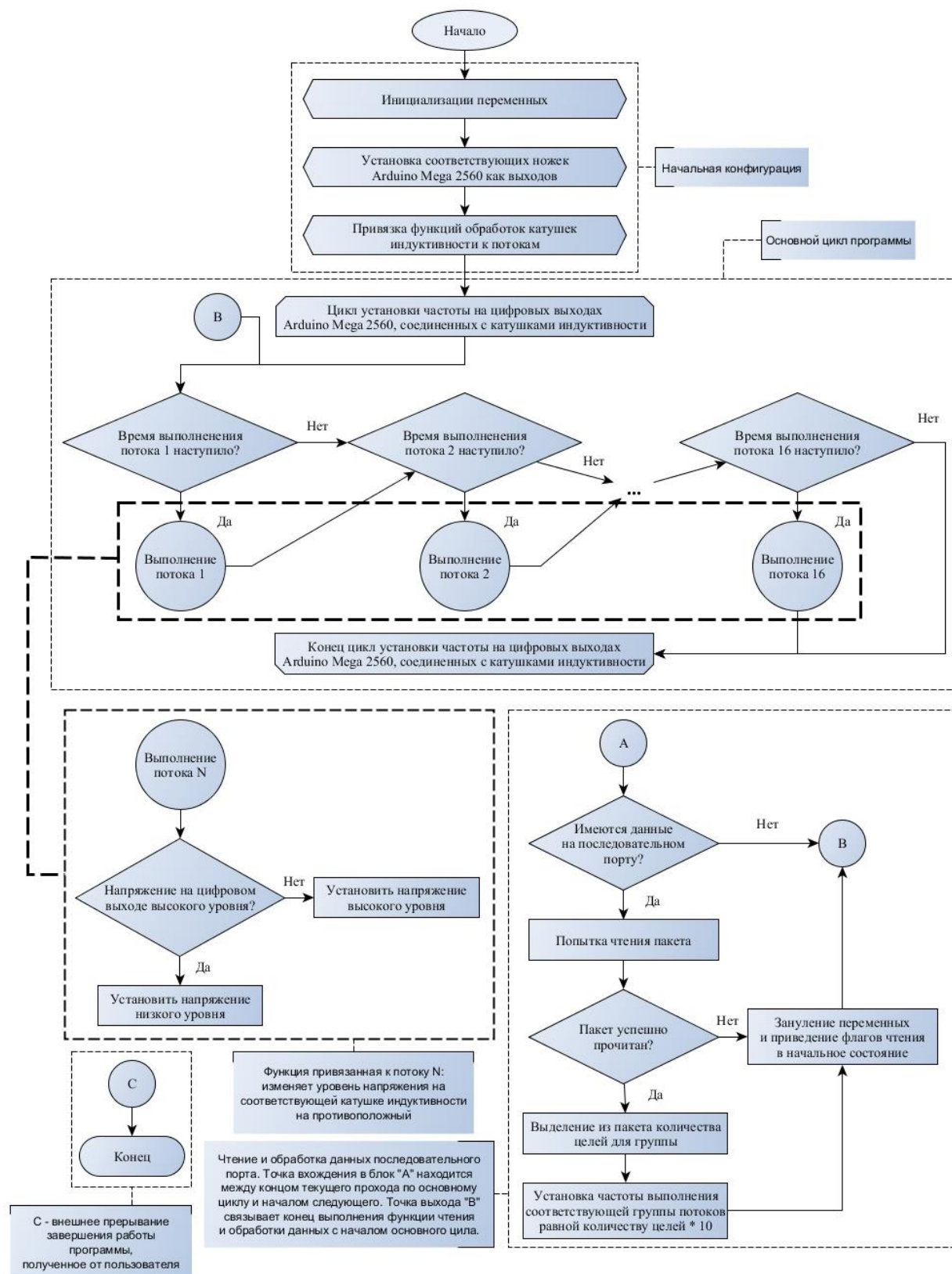


Рис. 13. Блок-схема алгоритма установки частоты изменения напряжения на катушках индуктивности

6 Тестирование системы

Процесс разработки аппаратно-программной системы состоял из нескольких этапов, на каждом из которых проводились соответствующие тесты. Ниже перечислены этапы разработки с описанием проводимого тестирования.

1. Разработка и изготовление катушек индуктивности.

Этот этап был выбран первым для ознакомления с характером тактильного воздействия иголок на руку человека, зависящего от материала сердечника, частоты изменения напряжения и его величины, количества витков и размера катушки. Для выявления оптимального воздействия иголки на руку человека был проведен ряд экспериментов, заключающихся в варьировании размеров прототипов катушки, количества витков, изменении формы сигнала напряжения. На рисунке 14 показано два прототипа катушки, отличающихся размерами и пластиком, иголочка с неодимовым магнитом и генератор сигналов, для выявления оптимальной формы напряжения.

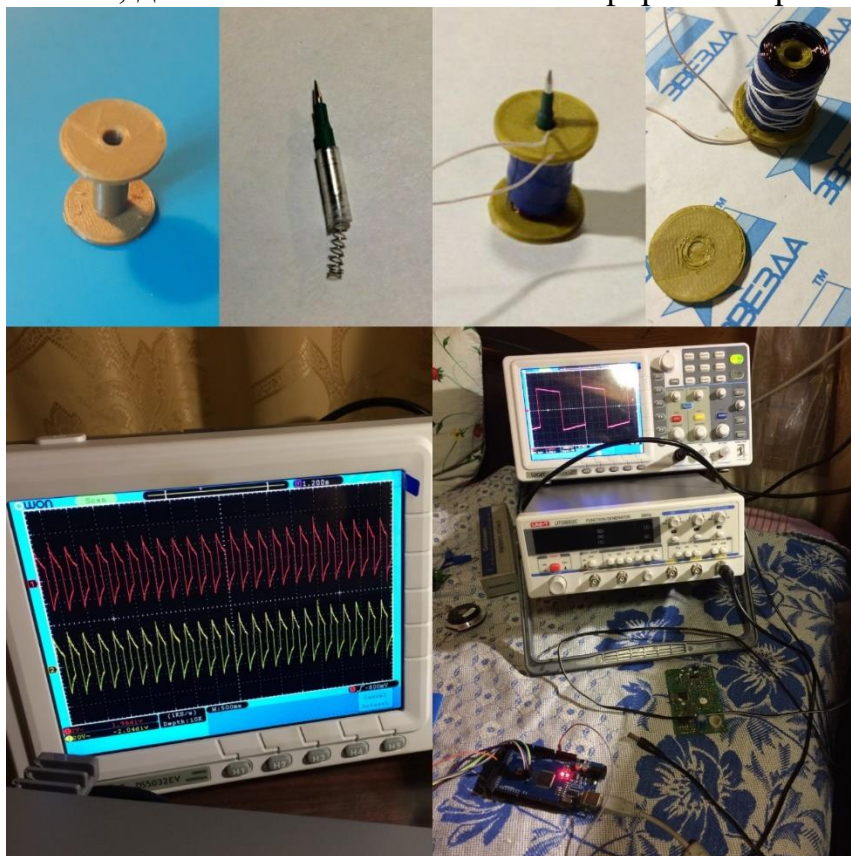


Рис. 14. Процесс разработки и тестирования катушек индуктивности

2. Разработка и изготовления корпуса механизма воздействия.

Корпус, в который помещаются катушки имеет две версии. Отличие первой версии от второй в высоте крепления катушек и механизме соединения крышки и дна корпуса. На рисунке 15 показаны версии корпусов – сверху первая версия, снизу - вторая. Во время тестирования особое внимание

амплитуде выдвижения иглоочек. Амплитуда выдвижения напрямую влияет на тактильное ощущение пользователя. Первая версия корпуса имела грубые недостатки – иглоочки застревали в отверстиях, а механизм крепления крышки был не надежным. Со второй версией вышеупомянутые недостатки пропали.

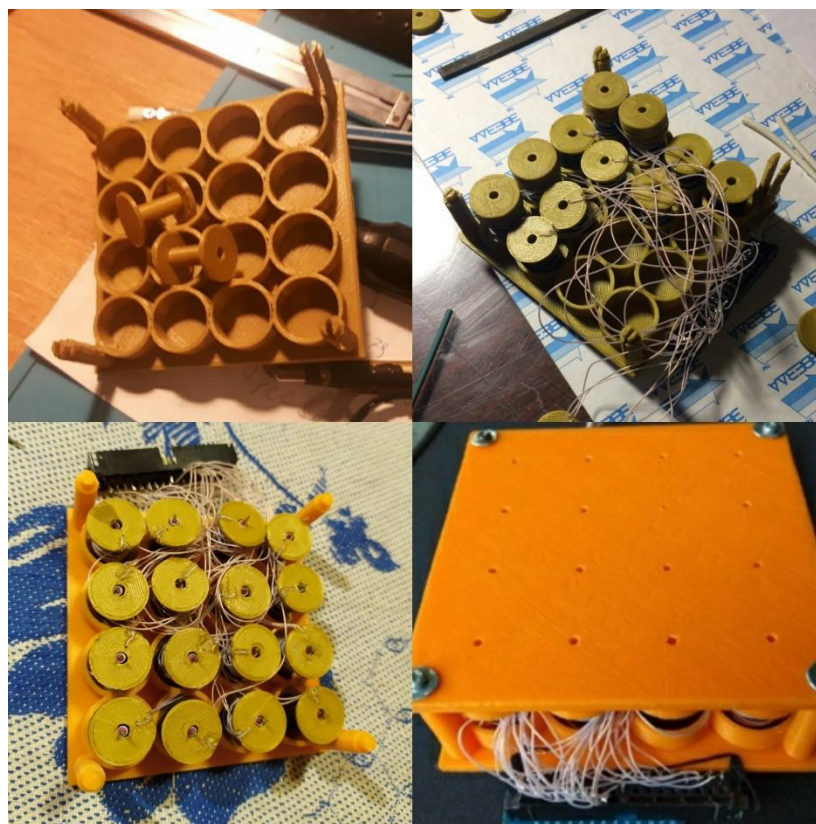


Рис. 15. Разные версии корпусов механизма воздействия

3. Разработка скрипта взаимодействия радара с микроконтроллером.

Тестирование скрипта взаимодействия с микроконтроллером проводилось в несколько этапов на протяжении всей разработки. Скрипт содержит проверки на существование путей, файлов и каталогов с соответствующими выводами сообщений в консоль, которые были протестированы в первую очередь.

Вторым этапом тестировалась автоматика – запуск приложения и нажатие кнопок, очищение каталога, автоматическое закрытие программы, при завершении работы скрипта.

На третьем этапе тестирования проверялась логика выделения значимых данных из результирующего файла, создаваемого сторонней программой. Байтовая строка размером 1Кб, прочитанная из файла должна быть разделена на строки и слова. Строки выделяются по символу новой строки ‘\n’, а слова по пробельным символам. Перевод байтовой строки к символьному типу оставляет сопровождающие символы, например, “b” в начале строки – остаток от типа данных byte. Такие закономерные символы, оставшиеся от приведения типа можно предсказать и обработать, однако, при

потере связи с радаром, данные могут потерять свой формат в промежутке подключения. Для обработки такого сценария используются исключения, обрабатывающие неверное приведение типа данных из символьной строки к целочисленному значению.

На 4 этапе тестировалась непрерывность чтения из файла. Для облегчения процесса разработки был написан сопутствующий скрипт, генерирующий данные похожего на результирующий файл стороннего приложения формата. На этом этапе выявлена проблема очищения результирующего файла – при удалении содержимого файла, указатель места записи сторонней программы не изменяется. Размер файла продолжает расти с прежнего места. Таким образом, очищать результирующий файл бесполезно. Единственное решение без переписывания кода сторонней программы – автоматизировать остановку записи, перезапуск приложения и удаление файла, однако, такой подход в рамках разработки прототипа не оправдан.

Последним этапом тестировалась запись обработанных данных в последовательный порт. Для этого использовалось свободно распространяемое приложение «PuTTY», позволяющее открывать последовательные порты с возможностью мониторинга.

Для отладки скрипта, а также для информирования пользователя создана консоль в строго зафиксированном месте экрана – справа снизу, причем стороннее приложение радара открывается слева вверху, таким образом, интерфейсы не перекрывают друг друга. Вывод начала работы скрипта в консоль показан на рисунке 16.

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		31

4.1 Расстояние до препятствия менее метра.

На рисунке 17 справа можно увидеть, что радар находится на расстоянии около 0.9 метра от стены. По отображаемым данным в консоли, показанным на рисунке 16 слева, видно, что 3 объекта определены в первую группу, а количество целей во всех остальных группах равно нулю, что соответствует ожиданиям. Можно привести примерные расчеты и объяснить почему количество объектов в первой группе равно 3. При угле излучения радара равном 14 градусам и расстоянию до стены в 0.9 метра, область, в которую могут попасть лучи радара равна примерно 0,4 метра. Не имея документации к радару, могу предположить, что в такой промежуток попадает 3 луча.

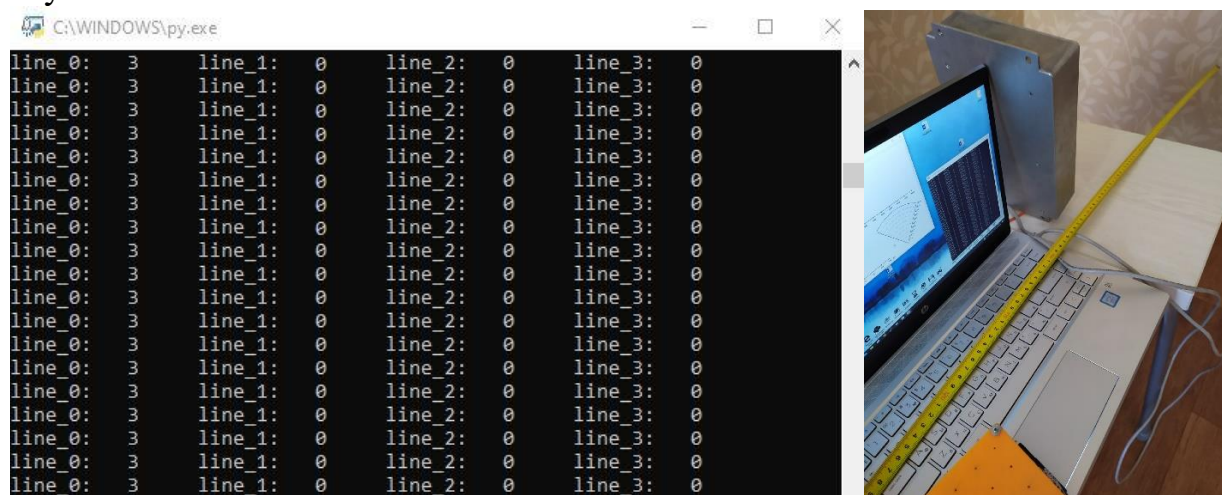


Рис. 17. Тестирование АПС с расстоянием до объекта менее 1 метра

При данном расположении радара была активирована первая группа катушек на интерактивной части с частотой 30 Гц, что по субъективным ощущениям весьма ощутимо и дает представление о препятствии. Таким образом пользователь понимает, что перед ним находится препятствие на расстоянии менее 1 метра.

4.2 Расстояние до препятствия от 1 до 2 метров.

Для проверки попадания препятствия во вторую группу радар был установлен на расстоянии 1,65 метра от стены. На рисунке 18, слева, видно, что обнаруженные цели были отнесены к группе объектов от 1 до 2 метров. Как и в предыдущем эксперименте – частота движения иголок составила около 30 Гц. Таким образом, пользователь сможет понять, что перед ним сплошной объект на расстоянии от 1 до 2 метров.

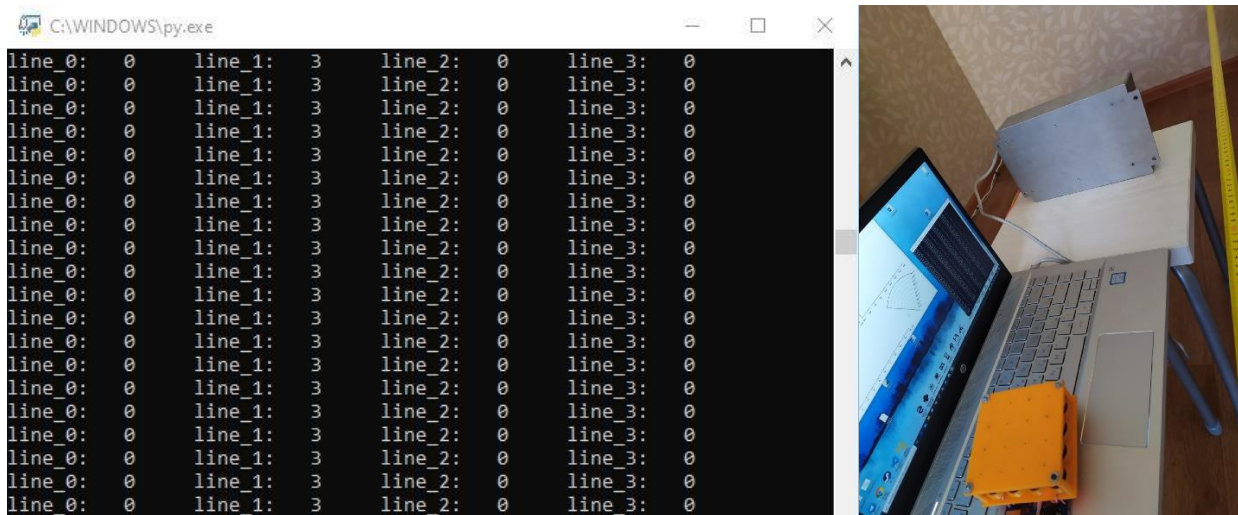


Рис. 18. Тестирование АПС с расстоянием до объекта от 1 до 2 метров

4.3 Расстояние до препятствия от 2 до 3 метров.

Подобным образом произведено тестирование для группы объектов от 2 до 3 метров. Обнаруженные цели были отнесены к 3 группе. На рисунке 19 видно, что радар размещен на расстоянии около 2,5 метров от стены.

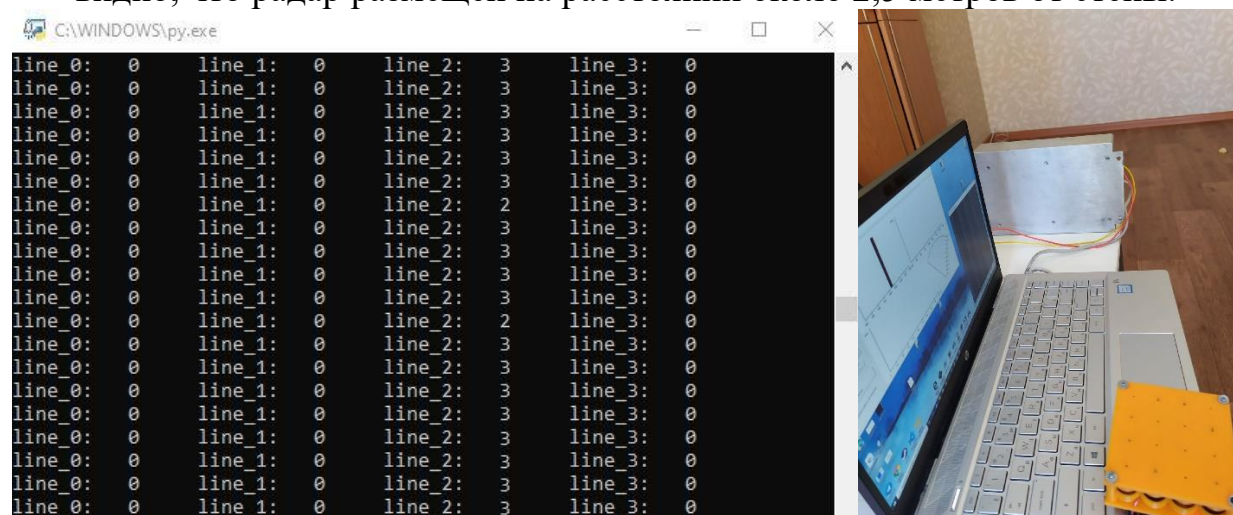


Рис. 19. Тестирование АПС с расстоянием до объекта от 2 до 3 метров

На интерактивной части активировался 3 ряд катушек с частотой около 30 Гц.

4.4 Препятствие, удаленное на более чем 3 метра.

Представим ситуацию – перед слепым человеком находится объект, за которым, через некоторое расстояние расположена стена. Чтобы протестировать данный сценарий радар был направлен на узкий столб, попадающий в группу объектов от 1 до 2 метров, а за столбом располагалась стена, отдаленная более чем на 3 метра от АПС. На рисунке 20 показан вывод отсортированных объектов по группам в консоль.

```

C:\WINDOWS\py.exe
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 1    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 2    line_2: 0    line_3: 3
line_0: 0    line_1: 1    line_2: 0    line_3: 3
line_0: 0    line_1: 1    line_2: 0    line_3: 3
line_0: 0    line_1: 0    line_2: 0    line_3: 3
line_0: 0    line_1: 1    line_2: 0    line_3: 3

```

Рис. 20. Тестирование АПС с двумя объектами: от 1 до 2 метров и от 3 метров

Как и предполагалось – активировался 2 и 4 ряд интерактивной части. Причем, площадь стены больше площади столба, поэтому и лучей, отразившихся от нее больше. Так, человек с ограниченными зрительными возможностями сможет понять, что перед ним на расстоянии от 1 до 2 метров находится объект, а за ним препятствие большего размера.

По проведенным тестированиям системы можно считать разработку прототипа аппаратно-программной системы для помощи людям с ограниченными возможностями для ориентации в пространстве успешной. По ходу тестирования на разных уровнях разработки и проектирования системы, было поставлено и решено множество вопросов. Тесты системы в сборе подтвердили успех разработки прототипа АПС.

Заключение

Данная выпускная квалификационная работа была направлена на изучение вопроса помощи людям с ограниченными возможностями для ориентации в пространстве. С каждым годом появляются инновационные изобретения, облегчающие жизнь слепым людям. Окружающая современного человека среда непрерывно совершенствуется и усложняется. Людям с ограниченными возможностями становится все сложнее ориентироваться в пространстве без дополнительных приспособлений. Создание прототипа такого устройства и было целью данной работы.

Требования, которым разработанный прототип аппаратно-программной системы должен отвечать, были выполнены:

1. Передаваемая информация о пространстве в виде механических движений, интуитивно понятна для человека – пространственные объекты разделены на группы, отличающиеся расстоянием, каждой группе привязана часть интерактивной системы. При попадании объекта в конкретную группу активируется определенная часть механизма воздействия, информируя тем самым пользователя посредством механических воздействий о препятствии.

2. Расстояние до препятствий определяется достоверно, объекты распределяются по 4 группам. Данное требование подтверждено тестированием АПС с разными сценариями.

3. Предусмотрена расширяемая архитектура, как программной, так и аппаратной части системы. Система состоит из нескольких заменяемых частей. Так, улучшив радар и адаптировав программное обеспечение можно добиться улучшения качества определения объектов. Интерактивная часть системы поддерживает избыточное для данного радара, используемого в прототипе, количество механизмов воздействия. Таким образом, предусмотрена возможность улучшения аппаратной части устройства. Расширяемость программной архитектуры предусмотрена в следующем – нет строгой привязки к группам объектов, частота задается для каждой катушки индуктивности отдельно, что позволяет без затруднений адаптировать программу под другой радар.

В результате выполнения выпускной квалификационной работы был разработан прототип аппаратно-программной системы для помощи людям с ограниченными возможностями для ориентации в пространстве. В последующих версиях АПС возможно предусмотреть обученную на распознавание движущихся объектов нейронную сеть, которая будет активировать ряд катушек по вектору движения цели, а пик частоты будет иметь катушка, максимально близко описывающая расстояние до текущего месторасположения объекта.

Список литературы

1. Электронный поводыр для слепых «Электросонар» [Электронный ресурс]. - URL: <https://www.pvsm.ru/e-lektronika/30094> (Дата обращения 21.03.2020).
2. Файзрахманов Р.Р., Зайнуллин Р.М. Прибор для реабилитации слепых и слабовидящих [Электронный ресурс] // Уфимский научно-исследовательский институт глазных болезней Академии наук Республики Башкортостан]. - URL: <https://eyepress.ru/article.aspx?22549> (Дата обращения 28.03.2020).
3. Ультразвуковой локатор для слепых разработан в России [Электронный ресурс]. - URL: https://www.cnews.ru/news/top/ultrazvukovoj_lokator_dlya_slepyh_razrabotan (Дата обращения 11.04.2020).
4. Зоткин Г. А. Поводыр для слепых [Электронный ресурс]. - URL: <https://findpatent.ru/patent/181/1811836.html> (Дата обращения 21.03.2019).
5. Крышкин Б.А. Устройство для обнаружения препятствий слепыми [Электронный ресурс]. - URL: <https://findpatent.ru/patent/206/2061446.html> (Дата обращения 11.04.2020).
6. Сапрыкин В.А. Устройство акустического представления пространственной информации для инвалидов по зрению [Электронный ресурс]. - URL: <https://findpatent.ru/patent/206/2060028.html> (Дата обращения 13.04.2020).
7. Жуков К. А. Способ и система точной локализации слабовидящего или слепого человека [Электронный ресурс]. - URL: <https://findpatent.ru/patent/268/2681346.html> (Дата обращения 13.04.2020).
8. Хоменок А.Т. Трость для инвалида по зрению [Электронный ресурс]. - URL: <https://findpatent.ru/patent/228/2280429.html> (Дата обращения 13.04.2020).
9. Ультразвуковой локатор для слепых [Электронный ресурс]. - URL: <https://findpatent.ru/patent/204/2040234.html> (Дата обращения 13.04.2020).
10. Статья на онлайн ресурсе «Фонтанка.ру» // Санкт-Петербург онлайн [Электронный ресурс]. - URL: <https://www.fontanka.ru/2018/11/26/054/> (Дата обращения 13.04.2020).
11. Пухальский, Г.И. Проектирование микропроцессорных систем: учебное пособие для вузов // СПб.: Политехника, 2001. — 544 с.
12. Gregory L. Charvat. Small and Short-Range Radar Systems // CRC Press, 427.
13. Нефедов В.И. Основы Радиоэлектроники и связи // Москва, Высшая школа, 2009.
14. Бьерн Страуструп. Язык программирования C++. Специальное издание [Электронный ресурс]. - URL:

https://codernet.ru/books/c_plus/bern_straustrop_yazyk_programmirovaniya_c_specialnoe_izdanie/ (Дата обращения 13.04.2020).

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						38
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А. Программа обработки данных

```
#!/usr/bin/python3
import os, sys
import subprocess
import ctypes
import sys
import serial
import colorama
from colorama import Fore, Back, Style
import pyautogui
import time

def searchPort():
    if sys.platform.startswith('win'):
        ports = ['COM%s' % (i + 1) for i in range(256)]
    elif sys.platform.startswith('linux') or sys.platform.startswith('cygwin'):
        ports = glob.glob('/dev/tty[A-Za-z]*')
    elif sys.platform.startswith('darwin'):
        ports = glob.glob('/dev/tty.*')
    else:
        raise EnvironmentError('Unsupported platform')
    result = []
    for port in ports:
        try:
            s = serial.Serial(port)
            s.close()
            result.append(port)
        except (OSError, serial.SerialException):
            pass
    return result

def sendFreq(ser, line1, line2, line3, line4):
    data = "<start>a{11}b{12}c{13}d{14}<stop>".format(11=line1, 12=line2, 13=line3,
14=line4)
    ser.write(b'%s' % data)
    return

colorama.init() # Maintenance color output on Windows 10

dir_autoRadar = r'c:/Users/bol/Desktop/AutoRadar/' # Base directory AutoRadar
exe_AutoRadarControl = 'AutoRadarControl.exe' # Target executable application
dir_ResultsFiles = 'ResultsFiles/'
```

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		39

```

hwnd = ctypes.windll.user32.GetForegroundWindow() # Saving the console
window handle
ctypes.windll.user32.SetWindowPos(hwnd, -1, 1300, 300, 600, 700, 0x0040) #
Setting the size and position of the script console

print('Changing directory on', Fore.YELLOW + dir_autoRadar, end="")
if (os.path.exists(dir_autoRadar) and os.path.isdir(dir_autoRadar)):
    os.chdir(dir_autoRadar)
    print(Fore.GREEN + ' OK', Style.RESET_ALL)
else:
    print(Fore.RED + '\nERROR: path to the target directory is incorrect',
Style.RESET_ALL)
    input()

print('Run', Fore.YELLOW + exe_AutoRadarControl, Style.RESET_ALL + 'as
subprocess...', end="")
if (not os.path.isfile(dir_autoRadar + exe_AutoRadarControl)):
    print(Fore.RED + '\nERROR: path to the target application is incorrect',
Style.RESET_ALL)
    input()
process = subprocess.Popen(dir_autoRadar + exe_AutoRadarControl)
if (process.poll() == None):
    print(Fore.GREEN + ' OK', Style.RESET_ALL)
else:
    print(Fore.RED + '\nERROR: failed to initialize process [' +
exe_AutoRadarControl + ']', Style.RESET_ALL)
    input()

# Cleaning logs directory
logs = [os.path.join(os.path.join(dir_autoRadar, dir_ResultsFiles), file) for file in
next(os.walk(dir_autoRadar+dir_ResultsFiles))[2]] # Getting a list of logs
for log in logs:
    os.remove(os.path.abspath(log))

time.sleep(1) # Waiting to start application
pyautogui.click(150, 90) # Start reading data from the radar
pyautogui.moveTo(960, 540)

print('Checking', Fore.YELLOW + dir_ResultsFiles, Style.RESET_ALL +
'directory...', end="")
if (os.path.exists(dir_autoRadar + dir_ResultsFiles) and os.path.isdir(dir_autoRadar
+ dir_ResultsFiles)):

```

					БКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		40

```

os.chdir(dir_autoRadar + dir_ResultsFiles)
print(Fore.GREEN + '          OK', Style.RESET_ALL)
else:
    print(Fore.RED + '\nERROR: path to the ResultsFiles is incorrect',
Style.RESET_ALL)

logs = [os.path.join(os.path.join(dir_autoRadar, dir_ResultsFiles), file) for file in
next(os.walk(dir_autoRadar+dir_ResultsFiles))[2]] # Getting a list of logs

print(logs[0])

print('Open log file...')
#logg= os.path.abspath(dir_autoRadar+"log.txt")
logg= os.path.abspath(logs[1])

handle_log = os.open(logg, os.O_RDWR) # os.O_NONBLOCK
dir_autoRadar+"log.txt" os.O_RDWR os.O_RDONLY

# Initial values
port = searchPort()
serialSpeed = 9600
total = b"
if (port):
    serialWrite = serial.Serial(port, serialSpeed, dsrdtr = 1, timeout = 0)
coil_line_amplitude_0 = 0
coil_line_amplitude_1 = 0
coil_line_amplitude_2 = 0
coil_line_amplitude_3 = 0

while (True): # Continuous log parsing
    unit = os.read(handle_log, 1024) # Primary reading of a log
    if (len(unit)==0): # If there's nothing more to read
        continue
    str_total=str(unit).replace("b", "") # Convert bytes to a string with removing the
prefix
    str_total = str_total.split("\\n")
    for line in str_total: # String parsing
        try:
            tuple_line=line.split()
            if (float(tuple_line[2])>0):
                dist = float(tuple_line[2])*0.527 # Getting distance to the target
                # Defining a target in one of the groups
                if (dist <=1 ):

```

					БКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		41

```

        coil_line_amplitude_0 += 1
    elif (dist > 1 and dist <= 2):
        coil_line_amplitude_1 += 1
    elif (dist > 2 and dist <= 3):
        coil_line_amplitude_2 +=1
    elif (dist > 3):
        coil_line_amplitude_3 +=1
except ValueError: # Catching string-to-integer type conversion errors
    pass
except IndexError: # Catching index access errors
    pass
if (coil_line_amplitude_0 != 0 or coil_line_amplitude_1 != 0
    or coil_line_amplitude_2 != 0 or coil_line_amplitude_3 != 0):
    print("line_0:      ", coil_line_amplitude_0, "      line_1:      ",
coil_line_amplitude_1, "      line_2:      ", coil_line_amplitude_2, "      line_3:      ",
coil_line_amplitude_3)
    if (port):
        sendFreq(serialWrite, coil_line_amplitude_0, coil_line_amplitude_1,
coil_line_amplitude_2, coil_line_amplitude_3)
        coil_line_amplitude_0=0
        coil_line_amplitude_1=0
        coil_line_amplitude_2=0
        coil_line_amplitude_3=0

code = process.wait()

```

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						42
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение Б. Программа для Arduino Mega 2560

```
#include <StaticThreadController.h>
#include <Thread.h>
#include <ThreadController.h>
```

```
const int coil1 = 22;
const int coil2 = 24;
const int coil3 = 26;
const int coil4 = 28;
const int coil5 = 30;
const int coil6 = 32;
const int coil7 = 34;
const int coil8 = 36;
const int coil9 = 38;
const int coil10 = 40;
const int coil11 = 42;
const int coil12 = 44;
const int coil13 = 46;
const int coil14 = 48;
const int coil15 = 50;
const int coil16 = 52;
```

```
Thread coilThread1 = Thread();
Thread coilThread2 = Thread();
Thread coilThread3 = Thread();
Thread coilThread4 = Thread();
Thread coilThread5 = Thread();
Thread coilThread6 = Thread();
Thread coilThread7 = Thread();
Thread coilThread8 = Thread();
Thread coilThread9 = Thread();
Thread coilThread10 = Thread();
Thread coilThread11 = Thread();
Thread coilThread12 = Thread();
Thread coilThread13 = Thread();
Thread coilThread14 = Thread();
Thread coilThread15 = Thread();
Thread coilThread16 = Thread();
```

```
String startBit;
String stopBit;
String stringData;
```

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						43
Изм.	Лист	№ докум.	Подпись	Дата		

```

int statusStartBit;
int statusStopBit;
int lengthData;
boolean availablePacket;

```

```

int indexLine1;
int indexLine2;
int indexLine3;
int indexLine4;
int freqLine1;
int freqLine2;
int freqLine3;
int freqLine4;

```

```
// Initial protocol initialization
```

```

void configuration()
{
    startBit = "<start>";
    stopBit = "<stop>";
    stringData.reserve(64);
    allReset();
}

```

```

void allReset()
{
    stringData = "";
    reset();
}

```

```

void reset()
{
    statusStartBit = 0;
    statusStopBit = 0;
    lengthData = 0;
    availablePacket = false;
}

```

```
// Called after loop () - if there is any data in the serial port buffer.
```

```

void serialEvent()
{
    readSerial();
    if(availablePacket)
    {

```

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		44

```

    setFreq();
    allReset();
}
}

void readSerial()
{
    while(Serial.available() && !availablePacket)
    {
        int bufferChar = Serial.read();
        if(statusStartBit < startBit.length())
        {
            if(startBit[statusStartBit] == bufferChar)
                statusStartBit++;
            else
                allReset();
        }
        else
        {
            if(lengthData <= 0)
                lengthData = bufferChar;
            else
            {
                if(lengthData > stringData.length())
                {
                    stringData += (char)bufferChar;
                }
                else
                {
                    if(statusStopBit < stopBit.length())
                    {
                        if(stopBit[statusStopBit] == bufferChar)
                        {
                            statusStopBit++;
                            if(statusStopBit == stopBit.length())
                            {
                                reset();
                                availablePacket = true;
                            }
                        }
                    }
                    else
                        allReset();
                }
            }
        }
    }
}

```

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		45


```

    }
    }
    }
    }
}

void setFreq()
{
    indexLine1 = stringData.indexOf("a") + 1;
    indexLine2 = stringData.indexOf("b") + 1;
    indexLine3 = stringData.indexOf("c") + 1;
    indexLine4 = stringData.indexOf("d") + 1;
    freqLine1 = stringData.charAt(indexLine1);
    freqLine2 = stringData.charAt(indexLine2);
    freqLine3 = stringData.charAt(indexLine3);
    freqLine4 = stringData.charAt(indexLine4);

    //frequency setting for calling functions
    coilThread1.setInterval(freqLine1); coilThread2.setInterval(freqLine1);
    coilThread3.setInterval(freqLine1); coilThread4.setInterval(freqLine1);
    coilThread5.setInterval(freqLine2); coilThread6.setInterval(freqLine2);
    coilThread7.setInterval(freqLine2); coilThread8.setInterval(freqLine2);
    coilThread9.setInterval(freqLine3); coilThread10.setInterval(freqLine3);
    coilThread11.setInterval(freqLine3); coilThread12.setInterval(freqLine3);
    coilThread13.setInterval(freqLine4); coilThread14.setInterval(freqLine4);
    coilThread15.setInterval(freqLine4); coilThread16.setInterval(freqLine4);
}

void setup() {
    Serial.begin(9600);
    configuration();
    //set pin's as output
    pinMode(coil1, OUTPUT); pinMode(coil3, OUTPUT);
    pinMode(coil2, OUTPUT); pinMode(coil4, OUTPUT);
    pinMode(coil4, OUTPUT); pinMode(coil5, OUTPUT);
    pinMode(coil6, OUTPUT); pinMode(coil7, OUTPUT);
    pinMode(coil8, OUTPUT); pinMode(coil9, OUTPUT);
    pinMode(coil10, OUTPUT); pinMode(coil11, OUTPUT);
    pinMode(coil12, OUTPUT); pinMode(coil13, OUTPUT);
    pinMode(coil14, OUTPUT); pinMode(coil15, OUTPUT);

    //bind functions to threads
    coilThread1.onRun(coilProcessing1); coilThread2.onRun(coilProcessing2);

```

```

coilThread3.onRun(coilProcessing3); coilThread4.onRun(coilProcessing4);
coilThread5.onRun(coilProcessing5); coilThread6.onRun(coilProcessing6);
coilThread7.onRun(coilProcessing7); coilThread8.onRun(coilProcessing8);
coilThread9.onRun(coilProcessing9); coilThread10.onRun(coilProcessing10);
coilThread11.onRun(coilProcessing11); coilThread12.onRun(coilProcessing12);
coilThread13.onRun(coilProcessing13); coilThread14.onRun(coilProcessing14);
coilThread15.onRun(coilProcessing15); coilThread16.onRun(coilProcessing16);
}

void loop() {
    frequency_matching();
}

void coilProcessing1() {
    static bool coilStatus1 = false;
    coilStatus1 = !coilStatus1;
    digitalWrite(coil1, coilStatus1);
}

void coilProcessing2() {
    static bool coilStatus2 = false;
    coilStatus2 = !coilStatus2;
    digitalWrite(coil2, coilStatus2);
}

void coilProcessing3() {
    static bool coilStatus3 = false;
    coilStatus3 = !coilStatus3;
    digitalWrite(coil3, coilStatus3);
}

void coilProcessing4() {
    static bool coilStatus4 = false;
    coilStatus4 = !coilStatus4;
    digitalWrite(coil4, coilStatus4);
}

void coilProcessing5() {
    static bool coilStatus5 = false;
    coilStatus5 = !coilStatus5;
    digitalWrite(coil5, coilStatus5);
}

```

					BKP-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						47
Изм.	Лист	№ докум.	Подпись	Дата		

```

void coilProcessing6() {
    static bool coilStatus6 = false;
    coilStatus6 = !coilStatus6;
    digitalWrite(coil6, coilStatus6);
}

void coilProcessing7() {
    static bool coilStatus7 = false;
    coilStatus7 = !coilStatus7;
    digitalWrite(coil7, coilStatus7);
}

void coilProcessing8() {
    static bool coilStatus8 = false;
    coilStatus8 = !coilStatus8;
    digitalWrite(coil8, coilStatus8);
}

void coilProcessing9() {
    static bool coilStatus9 = false;
    coilStatus9 = !coilStatus9;
    digitalWrite(coil9, coilStatus9);
}

void coilProcessing10() {
    static bool coilStatus10 = false;
    coilStatus10 = !coilStatus10;
    digitalWrite(coil10, coilStatus10);
}

void coilProcessing11() {
    static bool coilStatus11 = false;
    coilStatus11 = !coilStatus11;
    digitalWrite(coil11, coilStatus11);
}

void coilProcessing12() {
    static bool coilStatus12 = false;
    coilStatus12 = !coilStatus12;
    digitalWrite(coil12, coilStatus12);
}

void coilProcessing13() {

```

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		48

```

static bool coilStatus13 = false;
coilStatus13 = !coilStatus13;
digitalWrite(coil13, coilStatus13);
}

void coilProcessing14() {
static bool coilStatus14 = false;
coilStatus14 = !coilStatus14;
digitalWrite(coil14, coilStatus14);
}

void coilProcessing15() {
static bool coilStatus15 = false;
coilStatus15 = !coilStatus15;
digitalWrite(coil15, coilStatus15);
}

void coilProcessing16() {
static bool coilStatus16 = false;
coilStatus16 = !coilStatus16;
digitalWrite(coil16, coilStatus16);
}

void frequency_matching(){
if (coilThread1.shouldRun())
coilThread1.run();
if (coilThread2.shouldRun())
coilThread2.run();
if (coilThread3.shouldRun())
coilThread3.run();
if (coilThread4.shouldRun())
coilThread4.run();
if (coilThread5.shouldRun())
coilThread5.run();
if (coilThread6.shouldRun())
coilThread6.run();
if (coilThread7.shouldRun())
coilThread7.run();
if (coilThread8.shouldRun())
coilThread8.run();
if (coilThread9.shouldRun())
coilThread9.run();
if (coilThread10.shouldRun())

```

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
Изм.	Лист	№ докум.	Подпись	Дата		49

```

        coilThread10.run();
    if (coilThread11.shouldRun())
        coilThread11.run();
    if (coilThread12.shouldRun())
        coilThread12.run();
    if (coilThread13.shouldRun())
        coilThread13.run();
    if (coilThread14.shouldRun())
        coilThread14.run();
    if (coilThread15.shouldRun())
        coilThread15.run();
    if (coilThread16.shouldRun())
        coilThread16.run();
}

```

					ВКР-НГТУ-09.03.01-(16-В-1)-001-2020(ПЗ)	Лист
						50
Изм.	Лист	№ докум.	Подпись	Дата		