Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное   
образовательное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

ИНСТИТУТ ЭЛЕКТРОННЫХ И ИНФОРМАЦИОННЫХ СИСТЕМ

КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

Разработка драйвера устройства при помощи arduino

Курсовой проект по учебной дисциплине «Системное программирование»

по специальности 09.03.01 Информатика и вычислительная техника  
ИЭИС.КП 3094(ЗУ).01ПЗ

|  |  |
| --- | --- |
|  | Руководитель  / И. Ю. Кулаков  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |
|  | Студент группы 3094 (ЗУ)  / В.В. Бурбах  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

Содержание

[Введение 3](#_Toc166158403)

[1 Постановка задачи 4](#_Toc166158404)

[1.1 Обоснование необходимости разработки 4](#_Toc166158405)

[1.2 Технико-математическое описание задачи 4](#_Toc166158406)

[1.3 Характеристика бизнес-процесса 4](#_Toc166158407)

[1.4 Анализ предметной области 5](#_Toc166158408)

[1.5 Требования к программе 6](#_Toc166158409)

[1.6 Обоснование проектных решений 6](#_Toc166158410)

[2 Теоретическая часть 9](#_Toc166158411)

[2.1 История научных исследований по выбранной теме 9](#_Toc166158412)

[2.2 Определения и анализ ключевых терминов 10](#_Toc166158413)

[2.3 Актуальные взгляды на выбранную тему 11](#_Toc166158414)

[3 Практическая часть 13](#_Toc166158415)

[3.1 Анализ задачи 13](#_Toc166158416)

[3.2 Описание логической структуры 13](#_Toc166158417)

[3.3 Разработка программы для работы 14](#_Toc166158418)

[3.4 Сравнительная характеристика работы 16](#_Toc166158419)

[Заключение 18](#_Toc166158420)

[Список литературы 19](#_Toc166158421)

[Приложение А 20](#_Toc166158422)

[Приложение Б 23](#_Toc166158423)

[Приложение В 24](#_Toc166158424)

# Введение

Драйвер устройства – это компьютерная программа, управляющая строго определенным типом устройства, подключенным к или входящим в состав компьютера. Основная задача любого драйвера – это предоставление программного интерфейса для управления устройством, с помощью которого операционная система и другие компьютерные программы получают доступ к функциям данного устройства, не зная, как конкретно оно используется и работает. Разработчики могут писать с помощью драйверов высокоуровневые приложения и программы, не вдаваясь в подробности низкоуровневого функционала каждого из необходимых устройств в отдельности.

В рамках курсового проектирования была взята тема по разработке драйвера, который позволяет управлять компьютером при помощи джойстика от игровой консоли Atari 2600.

Приложение позволяет подключаться к игровому контроллеру, который, в свою очередь, подключен к компьютеру через микроконтроллер Arduino. Также приложение дает возможность самостоятельно устанавливать клавиши или сочетание клавиш на определенные нажатия джойстика.

Цель разработки:

* предоставить возможность использовать игровой контроллер в развлекательных целях на персональном компьютере;
* обеспечить возможность упрощать работу за компьютером, путем задания клавиш или комбинаций клавиш на нажатие кнопок контроллера.

Объектом разработки является программный комплекс, позволяющий использовать игровой контроллер в различных целях на персональном компьютере.

Задачами курсового проектирования в связи с указанной целью являются:

* разработать программное обеспечение, которое смогло бы улавливать сигналы, поступающие от джойстика и передавать их на компьютер;
* разработать приложение, которое будет считывать сигналы устройства и на основе сигналов эмулировать те или иные нажатия клавиш;
* создать механизм установки нажатия клавиши или комбинаций клавиш в ответ на определенные действия контроллера.

# 1 [Постановка задачи](file:///C:\Users\Student\Downloads\Постановка%20задачи.docx#_Toc452730688)

# 1.1 Обоснование необходимости разработки

В соответствии с заданием на курсовую работу, требуется разработать драйвер для джойстика от игровой консоли Atari 2600, который подключен к плате Arduino. Также необходимо разработать программу, которая позволит использовать устройство как в развлекательных целях, так и для упрощения выполнения каких-либо действий, путем задания ему команд, которые соответствуют нажатию клавиш или комбинации клавиш на персональном компьютере. Разработка данного программного комплекса обусловлена необходимостью использовать джойстик в личных целях, но при этом на рынке нет готового решения проблемы.

# 1.2 Технико-математическое описание задачи

В качестве устройства, для которого необходимо разработать драйвер, выступает джойстик с разъемом 9pin от игровой консоли Atari 2600. Для считывания сигнала джойстика был выбран микроконтроллер Arduino UNO R3, программное обеспечение для микроконтроллера написано на языке программирования Arduino. Этот микроконтроллер не способен самостоятельно эмулировать нажатие клавиш, но при помощи него можно передавать по последовательному порту сигнал о том, какая была клавиша нажата, и обрабатывать этот сигнал при помощи другого приложения. Второе приложение должно быть графическим, чтобы пользователь мог самостоятельно устанавливать клавишу или комбинацию клавиш, нажатие которых будет эмулироваться при работе с игровым устройством.

# 1.3 Анализ предметной области

Предметная область данного проекта включает в себя следующие аспекты:

* индикатор текущей даты и времени: важный элемент интерфейса, который позволяет пользователю видеть актуальную информацию о времени на компьютере. Здесь следует рассмотреть способы отображения времени, форматы даты и времени, а также возможность настройки пользователем;
* статистика ПК: анализ и отображение информации о работе компьютера, такой как загрузка процессора, использование памяти, сетевая активность и другие характеристики. Эта информация может быть полезна для пользователей, чтобы контролировать состояние своего устройства;
* интерфейс и визуализация: разработка удобного и интуитивно понятного пользовательского интерфейса для отображения текущей даты и времени, а также статистики ПК. Важно уделить внимание дизайну и удобству использования;
* технические аспекты: выбор языка программирования, технологий для реализации проекта, методы сбора статистики о ПК, обеспечение точности отображения времени и данных;
* функциональные возможности: определение основных функций, которые должны быть реализованы в проекте, такие как обновление времени в реальном времени, отображение графиков статистики, настройки внешнего вида и т.д.

Анализ предметной области поможет определить основные требования к проекту, его цели и функционал, что будет ключевым в дальнейшей разработке курсового проекта.

# 1.4 Требования к программе

Для работы драйвера игрового устройства, а также сопутствующему ему графического приложения необходимы следующие минимальные требования к персональному компьютеру:

* процессор: Intel Core i3 или аналогичный;
* оперативная память: 4 ГБ;
* дисковое пространство: 100 МБ на жестком диске;
* графическая карта: поддерживающая разрешение экрана не менее 1280x720 пикселей;
* операционная система: Windows 10/MacOS/Ubuntu;
* USB-порт для подключения к микроконтроллеру по последовательному порту.

Соблюдение этих минимальных требований обеспечит нормальное функционирование программы.

# 1.5 Обоснование проектных решений

1.5.1 Обоснование выбора языков программирования

Оптимальным выбором для реализации данного курсового проекта являются языки программирования Arduino и Python.

Язык программирования Arduino необходим для работы с микроконтроллерами одноименного семейства.

Для разработки графического приложения был выбран язык программирования Python по следующим причинам:

* Python очень прост в освоении и позволяет быстро создавать небольшие графические приложения, для выполнения простых, с точки зрения вычислительных затрат, задач;
* Python обладает большим количеством библиотек, написанных на низкоуровневом языке программирования Си, что позволяет выполнять некоторые поставленные задачи на уровне «железа».

1.5.2 Инструментальные средства

Для получения сигнала от игрового контроллера использовался микроконтроллер Arduino UNO R3. Arduino UNO — это плата микроконтроллера на базе ATmega328P. Он имеет 14 цифровых входов/выходов, 6 аналоговых входов, керамический резонатор 16 МГц, USB-соединение, разъем питания, разъем ICSP и кнопку сброса.

В качестве сред разработки в рамках курсового проекта используется Visual Studio Code и Arduino IDE. При помощи Visual Studio Code написано графическое приложение на Python. Код на языке программирования Arduino написан в среде программирования Arduino IDE.

Для написания графического интерфейса используется Custom Tkinter, который представляет собой Python библиотеку на базе стандартной библиотеки графического интерфейса Tkinter, но с расширенными возможностями для дизайна.

Для чтения последовательного порта, на который передает сигнал Arduino UNO, используется библиотека pyserial, с ее помощью можно читать, а также отправлять сообщения по COM порту.

Для симуляции нажатия клавиш используется Python библиотека pyautogui, в ней представлена возможность как простого нажатия клавиш, так и их зажатия.

1.5.3 Обоснование выбора среды программирования

В качестве среды программирования на Python выбрана Visual Studio Code. Из основных преимуществ данного редактора кода можно выделить следующие:

* множество настроек (как всей программы, так и интерфейса);
* расширяемая библиотека дополнений и готовых решений;
* простота и гибкость, Visual Studio Code очень прост в освоении и не требователен к системе.

Программирование на языке Arduino будет производиться в среде Arduino IDE, которая обладает всем необходимым набором инструментов для написания программ под плату Arduino.

1.5.4 Информационное обеспечение

Для написания пояснительной записки к курсовому проекту используется текстовый редактор MS Word. В качестве приложения для создания презентации на защиту курсового проекта используется MS PowerPoint.

1.6 Обзор и анализ аналогов существующих программных систем

Готовых коммерческих решений для подобной реализации нет, в основном это небольшие личные проекты конструкторов-энтузиастов. Однако можно отметить тот факт, что существуют переходники с разъема 9pin на Micro USB, но без дополнительного программного обеспечения считывать сигналы контроллера будет невозможно, так как игровые контроллеры с подобным разъемом имеют разное расположение контактов.

1.7 Выводы предпроектного исследования

Целью данного курсового проекта является возможность использования игрового джойстика как в развлекательных целях, так и в целях упрощения работы с компьютером. Работу над проектом можно разбить на несколько этапов:

1. Подключить игровой контроллер к Arduino.
2. Разработать код для считывания сигнала с джойстика на микроконтроллер.
3. Спроектировать графическое приложение.
4. Разработать графический интерфейс.
5. Написать код считывания сообщений последовательного порта.
6. Протестировать разработанную систему.

# 2 Теоретическая часть

# 2.1 История научных исследований по выбранной теме

Первые игровые геймпады проявились в 1972 году, тогда на свет вышла первая в мире коммерческая игровая приставка Magnavox Odyssey. В то время разработчики еще не особо не задумывались об эргономике и удобстве использования, был важен лишь функционал, именно эти взгляды и породили первый в истории геймпад. Поскольку в тех играх максимум что позволялось - перемещать игровой объект по полю, и именно поэтому было всего два "барашка", отвечающих за горизонтальное и вертикальное перемещение, а также центральная кнопка, которая просто перезагружала зависшую игру.

С тех пор началось активное развитие геймпадов, этот период можно условно разделить на 8 поколений, от простейших игровых контроллеров, вроде той, что была у приставки Magnavox Odyssey, до современных геймпадов от компаний Microsoft, Sony и Nintendo, которые являются по большей части беспроводными, позволяют регулировать усилие нажатий, обладают отдачей в виде вибраций, а также способны воспроизводить различные звуки.

Если говорить про игровой контроллер, который используется в рамках курсового проекта, то он использовался на игровых консолях Atari 2600. Сама игровая консоль была выпущена в 1977 как Atari Video Computer System, а в 1982 году переименована в Atari 2600, она была продана в количестве более 30 миллионов экземпляров и создала новый рынок, который существует и на сегодняшний день. До появления 2600-й большинство видеоприставок были либо игровыми автоматами, например в барах, либо устройствами с фиксированными функциями, ограниченными несколькими встроенными играми, такими как Pong. Первая домашняя система Atari стала началом новой эры. Сам контроллер на тот период времени являлся одним из самых лучших.

# 2.2 Определения и анализ ключевых терминов

Драйвер игрового контроллера — это удобный инструмент, который позволяет использовать персональный компьютер как в развлекательных целях, так и для упрощения работы с ним путем задания конкретных клавиш или их комбинации на нажатие кнопок джойстика.

Ключевые термины:

* джойстик – устройство ввода информации в персональный компьютер, которое представляет собой качающуюся в двух плоскостях вертикальную ручку, в данном курсовом проекте геймпад, игровой контроллер и игровое устройство и джойстик применяются как синонимы;
* микроконтроллер – это однокристальный компьютер, предназначенный для управления электронными устройствами, способный выполнять относительно простые задачи, а также имеет интегрированные устройства ввода-вывода, таймер и другие периферийные устройства.
* графический интерфейс – система взаимодействия, предоставляющая пользователям доступ к системным объектам и инструментам управления через визуализацию информации – через графические элементы на мониторе;
* настольное приложение – это программа, обрабатываемая на стороне клиента и запускаемая в виде обыкновенного исполняемого файла на устройстве пользователя;
* последовательный порт – название интерфейса стандарта RS-232 в IBM PC. Порт называется «последовательным», так как информация через него передаётся по одному биту, последовательно бит за битом.

Анализ ключевых терминов:

* при помощи микроконтроллера можно отслеживать нажатие кнопок на джойстике и отправлять на последовательный порт сообщение о том, какая именно кнопка была нажата;
* настольное приложение включает в себя графический интерфейс, также оно читает сообщение с последовательного порта и эмулирует нажатие клавиш в зависимости от того, какая клавиша соответствует конкретно нажатой кнопке в текущий момент времени.

# 2.3 Актуальные взгляды на выбранную тему

Интерес к написанию драйверов для игровых контроллеров уже много лет держится на умеренном уровне. Как правило подобное программное обеспечение пишут различные любители радиотехники и программисты, они выкладывают видео, пишут посты и статьи о своих наработках в сети интернет, где подобного рода контент просматривают такие же энтузиасты. В таких случаях у людей нет цели создать коммерческий продукт, они это делают только ради своих собственных интересов, и делятся этим с другими. Некоторые материалы действительно могут набрать высокую популярность, но это зачастую зависит от масштаба, сложности и особенности проекта.

# 3 Практическая часть

# 3.1 Анализ задачи

Для курсового проекта по разработке драйвера устройства при помощи Arduino необходимо выполнить следующие этапы:

* изучить особенности расположения контактов на джойстике;
* проанализировать требования к работе с микроконтроллером Arduino UNO R3;
* подключить джойстик к микроконтроллеру;
* разработать программный код для микроконтроллера для чтения сигналов от джойстика;
* разработать дизайн интерфейса с учетом всех требований;
* определить элементы управления, необходимые для отображения информации и взаимодействия с программой;
* разработать настольное приложение с графическим интерфейсом для подключения и настройки джойстика.

На рисунке А.1 представлена примерная схема подключения джойстика к плате Arduino. Также на рисунке Б.1 представлен графический интерфейс приложения в режиме «Стандарт», а на рисунке Б.2 представлен графический интерфейс приложения в режиме «Игра».

# 3.2 Описание логической структуры

3.2.1 Алгоритм программы

Алгоритмы программы для платы Arduino состоит в следующем: первым делом идет инициализация пинов как входных согласное схеме, представленной на рисунке А.1, далее в бесконечном цикле loop объявляется строковая переменная и идет последовательно считывание пинов, если сигнал есть, то к строке прибавляется значение, соответствующее пину: для третьего пина значение «0», для четвертого пина – «1», для пятого пина – «2», для шестого пина – «3», для седьмого пина – «4». При этом все значения отделяются точкой с запятой. В конце итерации цикла на COM-порт отправляется значение этой строки и происходит искусственная задержка на 100 миллисекунд для предотвращения дребезга контактов. Листинг кода для микроконтроллера Arduino представлен в приложении Г.1.

При запуске графического приложения открывается окно, представленное на рисунке Б.1. В окне представляется возможность выбрать порт, к которому подключено устройство, а также режим использования джойстика. По умолчанию установлен режим «Стандарт», при котором можно настроить различные комбинации клавиш при нажатии на одну из кнопок игрового контроллера. В режиме «Игра» можно настроить только одну клавишу на нажатие кнопок контроллера, при этом предусмотрено одновременное нажатие на несколько кнопок, это сделано в целях предоставления возможности движения по диагонали во время игры.

Основная сложность разработки заключалась как раз в использовании контроллера в развлекательных целях, для этого используется метод read\_serial\_game, алгоритм работы метода заключается в следующем: сначала инициализируется переменная ser, которая отвечает за чтение последовательного порта, далее в цикле проверяется, что входной буфер не пустой, если буфер пуст, то идет переход на следующую итерацию цикла, если же буфер не пуст, то идет чтение сообщения с последовательного порта, так как Arduino отправляет несколько нажатий одновременно, то команды записываются отдельно в список, в качестве разделителя команд выступает точка с запятой, в виду некоторых особенностей, в качестве последнего элемента списка он вставляет пустоту, поэтому предусмотрен момент, где последний элемент списка удаляется. Далее программа проходит по списку считанных команд. Следом идет проверка соответствует ли команда какому-либо ключу из словаря данных, где в качестве ключа хранится значение, которое отправляет Arduino, а в качестве значения хранит соответствующую клавишу, которую необходимо нажать. Если условие выполняется, то создается список клавиш, которые необходимо нажать и сразу же прожимаются соответствующие клавиши. Далее идет цикл, который выполняется до тех пор, пока пользователь не отпустил текущие кнопки на джойстике, внутри цикла происходит нажатие клавиш и чтение сообщения с последовательного порта, когда пользователь кнопки отпускает, то выполняется отжатие всех клавиш, соответствующих данным кнопкам, в конце цикла закрывается чтение последовательного порта. Также первоначально использовалась конструкция try except на случай, если по той или иной причине не получилось подключиться к последовательному порту и выводится соответствующее сообщение об ошибке. Стоит отметить, что метод вызывается в отдельном потоке, чтобы не прерывать возможность использовать графическое приложения для настроек. Блок-схема алгоритма метода представлена на рисунке В.1, а сам код представлен в приложении Г.2.

Подобным же образом работает метод считывания сигналов в режиме стандарт, за исключением того, что в методе не предусмотрено нажатие сразу нескольких кнопок, но программа берет первый прочитанный сигнал. Также в этом методе при одном нажатии кнопки можно привязать нажатие сразу нескольких клавиш последовательно, а также идет эмуляция кратковременного нажатия клавиш, а не их задержка.

3.2.2 Составные части программы и связи между ними

Программа содержит несколько модулей, каждый из которых выполняет определенные функции:

* программное обеспечение, написанное на Arduino и вшитое в плату Arduino UNO, данный код позволяет считывать сигналы от джойстика;
* модуль main запускает приложение;
* модуль joystick\_app содержит класс JoystickApp, в котором создается интерфейс и реализация эмулирования клавиш;
* модуль readline\_serial содержит класс Readline, который создан для чтения данный из последовательного порта, хотя это и может делать встроенный в библиотеку pyserial метод, но он делает это медленнее, чем написанный самостоятельно.

Взаимодействие между этими модулями происходит следующим образом: микроконтроллер считывает сигнал с джойстика и отправляет сообщение на последовательный порт. Модуль main создает экземпляр класса JoystickApp, тем самым создается графический интерфейс. В модуле joystick\_app происходит считывание сообщений, который отправляет плата Arduino, с последовательного порта, в процессе считывания вызывается метод readline класса Readline, который позволяет более оптимизировано считывать последовательный порт.

# 3.3 Разработка программы для работы

3.3.1 Входные и выходные данные

Входными данными для настольного приложения являются:

* порт, который использует плата Arduino;
* режим, в котором будут читаться сообщения с последовательного порта;
* клавиши или комбинация клавиш, нажатие которых будет эмулировать настольное приложение в зависимости от того, какое сообщение получено с последовательного порта;
* сообщение с последовательного порта;
* файл формата json с сохраненными привязками клавиш.

Выходными данными для настольного приложения является файл формата json, в который сохраняется привязка клавиш.

Входными данными в программе для платы Arduino являются сигналы с джойстика при нажатии на нем каких-либо кнопок.

В качестве выходных данных в программе для платы Arduino является сообщение на последовательный порт, в котором указано какие кнопки были нажаты.

3.3.2 Используемые методы

В курсового проекте используются следующие методы:

* create\_bind\_buttons, данный метод создает виджеты, в них указываются клавиши, которые необходимо эмулировать;
* bind\_key, метод срабатывает при попытке задать клавиши для эмулирования;
* set\_binding, метод меняет словарь, где указаны в качестве ключей команды, считываемые с последовательного порта, а в качестве значению указываются соответствующие клавиши или комбинации клавиш;
* combobox\_ports\_callback, метод срабатывает при изменении порта, к которому подключена плата Arduino и меняет переменную, в которой хранится название порта на выбранный;
* combobox\_mode\_callback, метод срабатывает на смену режима пользования джойстика, в зависимости от выбора останавливается один поток с методом и запускается другой поток с другим методом;
* get\_combination\_text, метод возвращает список клавиш как одну строку, а в качестве делителя представлен знак «+»;
* read\_serial\_standard, метод предназначен для пользования джойстика в качестве эмулирования клавиши или комбинации клавиш, выполняется в отдельном потоке;
* read\_serial\_game, метод предназначен для пользования джойстика в качестве игорового;
* readline, метод считывает сообщение с последовательного порта;
* create\_save\_load\_buttons, метод создает кнопку сохранения созданных привязок клавиш, а также кнопку их загрузки;
* save\_settings, метод срабатывает на нажатие кнопки на сохранение привязок клавиш в файл формата json;
* load\_settings, метод срабатывает на нажатие кнопки на загрузку привязок клавиш из файла формата json;
* start\_thread, метод запускает другой метод, переданный в качестве аргумента, в отдельном потоке.

3.3.3 Тестирование

Для тестирования метода get\_combination\_text был написан юнит тест. Была произведена проверка на составление комбинаций, а также проверка на пустой список клавиш. Все тесты выполнились успешно. Код юнит теста представлен в приложении Г.3, результат выполнения теста представлен на рисунке Д.1.

Было протестирована работоспособность джойстика во время его пользования в режиме «Игра», а именно движение персонажа в игре Krunker по диагонали вперед и влево. Данные тестирования представлены в таблице 1.

Таблица 1 – Тест-кейс № 1

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | | | TU01 |
| Заголовок | | | Проверка движения по диагонали вперед и влево при помощи джойстика |
| Предусловие | | | Джойстик подключен к компьютеру по последовательному порту COM 4. Запущено приложение В приложении выбран порт «COM 4», установлен режим «Игра».  При нажатии на копку «UP» срабатывает клавиша «w», при нажатии на кнопку «LEFT» срабатывает клавиша «a». Открыта ссылка в браузере по адресу «https://krunker.io/?game=FRA:cxwbj» |
| Тестовые данные | | | Нет |
| Ожидаемый результат | | | При зажатии одновременно кнопок «Вверх» и «Влево» на джойстике персонаж в игре движется по диагонали вверх и влево |
| **Шаг** | **Действие** | **Предполагаемый результат** | **Pass / Fail** |
| 1 | Одновременное зажатие кнопок «Вверх» и «Влево» на джойстике | Движение персонажа игры по диагонали вперед и влево | Pass |
| Постусловие | | | Движение персонажа игры по диагонали вперед и влево работает корректно |
| Результат | | | Pass |

В качестве следующего теста было протестировано нажатие на кнопку «FIRE» в режиме «Стандарт». При нажатии на кнопку срабатывает комбинация клавиш в виде ввода тексте «hello». Данные тестирования представлены в таблице 2.

Таблица 2 – Тест-кейс № 2

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | | | TU02 |
| Заголовок | | | Проверки ввода комбинации клавиш «hello» |
| Предусловие | | | Джойстик подключен к компьютеру по последовательному порту COM 4. Приложение запущено. В приложении выбран порт «COM 4», установлен режим «Стандарт».  При нажатии на копку «FIRE» срабатывает комбинация клавиш «h+e+l+l+o». Открыт текстовый файл test.txt |
| Тестовые данные | | | Нет |
| Ожидаемый результат | | | При кратковременном нажатии на кнопку «FIRE» на джойстике в тестовом файле выводится надпись «hello» |
| **Шаг** | **Действие** | **Предполагаемый результат** | **Pass / Fail** |
| 1 | Нажатие на кнопку «FIRE» на джойстике | Вывод в текстовом файле надписи «hello» | Pass |
| Постусловие | | | При кратковременном нажатии на кнопку «FIRE» на джойстике вывод надписи «hello» в текстовом файле работает корректно |
| Результат | | | Pass |

Была произведена проверка вывода сообщения об ошибке «Отказано в доступе» при попытке подключиться к джойстике по последовательному порту, который занят другим приложением. Данные тестирования представлены в таблице 3.

Таблица 3 – Тест-кейс № 3

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | | | TU03 |
| Заголовок | | | Проверка вывода ошибке о занятом последовательном порте. |
| Предусловие | | | Джойстик подключен к компьютеру по последовательному порту COM 4. Приложение запущено. В приложении выбран порт «COM 1», установлен режим «Стандарт». Запущено Arduino IDE, в качестве последовательного порта выбран порт COM 4. |
| Тестовые данные | | | Нет |
| Ожидаемый результат | | | При выборе в приложении порта «COM 4» выведется сообщение об ошибке «Отказано в доступе» |
| **Шаг** | **Действие** | **Предполагаемый результат** | **Pass / Fail** |
| 1 | Выбор из выпадающего списка доступных последовательных портов «COM 4» | Вывод сообщения об ошибке «Отказано в доступе» | Pass |
| Постусловие | | | Вывод сообщения об ошибке «Отказано в доступе» при выборе занятого последовательного порта работает корректно |
| Результат | | | Pass |

Произведена проверка сохранения настроек привязки клавиш. Данные тестирования представлены в таблице 4.

Таблица 4 – Тест-кейс № 4

|  |  |
| --- | --- |
| Идентификатор | TU04 |
| Заголовок | Проверка сохранения настроек привязки клавиш |
| Предусловие | Джойстик подключен к компьютеру по последовательному порту COM 4. Приложение запущено. В приложении выбран порт «COM 4», установлен режим «Стандарт». На кнопку «UP» привязана комбинация «t+e+s+t», на кнопку «DOWN» привязана комбинация «h+e+l+l+o» |
| Тестовые данные | Нет |

Окончание таблицы 4

|  |  |  |  |
| --- | --- | --- | --- |
| Ожидаемый результат | | | На компьютере сохраняется файл «test.json» |
| **Шаг** | **Действие** | **Предполагаемый результат** | **Pass / Fail** |
| 1 | Нажатие на кнопку «Сохранить результат» | Появляется диалоговое окно с выбором местоположения файла | Pass |
| 2 | Выбран путь «D:\test» | Открылось содержимое папки «D:\test» | Pass |
| 3 | Ввод названия файла «test.json» | На экране отображается название файла «test.json» | Pass |
| 4 | Нажать на кнопку «Сохранить» | Файл сохраняется, выводится сообщение об успешном сохранении настроек | Pass |
| Постусловие | | | На компьютере успешно сохраняется файл «test.json» с настройками привязки клавиш |
| Результат | | | Pass |

3.3.4 Эксплуатационные документы – руководство пользователя

3.3.4.1 Назначение программы

Целевой аудиторией приложения будут являться пользователи, изъявившие желание использовать джойстик от игровой консоли Atari 2600 в личных целях.

Также данное приложение будет полезно для изучения в образовательных организациях.

3.3.4.2 Условия выполнения

Для работы графического приложение необходимы следующие минимальные системные требования персонального компьютера:

* процессор: Intel Core i3 или аналогичный;
* оперативная память: 4 ГБ;
* дисковое пространство: 100 МБ на жестком диске;
* графическая карта: поддерживающая разрешение экрана не менее 1280x720 пикселей;
* операционная система: Windows 10/MacOS/Ubuntu;
* USB-порт для подключения к микроконтроллеру по последовательному порту.

Соблюдение этих минимальных требований обеспечит нормальное функционирование программы.

3.3.4.3 Выполнение программы

Перед запуском настольного приложения необходимо подключить плату Arduino к компьютеру через USB. Для запуска приложения необходимо открыть исполняемый файл. После запуска приложения появляется окно, в котором можно произвести следующие действия:

* выбор порта, к которому подключен джойстик;
* выбор режима пользования джойстика;
* выбор клавиш для привязки их к определенным нажатиям джойстика.

Если выбранный порт уже используется другая программа, то на экран выведется сообщение об ошибке «Отказано в доступе».

Для выбора привязки клавиш доступны 5 элементов. Для того, чтобы выбрать на какую кнопку на джойстике привязать нажатие клавиши, необходимо нажать на кнопку напротив соответствующей записи:

* «Привязка клавиши на UP», привязывает клавишу на нажатие «Вверх» на джойстике;
* «Привязка клавиши на DOWN», привязывает клавишу на нажатие «Вниз» на джойстике;
* «Привязка клавиши на LEFT», привязывает клавишу на нажатие «Влево» на джойстике;
* «Привязка клавиши на RIGHT», привязывает клавишу на нажатие «Вправо» на джойстике;
* «Привязка клавиши на SPACE», привязывает клавишу на нажатие «Огонь» на джойстике, это отдельная кнопка.

Когда будет нажата нужная кнопка, появится надпись «Нажмите клавишу для привязки к UP», вместо команды «UP» будет соответствующая команда, в зависимости от того, какую выбрал пользователь. После этого пользователь должен нажать на любую клавишу на клавиатуре, которую он хочет привязать на срабатывание.

Если необходимо использовать джойстик в развлекательных целях, то необходимо выбрать режим «Игра» и выбрать клавиши, после выбранных необходимых клавиш, пользователь заходит в игру и эксплуатирует джойстик как клавиатуру.

Если необходимо использовать джойстик для эмулирования комбинации клавиш, необходимо выбрать режим «Стандарт» и задать комбинации клавиш, прекратить задание комбинации можно, нажав на клавишу «Enter», после выбранных комбинаций, пользователь просто может эксплуатировать джойстик в своих целях.

По умолчанию джойстик находится в режиме «Стандарт», где никакие клавиши не привязаны и вместо них надпись «None». Если переключиться в режим «Игра», то в качестве привязанных клавиш по умолчанию используются:

* На команду «UP» – клавиша «up»;
* на команду «DOWN» – клавиша «down»;
* на команду «LEFT» – клавиша «left»;
* на команду «RIGHT» – клавиша «right»;
* на команду «SPACE» – клавиша «space»;

Также по умолчанию выбирается первый найденный доступный последовательный порт, но его, при желании, можно поменять на нужный из списка доступных.

Так как сама программа после перезапуска не сохраняет привязки, заданные пользователем, однако существует возможность их сохранения, для этого в окне доступны две кнопки «Сохранить настройки» и «Загрузить настройки». При нажатии на сохранение предоставляется возможность выбора пути к файлу и само название файла в формате JSON, после успешного сохранения на экран выводится соответствующая надпись, сообщение представлено на рисунке Б.3. При нажатии на кнопку загрузки настроек также открывается окно с выбором файла формата JSON, после выбора файла данные о привязках автоматически загрузятся в приложение и выведется соответствующее сообщение, которое представлено на рисунке Б.4.

# 3.4 Сравнительная характеристика работы

Аналоги:

* ClockTuner: Отображает текущее время, дату, статистику системы и многое другое в виде настраиваемого виджета;
* Rainmeter: Мощный инструмент для настройки рабочего стола, который позволяет создавать собственные скины, включая индикаторы даты и времени;
* Sidebar Diagnostics: Отображает подробную информацию о системе, включая дату и время, в виде боковой панели;
* Taskbar Time and Date: Простое приложение, которое добавляет индикатор даты и времени на панель задач;
* Date and Time Clock Widget: Виджет рабочего стола, который отображает текущее время и дату в различных стилях.

Общие недостатки:

* некоторые приложения могут быть несовместимы с определенными версиями операционной системы;
* могут отображать неточную информацию о дате и времени, если системные часы не настроены правильно;
* могут потреблять системные ресурсы, что может привести к замедлению работы компьютера.

# Заключение

В результате курсового проектирования были улучшены знания в области работы со специализированными программами, также повышен уровень знаний в области программирования. Проектирование прошло успешно, все функции были реализованы, а задача полностью выполнена.

На этапе формирования требований к программному продукту были сформулированы цели и задачи курсового проекта, которые в ходе работы были успешно реализованы с помощью программы IntelliJ IDEA. В приложении к пояснительной записке курсового проекта помещены разработанные блок-схемы алгоритмов, выполненные в соответствии со стандартами.

Исходя из всего вышеперечисленного можно сделать вывод, что курсовой проект выполнен в полном объеме.

# Список литературы

1. ГОСТ Р 2.105–2019 Единая система конструкторской документации
2. Шилдт Г. Программирование на C++ для начинающих, 2024 – 156 с
3. Страуструпп Б. Язык программирования C++, 2022 – 98 с
4. https://sharpened.ucoz.ru/lib/polnyj\_spravochnik\_po\_c-gerbert\_shildt-2006.pdf, [Дата обращения 04.04.2024]
5. <https://stackoverflow.com>, [Дата обращения 16.04.2024]

# Приложение А

(обязательно)

Изображение выглядит как текст, диаграмма, Электронная техника, схема

Автоматически созданное описание

Рисунок А.1 – Схема подключения джойстика к Arduino

# Приложение Б

(обязательно)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок Б.1 – Окно приложения в режиме «Стандарт»

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок Б.2 – Окно приложения в режиме «Игра»

Изображение выглядит как текст, снимок экрана, программное обеспечение, Операционная система

Автоматически созданное описание

Рисунок Б.3 – Сообщение об успешном сохранении настроек привязки клавиш

Изображение выглядит как текст, снимок экрана, мультимедиа, гаджет

Автоматически созданное описание

Рисунок Б.4 – Сообщение об успешной загрузке настроек привязки клавиш

# Приложение В

(обязательное)

Изображение выглядит как текст, диаграмма, Параллельный, снимок экрана

Автоматически созданное описание

Рисунок В.1 – Блок-схема метода чтения данных с джойстика в режиме «Игра»

Приложение Г

(обязательное)

Г.1 – Листинг кода программы для Arduino

#define BUTTON\_UP 3

#define BUTTON\_DOWN 4

#define BUTTON\_LEFT 5

#define BUTTON\_RIGHT 6

#define BUTTON\_FIRE 7

void setup() {

  Serial.begin(9600);

  // Настраиваем пины как входы

  pinMode(BUTTON\_UP, INPUT\_PULLUP);

  pinMode(BUTTON\_DOWN, INPUT\_PULLUP);

  pinMode(BUTTON\_LEFT, INPUT\_PULLUP);

  pinMode(BUTTON\_RIGHT, INPUT\_PULLUP);

  pinMode(BUTTON\_FIRE, INPUT\_PULLUP);

}

void loop() {

  String message = "";

  if (digitalRead(BUTTON\_UP) == LOW) {

    message += "1;";

  }

  if (digitalRead(BUTTON\_DOWN) == LOW) {

    message += "2;";

  }

  if (digitalRead(BUTTON\_LEFT) == LOW) {

    message += "3;";

  }

  if (digitalRead(BUTTON\_RIGHT) == LOW) {

    message += "4;";

  }

  if (digitalRead(BUTTON\_FIRE) == LOW) {

    message += "5;";

  }

  if (message.length() > 0) {

    Serial.println(message);

  }

  delay(100);

}

Г.2 – Листинг кода метода чтения сигналов джойстика в режиме «Игра»

def read\_serial\_game(self):

try:

ser = serial.Serial(self.\_\_selected\_port, BAUD\_RATE)

while self.\_\_is\_stop\_thread\_read\_serial\_game == False:

if ser.in\_waiting > 0:

line = Readline(ser).readline().decode('utf-8').strip()

commands = line.split(";")

if commands[len(commands)-1] == '':

commands = commands[0:len(commands)-1]

keys = []

for command in commands:

action = int(command)

if action in self.bindings:

keys.append(self.bindings[action])

pyautogui.keyDown(self.bindings[action])

while line == Readline(ser).readline().decode('utf-8').strip() and ser.in\_waiting > 0:

pass

for key in keys:

pyautogui.keyUp(key)

ser.close()

except serial.SerialException as e:

CTkMessagebox(title="Ошибка",

message=e,

icon="cancel",

option\_1="Ok")

Г.3 – Листинг кода тестировании метода get\_combination\_text

import unittest

from joystick\_app import JoystickApp

class TestGetCombinationText(unittest.TestCase):

def setUp(self):

self.joystick\_app = JoystickApp()

self.joystick\_app.bindings\_standard = {

1: ['f', 's', 'a'],

2: ['1', 'f', 'w'],

3: ['0', '1'],

4: ['5', '4', 't', 'r', '1'],

5: []

}

def test\_get\_combination\_text\_up(self):

self.assertEqual(self.joystick\_app.get\_combination\_text(1), "f+s+a")

def test\_get\_combination\_text\_down(self):

self.assertEqual(self.joystick\_app.get\_combination\_text(2), "1+f+w")

def test\_get\_combination\_text\_left(self):

self.assertEqual(self.joystick\_app.get\_combination\_text(3), "0+1")

def test\_get\_combination\_text\_right(self):

self.assertEqual(self.joystick\_app.get\_combination\_text(4), "5+4+t+r+1")

def test\_get\_combination\_text\_fire(self):

self.assertEqual(self.joystick\_app.get\_combination\_text(5), "None")

if \_\_name\_\_ == "\_\_main\_\_":

unittest.main()

Приложение Д

(обязательное)

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок Д.1 – Результат выполнения тестирования метода get\_combination\_text