Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

аппаратно-программного комплекса для проведения интеллектуальных игр

БГУИР КП 1-40 02 01 01 001 ПЗ

Студенты Козлова О.В.

Бардиян М.А.

Руководитель Понкратов А.А.

МИНСК 2016

Содержание

[Введение 2](#_Toc450636832)

[1. Обзор литературы 3](#_Toc450636833)

[1.1 Arduino 4](#_Toc450636834)

[1.2 Правила интеллектуальных игр 5](#_Toc450636835)

[1.3 Обзор аналогов 7](#_Toc450636836)

[2. Разработка структурной схемы 9](#_Toc450636837)

[4. Разработка принципиальной схемы 11](#_Toc450636838)

[4.2 Главный модуль 14](#_Toc450636839)

[4.3 Связь модулей 15](#_Toc450636840)

[4.4 Pin Change Interrupt 16](#_Toc450636841)

[5. Руководство пользователя 18](#_Toc450636842)

[Список литературы 20](#_Toc450636843)

[ПРИЛОЖЕНИЕ А 21](#_Toc450636844)

[ПРИЛОЖЕНИЕ Б 22](#_Toc450636845)

[ПРИЛОЖЕНИЕ В 23](#_Toc450636846)

[ПРИЛОЖЕНИЕ Г 24](#_Toc450636847)

Введение

В последнее время набирают популярность различные виды интеллектуальных игр: «Что? Где? Когда?», «Мелотрек», «Брейн-ринг», «Мозгобойня» и другие. Количество команд и интеллектуальных клубов увеличивается. Но для некоторых видов игр не обойтись без специальной системы, которая будет управлять ходом игры. С помощью данного устройства игра будет более зрелищная, динамичная и интересная. Практически из процесса игры исчезают споры между командами о первоочередности ответа, т.к. частота опроса устройства составляет всего одна сотая секунды.

Система для проведения и организации интеллектуальных игр «Брэйн-система» предназначена для использования в качестве автоматизированного устройства для определения первоочередности ответа игроков/команд в играх «Брэйн-ринг», «Своя игра», «Эрудит-квартет» и других известных викторинах «вопрос-ответ». Также может использоваться в качестве учебно-методического пособия для обучения детей и подростков в игровой форме в различных учреждениях дошкольного и школьного воспитания. Заложенный в программную часть устройства алгоритм игры, основан на общепринятых правилах интеллектуальных игр и позволяет точно определять первоочередность нажатия кнопки с блокированием кнопок остальных участников.

*Постановка задачи*

Целью данного курсового проекта является разработка аппаратно-программного комплекса для проведения интеллектуальных игр:

* Брейн-ринг
* Эрудит-квартет
* Что? Где? Когда?
* Своя игра
* Тройка

Система должна иметь четыре игровых пульта, блок управления, звуковое оповещение и возможность выбора режима игры.

Брейн-система должна подавать звуковой сигнал, означающий начало времени вопросного раунда и сигнал, означающий окончание времени вопросного раунда, а также должна сигнализировать ведущему о готовности игрока дать ответ. В случае когда одна из команд просигнализировала о готовности дать ответ, брейн-система должна блокировать сигналы от другой команды (или от всех других команд) до соответствующего действия ведущего.

Брейн-система должна давать ведущему или его ассистенту чёткую возможность определить, была ли кнопка нажата в отведённое для ответов время, до его начала или после его окончания.

1. Обзор литературы

***1.1 Arduino***

Arduino — торговая марка аппаратно-программных средств для построения простых систем автоматики и робототехники, ориентированная на непрофессиональных пользователей. Программная часть состоит из бесплатной программной оболочки (IDE) для написания программ, их компиляции и программирования аппаратуры. Аппаратная часть представляет собой набор смонтированных печатных плат, продающихся как официальным производителем, так и сторонними производителями. Полностью открытая архитектура системы позволяет свободно копировать или дополнять линейку продукции Ардуино.

Arduino может использоваться как для создания автономных объектов автоматики, так и подключаться к программному обеспечению на компьютере через стандартные проводные и беспроводные интерфейсы. [1]

*Arduino Nano*

Arduino Nano построена на микроконтроллере ATmega328 (Arduino Nano 3.0) или ATmega168 (Arduino Nano 2.x), имеет небольшие размеры. Она имеет схожую с Arduino Duemilanove функциональность, однако отличается сборкой. Отличие заключается в отсутствии силового разъема постоянного тока и работе через кабель Mini-B USB. Nano разработана и продается компанией Gravitech. Ниже приведены характеристики Arduino Nano(см. табл. 1.1). [2]

Таблица 1.1 Краткие характеристики Arduino Nano

|  |  |
| --- | --- |
| Микроконтроллер | Atmel ATmega168 или ATmega328 |
| Рабочее напряжение (логическая уровень) | 5 В |
| Входное напряжение (предельное) | 6-20 В |
| Входное напряжение (рекомендуемое) | 7-12 В |
| Цифровые Входы/Выходы | 14 (6 из которых могут использоваться как выходы ШИМ) |
| Аналоговые входы | 8 |
| Постоянный ток через вход/выход | 40 мА |
| Флеш-память | 16 Кб (ATmega168) или 32 Кб (ATmega328) при этом 2 Кб используются для загрузчика |
| ОЗУ | 1 Кб (ATmega168) или 2 Кб (ATmega328) |
| EEPROM | 512 байт (ATmega168) или 1 Кб (ATmega328) |

Продолжение таблицы 1.1

|  |  |
| --- | --- |
| Тактовая частота | 16 МГц |
| Размеры | * 1. м x 4.2 см |

***1.2 Правила интеллектуальных игр***

* *Индивидуальная своя игра*

В Индивидуальную свою игру играют не команды, а отдельные игроки. Для игры используется пакет из определённого количества тем. Каждая тема состоит из пяти вопросов разной сложности, за которые даётся по 10, 20, 30, 40 и 50 очков соответственно. Обычно игра проходит в один раунд. Участники разбиваются на группы по 3-4 человека, которым ведущий читает определённое количество тем, равное для каждой группы. После каждого вопроса игроки имеют право заявить о желании ответить с помощью хлопка или кнопки брейн-системы. При неправильном ответе игрок теряет право отвечать на заданный вопрос, а его соперники имеют право ещё подумать и дать свой ответ. Затем задаётся следующий вопрос. Игроки имеют право давать сигнал об ответе во время чтения вопроса. Сразу после сигнала ведущий должен прекратить чтение вопроса. Вопрос дочитывается только в случае, если пожелавший ответить игрок дал неверный ответ. Игрок имеет право спросить формулировку вопроса, если такое право заранее обговорено. Набравший больше очков игрок (игроки) из каждой группы выходит в следующий круг.

* *Эрудит-Квартета*

В каждой игре участвует четыре команды. В составе каждой команды - четыре игрока. Игра состоит из трех раундов, в каждом из которых разыгрывается четыре темы. Каждый игрок команды обязан сыграть по одной теме в каждом раунде.

*"Открытый раунд".* Перед началом раунда ведущий объявляет все темы раунда, и капитан команды в течение 20 секунд определяет, кто из игроков играет какую тему.

*"Полузакрытый раунд".* Ведущий объявляет тему непосредственно перед началом ее розыгрыша, и капитан команды в течение 10 секунд определяет игрока, играющего эту тему.

*"Закрытый раунд".* Перед началом раунда капитан в течение 20 секунд определяет, кто из игроков играет в каком порядке, после чего ведущий зачитывает темы непосредственно перед началом отыгрыша каждой из них после того, как игроки занимают свои игровые места.

В каждом теме разыгрываются 5 вопросов, имеющие номинал от 10 до 50 очков. Очки, набранные игроком в каждой теме, суммируются. Очки, набранные игроками одной команды в темах боя, суммируются.

Задача игроков - дать правильный ответ на вопрос раньше соперника. При выполнении этих условий игрок получает количество очков, соответствующее номиналу вопроса. Если игрок отвечает неправильно, он получает отрицательное количество очков, соответствующее номиналу вопроса.

Игрок сообщает ведущему о своем желании ответить на вопрос, нажимая на кнопку. Право отвечать предоставляется игроку, который первый нажал на кнопку. Игрок может сообщить о своем желании отвечать в любой момент после объявления темы и номинала вопроса. В случае неправильного ответа первого игрока, правом ответа может воспользоваться второй, затем -третий, затем - четвертый. Второй, третий и четвертый игроки имеют право сообщить о своем желании отвечать только после констатации ведущим неправильного ответа предыдущего игрока. Игрок не имеет права давать более одного ответа на один вопрос. На обдумывание каждого вопроса ведущий дает не более 5 секунд.

* *Правила Что? Где? Когда?*

ЧГК - командная игра, в которой команды соревнуются в умении находить правильный ответ на поставленный перед ними вопрос за ограниченное время. Цель команды в процессе игры - ответить на большее число вопросов, чем ее соперники.

Основные правила, рекомендованные МАК для всех турниров, описаны в кодексе ЧГК. Здесь приведены ключевые из них[3]:

В команде не допускается единовременное участие в игре более чем шести человек. Разрешается наличие ограниченного числа запасных игроков в команде и замены игроков между турами.

На обсуждение каждого вопроса выделяется 1 минута (для блица, состоящего из произвольного числа вопросов (как правило, 2—3), суммарное время обсуждения составляет также 1 минуту).

Ответы сдаются в письменном виде.

Основной показатель, по которому определяется место команды, — количество правильных ответов. При равенстве количества правильных ответов могут использоваться дополнительные показатели (определяемые регламентом турнира).

Если ответ не в точности совпадает с авторским, он может быть зачтён (при условии, если эти разночтения предусмотрены в ответе автора вопроса), либо игровым жюри, либо апелляционным жюри (в случае подачи апелляции).

Апелляционное жюри рассматривает апелляции двух типов: требования зачёта ответа, удовлетворяющего всем условиям вопроса (так называемая «дуаль»), либо требования снятия вопроса (в случае наличия в нём фактической ошибки, в зависимости от регламента конкретного турнира). Регламент конкретного турнира может запрещать апелляции одного или обоих типов.

* *Тройка*

«Тройка» как игра чрезвычайно проста. В ней три тура, в каждом туре три темы, в каждой теме — по три вопроса. В первом туре цена правильного ответа — одно очко, во втором — два, а в третьем — три. А после третьего тура — суперигра с тремя вопросами, которые стоят соответственно 5, 7 и 10 очков).

В «Тройку» играют две «упряжки» — команды из трех человек. Один игрок — «коренник», и в его руках кнопка от электронной системы, прочие — «пристяжные».

Обсуждение (а равно подача сигналов друг другу — «знаю, жми!») запрещены. В случае любого действия, похожего на подачу такого сигнала, команда лишается права отвечать — в пользу соперника. Игроки безмолвствуют до тех пор, пока к ним не обратится ведущий.

Игровой эпизод строится так. Ведущий читает вопрос, а «коренник» в любую секунду может подать сигнал. Тогда ведущий обращается за ответом сначала к «пристяжным», а потом и к «кореннику». Каждый из «упряжки» может ответить правильно; таким образом, команда может за один вопрос получить максимум три очка в первом туре, шесть во втором (когда ставки удвоены) и девять — в третьем (при утроении ставок). Правда, могут быть даны и три неправильных ответа — тогда ведущий (иногда — дочитав вопрос до конца) обращается за ответом (за тремя ответами, по той же схеме) к сопернику.

* *Брейн-ринг*

В "Брэйн ринг" играют две команды по шесть игроков. Все бои идут до 5 очков. О готовности дать ответ на прозвучавший вопрос, команда сигнализирует нажатием на кнопку. Нажимать кнопку можно во временном интервале от начала сигналы сирены (которая звучит после команды ведущего "Время!") и до истечения 20 секунд игрового времени. Если команда, допускает фальстарт, т.е. нажимает на кнопку до начала сигнала сирены, то она лишается права ответа на данный вопрос. За неправильный ответ на 1-ой секунде команда получает штраф - красную карточку игроку, который дал неправильный ответ. Этот игрок удаляется с ринга до конца боя.

Если команда, нажавшая на кнопку первой, ответила неправильно, вторая команда может дать свой ответ на вопрос. Для обсуждения вторая команда может использовать оставшееся игровое время.

Команда, которая даст правильный ответ, получает одно очко.

## ***1.3 Обзор аналогов***

В качестве аналога были рассмотрены система для «Своей игры» пользователя @Serezha\_zp и «Portable Arduino Brain System» Артёма Понкратова. [4]

Система, предназначенная для «Своей игры», имеет разъемы для пультов 4P4C (RJ11, обычно применяется в телефонных аппаратах), два возможных источника питания — внешний и внутренний. Также есть разъем внутрисхемного программирования (ISP). Изображение представлено на рисунке 1.3.1

Рисунок 1.3.1 Система для «Своей игры»

Portable Arduino Brain System используется для проведения турниров по спортивной версии игр «Брейн-ринг», «Своя игра», «Эрудит-квартет», «Тройка», «Что? Где? Когда?» (см. рис. 1.3.2). Система имеет пять игровых пультов, экраном отображения цифровой информации состояния игры, звуковым оповещением и возможностью выбора режима игры. Устройство обладает портативным источником питания для работы в автономном режиме в течение 8 часов и 5В через разъем mini USB.

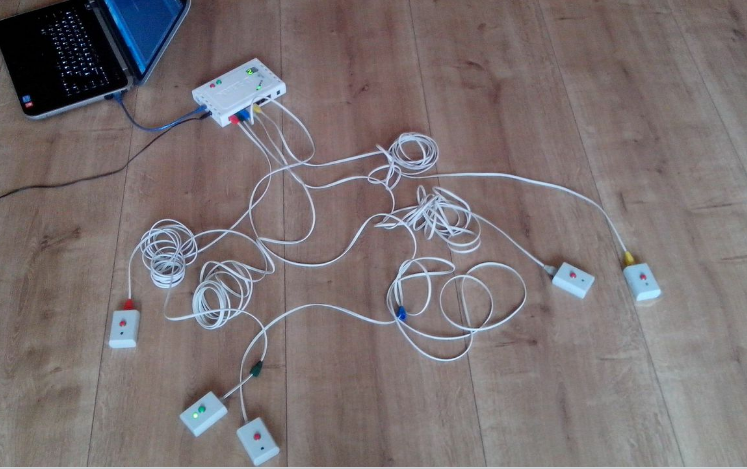


Рисунок 1.3.2 Portable Arduino Brain System

1. Разработка структурной схемы

Структурная схема определяет основные функциональные части изделия, их назначение и взаимосвязи между ними. Схема отображает принцип действия изделия в самом общем виде. Но позволяет описать основной принцип работы разрабатываемой системы.

Брейн-система состоит из трех блоков:

* *Главный модуль (пульт ведущего*)

Этот блок отображает состояние игры посредствам светодиодов. Один светодиод соответствует одному пульту игрока. Если игрок нажал кнопку на своем пульте, то на главном блоке загорается диод, соответствующий данному пульту игрока. Так ведущий может определить какой игрок нажал кнопку раньше остальных. Также данные модуль осуществляет выбор режима игры. Если выбранный режим игры требует отсчета времени, то главный модуль запускает таймер по истечение которого прозвучит звуковой сигнал.

* *Пульт игрока (кнопка игрока)*

Система содержит 4 пульта игрока, то есть одновременно могут играть четыре команды или четыре игрока в зависимости от вида игры. Данные блоки необходимы для определения игрока, который нажал на кнопку. Пульты игроков, которые не успели нажать, блокируются главным модулем. Если игрок успел нажать раньше остальных, то на его пульте загорается светодиод.

* *Соединительные кабели*

В данном курсовом проекте используется кабель Ethernet. Он обеспечивает связь между главным модулем и пультам игроков.

Структурная схема представлена в приложении А.

1. Разработка функциональной схемы

*Пульт игрока*

Пульт игрока состоит из:

* + кнопки без фиксации.
  + пластикового корпуса
  + светодиода
  + разъема RJ45 для связи с главным модулем.

При нажатии на кнопку загорается диод на кнопке и на пульте ведущего.

*Главный модуль*

Главный модуль состоит из следующих блоков:

* + корпус. В качестве корпуса использовался модем D-Link DWL - G700AP. Также использовались разъемы RJ45, входящие в состав модема.
  + программируемая плата Arduino
  + диод
  + динамик
  + кнопки «SET» и «RESET»
  + DIP-SWITCH-4. Переключатель используется для выбора игры.

*Связь блоков*

Для связи главного блока и пультов игрока используется разъем RJ-45 и кабель Ethernet. Разъем RJ-45 является распространенным портом для подключения персональных компьютеров к компьютерной сети.

Так как пульт и кабель являются подвижными элементами системы, то велика вероятность выхода кабеля из строя. Большинство интеллектуальных игр проходит в местах, где установлены компьютеры, у которых можно позаимствовать кабель в случае поломки основного.

Функциональная схема представлена в приложении Б.

1. Разработка принципиальной схемы

***4.1 Пульт игрока***

Для разработки пульта игрока использовался корпус G250, так как в нем компактно размещаются все необходимые элементы и он достаточно прочный (см. рис. 5.1.1).

Рисунок 4.1.1 Корпус для пульта игрока G250

В качестве кнопки испольовался переключатель кнопочный PBS-26B(Y) без фиксации OFF-on 25VAC (см. табл. 4.1.1). Данный переключатель представлен на рисунке 4.1.2

Таблица 4.1.1 Технические характеристики переключателя PBS-26B(Y)

|  |  |
| --- | --- |
| Напряжение коммутируемое максимальное, В | 125 AC |
| Коммутируемый ток, А | 3 |
| Тип | ON-OFF без фиксации |



Рисунок 4.1.2 Переключатель кнопочный PBS-26B(Y)

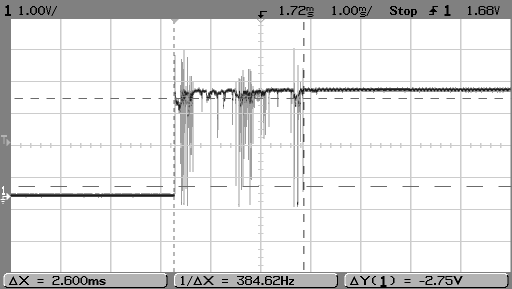
При нажатии на кнопку возникает дребезг сигнала. Дребезг контактов — явление, возникающее в электрических и электронных переключателях, при котором они вместо некоторого стабильного переключения производят случайные многократные неконтролируемые замыкания и размыкания контактов(см. рис. 4.1.3).

Рисунок 4.1.3 Дребезг сигнала

При нажатии на тактовую кнопку, перед тем, как контакты плотно соприкоснутся, они будут колебаться, порождая множество срабатываний вместо одного. Соответственно, микроконтроллер «поймает» все эти нажатия.

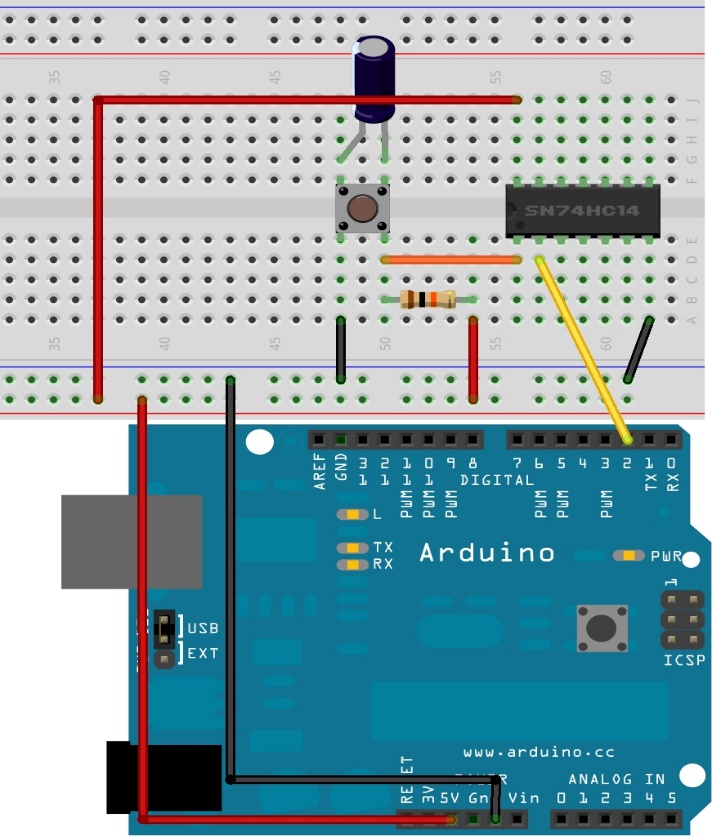
Для устранения данного эффекта будет использоваться аппаратный способ (см. рис. 4.1.4).

Рисунок 4.1.4 Схема, устраняющая дребезг сигнала

Аппаратный способ требует добавления инвертирующего триггера Шмитта, конденсатора на 1 мкф и резистора 10кОм. В качестве триггера Шмитта используется логическая микросхема CD74HC14E в корпусе dip14(см. рис 4.1.5).

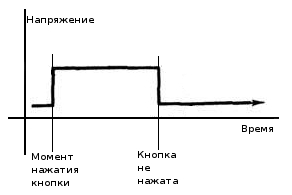
Рисунок 4.1.5 Триггер Шмитта

Резистор 10 кОм подтягивает вывод к +5В. Подтягивается вход инвертирующего триггера и конденсатор. Кнопка подключена к земле. Сделано это в связи с тем, что триггер инвертирующий. Входящий аналоговый сигнал может быть восходящим, либо нисходящим. Внутри триггера определены пороговые значения: ~1,6 В — для нисходящего сигнала и ~0,9 В — для восходящего. Выход триггера становится логической единицей (5 В), при прохождении нисходящим сигналом через свой порог в 0,9 В, при этом нисхождение восходящим сигналом верхнего порога в 1,6 В будет проигнорировано. Аналогично, выход триггера становится логическим нулём (0 В), только когда восходящий сигнал проходит свой порог в 1,6 В, при этом проигнорируется прохождение нисходящим сигналом нижнего порога в 0,9 В. Конденсатор с резистором образует RC-цепь. RC-цепь замедляет затухание сигнала. То есть при снятии напряжения, оно пропадёт не сразу, а будет плавно снижаться. При нажатии кнопки конденсатор начнёт разряжаться и напряжение начнёт плавно падать.

Инвертирующий триггер выполняет 2 функции:

1) он инвертирует (переворачивает) логические уровни: единица становится нулём, а ноль становится единицей.

2) он преобразует аналоговый сигнал в цифровой. То есть, в определённый момент, когда конденсатор разрядится до определённого уровня, триггер выдаст логическую единицу. При отпускании кнопки, при достижении определённого уровня заряда конденсатора, триггер установит на выходе низкий уровень. Сигнал на выходе триггера не будет зависеть от дребезга (см. рис. 4.1.6). [5]

Рисунок 4.1.6 Сигнал без дребезга

В пульте игрока также используется Светодиод выводной ARL2-5613LRD-B и резистор 100 Ом.

## ***4.2 Главный модуль***

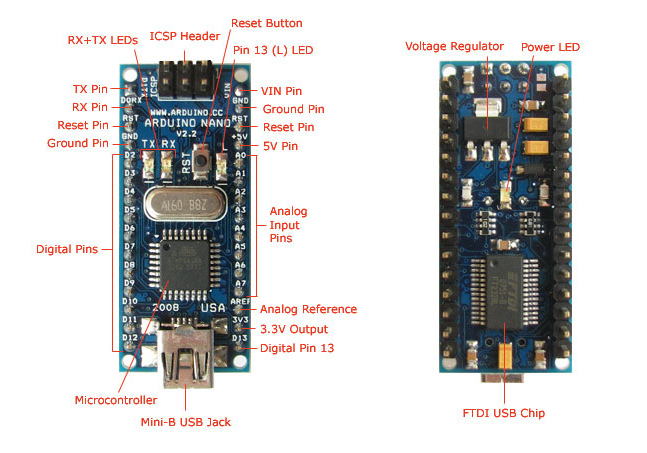
Основным процессорным устройством была выбрана плата Arduino Nano(см. рис. 4.2.1), так как она небольшого размера, обладает необходимым количеством портов ввода-вывода и имеет порт microUSB, который позволяет использовать большинство зарядных устройств.

Рисунок 4.2.1 Arduino Nano

В модуле ведущего также расположены 4 диода и 4 резистора 100 Ом, для определения нажатой кнопки. Для подачи звукового сигнала используется Звонок пьезоэлектрический KPI-G1410-06-K4814. Управление ходом игры осуществляется с помощью кнопок «SET» и «RESET». Для правильной работы данных кнопок также используется аппаратный способ подавления дребезга.

Проектируемая брейн-система должна предоставлять возможность выбора режима игры, это осуществляется с помощью DIP переключателя DS-04(см. рис. 4.2.2).

Рисунок 4.2.2 DIP переключателя DS-04

Для некоторых игр достаточно всего два пульта игрока, тогда в остальных разъемах не будет подключенных кнопок и на Arduino будет приходить «никакой» сигнал. Если вход не подключен, то digitalRead() и analogRead() может возвращать значения HIGH или LOW случайным образом. Чтобы гарантировать отсутствие напряжения при разомкнутой цепи, рядом с входом будет использоваться стягивающий резистор. Нежелательный ток будет уходить через резистор в землю. Для стягивания используются резисторы больших сопротивлений (10 кОм и более). В моменты, когда цепь замкнута, большое сопротивление резистора не даёт большей части тока идти в землю: сигнал пойдёт к входному контакту. [6]

В качестве стягивающего резистора используются резисторы 10кОм подключенные в главном модуле. Схема включения представлена на рисунке 4.2.3

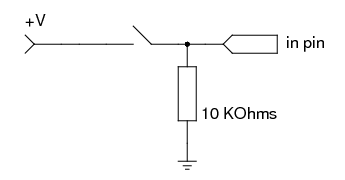
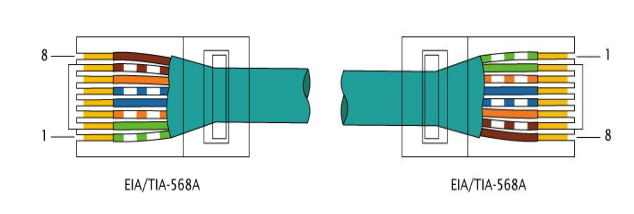


Рисунок 4.2.3 схема включения стягивающего резистора

## ***4.3 Связь модулей***

Для связи модулей используется Разъём телефонный TJ8-10P8 / розетка 10p8c (RJ45) и прямой Ethrnet-кабель 3м. Способы обжима прямого кабеля Ethernet представлены на рисунке 4.2.4

Разъемы на прямом кабеле имеют одинаковую последовательность. [7]



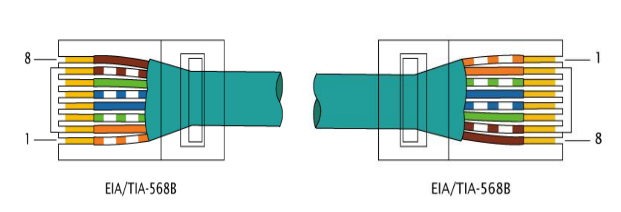


Рисунок 4.2.4 Способы обжима прямого кабеля Ethernet

Для обеспечения взаимодействия главного модуля и пультов игроков необходимо передавать от кнопок игроков в главный модуль сигнал нажатия кнопки, а из главного модуля в пульт игрока следует посылать логический нуль (0 В), логическую единицу (5 В) и сигнал, который подается на диод в кнопке игрока.

При проектировании данной системы было решено использовать первый проводник для передачи логической единицы(бело-оранжевый), второй проводник для логического нуля(оранжевый), третий проводник для передачи сигнала на диод (бело-зеленый) и шестой проводник для передачи сигнала с кнопки (зеленый).

Принципиальная схема представлена в приложении В

## ***4.4 Pin Change Interrupt***

В данной брейн-системе для обработки нажатий кнопок используются Pin Change Interrupt Requests, которые позволяют установить прерывания практически на все пины Ардуино:

* Pin Change Interrupt Request 0 (pins D8 to D13) (PCINT0\_vect)
* Pin Change Interrupt Request 1 (pins A0 to A5) (PCINT1\_vect)
* Pin Change Interrupt Request 2 (pins D0 to D7) (PCINT2\_vect)

Каждому из PCI запросов соответствует свой 8-битный Pin Change Mask Register (PCMSKn). Единицы в битах регистров указывают на пины для прерываний. В данном случае прерывания разрешены для следующих пинов (см. таблица 1):

* A0-A3 - кнопки пользователей;
* D8 – кнопка старта;
* D9 – кнопка сброса;

Для разрешения прерываний необходимым группам используется Pin Change Interrupt Control Register. Так как данная брейн-система использует всего две группы прерываний, то биты 0 и 1 установлены в единицу (см. таблица 1).

Для запрещения любых других прерываний на выбранных пинах, устанавливаются соответствующие группам биты в единицу в Pin Change Interrupt Flag Register (см. таблица 4.1).

Таблица 4.1 в Pin Change Interrupt Flag Register

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 7 bit | 6 bit | 5 bit | 4 bit | 3 bit | 2 bit | 1 bit | 0 bit |
|  |  |  | D13 | D12 | D11 | D10 | D9 | D8 |
| PCMSK0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
|  |  |  | A5 | A4 | A3 | A2 | A1 | A0 |
| PCMSK1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
|  |  |  |  |  |  |  |  |  |
| PCICR | - | - | - | - | - | 0 | 1 | 1 |
|  |  |  |  |  |  |  |  |  |
| PCIFR | - | - | - | - | - | 0 | 1 | 1 |

Когда пин изменяет свое состояние (c HIGH на LOW, с LOW на HIGH), соответствующий бит в PCMSKn регистре установлен в единицу, соответствующий бит в PCIFR регистре также установлен в единицу, то микроконтроллер переходит в соответствующий вектор прерывания.

Настройка прерываний осуществляется при передаче номера пина в функцию pinSetup() (см. Приложение Г).

Определены два обработчика прерывания для каждой из используемых групп прерываний.

* ISR(PCINT0\_vect) – для кнопок администратора.
* ISR(PCINT1\_vect) – для пользовательских кнопок.

Обработчики прерываний поочередно опрашивают пины, чтобы определить, какая кнопка была нажата. Такая ситуация является одним из ограничений данных прерываний. Брейн-система должна реагировать на нажатие кнопок, поэтому значение пинов сравнивается с уровнем HIGH (то есть кнопка была нажата) в обработчике прерываний.

1. Руководство пользователя

Перед запуском системы необходимо выбрать один из режимов игры с помощью dip-switch.

1 – Брейн-ринг

2 – Что? Где? Когда?

3 – Эрудит-квартет/Своя игра/Тройка

В режиме «брейн-ринга» система ожидает нажатия кнопки «SET» ведущего. До нажатия этой кнопки, все нажатия кнопок игроков интерпретируются как фальстарт. После нажатия кнопки «SET» ведущим следует звуковой сигнал, во время которого игроки могут нажимать свои кнопки без фальстарта. Так же, после нажатия кнопки игрока, система ожидает действия ведущего – нажатие кнопки «SET» для продолжения розыгрыша вопроса или нажатие кнопки «RESET» для перехода в начальное состояние и разблокировки всех кнопок ведущего. По истечение 20 секунд после первого нажатия и 10 секунд после второго нажатия на «SET» система подает звуковой сигнал.

Режим «своей игры» отличается от режима «брейн-ринга» тем, что отсутствует звуковой сигнал.

Режим «Что? Где? Когда?» использует только кнопки ведущего. По нажатию кнопки «Старт» начинается отсчет 60-ти. Нажатие кнопки «Сброс» останавливает и сбрасывает таймер. Предусмотрена возможность досрочного ответа, если игроки успели нажать кнопку в течение одной секунды после запуска таймера.

Вывод

В ходе разработки аппаратно-программного комплекса для проведения интеллектуальных игр, на практике были знания по различным направлениям, полученные в рамках обучения на специальности 1-40 02 01 "Вычислительные машины, системы и сети" в Белорусском государственном университете информатики и радиоэлектроники.

В результате была получена система, которая удовлетворяет поставленным требованиям. Система может использоваться для проведения турниров по спортивной версии игр «Брейн-ринг», «Своя игра», «Эрудит квартет», «Тройки», «Что? Где? Когда?», а также использоваться на тренировочных играх команд интеллектуальных игр.

Также планируется добавить возможность подачи звуковых сигналов через наушники.

Созданная система получила название FBS.

Ссылка на статью https://geektimes.ru/post/276356/

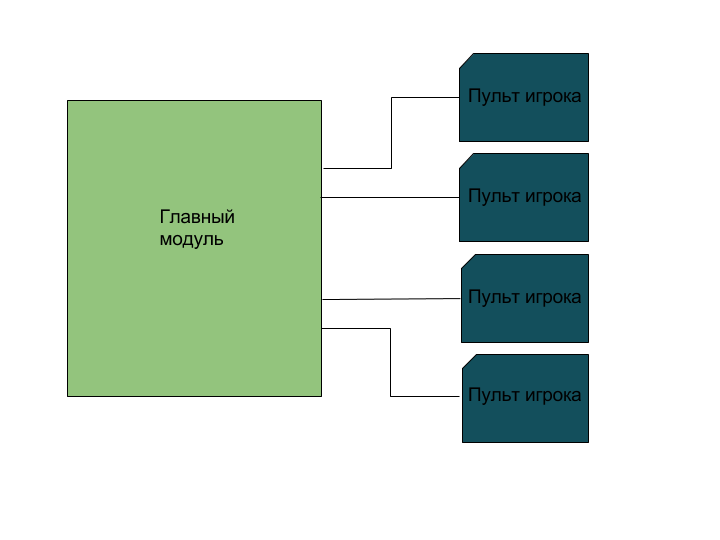
# Список литературы

1. Arduino - [Электронный ресурс]. – Электронные данные. – Режим доступа https://ru.wikipedia.org/wiki/Arduino
2. Arduino Nano - [Электронный ресурс]. – Электронные данные. – Режим доступа http://arduino.ru/Hardware/ArduinoBoardNano
3. Кодекс спортивного ЧГК - [Электронный ресурс]. – Электронные данные. – Режим доступа http://chgk.tvigra.ru/mak/?documents/kodeks
4. Система для «Своей игры» - [Электронный ресурс]. – Электронные данные. – Режим доступа https://geektimes.ru
5. Дребезг сигналов - [Электронный ресурс]. – Электронные данные. – Режим доступа https://uscr.ru/drebezg-kontaktov-i-sposoby-podavleniya-drebezga/
6. Стягивающий резистор - [Электронный ресурс]. – Электронные данные. – Режим доступа http://wiki.amperka.ru/%D1%81%D1%85%D0%B5%D0%BC%D0%BE%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0:%D1%80%D0%B5%D0%B7%D0%B8%D1%81%D1%82%D0%BE%D1%80%D1%8B
7. Ethernet – кабель - [Электронный ресурс]. – Электронные данные. – Режим доступа

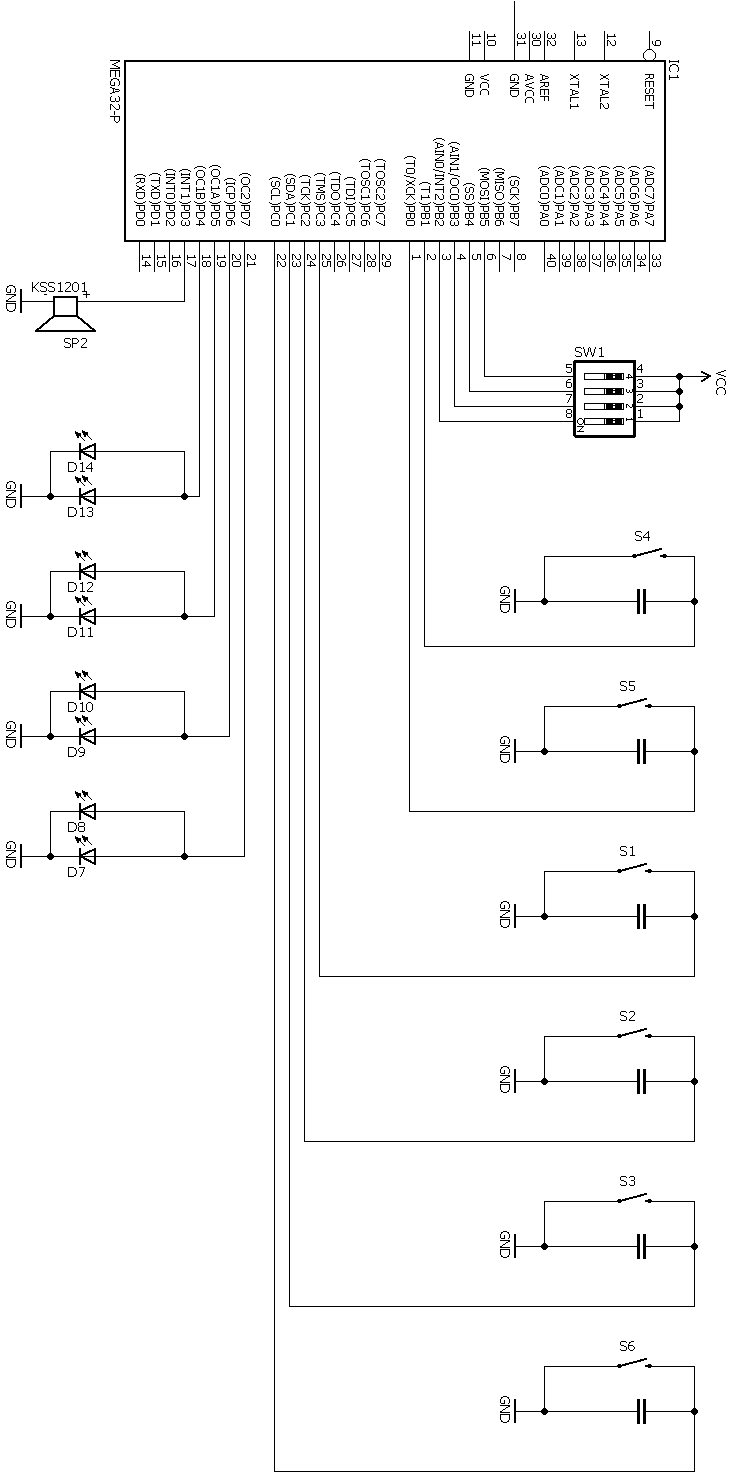
https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D1%82%D0%B0%D1%8F\_%D0%BF%D0%B0%D1%80%D0%B0

# ПРИЛОЖЕНИЕ А

Структурная схема

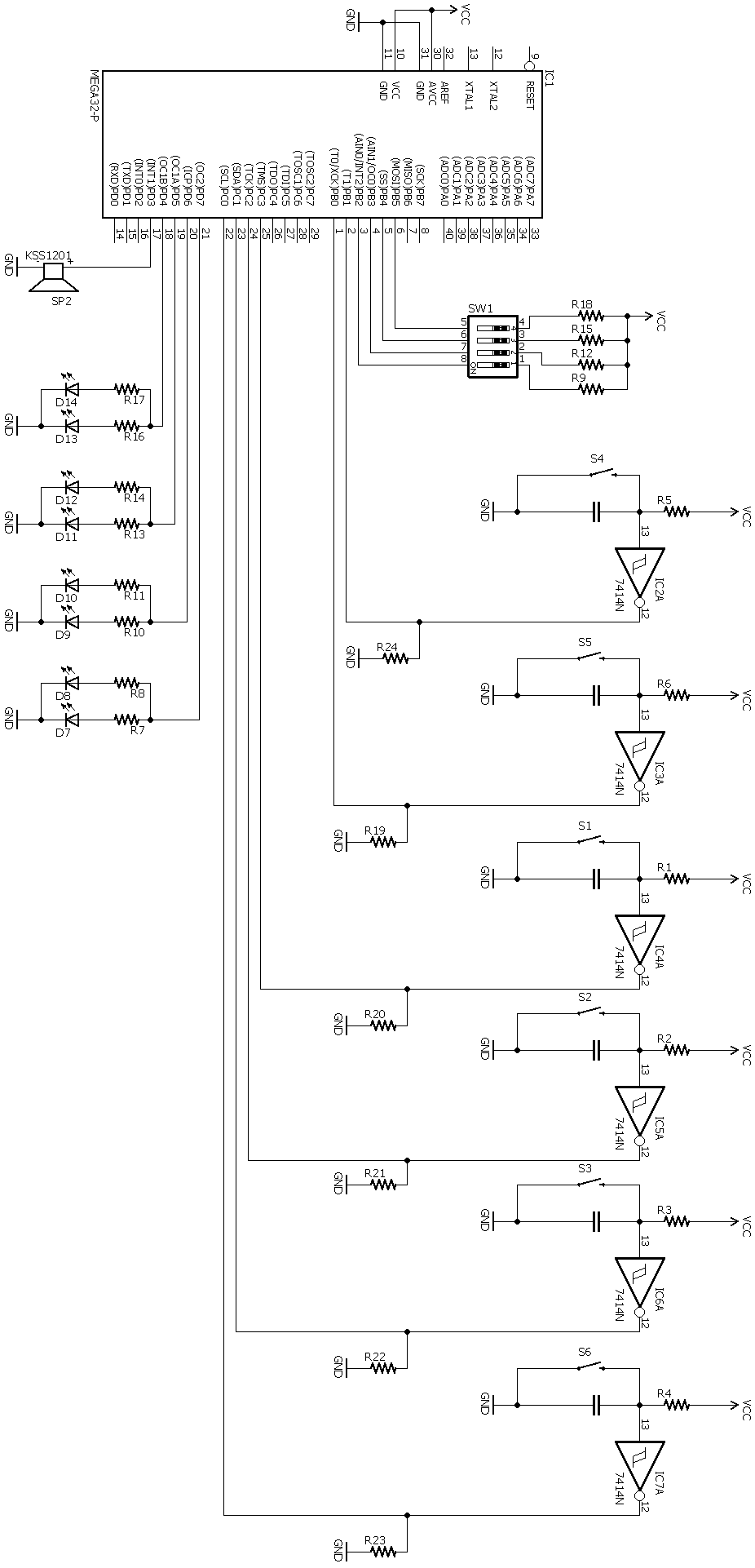


# ПРИЛОЖЕНИЕ Б



Функциональная схема

# ПРИЛОЖЕНИЕ В



Принципиальная схема

# ПРИЛОЖЕНИЕ Г

Код программы

BrainSystem.ino

#include "Constants.h"

#include "BrainRingMode.h"

#include "GameMode.h"

#include "TimerOne.h"

#include "WwwMode.h"

#include "QuartetMode.h"

GameMode \*gameMode;

bool isPushed = false;

void setup() {

for (int i = 0; i < ARRAY\_SIZE(ARRAY\_USER\_BUTTON); i++) {

pinMode(ARRAY\_USER\_BUTTON[i], INPUT);

pciSetup(ARRAY\_USER\_BUTTON[i]);

}

for (int i = 0; i < ARRAY\_SIZE(ARRAY\_LED); i++) {

pinMode(ARRAY\_LED[i], OUTPUT);

}

pinMode(ADMIN\_BUTTON\_RESET, INPUT);

pinMode(ADMIN\_BUTTON\_SET, INPUT);

pinMode(BRAIN\_RING\_MODE, INPUT);

pinMode(WWW\_MODE, INPUT);

pinMode(QUARTET\_MODE, INPUT);

pinMode(SOUND\_PIN, OUTPUT);

pciSetup(ADMIN\_BUTTON\_RESET);

pciSetup(ADMIN\_BUTTON\_SET);

if (digitalRead(BRAIN\_RING\_MODE) == HIGH ) {

gameMode = new BrainRingMode();

}

else if (digitalRead(WWW\_MODE) == HIGH) {

gameMode = new WwwMode();

}

else if (digitalRead(QUARTET\_MODE) == HIGH) {

gameMode = new QuartetMode();

}

else {

gameMode = new BrainRingMode();

}

Timer1.initialize(1000000);

Timer1.stop();

}

void loop() {

}

void pciSetup(byte pin) {

\*digitalPinToPCMSK(pin) |= bit(digitalPinToPCMSKbit(pin)); // enable pin

PCIFR |= bit(digitalPinToPCICRbit(pin)); // clear any outstanding interrupt

PCICR |= bit(digitalPinToPCICRbit(pin)); // enable interrupt for the group

}

ISR(PCINT0\_vect) {

noInterrupts();

isPushed = false;

if (digitalRead(ADMIN\_BUTTON\_SET) == HIGH ) {

gameMode->Set();

Timer1.attachInterrupt(TimerInterrupt);

gameMode->SetFalseStart(false);

}

if (digitalRead(ADMIN\_BUTTON\_RESET) == HIGH) {

Timer1.stop();

gameMode->Reset();

gameMode->SetFalseStart(true);

}

interrupts();

}

ISR(PCINT1\_vect){

noInterrupts();

if (isPushed == false){

for (int i = 0; i < ARRAY\_SIZE(ARRAY\_USER\_BUTTON); i++){

if (digitalRead(ARRAY\_USER\_BUTTON[i]) == HIGH){

isPushed = gameMode->UserButtonPushed(ARRAY\_LED[i]);

}

}

}

interrupts();

}

void TimerInterrupt() {

gameMode->SetTimer();

}

BrainRingMode.h

#pragma once

#include "GameMode.h"

#include "Constants.h"

#define FIRST\_TIMER 20

#define SECOND\_TIMER 10

class BrainRingMode : public GameMode {

private:

int setPushes;

int firstTimer;

int seconedTimer;

int timer;

public:

BrainRingMode();

~BrainRingMode();

virtual void Set();

virtual void Reset();

virtual void SetTimer();

virtual void FalseStart();

void CheckTimeLeft(int timeLeft);

virtual bool UserButtonPushed(int pin);

};

BraingRingMode.cpp

#include "BrainRingMode.h"

#include "Arduino.h"

#include "TimerOne.h"

BrainRingMode::BrainRingMode() {

setPushes = 0;

}

BrainRingMode::~BrainRingMode() {

}

void BrainRingMode::Set() {

if (setPushes == 0) {

tone(SOUND\_PIN, FREQUENCY, TIME);

timer = FIRST\_TIMER;

digitalWrite(GetPin(), LOW);

}

else if (setPushes == 1) {

tone(SOUND\_PIN, FREQUENCY, TIME);

digitalWrite(GetPin(), LOW);

digitalWrite(4, HIGH);

timer = SECOND\_TIMER;

}

setPushes++;

}

void BrainRingMode::SetTimer() {

timer--;

if (timer == 0) {

tone(SOUND\_PIN, FREQUENCY, TIME);

this->SetFalseStart(true);

}

}

void BrainRingMode::Reset(){

setPushes = 0;

for (int i = 0; i < ARRAY\_SIZE(ARRAY\_LED); i++){

digitalWrite(ARRAY\_LED[i], LOW);

}

digitalWrite(13, LOW);

}

void BrainRingMode::FalseStart() {

tone(SOUND\_PIN, FREQUENCY\_FALSESTART, TIME);

}

bool BrainRingMode::UserButtonPushed(int pin){

digitalWrite(pin, HIGH);

this->SetPin(pin);

if (this->GetFalseStart()){

this->FalseStart();

return false;

}

else{

tone(SOUND\_PIN, FREQUENCY\_USER, TIME);

Timer1.stop();

return true;

}

}

Constants.h

#pragma once

#define USER\_BUTTON\_1 14

#define USER\_BUTTON\_2 15

#define USER\_BUTTON\_3 16

#define USER\_BUTTON\_4 17

const int ARRAY\_USER\_BUTTON[] = { 14, 15, 16, 17 };

#define ARRAY\_SIZE(array) (sizeof((array))/sizeof((array[0])))

const int ARRAY\_LED[] = { 4, 5, 6, 7 };

#define ADMIN\_BUTTON\_SET 8

#define ADMIN\_BUTTON\_RESET 9

#define SOUND\_PIN 3

#define FREQUENCY 2000

#define FREQUENCY\_USER 1500

#define FREQUENCY\_FALSESTART 200

#define TIME 500

#define BRAIN\_RING\_MODE 10

#define WWW\_MODE 11

#define QUARTET\_MODE 12

GameMode.h

#pragma once

class GameMode {

private:

int pin;

bool isFalseStart = true;

public:

virtual void Set();

virtual void Reset();

virtual void SetTimer();

virtual void FalseStart();

void SetPin(int pin);

virtual bool UserButtonPushed(int pin);

int GetPin();

bool GetFalseStart();

void SetFalseStart(bool value);

};

GameMode.cpp

#include "Arduino.h"

#include "WwwMode.h"

#include "TimerOne.h"

void WwwMode::Set(){

tone(SOUND\_PIN, FREQUENCY, TIME);

timer = WWW\_TIMER;

}

void WwwMode::Reset(){

digitalWrite(GetPin(), LOW);

}

void WwwMode::SetTimer(){

timer--;

if (timer == 0) {

tone(SOUND\_PIN, FREQUENCY, TIME);

}

if (timer == 9){

tone(SOUND\_PIN, FREQUENCY, TIME);

}

}

bool WwwMode::UserButtonPushed(int pin){

if (timer >= 59){

digitalWrite(pin, HIGH);

tone(SOUND\_PIN, FREQUENCY\_USER, TIME);

Timer1.stop();

this->SetPin(pin);

}

return true;

}

void WwwMode::FalseStart(){

}

WwwMode::WwwMode(){

}

WwwMode::~WwwMode(){

}

QuartetMode.h

#pragma once

#include "GameMode.h"

#include "Constants.h"

class QuartetMode : public GameMode{

private:

int timer;

public:

QuartetMode();

~QuartetMode();

virtual void Set();

virtual void Reset();

virtual void SetTimer();

virtual void FalseStart();

virtual bool UserButtonPushed(int pin);

};

QuartetMode.cpp

#include "Arduino.h"

#include "QuartetMode.h"

#include "TimerOne.h"

void QuartetMode::Set(){

}

void QuartetMode::Reset(){

digitalWrite(GetPin(), LOW);

}

void QuartetMode::SetTimer(){

}

void QuartetMode::FalseStart(){

}

QuartetMode::QuartetMode(){

}

QuartetMode::~QuartetMode(){

}

bool QuartetMode::UserButtonPushed(int pin){

digitalWrite(pin, HIGH);

tone(SOUND\_PIN, FREQUENCY\_USER, TIME);

this->SetPin(pin);

return true;

}

WwwMode.h

#pragma once

#include "GameMode.h"

#include "Constants.h"

#define WWW\_TIMER 60

class WwwMode : public GameMode{

private:

int timer;

public:

WwwMode();

~WwwMode();

virtual void Set();

virtual void Reset();

virtual void SetTimer();

virtual void FalseStart();

virtual bool UserButtonPushed(int pin);

};

WwwMode.cpp

#include "Arduino.h"

#include "WwwMode.h"

#include "TimerOne.h"

void WwwMode::Set(){

tone(SOUND\_PIN, FREQUENCY, TIME);

timer = WWW\_TIMER;

}

void WwwMode::Reset(){

digitalWrite(GetPin(), LOW);

}

void WwwMode::SetTimer(){

timer--;

if (timer == 0) {

tone(SOUND\_PIN, FREQUENCY, TIME);

}

if (timer == 9){

tone(SOUND\_PIN, FREQUENCY, TIME);

}

}

bool WwwMode::UserButtonPushed(int pin){

if (timer >= 59){

digitalWrite(pin, HIGH);

tone(SOUND\_PIN, FREQUENCY\_USER, TIME);

Timer1.stop();

this->SetPin(pin);

}

return true;

}

void WwwMode::FalseStart(){

}

WwwMode::WwwMode(){

}

WwwMode::~WwwMode(){

}