МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БУРЯТИЯ
ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ УЧРЕЖДЕНИЕ ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БУРЯТИЯ «РЕСУРСНЫЙ ЦЕНТР ХУДОЖЕСТВЕННОГО И ТЕХНИЧЕСКОГО ТВОРЧЕСТВА «СОЗВЕЗДИЕ»
ДЕТСКИЙ ТЕХНОПАРК «КВАНТОРИУМ» РЕСПУБЛИКИ БУРЯТИЯ
МОБИЛЬНЫЙ ТЕХНОПАРК «КВАНТОРИУМ»

«Принято»
На Педагогическом совете

Протокол № 1
От « » _____ 20 г.

«Утверждаю»
Директор ГАУ ДО РБ
«РЦХТТ «Созвездие»
_____ Д.Г. Фролов
Печать

ДОПОЛНИТЕЛЬНАЯ ОБЩЕРАЗВИВАЮЩАЯ ПРОГРАММА

«IT квантум. Юный программист»

Модуль вводный

Направленность: *техническая*

Возраст детей: *12 - 17 лет*

срок реализации: 1 *год (36 часов)*

Автор-разработчик: *Бутин Александр Игоревич,*
педагог дополнительного образования.
Имитинов Василий Матвеевич,
педагог дополнительного образования.
Мархасаев Александр Владимирович,
педагог дополнительного образования.

г. Улан-Удэ

2021 год

I. Пояснительная записка

1. Основные характеристики программы

1.1 Направленность программы Данная общеразвивающая программа дополнительного образования вводного модуля «IT квантум. Юный программист» имеет техническую направленность и предусматривает развитие творческих способностей детей 12-17лет, формирование начальных технических ЗУНов, овладение soft и hard компетенциями, составлена на основании методических материалов IT-квантум тулкит. Белоусова Анна Сергеевна; Юбзаев Тимур Ильясович. – М.: Фонд новых форм развития образования, 2019форм, предназначенных для использования наставниками сети детских технопарков «Мобильный Кванториум» в соответствии с нормативными документами:

Программа направлена на формирование знаний и практических компетенций, современных устройств и существующего программного обеспечения и рассчитана на первый год обучения данному направлению.

Дополнительная общеобразовательная общеразвивающая программа приобщает учащихся к инженерно-техническим знаниям в области инновационных технологий, содействует развитию технического мышления.

Программа является модифицированной. При ее составлении были изучены и использованы следующие материалы:

- методические рекомендации Агентства стратегических инициатив «Новая модель дополнительного образования - Кванториум»;
- дополнительная общеразвивающая программа «Введение в основы алгоритмизации в средах визуального программирования и создание «умных» устройств» (составитель Белоусова А. С. Москва, 2019)

1.2 Актуальность изучения информационных технологий в следующем:

1. Доступность информации.
2. Интерактивность. Благодаря этому свойству, взаимодействие пользователя с объектом позволяет создавать большое количество различных способов обучения, так как объекты представляются очень реалистично. Например, человек может ремонтировать двигатель, и в настоящий момент получать инструкцию по выполнению работы.
3. «Вау-эффект». Необычный способ представления информации, который позволяет привлекать внимание, а также усиливать запоминание. На сегодняшний день это особенно актуально в образовании, так как дети могут воспринимать процесс обучения более увлекательным и наглядным.

В настоящее время процесс информатизации проявляется во всех сферах человеческой деятельности. Использование современных информационных технологий является необходимым условием успешного развития как отдельных отраслей, так и государства в целом. Отрасль информационных технологий является и будет являться в будущем одной из наиболее динамично развивающихся отраслей, как в мире, так и в России. Создание, внедрение, эксплуатация, а также совершенствование информационных технологий немыслимо без участия квалифицированных и увлеченных специалистов.

В IT-квантуме посредством метода проектной работы и с использованием кейс-технологий в рамках вводного модуля обучающиеся приобретут фундаментальные навыки и базовые знания в сфере IT:

- Микроэлектроника и схемотехника на примере интернета вещей;
- Языки и технологии программирования;
- Средства программирования;
- Веб-технологии.

1.3 Отличительные особенности программы

В рамках программы предусматривается изучение материала по модулям: стартовый, базовый и продвинутый. Основным методом изучения модуля является метод кейсов. (Кейс-это описание проблемной ситуации понятной и близкой обучающимся, решение которой требует всестороннего изучения, поиска дополнительной информации и моделирования ситуации или объекта, с выбором наиболее подходящего). Наряду с этим, программой предусматривается проектная деятельность.

Новизна общеразвивающей программы обусловлена использованием в образовательном процессе большого количества современных технических устройств виртуальной и дополненной реальности, что позволяет сделать процесс обучения не только интереснее, но и нагляднее и информативнее. Использование при обучении “открытого” программного обеспечения позволяет учащимся свободно использовать его на своих домашних устройствах, что позволяет им достаточно легко продолжить обучение и развиваться самостоятельно в этом направлении.

1.4. Педагогическая целесообразность программы

Педагогическая целесообразность состоит в том, что через изучение и овладение техническими знаниями и информационными технологиями формируется инженерное мышление современного ребенка, готового к разработке и внедрению инноваций в жизнь.

Основная задача педагога привлечь детей к исследовательской и изобретательской деятельности, развить у детей навыки, которые им потребуются в проектной работе и в дальнейшем освоении программы квантума.

Большое значение уделяется практике через **кейс-технологии** - это метод обучения, в основе которого лежат задачи из реальной жизни, и они направлены на развитие у детей soft и hard-компетенций.

Кейс-технология- это техника обучения, использующая описание реальной ситуации, специально подготовленный материал с описанием конкретной проблемы, которую необходимо разрешить в составе группы.

Кейс-технологии направлены на исследовательскую или инженерно-проектировочную деятельность. Интегрирует в себе технологию развивающего и проектного обучения. Выступают в обучении как синергетическая технология («погружение» в ситуацию, «умножение» знаний, «озарение», «открытие»). Позволяют создать ситуацию успеха.

1.5. Цель программы:

Целью программы является формирование уникальных компетенций по работе с IT технологиями и их применение в работе над проектами. Формирование интереса к техническим видам творчества, развитие конструктивного мышления с помощью языков программирования.

1.6. Задачи программы

- Сформировать практические и теоретические знания в области устройства и функционирования современных платформ быстрого прототипирования электронных устройств;
- Изучить основы алгоритмизации, построения алгоритмов и их формализации с помощью блок-схем;
- Научиться формулировать и анализировать алгоритмы;
- Научиться писать программы для решения простых и сложных инженерных задач в интегрированной среде разработки;
- Получить навыки работы с электронными компонентами, совместимыми с микроконтроллерами, такими как Arduino, Raspberry Pi и др.;
- Сформировать практические и теоретические навыки разработки приложений для операционной системы Android с использованием интерактивной среды разработки MIT App Inventor.
- Формирование научного мировоззрения;
- Усвоение определенного объема научных знаний.
- Развитие у обучающихся чувства ответственности, внутренней инициативы, самостоятельности, тяги к самосовершен-ствованию;
- Развитие познавательных интересов и формирование позна-вательной активности;
- Развитие творческих способностей обучающихся;
- Развитие алгоритмического мышления у обучающихся;

Формирование у обучающихся умения работать в команде и публично демонстрировать свои проекты

1.7. Возраст детей от 12-17 лет, с учетом возрастных и психологических особенностей содержание программы разделено на две возрастные группы 12-13 лет и 14-17 лет.

1.8. Основные формы и методы занятий:

Программой предусмотрено проведение комбинированных занятий, в том числе с применением дистанционных технологий. Занятия состоят из теоретической и практической частей, причём большее количество времени занимает практическая часть. При проведении занятий традиционно используются следующие формы работы:

· Практические занятия. ·

Занятие-соревнование. ·

Экскурсия.

· Workshop (рабочая мастерская-групповая работа, где все участники активны и самостоятельны);

· Консультация. ·

Выставка.

Методы:

- словесный: объяснение, рассказ, чтение, опрос, инструктаж, эвристическая беседа, дискуссия, консультация, диалог;
- наглядно - демонстрационный: показ, демонстрация образцов, иллюстраций, рисунков, фотографий, таблиц, схем, чертежей, моделей, предметов;
- практический: практическая работа, самостоятельная работа, творческая работа (творческие задания, работа с эмулятором), опыты;
- метод игры: ролевые, развивающие, метод диагностики: комплекс упражнений на развитие воображения, фантазии, задачи на плоскостное конструирование, творческие задания на рационально - логическое мышление, тесты на развитие у детей воссоздающего воображения, образного мышления, фантазии, словесно - логического мышления, задания на пространственное.
- методы стимулирования поведения и выполнения работы: похвала, поощрение;
- метод оценки: анализ, самооценка, взаимооценка, взаимоконтроль;
- метод информационно - коммуникативный поддержки: работа со специальной литературой, интернет ресурсами;
- метод компьютерного моделирования;
- метод проектный.

Форма проверки: итоговая аттестация – защита проекта.

1.9. Объём программы

1.9.1. Объём программы-36 часов

1.9.2. Срок реализации – 1 год

1.9.3. Режим занятий – очно- 3 раза в неделю по 2 часа.

II. Планируемые результаты освоения программы

2.1 Soft- компетенции

1.Владение информационным моделированием как основным методом приобретения знаний:
2.умение преобразовывать объект из чувственной формы в пространственно -графическую или знаково - символическую модель.
3.Формулирование вопросов, ответы на которые требуются для создания продукта, и другие навыки исследовательской деятельности.
4.Планирование и контроль процессов через проектную деятельность.

5.Разработка перечня ключевых показателей эффективности и их оценка.
6.Выступление с компьютерным сопровождением.

2.2 Hard – компетенции

1. Программирование микроконтроллерных платформ на языке C/C++.
2. Основы алгоритмизации и формализации алгоритмов.
3. Проектирование интерфейса пользователей и разработка приложений для мобильных устройств.
4. Разработка устройств интернета вещей и работа с облачными сервисами.

5. Основы языка разметки гипертекста HTML, языков программирования Python, JavaScript, формального языка CSS.
6. Базовые принципы объектно-ориентированного программирования.
7. Основы работы в специализированном ПО для создания презентаций.

III. Способы и формы проверки результатов

Для оценивания образовательного результата используются инструменты оценивания (оценочный лист) разработанных приложений — продукта проектной деятельности — по заранее составленным и согласованным с обучающимися критериям.
Универсальные навыки оцениваются через само- и взаимо- оценивание обучающихся. Предметные навыки и уровень их форсированности педагог может оценить вовремя презентации обучающимися своих продуктов с помощью журнала на- наблюдений.

Также используется инструмент «рефлексия», что подразумевает под собой, обратную связь, то что ученик рассказывает, что понял и не понял в процессе обучения, и работе с проектами, кейсами.

Показатели сформированности soft и hardskills

Уровень	**Описание поведенческих проявлений**
1 уровень - недостаточный	Обучающийся не владеет навыком, не понимает его важности, не пытается его применять и развивать.
2 уровень – развивающийся	- Обучающийся находится в процессе освоения данного навыка. - Обучающийся понимает важность освоения навыков, однако не всегда эффективно применяет его в практике.
3 уровень – опытный пользователь	- Обучающийся полностью освоил данный навык. - Обучающийся эффективно применяет навык во всех стандартных, типовых ситуациях.
4 уровень – продвинутый пользователь	- Особо высокая степень развития навыка. - Обучающийся способен применять навык в нестандартных ситуациях или ситуациях повышенной сложности.
5 уровень – мастерство	- Уровень развития навыка при котором обучающийся становится авторитетом и экспертом в среде сверстников. - Обучающийся способен передавать остальным необходимые знания и навыки для освоения и развития данного навыка.

Презентация проектных работ
Презентация проектных работ обучающихся. Выделенные микрокоманды в начале модуля работают над проектами по выбранной тематике, определяя её актуальность и пути решения.
Ход работы:

1. Подготовка к презентации работ.
2. Публичная презентация проектных работ.

Компетенции: умение аргументировать свою точку зрения и представлять её публично; ораторские навыки; умение грамотно формулировать и излагать свои мысли; навыки презентации.

IV. Учебно-тематическое планирование 12-13 лет

№	Наименование разделов и тем	Количество часов всего	Количество часов		Форма контроля
			теория	практика	
Стартовый уровень					
1	Вводное занятие. Правила техники безопасности и поведения на занятиях	2	2	0	опрос
2	Что такое Arduino?	2	2	0	Педагогическое наблюдение
3	Написание программного кода и схема техника**.** Кейс «Светофор»	8	2	6	
Базовый уровень					
4	Массивы и пьезоэлементы. Кейс «Маячок»	4	1	3	Педагогическое наблюдение
5	Кейс «Перетягивание каната»	2	0	2	Педагогическое наблюдение
6	Кейс «Азбука Морзе»	2	0	2	Педагогическое наблюдение
7	Шин и смешание цветов. Решение кейсов	4	0	4	Педагогическое наблюдение
Продвинутый уровень					
7	Сенсоры и датчики нажатия. Решение кейсов	4	1	3	Педагогическое наблюдение
8	Кейс «Кнопочные ковбои»	2	0	2	Педагогическое наблюдение
9.	Кейс «Секундомер»	2	0	2	
10.	Презентация проектных работ	4	0	4	зачет
	Итого	**36**	**8**	**28**	

Содержание программы учебно-тематического плана для учащихся 12-13 лет

Стартовый уровень

1. Вводное занятие. Правила техники безопасности и поведения на занятиях

Правила обращения с оборудованием. Изучаются основные понятия: Кванториум, наставник, форма подачи информации. Игра “Снежный ком” на знакомство с обучающимися.

Кол-во часов /занятий: 2/1

2. Что такое Arduino?

Деление на команды. Дата-скаутинг на тему Arduino (основные понятия, сфера применения, как работать с микроконтроллером). Подготовка презентации и защита (командная работа). Практическое занятие для освоения полученных знаний. Понятия программируемая среда, простейшие команды.

Кол-во часов /занятий: 2/1

3.Написание программного кода и схема техника. Кейс «Светофор»

Дети учатся писать программный код, и разбираются с схемами, деталями и техникой работы. В кейсе «Светофор» дети учатся собирать светофор, в котором должны меняться цвета.

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 8/4

Базовый уровень

4.Массивы и пьезоэлементы. Кейс «Маячок»

Теорияю Массивы и пьезоэлементы.Решение кейса «Маячок»В этом задании дети учатся собирать маячок, который мигает светодиодом.

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 4/2

5.Кейс «Перетягивание каната» В этом задании учащиеся создают еще одну игру, на этот раз нужно быстрее соперника нажать кнопку 20 раз.

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 2/1

6.Кейс «Азбука Морзе» В этом задании дети учатся собирать азбуку Морзе, и как ей пользоваться.

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 2/1

7.Шин и смешание цветов. Решение кейсов.

Кейс «Пульсар» В этом задании учащиеся плавно наращивают яркость светодиодной шкалы, управляя большой нагрузкой через транзистор.

Кейс «Ночной светильник» В этом задании светодиод должен включаться при падении уровня освещенности ниже порога, заданного потенциометром.

Кол-во часов /занятий: При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

4/2

Углубленный уровень

7. Сенсоры и датчики нажатия.Решение кейсов.

Кейс «Кнопочный переключатель» В этом задании учащиеся делают из тактовой кнопки триггер, борясь с «дребезгом».

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 4/2

8.Кейс «Кнопочные ковбои» В этом задании учащиеся создают игрушку на реакцию: кто быстрее нажмет кнопку по сигналу

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 2/1

9.Кейс «Секундомер» В этом задании учащиеся создают секундомер, который считает до 10

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 2/1

10. Презентация проектных работ

Презентация проектных работ обучающихся. Выделенные микрокоманды в начале модуля работают над проектами по выбранной тематике, определяя её актуальность и пути решения.

Кол-во часов /занятий: 4/2

V. Учебно-тематическое планирование 14-17 лет

№	Наименование разделов и тем	Количество часов всего	Количество часов		Форма контроля
			теория	практика	
Стартовый уровень					
1	Вводное занятие. Правила техники безопасности и поведения на занятиях	2	2	0	опрос
2	Что такое Arduino?	2	0	2	Педагогическое наблюдение
3	Написание программного кода и схема техника**.** Кейс «Светофор»	8	2	6	
Базовый уровень					
4	Массивы и пьезоэлементы. Кейс «Маячок»	4	1	3	Педагогическое наблюдение
5	Кейс «Перетягивание каната»	2	0	2	Педагогическое наблюдение
6	Кейс «Азбука Морзе»	2	0	2	Педагогическое наблюдение
7	Шин и смешание цветов. Решение кейсов	4	0	4	Педагогическое наблюдение
Продвинутый уровень					
7	Сенсоры и датчики нажатия. Решение кейсов	4	1	3	Педагогическое наблюдение
8	Кейс «Кнопочные ковбои»	2	0	2	Педагогическое наблюдение
9.	Кейс «Секундомер»	2	0	2	
10.	Презентация проектных работ	4	0	4	зачет
	Итого	**36**	**6**	**30**	

Содержание программы учебно-тематического плана для учащихся 14-17 лет

Стартовый уровень

1. Вводное занятие. Правила техники безопасности и поведения на занятиях

Правила обращения с оборудованием. Изучаются основные понятия: Кванториум, наставник, форма подачи информации. Игра “Снежный ком” на знакомство с обучающимися.

Кол-во часов /занятий: 2/1

2. Что такое Arduino?

Деление на команды. Дата-скаутинг на тему Arduino (основные понятия, сфера применения, как работать с микроконтроллером). Подготовка презентации и защита (командная работа). Практическое занятие для освоения полученных знаний. Понятия программируемая среда, простейшие команды.

Кол-во часов /занятий: 2/1

3.Написание программного кода и схема техника. Кейс «Светофор»

Дети учатся писать программный код, и разбираются с схемами, деталями и техникой работы. В кейсе «Светофор» дети учатся собирать светофор, в котором должны меняться цвета.

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 8/4

Базовый уровень

4.Массивы и пьезоэлементы. Кейс «Маячок»

Теорияю Массивы и пьезоэлементы.Решение кейса «Маячок»В этом задании дети учатся собирать маячок, который мигает светодиодом.

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 4/2

5.Кейс «Перетягивание каната» В этом задании учащиеся создают еще одну игру, на этот раз нужно быстрее соперника нажать кнопку 20 раз.

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 2/1

6.Кейс «Азбука Морзе» В этом задании дети учатся собирать азбуку Морзе, и как ей пользоваться.

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 2/1

7.Шин и смешание цветов. Решение кейсов.

Кейс «Пульсар» В этом задании учащиеся плавно наращивают яркость светодиодной шкалы, управляя большой нагрузкой через транзистор.

Кейс «Ночной светильник» В этом задании светодиод должен включаться при падении уровня освещенности ниже порога, заданного потенциометром.

Кол-во часов /занятий: При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

4/2

Углубленный уровень

7. Сенсоры и датчики нажатия.Решение кейсов.

Кейс «Кнопочный переключатель» В этом задании учащиеся делают из тактовой кнопки триггер, борясь с «дребезгом».

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.

Кол-во часов /занятий: 4/2

8.Кейс «Кнопочные ковбои» В этом задании учащиеся создают игрушку на реакцию: кто быстрее нажмет кнопку по сигналу

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.
Кол-во часов /занятий: 2/1

9.Кейс «Секундомер» В этом задании учащиеся создают секундомер, который считает до 10

При решении кейса предлагается следующее распределение участников в группе: участники работают в малых группах на всех этапах выполнения кейса по 3-4 человека.
Кол-во часов /занятий: 2/1

10. Презентация проектных работ

Презентация проектных работ обучающихся. Выделенные микрокоманды в начале модуля работают над проектами по выбранной тематике, определяя её актуальность и пути решения.
Кол-во часов /занятий: 4/2

VI. Организационно-педагогические условия

6.1.Методическое обеспечение программы - описание методов и приемов работы в соответствии с особенностями содержания и обучающихся; Согласно методических рекомендаций по созданию Кванториумов основным форматом образовательного процесса является проектная деятельность. Реализация Программы строится на принципах: «от простого к сложному». Обучение проводится с использованием программного обеспечения Blender. Занятия по Программе состоят из теоретической и практической частей Теоретическая часть проходит в виде лекций. Лекции сочетаются с дискуссиями и беседами, завершаются первичным закреплением материала; Практическая часть включает в себя выполнение практических заданий. Каждая тема сопровождается практической работой, что способствует лучшему усвоению теоретического материала и дает базовые навыки работы с программой Blender. В конце курса обучающиеся выполняют итоговую практическую (творческую) работу и представляют ее на конкурс. При обучении используются дидактические материалы и уроки сайта

При организации дистанционного обучения

Под дистанционными образовательными технологиями (далее - ДОТ) понимаются образовательные технологии, реализуемые в основном с применением информационно-телекоммуникационных сетей при опосредованном (на расстоянии) взаимодействии учащихся и педагогических работников. Образовательный процесс, реализуемый в дистанционной форме, предусматривает значительную долю самостоятельных занятий учащихся.

Дистанционное обучение, осуществляемое с помощью компьютерных телекоммуникаций, имеет следующие формы занятий:

Чат - занятия — учебные занятия, осуществляемые с использованием чат - технологий. Чат - занятия проводятся синхронно, то есть все участники имеют одновременный доступ к чату.

Форум-занятия — дистанционные уроки, конференции, семинары, деловые игры, лабораторные работы, практикумы и другие формы учебных занятий, проводимых с помощью средств телекоммуникаций и других возможностей сети Интернет. Для таких занятий используются разработанные образовательные форумы — форма работы учащихся по определённым темам внутри одного пространства. От чат - занятий форумы отличаются возможностью многодневной работы и несинхронным взаимодействием учащихся и педагога.

Для данных видов занятий используется сервисы Zoom и Discord.

Все методические материалы, рабочие тетради, практические задания, готовые проекты и кейсы расположены в Web-сервисе Trello. Базовым форматом образовательного процесса является проектная деятельность. В ходе работы над проектом реализовываются проекты в формате законченных научных исследований или инженерных разработок в виде выполненного продукта и проекты выносятся на защиту в конце обучения по очной и заочной форме.

При проведении занятий с применением дистанционных образовательных технологий учитываются режимы работы с ПЭВМ, согласно СанПиН 2.4.4.317214 и рекомендациям Роспотребнадзора для обучающихся в возрасте 12-13 лет не более 20 минут непрерывной работы, для обучающихся в возрасте 14-18 лет не более 30 минут непрерывной работы. Работу с гаджетами чередуют с другими видами деятельности - чтением или письмом

6.2 Материально-техническое обеспечение программы

№	Наименование разделов и тем	Кол-во часов	Список необходимого оборудования и расходных материалов	
			Наименование	**Кол-во**
Стартовый уровень				
1	Вводное занятие. Правила техники безопасности и поведения на занятиях	**2**	Ноутбук Raybook /Россия 1 шт	1 шт
			Проектор ACER H651/Китай	1 шт
			Экран CACTUS Triscreen CS-PST-124x221/Китай	1 шт
			Компьютер RAY /Россия 1	1 шт
			Монитор BENQ EW2775ZH/Китай	1 шт
			Матрёшка Z/Россия	13 шт
2	Написание программного кода и схема техника. Электронные компоненты	**8**	Матрёшка Z/Россия	13 шт
			Малина v4/Россия	13 шт
			AR-DEK-STR-01 Конструктор программируемых моделей инженерных систем. Базовый комплект/Россия	12 шт
			Знаток Arduino BASIC 5541547/Россия	13 шт
			Матрёшка Z/Россия	13 шт
			Малина v4/Россия	13 шт
			Iskra Mega/Китай	7 шт
			Arduino MKR Vidor 4000/Китай	7 шт.
Базовый уровень				
2	Массивы и пьезоэлементы	**6**	AR-DEK-STR-01 Конструктор программируемых моделей инженерных систем. Базовый комплект/Россия	12 шт
			Знаток Arduino BASIC 5541547/Россия	13 шт
			Малина v4/Россия	13 шт
			Iskra Mega/Китай	7 шт
			Датчик акселерометр/Россия	7 шт
3	ШИМ и смешание цветов	**6**	Знаток Arduino BASIC 5541547/Россия	13 шт
			Матрёшка Z/Россия	13 шт
			Iskra Mega/Китай	7 шт
			Датчик влажности почвы/Китай	7 шт
Продвинутый уровень				
4	Сенсоры и датчики нажатия.	**12**	AR-DEK-STR-01 Конструктор программируемых моделей инженерных систем. Базовый комплект/Россия	12 шт
			Матрёшка Z/Россия	13 шт

			Малина v4/Россия	13 шт
			Iskra Mega/Китай	7 шт
			Датчик ИК-приемник/Россия	7 шт.
			Датчик движения инфракрасный/Россия	7 шт
			Датчик клавиатура 4х3 кнопки/Китай	7 шт
			Датчик кнопка	7 шт
			Датчик потенциометр/Россия	7 шт
			Датчик термистор/Китай	7 шт
			Датчик фоторезистор/Китай	7 шт
			Датчик ультразвуковой дальномер/Китай	7 шт
			Датчик температуры и влажности/Россия	7 шт
			Плата расширения для подключения большого количества периферии/Россия	7 шт
			Модуль мини-реле/Россия	7 шт
			Модуль силовой ключ/Россия	7 шт
			Четырехразрядный индикатор/Россия	7 шт
			Модуль зуммер/Россия	7 шт
			Повышающий стабилизатор напряжения/Россия	7 шт
			Часы реального времени/Россия	7 шт
			Модуль Bluetooth /Китай	7 шт
			Модуль ИК-передатчик /Россия	7 шт
			Модуль Wi-Fi /Россия	7 шт
			Сервопривод/Китай	7 шт
			Погружная помпа с трубкой/Китай	7 шт
			Макетная плата /Китай	7 шт
			Блок питания/Китай	7 шт
			Модуль USB /Россия	7 шт
	Итого	**36**		

Карта образовательного модуля

Метод: Кейс 1.

Название: «Светофор»

Кол-во часов/занятий: 4/2.

Hard Skills: составление алгоритма программы и его формализация в виде блок-схемы; написание кода программы согласно алгоритму;

программирование микроконтроллерных платформ на языке С в Arduino IDE; получение и обработка показаний цифровых и аналоговых датчиков.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы навыки публичных выступлений.

Артефакт: прототип светофора.

Место проведения: IT-квантум.

Метод: Кейс 2.

Название: «Маячок».

Кол-во часов/занятий: 4/2.

Hard Skills: умение составлять алгоритм для решения конкретной проблемной задачи и писать программы на языке С в Arduino IDE; использование внешних библиотек; умение применять знания основ схемотехники на практике.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы; навыки публичных выступлений; умение осуществлять межквантумное взаимодействие.

Артефакт: Проблесковый маячок, также устройство для подачи специальных световых сигналов, спецсигнал, мигалка — устройство.

Место проведения: IT-квантум.

Метод: Кейс 3.

Название: «Перетягивание каната».

Кол-во часов/занятий: 4/12.

Hard Skills: умение применять комплектующие для создания умных вещей; умение разрабатывать мобильные приложения при помощи MIT App Inventor; умение налаживать взаимодействие между устройствами разных типов.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы; навыки публичных выступлений.

Артефакт: Создания игры.

Место проведения: IT-квантум.

Метод: Кейс 4.

Название: «Азбука Морзе».

Кол-во часов/занятий: 4/2.

Hard Skills: создание веб-страницы для выдачи показаний метеостанции; применение различных протоколов обмена информацией; обработка и хранение данных.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы; навыки публичных выступлений.

Артефакт: прототип переводчика.

Место проведения: IT-квантум.

Метод: Кейс 5.

Название: «Пульсар».

Кол-во часов/занятий: 4/2.

Hard Skills: создани управляемого светодиода;упровления напряжением.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы; навыки публичных выступлений.

Артефакт: прототип настройки уровня горения лампы.

Место проведения: IT-квантум.

Метод: Кейс 6.

Название: «Кнопочные ковбои»

Кол-во часов/занятий: 4/2.

Hard Skills: умение применять комплектующие для создания умных вещей; умение разрабатывать мобильные приложения при помощи MIT App Inventor; умение налаживать взаимодействие между устройствами разных типов.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы; навыки публичных выступлений.

Артефакт: Создания игры.

Место проведения: IT-квантум.

Метод: Кейс 7.

Название: «Секундомер»

Кол-во часов/занятий: 4/2.

Hard Skills: составление алгоритма программы и его формализация в виде блок-схемы; написание кода программы согласно алгоритму; программирование микроконтроллерных платформ на языке С в Arduino IDE; получение и обработка показаний цифровых и аналоговых датчиков.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы навыки публичных выступлений.

Артефакт: прототип секундамера.

Место проведения: IT-квантум.

Метод: Кейс 8.

Название: «Кнопочные переключатель »

Кол-во часов/занятий: 4/2.

Hard Skills: умение применять комплектующие для создания умных вещей; умение разрабатывать мобильные приложения при помощи MIT App Inventor; умение налаживать взаимодействие между устройствами разных типов.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы; навыки публичных выступлений.

Артефакт: Создания рабочего переключателя.

Место проведения: IT-квантум.

Метод: Кейс 8.

Название: «Ночной светильник»

Кол-во часов/занятий: 4/2.

Hard Skills: умение применять комплектующие для создания умных вещей; умение разрабатывать мобильные приложения при помощи MIT App Inventor; умение налаживать взаимодействие между устройствами разных типов.

Soft Skills: умение осуществлять поиск и анализ информации; навыки формулирования проблемы; выдвижение гипотезы; умение ставить вопросы; навыки публичных выступлений.

Артефакт: Создания рабочего переключателя.

Место проведения: IT-квантум.

Кейсы, входящие в программу

Ночной светильник

В этом эксперименте светодиод должен включаться при падении уровня освещенности ниже порога, заданного потенциометром.

Список деталей для эксперимента

- 1 плата Arduino Uno
- 1 беспаечная макетная плата
- 1 светодиод
- 1 фоторезистор
- 1 резистор номиналом 220 Ом
- 1 резистор номиналом 10 кОм
- 1 переменный резистор (потенциометр)
- 10 проводов «папа-папа»

Для дополнительного задания

- еще 1 светодиод
- еще 1 резистор номиналом 220 Ом
- еще 2 провода

Принципиальная схема

Схема на макетке

Обратите внимание

- В этом эксперименте мы устанавливаем фоторезистор между питанием и аналоговым входом, т.е. в позицию R1 в схеме делителя напряжения. Это нам нужно для того, чтобы при уменьшении освещенности мы получали меньшее напряжение на аналоговом входе.
- Постарайтесь разместить компоненты так, чтобы светодиод не засвечивал фоторезистор.

Программный код

p050_night_light.ino

```
#define LED_PIN  13

#define LDR_PIN  A0
```

```
#define POT_PIN  A1

void setup()

{  pinMode(LED_PIN, OUTPUT);}

void loop(){  // считываем уровень освещённости. Кстати, объявлят

 // переменную и присваивать ей значение можно разом  int lightness = analogRead(LDR_PIN);
// считываем значение с потенциометра, которым мы регулируем

// пороговое значение между условными темнотой и светом  int threshold =
analogRead(POT_PIN);

// объявляем логическую переменную и назначаем ей значение

// «темно ли сейчас». Логические переменные, в отличие от

// целочисленных, могут содержать лишь одно из двух значений:

// истину (англ. true) или ложь (англ. false). Такие значения

// ещё называют булевыми (англ. boolean).  boolean tooDark = (lightness < threshold);

 // используем ветвление программы: процессор исполнит один из

// двух блоков кода в зависимости от исполнения условия.

// Если (англ. «if») слишком темно...  if (tooDark) {

// ...включаем освещение    digitalWrite(LED_PIN, HIGH);  } else {

// ...иначе свет не нужен — выключаем его    digitalWrite(LED_PIN, LOW);  }}
```

Пояснения к коду

- Мы используем новый тип переменных — boolean, которые хранят только значения true (истина, 1) или false (ложь, 0). Эти значения являются результатом вычисления логических выражений. В данном примере логическое выражение — это lightness < threshold. На человеческом языке это звучит как: «освещенность ниже порогового уровня». Такое высказывание будет истинным, когда освещенность ниже порогового уровня. Микроконтроллер может сравнить значения переменных lightness и threshold, которые, в свою очередь, являются результатами измерений, и вычислить истинность логического выражения.
- Мы взяли это логическое выражение в скобки только для наглядности. Всегда лучше писать читабельный код. В других случаях скобки могут влиять на порядок действий, как в обычной арифметике.
- В нашем эксперименте логическое выражение будет истинным, когда значение lightness меньше значения threshold, потому что мы использовали оператор <. Мы можем использовать операторы >, <=, >=, ==, !=, которые значат «больше», «меньше или равно», «больше или равно», «равно», «не равно» соответственно.
- Будьте особенно внимательны с логическим оператором == и не путайте его с оператором присваивания =. В первом случае мы сравниваем значения выражений и получаем логическое значение (истина или ложь), а во втором случае присваиваем

левому операнду значение правого. Компилятор не знает наших намерений и ошибку не выдаст, а мы можем нечаянно изменить значение какой-нибудь переменной и затем долго разыскивать ошибку.

- Условный оператор if («если») — один из ключевых в большинстве языков программирования. С его помощью мы можем выполнять не только жестко заданную последовательность действий, но принимать решения, по какой ветви алгоритма идти, в зависимости от неких условий.
- У логического выражения lightness < threshold есть значение: true или false. Мы вычислили его и поместили в булеву переменную tooDark («слишком темно»). Таким образом мы как бы говорим «если слишком темно, то включить светодиод»
- С таким же успехом мы могли бы сказать «если освещенность меньше порогового уровня, то включить светодиод», т.е. передать в if всё логическое выражение:

```
if (lightness < threshold) {  // ...}
```

- За условным оператором if обязательно следует блок кода, который выполняется в случае истинности логического выражения. Не забывайте про обе фигурные скобки {}!
- Если в случае истинности выражения нам нужно выполнить только *одну* инструкцию, ее можно написать сразу после if (…) без фигурных скобок:

```
if (lightness < threshold)  digitalWrite(LED_PIN, HIGH);
```

- Оператор if может быть расширен конструкцией else («иначе»). Блок кода или единственная инструкция, следующий за ней, будет выполнен только если логическое выражение в if имеет значение false, «ложь». Правила, касающиеся фигурных скобок, такие же. В нашем эксперименте мы написали «если слишком темно, включить светодиод, иначе выключить светодиод».

Вопросы для проверки себя

4. Если мы установим фоторезистор между аналоговым входом и землей, наше устройство будет работать наоборот: светодиод будет включаться при увеличении количества света. Почему?
5. Какой результат работы устройства мы получим, если свет от светодиода будет падать на фоторезистор?
6. Если мы все же установили фоторезистор так, как сказано в предыдущем вопросе, как нам нужно изменить программу, чтобы устройство работало верно?
7. Допустим, у нас есть код if (условие) {действие;}. В каких случаях будет выполнено действие?
8. При каких значениях y выражение x + y > 0 будет истинным, если x > 0?
9. Обязательно ли указывать, какие инструкции выполнять, если условие в операторе if ложно?
10. Чем отличается оператор == от оператора =?
11. Если мы используем конструкцию if (условие) действие1; else действие2;, может ли быть ситуация, когда ни одно из действий не выполнится? Почему?

Задания для самостоятельного решения

12. Перепишите программу без использования переменной tooDark с сохранением функционала устройства.

13. Добавьте в схему еще один светодиод. Дополните программу так, чтобы при падении освещенности ниже порогового значения включался один светодиод, а при падении освещенности ниже половины от порогового значения включались оба светодиода.
14. Измените схему и программу так, чтобы светодиоды включались по прежнему принципу, но светились тем сильнее, чем меньше света падает на фоторезистор.

Секундомер

В этом эксперименте мы создаем секундомер, который считает до 10.

Список деталей для эксперимента

- 1 плата Arduino Uno
- 1 беспаечная макетная плата
- 1 семисегментный индикатор
- 7 резисторов номиналом 220 Ом
- 9 проводов «папа-папа»

Принципиальная схема

Схема на макетке

Обратите внимание

- Выводы 3 и 8 семисегментного индикатора оба являются катодами, к земле можете подключать любой из них.
- Внимательно рассмотрите схему, сопоставьте сегменты индикатора с номерами его ножек, а те, в свою очередь, с пинами Arduino, к которым мы их подключаем.
- Вывод 5 индикатора — это точка. Мы не используем её в этом эксперименте
- Сегменты индикатора — просто светодиоды, поэтому мы используем резистор с каждым из них.

Программный код

p130_seven_segment_counter.ino

```
#define FIRST_SEGMENT_PIN

#define SEGMENT_COUNT

// префикс «0b» означает, что целое число за ним записано в

// в двоичном коде. Единицами мы обозначим номера сегментов

// индикатора, которые должны быть включены для отображения

// арабской цифры. Всего цифр 10, поэтому в массиве 10 чисел.

// Нам достаточно всего байта (англ. byte, 8 бит) для хранения
```

```
// комбинации сегментов для каждой из цифр.byte numberSegments[10] = {  0b00111111, 0b00001010, 0b01011101, 0b01011110, 0b01101010,  0b01110110, 0b01110111, 0b00011010, 0b01111111, 0b01111110,}; void setup(){  for (int i = 0; i < SEGMENT_COUNT; ++i)    pinMode(i + FIRST_SEGMENT_PIN, OUTPUT);} void loop(){

// определяем число, которое собираемся отображать. Пусть им

// будет номер текущей секунды, зацикленный на десятке  int number = (millis() / 1000) % 10;

// получаем код, в котором зашифрована арабская цифра  int mask = numberSegments[number]; // для каждого из 7 сегментов индикатора...  for (int i = 0; i < SEGMENT_COUNT; ++i) {    // ...определяем: должен ли он быть включён. Для этого

// считываем бит (англ. read bit), соответствующий текущему

// сегменту «i». Истина — он установлен (1), ложь — нет (0)    boolean enableSegment = bitRead(mask, i);    // включаем/выключаем сегмент на основе полученного значения digitalWrite(i + FIRST_SEGMENT_PIN, enableSegment);  }}
```

Пояснения к коду

- Мы создали массив типа byte: каждый его элемент это 1 байт, 8 бит, может принимать значения от 0 до 255.
- Символы арабских цифр закодированы состоянием пинов, которые соединены с выводами соответствующих сегментов: 0, если сегмент должен быть выключен, и 1, если включен.
- В переменную mask мы помещаем тот элемент массива numberSegments, который соответствует текущей секунде, вычисленной в предыдущей инструкции.
- В цикле for мы пробегаем по всем сегментам, извлекая с помощью встроенной функции bitRead нужное состояние для текущего пина, в которое его и приводим с помощью digitalWrite и переменной enableSegment
- bitRead(x, n) возвращает boolean значение: **n**-ный бит *справа* в байте **x**

Вопросы для проверки себя

15. К которой ножке нашего семисегментного индикатора нужно подключать землю?
16. Как мы храним закодированные символы цифр?
17. Каким образом мы выводим символ на индикатор?

Задания для самостоятельного решения

18. Измените код, чтобы индикатор отсчитывал десятые секунды.
19. Поменяйте программу так, чтобы вместо символа «0» отображался символ «A».
20. Дополните схему и программу таким образом, чтобы сегмент-точка включался при прохождении четных чисел и выключался на нечетных

Светофор

Виджет из светофора: мониторим Travis CI на Iskra JS

- Платформы: Iskra JS
- Языки программирования: JavaScript
- Тэги: Continuous Integration, непрерывная разработка, Travis CI, автоматическое тестирование, виджет

Что это?

Однажды нам в руки попал списанный светофор. Приспособить его для отображения пробок? Сделать индикатор занятости переговорной комнаты? Прицепить к гаражному парктронику? Нет, это всё слишком просто и банально.

Мы решили помочь нашим программистам продемонстрировать усердную и кропотливую работу всей Амперке. Разработку программного обеспечения мы ведём на GitHub-е, а обеспечить непрерывную интеграцию нам помогает сервис Travis CI. Именно к нему мы и подключили светофор, превратив его в виджет, который сообщает как прошла последняя сборка и стоит ли заводить ветку hotfix в github.

Что нам понадобится?

Iskra JS

Ethernet Shield

Troyka Slot Shield

Контактные колодки Arduino

3 × Мини-реле (Troyka-модуль)

Патч-корд витой пары

Кабель USB (A — Micro USB)

Импульсный блок питания с USB-разъёмами (5 В, 3100 мА)

Светофор

Как собрать?

Установите Ethernet Shield на Iskra JS

.

Эти платы уже позволят взаимодействовать с API travis-ci. Travis-ci — это хостинг непрерывной интеграции для open source сообщества. Он используется для проведения сборки и тестирования исходного кода проекта.

Подключите Iskra JS кабелем micro-USB к компьютеру. Подключите Ethernet Shield к своей локальной сети через патч-корд витой пары.

21.

Настроим связь с Интернетом — напишем простой GET-запрос по URL. Загрузите код в Iskra JS. Устройство один раз сделает запрос к DHCP-серверу для получения IP-адреса, затем будет каждые 5 секунд отправлять запрос на URL-адрес example.com. В ответ придёт html-код запрашиваемой страницы.

get-request.js

```
// Настраиваем соединение с Ethernet Shield по протоколу SPI.SPI2.setup({baud: 3200000, mosi: B15, miso: B14, sck: B13});var eth = require('WIZnet').connect(SPI2, P10);

// Подключаем модуль http.var http = require('http'); // Получаем IP-адрес от DHCP-сервераeth.setIP();

// Задаём опции подключения к запрашиваемому серверуvar options = {  host : 'google.ru',  port : '80',  path :'/',  protocol : 'http:',  headers : {    'Accept': 'text/html'  }}; var request = function () {  var response = '';  http.get(options, function(res)  {    // Большие ответы от сервера могут приходить по частям.

// В переменной response собираем весь ответ целиком.    res.on('data', function(data) { response += data;    });

// После закрытия соединения обрабатываем весь пришедший ответ    res.on('close', function() { if (response === undefined) {        print('Error. Response is undefined.');      } else {  print(response);      }      setTimeout(request, 5000);    });  });};

// Производим запросrequest();
```

Подключимся к сервису travis-ci. Будем использовать функцию branches API, при вызове которой сервис travis-ci отдаёт информацию об указанной ветке в репозитории. Для начала произведём запрос через адресную строку браузера. Введите URL https://api.travis-ci.org/repos/niggor/buildMonitor/branches/master. В ответ travis-ci вернёт строку в формате JSON. С помощью онлайн редактора JSON можно посмотреть эту строку в понятном виде.


```
{  "branch": {    "id": 151571537,    "repository_id": 9173666,    "commit_id": 43049835,    "number": "8",    "config": {      "language": "node_js",      "node_js": [        "5"      ],      ".result": "configured",      "group": "stable",      "dist": "precise"    },    "state": "passed",    "started_at": "2016-08-11T17:10:50Z",    "finished_at": "2016-08-11T17:11:32Z",    "duration": 42,    "job_ids": [      151571538    ],    "pull_request": false  }}
```


В этом ответе нам интересны два поля: number и state. Поле number сообщает номер последней сборки. По нему виджет будет определять наличие новой сборки. Если новой сборки нет, сигнализировать о старых не стоит. Поле state отвечает за статус сборки и может принимать 4 значения: created, started, passed, failed.

Сделаем то же самое, но с помощью Iskra JS.

Travis-ci предоставляет API только по защищённому соединению https. Если у вас появляются ошибки TLS соединения, обновите прошивку Iskra JS.

get-travis-ci.js

// Настраиваем соединение с Ethernet Shield по протоколу SPI.SPI2.setup({baud: 3200000, mosi: B15, miso: B14, sck: B13});var eth = require('WIZnet').connect(SPI2, P10);

// Подключаем модуль http.var http = require('http');

// Получаем IP-адрес от DHCP-сервераeth.setIP();

// Задаём репозиторий в github и ветку, которую хотим отслеживатьvar repo = 'niggor/buildMonitor';var branch = 'master';

// Задаём опции подключения к запрашиваемому серверу

// Поля key, ca и cert необходимы для установления шифрованного соединения httpsvar options = { key : atob('MIIJKQIBAAKCAgEApB4OJT9Y9n05aDagDqXvohgEySccfWtxwQUQQNpVuv0+KCiRXFqLPkzIswK5ZLA43qDKTdnfvpScC1GeR3BxAda/uGy+BbpUozKI1CTQ/+Z+B1QWbMweRARmrctNSAhYcelCsYfBU7DprWXp63ZoNPCxtNl0y4QEY6PfUpnG/dStu4iyaqqseF1MVWMl2xpsHWrKhxvu3sASrOwR3KazKd4gkPqfTPLEDz4p8YlzLuovtOlXU1WxX9nOUYd1xv5U7rIw02OBIpZtyICHxH2KX5aIlMnkgR7fMgNXS4jwUywApYpe/+gPvD09+QmcLqpNzkTiyAc+wycE/aLH/2QEZXEBLEaP7l6vwzI8CwZ4tzsBKWB4whiJ7JJjeSYfB6U6keACmGCixMNuSG9giWIzVIrJgIztSJ2MnjMEWE1Y8CUOc9Px41OFM0dw8t2UfcqgQLbY4X0wQ0SHpOGmnP9uNE/mcSGpavk8sM9pgGA8GS+BicJPwJVex2iZsLei/nO8fZDqfI9xuMvPrJOqTEuY8zQaJvZvfrBYNLEAKZKzfuLKOJ1zjSybj6TZseJsI4sObrNrMs8n52rDuw+uWq8J76rnN/zqdCBysE1U2lQtjLy79EHA+/lUyUyS5AyxvEPdVwA+di0Qzb/jUs0ysGbs3PqJwMTC5FaUejDF7n7FbBsCAwEAAQKCAgAe/9pWKMgKDw834xGKksdXy8ejozKbch8TgXxXc6Y74rfJ6BfE5QlmJCTIw1v3a93iF2b/G4dKvVFYRftqrQ97dG/USx39gPuZs+Lqh2W+4G+vvK9br5aL7HTkoQEg+QqTwaaqV+UphMIu0ZBEvKHkVg6LBYQtKOuXUo3NUWSn73M9eA2TXlejiKPxgP/WigAf8lr8+Ea/ZnJgnVqmXgp7fwyRtfU/GJ/17R0JFZhfJTcLqgOnpwkQ/NgeyEJz6vhp+diFJUdwHA/yjmyWrcvkUE/OIvB5fzZTgRQNHXmfjDE3E5LsLY5DVSIOl0Ub/2kg7BwWAmBVhlpH+mZilCbMIOChD5r9LjHIPjr4jKwPpim3plgtnLca52x+HmVXwkFoUl5s3ybvAn0q4OZpjOxZjUrsmhgh/ibzQdvpCNRdESdO8QsBJJf4L35pAo58lYceCQI4vqzFqzOnAIS9we0MUstadsGccUs6n8wVdn9zYfcF3fGPJ4PmNF4a+n9a/vh/XO06T2DUBfrtAOzwB2xeUqR+T90UpBbAM4D4320NynvhX0758fCReFT0Zc/P+yR6+xHzJ4jpqszoIPvfKxvZmSSS/L5oNcaEDzweM0+U5iV4GbJHQlldzCNzPJoOhf1vsqdVhCSiMgScGNuU3V9lYkMesH28STZc8MooiGt7QQKCAQEA2ZuSfGlB4kNsh9R26uwLstg61C2bJbM5oAX5SvTMaGxeRkw192d1T86F5bCyvwZs8AjGxSH5t//cs8snsXHDvIgCh28P5t4b9/+2M/DEDzxWkfgswVVU9JFElgfqioxuq+fc1X+ZLEYxibzNqNDI+Y019us2S6R7b3oPzWXhdY4sbLdTHt9vSi/CwxMCCqVLjnZHWic7Z/ZIcXOc97I4KyYK1sLusLa85O+FELoKk+SN7rXOSIqiqFqVWIXSt8hHeHg37Lg15YBFH8f16Bte9G3V0qDyIKVqlcs/dwzJOnpTxOZgdSy5ChDPS901OyIjHguL6kXQRvlaTfxB4788YQKCAQEAwRKM5qsCvUGf+gTV0OGjZmGun9G4EEsIbXyh8HoIdI5JsQY0d+sr2jGxP3Bh3sHQGTkWKI/u/yx7KuglfQo6gGHgx0pYLchZb5u+uJTACgXES/U/G2hbB1OznMDqxo1PGuQ6Nl1pmFv5/OLqinuh6Nmsjfa59Ev5ScpgsMhCj+Ifz3X+zirGRhuvQGjyWwAmpNEU0C5xinKVx3SimdkxjOb1RiJ6mwcPp6XqVlA49Wq6f6ircuecE7vpL9PCiFPclH7WrPbu6wDGLQeZd+JHeQ0TGRErmqxD0b3fxSlMyaa52039SQAtoxcQ7WT/PFSlrzv9UIW05E8Ukhd3Z3fZ+wKCAQEAxvDGpDzqgbvYXfms2isAqvmqFs6292l59kVf2togghsIuUX1VIDeLB8IJKw9TgzGzU+IwTLFhSVClZDWchAqvZKulTjBFfTbJwg9gBDOfYtRLdPwvrOPnqOsS2ljUFUA3OJ17y3fUC3AHkdg0cRG6Hbj7iCtm8v9GTZDeil5TKq1DJMxR5lHulAdw4/+HtNtVwnlQ5N4aRwssKPczkq9tRzrbdn6s83stXQQd/FNuFWMOL83nvQ3rqhZvvthYKmWRZnKrvzeMTnqeBqsQw266sX7ZLvFgHkaD2YoWGN1GKQ7L1jauGUxqfUiPx42CyCzcEVoHb7Dog9XMtK7PkZYIQKCAQEAjZmOTyqCwszbiGF9c0nZLiBcEwibhw+QqcPrfNPzWZB1iWh0mLKYodkqFs3UaawZ

dDZDAxK1BN7ZollKDU+wIjImyeLDkyhKN6mxEey+f1tFYTQHyZEameAQbdmrFl3S83UJIJAX/
s8OJCNEQrZzw7PJcwDN1tbViQHtMhpdfusny17NtuT2L7rkKEgP3x/YN67EJf659QEeyn7HNjBtv
9ovbvWgmAxwtdLgb9c8Pvm8uB27SHWHKY36csJK/PMMZrgzt/0OsroJnkLhkmQZaTVmmU7Xf
QP0ZWJtOEuW7pnh225yYdTIo8EqAAG/k769GSzVFvZI+dzpyE9/mhnzewKCAQB6ZeaPI+k7Cs
Nfyt9j4HX9JKi5ZycsBUdBJ+kJ7xOd1Fa0Y/OcnQh6vLeC5oqtJLB4mLnRvutyK6NP3sI6dLNvsqj
VNexl5pSI0/ZlNxNiinFcchl19pnA4zniaLGLh+IO72OQrwMrEhTSuSBZNcR71+roACIVv5t1A57P
lXr+bqv/JIKMwiocFeoWDh9NfuhvA5ecv/Q1OS0rsvK1tSFW+dKzMQgE1P8yB9DmyVZ4F16Oyf
5dERcRzYLd+hFs4eQ+LpP11sMhF8TIN1optRT4+NdY+ZdJbhBYEriFOPVUcfn4bdkmGezmy9U
Fg+IArnHB8bXZowpFwfciEnZuwWNE'), cert :
atob('MIIFijCCA3KgAwIBAgIJANLNn7E5scNGMA0GCSqGSIb3DQEBBQUAMIGBMQswCQY
DVQQGEwJVUzELMAkGA1UECAwCTUExDzANBgNVBAcMBkJvc3RvbjETMBEGA1UECgw
KRXhhbXBsZSBDbzEQMA4GA1UECwwHdGVjaG9wczELMAkGA1UEAwwCY2ExIDAeBgkqh
kiG9w0BCQEWEWNlcnRzQGV4YW1wbGUuY29tMB4XDTE2MDgxMDE0MDEyNFoXDTE5M
DUwNjE0MDEyNFowgYYxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJNQTEPMA0GA1UE
BwwGQm9zdG9uMRMwEQYDVQQKDApFeGFtcGxlIENvMRAwDgYDVQQLDAd0ZWNob3Bz
MRAwDgYDVQQDDAdjbGllbnQxMSAwHgYJKoZIhvcNAQkBFhFjZXJ0c0BleGFtcGxlLmNvbT
CCAiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoCggIBAKQeDiU/WPZ9OWg2oA6l76IYBM
knHH1rccEFEEDaVbr9PigokVxaiz5MyLMCuWSwON6gyk3Z376UnAtRnkdwcQHWv7hsvgW6V
KMyiNQk0P/mfgdUFmzMHkQEZq3LTUgIWHHpQrGHwVOw6a1l6et2aDTwsbTZdMuEBGOj31
KZxv3UrbuIsmqqrHhdTFVjJdsabB1qyocb7t7AEqzsEdymsyneIJD6n0zyxA8+KfGJcy7qL7TpV1N
VsV/ZzlGHdcb+VO6yMNNjgSKWbciAh8R9il+WiJTJ5IEe3zIDV0uI8FMsAKWKXv/oD7w9PfkJn
C6qTc5E4sgHPsMnBP2ix/9kBGVxASxGj+5er8MyPAsGeLc7ASlgeMIYieySY3kmHwelOpHgAph
gosTDbkhvYIliM1SKyYCM7UidjJ4zBFhNWPAlDnPT8eNThTNHcPLdlH3KoEC22OF9MENEh6
Thppz/bjRP5nEhqWr5PLDPaYBgPBkvgYnCT8CVXsdombC3ov5zvH2Q6nyPcbjLz6yTqkxLmPM
0Gib2b36wWDSxACmSs37iyjidc40sm4+k2bHibCOLDm6zazLPJ+dqw7sPrlqvCe+q5zf86nQgcrBN
VNpULYy8u/RBwPv5VMlMkuQMsbxD3VcAPnYtEM2/41LNMrBm7Nz6icDEwuRWlHowxe5+x
WwbAgMBAAEwDQYJKoZIhvcNAQEFBQADggIBAIq0fsXM/NWlqHecu6EQM18x5JQIeCDrru
wrRe9Gj7jEdMka1UzlHWLLnTd6fiyBo0R0Vcdvp96Ic20o5WzK2lx5srBz+B604/Oud682ZE6Cog
QZUSjw4kxP2GBDjnB8wytgebl5QaLLIHCZTS33WvmE6K2cquxMNmeG9CW4kIi/LKJpcBf7t3C
b5fbusosB32oyRt8BpKi3476A1hhcFRTse0P3RV1NrIyrwgxq0PY+uEtEz9BQRCT1fZohPKlEUM
WYzV3Wj8eboaWohW+Tonr2zuu1WhF3bxKe9vd0E3hWoe7bI5I010XlDxogFFokr6MVLzXV95q
0zuULu1NqcIAwgttzwe9fsXMd+k+rD9YNNW3fXu4WtqVoovwLIUYNDRQENIIbyigyhCUzIXo9
7Xn43vKxxTtyu546QrcfgX59hFjxK0yRxwh2EjuUG10bo2UCes6k6QzBvWrH8gNl4mWFyXaqXJJ
I4o0R5hTIRyVp/lquXYz6pdWNcuR/5SzD/1F0+Z6SJ4xPbxRZTVolPRJ7FFlJ5GaFvkinA4pDskYD
vFtfmcy0hdY/0nCJgvYUMtZfMtWYTfKL+4EfjFk1/yqZsNAv/gkP87QQwqA+tn1NJ0kfM0I9n28X
DLhdU92f0yHjCScy1o3ewcB7I0smFcUDD4OCYFUiNahxAUA0mwiP'), ca :
atob('MIIFgDCCA2gCCQDQNeSIxshM8DANBgkqhkiG9w0BAQUFADCBgTELMAkGA1UEBh
MCVVMxCzAJBgNVBAgMAk1BMQ8wDQYDVQQHDAZCb3N0b24xEzARBgNVBAoMCkV4
YW1wbGUgQ28xEDAOBgNVBAsMB3RlY2hvcHMxCzAJBgNVBAMMAmNhMSAwHgYJKoZI
hvcNAQkBFhFjZXJ0c0BleGFtcGxlLmNvbTAeFw0xNjA4MTAxMzUxMjZaFw0xOTA1MDYxMz
UxMjZaMIGBMQswCQYDVQQGEwJVUzELMAkGA1UECAwCTUExDzANBgNVBAcMBkJvc
3RvbjETMBEGA1UECgwKRXhhbXBsZSBDbzEQMA4GA1UECwwHdGVjaG9wczELMAkGA1
UEAwwCY2ExIDAeBgkqhkiG9w0BCQEWEWNlcnRzQGV4YW1wbGUuY29tMIICIjANBgkqhki
G9w0BAQEFAAOCAg8AMIICCgKCAgEAs0Tdt33C5DJlFbJBeqbbGDU+ipNNHJeTFU0WC28B
UNOPSMcZiWqc99LOCHXaW3M66PYJ9qRoHSoPG2dUBdCTFDeRgh32vVrtKGEgIeILJ19k4d0
s2Px8fyOgQQybekDR/o2PWxYxlI+SRS+0SvwJVPqEDzjCaQC7RVMD/r98VvK3uiHziSlQVWTA
RpbkYWU6PFh9S+rD4Mf6fjY6IKotebHKj5CuS+nv71mUv1l4DmGwTn9kf7ZtLgKEFP51JsuuQ4r
XuYWmK2HOn92dquKyXwY+rWwHbXmSW9Toxg4AtQutXd4A6AiNmgNj7Gaa69nnXe7ug4ws

MDSLqYWG+1z1FGomTTMngq5eXrURAE5yIPD6KhookZSowdzJKE13S6ijbR/qx8HjaBQD0bi8yrvjGOaXaQwU0NBPBsTBEg6xSQDPgx0nv+4YSLrF9CeDNdWIBPrjrc7wvv7xkG4U6n8ZNG9N9RkC+J6UJQ3C9N57X3xefvi8PbqlMom0ERK9phLjIzyNWqtImO18egS2caYhnccmuHeulFBlfZEoDggdlnL5i0I9QG3Fu9LCE1cziA18q6qbve+sweAH4vVSX/G9ImYfodKWitbDov8lgI+fqaPBK5O1qqgT6ub+NonVPIeicfGBy7VZzDTDImgt8ucA1vXiOSng7ZmLfm3fHD3g3ScCAwEAATANBgkqhkiG9w0BAQUFAAOCAgEAs0SAB49Ws0DudLG+xkFUvMXju+K3o8kJCFNXwzHe6djfTr5dKDwl+Tsx2maQR8JMZfuDcUBeQK/sMxF2Ps3RmQ1rA4OcRYZPwPJXs7LeREZ1mEKJwm4HwOcmnNbuIQ6vqd/TEhc5tpSUpxHytez7QERmncnch4PSJngJCAVgGBUr6sVTJCTBX3hhcsR6JXMVoaXS3OPPpMDwnH4ooS5MV1uZqa9zsubjlJUGEKKFN7Z0FCvPJXI4XC6pHbxQGz4/LeKSeKvBPHG6sTkwJuQFyHm7SXf0E0708tX1FKOCvQFPHAAakccoKDgVxYJhOZ23ZK2ooi43pb7Yu706wgUyZ9B1PPvRymKq2KMaoWvPFC72JuUlLhkIEBONLhsV6gc+rY9HBx6WP6DATRnm5o5HiNQ0gnEjlUnb+ou0BWkVRbIin+iowmYEXuw9/nze8mpC62jufqTu9Uu5NjMr911A4E4VUIHxk7BQ9rLdDppv49bSMU4ccVzXcjCbiP/fo1pPfPHCC96En2sXKGxSEv3HVyJJN2n5E/vVCp/9A7nXZIlGILMD1HHuTqUzs1H6qgtPKWHPLPxo0XGiYhoKMWviP41oPJqRJC1lvrXQV2Ae1+Giz1Ah6s8dR7ufZ0tI+mv2Rg4VOa+gPd2lI/bMhID2hq2DSeZ+am7HHGQrhl5R4OI='),

// задаем опции подключения к серверу host : 'api.travis-ci.org', port : '443',

// Собираем URL-адрес для вызова функции branch path :'/repos/'+repo+'/branches/'+branch, protocol : 'https:', headers : {   'Accept': 'application/vnd.travis-ci.2+json' }};

// функция запроса данных с сервера Travis CIvar request = function() {   var response = ''; http.get(options, function(res) {

// Большие ответы от сервера могут приходить по частям.

// В переменной response собираем весь ответ целиком.   res.on('data', function(data) { response += data;   });

// После закрытия соединения обрабатываем весь пришедший ответ   res.on('close', function() {

// Преобразуем ответ сервера из формата JSON в объект JavaScript   var answer = JSON.parse(response);

// проверяем результат преобразования на корректность   if (answer === undefined) { print('answer is undefined.');   } else {

// Получаем номер и статус последнего теста   var build = answer.branch.number;   var state = answer.branch.state;   print('Newest test:', build, 'is', state);   }   setTimeout(request, 5000); }); });}; request();

Установите Slot Shield сверху на Ethernet Shield.

Разъём RJ-45 на Ethernet Shield довольно высокий, для установки Slot Shield потребуются контактные колодки Arduino. Они приподнимут Slot Shield над Ethernet Shield.

Установите на Slot Shield реле. Подключите их к пинам P11, P12 и P13. Реле будут отображать состояние текущего теста. Реле на P12 будет говорит, что сборка в данный момент

тестируется (поле state в состоянии  ле на P11 — тесты провалены (поле state в состоянии failed), а реле на P ены (поле state в состоянии passed).

Подключите к реле нагрузку. Мы использовали дорожный светофор. Лампы светофора питаются от напряжения 220 вольт. Если под рукой нет светофора, вы можете взять любой бытовой прибор. Например, подключите чайник к реле на контакте P12 — чайник включится, как только система начнёт тестирование. Тестирование — процесс не быстрый, поэтому кипяток будет очень кстати.

Добавим в код управление реле в зависимости от принятого ответа.

ci-trafficlight.js

// Настраиваем соединение с Ethernet Shield 2SPI2.setup({baud: 3200000, mosi: B15, miso: B14, sck: B13});var eth = require('WIZnet').connect(SPI2, P10);

// Подключаем модуль http.var http = require('http');

// Подключаем релеvar relay = require('@amperka/relay');var red = relay.connect(P11);var yellow = relay.connect(P12);var green = relay.connect(P13);

// Задаём репозиторий в github и ветку, которую хотим отслеживатьvar repo = 'niggor/buildMonitor';var branch = 'master';var lastBuild = 0;

// Задаём опции подключения к запрашиваемому серверу

// Поля key, ca и cert необходимы для установления шифрованного соединения httpsvar options = { key : atob('MIIJKQIBAAKCAgEApB4OJT9Y9n05aDagDqXvohgEySccfWtxwQUQQNpVuv0+KCiRXFqLPkzIswK5ZLA43qDKTdnfvpScC1GeR3BxAda/uGy+BbpUozKI1CTQ/+Z+B1QWbMweRARmrctNSAhYcelCsYfBU7DprWXp63ZoNPCxtNl0y4QEY6PfUpnG/dStu4iyaqqseF1MVWMl2xpsHWrKhxvu3sASrOwR3KazKd4gkPqfTPLEDz4p8YlzLuovtOlXU1WxX9nOUYd1xv5U7rIw02OBIpZtyICHxH2KX5aIlMnkgR7fMgNXS4jwUywApYpe/+gPvD09+QmcLqpNzkTiyAc+wycE/aLH/2QEZXEBLEaP7l6vwzI8CwZ4tzsBKWB4whiJ7JJjeSYfB6U6keACmGCixMNuSG9giWIzVIrJgIztSJ2MnjMEWE1Y8CUOc9Px41OFM0dw8t2UfcqgQLbY4X0wQ0SHpOGmnP9uNE/mcSGpavk8sM9pgGA8GS+BicJPwJVex2iZsLei/nO8fZDqfI9xuMvPrJOqTEuY8zQaJvZvfrBYNLEAKZKzfuLKOJ1zjSybj6TZseJsI4sObrNrMs8n52rDuw+uWq8J76rnN/zqdCBysE1U2lQtjLy79EHA+/lUyUyS5AyxvEPdVwA+di0Qzb/jUs0ysGbs3PqJwMTC5FaUejDF7n7FbBsCAwEAAQKCAgAe/9pWKMgKDw834xGKksdXy8ejozKbch8TgXxXc6Y74rfJ6BfE5QlmJCTIw1v3a93iF2b/G4dKvVFYRftqrQ97dG/USx39gPuZs+Lqh2W+4G+vvK9br5aL7HTkoQEg+QqTwaaqV+UphMIu0ZBEvKHkVg6LBYQtKOuXUo3NUWSn73M9eA2TXlejiKPxgP/WigAf8lr8+Ea/ZnJgnVqmXgp7fwyRtfU/GJ/17R0JFZhfJTcLqgOnpwkQ/NgeyEJz6vhp+diFJUdwHA/yjmyWrcvkUE/OIvB5fzZTgRQNHXmfjDE3E5LsLY5DVSIOl0Ub/2kg7BwWAmBVhlpH+mZilCbMIOChD5r9LjHIPjr4jKwPpim3plgtnLca52x+HmVXwkFoUl5s3ybvAn0q4OZpjOxZjUrsmhgh/ibzQdvpCNRdESdO8QsBJJf4L35pAo58lYceCQI4vqzFqzOnAIS9we0MUstadsGccUs6n8wVdn9zYfcF3fGPJ4PmNF4a+n9a/vh/XO06T2DUBfrtAOzwB2xeUqR+T90UpBbAM4D4320NynvhX0758fCReFT0Zc/P+yR6+xHzJ4jpqszoIPvfKxvZmSSS/L5oNcaEDzweM0+U5iV4GbJHQlldzCNzPJoOhf1vsqdVhCSiMgScGNuU3V9lYkMesH28STZc8MooiGt7QQKCAQEA2ZuSfGlB4kNsh9R26uwLstg61C2bJbM5oAX5SvTMaGxeRkw192d1T86F5bCyvwZs8AjGxSH5t//cs8snsXHDvIgCh28P5t4b9/+2M/DEDzxWkfgswVVU9JFElgfqioxuq+fc1X+ZLEYxibzNqNDI+

Y019us2S6R7b3oPzWXhdY4sbLdTHt9vSi/CwxMCCqVLjnZHWic7Z/ZIcXOc97I4KyYK1sLusLa
85O+FELoKk+SN7rXOSIqiqFqVWIXSt8hHeHg37Lg15YBFH8f16Bte9G3V0qDyIKVqlcs/dwzJOn
pTxOZgdSy5ChDPS901OyIjHguL6kXQRvlaTfxB4788YQKCAQEAwRKM5qsCvUGf+gTV0OGj
ZmGun9G4EEsIbXyh8HoIdI5JsQY0d+sr2jGxP3Bh3sHQGTkWKI/u/yx7KuglfQo6gGHgx0pYLchZ
b5u+uJTACgXES/U/G2hbB1OznMDqxo1PGuQ6Nl1pmFv5/OLqinuh6Nmsjfa59Ev5ScpgsMhCj+If
z3X+zirGRhuvQGjyWwAmpNEU0C5xinKVx3SimdkxjOb1RiJ6mwcPp6XqVlA49Wq6f6ircuecE7v
pL9PCiFPclH7WrPbu6wDGLQeZd+JHeQ0TGRErmqxD0b3fxSlMyaa52039SQAtoxcQ7WT/PFSlr
zv9UIW05E8Ukhd3Z3fZ+wKCAQEAxvDGpDzqgbvYXfms2isAqvmqFs6292l59kVf2togghsIuUX1
VIDeLB8IJKw9TgzGzU+IwTLFhSVClZDWchAqvZKulTjBFfTbJwg9gBDOfYtRLdPwvrOPnqOsS
2ljUFUA3OJ17y3fUC3AHkdg0cRG6Hbj7iCtm8v9GTZDeil5TKq1DJMxR5lHulAdw4/+HtNtVwnl
Q5N4aRwssKPczkq9tRzrbdn6s83stXQQd/FNuFWMOL83nvQ3rqhZvvthYKmWRZnKrvzeMTnqe
BqsQw266sX7ZLvFgHkaD2YoWGN1GKQ7L1jauGUxqfUiPx42CyCzcEVoHb7Dog9XMtK7PkZY
IQKCAQEAjZmOTyqCwszbiGF9c0nZLiBcEwibhw+QqcPrfNPzWZB1iWh0mLKYodkqFs3UaawZ
dDZDAxK1BN7ZollKDU+wIjImyeLDkyhKN6mxEey+f1tFYTQHyZEameAQbdmrFl3S83UJIJAX/
s8OJCNEQrZzw7PJcwDN1tbViQHtMhpdfusny17NtuT2L7rkKEgP3x/YN67EJf659QEeyn7HNjBtv
9ovbvWgmAxwtdLgb9c8Pvm8uB27SHWHKY36csJK/PMMZrgzt/0OsroJnkLhkmQZaTVmmU7Xf
QP0ZWJtOEuW7pnh225yYdTIo8EqAAG/k769GSzVFvZI+dzpyE9/mhnzewKCAQB6ZeaPI+k7Cs
Nfyt9j4HX9JKi5ZycsBUdBJ+kJ7xOd1Fa0Y/OcnQh6vLeC5oqtJLB4mLnRvutyK6NP3sI6dLNvsqj
VNexl5pSI0/ZlNxNiinFcchl19pnA4zniaLGLh+IO72OQrwMrEhTSuSBZNcR71+roACIVv5t1A57P
lXr+bqv/JIKMwiocFeoWDh9NfuhvA5ecv/Q1OS0rsvK1tSFW+dKzMQgE1P8yB9DmyVZ4F16Oyf
5dERcRzYLd+hFs4eQ+LpP11sMhF8TIN1optRT4+NdY+ZdJbhBYEriFOPVUcfn4bdkmGezmy9U
Fg+IArnHB8bXZowpFwfciEnZuwWNE'), cert :
atob('MIIFijCCA3KgAwIBAgIJANLNn7E5scNGMA0GCSqGSIb3DQEBBQUAMIGBMQswCQY
DVQQGEwJVUzELMAkGA1UECAwCTUExDzANBgNVBAcMBkJvc3RvbjETMBEGA1UECgw
KRXhhbXBsZSBDbzEQMA4GA1UECwwHdGVjaG9wczELMAkGA1UEAwwCY2ExIDAeBgkqh
kiG9w0BCQEWEWNlcnRzQGV4YW1wbGUuY29tMB4XDTE2MDgxMDE0MDEyNFoXDTE5M
DUwNjE0MDEyNFowgYYxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJNQTEPMA0GA1UE
BwwGQm9zdG9uMRMwEQYDVQQKDApFeGFtcGxlIENvMRAwDgYDVQQLDAd0ZWNob3Bz
MRAwDgYDVQQDDAdjbGllbnQxMSAwHgYJKoZIhvcNAQkBFhFjZXJ0c0BleGFtcGxlLmNvbT
CCAiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoCggIBAKQeDiU/WPZ9OWg2oA6l76IYBM
knHH1rccEFEEDaVbr9PigokVxaiz5MyLMCuWSwON6gyk3Z376UnAtRnkdwcQHWv7hsvgW6V
KMyiNQk0P/mfgdUFmzMHkQEZq3LTUgIWHHpQrGHwVOw6a1l6et2aDTwsbTZdMuEBGOj31
KZxv3UrbuIsmqqrHhdTFVjJdsabB1qyocb7t7AEqzsEdymsyneIJD6n0zyxA8+KfGJcy7qL7TpV1N
VsV/ZzlGHdcb+VO6yMNNjgSKWbciAh8R9il+WiJTJ5IEe3zIDV0uI8FMsAKWKXv/oD7w9PfkJn
C6qTc5E4sgHPsMnBP2ix/9kBGVxASxGj+5er8MyPAsGeLc7ASlgeMIYieySY3kmHwelOpHgAph
gosTDbkhvYIliM1SKyYCM7UidjJ4zBFhNWPAlDnPT8eNThTNHcPLdlH3KoEC22OF9MENEh6
Thppz/bjRP5nEhqWr5PLDPaYBgPBkvgYnCT8CVXsdombC3ov5zvH2Q6nyPcbjLz6yTqkxLmPM
0Gib2b36wWDSxACmSs37iyjidc40sm4+k2bHibCOLDm6zazLPJ+dqw7sPrlqvCe+q5zf86nQgcrBN
VNpULYy8u/RBwPv5VMlMkuQMsbxD3VcAPnYtEM2/41LNMrBm7Nz6icDEwuRWlHowxe5+x
WwbAgMBAAEwDQYJKoZIhvcNAQEFBQADggIBAIq0fsXM/NWlqHecu6EQM18x5JQIeCDrru
wrRe9Gj7jEdMka1UzlHWLLnTd6fiyBo0R0Vcdvp96Ic20o5WzK2lx5srBz+B604/Oud682ZE6Cog
QZUSjw4kxP2GBDjnB8wytgebl5QaLLIHCZTS33WvmE6K2cquxMNmeG9CW4kIi/LKJpcBf7t3C
b5fbusosB32oyRt8BpKi3476A1hhcFRTse0P3RV1NrIyrwgxq0PY+uEtEz9BQRCT1fZohPKlEUM
WYzV3Wj8eboaWohW+Tonr2zuu1WhF3bxKe9vd0E3hWoe7bI5I010XlDxogFFokr6MVLzXV95q
0zuULu1NqcIAwgttzwe9fsXMd+k+rD9YNNW3fXu4WtqVoovwLIUYNDRQENIIbyigyhCUzIXo9
7Xn43vKxxTtyu546QrcfgX59hFjxK0yRxwh2EjuUG10bo2UCes6k6QzBvWrH8gNl4mWFyXaqXJJ
I4o0R5hTIRyVp/lquXYz6pdWNcuR/5SzD/1F0+Z6SJ4xPbxRZTVolPRJ7FFlJ5GaFvkinA4pDskYD
vFtfmcy0hdY/0nCJgvYUMtZfMtWYTfKL+4EfjFk1/yqZsNAv/gkP87QQwqA+tn1NJ0kfM0I9n28X

```
DLhdU92f0yHjCScy1o3ewcB7I0smFcUDD4OCYFUiNahxAUA0mwiP'),  ca :
atob('MIIFgDCCA2gCCQDQNeSIxshM8DANBgkqhkiG9w0BAQUFADCBgTELMAkGA1UEBh
MCVVMxCzAJBgNVBAgMAk1BMQ8wDQYDVQQHDAZCb3N0b24xEzARBgNVBAoMCkV4
YW1wbGUgQ28xEDAOBgNVBAsMB3RlY2hvcHMxCzAJBgNVBAMMAmNhMSAwHgYJKoZI
hvcNAQkBFhFjZXJ0c0BleGFtcGxlLmNvbTAeFw0xNjA4MTAxMzUxMjZaFw0xOTA1MDYxMz
UxMjZaMIGBMQswCQYDVQQGEwJVUzELMAkGA1UECAwCTUExDzANBgNVBAcMBkJvc
3RvbjETMBEGA1UECgwKRXhhbXBsZSBDbzEQMA4GA1UECwwHdGVjaG9wczELMAkGA1
UEAwwCY2ExIDAeBgkqhkiG9w0BCQEWEWNlcnRzQGV4YW1wbGUuY29tMIICIjANBgkqhki
G9w0BAQEFAAOCAg8AMIICCgKCAgEAs0Tdt33C5DJlFbJBeqbbGDU+ipNNHJeTFU0WC28B
UNOPSMcZiWqc99LOCHXaW3M66PYJ9qRoHSoPG2dUBdCTFDeRgh32vVrtKGEgIeILJ19k4d0
s2Px8fyOgQQybekDR/o2PWxYxlI+SRS+0SvwJVPqEDzjCaQC7RVMD/r98VvK3uiHziSlQVWTA
RpbkYWU6PFh9S+rD4Mf6fjY6IKotebHKj5CuS+nv71mUv1l4DmGwTn9kf7ZtLgKEFP51JsuuQ4r
XuYWmK2HOn92dquKyXwY+rWwHbXmSW9Toxg4AtQutXd4A6AiNmgNj7Gaa69nnXe7ug4ws
MDSLqYWG+1z1FGomTTMngq5eXrURAE5yIPD6KhookZSowdzJKE13S6ijbR/qx8HjaBQD0bi8
yrvjGOaXaQwU0NBPBsTBEg6xSQDPgx0nv+4YSLrF9CeDNdWIBPrjrc7wvv7xkG4U6n8ZNG9N
9RkC+J6UJQ3C9N57X3xefvi8PbqlMom0ERK9phLjIzyNWqtImO18egS2caYhnccmuHeulFBlfZEo
DggdlnL5i0I9QG3Fu9LCE1cziA18q6qbve+sweAH4vVSX/G9ImYfodKWitbDov8lgI+fqaPBK5O1
qqgT6ub+NonVPIeicfGBy7VZzDTDImgt8ucA1vXiOSng7ZmLfm3fHD3g3ScCAwEAATANBgkq
hkiG9w0BAQUFAAOCAgEAs0SAB49Ws0DudLG+xkFUvMXju+K3o8kJCFNXwzHe6djfTr5dKD
wl+Tsx2maQR8JMZfuDcUBeQK/sMxF2Ps3RmQ1rA4OcRYZPwPJXs7LeREZ1mEKJwm4HwOc
mnNbuIQ6vqd/TEhc5tpSUpxHytez7QERmncnch4PSJngJCAVgGBUr6sVTJCTBX3hhcsR6JXMVo
aXS3OPPpMDwnH4ooS5MV1uZqa9zsubjlJUGEKKFN7Z0FCvPJXI4XC6pHbxQGz4/LeKSeKvBP
HG6sTkwJuQFyHm7SXf0E0708tX1FKOCvQFPHAAakccoKDgVxYJhOZ23ZK2ooi43pb7Yu706w
gUyZ9B1PPvRymKq2KMaoWvPFC72JuUlLhkIEBONLhsV6gc+rY9HBx6WP6DATRnm5o5HiN
Q0gnEjlUnb+ou0BWkVRbIin+iowmYEXuw9/nze8mpC62jufqTu9Uu5NjMr911A4E4VUIHxk7BQ
9rLdDppv49bSMU4ccVzXcjCbiP/fo1pPfPHCC96En2sXKGxSEv3HVyJJN2n5E/vVCp/9A7nXZIlG
ILMD1HHuTqUzs1H6qgtPKWHPLPxo0XGiYhoKMWviP41oPJqRJC1lvrXQV2Ae1+Giz1Ah6s8d
R7ufZ0tI+mv2Rg4VOa+gPd2lI/bMhID2hq2DSeZ+am7HHGQrhl5R4OI='),

// задаем опции подключения к серверу  host : 'api.travis-ci.org',  port : '443',  path
:'/repos/'+repo+'/branches/'+branch,  protocol : 'https:',  headers : {    'Accept': 'application/vnd.travis-
ci.2+json'  }}; // Функция включения реле в зависимости от состояния state

// blink(5) включает реле на 5 секунд, а затем автоматически выключает егоvar lightUp =
function(state) {  switch(state) {    case 'failed': red.blink(5); break;    case 'passed': reen.blink(5);
break;    case 'created': yellow.blink(5); break;    case 'started': yellow.blink(5); break;    default:
yellow.blink(5);  }};

// функция запроса данных с сервера Travis CIvar request = function() {   var response = '';
http.get(options, function(res)  {

// Большие ответы от сервера могут приходить по частям.

// В переменной response собираем весь ответ целиком.    res.on('data', function(data) {
response += data;    });

// После закрытия соединения обрабатываем весь пришедший ответ    res.on('close',
function(noError) {

// Преобразуем ответ сервера из формата JSON в объект JavaScript      var answer =
JSON.parse(response);
```

// проверяем результат преобразования на корректность if (answer !== undefined) {

// Получаем номер и статус последнего теста var build = answer.branch.number; var state = answer.branch.state;

// Если текущий тест в процессе выполнения,

// ещё успеваем отобразить его статус if (state === 'created' || state === 'started') { lightUp(state); } else {

// Если появились свежие выполненные тесты, // на которые мы ещё не реагировали. if (lastBuild !== build) {

// Если lastBuild равен нулю, светофор // только что включили в питание, а результаты

// тестов уже известны без виджета. if (lastBuild !== 0) { lightUp(state); }

// Запоминаем номер последнего теста,

// чтобы больше на него не реагировать. lastBuild = build; } } } // Повторяем запрос к Travis-ci через 5 секунд setTimeout(request, 5000); }); });}; // Получаем IP-адрес от DHCP-сервераeth.setIP();

// Делаем первый запрос к сервисуrequest();

Хак

Библиотека wiznet и https-соединение выбрасывают в консоль много отладочной информации. При запуске Iskra JS без подсоединения к консоли, код выполняться не будет, так как вывод в консоль замораживает выполнение программы, пока компьютер не запросит новую порцию данных. Чтобы запустить код, необходимо в первой строке программы прописать Serial3.setConsole(true);. Эта строка заставит Iskra JS выводить информацию в консоль через последовательный интерфейс на пинах P0 и P1. Замкните эти пины проводком, чтобы информация в консоль могла передаваться, не заполняя буфер последовательного интерфейса.

Маячок

В этом эксперименте мы просто мигаем светодиодом.

Список деталей для эксперимента

- 1 плата Arduino Uno
- 1 беспаечная макетная плата
- 1 светодиод
- 1 резистор номиналом 220 Ом
- 2 провода «папа-папа»

Принципиальная схема

Схема на макетке

Обратите внимание

- Не забудьте, как соединены рельсы в беспаечной макетной плате. Если на вашей макетке красная и синяя линии вдоль длинных рельс прерываются в середине, значит проводник внутри макетки тоже прерывается!
- Катод («минус») светодиода — короткая ножка, именно её нужно соединять с землёй (GND)
- Не пренебрегайте резистором, иначе светодиод выйдет из строя
- Выбрать резистор нужного номинала можно с помощью таблицы маркировки или с помощью мультиметра в режиме измерения сопротивления
- Плата Arduino имеет три пина GND, используйте любой из них

Скетч

p010_blink.ino

```
void setup(){

// настраиваем пин №13 в режим выхода,

 // т.е. в режим источника напряжения  pinMode(13, OUTPUT);} void loop(){

// подаём на пин 13 «высокий сигнал» (англ. «high»), т.е.

// выдаём 5 вольт. Через светодиод побежит ток.

// Это заставит его светиться  digitalWrite(13, HIGH);

// задерживаем (англ. «delay») микроконтроллер в этом

// состоянии на 100 миллисекунд  delay(100);

// подаём на пин 13 «низкий сигнал» (англ. «low»), т.е.

// выдаём 0 вольт или, точнее, приравниваем пин 13 к земле.

// В результате светодиод погаснет  digitalWrite(13, LOW);

// замираем в этом состоянии на 900 миллисекунд  delay(900);

// после «размораживания» loop сразу же начнёт исполняться

// вновь, и со стороны это будет выглядеть так, будто

// светодиод мигает раз в 100 мс + 900 мс = 1000 мс = 1 сек}
```

Пояснения к коду

- Процедура setup выполняется один раз при запуске микроконтроллера. Обычно она используется для конфигурации портов микроконтроллера и других настроек
- После выполнения setup запускается процедура loop, которая выполняется в бесконечном цикле. Именно этим мы пользуемся в данном примере, чтобы маячок мигал постоянно

- Процедуры setup и loop должны присутствовать в любой программе (скетче), даже если вам не нужно ничего выполнять в них — пусть они будут пустые, просто не пишите ничего между фигурными скобками. Например:

```
void setup(){}
```

- Запомните, что каждой открывающей фигурной скобке { всегда соответствует закрывающая }. Они обозначают границы некого логически завершенного фрагмента кода. Следите за вложенностью фигурных скобок. Для этого удобно после каждой открывающей скобки увеличивать отступ на каждой новой строке на один символ табуляции (клавиша Tab)
- Обращайте внимание на ; в концах строк. Не стирайте их там, где они есть, и не добавляйте лишних. Вскоре вы будете понимать, где они нужны, а где нет.
- Функция digitalWrite(pin, value) не возвращает никакого значения и принимает два параметра:
  pin — номер цифрового порта, на который мы отправляем сигнал
  value — значение, которое мы отправляем на порт. Для цифровых портов значением может быть HIGH (высокое, единица) или LOW (низкое, ноль)
- Если в качестве второго параметра вы передадите функции digitalWrite значение, отличное от HIGH, LOW, 1 или 0, компилятор может не выдать ошибку, но считать, что передано HIGH. Будьте внимательны
- Обратите внимание, что использованные нами константы: INPUT, OUTPUT, LOW, HIGH, пишутся заглавными буквами, иначе компилятор их не распознает и выдаст ошибку. Когда ключевое слово распознано, оно подсвечивается синим цветом в Arduino IDE

Вопросы для проверки себя

Что будет, если подключить к земле анод светодиода вместо катода?

Что будет, если подключить светодиод с резистором большого номинала (например, 10 кОм)?

Что будет, если подключить светодиод без резистора?

Зачем нужна встроенная функция pinMode? Какие параметры она принимает?

Зачем нужна встроенная функция digitalWrite? Какие параметры она принимает?

С помощью какой встроенной функции можно заставить микроконтроллер ничего не делать?

В каких единицах задается длительность паузы для этой функции?

Задания для самостоятельного решения

Сделайте так, чтобы маячок светился полсекунды, а пауза между вспышками была равна одной секунде

Измените код примера так, чтобы маячок включался на три секунды после запуска устройства, а затем мигал в стандартном режиме

Перетягивание каната

В этом эксперименте мы создаем еще одну игру, на этот раз нужно быстрее соперника нажать кнопку 20 раз.

Список деталей для эксперимента

- 1 плата Arduino Uno
- 1 беспаечная макетная плата
- 1 светодиодная шкала
- 10 резисторов номиналом 220 Ом
- 4 резисторов номиналом 100 кОм
- 2 тактовых кнопки
- 2 керамических конденсатора номиналом 100 нФ
- 1 пьезопищалка
- 1 инвертирующий триггер Шмитта
- 24 провода «папа-папа»

Для дополнительного задания

- 1 сервопривод
- 1 конденсатор 220 мкФ

Принципиальная схема

Схема на макетке

Обратите внимание

- Схема подключения кнопок с использованием конденсаторов, резисторов и микросхемы 74HC14, которая называется инвертирующий триггер Шмитта, нужна для аппаратного подавления дребезга. Посмотрите видеоурок на эту тему.
- В этом эксперименте нам нужно очень много цифровых портов, поэтому нам пришлось использовать порт 0. Пользоваться им неудобно из-за того, что он соединен с одним из каналов последовательного порта, поэтому перед прошивкой микроконтроллера нам придется отключать провод, идущий к пьезопищалке, а после прошивки подключать его обратно.

Скетч

p200_button_wrestling.ino

```
#define BUZZER_PIN     0

#define FIRST_BAR_PIN  4

#define BAR_COUNT      10

#define MAX_SCORE      20
```

// глобальные переменные, используемые в прерываниях (см. далее)

// должны быть отмечены как нестабильные (англ. volatile)volatile int score = 0; void setup(){ for (int i = 0; i < BAR_COUNT; ++i) pinMode(i + FIRST_BAR_PIN, OUTPUT); pinMode(BUZZER_PIN, OUTPUT);

// Прерывание (англ. interrupt) приостанавливает основную

// программу, выполняет заданную функцию, а затем возобновляет

// основную программу. Нам нужно прерывание на нажатие кнопки,

// т.е. при смене сигнала с высокого на низкий, т.е. на

// нисходящем (англ. falling) фронте attachInterrupt(INT1, pushP1, FALLING);

// INT1 — это 3-й пин attachInterrupt(INT0, pushP2, FALLING);

// INT0 — это 2-й пин} void pushP1() { ++score; }

// функция-прерывание 1-го игрокаvoid pushP2() { --score; }

// функция-прерывание 2-го игрокаvoid loop(){ tone(BUZZER_PIN, 2000, 1000);

// даём сигнал к старту.

// пока никто из игроков не выиграл, обновляем «канат» while (abs(score) < MAX_SCORE) { int bound = map(score, -MAX_SCORE, MAX_SCORE, 0, BAR_COUNT); int left = min(bound, BAR_COUNT / 2 - 1); int right = max(bound, BAR_COUNT / 2); for (int i = 0; i < BAR_COUNT; ++i) digitalWrite(i + FIRST_BAR_PIN, i >= left && i <= right); } tone(BUZZER_PIN, 4000, 1000);

// даём сигнал победы while (true) {}

// «подвешиваем» плату до перезагрузки}

Пояснения к коду

- Код нашей обычной программы исполняется инструкция за инструкцией и если мы, например, проверяем состояние датчика, мы к нему обратимся только в те моменты, когда очередь дойдет до соответствующей инструкции. Однако мы можем использовать прерывания:
  по наступлении *определенного события*
  на *определенном порту* ход программы будет приостанавливаться для выполнения *определенной функции*, а затем программа продолжит исполняться с того места, где была приостановлена.
- Arduino Uno позволяет делать прерывания на портах 2 и 3.
- В setup() прописывается инструкция attachInterrupt(interrupt, action, event), где interrupt может принимать значения INT0 или INT1 для портов 2 и 3 соответственно. Можно задать эти значения и с помощью функции digitalPinToInterrupt(pin), где вместо pin указать номер пина.

  action — имя функции, которая будет вызываться при наступлении события

event — событие, которое мы отслеживаем. Может принимать значение RISING (изменение от низкого уровня сигнала к высокому, от 0 к 1), FALLING (от высокого уровня к низкому, от 1 к 0), CHANGE (от 0 к 1 или от 1 к 0), LOW (при низком уровне сигнала).

- Глобальные переменные, к которым мы обращаемся из функции, обрабатывающей прерывания, должны объявляться с использованием ключевого слова volatile, как в данном эксперименте volatile int score = 0.
- Внутри функции, вызываемой по прерыванию, нельзя использовать delay().
- Функция abs(value) возвращает абсолютное значение value (значение по модулю). Обратите внимание, что функция может сработать некорректно, если передавать ей выражение, которое еще не вычислено, например abs(++a), лучше передавать ей просто переменную.
- Функция min(val1, val2) вернет меньшее из val1 и val2.
- Функция max(val1, val2) вернет большее из val1 и val2.
- В данном эксперименте мы вычисляем значение, которое записывается на светодиоды, прямо в digitalWrite()
- Мы уже знакомы с логическим «и» (&&). Нередко нужен оператор «логическое «или»: ||. Он возвращает «истину», если хотя бы один из операндов имеет значение «истина». false || false вернет false, a true || true, true || false и false || true вернут true.
- Мы использовали while(true){} для того, чтобы loop() остановился после того, как кто-то выиграл: у while всегда истинное условие и он бесконечно ничего не выполняет!

Вопросы для проверки себя

Каким образом мы подавляем дребезг аппаратно?

Для чего используются прерывания?

Каким образом можно включить обработку внешних прерываний?

О каких нюансах работы с уже известными нам вещами следует помнить при работе с прерываниями?

Как выбрать максимальное из двух значений? Минимальное?

Как получить абсолютное значение переменной? Чего следует избегать при использовании этой функции?

Когда оператор логическое «или» возвращает «ложь»?

Задания для самостоятельного решения

Вместо светодиодной шкалы подключите сервопривод и измените код таким образом, чтобы перетягивание демонстрировалось путем отклонения сервопривода от среднего положения.

Пульсар

В этом эксперименте мы плавно наращиваем яркость светодиодной шкалы, управляя большой нагрузкой через транзистор.

Прочтите перед выполнением

- Биполярный транзистор
- Светодиодные сборки

Список деталей для эксперимента

- 1 плата Arduino Uno
- 1 беспаечная макетная плата
- 1 биполярный транзистор
- 1 светодиодная шкала
- 1 резистор номиналом 1 кОм
- 10 резисторов номиналом 220 Ом
- 13 проводов «папа-папа»

Принципиальная схема

Схема на макетке

Обратите внимание

- Светодиодная шкала — это несколько светодиодов в одном корпусе. Нам нужно чтобы питание шло к их анодам, а катоды направлялись к земле. Скорее всего на вашей шкале аноды находятся со стороны маркировки. Если шкала не светится, когда должна, попробуйте перевернуть ее.
- База биполярного транзистора — это его средняя ножка. Если повернуть транзистор плоской стороной к себе, ножками вниз, то левая ножка это коллектор, а правая — эмиттер.
- Если эту схему собрать без резистора между базой транзистора и портом Arduino, мы практически устроим короткое замыкание порта на землю. Рано или поздно это выведет из строя транзистор или ножку микроконтроллера.
- Зачем здесь вообще транзистор? Без него такое количество светодиодов будет потреблять больше тока, чем 40 мА, которые может себе позволить цифровой пин платы. Поэтому мы берем питание из порта 5V, рассчитанного на ток до 500 мА, а на цифровой порт ставим транзистор, чтобы с помощью малого тока управлять большим.
- В данном случае мы включили 10 светодиодов параллельно, каждый через отдельный резистор. Включать их через один резистор неправильно: даже светодиоды из одной партии имеют минимальный разброс вольт-амперных характеристик, вследствие чего они:
  Светились бы с различной яркостью
  Из-за минимальной разницы во времени включения, больший ток, прошедший через первый включившийся светодиод, мог бы вывести его из строя. И так по цепочке.

Скетч

p060_pulse_bar.ino

```
#define CONTROL_PIN  9
```

// переменные верхнего уровня, т.е. объявленные вне функций,

// называют глобальными. Их значения сохраняются всё время,

// пока работает микроконтроллерint brightness = 0; void setup(){ pinMode(CONTROL_PIN, OUTPUT);} void loop(){

// увеличиваем значение яркости на единицу, чтобы нарастить

// яркость. Однако яркость не должна быть более 255, поэтому

// используем операцию остатка от деления, чтобы при

// достижении значения 255, следующим значением снова стал 0

// Y % X — это остаток от деления Y на X;

// плюс, минус, делить, умножить, скобки — как в алгебре. brightness = (brightness + 1) % 256;

// подаём вычисленный ШИМ-сигнал яркости на пин с базой

// управляющего транзистора analogWrite(CONTROL_PIN, brightness);

// ждём 10 мс перед следующим наращиванием яркости. Таким

// образом, полный накал будет происходить в течение

// 256×10 = 2560 мс delay(10);}

Пояснения к коду

- Как мы уже знаем, analogWrite(pin, value) в качестве value принимает значения от 0 до 255. Если передать значение из-за пределов этого диапазона, функция сработает, но в общем случае вы получите неожиданный результат.
- Оператор X % Y дает остаток от деления X на Y. Если X меньше Y, т.е. целая часть результата деления равна 0, оператор % будет возвращать X. Таким образом:
  Пока brightness + 1 меньше 256, в brightness записывается значение brightness + 1
  Как только brightness + 1 принимает значение 256, результатом (brightness + 1) % 256 становится 0 и на следующей итерации loop() всё начинается сначала.
- Оператор % работает только с целыми операндами.
- В выражении (brightness + 1) % 256 скобки используются для назначения порядка действий. Операция % имеет больший приоритет, чем +, а сложение нам нужно выполнять раньше. С операциями умножения и деления оператор взятия остатка имеет одинаковый приоритет.

Вопросы для проверки себя

Почему у светодиодной шкалы на 10 сегментов 20 ножек?

Зачем в схеме биполярный транзистор?

За счет чего увеличивается яркость шкалы?

Почему после достижения значения 255 переменная brightness обнуляется?

Задания для самостоятельного решения

Измените программу так, чтобы яркость шкалы росла только до половины от максимальной.

Измените программу так, чтобы шкала становилась максимально яркой в три раза быстрее, без изменения функции delay.

Измените исходную программу так, чтобы такой же результат был получен без использования операции %, но с применением условного оператора if.

Кнопочные ковбои

В этом эксперименте мы создаем игрушку на реакцию: кто быстрее нажмет кнопку по сигналу.

Список деталей для эксперимента

- 1 плата Arduino Uno
- 1 беспаечная макетная плата
- 2 тактовых кнопки
- 2 резистора номиналом 220 Ом
- 2 светодиода
- 1 пьезопищалка
- 10 проводов «папа-папа»

Принципиальная схема

Схема на макетке

Скетч

p120_button_cowboys.ino

```
#define BUZZER_PIN   12

// пин с пищалкой#define PLAYER_COUNT 2

// количество игроков-ковбоев

// вместо перечисления всех пинов по-одному, мы объявляем пару

// списков: один с номерами пинов с кнопками, другой — со

// светодиодами. Списки также называют массивами (англ. array)int
buttonPins[PLAYER_COUNT] = {3, 13};int ledPins[PLAYER_COUNT] = {9, 11}; void setup(){
pinMode(BUZZER_PIN, OUTPUT);  for (int player = 0; player < PLAYER_COUNT; ++player) {
// при помощи квадратных скобок получают значение в массиве
```

// под указанным в них номером. Нумерация начинается с нуля pinMode(ledPins[player], OUTPUT); pinMode(buttonPins[player], INPUT_PULLUP); }} void loop(){

// даём сигнал «пли!», выждав случайное время от 2 до 7 сек delay(random(2000, 7000)); tone(BUZZER_PIN, 3000, 250);

// 3 килогерца, 250 миллисекунд for (int player = 0; ; player = (player+1) % PLAYER_COUNT) {

// если игрок номер «player» нажал кнопку... if (!digitalRead(buttonPins[player])) {

// ...включаем его светодиод и сигнал победы на 1 сек digitalWrite(ledPins[player], HIGH); tone(BUZZER_PIN, 4000, 1000); delay(1000); digitalWrite(ledPins[player], LOW); break; // Есть победитель! Выходим (англ. break) из цикла } }}

Пояснения к коду

- Массив состоит из элементов одного типа, в нашем случае int.
- Объявить массив можно следующими способами:

int firstArray[6];

// 6 целых чисел с неопределёнными начальными значениямиint pwmPins[] = {3, 5, 6, 9, 10, 11};

// 6 целых чисел, длина вычисляется автоматомboolean buttonState[3] = {false, true, false};

// можно использовать элементы любого типа

- Когда мы объявляем массив с указанием количества его элементов n, это число всегда на 1 больше, чем номер последнего элемента (n-1), т.к. индекс первого элемента — 0.
- Считать или записать значение элемента массива можно, обратившись к нему по индексу, например firstArray[2] или buttonState[counter], где counter — переменная, такая как счетчик цикла
- В переменных типа long можно хранить значения до 2 147 483 647. unsigned int в этом случае нам будет недостаточно, потому что 65 535 миллисекунд пройдут чуть больше чем за минуту!
- Функция random(min, max) возвращает целое псевдослучайное число в интервале [min, max]. Для драматичности каждая игра начинается с паузы случайной длины.
- Благодаря массивам в этом эксперименте мы настраиваем порты, считываем кнопки и включаем светодиоды в циклах со счетчиком, который используется как индекс элемента.
- Мы используем цикл for без условия его завершения, поэтому пока мы явно того не потребуем, цикл будет крутиться до бесконечности.
- Мы использовали выражение player = (player+1) % PLAYER_COUNT для счётчика цикла, чтобы не только увеличивать его на единицу каждый раз, но и обнулять при достижении последнего игрока.
- Инструкция break прекращает работу цикла и выполнение программы продолжается с инструкции после его конца.

Вопросы для проверки себя

Можно ли поместить в один массив элементы типа boolean и int?

Обязательно ли при объявлении массива заполнять его значениями?

Чем удобно использование массива?

Как обратиться к элементу массива, чтобы прочитать его значение?

Чем отличаются инструкции continue и break?

Задания для самостоятельного решения

Сделайте напряженный вариант игры: пусть интервал между сигналами будет в диапазоне от 10 до 15 секунд.

В игре есть лазейка: кнопку можно зажать до сигнала «пли!» и таким образом сразу же выиграть. Дополните программу так, чтобы так выиграть было нельзя.

Добавьте в игру еще двух ковбоев!

Кнопочный переключатель

В этом эксперименте мы делаем из тактовой кнопки триггер, борясь с «дребезгом».

Список деталей для эксперимента

- 1 плата Arduino Uno
- 1 беспаечная макетная плата
- 1 тактовая кнопка
- 1 резистор номиналом 220 Ом
- 1 светодиод
- 5 проводов «папа-папа»

Для дополнительного задания

- еще 1 кнопка
- еще 2 провода

Принципиальная схема

Схема на макетке

Обратите внимание

- Мы могли бы один из контактов кнопки соединить проводом напрямую с одним из входов GND, но мы сначала «раздали» «землю» на длинную рельсу макетки. Если мы работаем с макетной платой, так поступать удобнее, т.к. в схеме могут появляться новые участки, которые тоже нужно будет соединить с «землей»

- Также полезно руководствоваться соображениями аккуратности изделия, поэтому катод светодиода мы соединяем с другим входом GND отдельным проводом, который не мешает нам работать в середине макетки.

Скетч

p100_led_toggle.ino

```
#define BUTTON_PIN  3

#define LED_PIN     13 boolean buttonWasUp = true;

// была ли кнопка отпущена?boolean ledEnabled = false;

// включен ли свет? void setup(){  pinMode(LED_PIN, OUTPUT);  pinMode(BUTTON_PIN, INPUT_PULLUP);} void loop(){

// определить момент «клика» несколько сложнее, чем факт того,

// что кнопка сейчас просто нажата. Для определения клика мы

// сначала понимаем, отпущена ли кнопка прямо сейчас...  boolean buttonIsUp = digitalRead(BUTTON_PIN);

// ...если «кнопка была отпущена и (&&) не отпущена сейчас»...  if (buttonWasUp && !buttonIsUp) {    // ...может это «клик», а может и ложный сигнал (дребезг),

// возникающий в момент замыкания/размыкания пластин кнопки,

// поэтому даём кнопке полностью «успокоиться»...    delay(10);

// ...и считываем сигнал снова    buttonIsUp = digitalRead(BUTTON_PIN);    if (!buttonIsUp) {  // если она всё ещё нажата...

// ...это клик! Переворачиваем сигнал светодиода      ledEnabled = !ledEnabled;  digitalWrite(LED_PIN, ledEnabled);    }  }

// запоминаем последнее состояние кнопки для новой итерации  buttonWasUp = buttonIsUp;}
```

Пояснения к коду

- Поскольку мы сконфигурировали вход кнопки как INPUT_PULLUP, при нажатии на кнопку на данном входе мы будем получать 0. Поэтому мы получим значение true («истина») в булевой переменной buttonIsUp («кнопка отпущена»), когда кнопка отпущена.
- Логический оператор && («и») возвращает значение «истина» только в случае истинности обоих его операндов. Взглянем на так называемую таблицу истинности для выражения buttonWasUp && !buttonIsUp («кнопка была отпущена и кнопка не отпущена»):

buttonWasUp	buttonIsUp	!buttonIsUp	buttonWasUp && !buttonIsUp
0	0	1	0
0	1	0	0

1	0	1	1
1	1	0	0

Здесь рассмотрены все возможные сочетания предыдущего и текущего состояний кнопки и мы видим, что наш условный оператор if сработает только в случае, когда кнопка нажата только что: предыдущее состояние 1 («была отпущена»), а текущее 0 («не отпущена»).

- Через 10 миллисекунд мы проверяем еще раз, нажата ли кнопка: этот интервал больше, чем длительность «дребезга», но меньше, чем время, за которое человек успел бы дважды нажать на кнопку. Если кнопка всё еще нажата, значит, это был не дребезг.
- Мы передаем в digitalWrite не конкретное значение HIGH или LOW, а просто булеву переменную ledEnabled. В зависимости от того, какое значение было для нее вычислено, светодиод будет зажигаться или гаситься.
- Последняя инструкция в buttonWasUp = buttonIsUp сохраняет текущее состояние кнопки в переменную предыдущего состояния, ведь на следующей итерации loop текущее состояние уже станет историей.

Вопросы для проверки себя

В каком случае оператор && возвращает значение «истина»?

Что такое «дребезг»?

Как мы с ним боремся в программе?

Как можно избежать явного указания значения уровня напряжения при вызове digitalWrite?

Задания для самостоятельного решения

Измените код так, чтобы светодиод переключался только после отпускания кнопки.

Добавьте в схему еще одну кнопку и доработайте код, чтобы светодиод зажигался только при нажатии обеих кнопок.

VII. Список литературы

Для преподавателей

Литература, педагогические издания и методические материалы

1. Страуструп Бьерн. Программирование. Принципы и практика с использованием C++, М.: Вильямс, 2016. — 1328 с.
2. Блум Джереми. Изучаем Arduino: инструменты и методы технического волшебства: Пер с англ. — СПб.: БХВ-Петербург, 2018. — 336 с.: ил.
3. Петин В. А. Arduino и Raspberry Pi в проектах Internet of Things. — СПб.: БХВ-Петербург, 2016 — 320 с.: ил. — (Электроника)
4. Липпман Стенли, Лайоже Жози, Му Барбара. Язык программирования C++. Базовый курс, 5-е издание, М.: Вильямс, 2017. — 1120 с.
5. Браун Этан. Изучаем JavaScript. Руководство по созданию современных веб-сайтов, М.: Альфа-книга, 2017. — 368 с.
6. Роббинс Д. Н. HTML5, CSS3 и JavaScript. Исчерпывающее руководство, М.: Эксмо, 2014. — 528 с.
7. Лутц, М. Программирование на Python. Т. 1 / М. Лутц. — М.: Символ, 2016. — 992 с.
8. Лутц, М. Программирование на Python. Т. 2 / М. Лутц. — М.: Символ, 2016. — 992 с.
9. Кузьменко, Н.Г. Компьютерные сети и сетевые технологии / Н.Г. Кузьменко. — СПб.: Наука и техника, 2013. — 368 с.
10. Куроуз, Д. Компьютерные сети. Нисходящий подход / Д. Куроуз, К. Росс. — М.: Эксмо, 2016. — 912 с.
11. Максимов, Н. В. Архитектура ЭВМ и вычислительных систем / Н.В. Максимов, И.И. Попов, Т.Л. Партыка. — М.: Форум, Инфра-М, 2013. — 512 с.
12. Азбука электроники. Изучаем Arduino / Ю. Ревич. — Москва: Издательство АСТ: Кладезь, 2017 — 224 с. — (Электроника для всех).

Тематические веб-ресурсы

1. Программирование Ардуино. — Режим доступа: http://www. http://arduino.ru/Reference
2. Основы программирования на языках С и С++ для начинающих. — Режим доступа: http://cppstudio.com/
3. Основы программирования на языке Python для начинающих. — Режим доступа: — Режим доступа: https:// pythonworld.ru/samouchitel-python
4. Основы программирования на языке Python для начинающих. — Режим доступа: https://itproger.com/

Для детей

Литература, педагогические издания и методические материалы

1. Браун Этан. Изучаем JavaScript. Руководство по созданию современных веб-сайтов, М.: Альфа-книга, 2017. — 368 с.
2. Роббинс Д. Н. HTML5, CSS3 и JavaScript. Исчерпывающее руководство, М.: Эксмо, 2014. — 528 с.
3. Scratch и Arduino для юных программистов и конструкторов/ Ю. А. Винницкий, А. Т. Григорьев. — СПб.: БХВ-Петербург, 2018. — 176 с.: ил.

Тематические веб-ресурсы

1. Программирование на Python. — Режим доступа: https:// stepik.org
2. Основы изучения HTML и CSS. — Режим доступа: http:// htmlbook.ru/
3. Книги по изучению Python, Swift, JavaScript для начинающих. — Режим доступа: https://bookflow.ru/knigi-poprogrammirovaniyu-dlya-detej/
4. Ресурсы для повышения кругозора по направлению
5. Свободно распространяемая программная система для изу-IT-квантум: тулкит

чения азов программирования дошкольниками и младшими школьниками. — Режим доступа: https://piktomir.ru/
6. CodeCombat — это платформа для учеников, чтобы изучать информатику во время игры. — Режим доступа: https:// codecombat.com/
7. 230 минут TED Talks: лучшие лекции о технологиях, бизнесе и интернете. — Режим доступа: https://www. cossa.ru/trends/228574/?utm_campaign=letters&utm_source=sendpulse&utm_medium=email&spush=b2tzc2VsbEB 5YWhvby5jb20