Автономная некоммерческая организация
«Красноярский детский технопарк «Кванториум»

РЕКОМЕНДОВАНО
методическим советом

Протокол № 13
от «30» мая 2025 г.

УТВЕРЖДАЮ
Генеральный директор
_____ Кениг С.Р.
Приказ № 42
от «30» мая 2025 г.

Дополнительная общеобразовательная общеразвивающая программа технической направленности

«Энергетика 3»

Срок реализации:
1 год
Возраст:
14-18 лет
Составитель программы:
Шереметьева Ю.А.

г. Красноярск, 2025 г.

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Дополнительная общеобразовательная общеразвивающая программа «Энергетика 3» (далее - программа) имеет техническую направленность, продвинутый уровень сложности и ориентирована на обучающихся 14-18 лет. Программа в объеме 144 часа рассчитана на 1 год обучения из расчета 4 часа в неделю.

1.1 АКТУАЛЬНОСТЬ

Актуальность и необходимость данной программы продиктована развитием современной энергетики, необходимостью широкого внедрения экологичных возобновляемых источников энергии, а также широким распространением индивидуального транспорта. Особенностью программы является то, что она, будучи междисциплинарной, направлена на формирование практических навыков в различных областях энергетики, актуальных в настоящее время: альтернативные источники энергии и их практическое применение, энергосберегающие технологии, новые источники энергии, проблемы построения закрытых и открытых энергосистем и другие.

Для развития энергетики в любых масштабах, будь то локальная система для одного дома или несколько станций для обеспечения целой страны, необходимы проекты с правильно поставленной целью. Важно не только уметь определять технологии будущего, разбираться в их устройстве, но также рассчитывать экономическую целесообразность строительства и эксплуатации тех или иных источников энергии в определенных условиях.

Нынешняя ситуация на рынке энергетики в Красноярском крае позволяет говорить о неэффективном использовании энергетических ресурсов. В крае остро стоят проблемы “чистой” энергии, связанной, прежде всего, с загрязнением окружающей среды при нынешних методах энергогенерации, что приводит к росту заболеваний и ухудшению общей экологической обстановки. К тому же, отсутствие новых технологий в области получения и передачи энергии не позволяют эффективно использовать энергетический потенциал края, связанный с большим количеством

природных ресурсов. Кроме того, по всей России уже встает вопрос о новой системе электроснабжения, где жители, использующие альтернативные источники энергии, могут не только снабжать свой дом, но также продавать лишнюю энергию городу.

Для развития новых технологий необходимо освоить проектную деятельность, которая позволит создавать новые источники энергии, улучшать и объединять уже существующие в эффективные системы.

1.2 ПЕДАГОГИЧЕСКАЯ ЦЕЛЕСООБРАЗНОСТЬ

В настоящий момент все чаще применяется проектный метод обучения для развития у детей навыков самостоятельной работы с информацией, критического мышления, применения знаний в практической деятельности.

Данная образовательная программа помогает в решении следующих актуальных педагогических задач, таких как:

- показать место и роль энергетики в структуре современных профессий;
- заинтересовать юношей и девушек проектированием жизненных и профессиональных планов, особенностями будущей профессии, возможными путями достижения высокой профессиональной квалификации;
- самостоятельно логично и последовательно организовать учебный процесс;
- развить навыки исследовательской, поисковой, творческой, ролевой, прикладной (практико-ориентированной) деятельности.

В процессе освоения программы, обучающиеся получат навыки по определению актуальных проблем, проектированию их решения, постановке цели и задач, теоретической и практической реализации решения. Кроме того, проектная деятельность предполагает общение со сторонними консультантами, получение новых междисциплинарных знаний, участие в грантовых конкурсах. Все это положительно влияет на умение правильно излагать информацию и представлять работу. Такие навыки необходимы для дальнейшего развития и обучения в ВУЗах.

1.3 ЦЕЛЬ

Целью программы является разработка обучающимися проекта по альтернативной энергетике с использованием современных топливных элементов.

1.4 ЗАДАЧИ

- Развить практические навыки работы с высокотехнологичным оборудованием;
- Развить навык проектирования и построения энергосистем.
- Развить навыки сборки и работы с интерактивными стендами и моделями, топливными элементами, энергосистемами, лабораторными и промышленными образцами энергетических установок.
- Развить у обучающихся представление о работе с электронными компонентами и устройствами.
- Развить навыки сотрудничества: работа в коллективе, в команде, малой группе (в паре) при создании проекта по альтернативной энергетике.
- Развить навыки разработки концепции и идеи проектов; понимать структуру проекта; понимать систему организации человеческого труда в проектах.

1.5. ОТЛИЧИТЕЛЬНЫЕ ОСОБЕННОСТИ ПРОГРАММЫ

Данная Программа разработана в соответствии с нормативными правовыми актами в области образования:

- Федеральный закон от 29.12.2012 № 273-ФЗ «Об образовании в Российской Федерации»;
- Указ Президента Российской Федерации «О национальных целях развития Российской Федерации на период до 2030 года»;
- Концепция развития дополнительного образования до 2030 года (утверждена распоряжением Правительства РФ от 31.03.2022 № 678-р);

- Стратегия развития воспитания в Российской Федерации на период до 2025 года (утверждена распоряжением Правительства Российской Федерации от 29.05.2015 № 996-р);
- План мероприятий по реализации в 2021 - 2025 годах Стратегии развития воспитания в Российской Федерации на период до 2025 года (утвержден распоряжением Правительства Российской Федерации от 12.11.2020 № 2945-р);
- Приказ Министерства образования и науки Российской Федерации от 23.08.2017 № 816 «Об утверждении Порядка применения организациями, осуществляющими образовательную деятельность, электронного обучения, дистанционных образовательных технологий при реализации образовательных программ»;
- Приказ Министерства просвещения России от 09.11.2018 № 196 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам»;
- Приказ Министерства просвещения РФ от 03.09.2019 № 467 «Об утверждении Целевой модели развития региональных систем дополнительного образования детей»;
- Приказ Министерства Просвещения Российской Федерации от 30.09.2020 № 533 «О внесении изменений в порядок организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам, утвержденный приказом Министерства Просвещения Российской Федерации от 09.11.2018 № 196»;
- Постановление Главного государственного санитарного врача РФ от 28.09.2020 № 28 «Об утверждении СП 2.4.3648-20 «Санитарно-эпидемиологические требования к организациям воспитания и обучения, отдыха и оздоровления детей и молодежи»;
- Письмо Министерства образования и науки РФ от 18.11.2015 № 09-3242 «О направлении информации» (с «Методическими рекомендациями

по проектированию дополнительных общеразвивающих программ (включая разноуровневые программы)».

В рамках программы, обучающиеся произведут сборку существующих моделей популярных энергетических решений, а также разработают собственные проектные решения. Работа над проектами обучающихся по данной программе проходит в режиме, разработанном самими обучающимися. Обучающиеся приобретают ценные навыки командной работы, целеполагания и постановки задач, рассматривают данный процесс с точки зрения методологии SCRAM и реализуют решение проблемы и рассчитывают его экономическую целесообразность.

Вышеперечисленные пункты позволяют подготовить обучающихся к работе над крупными и локальными проектами для энергообеспечения.

1.6 ТРЕБОВАНИЯ К ОБУЧАЮЩИМСЯ

АДРЕСАТ ПРОГРАММЫ, ТРЕБОВАНИЯ К ОБУЧАЮЩИМСЯ, ВОЗРАСТНЫЕ ОСОБЕННОСТИ ГРУППЫ

Программа адресована подросткам 14-18 лет, прошедшим обучение по ДООП «Энергетика» и «Энергетика 2». В связи с ориентированностью программы на разработку индивидуальных (групповых) проектов максимальное количество обучающихся в группе не должно превышать 12 человек.

Набор на Программу осуществляется в соответствии с Порядком организации образовательной деятельности по дополнительным общеобразовательным общеразвивающим программам и Правилами приема и отчисления обучающихся автономной некоммерческой организации «Красноярский детский технопарк «Кванториум».

1.7 ОБЪЕМ/СРОК ОСВОЕНИЯ ПРОГРАММЫ

Программа рассчитана на 1 год обучения. Годовая нагрузка на обучающегося составляет 144 часа.

1.8 ФОРМА ОБУЧЕНИЯ, ВИДЫ ЗАНЯТИЙ И РЕЖИМ ЗАНЯТИЙ

Учебные занятия проходят по очной форме обучения. Режим занятий – 2 раза в неделю по 2 академических часа (1 академический час 40 минут) с обязательным перерывом.

При проведении занятий используются комбинированные занятия – изложение нового материала, проверка пройденного материала, закрепление полученных знаний, самостоятельная работа.

При проведении занятий используются три формы работы:

- демонстрационная, когда обучающиеся слушают объяснения педагога и наблюдают за демонстрационным экраном или экранами компьютеров на ученических рабочих местах;
- фронтальная, когда обучающиеся синхронно работают под управлением педагога;
- самостоятельная, когда обучающиеся выполняют индивидуальные задания в течение части занятия.

Повторение и усвоение пройденного материала осуществляется через контрольные и проверочные работы, анализ полученных результатов.

Закрепление знаний, умений и навыков через постановку задачи и самостоятельную работу обучающегося под руководством педагога.

Применение полученных знаний и навыков через прикладную работу обучающегося, использующего на практике приобретенные компетенции.

В качестве основного метода обучения используется проектный метод.

1.9 ОЖИДАЕМЫЕ РЕЗУЛЬТАТЫ И СПОСОБЫ ИХ ПРОВЕРКИ

Особенностью программы является то, что она, будучи междисциплинарная, направлена на развитие у обучающихся навыков разработки проектов по альтернативной энергетике, технологии приборостроения с использованием современных топливных элементов.

В рамках программы развиваются следующие компетенции Soft и Hard skills:

Кластер профильных soft skills

В данный кластер попадают те компетенции, которые необходимы для управления проектами и своей деятельностью в энерджиквантуме, как базовым предметом собственной «профессиональной» деятельности.

- Разработка проектов. Способность разрабатывать концепции и идеи проектов; понимать логику и методологию проектирования; разбираться в проектных подходах; осуществлять проектное описание; понимать структуру проекта; понимать систему организации человеческого труда в проектах.
- Работа с рисками. Способность прогнозировать риски; сценировать риски; вырабатывать пути предотвращения рисков; оценивать риски; описывать риски.
- Работа с экономическим планированием проекта. Способность рассчитать себестоимость, плановую цену на этапах реализации и т.д.
- Работа в команде. Способность организовывать и создавать человеческие кооперации; способность построить систему разделения труда; способность оценить человеческий потенциал.

Кластер личностных soft skills

В данный кластер попадают те компетенции, которые необходимы для управления возникающими ситуациями социального характера.

- Переговороспособность и убедительность. Способность вести переговоры с разными субъектами деятельности и оказывать влияние в процессе реализации деятельности и при проведении переговоров.
- Лидерство. Способность создать команду высокой продуктивности; создать и поддерживать эффективные отношения беря на себя ответственность за достижение целей.
- Креативность. Умение видеть и создавать композиционные элементы в любом аспекте жизни; способность к абстрактному творчеству.
- Рефлексивность. Способность производить оценку совершенным действиям.

Кластер контекстуальных soft skills

В данный кластер попадают те компетенции, которые необходимы для обеспечения деятельности:

- Стратегическое и тактическое мышление. Способность удерживать аспект стратегирования и тактики в работе.

Кластер Hard skills

В рамках программы развиваются следующие профессиональные навыки и знания:

- Знания основных понятий электроники.
- Знания работы электронных компонентов.
- Знания элементов электронного взаимодействия узлов радиоэлектронных устройств.
- Знания основных принципов и приемов проектирования электронных систем;
- Навыки самостоятельного решения технических задач в процессе конструирования энергетических систем.
- Навыки изложения логически правильных действий модели (проекта).

- Навыки моделирования технических устройств, энергоузлов, энергосистем.
- Навыки работы с дополнительной литературой, с журналами, с каталогами, в интернете (изучение и обработка информации).
- Навыки демонстрирования технических возможностей созданных проектов.
- Навыки подготовки и форматирования текста в MS Word, создания презентаций в MS Powerpoint.

1.10 ФОРМЫ ПОДВЕДЕНИЯ ИТОГОВ ОБУЧЕНИЯ

Оценка уровня владения проводится преподавателем в процессе выполнения обучающимся собственного итогового проекта / участия в отборочных этапах мероприятий, входящих в Календарь Всероссийских мероприятий в сфере дополнительного образования детей и/или Рекомендованные региональные мероприятия в сфере дополнительного образования детей технической направленности ФГБОУ ДО ФЦДО.

По итогам каждого ключевого раздела проводится промежуточная аттестация в форме проверочной работы.

Аттестация по итогам освоения программы проводится в середине и в конце года и представляет собой предзащиту/защиту индивидуального или группового проекта по разработке и реализации моделей устройств и систем резервного или постоянного электропитания в энергетике (Energy-Net) или теоретических проектов перспективной направленности.

Технология проведения итогового контроля - экспертная оценка в рамках защиты проекта. В ней принимает участие преподавательский состав Кванториума. Конкретный пул экспертов формируется в ходе прохождения этапа подготовки проекта к презентации, что позволяет участникам получить экспертную обратную связь относительно представленного проекта, а также понять, через комментарии экспертов, перспективы развития проекта.

2. УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН

№ п/п	Наименование разделов и тем	Количество академический часов			Форма контроля
		всего	теория	практика	
1	Соблюдение правил техники безопасности и санитарно-гигиенических норм при работе с электрооборудованием и учебно-методическими материалами	2	2		
	Раздел 1. «Программирование на платформе Arduino»	**38**	**19**	**19**	
2	Arduino Nano Brick- основные операции.	8	4	4	
3	Аналого/цифровой преобразователь	8	4	4	
4	Шина I2C	6	3	3	
5	Реле	2	1	1	
6	Поворотный энкодер	2	1	1	
7	OLED-дисплей	6	3	3	
8	Цифро-аналоговый преобразователь	6	3	3	Проверочная работа
	Раздел 2. «Интернет вещей»	**16**	**7**	**9**	
9	Основа работы IoT	6	3	3	
10	Установка библиотек	4	2	2	
11	Схемы IoT	6	2	4	Проверочная работа
	Раздел 3. Проект.	**88**	**13**	**75**	
12	Командообразование	2	0	2	
13	Методология управления проектом.	2	1	1	
14	Реализация учебного творческого проекта (по выбору)	76	12	64	
15	Подготовка к защите проекта.	4	0	4	
16	Промежуточный контроль.	2	0	2	Предзащита проекта
17	Итоговый контроль.	2	0	2	Защита проекта
ИТОГО часов:		**144**	**41**	**103**	

3. СОДЕРЖАНИЕ ПРОГРАММЫ

1.Соблюдение правил техники безопасности и санитарно-гигиенических норм при работе с электрооборудованием и учебно-методическими материалами.

Общие правила безопасности в образовательном учреждении. Основы техники безопасности при работе с электрическими приборами. Техника безопасности при работе в лаборатории. Общие положения техники безопасности при работе с химическими реактивами. Техника безопасности при работе с лабораторными установками.

Раздел 1 «Программирование на платформе Arduino»

2. Arduino Nano Brick- основные операции.

Теория: Электроника. Техника безопасности при работе с электрическими схемами. Блоки набора Arduino. Arduino Nano. Светодиоды и кнопки. Простейшие электрические цепи.

Практика: Сборка простейших электрических цепей.

3. Аналого/цифровой преобразователь.

Теория: АЦП - принцип действия. Потенциометр. АЦП с фоторезистором LDR. Измерение температуры терморезистором.

Практика: Сборка электрических цепей. Вольтметр на АЦП и с подключением к ПК.

4. Шина I2C.

Теория: Принцип работы и команды шины I2C. Шина I2C и порт ввода-вывода. 7-сегментный дисплей - принцип действия. 7-сегментный дисплей на шине I2C - принцип действия.

Практика: Сборка электрических цепей. 7-сегментный дисплей - простой счётчик.

5. Реле.

Теория: Принцип работы герконового реле. Герконовое реле для управления дисплеем.

Практика: Сборка электрических цепей. Таймер со срабатыванием от герконового реле.

6. Поворотный энкодер.

Теория: Блок поворотного энкодера. Поворотный энкодер с отображением значений. Поворотный энкодер с 7-сегментным дисплеем на выходе.

Практика: Сборка электрических цепей. Поворотный энкодер с отображением значений.

7. OLED-дисплей

Теория: Принцип работы графического дисплея. Библиотека блока OLED. Блок OLED с шиной I2C. Блок OLED и набор символов. OLED-дисплей с АЦП для измерения напряжения.

Практика: Сборка электрических цепей. OLED дисплей с АЦП в сборке мини-осциллографа. OLED-дисплей с блоком АЦП в сборке двойного вольтметра.

8. Цифро-аналоговый преобразователь

Теория: Принцип работы цифро-аналогового преобразователя. Простой ЦАП на принципе Широтно-Импульсной Модуляции(ШИМ). ЦАП с управлением по шине I2C.

Практика: Сборка электрических цепей. Блок ЦАП с потенциометром. Блок OLED и ЦАП с АЦП. Блок OLED и ЦАП с АЦП с отображением синусоиды.

Раздел 2 «Интернет вещей»

9. Основа работы IoT

Теория: Блок заземления. Блок питания. Блок IoT и среда разработки Arduino. Основа набора IoT. Характеристики блока IoT. Контакты GPIO. Среда разработки Arduino.

Практика: Сборка простейших электрических цепей из блоков IoT.

10. Установка библиотек.

Теория: Установка библиотек Arduino. Драйвер виртуального COM-порта. Монитор последовательного интерфейса.

Практика: Сборка ночника с детектором движения. Установка соединения. Компиляция и загрузка программ. Режим программирования.

11. Схемы IoT.

Теория: OLED-дисплей – вывод текста. Фактор коррекции.

Практика: Конфигурация блока IoT в роли WiFi-клиента. Получение времени по интернету. Курс валют из интернета.

Раздел 3. Проект.

12. Командообразование.

Практика: Тест на определение роли в команде. Игра на командообразование.

13. Методология управления проектом.

Теория: Планирование проекта. Основы целеполагания. Методология SCRAM. Методология Kanban.

Практика: Заполнение плана реализации в программе Trello.

14.Реализация учебного творческого проекта (по выбору).

Теория: Основы экономического планирования. Определение целевой группы.

Практика: Выбор тематики и направлений развития в команде для решения проблем "рабочего" проекта. Определение проблемы, цели и задач. Определение целевой группы, актуальности проекта. Финансово-экономическое планирование. Определение рисков. Изготовление модели/макета/прототипа. Реализация проекта. Подготовка паспорта проекта (аннотация проекта, техническая значимость).

15. Подготовка к защите проекта.

Практика: Оформление презентационного материла.

16. Промежуточный контроль.

Практика: Защита проектной идеи.

17. Итоговый контроль.

Практика: Защита проектной идеи. Защита проекта.

4. СПИСОК ЛИТЕРАТУРЫ.

4.1. ДЛЯ НАСТАВНИКОВ

Книги:

1. В.Е. Фортов, О.С. Попель. «Энергетика в современноммире», ИД «Интеллект», 2011.

2. В.Е. Форотов, О.С. Попель. «Возобновляемая энергетика в современном мире», МЭИ, 2015.

3. А. да Роза. «Возобновляемые источники энергии. Физико-технические основы», ИД «Интеллект», 2010.

4. Б. Соренсен. «Преобразование, передача и аккумулирование энергии», ИД «Интеллект», 2011.

5. Даффи Дж. «Основы солнечной теплоэнергетики», ИД «Интеллект», 2013.

6. В.В. Тетельмин. «Физические основы традиционной и альтернативной энергетики», ИД «Интеллект», 2016.

7. В.К. Власов. «Полезный ветер. От паруса до...», ИД «Интеллект», 2017.

8. Ю.А. Котляр, В.В. Шинкаренко. «Водородный всеобуч в России. К истории вопроса. Документы. Материалы. Комментарий», АСМИ, 2008.

9. О.Е. Аверченков. «Схемотехника: аппаратура и программы», ДМК Пресс, 2012.

10. Ф.А. Ткаченко. «Электронные приборы и устройства», ИНФРА-М, 2011.

11. Энерджиквантум: тулкит.

Периодические издания:

«Наука и жизнь»;

«Популярная механика»;

«Техника молодёжи».

4.2. ДЛЯ ОБУЧАЮЩИХСЯ

Книги:

1. В.Е. Фортов, О.С. Попель. «Энергетика в современном мире», ИД «Интеллект», 2011;

2. В.Е. Форотов, О.С. Попель. «Возобновляемая энергетика в современном мире», МЭИ, 2015.

3. К. Пиковер. «Великая физика. От Большого взрыва до Квантового воскрешения. 250 основных вех в истории физики», Лаборатория знаний, 2015.

4. В.К. Власов. «Полезный ветер. От паруса до...», ИД «Интеллект», 2017.

5. Ю.А. Котляр, В.В. Шинкаренко. «Водородный всеобуч в России. К истории вопроса. Документы. Материалы. Комментарий», АСМИ, 2008.

6. О.Е. Аверченков. «Схемотехника: аппаратура и программы», ДМК Пресс, 2012.

7. Ф.А. Ткаченко. «Электронные приборы и устройства», ИНФРА-М, 2011.

Периодические издания:

«Наука и жизнь»;

«Популярная механика»;

«Техника молодёжи».

5. ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ И МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОСНАЩЕНИЕ ДОПОЛНИТЕЛЬНОЙ ОБЩЕОБРАЗОВАТЕЛЬНОЙ ОБЩЕРАЗВИВАЮЩЕЙ ПРОГРАММЫ «ЭНЕРГЕТИКА 3»

№ п/п	Название	Автор	Год издания (создания)	Вид (электронный, печатный)
Методические пособия				
1.	ADVANCED SET MANUAL	https://www.brickrknowledge.de/en/	2018	Электронный
2.	ARDUINO SET MANUAL	https://www.brickrknowledge.de/en/	2018	Электронный
3.	Теоретическая физика	Ландау, Л. Д., Лифшиц, Е. М. М. Физматлит.	2004	Электронный
4.	INTERNET OF THINGS SET MANUAL	https://www.brickrknowledge.de/en/	2018	Электронный
Материально – техническое обеспечение				
1.	Учебно-методический стенд «Водородная энергетика»	5 шт	2016	
2.	Напольная вентиляционная установка для имитации ветра в лаборатории	3 шт	2016	
3.	Напольно-настольная установка для имитации солнечного света в лаборатории	4 шт	2016	
5.	Стенд «Интеллектуальные энергетические системы»	1 шт	2016	

6.	Система практического использования топливных элементов	2 шт	2016	
7.	Комплект водородной энергетики для класса робототехники, артикул ВЭКР-8	2 шт	2016	
8.	Генератор водорода малой мощности для школьной лаборатории	1 шт	2023	
9.	Учебный набор от Brick'R'knowledge Advanced Set	4 шт	2018	
10.	Учебный набор от Brick'R'knowledge SOLAR SET	4 шт	2018	
11.	Учебный набор от Brick'R'knowledge INTERNET OF THINGS SET	4 шт	2018	
12.	Учебный набор от Brick'R'knowledge ARDUINO CODING SET	4 шт	2018	
13.	Учебный набор от Brick'R'knowledge LOGIC SET	4 шт	2018	
14.	Ноутбуки	12 шт	2017	
15.	Проектор	1 шт	2016	
16.	Столы, стулья, шкафы и стеллажи для хранения		2017	

6. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

Критерии оценивания промежуточных проверочных работ:

Критерий	Оценка
Электронная схема собрана правильно и выполняет необходимую функцию. Корректно прописан программный код. Верно сделан вывод по работе схемы.	зачет
Электронная схема не работает. Не сформулирован вывод по итогу работы.	незачет

Критерии оценки проектов

№ п/п	Объект оценки	Критерии Баллы	Баллы
1.	Оценка созданного изделия	Новизна и актуальность темы проекта	от 0 до 10
		Привлекательность и оригинальность внешнего вида созданного изделия	от 0 до 10
		Работоспособность и (или) возможность для внедрения изготовленного изделия	от 0 до 10
		Качество изготовленного изделия	от 0 до 10
		Перспективность и конкурентоспособность созданного изделия	от 0 до 10
2.	Оценка описания проекта	Исследование проблемы проекта	от 0 до 7
		Анализ аналогов проекта	от 0 до 7
		Формулировка цели и задач проекта	от 0 до 7
		Исследование целевой группы (аудитории)	от 0 до 7
		Обоснование экономической составляющей процесса создания изделия	от 0 до 7
		Соответствие результата проекта поставленной цели	от 0 до 7
		Дальнейшее развитие проекта	от 0 до 7
3.	Оценка защиты проекта	Соблюдение регламента публичной защиты	от 0 до 5
		Качество подачи материала и представления	от 0 до 5
		Понимание сути задаваемых вопросов и аргументированность ответов докладчика	от 0 до 5
		Качество презентации и презентационных материалов	от 0 до 5

Критерии оценивания промежуточной и итоговой аттестации:

Аттестация	Количество балов	Оценка
промежуточная	46-58	высокий
	37-45	средний
	29-36	низкий
	менее 28	не аттестация
итоговая	95-119	высокий
	77-94	средний
	59-76	низкий
	менее 58	не аттестация

7. ПРИЛОЖЕНИЯ

Приложение 1

Примерный перечень тем для проектов:

- Технологии «зеленой энергетики» в России и мире.
- Использование жидкостномембранных топливных элементов в условиях Красноярского края.
- Анализ энергетических систем города Красноярска на примере Октябрьского района.
- Методы устройства энергетических систем.
- Использование альтернативных источники энергии в северных районах Красноярского Края.
- Изучение влияния годичных температурных колебаний в г. Красноярске на рабочие характеристики водородных топливных элементов.
- Обеспечение энергией удаленных районов Красноярского края.
- Методы хранения водорода в промышленных масштабах.
- Использование высокоэффективных конденсаторов в городских энергетических системах.
- Методы накопления энергии солнца и ветра в Сибири.
- Исследование погодных условий для разработки эффективного ветряного генератора в Красноярском крае.
- Разработка ветряного генератора повышенной эффективности.
- Повышение КПД систем энергообеспечения электромобилей.
- Использование металлогидридных водородных аккумуляторов в автомобильной промышленности.
- Разработка универсального зарядного устройства на принципах альтернативной энергетики для гаджетов.
- Использование термоэлектрических генераторов в быту.
-

Приложение 2

Пример проверочных работ по итогам разделов.

Проверочная работа. Раздел 1. «Программирование на платформе Arduino».

Задание.

Соберите схему для графического отображения процесса зарядки и разрядки конденсатора. Сделайте вывод о перемещении кривой на дисплее.

Примечание:

Конденсатор заряжается за 3 секунды и разряжается за такое же время. На дисплее мини-осциллографа отображается кривая зарядки и разрядки. Для определения точного времени используйте команду millis(), считающая время, прошедшее после запуска программы в миллисекундах.

Постоянную времени зарядки или разрядки можно рассчитать по формуле: t=R*C, имеем 100000 Ом*10E-6E = 1 с. Это время, за которое конденсатор заряжается или разряжается на 63 %.

Необходимый результат.

На дисплее отображается график зарядки и разрядки конденсатора. Кривая плавно перемещается по дисплею.

Возможное решение.

```
// RU_37_Applications_dischargecurve

int milisec=0; // метка для миллисекунд
void setup() {
  Wire.begin(); // инициализация I2C
  i2c_oled_initall(i2coledssd); // инициализация OLED
  for (int i=0; i<64; i++) advalbuf[i]=47; // значение по умолчанию
}

void loop() { //
  // OLED-дисплей 64x48 пикселей
  static int cxx = 0; // циклический счётчик
  static int state = 0; // использование системы состояний
  int ms = 0; // время измеряется в мс
  static int daval = 0; // выходное значение
  int ana0 = analogRead(A0); // считывание канала 0
  char buffer[40]; // ASCII-буфер для показаний в вольтах
  disp_buffer_clear(COLOR_BLACK); // очистка экрана
  double p1 = (ana0*5000.0)/1023.0; // в мВ
  int y1 =0; // положение y
  sprintf(buffer, "A0=%d.%03dV",(int)p1/1000,(int)p1%1000);
  y1 = 47 - (p1 * 30.0)/5000.0; // 5 В максимум -> 30 пикселей
  advalbuf[cxx++] = y1; // хранение 0–4xx В +-128
  if (cxx >63) cxx =0; // циклический счётчик
  disp_print_xy_lcd(2, 0, (unsigned char *)buffer, COLOR_WHITE, 0);
  int i=0; // счётчик столбцов
  int yold = advalbuf[(cxx+1)%64]; // предыдущее значение
  for (i=0; i<63; i++) { // вывод всех столбцов
    y1 =advalbuf[(cxx+1+i)%64]; // циклический буфер
    disp_line_lcd (i, yold, i, y1, COLOR_WHITE);
    yold = y1; // сохранить в старое значение
  }
  disp_lcd_frombuffer(); // вывод на дисплей графика зарядки и разрядки конденсатора
  // с фиксированным интервалом измерений
  //ms=millis(); // текущее время
  if (ms > (milisec+3000)) { // задержка 3 с
    switch(state) { // зависит от состояния
      case 0:
        daval = 0xfff; // зарядка
        state = 1; // следующее состояние 1
      break;
      case 1: // разрядка
        daval = 0; // задержка 3 с
        state = 0; // повтор состояния 0
        break;
      default: // на всякий случай
        state = 0; // безопасное состояние 0
    }
    milisec = ms; // новое время
  }
  // ЦАП
  i2c_da_write_command(i2cdasel1,daval);
  i2c_da_write_command(i2cdasel2,daval);
  i2c_da_write_command(i2cdasel3,daval);
}
```

Рисунок 1.1. – Программный код

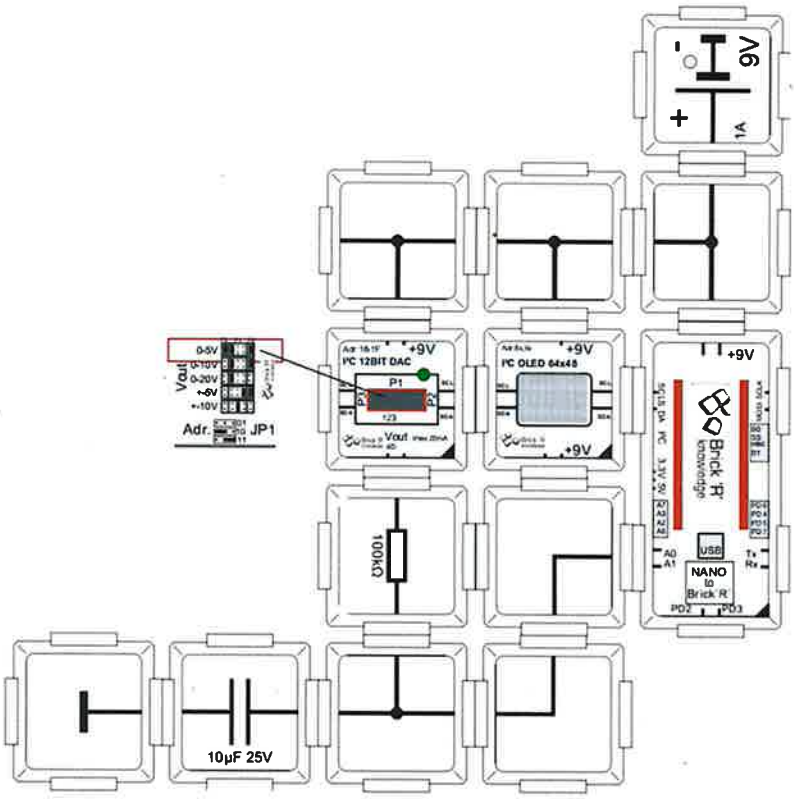

Рисунок 1.2 – Построение графика разрядки – модуль OLED

Проверочная работа. Раздел 2. «Интернет вещей».

Задание.

Соберите схему для измерения температуры и влажности.

Примечание.

Сначала необходимо ввести имя сети WiFi (SSD) и пароль от сети в соответствующие поля программы.

Для измерения температуры и влажности используqnt широко распространённый датчик DHT11, для которого. существует заранее подготовленная библиотека (DHT.h) и примеры кода. Помимо указанного, для Arduino существует масса датчиков, например:

- датчики температуры
- датчик газа
- инфракрасный фотодатчик
- датчик Холла
- датчик движения (ИК-сенсор)
- датчик вибрации
- датчик света (LDR)
- датчик срабатывания

Блок универсального адаптера (ALL-BRICK-0649) обеспечивает простоту подключения датчиков. Для многих из них существуют библиотеки и примеры программ, облегчающие их внедрение в схемы.

Необходимый результат.

На дисплей выводятся корректные данные влажности и температуры.

Возможное решение.

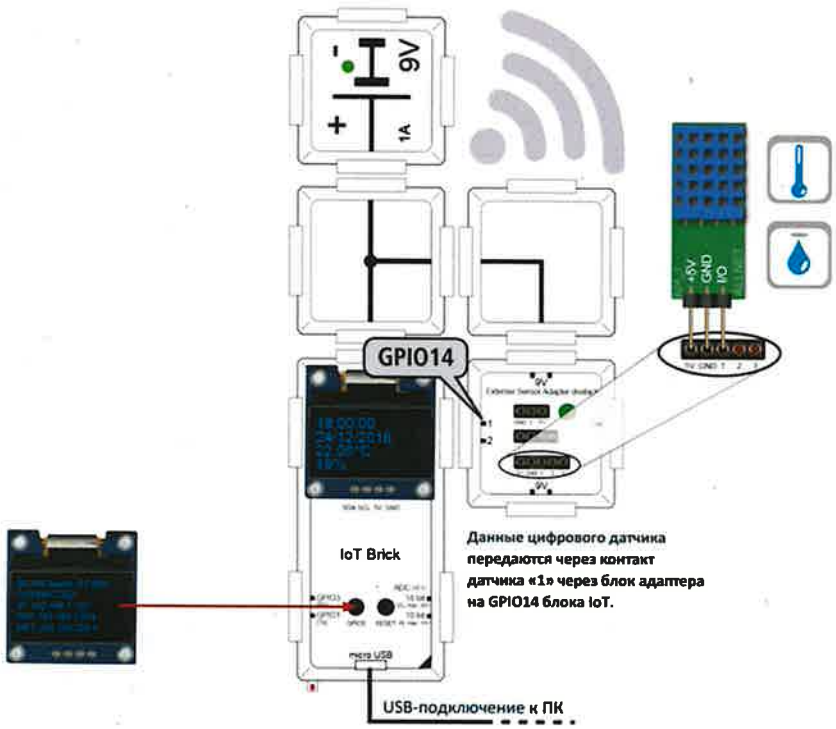

Рисунок 2.1 – Схема измерения температуры и влажности

Программа

```
#define DHT_TYPE DHT11     //определение типа датчика: DHT11
const int DHT_PIN = 14;  //переменная хранения данных GPIO4 блока IoT
char temp[20];           //строковая переменная для температуры
char humi[20];           //строковая переменная для влажности

DHT dht(DHT_PIN, DHT_TYPE);   //задание переменной типа DHT

…
void setup() {
…
   dht.begin();                //инициализация датчика
} void

loop() {

…
 if (counter%1000==0){ //Считывание данных датчика приблизительно раз в секунду
    float t = dht.readTemperature(); //Чтение показаний температуры(Цельсии)
    float h = dht.readHumidity();    //Чтение показаний влажности

    sprintDouble(temp,t,2);  //Преобразование температуры в тип string с
                             //двумя десятичными знаками после запятой
    sprintDouble(humi,h,0);  //Преобразование влажности в тип string
    strcat(temp," °C");      //добавление символа °C к температуре
    strcat(humi," %");       //добавление символа % к влажности

  }
  display.drawString(5, 30, temp);   //Подготовка вывода температуры
display.drawString(5, 45, humi);   //Подготовка вывода влажности
display.display();                 //обновление дисплея
…
}
```

Рисунок 2.2 – Программный код

Примеры практических работ по темам.

Раздел 1. «Программирование на платформе Arduino»

Тема 2. Arduino Nano Brick- основные операции.

Соберите схему «бегущий огонёк». Для этого в неё следует добавить модуль с двумя светодиодами, а также специальные модули для соединения нижних портов PD6 и PD7 со светодиодами. Также обратите внимание на назначение портов PD6 и PD7. Если двойной Т-образный соединительный модуль вставлен с другой ориентацией, «бегущий огонёк» не сможет «бегать». Обратите внимание на то, что LED 6 связан с битом 0, а LED 7 – с битом 1.

Что должно происходить: всегда горит только один из шести светодиодов, при этом свет циклически перемещается справа налево. Время свечения каждого светодиода 300 мс.

Возможное решение.

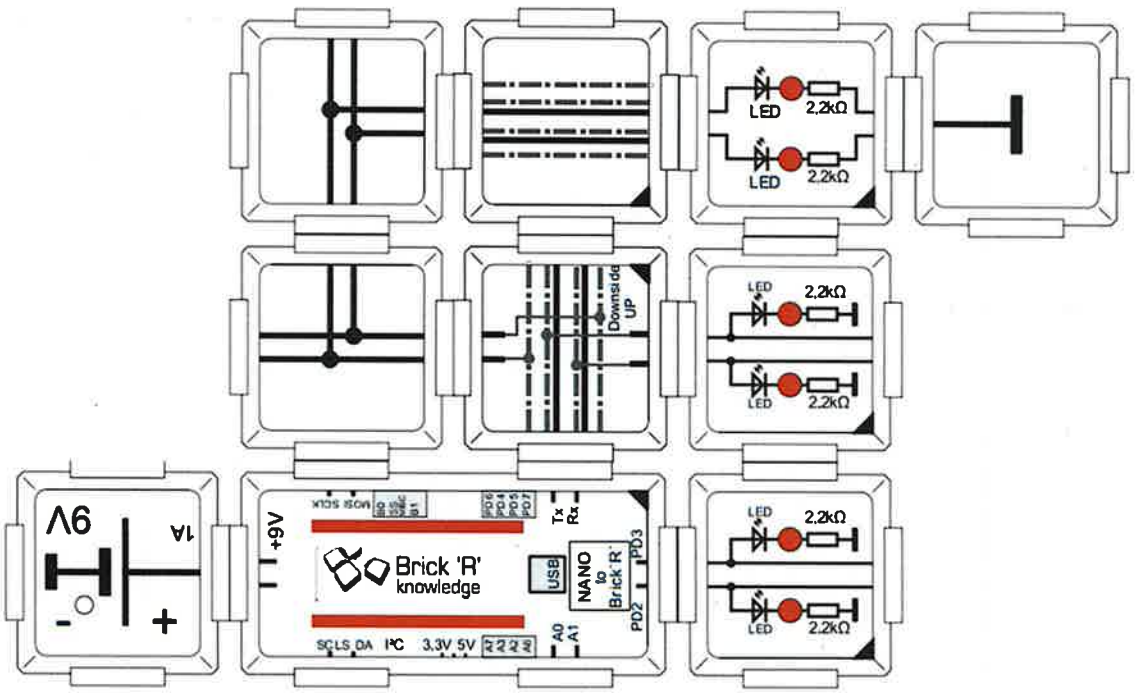

Рисунок 3.1.1 – Схема подключения

Программный код:

```
// RU_7_Chasing_lights_using_6_LEDs (Простой «бегущий огонёк»)

#define PORTLED2 2 // назначение портов PD2–PD7 для светодиодов
#define PORTLED3 3 // обратите внимание на соблюдение соответствия
#define PORTLED4 4 //
#define PORTLED5 5 //
#define PORTLED6 6 //
#define PORTLED7 7 //

void setup() { // задание входов и выходов
pinMode(PORTLED2,OUTPUT); // порт 2 в качестве выхода
  pinMode(PORTLED3,OUTPUT); // порт 3 в качестве выхода
  pinMode(PORTLED4,OUTPUT); // порт 4 в качестве выхода
  pinMode(PORTLED5,OUTPUT); // порт 5 в качестве выхода
  pinMode(PORTLED6,OUTPUT); // порт 6 в качестве выхода
  pinMode(PORTLED7,OUTPUT); // порт 7 в качестве выхода
}

void loop() {
  static int shiftreg = 1; // для «бегущего огонька» используются 6 бит
    if (shiftreg & 1) { // проверка битов 0–5 один за другим
     digitalWrite(PORTLED6,HIGH); // правый светодиод
    } else digitalWrite(PORTLED6,LOW); //
    if (shiftreg & 2) { // проверка бита 1
     digitalWrite(PORTLED7,HIGH); // левый светодиод
    } else digitalWrite(PORTLED7,LOW); //
    if (shiftreg & 4) { // проверка бита 2
     digitalWrite(PORTLED4,HIGH); //
    } else digitalWrite(PORTLED4,LOW); //
    if (shiftreg & 8) { // проверка бита 3
     digitalWrite(PORTLED5,HIGH); //
    } else digitalWrite(PORTLED5,LOW); //
    if (shiftreg & 0x10) { // проверка бита 4
     digitalWrite(PORTLED3,HIGH); //
    } else digitalWrite(PORTLED3,LOW); //
    if (shiftreg & 0x20) { // проверка бита 5
     digitalWrite(PORTLED2,HIGH); //
    } else digitalWrite(PORTLED2,LOW); //
    //
    delay(300); // задержка 300 мс
    shiftreg = shiftreg << 1; // 1, 2, 4, 8, 16, 32 команда смещения компилятора C
    if (shiftreg > 32) shiftreg = 1; // повтор процедуры, чтобы огонёк «бегал»
} // конец цикла
```

Рисунок 3.1.2 – Программный код

Тема 3. Аналого/цифровой преобразователь

Соберите схему АЦП – измерение температуры с помощью NTC-термистора.

При использовании NTC-термистора в качестве переменного сопротивления, можно построить схему для измерения температуры. Для терморезистора формула имеет вид:

$R_T = R_N * e^{B(1/T-1/TN),}$ из этого можно вывести следующую формулу:

$R= B*T_N/(B+\ln(R_T/R_N)*T_N)$.

T_N (контрольная температура), как правило, составляет 298,15 K. Используется комнатная температура 25 °C, пересчитанная в градусы Кельвина K (298,15 K). Следовательно, имеем сопротивление терморезистора R_N, равное 10 кОм. Значение B определяется производителем (как правило, между 2000 K и 4000 K), но оно также может быть определено проведением контрольного измерения. T является измеренной в K температурой, из которой выражается B (также измеренная в K). В данной схеме мы используем резистор с сопротивлением 100 кОм для расширения делителя напряжения, чтобы иметь возможность измерять температуры от 0 °C. Реперной точкой для 0 °C является температура плавления льда. Это позволяет рассчитать зависящее от температуры значение B. Его лучше всего подбирать, исходя из диапазона измеряемых температур.

Обратите внимание: необходимо нажать комбинацию CTRL-SHIFT-M на запущенном ПО Arduino, чтобы появилось окно терминала. При необходимости нужно установить скорость передачи данных в окне терминала на 9600, так как это согласовано с нашей программой.

Что должно происходить: после активации терминала на нём появятся две строчки текста: первая – с температурой в °C, вторая – со значением

сопротивления NTC-термистора. При нагреве терморезистора (достаточно тепла рук), значение температуры будет повышаться, сопротивления – падать.

Программный код:

```
// RU_11_AD_converter_and_NTC (АЦП – измерение температуры с помощью NTC-термистора)

// нажмите CTRL-SHIFT-M на ПК в окне Arduino для открытия терминала


#define PORTAD0 0 // выбор канала 0
// вывод производится на терминал ПК
// в следующих программах используется дисплей
void setup() { // начало
  Serial.begin(9600); // скорость передачи данных
  // передача данных на ПК
  // 9600 бод = 9600 бит/с
}

void loop() { // начало цикла
  int value; // значение с АЦП
  double Vdivider,RNTC; // напряжение на делителе и сопротивление NTC-термистора
  value = analogRead(PORTAD0); // опрос АЦП
  // пошаговое преобразование 0–1023 до максимального значения 5 В
  Vdivider = value * 5.0 / 1023.0;
  // пересчёт напряжения на делителе в сопротивление
  // измерьте напряжение батареи заранее, здесь предполагается, что напряжение равно 9 В
  double VBatt = 9.0;
  // если напряжение батареи отличается от 9 В, это значение необходимо настроить
  double Rdivider = 100000.0; // сопротивление делителя 100 кОм
  RNTC = (Vdivider*Rdivider)/(VBatt-Vdivider); // сопротивление NTC
  // расчёт температуры
  double B = 3800.0; // отличается для каждого терморезистора, необходимо индивидуальное определение
  double RN = 10000.0; // 10 кОм
  double TN = 298.15; // 25 °C в Кельвинах
  double T = (B*TN)/(B+log(RNTC/RN)*TN)-273.15; // T в °C
  Serial.print(T); // вывод значения температуры
  Serial.print("Grad C"); // вывод строки: Grad C
  Serial.print(RNTC); // вывод значения сопротивления NTC на экран
  Serial.println("Ohm"); // вывод символа Ohm (Ом)
}
```

Рисунок 3.2.1 – Программный код

Возможное решение.

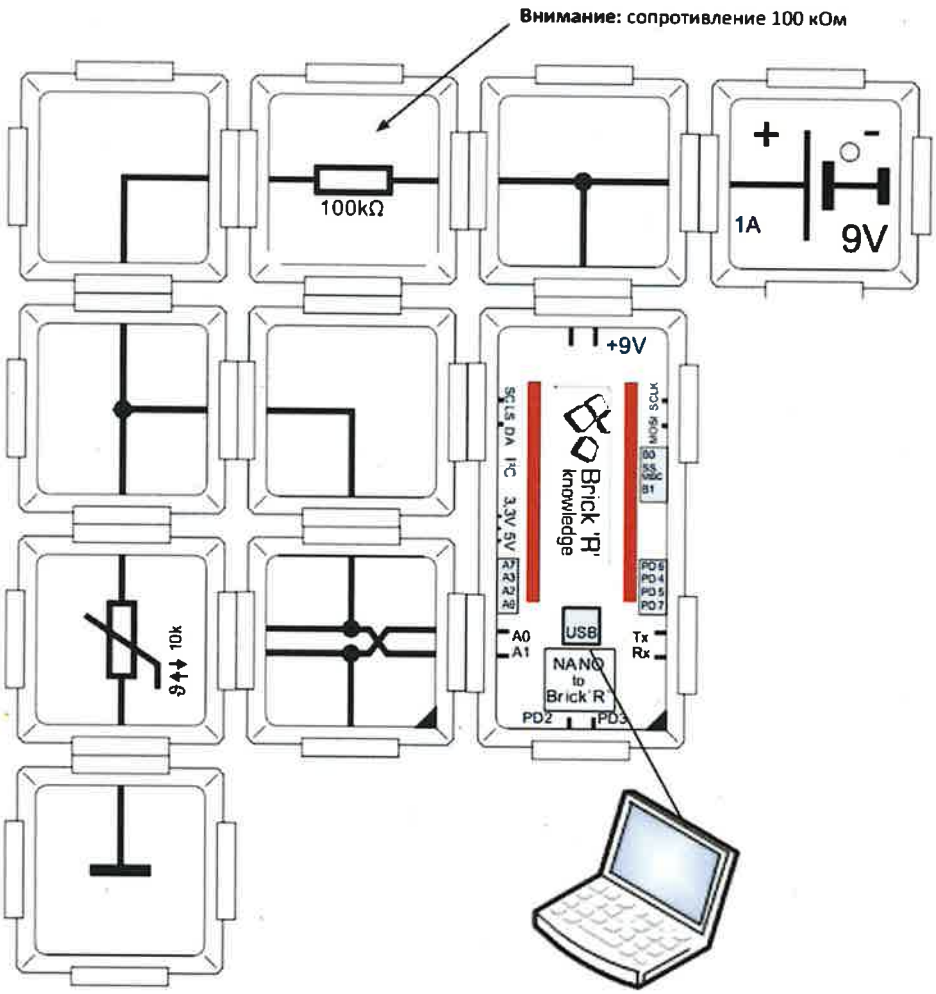

Рисунок 3.2.2 – Схема подключения

Тема 4. Шина I2C

Соберите схему 7-сегментный дисплей (простой счётчик). Для хранения значений счётчика существует переменная «counter». Каждые 500 мс программа должна увеличить значение счётчика на 1. Имея всего один модуль 7-сегментного дисплея, можно отобразить значение только до 99. По этой причине значение счётчика сравнивается с 99 и сбрасывается на 0 при превышении этого числа, счёт начинается заново.

Выполнение цикла занимает чуть больше 500 мс, хотя вызов «delay(500)» довольно точный, к этому времени добавляется время выполнения других команд. Для более точной работы Вам может понадобиться таймер.

Что должно происходить: 7-сегментный дисплей выводит последовательно числа 00, 01, 02, ..., 99. Смена значения происходит каждые полсекунды. После достижения значения 99 счётчик сбрасывается на 0.

Возможное решение:

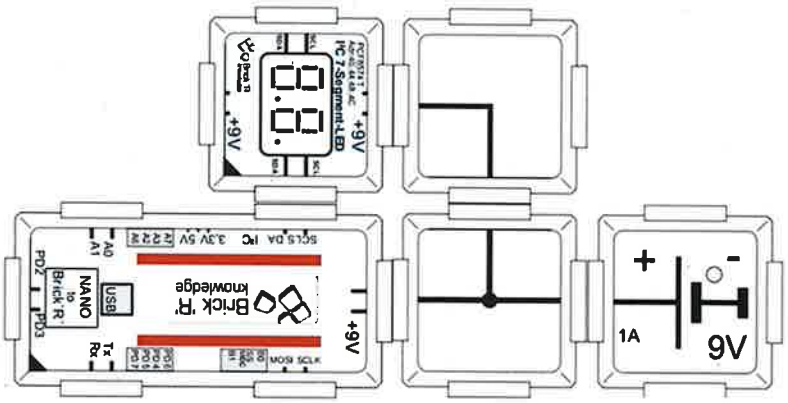

Рисунок 3.3.1 – Схема подключения

Программный код:

```
// RU_16_SevenSegmentDisplay_count – display_on_I2C_Brick (7-сегментный дисплей – простой счётчик)

#include <Wire.h>

// по умолчанию 8574T
#define i2cseg7x2alsb1 (0x40>>1)
.....
см. предыдущий пример RU_15
.....

// запуск один раз, чтобы инициализировать I2C
void setup() {
  Wire.begin(); // инициализация библиотеки I2C
}

void loop() {          // начало цикла
  char buffer[10];     // использование буфера символов
  static int counter = 0; // статическая переменная для счётчика с началом в 0
  sprintf(buffer,"%02d",counter++); // преобразование типа int в char
  if (counter >99) counter = 0; // запуск счётчика от 0 до 99
  // отображение счётчика в виде двух цифр из буферов 0 и 1
  display_seg1x(i2cseg7x2amsb1,buffer[0]); // символ MSB (наиболее значимого бита)
  display_seg1x(i2cseg7x2alsb1,buffer[1]); // символ LSB (наименее значимого бита)
  delay(500); // счёт каждые 500 мс (примерно)
} // конец цикла
```

Рисунок 3.3.2 – Программный код

Тема 5. Реле.

Создайте схему, в которой с помощью герконового реле можно управлять дисплеем. При приближении магнита значение счётчика увеличивается на единицу. Если прикрепить магнит к колесу, можно посчитать число оборотов. Как правило, герконовые контакты стоит нейтрализовывать от дребезга, хотя их срабатывание и происходит очень быстро. Для реализации счётчика оборотов используйте стандартную программу счёта.

Что должно произойти: если магнит приблизится и затем отдалится от герконового контакта (магнит должен быть достаточно мощным, и его поле должно быть направлено в нужную сторону), значение счётчика увеличится и отобразится на дисплее.

Программный код:

```
// RU_22_SevenSegment_Counter_with_Reed_Relais_I2C (Управление дисплеем с помощью герконового реле)
#include <Wire.h>

// по умолчанию 8574T
#define i2cseg7x2alsb1 (0x40>>1)
#define i2cseg7x2amsb1 (0x42>>1)

..... как в предыдущих примерах ...

// новый код:

#define PORTRELAIS 2 // подключение реле к PD2

// инициализация I2C и назначение реле
void setup() {
  Wire.begin(); // инициализация I2C
  pinMode(PORTRELAIS,INPUT_PULLUP); // подключение реле
{
void loop() { // начало цикла
  char buffer[10]; // буфер для значения счётчика в ASCII
  static int counter = 0; // счётчик
  sprintf(buffer,"%04d",counter); // преобразование значения счётчика в ASCII и запись в буфер
  // опрос кнопки на дребезг
  if (digitalRead(PORTRELAIS)==LOW) { // при нажатой кнопке сигнал переходит с высокого уровня на низкий
  delay(20); // затем происходит задержка 20 мс до окончания дребезга
    while (digitalRead(PORTRELAIS)==LOW) // при нажатой кнопке сигнал переходит с низкого уровня на высокий
    counter++; // увеличение значения счётчика при размыкании контакта
    delay(20); //  затем происходит задержка 20 мс (сумма задержек
    // при замыкании и размыкании кнопки должна равняться 40 мс)
  }
  if (counter > 9999) counter = 0; // счёт до 0000, затем переход на 9999
  // отображение счётчика в виде четырёх цифр из буферов 0, 1, 2, 3
  display_seg1x(i2cseg7x2bmsb1,buffer[0]); // крайняя левая позиция
  display_seg1x(i2cseg7x2blsb1,buffer[1]); // вторая позиция слева
  display_seg1x(i2cseg7x2amsb1,buffer[2]); // третья позиция слева
  display_seg1x(i2cseg7x2alsb1,buffer[3]); // крайняя правая позиция
} // конец цикла
```

Рисунок 3.4.1 – Программный код

Возможное решение:

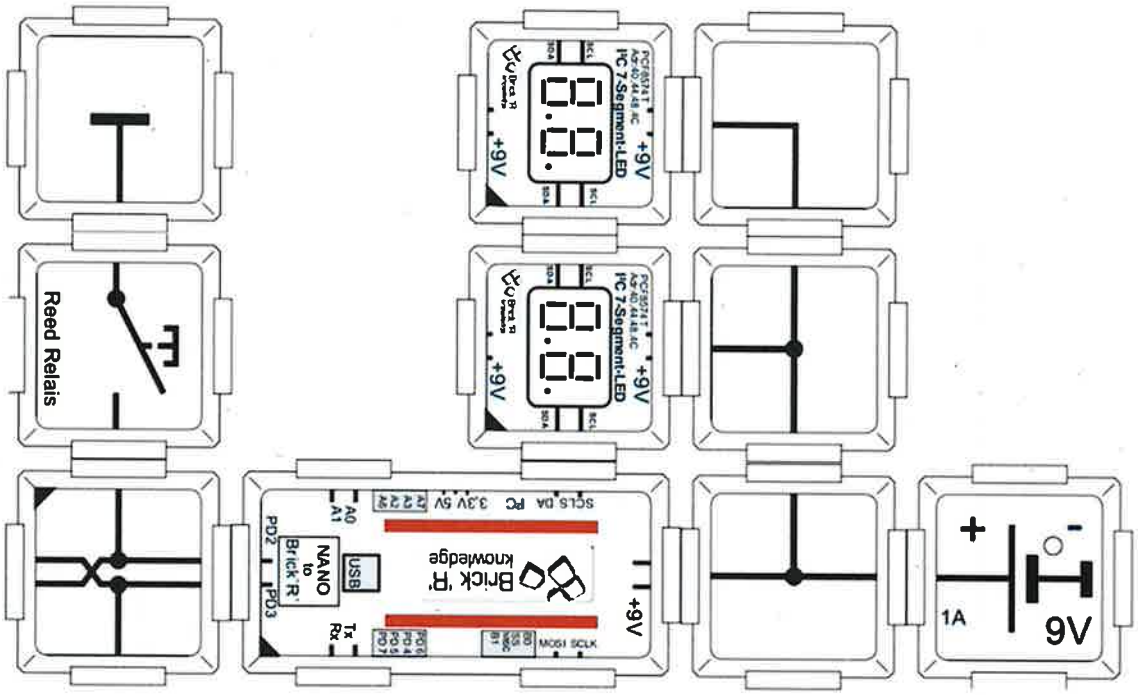

Рисунок 3.4.2 – Схема подключения

Тема 6. Поворотный энкодер

Создайте схему поворотного энкодера и 7-сегментного дисплея для вывода значений.

Чтобы не подключать к схеме ПК, для вывода значений счётчика можно использовать 7-сегментный дисплей. Процесс подсчёта выполняется с использованием диаграммы состояний, вывод производится на дисплей. Внимание: Код очень длинный, поэтому опущена его общая часть, полную версию можно посмотреть в учебнике.

Использование переменной «oldcounter» проходит по следующему сценарию: предыдущее значение переменной сравнивается с её новым значением «counter», и только при их разнице происходит обновление значения на 7-сегментном дисплее. Это экономит ресурсы процессора,

поскольку опрос вращающегося энкодера занимает некоторое время и одновременное выполнение двух процессов может замедлить выполнение программы и даже привести к пропуску считываний.

Нажатием кнопки можно также сбросить показания счётчика.

Что должно происходить: на 7-сегментном дисплее отображается значение счётчика. Оно уменьшается или увеличивается в зависимости от направления поворота ручки энкодера. В отличие от окна терминала, где можно фиксировать как отрицательные, так и положительные значения, на 7-сегментном дисплее отображаются только натуральные числа.

Возможный вид решения:

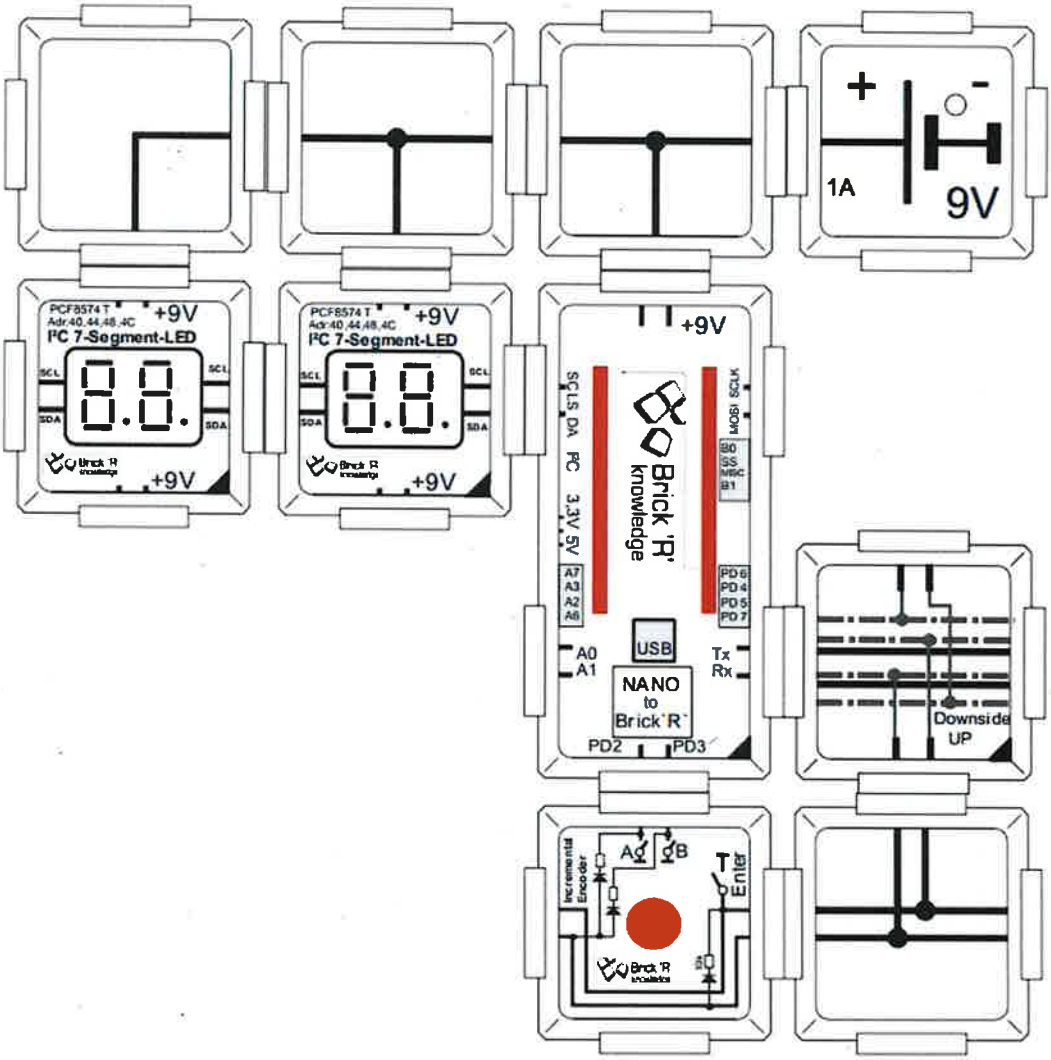

Рисунок 3.5.1 – Схема подключения

Часть кода:

```
// RU_26_SevenSegment_Incremental
// (Поворотный энкодер и 7-сегментный дисплей
для вывода значений)
#include <Wire.h>

... как в схеме с 7-сегментным дисплеем ...

// новый код:
#define SWITCHA 2 // контакт энкодера A
#define SWITCHB 3 // контакт энкодера B
#define SWITCHT 4 // контакт кнопки энкодера
#define PULLUP 5 // только для подтяжки

void setup() {
  Wire.begin(); // инициализация I2C
  pinMode(SWITCHA,INPUT_PULLUP);
  pinMode(SWITCHB,INPUT_PULLUP);
  pinMode(SWITCHT,INPUT_PULLUP);
}

void loop() { // начало цикла
  char buffer[10]; // буфер для значения счётчика в
  // строку ASCII
  static int counter = 0; // счётчик
  static int oldcounter = -999; // предыдущее значение
  int a= digitalRead(SWITCHA); // считать A
  int b= digitalRead(SWITCHB); //считать B
  int state = 0; // текущее состояние
  static int oldstate = 99; // предыдущее состояние
  int sw = digitalRead(SWITCHT);
  // считывание контакта
  // установка состояния энкодера
  if (sw == LOW) counter = 0; // сброс
  if (a==HIGH) { // контакт A
    state = 1;
  } else {
    state = 0;
  }
  if (b==HIGH) { // контакт B
    state += 2;
  } else {
    state += 0;
  }
  switch(state) {
    case 0: // AB=00
     if (oldstate == 1) {
      counter--;
    } else if (oldstate == 2) {
    counter++;
}
    break;
    case 1: // AB = 10
    if (oldstate == 0) {
    counter++;
  } else if (oldstate == 3) {
    counter--;
  }
  break;
    case 2: // AB = 01
     if (oldstate == 3) {
      counter++;
  } else if (oldstate == 0) {
    counter--;
  }
  break;
    case 3: // AB = 11
     if (oldstate == 2) {
      counter--;
    } else if (oldstate == 1) {
      counter++;
    }
    break;
  }
  oldstate = state; // сохранение предыдущего состояния
  // отображение счётчика в виде четырёх цифр из буферов 0–3
  sprintf(buffer,"%04d",counter); // преобразование значения
  // счётчика в ASCII и запись в буфер
  if (oldcounter != counter) { // только если значение изменилось
    display_seg1x(i2cseg7x2bmsb1,buffer[0]); //
    display_seg1x(i2cseg7x2blsb1,buffer[1]); //
    display_seg1x(i2cseg7x2amsb1,buffer[2]); //
    display_seg1x(i2cseg7x2alsb1,buffer[3]); //
    oldcounter = counter;
  }
}
```

Рисунок 3.5.2 – Программный код

Тема 7. OLED-дисплей.

Соберите схему OLED-дисплея с АЦП, которая будет работать как вольтметр.

В рассмотренных ранее примерах для отображения измеренного напряжения использовался 7-сегментный дисплей, но модуль OLED гораздо удобнее в использовании. Значения на нём отображаются в милливольтах (мВ).

Для этого используется оцифрованное значение с канала A0 в диапазоне 0–1023. Оцифрованные значения необходимо подстроить, чтобы получить значения напряжения в диапазоне от 0 до 5000 мВ, с помощью формулы: double mverg = (double)poti * 5000.0 / 1023.0;

Для обеспечения большей точности можно изменить значение «5000.0». Для этого нужно измерить напряжение источника питания внешним вольтметром и рассчитать поправочный коэффициент.

Что должно происходить: измеренное напряжение отображается на дисплее модуля OLED в мВ. При вращении ручки потенциометра оно может изменяться в пределах от 0 мВ до 5000 мВ, но значение не должно быть больше половины напряжения источника питания (в данном случае эта величина равна 4500 мВ).

Возможный вид решения:

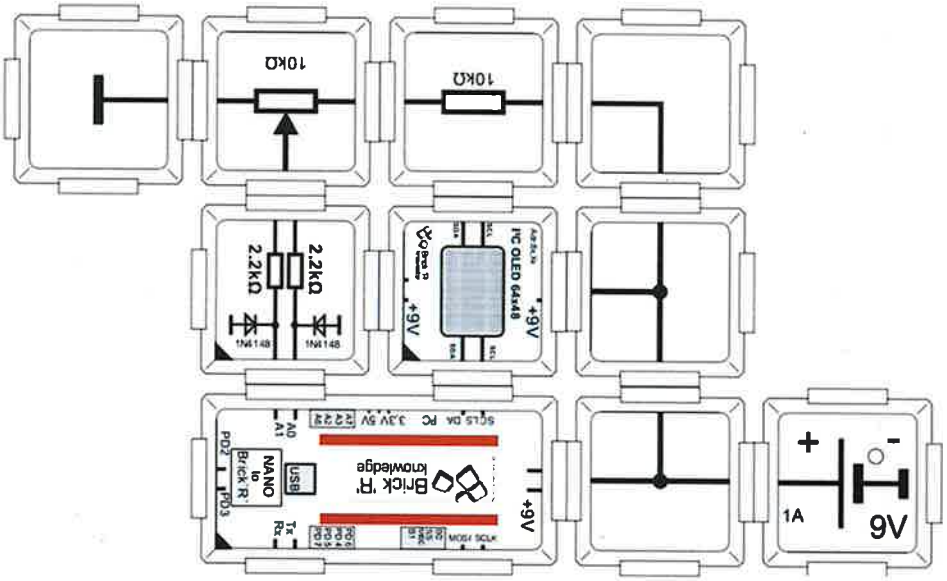

Рисунок 3.6.1 – Схема подключения

Программный код:

```
// RU_29_OLED_with_AD_converter_AD_Voltmeter (OLED-дисплей с АЦП как вольтметр)

#include <Wire.h>
#include <avr/pgmspace.h>

// При необходимости смените адрес на 78 или 7A согласно положению переключателей
#define i2coledssd (0x7A>>1) // 7A – адрес по умолчанию

// ----------------------OLED ----------------------------------------------

..... возьмите код библиотеки из Приложения

// ----------------------END OLED ----------------------------------------

void setup() {
  Wire.begin(); // инициализация I2C
  i2c_oled_initall(i2coledssd); // инициализация OLED
}

void loop() { // главный цикл
  // OLED-дисплей 64x48 пикселей
  int poti = analogRead(A0); // опрос АЦП
  char buffer[40]; // двойной буфер
  disp_buffer_clear(COLOR_BLACK); // очистка виртуального буфера
  double mverg = (double)poti * 5000.0 / 1023.0; // подстройка значений напряжения
  int mvint = (int) mverg; // опциональная подстройка
  sprintf(buffer,"%4d mV",mvint); // отображение значения в mV (мВ)
  disp_print_xy_lcd(10, 20, (unsigned char *)buffer, COLOR_WHITE, 0);
  disp_lcd_frombuffer(); // вывод значения
  delay(10); // для безопасности работы с быстрыми процессорами
}
```

Рисунок 3.6.2 – Программный код

Тема 8. Цифро-аналоговый преобразователь

Соберите схему модуля ЦАП и потенциометра.

В данном примере АЦП используется для считывания значения напряжения с потенциометра и для дальнейшей его передачи на ЦАП. Теперь АЦП блока Nano имеет диапазон значений 0–1023 (это 10 бит), а ЦАП имеет диапазон значений 0–4095 (это 12 бит). Следовательно, в программе необходимо преобразовать диапазон значений. Для этого необходимо умножить значения АЦП на 4. Для этого используется оператор сдвига «<<2». Но значение 4095 не достигается. Предел составляет 4092, что вполне нормально в данном случае.

При вращении потенциометра меняется яркость светодиодов. Светодиоды разных цветов начинают светиться при разных значениях напряжения из-за различных порогов напряжения (например, при постепенном росте напряжения жёлтый светодиод загорается раньше зелёного).

Что должно происходить: яркость обоих светодиодов регулируется положением потенциометра. Также можно наблюдать небольшую «мёртвую зону», в которой светодиод не светится, пока не достигнуто пороговое напряжение.

Возможный вид решения:

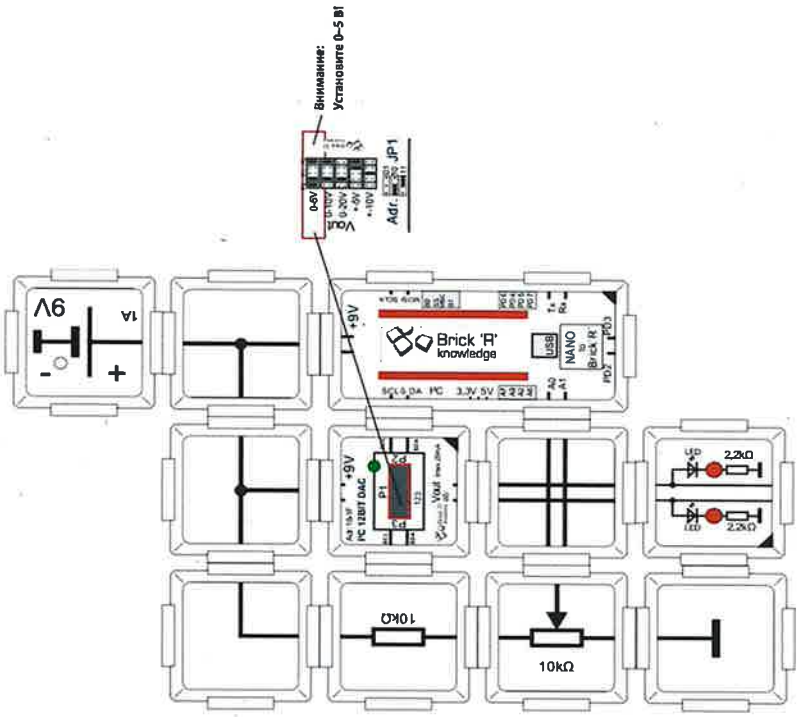

Рисунок 3.7.1 – Схема подключения

Программный код:

```
// RU_34_DA_example_DA_BrickI2C_with_potentiometer (Модуль ЦАП и потенциометр)

#include <Wire.h> // библиотека I2C
#include <avr/pgmspace.h> // дополнение

// 1E = Земля, 1c = Открыт, 1a = VCC  AD5622  открытые перемычки = 1Ch
// 0001 1aa0
// доступные адреса aa = 01 10 11

#define i2cdasel1 (0x1a>>1) // настройка адресов 1a, 1c, 1e
#define i2cdasel2 (0x1c>>1) //
#define i2cdasel3 (0x1e>>1) //

void i2c_da_write_command(unsigned char i2cbaseadr, unsigned short cmdvalue)
{
  // Bit15, 14 = 0 13, 12 = pd (std=0), затем 11–0 = значение ЦАП
  cmdvalue = cmdvalue & 0xFFF; // 12 бит, диапазон 0–4095
  Wire.beginTransmission(i2cbaseadr); // пуск I2C
  Wire.write((cmdvalue>>8)&0xff); // сначала наиболее значимый бит
  Wire.write(cmdvalue&0xff); // затем наименее значимый бит
  Wire.endTransmission(); // остановка I2C
}

void setup() {
  Wire.begin(); // инициализация I2C
}

void loop() {
  int daval=0; // временная переменная
  int potI = analogRead(A0); // также возможны a1, a2, a3
  daval = potI << 2; // настройка диапазона 0–1023 -> 0–4095
  // используются все возможные адреса I2C
  i2c_da_write_command(i2cdasel1,daval); //
  i2c_da_write_command(i2cdasel2,daval); //
  i2c_da_write_command(i2cdasel3,daval); //
}
```

Рисунок 3.7.2 – Программный код

Раздел 2. «Интернет вещей»

Тема 9. Основа работы IoT.

Соберите схему из элементов конструктора и подготовьте презентацию созданного продукта, в которой расскажите об ограничениях при подключении и возможностях контроля через IoT.

Тема 10. Установка библиотек

Создайте схему, в которой включение светодиода происходит по нажатию кнопки. В дополнение к кнопке сброса блок IoT содержит кнопку, связанную с GPIO0. Таким образом, можно использовать GPIO0 в качестве входа и считывать положение кнопки. Во избежание наличия неопределённого уровня на входе при ненажатой кнопке, предусмотрен подтягивающий резистор, поддерживающий уровень высоким (см. раздел 5.1.3 на стр. 14в учебнике). При нажатии кнопки уровень на входе становится равен 0В. Тем самым, уровень на входе всегда определён.

Не удивляйтесь тому, что светодиод в левой нижней части блока IoT загорается при нажатии кнопки. Этот светодиод связан с кнопкой и эта связь не зависит от написанного кода.

Пока кнопка не нажата, жёлтый светодиод на GPIO13 будет гореть. После нажатия кнопки, загорится красный светодиод на GPIO14.

Команда delay(100) в конце цикла отвечает за задержку длительностью в 100 миллисекунд. В это время микроконтроллер "отдыхает". Теперь – Ваш ход! Измените код для того, чтобы разобраться в нём. Например, заставьте светодиоды загораться последовательно при нажатии кнопки.

Внимание: запрещается использовать внешнюю кнопку, прямо подключающуюся к 9В питания, поскольку GPIO работают только при 5В. Рекомендуется применять исключительно кнопку GPIO0.

При напряжении более 5В на GPIO, блок IoT может быть необратимо повреждён!

Программный код:

Программа

```
void setup() {
  pinMode(0, INPUT);    // Контакт 0 - вход

  pinMode(14, OUTPUT);   // Контакт 14 - выход
  pinMode(13, OUTPUT);   // Контакт 13 - выход
}

void loop() {
  if (digitalRead(0)==LOW)
  {              // при нажатии кнопки

    digitalWrite(14,HIGH); // Светодиод на 14 контакте вкл
    digitalWrite(13,LOW); // Светодиод на 13 контакте выкл
  }
  else{         // если кнопка не нажата
    digitalWrite (14,LOW);// Светодиод на 14 контакте выкл
    digitalWrite(13,HIGH); // Светодиод на 13 контакте вкл
  }

  delay(100);                    //  ждать  100  мсек
}
```

Рисунок 3.8.1 – Программный код

Возможный вид схемы:

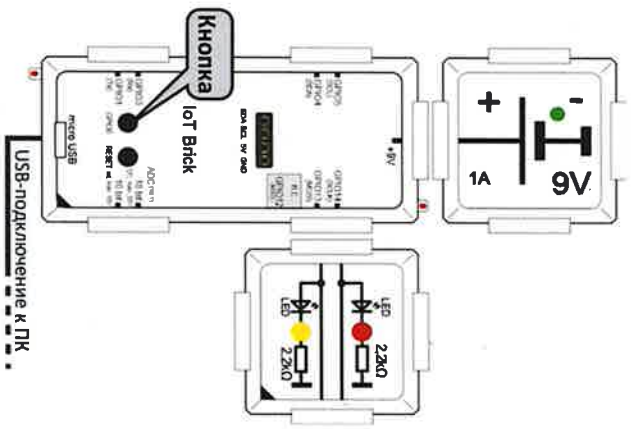

Рисунок 3.8.2 – Схема подключения

Тема 11. Схемы IoT

Откройте в Arduino IDE скетч: Example_6.6.2.ino. Необходимо также подключить следующие файлы заголовков: ESP8266WiFi.h, SSD1306Wire.h, TimeLib.h и NtpClientLib.h. Если необходимые библиотеки не были установлены, см. учебник Раздел 5.2.1 на стр. 16 и следуйте указанным инструкциям.

В этом задании сначала необходимо ввести имя сети WiFi (SSID) и пароль от сети в соответствующие поля программы. Инструкции по заполнению этих параметров приведены в разделе 6.6.1.

Т.н. «сервера времени» в интернете предоставляют данные о текущем времени и дате по протоколу NTP (NetworkTime-Protocol, сетевой протокол времени). С помощью домена pool.ntp.org любой пользователь может получить доступ к пулу серверов времени. Для получения данных в программе используются команды NtpClientLib.h, взятые из библиотек для данного примера. Полученные данные выводятся на OLED-дисплей.

При установке соединения по WiFi, команда NTP.begin("pool.ntp.org", 1, true); выполняет подключение к NTP серверу. Это может занять несколько секунд. Первый параметр назначает URL-адрес пула сервера, второй привязывает часовой пояс согласно “Всемирному координированному времени” (UTC) +1 час и значение ”true” третьего параметра говорит о переходе на летнее и зимнее время. Команда NTP.setInterval() задает временной интервал между обновлениями данных, syncEvent Triggered = true сообщает об успешном подключении к серверу NTP.

В цикле void loop() команды NTP.getTimeStr() и NTP.get DateStr() используются для представления значений даты и времени в виде строковой переменной и для подготовки их к выводу. Вывод на дисплей производится командой display.display();. Команда display.clear(); важна в этом цикле,

поскольку она очищает показания дисплея, не вызывая таким образом замещения при каждом проходе цикла.

Одновременно с дисплеем, данные сети выводятся на последовательный монитор в Arduino IDE. Для открытия последовательного монитора, нажмите на значок лупы (см. раздел 5.2.3 на стр. 20). Помимо информации о дате, времени и зимнем/летнем режимах, на последовательном мониторе также отображаются время, прошедшее с момента с первичной синхронизации (uptime) и время самой первичной синхронизации.

Программа

```
…
void setup() {
…
//При наличии доступа к Wi-Fi, подключиться к серверу времени:
{
   NTP.begin("pool.ntp.org", 1, true);  //подключение к серверу  времени NTP
NTP.setInterval(63);          //Синхронизация каждые 63 секунды
}

NTP.onNTPSyncEvent([](NTPSyncEvent_t event) {
ntpEvent = event;
syncEventTriggered = true;     //подключен к серверу времени
});
…
} void

loop() {

…
   String time = NTP.getTimeStr();  //сохранение времени в переменную
   String date = NTP.getDateStr();  //сохранение даты в переменную
   display.clear();
   display.setTextAlignment(TEXT_ALIGN_LEFT);
   display.setFont(ArialMT_Plain_24);
   display.drawString(15, 5, time);  //подготовка вывода времени
   display.drawString(5, 35, date);  //подготовка вывода даты
   display.display();  //обновление значений на дисплее
…
}
```

Рисунок 3.9.1 – Программный код

Возможный вид подключения:

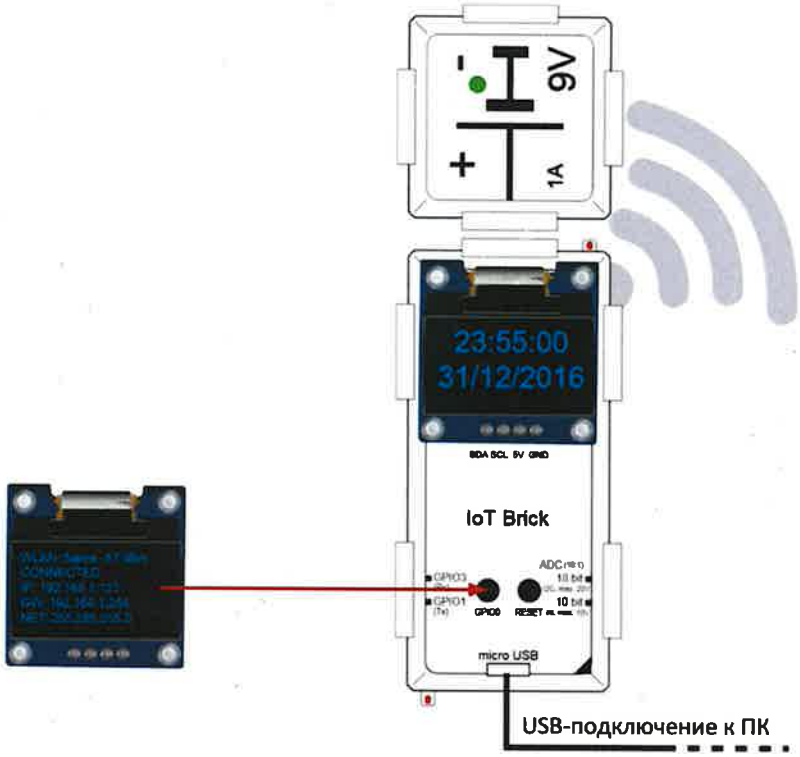

Рисунок 3.9.2 – Схема подключения

Примечание: Статус соединения блока IoT может быть отображён нажатием кнопки GPIO0.