МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)

*Е.В. ГЛУШАК, А.В. КУПРИЯНОВ*

# ВВЕДЕНИЕ В ИНТЕРНЕТ ВЕЩЕЙ
# (лабораторные работы)

Рекомендовано редакционно-издательским советом федерального государственного автономного образовательного учреждения высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева» в качестве практикума для обучающихся по основной образовательной программе высшего образования по направлению подготовки 11.03.03 Конструирование и технология электронных средств

САМАРА
Издательство Самарского университета
2023

УДК 004.7(075)+004.9(075)
ББК 381я7
Г555

Рецензенты: д-р техн. наук, проф. Н.Н. Васин,
канд. техн. наук, доц. С.В. Пальмов

***Глушак, Елена Владимировна***
Г555 **Введение в Интернет вещей (лабораторные работы):** практикум / *Е.В. Глушак, А.В. Куприянов.* – Самара: Издательство Самарского университета, 2023. – 124 с.: ил.

**ISBN 978-5-7883-2021-2**

Практикум содержит обучающий материал для выполнения лабораторных работ по курсу «Введение в Интернет вещей». Каждая работа сопровождается пояснениями и контрольными вопросами, направленными на повышение качества усвоения материала.

Дисциплина входит в раздел технических дисциплин по направлению 2.2.15 Системы, сети и устройства телекоммуникаций, рекомендована для научной специальности 2.3.8 Информатика и информационные процессы, а также предназначена для обучающихся по основной образовательной программе высшего образования по направлению подготовки 11.03.03 Конструирование и технология электронных средств.

Подготовлено на кафедре технической кибернетики.

УДК 004.7(075)+004.9(075)
ББК 381я7

ISBN 978-5-7883-2021-2 

# ОГЛАВЛЕНИЕ

# ВВЕДЕНИЕ

Концепция Интернета вещей (IoT) играет определяющую роль в дальнейшем развитии не только инфокоммуникационной отрасли, но и других отраслей. С каждым годом применение Интернета вещей все больше и больше входит в нашу жизнь.

Целью комплекса лабораторных работ является формирование знаний о принципах работы контроллеров в решениях IoT, получение умений и навыков разработки программ управления датчиками и актуаторами для контроллеров в решениях IoT в различных программах.

Задачами комплекса лабораторных работы являются изучение принципов работы контроллеров в решениях IoT на примере работы контроллера Arduino; получение навыков программирования контроллеров в решениях IoT на примере разработки программы управления датчиком и актуатором для контроллера Arduino, а также моделирование устройств IoT в различных программах.

Практикум содержит обучающий материал для выполнения лабораторных работ по курсу «Введение в Интернет вещей». Каждая работа сопровождается пояснениями и контрольными вопросами, направленными на повышение качества усвоения материала.

Дисциплина входит в раздел технических дисциплин по направлению подготовки 2.2.15 Системы, сети и устройства телекоммуникаций, рекомендована для научной специальности 2.3.8 Информатика и информационные процессы, а также предназначена для обучающихся по основной образовательной программе высшего образования по направлению подготовки 11.03.03 Конструирование и технология электронных средств.

# Лабораторная работа № 1
# РАЗРАБОТКА ТИПОВОГО ПРОЕКТА ИНТЕРНЕТА ВЕЩЕЙ. КОНЦЕПТ-ПРОЕКТ «УМНАЯ ПЛАНЕТА»

Под Интернетом вещей понимается совокупность разнообразных приборов, датчиков, устройств, объединённых в сеть посредством любых доступных каналов связи, использующих различные протоколы взаимодействия между собой и единственный протокол доступа к глобальной сети. В роли глобальной сети для Интернет-вещей в настоящий момент используется сеть Интернет. Общим протоколом является IP [1].

Следует отметить, что Интернет вещей не исключает участие человека. IoT не полностью автоматизирует вещи, так как он ориентирован на человека и предоставляет ему возможность доступа к вещам. В IoT каждая вещь имеет свой уникальный идентификатор, которые совместно образуют континуум вещей, способных взаимодействовать друг с другом, создавая временные или постоянные сети. Так вещи могут принимать участие в процессе их перемещения, делясь сведениями о текущей геопозиции, что позволяет полностью автоматизировать процесс логистики, а имея встроенный интеллект, вещи могут менять свои свойства и адаптироваться к окружающей среде, в том числе для уменьшения энергопотребления. Они могут обнаруживать другие, так или иначе связанные с ними вещи, и налаживать с ними взаимодействие. IoT позволяет создавать комбинацию из интеллектуальных устройств, объединенных сетями связи, и людей. Совместно они могут создавать самые разнообразные системы, например, для ра-

боты в средах, неудобных или недоступных для человека (в космосе, на большой глубине, на ядерных установках, в трубопроводах и т.п.).

Интернет вещей основывается на 3-х базовых принципах:

– повсеместно распространенная коммуникационная инфраструктура;

– глобальная идентификация каждого объекта;

– возможность каждого объекта отправлять и получать данные посредством персональной сети или сети Интернет, к которой он подключен.

Выделяют четыре уровня в новой сети.

1. Уровень связан с идентификацией каждого объекта.

2. Уровень предоставляет услуги по обслуживанию потребностей потребителя (можно рассматривать как сеть собственно «вещей», частный пример – «умный дом»).

3. Уровень связан с урбанизацией городской жизни, т.е. это концепция «умного города», где вся информация, которая касается жителей этого города, стягивается в конкретный жилой квартал, в конкретный дом и соседние дома.

4. Уровень – сенсорная планета.

Иными словами, Интернет вещей можно рассматривать как сеть сетей, в которой небольшие малосвязанные сети образуют более крупные.

Наиболее важными отличиями Интернета вещей от существующего интернета людей являются:

1) фокус на вещах, а не на человеке;

2) существенно большее число подключенных объектов;

3) существенно меньшие размеры объектов и невысокие скорости передачи данных;

4) фокус на считывании, а не на коммуникациях;

5) необходимость создания новой инфраструктуры и альтернативных стандартов.

Архитектура IoT (интернет вещей) включает четыре функциональных уровня.

**1. Уровень сенсоров и сенсорных сетей**

Самый нижний уровень архитектуры IoT состоит из «умных» объектов, интегрированных с сенсорами (датчиками). Сенсоры реализуют соединение физического и виртуального (цифрового) миров, обеспечивая сбор и обработку информации в реальном масштабе времени. Миниатюризация, приведшая к сокращению физических размеров аппаратных сенсоров, позволила интегрировать их непосредственно в объекты физического мира. Существуют различные типы сенсоров для соответствующих целей, например, для измерения температуры, давления, скорости движения, местоположения и др. Сенсоры могут иметь небольшую память, давая возможность записывать некоторое количество результатов измерений. Сенсор может измерять физические параметры контролируемого объекта/явления и преобразовать их в сигнал, который может быть принят соответствующим устройством. Сенсоры классифицируются в соответствии с их назначением, например, сенсоры окружающей среды, сенсоры для тела, сенсоры для бытовой техники, сенсоры для транспортных средств и т.д.

**2. Уровень шлюзов и сетей**

Большой объем данных, создаваемых на первом уровне IoT многочисленными миниатюрными сенсорами, требует надежной и высокопроизводительной проводной или беспроводной сетевой инфраструктуры в качестве транспортной среды. Существующие сети связи, использующие различные протоколы, могут быть использованы для поддержки межмашинных коммуникаций M2M и их приложений. Для реализации широкого спектра услуг и приложений в IoT необходимо обеспечить совместную работу множества сетей различных технологий и протоколов доступа в гетерогенной конфигурации. Эти сети должны обеспечивать требуемые

значения качества передачи информации, и прежде всего по задержке, пропускной способности и безопасности. Данный уровень состоит из конвергентной сетевой инфраструктуры, которая создается путем интеграции разнородных сетей в единую сетевую платформу. Конвергентный абстрактный сетевой уровень в IoT позволяет через соответствующие шлюзы нескольким пользователям использовать ресурсы в одной сети независимо и совместно без ущерба для конфиденциальности, безопасности и производительности.

**3. Сервисный уровень**

Сервисный уровень содержит набор информационных услуг, призванных автоматизировать технологические и бизнес операции в IoT: поддержки операционной и бизнес деятельности, различной аналитической обработки информации (статистической, интеллектуального анализа данных и текстов, прогностическая аналитика и др.), хранения данных, обеспечения информационной безопасности, управления бизнес-правилами, управления бизнес-процессами и др.

**4. Уровень приложений**

На четвертом уровне архитектуры IoT существуют различные типы приложений для соответствующих промышленных секторов и сфер деятельности (энергетика, транспорт, торговля, медицина, образование и др.). Приложения могут быть «вертикальными», когда они являются специфическими для конкретной отрасли промышленности, а также «горизонтальными», (например, управление автопарком, отслеживание активов и др.), которые могут использоваться в различных секторах экономики.

На основе Интернета вещей могут быть реализованы всевозможные «умные» приложения в различных сферах деятельности и жизни человека:

1. «Умная планета» – человек сможет буквально «держать руку на пульсе» планеты: своевременно реагировать на упущения в

планировании хозяйств, загрязнения и другие экологические проблемы, а значит, эффективно распоряжаться невозобновляемыми ресурсами.

2. «Умный город» – городская инфраструктура и сопутствующие муниципальные услуги, такие как образование, здравоохранение, общественная безопасность, ЖКХ, станут более связанными и эффективными.

3. «Умный дом» – система будет распознавать конкретные ситуации, происходящие в доме, и реагировать на них соответствующим образом, что обеспечит жильцам безопасность, комфорт и ресурсосбережение.

4. «Умная энергетика» – будет обеспечена надежная и качественная передача электрической энергии от источника к приемнику в нужное время и в необходимом количестве.

5. «Умный транспорт» – перемещение пассажиров из одной точки пространства в другую станет удобнее, быстрее и безопаснее.

6. «Умная медицина» – врачи и пациенты смогут получить удаленный доступ к дорогостоящему медицинскому оборудованию или к электронной истории болезни в любом месте, будет реализована система удаленного мониторинга здоровья, автоматизирована выдача лекарственных препаратов больным и другое.

### *Задания для выполнения лабораторной работы*

**Необходимо:** разработать концепт-проект.

Отчет по лабораторной работе должен содержать решение следующих задач:

1. Описать цели и задачи концепт-проекта.

2. Дать анализ (достоинства и недостатки) имеющихся зарубежных прототипов (по результатам поиска в Интернет и литературе).

3. Дать анализ (достоинства и недостатки) имеющихся отечественных прототипов (по результатам поиска в Интернет и литературе).

4. Разработать структурную схему концепт-проекта с указанием основных функциональных блоков.

5. Выбрать необходимые датчики/сенсоры/актуаторы для реализации концепт-проекта.

6. Выбрать необходимые аппаратно-программные комплексы для реализации концепт-проекта.

7. Разработать схему связи функциональных блоков концепт-проекта.

8. Выбрать необходимые технологии и средства проводной/беспроводной связи для реализации концепт-проекта.

9. Выбрать необходимые информационные технологии Интернет (порталы, сайты, сервисы и т.п.) для реализации концепт-проекта.

10. Описать алгоритм функционирования концепт-проекта.

11. Произвести подбор уже выпускающихся компонентов для реализации концепт-проекта.

12. Оценить приблизительную стоимость реализации концепт-проекта для выбранного масштаба реализации.

13. Сделать выводы о практической возможности реализации концепт-проекта (оценить основные преимущества и возможные трудности).

Вариант соответствует номеру студента в списке группы.

### *Варианты заданий*

1. «Умный» аэропорт.
2. Контроль цунами.
3. Контроль погоды.
4. Контроль ледников.

5. Контроль лесов.

6. Контроль сельскохозяйственных полей.

7. Контроль сельскохозяйственных животных.

8. Контроль диких животных.

9. Контроль водохранилищ, рек и озер.

10. Контроль вулканов.

11. Контроль океанов и морей.

12. Контроль воздушного пространства.

13. Контроль космоса.

14. Контроль атмосферы.

15. Контроль пустыней.

16. Контроль экологических загрязнений.

17. Контроль государственной границы.

18. Контроль международного автомобильного транспортного сообщения.

19. Контроль международного почтового обмена.

20. Международный контрольно-пропускной пункт для транспорта.

21. Контроль землетрясений.

22. «Умный» железнодорожный вокзал.

23. «Умная» армия.

24. «Умный» завод.

25. «Умная» полиция.

26. «Умное» здравоохранение.

### *Требования к оформлению отчета*

1. Отчет выполняется в печатном виде на листах формата А4.

2. Отчет должен содержать все необходимые компоненты (введение, пояснительная записка, заключение, список используемой литературы, список сокращений).

3. Все разрабатываемые схемы должны быть выполнены с использованием графического пакета MSVisio.

### *Вопросы для самоконтроля*

1. Что такое Интернет вещей?
2. Что такое сенсорная сеть?
3. Что такое самоорганизующаяся сеть связи?
4. Что такое беспроводная сенсорная сеть?
5. Какие интерфейсы входят в состав архитектуры беспроводных сенсорных сетей?
6. Перечислите примеры практического применения беспроводных сенсорных сетей.
7. Что такое интернет нано-вещей?
8. Что является основными компонентами архитектуры сети Интернета нано-вещей?
9. Назовите основные коммуникационные проблемы практической реализации нано-сетей.
10. Что такое когнитивный интернет вещей?
11. Перечислите функции нано-сенсоров и нано-актуаторов.
12. Приведите примеры практического применения когнитивного Интернета вещей и сетей на теле человека.
13. Приведите примеры практического применения интернета нано-вещей.

# Лабораторная работа № 2
# РАЗРАБОТКА ТИПОВОГО ПРОЕКТА ИНТЕРНЕТА ВЕЩЕЙ. КОНЦЕПТ-ПРОЕКТ «УМНЫЙ ГОРОД»

В последние годы в городах интенсивно создаются информационные системы для автоматизации отдельных сфер городской жизни: безопасности городской среды, транспорта, энергетики и ЖКХ, здравоохранения, образования, государственного и муниципального управления и др. Принципы и технологии IoT позволяют создать полносвязное интегрированное решение, необходимое для функционирования городской среды (рис. 1) и доступное всем жителям города, сотрудникам городских служб, чиновникам и управленцам разных уровней [1].


Рис. 1. Основные подсистемы «умного города»

Следует признать, что Интернет вещей пока еще не проник глубоко в элементы городской инфраструктуры и хозяйства, но уже сформировал сферу влияния, в рамках которой играет практически революционную роль. Это в первую очередь транспорт, энергетика и коммунальные услуги, экология, контроль преступности, информационное обеспечение жителей города и интерактивное управление домохозяйством.

Другой интересный пример – умные мусорные контейнеры [1]. Сигнал о наполнении подается в централизованную систему управления, которая отслеживает на карте все мусороуборочные машины и включает наполненный контейнер в маршрут ближайшего грузовика. И это тоже уже не фантастика: именно так работает мусоросборочная система в Дублине и Барселоне.

Идея использовать в Интернете вещей такую простую, получившую повсеместное распространение технологию, как сотовая связь, находит все большее применение во всем мире.

В будущем смартфоны горожан сформируют постоянно расширяющуюся сеть муниципальных датчиков. Сейчас ученые экспериментируют со встраиванием датчиков в сотовые телефоны для решения социальных проблем (например, сбора данных по загрязнению воздуха или уровню радиации) так, чтобы свести к минимуму или даже нулю необходимость в помощи со стороны горожан.

### *Задания для выполнения лабораторной работы*

**Необходимо:** разработать концепт-проект.

Отчет по лабораторной работе должен содержать решение следующих задач:

1. Описать цели и задачи концепт-проекта.

2. Дать анализ (достоинства и недостатки) имеющихся зарубежных прототипов (по результатам поиска в Интернет и литературе).

3. Дать анализ (достоинства и недостатки) имеющихся отечественных прототипов (по результатам поиска в Интернет и литературе).

4. Разработать структурную схему концепт-проекта с указанием основных функциональных блоков.

5. Выбрать необходимые датчики/сенсоры/актуаторы для реализации концепт-проекта.

6. Выбрать необходимые аппаратно-программные комплексы для реализации концепт-проекта.

7. Разработать схему связи функциональных блоков концепт-проекта.

8. Выбрать необходимые технологии и средства проводной/беспроводной связи для реализации концепт-проекта.

9. Выбрать необходимые информационные технологии Интернет (порталы, сайты, сервисы и т.п.) для реализации концепт-проекта.

10. Описать алгоритм функционирования концепт-проекта.

11. Произвести подбор уже выпускающихся компонентов для реализации концепт-проекта.

12. Оценить приблизительную стоимость реализации концепт-проекта для выбранного масштаба реализации.

13. Сделать выводы о практической возможности реализации концепт-проекта (оценить основные преимущества и возможные трудности).

Вариант соответствует номеру студента в списке группы.

### *Варианты заданий*

1. «Умный» городской транспорт.

2. «Умный» перекресток.

3. «Умное» водоснабжение.

4. «Умное» электроснабжение.
5. «Умное» газоснабжение.
6. «Умное» освещение улиц и домов.
7. «Умная» аптека.
8. «Умный» магазин.
9. «Умная» химчистка.
10. «Умная» библиотека.
11. «Умная» поликлиника.
12. «Умная» больница.
13. «Умная» школа.
14. «Умный» детсад.
15. «Умная» парикмахерская.
16. «Умный» лифт.
17. «Умный» подъезд.
18. «Умная» остановка транспорта.
19. «Умная» мусорная урна.
20. «Умный» музей.
21. «Умный» стадион.
22. «Умный» эскалатор.
23. «Умная» парковка.
24. «Умный» университет.
25. «Умный» пляж.

### *Требования к оформлению отчета*

1. Отчет выполняется в печатном виде на листах формата А4.

2. Отчет должен содержать все необходимые компоненты (введение, пояснительная записка, заключение, список используемой литературы, список сокращений).

3. Все разрабатываемые схемы должны быть выполнены с использованием графического пакета MSVisio.

### *Вопросы для самоконтроля*

1. Что такое Интернет вещей?

2. Перечислите примеры практического применения беспроводных сенсорных сетей и Интернета вещей?

3. На каких принципах основывается Интернет вещей?

4. Какие уровни включает в себя архитектура Интернета вещей?

5. Перечислите функции уровней архитектуры Интернета вещей.

6. Перечислите примеры практического применения Интернета вещей.

7. Что входит в понятие Интернета вещей?

8. Когда возник Интернет вещей и почему?

9. Укажите базовые принципы IoT.

10. Как соотносятся физические и виртуальные вещи?

11. Кто занимается стандартизацией Интернета вещей?

12. Поясните назначение функциональных уровней базовой архитектуры Интернетавещей.

13. Что общего и чем отличаются Интернет вещей Веб вещей?

14. Из чего состоит интернет нано вещей?

15. Поясните основные способы взаимодействия с интернет-вещами.

16. Какова зрелость концепции IoT и ее базовых составляющих?

17. Укажите основные характеристики подхода «большие данные».

# ЛАБОРАТОРНАЯ РАБОТА № 3
# РАЗРАБОТКА ТИПОВОГО ПРОЕКТА ИНТЕРНЕТА ВЕЩЕЙ. КОНЦЕПТ-ПРОЕКТ «УМНЫЙ ДОМ»

«Умный дом» предназначен для максимально комфортной жизни людей посредством использования современных высокотехнологичных средств.

Принцип работы системы «умный дом» заключается в автоматизации всего, из чего состоит жилая постройка: освещение, кондиционирование, система безопасности, электроэнергия, отопление, водоснабжение и водоотведение и так далее [1]. К основным подсистемам «умного дома» относятся: климат-контроль, освещение, мультимедиа (аудио и видео), охранные системы, связь и другие (рис. 2).

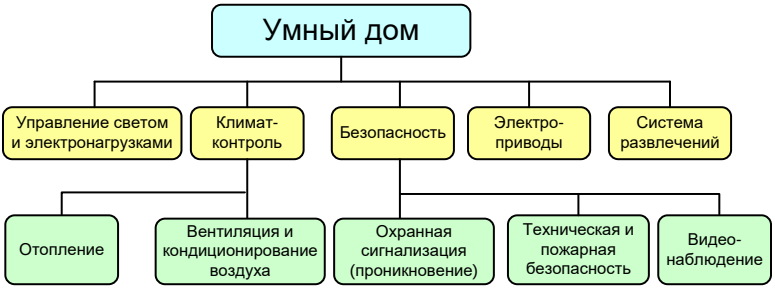


Рис. 2. Основные подсистемы «умного дома»

В стандартном проекте «умного дома» можно выделить три основные подсети: сеть мультимедийных устройств, сеть электро-

осветительного оборудования и сенсорную сеть. В последнем случае это датчики движения, света, температуры, давления, влажности, вибрации и т.п. Таким образом, «умный дом» состоит из программного и аппаратного обеспечения, датчиков и проводной/беспроводной сети (рис. 3).


Рис. 3. Основные компоненты «умного дома»

В общем случае, «умный дом» предоставляет его владельцу следующие преимущества [1]:

1) снижение потребления ресурсов (газ, вода, электроэнергия);
2) высокий уровень комфорта;
3) обеспечение необходимого взаимодействия всех автоматизируемых систем объекта недвижимости, задание различных режимов работы;
4) снижение вероятности возникновения аварийных ситуаций;
5) повышение оперативности, простоты и удобства управления.

Для автоматизации дома смарт-узлы могут быть интегрированы непосредственно в бытовые приборы, например в пылесосы,

микроволновые печи, холодильники и телевизоры. Они могут взаимодействовать друг с другом и с внешней сетью через интернет. Это позволит конечным пользователям легко управлять устройствами дома как локально, так и удаленно.

### *Задания для выполнения лабораторной работы*

**Необходимо:** разработать концепт-проект.

Отчет по лабораторной работе должен содержать решение следующих задач:

1. Описать цели и задачи концепт-проекта.

2. Дать анализ (достоинства и недостатки) имеющихся зарубежных прототипов (по результатам поиска в Интернет и литературе).

3. Дать анализ (достоинства и недостатки) имеющихся отечественных прототипов (по результатам поиска в Интернет и литературе).

4. Разработать структурную схему концепт-проекта с указанием основных функциональных блоков.

5. Выбрать необходимые датчики/сенсоры/актуаторы для реализации концепт-проекта.

6. Выбрать необходимые аппаратно-программные комплексы для реализации концепт-проекта.

7. Разработать схему связи функциональных блоков концепт-проекта.

8. Выбрать необходимые технологии и средства проводной/беспроводной связи для реализации концепт-проекта.

9. Выбрать необходимые информационные технологии Интернет (порталы, сайты, сервисы и т.п.) для реализации концепт-проекта.

10. Описать алгоритм функционирования концепт-проекта.

11. Произвести подбор уже выпускающихся компонентов для реализации концепт-проекта.

12. Оценить приблизительную стоимость реализации концепт-проекта для выбранного масштаба реализации.

13. Сделать выводы о практической возможности реализации концепт-проекта (оценить основные преимущества и возможные трудности).

Вариант соответствует номеру студента в списке группы.

### *Варианты заданий*

1. «Умный» холодильник.
2. «Умный» пылесос.
3. «Умный» тостер.
4. «Умная» микроволновая печь.
5. «Умное» освещение в квартире.
6. «Умное» отопление в квартире.
7. «Умное» кондиционирование в квартире.
8. «Умный» телевизор.
9. «Умный» туалет.
10. «Умная» ванная комната.
11. «Умная» кухня.
12. «Умный» домашний кинотеатр.
13. «Умный» шкаф для одежды/обуви.
14. «Умная» кровать.
15. «Умная» детская комната.
16. «Умный» кабинет.
17. «Умная» спальня.
18. «Умное» кресло/стул.
19. «Умный» стол.
20. «Умная» посудомоечная машина.
21. «Умный» гараж.

22. «Умная» дача.

23. «Умный» аквариум.

24. «Умный» горшок для домашних растений.

25. «Умная» кормушка для домашних животных.

### *Требования к оформлению отчета*

1. Отчет выполняется в печатном виде на листах формата А4.

2. Отчет должен содержать все необходимые компоненты (введение, пояснительная записка, заключение, список используемой литературы, список сокращений).

3. Все разрабатываемые схемы должны быть выполнены с использованием графического пакета MSVisio.

### *Вопросы для самоконтроля*

1. Перечислите основные направления практического внедрения IoT.

2. Укажите основные движущие силы и барьеры на пути внедрения Интернета вещей.

3. Как классифицируются по территории охвата телекоммуникационные сети, используемые в Интернете вещей?

4. Какие беспроводные сети малого радиуса используются в IoT?

5. Укажите особенности стандарта IEEE 802.15.4.

6. Какого назначение стандарта ZigBee?

7. Какие устройства входят в состав сети на базе стандарта ZigBee?

8. Для каких целей был разработан стандарт 6LoWPAN?

# Лабораторная работа № 4
# РАЗРАБОТКА ТИПОВОГО ПРОЕКТА ИНТЕРНЕТА ВЕЩЕЙ. КОНЦЕПТ-ПРОЕКТ «УМНАЯ ЖИЗНЬ»

Одним из самых первых примеров бытовой техники, имеющей подключение к Интернету, является обычный тостер, оснащенный интерфейсом для удаленного включения и сообщения о готовности поджаренного тоста.

Интернет-холодильник (Internet refrigerator или Smart refrigerator) – новый класс бытовых холодильников, появившийся в начале XXI века. Как правило, он имеет встроенный компьютер с постоянным подключением к сети интернет и сенсорный экран на фронтальной панели (рис. 4).

Рис. 4. Интернет-холодильник

Такой холодильник не только хранит продукты, но и даёт возможность пользоваться интернетом, через который можно получить доступ к различным сайтам (например, с кулинарными рецептами для приготовления блюд) и даже заказывать продукты в интернет-магазинах с доставкой на дом. Кроме того, с помощью интернет-холодильника можно общаться, используя электронную и видеопочту. Интернет-холодильник может предоставлять целый ряд сервисов: доступ в Интернет, видеотелефон, e-mail, TV, MP3-музыку, базу данных по кулинарным рецептам и правилам питания, электронное перо, чтобы оставить сообщение, голосовые послания. Кроме этого, интернет-холодильник отслеживает продукты с истекающим сроком годности.

Робот-пылесос может действовать автономно, программироваться и управляться через Интернет, для чего имеется ряд сенсоров и инфракрасная встроенная камера (рис. 5).

Рис. 5. Робот-пылесос VC-RL87W компании Samsung

Система управления работой пылесоса делает несколько снимков в секунду создавая, таким образом, карту всего дома или отдельных его комнат. Устройство также имеет возможность запоминать оптимальный путь уборки и определять своё местонахождение в доме. Аккумулятора хватает на определенное время уборки (обычно до 1,5 часов), по истечении которого робот сам отправляется на подзарядку. К пылесосу имеется беспроводный

доступ Wi-Fi с помощью компьютера или смартфона. Через эти устройства можно запустить его и в режиме реального времени наблюдать за тем, что происходит в комнате. Более того, можно поговорить с людьми, которые находятся в доме через систему голосовой связи. Встроенный источник света позволяет видеть в полной темноте и проверить помещение даже ночью [1].

Интернет микроволновая печь имеет встроенный модем для выхода в интернет, память для хранения скачиваемой информации и пульт управления. Она выполняет следующие задачи:

- скачивание рецептов из интернета и самопрограммирование;
- связь с компаниями – производителями продуктов;
- дает доступ к системе заказа продуктов по интернету (рис. 6).

Рис. 6. Микроволновая интернет-печь M-G270IT
компании LG Electronics

Интернет-кондиционер подключается к интернету по проводной или беспроводной сети Wi-Fi и дает пользователю доступ к управлению кондиционером из любой точки земного шара. Владелец может дистанционно включать и выключать систему, программировать настройки, выбирать режимы, температуру, скорость вентилятора, задавать параметры, словом совершать любые манипуляции, доступные с обычного пульта.

### *Задания для выполнения лабораторной работы*

**Необходимо:** разработать концепт-проект.

Отчет по лабораторной работе должен содержать решение следующих задач:

1. Описать цели и задачи концепт-проекта.

2. Дать анализ (достоинства и недостатки) имеющихся зарубежных прототипов (по результатам поиска в Интернет и литературе).

3. Дать анализ (достоинства и недостатки) имеющихся отечественных прототипов (по результатам поиска в Интернет и литературе).

4. Разработать структурную схему концепт-проекта с указанием основных функциональных блоков.

5. Выбрать необходимые датчики/сенсоры/актуаторы для реализации концепт-проекта.

6. Выбрать необходимые аппаратно-программные комплексы для реализации концепт-проекта.

7. Разработать схему связи функциональных блоков концепт-проекта.

8. Выбрать необходимые технологии и средства проводной/беспроводной связи для реализации концепт-проекта.

9. Выбрать необходимые информационные технологии Интернет (порталы, сайты, сервисы и т.п.) для реализации концепт-проекта.

10. Описать алгоритм функционирования концепт-проекта.

11. Произвести подбор уже выпускающихся компонентов для реализации концепт-проекта.

12. Оценить приблизительную стоимость реализации концепт-проекта для выбранного масштаба реализации.

13. Сделать выводы о практической возможности реализации концепт-проекта (оценить основные преимущества и возможные трудности).

Вариант соответствует номеру студента в списке группы.

### *Варианты заданий*

1. «Умные» часы.
2. «Умная» зубная щетка.
3. «Умные» кроссовки.
4. «Умные» очки.
5. «Умная» одежда.
6. «Умный» зонт.
7. «Умная» посуда.
8. «Умная» таблетка.
9. «Умный» термометр.
10. «Умная» упаковка товаров.
11. «Умные» туфли.
12. «Умный» стакан.
13. «Умная» микроволновая печь.
14. «Умная» сумка.
15. «Умное» кольцо.
16. «Умные» брюки.
17. «Умная» майка.
18. «Умная» мягкая игрушка.
19. «Умная» подушка.
20. «Умный» пульт от телевизора.
21. «Умный» шарф.
22. «Умная» шапка.
23. «Умный» лак для ногтей.

### *Требования к оформлению отчета*

1. Отчет выполняется в печатном виде на листах формата А4.

2. Отчет должен содержать все необходимые компоненты (введение, пояснительная записка, заключение, список используемой литературы, список сокращений).

3. Все разрабатываемые схемы должны быть выполнены с использованием графического пакета MSVisio.

### *Вопросы для самоконтроля*

1. На чем основывается «умная» планета?
2. Что входит в концепцию «умный» город?
3. Перечислите основные подсистемы «умного города».
4. Перечислите основные подсистемы «умного дома».
5. Перечислите преимущества и недостатки «умного дома».
6. Что входит в концепцию «умная энергия»?
7. Что входит в концепцию «Умный транспорт»?
8. Как работает система интеллектуального управления транспортом?
9. Что входит в концепцию «Умная жизнь»?
10. Что входит в концепцию «Умная носимая электроника»?
11. Что такое промышленный Интернет вещей?
12. Что такое Социальный Интернет вещей?
13. Что такое Семантический Интернет вещей?
14. Что входит в архитектуру семантического шлюза IoT?

# Лабораторная работа № 5
# МОДЕЛИРОВАНИЕ МИКРОКОНТРОЛЛЕРОВ IOT В ПРОГРАММЕ TINKERCAD

Микроконтроллер – это микрочип или плата с микрочипом для решения клиентских частей IoT-проектов. Некоторые проекты в IoT проще всего решить на микроконтроллерах. Они поддерживают множество стандартов ввода и вывода, работают с меньшим энергопотреблением и стоят дешевле по сравнению с микрокомпьютерами. Недостатком является меньшая вычислительная мощность и отсутствие операционной системы по умолчанию. Наиболее популярными микроконтроллерами являются Atmel, STM, Arduino, ESP8266, ESP32 и др.

Микрокомпьютер обычно представляет собой систему на чипе, включая классическую архитектуру фон Неймана с центральным процессором, видеокартой, оперативной памятью, модулями подключения к сетям связи и портами ввода-вывода. Современные микрокомпьютеры используют такие операционные системы, как Linux и Windows. Как правило, микрокомпьютеры имеют большую вычислительную мощность, чем микроконтроллеры, видеовыход на HDMI, высокоскоростной Wi-Fi и Bluetooth, подключение к картам флэш-памяти и M.2 и т.д. Недостатком микрокомпьютеров является более высокая цена и более высокое энергопотребление по сравнению с микроконтроллерами. Микрокомпьютеры используются в проектах IoT, если необходимо выполнять задачи высокого уровня, включая потоковое видео, сложные вычисления и т.п.

В данной лабораторной работе мы будем использовать модель контроллера Arduino Uno.

Arduino Uno – плата от компании Arduino, построенная на микроконтроллере ATmega 328. Плата имеет 6 аналоговых входов, 14 цифровых выводов общего назначения (могут являться как входами, так и выходами), кварцевый генератор на 16 МГц, два разъема: силовой и USB, разъем ISCP для внутрисхемного программирования и кнопку горячей перезагрузки устройства. Для стабильной работы плату необходимо подключить к питанию либо через встроенный USB разъем, либо к источнику от 7 до 12В. Через переходник питания плата также может работать от батареи. На рис. 7 представлены обозначения разъемов Arduino.

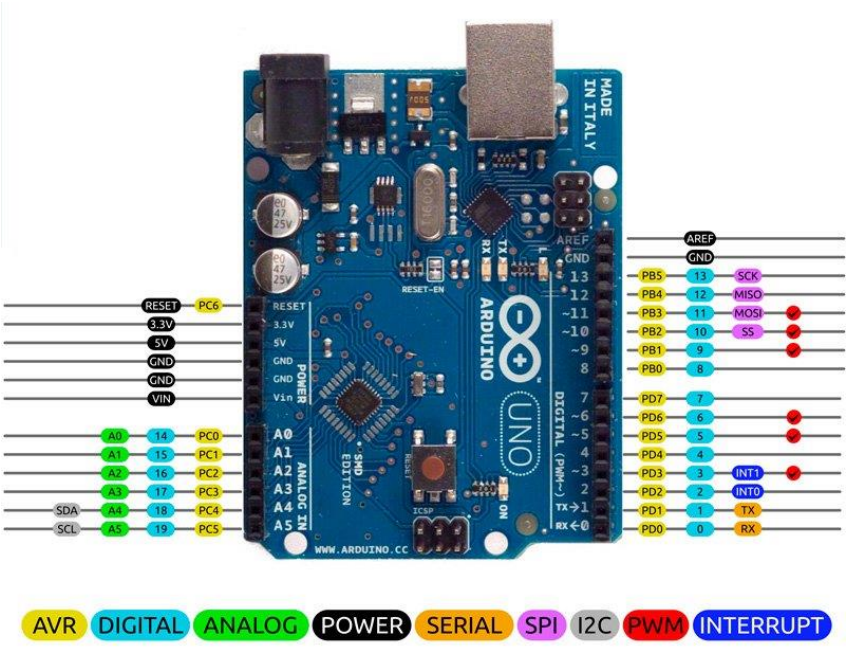


Рис. 7. Плата Arduino с обозначением разъёмов (пинов)

Плата Arduino Uno имеет 3 способа подключения питания: через USB, через внешний разъем питания и через разъем Vin. Плата имеет встроенный стабилизатор, позволяющий не только автоматически выбирать источник питания, но и выравнивать ток до стабильных 5 вольт, необходимых контроллеру для работы. Внешнее питание можно подавать как напрямую от USB порта компьютера, так и от любого AC/DC блока питания через разъем питания или USB.

На плате предусмотрено несколько выводов, позволяющих подключать к питанию датчики, сенсоры и актуаторы. Все эти выводы помечены:

Vin – вход питания, используется для получения питания от внешнего источника. Через данный вывод происходит только подача питания на плату, получить через него питание для внешних устройств невозможно. На вход Vin рекомендуется подавать напряжение в диапазоне от 7В до 20В, во избежание перегрева и сгорания встроенного стабилизатора.

5V – источник напряжения для питания внешних устройств. При получении питания платой из любых других источников (USB, разъем питания или Vin) на этом контакте вы всегда сможете получить стабильное напряжение 5 вольт. Его можно вывести на макетную плату или подать напрямую на необходимое устройство.

3V3 – источник напряжения 3.3 вольта для питания внешних устройств. Работает по такому-же принципу, что и контакт 5V. Его также можно вывести напряжение на макетную плату, либо подать на необходимый датчик/сенсор напрямую.

GND – контакт для подключения земли. Необходим для создания замкнутой цепи при подключении к контактам Vin, 5V или 3V3. Во всех случаях ножку GND необходимо выводить как минус, иначе цепь не будет замкнута и питание (что внешнее, что внутреннее) не будет подаваться.

Платформа Arduino Uno имеет оснащена микроконтроллером ATmega328, который обладает следующими типами памяти:

– FLASH 32 Кбайт, из которых 0.5 Кбайт используется для хранения загрузчика;

– SRAM (ОЗУ) 2 Кбайт;

– EEPROM – 1 Кбайт (доступна с помощью библиотеки EEPROM).

Для передачи данных с контроллера в шлюз или между контроллерами могут использоваться проводные и беспроводные технологии. Для чего к соответствующим интерфейсам платы контроллера могут быть подключены медные проводники или радио модули.

Во многих решениях IoT предпочтение отдается беспроводному подключению ввиду удобства и экономичности (во многих проектах конечные устройства могут размещаться в труднодоступных местах – подвалах, за трубами и т.п., куда сложно протянуть кабели). В настоящее время в большинство контроллеров могут быть встроены или подключаться в качестве плат расширения различные модули беспроводной связи, начиная с простейших одночастотных RF-модулей и заканчивая продвинутыми Bluetooth и Wi-Fi модулями.

Передача данных с контроллера, как правило, производится через специальные интерфейсы, такие как I2C, UART, SPI. В данной лабораторной работе мы будем использовать интерфейс I2C.

I2C – последовательная асимметричная шина для связи между интегральными схемами внутри электронных приборов. Разработана фирмой Philips Semiconductors в начале 1980-х как простая 8-битная шина внутренней связи для создания управляющей электроники. Была рассчитана на частоту 100 кГц.

Обычно программировать микроконтроллеры можно с помощью языков программирования, таких как C, C ++, Python и т.д.

В данной лабораторной работе используются модели контроллера Arduino, поэтому для разработки программ мы будет использовать специальный язык Arduino C.

Для эмуляции контроллера Arduino и симуляции его работы с внешними устройствами мы будем использовать среду моделирования Tinkercad Circuits Arduino.

Tinkercad Circuits Arduino – это бесплатный, простой и одновременно мощный онлайн-эмулятор Arduino, предоставляющий удобную среду для написания различных проектов**.**

Для начала работы необходимо получить аккаунт Autocad. Регистрация в Tinkercad бесплатная. Зайдите на сайт https://www.tinkercad.com/ [2] и выполните простые шаги регистрации. После регистрации зайдите в свою учетную запись и в меню слева выберите вкладку «Цепи» («Circuits»). После этого нажмите кнопку «Создать цепь». Вы перейдете на главный экран Tinkercad Circuits Arduino (рис. 8). Для моделирования контроллера Arduino Uno кликните по соответствующему элементу из списка в меню справа (рис. 9) и переместив курсор в рабочую область кликните по ней левой клавишей мыши.

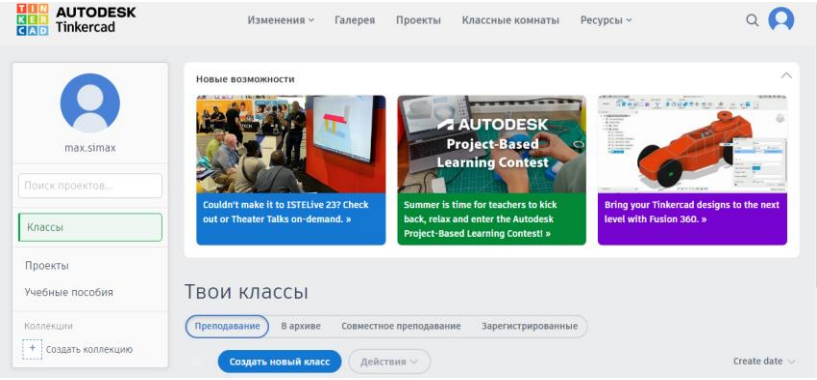


Рис. 8. Меню главного экрана Tinkercad

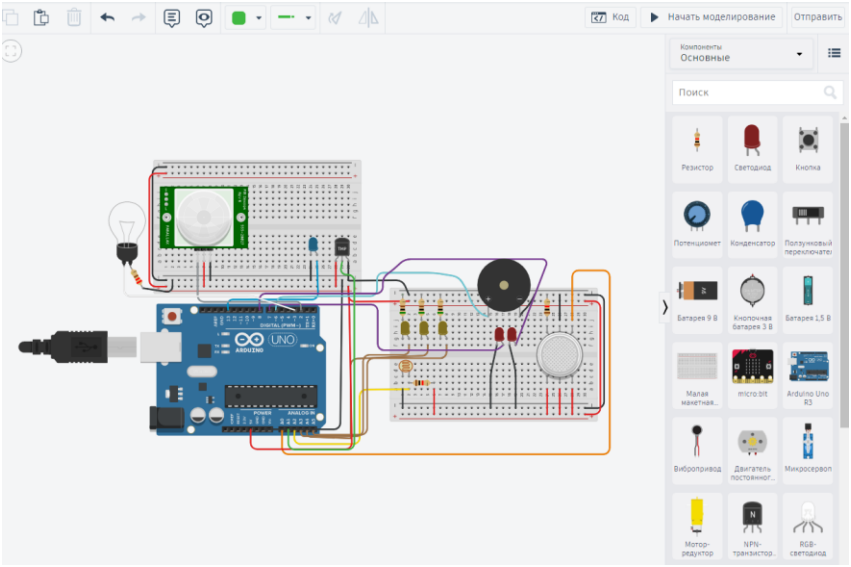


Рис. 9. Главный экран Tinkercad Circuits Arduino

Для создания цепи выберите необходимые элементы и выполните действия, аналогичные добавлению контроллера. После этого вы можете соединить элементы с помощью проводников, подкачаю проводники к ножкам электронных компонентов и контактам контроллеров либо соедините их через макетную плату.

Для написания кода кликните по кнопке «Код». Справа на главном экране откроется панель редактирования кода (рис. 10). Если в вашем проекте несколько контроллеров, то выбор контроллера для программирования осуществляется нажатием по соответствующей кнопке в меню справа.

Если в вашем коде предусмотрен вывод данных в последовательный интерфейс контроллера, то результат можно наблюдать на панели справа снизу «Монитор последовательного интерфейса». Если по умолчанию он закрыт, то кликните по соответствующей кнопке в нижней части панели справа. Монитор последова-

тельного интерфейса удобно использовать для проверки работоспособности вашего кода, добавив в код соответствующие функции. Также здесь отображаются ошибки, возникающие при компиляции кода.

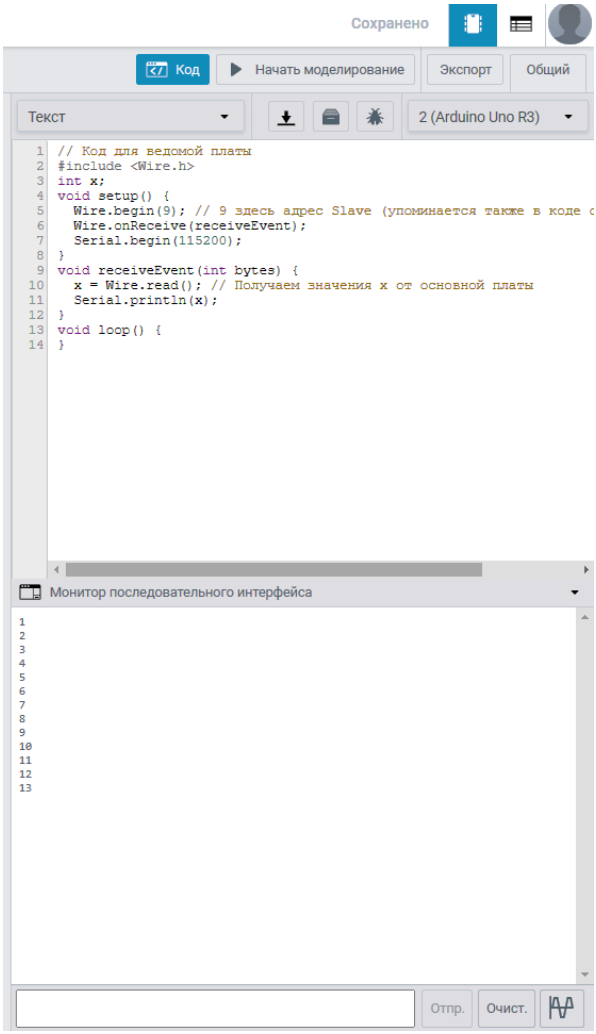


Рис. 10. Редактор кода и монитор вывода данных

Проекты в Tinkercad сохраняются автоматически. Для изменения масштаба отображения компонентов проекта наведите курсор мыши на рабочую область и крутите колесиком мыши вперед или назад для нужного масштаба. Для перемещения схемы нажмите и, удерживая левую клавишу мыши, перемещайте курсор на нужное положение в рабочей области.

**Модель счетчика электроэнергии**

Счетчик электроэнергии мы будем моделировать с помощью контроллера Arduino, который будет считывать параметры тока, получаемого путем прохождения сигнала от генератора через набор электронных компонентов, которые моделируют нагрузку электроприборов. Для моделирования экрана счетчика мы будем использовать модель ЖК дисплея. Для удобства соединения компонентов используем макетную плату, доступную в меню компонентов главного экрана справа.

Создайте новый проект в Tinkercad Circuits Arduino. Соберите счетчик электроэнергии по предложенной схеме на рис. 11. Параметры электронных компонентов и генератора сигнала представлены в табл. 1. В данной модели датчиком является сам контроллер, который вычисляет потребляемую электроэнергию, актуатором – ЖК дисплей.

Для работы модели счетчика электроэнергии необходимо написать программу управления для контроллера. Напишите предложенный программный код в контроллер, моделирующий счетчик электроэнергии. Проверьте работоспособность кода. Код предложен в приложении к данным учебно-методическим рекомендациям. Необходимо сделать выводы по проделанной работе, а также способах применения данного навыка при работе с IoT.

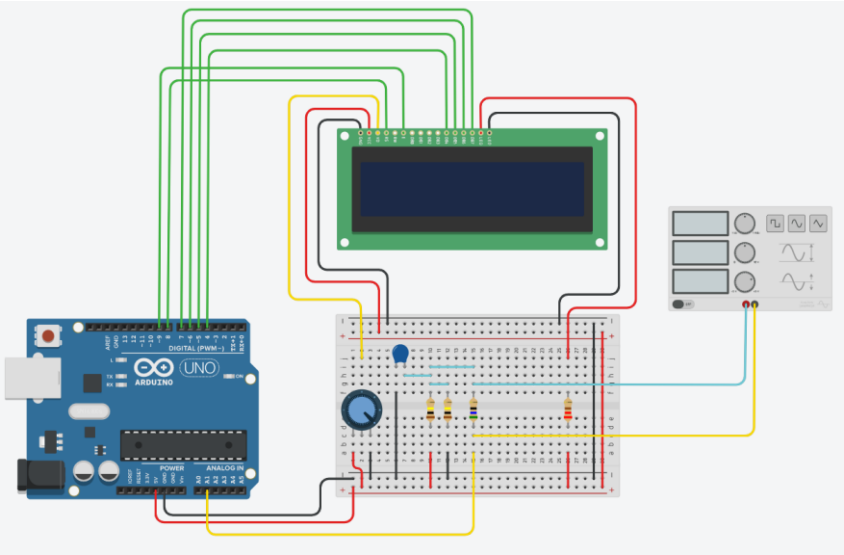

Рис. 11. Модель счетчика электроэнергии с контроллером Arduino

Таблица 1. **Перечень компонентов для модели счетчика электроэнергии**

| Компонент | Количество | Параметр | Значение параметра |
|---|---|---|---|
| ЖК-дисплей (16 x 2) | 1 | - | - |
| Генератор сигнала | 1 | Частота | 15,1 Гц |
| | | Напряжение | 10 В |
| | | Форма сигнала | синусоидальная |
| | | Сдвиг постоянной составляющей | -5В |
| Резистор | 1 | Сопротивление | |
| Резистор | 2 | Сопротивление | 100 кОм |
| Резистор | 1 | Сопротивление | 200 Ом |
| Резистор | 1 | Сопротивление | 56 Ом |
| Конденсатор | 1 | Емкость | 10 мкФ |
| Потенциометр | 1 | Сопротивление | 10 кОм |

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

5. Ссылку на готовый проект в Tinkercad (для получения ссылки в главном окне Tinkercad Circuits Arduino нажмите кнопку «Общий» в меню справа сверху.

### *Вопросы для самоконтроля*

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Охарактеризуйте применение контроллеров в решениях IoT.

3. Поясните, что такое датчик и что такое актуатор. В чем различие между этими устройствами?

4. Что такое микроконтроллер?

5. На какой части решений IoT сосредоточена данная лабораторная работа?

### *Порядок устного отчета о результатах выполненной работы преподавателю*

1. Выполнить работу по предложенному выше заданию.

2. По результатам работы оформить отчет согласно правилам, представленным выше.

3. Изучить вопросы и постараться дать краткий устный ответ на все предложенные вопросы.

4. Пояснить, какие сложности возникли при выполнении данной лабораторной работы.

# Лабораторная работа № 6
# ИЗУЧЕНИЕ ПРИНЦИПОВ РАБОТЫ IOT В ПРОГРАММЕ TINKERCAD

Основываясь на предыдущей работе, продолжим разбираться в моделировании в программе Tinkercad устройств IoT. Рассмотрим модель подключения контроллера IoT к шлюзу IoT.

На рис. 12 представлена модель подключения двух контроллеров Arduino по шине I2C. Используя данный пример, подключите второй контроллер Arduino который будет моделировать шлюз IoT к модели контроллера счетчика электроэнергии предыдущей лабораторной работы № 5.

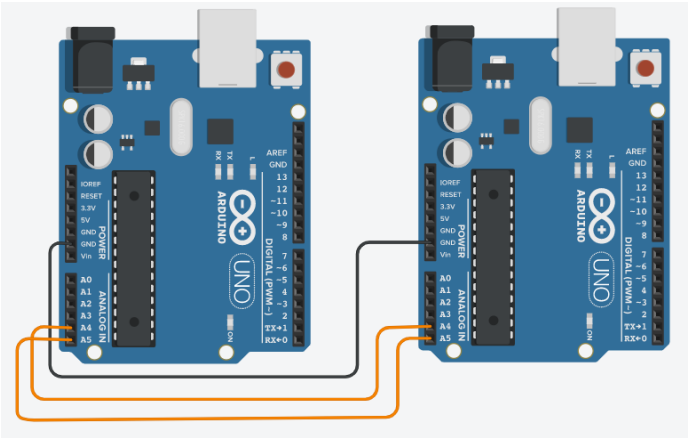


Рис. 12. Модель подключения двух плат Arduino по интерфейсу I2C

Используя примеры программного кода управления передачей данных между двумя контролерами по шине I2C, приведенные

приложении данных методических рекомендаций, напишите аналогичные программы для контроллеров счетчика электроэнергии и второго контроллера. Все параметры выбираются согласно табл. 2 и табл. 3, где номер варианта равен номеру студента в списке.

Таблица 2. **Варианты заданий**

| Параметр / Номер варианта (номер студента в списке) | Напряжение (В) | Частота переменного тока (Гц) | Количество циклов регистрации максимального и минимального значения тока | Задержка вывода значений на дисплей счетчика электроэнергии (мкс) | Скорость передачи данных (Кбит/с) |
|---|---|---|---|---|---|
| 1 | 220 | 50 | 50 | 1000 | 10 |
| 2 | 230 | 50 | 100 | 1010 | 100 |
| 3 | 240 | 50 | 150 | 1020 | 400 |
| 4 | 100 | 60 | 200 | 1030 | 1000 |
| 5 | 110 | 60 | 250 | 1040 | 3400 |
| 6 | 115 | 60 | 300 | 1050 | 5000 |
| 7 | 120 | 60 | 350 | 1060 | 100 |
| 8 | 127 | 60 | 400 | 1070 | 10 |
| 9 | 220 | 60 | 450 | 1080 | 400 |
| 10 | 230 | 60 | 500 | 1090 | 5000 |
| 11 | 240 | 60 | 550 | 1100 | 3400 |
| 12 | 100 | 50 | 600 | 1110 | 100 |
| 13 | 110 | 50 | 650 | 1120 | 5000 |
| 14 | 115 | 50 | 700 | 1130 | 10 |
| 15 | 127 | 50 | 750 | 1140 | 1000 |
| 16 | 220 | 50 | 800 | 1150 | 100 |
| 17 | 230 | 50 | 850 | 1160 | 400 |
| 18 | 240 | 50 | 900 | 1170 | 3400 |
| 19 | 100 | 60 | 950 | 1180 | 1000 |
| 20 | 110 | 60 | 1000 | 1190 | 10 |
| 21 | 115 | 60 | 1050 | 1200 | 100 |
| 22 | 120 | 60 | 1100 | 1210 | 400 |
| 23 | 127 | 60 | 1150 | 1220 | 1000 |
| 24 | 220 | 60 | 1200 | 1230 | 3400 |
| 25 | 127 | 50 | 1250 | 1240 | 5000 |
| 26 | 126 | 70 | 1250 | 1240 | 5000 |

Таблица 3. **Варианты заданий**

| Параметр / Номер варианта номер студента в списке) | Адрес ведомой платы контроллера | Задержка отправки данных (мкс) | Задержка приема данных (мкс) |
|---|---|---|---|
| 1 | 1 | 1000 | 1000 |
| 2 | 2 | 1010 | 1010 |
| 3 | 3 | 1020 | 1020 |
| 4 | 4 | 1030 | 1030 |
| 5 | 5 | 1040 | 1040 |
| 6 | 6 | 1050 | 1050 |
| 7 | 7 | 1060 | 1060 |
| 8 | 8 | 1070 | 1070 |
| 9 | 9 | 1080 | 1080 |
| 10 | 10 | 1090 | 1090 |
| 11 | 11 | 1100 | 1100 |
| 12 | 12 | 1110 | 1110 |
| 13 | 13 | 1120 | 1120 |
| 14 | 14 | 1130 | 1130 |
| 15 | 15 | 1140 | 1140 |
| 16 | 16 | 1150 | 1150 |
| 17 | 17 | 1160 | 1160 |
| 18 | 18 | 1170 | 1170 |
| 19 | 19 | 1180 | 1180 |
| 20 | 20 | 1190 | 1190 |
| 21 | 21 | 1200 | 1200 |
| 22 | 22 | 1210 | 1210 |
| 23 | 23 | 1220 | 1220 |
| 24 | 24 | 1230 | 1230 |
| 25 | 25 | 1240 | 1240 |
| 26 | 26 | 1230 | 1240 |

Программу для контроллера счетчика электроэнергии вам нужно просто дополнить необходимыми функциями аналогично примеру в предыдущей лабораторной работы № 5.

Таким образом, контроллер счетчика электроэнергии должен передавать значения потребляемой электроэнергии на второй кон-

троллер, где они должны будут выводиться в последовательный интерфейс второго контроллера.

Подумайте, как реализовать передачу значений потребляемой электроэнергии в виде отображаемых на ЖК дисплее десятичных дробей, ведь переменная, используемая в примерах, позволяет хранить только целые числа (с учетом типов данных, поддерживаемых для передачи по интерфейсу I2C)?

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Программные коды ведущей и ведомой платы Arduino.

4. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

5. Ссылку на готовый проект в Tinkercad (для получения ссылки в главном окне Tinkercad Circuits Arduino нажмите кнопку «Общий» в меню справа сверху.

### *Вопросы для самоконтроля*

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Охарактеризуйте применение контроллеров в решениях IoT.

3. Поясните, что такое датчик и что такое актуатор. В чем различие между этими устройствами?

4. На какой части решений IoT сосредоточена данная лабораторная работа?

5. Приведите примеры современных наиболее распространенных решений IoT.

6. Охарактеризуйте устройство современных контроллеров IoT.

7. Расскажите, какой контроллер использовался в данной работе и его устройство. Какие входы/выходы имеет этот контроллер?

8. Расскажите об основных интерфейсах передачи данных, которые есть у рассматриваемого в данной работе контроллера?

9. Опишите принципы работы интерфейса I2C. Какие стандартные скорости передачи данных он поддерживает и за чет чего реализуется определенная скорость передачи данных?

10. Как подключить два контролера для взаимодействия по интерфейсу I2C?

11. С помощью каких языков программирования можно программировать современные контроллеры IoT?

12. Опишите язык, используемый для программирования контроллеров в данной работе.

13. Расскажите о программе для ведущей платы контроллера.

14. Расскажите о программе для ведомой платы контроллера.

15. Какое устройство IoT моделировалось с помощью ведомой платы контроллера Arduino?

# Лабораторная работа № 7
# ПРОЕКТИРОВАНИЕ ЭЛЕКТРИЧЕСКИХ СХЕМ «УМНЫХ» УСТРОЙСТВ В IOT В ПРОГРАММЕ TINKERCAD ПРОЕКТИРОВАНИЕ ДАТЧИКА ТЕМПЕРАТУРЫ

В этой лабораторной работе необходимо сделать из Arduino термометр. Датчик температуры можно использовать для измерения температуры и регистрировать выходной сигнал с помощью трех светодиодов. Несмотря на то, что Arduino является цифровым инструментом, он может интерпретировать сигналы с аналогового входа, такого как датчик температуры, с помощью встроенного аналого-цифрового преобразователя (АЦП), доступ к которому осуществляется через аналоговые контакты.

Необходимо собрать образец схемы, показанный на рис. 13, на рабочей плоскости, запустив симуляцию, щелкнув датчик, затем, перетащив ползунок его температуры, чтобы отрегулировать смоделированный вход и наблюдать за результирующими светодиодными шаблонами.

Код приложен в приложении в конце учебно-методического пособия.

Чтобы дополнительно построить физическую схему, соберите плату Arduino Uno, USB-кабель, макетную плату без пайки, три светодиода, три одинаковых резистора (любое значение от 100 до 1K, предпочтительнее 220 Ом), датчик температуры TMP36 и провода макетной платы.

Начните с подключения Arduino и макетной платы с питанием и заземлением рядом с примерной схемой, а затем добавьте к ма-

кетной плате три красных светодиода, как показано на рис. 13. Это будут индикаторы или «гистограмма» для проекта. Перетащите Arduino Uno и макетную плату с панели компонентов на рабочую плоскость рядом с существующей схемой.

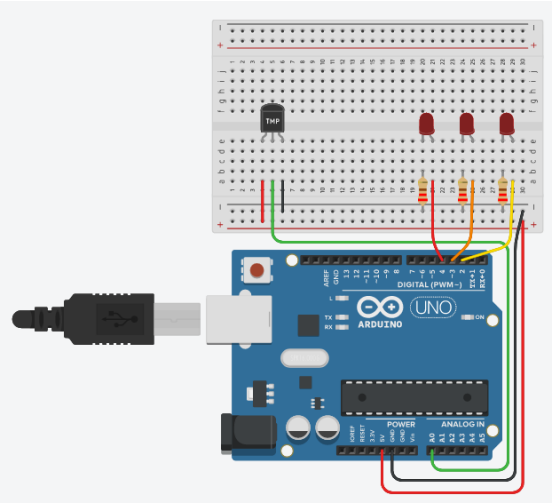


Рис. 13. Датчик температуры

Соедините 5-вольтовые контакты и контакты заземления на Arduino с шинами питания (+) и заземления (-) на макетной плате с помощью проводов. Можно изменить цвет проводов. Либо используйте раскрывающийся список, либо цифровые клавиши на клавиатуре.

Перетащите три светодиода на макетную плату в ряд E на расстоянии 2 разъемов макетной платы друг от друга. Можно изменить цвет светодиода с помощью инспектора, который появляется при нажатии на каждый из них.

Используйте резистор 220 Ом, чтобы соединить катод каждого светодиода (левая ножка) с шиной заземления (черной) макет-

ной платы. Можно изменить значение резистора, выделив его и используя раскрывающееся меню.

Подключите аноды светодиодов (правые ноги) к цифровым контактам 4, 3 и 2 на Arduino.

Анод светодиода (+) – это клемма, на которую течет ток. Это подключится к цифровым выходным контактам на Arduino. Катод (-) – это клемма, с которой течет ток. Это соединится с заземляющей шиной.

Датчик температуры создает изменяющийся сигнал напряжения в зависимости от измеряемой им температуры. Он имеет три контакта: один подключается к земле, другой подключается к 5 вольтам, а третий выводит переменное напряжение на Arduino, аналогично аналоговому сигналу от потенциометра.

Существует несколько различных моделей датчиков температуры. Эта модель TMP36 удобна тем, что ее выходное напряжение прямо пропорционально температуре в градусах Цельсия.

В редакторе схем найдите датчик температуры в ящике компонентов. Поместите датчик температуры (TMP36) на макетную плату закругленной частью в сторону от Arduino, как показано на рис. 8 (это ориентация по умолчанию).

Поместите датчик температуры на макетную плату в ряд E, как показано на рисунке. Подключите датчик температуры так, чтобы левый контакт подключался к шине напряжения 5 В, центральный контакт подключался к A0 на Arduino, а правый контакт подключался к шине GND.

На принципиальной схеме видно, что датчик температуры подключен к питанию (5 вольт) и земле (0 вольт) и аналоговому контакту A0. По мере повышения температуры контакт, подключенный к A0, увеличивает свое напряжение. Также видно, что каждый из трех светодиодов подключен к своему цифровому выводу.

Несмотря на то, что Arduino является цифровым инструментом, он может получать информацию от аналоговых датчиков для

измерения таких вещей, как температура или свет. Для этого вы воспользуетесь встроенным в Arduino аналого-цифровым преобразователем (АЦП).

Аналоговые контакты от A0 до A5 могут интерпретировать напряжения от 0 до 5 В и преобразовывать это напряжение в значение от 0 до 1023 для использования скетчем Arduino. Аналоговые контакты в основном используются для считывания информации с датчиков (но также могут использоваться как цифровые выходы 14-19, независимо), как показано на рис. 14.

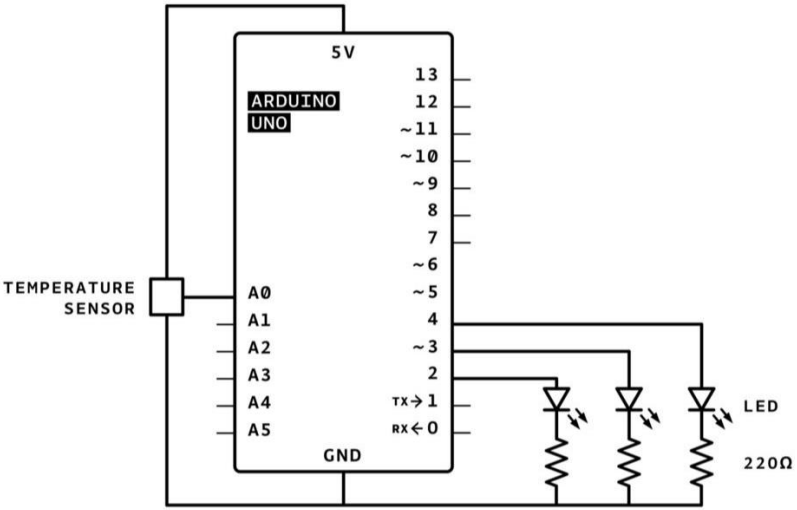


Рис. 14. Датчик температуры

Нажмите «Начать моделирование». Откройте редактор кода (код приложен в приложении) и найдите кнопку «Последовательный монитор», чтобы посмотреть, как вливаются значения датчиков.

Давайте воспользуемся редактором блоков кода, чтобы прослушать состояние датчика, а затем примем решение о том, какие светодиоды загорятся на основе значения датчика.

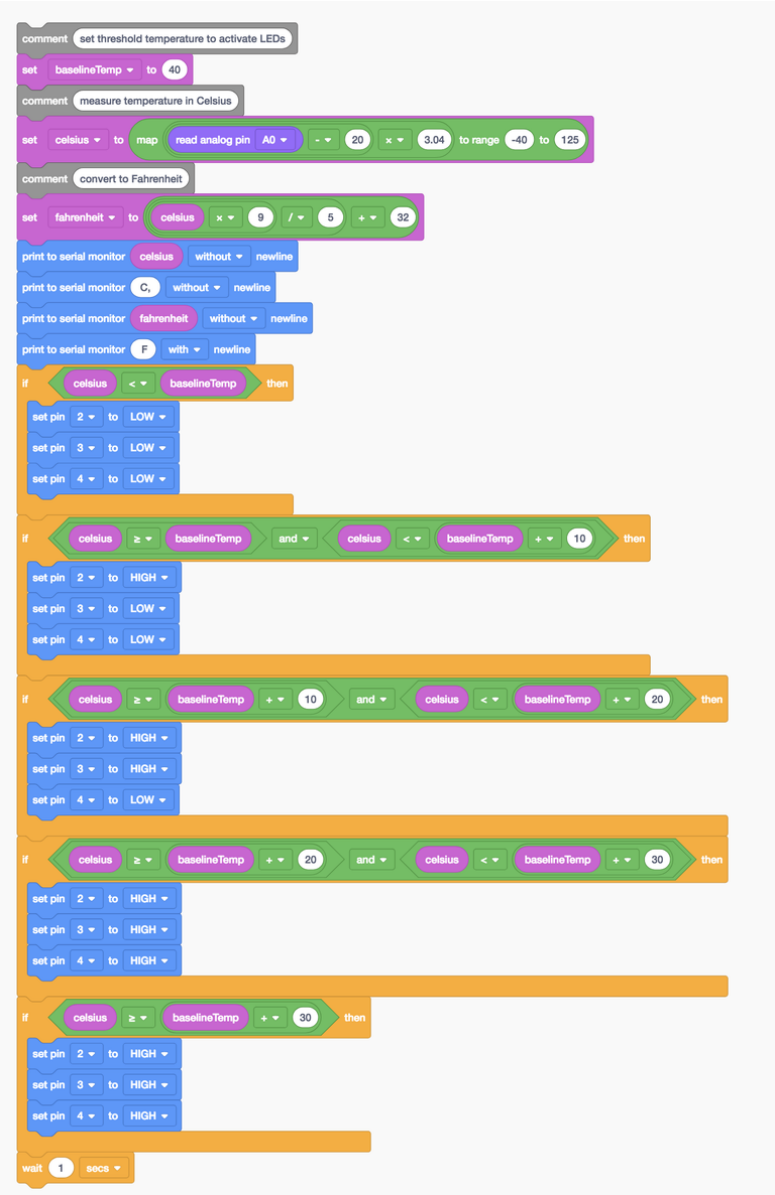


Рис. 15. Редактор блоков кода

Нажмите кнопку «Код», чтобы открыть редактор кода. Серые блоки обозначений – это комментарии для заметок о том, что вы собираетесь делать в своем коде, но этот текст не требуется и не выполняется как часть программы.

Нажмите на категорию Variables в редакторе кода. Создайте новую переменную с именем baselineTemp и используйте блок «set», чтобы установить ее на 40 (градусы C).

Чтобы сохранить значение датчика, создайте переменную с именем «celsius». Перетащите блок «set» и настройте раскрывающийся список на нашу новую переменную celsius. Настройте диапазон от -40 до 125. Нажмите на категорию «Ввод» и перетащите блок «аналоговый считывающий вывод» и поместите его в первое арифметическое поле внутри блока «карта». Настройте арифметические блоки на «(чтение аналогового контакта A0-20) x 3,04».

При желании создайте новую переменную для преобразования температуры в градусы Фаренгейта с блоком установки и некоторыми арифметическими блоками для чтения «установите градусы Фаренгейта в (цельсий x 9)/5 + 32». Добавьте несколько последовательных блоков мониторинга, чтобы распечатать температуру в одном или обоих C или F. Щелкните категорию «Управление» и перетащите блок «если-то», затем перейдите к «Математика» и перетащите блок сравнения на блок «если».

В категории «Переменные» возьмите переменную celsius и переменную baselineTemp и перетащите их в блок компаратора, настроив раскрывающийся список так, чтобы он читался «если celsius < baselineTemp then».

Добавьте три блока цифровых выходов в оператор if, чтобы установить контакты 2, 3 и 4 в НИЗКИЙ уровень.

Дублируйте этот оператор if четыре раза и добавьте арифметические блоки и/или блоки, чтобы создать пять операторов if для определения состояния. Первое состояние – «температура ниже

нашего целевого базового уровня», поэтому светодиоды не загораются. Когда температура больше или равна baselineTemp и меньше baselineTemp+10, загорается только светодиод вывода 2. Когда температура находится между baselineTemp+10 и baselineTemp+20, загораются два светодиода. И так далее для учета всех желаемых состояний.

Когда редактор кода открыт, вы можете щелкнуть раскрывающееся меню слева и выбрать «Блоки + текст», чтобы открыть код Arduino, сгенерированный блоками кода.

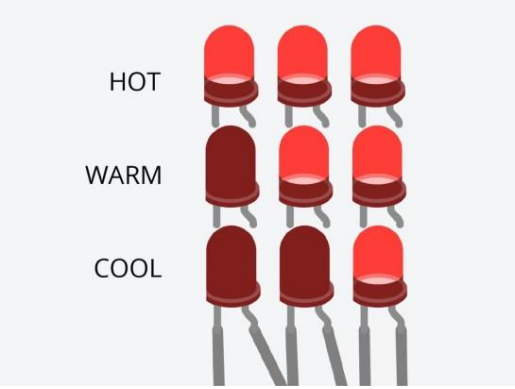


Рис. 16. Уровень температуры

```
интервал базовой линииТемп = 0;
целое число градусов Цельсия = 0;
интервал по Фаренгейту = 0;
```

Перед setup() мы создаем переменные для хранения целевой базовой температуры, а также значения датчика. Они вызываются int потому, что являются целыми числами или любыми целыми числами.

```
недействительная установка ()
{
 pinMode(A0, ВХОД);
 Серийный.начать(9600);
 pinMode(2, ВЫХОД);
 pinMode(3, ВЫХОД);
 pinMode(4, ВЫХОД);
}
```

Внутри установки контакты настраиваются с помощью pinMode() функции. Контакт A0 настроен как вход, поэтому можно «прослушивать» электрическое состояние датчика температуры. Контакты 2, 3 и 4 настроены как выходы для управления светодиодами.

```
пустой цикл ()
{
 // установить пороговую температуру для включения светодиодов
 базовая температура = 40;
 // измеряем температуру в градусах Цельсия
 Цельсия = карта (((аналоговое чтение (A0) – 20) * 3,04), 0, 1023, -40,
125);
```

Все, что следует после набора косых черт, // является комментарием, предназначенным только для чтения нами, людьми, и не включается в программу, когда ее запускает Arduino. В основном цикле baselineTemp устанавливается на целевое значение 40 градусов по Цельсию.

```
 // преобразовать в градусы Фаренгейта
 Фаренгейты = ((Цельсий * 9) / 5 + 32);
 Serial.print (по Цельсию);
 Serial.print("C,");
 Serial.print(по Фаренгейту);
 Serial.println ("F");
```

Формула для преобразования между градусами Цельсия и Фаренгейта выглядит так: F = (C * 9) / 5 + 32. Вывод на мониторе позволяет более детально наблюдать за изменением температуры, чем одно только состояние светодиодов.

```
 если (градусы Цельсия < baselineTemp) {
 цифровая запись (2, НИЗКИЙ);
 цифровая запись (3, НИЗКИЙ);
 цифровая запись (4, НИЗКИЙ);
 }
 если (градусы Цельсия >= baselineTemp && celsius < baselineTemp +
10)
 цифровая запись (2, ВЫСОКИЙ);
 цифровая запись (3, НИЗКИЙ);
 цифровая запись (4, НИЗКИЙ);
 }
 if (по Цельсию >= baselineTemp + 10 && по Цельсию < baselineTemp
+ 20) {
 цифровая запись (2, ВЫСОКИЙ);
 цифровая запись (3, ВЫСОКИЙ);
 цифровая запись (4, НИЗКИЙ);
 }
 if (по Цельсию >= baselineTemp + 20 && по Цельсию < baselineTemp
+ 30) {
 цифровая запись (2, ВЫСОКИЙ);
 цифровая запись (3, ВЫСОКИЙ);
 цифровая запись (4, ВЫСОКИЙ);
 }
 если (по Цельсию >= baselineTemp + 30) {
 цифровая запись (2, ВЫСОКИЙ);
 цифровая запись (3, ВЫСОКИЙ);
 цифровая запись (4, ВЫСОКИЙ);
 }
 задержка(1000); // Ждем 1000 миллисекунд(ы)
}
```

Шесть операторов цикла if оценивают разные сегменты определенного температурного диапазона от 40 до 46 градусов по Цельсию, и чем выше температура, тем больше светодиодов загорается.

Если нужно увидеть более очевидные изменения в освещении гистограммы, можно изменить базовую переменную температуры и/или диапазон, на который вы смотрите, изменив аргументы в операторах if().Это называется калибровкой.

Если вы построили физическую версию этой схемы, вы можете попробовать ее с монитором программного обеспечения Arduino (кнопка увеличительного стекла в правом верхнем углу окна эскиза), активировав датчик.

При использовании физической платы наблюдайте за температурой в помещении с помощью монитора последовательного порта и задайте для baselineTemp это значение. Отрегулируйте разные пороговые значения температуры в меньшем диапазоне (2, 4, 6 вместо 10, 20, 30). Загрузите свой код. По мере повышения температуры вы должны видеть, как светодиоды включаются один за другим. Вы использовали AnalogRead() и монитор для отслеживания изменений внутри вашего Arduino и создания простого дисплея температуры со светодиодами (рис. 17).

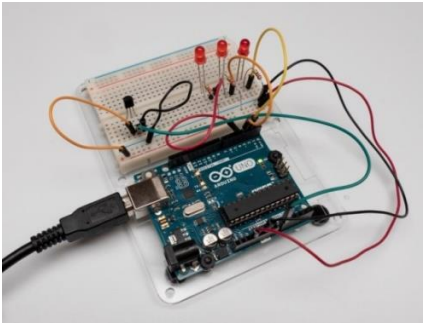

Рис. 17. Датчик температуры

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

4. Ссылку на готовый проект в Tinkercad (для получения ссылки в главном окне Tinkercad Circuits Arduino нажмите кнопку «Общий» в меню справа сверху.

### *Вопросы для самоконтроля*

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Как выглядит схема типового решение IoT?

3. Какие технологии подключения «вещей» вы знаете? В чем основное различие между стандартами подключения?

4. На какой части решений IoT сосредоточена данная лабораторная работа?

5. Приведите примеры современных наиболее распространенных решений IoT.

6. С помощью каких языков программирования можно программировать современные контроллеры IoT?

7. Опишите язык, используемый для программирования в данной работе.

8. Какое устройство IoT в данной работе моделировалось с помощью Arduino?

# Лабораторная работа № 8
# ИЗУЧЕНИЕ ПРИНЦИПОВ ФУНКЦИОНИРОВАНИЯ IOT

Изучим принципы функционирования IoT на примере датчика движения RIP в комнате с помощью Arduino. В данной лабораторной работе мы подключим схему с помощью макетной платы и используем простой код Arduino для управления тремя светодиодами [3].

Хотя датчик движения может показаться сложным со своей специальной печатной платой, он сконфигурирован для отправки ВЫСОКОГО или НИЗКОГО сигнала почти так же, как обычная кнопка.

PIR расшифровывается как Passive InfraRed, который описывает внутреннюю технологию – он пассивно определяет уровни инфракрасного света (в отличие от инфракрасной камеры, которая также может излучать инфракрасный свет, чтобы зафиксировать свое отражение).

Белый купол – это линза, расширяющая поле зрения ИК-детектора. По умолчанию датчик сообщает о НИЗКОМ сигнале, считывает количество проникающего окружающего инфракрасного света, а затем запускает ВЫСОКИЙ сигнал в течение определенного периода времени, когда уровни освещенности изменяются, указывая на движение. Он может сказать, есть ли движение в сцене, но не может определить расстояние – для этого мы можем рассмотреть тип датчика аналогового ввода, называемый ультразвуковым дальномером.

Необходимо собрать схему на рис. 18 на рабочей плоскости в программе Tinkercad, запустив симуляцию и щелкнув датчик движения. Это активирует выделенную область перед датчиком с кружком «объект» внутри. Возможно, придется изменить размер представления, если круг находится за пределами экрана.

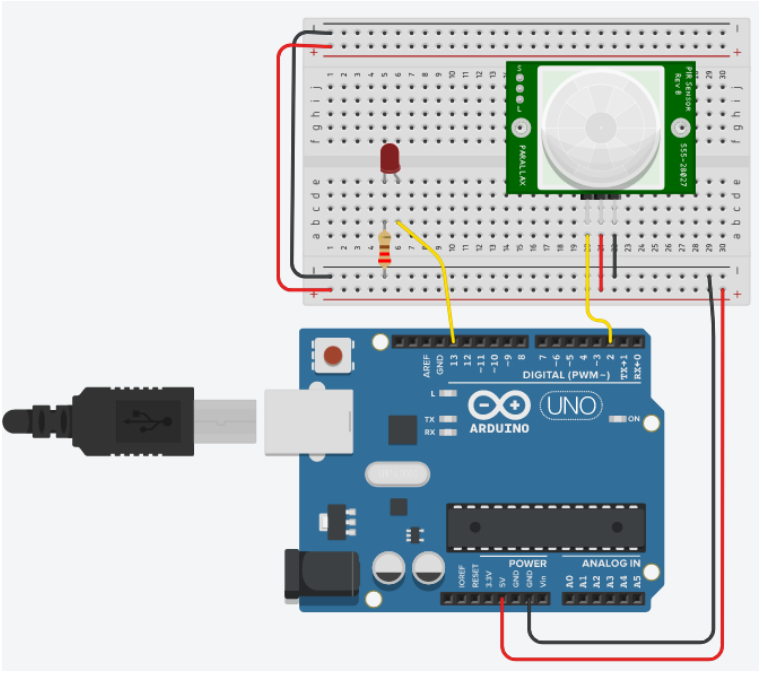


Рис. 18. Датчик движения

Нажмите и перетащите круг «объект» перед датчиком, чтобы обозначить движение.

В данной лабораторной работе вы смоделируете схему по готовому образцу. Чтобы дополнительно построить физическую схему, нужно взять плату Arduino Uno, USB-кабель, макетную плату без пайки, светодиод, резистор (любое значение от 100 до 1K), датчик движения PIR и провода макетной платы.

Взгляните на макетную схему в рабочей плоскости (рис. 18). Может быть полезно взглянуть на версию этого образца схемы со свободной разводкой для сравнения, изображенную на рис. 19. На этом шаге вы создадите собственную версию этой схемы рядом с образцом на рабочей плоскости.

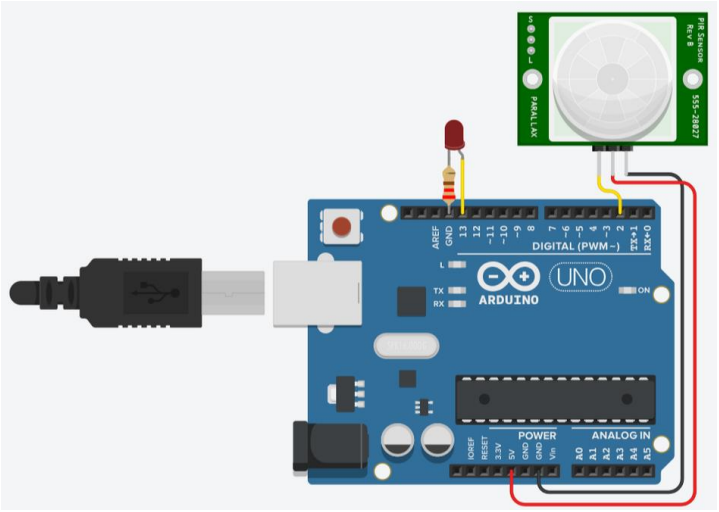


Рис. 19. Датчик движения со свободной разводкой

Определите датчик движения PIR, светодиод, резистор и провода, подключенные к Arduino, в рабочей плоскости Tinkercad Circuits. Перетащите Arduino Uno и макетную плату с панели компонентов на рабочую плоскость рядом с существующей схемой. Подключите шины питания (+) и земли (-) макетной платы к Arduino 5V и земле (GND) соответственно, щелкнув, чтобы создать провода. Продлите шины питания и заземления к соответствующим шинам на противоположном краю макетной платы, проложив красный провод между обеими шинами питания и черный провод между обеими шинами заземления. Вставьте светодиод в два разных ряда макетной платы так, чтобы катод (отрица-

тельная, более короткая ножка) соединился с одной ножкой резистора (подойдет любое сопротивление от 100 до 1 кОм). Подключите другую ногу резистора к земле. Подключите анод светодиода (положительный, более длинный) к контакту 13 Arduino.

Перетащите датчик движения PIR с панели компонентов на макетную плату, чтобы его ножки встали в три разных ряда. Нажмите, чтобы создать провод, соединяющий крайнюю правую ногу с питанием. Соедините центральную ногу с землей. Создайте провод, соединяющий крайнюю левую ногу с аналоговым выводом A0 Arduino. Давайте воспользуемся интерфейсом кодирования блоков для прослушивания датчика движения PIR, а затем примем решение о включении светодиода в зависимости от состояния датчика: активирован или не активирован (рис. 20).

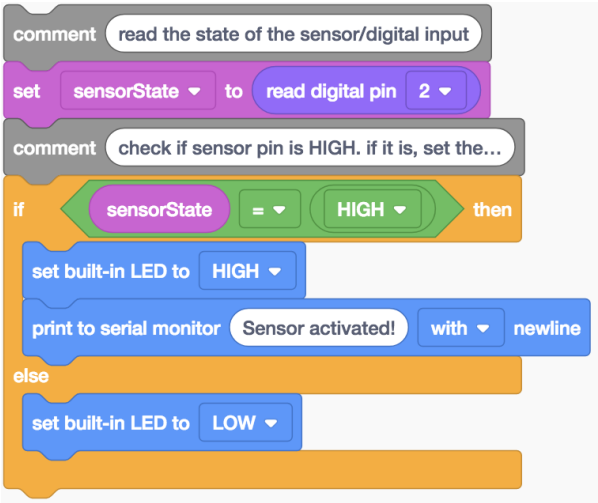


Рис. 20. Необходимые параметры

Нажмите кнопку «Код», чтобы открыть редактор кода. Код предложен в приложении к учебно-методическим рекомендациям. Нажмите на категорию Variables в редакторе кода. Создайте новую

переменную с именем sensorState. Перетащите блок «установить». Состояние нашего датчика движения PIR будет храниться в нашей переменной sensorState. Нажмите на категорию блока ввода, перетащите блок «чтение цифрового вывода» и поместите его в блок «установить» после слова «до». Поскольку наш датчик подключен к Arduino на контакте 2, измените раскрывающийся список блока «чтение цифрового контакта» на 2. Теперь ваши блоки должны читать «установить SensorState для чтения цифрового контакта 2», который сохраняет цифровое значение контакта датчика. Щелкните категорию «Управление» и перетащите блок «if then». Сконфигурируйте его, чтобы оценить, sensorState равно ли HIGH использование блока компаратора Math. Перетащите блок компаратора Math в оператор if, чтобы проверить, sensorState равна ли наша переменная HIGH. Мы хотим включить наш светодиод, если датчик активирован, в противном случае мы хотим, чтобы наш светодиод был выключен. В категории «Блок вывода» найдите блок «установить встроенный светодиод в состояние ВЫСОКИЙ». Попробуйте добавить два таких блока в наш оператор if, чтобы светодиод горел только при активации датчика. Когда редактор кода открыт, можно щелкнуть раскрывающееся меню слева и выбрать «Блоки + текст», чтобы открыть код Arduino, сгенерированный блоками кода. Перед setup() мы создаем переменную для хранения текущего состояния датчика. Он вызван int потому, что это целое число или любое целое число (хотя мы будем использовать только значения 0 и 1, LOW и HIGH).

```
недействительная установка ()
{
 pinMode(2, ВВОД);
 pinMode(13, ВЫХОД);
 Серийный.начать(9600);
}
```

Внутри установки контакты настраиваются с помощью pinMode() функции. Контакт 2 настроен как вход, поэтому мы можем «прослушивать» электрическое состояние датчика. Контакт 13 настроен как выход для управления светодиодом. Чтобы иметь возможность отправлять сообщения, Arduino открывает новый канал последовательной связи с Serial.begin(), который принимает аргумент скорости передачи (какая скорость для связи), в данном случае 9600 бит в секунду.

```
пустой цикл ()
{
 // считываем состояние датчика/цифрового входа
 Состояние датчика = цифровое Чтение(2);
```

Все, что следует после набора косых черт, // является комментарием, предназначенным только для чтения и не включается в программу, когда ее запускает Arduino. В основном цикле вызываемая функция digitalRead(); проверяет состояние контакта 2 (который будет либо 5В, он же ВЫСОКИЙ, либо заземленный, он же НИЗКИЙ), и сохраняет это состояние в переменной, которую sensorState мы создали вверху.

```
 // проверяем, находится ли вывод датчика в состоянии ВЫСОКОГО
уровня. если это так, установите
 // Светодиод горит.
 если (sensorState == ВЫСОКИЙ) {
 цифровая запись (13, ВЫСОКИЙ);
 Serial.println("Датчик активирован!");
 } еще {
 цифровая запись (13, НИЗКИЙ);
 }
 задержка (10); // Небольшая задержка для улучшения производитель
ности симуляции

}
```

Ниже еще двух строк комментариев находится оператор if, который проверяет, является ли значение sensorStateHIGH ( = = это оператор сравнения, не путать с = оператором присваивания). Если условие выполнено, встроенный светодиод становится ВЫСОКИМ (горит). Если нет, вместо этого выполняется код, содержащийся внутри else{: встроенный светодиод устанавливается в НИЗКИЙ (выкл). Операторы if могут существовать отдельно или с одним или несколькими операторами else.

Чтобы реально собрать макет для более подробного ознакомления с работой с физической платой Arduino Uno ознакомьтесь с бесплатным классом Instructables Arduino.

При реальной сборке скопируйте код из окна кода Tinkercad Circuits и вставьте его в пустой эскиз в программном обеспечении Arduino или нажмите кнопку загрузки (стрелка вниз) и откройте полученный файл с помощью Arduino. Вы также можете найти этот пример в программном обеспечении Arduino, перейдя в File -> Examples -> 02.Digital -> Button (с другим именем переменной, но в остальном то же самое).

Подключите USB-кабель и выберите плату и порт в меню инструментов программного обеспечения.

Загрузите код и наблюдайте, как ваш светодиод загорается. Некоторые датчики движения PIR поставляются с двумя регулируемыми потенциометрами для изменения чувствительности и продолжительности сигнала активации.

Попробуйте 3D-печать Tinkercad, чтобы построить корпус для электроники с отверстием для вашего датчика движения PIR. Попробуйте заменить датчик движения PIR на другие цифровые входы, такие как кнопка или переключатель наклона.

Но в данной лабораторной работе будет достаточно собрать макет в программе Tinkercad. Реальной сборки не требуется. Варианты заданий приведены в таблице 4. Выбирается по списку человека в группе.

Таблица 4. **Варианты заданий**

| Параметр пироэлектрического ИК-датчика / Номер варианта (номер студента в списке) | X | Z | Y |
|---|---|---|---|
| 1 | -4.60 | -194.68 | -294.30 |
| 2 | -5.40 | -200.56 | -302.40 |
| 3 | -5.08 | -134.67 | -199.70 |
| 4 | -5.40 | -200.56 | -302.40 |
| 5 | -5.40 | -134.67 | -294.30 |
| 6 | -5.08 | -200.56 | -294.30 |
| 7 | -4.60 | -194.68 | -294.30 |
| 8 | -5.40 | -200.56 | -302.40 |
| 9 | -5.08 | -134.67 | -199.70 |
| 10 | -5.40 | -200.56 | -302.40 |
| 11 | -5.40 | -134.67 | -294.30 |
| 12 | -5.08 | -200.56 | -294.30 |
| 13 | -4.60 | -194.68 | -294.30 |
| 14 | -5.40 | -200.56 | -302.40 |
| 15 | -5.08 | -134.67 | -199.70 |
| 16 | -5.40 | -200.56 | -302.40 |
| 17 | -5.40 | -134.67 | -294.30 |
| 18 | -4.60 | -194.68 | -294.30 |
| 19 | -5.40 | -200.56 | -302.40 |
| 20 | -5.08 | -134.67 | -199.70 |
| 21 | -5.40 | -200.56 | -302.40 |
| 22 | -5.40 | -134.67 | -294.30 |
| 23 | -5.08 | -200.56 | -294.30 |
| 24 | -4.60 | -194.68 | -294.30 |
| 25 | -5.40 | -200.56 | -302.40 |
| 26 | -5.08 | -134.67 | -199.70 |

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

4. Ссылку на готовый проект в Tinkercad (для получения ссылки в главном окне Tinkercad Circuits Arduino нажмите кнопку «Общий» в меню справа сверху.

### *Вопросы для самоконтроля*

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Как выглядит схема типового решение IoT?

3. Что было разработано в лабораторной работе?

4. На какой части решений IoT сосредоточена данная лабораторная работа?

5. Приведите примеры современных наиболее распространенных решений IoT.

6. С помощью каких языков программирования можно программировать датчики движения?

7. Опишите язык, используемый для программирования в данной работе.

8. Какое устройство IoT в данной работе моделировалось с помощью Arduino?

# Лабораторная работа № 9
# РАЗРАБОТКА ТИПОВОГО ПРОЕКТА ИНТЕРНЕТА ВЕЩЕЙ. КОНЦЕПТ-ПРОЕКТ «УМНЫЙ ДОМ» В ПРОГРАММЕ TINKERCAD

«Умный дом» предназначен для максимально комфортной жизни людей посредством использования современных высокотехнологичных средств.

Принцип работы системы «умный дом» заключается в автоматизации всего, из чего состоит жилая постройка: освещение, кондиционирование, система безопасности, электроэнергия, отопление, водоснабжение и водоотведение и так далее [1].

Соберите в программе Tinkercad проект «умного» дома, показанный на рис. 21. Программный код представлен в приложении данных учебно-методических рекомендаций.

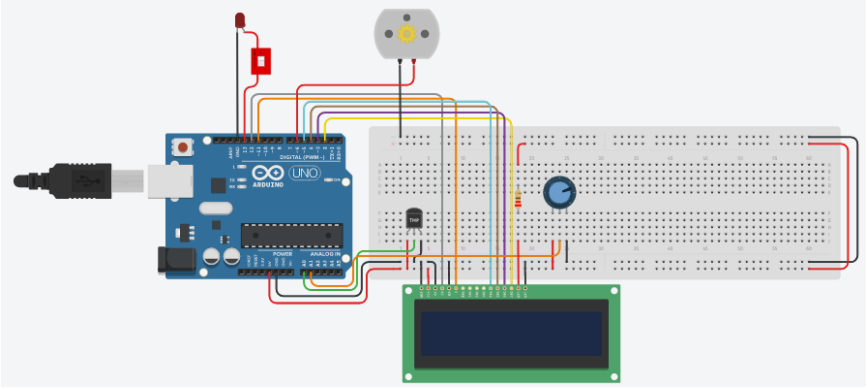

Рис. 21. Проект «умного» дома

Варианты заданий представлены в табл. 5. Вариант выбирается по списку студента в группе.

Двигая бегунок у датчика температуры, сделать выводы о функционировании «умного» дома. Что происходит с увеличением или уменьшением температуры?

Таблица 5. **Варианты заданий**

| Параметр / Номер варианта (номер студента в списке) | Сопротивление потенциометра, кОм | Цвет лампы | Сопротивление резистора, Ом |
|---|---|---|---|
| 1 | 250 | красный | 220 |
| 2 | 240 | зеленый | 250 |
| 3 | 230 | желтый | 240 |
| 4 | 200 | оранжевый | 230 |
| 5 | 210 | синий | 200 |
| 6 | 220 | белый | 210 |
| 7 | 250 | красный | 220 |
| 8 | 240 | зеленый | 250 |
| 9 | 230 | желтый | 230 |
| 10 | 200 | оранжевый | 200 |
| 11 | 210 | синий | 210 |
| 12 | 220 | белый | 220 |
| 13 | 250 | красный | 250 |
| 14 | 240 | зеленый | 240 |
| 15 | 230 | желтый | 230 |
| 16 | 200 | оранжевый | 200 |
| 17 | 210 | синий | 230 |
| 18 | 220 | белый | 200 |
| 19 | 250 | красный | 240 |
| 20 | 240 | зеленый | 230 |
| 21 | 230 | желтый | 200 |
| 22 | 200 | оранжевый | 210 |
| 23 | 210 | синий | 220 |
| 24 | 220 | белый | 250 |
| 25 | 250 | красный | 240 |
| 26 | 240 | зеленый | 240 |

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

4. Ссылку на готовый проект в Tinkercad (для получения ссылки в главном окне Tinkercad Circuits Arduino нажмите кнопку «Общий» в меню справа сверху.

### *Вопросы для самоконтроля*

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Поясните, какие составляющие входят в систему «умного» дома?

3. Что такое микроконтроллер?

4. На какой части решений IoT сосредоточена данная лабораторная работа?

5. Что происходит с увеличением или уменьшением температуры?

### *Порядок устного отчета о результатах выполненной работы преподавателю*

1. Выполнить работу по предложенному выше заданию.

2. По результатам работы оформить отчет согласно правилам представленным выше.

3. Пояснить, какие сложности возникли при выполнении данной лабораторной работы.

# Лабораторная работа № 10
# РАЗРАБОТКА «УМНОГО» САДА В ПРОГРАММЕ TINKERCAD

«Умный» сад – это особая система, которая облегчает управление тяжёлыми техническими процессами на дачном участке. Такие разработки применяются на масштабных садово-парковых зонах, а также на приусадебных участках.

В наше время многие действия в саду или огороде автоматизированы. Эта система не может сделать за дачника всю его работу, она способна оказать помощь только в поливе.

В умном саду применяются электронные линии, которые полностью автоматизируют полив. Работа системы зависит от степени влажности почвы, которая отражается на блоке управления, определяясь особыми датчиками.

Умный сад изготавливают, базируясь на имеющихся проектах ли по индивидуальному заказу, потому что дачные участки различаются помимо размеров, ещё и содержимым.

Для любого участка подбирается свой вид системы, план распределения приборов, разнообразные механизмы управления и наблюдения. В качестве вспомогательного объекта, по желанию можно обзавестись видеокамерой.

Что же включает в себя система умного сада? Приборы, фиксирующие движение. Они срабатывают на пернатых, представителей животного мира, людей. Реакция составных частей приводит к замыканию реле и передачу импульса на блок управления, и тут же срабатывает звуковое оповещение.

Приборы, реагирующие на изменение степени влажности. Если грунт сильно просыхает, то система включает режим полива.

Приборы освещения. Они подключают фонарики, с наступлением тёмного времени суток.

Система умного сада не сложная и комфортна в применении. Владелец этой системы может контролировать положение дел на своём огороде с использованием маленького дисплея с сенсорным управлением. На нем показан план участка, со всеми частями, используемыми для управления. Следует дотронуться до изображения лампочки, и она загорится, и на дисплее, и на участке.

Действия умного сада: осуществляет полив деревьев, клумб, овощей. Включением фонариков и декоративного освещения в саду. Приводит в действие декоративные фонтаны и музыкальное сопровождение. Деятельность многих умных садов базируется на привлечении разнообразных приборов и других электронных систем. Выпускаются более дорогостоящие модели, где нет проводов. Все составные части их соединены между собой системой WI-FI. Проверка подобных систем производится через мобильное приложение или веб-сайт.

«Умный сад» прост в использовании и безопасен. Однако есть и минусы. Основным из них, пожалуй, является высокая цена.

Отметим же достоинства системы:

1. Бережёт время. Наличие автоматизированных процессов, позволяет человеку отсутствовать при проведении некоторых мероприятий, связанных с уходом за культурами. Так же не возникает нужды включать кнопки, отвечающие за совершение определённых действий, вся деятельность производится в соответствии с программой.

2. Обособленность. Участок некоторый период времени, может прекрасно обходиться без участия владельца. Последний, может уехать отдохнуть или по работе и не думать при этом, что культуры не получают должного ухода.

3. Порядок и гарантированный урожай. На умный сад можно рассчитывать, он не подводит и положенный период производит

то или иное действие. Постоянный уход даёт возможность культурам хорошо расти и развиваться, радовать отличной урожайностью.

4. Обширное наблюдение. Система контролирует всё, что находится за пределами дома: коммуникации, декоративность и прочее.

В данной лабораторной работе реализуем такой проект в программе Tinkercad (рис. 22).

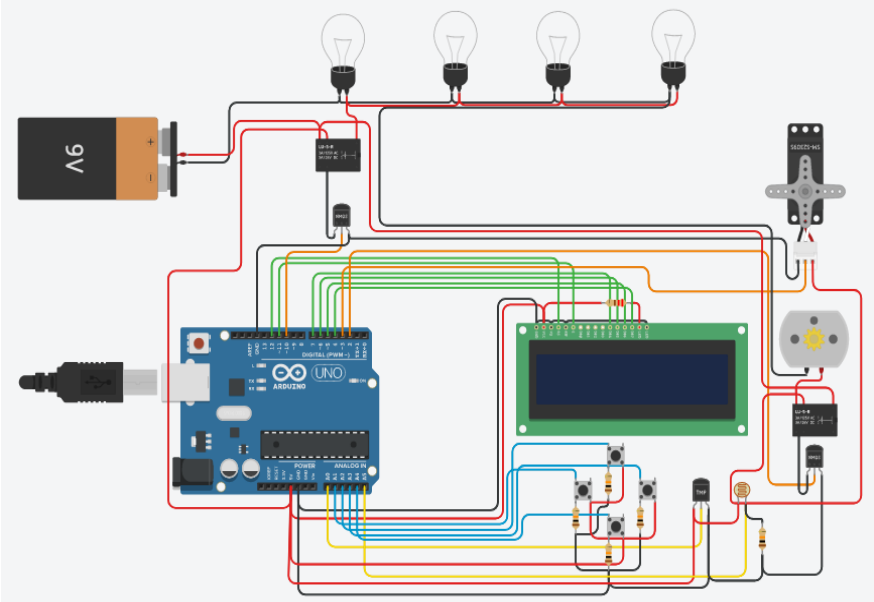


Рис. 22. Проект «умного» сада

Варианты заданий представлены в табл. 6. Вариант выбирается по списку студента в группе. Программный код представлен в приложении данных учебно-методических рекомендаций.

Таблица 6. **Варианты заданий**

| Параметр / Номер варианта (номер студента в списке) | Сопротивление резисторов № 1 и № 2, кОм | Сопротивление резисторов № 3 и № 4, кОм | Сопротивление резисторов № 5 и № 6, кОм |
|---|---|---|---|
| 1 | 11 | 15 | 10 |
| 2 | 13 | 10 | 12 |
| 3 | 14 | 12 | 11 |
| 4 | 15 | 11 | 13 |
| 5 | 10 | 13 | 14 |
| 6 | 12 | 14 | 15 |
| 7 | 11 | 15 | 10 |
| 8 | 13 | 10 | 12 |
| 9 | 14 | 12 | 11 |
| 10 | 15 | 11 | 13 |
| 11 | 10 | 13 | 14 |
| 12 | 12 | 15 | 15 |
| 13 | 11 | 10 | 10 |
| 14 | 13 | 12 | 12 |
| 15 | 14 | 11 | 11 |
| 16 | 15 | 13 | 13 |
| 17 | 10 | 14 | 14 |
| 18 | 12 | 15 | 15 |
| 19 | 11 | 10 | 10 |
| 20 | 13 | 12 | 12 |
| 21 | 14 | 11 | 11 |
| 22 | 15 | 13 | 13 |
| 23 | 10 | 15 | 14 |
| 24 | 12 | 10 | 15 |
| 25 | 11 | 12 | 10 |
| 26 | 13 | 11 | 12 |

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

5. Ссылку на готовый проект в Tinkercad (для получения ссылки в главном окне Tinkercad Circuits Arduino нажмите кнопку «Общий» в меню справа сверху.

### *Вопросы для самоконтроля*

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Поясните, какие составляющие входят в систему «умного» дома?

3. Что такое «умный» сад?

4. На какой части решений IoT сосредоточена данная лабораторная работа?

5. Что происходит с увеличением или уменьшением температуры?

### *Порядок устного отчета о результатах выполненной работы преподавателю*

1. Выполнить работу по предложенному выше заданию.

2. По результатам работы оформить отчет согласно правилам представленным выше.

3. Пояснить, какие сложности возникли при выполнении данной лабораторной работы.

# Лабораторная работа № 11
# РАЗРАБОТКА ПРОЕКТА «УМНЫЙ» ГОРОД В ПРОГРАММЕ TINKERCAD

Концепция «умный город» в последнее время у многих на слуху, но не все разбираются, что это такое и как работает. Это не значит, что мегаполис живет как самостоятельный организм, «умными» сделаны его некоторые компоненты. Пример концепт-проекта был выполнен нами в предыдущих лабораторных работах, где мы изучили достоинства и недостатки, а также возможности «умного» города. Теперь разработаем модель «умного» города в программе Tinkercad.

Еще раз поговорим о примерах. Подъезжая к корпусу 1 на ул. Молодогвардейская 151 г. Самары, мы могли бы наблюдать «умные» парковки, на въезде которых на табло указано количество пустых мест. А также камеры видеонаблюдения на вокзалах, в аэропортах, домофонах многоквартирных домов, которые подключены к единой системе.

«Умный город» способен решить вопрос с пробками на дорогах, освещением улиц, преступностью, вывозом мусора, предотвратить крупный пожар, спасти человека, которому внезапно стало плохо дома или в общественном месте, и многое другое. Благодаря информационно – коммуникационным технологиям (ИКТ) городские власти могут отслеживать развитие городской среды, выявлять наиболее эффективные инструменты для повышения качества жизни, напрямую взаимодействовать с сообществами и городской инфраструктурой.

Сбор, обработка и анализ данных, поступающих от жителей города, происходит за счет использования датчиков, функциони-

рующих в режиме реального времени. Полученные сведения позволяют выявлять и заменять нерабочие методики. Использование ИКТ позволяет повысить качество, производительность и интерактивность городских служб, снизить расходы и потребление ресурсов, улучшить качество связи между жителями города и государством. Цель развития технологий «умного города» заключается в повышении качества управления городскими потоками и скоростью решения сложных задач.

Компании Tesla, Uber и Google уже готовы запустить свои автомобили на городские улицы, а со временем беспилотники, способные распознавать дорожную разметку, заменят обычные механические транспортные средства.

Программа «умного города» объединит беспилотные авто в единую сеть, что позволит при планировании маршрута учитывать их взаимное местоположение, изучать расстояние между ними, избегать аварий. Такое решение повысит безопасность дорожного движения, поскольку беспилотные автомобили заранее будут получать данные о дорожной ситуации, пробках, ДТП и свободных парковочных местах.

В данной лабораторной работе реализуем простой проект в программе Tinkercad (рис. 23).

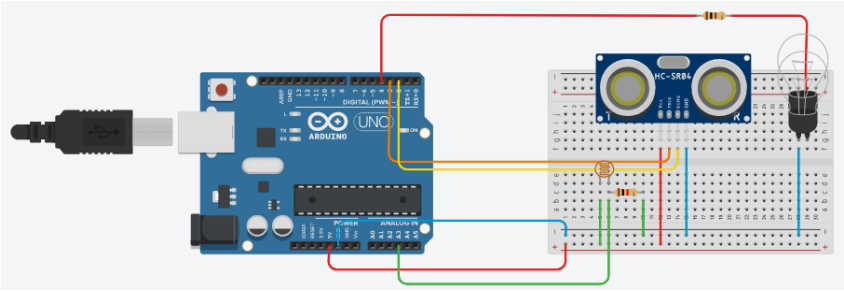


Рис. 23. Пример проекта «умный» город

Варианты заданий представлены в табл. 7. Вариант выбирается по списку студента в группе. Программный код представлен в приложении данных учебно-методических рекомендаций.

Таблица 7. **Варианты заданий**

| Параметр / Номер варианта (номер студента в списке) | Сопротивление резистора №1 кОм | Сопротивление резистора №2 кОм | Сопротивление фоторезистора №3, кОм |
|---|---|---|---|
| 1 | 11 | 15 | 10 |
| 2 | 13 | 10 | 12 |
| 3 | 14 | 12 | 11 |
| 4 | 15 | 11 | 13 |
| 5 | 10 | 13 | 14 |
| 6 | 12 | 14 | 15 |
| 7 | 11 | 15 | 10 |
| 8 | 13 | 10 | 12 |
| 9 | 14 | 12 | 11 |
| 10 | 15 | 11 | 13 |
| 11 | 10 | 13 | 14 |
| 12 | 12 | 15 | 15 |
| 13 | 11 | 10 | 10 |
| 14 | 13 | 12 | 12 |
| 15 | 14 | 11 | 11 |
| 16 | 15 | 13 | 13 |
| 17 | 10 | 14 | 14 |
| 18 | 12 | 15 | 15 |
| 19 | 11 | 10 | 10 |
| 20 | 13 | 12 | 12 |
| 21 | 14 | 11 | 11 |
| 22 | 15 | 13 | 13 |
| 23 | 10 | 15 | 14 |
| 24 | 12 | 10 | 15 |
| 25 | 11 | 12 | 10 |
| 26 | 13 | 11 | 12 |

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

4. Ссылку на готовый проект в Tinkercad (для получения ссылки в главном окне Tinkercad Circuits Arduino нажмите кнопку «Общий» в меню справа сверху.

### Вопросы для самоконтроля

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Поясните, какие составляющие входят в систему «умного» города?

3. Что такое «умный» город?

4. На какой части решений IoT сосредоточена данная лабораторная работа?

5. Что конкретно было реализовано в данной лабораторной работе?

### *Порядок устного отчета о результатах выполненной работы преподавателю*

1. Выполнить работу по предложенному выше заданию.

2. По результатам работы оформить отчет согласно правилам, представленным выше.

3. Пояснить, какие сложности возникли при выполнении данной лабораторной работы.

# Лабораторная работа № 12
# РАЗРАБОТКА ПРОЕКТА «УМНЫЙ» ГАРАЖ В ПРОГРАММЕ TINKERCAD

Умный гараж – это гараж, который подключен к интернету и может обмениваться данными с другими устройствами и сервисами. Такой гараж может реагировать на разные события и сценарии, например, включать свет при заезде автомобиля, оповещать о проникновении посторонних или снижать температуру при отсутствии людей.

Умный гараж также может экономить энергию и деньги, оптимизируя расход ресурсов и предотвращая поломки. Простые системы управления – это те, которые позволяют управлять гаражом с помощью пульта, смартфона или голосового ассистента.

Они обычно подключаются к интернету через Wi-Fi или Bluetooth и работают с облачными сервисами или приложениями на телефоне. Они не требуют сложной настройки или проводки и подходят для тех, кто хочет модернизировать свой гараж.

Примеры таких систем [4]:

1. Chamberlain MyQ. Это одна из самых популярных систем управления для гаражных ворот. Она состоит из специального модуля, который подключается к электроприводу ворот, и приложения на смартфоне. С помощью приложения можно открывать и закрывать ворота, проверять их состояние, получать уведомления о движении и настраивать расписание работы.

2. Nexx Garage. Это еще одна система управления для гаражных ворот, которая работает похожим образом. Она также состоит

из модуля и приложения, но имеет несколько дополнительных функций, таких как геолокация, которая позволяет открывать ворота автоматически при приближении к гаражу, или распознавание номеров, которое позволяет открывать ворота только для определенных автомобилей.

3. Яндекс Алиса. Это голосовой ассистент, который может управлять не только гаражными воротами, но и другими функциями гаража, такими как освещение, отопление, вентиляция и т.д.

В данной лабораторной работе реализуем простой проект «умного» гаража в программе Tinkercad (рис. 24).

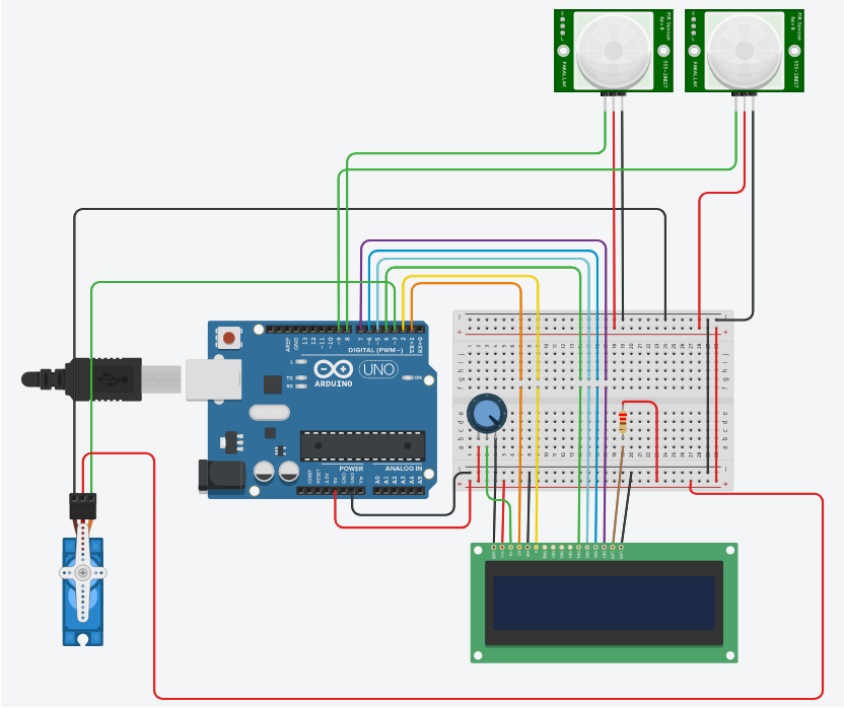


Рис. 24. Проект «умного» гаража

Варианты заданий представлены в табл. 8. Вариант выбирается по списку студента в группе. Программный код представлен в приложении данных учебно-методических рекомендаций.

Таблица 8. **Варианты заданий**

| Параметр / Номер варианта (номер студента в списке) | Сопротивление потенциометра, кОм | Сопротивление резистора, Ом | Пироэлектрический ИК-датчик 1, (X,Y,Z) | Пироэлектрический ИК-датчик 2, (X,Y,Z) |
|---|---|---|---|---|
| 1 | 250 | 230 | (-30,-234,-254) | (31,-181,-189) |
| 2 | 240 | 200 | (-40,-240,-255) | (40,-200,-199) |
| 3 | 230 | 210 | (-31,-235,-230) | (41,-210,-190) |
| 4 | 200 | 220 | (-32,-245,-231) | (21,-220,-195) |
| 5 | 210 | 250 | (-40,-240,-255) | (40,-200,-199) |
| 6 | 220 | 240 | (-31,-235,-230) | (41,-210,-190) |
| 7 | 250 | 230 | (-32,-245,-231) | (21,-220,-195) |
| 8 | 240 | 200 | (-40,-240,-255) | (40,-200,-199) |
| 9 | 230 | 210 | (-31,-235,-230) | (41,-210,-190) |
| 10 | 200 | 220 | (-30,-234,-254) | (31,-181,-189) |
| 11 | 210 | 250 | (-40,-240,-255) | (40,-200,-199) |
| 12 | 220 | 240 | (-31,-235,-230) | (41,-210,-190) |
| 13 | 250 | 230 | (-32,-245,-231) | (21,-220,-195) |
| 14 | 240 | 200 | (-40,-240,-255) | (40,-200,-199) |
| 15 | 230 | 210 | (-31,-235,-230) | (41,-210,-190) |
| 16 | 200 | 220 | (-32,-245,-231) | (21,-220,-195) |
| 17 | 210 | 250 | (-40,-240,-255) | (40,-200,-199) |
| 18 | 220 | 240 | (-31,-235,-230) | (41,-210,-190) |
| 19 | 250 | 230 | (-30,-234,-254) | (31,-181,-189) |
| 20 | 240 | 200 | (-40,-240,-255) | (40,-200,-199) |
| 21 | 230 | 210 | (-31,-235,-230) | (41,-210,-190) |
| 22 | 200 | 220 | (-32,-245,-231) | (21,-220,-195) |
| 23 | 210 | 250 | (-40,-240,-255) | (40,-200,-199) |
| 24 | 220 | 240 | (-31,-235,-230) | (41,-210,-190) |
| 25 | 250 | 230 | (-32,-245,-231) | (21,-220,-195) |
| 26 | 240 | 200 | (-40,-240,-255) | (40,-200,-199) |

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

4. Ссылку на готовый проект в Tinkercad (для получения ссылки в главном окне Tinkercad Circuits Arduino нажмите кнопку «Общий» в меню справа сверху.

### *Вопросы для самоконтроля*

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Поясните, какие составляющие входят в систему «умного» гаража?

3. Что такое «умный» гараж?

4. На какой части решений IoT сосредоточена данная лабораторная работа?

5. Что конкретно было реализовано в данной лабораторной работе?

### *Порядок устного отчета о результатах выполненной работы преподавателю*

1. Выполнить работу по предложенному выше заданию.

2. По результатам работы оформить отчет согласно правилам, представленным выше.

3. Пояснить, какие сложности возникли при выполнении данной лабораторной работы.

# Лабораторная работа № 13
# ПРОЕКТИРОВАНИЕ ЦИФРОВЫХ КОММУНИКАЦИОННЫХ СХЕМ И ИХ АВТОМАТИЗАЦИЯ ANYLOGIC

В данной лабораторной работе в результате построения этой модели будут рассмотрены следующие новые вопросы:

1. Создание нового класса активного объекта.
2. События.
3. Значки активного объекта.
4. Порты и сообщения;
5. Действия при получении сообщений.

В данной работе мы построим дискретно-событийную модель «умного» счетчика средствами AnyLogic. Системы называются дискретно-событийными, если изменения переменных состояния в них происходят только в явно определенные моменты времени или под влиянием явно определенных событий. Находясь в некотором состоянии, дискретная система сохраняет его до наступления очередного события, под воздействием которого переменные системы и, следовательно, ее состояние изменяются скачком [5].

**Дискретная модель «умного» счетчика**

В счетчике генератор посылает устройству отображения какой-то сигнал («тик»). Каждый разряд десятичного счетчика считает число пришедших на его вход «тиков» по модулю 10 и передает на свой выход сигнал переполнения после прихода на его вход каждого десятого сигнала. При построении модели такого

счетчика нужно использовать средства, характерные для моделей дискретно-событийных систем. Здесь нам достаточно трех таких средств: события, порта и передаваемых через порт сообщений.

Нужно построить модель трехразрядного десятичного счетчика, работающего от импульсного генератора. На рис. 25 представлен счетчик, насчитавший 258 импульсов от генератора.

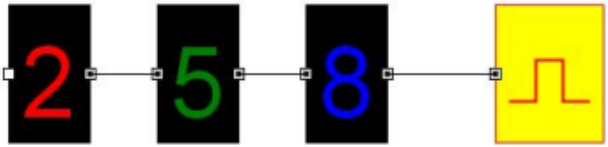


Рис. 25. Счетчик

Для реализации имитационной модели нужно построить генератор «тиков», работающий с заданной частотой, и три одинаковых десятичных разряда счетчика. Следовательно, модель должна содержать три класса активных объектов: генератор «тиков», разряд счетчика по модулю 10 и, кроме того, корневой активный объект, который будет включать в себя один экземпляр генератора и три одинаковых экземпляра одноразрядного счетчика, связанные подходящим образом. Необходимо скачать и установить программу AnyLogic по ссылке: https://cloud.mail.ru/public/86HY/B25dcUFtv.

Создайте в своей рабочей директории новый проект под названием DCounter. Корневой объект назовите Model. Модель будет состоять из нескольких подсистем, связанных между собой.

Генератор посылает сигналы с определенной частотой разряду счетчика. Для этого предназначен специальный пакет данных – сообщение. Сообщения принимаются и посылаются через специальные элементы активных объектов – порты. Обмен сообщениями возможен только между портами, соединенными соединителя-

ми – элементами, играющими роль путей движения сообщений. В нашей модели счетчика важен только сам факт передачи сообщений, а не содержимое. Прием сообщения будет вызывать увеличение значения разряда счетчика.

Постройте новый активный объект Gen. Для этого в панели **Проекты** щелкните правой кнопкой мыши по имени проекта DCounter и в появившемся контекстном меню выберите команду **Создать/Класс активного объекта**. Назовите новый класс Gen.

Класс активного объекта Gen должен посылать сообщения первому разряду счетчика с заданной частотой. Для генерации таких сообщений создадим **Событие**, для этого перетащите элемент **Событие**  из палитры **Основная** на диаграмму класса активного объекта Gen. В окне свойств этого события нужно оставить **Тип события** По таймауту без изменений, а **Режим** установить Циклический. Циклический режим события позволяет пользователю выполнять некоторые действия с требуемой периодичностью, например, каждое утро или ежегодно. Период срабатывания обратно-пропорционален частоте, поэтому в поле **Период** запишите 1/frequency. В нашей модели frequency – это частота следования импульсов генератора, ее нужно объявить как параметр модели. Создайте в классе Gen параметр frequency со значением по умолчанию равным 2.

Внешний вид генератора, а точнее объекта Gen представим прямоугольником, содержащим изображение импульса, как показано в правой части рис. 20. Откройте палитру **Презентация** и перетащите на диаграмму класса Gen прямоугольник, в свойствах укажите ширину этого прямоугольника – 80 единиц, а высоту – 100. Залейте его желтым цветом, цвет линии границы сделайте красным, а толщину линии равной 1 (рис. 26).

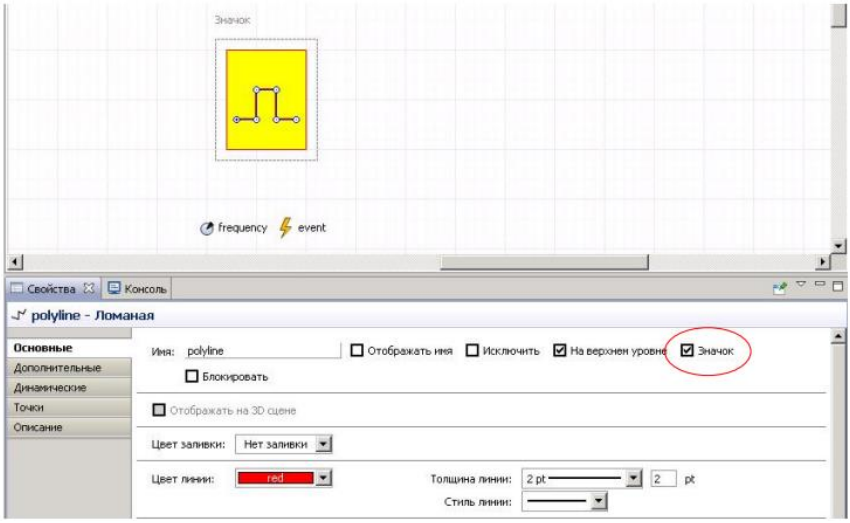


Рис. 26. Внешний вид генератора

Сделайте двойной щелчок мышью по элементу **Ломаная** в палитре (при этом его значок должен поменяться на этот ). Теперь Вы можете рисовать ломаную точка за точкой, последовательно щелкая мышью в тех точках диаграммы, куда вы хотите поместить вершины ломаной. Чтобы завершить рисование, добавьте последнюю точку ломаной двойным щелчком мыши. Нарисуйте в поле прямоугольника импульс в соответствии с рис. 27.

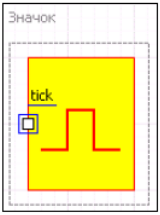


Рис. 27. Импульс

Для того чтобы вложенные объекты – экземпляры класса (Gen) отображались на структурной диаграмме верхнего уровня (Model) в графическом виде и имели интерфейсные элементы – порты, нужно нарисовать для вложенного объекта значок. Мы уже нарисовали прямоугольник с импульсом внутри, для того чтобы сделать их значком выделите последовательно каждую из фигур и на вкладке **Основные** панели **Свойства** установите флажок **Значок**, как показано на рис. 27. Вокруг прямоугольника появится пунктирная линия и надпись Значок.

Как уже говорилось, порты играют центральную роль в механизме передачи сообщений. Сообщения посылаются и получаются портами. Порты являются двунаправленными, через них сообщения могут и посылаться, и приниматься. Перетащите элемент **Порт** из палитры **Основная** на диаграмму класса Gen, поместите его на левую сторону желтого прямоугольника и назовите этот порт tick (рис. 27).

Через этот порт наш экземпляр класса активного объекта Gen может посылать и принимать сообщения от других активных объектов.

Генератор нашей модели должен периодически с периодом 1/frequency посылать сообщения через порт tick. Ранее мы создали событие event, циклически срабатывающее с частотой frequency. При каждом срабатывании генератор должен выполнить действие – послать новое сообщение типа Object через выходной порт с именем tick. Для этого введите в поле **Действие** окна свойств события следующую строку:

```
tick.send ( new Object( ) );
```

Введите еще один класс активного объекта в проект – Counter. Этот класс будет моделировать разряд десятичного счетчика. Он должен иметь один целый параметр – n, в котором будет храниться значение данного разряда, и два порта: tick и overflow. Создайте

параметр n целого типа в классе Counter с начальным значением = 0.

Разряд счетчика представим в виде прямоугольника с цифрой разряда внутри. На диаграмме класса Counter нарисуйте прямоугольник 60x100. Залейте его и линию рамки черным цветом. Внутрь прямоугольника вставьте цифру в виде текстового символа. Для этого поместите на прямоугольник текст (кнопка Aa на палитре **Презентация**) со значением 0 в поле **Текст**, вкладки **Основные**, свойств вставленного текста. Установите параметры текста: шрифт **SansSerif**, размер 72, цвет желтый. Во вкладку **Динамические** в поле **Текст** поместите имя переменной n, значение которой требуется отображать. Отметьте галочкой выбор **Значок** в свойствах текста и свойствах черного прямоугольника, рис. 28.

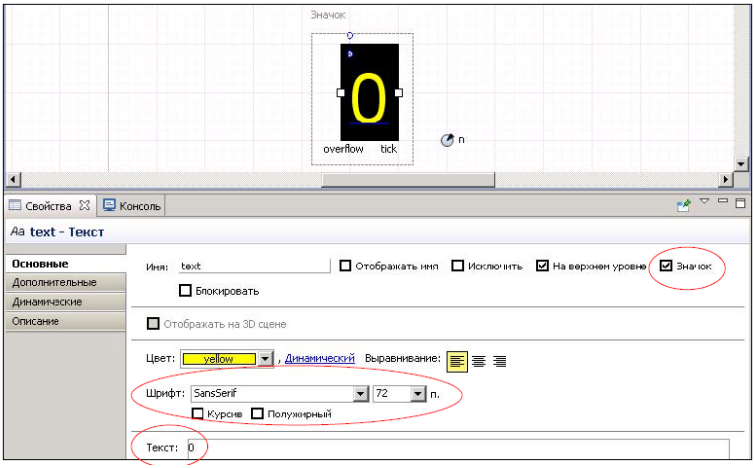


Рис. 28. Интерфейс выбора

Создайте два порта: tick и overflow и поместите их на диаграмму класса Counter, как показано на рис. 28.

Каждый раз, когда приходит сообщение в порт tick, параметр n должен увеличиваться на единицу по модулю 10 с извещением по порту overflow о переполнении, если оно наступило. Это значит, что разряд счетчика увеличивает значение своего параметра n на 1 при приеме каждого сообщения, пришедшего через порт tick, кроме случая, когда n равно 9. В этом случае параметр n должен принять значение 0, а через выходной порт overflow будет послано сообщение. Именно это следует записать в поле **Действие при получении** в свойствах порта tick (рис. 29).

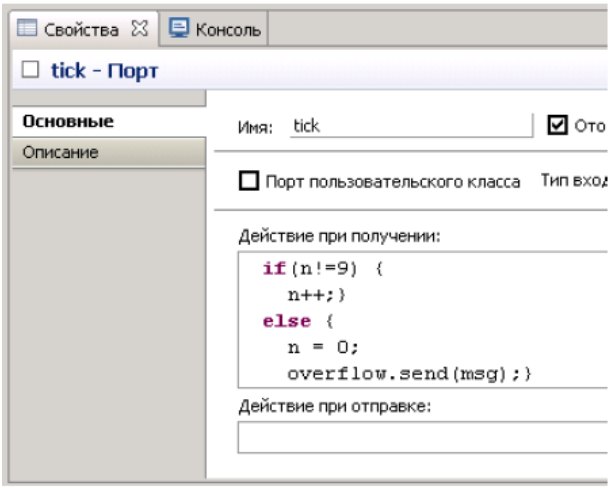


Рис. 29. Свойства

Через порт overflow, таким образом, будет передаваться каждое десятое сообщение, полученное в порт tick. Структура корневого объекта Model должна содержать один экземпляр активного объекта Gen и три экземпляра разряда счетчика Counter, соединенных по соответствующим портам. Для создания экземпляра генератора в корневом объекте, откройте диаграмму класса активного

объекта Model и перетащите на нее один экземпляр генератора (нажав левую клавишу мыши на имени Gen), а затем и три экземпляра счетчика – Counter. Экземпляры активных объектов получат имена по умолчанию будут выглядеть так, как показано на рис. 30.

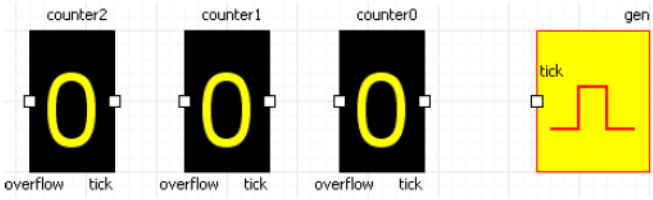


Рис. 30. Счетчик

Чтобы установить взаимодействие между объектами, Вам нужно соединить порты этих объектов с помощью соединителей. Соединитель – это линия, соединяющая два порта. Сделайте двойной щелчок мышью по одному из портов. Последовательно щелкните в тех местах диаграммы, где Вы хотите поместить точки изгиба соединителя (для нашей модели не обязательно). Завершите процесс соединения, сделав двойной щелчок мышью по второму порту. Запустите модель (рис. 31).

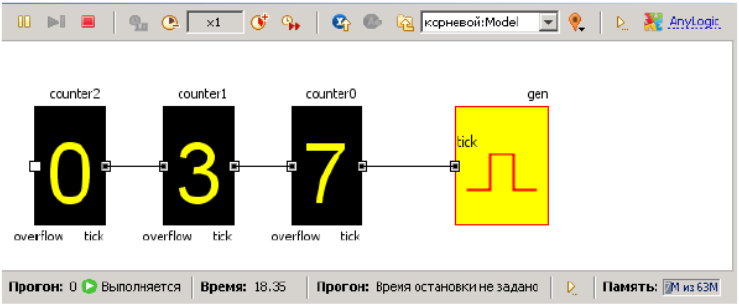


Рис. 31. Модель «умного» счетчика

Для придания созданной модели законченного вида следует добавить элементы описания иинтерактивности. Поместите в мо-

дель описание и слайдер, регулирующий частоту генератора, как показано на рис. 32. Продемонстрируйте свою модель преподавателю.

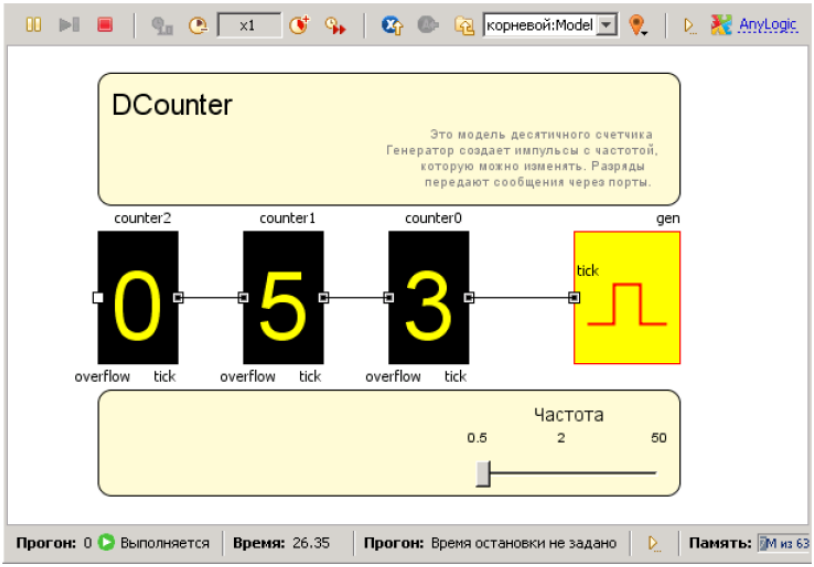


Рис. 32. Готовая модель «умного» счетчика

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

4. В отчете указать все ответы на вопросы.

5. По каждому вопросу (если он не теоретический, а практический по модели, сделать необходимые выводы и отразить их в отчете).

### *Вопросы для самоконтроля*

1. Измените модель таким образом, чтобы счетчик начинал счет с произвольного числа.

2. Добавьте в модель переменную, которая будет принимать значения, показываемые счетчиком.

3. Измените модель таким образом, чтобы счетчик работал на убывание с определенного числа.

4. Измените презентацию модели таким образом, чтобы четные цифры показывались одним цветом, а нечетные – другим.

5. Измените модель таким образом, чтобы параллельно основному счетчику работал второй счетчик, считающий каждый седьмой импульс.

6. Измените презентацию модели таким образом, чтобы цифры мигали с частотой, которую можно изменять слайдером.

7. Измените модель таким образом, чтобы цвет фона при значениях разряда счетчика = 0 был красным, а при остальных – синим.

8. Измените модель таким образом, чтобы параллельно основному счетчику работал второй счетчик, считающий каждый четный импульс.

9. Создайте модель 16-ричного счетчика, работающего параллельно с десятичным.

10. Измените модель таким образом, чтобы параллельно основному – десятичному счетчику, работал двоичный.

11. Измените модель таким образом, чтобы графическое изображение импульса генератора мигало синхронно генерации «тиков».

12. Измените модель таким образом, чтобы счетчик работал на убывание.

13. Измените модель таким образом, чтобы генератор создавал «тики» в случайные моменты времени.

14. Измените презентацию модели таким образом, чтобы каждая цифра счетчика показывалась своим цветом.

15. Дайте определение дискретно-событийной системы, приведите примеры.

16. Охарактеризуйте все типы событий, реализованных в AnyLogic.

17. Что такое порт, и какие элементы AnyLogic могут иметь порт?

18. Добавьте еще один разряд к счетчику.

19. Добавьте второй счетчик, работающий вместе с первым от одного генератора.

20. Какие еще модели IoT можно создавать в программе AnyLogic?

### *Порядок устного отчета о результатах выполненной работы преподавателю*

1. Выполнить работу по предложенному выше заданию.

2. По результатам работы оформить отчет согласно правилам представленным выше.

3. Пояснить, какие сложности возникли при выполнении данной лабораторной работы.

# Лабораторная работа № 14
# МОДЕЛИРОВАНИЕ В ПРОГРАММЕ CUPCARBON

CupCarbon – это симулятор беспроводной сенсорной сети «Умный город» и Интернета вещей (SCI-WSN). Его целью является разработка, визуализация, отладка и валидация распределенных алгоритмов мониторинга, сбора экологических данных и т.д., а также создание экологических сценариев, таких как пожары, газ, мобильные телефоны, и в целом в рамках образовательных и научных проектов. Это может не только помочь наглядно объяснить основные концепции сенсорных сетей и то, как они работают; это также может помочь ученым протестировать их беспроводные топологии, протоколы и т.д.

CupCarbon предлагает две среды моделирования. Первая среда моделирования позволяет разрабатывать сценарии мобильности и генерировать природные явления, такие как пожары и газообразование, а также имитировать мобильные устройства, такие как транспортные средства и летающие объекты (например, беспилотные летательные аппараты, насекомых и т.д.). Вторая среда моделирования представляет собой дискретное событийное моделирование беспроводных сенсорных сетей, которое учитывает сценарий, разработанный на основе первой среды [6].

Сети могут быть спроектированы и прототипированы с помощью эргономичного и простого в использовании интерфейса с использованием платформы OpenStreetMap (OSM) для развертывания датчиков непосредственно на карте. Он включает в себя скрипт под названием Sensscript, который позволяет программировать и настраивать каждый сенсорный узел индивидуально. С по-

мощью этого скрипта также можно генерировать коды для аппаратных платформ, таких как Arduino/XBee. Эта часть не полностью реализована в CupCarbon, она позволяет генерировать коды для простых сетей и алгоритмов.

Моделирование CupCarbon основано на прикладном уровне узлов. Это делает его реальным дополнением к существующим симуляторам. Он не имитирует все уровни протокола из-за сложной природы городских сетей, которым необходимо включать другую сложную и ресурсоемкую информацию, такую как здания, дороги, мобильность, сигналы и т.д.

CupCarbon предлагает возможность моделировать алгоритмы и сценарии в несколько этапов. Например, может быть этап для определения представляющих интерес узлов, за которым следует этап, связанный с характером связи между этими узлами для выполнения заданной задачи, такой как обнаружение события, и, наконец, этап, описывающий характер маршрутизации к базовой станции в случае, если это обнаружено событие. Разные версии CupCarbon позволяют динамически настраивать узлы, чтобы иметь возможность разделять узлы на отдельные сети или присоединяться к разным сетям, задача, которая основана на сетевых адресах и канале. Потребление энергии может быть рассчитано и отображено в зависимости от моделируемого времени. Это позволяет уточнить структуру, осуществимость и реалистичную реализацию сети до ее реального развертывания. Видимость распространения и модели помех интегрированы и включают в себя протоколы ZigBee, LoRa и Wi-Fi.

CupCarbon представляет собой основное ядро проекта ANR PERSEPTEUR, целью которого является разработка алгоритмов для точного моделирования распространения и интерференции сигналов в трехмерной городской среде.

Симулятор CupCarbon можно запустить следующими способами:

1. С помощью ярлыка на рабочем столе операционной системы компьютера.

2. С помощью командной строки. Для этого:

– запустите командную строку окно и перейдите в каталог, в котором находится файл «cupcarbon.jar».

– выполните следующую команду:

java – jar cupcarbon.jar

Попробуйте запустить программу каждым из указанных способов. Графический пользовательский интерфейс CupCarbon состоит из двух частей: главное окно (рис. 33) и окно консоли (рис. 34). В свою очередь главное окно состоит из следующих компонентов:

1. Карта (в центре)
2. Строка меню (вверху)
3. Панель инструментов (под меню)
4. Меню параметров (слева)
5. Панель состояния (внизу)

Карта является основным объектом CupCarbon. Это та часть, где могут быть спроектированы сеть и объекты проекта. Карта может быть изменена в соответствии с предпочтениями пользователя или способом представления информации. Есть 12 возможных карт или фонов, а именно:

(a) Светлый OpenStreetMap
(b) Темный OpenStreetMap
(c) Штриховой фон
(d) Белая сетка с мелкими ячейками фона
(e) Мелкоячеистая черная сетка фона
(f) Черный фон
(g) Белый фон
(h) Средний фон серых клеток

(i) Фон ноутбука

(j) Средний синий фон клетки

(к) карта Google

Карта Google (спутниковая) и черно-белая (черно-белая) карта OpenStreetMap.

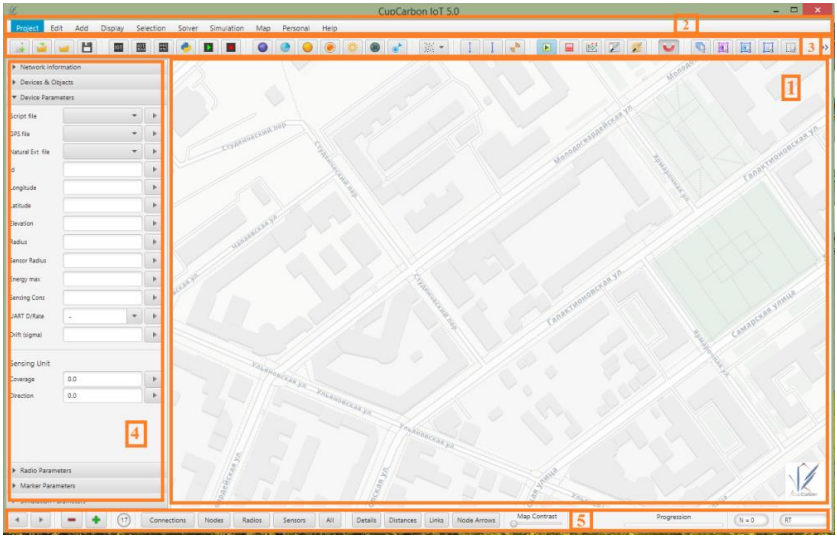


Рис. 33. Главное окно CupCarbon


Рис. 34. Окно консоли CupCarbon

В данной лабораторной работе изучим узел датчика, который отправит сообщение, запрашивающее у конкретного узла датчика в сети его координаты.

В этом примере мы увидим, что нет необходимости в процессе обнаружения соседей и сохранении таблицы соседей локально.

Это всего лишь пример, а не рекомендуемый протокол маршрутизации, потому что в реальной сети выполнение этого процесса для каждого действия (запрос координат) действительно будет очень энергозатратным. Обратите внимание, что моделирование CupCarbon и сценарий Sun выполняются на уровне приложения.

Создайте в программе CupCarbon новый проект и добавьте 100 сенсорных узлов случайным образом (рис. 35).

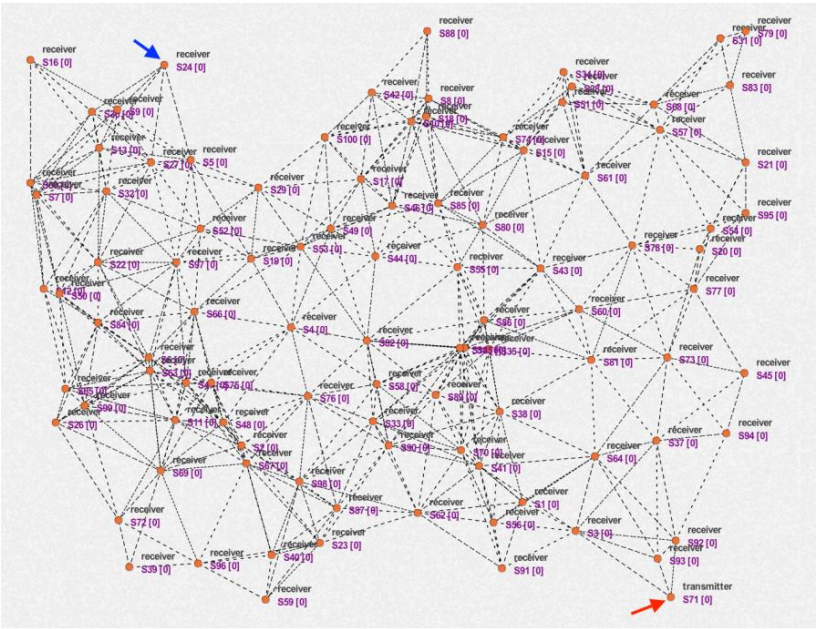


Рис. 35. Сенсорная сеть

Чтобы получить эту визуализацию, нажмите на кнопку Подключение на панели состояния. Давайте возьмем любой узел датчика, например S71, начальный узел, расположенный в крайнем правом нижнем углу сети, как показано стрелкой на рис. 35.

Этот сенсорный узел запросит координаты сенсорного конечного узла S24, расположенного в левом верхнем углу сети, как показано синей стрелкой на рис. 29. Эти идентификаторы (24 и 71) должны быть адаптированы и изменены в вашем примере. Затем замените эти значения в вашем скрипте значениями из вашего примера.

Во-первых, мы начнем со скрипта, который позволяет просто обнаружить и пометить соответствующий узел датчика (т.е. S24).

Чтобы сделать это, S71 отправит в широковещательной передаче сообщение, сформированное A и 24, которое означает «Я ищу узел датчика, имеющий идентификатор 24». Другие сенсорные узлы, которые получат это сообщение, сделают то же самое один раз, если они не являются сенсорным узлом S24. В противном случае они будут помечены.

Сценарий узла (S71) приведен следующим образом:

```
atget id id
data d id "A" 24
send d
loop
stop
```

Сценарий работы приемников (других сенсорных узлов) приведен следующим образом:

```
atget id id
set recA 0
loop
receive m
```

```
rdata m rid type info
if((type=="A") && (recA==0))
set recA 1
if(info==id)
mark 1
else
data d id "A" info
send d
end
end
```

Переменный идентификатор представляет собой текущий идентификатор датчика. Переменная RecA означает «уже получено сообщение A», чтобы отправить сообщение A один раз или чтобы оно было помечено один раз, если его идентификатор является тем, который исследуется (здесь, 24). Информация, представленная здесь 24, которая является исследуемым идентификатором.

Мы называем это информацией, потому что на следующем шаге эта информация станет координатами сенсорного узла 24, которые будут отправлены на сенсорный узел S71. Необходимо моделировать со скоростью моделирования = 0 и скоростью стрелки моделирования = 500. Моделирование даст следующий результат (рис. 36, рис. 37).

Теперь отмеченный узел датчика отобразит свои координаты и отправит их в виде сообщения B предыдущему узлу датчика (здесь, S5), который отправил ему сообщение A. Этот предыдущий узел датчика (S5) снова отправит B к своему предыдущему сенсорному узлу, и так далее до достижения начальных сенсорных узлов 71. Этот последний будет отмечен и отобразит полученные координаты. Предыдущие сценарии будут выполнены следующим образом.

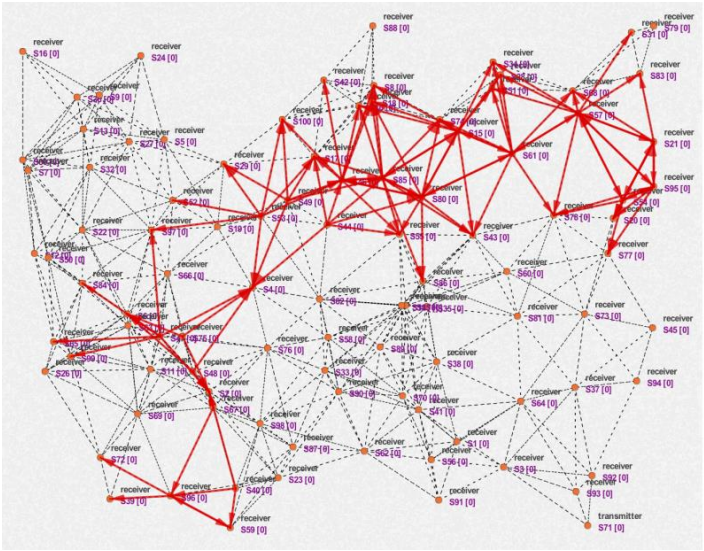

Рис. 36. Во время моделирования

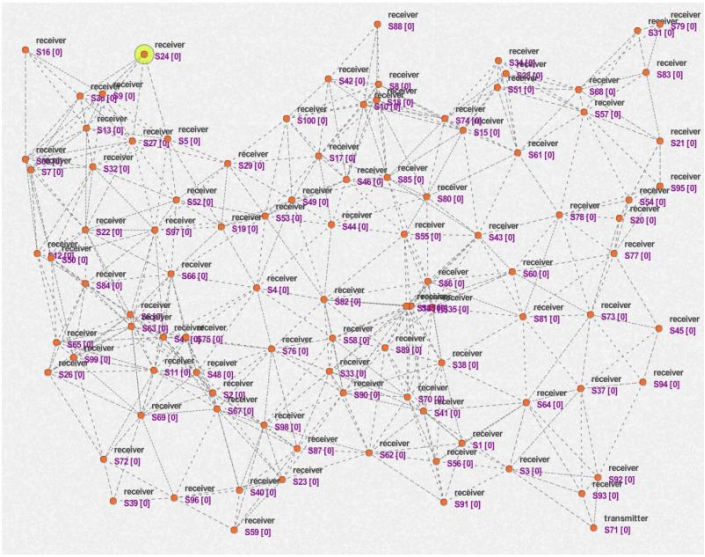

Рис. 37. Завершение моделирования

Сценарий стартового узла (S71) будет выглядеть следующим образом:

```
atget id id
data d id "A"
24 send d
loop
receive m
rdata m rid type x y
if (type=="B")
mark 1
print x " " y
stop
end
```

Сценарий работы других сенсорных узлов будет выглядеть следующим образом:

```
getpos p
atget id id
set recA 0
loop
receive m
rdata m rid type info info2
if((type=="A") && (recA==0))
edge 1 rid
set recA 1
if(info==id)
mark 1
print p
data d id "B" p
send d rid
else
set prev rid
```

```
data d id "A" info
send d
end
end
if(type=="B")
mark 1
data d id "B" info info2
send d prev
end
```

Моделирование даст следующий результат (рис. 38).

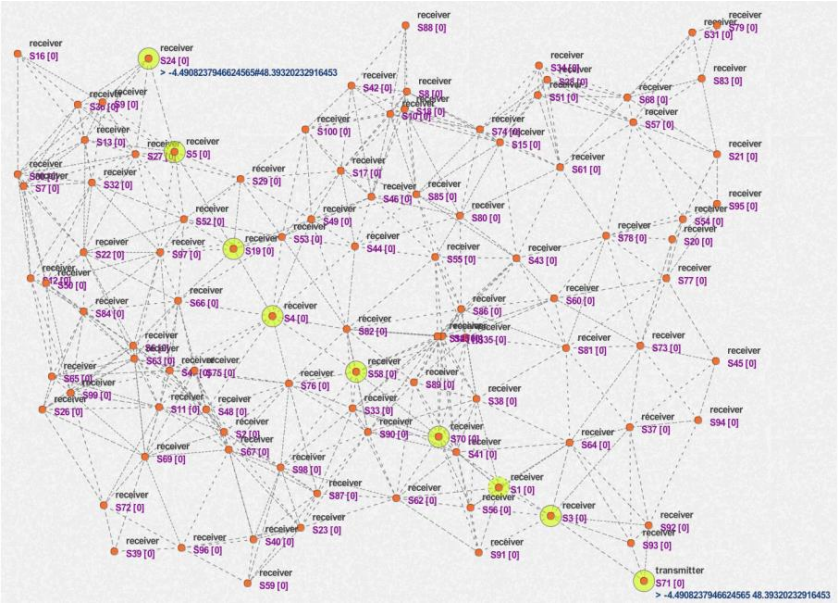


Рис. 38. Результат моделирования

Варианты заданий указаны в табл. 9.

Таблица 9. **Варианты заданий**

| Параметр / Номер варианта (номер студента в списке) | Начальный узел | Конечный узел |
|---|---|---|
| 1 | S16 | S93 |
| 2 | S7 | S37 |
| 3 | S32 | S94 |
| 4 | S22 | S45 |
| 5 | S44 | S1 |
| 6 | S4 | S3 |
| 7 | S76 | S91 |
| 8 | S69 | S93 |
| 9 | S72 | S37 |
| 10 | S39 | S94 |
| 11 | S16 | S45 |
| 12 | S7 | S1 |
| 13 | S32 | S3 |
| 14 | S22 | S91 |
| 15 | S44 | S93 |
| 16 | S4 | S37 |
| 17 | S76 | S94 |
| 18 | S69 | S45 |
| 19 | S72 | S1 |
| 20 | S39 | S3 |
| 21 | S16 | S91 |
| 22 | S7 | S93 |
| 23 | S32 | S37 |
| 24 | S22 | S94 |
| 25 | S44 | S45 |
| 26 | S4 | S1 |

### *Требования оформления отчета по лабораторной работе*

Отчет по выполненной лабораторной работе должен содержать:

1. Титульный лист с указанием названия лабораторной работы, ФИО студента, дату выполнения.

2. Кратко описать суть проделанной работы: что было задано, что конкретно было выполнено.

3. Описание использованных в программах библиотек, назначение использованных функций, переменных и т.д.

4. Скриншот готового проекта в программе CupCarbon.

### *Вопросы для самоконтроля*

1. Какие цели и задачи ставились в данной лабораторной работе?

2. Поясните, что можно моделировать в программе CupCarbon?

3. Что такое «умный» город? Можно ли его реализовать в программе CupCarbon?

4. На какой части решений IoT сосредоточена данная лабораторная работа?

5. Что конкретно было реализовано в данной лабораторной работе?

### *Порядок устного отчета о результатах выполненной работы преподавателю*

1. Выполнить работу по предложенному выше заданию.

2. По результатам работы оформить отчет согласно правилам, представленным выше.

3. Пояснить, какие сложности возникли при выполнении данной лабораторной работы.

# СПИСОК ЛИТЕРАТУРА

1. Росляков, А.В. Интернет вещей: учебное пособие. / А.В. Росляков, С.В. Ваняшин, А.Ю. Гребешков. – Самара: ПГУТИ, 2015. – 285 с.

2. Программа Tinkercad [сайт] / URL: https://www.tinkercad.com/ (дата обращения 15.07.2023).

3. Блум, Джереми. Изучаем Arduino / Джереми Блум; пер. с англ. – Санкт-Петербург: БХВ-Петербург, 2015. – 336 с.

4. Simon Monk. Программируем Arduino. Профессиональная работа со скетчами. – Санкт-Петербург: Питер, 2017. – 252 с.

5. Осоргин, А.Е. AnyLogic 6. Лабораторный практикум / А.Е. Осоргин. – Изд. 2-е, перераб. и доп. – Самара: ПГК, 2012. – 110 с.

6. Wander, A.S. Energy analysis of public-key cryptography for wireless sensor networks /. A.S. Wander, N. Gura, H. Eberle, V. Gupta, S.C. Shantz // Third IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE. – 2005. – pp. 324–328.

## Приложение

## ПРОГРАММНЫЕ КОДЫ К ЛАБОРАТОРНЫМ РАБОТАМ

### Код для счетчика (лабораторная работа № 5)

```
#include <Wire.h>
#include <LiquidCrystal.h>
#include <IRremote.h>
#include <EEPROM.h>
#include <LiquidCrystal.h>
int x = 0;
int currentPin = 1; //Assign CT input to pin 1
double kilos = 0;
int peakPower = 0;
unsigned long startMillis;
unsigned long endMillis;
LiquidCrystal lcd(8, 9, 4, 5, 6, 7); // Assign LCD screen pins, as per LCD
shield requirements
void setup()
{
 Wire.begin();
 lcd.begin(16,2); // columns, rows. use 16,2 for a 16x2 LCD, etc.
 lcd.clear();
 lcd.setCursor(0,0); // set cursor to column 0, row 0 (the first row)
 lcd.print("Arduino");
 lcd.setCursor(0,1); // set cursor to column 0, row 1 (the second row)
 lcd.print("Energy Meter");
 startMillis = millis();
}
void loop()
{
 int current = 0;
 int maxCurrent = 0;
 int minCurrent = 1000;
```

```
 for (int i=0 ; i<=50; i++) //Monitors and logs the current input for 200 cycles to determine max and min current
 {
 current = analogRead(currentPin); //Reads current input and records maximum and minimum current
 if(current >= maxCurrent)
 maxCurrent = current;
 else if(current <= minCurrent)
 minCurrent = current;
 }
 if (maxCurrent <= 517)
 {
 maxCurrent = 516;
 }
 double RMSCurrent = ((maxCurrent – 516)*0.707)/11.8337; //Calculates RMS current based on maximum value
 int RMSPower = 220*RMSCurrent; //Calculates RMS Power Assuming Voltage 220VAC, change to 110VAC accordingly
 if (RMSPower > peakPower)
 {
 peakPower = RMSPower;
 }
 endMillis = millis();
 unsigned long time = endMillis – startMillis;
 kilos = kilos + ((double)RMSPower * ((double)time/60/60/1000000)); //Calculate kilowatt hours used
 startMillis = millis();
 delay (2000);
 lcd.clear();
 lcd.setCursor(0,0); // Displays all current data
 lcd.print(RMSCurrent);
 lcd.print("A");
 lcd.setCursor(10,0);
 lcd.print(RMSPower);
 lcd.print("W");
```

```
 lcd.setCursor(0,1);
 lcd.print(kilos);
 lcd.print("kWh");
 lcd.setCursor(10,1);
 lcd.print(peakPower);
 lcd.print("W");
}
```

**Подключение двух плат Arduino (лабораторная работа № 6)**

```
// Код для Основной платы
#include <Wire.h> // библиотека I2C
int x;
void setup()
{
Wire.begin();
Serial.begin(115200);
}
void loop()
{
 x++;
 Wire.beginTransmission(9); // Цифра 9 – адрес ведомой платы
 Wire.write(x); // Передает значение x
 Wire.endTransmission();
 Serial.println(x);
 delay(1000);
}
// Код для ведомой платы
#include <Wire.h>
int x;
void setup() {
Wire.begin(9); // 9 здесь адрес Slave (упоминается также в коде основной платы)
Wire.onReceive(receiveEvent);
 Serial.begin(115200);
}
```

```
void receiveEvent(int bytes) {
 x = Wire.read(); // Получаем значения x от основной платы
 Serial.println(x);
}
void loop() {
}
```

### Код датчика температуры (лабораторная работа № 7)

```
// C++ code
//
int baselineTemp = 0;
int celsius = 0;
int fahrenheit = 0;
void setup()
{
pinMode(A0, INPUT);
 Serial.begin(9600);
 pinMode(2, OUTPUT);
 pinMode(3, OUTPUT);
 pinMode(4, OUTPUT);
}
void loop()
{
 // set threshold temperature to activate LEDs
 baselineTemp = 40;
 // measure temperature in Celsius
 celsius = map(((analogRead(A0) – 20) * 3.04), 0, 1023, -40, 125);
 // convert to Fahrenheit
 fahrenheit = ((celsius * 9) / 5 + 32);
 Serial.print(celsius);
 Serial.print(" C, ");
 Serial.print(fahrenheit);
 Serial.println(" F");
 if (celsius < baselineTemp) {
 digitalWrite(2, LOW);
```

```
 digitalWrite(3, LOW);
 digitalWrite(4, LOW);
 }
 if (celsius >= baselineTemp && celsius < baselineTemp + 10) {
 digitalWrite(2, HIGH);
 digitalWrite(3, LOW);
 digitalWrite(4, LOW);
 }
 if (celsius >= baselineTemp + 10 && celsius < baselineTemp + 20) {
 digitalWrite(2, HIGH);
 digitalWrite(3, HIGH);
 digitalWrite(4, LOW);
 }
 if (celsius >= baselineTemp + 20 && celsius < baselineTemp + 30) {
 digitalWrite(2, HIGH);
 digitalWrite(3, HIGH);
 digitalWrite(4, HIGH);
 }
 if (celsius >= baselineTemp + 30) {
 digitalWrite(2, HIGH);
 digitalWrite(3, HIGH);
 digitalWrite(4, HIGH);
 }
 delay(1000); // Wait for 1000 millisecond(s)
}
```

**Код датчика движения (лабораторная работа № 8)**

```
// C++ code
//
int sensorState = 0;
void setup()
{
 pinMode(2, INPUT);
 pinMode(LED_BUILTIN, OUTPUT);
 Serial.begin(9600);
```

```
}
void loop()
{
 // read the state of the sensor/digital input
 sensorState = digitalRead(2);
 // check if sensor pin is HIGH. if it is, set the
 // LED on.
 if (sensorState == HIGH) {
 digitalWrite(LED_BUILTIN, HIGH);
 Serial.println("Sensor activated!");
 } else {
 digitalWrite(LED_BUILTIN, LOW);
 }
 delay(10); // Delay a little bit to improve simulation performance
 }
```

### Код для «умного» дома (лабораторная работа № 9)

```
/*данный умный дом имеет возможность управлять яркостью света люстры, выводить температуру комнаты на дисплей и включать вентиляцию при высокой температуре */
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
int t;
int l;
byte degree[8] =
{
B00111,
B00101,
B00111,
B00000,
B00000,
B00000,
B00000,
};
```

```
void setup() {
 lcd.createChar(1, degree);
 lcd.begin(16, 2);
 lcd.print("Temp = ");
 lcd.setCursor(0, 1);
 lcd.print("Ventilation");
 lcd.setCursor(10, 0);
 lcd.print("\1C");
 pinMode(13, OUTPUT);
 pinMode(6, OUTPUT);
}
void loop() {
 t = 0.488 * analogRead(A0) – 49.76;
 l = analogRead(A1);
 l = map(l, 0, 1023, 0, 255);
 analogWrite(13, l);
 lcd.setCursor(7, 0);
 lcd.print(t);
 if (t <= 25){
 lcd.clear();
 lcd.setCursor(0, 0);
 lcd.print("Temp = ");
 lcd.setCursor(10, 0);
 lcd.print("\1C");
 lcd.setCursor(0, 1);
 lcd.print("Ventilation");
 lcd.setCursor(12, 1);
 lcd.print("OFF");
 digitalWrite(6, 0);
 }else{
 lcd.clear();
 lcd.setCursor(0, 0);
 lcd.print("Temp = ");
 lcd.setCursor(10, 0);
 lcd.print("\1C");
```

```
 lcd.setCursor(0, 1);
 lcd.print("Ventilation");
 lcd.setCursor(12, 1);
 lcd.print("ON");
 digitalWrite(6, 1);
 }
}
```

**Код для «умного» сада (лабораторная работа № 10)**

```
#include <Servo.h>
#include <LiquidCrystal.h>
//input
const int button_l = A1;
const int button_r = A3;
const int button_u = A2;
const int button_d = A4;
const int temp_in = A0;
const int light_in = A5;
//output
Servo servo1;
const int pinservo = 3;
const int pump = 2;
const int lamp = 10;
LiquidCrystal lcd(12, 11, 7, 6, 5, 4);
//символ градуса
byte dgr[8] =
{
 B01110,
 B10001,
 B10001,
 B01110,
 B00000,
 B00000,
 B00000,
```

```
 B00000
};
//страница
int mode = 0;
int pgmode0 = 0; // страница на mode 0
//время и температура
int h=12;
int m = 0;
int s = 0;
int d_t = 0;
float temp = 0;
int temp_out = 0;
int flag = 0;
int TIME = 0;
//таймер для насоса
int h_on=12;
int m_on = 0;
int h_off=12;
int m_off = 0;
int flag2 = 0;
int flag3 = 0;
int state_wtr = 0;
int state_time = 0;
int state2 = 1;
//температура для насоса
int temp_on = 30;
int state_temp = 0;
//кнопки
int state_l = 0;
int state_r = 0;
int state_u = 0;
int state_d = 0;
void setup()
{
 Serial.begin(9600);
```

```
 pinMode(button_l, INPUT);
 pinMode(button_r, INPUT);
 pinMode(button_u, INPUT);
 pinMode(button_d, INPUT);
 pinMode(temp_in, INPUT);
 pinMode(light_in, INPUT);
 lcd.begin(16, 2);
 lcd.createChar(1,dgr);
 pinMode(pinservo, OUTPUT);
 pinMode(pump, OUTPUT);
 pinMode(lamp, OUTPUT);
 servo1.attach(pinservo);
}
void loop()
{
 //проверка условий работы помпы
 if(state_wtr == 1){
 digitalWrite(pump,HIGH);
 servo1.write(90);
 }
 else{
 digitalWrite(pump,LOW);
 servo1.write(0);
 }
 //Отсчёт часов
 s=s+1;
 if(s==60){
 s=0;
 m=m+1;
 }
 if(m==60)
 {
 m=0;
 h=h+1;
 flag=flag+1;
```

```
}
if(h==13)
{
h=1;
}
//Данные температуры
d_t=analogRead(temp_in);
temp=d_t *0.004882814;
temp=(temp-0.5)*100;
temp_out = round(temp);
//Проверка кнопок на смену страницы
state_r=digitalRead(button_r);
if(digitalRead(button_r)==1){
mode++;
}
state_l=digitalRead(button_l);
if(digitalRead(button_l)==1){
mode--;
}
if(mode<0){
mode = 4;
}
if(mode>4){
mode = 0;
}
//вывод экрана и настройка времени, температуры на страницах
if(mode == 0){
pgmode0=pgmode0+1;
if(pgmode0==10){
pgmode0=0;
}
if(pgmode0<7){
lcd.setCursor(0,0);
lcd.print("TIME|");
lcd.print(h);
```

```
lcd.print(":");
lcd.print(m);
lcd.print(":");
lcd.print(s);
if(flag<12)lcd.print("AM");
if(flag==12)lcd.print("PM");
if(flag>12)lcd.print("PM");
if(flag==24)flag=0;
lcd.setCursor(0,1);
lcd.print("TEMP|");
lcd.print(temp_out);
lcd.write(1);
lcd.print("C");
}
else if(pgmode0>=7){
lcd.setCursor(0,0);
lcd.print("Lighting ");
if(analogRead(light_in)<900){
lcd.print("ON");
}
else if(analogRead(light_in)>=900){
lcd.print("OFF");
}
lcd.setCursor(0,1);
lcd.print("Watering ");
if(state_wtr == 1){
lcd.print("ON");
}
else if(state_wtr == 0){
lcd.print("OFF");
}
}
lcd.setCursor(13,1);
lcd.print("M:");
lcd.print(mode);
```

```
delay(1000);
lcd.clear();
}
else if(mode == 1){
lcd.setCursor(0,0);
lcd.print("TIME|");
lcd.print(h);
lcd.print(":");
lcd.print(m);
lcd.print(":");
lcd.print(s);
if(flag<12)lcd.print("AM");
if(flag==12)lcd.print("PM");
if(flag>12)lcd.print("PM");
if(flag==24)flag=0;
lcd.setCursor(0,1);
lcd.print("Set time");
lcd.setCursor(13,1);
lcd.print("M:");
lcd.print(mode);
delay(1000);
lcd.clear();
state_u = digitalRead(button_u);
if(state_u==1)
{
h=h+1;
flag=flag+1;
if(flag==24)flag=0;
if(h==13)h=1;
}
state_d = digitalRead(button_d);
if(state_d==1){
s=0;
m=m+1;
}
```

```
}
else if(mode == 2){
lcd.setCursor(0,0);
lcd.print("TIME|");
lcd.print(h_on);
lcd.print(":");
lcd.print(m_on);
if(flag2<12)lcd.print("AM");
if(flag2==12)lcd.print("PM");
if(flag2>12)lcd.print("PM");
if(flag2==24)flag2=0;
lcd.setCursor(0,1);
lcd.print("Set turn on");
lcd.setCursor(13,1);
lcd.print("M:");
lcd.print(mode);
delay(1000);
lcd.clear();
state_u = digitalRead(button_u);
if(state_u==1)
{
h_on=h_on+1;
flag2=flag2+1;
if(flag2==24)flag2=0;
if(h_on==13)h_on=1;
}
state_d = digitalRead(button_d);
if(state_d==1){
m_on=m_on+1;
}
}
else if(mode == 3){
lcd.setCursor(0,0);
lcd.print("TIME|");
lcd.print(h_off);
```

```
lcd.print(":");
lcd.print(m_off);
if(flag3<12)lcd.print("AM");
if(flag3==12)lcd.print("PM");
if(flag3>12)lcd.print("PM");
if(flag3==24)flag3=0;
lcd.setCursor(0,1);
lcd.print("Set turn off");
lcd.setCursor(13,1);
lcd.print("M:");
lcd.print(mode);
delay(1000);
lcd.clear();
state_u = digitalRead(button_u);
if(state_u==1)
{
h_off=h_off+1;
flag3=flag3+1;
if(flag3==24)flag3=0;
if(h_off==13)h_off=1;
}
state_d = digitalRead(button_d);
if(state_d==1){
m_off=m_off+1;
}
}
else if(mode == 4){
        lcd.setCursor(0,0);
lcd.print("TEMP|");
lcd.print(temp_on);
lcd.write(1);
lcd.print("C");
lcd.setCursor(0,1);
lcd.print("Set temp on");
lcd.setCursor(13,1);
```

```
lcd.print("M:");
lcd.print(mode);
delay(1000);
lcd.clear();
state_u = digitalRead(button_u);
if(state_u==1)
{
temp_on++;
}
state_d = digitalRead(button_d);
if(state_d==1){
temp_on--;
}
}
//проверка условий времени для работы полива
if(h==h_off && m==m_off){
state_time = 0;
}
else if(h==h_on && m==m_on){
state_time = 1;
}
//проверка условий температуры для работы полива
if(temp_out >= temp_on){
state_temp = 1;
}
else if(temp_out < temp_on){
state_temp = 0;
}
//Включение или выключение полива
if(state_time == 0 && state_temp == 1){
state_wtr = 1;
}
else if(state_time == 0 && state_temp == 0){
state_wtr = 0;
}
```

```
 else if(state_time == 1 && state_temp == 0){
 state_wtr = 1;
 }
 else if(state_time == 1 && state_temp == 1){
 state_wtr = 1;
 }
 //Работа освещения
 if(analogRead(light_in)<900){
 digitalWrite(lamp,HIGH);
 }
 else if(analogRead(light_in)>=900){
 digitalWrite(lamp,LOW);
 }
}
```

**Код для «умного» города (лабораторная работа № 11)**

```
// C++ code
//
int inches = 0;
int cm = 0;
int q = 0;
long readUltrasonicDistance(int triggerPin, int echoPin)
{
 pinMode(triggerPin, OUTPUT); // Clear the trigger
 digitalWrite(triggerPin, LOW);
 delayMicroseconds(2);
 // Sets the trigger pin to HIGH state for 10 microseconds
 digitalWrite(triggerPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(triggerPin, LOW);
 pinMode(echoPin, INPUT);
 // Reads the echo pin, and returns the sound wave travel time in microseconds
 return pulseIn(echoPin, HIGH);
}
```

```
void setup()
{
 pinMode(4, OUTPUT);
}
void loop()
{
 q = 0.01723 * readUltrasonicDistance(3, 2);
 if (q < 100) {
 digitalWrite(4, LOW);
 } else {
 digitalWrite(4, HIGH);
 }
 delay(10); // Delay a little bit to improve simulation performance
}
```

**Код для «умного» гаража (лабораторная работа № 12)**

```
#include <LiquidCrystal.h>// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(1, 2, 4, 5, 6, 7);
#include <Servo.h> //includes the servo library
Servo myservo1;
int ir1 = 8;
int ir2 = 9;
int Total = 6;
int Space;
int x = 0;
int y = 0;
void setup() {
pinMode(ir1, INPUT);
pinMode(ir2, INPUT);
myservo1.attach(3);
myservo1.write(100);
lcd.begin(16,2);
lcd.setCursor (0,0);
```

```
lcd.print(" Car Parking ");
lcd.setCursor (0,1);
lcd.print(" System ");
delay (2000);
lcd.clear();
Space = Total;
}
void loop(){
if( digitalRead(ir1)==HIGH && x==0 ){
if(Space>0){
 x=1;
 if(y==0){
 myservo1.write(0);
 Space = Space-1;
 }
 }else{
 lcd.setCursor (0,0);
 lcd.print(" Sorry no Space ");
 lcd.setCursor (0,1);
 lcd.print(" Available ");
 delay(3000);
 lcd.clear();
 }
 }
 if( digitalRead(ir2)==HIGH && y==0 ){
 y=1;
 if(x==0){
 myservo1.write(0);
 Space = Space+1;
 }
 }
 if(x==1 && y==1){
 delay (1000);
 myservo1.write(100);
 x=0, y=0;
```

```
 }
lcd.setCursor (0,0);
lcd.print("Total Space: ");
lcd.print(Total);
lcd.setCursor (0,1);
lcd.print("Have Space: ");
lcd.print(Space);
}
```

Учебное издание

***Глушак Елена Владимировна,***
***Куприянов Александр Викторович***

# ВВЕДЕНИЕ В ИНТЕРНЕТ ВЕЩЕЙ
## (лабораторные работы)

*Практикум*

Редакционно-издательская обработка
издательства Самарского университета

Подписано в печать 25.12.2023. Формат 60x84 1/16.
Бумага офсетная. Печ. л. 7,75.
Тираж 120 экз. (1-й з-д 1-27). Заказ № . Арт. – 8(Р2ДПР)/2023.

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)
443086, Самара, Московское шоссе, 34.

---

Издательство Самарского университета.
443086, Самара, Московское шоссе, 34.