Лекция 9. Разработка программного обеспечения для IoT-приложений

Цель лекции – ознакомить обучающихся с основами разработки программного обеспечения для IoT-приложений.

Введение

Для разработки программного обеспечения IoT необходимы пять уровней:
1. Сбор и консолидация;
2. Подключение;
3. Сбор и сборка;
4. Управление и анализ;
5. Приложения и службы.

В этой лекции рассмотрим пять уровней и необходимое программное обеспечение. Основное внимание уделяется методам разработки программного обеспечения, которые используются на уровнях устройств IoT, шлюзов, подключения к Интернету, а также веб- и облачных приложений.

Основные определения программирования

Компонент – это абстракция для основного набора фреймворков, сервисов или функций программного обеспечения, которые могут быть повторно использованы после их перенастройки для предоставления решений.

Фреймворк программного обеспечения – это абстракция, которая предоставляет повторно используемую программную среду и программное обеспечение с общими функциями. Фреймворк помогает пользователю выборочно изменять и добавлять новые функции при повторном использовании функций, которые создают новые приложения, сервисы, продукты и решения. Например, предположим, что фреймворк предоставляет общие функции для графики и рисования объектов. Разработчик приложения для рисования выбирает и повторно использует функции фреймворка, выбирая функции рисования и визуализации, такие как эллипс или круг, и добавляя требуемые параметры эллипса или круга, такие как размер, положение, цвет, затенение и узоры заливки. Приложение запускается и рисует объект в соответствии с параметрами и использует общие функции, предоставленные в фреймворке для рисования. Аналогию можно провести из фреймворка для изготовления свечей, который используется в свечной промышленности. Помещение нитей в соответствующие отверстия и заполнение расплавленным воском делает связку свечей. Фреймворк приложений – это фреймворк, который поддерживает фундаментальную структуру и предоставляет скелет, который ведет к разработке приложений, и предоставляет повторно используемую среду разработки с общими функциями. Фреймворк помогает пользователю выборочно изменять, добавлять новые функции и создавать приложение.

Фреймворк веб-приложений – это фреймворк, который поддерживает фундаментальную структуру и скелет, что ведет к разработке веб-приложений, включая веб-API и ресурсы. Фреймворк предоставляет повторно используемую среду разработки веб-приложений с общими функциями. Фреймворк помогает пользователю фреймворка выборочно изменять и добавлять новые функции и создавать веб-приложения или веб-сервисы и API.

Сообщество – это инициатива или модель разработки программного обеспечения, которая использует коллективные усилия для разработки программного обеспечения. Сообщество формируется по инициативе университетов, организаций и учреждений для проекта с открытым исходным кодом. Сообщество — это утвержденный независимый фонд с открытым исходным кодом для распространения, такой как OSGi. Фонд или сообщество

также могут обладать авторскими правами и обязуются следовать практикам распространения, соглашениям с участниками и лицензированию, например, Apache Foundation для сервера Apache. OSGi (Open Service Gateway Initiative) — это фреймворк разработки для Java, который используется для развертывания модульных программ, библиотек и пакетов. Пакет OSGi относится к тесно связанной коллекции классов, jar-файлов и файлов конфигурации. Пакет динамически загружается. Внешние зависимости явно объявляются, если они существуют.

Программный стек – это набор фреймворков и служб, который является минимально необходимым для предполагаемого полного решения. Примерами являются Eclipse IoT Stack для IoT и Berkeley Data Analytics Stack (BDAS) для аналитических решений. Стек может поддерживаться сообществом. Например, Open Services Gateway Initiative (OSGi) поддерживает Eclipse IoT Stack. Стек предоставляет набор программных функций, фреймворков, пакетов, модулей или подсистем. Стек создает полную платформу для поддержки приложений или служб при установке и настройке на отдельных системах или добавлении в шаблоны для автоматической установки. Sandbox-сервер — это сервер с данными из исходных кодов и других коллекций содержимого, данных или наборов кодов, частных или общедоступных, и защищен от преднамеренных или непреднамеренных изменений. Сервер функционирует как рабочий каталог, тестовый сервер или сервер разработки. Sandbox также используется для тестирования разработанного приложения. Sandbox-клиент использует минимальные функциональные возможности сервера, те же переменные среды или доступ к идентичной базе данных сервера для разработки определенных функциональных возможностей и приложений.

Linux-дистрибутив означает пакет или набор программных функций и модулей Linux, объединенных вместе для определенных функций или для определенного оборудования и распространяемых для более широкого использования и приложений.

Bootloader (загрузчик) – это системное программное обеспечение, которое загружается или встраивается в микросхему микроконтроллера или вычислительную платформу, чтобы позволить системе запустить свои функции.

OS – это системное программное обеспечение. Оно управляет процессами, памятью, вводом-выводом, сетевыми подсистемами и устройствами. Оно обеспечивает приоритезацию, многозадачность, одновременный запуск нескольких потоков и многих системных функций с использованием данного вычислительного оборудования устройства. ОС может быть с открытым исходным кодом, например, Linux или его дистрибутив. Дистрибутив ОС на основе Linux – это Debian Squeeze Arch Linux Fedora для Raspberry Pi или BeagleBone с использованием процессора ARM.

Многозадачность или многопоточность означает выполнение нескольких задач или потоков в соответствии с некоторым планом, например, активные или незаблокированные задачи запускаются в соответствии с последовательностью или одна за другой в выделенные временные интервалы или в соответствии с приоритетами, назначенными для каждой или запущенной первой вызванной задачей. ОС контролирует выполнение задач и обновления.

Операционная система реального времени (RTOS) также является системным программным обеспечением и представляет собой ОС, которая позволяет запускать процессы реального времени на вычислительном и коммуникационном оборудовании.

Скрипт – это фрагмент кода в текстовой форме, который запускается с использованием интерпретатора, который интерпретирует их во время выполнения. Примерами языков сценариев являются JavaScript, JSON, Perl и PHP.

C – это язык программирования высокого уровня, используемый для кодирования встроенных платформ. Коды компилируют и генерируют исполняемые коды для этой платформы или компьютера. C — это процедурно-ориентированный язык.

Графические пользовательские интерфейсы (GUI) относятся к средствам отображения на экране компьютера, таким как строка состояния, панель задач, кнопки,

флажки, меню и диалоговые окна, которые обеспечивают простое взаимодействие API между пользователем и приложением, приложением или программным обеспечением службы.

Программирование встраиваемых устройств на платформе Arduino с использованием IDE

Для разработки прототипов программ требуются загрузчик, ОС и IDE. Программное обеспечение встраивается в платформу устройства.

Первый уровень в архитектурной концепции IoT – сбор (данных с устройств/датчиков) и консолидация (обогащение). Второй уровень – подключение к Интернету. IDE позволяет разрабатывать программное обеспечение для функций на первом и втором уровнях. IDE также может позволять использование функций ОС или RTOS на встроенном устройстве.

Прошивка загрузчика хранится во флэш-памяти/ПЗУ микроконтроллера в устройстве и обеспечивает связь с компьютером, имеющим IDE. IDE, как правило, состоит из API, библиотек, компиляторов, RTOS, симулятора, редактора, ассемблера, отладчика, эмуляторов, логического анализатора, средства записи кода и другого программного обеспечения для комплексной разработки системы. IDE может быть с открытым исходным кодом. Например, Arduino имеет IDE с открытым исходным кодом, которую можно загрузить с веб-сайта Arduino.

IDE позволяет разрабатывать коды на компьютере, а затем загружать (проталкивать) коды на встроенное устройство, например, Arduino или плату микроконтроллера. Средство записи кода помещает коды во флэш-память или EEROM. Таким образом, конкретные коды приложений встраиваются в устройство.

Плата Arduino может быть запрограммирована с помощью инструментов avr-gcc. Плата Arduino имеет предустановленный загрузчик, встроенный в прошивку. Программатор Arduino разрабатывает коды с помощью графической кроссплатформенной IDE. Arduino обеспечивает простоту. IDE платы Arduino также отличается простотой, основана на языке обработки и упрощает программирование. Плата подключается к компьютеру, на котором работает IDE. Программа загрузчика передает управление и позволяет запустить загрузчик, который загружает требуемые функции ОС и программное обеспечение в системное оборудование и сетевые возможности платы.

Загрузчик Arduino обеспечивает многозадачность с помощью функций обработки прерываний (аналогично обработке событий) для каждой задачи. Многозадачность выполняется путем назначения нескольких значений числа n для задач ($n > 0$). Когда выполняется инструкция для прерывания, например, INT n, то вызывается функция обработки прерываний n для выполнения. Каждая задача или поток может иметь связанный с ней номер n. Функция обработки прерываний, похожая на callback(n), выполняется при событии n или похожая на функцию catch при исключении n.

IDE состоит из набора программных модулей, которые предоставляют программную и аппаратную среду для разработки и прототипирования программного обеспечения для конкретной платформы устройства. Сначала компьютер загружает соответствующую версию IDE в соответствии с ОС компьютера. Компьютер обычно работает под управлением Windows или Mac OS X или Linux.

Загрузчик позволяет компьютеру загружать разработанные коды на плату с помощью Arduino IDE через кабель USB или помеченный последовательный порт. Загрузчику Arduino не нужно инициировать загрузку ОС, как это делается на компьютере, где загрузчик загружает ОС со вторичного диска.

Arduino IDE доступна на веб-сайте Arduino. Программист загружает требуемую версию IDE. IDE работает на компьютере и позволяет разрабатывать коды, их симуляцию и загружать (встраивать) на платформу устройства.

Arduino IDE включает библиотеку C/C++. Библиотека называется Wiring по одноименному проекту с модулем с открытым исходным кодом на веб-сайте. Функции библиотеки Wiring упрощают кодирование для операций ввода-вывода Arduino.

Разработка кодов

Arduino IDE функционирует как редактор файлов для кодов, использующих среду обработки и библиотечные функции. Редактор обеспечивает автоматический отступ, выделяет синтаксис кодов и сопоставляет фигурные скобки. Отредактированный файл компилируется, проверяет и выводит список ошибок, если ошибок нет, позволяет вставлять коды для встраивания на плату через последовательный или USB-порт.

Простота Arduino очевидна из того факта, что для определения исполняемых программных функций для платы необходимы всего две функции, а именно setup() и loop(). Функция setup() запускается при запуске и используется для инициализации настроек, а функция loop() имеет программу в бесконечном цикле с использованием оператора «while (true) {statements ;}», который выполняется до выключения питания.

Последовательный монитор в IDE позволяет выводить сообщения из встроенного программного обеспечения для микроконтроллера на экран компьютера, на котором настроена IDE. Сообщения требуются во время тестирования и отладки загруженного программного обеспечения на этапах тестирования.

Эмулятор и отладка программного обеспечения

Эмуляция означает выполнение действий, аналогичных реальным. Программное обеспечение эмулятора эмулирует работу платформы встроенного устройства, известной как целевая плата. Программное обеспечение эмулирует платформу встроенного устройства на компьютере. Разработчик наблюдает за действиями на компьютере.

Внутрисхемная эмуляция (ICE) означает отладку оборудования путем эмуляции после подключения целевой платы к компьютеру. Компьютер устанавливает точки останова во встроенном программном обеспечении. Результаты в конце каждой точки останова позволяют программисту найти ошибку (ошибочный код или раздел кода). Компьютер может инициировать пошаговое выполнение кодов. На экране отображаются регистры в микропроцессоре или микроконтроллере, значения переменных в конце каждого шага или в конце указанного раздела кодов. Дисплей компьютера также отображает содержимое адресов памяти в каждой точке останова.

Когда на плате используется встроенный Linux, разработчик может использовать утилиту, называемую gdb (отладчик GNU), исходный код которой открыт на веб-сайте GNU. Подключение целевой платы осуществляется через последовательный или USB-порт компьютера. Ethernet также используется для платформ устройств, которые поддерживают сетевое взаимодействие. Стандарт JTAG (Joint Test Action Group) позволяет тестировать целевую плату на компьютере, когда программное обеспечение JTAG встраивается в оборудование, а разъем JTAG подключает оборудование к компьютеру.

Считывание с датчиков и устройств: использование аналогового входа АЦП

Предположим, что датчик температуры используется для измерения от 0 до 100°C. Датчик отправляет аналоговый выходной сигнал на аналоговый вход 10-битного АЦП. Выходной сигнал АЦП преобразуется в последовательный с помощью преобразователя параллельного входа в последовательный выход (PISO). Последовательный выход подключается к последовательному входному контакту SPI на плате Arduino Uno. Датчик RH% также может использоваться аналогичным образом, когда измеренное значение находится в RH% вместо °C.

Выходной сигнал АЦП для датчика при 100 градусах – это десятичное 1023 (=двоичное 1111111111) и десятичный 0 (=0000000000) для 0°.

Использование таймеров

Функции таймера требуются в ряде приложений. Доступно несколько библиотек таймеров. Набор библиотек функций таймера доступен в сети MsTimer обозначает миллисекундные таймеры с двумя состояниями. Он содержит простое использование. Он имеет две функции set() и start(). Первая устанавливает таймер для прерывания после заданного интервала, а вторая запускает работу таймера.

Использование библиотеки последовательного интерфейса программного обеспечения

Библиотека последовательного интерфейса имеет функции чтения и записи в соответствии с последовательными протоколами. Протокольная связь сначала передает биты заголовка. Они могут включать адрес устройства или регистров (управляющие, данные, статус или другие регистры). Биты данных передаются после битов заголовка и битов управления, если таковые имеются. Конечные биты передаются в конце. Каждый последовательный протокол имеет определенные способы форматирования и упорядочивания во время связи.

Связь на основе протокола UART использует два сигнала, обозначаемые Tx или TxD (для последовательной передачи заголовка, данных и других бит) и Rx или RxD (для последовательного приема). Предположим, что скорость передачи составляет 2400. Затем за одну секунду передается 240 раз новый набор 8-битных данных. Для каждого набора данных, символа, команды или адреса передается 10 бит. Байт представляет символ в строке, считанные данные и команду для принимающего устройства или адрес целевого или принимающего устройства. Плата Arduino имеет контакты 0 (Rx) и 1 (Tx) для последовательной связи UART. Компьютер подключается к плате с помощью этих контактов. Программная последовательная библиотека имеет функции, которые считывают и записывают последовательные данные с помощью платы Arduino. Ввод-вывод данных может быть на двух контактах цифровых контактов ввода-вывода, отличных от 0 и 1, а именно, контакт 2 и контакт 3 как Rx и Tx соответственно.

Функции UART в программной последовательной библиотеке устанавливают режимы контактов как Tx и Rx. Связь инициирует контакты RFID Rx и Tx на RFID IC, когда контакты RFID подключаются к цифровым контактам ввода-вывода Tx и Rx соответственно. IC TX становится низким с высокого, когда связь инициируется с Arduino. RFID IC обнаруживает это и сначала отправляет заголовок, который представляет собой предустановленный адрес устройства. Последовательный интерфейс Arduino считывает заголовок перед принятием идентификатора метки. ID состоит из 10 символов, и IC отправляет конечный символ из RFID в конце. Конечный символ ASCII-кода равен 13. Программная последовательная библиотека обеспечивает передачу данных с помощью простого использования последовательных функций чтения и записи. Пример 9.5 поясняет последовательное чтение RFID-метки с использованием UART с помощью библиотечных функций.

Обзор других видов встраиваемых устройств для IoT

Intel Galileo IDE

Intel Galileo IDE для Windows можно загрузить с веб-сайта Galileo. Объединенные возможности Intel Galileo и Arduino:

● Сходство IDE с Arduino: Galileo добавляет плату Arduino X86, а новая IDE также позволяет обновлять прошивку платы.

● Совместимость: совместимо с шилдами Arduino и аналогичными разъемами ICSP, последовательным портом, 14 цифровыми контактами ввода-вывода и 6 аналоговыми входами.

● Библиотека Ethernet: совместимо по удобству использования с библиотекой Ethernet Arduino и может подключаться с помощью HTTP-соединения, используя стандартный пример WebClient.

● Карты PCI Express Mini: могут подключаться к картам GSM, Bluetooth, WiFi с помощью библиотеки WiFi Arduino.

● Порт USB-хоста: USB – это выделенный порт, который можно использовать с библиотекой Arduino USB-хоста и подключением клавиатуры или мыши к компьютерам. ● Поддержка TWI/I2C, SPI, последовательного подключения и MicroSD: Совместимость со стандартными библиотеками Arduino.

● Дистрибутив Linux, работающий на плате: всего 8 МБ флэш-памяти позволяют использовать node.js (для веб-проектов), звуковые инструменты: ALSA, видеоинструменты: V4L2, Python, SSH, компьютерное зрение openCV.

Raspberry Pi IDE

Raspberry Pi (RPi) представляет собой вычислительную систему размером со смарт-карту с медиа-сопроцессорами. Разработчик может работать со стандартной клавиатурой и дисплеем. Pi поддерживает ряд языков, включая Python. Доступные библиотеки PyPi предназначены для программирования RPi с использованием Python. RPi поддерживает ОС Raspbian Ubuntu, дистрибутив Linux (также известный как Snappy), и имеет функции безопасного запуска автономных машин, устройств M2M и IoT.

Для разработки вычислительных устройств с медиа-возможностями доступно несколько IDE. Ниже приведены популярные из них:

AdafruitIDE – это веб-IDE. IDE предоставляет веб-среду разработки. RPi подключается к локальной сети, и кодирование может выполняться с использованием языков Python, JavaScript или Ruby. Adafruit также предоставляет настраиваемую ОС вместо Raspbian, которая работает с клавиатурой и отображает монитор, подключенный к RPi. Защищенная оболочка (SSH) для криптографического сетевого протокола для прикладного уровня обеспечивает удаленный вход в систему и удаленные подключения систем, клавиатуры и монитора.

BlueJIDE – это IDE на основе Java в BlueJIDE. BlueJIDE работает с помощью Java Development Kit, а также запускает программы Java, включая Java 8 на RPi с использованием библиотеки PI4J. Исходный код Ninja-IDE — это IDE, которая означает not-just-another IDE.

Ninja позволяет использовать файловые функции, а также создавать приложения на Python. Он работает в кроссплатформенной среде, RPi, а также Linux, Windows и Mac OSX. Его возможности:

● Возможность создания плагинов, тем самым расширяя функциональность редактора кода, помощь в разработке проекта и загрузка виджетов нескольких плагинов для многоцелевого использования.

● Редактирование кода на нескольких языках, помощник по коду для автодополнения кода, навигация по коду, веб-навигация и импорт из любой точки мира, контекстное меню во вкладках для выполнения быстрых действий, переходы по коду, реализация точек останова и навигации по закладкам, менеджер профилей, поиск ошибок кода и обнаружение файлов статических ошибок.

● Локатор кода для быстрого и прямого доступа к классу, функции или файлу с всплывающим окном над текстовыми полями.

● Добавлены экспортер символов, поиск файлов и их использования, веб-инспектор и virtualenv, которые можно указать при создании проекта из Ninja IDE или для существующего проекта в свойствах проекта.

● Поддерживает использование консоли для встроенного Python. Консоль автоматически управляет проектом Python. Она позволяет пользователю выполнять задачи, связанные с управлением файлами, в самой IDE и сохраняет описание файлов проекта.

Контрольные вопросы:

1. Сколько и какие уровни необходимы для разработки программного обеспечения IoT необходимы?
2. Что такое фрейворк программного обеспечения?
3. Объясни принципы программирования микроконтроллеров Arduino.
4. Какие виды встраиваемых устройств кроме Arduino вы знаете?