Вместо введения

Данный мануал не описывает теоретическую часть iOS CoreBluetooth и GATT протокола и принципы работы Bluetooth девайсов, он предназначен для практической части, в нем описана настройка среды разработки Arduino для установки различных примеров на плату ESP32 для создания Bluetooth-сервера, протокол общения для каждого из примеров, нюансы работы с core Bluetooth. Данный мануал подкреплен тестовым примером для iPhone написанным на swift 5.2 и два примера на С++ для ESP32 под Arduino.

Протокол общения

В данных примерах протокол общения следующий:

Peripheral name – ESP32

Периферал содержит 2 сервиса и по 1 характеристике в каждом из сервисов, подробнее в таблице:

|  |  |  |  |
| --- | --- | --- | --- |
| **Service UUID** | **Characteristic UUID** | **Сapability** | **Description** |
| 6e400001-b5a3-f393-e0a9-e50e24dcca9e | 6e400003-b5a3-f393-e0a9-e50e24dcca9e | PROPERTY\_WRITE | Input:  -on  -off |
| 6e400002-b5a3-f393-e0a9-e50e24dcca9e | 6e400004-b5a3-f393-e0a9-e50e24dcca9e | PROPERTY\_READ  PROPERTY\_NOTIFY | Output:  -on  -of  -Wrong data |

Итак, описание – у периферала есть сервис, у сервиса есть характеристика которая доступна только на запись и принимает на вход текст «on» или «off», второй сервис также имеет характеристику но она доступна только на чтение и она оповещает об изменении своего значения, ее выходные данные «on», «off» или «Wrong data».

Пример 1

Данный пример эмитирует работу iPhone с Bluetooth-выключателем (включение освещения, открытие машины), это действие не занимает длительного времени на выполнение отправленной команды. Так после подключения к девайсу и отправки соответствующей команды («on»/«off») или неверных данных, модуль моментально вернет результат ее выполнения (для того что бы это произошло, нужно подписаться на уведомления характеристики чтения), таким образом при включении или выключение освещения в доме модуль будет возвращать текущий статус и девайс сможет реагировать на изменение статуса и менять иконки, показывать анимацию или выполнять следующее действие. Также при отключении от девайса и повторном подключении можно прочитать характеристику чтения и узнать текущий статус освещения и показать актуально состояние юзеру. Также при мульти-подключении к девайсу (выключателю) при изменении одним пользователем состояния – все получат новое состояние автоматически. В примере после отправки команды на включение или выключение на плате будет включаться/выключаться светодиод.

Пример 2

Данный пример эмитирует работу iPhone с Bluetooth девайсом которое часто изменяет свои данные без отправки команд на него, например считывание показаний температуры и влажности в доме и при их изменении уведомление девайса о новых данных, данный пример при отправке команды «on» начинает на случайный период времени включать и выключать светодиод при изменении статуса светодиода он уведомляет слушателей о том что статус изменен на on или off. При отправке команды «off» генерация приостанавливается. Таким образом этот пример эмитирует, как и сказалось в самом начале датчик температуры в доме, пользователь открывает приложение, оно находит девайс и подключается к нему, подписывается на уведомления и ждет от девайса(датчика) обновленные данные, после изменения показаний температуры в доме датчик уведомит моментально о том что данные изменились а приложение сможет всегда отображать юзеру актуальные данные.

Пофиксить ардуинку чтоб вначале был сразу верный статус светодиода

Проверить мульти-подключение к esp

Дописать в туториал полезные фичи – настройка ардуино среды и проблемы с ней, и че делать если не находим нужный девайс с нужным именем, что некоторые девайсы ограничены по mtu на 20 байт

Расшарить поиск девайсов по UUID

Расшарить стрим на другой девайс (запись большого кол-ва байт)

Добавить пример с длительным действием

Добавить пример с подключением с паролем