IOTS BLE Central / Peripheral

Jonas Arnold, Simon Frei

November 22, 2021

Abstract

Auf dem Arduino MKR 1010 WiFi soll ein BLE Central und ein BLE Peripheral implementiert werden. Das Central soll vom Peripheral Daten lesen, ausserdem soll es mit einer custom Characteristic eine LED ansteuern können.

1 Implementierung BLE Central / Peripheral

1.1 Impelemtierung BLE Peripheral

Die Implementierung des Peripherals lief problemlos ab. Die Spezifikationen des vordefinierten BLE Services **Environmental Sensing**. haben wir auf der Spezifikationenseite von Bluetooth gefunden. Die vordefinierten UUIDs für den Service und die Characteristics waren ebenfalls bei bluetooth.com zu finden.

Damit das ENV Board nicht zu oft abgefragt wird, wird die funktion **millis()** von Arduino verwendet und nur alle 1000 ms neue Sensordaten geschickt. Die LED kann auch schneller geschaltet werden (diese Characteristic wird im Loop ausserhalb dieser IF Abfrage bedient):

```
// update sensors every SENSOR_UPDATE_TIME_MS to not stress the ENV board
if((millis() - last_update) > SENSOR_UPDATE_TIME_MS)
{
int16_t temperature = ENV.readTemperature();
int16_t humidity    = ENV.readHumidity();
int16_t illuminance = ENV.readIlluminance();

temperatureCharacteristic.writeValue(temperature);
humidityCharacteristic.writeValue(humidity);
illuminanceCharacteristic.writeValue(illuminance);

last_update = millis();  // update time
}
```

Listing 1: Limitierung der Sensordatenabfragen

Bezüglich der Datentypen haben wir folgendes gelernt: Der Datentyp ist nicht an eine vordefinierte Characteristic verknüpft wie das in anderen Standards der Fall ist. Die Sensordaten würden bestenfalls als *float* übermittelt (Datentyp der Characteristic *BLEFloatCharacteristic*). Leider kann das die verwendete App **LightBlue** nicht richtig anzeigen (nur den HEX Wert davon). Deshalb haben wir schlussendlich die Werte als Integer übermittelt (Datentyp der Characteristic *BLEIntCharacteristic*). Dies kann die App richtig anzeigen, vorausgesetzt das Datenformat wird auf **Unsigned Little-Endian** eingestellt (Übermittlungsreihenfolge der einzelnen Bytes).

1.2 Impelemtierung BLE Central

Die Implementierung für den Central gab es zuerst einige Probleme. Mit den Apps die verwendet wurden um ein Peripheral zu "simulieren" kamen wir nicht ganz klar. Deshalb liessen wir das Peripheral auf dem einen Arduino Wifi 1010 laufen und testeten auf dem zweiten Arudino Wifi 1010 den Central. Der Central is so aufgebaut, dass er Periphal Geräte sucht und wenn der local Name mit unserem Peripheral übereinstimmt untersucht er die Attribute. Dazu "abonniert" er die gewünschten Charakteristiken:

```
if(!temperatureCharacteristic){
  Serial.println('no temperatureCharacteristic');
  peripheral.disconnect();
  return;
}else if(!temperatureCharacteristic.canSubscribe()){
  Serial.println('cant subscribe temperatureCharacteristic');
  peripheral.disconnect();
  return;
}else if(!temperatureCharacteristic.subscribe()){
  Serial.println("subscription of temperatureCharacteristic failed!");
  peripheral.disconnect();
  return;
}else{
  Serial.println('subscribe temperatureCharacteristic success');
}
```

Listing 2: Abonnieren der Charakteristiken (Temp)

Solange das Gerät verbunden ist, werden die Werte über die Serielle Schnitstelle an den Computer gesendet, wenn ein neuer Wert gesendet wird. Wenn der Helligkeitswert 100 übersteigt, dann schaltet er die LED auf dem Peripheral Device an.

```
int16_t illVal, humVal, tempVal;
    while(peripheral.connected()){
      if(illuminanceCharacteristic.valueUpdated()){
        illuminanceCharacteristic.readValue(illVal);
        Serial.print("illuminance: ");
        Serial.println(illVal);
      }
      if(humidityCharacteristic.valueUpdated()){
        humidityCharacteristic.readValue(humVal);
        Serial.print("humidity: ");
        Serial.println(humVal);
      }
      if(temperatureCharacteristic.valueUpdated()){
        temperatureCharacteristic.readValue(tempVal);
        Serial.print("temp: ");
        Serial.println(tempVal);
      }

      if(illVal >= 100){
        illuminanceState = 1;
      }else{
        illuminanceState = 0;
      }

     if (oldIlluminanceState != illuminanceState) {
      oldIlluminanceState = illuminanceState;

      if (illuminanceState) {
        Serial.println("ill > 100");
        ledCharacteristic.writeValue((byte)0x01);
      } else {
        Serial.println("ill < 100");
        ledCharacteristic.writeValue((byte)0x00);
      }
     }

  }
```

Listing 3: Abonnieren der Charakteristiken (Temp)