Das Arduino Nano ESP32-Board entdecken

BLE und ESP-NOW anwenden, die Bus-Systeme verstehen, MicroPython lernen

Erik Bartmann

bombini verlag

Kommentare und Fragen können Sie gerne an uns richten:

Bombini Verlags GmbH
Richard-Wagner-Str. 11
53757 Sankt Augustin

E-Mail: service@bombini-verlag.de

Bibliografische Information Der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Satz: Bombini Verlag

Belichtung, Druck und buchbinderische Verarbeitung: Print Group Sp. z.o.o., Polen

ISBN 978-3-946496-36-6

Inhalt

Projekt 8: Synthesizer-Steuerung über BLE

Wer Spaß an elektronischer Musik und Sphärenklängen hat, der kennt auch die elektronischen Musikinstrumente, die sich Synthesizer nennen. Auf der folgenden Abbildung ist ein Synthesizer mit integriertem Keyboard und vielen Drehreglern zu sehen.

Abbildung 1: Der Synthesizer MatrixBrute von Arturia

Dies soll keine Einführung in den Umgang mit professionellen Synthesizern werden, sondern das Interesse wecken, sich mit derartigen Instrumenten und Klängen näher zu befassen. Viel ist für den Anfang nicht erforderlich, denn wir wollen in diesem Kapitel die MIDI-Steuerung über das Arduino-Nano-ESP32-Board realisieren.

Was ist MIDI?

MIDI ist die Abkürzung für Musical Instrument Digital Interface und es handelt sich dabei um einen Industriestandard für den Austausch von musikalischen Steuerinformationen zwischen elektronischen Instrumenten, wie zum Beispiel Keyboards oder Synthesizern. Es werden von MIDI keinerlei Audio-Daten versendet oder verarbeitet. Eine MIDI-Information enthält nicht direkt die Musik, wie sie zum Beispiel auf einer CD oder einer MP3-Datei enthalten ist, sondern nur Steueranweisungen, was zu tun ist.

Ich möchte einen freien Synthesizer in Form einer Android-App mit dem Namen SynprezFM rudimentär ansteuern und dafür das Arduino-Nano-ESP32-Board verwenden.

Abbildung 2: Die Synthesizer-App SynprezFM

Die Steuerung wird über BLE erfolgen.

Was ist BLE?

BLE ist die Abkürzung für Bluetooth Low Energy und ist eine Funktechnik, mit der sich Geräte in einer Umgebung von etwa zehn Metern vernetzen lassen.

Ich möchte zur Realisierung kein komplettes Keyboard nachbauen, sondern lediglich einen einzigen Taster mit zwei Potentiometern. Eine Taste ist zur Generierung des Tons erforderlich, das erste Potentiometer steuert die Tonhöhe und das zweite die Anschlaggeschwindigkeit. Recht simpel also, doch es steckt Potential in diesen Anfängen.

Und wer weiß, vielleicht regt den ein oder anderen dieses Kapitel auch dazu an, den Synthesizer auszubauen und noch ganz andere Dinge damit zu veranstalten.

Fangen wir mit dem Arduino-Sketch an, bevor es auf der Smartphone-Seite unter Android weitergeht.

Der Arduino-Sketch

Um die Funktionalität von MIDI über BLE nutzen zu können, ist es erforder-

lich, eine Library hinzuzufügen, die sich ESP32-BLE-MIDI nennt. Wir rufen also wieder den Library Manager auf und geben den markierten Suchbegriff ein.

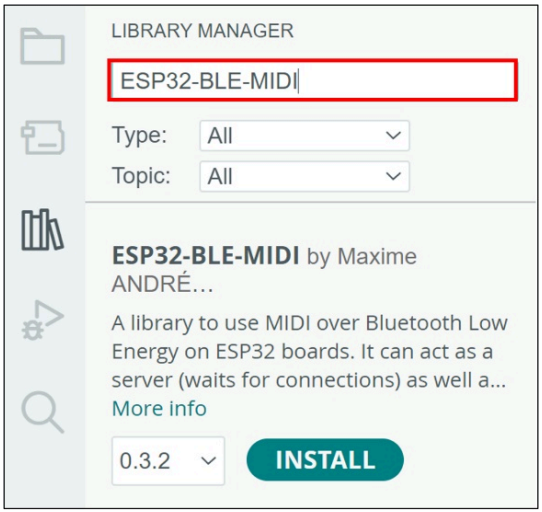

Abbildung 3: Die Installation der ESP32-BLE-MIDI-Library

Nähere Informationen zur Library sind unter dem folgenden Link zu finden.
https://www.arduino.cc/reference/en/libraries/esp32-ble-midi/

Nun können wir starten. In Zeile 1 wird die erforderliche BLE-MIDI-Library eingebunden, die gerade eben installiert wurde. In Zeile 3 wird der Pin definiert, an dem sich der Taster befindet. In den Zeilen 4 und 5 werden Variablen definiert, um den Zustand des Tasters zu überwachen, damit sowohl ein LOW-HIGH- als auch ein HIGH-LOW-Pegelwechsel erkannt werden kann. Die zu spielende Note wird als Integer-Wert in der Variablen midiNote gespeichert, die in Zeile 6 deklariert ist. Die Anschlagstärke wird in der Variablen velocity (Geschwindigkeit) gespeichert und in Zeile 7 deklariert.

```
1  #include <BLEMidi.h>
2
3  int buttonPin    = 2; // Button-Pin-Note
4  int btnState     = 0; // Actual button state
5  int lastBtnState = 0; // Last button state
6  int midiNote;         // MIDI-Note
7  int velocity;         // MIDI-Velocity
```

In der setup-Funktion in Zeile 12 wird der Taster über die pinMode-Funktion

```
 9  void setup() {
10    Serial.begin(115200);
11    Serial.println("Initializing bluetooth");
12    pinMode(buttonPin, INPUT_PULLDOWN);
13    BLEMidiServer.begin("Eriks-ESP32-MIDI-Device");
14    Serial.println("Waiting for connections...");
15  }
```

konfiguriert und in der darauffolgenden Zeile 13 der BLEMidiServer über die begin-Methode gestartet. Über den angegebenen Namen kann das BLE-Device später auf dem Smartphone ausfindig gemacht werden.

In der loop-Funktion kommt es nun in Zeile 18 zur Abfrage, ob der Server verbunden wurde, was über die isConnected-Methode erreicht wird. Bei erfolgreicher Verbindungsaufnahme kommt es zur kontinuierlichen Abfrage des Tasters (Zeile 19) und der beiden Potentiometer (Zeilen 20 und 22). Für die analogen Werte findet wieder ein Mapping über die map-Funktion statt, die wir schon kennengelernt haben. Findet ein Pegelwechsel des Tasters von LOW nach HIGH statt, kommt es zum Aufruf der noteOn-Methode in Zeile 23, mit der Angabe des MIDI-Kanals 0, der MIDI-Note (midiNote) und der Anschlagstärke (velocity). Die Note wird hörbar. Wird die Taste wieder losgelassen, erfolgt ein Pegelwechsel von HIGH nach LOW und die noteOff-Methode in Zeile 26 wird mit den gleichen Werten wie bei noteOn aufgerufen. Die Note verklingt.

```
17  void loop() {
18    if(BLEMidiServer.isConnected()) {
19      btnState = digitalRead(buttonPin);
20      velocity = map(analogRead(A6), 0, 4095, 0, 127);
21      if(btnState == 1 && lastBtnState == 0){ // LOW-HIGH-Level
22        midiNote = map(analogRead(A7), 0, 4095, 10, 90);
23        BLEMidiServer.noteOn(0, midiNote, velocity); // Note on
24        Serial.println("On");
25      } else if(btnState == 0 && lastBtnState == 1){  // HIGH-LOW-Level
26        BLEMidiServer.noteOff(0, midiNote, velocity); // Note off
27        Serial.println("Off");
28      }
29      lastBtnState = btnState; // Save last button state
30    }
31  }
```

Sowohl das Drücken als auch das Loslassen einer Taste löst ein entsprechendes MIDI-Ereignis aus.

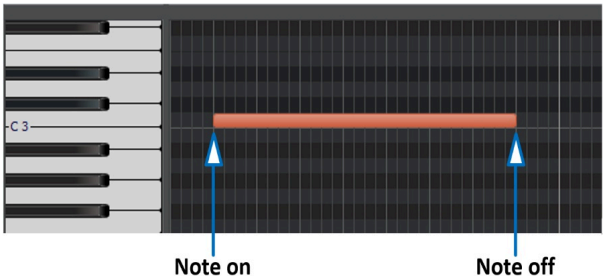

Abbildung 4: Note on und Note off in MIDI

Die Länge der Note wird durch die Länge des grafischen Balkens abgebildet und befindet sich genau zwischen den beiden MIDI-Befehlen Note on und Note off. Nun sind aber noch weitere Informationen erforderlich, denn es muss hinsichtlich der Note noch die Notennummer und die Velocity (Anschlagstärke) übermittelt werden. Hinsichtlich MIDI ist dann noch der Kanal wichtig, denn es gibt beim MIDI-Standard sechzehn verschiedene Kanäle. Eine Nachricht an ein MIDI-Gerät würde dann ungefähr wie folgt lauten: „Spiele auf MIDI-Kanal 0 die Note C3 mit der Länge, die zwischen Note on und Note off liegt sehr laut, also mit einem hohen velocity-Wert.“ Der velocity-Wert kann dabei im Bereich von 0 bis 127 liegen und deckt somit 128 unterschiedliche Werte ab.

Sehen wir uns sowohl den Schaltplan als auch den Schaltungsaufbau an.

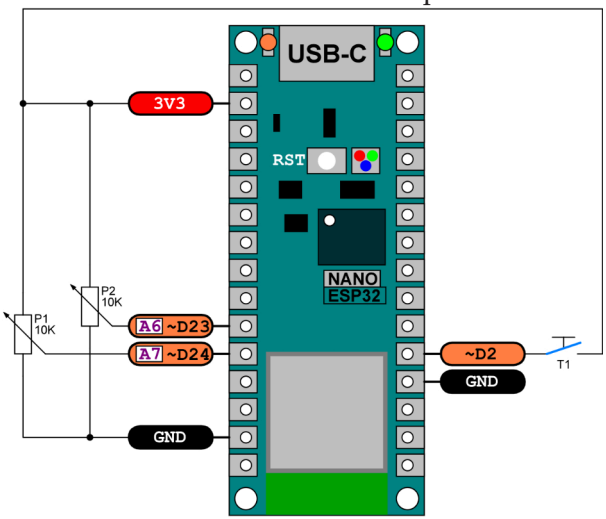

Abbildung 5: Der Schaltplan zur Synth-Ansteuerung

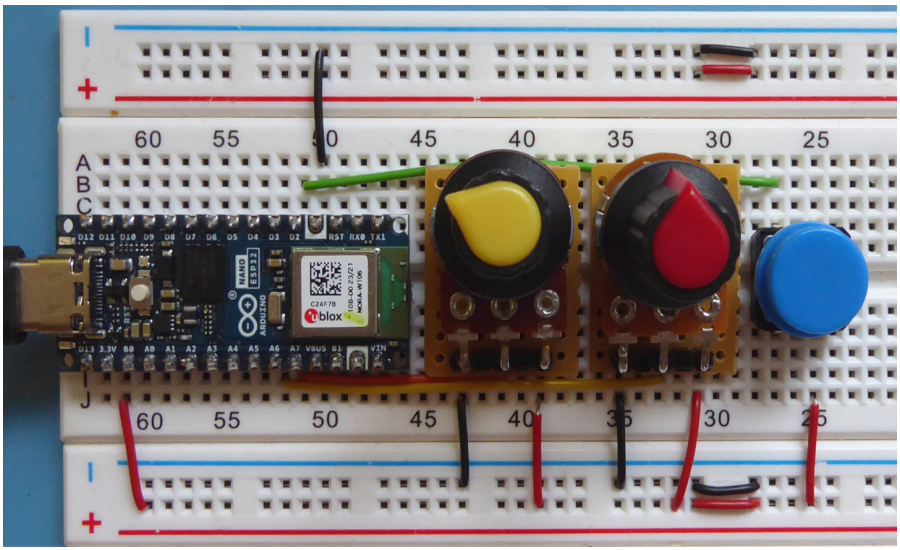

Abbildung 6: Der Schaltungsaufbau zur Synth-Ansteuerung

Jetzt laden wir den Sketch hoch und wenden uns dem Smartphone zu.

Kommen wir zur Android-Seite

Um auf der Android-Seite mit BLE zu arbeiten, muss eine entsprechende App mit Namen MIDI BLE Connect installiert werden. Dazu wird Google Playstore aufgerufen und in das Suchfeld der entsprechende Begriff eingegeben.

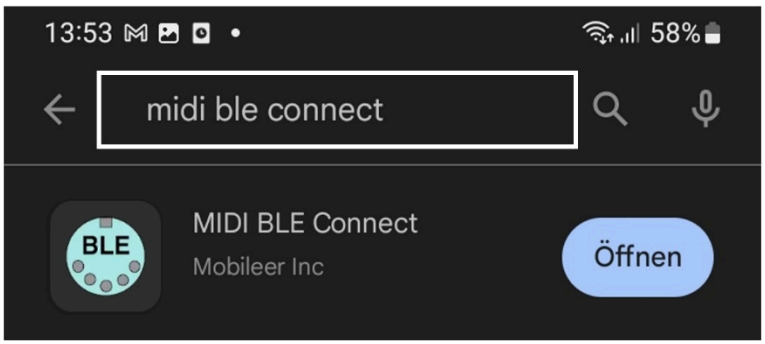

Abbildung 7: Die Installation von MIDI-BLE-Connect

Im Anschluss installieren wir den Synthesizer SynprezFM über das gleiche Verfahren.

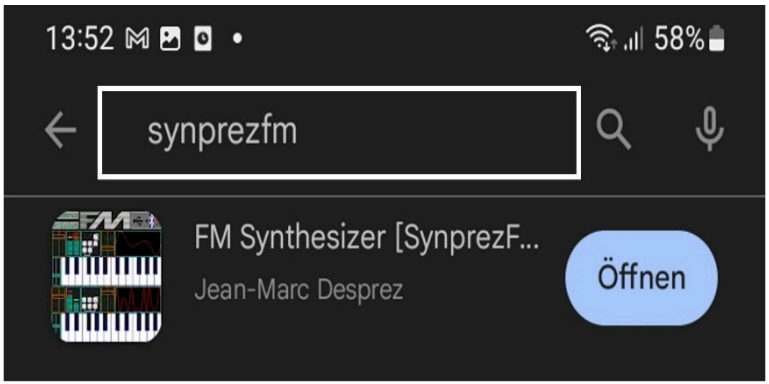

Abbildung 8: Die Installation von SynprezFM

Nach den beiden Installationen starten wir MIDI BLE Connect und stellen vorher sicher, dass auf dem Smartphone Bluetooth aktiviert ist. Nach der Auswahl der START BLUETOOTH SCAN-Schaltfläche sollte das ESP32-BLE-Device zur Auswahl angezeigt werden. Durch die Auswahl mit einem Fingertipp wird es dann wie in der Abbildung dargestellt.

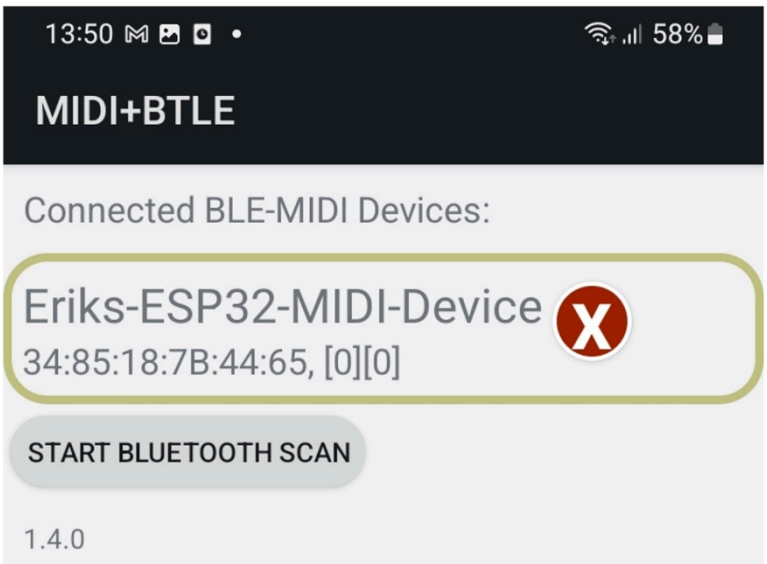

Abbildung 9: Das ESP32-BLE-Device wurde ausgewählt und aktiviert

Nun kann die SynprezFM-App gestartet werden, was mit dem unmittelbaren Erkennen des ESP32-BLE-Devices einhergehen sollte. Die Meldung sollte ungefähr so aussehen in der rot umrandeten Box auf der folgenden Abbildung.

Abbildung 10: Der MIDI-Controller wurde erkannt

Nun kann es losgehen und auf der Arduino-Seite der Taster mit unterschiedlichen Potentiometerstellungen gedrückt werden. Die SynprezFM-App sollte reagieren und die gedrückten Tasten sollten farbig aufleuchten und Töne sollten hörbar sein.