HomeKit Controller

Midihome von Philip Kindermann / Kirill Krasnoshchokov / Alex Lorenz

Bei diesem Instructable handelt es sich um eine Anleitung zum Nachbau des „Midihome“ getauften Apple Homekit Controllers. Der Hintegrundgedanke bei diesem Gerät ist der, dass viele Menschen gerne haptisches Feedback spüren, wenn sie Geräte bedienen. Grade in Zeiten von Apps zur Steuerung immer modernerer und ausgefeilter Smarthome Gegenstände wird dieses Feedback vernachlässigt – hier greift der Midihome Controller ein. Er stellt eine individuell konfigurierbare Steuerung für alle mit Apples Homekit kompatiblen Geräte dar indem er mit Hilfe eines Arduinos über Bluetooth mit einem iPhone / iPad kommuniziert und diese dann über den HomeKit Service Anweisung an die Geräte / Gegenstände weiterleiten. Die Gesamtkosten des Projekts betrugen bei unserem Layout um die 200 Euro.

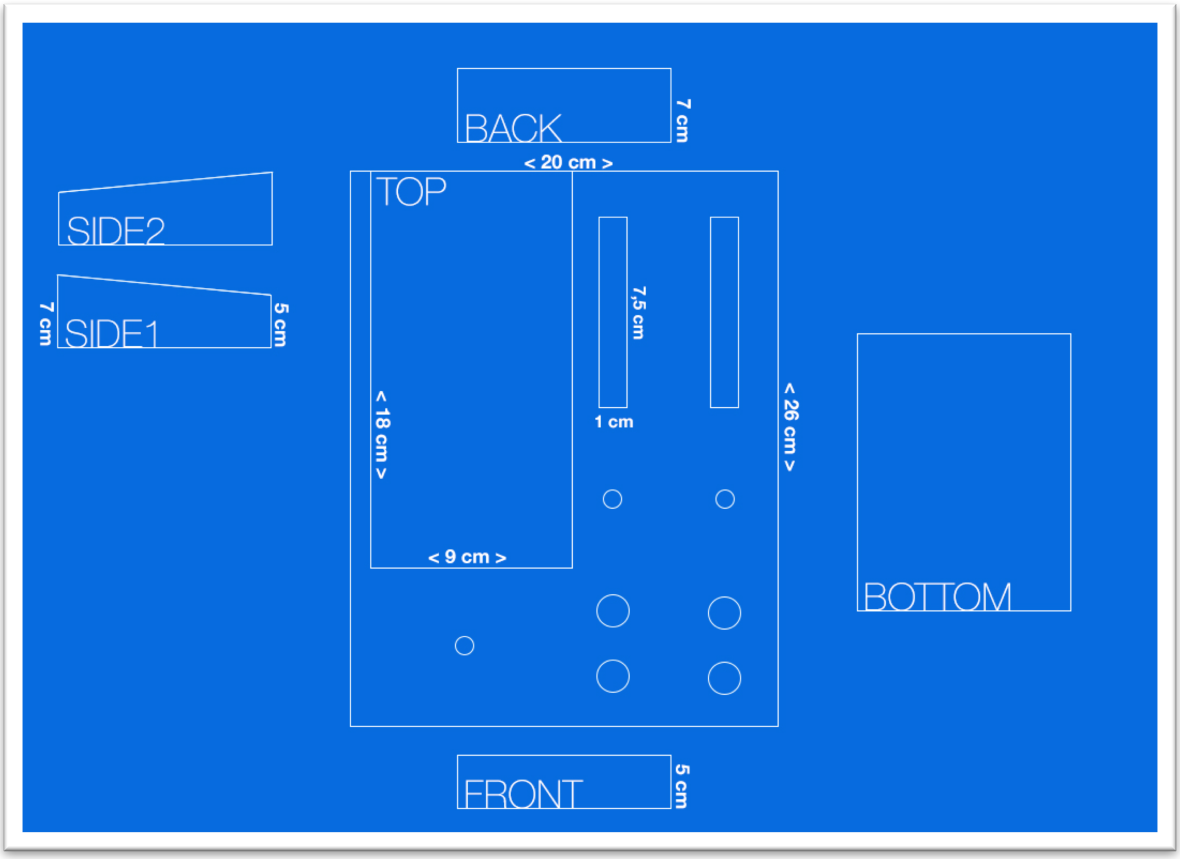
MATERIALIEN

- Arduino Uno / Arduino Mega (je nach Anzahl der gewünschten Buttons/Regler) mit einem Breadboard
- Buttons / Regler (nach Wunsch – begrenzt durch Arduino/Boxgröße)
- Ein Bündel Kupferkabel (optimalerweise 3 in verschiedenen Farben für V / GRND / Out)
- Einen Lötkolben / Lötzinn
- 1m x 1m Sperrholz (beliebige Holzart – abhängig von den geplanten Dimensionen des Controllers)
- Eine Stichsäge (oder Laubsäge – wobei es mit einer Stichsäge deutlich einfacher ist)
- Ein Stück Filz
- Holzgrundierung
- Holzfarbe (beliebig aber auf selber Basis wie Holzgrundierung – Wasser/Acryl)
- Holzleim
- 3 Stücke Schleifpapier (40 / 180 / >400)
- Nägel und einen Hammer
- Sekundenkleber
- Einen Mac bzw. einen Computer mit OSX und Xcode
- iPhone (ab 5)
- Ein Bluetooth Low Energy (BLE 4.0) kompatibles Board für den Arduino (mit einem anderen funktioniert eine Verbindung mit dem iPhone nicht)

DER START (Die Box)

Es sollte von Anfang an ein fester Plan für die gesamte Box erstellt werden. Dazu eignen sich CAD Programme (z.B. Autodesk) als auch Paint / Photoshop. Bei der Konzeption der Box sollten Button / Fader Größen sowie die Dimensionen des Arduinos und des verwendeten Breadboards berücksichtigt werden. Wir werden aus Gründen des Debuggings / der Erweiterung das Breadboard für die Stromzufuhr der Buttons / das Erden verwenden.

Das Ganze kann dann am Beispiel unserer Box so aussehen – ist natürlich aber vollkommen abhängig von den Dimensionen der gewählten Buttons / Fadern / Reglern und ob man eine Ladeschale für das iPhone einplanen möchte und wenn ja für welches (5/6/6Plus).



Nachdem nun ein möglichst genaues Modell / Zeichnung der geplanten Box vorliegt, setzt man sich daran die Einzelteile auf das Holz zu zeichnen und mit der Stichsäge unter akribischer Kontrolle in aller Ruhe auszusägen. Bei den Löchern für die Knöpfe und Co. kann man zu einer Fräse oder einem Bohrer mit einer Laubsäge greifen.

Sind alle Seiten ausgesägt und bereit – testet man noch einmal ob die Bedienelemente alle in die für sie vorgesehenen Öffnungen gleiten. Dabei ist es nicht problematisch, wenn einige nur mit oder ohne viel Druck hineinpassen, da wir sowieso noch lackieren / schleifen werden und die Löcher dann noch mal angepasst werden müssen.

LACKIEREN (Die Grundierung)

Zunächst sollte man alle Teile gründlich mit dem gröbsten Schleifpapier bearbeiten. Besonders die später nach außen sichtbaren Oberflächen sollten dabei mit größter Sorgfalt geschliffen werden. Hierbei können auch noch leichte Fehler (Unebenheiten) die beim vorherigen Sägen entstanden sind korrigiert werden.

Als nächstes sucht man sich eine Unterlage die verschmutzt werden kann (Zeitungspapier oder Karton) und streicht die Einzelteile großzügig mit der Holzgrundierung an. Ob man alle Seiten gleichzeitig oder in Abständen von einigen Stunden nacheinander lackiert wird dabei durch die verwendete Trockenfläche bestimmt. Wir empfehlen im Nachhinein immer nur 1-2 Seiten zu streichen, diese trockenen zulassen und danach mit den anderen weiter zu machen.
Tropfenbildung sollte hier möglichst verhindert werden, ist aber bei der Grundierung nicht weiter schlimm.

Sind die Teile getrocknet, benutzt man das mittlere Schleifpapier und schleift alle Flächen noch mal gründlich ab und verwendet dann das gröbere für die Entfernung von Unebenheiten / Tropfen der Grundierung. Ist das ganze vollbracht, trägt man eine weitere Schicht Grundierung auf, lässt diese Trocken und schleift am Ende mit dem feinsten Schleifpapier alle Flächen so glatt wie nur möglich.

Es sollte beachtet werden, dass je nach Grundierung zwischen 12-24 Stunden vergehen müssen bevor man schleifen und erneut lackieren kann.

LACKIEREN (Die Farbe)

Beim Auftragen der gewählten Farbe kann man sich an der Abfolge der Grundierung orientieren. Es wird eine Schicht Farbe aufgetragen, diese trocknet, wird abgeschliffen und erneut aufgetragen. Für ein optimales Ergebnis sollte das ganze 2-3-mal wiederholt werden. Besonders bei glänzendem Lack sollte man darauf achten großzügig Farbe aufzutragen und nicht wie beim Streichen einer Wand nur eine dünne Schicht zu verwenden – man darf die unterliegende Schicht nach jeder Streichiteration nirgendwo auch nur annährend durchscheinen sehen.
Die vorletzte Schicht wird mit dem feinsten Schleifpapier bearbeitet. Die letzte Farbschicht lässt man optimalerweise 12-24 Stunden an einem windstillen und trockenen Ort vollständig trocknen.

ZUSAMMENBAUEN

Hat man nun alle Seiten fertig, kann man den unteren Teil der Box verleimen. Ist der Leim getrocknet, verwendet man genügend Nägel um für Stabilität zu sorgen. Wir haben hierbei die Seiten an die Front/Back und den Boden von unten an alle Seiten mit jeweils zwei Nägeln befestigt. Das Ganze ist natürlich abhängig vom Design und dem gewählten Holz bzw. der Holzdicke. Des weiteren kann man vor dem Zusammenbauen noch Löcher in die Rückseite Fräsen um später das Kabel für den Arduino / das iPhone Ladekabel durchziehen zu können.

LÖTEN & EINSETZEN

Als nächstes stellt man sicher, dass alle Buttons in die vorgesehenen Öffnungen passen. Ist dies nicht der Fall, kann man mit Hilfe des Schleifpapiers (dass man um einen Stift spannt) oder der Laubsäge **vorsichtig**!!! die Öffnungen bis zur passenden Größe erweitern. Sobald alles passt geht's ans Löten.

Wir haben uns dabei für drei verschiedenfarbige Kupferkabel entschieden (rot für Stromzufuhr, blau für Erde und grün für den Pin/die Pins an denen wir Daten/Spannung aus- bzw. ablesen) um beim später folgenden Verkabeln einfacher die Übersicht zu behalten. Hier ist es empfehlenswert nach erfolgreichem Löten alle Bauteile mit Hilfe des Arduinos durch simple Abfragen auf ihre Funktionalität zu prüfen und ggf. bei Defekten nach Ursachen (schlecht gelötet, falsch verkabelt etc.) zu suchen. Ein weiterer Tipp ist das Nutzen von kleinen Kabelbindern mit denen man alle Kabel jedes einzelnen Bauteils zu Bündeln zusammenzieht.

Funktionieren alle Bauteile wie geplant, streicht man die Windungen mit Sekundenkleber ein und drückt sie von unten in die Öffnungen. Dabei sollte sehr darauf geachtet werden, dass der Kleber nach außen hin nicht sichtbar ist und jedes Bedienelement sowohl nach oben als auch in die Box hinein so ausgerichtet ist wie es optisch am ansprechendsten bzw. innen am sinnvollsten für ein übersichtliches Verkabeln ist.

VERKABELN

Jetzt wo alle Buttons und Regler an ihrem Platz sind und die Frontseite vollständig fertig ist kommen wir zum Verbinden mit dem Arduino. Als erstes befestigt man den Arduino in der Box so, dass die vorher gemachten Löcher für das USB Kabel am Kabeleingang des Arduinos liegen. Das Breadboard kann mit der selbstklebenden Unterseite möglichst zentriert am Boden der Box festgeklebt werden.

Da wir hier mit 5V arbeiten, wird zunächst der Arduino mit dem Breadboard verbunden - durch das im Endeffekt alle Bauteile ihren Strom beziehen werden.

Als nächstes werden dann alle Bauteile an die jeweilige Spur im Breadboard angeschlossen und je nach Bauteilart (Analog/Digital) in die passenden Pins am Arduino eingesteckt. Hierbei sollte akribisch darüber Buch geführt werden welches Bauteil wo eingesteckt wird um im späteren Code nicht die falschen Pins abzufragen.

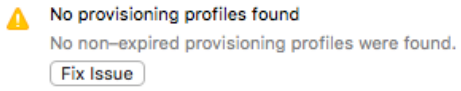
DER CODE (Arduino)

Hierzu gibt es nicht viel zu sagen, außer dass man die jeweiligen Pins als Ein bzw. Ausgänge definiert, Variablen für alle zu übermittelnden / speichernden Werte einführt und auf Delays soweit es geht verzichten sollte, da wir ja wollen, dass unser Controller durchgehend ansprechbar ist bzw. Informationen versendet ist.
Eine gute Quelle für Informationen ist *arduino.cc/en/Tutorial/HomePage* und die Herstellerseiten der jeweiligen Buttons, Drehregler und Potentiometer.

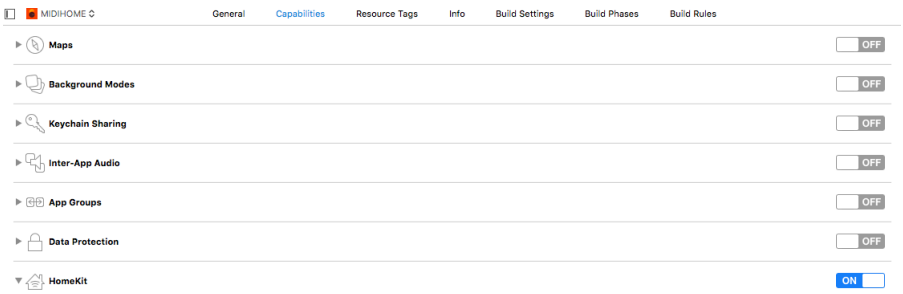
Das Herstellen einer Bluetooth Verbindung ist dabei abhängig vom verwendeten Modul.
Im Grunde startet man ein Serial für das BTLE und echoed die abgelesenen Werte der Bauteile nach einer aufgebauten Verbindung an das iPhone. Ein sehr empfehlenswertes Modul ist hier adafruit.com/product/1697 (Bluefruit LE).

DER CODE (iOS)

Hier wird es deutlich komplexer und aufwändiger – unter anderem weil ein Mac mit Xcode zum Programmieren und optimalerweise ein iPhone zum Debuggen benötigt wird.
Nach Erstellen eines neuen Projekts im Datei Menü von Xcode muss dem iPhone zunächst ein Entwicklerzertifikat des Entwicklers (uns) zugeordnet werden.
Das geschieht durch anschließen des iPhones an den Mac und betätigen des „Fix issue“ Buttons auf der Startseite des Projekts in Xcode.



Als nächstes muss im Reiter „Capabilities“ Homekit eingeschaltet werden.



Um die App zu programmieren werden gute Kenntnisse vieler Programmierkonzepte und insbesondere der Programmiersprache Swift benötigt. Eine empfehlenswerte Quelle ist hier das von Apple veröffentlichte und verfasste E-Book: „The Swift Programming Language“.

Des weiteren sollte man sich in die Homekit API einlesen (zu finden in Apples Developer Center). Diese ist aber sehr abstrakt gehalten, deshalb sollten noch Guides und Beispielcode betrachtet werden, wie z.B.:

- *developer.xamarin.com/guides/ios/platform_features/introduction_to_ios9/homekit/*
- *xmcgraw.com/learn-how-to-create-an-ios-app-using-homekit*

Hat man sich eingelesen beginnt man mit dem groben Konzept der App indem man in Storyboard das UI erstellt und Views mit Buttons verknüpft und „datasources“, „delegates“ und „ids“ für die einzelnen Elemente definiert. Dabei sollte man sich zunächst einmal auf eine iPhone Displaygröße beschränken da das Konfigurieren des automatischen Anpassens überflüssigen Aufwand mit sich bringt.

Wir empfehlen eine simple Benutzeroberfläche die zunächst nur über die notwendigen Einstellungsmöglichkeiten verfügt und hauptsächlich zur Erstellung eines „Homes“ inklusive der Räume und zur Zuordnung von per Bluetooth empfangenen Werten zu Homekit-Geräten dient (siehe xmcgraw.com Beispiel).

Hilfe bei der Herstellung einer Bluetooth Verbindung bzw. Verarbeiten von Daten seitens der iOS App erhält man beispielsweise unter:

- *raywenderlich.com/73306/arduino-tutorial-integrating-bluetooth-le-and-ios*

SONSTIGES

Erwähnenswert wäre hier noch, dass wir bei unserem Controller ein kleines Fach für das iPhone miteingebaut haben. Dazu wurde ein 19 cm * 10 cm großes Stück Holz dort an die Rückseite des Deckels geleimt und genagelt wo zuvor ein 18 cm * 9 cm großes Loch ausgesägt wurde. Diese entstandene Aussparung wurde dann mit Filz ausgekleidet, wobei davor noch ein kleines Loch gebohrt wurde um das iPhone Ladekabel von unten durchstecken zu können (um eine Dock-ähnliche Ablage zu schaffen).
Empfehlenswert ist es auch, den Deckel nicht permanent auf der Box zu befestigen, weil dies nachträgliches Reparieren, Erweitern und Debugging erschwert.

Eine einfache Möglichkeit wie man mit wenigen Buttons / Reglern alle Geräte im gesamten Haushalt ansteuern kann, ist das Erstellen von raumspezifischen Profilen für die Zuordnung der Buttons zu den jeweiligen Geräten. In der App sollte man dann zwischen den Profilen wechseln können.

Der einzuplanende Zeitaufwand bei diesem Projekt beträgt ohne Berücksichtigung des Trockenvorgangs der Farbe/Grundierung um die 60 Stunden (70% Code / 30% Box) unter Voraussetzung, dass alle benötigten Swift / iOS / Homekit Programmierkenntnisse schon vorhanden sind.

Das gesamte Konzept könnte man noch eine Ebene weitertragen und ein Dongle erstellen mit dessen Hilfe man jedes MIDI Gerät zu einer Steuereinheit für die programmierte App verwenden kann (MIDI Signale auslesen, ans iPhone übertragen und in der App entscheiden lassen um welchen MIDI-Controller es sich handelt und wie viele Bedienelemente dieser hat die man zur Steuerung der HomeKit Geräte verwenden kann).