



Un02 Firmware & Un02 ControlCenter

Bedienungsanleitung v.1.3a

Inhalt

<u>Einführung: was ist die UnO2 Firmware ?</u>	3
<u>Erste Schritte</u>	5
1. <u>Installation des UnO2 Chips</u>	5
2. <u>Installation des UnO2 ControlCenter Software</u>	5
3. <u>Verbinden von FCB1010 und Computer</u>	5
4. <u>Online-Registrierung der Firmware</u>	5
5. <u>Erstellen eines ersten Setups</u>	6
6. <u>Herunterladen des Setups auf das FCB1010</u>	6
7. <u>Test Ihres Setups</u>	6
<u>Die UnO2 ControlCenter Menüs</u>	7
1. <u>Das File-Menü</u>	8
2. <u>Das Edit-Menü (nur Mac)</u>	8
3. <u>Das FCB1010-Menü</u>	8
3.1. <u>Verbinden</u>	8
3.2. <u>Setup senden</u>	9
4. <u>Das Setup-Menü</u>	9
4.1. <u>Auswählen der MIDI-Ports</u>	9
4.2. <u>Registrieren der UnO2 Firmware</u>	10
5. <u>Das Hilfe-Menü</u>	11
5.1. <u>MIDI Monitor</u>	11
5.2. <u>User manual (Bedienungsanleitung)</u>	11
5.3. <u>UnO2 website (UnO2 Webseite)</u>	11
<u>Der UnO2 ControlCenter MIDI-Monitor</u>	12
<u>Die UnO2 Setup-Struktur</u>	14
<u>Beispiel 1 : Struktur eines typischen UnO2 Setups</u>	16
<u>Beispiel 2 : Senden von MIDI-Nachrichten</u>	18
<u>Beispiel 3 : Programmierung der Expression-Pedale</u>	19
<u>Beispiel 4 : Verwendung von Variablen</u>	20
<u>Beispiel 5 : Verwendung von bedingten Befehlen</u>	21
<u>Die UnO2 Programmiersprache</u>	22
0. <u>Kommentare</u>	22
1. <u>Definieren von Preset-, Effekt-, Trigger- und Sweep-Namen</u>	23
2. <u>Definieren der Bank-Struktur</u>	26
3. <u>Definieren von voreingestellten Inhalten (Presets)</u>	29

3.1.	<u>Definieren der MIDI-Kanäle</u>	30
3.2.	<u>Definieren der Daten-Variablen</u>	31
3.3.	<u>Definition des Anfangszustands des FCB1010</u>	32
3.4.	<u>Definieren der Bank-Initialisierung</u>	32
3.5.	<u>Definieren der Preset-Inhalte</u>	33
3.6.	<u>Definieren der Effekt-Inhalte</u>	33
3.7.	<u>Definieren der Trigger-Inhalte</u>	34
3.8.	<u>Definieren der Sweep-Inhalte</u>	35
4.	<u>Der Befehlsatz</u>	36
4.1.	<u>Fußtaster- und Pedalzuweisungsbefehle</u>	37
4.2.	<u>Befehle zur Aktivierung von Effekten und der Relais</u>	39
4.3.	<u>Navigation und Fernbedienung</u>	41
4.4.	<u>MIDI-Befehle</u>	42
4.5.	<u>Continuous Control-Befehle</u>	43
4.6.	<u>Verzögerungs-Befehl</u>	44
4.7.	<u>Variablen-Befehl</u>	45
4.8.	<u>Bedingte Befehle</u>	46
4.8.1.	<u>Die Bedingungs-Syntax</u>	47
4.8.2.	<u>if...then...else-Anweisungen</u>	48
4.8.3.	<u>while-Anweisungen</u>	49
4.8.4.	<u>switch-Anweisungen</u>	49
4.8.5.	<u>Bedingungen mit vordefinierten Variablen</u>	50
 <u>ANHANG : UnO2 Programmiersprachen-Referenz</u>		52
 <u>ANHANG : Werks-Reset, Selbsttest und Pedalkalibrierung</u>		55

Dokumenten-Versionen:

Version 1.3a	01/05/2022	Korrektur der wrt remote control (S. 41)
Version 1.3	01/03/2022	Zusätzliche Funktionen in Firmware v.1.3
Version 1.2	20/08/2020	Unterstützung für zusätzliche SysCommon- und SysRealtime-Nachrichten hinzugefügt. SwitchOn/SwitchOff-Befehle sind nicht mehr nur auf die Verwendung in Voreinstellungen beschränkt - Endlosschleifen-Erkennung der Effektaktivierung hinzugefügt.
Version 1.1	01/07/2020	Feature hinzugefügt: Verwendung von "while"-Schleifen. Anhang über Selbsttest & Kalibrierung hinzugefügt.
Version 1.0	20/05/2020	Erste Veröffentlichung

Einführung: was ist die UnO2 Firmware ?

Um die Philosophie hinter der UnO2-Firmware zu verstehen, ein kleines Wort zur Geschichte.

In den letzten 15 Jahren haben wir alternative Firmware für den Behringer FCB1010 herausgebracht. Wir nannten sie "UnO-Firmware", wie in "UnOffizielle Firmware", da wir nicht mit Behringer verbunden sind und diese Firmware daher kein offizielles Angebot von Behringer ist.

Wie viele Anwender in den letzten Jahren bezeugen werden, hat die UnO-Firmware mehrere Verbesserungen des ursprünglichen FCB1010 gebracht. Neben einigen wichtigen Fehlerbehebungen (wie z.B. die Korrektur von Fehlern in der MIDI-Merge-Funktionalität) haben wir einige stark nachgefragte Funktionen hinzugefügt, wie z.B. einen "echten Stompbox-Modus".

Selbst mit der installierten UnO-Firmware hat das FCB1010 noch einige große Einschränkungen, die hauptsächlich durch die sehr begrenzte Speicherkapazität des Gerätes beim Setup verursacht werden. Jedes Preset kann maximal 8 verschiedene MIDI-Befehle senden (5x ProgramChange (PC), 2x ControlChange (CC), 1x NoteOn), wobei jeder dieser 8 Befehle für alle Presets den gleichen MIDI-Kanal verwendet.

Um diese Einschränkungen zu überwinden bestand unser erster Ansatz darin, eine Hardware-Erweiterungsbox zu schaffen, die viel mehr Setup-Speicher enthält (128 Mal mehr als der FCB1010-Speicher, um genau zu sein) und somit viel mehr erweiterte Setups zusammen mit mehr Flexibilität bei diesen Setups ermöglicht. Wir nannten sie die TinyBox: <https://www.tinybox.rocks/>

Nach Fertigstellung des TinyBox-Designs und des zugehörigen Setup-Editors waren wir der Meinung, dass die enorme Flexibilität des TinyBox-Ansatzes auch in viel kleineren Setups funktionieren könnte, Setups, die sogar in den winzigen Setup-Speicher des FCB1010 passen würden. Anstelle der starren 10x10-Preset-Struktur mit 8 Meldungen pro Preset würden Sie es vielleicht vorziehen, nur einige wenige Bänke mit viel leistungsfähigeren Presets zu bestücken und diese flexibel anzuordnen (Bänke mit allen Presets, Bänke mit allen Stompboxes oder eine anpassbare Mischung aus beidem).

Damit sind wir bei der UnO2-Firmware angelangt. Es handelt sich nicht um eine Erweiterung der bekannten UnO-Firmware. Es handelt sich vielmehr um eine verkleinerte Version der TinyBox, die in das FCB1010 EPROM passt. Zusammen mit der Firmware kommt eine angepasste Version des TinyBox-Editors, die so benutzerfreundlich wie möglich gestaltet wurde. Er verfolgt einen völlig anderen Ansatz als die FCB/UnO ControlCenter-Software. Er ermöglicht es Ihnen, ein Setup zu erstellen, indem Sie es als Text in einem speziellen Texteditor aufschreiben, der eine Autovervollständigung bietet.

Die nächste Seite zeigt eine Vergleichsübersicht von UnO-Firmware vs. UnO2-Firmware vs. TinyBox. Einige der genannten Merkmale bedürfen einer genaueren Erläuterung, die Sie in weiteren Kapiteln dieses Handbuchs finden werden.

	UnO firmware	UnO2 firmware	TinyBox
# banks	10	Bis zu 200 (theoretisch*)	Bis zu 200
# presets	100	Bis zu 1000 (theoretisch*)	Bis zu 1000
# messages/preset	8	Unbegrenzt (theoretisch*)	Unbegrenzt
# MIDI channels	8 (global)	16 (für jede Nachricht)	16 (für jede Nachricht)
MIDI msg types	PC, CC, Note	Alle	Alle
MIDI Clock	-	Start, Continue, Stop	Start, Continue, Stop, Clock
Expr.Pedal Msg	CC	CC, AfterTouch, PitchBend	CC, AfterTouch, PitchBend
Pedal Sweep-Kurve	Linear	Linear, FastRising, SlowRising	Linear, FastRising, SlowRising
Bank-Layout-Möglichkeiten	10 x 10 Presets oder 19 x 5 Presets + 5 Globale Stompboxes	Beliebige Mischung aus Presets, Stompboxes, Momentan-Effekten für jede Bank	Beliebige Mischung aus Presets, Stompboxes, Momentan-Effekten für jede Bank
Direkt-Bank	-	+	+
Programmierbare Verzögerung	-	+	+
Verwendung von Datenvariablen	-	+	+
if..then...else Logik	-	+	+
Max. Setup-Größe	2048 bytes	2048 bytes	262144 bytes
Setlist-Unterstützung	-	-	+
MIDI Filter/Router	-	-	+
Eingebautes USB-MIDI-Interface	-	-	+
Drahtloses Status-Display	-	-	+
Phantom-Power	-	-	+

(*) die Gesamtzahl der Bänke oder Presets, die programmiert werden können, ist durch den 2048-Byte-Setup-Speicher begrenzt und hängt daher vom Inhalt jedes einzelnen Presets ab. Eine **Online-Version** des UnO2 ControlCenter-Editors ermöglicht es Ihnen, ein Setup zu erstellen und dessen Größe zu überprüfen, um festzustellen, ob es in die UnO2-basierte Lösung passt oder ob es die TinyBox-Hardware-Erweiterung erfordern würde.

Erste Schritte

1. Installation des UnO2 Chips

Anleitungen zum Austausch des Firmware-Chips finden Sie in diesem Behringer-Handbuch:
https://www.fcb1010.eu/downloads/Upgrade%20Manual_FCB1010_Rev_A.pdf

2. Installation des UnO2 ControlCenter Software

Das Software-Installationsprogramm (sowohl Windows- als auch Mac-Versionen) kann heruntergeladen werden unter: <https://www.fcb1010.eu/ccdownload.php>

Für den Zugang zu den Downloads benötigen Sie die auf der Chip-Verpackung angegebenen Registrierungsdaten. Bewahren Sie die Verpackung sicher auf, da Sie diese Registrierungsdaten später für das Herunterladen von Software-Updates oder für den Erhalt eines Rabatts auf zukünftige Firmware-Update-Chips benötigen!

3. Verbinden von FCB1010 und Computer

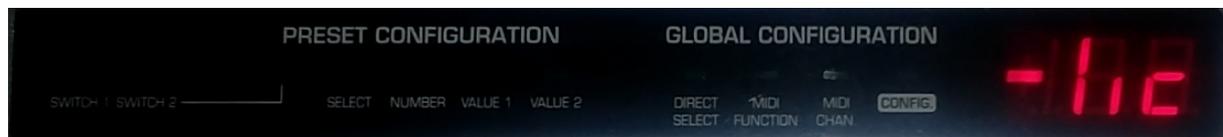
Verbinden Sie ein MIDI-Kabel vom MIDI-OUT-Anschluss des FCB1010 mit dem MIDI-IN-Anschluss Ihres MIDI-USB-Interfaces und ein zweites MIDI-Kabel vom MIDI-OUT-Anschluss des MIDI-USB-Interfaces mit dem MIDI-IN-Anschluss des FCB1010.

Starten Sie das UnO2 ControlCenter und wählen Sie den MIDI IN- und MIDI OUT-Anschluss, den Sie benutzen werden. Siehe auch Thema 4.1 des nächsten Kapitels.

Klicken Sie auf den Befehl "Verbinden", um die 2-Wege-Kommunikation zwischen FCB1010 und Computer einzuleiten. Siehe auch Thema 3.1 des nächsten Kapitels für weitere Einzelheiten.

4. Online-Registrierung der Firmware

Beim ersten Einschalten des FCB1010 nach dem Upgrade zeigt das Display "-lic" an:



Das bedeutet, dass die Firmware online registriert werden muss.

Vergewissern Sie sich, dass Ihr Computer mit dem Internet verbunden ist und folgen Sie den Anweisungen unter Punkt 4.2 des nächsten Kapitels, um die Registrierung durchzuführen.

Sobald die Registrierung abgeschlossen ist, ist die Firmware freigeschaltet und einsatzbereit:



5. Erstellen eines ersten Setups

Bevor Sie Ihr erstes Setup erstellen können, müssen Sie sich Zeit nehmen und die nächsten Kapitel des Handbuchs durcharbeiten. Diese erklären detailliert die Struktur der UnO2 Einrichtung, die verfügbaren UnO2 Funktionen und die "Programmiersprache", die zur Beschreibung Ihres Setups verwendet wird.

Wenn Sie in der Vergangenheit das FCB/UnO ControlCenter zur Erstellung Ihrer FCB1010-Setups verwendet haben, können Sie eine Starthilfe geben, indem Sie Ihr altes Setup in das UnO2-Format exportieren. Die neueste Version des FCB/UnO ControlCenter verfügt über eine entsprechende Menüoption. Beachten Sie jedoch, dass nicht alle "alten" Setups einfach so in die Struktur des UnO2-Setups passen werden. Wenn das Setup mehrere MIDI-Message für alle 100 Presets aktiviert hat, wird die entsprechende UnO2-Setup-Größe wahrscheinlich zu groß sein, um in das FCB1010 EEPROM zu passen. Sie müssen zuerst alle unbenutzten Messages filtern und sich dann die verschiedenen Möglichkeiten ansehen, Ihr Setup auf effizientere Weise neu zu schreiben, indem Sie den erweiterten UnO2-Befehlssatz nutzen.

6. Herunterladen des Setups auf das FCB1010

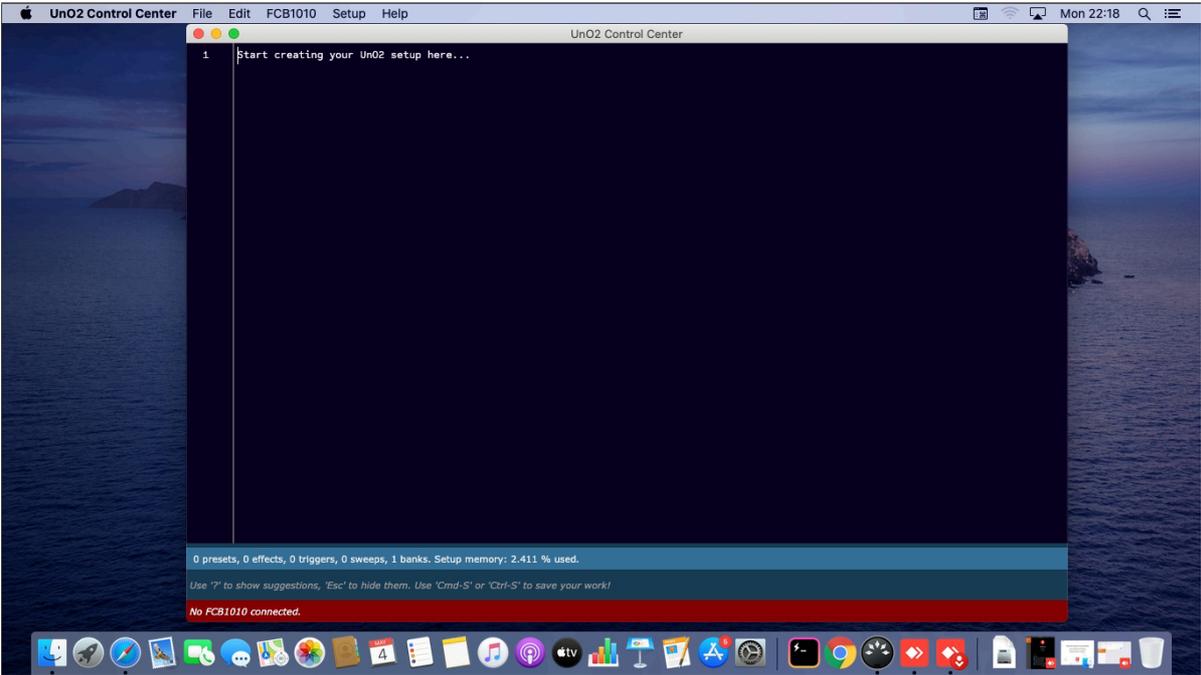
Beim Speichern Ihres Setups wird es automatisch in binäre Daten kompiliert, die mit einem einzigen Klick an den FCB1010 gesendet werden können. Siehe auch Thema 3.2 des nächsten Kapitels.

7. Test Ihres Setups

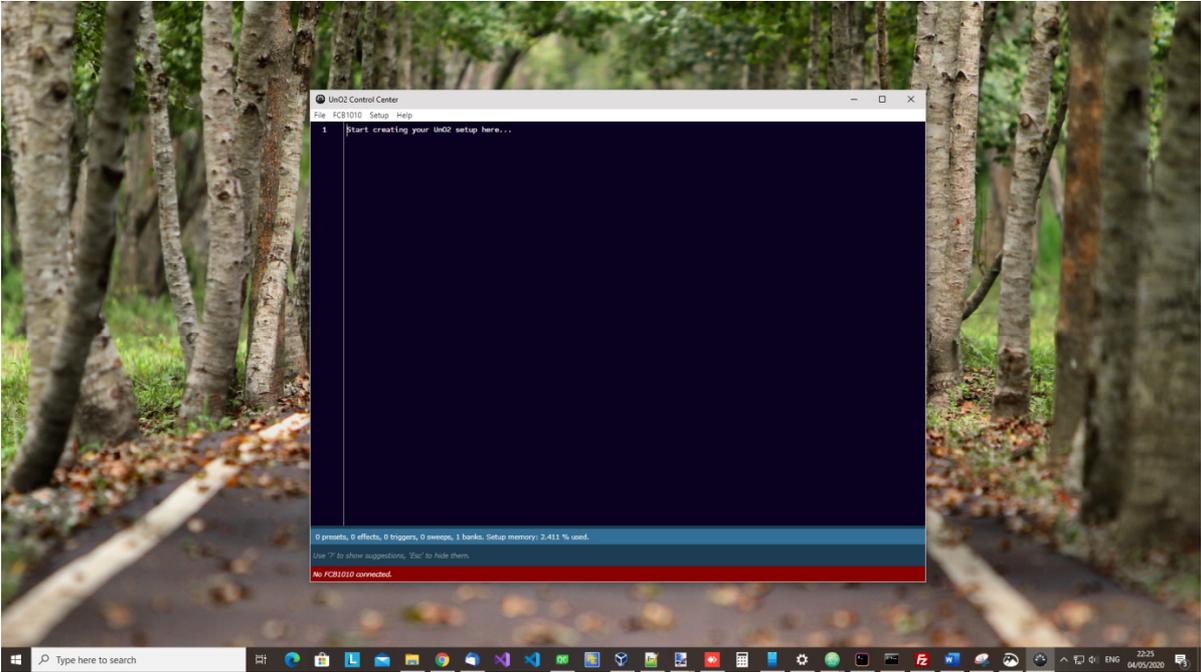
Die Dinge funktionieren nicht immer von Anfang an richtig. Wenn sich Ihr Gerät nicht wie erwartet verhält, sollten Sie die MIDI-Nachrichten, die der FCB1010 sendet, überprüfen. Das UnO2 ControlCenter hat dafür einen eingebauten MIDI-Monitor. Mit diesem Monitor können Sie sich einen detaillierten Einblick in den gesamten MIDI-Verkehr verschaffen und auch "manuell" MIDI-Befehle an Ihr Gerät senden, um zu experimentieren und die richtigen Steuerbefehle zu finden. Weitere Einzelheiten über den MIDI-Monitor finden Sie in einem nächsten Kapitel dieses Handbuchs.

- *Anmerkung* : Stellen Sie sicher, dass die Expressionspedale FCB1010 korrekt kalibriert sind. Dies nicht zu tun ist die häufigste Ursache für nicht funktionierende Expressionspedale. Kalibrierungsanweisungen finden Sie im Behringer-Handbuch oder durch googeln von "FCB1010 calibration".

Die UnO2 ControlCenter Menüs



Das UnO2 ControlCenter auf einem Mac



Das UnO2 ControlCenter unter Windows

1. Das File-Menü

Das Menü Datei bietet alle regulären Optionen zum Erstellen, Löschen, Speichern oder Kopieren von Setups. Während diese Setup-Dateien als Benutzereinstellungen "innerhalb" der Anwendung gespeichert werden, können Sie Ihr Setup zur Datensicherung sehr einfach als Textdatei auf einem USB-Stick speichern oder es online freigeben, indem Sie den vollständigen Text in ein normales Texteditor- oder Notepad-Programm kopieren und einfügen.

2. Das Edit-Menü (nur Mac)

Dieses Menü wird nur auf dem Mac benötigt, um die üblichen Tastaturkürzel für Kopieren und Einfügen im Texteditor zu aktivieren: Cmd-X, Cmd-C, Cmd-V, Cmd-A für Ausschneiden - Kopieren - Einfügen - Alles auswählen. Auch unter Windows stehen die Standard-Tastaturkürzel Ctrl-X, Ctrl-C, Ctrl-V, Ctrl-A zur Verfügung.

3. Das FCB1010-Menü

3.1. Verbinden

Der Befehl Connect stellt eine 2-Wege-Verbindung zwischen Computer und FCB1010 her. Mehrere Menüoptionen bleiben deaktiviert, solange keine Verbindung hergestellt wurde: das Senden eines Setups, die Registrierung der Firmware oder die Verwendung des MIDI-Monitors ist ohne Verbindung zum FCB1010 nicht möglich.

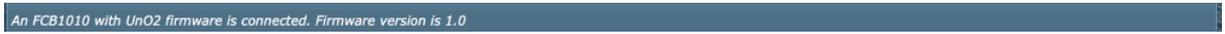
Der Verbindungsbefehl selbst ist deaktiviert, solange keine MIDI-Ports konfiguriert wurden. Siehe Thema 4.1.

Eine rote Statusleiste am unteren Rand der Anwendung zeigt an, dass eine Verbindung hergestellt werden muss :



No FCB1010 connected.

Sobald die Schaltfläche Verbinden angeklickt wird, färbt sich die Statusleiste blau und zeigt die Firmware-Version des angeschlossenen FCB1010 :



An FCB1010 with Uno2 firmware is connected. Firmware version is 1.0

Wenn keine Verbindung hergestellt werden kann, tritt ein Timeout auf :



A timeout occurred. Please check the MIDI connections.

Überprüfen Sie in diesem Fall **doppelt**, ob beide MIDI-Kabel zwischen Computer und FCB1010 korrekt angeschlossen sind und ob die richtigen MIDI-Ports im Einstellungsbildschirm ausgewählt wurden (siehe Thema 4.1).

3.2. Setup senden

Klicken Sie auf diesen Befehl, um das aktuelle Setup an Ihren FCB1010 zu senden. Wenn der Befehl deaktiviert ist, stellen Sie zunächst eine Verbindung mit dem FCB1010 her (siehe Thema 3.1).

Immer wenn Sie Ihr Setup im Texteditor ändern, wertet der eingebaute Compiler den Text aus. Unterhalb des Textbereichs zeigt eine Statusmeldung an, ob ein Fehler im Text entdeckt wurde. Wenn kein Fehler vorliegt, wandelt der eingebaute Compiler den Text in Binärdaten um, und die Statusleiste zeigt an, wie viel des FCB1010-Setup-Speichers für das Setup benötigt wird. Solange dieser Speicherverbrauch unter 100% bleibt, können Sie das Setup an den FCB1010 senden, indem Sie auf den Befehl "Send Setup" klicken.

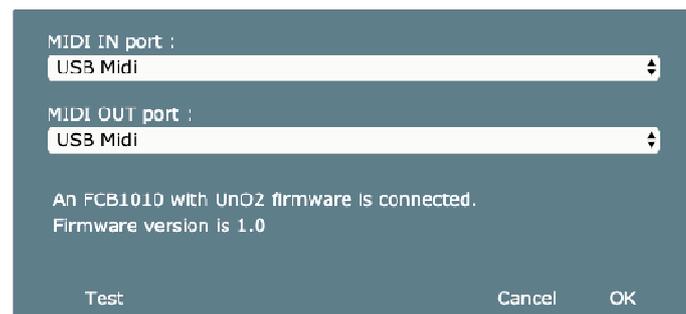
4. Das Setup-Menü

4.1. Auswählen der MIDI-Ports

Damit das UnO2 ControlCenter mit dem FCB1010 kommunizieren kann, müssen Sie angeben, welche MIDI-USB-Ports es verwenden kann:



Sie können Ihre Wahl erst dann bestätigen, wenn die erfolgreiche Kommunikation über diese Ports überprüft wurde. Klicken Sie auf die Schaltfläche **Test**. Wenn die 2-Wege-Kommunikation erfolgreich war, erscheint eine Meldung:



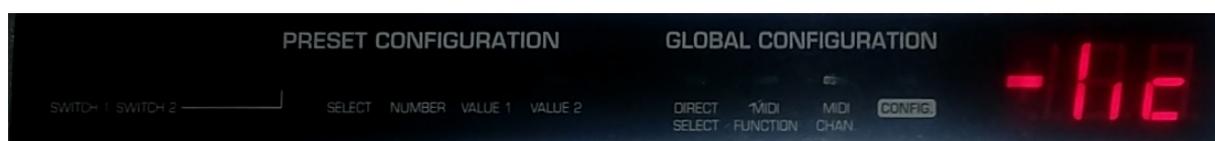
Wenn die Kommunikation fehlschlägt, überprüfen Sie bitte die MIDI-Verbindungen zwischen der MIDI-USB-Schnittstelle und dem FCB1010.

Die Auswahl der MIDI-Ports muss nur einmal vorgenommen werden. Wenn Sie das nächste Mal das UnO2 ControlCenter starten, können Sie direkt zum Befehl "Verbinden" gehen, ohne die MIDI-Ports erneut auswählen zu müssen. Es sei denn, die Ports sind natürlich nicht verfügbar. In diesem Fall schließen Sie das ControlCenter, schließen Sie das Interface an, starten Sie das ControlCenter und wählen Sie erneut die MIDI-Ports aus.

4.2. Registrieren der UnO2 Firmware

Bevor Sie die UnO2-Firmware verwenden können, muss sie online registriert werden. Diese Registrierung muss **nur einmal** durchgeführt werden, um die UnO2-Funktionalität freizuschalten. Dazu muss die FCB1010-Kommunikation initiiert werden (durch Klicken auf die Schaltfläche Verbinden - siehe 3.1), und der Editor muss mit dem Internet verbunden sein, um mit dem UnO2-Lizenzserver zu kommunizieren.

Zusammen mit dem Firmware-Chip haben Sie einen eindeutigen Registrierungscode sowie eine Kopie der bei der Bestellung eingegebenen persönlichen Daten erhalten. Kopieren Sie diese Daten in das Registrierungsformular und klicken Sie auf "Register". Die Firmware-Seriennummer muss nicht ausgefüllt werden, sie wird von der Software erkannt. Wenn alle Daten korrekt sind, wird die Firmware freigeschaltet und ist einsatzbereit.



UnO2 vor der Registrierung

Fill in the registration data received along with the firmware chip.
(use the exact data as received, with accents and special characters removed!)

Name :

Email :

Code :

Firmware serial number :

Register Close

Registrierungsformular



UnO2 nach der Registrierung

5. Das Hilfe-Menü

5.1. MIDI Monitor

Für die Fehlersuche steht ein eingebauter MIDI-Monitor zur Verfügung. Weitere Informationen finden Sie im nächsten Kapitel, das diesen MIDI-Monitor behandelt. Wenn die MIDI-Monitor-Option deaktiviert ist, müssen Sie zuerst eine Verbindung mit dem FCB1010 herstellen - siehe Thema 3.1

5.2. User manual (Bedienungsanleitung)

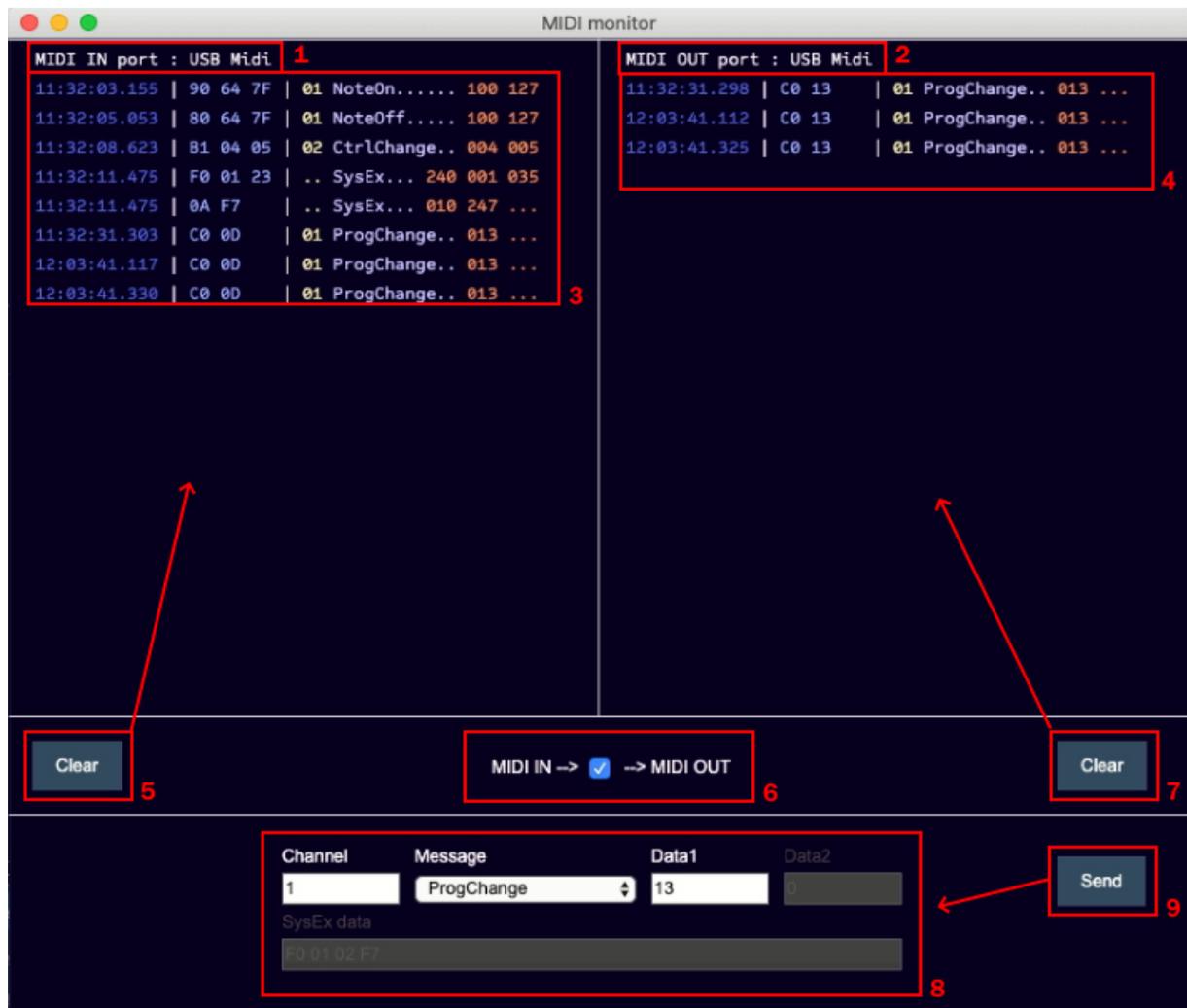
Diese Menüoption öffnet einen Link zur neuesten Version dieses Handbuchs.

5.3. UnO2 website (UnO2 Webseite)

Diese Menüoption öffnet einen Link zur fcb1010.eu Webseite.

Der UnO2 ControlCenter MIDI-Monitor

Bevor wir in die Details der Erstellung eines UnO2-Setups eintauchen, das den größten Teil dieses Handbuchs einnehmen wird, wollen wir kurz ein nützliches Hilfswerkzeug beschreiben, das Teil des UnO2 ControlCenters ist: der MIDI-Monitor.



Überblick über die MIDI-Monitor-Benutzeroberfläche :

- 1 : Vom ControlCenter verwendeter MIDI-IN-Anschluss
- 2 : Vom ControlCenter verwendeter MIDI-OUT-Anschluss
- 3 : Liste der **eingehenden** MIDI-Nachrichten
- 4 : Liste der **ausgehenden** MIDI-Nachrichten
- 5 : Löschen-Schaltfläche für die MIDI IN-Listenansicht
- 7 : Löschen-Schaltfläche für die MIDI IN-Listenansicht
- 6 : MIDI THRU-Funktion = Weiterleitung aller **eingehenden** MIDI-Nachrichten an den MIDI OUT-Anschluss
- 8 : Auswahl der MIDI-Nachricht: spezifizieren Sie eine beliebige MIDI-Nachricht, die an den MIDI-OUT-Anschluss gesendet werden soll.
- 9 : Send-Schaltfläche zum Senden der ausgewählten MIDI-Nachricht an den MIDI-OUT-Anschluss

Die MIDI-Nachrichtenliste enthält für jede empfangene oder gesendete MIDI-Nachricht folgende Daten :

- Zeitstempel, zu dem die Nachricht empfangen oder gesendet wurde
- Hexadezimale Darstellung der MIDI-Nachricht. Eine MIDI-Nachricht kann 1, 2 oder 3 Bytes enthalten, mit Ausnahme von SysEx-Nachrichten, die mehr Bytes enthalten können und auf mehreren Zeilen in der Nachrichtenliste angezeigt werden
- Textdarstellung der MIDI-Nachricht, d.h.:
 - Gegebenenfalls den MIDI-Kanal (zwischen 01 und 16)
 - Den MIDI-Nachrichtentyp
 - 0, 1 oder 2 MIDI-Datenbytes (oder mehrere Bytes im Falle von SysEx)

Mit der "MIDI THRU"-Funktion können Sie alle **eingehenden** MIDI-Nachrichten an den MIDI OUT-Anschluss weiterleiten. Dies geschieht einfach durch Setzen des Häkchens in der Checkbox zwischen "MIDI IN -->" und "--> MIDI OUT".

Sie benötigen diese "MIDI THRU"-Funktionalität, wenn Sie Ihr aktuelles FCB1010-Setup testen möchten. Schließen Sie dazu ein MIDI-Kabel vom MIDI-Ausgang des FCB1010 an den Computer und vom MIDI-Ausgang des Computers an Ihr Gerät an. Wenn Sie auf einen FCB1010-Fußschalter klicken, erscheinen alle gesendeten MIDI-Befehle in der Liste der MIDI-IN-Befehle, und da Sie diese an den MIDI-OUT-Port weiterleiten, erscheinen sie auch in der Liste der MIDI-OUT-Befehle. Wenn sich Ihr Gerät nicht wie erwartet verhält und Sie herausfinden möchten, welche MIDI-Nachrichten stattdessen gesendet werden sollen, verwenden Sie die Schaltfläche Senden, um einzelne MIDI-Nachrichten manuell zu senden. Sie erscheinen dann auch in der MIDI OUT-Liste.

ACHTUNG :

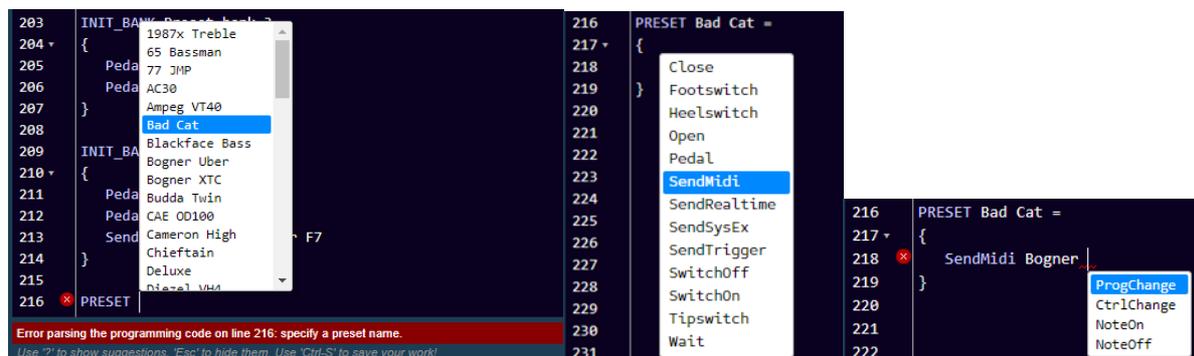
Aktivieren Sie **niemals** die Funktion "MIDI THRU", wenn Sie **beide** MIDI-Kabel an den FCB1010 angeschlossen haben (was bei Patchdumps der Fall ist)! Der FCB1010 mit UnO2-Firmware hat ebenfalls ein "MIDI THRU"-Verhalten: alle am MIDI IN-Port des FCB1010 ankommenden Meldungen werden automatisch an den MIDI OUT-Port weitergeleitet (mit Ausnahme spezifischer SysEx-Meldungen wie Versionsanforderungen, Patchdumps usw., welche intern verarbeitet werden). Sie werden verstehen, dass eine MIDI THRU-Funktionalität sowohl im FCB1010 als auch im MIDI-Monitor eine Endlosschleife erzeugen würde: eine vom FCB gesendete Nachricht würde vom MIDI-Monitor weitergeleitet und an den MIDI IN-Port des FCB zurückgeschickt, wo sie wiederum an den MIDI OUT-Port des FCB weitergeleitet wird, usw... Die Meldungen würden für immer 'im Kreis' herumlaufen, und die Meldungslisten des MIDI-Monitors würden sich schnell mit einem endlosen Nachrichtenstrom füllen. Wenn Sie dies bemerken, deaktivieren Sie einfach das Kontrollkästchen MIDI THRU: die Schleife wird unterbrochen und alles sollte wieder normal sein. Verwenden Sie die Clear-Schaltflächen, um die Nachrichtenlisten zu leeren.

Die UnO2 Setup-Struktur

Um den Inhalt eines UnO2-Presets festzulegen, verwenden Sie eine Programmiersprache, die eine begrenzte Anzahl von einfachen Befehlen wie `SendMidi`, `SendSysEx`, usw. enthält. Während Sie Ihr Setup erstellen, gibt Ihnen der Editor ständig Hinweise auf die Befehle, die im aktuellen Kontext zur Verfügung stehen. Am Anfang einer neuen Zeile tippen Sie einfach '?', um eine Liste der möglichen Befehle zu erhalten.

Ein Setup kann sehr einfach gehalten werden, indem Sie einfach einen oder einige wenige MIDI-Befehle für jedes Preset in Ihrem Setup angeben. Die verwendete Programmiersprache erlaubt es Ihnen jedoch, wenn Sie es wünschen, noch einen Schritt weiter zu gehen und zum Beispiel Variablen zu verwenden, ein wesentliches Konzept in jeder Programmiersprache. Der Inhalt einer Variable kann in einem Preset gesetzt oder modifiziert werden, dann können Sie in einem anderen Preset bedingte "if...then...else..."-Anweisungen verwenden, um auf den Variableninhalt einzuwirken. Auf diese Weise können Sie ein sehr dynamisches und intelligentes Setup erstellen.

Da ein UnO2-Setup im reinen Textformat beschrieben werden kann, ist es sehr einfach, Ihre Setups als Textdateien zu speichern, gemeinsam zu nutzen oder zu sichern. Auch das Editieren eines Setups ist sehr einfach: jeder Texteditor reicht aus. Wir stellen jedoch einen sehr intuitiven Setup-Editor zur Verfügung, der mehr kann als ein normaler Texteditor: Er verfügt über eine "intelligente Code-Vervollständigung", die Ihnen hilft, indem sie an jeder Stelle Ihres Setups die möglichen Optionen vorschlägt:



Editor mit Code-Vervollständigung

Es verfügt auch über eine clevere Autoformatierung, die zu einer tabellenkalkulationsartigen Setup-Übersicht führt, die während der Eingabe schön ausgerichtet bleibt :

```
110 BANKS =
111 {
112   Preset bank 1 : Vox Ac-30 | 65 Bassman | HiWatt Bright | Soldano SL01 | Budda Twin | Plexi Treble | 1987x Treble | JCM 800 | Mesa MKIV
113   Preset bank 2 : RectoOrangeV | Friedman HBE | Trainwreck X | Cameron High | JVM 410 001 | Marshall Bro | RectoRedMod | Bogner Uber | Peavey 5150
114   Preset bank 3 : Engl Powerb | Fry D60 Mor | FAS Modern | Diezel VM4 | Prince Tone | AC30 | Mesa Roadking | Marshall TLS60 | Mesa Mark IV
115   Preset bank 4 : Marshall JMP | Marshall JCM2000 | Ampeg VT40 | Marshall JMC2550 | Mesa Mark II | Bad Cat | Blackface Bass | Deluxe | Mesa Dual
116   Preset bank 5 : Jazz Chorus | Little Prince | Little Rebel | London | Matchless | Mr. Jack | Silverface clone | Twin | Soldano
117   Preset bank 6 : Chieftain | Sansamp | Vibrolux | RainHeart | Fuchs OD530 | CAE OD100 | JMP100 | Bogner XTC | KH ES 2002
118   Looper : Looper on/off | Looper rec/play | Looper dub | Looper reverse | Looper half | Looper bank | LOOP metronome | LOOP play once | LOOP undo
119   Standard song : Bogner Uber | Ampeg VT40 | Prince Tone | Looper rec/play | Looper on/off | Solo | Drive | Delay | Wahwah X/Y
120 }
```

Ein UnO2-Setup kann enthalten:

- bis zu 199 Bänke, durch die Sie mit den FCB1010 Up-/Down-Fußstastern blättern können, plus eine optionale "direkte" Bank, die mit einem Fußklick von jeder Bank aus zugänglich ist.
- bis zu 1000 Voreinstellungen. Jedes Preset kann eine praktisch unbegrenzte Anzahl von MIDI-Befehlen enthalten, die auf jedem der 16 MIDI-Kanäle gesendet werden können.
- bis zu 1000 Effekte. Wenn ein Fußtaster mit einem Effekt verbunden ist, verhält er sich wie ein Kippschalter mit 2 Zuständen (EIN/AUS), und er sendet für jeden der beiden Zustände einen anderen Satz MIDI-Befehle.
- bis zu 500 Trigger ('Auslöser'). Wenn ein Fußschalter mit einem Trigger verbunden ist, verhält er sich wie ein Momentschalter, der 2 verschiedene Sätze von MIDI-Befehlen senden kann: einen beim Drücken des Schalters und einen weiteren beim Loslassen des Schalters.
- bis zu 250 verschiedene "Sweeps". Sweeps sind Verhaltensweisen für die FCB1010 Expression-Pedale. Sie beschreiben nicht nur, welche kontinuierlichen MIDI-Befehle vom Expressionspedal gesendet werden müssen (ControlChange, PitchBend, ChannelPressure), sondern auch, welchem Wertebereich und sogar welcher Art von Sweep-Kurve (linear, schnell ansteigend (fast rising), langsam ansteigend (slow rising)) sie folgen müssen.

Jedes UnO2-Preset, jeder Effekt oder Trigger kann eine beliebige Kombination verschiedener Befehlsarten enthalten:

- MIDI-Befehle wie *ProgChange*, *CtrlChange*, *NoteOn*, *NoteOff*, *NoteOff*, *MIDIStart*, *MIDIContinue*, *MIDIStop*, *SysEx*
- einen Wartebefehl (Wait), der die MIDI-Übertragung für eine programmierbare Anzahl von Millisekunden oder Sekunden anhält
- Befehle zur (De)Aktivierung von Effekten oder Triggern, zur Änderung des Verhaltens der 2 FCB1010 Expressionspedale und zur Steuerung der 2 FCB1010 Buchsenausgänge
- Variablenbefehle, die den Inhalt der verschiedenen Arten von Datenvariablen einstellen oder ändern
- bedingte Befehle, die auf den aktuellen Inhalt dieser Variablen wirken

Ein UnO2-Setup kann bis zu 128 numerische Variablen, bis zu 256 boolesche Variablen und bis zu 256 String-Variablen verwenden.

- eine numerische Variable kann eine beliebige Zahl im Bereich 0-127 enthalten. Sie kann in jedem MIDI-Befehl verwendet werden, anstatt einen hartkodierten Wert anzugeben. Eine Ganzzahlvariable kann inkrementiert, dekrementiert, addiert oder subtrahiert und mit anderen Ganzzahlvariablen verglichen werden, um Entscheidungen zu treffen.
- eine boolesche Variable kann wahr oder falsch sein. Abhängig vom aktuellen Wert einer booleschen Variablen können verschiedene Nachrichten gesendet werden.
- eine String-Variable kann in Vergleichen zur Entscheidungsfindung verwendet werden. (eine Zeichenfolge im Kontext von Programmiersprachen ist ein kurzer Text, ein Wort). Die Verwendung von String-Werten anstelle von ganzen Zahlen kann dazu beitragen, Ihr Setup lesbarer zu machen.

Wichtige Anmerkung:

Die UnO2-Setup-Struktur wurde vom "großen Bruder" TinyBox entlehnt: eine Hardware-Erweiterung für den FCB1010, die dieselbe Setup-Struktur und Programmiersprache verwendet. Daher erwähnt die obige Übersicht riesige Obergrenzen wie 200 Bänke, 1000 Presets, 1000 Effekte, unbegrenzte MIDI-Befehle... Die Größe eines UnO2-Setups ist jedoch durch den begrenzten verfügbaren Speicherplatz auf dem FCB1010 stark eingeschränkt (2048 Bytes, im Vergleich dazu: 262000 Bytes in der TinyBox). Obwohl die Setup-Struktur also sehr hohe eingebaute Obergrenzen hat, begrenzt der Speicherplatz im FCB1010 die tatsächliche Größe eines UnO2-Setups.

Beispiel 1 : Struktur eines typischen UnO2 Setups

```
/* Unten sehen Sie die allgemeine Struktur eines UnO2-Setups.
   Sie können so viele Textkommentare zu einem Setup hinzufügen, wie Sie möchten.
   Ein einzeiliger Kommentar beginnt mit 2 Schrägstrichen, wie unten dargestellt.
   Ein mehrzeiliger Kommentar ist zwischen den Symbolen, die Sie hier sehen, eingebettet */

// Beginnen Sie mit der Auflistung aller Presets, Effekte, Trigger und Sweeps Ihres Setups:

PRESETS =
{
    Vox Ac-30
    65_Bassman
    HiWatt Bright
Soldano SLO1
    // etc...
}

EFFECTS =
{
    Chorus
    Compressor
Delay
    // etc...
}

TRIGGERS =
{
Sustain
    Looper rec/play
    Looperdub
    // etc...
}

SWEEPS =
{
    volume
    wah
    whammy
    // etc...
}

// Organisieren Sie dann alle Presets, Effekte und Trigger in einem Bank-Layout

BANKS =
{
    Looper          : Looper on/off | Looper rec/play | Looper dub   | Looper reverse ...
    Preset bank 1 : Vox Ac-30      | 65_Bassman   | HiWatt Bright | Soldano         ...
    Preset bank 2 : RectoOrangeV   | Friedman HBE | Trainwreck X  | Cameron High    ...
    Preset bank 3 : Engl Powerb    | Fry D60 Mor  | FAS Modern    | Diezel VH4     ...
    // etc...
}

// Jetzt können wir definieren, welche MIDI-Befehle jedes Preset senden muss

// Als erstes definieren Sie die MIDI-Kanäle, die in Ihrem Setup verwendet werden sollen.
// Jedem Kanal einen aussagekräftigen Namen zu geben hilft sehr, Ihr Setup lesbar zu machen.

CHANNEL Profiler= 1
CHANNEL Helicon = 2
CHANNEL Strymon = 3
CHANNEL Octaver = 10

// Bevor Sie die Presets definieren, sollten Sie vielleicht eine globale Initialisierung
// dessen festlegen, was geschieht, wenn der FCB1010 mit Strom versorgt wird:

INIT_FCB =
{
    Pedal 1 = volume
    Pedal 2 = wah
}
```

```

// Es ist auch möglich, MIDI-Befehle zu definieren, die gesendet werden, wenn eine bestimmte
// Bank aktiviert wird:

INIT_BANKPreset bank 3=
{
  SendMidi Profiler ProgChange 125
SendMidi Helicon ProgChange 3
  SwitchOn Chorus
  Pedal 2 = whammy
}

// Ein Preset-Inhalt kann so einfach sein wie ein einzelner MIDI-Befehl ... :
PRESET Vox Ac-30= SendMidi ProfilerProgChange 2

// ... oder mehrere von geschweiften Klammern umgebene Zeilen :
PRESET 65_Bassman=
{
SendMidi Profiler ProgChange 5
  SendMidi Helicon ProgChange 17
  SendMidi Octaver CtrlChange 13 127
}

// es ist auch möglich, dass Presets beim Loslassen des Schalters MIDI-Nachrichten senden:
PRESET_RELEASE 65_Bassman = SendMidi Profiler CtrlChange 11 127

// Ein Stompbox-Effekt sendet normalerweise mindestens 2 verschiedene MIDI-Nachrichten:
EFFECT_ON Chorus = SendMidi Strymon CtrlChange 3 127
EFFECT_OFF Chorus = SendMidi Strymon CtrlChange 3 0

// ein Trigger kann eine einzelne Nachricht senden ...:
TRIGGER_CLICK LooperOn/Off = SendMidi Helicon NoteOn 69 127

// ... oder eine Meldung beim Drücken des Schalters und eine weitere Meldung beim Loslassen
// des Schalters:
TRIGGER_CLICK OctaveUp = SendMidi Octaver CtrlChange 13 127
TRIGGER_RELEASE OctaveUp = SendMidi Octaver CtrlChange 13 0

// Ein Sweep definiert, welche MIDI-Befehle ein Expression-Pedal senden kann:
SWEEP volume = SendMidi Strymon CtrlChange 7 40-100 SlowRising
SWEEP whammy = SendMidi DX7 PitchBend 0-127

```

Beispiel 2 : Senden von MIDI-Nachrichten

```
// Das folgende Beispiel gibt einen Überblick über alle unterstützten MIDI-Nachrichten
// Ein Preset kann eine einzelne Nachricht oder mehrere Nachrichten auf verschiedenen Kanälen
// senden

CHANNEL MyGear = 10
CHANNEL MySynth = 3

// Anstatt feste Werte zu verwenden, können Sie auch "Variablen" in MIDI-Befehlen verwenden,
// wie später noch ausführlicher gezeigt wird

VAR $cc = 10
VAR $delay = 20
VAR $mix = 100

PRESET SimplePreset = SendMidi MyGear ProgChange 1

PRESET ComplexPreset =
{
  SendMidi MyGear ProgChange 125
  SendMidi MyGear CtrlChange 13 127
  SendMidi MySynth NoteOn 72 120
  SendMidi MySynth NoteOn 76 120
  Wait 20
  SendMidi MySynth NoteOff 76 0
  SendMidi MySynth NoteOff 72 0

  // Der obige Befehl Wait (Warten) fügt eine Verzögerung zwischen 2 MIDI-Nachrichten ein.
  // Die Verzögerung wird in 0,1s-Einheiten ausgedrückt, so dass "Wait 20" zu einer
  // Verzögerung von 2 Sekunden führt.

SendRealtime MIDISTart
SendRealtime MIDIContinue
SendRealtime MIDISTop
SendRealtime SystemReset

SendSysCommon SongSelect 100
SendSysCommon SongPointer 16000
SendSysCommon TuneRequest

SendSysEx F0 00 20 33 02 7F 01 00 32 59 00 40 F7

// Es können für alle Werte in den obigen MIDI-Nachrichten ganzzahlige Variablen verwendet
// werden. Sogar eine SysEx-Meldung kann Variablen enthalten:

SendMidi MyGear CtrlChange $cc 127
Wait $delay
SendSysEx F0 7F 01 $mix F7
}
```

Beispiel 3 : Programmierung der Expression-Pedale

```
CHANNEL MyGear = 10

// Sie können die MIDI-Befehle, den Bereich und den Sweep-Typ für jedes der Expression-Pedale
// ändern. Der Sweep-Typ ist standardmäßig linear, kann aber auch auf SlowRising oder
// FastRising eingestellt werden.

// Zuerst definieren Sie alle verfügbaren kontinuierlichen Kontrollelemente oder "Sweeps":

SWEEP volume = SendMidi MyGear CtrlChange 07 0-127 SlowRising
SWEEP whammy = SendMidi MyGear CtrlChange 19 0-127
...

// Dann können Sie in jedem Preset oder Effekt angeben, welcher Sweep
// auf jedem der 2 Expressionspedale aktiv sein soll.
// Zum Beispiel könnte ein Stomp-Switch das Wah-Pedal in einen Whammy-Effekt verwandeln:

EFFECT_ON ActivateWhammy = Pedal 2 = whammy
EFFECT_OFF ActivateWhammy = Pedal 2 = wah

// Sie können sogar einen virtuellen "Tipswitch" oder "Heelswitch" für jedes Expressionspedal
// definieren:

Tipswitch 2 = ActivateWhammy

// Wenn Sie den obigen Effekt mit dem "Tipswitch" von Pedal 2 verbinden, wechselt das Pedal
// zwischen beiden Funktionen, wenn die Pedalspitze ganz nach unten gedrückt wird. Es ist eine
// gute Idee, ein kleines Stück Schaumstoff unter das Pedal zu kleben. Auf diese Weise haben
// Sie durch Bewegen des Pedals den vollen Einstellbereich, und der virtuelle Schalter wird
// nur dann aktiviert, wenn Sie etwas mehr Kraft auf die Pedalspitze ausüben.
```

Beispiel 4 : Verwendung von Variablen

```
// Die Verwendung von 'Variablen' ist in Programmiersprachen etwas sehr Übliches.
// Es fügt dem Uno2-Setup enorme Möglichkeiten hinzu.

// Es gibt 3 Typen von Variablen:
// - integer ( = numeric (numerisch))
// - boolean ( = true or false (wahr oder falsch))
// - string ( = text (Text))
// Sie können eine Variable an ihrem führenden "$" erkennen:

VAR $currentBank = 1           // dies sind ganzzahlige Variablen
VAR $currentPreset = 35
VAR $delay = false           // dies ist eine boolesche Variable
VAR $currentSong = "Go with the flow" // dies ist eine Text-Variable

// Sie können den Wert jeder Variablen einstellen, wenn ein bestimmtes Preset getriggert wird:

PRESET no_one =
{
  $currentSong = "No one knows"
  $delay = true
  // ...
}

// Sie können auf verschiedene Weise mit ganzzahligen oder booleschen Werten arbeiten:

PRESET examples =
{
  $currentBank++           // increment (erhöhen)
  $currentBank--          // decrement (vermindern)
  $currentBank+= 10       // einen konstanten Wert hinzufügen
  $currentPreset = $currentBank + 5 // berechnen
  $delay = !$delay        // einen booleschen Wert invertieren
}

// Die wahre Stärke von Variablen wird im nächsten Beispiel deutlich, das bedingte Befehle
// behandelt
```

Beispiel 5 : Verwendung von bedingten Befehlen

```
// 'Bedingte Befehle' sind ein weiteres gängiges Konzept in traditionellen
// Programmiersprachen:
// 'if...then...else...' ('wenn...dann...sonst...') Anweisungen erlauben es einer Applikation,
// Entscheidungen zu treffen.
// Die folgenden Beispiele zeigen, wie sich Ihr Setup je nach dem Wert einer beliebigen
// Variablen unterschiedlich verhalten kann.

PRESET Sample =
{
// hier ist $solo eine boolesche Variable, sie kann wahr oder falsch sein:

    if($solo)
    {
        // sende eine Reihe von MIDI-Nachrichten...
    }
    else if($currentBank > 1)
    {
        // sende eine weitere Reihe von Nachrichten...
    }

    // eine 'switch'-Anweisung prüft den Wert einer Variablen
    // und spezifiziert den Code, der für jeden Wert ausgeführt werden soll:

switch($currentSong)
{
    case "No one knows":
        SendMidi MyGear CtrlChange 112 127
        break
    case "Go with the flow":
        SendMidi MyGear CtrlChange 12 0
        SendMidi MyGear CtrlChange 113 127
        break
    // und so weiter...
    default:
        SendMidi MyGear CtrlChange 12 127
        break
}

// Durch die Verwendung einer 'while'-Anweisung ist es sogar möglich, Schleifen zu erzeugen,
// zum Beispiel um einen Effekt ein- oder auszublenden.

    $fadeout = 100
    $fadein = 0
    while($fadeout > 0)
    {
        SendMidi MyGear CtrlChange 07 $fadeout
SendMidi MyOtherGear CtrlChange 07 $fadein
        Wait 1
    $fadeout -= 5
        $fadein += 5
    }
}
```

Die UnO2 Programmiersprache

Auf den folgenden Seiten wird die Syntax für die Erstellung eines UnO2-Setups im Detail beschrieben.

Es ist wichtig, die korrekte Reihenfolge der verschiedenen Teile in Ihrem Setup einzuhalten:

1. Definieren Sie Preset-, Effekt-, Trigger- und Sweep-Namen
2. Definieren Sie die Bankstruktur des Setups
3. Definieren Sie optional Befehle, die beim Einschalten des Geräts gesendet werden sollen
4. Definieren Sie optional Befehle, die bei der Auswahl jeder Bank gesendet werden sollen
5. Definieren Sie alle Inhalte für die Presets, Effekte, Trigger und Sweeps (in beliebiger Reihenfolge)

0. Kommentare

Mehrzeilige Kommentare können an jeder Stelle eines Setups verwendet werden, um die verschiedenen Teile eines Setups zu dokumentieren

```
/*  
[hier kann irgend ein  
    mehrzeiliger Text stehen...]  
*/
```

Inline-Kommentare können am Anfang jeder Zeile oder am Ende jedes Befehls verwendet werden

```
// [hier kann irgend ein Inline-Text stehen...]  
SendMidi KPA ProgChange 10 // [irgend ein Kommentar am Ende der  
                             // Zeile]
```

1. Definieren von Preset-, Effekt-, Trigger- und Sweep-Namen

Zu Beginn des Setups müssen Sie alle Namen der Setup-Elemente (Presets, Effekte, Trigger und Sweeps) auflisten. Diese Namen können dann bei der Festlegung des Setup-Layouts und der Befehle für jedes Preset verwendet werden.

Obwohl nicht alle 4 Elementtypen notwendig sind, müssen Sie zumindest einige Presets angeben, bevor Sie mit der Definition der anderen Teile Ihres Setups fortfahren können.

INFO

- ein **Preset** ist das Hauptelement in Ihrem Setup. Es wird normalerweise durch Anklicken eines der 10 Fußtaster des FCB1010 aktiviert. Beim Aktivieren eines Presets wird eine (praktisch unbegrenzte) Anzahl von MIDI-Befehlen gesendet, und die LED des Fußtasters leuchtet auf, um anzuzeigen, dass dieses Preset gerade aktiv ist. Es kann nur ein Preset zur gleichen Zeit aktiv sein, so dass die Auswahl eines anderen Presets automatisch die LED des vorherigen Presets ausschaltet. Obwohl wahrscheinlich wenig benutzt, kann ein Preset bei Bedarf auch Befehle beim Loslassen des Tasters senden.
- Ein **Effekt** kann auch einem der 10 Fußtaster des FCB1010 zugeordnet werden. Der Hauptunterschied zu einem Preset besteht darin, dass ein Effekt typischerweise 2 Zustände hat: EIN oder AUS. Durch Klicken des Effekt-Fußtasters wird zwischen diesen beiden Zuständen umgeschaltet, und die Taster-LED schaltet sich zusammen mit dem Effekt ein oder aus. Im Gegensatz zu Presets ist es durchaus möglich, mehrere Effekte gleichzeitig zu aktivieren. Sie müssen festlegen, welche MIDI-Befehle sowohl bei der **Effektaktivierung** als auch bei der **Effektdeaktivierung** gesendet werden sollen.
- ein **Trigger** ist einem Effekt sehr ähnlich, außer dass er bei jedem Klick nicht zwischen 2 Zuständen wechselt. Stattdessen geht er beim Drücken des Fußtasters auf EIN und beim Loslassen des Fußtasters auf AUS. Daher könnte man ihn auch als **Momentan-Effekt** bezeichnen. Ein typisches Beispiel für einen solchen Momentan-Effekt ist ein Sustain-Pedal: Es wird nur aktiviert, während Sie den Fußschalter gedrückt halten. Genau wie bei einem Effekt müssen Sie mindestens 2 MIDI-Befehle angeben: einen zum Aktivieren und einen zum Deaktivieren des Effekts.

Der Grund, warum wir dies als **Trigger** bezeichnen, liegt darin, dass dasselbe Setup-Element auch zum Auslösen einer bestimmten Aktion verwendet werden kann. Ein Looper-Befehl ist ein gutes Beispiel: Ein "REC/PLAY"- oder "UNDO/REDO"-Befehl ist nur ein einmaliger Befehl, der an den Looper gesendet wird. In diesem Fall werden Sie nur einen MIDI-Befehl für das Drücken des Fußtasters angeben, und wahrscheinlich keinen für das Loslassen des Fußtasters. Im Gegensatz zu einem Preset oder Effekt leuchtet die Fußtaster-LED eines Triggers nach dem Loslassen des Trigger-Tasters nicht mehr.

- ein Sweep ist eine ganz bestimmte Art von Preset. Sie können ihn nicht mit irgendeinem Fußtaster verbinden, sondern nur mit einem der beiden Expression-Pedale des FCB1010. Der Sweep-Inhalt gibt an, welche kontinuierlichen MIDI-Befehle jedes der Pedale sendet, wenn es bewegt wird - in der Regel sind es ControlChange-Befehle zur Änderung von Lautstärke, Expression oder kontinuierlichen Effekten wie Wah oder Whammy. Die Tatsache, dass ein UnO2-Setup "Sweep"-Elemente enthält, ermöglicht es Ihnen beispielsweise, das Verhalten des Expressionspedals in Abhängigkeit vom gerade aktiven Preset auf einfache Weise zu ändern.

SYNTAX :

```
PRESETS =
{
  [Preset-Name]
  ...
}

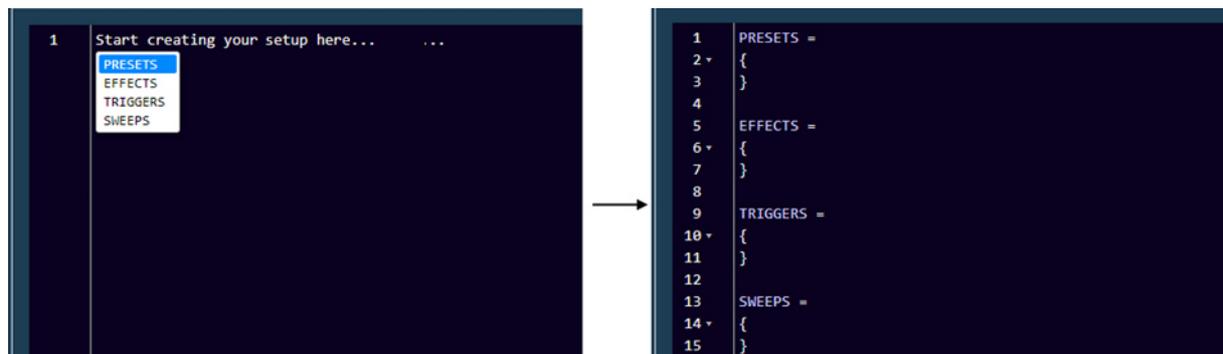
EFFECTS =
{
  [Effect-Name]
  ...
}

TRIGGERS =
{
  [Trigger-Name]
  ...
}

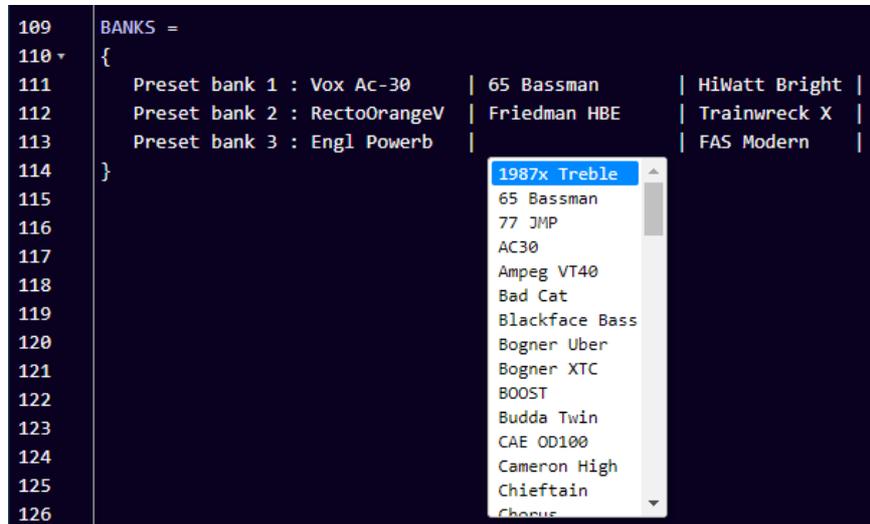
SWEEPS =
{
  [Continuous Control-Name]
  ...
}
```

Geben Sie in jeder der 4 Listen einen Namen pro Zeile an.

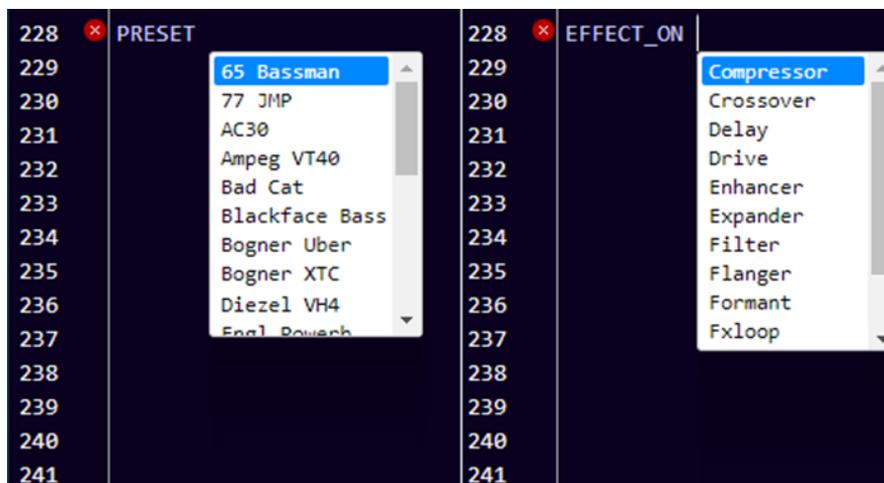
Die Autovervollständigungs- oder "Intellisense"-Funktionalität des UnO2 ControlCenter-Editors hilft Ihnen beim Einrichten dieser anfänglichen Setup-Struktur: Nachdem Sie ein neues Setup erstellt haben, geben Sie '?' ein, um eine Dropdown-Liste der verfügbaren Befehle zu erhalten. Wenn Sie <ENTER> 4 Mal auf die vorgeschlagene Befehlsliste klicken, erhalten Sie die 4 leeren Listen, die Sie nun mit Preset-Namen füllen können:



UnO2 ControlCenter verwendet diese Namenslisten, um Ihnen bei der Festlegung weiterer Teile des Setups zu helfen. Wenn Sie zum Beispiel das Bank-Layout definieren (wie später in diesem Kapitel erklärt wird), schlägt der Editor automatisch eine Liste der verfügbaren Presets zur Auswahl vor:



Auch bei der Definition der Preset-Inhalte weiter unten im Setup erscheint eine Dropdown-Liste zur Auswahl der verfügbaren Presets, Effekte, Trigger oder Sweeps:



Da Sie den Inhalt jedes Presets nur einmal in Ihrem Setup definieren sollten, passt der Editor die Dropdown-Listen schrittweise an und zeigt nur die Presets an, die noch keinen Inhalt definiert haben.

2. Definieren der Bank-Struktur

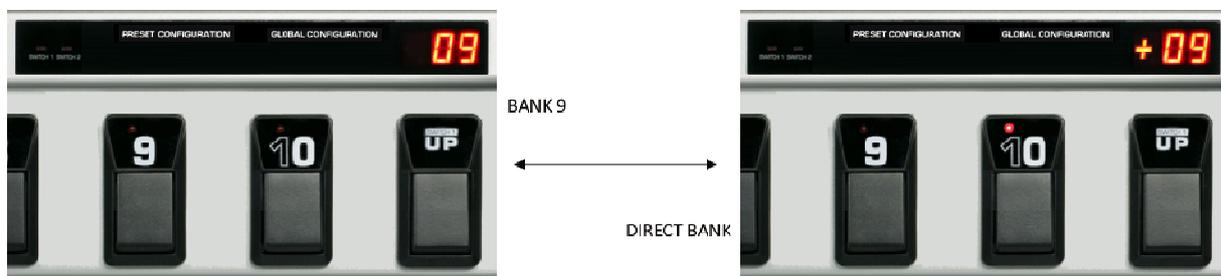
Sobald Sie alle Presets, Effekte und Trigger aufgelistet haben, die in Ihrem Setup verwendet werden sollen, können Sie damit beginnen, sie in Bänken zu organisieren.

Standardmäßig ist ein UnO2-Setup in Bänke von 10 Presets strukturiert, die den 10 Preset-Tastern auf dem FCB1010 entsprechen. Mit den Up- und Down-Tastern können Sie durch alle Bänke blättern. Theoretisch können Sie bis zu 199 Bänke definieren. Warum 199? Weil dies die höchste Zahl ist, die auf dem "2,5-stelligen" Display des FCB1010 angezeigt werden kann:



Direkt-Bank

Neben diesem Standard-Layout von 199 "sequentiellen" Bänken, die über die Up/Down-Taster zugänglich sind, können Sie auch eine "Direkt-Bank" hinzufügen. Diese Direkt-Bank enthält in der Regel eine Reihe von Presets oder Effekten, auf die Sie jederzeit problemlos zugreifen können. Oder sie kann einen bestimmten Befehlssatz enthalten (z.B. für die Looper-Steuerung), während die regulären Bänke verschiedene Sound-Presets enthalten. Wenn Sie eine Direkt-Bank verwenden, kann diese Bank mit dem Fußtaster 10 der FCB1010 aktiviert werden, unabhängig davon, in welcher Bank Sie sich gerade befinden. Ein Klick aktiviert die Direkt-Bank, ein weiterer Klick auf denselben Taster 10 führt zur vorherigen Bank zurück. Ein "+"-Zeichen auf dem Display zeigt an, dass die Direkt-Bank derzeit aktiv ist:



Da in diesem Modus der Fußtaster 10 ein dedizierter Direkt-Bank-Schalter ist, kann jede Bank nun 9 statt 10 Presets enthalten.

SYNTAX :

```
GLOBALSWITCH [1...10] = [presetname]

BANKS =
{
  [bank name] : [preset name] | [preset name] | ... | [preset name]
  [bank name] : [preset name] | [preset name] | ... | [preset name]
  ...
}

// oder wenn Sie die Direct Bank-Funktionalität nutzen möchten :

GLOBALSWITCH [1...10] = [presetname]

USE_DIRECT_BANK

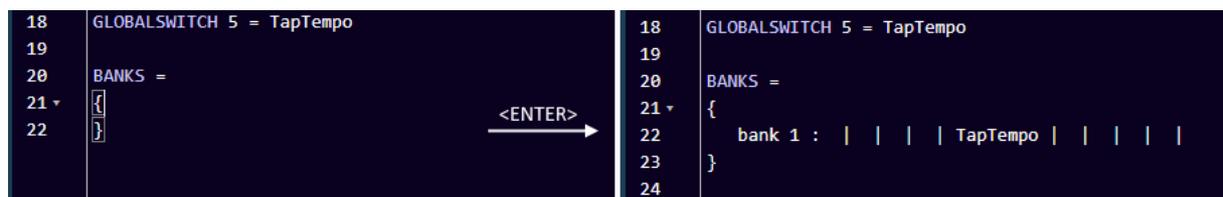
BANKS =
{
  Direct Bank: [preset name] | [preset name] | ... | [preset name]
  [bank name] : [preset name] | [preset name] | ... | [preset name]
  ...
}
```

Wenn Sie die Anweisung "USE_DIRECT_BANK" vor der Angabe BANKS hinzufügen, definiert die erste Zeile in der Bankenliste den Inhalt der Direkt-Bank. Sie werden feststellen, dass der Editor den Namen der ersten Bank automatisch in "Direct Bank" ändert, dies kann nicht geändert werden.

Globale Schalter

Auch ohne die "Direct Bank"-Struktur können Sie einen oder ein paar Effekte in allen Bänken zur Verfügung stellen, indem Sie einfach das gleiche Preset oder den gleichen Effekt mit dem gleichen Schalter in jeder Bank verknüpfen. Um Ihnen das zu erleichtern, können Sie eine oder mehrere "GLOBALSWITCH"-Definitionen vor der BANKS-Spezifikation hinzufügen. Wenn Sie dies tun, trägt der Editor automatisch den/die vorgegebenen Preset-Namen auf der/den vorgegebenen Schalterstellung(en) in jeder neuen Bank ein.

Wenn Sie z.B. Schalter 5 als globalen Schalter für TapTempo festgelegt haben, können Sie einfach <ENTER> im Abschnitt BANKS (zwischen den beiden geschweiften Klammern) anklicken, und der Editor wird eine neue Bank-Vorlage für Sie erstellen, wobei die Position von Schalter 5 bereits die TapTempo-Voreinstellung enthält:



Danach können Sie einfach damit beginnen, einen Preset-Namen für jede Schalterposition einzugeben, und es erscheint eine Dropdown-Liste, in der die Eingabe mit gültigen Preset-, Effekt- oder Triggernamen automatisch vervollständigt wird:



Zwischen jeder Schalterzuweisung wird automatisch ein "Pipe"-Zeichen (vertikale Linie) eingefügt, und der Smart Editor passt den Abstand zwischen den zugewiesenen Voreinstellungen automatisch an, um ein sauber formatiertes Rasterlayout zu erhalten:



Wie oben erklärt, können Sie eine "Direkt-Bank" angeben, bevor Sie die Liste der regulären Bänke erstellen. In diesem Fall werden alle Bankdefinitionen 9 statt 10 Presets enthalten.

Wenn Sie einen Schalter "leer" lassen möchten, lassen Sie einfach die Schalterzuweisung leer, ohne die vertikalen Trennlinien in der Bankdefinition zu entfernen.

Anmerkung :

Es ist von Vorteil, genau zu verstehen, wie die GLOBALSWITCH-Definitionen, die der Bankenliste vorangestellt werden können, funktionieren. Eigentlich handelt es sich dabei nur um eine Komfortfunktion des Editors, um zu vermeiden, dass in jeder Bank immer wieder das gleiche Preset für diesen globalen Schalter ausgewählt werden muss. Der Editor füllt die Schalterstellung automatisch für Sie aus. Abgesehen davon wird diese Global-Switch-Definition nicht an den FCB1010 gesendet, so dass der Switch nicht "gezwungen" wird, nur dieses eine Preset zu enthalten - Sie können den Inhalt eines Global Switches in jeder der Bänke danach immer noch ändern. Sie können z.B. eine GLOBALSWITCH-Definition hinzufügen oder ändern, nachdem Sie bereits eine große Anzahl von Bänken definiert haben, und diese Einstellung wird von da an für alle neu zum Setup hinzugefügten Bänke verwendet, ohne die Switch-Zuordnung bereits definierter Bänke zu ändern.

SYNTAX :

```
NO_UPDOWN_SWITCHES
BANKS =
{
[bank name] : [preset 1] | [preset 2] | ... | [preset 12]
}
// oder wenn Sie die Direktbankfunktionalität nutzen möchten :
NO_UPDOWN_SWITCHES
USE_DIRECT_BANKS
BANKS =
{
Direct Bank : [preset 12] | [preset 13] | ... | [preset 22]
[bank name] : [preset 1] | [preset 2] | ... | [preset 11]
}
```

Falls Sie nicht mehrere Bänke in Ihrem Setup benötigen, können Sie die Up/Down-Schalter des FCB1010 als 2 zusätzliche Preset-Schalter verwenden. Fügen Sie die Zeile **NO_UPDOWN_SWITCHES** vor der Bank-Definition in Ihr Setup ein, und Sie können 12 statt 10 Presets in einer Bank angeben.

Da die Up/Down-Schalter des FCB1010 keine LED enthalten, wird der Zustand dieser beiden Preset-Schalter stattdessen auf den kleinen LEDs "SWITCH1" und "SWITCH2" angezeigt: Die LED SWITCH1 leuchtet, wenn das Preset des "Up"-Schalters aktiv ist, die LED SWITCH2 leuchtet, wenn das Preset des "Down"-Schalters aktiv ist.

Diese Funktionalität kann auch mit **USE_DIRECT_BANK** kombiniert werden. Dies führt zu einem Setup mit 2 Bänken mit jeweils 11 Presets. In diesem Fall wird der "Down"-Fußschalter anstelle des Fußschalters 10 verwendet, um zwischen den 2 Bänken umzuschalten.

Bemerkung :

Diese Option ist in Firmware v.1.3 oder höher verfügbar.

3. Definieren von voreingestellten Inhalten (Presets)

SYNTAX :

```
CHANNEL channelname = [1...16]
VAR $intvarname      = [0...127]      // bis zu 128 Integer-Variablen
VAR $boolvarname    = [true/false] // bis zu 256 Boolean-Variablen
VAR $stringvarname = "any string" // bis zu 256 String-Variablen

INIT_FCB                = ... // einzelner Befehl,
                        // oder Liste von Befehlen zwischen
                        // geschweiften Klammern

INIT_BANK bank        = ...

PRESET preset         = ...
PRESET_RELEASE preset = ...

EFFECT_ON effect     = ...
EFFECT_OFF effect    = ...

TRIGGER_CLICK trigger = ...
TRIGGER_RELEASE trigger = ...

SWEEP sweep          = ... // Continuous Control Befehl(e)
```

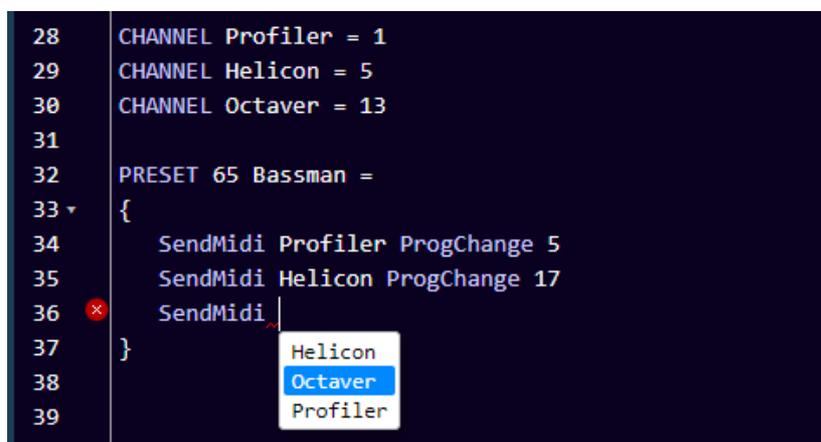
Nun beginnt der Hauptteil des Setups, der alle Details enthält, welche MIDI-Befehle jedes Preset senden wird. Die nächsten Unterthemen decken jeden der oben aufgeführten Setup-Teile ab. Das eigentliche Format der Befehle, die in jeder der Preset-Definitionen enthalten sein werden, wird in späteren Kapiteln beschrieben.

3.1. Definieren der MIDI-Kanäle

MIDI-Befehle können auf 16 verschiedenen Kanälen gesendet werden. In der Regel 'lauscht' jedes Gerät in der MIDI-Kette auf seinem spezifischen Kanal, so dass Sie mehrere Geräte gleichzeitig steuern können. Um das Setup zu vereinfachen und auch um das Setup lesbarer zu machen, erhält jeder verwendete MIDI-Kanal im Setup einen Namen. Diese MIDI-Kanaldefinitionen müssen die ersten Anweisungen in diesem Teil des Setups sein.

Sobald Sie einen Namen für alle MIDI-Kanäle, die Sie verwenden möchten, festgelegt haben, zeigt der Setup-Editor jedes Mal, wenn ein MIDI-Kanal angegeben werden muss, ein Dropdown-Feld mit diesen Namen an:

```
28 CHANNEL Profiler = 1
29 CHANNEL Helicon = 5
30 CHANNEL Octaver = 13
31
32 PRESET 65 Bassman =
33 {
34     SendMidi Profiler ProgChange 5
35     SendMidi Helicon ProgChange 17
36     SendMidi
37 }
38
39
```

The image shows a screenshot of a MIDI setup editor with a dark background. It displays a list of MIDI channel definitions and a preset configuration. Lines 28-30 define channels for 'Profiler' (1), 'Helicon' (5), and 'Octaver' (13). Line 32 starts a preset named '65 Bassman'. Lines 33-37 show a block of MIDI commands: 'SendMidi Profiler ProgChange 5', 'SendMidi Helicon ProgChange 17', and 'SendMidi'. A dropdown menu is open next to the 'SendMidi' command on line 36, showing three options: 'Helicon', 'Octaver' (which is highlighted in blue), and 'Profiler'. Line 38 is empty, and line 39 ends the preset definition.

Ein großer Vorteil dieses Ansatzes besteht auch darin, dass Sie den MIDI-Kanal, auf dem ein bestimmtes Gerät 'lauscht', sehr einfach ändern können. Passen Sie einfach die MIDI-Kanalnummer in der Kanaldefinition an, und der neue Kanal wird in allen MIDI-Befehlen für dieses Gerät in Ihrem gesamten Setup verwendet.

Achtung :

Diese Art der Definition von MIDI-Kanälen zu Beginn Ihres Setups ist nicht nur nützlich, sondern auch erforderlich. Sie können die von jedem Preset zu sendenden MIDI-Befehle nicht festlegen, solange Sie die zu verwendenden MIDI-Kanäle nicht benannt haben.

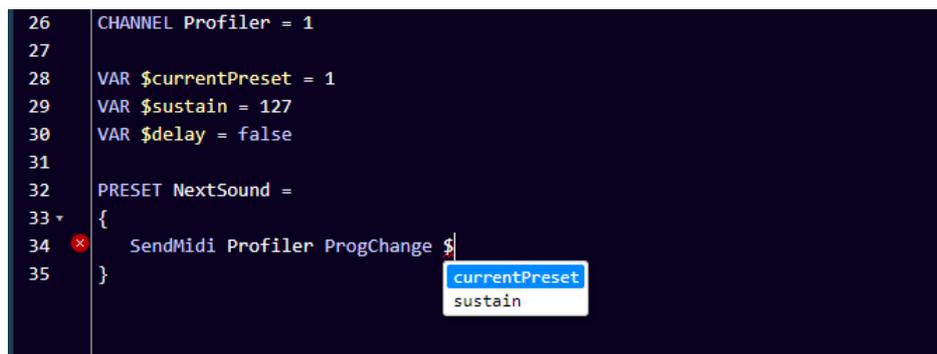
3.2. Definieren der Daten-Variablen

Die Verwendung von Datenvariablen in Ihrem Setup ist definitiv eine optionale erweiterte Funktion. Solange Sie nicht beabsichtigen, diese zu verwenden, können Sie diesen Teil der Dokumentation getrost überspringen.

Nachdem Sie, wie im vorigen Thema beschrieben, die verwendeten MIDI-Kanäle angegeben haben, geben Sie nun (optional) jede der Datenvariablen an, die Sie verwenden möchten. Ein Variablenname beginnt immer mit '\$'. Geben Sie der Variablen einen Anfangswert, so kann der Setup-Compiler erkennen, welches Datenformat die Variable enthalten wird: einen numerischen Wert (zwischen 0 und 127), einen booleschen Wert (wahr oder falsch) oder einen String-Wert (beliebiger Text zwischen Anführungszeichen):

```
VAR $currentPreset = 56
VAR $delay = false
VAR $currentSong = "Go with the flow"
```

Eine numerische Variable kann im Setup weit verbreitet sein: Da sie jeden Wert zwischen 0 und 127 enthalten kann, kann sie als Teil einer beliebigen MIDI-Nachricht direkt einen "hartkodierte" Wert ersetzen. Wann immer Sie normalerweise einen Wert zwischen 0 und 127 eingeben würden, können Sie "\$" eingeben, und der Editor schlägt eine Dropdown-Liste mit allen verfügbaren numerischen Datenvariablen vor:



The screenshot shows a code editor with a dark background. The code is as follows:

```
26 CHANNEL Profiler = 1
27
28 VAR $currentPreset = 1
29 VAR $sustain = 127
30 VAR $delay = false
31
32 PRESET NextSound =
33 {
34 SendMidi Profiler ProgChange $
35 }
```

A dropdown menu is open over the '\$' character on line 34, showing two options: 'currentPreset' and 'sustain'.

Auf die Möglichkeiten bei der Behandlung der "variablen Befehle" und der "bedingten Befehle" werden wir später in diesem Handbuch noch näher eingehen.

Es gibt eine vordefinierte numerische Variable mit dem Namen **\$MidiChannel (*)**

Sie können diese Variable in jedem Befehl verwenden, der die Angabe eines MIDI-Kanals erfordert, zum Beispiel :

```
SendMidi $MidiChannel ProgChange 13
```

Durch Ändern des Wertes der MidiChannel-Variable können Sie verschiedene Geräte mit demselben Befehlssatz ansprechen. In einer Bank könnten Sie zum Beispiel 8 Schalter programmieren, um ein Preset auszuwählen, und 2 Schalter um festzulegen, auf welchem Soundmodul dieses Preset aktiviert werden soll, was zu einer Auswahl von 16 verschiedenen Presets mit nur 10 Schaltern führt.

(*) verfügbar in Firmware v.1.3 oder höher

3.3. Definition des Anfangszustands des FCB1010

Vielleicht möchten Sie ja einige globale Einstellungen direkt nach dem Einschalten des FCB1010 initialisieren. Ein typisches Beispiel hierfür könnte z.B. das Standardverhalten der FCB1010 Expression-Pedale sein. Die Befehle, die während der Initialisierung des FCB1010 ausgeführt werden sollen, können mit der folgenden Syntax angegeben werden:

```
INIT_FCB =
{
    // irgendein Befehl zum Setzen eines Anfangszustands
    // ...
}
```

Es ist natürlich auch möglich, hier MIDI-Befehle anzugeben, die zur Initialisierung der verschiedenen an den FCB1010 angeschlossenen Geräte gesendet werden sollen. Es ist klar, dass es in diesem Fall notwendig ist, den FCB1010 als letzte Komponente in Ihrem Rigg mit Strom zu versorgen, oder aber ein schnelles Aus-Ein-Schalten des FCB1010 durchzuführen, sobald alle anderen Geräte angeschlossen und in Betrieb sind.

3.4. Definieren der Bank-Initialisierung

Es ist möglich, einen Satz MIDI-Befehle jedes Mal zu senden, wenn eine bestimmte Bank ausgewählt wird, d.h. wenn ein Klick auf den Up- oder Down-Taster des FCB1010 die neue Bank auswählt.

Das Format für die Angabe von Bankinitialisierungsbefehlen ist wie folgt:

```
INIT_BANK looper =
{
    // irgendein Bankinitialisierungsbefehl
    // ...
}
```

Bemerkung :

Ab Firmware-Version v.1.3 ist es möglich, Bankinitialisierungsbefehle auch für die Direct Bank zu geben.

3.5. Definieren der Preset-Inhalte

In vielen Fällen kann ein Preset sehr einfach sein: mit einem einzigen MIDI-ProgChange-Befehl kann beispielsweise ein bestimmtes Patch oder ein bestimmter Sound in einem Effektmodul ausgewählt werden. In diesem Fall kann der Preset-Inhalt auf einer einzigen Zeile im Setup beschrieben werden:

```
PRESET Vox Ac-30 = SendMidi Profiler ProgChange 2
```

In anderen Fällen kann ein Preset komplexer sein und mehrere MIDI-Befehle an mehrere Geräte senden. In diesem Fall werden diese Befehle auf mehreren Zeilen angegeben und mit geschweiften Klammern umgeben:

```
PRESET 65_Bassman =  
{  
    SendMidi Profiler ProgChange 5  
    SendMidi Helicon ProgChange 17  
    SendMidi Octaver CtrlChange 13 127  
}
```

Machen Sie dasselbe für jedes Preset in der Presetliste. Wenn Sie keinen Inhalt für ein Preset angeben, wird sich der Setup-Compiler nicht beschweren, aber offensichtlich wird nichts passieren, wenn dieses Preset mit den FCB1010 Preset-Tastern ausgewählt wird.

Optional kann ein Preset auch MIDI-Nachrichten beim Loslassen des Tasters senden, obwohl dieses Verhalten eher für "Trigger" geeignet ist, die genau dafür ausgelegt sind (siehe unten). Wollen Sie, daß ein Preset beim Loslassen des Tasters Nachrichten sendet, verwenden Sie die folgende Syntax :

```
PRESET_RELEASE Vox Ac-30 = SendMidi Profiler CtrlChange 23 100
```

3.6. Definieren der Effekt-Inhalte

Wie in einem früheren Kapitel erläutert, besteht der Hauptunterschied zwischen Effekten und Voreinstellungen darin, dass Effekte zwischen 2 Zuständen umschalten können: EIN und AUS. Für jeden dieser 2 Zustände müssen Sie den/die zu sendenden MIDI-Befehl(e) angeben. Auch hier wird in vielen Fällen das Senden eines einzigen MIDI-Befehls ausreichen, um einen Effekt zu (de)aktivieren, aber Sie können sie so komplex gestalten, wie Sie wollen, indem Sie folgende Syntax verwenden:

```
EFFECT_ON Chorus = SendMidi Strymon CtrlChange 3 127
```

```
EFFECT_OFF Chorus = SendMidi Strymon CtrlChange 3 0
```

```
EFFECT_ON Delay =  
{  
    // jegliche Kombination von mehreren Befehlen  
}
```

```
EFFECT_OFF Delay =  
{  
    // jegliche Kombination von mehreren Befehlen  
}
```

3.7. Definieren der Trigger-Inhalte

Ein Trigger (= "Auslöser") kann verwendet werden, um einen oder mehrere MIDI-Befehle auf das Klicken eines Fußtasters hin zu senden. Die Syntax unterstützt wiederum sowohl eine einzeilige als auch eine mehrzeilige Inhaltsdefinition :

```
TRIGGER_CLICK Overdub =SendMidi Looper NoteOn 15 127

TRIGGER_CLICK HalfSpeed =
{
  // jegliche Kombination von mehreren MIDI-Befehlen
}
```

Wie in einem früheren Kapitel erläutert, kann ein "Auslöser" auch zur Definition eines Momentan-Effekts verwendet werden, der beim Drücken des Tasters und beim Loslassen des Tasters eine jeweils unterschiedliche Nachricht aussendet. Beide Nachrichten (oder Nachrichtengruppen) können wie folgt spezifiziert werden:

```
TRIGGER_CLICK OctaveUp =SendMidi Octaver CtrlChange 15 127

TRIGGER_RELEASE OctaveUp =SendMidi Octaver CtrlChange 15 0

TRIGGER_CLICK PlayAmin7Chord =
{
  SendMidi Synth NoteOn 57 100
  SendMidi Synth NoteOn 60 100
  SendMidi Synth NoteOn 64 100
  SendMidi Synth NoteOn 67 100
  SendMidi Synth NoteOn 69 110
}

TRIGGER_RELEASE PlayAmin7Chord =
{
  SendMidi Synth NoteOff 57 127
  SendMidi Synth NoteOff 60 127
  SendMidi Synth NoteOff 64 127
  SendMidi Synth NoteOff 67 127
  SendMidi Synth NoteOff 69 127
}
```

3.8. Definieren der Sweep-Inhalte

Ein "Sweep" mag ein weniger intuitives Konzept in der Architektur der UnO2-Einrichtung sein. Es ist eine Möglichkeit, die Funktionalität eines Expression-Pedals zu beschreiben. Zuerst listen Sie die verschiedenen benötigten Funktionalitäten (Lautstärke, Wah, Ausdruck, ...) als mögliche "Sweeps" in Ihrem Setup auf. In der Sweep-Definition (siehe Syntax unten) geben Sie die für jede dieser Funktionen notwendigen MIDI-Befehle an. Sobald Sie dies festgelegt haben, kann ein einfacher Befehl jeden der Sweeps mit einem der beiden FCB1010 Expression-Pedale verbinden. Ein solcher Pedalzuweisungsbefehl (siehe Kapitel 5.1) kann zu jedem Preset oder Effekt hinzugefügt werden. Dies ermöglicht eine sehr flexible Verwendung der Expression-Pedale: sie können je nach aktuellem Kontext eine unterschiedliche Funktion erfüllen.

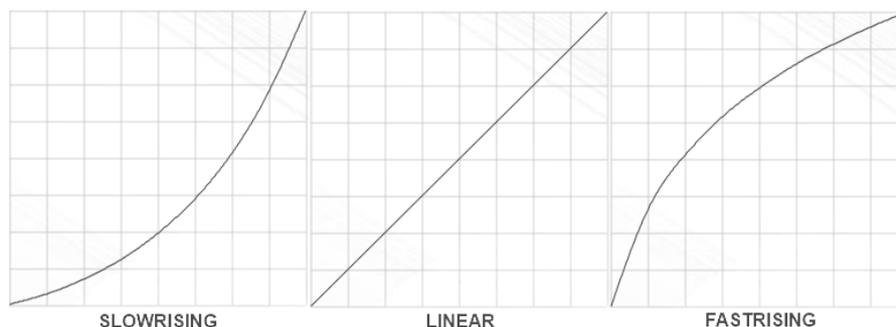
Die Syntax zur Definition von Sweep-Inhalten dürfte inzwischen vertraut aussehen:

```
SWEEP whammy = SendMidi DX7 PitchBend 0-127
```

```
SWEEP volume = SendMidi Profiler CtrlChange 7 0-127 SlowRising
```

Die Sweep-Definition kann nur "kontinuierliche Steuerbefehle (continuous control)" enthalten: Dies sind die MIDI-Befehle, die zur kontinuierlichen Parametereinstellung verwendet werden: "CtrlChange", "PitchBend" oder "ChannelPressure".

Standardmäßig ist die Sweepkurve beim Bewegen eines Expressionspedals linear. Sie können jedoch auch festlegen, dass es ein Sweep "SlowRising" oder "FastRising" sein soll. Ein analoges Lautstärkenpedal zum Beispiel verwendet typischerweise ein "Log(arithmetisches) Potentiometer" (manchmal auch "Audio-Potentiometer" genannt). Dieses Potentiometer erhöht die Lautstärke zu Beginn einer Dreh- bzw. Pedalbewegung langsamer (flacher) und am Ende schneller (steiler). Sie können dieses Verhalten mit Ihrem digitalen Expression-Pedal FCB1010 nachahmen, indem Sie ein "SlowRising"-Verhalten für diesen Sweep festlegen:



4. Der Befehlsatz

Im vorigen Kapitel haben wir die Syntax für die Angabe des Inhalts eines Presets, Effekts, Sweeps usw. beschrieben. Nun werden wir näher auf die eigentlichen Befehle eingehen, die in diesen Preset-Inhalten verwendet werden können. Natürlich wird der MIDI-Befehl der am häufigsten verwendete Befehlstyp sein, wahrscheinlich werden mehr als 90% Ihres Setups aus MIDI-Befehlen bestehen. Die UnO2-Firmware bietet jedoch viel mehr als nur die grundlegende MIDI-Funktionalität eines regulären FCB1010. In den nächsten Unterkapiteln werden die verschiedenen in einem UnO2-Setup unterstützten Befehlstypen behandelt:

- Fußtaster- und Pedalzuweisungsbefehle
- Effekt- und Relaisaktivierungsbefehle
- MIDI-Befehle
- Delay (= Verzögerungs)-Befehle
- Continuous Control Befehle
- Variablen-Befehle
- Bedingte Befehle

4.1. Fußtaster- und Pedalzuweisungsbefehle

SYNTAX :

```
Footswitch [1...10] = preset/effect/trigger-name/"nothing"  
Tipswitch [1...2] = preset/effect/trigger-name/"nothing"  
Heelswitch [1...2] = preset/effect/trigger-name/"nothing"  
Pedal [1...2] = sweep-name/"nothing"
```

Im Normalfall werden den FCB1010-Fußschaltern über das Bank-Setup, das in einem vorherigen Kapitel besprochen wurde, Presets zugewiesen. Für jede Bank legen Sie fest, welches Preset (oder Effekt oder Trigger) jedem der 10 Fußschalter zugewiesen wird. Es ist jedoch möglich, dieses vordefinierte Bank-Layout vorübergehend zu modifizieren, indem Sie Fußschalter-Zuweisungsbefehle zu einem Preset-Inhalt hinzufügen. Auf diese Weise können Sie ein sehr "dynamisches" Bank-Layout erstellen, z.B. die untere Reihe mit 5 Presets und die obere Reihe mit bis zu 5 Effekten, die je nach ausgewähltem Preset unterschiedlich sein können:

```
PRESET Chieftain =  
{  
    Footswitch 6 = BOOST  
    Footswitch 7 = Compressor  
    Footswitch 8 = Fxloop  
    // MIDI commands ...  
}  
  
PRESET Matchless =  
{  
    Footswitch 6 = Delay  
    Footswitch 7 = Flanger  
    Footswitch 8 = Chorus  
    // MIDI commands ...  
}
```

Pedalzuordnungen sind notwendig, um die Funktion für jedes der 2 Expressionspedale zu definieren. Wenn Sie ein festes Pedal-Setup wünschen (z.B. : linkes Pedal = Lautstärke, rechtes Pedal = Wah), können Sie die Pedalzuordnung in der zuvor erwähnten Initialisierung des FCB1010 festlegen:

```
INIT_FCB =  
{  
    Pedal 1 = Volume  
    Pedal 2 = Wah  
}
```

"Volume" und "Wah" sind in diesem Beispiel Sweeps, welche zuvor im Setup definiert wurden. Um Pedal 2 zu deaktivieren, spezifizieren Sie einfach: **Pedal 2 = nothing**

Natürlich können, genau wie bei Fußtasterzuweisungen, auch Pedalzuweisungsbefehle Teil jedes Preset-Inhalts sein. Auf diese Weise kann das Pedalverhalten sehr dynamisch sein und sich zusammen mit dem aktuell ausgewählten Preset ändern.

Neben den 10 Fußschaltern können Sie auch 2 virtuellen "Fußspitzen-Schaltern" und "Fersenschaltern", einem für jedes Expression-Pedal, ein Preset, einen Effekt oder einen Trigger zuweisen. Eine optionale, aber sehr leistungsfähige Funktion. Vielleicht haben Sie schon professionelle Expression-Pedale gesehen, die einen Fußschalter an der Unterseite montiert haben. Die Pedale verhalten sich wie reguläre Lautstärke- oder Expressionspedale, aber wenn Sie sie ganz nach unten drücken, können Sie auch den darunter liegenden Schalter betätigen, um eine Änderung auszulösen. Diese Pedale benötigen in der Regel 2 separate Klinkenkabel, die zu Ihrem Gerät führen. Mit der UnO2-Firmware können Sie dasselbe Verhalten mit den regulären FCB1010-Expression-Pedalen erreichen!

Sobald Sie dem "virtuellen Tipschalter 1" ein Preset oder einen Effekt zuweisen, können Sie dieses Preset auslösen, indem Sie das Expressionspedal 1 ganz nach unten drücken. Sie könnten z.B. einen Effekt mit dem Tipschalter verknüpfen, der das Verhalten des Expressionspedals zwischen 2 Funktionen umschaltet. Sie brauchen keinen separaten Fußschalter zu opfern, um das Verhalten des Pedals zu ändern, Sie können es mit dem Pedal selbst tun!

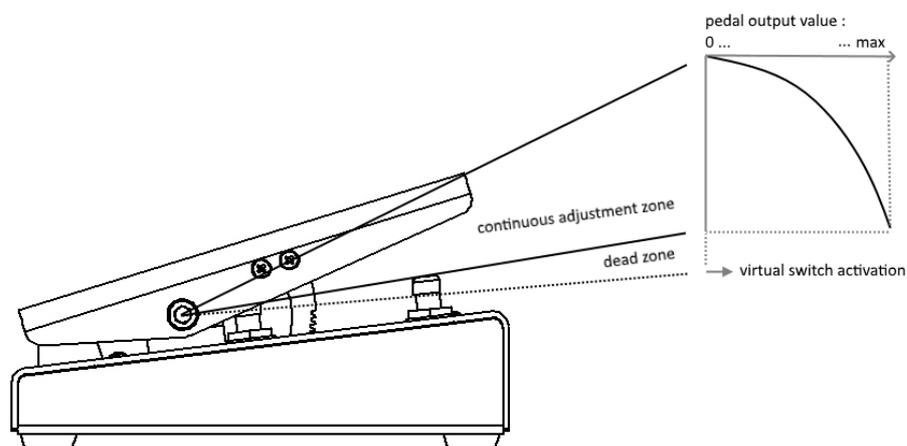
In ähnlicher Weise kann ein "virtueller Fersenschalter" definiert werden. Dieser wird ausgelöst, indem das Expressionspedal ganz nach oben (Ferse nach unten) bewegt wird. Eine Funktionalität, die es wahrscheinlich noch nicht einmal bei "echten" Fußschaltern gibt. Ein großartiges Beispiel dafür, wie man von dieser Funktion Gebrauch machen kann, ist das folgende Setup-Fragment, das das Stimmgerät automatisch aktiviert, sobald Sie die Lautstärke Ihrer Gitarre auf 0 herunterdrehen:

```
INIT_FCB =
{
    Pedal 1 = volume
    Heelswitch 1 = Tuner
}

TRIGGER_CLICK Tuner = SendMidi Eventide CtrlChange 99 127
TRIGGER_RELEASE Tuner = SendMidi Eventide CtrlChange 99 0
```

Anmerkung :

Wenn Sie einen Spitzen- oder Fersenschalter spezifizieren, führt die Firmware automatisch eine kleine "tote Zone" im Pedalbereich ein, so dass Sie den vollen Einstellbereich des Pedals nutzen können, ohne den Spitzen- (oder Fersen-) Schalter automatisch auszulösen. In diesem Fall ist es sehr hilfreich, dem Pedal an einem Punkt zwischen dem vollen Einstellbereich und der Schalterbetätigung einen gewissen Widerstand hinzuzufügen. Dies kann erreicht werden, indem ein Stück Schaumstoff unter das Pedal geklebt wird oder sogar ein echter (nicht angeschlossener) Fußschalter montiert wird.



4.2. Befehle zur Aktivierung von Effekten, der Relais und der Expressionpedale

SYNTAX :

```
SwitchOneffectname  
SwitchOffeffectname  
SendTrigger triggername  
Close Relay1  
Open Relay1  
Close Relay2  
Open Relay2
```

Einige einfache, aber mächtige Befehle: Wenn Sie eine bestimmte Voreinstellung auswählen, können Sie eine Reihe von Effekten zusammen mit ihr aktivieren. Wenn Sie einem Effekt einen Fußschalter zugewiesen haben, wird beim Aktivieren des Effekts mit diesem Befehl auch die LED des Fußschalters eingeschaltet, um den aktuellen Status des Effekts anzuzeigen.

Neben diesen Effekttaktivierungsbefehlen gibt es auch einen Triggeraktivierungsbefehl, mit dem alle Nachrichten eines bestimmten Triggers gesendet werden können, wenn Sie ein bestimmtes Preset anwählen.

Die Befehle "Open Relay" (Relaiskontakt geöffnet) und "Close Relay" (Relaiskontakt geschlossen) sind selbsterklärend. Der FCB1010 enthält 2 Buchsenausgänge, die von zwei Relais gesteuert werden. Diese können z.B. zur Steuerung eines Verstärkers verwendet werden, der nicht MIDI-fähig ist.

Ein "SendPedal"-Befehl (*) kann verwendet werden, um den mit der aktuellen Position eines Expression-Pedals verbundenen Befehl erneut zu senden. Bei einigen Geräten kann es z.B. notwendig sein, die aktuelle Lautstärke einzustellen, nach dem Wechsel des aktiven Presets. Durch das Senden dieses Befehls wird vermieden, dass Sie das Lautstärkepedal ein wenig bewegen müssen, um die zuvor aktive Lautstärkeinstellung wiederherzustellen.

(*) verfügbar in Firmware v. 1.3 oder höher

Anmerkung :

Wenn Sie die Befehle SwitchOn, SwitchOff oder SendTrigger in einem Effekt oder Trigger-Inhalt nutzen, riskieren Sie, eine "Endlosschleife" zu erzeugen. Das einfachste Beispiel ist, wenn Sie Folgendes angeben würden:

```
EFFECT_ON effect 1 = SwitchOn effect 1
```

Es liegt auf der Hand, dass diese Codezeile dazu führen würde, dass das FCB1010 blockiert wird und immer wieder derselbe Befehl ausgeführt wird. Diese Art von Schleife ist nicht immer so leicht zu erkennen wie im obigen Beispiel. Sie kann durch eine viel längere "Kette" von Befehlen zur Aktivierung von Effekten verursacht werden und schließlich in einem Effekt enden, der sich durch die Aktivierung anderer Effekte oder Auslöser wieder "selbst aktiviert". Glücklicherweise analysiert der Editor im UnO2-Kontrollzentrum Ihr Setup, während Sie die Befehle zur Aktivierung der Effekte eingeben, und der Code-Compiler gibt eine Fehlermeldung aus, wenn eine Endlosschleife erkannt wird.

4.3. Navigation und Fernbedienung

SYNTAX :

```
GotoBank bankname
```

Verwenden Sie diesen Befehl in einem Preset-Inhalt, um mit einem Fußklick direkt zu einer bestimmten Bank zu springen, anstatt mit den Auf-/Ab-Schaltern durch die gesamte Bankliste zu scrollen.

SYNTAX :

```
REMOTE_CONTROL_CHANNEL = nn
```

Wenn Sie diesen Befehl **vor** den anderen MIDI-Kanaldefinitionen Ihres Setups hinzufügen, können Sie das FCB1010 von einem anderen MIDI-Gerät aus fernsteuern, das an MIDI IN angeschlossen ist.

Folgende ControlChange- und ProgramChange-Befehle können auf dem angegebenen Fernsteuerungs-MIDI-Kanal gesendet werden, um das FCB1010 mit "Schalterdrücken" und "Pedalbewegungen" fernzusteuern:

```
ProgramChange 00-09 = click/release footswitch 1-10  
ProgramChange 14-15 = click/release Down/Up footswitch  
ControlChange 00/nn = goto bank nn  
ControlChange 04/07 value = move left/right expr.pedal
```

Diese Befehle sind in Firmware v.1.3 oder höher verfügbar

4.4. MIDI-Befehle

SYNTAX :

```
SendMidi channelname ProgChange value
SendMidi channelname CtrlChange value value
SendMidi channelname NoteOn      value value
SendMidi channelname NoteOff    value value
SendSysEx F0 ... F7
SendSysCommon SongSelect value
SendSysCommon SongPosition 14-bit-value
SendSysCommon TuneRequest
SendRealtime MIDISTart
SendRealtime MIDIContinue
SendRealtime MIDISTop
SendRealtime SystemReset
```

Da der FCB1010 in erster Linie ein MIDI-Controller ist, wird der "MIDI-Befehl" zweifellos der am häufigsten verwendete Befehlstyp in einem UnO2-Setup sein. Eine der Stärken von UnO2 ist die Tatsache, dass Sie eine beliebige Anzahl von MIDI-Befehlen, die auf verschiedenen MIDI-Kanälen gesendet werden, zu einem einzigen Preset kombinieren können. Alle diese Befehle werden mit einem einzigen Fußklick gleichzeitig gesendet.

Die obige Syntax ist selbsterklärend: *channelname* ist einer der MIDI-Kanalnamen, die zu Beginn des Setups festgelegt wurden (siehe Kapitel 4.1), *value* ist ein beliebiger numerischer Wert zwischen 0 und 127. Wie bereits erwähnt, können Sie auch eine numerische Variable verwenden, anstatt einen Wert in einer beliebigen MIDI-Nachricht (sogar im Inhalt einer SysEx-Nachricht!) anzugeben.

Im Prinzip ist die Länge einer SysEx-Meldung nur durch die Speichergröße des FCB1010 begrenzt. In einem realistischen Anwendungsfall wird dieser Befehl jedoch verwendet, um eher kurze SysEx-Meldungen zur Effekt- oder Preset-Parameteranpassung zu senden. Die Werte werden in hexadezimaler Notation geschrieben, beginnend mit F0, endend mit F7 und mit den Werten 00-7F zwischen dem Start- und Stopbyte.

4.5 Continuous Control-Befehle

SYNTAX :

```
SendMidi channel CtrlChange value from-to [SlowRising/FastRising]
SendMidi channel PitchBendvalue from-to [SlowRising/FastRising]
SendMidi channel ChannelPressurevalue from-to [SlowRising/FastRising]
```

Die Befehle "Continuous Control" ("Kontinuierliche Steuerung") werden zur Angabe eines Sweep-Inhalts verwendet. Sie beschreiben, welche Meldungen gesendet werden, wenn eines der Expression-Pedale des FCB1010 bewegt wird.

Bei den meisten (wenn nicht allen) MIDI-Controllern können Expression-Pedale spezifische ControlChange-Befehle senden. Die am häufigsten verwendeten Werte für die kontinuierliche Steuerung sind :

- CC 07 = Volume (Lautstärke)
- CC 04 = Foot Controller (Fußschalter)
- CC 01 = Modulation Wheel (Modulations-Rad)
- CC 11 = Expression Controller (Expression-Kontroller)
-

Der übertragene Wertebereich für diese MIDI-Nachrichten ist 0 (Ferse nach unten) bis 127 (Spitze nach unten).

Mit der UnO2 firmware haben Sie einige leistungsstarke Zusatzoptionen erhalten, die Sie wahrscheinlich in keinem anderen MIDI-Controller finden werden:

- neben CtrlChange-Nachrichten ist es auch möglich, PitchBend oder ChannelPressure-Nachrichten zu senden.
- der Sweep-Bereich ist vollständig anpassbar
- können Sie die Sweepkurve festlegen, der das Pedal folgen soll : linear, slow rising (langsam steigend) oder fast rising (schnell steigend) (siehe Kapitel 3.8 für weitere Details)

Wenn Sie den Befehl SendSysEx (*) verwenden, können Sie eine beliebige Länge der SysEx-Nachricht angeben und dabei das Wort "**value**" an einer beliebigen Stelle innerhalb der SysEx-Nachricht einfügen. Die aktuelle Position des Expression-Pedals (Wert zwischen 0 und 127) wird in die SysEx-Nachricht an der Stelle des Wortes "**value**" eingefügt.

() verfügbar in Firmware v.1.3 oder höher*

4.6. Verzögerungs-Befehl

SYNTAX :

```
Wait value
```

Dieser Befehl kann zwischen MIDI-Befehle eingefügt werden, um eine kleine Pause zwischen den Befehlen einzufügen. Es hat sich gezeigt, dass einige Geräte unzuverlässig reagieren, wenn mehrere MIDI-Befehle bei voller Geschwindigkeit an sie gesendet werden. Ein anderer Anwendungsfall könnte das Abspielen kurzer MIDI-Samples beim Anklicken eines Fußschalters sein, indem eine Reihe von NoteOn/NoteOff-Befehlen angegeben wird, getrennt durch die notwendigen Verzögerungsbefehle, um jeden Akkord für die gewünschte Dauer zu halten. Obwohl wir zugeben müssen, dass dies einige mühsame Programmierung erfordern würde...

Der Verzögerungswert kann zwischen 1 und 127 liegen und wird in 100ms-Einheiten ausgedrückt. Das bedeutet, dass ein Befehl `Wait 10` eine Verzögerung von 1 Sekunde einführt und die maximal mögliche Verzögerung 12,7 Sekunden beträgt. Beachten Sie, dass eine solche Verzögerung "blockierend" ist: Die Firmware kann während der Ausführung des Befehls `Wait` keinen Schalterdruck oder keine Pedalbewegung verarbeiten oder irgendetwas anderes tun!

4.7. Variablen-Befehl

Wir haben in einem früheren Kapitel erwähnt, wie Sie numerische, boolesche oder String-Variablen in Ihrem Setup definieren können. Natürlich macht die Verwendung von Variablen nur dann Sinn, wenn Sie die Möglichkeit haben, ihre Werte bei der Auswahl einer bestimmten Voreinstellung oder eines bestimmten Effekts zu ändern und dann auf diese geänderten Werte zu reagieren. Das können Sie mit den Variablen-Befehlen dieses Kapitels und den bedingten Befehlen des nächsten Kapitels tun.

SYNTAX :

```
$intvarname = value
$intvarname = $intvarname2
$intvarname = $intvarname2 [+ -] value
$intvarname [++ --]
$intvarname [+= -=] value
$boolvarname = [true/false]
$boolvarname = $boolvarname2
$boolvarname = !$boolvarname2
$stringvarname = "any string"
```

Eine numerische Variable kann auf einen beliebigen Wert zwischen 0 und 127 gesetzt werden. Sie kann inkrementiert (= *erhöht*) (++) oder dekrementiert (= *verringert*) (--) werden, ein fester Wert kann addiert (+=) oder subtrahiert (-=) werden, oder es kann der Wert einer anderen numerischen Variablen genommen und modifiziert werden (= \$intvarname2 +/- Wert)

Eine boolesche Variable kann auf wahr oder falsch gesetzt werden, kann den Wert einer anderen booleschen Variable übernehmen oder ihr Wert kann von wahr auf falsch und umgekehrt invertiert werden (unter Verwendung des Symbols "!")

Einer String-Variablen kann einfach ein beliebiger Textwert zugewiesen werden. Der Inhalt der Variablen kann dann mit einem der bedingten Befehle des nächsten Kapitels überprüft werden.

4.8. Bedingte Befehle

SYNTAX :

```
if (condition*) {
    ...
}
else if (condition*) {
    ...
}
else {
    ...
}

while(condition*) {
    ...
}

* condition : $intvarname      [ >>= == != <=< ] [0...127]
              $intvarname      [ >>= == != <=< ] $intvarname2
              $stringvarname    [ == != ]      "any string"
              $boolvarname      [ == != ]      $boolvarname2
              $boolvarname
              !$boolvarname

switch($intvarname) {
    case[0...127]:
        ...
        break
    ...
    default:
        ...
        break
}

switch($stringvarname) {
    case "any string":
        ...
        break
    ...
    default:
        ...
        break
}
```

Sie können Ihr Setup sehr "dynamisch" machen, indem Sie dasselbe Preset je nach Bedingung einen anderen Sound auswählen oder einen anderen Effekt aktivieren lassen. Datenvariablen werden verwendet, um diese Bedingungen einzustellen, und die oben genannten bedingten Befehle werden verwendet, um die notwendigen Entscheidungen zu treffen.

Die 'while'-Anweisung kann verwendet werden, um Schleifen zu erzeugen, z.B. um einen Ein-/Ausblendungseffekt zu erzielen (siehe "Beispiel 5" in einem früheren Kapitel als ein Beispiel dafür)

Natürlich muss die while-Anweisung mit Vorsicht verwendet werden, um sicherzustellen, dass Sie keine Endlosschleife in Ihrem Setup erzeugen. Dies würde das Gerät veranlassen, nicht mehr zu reagieren.

4.8.1. Die Bedingungs-Syntax

Alle bedingten Befehle prüfen den Wert einer Datenvariablen, um festzustellen, ob eine bestimmte Bedingung erfüllt ist. Die möglichen Prüfungen sind :

- für numerische Variablen :

```
$var>nn           // ist größer als  
$var>= nn        // ist größer als oder gleich  
$var== nn        // ist gleich  
$var!= nn        // ist nicht gleich  
$var<= nn        // ist kleiner als oder gleich  
$var< nn         // ist kleiner als  
                // nn ein Wert zwischen 0 und 127,  
                // oder eine andere numerische Variable
```

- für String (= Text-) Variablen :

```
$var == "any string"  
$var != "any string"
```

- für boolean Variablen :

```
$var1==$var2     // $var1 und $var2 sind beide wahr oder  
                // falsch  
$var1!=$var2     // $var1 unterscheidet sich von $var2  
$var             // $var is wahr  
!$var           // $var is falsch
```

Bei Bedarf können mehrere Bedingungen kombiniert werden:

(**&&** meint "und", **||** meint "oder")

```
// alle der folgenden Bedingungen müssen erfüllt sein:  
((condition1) && (condition2) && ... )  
  
// mindestens eine der folgenden Bedingungen muss erfüllt sein:  
((condition1) || (condition2) || ... )
```

4.8.2. if...then...else-Anweisungen

Die Syntax für eine if...then...else... Art der Prüfung ist:

```
if (condition) {
    // beliebige Anzahl von Befehlen...
}
else if (another condition) {
    // beliebige Anzahl von Befehlen...
}
else if (yet another condition) {
    // beliebige Anzahl von Befehlen...
}
else {
    // wenn keine der Bedingungen zutrifft
    // führe die Befehle in diesem Segment aus
}
```

Sie können sogar verschachtelte bedingte Anweisungen haben (obwohl wir nicht glauben, dass ein UnO2-Setup so viel Komplexität erfordert...):

```
if (condition) {
    if (subcondition) {
        ...
    }
    else {
        ...
    }
}
else {
    ...
}
```

Wenn Sie es vorziehen, können Sie aus Gründen der Übersichtlichkeit auch alle geschweiften Klammern auf eine neue Zeile setzen:

```
if (condition)
{
    // beliebige Anzahl von Befehlen...
}
```

Andererseits sind die geschweiften Klammern nicht unbedingt erforderlich, wenn sie nur 1 Befehl umgeben:

```
if ($delay)
    SendMidi MyGear CtrlChange 112 127
else
    SendMidi MyGear CtrlChange 112 0
```

4.8.3. while-Anweisungen

Die while-Anweisung hat eine identische Syntax wie die if-Anweisung :

```
while (condition) {  
    // beliebige Anzahl von Befehlen...  
}
```

4.8.4. switch-Anweisungen

Eine Schaltanweisung ist eine Abkürzung für eine lange Reihe von if-Anweisungen und wird verwendet, um eine Variable gegen eine größere Reihe von verschiedenen möglichen Werten zu prüfen.

- für eine numerische Variable :

```
switch($currentPreset) {  
    case 2:  
        // beliebige Anzahl von Befehlen  
        break  
    case 15:  
        // beliebige Anzahl von Befehlen  
        break  
    case 123:  
        // beliebige Anzahl von Befehlen  
        break  
    default:  
        // wenn keiner der oben genannten Werte  
        // übereinstimmt,  
        // führe die Befehle in diesem Segment aus  
        break  
}
```

- für eine String-Variablen :

```
switch($currentSong) {  
    case "Go with the flow":  
        // beliebige Anzahl von Befehlen  
        break  
    case "No one knows":  
        // beliebige Anzahl von Befehlen  
        break  
    case "Do it again":  
        // beliebige Anzahl von Befehlen  
        break  
    default:  
        // wenn keiner der oben genannten Werte  
        // übereinstimmt,  
        // führe die Befehle in diesem Segment aus  
        break  
}
```

4.8.5. Bedingungen mit vordefinierten Variablen

Die Programmiersprache UnO2 stellt einige vordefinierte Variablen zur Verfügung, die in bedingten Befehlen verwendet werden können. Diese vordefinierten Variablen sind :

```
#CURRENT_BANK
#CURRENT_PRESET
EFFECT_ON "effectname"
EFFECT_OFF "effectname"
```

Im vorangegangenen Beispiel über die switch-Anweisung könnten Sie beispielsweise typischerweise diese Variablen verwenden, anstatt eine eigene \$songName-Variable zu erstellen, die Sie selbst mit dem aktuell aktiven Songnamen füllen müssten:

```
switch(#CURRENT_BANK) {
    ...
}

if(EFFECT_ON "Delay") {
}
```

Diese Funktion ist in Firmware v.1.3 oder höher verfügbar.

Ein kleines Beispiel für die bedingte Logik in Aktion: mit den wenigen Zeilen Code unten können Sie 2 Fußschalter so programmieren, dass sie durch alle Sounds Ihres Synthesizers oder Modelers blättern:

```
119 CHANNEL Profiler = 1
120
121 VAR $currentPreset = 1
122
123 PRESET Next sound =
124 {
125   if($currentPreset < 100) {
126     $currentPreset++
127   }
128   else {
129     $currentPreset = 1
130   }
131   SendMidi Profiler ProgChange $currentPreset
132 }
133
134 PRESET Previous sound =
135 {
136   if($currentPreset > 1) {
137     $currentPreset--
138   }
139   else {
140     $currentPreset = 100
141   }
142   SendMidi Profiler ProgChange $currentPreset
143 }
```



Und jetzt ist es Zeit, Spaß zu haben!



ANHANG : UnO2 Programmiersprachen-Referenz

Comments : **(Kommentare)**

```
// single-line comment : any text...

/*
multi-line comment :
any text...
*/
```

Defining presets, effects, triggers, sweeps and bank layout: **(Definieren von Presets, Effekten, Triggern, Sweeps und Bank-Layout:)**

```
PRESETS =
{
[preset name]
...
}

EFFECTS =
{
[effect name]
...
}

TRIGGERS =
{
[trigger name]
...
}

SWEEPS =
{
[continuous control name]
...
}

NO_UPDOWN_SWITCHES

GLOBALSWITCH [1...10] = [presetname]

USE_DIRECT_BANK

BANKS =
{
[bankname]: [preset name] | [preset name] | ... | [preset name]
[bankname]: [preset name] | [preset name] | ... | [preset name]
...
}
```

Defining preset content :
(Definieren von Preset-Inhalten)

```
CHANNEL channelname = [1...16]

VAR $intvarname = [0...127]
VAR $boolvarname = [true/false]
VAR $stringvarname = "any string"

INIT_FCB                = ... (single command, or list of commands between curly braces)
INIT_BANK bank= ... ""

PRESET preset           = ... ""
EFFECT_ON effect       = ... ""
EFFECT_OFF effect      = ... ""
TRIGGER_CLICK trigger  = ... ""
TRIGGER_RELEASE trigger = ... ""
SWEEP sweep            = ... (single continuous control command)
```

Commands :
(Befehle)

Dynamic switch and pedal assignment commands :
(Dynamische Schalt- und Pedalzuweisungsbefehle)

```
Footswitch [1...10] = preset/effect/trigger-name/"nothing"
Tipswitch   [1...2] = preset/effect/trigger-name/"nothing"
Heelswitch  [1...2] = preset/effect/trigger-name/"nothing"
Pedal       [1...2] = sweep-name/"nothing"
```

Effect activation commands :
(Effektaktivierungsbefehle)

```
SwitchOn   effectname
SwitchOff  effectname
SendTrigger triggername
SendPedal  [1...2]
```

MIDI Commands :
(MIDI-Befehle)

```
SendMidi channelname ProgChange value
SendMidi channelname CtrlChange value value
SendMidi channelname NoteOn   value value
SendMidi channelname NoteOff  value value
SendSysEx F0 ... F7
SendRealtime MIDISTart
SendRealtime MIDIContinue
SendRealtime MIDISTop
```

Continuous control commands :
(Kontinuierliche Steuerbefehle)

```
SendMidi channelname CtrlChange value [from-till][FastRising/SlowRising]
SendMidi channelname PitchBend      [from-till][FastRising/SlowRising]
SendMidi channelname ChannelPressure [from-till][FastRising/SlowRising]
SendSysEx F0 ... value ... F7
```

Delay command :
(Verzögerungs-Befehl)

```
Wait value      (expressed in 100ms units)
```

Variable Commands :
(Variablen-Befehle)

```
$intvarname = value  
$intvarname = $intvarname2  
$intvarname = $intvarname2 [+ -] value  
$intvarname [++ --]  
$intvarname [+ = - =] value  
$boolvarname = [true/false]  
$boolvarname = $boolvarname2  
$boolvarname = !$boolvarname2  
$stringvarname = "any string"
```

Conditional Commands :
(Bedingte Befehle)

```
if (condition*) {  
    ...  
}  
else if (condition*) {  
    ...  
}  
else {  
    ...  
}  
  
while (condition*) {  
    ...  
}  
  
switch($intvarname) {  
    case[0...127]:  
        ...  
        break  
    ...  
    default:  
        ...  
        break  
}  
  
switch($stringvarname) {  
    case "any string":  
        ...  
break  
    ...  
    default:  
        ...  
        break  
}  
  
* condition : $intvarname [ >>= == != <= < ] [0...127]  
               $intvarname [ >>= == != <= < ] $intvarname2  
               $stringvarname [ == != ] "any string"  
               $boolvarname [ == != ] $boolvarname2  
               $boolvarname  
               !$boolvarname  
               #CURRENT_BANK [ == != ] bankname  
               #CURRENT_preset [ == != ] presetname  
               EFFECT_ON effectname  
               EFFECT_OFF effectname
```

ANHANG : Werks-Reset, Selbsttest und Pedalkalibrierung

Die UnO2-Firmware enthält die gleichen Selbsttest- und Expressionspedal-Kalibrierverfahren wie die originale Behringer-Firmware. Kalibrierungsanweisungen finden Sie daher im Behringer-Handbuch oder online, indem Sie nach "FCB1010 calibration" googeln.

- Selbsttest : halten Sie die Fußtaster 1+3 während des Einschaltens gedrückt
- Kalibrierung : halten Sie die Fußtaster 1+5 während des Einschaltens gedrückt

Die in der Behringer-Firmware vorhandenen Werks-Resets sind **nicht mehr verfügbar**:

- V-AMP factory preset : halten Sie die Fußtaster **1+6** während des Einschaltens gedrückt
- Behringer guitar amp factory preset : halten Sie die Fußtaster **1+7** während des Einschaltens gedrückt
- BASS V-AMP factory preset : halten Sie die Fußtaster **1+8** während des Einschaltens gedrückt

Stattdessen gibt es eine Möglichkeit zum Zurücksetzen auf die Werkseinstellungen, wodurch das aktuelle UnO2-Setup **gelöscht** wird!:

- Werks-Reset: halten Sie die Fußtaster **1+4** während des Einschaltens gedrückt

© by Xavier De Donder

Deutsche Übersetzung: v.1.3a Michael Wolter 05/2022

Übersetzt mit Hilfe von www.deepl.com