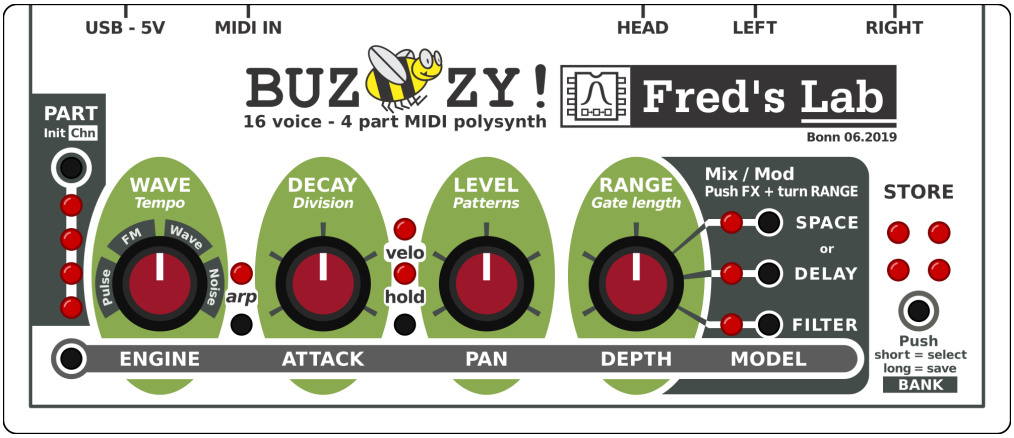
BUZZZY! module
User and service Manual

Revision 1.0 EN - 27/01/2020
for Firmware V1.21 - 08/10/2019



Frédéric Meslin / Fred's Lab

January 27, 2020

Introduction

Thank you for purchasing the Buzzzy!

Be sure I have put all my talent, skills and efforts to design this music instrument! Hopefully, you will like its compactness, love its sonic personality ... and come up with so many great ideas and new sounds for your music.

Love from Germany!

Fred, from Fred's Lab

Legal notices

Fred's Lab cannot be liable for any erroneous information contained in this manual. The contents of this manual may be updated at any time without prior notice. We have put best effort to ensure the information provided here is useful and accurate. Fred's Lab extends no liabilities in regard to this manual other than those required by the local laws.

Frédéric Meslin Audiogeräte
Herwarthstraße, 20
53115 Bonn, Germany
`info@fredslab.net`
`http://fredslab.net`

Support requests
For support requests, you can reach me per e-mail at:

`support@fredslab.net`

or per post, using the previously mentioned company address.

For each support request, please include the instrument model, serial number and a precise description of the problem encountered with a maximum of details and supporting elements for a quick resolution.

Copyright information

Warranty

Fred's Lab warranty this product free of defects **3 years** from its date of purchase.

This warranty covers product from manufacturing defects, when the product is used observing normal operating conditions. However, the warranty **does not cover**:

- normal product wear-out
- damages caused by failure to observe the rules of use
- damages due to negligence of the user
- products having been modified or repaired by the user or a third person

More information about product warranty can be found in the **General Terms and Conditions of Sale document** available at: `https://fredslab.net/en/terms.html`

Special thanks

I would like to personally thank all friends, relatives and colleagues, for their support, advises and reviews during the development of the **Buzzzy!**.

- **Fanny & Neal Homberger**
- **Jacques & Marie-Francoise Meslin**
- **Serge Beauchamp**, **Benoit Ruelle**, **Emilie Gillet**, **Oliver Rockstedt**

Front panel graphics (except the Bee) were designed by **Serge Beauchamp**. The German version of this user manual was translated by **Fanny Homberger** and the French version by **Benoit Ruelle**. Also, this project wouldn't have been possible without the financial support from the **Kickstarter Backers**. Thank you for your trust!

Precautions

Before plugging in the **Buzzzy!** and go **rocking the World**, have a sit and read this precautions through:

- Always use the device in a dry and warm environment
- Never drop the device or expose to too much pressure or vibration
- Never spill liquids or bath the device in beer
- Never clean the device with an aggressive solvent
- Only use quality and certified USB power adapters to power the device
- Never wiggle the plugs to disconnect the cords
- Never connect the line outputs to the power outputs of an amplifier
- Never connect the headphone output to a line output of any device
- Only modify the unit at your own risk!

The Buzzzy! used in conjunction with headphones and speaker systems can produce **very loud sounds** in a wide range of frequencies.

Human hearing is **very sensitive** and can be damaged quickly. So watch out your hears and those of your audience!

Contents

1 Presentation

The **Buzzzy!** is a 16-voices polyphonic, 4-parts multi-timbral, digital sound MIDI module.

- **Polyphonic** means it can play several notes at the same time.
- **Multi-timbral** means it can play several instruments at the same time.
- **Digital sound** means the sound is created using audio algorithms.
- **MIDI module** means it needs a MIDI controller to generate notes.

What is MIDI?

MIDI stands for **Musical Instrument Digital Interface**. It is a technical standard allowing music instruments to communicate and be used in concerto. It defines a protocol, messages, a digital interface and cables / connectors formats.
A MIDI connection carries notes, controller changes, clocks and various configuration data.

The **Buzzzy!** is a digital instrument, it computes the sounds using mathematical models and algorithms, executed by a powerful ARM micro-processor.

Remarks on polyphony

The 16-voices polyphony is the **absolute maximal** number of notes that can be played at the same time. If you try to play more notes simultaneously, the older voices will be muted and reused for newer notes. This mechanism is called "voice-stealing" and will result in audio artifacts or parts of your music not being played when triggered. In order to avoid it, keep in mind the module limitations while composing your music.

The **Buzzzy!** uses a dynamic allocation mechanism for the processor resources. When using multiple audio FXs and more processing hungry voices, polyphony decreases. Always disable the FXs and mute the voices you are not actively using to get the maximum polyphony out of this instrument.

2 Requirements

To play with the **Buzzzy!** you need:

- **An audio system** with line-level inputs or a pair of headphones
- **A MIDI controller**, a sequencer or a computer
- **A USB power source**

An audio system
can be a HiFi stereo system, a power mixing desk connected to speakers or a pair of active speakers.

A MIDI controller
can be any master-keyboard, wind-controller, surface ... that sends MIDI notes. You can also attach the instrument to a hardware or a software MIDI sequencer. It can run on a computer, a laptop or a tablet / smartphone. You may need a dedicated external MIDI interface or just use the USB connection.

A USB power source
can be a USB host, a power bank or a USB wall adapter / charger. Never use cheap USB chargers that can be very noisy.

Remarks on USB noise

If you keep hearing irritating hum and noises when the **Buzzzy!** is connected via USB, you may experience a **USB ground loop problem**. Fortunately, there are several solutions to overcome this problem.

Solution 1:
• Use an other USB power source (power bank or wall adapter)
• Connect the instrument to the computer with a MIDI DIN cable

Solution 2:
• Use a USB galvanic isolator

Example: Olimex USB-ISO module
`https://www.olimex.com/Products/USB-Modules/USB-ISO`

MIDI DIN connections are, by design, prevented from ground loop problems. It is the preferred solution to avoid these annoying issues.

3 Instrument setup

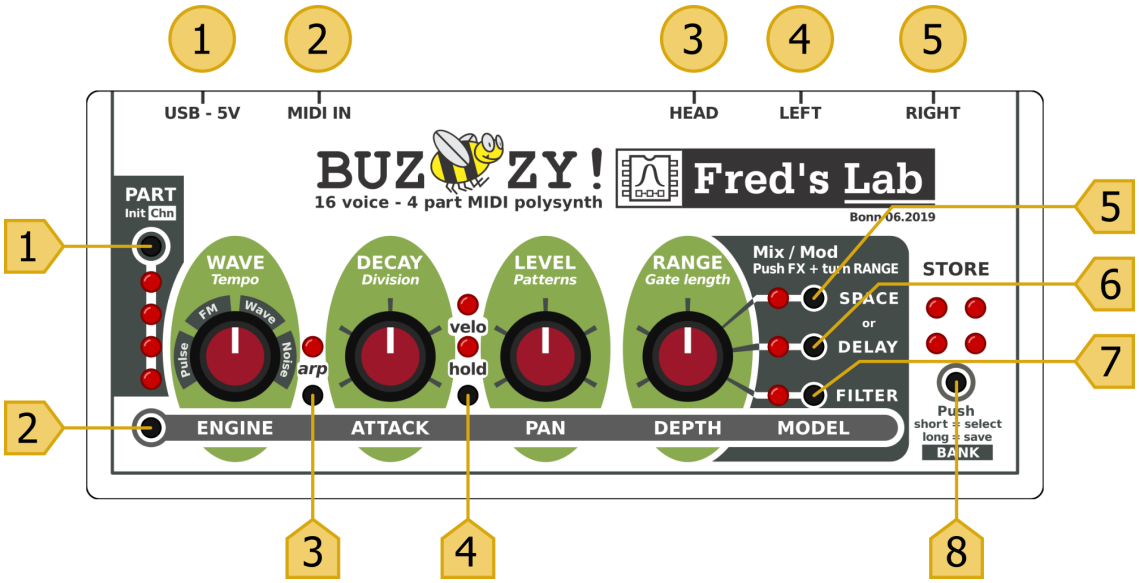
Step 1: Audio
Connect the **Buzzzy!** line-level outputs (left & right) to your audio system inputs using two 6.35mm unbalanced jacks.
Or
Connect the headphones output (head) to your headphones pair using the 3.5mm stereo jack cable. An adapter is required if your headphones have a 6.35mm jack cable.

Step 2: MIDI
Connect the **Buzzzy!** MIDI DIN input (MIDI IN) to your MIDI compatible controller or sequencer / computer using a MIDI DIN cable.
Or
Connect the USB port (USB - 5V) to your computer, laptop, tablet or smartphone.

Step 3: Power
Connect the **Buzzzy!** power input (USB - 5V) to your power source of choice.
Switch on you audio system, adjust the volume and play some notes!

4 Instrument operation

Once the instrument is properly installed and you can hear some sounds when playing notes, let's have a good look at its front panel:



4.1 Connections

(1) USB - 5V

This is the USB-B connector for both power and MIDI over USB function. It should be attached to a source to power up the **Buzzzy!**.

(2) MIDI IN

This is the MIDI Input DIN connector for a MIDI compatible controller or sequencer. It should be attached to a MIDI Output DIN connector of an other device.

(3) HEAD

This is the headphones output connector to attach an headphones pair. The amplifier is designed for low to medium impedance headphones. It can also be used as an additional line-level stereo audio output.

(4) (5) LEFT & RIGHT

This is the headphones output connector to attach an headphones pair. The amplifier is designed for low to medium impedance headphones. It can also be used as an additional line-level stereo audio output.

4.2 Potentiometers

Turning a potentiometer adjusts the *principal sound parameter*, labeled above it, in white with the bigger font. This parameter is only modified for the currently selected part.

To access the *secondary sound parameter*, labeled below the potentiometer, the **Alt** switch must be pressed or the alt function locked.

Arpeggiator parameters, in italic font, can be adjusted by turning the potentiometer WHILE maintaining the **Arp** switch pressed.

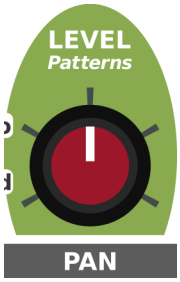
WAVE
this potentiometer
1. selects which variation of sound algorithm
2. selects which engine (Pulse, FM, Waves, Noise)
for the next note played.

With the **Arp** switch pressed, the Arpeggiator tempo can be adjusted.

DECAY
This potentiometer
1. adjusts the envelope decay time
2. adjusts the envelope attack time
of all notes currently played.

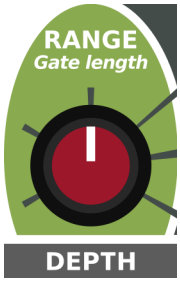
With the **Arp** switch pressed, the Arpeggiator clock division can be adjusted.

LEVEL
This potentiometer
1. adjusts the volume
2. adjusts the panning
of all notes currently played.



With the **Arp** switch pressed, the Arpeggiator "note repeat count" or pattern can be adjusted.

RANGE
This potentiometer
1. adjusts the FX range parameter
2. adjusts the FX depth parameter
3. controls the FX mix level or the filter modulation amount and rate of the FX currently selected.



With the **Arp** switch pressed, the Arpeggiator gate length can be adjusted. While a specific **FX** switch is still pressed, turning the **RANGE** potentiometer adjusts the FX mix or the modulation amount.

4.3 Switches & indicators

(1) PART

This is the **Part** switch.
By pressing it multiple times, the currently edited part can be selected.
The LEDs below the switch display the edited part.

This switch also allows:
1. initializing the selected part sound
2. changing the selected part *MIDI Channel*

These two extended functions are accessed by pressing the switch long enough (in combination with **Alt** for *MIDI channel learn*) and are described later in this user manual.

(2) ALT

This is the **Alt** switch.
Pressing it down allows access to the *secondary sound parameters*. The alt state can be **locked** by performing a "double-click" (two rapid successive presses) on the switch. The state can be **unlocked** by pressing the alt switch once.

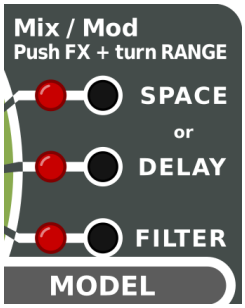
When the alt state is locked, the part LEDs are all blinking.

(3) ARP

This is the **Arp** enable switch.
By pressing it, the *Arpeggiator* is either enabled or disabled. The LED above the switch display the current state of the Arpeggiator.
By pressing the **Arp** switch long enough, the *Arpeggiator* **hold** function is enabled.
Keeping the **Arp** switch pressed while turning potentiometers allows the editing of the Arpeggiator specific parameters.
More information about the *Arpeggiator* can be found in the *Arpeggiator* section.

(4) HOLD

This is the **Hold** enable switch.
By pressing it multiple times, the mode of the *Envelope* is selected.
When the **Hold** LED is lit, the *Envelope* an AHD envelope.
When the **Velocity** LED is lit, the controller transmitted **velocity** affects the volume of the notes played.
By pressing the **Hold** switch long enough, the *Envelope* becomes a looping envelope and the **Hold** LED blinks.
More information about the *Envelope* can be found in the *Envelope* section.



(5) SPACE

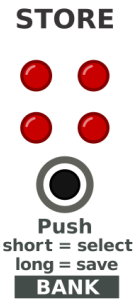
This is the **Space** switch.
By pressing it, one of the *Space FX* is either enabled or disabled. The FX is enabled for the edited part. If a *Delay FX* was previously enabled, it is immediately disabled as *Space* and *Delay* FXs are mutually exclusive.
The *Space* LED, at the switch left, indicates the state of the FX.
Using the **Alt** switch, one of the 3 available *Space FX* can be selected.

(6) DELAY

This is the **Delay** switch.
By pressing it, one of the *Delay FX* is either enabled or disabled. The FX is enabled for the edited part. If a *Space FX* was previously enabled, it is immediately disabled as *Space* and *Delay* FXs are mutually exclusive.
The *Delay* LED, at the switch left, indicates the state of the FX.
Using the **Alt** switch, one of the 3 available *Delay FX* can be selected.

(7) FILTER

This is the **Filter** switch.
By pressing it, one of the *Filter FX* is either enabled or disabled. The FX is enabled for the edited part. The *Filter* LED, at the switch left, indicates the state of the FX.
Using the **Alt** switch, one of the 3 available *Filter FX* can be selected.



(8) STORE

This is the **Store** switch.
By pressing it shortly, the selected *multi* slot is recalled. A *multi* consists in the state of all sound parameters from the 4 parts. By pressing it long enough, the current edited *multi* is saved on the selected *multi* slot.

Do not forget to **save your changes** before recalling a *multi* or the modifications of the current *multi* will be forever lost!

5 The synthesis

6 Engines

The **Buzzzy!** is equipped with 4 distinct sound engines:

Pulse, **FM**, **Wave** and **Noise**.

Each engine offers a set of sound variations.
The selected engine makes the core of a note sound and a unique timbre palette. Let's explore the different facets of these engines.

6.1 Pulse

Pulse is an engine made to produce clean and anti-aliased waveforms with sharp edges. It generates the squares, the pulses, the alternative pulses (with richer bass) waves, all with or without pulse-width modulation and vibrato.

List of the variations available:

N°	Name	Description
1	Square	Square wave
2	Pulse25	Pulse with 25.00% PW ratio
3	Pulse12	Pulse with 12.50% PW ratio
4	Pulse6	Pulse with 6.25% PW ratio
5	Alt38	Alternate pulse with 37.50% PW ratio
6	Alt25	Alternate pulse with 25.00% PW ratio
7	Alt12	Alternate pulse with 12.50% PW ratio
8	Alt6	Alternate pulse with 6.25% PW ratio
9	SquareSlow	Square with slow 87.50% PWM
10	PulseSlow	25.00% pulse with slow 43.75% PWM
11	SquareFast	Square with fast 50.00% PWM
12	PulseFast	25.00% pulse with fast 25.00% PWM
13	Alt50Slow	50.00% alternate pulse with slow 87.50% PWM
14	Alt25Slow	25.00% alternate pulse with slow 43.75% PWM
15	Alt50Fast	50.00% alternate pulse with fast 50.00% PWM
16	Alt25Fast	25.00% alternate pulse with fast 25.00% PWM

6.2 FM

FM is an engine made to produce both rounder and grittier waveforms based on traditional linear phase modulation, using 2 operators in series. It generates classic and distorted FM tones, with sinus, triangle or sawtooth waveforms as carrier and modulator.

Vibrato is also available for this engine.

List of the variations available:

N°	Name	Description
1	HalfSoft	1/2x Sine/Sine with 10% phase modulation
2	HalfMedium	1/2x Sine/Sine with 30% phase modulation
3	HalfHard	1/2x Sine/Sine with 60% phase modulation
4	1xSoft	1x Sine/Sine with 15% phase modulation
5	1xMedium	1x Sine/Sine with 30% phase modulation
6	2xSoft	2x Sine/Sine with 15% phase modulation
7	3xSoft	3x Sine/Sine with 15% phase modulation
8	3xSoft	3x Sine/Sine with 30% phase modulation
9	TriSoft	1/2x Triangle/Sine with 10% phase modulation
10	1xTri	1x Triangle/Sine with 15% phase modulation
11	2xTri	2x Triangle/Sine with 10% phase modulation
12	3xTri	3x Triangle/Sine with 10% phase modulation
13	HalfSaw	1/2x Saw/Sine with 20% phase modulation
14	2xSaw	2x Saw/Sine with 10% phase modulation
15	SawHard	1/2x Saw/Saw with 5% phase modulation
16	Trash	2x Saw/Saw with 10% phase modulation

6.3 Wave

Wave is an engine made to produce low resolution waveforms, as they would be directly played from a ROM, like in some early sound generators.

Vibrato is also available for this engine.

List of variations available:

N°	Name	Description
1	Sinus	A regular sine wave
2	FlatSine	A slightly clipped sine wave
3	SlantedSine	A phase distorted sine wave
4	DoubleSine	Sum of two sine waves (octaves)
5	Triangle	A regular triangle wave
6	FlatTri	A slightly clipped triangle wave
7	SlantedTri	A phase distorted triangle wave
8	DoubleTri	Sum of two triangle waves (octaves)
9	Smooth	A smoothed random wave
10	Random	A harsh random wave
11	SawSoft	A soft fake saw wave
12	SawHard	A hard fake saw wave
13	Sin2Tri	A wave evolving from sine to triangle
14	Tri2Dbl	A wave evolving from triangle to double sine
15	Smo2Rnd	A wave evolving from smooth to random wave
16	Soft2Hard	A wave evolving from soft to hard wave

6.4 Noise

Noise is an engine made to produce colored noises with different kind of filtering (low pass and band pass), loudness compensation, as well as at different sampling rates.

Pitch-bend control can be used to adjust the noise filters respective cutoff frequencies.

List of variations available:

N°	Name	Description
1	NoiseLP	low passed colored noise
2	NoiseBP	band passed colored noise
3	HalfLP	Under-sampled low passed noise
4	HalfBP	Under-sampled band passed noise
5	CrapLP	Crappy low-passed noise
6	CrapBP	Crappy band-passed noise

7 Envelope

The *envelope* controls the dynamic level of a note. It also acts on several other sound parameters (depending on selected engine and variation). The **Buzzzy!** offers one envelope per voice. While this can sound limited, the envelopes offer different modes.

7.1 Attack and Decay

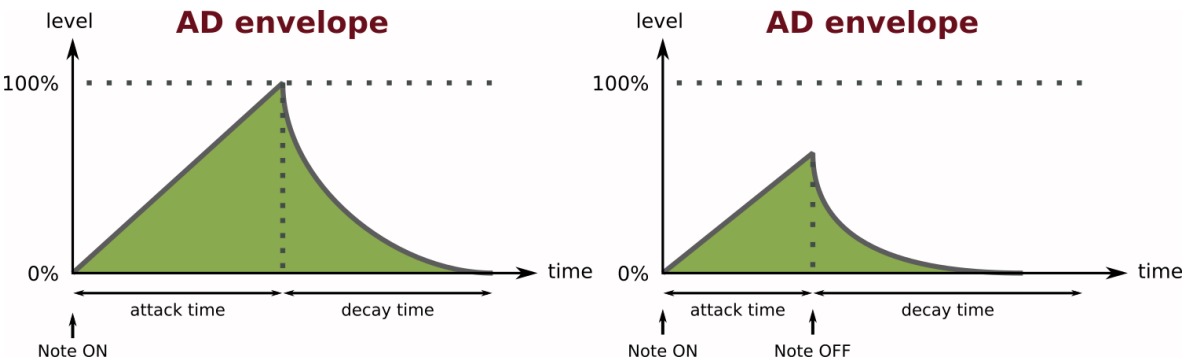
The basic attributes for a simple envelope are its **attack time** and **decay time**.

The **attack time** defines the time taken by the volume of a note to rise from minimum to its maximum.

The **decay time** defines the time taken by the volume of a note to fall from its release level to the minimum.

Times range from very short (< 1ms) to extended (18s).

7.1.1 AD envelope



This is the default behavior of the *envelope*.

When a note is triggered, its associated voice AD *envelope* rises from the minimum to the maximum level. When the maximum is reached, the *envelope* immediately falls from the maximum back to the minimum level. After this cycle, the note is considered to be terminated and its related voice is consequently released.

The *envelope* can be interrupted in its **attack phase**, if the note is released before the maximum level is reached. In this situation, the *envelope* will fall from its release level to the minimum, at the specified **decay** time.

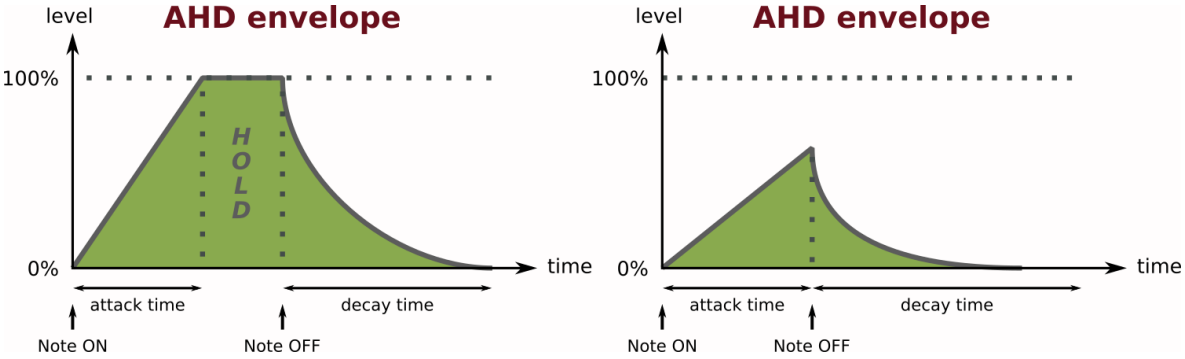
Remark
AD *envelopes* are very useful for percussive sounds, short melodic bleeps and plucks but also for arpeggiated patterns.

7.2 Hold, Velocity & Looping

For longer notes or chords, an additional **hold** stage is often required.
By pressing the **hold** switch several times, the AHD mode can be enabled or disabled. To select the *envelope* **looping** mode, press the **hold** switch long enough.

The **hold** and **velocity** LEDs, located above the switch, indicates the current *envelope* mode and its sensitivity to **velocity**. When the **hold** LED blinks, the *envelope* is in looping mode.

7.2.1 AHD envelope



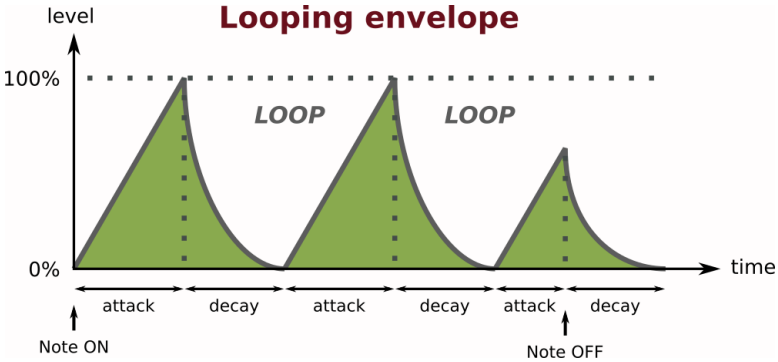
This mode is similar to the AD *envelope*, excepting it has an **hold phase**.

When a note is triggered, its AHD *envelope* rise from the minimum to the maximum level. When the maximum is reached, the *envelope* **stays at this level** until the note is released. When this happens, the *envelope* level falls from the maximum back to the minimum level. After this cycle, the note is terminated and its related voice is released.

As in AD mode, the AHD *envelope* can also be interrupted in its **attack phase**. The AHD behavior is then strictly identical to the AD *envelope* one.

Remark
AHD *envelopes* are perfect for chords or drone sounds.

7.2.2 Looping envelope



This mode is similar to the AD *envelope*, excepting the *envelope* immediately restarts when it reaches its minimum level.

When a note is triggered, the looping *envelope* rises from the minimum to the maximum level. When the maximum is reached, the *envelope* immediately falls from the maximum back to the minimum level. When the minimum level is reached, the cycle restarts. This loop repeats until the note is released. When this happens, the *envelope* level falls from the current level to the minimum level and no longer loops. The note is then terminated and its related voice is released.

Remark
Looping *envelopes* have great use for sound effects, percussion noises and pulsating chords.

7.2.3 Velocity

If the *envelope* **velocity** is enabled, notes velocity information sent by the MIDI controller or the sequencer to the **Buzzzy!** affects the volume of the notes played on the edited part. This gives more expressiveness to an instrument by letting the musician control the loudness of the notes separately.

Remark
Velocity variations come in handy to accentuate or de-emphasize elements in a rhythmic pattern. When playing complex chords, different **velocity** levels can help to finely distribute the predominance of the chord degrees or notes.

8 Mixer

The *mixer* sets the **level** and the **panning** of a part. It is meant to adjust the music instrument loudness and position in the stereo space. The mixer uses a sinus / cosine panning law.

8.1 Volume & Pan

Only two parameters are available: **level** and **pan**.

To adjust the **level**, turn the **Level** potentiometer.
To adjust the **pan**, maintain the **Alt** switch pressed (or **lock** the **alt** state)
and turn the **Level** potentiometer.

Remark
To avoid sound distortion or even hard-clipping when mixing together several parts or instruments, try to leave some safe **level margin**.

The **Buzzzy!** has no **master level** control. The overall volume is the result of the sum of each part sound volume.

9 FXs

The **Buzzzy!** incorporates a wide selection of quality sound effects, with several **delays**, **reverbs** and **filters**. These effects can be chained and set differently on every part.

Important
Sound effects require **processing power**. Some effects as *filters* or *delays* not that much, some others like *flanger* and *reverbs* much more. To maximize the polyphony, disable the unused effects and limit the number of *reverbs* used simultaneously. Otherwise, the number of voices available will drop drastically.

Both *Delays FXs* and *Space FXs* (special *reverbs*) require a significant amount of memory. This is the reason why *Delays FXs* and *Space FXs* are exclusive. Put differently, a *delay* and a *space* cannot be chained on a *part*.

9.1 Selection & configuration

By pressing one of the 3 **effect** switches, the respective *effect* is enabled or disabled on the edited *part*. If the *effect* enabled is one of the *Space FXs* and there was a *Delay FX* already engaged, the later is disabled and vice versa. A *Filter FX* can always be added to a time time-domain *effect*.

Effects have four parameters: **range**, **depth**, **model**, **mix** (**mod** for *filters*).

Their exact influence of these parameters on the sound depends on the combination of effect and model selected. These parameters will be detailed in the following sections.

Selecting a model
Maintaining the **Alt** switch pressed (or having the **alt** state **locked**) while pressing any **effect** switch changes the related *effect* **model**. Models can be selected whether the *effect* is enabled or not.

Adjusting the mix
The **mix** parameter sets the amount of effect "wet" signal blend with the original "dry" signal. Maintaining the desired **effect** switch pressed while turning the **Range** potentiometer adjusts the **mix** parameter.

Filters don't have a **mix** parameter but they can be modulated using the dedicated*LFO* (low frequency oscillator). Controlling the modulation **amount** and **rate** is achieved the same way as adjusting the **mix** parameter for the other *effects*.

Let's now have detailed look at all the *effects*.

9.2 Space FXs

Space FXs belong to the **reverberation** type of effects. These are time-domain effects. They are made to simulate the influence of a certain "space" or "room" on the sound. Here, they are deliberately not called *reverbs*, as this term refers to a specific category of effects, the algorithmic **reverberations**.

Models used in the **Buzzzy!** are very simplified and were designed empirically. The goal was to obtain the best sound character and musicality using a quite limited computing power.

9.2.1 Fort

Fort is a large empty room with wall / ceiling / floor kind of reflections model. The reflections are relatively dry, bright and little modulated. There is no tail sound with this algorithm. The effect produces stereo filtered reflections.

The **decay** of the reflections is set with the **decay** (= **Range pot.**) parameter and the room **size** using the **size** (= **Depth pot.**) parameter.

9.2.2 Reservoir

Reservoir is very similar to *Fort*. The difference is the modulation of the reflections. The room size is fixed but the speed and amount of the modulations are both adjustable.

The **decay** of the reflections is set with the **decay** (= **Range pot.**) parameter and the **modulation** speed & amount using the **modulation** (= **Depth pot.**) parameter.

9.2.3 Abyss

Abyss share some similarities to large *hall* **reverb** algorithms. The focus of this model is on the sound tail quality and length.

The **decay** of the tail is set with the **decay** (= **Range pot.**) parameter and the room **size** using the **size** (= **Depth pot.**) parameter.

Remark
Abyss is the most demanding effect model for computing power ... and by far!
Use it sporadically to preserve the instrument polyphony.

9.3 Delay FXs

Delays FXs belong to the **echo / delay** type of effects. These are also time-domain effects and they are based on the repetition of the original signal at different intervals, with playback modulations and filtering.

Remark
In the **Buzzzy!** sound engine, the **delay lengths** are tuned. They respect an exponential law and therefore the *Delays FX* can also be used as resonators or comb filters.

9.3.1 Mono

The *mono* delay repeats the original signal a defined number of times. The repetitions are progressively degraded, they lose their loudness, dynamic precision and some of their frequency content.

The frequency of the repetitions is set with the **length** (= **Range pot.**) parameter and the number of the repetitions using the **feedback** (= **Depth pot.**) parameter.

Feedback parameter extend to infinite repeat, allowing drones and ambient lush sounds.

9.3.2 Ping-pong

The *ping-pong* delay is very similar to the *mono* delay. The essential difference is the repetitions come first from the left channel and then from the right channel and so on. It gives a nice stereo image to the sound.

The frequency of the repetitions is set with the **length** (= **Range pot.**) parameter and the number of the repetitions using the **feedback** (= **Depth pot.**) parameter.

Feedback parameter also extend to infinite repeat.

9.3.3 Chorus / Flanger

Although this effect does not feel like a delay, it uses the same ground principles. The repetitions are closer and modulated, the perceived effect is a constant variation of the sound spectrum giving it a thicker character. This stereo effect has multiple uses, try to experiment with different settings!

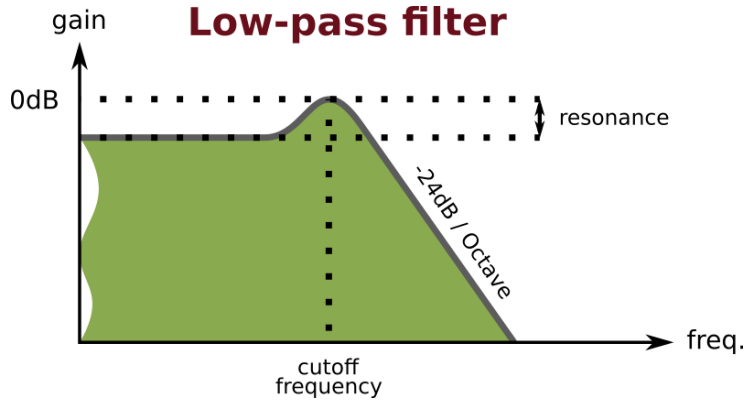
The modulation rate is set with the **speed** (= **Range pot.**) parameter and the amount of feedback using the **feedback** (= **Depth pot.**) parameter.

9.4 Filter FXs

The *filter* is an effect shaping the frequency content (audio spectrum) of the sound of a part. It can attenuate or amplify full bands of frequencies. The *filter* can always be activated and is applied prior to any other effect (*Space FX* or a *Delay FX*).

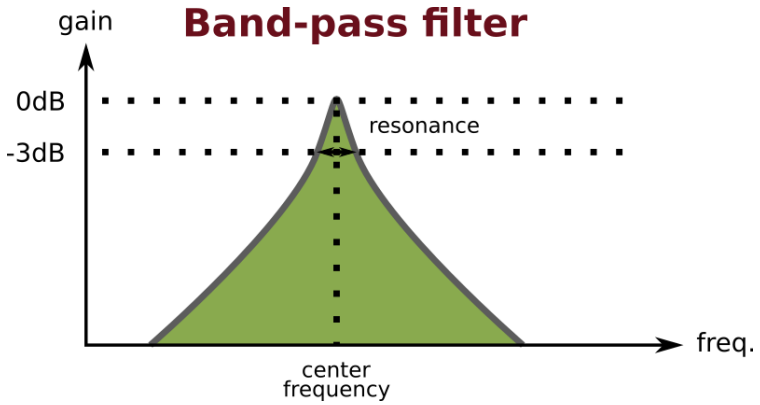
The *filter* exists in 3 modes: **Low-pass** (LP), **Band-pass** (BP) and **High-pass** (HP). The attenuation slope / filter order is fixed and is 24dB per octave or 4th order.

9.4.1 Low-pass



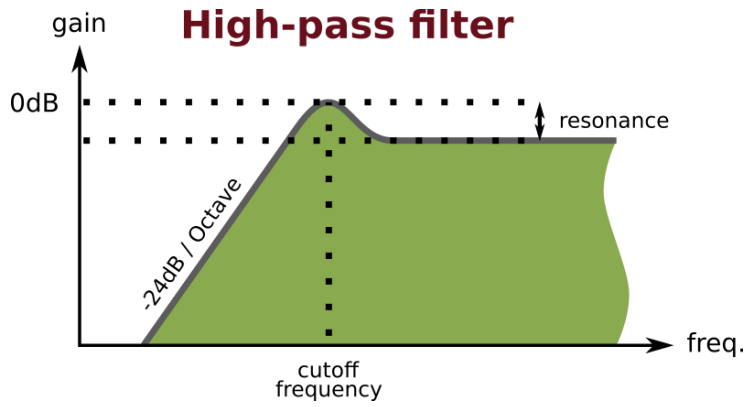
The **low-pass** *filter* only keeps frequencies below its **cutoff frequency**. The frequencies above this limit are progressively attenuated. **Resonance** parameter creates an adjustable "bump" that accentuates the frequencies right before the **cutoff frequency**.

9.4.2 Band-pass



The **band-pass** *filter* keeps a band of frequencies around its **center frequency**. The frequencies outside of this band are progressively attenuated. **Resonance** parameter adjust the size of the band (or the filter **selectivity**) around the **center frequency**.

9.4.3 High-pass



The **high-pass** *filter* only keeps frequencies above its **cutoff frequency**. The frequencies below this limit are progressively attenuated. **Resonance** parameter creates an adjustable "bump" that accentuates the frequencies right after the **cutoff frequency**.

9.4.4 Modulation

Filter FXs do not have a **mix** parameter but they come with a *built-in LFO* to enliven the sound by animating the filter **center** or **cutoff frequency**. The *LFO* rate is based (multiple) on the part *Arpeggiator* tempo. Two rates can be selected: slow (a 2 bars cycle) or fast rate (a half bar cycle).

Turning the **Range** potentiometer while keeping the **Filter** switch pressed will adjust both modulation **amount** and **rate**. Using the left half of the control applies a slow modulation and using the right half a fast modulation. The closer the **Range** potentiometer is set to its extremities, the strongest is the modulation.

9.4.5 Distortion

The filters are coupled with a *soft-clippling distortion* which adds extra character to the sound and round off the harsh resonance peaks. To adjust the *distortion* effect, set the *part* volume to an appropriate level.

9.4.6 Editing

To adjust the *filter* **cutoff frequency** or **center frequency**, turn the **Range** potentiometer, while the *Filter FX* is activated.

To adjust the *filter* **resonance** or **selectivity**, maintain the **Alt** switch pressed (or **lock** the **alt** state) and turn the **Range** potentiometer, while the *Filter FX* is activated.

Remark
For editing, the *filter*, when actived, always has the priority. In other words, you cannot edit the parameters from a *Space FX* or a *Delay FX* while the *Filter FX* is enabled. This is a limitation of the device.

10 Arpeggiator

The *Arpeggiator* is a handy function allowing the creation of complete melodic patterns from chords or single notes. **Buzzzy!** *Arpeggiator* always plays notes respecting the order (in time) in which they have been entered. Each part has its own *Arpeggiator* that can be used independently.

10.1 On & Hold

To enable the *Arpeggiator* on the edited part, press the **Arp** switch. The **Arp** LED located above indicates the state (enable or disable) the *Arpeggiator* is.

10.2 Tempo & Clocking

There are two ways to set the base **tempo** for the *Arpeggiator*:

- **Free-running**, the **tempo** is set manually, using the tempo parameter.
- **Synchronized**, the **tempo** is set automatically, with MIDI clock ticks (USB or DIN).

The **Buzzzy!** automatically switch to **synchronized** mode if an external clock is received and revert back to the **free running** mode when no other clock source is available.

Tempo (for **free-running** mode only) is set by holding the **Arp** switch down while turning the **Wave** potentiometer.

10.3 Divisions, Repetitions & Gate length

In addition to the **tempo**, the step length of the notes sequence can be set using the **division** parameter. The **division** rate ranges from 1/4 (corresponding to a step length of a sixteenth note) to 2 (corresponding to a step length of a half note) by increments of 1/4.

These are the available divisions:

N°	Ratio	Note
1	4.00	A whole note
2	2.00	A half note
3	1.33	A dotted eighth note
4	1.00	A quarter note
5	0.66	A triplet quarter note
6	0.50	A eighth note
7	0.33	A triplet eighth note
8	0.25	A sixteenth note

Division is set by holding the **Arp** switch down while turning the **Decay** potentiometer.

The *Arpeggiator* offers to repeat the notes of the sequence. The **repeat** parameter ranges from 1 (no repeat) to 16 times.

Repetition is set by holding the **Arp** switch down while turning the **Level** potentiometer.

The notes duration can also be adjusted using the **gate length** parameter. The duration ranges from 0% to 100% of the selected step length.

Gate length is set by holding the **Arp** switch down while turning the **Range** potentiometer.

11 Parts

A *part* comprises an instrument and a dedicated MIDI channel. To play notes from a specific *part*, the MIDI controller or sequencer must send its MIDI events to the appropriate MIDI channel.

The **part** switch allows to select which *part* is currently edited using the **Buzzzy!** interface. The **part** LEDs below display which *part* is currently selected.

11.1 Initializing a part

To initialize a *part* to its default configuration, select the desired *part* using the **part** switch and then press long enough this very same **part** switch. All sound parameters from this edited *part* will be set back to their default value.

11.2 Setting a part MIDI channel

By default, *MIDI channels* 1, 2, 3 and 4 are mapped to *parts* 1, 2, 3, 4 respectively. In this configuration, MIDI notes sent on channel 1 will be played by the **Buzzzy!** instrument 1 and so on. It is possible to change this attribution by activating the *MIDI learn* function.

When *MIDI learn* is activated on a specific *part*, the channel of the next *MIDI Note on* event received by the **Buzzzy!** (either on MIDI USB or MIDI DIN) will be attributed to this *part*. The *MIDI learn* state will automatically be deactivated.

To activate *MIDI learn*, first select the desired *part* using the **part** switch and then, while having **Alt** pressed (or the **alt** state **locked**), press long enough on the **part** switch. The **part** LED of the parts set in *MIDI learn* should blink rapidly.

Remark
The mapping of the **MIDI channels** is **saved globally** and **not per multi.** When channels are changed, it applies to all of your presets. The same *MIDI channel* can be attributed to different *parts*. This is referred to as the **"stacking-mode"** and it permits easy layering of *parts* for more complex tones. Finally, *MIDI Learn* can be activated on more than one *part* at once for this purpose.

12 Store

The store is used as the user configuration memory and allows to save up to 4 *multis* for a later recall. A *multi* consist in the state of all parameters from the 4 parts. The *multis* are stored in the machine internal **Flash** memory. Therefore, the configurations are preserved even when the machine is powered off.

Important remark

Never ever switch the machine off while saving a *multi*.

This will result in the corruption of the *flash memory* and **all user data stored will be definitely lost!**

12.1 Loading a multi

Press the **store** switch multiple times to load the desired *multi*. Be careful, all unsaved edited parameters will be discarded without notification! The store **store** LEDs above the switch indicate which *multi* has been recalled.

12.2 Saving a multi

Press the **store** switch long enough and all the configurations of the 4 parts edited *multi* will be saved in the selected *multi* slot.

13 MIDI implementation

13.1 Channel Voice Messages

Channel Voice Messages are used to send basic music events to one specific channel or part. They consists in 3 bytes: the **status** byte, followed by two **data** bytes. Here follows the list of supported messages.

Note Off
Stops the playback of a note.

```
Byte: 0     1     2
Data: 0x8c  note  velo
      c    = MIDI channel (0 - 15)
      note = MIDI note (0 - 127)
      velo = note velocity (0 - 127)
```

Note On
Starts the playback of a note.

```
Byte: 0     1     2
Data: 0x9c  note  velo
      c    = MIDI channel (0 - 15)
      note = MIDI note (0 - 127)
      velo = note velocity (1 - 127)*

* Note: a velocity of 0 corresponds to a Note off event.
```

Program Change
Selects the current *multi*.

```
Byte: 0     1
Data: 0xCc  multi
      c     = MIDI channel (0 - 15)
      multi = multi number (0 - 15)
```

Pitch Bend
Increases or decreases the pitch of the given channel (= part) notes.

```
Byte: 0     1     2
Data: 0xEc  least most
      c     = MIDI channel (0 - 15)
      least = least significant 7bit of 14bit bend value*
      most  = most significant 7bit of 14bit bend value*

* Note: bend value is centered around 0x4000 = 0 cents.
```

13.2 System Real Time Messages

System Real Time Messages are used to send high-priority messages relative to music timing and sequences.

Clock start
Start all parts arpeggiators if enabled.

```
Byte: 0
Data: 0xFA
```

Clock continue
Re-start all parts arpeggiators if enabled.

```
Byte: 0
Data: 0xFB
```

Clock stop
Stop all parts arpeggiator.

```
Byte: 0
Data: 0xFC
```

Clock tick
Synchronizes the MIDI master clock of the sound module.
A tick corresponds to 1/24th of a quarter note.

```
Byte: 0
Data: 0xF8
```

Active sensing
Ensure the MIDI connection is working.

Once first sent, **active sensing** messages must be sent to the module periodically. If the module stops receiving **active sensing** messages for an extended period of time, it will automatically perform a **all sounds off**.

```
Byte: 0
Data: 0xFE
```

13.3 Controller Changes / CCs

Controller Change Messages are used to alter sound parameters of a given channel. They allow more musical expressivity by adjusting the sound character while playing or allow the editing of the part parameters from an extended / remote MIDI interface or a computer software (sequencer or preset editor).

The messages consist in 3 bytes: the **status** byte, followed by the **control number** byte and then terminated by the **control value** byte. Here follows the list of supported messages.

Controller Change
Modifies a specific sound parameter value of the given channel (= part).

```
Byte: 0     1      2
Data: 0xBc  index  value
      c      = MIDI channel (0 - 15)
      index  = controller number (see the CC list)
      value  = controller value* (see the CC list)

* Note: internal parameter values are represented in 10 bit format.
        Since MIDI CCs can only transmit 7 bit values, values transmitted
        are internally interpolated and smoothed.
```

List of the implemented MIDI CC.

Parameter	**Decimal**	**Hexa**	**Description**
Modwheel	1	0x01	Modulation wheel (vibrato level)
Volume	7	0x07	Mixer level
Pan	10	0x0A	Mixer pan
Coarse	12	0x0C	Oscillator coarse (semitones)
Fine	13	0x0D	Oscillator fine (cents)
Tempo	16	0x10	Arpeggiator tempo (when not clocked)
Div	17	0x11	Arpeggiator division
Repeat	18	0x12	Arpeggiator repeat
Gate	19	0x13	Arpeggiator gate length
Wave	70	0x46	Oscillator wave (variation)
Resonance	71	0x47	Filter resonance
Decay	72	0x48	Envelope decay time
Attack	73	0x49	Envelope attack time
Cutoff	74	0x4A	Filter cutoff / center frequency
Engine	75	0x4B	Oscillator engine
Filter mod.	80	0x50	Filter modulation rate & amount
Space mix	82	0x52	Space mixer level
Delay mix	83	0x53	Delay mixer level
Space model	85	0x55	Space effect model
Delay model	86	0x56	Delay effect model
Filter type	87	0x57	Filter effect type
Arp. mode	88	0x58	Arpeggiator mode (off, on, hold)
Env. mode	89	0x59	Envelope mode (AD, AHD, velocity)
Env. loop	90	0x5A	Envelope looping (off, on)
Space decay	91	0x5B	Space effect decay time
Space diffuse	92	0x5C	Space effect diffusion factor
Delay feedback	93	0x5D	Delay effect feedback level
Delay length	94	0x5E	Delay effect delay length
All sounds off	120	0x78	Mute all sounds
Reset all controllers	121	0x79	Set all controllers to their initial value
All notes off	123	0x7B	Release immediately all notes

14 Sysex implementation

Sysex or System Exclusive Messages are lengthier MIDI messages used to transfer, save and load multis and global parameters. Two types of messages exist: **requests** and **dumps**.

Requests
Requests are sent to the music instrument to ask for a dump of a specific multi or the globals.

Dumps
Dumps are sent by the instrument in response to the reciprocal request or are sent to the instrument to replace a specific multi or configure the globals. Dumps previously requested can be sent back without any alteration, to restore a previous memory configuration.

14.1 Sysex requests

Globals request
Request the globals configuration from the memory.

```
Byte: 0    1    2    3
Data: 0xF0 0x00 0x00 0xF7
      0xF0 = Sysex start
      0x00 = Service 0 => "Globals request"
      0x00 = Reserved
      0xF7 = Sysex end
```

Multi request
Request a specific multi from the memory.

```
Byte: 0    1    2    3
Data: 0xF0 0x01 0xMM 0xF7
      0xF0 = Sysex start
      0x01 = Service 1 => "Multi request"
      0xMM = Multi id (0-15 = memory slot, 127 = current edit buffer)
      0xF7 = Sysex end
```

14.2 Sysex dumps

Globals dump
Representation of the globals configuration.
Sysex is 12 bytes long.

```
Byte: 0    1    2    n   7
Data: 0xF0 0x10 0x00 ... 0xF7
      0xF0 = Sysex start
      0x10 = Service 16 => "Globals dump"
      0x00 = Reserved
     ... data (8 bytes)
      0xF7 = Sysex end
```

Data content:

```
Byte Description
0:   master tune, lowest 5 bits (-512 to +511)
1:   master tune, highest 5 bits
2:   bendrange, lowest 5 bits (-16 to +15)
3:   bendrange, highest 5 bits
4:   part 1 MIDI channel (0 to 15)
5:   part 2 MIDI channel (0 to 15)
6:   part 3 MIDI channel (0 to 15)
7:   part 4 MIDI channel (0 to 15)

Mastertune is in thousandth of a semitone.
Bendrange is in semitones.
```

Multi dump
Request a specific multi from the memory.
Sysex is 324 bytes long.

```
Byte: 0    1    2    n   323
Data: 0xF0 0x11 0xMM ... 0xF7
      0xF0 = Sysex start
      0x01 = Service 1 => "Multi dump"
      0xMM = Multi number (0-15 = memory slot, 127 = current edit buffer)
     ... data (320 bytes)
      0xF7 = Sysex end
```

Data packing:

```
Byte Description
0:   byte 0, lowest 7 bits
1:   byte 1, lowest 7 bits
2:   byte 2, lowest 7 bits
3:   byte 3, lowest 7 bits
4:   tops, top 7th bits of the 4 preceeding bytes
5:   byte 4, lowest 7 bits
6:   byte 5, lowest 7 bits
7:   byte 6, lowest 7 bits
8:   byte 7, lowest 7 bits
9:   tops, top 7th bits of the 4 preceeding bytes
...
```

Data content (for each part):

```
0, 1:   coarse *
2, 3:   fine *
4, 5:   wave *
5, 6:   decay *
7, 8:   level *
9, 10:  variation *
11, 12: attack *
13, 14: pan *
15, 16: reverbDecay *
17, 18: reverbDiffuse *
19, 20: reverbMix *
21, 22: delayLength *
23, 24: delayFeedback *
25, 26: delayMix *
27, 28: filterCutoff *
29, 30: filterResonance *
31, 32: filterModulation *
33, 34: tempo *
35:     reverbMode
36:     delayMode
37:     filterMode
38:     arpDiv
39:     arpRepeat
40:     arpGate
41:     flags
42-63:  reserved

* Note: 10bit parameters uses 16bit words and are packed LSB first.

This data structure is repeated 4 times, to represent the sound parameters
of the four parts included in the dumped multi.
```

15 Firmware update

To benefit from new features (and potential bug-fixes), the internal *Firmware* of the **Buzzzy!** can be re-programmed. Updates should only be performed when **absolutely necessary** and the procedure steps and recommendations **should be strictly followed.**

Important remark
Failing to observe the update procedure may simply **brick your device!** Therefore, please proceed carefully.

To update your **Buzzzy!** instrument to the **latest firmware revision**, please refer to the appropriate update documentation and read the following tutorials:

How to get the internal revision:
`https://www.kickstarter.com/projects/1304489933/buzzzy-the-digital-polysynth/posts/2643964`

How to update the firmware:
`https://www.kickstarter.com/projects/1304489933/buzzzy-the-digital-polysynth/posts/2643004`

16 Technical details

Here you will find all technical data concerning the **Buzzzy!**.

• **Number of parts & maximal polyphony:**
4 parts / 16 voices

• **Number of effects:**
3 Space effects: Fort, Reservoir & Abyss
3 Delay effects: Echo, Ping-pong & Flanger
3 Filter effects: LP, BP & HP, 24dB / Octave

• **User preset memory:**
4 x 4 = 16 multi slots

• **Device dimensions & weight:**
Unit 148 (width) x 70 (depth) x 68 (height) mm
Unit 200 grams (approximately)

• **USB classification:**
USB 1.1 full-speed (12 Mbit/s)
USB MIDI Class-Compliant device

• **Power requirements:**
+5V +/- 5%, 75mA (maximum)
375mW (maximum)

• **Line output characteristics:**
+8.5dBu / 2.1V_{RMS} / 5.8Vpkpk peak level
Impedance DC 470Ω, AC 0Ω (f < 24kHz)

• **Headphones output characteristics:**
90mW average / 16Ω headphones
64mW average / 32Ω headphones
Impedance DC / AC 10Ω

• **Connectors:**
USB type B socket
MIDI Input DIN5 socket
Headphones 3.5mm female stereo jack socket
Line outputs 2x 6.35mm female jack sockets

• **Master clocks:**
MCU @ 81.92MHz, USB @ 48MHz, I2S @ 12.288MHz

• **Audio resolution:**
48.00kHz, 24Bit stereo audio
100dB SNR & dynamic range

17 Schematics and BOM

To minimize **electronic waste** and ensure **long product life**, Fred's Lab is willing to provide all technical documents needed to repair or adapt his products for final users needs. The following schematics and bill-of-material documents are provided "as is" with no warranty of any kind.

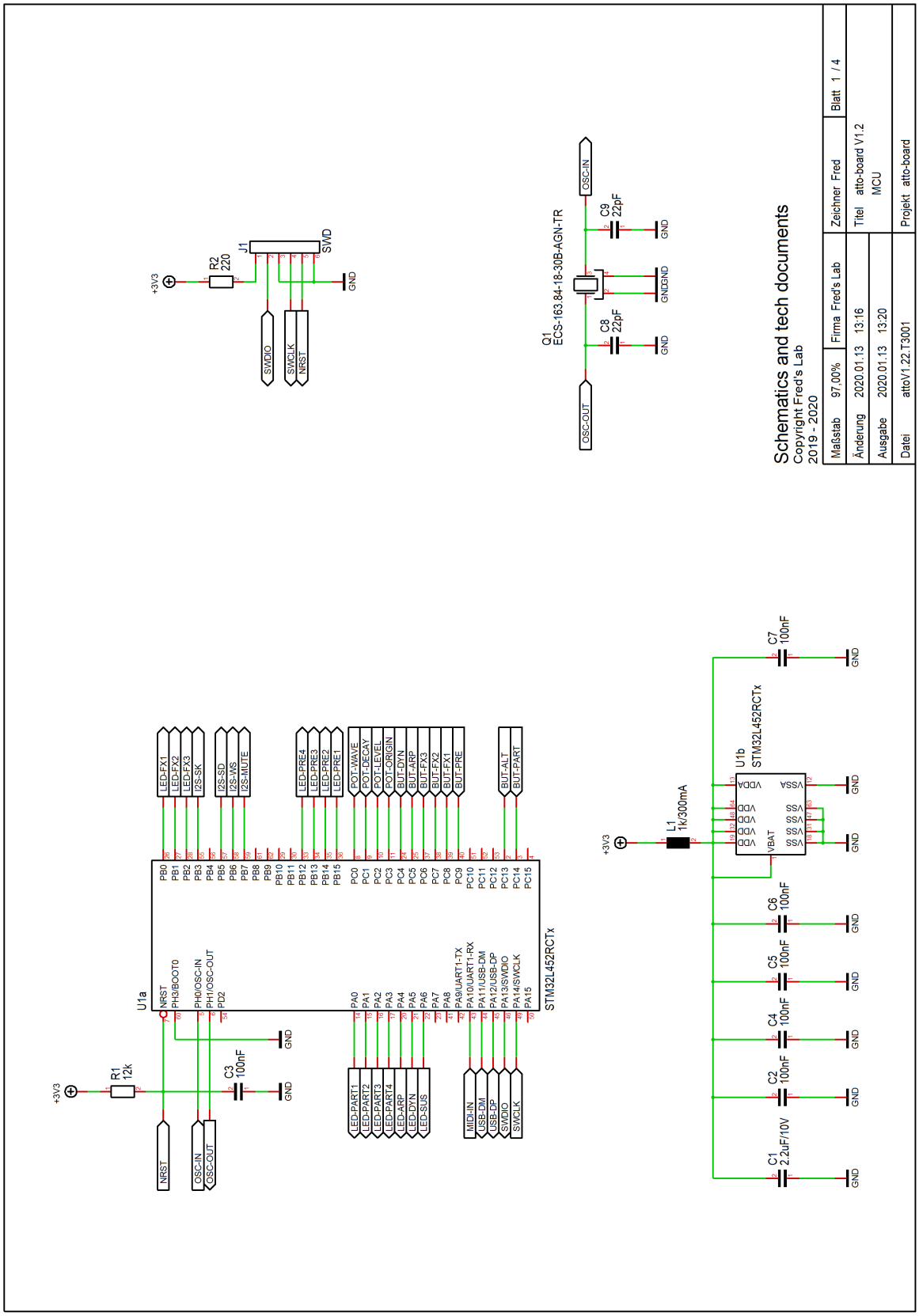
Any modification of a Fred's Lab instrument voids the included 3-year product warranty. Mods are at your own risk. Repairs must be carried out by a competent person with the appropriate SMD equipment.

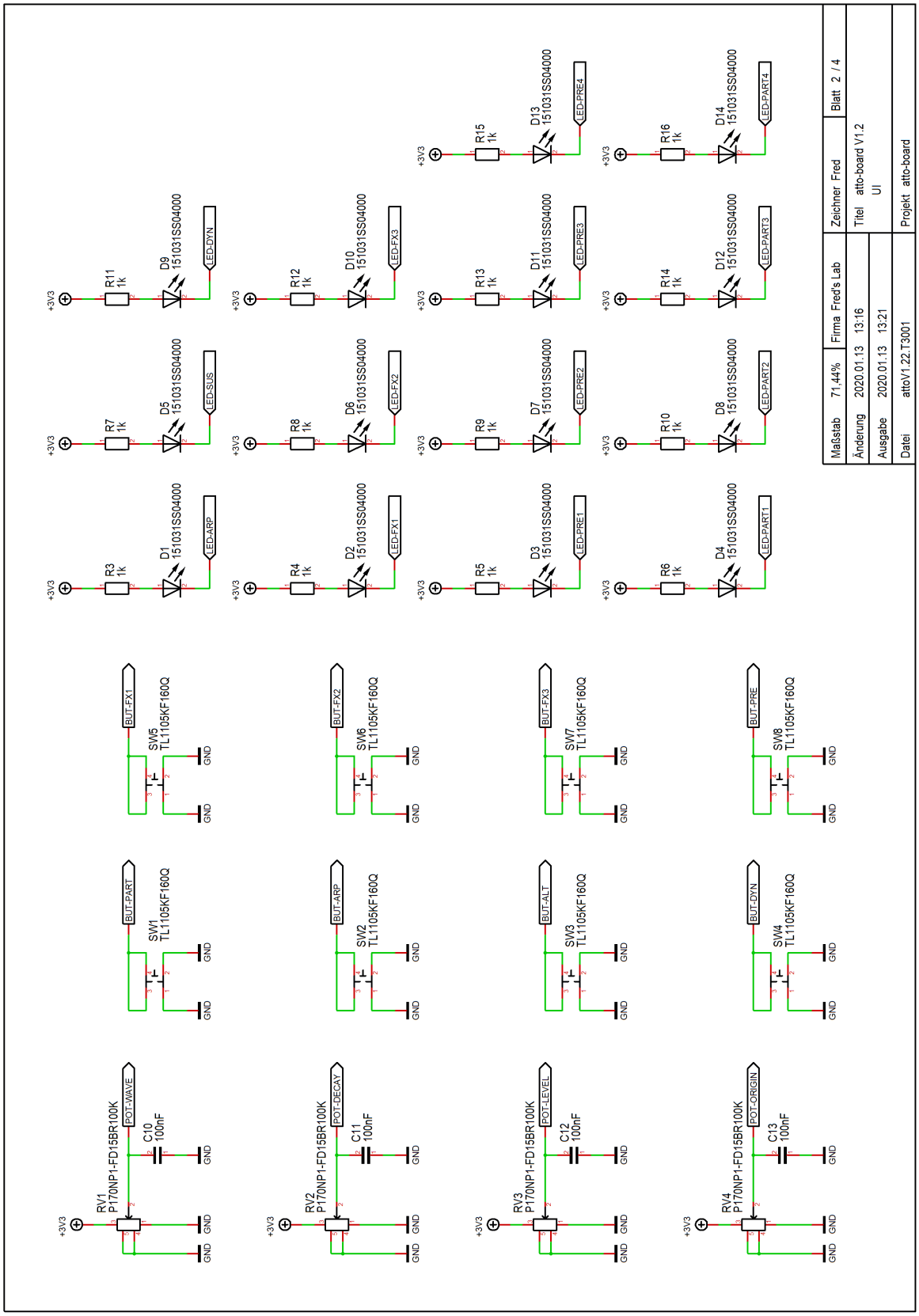
Fred's Lab is **always available for maintenance and repair** of your musical instruments, so do not hesitate to contact the support service for a free quote. Spare parts can be directly ordered from us.

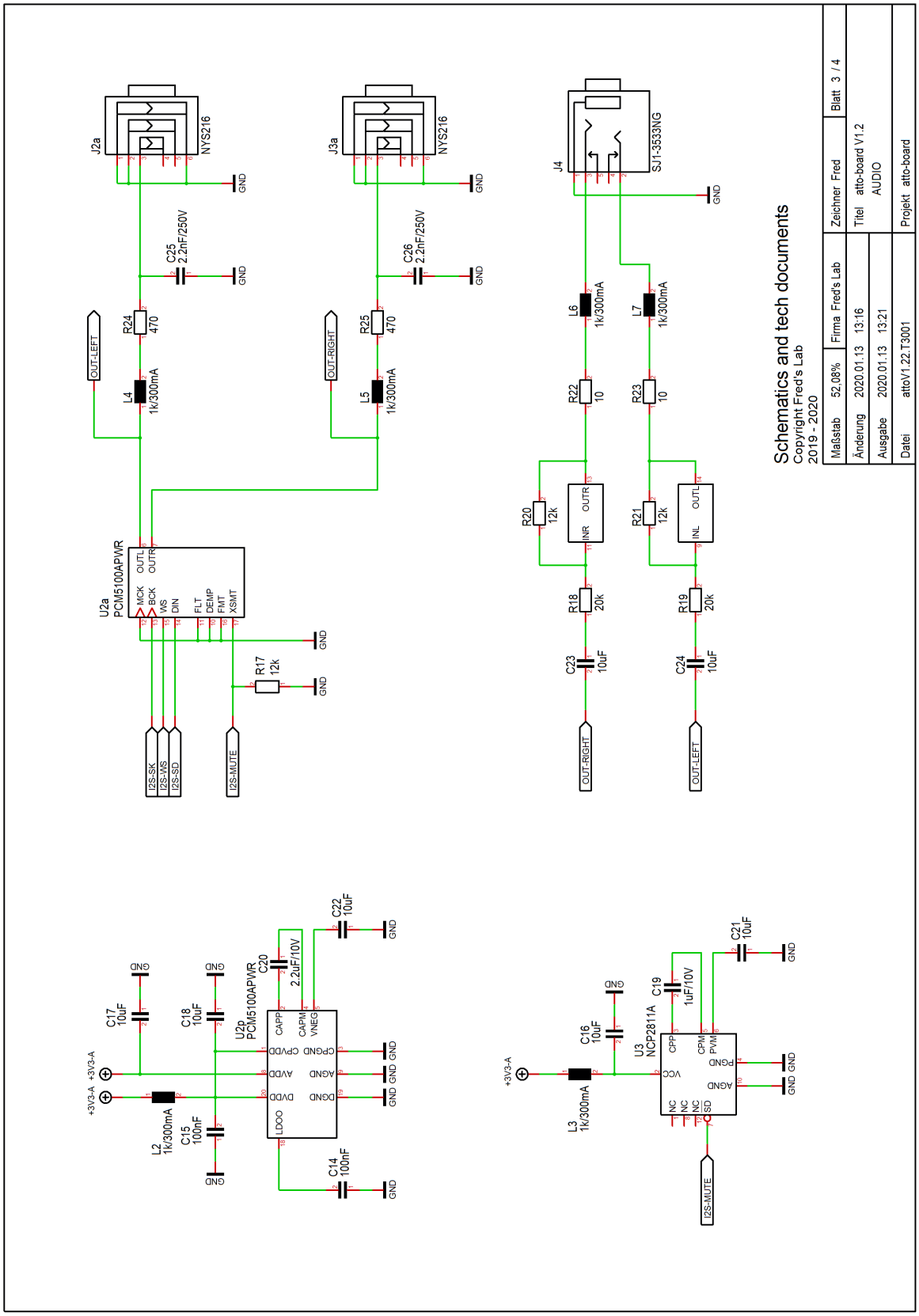
Intellectual property

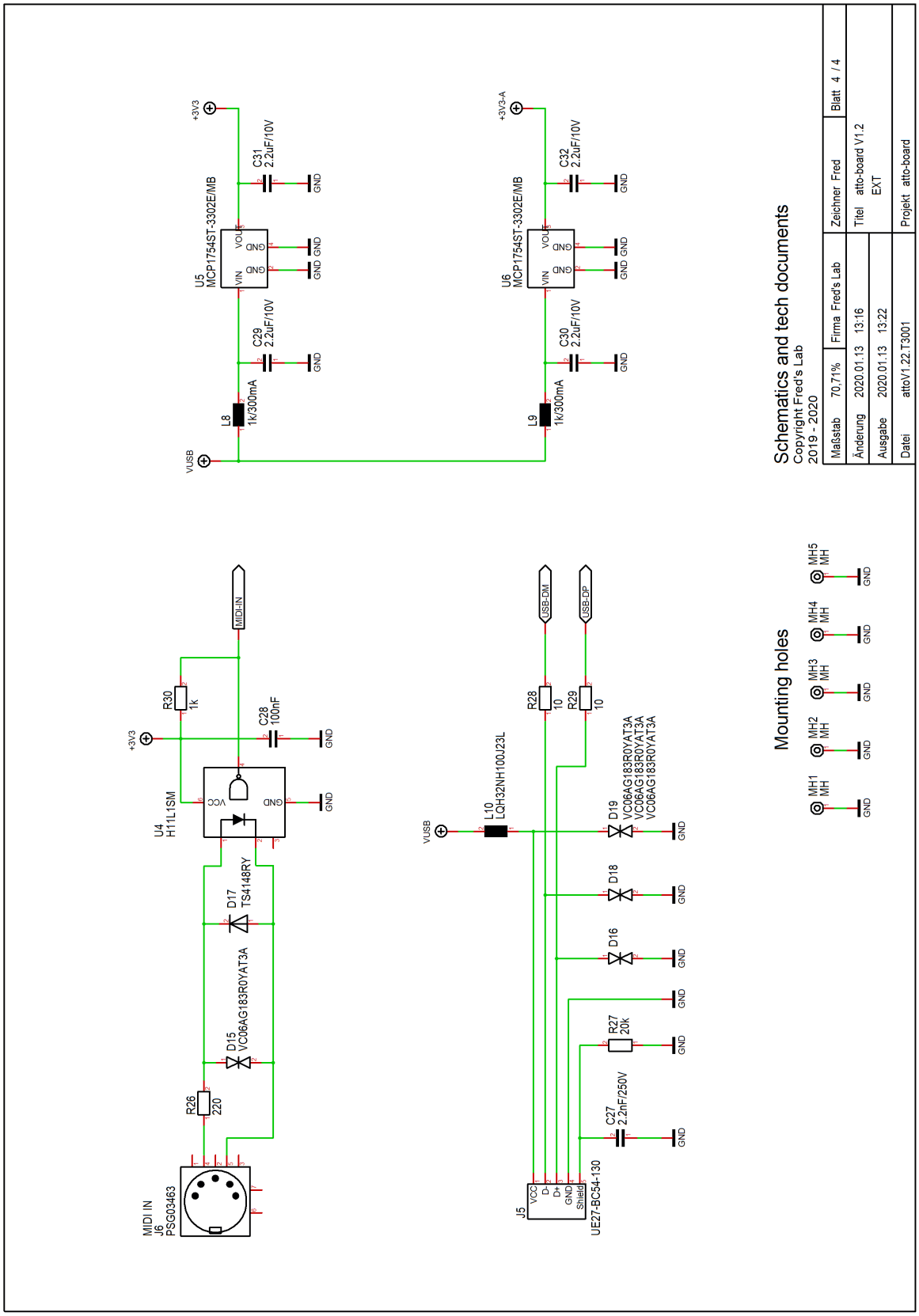
The technical documents provided below are for **advisory, repair and educational purposes only.** They remain the entire intellectual property of their respective author and cannot be reproduced without a written authorization.

The reader is granted to draw inspiration from this information for his projects (commercial or not), while respecting the limits of non **cloning or counterfeiting** the original product. If in doubt about legal matters, **please contact our company directly.**


Schematics and tech documents
Copyright Fred's Lab
2019 - 2020
Maßstab 97.00%
Firma Fred's Lab
Zeichner Fred
Blatt 1 / 4
Änderung 2020.01.13 13:16
Titel atto-board V1.2
MCU
Ausgabe 2020.01.13 13:20
Datei attoV1.22.T3001
Projekt atto-board
U1a
STM32L452RCTx
U1b
STM32L452RCTx
Q1
ECS-163.84-18-30B-AGN-TR
J1
SWD
L1
1k/300mA
R1
12k
R2
220
C1
2.2uF/10V
C2
100nF
C3
100nF
C4
100nF
C5
100nF
C6
100nF
C7
100nF
C8
22pF
C9
22pF
+3V3
GND
NRST
OSC-IN
OSC-OUT
SWDIO
SWCLK
LED-FX1
LED-FX2
LED-FX3
I2S-SK
I2S-SD
I2S-WS
I2S-MUTE
LED-PRE4
LED-PRE3
LED-PRE2
LED-PRE1
POT-WAVE
POT-DECAY
POT-LEVEL
POT-ORGIN
BUT-DYN
BUT-ARP
BUT-FX3
BUT-FX2
BUT-FX1
BUT-PRE
BUT-ALT
BUT-PART
LED-PART1
LED-PART2
LED-PART3
LED-PART4
LED-ARP
LED-DYN
LED-SUS
MIDI-IN
USB-DM
USB-DP


Maßstab 71.44% Firma Fred's Lab Zeichner Fred Blatt 2 / 4
Änderung 2020.01.13 13:16 Titel atto-board V1.2
Ausgabe 2020.01.13 13:21 UI
Datei attoV1.22.T3001 Projekt atto-board
+3V3 R3 1k D1 151031SS04000 LED-ARP
+3V3 R4 1k D2 151031SS04000 LED-FX1
+3V3 R5 1k D3 151031SS04000 LED-PRE1
+3V3 R6 1k D4 151031SS04000 LED-PART1
+3V3 R7 1k D5 151031SS04000 LED-SUS
+3V3 R8 1k D6 151031SS04000 LED-FX2
+3V3 R9 1k D7 151031SS04000 LED-PRE2
+3V3 R10 1k D8 151031SS04000 LED-PART2
+3V3 R11 1k D9 151031SS04000 LED-DYN
+3V3 R12 1k D10 151031SS04000 LED-FX3
+3V3 R13 1k D11 151031SS04000 LED-PRE3
+3V3 R14 1k D12 151031SS04000 LED-PART3
+3V3 R15 1k D13 151031SS04000 LED-PRE4
+3V3 R16 1k D14 151031SS04000 LED-PART4
BUT-PART SW1 TL1105KF160Q GND GND
BUT-ARP SW2 TL1105KF160Q GND GND
BUT-ALT SW3 TL1105KF160Q GND GND
BUT-DYN SW4 TL1105KF160Q GND GND
BUT-FX1 SW5 TL1105KF160Q GND GND
BUT-FX2 SW6 TL1105KF160Q GND GND
BUT-FX3 SW7 TL1105KF160Q GND GND
BUT-PRE SW8 TL1105KF160Q GND GND
+3V3 RV1 P170NP1-FD15BR100K POT-WAVE C10 100nF GND GND GND
+3V3 RV2 P170NP1-FD15BR100K POT-DECAY C11 100nF GND GND GND
+3V3 RV3 P170NP1-FD15BR100K POT-LEVEL C12 100nF GND GND GND
+3V3 RV4 P170NP1-FD15BR100K POT-ORGIN C13 100nF GND GND GND

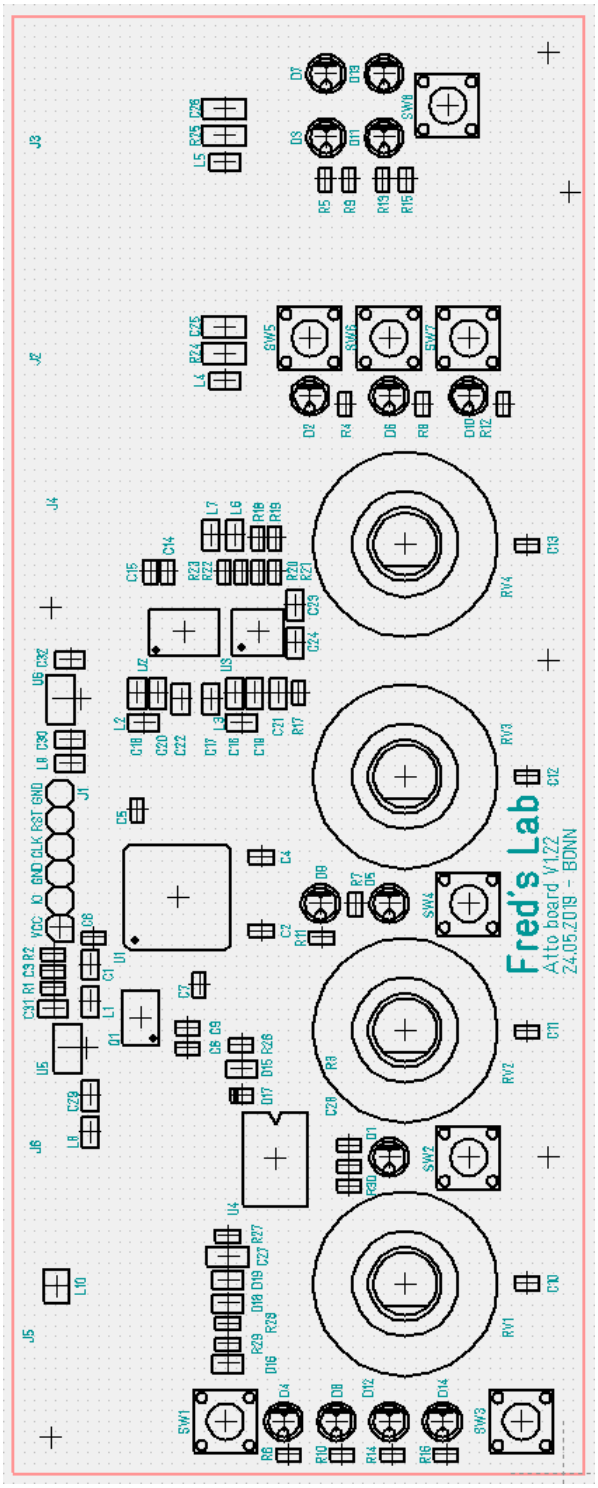

Schematics and tech documents
Copyright Fred's Lab
2019 - 2020
Maßstab 52,08%
Firma Fred's Lab
Zeichner Fred
Blatt 3 / 4
Änderung 2020.01.13 13:16
Titel atto-board V1.2
AUDIO
Ausgabe 2020.01.13 13:21
Datei attoV1.22.T3001
Projekt atto-board
J2a
NYS216
J3a
NYS216
J4
SJ1-3533NG
C25 2.2nF/250V
C26 2.2nF/250V
R24 470
R25 470
L4 1k/300mA
L5 1k/300mA
L6 1k/300mA
L7 1k/300mA
R22 10
R23 10
R20 12k
R21 12k
R18 20k
R19 20k
C23 10uF
C24 10uF
R17 12k
U2a PCM5100APWR
OUTL
OUTR
MCK
BCK
WS
DIN
FLT
DEMP
FMT
XSMT
OUT-LEFT
OUT-RIGHT
I2S-SK
I2S-WS
I2S-SD
I2S-MUTE
GND
U2p PCM5100APWR
C20 2.2uF/10V
C22 10uF
C17 10uF
C18 10uF
C15 100nF
C14 100nF
L2 1k/300mA
+3V3-A
U3 NCP2811A
C19 1uF/10V
C16 10uF
C21 10uF
L3 1k/300mA


Schematics and tech documents
Copyright Fred's Lab
2019 - 2020
Maßstab 70.71%
Firma Fred's Lab
Zeichner Fred
Blatt 4 / 4
Änderung 2020.01.13 13:16
Titel atto-board V1.2
EXT
Ausgabe 2020.01.13 13:22
Datei attoV1.22.T3001
Projekt atto-board
Mounting holes
MH1 MH
MH2 MH
MH3 MH
MH4 MH
MH5 MH
GND
+3V3
+3V3-A
VUSB
U5 MCP1754ST-3302E/MB
U6 MCP1754ST-3302E/MB
C31 2.2uF/10V
C32 2.2uF/10V
C29 2.2uF/10V
C30 2.2uF/10V
L8 1k/300mA
L9 1k/300mA
MIDI-IN
U4 H11L1SM
R30 1k
C28 100nF
R26 220
D15 VC06AG183R0YAT3A
D17 TS4148RY
MIDI IN J6 PSG03463
USB-DM
USB-DP
L10 LQH32NH100J23L
R28 10
R29 10
D19 VC06AG183R0YAT3A
D18
D16
R27 20k
C27 2.2nF/250V
J5 UE27-BC54-130
VCC
D-
D+
GND
Shield

Atto board BOM

V1.22 – 01.07.2019

Pos	Qty	Name	Description	Value	Vmax	Cmax	Pmax	Material	Tolerance	Manufacturer	Package	Reference
PCBA												
Ress												
	15	R3,R4,R5,R6,R7,R8,R9,R10,R11,R12,R13,R14,R15,R16,R30	Thick film resistor	1k	50V		62.5mW		1,00%	Yageo	0402	RC0402FR-071KL
	4	R22,R23,R28,R29	Thick film resistor	10	50V		62.5mW		1,00%	Yageo	0402	RC0402FR-0710RL
	2	R2,R26	Thick film resistor	220	50V		62.5mW		1,00%	Yageo	0402	RC0402FR-07220RL
	2	R24,R25	Thick film resistor	470	50V		500mW		1,00%	Panasonic	0805	ERJ-UP6F4700V
	4	R1,R17,R20,R21	Thick film resistor	12k	50V		62.5mW		1,00%	Yageo	0402	RC0402FR-0712KL
	3	R18,R19,R27	Thick film resistor	20k	50V		62.5mW		1,00%	Yageo	0402	RC0402FR-0720KL
	2	R26,R27	Thick film resistor	NM							0805	
Caps												
	2	C8,C9	Multilayer ceramic capacitor	22pF	16V			C0G	5,00%	Würth Electronics	0402	885012005027
	13	C2,C3,C4,C5,C6,C7,C10,C11,C12,C13,C14,C15,C28	Multilayer ceramic capacitor	100nF	16V			X7R	10,00%	Würth Electronics	0402	885012205037
	3	C25,C26,C27	Multilayer ceramic capacitor	2.2nF	250V			X7R	10,00%	TDK	0805	C2012X7R2E222K085AA
	1	C19	Multilayer ceramic capacitor	1uF	10V			X5R	20,00%	Würth Electronics	0603	885012106010
	6	C1,C20,C29,C30,C31,C32	Multilayer ceramic capacitor	2.2uF	10V			X5R	20,00%	Würth Electronics	0603	885012106011
	7	C16,C17,C18,C21,C22,C23,C24	Multilayer ceramic capacitor	10uF	10V			X5R	10,00%	Murata	0603	GRM188R61A106KE69J
	2	C25,C26	Ceramic capacitor	NM							0402	
Inductors												
	9	L1,L2,L3,L4,L5,L6,L7,L8,L9	Ferrite pearl (1k @ 100Meg)	1k/300mA		300mA				Murata	0603	BLM18AG102SN1D
	1	L10	10uH / 325mA non-shielded coil	10uH/325mA		325mA			5,00 %	Murata	1210	LQH32NH100J23L
Diodes												
	14	D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12,D13,D14	3mm super red LED	151031SS04000						Würth Electronics	T3	151031SS04000
	4	D15,D16,D18,D19	TVS diode	TVS 14V						AVX	0603	VC06AG183R0YAT3A
	1	D17	Small signal diode	TS4148RY						Taiwan semiconductor	0805-D	TS4148 RYG
Miscs												
	1	Q1	16,384MHz 5x3,2mm crystal	ECS-163.84-18-30B-AGN-TR					25ppm	ECS		ECS-163.84-18-30B-AGN-TR
	4	RV1,RV2,RV3,RV4	Linear conductive plastic pot	P170NP1-FD15BR100K	200V		0.1W		20,00%	TT Electronics		P170NP1-FD15BR100K
	8	SW1,SW2,SW3,SW4,SW5,SW6,SW7,SW8	Tact switch	TL1105KF160Q	50V					E-Switch		TL1105KF160Q
ICS												
	1	U1	ARM 32bit MCU	STM32L452RCTx						ST Microelectronics	LQFP64	STM32L452RCT6
	1	U2	24bit DAC and line driver	PCM5100APWR					100dB	Texas Instrument	TSSOP20	PCM5100APWR
	1	U3	Headphones amplifier	NCP2811A						On Semiconductor	TSSOP14	NCP2811ADTBR2G
	1	U4	Optocoupler(schmitt trigger)	H11L1SM						On Semiconductor	DIL6 SMD	H11L1SM
	2	U5,U6	Precision LDO regulator	MCP1754ST-3302E						Microchip	SOT89	MCP1754ST-3302E/MB
Connectors												
	1	J1	SIL connectors	NM								
	2	J2,J3	6,35mm female jack socket	NYS216						REAN / Neutrik		NYS216
	1	J4	3,5mm female jack socket	SJ1-3533NG						CUI Inc		SJ1-3533NG
	1	J5	USB B shielded socket	UE27-BC54-130						Amphenol		UE27-BC54-130
	1	J6	DIN 5 contact 180° socket	PSG03463						Pro Signal		PSG03463
Others												
	14		8mm - 3mm leds holder	ELM-4-8MM						Bivar		ELM-4-8MM
	4		11mm push-on 6mm D-shaft knob	DRN110 006						Sifam		DRN110 006
	4		11mm red cap with line	C111 RED						Sifam		C111 RED


Fred's Lab
Atto board V1.2.2
24.05.2019 - BDNN

18 Norms

18.1 Europe: CE

CE DECLARATION OF CONFORMITY

1. Product unique identification:
Buzzzy! digital sound module
Belonging to the category "multimedia electronic equipment"

2. Address of the manufacturer and his authorized representative:
Frédéric Meslin Audiogeräte
Herwarthstraße, 20
53115 Bonn, Germany
Email: fred@fredslab.net Telephone: +49 228 53451657 (office hours)

3. Object of the declaration:
This equipment **conforms to the following requirements:**
•EN 55032:2015 (emission), EN 55035:2017 (immunity)
•EN 61000-4-2:2009 (ESD)
•EN 61000-4-3:2006 + A1:2008 + A2:2010 (immunity)
•EN 61000-4-8:2010 (immunity)
•EN 61000-6-3 (interference)
•2011/65/EU (ROHS 2), 2012/19/EU (WEEE)

After examinations conducted by the independent laboratory:
Transferstelle für Elektromagnetische Verträglichkeit
Hochschule Koblenz
Konrad-Zuse Straße 1
56075 Koblenz, Germany
Report: EMC Testreport 1193 / 2019 (available on request)

5. Signed for and on behalf of **Frédéric Meslin Audiogeräte**:

Frédéric Meslin, Lead Engineer of Fred's Lab
Bonn, the 10/04/2019

18.2 Canada: Interference regulation

This device does not exceed the Class B limits for radio noise emissions from digital apparatus set out in the radio interference regulation of the Canadian Department of Communications.

Cet équipement n'émet pas de bruits radiofréquence dépassant les limites applicables aux appareils numériques de la Classe B prescrites dans le règlement sur les interférences radio-électriques édicté par le Ministère Des Communications du Canada.

18.3 USA: FCC Information

This equipment has been verified to comply with the limits for a class B computing device, pursuant to FCC Rules. In order to maintain compliance with FCC regulations, shielded cables must be used with this equipment. Operation with non-approved equipment or unshielded cables is likely to result in interference to radio and TV reception.

Important: Changes and modifications made to the equipment without the approval of the manufacturer can void your authority to operate this equipment.

Note: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications.

However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna
- Increase the separation between the equipment and receiver
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- Consult the dealer or an experienced radio/TV technician for help