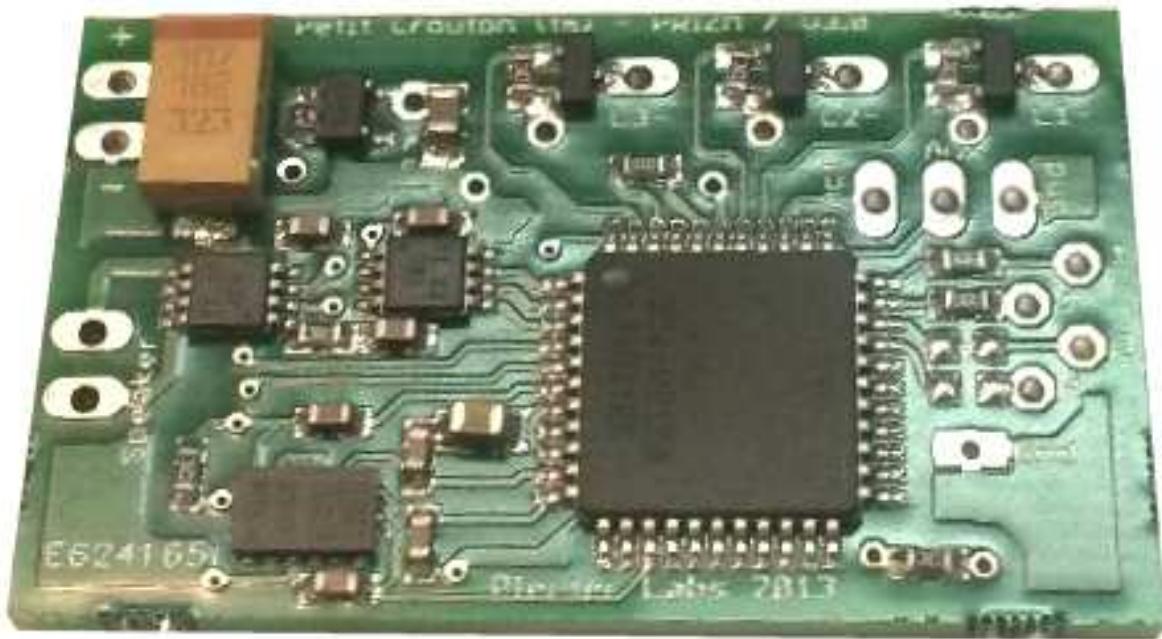


Petit Crouton™ PRIZM v3.5

Single Cell Middle Range Illuminated Saber Controller

User's manual

© Erv' - Plecter Labs
Edition 1.0 - November 2014
erv@plecterlabs.com
<http://www.plecterlabs.com>



Important release information

- 3 channel direct drive Color mixing with the FlexiBlend™ engine
- Supports luxeon III, V, rebel (old or new), seoul LEDs, Ledengin, tri-rebels, tri-Cree etc

We spent a lot of time writing this manual to ensure all the important information is provided for proper use of that board. If you are new to saber building, to the use of Petit Crouton boards, or simply to electronics in general, we highly recommend you print a copy of that document and keep it with you during the whole process of installing PC in your hilt.

Modification, copies or distribution of that document is strictly prohibited
© Plecter Labs / Erv' Plecter 2005-2014

Index

<u>PETIT CROUTON™ PRIZM V3.5</u>	<u>1</u>
IMPORTANT RELEASE INFORMATION	1
INTRODUCTION	4
High-Power LEDs (aka Luxeon™)	4
Sound section	5
Petit Crouton PRIZM V3.5 Features & Maximum Ratings	6
Placement & Installation	7
TOOLS AND PARTS REQUIRED TO INSTALL/OPERATE THE MODULE	8
HOW DOES IT WORK ?	8
SD CARD CONTENTS, SOUND BANKS AND SLOTS	9
BOARD OVERVIEW	10
USER'S NOTES	10
GETTING STARTED WITH PETIT CROUTON	11
WIRING AND OPERATING THE MODULE	11
General Power Switch & Recharge Port	11
General wiring	13
USER'S NOTES	13
Animated Accent LEDs	14
Calculating resistors for LEDs	14
High-power LED wiring	15
<u>HIGH POWER LED RESISTOR CALCULATIONS</u>	<u>16</u>
STUNT UPGRADE	16
RESISTOR CALCULATION	16
WATTAGE CALCULATION	16
RESISTOR BARGAIN	17
MAIN CONFIGURATION FILE	18
PARAMETERS AND FINE TUNING THE SABER	19
THE "OVERRIDE" CONFIGURATION FILE	23
USER'S NOTES:	23
COLOR PROFILES	24
Profiles definition	24
Profiles browsing	24
<u>DRIVE ADJUSTMENTS</u>	<u>25</u>
RESONANT CHAMBER	26
BROWSING THE SOUND BANKS – REBOOTING THE SABER	26
CREATING YOUR OWN SOUNDS	26
INSTALLING A SOUND FONT ON THE SD CARD	27
<u>ADVANCED WIRING & USAGE</u>	<u>28</u>

WIRING A GENERAL POWER-ON INDICATOR / ACCENT LED	28
ADD A CRYSTAL CHAMBER TO YOUR SABER	28
FLASH ON CLASH™ MIXING TECHNIQUES	30
 <u>ACCENT LEDS SEQUENCER</u>	 <u>32</u>
Stages & Delays	32
DEEP SLEEP FLASHING LED	34
 <u>FORCE PUSH” EFFECT</u>	 <u>35</u>
 <u>MUTE ON THE GO™</u>	 <u>36</u>
 USER’S NOTES	 36
 <u>USING R.I.C.E.(REAL-TIME INTERNAL CONFIGURATION EDITOR)</u>	 <u>37</u>
 GETTING STARTED WITH R.I.C.E.	 37
READING THE CURRENT SETTINGS	39
CHANGING SETTINGS	39
DISCARDING SETTINGS	39
SAVING THE SETTINGS	39
USING R.I.C.E. AS A DEBUG TOOL	40
COLOR SETUP & COLOR MIXING	40
Rough color setup	41
Color fine tuning	41
USER’S NOTES	42
TROUBLESHOOTING & FAQ	43

Introduction

This **Saber Controller** is the union of our evolutive saber sound module and our luxeon driver board we designed back in 2005. Driven by a single processor, this module features a perfect synchronization between light and sound effects with the possibility to setup each effect with parameters stored on our *SD-Config*[™] technology. The Petit Crouton is the little brother of our high-end saber controller Crystal Focus (CF). The PC version 3 is mostly based on CF v6.5 and features 16 bit sound playback, WAV format support, 3 different sound banks, blaster blocking, force push effect and more !

This particular release is called the PRIZM version. It is a variant of the Petit Crouton board on which the onboard current driver and power extenders were replaced by 3 direct drive channels (no current regulation) and that is designed to run exclusively on a single li-ion cell (3.7V).

Warning : You've just acquired an electronic board containing parts sensitive to ESD. Final wiring & assembly is under responsibility of the user with the appropriate tools and ESD protection.

If you're not familiar with ESD, please visit :

http://en.wikipedia.org/wiki/Electrostatic_discharge

Plecter Labs can not be held responsible for improper use or assembly of the Petit Crouton board.

High-Power LEDs (aka Luxeon[™])

DIY illuminated sabers have suffered for too long of the lack of a terrific and impressive blade retraction/ignition effect. EL wire technology did not allow this effect since it fades in and out in a homogeneous way all along its length, because of the phosphor composing the wire coating. MR/Hasbro Fx sabers found a workaround by using a 64 LED strip on a flexible PCB which makes the retraction effect by switching the LEDs by group of 8 but this setup remains very fragile.

The high-power LED technology allows a realistic ignition/retraction effect of the blade while keeping it almost empty and therefore not fragile when hit.

An additional feature of the luxeon driving section is the configurable generation of a flickering effect of the blade brightness. It's a random alteration of the light produced by the high-power LED suggesting energy variations for a more realistic result which is pretty close to the sabers seen in the movies. The effect is not a constant pulse but is more like a "candle effect".

The Plecter Labs direct drive (3 channels) section of the Petit Crouton PRIZM Board can drive up to 2A and works with any high power LED featuring a forward voltage (Vf) lower or equal to 3.8V. Luxeon, Rebel, Seoul, Prolight and Ledengin branded

LEDs have been tested successfully. Please note that as we're writing this manual, we cannot guarantee the use with ANY kind of high-power LEDs appearing in the market in the future.

Warning : High-power LEDs (such as the Luxeon brand LED, which is mentioned in this document) are *extremely bright*. They are considered "class 2 lasers"! You should neither look directly to the beam nor point someone with it when the blade is not attached to the hilt, just like a powerfull lamp or flashlight. Plecter Labs could not be held responsible for any bad use of high-power LEDs.

To avoid injuries and retina damage due to the high brightness of those high-power LEDs, simple "emitter plugs" can be built using a piece of blade tubing ended with some decorative greeblies.

Sound section

The Plecter Labs sound board is unique. It has been developed in the purpose of improving the quality of DIY sabers sound FX in a significant way. During too many years, sound modules were obtained from sacrificed toys and remained low quality. Master Replica FX sabers broke the line with better sounds and good dynamics. However, the low resolution motion sensors used as well as closed electronics made those boards impossible to adjust in term of sensitivity or sound contents.

We have monitored several attempts for building an embedded sound module playing custom & changeable sounds, often based on chipcorders. Using bulky parts, those were often unreliable and hard to fit in a hilt. Not to add those chipcorders were designed for digital answering machines, and therefore feature a bad restitution quality (voice sample rate).

Plecter Labs decided to process the internal motion sensors **and** the sound generation on the same board which requires some non-volatile memory. Second, we needed a simple way to upload or download sound contents or configuration of the saber through a simple and standard way.



To avoid any plugging problem with a small connector and an easy-to-loose cable, we opted for a high-end flash memory card in the SD format (now microSD).

Inserted in a USB card reader like the one we sell, the card is seen as a USB storage key and it takes a few seconds only to transfer files to or from the card, on Mac or PC, without the need of any custom piece of software.

Petit Crouton PRIZM V3.5 Features & Maximum Ratings

- Dimensions: 40x23.5x6.5 mm (with the microSD card).
- Power supply : 5.5 MAX / 2.5A (with a single die High-power LED). **SINGLE** li-ion cell (18650 or 14500) battery recommended.
- Idle current consumption : 9 mA (deep sleep mode)
- Speaker: 4 to 8 ohm.
- Audio output Power : 2W
- Accent LEDs : 4
- Accent LEDs pad current source : 18 mA max per pad
- Handles momentary or latching for blade activation
- Up to 6 selectable sound banks via Font Xchange™
- Up to 10 color profiles triggered by a switch action combo
- Blaster Blocking, Force, Force Clash™ and Lockup Fx
- Blade Flickering Fx
- Blade Shimmering on Clash
- Blaster Sounds
- Up to 4 boot sounds
- Up to 8 swing and 8 clash sounds
- Flash on Clash™ (FoC™) and FoC mixing selection
- Anti Power Off technology (A-POP™)
- 32 stage accent LED sequencer
- WAV file support
- True 16 bit, 22.050 kSamples/sec crystal clear DAC
- SD card support: up to **16GB (SD & SDHC)**, FAT16 or FAT32. Sandisk and Kingston brands preferred.
- FlexiBlend™ powered Color Mixing on 3 channels
- Configurable normal blade & FoC colors
- Real Time Configuration Editor (R.I.C.E.™)

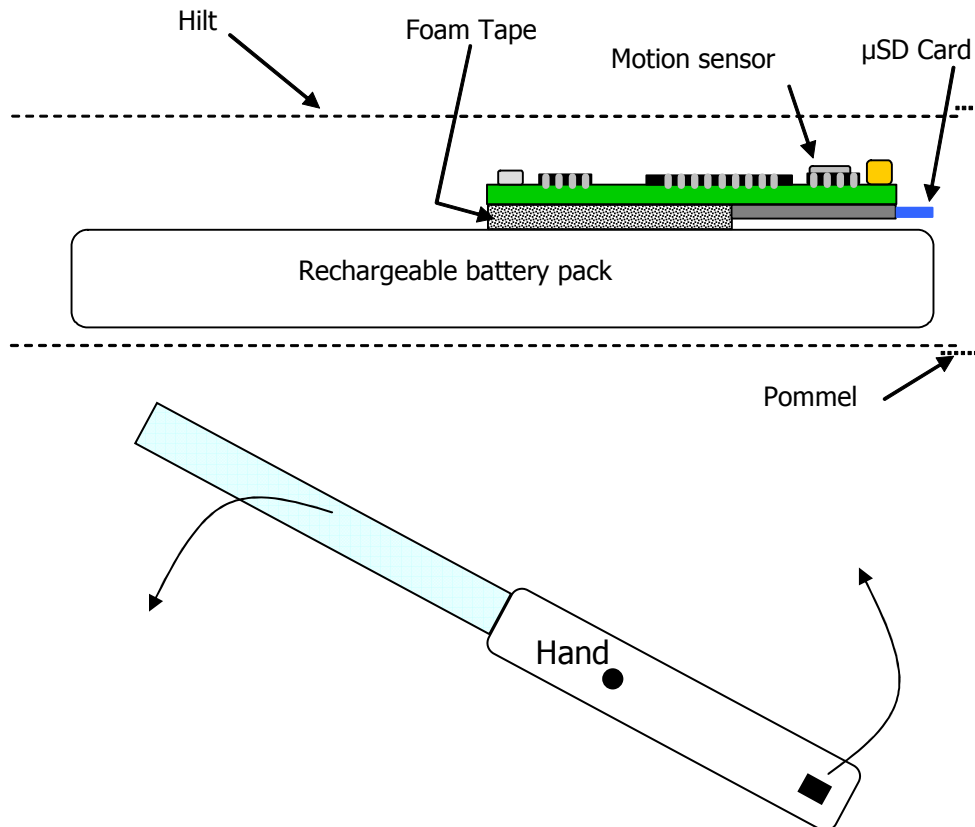
Placement & Installation

Ideally, the module is placed in the hilt so that :

- the motion sensor is at 1" or more from the spinning center of the hilt
- the SD card remains easily accessible.

Usually the pommel area is a good choice, however hilts made of 2 halves can have the board installed in the top side of the saber.

Securing the board can be done using double sided foam tape or a velcro strap.



Tools and Parts required to install/operate the module

- an ESD safe soldering station & soldering wire (60/40, 1mm OD or eq.)
- pliers (flat and cutting)
- a Digital Multimeter / DMM (strongly advised, so useful)
- a latching or momentary switch for the blade ignition, and a momentary switch for the auxiliary switch.
- wire & heat shrink
- rechargeable Batteries
- recharge port (canon 2.1mm socket)
- appropriate Battery charger
- a USB SD card reader accepting micro SD card or a regular SD card reader with a micro to regular SD card adapter.
- a computer
- a digital audio editor software handling WAV files if you wish to create your own sound fonts.

How does it work ?

The two main effects of the saber is the production of a sound when the blade is cutting the air (swing, producing a sort of Doppler effect) and the impact between two blades (clash). The motion sensor we use is capable of detecting rotation movement and shocks. The main difficulty is to make the proper difference between the two classes of movements. The sensor is digitized by a microcontroller, then analyzed in real-time and compared to a modelization of clash and swing gestures using low latency DSP techniques (now down to 10 ms).

The algorithm has many trimming parameters in order to be adjusted to the fighting style of each user or fighter, and also to each saber hilt design. As a matter of fact, each saber is unique and various interaction scenarios can be desired. A setup allows then to change the sensitivity to the swing and the clash, depending if the user wants a really verbose saber, or casual sound FX. Along the different versions we improved our gesture recognition algorithms which now have semi automated parameterization, the user selecting only basic thresholds and the general sensitivity. Moreover, default settings usually suit most users.

The swing gesture is a rotation of the blade leading it to cut the air at an average speed. The clash gesture is a sudden shock of the blade on an obstacle, or a hard shake of the hilt.

The saber setup is located on the SD card which also stores the sounds. A configuration text file is editable with a simple text editor such a windows notepad.

SD card contents, Sound Banks and Slots

Sounds are stored in the WAV format (16 bits, 22050 samples per second).

[The previously used RAW format is no longer supported]

WAV sound files must comply with the format above or they will be skipped during the boot, leading to sound gaps or board failure.

Petit Crouton PRIZM Version 3.5 can have up to 6 sound banks. This allows storing different “style” in the same saber. Each sound bank is stored on the SD card in the sub-directories (or folders) `bank1` to `bank6`. The contents of a bank is called a Sound Font

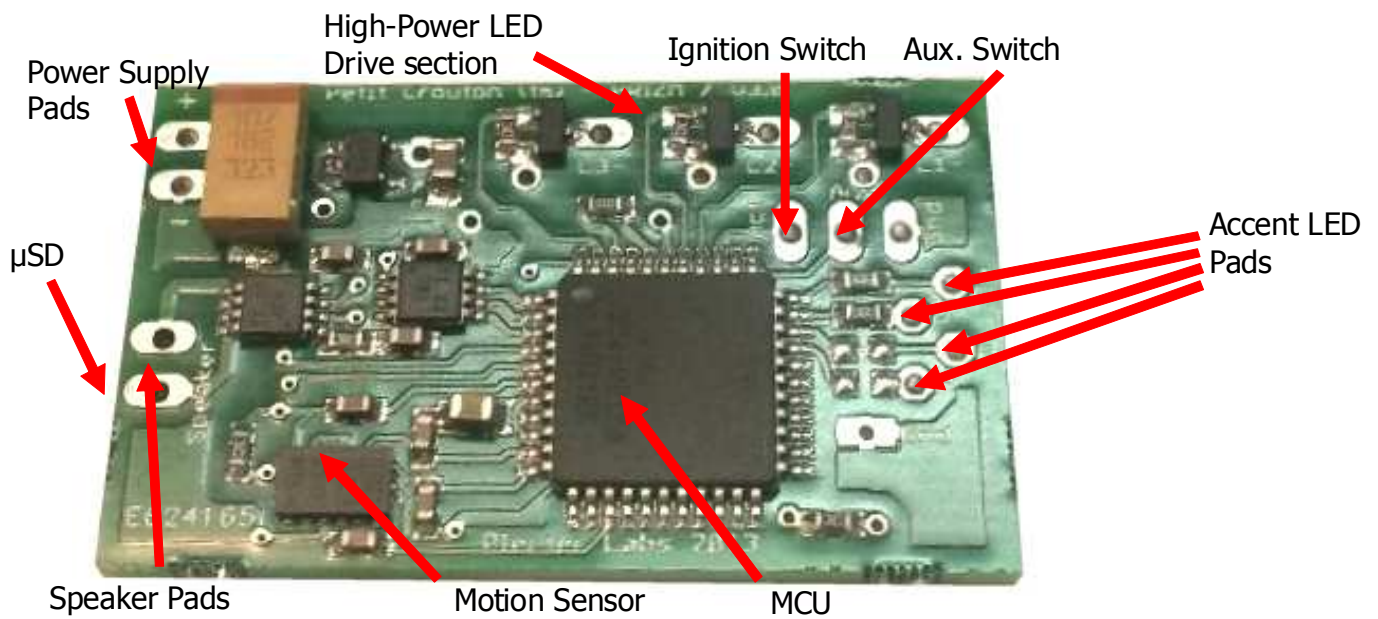
Each sound bank has 32 sound slots split as below :

- up to 4 boot sound (`boot[2-4].wav`)
- 1 power on sound (`poweron.wav`)
- 1 power off sounds (`poweroff.wav`)
- continuous humming (`hum.wav`)
- 8 clash sounds (`clash1.wav` to `clash8.wav`)
- 8 swing sounds (`swing1.wav` to `swing8.wav`)
- 4 blaster blocking sound (`blaster.wav` to `blaster4.wav`)
- 1 blade lockup sound (`lockup.wav`)
- 1 force effect sounds (`force.wav`)

When the power supply voltage is applied to the board, our board “boots” and plays a little logo sound to notify the user, just like a digital camera. This little logo makes sure the Petit Crouton Saber Core started properly and it gives a special identity to the saber and to the loaded sound font. This sound can be of course customized. If the boot sound `boot.wav` is not on the SD card, a little beep is played instead. If you don’t want any sound when powering the module, create a WAV sound file with 100 ms of silence.

The sounds **must** be all there on the SD card and be named properly (lower case) to have the module operating properly. Same thing for the configuration files (`.txt`). In case of losing files, the original package of sounds and configuration file are available from Plecter Labs on request.. We advice the user to keep all its sound and configuration files in specific folders on the hard disk on the computer so that changing the saber’s contents remains easy. Use some explicit naming of the folders so that you can easily remember what the sound font and configuration files are doing, for instance [`very_sensitive_dark_lord_saber`].

Board Overview



User's Notes

Getting Started with Petit Crouton

The board has been designed so that the user can enjoy an “out of the box” experience. The default package of the SD card contains 6 sound banks with ready made configuration files and accent led sequence files.

The `switch` parameter is set to 1 by default, which corresponds to a **normally closed (NC) latching switch**. This way, the user doesn’t need to hook up a switch to the board, the open contact on the activation pad tells the boards to start just after power up, allowing the user to test the board with a minimal soldering job of 6 connections: power supply, speaker, high-power LED.

Further install of the board in the hilt and customization of the PC board will require the user to change the parameters in the configuration files. Keep in mind that, especially if this is your first PC, and due to the high configurability of the board, you’ll spend quite some time on adjusting the parameters to reach the desired look & feel. The SD card slot should remain accessible during that process and possibly once the saber is completed too.

Wiring and Operating the Module

The board must be powered with an appropriate battery pack. We highly recommend the use of good quality **li-ion battery** packs made of 14500 or 18650 cells **and including protection PCBs**. The AW brand makes superior quality batteries while the Ultrafire remains a cost effective solution.

Unless you have a convenient way to open the hilt and access the inside of the saber (Graflex base for instance), we strongly recommend the use of a directly connected battery pack (with a “recharge port”) vs. removable cells. Moreover, for dueling sabers, directly wired battery packs have more reliable connections compared to spring action battery holders.

A 2-cell li-ion will provide a nominal voltage of 7.4V to the board. The board isn’t compatible with a 3-cell solution without some modifications of the electronics.

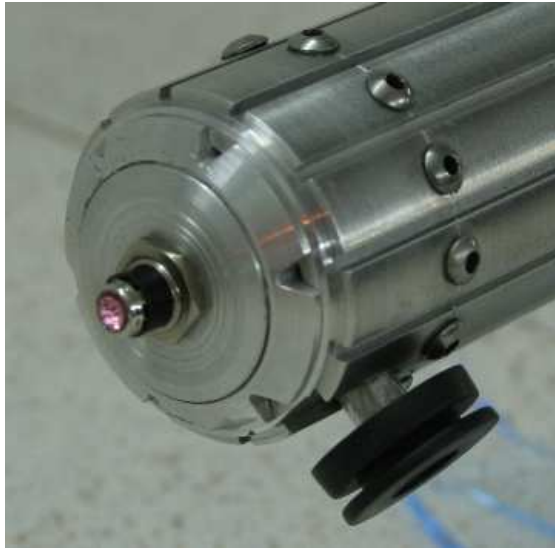
Ni-MH battery packs are simply not recommended since they have a bigger energy storage/volume ratio and the cost of li-ion cells isn’t an issue anymore.

General Power Switch & Recharge Port

Despite the PC board has a very low idle current use when the blade is off and board is in deep sleep mode, long term storage of the hilt on a shelf or display case requires the electronics to be fully shut off. To avoid the use of an additional general power switch, we use the recharge port for that very purpose. A pin 2.1 mm “Canon” socket is a popular choice. Two of those pins are connected when nothing is inserted in the socket. Contact is disrupted when a plug is inserted.

Along the years, the “kill key” technique has been developed: a fake plastic plug is decorated to look like an actual part of the hilt. When inserted, it cuts the power supply to the board in the recharge port. Of course, the port recharges the internal battery pack when an actual charger plug is inserted.

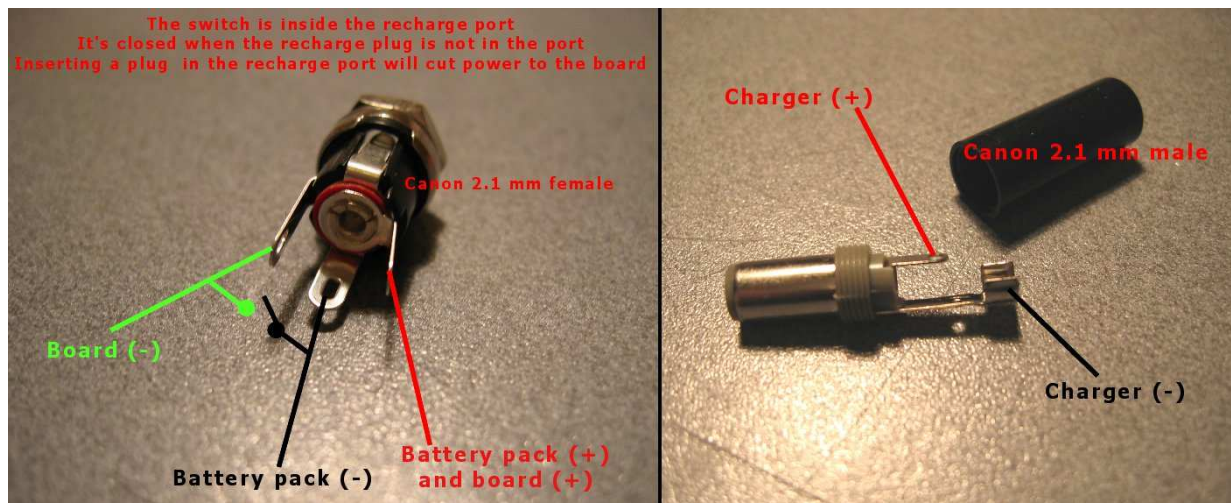
Below, an example of a decorative kill key (July 2010)



The Kill Key must be made out of a non-conductive material (PVC, Nylon etc).

Here's the usual wiring of the recharge port. Please note that not all recharge ports have the exact same pinout. User must understand the principle of wiring a recharge port and must be able to identify the different pins of a socket.

The idea is fairly simple: the positive of the battery pack goes to the recharge port central pin (referred as **tip**) and the to the positive of the board. It's not affected by the kill key. The negative of the battery pack goes to the pin of the recharge port that is connected to the outer **sleeve of the socket**. The last pin, referred as **switched negative pin** and goes to the negative of the board.



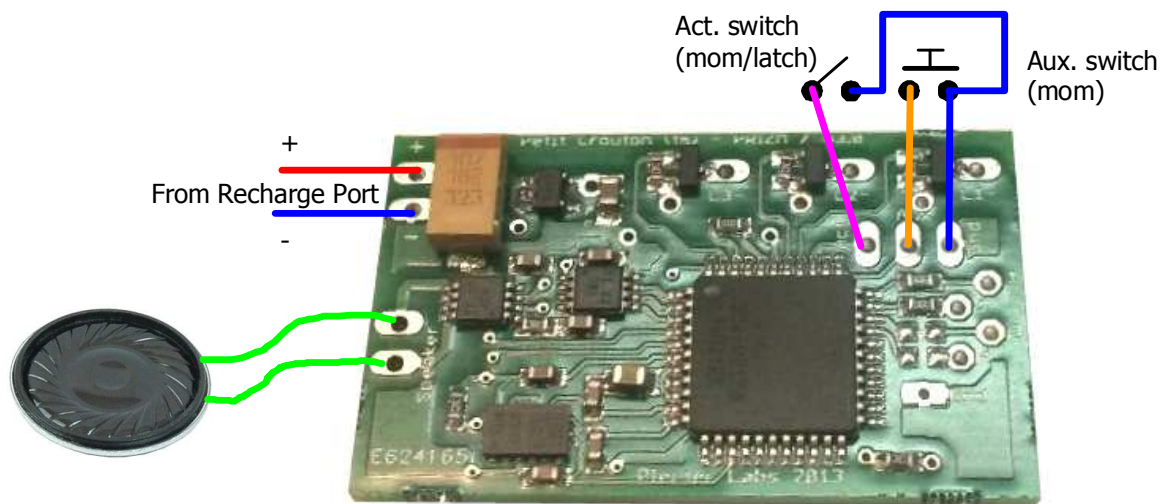
When nothing is inserted in the port, the negative of the battery pack is internally connected to the switched negative tab, hence powering the board. When a Kill Key is inserted in the port, the negative of the board is no longer connected to the negative of the battery pack: the board is fully powered down. When a charger plug is inserted in the recharge port, the charging voltage is reaching both leads of the battery pack

while the negative of the board is still unconnected from the circuit, preventing damages to the electronics and ensuring only the battery pack is connected to the charger for proper charge.

In the previous picture the green-black drawn switched doesn't need to be wired per say, it only illustrates the recharge socket internal switch.

General wiring

The board doesn't need so many connections for basic operation. Aside of the recharge port (if needed, a removable single cells might not request one) / power supply detailed above, only a pair of switches, the high power LED and the speaker are required to be soldered to get 80% of the features the PC board proposes.

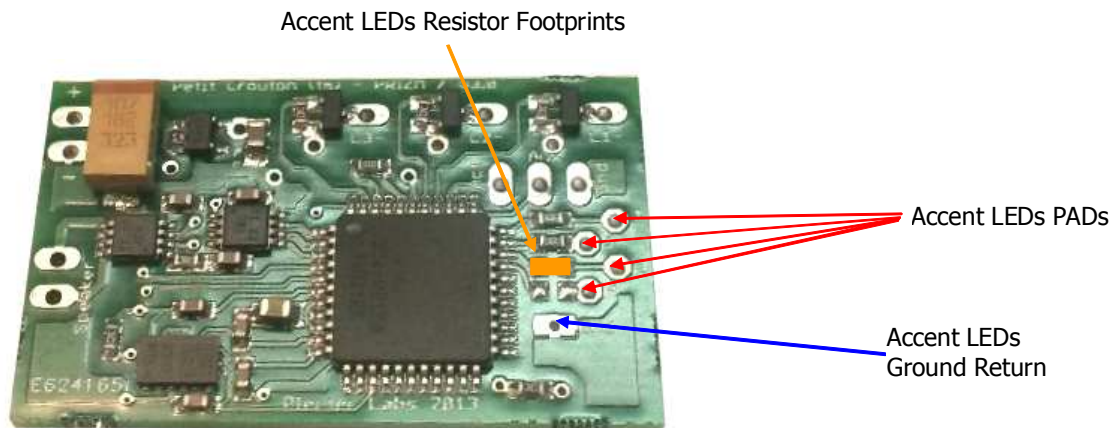


User's Notes

Animated Accent LEDs

There are many ways to “pimp” your saber hilt using additional small LEDs further referred in this document as **Accent LEDs**.

Petit Crouton features a 32 stage sequencer that allows the user to setup a blinking animated sequence for up to 7 LEDs. The board outputs 3.3V / 18mA max per accent LED pad. User must ensure the used accent LEDs have a forward voltage (Vf) lower or equal to 3.3V.



As a space saver PC embeds the footprints for SMT resistors. The footprint is made for 0603 resistors.

To install those resistors on the board, pre-tin one pad only, then grab the resistor with a pair of sharp tweezers, slide it against the pre-tinned pad, heat up the joint, wait for it to cool down, then solder the other side of the resistor.

If the user prefers to use classic resistor with leads, the SMT footprint must be bridged: tin both pads first then add a bit of solder while the soldering iron tip is right in the middle of the pads. Some stripped wrapping wire can also be used to achieve the bridged connection.

On the picture above, the red arrows point to the positive pads of the accent leds, use small gauge wire to send those signals to the positive of the LEDs. Flat/Ribbon cable can be very handy for that purpose. Then all negatives of the LEDs return to a single pad pointed by the blue arrow (ground return). The main negative of the board can be also used as the accent led ground return.

Calculating resistors for LEDs

$$R = (V_{\text{supply}} - V_{\text{led}}) / \text{LedCurrent}$$

In our case, V_{supply} is the voltage the board provides to power the accent LEDs, ie 3.3V. The V_{led} is the forward voltage of the LED, usually referred as V_f in the

datasheet. The led current has to be decided by the user, depending on the brightness and the maximum rating of the used LED. 5 to 15 mA are fairly common for most accent LEDs.

As an example, let's consider a 1.6 volt LED (red) at 10 mA

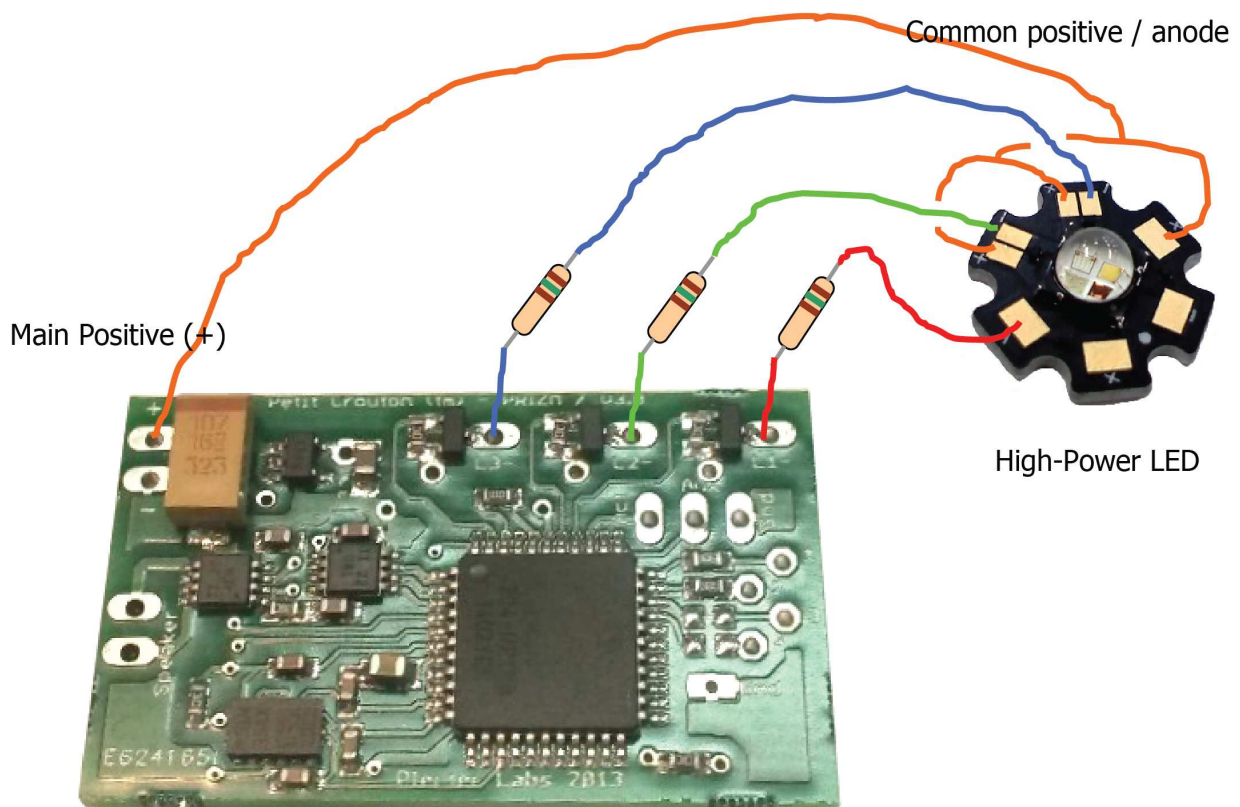
$R = (3.3 - 1.6) / 0.01 = \underline{170 \text{ ohm}}$ (\rightarrow 150 ohm in the classic E12 resistor serie)

Be sure not to drive too much current in the LED (18 mA max). If you wish a good brightness with a low current, use high efficiency LED (generally coming in a transparent "crystal" casing).

Please see further in this document for the sequencing of the accent LEDs.

High-power LED wiring

PRIZM is dedicated to color mixing, so we will cover only the wiring of a multi-die LED in this paragraph. 3-die LEDs are obviously proposed for PRIZM, but a 4-die LED is also possible, having 2 dices wired in parallel on a particular channel.



The example above shows a power LED with individual dice. Common anode LEDs can be wired just the same.

In most cases a resistor is required on each die, especially with the rise of modern LEDs with low forward voltage. In certain special cases, only a drive adjustment is requested to limit the current in the LED.

High Power LED resistor calculations

There are many online resistor calculators but here are the calculations for both resistor and wattage. They aren't difficult and it's important for the user to understand and memorize them.

Stunt Upgrade

If the user is adding sound to an existing saber and is already using a resistor, it can be used as it is. Beware though, this applies only if the battery solution is already Nano Biscotte compliant (single cell, < 5.5v). If the saber was made with a 2 cell pack, the resistor needs to be changed, see calculations below.

Resistor calculation

The Ohm's Law gives us $R = U / I$. R is the resistor we're looking for, U is the voltage across the resistor (unknown for now) and I is the current at which you want to drive the LED.

The Voltage across the resistor is simply the difference between the supply voltage (the cell voltage) and the LED voltage (also called forward voltage or Vf):

$$R = (V_{\text{supply}} - V_{\text{led}}) / \text{MaxLedCurrent}$$

The LED voltage is often mentioned in the datasheet or in the product page. The LED current is chosen by the user, depending on the battery solution and the LED ratings. For instance, even if a LED can take up to 1.5A, a regular 14500 cell will not be able to deliver such a current due to its limited maximum discharge current. A 18650 cell will be able to deliver such a current.

Let's take for instance a red LED with a 2.9V forward voltage, driven at 1.5A with a 18650 cell (3.7V)

$$R = (3.7 - 2.9) / 1.5 = 0.53 \text{ ohm (the closest resistor might be 0.56 ohm)}$$

Wattage calculation

The Ohm's Law gives us $P = U \cdot I$ with again U being the voltage across the resistor and I the current in the circuit, and we now know both of them.

$$P = (V_{\text{supply}} - V_{\text{led}}) * \text{LedCurrent}$$

In the previous example:

$$P = (3.7 - 2.9) * 1.5 = 1.2 \text{ W}$$

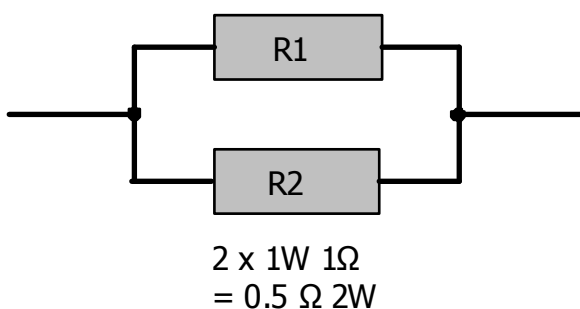
Ideally a 1.5 to 2 W resistor is recommended in this case.

Resistor bargain

As seen in the example above, a 0.56 ohm resistor might not be easy to find. Also, the user should not focus on the absolute precision of the resistor value if he is still within the LED specifications and ratings. Instead of looking for an expensive source of a rare resistor, the user can craft his own using 2 (or more) resistors.

Without going too much into boring details, there's a simple rule of the thumb saying that wiring 2 identical resistors in parallel divides the resistor value by 2 and multiplies the wattage by 2.

In our previous example, a 2W, 0.5 ohm (ish) resistor could be obtained with 2 x 1 W, 1 ohm resistor.



Use either ceramic or thick carbon film resistors only as your high-power LED resistor (NO metal film). When combining them in parallel, ceramic resistors can be directly wired directly. In the case of carbon film resistor, I recommend individually wrapping them in heat-shrinking tube to avoid shorts if their paint chips (which sometimes happens if they are touching each others)

If the resistor calculation ends up providing either a very small resistor value or a negative value (!), you might skip the use of a resistor and adjust the LED drive instead. Refer to the color profile section for more details.

Main Configuration File

The `config.txt` configuration (stored in each bank) file is a simple text file to be edited with windows notepad.

It is composed of 17 parameters that **must all** be present in the file (or then in the override file, see further). Otherwise, the module will use default parameters for the missing parameters.

The text file accepts comments on a stand alone line (not mixed with a parameter line). The comment symbol is the C language double slash `//'` as the very first characters of the line.

PC now also has another style of comments using the Emacs script syntax `'##'`.

To "comment" has 2 kinds of use. One is to disable a line like in the example below:

```
## switch=2  
switch=1
```

The other use is to leave a comment or a note to remember something about the configuration or the soundbank, like in this example.

```
## beware, ch1 drive should not exceed 900  
color0=700,400,100
```

the double slash `//'` comment is now reserved as the single comment exported to the Real Time Configuration Editor (R.I.C.E.™). Use the `//` comment to name the configuration to reflect its "theme" like:

```
// Bank1 - Novastar sound font
```

Certain parameters are integers, others are floating point numbers. Format must be respected: even for a round value like `'1'` for a floating point parameter, `'1.0'` has to be entered.

To modify the file, insert the SD card in the USB card reader, and then browse the contents with windows file explorer (on `E:` for instance). Double-click on file `config.txt` : the notepad opens. You can directly save the file on the SD card. Once the configuration is over, simply remove the card from the reader after having it "ejected" (right click on the reader device in windows explorer, contextual menu, eject). Put the card back in the saber and test you new setup!

The configuration file **MUST BE LESS than 1024 bytes**. If the size is bigger, the file will be skipped without further analysis and defaults parameters will be used. The basic configuration file is about 200 bytes, and 450 bytes for the override file, with a few comment lines for an easier reading. Make sure not to add too many comments in the file. If you are not sure of the file size, check it in Windows file Explorer, with a right click on the file, then "properties" in the contextual menu.

Make sure you have no space characters at the beginning of the line, or between the '=' sign and the value of a parameter.

Parameters and fine tuning the saber

The configuration file includes a set of parameters dedicated to the sound section of the controller and the gestural / motion detection (both being linked). A second set of parameters handles the behavior of the high-power LED. Some parameters influence both categories, since visual and sound effects are in tight relationship intrinsically. All parameters are lowercase.

Certain parameters involve time / duration / delay. We tried to normalize those parameters to a single unit: a multiple of 2ms. Unless otherwise indicated, that's what is used to define those timing parameters and it matches the internal clock of the board.

Motion & Gesture detection parameters :

Motion recognition is processed using complex low latency DSP algorithms, however, most of the parameters used for those are internally computed so the user only has to setup a few thresholds & sensitivity level as high-level parameters.

The overall sensitivity of the board is controlled by the "i" parameter. If you want a more "verbose" saber while you're satisfied of the swing / clash discrimination, just increase a little bit that parameter, turning it to 55 instead of the default 53 value.

Swings and clashes are separated using 3 thresholds. A swing motion must exceed **ls** (low swing) and must remain under **hs** (high swing). A clash must simply exceed **lc** (low clash).

It's important to maintain a dead-zone between **hs** and **lc** to maximize the quality of the motion detection.

Default parameters have been setup for you and usually, only the "i" parameter has to be touched up.

However, some special cases can be considered:

- The saber is not sensitive enough to the swing (placement in the hilt too close to the spinning center, spinning style etc) : decrease parameter **ls** of one point of two. Don't decrease **ls** too much or undesired swing sounds will be triggered.
- When the saber cuts the air too fast, the swing is not triggered (it's in the dead-zone). Increase a little bit parameter **hs** (or reduce the speed of your swings).
- The saber often produces a clash sound when the user executes a swing: parameter **hs** is probably too high and too close to parameter **lc**. If the clash sensitivity is correct, decrease parameter **hs**.
- The saber is not sensitive enough to clashes: decrease parameter **lc**.
- The user wants the clash to happen **only** when blade is smacked very hard: increase **lc** (keep it under 1023).

Rule of the thumb for a proper configuration process: modify only a single parameter at once and work separately on clash and swing parameters. Fine tuning the module might take some time, but a good configuration will lead you to a very satisfying interaction with the saber.

Gesture flows & priorities :

Our gesture analysis is so fast that sounds could be chained one after the other at light speed! We therefore have to slow down the module because too many swing sounds played in a short time are not so realistic. For that reason we implement gesture flow limiters for swing and clash sounds.

Clash sounds generally have the priority over all other sounds except in the following cases:

- lockup is engaged : no other sound will be triggered until the aux. switch is released
- a blaster sound is triggered. Swings cannot interrupt it. A clash can interrupt it

A clash sound can interrupt a swing sound even if the swing flow limiter is engaged (just after a swing was triggered).

A clash sound cannot interrupt a previously triggered clash sound if the clash flow limiter is still engaged (delay for triggering another one hasn't expired).

A swing sound cannot interrupt a previously triggered swing sound if the swing flow limiter is still engaged (delay for triggering another one hasn't expired).

A swing sound can never interrupt a clash sound **if the clash flow limiter is still engaged**, and whether or not the swing rate limiter is engaged.

If the clash flow limiter has expired, and even if the clash sound is still playing, a swing sound can interrupt it.

- **swing** [0-500]: swing rate flow limiter. Delay during which swings cannot be furthermore triggered.
- **clash** [0-500]: clash rate flow limiter. Delay during which clashes cannot be furthermore triggered.

Sound parameters :

- **vol** [0-4] : digital volume setup. 0 mutes the sound output, 4 is the maximum volume.
- **beep** [0-127] : Sets the volume of the beeps emitted by the unit during reboot.
- **shmrld** [10-500]: duration of the shimmering effect of the high-power LED during a clash. Make sure this duration is not too much longer than the associated sound to keep a nice result.

- **shmrp** [5-25]: periodicity of the light bursts during the clash effect. A slow period will produce tight bursts.
- **shmr** [0-25]: random value applied to the periodicity of the light burst during a clash effect. Allows having bursts that are not regularly spaced in time which increases the realism. For instance, a period **shmrp** of 20 and a random value **shmr** of 10 will produce a period between two bursts varying between 20 and 30 (ie 40 and 60 ms).
- **shmr%** [0-100],[0-100] : shimmer effect depth. Defines how the shimmer will "dig" the defined brightness during a clash or a lockup effect. A static flash is achieved by leaving that value to 0. That parameter is actually composed of 2 numbers separated by a comma to define the range to apply to the brightness modification during the shimmer.
- **focd** [0-500]: Flash on Clash™ (FoC™) duration. Used when an extra LED die is wired to the board with a Power Xtender™ circuit and the Flash on Clash™ Pad.
- **focp** [5-25]: periodicity of the light bursts during the Flash on Clash™ effect. A slow period will produce tight bursts. Similar to the **shmrp** parameter.
- **focr** [0-25]: random value applied to the periodicity of the light burst during a Flash on Clash™ effect. Similar to **shmr** but applied to FoC™.
- **foc%** [0-100] , [0-100]: Flash on Clash effect depth. Defines how deep the FoC brightness is dug for "on-top" (or non-mixed) FoC di(c)e. Also composed of 2 comma-separated numbers, like the shimmer depth (see above).
- **switch** [0-2]: selects if the saber is activated by a normally-open or a normally-closed switch. Certain "push-on push-off" switches are more practical and more reliable when released for activating the saber. Other switches might simply have an "inverted" logic (normally closed contact). When **switch** is set to **1**, the saber lights up when the electrical contact of the switch is closed and conversely when **switch** is set to 0. If you wish to use a **momentary** switch for the blade activation, set **switch** to 2.
- **offp** [0-1]: Anti power off protection (A-POP). To avoid accidentally powering off the saber, especially when using a momentary button for activation, we added a power off protection. When this parameter is set to 1, the user must press the activation button and confirm with the auxiliary button. It is not necessary to press both buttons at the same time, keep the activation switch pressed first, then press the auxiliary switch: the blade goes off.
- **offd** [0-65535]: Anti power-off delay. An alternative to A-POP™ defining how long the user must press the activation switch before the blade goes off. A value of 65535 corresponds to 2min12, which is useful if you just upgraded an ultrasound saber with a PC and you're feeling some power-off scheme nostalgia. Values from 70 to 200 (140 to 400 ms) are relevant and very efficient for just ensuring the saber won't be turned off accidentally. **Set to zero if you use an anti power off protection with parameter **offp**.**
- **qon** [0-3000]: "quick-on". Allows having the blade ignited in a specific amount of time rather than matching the duration of the power on sound and should not exceed the duration of any power on sounds.

- **qoff** [0-3000]: same thing as above, but for the blade retraction.
- **valsnd** [0-1]: enables or disables the repeat of the selected soundbank description sound in the vocal menu. When disabled, it saves time in the selection process. Also, when disabled, the aux. switch confirmation process for rebooting the saber is disabled (see "rebooting the saber" paragraph).
- **lockup** [0-1000]: our module features an auxiliary switch to trigger additional sound/visual effects. A short pressure on the switch generates the blaster effect (the saber blade stops a blaster ray), plays one of the blaster sounds. A longer pressure (maintained) triggers a blade lockup effect: while the switch is pressed, the sound `lockup.wav` is played in loop with some shimmering applied to the high-power LED. The parameter **lockup** specifies the duration of the delay before triggering the lockup effect. A short value (50 to 100) will trigger the effect almost immediately: to trigger a blaster effect, the user will have to release the button quickly. Conversely, a higher value will leave more time to produce a blaster blocking feature.
- **resume** [0-1]: Enables the hum resume feature. Instead of starting the hum from the beginning, the sound will resume from the position where it has been interrupted within a ballpark of 10 ms.

High-power LED parameters :

- **focmix** [000-111] : Defines how the led dice are either mixed during the Flash on Clash or added "on top" the main blade color (legacy CFv5 FoC effect). See further in this document about that specific topic.
- **flks** [0-20]: speed of the energy variation / flickering effect of the blade. A high value produces a damaged saber effect while a small value generates subtle energy changes. The value 0 disables the effect (static blade).
- **flkd** [0-100]: depth (in %) of the energy fluctuation effect, i.e. the range over which the LED brightness will be affected during the effect. A low value does not modify the energy very much while a high value « digs » big steps of light intensity. To be used with the parameter **flkrs**.
- **fade** [0-3000]: color profile transition time. Must be lower than the Force sound duration

The "Override" configuration File

The `override.txt` file present in the root directory of the SD card contains parameters that will override any settings present in the configuration files of each bank. It also contains settings that should really be global for the saber and not specific to each bank such as the motion detection settings and switch type.

This comes very handy to define a global setting for the blade flicker, or other aspects of the saber which once decided don't have to be copied and pasted in all banks. The bank configuration will be always parsed in first place, then settings present in the override file will replace the bank settings. We suggest to use this file with caution as the user will tend to forget which parameters are overridden, then trying to change the settings in the bank config will remain with no effect.

The override file is by default populated with 13 parameters (plus the color profiles) concerning the general behavior of the saber and not which aren't really particulars to a certain sound font. That split of parameters is conserved by RICE when saving either the current bank config file or the override file. Therefore, in the case of manual editing of certain parameters or moving a parameter from the config to the override (and conversely), special care must be taken .

User's Notes:

Color profiles

Profiles definition

PRIMZ can embed up to 10 color profiles defining both a blade color and a Flash On Clash (FoC) color. Those profiles are stored in the `override.txt` file with the keyword **color** and **fcolor**. Each profile (for blade or FoC) is composed of 3 drive values for their respective channel #1 2 and 3.

Each drive value can be in the range of 0 (led die off) to 1023 (led die at its maximum current limited by the external resistor)

The board is provided with 10 default color profiles

```
## Color profiles
color0=0,940,0
fcolor0=940,430,0
color1=0,0,940
fcolor1=900,0,940
color2=940,0,0
fcolor2=0,630,400
color3=0,830,400
fcolor3=700,940,0
color4=840,0,630
fcolor4=0,0,940
color5=940,430,0
fcolor5=0,940,0
color6=740,930,0
fcolor6=0,740,600
```

If the user prefers to not use the color profiles, only one has to be left to define the default color of the blade.

When edited / adjusted via RICE, the profiles are always stored in the `override.txt` file. However, it is possible to manually remove the color profiles from the override file and create different sets in each bank, obtaining various shades and profiles in each sound font. Be aware that this profile organization will be discarded and reset if the board is further configured via RICE.

Profiles browsing

To initiate a color change and move to the next profile, the user must press the aux. switch and then, ***before*** the lockup timer elapses (see the **lockup** parameter) press the activation switch. That combo action is easier to achieve with a momentary activation switch, however it is also possible to trigger it with a latching activator. Once triggered, the color starts to change and the `force.wav` sound is played.

Drive adjustments

On direct drive boards, no current control is provided to the LED. The maximum current to a die is limited by an external resistor. From there, the board acts as a color mixer between zero and that maximum current to realize the desired blade (or FoC) color. Those channel drive values are set in the color profiles.

There are always special cases and exceptions. One is to end with a resistor value calculation that gives an extremely small value (<0.5 ohm).

You can always wire several resistor in parallel to reach the right value but sometimes, it's simpler to just limit what the maximum drive will be on a particular channel, so that the current doesn't become excessive on that die.

The drive values present in the **color** and **fcolor** parameter defines the max drive that is applied to the high-power LED to obtain a certain color. Set to its max (1023) and using no resistor on the die, it will apply the battery voltage

Let's take a green LED that has a forward voltage of 3.45V @1A. It is really close to the battery voltage. The resistor calculation remains the same

$$R = (V_{cell} - V_{led}) / I_{led}$$

$$R = (3.7 - 3.45) / 1 = 0.15 \text{ ohm}$$

On the one hand, such a small resistor is unpractical to source and tolerance will not lead to an accurate current limitation. On the other hand, not using a resistor will overdrive the LED which might fry or, in practical conditions, it will create a current spike overloading the battery which then has its voltage dropping below the board requirement, usually leading to errant, unwanted swing sounds being played.

The board (direct) driver can act as a linear, configurable resistor. When getting such a small resistor value (below 0.5 ohm, AND $V_{led} < V_{cell}$), the drive can be calculated as follows:

$$drive = 1023 * (V_{led} / V_{cell})$$

$$\text{in our case, } drive = 1023 * (3.45 / 3.7) = 954$$

That means that the maximum drive value for that die in the color profile must not exceed 954. The max drive value can be also fine tuned using an amp-meter in the high-power LED circuit: adjust the drive until you reach the exact average current required for your LED. Never set the max drive under 80% of the maximum value (820) without using an external resistor.

The user must also understand that the forward voltage of an LED or die cannot just be "guessed" from the spec sheet of the LED which only provides an average value. For best performance, the LED Vf must be measured using a bench power supply

Resonant chamber

A little 28mm diameter speaker cannot compete with some Hi-Fi stereo system, neither in term of quality, nor loudness. However, in order to maximize the volume of the saber sound, a resonant chamber can be created in the hilt (same idea than the bass reflex in a closed speaker).

To achieve this, make sure you have 1 or 2 cm of space (and therefore air) between the speaker and the hilt cap or pommel. The saber is generally not totally full, the so internal space of the hilt will make the back resonant chamber for the speaker which will seriously contribute in increasing the volume.

This amplification with the resonant chamber technique is selective for a certain range of frequencies (the bandwidth): the length of the chamber will determine the **timbre** of the saber. Try different configurations with an empty camera film box until you find the desired sound coloration. To check the obtained volume before putting the whole electronics in the hilt, circle the loud speaker between the thumb and the first finger and curve the palm of your hand: it will create a quite accurate simulation of a short frontal chamber and a bigger back resonant chamber.

For comparison purpose, a good resonant chamber and a volume set to the maximum produces a sound volume twice to three times louder than a Hasbro™.

Browsing the sound banks – rebooting the saber

To browse the sound banks and select one, the user must use the auxiliary switch. After the board has played the boot sound and while the blade is off, press the aux. switch. After one second, the board will emit a beep. A second pressure on the auxiliary switch will reboot the board and will jump to the next sound bank. If the user was in bank 3 (and there are only 3 banks on the card), the board will automatically go back to bank 1.

The double press system was initially designed to be fail-safe in case of an accidental press on the aux. switch. If the user prefers to jump to the next font right after the first press, set the **valsnd** parameter to **0** in **override.txt**.

Creating Your Own Sounds

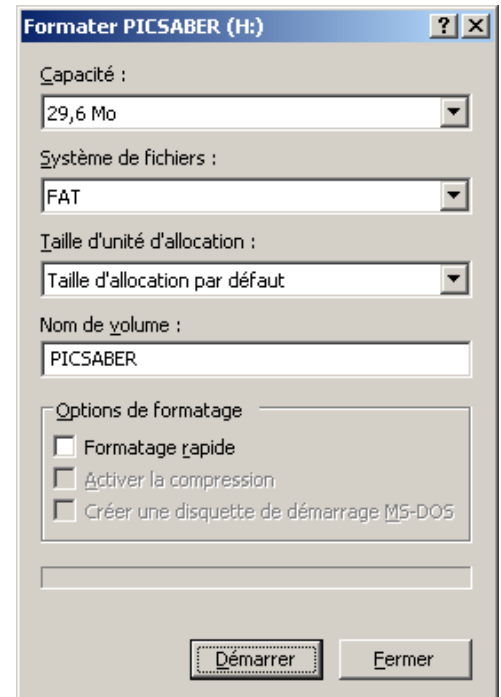
It's a great personal satisfaction to build up your own soundfont. Please read our online tutorial about that particular topic.

Installing a Sound Font on the SD card

The one-stop shop for sound fonts is www.saberfont.com. There is a growing number of sound fonts available at that website including all the legacy fonts from the Novastar Sound CDs (NSCD), providing so many sound “universe” for your favorite weapon. The board comes with 3 sound font installed in the different sound banks but of course you might install others.

Although editing the configuration files (text files) can be done directly onto your microSD card, changing the SOUND FILES (.wav of ANY kind) requires you to format the microSD card! This is not a recommendation, it is A REQUIREMENT: YOU MUST FORMAT THE SD CARD prior to changing any sound files!

1. First thing: backup the SD card on the hard disk. Easy to do, however most people don't bother doing so and end loosing their initial configuration.
2. Unzip the sound font archive in a directory of your hard disc.
3. Take all the WAV sounds. If the sound font zip archive comes configuration files that are supposed to match the theme of the sound font, you can take it as well, but make sure the settings won't harm anything (especially the high-power LED current).
4. Copy the files in desired sound bank and overwrite the files
5. Copy the whole SD card (Ctrl+a, Ctrl+c) to a temporary folder of the hard drive.
6. Format the SD card in FAT (FAT16 or FAT32)
7. Select the whole content of the temporary directory on the hard drive and copy it in one run to the SD card (Ctrl+a, Ctrl+c, Ctrl+v on the SD drive).



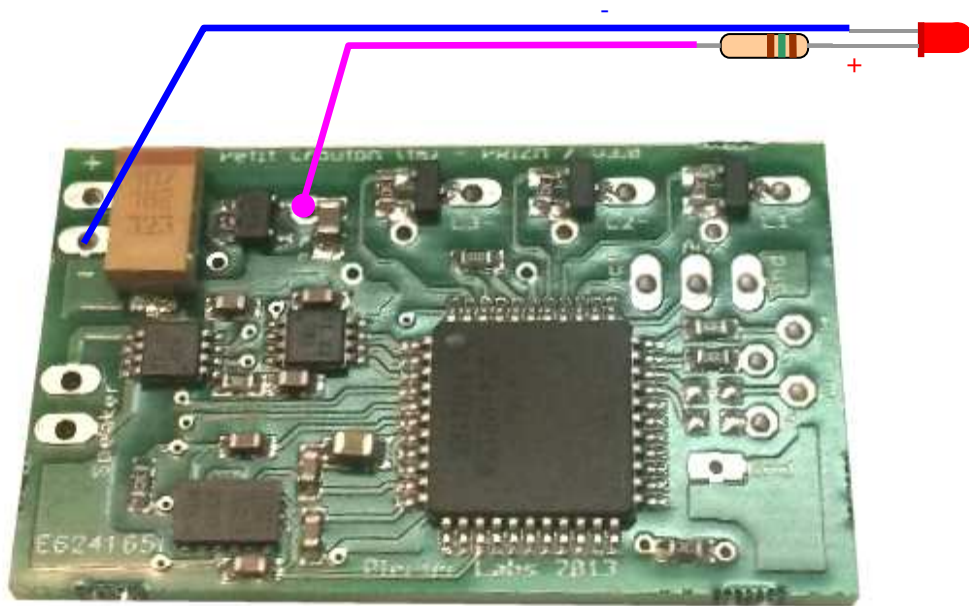
If you wish to try different sound fonts quickly, copying back and forth all the files will soon be boring and time consuming. Start by configuring your board to use bank1 with a working SD card. Use the vocal menu to select that bank. Then, on the hard drive, make a temporary file structure of the SD card that has only the bank1 directory and nothing else. Use that directory to test the different sound fonts. This way you will transfer only a limited number of files. Once the fonts have been selected, you can store them in the different sound banks and start adjusting the parameters, which you can now do in real time with R.I.C.E. (see further in this document).

Advanced Wiring & Usage

Wiring a general Power-On Indicator / Accent LED

It's sometimes useful (or good looking) to have a LED lighting up as soon as the kill key is removed. Of course, 1 of the 7 accent LEDs could still blink as idle mode indicator, but a permanently on LED could be nice to simply indicate the battery pack is working properly. This is also very nice to have indicator lighting up the internal ring of a chromed anti-vandal switch.

The PC board provides a regulated 3.3V pad for that purpose, shown in purple in the illustration below.



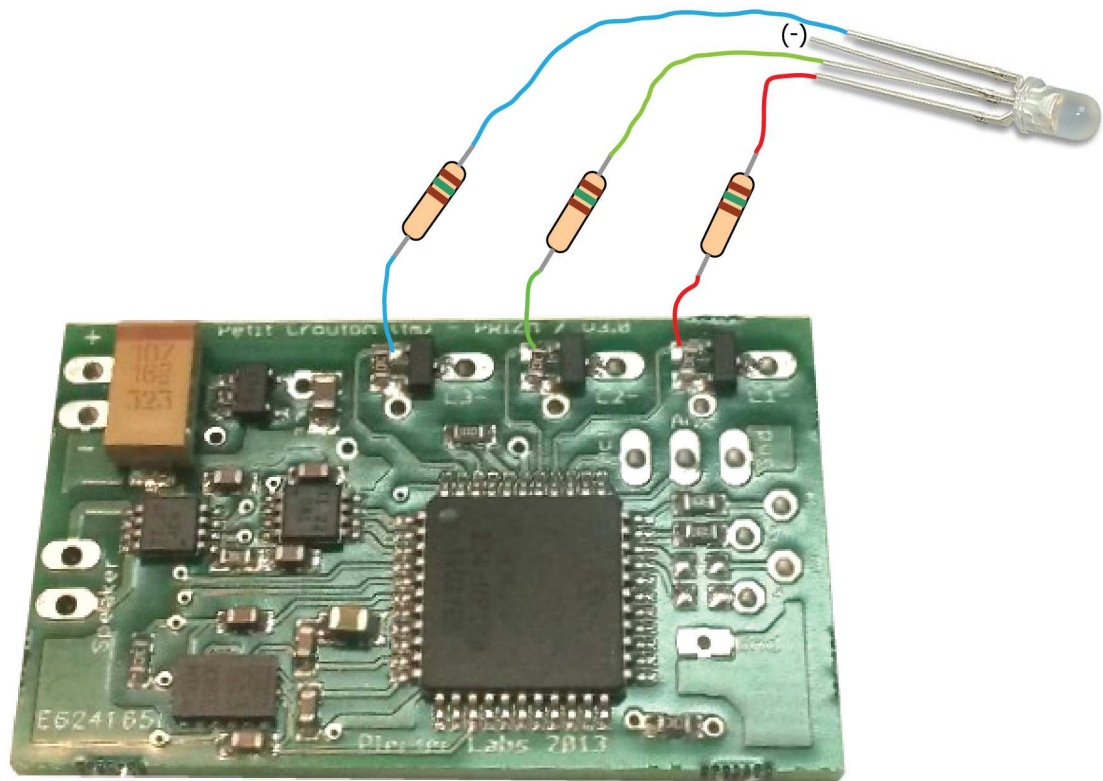
Add a crystal chamber to your saber

A crystal chamber is a nice detail to add to your saber. The crystal can be lit up by a multicolor LED which can mirror the blade behavior, for more realism. This way, the crystal chamber color is affected by the Flash on Clash color, or simply by the main blade flicker.

Whether you use or not the color extender board, FlexiBlend driving signals (PWM) can be extracted to power the crystal chamber LED. Ideally, you'll need to use an LED that matches your high-power LED setup. For instance, if you are using a RGB tri-rebel, the best choice is obviously a RGB accent LEDs. You will need a common cathode type of LED, those are fairly common on ebay and in many electronic parts outlets. Like the other accent LED pad, the maximum available current per PWM signal is 18 mA. Most LEDs are bright and fully happy with 5 to 10 mA.

The on-board driver PWM signal doesn't have a pad per say but can be easily grab right next to each direct drive channel. Example below provided with color channels #1, 2, 3 wired as R, G, B. The common cathode (-) of the accent LED can return to the main negative of the board, or the accent LED ground pad

An alternative solution is to wire each accent die of the crystal chamber LED (and resistor) in parallel with the matching high-power LED die. In this case, the accent LED has to be common anode like the main LED.



Flash on Clash™ Mixing Techniques

During the development of CFv6 and our Flexiblend color mixing engine, the Flash on Clash effect was modified to fit specific color mixing requirements, so that the Flash on Clash color (defined by fled1, 2 and 3). However, customers liked a lot the FoC effect developed for CFV5 and further referred as "on-top" FoC as it is defined by adding a secondary color or die (dice) on top of the blade main color. So, technically, from the Flexiblend color mixing engine point of view, it is not a color change but rather the addition of another light source (which defines pretty well what the Flash on Clash is).

So due to popular demand, I designed a Flash on Clash engine that is compatible with both FoC styles by selecting, for each led die, which one will be color mixed during the Flash on Clash, and which one(s) will be added "on top".

The FoC configuration uses a new parameter **focmix** in the configuration file. The parameter is defined as a bit field, in a similar way than the accent LED sequence states. The left most digit (0 or 1) is color channel #3 (or formerly FoC2) while the rightmost digit is color channel #1 (onboard driver).

A zero indicates the concerned die isn't mixed (therefore "on-top") while a one means that the die will be color mixed.

As an example, the default focmix value is:

focmix=011

which means that color channels #1 and #2 will be color mixed while color channel #3 will be added on top.

The di(c)e added on top have their own timeline and behavior using the parameters **focd** (duration), **focp** (period), **focr** (randomness) and **foc%** (strength) while the mixed di(c)e will be handled by the color mixing engine and shimmer parameters & timeline.

That very flexible FoC mixing system allows a lot of scenarios to be achieved. Here are a few examples.

- A) Using a GGW tri-rebel or tri-cree. Both green dice are wired in parallel. The main blade color isn't affected per say during Flash on Clash, the white die is added on top of the 2 green ones during the FoC effect.

The green dice are configured for a drive of 900 channel #1, white dice is on color channel #2, leading to the following configuration:

```
color= 900,0,0  
fcolor = 900,900,0  
focmix=001
```

- B) Using a RGB tri-rebel or tri-cree. The 3 dice are mixed. One color is defined as the main blade color, while the FoC has another color. That configuration will use FoC mixing on all dice. Let's say the led is wired as RGB. Main blade color is purple (red + blue) and FoC color is whitish. FoC parameters aren't used, only

the shimmer parameters affect the blade behavior during the Flash on Clash effect. That special "all mixed" mode will swap back and forth between the main blade and FoC blade colors ensuring that the FoC color doesn't wipe out the main blade color

```
color=300,0,750  
fcolor = 500,500,500  
focmix=111
```

- C) Using a RGW tri-rebel or tri-cree. FoC color is an alteration of the blade main color (ie : the blade color changes) but the third die is also added on top. Let's take this time the example of a blue blade that would become cyanish during FoC plus the white die on top, in order to both affect the color and overall increase the blade brightness. The first 2 color channels remain mixed during the Flash on Clash effect, for the whole duration of the shimmer, while the white die is applied on top, during its own specific duration (**focd**).

```
color=1000,200,0  
fcolor=1000,500,700  
focmix=011
```

Whether you plan just a color change or see the FoC as another die lighting up, the user must consider the amount of current available from the battery pack. A color change might actually consume a lot more current than the "normal blade" (like moving from a red blade to a white FoC). To match the battery pack capabilities, the overall current of the different die could need to be lowered to obtain the right color but with less brightness.

Accent LEDs sequencer

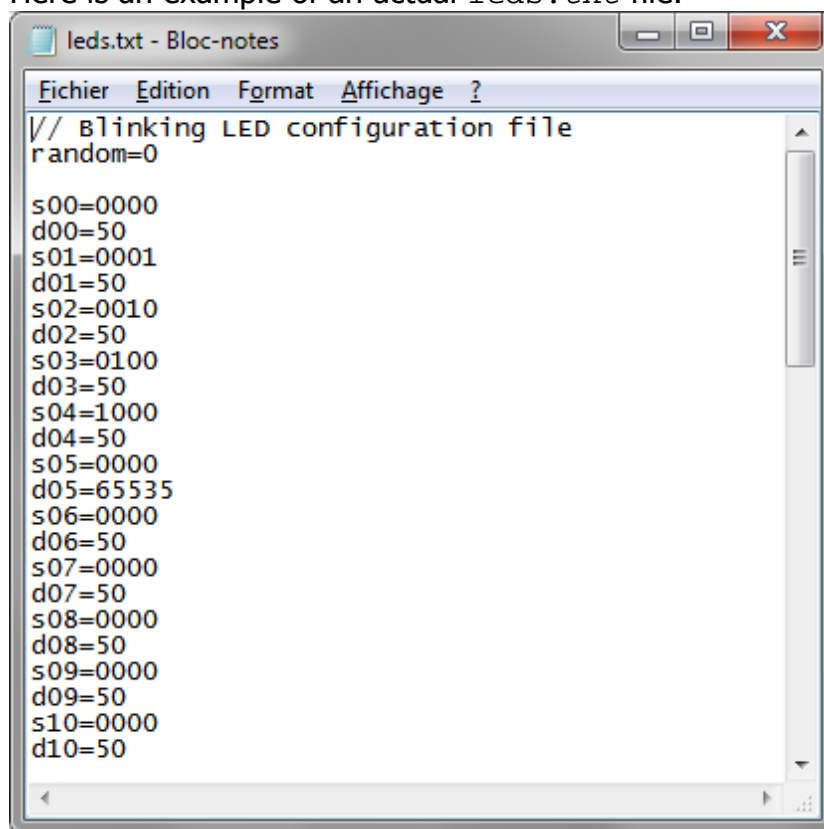
Four LEDs can produce 16 different patterns; for the sake of editing the sequence we limited the sequencer to 32 stages.

The working principle of the sequencer is to put the LEDs in a given pattern/state during a certain time (delay). Once the delay is over, a new pattern is output and so on for the 32 stages before looping to the first one.

The main LED sequence is described for each sound bank in the `leds.txt` text file. Simply open the file with window notepad to modify it, directly on the SD card or on the hard disk.

PC has one main accent LED configuration file `leds.txt`.

Here is an example of an actual `leds.txt` file.



```
// Blinking LED configuration file
random=0

s00=0000
d00=50
s01=0001
d01=50
s02=0010
d02=50
s03=0100
d03=50
s04=1000
d04=50
s05=0000
d05=65535
s06=0000
d06=50
s07=0000
d07=50
s08=0000
d08=50
s09=0000
d09=50
s10=0000
d10=50
```

Stages & Delays

The example below shows the accent LEDs with a chasing pattern. The state of each LED is represented either by 0 (off) or 1 (on). Each "s" line defines a pattern of the 7 LEDs (state) while the "d" line is the duration of the pattern (delay).

The closest digit (0 or 1) to the equal sign is the state of LED4 (MSB) while the one at the end of the line is the state of LED1 (LSB)

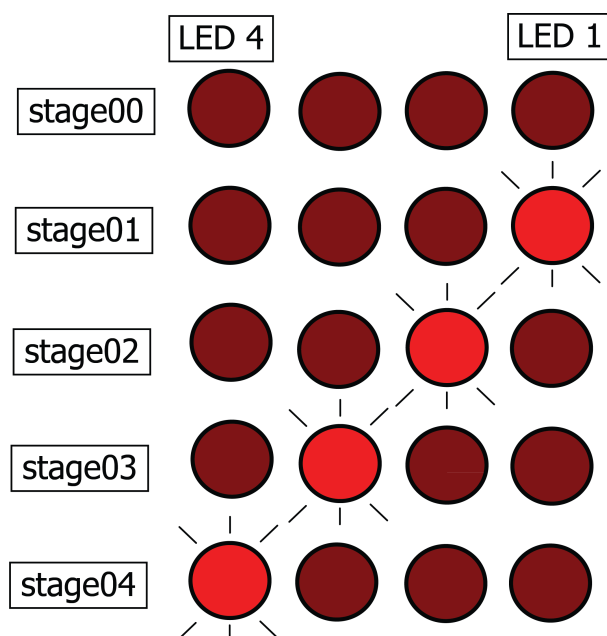

```

leds.txt - Bloc-notes
Fichier Edition Format Affichage ?
// Blinking LED configuration file
random=0

s00=0000
d00=50
s01=0001
d01=50
s02=0010
d02=50
s03=0100
d03=50
s04=1000
d04=50
s05=0000
d05=65535

```

The figure below represents the 5 first lines of the sequence of the example text file (note the "loop" instruction using the max value of the delay on line #5)



The delay between each state can be different for each "state" line, it's not necessary to have a pattern evolution with a constant timing. A value of 0 for a "delay" line will stop the sequencer. Hence, it's possible to have an accent LED animation when the saber starts, and the animation stop at the end of the 32 stages (or before) when a 0 delay is found in the file.

The Petit Crouton accent LED sequencer also features a **loop** instruction with a delay of 65535. Using that delay in the sequence will make it loop before reaching the end of the sequence (stage31) which makes writing short sequences easier by removing the need of any filling or padding with non-changing stages.

When setting **random** to 1 the sequencer will generate a random pattern at each stage of the sequence, ideal for a changing or non conventional animation. The delays

will remain the ones specified in the sequence. Like all the timing parameters used in Petit Crouton, **the delay is expressed in multiples of 2ms**

The delay can go up to 60,000 which corresponds roughly to 2 minutes. It's of course possible to use a LED output to drive an accent LED **AND** the remote control wire of a power xtender board to drive an additional peripheral requiring more power than what the accent LED pads can offer.

Deep Sleep flashing LED

It's also possible to use one of the 4 accent LEDs to indicate the saber is in deep sleep mode (power saving). For that purpose, 2 parameters have been added to the configuration file `config.txt`.

- **idleled** [0-3]: selection of the LED used as idle indicator. **0** matches LED number one, 3 indicates accent LED pad four. Since there is a specific `config.txt` file in each sound bank folder, the idle LED can be different in each sound font and why not matching the bank number. Hence, just by looking to the flashing LED the user can remember which bank is currently selected in the saber.
- **idlepulsing** [0-40]: duration of the the idle flash of the selected LED. In multiple of 1/40th of second. **0** disable the effect. With **1** we get a really short flash. With **28-30**, the idle LED stays on all the time except during a brief moment, and with **40**, the idle LED remains always on.

Force push" effect

One of the great improvements since version 2 is the "Force Push" sound effect. In our favorite movies, the characters have the ability to push an opponent using telekinesis. This effect is by definition silent, however, in the video games for instance, an audio effect is added to provide the appropriate feedback to the player or to the audience.

The "Force Push" gesture is (usually) done by the hand which is not holding the saber, it's therefore impossible to detect the gesture with it (without additional sensors). We implemented a "trick" to control the "Force Push" using our gesture detection system and the auxiliary button.

The main idea is to use the hand holding the saber as the "real" trigger of the effect while the other hand fakes to do the gesture.

The fighter will "push" his opponent with one hand while moving a bit the saber/hilt and pressing the aux. button at the same time.

The aim is to trigger a swing gesture **while pressing the aux. button at the same time**: it will play the corresponding "force" sound effect (`force.wav`).

This combo is not so easy to achieve especially if the fighter is strongly left or right handed. It might be easier for an ambidextrous person, or someone playing a musical instrument. But just like dueling with a saber, it just requires practice and training to master an expert gesture.

To get used to the Force Push technique, increase in a significant way the value of `lockup` to stay away from an easy blade lockup triggering. Press the aux. button first then wave the hilt to trigger a Force Push effect. Once you'll get used to this combo (aux. button then quick waving), reduce again the parameter `lockup` until you obtain a good access to the 3 effects activated by the aux. button (Blaster Deflection, Blade Lockup, Force Push)

Once the effect has been played, if the auxiliary button is kept pressed, the blade lockup, blaster and further clash and swing effects are disabled until the button is released.

Mute On The Go™

Proposed and requested by Novastar for the specific situation of teaching saber classes and interacting with students. We all enjoy our saber to be loud when playing with it, however, it's hard to show saber moves and interact with the students with the saber sound in the background.

If was already possible to mute the saber in the previous versions by using a dedicated bank in which **mute** was enabled, however it was not very convenient to go back and forth between banks just to show a few saber moves in a quiet way.

We therefore tweaked the firmware so that, pressing the aux. switch prior pressing the activator switch will mute the saber until powering the blade off (the power-off sound will be played). Further "regular" ignition (without pressing the aux. switch prior ignition) will have the normal sound volume as setup in the configuration file of the selected sound bank.

User's Notes

Using R.I.C.E.(Real-time Internal Configuration Editor)

R.I.C.E. is a computer program allowing the user to remotely control, change and save the configuration of the current sound bank without disassembling the saber, using a data cable like the one sold at [TCSS](#). R.I.C.E. is now available on PC (windows XP, Vista and Seven) and on MacOS.



R.I.C.E. is very easy to use. The program is available for MacOS and Windows computers. With Windows 7, the TCSS cable will be recognized automatically without the need of installing drivers.

The data cable is a USB serial port. You can source your own, there are mostly 3 different chipsets on the market, FTDI, the Prolific PL2303 and the Silicon Labs (Silabs) CP2102.

The R.I.C.E. data connection to the sound board just requires 3 wires : Rx, Tx and the ground. Make sure you use a RICE cable or USB serial port that is 3.3V compliant.

Getting started with R.I.C.E.

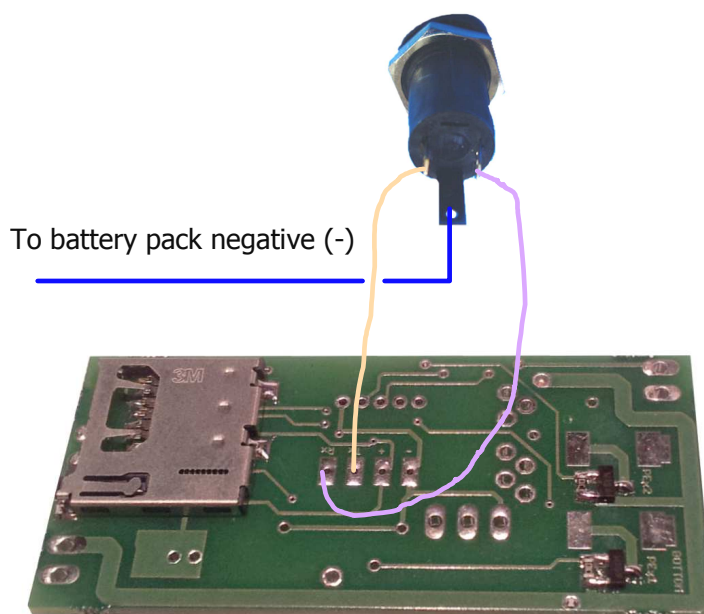
The R.I.C.E. pads are under the sound board and are labeled. To connect the data cable we recommend using a 3.5 mm stereo female jack with the following wiring:

Ring=TXD

Tip=RXD

Sleeve=GND

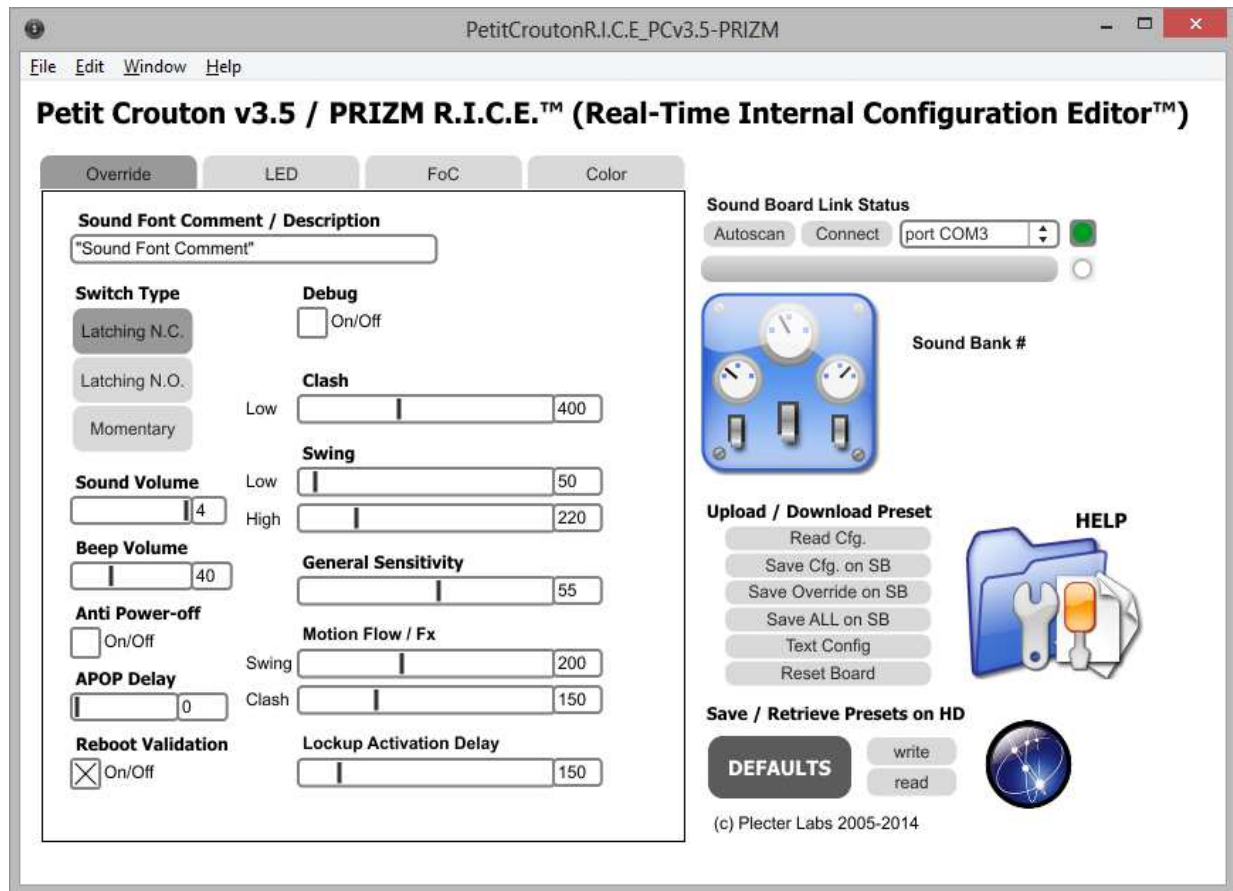
While it's possible to wire the sleeve (GND) wire to the GND pad of the board, we recommend to connect it **to the battery pack negative** and not to the board negative to avoid a kill key / recharge port bypass in the case a metal female jack is used. Prefer a female jack with a plastic housing.



Note: When not using the saber, especially when the kill key is inserted in the recharge port, disconnect the R.I.C.E. cable from the data port.

If you got a USB serial port adapter from another source, the pinout might differ. If you cannot get the RICE software to talk to the board, **try swapping Rx and Tx.**

To configure the board in real time with R.I.C.E. , just unzip the software archive in a directory of the hard drive and double click on the program icon. You'll get straight in the R.I.C.E. main window.



The numerous parameters of the main PC configuration file (`config.txt`) have been split in 4 tabs, organized by "themes" as much as possible to ease the configuration process. If you need any **help** to remember how R.I.C.E. works, click on the *Folder+Tools* icon on the lower right side of the program, it will open the help section.

To establish the communication with the saber, turn the blade on, then select the serial port. Make sure to always click the serial port in the drop-down list even if the correct one appears as selected.



It's advised to use a single serial port at a time. Sometimes you'll have your Bluetooth serial port enabled at the same time and appearing in the port selection menu: make sure you identify the serial / COM port # corresponding to your R.I.C.E. cable. If you have a doubt, go to the System control panel and open the Peripherals list. Plugging / unplugging your cable will show up in the list. You can change the port # by right clicking on the COM peripheral and going in the advanced properties. I tend to renumber all my data cable as COM2 but I use only one at once.

If you plugged your cable after launching the RICE program and your port doesn't appear in the drop-down PORT menu, click the **autoscan** button first, it will update the list of available ports.

Note : if by mistake you unplug your cable while the R.I.C.E. program is open, click "autoconnect" a couple of times to ensure the COM port list is updated again once it's plugged back into the USB port of the computer..

With the saber on and the COM port selected, click connect. The saber should answer echo and the green light will turn on.

From there, your saber is ready to receive real-time configuration changes.

Reading the current settings

Just click on the "Read Cfg" button, and all the parameters of the different tabs will refresh with the current settings of the saber (for the currently loaded sound bank which number appears on the right side of the app).

Changing settings

The key point with R.I.C.E. is that the data cable doesn't need to stay connected all the time to the saber once you've altered a parameter and you want to test it in an actual condition. For example, the motion sensitivity parameters have to be tested by waving the hilt. It's not a problem at all : R.I.C.E. supports to have data cable "hot unplugged" and the blade led can even be turned off and turned on again (for testing parameters like quick on and quick off durations. Then finally, the cable can be "hot plugged" again to proceed to the saving operation.

Discarding settings

Not sure of what you've done? You want to start over? Discarding settings can be done in 2 ways:

- Insert the kill key in the recharge port to cut the power of the saber. Non saved changes made to the configuration will be lost / discarded.
- Click on the default button in the app to have a default configuration applied to the saber. Beware, if your switch setup doesn't match the default, you might have to play with the switch to maintain the saber on until you have actually changed the switch setting.

Saving the settings

Once you're satisfied with the configuration of the saber, you can click on the "Save on SB" button. The saber will play the power off sound followed by a 3 note melody and will reboot with the settings saved. As settings are now split between the `override.txt` file and the `bank config.txt` file, the user can select which part of the settings to save, or both. Remember that the color profiles are stored in the `override.txt` file.

Using R.I.C.E. as a debug tool

Your saber has issues, you don't know where it's stuck or why. Manually edit the configuration file of the (expected) sound bank that is accessed when you boot the saber add `debug=1` on the first line of the file.

The board will be a lot more verbose during the boot process and will send out some useful debugging information.

To see the boot process log, press Ctrl-M (or Apple-M on MacOS) to open the Max Window of the R.I.C.E. program. Before powering the saber (ie removing the kill key) make sure the data cable is inserted in the saber data port and in the USB port on the computer side, then select the right COM port in R.I.C.E.

Finally, remove the kill key to power the saber and look at the log of information printed in the Max window.

Color Setup & Color Mixing

The screenshot shows the R.I.C.E. software interface with the 'Color' tab selected. The interface is divided into four main sections: 'Wiring', 'PRIZM', 'Main Blade Color', and 'FoC Color'. Each section has a set of sliders and a color picker.

- Wiring:** Three buttons labeled RGB, RBG, and GRB. The 'BRG' button is highlighted.
- PRIZM:** Three sliders labeled '0', '10', and '1'. The '1' slider is highlighted.
- Main Blade Color:** Three sliders labeled 'Drive1', 'Drive2', and 'Drive3'. The 'Drive1' slider is set to 700, while 'Drive2' and 'Drive3' are set to 0. A color picker is shown to the right.
- FoC Color:** Three sliders labeled 'Drive1', 'Drive2', and 'Drive3'. The 'Drive1' slider is set to 700, while 'Drive2' and 'Drive3' are set to 0. A color picker is shown to the right.

A few special features have been incorporated to R.I.C.E. to ease the color selection process.

Rough color setup

With Prizm, the blade (and FoC) color are defined in profiles. We therefore set a single color setting tab with the selection of the color profile being edited.

While it is possible to change on the fly the amount of profiles used on the saber, the best practice is to first set as many profiles as you want to use manually in the configuration file, with a default drive on channel 1, like the following:

```
color0=800,0,0  
fcolor0=800,400,0
```

From there, you can edit those in RICE by selecting the color with the swatch.

The swatch of course doesn't know how your LEDs are wired (multi-led or multi-die). so we setup a color channel swapping system so the color selection with the swatch affects the right current channels 1, 2 and 3. Obviously, as the swatch deals with RGB, only RGB combinations are present there.

Color fine tuning

Once you're happy with the shade, you can still adjust individually each drive to make the color more saturated, or brighter.

If you're having a channel on which you discarded the resistor, make sure you don't exceed the drive you previously found out as the limit.

User's Notes

Troubleshooting & FAQ

Q : I've updated the sounds on the SD card and now the module does not work anymore. It generates some beeps when it starts.

A : you must format the SD card (in FAT16 / FAT32) before updating the sounds, while it's not necessary to do so for the configuration files that can be edited in place on the SD card. To simplify the update process, put the new sounds on the SD, overwrite the old ones. Then select the whole contents of the SD card, and copy it in a temporary directory/folder on the hard disk. Format then the card, and next copy the files back to the SD card in one run.

Q : what's the meaning of the beeps when the module starts.

A : it means that a file is missing or hasn't been found. It can be the boot sound, or the configuration files. Three beeps generally mean that many important files haven't been found. You can get more details about what's happening by hooking up your RICE cable, connect, and press ctrl+M (apple key + M on mac) to display the debug window and see the boot log of your board. Add debug=1 at the beginning of your configuration file for additional information.

Q : when I power up the saber, I get 3 beeps. I haven't updated the card.

A : You might have a corrupted preference file (`prefs.txt`). Get the default one from the default package.

Q : can I rename the sub-folders on the SD card ?

A : no, the module expects a certain organization of the file system on the card so that it can find the sounds.

Q : I don't have the aux. Button wired, can I rename the sub-folders to « swap » them ?

A : no, because it will modify the SD card file structure. Wire an auxiliary button to access the different sound banks.

Q : I wired a rumbling motor in my hilt and now I have swings triggered while the hilt rests on a flat surface.

A : The motor makes the board vibrating enough so that it's interpreted as a swing. Reduce the speed of the motor and/or mechanically isolate the board from the motor. Increasing **ls** can also help, but it will make swings more difficult to execute.

Plecter Labs would like to thank his affiliates for their useful input on the Petit Crouton Board specs, new features ideas, constructive criticism as well as proof reading of that user's manual.

I also would like to deeply thank our users, customers and the illuminated saber hobby community for the trust and support input in my work along the years, pushing the design of our favorite props toward perpetual improvement.

"Custom Electronics for Props that WORK!"

Non exhaustive list of trademarks owned by Plecter Labs

Blaster Move™
Wake on Move™
Power on Move™
Power on Force™
FoC™ / Flash On Clash™
Vocal Menu™
Mute on the go™
Anti Power On/Off Protection - APOP™
R.I.C.E. - Real time Internal Configuration Editor™
Crystal Pulse™
Crystal Focus™ and Crystal Focus Saber Core™
Petit Crouton™
Nano Biscotte™
Power On Angular Selection™
SD config™
Force Clash™
Force Swing™
Buttered Toast™
S.S.B.T.™
Secret Society of the Buttered Toast™
Power Extender™
Power Xtender™
Saber Audio Player™ (SAP™) aka iSaber.
AccuBolt™
FlexiBlend™

The use of Plecter Labs trademarked terms is prohibited for use in advertisement or sales of a product not made by Plecter Labs or for a product not containing a Plecter Labs electronic device.