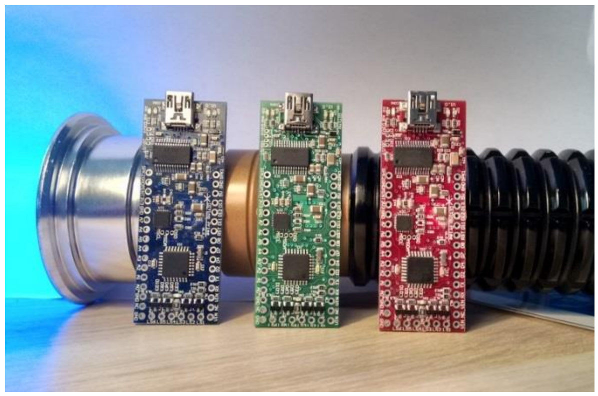
DIYino Prime v1.0 User Manual

Table of Contents

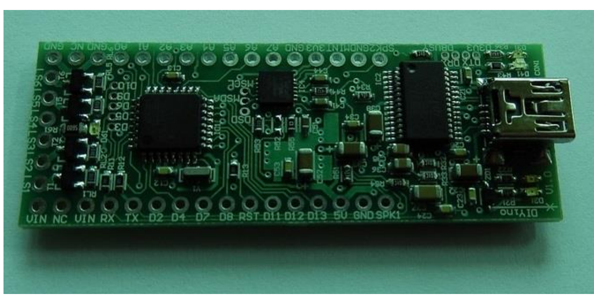
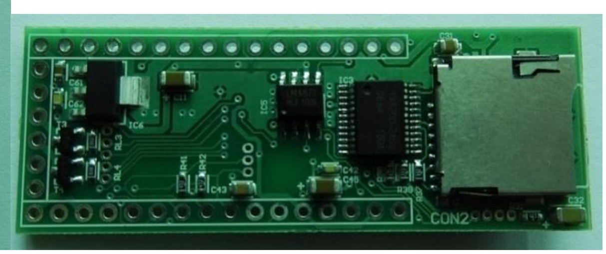
DIYino Prime V1.0

1 Introduction

DIYino Prime is an Arduino compatible integrated circuit board for all projects implementing Light/Sound/Motion. Its main field of application is to control lightsaber and other prop/replica electronics. It is Arduino compatible, i.e. it can be programmed using Arduino compatible IDE's (Arduino IDE, Eclipse etc.)



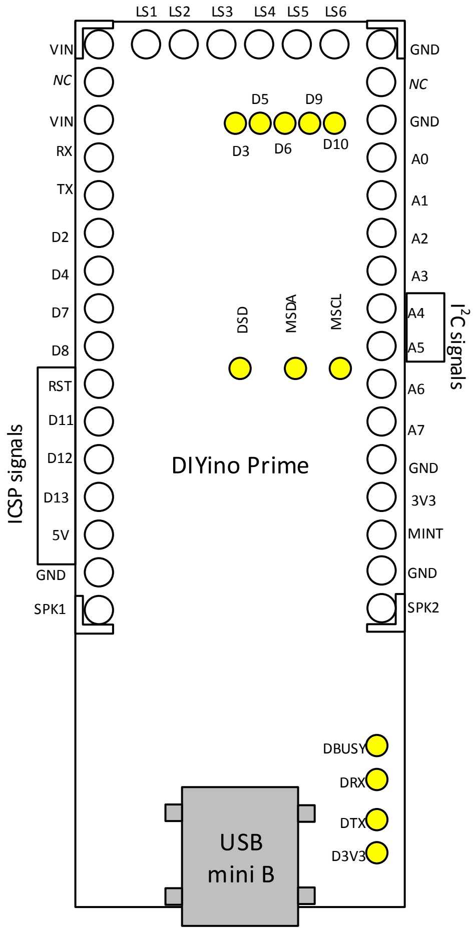
2 General Product Characteristics

2.1 Circuit overview (size, short feature overview)

- Very compact size, dimensions: **22** mm(W)x **60** mm(L)x **8** mm(H)
- Arduino Compatible
- Gapless Wav-audio playback
- 3W audio amplifier (supports 8Ω/4Ω speakers up to 3W)
- Built in high-end gesture detection sensor (6-axis accelerometer and gyro)
- Includes FTDI USB2Serial chipset for plug-and-play programming using Arduino IDE or similar
- 6 PWM controlled build-in low-side drivers as power extenders with up to 1.1A current capability each
- Input voltage range 3.3V-5V or 6V-20V

2.2 Circuit Pinout



PCB Top View

Pin	Functionality	Comment
VIN	Voltage-in pin. Connect to Battery(+)/power source(+). Input to 5V LDO.	Input voltage must stay between 6V and 20V for correct operation.
NC	Non-connected pin. Can be used on the PCB for signal routing.	
VIN	Voltage-in pin. Connect to Battery(+)/power source(+). Input to 5V LDO.	Input voltage must stay between 6V and 20V for correct operation.
RX	D0 digital I/0 of Atmega328P. RX pin of Serial UART.	

TX	D1 digital I/0 of Atmega328P. TX pin of Serial UART.	
D2	D2 digital I/O of Atmega328P. INT capable pin.	
D4	D4 digital I/O of Atmega328P.	
D7	D7 digital I/O of Atmega328P.	Connected to YX5200-24SS TX pin. Used for communication between Atmega328P and MP3/wav-decoder.
D8	D8 digital I/O of Atmega328P.	Connected to YX5200-24SS RX pin. Used for communication between Atmega328P and MP3/wav-decoder.
RST	Reset pin of the Atmega328P, with pull-up to 5V.	Pin used for burning bootloader.
D11	D11 digital I/O of Atmega328P, PWM capable, connected to Gate of LS6 Low-Side driver, with 100kΩ pull-down to GND. MOSI pin for ICP.	Pin used for burning bootloader.
D12	D12 digital I/O of Atmega328P. MISO pin for ICP.	Pin used for burning bootloader.
D13	D13 digital I/O of Atmega328P. SCK pin for ICP.	Pin used for burning bootloader.
5V	Output of the 5V LDO (type 7805). Supply pin for Atmega328P, FTDI, YX5200-24SS chipsets.	Pin used for burning bootloader.
GND	Board(-) or GND. Connected to GND plane of the PCB.	Pin used for burning bootloader.
SPK1	Speaker terminal 1. Connect to speaker directly. Other speaker	

	terminal to SPK2.	
GND	Board(-) or GND. Connected to GND plane of the PCB.	
NC	Non-connected pin. Can be used on the PCB for signal routing.	
GND	Board(-) or GND. Connected to GND plane of the PCB.	
A0	A0 digital I/O of Atmega328P with input 10-bit ADC.	
A1	A1 digital I/O of Atmega328P with input 10-bit ADC.	
A2	A2 digital I/O of Atmega328P with input 10-bit ADC.	
A3	A3 digital I/O of Atmega328P with input 10-bit ADC.	
A4	A4 digital I/O of Atmega328P with input 10-bit ADC.	SDA I2C signal for communication with MPU6050. Connected to auxiliary signal MSDA.
A5	A5 digital I/O of Atmega328P with input 10-bit ADC.	SCL I2C signal for communication with MPU6050. Connected to auxiliary signal MSCL.
A6	A6 analog input pin with 10-bit ADC.	Only analog input, cannot be used as digital I/O.
A7	A7 analog input pin with 10-bit ADC.	Only analog input, cannot be used as digital I/O.
GND	Board(-) or GND. Connected to GND plane of the PCB.	
3V3	Output of the 3.3V LDO of the FTDI (FT232RL) chip.	Current capability up to 50mA. 100nF decoupling cap integrated

		on board.
MINT	INT output pin of the MPU6050.	
GND	Board(-) or GND. Connected to GND plane of the PCB.	
SPK2	Speaker terminal 2. Connect to speaker directly. Other speaker terminal to SPK1.	
LS1	Drain of the Low-Side switch 1	
LS2	Drain of the Low-Side switch 2	
LS3	Drain of the Low-Side switch 3	
LS4	Drain of the Low-Side switch 4	
LS5	Drain of the Low-Side switch 5	
LS6	Drain of the Low-Side switch 6	
D3	D3 digital I/O of Atmega328P, PWM capable, INT capable, connected to Gate of LS1 Low-Side driver, with 100kΩ pull-down to GND	Auxiliary signals, see Chapter 4.4
D5	D5 digital I/O of Atmega328P, PWM capable, connected to Gate of LS2 Low-Side driver, with 100kΩ pull-down to GND	Auxiliary signals, see Chapter 4.4
D6	D6 digital I/O of Atmega328P, PWM capable, connected to Gate of LS3 Low-Side driver, with 100kΩ pull-down to GND	Auxiliary signals, see Chapter 4.4
D9	D9 digital I/O of Atmega328P, PWM capable, connected to Gate of LS4 Low-Side driver, with 100kΩ pull-down to GND	Auxiliary signals, see Chapter 4.4

D10	D10 digital I/O of Atmega328P, PWM capable, connected to Gate of LS5 Low-Side driver, with 100kΩ pull-down to GND	Auxiliary signals, see Chapter 4.4
SDS	Shut-down pin of the LM4871 3W audio-amplifier.	Auxiliary signals, see Chapter 4.4
MSDA	SDA I2C signal of MPU6050	Auxiliary signals, see Chapter 4.4
MSCL	SCL I2C signal of MPU6050	Auxiliary signals, see Chapter 4.4
DBUSY	Busy signal of the YX5200-24SS wav decoder.	Auxiliary signals, see Chapter 4.4
DRX	RX line of the YX5200-24SS wav decoder, connected to D8.	Auxiliary signals, see Chapter 4.4
DTX	RX line of the YX5200-24SS wav decoder, connected to D7.	Auxiliary signals, see Chapter 4.4
D3V3	Output of the 3.3V LDO of the YX5200-24SS chip.	Auxiliary signals, see Chapter 4.4

2.3 Module description

2.3.1 Low-side drivers

In order to control High-Power LEDs or LED strings consisting or multiple LEDs, the DIYino Prime board implements so called Low-Side drivers to connect the negative side of loads (i.e. cathode of LEDs) to the GND. A Low-Side driver consists of an n-channel type MOS transistor with its source connected to GND of the board, the drain is connected to an LS terminals (LS1 to LS6), and the gate is controlled via PWM by PWM capable pins of the Atmega328 uController.

Therefore code-wise in order to adjust the drive from the Low-Side drivers the PWM capable pins have to be addressed; the mapping is shown in the next table:

LS pin on DIYino Prime	Corresponding PWM capable pin
LS1	D3
LS2	D5
LS3	D6
LS4	D9
LS5	D10
LS6	D11

TABLE 1: MAPPING OF PWM CAPABLE ATMEGA328 PINS TO THE LOW-SIDE DRIVER PINS

Wiring of loads via the LS pins is depicted in Figure 1, using as example a RGB(W) HP-LED setup., but the concept is the same using LED strings or serially connected LEDs in general.

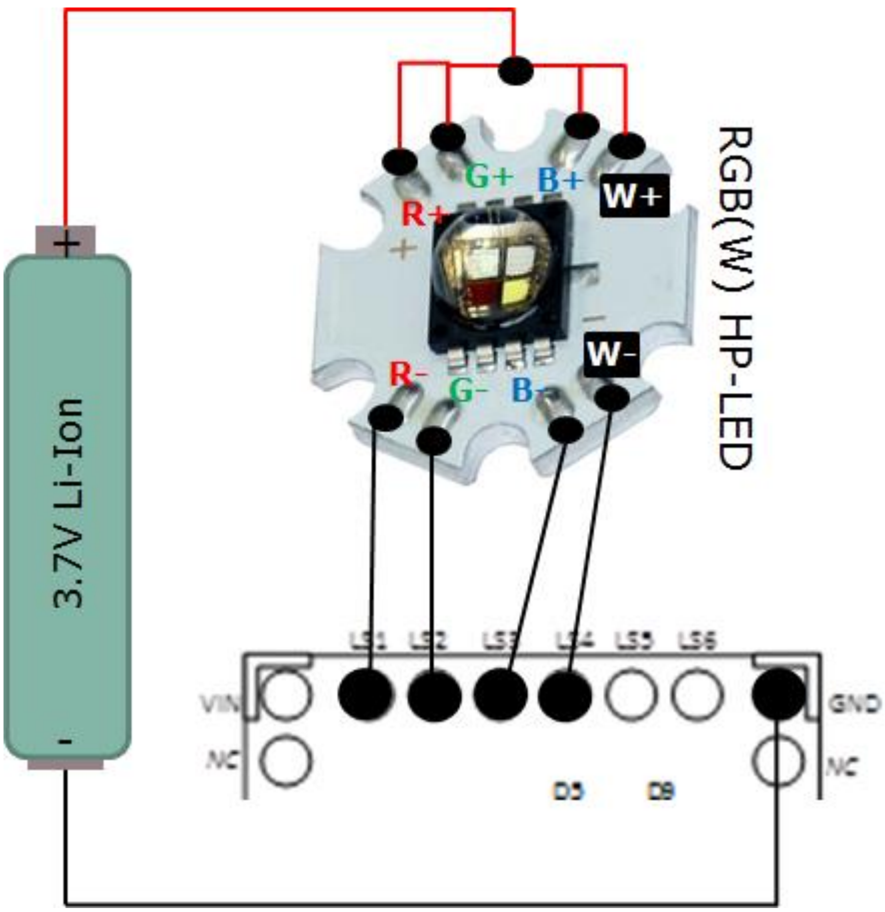


FIGURE 1: CONNECTION OF THE LOAD TO THE LOW-SIDE DRIVER PINS (LS1…LS6). EXAMPLE SHOWS WIRING OF AN RGB(W) HIGH-POWER LED MODULE ON STAR PCB. CURRENT LIMITING RESISTORS ARE NOT SHOWN FOR SIMPLICITY'S SAKE.

2.3.2 GESTURE- AND MOTION DETECTION WITH THE MPU6050 IMU

Link to Datasheet: https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
Link to Register Map Description (for low level programming): https://www.invensense.com/wp-content/uploads/2015/02/MPU-6500-Register-Map2.pdf
Link to MPU6050 Library for Arduino:
https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050
Link to MPU6050 Library for LSOS:
https://github.com/neskweek/LightSaberOS/tree/master/Libraries/MPU6050
Link to MPU6050 USaber MotionManager:
https://github.com/JakeS0ft/USaber/blob/master/motion/Mpu6050MotionManager.h

2.3.3 WAV/MP3 DECODE CHIPSET AND AUDIO AMP

The DIYino Prime v1 implements the chipset of the MP3-TF-16P Arduino Shield/MP3 module, which includes an YX5200-24SS MP3/Wav decoder chip, a micro-SD card slot and a 3W audio amp.
Link to Datasheet of the sound module:
http://www.dfrobot.com/image/data/DFR0299/DFPlayer%20Mini%20Manul.pdf
Link to the 3W Audio Amp Datasheet: http://www.ti.com/lit/ds/symlink/lm4871.pdf
Link to simple Library: https://github.com/DFRobot/DFPlayer-Mini-mp3
Link to neskweek's improved Library:
https://github.com/neskweek/LightSaberOS/tree/master/Libraries/DFPlayer

Warning: DIYino Prime is an electronic board containing parts sensitive to ESD. Final wiring & assembly is under the 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
The manufacturer cannot be held responsible for improper use or assembly of the DIYino Prime board.

3 BOARD SUPPLY CONCEPTS

Warning: Please note that the DIYino Prime board does not implement a reverse polarity protection. Reversing the polarity of the supply might lead to board damage!

3.1 STANDARD SUPPLY SCHEME

The DIYino Prime board shall be supplied from the VIN pin (similar to the Arduino Nano on which the DIYino design is based on). Due to the fact that the DIYino Prime board implements an 5V LDO between VIN and 5V pins, the voltage of the VIN pin has to be between 6V(min) and 20V(max).
Figure 2 shows wiring of the voltage source used to supply the board. This wiring scheme can be used if the voltage source has an output voltage between 6V(min) and 20V(max). The integrated LDO will regulate the input voltage on VIN to 5V and supply the rest of the components on the board. Please note that any VIN pin can be used to connect the voltage source to. The two VIN pins are connected together on the board.
If the USB cable is plugged in, i.e. as during sketch upload, the circuit will auto-select the higher voltage between the one supplied by the USB and the on board 5V LDO.

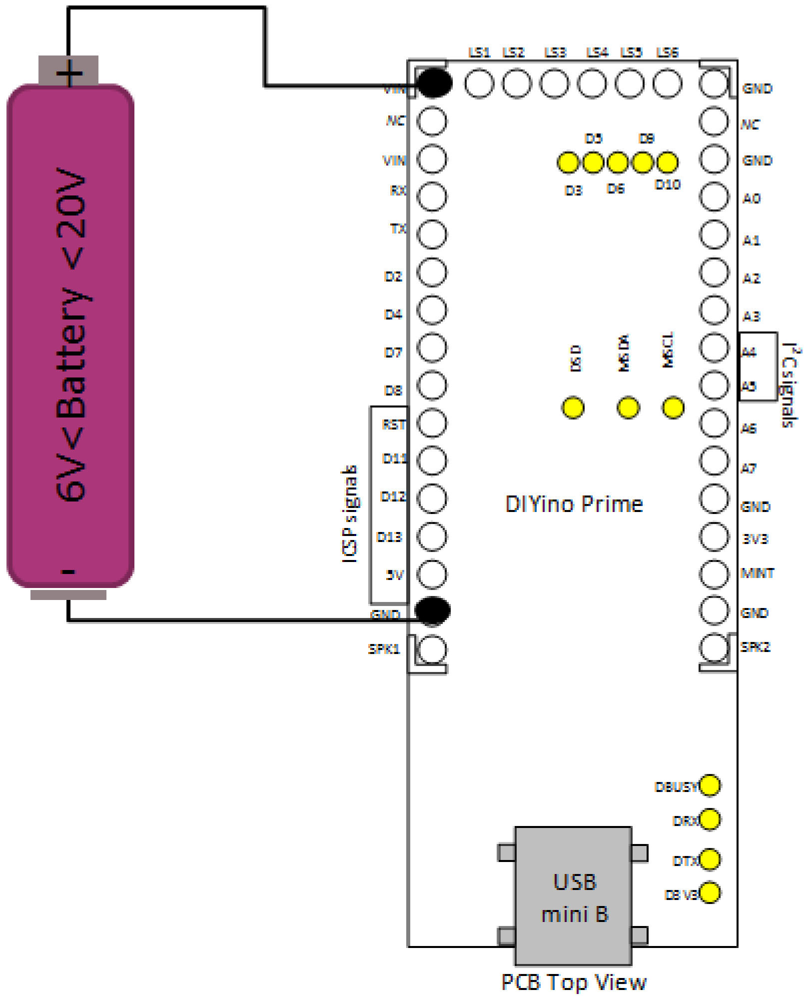


Figure 2: Standard supply wiring

3.2 Alternate supply concepts

The standard supply scheme is explained in Chapter 3.1 The supply schemes introduced in the current Chapter are considered technically feasible to supply the DIYino Prime board, but due to the nominal supply requirements of certain circuit components and the uncertainty arising from static/dynamic behavior of the used external components (for instance DC/DC supply, battery etc.), the manufacturer does not take any liability nor does he guarantee performance of the board if the below described supply schemes are used. What does it mean in simpler terms?

1. In both cases the on-board LDO is bypassed and external components are directly wired to the supply node of sensitive IC’s (5V pin), without the LDO regulating this voltage to 5V. Glitches/transients on the external components might cause irreversible damage to the circuit components.

2. Some of the components are nominally rated for supply voltages close to 5V. Supplying them with a voltage below 5V might degrade their performance.
3. The board is not fully characterized for operation below 5V. Early results indicate the board works down to 3.1V, but it strongly depends on factors like contact resistances, IR-drop, parasitic effects etc.

3.2.1 Sub-5V supply

Due to the

- limited space in the application builds: i.e. prop lightsaber hilts
- the capacity requirements of used batteries: typically >3000mAh, otherwise the saber might shut-off prematurely during a show/presentation
- supply requirements of LEDs: except for serially connected LED-strings, other type of lightsaber light sources require a voltage source between 2V (Vth_typ of red LEDs) and 4V (Vth_max of Blue/Green LEDs).

Li-Ion batteries with a nominal voltage of 3.7V are the most commonly used voltage sources. They can be used to fully supply the DIYino Prime board, the wiring is shown in Figure 3. Instead of connecting the voltage source to VIN, it has to be connected directly to the 5V pin, which is the output of the 5V LDO. All parts directly supplied by the LDO (Atmega328 and FTDI chips) will be supplied now from the external voltage source. VIN can be left unconnected or if necessary it can be connected to the voltage source as well.

Warning: in case this supply scheme is implemented, please ensure to disconnect the power source from the DIYino Prime board before connecting the board via the USB cable to PC. Reason is that the USB in this case will be connected over a diode to the power source, thus acting as a battery charger, which – depending on the battery type and electrical characteristics, such as charging curve – can lead to battery damage.

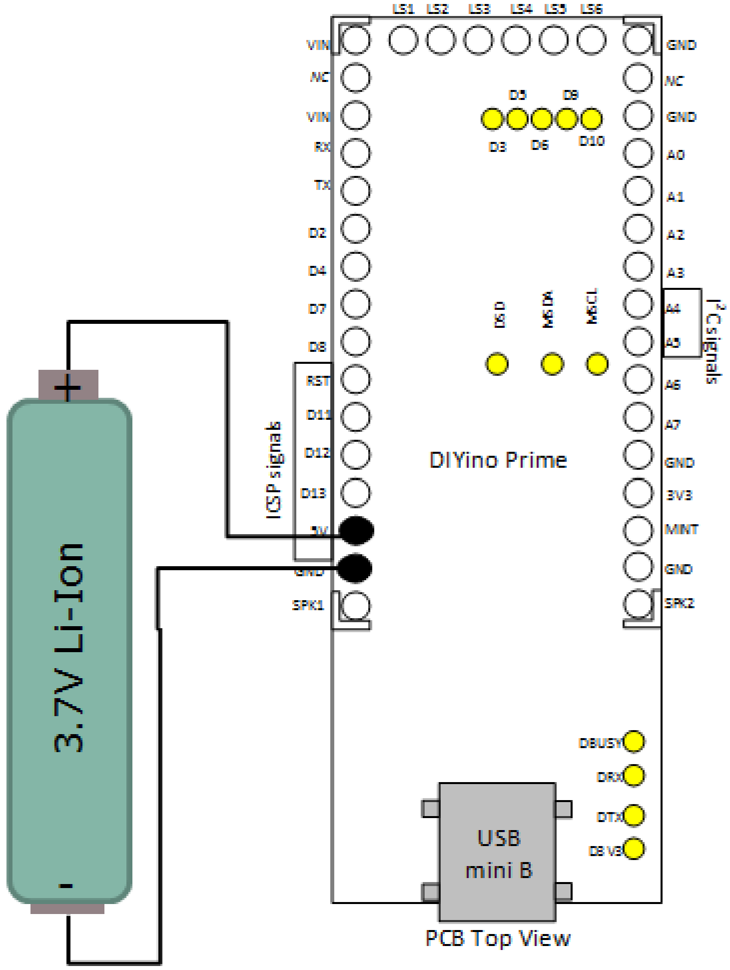


FIGURE 3: SUPPLY OF THE DIYINO BOARD WITH A VOLTAGE SOURCE BELOW 5V

3.2.2 SUPPLY USING 5V DC/DC BOOST

As mentioned at the beginning of Chapter 3.2, supplying the DIYino Prime circuitry with less than 5V on the 5V pin/node might lead to performance and/or premature undervoltage reset condition. For this reason, a hybrid supply scheme is introduced here, which uses a voltage source with a nominal voltage below 5V but still can supply the board circuitry with 5V. This scheme uses a so called DC/DC boost converter. A boost converter regulates ("boosts") the voltage from a lower input voltage to a higher output voltage. Please use a DC/DC boost converter with a 5V output.

Warning: DC/DC boost modules which are not rated at 5V output voltage usually have an adjustable voltage output, which can far exceed 5V. Using such a DC/DC boost module could cause board damage if the output voltage is not adjusted to 5V (or below) prior to connecting it to the board!
Even DC/DC converters with a nominal regulated (boosted) voltage of 5V might exceed 5V, so always measure the output voltage of a DC/DC boost converter before connecting to the DIYino Prime to avoid board damage.

The clear advantage of this supply scheme is that the board is supplied with the required 5V for best performance, while the lower input supply voltage can be used to power the LEDs/other external circuits with nominal voltages below 5V.

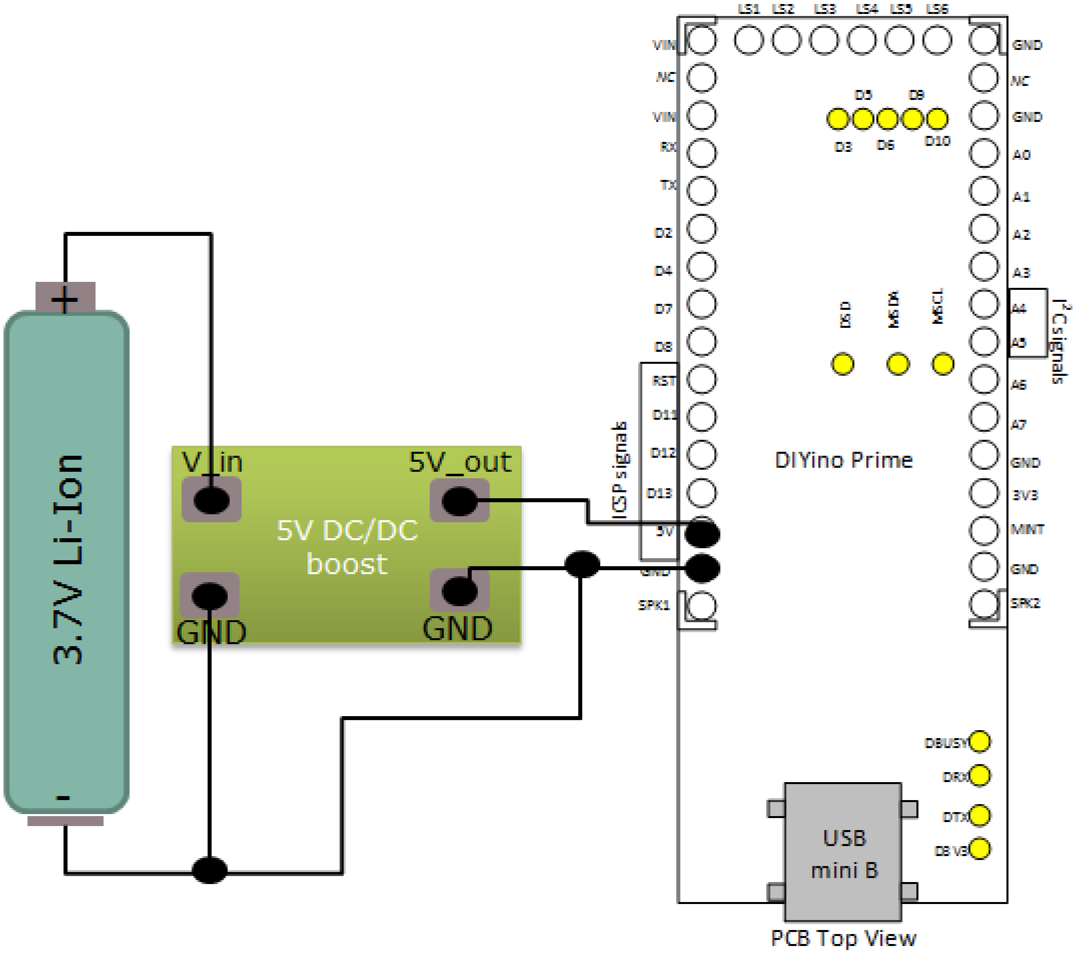


FIGURE 4: SUPPLY SCHEME USING 5V DC/DC BOOST CONVERTER

4 BASIC WIRINGS OF EXTERNAL COMPONENTS

4.1 WIRING BUTTONS AND SPEAKER

Figure 5 shows wiring of the switches and that of the speaker.
The speaker has to be connected between the SPK1 and SPK2 terminals/pins of the board. 4Ω/8Ω speaker can be used, up to 3W output power. It does not matter which terminal of the speaker you connect to which pin.
Switches – latching or momentary, although mostly momentary switches are supported by Arduino libraries – are connected with one terminal connected to GND and the other terminal to a digital I/O. Please note that although in the Figure 5 the Main and Aux. switches are connected to D12 and D4 respectively, any digital I/O of the DIYino Prime board can be used from D2 to D13 and A0 to A5. Please

note that certain pins have pre-defined functionality, please refer to the DIYino Prime schematics and the Chapter 2.2 .

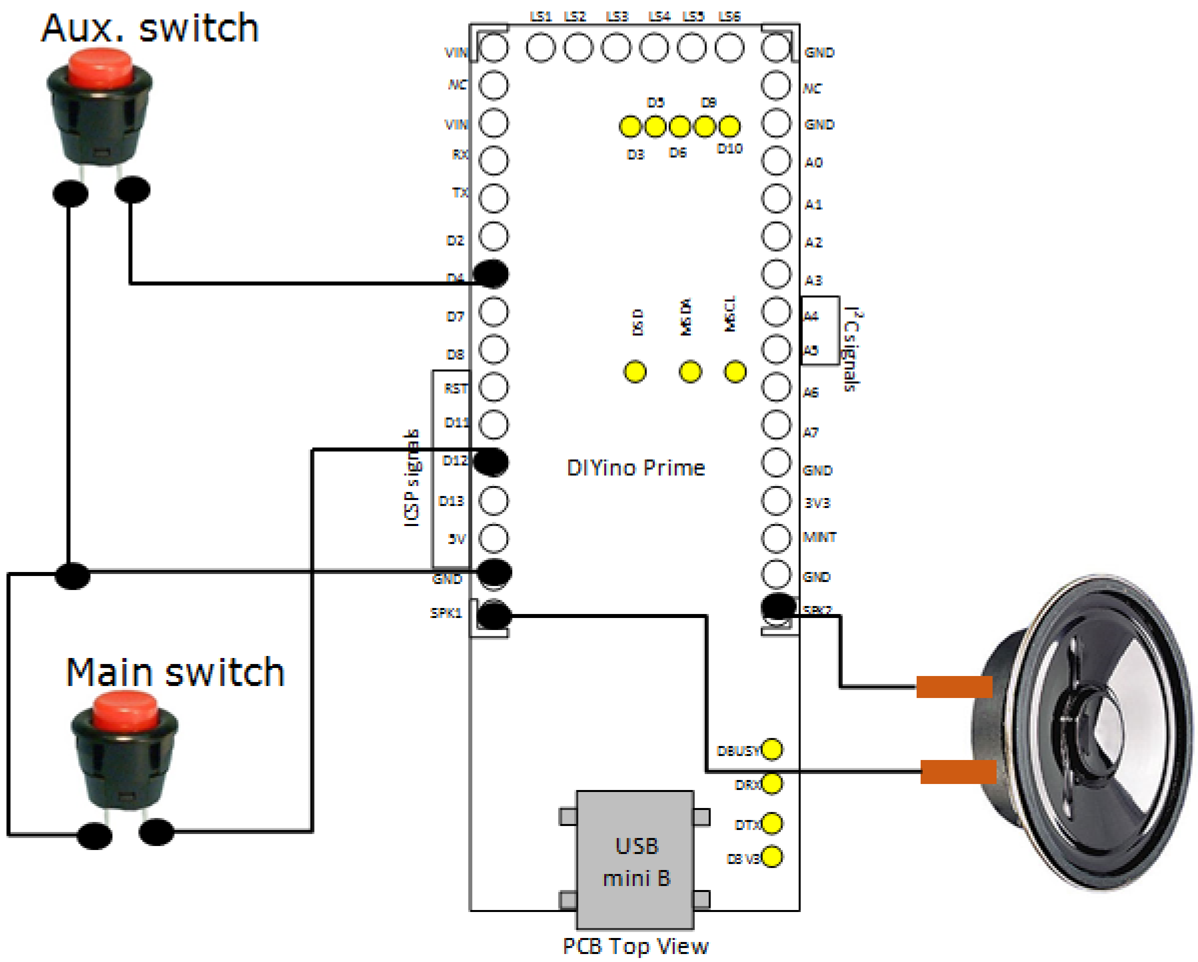


FIGURE 5: WIRING SWITCHES AND THE SPEAKER

4.2 IN-HILT RECHARGE

It is very convenient if the battery does not have to be removed from the hilt every time it needs recharging. Therefore so called in-hilt recharge ports found a wide-spread use in saber hilts together with rechargeable batteries (most common type being the 3.7V type 18650). The wiring of the recharge port is depicted on Figure 6, showing the wiring only for the Standard Supply Scheme introduced in Chapter 3.1. The commonly used 2.1mm DC sockets have 3 terminals, one is wired to the rechargeable voltage source (~battery) negative labelled ***Battery(-),*** one to the Board GND labelled ***Board(-),*** and the third one labelled ***Battery(+) and Board(+)*** to both the rechargeable voltage source positive lead and to the Board. The exact connection of ***Battery(+) and Board(+)*** to the Board depends on the supply scheme used, described in the Chapter 3.

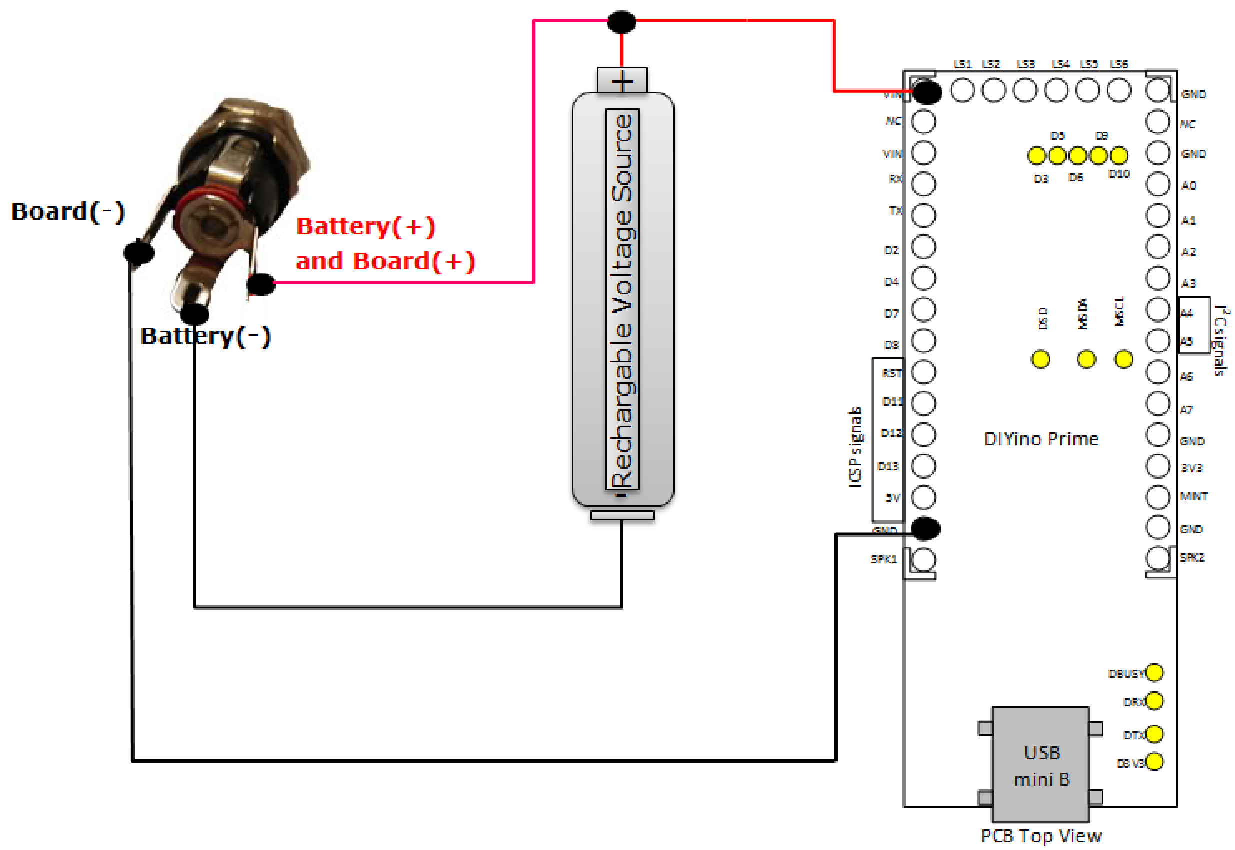


FIGURE 6: IN-HILT RECHARGE USING A 2.1MM DC SOCKET

4.3 AUDIO CONTROLLED FLICKER

It gives a very organic and vivid feel to your lightsaber if the subtle power fluctuations of the plasma blade are simulated by adjusting the brightness of the blade. It is especially spectacular if the fluctuations are synchronized with the beat of the sound font audio, therefore a method called Audio Controlled Flicker has been developed. The output signals of the audio amplifier (SPK1 and SPK2) are converted by Analog-to-Digital converters, available at each A(x) pins, and the resulting differential value is used to adjust the PWM level of the LS(x) drivers.

To enable this feature, connect SPK1 to A6 and SPK2 to A7 (any other A(x) pins can be use, except for A4 and A5, but since A6 and A7 are analog inputs only, it makes sense to use them). In the code simply capture the analog values in each loop with the analogRead() function and calculate the amount by which you want to adjust the PWM level of the LS(x) drivers.

Please note: there is no required intra-board connection necessary for operation of the DIYino Prime board. The connections described in Chapters 4.3 and 4.4 implement additional features which might be useful for advance users but in no way necessary for the proper working of the integrated circuit board.

4.4 AUXILIARY SIGNAL WIRING

Auxiliary signals are provided to access certain internal signals of the DIYino Prime board which are not connected to the periphery but might be needed for special application cases. Auxiliary signals are marked with a yellow dot on the board drawing throughout this document. They are not needed to be wired for proper operation of the board. Usage of the auxiliary signals is in the sole responsibility of the user and therefore please ensure that you understand the purpose of these signals and know what you are doing.

D3,D5,D6,D9,D10: these auxiliary signals connect to the PWM capable Atmega328P pins, except for D11, which is connected to the periphery as a main signal due to its alternate functionality as MOSI SPI pin. D3 is an interrupt capable pin, so if you do not use LS3 (for which D3 is the gate signal), you can also use D3 as an interrupt input.

MSDA,MSCL: I2C signals of the MPU6050 accelero- and gyrometer. MSDA is connected to A4 pin of the Atmega328P and MSCL to A5, both of which are also accessible on the periphery as main signals. Provided for debug purposes.

DSD: shut-down input of the audio amplifier chip LM4871 (IC5). If connected to a digital output it can be used disable the audio amplifier in low power mode. To enable this feature, R55 resistor has to be removed and R52 jumper to be bridged in order to facilitate the R53 pull-up resistor. You can also choose to program your digital output with pull-up enabled, in this case you can leave the R52 untouched. Connect a free digital output to SDS and program Low level to enable the audio amp.

DBUSY: active low busy signal of the YX5200-24SS MP3, with pull-up resistor. When driving digital Low, a sound file is being played.

DTX: TX line of the YX5200-24SS MP3 for software serial programming. Connected to main pin D7. Provided for debug purposes.

DRX: RX line of the YX5200-24SS MP3 for software serial programming. Connected to main pin D8. Provided for debug purposes.

R3V3: 3.3V LDO of the YX5200-24SS MP3 chip. It can be used to power devices requiring 3.3V supply. As the chip manufacturer does not specify the current capability of this LDO, please only supply devices which require a few tens of mA max.

4.5 Wiring an OLED display (for blaster props)

If you want to add a small display to your prop project - especially popular with sci-fi blaster, but you can include one for a custom MP3 player, armor gadgets etc. – there are good OLED displays using the SSD1306 driver IC.

The wiring of such an OLED display using I2C communication can be seen on Figure 7 . You need to supply the OLD display with 3.3V (some OLED displays have in-built 3.3V LDO, in which case you can also use the 5V pin to supply them) and use the Atmega328P's I2C bus signals A4(SDA)/A5(SCL) to communicate with the display.

Display size consideration: most OLED libraries (for instance the one from Adafruit: https://github.com/adafruit/Adafruit_SSD1306) reserve RAM space for the entire display. Let's consider a commonly available 128x64 OLED display. It has 128x64=8192bits=1024bytes, that is exactly 50% of available RAM space. If you would like to have space allocated for variables, you need to go down with the size to 128x32 ("only" 25% taken up) for instance, or lower.

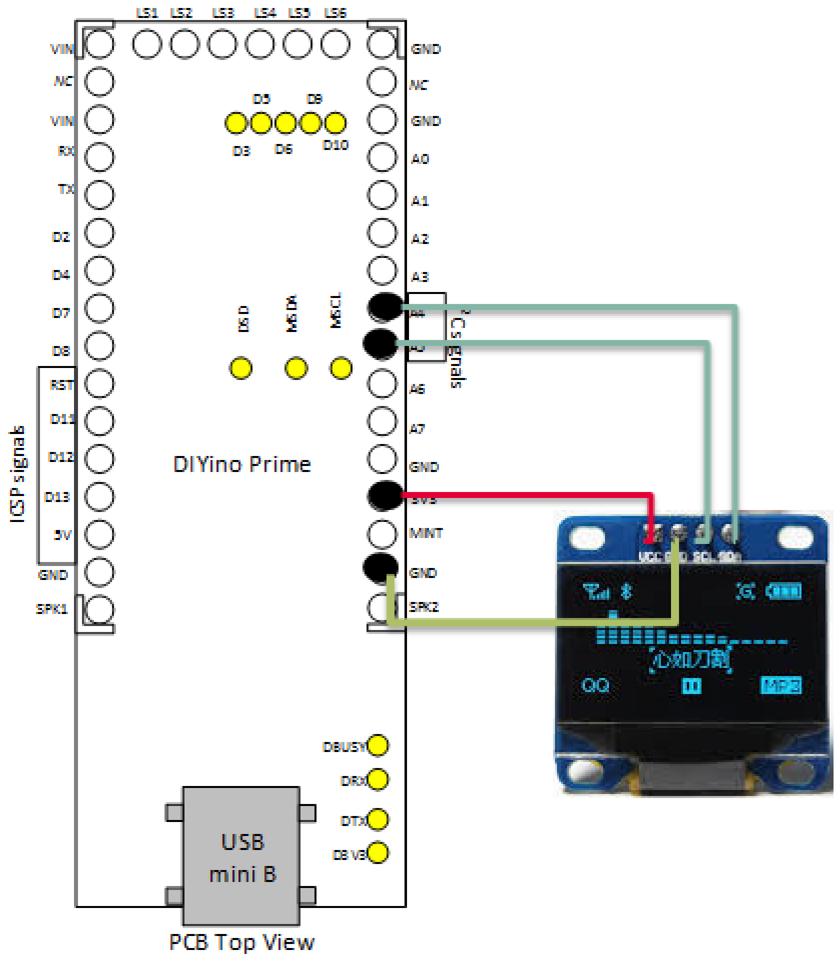


Figure 7: wiring of an OLED display using the SD1306 OLED controlled (128x32)

4.6 SD-Card Preparation

On DIYino Prime V1 the MP3/Wav player chip YX5200-24SS is connected to the SD-card. Sound files stored on the SD-card will be indexed and accessed according to their physical copy order to the SD-card. Therefore some simple rules apply to define SD-card content.

1. Format your SDCard!
2. Select all the files from this folder and **"Drag and Drop"** them to your SDCard. **NO COPY AND PASTE !!!** : We need to have this file copied in the same order as their filename order. On Microsoft Windows, Copy/paste produce an anarchic copy order, but Drag and Drop produce an ordered copy...

You can organize your files in folders to keep a better overview of the content. All what counts to the MP3/Wav chip is the physical order of the files on the SD-card. You can even drag-and-drop whole folders, but inside the folder you need to establish a defined copy order by preceding files names with a numbering for instance:

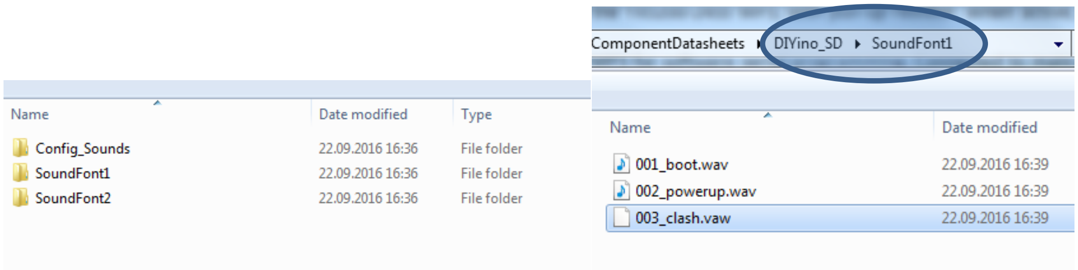


FIGURE 8: ORGANIZING FILES ON THE SD-CARD

4.7 SOUND FONT PREPARATION – HUM-EXTENSION

The MP3/WAV chipset YX5200-24SS is capable of gapless playback in case of WAV files, i.e. it can change from one sound track to another without an gap in the audio. This feature is mandatory for lightsaber electronics where a seamless transition is expected between the different sound files (like hum, clash, swings etc.).

While the MP3/WAV chips is capable of gapless playback, especially when it comes to relaunching the background noise of a lightsaber (so called “hum”), it cannot be done without spending a significant amount of effort and code space. For this reason a more pragmatic approach to hum-relaunch is implemented in most of the open source lightsaber software which simply relaunches the hum at certain time intervals. In order not to “loose” the hum if the saber is idling, each sound file (except lockup) has to be extended with a large buffer of hum sounds. This is called a hum-extension. Modifying an existing sound font file is easy by simply copy+paste a chunk of hum sound after the end of the sound file. This can be done with freewares (Audacity for instance).

4.8 UPLOAD SOFTWARE USING ARDUINO IDE

Please visit the Arduino Home Page for instructions how to install the Arduino IDE and upload code using it to your DIYino Prime board:

https://www.arduino.cc/en/Guide/Windows (instructions are also available for Mac OS X and Linux)

The DIYino Prime boards are configured as an Arduino Nano compatible boards, so when choosing your board please select Arduino Nano:

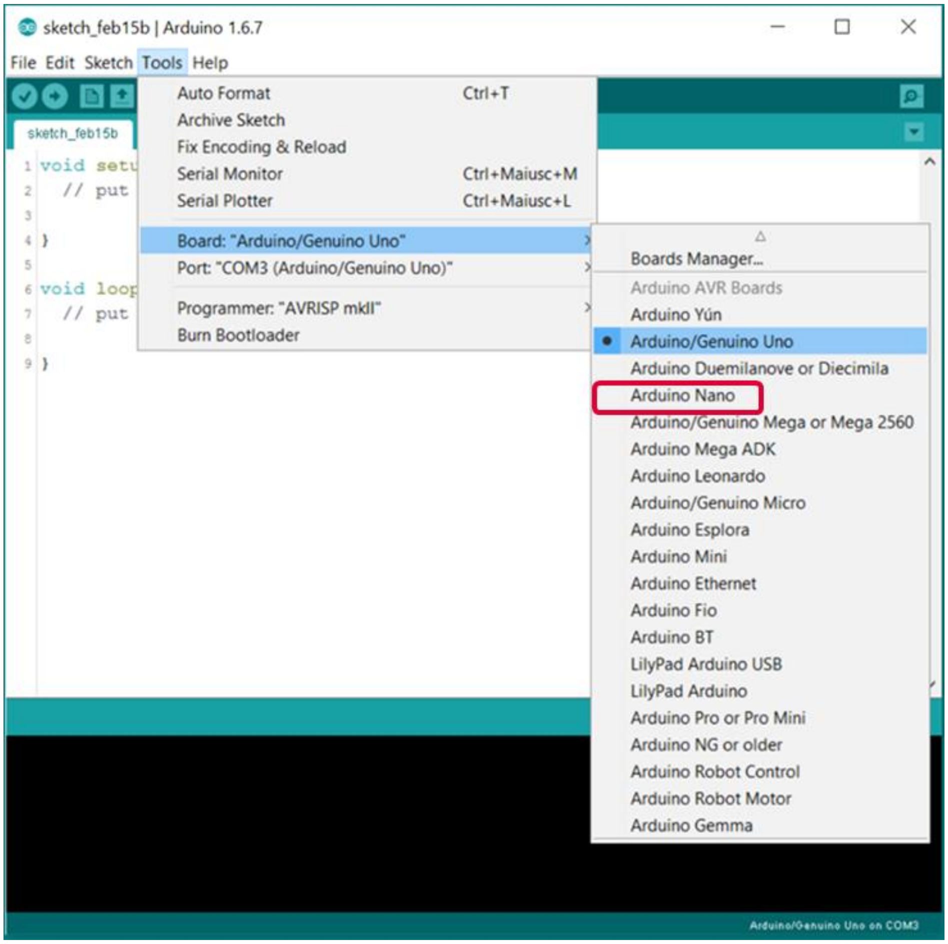


FIGURE 9: SELECTING THE BOARD TYPE IN THE ARDUINO IDE

5 APPLICATION EXAMPLES

Warning : High-power LEDs (such as the Luxeon, Cree atc. Brand LED) and strings/stripes of LEDs (such as LED strings composed of many single LEDs or neopixel LED moduls such as WS2812B) are extremely bright. Especially High-power LEDs are considered “class 2 lasers”! You should neither look directly to the beam nor point someone with it when the light source is not diffused/blocked, just like a powerful lamp or flashlight. Manufacturer of the DIYino Prime board could not be held responsible for any bad use of high-power or other type of LEDs/LED modules. To avoid injuries and retina damage due to the high brightness of LEDs, always use protective googles or other means to avoid looking directly into the light source and also take care to protect others (like children) from being able to look directly into the light source.

In this Chapter the most common lighting options of saber designs will be discussed with application notes and wiring diagrams showing how these options can be interfaced to the DIYino Prime board.

Caution: in case of a High-Power LED or LED-string setups, you can use the LS pins to adjust the current flowing through the LEDs using PWM control (Pulse Width Modulation) of the transistor gates. The transistors which connect the cathode (-) of the LEDs to the GND can act as voltage controlled variable resistors, thus limiting the current through the LED. For those LEDs having a Vth above the battery voltage, this intrinsic current limiting is sufficient to ensure no overvoltage/overcurrent to the LED, while still offering the full dynamic range of brightness control. But in case the LED has a Vth below (or even far below) the nominal battery voltage (i.e. for red/amber/yellow LEDs), depending on the electrical characteristic of the LED, this intrinsic limiting might not be enough to ensure no overvoltage to the LED, which in turn can lead to damage or degradation of the life time of the LED. Even if the limiting through the transistors is deemed sufficient, during debug the PWM level can be set accidentally to a level which causes overvoltage. Last but not least, if only a small portion of the available PWM range can be used to control the brightness, it can lead to less smooth color blending. Therefore in doubt please include a limiting series resistor, which can be calculated using the following formula:

$$Rseries = \frac{Vbattery - Vth\_LED}{I\_LED}$$

Example: suppose you use a Red LED with Vth_LED=2V and you want the current to be I_LED=700mA, you use a Vbattery=3.7V, type 18650 battery. The value of the resistor you need to wire in series to the LED is:

$$\frac{3{,}7V - 2{,}0V}{0{,}7A} = 2.4\Omega$$

Please check the power rating of the resistor you intend to use.

5.1 High-Power RGB(W) LED setup

Figure 1 depicts connection of a High-Power LED module to the DIYino Prime. It depicts a 4-color LED module consisting of 4 LED dies in colors Red, Green, Blue and White. The same wiring can be extended to all different variants of HP-LEDs, like RGB, in which case only 3 of the available 6 Low-side drivers are used, or even for a single die HP-LED which can be controlled with a single Low-side driver.

Please note that that maximum DC current which can be switched by the LS pins is 1.1A. If the HP-LED used in the design involves LED dies with a max current above this rating, you must connect the cathode of the LED die to multiple of these LS pins and ensure they are controlled in tandem to avoid violating the maximum rating of the transistors.

5.2 (Segmented) LED-String setup

The original Master Replicas/Hasbro lightsabers used a ladder or string of LEDs clustered in segments to light them up/down successively thus imitating the ignition and retraction of the blade seen in the movies (also called scrolling FX). Although these commercially available LED string sabers have very high quality hilts, they had a number of shortcomings:

- their LED string brightness is toned down
- the PCB used to hold the LEDs is prone to break or LEDs might fall out after moderate impacts
- due to the LED type used not available commercially, replacing broken LEDs can easily become an issue, because replacement LEDs tended to have a different brightness which will be visible as a spot in the blade
- the blade material was optimized for diffusion at the set working point of the LEDs and if LEDs were replaced/limiting resistors removed to increase brightness, the blade easily becomes "pixely"

Thus people started to build their own LED strings, optimizing them for endurance, but mostly for brightness. Pros and cons of LED string blades are:
Pros:

- without doubt the brightest blades out there are built with this technique
- homogenous brightness along the whole length of the blade (as opposed to HP-LEDs whose brightness diminishes towards the tip of the blade)
- most movie accurate ignition/retraction FX with scrolling effect

Cons:

- not recommended for full contact dueling as individual LEDs/strings are prone to break
- no color changing possible – only for the dedicated!!!
- special diffusion layers and blade tubes needed
- major effort to build (not really a con though for those who like 'em!)

There are a number of very good tutorials in the web showing how LED strings can be assembled:
Wayne Schmidt's Ligthsaber Blade Research Page: http://www.waynesthisandthat.com/lightsaber.html
LED String blade for Hampton's Hand-crafted Lightsabers:
https://www.youtube.com/watch?v=_tw0pD8LPtw

Most LED string blades have 6 segments which can be controlled individually; it's on one hand side owing to the original Master Replicas blades having 6 segments, on the other hand sound boards mostly use microcontrollers having up to 6 PWM-capable channels.

The Figure 10 depicts the most commonly used LED string wiring techniques (only one segment is shown, each segment is wired identically). The Serial I. is a straightforward full parallel wiring of all LEDs, i.e. connect all anodes(+) together and all cathodes(-) together. The Master Replicas/Hasbro sabers use this technique as they use 3xAA/AAA batteries as voltage supply, resulting in 3x1.5V=4.5V max. To lower the voltage to the Vth of the used LEDs, small SMD resistors in series to the individual LEDs are used.
Due to the large number of LEDs used in LED string blades (usually between 60 and 100, but more can be added, depending on the LED length etc.) the current consumption of a Serial I. string can reach several Amps. To lower the overall current but still drive the LEDs at the same working point, LEDs can be connected in series, resulting in half the current in comparison with a full parallel connected string, of course the trade-off is, that in order to supply the same power, the voltage has to be increased accordingly.
In the Serial II. technique two times N LEDs connected in parallel are connected in series in turn. This technique has been developed for blades supplied with a 7.4V battery setup and green or blue LEDs, as their Vth of typical 3.4V is close to the voltage of the battery (2x3.4V=6.8V, 600mV below the nominal battery voltage). The Serial III. technique is similar to the Serial II. with the difference that it connects one additional group of N LEDs connected in parallel serially to the rest. It is ideal if the Vth of the used LEDs

is lower as is the case for i.e. warmer colors like red, amber or yellow LEDs with a typical Vth of ~2V. Connected in series 3x2V=6V is close to 7.2V.
In the Chapter 6.1 a full wiring diagram of a Serial II. wired blade with the DIYino Prime is shown as reference (see Figure 13).

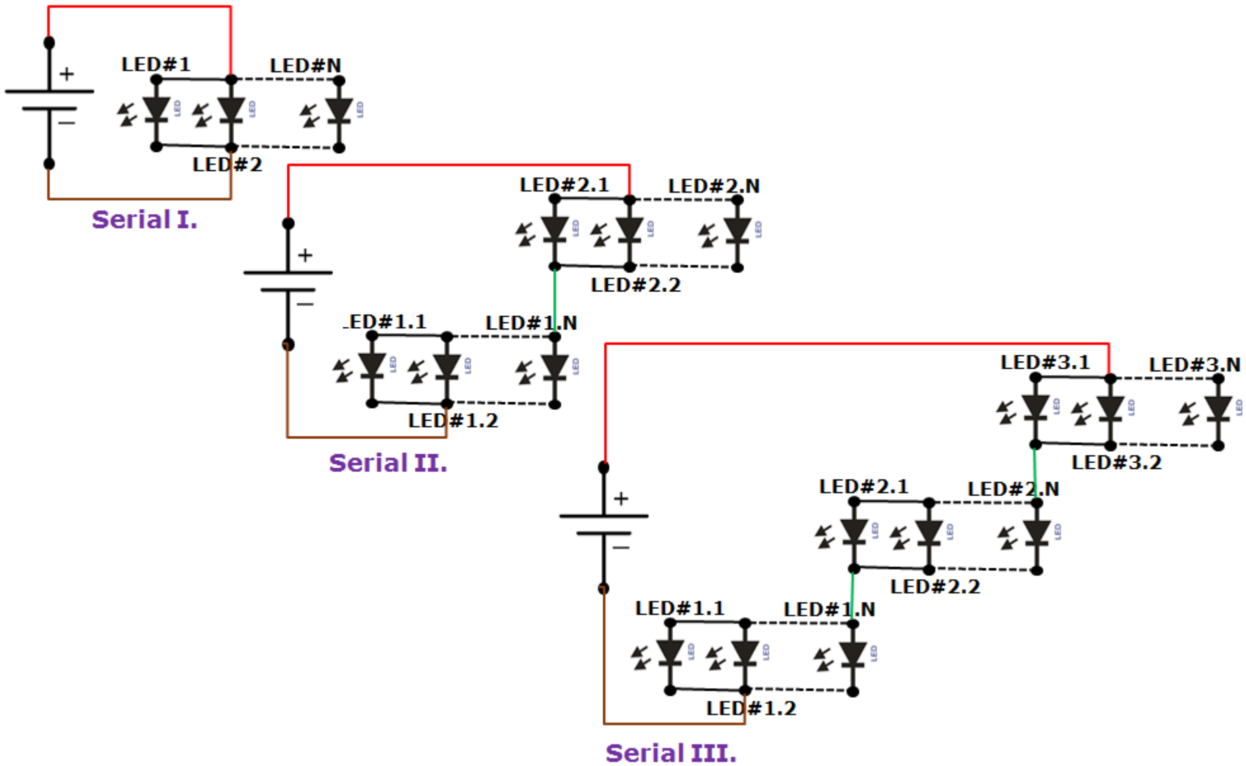


FIGURE 10: LED STRING WIRING TECHNIQUES – SERIAL I., II. AND III.

The number of LEDs in a LED string blade can be chosen according to individual taste, target brightness, LED dimensions etc. Most commonly used blades are ~30" long (~80cm), i.e. if 5mm LEDs are used, a maximum of 160 can be squeezed in theoretically. Practically the LEDs cannot be pressed together so tightly and the wiring/assembling will leave small gaps between the LEDs. Based on personal experience the recommendation is to use max. ~100 LEDs. Of course the final number must be divisible by 6, closest number is 96, with Serial II. wiring it gives 6 segments x 2 serial x 8 parallel=96. In case Serial III. wiring is used (e.g. red blade) the total number must be also divisible by 3. A good number would be 6 segments x 3 serial x 6 in parallel=108 (as red LEDs tend to be less bright, more LEDs give a better result).

5.3 NEOPIXELS SETUP

"NeoPixel" is Adafruit's brand for individually-addressable RGB color pixels and strips based on the **WS2812**, **WS2811** and **SK6812** LED/drivers, using a single-wire control protocol. Commonly used neopixels stripes are composed of individual LED segments connected together to form a ladder similar to LED-strings. The stripes can be cut at any joint and multiple striped can be connected together at these joints as well.

Neopixel LED modules integrate RGB LEDs and a control circuit which uses PWM (Pulse Width Modulation) to control the brightness of each die individually. For that purpose each LED module has a shift register composed of 24-bits, 8-bits belonging to each of the colors Red, Green and Blue. The shift registers are connected between the DI (Data-In) and DO (Data-Out) pins of the individual segments. In a stripe configuration one segment shift register is connected serially to the shift register of the next segment. If a blade has a stripe with 100LEDs, it means during programming 100x24=2400 bits of data have to be transmitted to the stripe using neopixels own serial protocol to fill up all PWM registers, which in turn determine the brightness of the LEDs. This takes only a few us, so programming can happen so fast, that transitions seem smooth to the eye.
A neopixel stripe has only 3+1 signals:
5V: supply of the stripe
DI: Data-In for the single-line serial protocol
GND: Ground or negative of the stripe
DO: Data-out, this signal has to be used only of you want to connect several stripes together which are not continuous (back-to-back stripes or think about the cross guard of a Kylo Ren style saber)
The specification of the neopixel LED segments defines 5V as nominal voltage for the stripes, however the module can work with a much lower voltage as well. A lower voltage is even necessary to minimize power loss during operation, because any excess voltage above the voltage threshold of the used LEDs (Red ~2V, Green and Blue ~3.5V) is “wasted” over protection circuits in the control logic. Therefore a voltage source around the LED Vth is ideal to power neopixel strings.
For a complete characterization of neopixel brightness and current consumption please see LINK
Neopixel stripes can be wired to the DIYino Prime board in two different ways, illustrated in the following two Chapters.

5.3.1 Basic neopixels wiring

The simplest way to connect neopixel stripe to the DIYino is to connect the Battery(+) of a 3.7V supply to the 5V terminal of the first segment of the stripe, the Battery(-) to the GND terminal of the first segment and a digital I/O (in this example D13 is used, as depicted on Figure 11) to the DI (Digital-In) terminal of the first segment.
Connection of the GND signal can be realized over the DIYino Prime GND pins as well. Please note that if you use in-hilt recharge, you connect the recharge port to the GND of the board and the stripe as in Figure 6.

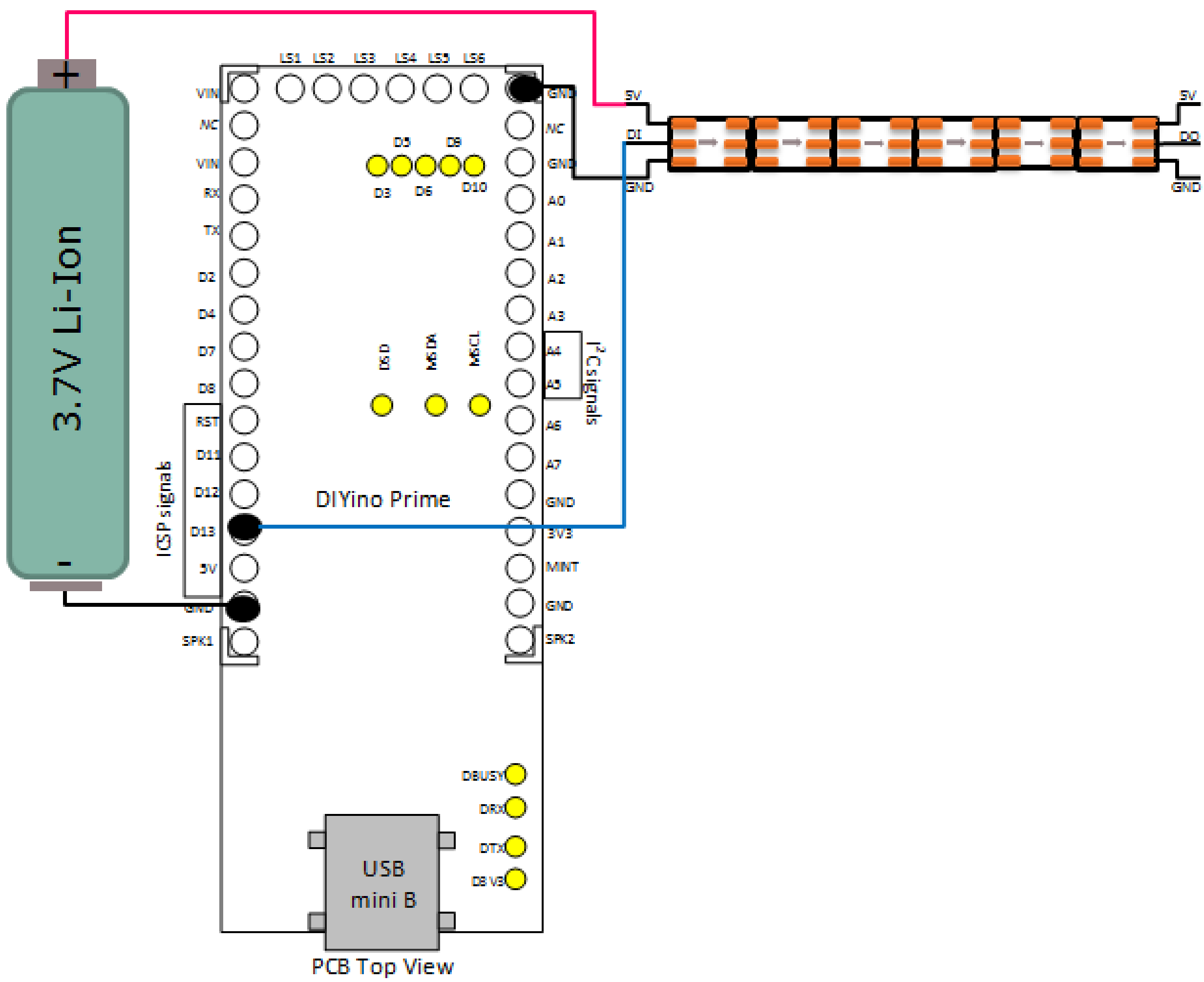


FIGURE 11: WIRING OF A NEOPIXELS STRIPE USING 3.7V VOLTAGE SOURCE

5.3.2 NEOPIXELS WIRING WITH PROGRAMMABLE KILL KEY

Neopixels chips consume considerable power even when all the LEDs are switched off (all 0's). This static current consumption amounts to 1mA per LED. You can quickly calculate what this means to your battery life time if you use – let's say – 60LED/m type of neopixels, back to back, in a 80cm blade. There will be 100 LEDs in your blade, drawing 100mA even if all LEDs are switched off. It will even discharge the best battery in less than one day. It is not hard to predict, that with the introduction of the 144LED/m type neopixel stripes, some blades will include much more than 100 LEDs, which will lead to an even quicker discharging of the battery. Killing power to the circuitry using a kill-key is a good method to lengthen the shelf life of your saber, but quite annoying during a show if you have to keep plugging in/out the kill key. Therefore x a unique method was invented using the existing DIYino Prime architecture. If the GND pin of the neopixels stripe is connected to the LS pins instead of the GND of the battery/DIYino Prime, the transistors of the LS pins can be used to cut power to the neopixels stripe. If the blade is activated, the transistors have to be fully switched on (using the digitalWrite function) to connect the GND of the neopixels to the Battery(-), therefore powering the stripe. If the blade is retracted/switched off, the transistors have to be fully switched off in order to avoid the static current consumption of the stripes discharging the battery. Please take note that the restrictions as to the maximum current capability of the DIYino Prime board apply also here (max 3A!!!)

Connection of the GND signal can be realized over the DIYino Prime GND pins as well. Please note that if you use in-hilt recharge, you connect the recharge port to the GND of the board and the stripe as in Figure 6.

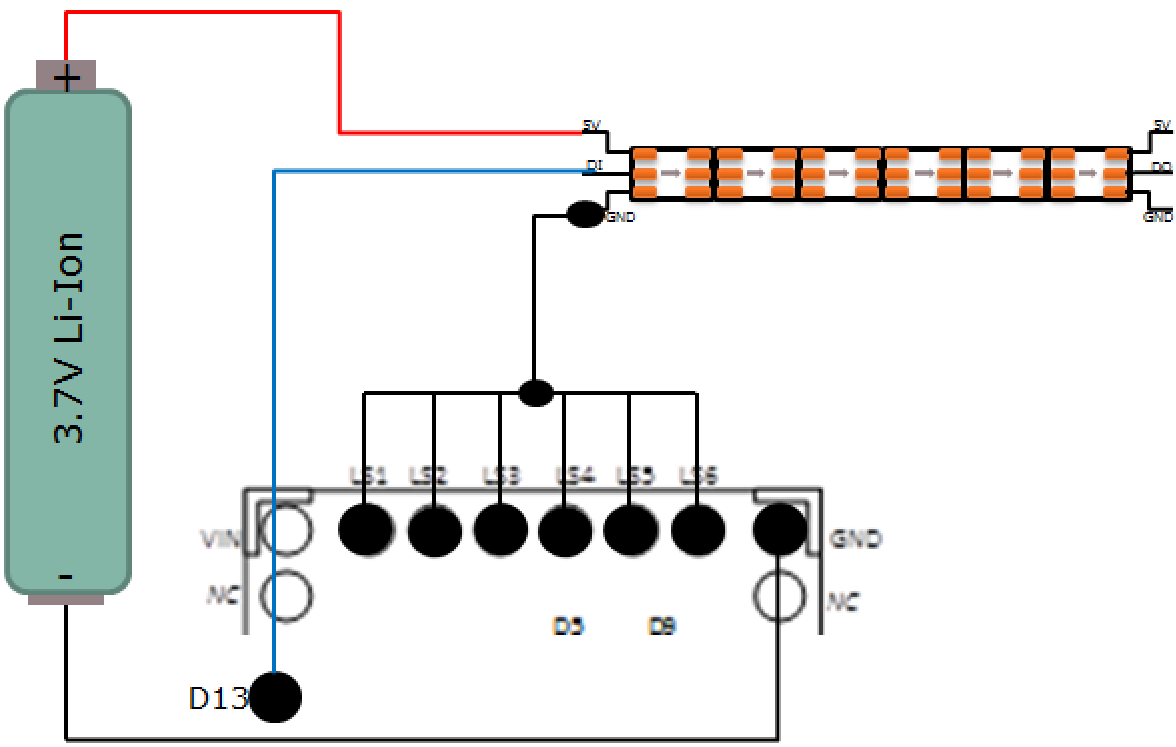


FIGURE 12: NEOPIXELS WIRING USING 3.7V VOLTAGE SOURCE, "KILL-KEY" OF LEDS THRU LS PINS

6 FULL WIRING EXAMPLES

6.1 7.4V SUPPLY LED-STRING SETUP WITH SERIAL-II TECHNIQUE

The wiring diagram on Figure 13 shows a LED String blade wiring using Serial II. LED wiring (please see Chapter 5.2). It features:

- 2x3.7V batteries connected in series to give 7.4V
- 6 segments of green/blue LEDs
- A main and and aux. switches
- Speaker
- Connections for Audio controlled flicker
- an in-hilt recharge port

The common anode of the Serial II. segments are connected directly to the battery+ (or alternatively to VIN pin of the DIYino Prime board). The individual cathodes of the Serial II. segments are connected to an LSx pin each and controlled using PWM. The Battery+ is connected to the VIN input of the board, which is regulated to 5V by the on-board LDO, thus supplying the on-board electronics.

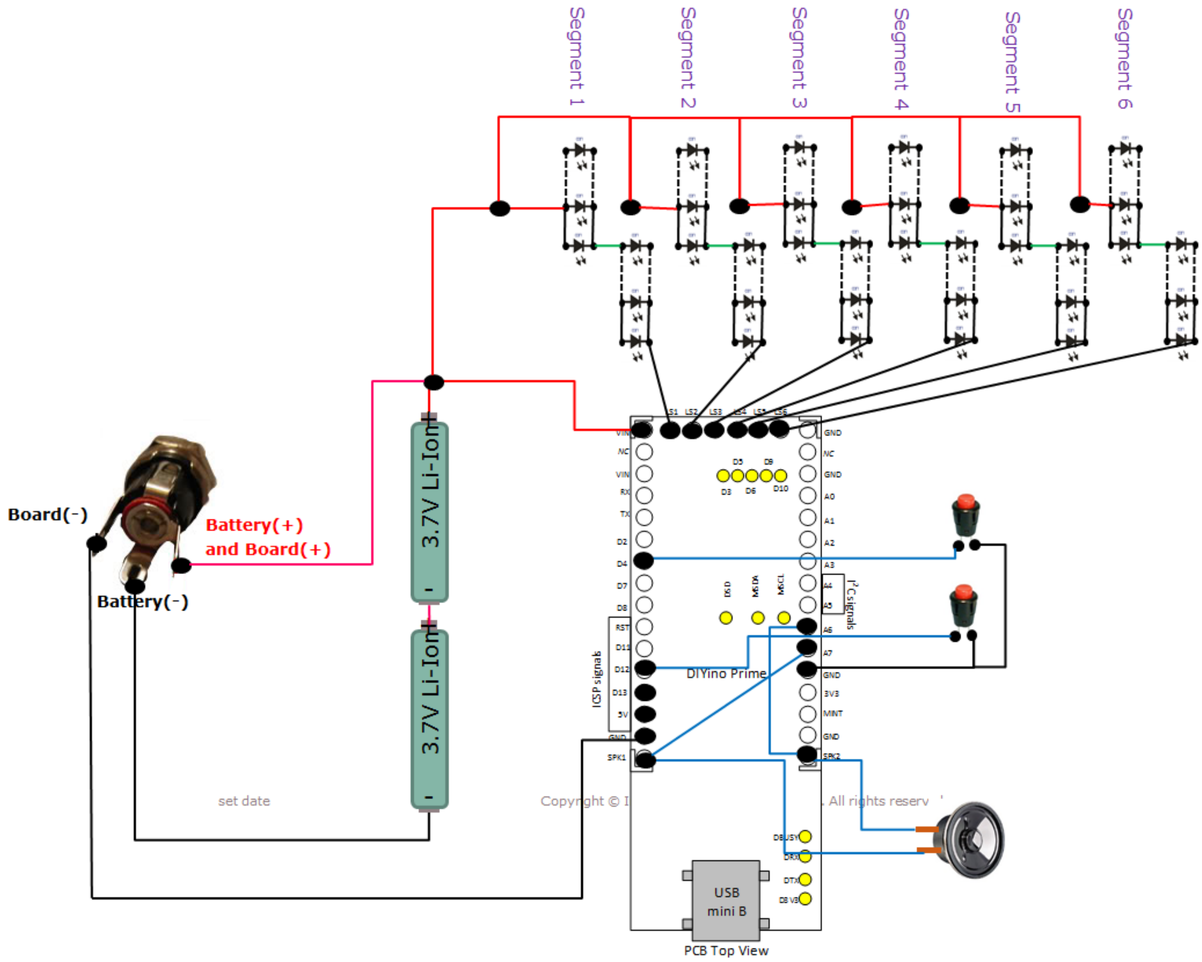


FIGURE 13: FULL WIRING DIAGRAM FOR SABERS BUILT WITH SERIAL II. LED STRING BLADE

6.2 RGBW HIGH-POWER LED SETUP WITH 3.7V SUPPLY AND 5V DC/DC

The Figure 14 shows the full wiring diagram of a HP-LED saber using a single 3.7V Li-Ion battery as supply, boosting the voltage to 5V with the help of a 5V DC/DC boost converter. The HP-LED module is directly supplied from the Battery+.

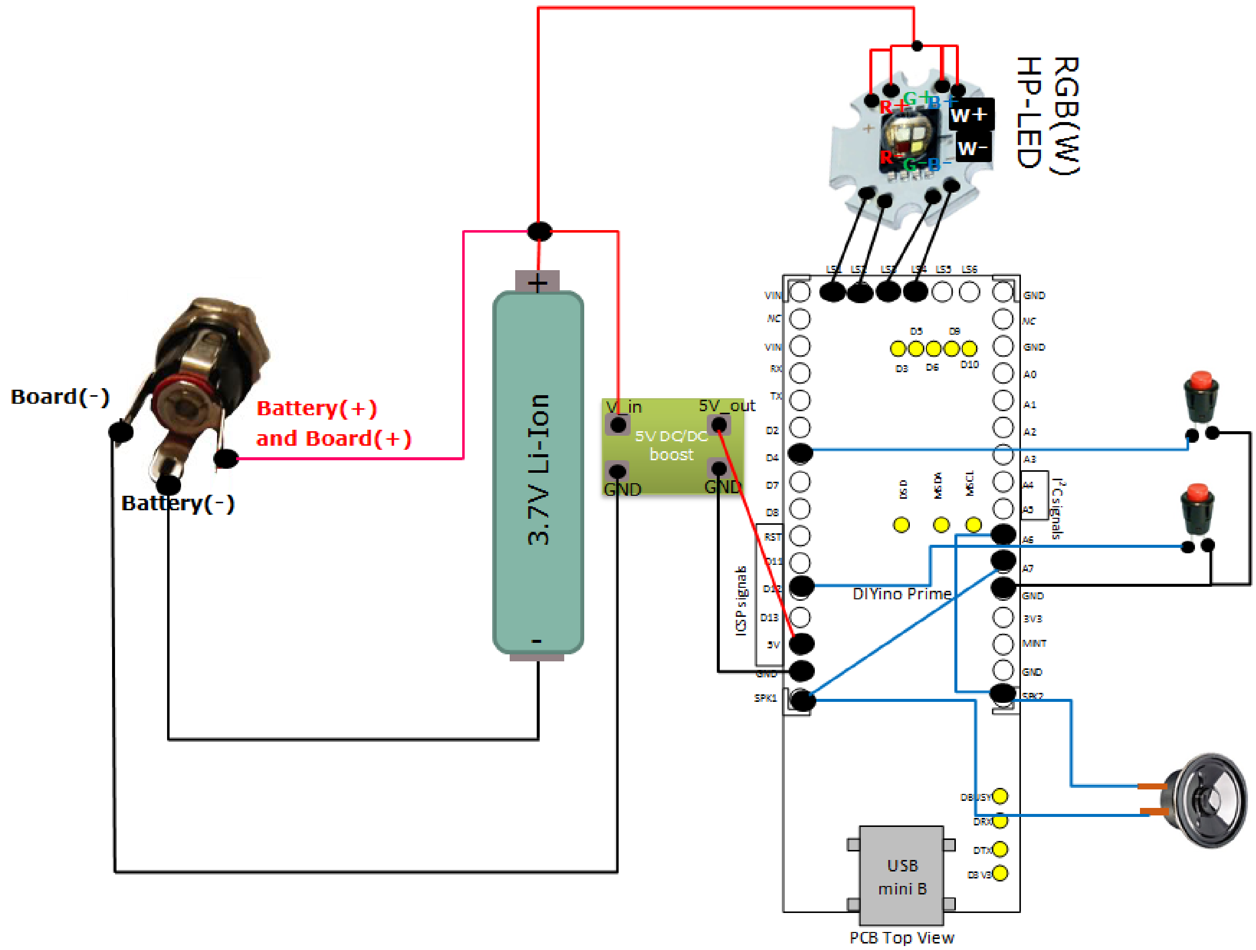


FIGURE 14: FULL WIRING DIAGRAM OF A HIGH-POWER LED (RGBW) SABER

6.3 NEOPIXELS SETUP WITH PROGRAMMABLE KILL-KEY AND 3.7V SUPPLY

On Figure 15 the full saber wiring diagram can be seen. It includes

- an in-hilt recharge port
- 3.7V Li-Ion battery as voltage source (for instance 18650)
- A neopixels LED-stripe of (theoretically ☺) any length
- Programmable neopixels stripe kill-key setup
- A main and and aux. switches
- Speaker
- Connections for Audio controlled flicker

When connecting the programmable neopixels stripe kill-key, you have to calculate max. current consumption of your LED stripe and connect the LSx pins accordingly, i.e. if you anticipate 2A-3A's, connect 3 of the LS pins to the GND terminal of the stripe. Up to max. 3A over all LSx pins due to limited heat dissipation on the board!!!

All considerations in the Chapters describing the individual parts of this circuit diagram apply here as well. Please read them carefully.

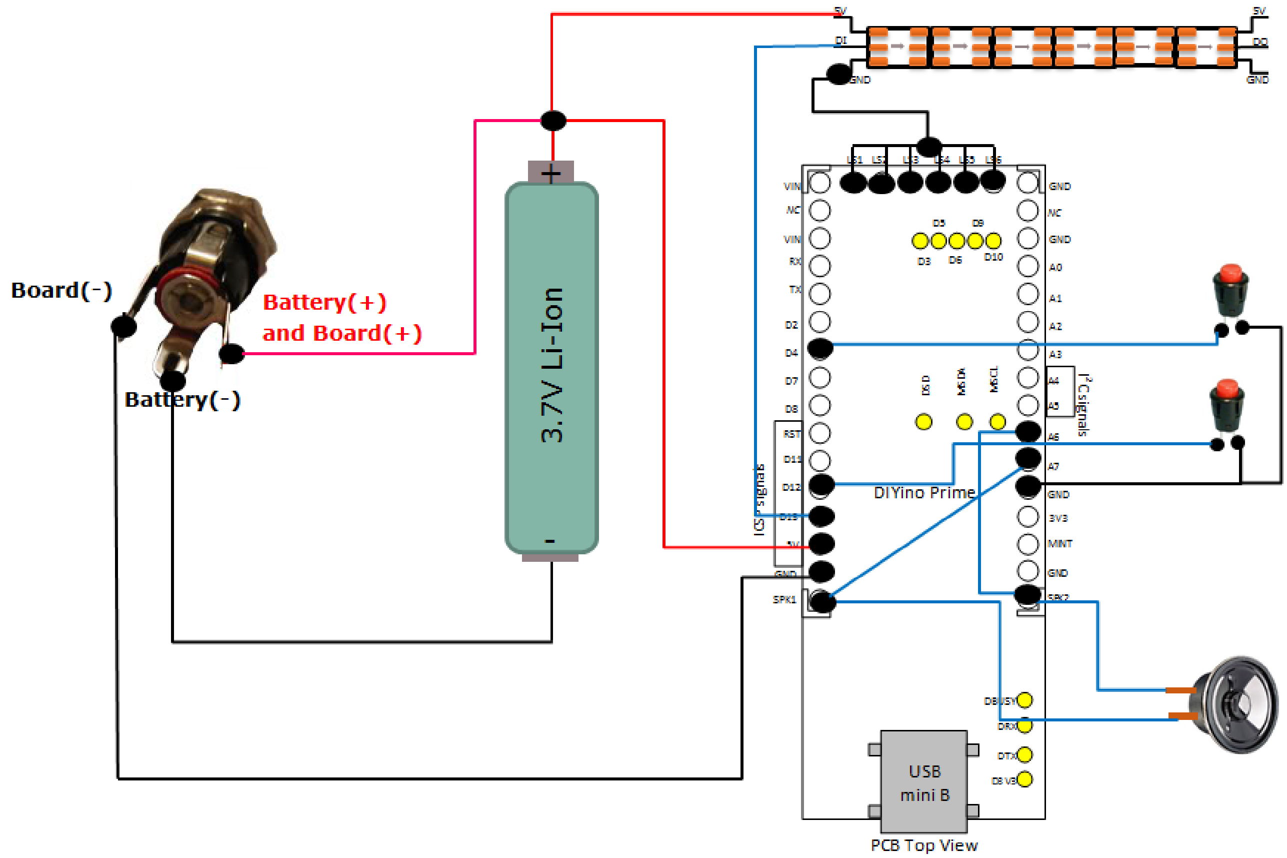
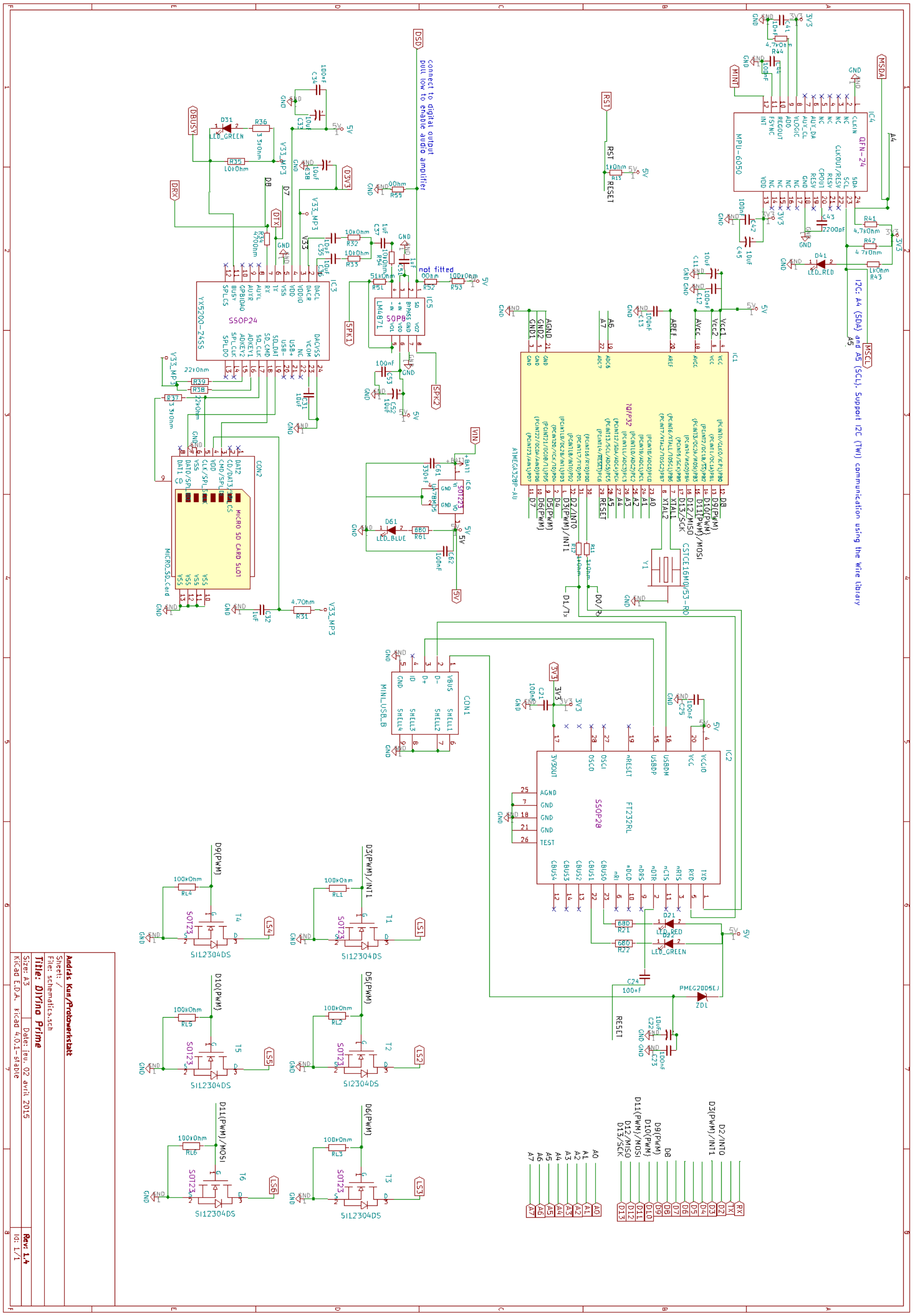


Figure 15: full wiring diagram of a neopixel stripe based saber

7 Circuit schematics


MPU-6050
YX5200-24SS
LM4871
ATMEGA328P-AU
FT232RL
MINI_USB_B
MICRO_SD_Card
SI2304DS

8 RELATED LINKS

LSOS: https://github.com/neskweek/LightSaberOS
USaber: https://github.com/JakeS0ft/USaber