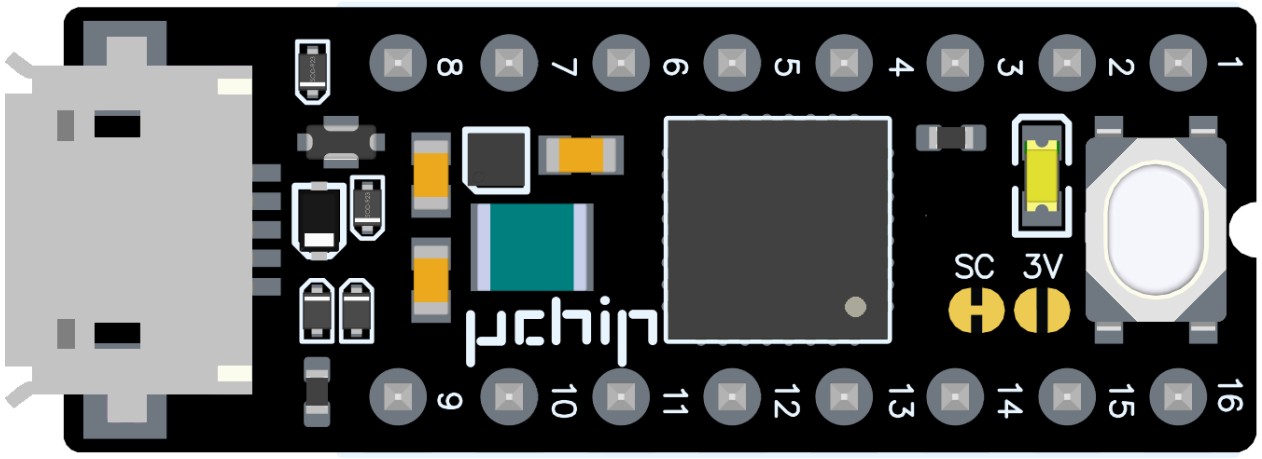
µchip quick start manual



Document Version 1.5 – 11/01/2020

Table of contents

INTRODUCTION

Thank you for buying uChip!

This document covers the first steps required to use this small development board.

- Integration with Arduino
- Driver Installation
- Integration with Atmel Studio 7.
- Hardware and pinout

uChip is a small, open-source, Arduino[1]-compatible development board, which fits a low cost 16-pin socket. Featuring a Cortex M0+ ATSAMD21E18, it implements most of the functions of the Arduino Zero[2], including a fully functional OTG USB Host. Like Arduino Zero has 256 kB of flash and 32 kB of RAM.

uChip also includes integrated buck and boost DC-DC converters, which allow:

- to generate the 5V@500mA for the USB Host, when the external voltage is between 3.3V and 5V.
- to provide externally 3.3V or the USB voltage at up to 1A, when uChip is USB-powered.

A support circuitry also prevents reverse current between the external voltage and the USB, when uChip is not working as USB host. Furthermore, the support circuitry also implements an automatic power selection between the USB and the external voltage.

When uChip is USB powered, the output voltage (VEXT) provided to pin 16 can be software selected, between 3.3V and the USB voltage (nominally 5V[3]).

Every uChip is 100% tested, with a very rigorous procedure:

1) The PCB is optically and electrically inspected.
2) The assembled PCB is optically inspected.
3) A test program is uploaded. The test involves checking all the pins, the DC/DC converters (up to the maximum current), the USB port, the on-chip flash and RAM and read-out of the SAMD21 chip ID. During all the steps, the input and output currents and voltages are monitored so that they comply with the expected values.
4) The microcontroller is erased and a bootloader is flashed. Bootloader protection fuses are programmed as well.
5) Finally, the bootloader functionality is tested.

This makes sure that defective units do not reach the market.

ESD Warning! uChip contains electrostatic sensitive devices. Electrostatic charge might permanently damage uChip. Observe precautions when handling it. Itaca Innovation is not responsible for any damage caused by ESD or improper handling/usage.

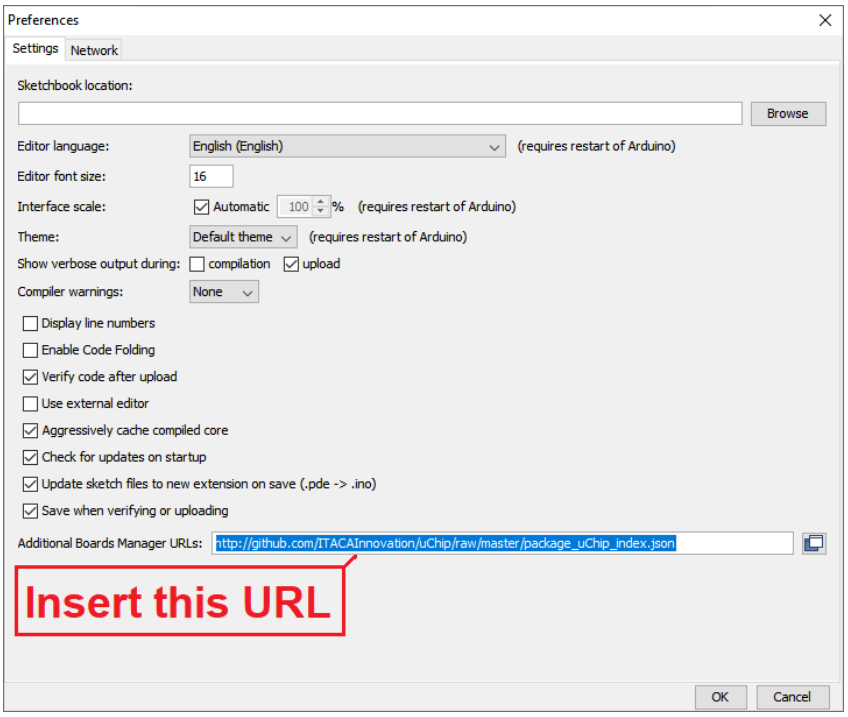
[1] Arduino is a trademark of Arduino s.r.l. There is no connection between Arduino s.r.l. and Itaca Innovation S.R.L.
[2] Some functions and pins might not be 100% implemented due to the reduced pin-count.
[3] Note that, when the USB voltage is selected, the actual output voltage will depend on the USB voltage (which might be as low as 4.4V) and the output current (the voltage drop is typically 0.5Ω × I, where I is the output current in amperes). When the 3.3V is selected, the output voltage will depend only on the current, with the typical relationship: VEXT= 3.3V - 0.15Ω × I.

GETTING STARTED

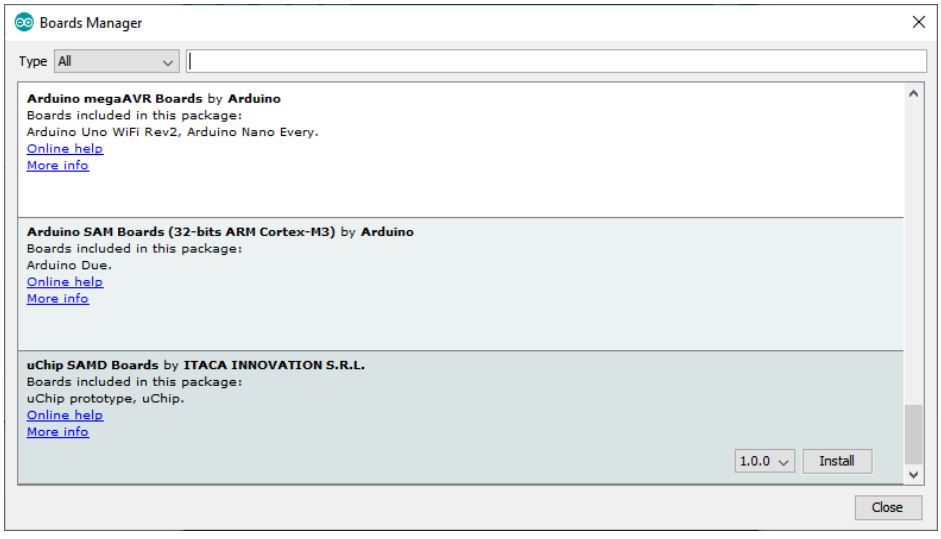
Integration with Arduino IDE

We assume that you already installed the Arduino IDE. If not, download it from here https://www.arduino.cc/en/Main/software and follow the instructions.

1) Open the Arduino IDE.
2) Choose File → preferences from the menu. This will open the "Preferences" window.
3) In the settings tab, add the following line, to the "additional Boards Manager URLs"
   http://github.com/ITACAInnovation/uChip/raw/master/package_uChip_index.json



4) Press OK. This will close the "Preferences" window.
5) Select "Boards Manager" from the Tools→Board: menu. This will open the "Boards Manager" Window.



6) Find and select uChip SAMD boards from ITACA INNOVATION S.R.L. and click on “Install”.
7) Close the window.
8) Now, depending on your OS:
   Windows 10, linux, Mac OS X: **No drivers are required,** and you are ready to go!
   Windows 8, 8.1, 7, Vista and XP: Driver installation is **required (see next section).**

Driver Installation

If you installed the uChip package on Arduino, the drivers are located here[4]:

%USERPROFILE%\AppData\Local\Arduino15\packages\uChip\hardware\samd\1.0.1\drivers\

You can launch the installer by following these steps:

1) Open the “Run” dialog box by pressing and holding the Windows key, then press the R key (“Run”).
2) Write the following line:
   %USERPROFILE%\AppData\Local\Arduino15\packages\uChip\hardware\samd\1.0.1\drivers\uchip_driver_installer.exe
3) Press OK

The installation procedure should take few seconds. Windows will ask the permission to install the drivers.

Alternatively, if you do not want to install the Arduino IDE, you can download the drivers here[5]:

https://github.com/ITACAInnovation/uChip/raw/master/uChip1.0.1.zip

Once you unzip this archive, you will find the drivers “uchip_driver_installer.exe” in the directory:

uChip1.0.1\drivers\

You can just double-click on uchip_driver_installer.exe.

Integration with Atmel Studio 7 IDE

uChip can be either programmed using a SWD programmer/debugger, such as JLINK, or by its USB port.

To be able to program uChip through the USB port, you must perform the following preliminary steps.

Warning: THESE STEPS ARE FOR BOSSA 1.7. BOSSA 1.7 is much faster than BOSSA 1.8 and 1.9. If you still want to use BOSSA1.8 or newer, see next part! (“If you still want to use BOSSA 1.8 or 1.9”)

Warning: these steps assume that you either installed the Arduino IDE and the uChip package, or you have downloaded and installed “Bossa 1.7.0”, available in the archive below:

https://github.com/shumatech/BOSSA/releases/download/1.7.0/bossac-1.7.0-mingw32.tar.gz.

1) Install the drivers, if you are not using Windows 10.
2) Open the Atmel Studio 7 IDE.
3) For the Menu “Tools” choose “External Tools”.
4) Click on “Add”.
5) In “Title” write something meaningful, such as “Program uChip”.
6) In “Command”, locate and choose the “bossac.exe”. For instance, in our system we wrote:
   %USERPROFILE%\AppData\Local\Arduino15\packages\arduino\tools\bossac\1.7.0-arduino3\bossac.exe

[4] Please note that your current installed version might differ from this guide.
[5] Please note that there might be a newer version available. Always check if there is a newer version here: https://github.com/ITACAInnovation/uChip/

Please note that if you downloaded manually BOSSAC 1.7, then bossac.exe will be where you have extracted the archive.

7) In "Arguments" write (including quotes!):
   -d -i -e -w -v -R "$(TargetDir)$(TargetName).bin"
8) Press OK.

Now in the "Tools" menu, you should see "Program uChip" (or whatever you typed in "Title" on step 5).

Go to "**Uploading your code on uChip using Atmel Studio 7**".

If you still want to use BOSSA 1.8 or 1.9 (much slower!)

1) Install the drivers, if you are not using Windows 10.
2) Install BOSSAC 1.9 from here: https://github.com/shumatech/BOSSA/releases/tag/1.9.1 (choose the appropriate 64 or 32 bit version according to your OS).
3) Open the Atmel Studio 7 IDE.
4) For the Menu "Tools" choose "External Tools".
5) Click on "Add".
6) In "Title" write something meaningful, such as "Program uChip".
7) In "Command", locate and choose the "bossac.exe". For instance, in our system we wrote:
   C:\Program Files (x86)\BOSSA\bossac.exe
   Please note that the bossac.exe location might differ on your computer.
8) In "Arguments" write (including quotes!):
   -d -i -e -w -v -o 0x2000 -R "$(TargetDir)$(TargetName).bin"
   WARNING! THESE ARGUMENTS ARE DIFFERENT FROM THOSE REQUIRED BY BOSSA 1.7.0 !!! In particular, notice the option -o 0x2000, which indicates the offset due to the bootloader.
9) Press OK.

Uploading your code on uChip using Atmel Studio 7

1) Build your code.
2) Connect uChip to the USB port (if you did not already do it)
3) Press twice (double click) on the program/reset button on the uChip board (uChip's LED should slowly pulse now from the maximum to the minimum brightness).
4) Select from the menu: Tools→ Program uChip (or whatever you typed in "Title" on step 5 of the previous procedure).
5) You will see that uChip's LED will flash and after that your code will be already running.

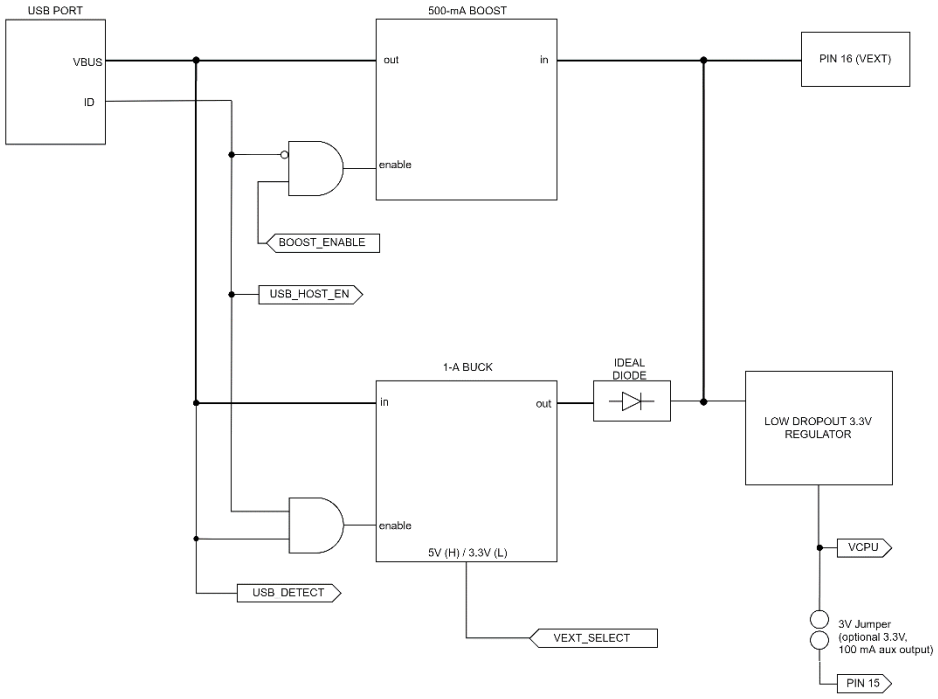
UCHIP POWER SUPPLY BLOCK DIAGRAM

uChip is not just a microcontroller mounted on a board. It contains also a power management circuitry that allows:

1) To output the USB voltage[6] or regulated 3.3V on pin 16, when uChip is USB-powered.
2) To provide a 5.0V USB voltage, when uChip is externally powered (pins 16-8) with a 3.3V to 5V voltage, and an USB OTG cable is connected on the micro-USB port of uChip. This 5.0V can be enabled/disabled by software.
3) To automatically switch between USB and external voltage.
4) To prevent current flowing into the USB port, when uChip is connected to an USB port, and an external voltage is applied between pins 16-8.

The block diagram below shows a general view of uChip. Please refer to the latest schematics for the actual detailed implementation.

uCHIP POWER SUPPLY BLOCK DIAGRAM

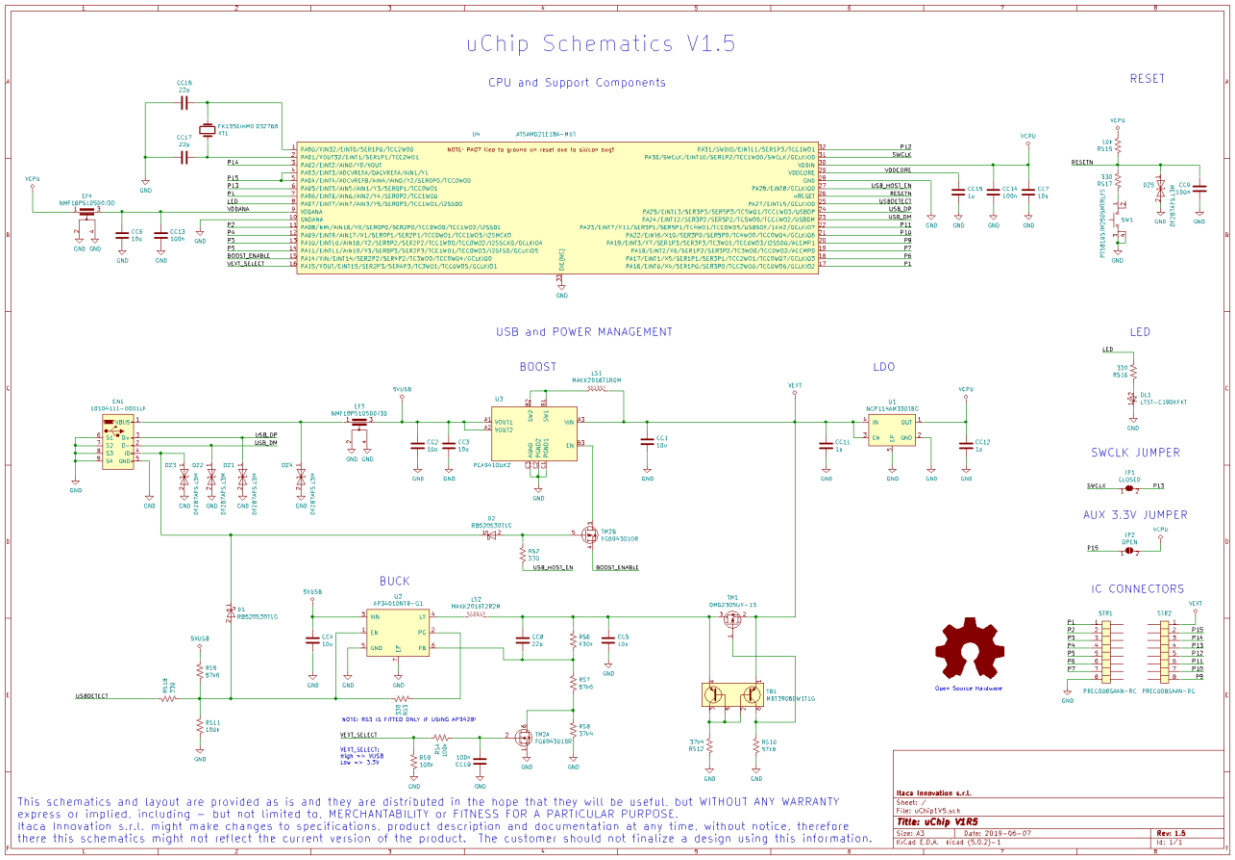


Warning! Although uChip accepts and can provide ad external voltage of up to 5V in pin 16, all other I/O pins are NOT 5V tolerant, and they only accept 3.3V digital or analog levels. **Forcing a voltage higher than 3.3V on I/O pins will irreversibly damage uChip.**

Furthermore, do not apply a voltage higher than 5.5V on pin 16, as it will damage uChip as well. Also, if you need to connect an inductive load (e.g. servo motors) to uChip's VEXT line, see section "Special precautions when driving inductive loads"!

[6] When the USB voltage is chosen, the actual output voltage depends on the USB input voltage, and on the current drawn on pin 16. In this case, the typical output voltage is V(pin 16) = VUSB - 0.5Ω × I, where I is the current drawn on pin 16, in amperes.
When 3.3V output is chosen, the typical relationship is V(pin 16) = 3.3V - 0.15Ω × I, where I is the current drawn on pin 16, in amperes.

SCHEMATICS



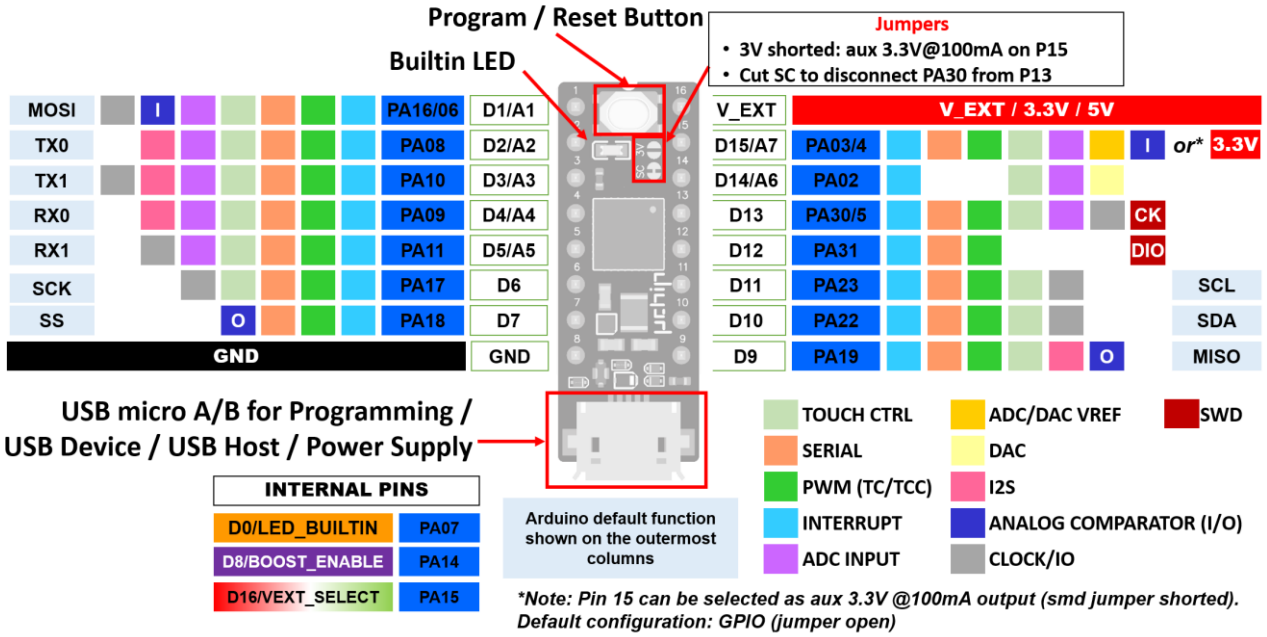
Warning: always check for the latest schematics on itaca-innovation.com. The schematics are provided as-is, in the hope that they will be useful for your purpose, without any warranty (express or implied). Itaca innovation s.r.l. declines any responsibility for any error in the schematics and reserves any right to change specifications at any time without notice.

PINOUT, JUMPERS, LED AND PUSHBUTTON

In the following figure, the simplified pinout is shown.

In particular, from the innermost toward the outermost columns, we show:

1) The Arduino digital/analog pin number (innermost column).
2) The actual port number on the SAMD21 (this is useful especially if you do not use the Arduino Framework).
3) The capabilities of the particular pin.
4) The default Arduino pin function (outermost columns).



Please refer to the ATSAMD21 datasheet for further details on pin functions.

Jumpers

uChip features two jumpers, SC and 3V.

SC Jumper

SC is by default shorted, and it connects PA30 (which is by default the clock signal of SWD) to Pin 13.

Pin 13 is also connected to PA05. If SC is cut, P13 will be still connected to PA05.

Warning! PA30 is pulled up by default. Furthermore, it must be held high during reset, to allow the SAMD21 to run, otherwise it will remain in the reset state. This is due to the SWD implementation on the SAMD21 microcontroller.

If your application does not need SWD debugging and if pin 13 might be low even during reset you can cut the small track that connects the pads of the SC jumper. If you later need to use PA30 or the SWD function, you can just use a drop of tin, to short again the jumper. Please **exercise extreme caution when cutting SC**, as there are other traces close to it!

Warning! Itaca Innovation is not responsible for any damage caused by soldering the jumpers or by cutting the trace.

Warning! Some examples from Itaca Innovation might require pin PA30!

3V Jumper

The 3V jumper allows uChip to provide an auxiliary 3.3V, 100-mA **OUTPUT-ONLY** on Pin 15. By default the 3V jumper is open, so Pin 15 is a regular GPIO.

In this way, uChip can provide both 5V (on pin 16) and 3.3V (on pin 15).

Warning! This output is provided by a 3.3 V linear regulator, connected to VEXT (pin 16). This regulator also provide powers to the SAMD21. Even if the regulator allows up to 300mA, you should draw no more than 100mA, due to thermal constraints.

Built-in LED and reset/program button

uChip features a built-in LED, which can be easily controlled with a single instruction. This can be used for instance as a general purpose indicator.

uChip also includes a dual function pushbutton. A single press will reset uChip. A double click will bring uChip in a bootloader mode. This latter mode is useful when:

- You are coding without using the Arduino Framework (e.g. if you are using Atmel Studio).
- Your Arduino sketch is not responding (typically caused by buffer overflow, disabled interrupts, etc).

POWER MANAGEMENT LIBRARY FOR THE ARDUINO IDE

To facilitate the integration of the power management features of uChip, the board support package includes a small library. To use this library you must add "#include <uChipPowerManagement.h>" to your sketch (without quotes) or select in the menu: Sketch→Include Library→ uChipPowerManagement (you can find it under "Contributed Libraries").

void **uChipEnableBoost**(uint8_**t enable**)

Enables the internal boost, if an OTG cable is inserted (pin ID low).
NOTE: The boost is in any case enabled only when an OTG cable is inserted. At the same time the buck is disabled!

Input Parameters:

enable = UCHIP_BOOST_DISABLED: Boost always disabled, regardless the presence of the OTG cable.
enable = UCHIP_BOOST_ENABLED: Boost is enabled in presence of the OTG cable.

void **uChipSetVextValue**(uint8_**t value**)

Sets the output voltage. This works only if uChip is USB powered!

Input Parameters:

value = VEXT_USB: VEXT (Pin 16) is set to the USB voltage (typically 5V +/- 10%. Note the actual voltage will depend on the output current (see note 6 at page **Errore. Il segnalibro non è definito.**).
value = VEXT_3V3: VEXT (pin 16) is set to 3.3V.

void **uChipSleep**(uint8_t **sleepMode,** uint8_t **sleepOnExit)**

Put uChip in a low power mode.

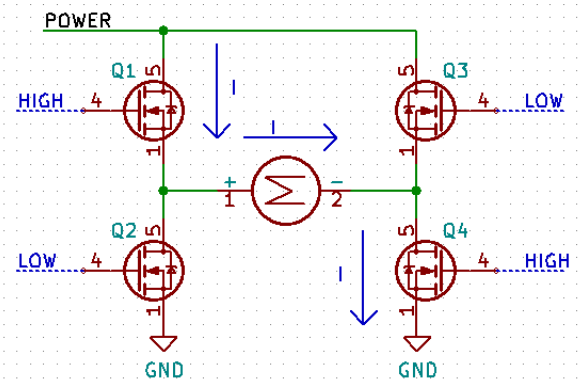
Input Parameters:

sleepMode = SLEEP_IDLE0: The CPU clock domain is stopped
sleepMode = SLEEP_IDLE1: The CPU, and AHB clock domains are stopped
sleepMode = SLEEP_IDLE2: The CPU, AHB and APB clock domains are stopped
sleepMode = SLEEP_DEEP: All clocks are stopped except those which are kept running if requested by a running module or have the ONDEMAND bit set to zero. Furthermore the regulator and the RAM are set in low power mode.
sleepOnExit = SLEEP_ON_EXIT: When an interrupt occurs, it wakes up uChip, and the interrupt routine (ISR) is serviced. After the ISR is complete, uChip automatically goes back to sleep.
sleepOnExit = NO_SLEEP_ON_EXIT: same as SLEEP_ON_EXIT, but when the ISR is complete, uChip does not automatically go into sleep mode again.

SPECIAL PRECAUTIONS WHEN DRIVING INDUCTIVE LOADS

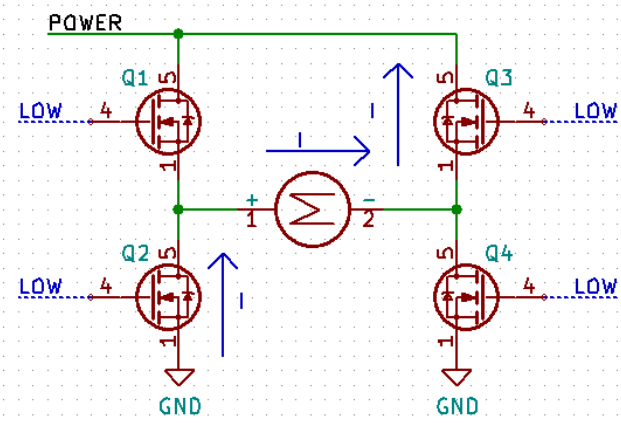
If you need to power an inductive load through uChip's VEXT pin, or even if the inductive load is powered externally, but VEXT is connected to the same power source, pay special attention to the inductive kickback, which might cause VEXT to rise above 5.5V and destroy uChip!

Let us consider this example, where an H bridge drives a DC motor (this is the typical internal drive circuitry of some servomotors).



In the condition above, Q1 and Q4 are in the ON state, while Q2 and Q3 are in the OFF state. Current is flowing from POWER to GND through Q1, the motor and Q4.

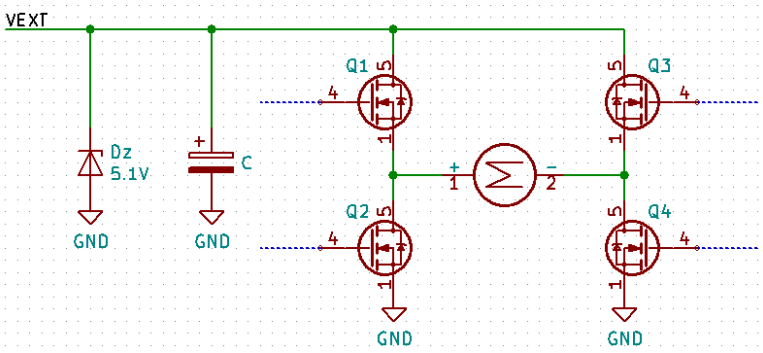
Let us assume that, for any reason, the gate of Q1 and Q4 are now brought at a low voltage. Q1 and Q4 will turn off, but the current on the motor's coil cannot vary instantaneously. The energy stored in the coil must be dissipated, therefore the current will now flow on the body-drain diode of Q2 and Q3, as shown below.



In this condition, current is now flowing INTO the power line. Typically, power regulator do not accept a "negative output current" (i.e. they cannot absorb current/power, they can just output it). uChip's regulator makes no exception, so if there are no other loads[7] connected to the "VEXT" (POWER in the figure above)line, then the voltage on VEXT will rise indefinitely, until breakdown occurs.

To avoid this, we recommend placing a good sized enough 5.1V Zener diode between VEXT and ground, and an electrolytic capacitor, as shown in the following figure.

[7] Or if the loads connected do not consume enough current.



Note! The particular values and ratings depend on the particular application/load.

FAQS

Q: Are the IOs pins 5V-tolerant on uChip?

A: No, **any voltage exceeding 3.3V+10% (3.6V) could irreversibly damage uChip**. Please use adequate 5V to 3.3V conversion circuitry to interface uChip to a 5V system.

Q: Is uChip open source?

A: Yes. The KiCAD project of uChip is available in the download section on shop.itaca-innovation.com!

Q: Can I sell uChip boards based your sources?

A: Yes, provided that you do not use the uChip name and logo, and make a clear statement that you are not affiliated to Itaca Innovation in any way. Furthermore, you should be responsible for any support to your customers for your boards.

Q: Do I have to pay you to produce/sell modified versions of uChip?

A: No, although we will not reject donations as a token of gratitude :)

CONTACT US

For any question, bugs, support, suggestion, etc, contact info@itaca-innovation.com.

Due to the large number of emails, we kindly ask you to be very concise, clear, while being exhaustive in order for us to reproduce your issue.

DOCUMENT REVISIONS

1.2 (27/08/2019):

- First release

1.3 (11/09/2019):

- Added clarification for the integration with Atmes Studio 7 using BOSSAC 1.8 and newer versions.
- Changed driver installation.
- Added footnotes for drivers versions.
- Changed references to the updated 1.0.1 version.

1.4 (07/10/2019)

- Added section “Special Precautions When Driving Inductive Loads”.

1.5 (11/01/2020)

- Corrected pinout: ADC/DAC VREF and DAC had wrong color codes.