Arduino IDE installation and EOD logger operation

William G.R. Crampton, M.A. Haag, and L. Poon (Crampton Lab)

The following instructions have been tested for both Windows 10 and Windows 11

ARDUINO INSTALL

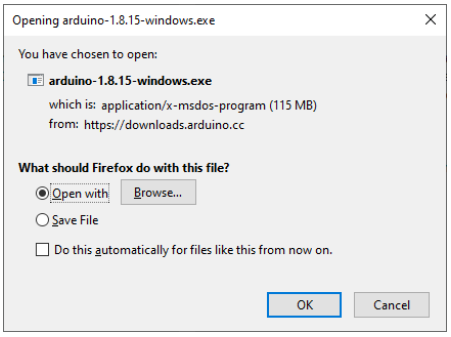
Step 1. Load Arduino IDE 1.8.15:

https://www.arduino.cc/en/software/OldSoftwareReleases

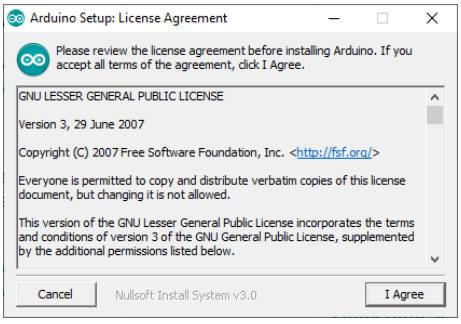
Arduino 1.8.x

These packages are no longer supported by the development team.

Version	Windows	MAC	Linux	Source Code
1.8.18	Windows Windows Installer	MAC OS	Linux 32 Bit Linux 64 Bit Linux ARM 32 Linux ARM 64	Source code on Github
1.8.17	Not released, superseded by 1.8.18.			Source code on Github
1.8.16	Windows Windows Installer	MAC OS	Linux 32 Bit Linux 64 Bit Linux ARM 32 Linux ARM 64	Source code on Github
1.8.15	Windows Windows Installer	MAC OS	Linux 32 Bit Linux 64 Bit Linux ARM 32 Linux ARM 64	Source code on Github



Follow installation instructions and accept the following license agreement:



Install to C:\Program Files (x86)\Arduino

Step 2. Install version 1.58 of Teensyduino software

Note: The tested version of the script was 1.58

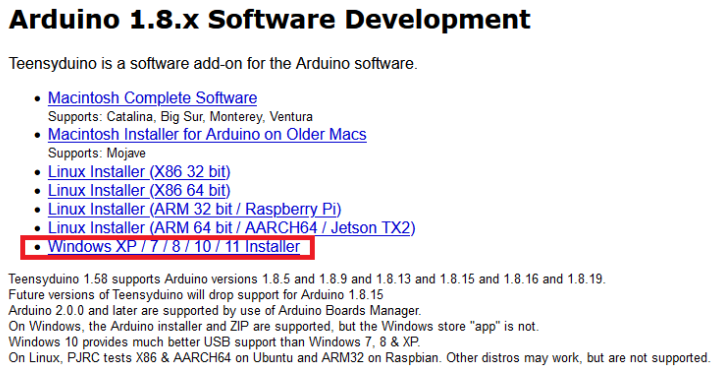
Go to PJRC website: https://www.pjrc.com/teensy/td_download.html

The download will be from Windows XP/7/8/10/11 Installer in section below:

DO NOT INSTALL IT YET !!!!! Read the text below before doing anything.

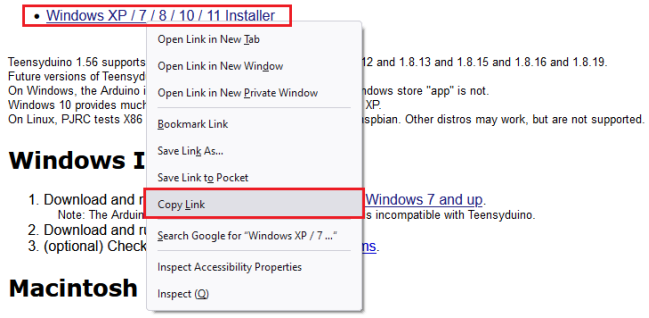
It is essential to load version 1.55 following the exact instructions below. Do not load the latest version of this software or any version after 1.55. Instead, RIGHT CLICK on the link and copy it to the the browser, and edit the link from (change version number from the latest (e.g., _156) to _155). Then press the arrow to right of the web browser window to effect the download.

Note: if a later version of Teensyduino is accidentally loaded, it will be necessary to uninstall the Arduino software, and then re-download and install the Teensyduino software.



Load Teensyduino version1.55:

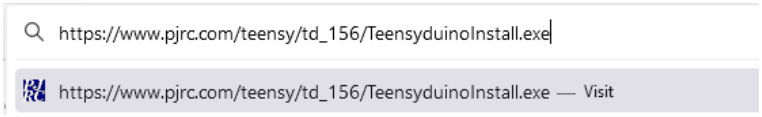
i. Right-click on the Windows XP/7/8/10/11/ Installer and click “copy link



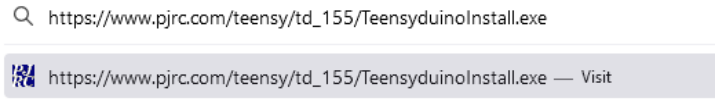
ii. Paste the link into url window of your web browser. DO NOT HIT ENTER YET!

ii. Edit the version number from the latest (e.g., _156) to _155.

e.g.,



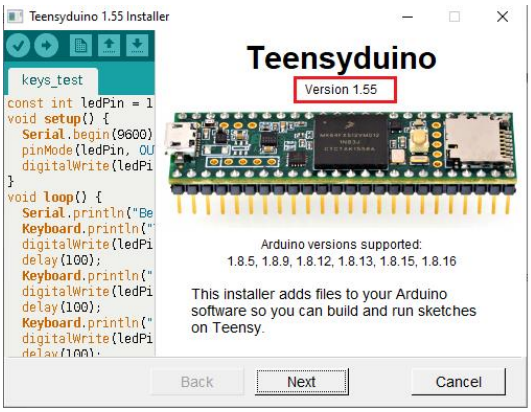
Modify to:



Now press enter.

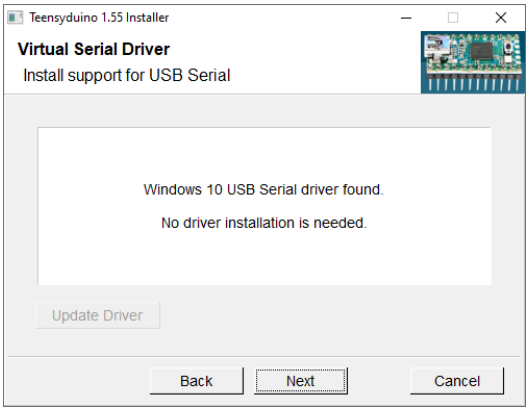
Go to downloaded files in web browser download folder and run the setup file (one can copy this to a hard drive location and run it from this location).

Check that the installer shows version 1.55 when this version is reached:

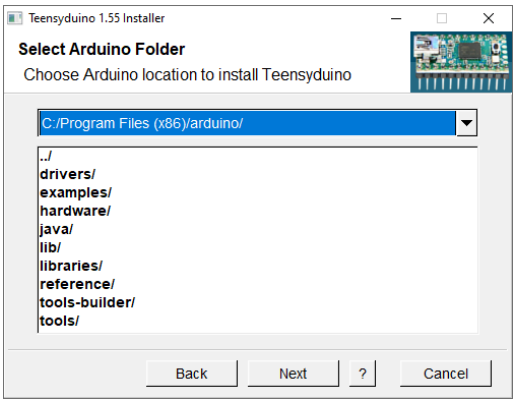


The proceed with the installation. Accept all defaults.

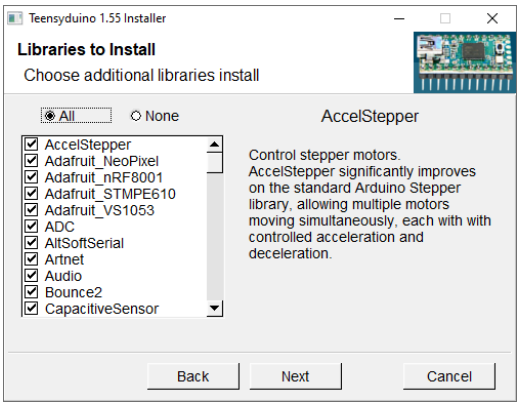
This appears (click Next)



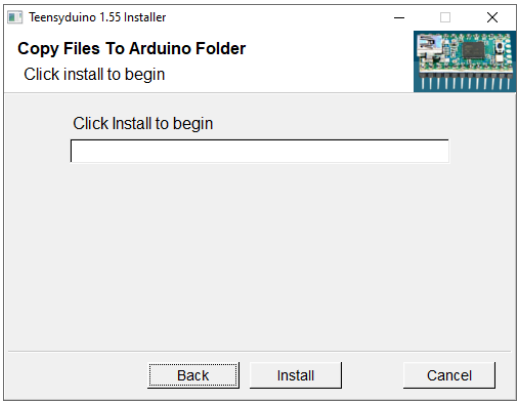
This appears. This is the correct location to install, so click Next:



This appears: click next, leaving all boxes ticked:



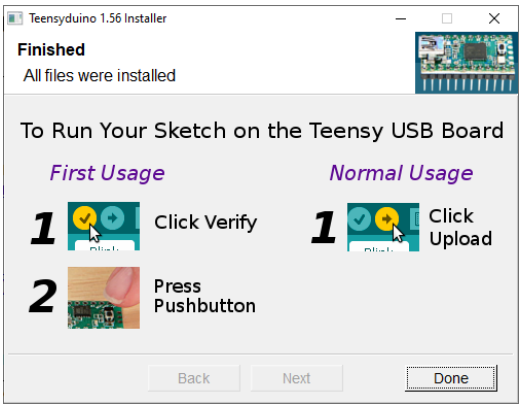
This appears: click Install – yellow bar fills as files installed



Note that Teensyduino was loaded into the same folder as the Arduino software

e.g., in this case: C:/Program Files (x86)/Arduino/

The following message will appear when installation is complete:



When the installation is complete: open the two/four channel version of the microcontroller Arduino sketch developed by Stefan Mucha.

e.g., “Teensy_2_4_channel_1224021”

Note; when you open it, aduino will automatically prompt you to create a folder in the same location that the file is. This has to be done.

At this point we recommend closing Arduino IDE and then relabeling the script to the current date, and also renaming the folder within which the Arduino IDE is contained to the same date: e.g.

Contains:

You **must** keep a backup of the original elsewhere and replace the working file with the backup if there is an unintended change or you want to go back to the original code.

One can check which version of Teensyduino is being used by going to the Help menu in Arduino IDE and clicking on the About Arduino menu item. The windows that pops up reports the current Arduino IDE version in the **top left** and the Teensyduino version in the **top right**.

Step 3. Install Arduino Libraries:

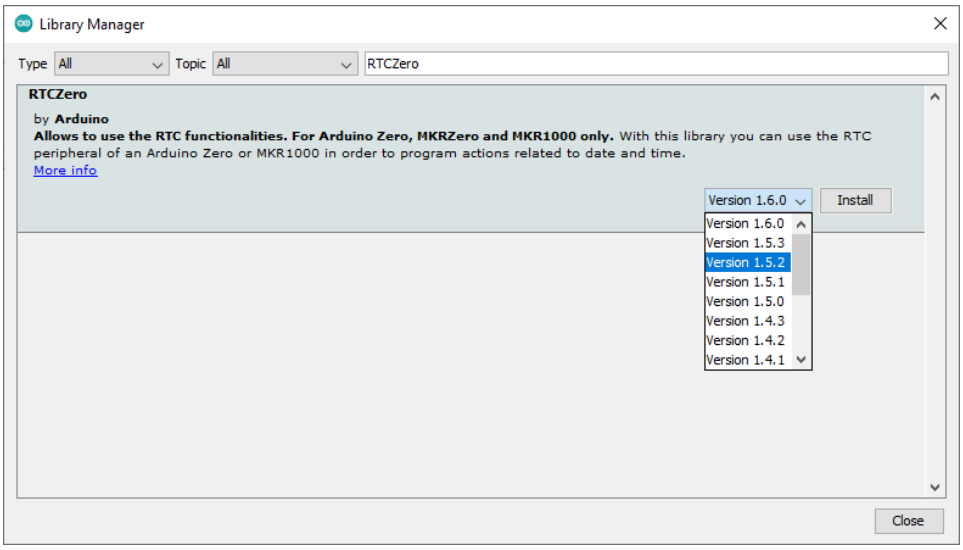
NOTE: these **must be installed from the library manager**. For the relevant library the “More info” link in some , certainly not all, cases will provide a download of the library. In most cases there is additional technical information of a highly specialist nature.

3.1. Open Arduino. In menu to “Sketch” > “include library” > “manage libraries’.

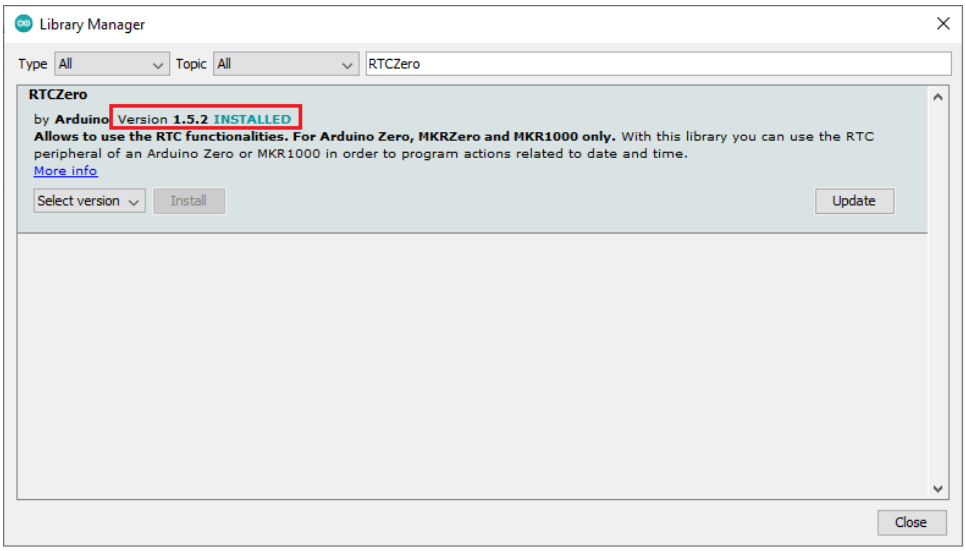
Be patient : it takes a while to load all the libraries.

In the search window search for RTCZero and then choose version 1.5.2.

Click Install to install this version of RTCZero



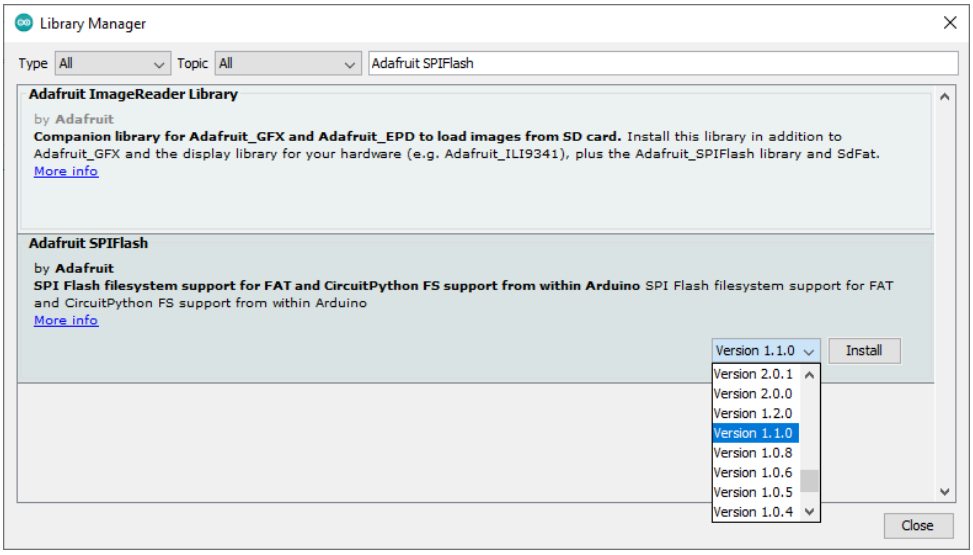
Note. If the library has already been installed it will show the version that is installed, as below – this is the case for all libraries:



3.2. In menu go to “Sketch” > “include library” > “manage libraries’.

In the search window search for “Adafruit SPIFlash” and then choose version 1.1.0

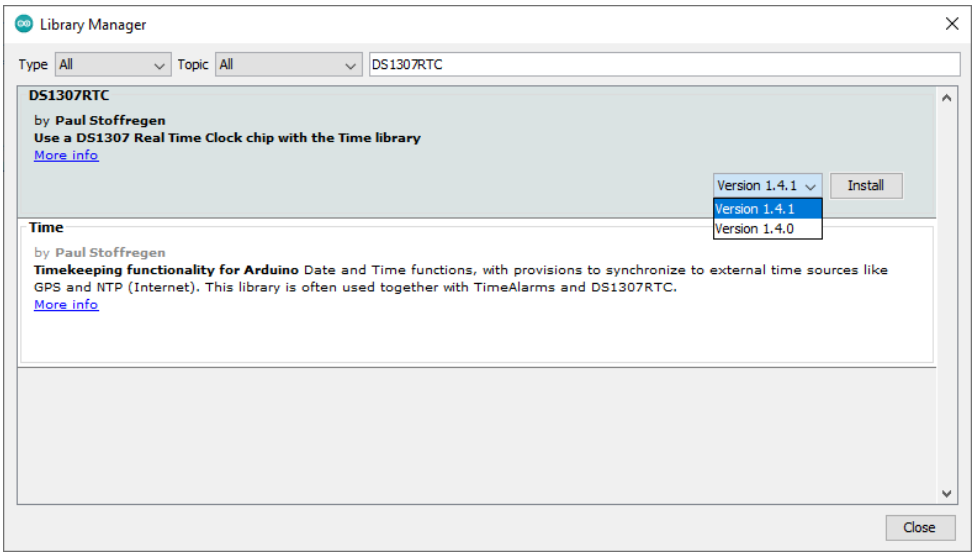
Click Install to install this version of Adafruit SPIFlash



3.3. In menu go to “Sketch” > “include library” > “manage libraries’.

In the search window search for “DS1307RTC” and then choose version 1.4.1

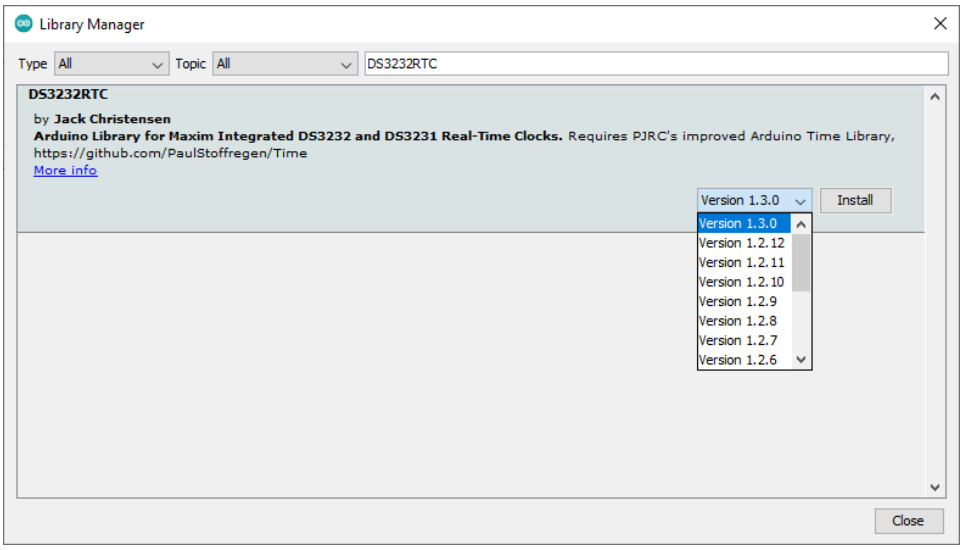
Click Install to install this version of DS1307RTC



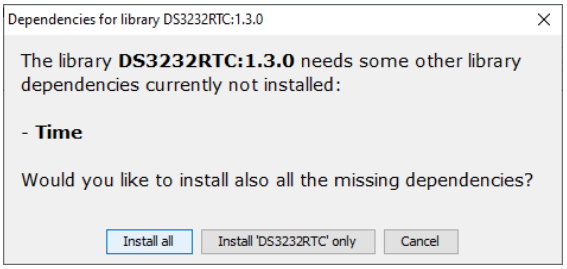
3.4. In menu go to “Sketch” > “include library” > “manage libraries’.

In the search window search for “DS3232RTC” and then choose version 1.3.0

Click Install to install this version of DS3232RTC



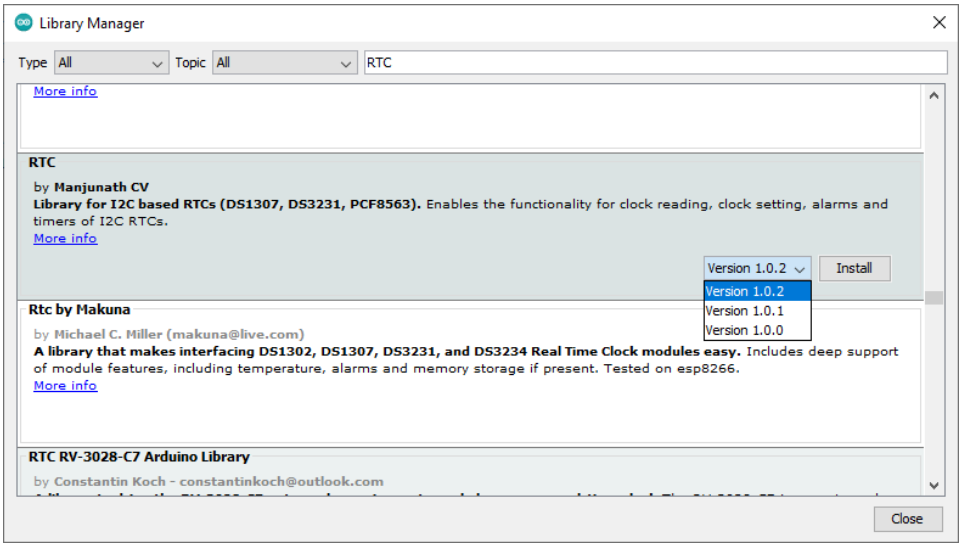
Then select 'Install all"



3.5. In menu go to "Sketch" > "include library" > "manage libraries'.

In the search window search for "RTC" (**here you have to scroll down for a bit!**). Then choose version 1.0.2

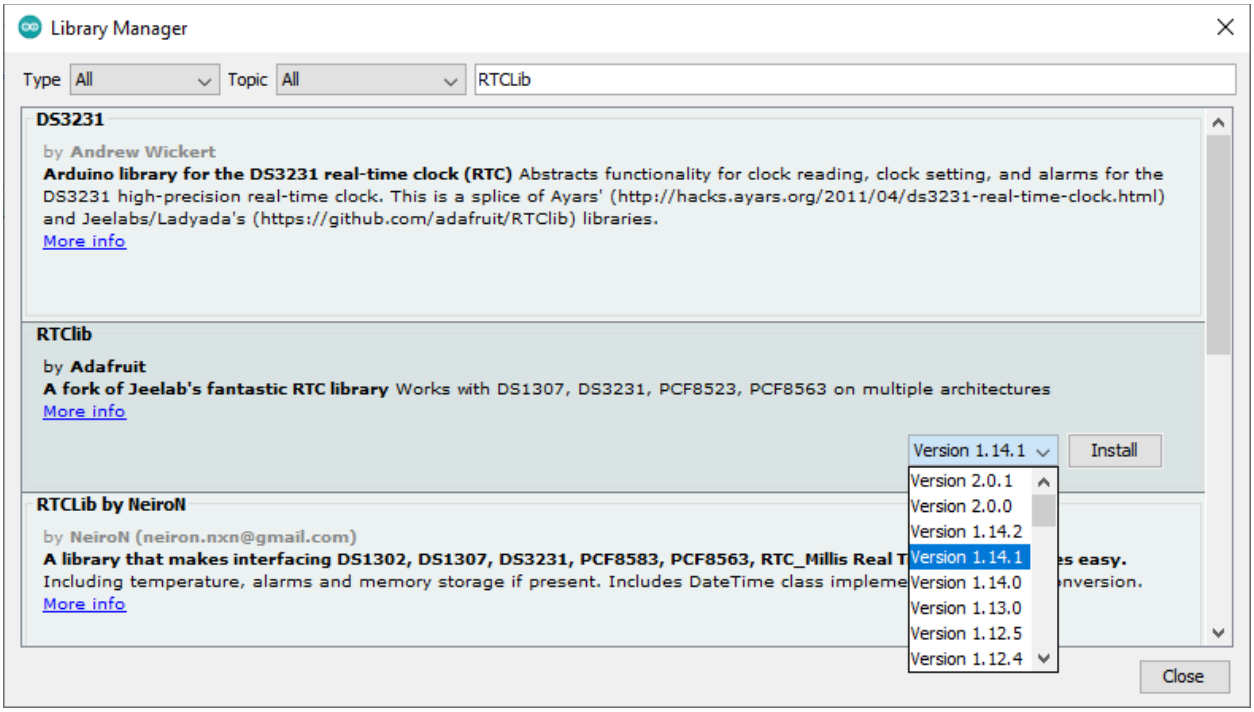
Click Install to install this version of RTC



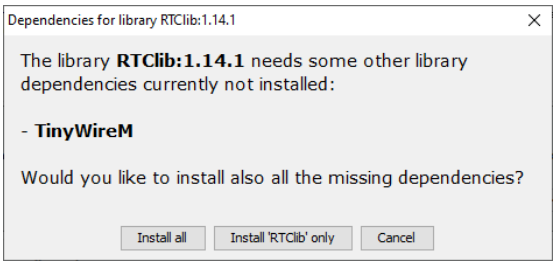
3.6. In menu go to “Sketch” > “include library” > “manage libraries’.

In the search window search for “RTCLib” (once again it is necessary to scroll down a while), and then choose version 1.14.1

Click Install to install this version of RTC



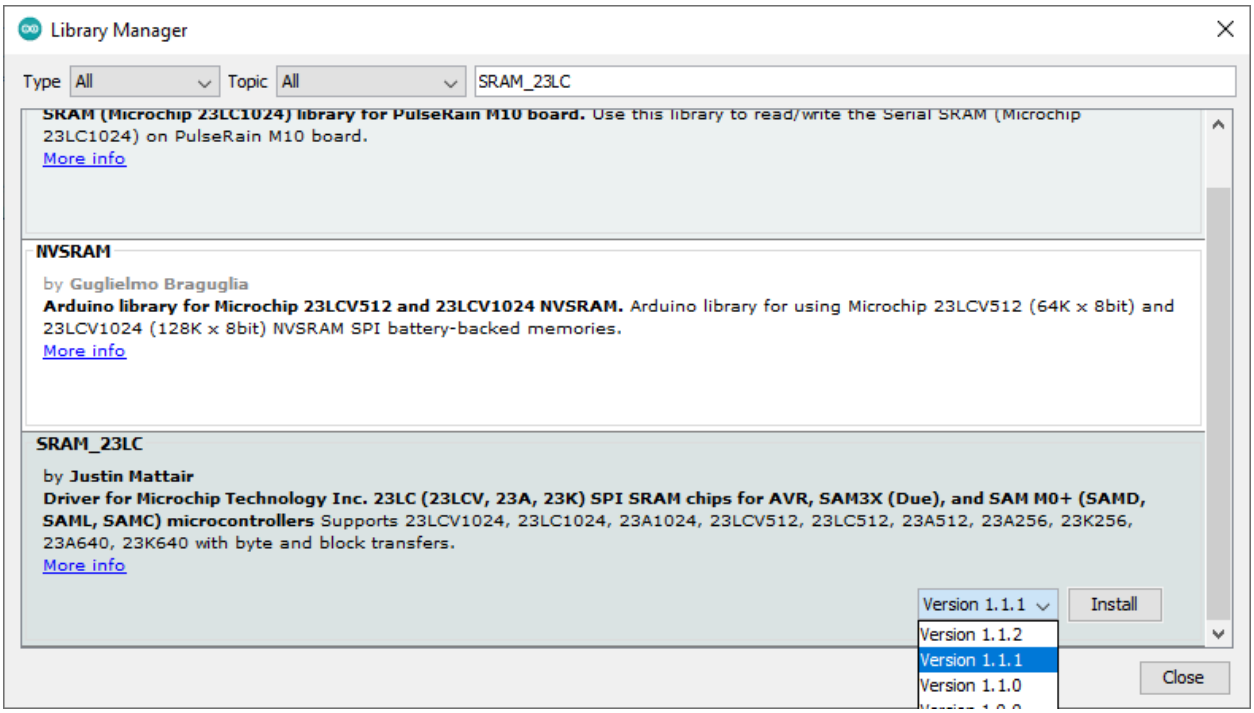
Here click “install all”



3.7. In menu go to “Sketch” > “include library” > “manage libraries’.

In the search window search for “SRAM_23LC” and then choose version 1.1.1

Click Install to install this version of SRAM_23LC

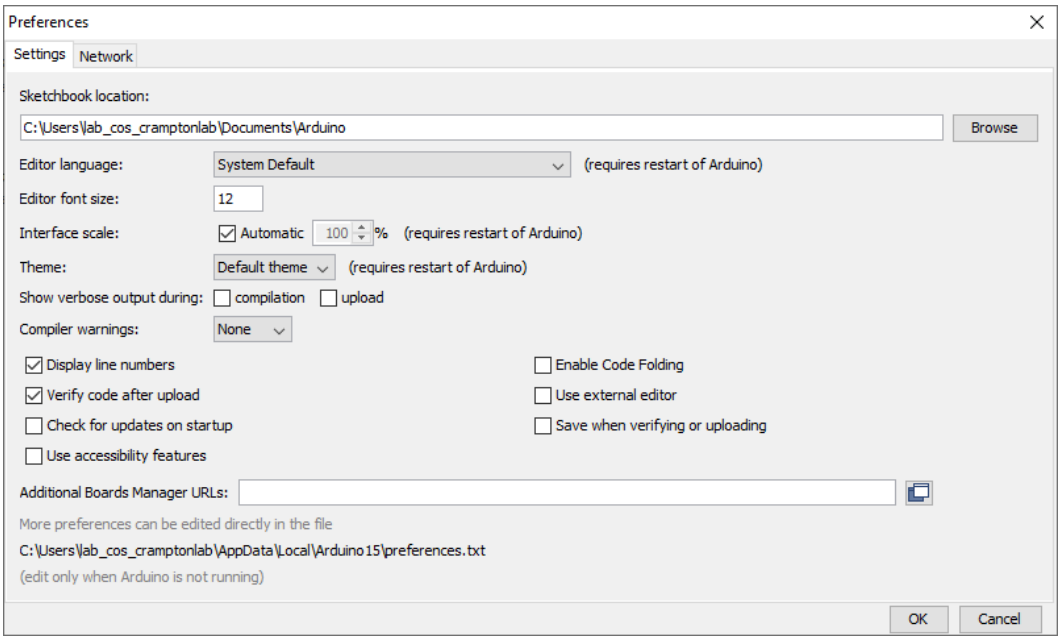


Step 4. Define Arduino preferences.

Go to Files > Preferences and select the options below.

check “display line numbers” and “verify code after upload”; **uncheck** “check for updates on startup”; uncheck “Save when verifying or uploading”. Leave all other options as defaults.

Click “OK” when done.



Step 5. Prepare Arduino for working with the Teensy 3.5.

1. Set up the Teensy 3.5 in Arduino.

Open a microcontroller sketch (e.g., 2 channel, 4 channel, or 8 channel).

Connect the Teensy 3.5 to the computer via the **non-powered** USB cable.

Go to Tools > Board:

If this field does not read Board: Teensy 3.5 (indicating that this device is set up in Arduino)

Go to Board: > Teensyduino > select Teensy 3.5 from the list of Teensy devices

The Tools > Board: field will then read:

Board: “Teensy 3.5”

2. Configure the Teensy 3.5 CPU speed.

Go to Tools > CPU Speed and select from available speeds.

For 2 channel operation at 49,152 Hz sampling rate set CPU speed to 72 MHz. For 4 and 8 channel acquisition ONLY use 120 MHz. For 2-channel acquisition at 98,304 Hz, also use the 120 MHz CPU spped.

Step 6. (with example of 2 channel microcontroller sketch) Configure Teensy_2ch scripts for desired sample rate and file duration.

The user has control over two variables: sample rate (adc_freq) and the file duration in seconds.

Files are created one after another with no gap, so if the file duration is 60s, the recording interval is also 60s.

The file duration in seconds is calculated by the following formula:

Recording duration = buffer_dimension * times_buffer / (adc_freq * 2)

Here:

buffer_dimension is a hard-coded variable on line 14

times_buffer is the buffer size defined by user at line 106

```
106 const int     times_buffer    = 360;
```

Adc_freq is the sampling rate defined by user at line 69. The sampling rate will always be equal on the two channels (A and B).

```
69 const uint32_t adc_freq = 49152;
```

The denominator value of **2** is used because there are two channels.

Rules:

1. The adc_freq must be a whole number above twice the highest frequency content in gymnotiform signals, which is around 36 kHz.

2. The times_buffer must be a whole number

3. The file duration must be a whole number divisible into 60 (i.e., 1, 2, 3, 4, 5, 6, 10, 15, 20, 30, 60).

At a given sample rate, the recording duration can be calculated using the formula above, by changing the times_buffer.

For example, at a sampling rate of 49152, if the times_buffer is set to 360, the recording duration = 60s. If the times_buffer is set to 90, the recording duration = 15s.

Use the Excel spreadsheet **sample-rate-calculator.xlsx** to calculate other sample rates.

We recommend 60 second recordings at 49152 Hz, where times_buffer is set to 360 for greatest stability.

Some common options are:

adc_freq = **49152**, times_buffer = **90**, recording duration = **15s**

adc_freq = **98304**, times_buffer = **720**, recording duration = **60s**

adc_freq = **36864**, times_buffer = **270**, recording duration = **60s**

Part 2 - Operation of EOD Loggers

Step 1. Prepare the EOD Logger

1. Install the fully charged NiMh batteries in the EOD Logger power pack. We recommend Eneloop Pro batteries. Confirm full charge for each battery with a multimeter. Do not use any under-charged batteries!

2. A CR-1220 lithium coin battery does not need to be added to the battery slot in the DS3231 RTC circuit board. The RTC draws its power from the Teensy and will lose its time if the power to the EOD logger is disconnected. Adding a coin battery does not mitigate this effect (given the way the Arduino code is written) and is therefore redundant.

3. Connect micro-USB end of a **non-powered** micro-USB-USB cable into the Teensy micro-USB port. See below for preparation of a non-powered micro-USB cable.

4. Insert a compatible micro-SD card into the Teensy drive bay (see Appendix 1 – SD cards).

5. Check all connections are firm. Check batteries are not loose in bay (we recommend elastic bands to secure them).

Step 2. Prepare the Arduino program

1. Connect the EOD Logger to the host computer after the EOD Logger is prepared as described in Part 1. (avoid using a USB Hub – instead plug into one of the computer's available USB ports).

Note: the battery pack can be connected at this point. However, if the logger had not completed its previous recording task (i.e., the specified number of files were not recorded) it will resume that task leading to potentially unwanted files on the microSD drive). We recommend connecting the battery pack later. Be sure to never have the logger in a completely unpowered state (i.e., not connected to the USB cable [which itself must be connected to the computer] or to the battery pack).

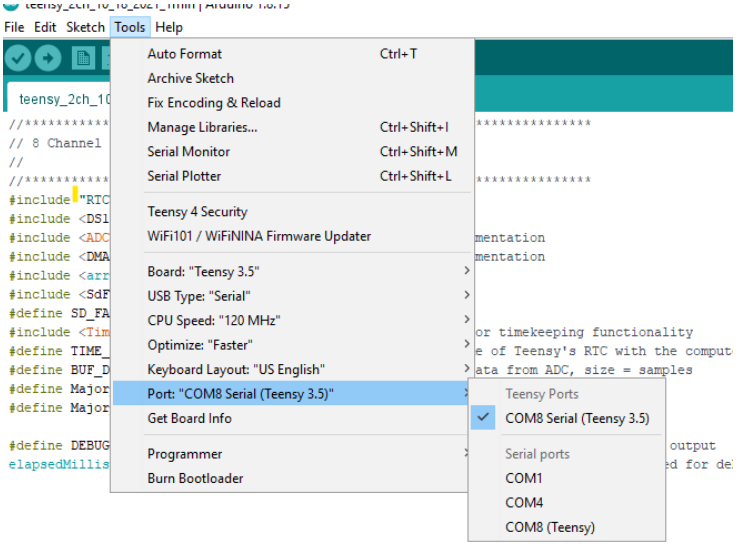
2. Open the Arduino program (e.g. "Teensy_2_4_ch_12242021") in Arduino (in the folder that is automatically created earlier)

Do a final check that: 1) the microSD card is installed; 2) the USB port is connected to both the Teensy and the host computer.

3. Go to "Tools" > "Board"* > "Teensyduo" > Teensy 3.5

* when configured this part of the Tools menu will read "Board: "Teensy 3.5""

4. Be sure that the right USB port is selected: as follows. Make sure you select the one with Teensy 3.5 is selected



When correctly set it will look like this in the Tools menu:

"Port: COM3 Serial (Teensy 3.5) (it may be another COM channel instead of Com3)

Note: If there are any problems with establishing the com channel for the USB it may be necessary to press the button on the Teensy

The button on the teensy will disconnect and reconnect the connection channel.

5. Change the Teensy CPU speed if desired.

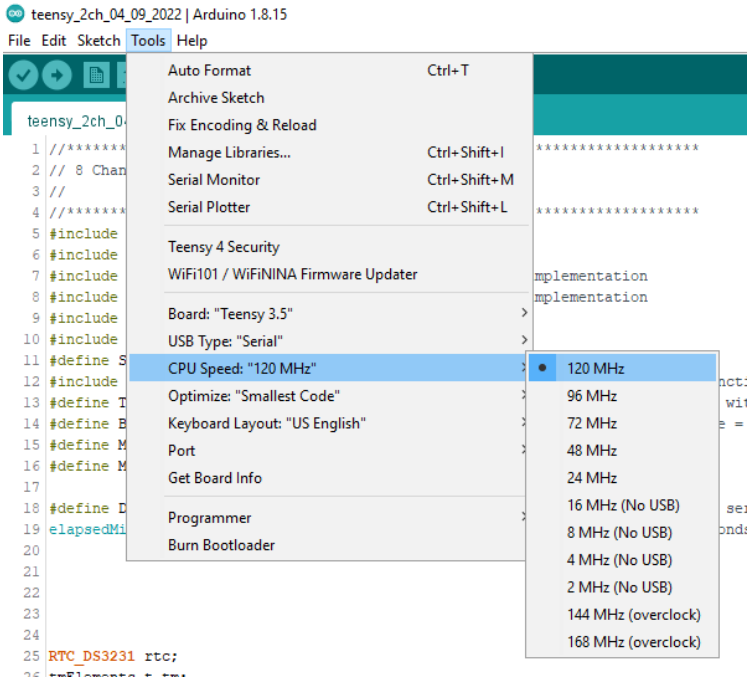
For two channel version of microcontroller Arduino sketch:

Battery power can be saved by lowering the CPU speed on the Teensy 3.5. We have tested the device at 72 MHZ and 48 MHZ. Lower speeds are not compatible. Going to lower CPU speeds appears to result in very slightly less resolution to the PPF of the waveforms at frequencies exceeding 10 kHZ, but the waveforms are indistinguishable from those at higher clock speeds, and indistinguishable from waveforms recorded with the SR-5113 amplifier.

To change to 72 MHZ: Go to: Tools > CPU > 72 MHZ.

For four and eight channel versions of microcontroller Arduino sketch:

The CPU must be set at its default of 120 MHZ. Do **not** lower the speed. It will cause failures.



Note: ***The line numbers for code in the example below may not be precisely representative!***

6. Program the launch time (and year) in the teensy_2ch script

i. Change the year if necessary (if not previously saved). For example, for 2022:

At line 29 make sure text reads y = 22:

```
25 RTC_DS3231 rtc;
26 tmElements_t tm;
27 uint8_t d = 0;
28 uint8_t mo = 0;
29 uint8_t y = 22;
```

(save Arduino file after making a major change like this using: File Save or Ctrl S).

ii. Change start time in code section ‘start time vars” Lines 50-53.

Start time variables: these times are intrinsic (i.e., the desired start time) in 24 hour time:

e.g., 3 or 03 for 0300, 23 for 2300.

So. If you were to launch at 4.45pm for a 4.50 pm start:

The following settings would yield 16:50:00

```
50 ////////////start time vars/////////////////////
51 byte starthourTimer = 16;
52 byte startminuteTimer = 50;
53 byte startsecondTimer = 0;
54 ////////////////////////////////////////////////
```

The following setting would yield 03:05:00

```
50 ////////////start time vars/////////////////////
51 byte starthourTimer = 3;
52 byte startminuteTimer = 5;
53 byte startsecondTimer = 0;
54 ////////////////////////////////////////////////
```

Hours and Minutes: numbers under 10 can be reported as e.g., 03 or 3

Seconds: we recommend leaving this as 0.

iii. Define n. files written at line 57.

If the recording is for 24 hours, and recordings are minute long, make sure that at least 1,440 files are specified.

A good number in this case would be 3,000:

```
56 ////////////end time vars/////////////////////
57 uint16_t filesSaved = 3000;
58 ////////////////////////////////////////////////
```

Note that if the Teensy 3.5 logger does not finish all specified recordings and the SD card is removed, when the card is replaced, the Teensy 3.5 will continue to record files, thus depleting the battery. In these cases, launch the device soon after it is powered on and be prepared to remove the excess unwanted files from the beginning of the sequence of recordings.

The first file and the last file will typically fail. Thus start acquisition a few minutes before and after the required recording time to ensure that no files are missing. Minimally, request 2 more files than are required. The first and last will be blank or won't open. This is a bug that cannot be resolved.

Step 4: Launch the logger

Be sure the non-power USB cable connects the computer to the Teensy 3.5. Be sure that the Teensy 3.5 is powered on with 3.7-6.0 V DC. Be sure that the SD card is in the card bay. Be very careful not to use a powered USB cable when the Teensy is connected to DC power.

The Teensy instructions provided by the PJRC website are that when a logger is launched for the first time (or the Arduino softwrae loaded for the first time).

1. Pres tick in far left (white tick below). The little Teensyduino window will open.

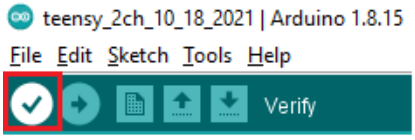
2. Depress the little white button on the Teensy 3.5 and force a manual reboot. The Teensyduino window will report that this has happened.

3. Then tick the right-pointing arrow to the immediate right of the white tick.

This will load the sketch. There will be a double chime. The USB cable should then be dettached from the Teensy.

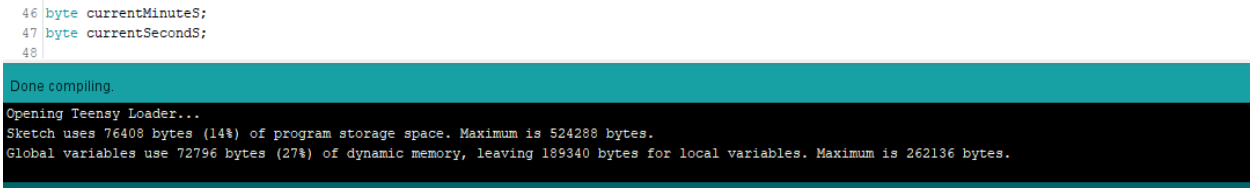
At all other times. All that is required to upload the sketch is to press the right-pointing arrow.

1. Click “verify” , the upper left check (white tick) sign button.

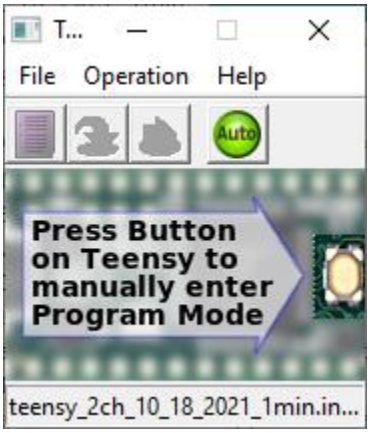


You will see “Compiling Sketch” in the bottom left, and a progress bar in the bottom right. This will take a while.

When successfully completed “Done compiling “ will appear and the text below it will be white on black background (if unsuccessful text will have an orange background)

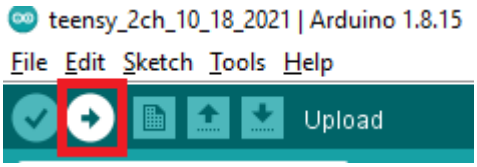


More importantly: this also appears:



This window needs to stay open. Reduce it but don't close it down.

2. Enter Ctrl + U to upload code (or press the right arrow icon to the right of the Verify (tick icon) in top left of screen). Do this at least 3 min before the programmed start time.



Wait until it says 'Done uploading' at bottom. There may (or not) be a chime.

3. Connect the battery pack at this point. Make sure white battery pack is firmly pressed in.

4. Pull USB cord from the Teensy device. Caution: check the battery pack is connected before the USB cord is pulled!

Caution! If the device is at any point completely unpowered then the RTC will lose its time signal and the time values of the recordings will not match the reported time stamps in the file names of the recordings!

The Logger should start blinking around 1.5 minutes after the defined start time. We recommend watching it for 5 min before concluding that the launch has been successful.

Note: although the logger can be launched at a point I the more distant future (e.g., a few hours ahead). This has the advantage of not depleting the battery or filling the SD card. However, the LED will not flash to inform user (reassuringly) that the device is operating.

3. Launch next device if required or close Arduino software.

Closing Arduino software. Click X in top right of software. When closing we recommend selecting "No" when the following message appears : Save changes to "teensy_2ch_xx_1min" (or current version). It is better to save this program only when a specific change is made that the user knows will be required in the future. Be sure to always have a backup of the original version of Teensy_2ch_0.

Close the Teensy software.

Step 5. Retrieving data from the EOD Logger.

1. Power off device by disconnecting the power pack (or removing batteries from the power pack).

2. Pull microSD card out

3. Use microSD to USB card reader to copy data from drive. Delete all data on drive when this operation is complete.

4. Recharge the batteries to full capacity.

5. Inspect EOD Logger (make sure there is no water infiltration to the waterproof enclosure).

6. Store EOD Logger in air-tight container with silica gel when not in operation.

Notes on USB cable when launching multiple EOD Loggers.

When launching multiple loggers we recommend always inserting the USB-microUSB cable in the same USB port on the host computer. If the USB port used to connect to the Teensy via the USB-microUSB cable is changed, the user must go to Tools > Port and select the port connected to the Teensy.

Appendix Powering the EOD Logger.

The Teensy requires 3.3V and the Dual amp board also requires 3.3V, which it draws from the Teensy.

The Teensy operates to a MAXIMUM allowable voltage of 6V.

Power pack.

See section “Battery power”: front page of: https://github.com/Crampton-Lab/eFish-Field-Logger

Caution! Never use battery packs other than the custom-built ones because the voltages may exceed the capacity of the Teensy 3.5 device!

Caution! Never power EOD Logger with USB power banks!

Preparing a non-powered USB to micro-USB cable

The PJRC website instructions for the Teensy 3.5 state that the USB to micro-B cable used to launch Arduino code on the Teensy 3.5 should **not** be powered if the microcontroller is powered at VIN (see “VIN Pin” at https://www.pjrc.com/store/teensy35.html). Since the loggers are typically connected to a battery pack at the time of launching, a non-powered USB to micro-B cable should be prepared, as follows:

Slice through the outer plastic sheath carefully with a scalpel blade, down to the braided metal shield. Be careful not to cut through the metal shield. Prize the shield apart and tease out the red power wire. Cut out a section about 0.5 cm long a below.

Heat shrink the ends of the severed wire to ensure that power does not contact the metal shield. Wrap the cable up with insulating tape and label “No Power!” so that this label is visible when the micro-B plug is inserted into the Teensy 3.5. This will ensure that the user is always able to confirm that a non-powered cable is used.