PULSE system

user manual v1.8, updated on 28/10/2019

BASIS OF OPERATION

The sensors typically used with the Infra-Red (IR) method of heart rate measurement combine an IR emitter and an IR detector in a small package (see the sensor section further below). Fixing the sensor to the exoskeleton of an animal, above its heart, allows IR light to pass through the shell of the animal and illuminate the heart and nearby circulatory vessels. Changes in the shape or volume of the circulatory structures during a heart contraction, or heartbeat, cause a change in the amount of IR light reflected from the animal's internal anatomy back to the IR detector. These changes in reflected IR light, transduced to changes in electrical current, are then processed by the electronic circuit and by the software (amplified, filtered, etc). For further information see the following paper. It describes an older version of this system that is not stand-alone (requires an oscilloscope, a PC and user supervision and control) and can only monitor one individual at time. Besides that, everything is similar to the current system. The paper also describes some common problems and how to solve them.

Burnett, Nicholas P., et al. "An improved noninvasive method for measuring heartbeat of intertidal animals." Limnology and Oceanography: Methods 11.2 (2013): 91-100

There are already some papers describing how the PULSE system was used to acquire a huge volume of heart beat data which was posteriorly processed and interpreted in a wider context. See for example:

Seabra, Rui, et al. "Equatorial range limits of an intertidal ectotherm are more linked to water than air temperature." Global Change Biology 22.10 (2016): 3320-3331

Gurr, Samuel J., et al. "Cardiac responses of the bay scallop Argopecten irradians to diel-cycling hypoxia." Journal of Experimental Marine Biology and Ecology 500 (2018): 18-29

HARDWARE CHECK

The PULSE system (Fig. 1) is composed of:

1. A white box with eight connections on the top.
2. An SD memory card.
3. Up to 8 cables with heartbeat sensors.

If you are using the PULSE system indoors, you may want to plug it to the mains electricity. In that case you will need a 7-28 V AC/DC wall adapter (a regular 9 V will work well). You can also power it with 5V through the USB port. In alternative, if you are deploying the PULSE in the field, you should use a 9V battery or a 6 x 1.5V pack. You may use any FAT formatted SD card (2 or 4 GB is more than enough).

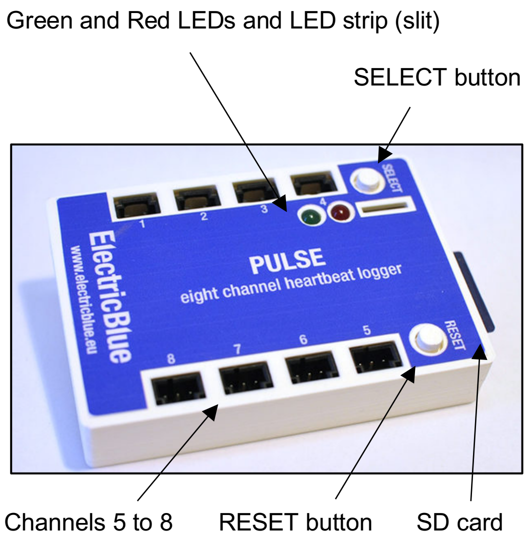


Fig. 1. Overall view of the system.
Sensors are not shown.

SOFTWARE/FIRMWARE INSTALLATION

[You do not need to do this unless you want to update the PULSE built-in clock or want to modify the firmware]

1. First you will need to download and install the following software packages in a PC:
   1.1. Arduino IDE (`http://arduino.cc/en/Main/Software`).
   1.2. Processing IDE (`https://processing.org/download/?processing`).
   1.3. R (`http://www.r-project.org`).

2. After the software installation, the following libraries/packages must also be installed:
   2.1. For Arduino IDE:
      2.1.1. Go to `https://goo.gl/E53hmh`, navigate to the folder **`/software/`** and copy the **`Libraries`** folder. It has 2 libraries inside, RTClib and SdFat (more recent versions may exist, but these will work).
      2.1.2. Paste the **`Libraries`** folder into your **`Documents/Arduino/`** folder (MacOS) or **`My Documents/Arduino/`** folder (Windows).
      2.1.3. Restart the Arduino IDE.

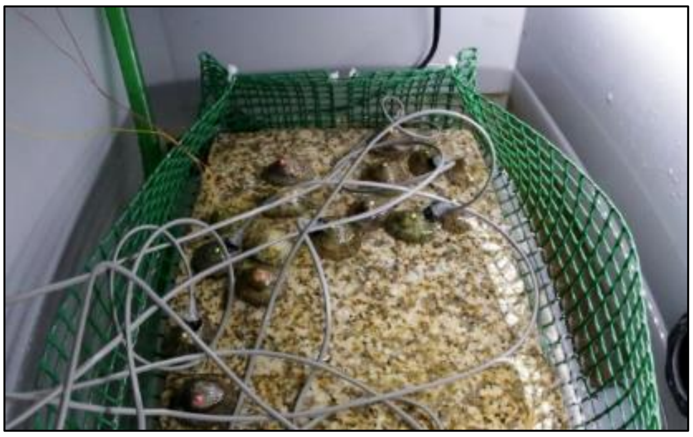
CLOCK SYNCING AND LOGGER PROGRAMMING

[You do not need to do this unless you want to update the time in the PULSE built-in clock]

1. Update the PC clock and check that it is set to the appropriate time zone.
2. Connect the Arduino to the PC via the USB cable.
3. In the PC browse to **`/software/start_clock/`** and open **`start_clock.ino`** using Arduino IDE.
4. From the top bar select **`FILE > UPLOAD`**.
5. With a successful upload the logger's clock will be synced.
6. The clock is quite stable and will not need to be re-synced for many months.
7. After uploading this sketch, you CANNOT open the serial communication. That will mess with the clock settings. If you want to check if the clock I running properly, you should do this:
   7.1. In the PC browse to **`/software/check_clock/`** and open **`check_clock.ino`** using Arduino IDE.
   7.2. From the top bar select **`FILE > UPLOAD`**.
   7.3. Start the serial monitor and check if the clock is running and up to date (a few seconds out of sync is normal and expected).
8. To get the Heart-beat logging program back into the logger, in the PC browse to **`/software/logger_hb_45_7/logger_hb_45/`** and open **`logger_hb_45.ino`** using Arduino IDE.
9. From the top bar select **`FILE > UPLOAD`**.
10. The logger is now ready to be used.

SENSOR PLACEMENT

1. The sensors should be glued with superglue to the shell or carapace of the studied animal.
2. Attaching the sensor right on top of the heart is probably the most important single aspect in getting a good signal.
3. Study the anatomy of your species and try to locate the approximate position of the heart.
4. If you are using a brand-new sensor you should consider shaving off the outer plastic edge with sand paper or a Dremell abrasive tool in order to obtain a flat surface – see the SENSOR ASSEMBLY AND REPLACEMENT section below.
5. Clean the shell or carapace with a cloth. Superglue does not work so well with moisture.
6. Put a thin layer of superglue on the sensor.
7. Attach the sensor to the shell. Hold in place for some seconds. Be careful not to use too much superglue, or it may fall on the animal while still liquid.
8. The attachment should be strong and the sensor should lay flat against the specimen.
9. Sensors can be detached by breaking the superglue bind. If they are going to be reattached, shave off the superglue from the surface and repeat the previous steps.



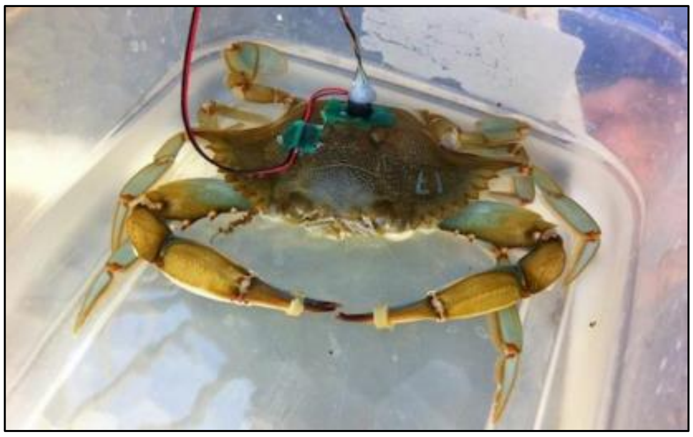


Fig. 2. Examples of sensors attached to limpets (left) and to a crab (right).
The crab has a conductance sensor as well

SYSTEM TESTING AND SET-UP

1. Disconnect the heartbeat logger from any power source (USB, batteries or wall adapter).
2. Connect up to 8 sensors to the top board of the heartbeat logger. The eight sensors should be labelled sequentially from 1 to 8.
3. Insert the SD card in a PC. Copy the settings file in `/software/logger_hb_45_7/hbsett.txt` to the SD card. A template of the settings files can also be found at the end of this manual.
4. Edit the settings file:
   4.1. Choose an **ID** for the logger (`logger_xyz` in the example file). This id is the device name, and will be associated to the files logged in that particular system so when you have several systems working simultaneously you know where the logs came from.
   4.2. **SENSOR_MAX** is the maximum number of sensors you are using. If you are using three sensors, **SENSOR_MAX** will be *3* (i.e, the logger will read the signal from sensors 1, 2 and 3).
   4.3. The variable **rateHB** defines the frequency (in milliseconds) at which the HB signal will be processed. If the number is too low, then data will lose quality. If it is too high, the system will not be able to process everything. Our own experience says that a value of 15-20 is ideal.
   4.4. The variable **READ** sets the number of seconds each sensor will be turned on, resulting in the corresponding channel being logged. *60* means that each animal will be logged for 1 minute.
   4.5. The variable **SESSION** defines the sampling frequency in minutes. On each sampling session the logger will cycle once through the sensors defined in **SENSOR_MAX**, will log each one for the period defined in **READ** and then sleep until it is time to start again. After the X minutes defined in **SESSION** have elapsed a new cycle begins. For example, 15 means all channels will be logged every 15 minutes.
   4.6. The variables **S1** through to **S8** can be used to assign ids to each channel of the device. The respective channel id will be stored in each log file.
5. Place the SD card back in the heartbeat logger SD card slot.
6. Connect the logger to the PC via USB.
7. In the PC open Processing IDE.
8. Choose `FILE > OPEN`, go to the folder `/software/logger_hb_45_7/real_time_hb_7/` and open the `real_time_hb_7.pde`.
9. Hold sensor 1 (the first one) against your thumb to measure your heartbeat (see Fig. 3).

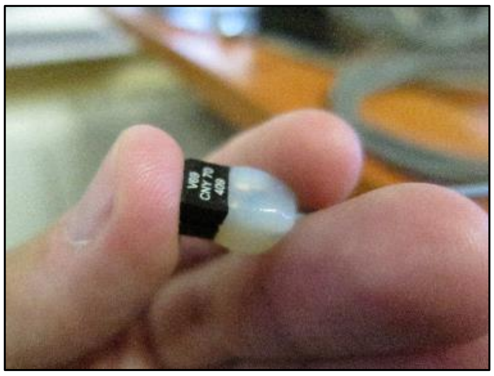


Fig. 3. Testing the system by directing the CNY70 towards the thumb to capture the operator's heartbeat. Caution must be taken not to press too lightly (which will break the close contact between the sensor and the finger), nor too hard (which will decrease circulation and make it harder to see the heartbeat).

10. From the top bar select `SKETCH > RUN`. At this point you will either see:
    10.1. A window popping up with a signal wave showing your heartbeat - in this case everything worked fine. At this point data is being saved in the SD card as well. Go to step 11.
    10.2. An error message in the bottom of the Processing window or a window popping up but without the 'correct' signal wave - this means the COM port was not properly set, in which case you will need to follow these steps:
        10.2.1. Scroll through the error message and look for the number for the correct COM port.
        10.2.2. Look for the following line in the code `port = new Serial(this, Serial.list()[7], 9600)` and edit the number inside square brackets ([7] in this example) to match the COM port number to which the heartbeat logger is connected.
        10.2.3. If you cannot find the correct port number in 8.2.1 just try numbers sequentially from 0 until the error message disappears AND the window pops up AND the expected wave signal is displayed. Go to step 11.
11. After some seconds (defined in the variable **READ** in hbsett.txt file in the SD card) the sensor being read will automatically change to sensor 2, then to sensor 3, etc.… In this testing phase you may want to do things slower, so you can:
    11.1. Continuously press the SELECT button (see Fig. 4) for more than 3 seconds. This will make sensor switching manual instead of automatic. Thus, the logger will keep on the current sensor and will not change until you press the button again. Please be aware that in this mode data is saved to a different folder in the SD card (folder "M", as in manual mode).
    11.2. To change the sensor being read, click (less than 3 seconds) the SELECT button.
    11.3. Keep clicking the SELECT button to cycle through all sensors (but be aware of **SENSOR_MAX** in the settings file).
    11.4. To return to automatic cycling, press the SELECT button for more than 3 seconds. In the automatic cycling, data will be saved to the SD card inside folder "A", as in automatic mode.
12. Make sure the PULSE is placed in a secure location and will not need to be moved during the entire data collection.

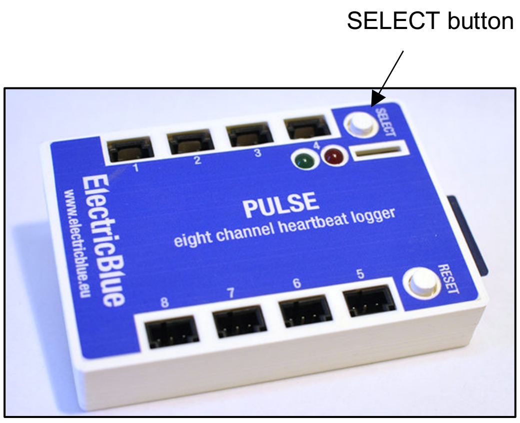


Fig. 4. The SELECT button is used to change between reading modes (long press changes between automatic and manual), and to cycle through the various sensors if the manual mode has been selected.

DATA COLLECTION

1. For regular data collection, connect the heartbeat logger to a power source (USB, batteries or wall adapter) and data collection will start automatically within a few seconds.
2. Important: you can connect the heartbeat logger to a PC via USB (see above **SYSTEM TESTING AND SET-UP**) and use the real time visualization tool to check the HB at any time, but be aware that every time you plug/unplug the device the operation is restarted, meaning that any ongoing collection of data is interrupted.
3. As explained above, the logger will automatically cycle through sensors. You can always stop that and enter manual mode by long pressing SELECT button for more than 3 seconds. Then click the same button to change readings from one sensor to the next. Long press the button again to return to automatic cycling. Data is saved in both modes to the SD card, but into different folders.
4. Usually, before collecting data from an experiment, we do both things explained in items 2. and 3. We connect the logger to the PC so we see what is going on and then we use the manual mode to carefully check the heart beat signal from each animal, sequentially. Only when everything is running properly, we start the experiment for real.
5. Stop data collection by simply disconnecting the logger from its power supply. Data are saved every 10 seconds, so disconnecting the device will mean losing 10 seconds of data at most.
6. The SD card should have a series of log files stored inside two folders – folder "A" for files saved during automatic mode, and folder "M" for files saved during manual mode. Files are kept in separate folders because files saved during manual mode will inevitably have different read lengths, and are usually not meant for analysis. However, they still have exactly the same file structure, and thus can be used for subsequent analysis. All file names are unique, so in the end all files may be moved into a single folder – no overwriting will occur.
7. Each file is named according to the following format: **`X_yyyymmdd_HHMM.csv`**
   7.1. `X` stands for the sensor number (1 to 8).
   7.2. **`yyyymmdd`** is the date as in "20180803".
   7.3. **`HHMM`** is the time as in "1344".
   7.4. For example, the file **`6_20180803_1344.csv`** is a file collected by the sensor 6, on the 3rd of August, 2018, at 13:44, or 1:44 pm.
8. An example of a log file can be found at the end of this manual.
9. Also note that there may be incomplete or zero bytes files in the SD card – those result from disconnecting the power source before the logger finished writing any data into a file, and since the logger starts collecting data automatically it is common that these files get generated, especially during system setup.

DATA ANALYSIS

1. Data are stored as integer values between 0 (minimum) and 1023 (maximum). Use your favorite software (excel, R, Matlab) to plot the signal variation against time (Fig. 5).
2. If you want to convert the 0-1023 values into voltage, use this formula:

```
Voltage = (1024 / (value + 1) * 5)
```

3. The number of full oscillations divided by elapsed time in seconds is the heart beat rate in Hz.
4. It is extremely important to acknowledge this fact: - **this system can only be used to calculate heart rates (frequency)**. Since the "strength" of the signal depends on the sensor, on the position of the sensor, etc etc etc, the shape of the signal can be completely different from one animal to another. Even in the same animal, the **shape of the signal** varies a lot, depending on the heart frequency, temperature, body position etc. Thus, the only reliable information that is given by the system is the **frequency** of oscillations. How do you know if the oscillations are a real heart beat and not an artifact or just noise? This is one of those cases that after seeing it once, you have no doubts. Basically, a true heart beat is **regular**.
Do not attempt to extract any other information whatsoever from the shape of the signal!

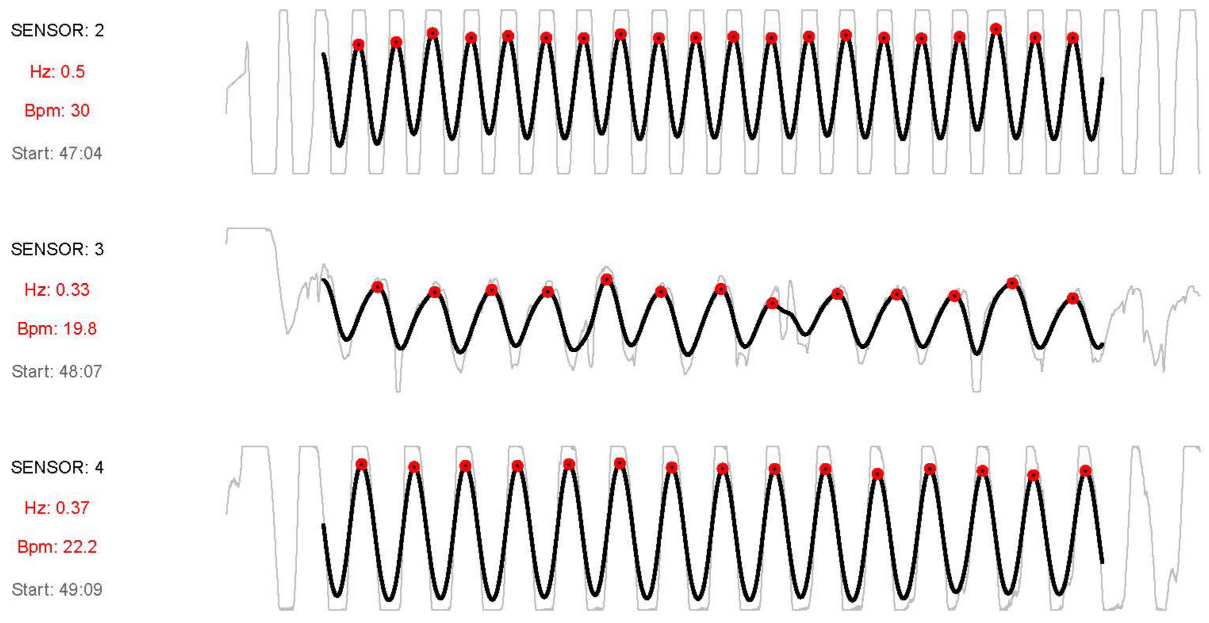


Fig. 5. Examples of good heart rate signals. The gray line is the original data and the black thick line is a kernel smoothing to help on the peak identification (red circles). The number of red peaks per second gives the frequency of the heart. The shape of the signal (more squared as in sensor 2 or more round as in sensor 3) tells us nothing. The amplitude of the signal (higher on sensors 2 and 4 as shallower on sensor 3) has no significance either (that can be due to the placement of the sensor or shell thickness, for example).

5. If you are using R to process the signal:

5.1. In the PC browse to the folder `software/R_analysis_of_hb_data/45_7/`, open the `hb.Rproj` RStudio project, and then open the file `hb_analysis_v45_7.R`.

5.2. Edit the path (line 15) to match the folder where the files to be processed are (this can be done directly on the contents of the folder `/DATA/`, in the SD card, or from any other folder that ONLY CONTAINS heartbeat logging csv files).

5.3. After editing the path select all the text from the `hb_analysis_v45_7.R`.

5.3.1. Copy and paste it into the R console and hit RETURN; R will then import all data that has not been processed yet - this may take a while, so go for a coffee.

5.3.2. A window will pop up revealing a plot of the heartbeat data from the first file in the target folder; if not, check that the path was properly set, or that you are not trying to process data from a folder where all data has already been processed (if problems persist email ruisea@gmail for support).

5.3.3. Maximize the window so all buttons become visible. The window displays:

5.3.3.1. TOP LEFT → details about the file being plotted

5.3.3.2. TOP RIGHT → details about the heart beat frequency

5.3.3.3. TOP CENTER → a detailed view of the area selected in the main plot (CENTER, grey on white background); this is the portion of the file that is effectively used to compute the heart beat frequency

5.3.3.4. CENTER → the main plot, showing all available data in the current file; click and drag to select a shorter range of data for computing the heart beat frequency (the area selected will be shown in the top plot with greater detail)

5.3.3.5. BOTTOM → tools for controlling the analysis

5.3.4. Evaluate the heartbeat using the available button tools, for which you can find the details below. **Adjustments made to these parameters carry to the next/previous file, so take great care to always be mindful of that, especially for the MULTIPLIER and CONFIDENCE parameters.**

5.3.4.1. BANDWIDTH → the bandwidth applied to the smooth function – the higher the value the greater the smoothing; adjust this value file by file to obtain a proper identification of heart beats.

5.3.4.2. MULTIPLIER → in the event that each heartbeat has 2 peaks - thus resulting in a heartbeat frequency twice the real value - set MULTIPLIER to 0.5 to halve the frequency. **ATTENTION: make sure not to forget to switch back to 1x as soon as appropriate.**

5.3.4.3. FILE → either type the index of the file you want to go to (not recommended) or use the up and down arrows to move to the next or previous file, respectively. **ATTENTION: be mindful that whenever you move to previous files (including by typing in the index of a previous file) all heart beat frequencies already computed for the files after that one are erased (!!) and will need to be processed again.**

5.3.4.4. CONFIDENCE → choose one of OK, Low and Discard to add that tag to the heart beat frequency of each file; this ensures you can later easily exclude all the discards. Importantly, it also postpones the decision to include/exclude files for which the data was not very good (Low confidence) afterwards, without having to make that decision on the fly. **ATTENTION: it is not uncommon to set confidence to Discard or Low because of a single file and then forget to switch it back to OK – make sure to avoid that.**

5.3.4.5. REMOVE x SECS → truncate "x" seconds from the beginning (left) using the top slider or from the end (right) using the bottom slider. This is intended to make the screening of the files faster, as data quality is typically reduced at the edges, especially at the beginning. By automatically truncating a few seconds in the beginning of the file one does not have to do it manually.

5.3.4.6. SAVE → for large datasets saving data is a slow process, and if it was done at each change of file the script would run visibly slower in many computers; instead, in this script **data is only save upon clicking the SAVE button (!!!).** Don't forget to do it every few files!!! Quitting R, or if the computer crashes, will erase all data processed after the last time the SAVE button was clicked!!!

5.3.5. Eventually a point will be reached were the file does not change, meaning all files have been processed.

5.4. Check the folder where the heartbeat files are stored to find two files prefixed `_output`. The file `_output.csv` holds all heart beat frequency data in a csv format, while `_output.RData` stores the same data but in a format ready for using in R.

OPTIMIZATION

1. If you are working with the tip of the sensors submerged in water, place the system above the water level to prevent water from infiltrating the sensor cables and flooding the entire system.
2. If you are using the system on a new species for the first time it may be hard to find the heart position. The PULSE system has a narrow slit to allow the light of a series of small LEDs to shine through (Fig. 6). That LED strip should flash up and down together with the heart beat (resembling the “Knight Rider’s KITT” bump scanner), and so, it should help locating the optimal place for gluing the IR sensor. The large green LED also shows the output of the IR sensor and if a heartbeat signal is present it will fade on and off in a regular fashion.
3. Having the board connected to a PC increases the noise of the signal and introduces artifacts, in particular if the PC is connected to a wall plug. The same holds true for powering the system with an AC adapter. If electrical noise is a problem try using the system connected to a computer that is running on batteries or try powering the system with a 9V battery.
4. It is quite common that animals do not show a good signal after being disturbed. Usually one has to let them rest at least 15 minutes but sometimes it is necessary to leave them for 30 or more minutes without being disturbed and only then they start showing a regular heartbeat.
5. The red LED blinks to indicate which sensor is being red; sensor 1 = 1 blink, sensor 8 = 8 blinks.

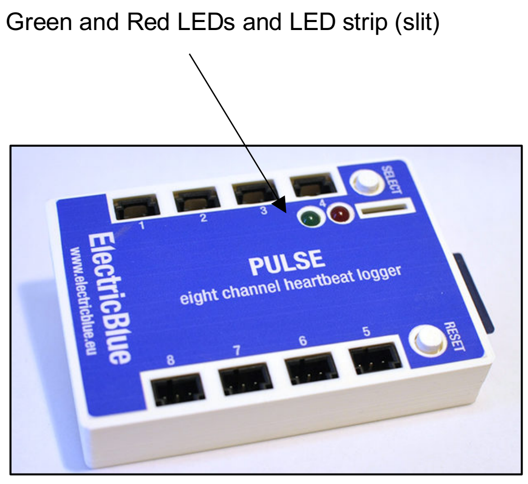

Green and Red LEDs and LED strip (slit)
ElectricBlue
www.electricblue.eu
PULSE
eight channel heartbeat logger
SELECT
RESET
1
2
3
4
5
6
7
8

Fig. 6. The two LEDs and the slit through which the micro-LED strip shines through.

SENSOR ASSEMBLY AND REPLACEMENT

The used infrared sensor is a CNY70 (Fig. 7), but there are a variety of equivalent sensors that can be used. There are many shapes and sizes. Some are cubic, some flat, some have the IR emitter and the receiver together while others have them separated. The CNY70 should be appropriate for most applications, but you may try others if you want to improve the basic design. The sensor is connected to a shielded cable that has two wires + shield (Pro Power 3027442 or equivalent).

If handled with caution, each sensor can be glued to an animal, detached and re-used multiple times. Each time a sensor is detached, it is advisable to shave off the remnants of glue, keeping the surface clean and flat. Due to variations in assembling quality (especially due to variations in soldering quality) it is likely that within the same batch of sensors some will last a long time while others may malfunction since day one.

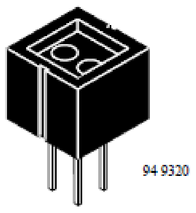


Fig. 7. CNY70 sensor drawing. The top has a protruding lip that can be shaved off for increased adherence to the animal's shell or carapace. The two circles on the top surface represent the IR emitter and receiver.

To assemble a sensor + cable follow these steps:

1. Cut the cable to a convenient length (e.g., 1m). Strip the wire ends and tin them (i.e., apply a small amount of solder to the tip of each wire).
2. Take the CNY70 sensor and trim the 4 legs (a few mm are enough) to reduce the overall size (Fig. 8).

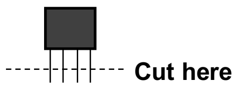


Fig.8. Schematics showing how to trim the CNY70 legs.

3. Following the schematic below, and having the sensor with the legs pointing to you and with the marked side up, bend the top right leg towards the low left leg and solder them together (Fig. 9, black line).

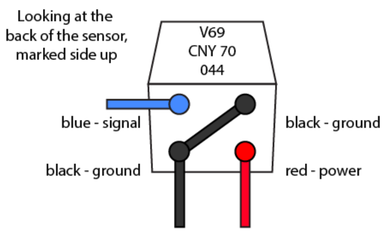


Fig. 9. Wiring of the CNY70.

4. Solder the blue wire to the top left leg and the red wire to the bottom right leg.
5. Twist the wire shield and solder it to the bottom left leg (the one that is already connected to the top right leg, shown in black in the figure).

6. Make sure the legs do not touch in order to not short circuit.
7. Cover the legs and the back of the sensor with hot glue to waterproof everything.
8. (Optional) Shave the outer plastic lip of the sensor with a Dremell abrasive rotary tool to flatten the face of the sensor (Fig. 10).
9. Take the other side of the cable, and solder the wires to three Molex pins (part number 16-02-0102).
10. The cable shielding should also be twisted and soldered to a Molex pin.
11. Insert the pins into the Molex plastic socket 50-57-9403, respecting the color order as on Fig. 11.

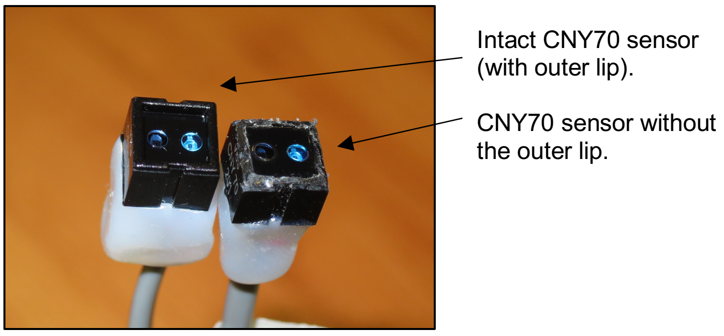

Intact CNY70 sensor (with outer lip).
CNY70 sensor without the outer lip.

Fig. 10. Two CNY70 sensors side by side. The one on the left is intact, but the other is missing its outer lip, which was removed with an abrasive tool.

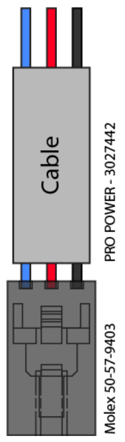


Fig. 11. How to connect the Molex plug to the sensor cable.

REPLACING THE LOGGER'S BATTERY

The SD Logger has a CR1220 battery backup, being able to keep track of time even when the power is disconnected. Even though the battery should last for about seven years, it will inevitably need to be replaced. You will have to unscrew the base of the box and replace the small coin-cell battery inside.

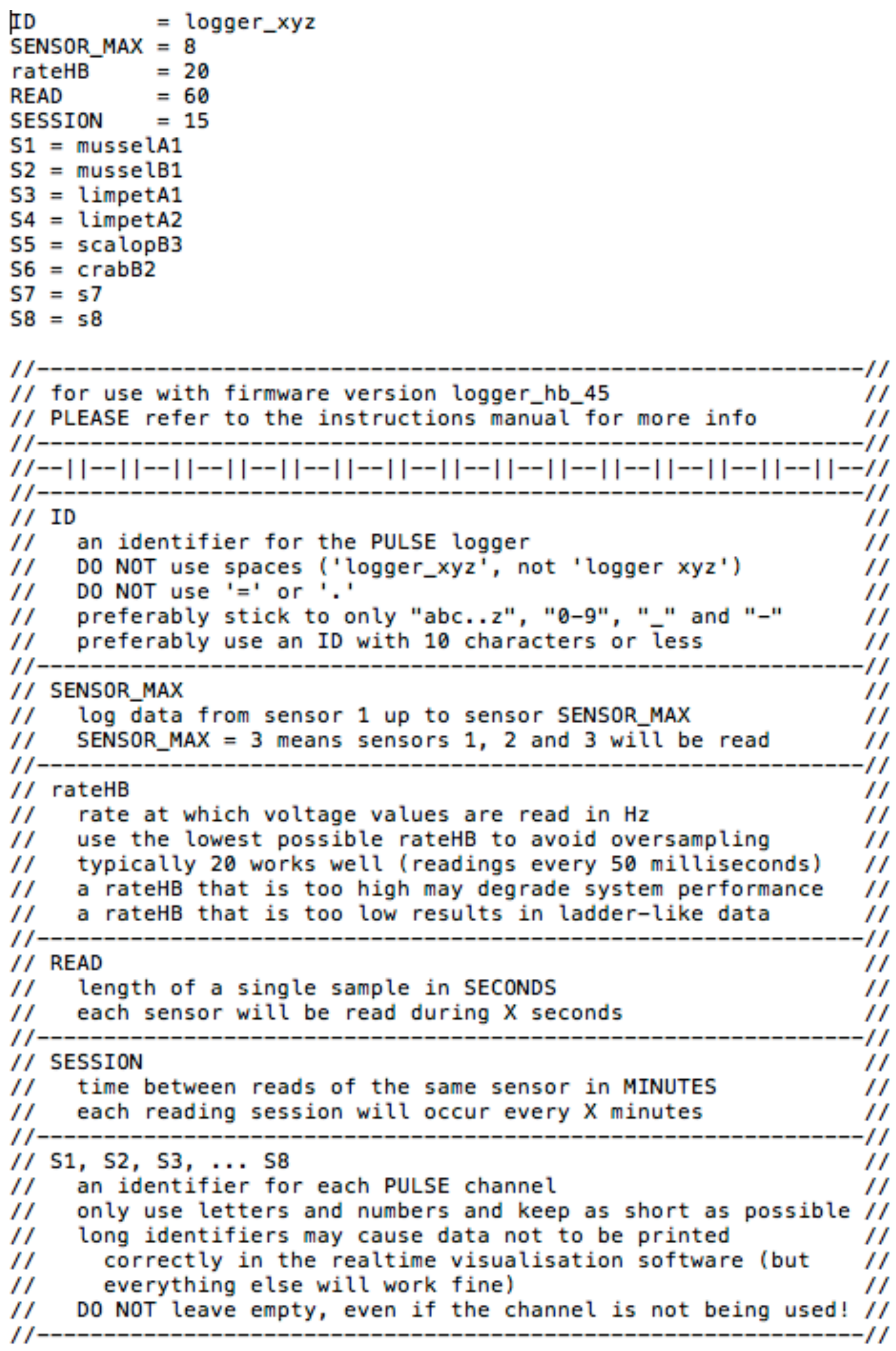
When replacing the battery, the clock will be momentarily disconnected from power and thus will stop. This will make the SD logger to reach an error state and the system will not work.

To restart the logger clock:

1. Upload the sketch `start_clock.ino` that can be found in the folder `/software/start_clock/`.
2. After restarting the clock, repeat the procedure described on **CLOCK SYNCING** to sync the clock.

Don't forget to re-program the Arduino as explained above in section **SYSTEM TESTING AND SET-UP.**

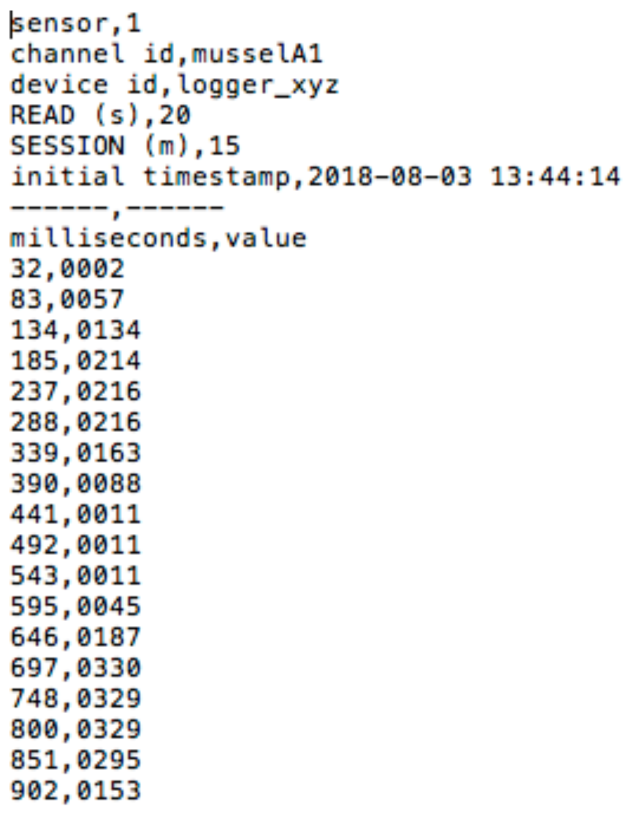
EXAMPLE OF SETTINGS FILE `hbsett.txt`



```
ID          = logger_xyz
SENSOR_MAX = 8
rateHB      = 20
READ        = 60
SESSION     = 15
S1 = musselA1
S2 = musselB1
S3 = limpetA1
S4 = limpetA2
S5 = scalopB3
S6 = crabB2
S7 = s7
S8 = s8

//--------------------------------------------------------------//
// for use with firmware version logger_hb_45                  //
// PLEASE refer to the instructions manual for more info       //
//--------------------------------------------------------------//
//--||--||--||--||--||--||--||--||--||--||--||--||--||--||--||--//
//--------------------------------------------------------------//
// ID                                                            //
//   an identifier for the PULSE logger                         //
//   DO NOT use spaces ('logger_xyz', not 'logger xyz')          //
//   DO NOT use '=' or '.'                                       //
//   preferably stick to only "abc..z", "0-9", "_" and "-"       //
//   preferably use an ID with 10 characters or less             //
//--------------------------------------------------------------//
// SENSOR_MAX                                                    //
//   log data from sensor 1 up to sensor SENSOR_MAX              //
//   SENSOR_MAX = 3 means sensors 1, 2 and 3 will be read        //
//--------------------------------------------------------------//
// rateHB                                                        //
//   rate at which voltage values are read in Hz                 //
//   use the lowest possible rateHB to avoid oversampling        //
//   typically 20 works well (readings every 50 milliseconds)    //
//   a rateHB that is too high may degrade system performance    //
//   a rateHB that is too low results in ladder-like data        //
//--------------------------------------------------------------//
// READ                                                          //
//   length of a single sample in SECONDS                        //
//   each sensor will be read during X seconds                   //
//--------------------------------------------------------------//
// SESSION                                                       //
//   time between reads of the same sensor in MINUTES            //
//   each reading session will occur every X minutes             //
//--------------------------------------------------------------//
// S1, S2, S3, ... S8                                            //
//   an identifier for each PULSE channel                        //
//   only use letters and numbers and keep as short as possible //
//   long identifiers may cause data not to be printed           //
//     correctly in the realtime visualisation software (but     //
//     everything else will work fine)                           //
//   DO NOT leave empty, even if the channel is not being used! //
//--------------------------------------------------------------//
```

EXAMPLE OF A LOG FILE



```
sensor,1
channel id,musselA1
device id,logger_xyz
READ (s),20
SESSION (m),15
initial timestamp,2018-08-03 13:44:14
------,------
milliseconds,value
32,0002
83,0057
134,0134
185,0214
237,0216
288,0216
339,0163
390,0088
441,0011
492,0011
543,0011
595,0045
646,0187
697,0330
748,0329
800,0329
851,0295
902,0153
```

Line by line:

1. the sensor was logged (sensor 1 in this example)
2. the id attributed to the current sensor channel (musselA1 in this case)
3. the id attributed to the device (logger_xyz in the example above)
4. reading time per sensor in seconds
5. session time in minutes
6. timestamp of the first reading
7. then, each log's line has the time in milliseconds since the initial time stamp and the sensor's recorded value from 0 (minimum) to 1023 (maximum).