Inlinino

A MODULAR SOFTWARE DATA LOGGER FOR OCEANOGRAPHY

By Nils Haëntjens and Emmanuel Boss

ABSTRACT. Inlinino is an open-source software data logger whose main purpose is to log scientific measurements collected during extended periods at sea. Here, we present an application of this software to data collected with commercial instrumentation. Inlinino also provides real-time visualization of the recorded observations, which helps users troubleshoot instruments in the field and prevents the collection of bad data. Inlinino is written in Python and runs on regular computers for instruments that have a serial interface. For less than $57, we built a separate data acquisition module—a precision analog-to-serial converter—for interfacing instruments that output analog signals to Inlinino. Inlinino was designed for optical sensors but can be used with any environmental sensor that communicates through analog or serial ports. The code is sufficiently modular that anyone with moderate coding skills can add new sensors. To date, Inlinino has been deployed successfully on several research vessels and logged more than 650 days of operation.

INTRODUCTION

Our laboratory equips research vessels with instruments to continuously measure the optical properties of the ocean while underway, offering extended spatial and temporal coverage of the state of the ocean's surface (Boss et al., 2019). These observations are used, for example, to assess phytoplankton biomass and size (Boss et al., 2015; Graff et al., 2015) and suspended sediments (Dierssen et al., 2006). These products are computed from optical properties such as fluorescence, backscattering, absorption, and attenuation. Each instrument used—for example, WET Labs ECO-BB3, WET Labs WETStar WSCD, WET Labs ACS—measures a specific property that needs to be recorded externally. Given the instrumentation involved and our needs, the data logging system has to be capable of (1) interfacing with instruments through serial (RS-232) or analog (0 V to 5 V) communication, (2) accurately timestamping the data received with the research vessel NTP server, (3) storing the data in files easily readable by data analysis software (e.g., R, MATLAB), and (4) providing data visualization for real-time monitoring of the measurements.

We first looked for instrument manufacturer's logging solutions that might match our requirements. WET Labs designed systems for two broad situations: (1) short-term use in a laboratory environment (e.g., testing equipment or experiments over a few hours), and (2) autonomous deployment underwater (e.g., profiling the ocean to several hundred meters depth). While both solutions worked, we found them impractical for continuous, month-long use. The first solution generated oversized data files, and the second was found to have a drifting timestamp when used over extended periods, in addition to having memory limitations (e.g., limited to the memory of the device, with hour-long data download from the data logger to the computer, preventing data acquisition).

While looking for alternatives, we came across many commercial hardware data loggers (e.g., Campbell Scientific). Although hardware data loggers would have likely met our requirements, we oriented our approach toward software data loggers running on computers, as they had several advantages: (1) computers could easily be synchronized with the boat NTP server, (2) computer screens would allow visualizing the data in real time, and (3) computers are widely available in most scientific laboratories.

We chose the software TeraTerm for a first test at sea. TeraTerm, a proven serial terminal emulator, was extremely reliable and covered all the required specifications except that it was not capable of providing real-time visualization, an attribute critical to ensuring that the instruments operate well. We then adapted an open-source solution to our needs called Instrumentino (Koenka et al., 2014, 2015). We modified this software to make it suitable for long-term deployments by improving its memory management (restricting the data buffer to a fixed-size array). This proof-of-concept was successfully tested in 2016 on a transatlantic cruise aboard R/V *Tara*. Subsequently, we sought to design an open-source, modular (seven unique instruments implemented to date), and robust (logs without user interaction for

>30 days) data logging solution with real-time visualization for in situ ocean optical instrumentation. We call our software product Inlinino. In addition to the software, and also described in this paper, we built a data acquisition (DAQ) device for interfacing instruments that have analog outputs to computers using an open-source philosophy. The DAQ is a precision analog-to-serial RS-232 converter (PASC).

INLININO SOFTWARE

The core of Inlinino includes a series of simple features essential for long-term deployment that also minimizes its computational footprint. Figure 1 shows the flow of data between the instruments and the computer. Inlinino is entirely configured (e.g., for selected instruments) through a file loaded when the software starts. Incoming data from instruments are first timestamped with the computer set to UTC and synchronized to the research vessel's NTP server using the operating system service (e.g., Windows Time Service) to ensure accurate timestamping (within a second). Next, the data are stored in a ring buffer, similar to a first-in, first-out (FIFO) array except that the size of the array is constant. This enables data visualization of the last few minutes (using the fast PyQtGraph library, http://www.pyqtgraph.org/) and control of the software memory footprint. The data are then written to the computer's hard drive every minute in comma-separated values (.csv) files, limiting the loss of data in case of unexpected computer failure. The size of each file generated is constrained to one hour of records, restricting the file size and preventing the creation of impractical, oversized files. The data directory is shared on the local network with read-only permission, enabling automatic daily backup from other computers using synchronization software (e.g., SyncToy on Windows and rsync for Linux and macOS).

At the time of writing, available instruments include the WET Labs ECO-Triplets (e.g., BB3, BBFL2), BB9, BBRT, and UBAT; the Sea-Bird SBE45 thermosalinograph; and the Satlantic photosynthetically active radiation (PAR) sensor. In addition, any analog instrument having up to 16-bit precision with a voltage range of 0 V to 6.144 V can be used (using the PASC). Inlinino is modular and designed to interface with any type of serial instrument; therefore, users with good knowledge of Python can add new instruments. See https://inlinino.readthedocs.io/en/latest/cfg.html#adding-a-custom-instrument for explanations of the code.

MATERIALS AND INITIAL INSTRUCTIONS

Inlinino hardware and software requirements are described below. Inlinino software was tested on Windows 7 and 10 and macOS 10.12, 10.13, and 10.15 operating systems but should run in any environment supporting Python 3 and pyQt 4 (e.g., Windows >7, macOS >10.15, and Linux). The computer running Inlinino requires one serial port or analog input channel per instrument used. A serial-to-USB converter is likely needed, as today's computers rarely ship with serial ports. We had positive experiences using Aten, Moxa, and StarTech adapters. Regarding the analog input channels, a DAQ device compatible with Inlinino is required only if data from analog instruments are to be logged. To date, the PASC is the only DAQ device supported by Inlinino, but others could be implemented (e.g., DATAQ Instruments Model DI-1100, Measurement Computing USB-201, National Instruments USB-6000, Omega USB-1208FS).

Inlinino's installation and first-use instructions are briefly described below. For more detailed documentation, we recommend using the instructions available on ReadTheDocs (https://inlinino.readthedocs.io/). Inlinino Windows installer can be downloaded from ftp://misclab.umeoce.maine.edu/software/Inlinino/Inlinino_setup_win32.exe. Inlinino can then be installed by following the setup instructions from the executable files. For other operating systems (macOS and Linux), Inlinino must be installed from the source code; the procedures and codes are available on GitHub (https://github.com/OceanOptics/Inlinino). Once installed, application parameters (e.g., type of user interface, theme), logging parameters (e.g., interval at which data are written to disk, periods of data to keep in cache), and instrument parameters (e.g., instruments to log, instrument types) can be set up through the configuration file (https://inlinino.readthedocs.io/en/latest/cfg.html). The application can be started by double-clicking on its icon (available in Start

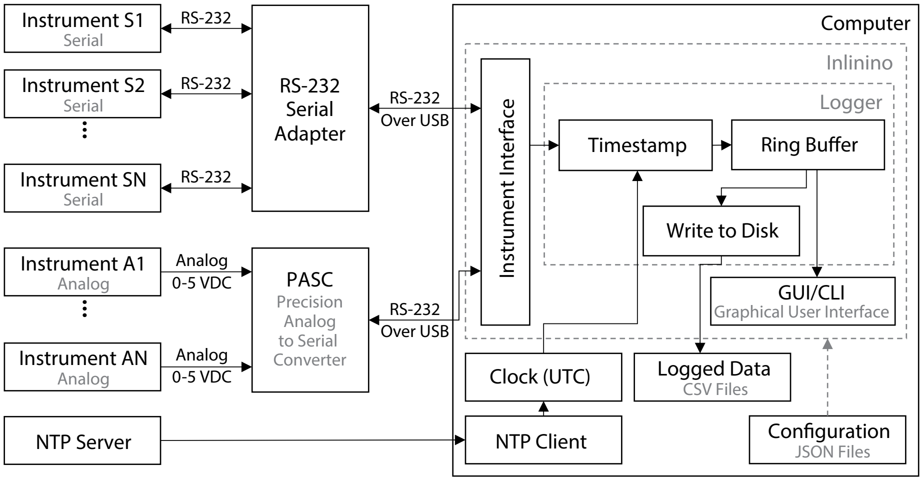


FIGURE 1. Flow chart representation of Inlinino software running on a computer and associated instruments.

Menu/Programs/Inlinino for Windows). The graphical user interface (GUI) of Inlinino (Figure 2) is designed to be intuitive, allowing experienced users to skip the user manual instructions.

For interfacing analog instruments to the computer running Inlinino, we needed a DAQ device capable of converting the analog signal to a digital signal. In the open-source philosophy of Inlinino, we built a simple PASC, primarily for educational and low-cost purposes. The PASC (Figure 3) is composed of a precision analog-to-digital converter (ADC, Texas Instruments ADS1015 or ADS1115) and a microcontroller (Arduino Uno) that "translates" the digital signal (I^2C protocol) from the ADC to serial communication over USB. These components are packed into a water-resistant box (see Table 1 for a complete list of parts). To further reduce the cost of the PASC, the Arduino Uno could be replaced by a cheaper microcontroller with one I^2C port and one UART port, and the latter could be shifted to RS-232 level or directly converted to a virtual serial port through USB (Table 1b). The wiring of the PASC is explained on Adafruit ADS1015's web page (https://learn.adafruit.com/adafruit-4-channel-adc-breakouts/assembly-and-wiring), and the firmware to upload on the microcontroller is available on the Inlinino GitHub repository (https://github.com/OceanOptics/Inlinino).

VALIDATION AND FIELD APPLICATION

To evaluate Inlinino's specifications, we compared the logged data to TeraTerm (downloaded >7.6 million times as of January 23, 2020). We ran an experiment with a WET Labs BB3 sensor (serial number 1502) connected to a "spy" RS232 cable connected to two serial ports on a computer. A "spy" RS232 cable consists of two receivers for the same transmitter. Therefore, the exact same data could be received at the same time by both TeraTerm and Inlinino. The experiment ran for four hours and 16 minutes. We then analyzed the data to check the following:

(1) Dropout Rate

Does the software miss data?

The exact same number of data lines (n = 13,596) were received and written to disk on both TeraTerm and Inlinino.

(2) Timestamp Accuracy

How accurate is the timestamp?

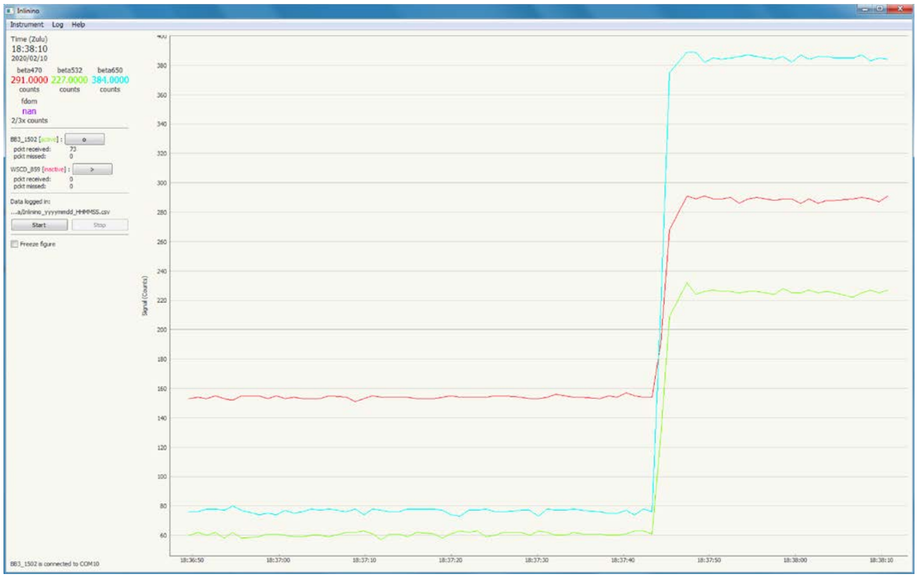
Inlinino's timestamp was accurate within 1 second with an average delay of 0.54 seconds and 75th percentile at 0.8 seconds. This lag can be explained by the fact that Inlinino writes data from all the instruments it is connected to every N seconds (as set up in the configuration of Inlinino). For oceanographic underway measurements, this is acceptable as most measurements are averaged over 30 seconds or more.

(3) Data Quality

Do the data received match the data sent by the sensor?

Identical values were written to disk for each channel of the BB3 sensor.

The data visualization part of Inlinino was qualitatively assessed during the same experiment. The data displayed in TeraTerm was the same as the data plotted in Inlinino. A short delay (<1 second) was noticed on Inlinino's interface. Because Inlinino uses the same buffer to write the data to disk and render the visualization, the visualization was assumed to be correct. In addition, Inlinino was



Time (Zulu)
18:38:10
2020/02/10

beta470	beta532	beta650
291.0000	227.0000	384.0000
counts	counts	counts

fdom
nan
2/3x counts

BB3_1502 [active] :
pckt received: 73
pckt missed: 0

WSCD_859 [inactive] :
pckt received: 0
pckt missed: 0

Data logged in:
...a/Inlinino_yyyymmdd_HHMMSS.csv

BB3_1502 is connected to COM10

FIGURE 2. Inlinino graphical user interface (GUI).

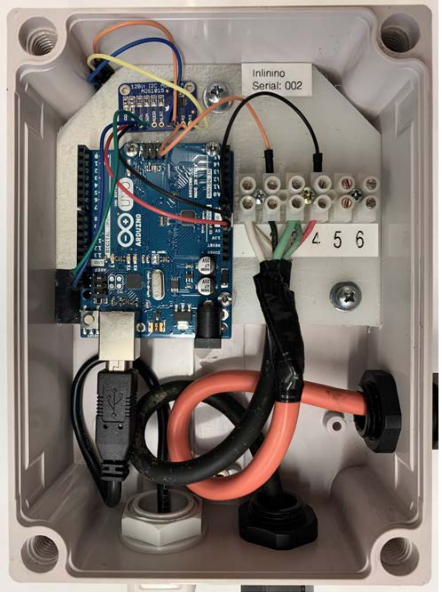


FIGURE 3. Assembly of the precision analog-to-serial converter (PASC) in a waterproof enclosure rated IP66.

deployed on several research vessels (Table 2) during more than 650 days at sea. For most of the cruises, the data were quality checked manually (see Boss et al., 2019) and published in the NASA SeaBASS database.

We assessed Inlinino's computing performance by running it with one instrument (WET Labs BB3 serial number 1502) on a laptop (Lenovo E530, CPU i3-3120 @2.5 GHz, 4Gb RAM, Windows 7, ~$500). The CPU usage was monitored for two hours and five minutes with the Windows Performance Monitor utility. Inlinino was the only application running during the test. The processor's clock frequency was automatically down-throttled to 1.3 GHz 66% of the time with a median CPU usage of 1.2% and 75th percentile of 2.0%, validating its low computing power characteristics. When Inlinino was also used on several laptops during multiple month cruises (Table 2), CPU usage was minimal (qualitatively assessed through Windows Task Manager) and no data logging problems were encountered.

PASC's accuracy was first assessed in the laboratory when it was built and during its first cruises at sea with a Fluke 85 III voltmeter. While it performed as expected (close to the Texas Instruments ADS1015 specifications), no data were kept for a scientific evaluation of its performance. The accuracy and precision of PASC serial numbers 001 and 002 were evaluated after 145 days and 380 days of operation at sea, respectively (figure available in Inlinino's documentation, https://inlinino.readthedocs.io/en/latest/pasc.html). The Wet Labs WETStar WSCD serial number 859 was connected to the PASC for the test and the full range of values output by this sensor (0 V to 5 V) was used. The difference between the two PASCs and the reference voltmeter was never greater than 0.05 V, with a root mean squared error of 0.0053 V. While no significant bias was observed, the precision was twice that of the manufacturer's specification (0.003 V). For other characteristics of the electronic chip used in the PASC such as drift with temperature (0.015 mV/°C), we refer the user to the manufacturer data sheet.

SUMMARY AND FUTURE WORK

The current version of Inlinino is operational, robust, and easy to use on a routine basis while at sea. It permits continuous collection of days of data and fits the needs for logging measurements from underway systems. Because Inlinino is open source, it can be adjusted as needed as more people use it.

From our experience using Inlinino, we identified a few features that need improvement:

- Several times (about once per expedition) when the instruments were running and data were displayed in real time, the logging was disabled, so the data were not recorded. We concluded that this problem could be reduced by including a large visual and/or audio signal in the interface to warn the user that data are not being logged.
- Instrument configuration at the beginning of each cruise was not intuitive, so users with limited coding skills could find it difficult to write the correct instructions. It would be good to add a user interface to Inlinino that configures instruments automatically at the beginning of each cruise.

Future development of Inlinino could also include:

- Support of additional sensors, in particular, the WET Labs ACS and the

TABLE 1. Parts used to build a PASC. (a) First version built for $56.90. (b) Smaller version built for $28.40.

(a) COMPONENTS	PRICE (US)
Arduino Uno	$22.00
Adafruit ADS1015	$9.95
Waterproof Enclosure IP66	$19.95
Barrier Strips + Electrical Wires	<$5.00

(b) COMPONENTS	PRICE (US)
Adafruit Trinket M0	$8.95
Adafruit ADS1015	$9.95
Aluminum Enclosure IP54 Sparkfun PRT-13839	$4.50
Barrier Strips + Electrical Wires	<$5.00

TABLE 2. List of cruises on which the Inlinino software and the PASC were deployed.

CRUISE NAME	CRUISE DATE	INSTRUMENTS LOGGED	DAYS LOGGING
NAAMES II	May 2016	BB3 & WSCD	26
Schiller Coastal Studies Center	Summer 2016	TSG, BB3, BBFL2, & 3X1M	90
PEACETIME	May 2017	FLBBCD, BB3, & WSCD	33
NAAMES III	Sep 2017	BB3 & WSCD	26
SPURS-2	Oct–Nov 2017	BB3	30
NAAMES IV	Mar–Jun 2018	BB3 & WSCD	26
EXPORTS I: R/V *Roger Revelle*	Aug–Sep 2018	BB3 & WSCD	34
EXPORTS I: R/V *Sally Ride*	Aug–Sep 2018	TSG, BB3, BBFL2, & 3X1M	36
Tara Pacific	May 2016–Oct 2018	PAR, BB3, & WSCD	495, 261, & 373*
Tara Breizh Bloom	May 2019	BB3 & WSCD	7

* During the Tara Pacific Expedition, the instruments were installed at different times and had to be shipped for maintenance, resulting in different numbers of days logged between instruments.

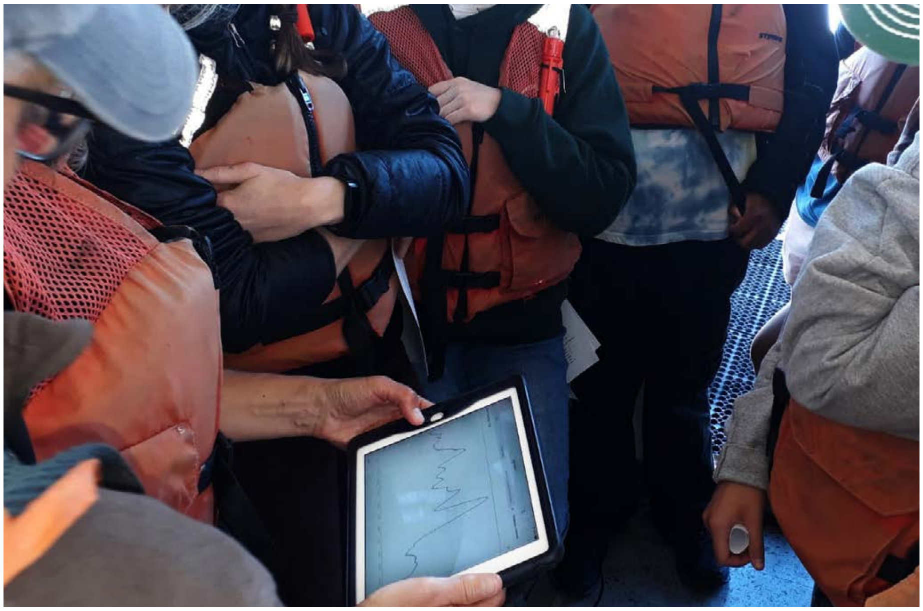


FIGURE 4. Biological Oceanography Class (SMS 501) from the University of Maine observing chlorophyll *a* fluorescence as they progress through the Damariscotta (Maine) Estuary (Maine, USA) on R/V *Ira C*. Measurements from the WET Labs WETStar WS3S fluorometer and the Sea-Bird SBE 16plus CTD sensor were logged by Inlinino and ThingsBoard running on a single board computer (Raspberry Pi 3) and streamed to a tablet in real time. ©*Guillaume Bourdin*

Sequoia LISST. The ACS was recently integrated using the Python library pyACS (https://pypi.org/project/pyACS/) but requires further testing before it can be deployed.

- Enable state-of-the-art processing, applying optical corrections, and computing products such as phytoplankton carbon or chlorophyll (Boss et al., 2019) in near-real time. The processing algorithms were developed and tested on most of the cruises mentioned above; however, the graphical user interface currently provided limits the utility of the software (InlineAnalysis, https://github.com/OceanOptics/InLineAnalysis). The real-time processing inside Inlinino would enable troubleshooting earlier in data acquisition, helping scientists to make data-informed decisions in near-real time in the field.
- Streaming real-time data recorded by Inlinino to any web browser, including on phones and tablets, would allow system monitoring from anywhere on the research vessel while at sea. For use in a field class, we developed a first proof of concept (Figure 4) that takes advantage of the recent development of the Internet of Things (IoT) and the ThingsBoard platform (https://thingsboard.io). The source code for this module is available on the dedicated branch of Inlinino on GitHub (https://github.com/OceanOptics/Inlinino/tree/tb-app).

REFERENCES

Boss, E., L. Guidi, M.J. Richardson, L. Stemmann, W. Gardner, J.K.B. Bishop, R.F. Anderson, and R.M. Sherrell. 2015. Optical techniques for remote and in-situ characterization of particles pertinent to GEOTRACES. *Progress in Oceanography* 133:43–54, https://doi.org/10.1016/j.pocean.2014.09.007.

Boss, E., N. Haëntjens, S. Ackleson, B. Balch, A. Chase, G. Dall'Olmo, S. Freeman, Y. Liu, J. Loftin, W. Neary, and others. 2019. *Inherent Optical Property Measurements and Protocols: Best Practices for the Collection and Processing of Ship-Based Underway Flow-Through Optical Data*. Report of a NASA-sponsored workshop. A.R. Neeley and A. Mannino, eds, IOCCG Protocol Series Volume 4.0, 22 pp., https://ioccg.org/wp-content/uploads/2019/11/inline-protocols-4.0-nov2019.pdf.

Dierssen, H.M., R.M. Kudela, J.P. Ryan, and R.C. Zimmerman. 2006. Red and black tides: Quantitative analysis of water-leaving radiance and perceived color for phytoplankton, colored dissolved organic matter, and suspended sediments. *Limnology and Oceanography* 51(6):2,646–2,659, https://doi.org/10.4319/lo.2006.51.6.2646.

Graff, J.R., T.K. Westberry, A.J. Milligan, M.B. Brown, G. Dall'Olmo, V. Dongen-Vogels, K.M. Reifel, and M.J. Behrenfeld. 2015. Analytical phytoplankton carbon measurements spanning diverse ecosystems. *Deep Sea Research Part I* 102:16–25, https://doi.org/10.1016/j.dsr.2015.04.006.

Koenka, I.J., J. Sáiz, and P.C. Hauser. 2014. Instrumentino: An open-source modular Python framework for controlling Arduino based experimental instruments. *Computer Physics Communications* 185(10):2,724–2,729, https://doi.org/10.1016/j.cpc.2014.06.007.

Koenka, I.J., J. Sáiz, and P.C. Hauser. 2015. Instrumentino: An open-source software for scientific instruments. *CHIMIA International Journal for Chemistry* 69(4):172–175, https://doi.org/10.2533/chimia.2015.172.

ACKNOWLEDGMENTS

We would like to thank Collin Roesler and Guillaume Bourdin, the first users of Inlinino, for working with the earlier versions of the software, providing insightful feedback, and helping Inlinino to achieve a higher technology readiness level. We thank Andrew Davies and two anonymous reviewers for detailed reviews that significantly improved the manuscript. This work was funded by NASA grant NNX15AE67G (NAAMES). The data collected during the cruises are available on NASA SeaBASS.

DECLARATION OF INTEREST

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no financial support for this work or relationship with a company that could have influenced its outcome.

AUTHORS

Nils Haëntjens (nils.haentjens@maine.edu) is PhD Candidate, and **Emmanuel Boss** is Professor, School of Marine Science, University of Maine, Orono, ME, USA.

ARTICLE CITATION

Haëntjens, N., and E. Boss. 2020. Inlinino: A modular software data logger for oceanography. *Oceanography* 33(1):80–84, https://doi.org/10.5670/oceanog.2020.112.

COPYRIGHT & USAGE