**Functional Specification for Real Time Data Display for OSU/OTIC Heat Flow System**

A. Fisher, B. Morris, etc.

v. 0.9, last updated 4/12/21

**I. Introduction and Purpose**

This document describes desired functional specification for software that will receive and display data telemetered to an oceanographic vessel during operation of a multi-penetration heat flow system that is in development by a team of OSU, WHOI, and UCSC researchers and engineers. The end of this document also contains a "wish list" of features for discussion. Some will be easier to implement than others.

Complete data collected by the seafloor heat flow system will be stored internally on a logger, recovered and processed when the system is returned to the ship. A subset of data will be telemetered using a SonarDyne system that is also used for ultra-short baseline (USBL) navigation. Received data will be parsed by a dedicated computer, separating navigation and heat flow data. The heat flow data will arrive by serial connection (we believe…detailed TBD). This document focuses on how we would like to display the heat flow data once it is received on a dedicated computer.

The overall approach is similar in some ways to that developed for the Alvin-style heat flow probe. However, the Alvin probe is operated with a cabled serial connection that allows two-way communication, probe control, and data transmission. For the OSU/OTIC heat flow system, communication is by acoustic modem and is probe-to-ship only. In principle we could have two-way communication and probe control as part of this system, but this would require a large power draw on the seafloor (supplied by batteries in a limited space) and would increase the complexity of operating software on the probe.

The operating system for the OSU/OTIC heat flow logger is described in a separate document, which includes an explanation as to how the data stream to be telemetered from instrument to ship will be determined. The next section of this document describes data selection for telemetry, then focuses on how the data will be displayed and logged.

**II. Data to be Telemetered**

The operator of the heat flow system will set a series of parameters in a file that is accessed by the data logger on start up. Once these are set, they can be changed only by restarting the heat flow probe with a different parameter file. Among the parameters set for probe operation are these related to the interval for data collection and telemetry, and what data are to be included in the telemetry stream.

|  |  |  |
| --- | --- | --- |
| **Command** | **Purpose** | **Comments/questions** |
| sps = n | Sample period set, where n = 1 to 180 s | Interval of write to data file |
| tps = m | Telemetry period set, m = multiplier of sps | Interval of send data to USBL system for telemetry |
| tdls = XX XX… | telemetry data line set | Tell logger what to telemeter |

**Explanations**

- sps = n

Typical time interval for sample collection by the heat flow system will be 1 to 10 s, although longer sample intervals could be selected to save power, if desired.

- tps = m

This is a multiplier, determines the time interval for a subset of data to be sent to the USBL unit for telemetry. For example, if sps = 2, indicating 2-second sampling by the system, a value of tps = 10 would result in data sent from the logger to the USBL modem every 20 seconds (2 x 10). Data going to the telemetry unit will sit in buffer until the modem is "pinged" from surface and asked to send a location and data. The contents of this data buffer will be replaced each time new data is sent to the modem. Thus the modem will send whatever is the most recent dataset received from the logger, whether that was 1 second earlier or 10 seconds earlier, etc.

- This could result in an irregular data transmission interval if, for example, we ping for a probe position from the ship every 30 seconds, but send data to the modem from the probe every 20 seconds. To avoid this, it might be safest to set the ping interval from the surface to be an integer product of sps x tps. For example, we could have data logging to the file at 2 s (sps = 2), data sent to the USBL system at 10 s (tps = 5), then ping from the surface at 20 s.

- The main considerations at present are (a) benefits of more frequent updates showing system performance versus (b) power requirements associated with more frequent data transmission. With excessively frequent data transmission, there might also be interference or loss of fidelity if there are echoes. All of this will need to be sorted once we know more about real-world system performance and power requirements.

- tdls =…: Data set sent to modem for transmission to surface in near-real time.

This should be a subset of data written to memory, using similar format.

Example: tlds = DateTime, V, tilt, P, TBW, T01, T08, T15, T22

DateTime = date and time (format = YY-MM-DD hh:mm:ss,' ')

V = system voltage (f5.2,' ')

Tilt = angle from vertical, decimal degrees (f5.2,' ')

- The probe logger will measure and record three-component accelerometer data. For telemetry, probe will use vertical acceleration (AZ) to estimate tilt with a function like tilt = degrees(arccos(AZ/g)).

P = output of pressure gauge, converted to equivalent water depth in meters (f8.3,' ').

- Parameter set in heat probe configuration file will include conversion factor from pressure to depth = pressure/(rhow x g), where rhow = approximate water density (typically ~1040 kg/m3 for oceanic depths)

TBW = temperature of bottom water (f7.3,' ')

TXX to TYY = temperature values for four (4) selected sensors (f7.3,' ')

- The main probe system will record data from ≥11 sensors (most likely 17-18 for a 3.5 m lance), but only four (4) values will be telemetered, to aid in reading the real time record.

- Final data value to be followed by CR/NL (or LF), as appropriate

**Telemetry Data Storage**

As noted earlier, telemetered data is intended mainly for assessment of system status during operation, and to make rough estimates of geothermal conditions that could influence subsequent measurements. However, it is possible that the telemetered data will be needed for quantitative assessment if the seafloor probe system is damaged. Therefore, it is desired that we store a complete record of the telemetered data stream, following the listing and format of data shown above. The name of the file used for shipboard storage of telemetered data can be set with a parameter file and/or when the real-time monitoring program is launched.

FYI, an additional parameter in the probe operating software will specify the name of a file that will log the data stream sent for telemetry in memory on the probe logger, with a default filename = logfileUSBL.raw. This datafile will be stored in the same directory/folder on the data logger as the complete data file. This "telemetry file" may be useful for verifying behavior of the telemetry components and software, after the full system is recovered at the end of a station.

**III. Data Display**

The data display for the real time system is to be configured to show a variety of information. The display screen is divided into six "data windows," a "status window" to the left, a legend at the bottom and a drop-down menu at the top, as shown with this draft from 4/11/21 (additional features/function in development, discussed below):

A picture containing diagram

Description automatically generated

**Window 1:** Text display of last several lines of telemetered data, with full resolution.

\

**Windows 2 to 6:** Graphical display of telemetered data in "value versus record number" format. Using record number instead of time allows for much easier data handling and display, and eases manual manipulation of the X-axis, if desired. Data streams are as labeled above, with four (4) temperature records shows in Window 2. At present, these values are labeled as Probe 1 to Probe 4, but as noted below in Desired features, it would be better to have actual sensor IDs listed.

In general, data will be collected for one seafloor penetration (station) at a time. The example record above shows four penetrations, three with a heat pulse to determine thermal conductivity (1st, 3rd, and 4th) and one without a heat pulse (2nd). In an effort to save power for both the heat flow probe and the telemetry system, the probe will be powered down between penetrations of the seafloor, when the ship is being maneuvered for the next measurement. We expect that the real-time program will be shut down between measurements, then restarted when on site and before lowering to seafloor for next penetration. That said, it could be that the system is kept running for >1 penetration, in which case a record somewhat like that above could be generated.

**Status window:** This window currently shows Probe # with a check box, allowing the user to turn on/off the display of the selected sensors. By default, all will be shown.

– Please see note below asking for some additions to the status window that will be helpful in running heat flow stations.

**Legend:** If we are able to display the colors of the sensors in the Status window, we don't need these in the legend. The colors could be used for the "Sensor" in the status window. It would be best to agree on a set of standard colors for use with the four sensors, dark enough to show up clearly on the screen, maybe dark red, blue, green, black(?). Lines should be thick enough as well, maybe 1.5 pt? I wonder if this will be adjustable on the fly.

**IV. Desired features**

**Telemetry Data Storage**

We would like to write to disk a complete record of the telemetered data stream, using the format with which it is transmitted. The name of the file used for shipboard storage of telemetered data can be set with a parameter file and/or when the real-time monitoring program is launched.

**Window 1**

– Would be helpful to have the option to expand this window to show more lines of data, by reducing or minimizing other windows to make room. It would also be good to have the option to scroll this window so we can manually inspect earlier data. Scroll function could be implemented when window shows three lines, as in the example above, and/or when a larger (sub)set of data is displayed.

**Window 3**

– Change Y-axis label from "Base Temp (C)" to "BW Temp (C)."

**Window 4**

– Change Y-axis label from "Depth" to "Depth (m)."

**Window 5**

– Change Y-axis label from "Tilt" to "Tilt (deg)."

**Window 6**

– Change Y-axis label from "Battery" to "Batt (V)."

– It will be helpful to allow for Windows 1 to 6 to be minimized if desired, to allow other window(s) to be expanded

**Status window**

– We would like to have the time interval between the last two data lines shown, so the operator can verify that the time interval is as desired (could influence ping frequency). Label can be "Telemetry interval," then show ∆time (seconds) between last two readings. This could be helpful both in making good sense out of the data display in general, and especially if we have an awkward ping interval that is not an integer multiple of the interval for when data are being sent to the modem, resulting in periodic repeats or gaps.

– Would be better to have actual sensor numbers shown, rather than generic 1, 2, 3, 4. For operating the main system, we number subseafloor probe sensors from the bottom-up, so that deepest is T01 and shallowest might be T17 (e.g., 17 probes in the tube adjacent to the lance). Would be good to have operator enter sensor number that are to be telemetered in a Configuration file that the real-time program reads on start up. These would be read and displayed in the status window. In some cases, sensors will be evenly spaced (e.g., T01, T06, T011, T16), but in other cases, it might be desired to display pairs that are more closely spaced separated by a larger gap (e.g., T01, T04, T13, T16). The latter can be useful if we would like to assess the occurrence of curved gradients.

- One relatively simple way to implement this is to have probe IDs that are read from a Configuration file, by default, but also can be changed when the program is running using a menu…or perhaps by having the probe IDs appear in boxes in the Status window that can be edited as desired. Also, if the user changes these numbers, then the Configuration file could be overwritten as modified so that on next start up, the new configuration is used. As a practical matter, once the probe IDs are set for an initial survey during an expedition, it is likely that they will be used for subsequent surveys on that expedition.

– As another part of the Configuration file for the real time system, we would like to have the operator enter the spacing between each pair of sensors, so that the display can also show the apparent gradient(s) in the Status window. For example, with 20 cm sensor spacing and selection of evenly spaced sensors for telemetry (e.g., T01, T06, T011, T16), the spacing would be: "1.0, 1.0, 1.0." In contrast, with selection of sensors for telemetry that are not evenly spaced (e.g., T01, T04, T13, T16), the individual spacing would be listed as: "0.60, 1.80, 0.60."

– Given these spacings, we would like to have four (4) additional values displayed in the Status window, below sensor numbers, gradients for each of the pairs and the gradient between the deepest and shallowest telemetered sensors. These could be listed as "Gradients" with sublabels of "Full," "Upper," "Middle," and "Lower." These values would be calculated for each data line telemetered as: ∆T/∆z, where the three ∆z values in the configuration file correspond to the lower, middle, and upper intervals, and the total of these ∆z values are added for calculation of the "Full" gradient.

– We would like to save a screen shot that will help with recording conditions during a station, as part of the "station record" (along with a hand-written data sheet). This could be a function available in the menu, or could be button to lower end of the Status window – "ScreenShot" – that would save image at relatively high resolution to a raster file format (jpg, png, etc). Saving as postscript could also be useful, but not essential, especially if there is not a tool for this. Raster will be fine.

- It appears that there is room in the Status window to display the above information. In summary, if all of the requests were implemented, the Status window would show (from top-down):

Time Int. (sec): Value (f4.1)

Sensors:

Checkbox Sensor II [FYI this is shallowest sensor]

Checkbox Sensor JJ

Checkbox Sensor KK

Checkbox Sensor LL [FYI this is deepest sensor]

Gradients (degC/m):

Full (II to JJ): Value (f6.3)

Upper (JJ to II): Value (f6.3)

Mid (KK to JJ): Value (f6.3)

Lower (LL to KK): Value (f6.3)

Button for "Print Current Screen" to Raster, asks for file name and location