L0toL1.cpp Documentation

The c++ code L0toL1.cpp takes the L0 (original) json files created by Ben Greenwood's BGC_S2A decoder and takes them to L1 (perhaps not completely…TBD).

What does this mean?

1) Presents the config in "Mission" and "SCI_Parameters" applicable to a given cycle, within that cycle. The same format for these are followed.
2) L0 reports only what data was transmitted. L1 should report what data should have been sent. Any missing sensor data is reported as an empty variable. Any missing data channels are reported as fillvalue (-999).
3) GPS_Start is moved from cycle N+1 (where it was transmitted) back to cycle N. This brings the GPS representation back in line with the phy file.
4) Replace some CONFIG/Fields with fillvalue that are not valid
    a) Ice_Mn (No ice algorithm in 10.1, there will be Ice_Mn in 10.2)
    b) SCI_Paramters ("ctd") : None of these values are used/valid
    c) Removed bad DO_Surface "PRES" values.
5) Fixed data resolution of CTD data
6) Made extensive use of the CONFIG to determine the correct output of man data
7) Some more modest changes include a meta FILE_UPDATE_DATE

Compile: g++ L0toL1.cpp

I created a directory under /src called L1. I've been compiling it there. If one places it elsewhere there are a few hardcoded paths that will need to be changed.

```
#include "../hexfile/hexfile.h"
#include "../json/json.hpp"  // Include the Nlohmann JSON library
#include "../version.h"
```

I used the same environmental variable as Ben and "borrowed" where I could. If there are additional ways to do to bring the codes into alignment let me know.

One calls the routine with the float SN (or perhaps better, the TRANSMISSION_ID?) as the first input (e.g. "./a.out 4019"). This implies to the code that every cycle will be run, and the code assumes that the 'starting' "Mission" and "SCI" are located in the L0 ('original' Bens json) -1 cycle.
Alternatively, one can use 2 parameters (e.g. "./a.out 4019 8"), where the 2nd parameter indicates the L1 starting cycle & that the target "Mission" and "SCI" are found in that cycle. I'll reiterate that the source is the L1 directory tree. It is assumed that every file that exists in L1, from a previous run of this code, will have the "MIssion" and "SCI" fully listed. There are checks to best implement the parameters provided. For now, the safest is to run all cycles ..however as has been pointed out, running only a subset would be beneficial for time/CPU issues. My limited testing is that what I've done here works, but it is only on a very limited dataset. The final goal arrive at the same output no matter what the cycle number the code starts with.

The code writes the new files into a ./data/4019/L1 directory, alongside the previous json directory. I make no changes to the original files. (E.g. <INTERNAL_ID>_<TRANSMISSION_ID>_L1_CYCLE.json)

Without the “Mission” and “SCI_Parameters” variables in cycle -1 of the L0, the output of these variables in later cycles can be wrong. In the case of floats, that did NOT send these values in -1, I recommend creating a json of cycle -2 (e.g. …L1_-02.json) that contains the “Mission” and “SCI_Parameters” variables from the float. Place it into the floats L1 data directory. This can be filled by hand, or copied from another float that was set up in the same way. The L0toL1.cpp code will read this -02 file, if it is present. I felt this was a better solution than modifying the -01.json created by Ben’s code.

If the code is run on a float that does NOT have a full “Mission” and/or “SCI_Paramters” json from pre-deployment, the later cycle “Mission” and “SCI_Parameters” will be incomplete. In my testing the code still runs without the full values, but again the result is likely to be incorrect.

In the rest of this document I will show some examples…

FILE_UPDATE_DATE:
"FILE_CREATION_DATE": "2025-08-23T00:46:56Z",
"FILE_UPDATE_DATE": "2025-09-05T21:21:58Z",
"DECODER_VERSION": "0.154",

“Mission”[“Ice_Mn”]: Indicating that there is no algorithm onboard (or should I remove these fully?)

```
"Ice_Pd": { "value":     40, "unit": "dbar", "description": "Ice-Algorithm Deep IceTc limit" },
"Ice_Ps": { "value":     20, "unit": "dbar", "description": "Ice-Algorithm Shallow IceTc limit"
"Ice_Mn": { "value":     -1, "unit": " hex", "description": "Enable Ice-Algorithm by month" },
"Ice_Tc": { "value":  -165, "unit": "degC*100", "description": "Ice-Algorithm ml temperature
"Ice_Sc": { "value":     20, "unit": "    1", "description": "Ice-Algorithm number of scans" },
```

```
"SCI_Parameters": {
    "ctd": {
    "Enabled": -1,
    "EnDrift": -1,
    "DecDrift": -1,
    "DataType": -1,
    "PackType": -1,
    "gain": -1,
    "offset": -1,
```

"region1": {"zMin": -1, "zMax": -1, "dz": -1, "scanT": -1},
"region2": {"zMin": -1, "zMax": -1, "dz": -1, "scanT": -1},
"region3": {"zMin": -1, "zMax": -1, "dz": -1, "scanT": -1},
"region4": {"zMin": -1, "zMax": -1, "dz": -1, "scanT": -1},
"region5": {"zMin": -1, "zMax": -1, "dz": -1, "scanT": -1}
},
"DO": {
"Enabled": 1,
"EnDrift": 100,

Utilized CONFIG Sgain, Tgain, and Pgain to adjust the output resolution of the CTD data. Also shows the inclusion of fillvalue for PRES/TEMP in this cycle, as it wasn't transmitted.

{ "PRES": -999, "TEMP": -999, "PSAL": 32.501 },

Instead of…

{ "PRES": -999, "TEMP": -999, "PSAL": 32.5010 },

The following variables were not transmitted, but the CONFIG says they should have been collected. Thus empty variables are included in L1.

"DO_Surface": [
],
"DO_Drift": [
],

Some channels of BGC data were not sent: The L1 file includes fill values for those channels.

"pH_Discrete": [
{ "PRES": -999, "VRS_PH": -0.867999, "VK_PH": -999, "IK_PH": -999, "IB_PH": -999 },
{ "PRES": -999, "VRS_PH": -0.868351, "VK_PH": -999, "IK_PH": -999, "IB_PH": -999 },
{ "PRES": -999, "VRS_PH": -0.868836, "VK_PH": -999, "IK_PH": -999, "IB_PH": -999 },

GPS shifting. In L0 the GPS for cycles N and N+1 look like this…

"GPS": [
{ "description": "GPS_START", "TIME": "2024-12-14T04:54:00Z", "LATITUDE": 41.58376, "LONGITUDE": -125.90180, "HDOP": 1.6, "sat_cnt": 4, "snr_min": 31, "snr_mean": 39, "snr_max": 43, "time_to_fix": 70, "valid": -2 },
{ "description": "GPS_END", "TIME": "2024-12-24T02:48:00Z", "LATITUDE": 41.49570, "LONGITUDE": -125.79582, "HDOP": 1.7, "sat_cnt": 3, "snr_min": 27, "snr_mean": 34, "snr_max": 43, "time_to_fix": 600, "valid": -2 }
],
"GPS": [
{ "description": "GPS_START", "TIME": "2024-12-24T03:00:00Z", "LATITUDE": 41.49862, "LONGITUDE": -125.79861, "HDOP": 2.2, "sat_cnt": 4, "snr_min": 27, "snr_mean": 31, "snr_max": 36, "time_to_fix": 250, "valid": -2 },

{ "description": "GPS_END", "TIME": "2025-01-03T00:47:00Z", "LATITUDE": 41.10981, "LONGITUDE": -125.93177, "HDOP": 2.7, "sat_cnt": 4, "snr_min": 44, "snr_mean": 44, "snr_max": 45, "time_to_fix": 140, "valid": -2 }
],

In L1 cycle 40 looks like this…
"GPS": [
{ "description": "GPS_END", "TIME": "2024-12-24T02:48:00Z", "LATITUDE": 41.49570, "LONGITUDE": -125.79582, "HDOP": 1.7, "sat_cnt": 3, "snr_min": 27, "snr_mean": 34, "snr_max": 43, "time_to_fix": 600, "valid": -2 },
{ "description": "GPS_START", "TIME": "2024-12-24T03:00:00Z", "LATITUDE": 41.49862, "LONGITUDE": -125.79861, "HDOP": 2.2, "sat_cnt": 4, "snr_min": 27, "snr_mean": 31, "snr_max": 36, "time_to_fix": 250, "valid": -2 }
],

CONFIG that were implemented from L0 to L1
"DO": { (sample SCI_Parameter)
"Enabled": 1,
"EnDrift": 100,
"DecDrift": 1,
"DataType": 0,

“Enabled” : If a BGC sensor is turned off, the data is not added to the L1

Note: The BGC drift measurements are completely independent of the CTD drift measurements. There can not be any more BGC drift then CTD drift measurements, but there can be less in a couple ways a) starting later in the drift mission or b) decimation.

“EnDrift” : If a BGC sensor has drift turned off, it is not added to the L1 (EnDrift = 0)
If a BGC sensor starts measurements during drift after the CTD, fillvalue are added to match the CTD drift measurements (Endrift < 100)
“DecDrift” : If a BGC sensor decimates the drift measurements, fillvalues are added.

The net result of DecDrift and EnDrift should result in the CTD_Drift variable and the BGC_Drift variables being the same length in L1, whereas in L0 the BGC length will be <= to the CTD length.

“DrfDat” : If CTD “DrfDat” is not set, the no CTD_Drift is reported. (shouldn’t be transmitted either)
“SndBak” : Excludes a wide range of CTD data being transmitted.

“sSendPr” : Controls whether PRES is returned by the BGC sensors. If no PRES is sent, the PRES channel is removed.

“phDec” : Controls the decimation of “IB” and “IK” pH channels. The code makes it so that the length of “IB” and “IK” are the same as the other channels in L1 (How is it handled in L0?)