Environmental Sample Processor Contextual Sensors

5/22/14 Brent Roman brent@mbari.org

MBARI

Supported Instruments

- Can => internal environmental sensors within ESP core's housing
  - Temperature, humidity, pressure, battery voltage, amperage
  - Updates every 10 minutes as long as ESP application runs

- CTD => Seabird SBE 16plus V2 interfaced via RS-232 sensor 1
  - Temperature, pressure, conductivity, plus *optional*...
  - Fluorometer, Transmissometer, Oxygen Sensor (1 of 2 types)

- ISUS => one of two types interfaced via RS-232 sensor 2
  - Concentration of nitrate and, optionally, bisulfide
  - Support for all manufactured at MBARI
    - Some later models from Satlantic (in use at WHOI)

- TBD = Something new can yet be interfaced as RS-232 sensor 3
  - Note: this port is not currently wired to lid of the can

Polling Contextual Sensors

- Trickier than it would first seem
  - ISUS must synchronize with CTD to receive timely updates
  - Sample rate optionally quickens during sampling
  - Multiple threads may not access instruments simultaneously
  - The Can's internal sensor polling is controlled independently

- Code is in Polling object in mission/skeleton.rb
  - Polling.start #starts SensorPolling with new parameters
  - Polling.stop #stops polling and properly closes instrument files
  - Polling.pause #stops until resumed
  - Polling.resume #resumes previous polling schedule if paused

- Instrument shows last sampled state of all Instruments
  - CTD, ISUS, Can show last sampled state of each Instrument

Internal Environmental sensors

- can is short for Sleepy.queryCan --> forces immediate sampling
  - can.temperature => internal temp. at top of can in degrees C
  - can.humidity => humidity in % of saturation
  - can.pressure => internal pressure in psia
  - can.voltage => instantaneous battery voltage
  - can.current => instantaneous battery load in amps
  - can.avgCurrent => averaged battery load in amps
  - can.waterAlarm => percent “wet” (0..100) usually < 1
  - Wattage is merely can.current * can.voltage
- Sleepy.can accesses most recent sample
  - Typically updated every 10 minutes
  - Recorded in binary 'real.log' file

  $ dumplog @object.is Can #will list Can environment samples
- Sleepy.canPollInterval = desired update rate for Sleepy.can

  Sleepy.canPollInterval = Delay.new “7:00” #change rate
  - Set to zero to disable can environmental sampling entirely
    - Zero is the default for MFBs lacking Sleepy board.

Seabird CTD

- Seabird 16plus V2 CTD with
  - support for fluorometer, transmissometer, oxygen sensor, ...
  - Generates file CTD-*.hex of raw samples
- CTD.status # shows instrument status
- CTD.pumpmode = *mode*, where *mode* is either:
  - :off, :beforeSample, or :duringSample
- s = CTD.sample => returns sample object, assigns it to variable s
  - s.temperature => sea temperature in degrees C
  - s.conductivity => conductivity in S/m
  - s.pressure => pressure in decibars
  - s.transmissometer => % optical transmission
  - s.beamAttenuation => extinction coefficient in 1/m
  - s.sampleTime => time at which this sample was started
  - s.dataTime => time at which this sample was finished
  - s.depth => depth in meters (derived from pressure)
  - s.salinity => salinity in mythical PSUs
- More documentation in lib/instrument/ctd.rb

ISUS

- ISUS = In-Situ Ultraviolet Spectrometer
  - Stores raw spectra in ISUS-*.dat (MBARI's ISUS only!)
  - Logs errors in ISUS-*.err
  - ***Requires temp., salinity & depth from the CTD !!***
- ISUS.status # shows instrument status
- ISUS.species = 2 (or 3) #three to include bisufide
- ISUS.fit = 217..240 #spectral fit window in nm (tweak for species)
- ISUS.fromCTD temp, salinity, depth #update ISUS from CTD
- s=ISUS.sample => sample with most recent values fromCTD
  - s.no3 => Nitrate concentration in uM/L
  - s.br => Bromide in uM/L
  - s.hs => Bisulfide in uM/L (only valid if species>2 and fit tweaked)
  - s.sampleTime => when sample was requested
  - s.dataTime => when sample was recorded
- More documentation in lib/instrument/isus.rb

Parameters controlling Contextual Sensor Polling

- *$global* variables determine instruments' configuration/polling rates
- These may be assigned anytime before Polling.start
  - But, usually they get set once in mission/phasecfg.rb
  - Missions with :until=>*time* automatically invoke Polling.start
- CTD
  - $ctdPumpMode=:duringSample #may be :beforeSample or :off
  - $ctdInterval=Delay.new “5:00” #sample CTD every 5 minutes
  - $ctdPeriod=Delay.new “1:00:00” #upload CTD data every hour
  - $samplingCTDinterval=Delay.new “2:30” #2x faster ...
  - $samplingCTDperiod=Delay.new “30:00” # while sampling
- ISUS
  - $isusSpecies = 2 #ignore sulfides by default (3 to include them)
  - $isusFit =217.240 #because Luke says it should be so :-)
- ISUS polling rate is CTD sampling rate + 10 minutes
  - ISUS auto-sampling cannot be disabled