Communications

**Stream**

* 6 packets of data in the range of [12B – 20B] each including addressing, checksum, etc.
* Cycles through each packet so that each dataset is updated every 50msec
  1. Shelf Info – 16 Bytes
     + Total Shelf Volt - Int 2B
     + Shelf Avg. Temp - Int 2B
     + Shelf Max Temp - Int 2B
     + Shelf Min Temp -Int 2B
     + Shelf Max Temp ID - Int 2B
     + Shelf Min Temp ID - Int 2B
     + Shelf Alarms - Int 2B
     + Shelf Errors - Int 2B
  2. Current Module Info – 8 Bytes
     + Current Mod - Int 2B
     + Mod Voltage - Int 2B
     + Mod Temp - Int 2B
     + Mod Errors - Int 2B
  3. Conv. Voltages – 12 Bytes
     + Set Point - Long 4B
     + DC Bus - Long 4B
     + ESD Bus - Long 4B
  4. Conv. Currents – 16 Bytes
     + Source - Long 4B
     + Load - Long 4B
     + Conv. DC - Long 4B
     + ESD - Long 4B
  5. Conv. Temperatures – 16 Bytes
     + Temp 1 - Int 2B
     + Temp 2 - Int 2B
     + Temp 3 - Int 2B
     + ...
     + Temp 8 - Int 2B
  6. Conv. Status/Message Counts
     + TBD – Waiting for more info on Converter status/modes/etc.
     + Expecting ~16 Bytes

**2 Means of creating data stream**

Initiate stream on BMS (based on ProD clock cycles)

Pros:

* More consistent timing
* Easy to initiate and “straightforward”

Cons:

* Hard to reinitiate once stream is dropped without purposefully stopping it
* Overhead is now on BMS side which is responsible for the entire system

Initiate Stream in GUI (based on MS .Net System.Timers.Timer object)

Pros

* Easy to reinitiate/manage stream while streaming
* Easy to change message lengths/structure in the future

Cons

* Less consistent timing due to Timer (reportedly +/- 1msec deviation)
* Must check/guard against garbage collection in .Net engine

**Current Setup** – GUI Stream using Timers

**Reason** – More flexible. It is much easier to pick up and drop the stream, and simpler to change the structure of the communication protocol in the future if needed (and almost every project does as it grows). The Timers do have a reported deviation of about 1msec but seeing as how everything in the GUI is refreshed at 50msec, this seems like a small price to pay for the gains. This Timer does not affect the time for transmitting/receiving commands to/from the BMS/GUI. For example, if the stream is enabled and the user changes the current module information to be displayed. As soon as the user clicks on the new module’s radio button, the command to “set [new/selected] current module” is sent to the BMS as quickly as possible, regardless of data stream. Commands like this are not part of the streaming protocol and do not have to wait for transmission/reception.