ALMA Station Electronics Details

Rich Lacasse

Atacama Large Millimeter/submillimeter Array
Karl G. Jansky Very Large Array
Robert C. Byrd Green Bank Telescope
Very Long Baseline Array

Outline

- Purpose of the Station Electronics
- Architecture
  - Astronomical Signal Flow through the Station Electronics
  - Monitor and Control of the Station Electronics
- Programming
  - Normal sequence
  - Some details on 3 protocols
  - Bottlenecks and possible improvements

Purposes of the Station Electronics

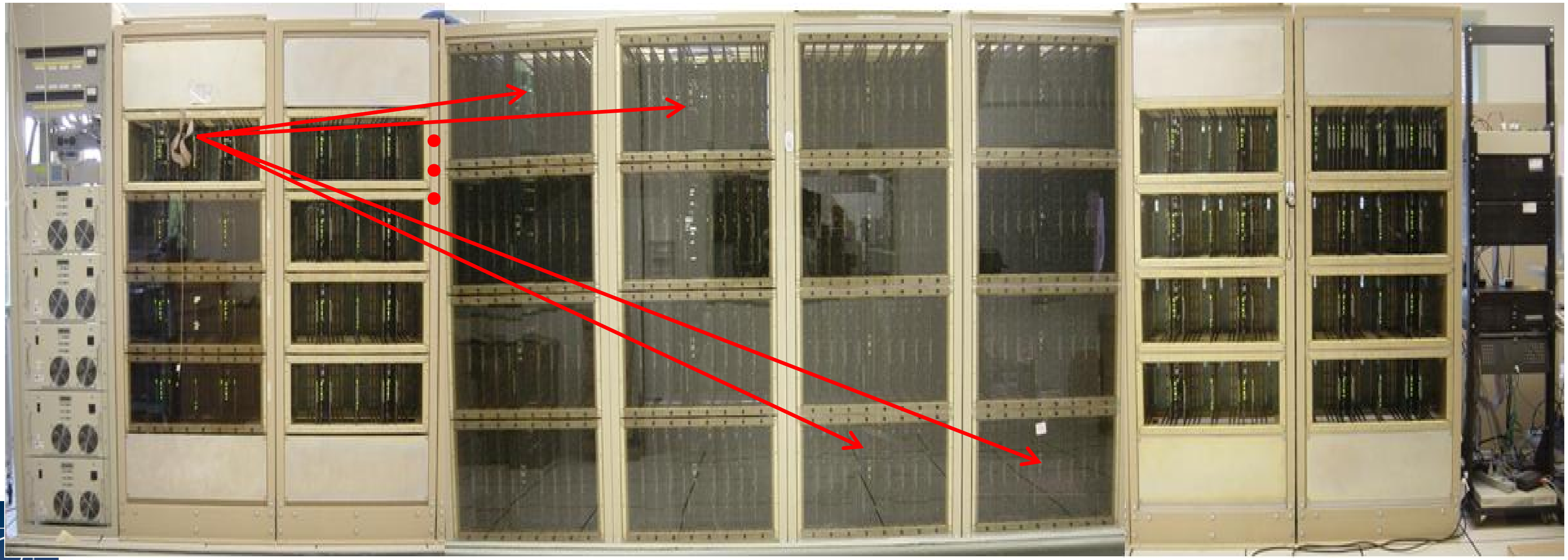
- In the interest of time, avoid excessive detail!
- Do everything that needs to be done on a ***per antenna*** basis, primarily:
  - Delay compensation, coarse and bulk (250 ps resolution)
  - Filtering (FDM only)
    - 1, 2, 4, 8, 16 or 32 filters
    - Filter BW of 62.5 or 31.25 MHz
    - Center frequency of each filter.
  - Mode generation
    - Shuffle and delay samples to enable 67 correlator modes

Architecture: Station Rack Signal Flow

- Inputs: Data from antennas, control from CCC
- Outputs: Data to correlator matrix, monitor to CCC

Data Flow

CAI 0:15 CAI 16:31 CAI 32:47 CAI 48:63

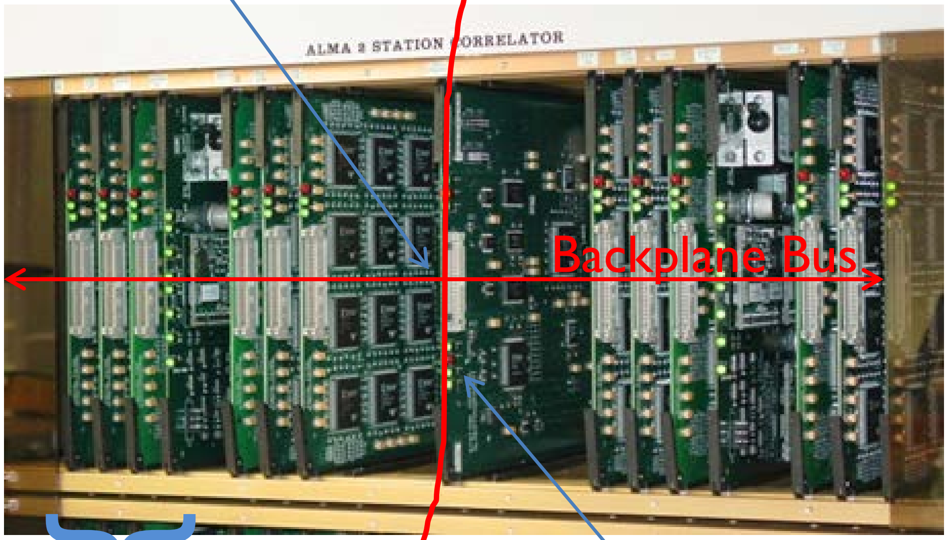


Architecture: Station Rack Signal Flow

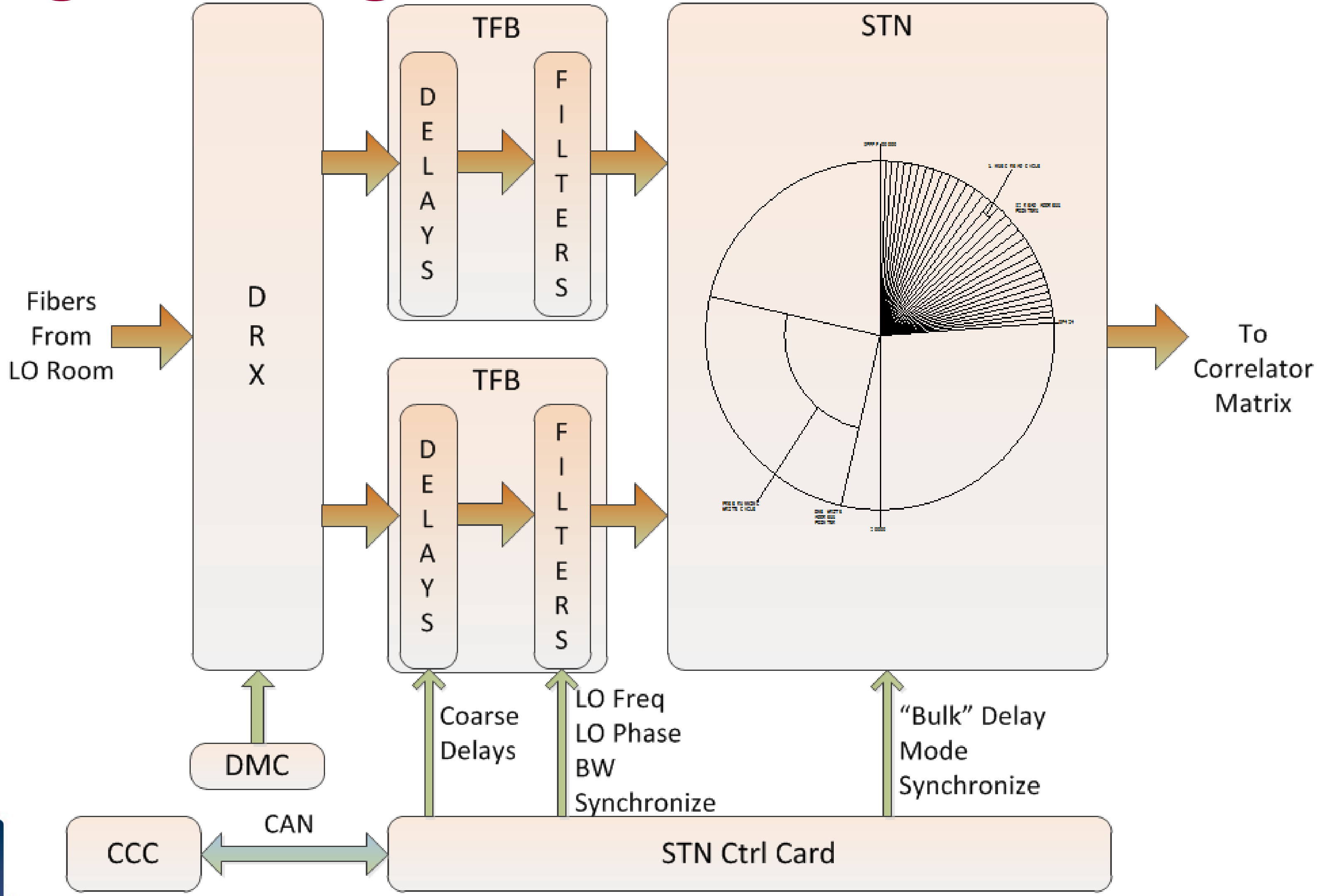
- Station rack to correlator rack interconnects: 1024 cables/quad, 1 Tb/s

Architecture: Monitor and Control

- One Control Card, 16 “logic” cards, power distribution cards



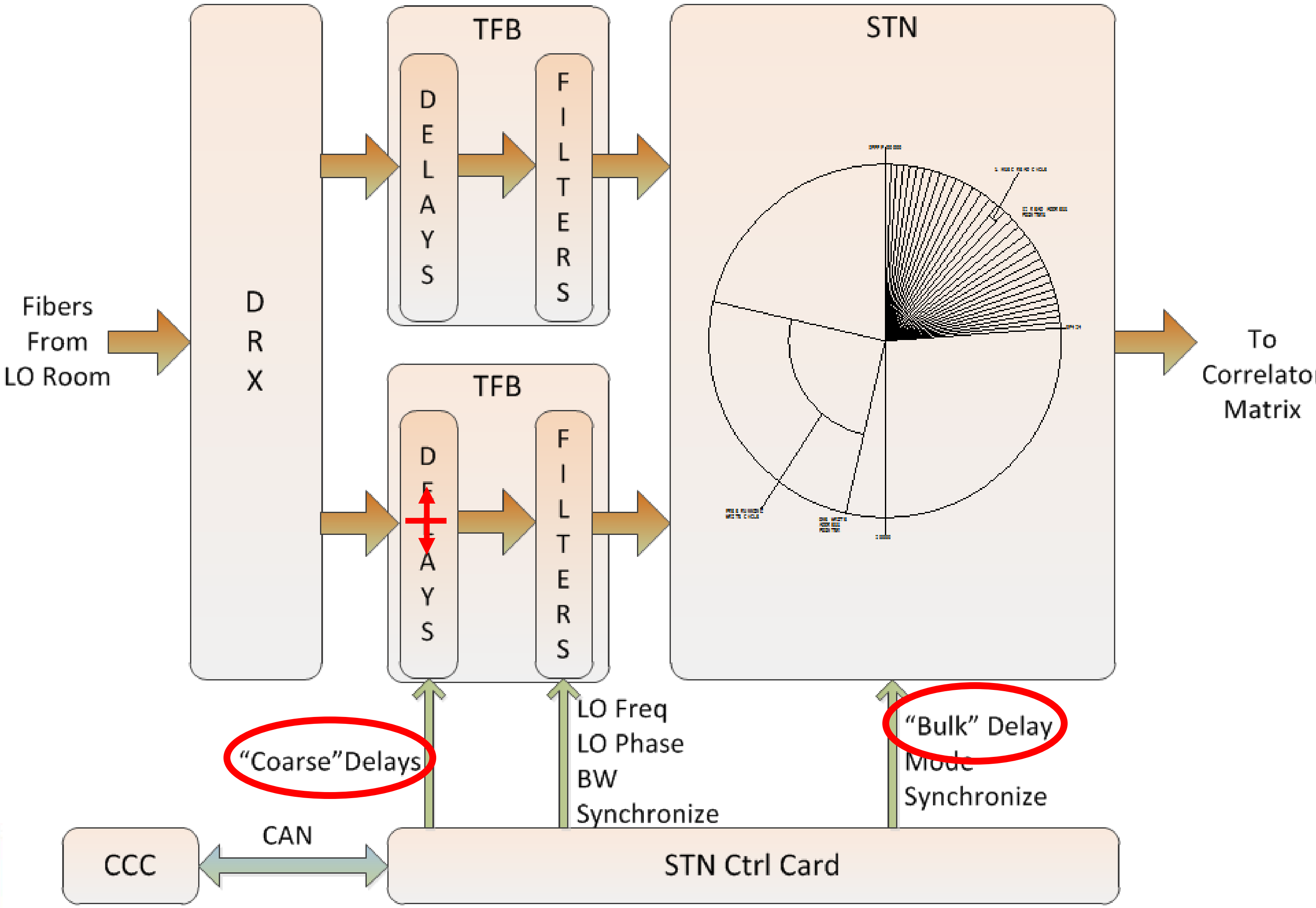
Programming: Overview



Programming: Delay CAN Protocol

Message Payload Bytes	**Reply Message Contents** **Message RCA 0x20401**
Data[0]	**spare**
Data[1]	**Delay value LS byte**
Data[2]	**Delay value next byte**
Data[3]	**Delay value next byte**
Data[4]	**Delay value MS byte**
Data[5]	**Delay Destination Mask**
Data[6]	**Relative 1 msec: 0 – 47**
Data[7]	**Error flags on GET (see below); Options on SET (see below)**

Programming: Delay Protocol, cont'd



Programming: Mode CAN Protocol

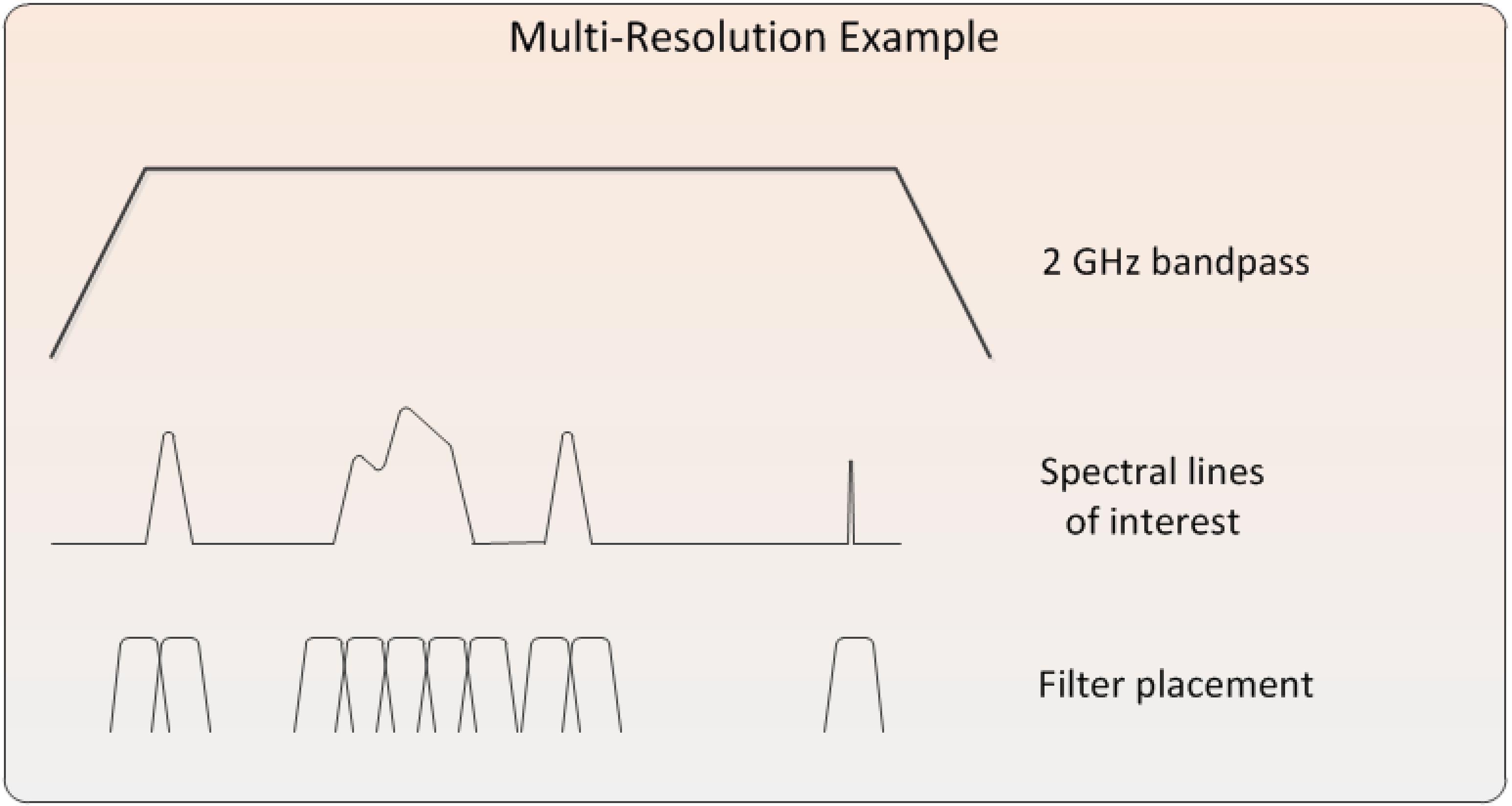
Data Msg 0 Msg ID 8	ta Msg 1 Msg ID 8	Data Msg 2 Msg ID 8	Data Msg 3 Msg ID 8	Data Msg 4 Msg ID 8	Data Msg 5 Msg ID 8	Data Msg 1027 (last msg) Msg ID 9

ConfigSetID **(0-15)**	cM[0]	cM[2]	caiMask[0]	dds[0][0][0]	…	dds[127][28][0]
numberOfModes	nP[0]	nP[2]	caiMask[1]	dds[0][0][1]	…	dds[127][28][1]
configSrc	1stP[0]	1stP[2]	caiMask[2]	dds[0][1][0]	…	dds[127][29][0]
0 (spare)	1stF[0]	1stF[2]	caiMask[3]	dds[0][1][1]	…	dds[127][29][1]
0 (spare)	cM[1]	cM[3]	caiMask[4]	dds[0][2][0]	…	dds[127][30][0]
0 (spare)	nP[1]	nP[3]	caiMask[5]	dds[0][2][1]	…	dds[127][30][1]
0 (spare)	1stP[1]	1stP[3]	caiMask[6]	dds[0][3][0]	…	dds[127][31][0]
0 (spare)	1stF[1]	1stF[3]	caiMask[7]	dds[0][3][1]	…	dds[127][31][1]

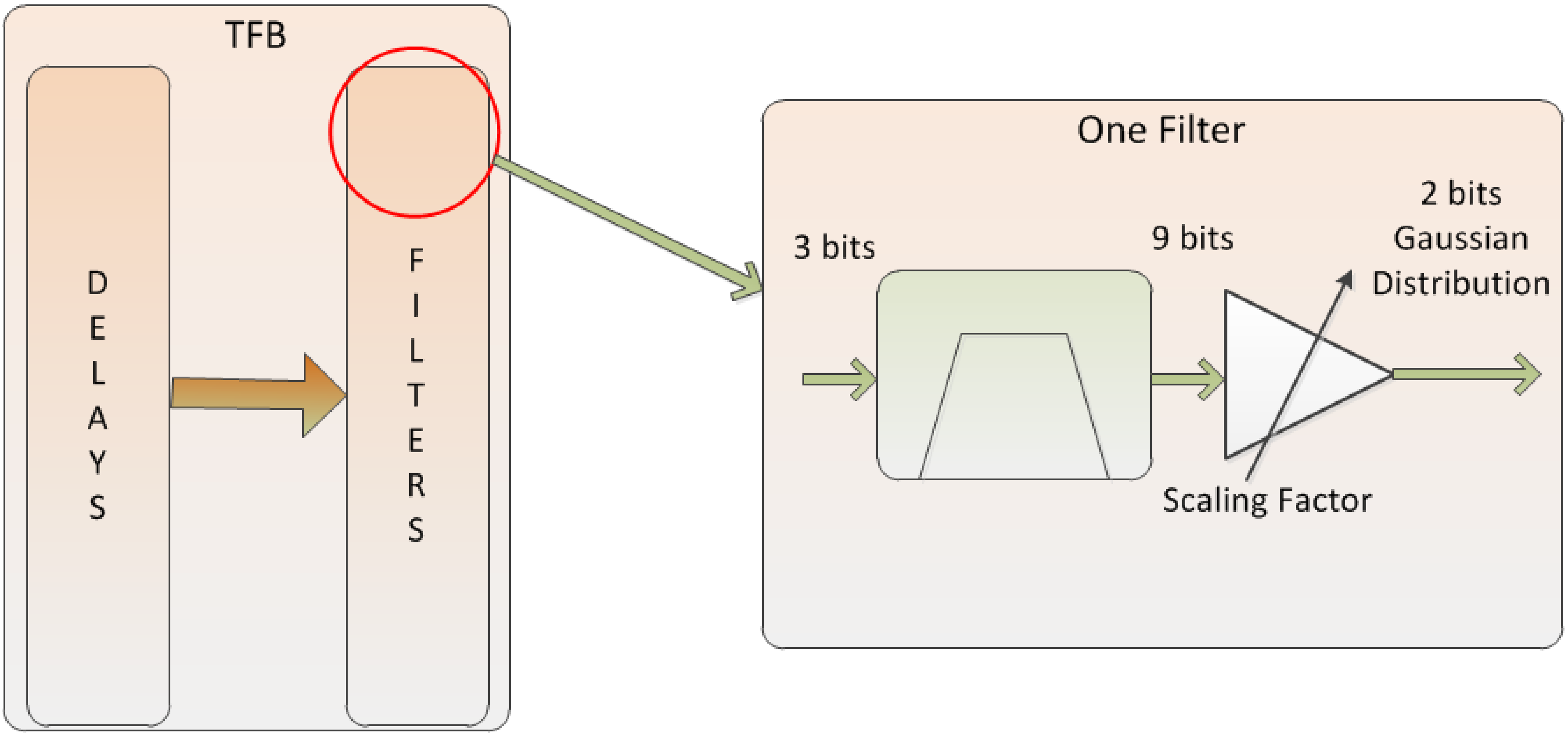
For Multi-resolution modes

For all 4096 filters in quadrant!

Programming: Mode CAN Protocol



Programming: Scaling Factors



Programming: Scaling Factors

– The protocol is simple:

Message Payload Bytes	**Broadcast Setup Message** Transmit using Message ID 5	**Header Message** Transmit using Message ID 8
Data[0]	**Data_msg_ID = 8**	**Func_code=10 (0xA)**
Data[1]	0 (spare)	Configuration Number (0 to 15)
Data[2]	Target mask 7 - 0	Scale Factor Set Number (0 to 15)
Data[3]	Target mask 15 - 8	spare
Data[4]	Target mask 23 - 16	spare
Data[5]	Target mask 31 - 24	spare
Data[6]	Target mask 39 - 32	spare
Data[7]	Target mask 47 - 40	spare

– The SCC must do a lot of work behind the scenes (floating point multiplies, divides and square-roots)

– Result is one 8-bit number per filter.

Programming: Normal sequence

(FDM only)

- Calibration
  - Download configuration (in uP memory only)
    - Which CAIs
    - Up to 4 modes (for multi-resolution) … (implemented?)
    - TFB frequencies (for all 128 cards in the quadrant)
  - Apply configuration (data changes during apply)
    - Which configs, when
  - Calculate scaling factor set
    - For which config, which set #
  - Get scaling factor set (to CCC)
    - For which set #, which antennas

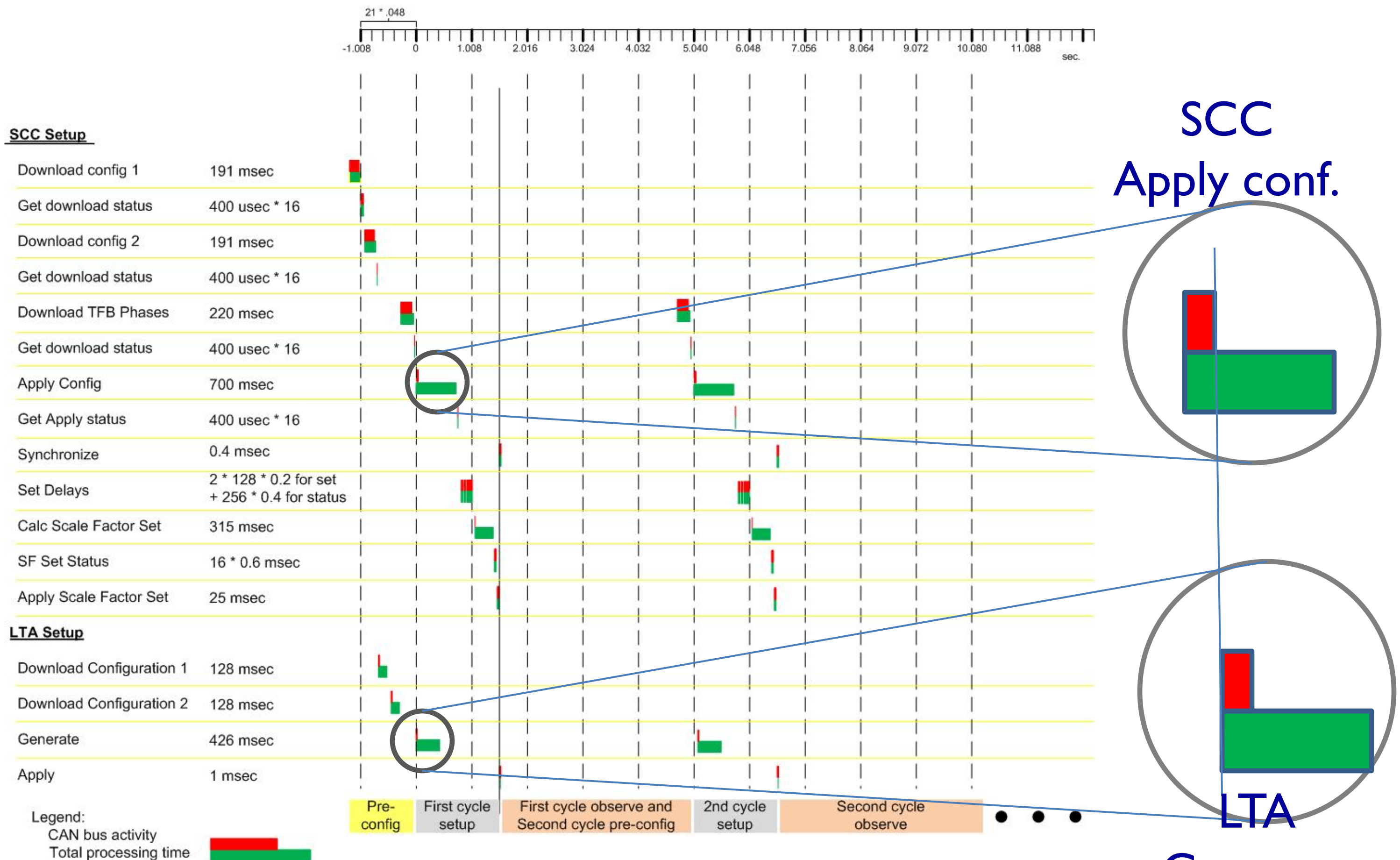
NRAO

Programming: Normal sequence

(FDM only)

– Subscan configuration
  - Download configuration (as in calibration step)
  - Download scaling factor set (to uP memory)
  - Apply configuration (data changes)
  - Apply scaling factor set (to hardware)
  - Download TFB Phases
  - Send "initialize" delays
  - Synchronize

Programming: Potential Improvements



Example of setting up one quadrant of the correlator for 64 antennas in mode 7 (all filters used), with 10-second 2-source observing.

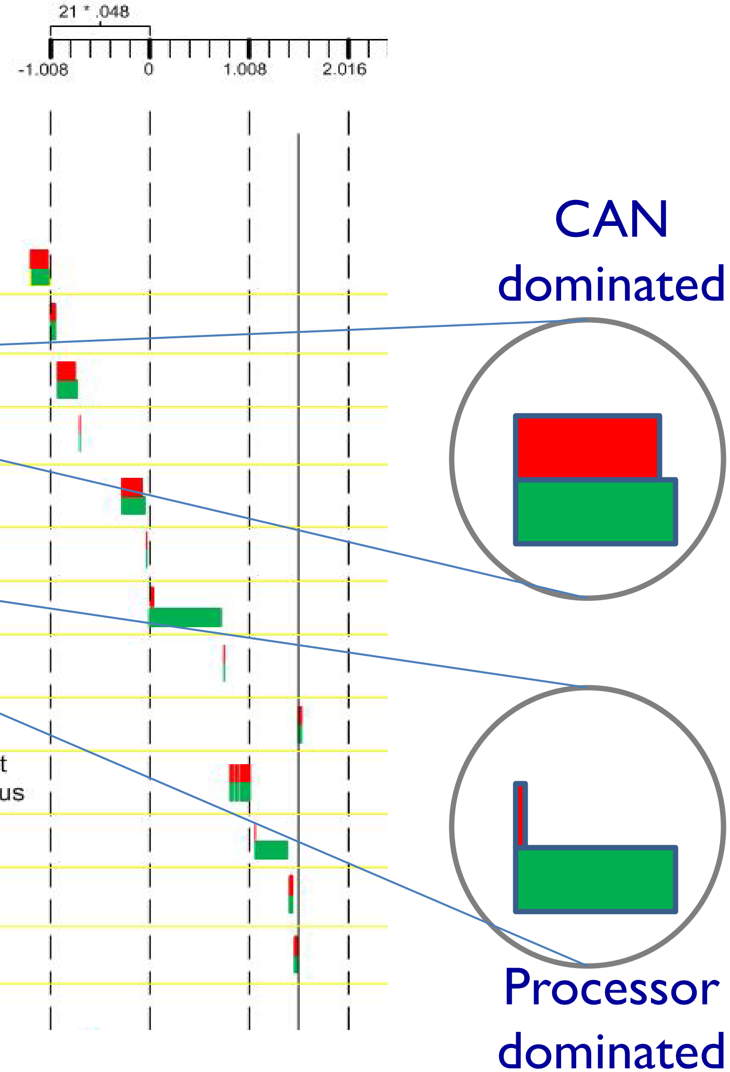
Programming: Potential Improvements

- More CANs (diagnostics disappear!)
- Smarter protocols

SCC Setup

Download config 1	191 msec
Get download status	400 usec * 16
Download config 2	191 msec
Get download status	400 usec * 16
Download TFB Phases	220 msec
Get download status	400 usec * 16
Apply Config	700 msec
Get Apply status	400 usec * 16
Synchronize	0.4 msec
Set Delays	2 * 128 * 0.2 for set + 256 * 0.4 for status
Calc Scale Factor Set	315 msec
SF Set Status	16 * 0.6 msec
Apply Scale Factor Set	25 msec

LTA Setup



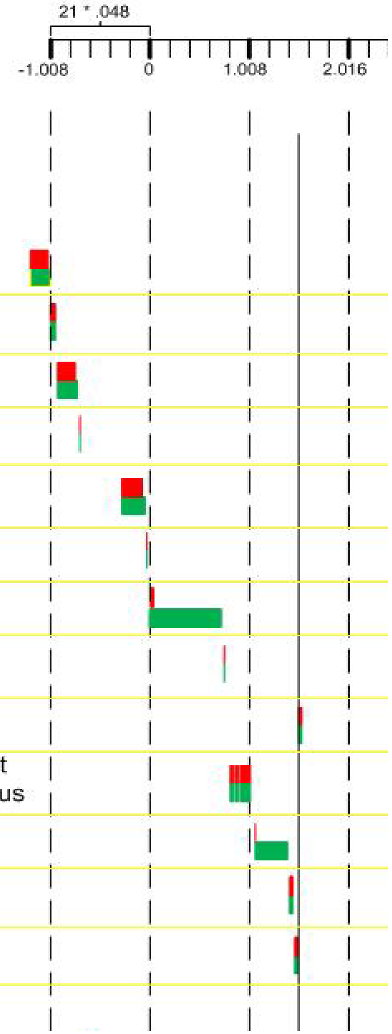
Programming: Potential Improvements

- New hardware uP
- Calculation offload

SCC Setup

Download config 1	191 msec
Get download status	400 usec * 16
Download config 2	191 msec
Get download status	400 usec * 16
Download TFB Phases	220 msec
Get download status	400 usec * 16
Apply Config	700 msec
Get Apply status	400 usec * 16
Synchronize	0.4 msec
Set Delays	2 * 128 * 0.2 for set + 256 * 0.4 for status
Calc Scale Factor Set	315 msec
SF Set Status	16 * 0.6 msec
Apply Scale Factor Set	25 msec

LTA Setup



Programming: Potential Improvements

- Calibration (ref. ALMA Memo 583 and ICT-1683)
  - Lag 0 of TDM observation is currently used as a proxy for total power – lots of overhead
  - Both total power and DC value of 3-bit input are available from the TFB. Could this potentially save lots of overhead? (For instance, these could be measured and transmitted with scaling factors with ~ 5 msec of overhead.
- Calculation offloading
  - Square roots and divides in CCC or FPGA
- Delay models executed in in FPGAs

Summary

- Purpose of the Station Electronics: delay and mode config
- Architecture: bin controller; 4 antenna's worth of processing per bin
- Programming
  - Normal sequence only; lots of details left out
  - Some details on 3 protocols: most "troublesome" ones
  - Bottlenecks and possible improvements:
    - CAN and processing power
    - smarter protocols;
    - smart sequencing of commands to allow parallel processing in the microprocessors,
    - off-loading of complex calculations to the CCC,
    - Getting total power from TFBs
    - redesign with more modern hardware at the expense of software effort.