PROTOCOL SPECIFICATION

HM Gbps Cable Signaling Specification

PROTOCOL Document: **A25022N0041**

Revision: 1.43

Brent Carlson, May 21, 2010

National Research Council Canada
Herzberg Institute of Astrophysics
Dominion Radio Astrophysical Observatory

P.O. Box 248, 717 White Lake Rd
Penticton, B.C., Canada
V2A 6K3

Table of Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

CRC-4 – 4-bit "Cyclic Redundancy Check". The CRC uses a linear feedback shift register (LFSR) to calculate a code that takes into account all input bits. The generator polynomial defines the LFSR configuration, and different generator codes have different error detection capability. A CRC-4 will catch about 95% of all types of errors.

Baseband (BB) – For the EVLA correlator, this refers to a 2 GHz bandwidth signal sampled with 3 bits at 4 Gs/s, or a 1 GHz bandwidth signal sampled with 8 bits at 2 Gs/s. When it comes to identifiers embedded in a data streams, this refers to a "64-bit data highway input" to a Station Board that carries 2 GHz or 1 GHz of bandwidth as defined above. Since one "station input" to the correlator is defined as 4 Station Boards, and each Station Board has two 64-bit data highways, then there are 8 BBs into each station input. In other contexts this refers to a sampled data stream (whatever format that may be) that came from one physical sampler (or, analog-to-digital converter).

Sub-band (SB) – Bandwidths produced by digital FIR filters operating on BBs.

HM Wafer – a 4-pair connector that plugs into one row of an 8-pin HM (hardmetric) header. Wafers can be stacked side-by-side to allow various wiring configurations in the correlator.

HM – Hard Metric, 2.0 mm connector system. This connector system is most often utilized in Compact PCI systems, and is an industry standard.

1 Revision History

Revision	Date	Changes/Notes	Author
DRAFT	April 13, 2004	Initial DRAFT release	B. Carlson
1.0	May 5, 2004	Specify that unused samples in the DATA/BB streams are filled with a '1010' pattern for maximum transition density on the Gbps line. First full release after implementation and testing of the Recirculation Controller lock circuitry.	B. Carlson
1.1	February 24, 2005	For 4-bit data, DATA invalid is specified as "5", and "-8" is the code for the data value of "5". This prevents DC bias on the line when long stretches of invalid data appear. This fix suggested by Dave Fort.	B. Carlson
1.2	August 18, 2005	Add the command '110', that forces the recirculation chip to reset to the start block the recirculation counter, without taking any dump action otherwise. Each PHASEMOD stream now only contains phase models for sampled data streams within the same wafer. This eliminates the need for a "master" Station Board to generate phase models for all sub-band outputs.	B. Carlson
1.3	November 5, 2007	Add COMMAND and STATUS streams to the CTRL mux data stream to allow for X-bar Board M&C via these lines. Remove some obsolete fluff like reference to the TGB, and the Phasing Board. Define SID all 1's wildcard to the DUMPTRIG frame.	B. Carlson

1.4	October 1, 2008	Add DATA chopper circuit, changes to TIMECODE protocol to support it. Clarify PHASERR use, and F-bit time sequence order. Correct error in PHASEMOD diagram Figure 5-9. Add chopper circuit details to the Appendix. Add X-bar Board COMMAND protocol to the Appendix.	B. Carlson
1.41	March 18, 2008	Update the "HM Gbps X-Bar COMMAND payload", Figure 7-4, to include an "ALL" switch command for CMD type 000. Update NOTES to include "PHASEMOD merge" function; PHASEMOD switches with BBs and are merged together for final wafer output. Update section 5.4.3 to include an additional note about PHASEMOD generation for Cross-Bar Board merging. PHASEMOD merging function is added to support dual-polarization in 8-bit initial sampling where BBs from different Station Board must end up in the same wafer out of the Cross-Bar Board.	B. Carlson
1.42	December 22, 2009	Add X-bar Board STATUS signaling payload protocol (section 7.5) to the appendix	B. Carlson
1.43	May 21, 2010	Add DumpTrig RRC (CMD=110) "PB" ID protocol, for tracing CTRL connectivity through the system. See section 5.4.2. Use CTRL nomenclature more consistently. Update Table 5-1 with established dump control pneumonics.	B. Carlson

2 Introduction

This document describes the signaling protocol for data and control signals traveling from the Station Boards to the X-bar Boards and the Baseline Boards on the 1 Gbps, source-synchronous transmission system. The companion document to this document and that describes the physical cable and pin assignments can be found in [1]. This document describes only the layer-2 protocol for the signals and data traveling on these lines.

3 Context

Figure 3-1 is a simplified diagram showing the connection of Station Boards to the X-bar Board, and on to the Baseline Board. The signaling described in this document, is the signaling present on these cables traveling between these boards with RX and TX functions as shown.

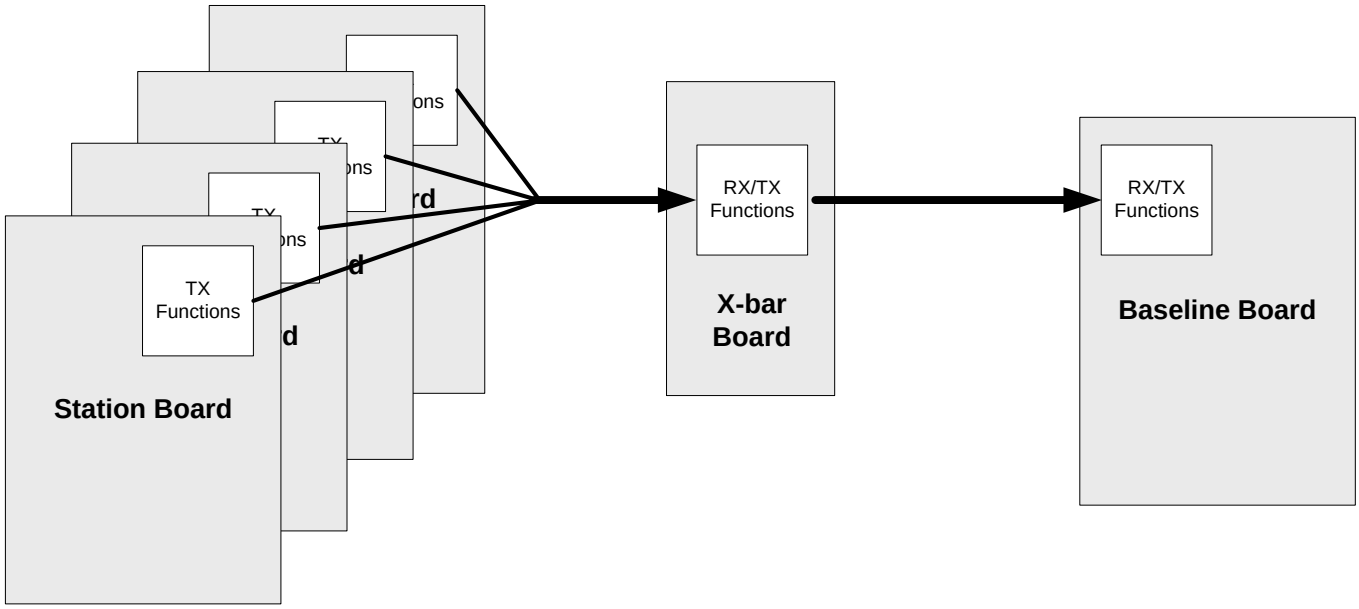


Figure 3-1 Simplified diagram showing the connection of the Station Boards to the X-bar Board, and on to the Baseline Board.

4 Overview

A simplified block diagram of basic transmitter and receiver functions that this protocol specification addresses is shown in Figure 4-1.

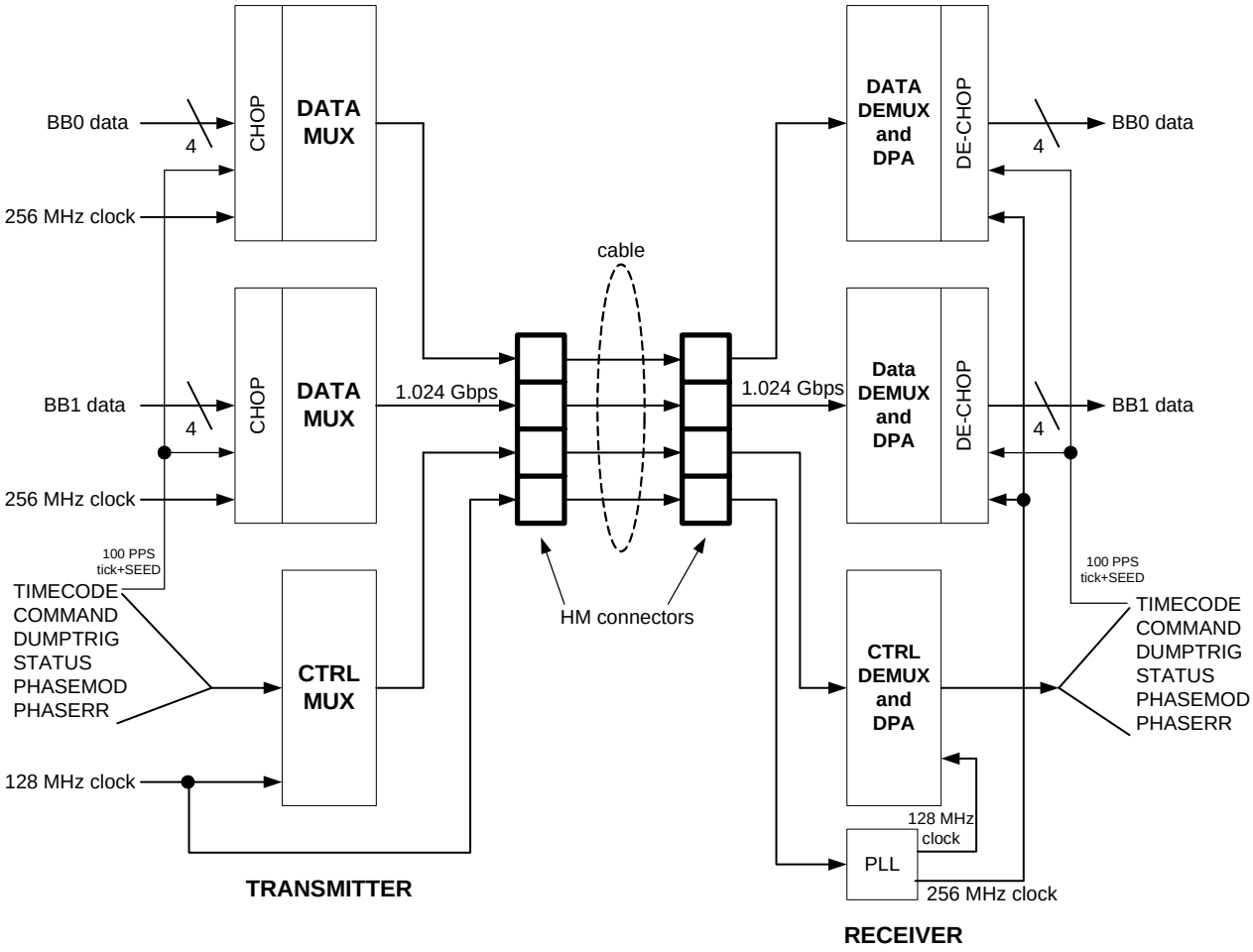


Figure 4-1 Simplified block diagram of basic transmitter and receiver functions that this protocol specification addresses. The "HM connector" shown is a four-pair "wafer" containing two sampled data streams, a control stream and a clock. As shown in Figure 3-1, each wafer is sourced from a different Station Board, and a stack of four wafers terminates at X-bar Boards and Baseline Boards.

The transmitter "chops" and then multiplexes 4-bit data operating at 256 MHz to 1 serial stream operating at 1.024 Gbps. The transmitter also multiplexes the control signals operating at 128 Mbps to 1 serial stream operating at 1.024 Gbps. A 128 MHz clock is sent with the 1.024 Gbps signals to be used for recovery using DPA (Dynamic Phase Alignment[1]). At the receiver, dedicated circuitry, as well as programmable logic works

[1] Altera FPGA capability and terminology.

to de-multiplex, de-chop, and present the data and control signals as they were before the MUXes at the transmitter.

Since the PLL at the receiver is operating on a 128 MHz clock signal, there is no need for 8B/10B encoding[2] of the input signals to maintain 1's density on the Gbps signals—however during testing it was determined that to maintain a certain level of 1's density and to prevent 0/1 disparity in external AC-coupled signal transmission, a chopper circuit was required, which randomizes the data before multiplexing to 1.024 Gbps. This chopper ensures that even if a constant or slowly changing signal is present before the DATA MUX, 1.024 Gbps line conditions and DPA receiver 1's density requirements are met[3].

During testing it was also found that in conditions where many or all signals into/out of[4] an FPGA containing a DPA transmitter or receiver, are doing the same thing (i.e. all switching in the same direction at the same time), that it produced excessive FPGA clock jitter, which caused operational problems for downstream receivers. As such, a scheme involving the chopper circuit wherein each output or input uses a different 8-bit chopper "seed" was developed, which deals with this problem satisfactorily. Chopper seeds for DATA streams ("BB0", "BB1") within a particular wafer are encoded and sent in the TIMECODE stream contained within that wafer. Thus, the receiver always knows which seed to use in the de-chopper, without software intervention.

[2] And, no bandwidth available on the connectors or 1 Gbps links for 8B10B encoding.

[3] Except in the case where the signal itself contains the identical chopper signal—a vanishingly small probability of occurrence.

[4] It was not possible to determine if this was an internal FPGA noise problem causing clock jitter, or an SSO I/O problem. What is known, is that it is not a board-level VPLL or VIO or VCORE problem.

5 Protocol

This section defines the signaling protocol on the data and control lines that make up the 1 Gbps transmission system.

5.1 Chopper and Data Multiplexer

The Chopper and Data Multiplexer consist of three stages. The first stage is the chopper circuit, which contains an 8-bit PRN (pseudo-random number) generator, which performs a bit-wise exclusive-OR of the INPUT data with a 4-bit combination of the PRN code. The second stage is a 1:2 de-multiplexer necessary to convert 4-bit-wide data operating at 256 Mbps to 8-bit wide data operating at 128 Mbps. This is because the specialized second-stage Gbps multiplexer (the third stage) in the chip operates on data 8 bits wide up to 128 Mbps. This three-stage process is shown in Figure 5-1.

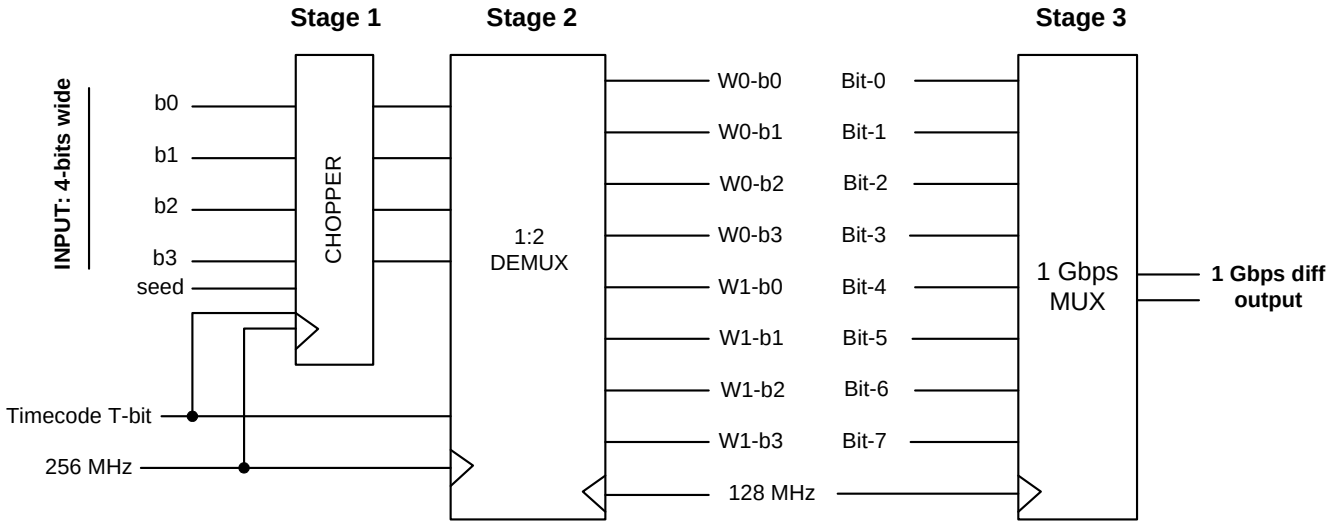


Figure 5-1 Three-stage data chopper and multiplexer. The first stage is a 4-bit chopper, which randomizes the data. The second stage is a 1:2 de-multiplexer to convert 4-bit data at 256 Mbps to 8-bit data at 128 Mbps. The third stage converts the 8-bit data to 1.024 Gbps.

The bit assignments in Figure 5-1 are important to note. W0 and W1 are the even and odd 4-bit words from the 256 Mbps data stream. A timing diagram of this word assignment, relative to the reference TIMECODE T-bit is shown in Figure 5-2.

Note: The chopper circuit operates on the DATA stream after embedded ID insertion and CRC-4 calculation of Figure 5-5; at the receiver, the de-chopper operates before CRC-4 calculation. Thus, DATA alignment and CRC-4 calculation is end-to-end and any mixup in chopper circuit synchronization or seed will be easily detected.

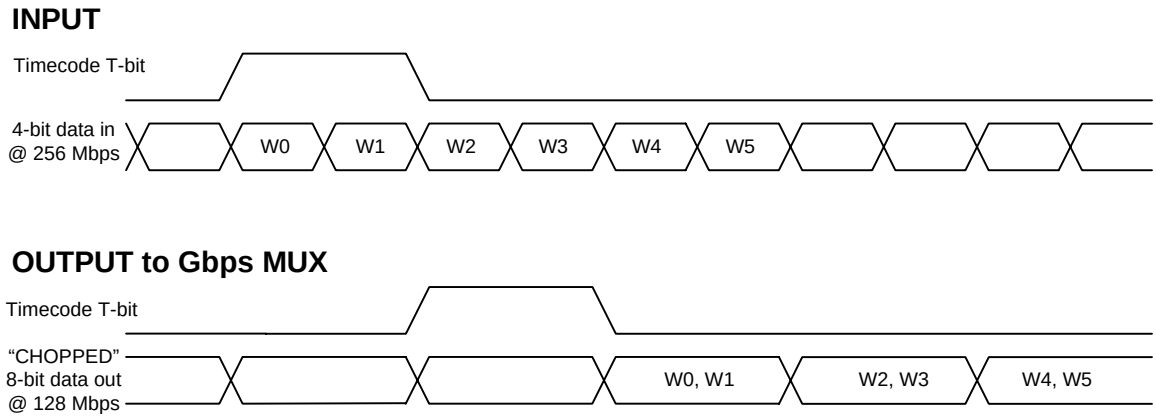


Figure 5-2 256 Mbps and 128 Mbps word assignments.

Note that as shown in Figure 5-2 and Figure 5-5, W0 is the first 4-bit number defining the data-stream number within the 4-bit wide data stream, and W1 is the second 4-bit number defining the data stream number within the 4-bit wide data stream. Together, and with the CRC check code, these two words allow the receiver to sort out the correct data stream numbers within the 4-bit wide data stream. This encoding is needed because the 1 Gbps de-multiplexer at the receiver has no concept of bit alignment within the 8-bit byte. This must be done by logic in the FPGA. Note that W0 is the actual time of the 1 second epoch.

The chopper and de-chopper must always operate on DATA with the input Timecode T-bit (100 PPS tick) aligned to W0 and W1 as shown in the upper half of Figure 5-2, whether the Timecode C-bit is set or not (i.e. independent of whether the DATA stream contains embedded IDs for this particular 100 PPS tick or not). Not doing so will result in the inability of the receiver to synchronize to the signal[5]*. Also, the chopper PRN circuit is synchronously pre-set every Timecode T-bit.*

[5] Although, normally the receiver is built to have the ability to search for some time-skew between the DATA and the Timecode T-bit.

5.2 Control (CTRL) Signal Multiplexer

The control signal multiplexer multiplexes the control streams (TIMECODE, PHASEMOD, PHASERR, DUMPTRIG) and a COMMAND and STATUS channel into a single 1.024 Gbps data stream. Since the control signals are already operating at 128 Mbps, there is no need for a two-stage multiplexing process as for the data just described. A diagram of the control stream multiplexer, complete with bit assignments is shown in Figure 5-3. Take careful note of control stream-to-input bit assignments.

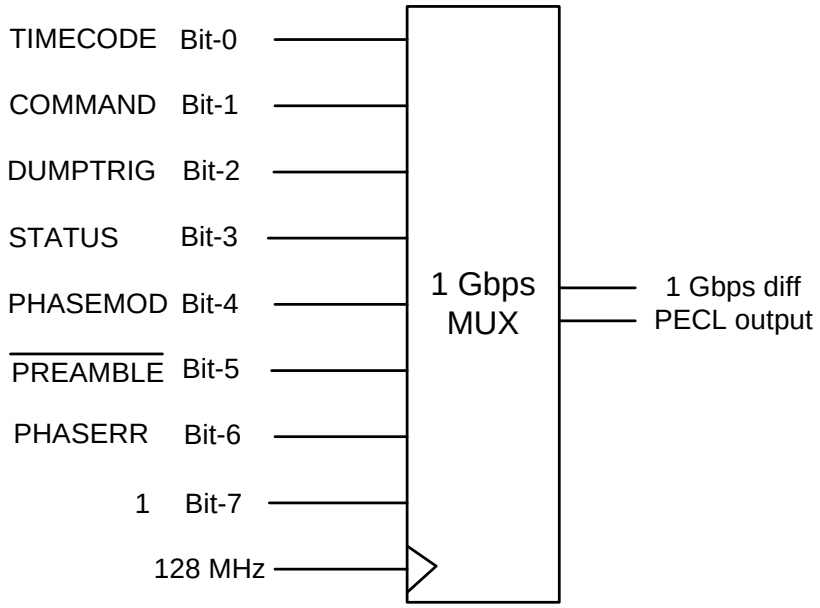


Figure 5-3 Bit assignments for the CTRL stream multiplexer. Bit-7 is always a 1, to allow the receiver to easily unravel which bit stream is which.

One of the input streams is the inverted preamble ($\overline{\text{PREAMBLE}}$), the inverse of that which is present in TIMECODE, DUMPTRIG, and PHASEMOD when no information is being transferred. To obtain good transition density on the line, needed by the DPA receiver for clock phase alignment, the COMMAND and STATUS channels have the same bit phase as $\overline{\text{PREAMBLE}}$. The Bit-7 input is always a 1 and is present to easily and automatically allow the receiver to establish control stream bit alignment within the 8-bit word. A functional timing diagram showing the relationship of the control bitstreams and inverted preamble is shown in Figure 5-4.

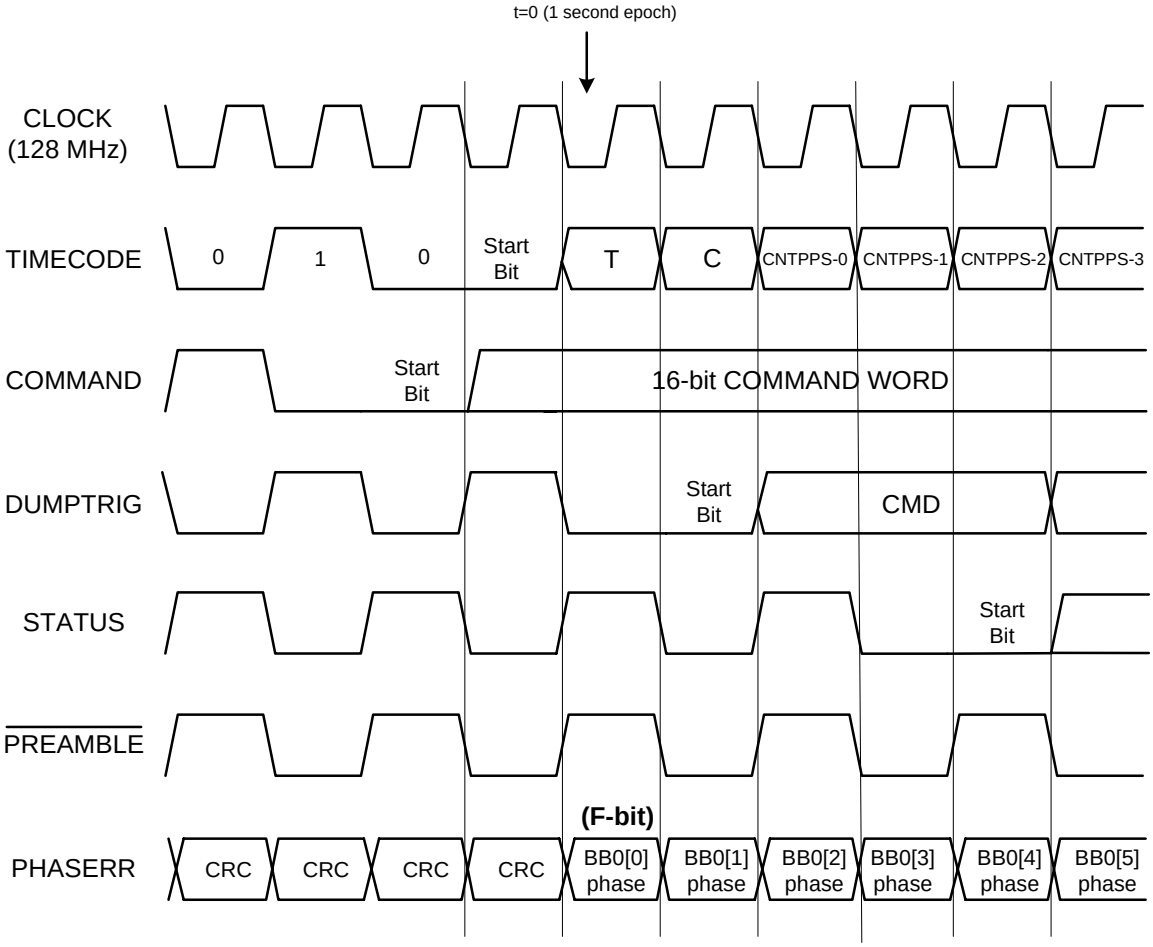


Figure 5-4 CTRL stream bit alignments going into the CTRL signal multiplexer. Note that DUMPTRIG is only aligned to TIMECODE inasmuch as their preambles are aligned (i.e. the location of the DUMPTRIG Start Bit is for illustration only), similarly for COMMAND and STATUS relative to $\overline{\text{PREAMBLE}}$.

5.3 Data Stream Content

The 256 Mbps data stream that gets chopped and multiplexed via a three-stage process to 1 Gbps as shown in Figure 5-1 contains a bit format that allows it to be "unraveled" at the receiving end. The bit format, before it enters the chopper of Figure 5-1 is as shown in the following diagram:

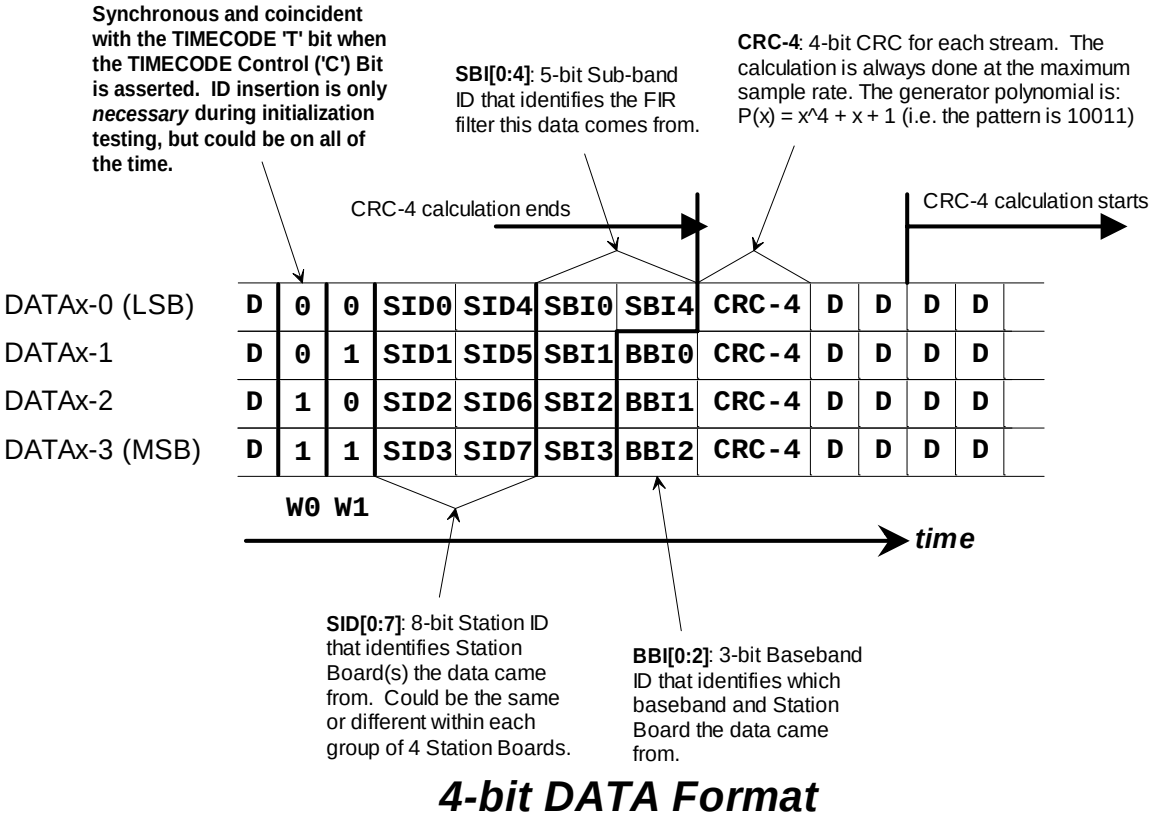


Figure 5-5 4-bit DATA format. Embedded information uniquely identifies the source of the data, arrangement of the bit streams, and the CRC-4 code provides robust error checking capability. The CRC is not calculated on two bits (D) in the stream to provide the dead time needed so that only one CRC transmitter or receiver is needed for each stream.

In addition, the following points regarding the encoding of the data samples themselves are important to note:

- The data words (**D** words) represent 4-bit, 2's complement data encoding at a sample rate up to and including 256 Msamples/sec. Alternatively, alternating **D** words represent 7-bit 2's complement data encoding at a sample rate up to and including 128 Msamples/sec (i.e. data is interleaved LSN-0, MSN-0, LSN-1, MSN-1 etc.)
- For 4-bit encoding, the most-significant bit (MSB) is bit 3 (DATAx-3 in the diagram). For 7-bit encoding, the MSBit of the MSN (most significant nibble) is bit 2. Note that the LSN of the 7-bit encoded word is aligned with first half of the

TIMECODE T-bit (‘**W0**’ in Figure 5-5) (i.e. the interleaved sequence starts off as LSN, MSN, LSN, MSN etc.)

- For 4-bit encoding, the ‘5’ state (0101b), if present, represents data invalid (i.e. a ‘5’ word indicates an invalid sample[6]).
- For 4-bit encoding, the ‘-8’ state (1000b), if present, represents the data value of ‘5’ (0101b).
- For 7-bit encoding, bit 3 of the MSN indicates whether the data sample, including the immediately *preceding* LSN, is valid (1) or invalid (0).
- Generally then, for either 4-bit or 7-bit encoding, the ‘5’ state (0101b) represents data invalid.
- If one or more particular DATA streams represent data rates at lower sample rates than 256 Ms/sec, then the actual valid samples (and in the 7-bit case, the LSN of the 7-bit word) are frame-aligned with the (first half) of the TIMECODE ‘T’ bit. Other unused samples **must** be set to a ‘1010’ pattern so that there are always enough transitions in the resulting 1 Gbps stream to maintain dynamic phase alignment[7]. Functional timing of several of these cases is shown in Figure 5-6.

[6] This was originally put into place to prevent DC on the 1 Gbps multiplexed line when data is invalid. This requirement has disappeared with the chopper circuit, although this encoding is still and will continue to be in use.

[7] Not a requirement anymore with the chopper circuit in place. The chopper can and does easily handle a DC (all 0000’s or all 1111’s DATA input).

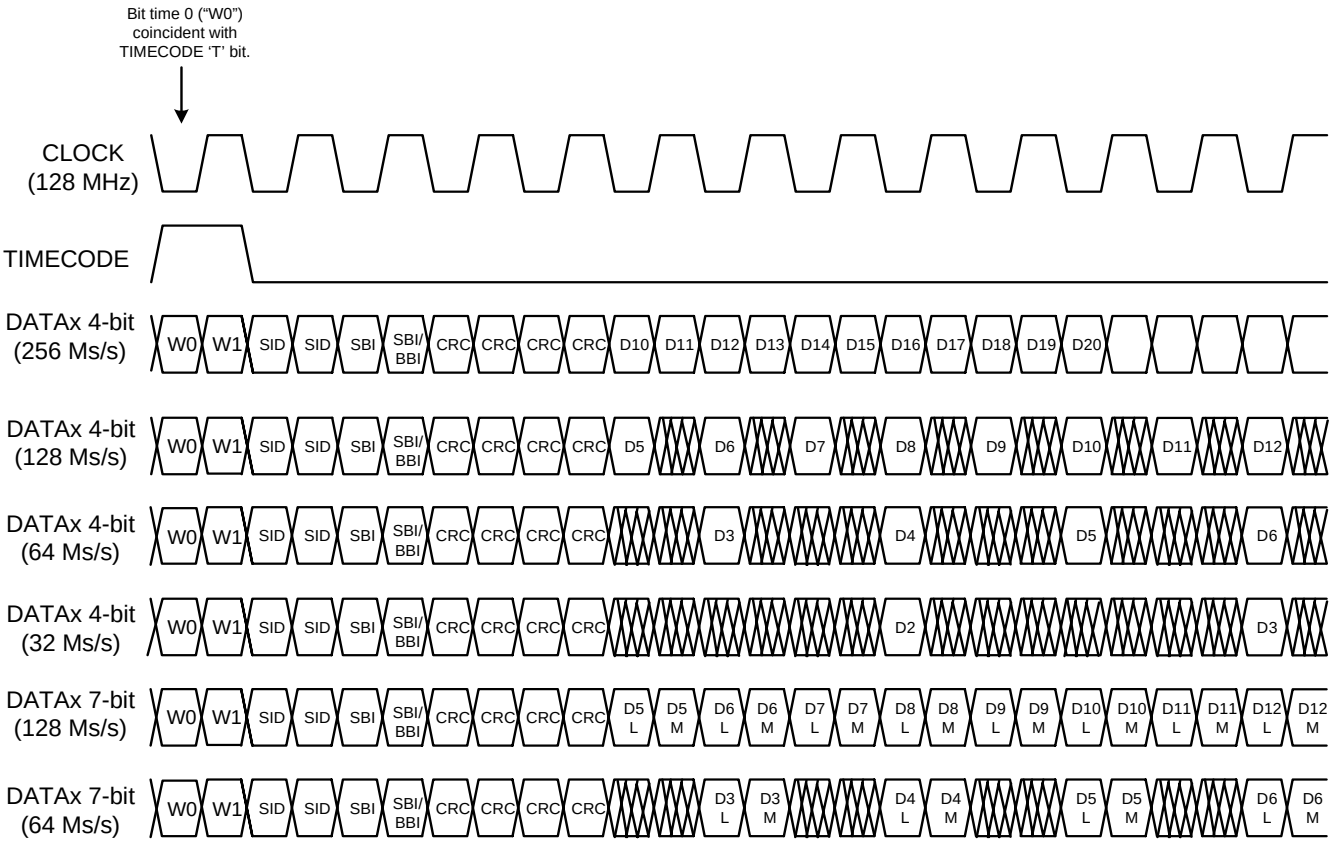


Figure 5-6 Functional timing diagram of 4-bit and 7-bit data for several different sample rates. Note that 'W0' ALWAYS marks the t=1 second epoch, and all data is subsequently aligned to that.

- If a particular data stream is not being used by the correlator, then the generator of the unused DATA stream must still keep it active. This includes the embedded information as well as the insertion of the '1010' bit pattern. If there is no DATA stream generator available (i.e. no Station Board available to generate it), then the wires may be left in the un-driven/floating state, and software will have to recognize that the bitstream is dead.

5.4 CTRL Stream Content

The 1 Gbps multiplexed CTRL stream generated by the multiplexer shown in Figure 5-3 contains all of the control streams needed for downstream hardware to process the BB0 and BB1 streams in the associated connector (Figure 4-1). This section defines the content of each of these control streams.

5.4.1 TIMECODE Format

The TIMECODE bit stream contains the master time reference against which all other control streams and sampled data streams will be aligned in the receivers on the Baseline Board. The primary epoch is the TIMECODE T-bit that occurs every 10 milliseconds.

The format of the TIMECODE stream is shown in Figure 5-7 below, and its alignment to other streams is shown in Figure 5-4 and Figure 5-6.

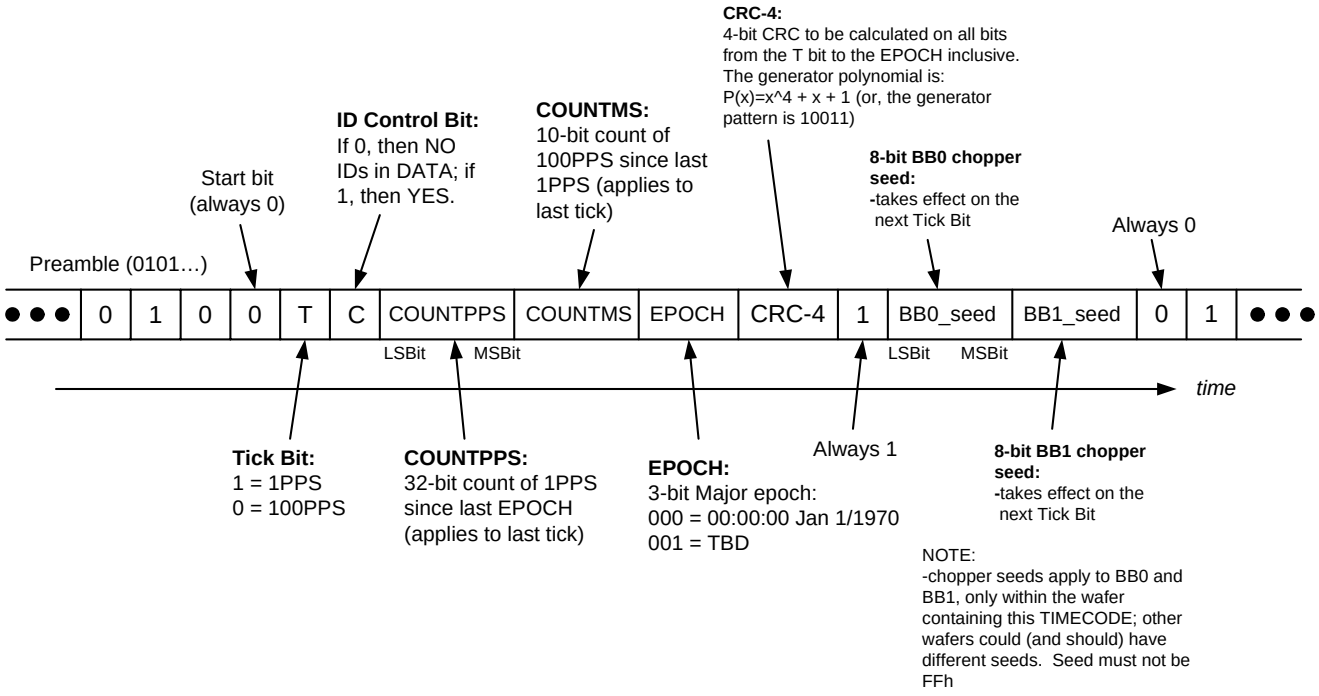


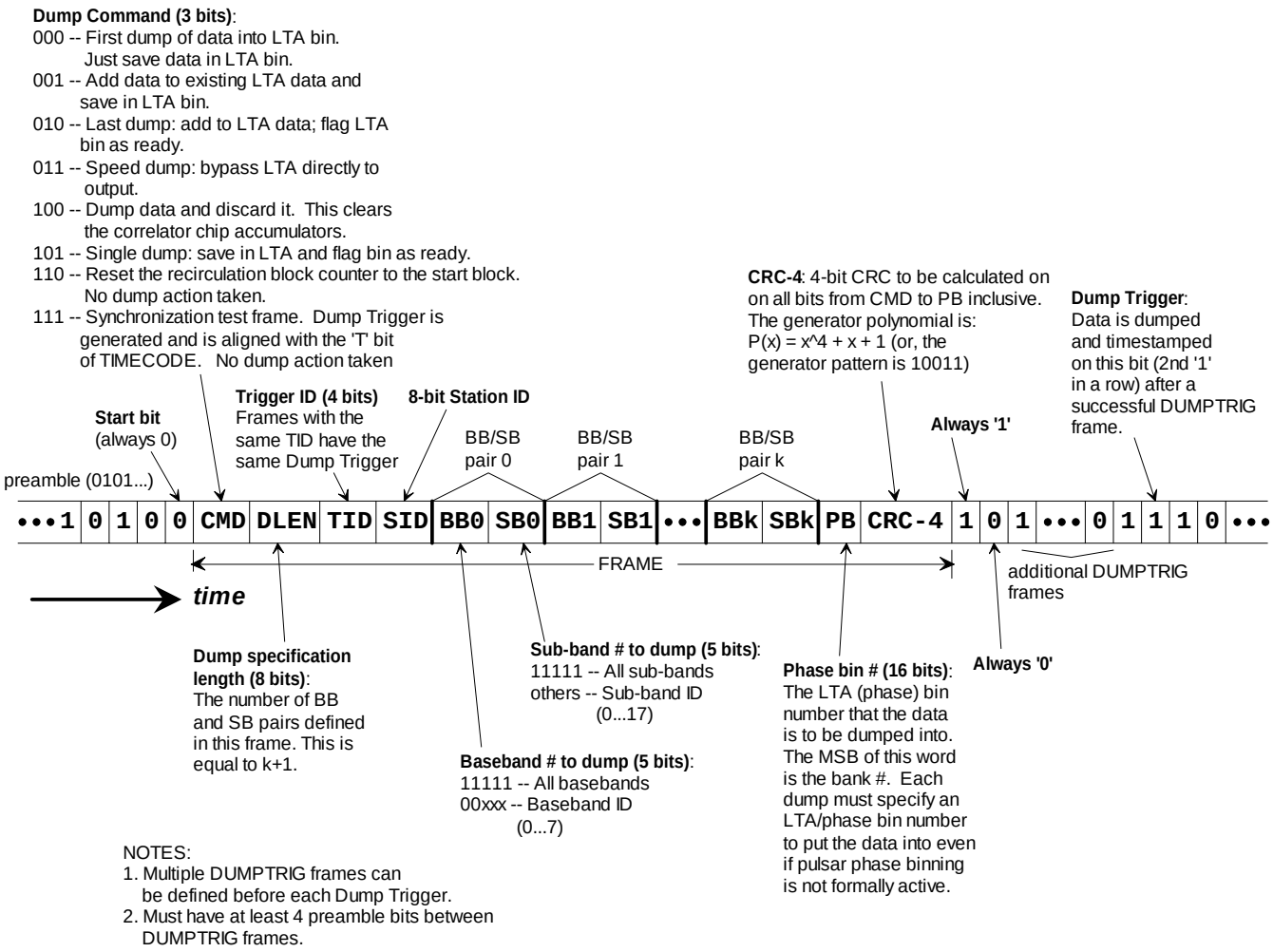
Figure 5-7 TIMECODE format. The 'T' bit marks the 1 second or 10 millisecond epoch. The 'C' bit is the ID control bit and defines whether there are identifiers embedded in the DATA streams at this epoch or not. If (0), then no identifiers, if (1), then identifiers are present.

The TIMECODE signal also contains the 8-bit chopper PRN seeds ("BB0_seed" and "BB1_seed") that are used in the transmitter chopper and the receiver de-chopper. These are not error-checked in the TIMECODE stream, but are checked by virtue of their use, and with error checking already in TIMECODE. Thus, the receiver extracts the required seeds from TIMECODE and uses them on the next received Tick Bit. Normally, seeds for a particular wafer are static and not changing with time.

5.4.2 DUMPTRIG Format

The DUMPTRIG bit stream contains the real-time dump control information for the particular Station Board[8] that sources it. This signal controls the dumping and readout of data from the correlator chips if this station is selected for dump control (i.e. selected in the correlator chip). Even if this station isn't selected, if recirculation is active, this signal is used for recirculation control and synchronization. DUMPTRIG operates at a 128 Mbit/sec rate and all control words are transmitted least significant bit first. The DUMPTRIG bit format is defined in Figure 5-8.

[8] Each Recirculation Controller only decodes and uses one DUMPTRIG signal within the 4 wafers that it has access to.



DUMPTRIG Format

Figure 5-8 DUMPTRIG frame format. A DUMPTRIG frame can contain dump control information for one or more sampled data streams, determined by the stream's BB/SB IDs that match IDs embedded in the data stream format of Figure 5-4. Bits are transmitted least-significant bit first. A DLEN value of 0 is acceptable, and is normally the case for signaling frames where no dump action is taken.

The following notes in regards to the operation of DUMPTRIG need to be considered:

- DUMPTRIG operates in a "last overwrite" fashion so that the last dump specification for any particular SBB/SB pair overrides any previous specification up to the point of where the Dump Trigger actually occurs. In this way it is possible, for example, to specify that everything is dumped and then subsequently only specify the BB/SB pairs that are exceptions to that blanket specification. Additionally, it is possible to specify that a BB/SB pair is dumped, even though the associated data stream is not active—in this case, the dump request is simply ignored. The 8-bit SID also has a wildcard of all 1's, not indicated in the figure.
- If the dump command specifies a "synchronization test frame" (command = 111b) then dumping may or may not occur depending on whether or not normal

DUMPTRIG frames have been generated before the Dump Trigger bit. If a synchronization test frame is generated, then the receiver will expect the next Dump Trigger bit to be generated coincident with the TIMECODE T-bit. This mechanism is used to ensure[9] that all time-skew compensation in the system properly lines up DUMPTRIG with TIMECODE, since DUMPTRIG may not necessarily be synchronized to the TIMECODE epochs (e.g. when pulsar binning).

- DUMPTRIG command sequences control recirculation counters in the Recirculation Controller, and there is an explicit RRC command (110). Refer to the Appendix of this document for more detailed information.
- When the DUMPTRIG command "RRC" (110) is present, the 16-bit PB field contains the DUMPTRIG source identifier, used for end-to-end DUMPTRIG routing identification. PB[7:0] is the antenna ID (normally the same as "SID"); PB[15:13] is the source baseband ID; PB[12:8] is the source DumpTrig ID (DTID). When *dynamic* recirculation is not active, RRC frames form no function and have no effect other than providing a carrier for this identification.

For clarity, DUMPTRIG commands from Figure 5-8 are re-iterated in Table 5-1:

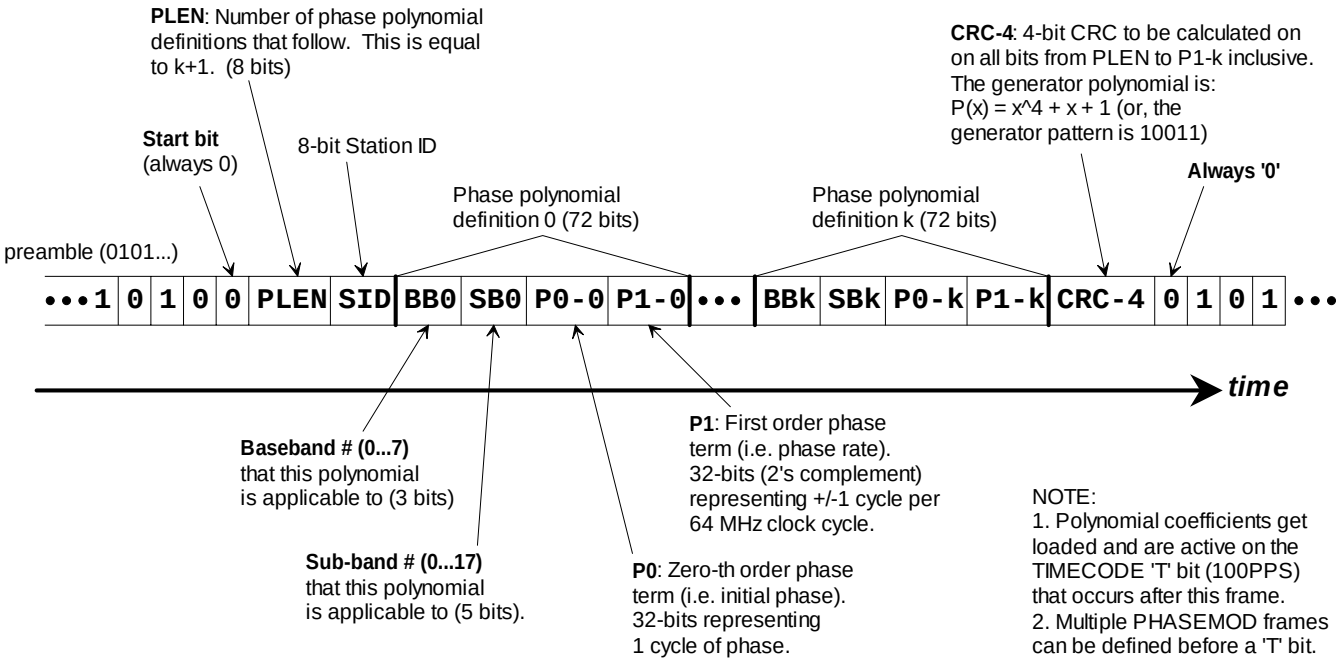
Dump Command	Description
000 (**FDS**)	First Dump Save: first dump of frame data into LTA bin. Save frame into LTA bin and overwrite existing data in bin **unless** the bin is flagged as ready for readout, in which case the new frame will be discarded.
001 (**AFS**)	Add Frame Save: add frame data to existing LTA bin and save the result back into the same bin.
010 (**LDS**)	Last Dump Save: add frame data to LTA bin, save result back into the same bin, and flag the bin as ready for readout (transmission to backend).
011 (**SPD**)	SPeed Dump: bypass LTA and transfer frame data to output for transmission to the backend.
100 (**DD**)	Dump Discard: dump data and discard it. This clears the correlator chip accumulators. This action is taken by the correlator chip; the LTA does not see the frame.
101 (**SDS**)	Single Dump Save: save frame data in LTA bin and flag bin as ready.
110 (**RRC**)	Reset Recirculation Counter: force reset of the recirculation block counter to the start block. No dump action is taken. This applies to any recirculation chip that receives this command (i.e. no selection of SID, BBID, SBID etc. is performed). Also, PB contains DumpTrig source ID.
111 (**STF**)	Synchronization Test Frame: Dump Trigger is expected to be aligned with the 'T' bit of TIMECODE. No dump action taken.

Table 5-1 DUMPTIG Dump Commands.

[9] As TC and DT are contained within the same 1 Gbps CTRL stream, this mechanism is no longer useful, but is still active.

5.4.3 PHASEMOD Format

PHASEMOD contains the 32-bit point-slope phase models <u>for the sampled data streams *within the same HM wafer*</u>. The format for PHASEMOD is shown in Figure 5-9 below:



PHASEMOD Format

Figure 5-9 PHASEMOD frame format. Phase models consist of 32-bit point-slope phase models, valid on the next TIMECODE T-bit epoch (10 millisecond epoch). If a phase model for a stream is not present, then the hardware keeps using the existing phase model.

Some important points to note in regards to the operation of PHASEMOD are as follows:

- Each phase model consists of a 32-bit point-slope. The 0^{th} order model (P0) is a 2's complement number that represents $\pm 180^{o}$ of phase (i.e. the full-scale unsigned interpretation is 360^{o} of phase), valid at the *next* TIMECODE T-bit epoch. The 1^{st} order model (P1) is a 2's complement number that is phase rate in <u>cycles per clock period at a 64 MHz clock rate</u> (i.e. the downstream phase synthesizers operate at a 64 MHz clock rate, or equivalently, generate a new phase value every 15.625 nsec).

- The PHASEMOD format is flexible in that it allows only the desired models to be updated, leaving any ones not defined alone. The models defined in one or more frames become active on the Baseline Board on the next TIMECODE T-bit epoch. It is important to note that the BB# refers to the baseband number the data came from, and the SB# is the FIR filter the data came from (and both numbers are consistent with the numbering in the data stream defined in Figure 5-4)

- If serial phase is chosen for recirculation for a particular sample data stream, then 3.5 sample times, at the sampled data rate of the sampled data stream, must be added to the time for when the initial phase (P0) for the particular 10 millisecond epoch is calculated. **Example:** if the sample rate is 32 Msamples/sec, and t=1.010 seconds for the next 10 millisecond epoch, then instead of using t=1.010 seconds in the phase calculation, $t=1.010 + 3.5/32x10^6$, or 1.010000109375 seconds is used.

- If PHASEMODs are merged in a Cross-Bar Board (section 7.4, Figure 7-4), then they must be generated such that their preambles are consistently aligned to the TIMECODE 'T' bit, and that PHASEMOD frames from different sources that end up merged into the same output do not occur at the same time. This is because 'merging' is not a packet switching operation.

5.4.4 PHASERR Format

The PHASERR bit stream contains phase errors used to perform sub-sample delay tracking for the sampled data streams within the same HM wafer (refer to Figure 4-1). The format of the PHASERR bit stream is defined in Figure 5-10 below:

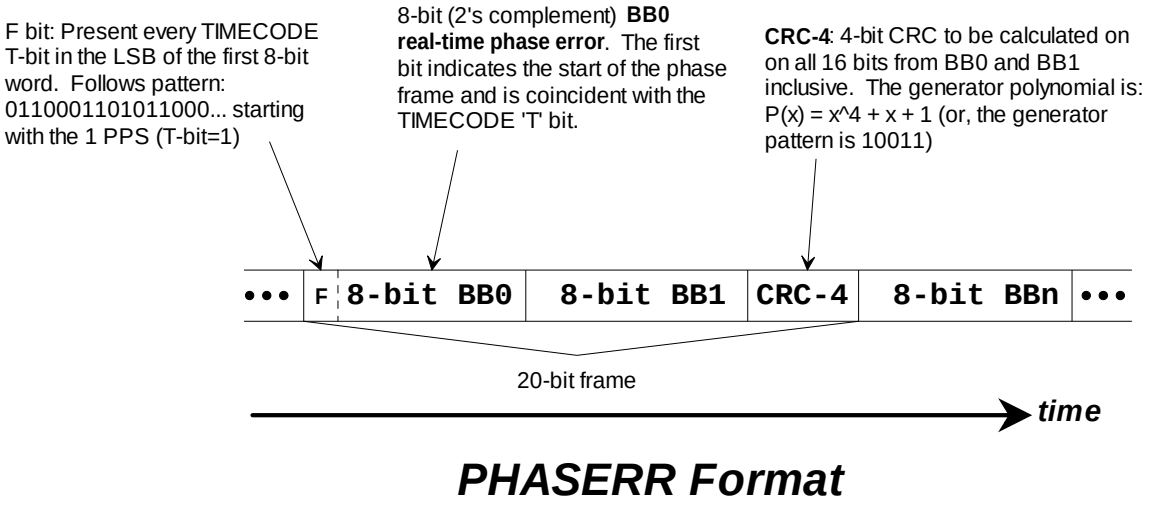


Figure 5-10 PHASERR format. PHASERR contains the real-time phase error models for the two sampled data streams (BB0, BB1) within the same HM wafer.

Important notes in regards to the operation of PHASERR are as follows:

- The 8-bit BB0 and BB1 PHASERR numbers are applied to the associated BB within the HM wafer.

- Each 8-bit value is a 2's complement number that represents the phase error between $\pm 90^o$. That is, the value "-128" is exactly equal to -90^o of phase, and the value +127 is exactly equal to $+90^o$-ε of phase—where ε is 180/256 degrees of phase. This number is ADDED to the phase synthesizer phase accumulator to produce final phase for the stream for phase rotation. (For the case where

PHASERR sign is the same as $tau_{digital}$ - tau_{model} it will thus de-rotate signal phase due to discrete delay tracking error of tau_{model} - $tau_{digital}$ to zero.)

- After 2 consecutive 8-bit numbers, a 4-bit CRC with generator polynomial $x^4 + x + 1$ (pattern=1001) is inserted. Thus, a 20-bit repeating frame (8-bit BB0><8-bit BB1><CRC-4>) is formed as shown in Figure 5-10, that always starts coincident with the TIMECODE T bit (refer to Figure 5-4). In all cases, the LSB is transmitted first.

- The 'F' bit in PHASERR is present and coincident with the TIMECODE T bit, and replaces the least-significant bit of the 8-bit BB0 PHASERR word every 10 milliseconds. This bit follows the repeating 10-bit pattern "0110001101, 011000…" starting on the 1 PPS epoch (i.e. T-bit=1), where the first "0" in the sequence is the first bit out. This embedded pattern ensures that the receiver does not falsely lock onto the frame as could happen if PHASERR is not changing very often, and if the CRC happens to be correct for the alignment chosen.

5.4.5 COMMAND and STATUS Formats

The COMMAND and STATUS lines have the same format. Each consists of a 16-bit payload, followed by a 16-bit checksum, and a 4-bit CRC-4 code calculated on the two 16-bit words, as shown in Figure 5-11. The start bit can occur at any time—its bit position relative to any other Control stream is not important, except that it must be aligned to $\overline{\text{PREAMBLE}}$ according to Figure 5-4.

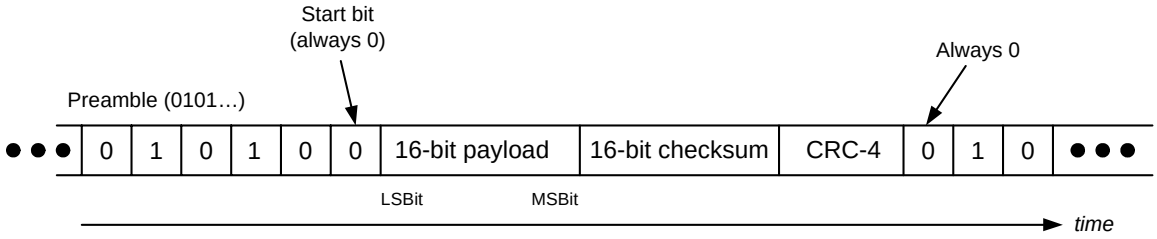


Figure 5-11 COMMAND and STATUS frame. The contents of the 16-bit payload is not defined here, and is a function of the particular receiver's and transmitter's needs, depending on whether it is a COMMAND or a STATUS line. The 16-bit checksum is the least-significant 16 bits of the sum of the 16-bit payload and the sync word 0x5e2d, calculated 16 bits wide.

COMMAND is used by a transmitting device to tell the next receiving device to do something. STATUS is used by a transmitting device to transmit its status information to a downstream receiver. These lines are primarily included in this specification to allow for communication with the X-bar Board, which doesn't have a CPU, and needs to perform programmable cross-bar switching functions. In this case, the Station Board sends COMMANDs, which are received and acted on by the X-bar Board, and the X-bar Board sends STATUS words, which are decoded on the Baseline Board.

The content/format of the 16-bit payload is not part of this protocol specification (although COMMAND protocol for the X-bar Board is included in the Appendix) and is

a function of the particular receiver's and transmitter's board's needs, and depending on whether it is a COMMAND or STATUS line.

The 16-bit checksum provides additional error checking ability on the 16-bit payload and is the least-significant 16 bits of the sum of the 16-bit payload and the **sync word 0x5E2D** (hexadecimal), calculated 16 bits wide.

If a transmitter does not require the use of COMMAND or STATUS, then each should be set the same as $\overline{\text{PREAMBLE}}$.

Example: if the payload is 0x346D, then the 16-bit checksum is 0x929A.

6 References

[1] Zhang, Heng, INTERFACE CONTROL DOCUMENT: HM Gbps Cable Physical Specification, ICD Document A25022N0040, Revision Draft, April 13, 2004.

7 Appendices

7.1 CRC-4 Circuitry

Hardware calculation of a CRC is a simple, efficient, and straightforward process at both the transmitter and receiver. The basic schematic for the CRC-4 with generator pattern 10011 required by this interface is shown in Figure 7-1. For the transmitter, the following steps are required:

- Reset all of the DFFs.
- Input the transmit bit sequence, LSB first, into the INPUT.
- Append 4 zeros to the end of the transmit bit sequence—once the last zero is clocked to position C0, then the calculated CRC is C0…C3.
- C0…C3 is transmitted as the CRC-4 code, LSB (C0) first.

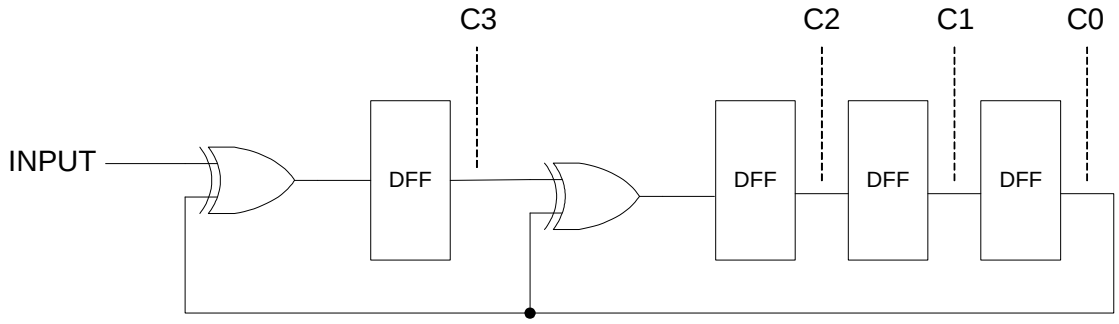


Figure 7-1 CRC-4 circuitry with generator pattern 10011. The coefficients C0, C1, C2, C3 represent the calculated CRC for the transmitter, 4 clock cycles (zero padded) after the transmitted bit stream enters the INPUT. For the receiver, the entire input and the 4-bit CRC should yield an all zero result (C0…C3) if there was no transmission error.

For the receiver, the following steps are required:

- Shift the entire received bit sequence and appended 4-bit CRC code into the INPUT, until the last bit is shifted through to the C3 location. If there was no transmission error, then C0 through C3 should all be zero.

Example: The transmitted bit sequence (LSB…MSB) 010111001011101 results in a CRC code of 0111. Receiving the sequence and CRC 0101110010111010111 results in all zeros in locations C0…C3.

Refer to http://www.cee.hw.ac.uk/~pjbk/nets/crctutorial.html for a useful tutorial and demonstration of CRC transmission and reception. See also Xilinx Application note 209.

7.2 DUMPTRIG and Recirculation

When recirculation is active on the Baseline Board, DUMPTRIG command sequences (i.e. the **CMD** word in the DUMPTRIG frame) affect how the recirculation block counter in the Recirculation Controller on the Baseline Board counts, within the configuration settings of the chip. The following table indicates how the recirculation counter is affected for DUMPTRIG commands used when recirculation is active.

Table 7-1 DUMPTRIG CMD and how it affects recirculation counting in the Recirculation Controller on the Baseline Board.

CMD	Recirculation counter	Description/explanation
100	Reset to start block	Dump and discard command always resets the recirculation block counter
000	Count up	Each command acquires first data for a different lag block.
001	Count up	Each command accumulates data for a different lag block.
First 010 after 000 or 001	Reset to start block	Start last dump and flag as ready—we know the block counter will be reset at this point since we are hitting the first recirculation block.
2nd and subsequent 010	Count up	Each command dumps, integrates, and flags data as ready.
101	Count up	Single dump and flag as ready.
110	Reset to start block	Always forces "reset to start block" independent of any other commands sent for the same Dump Trigger. This command, in itself, does not generate any dump action, however, normally dump action should be forced by some other command for the same Dump Trigger. It is strongly recommended to generate this command on a regular basis when "reset to start block" is expected to ensure proper synchronization of the block counters across the system on a periodic basis.

7.3 Chopper Circuit

The chopper circuit uses an 8-bit pseudo-random number (PRN) generator taken from Xilinx XAPP 052 and exclusive-OR gates to randomize data before multiplexing and transmission, effectively multiplying the data by noise. At the receive end, it de-chops the incoming chopped data by performing the same operation. The PRN generator is synchronous with the data 256 MHz clock, and is synchronously set to SEED values every Timecode T-bit (100 PPS; i.e. every 10 msec), independent of whether the DATA signals contain embedded IDs or not (i.e. independent of the C-bit). There are 2 pipeline stages in the chopper so any mux/demux operations synchronized to the 128 MHz clock domain are not affected (i.e. 1 pipeline delay at 128 MHz rather than 0.5 pipeline delays). Going into the chopper module, the Timecode T-bit is always 2, 256 MHz clock cycles wide, and always aligned to W0 and W1 of the DATA (DIN) input of Figure 5-5.

A schematic of the chopper circuit, including the PRN generator is shown in Figure 7-2:

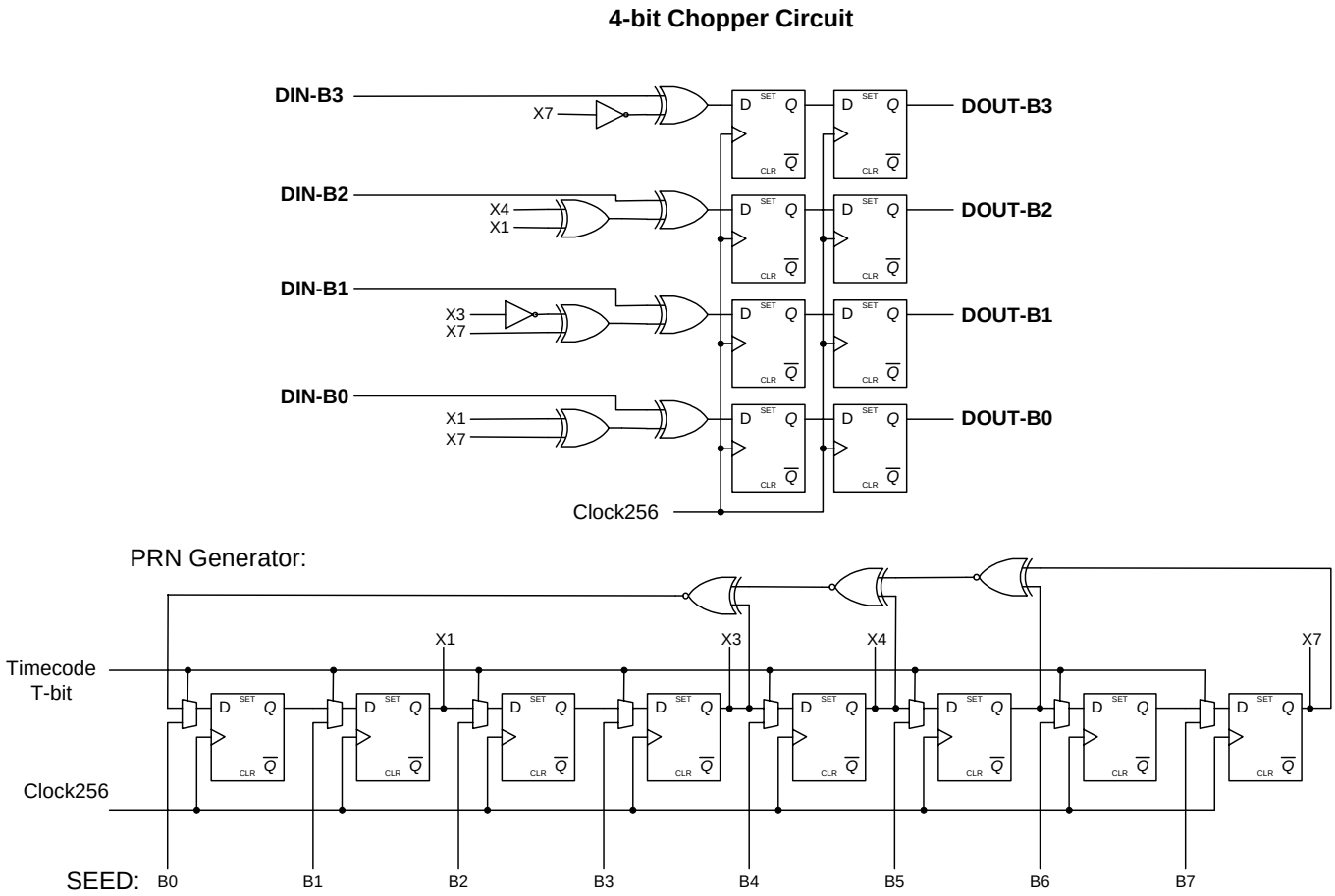


Figure 7-2 Chopper circuit schematic.

Verilog source code of the chopper circuit "chop_4bit.v" is included below:

```
module chop_4bit (clock256, rst, Tbit, seed, Din, Dout);

input             clock256;
input             rst;
input             Tbit;
input [7:0]       seed;
input [3:0]       Din;
output[3:0]       Dout;

wire              clock256;   // 256 MHz clock
wire              rst;        // active-low asynchronous reset
wire              Tbit;       // Timecode T-bit, 2 clock256 cycles wide
wire  [7:0]       seed;       // 8-bit seed; must not be all 1's
wire  [3:0]       Din;        // 4-bit input data
reg   [3:0]       Dout;       // Chopped 4-bit output data

reg   [7:0] lfsr;
reg   [3:0] Dout_internal;

always @ (posedge clock256 or negedge rst)
begin
      if( ~rst )
        begin
          Dout<=4'h5;
          Dout_internal<=4'h5;
        end
      else
        begin
          Dout_internal <= Din^{~lfsr[7],lfsr[4]^lfsr[1],
                            ~lfsr[3]^lfsr[7],lfsr[1]^lfsr[7]};
          Dout<=Dout_internal;
        end
end

// Generate the LFRS PRN pattern, synchronous with the
// Timecode T-bit

wire  [7:0] lfsr_in;

// This is from Xilinx app note XAPP 052; its repeat cycle is 256.
assign lfsr_in[7:0]={lfsr[6],lfsr[5],lfsr[4],
                     lfsr[3],lfsr[2],lfsr[1],lfsr[0],
                     ~((~((~(lfsr[7]^lfsr[5]))^lfsr[4]))^lfsr[3])};

always @ (posedge clock256 or negedge rst)
begin
      if( ~rst )
        lfsr<=8'b0;
      else
        begin
          if( Tbit )
            lfsr<=seed;
          else
            lfsr<=lfsr_in;
        end
end
endmodule
```

7.3.1 Chopper SEED Algorithms

Different seeds are used in different chopper circuits within the same FPGA since it was found during testing that under some conditions (i.e. many DATA streams containing the same data signal), excessive noise is generated internal to the FPGAs causing output 1.024 Gbps lines and 128 MHz clocks to contain excessive jitter. By using different seeds, this analog side-effect was eliminated.

Algorithms have been developed for each stage of signal generation shown in Figure 3-1, and they are documented here for completeness. The requirement is that inasmuch as possible, all 1.024 Gbps DATA lines going out of an FPGA and into a destination FPGA at the next stage use different chopper seeds so the condition of many signals switching at the same time in the same direction is avoided. To avoid effects of cross-bar switching (i.e. all outputs could be connected to the same input), chopping is done ***after*** any cross-bar switching in a chip.

Station Board Output FPGA seed:

Bit 7 – Crate ID
Bits[6:4] – Slot ID in the crate
Bits[3:0] – (Output wafer # + Output FPGA #) modulo 15

Where “Output wafer #” is the range 0-17, and Output FPGA # is 0 or 1.

The “modulo” 15 operation on Bits[3:0] ensures that the all 1’s seed (FFh) is never used, as this seed causes no chopper switching signals to be generated. The destination for these signals is the X-bar Board, or direct to a Baseline Board and the combinations above largely provide for uniqueness at the receiving end, as well as at the transmitter.

X-bar Board FPGA seed:

Bit 7 – 0==BB0; 1==BB1 within the wafer
Bits[6:3] – Output wafer #
Bits[2:0] – Station Rack ID - 1, set by jumpers on the X-bar Board.

If a seed of FFh is detected, then Bits[2:0] are set to 000.

RXP FPGA seed:

No particular algorithm used, all outputs set differently and are different for the Upper RXP and the Lower RXP. Ensures that signals the route to the next Baseline Board RXP chips all have different seeds.

Recirc FPGA seed:

Not applicable. The Y Recirc FPGA that re-transmits data to the Y output connector does not do any cross-bar switching in that path and so no seed re-generation is required.

7.4 X-bar Board COMMAND Protocol

The X-bar Board executes COMMANDs originating upstream (normally from the Station Board) to change data routing and to control output drivers that drive long cables going to downstream Baseline Racks. Each X-bar Board contains 2 FPGAs with identical functionality, and as wired for the EVLA correlator are logically connected as follows:

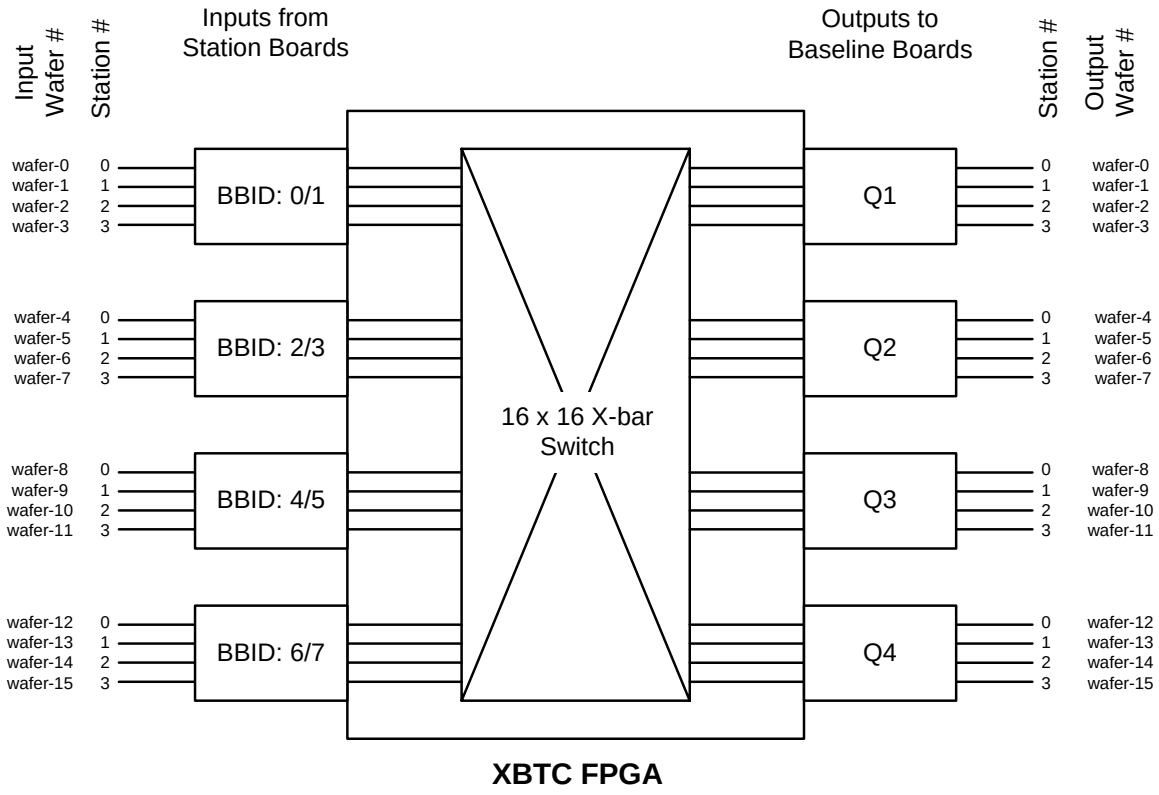


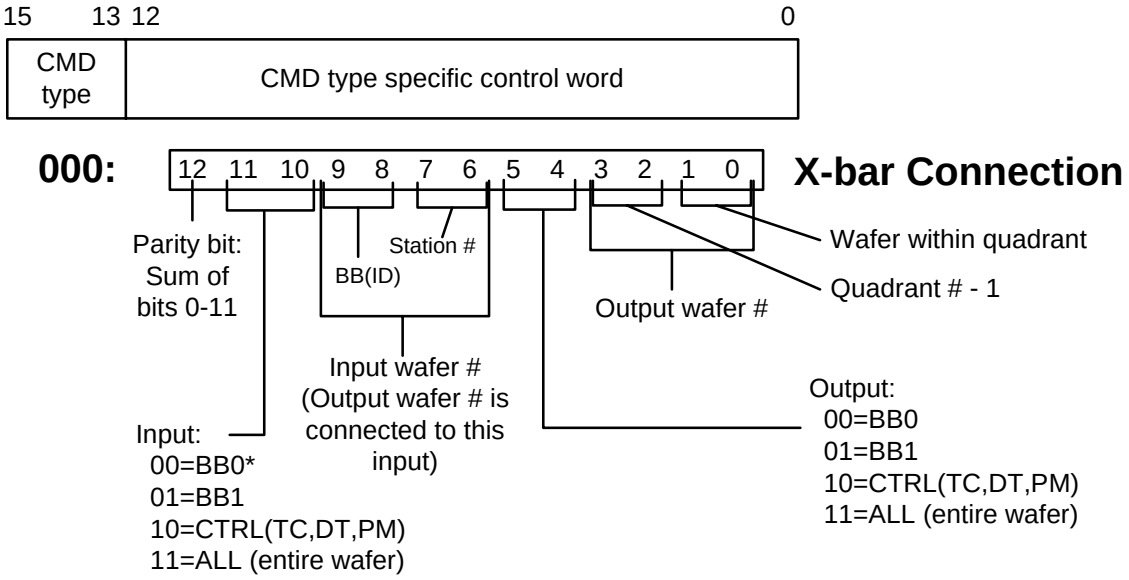
Figure 7-3 X-bar Board FPGA functional connections.

Note that X-bar Switch and output driver settings, once established, remain in effect until the board is power cycled, or the FPGA is hard reset (normally done only with a power cycle). Thus, temporary PLL lock loss does not affect switch settings.

The COMMAND protocol is shown in Figure 7-4. Any COMMANDs that are received with any errors (e.g. checksum of Figure 5-11 is wrong, CMD type is incorrect etc.) are errors, are ignored, and do not change switch or output settings.

COMMANDs are handled and executed very fast in the FPGA, but to avoid missing COMMANDs, only 1 COMMAND should be sent on each wafer every 10 μsec; a faster rate could result in lost/ignored COMMANDs.

HM Gbps X-Bar COMMAND payload:



NOTES:
1. COMMANDs can come from any input wafer and are applicable to any output wafer.
2. Switching a BB (BB0 and/or BB1) also automatically causes PHASERR and PHASEMOD to switch with it/follow it. PHASEMODs from two sources (BB0 wafer and BB1 wafer, which) may be different) are then 'merged' to produce one final PHASEMOD signal for the output wafer. 'Merging' requires that PHASEMOD packet generation from the two sources be mutually exclusive, with preambles synchronized; **'merging' is a pure ANDing operation and no collision detection/avoidance is attempted**.
3. Wafers' CTRL streams containing TIMECODE and DUMPTRIG must be switched explicitly.
4. If a COMMAND tries to connect a BB output to a CTRL input (or vice versa), it is an error and is ignored.

* BB0 refers to sub-band from StBrd Data Path 0;
BB1 refers to sub-band from StBrd Data Path 1;

Figure 7-4 X-bar Board COMMAND protocol.

7.5 X-bar Board STATUS Protocol

The X-bar Board generates continuous STATUS signaling on every wafer output's CTRL STATUS line (Bit-3, Figure 5-3) containing most of the board's front-panel LED colours. Any wafer receiver can capture this signaling, and determine the source X-bar Board's front-panel LEDs (except for voltage LEDs and supervisor status LEDs).

Signaling format and protocol is according to section 5.4.5; the 16-bit payload is according to the following table:

Payload Bits	Description
[1:0]	Wafer Rx LED group: 00 – wf 0-3 inputs 01 – wf 4-7 inputs 10 – wf 8-11 inputs 11 – wf 12-15 inputs
[3:2]	TCR LED
[5:4]	TCF LED
[7:6]	TCL LED
[9:8]	wf 0 (,4,8,12[10]) LED
[11:10]	wf 1 (,5,9,13) LED
[13:12]	wf 2 (,6,10,14) LED
[15:14]	wf 3 (,7,11,15) LED

Table 7-2 X-bar Board STATUS payload bit encoding.

LED colours are as follows (Bit1 Bit0):

00 – LED is OFF (dark).

01 – LED is RED.

10 – LED is GREEN.

11 – LED is YELLOW.

Note that STATUS signaling is sent continuously and fast enough (every few microseconds) to easily sample the dynamic nature of the LEDs, which indicates various conditions as defined in the X-bar Board User Manual. STATUS signaling is not synchronized or driven by the TIMECODE T-bit, so it may be obtained as long as the X-bar Board has a good clock, the receiver has a good clock and, at a minimum, the receiver has aligned the incoming de-serializer to find the "Bit-7 all 1's" code (Figure 5-3).

[10] i.e. depending on the Wafer Rx LED group; e.g. if Wafer Rx LED group is "01", then wf 4 LED status is present in bits [9:8].