EVN Correlator Design

Version 2.1

June 13th 2012

Jonathan Hargreaves (JIVE)
Harro Verkouter (JIVE)

EVN Correlator Design

Contents

Table of Contents

List of Abbreviations

1GbE	One gigabit per second Ethernet
10GbE	Ten gigabit per second Ethernet
BN	Back Node. One of the four FPGAs on the right half of the UniBoard
DDR	Double Data Rate. Data transferred on both edges of the clock
DDR3	A memory module conforming to JEDEC standard JESD79-3B
DFT	Discrete Fourier Transform
DSP	Digital Signal Processing
EEPROM	Electrically Erasable Programmable Read Only Memory
EPCS	Altera programmable configuration device
EVN	European VLBI Network
FFT	Fast Fourier Transform. An efficient implementation of a DFT
FN	Front Node. One of four FPGAs on the left side of the UniBoard
FPGA	Field Programmable Gate Array
GMAC/s	One billion multiply-accumulate operations per second
INTA,INTB	General purpose signals connected to all FPGAs on the UniBoard
MAC	(1) In Networking: Media Access Controller
	(2) In DSP: Multiply-Accumulate
MTU	Maximum Transmission Unit. The largest packet allowed on a network
MT/s	Million transfers per second
PFB	Polyphase Filter Bank
PPS	Pulse per second
PSN	Packet Sequence Number
SOPC	System on a Programmable Chip
SRAM	Static Random Access Memory
UDP	User Datagram Protocol
VDIF	VLBI Data Interchange Format
VLBI	Very Long Baseline Interferometry
WDI	Watchdog Interrupt

1 Control Software

Each of the eight Altera GX230 FPGAs on the Uniboard includes, as part of its power-up configuration, a 1 Gigabit Ethernet port and embedded Nios II processor to send and receive control and status information. The embedded processor together with its peripherals is referred to as an SOPC system, and can be assembled using Altera's SOPC builder tool.

The FPGA Ethernet ports connect to the outside world via a Vitesse Semiconductors VSC7389 single chip switch which provides a non-blocking wire speed connection to four RJ-45 connectors. The scheme is shown in Figure 1.1

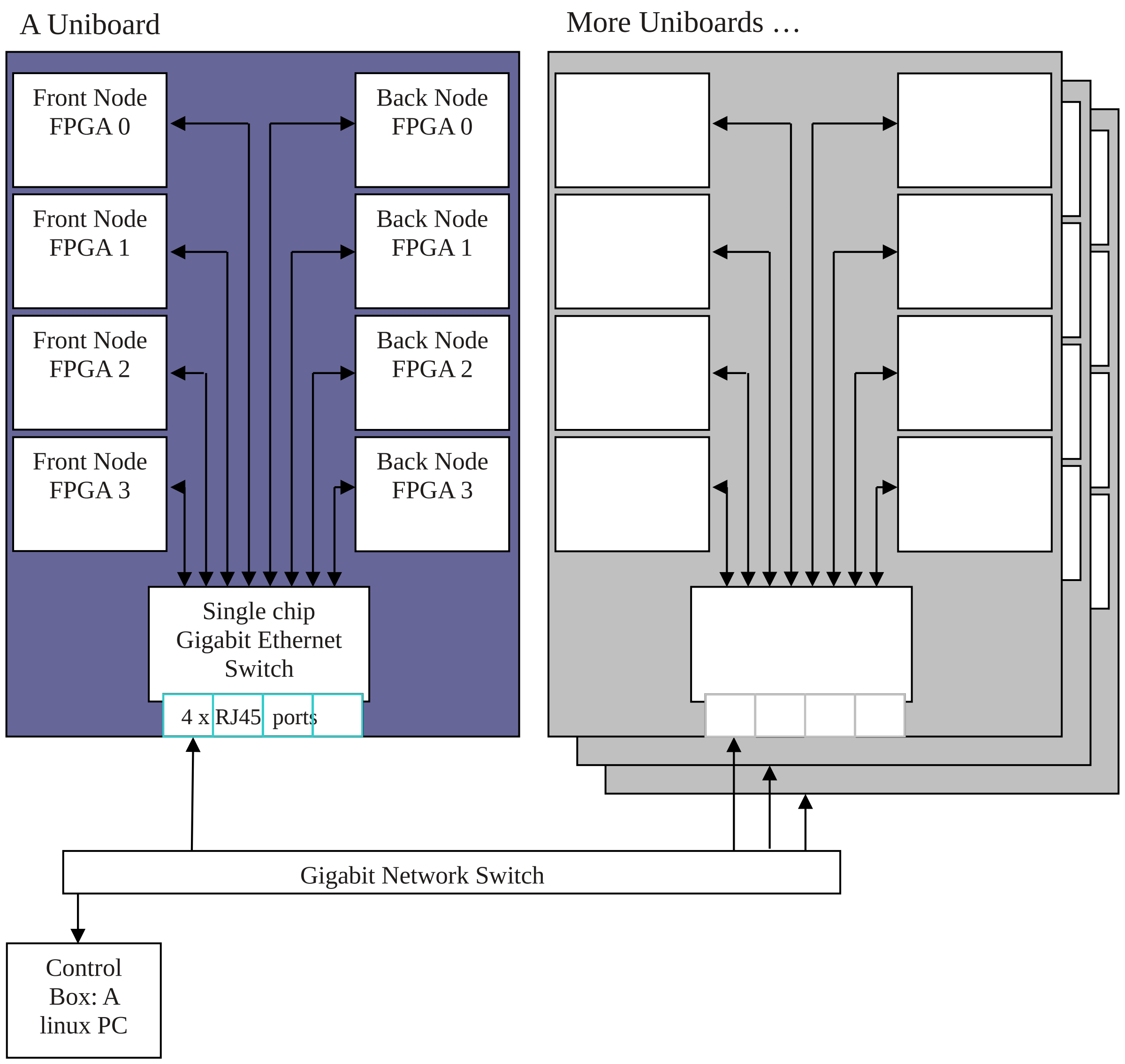


Figure 1.1: *Black lines show Gigabit Ethernet control network for Uniboard*

The figure shows a single Gigabit connection to several UniBoards via a switch to an external computer, an arrangement which provides sufficient bandwidth for control, including updating delay models, of at least 64 UniBoards. Expansion to 4 Gbps per UniBoard is possible using the remaining three RJ-45 connectors, allowing correlation product output over the 1GbE network in configurations with a long integration time. Since the control and monitoring flow is primarily in the opposite direction to the data flow, most of the 1 Gbps per FPGA is available for data.

A set of registers within each FPGA provides a control/monitor interface to the firmware. Some registers are provided as part of ready made blocks of IP, such as ten Gigabit MACs and DDR memory interfaces, while others are created using parallel IO ports (PIOs) to link the Nios software to the fabric of the FPGA. Within the SOPC system these registers have symbolic addresses – software running on the Nios processor translates these to present a uniform memory mapped register set to the outside world.

The control computer communicates with the FPGAs by sending non-jumbo (1400 byte MTU) UDP control packets containing a combination of the read/write instruction fields detailed below. Packets from the control computer may include requests to read, write or read-and-write-back data to or from specific registers, or blocks of registers. A read-and-write-back performs a masked AND, OR or XOR operation to reset, set or toggle individual bits within a register. In addition, configuration data can be streamed to an FPGA for storage in the flash memory.

Each control packet includes a unique packet sequence number (PSN). Once an FPGA receives a control packet it starts executing the commands embedded therein, accumulating the return values in a reply having the PSN of the currently executing command packet. It is vital to recognize that *every* command yields a return value. Since almost all commands take a 32-bit address as operand they will return at least that address if the action was successful and the bitwise NOT of the address if a fault occurred (eg writing to a read-only location). This may be followed with the results of the action, which is highly command dependant. The (accumulated) reply packet is not sent until all commands have been processed.

The FPGA may not transmit to the control computer without first receiving a control packet from it. Correlation data output should be directed to a back end processor at a different IP address from the control computer.

After a power-up or reset the control computer reads a version code from each FPGA to determine the configuration of each chip. The FN chip contains a different set of registers to the BN, and there may be alternative versions of each to accommodate different operating modes.

MAC and IP address assignment

At power-up each 1GbE port must be assigned a unique MAC address before it can be used. By default a MAC address is assembled by the Nios processor using a fixed component coded into the configuration file, and a board and chip-specific component hardwired to each FPGA. Later, when many UniBoards are deployed, it will be more flexible to save the MAC address in an unused area of the configuration flash memory. Once the MAC address is set the control computer assigns an IP address to the port by DCHP.

The control computer is also responsible for assigning MAC, IP, port and gateway to any 10GbE ports in the design.

Reset

Soft Reset
At initialisation after power-up, and on request from the control computer, the Nios processor issues

an active high soft reset via a bit in the control register. This signal propagates through the FPGA fabric and resets all state machines and logic to a known good state.

Hard Reset
On a hard reset the FPGA is reconfigured from its configuration EEPROM as it would be after a power cycle. This can be used to load a new configuration or, during debugging, to recover from error. Each FPGA is accompanied by a STM6823 watchdog timer which generates a hard reset automatically if the WDI output from the FPGA does not change state for approximately 1.6 seconds.

During normal operation the Nios processor toggles the WDI line periodically on receipt of an interrupt from an internal timer. This interrupt is set at a lower priority than signals from the control Ethernet port, so that a hard reset occurs if a bug causes the processor to stop responding. It is also possible for the control computer to force a hard reset by telling the Nios to set a bit in the control register.

FPGA Configuration

Each FPGA loads its configuration from a Numonyx M25P128 128Mbit serial flash memory. The SOPC system interfaces to this device using Altera's EPCS Device Controller Core which allows block and fine-grained read/write access to the flash. An uncompressed configuration image for the Stratix IV GX230 FPGA is approximately 104Mbits. The remaining memory will be available as non-volatile general purpose storage and may be used for example to store default MAC and IP addresses.

The flash memory is divided into pages of 256 bytes. The configuration image is streamed from the control computer to the chip in packets containing an integer number of pages – up to 5 pages fit within the control packet MTU. Programming the flash is on the order of 1000 times slower than sending the data over the network - it takes 2.5ms to program each page. To ensure that the configuration data are received and programmed in the correct order, a status register is updated with the new page count after each page has been accepted by the flash. The control computer must check this status register before transmitting any more configuration data. Of course many, or all, FPGAs on multiple UniBoards may receive configuration streams concurrently.

Time Synchronization

The UniBoards clocks are formally asynchronous with respect each other and the stations. When data starts to flow each UniBoard is roughly (to about 10ms) synchronized to the time stamps on the incoming data. The FPGAs on a single UniBoard are synchronized more closely – to within tens of nanoseconds using a common second tick (PPS) and ten millisecond tick distributed amongst them. More details are given in Section 2.

As some actions must be performed on a particular second (or subdivision) a mechanism is required to synchronize timing between the correlator and the control computer. To that effect a specific wait-for-pps instruction prefix is available. Each command described in the following section on format of data transfer fields may be preceded with this prefix. When the command-execution code encounters the prefix it will suspend itself and resumes execution after the next PPS tick has occurred. This allows a control system to synchronize command(s) as it sees fit.

One example is writing model coefficients: these registers should not be written to across a PPS boundary. By simply prefixing the write command with the wait-for-pps prefix, the sender of the command may rest assured that the command will not be executed across a PPS tick. As long as the coefficients are sent within the right second this is presumably good enough.

Given that *every* command has a return value, time synchronization between the control computer and an FPGA can be easily achieved by just issuing a wait-for-pps instruction. It has a reply and it will only be sent after the next PPS tick occurred. This reply typically arrives at the control computer between 0 and 50ms after the PPS, leaving the control computer with the better part of a second to arrange thing for the next second/PPS tick.

Format of data transfer fields

This section gives an outline of the proposed types of request/response between the control computer and the FPGAs. Several requests, including a mixture of reads, writes and configuration data, may be included within one packet following the PSN.

8 bits	8 bits	16 bits	Variable length
TYPE	OPERAND SIZE	NUMBER OF OPERANDS	DATA FIELD

General Format. Each request starts with a TYPE field. TYPEs 0x00 to 0x07 are commands from the control computer to an FPGA. TYPE 0x80 denotes an acknowldgement from the FPGA to the control computer. The length field gives the number of consecutive registers to be read or written. In general registers are 32 bit wide, though 8 or 16 bit registers may be defined as well. The control computer is expected to know the size of the registers it is addressing from the memory map of the FPGA.

8 bits	256 bytes
0x00	ONE PAGE OF CONFIGURATION DATA

Configuration Write: Transfer one page of configuration data to the FPGA. There is no address field since there is only one configuration flash memory per FPGA. The system may fit up to 5 pages of configuration data in one command packet. The reply is the counter value after having written the page.

Wait-for-pps instruction prefix

8 bits
0xFF

The system suspends execution of commands in the current packet until a PPS tick is seen. The wait should timeout in a period between 1.0s and 1.5s – the upper bound set by the watchdog interval: the FPGA will be automatically reset if the FPGA does not acknowledge the PPS tick for a period longer than 1.6s. Return value is “1PPS” if the PPS tick was detected or “0PPS” if not.

Read

8 bits	8 bits	16 bits	32 bits
0x01	SIZE [1|2|4]	N	START ADDRESS

Request to read N registers starting at START ADDRESS of size SIZE bytes per register. Returns the START ADDRESS + N*SIZE bytes if successful, or NOT START ADDRESS in case of failure to read.

Write (overwrite)

8 bits	8 bits	16 bits	32 bits	N*SIZE bytes
0x02	SIZE [1|2|4]	N	START ADDRESS	N words of SIZE bytes of DATA

Request to write the given DATA to a block of N registers of size SIZE starting at START ADDRESS. Returns START ADDRESS if successful write, NOT START ADDRESS in case of failure.

Read/Modify/Write bitwise operations

8 bits	8 bits	16 bits	32 bits	N*SIZE bytes
CODE	SIZE [1|2|4]	N	START ADDRESS	N words of SIZE bytes of MASK

Request to read the existing contents of registers starting at START ADDRESS, compute a new value from the bitwise operation indicated by CODE with the old value and MASK. Finally write the result back to the registers. The return value is START ADDRESS if all RMW sequences were performed successfully, NOT START ADDRESS in case of failure.

CODE	0x03	0x04	0x05
BITWISE OPERATION	AND	OR	XOR

2 FPGA Firmware

Introduction

Specifications for the new EVN correlator are the following[1]

Stations	32
Polarizations	2
Total Bandwidth	4096 MHz (expansion to 8192 MHz)
Bandwidth per UniBoard	64MHz (expansion to 128MHz)
Sub-band width	1, 2, 4, 8, 16, 32, 64MHz
Input resolution (max)	1-8 bits
Integration time	0.022s – 1s
Correlation points	2112 incl cross, auto, and cross-polarization
Frequency points per sub-band	4096 per 64MHz continuum, 10000 spectral line
Data Input Format	VDIF

The overall scheme for correlating real time data is shown in Figure 2.1. A UniBoard at each station, configured as a digital receiver, divides the sampled signal into sub-bands of up to 64MHz, 8 bit resolution. The sub-bands are packetized and transmitted across the 10GbE network to the correlator.

Destination IP addresses of each packet are allocated such that all the data for a given sub-band arrives at a single correlator UniBoard. Each UniBoard can receive sub-bands totalling 64MHz, with 1 to 8 bit resolution from 32, two polarization stations. Expansion to 128MHz per UniBoard is possible given sufficiently large and fast DDR3 memory modules.

[1] Note on units

Network transfer rates are base 10: 10Gbps = 10^{10} bits per second.
VLBI sample rates are binary in megasamples per second 1GSps = 1024 x 10^6 samples per second.
Memory sizes are binary: 1GByte = 2^{30} bytes

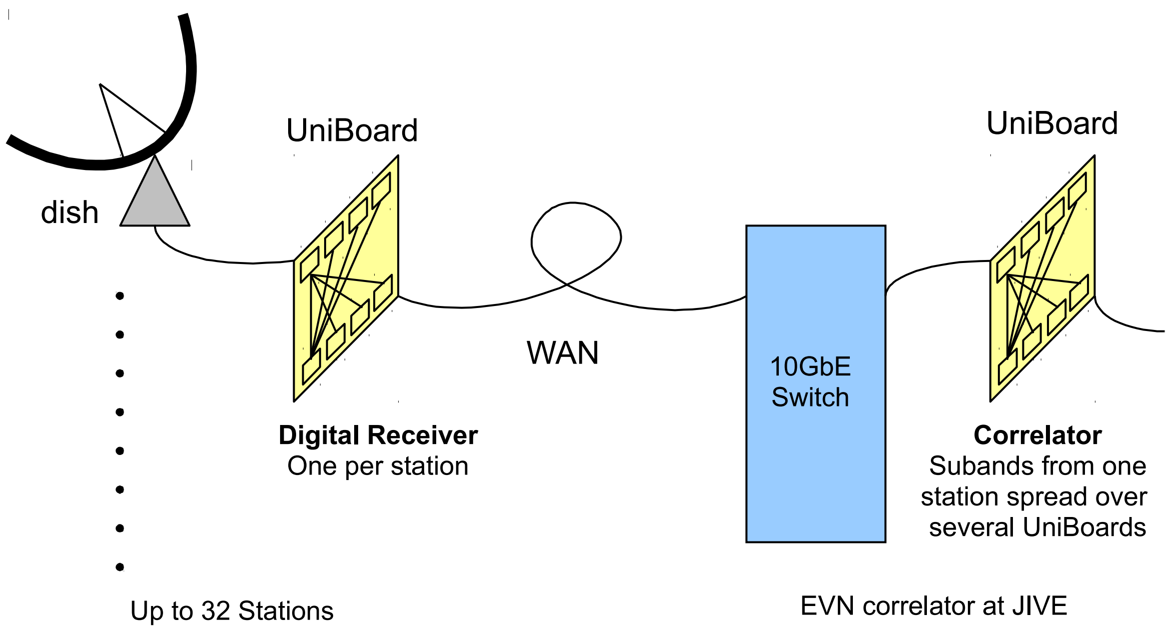


Figure 2.1: *UniBoards at the Stations Send Data to the Correlator Over a 10Gb Network*

The station data are distributed evenly between the four ‘Front Node’ (FN) FPGAs on a UniBoard as shown in Figure 2.2. The maximum data rate into each FN FPGA is

128MHz (bandwidth) x 2 (Nyquist) x 8 (bits) x 8 (stations) x 2 (pols.) = 32.8Gbps

which would require use of all four 10GbE ports. In fact only two 10GbE ports are instantiated in the design which limits the bandwidth to 64MHz at 8 bit sampling, or 128MHz at 2 and 4 bit sampling.

The FN FPGA performs all station-based processing, including compensating for network and geodetic delays, and conversion to the spectral domain using a polyphase filter bank and FFT. After the FFT the data are distributed amongst the ‘Back Node’ BN FPGAs with each BN receiving a quarter of the frequency points. The red, orange, green and blue lines in Figure 2.2 show the paths of each quartile of the frequency points from the FNs to the BNs. Each arrow represents a four lane serial path with a raw capacity of 25Gbps.

The BN chips corner-turn the data and then perform the correlation on the frequency ordered data. Two independent correlator engines which can each process 16MHz of bandwidth are placed in each BN chip for a total of 128MHz per UniBoard. The correlation products can be exported via either the 1GbE control port or a 10GbE CX4 connection.

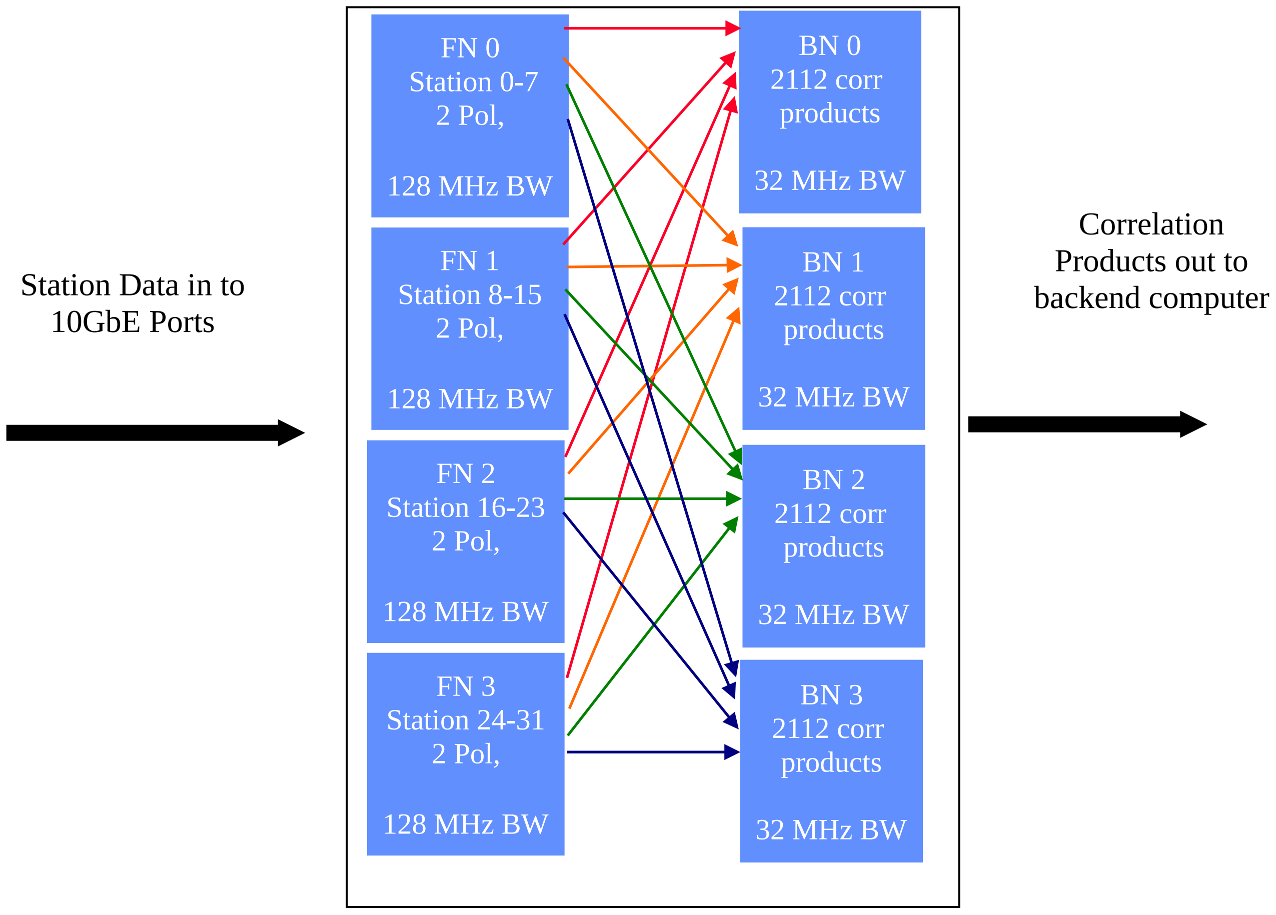

Station Data in to 10GbE Ports
FN 0 Station 0-7 2 Pol, 128 MHz BW
FN 1 Station 8-15 2 Pol, 128 MHz BW
FN 2 Station 16-23 2 Pol, 128 MHz BW
FN 3 Station 24-31 2 Pol, 128 MHz BW
BN 0 2112 corr products 32 MHz BW
BN 1 2112 corr products 32 MHz BW
BN 2 2112 corr products 32 MHz BW
BN 3 2112 corr products 32 MHz BW
Correlation Products out to backend computer

Figure 2.2: *Data flow in an EVN Correlator UniBoard*

The design has a fall-back position which is to process only 64MHz per UniBoard. This would be used for example if DDR3 memory modules large or fast enough to perform the corner-turning are not available, or if it proves difficult to place all the logic needed to process 128MHz.

For spectral line studies, it is possible to re-transmit a subset of the frequency points to another UniBoard for further high spectral resolution sub-banding. This can be done using a spare 10GbE port on either the FN or BN FPGA.

Figure 2.3 shows the main signal processing blocks in the design. The blocks coloured green are included in the first development phase intended to prove the signal flow through a single FN and BN. The grey blocks, including the delay model and the second correlator engine, will be integrated into the design after the first phase has been tested.

Figure B.1 in Appendix B shows the signal flow for one, two polarization station through the FN FPGA, while the signal flow in the BN is shown in Figure B.2.

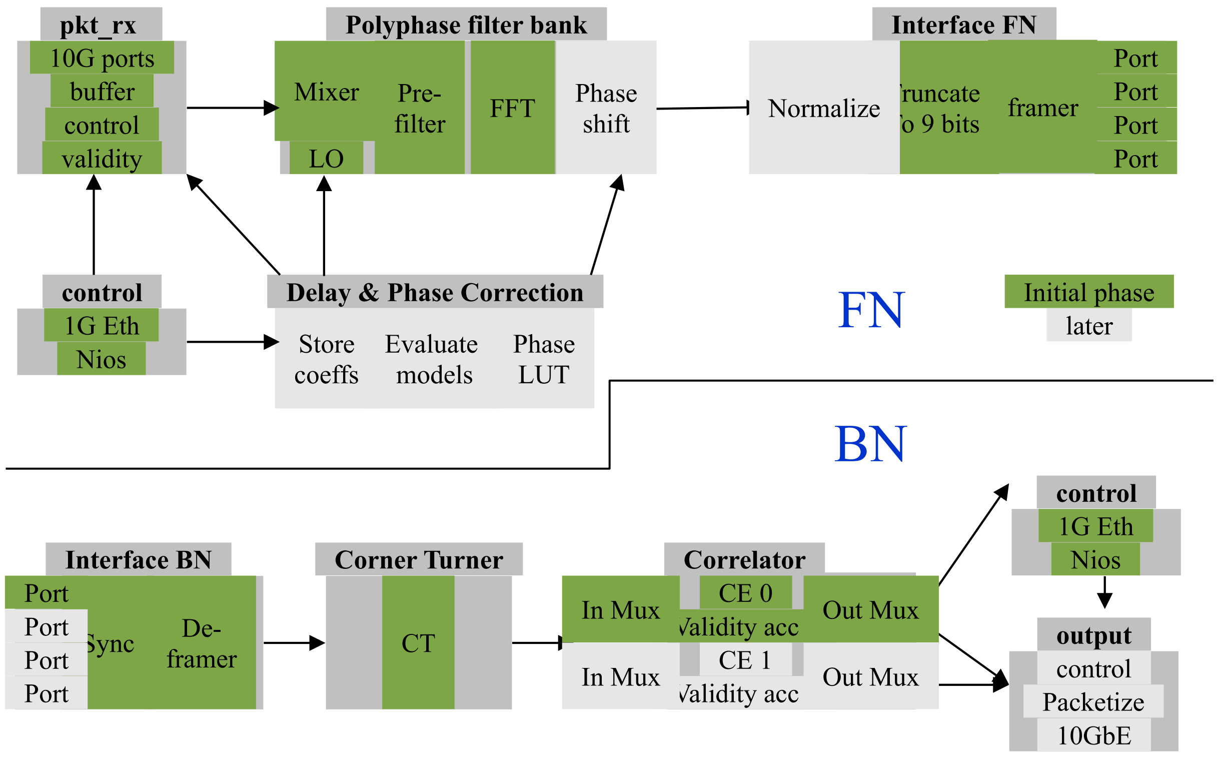


Figure 2.3: *EVN Correlator Development Stages*

Packet Reception

Data are transmitted from the stations in VDIF formatted jumbo UDP packets [1]. Any valid frame length up to 8192 bytes can be used as long as the frame length remains constant during an experiment. Network bandwidth and DDR3 space are used most efficiently when the frame length is 8192 bytes, but for compatibility with Mark V a frame length of 5000 bytes is considered here. When two polarizations are used, corresponding samples from each polarization may be packed into a single frame or transmitted separately.

The control computer sets up which stations, polarizations, and sub-bands are to be received on each 10GbE port. From this the receiver logic allocates a suitably sized buffer in DDR3 memory to store each station and sub-band (the two polarizations are stored together within a sub-band buffer).

Upon receipt of a packet the VDIF header is read to determine the originating station, sub-band(s) and time slot of the data frame. Each packet is stored in a new row in the DDR3 module whose address is calculated from the time stamp in the VDIF header. Since a DDR3 row contains 8192 bytes, some space is left unused when a smaller frame length is used. This process removes the network delay, expected to be at most 250ms [2], and re-orders any packets received out of time order. The data can then be read out sequentially a fixed time later starting when the buffer is half full.

The storage allocation of 5000 byte packets within a 4GB DDR3 module is illustrated in Figure 2.4

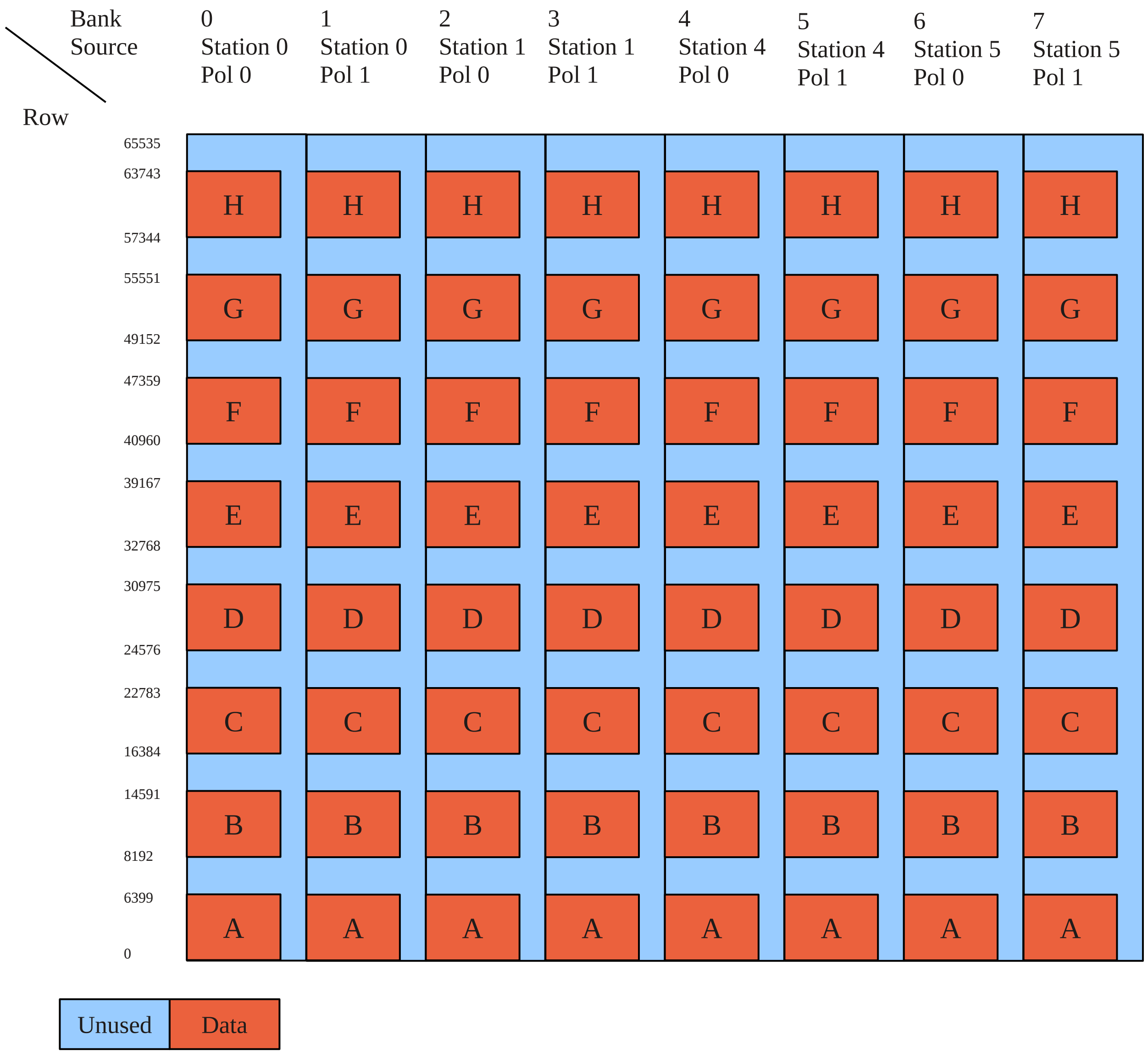


Figure 2.4: *Storage Allocation within DDR3 Buffer*

In Figure 2.4 the eight blue columns represent the 8 banks inside the DDR3 module. Each bank is subdivided into 65536 rows of 1024 columns. A 5000 byte packet fills 625 columns as shown by the horizontal dimension of the red blocks. The letters A to G denote individual 16MHz sub-bands; A, B, C & D are destined for processing by correlator engine 0 in each back node, while E, F, G & H will be sent to correlator engine 1.

Each sub-band starts on a 8192 row boundary and extends for 6400 rows. This is equivalent to one second of 8 bit, 2 seconds of 4 bit or 4 seconds of 2 bit sampled data.

A second DDR3 module stores the data arriving from Stations 2, 3, 7 & 8 in the same order.

The design aims to minimize the overhead involved when switching between read and write operations. On the write side there is a FIFO buffer large enough to store two 8192 byte packets from each active 10GbE port. Read access is blocked during the write operation so the whole packet can effectively be written to DDR3 in one burst. Simulation shows that it takes 735ns to write a 5000 byte packet which is equivalent to 850MT/s, compared with the theoretical maximum bandwidth of 1066MT/s

On the read side, small amounts of data must be read almost simultaneously from several stations, polarizations and sub-bands. Here the overhead is minimized by hopping between banks where possible.

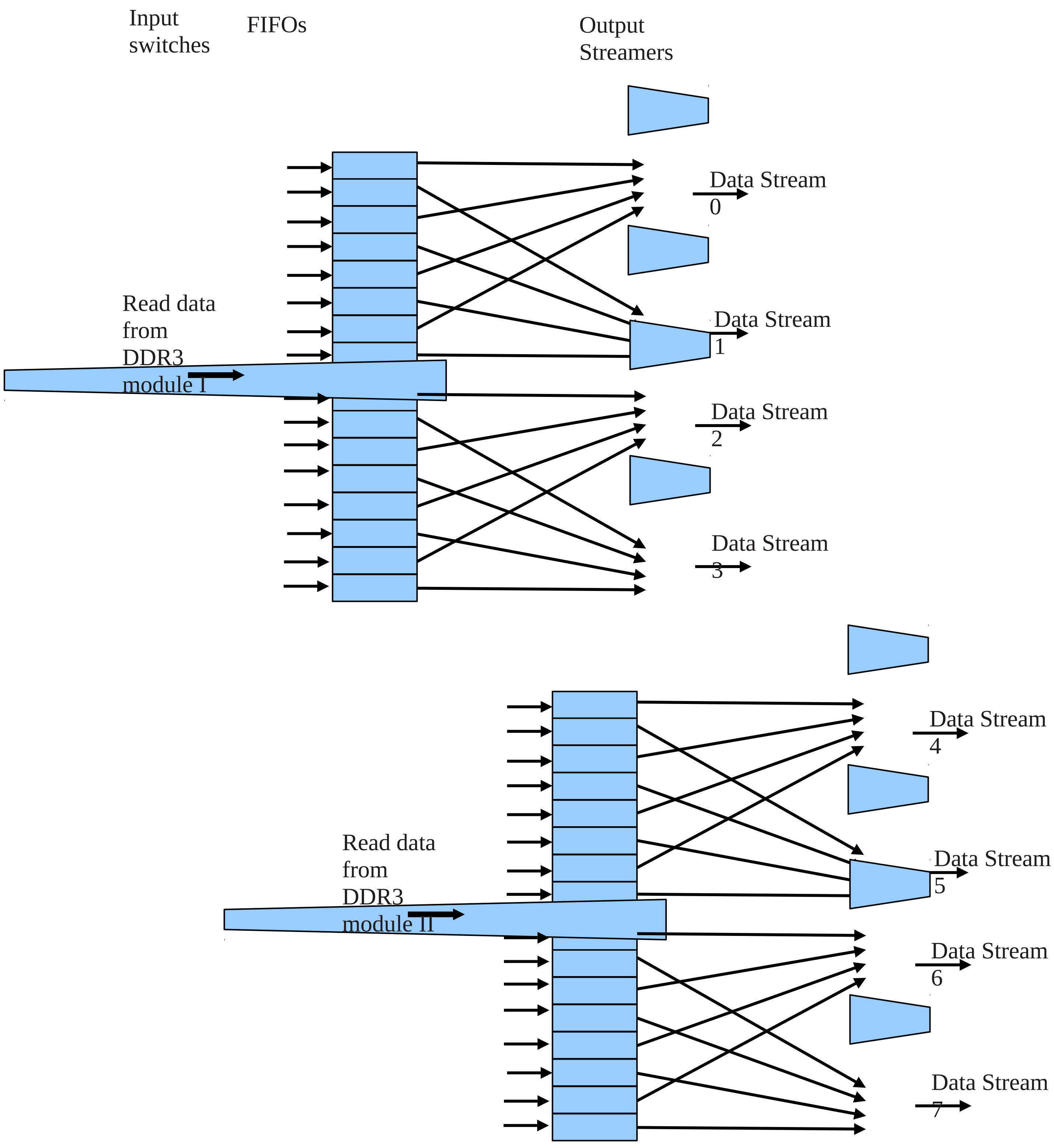


Figure 2.5: *Data Flow on the Read Side of the DDR3 Buffers*

The data flow between the DDR3 buffers and the filterbank inputs is shown in Figure 2.5. Data from each module are switched into one of sixteen 256 bit wide 16 word deep FIFOs depending on their source station, sub-band and polarization. The FIFO outputs, still 256 bits wide are fed to the output streamers, shown as multiplexer blocks in Figure 2.5. These blocks assemble the output streams by selecting samples from the input words, time multiplexing the samples from four sub-bands, and padding 2 and 4 bit samples with zeros to make a two sample, 16 bit wide, datastream output at 266.5MHz. This clock is derived from the Module I local clock and is used as a system clock in all eight filter banks. For Module II data the FIFOs are used to pass the data from the Module II local clock domain to this common system clock domain.

Data Validity

A table recording the validity of the incoming data is maintained in on-chip memory. Arriving packets are marked valid or not valid, so only a single bit is required to store the validity of all the data in one row of DDR3. A total of 819,200 bits are required for the 5000 bytes per packet case, though in the practical implementation 1Mbit is used.

All the bits in the validity store are initially '0'. When a valid packet arrives the bit for its row is set '1'. When, later, that row has been completely read out from the buffer its validity bit is reset '0'. If a valid packet does not arrive to fill that row by the time it is read again, the validity bit remains '0'.

As data are read from the buffer, the corresponding validity bits are sampled. Output validity signals are generated for each of the four sub-bands in each of the eight data streams. These outputs are generated with the same latency as the filterbank itself and so are fed directly to the framer. The algorithm is that if any data input contributing to a particular FFT period is invalid, all the frequency bin outputs of that FFT period are marked invalid.

Correlator Clock

The correlator clock is always asynchronous with respect to the stations, even though the station clocks are synchronized using maser time standards. For this reason the correlator clock must run slightly faster than real time, then be ‘synchronized’ to the incoming station data by inserting delays as necessary. The scheme is shown in Figure 2.6.

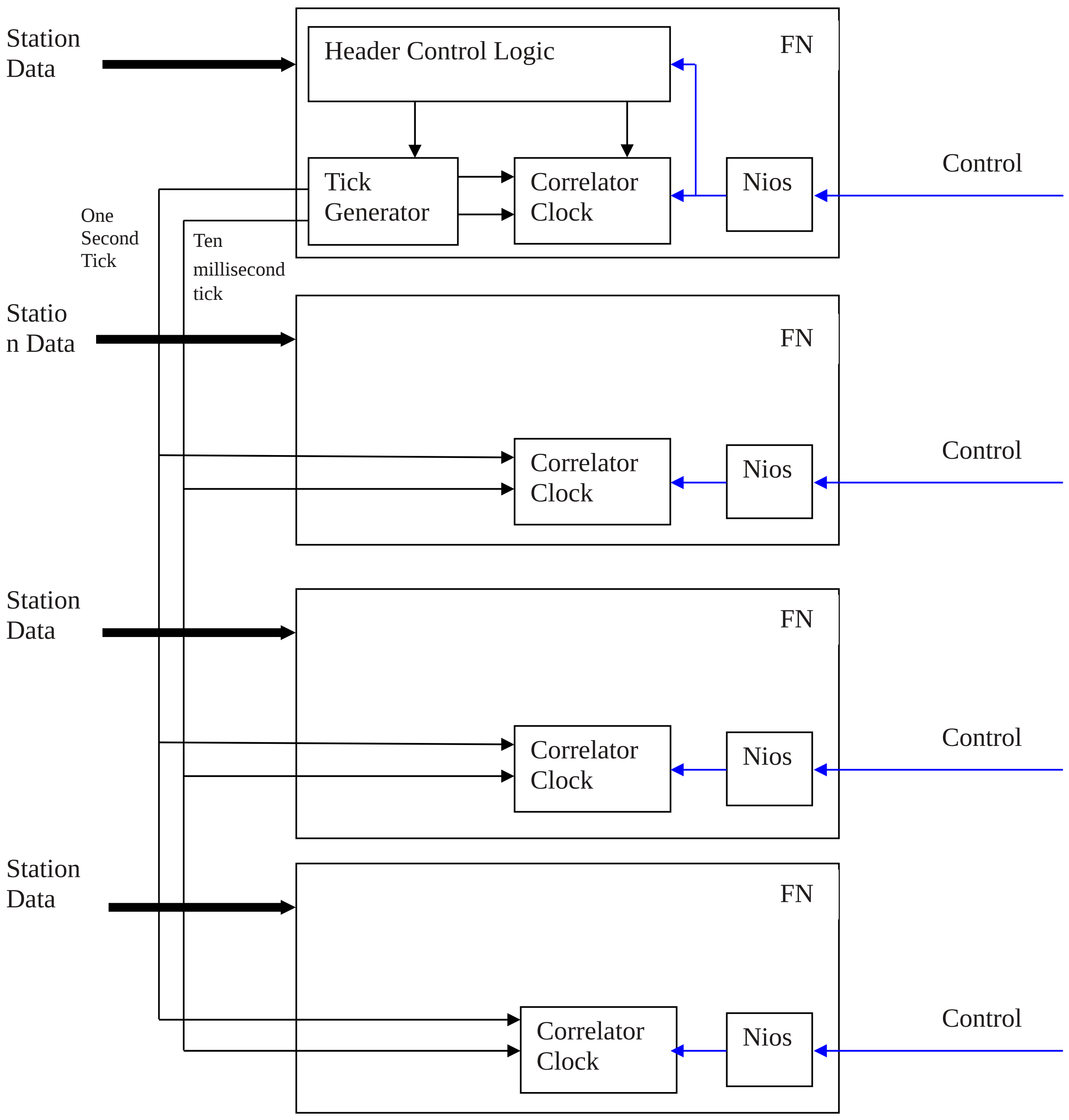


Figure 2.6: *Timing Synchronization Between FN FPGAs*

For the purpose of timing synchronization the control system sets one FN FPGA as ‘master’ and the other three as ‘slaves’. When station data starts to flow to the master the correlator time can be set with a suitable delay (typically 0.5s) after the station time. The master generates ticks every 1s and 10ms of correlator time and distributes them to the other FN chips via the INTA and INTB connections. The 1s tick is used to synchronize the clocks and delay models across all four FPGAs.

As the correlator clock is deliberately set to run faster than the station clocks, the DDR3 buffers would gradually empty were the clock allowed to free-run. To prevent this, the buffer size is monitored and delays inserted between ticks as necessary to keep the buffer stable. The 10ms tick allows this adjustment to be made on a finer resolution.

Pre-recorded Data

In principle pre-recorded data can be processed in batch mode:

- The correlator clock is preset to the time stamp in the record where processing is to start
- The correlator requests the disks to start sending data
- When the start time is reached, the data are stored in the FN DDR3 buffers
- The correlator requests the disks to stop sending data for individual stations and sub-bands as the buffers fill up
- Once all the data has arrived at all four FN chips, data flow is halted while they are processed

This avoids the need to synchronize disks but is quite inefficient – the correlator is idle for at least half the time. When the pre-recorded data rate is less than the capacity of a UniBoard, it is possible to speed up processing by running the correlator clock faster than real-time.

If the disk packs are synchronized then recorded data can be processed continuously as per real time data. Small, 100 millisecond scale, offsets in the arrival times of different pre-recorded channels can be removed in the same way as the network delays inherent in real-time data.

Delay Model and Phase Rotation

The control computer calculates a geodetic delay model for each station and transmits it to the FN FPGAs as coefficients of a quadratic polynomial of the form

$\tau = d_0 + d_1t + d_2t^2$
in which t is time during the period during which the coefficients d_0, d_1 and d_2 are valid, and τ is the delay correction for that station. A phase correction is required because the delay correction is done at sub-band frequency, not sky frequency. This is calculated as

$\phi = p_0 + p_1t + p_2t^2$

Using coefficients p_0, p_1 and p_2 calculated by the control computer. Both the delay and phase model coefficients are updated 32 times per second. The control computer calculates a seconds worth of coefficients in advance and transmit them soon after the preceding PPS tick. The coefficients are stored in a fifo buffer large enough to store two seconds worth of coefficients.

Coefficient Resolution, Storage and Bandwidth

Delay coefficients are transmitted with 32 bit resolution and phase coefficients with 48 bit resolution. Two seconds worth of coefficients occupies 6,144 bits to store a set of delay values and 9,216 bits to store a set of phase values.

Each FN can process eight stations, and initially there are four 16MHz sub-bands per station. Given that delay is calculated per station and phase per sub-band per station, the total storage requirement is 344,064 bits per FN chip.

Half this data must be transmitted to each FN chip every second, so the bandwidth required on the control port is 0.7Mbps per board. Sixty-four UniBoards would require 44Mbps which should be easily accommodated on a 1Gbps link

Evaluation of the Models

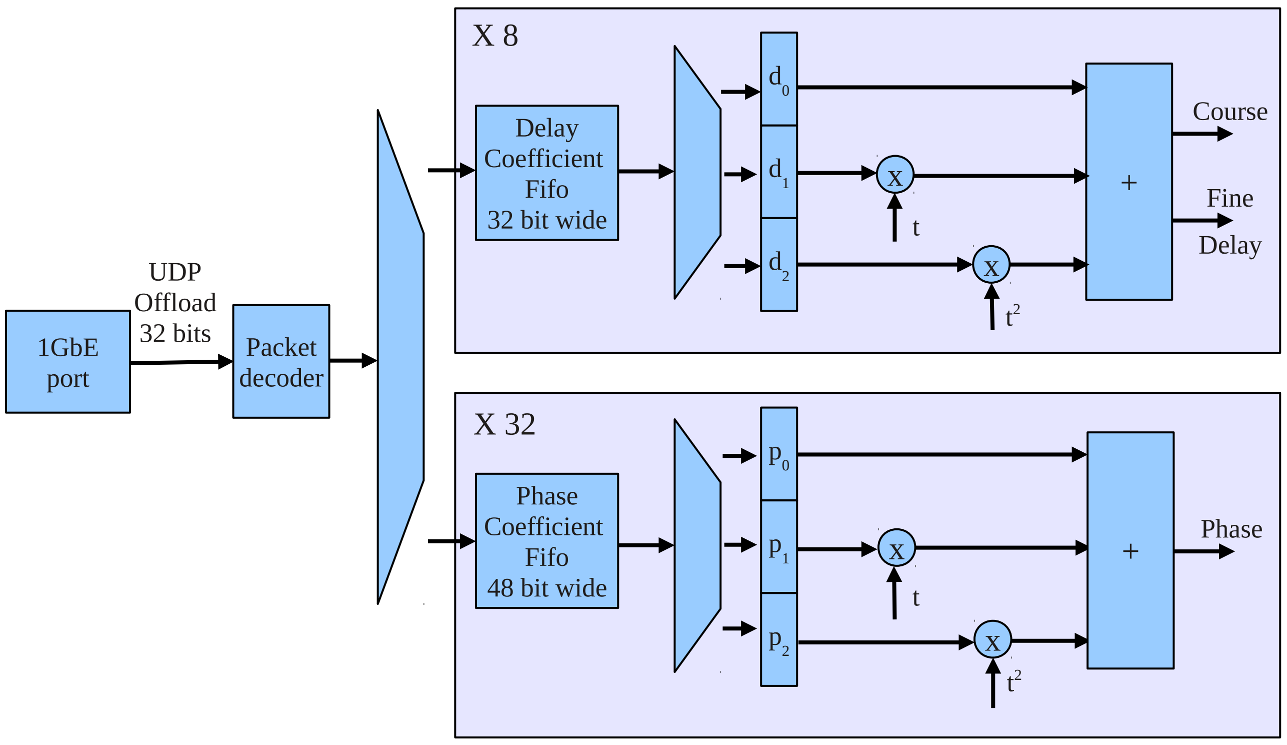


Figure 2.7: *Evaluation of the Delay and Phase Coefficients*

The coefficients are scaled so that they can be evaluated in correlator time, ie. over 1/32nd of a second, t increases from 0,1,2 … 1000000 for the case of a 16MHz sub-band. The delay model must be evaluated every eight sub-band samples and the phase model every sample. Since the data are processed at the correlator system clock rate (266.5MHz) rather than real time, the delay value for each station must be updated every 30ns and the phase for each sub-band every 3.75ns.

The precision required for the calculation is the input coefficient precision plus the precision needed for the range of t^2 . As the latter varies from 0 to 10^{12} (40 bits) at total of 72 bits and 88 bits are needed for the delay and phase respectively. The calculation is implemented using sequenced accumulators (DDS), although DSP multipliers could also be used.

Application of the Models

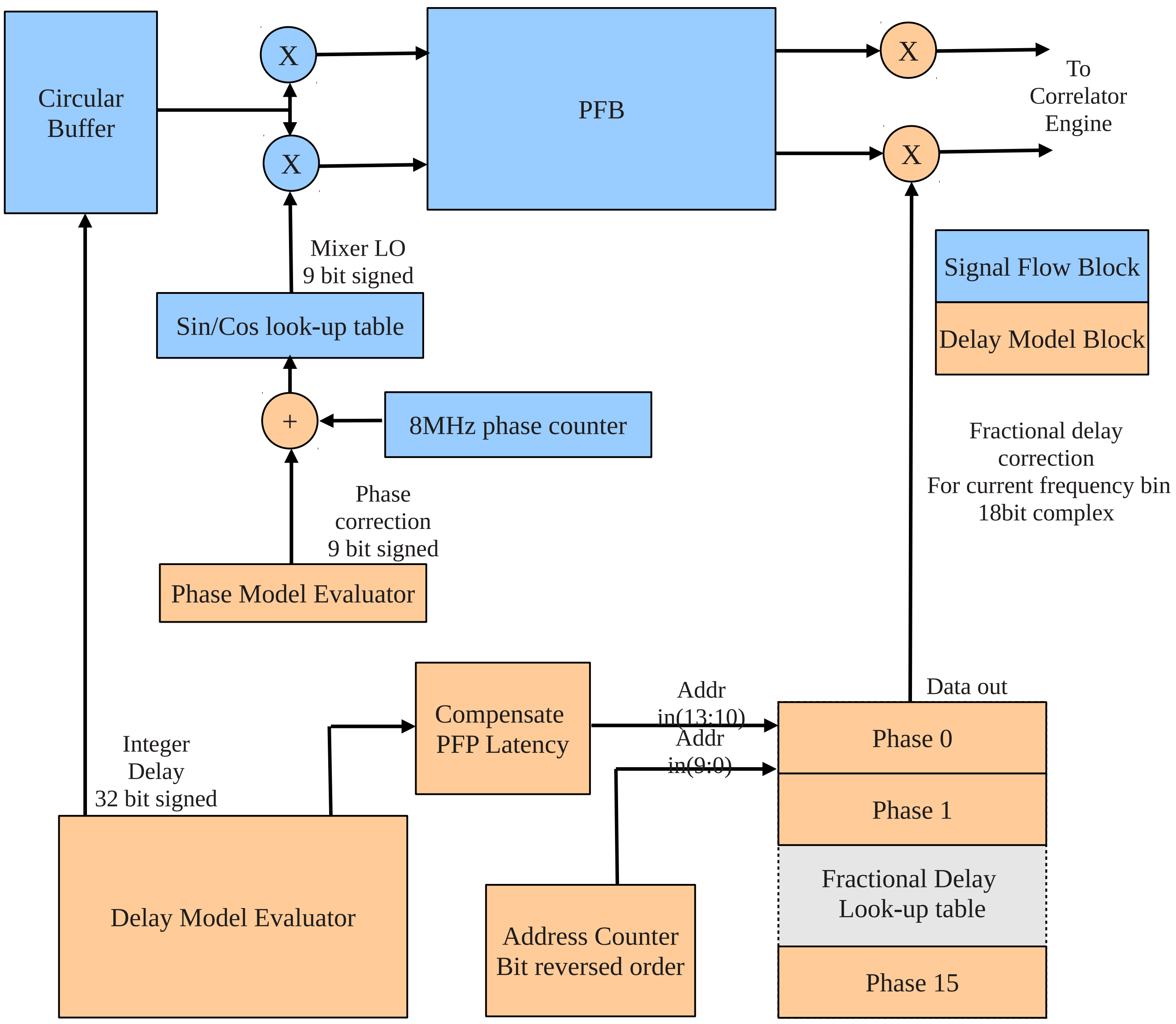


Figure 2.8: *Applying the Delay Models to the Data*

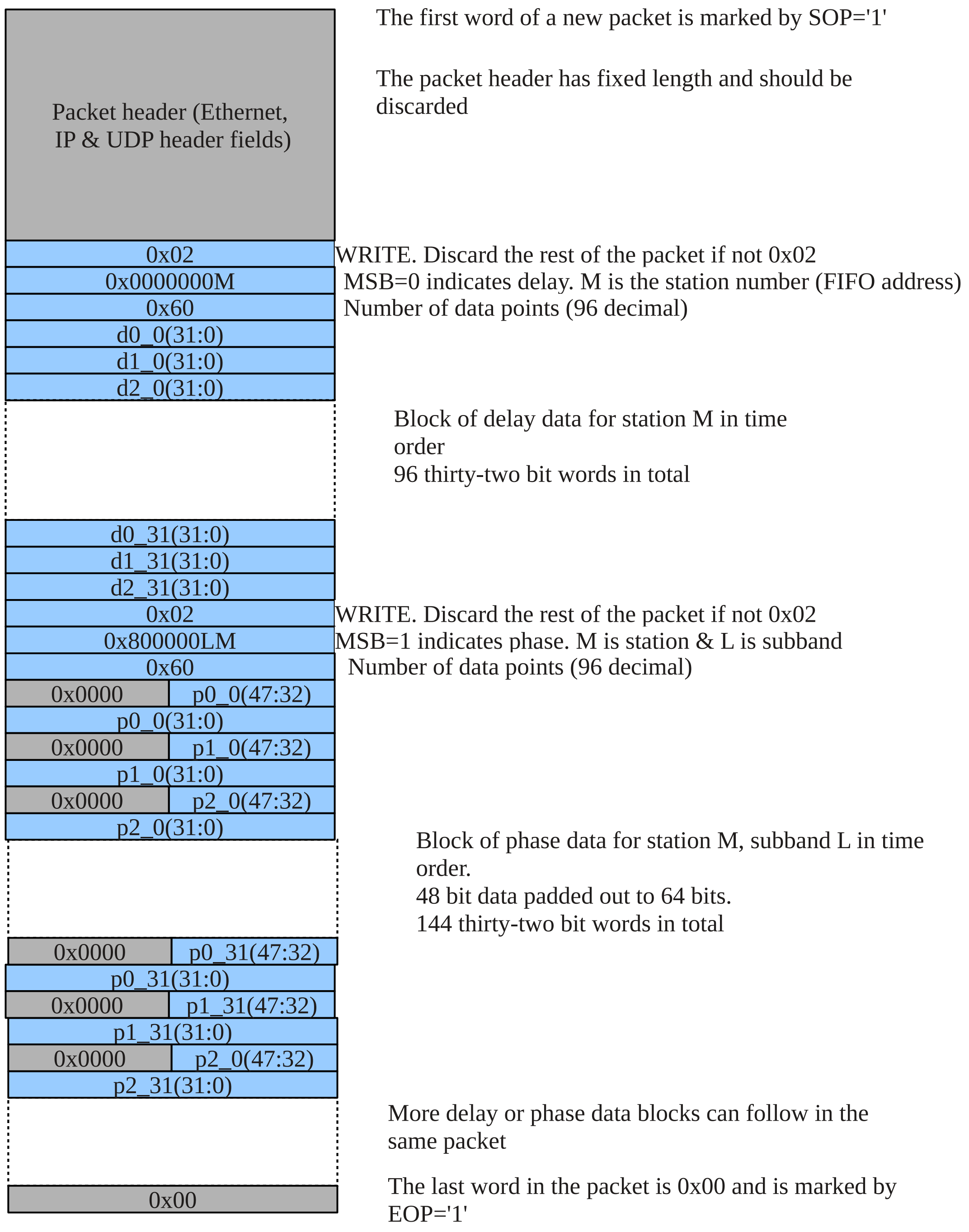
Delay
The calculated delay represents a time in seconds. It should be scaled such that it can be divided into an integral multiple of the sample rate of 31.25ns. The integer part is used as an offset to the read pointer to the circular data buffer. The four most significant bits of the fractional part represent the fine delay to an accuracy of 1/16th of a sub-band sample. This is used as the input to a look up table containing phase corrections to be applied to each frequency bin output from the PFB.

In the initial design the delay calculated at the centre of each 1024 sample FFT period is used, even though it can be updated every 8 samples. More frequent updating can be added later if necessary.

Phase
The phase model is scaled such that the full scale output represents 2pi of phase. The 9 most significant bits are added to the phase input to the relevant sub-band mixer at the input to the PFB

Coefficient Transmission

Coefficients are transmitted to the FPGAs via the UDP offload port of the 1GbE module. The packet format is shown below.



Packet header (Ethernet, IP & UDP header fields)
The first word of a new packet is marked by SOP='1'
The packet header has fixed length and should be discarded
0x02
WRITE. Discard the rest of the packet if not 0x02
0x0000000M
MSB=0 indicates delay. M is the station number (FIFO address)
0x60
Number of data points (96 decimal)
d0_0(31:0)
d1_0(31:0)
d2_0(31:0)
Block of delay data for station M in time order
96 thirty-two bit words in total
d0_31(31:0)
d1_31(31:0)
d2_31(31:0)
0x02
WRITE. Discard the rest of the packet if not 0x02
0x800000LM
MSB=1 indicates phase. M is station & L is subband
0x60
Number of data points (96 decimal)
0x0000
p0_0(47:32)
p0_0(31:0)
0x0000
p1_0(47:32)
p1_0(31:0)
0x0000
p2_0(47:32)
p2_0(31:0)
Block of phase data for station M, subband L in time order.
48 bit data padded out to 64 bits.
144 thirty-two bit words in total
0x0000
p0_31(47:32)
p0_31(31:0)
0x0000
p1_31(47:32)
p1_31(31:0)
0x0000
p2_0(47:32)
p2_31(31:0)
More delay or phase data blocks can follow in the same packet
0x00
The last word in the packet is 0x00 and is marked by EOP='1'

Figure 2.9: *UDP Delay/Phase Coefficient Packet Format*

Each write command contains one seconds worth of data for a particular station or sub-band. The address field is used to determine which FIFO to write the data to.

Hardware Interface

The signals shown in the table are used to connect the delay module to surrounding logic. Initially connections are provided for one station with four sub-bands. Additional stations will be accomodated when the delay model is integrated into the signal flow.

Signal name	*Direction and type*	*Purpose*
phase0	out std_logic_vector(8 downto 0)	Phase correction output sub-band 0
phase1	out std_logic_vector(8 downto 0)	Phase correction output sub-band 1
phase2	out std_logic_vector(8 downto 0)	Phase correction output sub-band 2
phase3	out std_logic_vector(8 downto 0)	Phase correction output sub-band 3
delay_frac	out std_logic_vector(17 downto 0)	Fine delay output as a phase correction
delay_integral	out std_logic_vector(31 downto 0)	Coarse delay output to the buffer
clk	in std_logic	266.5MHz clock
rst	in std_logic	Active high reset
one_pps	in std_logic	Start of second
start_of_fftperiod	in std_logic	Start of FFT period

Avalon ST Interface to the Eth 1G

An Avalon ST interface is used to connect to the UDP offload port of the 1G Ethernet module. See the 1Gb Ethernet Module Description document (ASTRON-RP-396) for details.

Polyphase Filter Bank (PFB)

Polyphase filter banks are commonly used in FX correlators and provide an efficient implementation of a near-rectangular FIR windowing filter and DFT [3]. The number of complex multiplications required to compute an N_P point DFT (as an FFT) is $N_P/2\log_2 N_P$, however in a PFB the filter and DFT run at the output rate which is a factor N_P slower than the input. The processing load is therefore proportional to $\log_2 N_P$, the input bandwidth, and the number of input channels.

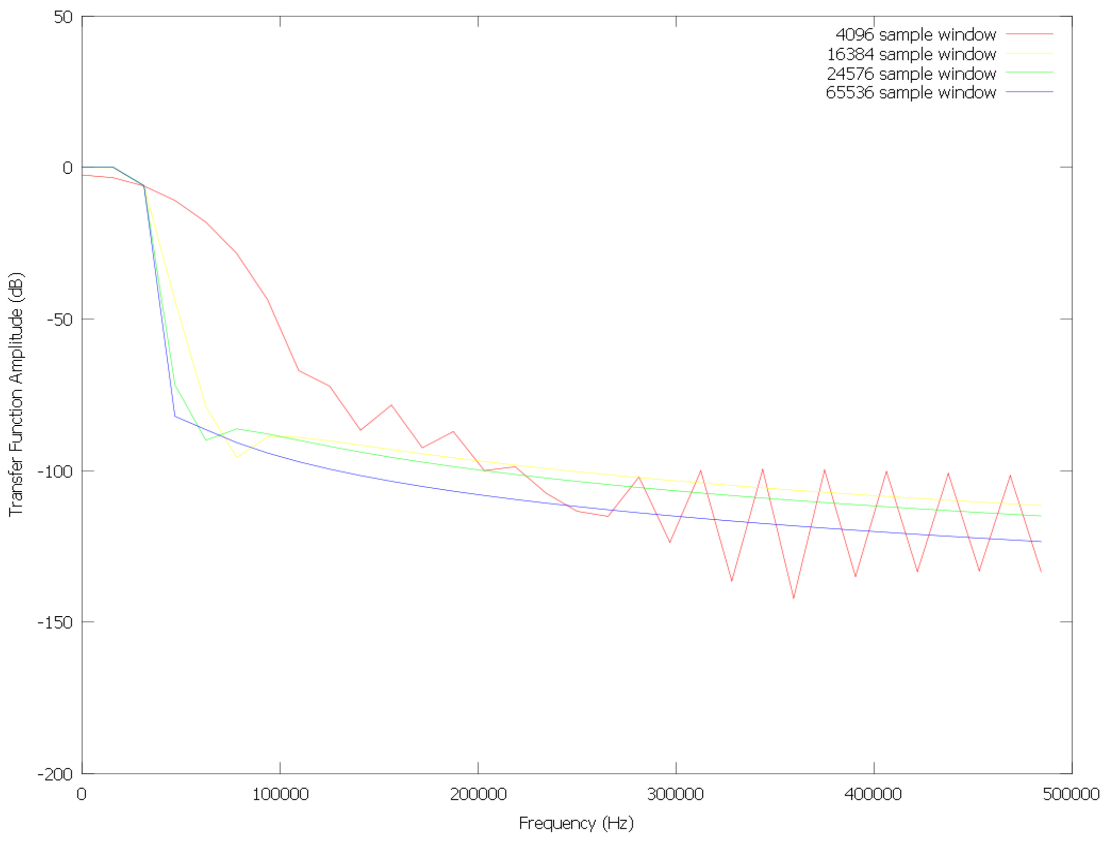


Figure 2.10: *Comparison of Four Lengths of Filter using a Blackman-Harris Window*

The window function is designed to improve the shape and selectivity of the filter bank compared to the raw FFT. A wider window provides a flatter pass-band and steeper cut-off, but at the cost of a higher storage requirement for both coefficients and taps. As an example, Figure 2.10 shows the effect of four Blackman-Harris windows of different lengths: N=4096, 16384, 24576 and 65536 taps. The nth window coefficient is calculated as:

$$W(n) = 0.44959 - 0.49364\cos(2\pi n/N) + 0.05677\cos(4\pi n/N).$$

In the example the windows are applied to a low pass FIR filter with a cut-off frequency of 31kHz for an input sample rate of 128 MSPS. This is analogous to a 4096 point filter bank in which the four windows correspond to 1x, 4x, 6x and 16x the FFT width. It can be seen in Figure 2.10 that the 4x and 6x windows provide successively steeper cutoffs and better sidelobe suppression than the 1x. The 16x contributes an additional 8 to 10 dB of sidelobe suppression, but at considerable cost in coefficient and tap storage.

Initial Implementation

The following mode has been selected for the initial implementation

Input sub-bands	4
Sub-band BW	16MHz
Frequency bins per sub-band	1024
Taps window length	6144

The four sub-bands are processed in a time-multiplexed fashion all the way through the filterbank. Defining a channel as one polarization from one station, there are eight 16MHz sub-bands per channel and 16 channels per FN, so it seems 32 filterbanks are needed. It is possible to place 8 such filterbanks per FN, however each filterbank can be clocked 266.5MHz which is more than four times the complex sample rate of the four sub-bands combined, so the desired throughput can be achieved by processessing the data in four batches.

In fact the filterbanks need only be clocked at 256MHz to process all the data in real time, however it is convenient to use 266.5MHz for several reasons:

- It is the DDR3 local bus clock so another clock domain transition is avoided
- It allows real time processing to be synchronized with the stations by processing faster than real time and periodically waiting for the stations to catch up.
- It accomodates overhead in the signal processing chain for example when switching between batches.

Figure 2.11 shows the order in which signals from the stations, polarizations, and sub-bands shown in the DDR3 buffer in Figure 2.4 are processed by the eight filterbanks in FN0. The synchronization pulse is generated every 10ms using the timestamps of the data coming from the stations. Each filterbank processes a 10ms slice of data from the four sources shown, then waits for the next synchronization pulse before processing the next slice.

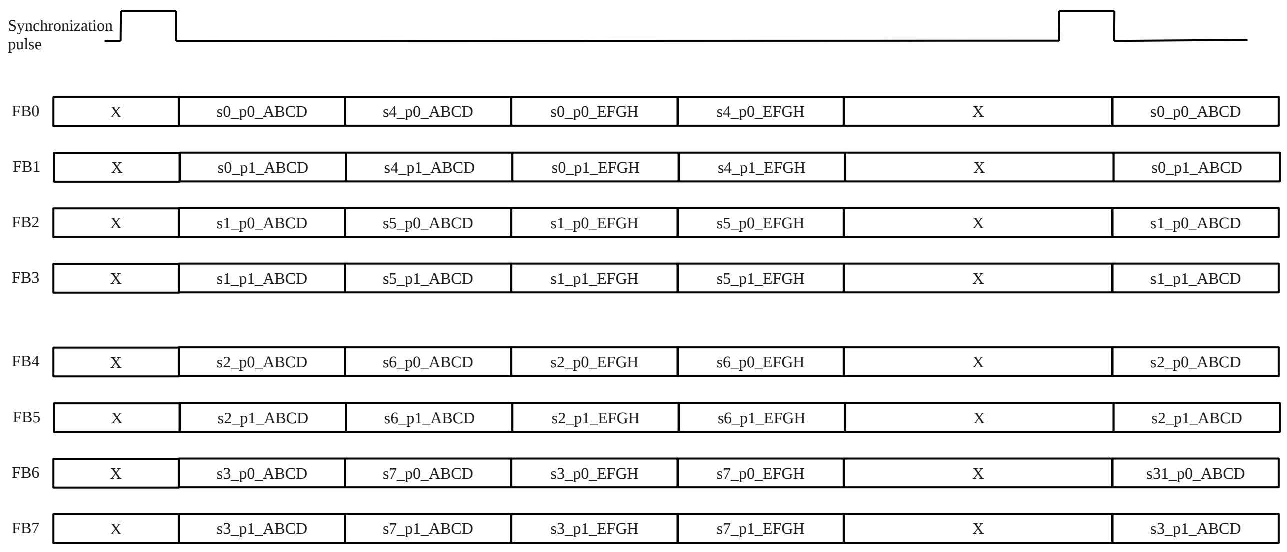

Synchronization pulse
FB0 X s0_p0_ABCD s4_p0_ABCD s0_p0_EFGH s4_p0_EFGH X s0_p0_ABCD
FB1 X s0_p1_ABCD s4_p1_ABCD s0_p1_EFGH s4_p1_EFGH X s0_p1_ABCD
FB2 X s1_p0_ABCD s5_p0_ABCD s1_p0_EFGH s5_p0_EFGH X s1_p0_ABCD
FB3 X s1_p1_ABCD s5_p1_ABCD s1_p1_EFGH s5_p1_EFGH X s1_p1_ABCD
FB4 X s2_p0_ABCD s6_p0_ABCD s2_p0_EFGH s6_p0_EFGH X s2_p0_ABCD
FB5 X s2_p1_ABCD s6_p1_ABCD s2_p1_EFGH s6_p1_EFGH X s2_p1_ABCD
FB6 X s3_p0_ABCD s7_p0_ABCD s3_p0_EFGH s7_p0_EFGH X s31_p0_ABCD
FB7 X s3_p1_ABCD s7_p1_ABCD s3_p1_EFGH s7_p1_EFGH X s3_p1_ABCD

Figure 2.11: *Data Processing Sequence of Filterbanks in FN0*

Storage Requirements

In the initial implementation a taps length of 6x is chosen to ensure that enough internal memory is available to place 8 filterbanks in each FN. Later it may be possible to increase this to 8x or even 10x depending on the requirements of the rest of the design.

Coefficients
An 18 bit coefficient representation is needed to approach the floating point performance shown in Figure 2.10, however only coefficients close to the central peak of the window require the full 18 bits of storage, so 18 bits is the worst case. Symmetry halves the storage requirement, so for the 24576 tap window we have

24576 (taps) x 18 (bits) / 2 (symmetry) = 221,184 bits

The coefficients need only be stored once for all stations, polarizations and the real and imaginary branches of the filter.

Taps
Since 14 bits of the complex mixer output are carried through the filter, each filterbank requires
24576 (taps) x 14 (bits) x 2 (complex) = 688128 bits.

DSP Resources for the Filterbank

Pre-filter
Each pre-filter consumes twelve 18 bit multipliers, with two more required for the real to complex mixer at the input

FFT
A multiplier-efficient FFT has been created by modifying the LOFAR pipelined FFT [4] to operate on time multiplexed data. When configured for naturally ordered inputs and bit-reversed outputs (since re-ordering can be done in the corner turner in the BN), the FFT requires sixteen 18 bit multipliers for each filterbank.

Signal Statistics and re-quantization

For initial testing the data from the filterbanks are truncated to 9 bits using a fixed offset. Later it will be possible to adjust the normalization level using statistical measurements on the data. Some methods of doing this are discussed here.

Figure 2.12 shows the locations of power measurement logic before and after the filter-bank. The mean squared power of each channel (station, polarization or sub-band) is measured before the complex mixer. The resources required per station/polarization are a 9 x 9 bit multiplier to square the signal and an accumulator to average the signal over time. A 1 second sample measurement on a 64MHz sub-band requires a 45 bit accumulator. Using general FPGA logic for the accumulator requires 45 logic cell registers, while four 9 x 9 bit multipliers fit within a half-DSP block; Assuming the multipliers at over 256MHz, power meters for eight two polarization stations occupy 720 logic cell registers plus 2 half DSP blocks per 64MHz of bandwidth. An additional 384 registers or memory bits are needed to hold the previous measurement to 24 bit accuracy during readout.

More information about the signals can be gained by calculating state count histograms, however the resources needed to do this on every channel are quite high. For example to record state counts over a 4096 sample period requires a 12 bit counter for each state – there are 256 for an eight bit counter – so a total of 3072 logic registers are needed. Sending this information back to the control computer would consume 98Mbps of bandwidth on the 1GbE port. It is feasible to implement one or two such state counters which can be switched to analyze any channel.

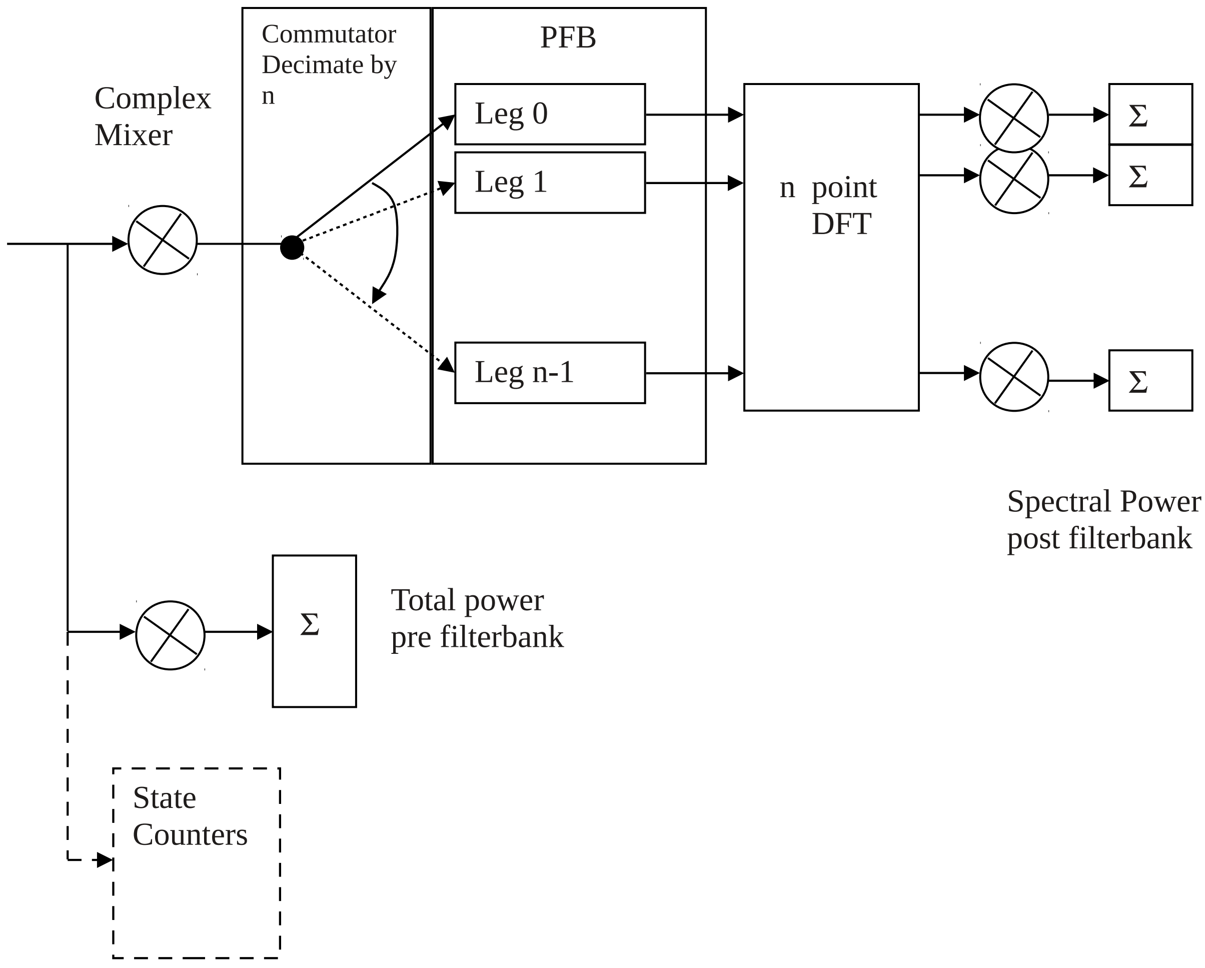


Figure 2.12: *Signal Statistics and Re-quantization*

After the filter-bank spectral power can be measured by multiplying each frequency point by its complex conjugate. This is done using 18 x 18 bit multipliers and requires 0.5 half-DSP blocks per station/polarization per 64MHz of bandwidth. The instantaneous powers can be averaged using a 48 bit accumulator with intermediate results stored in DDR3 memory. Normalizing the frequency points requires an additional 18 bit multiplier per 64MHz of bandwidth, although normalization to the nearest power of 2 could be done by shifting.

Data are then truncated to the chosen representation for input to the correlation engine. The control computer reads the normalization factors every second so that the true amplitude of the data can be reconstructed. Statistics on the truncated data can be done using state counters in the BN FPGAs which have more resources free than the FN.

FN BN Interface for EVN Correlator

This module forms an interface to distribute data from the FN filter bank outputs, across the mesh to the inputs to the BN corner turners. After 9 bit quantization the total maximum data output from a filterbank running at 266.5MHz is

2 (complex) x 9 (bits) x 266.5 = 4797Mbps

There are 8 filterbanks but the output is split four ways to the four Bns, thus the bandwidth required per FN-BN link is 9594Mbps. This fits on two transceiver lines running at double rate with 8b/10b encoding.

The overall scheme is shown in Figure 2.13 below

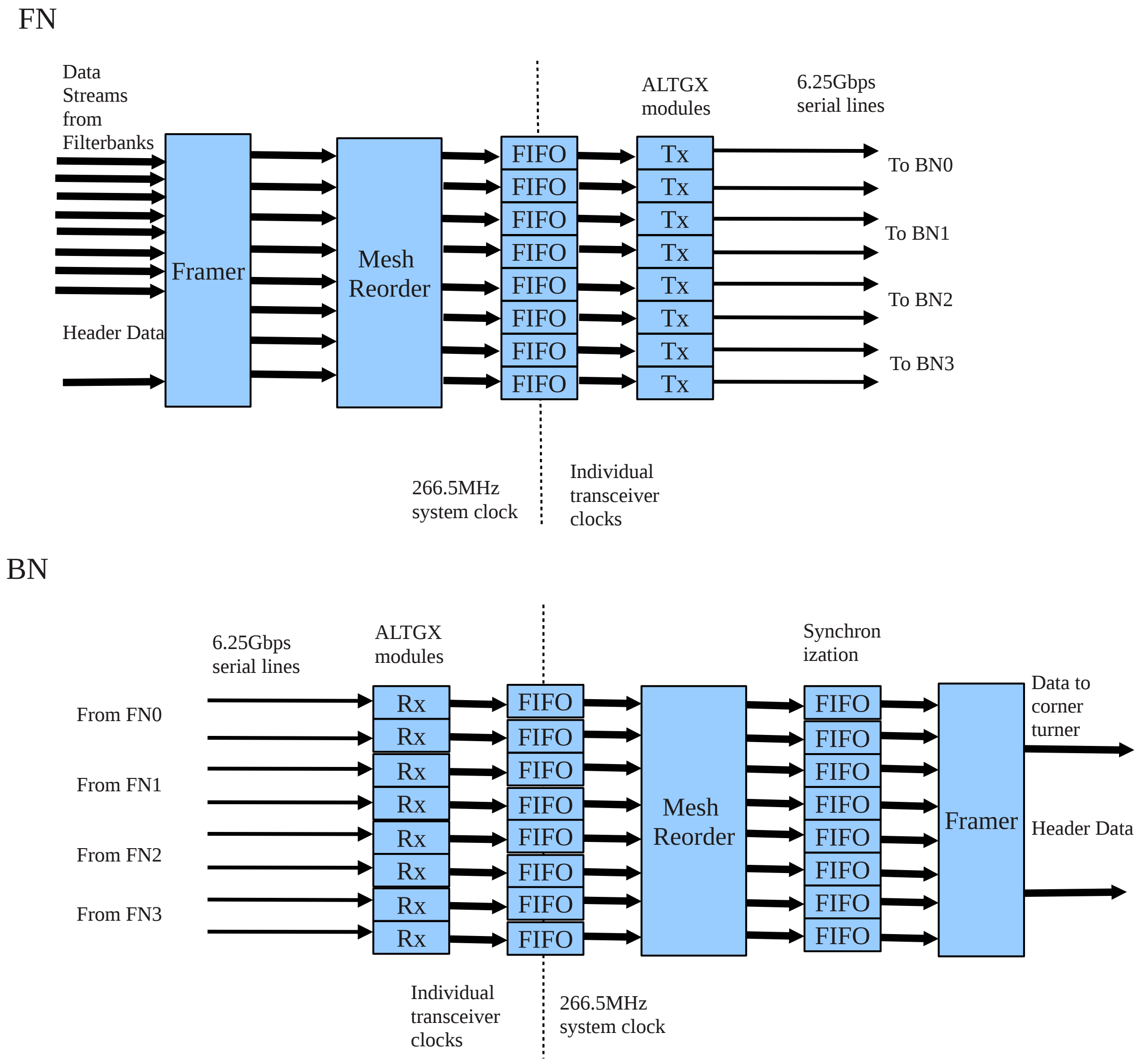


Figure 2.13: *Signal Flow in the FN-BN interface*

Framing

The data streams are de-multiplexed to separate the four time multiplexed sub-bands (A, B, C and

D) processed by each filter bank. All the data from sub-band A are sent to BN0, B to BN1, C to BN2 and D to BN3.

The *start_of_fftperiod* pulse is high during the first frequency bin output for sub-band A.

Four validity bits are associated with each data stream. They indicate the validity of each sub-band over the complete FFT period 1024 frequency bin outputs. The validity bits for the new fft period should be sampled when *start_of_fftperiod* is high.

Validity0(0) – sub-band A in data stream 0
Validity0(1) – sub-band B in data stream 0
Validity0(2) – sub-band C in data stream 0
Validity0(3) – sub-band D in data stream 0

.
.
.

Validity7(0) – sub-band A in data stream 7
Validity7(1) – sub-band B in data stream 7
Validity7(2) – sub-band C in data stream 7
Validity7(3) – sub-band D in data stream 7

If the validity bit is '0', the data for that sub-band is ignored and substituted by “000000000000000000” for the entire FFT period.

The data destined for each BN are formed into two frames as shown. The two frames combined contain all the data for one sub-band from all 8 filter banks for one FFT period.

Frame one contains data from streams 0 to 3

Fft period within The integration period 32 bits	Source 0 8 bits	Source 1 8 bits	Source 2 8 bits	Source 3 8 bits	Time of first sample in frame 64 bits

Validity 0 1 bit	Validity 1 1 bit	Validity 2 1 bit	Validity 3 1 bit	Padding 28 bits	Data stream 0 Frequency bin 0 18 bits	Data stream 1 Frequency bin 0 18 bits	Data stream 2 Frequency bin 0 18 bits	Data stream 3 Frequency bin 0 18 bit

	Data stream 0 Frequency bin 1023 18 bits	Data stream 1 Frequency bin 1023 18 bits	Data stream 2 Frequency bin 1023 18 bits	Data stream 3 Frequency bin 1023 18 bit

Frame two contains data from streams 4 to 7

Fft period within The integration period 32 bits	Source 4 8 bits	Source 5 8 bits	Source 6 8 bits	Source 7 8 bits	Time of first sample in frame 64 bits

Validity 4 1 bit	Validity 5 1 bit	Validity 6 1 bit	Validity 7 1 bit	Padding 28 bits	Data stream 4 Frequency bin 0 18 bits	Data stream 5 Frequency bin 0 18 bits	Data stream 6 Frequency bin 0 18 bits	Data stream 7 Frequency bin 0 18 bit

	Data stream 4 Frequency bin 1023 18 bits	Data stream 5 Frequency bin 1023 18 bits	Data stream 6 Frequency bin 1023 18 bits	Data stream 7 Frequency bin 1023 18 bit

Each frame is 73888 bits long. Each is fed to the input of one of the transceivers transmitting to the BN.

Mesh re-ordering and Transceiver Links

The mesh re-ordering module is used to unwind the complicated mapping between the transceivers and the destination Bns. It has to be done on both sides of the link.

Small, 8 word deep FIFOs are used at both sides of the link to pass the data between the common system clock domain and the individual transmit and receive clock domains provided by each ALTGX transceiver.

Since 8b/10b encoding is used, idle patterns are inserted into the data stream when no data is available to transmit. These are detected as control symbols by the receiver and used to regenerate a data valid signal in the BN.

In the clock cycle before the start of a new frame a unique control pattern is inserted into the data stream. This is used for re-synchronizing the data streams at the BN.

Synchronization

Three levels of synchronization are required at the BN. First the ALTGX must recognize a word alignment pattern to determine the start of the 32 bit word in the serial bitstream. Secondly the ALTGX requires a byte synchronization pattern to ensure the date output always starts on the correct 16 bit word. Finally data from the eight ALTGX receivers must be aligned. Since the four Fns do not necessarily transmit at the same time, and differences between the transceiver traces on the PCB can cause an offset of one or more words.

All three types of synchronization are achieved using the control word transmitted at the start of each frame. The ALTGX automatically detects the word alignment ,byte synchronization and outputs a pattern_detect pulse which is used to align the data streams in the second bank of FIFOs shown in Figure 2.13.

If only 1, 2 or 3 Fns are in use, the synchronization is only done on 2, 4 or 6 data streams respectively, and the outputs of the unused streams is blanked to '0'.

Corner Turner

Overview

Due to the limited storage available in the FPGA it is not possible to store post-correlation products on-chip. Neither is the DDR3 memory fast enough for this purpose. Instead data are 'corner-turned' so they can be processed in frequency order.

Data for a whole integration period are stored in DDR3 as they arrive from the FN. Then they are read out and correlated one frequency bin at a time. The frequency bin products are dispatched to the backend processor while correlation of the next frequency bin begins.

The two BN DDR3 modules are operated in a ping-pong configuration to permit a continuous flow of data. The integration time is limited by the size of DDR3: 1 second of data from 32 stations requires 4.6GB per module, so 8GB modules would be needed. A block diagram of the corner turner is shown in Figure 2.14.

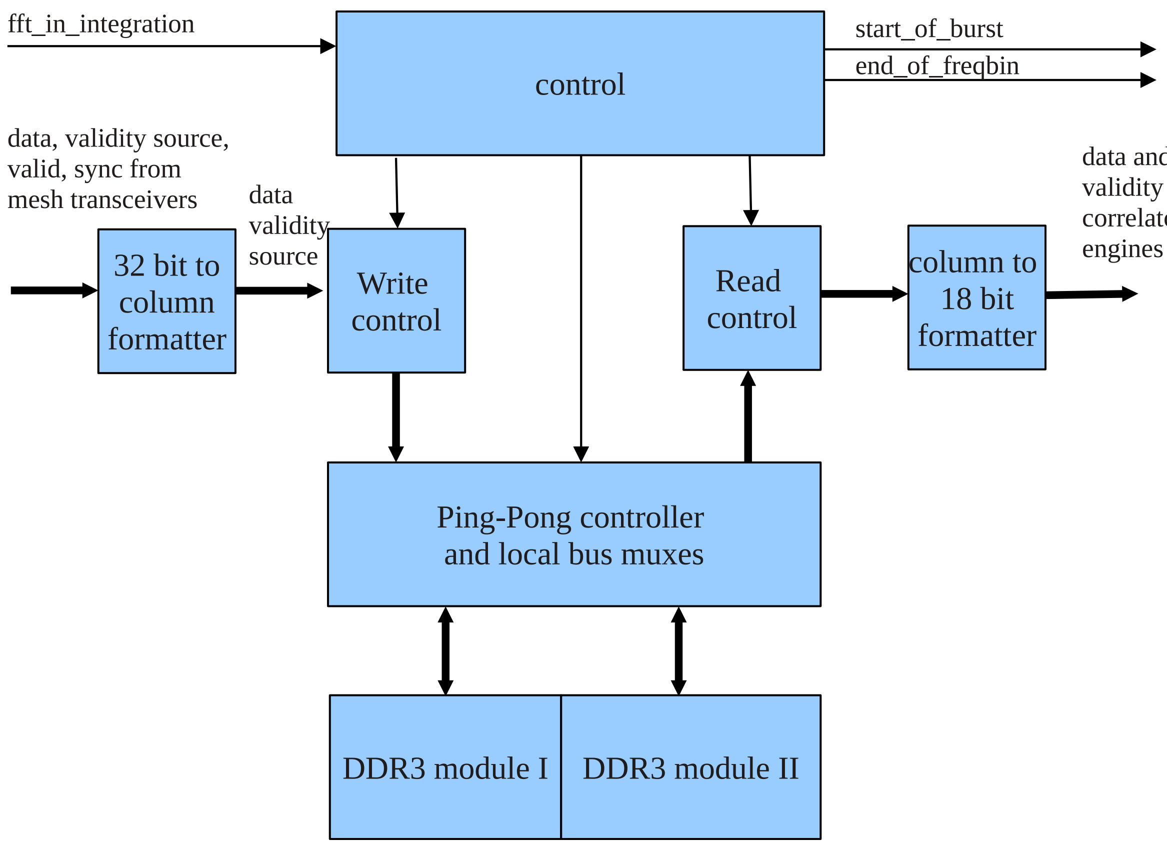

fft_in_integration
control
start_of_burst
end_of_freqbin
data, validity source, valid, sync from mesh transceivers
data validity source
32 bit to column formatter
Write control
Read control
column to 18 bit formatter
data and validity to correlator engines
Ping-Pong controller and local bus muxes
DDR3 module I
DDR3 module II

Figure 2.14: *Block Diagram of Corner Turner*

Figures 2.15 and 2.16 show the general scheme for ordering the data in the DDR3 memory so they can be written and read efficiently. In the write case, the blocks n, n+1 and so on represent complete sets of frequency points from all stations/polarizations. Each block is written to one or more rows within a bank. There is an overhead associated with changing rows within a bank, but this should be quite small (around 50ns) compared to the time to fill a row (1280ns at 800MT/s). Subsequent blocks are written to the next bank, wrapping back round to bank 0 after bank 7 as shown in Figure 2.9.

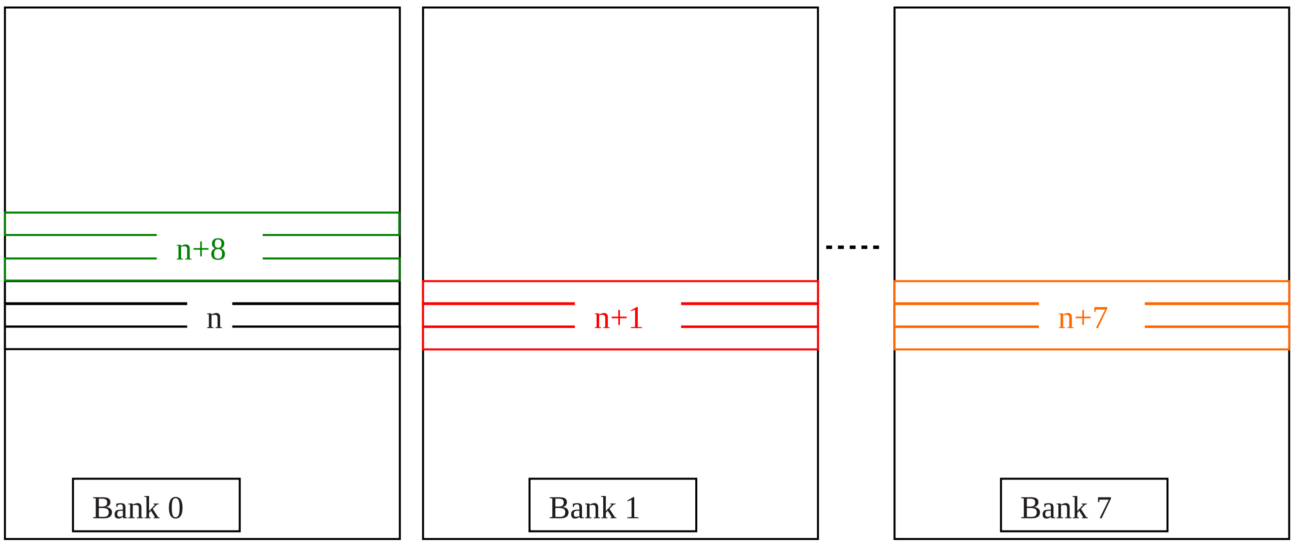


Figure 2.15: *Blocks of frequency points are written to the DDR3 in time order*

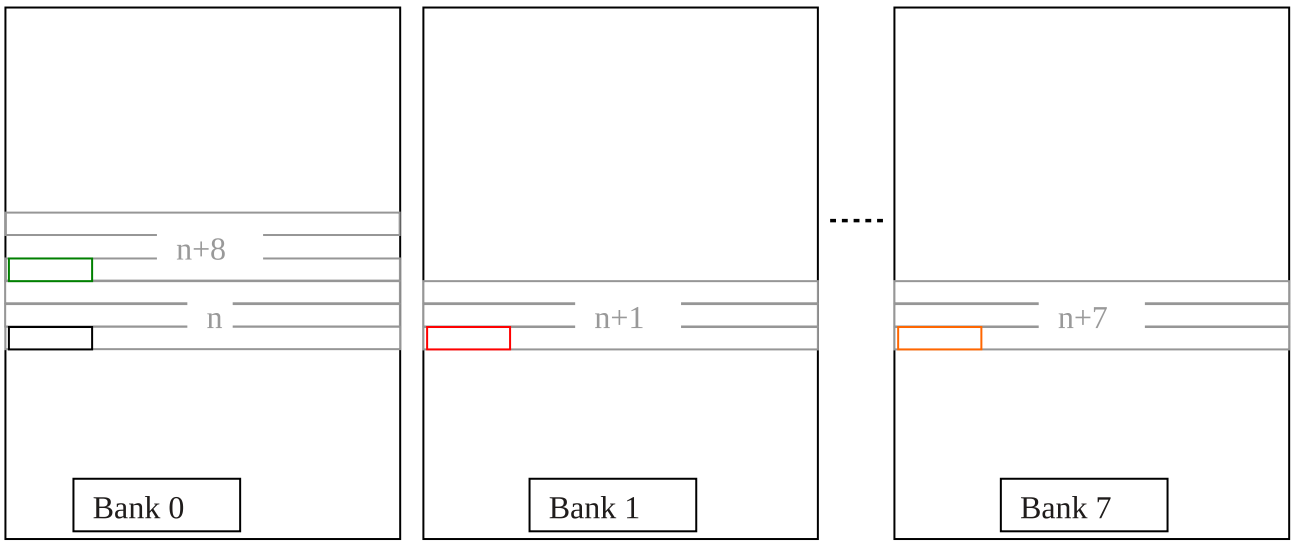


Figure 2.16: *Data read from the DDR3 in frequency order*

Data are read out of the DDR3 one frequency point at a time from each block as shown in Figure 2.16. Because the next point always comes from a different bank it is possible to activate its row before closing the previous one. This avoids the overhead which would be incurred if the next data came from another row within the same bank, so data can be read out continuously at burst rate.

Hardware Interface From Mesh Transceivers

The table below lists the signals which carry the de-framed data from the mesh transceivers to the corner-turner. Figure 2.17 shows the timing relationship between them and the contents of the first few data words.

Signal name	*Direction*	*Width*	*Purpose*	*Source*
sysclk	in	1	200MHz external reference clock	top
clk	out	1	266.5MHz clock	ct
rst	in	1	Active high reset	top
valid	in	1	Data streams valid when high	tcvr
sync	in	1	Common sync for all streams	tcvr
data_fn0_0	in	32	Data stream from FN0 frame 0	tcvr
data_fn0_1	in	32	Data stream from FN0 frame 1	tcvr
data_fn1_0	in	32	Data stream from FN1 frame 0	tcvr
data_fn1_1	in	32	Data stream from FN1 frame 1	tcvr
data_fn2_0	in	32	Data stream from FN2 frame 0	tcvr
data_fn2_1	in	32	Data stream from FN2 frame 1	tcvr
data_fn3_0	in	32	Data stream from FN3 frame 0	tcvr
data_fn3_1	in	32	Data stream from FN3 frame 1	tcvr
validity_fn0_0	in	4	Validity bits for the FN0 frame 0	tcvr
validity_fn0_1	in	4	Validity bits for the FN0 frame 1	tcvr
validity_fn1_0	in	4	Validity bits for the FN1 frame 0	tcvr
validity_fn1_1	in	4	Validity bits for the FN1 frame 1	tcvr
validity_fn2_0	in	4	Validity bits for the FN2 frame 0	tcvr
validity_fn2_1	in	4	Validity bits for the FN2 frame 1	tcvr
validity_fn3_0	in	4	Validity bits for the FN3 frame 0	tcvr
validity_fn3_1	in	4	Validity bits for the FN3 frame 1	tcvr
source_fn0_0	in	32	Source codes for FN0 frame 0	tcvr
source_fn0_1	in	32	Source codes for FN0 frame 1	tcvr
source_fn1_0	in	32	Source codes for FN1 frame 0	tcvr
source_fn1_1	in	32	Source codes for FN1 frame 1	tcvr
source_fn2_0	in	32	Source codes for FN2 frame 0	tcvr
source_fn2_1	in	32	Source codes for FN2 frame 1	tcvr
source_fn3_0	in	32	Source codes for FN3 frame 0	tcvr
source_fn3_1	in	32	Source codes for FN3 frame 1	tcvr
fft_in_integration	in	32	Count from 0 at start of integration	tcvr
time_stamp	in	64	Time of first sample in frame	tcvr

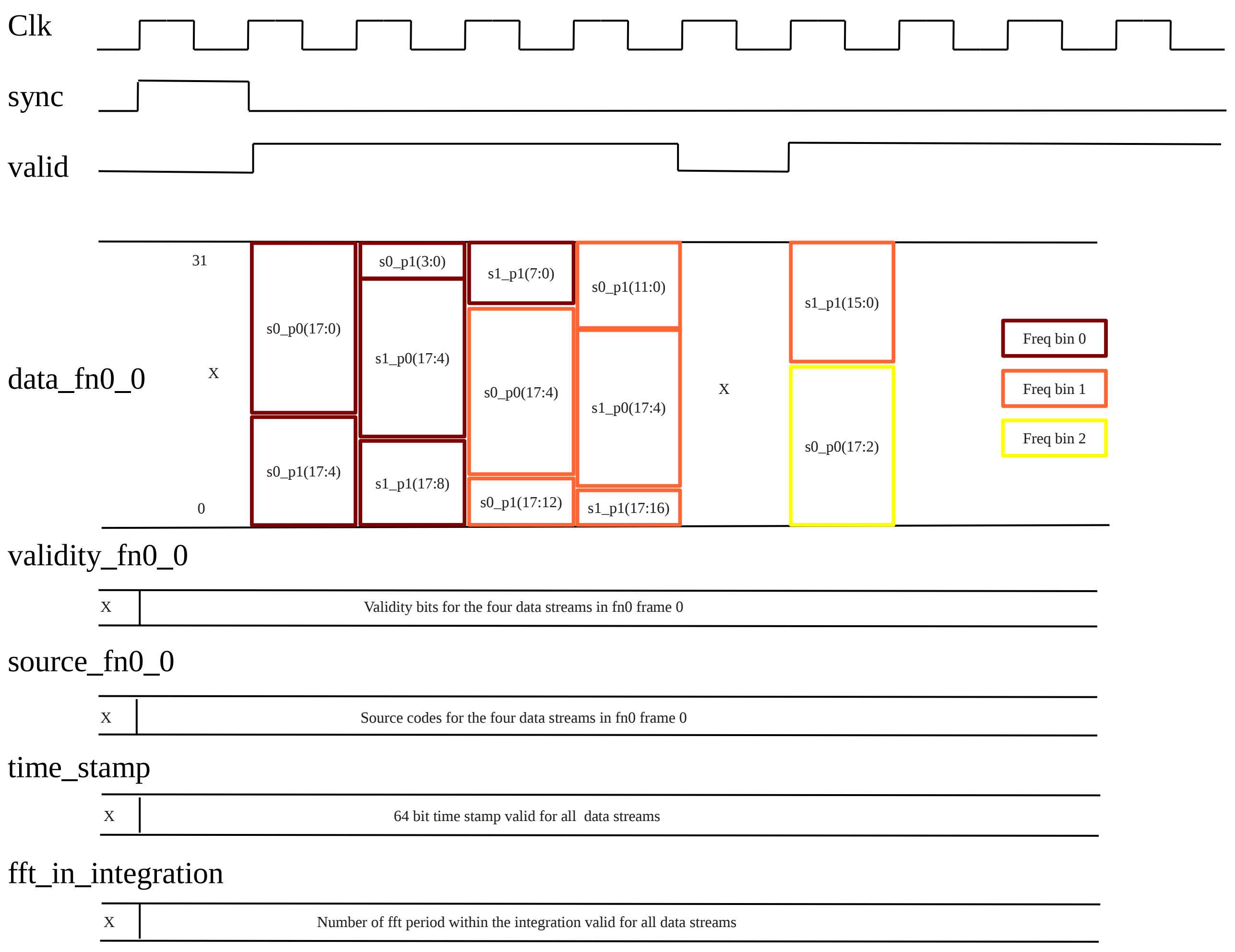


Figure 2.17: *Timing Diagram for Signals from the Mesh Transceivers to the Corner Turner*

Data Storage Format

1. Allocation of columns within a row

A whole integration period of data is accumulated in DDR3 memory. A 4GB DDR3 module contains 8 banks of 65,536 rows. Each row is divided into 1024 columns each of which stores 64 bits. The allocation of data to the first 96 columns in the first row of bank 0 is shown below

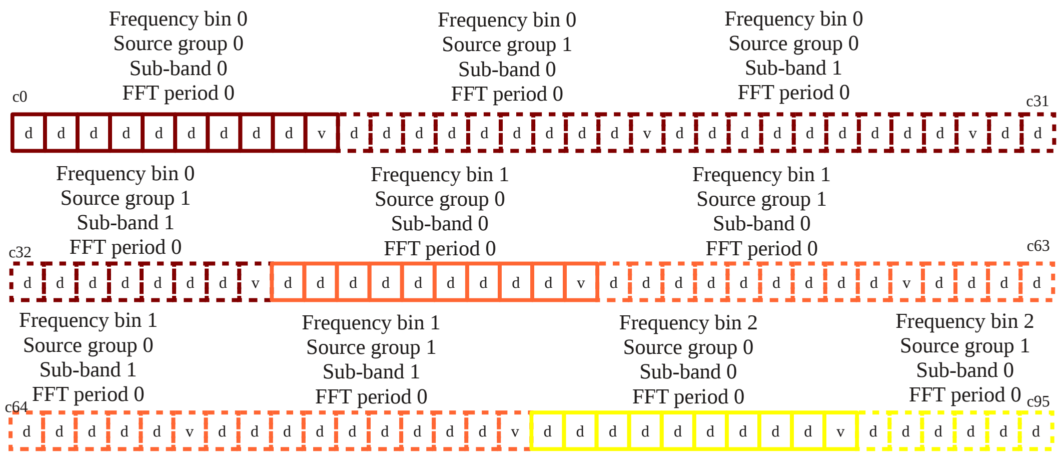


Figure 2.18: *Allocation of Columns within a Row*

The data for one frequency bin from all four Fns fits in 9 columns plus 1 column for the validity bits. Since the FN filter banks can process data from four sources sequentially a total of 40 columns are needed. The source groups contain the following data.

	Source Group 0	Source Group 1
from FN0	s0_p0	s4_p0
	s0_p1	s4_p1
	s1_p0	s5_p0
	s1_p1	s5_p1
	s2_p0	s6_p0
	s2_p1	s6_p1
	s3_p0	s7_p0
	s3_p1	s7_p1
from FN1	s8_p0	s12_p0
	s8_p1	s12_p1
	s9_p0	s13_p0
	s9_p1	s13_p1
	s10_p0	s14_p0
	s10_p1	s14_p1
	s11_p0	s15_p0
	s11_p1	s15_p1
from FN2	s16_p0	s20_p0
	s16_p1	s20_p1
	s17_p0	s21_p0
	s17_p1	s21_p1
	s18_p0	s22_p0
	s18_p1	s22_p1
	s19_p0	s23_p0

	s19_p1	s23_p1
from FN3	s24_p0	s28_p0
	s24_p1	s28_p1
	s25_p0	s29_p0
	s25_p1	s29_p1
	s26_p0	s30_p0
	s26_p1	s30_p1
	s27_p0	s31_p0
	s27_p1	s31_p1

The two sub-bands 0 and 1 are destined for correlator engines 0 and 1 respectively.

2. Allocation of data samples to columns

Since there are eight 32 bit inputs from the transceivers, and the DDR3 local bus data width is 256 bits, it is convenient to store the data in the order they arrive. The 18 bit complex samples must be restored later when the data are read out. The data stored in the first 11 columns of row 0 are shown below.

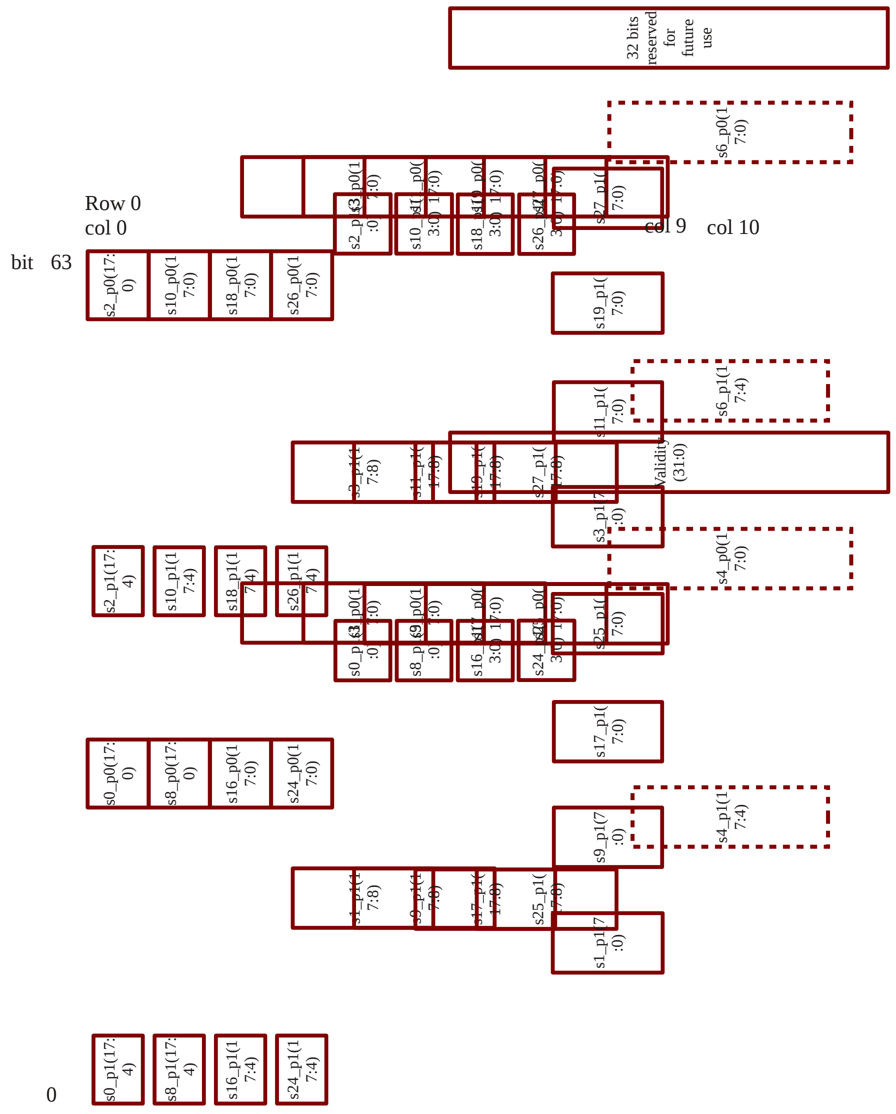


Figure 2.19: Allocation of Data Samples to Columns

3. Allocation of complete FFT periods to banks and rows

A complete set of 1023 frequency bin points occupies 40,920 columns (40 rows). Data for the next FFT period should be written to the next bank in rotation so that for example the first 10 FFT periods are stored as shown below.

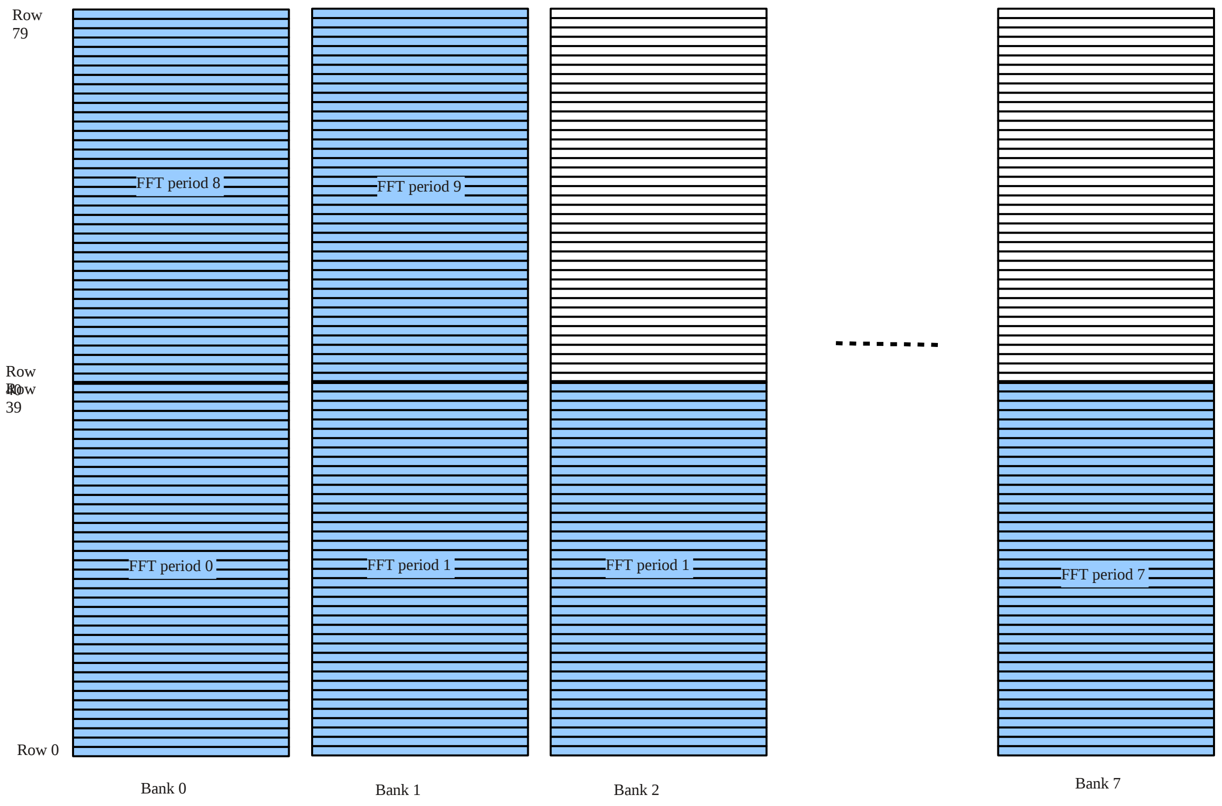
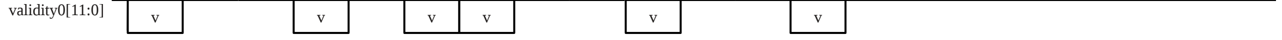


Figure 2.20: *Allocation of FFT Period Data to Banks*



Hardware Interface To Correlator Engine(s)

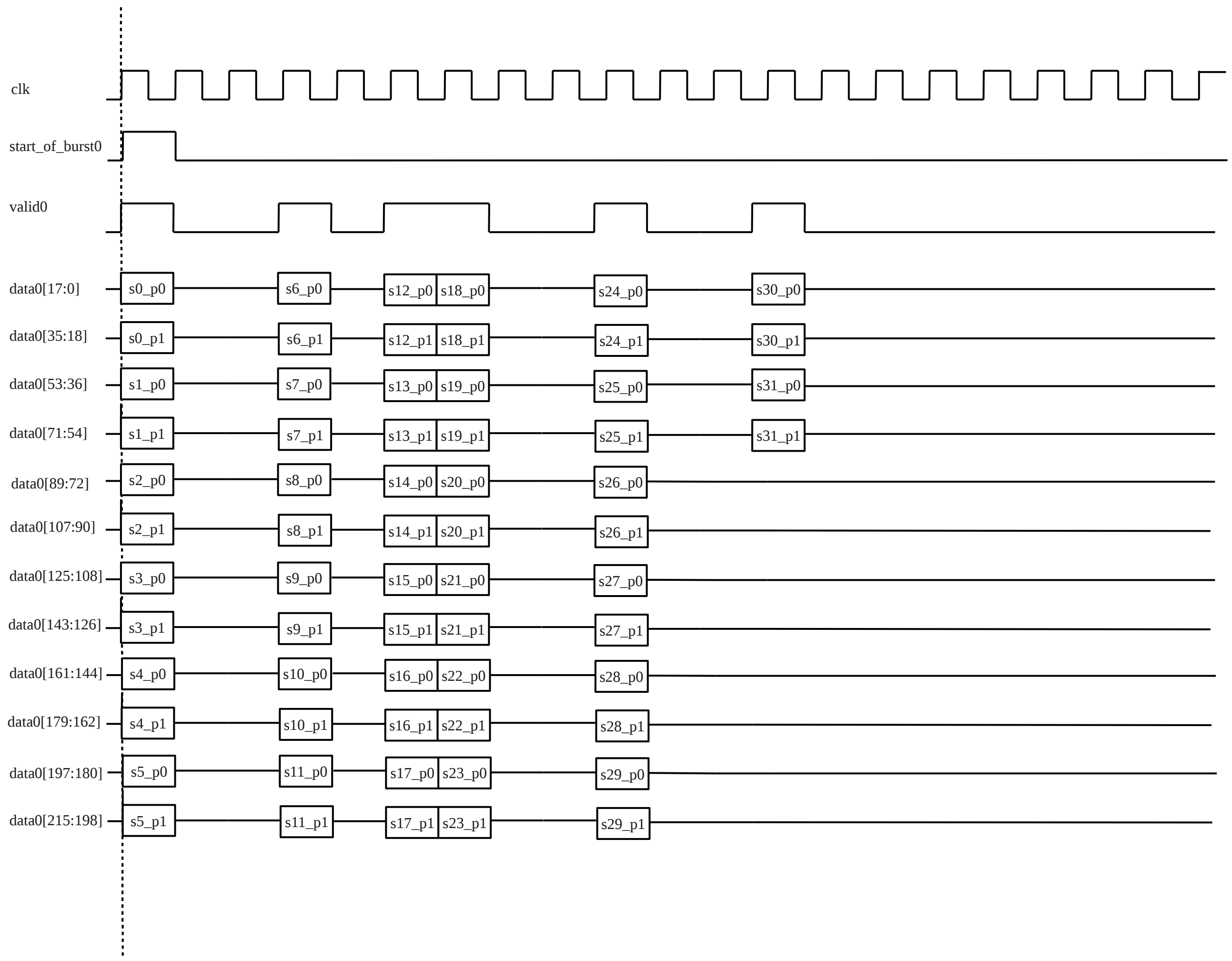
The data for each station is composed of two 9 bit complex values, one for each polarization. These are denoted sn_p0 and sn_p1 respectively for station number n. Within each complex value bits [8:0] contain the real part and bits [17:9] contain the imaginary part.

Signal name	*Direction*	*Width*	*Purpose*	*Source*
clk	out	1	266.5MHz clock	
end_of_freqbin				
data0	out	216	Data to correlator engine 0	
validity0	out	12	Validity bits associated with data0	
start_of_burst0	out	1	High coincident with first data0 word	
valid0	out	1	Data streams valid when high	
data1	out	216	Data to correlator engine 1	
validity1	out	12	Validity bits associated with data1	
start_of_burst1	out	1	High coincident with first data1 word	
valid1	out	1	Data streams valid when high	

The validity signal contains the validity bits for the station data in the coincident data word. For example on the first word of the cycle Validity0[11:0] is composed of

Validity0[0]	validity bit for s0_p0
Validity0[1]	validity bit for s0_p1
Validity0[2]	validity bit for s1_p0
Validity0[3]	validity bit for s1_p1
Validity0[4]	validity bit for s2_p0
Validity0[5]	validity bit for s2_p1
Validity0[6]	validity bit for s3_p0
Validity0[7]	validity bit for s3_p1
Validity0[8]	validity bit for s4_p0
Validity0[9]	validity bit for s4_p1
Validity0[10]	validity bit for s5_p0
Validity0[11]	validity bit for s5_p1

All signals are synchronous with the 266.5MHz system clock. Ports 0 and 1 drive two separate correlator engines. A timing diagram for port0 is shown below. All the data for a given frequency bin are clocked out in a cycle of 6 words. There may be gaps between the words, indicated by valid0='0', but the cycle must be complete within 16 clocks.



Correlation Engine

Each FPGA contains a total of 1288 18x18 bit multipliers: four in each of 322 half-DSP blocks. A half DSP block can be configured as a single complex multiplier such that 18 bit complex inputs

a + jb and c + jd

are combined to give 32 bit outputs

ac – bd (real)
ad + bc (imaginary)

The complex multipliers can be clocked at over 400MHz with the resulting products summed in accumulators formed out of array logic or other DSP blocks.

For a 32 station correlator there are 496 cross correlation products and 32 autocorrelation products per polarization. For two polarizations, including cross polarization products, the total number of correlation products is 2112. Each correlation product requires a complex multiply, and must be done at a rate equal to the input bandwidth (64MHz), giving a total multiply-accumulate rate of

2112 x 64/1000 = 135GMAC

In practice this processing load is spread evenly over the four BN chips, so each chip requires

135/4 = 34GMAC

This processing rate can be attained using 85 out of 322 half-DSP blocks clocked at 400MHz, however resources for accumulation must also be considered – they may be constructed either using additional DSP block resources, or generic FPGA logic.

Implementation

This section describes a possible implementation of the multiply-accumulate engine for 2112 correlation points. An underlying assumption is that only 20 significant bits from the complex multiplier output are accumulated to 36 bits, allowing up to 2^{16} accumulations. This assumption is valid if the inputs to the complex multipliers are rescaled to a 9 bit signed representation.

Timing analysis shows that such a multiply-accumulate cell can be clocked at over 300MHz and occupies a half DSP block for the complex multiply, and 72 logic cells for the real and imaginary accumulators. It can be seen that all the correlation points can be computed by 132 such cells in 16 passes, provided they are clocked at over 256MHz. The correlation points can be processed in four equal (528 point) stages defined by the polarization: left x left, right x right, left x right and right x left. Within each stage the points are calculated in four passes of the MAC array. In Figure 2.8 each dot represents a MAC cell, and the numbers to the right and below represent the station data fed to that row or column at each pass. During the first stage both the row and column data come from the left polarization, then the right, then for the cross polarizations the rows are left polarization while the columns are right and vice-versa.

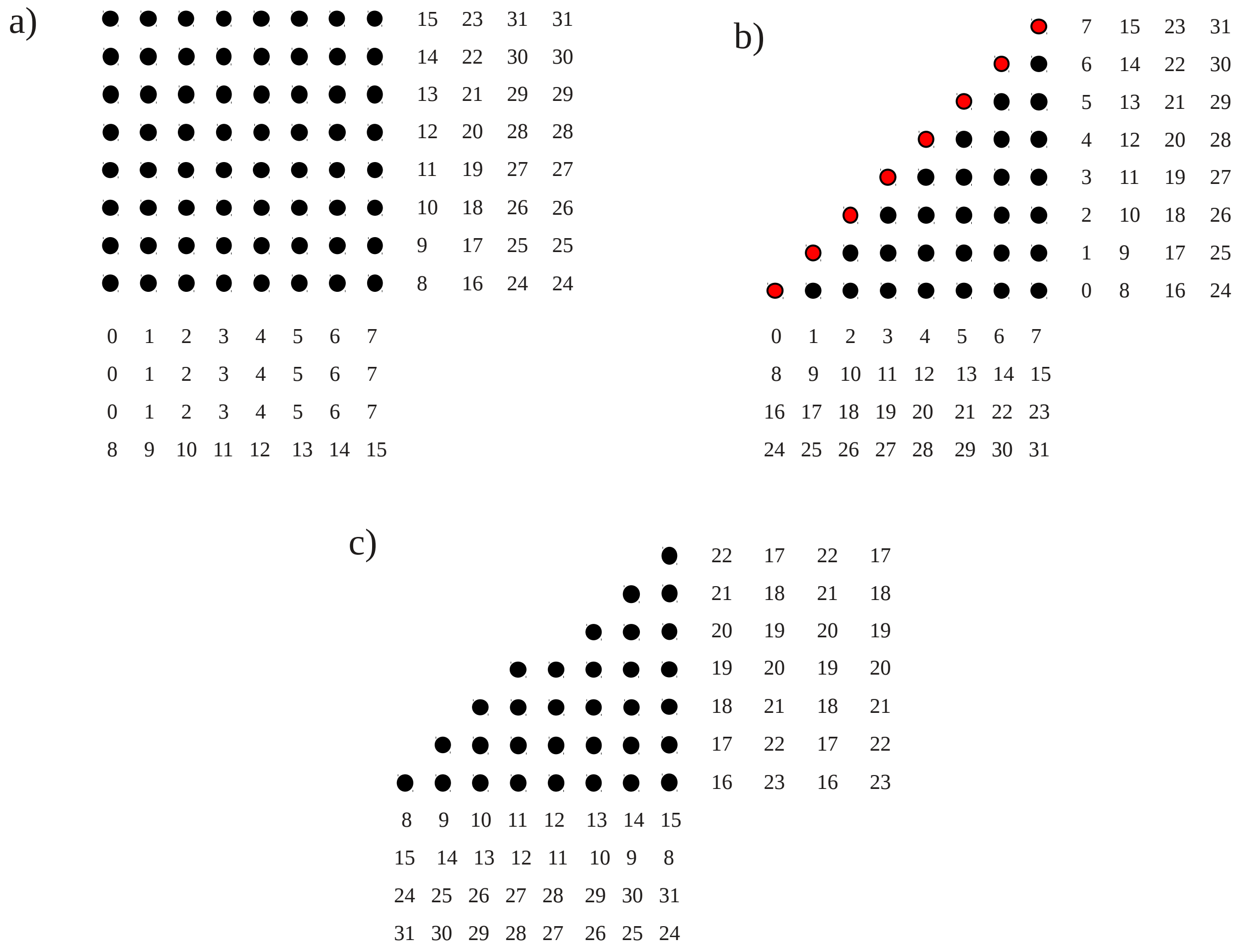


Figure 2.22: *Correlation Points Processed by 132 MAC Cells*

The stations are grouped into quadrants (0-7, 8-15, 16-23 and 24-31) and distributed to the MAC cells as follows

a) A square array of 64 MAC cells processes the bulk of the cross polarizations. The row data are always from a different quadrant of stations than the column data.
b) A triangular array of 36 MAC cells processes all the ‘same quadrant’ points including the auto-correlations which are shown in red. Note that during the last stage the ‘auto’ cross-polarisation points for LxR are the same as for RxL and need not be saved
c) A triangular array processes the cross-quadrant points which do not fit in (a). Four cells are omitted to avoid repeating those points so this array has 32 MAC cells.

The DSP resources needed for this implementation are 132 half DSP blocks plus 9504 logic cells for the accumulators.

Figure B.2 in Appendix B below shows the main signal flows through the BN chip using the corner-turning architecture. Deserialised data from the FN chips enter one of four FIFOs which ensure the data based on sync pulses transmitted at the start of each set of frequency points. Data from the FIFO are passed directly to the DDR interface which is currently active for WRITE, so the FIFO must be large enough to prevent overflow during typical DDR precharge and activate latencies. A total FIFO size of 16 kbits should be sufficient.

While one DDR is active WRITE, data from the other are read out as described in the previous section and distributed to the MAC cells. A thin, 16 bit deep by 64 bit wide, FIFO passes the data from the DDR3 clock domain to the DSP clock domain. The data are then held in a set of 2304 registers while the ‘sequencer’ stores them in appropriate locations in the MAC input data buffers. There are six of these buffers, corresponding to the rows and columns of the three MAC cell groups in Figure 2.22. Each buffer is 16 deep by 8 complex points wide: 4608 bits for 18 bit precision. Two complete sets of registers and input buffers are required to ensure a continuous flow of data to the MAC cells. So the pre-accumulation storage requirements are

FIFO between clock domains:	1024 bits
hold data during distribution	2 x 2304 = 4608 registers
MAC cell input buffers	2 x 6 x 4608 = 55,296 bits

Storage is also required for the intermediate accumulation products. For the scheme shown in Figure 2.22 each of the 132 MAC cells needs storage for sixteen 36 bit complex results, totaling:

132 (MAC cells) x 16 (passes) x 36 (bits) x 2 (cplx) = 152,064 bits.

Again, double buffering is required to ensure a continuous data flow to the output ports, so the total allocated is 304,128 bits.

After complete integration of a frequency point, the data pass through a 16k bit FIFO to the output port clock domain. In general the products are transmitted to the backend computer via one or more of the four BN 10GbE ports, but the 1GbE port can be used for long integration times or small number of points. The data rate on the 1GbE port should not exceed 800Mbps per BN to avoid conflict with the control data

Validity in the BN

Each frequency point from every station/polarization is accompanied by a data valid bit, thus 64 validity bits are stored in the corner turning memory along with each frequency point. The valid bits enable or disable correlation of individual channels for a given frequency point. When invalid, the MAC cell output is set to 0. When valid the MAC cell performs the multiply-accumulate and a validity counter is incremented. At the end of an integration the validity counter is transmitted to the backend processor along with the accumulated correlation products.

There is a validity counter for each of the 2112 correlation points; Each counter needs 16 bits to accommodate 1s integrations on 64MHz, 4096 frequency point, data. The counters can be implemented using 132 counters (2112 logic cell registers) overclocked by a factor of 16, and 34,000 bits of storage for the intermediate results.

Correlator Product Output

Frequency ordered Architecture

Because of the corner turning architecture, the correlation engines process the frequency bins sequentially within an integration period. After each frequency bin integration all single-, cross- and auto-correlation products are finished and must be dispatched to the backend computer while the next frequency bin is processed.

Each BN contains one correlation engine which can process 1024 frequency bins from different sub-bands. One frequency bin integration generates 528 complex products per single polarization (including auto-correlations). For two polarization data there are 2112 products, neglecting repeats.

For integration times < 0.5s the products fit into two 32 bit words. For longer integrations the data are padded to 64 bit words before transmission to the backend computer.

Validity bits are accumulated in parallel with the data and sent in the same packet as the data.

Output Format

The correlation products are sent to the backend computer in UDP packets. The packet has a four 32 bit word header which identifies the data by its frequency bin, correlator engine, FPGA and Uniboard number. Two 32 bit word fields contain the time stamp of the first sample in the integration period in the form of the number of whole seconds since some epoch and its sample number within the current second.

A packet may contain any number of correlation products plus their validity counts, provided the MTU of 9000 bytes is not exceeded. A flag in the header denotes whether the data fields are 32 bit or 64 bit. If there are an odd number of products in a packet, one additional padding product with zero real, imaginary and validity fields, is appended at the end.

The following table shows the first 7 words for the 32 bit case. For 64 bit data the real part of the first product would occupy words 4 and 5, the imaginary part words 6 and 7, and the validity count word 8.

Word 0	VVV$_3$	F$_1$	Chip ID$_8$	CE$_2$	reserved$_6$	Frequency bin #$_{12}$
Word 1	reserved$_8$		Payload size$_{12}$			First product #$_{12}$
Word 2	Integer no of seconds since epoch to start of integration period$_{32}$					
Word 3	No of samples since last second to start of integration period$_{32}$					
Word 4	Data for first product (read)$_{32}$					
Word 5	Data for first product (imaginary)$_{32}$					
Word 6	Data for first product (validity)$_{32}$					

First 7 words of an output packet for 32 bit correlation products. Msb at left.

Word 0		
Bits 0-11	12	Frequency bin number (0-1023)
Bits 12-17	6	Reserved
Bits 18-19	2	Correlator engine number within an FPGA
Bits 20-27	8	Chip (node) ID. Bits 20-22 are the FPGA; bits 23-27 are the board number.
Bit 28	1	Flag to indicate 32/64 bit data representation (0/1)
Bits 29-31	3	Header version code (0-7) default 0
Word 1		
Bits 0-11	12	Number of the first product in this packet (0-2111)
Bits 12-23	12	Payload size. The number of products in this packet (not including padding when payload size is odd).
Bits 24-31	8	Reserved
Word 2		
Bits 0-31	32	Integer no. of seconds at start of integration period
Word 3		
Bits 0-31	32	Sample no. within second at start of integration period

Function of the header fields in the output packet

Field Testing

Test pattern generators are placed in the FN before the PFB and in the BN before the corner turner and again before the correlation engine to allow testing of those structures. Test results can be examined using the signal statistics (state counters and power meters) or by capturing a snapshot of data in SRAM. The resources need to generate the test patterns are quite small, but SRAM in the FN is very limited so the capture feature may need to be reduced or omitted in the final design. The packet receiver logic can be tested using VDIF packets generated by computer software or another UniBoard.

Appendix A Estimates of FPGA Resources

Front Node

Stage	**Multipliers (18x18 equivalent)**	**ALUTs**	**SRAM(kbits)**	**Registers**
Packet Receiver	0	28000	1656	20000
Nios Controller	4	1200	560	940
1GbE Control port	0	TBD	8	TBD
Delay Module	0	4528	612	6277
Complex Mixer (x8)	16	0	0	0
Filter Bank (x8)	224	26936	7596	42808
Interface to BN	0	2649	4	4382
Total/Available	**244/1288**	**63313/182400**	**10436/14200**	**74407/182400**

Entries marked TBD are unknown but not expected to be limiting

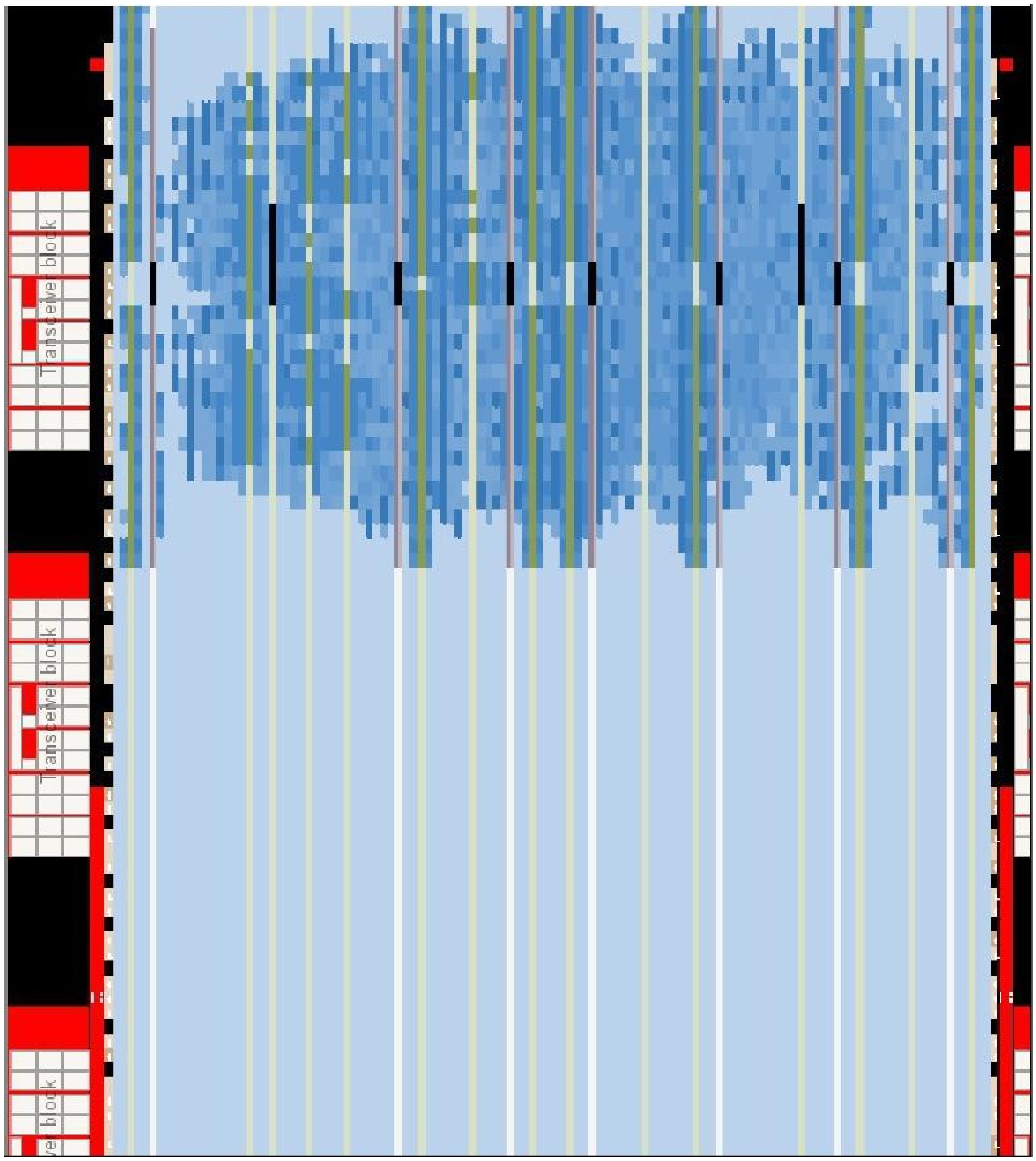
Back Node

Figures are for two correlator engines per chip.

Stage	**Multipliers (18x18 equivalent)**	**ALUTs**	**SRAM(kbits)**	**Registers**
Interface from FN	0	3000	8	5000
Nios Controller	4	1200	560	940
1GbE Control port	0	TBD	8	TBD
Corner Turner	0	35000	477	18800
Correlator Engine (x2)	1056	31936	574	25244
10GbE Product out ports[1]	0	13000	30	10226
Total/Available	**1060/1288**	**84138/182400**	**1657/14200**	**60210/182400**

Notes
[1] Two 10Gb Ethernet ports per FPGA
Entries marked TBD are unknown but not expected to be limiting

The image below shows part of the BN FPGA containing a single correlator engine and validity accumulator. The dark blue areas represent high logic utilization while light blue areas are still empty. Overall this part of the design is constrained to about 40% of the chip, determined by the distrubution of DSP blocks (red) and memory blocks (green). Only about 45% of the general logic and routing resources within that area are occupied however.



Appendix B Signal Flow Diagrams

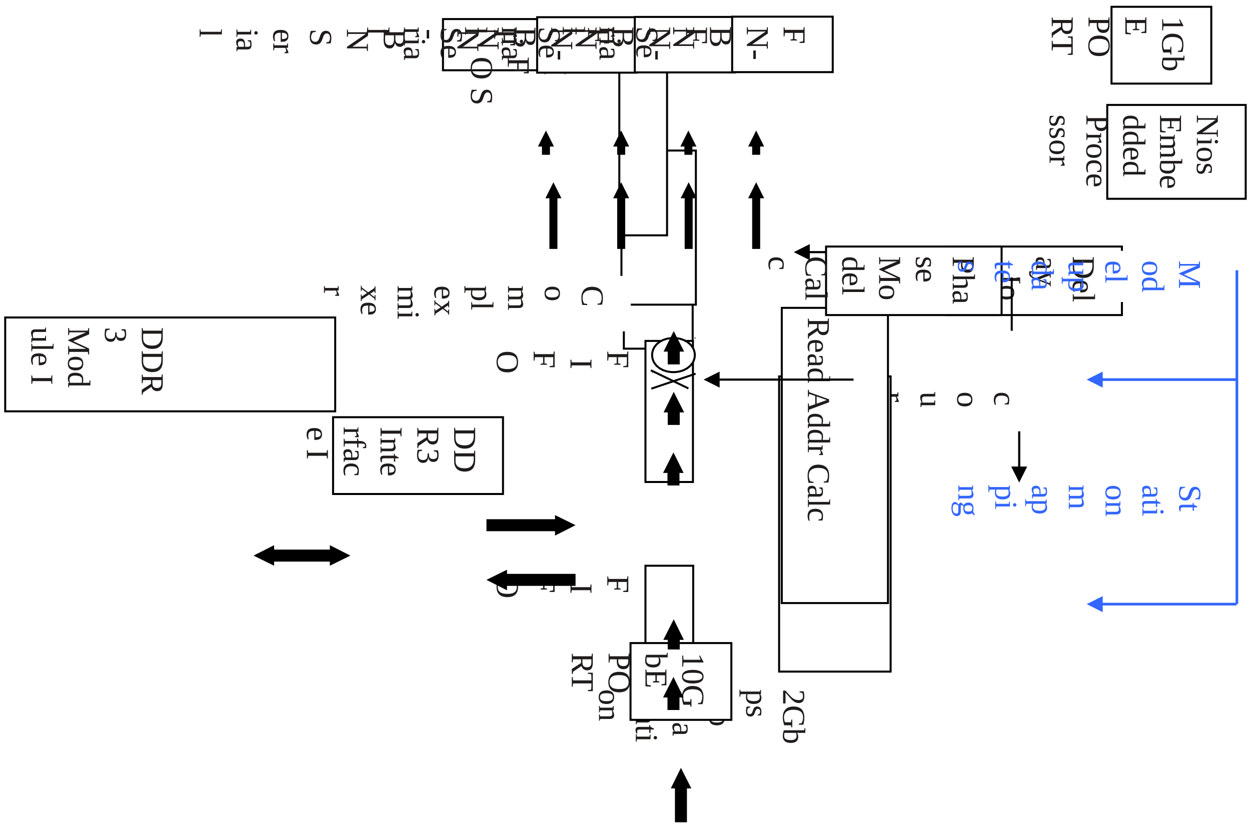


Figure B.1: *Signal Flow through the FN FPGA*

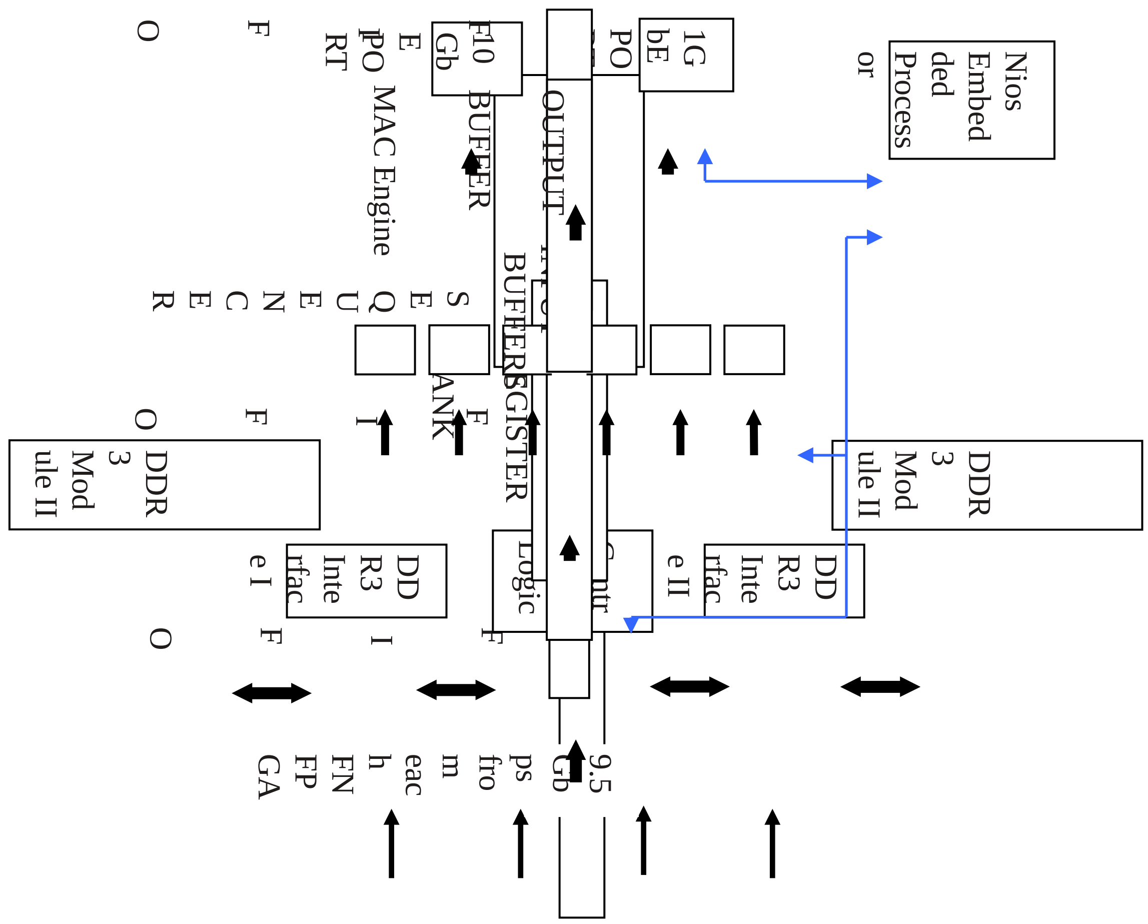


Figure B.2: *Main Signal Flows Through the BN FPGA*

References

[1] “VLBI Data Interchange Format (VDIF) Specification”, Release 1.0, Ratified 26 June 2009, Madrid, Spain. Available on the UniBoard Wiki

[2] “Requirements for a VLBI Correlator – Draft”, Paul Boven, March 13, 2009. Available on Unoboard Wiki.

[3] “A 2GHZ Digital Filterbank Correlator”, Ferris, R.H., Bunton, J.D., Stuber, B., The SKA: Defining the Future, Berkeley California, July 9-12, 2001

[4] “A New Approach to Pipeline FFT Processor”, Shousheng He and Mats Torkelson ISSN 1063-7133/96