REQUIREMENTS AND FUNCTIONAL SPECIFICATION

Station Board Filter FPGA

RFS Document: **A25044N0000**

Revision: 1.10

Dave Fort, 01 April 2011

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

P.O. Box 248, 717 White Lake Rd
Penticton, BC, Canada
V2A 6J9

Table of Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

BBC	Base Band Converter (VLBA)
CMIB	Correlator Module Interface Board
DLL	Delay Locked Loop
ESD	Electro-Static Discharge
EVLA	Expanded Very Large Array
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
IOB	Input Output Buffer
ISR	Interrupt Service Routine
JEDEC	Joint Electron Device Engineering Council
JTAG	Joint Test Action Group (Boundary Scan Architecture)
MHz	Megahertz (10^6 cycles per second)
MTBF	Mean Time Before Failure
Ms/s	Mega Samples per Second
Mw/s	Mega Words per Second
PAR	Place and Route
PCB	Printed Circuit Board
PLL	Phase Locked Loop
RFI	Radio Frequency Interference
VLBA	Very Long Baseline Array
VLBI	Very Long Baseline Interferometry
WIDAR	Wideband Interferometric Digital ARchitecture

1 Revision History

Revision	Date	Changes/Notes	Author
0.0	01 Apr 2003	Initial Draft	D. Fort
1.0	01 Jan 2005	Peer Review and Virtex 4 Changes	D. Fort
1.1	05 Aug 2005	Removed stuff in IO and D2 for PAR	D. Fort
1.2	15 Jan 2006	Changes from 1.1 are highlighted	D. Fort
1.3	15 Jul 2006	Changes from 1.2 are highlighted	D. Fort
1.4	25 Jan 2007	Input delays replaced by DDR. Removed Daisy Chain Delay. Changes from 1.3 are highlighted	D. Fort
1.5	25 Apr 2007	More test abilities. Removed PROTECT.	D. Fort
1.6	25 Aug 2007	Removed MCB burst mode. Status bit additions. Added system tick delay. Sideband flipper synchronization changes.	D. Fort
1.7	15 May 2009	Corrected documentation in INOUT Registers. Add delay error read register. Add registers for removing sampler DC offset.	D. Fort
1.8	19 May 2009	Corrected documentation. Brought Appendix up to date.	D. Fort
1.9	30 Mar 2010	Corrected stage clipping algorithm. Add new test feature. Add DCM reset for no Raltron case. Added toggle status bit for SCLK. Added bit to change sign of PHASERR. Added DCM phase shifting. Added IO_SID register to allow staggering of clock re-acquisition. Added STICK edge measurement. Improve noise generator.	D. Fort
1.10	01 Apr 2011	Add pinouts etc..	D. Fort

2 Introduction

This document describes detailed requirements and design concepts for the Filter FPGA. The Filter FPGA is a rather complicated and important element in the WIDAR correlator design. It must cope with a large range of input data organizations and provide a large range of output bandwidths as well as perform a number of other functions.

Background on the Filter FPGA can be found in NRC-EVLA Memo #014 (EVLA Memo 31), "Refined EVLA WIDAR Correlator Architecture" and Memo #003 (EVLA Memo 13), "A Closer Look at 2-stage Digital Filtering in the Proposed WIDAR Correlator for the EVLA". The Filter FPGA described in this document is very similar to those described in the above memos except that the narrower bandwidths, including radar mode are included in the Filter FPGA by increasing the number of decimation stages from two to four. This added capability is possible because the input sample rates to stages three and four are sufficiently low that the increase in hardware necessary to do the job is reasonably small.

The current development plan for the Filter FPGA is as follows:

> Develop and test a working design in a Xilinx XC4VSX35 FPGA for the prototype and production Station Board. The design will be done in Verilog HDL.

3 Overview

The Station Board receives two wide bands and selects 18 narrow bands from each of the wide bands and sends them to the baseline part of the correlator. A simplified block diagram of the Station Board is shown in Figure 3-1. Wide band optical inputs are recovered by the Fiber Optic Receiver Module, re-encoded by the Input FPGA, delayed by the Delay Modules, passed through the WBC FPGA and filtered by the Filter Bank. The narrow band outputs of the Filter Bank are sent to the rest of the correlator via the Output FPGAs. The Input FPGA can also select the wide band input from a VLBI playback system via the VSI FPGAs. The Filter Bank also sends the narrow band signals FPGAs to a VLBI recording system via the VSI FPGA.

The primary function of the Filter FPGA is to produce a narrow band from a wide band. Inside the Filter Bank block the Filter FPGAs are arranged in a “square daisy chain” as shown in Figures 3-2 and 3-3. The wide band signals pass through the FPGA and, after re-clocking, are sent on to the next Filter FPGA in the chain. This arrangement simplifies routing and improves signal integrity by avoiding multi-drop transmission lines at the expense of somewhat increased power dissipation. Another consequence of the daisy chain approach is that the wide band signals are delayed at least two clocks as they pass through each of the 18 Filter FPGAs in the chain. The first Filter FPGA in the chain receives its inputs from the WBC FPGA. The wide band outputs of the last Filter FPGA in the chain are not connected to anything. In addition, there are three groups of narrow band signals produced by the Filter FPGA. Two of these are the filtered data which are sent to the Output FPGA and the VSI FPGA. The third group is the corresponding phase error originating from the fractional sample delay error caused by the integer nature of the delay corrections applied in the Delay Module and in the Filter FPGA and is sent to the Timing FPGA. The Output FPGAs and VSI FPGAs each receive data from only one Filter Bank whereas the Timing FPGA receives phase errors from both Filter Banks. A slightly simplified input/ output diagram of the Filter FPGA is shown is Figure 3-4 (the JTAG interface, test port and FPGA configuration, interfaces are not shown).

Note that each Filter FPGA has two wideband input ports and two wideband output ports. Internally, one of the input ports is chosen and zero, one or two of the output ports can be enabled. The purpose of these extra ports is to simplify PCB routing and allows for some (hopefully unnecessary) fault tolerance. Each wideband input IOB contains a programmable tapped delay line to facilitate proper input clocking. In addition, each FPGA contains a programmable delay line to compensate the internal wideband signals for having passed through previous Filter FPGAs. The dashed boxes are the VSI and Output FPGAs for the corresponding filter bank. They are in different places relative to the Filter FPGAs in order to shorten the high speed connections to the connector.

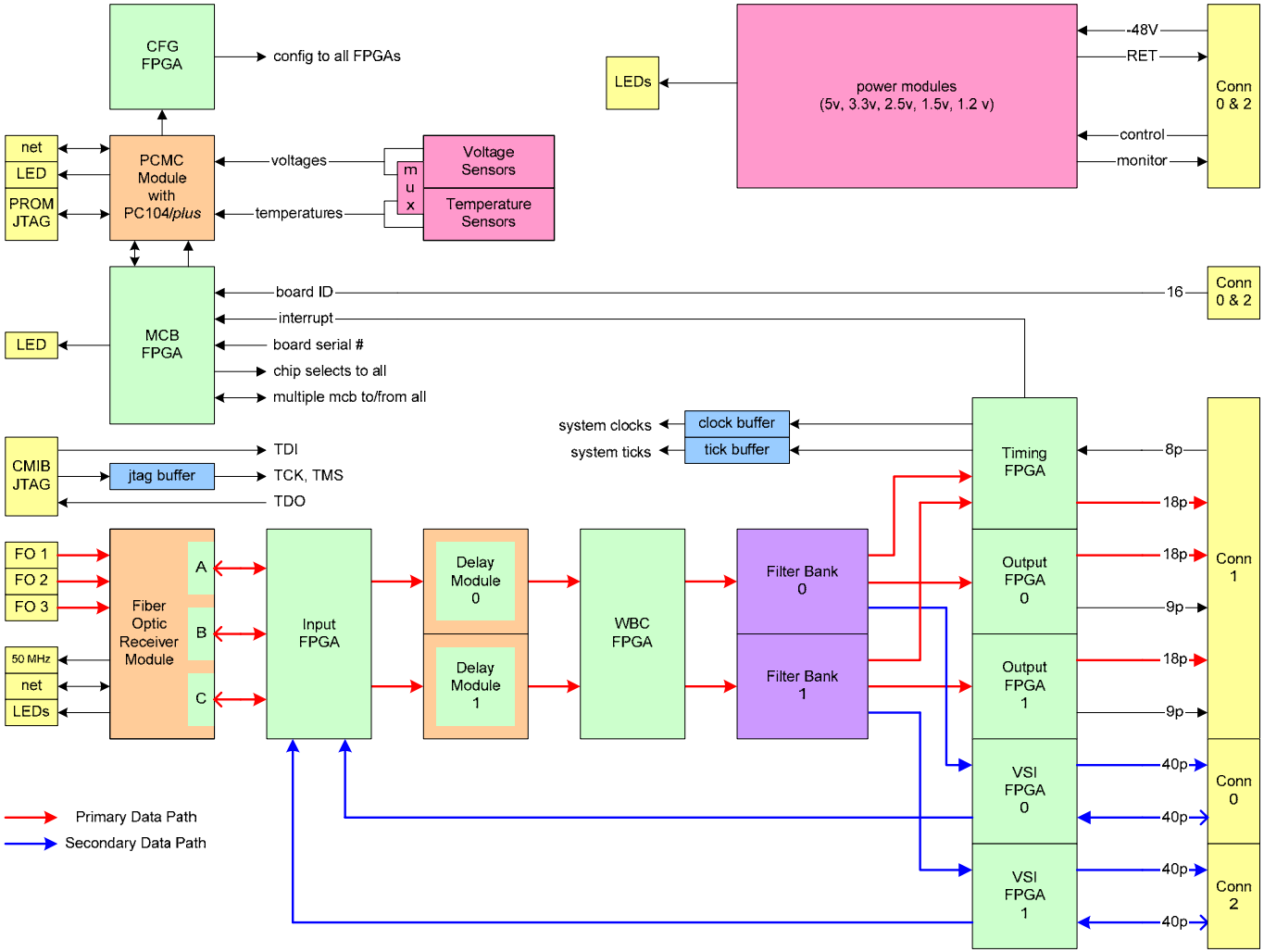

Figure 3-1 Station Board block diagram.

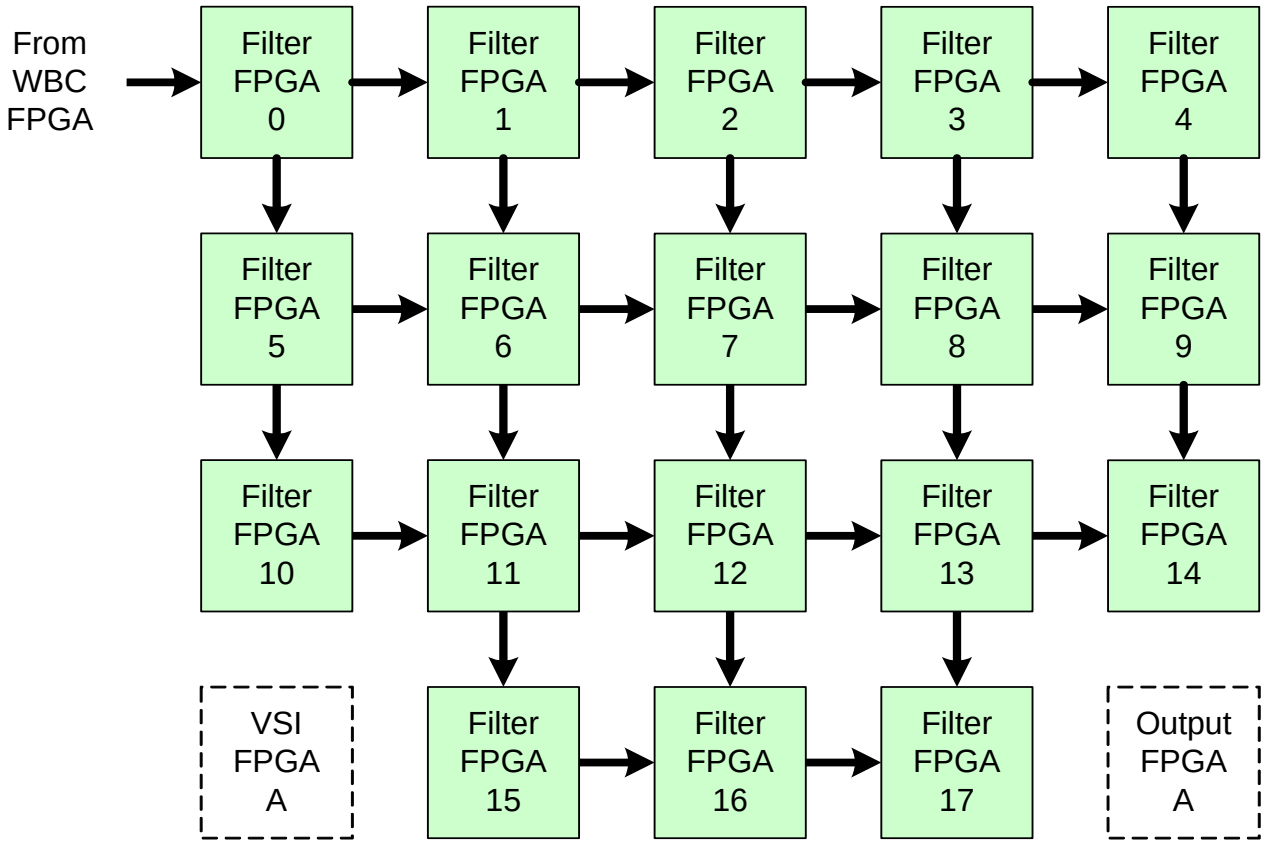

Figure 3-2 Filter Bank A Physical Arrangement.

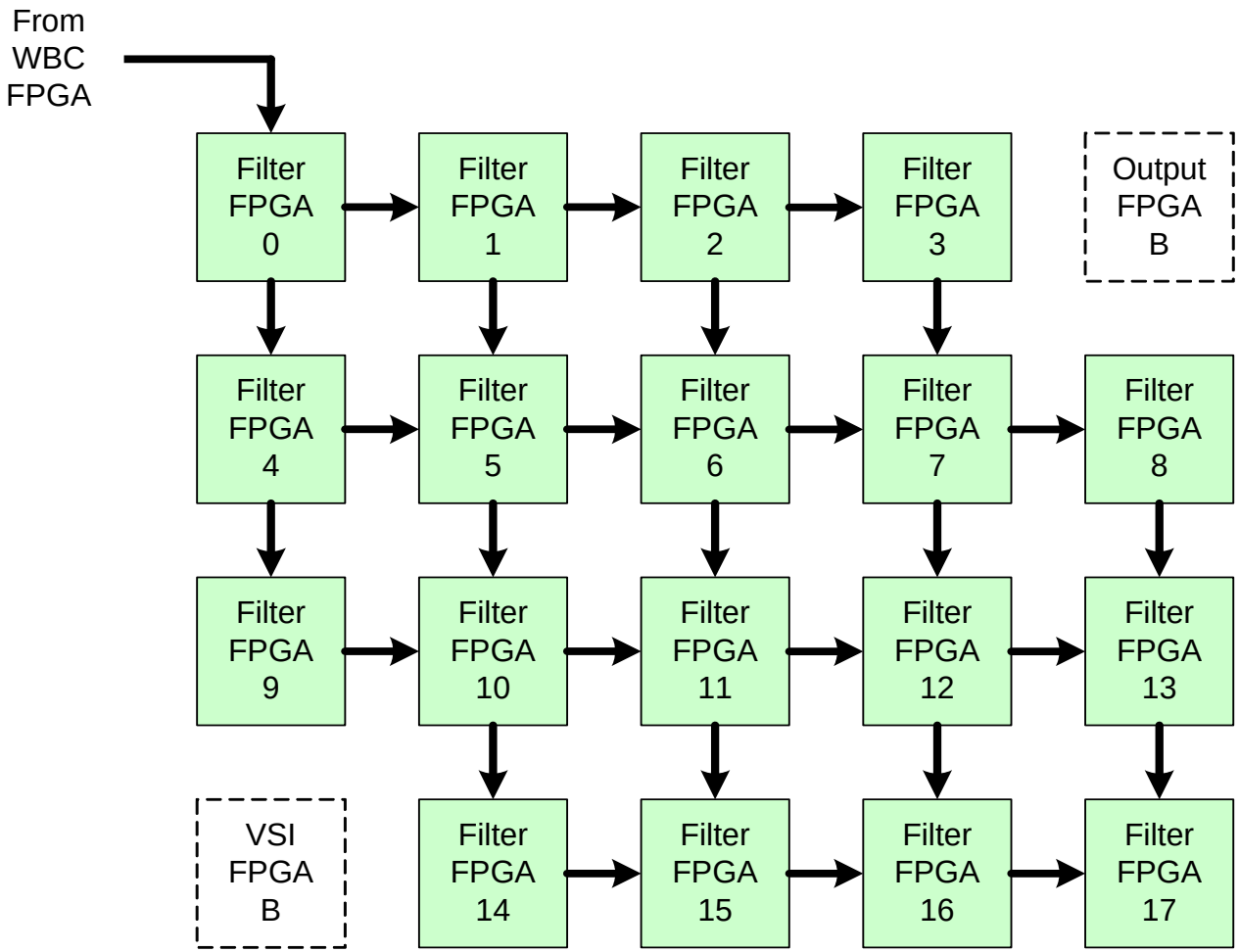

Figure 3-3 Filter Bank B Physical Arrangement.

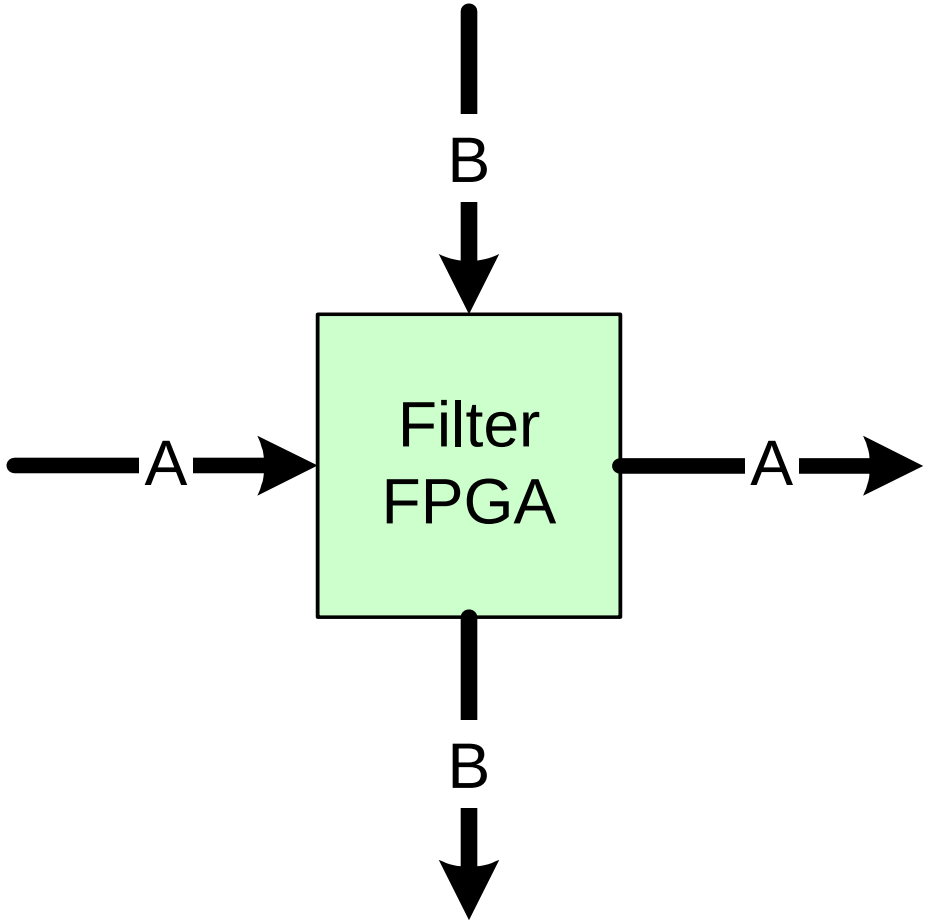

Figure 3-3A Filter FPGA Input Port Definition.

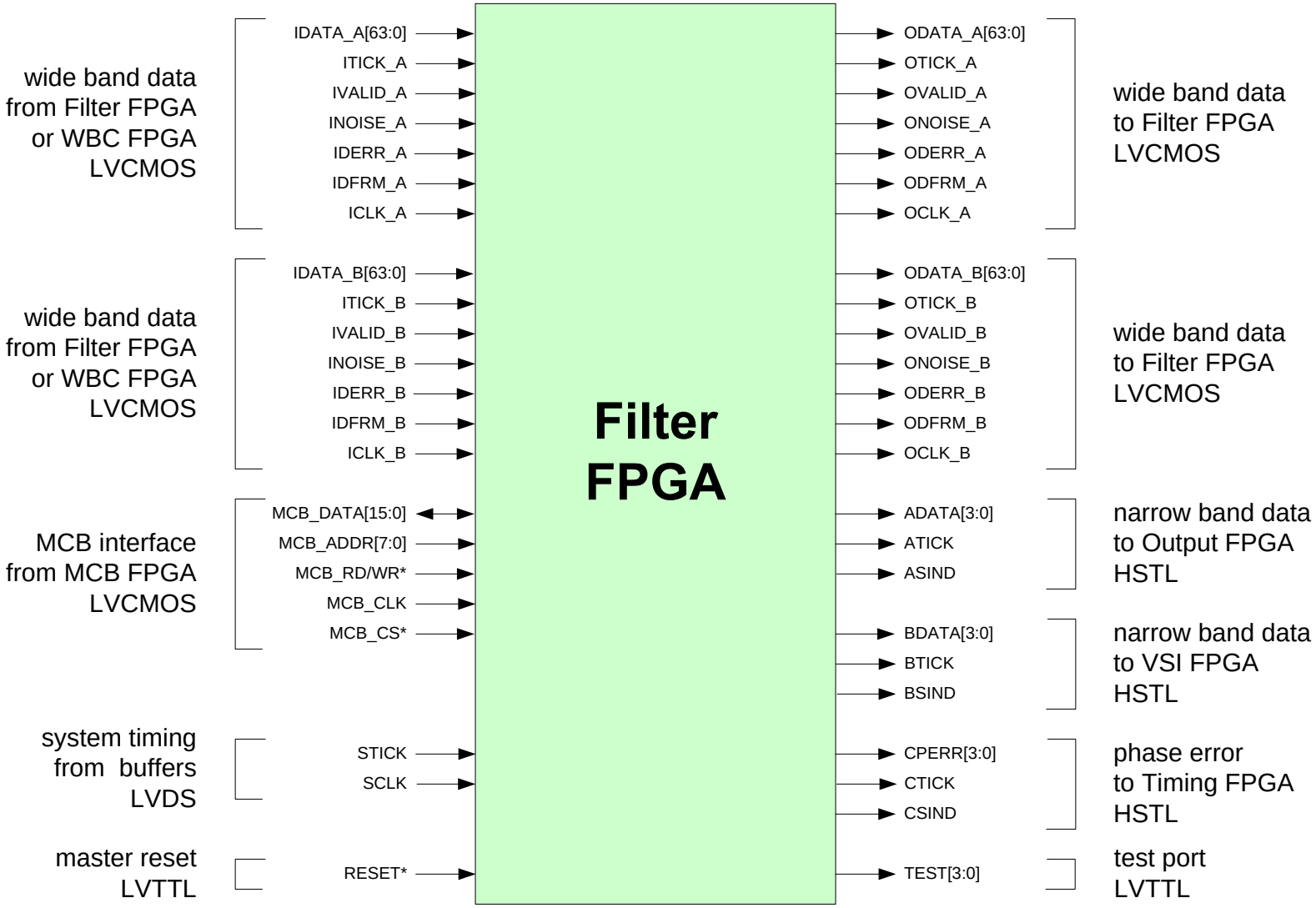

Figure 3-4 Input/Output diagram of the Filter FPGA.

The wide band data input consists of 64 data lines as well as valid, noise diode on/off, delay error, delay error frame, tick and clock lines. The data, valid, noise diode and tick lines are running at 256 Mbits/second. The delay error and delay error frame lines are running at 128 Mbits/second and the clock is 128 MHz. In addition, there are system tick and system clock differential lines coming to each FPGA.

The Filter FPGA outputs the same 70 signals after re-timing using the system inputs which preserves the overall system timing and prevents the accumulation of clock jitter. The scheme of sending clock and timing with the data simplifies the routing of the station printed circuit board (PCB) since the length of the 70 signal paths should all be the same relative to each other but the overall length of the signal bundle is not important.

For the EVLA, the 64 data lines originate from a single base band sampler. The samplers are either 3-bit for a 2048 MHz base band or 8-bits for a 1024 MHz base band. The Filter FPGA assumes that the input samples are 4-bits or 8-bits (the 3-bit samples are converted to 4-bit samples in the Station Board Input FPGA). For 4-bit operation, 16 consecutive samples appear as a 64-bit "word" and for 8-bit operation there are 8 consecutive samples per word, thus the sample rate is reduced from 4096 Ms/s and 2048 Ms/s to 256 Mw/s ("words") by using a de-multiplexing factor of 16 and 8 respectively. Other organizations of the 64 input data lines are possible and some will be used. They include cases where the 1024 or 2048 MHz base band is divided into narrower "bands" before the samplers. The current VLBA system, for example, can divide a wide base band into 16 narrow bands which are individually sampled (1 or 2 bits) before being recorded for later processing. e-MERLIN intends to combine the left and right circular polarizations, each with a bandwidth of 512 MHz and sampled at 8-bits. In general, the 64 incoming data lines can be arranged as follows.

Number of Bands	Width of Band (MHz)	Demux Factor	Bits per Sample	Mode ID
1	2048	16	4	4_16
2	1024	8	4	4_8
4	512	4	4	4_4
8	256	2	4	4_2
16	$128/2^n$ n=0,...12	1	4	4_1
1	1024	8	8	8_8
2	512	4	8	8_4
4	256	2	8	8_2
8	$128/2^n$ n=0,...12	1	8	8_1

Table 3-1 Wide band input data organizations.

Modes 4_16 and 8_8 are the standard EVLA data organizations. Mode 8_4 is for the 8-bit e-MERLIN case. Mode 4-1 could be used by the current VLBA and represents cases where the bandwidth is 128 MHz or less. The others are included for completeness. The Station Board FPGAs should deal with all of the above organizations, if possible. In order that the Delay Module and the Filter FPGA be able to implement the above organizations, it is necessary that all the bits that are sampled at the same time (called "stream" in the design) are adjacent in the 64-bit data word. This allows the bits and bands to be treated as a single (wide) sample and be delayed together.

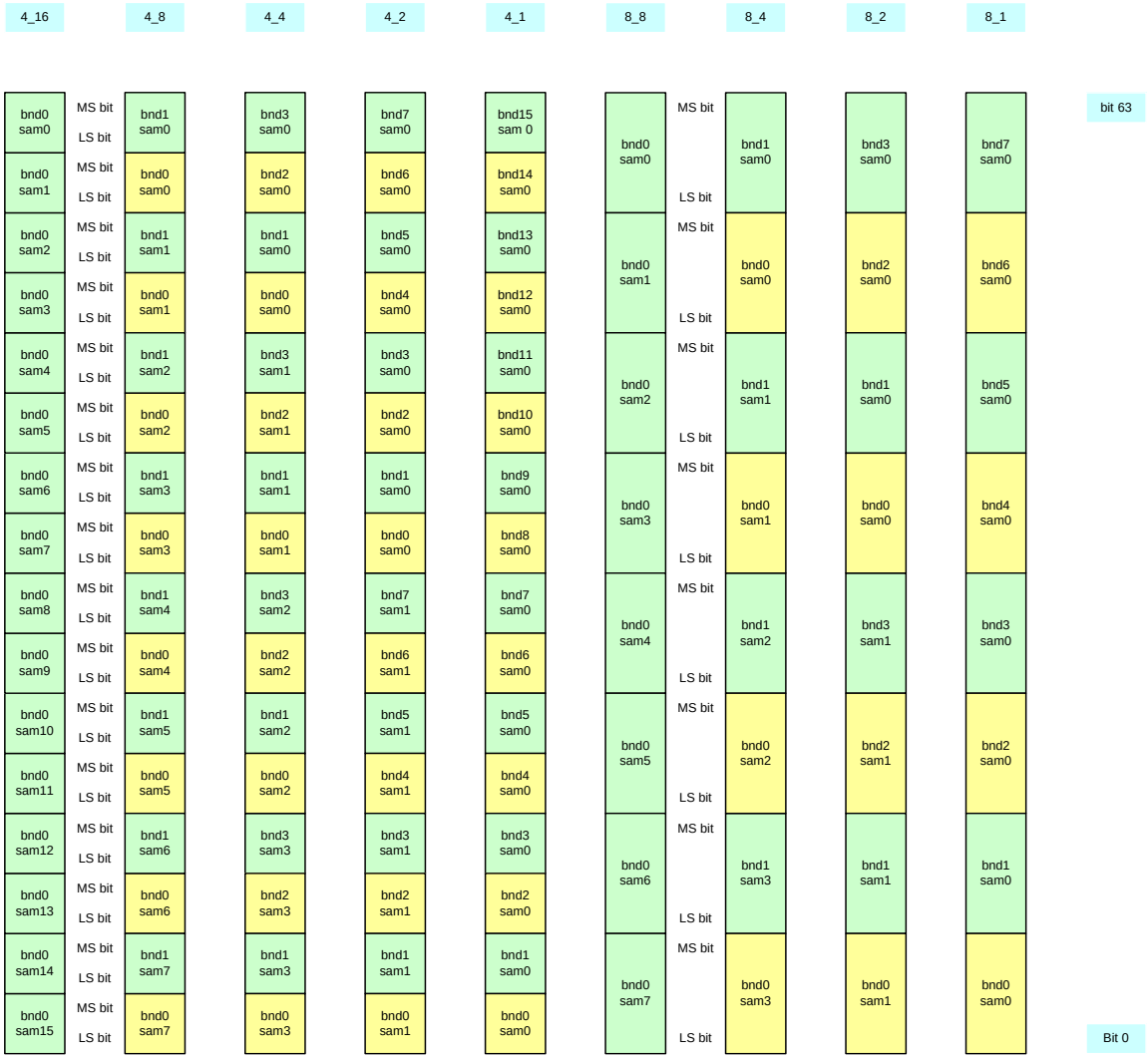

Figure 3-5 Wideband Input Data Organization

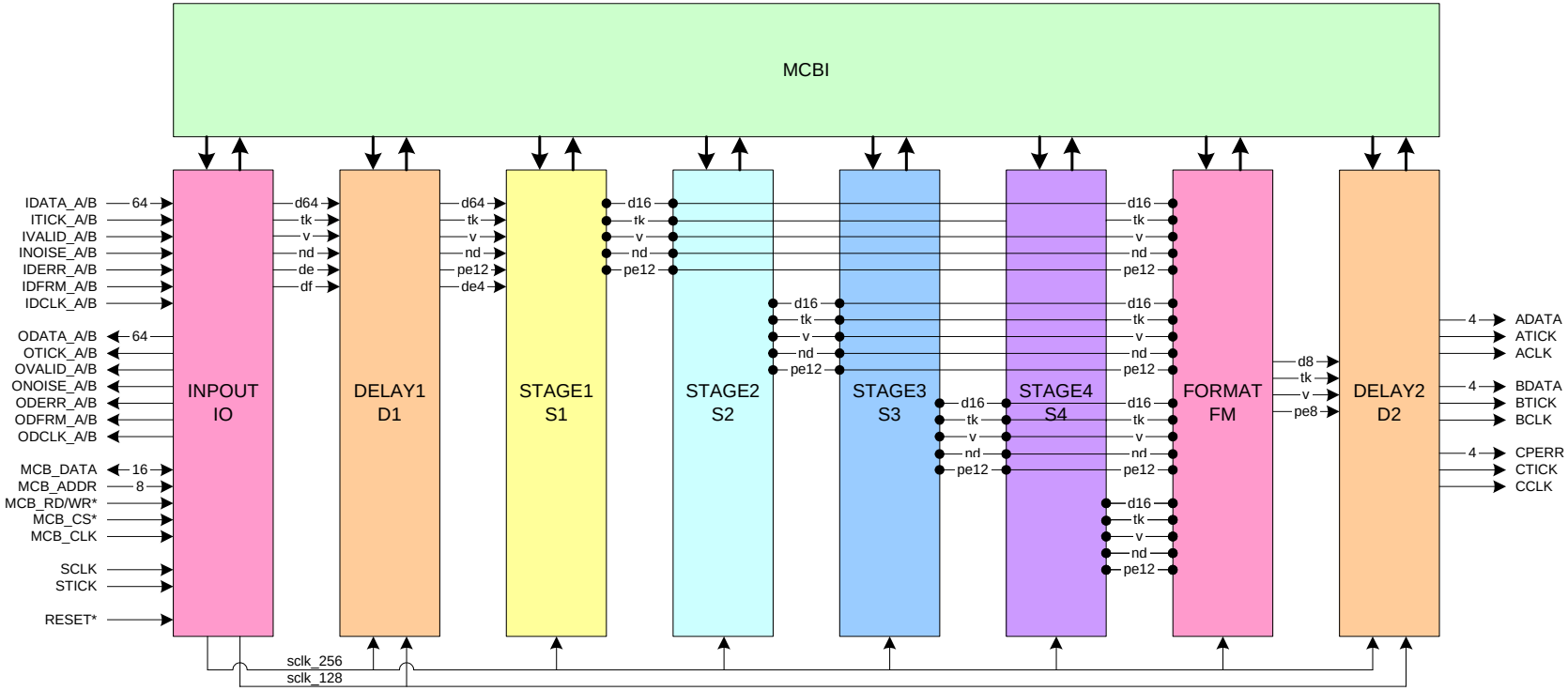

Figure 3-6 Block diagram of the Filter FPGA

The Filter FPGA is divided into nine blocks that are expanded in the section 5. MCBI is the Monitor and Control Bus Interface that contains the register set used to configure and control the other blocks. INOUT receives the wide band data from the previous FPGA and re-transmits it to the next FPGA as well as providing signals for use within the FPGA. DELAY1 implements a sub-band delay using a time-varying delay model. STAGE1 reduces the input bandwidth from 2048 MHz to 128 MHz. STAGE2 can reduce the bandwidth from 128 MHz to 64, 32, 16 or 8 MHz. STAGE3 can reduce the bandwidth from 8 MHz to 4, 2, 1 or 0.5 MHz. STAGE4 can reduce the bandwidth from 500 KHz to 250, 125, 62.5 or 31.25 KHz. FORMAT (historical name) re-quantizes the data to 4 bits or 8 bits (actually 7 bits due to Baseline Board limitations) and makes measurements that allow the output of the whole correlator to be calibrated. DELAY2 serializes the narrow band data and phase error to allow transmission to the Output, Timing and VSI FPGAs on six wires (4 data, 1 tick and 1 sample indicator). It also implements a delay of up to 4095 256 MHz clocks that allows different bandwidths produced in separate Filter FPGAs to line up in time to the nearest sample.

Each stage accounts for its own pipeline delay so that the signals coming out of each block are in the correct timing relationship relative to each other, regardless of the amount of pipeline delay present in that block. The data output of all blocks will be at 256 Ms/s regardless of the bandwidth of the filters involved. The actual sample rate is determined by a clock enable for the 256 MHz system clock. The output of the FPGA (DELAY2) for lower bandwidths requires a filler nibble with alternating ones and zeroes to replace the data as per A25022N0041 (for 31.25 KHz bandwidth and 4-bit samples there are 4095 filler nibbles for each actual data sample).

STAGE1, STAGE2, STAGE3 and STAGE4 consist of FIR filters with 512 taps for the divide-by-16 case. Note that the software should implement odd tap filters (by setting the last coefficient to zero) so that the pipeline delay through the filter will be an integral number of clocks (as opposed to an integral number of clocks plus half a clock). STAGE2 includes an up-front phase shifter that allows bands narrower than 128 MHZ to come from anywhere in the 128 MHz band. STAGE4 can produce 512 taps for all decimations.

4 Requirements

The following is a list of Filter FPGA requirements.

4.1 Functional Requirements

1. The Filter FPGA will input all data organizations given in Table 3-1 although only modes 4_16 and 8_8 are required for the EVLA. Mode 8_4 is required for e-MERLIN and Mode 4_1 is required for VLBI . The input bandwidth of the Filter FPGA is a binary sub-multiple of 2048 MHz and can be as narrow as 8 MHz. The output bandwidth is a binary sub-multiple of 128 MHz and can be as narrow as 31.25 KHz. The output data will come from any frequency slot in the input band whose nominal edge frequencies are integral multiples of the output bandwidth. The input and output data streams are real valued. A relaxation of this requirement will be possible for bandwidths less than 128 MHz by incorporating a mixer stage in STAGE2.

2. The width of the output data will be 4 bits for all output bandwidths or up to 8 bits for bandwidths from 64 MHz down to 31.25 KHz. The output data path to the Baseline Board is 4 bits wide; therefore, serialization is necessary for the 8-bit modes. Because of limitations in the Correlator FPGA on the Baseline Board, the Station Board must be able to deliver 4-bit and 7-bit samples.

3. A signal outside of the desired band and small regions adjacent to the desired band must be suppressed by at least 60 dB in the output band. The ability to suppress strong radio frequency interference (RFI) is required. It must be possible to force the frequency range of the desired band to be “flipped” so that it goes from low to high frequencies.

4. All quantities will be measured that are necessary to calibrate the system that uses the output of the Filter FPGA. The power in the signal before re-quantization must be measured and separate results for the off and on states of the noise diode must be kept. In addition, power measurements and state counts made after re-quantization are required. Another required measurement is tone amplitude and phase normally used for VLBI. All of these measurements require a count of the corresponding valid samples used. The maximum integration time for all of the above measurements is 10 milliseconds.

5. Each Filter FPGA is required to implement an independent delay to allow for moving the synthesized beam in a useful region around the centre of the primary antenna beam. It should be noted that the narrowest four output bandwidths, especially if they originate near the upper end of the wide band input, need special processing to constrain the phase excursion due to fractional delay errors that vary over the rather long sample intervals.

6. The Filter FPGA must be capable of fractional delay correction to ±1/32 sample for input bandwidths equal to or less than 128 MHz. This functionality will be used for VLBI and will be done in STAGE1 instead of the normal filtering.

7. The output of an EVLA antenna (a group of four Station Boards) consists of 18 sub-bands. Each sub-band consists of one Filter FPGA output from each of the 8 base band inputs as well as TIMECODE, DUMPTRIG, PHASEMOD and PHASERR (see RFS Document AN25022N0000). Since there is only one independent TIMECODE, it is desirable that all the output data is time aligned with TIMECODE, regardless of the bandwidth (group delay) of the filters. If this is not done (the current case), then DUMPTRIG and PHASEMOD must be modified accordingly and a time stamp offset must be associated with the output of the correlator for each base band by software. Since the headers (IDs and CRCs) inserted by the Output FPGA must be inserted at the same time as the TIMECODE, the Filter FPGA will implement a delay in the output stream that allows a sample of even the lowest bandwidth to coincide with the TIMECODE. This operation could be performed (18 times) in the Output FPGA but is more simply done with less logic in the Filter FPGA.

4.2 Performance Requirements

1. The Filter FPGA will develop its own internal 256 MHz (SCLK_256) clock from the 128 MHz system clock (SCLK). SCLK will have a maximum cycle-to cycle jitter of ±100 picoseconds. The relative timing of the wideband inputs relative to SCLK_256 must be adjustable independently for each FPGA to simplify PCB clock routing. The actual position of the rising edge of the clocks should be set to maximize the reliability of capturing the bits correctly. The 128 MHz output clocks are rising/falling edge coincident with their associated bit cells and are derived from SCLK to keep from accumulating clock jitter. ICLK, OCLK, ACLK, BCLK and CCLK may not used as clocks but as even/odd indicators.

2. It must be possible to operate the FPGA within the range of 128 MHz ±1%.

3. The synchronous MCB interface shall be capable of operating with a clock that is neither frequency nor phase synchronous with the system clock. The FPGA will support an MCB interface clock with a maximum rate of 33 MHz. Because of delays in MCB_DATA reaching the PCMC, read data should appear on the MCB_DATA lines at least 10 nanoseconds before the next rising edge of MCB_CLK. The MCB_DATA output drivers must be 12 milliamp drivers and have a transition time between 1 and 5 nanoseconds.

4. The power dissipation of the FPGA, running at 256 MHz with all four filter stages operating shall not exceed 7 Watts; however, the goal for the maximum power dissipation is 6 Watts.

5. The FPGA can have multiple voltage supplies for different I/O standards and on for the core voltage. The core voltage is specified at 1.2V±5%. An adequate number of power and ground pins must be provided for the expected number of voltages, inputs, outputs and power dissipation.

6. The absolute maximum DC input voltage relative to ground on any pin is –0.5V to V_{CCO}+0.5V, where V_{CCO} is the "rail" voltage of the particular I/O standard.

7. The device must tolerate transient voltages (overshoot and undershoot) during transitions for durations of <10 nanoseconds and current <100 mA of a minimum of –2V and a maximum of V_{CCO}+1.0V, relative to ground.

8. The mean time before failure (MTBF) of the Filter FPGA must be greater than 10 million hours.

9. The device must meet JEDEC standard JESD22-A114 (Mil Std 3015.7) for ESD with full qualification at 1000 V minimum.

10. The FPGA technology must be resistant to "latch-up" and tested as per JEDEC standard No. 78 entitled "IC Latchup Test".

4.3 Environmental Requirements

1. The Filter FPGA will be surface mounted on the Station Board PCB. It is likely that a single large heat spreader/sink will cover all 36 Filter FPGAs. This monolithic heat sink will be required to keep the junction temperature below 50 °C. The MTBF is reduced by a factor of about two for each 10 °C increase in the junction temperature.

2. The Filter FPGA must be packaged in a "flip-chip" (cavity down) package no larger than 35 mm on a side for board space and thermal considerations. A smaller package is desirable. The actual package is "flip-chip" 27 mm square.

3. The board will use forced-air cooling with a normal operating ambient temperature of 10-15 °C and with a maximum ambient temperature of 40 °C.

4.4 Interface Requirements

4.4.1 Master Reset

- RESET* is the input low-true master reset signal.

- RESET* is LVTTL.

4.4.2 System Input Signals

- SCLK is the system 128 MHz clock. This clock is derived on the Station Board from a 64 MHz that comes from the TIMECODE Generator Box (RFS Document AN25151N0000) using a crystal phase-locked loop. All FPGAs on the Station Board use SCLK in order to prevent the buildup of clock jitter. Input signals from the previous FPGA are clocked in using both edges of a 256 MHz version of SCLK. The CMIB can choose which edge to use. The edge required is dependent on the length of PCB traces and will be constant for a specific FPGA on all boards.

- STICK is the system timing tick (10 millisecond period). It is clocked in DDR like the data (above) and the CMIB can choose which edge to use. It is used as the stop pulse for a time interval counter.

- All system input signals are 2.5V LVDS.

4.4.3 Wideband Input Signals (A and B)

- ICLK is the 128 MHz clock accompanying the wideband input signals. It is not used as a clock in the Filter FPGA but is used for system clocking alignment

- IDATA[63:0] is clocked in by SCLK_256. IDATA is the input data "highway" and can be in any of the organizations listed in Table 4-1. IDATA[63] is the most significant bit of the oldest sample.

- IVALID is clocked in by SCLK_256. If low, the associated data is invalid and does not form part of the output data.

- INOISE is clocked in by SCLK_256. If high, the calibration noise diode at the antenna was on for the associated samples. If low, the noise source was off.

- ITICK is clocked in by SCLK_256. ITICK goes for high for one half a cycle of ICLK every tick time (10 milliseconds) and marks the time of the corresponding sample in IDATA. Delay and phase models sent by the CMIB take effect on the first sample following ITICK.

- IDFRM is clocked in by SCLK_256 even though it is a 128 Mb/s signal. IDFRM occurs every 20 cycles of ICLK and is time-aligned to ITICK. The width of IDFRM is one cycle of ICLK.

- IDERR is a serial data stream clocked in by SCLK_256 even though it is a 128 Mb/s signal. A delay error frame consists of 16 bits of delay error (fraction of a sample) and 4-bit pattern (0xA) as a serial stream. IDFRM is opposite the first (LS) bit of IDERR. The Delay Module (RFS Document AN25043N0000) produces IDERR. The delay error is sent in advance of the data to which it

applies; therefore, the delay error belongs to the IDATA word immediately after IDFRM. The delay error is defined such that:

- ***desired delay = actual delay + delay error***
- ***+0.5 (0x7FFF) > delay error >= -0.5 (0x8000)***.

- All wideband input signals are 1.5V LVCMOS but could be HSTL.

4.4.4 Wideband Output Signals

- OCLK is a 128 MHz clock derived from SCLK_256. It is used to clock in its accompanying data by the next FPGA in the chain. The timing relationship of OCLK to its accompanying data is the same as for ICLK.

- ODATA, OTICK, OVALID, ONOISE, ODERR and ODFRM are copies of IDATA, ITICK, IVALID, INOISE, IDERR and IDFRM are clocked out by SCLK_256. The output signals will suffer a pipeline delay of a few clocks as they pass through the Filter FPGA and will accumulate with each FPGA chained together. This delay can be compensated for by a CMIB setting.

- All wideband output signals are LVCMOS.

4.4.5 Narrow Band Output signals

The following signals go from the Filter FPGA to the Output FPGA (DATA), VSI FPGA (DATA) or Timing FPGA (PERR) where all the filtered outputs are time aligned, switched, formatted and sent to the backplane. The data and phase error samples are accompanied by timing and clock signals to make this bundle of signals independent of trace lengths on the Station Board (all traces inside the bundle should have the same length).

- (A/B)DATA is the quantized output data samples from the filter. The quantization may be from 1-bit to 8-bits limited by the 4-bit output data path and the output bandwidth. For up to 4-bits the samples are clocked out at 256 MHz. For more than 4-bits the two nibbles are clocked out sequentially with the LS nibble first and the LS part of the sample aligned with TICK. Samples are justified to the LS end and the MS end is sign-extended to fill the 4-bit or 8-bit samples. For 4-bit samples, an invalid sample is encoded into the data by using the forbidden state 0x8. The same scheme (using the forbidden state 0x80) could have been used for 5-bit to 8-bit samples but the Baseline Board only accepts 7-bit samples so the valid signal is encoded in the MS bit.

- CPERR is the 8-bit parallel phase error, produced in the Filter FPGA at 128 MHz and is clocked out as two nibbles at 256 MHz. The LS nibble of the phase error is clocked out first and is time aligned with TICK.

- (A/B/C)TICK determines the timing for DATA. The output data is a filtered version of the input data and hence suffers the group delay of the filter. TICK is delayed to match the data timing and any other pipeline delays in the FPGA. Both DATA and TICK may be further delayed in the Filter FPGA so that the TICKs from different bandwidths are aligned at the Output FPGA.

- (A/B/C)SIND is a sample indicator that shows where the samples are located. For 256 Ms/s it will always be 1. For 62.5 ks/s 4-bit samples it will be 1 for only 1 clock and 0 for 4095 clocks. For 8-bit samples it will be 1 for both the MS and LS parts of the sample.

- All narrow band output signals are 1.5V HSTL.

4.4.6 MCB Interface Signals

- MCB_ADDR[7:0] is the input 8-bit address bus for accessing internal Filter FPGA configuration, monitor and control registers.

- MCB_DATA[15:0] is the bi-directional 16-bit microprocessor data bus.

- MCB_CS* is the input low-true FPGA select that enables the MCB interface drivers.

- MCB_CLK is the input clock for the synchronous MCB interface. The phase and frequency of MCB_CLK is independent of SCLK.

- MCB_RD/WR* is the input read/write enable.

- All MCB signals are LVTTL.

4.4.7 Test Port

These four outputs can be attached to a number of internal signals TBD to provide a simple diagnostic capability.

4.4.8 System Interface Timing Requirements

Shown below is the functional relative timing for the system input signals.

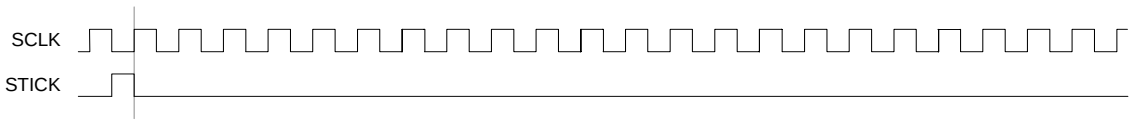

Figure 4-1 System Input Relative Timing

The relative timing inside and outside the FPGA are the same as shown below. The left vertical line indicates the position of the internally generated rising clock edge that is used to clock the I/O registers. The actual position of the 256 MHz clock is at 0.75 of the bit cell. Changes in the value of the output data coincides with the falling edge of the tick, regardless of the decimation ratio.

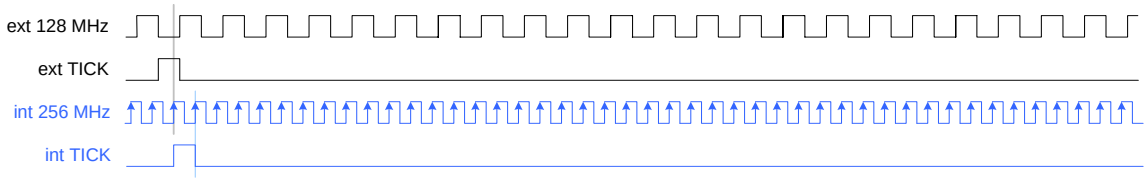

Figure 4-2 External/Internal Relative Timing

4.4.9 Wide Band Data Interface Timing Requirements

Shown below is the functional relative timing for the wide band input and output signals. The signal names are prefixed with I for input and O for output.

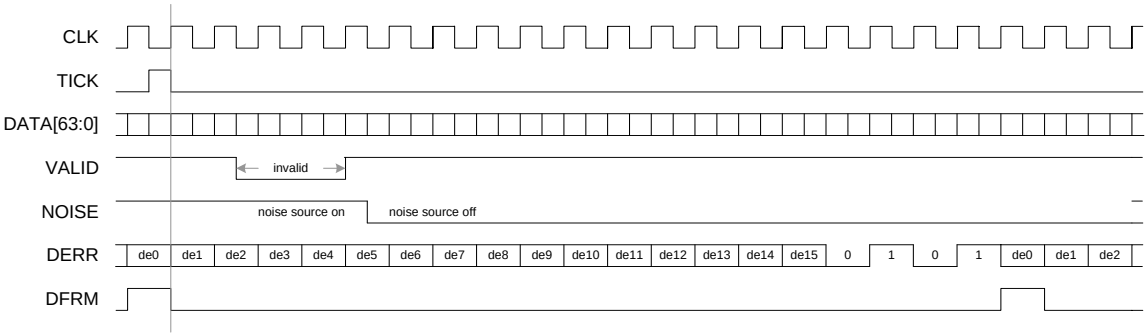

Figure 4-3 Wide Band Input and Output Relative Timing

The 16-bit delay error represents the fractional sample error and arrives from the Delay Module in serial form with a 4-bit pattern for signal integrity checking. The LS bit of the delay error is sent first and the pattern is sent at the end. Delay changes in the Delay Module can only occur at DFRM and the previous delay error refers to the sample in the 2nd half of DFRM. The wide band input external/internal relative timing is the same as figure 4-2.

4.4.10 Narrow Band Data Interface Timing Requirements

Shown below is the functional relative timing for the narrow band output signals.

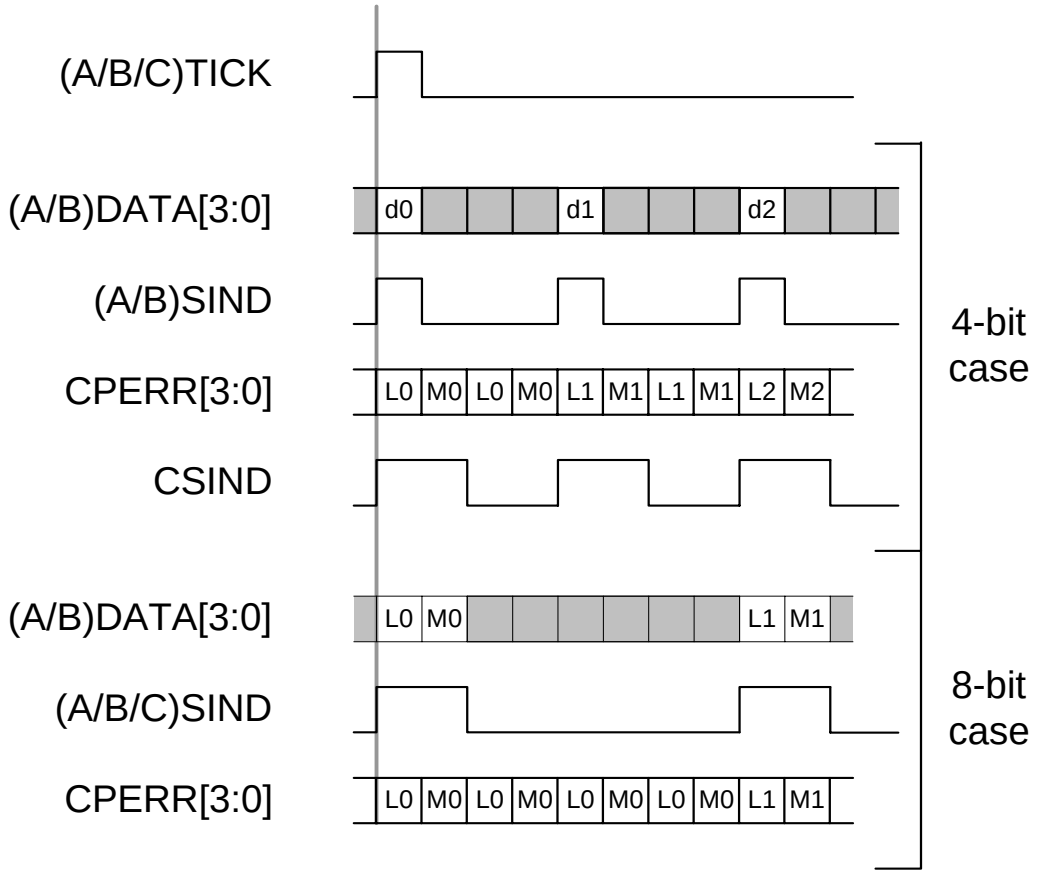

Figure 4-4 Narrow Band Output Relative Timing

The CPERR waveform shows the position of the LS and MS nibbles of the (always) 8-bit phase error relative to CTICK. If the width of the output data is more than 4 bits, then the LS and MS nibbles are in the same relative location as those for CPERR. The grey areas are "don't care" for the data and will be set to the invalid state. The Baseline Board is expecting CPERR all the time.

4.4.11 MCB Interface Timing Requirements

The MCB (Monitor & Control Bus) interface allows a microprocessor (CMIB) to write to and read from the FPGA. Writing will be used to configure and control the hardware. Reading will be used to verify the configuration, receive status indicators and obtain monitor information and other data.

The write cycle is shown in Figure 4-5. The microprocessor writes to the FPGA by putting the data on the MCB_DATA bus, the target register address on the MCB_ADDR

bus, driving MCB_RD/WR* and MCB_CS* low after a rising edge of the MCB_CLK (**A**) and keeping the signals stable until after the next rising edge of the MCB_CLK (**B**). The Filter FPGA captures the information at this time (**B**). The Filter FPGA does not support burst mode writing or reading.

The read cycle is shown in Figure 4-6. The microprocessor reads from a FPGA by driving MCB_RD/WR* high, putting the desired register address on the MCB_ADDR bus and driving the corresponding MCB_CS* low after a rising edge of MCB_CLK. The address is valid at the next rising edge of MCB_CLK (**B**). The Filter FPGA places the requested data onto the MCB_DATA bus before the next rising edge of MCB_CLK (**C**) and holds it stable for as long as MCB_CS* is low. It should be noted that for the read case, the MCB_CLK from the PCMC through the MCB FPGA and into the Filter FPGA suffers a delay. For the data path back to the PCMC the delay is even worse (bus selectors in the MCB FPGA); therefore, the data should be placed onto the MCB_DATA bus as soon as possible.

Setup time t_{su} > 15 ns.

Clock to Output time t_{co} < 13 ns.

Hold Time t_h > ? ns.

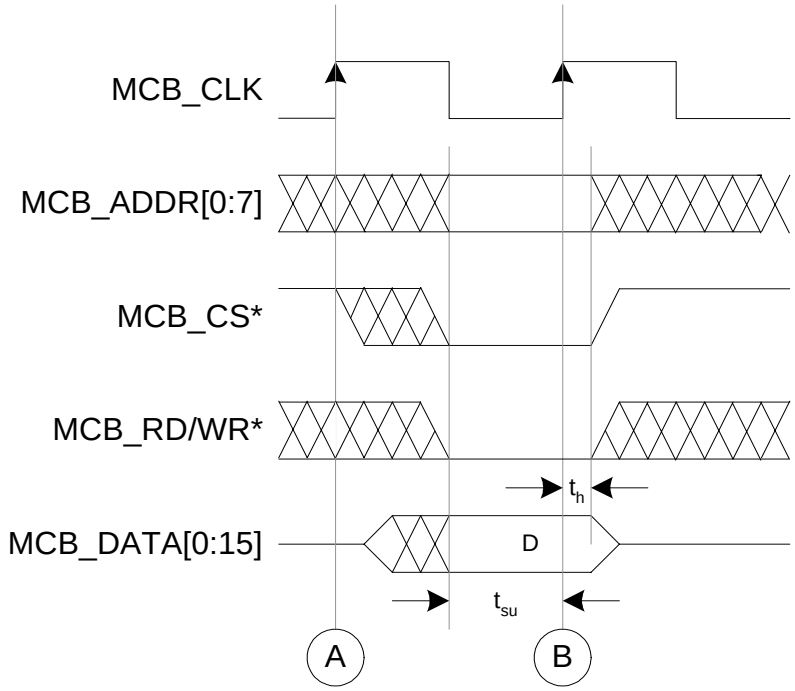

Figure 4-5 MCB Interface WRITE Relative Timing

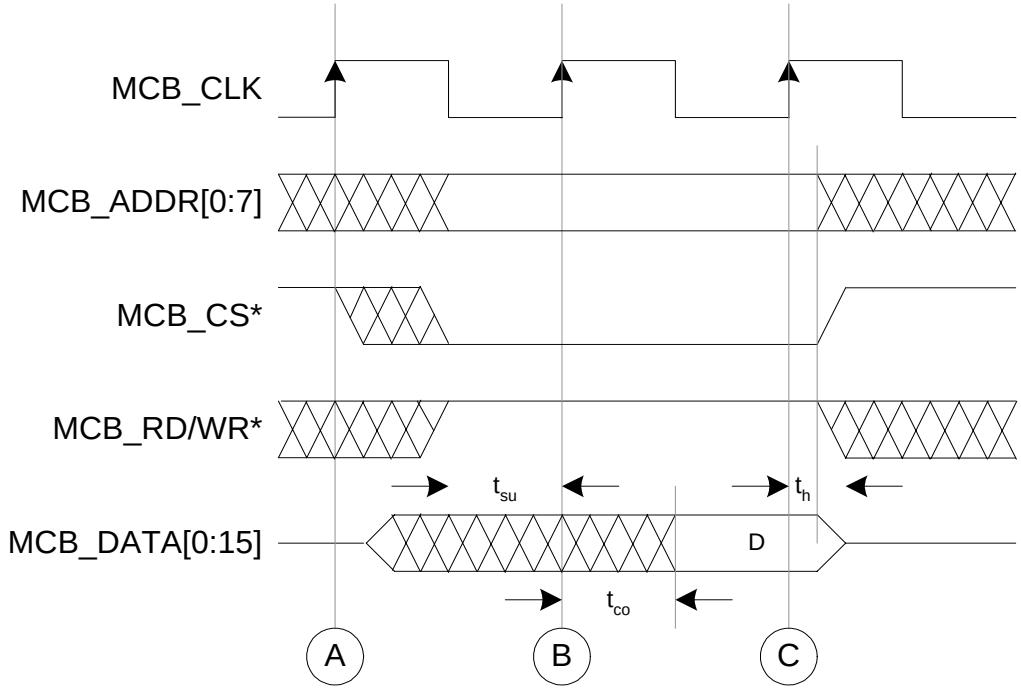

Figure 4-6 MCB Interface READ Relative Timing

The microprocessor will be interrupted on a version of STICK that has been delayed sufficiently to make sure that all FPGAs on the board have received their local version of the TICK. This allows for a pipeline delay through the FPGAs on the board, including the tap registers in the Filter FPGA. Configuration information sent to the Filter FPGA may take effect immediately; therefore, it should only be sent when the output of the FPGA is irrelevant. Model information sent from the microprocessor to the Filter FPGA from within the Interrupt Service Routine (ISR) will be become active in hardware at the next TICK. Other information, such as which state to count, must be in this category as well. Monitor information, such as counts or accumulations, read by the microprocessor from within the ISR will be the result of operations occurring between the previous two TICKS. It is assumed that status information, such as error bits, will be read by the ISR on every interrupt (TICK); therefore, the information is latched by the TICK and the primary register cleared to ensure that no momentary error bits are missed or counted twice.

5 Hardware Functional Design

The wide band input data

5.1 MCBI

The MCBI block contains the register set. The configuration information contained therein is fed to the other blocks as "constant" input. Some of these inputs will be latched in the block by the local version of the tick to insure correct timing. The other blocks may also return information that has usually been latched in the block by the local tick to insure correct timing. By "timing" in these cases we mean, for example, that a power measurement has been made by summing the squares of the input data over a precisely known time period or that the delay model takes effect at an exactly known time relative to the data.

5.2 INOUT

A simplified block diagram of the INOUT block is shown in Figure 5-1. INOUT receives data from the previous FPGA (usually another Filter FPGA), clocks it in with both edges of an internal 256 MHz version the system clock (SCLK_256). The CMIB can choose which edge to use. The same signals are clocked out with SCLK_256 for use by the next Filter FPGA. Using a first generation of the system clock (rather than passing the clock from one FPGA to the next) prevents buildup of clock jitter (Xilinx uses a DLL instead of a PLL for internal clock generation which always causes the clock jitter to increase).

In order to simplify PCB layout, the Filter FPGA has two input and two output wideband data paths. The CMIB selects which input to process and which, if any, outputs to enable. Only one input can be processed but zero, one or two outputs can be enabled, depending on the position of the Filter FPGA on the Station Board. Outputs that are not enabled consume less power. The CRC calculation can used by the CMIB to detect transmission errors between FPGAs.

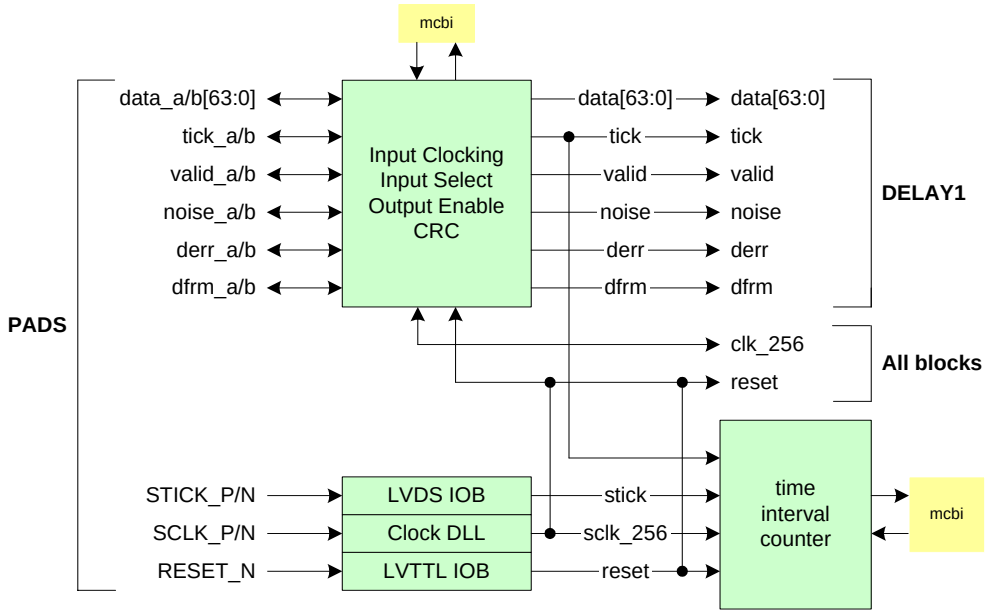

Figure 5-1 Block diagram of INOUT

5.3 DELAY1

A simplified block diagram of DELAY1 is shown in Figure 5-3. DELAY1 puts a delay in the data relative to the timing. The delay to be implemented is given to the Filter FPGA in the form of a delay and delay rate to be correct on the next tick. If the delay is zero, then the output signals should look exactly like the input signals relative to each other. A constant pipeline delay is OK. Of course, things are not that simple. The wideband input data has been previously delayed by a similar, but usually much larger and faster varying delay model in the Delay Module. Since delays can only be accurate to ±0.5 samples, there is an error in delay. This error needs to be corrected on the Baseline Board. DERR is 16 bits of fractional delay between ±0.5 (not including +0.5) followed by a 4-bit check pattern all clocked at 128 MHz. The 20-bit frame begins on DFRM and the first bit is the LS bit of the error. This fraction must be made into a parallel 16-bit word and combined with the model sent by the microprocessor to get the actual delay. In addition, a new fractional bit error must be calculated, converted to a phase error by a multiplication with a CMIB supplied constant and truncated to 12 bits. The data is presented to the RAM as two consecutive words (128-bits wide) at 256 Mw/s (or less if the input data is decimated). This arrangement (two words instead of one) is used to ensure that no data is lost by skipping a 64-bit word of data as the delay is decreasing.

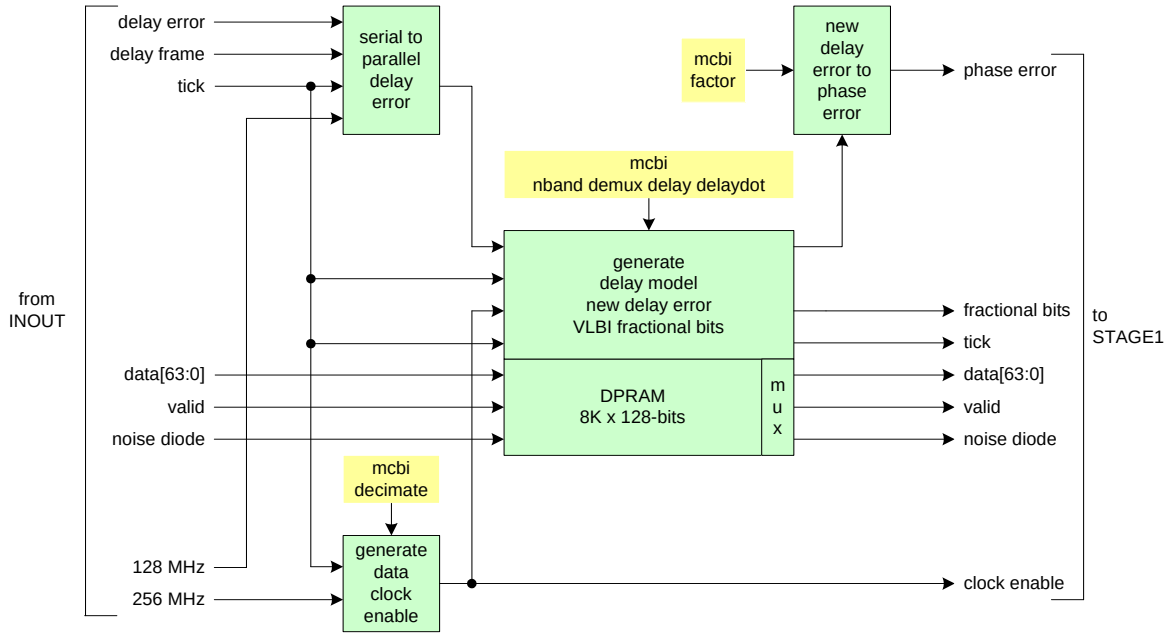

Figure 5-2 Block diagram of DELAY

5.4 STAGE1

A simplified block diagram of the STAGE1 filter is shown in Figure 5-4a.

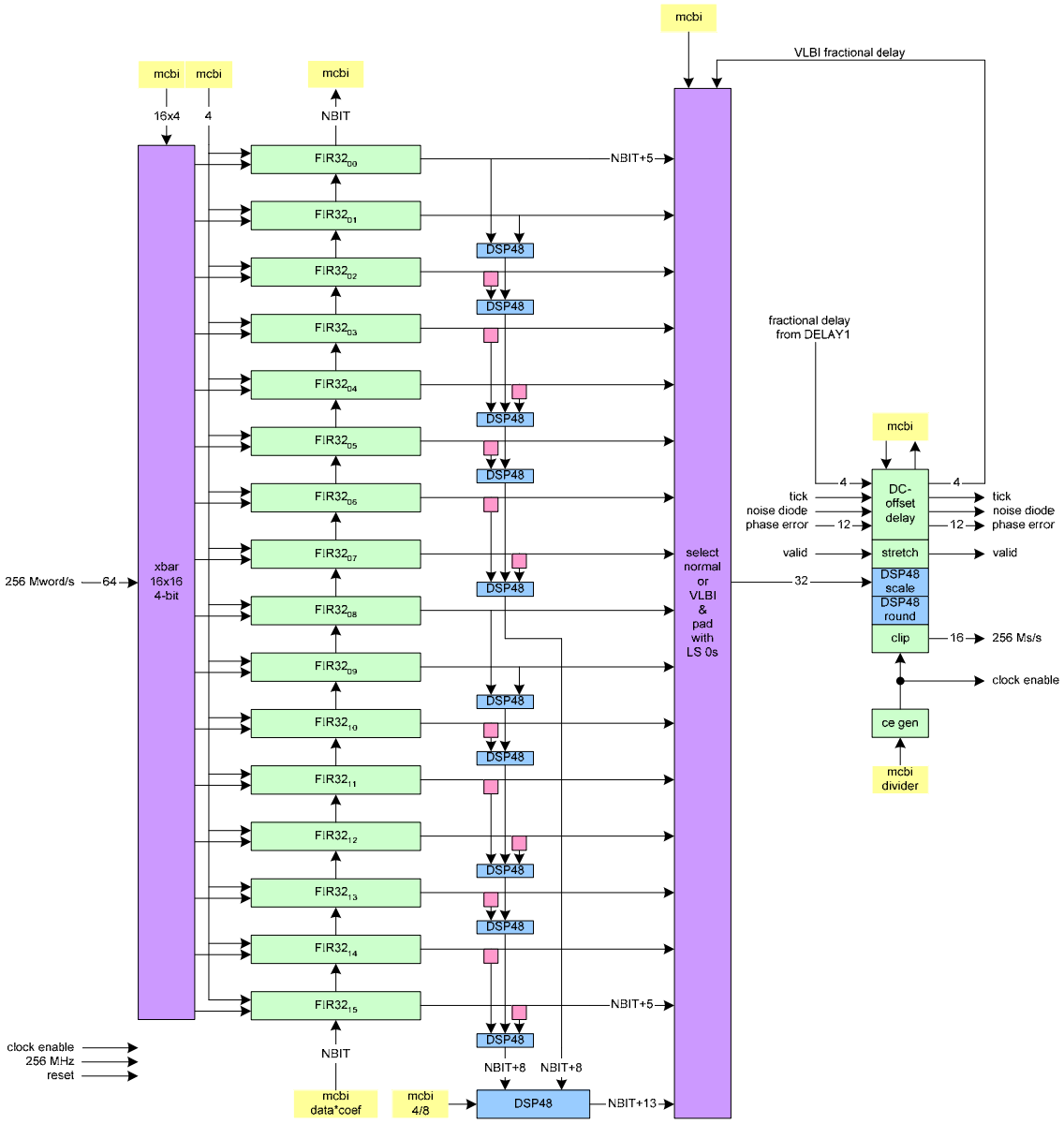

Figure 5-3A Block diagram of STAGE1

STAGE1 begins with a 4-bit 16 by 16 crossbar switch that allows all the data organizations of Table 3-1. For example, to cope with the normal 8-bit EVLA mode (the 64-bit data path is composed of eight 8-bit samples) the MS 4-bits of each sample is fed to upper eight 32-tap FIR filters while the LS 4-bits are fed to the lower eight. The final sum is the upper half sum shifted up by four bits and added to the lower half sum. The input data is only 4 bits wide; therefore, the most efficient implementation of STAGE1 in

terms of gate consumption uses NBIT wide by 16 deep lookup tables (LUTs) for the product of the data times the coefficients. NBIT is the number of bits of the resulting product and was originally meant to be 12 to give the required 60 dB out-of-band suppression but simulations show that a bit is lost in the 8-bit approximation and a NBIT of 13 (or more) is desirable. STAGE1 can operate as a 512-tap poly-phase FIR filter or as 16 independent 32-tap filters.

The heart of STAGE1 is the 32-tap FIR (FIR32) filter shown in Figure 5-4b. It is a standard FIR filter implementation using 4-bit in and 16-bit out look-up tables (LUTs) for multipliers. The 4-bit input data can be selected from two sources, the output of the crossbar switch (normal) or the CMIB (allows the LUT to be loaded). The LUTs are joined together as a 16-bit wide parallel shift register and are connected to the FIR32 in a long chain of the whole 512 taps. The far end of each chain can be read by the microprocessor to verify correct operation.

The combined block at the output Figure 5-4a is common to all four filter stages. It takes the most significant 32 bits of the final sum as an input and applies scaling, rounding and clipping to produce a 16-bit result. The scaling is supplied as a 16-bit integer by the CMIB. A value of 0x0001 means that the bits [31:16] (which can never overflow) are taken and rounded by the bit 15 to produce the STAGEx result. Values greater than 1 increase the output RMS and overflow can occur so the 16-bit result is clipped to +32767 or –32767. It is assumed that clipping is the result of statistical noise spikes and is treated as valid data.

In the VIRTEX-4 implementation of the Filter FPGA, the DSP48 slices are used for multipliers and adders in order to reduce the slice count and the power dissipated. Because of the architecture of the DSP48, adder trees are implemented as ‘systolic adders’ (the adders are in a chain and add the input to the previous sum) where possible.

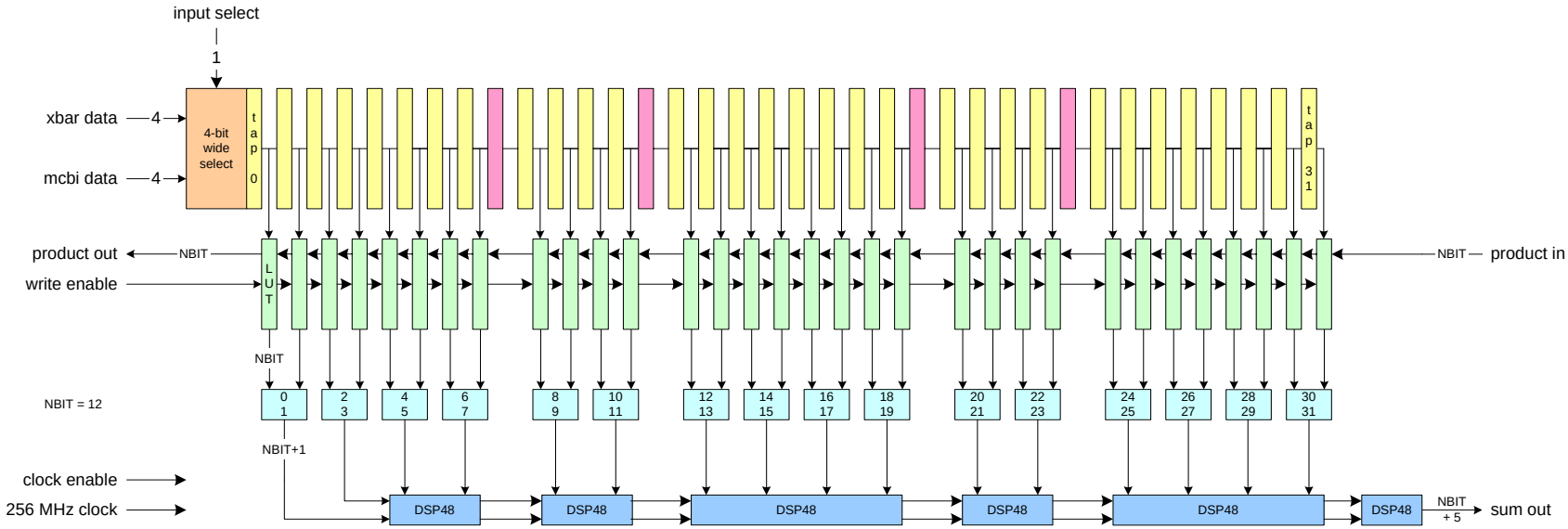

Figure 5-3B Block diagram of STAGE1 FIR32

The valid signal can be altered under program control to make sure that invalid samples in the FIR taps do not contribute to a valid output sample. The tick, noise and perr signals can be delayed under program control so that they match the effective pipeline delay of the FIR filter. This is only accurate if an odd number of taps is used so that the pipeline delay is an integer number of input samples (set the unused tap to zero). These functions are the same for all four stages of the filter.

5.5 STAGE2

STAGE2 consists of a phase shifter followed by a FIR filter with up to 512 taps. The shifter section allows bandwidths narrower than 128 MHz to come from anywhere in the 128 MHz band rather than just from integral slots. It also allows the removal of the phase shift caused by the fractional delay error and/or the Doppler shift produced by earth rotation on the Station Board rather than on the Baseline Board. The correction of the phase error due to fractional delay error may be necessary for the narrowest bandwidths because the delay error can go through several cycles during the effective time constant of the FIR filter. The shifter and FIR sections run at 256 MHz. The STAGE2 filter section allows a decimation of 2, 4, 8 or 16 and requires 32 multipliers and associated small dual-port RAMs to yield 64, 128, 256 and 512 taps respectively. The number of taps is reduced by a factor of two when the shifter is used and the filter coefficients include a phase shift of -45° in the upper half and +45°in the lower half to produce the quadrature outputs that are then added to complete the single sideband mix. The shifter can be bypassed. Of course, the filter coefficients are different and arranged differently for the two cases.

5.6 STAGE3

STAGE3 is similar to the filter part of STAGE2 but only uses two 16-bit by 16-bit multipliers that run at 256 MHz because the input data rate has been reduced by STAGE1 and STAGE2 to a maximum of 16 Ms/s.

5.7 STAGE4

STAGE4 is similar to STAGE3 but only requires one-eighth of a multiplier; therefore, the number of taps could be increased by a factor of four with a corresponding increase in the size of the data DPRAM and the coefficient RAM. This is not done here to save resources; however, 512 taps can be achieved for all decimations.

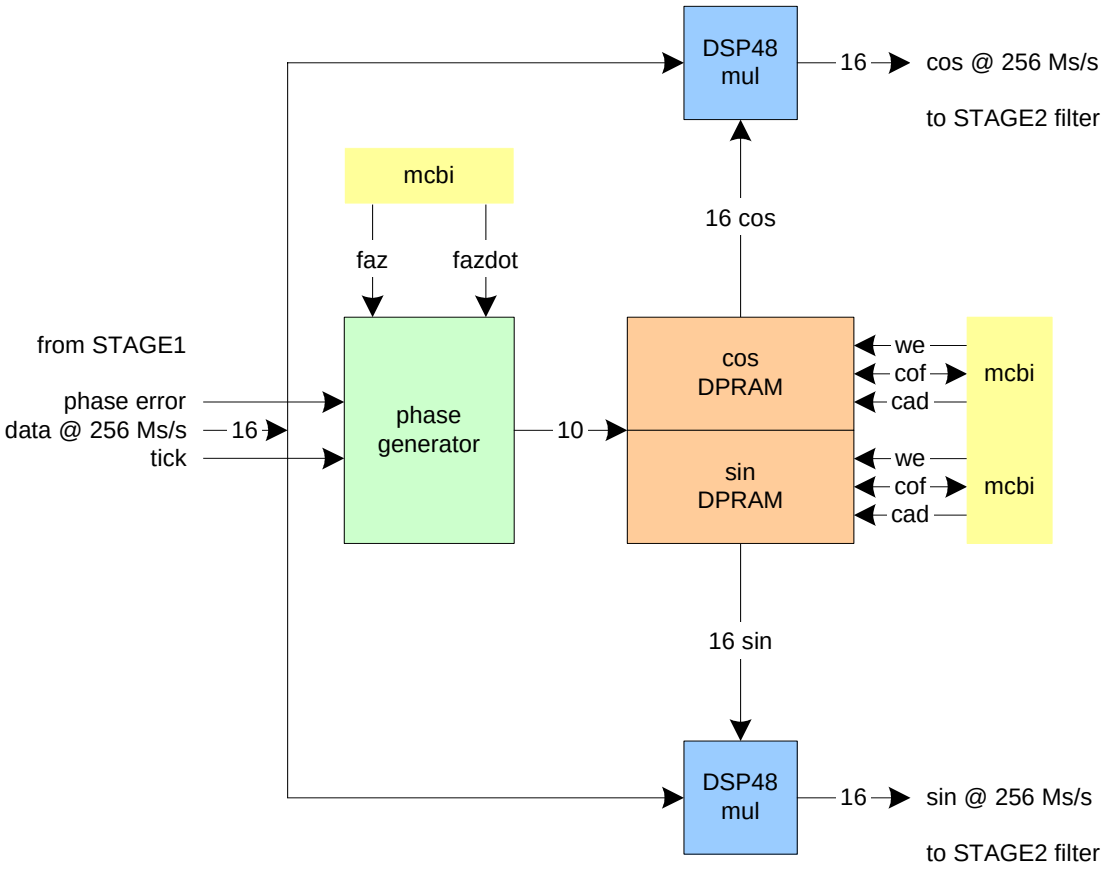

Figure 5-4A Block diagram of the STAGE2 mixer

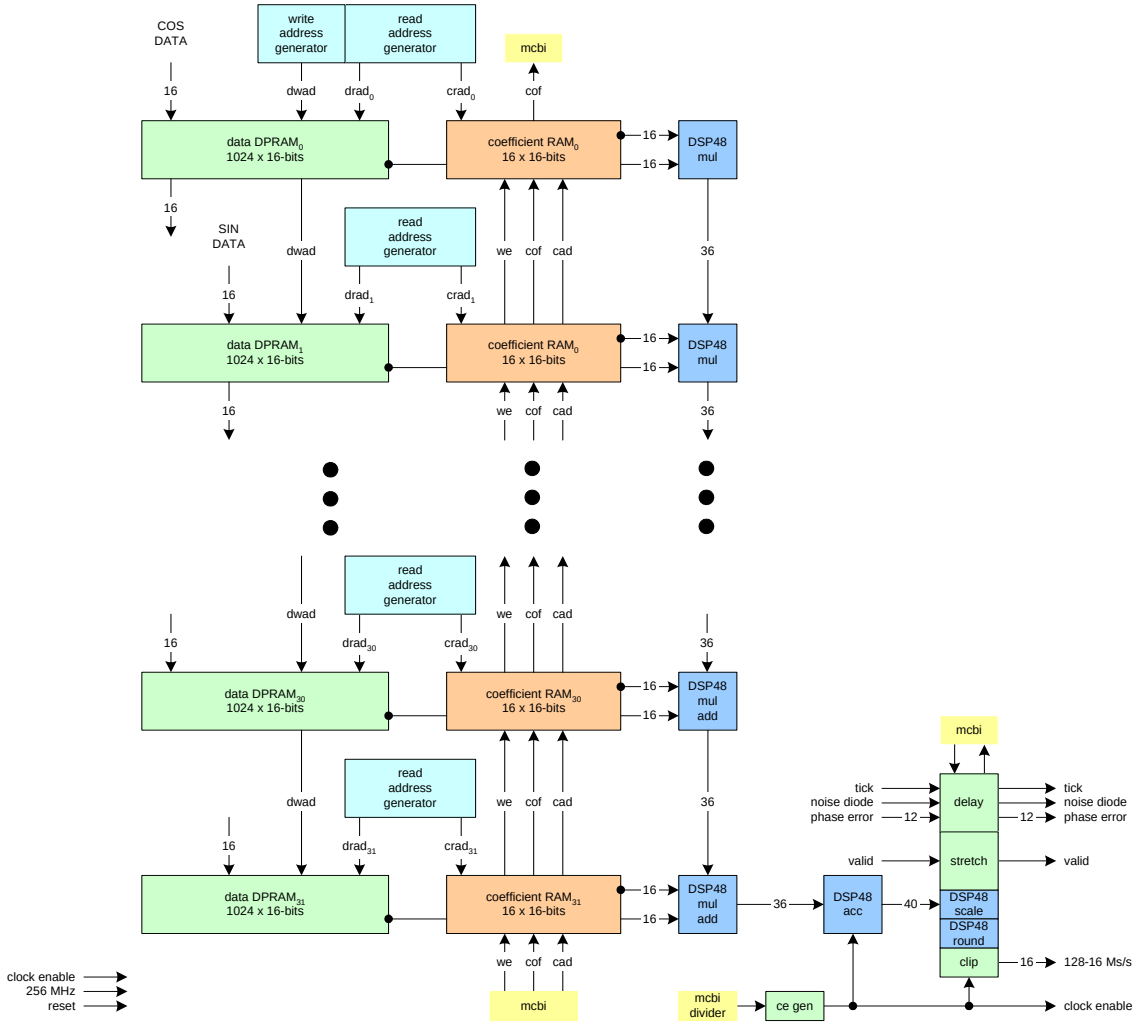

Figure 5-4B Block diagram of the STAGE2 filter

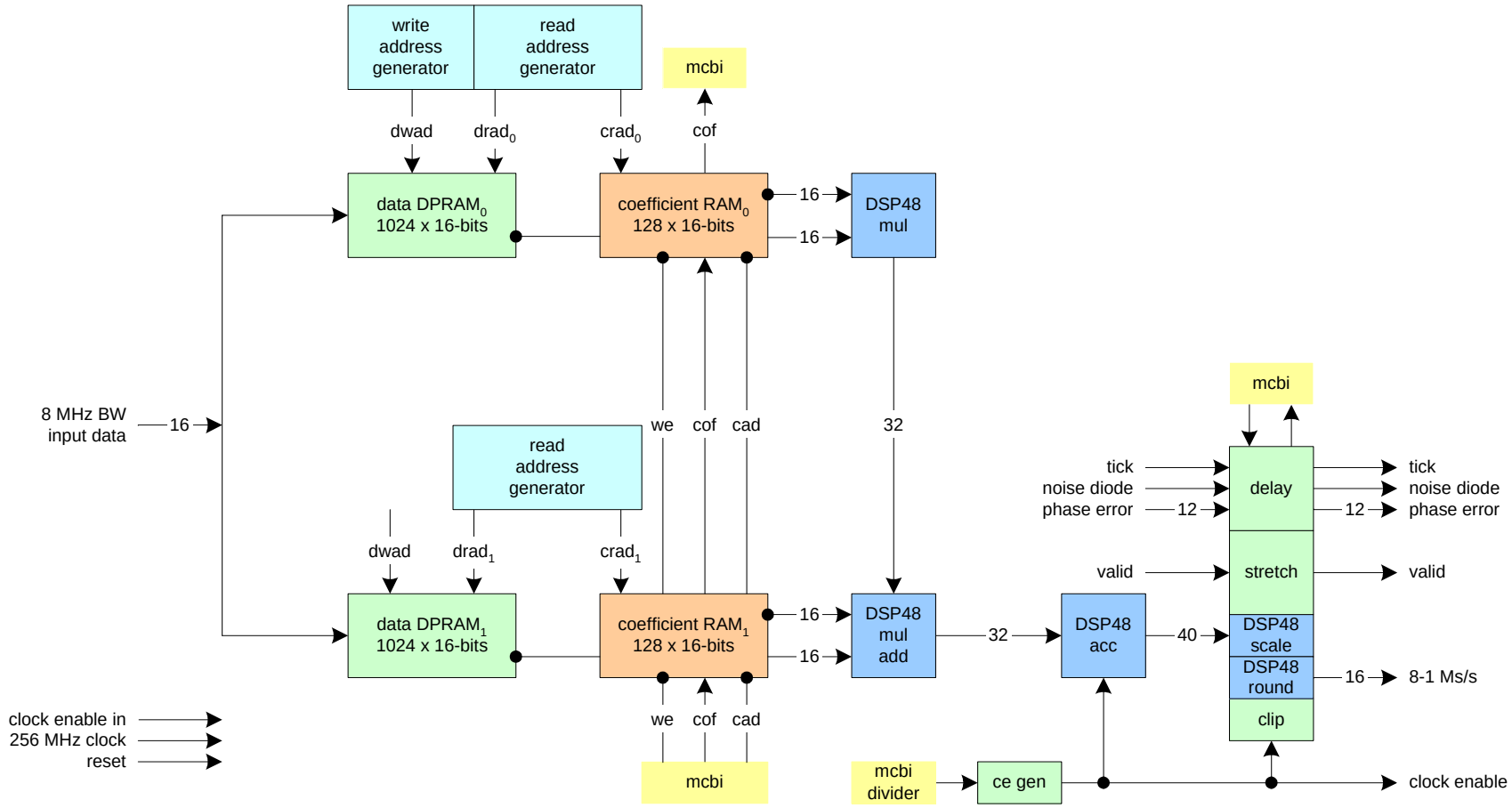

Figure 5-5 Block diagram of STAGE3

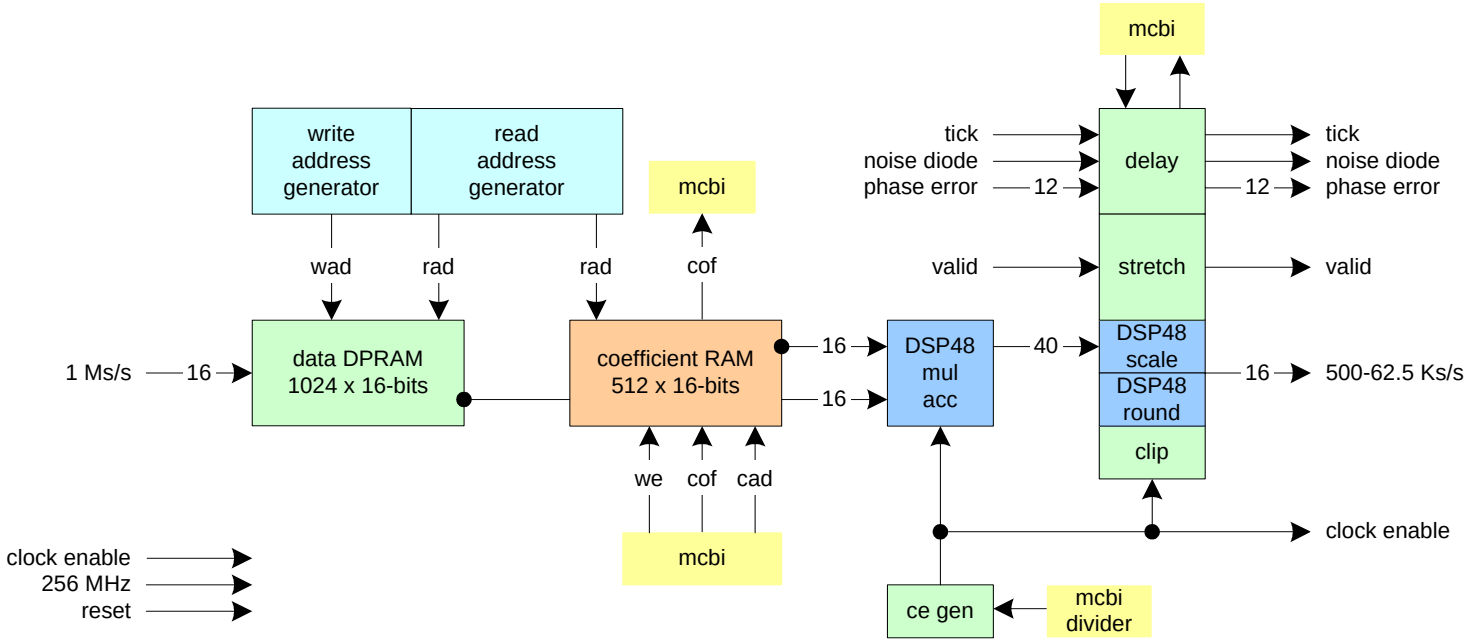

Figure 5-6 Block diagram of STAGE4

5.8 FORMAT

A simplified block diagram of FORMAT (historical name) is shown below. It makes a number of measurements on the data for later calibration purposes and quantizes the data from four to eight bits. The four inputs come from the four filter stages. The 16-bit data samples are assumed to have amplitudes near but greater than the desired output amplitudes so that multiplication by a 16-bit fraction between 0.5 and 1 and rounding will yield the desired results.

The “select” block makes two choices: which stage to use for counting clipped input data and which stage to use for the main processing and output.

The “clip counter” counts the number of valid clipped samples between ticks from any of the four FIR stages.

The “RFI processor” detects whether the absolute value of the data is greater than the supplied detection level and invalidates the data for the requested length of time. In addition, the number of valid detections is counted.

The “sideband flipper” block flips the sign of every other independent sample synchronously with the accompanying TICK when enabled.

The “power meter” block measures the signal power between TICKs by multiplying each valid sample by itself and accumulating the result. It is necessary to keep separate accumulations for noise diode on and off. TICK latches the result and clears the accumulation. The multiplier is 16-bit by 16-bit giving a 30-bit result (positive) and 22 more bits are necessary to accumulate between TICKs forcing a 52-bit accumulator.

The “quantizer block” consists of a fractional multiplier for scaling followed by rounding and a clip detector and counter. The clip detector needs to know the number of bits desired which can be 4, 5, 6, 7 or 8. The output data is clipped but the valid line is not changed.

The “valid counter” block counts the number of valid samples between TICKs. It is necessary to keep separate accumulations for noise diode on and off. The maximum number of samples per TICK (10 milliseconds) is 2,560,000 so that two 22-bit counters are required and the value latched and cleared by TICK.

The “quantized power meter” block measures the post-quantizer power using only valid quantized samples but the noise diode state is ignored.

The “tone extractor” cross-correlates the data with the cosine and sine of a supplied phase model. The phase model generator runs at the output date rate. The phase model given to the tone extractor will have to take into account the delay corrections and frequency translations done ahead of this point.

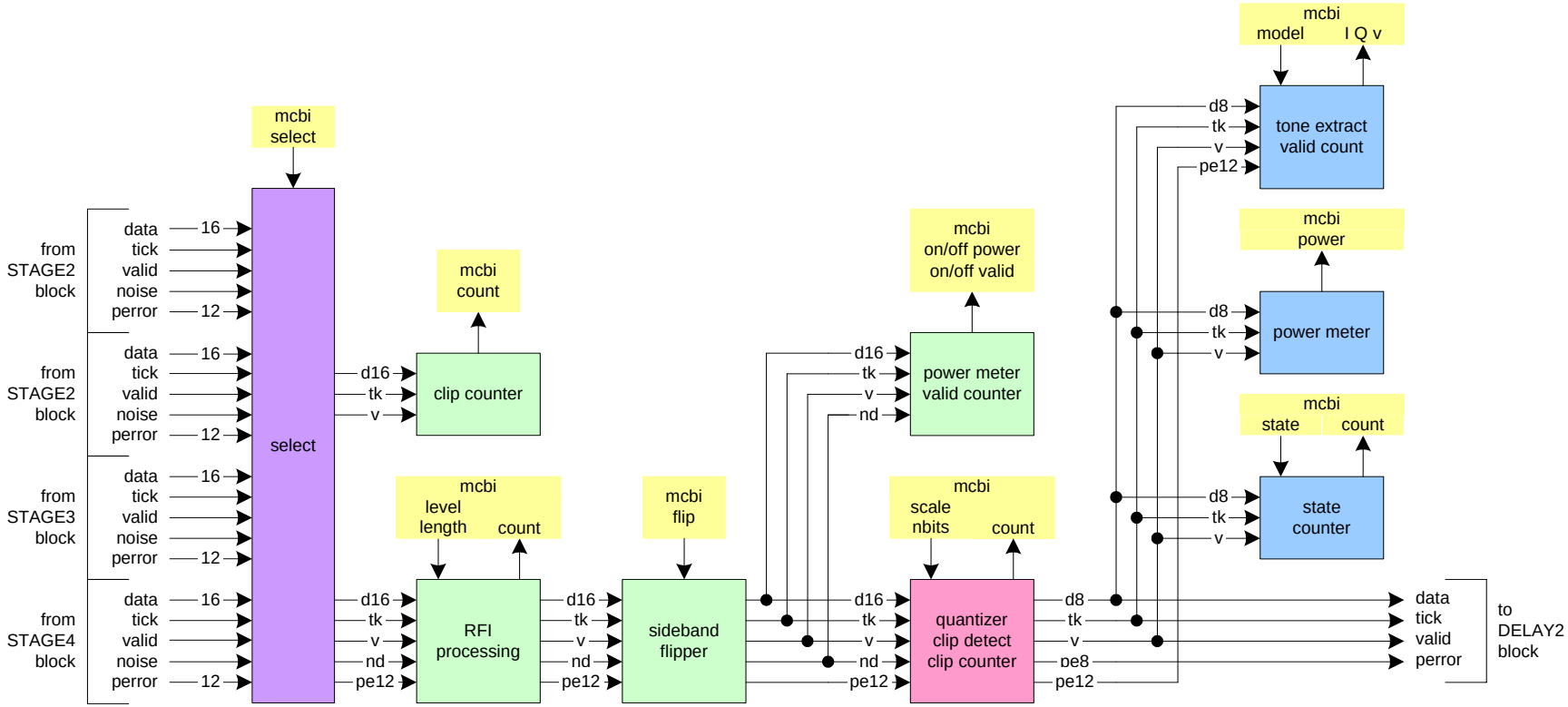

Figure 5-7A Block diagram of FORMAT

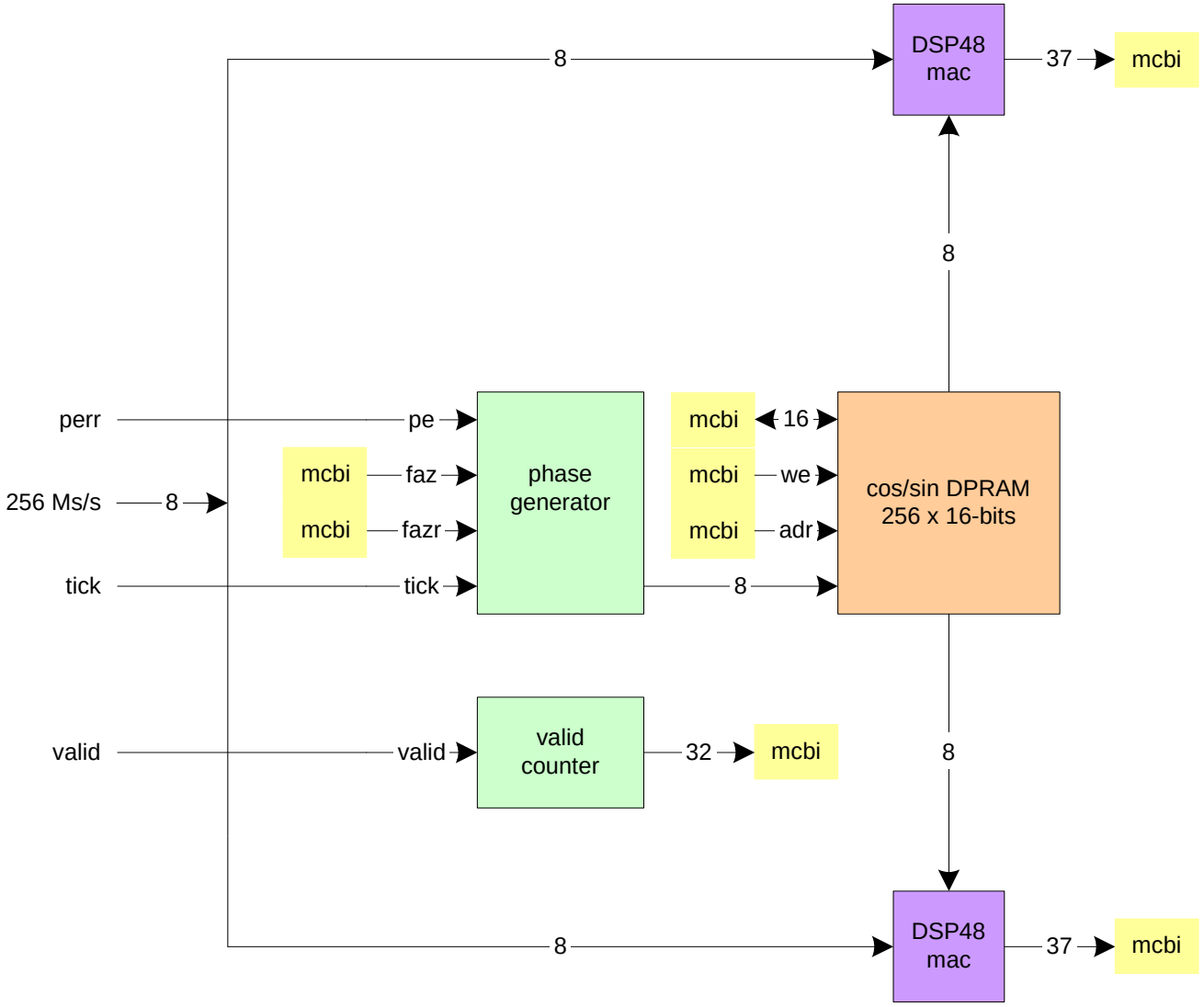

Figure 5-7B Block diagram of the FORMAT tone extractor

5.9 DELAY2

DELAY2 serializes 8-bit data, validates 4-bit and 8-bit data, calculates the 4-bit CRCs of the selected bit of the data and perr streams and delays the data + tick + sind bundles by specified amounts. Time alignment of the samples for all Filter FPGA bandwidths is necessary for the proper operation of the Output FPGA (also see A250220041). It is done here to spread the logic over the 18 Filter FPGAs which already have the necessary information. The delay lines can also take care of the pipeline delays due to daisy chaining the Filter FPGAs and are independent in order to account for PCB trace length to the Timing, Output and VSI FPGAs. The value for the delays will be determined using the time interval counters in the Timing, Output and VSI FPGAs.

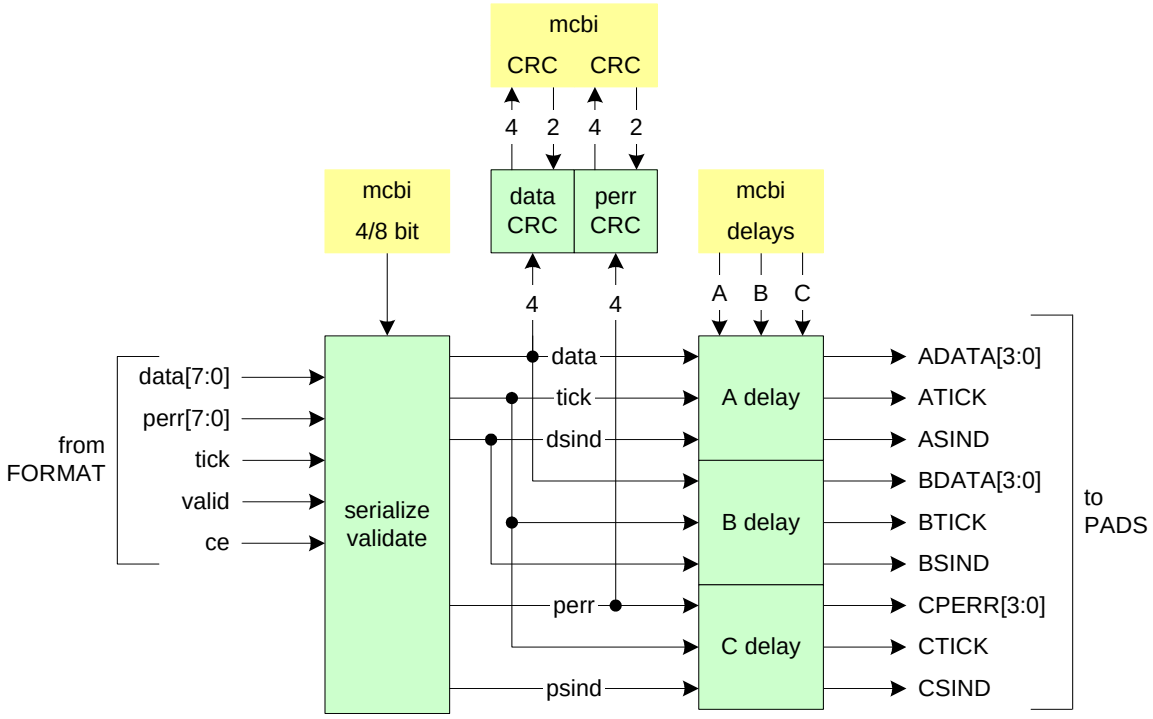

Figure 5-8 Block diagram of DELAY2

6 Software Design Information

6.1 MCB Interface Register Map Summary

A summary of the registers for the Filter FPGA is shown below. Note that the number of bits reflects the actual hardware register but on read the upper bits in the register are sign extended or zero; therefore, no masking by the CMIB is necessary when reading the full 16-bit register.

Address	Name	Dir	Bits	Description
COMMON				**Common registers**
0x00	CM_STS	R/W	15:0	Status Register
0x01	CM_CFG	R/W	15:0	Configuration Register
0x02	CM_CTL	R/W	15:0	Control Register
0x03	CM_ERR	R/W	15:0	Error Register
0x04	CM_DEF	R/W	15:0	Default Register
0x05	CM_DID	R	15:0	Design identifier
0x06	CM_TST0	R/W	7:0	Address of signal to connect to test port 0
0x07	CM_TST1	R/W	7:0	Address of signal to connect to test port 1
0x08	CM_TST2	R/W	7:0	Address of signal to connect to test port 2
0x09	CM_TST3	R/W	7:0	Address of signal to connect to test port 3

Address	Name	Dir	Bits	Description
INOUT				**Wideband input/output**
0x10	IO_ESEL	R/W	15:0	Select input data bit for CRC error
0x11	IO_DSEL	R/W	5:0	Select input data bit for CRC calculation
0x12	IO_CRC	R	3:0	4-bit CRC for selected input bit
0x13	IO_SDLY	R/W	15:0	System tick delay to time interval counter.
0x14	IO_TINT1	R/W	5:0	MS time interval count
0x15	IO_TINT0	R/W	15:0	LS time interval count
0x16	IO_SID	R/W	15:0	SID from MCB FPGA
DELAY1				**Sub-band geometric delay**
0x20	D1_DDEC	R/W	3:0	Data input rate
0x21	D1_DMUX	R/W	3:0	Demux Factor–1
0x22	D1_DLY2	R/W	15:0	Delay [47:32]
0x23	D1_DLY1	R/W	15:0	Delay [31:16]
0x24	D1_DLY0	R/W	15:0	Delay [15:0]
0x25	D1_DLYR1	R/W	15:0	Delay rate [31:16]
0x26	D1_DLYR0	R/W	15:0	Delay rate [15:0]
0x27	D1_DEPE	R/W	15:0	Delay error to phase error factor
0x28	D1_TDLY	R/W	12:0	Tick delay
0x29	D1_DERR	R	15:0	Delay error before model saved at tick
0x2A	D1_PERR	R	15:0	Phase error after model saved at tick
0x2B	D1_ODLY2	R	15:0	Output Delay [47:32]
0x2C	D1_ODLY1	R	15:0	Output Delay [31:16]

Address	Name	Dir	Bits	Description
0x2D	D1_ODLY0	R	15:0	Output Delay [15:0]
STAGE1				**Maximum input rate 4096 Ms/s Filter**
0x40	S1_DDEC	R/W	3:0	Data output rate
0x41	S1_XBAR3	R/W	15:0	Crossbar addresses for FIR32 3-0
0x42	S1_XBAR2	R/W	15:0	Crossbar addresses for FIR32 7-4
0x43	S1_XBAR1	R/W	15:0	Crossbar addresses for FIR32 11-8
0x44	S1_XBAR0	R/W	15:0	Crossbar addresses for FIR32 15-12
0x45	S1_CADD	R/W	3:0	Product address
0x46	S1_CVAL	R/W	11:0	Product value
0x47	S1_VLEN	R/W	9:0	Invalid stretch length
0x48	S1_FDLY	R/W	9:0	Filter delay
0x49	S1_SCALE	R/W	15:0	Scale factor
0x4A	S1_FBIT	R/W	3:0	Fractional bit select
0x4B	S1_IDC1	R/W	15:0	DC offset to stage 1 [31:16]
0x4C	S1_IDC0	R/W	15:0	DC offset to stage 1 [15:0]
0x4D	S1_ODC2	R/W	15:0	DC offset from stage 1 [48:32]
0x4E	S1_ODC1	R/W	15:0	DC offset from stage 1 [31:16]
0x4F	S1_ODC0	R/W	15:0	DC offset from stage 1 [15:0]
0x50	S1_VDC1	R/W	6:0	DC valid count from stage 1 [21:16]
0x51	S1_VDC0	R/W	15:0	DC valid count from stage 1 [15:0]

Address	Name	Dir	Bits	Description
STAGE2				**Maximum input rate 256 Ms/s Filter**
0x60	S2_DDEC	R/W	3:0	Output data rate
0x61	S2_CADD	R/W	3:0	Coefficient address
0x62	S2_CVAL	R/W	15:0	Coefficient value
0x63	S2_VLEN	R/W	9:0	Invalid stretch length
0x64	S2_FDLY	R/W	9:0	Filter delay
0x65	S2_SCALE	R/W	15:0	Scale factor
0x66	S2_MADD	R/W	9:0	Mixer LUT address
0x67	S2_MCOS	R/W	15:0	Mixer COS LUT value
0x68	S2_MSIN	R/W	15:0	Mixer SIN LUT value
0x69	S2_MFAZ1	R/W	15:0	Mixer phase [31:16]
0x6A	S2_MFAZ0	R/W	15:0	Mixer phase [15:0]
0x6B	S2_MFAZR1	R/W	15:0	Mixer phase rate [31:16]
0x6C	S2_MFAZR0	R/W	15:0	Mixer phase rate [15:0]
0x6D	S2_CDEC	R/W	3:0	Filter calculation rate
0x6E	S2_NTAP	R/W	8:0	Number of taps-1
0x6F	S2_WADD	R	9:0	Data RAM write address last tick

Address	Name	Dir	Bits	Description
STAGE3				**Maximum input rate 16 Ms/s Filter**
0x80	S3_DDEC	R/W	3:0	Data output rate
0x81	S3_CADD	R/W	6:0	Coefficient address
0x82	S3_CVAL	R/W	15:0	Coefficient value
0x83	S3_VLEN	R/W	9:0	Invalid stretch length
0x84	S3_FDLY	R/W	9:0	Filter delay
0x85	S3_SCALE	R/W	15:0	Scale factor
0x86	S3_CDEC	R/W	3:0	Filter calculation rate
0x87	S3_NTAP	R/W	8:0	Number of taps-1
0x88	S3_WADD	R	9:0	Data RAM write address last tick
STAGE4				**Maximum input rate 1 Ms/s Filter**
0xA0	S4_DDEC	R/W	3:0	Data output rate
0xA1	S4_CADD	R/W	8:0	Coefficient address
0xA2	S4_CVAL	R/W	15:0	Coefficient value
0xA3	S4_VLEN	R/W	9:0	Invalid stretch length
0xA4	S4_FDLY	R/W	9:0	Filter delay
0xA5	S4_SCALE	R/W	15:0	Scale factor
0xA6	S4_CDEC	R/W	3:0	Filter calculation rate
0xA7	S4_NTAP	R/W	8:0	Number of taps-1
0xA8	S4_WADD	R	9:0	Data RAM write address last tick

Address	Name	Dir	Bits	Description
FORMAT				**Measure calibration parameters**
0xC0	FM_DSEL	R/W	3:0	Select signals.
0xC1	FM_VCF1	R	5:0	Valid count [21:16] for noise diode off
0xC2	FM_VCF0	R	15:0	Valid count [15:0] for noise diode off
0xC3	FM_VCN1	R	5:0	Valid count [21:16] for noise diode on
0xC4	FM_VCN0	R	15:0	Valid count [15:0] for noise diode on
0xC5	FM_PWF3	R	3:0	Power [51:48] for noise diode off
0xC6	FM_PWF2	R	15:0	Power [47:32] for noise diode off
0xC7	FM_PWF1	R	15:0	Power [31:16] for noise diode off
0xC8	FM_PWF0	R	15:0	Power [15:0] for noise diode off
0xC9	FM_PWN3	R	3:0	Power [51:48] for noise diode on
0xCA	FM_PWN2	R	15:0	Power [47:32] for noise diode on
0xCB	FM_PWN1	R	15:0	Power [31:16] for noise diode on
0xCC	FM_PWN0	R	15:0	Power [15:0] for noise diode on
0xCD	FM_TADD	R/W	7:0	Tone LUT address
0xCE	FM_TVAL	R/W	15:0	Tone COS [15:8] & SIN [7:0] values
0xCF	FM_TFAZ1	R/W	15:0	Tone model phase [31:16]
0xD0	FM_TFAZ0	R/W	15:0	Tone model phase [15:0]
0xD1	FM_TFAZR1	R/W	15:0	Tone model phase rate [31:16]
0xD2	FM_TFAZR0	R/W	15:0	Tone model phase rate [15:0]
0xD3	FM_TVC1	R	5:0	Tone valid count [21:16]
0xD4	FM_TVC0	R	15:0	Tone valid count [15:0]

Address	Name	Dir	Bits	Description
0xD5	FM_TCOS2	R	3:0	Tone extractor COS result [35:32]
0xD6	FM_TCOS1	R	15:0	Tone extractor COS result [31:16]
0xD7	FM_TCOS0	R	15:0	Tone extractor COS result [15:0]
0xD8	FM_TSIN2	R	3:0	Tone extractor SIN result [35:32]
0xD9	FM_TSIN1	R	15:0	Tone extractor SIN result [31:16]
0xDA	FM_TSIN0	R	15:0	Tone extractor SIN result [15:0]
0xDB	FM_QSCL	R/W	15:0	Quantizer scaling
0xDC	FM_QCC1	R	5:0	Quantizer clip count [21:16]
0xDD	FM_QCC0	R	15:0	Quantizer clip count [15:0]
0xDE	FM_QBIT	R/W	2:0	Quantized number of bits – 1
0xDF	FM_QST	R/W	7:0	Quantizer output state to count
0xE0	FM_QSTC1	R	5:0	Quantizer state count [21:16]
0xE1	FM_QSTC0	R	15:0	Quantizer state count [15:0]
0xE2	FM_QPW2	R	3:0	Quantized power [35:32]
0xE3	FM_QPW1	R	15:0	Quantized power [31:16]
0xE4	FM_QPW0	R	15:0	Quantized power [15:0]
0xE5	FM_CCNT1	R	5:0	Clip count [21:16]
0xE6	FM_CCNT0	R	15:0	Clip count [15:0]
0xE7	FM_BLEV	R/W	15:0	RFI detection level
0xE8	FM_BLEN	R/W	15:0	RFI invalidation length
0xE9	FM_BCNT1	R	5:0	RFI detection count [21:16]
0xEA	FM_BCNT0	R	15:0	RFI detection count [15:0]

Address	Name	Dir	Bits	Description
0xEB	FM_IDATA	R	15:0	Primary input data at tick for testing
DELAY 2				**Insert output pipeline delay**
0xF0	D2_DSEL	R	2:0	Select wire for the three CRCs
0xF1	D2_CRC	R/W	11:0	4-bit CRCs for selected wire
0xF2	D2_ESEL	R/W	15:0	Select wire and enable for CRC error
0xF3	D2_ADLY	R/W	12:0	Output data delay for the A Outputs
0xF4	D2_BDLY	R/W	12:0	Output data delay for the B Outputs
0xF5	D2_CDLY	R/W	12:0	Output perr delay for the C Outputs
0xF6	D2_SEED	R/W	15:0	Seed for pseudo random bit generator.

Table 6-1 Register Map of Filter FPGA

6.2 Loading FIR Coefficients

All stages are capable of 512 taps.

6.2.1 Stage 1

The stage 1 lookup tables require 16 sets of 512 products, one for of the 16 possible values of the 4-bit data. If stage 1 is a 4-bit poly-phase filter, then the 512 lookup table values for each 4-bit address are written in the following order {0, 16, …, 496}, {1, 17, …, 497}, …, {15, 31, …, 511}. If stage 1 is an 8-bit poly-phase filter then there are 256 taps for each of the most and least significant parts. The 256 MS values are sent first and then the 256 LS values for each 4-bit address. The 256 lookup table values for each 4-bit address are written in the following order {0, 8, …, 248}, {1, 9, …, 249}, …, {7, 15, …, 255}. Because the input data is encoded as offset binary, the MS and LS LUTs are identical for the same 4-bit address (NRC-EVLA Memo# 010). If stage 1 is used as 16 32-tap filters for fractional bit correction, the products for –8/16 bit are loaded first and those for +7/16 bit are loaded last. Note that the files for each case have already been shuffled as above before the CMIB gets them; therefore, the CMIB just writes the LUT in the order in the file.

During the writing or reading of STAGE1 products, the appropriate control bit in CM_CTL (see Table 7-4) must be set to a 1. For normal operation the control bit must be set to 0. While loading the STAGE1 products, S1_CADD must be set to values 0, 1, … 15 and 512 values written for each address. After setting S1_CADD to a new value, there must be a wait of at least 32 data clocks before writing the LUT contents for that address. This wait allows the 32 taps in each FIR32 to fill with S1_CADD (which is treated like data). Due to the design of the FPGA, the LUTs should only be written when D1_DDEC = 0.

6.2.2 Stage 2

Stage 2 requires 32 sets of 16 coefficients. The coefficients are written out in a 'natural' order. When stage 2 is using less than 512 taps (256, 128 or 64) then each set is repeated multiple (2, 4 or 8) times. When the stage 2 mixer is used, then the coefficients are written differently. Half of the taps are multiplied against the cosine output of the mixer and the other half against the sine outputs. The coefficients are written interleaved – first the cosine coefficient and then the sine coefficient. In the coefficient file that the CMIB will get there are 512 lines of coefficients which includes any repetitions and interleaving described above. The procedure is to set S2_CADD = 0 and write the first 32 coefficients to S2_CVAL, set S2_CADD = 1 and write the next 32 coefficients, and so on until all 512 are written. To read back, check and re-write the coefficients, the same procedure is followed except that the CMIB reads each coefficient from S2_CVAL, checks the value and then writes the same (or new) coefficient back to S2_CVAL. Due to the design of the FPGA, the coefficients should only be written when S2_CDEC = 0.

6.2.3 Stage 3

Stage 3 requires 2 sets of 256 coefficients. The coefficients are written out in a 'natural' order. When stage 3 is using less than 512 taps (256, 128 or 64) then each set is repeated multiple (2, 4 or 8) times. In the coefficient file that he CMIB will get there are 512 lines of coefficients which includes any repetitions described above. The procedure is to set S3_CADD = 0 and write the first 2 coefficients to S3_CVAL, set S3_CADD = 1 and write the next 2 coefficients, and so on until all 512 are written. To read back, check and re-write the coefficients, the same procedure is followed except that the CMIB reads each coefficient from S3_CVAL, checks the value and then writes the same (or new) coefficient back to S3_CVAL.

6.2.4 Stage 4

Stage 4 requires one set of 512 coefficients. The coefficients are written out in a 'natural' order. Because of the slow input data rate, stage 4 can (and should) process 512 taps for all decimations. In the coefficient file that he CMIB will get there are 512 lines of coefficients. The procedure is to set S4_CADD = 0 and write the first coefficient to S4_CVAL, set S4_CADD = 1 and write the second coefficient, and so on until all 512 are written. To read back, check and re-write the coefficients, the same procedure is

followed except that the CMIB reads each coefficient from S4_CVAL, checks the value and then writes the same (or new) coefficient back to S4_CVAL.

6.3 Loading Mixer LUTs

There are two mixer LUTs, one for cosine and one for sine. Each LUT is loaded with 1024 values corresponding to one complete cycle of phase. The LUT addresses are set by writing to S2_MADD. The corresponding values are written to or read from S2_COS and S2_SIN. Address 0 corresponds to phase zero, 1 to phase 1/1024 cycles, and so on. Both the writes and reads can be done in any order.

6.4 Loading Tone Extractor LUTs

There is one tone extractor LUT which contains 256 16-bit cos/sin pairs (8 bits each) corresponding to one complete cycle of phase. The LUT address is set to zero by writing to FM_TADD. The corresponding value is written to or read from FM_TVAL. Address 0 corresponds to phase zero, 1 to phase 1/256 cycles, and so on. Both the writes and reads can be done in any order.

6.5 Filter Scaling etc.

Each filter stage is followed by a module which scales the data, stretches invalids and delays the tick, noise diode on/off indicator and phase error signals.

An invalid can be stretched from 0 to 1023 decimated clocks. Normally, this would be set to the number of taps divided by the decimation ratio to ensure that no valid outputs are produced while an invalid sample is in the filter taps.

The tick, noise diode on/off indicator and the phase error can be delayed up to 1023 decimated clocks to match the delay in the filter taps. Normally this would be set to the half the number of taps divided by the decimation ratio.

In the following, it is assumed that the coefficients of the filter are scaled so that the largest coefficient has the maximum value of nearly unity (0x7FFF or 0x7FF for stage 1). At this point, the samples are the output of an adder which produces a 40-bit result (the output of the stage 1 adder tree is scaled to comply with stages 2, 3 and 4). This number of bits results from a 16-bit by 16-bit multiply giving a 31 bit result (MS bit is not significant) plus 9 bits for growth (512 tap large signal test cases only).

For a noise dominated signal, a 16-bit input sample will grow by the square root of the equivalent number of number of input samples that make up the output samples. This number will be approximately the square root of the decimation ratio. To maintain the original scaling while decimating by 16 would require the scale be set to 512/4 = 128. If significant bits are lost in the re-quantization of the data to 16 bits, then the resultant output is set to 0x7FFF or 0x8001, depending on the sign bit; however, the valid indicator is not changed and the occurrence is not counted. This condition can be by caused by

incorrect scaling or a strong signal (natural or RFI). It is unlikely that a natural signal will be strong enough to cause clipping even at the narrowest bandwidths (while maintaining a suitable number of noise bits).

If the signal dominates, the growth is approximately equal to the decimation ratio. And the scaling would have to change accordingly.

6.6 Decimation Ratios

6.6.1 Delay1

The incoming data rate on the 64-bit data highway is normally 256 Mw/s (EVLA and e-MERLIN) but can be lower in factors of two down to 62.5 kw/s (VLBA). In the former case the decimation ratio would be 1 (CMIB sends 0 meaning no decimation). In the latter case the data word represents 16 sub-bands formed at the VLBA antenna and a decimation by 8 (CMIB sends 7) to 32 Mw/s would be used for the 16 MHz VLBA BBC bandwidth.

6.6.2 Stage 1

This stage would normally decimate 256 Mw/s to 256 Ms/s. This corresponds to a clock decimation by 1 (CMIB sends 0). Further decimation by factors of 2, 4, 8 or 16 are allowed. The VLBA would likely use stage 1 to do the fractional bit correction and a decimation ratio of 1 would be used in this case also. In the 8-bit EVLA or e-MERLIN case stage 1 can be used to produce a sub-band bandwidth of 64 MHz (the maximum capable of being transmitted to the Baseline Board for 7-bit correlation) which corresponds to a decimation of 2. It would also be possible to do this with a decimation of 1 in stage 1 and a decimation of 2 in stage 2.

6.6.3 Stages 2, 3 and 4

These last three stages decimate the incoming data rate by factors of 2, 4, 8 or 16. These stages time multiplex multipliers and therefore have a “calculation” rate associated with them. The calculation rate depends on the decimation rates for the previous stages but are specified specifically by the CMIB in order to make the hardware designer’s job simpler. In the normal case, where the incoming data is at the maximum for the stage, the calculation decimation is set to 1 (CMIB sends 0). Stage 4 uses a single multiplier which can run faster than is necessary for the usual 64, 128, 256 and 512 taps for decimation by 2, 4, 8 and 16. Therefore, it is possible to get more taps for the lower decimations subject to a maximum of 512. If the input data rate to stages 2, 3 and 4 is lower than the maximum for that stage then the number of taps that can be achieved increases to a maximum of 512.

The output data rate should not fall below 62.5 Ks/s for a tick period of 10 milliseconds because many of the functions in the FPGA use the tick for synchronization and need an integer number of samples between ticks.

6.7 Delay and Phase Models

6.7.1 Delay1 Model

The delay is represented by a 48-bit positive number. Bits [47:35] of the delay represent the 64-bit wide RAM address. Decimated 64-bit wide data is written into the RAM, so these bits represent different amounts of delay depending on the decimation. Normally, only the VLBI case (1 stream) would be decimated. The remaining bits just add more accuracy as one would expect. The delay rate always aligns with the lower 32 bits of the delay in significance. If the delay goes negative due to the delay rate, unpredictable things will happen. In the EVLA 4-bit 4-Gs/s case, bits [34:31] represent the sample inside the 64-bit word. In the EVLA 8-bit 2-Gs/s case, bits [34:32] represent the sample inside the 64-bit word. In the e-MERLIN 8-bit 2-band 1-Gs/s case, bits [34:33] represent the 'sample' (one from each band) inside the 64-bit word.

6.7.2 Mixer Phase Model

If the mixer is used, then a phase and constant phase rate must be supplied by the CMIB during each interrupt (S2_MFAZ and S2_MFAZR). If the mixer frequency is a multiple of 100 Hz, then S2_MFAZ can also be a constant (probably zero) and no updates are required. S2_MFAZ and S2_MFAZR depend on the frequency desired for the final output band. Appendix 8.1 shows a piece of (rather badly written) C code that derives these values. The equivalent of this code is incorporated in the Filter FPGA GUI and will have to appear in the CMIB. It describes a particular way at arriving at all the configuration registers from high level inputs but other assumptions could be made.

6.7.3 Tone Extractor Phase Model

If the tone extractor is used, then a phase and phase rate must be supplied by the CMIB during each interrupt (FM_TFAZ and FM_TFAZR). In general, FM_TFAZ and FM_TFAZR will have to updated by the CMIB every interrupt. The phase rate must be modified by the CMIB (see Appendix 8.1) to represent the mean frequency of the tone after the Delay Module and the sub-band filter delay.

6.8 Delay Error to Delay Phase Conversion

A new delay error is calculated and is converted to a phase error (PHASERR) which will be used by the Baseline Board to the loss of SNR due to a delay error (fractional sample correction). The delay error is in the range -0.5 to <+0.5 of a sample. The conversion factor is an unsigned fraction >= 0 and < 1.0 (0x0000 to <0x10000) and depends on where the sub-band comes from in the base band and is equal to $f_c/B*0x10000$ where f_c is the base band frequency of the centre of the sub-band and B is the base band width. For f_c =192 MHz and B = 2048 MHz the conversion factor is 0x1800.

6.9 Tick Delay

It is likely that Filter FPGA delay line will be operated with a constant delay offset approximately one half of the delay line length to facilitate positive and negative delays. This parameter delays the tick and hence the time at which the delay model is imposed to match the constant delay offset. It should be set to the constant delay offset in 64-bit words of actual data.

6.10 Invalid Stretch Length

If an invalid sample enters the tap line it can affect the result for the whole time it is in the tap line. The associated invalid signal can be stretched to cover this period of time and hence the output of that stage will be declared invalid for that period. The stretching is done at the output rate of the stage; therefore, it would normally equal the number of taps divided by the decimation. If the value is set to zero, all data will become valid. If set to one, the original validity will be passed on.

6.11 FIR Delay Compensation

Because the data is delayed by half the number of taps, the tick, noise diode on/off signal and the phase error must be delayed by the same amount to maintain timing alignment. The delay is performed at the output of the stages at the decimated clock rate; therefore, the value set would normally be the number of taps divided by two divided by the decimation factor.

6.12 Clip Counts

Which stage to count clipped samples can be chosen independently from the processed stage. This allows an inspection of the scaling of each of the four stages. If the clip count for a stage is too high then the scaling for that stage may be set too high.

6.13 RFI Mitigation

To deal with the RFI case, the blanking value is used. Each data sample is compared to the blanking level. If the data is grater than the blanking value or less than minus the blanking value, then the blanking counter is incremented. If the blanking length is greater than zero, the data is invalidated for at least the blanking length number of samples. If RFI is detected during the invalidated interval, the interval is extended by the blanking length. This allows RFI to be rapidly detected and the data invalidated for a programmable number of samples.

6.14 Re-quantizer Scaling

The re-quantizer scaling determines the number of bits in the output sample. The 16-bit input sample is multiplied by the scale factor (a 16-bit fraction) and rounded. The

number of clipped samples is counted and the post-quantizer power is measured. Too high a clip count or too low a power means the scale is set incorrectly.

6.15 Normalization

Many of the measured quantities in the Format block need to normalized by the associated valid count. The tone extractor measures the valid count that is to be used by the post-quantizer power, clip count and state count.

6.16 The Time Interval Counter

The time interval counter can be used to measure the time interval between the data tick and the system tick. These measurements are meant to aid in determining the wide band input delay, IO_IDLY. The top two bits of the MS part of the time interval can be set to measure the individual periods of the start and stop pulses.

6.17 The DELAY2 Delays

These delays are set by the CMIB to ensure that the samples from different filters are aligned in the destination FPGA (Output, Timing and VSI). Different delays are needed to account for the daisy chaining of the wideband data through the Filter FPGAs in a filter bank and for different sub-band bandwidths. If all the sub-band bandwidths are the 128 MHz, then the ticks for all sub-bands can be aligned with each other and with the header insertion (Output FPGA) and TIMECODE (Timing FPGA). This is done by setting the delays so that the appropriate Time Interval Counts are zero in the destination FPGA. If the bandwidths are not the same then it may not be possible to make the counters go to zero for the narrowest bandwidths. In this case, the delays must be set so that the Time Interval Count is a multiple of the sample period to ensure that the data header goes in correctly relative to the actual sample. In this case, the phase models will have to be altered (in real time) and the time stamps will have to be altered (in post-processing) to account for the error in timing.

6.18 Correcting Sampler DC Offset in Stage 1

If the samplers have a DC offset then peculiar things can happen to the slot 0 output. This can be alleviated by either changing the slot 0 filter from a low pass filter to a band pass filter or by removing the DC offset in stage 1 using the S1_IDC registers. The first method can result in a weird shape for the low frequency part of the slot 0 output. The second method needs an estimate of the DC offset which is provided by S1_ODC which is the sum of all the stage 1 valid output samples between the previous two ticks (before scaling). S1_ODC (48-bits) is normalized by S1_VDC (21-bits), the associated valid count. These registers can be read by the CMIB during the ISR and averaged for any length of time, normalized to the value for one sample, negated and fed back to stage 1 through the S1_IDC (32-bits). Note that S1_ODC is measured after DC removal; therefore, S1_ODC should becomes small and measures the *changes* to S1_IDC.

7 Detailed Register Description

The following is a detailed description of the Filter FPGA register set for each block of the design. The number of bits reflects the actual hardware register but on read the upper bits in the register are sign extended or zero; therefore, no masking by the CMIB is necessary when reading the full 16-bit register. Many of the registers are configuration information which is only changed for a new observation.

7.1 COMMON

The following registers are used by all blocks for status, configuration, control, error, testing and design identifier. The status, configuration, control and error registers are collections of single bits gathered together for simplicity.

Address	Name	Dir	Bits	Description
0x00	CM_STS	R/W	15:0	Status Register. The status register indicates various error conditions in the Filter FPGA as shown in Table 7-1. Each bit has a separate meaning. The status bits are saved and the working register cleared on the tick; therefore, the ISR should read the status register every interrupt to avoid the loss of an error indicator. The register should be all zero if there are no errors. To create errors for software test purposes, write to the register. The result will be the original bits XOR the written bits.
0x01	CM_CFG	R/W	15:0	Configuration Register. This register is composed of individual bits that are normally set during configuration and then left alone during operation. See Table 7-2.
0x02	CM_CTL	R/W	15:0	Control Register. This register is composed of individual bits that can be changed during operation. See Table 7-3.
0x03	CM_ERR	R/W	2:0	Error Register. This register should always be zero. If not, one of the errors in Table 7-4 has occurred since it was last cleared. To clear errors, write zero to the register. To create errors, commit an error or write a non-zero value to the register.
0x04	CM_DEF	R/W	15:0	Default Register. This register will contain the last attempt to write to a non-existent or read-only register and will be returned if an attempt is made to read from a non-existent register. It can also used as a test register.
0x05	CM_DID	R	15:0	Design identifier. The bits are divided into

				design, revision and version. Since the Filter FPGA is programmable, different designs are possible.
0x06	CM_TST0	R/W	7:0	Address of signal to connect to test port 0
0x07	CM_TST1	R/W	7:0	Address of signal to connect to test port 1
0x08	CM_TST2	R/W	7:0	Address of signal to connect to test port 2
0x09	CM_TST3	R/W	7:0	Address of signal to connect to test port 3

Table 7-1 Common Registers

Bit	Block	Meaning (1 = error)
0	INOUT	1 => SCLK not locked
1	INOUT	1 => STICK error (width)
2	DELAY1	1 => derr pattern error
3	DELAY1	1 => dclk phase error
4	DELAY1	1 => dfrm phase error
5	INOUT	1 => input SCLK is not toggling
6	INOUT	1 => DCM phase shift is complete (test only)
7	INOUT	1 => DCM phase shift overflow or underflow (test only)
8	INOUT	1 => STICK with edge + matches STICK with edge - (test only)
9	INOUT	1 => STICK with chosen edge leads STICK not chosen (test only)

Table 7-2 Status Bit Definitions for the Filter FPGA

Bit	Block	Meaning (default = 0)
0	DELAY1	1 => change sign of PHASERR, 0 => don't
1	INOUT	0 => select input port A, 1 => select input port B.
2	INOUT	1 => enable output port A, 0 => disable output port A.
3	INOUT	1 => enable output port B, 0 => disable output port B.
4	INOUT	Select edge for clocking in system tick
5	INOUT	Select edge for clocking in wideband data
6	STAGE1	0 => 4-bit mode, 1 => 8-bit mode.
7	DELAY1 STAGE1	0 => normal, 1 => STAGE1 is used for fractional delay. This bit is set to 1 for VLBI and STAGE1 loaded with 16 sets of products, each having a different fractional bit delay. The output phase error (due to fractional delay error) will be zero.
8	STAGE2	1 => use mixer, 0 => don't. When using the mixer, a different set of coefficients are needed and the phase model must be supplied.
9	STAGE2	1 => include phase error in mixer, 0 => don't. Normally, if the mixer is being used, the phase error will be incorporated, and the output phase error (due to fractional delay error) will be zero.
10	FORMAT	0 => disable sideband flipper, 1 => enable sideband flipper. Setting this bit to 1 flips the frequency order of this sub-band in the resulting spectrum. This may be used depending on whether the frequency slot is even or odd.
11	DELAY2	0 => 4-bit, 1 => 8-bit serialization. If this bit is set to 1 the 4-bit data to the Output FPGA will consist of two consecutive nibbles of the 8-bit sample from the delay line. The phase error output to the Timing FPGA is always in this mode.
12	DELAY2	0 => normal, 1 => use internally generated narrow band signals.
13	DELAY2	0 => normal, 1 => do not AC balance A output.
14	DELAY2	0 => normal, 1 => do not use bit 7 as valid on A output.

Table 7-3 Configuration Bit Definitions for the Filter FPGA

Bit	Block	Meaning
0	INOUT	1 => software reset, 0 => don't. Momentary reset.
1	INOUT	1 => disable clocking, 0 => don't. Use to reduce heat production.
2	DELAY1	0 => update delay model next tick (normal), 1 => don't (test).
3	STAGE1	1 => allow CMIB access to coefficients, 0 => don't. This bit is used to write and read STAGE1 product lookup tables.
4	STAGE1	0 => automatic, 1 => enable MCB setting of fractional bit (test).
5	STAGE2	1 => allow CMIB access to coefficients, 0 => don't. This bit is used to write and read STAGE2 coefficient lookup tables.
6	STAGE2	0 => update mixer phase model next tick, 1 => don't (test).
7	STAGE2	1 => reset data RAM addresses next tick, 0 => don't. This bit must set by the ISR at least once and must be cleared by the ISR if the tick interval is not an integral multiple of the length of a data RAM (1024 input samples). If cleared, continued proper operation should be checked by reading S2_WADD and comparing it to the expected value in the ISR.
8	STAGE3	1 => reset data RAM addresses next tick, 0 => don't. See above.
9	STAGE4	1 => reset data RAM addresses next tick, 0 => don't. See above
10	FORMAT	0 => update tone phase model next tick, 1 => don't (test).
11	FORMAT	0 => normal operation of sideband flipper synchronization (synchronized every other tick which is needed for the narrowest bandwidth only), 1 => synchronize on the next tick.
12	INOUT	1 => increment DCM phase, 0 => decrement DCM phase
13	INOUT	1 => shift DCM phase (must return to zero before another shift)
15	INOUT	1 => reset SCLK DCM (must be set back to 0).

Table 7-4 Control Bit Definitions for the Filter FPGA

Bit	Meaning
0	1=> attempt to write to a read-only register
1	1=> attempt to write to a non-existent register
2	1 => attempt to read from a non-existent register

Table 7-5 Error Bit Definitions for the Filter FPGA

ID Part	Bits	CM_DID
Design	8	[15:8]
Version	4	[7:4]
Revision	4	[3:0]

Table 7-6 Design ID Definition

Divider	Value	MHz
1	0	256
2	1	128
4	2	64
8	3	32
16	4	16
32	5	8
64	6	4
128	7	2
256	8	1
512	9	0.5
1024	10	0.25
2048	11	0.125
4096	12	0.0625
4096	13	0.0625
4096	14	0.0625
4096	15	0.0625

Table 7-7 Clock Divider Values

The range of dividers is limited to 4096 to ensure that there is an integral number of the lowest clock rate (0.0625 MHz) in a tick interval (10 milliseconds). If the tick (interrupt) interval were ever shortened to 5 or 2 milliseconds (the only other feasible values) in order to produce more accurate models for VLBI, then the divider range would be limited to 2048 and 4096 respectively. The upper limit of 10 milliseconds for the tick interval is set by the number of bits in the accumulators.

7.2 INOUT

Address	Name	Dir	Bits	Description
0x10	IO_ESEL	R/W	6:0	Select input data bit for creating a CRC error. An error is created when bit 6 = 1 and bits [5:0] = bits [5:0] of IO_DSEL
0x11	IO_DSEL	R/W	6:0	Select wire for CRC calculation. If bit 6 is set to 1 then other lines replace the LS data bit wires for CRC calculation only: wire[0]=>VALID, wire [1]=>NOISE, wire[2]=>DERR, wire [3]=>DFRM, wire[4]=>CLK
0x12	IO_CRC	R	3:0	4-bit CRC for selected bit. It is used to check correct data transmission between FPGAs. The CMIB sets the bit in the last FPGA to be the same as the bit in this FPGA and checks that the CRCs are the same. This register is read by the ISR and is the CRC for the last interrupt interval.
0x13	IO_SDLY	R/W	15:0	System Tick delay in 256 MHz clocks. This register can to offset the value of IO_TINT for ease of testing.
0x14	IO_TINT1	R/W	15:0	Bits 15:14 select interval to measure 00:dtick>stick, 01:dtick, 10:stick, 11:stick>dtick MS time interval in 256 MHz clocks in bits 5:0.
0x15	IO_TINT0	R	15:0	LS time interval count.
0x16	IO_SID	R	15:0	Station ID from MCB FPGA. Needs to be written by software during initial configuration to allow staggering of clock re-acquisition to prevent -48V current spikes.

Table 7-8 INOUT Registers

7.3 DELAY1

Address	Name	Dir	Bits	Description
0x20	D1_DDEC	R/W	3:0	Data input and output rate (Table 7-7). For EVLA and e-MERLIN, its value is set to zero. For VLBI, it may be set other values, depending on the recorded bandwidth. When set to higher values the effective delay range becomes longer.
0x21	D1_DMUX	R/W	3:0	Demux Factor–1 (0, 1, 3, 7 or 15). This register corresponds to Table 3-1. It determines the number of consecutive time samples in a 64-bit word.
0x22-0x24	D1_DLY	R/W	47:0	Delay in units of samples at the original sample rate. The MS 17 bits is the integer sample delay and the LS 31 bits is the fractional sample delay. The delay is always positive (unsigned) and is normally updated every interrupt by the CMIB. The new delay becomes active on the next tick.
0x25-0x26	D1_DLYR	R/W	31:0	Delay rate in units of fractional samples per model update interval. The delay rate is signed. The new delay rate becomes active on the next tick.
0x27	D1_DEPE	R/W	15:0	Conversion factor from delay error to phase error. This quantity is used to convert the fractional sample delay error to a phase error. The factor = fc/B*0x10000 where fc is the center frequency of the sub-band in the base band and B is the base band width.
0x28	D1_TDLY	R/W	12:0	Tick delay in 256 MHz clocks decimated by D1_DEC. This parameter is used to delay the tick when there is a constant delay offset.
0x29	D1_DERR	R	15:0	Delay error before model saved at tick.
0x2A	D1_PERR	R	15:4	Phase error after model saved at tick.
0x2B	D1_ODLY2	R	15:0	Output Delay [47:32] for testing

Address	Name	Dir	Bits	Description
0x2C	D1_ODLY1	R	15:0	Output Delay [31:16] for testing
0x2D	D1_ODLY0	R	15:0	Output Delay [15:0] for testing

Table 7-9 DELAY1 Registers

7.4 STAGE1

Address	Name	Dir	Bits	Description
0x40	S1_DDEC	R/W	3:0	Data output rate (Table 7-7).
0x41-0x44	S1_XBAR	R/W	63:0	Crossbar addresses (4 bits each)
0x45	S1_CADD	R/W	3:0	Address for writing product chain of 512
0x46	S1_CVAL	R/W	11:0	Product value (read looks at end of chain)
0x47	S1_VLEN	R/W	9:0	Invalid stretch count in output clocks
0x48	S1_FDLY	R/W	9:0	Filter delay in output clocks
0x49	S1_SCALE	R/W	15:0	Data scaling factor. (1 => no clip possible)
0x4A	S1_FBIT	R/W	3:0	Fractional bit select if enabled by control bit
0x4B	S1_IDC1	R/W	15:0	DC offset to stage 1 [31:16]
0x4C	S1_IDC0	R/W	15:0	DC offset to stage 1 [15:0]
0x4D	S1_ODC2	R/W	15:0	DC offset from stage 1 [47:32]
0x4E	S1_ODC1	R/W	15:0	DC offset from stage 1 [31:16]
0x4F	S1_ODC0	R/W	15:0	DC offset from stage 1 [15:0]
0x50	S1_VDC1	R/W	5:0	DC Valid count from stage 1 [21:16]
0x51	S1_VDC0	R/W	15:0	DC Valid count from stage 1 [15:0]

Table 7-10 STAGE1 Registers

7.5 STAGE2

Address	Name	Dir	Bits	Description
0x60	S2_DDEC	R/W	3:0	Data output rate (Table 7-7).
0x61	S2_CADD	R/W	3:0	Address for reading and writing coefficients
0x62	S2_CVAL	R/W	15:0	Coefficients (read looks at end of chain)
0x63	S2_VLEN	R/W	9:0	Invalid stretch count in output clocks
0x64	S2_FDLY	R/W	9:0	Filter delay in output clocks
0x65	S2_SCALE	R/W	15:0	Data scaling factor. (1 => no clip possible)
0x66	S2_MADD	R/W	9:0	Mixer LUT address
0x67	S2_MCOS	R/W	15:0	Mixer COS LUT value
0x68	S2_MSIN	R/W	15:0	Mixer SIN LUT value
0x69-0x6A	S2_MFAZ	R/W	15:0	Mixer phase
0x6B-0x6C	S2_MFAZR	R/W	15:0	Mixer phase rate
0x6D	S2_CDEC	R/W	3:0	Filter calculation rate (Table 7-7). Normally, the calculations are done at 256 MHz to get the maximum number of taps. The maximum number of taps possible is 512 so under some circumstances the calculations must be slowed down.
0x6E	S2_NTAP	R/W	8:0	Number of taps-1 (63, 127, 255, 511)
0x6F	S2_WADD	R	9:0	Data RAM write address last tick

Table 7-11 STAGE2 Registers

7.6 STAGE3

Address	Name	Dir	Bits	Description
0x80	S3_DDEC	R/W	3:0	Data output rate (Table 7-7).
0x81	S3_CADD	R/W	6:0	Address for reading and writing coefficients
0x82	S3_CVAL	R/W	15:0	Coefficient value
0x83	S3_VLEN	R/W	9:0	Invalid stretch count in output clocks
0x84	S3_FDLY	R/W	9:0	Filter delay in output clocks
0x85	S3_SCALE	R/W	15:0	Data scaling factor. (1 => no clip possible)
0x86	S3_CDEC	R/W	3:0	Filter calculation rate (Table 7-7). Normally, the calculations are done at 256 MHz to get the maximum number of taps. The maximum number of taps possible is 512 so under some circumstances the calculations must be slowed down.
0x87	S3_NTAP	R/W	8:0	Number of taps-1 (63, 127, 255, 511)
0x88	S3_WADD	R	9:0	Data RAM write address last tick

Table 7-12 STAGE3 Registers

7.7 STAGE4

Address	Name	Dir	Bits	Description
0xA0	S4_DDEC	R/W	3:0	Data output rate (Table 7-7). The maximum input rate to STAGE4 is 1 Ms/s and the maximum output rate is 0.5 Ms/s or s4_DDCE = 511.
0xA1	S4_CADD	R/W	8:0	Address for reading and writing coefficients. STAGE4 is simple in that one coefficient per address is written by the CMIB since there is only one multiplier-accumulator.
0xA2	S4_CVAL	R/W	15:0	Coefficient value. The lowest address coefficient is also the first coefficient that the data encounters. STAGE4 uses a single dual-port RAM for the coefficients and hence is random access.
0xA3	S4_VLEN	R/W	9:0	Invalid stretch count in output clocks. This register is used to prevent invalid data in the tap line from affecting downstream measurements by invalidating the filter output if there is any invalid data in the tap delays.
0xA4	S4_FDLY	R/W	9:0	Filter delay in output clocks. This register is used to delay the tick, noise and phase error to match the pipeline delay suffered by the data in the FIR taps.
0xA5	S4_SCALE	R/W	15:0	Data scaling factor. (1 => no clip possible)
0xA6	S4_CDEC	R/W	3:0	Filter calculation rate (Table 7-7). Normally, the calculations are done at 256 MHz to get the maximum number of taps. The maximum number of taps possible is 512 so under some circumstances the calculations must be slowed down. This is especially true of STAGE4 because it can produce a 512 tap filter for decimation by 2. For decimations by 4, 8 and 16 S4_CDE must be 1, 3 and 7 to limit the taps to 512.

Address	Name	Dir	Bits	Description
0xA7	S4_NTAP	R/W	8:0	Number of taps-1 (63, 127, 255, 511). The actual number of taps probably should be odd to produce a pipeline delay of an integral number of clocks. To do this, set one of the coefficients to zero but set the number of taps to one of the four allowed values.
0xA8	S4_WADD	R	9:0	Data RAM write address last tick. Used as a diagnostic.

Table 7-13 STAGE4 Registers

7.8 FORMAT

Address	Name	Dir	Bits	Description
0xC0	FM_DSEL	R/W	3:0	Select signals. Choose 1 of the 4 filter stages. Bits [1:0] select the primary data path. Bits [3:2] select the secondary data path. Normally, the primary select would remain as configured but the secondary select could be changed to monitor the output of any of the stages without affecting the output. Only FM_CCNT is fed by the secondary path.
0xC1-0xC2	FM_VCF	R	21:0	Valid count for noise diode off. Read by the ISR. Applies to the last tick interval. Counts the number of valid samples with the noise diode off in the primary data.
0xC3-0xC4	FM_VCN	R	21:0	Valid count for noise diode on. Read by the ISR. Applies to the last tick interval. Counts the number of valid samples with the noise diode on in the primary data.
0xC5-0xC8	FM_PWF	R	51:0	Power for noise diode off. Read by the ISR. Applies to the last tick interval. The sum of squares of the valid samples with the noise diode off in the primary data.
0xC9-0xCC	FM_PWN	R	51:0	Power for noise diode on. Read by the

Address	Name	Dir	Bits	Description
				ISR. Applies to the last tick interval. The sum of squares of the valid samples with the noise diode on in the primary data.
0xCD	FM_TADD	R/W	7:0	Tone LUT address.
0xCE	FM_TVAL	R/W	15:0	Tone COS [15:8] & SIN [7:0] LUT values. This table is loaded at configuration.
0xCF-0xD0	FM_TFAZ	R/W	31:0	Tone model phase. Set by the ISR. Becomes active at the next tick. Signed fraction in cycles.
0xD1-0xD2	FM_TFAZR	R/W	31:0	Tone model phase rate. Set by the ISR. Becomes active at the next tick. Signed fraction in cycles per update clock.
0xD2-0xD4	FM_TVC	R	21:0	Tone valid count. Read by the ISR. Applies to the last tick interval. The number of valid quantized samples.
0xD5-0xD7	FM_TCOS	R	35:0	Tone extractor COS result. Read by the ISR. Applies to the last tick interval. The result of multiplying and accumulating the product of the quantized data times the output of the COS table. The lookup table address given comes from the tone phase model.
0xD8-0xDA	FM_TSIN	R	35:0	Tone extractor SIN result. Read by the ISR. Applies to the last tick interval. The result of accumulating the product of the quantized data and the output of the SIN table. The lookup table address comes from the tone phase model.
0xDB	FM_QSCL	R/W	15:0	Quantizer scaling. The 16-bit output of the selected filter stage is multiplied by this number and rounded to the LS bit of the original sample. If the scaling is set to 0x7FFF (the maximum), then the result will be the original sample.

Address	Name	Dir	Bits	Description
0xDC-0xDD	FM_QCC	R	21:0	Quantizer clip count. Read by the ISR and applies to the last interrupt interval. If any of the upper bits that are lopped off (due to the value of FM_QBIT) are significant and the sample is valid, the count increments.
0xDE	FM_QBIT	R/W	2:0	Quantized number of bits – 1 (3, 4, 5, 6 or 7). This register determines at which bit the scaled sample is clipped.
0xDF	FM_QST	R/W	7:0	Quantizer output state to count. Set by the ISR. Becomes active at the next tick. For 4-bit samples, sign extend to 8 bits.
0xE0-0xE1	FM_QSTC	R	21:0	Quantizer state count. Read by the ISR. The number of valid samples with the FM_QST value in the last tick interval.
0xE2-0xE4	FM_QPW	R	35:0	Quantized power. Read by the ISR. Is the sum of squares of the quantized samples from the last tick interval.
0xE5-0xE6	FM_CCNT	R	21:0	Clip count. Read by the ISR. Applies to the previous tick interval. The number of clipped signals in the secondary data.
0xE7	FM_BLEV	R/W	15:0	RFI detection level. This number must be positive and in the range 0 to 32767. Any sample whose absolute value is greater than this level will be detected.
0xE8	FM_BLEN	R/W	15:0	RFI invalidation length in sample clocks. If RFI is detected the data can be invalidated for up to 65535 sample clocks. If a length of 0 is set, no invalidation takes place but the detections are still counted.
0xE9-0xEA	FM_BCNT	R	21:0	RFI detection count. Read by the ISR. Counts the number of valid RFI detections in the previous tick interval.
0xEB	FM_IDATA	R	15:0	Selected primary input data at tick.

Table 7-13 FORMAT Registers

7.9 DELAY2

Address	Name	Dir	Bits	Description
0xF0	D2_DSEL	R/W	2:0	Select data input bit for all three CRCs. If bit 2 = 1, CRC is calculated on SIND in place of DATA[0].
0xF1	D2_CRC	R/W	11:0	4-bit data CRCs for selected bit. [3:0] for ADATA, [7:4] for BDATA, [11:8] for CPERR. CRCs are used to check correct data transmission between FPGAs. The CMIB sets the output bit in the source FPGA to be the same as the input bit in the destination FPGA and checks that the CRCs are the same. This register is read by the ISR and is the CRC for the last tick interval.
0xF2	D2_ESEL	R/W	2:0	Select bit for CRC errors. If bit 2 is a 1 and bits [1:0] = D2_DSEL, then D2_CRC will be incorrect for A, B and C outputs.
0xF3	D2_ADLY	R/W	12:0	Delay for the ADATA + ATICK + ASIND bundle to the Output FPGA. This can be from 0 to 8190 256 MHz clocks to cover the range of bandwidths possible. The Time Interval Counter in the destination FPGA can be used to adjust this value.
0xF4	D2_BDLY	R/W	12:0	Delay for the BDATA + BTICK + BSIND bundle to the VSI FPGA (see above).
0xF5	D2_CDLY	R/W	12:0	Delay for the CPERR + CTICK + CSIND bundle to the Timing FPGA (see above).
0xF6	D2_SEED	R/W	15:0	Seed for pseudo random bit generator. Default value is 0x1357.

Table 7-15 DELAY2 Registers

8 Pinouts, Pin Locations and Programming Notes

8.1 Pinouts by signal name

Note: UNUSED and NC pins have been removed.

```
Release 10.1.03 - par K.39 (nt)
Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.

Wed Jul 21 14:36:58 2010

INPUT FILE:       filter_top_map.ncd
OUTPUT FILE:      filter_top_pad.txt
PART TYPE:        xc4vsx35
SPEED GRADE:      -10
PACKAGE:          ff668
```

Pin Number	Signal Name	Pin Usage	Pin Name	Direction	IO Standard	IO Bank
H8	adata<0>	IOB	IO_L21P_6	OUTPUT	HSTL_I_DCI	6
G8	adata<1>	IOB	IO_L10N_6	OUTPUT	HSTL_I_DCI	6
G9	adata<2>	IOB	IO_L9N_CC_LC_6	OUTPUT	HSTL_I_DCI	6
G10	adata<3>	IOB	IO_L9P_CC_LC_6	OUTPUT	HSTL_I_DCI	6
G7	asind	IOB	IO_L19N_6	OUTPUT	HSTL_I_DCI	6
H7	atick	IOB	IO_L21N_6	OUTPUT	HSTL_I_DCI	6
P8	bdata<0>	IOB	IO_L19P_10	OUTPUT	HSTL_III_DCI	10
P7	bdata<1>	IOB	IO_L21P_10	OUTPUT	HSTL_III_DCI	10
N8	bdata<2>	IOB	IO_L19N_10	OUTPUT	HSTL_III_DCI	10
N7	bdata<3>	IOB	IO_L17P_10	OUTPUT	HSTL_III_DCI	10
R8	bsind	IOB	IO_L25P_CC_LC_10	OUTPUT	HSTL_III_DCI	10
T8	btick	IOB	IO_L31P_10	OUTPUT	HSTL_III_DCI	10

N21	cperr<0>	IOB	IO_L17P_9	OUTPUT	HSTL_I_DCI	9
N20	cperr<1>	IOB	IO_L17N_9	OUTPUT	HSTL_I_DCI	9
N19	cperr<2>	IOB	IO_L9N_CC_LC_9	OUTPUT	HSTL_I_DCI	9
P20	cperr<3>	IOB	IO_L21P_9	OUTPUT	HSTL_I_DCI	9
M21	csind	IOB	IO_L13P_9	OUTPUT	HSTL_I_DCI	9
P19	ctick	IOB	IO_L21N_9	OUTPUT	HSTL_I_DCI	9
W1	idata_a<0>	IOB	IO_L1N_8	INPUT	LVCMOS15	8
V1	idata_a<1>	IOB	IO_L30N_10	INPUT	LVCMOS15	10
G2	idata_a<10>	IOB	IO_L30P_6	INPUT	LVCMOS15	6
Y2	idata_a<11>	IOB	IO_L6P_8	INPUT	LVCMOS15	8
W2	idata_a<12>	IOB	IO_L1P_8	INPUT	LVCMOS15	8
V2	idata_a<13>	IOB	IO_L30P_10	INPUT	LVCMOS15	10
U3	idata_a<14>	IOB	IO_L28P_10	INPUT	LVCMOS15	10
T3	idata_a<15>	IOB	IO_L26N_10	INPUT	LVCMOS15	10
R2	idata_a<16>	IOB	IO_L22P_10	INPUT	LVCMOS15	10
P3	idata_a<17>	IOB	IO_L16P_10	INPUT	LVCMOS15	10
N3	idata_a<18>	IOB	IO_L14P_10	INPUT	LVCMOS15	10
L3	idata_a<19>	IOB	IO_L8N_CC_LC_10	INPUT	LVCMOS15	10
U1	idata_a<2>	IOB	IO_L24P_CC_LC_10	INPUT	LVCMOS15	10
K3	idata_a<20>	IOB	IO_L7P_10	INPUT	LVCMOS15	10
H3	idata_a<21>	IOB	IO_L31N_6	INPUT	LVCMOS15	6
Y4	idata_a<22>	IOB	IO_L8P_CC_LC_8	INPUT	LVCMOS15	8
W4	idata_a<23>	IOB	IO_L4P_8	INPUT	LVCMOS15	8
V4	idata_a<24>	IOB	IO_L29P_10	INPUT	LVCMOS15	10
U4	idata_a<25>	IOB	IO_L29N_10	INPUT	LVCMOS15	10
T4	idata_a<26>	IOB	IO_L26P_10	INPUT	LVCMOS15	10
R4	idata_a<27>	IOB	IO_L20P_10	INPUT	LVCMOS15	10
P4	idata_a<28>	IOB	IO_L18N_10	INPUT	LVCMOS15	10
N4	idata_a<29>	IOB	IO_L15N_10	INPUT	LVCMOS15	10
T1	idata_a<3>	IOB	IO_L24N_CC_LC_10	INPUT	LVCMOS15	10
M4	idata_a<30>	IOB	IO_L12P_10	INPUT	LVCMOS15	10
L4	idata_a<31>	IOB	IO_L8P_CC_LC_10	INPUT	LVCMOS15	10
K4	idata_a<32>	IOB	IO_L6N_10	INPUT	LVCMOS15	10
J4	idata_a<33>	IOB	IO_L2N_10	INPUT	LVCMOS15	10

H4	idata_a<34>	IOB	IO_L31P_6	INPUT	LVCMOS15	6
G4	idata_a<35>	IOB	IO_L28P_6	INPUT	LVCMOS15	6
Y5	idata_a<36>	IOB	IO_L9N_CC_LC_8	INPUT	LVCMOS15	8
W5	idata_a<37>	IOB	IO_L5N_8	INPUT	LVCMOS15	8
V5	idata_a<38>	IOB	IO_L2N_8	INPUT	LVCMOS15	8
U5	idata_a<39>	IOB	IO_L32N_10	INPUT	LVCMOS15	10
P2	idata_a<4>	IOB	IO_L16N_10	INPUT	LVCMOS15	10
P5	idata_a<40>	IOB	IO_L18P_10	INPUT	LVCMOS15	10
N5	idata_a<41>	IOB	IO_L15P_10	INPUT	LVCMOS15	10
M5	idata_a<42>	IOB	IO_L13N_10	INPUT	LVCMOS15	10
K5	idata_a<43>	IOB	IO_L6P_10	INPUT	LVCMOS15	10
J5	idata_a<44>	IOB	IO_L2P_10	INPUT	LVCMOS15	10
H5	idata_a<45>	IOB	IO_L29N_6	INPUT	LVCMOS15	6
Y6	idata_a<46>	IOB	IO_L9P_CC_LC_8	INPUT	LVCMOS15	8
W6	idata_a<47>	IOB	IO_L5P_8	INPUT	LVCMOS15	8
V6	idata_a<48>	IOB	IO_L2P_8	INPUT	LVCMOS15	8
U6	idata_a<49>	IOB	IO_L32P_10	INPUT	LVCMOS15	10
N2	idata_a<5>	IOB	IO_L14N_10	INPUT	LVCMOS15	10
T6	idata_a<50>	IOB	IO_L27N_10	INPUT	LVCMOS15	10
P6	idata_a<51>	IOB	IO_L21N_10	INPUT	LVCMOS15	10
M6	idata_a<52>	IOB	IO_L13P_10	INPUT	LVCMOS15	10
L6	idata_a<53>	IOB	IO_L5N_10	INPUT	LVCMOS15	10
K6	idata_a<54>	IOB	IO_L3N_10	INPUT	LVCMOS15	10
J6	idata_a<55>	IOB	IO_L1N_10	INPUT	LVCMOS15	10
H6	idata_a<56>	IOB	IO_L29P_6	INPUT	LVCMOS15	6
AA10	idata_a<57>	IOB	IO_L27N_8	INPUT	LVCMOS15	8
Y8	idata_a<58>	IOB	IO_L26N_8	INPUT	LVCMOS15	8
W7	idata_a<59>	IOB	IO_L3P_8	INPUT	LVCMOS15	8
M2	idata_a<6>	IOB	IO_L11P_10	INPUT	LVCMOS15	10
V7	idata_a<60>	IOB	IO_L3N_8	INPUT	LVCMOS15	8
U7	idata_a<61>	IOB	IO_L31N_10	INPUT	LVCMOS15	10
T7	idata_a<62>	IOB	IO_L27P_10	INPUT	LVCMOS15	10
R7	idata_a<63>	IOB	IO_L25N_CC_LC_10	INPUT	LVCMOS15	10
K2	idata_a<7>	IOB	IO_L7N_10	INPUT	LVCMOS15	10

J2	idata_a<8>	IOB	IO_L4P_10	INPUT	LVCMOS15	10
H2	idata_a<9>	IOB	IO_L32P_6	INPUT	LVCMOS15	6
F7	idata_b<0>	IOB	IO_L19P_6	INPUT	LVCMOS15	6
F9	idata_b<1>	IOB	IO_L7N_6	INPUT	LVCMOS15	6
E3	idata_b<10>	IOB	IO_L24P_CC_LC_6	INPUT	LVCMOS15	6
E4	idata_b<11>	IOB	IO_L22N_6	INPUT	LVCMOS15	6
D6	idata_b<12>	IOB	IO_L17N_6	INPUT	LVCMOS15	6
F8	idata_b<13>	IOB	IO_L10P_6	INPUT	LVCMOS15	6
E17	idata_b<14>	IOB	IO_L5P_5	INPUT	LVCMOS15	5
D22	idata_b<15>	IOB	IO_L14P_5	INPUT	LVCMOS15	5
E23	idata_b<16>	IOB	IO_L18P_5	INPUT	LVCMOS15	5
E24	idata_b<17>	IOB	IO_L27N_5	INPUT	LVCMOS15	5
D2	idata_b<18>	IOB	IO_L25P_CC_LC_6	INPUT	LVCMOS15	6
D4	idata_b<19>	IOB	IO_L16N_6	INPUT	LVCMOS15	6
F10	idata_b<2>	IOB	IO_L5P_6	INPUT	LVCMOS15	6
C5	idata_b<20>	IOB	IO_L12P_6	INPUT	LVCMOS15	6
C6	idata_b<21>	IOB	IO_L8N_CC_LC_6	INPUT	LVCMOS15	6
E7	idata_b<22>	IOB	IO_L17P_6	INPUT	LVCMOS15	6
D20	idata_b<23>	IOB	IO_L4P_5	INPUT	LVCMOS15	5
C21	idata_b<24>	IOB	IO_L6P_5	INPUT	LVCMOS15	5
C22	idata_b<25>	IOB	IO_L14N_5	INPUT	LVCMOS15	5
D23	idata_b<26>	IOB	IO_L21P_5	INPUT	LVCMOS15	5
D24	idata_b<27>	IOB	IO_L16P_5	INPUT	LVCMOS15	5
D26	idata_b<28>	IOB	IO_L25P_CC_LC_5	INPUT	LVCMOS15	5
D3	idata_b<29>	IOB	IO_L22P_6	INPUT	LVCMOS15	6
F17	idata_b<3>	IOB	IO_L5N_5	INPUT	LVCMOS15	5
C4	idata_b<30>	IOB	IO_L16P_6	INPUT	LVCMOS15	6
B6	idata_b<31>	IOB	IO_L8P_CC_LC_6	INPUT	LVCMOS15	6
C7	idata_b<32>	IOB	IO_L11N_6	INPUT	LVCMOS15	6
D9	idata_b<33>	IOB	IO_L2P_6	INPUT	LVCMOS15	6
D18	idata_b<34>	IOB	IO_L7N_5	INPUT	LVCMOS15	5
C20	idata_b<35>	IOB	IO_L2P_5	INPUT	LVCMOS15	5
B21	idata_b<36>	IOB	IO_L6N_5	INPUT	LVCMOS15	5
C23	idata_b<37>	IOB	IO_L21N_5	INPUT	LVCMOS15	5

C24	idata_b<38>	IOB	IO_L16N_5	INPUT	LVCMOS15	5
B3	idata_b<39>	IOB	IO_L14N_6	INPUT	LVCMOS15	6
F18	idata_b<4>	IOB	IO_L11P_5	INPUT	LVCMOS15	5
B4	idata_b<40>	IOB	IO_L15N_6	INPUT	LVCMOS15	6
A5	idata_b<41>	IOB	IO_L6N_6	INPUT	LVCMOS15	6
A6	idata_b<42>	IOB	IO_L6P_6	INPUT	LVCMOS15	6
B7	idata_b<43>	IOB	IO_L11P_6	INPUT	LVCMOS15	6
C8	idata_b<44>	IOB	IO_L2N_6	INPUT	LVCMOS15	6
B9	idata_b<45>	IOB	IO_L13N_6	INPUT	LVCMOS15	6
B18	idata_b<46>	IOB	IO_L3P_5	INPUT	LVCMOS15	5
C19	idata_b<47>	IOB	IO_L7P_5	INPUT	LVCMOS15	5
B20	idata_b<48>	IOB	IO_L2N_5	INPUT	LVCMOS15	5
A21	idata_b<49>	IOB	IO_L15N_5	INPUT	LVCMOS15	5
F20	idata_b<5>	IOB	IO_L19P_5	INPUT	LVCMOS15	5
A22	idata_b<50>	IOB	IO_L15P_5	INPUT	LVCMOS15	5
B23	idata_b<51>	IOB	IO_L10N_5	INPUT	LVCMOS15	5
B24	idata_b<52>	IOB	IO_L10P_5	INPUT	LVCMOS15	5
C2	idata_b<53>	IOB	IO_L20P_6	INPUT	LVCMOS15	6
A3	idata_b<54>	IOB	IO_L14P_6	INPUT	LVCMOS15	6
A4	idata_b<55>	IOB	IO_L15P_6	INPUT	LVCMOS15	6
A7	idata_b<56>	IOB	IO_L3N_6	INPUT	LVCMOS15	6
A8	idata_b<57>	IOB	IO_L3P_6	INPUT	LVCMOS15	6
A9	idata_b<58>	IOB	IO_L13P_6	INPUT	LVCMOS15	6
A18	idata_b<59>	IOB	IO_L3N_5	INPUT	LVCMOS15	5
E21	idata_b<6>	IOB	IO_L12P_5	INPUT	LVCMOS15	5
A19	idata_b<60>	IOB	IO_L13N_5	INPUT	LVCMOS15	5
A20	idata_b<61>	IOB	IO_L13P_5	INPUT	LVCMOS15	5
A23	idata_b<62>	IOB	IO_L8N_CC_LC_5	INPUT	LVCMOS15	5
A24	idata_b<63>	IOB	IO_L8P_CC_LC_5	INPUT	LVCMOS15	5
E22	idata_b<7>	IOB	IO_L18N_5	INPUT	LVCMOS15	5
F23	idata_b<8>	IOB	IO_L24N_CC_LC_5	INPUT	LVCMOS15	5
F24	idata_b<9>	IOB	IO_L24P_CC_LC_5	INPUT	LVCMOS15	5
G1	idclk_a	IOB	IO_L30N_6	INPUT	LVCMOS15	6
F1	idclk_b	IOB	IO_L26N_6	INPUT	LVCMOS15	6

R1	iderr_a	IOB	IO_L22N_10	INPUT	LVCMOS15	10
E6	iderr_b	IOB	IO_L18P_6	INPUT	LVCMOS15	6
M1	idfrm_a	IOB	IO_L11N_10	INPUT	LVCMOS15	10
E5	idfrm_b	IOB	IO_L18N_6	INPUT	LVCMOS15	6
L1	inoise_a	IOB	IO_L10P_10	INPUT	LVCMOS15	10
F4	inoise_b	IOB	IO_L27P_6	INPUT	LVCMOS15	6
H1	itick_a	IOB	IO_L32N_6	INPUT	LVCMOS15	6
E2	itick_b	IOB	IO_L24N_CC_LC_6	INPUT	LVCMOS15	6
K1	ivalid_a	IOB	IO_L10N_10	INPUT	LVCMOS15	10
F3	ivalid_b	IOB	IO_L27N_6	INPUT	LVCMOS15	6
AA13	MCB_ADDR<0>	IOB	IO_L4N_D8_VREF_LC_2	INPUT	LVCMOS25	2
AB13	MCB_ADDR<1>	IOB	IO_L4P_D9_LC_2	INPUT	LVCMOS25	2
AA15	MCB_ADDR<2>	IOB	IO_L3N_D10_LC_2	INPUT	LVCMOS25	2
AA16	MCB_ADDR<3>	IOB	IO_L3P_D11_LC_2	INPUT	LVCMOS25	2
AC11	MCB_ADDR<4>	IOB	IO_L2N_D12_LC_2	INPUT	LVCMOS25	2
AC12	MCB_ADDR<5>	IOB	IO_L2P_D13_LC_2	INPUT	LVCMOS25	2
AB14	MCB_ADDR<6>	IOB	IO_L1N_D14_CC_LC_2	INPUT	LVCMOS25	2
AA14	MCB_ADDR<7>	IOB	IO_L1P_D15_CC_LC_2	INPUT	LVCMOS25	2
AE14	MCB_CLK	IOB	IO_L5P_GC_LC_4	INPUT	LVCMOS25	4
AE12	MCB_CS_N	IOB	IO_L1N_GC_LC_4	INPUT	LVCMOS25	4
D12	MCB_DATA<0>	IOB	IO_L8N_D16_CC_LC_1	BIDIR	LVCMOS25	1
E13	MCB_DATA<1>	IOB	IO_L8P_D17_CC_LC_1	BIDIR	LVCMOS25	1
F15	MCB_DATA<10>	IOB	IO_L3N_D26_LC_1	BIDIR	LVCMOS25	1
F16	MCB_DATA<11>	IOB	IO_L3P_D27_LC_1	BIDIR	LVCMOS25	1
F11	MCB_DATA<12>	IOB	IO_L2N_D28_LC_1	BIDIR	LVCMOS25	1
F12	MCB_DATA<13>	IOB	IO_L2P_D29_LC_1	BIDIR	LVCMOS25	1
F13	MCB_DATA<14>	IOB	IO_L1N_D30_LC_1	BIDIR	LVCMOS25	1
F14	MCB_DATA<15>	IOB	IO_L1P_D31_LC_1	BIDIR	LVCMOS25	1
C16	MCB_DATA<2>	IOB	IO_L7N_D18_LC_1	BIDIR	LVCMOS25	1
D16	MCB_DATA<3>	IOB	IO_L7P_D19_LC_1	BIDIR	LVCMOS25	1
D11	MCB_DATA<4>	IOB	IO_L6N_D20_LC_1	BIDIR	LVCMOS25	1
C11	MCB_DATA<5>	IOB	IO_L6P_D21_LC_1	BIDIR	LVCMOS25	1
E14	MCB_DATA<6>	IOB	IO_L5N_D22_LC_1	BIDIR	LVCMOS25	1
D15	MCB_DATA<7>	IOB	IO_L5P_D23_LC_1	BIDIR	LVCMOS25	1

D13	MCB_DATA<8>	IOB	IO_L4N_D24_VREF_LC_1	BIDIR	LVCMOS25	1
D14	MCB_DATA<9>	IOB	IO_L4P_D25_LC_1	BIDIR	LVCMOS25	1
AF12	MCB_RD	IOB	IO_L1P_GC_LC_4	INPUT	LVCMOS25	4
W20	odata_a<0>	IOB	IO_L5P_7	OUTPUT	LVCMOS15	7
V20	odata_a<1>	IOB	IO_L5N_7	OUTPUT	LVCMOS15	7
H22	odata_a<10>	IOB	IO_L26P_5	OUTPUT	LVCMOS15	5
Y20	odata_a<11>	IOB	IO_L20P_7	OUTPUT	LVCMOS15	7
W21	odata_a<12>	IOB	IO_L3P_7	OUTPUT	LVCMOS15	7
V21	odata_a<13>	IOB	IO_L1P_7	OUTPUT	LVCMOS15	7
U21	odata_a<14>	IOB	IO_L29N_9	OUTPUT	LVCMOS15	9
T21	odata_a<15>	IOB	IO_L30P_9	OUTPUT	LVCMOS15	9
R20	odata_a<16>	IOB	IO_L25P_CC_LC_9	OUTPUT	LVCMOS15	9
P23	odata_a<17>	IOB	IO_L19P_9	OUTPUT	LVCMOS15	9
N23	odata_a<18>	IOB	IO_L16P_9	OUTPUT	LVCMOS15	9
L23	odata_a<19>	IOB	IO_L10N_9	OUTPUT	LVCMOS15	9
U20	odata_a<2>	IOB	IO_L31P_9	OUTPUT	LVCMOS15	9
K23	odata_a<20>	IOB	IO_L7N_9	OUTPUT	LVCMOS15	9
H23	odata_a<21>	IOB	IO_L30N_5	OUTPUT	LVCMOS15	5
Y22	odata_a<22>	IOB	IO_L12P_7	OUTPUT	LVCMOS15	7
W22	odata_a<23>	IOB	IO_L3N_7	OUTPUT	LVCMOS15	7
V22	odata_a<24>	IOB	IO_L1N_7	OUTPUT	LVCMOS15	7
U22	odata_a<25>	IOB	IO_L29P_9	OUTPUT	LVCMOS15	9
T23	odata_a<26>	IOB	IO_L24N_CC_LC_9	OUTPUT	LVCMOS15	9
R23	odata_a<27>	IOB	IO_L22N_9	OUTPUT	LVCMOS15	9
P24	odata_a<28>	IOB	IO_L18N_9	OUTPUT	LVCMOS15	9
N24	odata_a<29>	IOB	IO_L15N_9	OUTPUT	LVCMOS15	9
T20	odata_a<3>	IOB	IO_L30N_9	OUTPUT	LVCMOS15	9
M23	odata_a<30>	IOB	IO_L14P_9	OUTPUT	LVCMOS15	9
L24	odata_a<31>	IOB	IO_L10P_9	OUTPUT	LVCMOS15	9
K24	odata_a<32>	IOB	IO_L7P_9	OUTPUT	LVCMOS15	9
J23	odata_a<33>	IOB	IO_L2P_9	OUTPUT	LVCMOS15	9
H24	odata_a<34>	IOB	IO_L30P_5	OUTPUT	LVCMOS15	5
G24	odata_a<35>	IOB	IO_L28P_5	OUTPUT	LVCMOS15	5
Y24	odata_a<36>	IOB	IO_L8N_CC_LC_7	OUTPUT	LVCMOS15	7

W23	odata_a<37>	IOB	IO_L4P_7	OUTPUT	LVCMOS15	7
V23	odata_a<38>	IOB	IO_L27N_9	OUTPUT	LVCMOS15	9
U23	odata_a<39>	IOB	IO_L27P_9	OUTPUT	LVCMOS15	9
P22	odata_a<4>	IOB	IO_L19N_9	OUTPUT	LVCMOS15	9
P25	odata_a<40>	IOB	IO_L18P_9	OUTPUT	LVCMOS15	9
N25	odata_a<41>	IOB	IO_L15P_9	OUTPUT	LVCMOS15	9
M25	odata_a<42>	IOB	IO_L11P_9	OUTPUT	LVCMOS15	9
K25	odata_a<43>	IOB	IO_L8N_CC_LC_9	OUTPUT	LVCMOS15	9
H25	odata_a<44>	IOB	IO_L32N_5	OUTPUT	LVCMOS15	5
G25	odata_a<45>	IOB	IO_L31N_5	OUTPUT	LVCMOS15	5
Y25	odata_a<46>	IOB	IO_L6P_7	OUTPUT	LVCMOS15	7
W25	odata_a<47>	IOB	IO_L2P_7	OUTPUT	LVCMOS15	7
V25	odata_a<48>	IOB	IO_L32N_9	OUTPUT	LVCMOS15	9
U25	odata_a<49>	IOB	IO_L28P_9	OUTPUT	LVCMOS15	9
N22	odata_a<5>	IOB	IO_L16N_9	OUTPUT	LVCMOS15	9
T24	odata_a<50>	IOB	IO_L24P_CC_LC_9	OUTPUT	LVCMOS15	9
R24	odata_a<51>	IOB	IO_L22P_9	OUTPUT	LVCMOS15	9
L26	odata_a<52>	IOB	IO_L12P_9	OUTPUT	LVCMOS15	9
K26	odata_a<53>	IOB	IO_L8P_CC_LC_9	OUTPUT	LVCMOS15	9
J26	odata_a<54>	IOB	IO_L4P_9	OUTPUT	LVCMOS15	9
H26	odata_a<55>	IOB	IO_L32P_5	OUTPUT	LVCMOS15	5
G26	odata_a<56>	IOB	IO_L31P_5	OUTPUT	LVCMOS15	5
AA26	odata_a<57>	IOB	IO_L10N_7	OUTPUT	LVCMOS15	7
Y26	odata_a<58>	IOB	IO_L6N_7	OUTPUT	LVCMOS15	7
W26	odata_a<59>	IOB	IO_L2N_7	OUTPUT	LVCMOS15	7
M22	odata_a<6>	IOB	IO_L14N_9	OUTPUT	LVCMOS15	9
V26	odata_a<60>	IOB	IO_L32P_9	OUTPUT	LVCMOS15	9
U26	odata_a<61>	IOB	IO_L26N_9	OUTPUT	LVCMOS15	9
T26	odata_a<62>	IOB	IO_L26P_9	OUTPUT	LVCMOS15	9
R26	odata_a<63>	IOB	IO_L20P_9	OUTPUT	LVCMOS15	9
L21	odata_a<7>	IOB	IO_L6P_9	OUTPUT	LVCMOS15	9
K22	odata_a<8>	IOB	IO_L3P_9	OUTPUT	LVCMOS15	9
J22	odata_a<9>	IOB	IO_L2N_9	OUTPUT	LVCMOS15	9
AF7	odata_b<0>	IOB	IO_L25N_CC_LC_8	OUTPUT	LVCMOS15	8

AF8	odata_b<1>	IOB	IO_L25P_CC_LC_8	OUTPUT	LVCMOS15	8
AE3	odata_b<10>	IOB	IO_L15N_8	OUTPUT	LVCMOS15	8
AE4	odata_b<11>	IOB	IO_L17N_8	OUTPUT	LVCMOS15	8
AE6	odata_b<12>	IOB	IO_L30P_8	OUTPUT	LVCMOS15	8
AE9	odata_b<13>	IOB	IO_L31N_8	OUTPUT	LVCMOS15	8
AE18	odata_b<14>	IOB	IO_L31N_SM2_7	OUTPUT	LVCMOS15	7
AE21	odata_b<15>	IOB	IO_L32P_SM1_7	OUTPUT	LVCMOS15	7
AE23	odata_b<16>	IOB	IO_L19N_7	OUTPUT	LVCMOS15	7
AE24	odata_b<17>	IOB	IO_L22N_7	OUTPUT	LVCMOS15	7
AD3	odata_b<18>	IOB	IO_L18P_8	OUTPUT	LVCMOS15	8
AD4	odata_b<19>	IOB	IO_L22N_8	OUTPUT	LVCMOS15	8
AF9	odata_b<2>	IOB	IO_L31P_8	OUTPUT	LVCMOS15	8
AD5	odata_b<20>	IOB	IO_L22P_8	OUTPUT	LVCMOS15	8
AD6	odata_b<21>	IOB	IO_L30N_8	OUTPUT	LVCMOS15	8
AD8	odata_b<22>	IOB	IO_L32P_8	OUTPUT	LVCMOS15	8
AD19	odata_b<23>	IOB	IO_L25P_CC_SM7_LC_7	OUTPUT	LVCMOS15	7
AD21	odata_b<24>	IOB	IO_L32N_SM1_7	OUTPUT	LVCMOS15	7
AD22	odata_b<25>	IOB	IO_L15P_7	OUTPUT	LVCMOS15	7
AD23	odata_b<26>	IOB	IO_L15N_7	OUTPUT	LVCMOS15	7
AD25	odata_b<27>	IOB	IO_L11P_7	OUTPUT	LVCMOS15	7
AD26	odata_b<28>	IOB	IO_L11N_7	OUTPUT	LVCMOS15	7
AC3	odata_b<29>	IOB	IO_L18N_8	OUTPUT	LVCMOS15	8
AF18	odata_b<3>	IOB	IO_L31P_SM2_7	OUTPUT	LVCMOS15	7
AC4	odata_b<30>	IOB	IO_L11P_8	OUTPUT	LVCMOS15	8
AC6	odata_b<31>	IOB	IO_L24P_CC_LC_8	OUTPUT	LVCMOS15	8
AC7	odata_b<32>	IOB	IO_L28P_8	OUTPUT	LVCMOS15	8
AC9	odata_b<33>	IOB	IO_L29P_8	OUTPUT	LVCMOS15	8
AC19	odata_b<34>	IOB	IO_L25N_CC_SM7_LC_7	OUTPUT	LVCMOS15	7
AC21	odata_b<35>	IOB	IO_L24P_CC_LC_7	OUTPUT	LVCMOS15	7
AC22	odata_b<36>	IOB	IO_L13P_7	OUTPUT	LVCMOS15	7
AC23	odata_b<37>	IOB	IO_L16P_7	OUTPUT	LVCMOS15	7
AC24	odata_b<38>	IOB	IO_L16N_7	OUTPUT	LVCMOS15	7
AB1	odata_b<39>	IOB	IO_L10P_8	OUTPUT	LVCMOS15	8
AF19	odata_b<4>	IOB	IO_L17P_7	OUTPUT	LVCMOS15	7

AB3	odata_b<40>	IOB	IO_L12P_8	OUTPUT	LVCMOS15	8
AB4	odata_b<41>	IOB	IO_L11N_8	OUTPUT	LVCMOS15	8
AB5	odata_b<42>	IOB	IO_L13N_8	OUTPUT	LVCMOS15	8
AB6	odata_b<43>	IOB	IO_L24N_CC_LC_8	OUTPUT	LVCMOS15	8
AB9	odata_b<44>	IOB	IO_L29N_8	OUTPUT	LVCMOS15	8
AB18	odata_b<45>	IOB	IO_L29N_SM4_7	OUTPUT	LVCMOS15	7
AB20	odata_b<46>	IOB	IO_L28P_7	OUTPUT	LVCMOS15	7
AB21	odata_b<47>	IOB	IO_L24N_CC_LC_7	OUTPUT	LVCMOS15	7
AB22	odata_b<48>	IOB	IO_L13N_7	OUTPUT	LVCMOS15	7
AB23	odata_b<49>	IOB	IO_L14P_7	OUTPUT	LVCMOS15	7
AF20	odata_b<5>	IOB	IO_L17N_7	OUTPUT	LVCMOS15	7
AB24	odata_b<50>	IOB	IO_L7P_7	OUTPUT	LVCMOS15	7
AB25	odata_b<51>	IOB	IO_L7N_7	OUTPUT	LVCMOS15	7
AB26	odata_b<52>	IOB	IO_L10P_7	OUTPUT	LVCMOS15	7
AA1	odata_b<53>	IOB	IO_L10N_8	OUTPUT	LVCMOS15	8
AA3	odata_b<54>	IOB	IO_L7N_8	OUTPUT	LVCMOS15	8
AA4	odata_b<55>	IOB	IO_L7P_8	OUTPUT	LVCMOS15	8
AA7	odata_b<56>	IOB	IO_L20P_8	OUTPUT	LVCMOS15	8
AA8	odata_b<57>	IOB	IO_L26P_8	OUTPUT	LVCMOS15	8
AA9	odata_b<58>	IOB	IO_L21P_8	OUTPUT	LVCMOS15	8
AA18	odata_b<59>	IOB	IO_L21P_7	OUTPUT	LVCMOS15	7
AF21	odata_b<6>	IOB	IO_L30P_SM3_7	OUTPUT	LVCMOS15	7
AA19	odata_b<60>	IOB	IO_L26P_SM6_7	OUTPUT	LVCMOS15	7
AA20	odata_b<61>	IOB	IO_L26N_SM6_7	OUTPUT	LVCMOS15	7
AA23	odata_b<62>	IOB	IO_L14N_7	OUTPUT	LVCMOS15	7
AA24	odata_b<63>	IOB	IO_L8P_CC_LC_7	OUTPUT	LVCMOS15	7
AF22	odata_b<7>	IOB	IO_L30N_SM3_7	OUTPUT	LVCMOS15	7
AF23	odata_b<8>	IOB	IO_L19P_7	OUTPUT	LVCMOS15	7
AF24	odata_b<9>	IOB	IO_L22P_7	OUTPUT	LVCMOS15	7
G20	odclk_a	IOB	IO_L22N_5	OUTPUT	LVCMOS15	5
AD1	odclk_b	IOB	IO_L16N_8	OUTPUT	LVCMOS15	8
R19	oderr_a	IOB	IO_L25N_CC_LC_9	OUTPUT	LVCMOS15	9
AF6	oderr_b	IOB	IO_L19P_8	OUTPUT	LVCMOS15	8
M20	odfrm_a	IOB	IO_L13N_9	OUTPUT	LVCMOS15	9

AF5	odfrm_b	IOB	IO_L19N_8	OUTPUT	LVCMOS15	8
L20	onoise_a	IOB	IO_L6N_9	OUTPUT	LVCMOS15	9
AF4	onoise_b	IOB	IO_L17P_8	OUTPUT	LVCMOS15	8
AF11	otest<0>	IOB	IO_L4P_GC_LC_4	OUTPUT	LVCMOS25	4
AC17	otest<1>	IOB	IO_L3N_GC_LC_4	OUTPUT	LVCMOS25	4
AB17	otest<2>	IOB	IO_L3P_GC_LC_4	OUTPUT	LVCMOS25	4
AB10	otest<3>	IOB	IO_L2N_GC_LC_4	OUTPUT	LVCMOS25	4
H20	otick_a	IOB	IO_L22P_5	OUTPUT	LVCMOS15	5
AD2	otick_b	IOB	IO_L16P_8	OUTPUT	LVCMOS15	8
K20	ovalid_a	IOB	IO_L5N_9	OUTPUT	LVCMOS15	9
AF3	ovalid_b	IOB	IO_L15P_8	OUTPUT	LVCMOS15	8
A10	reset_N	IOB	IO_L6P_GC_LC_3	INPUT	LVCMOS25	3
C14	sclk_N	LOWCAPIOB	IO_L3N_GC_LC_3	INPUT	LVDS_25	3
C15	sclk_P	LOWCAPIOB	IO_L3P_GC_LC_3	INPUT	LVDS_25	3
B14	stick_N	IOB	IO_L1N_GC_CC_LC_3	INPUT	LVDS_25	3
B15	stick_P	IOB	IO_L1P_GC_CC_LC_3	INPUT	LVDS_25	3

Table 8-1 Pinout by Signal Name

8.2 Pinouts by pin number

Note: UNUSED and NC pins have been removed.

```
Release 10.1.03 - par K.39 (nt)
Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.

Wed Jul 21 14:36:58 2010

INPUT FILE:       filter_top_map.ncd
OUTPUT FILE:      filter_top_pad.txt
PART TYPE:        xc4vsx35
SPEED GRADE:      -10
PACKAGE:          ff668
```

Pin Number	Signal Name	Pin Usage	Pin Name	Direction	IO Standard	IO Bank
A10	reset_N	IOB	IO_L6P_GC_LC_3	INPUT	LVCMOS25	3
A13			GND			
A14			GND			
A18	idata_b<59>	IOB	IO_L3N_5	INPUT	LVCMOS15	5
A19	idata_b<60>	IOB	IO_L13N_5	INPUT	LVCMOS15	5
A2			GND			
A20	idata_b<61>	IOB	IO_L13P_5	INPUT	LVCMOS15	5
A21	idata_b<49>	IOB	IO_L15N_5	INPUT	LVCMOS15	5
A22	idata_b<50>	IOB	IO_L15P_5	INPUT	LVCMOS15	5
A23	idata_b<62>	IOB	IO_L8N_CC_LC_5	INPUT	LVCMOS15	5
A24	idata_b<63>	IOB	IO_L8P_CC_LC_5	INPUT	LVCMOS15	5
A25			GND			
A3	idata_b<54>	IOB	IO_L14P_6	INPUT	LVCMOS15	6
A4	idata_b<55>	IOB	IO_L15P_6	INPUT	LVCMOS15	6

A5	idata_b<41>	IOB	IO_L6N_6	INPUT	LVCMOS15	6
A6	idata_b<42>	IOB	IO_L6P_6	INPUT	LVCMOS15	6
A7	idata_b<56>	IOB	IO_L3N_6	INPUT	LVCMOS15	6
A8	idata_b<57>	IOB	IO_L3P_6	INPUT	LVCMOS15	6
A9	idata_b<58>	IOB	IO_L13P_6	INPUT	LVCMOS15	6
AA1	odata_b<53>	IOB	IO_L10N_8	OUTPUT	LVCMOS15	8
AA10	idata_a<57>	IOB	IO_L27N_8	INPUT	LVCMOS15	8
AA13	MCB_ADDR<0>	IOB	IO_L4N_D8_VREF_LC_2	INPUT	LVCMOS25	2
AA14	MCB_ADDR<7>	IOB	IO_L1P_D15_CC_LC_2	INPUT	LVCMOS25	2
AA15	MCB_ADDR<2>	IOB	IO_L3N_D10_LC_2	INPUT	LVCMOS25	2
AA16	MCB_ADDR<3>	IOB	IO_L3P_D11_LC_2	INPUT	LVCMOS25	2
AA18	odata_b<59>	IOB	IO_L21P_7	OUTPUT	LVCMOS15	7
AA19	odata_b<60>	IOB	IO_L26P_SM6_7	OUTPUT	LVCMOS15	7
AA2			VCCO_8			8
AA20	odata_b<61>	IOB	IO_L26N_SM6_7	OUTPUT	LVCMOS15	7
AA21			GND			
AA22			VCCO_7			7
AA23	odata_b<62>	IOB	IO_L14N_7	OUTPUT	LVCMOS15	7
AA24	odata_b<63>	IOB	IO_L8P_CC_LC_7	OUTPUT	LVCMOS15	7
AA25			VCCO_7			7
AA26	odata_a<57>	IOB	IO_L10N_7	OUTPUT	LVCMOS15	7
AA3	odata_b<54>	IOB	IO_L7N_8	OUTPUT	LVCMOS15	8
AA4	odata_b<55>	IOB	IO_L7P_8	OUTPUT	LVCMOS15	8
AA5			VCCO_8			8
AA6			GND			
AA7	odata_b<56>	IOB	IO_L20P_8	OUTPUT	LVCMOS15	8
AA8	odata_b<57>	IOB	IO_L26P_8	OUTPUT	LVCMOS15	8
AA9	odata_b<58>	IOB	IO_L21P_8	OUTPUT	LVCMOS15	8
AB1	odata_b<39>	IOB	IO_L10P_8	OUTPUT	LVCMOS15	8
AB10	otest<3>	IOB	IO_L2N_GC_LC_4	OUTPUT	LVCMOS25	4
AB11			VCCO_2			2
AB12			GND			
AB13	MCB_ADDR<1>	IOB	IO_L4P_D9_LC_2	INPUT	LVCMOS25	2
AB14	MCB_ADDR<6>	IOB	IO_L1N_D14_CC_LC_2	INPUT	LVCMOS25	2

AB15			|GND				
AB16			|VCCO_2			|2
AB17	|otest<2>	|IOB	|IO_L3P_GC_LC_4	|OUTPUT	|LVCMOS25	|4
AB18	|odata_b<45>	|IOB	|IO_L29N_SM4_7	|OUTPUT	|LVCMOS15	|7
AB19			|VCCO_7			|7
AB20	|odata_b<46>	|IOB	|IO_L28P_7	|OUTPUT	|LVCMOS15	|7
AB21	|odata_b<47>	|IOB	|IO_L24N_CC_LC_7	|OUTPUT	|LVCMOS15	|7
AB22	|odata_b<48>	|IOB	|IO_L13N_7	|OUTPUT	|LVCMOS15	|7
AB23	|odata_b<49>	|IOB	|IO_L14P_7	|OUTPUT	|LVCMOS15	|7
AB24	|odata_b<50>	|IOB	|IO_L7P_7	|OUTPUT	|LVCMOS15	|7
AB25	|odata_b<51>	|IOB	|IO_L7N_7	|OUTPUT	|LVCMOS15	|7
AB26	|odata_b<52>	|IOB	|IO_L10P_7	|OUTPUT	|LVCMOS15	|7
AB3	|odata_b<40>	|IOB	|IO_L12P_8	|OUTPUT	|LVCMOS15	|8
AB4	|odata_b<41>	|IOB	|IO_L11N_8	|OUTPUT	|LVCMOS15	|8
AB5	|odata_b<42>	|IOB	|IO_L13N_8	|OUTPUT	|LVCMOS15	|8
AB6	|odata_b<43>	|IOB	|IO_L24N_CC_LC_8	|OUTPUT	|LVCMOS15	|8
AB8			|VCCO_8			|8
AB9	|odata_b<44>	|IOB	|IO_L29N_8	|OUTPUT	|LVCMOS15	|8
AC11	|MCB_ADDR<4>	|IOB	|IO_L2N_D12_LC_2	|INPUT	|LVCMOS25	|2
AC12	|MCB_ADDR<5>	|IOB	|IO_L2P_D13_LC_2	|INPUT	|LVCMOS25	|2
AC17	|otest<1>	|IOB	|IO_L3N_GC_LC_4	|OUTPUT	|LVCMOS25	|4
AC19	|odata_b<34>	|IOB	|IO_L25N_CC_SM7_LC_7	|OUTPUT	|LVCMOS15	|7
AC21	|odata_b<35>	|IOB	|IO_L24P_CC_LC_7	|OUTPUT	|LVCMOS15	|7
AC22	|odata_b<36>	|IOB	|IO_L13P_7	|OUTPUT	|LVCMOS15	|7
AC23	|odata_b<37>	|IOB	|IO_L16P_7	|OUTPUT	|LVCMOS15	|7
AC24	|odata_b<38>	|IOB	|IO_L16N_7	|OUTPUT	|LVCMOS15	|7
AC3	|odata_b<29>	|IOB	|IO_L18N_8	|OUTPUT	|LVCMOS15	|8
AC4	|odata_b<30>	|IOB	|IO_L11P_8	|OUTPUT	|LVCMOS15	|8
AC6	|odata_b<31>	|IOB	|IO_L24P_CC_LC_8	|OUTPUT	|LVCMOS15	|8
AC7	|odata_b<32>	|IOB	|IO_L28P_8	|OUTPUT	|LVCMOS15	|8
AC9	|odata_b<33>	|IOB	|IO_L29P_8	|OUTPUT	|LVCMOS15	|8
AD1	|odclk_b	|IOB	|IO_L16N_8	|OUTPUT	|LVCMOS15	|8
AD15			|VCCO_4			|4
AD18			|GND				

AD19	odata_b<23>	IOB	IO_L25P_CC_SM7_LC_7	OUTPUT	LVCMOS15	7
AD2	otick_b	IOB	IO_L16P_8	OUTPUT	LVCMOS15	8
AD21	odata_b<24>	IOB	IO_L32N_SM1_7	OUTPUT	LVCMOS15	7
AD22	odata_b<25>	IOB	IO_L15P_7	OUTPUT	LVCMOS15	7
AD23	odata_b<26>	IOB	IO_L15N_7	OUTPUT	LVCMOS15	7
AD24			GND			
AD25	odata_b<27>	IOB	IO_L11P_7	OUTPUT	LVCMOS15	7
AD26	odata_b<28>	IOB	IO_L11N_7	OUTPUT	LVCMOS15	7
AD3	odata_b<18>	IOB	IO_L18P_8	OUTPUT	LVCMOS15	8
AD4	odata_b<19>	IOB	IO_L22N_8	OUTPUT	LVCMOS15	8
AD5	odata_b<20>	IOB	IO_L22P_8	OUTPUT	LVCMOS15	8
AD6	odata_b<21>	IOB	IO_L30N_8	OUTPUT	LVCMOS15	8
AD8	odata_b<22>	IOB	IO_L32P_8	OUTPUT	LVCMOS15	8
AD9			GND			
AE1			GND			
AE11			VCCO_4			4
AE12	MCB_CS_N	IOB	IO_L1N_GC_LC_4	INPUT	LVCMOS25	4
AE14	MCB_CLK	IOB	IO_L5P_GC_LC_4	INPUT	LVCMOS25	4
AE15			VREFN_SM			
AE16			VREFP_SM			
AE17			AVSS_SM			
AE18	odata_b<14>	IOB	IO_L31N_SM2_7	OUTPUT	LVCMOS15	7
AE19			VCCO_7			7
AE2			GND			
AE21	odata_b<15>	IOB	IO_L32P_SM1_7	OUTPUT	LVCMOS15	7
AE22			VCCO_7			7
AE23	odata_b<16>	IOB	IO_L19N_7	OUTPUT	LVCMOS15	7
AE24	odata_b<17>	IOB	IO_L22N_7	OUTPUT	LVCMOS15	7
AE25			GND			
AE26			GND			
AE3	odata_b<10>	IOB	IO_L15N_8	OUTPUT	LVCMOS15	8
AE4	odata_b<11>	IOB	IO_L17N_8	OUTPUT	LVCMOS15	8
AE5			VCCO_8			8
AE6	odata_b<12>	IOB	IO_L30P_8	OUTPUT	LVCMOS15	8

AE8			VCCO_8			8	
AE9	odata_b<13>	IOB	IO_L31N_8	OUTPUT	LVCMOS15	8	
AF11	otest<0>	IOB	IO_L4P_GC_LC_4	OUTPUT	LVCMOS25	4	
AF12	MCB_RD	IOB	IO_L1P_GC_LC_4	INPUT	LVCMOS25	4	
AF13			GND				
AF14			GND				
AF17			AVDD_SM				
AF18	odata_b<3>	IOB	IO_L31P_SM2_7	OUTPUT	LVCMOS15	7	
AF19	odata_b<4>	IOB	IO_L17P_7	OUTPUT	LVCMOS15	7	
AF2			GND				
AF20	odata_b<5>	IOB	IO_L17N_7	OUTPUT	LVCMOS15	7	
AF21	odata_b<6>	IOB	IO_L30P_SM3_7	OUTPUT	LVCMOS15	7	
AF22	odata_b<7>	IOB	IO_L30N_SM3_7	OUTPUT	LVCMOS15	7	
AF23	odata_b<8>	IOB	IO_L19P_7	OUTPUT	LVCMOS15	7	
AF24	odata_b<9>	IOB	IO_L22P_7	OUTPUT	LVCMOS15	7	
AF25			GND				
AF3	ovalid_b	IOB	IO_L15P_8	OUTPUT	LVCMOS15	8	
AF4	onoise_b	IOB	IO_L17P_8	OUTPUT	LVCMOS15	8	
AF5	odfrm_b	IOB	IO_L19N_8	OUTPUT	LVCMOS15	8	
AF6	oderr_b	IOB	IO_L19P_8	OUTPUT	LVCMOS15	8	
AF7	odata_b<0>	IOB	IO_L25N_CC_LC_8	OUTPUT	LVCMOS15	8	
AF8	odata_b<1>	IOB	IO_L25P_CC_LC_8	OUTPUT	LVCMOS15	8	
AF9	odata_b<2>	IOB	IO_L31P_8	OUTPUT	LVCMOS15	8	
B1			GND				
B11			VCCO_3			3	
B14	stick_N	IOB	IO_L1N_GC_CC_LC_3	INPUT	LVDS_25	3	
B15	stick_P	IOB	IO_L1P_GC_CC_LC_3	INPUT	LVDS_25	3	
B16			VCCO_3			3	
B18	idata_b<46>	IOB	IO_L3P_5	INPUT	LVCMOS15	5	
B19			VCCO_5			5	
B2			GND				
B20	idata_b<48>	IOB	IO_L2N_5	INPUT	LVCMOS15	5	
B21	idata_b<36>	IOB	IO_L6N_5	INPUT	LVCMOS15	5	
B22			VCCO_5			5	

B23	idata_b<51>	IOB	IO_L10N_5	INPUT	LVCMOS15	5
B24	idata_b<52>	IOB	IO_L10P_5	INPUT	LVCMOS15	5
B25			GND			
B26			GND			
B3	idata_b<39>	IOB	IO_L14N_6	INPUT	LVCMOS15	6
B4	idata_b<40>	IOB	IO_L15N_6	INPUT	LVCMOS15	6
B5			VCCO_6			6
B6	idata_b<31>	IOB	IO_L8P_CC_LC_6	INPUT	LVCMOS15	6
B7	idata_b<43>	IOB	IO_L11P_6	INPUT	LVCMOS15	6
B8			VCCO_6			6
B9	idata_b<45>	IOB	IO_L13N_6	INPUT	LVCMOS15	6
C11	MCB_DATA<5>	IOB	IO_L6P_D21_LC_1	BIDIR	LVCMOS25	1
C14	sclk_N	LOWCAPIOB	IO_L3N_GC_LC_3	INPUT	LVDS_25	3
C15	sclk_P	LOWCAPIOB	IO_L3P_GC_LC_3	INPUT	LVDS_25	3
C16	MCB_DATA<2>	IOB	IO_L7N_D18_LC_1	BIDIR	LVCMOS25	1
C18			GND			
C19	idata_b<47>	IOB	IO_L7P_5	INPUT	LVCMOS15	5
C2	idata_b<53>	IOB	IO_L20P_6	INPUT	LVCMOS15	6
C20	idata_b<35>	IOB	IO_L2P_5	INPUT	LVCMOS15	5
C21	idata_b<24>	IOB	IO_L6P_5	INPUT	LVCMOS15	5
C22	idata_b<25>	IOB	IO_L14N_5	INPUT	LVCMOS15	5
C23	idata_b<37>	IOB	IO_L21N_5	INPUT	LVCMOS15	5
C24	idata_b<38>	IOB	IO_L16N_5	INPUT	LVCMOS15	5
C3			GND			
C4	idata_b<30>	IOB	IO_L16P_6	INPUT	LVCMOS15	6
C5	idata_b<20>	IOB	IO_L12P_6	INPUT	LVCMOS15	6
C6	idata_b<21>	IOB	IO_L8N_CC_LC_6	INPUT	LVCMOS15	6
C7	idata_b<32>	IOB	IO_L11N_6	INPUT	LVCMOS15	6
C8	idata_b<44>	IOB	IO_L2N_6	INPUT	LVCMOS15	6
C9			GND			
D11	MCB_DATA<4>	IOB	IO_L6N_D20_LC_1	BIDIR	LVCMOS25	1
D12	MCB_DATA<0>	IOB	IO_L8N_D16_CC_LC_1	BIDIR	LVCMOS25	1
D13	MCB_DATA<8>	IOB	IO_L4N_D24_VREF_LC_1	BIDIR	LVCMOS25	1
D14	MCB_DATA<9>	IOB	IO_L4P_D25_LC_1	BIDIR	LVCMOS25	1

D15	MCB_DATA<7>	IOB	IO_L5P_D23_LC_1	BIDIR	LVCMOS25	1
D16	MCB_DATA<3>	IOB	IO_L7P_D19_LC_1	BIDIR	LVCMOS25	1
D18	idata_b<34>	IOB	IO_L7N_5	INPUT	LVCMOS15	5
D2	idata_b<18>	IOB	IO_L25P_CC_LC_6	INPUT	LVCMOS15	6
D20	idata_b<23>	IOB	IO_L4P_5	INPUT	LVCMOS15	5
D22	idata_b<15>	IOB	IO_L14P_5	INPUT	LVCMOS15	5
D23	idata_b<26>	IOB	IO_L21P_5	INPUT	LVCMOS15	5
D24	idata_b<27>	IOB	IO_L16P_5	INPUT	LVCMOS15	5
D26	idata_b<28>	IOB	IO_L25P_CC_LC_5	INPUT	LVCMOS15	5
D3	idata_b<29>	IOB	IO_L22P_6	INPUT	LVCMOS15	6
D4	idata_b<19>	IOB	IO_L16N_6	INPUT	LVCMOS15	6
D6	idata_b<12>	IOB	IO_L17N_6	INPUT	LVCMOS15	6
D9	idata_b<33>	IOB	IO_L2P_6	INPUT	LVCMOS15	6
E11			VCCO_1			1
E12			GND			
E13	MCB_DATA<1>	IOB	IO_L8P_D17_CC_LC_1	BIDIR	LVCMOS25	1
E14	MCB_DATA<6>	IOB	IO_L5N_D22_LC_1	BIDIR	LVCMOS25	1
E15			GND			
E16			VCCO_1			1
E17	idata_b<14>	IOB	IO_L5P_5	INPUT	LVCMOS15	5
E19			VCCO_5			5
E2	itick_b	IOB	IO_L24N_CC_LC_6	INPUT	LVCMOS15	6
E21	idata_b<6>	IOB	IO_L12P_5	INPUT	LVCMOS15	5
E22	idata_b<7>	IOB	IO_L18N_5	INPUT	LVCMOS15	5
E23	idata_b<16>	IOB	IO_L18P_5	INPUT	LVCMOS15	5
E24	idata_b<17>	IOB	IO_L27N_5	INPUT	LVCMOS15	5
E3	idata_b<10>	IOB	IO_L24P_CC_LC_6	INPUT	LVCMOS15	6
E4	idata_b<11>	IOB	IO_L22N_6	INPUT	LVCMOS15	6
E5	idfrm_b	IOB	IO_L18N_6	INPUT	LVCMOS15	6
E6	iderr_b	IOB	IO_L18P_6	INPUT	LVCMOS15	6
E7	idata_b<22>	IOB	IO_L17P_6	INPUT	LVCMOS15	6
E8			VCCO_6			6
F1	idclk_b	IOB	IO_L26N_6	INPUT	LVCMOS15	6
F10	idata_b<2>	IOB	IO_L5P_6	INPUT	LVCMOS15	6

F11	MCB_DATA<12>	IOB	IO_L2N_D28_LC_1	BIDIR	LVCMOS25	1
F12	MCB_DATA<13>	IOB	IO_L2P_D29_LC_1	BIDIR	LVCMOS25	1
F13	MCB_DATA<14>	IOB	IO_L1N_D30_LC_1	BIDIR	LVCMOS25	1
F14	MCB_DATA<15>	IOB	IO_L1P_D31_LC_1	BIDIR	LVCMOS25	1
F15	MCB_DATA<10>	IOB	IO_L3N_D26_LC_1	BIDIR	LVCMOS25	1
F16	MCB_DATA<11>	IOB	IO_L3P_D27_LC_1	BIDIR	LVCMOS25	1
F17	idata_b<3>	IOB	IO_L5N_5	INPUT	LVCMOS15	5
F18	idata_b<4>	IOB	IO_L11P_5	INPUT	LVCMOS15	5
F2			VCCO_6			6
F20	idata_b<5>	IOB	IO_L19P_5	INPUT	LVCMOS15	5
F21			GND			
F22			VCCO_5			5
F23	idata_b<8>	IOB	IO_L24N_CC_LC_5	INPUT	LVCMOS15	5
F24	idata_b<9>	IOB	IO_L24P_CC_LC_5	INPUT	LVCMOS15	5
F25			VCCO_5			5
F3	ivalid_b	IOB	IO_L27N_6	INPUT	LVCMOS15	6
F4	inoise_b	IOB	IO_L27P_6	INPUT	LVCMOS15	6
F5			VCCO_6			6
F6			GND			
F7	idata_b<0>	IOB	IO_L19P_6	INPUT	LVCMOS15	6
F8	idata_b<13>	IOB	IO_L10P_6	INPUT	LVCMOS15	6
F9	idata_b<1>	IOB	IO_L7N_6	INPUT	LVCMOS15	6
G1	idclk_a	IOB	IO_L30N_6	INPUT	LVCMOS15	6
G10	adata<3>	IOB	IO_L9P_CC_LC_6	OUTPUT	HSTL_I_DCI	6
G11			CS_B_0			
G12			D_IN_0			
G13			TDN_0			
G14			CCLK_0			
G15			INIT_0			
G16			HSWAPEN_0			
G2	idata_a<10>	IOB	IO_L30P_6	INPUT	LVCMOS15	6
G20	odclk_a	IOB	IO_L22N_5	OUTPUT	LVCMOS15	5
G24	odata_a<35>	IOB	IO_L28P_5	OUTPUT	LVCMOS15	5
G25	odata_a<45>	IOB	IO_L31N_5	OUTPUT	LVCMOS15	5

G26	odata_a<56>	IOB	IO_L31P_5	OUTPUT	LVCMOS15	5
G4	idata_a<35>	IOB	IO_L28P_6	INPUT	LVCMOS15	6
G7	asind	IOB	IO_L19N_6	OUTPUT	HSTL_I_DCI	6
G8	adata<1>	IOB	IO_L10N_6	OUTPUT	HSTL_I_DCI	6
G9	adata<2>	IOB	IO_L9N_CC_LC_6	OUTPUT	HSTL_I_DCI	6
H1	itick_a	IOB	IO_L32N_6	INPUT	LVCMOS15	6
H10			VCCO_6			6
H11			VCCAUX			
H12			RDWR_B_0			
H13			TDP_0			
H14			DONE_0			
H15			PROGRAM_B_0			
H16			VCCAUX			
H17			VCCAUX			
H18			VCCO_5			5
H19			VCCO_5			5
H2	idata_a<9>	IOB	IO_L32P_6	INPUT	LVCMOS15	6
H20	otick_a	IOB	IO_L22P_5	OUTPUT	LVCMOS15	5
H22	odata_a<10>	IOB	IO_L26P_5	OUTPUT	LVCMOS15	5
H23	odata_a<21>	IOB	IO_L30N_5	OUTPUT	LVCMOS15	5
H24	odata_a<34>	IOB	IO_L30P_5	OUTPUT	LVCMOS15	5
H25	odata_a<44>	IOB	IO_L32N_5	OUTPUT	LVCMOS15	5
H26	odata_a<55>	IOB	IO_L32P_5	OUTPUT	LVCMOS15	5
H3	idata_a<21>	IOB	IO_L31N_6	INPUT	LVCMOS15	6
H4	idata_a<34>	IOB	IO_L31P_6	INPUT	LVCMOS15	6
H5	idata_a<45>	IOB	IO_L29N_6	INPUT	LVCMOS15	6
H6	idata_a<56>	IOB	IO_L29P_6	INPUT	LVCMOS15	6
H7	atick	IOB	IO_L21N_6	OUTPUT	HSTL_I_DCI	6
H8	adata<0>	IOB	IO_L21P_6	OUTPUT	HSTL_I_DCI	6
H9			VCCO_6			6
J10			VCCINT			
J11			VCCINT			
J12			VCCAUX			
J13			GND			

J14				GND			
J15				VCCO_0			0
J16				VCCINT			
J17				VCCINT			
J19				VCCO_5			5
J2	idata_a<8>	IOB		IO_L4P_10	INPUT	LVCMOS15	10
J22	odata_a<9>	IOB		IO_L2N_9	OUTPUT	LVCMOS15	9
J23	odata_a<33>	IOB		IO_L2P_9	OUTPUT	LVCMOS15	9
J24				GND			
J26	odata_a<54>	IOB		IO_L4P_9	OUTPUT	LVCMOS15	9
J3				GND			
J4	idata_a<33>	IOB		IO_L2N_10	INPUT	LVCMOS15	10
J5	idata_a<44>	IOB		IO_L2P_10	INPUT	LVCMOS15	10
J6	idata_a<55>	IOB		IO_L1N_10	INPUT	LVCMOS15	10
J8				VCCO_6			6
K1	ivalid_a	IOB		IO_L10N_10	INPUT	LVCMOS15	10
K10				VCCINT			
K11				GND			
K12				GND			
K13				GND			
K14				GND			
K15				GND			
K16				GND			
K17				VCCINT			
K18				VCCINT			
K19				VCCO_9			9
K2	idata_a<7>	IOB		IO_L7N_10	INPUT	LVCMOS15	10
K20	ovalid_a	IOB		IO_L5N_9	OUTPUT	LVCMOS15	9
K22	odata_a<8>	IOB		IO_L3P_9	OUTPUT	LVCMOS15	9
K23	odata_a<20>	IOB		IO_L7N_9	OUTPUT	LVCMOS15	9
K24	odata_a<32>	IOB		IO_L7P_9	OUTPUT	LVCMOS15	9
K25	odata_a<43>	IOB		IO_L8N_CC_LC_9	OUTPUT	LVCMOS15	9
K26	odata_a<53>	IOB		IO_L8P_CC_LC_9	OUTPUT	LVCMOS15	9
K3	idata_a<20>	IOB		IO_L7P_10	INPUT	LVCMOS15	10

K4	idata_a<32>	IOB	IO_L6N_10	INPUT	LVCMOS15	10
K5	idata_a<43>	IOB	IO_L6P_10	INPUT	LVCMOS15	10
K6	idata_a<54>	IOB	IO_L3N_10	INPUT	LVCMOS15	10
K8			VCCO_10			10
K9			VCCINT			
L1	inoise_a	IOB	IO_L10P_10	INPUT	LVCMOS15	10
L10			VCCINT			
L11			VCCINT			
L12			GND			
L13			GND			
L14			GND			
L15			GND			
L16			VCCINT			
L17			VCCINT			
L18			VCCINT			
L2			VCCO_10			10
L20	onoise_a	IOB	IO_L6N_9	OUTPUT	LVCMOS15	9
L21	odata_a<7>	IOB	IO_L6P_9	OUTPUT	LVCMOS15	9
L22			VCCO_9			9
L23	odata_a<19>	IOB	IO_L10N_9	OUTPUT	LVCMOS15	9
L24	odata_a<31>	IOB	IO_L10P_9	OUTPUT	LVCMOS15	9
L25			VCCO_9			9
L26	odata_a<52>	IOB	IO_L12P_9	OUTPUT	LVCMOS15	9
L3	idata_a<19>	IOB	IO_L8N_CC_LC_10	INPUT	LVCMOS15	10
L4	idata_a<31>	IOB	IO_L8P_CC_LC_10	INPUT	LVCMOS15	10
L5			VCCO_10			10
L6	idata_a<53>	IOB	IO_L5N_10	INPUT	LVCMOS15	10
L9			VCCINT			
M1	idfrm_a	IOB	IO_L11N_10	INPUT	LVCMOS15	10
M10			GND			
M11			GND			
M12			VCCINT			
M13			GND			
M14			GND			

M15				VCCINT			
M16				GND			
M17				GND			
M18				VCCO_9			9
M2	idata_a<6>	IOB	IO_L11P_10	INPUT	LVCMOS15	10
M20	odfrm_a	IOB	IO_L13N_9	OUTPUT	LVCMOS15	9
M21	csind	IOB	IO_L13P_9	OUTPUT	HSTL_I_DCI	9
M22	odata_a<6>	IOB	IO_L14N_9	OUTPUT	LVCMOS15	9
M23	odata_a<30>	IOB	IO_L14P_9	OUTPUT	LVCMOS15	9
M25	odata_a<42>	IOB	IO_L11P_9	OUTPUT	LVCMOS15	9
M4	idata_a<30>	IOB	IO_L12P_10	INPUT	LVCMOS15	10
M5	idata_a<42>	IOB	IO_L13N_10	INPUT	LVCMOS15	10
M6	idata_a<52>	IOB	IO_L13P_10	INPUT	LVCMOS15	10
M9			VCCAUX			
N1			VCCO_10			10
N10			GND			
N11			GND			
N12			GND			
N13			GND			
N14			GND			
N15			GND			
N16			GND			
N17			GND			
N18			VCCAUX			
N19	cperr<2>	IOB	IO_L9N_CC_LC_9	OUTPUT	HSTL_I_DCI	9
N2	idata_a<5>	IOB	IO_L14N_10	INPUT	LVCMOS15	10
N20	cperr<1>	IOB	IO_L17N_9	OUTPUT	HSTL_I_DCI	9
N21	cperr<0>	IOB	IO_L17P_9	OUTPUT	HSTL_I_DCI	9
N22	odata_a<5>	IOB	IO_L16N_9	OUTPUT	LVCMOS15	9
N23	odata_a<18>	IOB	IO_L16P_9	OUTPUT	LVCMOS15	9
N24	odata_a<29>	IOB	IO_L15N_9	OUTPUT	LVCMOS15	9
N25	odata_a<41>	IOB	IO_L15P_9	OUTPUT	LVCMOS15	9
N26			GND			
N3	idata_a<18>	IOB	IO_L14P_10	INPUT	LVCMOS15	10

N4	idata_a<29>	IOB	IO_L15N_10	INPUT	LVCMOS15	10
N5	idata_a<41>	IOB	IO_L15P_10	INPUT	LVCMOS15	10
N6			GND			
N7	bdata<3>	IOB	IO_L17P_10	OUTPUT	HSTL_III_DCI	10
N8	bdata<2>	IOB	IO_L19N_10	OUTPUT	HSTL_III_DCI	10
N9			VCCAUX			
P1			GND			
P10			GND			
P11			GND			
P12			GND			
P13			GND			
P14			GND			
P15			GND			
P16			GND			
P17			GND			
P18			VCCAUX			
P19	ctick	IOB	IO_L21N_9	OUTPUT	HSTL_I_DCI	9
P2	idata_a<4>	IOB	IO_L16N_10	INPUT	LVCMOS15	10
P20	cperr<3>	IOB	IO_L21P_9	OUTPUT	HSTL_I_DCI	9
P21			GND			
P22	odata_a<4>	IOB	IO_L19N_9	OUTPUT	LVCMOS15	9
P23	odata_a<17>	IOB	IO_L19P_9	OUTPUT	LVCMOS15	9
P24	odata_a<28>	IOB	IO_L18N_9	OUTPUT	LVCMOS15	9
P25	odata_a<40>	IOB	IO_L18P_9	OUTPUT	LVCMOS15	9
P26			VCCO_9			9
P3	idata_a<17>	IOB	IO_L16P_10	INPUT	LVCMOS15	10
P4	idata_a<28>	IOB	IO_L18N_10	INPUT	LVCMOS15	10
P5	idata_a<40>	IOB	IO_L18P_10	INPUT	LVCMOS15	10
P6	idata_a<51>	IOB	IO_L21N_10	INPUT	LVCMOS15	10
P7	bdata<1>	IOB	IO_L21P_10	OUTPUT	HSTL_III_DCI	10
P8	bdata<0>	IOB	IO_L19P_10	OUTPUT	HSTL_III_DCI	10
P9			VCCAUX			
R1	iderr_a	IOB	IO_L22N_10	INPUT	LVCMOS15	10
R10			GND			

R11			GND				
R12			VCCINT				
R13			GND				
R14			GND				
R15			VCCINT				
R16			GND				
R17			GND				
R18			VCCAUX				
R19	oderr_a	IOB	IO_L25N_CC_LC_9	OUTPUT	LVCMOS15	9	
R2	idata_a<16>	IOB	IO_L22P_10	INPUT	LVCMOS15	10	
R20	odata_a<16>	IOB	IO_L25P_CC_LC_9	OUTPUT	LVCMOS15	9	
R23	odata_a<27>	IOB	IO_L22N_9	OUTPUT	LVCMOS15	9	
R24	odata_a<51>	IOB	IO_L22P_9	OUTPUT	LVCMOS15	9	
R26	odata_a<63>	IOB	IO_L20P_9	OUTPUT	LVCMOS15	9	
R4	idata_a<27>	IOB	IO_L20P_10	INPUT	LVCMOS15	10	
R7	idata_a<63>	IOB	IO_L25N_CC_LC_10	INPUT	LVCMOS15	10	
R8	bsind	IOB	IO_L25P_CC_LC_10	OUTPUT	HSTL_III_DCI	10	
R9			VCCO_10			10	
T1	idata_a<3>	IOB	IO_L24N_CC_LC_10	INPUT	LVCMOS15	10	
T10			VCCINT				
T11			VCCINT				
T12			GND				
T13			GND				
T14			GND				
T15			GND				
T16			VCCINT				
T17			VCCINT				
T18			VCCINT				
T2			VCCO_10			10	
T20	odata_a<3>	IOB	IO_L30N_9	OUTPUT	LVCMOS15	9	
T21	odata_a<15>	IOB	IO_L30P_9	OUTPUT	LVCMOS15	9	
T22			VCCO_9			9	
T23	odata_a<26>	IOB	IO_L24N_CC_LC_9	OUTPUT	LVCMOS15	9	
T24	odata_a<50>	IOB	IO_L24P_CC_LC_9	OUTPUT	LVCMOS15	9	

T25			VCCO_9			9	
T26	odata_a<62>	IOB	IO_L26P_9	OUTPUT	LVCMOS15	9	
T3	idata_a<15>	IOB	IO_L26N_10	INPUT	LVCMOS15	10	
T4	idata_a<26>	IOB	IO_L26P_10	INPUT	LVCMOS15	10	
T5			VCCO_10			10	
T6	idata_a<50>	IOB	IO_L27N_10	INPUT	LVCMOS15	10	
T7	idata_a<62>	IOB	IO_L27P_10	INPUT	LVCMOS15	10	
T8	btick	IOB	IO_L31P_10	OUTPUT	HSTL_III_DCI	10	
T9			VCCINT				
U1	idata_a<2>	IOB	IO_L24P_CC_LC_10	INPUT	LVCMOS15	10	
U10			VCCINT				
U11			GND				
U12			GND				
U13			GND				
U14			GND				
U15			GND				
U16			GND				
U17			VCCINT				
U18			VCCINT				
U19			VCCO_9			9	
U20	odata_a<2>	IOB	IO_L31P_9	OUTPUT	LVCMOS15	9	
U21	odata_a<14>	IOB	IO_L29N_9	OUTPUT	LVCMOS15	9	
U22	odata_a<25>	IOB	IO_L29P_9	OUTPUT	LVCMOS15	9	
U23	odata_a<39>	IOB	IO_L27P_9	OUTPUT	LVCMOS15	9	
U25	odata_a<49>	IOB	IO_L28P_9	OUTPUT	LVCMOS15	9	
U26	odata_a<61>	IOB	IO_L26N_9	OUTPUT	LVCMOS15	9	
U3	idata_a<14>	IOB	IO_L28P_10	INPUT	LVCMOS15	10	
U4	idata_a<25>	IOB	IO_L29N_10	INPUT	LVCMOS15	10	
U5	idata_a<39>	IOB	IO_L32N_10	INPUT	LVCMOS15	10	
U6	idata_a<49>	IOB	IO_L32P_10	INPUT	LVCMOS15	10	
U7	idata_a<61>	IOB	IO_L31N_10	INPUT	LVCMOS15	10	
U8			VCCO_10			10	
U9			VCCINT				
V1	idata_a<1>	IOB	IO_L30N_10	INPUT	LVCMOS15	10	

V10			VCCINT			
V11			VCCINT			
V12			VCCO_0			0
V13			GND			
V14			GND			
V15			VCCAUX			
V16			VCCINT			
V17			VCCINT			
V19			VCCO_7			7
V2	idata_a<13>	IOB	IO_L30P_10	INPUT	LVCMOS15	10
V20	odata_a<1>	IOB	IO_L5N_7	OUTPUT	LVCMOS15	7
V21	odata_a<13>	IOB	IO_L1P_7	OUTPUT	LVCMOS15	7
V22	odata_a<24>	IOB	IO_L1N_7	OUTPUT	LVCMOS15	7
V23	odata_a<38>	IOB	IO_L27N_9	OUTPUT	LVCMOS15	9
V24			GND			
V25	odata_a<48>	IOB	IO_L32N_9	OUTPUT	LVCMOS15	9
V26	odata_a<60>	IOB	IO_L32P_9	OUTPUT	LVCMOS15	9
V3			GND			
V4	idata_a<24>	IOB	IO_L29P_10	INPUT	LVCMOS15	10
V5	idata_a<38>	IOB	IO_L2N_8	INPUT	LVCMOS15	8
V6	idata_a<48>	IOB	IO_L2P_8	INPUT	LVCMOS15	8
V7	idata_a<60>	IOB	IO_L3N_8	INPUT	LVCMOS15	8
V8			VCCO_8			8
W1	idata_a<0>	IOB	IO_L1N_8	INPUT	LVCMOS15	8
W10			VCCAUX			
W11			VCCAUX			
W12			TCK_0			
W13			PWRDWN_B_0			
W14			M2_0			
W15			M0_0			
W16			VCCAUX			
W17			VCCO_7			7
W18			VCCO_7			7
W2	idata_a<12>	IOB	IO_L1P_8	INPUT	LVCMOS15	8

W20	odata_a<0>	IOB	IO_L5P_7	OUTPUT	LVCMOS15	7
W21	odata_a<12>	IOB	IO_L3P_7	OUTPUT	LVCMOS15	7
W22	odata_a<23>	IOB	IO_L3N_7	OUTPUT	LVCMOS15	7
W23	odata_a<37>	IOB	IO_L4P_7	OUTPUT	LVCMOS15	7
W25	odata_a<47>	IOB	IO_L2P_7	OUTPUT	LVCMOS15	7
W26	odata_a<59>	IOB	IO_L2N_7	OUTPUT	LVCMOS15	7
W4	idata_a<23>	IOB	IO_L4P_8	INPUT	LVCMOS15	8
W5	idata_a<37>	IOB	IO_L5N_8	INPUT	LVCMOS15	8
W6	idata_a<47>	IOB	IO_L5P_8	INPUT	LVCMOS15	8
W7	idata_a<59>	IOB	IO_L3P_8	INPUT	LVCMOS15	8
W8			VCCO_8			8
W9			VCCO_8			8
Y11			TMS_0			
Y12			TDI_0			
Y13			TDO_0			
Y14			DOUT_BUSY_0			
Y15			M1_0			
Y16			VBATT_0			
Y2	idata_a<11>	IOB	IO_L6P_8	INPUT	LVCMOS15	8
Y20	odata_a<11>	IOB	IO_L20P_7	OUTPUT	LVCMOS15	7
Y22	odata_a<22>	IOB	IO_L12P_7	OUTPUT	LVCMOS15	7
Y24	odata_a<36>	IOB	IO_L8N_CC_LC_7	OUTPUT	LVCMOS15	7
Y25	odata_a<46>	IOB	IO_L6P_7	OUTPUT	LVCMOS15	7
Y26	odata_a<58>	IOB	IO_L6N_7	OUTPUT	LVCMOS15	7
Y4	idata_a<22>	IOB	IO_L8P_CC_LC_8	INPUT	LVCMOS15	8
Y5	idata_a<36>	IOB	IO_L9N_CC_LC_8	INPUT	LVCMOS15	8
Y6	idata_a<46>	IOB	IO_L9P_CC_LC_8	INPUT	LVCMOS15	8
Y8	idata_a<58>	IOB	IO_L26N_8	INPUT	LVCMOS15	8

Table 8-2 Pinout by Pin Number

8.3 Xilinx XC4VSX35-10FF668-CS2 Pin Layout

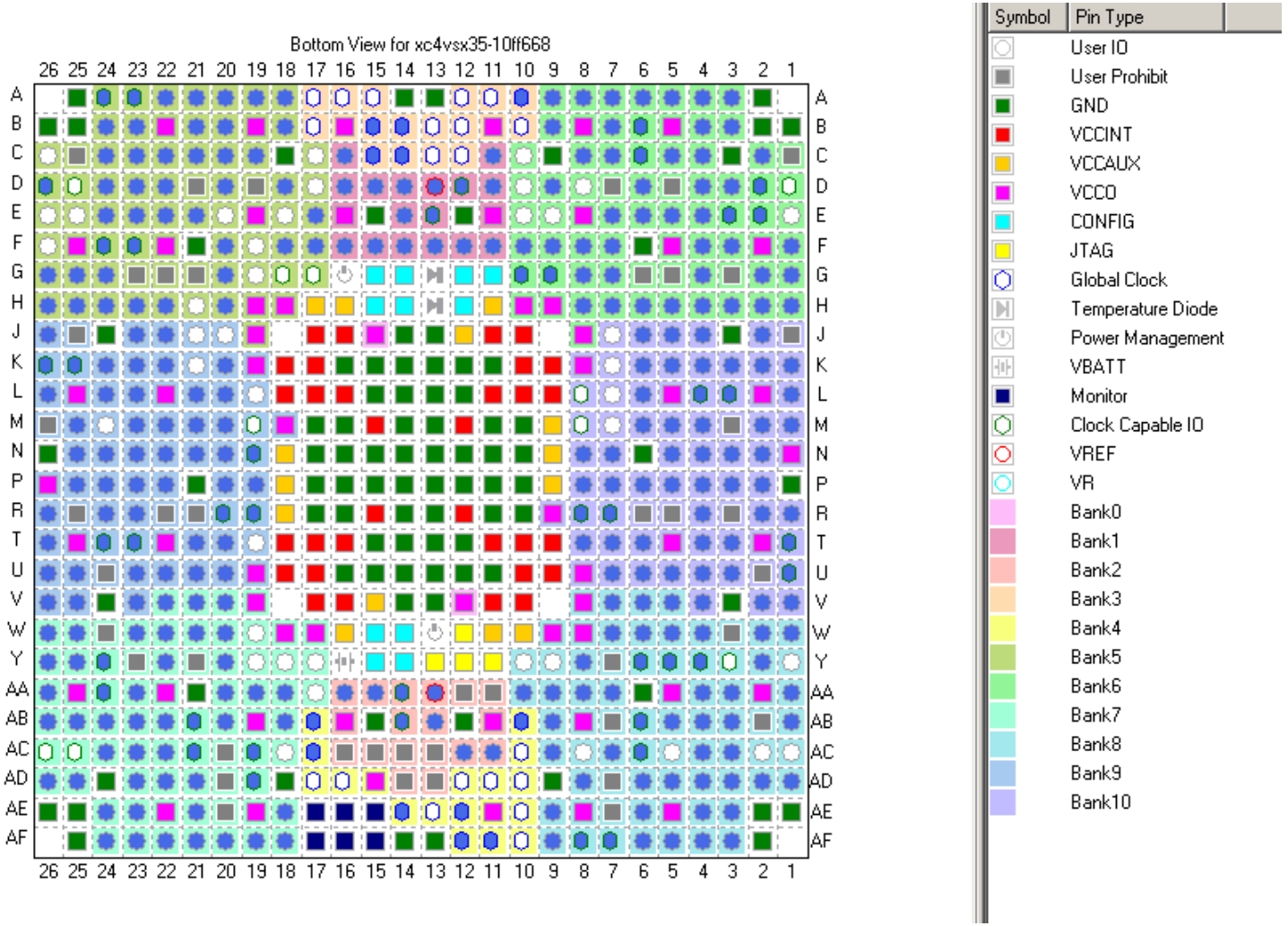

Figure 8-1 Pin Location

8.4 Programming Notes

All FPGAs on the Station Board are programmed through their 8-bit wide configuration port. The Station Board CMIB software requires the Binary (.bin) output file to program the Xilinx FPGAs. This is set by selecting “Properties…” from the “Process” pull-down menu in the Xilinx ISE software. Select the “General Options” and check “Create Binary Configuration File”.

9 References

Brent Carlson, “Refined EVLA WIDAR Correlator Architecture”, NRC-EVLA Memo# 014, October 2, 2001.

Brent Carlson, “Detailed Technical Points”, EVLA Correlator Conceptual Design Review, November 2, 2001.

Brent Carlson, “A Closer Look at 2-stage Digital Filtering in the Proposed WIDAR Correlator for the EVLA” NRC-EVLA Memo# 003, June 29, 2000.

Brent Carlson, “Requirements for 8-bit Processing in the Proposed WIDAR Correlator for the EVLA”, NRC-EVLA Memo# 010, January 29, 2001.

Brent Carlson, “HM Gbps Cable Signaling Specification”, NRC-EVLA Document A25022N0041.

10 Index