REQUIREMENTS AND FUNCTIONAL SPECIFICATION

Baseline Board RXP (Re-timing, X-bar, and Phasing) FPGA

RFS Document: **A25093N0000**

Revision: 2.1f

B. Carlson, June 7, 2012

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

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

Table of Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

LVDS – Low voltage differential signaling. Nominally the receiver requires 200 mV differential inputs minimum, and the output is ~800 mV differential.

RXP – Refers to this design. Acronym for "Re-timing, X-bar, and Phasing".

DDR – Double Data Rate. Generally this refers to data sampled on both the rising and falling edges of the associated clock. So, with a 256 MHz clock, a 512 Mbps data rate is possible.

VDIF – VLBI Data Interchange. This defines the payload of a packet containing data for a VLBI recorder/processing system.

MCB – Monitor/Control Bus. This is a simple synchronous read/write bus interface, which is CPU implementation independent.

ERNI connector – 2.0 mm "Hardmetric" connector. In this design, it is a 36-row, 8 signal pins per row connector. On the Baseline Board there are 2 such connectors, the X-ERNI connector feeds the RXPs.

SDD – Sample rate Dependent Delay.

Definitions

BB0, **BB1** – These refer to sampled data streams contained within a 4-pair HM Gbps 2 mm hardmetric wafer that plugs into a row of the Baseline Board input connector. BB0 is the innermost LVDS pair, and BB1 is the outermost LVDS pair (refer to ICD Document A25022N0040).

Upper RXP – The FPGA described in this design, fed by the upper 16 wafers (0-15) into the X-ERNI connector, and feeding X/Y Recirc FPGAs 0-3. In testmode, this is the master signal reference generator. Refer to Figure 3-1.

Lower RXP – The FPGA described in this design, fed by the lower 16 wafers (16-31) into the X-ERNI connector, and feeding X/Y Recirc FPGAs 4-7.

wafer – Refers to a 4-pair input or output of the 8-row (+ 2 GND) ERNI Hardmetric 2.0 mm connector. In this design, each "wafer" contains a 128 MHz clock, a CTRL signal, and BB0 and BB1 signals.

HM Gbps protocol – Refers to the protocol of the signaling appearing on the wafer signals CTRL, BB0, and BB1. The CTRL line contains the TIMECODE, DUMPTRIG, PHASEMOD, PHASERR, COMMAND, and STATUS signals.

1 Revision History

Revision	Date	Changes/Notes	Author
DRAFT	Oct. 4, 2007	Initial DRAFT release for review	B. Carlson
V1.0	Nov. 16, 2007	Add test mode TM bit to the MCSR register; some clean-up from comments, but otherwise no major changes.	B. Carlson
V1.1	Dec. 4, 2007	Add the TVT bit to the MCSR.	B. Carlson
V1.2	Sep. 19, 2008	Add the TICKSYNCERR registers for wafer tick alignment error detection. Modify the CCACR register to allow phased data to go to multiple columns/rows of the Correlator Chip array.	B. Carlson
2.0	Dec. 16, 2009	Completely re-vamp phasing control registers to reflect the implemented design. Many simplifications; output is now VDIF Version A.5.8, and compatible with GigE FPGA V2. TIMING CLOSURE HAS NOT YET BEEN ACHIEVED; SOME FUNCTIONALITY MAY BE REDUCED OR DELETED, AS NOTED.	B. Carlson
2.1	Jan. 30, 2010	Add XBAR Board Control/Status registers. PMCR/ICM bit meaning change. Change some phasing reset defaults. Implementation uses either a 63-tap or 127-tap Hilbert FIR filter with no s/w implications. Add PMM bit to the PCONTROL reg. Add appendix with chip interface details.	B. Carlson
2.1a	Feb. 1, 2010	Fix error in section 9.5; FPGAs are prg'd via the PCMC FPGA.	B. Carlson
2.1b	Mar. 2, 2010	Correct and update Appendix I with VDIF V1.0 ratified specification.	B. Carlson
2.1c	Apr. 15, 2010	Add explanatory notes to the 2-bit and 4-bit power measurement registers (p. 53)	B. Carlson
2.1d	June 2, 2011	Update Figure 6-1 for 127-tap FIR, and rounding for no output DC bias. Update Table 9-3 and Table 9-8 indicating acceptable SPARE pad data directions.	B. Carlson

2.1e	Nov. 30, 2011	Fix input row and wafer numbering faux-pas with Tables 9-8 and 9-9; Lower RXP DDR outputs source from wafers 16-31, and Lower RXP DDR inputs source from wafers 0-15.	B. Carlson
2.1f	June 7, 2012	Add detailed information on page 14 regarding sample rate and frame size VDIF packet delays to GigE FPGA output.	B. Carlson

2 Introduction

This document describes detailed requirements and design concepts for the RXP (Re-timing, X-bar, and Phasing) FPGA for the Baseline Board. There are two of these FPGAs on each Baseline Board, the UPPER RXP and the LOWER RXP, and they facilitate re-timing and distribution of signals to the row and column of correlator chips, include a X-bar (cross-bar) switch to allow full antenna routing and phasing capability, and include phasing functions for 32 antennas/streams of data.

The logic described in this RFS is implemented in an Altera EP2S60F672C4 FPGA. The design and functionality of each RXP FPGA is the same, although for board routing considerations, each one has a slightly different physical device pin assignment.

3 Context

RXP FPGAs reside on the Baseline Board near the bottom "X" input connector. Signals entering this connector [1] are received, synchronized, cross-bar switched, and regenerated on the outputs to go to the rows and columns of correlator chips (with the aid of external 1:2 LVDS buffers). Each RXP FPGA accepts only 16 wafers of HM Gbps data from the connector; to allow each device to have access to all 32 wafers requires a high-speed interconnect. This is accomplished with ~80 lines of 512 Mbps DDR using 1.8 V levels running each direction between chips—each chip therefore has access to the other chip's data. A simplified diagram showing the connectivity of the upper and lower RXP FPGAs is shown in Figure 3-1.

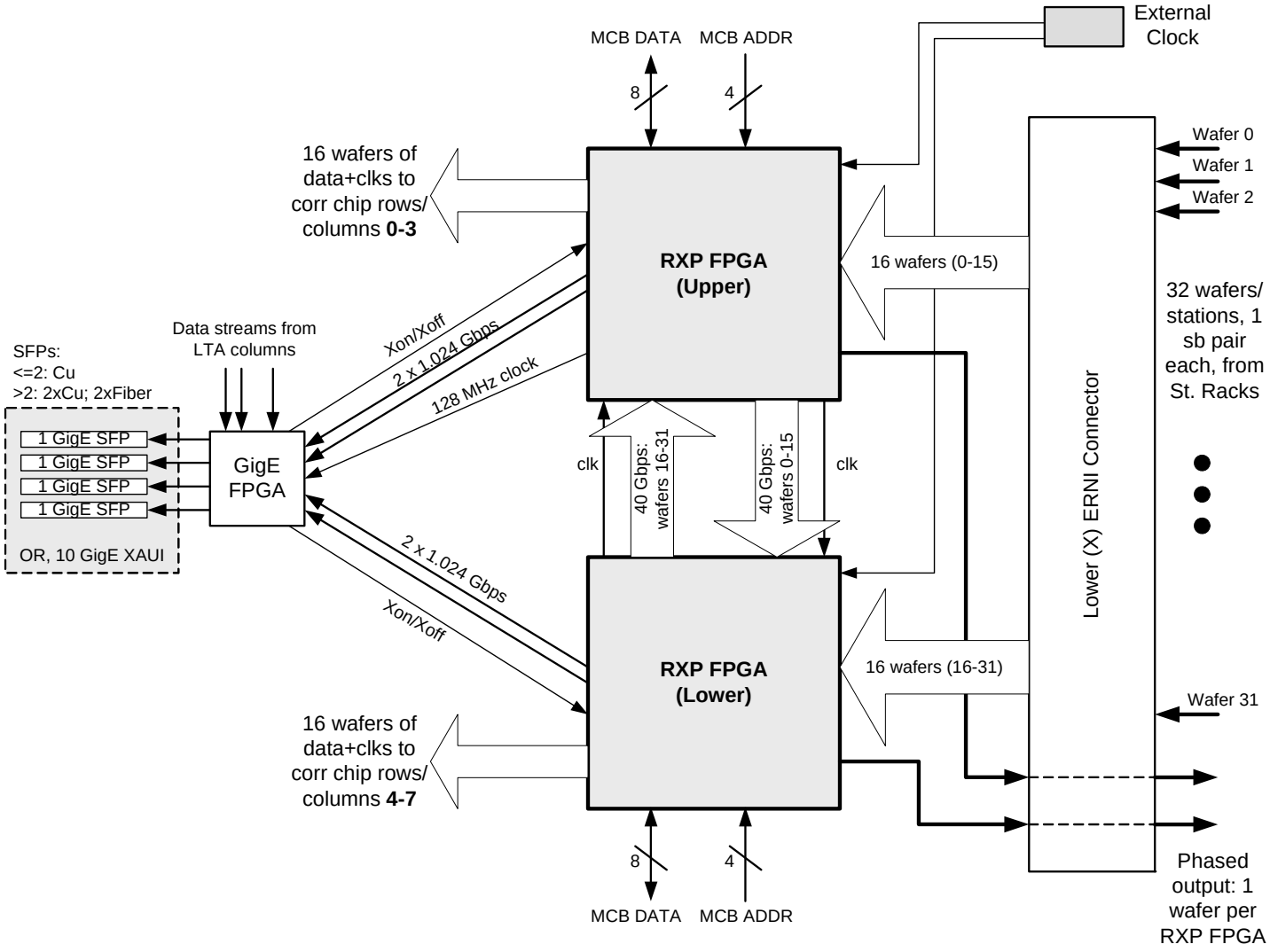


Figure 3-1 Simplified block diagram of the RXP FPGA's external world connections on the Baseline Board.

Each FPGA's 128 MHz clock normally sources from the clock contained in its lowest-numbered wafer input (i.e. wafer 0 for the UPPER chip, and wafer 16 for the LOWER chip). However, each FPGA can instead be set to select the external clock, available to the Baseline Board via a rear-panel SMA connector.

In normal operation, each FPGA selects 16 of the 32 input wafers (16 coming directly from the ERNI connector, 16 coming from the other FPGA), and these are transmitted to rows and columns of Correlator Chips via the Recirculation FPGAs.

Simultaneous with providing the Correlator Chip array with data, the chip may be configured to produce phased data (each chip can phase one sub-band). There are several outlets available for the phased data:

1. 1 GigE packets using the VDIF Version A.5.8 Frame Format [2]. In this case, the RXP chip builds the payload portion of the Ethernet frame, sends it to the GigE FPGA, where it is embedded in a complete UDP/IP Ethernet frame and sent out on one of the selected 1 GigE SFP ports. When using this method there are restrictions on the number of bits per sample and the phased bandwidth that may be contained within a single 1.024 Gbps link (and subsequent 1 GigE output). The Xon/Xoff signal from the GigE FPGA is available to use to enable/disable frame data flow, but this function is not implemented. If the GigE FPGA cannot handle the VDIF frame rate, it discards frames.

2. Phased data may be routed to wafers routing to select rows/columns of the Correlator Chip array (depending on what is selected in the UPPER or LOWER chip). This is useful if it is desired to auto-correlate phased data.

3. Phased data may be routed to output wafers (lower two rows of the X ERNI connector) (lower right of the figure) for use by an external piece of hardware.

All routes for phased data may be active simultaneously.

The RXP FPGA is configured and monitored via the Baseline Board MCB bus. For each chip, there are 8 bi-directional DATA lines, and 4 ADDRess lines; the limited number of address lines requires maximal use of indirect addressing when accessing chip internal registers.

4 Overview

A simplified block diagram of the RXP FPGA is shown in Figure 4-1.

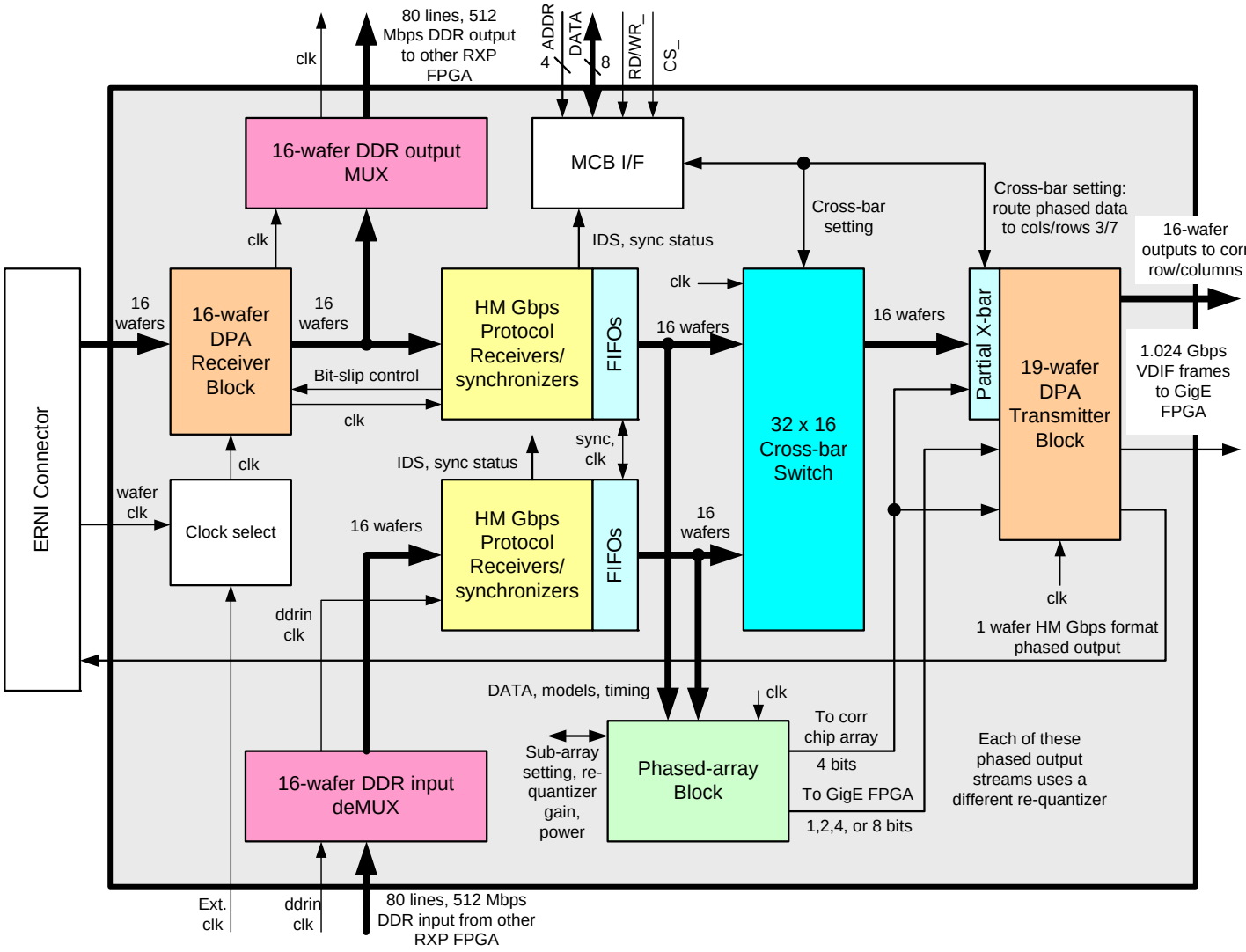


Figure 4-1 RXP FPGA simplified block diagram

The chip consists of two main functional components: a 32 x 16 cross-bar switch and the Phased-array Block. The rest of the chip's functions are for synchronization and re-timing of the input HM Gbps signals from the ERNI connector, synchronization of the DDR inputs and outputs, transmission of phased-array data to the GigE FPGA, and re-transmission of data, after cross-bar switching, to the Correlator Chip array in HM Gbps format.

A brief description of each block is as follows:

Clock Select – This is a built-in chip block that allows the controlling CPU to select the source of the 128 MHz clock—either from the 128 MHz clock on wafer 0, or from an external clock via a rear-entry External Clock SMA connector.

16-wafer DPA Receiver Block – This block contains the 1.024 Gbps LVDS receivers, Dynamic Phase Aligners, and serial-to-parallel converters. Bit alignment within the byte for each receiver is achieved by signaling from the HM Gbps Protocol Receivers. Because of bank requirements in the FPGA, this block contains an **A** and **B** sub-block, with each sub-block handling 8 wafers (otherwise A and B sub-blocks have no significance). This block also contains a 16-wafer HM Gbps signal generator/emulator invoked under software control; each wafer contains identical signaling, with simple dump and integrate parameters. This signal generator can be used to completely test the device, connections to/from its companion, and the entire Baseline Board with only a single External Clock (or wafer-0 clock) connection.

16-wafer DDR Output Mux – This mux converts the 320 signals at 128 MHz present at the output of the DPA receiver outputs to 80 512 Mbps DDR signals for transport to the companion RXP device.

16-wafer DDR Input DeMux – This block converts the 80 512 Mbps DDR input signals from the companion RXP device to 320 signals at 128 MHz for presentation to HM Gbps Protocol Receivers. Any bit-within-byte alignment signaling from these HM Gbps Protocol Receivers is ignored as alignment will be achieved once the companion RXP device has achieved alignment. As with the DPA Receiver Block, there are **A** and **B** sub-blocks here, with each sub-block handling 8 wafers.

HM Gbps Protocol Receivers/synchronizers – These blocks (one for the native wafers, and one for the wafers from the companion RXP device) automatically synchronize and lock to the HM Gbps signaling in each wafer, and provide lock status information and extracted ID information to the CPU via chip registers. These blocks also contain 64 sample deep FIFOs (allowing +/-125 nsec of delay difference) so that all 32 wafers' signals (which could source from Station Boards with large cable delay differences) are synchronized in time and into one clock domain before going into the 32x16 cross-bar switch and the Phased Array Block.

32x16 Cross-bar Switch – This is a full cross-bar switch that allows any of the 32 input wafers to be routed to any of the 16 wafer outputs.

Phased-array Block – This block performs single-stream phasing of any or all of the 32 input wafers. This block contains 2 re-quantizers—one 4-bit only re-quantizer for data fed to the correlator chip array and the X ERNI connector output, and one re-quantizer, with 1, 2, 4, or 8-bit offset binary capability for VDIF frame transmission. Gains can be dynamically (or statically) set for each re-quantizer, and simple 10 msec integration 2-bit and 4-bit re-quantizer power measurements are available to aid in setting gains. The total delay from the input of this block w.r.t. the TIMECODE T-bit marker, to the VDIF frame output, is given by the following equation:

$$RXP\_VDIF\_delay_{\sec} = \frac{12}{256x10^6} + \frac{109 + SDD}{sample\_rate_{Hz}} + \frac{N_{samples\_per\_word}}{32x10^6} \cdot VDIF_{payload\_size\_words}$$

Where "***SDD***" is an additional Sample rate Dependent Delay, making up for the fact that at the lowest 5 sample rates the data is delayed relative to the TIMECODE T-bit exiting the Station Board according to numbers found by Ken Sowinski and provided to me in an email from Dave Fort on December 2, 2009:

Sample Rate	**Delay (usec)**	***SDD* (samples)**
1 Ms/s	12	12
500 ks/s	298	149
250 ks/s	340	85
125 ks/s	424	53
62.5 ks/s	592	37
All others	0	0

Note that there is an additional frame-dependent delay in the GigE FPGA before data is finally pipelined and transmitted frame and sample rate independent on SFP2[1]:

$$GigE\_VDIF\_delay_{\sec} = \frac{VDIF_{frame\_size\_words}}{32x10^6}$$

Combining these two equations to determine the total sample rate, sample size, and frame size dependent delay (as might be needed for deterministic VDIF packet scheduling sourcing from different Baseline Boards) yields:

$$VDIF\_delay_{\sec} = \frac{12}{256x10^6} + \frac{109 + SDD}{sample\_rate_{Hz}} + \frac{N_{samples\_per\_word}}{sample\_rate_{Hz}} \cdot VDIF_{payload\_size\_words} + \frac{VDIF_{frame\_size\_words}}{32x10^6}$$

A "word" here is defined as a 32-bit VDIF frame word, $N_{samples\_per\_word}$ is the number of VDIF samples per 32-bit VDIF word, $VDIF_{payload\_size\_words}$ is the number of VDIF frame words not including the header, and $VDIF_{frame\_size\_words}$ is the total number of words in the frame including the header.

[1] If the 10G interface is activated, there is an additional delay in the GigE FPGA of ~90% of the frame at 125 Mbytes/sec due to the fact that the output rate on the 10G interface is 10X faster than it is being written and so minimum frame size independent pipelining in the UDP/IP frame buffer can't be used. A 2000-word frame would then incur an additional 0.9 x 2008 x 4/125 MHz ~= 58 µsec on top of the delays indicated by the above equations.

It is important to note that these equations provide delay information relative to the TIMECODE T-bit marker/epoch at the phasing block insertion point in the RXP FPGA for the purposes of setting the "RXP Packet Delay" registers in the GigE FPGA for deterministic packet scheduling. In the case where Baseline Boards are daisy-chained, the absolute delay of the T-bit (at the phasing block insertion point) in the second board compared to the first (at the same point) is ~0.38 μsec.

The TIMECODE T-bit is properly carried along with the data such that it consistently marks the 1 second epoch of the VDIF frame data, independent of the delay rate (provided the above ***SDD*** is correct). The ***SDD*** is hard-coded in the RXP FPGA; to change it the "base_delay_adder" assignment in the "p_hilbert127_8.v" file of the RXP FPGA design must be modified accordingly and then the chip re-compiled.

19-wafer DPA Transmitter – This block serializes the cross-bar output for transmission to the Correlator Chip array. It also routes and serializes phased data for transmission out the X ERNI connector for external use, to the Correlator Chip array for auto-correlation (i.e. using the "Partial X-bar Switch"), and for VDIF frame-format packets for transmission to the GigE FPGA and eventually out on Gig Ethernet. All of these outputs may be active simultaneously.

The design of each RXP chip is identical, however, pin assignments for each device are different to facilitate optimum board routing.

4.1 ERNI Connector Interface Signals

- Diff_in_pad[47:0] – Each of these 48 inputs are LVDS, each running at 1.024 Gbps. Chip input levels are set for thresholds of 200 mV pk-pk differential for detection. These use on-chip 100 ohm differential termination. Refer to the Appendix for detailed signal mapping to ERNI connector inputs for both the Upper and Lower RXPs.

- REF_CLOCK_pad[3:0] – 4 LVDS 128 MHz input clocks. Selection of which clock to use is FPGA compile-time defined; input [0] is currently used (sources from wafer 0 for the Upper RXP, and wafer 16 for the Lower RXP).

- Phasing_out_pad[3:0] – LVDS outputs that contain an entire wafer's worth of phased output data. For the Upper RXP, this routes to ERNI connecter row 34; for the Lower RXP, this routes to ERNI connector row 35. Channel [0] is the 128 MHz reference clock. Channel [1] is the CTRL stream; only the embedded TIMECODE[2] is applicable, all other signals are not applicable. Channel [2] is not active and is a "0". Channel [3] is the phased output, always 4 bits, chopped with the 4-bit chopper using the seed present in the companion CTRL-TIMECODE signal, and encoded using the HM Gbps Protocol DATA format, including

[2] The data is delayed in time relative to TIMECODE, depending on the sample rate. For the VDIF output, no such time skew exists as it is internally compensated.

embedded IDs defined by the PH_2CC_SID and PH_2CC_IDS registers. This is, effectively, the very same phased signal that gets routed to the correlator chip array.

4.2 512 Mbps DDR Interface Signals

All signals on this interface are 1.8 V level, single ended, using 12 mA drivers to meet performance requirements.

- DDROUT_pad[79:0] – 80 x 512 Mbps DDR outputs, 1.8 V levels. Refer to the Appendix for further detailed information on signal assignments.
- DDRIN_pad[79:0] – 80 x 512 Mbps DDR inputs, 1.8 V levels. Refer to the Appendix for further detailed information on signal assignments.
- DDROUT_CLK_pad – 256 MHz output clock, synchronous with DDROUT_pad.
- DDRIN_CLK_pad – 256 MHz input clock, synchronous with DDRIN_pad.
- SPARE_IO_pad[23:0] – These are 24 spare I/O signals running to/from the companion RXP.

4.3 Correlator Array row/column Interface Signals

- Diff_out_pad[47:0] – Each of these 48 outputs are LVDS, each running at 1.024 Gbps. Refer to the Appendix for detailed signal mapping to X/Y Recirc FPGAs for both the Upper and Lower RXPs.
- Clk_out_pad[7:0] – 8 LVDS 128 MHz clocks for driving X/Y Recirc FPGA reference clock inputs. The phase of these clocks w.r.t. Diff_out_pad signals is immaterial, but must be stable.

4.4 Phased Data Packet Interface Signals (to GigE FPGA)

- toGige_Data_pad[1:0] – Dual LVDS, 1.024 Gbps data lines routing to the GigE FPGA for transmission of VDIF phased data packets. Due to memory limitations in the GigE FPGA, only channel [1] is actively sending frames, channel [0] sends idle codes.
- toGigE_Clk_pad – LVDS, 128 MHz reference clock for the GigE FPGA. The phase of this clock w.r.t. toGige_Data_pad is immaterial, but must be stable.

4.5 MCB Interface Signals

All signals on this interface are 1.8 V levels; according to the Altera datasheet, these are 2.5 V tolerant (although the board does use a 2.5 V to 1.8 V level translator). Refer to the Appendix for more detailed information about signals on this interface.

- RESET_pad_ – Active -low asynchronous reset. Pulling this signal low resets all registers on the chip, but does not de-configure the FPGA.

- MCB_CS_pad_ — Active-low chip select. If asserted (low), and MCB_RD_WR_pad_ is high, the MCB_DATA_pad output drivers are turned on.

- MCB_CLK_pad – 33 MHz CPU clock. All MCB* signals are sampled on the rising edge of this clock. This clock must not be synchronous or frequency related to the reference 128 MHz clocks, as this clock is also used for internal monitor functions.

- MCB_RD_WR_pad_ — R/W select signal. If this is low, writing happens.

- MCB_ADDR_pad[3:0] – 4-bit address bus.

- MCB_DATA_pad[7:0] – 8-bit bidirectional data bus.

4.6 Miscl Interface Signals

- EXT_CLK128_in_pad – LVDS 128 MHz input clock, sourcing from the board's external clock input SMA connector.

There are a number of FPGA-specific programming and JTAG signals. Refer to the pinout listings in the Appendix, and the Altera Stratix-II data sheet for further information.

5 Requirements

The following is a list of RXP FPGA requirements.

5.1 Functional Requirements

Data reception, synchronization, cross-bar switching, and re-transmission

1. The FPGA must receive 16 HM Gbps wafer receivers. Since each wafer contains a 128 MHz clock, and 3 x 1.024 Gbps inputs, this implies that the FPGA must have 48 1.024 Gbps DPA receivers.

2. The FPGA must be able to select between the lowest-numbered wafer's clock, and an External 128 MHz clock, which sources from an external clock input to the board.

3. An on-board 16-wafer HM Gbps data generator must be enabled or disabled by the CPU via the MCB bus. The signals on each wafer are identical (i.e. one wafer's worth of signals are generated, and they are copied to all 16) and contain all the signaling necessary to allow the FPGA, its companion, and the entire Baseline Board to be tested with only a single 128 MHz External Clock (via the boards' External Clock SMA connector) or wafer 0 connection.

4. The FPGA must transmit "native" HM Gbps signals to its companion device on 512 Mbps DDR. Only the active DATA and CTRL signals need be transmitted on this interface (i.e. "fill" signals in the CTRL lines such as the "all 1's" and the PREAMBLE_BAR need not be transmitted), and approximately 80 lines and a clock are required.

5. The FPGA must receive 80 lines and a clock of 512 Mbps DDR from its companion device, and synchronize to them. It is left up to the companion device to perform proper DPA (Dynamic Phase Alignment), and bit-within-byte alignment of these signals, but once achieved, the FPGA must be able to synchronize to the HM Gbps protocol.

6. The FPGA must synchronize all of the 16 native and all of the 16 companion (via the DDR inputs) HM Gbps protocol signals into one clock domain, all time-aligned using the TIMECODE signals embedded in each wafer's CTRL line as a reference. What this requires, is that each wafer has a separate FIFO, which is 64 samples deep at 256 MHz—enough to absorb any time-skew differences in the system.

7. A full 32 x 16 cross-bar switch is required. This cross-bar switch allows each of the 16 wafer outputs that travel to a row/column of the Correlator Chip matrix to be selected from any of the 32 wafer inputs (16 native inputs, and 16 inputs from the companion RXP FPGA).

8. After the cross-bar switch, a partial cross-bar switch ("Partial X-bar") allows selection of phased data to be routed to BB0 or BB1 of any of the 16 output wafers. This allows unused correlator capacity to have the potential to (auto)-correlate phase data, if so selected.

9. After the cross-bar switch all wafer data is re-formatted to HM Gbps format for transmission to the board's row/column Recirculation FPGAs.

10. Cross-bar switching and re-formatting to HM Gbps format must not change the timing relationships of the data and the control signals within each wafer. Time skew between output wafers within the same RXP FPGA is now eliminated; however, there will still be time skew between output wafers of different RXP FPGAs. This remaining time skew will be removed in the Correlator Chip with its input FIFOs and does not show up as time skew in Recirculation FPGAs.

DDR outputs and inputs

11. Native wafer data, after dynamic phase alignment and de-multiplexing to 128 Mbps bit streams is multiplexed using a 2:1 multiplexer followed by a 2:1 DDR multiplexer to 512 Mbps for transmission to its companion RXP device. Each wafer requires 5 x 512 Mbps lines (4 for DATA, 1 for the 4 active CTRL signals TIMECODE, PHASERR, PHASEMOD, DUMPTRIG), for a total of 80 lines. A single overall synchronization signal for the 1st-stage mux is sent as well, along with a 256 MHz clock (this clock could be at a lower frequency).

12. The 80 + 1 sync + clock DDR input lines are resynchronized and de-multiplexed back to the 128 Mbps bit streams and so appear to on-chip HM Gbps receivers/synchronizers as native streams coming out of the DPA receiver block, except that bit-within-byte alignment is achieved by the companion device. DDR inputs are phase-synchronous with the input DDR clock, but not phase-synchronous with the native clock. This implies that a PLL with synchronization to the native clock is required, with the use of delay mismatch absorption FIFOs.

Phased-array block

13. The RXP FPGA must provide one phased-output on one data stream, operating on all 32 wafers. It must be possible via the MCB CPU interface to select which BB of the wafer (i.e. BB0 or BB1), and which of the 32 inputs are to be phased. It is not necessary to select a different BB from each wafer, it is either BB0 or BB1 from all wafers selected for phasing. It is necessary to be able to select any or all of the 32 inputs for phasing with complete flexibility.

Note: "Y1" output is produced when one antenna is selected for phasing. In this case, the output has delay and phase stopped for the phase center of the array, however that is chosen.

14. It must be possible, under CPU control, either to set data samples that are flagged invalid on the input streams with pseudo-random noise, derived from the input data, ***before*** complex mixing OR to set these flagged invalid samples to zero. Normally these are a few samples (10), at most once every 10 msec for each input stream[3]. If a wafer input that is to be phased has lost synchronization, data samples are set to zero to avoid adding pure noise to the phased output.

15. The first stage of phasing is **complex mixing**. Requirements are as follows:

    a. Phase for the complex mixer for each stream (i.e. BB0 or BB1 of each wafer) is developed from the PHASEMOD and PHASERR signals in the wafer in an identical fashion as for the Recirculation Controller FPGA. This requires a point-slope 32-bit phase synthesizer for each wafer.

    b. 32 bits of phase is translated into 9 bits of sine/cosine using on-chip fixed phase-to-sine/cosine LUTs (i.e. 512 x 9-bit LUTs[4]). 9 bits of phase is used to produce 9 bits of sine/cosine output. 9 bits limits the dynamic range of phased-array output to 54 dB (i.e. mixer harmonics are –54 dB down from the main mixer frequency). A fixed LUT does not allow for weighting of data before addition.

    c. 9 bit sine/cosine outputs of the LUT are then multiplied by the 4-bit or 7-bit DATA values, zero padded as necessary to 9 bits, to yield an 18-bit complex result. <u>All inputs, as set by the CPU, are selectively treated as 4 bits or 7 bits, but not combinations of both</u>. The 18 bits of this complex multiplier result from every input are then added in a 5-stage complex adder tree. <u>If one or more inputs are not to be included in the phased sum, as selected by the CPU, the complex multiplier result for that input is set to zero going into the adder tree</u>.

16. The next stage is the **complex adder tree**. Requirements are as follows:

    a. 32, 18-bit inputs. No truncation or rounding of data until the final result is obtained.

    b. Dynamically round and truncate the final adder tree result to 8 bits; the 8-bit window within the 23-bit sum is automatically determined in hardware

[3] With the HM Gbps signaling, it is possible that this is much less frequent, and not simultaneous on every wafer.

[4] A 256 x 8-bit LUT may instead be used if there is not enough memory in the FPGA for 512 x 9.

and applied under CPU control, and is the same for both the in-phase and quadrature sums. The 8-bit window will be determined by how many signals are active into the adder tree, and what their collective signal levels are.

17. The next stage is the **Hilbert transform**. Requirements are as follows:

    a. 63 or 127 (preferably) tap FIR filter, operating on only the quadrature summed result. Every-other tap weight is 0, tap coefficients are sign-asymmetric and amplitude symmetric about the center tap, and are 9 bits each. Tap coefficients are hard-coded into the FPGA and cannot be changed by the CPU, although they can be changed with a new FPGA compile and bitstream (personality). Tap coefficients are Hamming-windowed to avoid band-edge Gibbs affects.

    b. 8-bit DATA that has a symmetrical delay distance about the center tap ((N-1)/2) is first differenced, and then multiplied by a 9-bit tap coefficient. Since every other tap coefficient is 0, this means there are really (N+1)/4 (or (N-1)/4, which ever produces an integer result) subtractors and multipliers; for 127 Hilbert taps there are therefore 32 of these. This accomplishes a dual-tap multiply and add operation, where the differencing and then multiplication by a positive tap coefficient is the same as multiplying each DATA point by sign asymmetric tap coefficients that are identical in magnitude:

    $$D_{-i} * -coeff_i + D_{+i} * coeff_i = (D_{+i} - D_{-i}) * coeff_i$$

    A 9x9 multiply produces an 18-bit result. The 32, 18-bit results are added in a 5-stage adder tree, and produce a 23-bit result.

    c. The rounded upper 8 (of 23) bits of the quadrature-Hilbert-FIR'd result is scaled added to the scaled in-phase data, delayed by the appropriate amount ((N-1)/2 + 5 pipeline stages) to produce a simple 9-bit data stream. This 9-bit data stream goes into the re-quantizers.

18. The final stage of the Phased-array Block is the **re-quantizers**. There are 3 destinations for phased data. Data destined for the correlator chip array and the X ERNI connector output always use the same 4-bit, 2's complement symmetrical re-quantizer (**a 7-bit re-quantizer is NOT provided**); data destined for VDIF packets uses a selectable 1, 2, 4, or 8-bit offset binary re-quantizer.

    Each re-quantizer (except for 8-bit re-quantization as noted below) contains a scaling multiplier wherein the 9-bit real phased data is multiplied by an 8-bit CPU-defined unsigned coefficient to produce a scaled output, which is quantized with no bias to produce the final output. This scaling multiplier effectively sets the gain of the re-quantizer.

The 8-bit unsigned scaling coefficient can be set by the CPU via the MCB, to allow for a software AGC loop, if so desired.

When data is re-quantized to 8 bits for VDIF output, no scaling is performed, rather, the 9-bit 2's complement real data is no-bias converted to 8-bit offset binary.

19. A simple, 10 msec integration power measurement for 2-bit offset binary data destined for VDIF frames, and for 4-bit 2's complement data destined for the correlator chip array should be provided. If this is not possible, power measurements of these signals must be made using other means (software analysis of VDIF frames captured via the GigE FPGA, and correlator chip autocorrelation).

Phased data packet generation for VLBI recording; VDIF frame format

20. The number of bits per sample transmitted in the VDIF frame [2] is the same as the number of bits per sample selected out of the re-quantizer used for this purpose.

21. The following VDIF header parameters are explicitly CPU configurable:

    a. Ref Epoch (Word 1, 6 bits).

    b. Thread ID (Word 3, 10 bits).

    c. Station ID (Word 3, 16 bits).

    d. Invalid bit (Word 0, bit 31).

    All other VDIF frame parameters are either determined dynamically (as noted in following bullets), or are fixed in the FPGA:

    e. All Extended User Data (Words 4-7) are fixed and set to zero in the FGPA.

    f. Legacy bit (Word 0, bit 30) is set to 0.

    g. "Un-assigned" (Word 1 bits 30, 31) is set to 0.

    h. VDIF version (Word 2, bits 29-31) is set to 0.

    i. $\log_2$(#chans) (Word 2, bits 24-28) is set to 0.

    j. "C" (Word 3, bit 31) is set to 0 (Real data).

22. The CPU must be able to enable or disable flow of phased packets out of the FPGA to the GigE FPGA.

23. The data frame number and the 1-second count must be set automatically in the RXP FPGA. Note that the VDIF specification generally can use a different epoch than the correlator. To facilitate a difference between this epoch, and the epoch used in the correlator's TIMECODE, the CPU can set a 32-bit 2's complement offset that is ADDED to the correlator TIMECODE 1 second count to obtain the 1 second count that is put into the frame.

24. Delay that the data might experience relative to TIMECODE is properly compensated in the signal path. What this means is that phased data delay w.r.t. TIMECODE is zero, independent of phased sub-band bandwidth.

25. The length of the Data Array in the VDIF packet IN WORDS is set by the CPU via a 11-bit register and ****MUST** be in the range of 250-2000** (2000 max!!!) (1000 bytes to 8000 bytes). As indicated in the specification, care must be taken in setting the number of words per frame to ensure that there are an integer number of data frames per second, otherwise frame build sync errors occur, an indication of which is provided to the CPU. Note the number of bits per 32-bit word (4 bytes) for the supported number of bits per sample, and the maximum sample rate if only one link to the GigE FPGA is used:

    a. 1 bit samples: 32 samples per 4 bytes. 256 Ms/s max sample rate.
    b. 2 bit samples: 16 samples per 4 bytes. 256 Ms/s max sample rate.
    c. 4 bit samples: 8 samples per 4 bytes. 128 Ms/s max sample rate.
    d. 8 bit samples: 4 samples per 4 bytes. 64 Ms/s max sample rate.

26. As defined in the VDIF frame format specification, sample encoding is offset binary, with the lowest (most negative) value represented by all 0's, and the highest (most positive) value represented by all 1's.

27. Each RXP FPGA uses only one 1.024 Gbps data link to send VDIF data payload packets to the GigE FPGA.

5.2 Performance Requirements

1. The main clock domain of the chip is 128 MHz, supplied via an input wafer, or via an external clock sourcing from an SMA connector on the back of the Baseline Board.

2. The CPU "MCB" interface operates in a 33 MHz clock domain and is not frequency or phase synchronous with the 128 MHz clock domain.

3. The chip must handle 16 wafers of input data and control, and generate 16 wafers of output data and control. There are therefore 48, 1.024 Gbps inputs, and 48, 1.024 Gbps outputs.

4. The chip transmits “native” wafer data to its companion via 512 Mbps DDR. Approximately 8 lines in each direction are required.

5.3 Environmental Requirements

1. The RXP FPGA is in an F672 27x27 mm package and will be soldered on the Baseline Board in very close proximity to its companion.

2. Power dissipation of the chip, due to the number of high-speed I/Os is likely to be high. Initial vectorless power dissipation estimates indicate that power dissipation could be in the 4-8 W range.

3. All differential I/Os are LVDS. All single-ended I/Os are 1.8 V to meet speed requirements for 512 Mbps DDR between companion devices.

4. The device used to implement the FPGA is an Altera Stratix-II FPGA, EP2S60F672C4.

5.4 Interface Requirements

5.4.1 GigE FPGA Interface for Phased Data Requirements

1. For output to the GigE FPGA, and eventual recording by a VDIF VLBI disk-based recording system, packets of phased data, from the associated re-quantizer, are packaged and sent via one 1.024 Gbps LVDS link to the Gigabit Ethernet FPGA. The number of bits per sample x sample rate must not exceed ~950 Mbps otherwise data packets will be dropped.

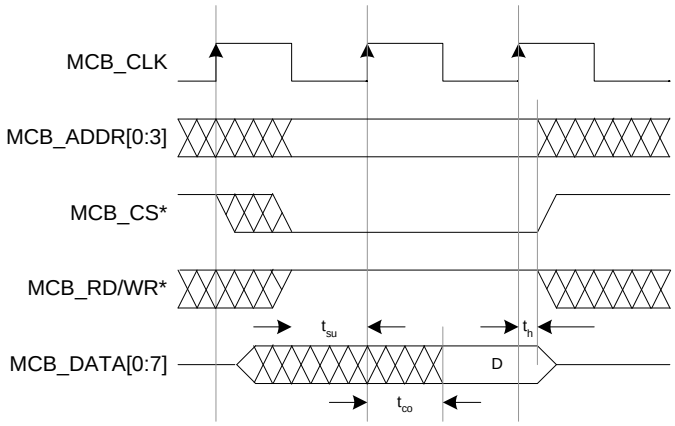
2. The RXP FPGA encapsulates the VDIF header and DATA payload in a frame for transmission to the GigE FPGA. Refer to Figure 5-3 of Gbit Ethernet Chip V2 RFS Version 1.1 or higher for more detailed information. Note that data is sent out, 1 byte at a time, reading the VDIF frame left to right, so as to meet Internet UDP/IP convention of the same.

3. XON or XOFF control from the GigE FPGA is not implemented. If the GigE FPGA can’t transmit an incoming packet from the RXP FPGA, it is dropped.

5.4.2 MCB (Microprocessor) Interface Requirements

The MCB (Monitor & Control Bus) interface allows a microprocessor to write into the RXP FPGA to configure it, and to read from it to verify configuration information and obtain status information. Physical interface timing requirements are shown in Figure 5-1.

MCB interface READ functional timing



MCB interface WRITE functional timing

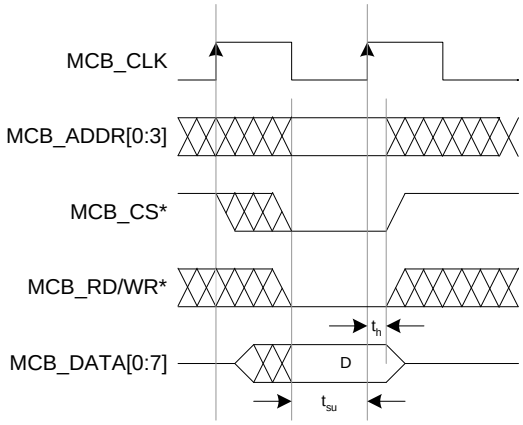


Figure 5-1 MCB interface functional timing requirements. READs require one clock cycle setup before data is ready—however, successive read take one clock cycle each. WRITEs require one clock cycle.

The FPGA will nominally have a $t_{su} \cong 5$ nsec, $t_{co} \cong 20$ nsec, and $t_h = 0$. If necessary, because there are no time-critical high-speed data transfer requirements, the interface FPGA on the Baseline Board can add more wait states if these timing requirements can't be met.

MCB interface voltage levels are 1.8 V, because of FPGA bank restrictions and the need for 1.8 V I/O to support 512 Mbps DDR.

The FPGA contains a number of configuration and status registers that can be accessed via the MCB. These registers are defined in the next section.

6 Functional Specifications

This section presents a plan for implementing the functionality of the RXP FPGA, in particular register set descriptions, some of the details of phasing logic, and example GUI screens.

6.1 MCB Registers

This section contains a detailed description of registers in the RXP FPGA. Since address space is limited (16 locations, 4 address bits) because of board routing restrictions, heavy use of indirect addressing is made.

6.1.1 Master Control/Status Register (MCSR); R/W Address=0x00

This register sets global chip configuration, and detects global status information. A complete description of register bits is as follows.

MCSR (R/W) Addr=0x0 (Reset=0x00)

7	6	5	4	3	2	1	0
TVT	TM	DPAmode	PLLlock	LEN	RXen	PLLen	CS

CS (R/W) – Clock select. If reset (0), then the wafer 128 MHz clock is used. If set (1), then the external clock, via the Baseline Board's external clock input is used.

PLLen (R/W) – 128 MHz DPA PLL enable. If reset (0), the main 128 MHz PLL is not enabled. If set (1), the PLL is enabled. Normally, once CS is set as desired, this bit is pulled and left high, unless the PLLlock bit indicates that that PLL is not locking…in which case pulling it low and then high could help in establishing PLL lock.

RXen (R/W) – This bit is used to enable or disable the DPA (Dynamic Phase Alignment) circuitry in the FPGA. Once the PLLlock bit is high indicating the 128 MHz PLL is locked, pulling this bit high starts the DPA circuitry locking on the correct phase of the input data, without knowledge of the HM Gbps protocol. Full HM Gbps protocol signal lock is then achieved by asserting the LEN bit.

LEN (R/W) – This bit is used to enable or disable (and restart) HM Gbps signal locking. If reset (0) then the HM Gbps protocol receivers are disabled. If set (1), then HM Gbps protocol receivers try to lock onto the input signals. Once lock is achieved (determined by status bits in the SRSR/SR_RX*_LS registers), then no further locking is attempted even if errors start occurring. In this case, if there are too many errors and the CPU

wishes to re-establish lock, it must pull this bit low and then high to restart lock acquisition.

PLLlock (R) – This bit indicates lock status of all PLLs on the FPGA. If reset (0), then lock has not been established on all PLLs. If set (1), all PLLs are locked.

DPAmode (R/W) – This bit is used to select Dynamic Phase Alignment mode for the DPA receivers. If reset (0), the DPA receivers are always dynamically tracking and adjusting to phase changes on the input. If set (1), the DPA receivers track and adjust to phase changes on the input until HM Gbps lock is established for each particular input pair. Once HM Gbps lock is established, the receiver phase aligners are turned off and sample the input on a fixed phase of the clock. This mode may be more robust to temporary dropouts or loss of signal coming from the source. It is unclear as to which mode will be the best once the entire system is installed. For now, the default should be 0.

TM (R/W) – Test Mode. If reset (0), the DPA receivers are synchronizing to input data streams and the FPGA works as normal. If set (1), input data is ignored, and the FPGA generates its own test vectors and feeds the signals to the rest of the chip, including the cross-bar and the phasing block. In this case, dump control signaling with 100 msec LTA integration times is included, as well as data streams, phase information etc. to allow the FPGA to fully stimulate the rest of the Baseline Board. Four wafers of vectors are included, and these are copied into the input 16-wafer data paths in four wafer blocks (i.e. inputs 0-3, 4-7, 8-11 etc. contain the same vectors).

Test Mode may be used with either the wafer clock (CS=0), in which case a clock must be supplied via the input wafer, or using the external clock (CS=1), in which case the clock must be supplied by the external connector.

TVT (R/W) – Test Vector Type. When TM=1, if this bit is reset (0), then data vectors contain noise which will correlate to produce fringes. In this case data streams contain noise vectors for only the first 10 μsec after each 10 msec time tick, the rest of the time they are invalid. When TM=1, if this bit is set (1), then data vectors always contain pseudo-random test patterns which will not correlate to produce fringes. However, in this case, data streams always contain test patterns and so this is a powerful mode for comparison testing. When TM=0 this bit is ignored.

NOTE: When TVT=0, internal DUMPTRIG signaling is such that data is output from the LTA every 100 msec. When TVT=1, internal DUMPTRIG signaling is such that data is output from the LTA every 1 sec. This allows some flexibility in test output data rate generation.

6.1.2 Status Register Select Register (SRSR) R/W Address=0x1

SRSR (R/W) Addr=0x1 (Reset=0x00)

This register sets the address that determines the contents of the **Status Register** according to the following table:

SRSR value	Status Register	Description
0x00	SR_PLL	PLL lock status register; a bit for each of 5 PLLs
0x01	SR_RXA_LS	Sub-block A DPA receiver lock status, wafers 0-7
0x02	SR_RXB_LS	Sub-block B DPA receiver lock status, wafers 8-15
0x03	WID_SELECT	Wafer ID select register; selects wafer and BB for the WID register
0x04	WIDR	Wafer ID register; contains SID and BBID/SBID for selected wafer and BB in WID_SELECT
0x05-0x07	reserved	unused
0x08-0x0F	SR_HMLS_A[0:7]	Sub-block A, HM Gbps lock status, wafers 0-7
0x10-0x17	SR_HMLS_B[0:7]	Sub-block B, HM Gbps lock status, wafers 8-15
0x18-0x1F	SR_HMLS_DDRA[0:7]	Sub-block A, DDRin HM Gbps lock status, wafers 0-7
0x20-0x27	SR_HMLS_DDRB[0:7]	Sub-block B, DDRin HM Gbps lock status, wafers 8-15
0x28-0x2F	SR_HMES_A[0:7]	Sub-block A, HM Gbps toggle/error status, wafers 0-7
0x30-0x37	SR_HMES_B[0:7]	Sub-block B, HM Gbps toggle/error status, wafers 8-15
0x38-0x3F	SR_HMES_DDRA[0:7]	Sub-block A, DDRin HM Gbps toggle/er status, 0-7
0x40-0x47	SR_HMES_DDRB[0:7]	Sub-block B, DDRin HM Gbps toggle/er status, 8-15
0x48-0x4F	reserved	unused
0x50-0x53	TICKSYNCERR[0:3]	Tick synchronization error status registers 0-3

Table 6-1 Status Register Select Register—SR register selections

The SRSR is an indirect address register only, and contains no actual status information. The procedure for accessing chip status via the Status Register is as follows:

1. Write the address of the desired Status Register to the SRSR register, according to the above table. For example, if the SR_HMLS_DDRB[0] register is to be accessed, write 0x20 to the SRSR register.

2. Read (and in some cases write) from (to) the Status Register at MCB address 0x2 to access the selected register. Following the example, reading from MCB address 0x2, accesses the SR_HMLS_DDRB[0] register.

6.1.3 Status Register (SR) R/W Address=0x2

SR (R/W) Addr=0x2

This is the register that, once the SRSR is set, provides access to most chip status information, including PLL lock status, HM Gbps receiver status, HM Gbps IDs etc., according to Table 6-1. A detailed description of each register defined in Table 6-1, is contained in the following sub-sections.

6.1.3.1 SR_PLL (R)

This register indicates detailed PLL lock status for the FPGA. SR_PLL bit definitions are according to the following table:

Bit	Description
0	Sub-block A (wafers0-7), RX PLL lock status; 0=not locked; 1=locked
1	Sub-block B (wafers8-15), RX PLL lock status; 0=not locked; 1=locked
2	DDR in PLL lock status; 0=not locked; 1=locked
3	reserved
4	Sub-block A (wafers0-7), TX PLL lock status; 0=not locked; 1=locked
5	Sub-block B (wafers8-15), TX PLL lock status; 0=not locked; 1=locked
6-7	reserved

6.1.3.2 SR_RXA_LS (R)

This register indicates DPA receiver lock status for sub-block A, wafers 0-7. This is ***not*** HM Gbps lock status, rather it is an indication as to whether the DPA receiver hard-block in the FPGA has acquired signal lock, for each wafer. Each bit corresponds to a wafer (B0—wafer0; B1—wafer1 etc.). If a bit is reset (0), lock has not been established. If a bit is set (1), lock has been established.

If DPA receiver lock status has not been acquired on a particular wafer, then it is unlikely that HM Gbps lock will be achieved for that wafer.

6.1.3.3 SR_RXB_LS (R)

Identical description to SR_RXA_LS, except for sub-block B, wafers 8-15.

6.1.3.4 WID_SELECT (R/W)

This register selects which embedded ID from which wafer will be present when the WIDR register is read. Selection is according to the following table:

Bit	Description
0	0=read the SID; 1=read the BBID/SBID
1	0=read ID from BB0 input of the wafer; 1=read ID from BB1 input
2-6	WAFER SELECT*, range of 0-31
7	unused

The actual ERNI connector input wafer that is selected depends on whether the UPPER or LOWER RXP FPGA (refer to Figure 3-1) is being read:

- **UPPER**: WAFER SELECT is identically equal to the input wafer numbering according to Figure 3-1. Wafers 0-15 are native inputs to this FPGA, wafers 16-31 are inputs via the LOWER RXP FPGA.

- **LOWER**: WAFER SELECT is somewhat reversed. WAFER SELECT 0-15 selects input wafers (i.e. from the ERNI connector) 16-31. WAFER SELECT 16-31 selects ERNI input wafers 0-15.

In any case WAFER SELECT numbering is consistent with input numbering in the 32 x 16 cross-bar switch and the Phased-array Block. For example, if WAFER SELECT is 3, then this ***is*** input 3 to the 32 x 16 cross-bar switch and input 3 to the Phased-array Block even though for the UPPER FPGA it is wafer 3 on the ERNI connector, but wafer 19 on the ERNI connector for the LOWER FPGA.

6.1.3.5 WIDR (R)

This is the wafer ID register that contains the embedded HM Gbps ID for the wafer selected with the WAFER_SELECT register. This can either be the 8-bit SID, or the 8-bit BBID/SBID combination, depending on Bit 0 of the WID_SELECT register. If the BBID/SBID combination, the SBID occupies bits 0-4, and the BBID occupies bits 5-7. (Note: ID bit assignments are identical to the SSIDRn-A and SSIDRn-B registers of the Recirculation FPGA)

To read a particular ID, the following steps are required by the CPU:

1. Write 0x03 to MCB address 0x1 (SRSR). This sets the SR to be the WID_SELECT register.

2. Write the desired ID selection to MCB address 0x2 (SR), according to the table in the WID_SELECT sub-section. For example, to read the BB0 SID, from wafer 3, write 0x0C to MCB address 0x2.

3. Write 0x04 to MCB address 0x1 (SRSR). This sets the SR to be the WIDR register.

4. Read MCB address 0x2, this contains the 8-bit SID (BB0, from wafer 3 for the example).

6.1.3.6 SR_HMLS_A[0]…SR_HMLS_A[7] (R)

These 8 registers contain HM Gbps receiver lock status for Sub-block A, native wafers 0-7. There is one register for each wafer. For the UPPER FPGA, these are ERNI input wafers 0-7. For the LOWER FPGA, these are ERNI input wafers 16-23. Each register's bit assignments are according to the following table:

Bit	Description
0	CTRL/TIMECODE lock status; 0=not locked; 1=locked
1	BB0 lock status; 0=not locked; 1=locked
2	BB1 lock status; 0=not locked; 1=locked
3	unused
4	CTRL/TIMECODE locking attempt; 0=no attempt; 1=attempting to lock
5	BB0 locking attempt; 0=no attempt; 1=attempting to lock
6	BB1 locking attempt; 0=no attempt; 1=attempting to lock
7	unused

Note that bits 0-2 provide essential information to determine if lock has been acquired on an input bit stream. Bits 4-6 provide auxiliary information to determine if internal lock-acquisition logic is functioning or not and is normally only ever required to be accessed during debugging.

6.1.3.7 SR_HMLS_B[0]…SR_HMLS_B[7] (R)

These 8 registers contain HM Gbps receiver lock status for Sub-block B, native wafers 8-15. There is one register for each wafer. For the UPPER FPGA, these are ERNI input wafers 8-15. For the LOWER FPGA, these are ERNI input wafers 24-31. Each register's bit assignments are identical to the table for the SR_HMLS_A[0:7] registers.

6.1.3.8 SR_HMLS_DDRA[0]… SR_HMLS_DDRA[7] (R)

These 8 registers contain HM Gbps receiver lock status for Sub-block A, DDR input wafers 0-7. There is one register for each wafer. For the UPPER FPGA, these are inputs from the LOWER FPGA via the DDR inputs, and source from ERNI wafers 16-23. For the LOWER FPGA, these are inputs from the UPPER FPGA via the DDR inputs, and source from ERNI wafers 0-7. Each register's bit assignments are identical to the table for the SR_HMLS_A[0:7] registers.

6.1.3.9 SR_HMLS_DDRB[0]… SR_HMLS_DDRA[7] (R)

These 8 registers contain HM Gbps receiver lock status for Sub-block B, DDR input wafers 8-15. There is one register for each wafer. For the UPPER FPGA, these are inputs from the LOWER FPGA via the DDR inputs, and source from ERNI wafers 24-31. For the LOWER FPGA, these are inputs from the UPPER FPGA via the DDR inputs, and source from ERNI wafers 8-15. Each register's bit assignments are identical to the table for the SR_HMLS_A[0:7] registers.

6.1.3.10 SR_HMES_A[0]… SR_HMES_A[7] (R)

These 8 registers contain HM Gbps receiver error status information for Sub-block A, native wafers 0-7. For the UPPER FPGA, these are ERNI wafers 0-7, and for the LOWER FPGA, these are ERNI wafers 16-23. **Each register is clear on read**. Each register's bit assignments are according to the following table:

Bit	Description
0	TIMECODE CRC error: 0=no error detected; 1=error detected
1	TIMECODE toggle detect: 0=not toggling; 1=toggling
2	BB0 Offset error: 0=no error; 1= error
3	BB0 CRC error: 0=no error; 1=error
4	BB0 toggle detect: 0=not toggling; 1=toggling
5	BB1 Offset error: 0=no error; 1= error
6	BB1 CRC error: 0=no error; 1=error
7	BB1 toggle detect: 0=not toggling; 1=toggling

6.1.3.11 SR_HMES_B[0]…SR_HMES_B[7] (R)

These 8 registers contain HM Gbps receiver error status information for Sub-block B, native wafers 8-15. For the UPPER FPGA, these are ERNI wafers 8-15, and for the LOWER FPGA, these are ERNI wafers 24-31. **Each register is clear on read**. Bit encoding is identical to the SR_HMES_A registers.

6.1.3.12 SR_HMES_DDRA[0]… SR_HMES_DDRA[7] (R)

These 8 registers contain HM Gbps receiver error status information for Sub-block A, DDR wafers 0-7. For the UPPER FPGA, these are ERNI wafers 16-23, and for the LOWER FPGA, these are ERNI wafers 0-7. **Each register is clear on read**. Bit encoding is identical to the SR_HMES_A registers.

6.1.3.13 SR_HMES_DDRB[0]… SR_HMES_DDRB[7] (R)

These 8 register contain HM Gbps receiver error status information for Sub-block B, DDR wafers 0-7. For the UPPER FPGA, these are ERNI wafers 24-31, and for the LOWER FPGA, these are ERNI wafers 8-15. **Each register is clear on read**. Bit encoding is identical to the SR_HMES_A registers.

6.1.3.14 TICKSYNCERR[0]… TICKSYNCERR[3] (R)

These 4, 8-bit registers provide tick synchronization error status information for each input wafer's time tick (100 PPS, TIMECODE T-bit) coming out of the FIFOs after the HM Gbps receivers and before the 32x16 cross-bar switch of Figure 4-1. If a bit is set (1) in this registers, then the time tick from the associated wafer is time-skewed relative to

the master wafer[5] and is an error condition. These registers are all cleared when TICKSYNCERR[3] is read (SRSR=0x53). Bit assignments to Upper and Lower RXP chips in these registers are as follows:

Input Wafer	UPPER RXP	LOWER RXP
0	TICKSYNCERR[0]-B0	TICKSYNCERR[2]-B0
1	TICKSYNCERR[0]-B1	TICKSYNCERR[2]-B1
2	TICKSYNCERR[0]-B2	TICKSYNCERR[2]-B2
3	TICKSYNCERR[0]-B3	TICKSYNCERR[2]-B3
4	TICKSYNCERR[0]-B4	TICKSYNCERR[2]-B4
5	TICKSYNCERR[0]-B5	TICKSYNCERR[2]-B5
6	TICKSYNCERR[0]-B6	TICKSYNCERR[2]-B6
7	TICKSYNCERR[0]-B7	TICKSYNCERR[2]-B7
8	TICKSYNCERR[1]-B0	TICKSYNCERR[3]-B0
9	TICKSYNCERR[1]-B1	TICKSYNCERR[3]-B1
10	TICKSYNCERR[1]-B2	TICKSYNCERR[3]-B2
11	TICKSYNCERR[1]-B3	TICKSYNCERR[3]-B3
12	TICKSYNCERR[1]-B4	TICKSYNCERR[3]-B4
13	TICKSYNCERR[1]-B5	TICKSYNCERR[3]-B5
14	TICKSYNCERR[1]-B6	TICKSYNCERR[3]-B6
15	TICKSYNCERR[1]-B7	TICKSYNCERR[3]-B7
16	TICKSYNCERR[2]-B0	TICKSYNCERR[0]-B0
17	TICKSYNCERR[2]-B1	TICKSYNCERR[0]-B1
18	TICKSYNCERR[2]-B2	TICKSYNCERR[0]-B2
19	TICKSYNCERR[2]-B3	TICKSYNCERR[0]-B3
20	TICKSYNCERR[2]-B4	TICKSYNCERR[0]-B4
21	TICKSYNCERR[2]-B5	TICKSYNCERR[0]-B5
22	TICKSYNCERR[2]-B6	TICKSYNCERR[0]-B6
23	TICKSYNCERR[2]-B7	TICKSYNCERR[0]-B7
24	TICKSYNCERR[3]-B0	TICKSYNCERR[1]-B0
25	TICKSYNCERR[3]-B1	TICKSYNCERR[1]-B1
26	TICKSYNCERR[3]-B2	TICKSYNCERR[1]-B2
27	TICKSYNCERR[3]-B3	TICKSYNCERR[1]-B3
28	TICKSYNCERR[3]-B4	TICKSYNCERR[1]-B4
29	TICKSYNCERR[3]-B5	TICKSYNCERR[1]-B5
30	TICKSYNCERR[3]-B6	TICKSYNCERR[1]-B6
31	TICKSYNCERR[3]-B7	TICKSYNCERR[1]-B7

Table 6-2 TICKSYNCERR[0]-[3] register bit assignments to input wafers.

For correct error free operation, these registers must always read 0. If not, then the wafer synchronization in the RXP chips could appear fine, but there will be synchronization errors in the Recirc FPGA HM Gbps receivers, depending on cross-bar switch settings.

[5] For the Upper RXP FPGA, the “master wafer” is wafer 0; for the Lower RXP FPGA, the “master wafer” is wafer 1.

6.1.4 Cross-bar Output Select Register (XBSELR) R/W Address=0x3

This register is used to select which output *wafer* (0…15) of the 32x16 cross-bar switch, any subsequent writes/reads to/from the XBCFGR register apply to. **It selects the *output* of the switch for which the input will be configured**.

XBSELR (R/W) Addr=0x3

Bits 0-3 select the output, 0-15. Bits 4-7 are unused.

Note that on reset/reboot, the cross-bar switch is set to route data straight through (i.e. input 0 routes to output 0, input 1 routes to output 1 etc.).

6.1.5 Cross-bar Configuration Register (XBCFGR) R/W Address=0x4

Writing to this register connects the output (selected by the XBSELR register), to the input set by the contents written to this register. Reading from this register reads back the value that was written.

XBCFGR (R/W) Addr=0x4

Bits 0-4 sets the input that the output is connected to.

Bits 5-7 are unused.

Example: if 6 is written to the UPPER FPGA's XBSELR register, and 23 is subsequently written to the XBCFGR register, then output 6 is connected to input 23.

Note that on reset/reboot, the cross-bar switch is set to route data straight through (i.e. input 0 connects to output 0, input 1 connects to output 1 etc.).

For the UPPER FPGA, inputs 0-31 of the cross-bar switch map directly to ERNI connector inputs 0-31.

For the LOWER FPGA, inputs 0-31 of the cross-bar switch map to ERNI inputs 16-31 and 0-15 respectively.

6.1.6 Phased Array Select Register (PHASELR) R/W Addr=0x5

PHASELR (R/W) Addr=0x5 (Reset=0x00)

This register sets the address for **Phasing Register** (PHR) function, according to the following table:

PHASELR value	Phasing Register selected via PHR	Description (page number)
0x00	PMCR	Phasing Master Control Register (39)
0x01	PHENR-0	Phasing Enable Register, wafers 0-7[6] (40)
0x02	PHENR-1	Phasing Enable Register, wafers 8-15 (40)
0x03	PHENR-2	Phasing Enable Register, wafers 16-23 (40)
0x04	PHENR-3	Phasing Enable Register, wafers 24-31 (40)
0x05	ATWINCR-0	Adder Tree WINdow Control Register 0 (41)
0x06	ATWINCR-1	Adder Tree WINdow Control Register 1 (42)
0x07	RG_4B_CC	Re-quantizer Gain, 4-bit, to Corr Chip Array (43)
0x08	RG_VDIF	Re-quantizer Gain, VDIF output (43)
0x09	VMCSR	VDIF Master Control/Status Register (44)
0x0A	VCONFIG1	VDIF Config Register 1 (Ref epoch, Thread ID MSBits) (46)
0x0B	VCONFIG2	VDIF Config Register 2 (Thread ID LSBits) (46)
0x0C	VCONFIG3	VDIF Config Register 3 (Frame size LSBits) (46)
0x0D	VCONFIG4	VDIF Config Register 4 (Frame size MSBits) (46)
0x0E	VSID-0	VDIF Station ID, LSBits (47)
0x0F	VSID-1	VDIF Station ID, MSBits (47)
0x10	VSECADD-0	VDIF SEConds ADDer, bits 0-7 (47)
0x11	VSECADD-1	VDIF SEConds ADDer, bits 8-15 (47)
0x12	VSECADD-2	VDIF SEConds ADDer, bits 16-23 (47)
0x13	VSECADD-3	VDIF SEConds ADDer, bits 24-31 (47)
0x1A	PH_2CC_CR	Phased data to Corr Chip array Control Register (48)
0x1B	PH_2CC_SID	Phased data to Corr Chip array SID (49)
0x1C	PH_2CC_IDS	Phased data to Corr Chip array BBID, SBID (49)
0x30	PMQERR	PhaseMod Quad wafer ERRor detect register (50)
0x31	PEQERR	PhasErr Quad wafer ERRor detect register (50)
0x32	PMDETECT-0	Phasemod detect register, wafers 0-7 (51)
0x33	PMDETECT-1	Phasemod detect register, wafers 8-15 (51)
0x34	PMDETECT-2	Phasemod detect register, wafers 16-23 (51)
0x35	PMDETECT-3	Phasemod detect register, wafers 24-31 (51)

[6] As with other parts of the RXP design, for the Upper RXP, there is a one-to-one mapping of input wafers 0-31 to these internal wafer numbers; for the Lower RXP, connector input wafers 0-15 are internal wafer numbers 16-31, and connector input wafers 16-31, are internal wafer numbers 0-15.

0x40	PCONTROL	Re-quantizer Power Control (and ready status) (52)
0x41	P_2BIT-0	2-bit re-quantizer power reg 0, bits 7-0 (53)
0x42	P_2BIT-1	2-bit re-quantizer power reg 1, bits 15-8 (53)
0x43	P_2BIT-2	2-bit re-quantizer power reg 2, bits 23-16 (53)
0x44	P_4BIT-0	4-bit re-quantizer power reg 0, bits 7-0 (53)
0x45	P_4BIT-1	4-bit re-quantizer power reg 1, bits 15-8 (53)
0x46	P_4BIT-2	4-bit re-quantizer power reg 2, bits 23-16 (53)
0x47	P_4BIT-3	4-bit re-quantizer power reg 3, bits 31-24 (53)

Table 6-3 Phased Array Select Register (PHASELR).

The PHASELR register is an indirect address register only as it is purely used to select which register is accessed when reading/writing the Phasing Register.

6.1.7 Phasing Register (PHR) R/W Address=0x6

PHR (R/W) Addr=0x6

This is the register that, once the PHASELR is set, provides access to phasing control and status information, according to Table 6-3. A detailed description of each register defined in Table 6-3, is contained in the following sub-sections. **THE ADDRESS SHOWN IN EACH SUB-SECTION** FOR THE REGISTER **IS THE PHASELR REGISTER VALUE** REQUIRED TO ACCESS THE REGISTER AT THE PHR MCB ADDRESS.

For example: To access the RG_VDIF register, first write 0x08 to the PHASELR register at MCB address 0x5; then read/write MCB address 0x6 (PHR address) to read/write the RG_VDIF register.

6.1.7.1 PMCR (Phasing Master Control Register) (0x00) (R/W)

PMCR (R/W) (PHASELR Addr=0x00) Reset: 0x00

This register sets global phasing parameters as defined below.

7	6	5	4	3	2	1	0
SR-3	SR-2	SR-1	SR-0	Dsize	BBsel	ICM	PEN

PEN (R/W) – Phasing Enable. If reset (0), phasing logic is disabled and held reset for minimum power use. If set (1), phasing logic is enabled. If this is low, all phasing logic, including the VDIF output, output to the correlator chip array, and output to the X ERNI connector is disabled.

ICM (R/W) – Invalid code mode. If reset (0), when the data stream for a particular station/antenna contains invalid data, zeros are inserted into the signal processing path for the data. If set (1), data flagged as invalid is not set to zero, rather it is treated as data "as is". Note that if a wafer receiver loses lock, and that wafer is selected to be part of the phased sum, the data is always set to zero to avoid contaminating the phased data with noise.

BBsel (R/W) – BB select. If reset (0), then "BB0" of all enabled wafers is phased. If set (1), then "BB1" of all enabled wafers is phased.

Dsize (R/W) – Data size select. If reset (0), then wafer data is 4-bits wide. If set (1), then wafer data is 7-bits wide. Note that if this is set (1), the sample rate MUST be <=128 Ms/s.

SR-[3:0] (R/W) – Sample rate. This is the same setting as the SRCRm_n registers in the Recirc FGPA (A25090N0000 RFS, Table 5-4). The sample rate is 256 Ms/s / 2^{SR}.

6.1.7.2 PHENR-0:3 (PHasing ENable Registers 0:3) (0x01…0x04) (R/W)

PHENR-0:3 (R/W) (PHASELR Addr=0x01-4) Reset: 0x07ffffff—Upper; 0xffff07ff—Lower

These are four 8-bit registers, which control whether or not a particular input wafer is included in the phased output. Assignments are according to the following table:

Register [bits]	ERNI Connector input wafers (Upper RXP)	ERNI Connector input wafers (Lower RXP)
PHENR-0 [7:0]	wf7 – wf0	wf23 – wf16
PHENR-1 [7:0]	wf15 – wf8	wf31 – wf24
PHENR-2 [7:0]	wf23 – wf16	wf7 – wf0
PHENR-3 [7:0]	wf31 – wf24	wf15 – wf8

Table 6-4 PHENR-0:3 register bit assignments to input wafers.

If a wafer is not included in the phased output, its contribution is set to zero, and the complex adder tree output window selection will adjust accordingly. Thus, the full range of any of the 32 input wafers can be included in the phased output.

6.1.7.3 ATWINCR-0 (Adder Tree Window Control Register-0) (0x05) (R/W)

ATWINCR-0 (R/W) (PHASELR Addr=0x05) Reset: 0x01

This register is used to control the bit selection window for the complex adder tree output, before the data enters the Hilbert transform FIR as follows:

7	6	5	4	3	2	1	0
Ws-3	Ws-2	Ws-1	Ws-0	Push	We	Ws	CON

CON (R/W) – Control mode. If reset (0), the integration start and stop time to determine the appropriate window is controlled by writing to the **Ws** (Window Start integration), and **We** (Window End integration) bits of this register. In this way, the integration time and epoch is completely controlled in software. If set (1), then the integration start and stop time uses an internal 10 second timer, continuously active, and aligned with the TIMECODE 1 PPS tick to determine the window. In this case, the Ws and We bits have no effect.

Ws (R/W) – If the CON bit is 0, writing a 1 to this bit starts the window integration. A 0 should be written immediately after. (i.e. s/w should set and then reset this bit to start the window integration). If the CON bit is 1, this bit has no effect or function.

We (R/W) – If the CON bit is 0, writing a 1 and then 0 to this bits stop the window integration. Whether the window is actually applied in hardware is determined by the setting of bit 0 of the ATWINCR-1 register. If the CON bit is 1, this bit has no effect or function.

Push (R/W) – This bit is used to determine whether there is additional headroom built into the window determination. If reset (0), the window is determined completely by the integration, and any time signals are such that the level exceeds the selected window, data overflow (resulting in signal sign inversion) will occur. If set (1), an additional bit is added to the window selection so that 6 dB of headroom is built in, resulting in a lower probability of data overflow. In this case, the data width into the Hilbert transform FIR is, effectively, 7 bits. This bit can be set or reset at any time to adjust the window to provide headroom or not, even if the result of the current integration is not actively applied (bit 0 of the ATWINCR-1 register 0).

Ws-[3:0] (R) – These 4 bits indicate the window that was determined with the last window integration determination. A lower value indicates the window is using lower order bits of the adder tree output, and a higher value indicates the window is using higher order bits of the adder tree output. This only reflects the actual applied window, if bit 0 of the ATWINCR-1 register has been set (1) before the last window integration.

6.1.7.4 ATWINCR-1 (Adder Tree Window Control Register-1) (0x06) (R/W)

ATWINCR-1 (R/W) (PHASELR Addr=0x06) Reset: 0x01

A single bit in this register (bit 0) is used to select whether the window found by window integration is applied (if bit set) or not applied (bit reset). If this bit is reset (0), any window integrations are not applied in hardware, but are reflected in the Ws-[3:0] bits of the ATWINDR-0 register. If this bit is set (1), then any window integrations are applied.

The standard method for using this bit is as follows:

1. Select all desired wafers and BB for phasing.

2. Ensure all receivers whose wafers are to be phased are synchronized.

3. Turn this bit on and perform the desired window integration. Use the Push bit of the ATWINCR-0 register if an additional 6 dB of headroom is desired, in cases where the signal level from one or more inputs may or might fluctuate.

4. Set the re-quantizer gains to get the desired output signal level (RG_4B_CC and RG_VDIF registers), for the selected number of re-quantization bits.

5. Turn this bit off, continue continuous window integrations (either auto or manual CON mode). Monitor the values of Ws-[3:0] to see if the window changes rapidly, is constant, or has changed to a new value for some time.

6. If a permanent change in Ws-[3:0] is observed (especially if it gets larger), then turn this bit on, get and apply a new window, and re-set the re-quantizer gains to achieve the desired signal level.

All other bits (b7-b1) in the register are unused, and should always read 0. Note that each step up or down of the Ws-[3:0] value is 6 dB of peak signal level.

6.1.7.5 RG_4B_CC (Re-quantizer Gain, 4-bit, to Corr Chip Array) (0x07) (R/W)

RG_4B_CC (R/W) (PHASELR Addr=0x07) Reset: 17d

This unsigned 8-bit register is used to set the re-quantizer gain for data destined for the correlator chip array and/or the X ERNI connector output. This number can be adjusted at any time to adjust the re-quantizer output signal level to the desired level. The re-quantizer output level can be determined by analyzing P_4BIT register contents. For 4-bit data, the optimum autocorrelation power (lag 0) is $\sim 2.7^2 = 7.29$. If the applied adder tree window changes, this gain normally has to be adjusted to achieve the stated optimum level.

6.1.7.6 RG_VDIF (Re-quantizer Gain, VDIF output) (0x08) (R/W)

RG_VDIF (R/W) (PHASELR Addr=0x08) Reset: 25d

This unsigned 8-bit register is used to set the re-quantizer gain for 1, 2, and 4-bit data that will be packed into VDIF packets to the GigE FPGA and onto an external data acquisition system. This number can be adjusted at any time to adjust the re-quantizer output signal level to the desired level, determined using the P_2BIT or P_4BIT registers. This register will have a different value depending on the number of re-quantization bits (1^7, 2, or 4), and depending on whether the adder tree window has changed. If 8-bit re-quantization is used, this register is ignored, as the Hilbert transform FIR filter output is used virtually directly by rounding to 8 bits and converting to offset binary.

Refer to the IMPORTANT EXPLANATORY NOTES on page 53 for more information on how these gain registers are used.

[7] For 1-bit re-quantization, clearly any non-zero value will be sufficient.

6.1.7.7 VMCSR (VDIF Master Control/Status Register) (0x09) (R/W)

VMCSR (R/W) (PHASELR Addr=0x09) Reset: 0x06

This is the primary register for control and monitor of VDIF frame generation and transmission activity.

7	6	5	4	3	2	1	0
0	FMD	FED	FGD	Nb-1	Nb-0	I	FTE

FTE (R/W) – Frame transmission enable. If this bit is reset (0), VDIF frame building and transmission is disabled. If this bit is set (1), VDIF frame building and transmission is enabled. **This bit should only be set once all other registers (except the RG_VDIF register, which can change at any time) are set to desired values, to avoid generating errors and garbage frames.** If, subsequently, either of the FED of FMD bits is detected high, the reasons could be one or more of the following:

1. Frame transmission is not properly synchronized to the system 1 PPS epoch (normally FED bit detected high indicates this). This could be the result of a recent loss of master wafer sync on the input to the RXP. Pulling the FTE bit low, waiting for **2 seconds**, and then pulling it high is required to get the VDIF payload builder properly synchronized to the 1 PPS epoch.

2. The chosen number of sample bits is too high for the selected sample rate, resulting in missing frames. Normally, the FMD bit detected high indicates this.

3. The chosen frame size, sample rate, and number of sample bits is such that there are not exactly an integer number of frames in 1 second. Normally, the FED bit detected high indicates this.

I (R/W) – This bit is written to all outgoing VDIF frames' "I" (Invalid) bit (Word 0, bit 31). N.B. I assume that if this is set (1), frames are marked invalid, whereas if it is reset (0), frames are marked as valid. The VDIF spec is a bit vague here.

Nb[1:0] (R/W) – This is used to set the number of re-quantization bits used in the VDIF re-quantizer, and included in the VDIF sampled data payload. Encoding is as follows: 00=1-bit; 01=2-bit; 10=4-bit; 11=8-bit.

FGD (R) – Frame Generation Detect. This bit is set (1) if at least one VDIF frame was generated since the last time this register was read. Cleared on read. If an actual count of the number of VDIF frames is required, refer to the GigE FPGA SFP2 output frame count capability.

FED (R) – Frame Error Detect. This bit will be set (1), if sample data payload is not properly synchronized to the 1 PPS epoch sourced from system TIMECODE, and will result in missing or incorrect time-aligned frames being generated. When this occurs, the FTE bit should be brought low and then high (refer to the FTE bit description) to re-synchronize the data payload builder to the 1 PPS. This bit is cleared on read.

FMD (R) – Frame Missing Detect. This bit will be set (1), if output frames are missing because the chosen sample rate and number of re-quantization bits is such that it exceeds the 1.024 Gbps transmission bandwidth to the GigE FPGA. This bit is cleared on read.

6.1.7.8 VCONFIG1 (VDIF Config Register 1) (0x0A) (R/W)

VCONFIG1 (R/W) (PHASELR Addr=0x0A) Reset: 0x00

This register is the first of 4 general VDIF frame configuration registers. Bit assignments are as follows:

B5-B0 – 6-bit Reference Epoch, included in the VDIF frame header, Word 1 bits 29-24.

B7-B6 – Most significant bits (bits 9-8) of the Thread ID, included in the VDIF frame header, Word 3, bits 25-24.

6.1.7.9 VCONFIG2 (VDIF Config Register 2) (0x0B) (R/W)

VCONFIG2 (R/W) (PHASELR Addr=0x0B) Reset: 0x00

This register is the second of 4 general VDIF frame configuration registers. Bit assignments are as follows:

B7-0 – Lease significant 8 bits (bits 7-0) of the Thread ID, included in the VDIF frame header, Word 3, bit 23-16.

6.1.7.10 VCONFIG3, VCONFIG4 (VDIF Config Registers 3, 4) (0x0C) (R/W)

VCONFIG3,4 (R/W) (PHASELR Addr=0x0C, 0x0D) Reset: 250d

These registers are the last two of the 4 general VDIF frame configuration registers. These registers contain the VDIF sampled data payload frame size in 32-bit **words**.

VCONFIG3 B7-0 – Least significant 8 bits of the frame size.

VCONFIG4 B2-0 – Most significant 3 bits of the frame size.

VCONFIG4 B7-3 – Unused, always reads 0.

Note that the frame size, in 32-bit words, MUST be in the range 250 to 2000. Frame sizes outside of this range result in indeterminate operation. The VDIF header "Data Frame Length – 1" is calculated internally and included in the VDIF header, Word 2, bits 23-0.

6.1.7.11 VSID-0,1 (VDIF Station ID registers 0, 1) (0x0E,0x0F) (R/W)

VSID-0,1 (R/W) (PHASELR Addr=0x0E, 0x0F) Reset: 8780d

These registers, together, set the 16-bit "Station ID" in Word 3, bits 15-0 of the VDIF frame header. VSID-0 (addr 0x0E) is the least-significant byte; VSID-1 (addr 0x0F) is the most-significant byte;

6.1.7.12 VSECADD-0…3 (VDIF Seconds Add registers 0-3) (0x10, 0x11, 0x12, 0x13) (R/W)

VSECADD-0…3 (R/W) (PHASELR Addr=0x10-0x013) Reset: 0x00000000

These 4 registers contain a signed (2's complement) 32-bit integer that is added to the TIMECODE "COUNTPPS" field to form the resulting VDIF header "Seconds from reference epoch", Word 0, bits 29-0 (i.e. the least significant 30 bits of the sum of COUNTPPS and VSECADD).

With these registers, the EVLA correlator system reference epoch can be transferred to the VDIF header reference epoch, provided they are within range. If the EVLA correlator system and VDIF epochs are the same, these registers are 0.

VSECADD-0 is the least-significant byte, and VSECADD-3 is the most significant byte.

6.1.7.13 PH_2CC_CR (PHasing to Corr Chip array Control Register) (0x1A) (R/W)

PH_2CC_CR (R/W) (PHASELR Addr=0x1A) Reset: 0x00

A single bit of this register (Bit 0) is used to control whether phased data is routed to the correlator chip array or not. If this bit is reset (0), then 4-bit phased data is NOT routed to the correlator chip array. If this bit is set (1), then 4-bit phased data (the output of the 4-bit re-quantizer, whose gain is controlled by the RG_4B_CC register) IS routed to the correlator chip array. Bits 1-6 of this register are unused, and always read 0.

If this bit is set, phased data is routed to the correlator chip array **via output wafer 15, BB1 signal path ONLY**, for both Upper and Lower RXPs. Therefore, for the Upper RXP, it means data is routed to Recirc X3/Y3 FPGAs; for the Lower RXP, it means data is routed to Recirc X7/Y7 FPGAs.

Note that system timing ticks are not affected by this routing. Phased data for this purpose is re-tagged with DATA headers (IDs, CRC-4 etc.) at the current system time, and so this means that any phased (auto-) correlated data should have their timestamps adjusted by plus 90 *samples* of delay plus 16 nsec.

IMPORTANT NOTE: *When not in testmode, within the RXP FPGA**, there is no DUMPTRIG generation for phased data**. Any DUMPTRIG generation for specific phased data IDs (set by the PH_2CC_SID and PH_2CC_IDS registers), which might be required, must still be performed on the Station Board.*

6.1.7.14 PH_2CC_SID (PHasing to Corr Chip array SID register) (0x1B) (R/W)

PH_2CC_SID (R/W) (PHASELR Addr=0x1B) Reset: 200d

This 8-bit register sets the SID (Station ID) contained in the DATA header for phased data routed to the correlator chip array.

6.1.7.15 PH_2CC_IDS (PHasing to Corr Chip array BBID/SBID register) (0x1C) (R/W)

PH_2CC_IDS (R/W) (PHASELR Addr=0x1C) Reset: 0x31 (BBID=1; SBID=17)

This 8-bit register sets the BBID and SBID contained in the DATA header for phased data routed to the correlator chip array. B7-B5 is the BBID; B4-B0 is the SBID.

IMPORTANT NOTE: When the RXP FPGA is in "testmode" (MCSR TM bit set), an internal DUMPTRIG signal is generated (with the same timing and integration parameters as for the other test streams), with SID=200, BBID=1, and SBID=17, to allow dumping of test phased output data to the correlator chip array. Therefore, if the IDs in the PH_2CC_SID and PH_2CC_IDS are changed from the above defaults, the test data DUMPTRIG will not match up, and no test phased data will be output.

6.1.7.16 PMQERR (PHASEMOD Quad wafer ERRor detect register) (0x30) (R)

PMQERR (R) (PHASELR Addr=0x30) Reset: 0x00

This 8-bit read-only register is used to provide PHASEMOD error detect status information on a quad-wafer basis. That is, each bit provides amalgamated error status for PHASEMOD from 4 wafers (grouped sequentially in 4, starting at wafer 0).

Bit 0 is status for wafers 0-3; Bit 1 is status for wafers 4-7 etc. ., where wafer numbering is internal inputs to the phasing block. Refer to Table 6-4 for ERNI connector input to phasing input mapping.

If a bit is set (1), then there has been a PHASEMOD CRC error detected on at least one of the 4 wafers in the quad wafer group since the last time this register was read.

This register is cleared on read.

If an antenna within the quad wafer group is NOT selected for phasing, then any errors detected on its associated PHASEMOD do not show up in the associated bit in this register. i.e. this register shows errors only for actively phased wafers, for both BB0 and BB1, no matter which BB is phased.

6.1.7.17 PEQERR (PHASERR Quad wafer ERRor detect register) (0x30) (R)

PEQERR (R) (PHASELR Addr=0x31) Reset: 0x00

This 8-bit read-only register is used to provide PHASERR error detect status information on a quad-wafer basis. That is, each bit provides amalgamated error status for PHASERR from 4 wafers (grouped sequentially in 4, starting at wafer 0).

Bit 0 is status for wafers 0-3; Bit 1 is status for wafers 4-7 etc., where wafer numbering is internal inputs to the phasing block. Refer to Table 6-4 for ERNI connector input to phasing input mapping.

If a bit is set (1), then there has been a PHASERR CRC (or offset) error detected on at least one of the 4 wafers in the quad wafer group since the last time this register was read. This register is cleared on read.

If an antenna within the quad wafer group is NOT selected for phasing, then any errors detected on its associated PHASERR do not show up in the associated bit in this register. i.e. this register shows errors only for actively phased wafers, for both BB0 and BB1, no matter which BB is phased.

6.1.7.18 PMDETECT-0…3 (PHASEMOD DETECT registers 0-3) (0x32…0x35) (R)

PMDETECT-0…3 (R) (PHASELR Addr=0x32…0x35) Reset: 0x00000000

These four 8-bit registers provide PHASEMOD detect status information for every wafer that is selected for phasing (refer to Table 6-4 for ERNI connector input to phasing input mapping).

If a bit is set (1) in this register, it means a PHASEMOD model for that particular wafer, for the particular selected BB, has been detected since the last time the register was read.

PMDETECT-0 Bits 7-0 are status bits for phasing block input wafers 7-0.

PMDETECT-1 Bits 7-0 are status bits for phasing block input wafers 15-8.

PMDETECT-2 Bits 7-0 are status bits for phasing block input wafers 23-16.

PMDETECT-3 Bits 7-0 are status bits for phasing block input wafers 31-17.

Each register is individually cleared on read.

6.1.7.19 PCONTROL (Power Control register) (0x40) (R/W)

PCONTROL (R/W) (PHASELR Addr=0x40) Reset: 0x00

This register is used to control acquisition of 2-bit offset binary (for VDIF frames), and 4-bit 2's complement (for correlator chip array) re-quantizer output power measurements.

7	6	5	4	3	2	1	0
0	0	0	0	PMM	R-4	R-2	PR

PR (R/W) – Power measurement Request. Pulling this high, and then immediately low (or, held high), initiates a power measurement, which will occur on the next 10 msec "system timing" interval. i.e. when PR is brought high at some arbitrary time, it will take a maximum of 20 msec for the power measurement to be complete.

R-2 (R) – 2-bit power Ready. This bit goes high when the 2-bit power measurement is complete and the P_2BIT-0…2 registers are ready to be read. This bit will stay high, and the P_2BIT-0…2 registers unchanged, until the next time the PR bit is brought low, and then high, initiating a new measurement.

R-4 (R) – Same function as R-2, except applies to 4-bit 2's complement re-quantizer power measurement. Normally R-2 and R-4 go high at the same time, but might not if the PR bit is brought high at about the same time as the system tick occurs.

PMM (R/W) – Power Measurement Mode. If reset (0), then operation is single-shot, CPU-controlled trigger and data acquisition exactly as described above. If set (1), then operation is continuous mode, driven by the FPGA's internal 10 msec time tick, synchronized to system timing (i.e. in phase with the Recirc FPGA output tick interrupt), in which case, every 10 msec a new measurement is made available in the P_2BIT and P_4BIT registers. In this case, the R-2 and R-4 bits are continuously high, and the PR bit must be held high. If the PR bit is pulled low, continuous acquisition ends.

6.1.7.20 P_2BIT-0…2 (2-bit re-quantizer Power register) (0x41…0x43) (R)

P_2BIT-0…2 (R) (PHASELR Addr=0x41…0x43) Reset: 0x000000

These three 8-bit registers contain the 2-bit re-quantizer raw accumulated power measurement. Power accumulation always happens at a 128 Ms/s sample rate for exactly 10 msec, no matter what the sample rate setting for phased data. P_2BIT-0 is bits 7-0, P_2BIT-1 is bits 15-8 etc.

To convert P_2BIT contents to a floating-point <R2> measurement, divide by 1.28×10^6.

6.1.7.21 P_4BIT-0…3 (4-bit re-quantizer Power register) (0x44…0x47) (R)

P_4BIT-0…3 (R) (PHASELR Addr=0x44…0x47) Reset: 0x00000000

These four 8-bit registers contain the 4-bit re-quantizer raw accumulated power measurement. Power accumulation always happens at a 128 Ms/s sample rate for exactly 10 msec, no matter what the sample rate setting for phased data. P_4BIT-0 is bits 7-0, P_4BIT-1 is bits 15-8 etc.

To convert P_4BIT contents to a floating-point <R4> measurement, divide by 1.28×10^6.

IMPORTANT EXPLANATORY NOTES:

The "P_2BIT" power register is used and useful only if the VDIF output is selected for 2 bits. If the VDIF output is selected for 1, 4, or 8 bits, the "P_2BIT" register does not contain useful data and must not be used.

If VDIF output is set for 1-bit, no power measurement is necessary and the RG_VDIF gain register should be set to some non-zero, arbitrary value (the default value is the best). Same for the case where VDIF output is set for 8 bits.

If VDIF output is set for 4-bits, the RG_4B_CC and the RG_VDIF gain registers must be set identically, and the P_4BIT register is used to obtain the VDIF power measurement, as well as always the data-to-corr chip-array power measurement.

In 8-bit VDIF output mode, the RG_VDIF has no effect, and no 8-bit power measurement is performed.

6.1.8 XBAR Board Control/Status Register (XBCSR) R/W Address=0x7

This register is used to obtain STATUS information from source Cross-Bar Boards (refer to section 7.5 of [1], version 1.42 or later). Note that STATUS information is available *only* from inputs that source *directly* from Cross-Bar Board outputs.

XBCSR (R/W) Addr=0x7 (Reset=0x00)

7	6	5	4	3	2	1	0
SED	SD	EN	RLG-1	RLG-0	Byte	Ws-1	Ws-0

Ws[1:0] (R/W) – Wafer select. This selects one of 4 input wafers to decode STATUS streams from. Selection is according to the following table:

RXP Chip	Ws-1	Ws-0	Input wafer	Source Station Rack
Upper	0	0	0	S001
	0	1	4	S002
	1	0	8	S003
	1	1	12	S004
Lower	0	0	16	S005
	0	1	20	S006
	1	0	24	S007
	1	1	28	S008

Note that the source sub-band depends on the particular Baseline Board the RXP resides on. For all even-slot Baseline Boards, no STATUS streams are present. For all odd-slot Baseline Boards STATUS information is from the sub-band the Baseline Board is assigned to. **Note that STATUS payloads for only sub-bands 0-7 contain useful TCR, TCF, and TCL LED status information.**

Byte (R/W) – Byte select. The STATUS stream consists of a 16-bit word, and this selects the byte within the word that is present when the XBSR register is read. 0=low byte; 1=high byte.

RLG[1:0] (R/W) – Wafer Rx LED group. Refer to Table 7-2 of the HM Gbps protocol spec [1], version 1.42 or higher. Set this as desired to select the group present when the XBSR is read.

EN (R/W) – ENable decode. If reset (0), the STATUS stream is not decoded. If set (1), it is. This bit should be pulled low any time Ws[1:0] are changed to select a different wafer input.

SD (R/W) – Status Detect. This bit will be set (1) if one or more valid STATUS stream words were detected since the last time this bit was written with a 0. i.e. this bit is cleared by writing 0 to it. On chip reset, the value of this bit is indeterminate.

SED (R/W) – Status Error Detect. This bit will be set (1) if one or more errored STATUS stream words were detected since the last time this bit was written with a 0. i.e. this bit is cleared by writing a 0 to it. On chip reset, the value of this bit is indeterminate.

6.1.9 XBAR Board Status Register (XBSR) R Addr=0x8

This read-only register contains a selected byte of a selected input wafer's STATUS information. If the **Byte** bit in the XBCSR is low, this will contain bits [7:0] of the STATUS payload for the last STATUS frame received. If the Byte bit in the XBCSR is high, this will contain bits [15:8] of the STATUS payload for the last STATUS frame received.

6.2 MCB Registers Summary Table

A summary table of all MCB registers is shown below:

Register	Page	R/W?	Address	Description
MCSR	26	R/W	0x0	Master Control/Status Register
SRSR	28	R/W	0x1	Status Register Select Register
SR		R/W	0x2	Status Register, **function depends on SRSR**:
SR_PLL	29	R	0x00	PLL lock status register
SR_RXA_LS	29	R	0x01	A DPA Rx lock status, wafers 0-7
SR_RXB_LS	29	R	0x02	B DPA Rx lock status, wafers 8-15
WID_SELECT	29	R/W	0x03	Wafer ID select register
WIDR	30	R	0x04	Wafer ID register
SR_HMLS_A	31	R	0x08-0x0F	A HM Gbps lock status, wafers 0-7
SR_HMLS_B	31	R	0x10-0x17	B HM Gbps lock status, wafers 8-15
SR_HMLS_DDRA	31	R	0x18-0x1F	A DDRin HM Gbps lock status, wafers 0-7
SR_HMLS_DDRB	31	R	0x20-0x27	B DDRin HM Gbps lock status, wafers 8-15
SR_HMES_A	32	R	0x28-0x2F	A, HM Gbps toggle/error status, wafers 0-7
SR_HMES_B	32	R	0x30-0x37	B, HM Gbps toggle/error status, wafers 8-15
SR_HMES_DDRA	32	R	0x38-0x3F	A DDRin HM Gbps tog/err stat, wafers 0-7
SR_HMES_DDRB	32	R	0x40-0x47	B DDRin HM Gbps tog/err stat, wafers 8-15
XBSELR	34	R/W	0x3	Cross-bar output Select Register (sets output)
XBCFGR	35	R/W	0x4	Cross-bar Configuration Register (sets input)
PHASELR	36	R/W	0x5	Phased Array Select Register
PHR		R/W	0x6	PHasing Reg, **func depends on PHASELR**:
PMCSR		R/W	0x00	Phasing Master Control/Status Reg
PHENR-0		R/W	0x01	Phasing Enable Reg-0
PHENR-1		R/W	0x02	-1
PHENR-2		R/W	0x03	-2
PHENR-3		R/W	0x04	-3
ATWINCR-0		R/W	0x05	Adder Tree WINdow Control Reg-0
ATWINCR-1		R/W	0x06	-1
RG_4B_CC		R/W	0x07	Re-quantizer Gain 4-bit to Corr Chip array
RG_VDIF		R/W	0x08	Re-quantizer Gain to VDIF output
VMCSR		R/W	0x09	VDIF Master Control/Status Reg
VCONFIG1		R/W	0x0A	VDIF Config register 1 (epoch, thread ID)
VCONFIG2		R/W	0x0B	VDIF Config register 2 (thread ID)
VCONFIG3		R/W	0x0C	VDIF Config register 3 (frame size)
VCONFIG4		R/W	0x0D	VDIF Config register 4 (frame size)
VSID-0		R/W	0x0E	VDIF Station ID-0
VSID-1		R/W	0x0F	-1
VSECADD-0		R/W	0x10	VDIF Seconds ADD-0
VSECADD-1		R/W	0x11	-1
VSECADD-2		R/W	0x12	-2
VSECADD-3		R/W	0x13	-3

PH_2CC_CR		R/W	0x1A	Phasing to Corr Chip array Control Reg
PH_2CC_SID		R/W	0x1B	Phasing to Corr Chip array SID
PH_2CC_IDS		R/W	0x1C	Phasing to Corr Chip BBID/SBID
PMQERR		R	0x30	PHASEMOD Quad wafer error detect
PEQERR		R	0x31	PHASERR Quad wafer error detect
PMDETECT-0		R	0x32	PHASEMOD Detect register-0
PMDETECT-1		R	0x33	-1
PMDETECT-2		R	0x34	-2
PMDETECT-3		R	0x35	-3
PCONTROL		R/W	0x40	Power Control and ready status register
P_2BIT-0		R	0x41	2-bit power register, bits 7-0
P_2BIT-1		R	0x42	2-bit power register, bits 15-8
P_2BIT-2		R	0x43	2-bit power register, bits 23-16
P_4BIT-0		R	0x44	4-bit power register, bits 7-0
P_4BIT-1		R	0x45	4-bit power register, bits 15-8
P_4BIT-2		R	0x46	4-bit power register, bits 23-16
P_4BIT-3		R	0x47	4-bit power register, bits 31-24
XBCSR	54	R/W	0x7	XBAR Board Control/Status Register
XBSR	55	R/W	0x8	XBAR Board Status Register

Table 6-5 MCB registers summary table

6.3 Phased Array Signal Processing

Detailed signal processing flow, including bit allocations, rounding, and selection is shown in Figure 6-1.

Rounding is used at various points to keep word width from growing excessively large, and to prevent DC bias accumulation.

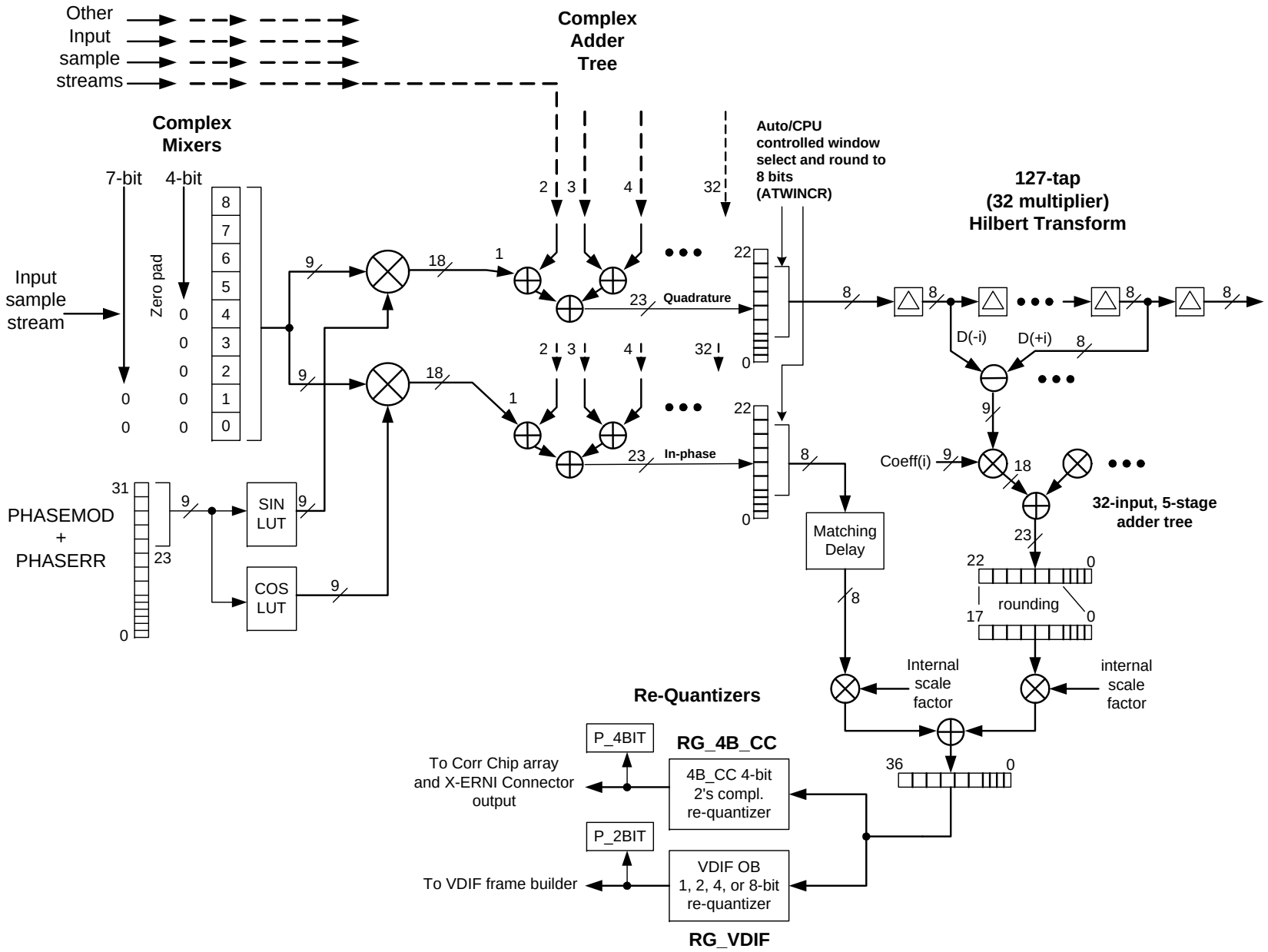


Figure 6-1 Simplified phased array signal processing, showing bit allocations, rounding etc. at each step.

6.4 RXP FPGA Graphical User Interfaces (GUIs)

This section presents representative examples of GUIs screens for the RXP FPGA. A 2-level hierarchy of screens is envisioned since it is thought to be impossible to present all available information on one screen. GUI screens are shown in following pages, and discussed in the following sub-sections. Not shown in any of the screens presented are blocks common to all GUI screens in the correlator.

6.4.1 Top-level RXP GUI Screen

This screen, shown in Figure 6-2, presents the entire chip in summary form. Clicking on select wafers of this GUI allows more detailed wafer information to be determined. Some settings are allowed:

1. Cross-bar connections are set by specifying which input wafer a particular output is connected to (far right of the figure). Output wafers are grouped in quads, and the screen shows where the quads go on the Baseline Board (different for the UPPER and LOWER RXP). The "PHS" shown in Figure 6-2 indicates that the output (BB1 in the wafer) is connected to phased output, rather than an input wafer.

2. Each wafer can be selected to be phased or not, as part of the wafer status block. If a wafer is selected for phasing, but the receiver is not locked hardware automatically disables phasing for that wafer. Until lock is established, the wafer check box is RED indicating a problem. If there are PHASEMOD or PHASERR errors (or missing PHASEMOD) for a particular phased wafer, the check box is also RED.

3. In the Phasing Block, various top-level parameters can be set as shown in the figure. For VDIF frame generation, a "VDIF settings" button opens another GUI allowing for setting of allowed VDIF frame parameters.

4. Selection of the 128 MHz clock source, either from the wafer, or from the external clock. PLL start, DPA receiver start, HM Gbps lock enable.

5. Selection of the input signal source, either from the input wafers, or from an internally generated signal for testing. There is no internal clock, a clock must be provided on the wafer 0 input, or via the External clock connector.

In the input wafer status block, LEDs are used to represent overall status for each wafer. If the lock LED is green, then lock is established on the CTRL, BB0, BB1 inputs of that wafer. Same for the inverse, and similarly for other things like "lock try", "errs", "tog" etc. Captured IDs in the input DATA streams are also displayed in "BB0/BB1" notation.

The diagram shows basic data routing, but these are not highlighted—highlighting should follow GUI screen conventions established for the correlator.

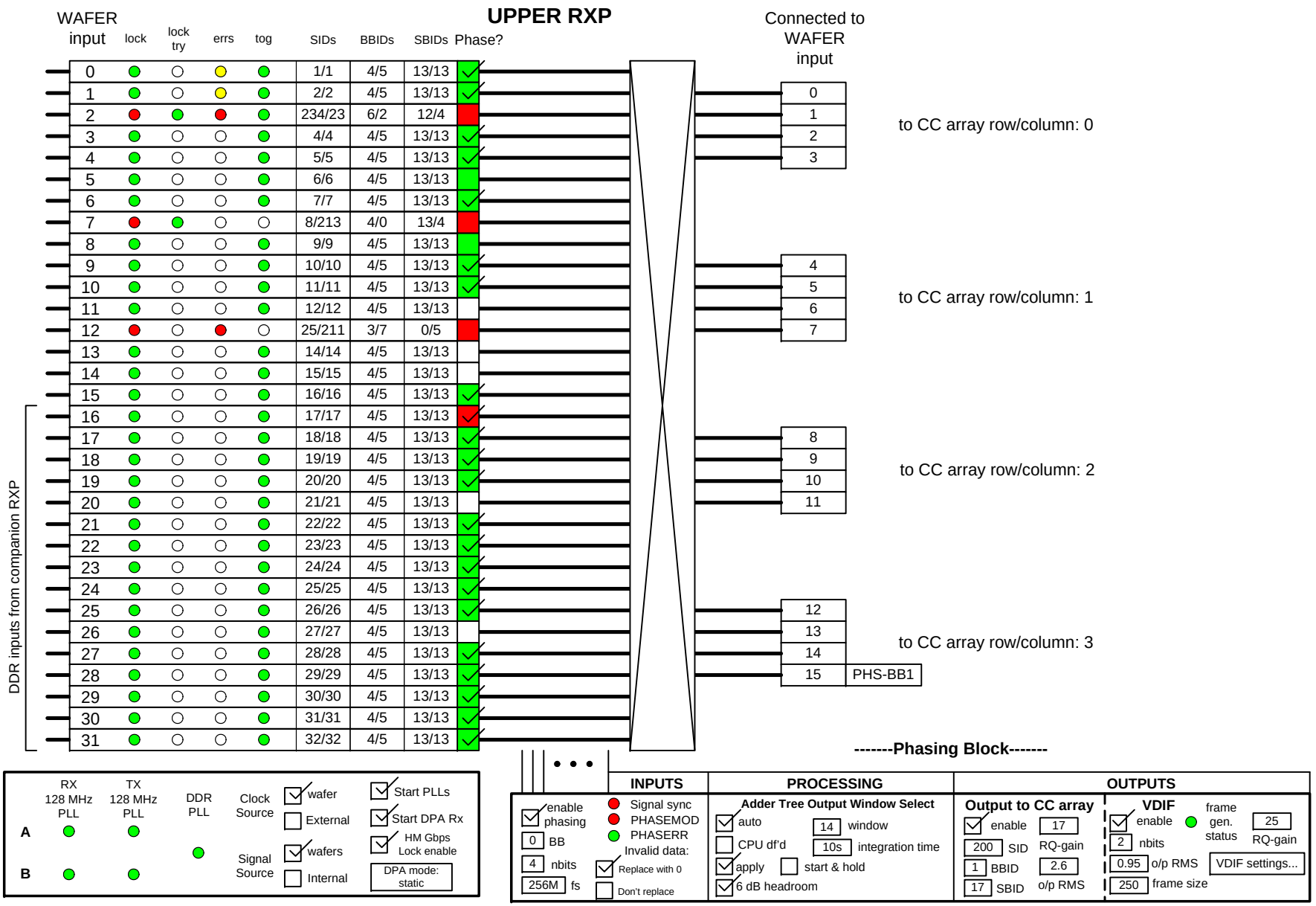


Figure 6-2 RXP FPGA top-level GUI screen concept

6.4.2 Single Wafer GUI Screen

Clicking on any of the "WAFER input" boxes of Figure 6-2 highlights that wafer to show more detailed information in a temporary or permanent window as shown in Figure 6-3. This screen is status only information and includes more detailed phasing status information (PHASEMOD detect/errors, PHASERR errors etc.)—not shown in the figure.

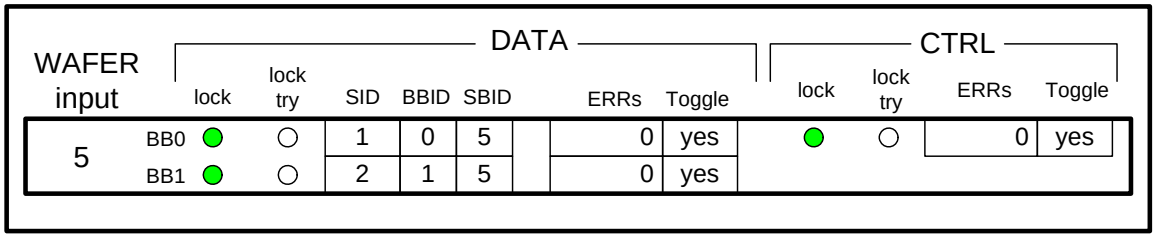


Figure 6-3 Wafer-only mini-GUI screen example

6.4.3 Phasing Block GUI Screen

This has been deleted in this version. There should be no need for a separate phasing GUI screen, except for a "VDIF settings" box (not shown in a figure) to allow VDIF header information to be defined; refer to the V* registers starting on page 44 for more information. All wafer-based status information for phasing should be represented in the top-level GUI, and in the top-level GUI's wafer-only mini screen (phasing part not shown).

7 References

[1] Carlson, B., PROTOCOL SPECIFICATION, HM Gbps Cable Signaling Specification, Protocol Document: A25022N0041, Revision 1.2, August 18, 2005.

[2] VLBI Data Interchange Format (VDIF) Specification, Ratified 26 June 2009, VDIF Task Force, Version 1.0.

[3] Zhang, H., INTERFACE CONTROL DOCUMENT, HM Gbps Cable Physical Specification, ICD: A25022N0040, Revision DRAFT, April 13, 2004.

8 Appendix I – VDIF frame

For convenience and reference, this appendix contains VDIF frame information copied from the VDIF specification [2].

VDIF Header

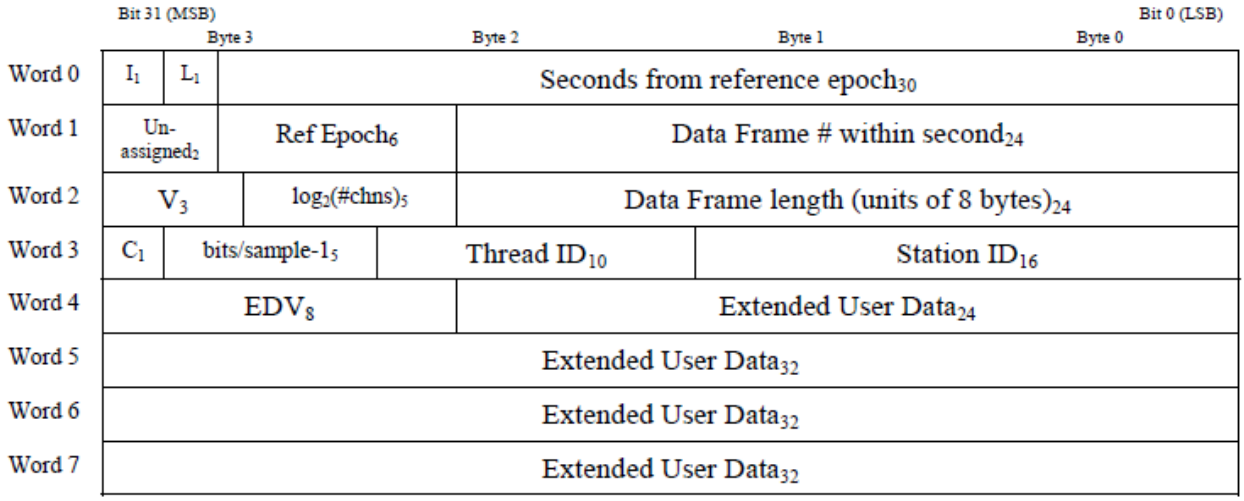


Figure 3: VDIF Data Frame Header format; subscripts are field lengths in bits; byte #s indicate relative byte address within 32-bit word (little endian format)

Refer to the VDIF specification for VDIF-specific notes. Note that the UDP/IP frame within which the VDIF frame is embedded contains VDIF bytes reading the above figure from right to left, as per the VDIF "little endian". Thus, the first byte in the UDP/IP frame is "Byte 0 of Word 0". The RXP transmits the bytes in the opposite order (big endian), and the GigE FPGA translates to the correct order.

RXP implementation specific notes:

- Word 0 Bit 30 is set to 0.
- Word 1 "Un-assigned", bits 30, 31, set to 0.
- Word 2 "V3", bits 29-30, and "$\log_2$(#chns), bits 28-24 set to 0.
- Word 3 "C1", bit 31, set to 0.
- Words 4-7 are all set to 0.
- Refer to text on page 14 for equations regarding sample rate and frame-size dependent VDIF packet delays to the GigE FPGA SFP2 output.

VDIF sampled data payload

The RXP FPGA supports VDIF payload sampled data sizes of 1, 2, 4, and 8-bits. One word of each of these formats is shown in the following figures.

Figure 4: Real 1-bit/sample data-word format (numbers in boxes indicate relative sample #'s in time order); byte #s indicate relative byte address within 32-bit word (little endian format).

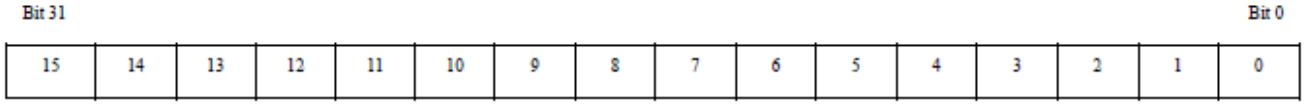


Figure 5: Real 2 bits/sample data-word format

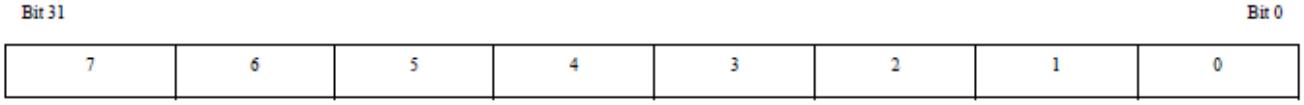


Figure 7: Real 4 bits/sample data-word format

8-bit samples are similar to the above, with 4 samples per 32-bit word.

The RXP FPGA does not support any other sampled data sizes.

Sample encoding is offset binary (e.g. “0000” is the most negative, and “1111” is the most positive).

9 Appendex II – Chip Pinouts, Pin, and Package Notes

This section contains exhaustive chip pinout listings generated by the FPGA compiler, and notes regarding pin connectivity and use on the board. Both Upper and Lower RXPs are implemented in an Altera EP2S60F672C4 FPGA.

9.1 UPPER RXP FPGA Pinouts

Pin Name/Usage	Location	Dir.	I/O Standard	Voltage	I/O Bank
GND	A2	gnd			
DDRIN_pad[70]	A3	input	1.8 V		4
VCCIO4	A4	power		1.8V	4
DDRIN_pad[2]	A5	input	1.8 V		4
DDROUT_pad[75]	A6	output	1.8 V		4
DDRIN_pad[55]	A7	input	1.8 V		4
DDROUT_pad[62]	A8	output	1.8 V		4
GND*	A9				4
GND*	A10				4
VCCIO4	A11	power		1.8V	4
Clk_out_pad[3]	A12	output	LVDS		9
GND	A13	gnd			
GND	A14	gnd			
GND*	A15				3
VCCIO3	A16	power		1.8V	3
SPARE_IO_pad[18]	A17	input	1.8 V		3
GND*	A18				3
SPARE_IO_pad[8]	A19	input	1.8 V		3
testmode_tick_out_pad	A20	output	1.8 V		3
testmode_tick_in_pad	A21	input	1.8 V		3
GND*	A22				3
VCCIO3	A23	power		1.8V	3
DDRIN_pad[35]	A24	input	1.8 V		3
GND	A25	gnd			
Diff_in_pad[12]	AA1	input	LVDS		6
Diff_in_pad[12](n)	AA2	input	LVDS		6
GND*	AA3				6
GND*	AA4				6
Diff_out_pad[41]	AA5	output	LVDS		6
Diff_out_pad[41](n)	AA6	output	LVDS		6
GND	AA7	gnd			
DDROUT_pad[12]	AA8	output	1.8 V		7
VREFB7	AA9	power			7
DDROUT_pad[46]	AA10	output	1.8 V		7
DDROUT_pad[11]	AA11	output	1.8 V		7
DDROUT_pad[69]	AA12	output	1.8 V		7
GND	AA13	gnd			
DDROUT_pad[37]	AA14	output	1.8 V		8
SPARE_IO_pad[15]	AA15	input	1.8 V		8
MCB_CLK_pad	AA16	input	1.8 V		8
DDROUT_pad[34]	AA17	output	1.8 V		8
VREFB8	AA18	power			8
DDRIN_pad[41]	AA19	input	1.8 V		8
nCONFIG	AA20				8
Diff_out_pad[5](n)	AA21	output	LVDS		1
Diff_out_pad[5]	AA22	output	LVDS		1
Diff_in_pad[8](n)	AA23	input	LVDS		1
Diff_in_pad[8]	AA24	input	LVDS		1
Diff_in_pad[9](n)	AA25	input	LVDS		1

Diff_in_pad[9]	AA26	input	LVDS			1
Diff_in_pad[3]	AB1	input	LVDS			6
Diff_in_pad[3](n)	AB2	input	LVDS			6
Diff_in_pad[2]	AB3	input	LVDS			6
Diff_in_pad[2](n)	AB4	input	LVDS			6
nCEO	AB5					7
PLL_ENA	AB6					7
DDROUT_pad[68]	AB7	output	1.8 V			7
DDRIN_pad[7]	AB8	input	1.8 V			7
DDROUT_pad[45]	AB9	output	1.8 V			7
DDROUT_pad[67]	AB10	output	1.8 V			7
DDRIN_pad[15]	AB11	input	1.8 V			7
DDRIN_pad[51]	AB12	input	1.8 V			7
GND*	AB13					7
GND*	AB14					8
DDRIN_pad[19]	AB15	input	1.8 V			8
DDRIN_pad[42]	AB16	input	1.8 V			8
MCB_DATA_pad[2]	AB17	bidir	1.8 V			8
DDRIN_pad[40]	AB18	input	1.8 V			8
DDROUT_pad[33]	AB19	output	1.8 V			8
DDRIN_pad[30]	AB20	input	1.8 V			8
TRST	AB21	input				8
TCK	AB22	input				8
Diff_in_pad[6](n)	AB23	input	LVDS			1
Diff_in_pad[6]	AB24	input	LVDS			1
Diff_in_pad[7](n)	AB25	input	LVDS			1
Diff_in_pad[7]	AB26	input	LVDS			1
VCCIO6	AC1	power			2.5V	6
Diff_in_pad[1]	AC2	input	LVDS			6
Diff_in_pad[1](n)	AC3	input	LVDS			6
VREFB6	AC4	power				6
VREFB7	AC5	power				7
DDROUT_pad[44]	AC6	output	1.8 V			7
DDRIN_pad[5]	AC7	input	1.8 V			7
DDROUT_pad[43]	AC8	output	1.8 V			7
DDRIN_pad[49]	AC9	input	1.8 V			7
DDRIN_pad[77]	AC10	input	1.8 V			7
VREFB7	AC11	power				7
Clk_out_pad[5](n)	AC12	output	LVDS			10
GND*	AC13					7
GND*	AC14					8
DDROUT_pad[3]	AC15	output	1.8 V			12
VREFB8	AC16	power				8
MCB_RD_WR_pad_	AC17	input	1.8 V			8
DDRIN_pad[24]	AC18	input	1.8 V			8
DDROUT_pad[2]	AC19	output	1.8 V			8
DDRIN_pad[39]	AC20	input	1.8 V			8
DDRIN_pad[31]	AC21	input	1.8 V			8
VREFB8	AC22	power				8
VREFB1	AC23	power				1
Diff_in_pad[4](n)	AC24	input	LVDS			1
Diff_in_pad[4]	AC25	input	LVDS			1
VCCIO1	AC26	power			2.5V	1
Diff_in_pad[0]	AD1	input	LVDS			6
Diff_in_pad[0](n)	AD2	input	LVDS			6
DDRIN_pad[0]	AD3	input	1.8 V			7
DDRIN_pad[48]	AD4	input	1.8 V			7
DDROUT_pad[42]	AD5	output	1.8 V			7
DDRIN_pad[72]	AD6	input	1.8 V			7
DDROUT_pad[9]	AD7	output	1.8 V			7
DDRIN_pad[46]	AD8	input	1.8 V			7
DDRIN_pad[9]	AD9	input	1.8 V			7
DDROUT_pad[10]	AD10	output	1.8 V			7

DDRIN_pad[50]	AD11	input	1.8 V		7
Clk_out_pad[5]	AD12	output	LVDS		10
Clk_out_pad[0](n)	AD13	output	LVDS		10
GND*	AD14				7
DDRIN_pad[43]	AD15	input	1.8 V		12
MCB_DATA_pad[7]	AD16	bidir	1.8 V		8
MCB_DATA_pad[5]	AD17	bidir	1.8 V		8
MCB_ADDR_pad[1]	AD18	input	1.8 V		8
MCB_ADDR_pad[0]	AD19	input	1.8 V		8
DDROUT_pad[32]	AD20	output	1.8 V		8
DDROUT_pad[1]	AD21	output	1.8 V		8
DDRIN_pad[38]	AD22	input	1.8 V		8
testmode_1pps_out_pad	AD23	output	1.8 V		8
TMS	AD24	input			8
Diff_in_pad[5](n)	AD25	input	LVDS		1
Diff_in_pad[5]	AD26	input	LVDS		1
GND	AE1	gnd			
nIO_PULLUP	AE2				7
DDRIN_pad[47]	AE3	input	1.8 V		7
DDROUT_pad[41]	AE4	output	1.8 V		7
DDRIN_pad[71]	AE5	input	1.8 V		7
DDROUT_pad[8]	AE6	output	1.8 V		7
DDRIN_pad[53]	AE7	input	1.8 V		7
DDROUT_pad[65]	AE8	output	1.8 V		7
DDROUT_pad[39]	AE9	output	1.8 V		7
DDRIN_pad[12]	AE10	input	1.8 V		7
GND*	AE11				7
Clk_out_pad[4](n)	AE12	output	LVDS		10
Clk_out_pad[0]	AE13	output	LVDS		10
GND*	AE14				7
GND*	AE15				8
MCB_CS_pad_	AE16	input	1.8 V		8
MCB_DATA_pad[4]	AE17	bidir	1.8 V		8
MCB_ADDR_pad[3]	AE18	input	1.8 V		8
DDRIN_pad[27]	AE19	input	1.8 V		8
SPARE_IO_pad[21]	AE20	input	1.8 V		8
DDRIN_pad[37]	AE21	input	1.8 V		8
SPARE_IO_pad[23]	AE22	input	1.8 V		8
DDRIN_pad[36]	AE23	input	1.8 V		8
DDROUT_pad[0]	AE24	output	1.8 V		8
TDI	AE25	input			8
GND	AE26	gnd			
GND	AF2	gnd			
DDROUT_pad[7]	AF3	output	1.8 V		7
VCCIO7	AF4	power		1.8V	7
DDRIN_pad[45]	AF5	input	1.8 V		7
DDROUT_pad[66]	AF6	output	1.8 V		7
DDROUT_pad[40]	AF7	output	1.8 V		7
DDRIN_pad[75]	AF8	input	1.8 V		7
DDROUT_pad[64]	AF9	output	1.8 V		7
DDROUT_pad[38]	AF10	output	1.8 V		7
VCCIO7	AF11	power		1.8V	7
Clk_out_pad[4]	AF12	output	LVDS		10
GND	AF13	gnd			
GND	AF14	gnd			
GND*	AF15				8
VCCIO8	AF16	power		1.8V	8
MCB_DATA_pad[6]	AF17	bidir	1.8 V		8
MCB_DATA_pad[3]	AF18	bidir	1.8 V		8
MCB_ADDR_pad[2]	AF19	input	1.8 V		8
DDRIN_pad[29]	AF20	input	1.8 V		8
DDROUT_pad[31]	AF21	output	1.8 V		8
DDRIN_pad[32]	AF22	input	1.8 V		8

VCCIO8	AF23	power		1.8V	8	
testmode_1pps_in_pad	AF24	input	1.8 V		8	
GND	AF25	gnd				
GND	B1	gnd				
MSEL1	B2				4	
fromGigE_XonXoff_pad	B3	input	1.8 V		4	
DDRIN_pad[1]	B4	input	1.8 V		4	
DDROUT_pad[74]	B5	output	1.8 V		4	
DDRIN_pad[73]	B6	input	1.8 V		4	
DDRIN_pad[4]	B7	input	1.8 V		4	
DDROUT_pad[23]	B8	output	1.8 V		4	
DDROUT_pad[76]	B9	output	1.8 V		4	
DDRIN_pad[59]	B10	input	1.8 V		4	
DDRIN_pad[13]	B11	input	1.8 V		4	
Clk_out_pad[3](n)	B12	output	LVDS		9	
DDRIN_CLK_pad	B13	input	1.8 V		4	
testmode_dummy_pad	B14	input	1.8 V		4	
GND*	B15				3	
DDRIN_pad[64]	B16	input	1.8 V		3	
SPARE_IO_pad[4]	B17	input	1.8 V		3	
SPARE_IO_pad[5]	B18	input	1.8 V		3	
DDROUT_pad[60]	B19	output	1.8 V		3	
SPARE_IO_pad[10]	B20	input	1.8 V		3	
DDROUT_pad[28]	B21	output	1.8 V		3	
DDRIN_pad[33]	B22	input	1.8 V		3	
SPARE_IO_pad[12]	B23	input	1.8 V		3	
SPARE_IO_pad[13]	B24	input	1.8 V		3	
CONF_DONE	B25				3	
GND	B26	gnd				
Diff_in_pad[46]	C1	input	LVDS		5	
Diff_in_pad[46](n)	C2	input	LVDS		5	
GND*	C3				4	
GND*	C4				4	
DDROUT_pad[22]	C5	output	1.8 V		4	
DDRIN_pad[54]	C6	input	1.8 V		4	
DDROUT_pad[63]	C7	output	1.8 V		4	
DDRIN_pad[6]	C8	input	1.8 V		4	
DDRIN_pad[57]	C9	input	1.8 V		4	
DDRIN_pad[78]	C10	input	1.8 V		4	
DDROUT_pad[78]	C11	output	1.8 V		4	
Clk_out_pad[2]	C12	output	LVDS		9	
GND*	C13				4	
GND*	C14				4	
GND*	C15				3	
SPARE_IO_pad[2]	C16	input	1.8 V		3	
DDROUT_pad[30]	C17	output	1.8 V		3	
DDROUT_pad[29]	C18	output	1.8 V		3	
SPARE_IO_pad[7]	C19	input	1.8 V		3	
DDRIN_pad[69]	C20	input	1.8 V		3	
SPARE_IO_pad[22]	C21	input	1.8 V		3	
DDROUT_pad[27]	C22	output	1.8 V		3	
DDRIN_pad[34]	C23	input	1.8 V		3	
DCLK	C24				3	
Diff_in_pad[47](n)	C25	input	LVDS		2	
Diff_in_pad[47]	C26	input	LVDS		2	
VCCIO5	D1	power		2.5V	5	
Diff_in_pad[44]	D2	input	LVDS		5	
Diff_in_pad[44](n)	D3	input	LVDS		5	
VREFB5	D4	power			5	
VREFB4	D5	power			4	
DDRIN_pad[3]	D6	input	1.8 V		4	
DDROUT_pad[21]	D7	output	1.8 V		4	
DDRIN_pad[56]	D8	input	1.8 V		4	

DDRIN_pad[10]	D9	input	1.8 V		4
DDROUT_pad[77]	D10	output	1.8 V		4
VREFB4	D11	power			4
Clk_out_pad[2](n)	D12	output	LVDS		9
Clk_out_pad[1]	D13	output	LVDS		9
GND*	D14				11
GND*	D15				3
VREFB3	D16	power			3
DDROUT_pad[59]	D17	output	1.8 V		3
DDRIN_pad[25]	D18	input	1.8 V		3
DDRIN_pad[68]	D19	input	1.8 V		3
DDROUT_pad[58]	D20	output	1.8 V		3
SPARE_IO_pad[11]	D21	input	1.8 V		3
VREFB3	D22	power			3
VREFB2	D23	power			2
Diff_in_pad[40](n)	D24	input	LVDS		2
Diff_in_pad[40]	D25	input	LVDS		2
VCCIO2	D26	power		2.5V	2
Diff_in_pad[39]	E1	input	LVDS		5
Diff_in_pad[39](n)	E2	input	LVDS		5
GND*	E3				5
GND*	E4				5
TEMPDIODEp	E5				
MSEL2	E6				4
DDRIN_pad[74]	E7	input	1.8 V		4
DDROUT_pad[73]	E8	output	1.8 V		4
DDROUT_pad[57]	E9	output	1.8 V		4
DDRIN_pad[79]	E10	input	1.8 V		4
DDROUT_pad[56]	E11	output	1.8 V		4
DDRIN_pad[61]	E12	input	1.8 V		4
Clk_out_pad[1](n)	E13	output	LVDS		9
DDROUT_pad[55]	E14	output	1.8 V		3
DDROUT_pad[54]	E15	output	1.8 V		3
~DATA0~ / RESERVED_INPUT	E16	input	1.8 V		3
SPARE_IO_pad[3]	E17	input	1.8 V		3
DDRIN_pad[67]	E18	input	1.8 V		3
DDRIN_pad[28]	E19	input	1.8 V		3
SPARE_IO_pad[9]	E20	input	1.8 V		3
nSTATUS	E21				3
nCE	E22				3
Diff_in_pad[34](n)	E23	input	LVDS		2
Diff_in_pad[34]	E24	input	LVDS		2
Diff_in_pad[45](n)	E25	input	LVDS		2
Diff_in_pad[45]	E26	input	LVDS		2
Diff_in_pad[38]	F1	input	LVDS		5
Diff_in_pad[38](n)	F2	input	LVDS		5
GND*	F3				5
GND*	F4				5
TEMPDIODEn	F5				
TDO	F6	output			4
MSEL3	F7				4
DDRIN_pad[76]	F8	input	1.8 V		4
VREFB4	F9	power			4
DDRIN_pad[11]	F10	input	1.8 V		4
DDROUT_pad[20]	F11	output	1.8 V		4
DDRIN_pad[17]	F12	input	1.8 V		4
DDROUT_pad[79]	F13	output	1.8 V		4
DDRIN_pad[18]	F14	input	1.8 V		3
DDROUT_pad[24]	F15	output	1.8 V		3
SPARE_IO_pad[16]	F16	input	1.8 V		3
DDRIN_pad[23]	F17	input	1.8 V		3
VREFB3	F18	power			3
SPARE_IO_pad[6]	F19	input	1.8 V		3

DDROUT_pad[26]	F20	output	1.8 V		3
Diff_out_pad[14](n)	F21	output	LVDS		2
Diff_out_pad[14]	F22	output	LVDS		2
Diff_in_pad[32](n)	F23	input	LVDS		2
Diff_in_pad[32]	F24	input	LVDS		2
Diff_in_pad[43](n)	F25	input	LVDS		2
Diff_in_pad[43]	F26	input	LVDS		2
Diff_in_pad[37]	G1	input	LVDS		5
Diff_in_pad[37](n)	G2	input	LVDS		5
GND*	G3				5
GND*	G4				5
VREFB5	G5	power			5
Diff_out_pad[38]	G6	output	LVDS		5
Diff_out_pad[38](n)	G7	output	LVDS		5
MSEL0	G8				4
DDROUT_pad[53]	G9	output	1.8 V		4
DDROUT_pad[52]	G10	output	1.8 V		4
DDROUT_pad[19]	G11	output	1.8 V		4
VCCA_PLL5	G12	power		1.2V	
GND	G13	gnd			
SPARE_IO_pad[14]	G14	input	1.8 V		3
DDRIN_pad[63]	G15	input	1.8 V		3
DDROUT_pad[51]	G16	output	1.8 V		3
DDRIN_pad[65]	G17	input	1.8 V		3
DDROUT_pad[61]	G18	output	1.8 V		3
DDROUT_pad[25]	G19	output	1.8 V		3
Diff_out_pad[13](n)	G20	output	LVDS		2
Diff_out_pad[13]	G21	output	LVDS		2
VREFB2	G22	power			2
Diff_in_pad[30](n)	G23	input	LVDS		2
Diff_in_pad[30]	G24	input	LVDS		2
Diff_in_pad[41](n)	G25	input	LVDS		2
Diff_in_pad[41]	G26	input	LVDS		2
Diff_in_pad[42]	H1	input	LVDS		5
Diff_in_pad[42](n)	H2	input	LVDS		5
GND*	H3				5
GND*	H4				5
Diff_out_pad[37]	H5	output	LVDS		5
Diff_out_pad[37](n)	H6	output	LVDS		5
Diff_out_pad[26]	H7	output	LVDS		5
Diff_out_pad[26](n)	H8	output	LVDS		5
DDRIN_pad[8]	H9	input	1.8 V		4
DDROUT_pad[72]	H10	output	1.8 V		4
DDRIN_pad[60]	H11	input	1.8 V		4
VCC_PLL5_OUT	H12	power		3.3V	9
GNDA_PLL5	H13	gnd			
VCCD_PLL5	H14	power		1.2V	
DDRIN_pad[20]	H15	input	1.8 V		3
SPARE_IO_pad[1]	H16	input	1.8 V		3
SPARE_IO_pad[19]	H17	input	1.8 V		3
DDROUT_pad[15]	H18	output	1.8 V		3
Diff_out_pad[15](n)	H19	output	LVDS		2
Diff_out_pad[15]	H20	output	LVDS		2
Diff_out_pad[2](n)	H21	output	LVDS		2
Diff_out_pad[2]	H22	output	LVDS		2
Diff_in_pad[28](n)	H23	input	LVDS		2
Diff_in_pad[28]	H24	input	LVDS		2
Diff_in_pad[35](n)	H25	input	LVDS		2
Diff_in_pad[35]	H26	input	LVDS		2
Diff_in_pad[27]	J1	input	LVDS		5
Diff_in_pad[27](n)	J2	input	LVDS		5
GND*	J3				5
GND*	J4				5

Diff_out_pad[39]	J5	output	LVDS		5	
Diff_out_pad[39](n)	J6	output	LVDS		5	
Diff_out_pad[25]	J7	output	LVDS		5	
Diff_out_pad[25](n)	J8	output	LVDS		5	
DDROUT_pad[18]	J9	output	1.8 V		4	
DDROUT_CLK_pad	J10	output	1.8 V		4	
DDRIN_pad[16]	J11	input	1.8 V		4	
VCCPD4	J12	power		3.3V	4	
GNDA_PLL5	J13	gnd				
DDRIN_pad[62]	J14	input	1.8 V		3	
SPARE_IO_pad[0]	J15	input	1.8 V		3	
VCCPD3	J16	power		3.3V	3	
DDROUT_pad[50]	J17	output	1.8 V		3	
DDRIN_pad[66]	J18	input	1.8 V		3	
Diff_out_pad[19](n)	J19	output	LVDS		2	
Diff_out_pad[19]	J20	output	LVDS		2	
Diff_out_pad[1](n)	J21	output	LVDS		2	
Diff_out_pad[1]	J22	output	LVDS		2	
Diff_in_pad[22](n)	J23	input	LVDS		2	
Diff_in_pad[22]	J24	input	LVDS		2	
Diff_in_pad[33](n)	J25	input	LVDS		2	
Diff_in_pad[33]	J26	input	LVDS		2	
Diff_in_pad[26]	K1	input	LVDS		5	
Diff_in_pad[26](n)	K2	input	LVDS		5	
GND*	K3				5	
GND*	K4				5	
VREFB5	K5	power			5	
Diff_out_pad[43]	K6	output	LVDS		5	
Diff_out_pad[43](n)	K7	output	LVDS		5	
Diff_out_pad[27]	K8	output	LVDS		5	
Diff_out_pad[27](n)	K9	output	LVDS		5	
DDROUT_pad[17]	K10	output	1.8 V		4	
DDROUT_pad[71]	K11	output	1.8 V		4	
GND	K12	gnd				
VCCIO4	K13	power		1.8V	4	
GND	K14	gnd				
VCCIO3	K15	power		1.8V	3	
DDROUT_pad[16]	K16	output	1.8 V		3	
DDRIN_pad[22]	K17	input	1.8 V		3	
SPARE_IO_pad[20]	K18	input	1.8 V		3	
Diff_out_pad[22](n)	K19	output	LVDS		2	
Diff_out_pad[22]	K20	output	LVDS		2	
Diff_out_pad[3](n)	K21	output	LVDS		2	
Diff_out_pad[3]	K22	output	LVDS		2	
Diff_in_pad[20](n)	K23	input	LVDS		2	
Diff_in_pad[20]	K24	input	LVDS		2	
Diff_in_pad[31](n)	K25	input	LVDS		2	
Diff_in_pad[31]	K26	input	LVDS		2	
VCCIO5	L1	power		2.5V	5	
Diff_in_pad[36]	L2	input	LVDS		5	
Diff_in_pad[36](n)	L3	input	LVDS		5	
Phasing_out_pad[3]	L4	output	LVDS		5	
Phasing_out_pad[3](n)	L5	output	LVDS		5	
Phasing_out_pad[1]	L6	output	LVDS		5	
Phasing_out_pad[1](n)	L7	output	LVDS		5	
Diff_out_pad[46]	L8	output	LVDS		5	
Diff_out_pad[46](n)	L9	output	LVDS		5	
VCCINT	L10	power		1.2V		
GND	L11	gnd				
VCCINT	L12	power		1.2V		
GND	L13	gnd				
VCCINT	L14	power		1.2V		
GND	L15	gnd				

VCCINT	L16	power		1.2V		
GND	L17	gnd				
Diff_out_pad[31](n)	L18	output	LVDS		2	
Diff_out_pad[31]	L19	output	LVDS		2	
Diff_out_pad[12](n)	L20	output	LVDS		2	
Diff_out_pad[12]	L21	output	LVDS		2	
Diff_out_pad[7](n)	L22	output	LVDS		2	
Diff_out_pad[7]	L23	output	LVDS		2	
Diff_in_pad[18](n)	L24	input	LVDS		2	
Diff_in_pad[18]	L25	input	LVDS		2	
VCCIO2	L26	power		2.5V	2	
Diff_in_pad[25]	M1	input	LVDS		5	
Diff_in_pad[25](n)	M2	input	LVDS		5	
Phasing_out_pad[2]	M3	output	LVDS		5	
Phasing_out_pad[2](n)	M4	output	LVDS		5	
Phasing_out_pad[0]	M5	output	LVDS		5	
Phasing_out_pad[0](n)	M6	output	LVDS		5	
Diff_out_pad[36]	M7	output	LVDS		5	
Diff_out_pad[36](n)	M8	output	LVDS		5	
VCCPD5	M9	power		3.3V	5	
GND	M10	gnd				
VCCINT	M11	power		1.2V		
GND	M12	gnd				
VCCINT	M13	power		1.2V		
GND	M14	gnd				
VCCINT	M15	power		1.2V		
GND	M16	gnd				
VCCIO2	M17	power		2.5V	2	
VCCPD2	M18	power		3.3V	2	
Diff_out_pad[34](n)	M19	output	LVDS		2	
Diff_out_pad[34]	M20	output	LVDS		2	
Diff_out_pad[16](n)	M21	output	LVDS		2	
Diff_out_pad[16]	M22	output	LVDS		2	
Diff_out_pad[10](n)	M23	output	LVDS		2	
Diff_out_pad[10]	M24	output	LVDS		2	
Diff_in_pad[29](n)	M25	input	LVDS		2	
Diff_in_pad[29]	M26	input	LVDS		2	
GND	N1	gnd				
GND+	N2				5	
GND+	N3				5	
Diff_out_pad[40]	N4	output	LVDS		5	
Diff_out_pad[40](n)	N5	output	LVDS		5	
Diff_out_pad[24]	N6	output	LVDS		5	
Diff_out_pad[24](n)	N7	output	LVDS		5	
GNDA_PLL4	N8	gnd				
GNDA_PLL4	N9	gnd				
VCCIO5	N10	power		2.5V	5	
GND	N11	gnd				
VCCINT	N12	power		1.2V		
GND	N13	gnd				
VCCINT	N14	power		1.2V		
GND	N15	gnd				
VCCINT	N16	power		1.2V		
GND	N17	gnd				
GNDA_PLL1	N18	gnd				
Diff_out_pad[18](n)	N19	output	LVDS		2	
Diff_out_pad[18]	N20	output	LVDS		2	
Diff_out_pad[0](n)	N21	output	LVDS		2	
Diff_out_pad[0]	N22	output	LVDS		2	
VREFB2	N23	power			2	
GND+	N24				2	
GND+	N25				2	
GND	N26	gnd				

GND	P1	gnd				
REF_CLOCK_pad[2]	P2	input	LVDS			5
REF_CLOCK_pad[2](n)	P3	input	LVDS			5
Diff_out_pad[28]	P4	output	LVDS			5
Diff_out_pad[28](n)	P5	output	LVDS			5
GND	P6	gnd				
VCCD_PLL3	P7	power		1.2V		
VCCA_PLL4	P8	power		1.2V		
VCCD_PLL4	P9	power		1.2V		
GND	P10	gnd				
VCCINT	P11	power		1.2V		
GND	P12	gnd				
VCCINT	P13	power		1.2V		
GND	P14	gnd				
VCCINT	P15	power		1.2V		
GND	P16	gnd				
VCCIO1	P17	power		2.5V	1	
GNDA_PLL1	P18	gnd				
VCCD_PLL1	P19	power		1.2V		
VCCD_PLL2	P20	power		1.2V		
VCCA_PLL1	P21	power		1.2V		
EXT_CLK128_in_pad(n)	P22	input	LVDS		1	
EXT_CLK128_in_pad	P23	input	LVDS		1	
REF_CLOCK_pad[1](n)	P24	input	LVDS		2	
REF_CLOCK_pad[1]	P25	input	LVDS		2	
GND	P26	gnd				
REF_CLOCK_pad[3]	R1	input	LVDS		6	
REF_CLOCK_pad[3](n)	R2	input	LVDS		6	
GND+	R3				6	
GND+	R4				6	
VREFB6	R5	power			6	
VCCA_PLL3	R6	power		1.2V		
GNDA_PLL3	R7	gnd				
GNDA_PLL3	R8	gnd				
VCCPD6	R9	power		3.3V	6	
VCCIO6	R10	power		2.5V	6	
GND	R11	gnd				
VCCINT	R12	power		1.2V		
GND	R13	gnd				
VCCINT	R14	power		1.2V		
GND	R15	gnd				
VCCINT	R16	power		1.2V		
GND	R17	gnd				
VCCPD1	R18	power		3.3V	1	
GNDA_PLL2	R19	gnd				
GNDA_PLL2	R20	gnd				
VCCA_PLL2	R21	power		1.2V		
GND	R22	gnd				
Diff_out_pad[30](n)	R23	output	LVDS		1	
Diff_out_pad[30]	R24	output	LVDS		1	
REF_CLOCK_pad[0](n)	R25	input	LVDS		1	
REF_CLOCK_pad[0]	R26	input	LVDS		1	
VCCIO6	T1	power		2.5V	6	
GND*	T2				6	
GND*	T3				6	
Clk_out_pad[6]	T4	output	LVDS		6	
Clk_out_pad[6](n)	T5	output	LVDS		6	
GND*	T6				6	
GND*	T7				6	
Diff_out_pad[35]	T8	output	LVDS		6	
Diff_out_pad[35](n)	T9	output	LVDS		6	
GND	T10	gnd				
VCCINT	T11	power		1.2V		

GND	T12	gnd				
VCCINT	T13	power		1.2V		
GND	T14	gnd				
VCCINT	T15	power		1.2V		
GND	T16	gnd				
VCCINT	T17	power		1.2V		
GND	T18	gnd				
Diff_out_pad[23](n)	T19	output	LVDS		1	
Diff_out_pad[23]	T20	output	LVDS		1	
Diff_out_pad[4](n)	T21	output	LVDS		1	
Diff_out_pad[4]	T22	output	LVDS		1	
VREFB1	T23	power			1	
Diff_in_pad[23](n)	T24	input	LVDS		1	
Diff_in_pad[23]	T25	input	LVDS		1	
VCCIO1	T26	power		2.5V	1	
Diff_in_pad[24]	U1	input	LVDS		6	
Diff_in_pad[24](n)	U2	input	LVDS		6	
toGige_Data_pad[0]	U3	output	LVDS		6	
toGige_Data_pad[0](n)	U4	output	LVDS		6	
toGigE_Clk_pad	U5	output	LVDS		6	
toGigE_Clk_pad(n)	U6	output	LVDS		6	
Diff_out_pad[42]	U7	output	LVDS		6	
Diff_out_pad[42](n)	U8	output	LVDS		6	
GND	U9	gnd				
VCCINT	U10	power		1.2V		
GND	U11	gnd				
VCCIO7	U12	power		1.8V	7	
GND	U13	gnd				
VCCIO8	U14	power		1.8V	8	
GND	U15	gnd				
VCCINT	U16	power		1.2V		
GND	U17	gnd				
VCCINT	U18	power		1.2V		
Diff_out_pad[32](n)	U19	output	LVDS		1	
Diff_out_pad[32]	U20	output	LVDS		1	
Diff_out_pad[20](n)	U21	output	LVDS		1	
Diff_out_pad[20]	U22	output	LVDS		1	
Diff_out_pad[6](n)	U23	output	LVDS		1	
Diff_out_pad[6]	U24	output	LVDS		1	
Diff_in_pad[21](n)	U25	input	LVDS		1	
Diff_in_pad[21]	U26	input	LVDS		1	
Diff_in_pad[15]	V1	input	LVDS		6	
Diff_in_pad[15](n)	V2	input	LVDS		6	
toGige_Data_pad[1]	V3	output	LVDS		6	
toGige_Data_pad[1](n)	V4	output	LVDS		6	
Clk_out_pad[7]	V5	output	LVDS		6	
Clk_out_pad[7](n)	V6	output	LVDS		6	
Diff_out_pad[47]	V7	output	LVDS		6	
Diff_out_pad[47](n)	V8	output	LVDS		6	
VCCINT	V9	power		1.2V		
DDROUT_pad[14]	V10	output	1.8 V		7	
VCCPD7	V11	power		3.3V	7	
GND*	V12				7	
VCC_PLL6_OUT	V13	power		3.3V	10	
DDROUT_pad[6]	V14	output	1.8 V		8	
VCCPD8	V15	power		3.3V	8	
DDRIN_pad[21]	V16	input	1.8 V		8	
DDRIN_pad[44]	V17	input	1.8 V		8	
DDRIN_pad[26]	V18	input	1.8 V		8	
Diff_out_pad[33](n)	V19	output	LVDS		1	
Diff_out_pad[33]	V20	output	LVDS		1	
Diff_out_pad[21](n)	V21	output	LVDS		1	
Diff_out_pad[21]	V22	output	LVDS		1	

```
Diff_out_pad[11](n)          : V23       : output : LVDS              :         : 1         :
Diff_out_pad[11]             : V24       : output : LVDS              :         : 1         :
Diff_in_pad[19](n)           : V25       : input  : LVDS              :         : 1         :
Diff_in_pad[19]              : V26       : input  : LVDS              :         : 1         :
Diff_in_pad[14]              : W1        : input  : LVDS              :         : 6         :
Diff_in_pad[14](n)           : W2        : input  : LVDS              :         : 6         :
GND*                         : W3        :        :                   :         : 6         :
GND*                         : W4        :        :                   :         : 6         :
Diff_out_pad[44]             : W5        : output : LVDS              :         : 6         :
Diff_out_pad[44](n)          : W6        : output : LVDS              :         : 6         :
Diff_out_pad[29]             : W7        : output : LVDS              :         : 6         :
Diff_out_pad[29](n)          : W8        : output : LVDS              :         : 6         :
DDRIN_pad[58]                : W9        : input  : 1.8 V             :         : 7         :
DDRIN_pad[14]                : W10       : input  : 1.8 V             :         : 7         :
DDROUT_pad[49]               : W11       : output : 1.8 V             :         : 7         :
DDROUT_pad[13]               : W12       : output : 1.8 V             :         : 7         :
GNDA_PLL6                    : W13       : gnd    :                   :         :           :
GNDA_PLL6                    : W14       : gnd    :                   :         :           :
RESET_pad_                   : W15       : input  : 1.8 V             :         : 8         :
MCB_DATA_pad[0]              : W16       : bidir  : 1.8 V             :         : 8         :
SPARE_IO_pad[17]             : W17       : input  : 1.8 V             :         : 8         :
DDROUT_pad[5]                : W18       : output : 1.8 V             :         : 8         :
Diff_out_pad[17](n)          : W19       : output : LVDS              :         : 1         :
Diff_out_pad[17]             : W20       : output : LVDS              :         : 1         :
Diff_out_pad[8](n)           : W21       : output : LVDS              :         : 1         :
Diff_out_pad[8]              : W22       : output : LVDS              :         : 1         :
Diff_in_pad[16](n)           : W23       : input  : LVDS              :         : 1         :
Diff_in_pad[16]              : W24       : input  : LVDS              :         : 1         :
Diff_in_pad[17](n)           : W25       : input  : LVDS              :         : 1         :
Diff_in_pad[17]              : W26       : input  : LVDS              :         : 1         :
Diff_in_pad[13]              : Y1        : input  : LVDS              :         : 6         :
Diff_in_pad[13](n)           : Y2        : input  : LVDS              :         : 6         :
GND*                         : Y3        :        :                   :         : 6         :
GND*                         : Y4        :        :                   :         : 6         :
VREFB6                       : Y5        : power  :                   :         : 6         :
Diff_out_pad[45]             : Y6        : output : LVDS              :         : 6         :
Diff_out_pad[45](n)          : Y7        : output : LVDS              :         : 6         :
PORSEL                       : Y8        :        :                   :         : 7         :
DDROUT_pad[70]               : Y9        : output : 1.8 V             :         : 7         :
DDRIN_pad[52]                : Y10       : input  : 1.8 V             :         : 7         :
DDROUT_pad[48]               : Y11       : output : 1.8 V             :         : 7         :
DDROUT_pad[47]               : Y12       : output : 1.8 V             :         : 7         :
VCCA_PLL6                    : Y13       : power  :                   : 1.2V    :           :
VCCD_PLL6                    : Y14       : power  :                   : 1.2V    :           :
DDROUT_pad[36]               : Y15       : output : 1.8 V             :         : 8         :
MCB_DATA_pad[1]              : Y16       : bidir  : 1.8 V             :         : 8         :
DDROUT_pad[4]                : Y17       : output : 1.8 V             :         : 8         :
DDROUT_pad[35]               : Y18       : output : 1.8 V             :         : 8         :
VCCSEL                       : Y19       :        :                   :         : 8         :
Diff_out_pad[9](n)           : Y20       : output : LVDS              :         : 1         :
Diff_out_pad[9]              : Y21       : output : LVDS              :         : 1         :
VREFB1                       : Y22       : power  :                   :         : 1         :
Diff_in_pad[10](n)           : Y23       : input  : LVDS              :         : 1         :
Diff_in_pad[10]              : Y24       : input  : LVDS              :         : 1         :
Diff_in_pad[11](n)           : Y25       : input  : LVDS              :         : 1         :

Diff_in_pad[11]              : Y26       : input  : LVDS              :         : 1         :
-------------------------------------------------------------------------------------------
 -- NC           : No Connect. This pin has no internal connection to the device.
 -- DNU          : Do Not Use. This pin MUST NOT be connected.
 -- VCCINT       : Dedicated power pin, which MUST be connected to VCC  (1.2V).
 -- VCCIO        : Dedicated power pin, which MUST be connected to VCC
 --                of its bank.
 --                              Bank 1:        2.5V
```

```
--                                  Bank 2:        2.5V
--                                  Bank 3:        1.8V
--                                  Bank 4:        1.8V
--                                  Bank 5:        2.5V
--                                  Bank 6:        2.5V
--                                  Bank 7:        1.8V
--                                  Bank 8:        1.8V
--                                  Bank 9:        3.3V
--                                  Bank 10:       3.3V
-- GND           : Dedicated ground pin. Dedicated GND pins MUST be connected to GND.
--
-- GND+          : Unused input pin. It can also be used to report unused dual-purpose pins.
--                                  This pin should be connected to GND.
-- GND*          : Unused  I/O  pin.
-- RESERVED      : Unused I/O pin, which MUST be left unconnected.
-- RESERVED_INPUT    : Pin is tri-stated and should be connected to the board.
-- RESERVED_INPUT_WITH_WEAK_PULLUP    : Pin is tri-stated with internal weak pull-up resistor.
-- RESERVED_INPUT_WITH_BUS_HOLD       : Pin is tri-stated with bus-hold circuitry.
-- RESERVED_OUTPUT_DRIVEN_HIGH        : Pin is output driven high.
--------------------------------------------------------------------------------
```

9.1.1 Upper RXP Pin Notes

The Upper RXP pin file contains a number of generically-labeled I/Os. This section contains tables that define I/O connectivity, within the context of the Baseline Board.

X-ERNI Row #	Wafer #	CTRL (pin name)	Pin #	BB0 (pin name)	Pin #	BB1 (pin name)	Pin #
1	0	Diff_in_pad[0]	AD1 AD2	Diff_in_pad[4]	AC25 AC24	Diff_in_pad[5]	AD26 AD25
2	1	Diff_in_pad[1]	AC2 AC3	Diff_in_pad[6]	AB24 AB23	Diff_in_pad[7]	AB26 AB25
3	2	Diff_in_pad[2]	AB3 AB4	Diff_in_pad[8]	AA24 AA23	Diff_in_pad[9]	AA26 AA25
4	3	Diff_in_pad[3]	AB1 AB2	Diff_in_pad[10]	Y24 Y23	Diff_in_pad[11]	Y26 Y25
5	4	Diff_in_pad[12]	AA1 AA2	Diff_in_pad[16]	W24 W23	Diff_in_pad[17]	W26 W25
6	5	Diff_in_pad[13]	Y1 Y2	Diff_in_pad[18]	L25 L24	Diff_in_pad[19]	V26 V25
7	6	Diff_in_pad[14]	W1 W2	Diff_in_pad[20]	K24 K23	Diff_in_pad[21]	U26 U25
8	7	Diff_in_pad[15]	V1 V2	Diff_in_pad[22]	J24 J23	Diff_in_pad[23]	T25 T24
9	8	Diff_in_pad[24]	U1 U2	Diff_in_pad[28]	H24 H23	Diff_in_pad[29]	M26 M25
10	9	Diff_in_pad[25]	M1 M2	Diff_in_pad[30]	G24 G23	Diff_in_pad[31]	K26 K25
11	10	Diff_in_pad[26]	K1 K2	Diff_in_pad[32]	F24 F23	Diff_in_pad[33]	J26 J25
12	11	Diff_in_pad[27]	J2 J1	Diff_in_pad[34]	E24 E23	Diff_in_pad[35]	H26 H25
13	12	Diff_in_pad[36]	L2 L3	Diff_in_pad[40]	D25 D24	Diff_in_pad[41]	G26 G25
14	13	Diff_in_pad[37]	G1 G2	Diff_in_pad[42]	H1 H2	Diff_in_pad[43]	F26 F25
15	14	Diff_in_pad[38]	F1 F2	Diff_in_pad[44]	D2 D3	Diff_in_pad[45]	E26 E25
16	15	Diff_in_pad[39]	E1 E2	Diff_in_pad[46]	C1 C2	Diff_in_pad[47]	C26 C25

Table 9-1 Upper RXP FPGA wafer input to "pin name" and device pin number assignments. The X-ERNI connector row # is to the left; X-ERNI row numbering starts at 1 at the top of the connector. Two pin numbers are assigned for each signal, as these are 1.024 Gbps LVDS inputs; the first pin number is the "+" line of the pair, the second is the "-" line. All signals use on-chip 100 ohm differential termination.

X/Y Recirc # [channel]	CTRL (pin name)	Pin #	BB0 (pin name)	Pin #	BB1 (pin name)	Pin #
0 [0]	Diff_out_pad[0]	N22 N21	Diff_out_pad[4]	T22 T21	Diff_out_pad[5]	AA22 AA21
0 [1]	Diff_out_pad[1]	J22 J21	Diff_out_pad[6]	U24 U23	Diff_out_pad[7]	L23 L22
0 [2]	Diff_out_pad[2]	H22 H21	Diff_out_pad[8]	W22 W21	Diff_out_pad[9]	Y21 Y20
0 [3]	Diff_out_pad[3]	K22 K21	Diff_out_pad[10]	M24 M23	Diff_out_pad[11]	V24 V23
1 [0]	Diff_out_pad[12]	L21 L20	Diff_out_pad[16]	M22 M21	Diff_out_pad[17]	W20 W19
1 [1]	Diff_out_pad[13]	G21 G20	Diff_out_pad[18]	N20 N19	Diff_out_pad[19]	J20 J19
1 [2]	Diff_out_pad[14]	F22 F21	Diff_out_pad[20]	U22 U21	Diff_out_pad[21]	V22 V21
1 [3]	Diff_out_pad[15]	H20 H19	Diff_out_pad[22]	K20 K19	Diff_out_pad[23]	T20 T19
2 [0]	Diff_out_pad[24]	N6 N7	Diff_out_pad[28]	P4 P5	Diff_out_pad[29]	W7 W8
2 [1]	Diff_out_pad[25]	J7 J8	Diff_out_pad[30]	R24 R23	Diff_out_pad[31]	L19 L18
2 [2]	Diff_out_pad[26]	H7 H8	Diff_out_pad[32]	U20 U19	Diff_out_pad[33]	V20 V19
2 [3]	Diff_out_pad[27]	K8 K9	Diff_out_pad[34]	M20 M19	Diff_out_pad[35]	T8 T9
3 [0]	Diff_out_pad[36]	M7 M8	Diff_out_pad[40]	N4 N5	Diff_out_pad[41]	AA5 AA6
3 [1]	Diff_out_pad[37]	H5 H6	Diff_out_pad[42]	U7 U8	Diff_out_pad[43]	K6 K7
3 [2]	Diff_out_pad[38]	G6 G7	Diff_out_pad[44]	W5 W6	Diff_out_pad[45]	Y6 Y7
3 [3]	Diff_out_pad[39]	J5 J6	Diff_out_pad[46]	L8 L9	Diff_out_pad[47]	V7 V8

Table 9-2 Upper RXP FPGA output pin number assignments, showing destination Recirculation FPGA numbers (rows-Y, columns-X) signals are destined for.

Note that within Recirculation FPGA nomenclature, including GUI, channel [0] is “BB0, BB1”, channel [1] is “BB2, BB3” etc.

Each RXP FPGA shares its input data from the ERNI connector to its companion RXP, via a number of single-ended 1.8 V (12 mA driver) 512 Mbps DDR data lines. Each wafer, consisting of the CTRL, BB0, and BB1 signals consumes and is multiplexed across 5 of these lines, and therefore each RXP FPGA has, at minimum 5 x 16 = 80 lines. In addition, due to signal integrity issues (cross-talk minimization), only lines originating from the same source clock domain are allowed to be on the same PCB layer (refer to section 9.3).

The following table shows which of the "DDROUT_pad[79:0]" signals exiting the Upper RXP FPGA are assigned to (i.e. source from) each input wafer. In addition, there are 24 spare I/O pads available for Upper to/from Lower RXP future use.

X-ERNI Row #	Wafer #	DDROUT_pad lines	Pins
1	0	4:0	Y17, AC15, AC19, AD21, AE24
2	1	9:5	AD7, AE6, AF3, V14, W18
3	2	14:10	V10, W12, AA8, AA11, AD10
4	3	19:15	G11, J9, K10, K16, H18
5	4	24:20	F15, B8, C5, D7, F11
6	5	29:25	C18, B21, C22, F20, G19
7	6	34:30	AA17, AB19, AD20, AF21, C17
8	7	39:35	AE9, AF10, AA14, Y15, Y18
9	8	44:40	AC6, AC8, AD5, AE4, AF7
10	9	49:45	W11, Y11, Y12, AA10, AB9
11	10	54:50	E15, G9, G10, G16, J17
12	11	59:55	D17, D20, E9, E11, E14
13	12	64:60	AF9, C7, A8, G18, B19
14	13	69:65	AA12, AB7, AB10, AF6, AE8
15	14	74:70	B5, E8, H10, K11, Y9
16	15	79:75	F13, C11, D10, B9, A6
		SPARE_IO_pad[23:0]	Refer to pin file. SPARE_IO_pad[13:0] can be used ONLY for Upper->Lower data direction, due to layer signal restrictions. (section 9.3)

Table 9-3 Upper RXP DDROUT_pad[79:0] wafer and pin assignments.

DDROUT_pad[79:0] signals exiting the Upper RXP, enter the Lower RXP on its DDRIN_pad[79:0] pins.

SPARE_IO_pad[13:0] signals exiting the Upper RXP, enter the Lower RXP on its SPARE_IO_pad[13:0] pins.

The following table shows the DDRIN_pad[79:0] pin and wafer assignments for the Upper RXP. These are connected to DDROUT_pad[79:0] pins leaving the Lower RXP.

X-ERNI Row #	Wafer #	DDRIN_pad lines	Pins
18	16	4:0	B7, D6, A5, B4, AD3
19	17	9:5	AD9, H9, AB8, C8, AC7
20	18	14:10	W10, B11, AE10, F10, D9
21	19	19:15	AB15, F14, F12, J11, AB11
22	20	24:20	AC18, F17, K17, V16, H15
23	21	29:25	AF20, E19, AE19, V18, D18
24	22	34:30	C23, B22, AF22, AC21, AB20
25	23	39:35	AC20, AD22, AE21, AE23, A24
26	24	44:40	V17, AD15, AB16, AA19, AB18
27	25	49:45	AC9, AD4, AE3, AD8, AF5
28	26	54:50	C6, AE7, Y10, AB12, AD11
29	27	59:55	B10, W9, C9, D8, A7
30	28	64:60	B16, G15, J14, E12, H11
31	29	69:65	C20, D19, E18, J18, G17
32	30	74:70	E7, B6, AD6, AE5, A3
33	31	79:75	E10, C10, AC10, F8, AF8

Table 9-4 Upper RXP DDRIN_pad wafer and pin assignments. These are inputs with data sourcing from the Lower RXP.

The following table contains a list of miscellaneous signals, their functions, and pin assignments for the Upper RXP FPGA.

Signal	Pin(s)	Description/Function
REF_CLOCK_pad[0]	R26, R25	Primary 128 MHz wafer clock input. This sources from wafer 0 (ERNI connector top row).
REF_CLOCK_pad[3:1]	(R1,R2) (P2,P3) (P25,P24)	Auxiliary 128 MHz wafer clock inputs. These are not used in the FPGA, but may be with compile-time selection. These source from wafers 12, 8, and 4 respectively.
EXT_CLK128_in_pad	P23, P22	External 128 MHz input; sources from the board's SMA external clock input connector.
DDROUT_CLK_pad	J10	256 MHz clock output synchronous with DDROUT_pad signals.
DDRIN_CLK_pad	B13	256 MHz clock input synchronous with DDRIN_pad signals.
toGige_Data_pad[1:0]	*see pin file*	[1]=encoded VDIF output packets; [0]=idle codes
toGigE_Clk_pad	U5, U6	128 MHz clock out to GigE FPGA
Phasing_out_pad[3:0]	*see pin file*	[3]=4-bit Phased output in HM Gbps format; [2]=0; [1]=HM Gbps CTRL; [0]=128 MHz clock. Out on ERNI row 34.
Clk_out_pad[7:0]	*see pin file*	8, 128 MHz clocks to X/Y0:3 Recirc FPGAs
RESET_pad_	W15	Active-low chip reset, resets all regs on the chip. Does not de-configure
MCB_CLK_pad	AA16	33 MHz CPU clock; inputs have a 15 nsec t_{su} requirement
MCB_ADDR_pad[3:0]	*see pin file*	4-bit CPU address bus
MCB_DATA_pad[7:0]	*see pin file*	8-bit bi-directional CPU data bus; on read, there is a 25 nsec MCB_CLK_pad t_{co} delay.
MCB_CS_pad_	AE16	Active-low CPU chip select
MCB_RD_WR_pad_	AC17	Active-low write (read=high)
testmode_tick_out_pad	A20	10 msec output time tick for testmode operation (sync U/L RXPs)
testmode_tick_in_pad	A21	Unused in the Upper RXP
testmode_1pps_out_pad	AD23	1 sec output time tick for testmode operation (sync U/L RXPs)
testmode_1pps_in_pad	AF24	Unused in the Upper RXP
testmode_dummy_pad	*see pin file*	Unused, connect high or low or leave unconnected

Table 9-5 Upper RXP miscellaneous signals' descriptions.

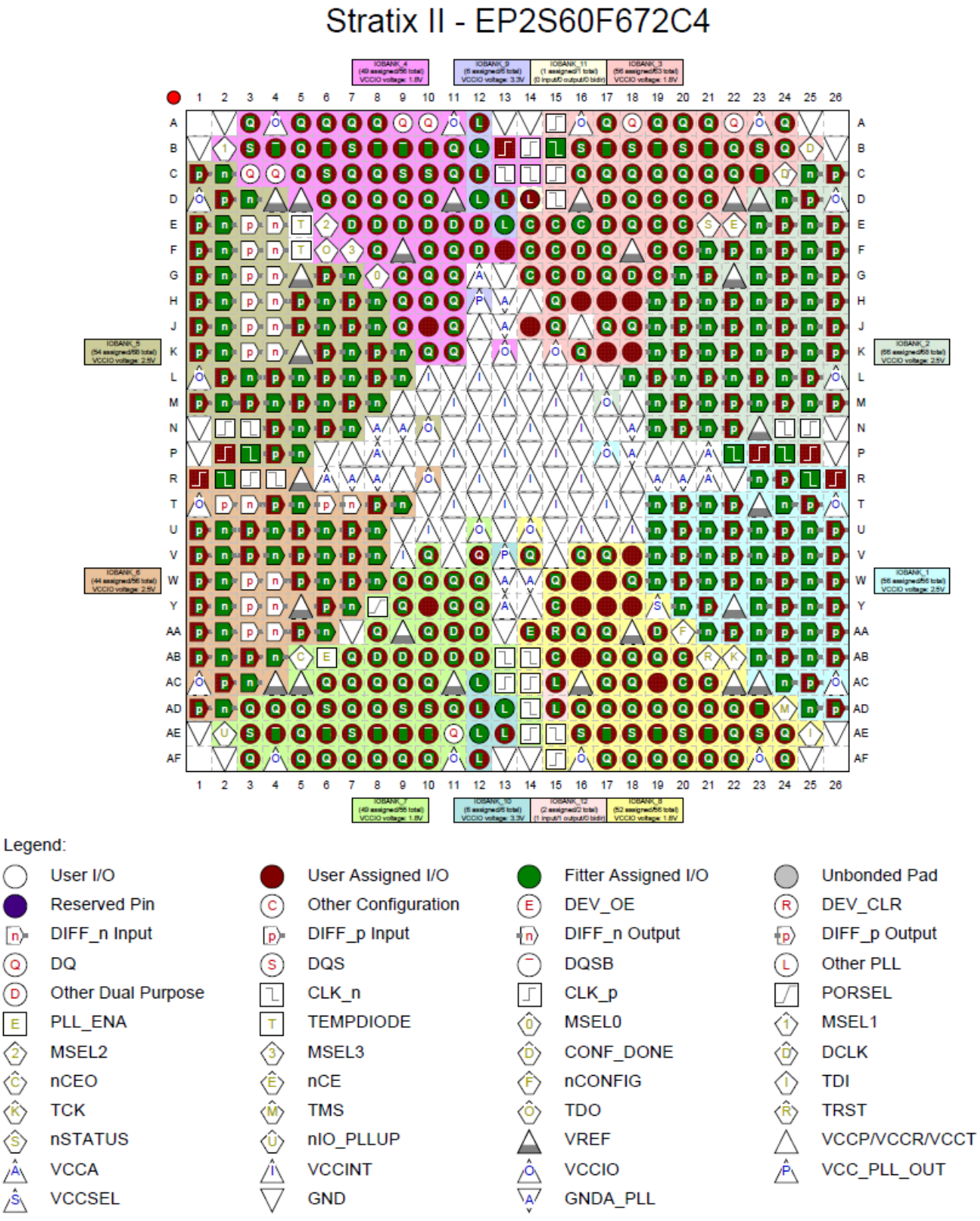


Figure 9-1 Upper RXP Altera Quartus-II pin planner output. Top view of the chip.

9.2 Lower RXP FPGA Pinouts

Pin Name/Usage	Location	Dir.	I/O Standard	Voltage	I/O Bank
GND	A2	gnd			
DDROUT_pad[70]	A3	output	1.8 V		4
VCCIO4	A4	power		1.8V	4
DDROUT_pad[2]	A5	output	1.8 V		4
DDRIN_pad[75]	A6	input	1.8 V		4
DDROUT_pad[55]	A7	output	1.8 V		4
DDRIN_pad[62]	A8	input	1.8 V		4
GND*	A9				4
GND*	A10				4
VCCIO4	A11	power		1.8V	4
Clk_out_pad[3]	A12	output	LVDS		9
GND	A13	gnd			
GND	A14	gnd			
GND*	A15				3
VCCIO3	A16	power		1.8V	3
SPARE_IO_pad[18]	A17	input	1.8 V		3
GND*	A18				3
SPARE_IO_pad[8]	A19	input	1.8 V		3
testmode_tick_in_pad	A20	input	1.8 V		3
testmode_tick_out_pad	A21	output	1.8 V		3
GND*	A22				3
VCCIO3	A23	power		1.8V	3
DDROUT_pad[35]	A24	output	1.8 V		3
GND	A25	gnd			
Diff_in_pad[12]	AA1	input	LVDS		6
Diff_in_pad[12](n)	AA2	input	LVDS		6
GND*	AA3				6
GND*	AA4				6
Diff_out_pad[38]	AA5	output	LVDS		6
Diff_out_pad[38](n)	AA6	output	LVDS		6
GND	AA7	gnd			
DDRIN_pad[12]	AA8	input	1.8 V		7
VREFB7	AA9	power			7
DDRIN_pad[46]	AA10	input	1.8 V		7
DDRIN_pad[11]	AA11	input	1.8 V		7
DDRIN_pad[69]	AA12	input	1.8 V		7
GND	AA13	gnd			
DDRIN_pad[37]	AA14	input	1.8 V		8
SPARE_IO_pad[15]	AA15	input	1.8 V		8
MCB_CLK_pad	AA16	input	1.8 V		8
DDRIN_pad[34]	AA17	input	1.8 V		8
VREFB8	AA18	power			8
DDROUT_pad[41]	AA19	output	1.8 V		8
nCONFIG	AA20				8
Diff_out_pad[5](n)	AA21	output	LVDS		1
Diff_out_pad[5]	AA22	output	LVDS		1
Diff_in_pad[8](n)	AA23	input	LVDS		1
Diff_in_pad[8]	AA24	input	LVDS		1
Diff_in_pad[9](n)	AA25	input	LVDS		1
Diff_in_pad[9]	AA26	input	LVDS		1
Diff_in_pad[3]	AB1	input	LVDS		6
Diff_in_pad[3](n)	AB2	input	LVDS		6
Diff_in_pad[2]	AB3	input	LVDS		6
Diff_in_pad[2](n)	AB4	input	LVDS		6
nCEO	AB5				7
PLL_ENA	AB6				7
DDRIN_pad[68]	AB7	input	1.8 V		7

DDROUT_pad[7]	AB8	output	1.8 V		7	
DDRIN_pad[45]	AB9	input	1.8 V		7	
DDRIN_pad[67]	AB10	input	1.8 V		7	
DDROUT_pad[15]	AB11	output	1.8 V		7	
DDROUT_pad[51]	AB12	output	1.8 V		7	
GND*	AB13				7	
GND*	AB14				8	
DDROUT_pad[19]	AB15	output	1.8 V		8	
DDROUT_pad[42]	AB16	output	1.8 V		8	
MCB_DATA_pad[2]	AB17	bidir	1.8 V		8	
DDROUT_pad[40]	AB18	output	1.8 V		8	
DDRIN_pad[33]	AB19	input	1.8 V		8	
DDROUT_pad[30]	AB20	output	1.8 V		8	
TRST	AB21	input			8	
TCK	AB22	input			8	
Diff_in_pad[6](n)	AB23	input	LVDS		1	
Diff_in_pad[6]	AB24	input	LVDS		1	
Diff_in_pad[7](n)	AB25	input	LVDS		1	
Diff_in_pad[7]	AB26	input	LVDS		1	
VCCIO6	AC1	power		2.5V	6	
Diff_in_pad[1]	AC2	input	LVDS		6	
Diff_in_pad[1](n)	AC3	input	LVDS		6	
VREFB6	AC4	power			6	
VREFB7	AC5	power			7	
DDRIN_pad[44]	AC6	input	1.8 V		7	
DDROUT_pad[5]	AC7	output	1.8 V		7	
DDRIN_pad[43]	AC8	input	1.8 V		7	
DDROUT_pad[49]	AC9	output	1.8 V		7	
DDROUT_pad[77]	AC10	output	1.8 V		7	
VREFB7	AC11	power			7	
Clk_out_pad[5](n)	AC12	output	LVDS		10	
GND*	AC13				7	
GND*	AC14				8	
DDRIN_pad[3]	AC15	input	1.8 V		12	
VREFB8	AC16	power			8	
MCB_RD_WR_pad_	AC17	input	1.8 V		8	
DDROUT_pad[24]	AC18	output	1.8 V		8	
DDRIN_pad[2]	AC19	input	1.8 V		8	
DDROUT_pad[39]	AC20	output	1.8 V		8	
DDROUT_pad[31]	AC21	output	1.8 V		8	
VREFB8	AC22	power			8	
VREFB1	AC23	power			1	
Diff_in_pad[4](n)	AC24	input	LVDS		1	
Diff_in_pad[4]	AC25	input	LVDS		1	
VCCIO1	AC26	power		2.5V	1	
Diff_in_pad[0]	AD1	input	LVDS		6	
Diff_in_pad[0](n)	AD2	input	LVDS		6	
DDROUT_pad[0]	AD3	output	1.8 V		7	
DDROUT_pad[48]	AD4	output	1.8 V		7	
DDRIN_pad[42]	AD5	input	1.8 V		7	
DDROUT_pad[72]	AD6	output	1.8 V		7	
DDRIN_pad[9]	AD7	input	1.8 V		7	
DDROUT_pad[46]	AD8	output	1.8 V		7	
DDROUT_pad[9]	AD9	output	1.8 V		7	
DDRIN_pad[10]	AD10	input	1.8 V		7	
DDROUT_pad[50]	AD11	output	1.8 V		7	
Clk_out_pad[5]	AD12	output	LVDS		10	
Clk_out_pad[0](n)	AD13	output	LVDS		10	
GND*	AD14				7	
DDROUT_pad[43]	AD15	output	1.8 V		12	
MCB_DATA_pad[7]	AD16	bidir	1.8 V		8	
MCB_DATA_pad[5]	AD17	bidir	1.8 V		8	
MCB_ADDR_pad[1]	AD18	input	1.8 V		8	

MCB_ADDR_pad[0]	AD19	input	1.8 V		8	
DDRIN_pad[32]	AD20	input	1.8 V		8	
DDRIN_pad[1]	AD21	input	1.8 V		8	
DDROUT_pad[38]	AD22	output	1.8 V		8	
testmode_1pps_in_pad	AD23	input	1.8 V		8	
TMS	AD24	input			8	
Diff_in_pad[5](n)	AD25	input	LVDS		1	
Diff_in_pad[5]	AD26	input	LVDS		1	
GND	AE1	gnd				
nIO_PULLUP	AE2				7	
DDROUT_pad[47]	AE3	output	1.8 V		7	
DDRIN_pad[41]	AE4	input	1.8 V		7	
DDROUT_pad[71]	AE5	output	1.8 V		7	
DDRIN_pad[8]	AE6	input	1.8 V		7	
DDROUT_pad[53]	AE7	output	1.8 V		7	
DDRIN_pad[65]	AE8	input	1.8 V		7	
DDRIN_pad[39]	AE9	input	1.8 V		7	
DDROUT_pad[12]	AE10	output	1.8 V		7	
GND*	AE11				7	
Clk_out_pad[4](n)	AE12	output	LVDS		10	
Clk_out_pad[0]	AE13	output	LVDS		10	
GND*	AE14				7	
GND*	AE15				8	
MCB_CS_pad_	AE16	input	1.8 V		8	
MCB_DATA_pad[4]	AE17	bidir	1.8 V		8	
MCB_ADDR_pad[3]	AE18	input	1.8 V		8	
DDROUT_pad[27]	AE19	output	1.8 V		8	
SPARE_IO_pad[21]	AE20	input	1.8 V		8	
DDROUT_pad[37]	AE21	output	1.8 V		8	
SPARE_IO_pad[23]	AE22	input	1.8 V		8	
DDROUT_pad[36]	AE23	output	1.8 V		8	
DDRIN_pad[0]	AE24	input	1.8 V		8	
TDI	AE25	input			8	
GND	AE26	gnd				
GND	AF2	gnd				
DDRIN_pad[7]	AF3	input	1.8 V		7	
VCCIO7	AF4	power		1.8V	7	
DDROUT_pad[45]	AF5	output	1.8 V		7	
DDRIN_pad[66]	AF6	input	1.8 V		7	
DDRIN_pad[40]	AF7	input	1.8 V		7	
DDROUT_pad[75]	AF8	output	1.8 V		7	
DDRIN_pad[64]	AF9	input	1.8 V		7	
DDRIN_pad[38]	AF10	input	1.8 V		7	
VCCIO7	AF11	power		1.8V	7	
Clk_out_pad[4]	AF12	output	LVDS		10	
GND	AF13	gnd				
GND	AF14	gnd				
GND*	AF15				8	
VCCIO8	AF16	power		1.8V	8	
MCB_DATA_pad[6]	AF17	bidir	1.8 V		8	
MCB_DATA_pad[3]	AF18	bidir	1.8 V		8	
MCB_ADDR_pad[2]	AF19	input	1.8 V		8	
DDROUT_pad[29]	AF20	output	1.8 V		8	
DDRIN_pad[31]	AF21	input	1.8 V		8	
DDROUT_pad[32]	AF22	output	1.8 V		8	
VCCIO8	AF23	power		1.8V	8	
testmode_1pps_out_pad	AF24	output	1.8 V		8	
GND	AF25	gnd				
GND	B1	gnd				
MSEL1	B2				4	
fromGigE_XonXoff_pad	B3	input	1.8 V		4	
DDROUT_pad[1]	B4	output	1.8 V		4	
DDRIN_pad[74]	B5	input	1.8 V		4	

DDROUT_pad[73]	B6	output	1.8 V		4	
DDROUT_pad[4]	B7	output	1.8 V		4	
DDRIN_pad[23]	B8	input	1.8 V		4	
DDRIN_pad[76]	B9	input	1.8 V		4	
DDROUT_pad[59]	B10	output	1.8 V		4	
DDROUT_pad[13]	B11	output	1.8 V		4	
Clk_out_pad[3](n)	B12	output	LVDS		9	
DDRIN_CLK_pad	B13	input	1.8 V		4	
GND*	B14				4	
testmode_dummy_pad	B15	input	1.8 V		3	
DDROUT_pad[64]	B16	output	1.8 V		3	
SPARE_IO_pad[4]	B17	input	1.8 V		3	
SPARE_IO_pad[5]	B18	input	1.8 V		3	
DDRIN_pad[60]	B19	input	1.8 V		3	
SPARE_IO_pad[10]	B20	input	1.8 V		3	
DDRIN_pad[28]	B21	input	1.8 V		3	
DDROUT_pad[33]	B22	output	1.8 V		3	
SPARE_IO_pad[12]	B23	input	1.8 V		3	
SPARE_IO_pad[13]	B24	input	1.8 V		3	
CONF_DONE	B25				3	
GND	B26	gnd				
Diff_in_pad[46]	C1	input	LVDS		5	
Diff_in_pad[46](n)	C2	input	LVDS		5	
GND*	C3				4	
GND*	C4				4	
DDRIN_pad[22]	C5	input	1.8 V		4	
DDROUT_pad[54]	C6	output	1.8 V		4	
DDRIN_pad[63]	C7	input	1.8 V		4	
DDROUT_pad[6]	C8	output	1.8 V		4	
DDROUT_pad[57]	C9	output	1.8 V		4	
DDROUT_pad[78]	C10	output	1.8 V		4	
DDRIN_pad[78]	C11	input	1.8 V		4	
Clk_out_pad[2]	C12	output	LVDS		9	
GND*	C13				4	
GND*	C14				4	
GND*	C15				3	
SPARE_IO_pad[2]	C16	input	1.8 V		3	
DDRIN_pad[30]	C17	input	1.8 V		3	
DDRIN_pad[29]	C18	input	1.8 V		3	
SPARE_IO_pad[7]	C19	input	1.8 V		3	
DDROUT_pad[69]	C20	output	1.8 V		3	
SPARE_IO_pad[22]	C21	input	1.8 V		3	
DDRIN_pad[27]	C22	input	1.8 V		3	
DDROUT_pad[34]	C23	output	1.8 V		3	
DCLK	C24				3	
Diff_in_pad[47](n)	C25	input	LVDS		2	
Diff_in_pad[47]	C26	input	LVDS		2	
VCCIO5	D1	power		2.5V	5	
Diff_in_pad[44]	D2	input	LVDS		5	
Diff_in_pad[44](n)	D3	input	LVDS		5	
VREFB5	D4	power			5	
VREFB4	D5	power			4	
DDROUT_pad[3]	D6	output	1.8 V		4	
DDRIN_pad[21]	D7	input	1.8 V		4	
DDROUT_pad[56]	D8	output	1.8 V		4	
DDROUT_pad[10]	D9	output	1.8 V		4	
DDRIN_pad[77]	D10	input	1.8 V		4	
VREFB4	D11	power			4	
Clk_out_pad[2](n)	D12	output	LVDS		9	
Clk_out_pad[1]	D13	output	LVDS		9	
GND*	D14				11	
GND*	D15				3	
VREFB3	D16	power			3	

DDRIN_pad[59]	D17	input	1.8 V		3
DDROUT_pad[25]	D18	output	1.8 V		3
DDROUT_pad[68]	D19	output	1.8 V		3
DDRIN_pad[58]	D20	input	1.8 V		3
SPARE_IO_pad[11]	D21	input	1.8 V		3
VREFB3	D22	power			3
VREFB2	D23	power			2
Diff_in_pad[40](n)	D24	input	LVDS		2
Diff_in_pad[40]	D25	input	LVDS		2
VCCIO2	D26	power		2.5V	2
Diff_in_pad[39]	E1	input	LVDS		5
Diff_in_pad[39](n)	E2	input	LVDS		5
GND*	E3				5
GND*	E4				5
TEMPDIODEp	E5				
MSEL2	E6				4
DDROUT_pad[74]	E7	output	1.8 V		4
DDRIN_pad[73]	E8	input	1.8 V		4
DDRIN_pad[57]	E9	input	1.8 V		4
DDROUT_pad[79]	E10	output	1.8 V		4
DDRIN_pad[56]	E11	input	1.8 V		4
DDROUT_pad[61]	E12	output	1.8 V		4
Clk_out_pad[1](n)	E13	output	LVDS		9
DDRIN_pad[55]	E14	input	1.8 V		3
DDRIN_pad[54]	E15	input	1.8 V		3
~DATA0~ / RESERVED_INPUT	E16	input	1.8 V		3
SPARE_IO_pad[3]	E17	input	1.8 V		3
DDROUT_pad[67]	E18	output	1.8 V		3
DDROUT_pad[28]	E19	output	1.8 V		3
SPARE_IO_pad[9]	E20	input	1.8 V		3
nSTATUS	E21				3
nCE	E22				3
Diff_in_pad[34](n)	E23	input	LVDS		2
Diff_in_pad[34]	E24	input	LVDS		2
Diff_in_pad[45](n)	E25	input	LVDS		2
Diff_in_pad[45]	E26	input	LVDS		2
Diff_in_pad[38]	F1	input	LVDS		5
Diff_in_pad[38](n)	F2	input	LVDS		5
GND*	F3				5
GND*	F4				5
TEMPDIODEn	F5				
TDO	F6	output			4
MSEL3	F7				4
DDROUT_pad[76]	F8	output	1.8 V		4
VREFB4	F9	power			4
DDROUT_pad[11]	F10	output	1.8 V		4
DDRIN_pad[20]	F11	input	1.8 V		4
DDROUT_pad[17]	F12	output	1.8 V		4
DDRIN_pad[79]	F13	input	1.8 V		4
DDROUT_pad[18]	F14	output	1.8 V		3
DDRIN_pad[24]	F15	input	1.8 V		3
SPARE_IO_pad[16]	F16	input	1.8 V		3
DDROUT_pad[23]	F17	output	1.8 V		3
VREFB3	F18	power			3
SPARE_IO_pad[6]	F19	input	1.8 V		3
DDRIN_pad[26]	F20	input	1.8 V		3
Diff_out_pad[14](n)	F21	output	LVDS		2
Diff_out_pad[14]	F22	output	LVDS		2
Diff_in_pad[32](n)	F23	input	LVDS		2
Diff_in_pad[32]	F24	input	LVDS		2
Diff_in_pad[43](n)	F25	input	LVDS		2
Diff_in_pad[43]	F26	input	LVDS		2
Diff_in_pad[37]	G1	input	LVDS		5

Diff_in_pad[37](n)	G2	input	LVDS		5	
GND*	G3				5	
GND*	G4				5	
VREFB5	G5	power			5	
Diff_out_pad[41]	G6	output	LVDS		5	
Diff_out_pad[41](n)	G7	output	LVDS		5	
MSEL0	G8				4	
DDRIN_pad[53]	G9	input	1.8 V		4	
DDRIN_pad[52]	G10	input	1.8 V		4	
DDRIN_pad[19]	G11	input	1.8 V		4	
VCCA_PLL5	G12	power		1.2V		
GND	G13	gnd				
SPARE_IO_pad[14]	G14	input	1.8 V		3	
DDROUT_pad[63]	G15	output	1.8 V		3	
DDRIN_pad[51]	G16	input	1.8 V		3	
DDROUT_pad[65]	G17	output	1.8 V		3	
DDRIN_pad[61]	G18	input	1.8 V		3	
DDRIN_pad[25]	G19	input	1.8 V		3	
Diff_out_pad[13](n)	G20	output	LVDS		2	
Diff_out_pad[13]	G21	output	LVDS		2	
VREFB2	G22	power			2	
Diff_in_pad[30](n)	G23	input	LVDS		2	
Diff_in_pad[30]	G24	input	LVDS		2	
Diff_in_pad[41](n)	G25	input	LVDS		2	
Diff_in_pad[41]	G26	input	LVDS		2	
Diff_in_pad[42]	H1	input	LVDS		5	
Diff_in_pad[42](n)	H2	input	LVDS		5	
GND*	H3				5	
GND*	H4				5	
Diff_out_pad[45]	H5	output	LVDS		5	
Diff_out_pad[45](n)	H6	output	LVDS		5	
Diff_out_pad[26]	H7	output	LVDS		5	
Diff_out_pad[26](n)	H8	output	LVDS		5	
DDROUT_pad[8]	H9	output	1.8 V		4	
DDRIN_pad[72]	H10	input	1.8 V		4	
DDROUT_pad[60]	H11	output	1.8 V		4	
VCC_PLL5_OUT	H12	power		3.3V	9	
GNDA_PLL5	H13	gnd				
VCCD_PLL5	H14	power		1.2V		
DDROUT_pad[20]	H15	output	1.8 V		3	
SPARE_IO_pad[1]	H16	input	1.8 V		3	
SPARE_IO_pad[19]	H17	input	1.8 V		3	
DDRIN_pad[15]	H18	input	1.8 V		3	
Diff_out_pad[15](n)	H19	output	LVDS		2	
Diff_out_pad[15]	H20	output	LVDS		2	
Diff_out_pad[2](n)	H21	output	LVDS		2	
Diff_out_pad[2]	H22	output	LVDS		2	
Diff_in_pad[28](n)	H23	input	LVDS		2	
Diff_in_pad[28]	H24	input	LVDS		2	
Diff_in_pad[35](n)	H25	input	LVDS		2	
Diff_in_pad[35]	H26	input	LVDS		2	
Diff_in_pad[27]	J1	input	LVDS		5	
Diff_in_pad[27](n)	J2	input	LVDS		5	
GND*	J3				5	
GND*	J4				5	
Diff_out_pad[44]	J5	output	LVDS		5	
Diff_out_pad[44](n)	J6	output	LVDS		5	
Diff_out_pad[25]	J7	output	LVDS		5	
Diff_out_pad[25](n)	J8	output	LVDS		5	
DDRIN_pad[18]	J9	input	1.8 V		4	
DDROUT_CLK_pad	J10	output	1.8 V		4	
DDROUT_pad[16]	J11	output	1.8 V		4	
VCCPD4	J12	power		3.3V	4	

GNDA_PLL5	J13	gnd				
DDROUT_pad[62]	J14	output	1.8 V		3	
SPARE_IO_pad[0]	J15	input	1.8 V		3	
VCCPD3	J16	power		3.3V	3	
DDRIN_pad[50]	J17	input	1.8 V		3	
DDROUT_pad[66]	J18	output	1.8 V		3	
Diff_out_pad[19](n)	J19	output	LVDS		2	
Diff_out_pad[19]	J20	output	LVDS		2	
Diff_out_pad[1](n)	J21	output	LVDS		2	
Diff_out_pad[1]	J22	output	LVDS		2	
Diff_in_pad[22](n)	J23	input	LVDS		2	
Diff_in_pad[22]	J24	input	LVDS		2	
Diff_in_pad[33](n)	J25	input	LVDS		2	
Diff_in_pad[33]	J26	input	LVDS		2	
Diff_in_pad[26]	K1	input	LVDS		5	
Diff_in_pad[26](n)	K2	input	LVDS		5	
GND*	K3				5	
GND*	K4				5	
VREFB5	K5	power			5	
Diff_out_pad[47]	K6	output	LVDS		5	
Diff_out_pad[47](n)	K7	output	LVDS		5	
Diff_out_pad[27]	K8	output	LVDS		5	
Diff_out_pad[27](n)	K9	output	LVDS		5	
DDRIN_pad[17]	K10	input	1.8 V		4	
DDRIN_pad[71]	K11	input	1.8 V		4	
GND	K12	gnd				
VCCIO4	K13	power		1.8V	4	
GND	K14	gnd				
VCCIO3	K15	power		1.8V	3	
DDRIN_pad[16]	K16	input	1.8 V		3	
DDROUT_pad[22]	K17	output	1.8 V		3	
SPARE_IO_pad[20]	K18	input	1.8 V		3	
Diff_out_pad[22](n)	K19	output	LVDS		2	
Diff_out_pad[22]	K20	output	LVDS		2	
Diff_out_pad[3](n)	K21	output	LVDS		2	
Diff_out_pad[3]	K22	output	LVDS		2	
Diff_in_pad[20](n)	K23	input	LVDS		2	
Diff_in_pad[20]	K24	input	LVDS		2	
Diff_in_pad[31](n)	K25	input	LVDS		2	
Diff_in_pad[31]	K26	input	LVDS		2	
VCCIO5	L1	power		2.5V	5	
Diff_in_pad[36]	L2	input	LVDS		5	
Diff_in_pad[36](n)	L3	input	LVDS		5	
Phasing_out_pad[2]	L4	output	LVDS		5	
Phasing_out_pad[2](n)	L5	output	LVDS		5	
Phasing_out_pad[1]	L6	output	LVDS		5	
Phasing_out_pad[1](n)	L7	output	LVDS		5	
Diff_out_pad[42]	L8	output	LVDS		5	
Diff_out_pad[42](n)	L9	output	LVDS		5	
VCCINT	L10	power		1.2V		
GND	L11	gnd				
VCCINT	L12	power		1.2V		
GND	L13	gnd				
VCCINT	L14	power		1.2V		
GND	L15	gnd				
VCCINT	L16	power		1.2V		
GND	L17	gnd				
Diff_out_pad[31](n)	L18	output	LVDS		2	
Diff_out_pad[31]	L19	output	LVDS		2	
Diff_out_pad[12](n)	L20	output	LVDS		2	
Diff_out_pad[12]	L21	output	LVDS		2	
Diff_out_pad[7](n)	L22	output	LVDS		2	
Diff_out_pad[7]	L23	output	LVDS		2	

Diff_in_pad[18](n)	L24	input	LVDS			2
Diff_in_pad[18]	L25	input	LVDS			2
VCCIO2	L26	power			2.5V	2
Diff_in_pad[25]	M1	input	LVDS			5
Diff_in_pad[25](n)	M2	input	LVDS			5
Phasing_out_pad[3]	M3	output	LVDS			5
Phasing_out_pad[3](n)	M4	output	LVDS			5
Phasing_out_pad[0]	M5	output	LVDS			5
Phasing_out_pad[0](n)	M6	output	LVDS			5
Diff_out_pad[40]	M7	output	LVDS			5
Diff_out_pad[40](n)	M8	output	LVDS			5
VCCPD5	M9	power			3.3V	5
GND	M10	gnd				
VCCINT	M11	power			1.2V	
GND	M12	gnd				
VCCINT	M13	power			1.2V	
GND	M14	gnd				
VCCINT	M15	power			1.2V	
GND	M16	gnd				
VCCIO2	M17	power			2.5V	2
VCCPD2	M18	power			3.3V	2
Diff_out_pad[34](n)	M19	output	LVDS			2
Diff_out_pad[34]	M20	output	LVDS			2
Diff_out_pad[16](n)	M21	output	LVDS			2
Diff_out_pad[16]	M22	output	LVDS			2
Diff_out_pad[10](n)	M23	output	LVDS			2
Diff_out_pad[10]	M24	output	LVDS			2
Diff_in_pad[29](n)	M25	input	LVDS			2
Diff_in_pad[29]	M26	input	LVDS			2
GND	N1	gnd				
GND+	N2					5
GND+	N3					5
Diff_out_pad[36]	N4	output	LVDS			5
Diff_out_pad[36](n)	N5	output	LVDS			5
Diff_out_pad[24]	N6	output	LVDS			5
Diff_out_pad[24](n)	N7	output	LVDS			5
GNDA_PLL4	N8	gnd				
GNDA_PLL4	N9	gnd				
VCCIO5	N10	power			2.5V	5
GND	N11	gnd				
VCCINT	N12	power			1.2V	
GND	N13	gnd				
VCCINT	N14	power			1.2V	
GND	N15	gnd				
VCCINT	N16	power			1.2V	
GND	N17	gnd				
GNDA_PLL1	N18	gnd				
Diff_out_pad[18](n)	N19	output	LVDS			2
Diff_out_pad[18]	N20	output	LVDS			2
Diff_out_pad[0](n)	N21	output	LVDS			2
Diff_out_pad[0]	N22	output	LVDS			2
VREFB2	N23	power				2
GND+	N24					2
GND+	N25					2
GND	N26	gnd				
GND	P1	gnd				
REF_CLOCK_pad[2]	P2	input	LVDS			5
REF_CLOCK_pad[2](n)	P3	input	LVDS			5
Diff_out_pad[28]	P4	output	LVDS			5
Diff_out_pad[28](n)	P5	output	LVDS			5
GND	P6	gnd				
VCCD_PLL3	P7	power			1.2V	
VCCA_PLL4	P8	power			1.2V	

VCCD_PLL4	P9	power		1.2V		
GND	P10	gnd				
VCCINT	P11	power		1.2V		
GND	P12	gnd				
VCCINT	P13	power		1.2V		
GND	P14	gnd				
VCCINT	P15	power		1.2V		
GND	P16	gnd				
VCCIO1	P17	power		2.5V	1	
GNDA_PLL1	P18	gnd				
VCCD_PLL1	P19	power		1.2V		
VCCD_PLL2	P20	power		1.2V		
VCCA_PLL1	P21	power		1.2V		
EXT_CLK128_in_pad(n)	P22	input	LVDS		1	
EXT_CLK128_in_pad	P23	input	LVDS		1	
REF_CLOCK_pad[1](n)	P24	input	LVDS		2	
REF_CLOCK_pad[1]	P25	input	LVDS		2	
GND	P26	gnd				
REF_CLOCK_pad[3]	R1	input	LVDS		6	
REF_CLOCK_pad[3](n)	R2	input	LVDS		6	
GND+	R3				6	
GND+	R4				6	
VREFB6	R5	power			6	
VCCA_PLL3	R6	power		1.2V		
GNDA_PLL3	R7	gnd				
GNDA_PLL3	R8	gnd				
VCCPD6	R9	power		3.3V	6	
VCCIO6	R10	power		2.5V	6	
GND	R11	gnd				
VCCINT	R12	power		1.2V		
GND	R13	gnd				
VCCINT	R14	power		1.2V		
GND	R15	gnd				
VCCINT	R16	power		1.2V		
GND	R17	gnd				
VCCPD1	R18	power		3.3V	1	
GNDA_PLL2	R19	gnd				
GNDA_PLL2	R20	gnd				
VCCA_PLL2	R21	power		1.2V		
GND	R22	gnd				
Diff_out_pad[30](n)	R23	output	LVDS		1	
Diff_out_pad[30]	R24	output	LVDS		1	
REF_CLOCK_pad[0](n)	R25	input	LVDS		1	
REF_CLOCK_pad[0]	R26	input	LVDS		1	
VCCIO6	T1	power		2.5V	6	
GND*	T2				6	
GND*	T3				6	
Clk_out_pad[6]	T4	output	LVDS		6	
Clk_out_pad[6](n)	T5	output	LVDS		6	
GND*	T6				6	
GND*	T7				6	
Diff_out_pad[29]	T8	output	LVDS		6	
Diff_out_pad[29](n)	T9	output	LVDS		6	
GND	T10	gnd				
VCCINT	T11	power		1.2V		
GND	T12	gnd				
VCCINT	T13	power		1.2V		
GND	T14	gnd				
VCCINT	T15	power		1.2V		
GND	T16	gnd				
VCCINT	T17	power		1.2V		
GND	T18	gnd				
Diff_out_pad[23](n)	T19	output	LVDS		1	

Diff_out_pad[23]	T20	output	LVDS			1
Diff_out_pad[4](n)	T21	output	LVDS			1
Diff_out_pad[4]	T22	output	LVDS			1
VREFB1	T23	power				1
Diff_in_pad[23](n)	T24	input	LVDS			1
Diff_in_pad[23]	T25	input	LVDS			1
VCCIO1	T26	power			2.5V	1
Diff_in_pad[24]	U1	input	LVDS			6
Diff_in_pad[24](n)	U2	input	LVDS			6
toGige_Data_pad[0]	U3	output	LVDS			6
toGige_Data_pad[0](n)	U4	output	LVDS			6
toGigE_Clk_pad	U5	output	LVDS			6
toGigE_Clk_pad(n)	U6	output	LVDS			6
Diff_out_pad[46]	U7	output	LVDS			6
Diff_out_pad[46](n)	U8	output	LVDS			6
GND	U9	gnd				
VCCINT	U10	power			1.2V	
GND	U11	gnd				
VCCIO7	U12	power			1.8V	7
GND	U13	gnd				
VCCIO8	U14	power			1.8V	8
GND	U15	gnd				
VCCINT	U16	power			1.2V	
GND	U17	gnd				
VCCINT	U18	power			1.2V	
Diff_out_pad[33](n)	U19	output	LVDS			1
Diff_out_pad[33]	U20	output	LVDS			1
Diff_out_pad[20](n)	U21	output	LVDS			1
Diff_out_pad[20]	U22	output	LVDS			1
Diff_out_pad[6](n)	U23	output	LVDS			1
Diff_out_pad[6]	U24	output	LVDS			1
Diff_in_pad[21](n)	U25	input	LVDS			1
Diff_in_pad[21]	U26	input	LVDS			1
Diff_in_pad[15]	V1	input	LVDS			6
Diff_in_pad[15](n)	V2	input	LVDS			6
toGige_Data_pad[1]	V3	output	LVDS			6
toGige_Data_pad[1](n)	V4	output	LVDS			6
Clk_out_pad[7]	V5	output	LVDS			6
Clk_out_pad[7](n)	V6	output	LVDS			6
Diff_out_pad[43]	V7	output	LVDS			6
Diff_out_pad[43](n)	V8	output	LVDS			6
VCCINT	V9	power			1.2V	
DDRIN_pad[14]	V10	input	1.8 V			7
VCCPD7	V11	power			3.3V	7
GND*	V12					7
VCC_PLL6_OUT	V13	power			3.3V	10
DDRIN_pad[6]	V14	input	1.8 V			8
VCCPD8	V15	power			3.3V	8
DDROUT_pad[21]	V16	output	1.8 V			8
DDROUT_pad[44]	V17	output	1.8 V			8
DDROUT_pad[26]	V18	output	1.8 V			8
Diff_out_pad[32](n)	V19	output	LVDS			1
Diff_out_pad[32]	V20	output	LVDS			1
Diff_out_pad[21](n)	V21	output	LVDS			1
Diff_out_pad[21]	V22	output	LVDS			1
Diff_out_pad[11](n)	V23	output	LVDS			1
Diff_out_pad[11]	V24	output	LVDS			1
Diff_in_pad[19](n)	V25	input	LVDS			1
Diff_in_pad[19]	V26	input	LVDS			1
Diff_in_pad[14]	W1	input	LVDS			6
Diff_in_pad[14](n)	W2	input	LVDS			6
GND*	W3					6
GND*	W4					6

Diff_out_pad[39]	W5	output	LVDS		6	
Diff_out_pad[39](n)	W6	output	LVDS		6	
Diff_out_pad[35]	W7	output	LVDS		6	
Diff_out_pad[35](n)	W8	output	LVDS		6	
DDROUT_pad[58]	W9	output	1.8 V		7	
DDROUT_pad[14]	W10	output	1.8 V		7	
DDRIN_pad[49]	W11	input	1.8 V		7	
DDRIN_pad[13]	W12	input	1.8 V		7	
GNDA_PLL6	W13	gnd				
GNDA_PLL6	W14	gnd				
RESET_pad_	W15	input	1.8 V		8	
MCB_DATA_pad[0]	W16	bidir	1.8 V		8	
SPARE_IO_pad[17]	W17	input	1.8 V		8	
DDRIN_pad[5]	W18	input	1.8 V		8	
Diff_out_pad[17](n)	W19	output	LVDS		1	
Diff_out_pad[17]	W20	output	LVDS		1	
Diff_out_pad[8](n)	W21	output	LVDS		1	
Diff_out_pad[8]	W22	output	LVDS		1	
Diff_in_pad[16](n)	W23	input	LVDS		1	
Diff_in_pad[16]	W24	input	LVDS		1	
Diff_in_pad[17](n)	W25	input	LVDS		1	
Diff_in_pad[17]	W26	input	LVDS		1	
Diff_in_pad[13]	Y1	input	LVDS		6	
Diff_in_pad[13](n)	Y2	input	LVDS		6	
GND*	Y3				6	
GND*	Y4				6	
VREFB6	Y5	power			6	
Diff_out_pad[37]	Y6	output	LVDS		6	
Diff_out_pad[37](n)	Y7	output	LVDS		6	
PORSEL	Y8				7	
DDRIN_pad[70]	Y9	input	1.8 V		7	
DDROUT_pad[52]	Y10	output	1.8 V		7	
DDRIN_pad[48]	Y11	input	1.8 V		7	
DDRIN_pad[47]	Y12	input	1.8 V		7	
VCCA_PLL6	Y13	power		1.2V		
VCCD_PLL6	Y14	power		1.2V		
DDRIN_pad[36]	Y15	input	1.8 V		8	
MCB_DATA_pad[1]	Y16	bidir	1.8 V		8	
DDRIN_pad[4]	Y17	input	1.8 V		8	
DDRIN_pad[35]	Y18	input	1.8 V		8	
VCCSEL	Y19				8	
Diff_out_pad[9](n)	Y20	output	LVDS		1	
Diff_out_pad[9]	Y21	output	LVDS		1	
VREFB1	Y22	power			1	
Diff_in_pad[10](n)	Y23	input	LVDS		1	
Diff_in_pad[10]	Y24	input	LVDS		1	
Diff_in_pad[11](n)	Y25	input	LVDS		1	
Diff_in_pad[11]	Y26	input	LVDS		1	

```
-- NC           : No Connect. This pin has no internal connection to the device.
 -- DNU          : Do Not Use. This pin MUST NOT be connected.
 -- VCCINT       : Dedicated power pin, which MUST be connected to VCC  (1.2V).
 -- VCCIO        : Dedicated power pin, which MUST be connected to VCC
 --                of its bank.
 --                                Bank 1:       2.5V
 --                                Bank 2:       2.5V
 --                                Bank 3:       1.8V
 --                                Bank 4:       1.8V
 --                                Bank 5:       2.5V
 --                                Bank 6:       2.5V
 --                                Bank 7:       1.8V
 --                                Bank 8:       1.8V
 --                                Bank 9:       3.3V
 --                                Bank 10:      3.3V
 -- GND          : Dedicated ground pin. Dedicated GND pins MUST be connected to GND.
 --
```

```
-- GND+          : Unused input pin. It can also be used to report unused dual-purpose pins.
--                                   This pin should be connected to GND.
-- GND*          : Unused  I/O  pin.
-- RESERVED      : Unused I/O pin, which MUST be left unconnected.
-- RESERVED_INPUT    : Pin is tri-stated and should be connected to the board.
-- RESERVED_INPUT_WITH_WEAK_PULLUP    : Pin is tri-stated with internal weak pull-up resistor.
-- RESERVED_INPUT_WITH_BUS_HOLD       : Pin is tri-stated with bus-hold circuitry.
-- RESERVED_OUTPUT_DRIVEN_HIGH        : Pin is output driven high.
--------------------------------------------------------------------------------
```

9.2.1 Lower RXP Pin Notes

The Lower RXP pin file contains a number of generically-labeled I/Os. This section contains tables that define I/O connectivity, within the context of the Baseline Board.

X-ERNI Row #	Wafer #	CTRL (pin name)	Pin #	BB0 (pin name)	Pin #	BB1 (pin name)	Pin #
18	16	Diff_in_pad[0]	AD1 AD2	Diff_in_pad[4]	AC25 AC24	Diff_in_pad[5]	AD26 AD25
19	7	Diff_in_pad[1]	AC2 AC3	Diff_in_pad[6]	AB24 AB23	Diff_in_pad[7]	AB26 AB25
20	18	Diff_in_pad[2]	AB3 AB4	Diff_in_pad[8]	AA24 AA23	Diff_in_pad[9]	AA26 AA25
21	19	Diff_in_pad[3]	AB1 AB2	Diff_in_pad[10]	Y24 Y23	Diff_in_pad[11]	Y26 Y25
22	20	Diff_in_pad[12]	AA1 AA2	Diff_in_pad[16]	W24 W23	Diff_in_pad[17]	W26 W25
23	21	Diff_in_pad[13]	Y1 Y2	Diff_in_pad[18]	L25 L24	Diff_in_pad[19]	V26 V25
24	22	Diff_in_pad[14]	W1 W2	Diff_in_pad[20]	K24 K23	Diff_in_pad[21]	U26 U25
25	23	Diff_in_pad[15]	V1 V2	Diff_in_pad[22]	J24 J23	Diff_in_pad[23]	T25 T24
26	24	Diff_in_pad[24]	U1 U2	Diff_in_pad[28]	H24 H23	Diff_in_pad[29]	M26 M25
27	25	Diff_in_pad[25]	M1 M2	Diff_in_pad[30]	G24 G23	Diff_in_pad[31]	K26 K25
28	26	Diff_in_pad[26]	K1 K2	Diff_in_pad[32]	F24 F23	Diff_in_pad[33]	J26 J25
29	27	Diff_in_pad[27]	J2 J1	Diff_in_pad[34]	E24 E23	Diff_in_pad[35]	H26 H25
30	28	Diff_in_pad[36]	L2 L3	Diff_in_pad[40]	D25 D24	Diff_in_pad[41]	G26 G25
31	29	Diff_in_pad[37]	G1 G2	Diff_in_pad[42]	H1 H2	Diff_in_pad[43]	F26 F25
32	30	Diff_in_pad[38]	F1 F2	Diff_in_pad[44]	D2 D3	Diff_in_pad[45]	E26 E25
33	31	Diff_in_pad[39]	E1 E2	Diff_in_pad[46]	C1 C2	Diff_in_pad[47]	C26 C25

Table 9-6 Lower RXP FPGA wafer input to "pin name" and device pin number assignments. This table is nearly identical to Table 9-1, except for input wafer and ERNI connector row assignments. All signals use on-chip 100 ohm differential termination.

X/Y Recirc # [channel]	CTRL (pin name)	Pin #	BB0 (pin name)	Pin #	BB1 (pin name)	Pin #
4 [0]	Diff_out_pad[0]	N22 N21	Diff_out_pad[4]	T22 T21	Diff_out_pad[5]	AA22 AA21
4 [1]	Diff_out_pad[1]	J22 J21	Diff_out_pad[6]	U24 U23	Diff_out_pad[7]	L23 L22
4 [2]	Diff_out_pad[2]	H22 H21	Diff_out_pad[8]	W22 W21	Diff_out_pad[9]	Y21 Y20
4 [3]	Diff_out_pad[3]	K22 K21	Diff_out_pad[10]	M24 M23	Diff_out_pad[11]	V24 V23
5 [0]	Diff_out_pad[12]	L21 L20	Diff_out_pad[16]	M22 M21	Diff_out_pad[17]	W20 W19
5 [1]	Diff_out_pad[13]	G21 G20	Diff_out_pad[18]	N20 N19	Diff_out_pad[19]	J20 J19
5 [2]	Diff_out_pad[14]	F22 F21	Diff_out_pad[20]	U22 U21	Diff_out_pad[21]	V22 V21
5 [3]	Diff_out_pad[15]	H20 H19	Diff_out_pad[22]	K20 K19	Diff_out_pad[23]	T20 T19
6 [0]	Diff_out_pad[24]	N6 N7	Diff_out_pad[28]	P4 P5	Diff_out_pad[29]	T8 T9
6 [1]	Diff_out_pad[25]	J7 J8	Diff_out_pad[30]	R24 R23	Diff_out_pad[31]	L19 L18
6 [2]	Diff_out_pad[26]	H7 H8	Diff_out_pad[32]	V20 V19	Diff_out_pad[33]	U20 U19
6 [3]	Diff_out_pad[27]	K8 K9	Diff_out_pad[34]	M20 M19	Diff_out_pad[35]	W7 W8
7 [0]	Diff_out_pad[36]	N4 N5	Diff_out_pad[40]	M7 M8	Diff_out_pad[41]	G6 G7
7 [1]	Diff_out_pad[37]	Y6 Y7	Diff_out_pad[42]	L8 L9	Diff_out_pad[43]	V7 V8
7 [2]	Diff_out_pad[38]	AA5 AA6	Diff_out_pad[44]	J5 J6	Diff_out_pad[45]	H5 H6
7 [3]	Diff_out_pad[39]	W5 W6	Diff_out_pad[46]	U7 U8	Diff_out_pad[47]	K6 K7

Table 9-7 Lower RXP FPGA output pin number assignments, showing destination Recirculation FPGA numbers (rows-Y, columns-X) signals are destined for.

Note that within Recirculation FPGA nomenclature, including the GUI, channel [0] is “BB0, BB1”, channel [1] is “BB2, BB3” etc.

Each RXP FPGA shares its input data from the ERNI connector to its companion RXP, via a number of single-ended 1.8 V (12 mA driver) 512 Mbps DDR data lines. Each wafer, consisting of the CTRL, BB0, and BB1 signals consumes and is multiplexed across 5 of these lines, and therefore each RXP FPGA has, at minimum 5 x 16 = 80 lines. In addition, due to signal integrity issues (cross-talk minimization), only lines originating from the same source clock domain are allowed to be on the same PCB layer (refer to section 9.3).

The following table shows which of the "DDROUT_pad[79:0]" signals exiting the Lower RXP FPGA are assigned to (i.e. source from) each input wafer. In addition, there are 24 spare I/O pads available for Upper to/from Lower RXP future use.

X-ERNI Row #	Wafer #	DDROUT_pad lines	Pins
18	16	4:0	B7, D6, A5, B4, AD3
19	17	9:5	AD9, H9, AB8, C8, AC7
20	18	14:10	W10, B11, AE10, F10, D9
21	19	19:15	AB15, F14, F12, J11, AB11
22	20	24:20	AC18, F17, K17, V16, H15
23	21	29:25	AF20, E19, AE19, V18, D18
24	22	34:30	C23, B22, AF22, AC21, AB20
25	23	39:35	AC20, AD22, AE21, AE23, A24
26	24	44:40	V17, AD15, AB16, AA19, AB18
27	25	49:45	AC9, AD4, AE3, AD8, AF5
28	26	54:50	C6, AE7, Y10, AB12, AD11
29	27	59:55	B10, W9, C9, D8, A7
30	28	64:60	B16, G15, J14, E12, H11
31	29	69:65	C20, D19, E18, J18, G17
32	30	74:70	E7, B6, AD6, AE5, A3
33	31	79:75	E10, C10, AC10, F8, AF8
		SPARE_IO_pad[23:0]	Refer to pin file. SPARE_IO_pad[23:14] can be used ONLY for Lower->Upper data direction, due to layer signal restrictions. (section 9.3)

Table 9-8 Lower RXP DDROUT_pad[79:0] wafer and pin assignments.

DDROUT_pad[79:0] signals exiting the Lower RXP, enter the Upper RXP on its DDRIN_pad[79:0] pins.

SPARE_IO_pad[23:14] signals exiting the Lower RXP, enter the Upper RXP on its SPARE_IO_pad[23:14] inputs.

The following table shows the DDRIN_pad[79:0] pin and wafer assignments for the Lower RXP. These are connected to DDROUT_pad[79:0] pins leaving the Upper RXP.

X-ERNI Row #	Wafer #	DDRIN_pad lines	Pins
1	0	4:0	Y17, AC15, AC19, AD21, AE24
2	1	9:5	AD7, AE6, AF3, V14, W18
3	2	14:10	V10, W12, AA8, AA11, AD10
4	3	19:15	G11, J9, K10, K16, H18
5	4	24:20	F15, B8, C5, D7, F11
6	5	29:25	C18, B21, C22, F20, G19
7	6	34:30	AA17, AB19, AD20, AF21, C17
8	7	39:35	AE9, AF10, AA14, Y15, Y18
9	8	44:40	AC6, AC8, AD5, AE4, AF7
10	9	49:45	W11, Y11, Y12, AA10, AB9
11	10	54:50	E15, G9, G10, G16, J17
12	11	59:55	D17, D20, E9, E11, E14
13	12	64:60	AF9, C7, A8, G18, B19
14	13	69:65	AA12, AB7, AB10, AF6, AE8
15	14	74:70	B5, E8, H10, K11, Y9
16	15	79:75	F13, C11, D10, B9, A6

Table 9-9 Lower RXP DDRIN_pad wafer and pin assignments. These are inputs with data sourcing from the Upper RXP.

The following table contains a list of miscellaneous signals, their functions, and pin assignments for the Lower RXP FPGA.

Signal	Pin(s)	Description/Function
REF_CLOCK_pad[0]	R26, R25	Primary 128 MHz wafer clock input. This sources from wafer 16 (ERNI connector row 18).
REF_CLOCK_pad[3:1]	(R1,R2) (P2,P3) (P25,P24)	Auxiliary 128 MHz wafer clock inputs. These are not used in the FPGA, but may be with compile-time selection. These source from wafers 28, 24, and 20 respectively.
EXT_CLK128_in_pad	P23, P22	External 128 MHz input; sources from the board's SMA external clock input connector.
DDROUT_CLK_pad	J10	256 MHz clock output synchronous with DDROUT_pad signals.
DDRIN_CLK_pad	B13	256 MHz clock input synchronous with DDRIN_pad signals.
toGige_Data_pad[1:0]	*see pin file*	[1]=encoded VDIF output packets; [0]=idle codes
toGigE_Clk_pad	U5, U6	128 MHz clock out to GigE FPGA
Phasing_out_pad[3:0]	*see pin file*	[3]=4-bit Phased output in HM Gbps format; [2]=0; [1]=HM Gbps CTRL; [0]=128 MHz clock. Out to ERNI row 35.
Clk_out_pad[7:0]	*see pin file*	8, 128 MHz clocks to X/Y4:7 Recirc FPGAs
RESET_pad_	W15	Active-low chip reset, resets all regs on the chip. Does not de-configure FPGA.
MCB_CLK_pad	AA16	33 MHz CPU clock; inputs have a 15 nsec t_{su} requirement
MCB_ADDR_pad[3:0]	*see pin file*	4-bit CPU address bus
MCB_DATA_pad[7:0]	*see pin file*	8-bit bi-directional CPU data bus; on read, there is a 25 nsec MCB_CLK_pad t_{co} delay.
MCB_CS_pad_	AE16	Active-low CPU chip select
MCB_RD_WR_pad_	AC17	Active-low write (read=high)
testmode_tick_out_pad	A21	Unused in the Upper RXP
testmode_tick_in_pad	A20	10 msec input time tick for testmode operation (sync U/L RXPs)
testmode_1pps_out_pad	AF24	Unused in the Upper RXP
testmode_1pps_in_pad	AD23	1 sec input time tick for testmode operation (sync U/L RXPs)
testmode_dummy_pad		Unused, connect high or low or leave unconnected

Table 9-10 Lower RXP miscellaneous signals' descriptions.

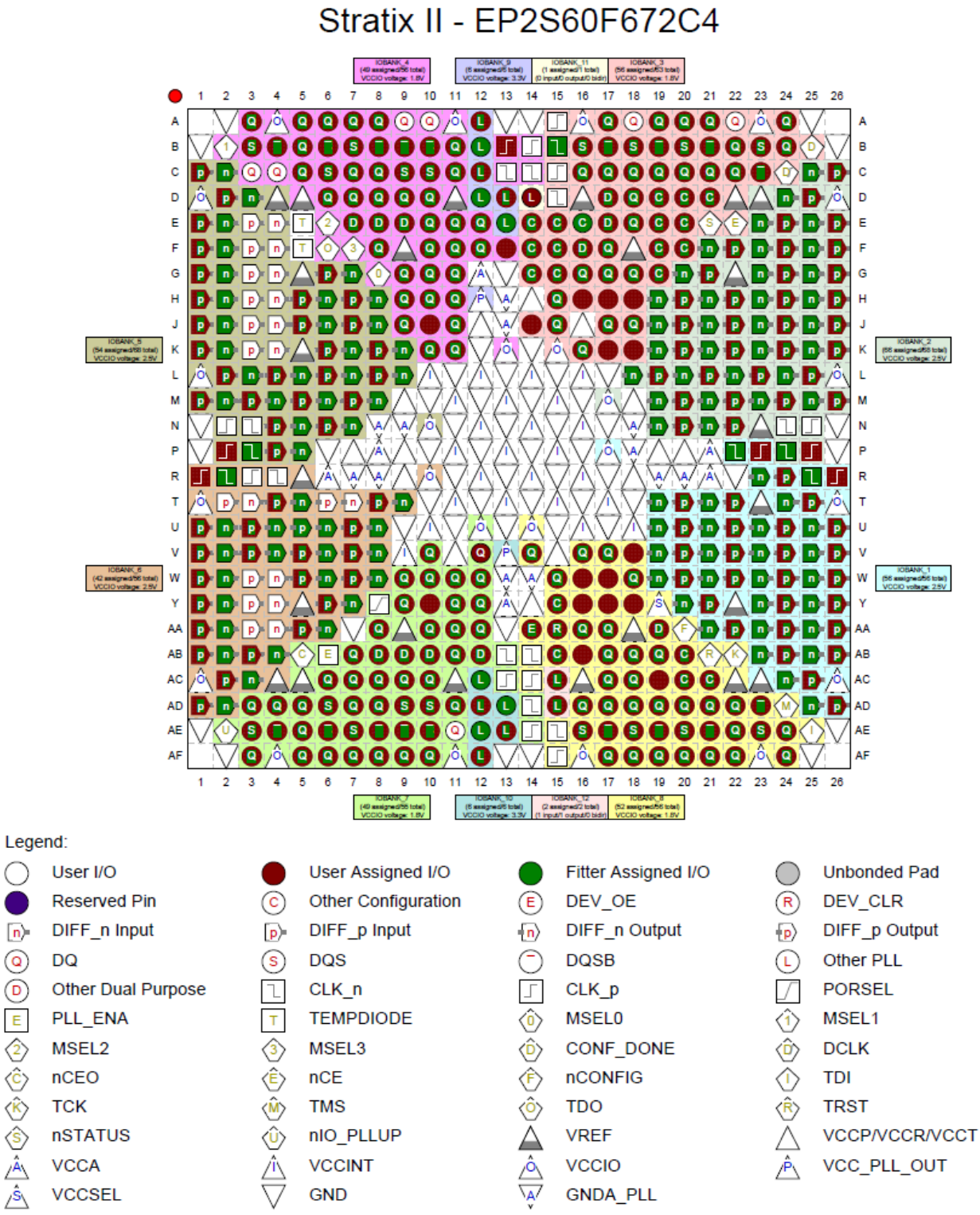


Figure 9-2 Lower RXP Altera Quartus-II pin planner output. Top view of the chip.

9.3 DDROUT_pad and DDRIN_pad Routing Layer Assignments/Restrictions

As previously mentioned, the DDROUT and DDRIN (and SPARE_IO_pad) signals must be routed on specific PCB layers such that all signals that are on the same layer or adjacent layers without a power or GND plane isolating them, **launch from the same chip at the same time**. This prevents signal-to-signal coupled switching noise from causing transients in the middle of the eye of a signal. This measure is necessary because of the high density and high speed of the single-ended, un-terminated 1.8 V signals.

The applicable PCB layer stack-up (layers 1-14 only are used) is as follows:

L1
L2
--GND-- (V2.0: +5V; V2.1: DGND)
L4
L5
--GND-- 1.2V
L7
L8
--GND-- DGND
L10
--GND-- 2.5V
L12
--GND-- AGND
L14

Layer 1 is unusable as it is the top layer used for BGA ball-pad breakout to vias. Therefore, layers 2, 4, 5, 7, 8, 10, 12, and 14 are usable routing layers for this purpose.

Pin assignments to each layer are as follows:

Layer 2 pins:
AF21, AD20, AB19, AA17, Y18,
Y15, AA14, AF10, AF7, AE4,
AD5, AC8, AC6, AB9, AA10, Y12,
Y11, W11, J17, G16, G10, G9, E15,
E14, E11, E9, D20, D17, B19,
G18, A8, C7

Layer 4 pins:
AE24, AD21, AC19, Y17, W18,
V14, AF3, AE6, AD7, AD10, AA11,
AA8, W12, V10, H18, K16, K10, J9, G11,
F11, D7, C5, B8, F15, G19, F20,
C22, B21, A20, C18, C17

Layer 5 pins:
AF9, AE8, AF6, AB10, AB7,
AA12, Y9, K11, H10, E8, B5,
A6, B9, D10, C11, F13, D14,
J15, H16, C16, E17, B17,
B18, F19, C19, A19, E20,
B20, D21, B23, B24

Layer 10 pins:
AE23, AE21, AD22, AC20, AB18,
AA19, AB16, AD15, V17, AF5,
AD8, AE3, AD4, AC9, AD11, AB12,
Y10, AE7, C6, A7, D8, C9,
W9, B10, H11, E12, J14, G15,
B16, G17, J18, E18, D19,
C20, A21

Layer 12 pins:
AD3, B4, A5, D6, B7, AC7,
C8, AB8, H9, AD9, D9, F10,
AE10, B11, W10, AB11, J11, F12,
F14, AB15, H15, V16, K17, F17, AC18,
D18, V18, AE19, E19, AF20, AB20,
AC21, AF22, B22, C23, A24

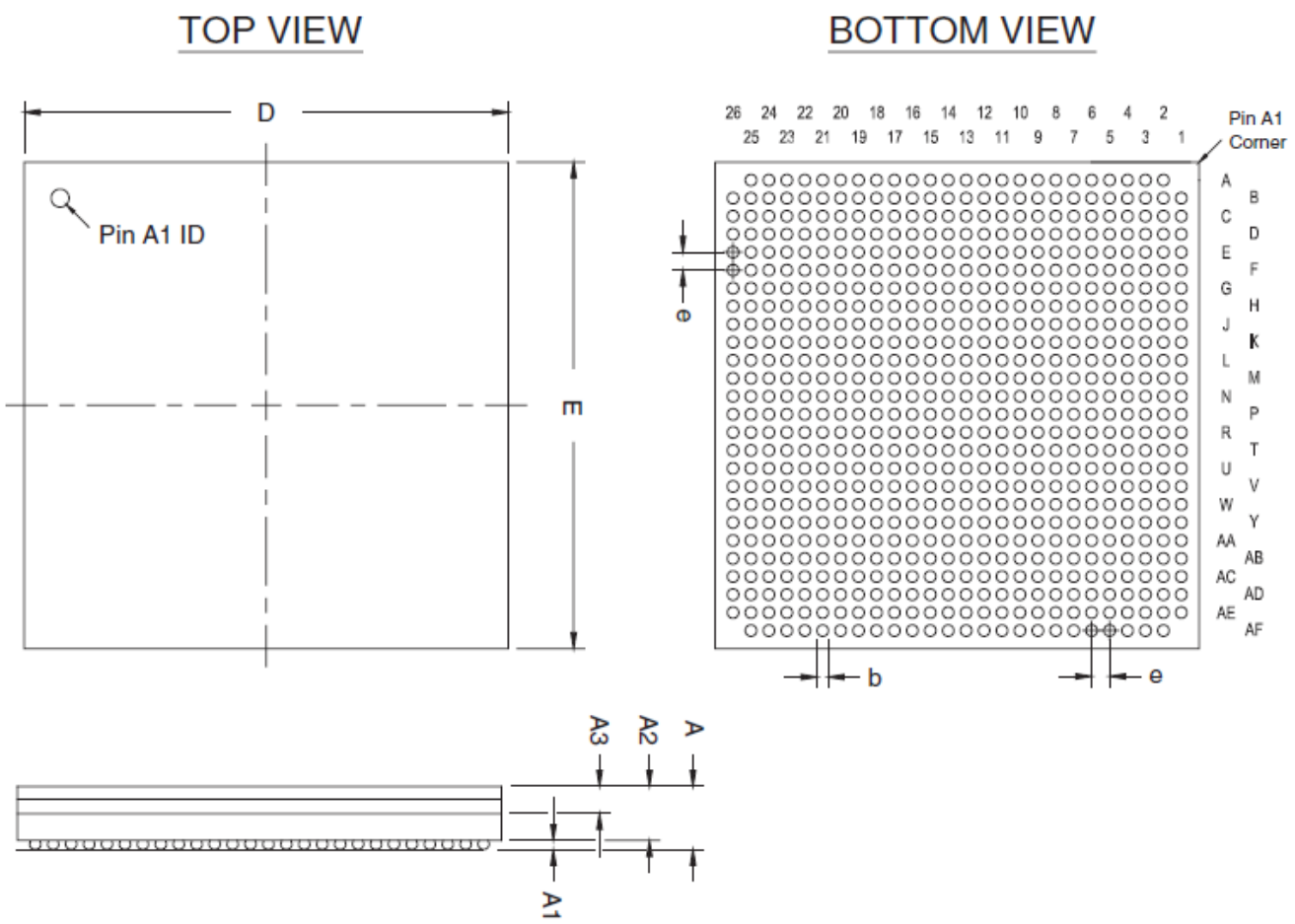
Layer 14 pins:
A3, AE5, AD6, B6, E7, AF8, F8,
C10, E10, V12, G14, AA15, F16, W17,
A17, H17, K18, AE20, C21, AE22,
AD23, AF24, AE9

Layer 7 pins:
This is the clock layer, but contains one data
line, namely AC10.

J10, and B13 are clkout and clkin pins/lines respectively.

IF ANY DESIGN CHANGES REQUIRE MODIFICATION OR USE OF DDROUT_pad, DDRIN_pad, OR SPARE_IO_pad, THEN THESE RESTRICTIONS MUST BE ADHERRED TO FOR ERROR-FREE OPERATION.

9.4 Altera EP2S60F672C4 Package Drawing



Symbol	Millimeters		
	Min.	Nom.	Max.
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
D	27.00 BSC		
E	27.00 BSC		
b	0.50	0.60	0.70
e	1.00 BSC		

Figure 9-3 RXP FPGA F672 package outline and dimensions. Drawings and dimensions taken from the Altera Stratix-II package information data sheet.

9.5 Altera FPGA Programming Notes

The Baseline Board on which the RXP FPGAs reside is set up for 1-bit Passive Serial programming via the PCMC FPGA. The RXP FPGAs are in series to allow each FPGA to have a different personality, and so there is one file that contains the program for both FPGAs. The following figure is a screen shot of the "Convert Programming Files" dialog in the Altera Quartus-II software, showing the setup required to produce the .rbf (raw binary file), needed by the software to program the FPGA.

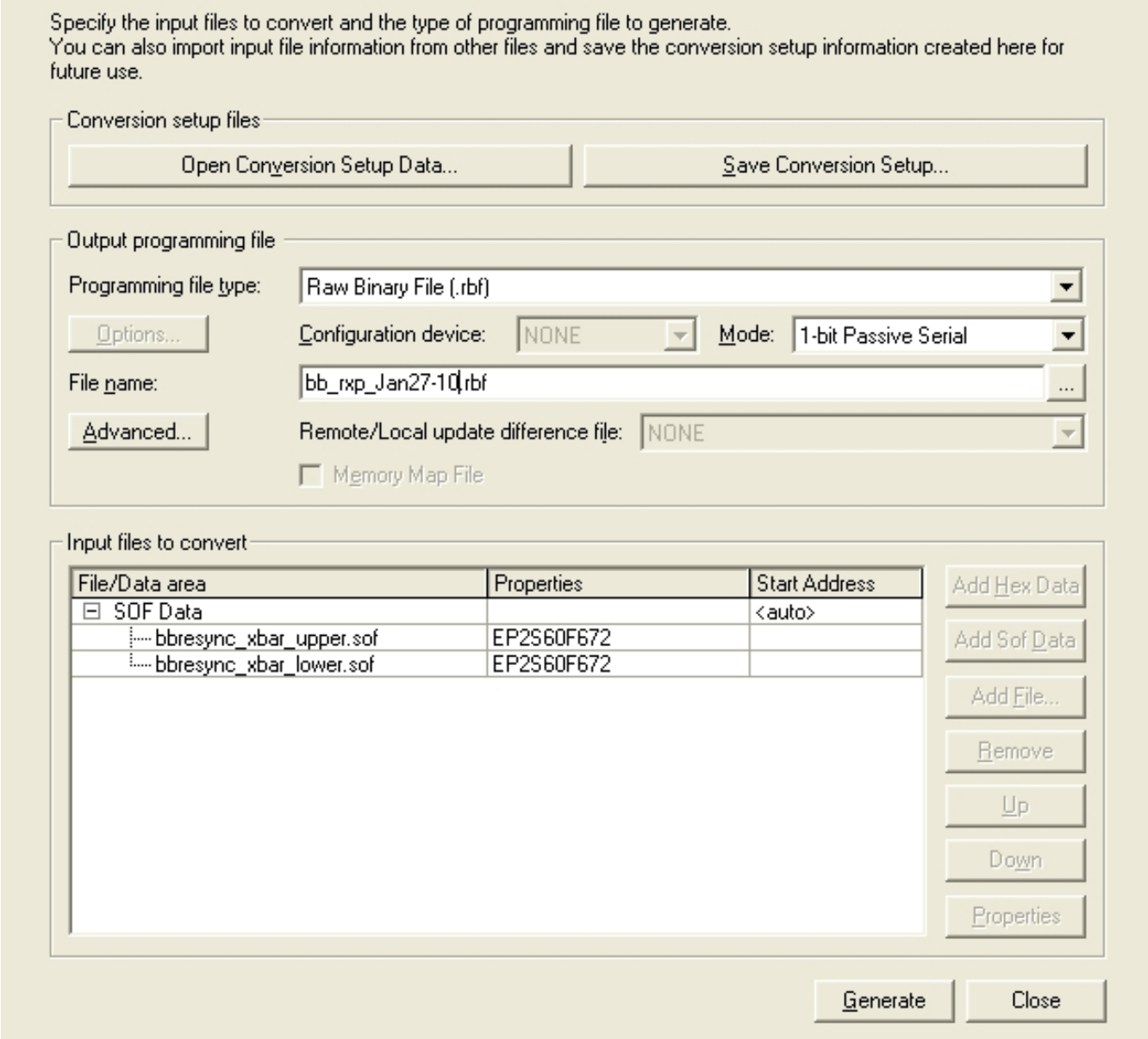


Figure 9-4 Altera Quartus-II "Convert Programming Files" screenshot showing settings necessary for generating the .rbf file for the RXP FPGAs.

10 Index

R

S

T