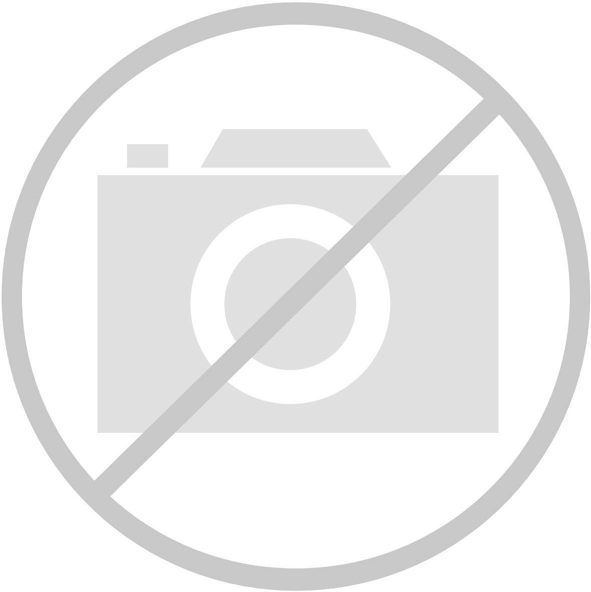
E·XFL Lattice Semiconductor Corporation - ORT8850L-1BM680C Datasheet



Welcome to E-XFL.COM

Understanding Embedded - FPGAs (Field Programmable Gate Array)

Embedded - FPGAs, or Field Programmable Gate Arrays, are advanced integrated circuits that offer unparalleled flexibility and performance for digital systems. Unlike traditional fixed-function logic devices, FPGAs can be programmed and reprogrammed to execute a wide array of logical operations, enabling customized functionality tailored to specific applications. This reprogrammability allows developers to iterate designs quickly and implement complex functions without the need for custom hardware.

Applications of Embedded - FPGAs

The versatility of Embedded - FPGAs makes them indispensable in numerous fields. In telecommunications,

Details	
Product Status	Obsolete
Number of LABs/CLBs	-
Number of Logic Elements/Cells	4992
Total RAM Bits	75776
Number of I/O	278
Number of Gates	397000
Voltage - Supply	1.425V ~ 3.6V
Mounting Type	Surface Mount
Operating Temperature	0°C ~ 70°C (TA)
Package / Case	680-BBGA
Supplier Device Package	680-FPBGA (35x35)
Purchase URL	https://www.e-xfl.com/product-detail/lattice-semiconductor/ort8850l-1bm680c

Table of Contents

Introduction 1
Table of Contents 2
Features 3
Embedded Core Features 3
FPGA Features 4
Programmable Logic System Features 5
Description 6
What is an FPSC? 6
FPSC Overview 6
ispLEVER Development System 7
FPSC Design Kit 7
FPGA Logic Overview 8
System-Level Features 9
Configuration 10
Additional Information 10
ORT8850 Overview 11
Embedded Core Overview 11
SONET Logic Blocks - Overview 12
System Considerations for Reference Clock Distribution 15
SONET Bypass Mode 16
STM Macrocells - Overview 17
HSI Macrocell - Overview 19
Supervisory and Test Support Features - Overview 19
Protection Switching - Overview 21
FPSC Configuration - Overview 22
Backplane Transceiver Core Detailed Description 25
SONET Logic Blocks, Detailed Description 25
Receive Path Logic 34
FPGA/Embedded Core Interface Signals 47
Clock and Data Timing at the FPGA/Embedded Core Interface - SONET Block 49
Powerdown Mode 56
Protection Switching 56
Memory Map 57
Registers Access and General Description 57
Electrical Characteristics 69
Absolute Maximum Ratings 69
Recommended Operating Conditions 69
Power Supply Decoupling LC Circuit 70
HSI Electrical and Timing Characteristics 71
Embedded Core LVDS I/O 73
Pin Information 77
Package Pinouts 82
Package Thermal Characteristics Summary 100
qJA 100
YJC 100
qJC 100
qJB 100
FPSC Maximum Junction Temperature 101
Package Thermal Characteristics 101
Heat Sink Information 101
Package Coplanarity 101
Package Parasitics 102
Package Outline Diagrams 102
Terms and Definitions 102
Package Outline Drawings 103
Ordering Information 104
Revision History 105

FPGA Logic Overview

The ORCA Series 4 architecture is a new generation of SRAM-based programmable devices from Lattice. It includes enhancements and innovations geared toward today's high-speed systems on a single chip. Designed with networking applications in mind, the Series 4 family incorporates system-level features that can further reduce logic requirements and increase system speed. ORCA Series 4 devices contain many new patented enhancements and are offered in a variety of packages and speed grades.

The hierarchical architecture of the logic, clocks, routing, RAM, and system-level blocks create a seamless merge of FPGA and ASIC designs. Modular hardware and software technologies enable System-on-a-Chip integration with true plug-and-play design implementation.

The architecture consists of four basic elements: Programmable Logic Cells (PLCs), Programmable I/O cells (PIOs), Embedded Block RAMs (EBRs) and system-level features. These elements are interconnected with a rich routing fabric of both global and local wires. An array of PLCs are surrounded by common interface blocks which provide an abundant interface to the adjacent PLCs or system blocks. Routing congestion around these critical blocks is eliminated by the use of the same routing fabric implemented within the programmable logic core. Each PLC contains a PFU, SLIC, local routing resources, and configuration RAM. Most of the FPGA logic is performed in the PFU, but decoders, *PAL*-like functions, and 3-state buffering can be performed in the SLIC. The PIOs provide device inputs and outputs and can be used to register signals and to perform input demultiplexing, output multiplexing, uplink and downlink functions, and other functions on two output signals. Large blocks of 512 x 18 quad-port RAM complement the existing distributed PFU memory. The RAM blocks can be used to implement RAM, ROM, FIFO, multiplier, and CAM. Some of the other system-level functions include the MicroProcessor Interface (MPI), Phase-Locked Loops (PLLs), and the Embedded System Bus (ESB).

PLC Logic

Each PFU within a PLC contains eight 4-input (16-bit) LUTs, eight latches/flip-flops, and one additional Flip-Flop that may be used independently or with arithmetic functions.

The PFU is organized in a twin-quad fashion; two sets of four LUTs and flip-flops that can be controlled independently. Each PFU has two independent programmable clocks, clock enables, local SET/RESET, and data selects. LUTs may also be combined for use in arithmetic functions using fast-carry chain logic in either 4-bit or 8-bit modes. The carry-out of either mode may be registered in the ninth flip-flop for pipelining. Each PFU may also be configured as a synchronous 32 x 4 single- or dual-port RAM or ROM. The flip-flops (or latches) may obtain input from LUT outputs or directly from invertible PFU inputs, or they can be tied high or tied low. The flip-flops also have programmable clock polarity, clock enables, and local SET/RESET.

The SLIC is connected from PLC routing resources and from the outputs of the PFU. It contains eight 3-state, bidirectional buffers, and logic to perform up to a 10-bit AND function for decoding, or an AND-OR with optional INVERT to perform *PAL*-like functions. The 3-state drivers in the SLIC and their direct connections from the PFU outputs make fast, true, 3-state buses possible within the FPGA, reducing required routing and allowing for real-world system performance.

Programmable I/O

The Series 4 PIO addresses the demand for the flexibility to select I/Os that meet system interface requirements. I/Os can be programmed in the same manner as in previous *ORCA* devices, with the additional new features which allow the user the flexibility to select new I/O types that support High-Speed Interfaces.

Each PIO contains four Programmable I/O pads and is interfaced through a common interface block to the FPGA array. The PIO is split into two pairs of I/O pads with each pair having independent clock enables, local SET/RESET, and global SET/RESET. On the input side, each PIO contains a programmable latch/Flip-Flop which enables very fast latching of data from any pad. The combination provides very low setup requirements and zero hold times for signals coming on-chip. It may also be used to demultiplex an input signal, such as a multiplexed address/data signal, and register the signals without explicitly building a demultiplexer with a PFU. On the output side of each PIO, an output from the PLC array can be routed to each output flip-flop, and logic can be associated with each I/O pad. The output logic associated with each pad allows for multiplexing of output signals and other functions of two output signals.

The output flip-flop, in combination with output signal multiplexing, is particularly useful for registering address signals to be multiplexed with data, allowing a full clock cycle for the data to propagate to the output. The output buffer signal can be inverted, and the 3-state control can be made active-high, active-low, or always enabled. In addition, this 3-state signal can be registered or nonregistered.

The Series 4 I/O logic has been enhanced to include modes for speed uplink and downlink capabilities. These modes are supported through shift register logic, which divides down incoming data rates or multiplies up outgoing data rates. This new logic block also supports high-speed DDR mode requirements where data is clocked into and out of the I/O buffers on both edges of the clock.

The new Programmable I/O cell allows designers to select I/Os which meet many new communication standards permitting the device to hook up directly without any external interface translation. They support traditional FPGA standards as well as high-speed, single-ended, and differential-pair signaling. Based on a programmable, bank-oriented I/O ring architecture, designs can be implemented using 3.3V/ 2.5V/1.8V/1.5V referenced output levels.

Routing

The abundant routing resources of the Series 4 architecture are organized to route signals individually or as buses with related control signals. Both local and global signals utilize high-speed buffered and nonbuffered routes. One PLC segmented (x1), six PLC segmented (x6), and bused half-chip (xHL) routes are patterned together to provide high connectivity with fast software routing times and high-speed system performance.

Eight fully distributed primary clocks are routed on a low-skew, high-speed distribution network and may be sourced from dedicated I/O pads, PLLs, or the PLC logic. Secondary and edge-clock routing is available for fast regional clock or control signal routing for both internal regions and on device edges. Secondary clock routing can be sourced from any I/O pin, PLLs, or the PLC logic.

The improved routing resources offer great flexibility in moving signals to and from the logic core. This flexibility translates into an improved capability to route designs at the required speeds when the I/O signals have been locked to specific pins.

System-Level Features

The Series 4 also provides system-level functionality by means of its MicroProcessor Interface, embedded system bus, quad-port embedded block RAMs, universal Programmable Phase-Locked Loops, and the addition of highly tuned networking specific phase-locked loops. These functional blocks support easy glueless system interfacing and the capability to adjust to varying conditions in today's high-speed networking systems.

MicroProcessor Interface

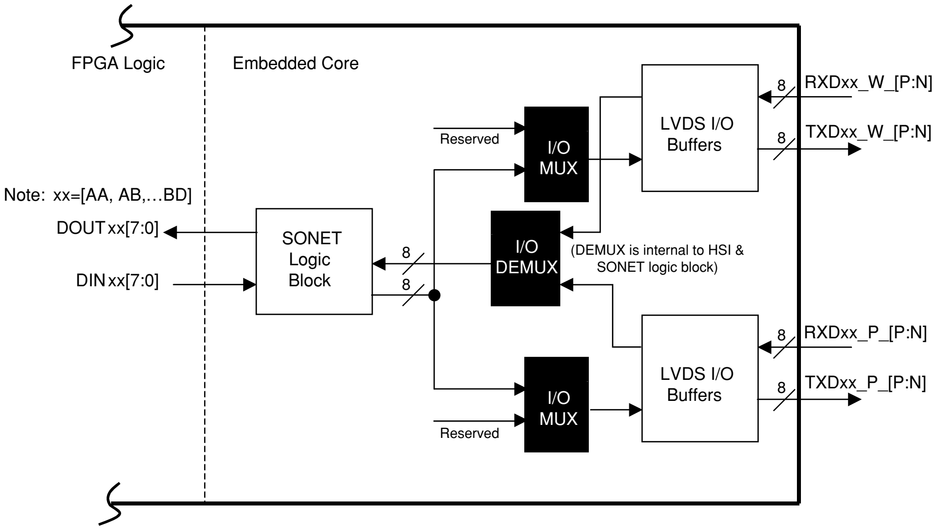
The MPI provides a glueless interface between the FPGA and *PowerPC* microprocessors. Programmable in 8, 16, and 32-bit interfaces with optional parity to the *Motorola*® *PowerPC MPC*860 and MPC8260 bus, it can be used for configuration and readback, as well as for FPGA control and monitoring of FPGA status. All MPI transactions utilize the Series 4 embedded system bus at 66 MHz performance.

The MPI provides a system-level MicroProcessor Interface to the FPGA user-defined logic, following configuration, through the system bus, including access to the embedded block RAM and general user-logic. The MPI supports burst data read and write transfers, allowing short, uneven transmission of data through the interface by including data FIFOs. Transfer accesses can be single beat (1 x 4-bytes or less), 4-beat (4 x 4-bytes), 8-beat (8 x 2-bytes), or 16-beat (16 x 1-bytes).

System Bus

An on-chip, multimaster, 32-bit system bus with 1-bit parity facilitates communication among the MPI, configuration logic, FPGA control, and status registers, embedded block RAMs, as well as user logic. Utilizing the *AMBA* specification Rev 2.0 AHB protocol, the embedded system bus offers arbiter, decoder, master, and slave elements. Mas-

Figure 2. ORT8850 Embedded Core, Top Level Functionality and Data Flow



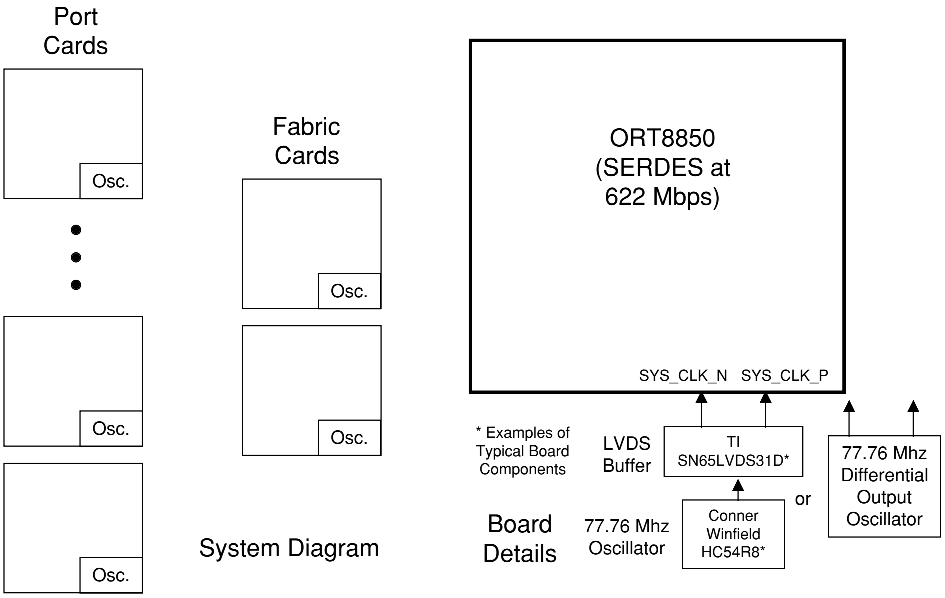
SONET Logic Blocks - Overview

The 850 Mbits/s SONET logic blocks allows the ORT8850 to communicate across a backplane or on a given board at an aggregate speed of 6.8 Gbits/s, allowing high-speed asynchronous serial data transfer between system devices. The external serial interfaces are implemented as eight channels of bidirectional 850 Mbits/s LVDS links and use a pseudo-SONET framing protocol, which can be bypassed.

The SONET logic blocks are organized into two quads. Each quad supports four full duplex serial links (quad A contains channels AA, AB, AC, and AD while quad B contains channels BA, BB, BC, and BD). A top level block diagram of one channel of the SONET logic is shown in Figure 3.

this feature the alignment FIFO cannot be used with this clock architecture. The recovered clock is used for all receive timing in the embedded core and supplied to the FPGA logic which must provide the clock domain transfer functionality.

Figure 6. Independent Clock Architecture



SONET Bypass Mode

It is possible to utilize only the serializer and deserializer (SERDES) blocks in the ORT8850 and to bypass all of the SONET framing and scrambling/descrambling. In this mode the parallel data from the FPGA is serialized and sent out the LVDS pins. The serial data in the receive direction will be run through the SERDES and then received as parallel data with a recovered clock into the FPGA.

In the SONET Bypass mode there exists half and quarter rate selection options. Half rate allows the SERDES to operate at 4x the reference clock. When using half rate mode only the bits 7:4 of the parallel FPGA bus are utilized. Quarter rate allows the SERDES to operate at 2x the reference clock. When using quarter rate mode only bits 7:6 of the parallel FPGA bus are utilized. Half rate and quarter rate are selectable per channel and can be mixed per channel so that some channels can run in full rate mode while others operate in half rate mode and still others operate in quarter rate mode.

As shown in Table 3, 63.00 MHz is the slowest reference clock and 106.25 MHz is the fastest reference clock frequency supported. For all three modes, all bandwidths within the reference clock limits are supported. Note that there are gaps between the bandwidths supported in the three modes.

Table 3. SONET Bypass Mode Bandwidth Options

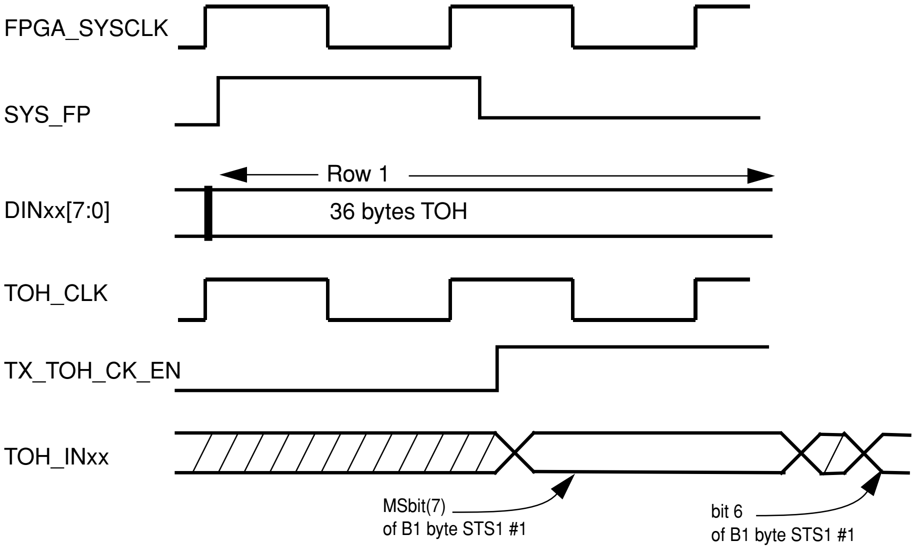
Reference Clock	Full Mode	Half Mode Bits [7:4] Used	Quarter Mode Bits [7:6] Used
63	504.00 Mbits/s	252.00 Mbits/s	126.00 Mbits/s
77.76	622.08Mbits/s	311.04Mbits/s	155.52Mbits/s
106.25	850.00 Mbits/s	425.00 Mbits/s	212.50 Mbits/s

In the SONET Bypass mode a 1's density function similar to SONET scrambling must be implemented in the FPGA logic to assure reliable clock recovery at the receiver.

Serial TOH Insertion Mode

In the transmit direction the SPE bytes are always transferred unaltered from the input parallel bus to the serial LVDS output. On the other hand, TOH bytes are received from the serial input port and are inserted in the STS-12 frame before being sent to the LVDS output in the Serial TOH Insertion mode. The FPGA logic must provide framing information to the Core using the TX_TOH_CK_EN Input signal. TOH data is input on a row by row basis, with a one clock cycle frame pulse delineating the start of a row, as shown in Figure 14. As shown in the figure, while the SPE bytes are being transmitted for one row, the FPGA logic must simultaneously supply the Core with the TOH data for the next row. Detailed timing for the TOH serial input is shown later in Figure 31.

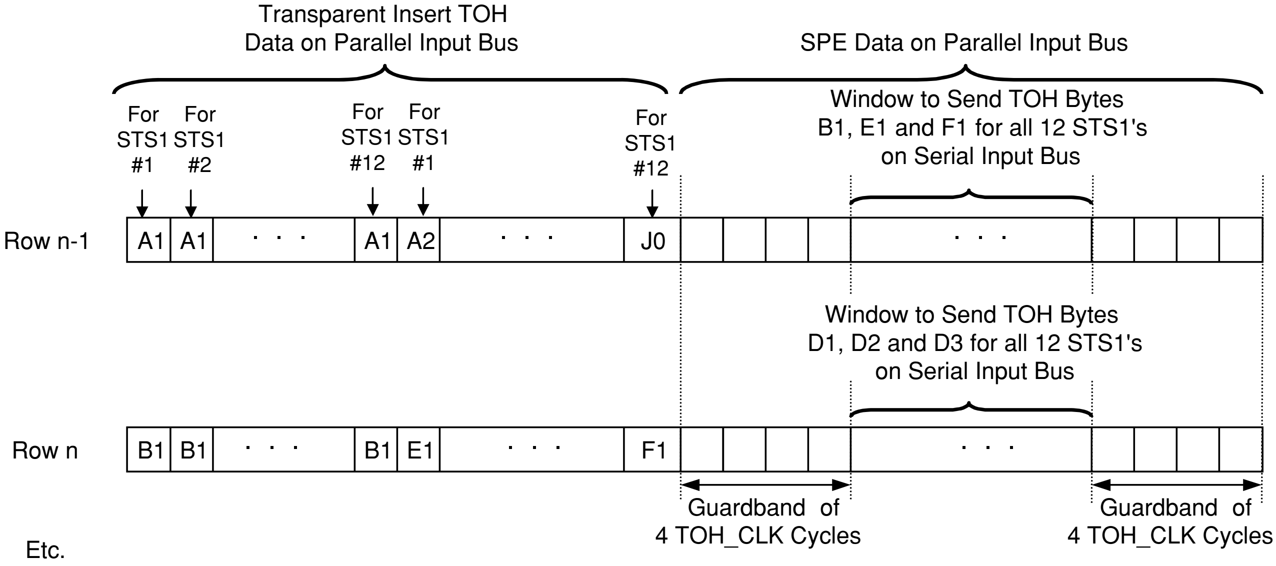
Figure 14. TOH Serial Port Input Framing Signals (FPGA to Core)



Incoming serial TOH data is synchronized initially to the free running clock, TOH_CLK. TOH_CLK can operate from a minimum frequency of 25 Mhz. to a maximum frequency of 106 MHz. TOH bytes are transferred in the order shown in Figure 15. Bytes are transferred over the serial links with the MSB first. Data should be transferred over the serial link on a row-by-row basis. With three TOH bytes/per row for each STS-1 stream and a total of 12 STS-1 streams per STS-12 frame, a total of 288 TOH bits must be transferred for each row. The 288 TOH bits per row can be sent back-to-back. In this case, TX_TO_CLK_EN will be high continuously for 288 TOH_CLK cycles.

It is the responsibility of the user to synchronize transfer of the TOH bytes to a pre-determined window of time relative to the STS-12 frame position on the parallel input bus, i.e., the 36 TOH bytes to be inserted in row number n must be transferred to the Core during the time the SPE bytes of row n-1 are being transferred to the Core over the parallel input bus. Within each SPE row, a guard band of four TOH_CLK cycles must be provided on each side of the TOH transfer window. No data may be transferred in these guard bands.

Figure 15. TOH Serial Port Input Framing Signals (FPGA to Core)


Transparent Insert TOH
Data on Parallel Input Bus
SPE Data on Parallel Input Bus
For STS1 #1
For STS1 #2
For STS1 #12
For STS1 #1
For STS1 #12
Window to Send TOH Bytes B1, E1 and F1 for all 12 STS1's on Serial Input Bus
Row n-1
A1
A1
· · ·
A1
A2
· · ·
J0
· · ·
Window to Send TOH Bytes D1, D2 and D3 for all 12 STS1's on Serial Input Bus
Row n
B1
B1
· · ·
B1
E1
· · ·
F1
· · ·
Guardband of 4 TOH_CLK Cycles
Guardband of 4 TOH_CLK Cycles
Etc.

Although all TOH bytes from the 12 STS-1s are transferred into the device from each serial port, not all of them get inserted in the frame. There are three hard coded exceptions to the TOH byte insertion:

- Framing bytes (A1/A2 of all STS-1s) are not inserted from the serial input bus. Instead, they can always be regenerated.
- Parity byte (B1 of STS#1) is not inserted from the serial input bus. Instead, it is always recalculated (the 11 bytes following B1 are replaced with all zeros).
- Pointer bytes (H1/H2/H3 of all STS-1s) are not inserted from the serial input bus. Instead, they always flow transparently from parallel input to LVDS output.

Except for the above hardcode exceptions, the source of some TOH bytes can be controlled by bits in the control registers. The 12 STS-1 bytes forming a single STS-12 TOH header block are controlled as a whole. When configured to be in the transparent mode, the specific bytes must flow transparently from the parallel input. The 15 overhead bytes that can be controlled on a per STS-1 basis are the following:

- K1 and K2 bytes of the 12 STS-1s (24 bytes)
- S1 and M0 bytes of the 12 STS-1s (24 bytes)
- E1, F1, E2 bytes of the STS-1s (36 bytes)
- D1 through D12 bytes of the STS-1s (144 bytes)

The C1(J0) and B2 bytes (unshaded in the following table) are also passed through transparently from the parallel bus to the serial link.

Table 10 shows the order in which data is transferred to the serial LVDS output, starting with the most significant bit of the first A1 byte. The first bit of the first byte is replaced by an even parity check bit over all TOH bytes from the previous TOH frame. The source for the TOH bytes in the Serial TOH insert mode is summarized in the table.

passes, the next state will either still be Frame Confirm or will be In Frame. For the framer to declare an In Frame state the framer must detect 4 consecutive correct A1/A2 framing patterns.

This state is similar to the Frame Confirm state except that if the comparison at the A1/A2 time is incorrect, the next state will be the Errored Frame state. If the comparison is correct, the next state will be In Frame. Data is only valid in the Frame state

Errored Frame State

Once the Errored Frame state has been reached, if the next comparison is incorrect, the next state will be OOF i.e., after two transitions are missed, the state machine goes into the OOF state which will also generate an alarm. Otherwise, if the comparison correct, the next state will be In Frame. Also, when the framer detects an errored frame it increments an A1/A2 frame error counter register accessible from the system bus. The counter can be monitored by a processor to compile performance status on the quality of the backplane.

B1 Parity Error Check

The B1 parity error check block receives byte-wide scrambled byte-wide parallel data and a frame sync from the framer. The B1 error check calculation block computes a BIP-8 (bit interleaved parity 8 bits) code, using even parity over all bits of the current STS-12 frame before descrambling.

The same calculation had previous been done for the previous STS-12 frame. The value obtained then is checked against the B1 byte of the current frame after descrambling. A per-stream B1 error counter is incremented for each bit that is in error. The error counter register is accessible from the system bus.

Descrambler

The received streams from the framer are descrambled using a frame synchronous descrambler with the same polynomial ($1 + x^6 + x^7$) that was used in the transmit path. If the incoming data is not scrambled, the descrambling function can be disabled by setting a control register bit (0x3000C). The A1/A2 framing bytes, the section trace byte (C1/J0) and the growth bytes (Z0) are not descrambled.

AIS-L Insertion

The Alarm Indication Signal (AIS) is a continuous stream of unframed 1s sent to alert downstream equipment that the near-end terminal has failed, lost its signal source, or has been temporarily taken out of service. AIS-L is inserted into the received frame by writing all ones for all bytes of the descrambled stream under two conditions:

1. If a force AIS_L state is enabled by a bit in the AIS-L force register, AIS-L is inserted into the received frame continuously. This will cause all bytes within a STS-12 frame to be FF
2. If an AIS-L Insertion on Out-Of-Frame enabled via a register, AIS-L is inserted into the received frame when the framer indicates that an out-of-frame condition exists.

Since this occurs after the overhead processing block, all Transport Overhead can continue to byte read and B1 can still be used to monitor link integrity.

Alignment FIFO and Multi-Channel Alignment

The alignment FIFO in the ORT8850 performs two functions, clock domain transfer and multi-channel alignment. The depth of the alignment FIFO is 10 bit words which allows it to absorb channel timing differences of up to 18 clock cycles. Multi-channel alignment is based on the incoming A1/A2 bytes.

The alignment FIFO is always written from the SONET framer using the per channel recovered clock. The FIFO is always read using the local reference clock (FPGA_SYSCLK). For this reason when doing multi-channel alignment there must be 0 ppm between the transmit ORT8850 reference clock and the receiving ORT8850 reference clock. This can only be accomplished by using a single clock source for both the transmitting and receiving devices.

The alignment FIFO has several alarm and control indicators that are accessible via control and alarm registers available via the system bus or the MPI. The default alignment threshold values for the alignment FIFO are set in registers at 0x3000A and 0x3000B. Here the min and max threshold values can be programmed. The default min is set to 2 clocks and the max default is set to 15. If the alignment FIFO determines that these thresholds have been violated a per channel alarm bit will be set indicating that this channel has exceeded the threshold, as well as a FIFO out-of-sync alarm bit to indicate the channel is not longer in sync with the reset of the alignment group.

The incoming data can be considered as 4 STS-12 channels (A, B, C, and D) per quad. Thus we have STS-12 channels AA to AD from quad A of the STM and STS-12 channels BA to BD of quad B. The 8 channels of parallel SONET data can be grouped into an alignment group by 2, by 4 or all 8 channels. As the serial data is run through the backplane and SERDES the parallel data can be slightly varied. The alignment FIFO can absorb this difference in the channels and create a byte aligned grouping.

These streams can be frame aligned in the following patterns. Streams can be aligned on a twin STS-12 basis as shown in Figure 18. In STS-48 mode, all four STS-12s of each STM quad are aligned with each other (i.e. AA, AB, AC, AD) as shown in Figure 19. Optionally in STS-48 mode all eight STS-12s (STMs A and B) can be aligned which allows hitless switching since all streams will be byte aligned (Figure 20). Multiple ORT8850 devices can be aligned with each other using a common system frame pulse to enable STS-192 or higher modes.

Figure 18. Twin Channel Alignment

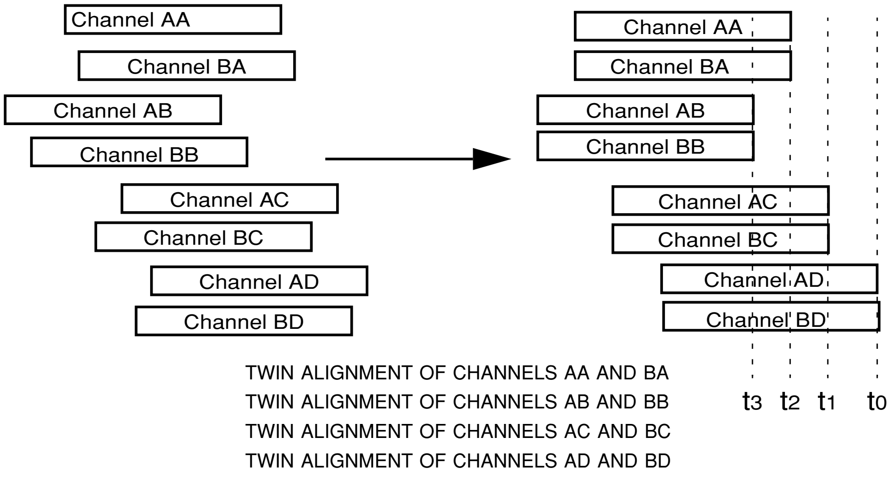
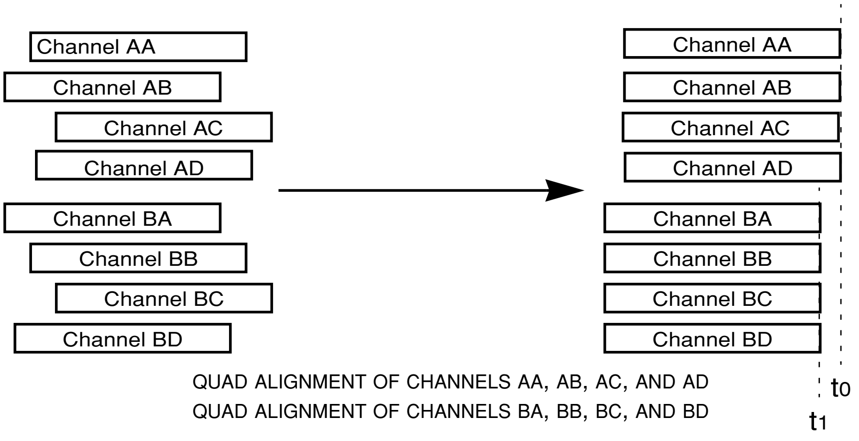


Figure 19. Alignment of SERDES Quads A and B



condition. It is also possible during operation for the channel to go into OOF. This may occur due to the removal of either the frame pulse or the cable. If this is the case, AND is is part of a multi-channel alignment group, the realignment procedure must be re-executed once the channel goes back into frame.

When a channel goes from the OOF state to the In-Frame state the OOF alarm bit is set per channel. The OOF alarm bit is a per channel bit contained in the channel alarm register. It takes the receiver at least 4 full SONET frames for the state machine to declare the In-Frame state. When the OOF bit is high the channel is in OOF. When the OOF bit changes to a ‘0’ then the channel is back in frame and the realignment procedure should be executed.

Table 11 lists the register values to set up the ORT8850 for alignment FIFO sync realignment. The order is specific. The values are given from the PowerPC point of view. If using the MPI to write data to the ORT8850, the value given in the table is the value that should be used. If using the UMI of the system bus, the data value would need to be byte flipped. The following setup procedures should be followed after the enabled channels have a valid frame pulse, and are in the Frame state:

Table 11. Alignment FIFO Synch Realignment

Register Address	Value (Binary)	Description
0x30020, bit 6	1	Force AIS-L in all channels of the group to be synchronized.
0x30038, bit 6	1	
0x30050, bit 6	1	
0x30068, bit 6	1	
0x30080, bit 6	1	
0x30098, bit 6	1	
0x300B0, bit 6	1	
0x300C8, bit 6	1	
Wait for 4 SONET Frames (~500µs)		
0x30017, specific bits	1	Issue FIFO realignment commands.
0x30018, specific bits	1	
Wait for Another 4 SONET Frames (~500µs)		
0x30017, specific bits	0	Clear FIFO alignment command register bits written in previous steps.
0x30018, specific bits	0	
0x30020, bit 6	0	Release AIS-L in all channels of the group to allow normal data flow through the reveiver.
0x30038, bit 6	0	
0x30050, bit 6	0	
0x30068, bit 6	0	
0x30080, bit 6	0	
0x30098, bit 6	0	
0x300B0, bit 6	0	
0x300C8, bit 6	0	

RX Serial TOH Processing

Transport overhead is extracted from the receive data stream by the TOH extract block. The incoming data gets loaded into a 36-byte shift register on the system clock domain. This, in turn, is clocked onto the TOH clock domain at the start of the SPE time, where it can be clocked out.

The TOH processor is responsible for serializing all received TOH bytes of each channel through that channel's corresponding serial TOH data port. The TOH serial ports are synchronized to the TOH clock (the same clock that is being used by the serial ports on the transmitter side). This free-running TOH clock is provided to the core by external circuitry and operates at a minimum frequency of 25 MHz and a maximum frequency of 77.76 MHz. Data is transferred over serial links in a bursty fashion as controlled by the RX TOH clock enable signal, and is common phases (i.e., received and system) are determined. This latch point is then stable unless the relative framing changes and the received H byte times collide with the system F1 or E2 times, in which case the latch point would be switched to the collision-free byte time.

There is no restriction on how many or how often increments and decrements are processed. Any received increment or decrement is immediately passed to the generator for implementation regardless of when the last pointer adjustment was made. The responsibility for meeting the SONET criteria for maximum frequency of pointer adjustments is left to an upstream pointer processor.

Receive Bypass Options

Not all of the blocks in the receive direction are required to be used. The following bypass options are valid in the receive (backplane → FPGA) direction:

- STM Pointer Mover bypass:
  - In this mode, data from the alignment FIFOs is transferred to the FPGA logic. All channels are synchronous to the FPGA_SYSCLK signals driven to the FPGA logic, as is also the case when the pointer mover is not bypassed. During bypass SPE, C1J1, and data parity signals are not valid. When the pointer mover is bypassed, eight frame pulses (DOUTxx_FP) from aligned channels are provided by the embedded core to the FPGA.
  - When the pointer mover is used, the FPGA logic provides the frame pulse on the LINE_FP (recall: there is only one LINE_FP just like there is only one SYS_FP) signal essential for the Pointer Mover to move the data. The FPGA gets eight channels of SONET data with the A1 byte position of each channel of the TOH arbitrarily offset from the LINE_FP. The DOUTxx_FP signals are not valid when the pointer mover is used.
- STM Pointer Mover and Alignment FIFO bypass:
  - In this mode, data from the framer block is transferred to the FPGA logic. All channels supply data and frame pulses synchronous with their individual recovered clock (CDR_CLK_xx) per channel. During bypass, SPE, C1J1, and data parity signals are not valid. Additionally, no serial TOH_OUT_xx data and frame pulse signals will be available. The DOUTxx_FP signals are aligned with the A1 byte position of each channel, as shown in Figure 26.

Figure 26. Pointer Mover and Alignment FIFO Bypass Timing

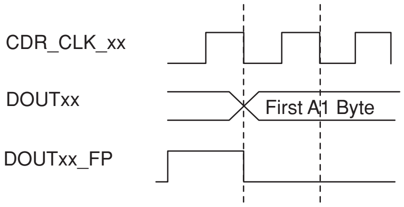


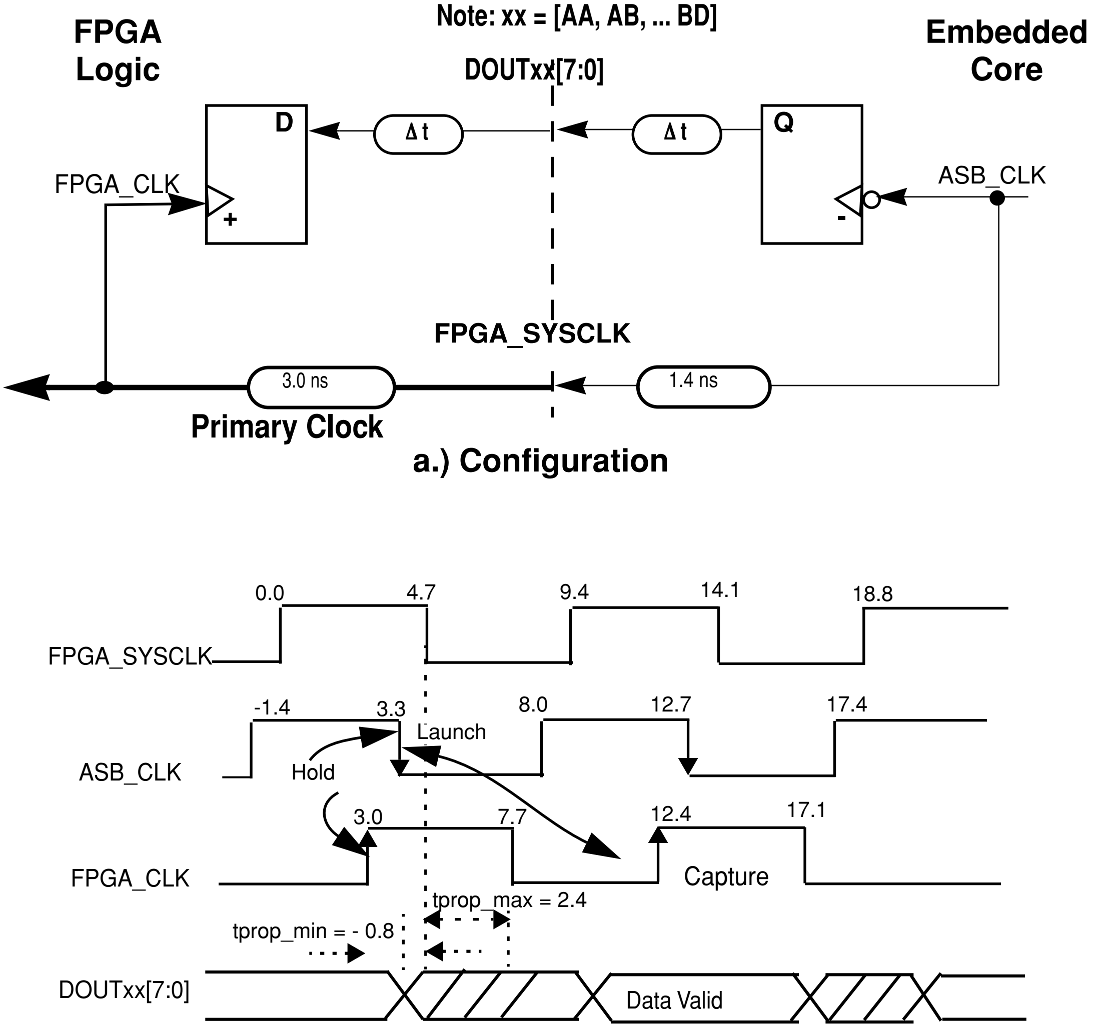
Table 14 shows the register settings to enable the bypass modes.

Table 14. Register Settings for Bypass Mode

Register Address	Value	Description
0x3000C	0x04	Turn off the SONET scrambler/descrambler
0x30020	0x07	Channel AA in functional mode
0x30038	0x07	Channel AB in functional mode
0x30050	0x07	Channel AC in functional mode
0x30068	0x07	Channel AD in functional mode
0x30080	0x07	Channel BA in functional mode
0x30098	0x07	Channel BB in functional mode
0x300B0	0x07	Channel BC in functional mode
0x300C8	0x07	Channel BD in functional mode

In the case shown in Figure 28 the alignment FIFO is used and all timing is with respect to the single reference clock, which is routed through the FPGA as a primary clock. The capturing clock edge occurs after the launch of the next data byte, so hold time margin is of concern and an acceptably margin should be verified. Launched data has nearly a full clock period to become stable at the capture latch, so setup margin should not be a problem. Moving the capture to the rising clock edge might give a setup time margin problem.

Figure 28. Half Cycle, Alignment Mode Output Configuration and Timing (-1 Speed Grade)



a.) Configuration

b.) Timing (times in ns.)

Table 19. Memory Map Descriptions (Continued)

(0x) Absolute Address	Bit	Type	Name	Reset Value (0x)	Description
30012	[0]	R	frame offset error flag	0	If in the receive direction the phase offset between any two channels exceeds 17 bytes, then a frame offset error event will be issued. This condition is continuously monitored. Write a “1” to clear this bit
	[1]	R	write to locked register error flag	0	If the core memory map has not been unlocked (by writing to the lock registers), and any address other than the lockreg registers or scratch pad register is written to, then a “write to locked register” event will be generated. Write a “1” to clear this bit
	[2-7]	-	Not Used	N/A	
30013	[0]	R/W	frame offset error enable	0	Frame offset error flag enable. 0 = not enable 1 = enable
	[1]	R/W	write to locked register for error enable	0	Write to locked register error flag enable 0 = not enable 1 = enable
	[2-7]	-	Not Used	0	
30014	[0]	R	BA alarm	0	Consolidation alarm for channel BA 0 = no alarm 1 = alarm
	[1]	R	BB alarm	0	Consolidation alarm for channel BB 0 = no alarm 1 = alarm
	[2]	R	BC alarm	0	Consolidation alarm for channel BC 0 = no alarm 1 = alarm
	[3]	R	BD alarm	0	Consolidation alarm for channel BD 0 = no alarm 1 = alarm
	[4-7]	R	Not Used	0	
30015	[0:7]	-	Not Used	00	
30016	[0:1]	R/W	STM A mode control	0	00 - Quad STS-12 or STS-48. 10 - Quad STS-3.
	[2:3]	R/W	STM B mode control	0	00 - Quad STS-12 or STS-48. 10 - Quad STS-3.
	[4-7]	-	Not Used	0	

Table 19. Memory Map Descriptions (Continued)

(0x) Absolute Address	Bit	Type	Name	Reset Value (0x)	Description
30019	[0:7]	-	Not Used	00	
Channel Register Blocks					
30020* 30038 30050 30068 30080 30098 300B0 300C8	[0]	R/W	AIS-L insert in OOF	0	0 = When RX direction OOF occurs, do not insert AIS-L. 1 = When RX direction OOF occurs, insert AIS-L.
	[1]	R/W	AIS-L control	0	0 = Do not force AIS-L insert 1 = Always force AIS-L insert
	[2]	R/W	TOH output parity error insert	0	0 = Do not insert a parity error 1 = Insert parity error in parity bit of receive TOH serial output for as long as this bit is set
	[3]	R/W	RX K1/K2 source select	0	0 = Set receive direction K1 K2 bytes to 0. 1 = Pass receive direction K1 K2 through pointer mover.
	[4]	R/W	DOUTxx bus parity error insert	0	0 = Do not insert parity error. 1 = Insert parity error in DOUTxx_PAR for as long as this bit is set.
	[5]	R/W	channel enable/disable control	0	0 = Power down CDR channels 1 = Functional mode.
	[6]	R/W	DOUTxx_EN	0	DOUTxx_EN signal
	[7]	R/W	TOH_EN	0	TOHxx_EN signal
30021* 30039 30051 30069 30081 30099 300B1 300C9	[0]	R/W	D9 source select	0	0 = Insert D9 from TOH_INxx 1 = Pass through D9 from DINxx
	[1]	R/W	D10 source select	0	0 = Insert D10 from TOH_INxx 1 = Pass through D10 from DINxx
	[2]	R/W	D11 source select	0	0 = Insert D11 from TOH_INxx 1 = Pass through D11 from DINxx
	[3]	R/W	D12 source select	0	0 = Insert D12 from TOH_INxx 1 = Pass through D12 from DINxx
	[4]	R/W	K1 K2 source select	0	0 = Insert K1, K2 from TOH_INxx 1 = Pass through K1, K2 from DINxx
	[5]	R/W	S1 M0 source select	0	0 = Insert S1, M0, from TOH_INxx 1 = Pass through S1 M0 of DINxx
	[6]	R/W	E1 F1 E2 source select		0 = Insert E1, F1, E2 from TOH_INxx on FPGA interface 1 = Pass through E1, F1, E2 TOH bytes of DINxx
	[7]	R/W	TOH source select	0	0 = Insert TOH from TOH_INxx on FPGA interface for transmit 1 = Pass through all TOH DINxx for transmit
30022* 3003A 30052 3006A 30082 3009A 300B2 300CA	[0:7]	R/W	D1~D8 source select	00	0 = Insert TOH for transmit from TOH_INxx from the FPGA interface. 1 = Pass through D1~D8 TOH bytes from DINxx.

Table 19. Memory Map Descriptions (Continued)

(0x) Absolute Address	Bit	Type	Name	Reset Value (0x)	Description
30023* 3003B 30053 3006B 30083 3009B 300B3 300CB	[0]	R/W	A1 A2 error insert command	0	0 = Do not insert error. 1 = Insert error for number of frames in register 0x3000C. The error insertion is based on a rising edge detector. As such, the control must be set to value 0 before trying to indicate a second A1, A2 corruption
	[1]	R/W	B1 error insert command	0.	0 = Do not insert error 1 = Insert error marked in register 0x3000F. The error insertion is based on a rising edge detector. As such, the control must be set to value 0 before trying to initiate a second B1 corruption.
	[2]	R/W	disable B1 insert	0	0 = B1 is inserted in the transmit direction by the SONET block 1 = B1 is not inserted in the transmit direction
	[3]	R/W	disable A1/A2 insert	0	0 = A1/A2 is inserted in the transmit direction by the SONET block 1 = A1/A2 is not inserted in the transmit direction
	[4-7]	-	Not Used	0	
30024* 3003C 30054 3006C 30084 3009C 300B4 300CC	[0:3]	R	concat indication 3, 6, 9, 12	0	The value 1 in any bit location indicates that STS# is in CONCAT mode. 0 = Not in concatenation mode or is the head of concatenated group 1 = indicates the channel is concatenated
	[4-7]	-	Not Used	0	
30025* 3003D 30055 3006D 30085 3009D 300B5 300CD	[0:7]	R	concat indication 1, 4, 7, 10, 2, 5, 8, 11	0	The value 1 in any bit location indicates that STS# is in CONCAT mode. 0 = Not in concatenation mode or is the head of concatenated group 1 = indicates the channel is concatenated
30026* 3003E 30056 3006E 30086 3009E 300B6 300CE	[0]	R	Channel alarm bit	0	Set when any of the alarms in the channel alarm register (0x30028) are set and the alarm is enabled. This alarm is enabled in 0x30027 bit 0 for channel AA etc.
	[1]	R	AIS-P flag	0	Set when any alarm for AIS-P is set and the corresponding enable is set.
	[2]	R	Pointer mover elastic store overflow flag	0	Set when the elastic store in the pointer mover write and read address is within 1 byte. Alarm enable is 0x30027 bit 2.
	[3-7]	-	Not Used	0	

Package Information

Table 34 summarizes the programmable I/O clock and power pins available to the ORT8850 devices.

Table 34. ORT8850 IO and Power Pin Summary

I/O or Power Type	ORT8850L	ORT8850H
User I/O Single Ended	278	297
User I/O Differential Pairs (LVDS, LVPECL)	129	129
Configuration	7	7
Dedicated Function	3	3
VDD15	48	48
VDD33	28	28
VDDIO	38	38
Vss	89	89
Single-Ended/Differential I/O per Bank		
Bank 0	64/32	68/32
Bank 1	47/20	47/20
Bank 2	ASIC I/O	ASIC I/O
Bank 3	ASIC I/O	ASIC I/O
Bank 4	ASIC I/O	ASIC I/O
Bank 5	44/18	44/18
Bank 6	76/32	76/32
Bank 7	55/27	62/27

There are some incompatibilities between the ORT8850H and ORT8850L due to the fact that the ORT8850L is a much smaller array and hence does not provide as many programmable IOs (PIOs). In order to allow pin-for-pin compatible board layouts that can accommodate either device, key compatibility issues include the following:

- **Unused Pins** Table 35 shows a list of bonded ORT8850H PIOs that are unused in the ORT8850L. As shown in the table, there are 19 balls that are not available in the ORT8850L, but are available in the ORT8850H. These user I/Os should not be used if an ORT8850L will be used.
- **Shared Control Signals on I/O Registers.** The ORCA Series 4 architecture shares clock and control signals between two adjacent I/O pads. If I/O registers are used, incompatibilities may arise between ORT8850L and ORT8850H when different clock or control signals are needed on adjacent package pins. This is because one device may allow independent clock or control signals on these adjacent pins, while the other may force them to be the same. There are two ways to avoid this issue.
  - Always keep an open bonded pin (non-bonded pins for the ORT8850L do not count) between pins that require different clock or control signals. Note that this open pin can be used to connect signals that do not require the use of I/O registers to meet timing.
  - Place and route the design in both the ORT8850H and ORT8850L to verify both produce valid designs. Note that this method guarantees the current design, but does not necessarily guard against issues that can occur when design changes are made that affect I/O registers.
  - **2X/4X I/O Shift Registers.** If 2X I/O shift registers or 4X I/O shift registers are used in the design, this may cause incompatibilities between the ORT880L and ORT8850H because only the A and C I/Os in a PIC support 2X I/O shift registers and only A I/Os supports 4X I/O shift register mode. A and C I/Os are shown in the following pinout tables under the I/O pad columns as those ending in A or C.
- **Edge Clock Input Pins.** The input buffers for fast edge clocks are only available at the C I/O pad. The C I/Os are shown in the following pinout tables under the I/O pad columns as those ending in C.

Table 36. ORT8850L and ORT8850H 680-Pin PBGAM Pinout (note: pins labeled "Reserved" should be left unconnected) (Continued)

BM680	VDDIO Bank	VREF Group	I/O	ORT8850L	ORT8850H	Additional Function	Pair
AJ3	6 (BL)	4	IO	PL26A	PL46A	—	—
AK2	6 (BL)	4	IO	PL27D	PL47D	PLL_CK7C/HPPLL	L9C_D0
AL1	6 (BL)	4	IO	PL27C	PL47C	PLL_CK7T/HPPLL	L9T_D0
AB20	—	—	VSS	VSS	VSS	—	—
AJ5	6 (BL)	4	IO	PL27B	PL47B	—	L10C_A0
AJ4	6 (BL)	4	IO	PL27A	PL47A	—	L10T_A0
AB21	—	—	VSS	VSS	VSS	—	—
AK3	—	—	I	PTEMP	PTEMP	PTEMP	—
AM1	6 (BL)	—	VDDIO6	VDDIO6	VDDIO6	—	—
AL2	—	—	IO	LVDS_R	LVDS_R	LVDS_R	—
AK4	—	—	VDD33	VDD33	VDD33	—	—
AB22	—	—	VSS	VSS	VSS	—	—
AK6	—	—	VDD33	VDD33	VDD33	—	—
AL5	6 (BL)	5	IO	PB2A	PB2A	DP2	L11T_D1
AN4	6 (BL)	5	IO	PB2B	PB2B	—	L11C_D1
AM2	6 (BL)	—	VDDIO6	VDDIO6	VDDIO6	—	—
AM5	6 (BL)	5	IO	PB2C	PB2C	PLL_CK6T/PPLL	L12T_D1
AK7	6 (BL)	5	IO	PB2D	PB2D	PLL_CK6C/PPLL	L12C_D1
AL6	6 (BL)	5	IO	PB3A	PB3C	—	L13T_D1
AN5	6 (BL)	5	IO	PB3B	PB3D	—	L13C_D1
AM4	6 (BL)	—	VDDIO6	VDDIO6	VDDIO6	—	—
AM6	6 (BL)	5	IO	PB3C	PB4C	VREF_6_05	L14T_D0
AL7	6 (BL)	5	IO	PB3D	PB4D	DP3	L14C_D0
AK8	6 (BL)	6	IO	PB4A	PB5C	—	L15T_D3
AP5	6 (BL)	6	IO	PB4B	PB5D	—	L15C_D3
AB32	—	—	VSS	VSS	VSS	—	—
AK9	6 (BL)	6	IO	PB4C	PB6C	VREF_6_06	L16T_D2
AN6	6 (BL)	6	IO	PB4D	PB6D	D14	L16C_D2
AM7	6 (BL)	6	IO	PB5A	PB7C	—	L17T_D1
AP6	6 (BL)	6	IO	PB5B	PB7D	—	L17C_D1
AN3	6 (BL)	—	VDDIO6	VDDIO6	VDDIO6	—	—
AL8	6 (BL)	7	IO	PB5C	PB8C	D15	L18T_D1
AN7	6 (BL)	7	IO	PB5D	PB8D	D16	L18C_D1
AM8	6 (BL)	7	IO	PB6A	PB9C	D17	L19T_D0
AL9	6 (BL)	7	IO	PB6B	PB9D	D18	L19C_D0
AL4	—	—	VSS	VSS	VSS	—	—
AP7	6 (BL)	7	IO	PB6C	PB10C	VREF_6_07	L20T_D0
AN8	6 (BL)	7	IO	PB6D	PB10D	D19	L20C_D0
AL10	6 (BL)	8	IO	PB7A	PB11C	D20	L21T_D2
AP8	6 (BL)	8	IO	PB7B	PB11D	D21	L21C_D2
AL11	6 (BL)	8	IO	PB7C	PB12C	VREF_6_08	L22T_D0
AM10	6 (BL)	8	IO	PB7D	PB12D	D22	L22C_D0

Table 36. ORT8850L and ORT8850H 680-Pin PBGAM Pinout (note: pins labeled "Reserved" should be left unconnected) (Continued)

BM680	VDDIO Bank	VREF Group	I/O	ORT8850L	ORT8850H	Additional Function	Pair
AL19	5 (BC)	2	IO	PB17D	PB25D	—	L7C_A0
AP20	5 (BC)	3	IO	PB18A	PB26C	—	L8T_D3
AK19	5 (BC)	3	IO	PB18B	PB26D	VREF_5_03	L8C_D3
AM15	5 (BC)	—	VDDIO5	VDDIO5	VDDIO5	—	—
AN20	5 (BC)	3	IO	PB18C	PB27A	—	—
Y21	—	—	VSS	VSS	VSS	—	—
AP21	5 (BC)	3	IO	PB19A	PB27C	—	L9T_D2
AL20	5 (BC)	3	IO	PB19B	PB27D	—	L9C_D2
Y22	—	—	VSS	VSS	VSS	—	—
AK20	5 (BC)	3	IO	PB19C	PB28A	—	—
AN21	5 (BC)	3	IO	PB20A	PB28C	PBCK1T	L10T_A0
AM21	5 (BC)	3	IO	PB20B	PB28D	PBCK1C	L10C_A0
AM20	5 (BC)	—	VDDIO5	VDDIO5	VDDIO5	—	—
AK21	5 (BC)	3	IO	PB20C	PB29A	—	—
AP22	5 (BC)	4	IO	PB21A	PB29C	—	L11T_D2
AL21	5 (BC)	4	IO	PB21B	PB29D	—	L11C_D2
AA15	—	—	VSS	VSS	VSS	—	—
AN22	5 (BC)	4	IO	PB21C	PB30A	—	—
AP23	5 (BC)	4	IO	PB22A	PB30C	—	L12T_A0
AN23	5 (BC)	4	IO	PB22B	PB30D	VREF_5_04	L12C_A0
AA13	—	—	VSS	VSS	VSS	—	—
AK22	5 (BC)	4	IO	PB22C	PB31C	—	L13T_A0
AL22	5 (BC)	4	IO	PB22D	PB31D	—	L13C_A0
AN24	5 (BC)	5	IO	PB23C	PB32C	—	L14T_D2
AK23	5 (BC)	5	IO	PB23D	PB32D	VREF_5_05	L14C_D2
AA14	—	—	VSS	VSS	VSS	—	—
AL23	5 (BC)	5	IO	PB24C	PB33C	—	L15T_D0
AM24	5 (BC)	5	IO	PB24D	PB33D	—	L15C_D0
AP25	5 (BC)	5	IO	PB25A	PB34C	—	L16T_A0
AN25	5 (BC)	5	IO	PB25B	PB34D	—	L16T_A0
AP26	5 (BC)	6	IO	PB25C	PB35A	—	—
AK25	5 (BC)	6	IO	PB26A	PB35C	—	L17T_A0
AN26	5 (BC)	6	IO	PB26B	PB35D	VREF_5_06	L17C_A0
AP27	5 (BC)	6	IO	PB26C	PB36A	—	—
AM25	5 (BC)	6	IO	PB27A	PB36C	—	L18T_D3
AK26	5 (BC)	6	IO	PB27B	PB36D	—	L18C_D3
N32	—	—	VSS	VSS	VSS	—	—
AL24	—	—	O	TXDAA_P_N	TXDAA_P_N	—	L1N_A0
AK24	—	—	O	TXDAA_P_P	TXDAA_P_P	—	L1P_A0
A32	—	—	VDD33	VDD33	VDD33	—	—
AN27	—	—	O	TXDAB_P_N	TXDAB_P_N	—	L2N_D0
AP28	—	—	O	TXDAB_P_P	TXDAB_P_P	—	L2P_D0

Table 36. ORT8850L and ORT8850H 680-Pin PBGAM Pinout (note: pins labeled "Reserved" should be left unconnected) (Continued)

BM680	VDDIO Bank	VREF Group	I/O	ORT8850L	ORT8850H	Additional Function	Pair
T32	—	—	O	TXDAA_W_N	TXDAA_W_N	—	L32N_D1
R34	—	—	O	TXDAA_W_P	TXDAA_W_P	—	L32P_D1
AM33	—	—	VDD33	VDD33	VDD33	—	—
U30	—	—	O	TXDAB_W_N	TXDAB_W_N	—	L33N_D0
T31	—	—	O	TXDAB_W_P	TXDAB_W_P	—	L33P_D0
V17	—	—	VSS	VSS	VSS	—	—
R33	—	—	O	TXDAC_W_N	TXDAC_W_N	—	L34N_D0
P34	—	—	O	TXDAC_W_P	TXDAC_W_P	—	L34P_D0
AM34	—	—	VDD33	VDD33	VDD33	—	—
P33	—	—	O	TXDAD_W_N	TXDAD_W_N	—	L35N_D0
N34	—	—	O	TXDAD_W_P	TXDAD_W_P	—	L35P_D0
V18	—	—	VSS	VSS	VSS	—	—
T30	—	—	O	Reserved	Reserved	—	L36N_D0
R31	—	—	O	Reserved	Reserved	—	L36P_D0
AN32	—	—	VDD33	VDD33	VDD33	—	—
P32	—	—	O	Reserved	Reserved	—	L37N_D1
R30	—	—	O	Reserved	Reserved	—	L37P_D1
V19	—	—	VSS	VSS	VSS	—	—
N33	—	—	O	TXDBA_W_N	TXDBA_W_N	—	L38N_D0
M34	—	—	O	TXDBA_W_P	TXDBA_W_P	—	L38P_D0
AP32	—	—	VDD33	VDD33	VDD33	—	—
P31	—	—	O	TXDBB_W_N	TXDBB_W_N	—	L39N_D1
M33	—	—	O	TXDBB_W_P	TXDBB_W_P	—	L39P_D1
V34	—	—	VSS	VSS	VSS	—	—
N31	—	—	O	TXDBC_W_N	TXDBC_W_N	—	L40N_D0
P30	—	—	O	TXDBC_W_P	TXDBC_W_P	—	L40P_D0
L33	—	—	O	TXDBD_W_N	TXDBD_W_N	—	L41N_D0
K34	—	—	O	TXDBD_W_P	TXDBD_W_P	—	L41P_D0
W16	—	—	VSS	VSS	VSS	—	—
M31	—	—	I	Reserved	Reserved	—	L42N_D0
L32	—	—	I	Reserved	Reserved	—	L42P_D0
K33	—	—	I	Reserved	Reserved	—	—
W17	—	—	VSS	VSS	VSS	—	—
N30	—	—	VDDA_PDI	Reserved	Reserved	—	—
L30	—	—	VSSA_PDI	Reserved	Reserved	—	—
W18	—	—	VSS	VSS	VSS	—	—
M30	—	—	I	Reserved	Reserved	—	L43N_D0
L31	—	—	I	Reserved	Reserved	—	L43P_D0
W19	—	—	VSS	VSS	VSS	—	—
J34	—	—	I	Reserved	Reserved	—	L44N_D1
K32	—	—	I	Reserved	Reserved	—	L44P_D1
J33	—	—	I	Reserved	Reserved	—	—

Table 36. ORT8850L and ORT8850H 680-Pin PBGAM Pinout (note: pins labeled "Reserved" should be left unconnected) (Continued)

BM680	VDDIO Bank	VREF Group	I/O	ORT8850L	ORT8850H	Additional Function	Pair
C11	0 (TL)	2	IO	PT9A	PT13C	MPI_TEA_N	L6T_D0
B10	0 (TL)	3	IO	PT8D	PT12D	—	L7C_D0
A9	0 (TL)	3	IO	PT8C	PT12C	—	L7T_D0
C10	0 (TL)	3	IO	PT8B	PT11D	VREF_0_03	L8C_D0
B9	0 (TL)	3	IO	PT8A	PT11C	—	L8T_D0
A8	0 (TL)	3	IO	PT7D	PT10D	D0	L9C_D2
D10	0 (TL)	3	IO	PT7C	PT10C	TMS	L9T_D2
B1	—	—	VSS	VSS	VSS	—	—
C9	0 (TL)	4	IO	PT7B	PT9D	A20/MPI_BDIP_N	L10C_D0
B8	0 (TL)	4	IO	PT7A	PT9C	A19/MPI_TSZ1	L10T_D0
A7	0 (TL)	4	IO	PT6D	PT8D	A18/MPI_TSZ0	L11C_D4
E12	0 (TL)	4	IO	PT6C	PT8C	D3	L11T_D4
B3	0 (TL)	—	VDDIO0	VDDIO0	VDDIO0	—	—
D9	0 (TL)	4	IO	PT6B	PT7D	VREF_0_04	L12C_D0
C8	0 (TL)	4	IO	PT6A	PT7C	—	L12T_D0
E11	0 (TL)	5	IO	PT5D	PT6D	D1	L13C_D3
B7	0 (TL)	5	IO	PT5C	PT6C	D2	L13T_D3
B2	—	—	VSS	VSS	VSS	—	—
A6	0 (TL)	5	IO	PT5B	PT5D	—	L14C_D2
D8	0 (TL)	5	IO	PT5A	PT5C	VREF_0_05	L14T_D2
C7	0 (TL)	5	IO	PT4D	PT4D	TDI	L15C_D1
A5	0 (TL)	5	IO	PT4C	PT4C	TCK	L15T_D1
C1	0 (TL)	—	VDDIO0	VDDIO0	VDDIO0	—	—
E10	0 (TL)	5	IO	PT4B	PT4B	—	L16C_D2
D7	0 (TL)	5	IO	PT4A	PT4A	—	L16T_D2
A4	0 (TL)	6	IO	PT3D	PT3D	—	L17C_D4
E9	0 (TL)	6	IO	PT3C	PT3C	VREF_0_06	L17T_D4
B33	—	—	VSS	VSS	VSS	—	—
B6	0 (TL)	6	IO	PT3B	PT3B	—	L18C_A0
C6	0 (TL)	6	IO	PT3A	PT3A	—	L18T_A0
B5	0 (TL)	6	IO	PT2D	PT2D	PLL_CK1C/PPLL	L19C_D1
D6	0 (TL)	6	IO	PT2C	PT2C	PLL_CK1T/PPLL	L19T_D1
C2	0 (TL)	—	VDDIO0	VDDIO0	VDDIO0	—	—
C5	0 (TL)	6	IO	PT2B	PT2B	—	L20C_D0
B4	0 (TL)	6	IO	PT2A	PT2A	—	L20T_D0
E8	—	—	O	PCFG_MPI_IRQ	PCFG_MPI_IRQ	CFG_IRQ_N/MPI_IRQ_N	—