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	10368
Total RAM Bits	113664
Number of I/O	372
Number of Gates	643000
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/ort82g5-1f680c

Table of Contents

Embedded Function Features

- High-speed SERDES with programmable serial data rates over the range 0.6 to 3.7 Gbps. Operation has been demonstrated on design tolerance devices at 3.7 Gbps across 26 in. of FR-4 backplane and at 3.125 Gbps across 40 in. of FR-4 backplane across temperature and voltage specifications.
- Asynchronous operation per receive channel with the receiver frequency tolerance based on one reference clock per block channels (separate PLL per channel).
- Ability to select full-rate or half-rate operation per transmit or receive channel by setting the appropriate control registers.
- Programmable one-half amplitude transmit mode for reduced power in chip-to-chip application.
- Transmit preemphasis (programmable) for improved receive data eye opening.
- 32-bit (8b/10b) or 40-bit (raw data) parallel internal bus for data processing in FPGA logic.
- Provides a 10 Gbps backplane interface to switch fabric. Also supports multiple port cards at 2.5 Gbps.
- 3.125 Gbps SERDES compliant with XAUI serial data specification for 10 G Ethernet applications with protection.
- IEEE 802.3ae compliant XAUI transceiver. Includes embedded IEEE 802.3ae-based XAUI link state machine.
- Compliant to FC-0 specification for 1 Gbps, 2Gbps, 10 Gbps (FC-XAUI) modes. Includes Fibre Channel link state machine.
- High-Speed Interface (HSI) function for clock/data recovery serial backplane data transfer without external clocks.
- SERDES has low-power CML buffers. Support for 1.5V/1.8V I/Os. Allows use with optical transceiver, coaxial copper media, shielded twisted pair wiring or high-speed backplanes such as FR-4.
- Power down option of SERDES HSI receiver or transmitter on a per-channel basis.
- Automatic lock to reference clock in the absence of valid receive data.
- High-speed and low-speed loopback test modes.
- Requires no external component for clock recovery and frequency synthesis.
- SERDES characterization pins available to control/monitor the internal interface to one SERDES block (ORT82G5 only).
- SERDES HSI automatically recovers from loss-of-clock once its reference clock returns to normal operating state.
- Built-in boundary scan (IEEE ® 1149.1 and 1149.2 JTAG) for the programmable I/Os, not including the SERDES interface.
- FIFOs can align incoming data either across all eight channels (ORT82G5 only), across one or two groups of four channels, or across two or four groups of two channels. Alignment is done either using comma characters or by using the /A/ character in XAUI mode. Optionally, the alignment FIFOs can be bypassed for asynchronous operation between channels. (Each channel includes its own clock and frame pulse or comma detect.)
- Addition of two 4K x 36 dual-port RAMs with access to the programmable logic.
- The ORT82G5 is pinout compatible to the ORCA ORSO82G5 SONET backplane driver FPSC. The ORT42G5 is pin compatible to the ORSO42G5.

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 FF, 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, and 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, block-port Embedded Block RAMs, universal programmable Phase-Locked Loops, and the addition of highly tuned networking specific Phase-locked Loops. These functional blocks allow for 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 860 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.

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, is provided by the MPI. 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).

Following the definitions and conventions used in defining the 8b/10b coding rules, each valid coded character has a name corresponding to its 8-bit binary value:

- Dxx.y for data characters
- Kxx.y for special characters
- xx = the 5-bit input value, base 10, for bits ABCDE
- y = the 3-bit input value, base 10, for bits FGH

An 8b/10b encoder is designed to maintain a neutral average disparity. Disparity is the difference between the number of 1s and 0s in the encoded word. Neutral disparity indicates the number of 1s and 0s are equal. Positive disparity indicates more 1s than 0s. Negative disparity indicates more 0s than 1s. The average disparity determines the DC component of the signals on the serial line. Running disparity is a record of the cumulative disparity of every encoded word, and is tracked by the encoder.

In order to maintain neutral disparity, two different codings are defined for each data value. The 8b/10b encoder in the transmit path selects between (+) and (-) encoded word based on calculated disparity of the present data to maintain neutral disparity

In the receive path, the clock and data recovery blocks retime the incoming data and 8b/10b decoders generate 8-bit data based on the received 10-bit data. A sequence of valid 8b/10b coded characters has a maximum run length of 5 bits (i.e., 5 consecutive ones or 5 consecutive zeros before a mandatory bit transition). This assures adequate transitions for robust clock recovery.

The recovered data is aligned on a 10-bit boundaries by detecting and aligning to special characters in the incoming data stream. Data is word aligned using the comma (/K/) character. A comma character is a special character that contains a unique pattern (0011111 or its complement 1100000) in the 10-bit space that makes it useful for delimiting word boundaries. The special characters K28.1, K28.5 and K28.7 contain this comma sequence and are treated as valid comma characters by the SERDES.

The following table shows all of the valid special characters. All of the special characters are made available to the FPGA logic; however only the comma characters are used by the SERDES logic. The different codings that are possible for each data value are shown as encoded word (+) and encoded word (-). The table also illustrates the 8b/10b bit labeling convention. The bit positions of the 8-bit characters are labeled as H,G,F,E,D,C,B and A and the bit positions of the 10-bit encoded characters are labeled as a, b, c, d, e, i, f, g, h, and j. The encoded words are transmitted serially with bit ‘a’ transmitted first and bit ‘j’ transmitted last.

Table 2. Valid Special Characters

K Character	HGF EDCBA 765 43210	K Control	Encoded Word (–) abcdei fghj	Encoded Word (+) abcdei fghj
K28.0	000 11100	1	001111 0100	110000 1011
K28.1 /comma/	001 11100	1	001111 1001	110000 0110
K28.2	010 11100	1	001111 0101	110000 1010
K28.3 /A/	011 11100	1	001111 0011	110000 1100
K28.4	100 11100	1	001111 0010	110000 1101
K28.5 /comma/	101 11100	1	001111 1010	110000 0101
K28.6	110 11100	1	001111 0110	110000 1001
K28.7 /comma/	111 11100	1	001111 1000	110000 0111
K23.7	111 10111	1	111010 1000	000101 0111
K27.7	111 11011	1	110110 1000	001001 0111
K29.7	111 11101	1	101110 1000	010001 0111
K30.7	111 11110	1	011110 1000	100001 0111

Since this effect is predictable for a given type of PCB material, it is possible to compensate for this effect in two ways - transmitter preemphasis and receiver equalization. Each of these techniques boosts the high frequency components of the signal but transmit preemphasis is preferred due to the ease of implementation and the better power utilization. It also gives a better signal-to-noise ratio because receiver equalization amplifies both the signal and the noise at the receiver

Applying too much preemphasis when it is not required, for example when driving a short backplane path, will also degrade the data eye opening at the receiver. In the ORT42G5 and ORT82G5 the degree of transmit preemphasis can be programmed with a two-bit control from the microprocessor interface as shown in Table 3. The high-pass transfer function of the preemphasis circuit is given by the following equation, where the value of a is shown in Table 3.

$$H(z) = (1 - az^{-1}) \quad (1)$$

Table 3. Preemphasis Settings

PE1	PE0	Amount of Preemphasis (a)
0	0	0% (No Preemphasis)
0	1	12.5%
1	0	12.5%
1	1	25%

Receive Path (Backplane to FPGA) Logic

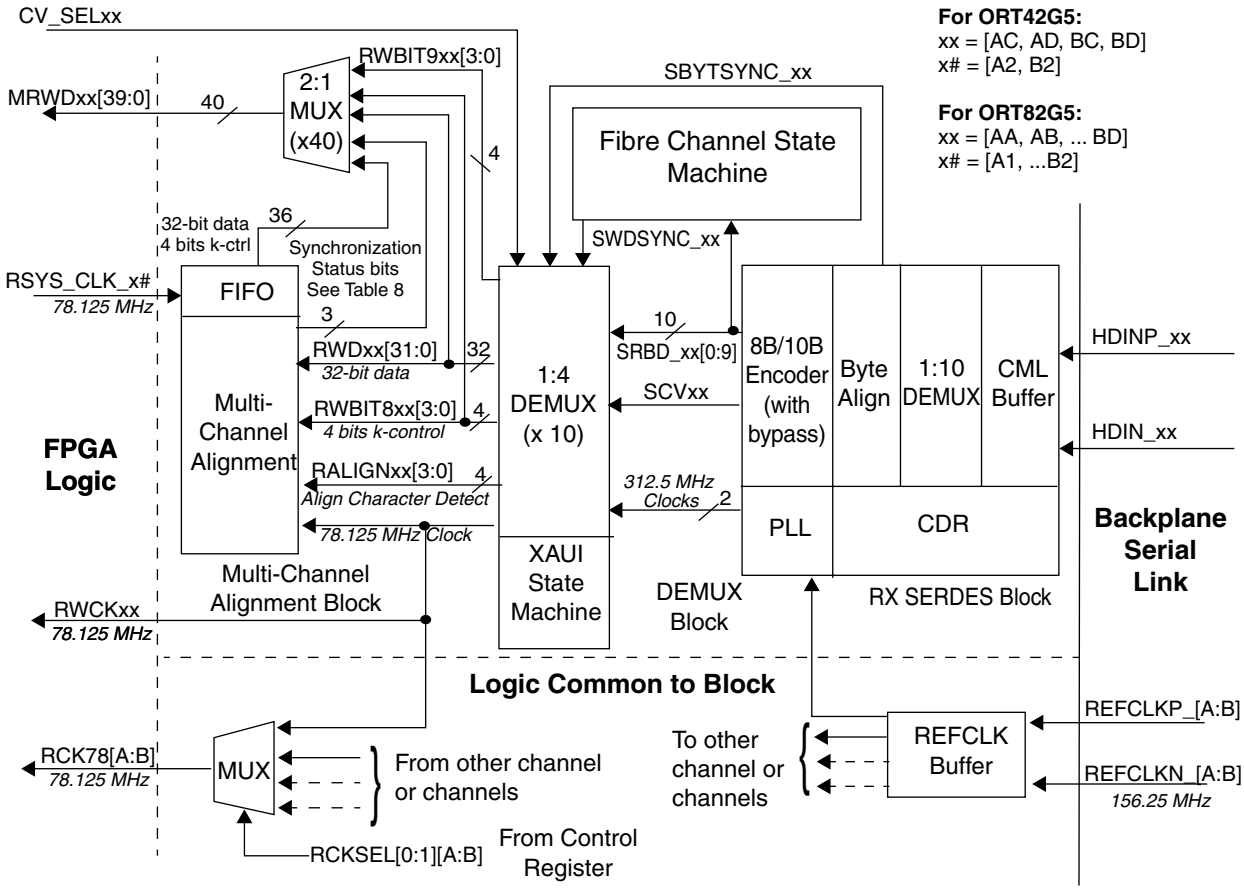
The receiver section receives high-speed serial data at the external differential CML input pins. These data are fed to the clock recovery section which generates a recovered clock and retimes the data. Therefore the receive clocks are asynchronous between channels. The retimed data are deserialized and presented as an 8-bit decoded or a 10-bit unencoded parallel data on the output port. The receiver also optionally recognizes comma characters, detects code violations and aligns the bit stream to the proper word boundary.

As shown in Figure 6, the basic blocks in the receive path include:

Receive SERDES Block

- CML input buffer
- Receive PLL
- 1:10 demultiplexer (DEMUX)
- Clock and Data Recovery (CDR) section
- 10b/8b decoder
- 1:4 demultiplexer and Embedded Core/FPGA interface
- 1:4 DEMUX
- Low speed parallel Embedded Core/FPGA logic interface
- Multi-channel alignment logic

Figure 6. Basic Logic Blocks, Receive Path, Single Channel (Typical Reference Clock Frequency)



Each channel provides its own received clock, received data and K-character detect signals to the FPGA logic. Incoming data from multiple channels can be aligned using comma (/K/) characters or /A/ character (as specified either in Fibre Channel specifications or in IEEE 802.3ae for XAUI based interfaces). If the 8b/10b decoders are bypassed, then 40-bit data streams are passed to the FPGA logic. No channel alignment can be done in this 8b/10b bypass mode.

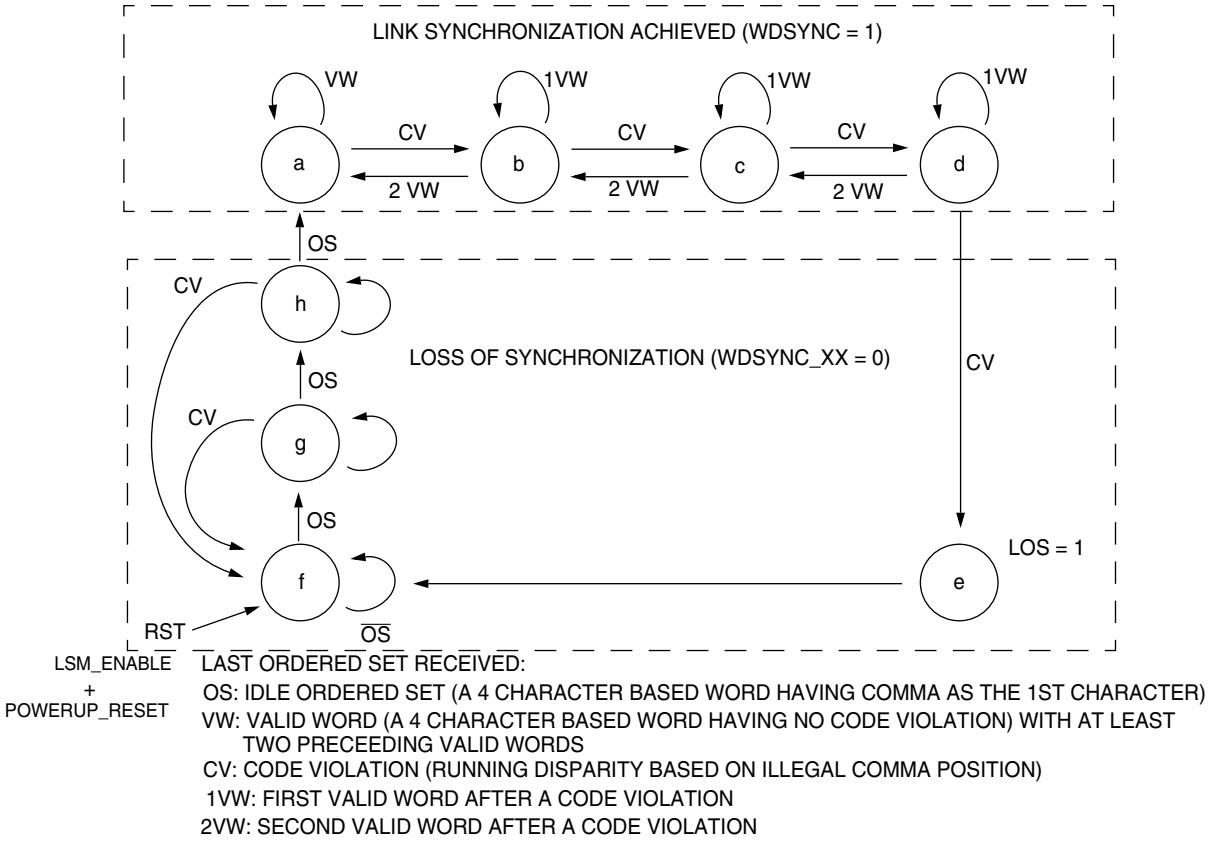
Detailed descriptions of data synchronization, of the SERDES, DEMUX and Multi-Channel Alignment blocks and of the Fibre Channel and XAUI state machines are given in following sections. Receive clock distribution is described in a later section of this data sheet.

Synchronization

The SERDES RX logic performs four levels of synchronization on the incoming serial data stream. Each level builds upon the previous, providing first bit, then byte (character), then channel (32-bit word), and finally multi-channel alignment. Each step is described functionally in the following paragraphs. The details of the logical implementations are described in subsequent sections.

Bit alignment is the task of the Clock/Data Recovery (CDR) block. This block utilizes a PLL that locks to the transitions in the incoming high-speed serial data stream, and outputs the extracted clock as well as the data. If the PLL is unable to lock to the serial data stream, it instead locks to REFCLK[A:B] to stabilize the voltage-controlled oscillator (VCO), and periodically switches back to the serial data stream to again attempt synchronization. This process continues until a valid input data stream is detected and lock is achieved. The CDR can maintain lock on data as long as the input data stream contains an adequate data “eye” (i.e., jitter is within specification) and the maximum data stream run length is not exceeded.

Figure 9. Fibre Channel Link State Machine State Diagram



XAUI Link Synchronization Function

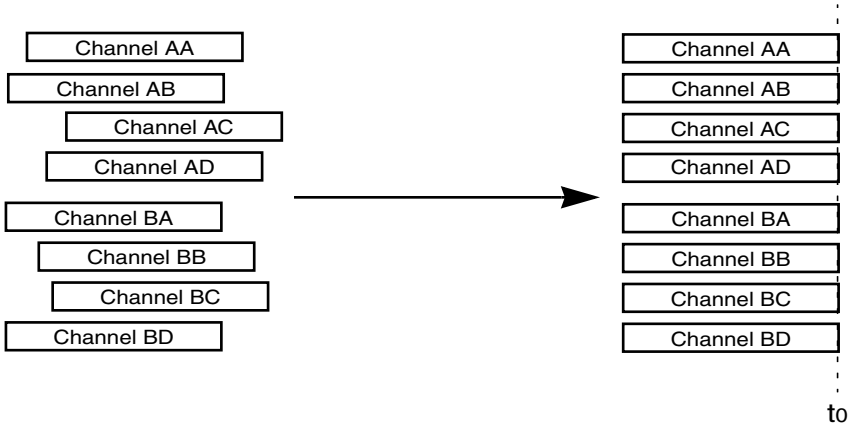
For each lane, the receive section of the XAUI link state machine incorporates a synchronization state machine that monitors the status of the 10-bit alignment. A 10-bit alignment is done in the SERDES based on a comma character such as K28.5. A comma (0011111 or its complement 1100000) is a unique pattern in the 10-bit space that cannot appear across the boundary between any two valid 10-bit code-groups. This property makes the comma useful for delimiting code-groups in a serial stream.This mechanism incorporates a hysteresis to prevent false synchronization and loss of synchronization due to infrequent bit errors. For each lane, the sync_complete signal is disabled until the lane achieves synchronization. The synchronization state diagram is shown in Figure 10. This state machine is modeled after draft *IEEE* 802.3ae, version 2.1 but will also operate with version 4.1 implementations. Table 4 and Table 5 describe the state variables used in Figure 10. The XAUI state machine does not have any control over the SERDES byte aligner. It is the user's responsibility to control the byte aligner through software access of register map addresses 30800 and 30900.

Note that it takes four idle ordered sets (e.g. K28.5, Dxx.y, Dxx.y, Dxx.y) to bring the state machine from a loss_of_sync to a synch_acq'd_1 state. When back-to-back commas are used instead, it takes a total of five commas to achieve the same result as with idle ordered sets.

Table 4. XAUI Link Synchronization State Diagram – Functions

Function	Description
sync_complete	Indication that alignment code-group alignment has been established at the boundary indicated by the most recently received comma.
cg_comma	Indication that a valid code-group, with correct running disparity, containing a comma has been received.
cg_good	Indication that a valid code-group with the correct running disparity has been received.
cg_bad	Indication that an invalid code-group has been received.
no_comma	Indication that comma timer has expired. The timer is initialized upon receipt of a comma.

Figure 15. Alignment of all Eight SERDES Channels.



Note that any channel within an alignment group can be removed from that alignment group by setting FMPU_STR_EN_xx to 0. The disabling of any channel(s) within an alignment group will not affect the operation of the remaining active channels. If the active channels are synchronized, that synchronization will be maintained and no data loss will occur.

For every alignment group, there are both an OVFL and an OOS status register bit. The OVFL bit is set when alignment FIFO overflow occurs. The OOS bit is flagged when the down counter in the synchronization algorithm has reached a value of 0 and alignment characters from all channels within an alignment group have not been received. In the memory map section for the ORT42G5 the bits indicating OOS and OVFL are referred to as SYNC2_[A:B]_OOS and SYNC4_OOS and the bits indicating OVFL are SYNC2_[A:B]_OVFL and SYNC4_OVFL.

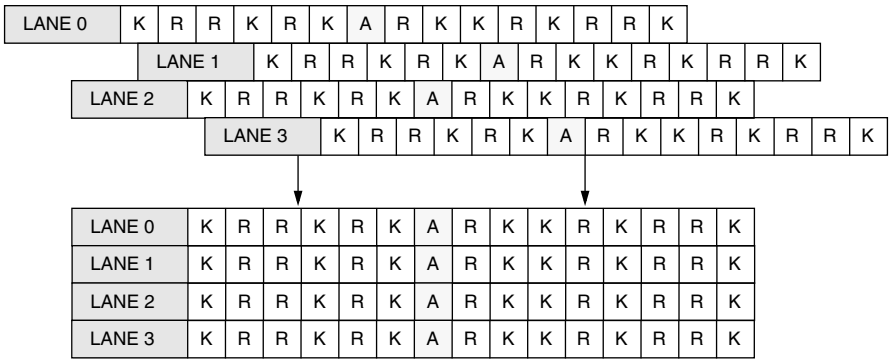
In the memory map section for the ORT82G5, the bits indicating OOS and OVFL are referred to as SYNC2_[A1,A2,B1,B2]_OOS, SYNC4_[A:B]_OOS and SYNC8_OOS and the bits indicating OVFL are SYNC2_[A1,A2,B1,B2]_OVFL, SYNC4_[A:B]_OVFL and SYNC8_OVFL.

Alignment can also be done between the receive channels on two ORT82G5 devices. Each of the two devices needs to provide its aligned K_CTRL or other alignment character to the other device, which will delay reading from a second alignment FIFO until all channels requesting alignment on the current device and all channels requesting alignment on the other device are aligned (as indicated on the K_CTRL character). These second alignment FIFOs will be implemented in FPGA logic on the ORT82G5. This scheme also requires that the reference clock for both devices be driven by the same signal.

XAUI Lane Alignment Function (Lane Deskew)

In XAUI mode, the receive section in each lane uses the /A/ code group to compensate for lane-to-lane skew. The mechanism restores the timing relationship between the 4 lanes by lining up the /A/ characters into a column. Figure 16 shows the alignment of four lanes based on /A/ character. A minimum spacing of 16 code-groups implies that at least ± 80 bits of skew compensation capability should be provided, which the devices significantly exceed.

Figure 16. Deskew Lanes by Aligning /A/ Columns



Mixing Half-rate, Full-rate Modes

When channel alignment is enabled, all receive channels within an alignment group should be configured at the same rate. For example, in the ORT82G5 channels AA, AB, can be configured for twin alignment and full-rate mode, while channels AC, AD that form an alignment group can be configured for half-rate mode. In block alignment mode, each receive block can be configured in either half or full-rate mode.

When channel alignment is disabled within a block, any receive channel within the block can be used in half-rate or full-rate mode. The clocking strategy for half-rate mode in both scenarios (channel alignment enabled or disabled) is described in the Reference Clocks and Internal Clock Distribution sections later in this data sheet.

Multi-channel Alignment Configuration

ORT42G5 Configuration

At startup, the legacy SERDES channel logic must be powered down and removed from any multi-channel alignment groups:

- Setting bit 1 to one in registers at locations 30002, 30012, 30102, 30112, 30003, 30013, 30103 and 30113 powers down the legacy logic. (Note that the reset value for these bits is 0.)
- Setting bits 4 and 5 to zero (reset condition) in the register at locations 30810 and 30910 removes the legacy logic from any alignment group.

Register settings for multi-channel alignment are shown in Table 6.

Table 6. Multichannel Alignment Modes

Register Bits FMPU_SYNMODE_[A:B][0:7]	Mode
00000000	No multichannel alignment.
00001010	Twin channel alignment.
00001111	Four channel alignment.

To align two channels in SERDES A:

- FMPU_SYNMODE_A = 00001010 (Register Location 30811)

To align two channels in SERDES B:

- FMPU_SYNMODE_B = 00001010 (Register Location 30911)

To align all four channels:

- FMPU_SYNMODE_A = 00001111 (Register Location 30811)

Embedded Core/FPGA Interface

This block provides the data formatting and receive data and clock signal transfers between the Embedded Core and the FPGA Logic. There are also control and status registers in the FPGA portion of the chip which contain bits to control the receive logic and to record status. These are described in later sections of this data sheet and communicate with the core using the System Bus.

The demultiplexed, receive word outputs to the FPGA are shown in Figure 6. These are each 40 bits wide. There are eight of these interfaces, one for each SERDES channel. Each consist of four groups of 10-bit data or four groups of decoded information depending on setting of 8b10bR_xx control register bits.

Each 10-bit group of decoded information includes 8 bits of data and a 1 bit K_CTRL indicator derived from the received data and a tenth bit of status information. The function of the tenth bit varies from group to group and includes code violation, Out of Synchronization (OOS) indicators and the CH24_SYNC24_xx and CH248_SYNC_xx status bits. CH24_SYNC or CH248_SYNC_xx indicates the status of multi-channel alignment of channel xx and are high when the count for the multi-channel alignment block reaches zero regardless of whether or not multi-channel alignment is successful. The mapping of the 10-bit groups to the MRWD_xx[39:0] bits output to the FPGA logic is summarized in Table 8. The various functions of the bits that vary from channel to channel, i.e., bits 29 and 19, are also described in Table 9 and Table 10.

Table 8. Definition of Bits of MRWDxx[39:0]

	8b10bR=0	8b10bR=1	
Bit Index	NOCHALGN[A:B]=1 CV_SELxx=0	NOCHALGN[A:B]=1 CV_SELxx=1	NOCHALGN[A:B]=0 CV_SELxx=1
39	bit 9 of 10-bit data 3	CV_xx3, code violation, byte 3	See Table 9 and Table 10
38	bit 8 of 10-bit data 3	K_CTRL for byte 3	K_CTRL for byte 3
37	bit 7 of 10-bit data 3	bit 7 of byte3	bit 7 of byte3
36	bit 6 of 10-bit data 3	bit 6 of byte 3	bit 6 of byte 3
35	bit 5 of 10-bit data 3	bit 5 of byte 3	bit 5 of byte 3
34	bit 4 of 10-bit data 3	bit 4 of byte 3	bit 4 of byte 3
33	bit 3 of 10-bit data 3	bit 3 of byte 3	bit 3 of byte 3
32	bit 2 of 10-bit data 3	bit 2 of byte 3	bit 2 of byte 3
31	bit 1of 10-bit data 3	bit 1 of byte 3	bit 1 of byte 3
30	bit 0 of 10-bit data 3	bit 0 of byte 3	bit 0 of byte 3
29	bit 9 of 10-bit data 2	CV_xx2, code violation, byte 2	See Table 9 and Table 10
28	bit 8 of 10-bit data 2	K_CTRL for byte 2	K_CTRL for byte 2
27	bit 7 of 10-bit data 2	bit 7 of byte 2	bit 7 of byte 2
26	bit 6 of 10-bit data 2	bit 6 of byte 2	bit 6 of byte 2
25	bit 5 of 10-bit data 2	bit 5 of byte 2	bit 5 of byte 2
24	bit 4 of 10-bit data 2	bit 4 of byte 2	bit 4 of byte 2
23	bit 3 of 10-bit data 2	bit 3 of byte 2	bit 3 of byte 2
22	bit 2 of 10-bit data 2	bit 2 of byte 2	bit 2 of byte 2
21	bit 1 of 10-bit data 2	bit 1 of byte 2	bit 1 of byte 2
20	bit 0 of 10-bit data 2	bit 0 of byte 2	bit 0 of byte 2
19	bit 9 of 10-bit data 1	CV_xx1, code violation, byte 1	See Table 9 and Table 10
18	bit 8 of 10-bit data 1	K_CTRL for byte 1	K_CTRL for byte 1
17	bit 7 of 10-bit data 1	bit 7 of byte 1	bit 7 of byte 1
16	bit 6 of 10-bit data 1	bit 6 of byte 1	bit 6 of byte 1
15	bit 5 of 10-bit data 1	bit 5 of byte 1	bit 5 of byte 1
14	bit 4 of 10-bit data 1	bit 4 of byte 1	bit 4 of byte 1
13	bit 3 of 10-bit data 1	bit 3 of byte 1	bit 3 of byte 1

RSYS_CLK_[A:B][1:2]
These clocks are inputs to the SERDES quad block A and B respectively from the FPGA. These are used by each channel as the read clock to read received data from the alignment FIFO within the embedded core. Clocks RSYS_CLK_A[1:2] are used by channels in the SERDES quad block A and RSYS_CLK_B[1:2] by channels in the SERDES quad block B. To guarantee that there is no overflow in the alignment FIFO, it is an absolute requirement that the write and read clocks be frequency locked within 0 ppm. Examples of how to achieve this are shown in the later section on recommended board-level clocking.

TCK78[A:B]:
This is a muxed output from the core to the FPGA across the core-FPGA interface of one of the 4 transmit SERDES clocks per quad operating at up to 92.5 MHz in the embedded core. There is one clock output per SERDES quad block.

TSYS_CLK[AA,…BD]:
These clocks are inputs to the SERDES quad block A and B respectively from the FPGA. These are used by each channel to control the timing of the Transmit Data Path. To guarantee correct transmit operation theses clocks must be frequency locked within 0 ppm to TCK78[A:B].

Transmit and Receive Clock Rates

Table 16 shows the typical relationship between the data rates, the reference clock, the transmit TCK78[A:B] clock and the receive RCK78[A:B] clock. The selection of full-rate or half-rate for a given reference clock speed is set by bits in the transmit and receive control registers and can be set per channel.

Table 16. Transmit Data and Clock Rates

Data Rate	Reference Clock	TCK78[A: B] and RCK78[A:B] Clocks	Rate of Channel Selected as Clock Source
0.6 Gbps	60 MHz	15 MHz	Half
1.0 Gbps	100 MHz	25 MHz	Half
1.25 Gbps	125 MHz	31.25 MHz	Half
2.0 Gbps	100 MHz	50 MHz	Full
2.5 Gbps	125 MHz	62.5 MHz	Full
3.125 Gbps	156 MHz	78 MHz	Full
3.7 Gbps	185 MHz	92.5 MHz	Full

Besides taking in a TSYS_CLK_xx from the FPGA logic for each channel, the transmit path logic sends back a clock of the same frequency, but arbitrary phase. This clock, TCK78[A:B], is derived from the MUX block of one of the 4 channels in its SERDES quad. The MUX blocks provide the potential source for TCK78[A:B] by a divide-by-4 of the SERDES STBC311xs clock used in synchronizing the transmit data words in the STBC311xx clock domain. The STBC311xx clocks are internal to the core and are not brought across the core/FPGA interface.

The receiver section receives high-speed serial data at its differential CML input port and sends in to the Clock and Data Recovery (CDR) block. The CDR block then generates a recovered clock (RWCKxx) and retimes the data. Thus, the recovered receive clocks are asynchronous between channels.

Transmit Clock Source Selection

The TCKSEL[0:1][A:B] bits select the source channel of TCK78[A:B]. The selection of the source for TCK78[A:B] is controlled by these bits as shown in Table 17.

Test Modes

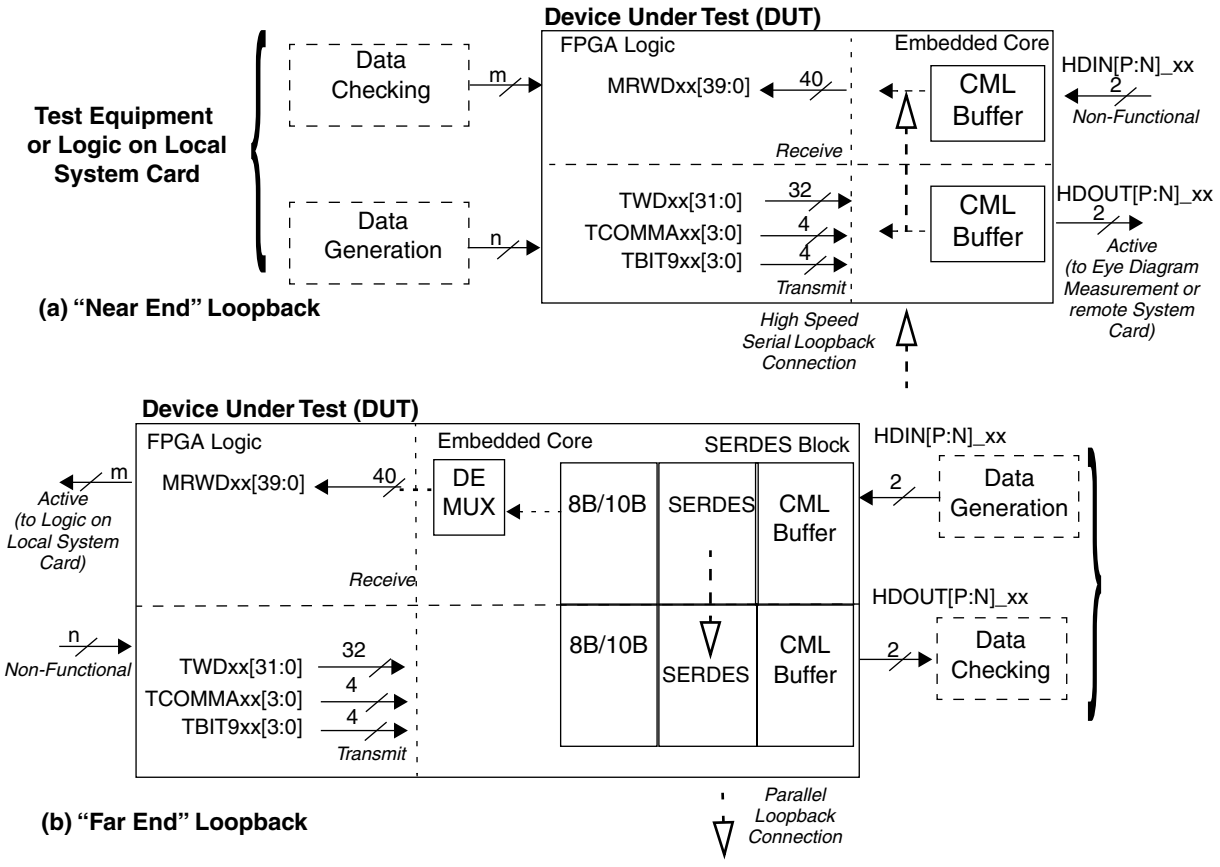
In addition to the operational logic described in the preceding sections, the Embedded Core contains logic to support various test modes - both for device validation and evaluation and for operating system level tests. The following sections discuss two of the test support logic blocks, supporting various loopback modes and SERDES characterization.

Loopback Testing

Loopback testing is performed by looping back (either internal to the Embedded Core, by configuring the FPGA logic or by external connections) transmitted data to the corresponding receiver inputs, or received data to the transmitter output. The loopback path may be either serial or parallel.

In general, loopback tests can be classified as “near end” or “far end.” In “near end” loopback (Figure 32(a)), data is generated and checked locally, i.e. by logic on, or connection of, test equipment to the same card as the FPSC. In “far end” loopback (Figure 32(b)), the generating and checking functions are performed remotely, either by test equipment or a remote system card.

Figure 32. “Near End” vs. “Far End” Loopback



The loopback mode can also be characterized by the physical location of the loopback connection. There are three possible loopback modes supported by the Embedded Core logic:

- High-speed serial loopback at the CML buffer interface (near end)
- Parallel loopback at the SERDES boundary (far end)

As mentioned earlier, both sections of a slice can be written independently / simultaneously, due to the independent CSW per section.

The same signal illustration above applies to slice B by changing _A to _B.

SDRAM A and SDRAM B in Figure 34 refer to the built-in sections A and B of one EAC RAM slice.

These SDRAMS should not be confused with the FPGA SDRAMS, which are generated through Module Generator in ispLEVER. The EAC SDRAMs are always built-in to the embedded core section of the ORT82G5/42G5 and their pins are accessed through the EAC interface. In order for these pins to be available at the interface in the generated HDL models from ispLEVER, the “Use the Extra Memory in FPSC Core” checkbox needs to be checked in the customization window (after hitting the "customize" button) in Module Generator, while generating the ORT82G5/42G5 core HDL. These signals will not otherwise show in the interface model.

Figure 35 and Figure 36 show, per slice, timing diagrams for both write and read accesses. These figures do not include the _x section, which refers to either slice A or B, even though this is implied. Signal names and functions are summarized in Table 26 and follow the general ORCA Series 4 naming conventions.

Figure 34. Block Diagram, Embedded Core Memory Slice

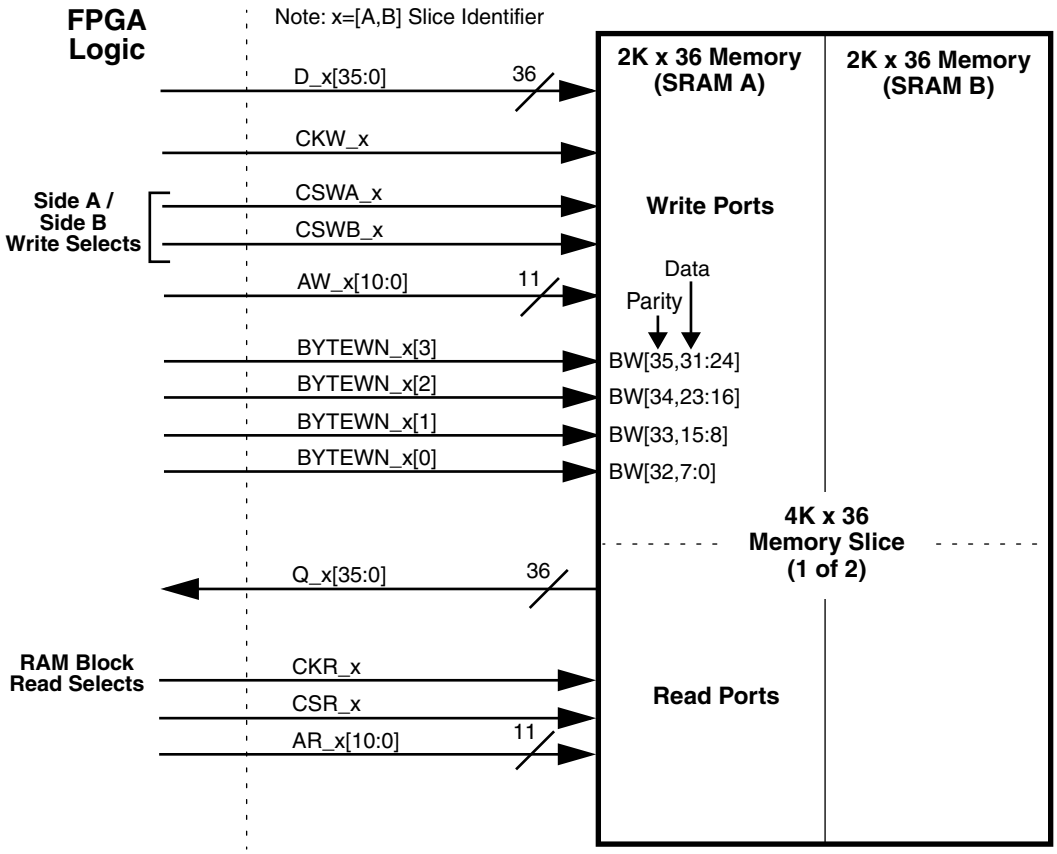


Figure 35. Minimum Timing Specs for Memory Blocks-Write Cycle (-1 Speed Grade)

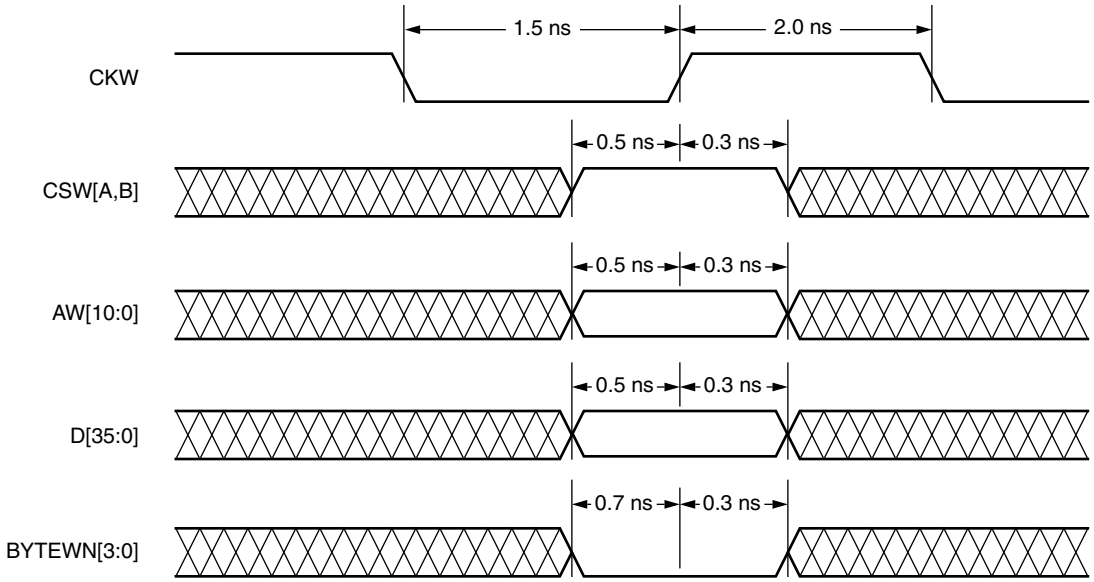
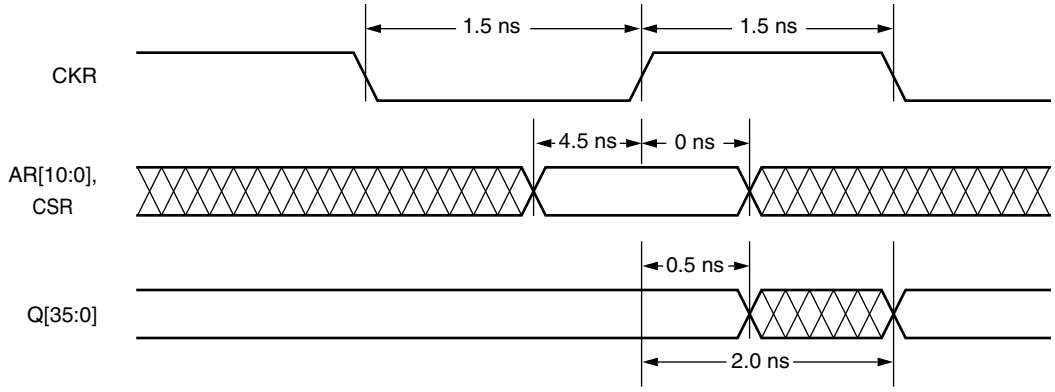


Figure 36. Minimum Timing Specs for Memory Blocks-Read Cycle (-1 Speed Grade)



In Table 26, an input refers to a signal flowing into the embedded core and an output refers to a signal flowing out of the embedded core.

Table 28. ORT42G5 Memory Map (Continued)

(0x) Absolute Address	Bit	Name	Reset Value (0x)	Description
Control Registers (Read/Write), xx=[AC, AD, BC or BD]				
30800 - Ax 30900 - Bx	[0]	—	00	Reserved for future use
	[1]	—		Reserved for future use
	[2]	ENBYSYNC_xC		ENBYSYNC_xC= 1 Enables Receiver Byte Synchronization for Channel xC. ENBYSYNC_xC = 0 on device reset.
	[3]	ENBYSYNC_xD		ENBYSYNC_xD = 1 Enables Receiver Byte Synchronization for Channel xA. ENBYSYNC_xD = 0 on device reset.
	[4]	—		Reserved for future use
	[5]	—		Reserved for future use
	[6]	LCKREFN_xC		LCKREFN_xC = 0 Locks the receiver PLL to reference clock for Channel xC. LCKREFN_xC =1 = Locks the receiver to data for Channel xx. NOTE: When LCKREFN_xx = 0, the corresponding LKI_xx bit is also 0. LCKREFN_xC = 0 on device reset.
	[7]	LCKREFN_xD		LCKREFN_xD = 0 Locks the receiver PLL to reference clock for Channel xD. LCKREFN_xD =1 = Locks the receiver to data for Channel xA. NOTE: When LCKREFN_xx = 0, the corresponding LKI_xx bit is also 0. LCKREFN_xD = 0 on device reset.
30801 - Ax 30901 - Bx	[0]	—	00	Reserved for future use
	[1]	—		Reserved for future use
	[2]	LOOPENB_xC		Enable Loopback Mode for Channel xC. When LOOPEN_xC=1, the transmitter high-speed output is looped back to the receiver high-speed input. This mode is similar to high-speed loopback mode enabled by TESTMODE_xx except that LOOPEN_xx disables the high-speed serial output. LOOPEN_xC=0 on device reset.
	[3]	LOOPENB_xD		Enable Loopback Mode for Channel xD. When LOOPEN_xD=1, the transmitter high-speed output is looped back to the receiver high-speed input. This mode is similar to high-speed loopback mode enabled by TESTMODE_xx except that LOOPEN_xx disables the high-speed serial output. LOOPEN_xD=0 on device reset.
	[4]	—		Reserved for future use
	[5]	—		Reserved for future use
	[6]	NOWDALIGN_xC		Word Align Disable Bit. When NOWDALIGN_xC=1, receiver word alignment is disabled for Channel xC. NOWDALIGN_xC=0 on device reset.
	[7]	NOWDALIGN_xD		Word Align Disable Bit. When NOWDALIGN_xD=1, receiver word alignment is disabled for Channel xD. NOWDALIGN_xD=0 on device reset.

Table 30. ORT82G5 Memory Map (Continued)

(0x) Absolute Address	Bit	Name	Reset Value (0x)	Description
30A02	[0:1]	RX_FIFO_MIN	00	MSb's for the threshold for low address in RX_FIFOs. RX_FIFO_MIN, Bit 1 is MSb. Useful values for RX_FIFO_MIN [0:4] are 0 to 17(decimal).
	[2]	FMPU_RESYNC8		Resynchronizes all 8 channels when it transitions from 0 to 1. Status is a 0 on device reset.
	[3:7]	—		Reserved for future use.
Common Status Registers xx=[AA,...,BD]				
30A03	[0]	SYNC8_OVFL	00	Read-Only Multi-Channel Overflow Status. When SYNC8_OVFL=1, 8-channel synchronization FIFO overflow has occurred. SYNC8_OVFL=0 on device reset.
	[1]	SYNC8_OOS		Read-Only Multi-Channel Out-Of-Sync Status. When SYNC8_OOS=1, 8-channel synchronization has failed. SYNC8_OOS=0 on device reset.
	[2:7]	Reserved for future use.		

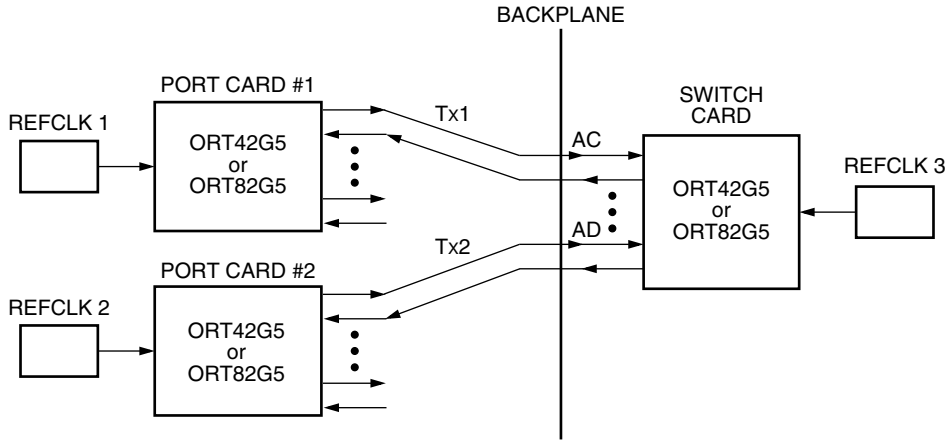
Recommended Board-level Clocking for the ORT42G5 and ORT82G5

Option 1: Asynchronous Reference Clocks Between Rx and Tx Devices

Each board that uses the ORT42G5 or ORT82G5 as a transmit or receive device will have its own local reference clock as shown in Figure 37. Figure 37 shows the ORT82G5 device on the switch card receiving data on two of its channels from a separate source. Data tx1 is transmitted from a tx device with refclk1 as the reference clock and Data tx2 is transmitted from a tx device with refclk2 as the reference clock. Receive channel AA locks to the incoming data tx1 and receive channel AB locks to the incoming data tx2.

The advantage of this clocking scheme is the fact that it is not necessary to distribute a reference clock (typically 156 MHz for 10GE and 155.52 MHz for OC-192 applications) across a backplane.

Figure 37. Asynchronous Clocking Between Rx and Tx Devices



Option 2: Synchronous Reference Clocks to Rx and Tx Devices

In this type of clocking, a single reference clock is distributed to all receive and transmit devices in a system (Figure 38). This distributed clocking scheme will permit maximum flexibility in the usage of transmit and receive channels in the current silicon such as:

- All transmit and receive channels can be used within any quad in receive channel alignment or alignment bypass mode.
- In channel alignment mode, each receive channel operates on its own independent clock domain.

Table 44. ORT42G5 484-pin PBGAM (fpBGA) Pinout (Continued)

484-PBGAM	VDDIO Bank	VREF Group	I/O	Pin Description	Additional Function	484-PBGAM
V9	5 (BC)	1	IO	PB17A	-	-
W9	5 (BC)	1	IO	PB17C	-	L35T
Y9	5 (BC)	1	IO	PB17D	-	L35C
U9	5 (BC)	1	IO	PB18A	-	-
AA9	5 (BC)	1	IO	PB18C	VREF_5_01	L36T
AB9	5 (BC)	1	IO	PB18D	-	L36C
G16	-	-	VDD15	VDD15	-	-
H13	-	-	VSS	VSS	-	-
AB10	5 (BC)	2	IO	PB19A	-	L37T
AA10	5 (BC)	2	IO	PB19B	-	L37C
W10	5 (BC)	2	IO	PB19C	PBCK0T	L38T
Y10	5 (BC)	2	IO	PB19D	PBCK0C	L38C
V10	5 (BC)	2	IO	PB20A	-	-
U13	5 (BC)	-	VDDIO5	VDDIO5	-	-
AB11	5 (BC)	2	IO	PB20C	VREF_5_02	L39T
AA11	5 (BC)	2	IO	PB20D	-	L39C
U10	5 (BC)	2	IO	PB21A	-	-
H6	-	-	VDD15	VDD15	-	-
Y11	5 (BC)	3	IO	PB21C	-	L40T
W11	5 (BC)	3	IO	PB21D	VREF_5_03	L40C
U11	5 (BC)	3	IO	PB22A	-	-
J7	-	-	VSS	VSS	-	-
AB12	5 (BC)	3	IO	PB22C	-	L41T
AA12	5 (BC)	3	IO	PB22D	-	L41C
U12	5 (BC)	3	IO	PB23A	-	-
Y12	5 (BC)	3	IO	PB23C	PBCK1T	L42T
W12	5 (BC)	3	IO	PB23D	PBCK1C	L42C
V11	5 (BC)	3	IO	PB24A	-	-
J8	-	-	VSS	VSS	-	-
AB13	5 (BC)	4	IO	PB24C	-	L43T
AA13	5 (BC)	4	IO	PB24D	-	L43C
V12	5 (BC)	4	IO	PB25A	-	-
U14	5 (BC)	-	VDDIO5	VDDIO5	-	-
AB14	5 (BC)	4	IO	PB25C	-	L44T
AA14	5 (BC)	4	IO	PB25D	VREF_5_04	L44C
J9	-	-	VSS	VSS	-	-
Y13	5 (BC)	5	IO	PB26C	-	L45T
W13	5 (BC)	5	IO	PB26D	VREF_5_05	L45C
U15	5 (BC)	-	VDDIO5	VDDIO5	-	-
AB15	5 (BC)	5	IO	PB27C	-	L46T
AA15	5 (BC)	5	IO	PB27D	-	L46C
AB16	5 (BC)	6	IO	PB28C	-	L47T
AA16	5 (BC)	6	IO	PB28D	VREF_5_06	L47C
H14	-	-	VDD15	VDD15	-	-

Table 44. ORT42G5 484-pin PBGAM (fpBGA) Pinout (Continued)

484-PBGAM	VDDIO Bank	VREF Group	I/O	Pin Description	Additional Function	484-PBGAM
Y14	5 (BC)	6	IO	PB29C	-	L48T
W14	5 (BC)	6	IO	PB29D	-	L48C
J10	-	-	VSS	VSS	-	-
AB17	5 (BC)	7	IO	PB30C	-	L49T
AA17	5 (BC)	7	IO	PB30D	-	L49C
U16	5 (BC)	-	VDDIO5	VDDIO5	-	-
Y15	5 (BC)	7	IO	PB31C	VREF_5_07	L50T
W15	5 (BC)	7	IO	PB31D	-	L50C
V13	5 (BC)	-	VDDIO5	VDDIO5	-	-
AB18	5 (BC)	8	IO	PB33C	-	L51T
AA18	5 (BC)	8	IO	PB33D	VREF_5_08	L51C
J11	-	-	VSS	VSS	-	-
V14	5 (BC)	8	IO	PB34D	-	-
V16	5 (BC)	9	IO	PB35B	-	-
Y16	5 (BC)	9	IO	PB36C	-	L52T
W16	5 (BC)	9	IO	PB36D	-	L52C
V15	-	-	VDD33	VDD33	-	-
J12	-	-	VSS	VSS	-	-
H15	-	-	VDD15	VDD15	-	-
J13	-	-	VSS	VSS	-	-
J6	-	-	VDD15	VDD15	-	-
J14	-	-	VSS	VSS	-	-
Y17	-	-	VDD33	VDD33	-	-
K8	-	-	VSS	VSS	-	-
J15	-	-	VDD15	VDD15	-	-
K7	-	-	VDD15	VDD15	-	-
Y18	-	-	VDD33	VDD33	-	-
K9	-	-	VSS	VSS	-	-
W21	-	-	VSS	VSS	-	-
W22	-	-	VDDGB_B	VDDGB_B	-	-
F18	-	-	VDD_ANA	VDD_ANA	-	-
V21	-	-	O	REXT_B	-	-
V22	-	-	O	REXTN_B	-	-
U21	-	-	I	REFCLKN_B	-	HSN_1
U22	-	-	I	REFCLKP_B	-	HSP_1
E20	-	-	VSS	VSS	-	-
G17	-	-	VDD_ANA	VDD_ANA	-	-
G18	-	-	VDD_ANA	VDD_ANA	-	-
J16	-	-	VDD_ANA	VDD_ANA	-	-
J17	-	-	VDD_ANA	VDD_ANA	-	-
T20	-	-	VDDIB	VDDIB_BC	-	-
J18	-	-	VDD_ANA	VDD_ANA	-	-
T21	-	-	I	HDINN_BC	-	HSN_2
F19	-	-	VSS	VSS	-	-

Table 45. ORT82G5 680-Pin PBGAM (fpBGA) Pinout (Continued)

680-PBGAM	VDDIO Bank	VREF Group	I/O	Pin Description	Additional Function	680-PBGAM
AM4	6 (BL)	5	IO	PB2A	DP2	L13T_D0
AL5	6 (BL)	5	IO	PB2B	—	L13C_D0
AN7	6 (BL)	—	VDDIO6	VDDIO6	—	—
AP3	6 (BL)	5	IO	PB2C	PLL_CK6T/PPLL	L14T_A0
AP4	6 (BL)	5	IO	PB2D	PLL_CK6C/PPLL	L14C_A0
AN4	6 (BL)	5	IO	PB3B	—	—
U16	—	—	VSS	VSS	—	—
AK6	6 (BL)	5	IO	PB3C	—	L15T_A0
AK7	6 (BL)	5	IO	PB3D	—	L15C_A0
AL6	6 (BL)	5	IO	PB4A	VREF_6_05	L16T_A0
AM6	6 (BL)	5	IO	PB4B	DP3	L16C_A0
AP1	6 (BL)	—	VDDIO6	VDDIO6	—	—
AN5	6 (BL)	6	IO	PB4C	—	L17T_A0
AP5	6 (BL)	6	IO	PB4D	—	L17C_A0
AK8	6 (BL)	6	IO	PB5B	—	—
U17	—	—	VSS	VSS	—	—
AP6	6 (BL)	6	IO	PB5C	VREF_6_06	L18T_D0
AP7	6 (BL)	6	IO	PB5D	D14	L18C_D0
AM7	6 (BL)	6	IO	PB6A	—	L19T_D0
AN6	6 (BL)	6	IO	PB6B	—	L19C_D0
AP2	6 (BL)	—	VDDIO6	VDDIO6	—	—
AL8	6 (BL)	7	IO	PB6C	D15	L20T_A0
AL9	6 (BL)	7	IO	PB6D	D16	L20C_A0
AK9	6 (BL)	7	IO	PB7B	—	—
U18	—	—	VSS	VSS	—	—
AN8	6 (BL)	7	IO	PB7C	D17	L21T_A0
AM8	6 (BL)	7	IO	PB7D	D18	L21C_A0
AN9	6 (BL)	7	IO	PB8A	—	L22T_D0
AP8	6 (BL)	7	IO	PB8B	—	L22C_D0
AK10	6 (BL)	7	IO	PB8C	VREF_6_07	L23T_A0
AL10	6 (BL)	7	IO	PB8D	D19	L23C_A0
AP9	6 (BL)	8	IO	PB9B	—	—
U19	—	—	VSS	VSS	—	—
AM10	6 (BL)	8	IO	PB9C	D20	L24T_A0
AM11	6 (BL)	8	IO	PB9D	D21	L24C_A0
AK11	6 (BL)	8	IO	PB10B	—	—
AN10	6 (BL)	8	IO	PB10C	VREF_6_08	L25T_A0
AP10	6 (BL)	8	IO	PB10D	D22	L25C_A0
AN11	6 (BL)	9	IO	PB11A	—	L26T_A0
AP11	6 (BL)	9	IO	PB11B	—	L26C_A0
V16	—	—	VSS	VSS	—	—
AL12	6 (BL)	9	IO	PB11C	D23	L27T_A0
AK12	6 (BL)	9	IO	PB11D	D24	L27C_A0
AN12	6 (BL)	9	IO	PB12A	—	L28T_A0

Table 45. ORT82G5 680-Pin PBGAM (fpBGA) Pinout (Continued)

680-PBGAM	VDDIO Bank	VREF Group	I/O	Pin Description	Additional Function	680-PBGAM
C4	0 (TL)	6	IO	PT2D	PLL_CK1C/PPLL	L19C_A0
B4	0 (TL)	6	IO	PT2C	PLL_CK1T/PPLL	L19T_A0
A4	0 (TL)	6	IO	PT2B	—	L20C_A0
A3	0 (TL)	6	IO	PT2A	—	L20T_A0
D5	—	—	O	PCFG_MPI_IRQ	CFG_IRQ_N/MPI_IRQ_N	—
E6	—	—	IO	PCCLK	CCLK	—
D4	—	—	IO	PDONE	DONE	—
E5	—	—	VDD33	VDD33	—	—
AB15	—	—	Vss	Vss	—	—
AL33	—	—	VDD15	VDD15	—	—
AL34	—	—	VDD15	VDD15	—	—
AM34	—	—	VDD15	VDD15	—	—
AN34	—	—	VDD15	VDD15	—	—
B34	—	—	VDD15	VDD15	—	—
C33	—	—	VDD15	VDD15	—	—
C34	—	—	VDD15	VDD15	—	—
D33	—	—	VDD15	VDD15	—	—
D34	—	—	VDD15	VDD15	—	—
E32	—	—	VDD15	VDD15	—	—
E33	—	—	VDD15	VDD15	—	—
F32	—	—	VDD15	VDD15	—	—
F34	—	—	VDD15	VDD15	—	—
N16	—	—	VDD15	VDD15	—	—
N17	—	—	VDD15	VDD15	—	—
N18	—	—	VDD15	VDD15	—	—
N19	—	—	VDD15	VDD15	—	—
P16	—	—	VDD15	VDD15	—	—
P17	—	—	VDD15	VDD15	—	—
P18	—	—	VDD15	VDD15	—	—
P19	—	—	VDD15	VDD15	—	—
R16	—	—	VDD15	VDD15	—	—
R17	—	—	VDD15	VDD15	—	—
R18	—	—	VDD15	VDD15	—	—
R19	—	—	VDD15	VDD15	—	—
T13	—	—	VDD15	VDD15	—	—
T14	—	—	VDD15	VDD15	—	—
T15	—	—	VDD15	VDD15	—	—
T20	—	—	VDD15	VDD15	—	—
T21	—	—	VDD15	VDD15	—	—
T22	—	—	VDD15	VDD15	—	—
U13	—	—	VDD15	VDD15	—	—
U14	—	—	VDD15	VDD15	—	—
U15	—	—	VDD15	VDD15	—	—
U20	—	—	VDD15	VDD15	—	—