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	1568
Total RAM Bits	25600
Number of I/O	171
Number of Gates	48000
Voltage - Supply	3V ~ 3.6V
Mounting Type	Surface Mount
Operating Temperature	-40°C ~ 85°C (TA)
Package / Case	208-BFQFP
Supplier Device Package	208-PQFP (28x28)
Purchase URL	https://www.e-xfl.com/product-detail/lattice-semiconductor/or3t306s208i-db

Table of Contents

System-Level Features

System-level features reduce glue logic requirements and make a system on a chip possible. These features in the *ORCA* Series 3 include:

- Full PCI local bus compliance.
- Dual-use microprocessor interface (MPI) can be used for configuration, readback, device control, and device status, as well as for a general-purpose interface to the FPGA. Glueless interface to *i960** and *PowerPC*† processors with user-configurable address space provided.
- Parallel readback of configuration data capability with the built-in microprocessor interface.
- Programmable clock manager (PCM) adjusts clock phase and duty cycle for input clock rates from 5 MHz to 120 MHz. The PCM may be combined with FPGA logic to create complex functions, such as digital phase-locked loops (DPLL), frequency counters, and frequency synthesizers or clock doublers. Two PCMs are provided per device.
- True, internal, 3-state, bidirectional buses with simple control provided by the SLIC.
- 32 x 4 RAM per PFU, configurable as single- or dual-port at >176 MHz. Create large, fast RAM/ROM blocks (128 x 8 in only eight PFUs) using the SLIC decoders as bank drivers.

* *i960* is a registered trademark of Intel Corporation.
† *PowerPC* is a registered trademark of International Business Machines Corporation.

Table 2. *ORCA* Series 3 System Performance

Parameter	# PFUs	Speed				Unit
		-4	-5	-6	-7	
16-bit Loadable Up/Down Counter	2	78	102	131	168	MHz
16-bit Accumulator	2	78	102	131	168	MHz
8 x 8 Parallel Multiplier:						
Multiplier Mode, Unpipelined[1]	11.5	19	25	30	38	MHz
ROM Mode, Unpipelined[2]	8	51	66	80	102	MHz
Multiplier Mode, Pipelined[3]	15	76	104	127	166	MHz
32 x 16 RAM (synchronous):						
Single-port, 3-state Bus[4]	4	97	127	151	192	MHz
Dual-port[5]	4	127	166	203	253	MHz
128 x 8 RAM (synchronous):						
Single-port, 3-state Bus[4]	8	88	116	139	176	MHz
Dual-port[5]	8	88	116	139	176	MHz
8-bit Address Decode (internal):						
Using Softwired LUTs	0.25	4.87	3.66	2.58	2.03	ns
Using SLICs[6]	0	2.35	1.82	1.23	0.99	ns
32-bit Address Decode (internal):						
Using Softwired LUTs	2	16.06	12.07	9.01	7.03	ns
Using SLICs[7]	0	6.91	5.41	4.21	3.37	ns
36-bit Parity Check (internal)	2	16.06	12.07	9.01	7.03	ns

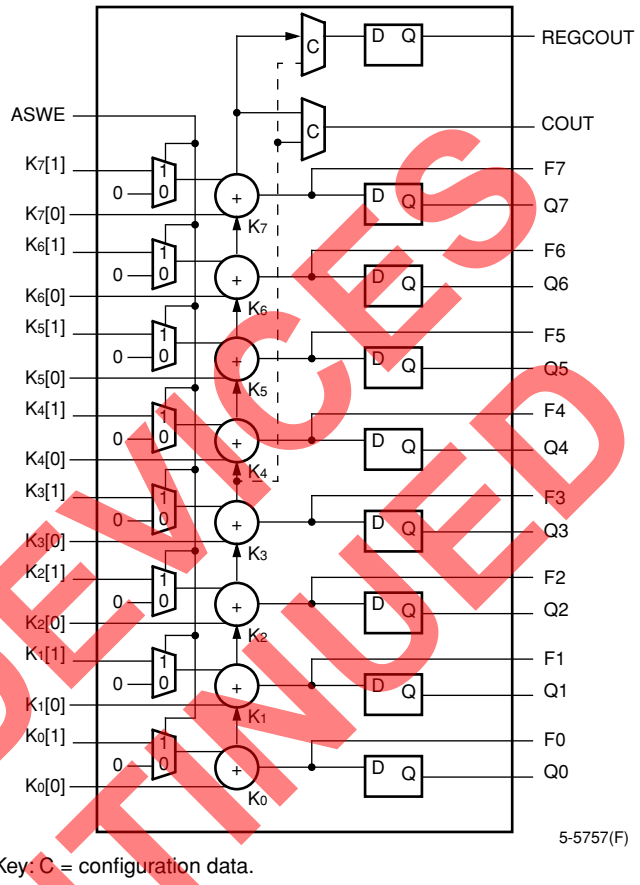
1. Implemented using 8 x 1 multiplier mode (unpipelined), register-to-register, two 8-bit inputs, one 16-bit output.
2. Implemented using two 32 x 12 ROMs and one 12-bit adder, one 8-bit input, one fixed operand, one 16-bit output.
3. Implemented using 8 x 1 multiplier mode (fully pipelined), two 8-bit inputs, one 16-bit output (7 of 15 PFUs contain only pipelining registers).
4. Implemented using 32 x 4 RAM mode with read data on 3-state buffer to bidirectional read/write bus.
5. Implemented using 32 x 4 dual-port RAM mode.
6. Implemented in one partially occupied SLIC with decoded output set up to CE in same PLC.
7. Implemented in five partially occupied SLICs.

Programmable Logic Cells (continued)

In the third submode, **multiplier submode**, a single PFU can affect an 8 x 1 bit (4 x 1 for half-ripple mode) multiply and sum with a partial product (see Figure 8). The multiplier bit is input at ASWE, and the multiplicand bits are input at Kz[1], where K7[1] is the most significant bit (MSB). Kz[0] contains the partial product (or other input to be summed) from a previous stage. If ASWE is logical 1, the multiplicand is added to the partial product. If ASWE is logical 0, 0 is added to the partial product, which is the same as passing the partial product. CIN/FCIN can bring the carry-in from the less significant PFUs if the multiplicand is wider than 8 bits, and COUT/FCOUT holds any carry-out from the multiplication, which may then be used as part of the product or routed to another PFU in multiplier mode for multiplicand width expansion.

Ripple mode's fourth submode features **equality comparators.** The functions that are explicitly available are A > B, A ≠ B, and A < B, where the value for A is input on Kz[0], and the value for B is input on Kz[1]. A value of 1 on the carry-out signals valid argument. For example, a carry-out equal to 1 in AB submode indicates that the value on Kz[0] is greater than or equal to the value on Kz[1]. Conversely, the functions A < B, A + B, and A > B are available using the same functions but with a 0 output expected. For example, A > B with a 0 output indicates A < B. Table 5 shows each function and the output expected.

If larger than 8 bits, the carry-out signal can be cascaded using fast-carry logic to the carry-in of any adjacent PFU. The use of this submode could be shown using Figure 6, except that the CIN/FCIN input for the least significant PFU is controlled via configuration.



Key: C = configuration data.

Figure 8. Multiplier Submode

Table 5. Ripple Mode Equality Comparator Functions and Outputs

Equality Function	ispLEVER Submode	True, if Carry-Out Is:
A > B	A > B	1
A < B	A < B	1
A ≠ B	A ≠ B	1
A < B	A > B	0
A > B	A < B	0
A = B	A ≠ B	0

Programmable Logic Cells (continued)

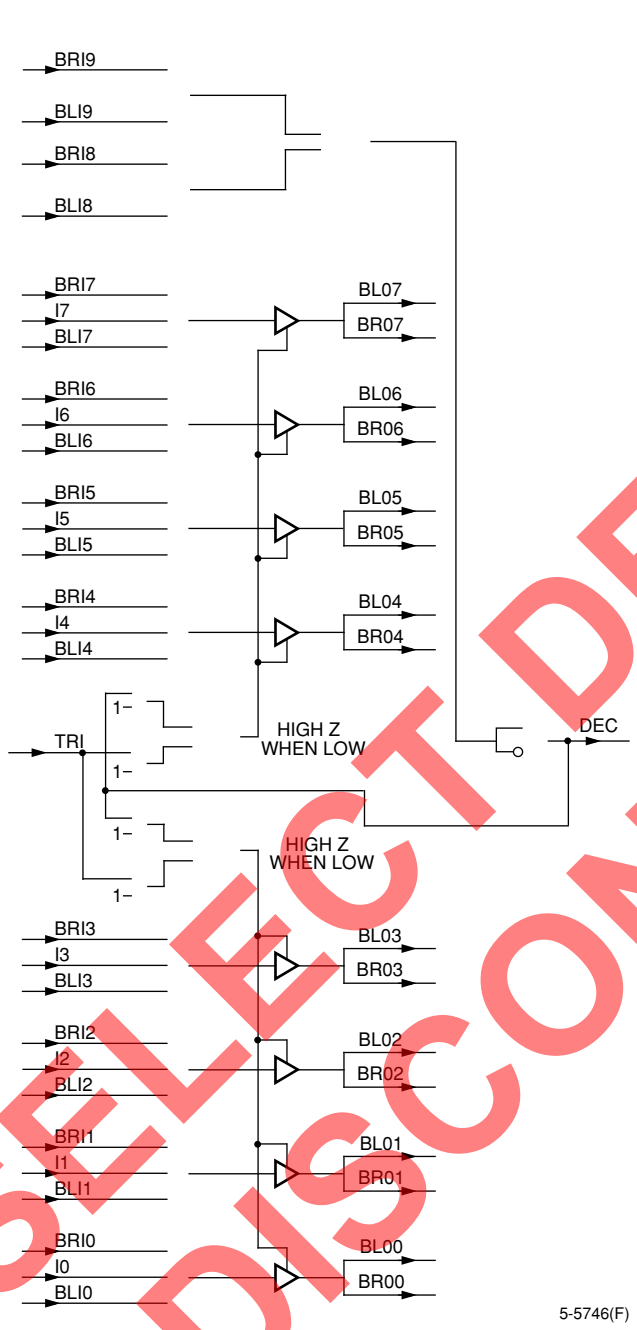


Figure 13. Buffer-Buffer-Decoder Mode

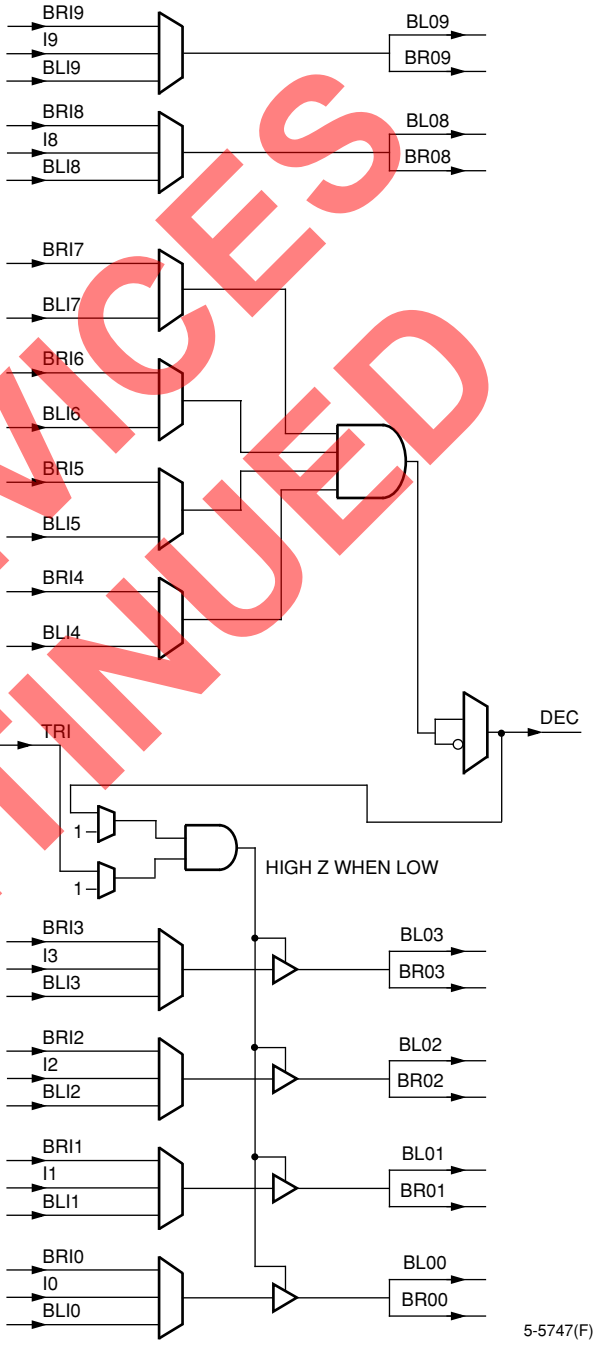


Figure 14. Buffer-Decoder-Buffer Mode

Programmable Logic Cells (continued)

PLC Routing Resources

Generally, the ispLEVER Development System is used to automatically route interconnections. Interactive routing with the ispLEVER design editor (EPIC) is also available for design optimization. To use EPIC for interactive layout, an understanding of the routing resources is needed and is provided in this section.

The routing resources consist of switching circuitry and metal interconnect segments. Generally, the metal lines which carry the signals are designated as routing segments. The switching circuitry connects the routing segments, providing one or more of three basic functions: signal switching, amplification, and isolation. A net running from a PFU or PIC output (source) to a PLC or PIC input (destination) consists of one or more routing segments, connected by switching circuitry called configurable interconnect points (CIPs).

The following sections discuss PLC, PIC, and interquad routing resources. This section discusses the PLC switching circuitry, intra-PLC routing, inter-PLC routing, and clock distribution.

Configurable Interconnect Points

The process of connecting routing segments uses three basic types of switching circuits: two types of configurable interconnect points (CIPs) and bidirectional buffers (BIDIs). The basic element in CIPs is one or more pass transistors, each controlled by a configuration RAM bit. The two types of CIPs are the mutually exclusive (or multiplexed) CIP and the independent CIP.

A mutually exclusive set of CIPs contains two or more CIPs, only one of which can be on at a time. An independent CIP has no such restrictions and can be on independent of the state of other CIPs. Figure 18 shows an example of both types of CIPs.

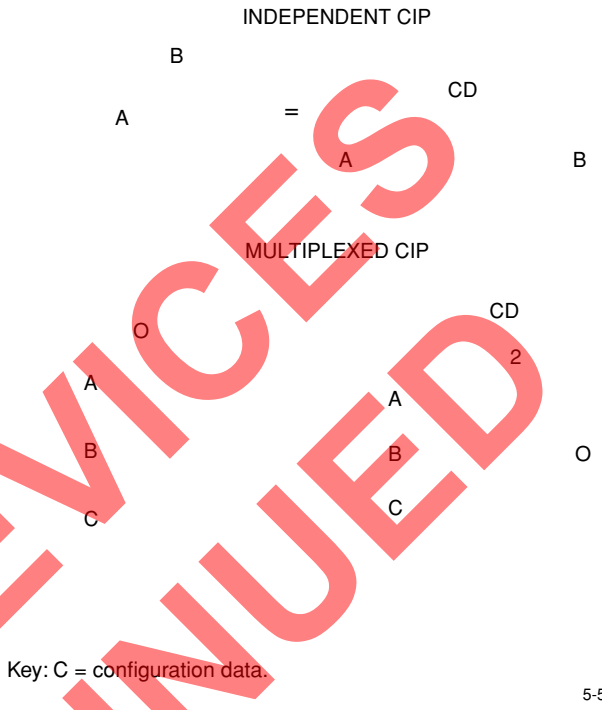


Figure 18. Configurable Interconnect Point

3-Statable Bidirectional Buffers

Bidirectional buffers, previously described in the SLIC section of the programmable logic cell discussion, provide isolation as well as amplification for signals routed a long distance. Bidirectional buffers are also used to route signals diagonally in the PLC (described later in the subsection entitled Intra-PLC Routing), and BIDIs can be used to indirectly route signals through the switching routing (xSW) segments. Any number from zero to ten BIDIs can be used in a given PLC.

Special Function Blocks (continued)

Start-Up Logic

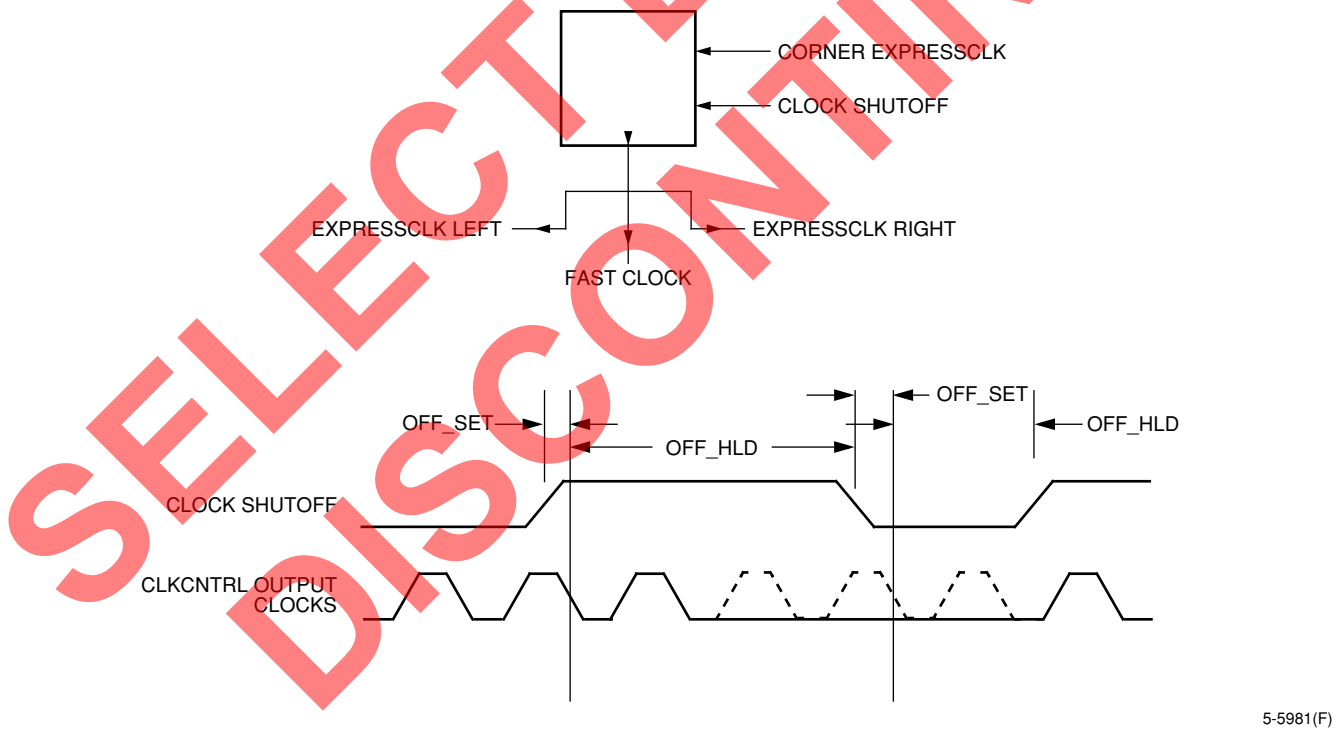
The start-up logic block is located in the lower right corner of the FPGA. This block can be configured to coordinate the relative timing of the release of GSRN, the activation of all user I/Os, and the assertion of the DONE signal at the end of configuration. If a start-up clock is used to time these events, the start-up clock can come from CCLK, or it can be routed into the start-up block using lower right corner routing resources. These signals are described in the Start-Up subsection of the FPGA States of Operation section.

Clock Control (CLKCNTRL) and StopCLK

There is one CLKCNTRL block in the MID section of the interquad routing on each side of the FPGA. This block is used to selectively distribute the fast clock to the PLC array and the left (top) and right (bottom) ExpressCLKs (ECKL and ECKR) to the side of the array on which the CLKCNTRL block resides.

The source clock for the CLKCNTRL block comes either from the ExpressCLK pad at the middle of the side of the FPGA or from the corner ExpressCLK route that comes from the corner ExpressCLK pad (at the lower left or upper right of the device, whichever is closer). The programmable clock manager ExpressCLK output can also be sourced to this corner routing for distribution at the two closest CLKCNTRL blocks.

Each CLKCNTRL block also features an invertible StopCLK shutoff input that is available from local routing. This feature may be used to glitchlessly stop and start the clock at the three outputs of each CLKCNTRL block and has the option of doing so on either the rising or falling edge of the clock. When the clock is halted based on its rising edge, it stops and stays at VDD. When it is stopped based on its falling edge, it stops and stays at GND. If the StopCLK shutoff signal meets the CLKCNTRL setup and hold times, the clock is stopped on the second clock cycle after the shutoff signal. A diagram of the bottom CLKCNTRL block and StopCLK timing is shown in Figure 35.



Notes:
CLKCNTRL output clocks are ExpressCLK left and right and fast clock.

Clock shutoff shown active-high acting on clock falling edge.

Figure 35. Top CLKCNTRL Function Block

Special Function Blocks (continued)

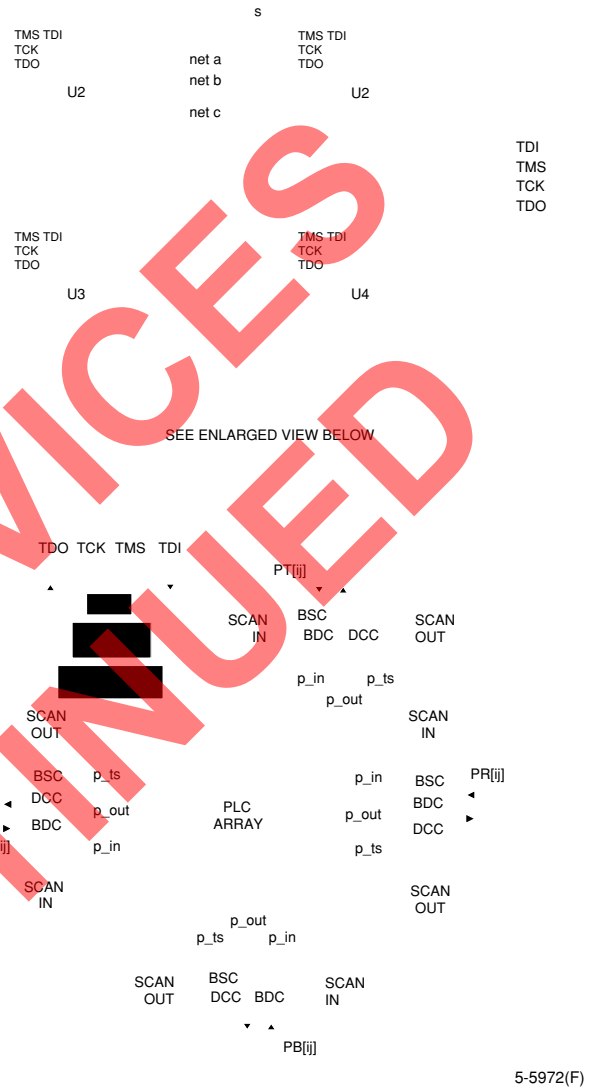
Boundary Scan

The increasing complexity of integrated circuits (ICs) and IC packages has increased the difficulty of testing printed-circuit boards (PCBs). To address this testing problem, the *IEEE* standard 1149.1/D1 (*IEEE* Standard Test Access Port and Boundary-Scan Architecture) is implemented in the *ORCA* series of FPGAs. It allows users to efficiently test the interconnection between integrated circuits on a PCB as well as test the integrated circuit itself. The *IEEE* 1149.1/D1 standard is a well-defined protocol that ensures interoperability among boundary-scan (BSCAN) equipped devices from different vendors.

The *IEEE* 1149.1/D1 standard defines a test access port (TAP) that consists of a four-pin interface with an optional reset pin for boundary-scan testing of integrated circuits in a system. The *ORCA* Series FPGA provides four interface pins: test data in (TDI), test mode select (TMS), test clock (TCK), and test data out (TDO). The $\overline{\text{PRGM}}$ pin used to reconfigure the device also resets the boundary-scan logic.

The user test host serially loads test commands and test data into the FPGA through these pins to drive outputs and examine inputs. In the configuration shown in Figure 36, where boundary scan is used to test ICs, test data is transmitted serially into TDI of the first BSCAN device (U1), through TDO/TDI connections between BSCAN devices (U2 and U3), and out TDO of the last BSCAN device (U4). In this configuration, the TMS and TCK signals are routed to all boundary-scan ICs in parallel so that all boundary-scan components operate in the same state. In other configurations, multiple scan paths are used instead of a single ring. When multiple scan paths are used, each ring is independently controlled by its own TMS and TCK signals.

Figure 37 provides a system interface for components used in the boundary-scan testing of PCBs. The three major components shown are the test host, boundary-scan support circuit, and the devices under test (DUTs). The DUTs shown here are *ORCA* Series FPGAs with dedicated boundary-scan circuitry. The test host is normally one of the following: automatic test equipment (ATE), a workstation, a PC, or a microprocessor.



5-5972(F)

Key: BSC = boundary-scan cell, BDC = bidirectional data cell, and DCC = data control cell.

Figure 36. Printed-Circuit Board with Boundary-Scan Circuitry

Special Function Blocks (continued)

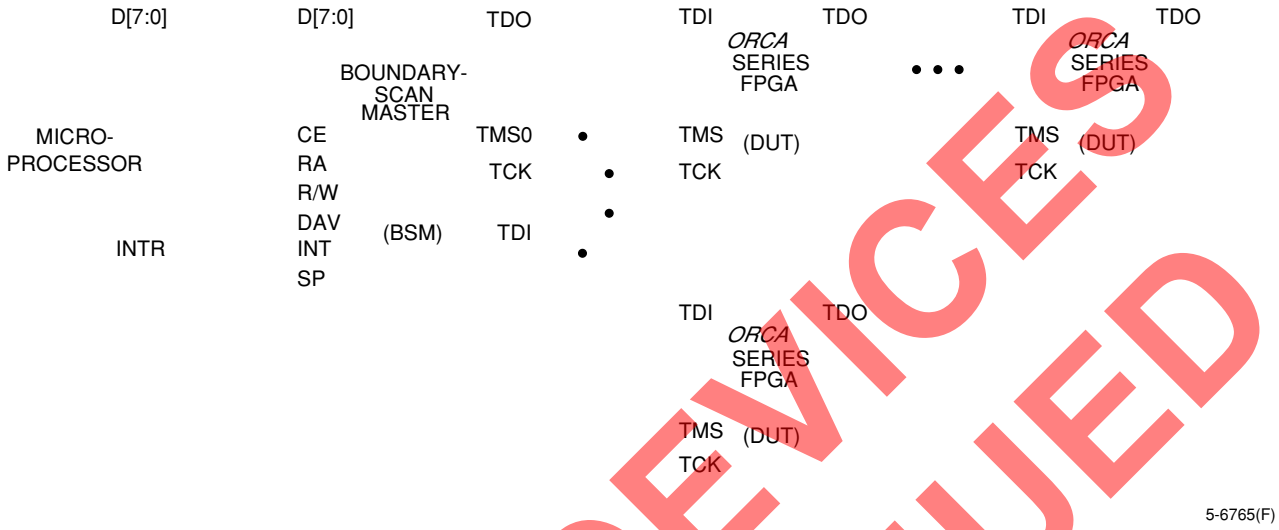


Figure 37. Boundary-Scan Interface

The boundary-scan support circuit shown in Figure 37 is the 497AA Boundary-Scan Master (BSM). The BSM off-loads tasks from the test host to increase test throughput. To interface between the test host and the DUTs, the BSM has a general microprocessor interface and provides parallel-to-serial/serial-to-parallel conversion, as well as three 8K data buffers. The BSM also increases test throughput with a dedicated automatic test-pattern generator and with compression of the test response with a signature analysis register. The PC-based boundary-scan test card/software allows a user to quickly prototype a boundary-scan test setup.

Boundary-Scan Instructions

The *ORCA* Series boundary-scan circuitry is used for three mandatory *IEEE* 1149.1/D1 tests (EXTEST, SAMPLE/PRELOAD, BYPASS), the optional *IEEE* 1149.1/D1 IDCODE instruction, and five *ORCA*-defined instructions. The 3-bit wide instruction register supports the nine instructions listed in Table 13, where the use of PSR1 or USERCODE is selectable by a bit stream option.

Table 13. Boundary-Scan Instructions

Code	Instruction
000	EXTEST
001	PLC Scan Ring 1 (PSR1)/USERCODE
010	RAM Write (RAM_W)
011	IDCODE
100	SAMPLE/PRELOAD
101	PLC Scan Ring 2 (PSR2)
110	RAM Read (RAM_R)
111	BYPASS

Microprocessor Interface (MPI) (continued)

MPI Setup and Control

The MPI has a series of addressable registers that provide MPI control and status, configuration and readback data transfer, FPGA device identification, and a dedicated user scratchpad register. All registers are 8 bits wide. The address map for these registers and the user-logic address space are shown in Table 19, followed by descriptions of the register and bit functions. Note that for all registers, the most significant bit is bit 7, and the least significant bit is bit 0.

Table 19. MPI Setup and Control Registers

Address (Hex)	Register
00	Control Register 1.
01	Control Register 2.
02	Scratchpad Register.
03	Status Register.
04	Configuration/Readback Data Register.
05	Readback Address Register 1 (bits [7:0]).
06	Readback Address Register 2 (bits [15:8]).
07	Device ID Register 1 (bits [7:0]).
08	Device ID Register 2 (bits [15:8]).
09	Device ID Register 3 (bits [23:16]).
0A	Device ID Register 4 (bits [31:24]).
0B—0F	Reserved.
10—1F	User-definable Address Space.

Control Register 1

The MPI control register 1 is a read/write register. The host processor writes a control byte to configure the MPI. It is readable by the host processor to verify the status of control bits previously written.

Table 20. MPI Setup and Control Registers Descriptions

Bit #	Description
Bit 0	**GSR Input.** Setting this bit to a 1 invokes a global set/reset on the FPGA. The host processor must return this bit to a 0 to remove the GSR signal. GSR does not affect the registers at MPI addresses 0 through F hexadecimal or any configuration registers. Default state = 0.
Bit 1	**Reserved.**
Bit 2	**Reserved.**
Bit 3	**Reserved.**
Bit 4	**Reserved.**
Bit 5	**$\overline{\text{RD\_CFG}}$ Input.** Changing this bit to a 0 after configuration will initiate readback. The host processor must return this bit to a 1 to remove the $\overline{\text{RD\_CFG}}$ signal. Since this bit works exactly like the $\overline{\text{RD\_CFG}}$ input pin, please see the FPGA pin descriptions for more information on this signal. Default state = 1.
Bit 6	**Reserved.**
Bit 7	**$\overline{\text{PRGM}}$ Input.** Setting this bit to a 0 causes the FPGA to begin configuration and resets the boundary-scan circuitry. The host processor must return this bit to a 1 to remove the $\overline{\text{PRGM}}$ signal. Since this bit works exactly like the $\overline{\text{PRGM}}$ input pin (except that it does not reset the MPI), please see the FPGA pin descriptions for more information on this signal. Default state = 1.

Microprocessor Interface (MPI) (continued)

Device ID Registers

The MPI device ID is broken into four registers holding 1 byte each. The device ID that is available through the MPI is the same as the boundary-scan ID code, except that the device ID in the MPI has a reverse bit order. There is no means to overwrite any of the device ID as can be done with the boundary-scan ID, but the MPI scratchpad register can be used as a personalization register. The format for the entire device ID is shown below followed by family and device values and the partitioning of the device ID into the four device ID registers.

Table 23. Device ID Code

Version	Part*	Family	Manufacturer	MSB
4 bits	10 bits	6 bits	11 bits	1 bit
Example: (First version of OR3C80)			0000 0110100000 110000 00000011101 1	

* PLC array size of FPGA.

Table 24 shows the family and device values for all parts covered by this data sheet.

Table 24. Series 3 Family and Device ID Values

Part Name	Family ID (Hex)	Device ID (Hex)
OR3T20	03	0C
OR3T30	03	0E
OR3T55	03	12
OR3C/T80	03	16
OR3T125	03	1C

Table 25 describes the device IDs for all parts covered by this data sheet as they are partitioned into the four registers found in the MPI.

Table 25. *ORCA* Series 3 Device ID Descriptions

Device ID Register 1	
Bit 0	Logic 1. This bit is always a one.
Bits [7:1]	0011101, the 7 least significant bits of the manufacturer ID.
Device ID Register 2	
Bits [3:0]	0000, the 4 most significant bits of the manufacturer ID.
Bits [7:4]	The 4 least significant bits of the 10-bit part number.
Device ID Register 3	
Bits [5:0]	The 6 most significant bits of the 10-bit part number.
Bits [7:6]	The 2 least significant bits of the device family code.
Device ID Register 4	
Bits [3:0]	The 4 most significant bits of the device family code.
Bits [7:4]	The 4-bit device version code.

Programmable Clock Manager (PCM) (continued)

PCM Detailed Programming

Descriptions of bit fields and individual control bits in the PCM control registers are provided in Table 31. Refer to Figure 46 for more information on the location of the PCM elements that are discussed. In the following discussion, the duty cycle is in the percentage of the clock period where the clock is high.

Table 31. PCM Control Registers

Bit #	Function
Register 0 Divider 0 Programming	
Bits [3:0]	**4-Bit Divider, DIV0, Value**. This value enables the input clock to immediately be divided by a value from 1 to 8. A 0 value (the default) indicates that DIV0 is bypassed (no division). Bypass incurs less delay than dividing by 1. Hexadecimal values greater than 8 for bits [3:0] yield their modulo 8 value. For example, if bits [3:0] are 1001 (9 hex), the result is divide by 1 (remainder 9/8 = 1).
Bits [6:4]	**Reserved**.
Bit 7	**DIV 0 Reset Bit**. DIV0 may not be reset by GSRN depending on the value of register 7, bit 7. This bit may be set to 1 to reset DIV0 to its default value. Bit 0 must be set to 0 (the default) to remove the reset.
Register 1 Divider 1 Programming	
Bits [3:0]	**4-Bit Divider, DIV1, Value**. This value enables the feedback clock to be divided by a value from 1 to 8. A 0 value (the default) indicates that DIV1 is bypassed (no division). Bypass incurs less delay than dividing by 1. Hexadecimal values greater than 8 for bits [3:0] yield their modulo 8 value. For example, if bits [3:0] are 1001 (9 hex), the result is divide by 1 (remainder 9/8 = 1).
Bits [6:4]	**Reserved**.
Bit 7	**DIV1 Reset Bit**. DIV1 may not be reset by GSRN, depending on the value of register 7, bit 7. This bit may be set to 1 to reset DIV1 to its default value. Bit 0 must be set to 0 (the default) to remove the reset.
Register 2 Divider 2 Programming	
Bits [3:0]	**4-Bit Divider, DIV2, Value**. This value enables the tapped delay line output clock driven onto ExpressCLK to be divided by a value from 1 to 8. A 0 value (the default) indicates that DIV2 is bypassed (no division). Bypass incurs less delay than dividing by 1. Hexadecimal values greater than 8 for bits [3:0] yield their modulo 8 value. For example, if bits [3:0] are 1001 (9 hex), the result is divide by 1 (remainder 9/8 = 1).
Bits [6:4]	**Reserved**.
Bit 7	**DIV2 Reset Bit**. DIV2 may not be reset by GSRN, depending on the value of register 7, bit 7. This bit may be set to 1 to reset DIV2 to its default value. Bit 7 must be set to 0 (the default) to remove the reset.
Register 3 DLL 2x Duty-Cycle Programming	
Bits [2:0]	Duty-cycle selection for the doubled clock period associated with the input clock high. The duty cycle is (value of bit 6) * 50% + ((value of bits [2:0]) + 1) * 6.25%. See the description for bit 6.
Bits [5:3]	Duty-cycle selection for the doubled clock period associated with the input clock low. The duty cycle is (value of bit 7) * 50% + ((value of bits [2:0]) + 1) * 6.25%. See the description for bit 7.
Bit 6	Master duty-cycle control for the first clock period of the doubled clock: 0 = less than or equal to 50%, 1 = greater than 50%.
Bit 7	Master duty-cycle control for the second clock period of the doubled clock: 0 = less than or equal to 50%, 1 = greater than 50%. Example: Both clock periods having a 62.5% duty cycle, bits [7:0] are 11 001 001.

FPGA States of Operation (continued)

If configuration has begun, an assertion of $\overline{\text{RESET}}$ or $\overline{\text{PRGM}}$ initiates an abort, returning the FPGA to the initialization state. The $\overline{\text{PRGM}}$ and $\overline{\text{RESET}}$ pins must be pulled back high before the FPGA will enter the configuration state. During the start-up and operating states, only the assertion of $\overline{\text{PRGM}}$ causes a reconfiguration.

In the master configuration modes, the FPGA is the source of configuration clock (CCLK). In this mode, the initialization state is extended to ensure that, in daisy-chain operation, all daisy-chained slave devices are ready. Independent of differences in clock rates, master mode devices remain in the initialization state an additional six internal clock cycles after $\overline{\text{INIT}}$ goes high.

When configuration is initiated, a counter in the FPGA is set to 0 and begins to count configuration clock cycles applied to the FPGA. As each configuration data frame is supplied to the FPGA, it is internally assembled into data words. Each data word is loaded into the internal configuration memory. The configuration loading process is complete when the internal length count equals the loaded length count in the length count field, and the required end of configuration frame is written.

All OR3Cxx I/Os operate as TTL inputs during configuration (OR3Txxx I/Os are CMOS-only). All I/Os that are not used during the configuration process are 3-stated with internal pull-ups.

Warning: During configuration, all OR3Txxx inputs have internal pull-ups enabled. If these inputs are driven to 5V, they will draw substantial current (≅ 5 ma). This is due to the fact that the inputs are pulled up to 3V.

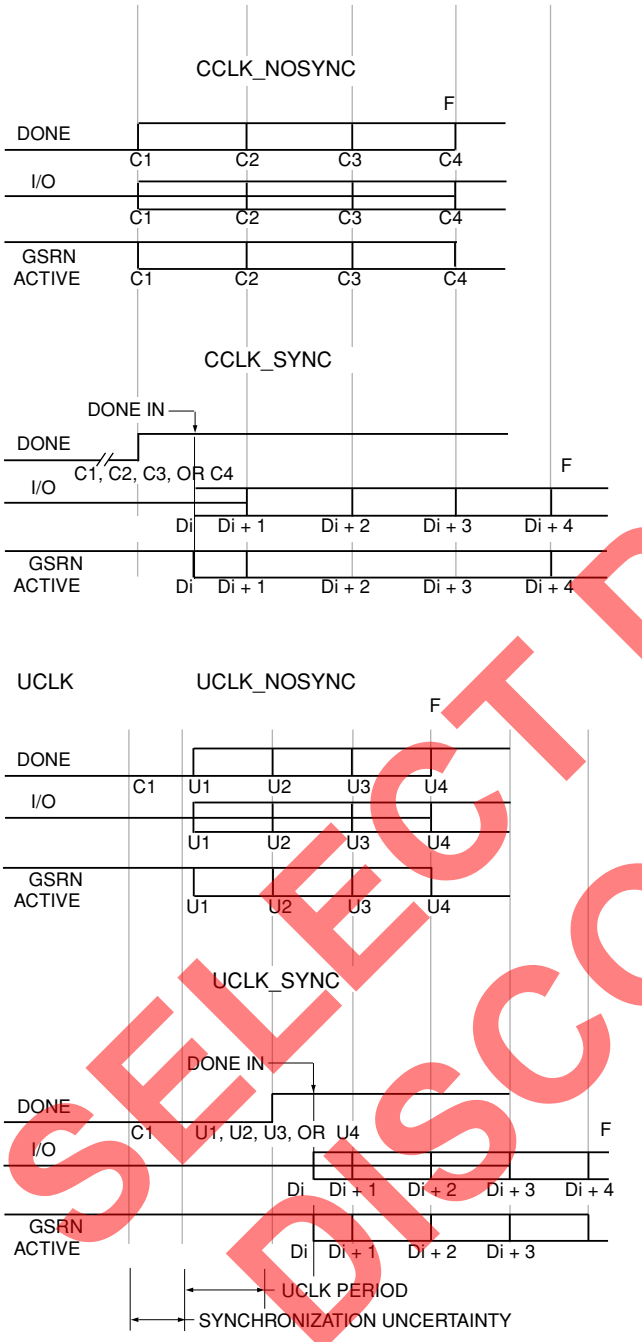
During configuration, the PIC and PLC latches/FFs are held set/reset and the internal BIDI buffers are 3-stated. The combinatorial logic begins to function as the FPGA is configured. Figure 50 shows the general waveform of the initialization, configuration, and start-up states.

Configuration

The *ORCA* Series FPGA functionality is determined by the state of internal configuration RAM. This configuration RAM can be loaded in a number of different modes. In these configuration modes, the FPGA can act as a master or a slave of other devices in the system. The decision as to which configuration mode to use is a system design issue. Configuration is discussed in detail, including the configuration data format and the configuration modes used to load the configuration data in the FPGA, following a description of the start-up state.

Figure 50. Initialization/Configuration/Start-Up Waveforms

FPGA States of Operation (continued)



Note: F = finished, no more CLKs required.

5-2761(F)

Figure 51. Start-Up Waveforms

Reconfiguration

To reconfigure the FPGA when the device is operating in the system, a low pulse is input into $\overline{\text{PRGM}}$. The configuration data in the FPGA is cleared, and the I/Os not used for configuration are 3-stated. The FPGA then samples the mode select inputs and begins reconfiguration. When reconfiguration is complete, DONE is released, allowing it to be pulled high.

Partial Reconfiguration

All *ORCA* device families have been designed to allow a partial reconfiguration of the FPGA at any time. This is done by setting a bit stream option in the previous configuration sequence that tells the FPGA to not reset all of the configuration RAM during a reconfiguration. Then only the configuration frames that are to be modified need to be rewritten, thereby reducing the configuration time.

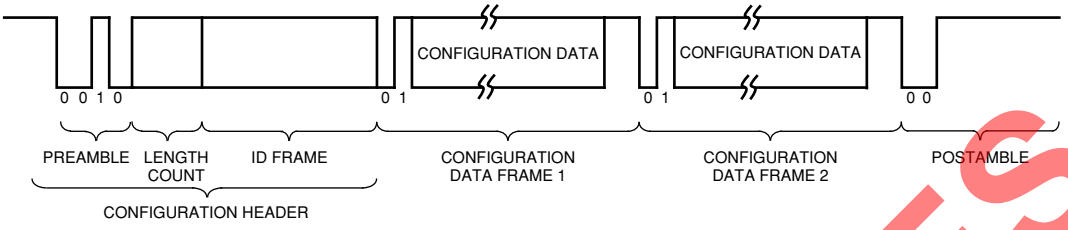
Other bit stream options are also available that allow one portion of the FPGA to remain in operation while a partial reconfiguration is being done. If this is done, the user must be careful to not cause contention between the two configurations (the bit stream resident in the FPGA and the partial reconfiguration bit stream) as the second reconfiguration bit stream is being loaded.

Other Configuration Options

There are many other configuration options available to the user that can be set during bit stream generation in ispLEVER. These include options to enable boundary scan and/or the microprocessor interface (MPI) and/or the programmable clock manager (PCM), readback options, and options to control and use the internal oscillator after configuration.

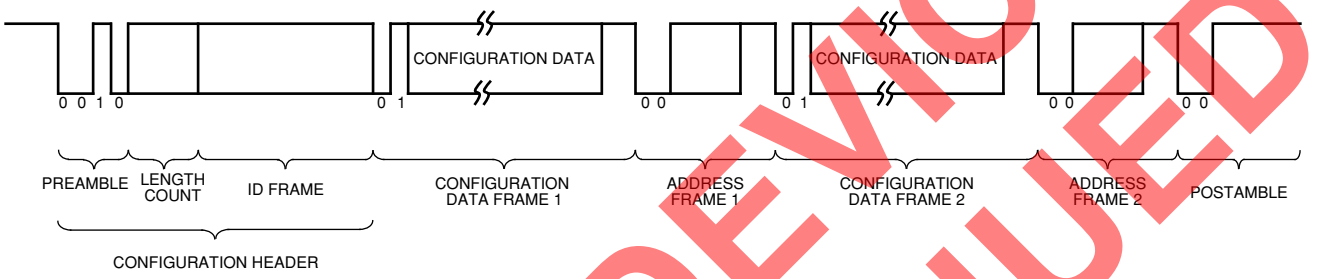
Other useful options that affect the next configuration (not the current configuration process) include options to disable the global set/reset during configuration, disable the 3-state of I/Os during configuration, and disable the reset of internal RAMs during configuration to allow for partial configurations (see above). For more information on how to set these and other configuration options, please see the ispLEVER documentation.

Configuration Data Format (continued)



5-5759(F)

Figure 52. Serial Configuration Data Format—Autoincrement Mode



5-5760(F)

Figure 53. Serial Configuration Data Format—Explicit Mode

Table 32. Configuration Frame Format and Contents

Header	11110010	Preamble
	24-bit Length Count	Configuration frame length.
	11111111	Trailing header—8 bits.
ID Frame	0101 1111 1111 1111	ID frame header.
	Configuration Mode	00 = autoincrement, 01 = explicit.
	Reserved [41:0]	Reserved bits set to 0.
	ID	20-bit part ID.
	Checksum	8-bit checksum.
	11111111	Eight stop bits (high) to separate frames.
Configuration Data Frame (repeated for each data frame)	01	Data frame header.
	Data Bits	Number of data bits depends upon device.
	Alignment Bits = 0	String of 0 bits added to bit stream to make frame header, plus data bits reach a byte boundary.
	Checksum	8-bit checksum.
	11111111	Eight stop bits (high) to separate frames.
Configuration Address Frame	00	Address frame header.
	14 Address Bits	14-bit address of location to start data storage.
	Checksum	8-bit checksum.
	11111111	Eight stop bits (high) to separate frames.
Postamble	00	Postamble header.
	11111111 111111	Dummy address.
	1111111111111111	16 stop bits.*

* In MPI configuration mode, the number of stop bits = 32.

Note: For slave parallel mode, the byte containing the preamble must be 11110010. The number of leading header dummy bits must be (n * 8) + 4, where n is any nonnegative integer and the number of trailing dummy bits must be (n * 8), where n is any positive integer. The number of stop bits/frame for slave parallel mode must be (x * 8), where x is a positive integer. Note also that the bit stream generator tool supplies a bit stream that is compatible with all configuration modes, including slave parallel mode.

Timing Characteristics (continued)

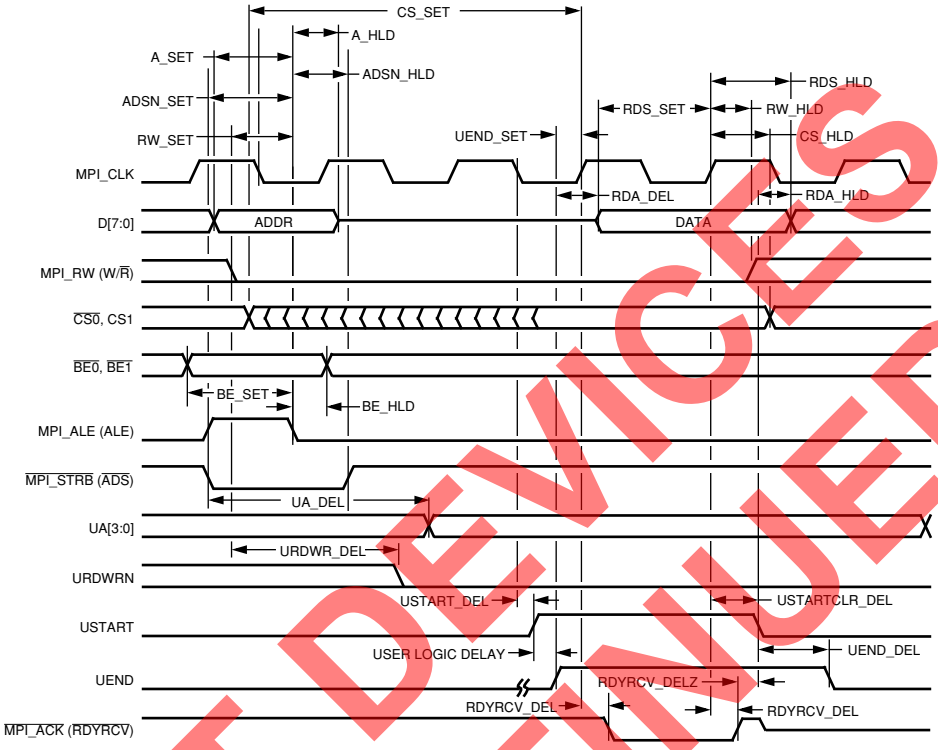
Table 43. Ripple Mode PFU Timing Characteristics (continued)

OR3Cxx Commercial: V_{DD} = 5.0 V ± 5%, 0 °C < T_A < 70 °C; Industrial: V_{DD} = 5.0 V ± 10%, –40 °C < T_A < +85 °C.
OR3Txxx Commercial: V_{DD} = 3.0 V to 3.6 V, 0 °C < T_A < 70 °C; Industrial: V_{DD} = 3.0 V to 3.6 V, –40 °C < T_A < +85 °C.

Parameter (T_J = +85 °C, V_{DD} = min)	Symbol	Speed								Unit
		-4		-5		-6		-7		
		Min	Max	Min	Max	Min	Max	Min	Max	
Full Ripple Delays (byte wide):										
Operands to Carry-out (Kz[1:0] to COUT)	RIPCO_DEL	—	5.32	—	4.11	—	2.98	—	2.32	ns
Operands to Carry-out (Kz[1:0] to FCOUT)	RIPFCO_DEL	—	5.30	—	4.10	—	2.98	—	2.32	ns
Operands to PFU Out (Kz[1:0] to F[7:0])	RIP_DEL	—	7.37	—	5.60	—	4.18	—	3.10	ns
Bitwise Operands to PFU Out (Kz[1:0] to F[z])	FRIP_DEL	—	2.34	—	1.80	—	1.32	—	1.05	ns
Fast Carry-in to Carry-out (FCIN to COUT)	FCINCO_DEL	—	2.59	—	1.99	—	1.43	—	1.14	ns
Fast Carry-in to Fast Carry-out (FCIN to FCOUT)	FCINFCO_DEL	—	2.57	—	1.98	—	1.41	—	1.13	ns
Carry-in to Carry-out (CIN to COUT)	CINCO_DEL	—	3.47	—	2.65	—	1.79	—	1.43	ns
Carry-in to Fast Carry-out (CIN to FCOUT)	CINFCO_DEL	—	3.46	—	2.64	—	1.78	—	1.43	ns
Fast Carry-in PFU Out (FCIN to F[7:0])	FCIN_DEL	—	6.03	—	4.55	—	3.21	—	2.51	ns
Carry-in PFU Out (CIN to F[7:0])	CIN_DEL	—	6.91	—	5.21	—	3.53	—	3.05	ns
Add/Subtract to Carry-out (ASWE to COUT)	ASCO_DEL	—	8.28	—	5.89	—	4.58	—	3.45	ns
Add/Subtract to Carry-out (ASWE to FCOUT)	ASFCO_DEL	—	8.11	—	5.78	—	4.48	—	3.38	ns
Add/Subtract to PFU Out (ASWE to F[7:0])	AS_DEL	—	10.66	—	7.55	—	5.85	—	4.38	ns
Half Ripple Delays (nibble wide):										
Operands to Carry-out (Kz[1:0] to COUT)	HRIPCO_DEL	—	5.32	—	4.11	—	2.98	—	2.32	ns
Operands to Fast Carry-out (Kz[1:0] to FCOUT)	HRIPFCO_DEL	—	5.30	—	4.10	—	2.98	—	2.32	ns
Operands to PFU Out (Kz[1:0] to F[3:0])	HRIP_DEL	—	5.50	—	4.07	—	3.20	—	2.40	ns
Bitwise Operands to PFU Out (Kz[1:0] to F[z])	HFRIP_DEL	—	2.34	—	1.80	—	1.32	—	1.05	ns
Fast Carry-in to Carry-out (FCIN to COUT)	HFCINCO_DEL	—	2.59	—	1.99	—	1.43	—	1.14	ns
Fast Carry-in to Fast Carry-out (FCIN to FCOUT)	HFCINFCO_DEL	—	2.57	—	1.98	—	1.41	—	1.13	ns
Carry-in to Carry-out (CIN to COUT)	HCINCO_DEL	—	3.47	—	2.65	—	1.79	—	1.43	ns
Carry-in to Carry-out (CIN to FCOUT)	HCINFCO_DEL	—	3.46	—	2.64	—	1.78	—	1.43	ns
Fast Carry-in PFU Out (FCIN to F[3:0])	HFCIN_DEL	—	3.76	—	2.84	—	2.01	—	1.58	ns
Carry-in PFU Out (CIN to F[3:0])	HCIN_DEL	—	4.65	—	3.50	—	2.33	—	2.12	ns
Add/Subtract to Carry-out (ASWE to COUT)	HASCO_DEL	—	8.28	—	5.89	—	4.58	—	3.45	ns
Add/Subtract to Carry-out (ASWE to FCOUT)	HASFCO_DEL	—	8.11	—	5.78	—	4.48	—	3.38	ns
Add/Subtract to PFU Out (ASWE to F[3:0])	HAS_DEL	—	9.12	—	6.49	—	4.86	—	3.69	ns

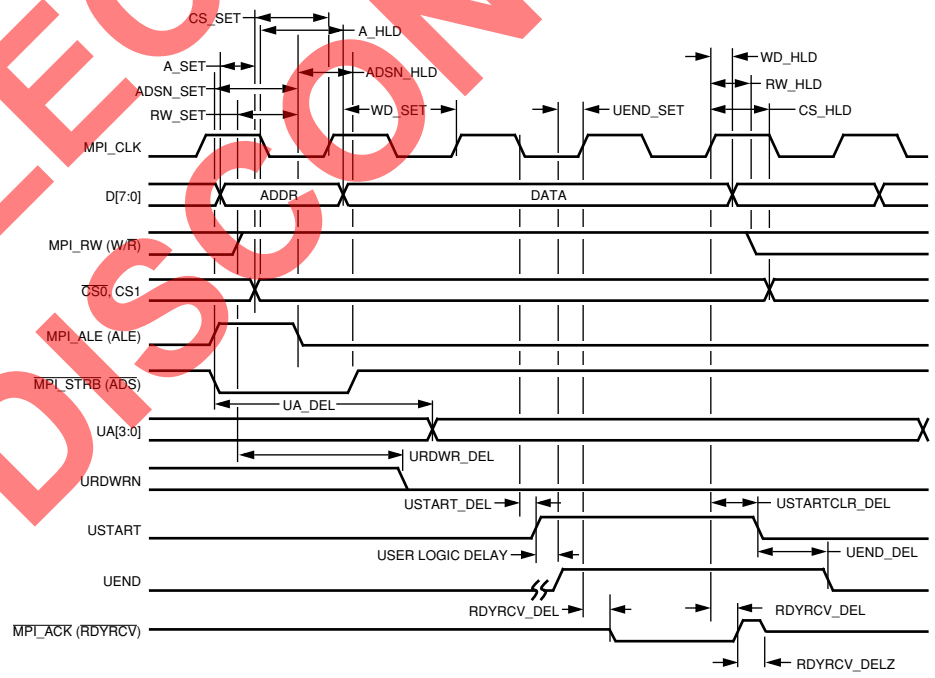
Note: The table shows worst-case delay for the ripple chain. ispLEVER reports the delay for individual paths within the ripple chain that will be less than or equal to those listed above.

Timing Characteristics (continued)



5-5831(F).b

Figure 71. MPI *i960* User Space Read Timing



5-5830(F).b

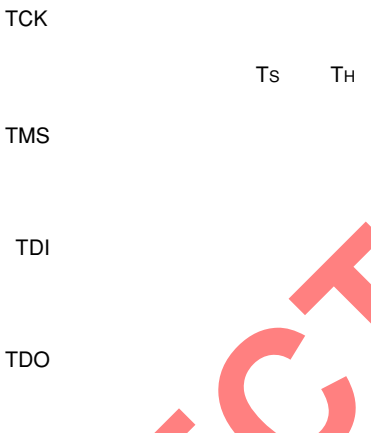
Figure 72. MPI *i960* User Space Write Timing

Timing Characteristics (continued)

Table 51. Boundary-Scan Timing Characteristics

OR3Cxx Commercial: V_{DD} = 5.0 V ± 5%, 0 °C < T_A < 70 °C; Industrial: V_{DD} = 5.0 V ± 10%, –40 °C < T_A < +85 °C.
OR3Txxx Commercial: V_{DD} = 3.0 V to 3.6 V, 0 °C < T_A < 70 °C; Industrial: V_{DD} = 3.0 V to 3.6 V, –40 °C < T_A < +85 °C.

Parameter	Symbol	Min	Max	Unit
TDI/TMS to TCK Setup Time	T_S	25.0	—	ns
TDI/TMS Hold Time from TCK	T_H	0.0	—	ns
TCK Low Time	T_{CL}	50.0	—	ns
TCK High Time	T_{CH}	50.0	—	ns
TCK to TDO Delay	T_D	—	20.0	ns
TCK Frequency	T_{TCK}	—	10.0	MHz



5-6764(F)

Figure 75. Boundary-Scan Timing Diagram

Pin	OR3C/T80 Pad	OR3T125 Pad	Function
AC30	PL18D	PL22D	I/O
AC31	PL17A	PL21A	I/O
AB29	PL17B	PL21B	I/O-A13
AB30	PL17C	PL21C	I/O
AB31	PL17D	PL21D	I/O
AA29	PL16A	PL20A	I/O
Y28	PL16B	PL20B	I/O
AA30	PL16C	PL20C	I/O
AA31	PL16D	PL20D	I/O-A12
Y29	PL15A	PL19A	I/O-A11/$\overline{\text{MPI\_IRQ}}$
W28	PL15B	PL19D	I/O
Y30	PL15C	PL18A	I/O
W29	PL14A	PL18C	I/O
W30	PL14B	PL18D	I/O
V28	PL14C	PL17A	I/O-A10/$\overline{\text{MPI\_BI}}$
W31	PL14D	PL17C	I/O
V29	PL13A	PL17D	I/O
V30	PL13B	PL16A	I/O
V31	PL13C	PL16C	I/O
U29	PL13D	PL16D	I/O-A9/$\overline{\text{MPI\_ACK}}$
U30	PL12A	PL15A	I/O-A8/MPI_RW
U31	PL12B	PL15B	I/O
T30	PL12C	PL15C	I/O
T28	PL12D	PL15D	I/O
T29	PL11A	PL14A	I/O-A7/MPI_CLK
R31	PL11B	PL14B	I/O
R30	PL11C	PL14C	I/O
R29	PECKL	PECKL	I-ECKL
P31	PL10A	PL13A	I/O-A6
P30	PL10B	PL13D	I/O
P29	PL10C	PL12A	I/O
N31	PL10D	PL12C	I/O
P28	PL9A	PL12D	I/O-A5
N30	PL9B	PL11A	I/O-A4
N29	PL9C	PL11C	I/O
M30	PL9D	PL11D	I/O
N28	PL8A	PL10A	I/O
M29	PL8C	PL10C	I/O
L31	PL8D	PL10D	I/O
L30	PL7A	PL9A	I/O-A3
M28	PL7B	PL9B	I/O
L29	PL7C	PL9C	I/O
K31	PL7D	PL9D	I/O
K30	PL6A	PL8A	I/O-A2
K29	PL6B	PL8B	I/O
J31	PL6C	PL8C	I/O
J30	PL6D	PL8D	I/O
K28	PL5A	PL7D	I/O-A1/$\overline{\text{MPI\_BE1}}$
J29	PL5B	PL6B	I/O
H30	PL5C	PL6C	I/O
H29	PL5D	PL6D	I/O
J28	PL4A	PL5D	I/O
G31	PL4B	PL4B	I/O
G30	PL4C	PL4C	I/O
G29	PL4D	PL4D	I/O
H28	PL3A	PL3A	I/O
F31	PL3B	PL3B	I/O
F30	PL3C	PL3C	I/O
F29	PL3D	PL3D	I/O
E31	PL2A	PL2A	I/O
E30	PL2B	PL2B	I/O
E29	PL2C	PL2C	I/O
F28	PL2D	PL2D	I/O-A0/$\overline{\text{MPI\_BE0}}$
D31	PL1A	PL1A	I/O
D30	PL1B	PL1B	I/O
D29	PL1C	PL1C	I/O
E28	PL1D	PL1D	I/O
D27	PRD_DATA	PRD_DATA	RD_DATA/TDO
C28	PT1A	PT1A	I/O-TCK
B28	PT1B	PT1B	I/O
A28	PT1C	PT1C	I/O
D26	PT1D	PT1D	I/O
C27	PT2A	PT2A	I/O
B27	PT2B	PT2B	I/O
A27	PT2C	PT2C	I/O
C26	PT2D	PT2D	I/O
B26	PT3A	PT3A	I/O
A26	PT3B	PT3B	I/O
D24	PT3C	PT3C	I/O
C25	PT3D	PT3D	I/O
B25	PT4A	PT4A	I/O-TMS
A25	PT4B	PT4B	I/O
D23	PT4C	PT4C	I/O
C24	PT4D	PT4D	I/O
B24	PT5A	PT5A	I/O
C23	PT5B	PT5B	I/O
D22	PT5C	PT5C	I/O
B23	PT5D	PT5D	I/O
A23	PT6A	PT6A	I/O-TDI
C22	PT6B	PT6D	I/O
B22	PT6C	PT7A	I/O
A22	PT6D	PT7D	I/O
C21	PT7A	PT8A	I/O
D20	PT7B	PT8D	I/O
B21	PT7C	PT9A	I/O
A21	PT7D	PT9D	I/O
C20	PT8A	PT10A	I/O-DOUT
D19	PT8C	PT10D	I/O
B20	PT8D	PT11A	I/O
C19	PT9A	PT11C	I/O

ψJC

This JEDEC designated parameter correlates the junction temperature to the case temperature. It is generally used to infer the junction temperature while the device is operating in the system. It is not considered a true thermal resistance, and it is defined by:

$$\psi_{JC} = \frac{T_J - T_C}{Q}$$

where T_C is the case temperature at top dead center, T_J is the junction temperature, and Q is the chip power. During the Θ_{JA} measurements described above, besides the other parameters measured, an additional temperature reading, T_C, is made with a thermocouple attached at top-dead-center of the case. ψ_{JC} is also expressed in units of °C/watt.

ΘJC

This is the thermal resistance from junction to case. It is most often used when attaching a heat sink to the top of the package. It is defined by:

$$\Theta_{JC} = \frac{T_J - T_C}{Q}$$

The parameters in this equation have been defined above. However, the measurements are performed with the case of the part pressed against a water-cooled heat sink so as to draw most of the heat generated by the chip out the top of the package. It is this difference in the measurement process that differentiates Θ_{JC} from ψ_{JC}. Θ_{JC} is a true thermal resistance and is expressed in units of °C/watt.

ΘJB

This is the thermal resistance from junction to board (a.k.a. Θ_{JL}). It is defined by:

$$\Theta_{JB} = \frac{T_J - T_B}{Q}$$

where T_B is the temperature of the board adjacent to a lead measured with a thermocouple. The other parameters on the right-hand side have been defined above. This is considered a true thermal resistance, and the measurement is made with a water-cooled heat sink pressed against the board so as to draw most of the heat out of the leads. Note that Θ_{JB} is expressed in units of °C/watt, and that this parameter and the way it is measured is still in JEDEC committee.

Package Outline Diagrams (continued)

208-Pin SQFP2

Dimensions are in millimeters.

5-3828(F)

DETAIL C (SQFP2 CHIP-UP)

5-3828(F).a