MachXO Family Handbook

HB1002 Version 02.5, December 2010

MachXO Family Handbook Table of Contents

December 2010

Section I. MachXO Family Data Sheet

DS1002 Version 02.9, July 2010

MachXO Family Data Sheet Introduction

June 2009 Data Sheet DS1002

Features

- **Non-volatile, Infinitely Reconfigurable**
  - Instant-on – powers up in microseconds
  - Single chip, no external configuration memory required
  - Excellent design security, no bit stream to intercept
  - Reconfigure SRAM based logic in milliseconds
  - SRAM and non-volatile memory programmable through JTAG port
  - Supports background programming of non-volatile memory
- **Sleep Mode**
  - Allows up to 100x static current reduction
- **TransFR™ Reconfiguration (TFR)**
  - In-field logic update while system operates
- **High I/O to Logic Density**
  - 256 to 2280 LUT4s
  - 73 to 271 I/Os with extensive package options
  - Density migration supported
  - Lead free/RoHS compliant packaging
- **Embedded and Distributed Memory**
  - Up to 27.6 Kbits sysMEM™ Embedded Block RAM
  - Up to 7.7 Kbits distributed RAM
  - Dedicated FIFO control logic
- **Flexible I/O Buffer**
  - Programmable sysIO™ buffer supports wide range of interfaces:
    - LVCMOS 3.3/2.5/1.8/1.5/1.2
    - LVTTL
    - PCI
    - LVDS, Bus-LVDS, LVPECL, RSDS
- **sysCLOCK™ PLLs**
  - Up to two analog PLLs per device
  - Clock multiply, divide, and phase shifting
- **System Level Support**
  - IEEE Standard 1149.1 Boundary Scan
  - Onboard oscillator
  - Devices operate with 3.3V, 2.5V, 1.8V or 1.2V power supply
  - IEEE 1532 compliant in-system programming

Introduction

The MachXO is optimized to meet the requirements of applications traditionally addressed by CPLDs and low capacity FPGAs: glue logic, bus bridging, bus interfacing, power-up control, and control logic. These devices bring together the best features of CPLD and FPGA devices on a single chip.

Table 1-1. MachXO Family Selection Guide

Device	LCMXO256	LCMXO640	LCMXO1200	LCMXO2280
LUTs	256	640	1200	2280
Dist. RAM (Kbits)	2.0	6.1	6.4	7.7
EBR SRAM (Kbits)	0	0	9.2	27.6
Number of EBR SRAM Blocks (9 Kbits)	0	0	1	3
V_{CC} Voltage	1.2/1.8/2.5/3.3V	1.2/1.8/2.5/3.3V	1.2/1.8/2.5/3.3V	1.2/1.8/2.5/3.3V
Number of PLLs	0	0	1	2
Max. I/O	78	159	211	271
Packages				
100-pin TQFP (14x14 mm)	78	74	73	73
144-pin TQFP (20x20 mm)		113	113	113
100-ball csBGA (8x8 mm)	78	74		
132-ball csBGA (8x8 mm)		101	101	101
256-ball caBGA (14x14 mm)		159	211	211
256-ball ftBGA (17x17 mm)		159	211	211
324-ball ftBGA (19x19 mm)				271

The devices use look-up tables (LUTs) and embedded block memories traditionally associated with FPGAs for flexible and efficient logic implementation. Through non-volatile technology, the devices provide the single-chip, high-security, instant-on capabilities traditionally associated with CPLDs. Finally, advanced process technology and careful design will provide the high pin-to-pin performance also associated with CPLDs.

The ispLEVER® design tools from Lattice allow complex designs to be efficiently implemented using the MachXO family of devices. Popular logic synthesis tools provide synthesis library support for MachXO. The ispLEVER tools use the synthesis tool output along with the constraints from its floor planning tools to place and route the design in the MachXO device. The ispLEVER tool extracts the timing from the routing and back-annotates it into the design for timing verification.

MachXO Family Data Sheet Architecture

February 2007 Data Sheet DS1002

Architecture Overview

The MachXO family architecture contains an array of logic blocks surrounded by Programmable I/O (PIO). Some devices in this family have sysCLOCK PLLs and blocks of sysMEM™ Embedded Block RAM (EBRs). Figures 2-1, 2-2, and 2-3 show the block diagrams of the various family members.

The logic blocks are arranged in a two-dimensional grid with rows and columns. The EBR blocks are arranged in a column to the left of the logic array. The PIO cells are located at the periphery of the device, arranged into Banks. The PIOs utilize a flexible I/O buffer referred to as a sysIO interface that supports operation with a variety of interface standards. The blocks are connected with many vertical and horizontal routing channel resources. The place and route software tool automatically allocates these routing resources.

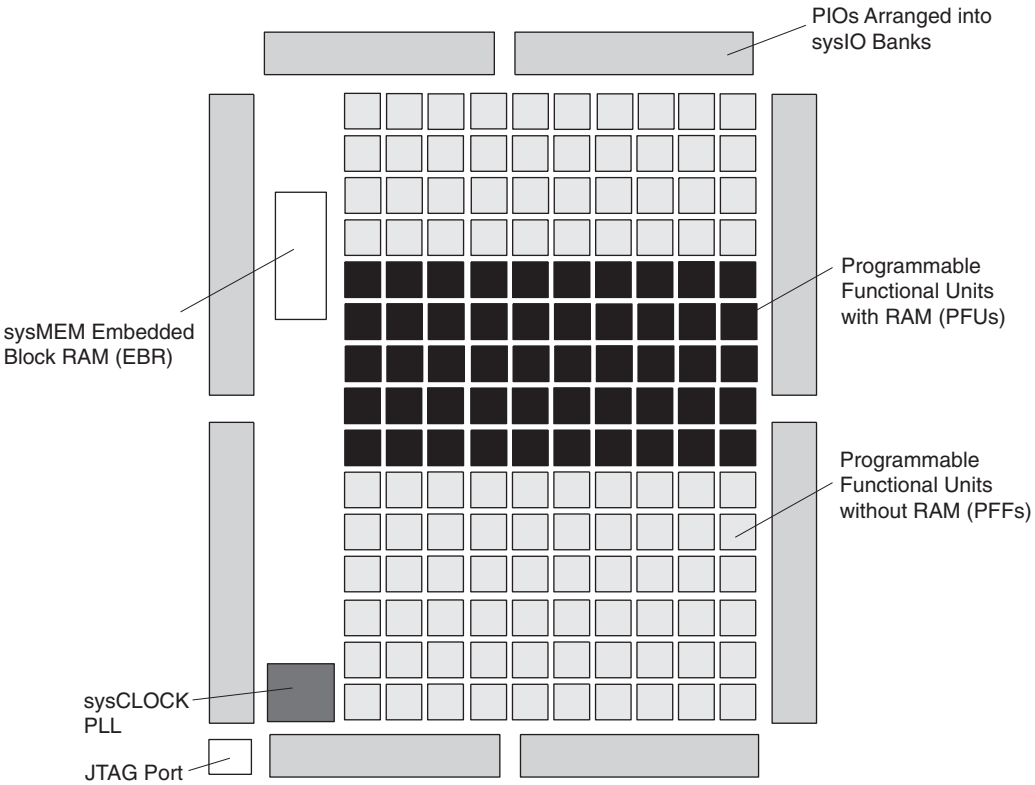
There are two kinds of logic blocks, the Programmable Functional Unit (PFU) and the Programmable Functional unit without RAM (PFF). The PFU contains the building blocks for logic, arithmetic, RAM, ROM, and register functions. The PFF block contains building blocks for logic, arithmetic, ROM, and register functions. Both the PFU and PFF blocks are optimized for flexibility, allowing complex designs to be implemented quickly and effectively. Logic blocks are arranged in a two-dimensional array. Only one type of block is used per row.

In the MachXO family, the number of sysIO Banks varies by device. There are different types of I/O Buffers on different Banks. See the details in later sections of this document. The sysMEM EBRs are large, dedicated fast memory blocks; these blocks are found only in the larger devices. These blocks can be configured as RAM, ROM or FIFO. FIFO support includes dedicated FIFO pointer and flag “hard” control logic to minimize LUT use.

The MachXO architecture provides up to two sysCLOCK™ Phase Locked Loop (PLL) blocks on larger devices. These blocks are located at either end of the memory blocks. The PLLs have multiply, divide, and phase shifting capabilities that are used to manage the frequency and phase relationships of the clocks.

Every device in the family has a JTAG Port that supports programming and configuration of the device as well as access to the user logic. The MachXO devices are available for operation from 3.3V, 2.5V, 1.8V, and 1.2V power supplies, providing easy integration into the overall system.

Figure 2-1. Top View of the MachXO1200 Device[1]



1. Top view of the MachXO2280 device is similar but with higher LUT count, two PLLs, and three EBR blocks.

Figure 2-2. Top View of the MachXO640 Device

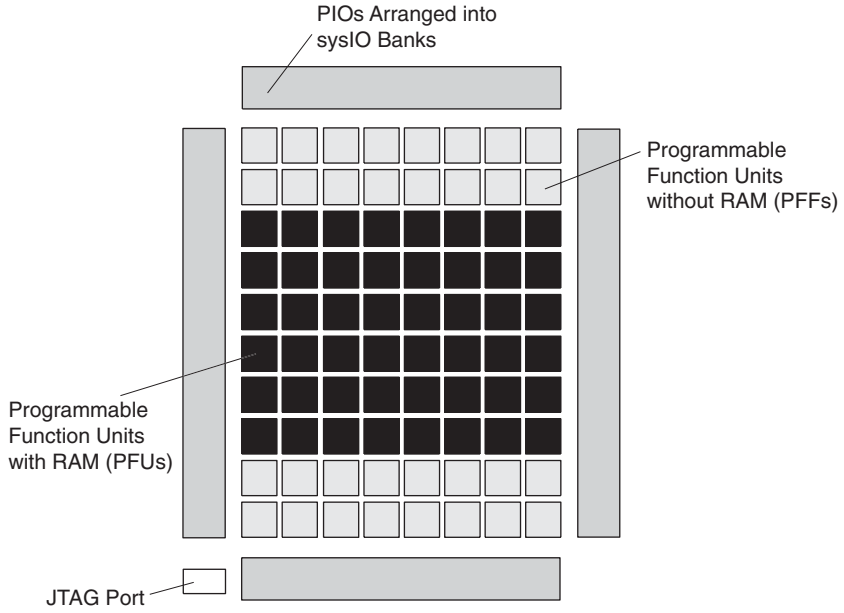
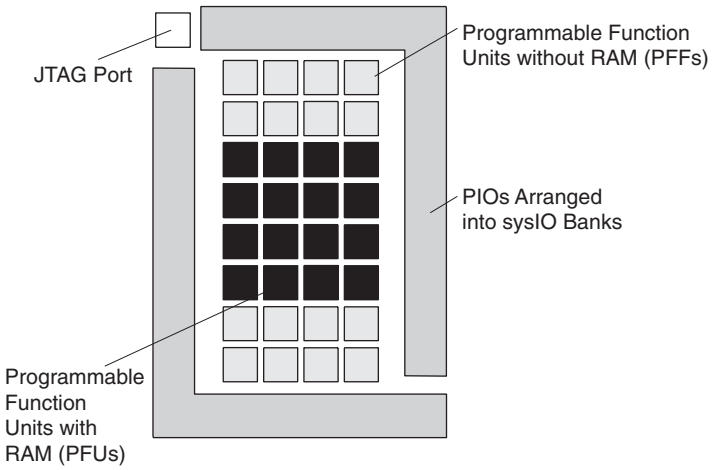


Figure 2-3. Top View of the MachXO256 Device

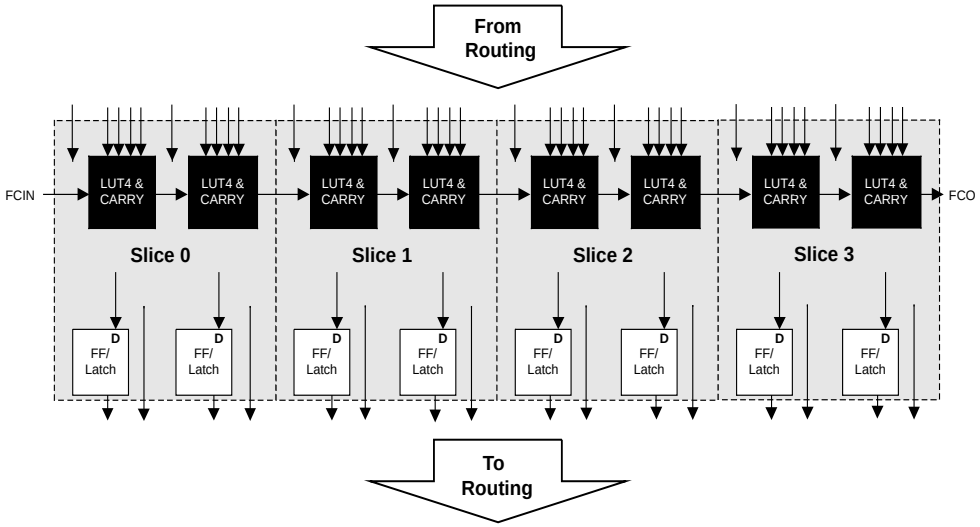


PFU Blocks

The core of the MachXO devices consists of PFU and PFF blocks. The PFUs can be programmed to perform Logic, Arithmetic, Distributed RAM, and Distributed ROM functions. PFF blocks can be programmed to perform Logic, Arithmetic, and Distributed ROM functions. Except where necessary, the remainder of this data sheet will use the term PFU to refer to both PFU and PFF blocks.

Each PFU block consists of four interconnected Slices, numbered 0-3 as shown in Figure 2-4. There are 53 inputs and 25 outputs associated with each PFU block.

Figure 2-4. PFU Diagram

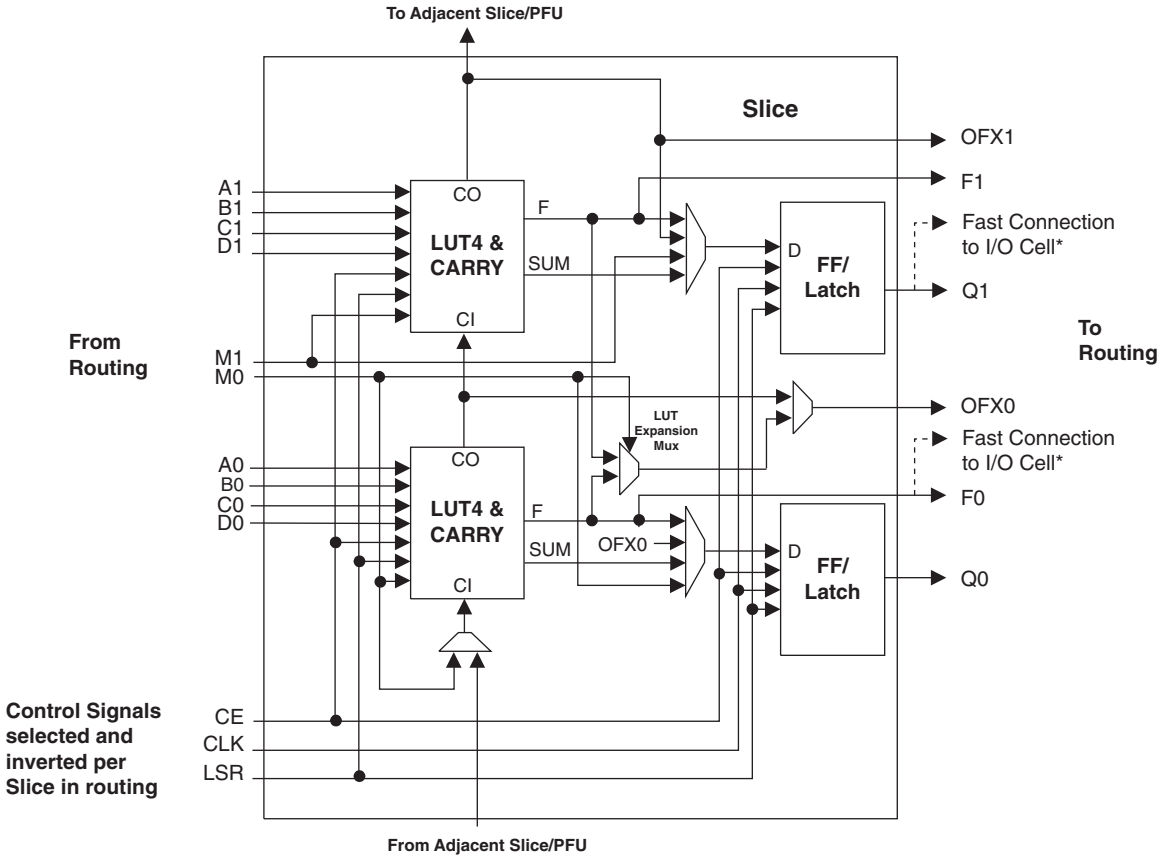


Slice

Each Slice contains two LUT4 lookup tables feeding two registers (programmed to be in FF or Latch mode), and some associated logic that allows the LUTs to be combined to perform functions such as LUT5, LUT6, LUT7, and LUT8. There is control logic to perform set/reset functions (programmable as synchronous/asynchronous), clock select, chip-select, and wider RAM/ROM functions. Figure 2-5 shows an overview of the internal logic of the Slice. The registers in the Slice can be configured for positive/negative and edge/level clocks.

There are 14 input signals: 13 signals from routing and one from the carry-chain (from the adjacent Slice/PFU). There are 7 outputs: 6 to the routing and one to the carry-chain (to the adjacent Slice/PFU). Table 2-1 lists the signals associated with each Slice.

Figure 2-5. Slice Diagram



Notes:
Some inter-Slice signals are not shown.
*** Only PFUs at the edges have fast connections to the I/O cell.**

Table 2-1. Slice Signal Descriptions

Function	Type	Signal Names	Description
Input	Data signal	A0, B0, C0, D0	Inputs to LUT4
Input	Data signal	A1, B1, C1, D1	Inputs to LUT4
Input	Multi-purpose	M0/M1	Multipurpose Input
Input	Control signal	CE	Clock Enable
Input	Control signal	LSR	Local Set/Reset
Input	Control signal	CLK	System Clock
Input	Inter-PFU signal	FCIN	Fast Carry In[1]
Output	Data signals	F0, F1	LUT4 output register bypass signals
Output	Data signals	Q0, Q1	Register Outputs
Output	Data signals	OFX0	Output of a LUT5 MUX
Output	Data signals	OFX1	Output of a LUT6, LUT7, LUT8[2] MUX depending on the Slice
Output	Inter-PFU signal	FCO	Fast Carry Out[1]

1. See Figure 2-4 for connection details.
2. Requires two PFUs.

Modes of Operation

Each Slice is capable of four modes of operation: Logic, Ripple, RAM, and ROM. The Slice in the PFF is capable of all modes except RAM. Table 2-2 lists the modes and the capability of the Slice blocks.

Table 2-2. Slice Modes

	Logic	Ripple	RAM	ROM
PFU Slice	LUT 4x2 or LUT 5x1	2-bit Arithmetic Unit	SP 16x2	ROM 16x1 x 2
PFF Slice	LUT 4x2 or LUT 5x1	2-bit Arithmetic Unit	N/A	ROM 16x1 x 2

Logic Mode: In this mode, the LUTs in each Slice are configured as 4-input combinatorial lookup tables (LUT4). A LUT4 can have 16 possible input combinations. Any logic function with four inputs can be generated by programming this lookup table. Since there are two LUT4s per Slice, a LUT5 can be constructed within one Slice. Larger lookup tables such as LUT6, LUT7, and LUT8 can be constructed by concatenating other Slices.

Ripple Mode: Ripple mode allows the efficient implementation of small arithmetic functions. In ripple mode, the following functions can be implemented by each Slice:

- Addition 2-bit
- Subtraction 2-bit
- Add/Subtract 2-bit using dynamic control
- Up counter 2-bit
- Down counter 2-bit
- Ripple mode multiplier building block
- Comparator functions of A and B inputs
  - A greater-than-or-equal-to B
  - A not-equal-to B
  - A less-than-or-equal-to B

Two additional signals, Carry Generate and Carry Propagate, are generated per Slice in this mode, allowing fast arithmetic functions to be constructed by concatenating Slices.

RAM Mode: In this mode, distributed RAM can be constructed using each LUT block as a 16x2-bit memory. Through the combination of LUTs and Slices, a variety of different memories can be constructed.

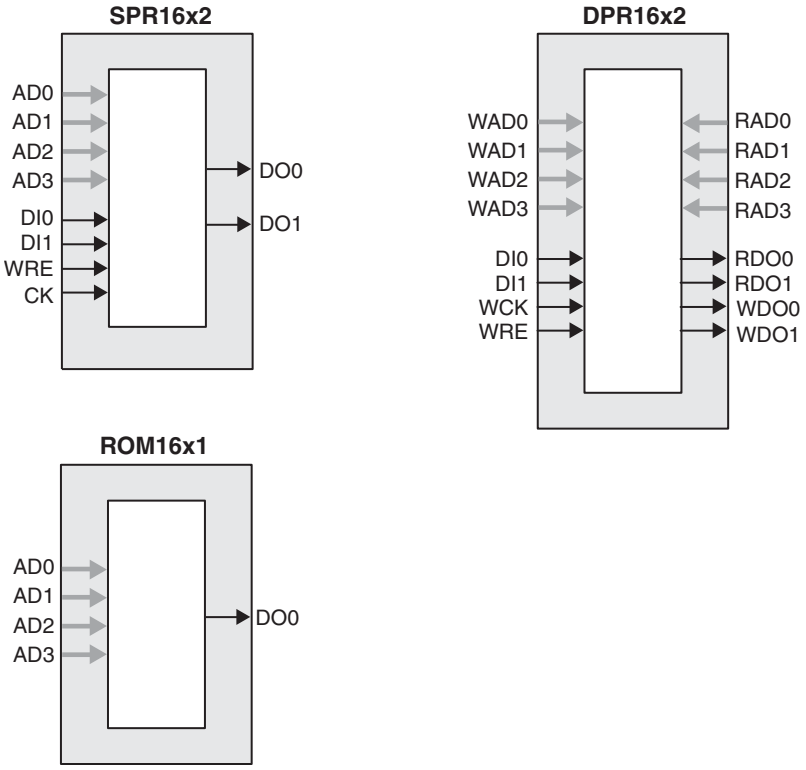
The ispLEVER design tool supports the creation of a variety of different size memories. Where appropriate, the software will construct these using distributed memory primitives that represent the capabilities of the PFU. Table 2-3 shows the number of Slices required to implement different distributed RAM primitives. Figure 2-6 shows the distributed memory primitive block diagrams. Dual port memories involve the pairing of two Slices. One Slice functions as the read-write port, while the other companion Slice supports the read-only port. For more information on RAM mode in MachXO devices, please see details of additional technical documentation at the end of this data sheet.

Table 2-3. Number of Slices Required For Implementing Distributed RAM

	SPR16x2	DPR16x2
Number of Slices	1	2

Note: SPR = Single Port RAM, DPR = Dual Port RAM

Figure 2-6. Distributed Memory Primitives



ROM Mode: The ROM mode uses the same principal as the RAM modes, but without the Write port. Pre-loading is accomplished through the programming interface during configuration.

PFU Modes of Operation

Slices can be combined within a PFU to form larger functions. Table 2-4 tabulates these modes and documents the functionality possible at the PFU level.

Table 2-4. PFU Modes of Operation

Logic	Ripple	RAM	ROM
LUT 4x8 or MUX 2x1 x 8	2-bit Add x 4	SPR16x2 x 4 DPR16x2 x 2	ROM16x1 x 8
LUT 5x4 or MUX 4x1 x 4	2-bit Sub x 4	SPR16x4 x 2 DPR16x4 x 1	ROM16x2 x 4
LUT 6x 2 or MUX 8x1 x 2	2-bit Counter x 4	SPR16x8 x 1	ROM16x4 x 2
LUT 7x1 or MUX 16x1 x 1	2-bit Comp x 4		ROM16x8 x 1

Routing

There are many resources provided in the MachXO devices to route signals individually or as buses with related control signals. The routing resources consist of switching circuitry, buffers and metal interconnect (routing) segments.

The inter-PFU connections are made with three different types of routing resources: x1 (spans two PFUs), x2 (spans three PFUs) and x6 (spans seven PFUs). The x1, x2, and x6 connections provide fast and efficient connections in the horizontal and vertical directions.

The ispLEVER design tool takes the output of the synthesis tool and places and routes the design. Generally, the place and route tool is completely automatic, although an interactive routing editor is available to optimize the design.

Clock/Control Distribution Network

The MachXO family of devices provides global signals that are available to all PFUs. These signals consist of four primary clocks and four secondary clocks. Primary clock signals are generated from four 16:1 muxes as shown in Figure 2-7 and Figure 2-8. The available clock sources for the MachXO256 and MachXO640 devices are four dual function clock pins and 12 internal routing signals. The available clock sources for the MachXO1200 and MachXO2280 devices are four dual function clock pins, up to nine internal routing signals and up to six PLL outputs.

Figure 2-7. Primary Clocks for MachXO256 and MachXO640 Devices

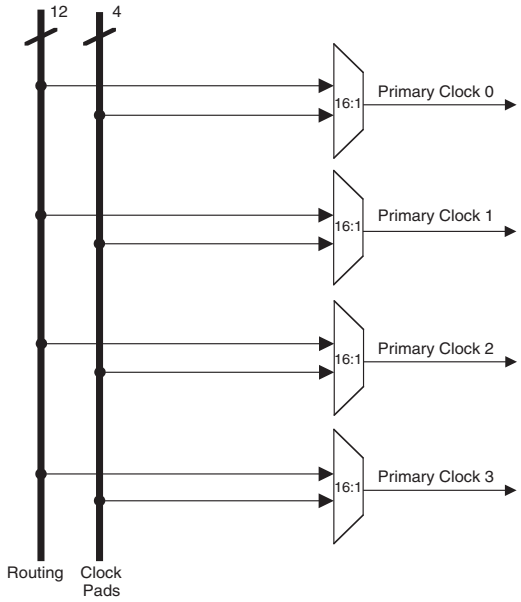
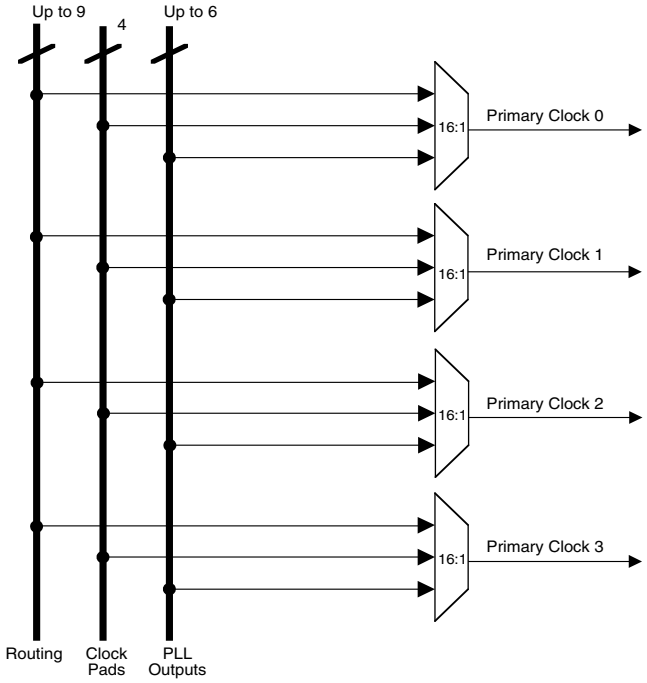
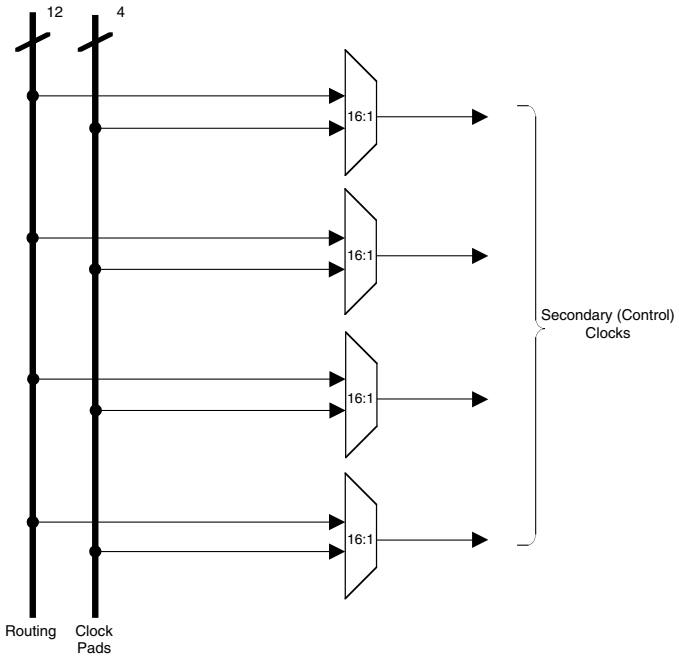


Figure 2-8. Primary Clocks for MachXO1200 and MachXO2280 Devices



Four secondary clocks are generated from four 16:1 muxes as shown in Figure 2-9. Four of the secondary clock sources come from dual function clock pins and 12 come from internal routing.

Figure 2-9. Secondary Clocks for MachXO Devices



sysCLOCK Phase Locked Loops (PLLs)

The MachXO1200 and MachXO2280 provide PLL support. The source of the PLL input divider can come from an external pin or from internal routing. There are four sources of feedback signals to the feedback divider: from CLKINTFB (internal feedback port), from the global clock nets, from the output of the post scalar divider, and from the routing (or from an external pin). There is a PLL_LOCK signal to indicate that the PLL has locked on to the input clock signal. Figure 2-10 shows the sysCLOCK PLL diagram.

The setup and hold times of the device can be improved by programming a delay in the feedback or input path of the PLL which will advance or delay the output clock with reference to the input clock. This delay can be either programmed during configuration or can be adjusted dynamically. In dynamic mode, the PLL may lose lock after adjustment and not relock until the t_{LOCK} parameter has been satisfied. Additionally, the phase and duty cycle block allows the user to adjust the phase and duty cycle of the CLKOS output.

The sysCLOCK PLLs provide the ability to synthesize clock frequencies. Each PLL has four dividers associated with it: input clock divider, feedback divider, post scalar divider, and secondary clock divider. The input clock divider is used to divide the input clock signal, while the feedback divider is used to multiply the input clock signal. The post scalar divider allows the VCO to operate at higher frequencies than the clock output, thereby increasing the frequency range. The secondary divider is used to derive lower frequency outputs.

Figure 2-10. PLL Diagram

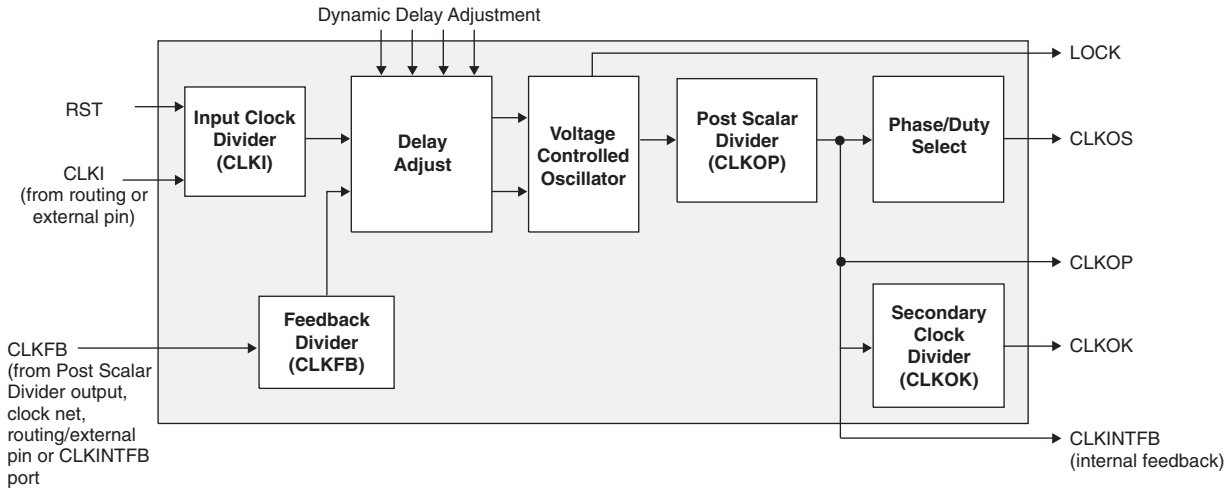


Figure 2-11 shows the available macros for the PLL. Table 2-5 provides signal description of the PLL Block.

Figure 2-11. PLL Primitive

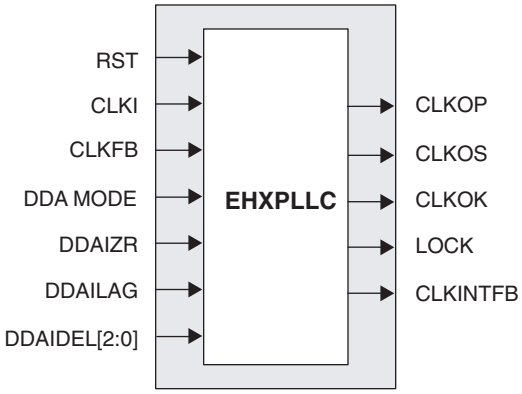


Table 2-5. PLL Signal Descriptions

Signal	I/O	Description
CLKI	I	Clock input from external pin or routing
CLKFB	I	PLL feedback input from PLL output, clock net, routing/external pin or internal feedback from CLKINTFB port
RST	I	"1" to reset the input clock divider
CLKOS	O	PLL output clock to clock tree (phase shifted/duty cycle changed)
CLKOP	O	PLL output clock to clock tree (No phase shift)
CLKOK	O	PLL output to clock tree through secondary clock divider
LOCK	O	"1" indicates PLL LOCK to CLKI
CLKINTFB	O	Internal feedback source, CLKOP divider output before CLOCKTREE
DDAMODE	I	Dynamic Delay Enable. "1": Pin control (dynamic), "0": Fuse Control (static)
DDAIZR	I	Dynamic Delay Zero. "1": delay = 0, "0": delay = on
DDAILAG	I	Dynamic Delay Lag/Lead. "1": Lag, "0": Lead
DDAIDEL[2:0]	I	Dynamic Delay Input

For more information on the PLL, please see details of additional technical documentation at the end of this data sheet.

sysMEM Memory

The MachXO1200 and MachXO2280 devices contain sysMEM Embedded Block RAMs (EBRs). The EBR consists of a 9-Kbit RAM, with dedicated input and output registers.

sysMEM Memory Block

The sysMEM block can implement single port, dual port, pseudo dual port, or FIFO memories. Each block can be used in a variety of depths and widths as shown in Table 2-6.

Table 2-6. sysMEM Block Configurations

Memory Mode	Configurations
Single Port	8,192 x 1 4,096 x 2 2,048 x 4 1,024 x 9 512 x 18 256 x 36
True Dual Port	8,192 x 1 4,096 x 2 2,048 x 4 1,024 x 9 512 x 18
Pseudo Dual Port	8,192 x 1 4,096 x 2 2,048 x 4 1,024 x 9 512 x 18 256 x 36
FIFO	8,192 x 1 4,096 x 2 2,048 x 4 1,024 x 9 512 x 18 256 x 36

Bus Size Matching

All of the multi-port memory modes support different widths on each of the ports. The RAM bits are mapped LSB word 0 to MSB word 0, LSB word 1 to MSB word 1 and so on. Although the word size and number of words for each port varies, this mapping scheme applies to each port.

RAM Initialization and ROM Operation

If desired, the contents of the RAM can be pre-loaded during device configuration. By preloading the RAM block during the chip configuration cycle and disabling the write controls, the sysMEM block can also be utilized as a ROM.

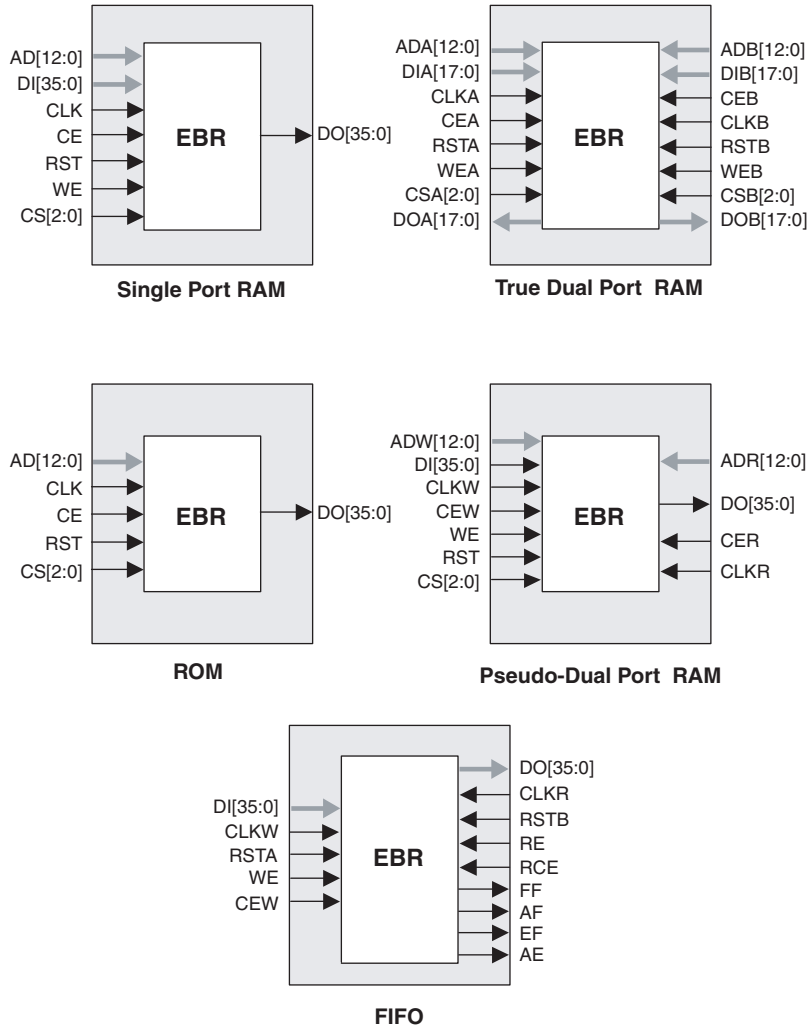
Memory Cascading

Larger and deeper blocks of RAMs can be created using EBR sysMEM Blocks. Typically, the Lattice design tools cascade memory transparently, based on specific design inputs.

Single, Dual, Pseudo-Dual Port and FIFO Modes

Figure 2-12 shows the five basic memory configurations and their input/output names. In all the sysMEM RAM modes, the input data and address for the ports are registered at the input of the memory array. The output data of the memory is optionally registered at the memory array output.

Figure 2-12. sysMEM Memory Primitives



The EBR memory supports three forms of write behavior for single or dual port operation:

1. **Normal** – data on the output appears only during the read cycle. During a write cycle, the data (at the current address) does not appear on the output. This mode is supported for all data widths.

2. **Write Through** – a copy of the input data appears at the output of the same port. This mode is supported for all data widths.

3. **Read-Before-Write** – when new data is being written, the old contents of the address appears at the output. This mode is supported for x9, x18 and x36 data widths.

FIFO Configuration

The FIFO has a write port with Data-in, CEW, WE and CLKW signals. There is a separate read port with Data-out, RCE, RE and CLKR signals. The FIFO internally generates Almost Full, Full, Almost Empty and Empty Flags. The Full and Almost Full flags are registered with CLKW. The Empty and Almost Empty flags are registered with CLKR. The range of programming values for these flags are in Table 2-7.

Table 2-7. Programmable FIFO Flag Ranges

Flag Name	Programming Range
Full (FF)	1 to (up to 2^N-1)
Almost Full (AF)	1 to Full-1
Almost Empty (AE)	1 to Full-1
Empty (EF)	0

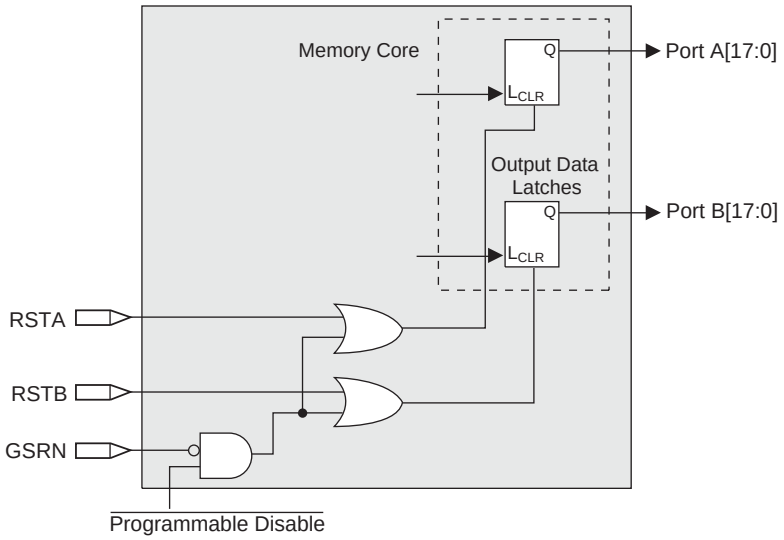
N = Address bit width

The FIFO state machine supports two types of reset signals: RSTA and RSTB. The RSTA signal is a global reset that clears the contents of the FIFO by resetting the read/write pointer and puts the FIFO flags in their initial reset state. The RSTB signal is used to reset the read pointer. The purpose of this reset is to retransmit the data that is in the FIFO. In these applications it is important to keep careful track of when a packet is written into or read from the FIFO.

Memory Core Reset

The memory array in the EBR utilizes latches at the A and B output ports. These latches can be reset asynchronously. RSTA and RSTB are local signals, which reset the output latches associated with Port A and Port B respectively. The Global Reset (GSRN) signal resets both ports. The output data latches and associated resets for both ports are as shown in Figure 2-13.

Figure 2-13. Memory Core Reset

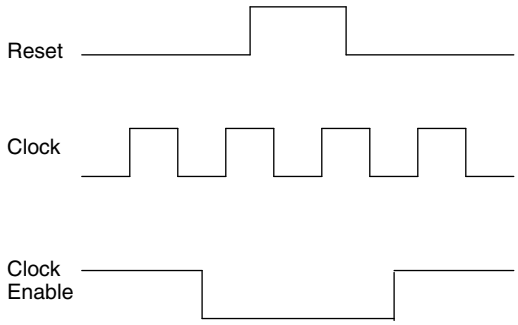


For further information on the sysMEM EBR block, see the details of additional technical documentation at the end of this data sheet.

EBR Asynchronous Reset

EBR asynchronous reset or GSR (if used) can only be applied if all clock enables are low for a clock cycle before the reset is applied and released a clock cycle after the reset is released, as shown in Figure 2-14. The GSR input to the EBR is always asynchronous.

Figure 2-14. EBR Asynchronous Reset (Including GSR) Timing Diagram



If all clock enables remain enabled, the EBR asynchronous reset or GSR may only be applied and released after the EBR read and write clock inputs are in a steady state condition for a minimum of $1/f_{MAX}$ (EBR clock). The reset release must adhere to the EBR synchronous reset setup time before the next active read or write clock edge.

If an EBR is pre-loaded during configuration, the GSR input must be disabled or the release of the GSR during device Wake Up must occur before the release of the device I/Os becoming active.

These instructions apply to all EBR RAM, ROM and FIFO implementations. For the EBR FIFO mode, the GSR signal is always enabled and the WE and RE signals act like the clock enable signals in Figure 2-14. The reset timing rules apply to the RPReset input vs the RE input and the RST input vs. the WE and RE inputs. Both RST and RPReset are always asynchronous EBR inputs.

Note that there are no reset restrictions if the EBR synchronous reset is used and the EBR GSR input is disabled

PIO Groups

On the MachXO devices, PIO cells are assembled into two different types of PIO groups, those with four PIO cells and those with six PIO cells. PIO groups with four IOs are placed on the left and right sides of the device while PIO groups with six IOs are placed on the top and bottom. The individual PIO cells are connected to their respective sysIO buffers and PADs.

On all MachXO devices, two adjacent PIOs can be joined to provide a complementary Output driver pair. The I/O pin pairs are labeled as "T" and "C" to distinguish between the true and complement pins.

The MachXO1200 and MachXO2280 devices contain enhanced I/O capability. All PIO pairs on these larger devices can implement differential receivers. In addition, half of the PIO pairs on the left and right sides of these devices can be configured as LVDS transmit/receive pairs. PIOs on the top of these larger devices also provide PCI support.

Figure 2-15. Group of Four Programmable I/O Cells

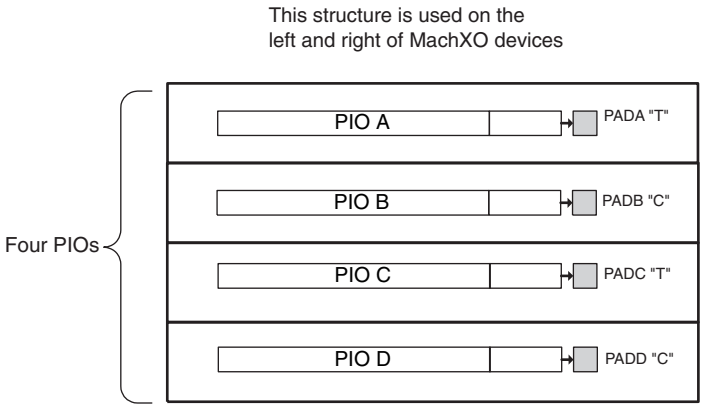
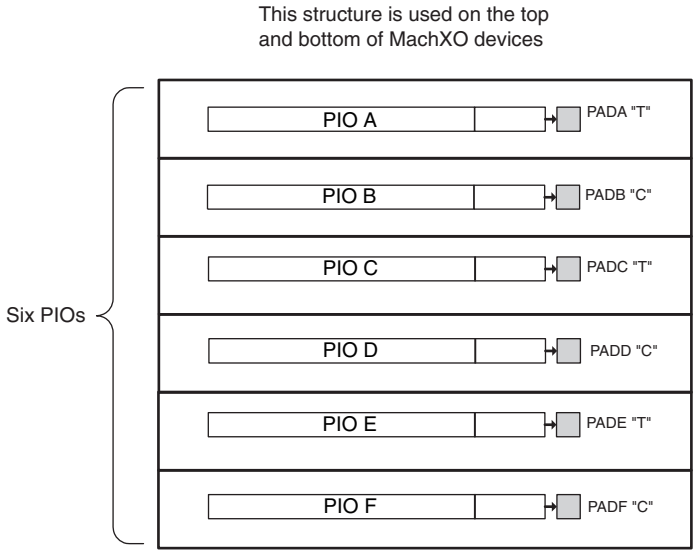


Figure 2-16. Group ofpSix Programmable I/O Cells



PIO

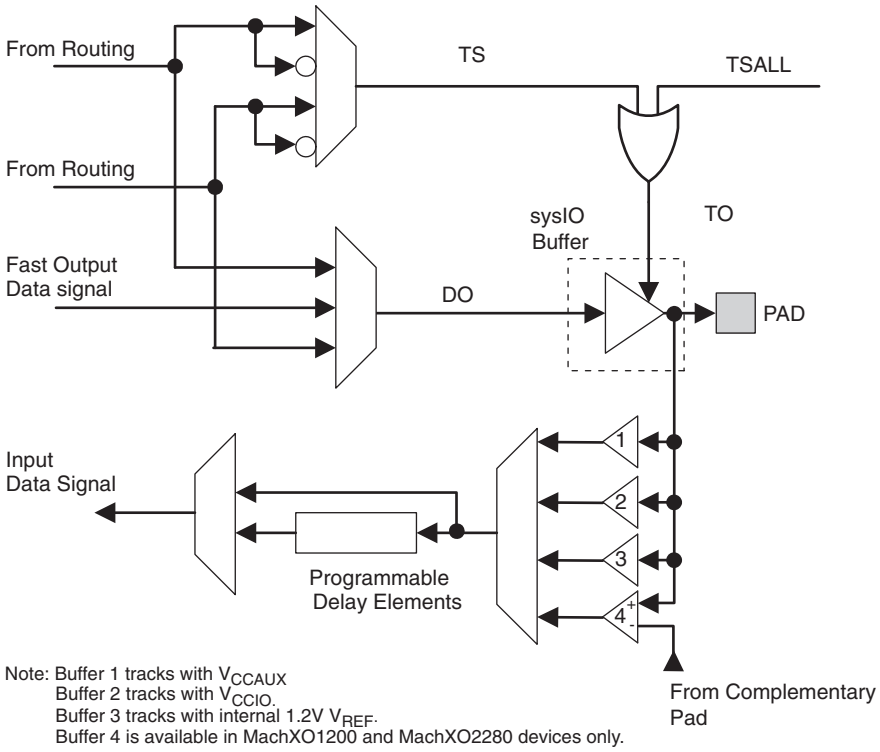
The PIO blocks provide the interface between the sysIO buffers and the internal PFU array blocks. These blocks receive output data from the PFU array and a fast output data signal from adjacent PFUs. The output data and fast

output data signals are multiplexed and provide a single signal to the I/O pin via the sysIO buffer. Figure 2-17 shows the MachXO PIO logic.

The tristate control signal is multiplexed from the output data signals and their complements. In addition a global signal (TSALL) from a dedicated pad can be used to tristate the sysIO buffer.

The PIO receives an input signal from the pin via the sysIO buffer and provides this signal to the core of the device. In addition there are programmable elements that can be utilized by the design tools to avoid positive hold times.

Figure 2-17. MachXO PIO Block Diagram



sysIO Buffer

Each I/O is associated with a flexible buffer referred to as a sysIO buffer. These buffers are arranged around the periphery of the device in groups referred to as Banks. The sysIO buffers allow users to implement the wide variety of standards that are found in today's systems including LVCMOS, TTL, BLVDS, LVDS and LVPECL.

In the MachXO devices, single-ended output buffers and ratioed input buffers (LVTTL, LVCMOS and PCI) are powered using V_{CCIO}. In addition to the Bank V_{CCIO} supplies, the MachXO devices have a V_{CC} core logic power supply, and a V_{CCAUX} supply that powers up a variety of internal circuits including all the differential and referenced input buffers.

MachXO256 and MachXO640 devices contain single-ended input buffers and single-ended output buffers with complementary outputs on all the I/O Banks.

MachXO1200 and MachXO2280 devices contain two types of sysIO buffer pairs.

1. **Top and Bottom sysIO Buffer Pairs**
   The sysIO buffer pairs in the top and bottom Banks of the device consist of two single-ended output drivers and two sets of single-ended input buffers (for ratioed or absolute input levels). The I/O pairs on the top and bottom

of the devices also support differential input buffers. PCI clamps are available on the top Bank I/O buffers. The PCI clamp is enabled after V_{CC}, V_{CCAUX}, and V_{CCIO} are at valid operating levels and the device has been configured.

The two pads in the pair are described as “true” and “comp”, where the true pad is associated with the positive side of the differential input buffer and the comp (complementary) pad is associated with the negative side of the differential input buffer.

2. **Left and Right sysIO Buffer Pairs**
The sysIO buffer pairs in the left and right Banks of the device consist of two single-ended output drivers and two sets of single-ended input buffers (supporting ratioed and absolute input levels). The devices also have a differential driver per output pair. The referenced input buffer can also be configured as a differential input buffer. In these Banks the two pads in the pair are described as “true” and “comp”, where the true pad is associated with the positive side of the differential I/O, and the comp (complementary) pad is associated with the negative side of the differential I/O.

Typical I/O Behavior During Power-up

The internal power-on-reset (POR) signal is deactivated when V_{CC} and V_{CCAUX} have reached satisfactory levels. After the POR signal is deactivated, the FPGA core logic becomes active. It is the user’s responsibility to ensure that all V_{CCIO} Banks are active with valid input logic levels to properly control the output logic states of all the I/O Banks that are critical to the application. The default configuration of the I/O pins in a blank device is tri-state with a weak pull-up to VCCIO. The I/O pins will maintain the blank configuration until VCC, VCCAUX and VCCIO have reached satisfactory levels at which time the I/Os will take on the user-configured settings.

The V_{CC} and V_{CCAUX} supply the power to the FPGA core fabric, whereas the V_{CCIO} supplies power to the I/O buffers. In order to simplify system design while providing consistent and predictable I/O behavior, the I/O buffers should be powered up along with the FPGA core fabric. Therefore, V_{CCIO} supplies should be powered up before or together with the V_{CC} and V_{CCAUX} supplies

Supported Standards

The MachXO sysIO buffer supports both single-ended and differential standards. Single-ended standards can be further subdivided into LVCMOS and LVTTL. The buffer supports the LVTTL, LVCMOS 1.2, 1.5, 1.8, 2.5, and 3.3V standards. In the LVCMOS and LVTTL modes, the buffer has individually configurable options for drive strength, bus maintenance (weak pull-up, weak pull-down, bus-keeper latch or none) and open drain. BLVDS and LVPECL output emulation is supported on all devices. The MachXO1200 and MachXO2280 support on-chip LVDS output buffers on approximately 50% of the I/Os on the left and right Banks. Differential receivers for LVDS, BLVDS and LVPECL are supported on all Banks of MachXO1200 and MachXO2280 devices. PCI support is provided in the top Banks of the MachXO1200 and MachXO2280 devices. Table 2-8 summarizes the I/O characteristics of the devices in the MachXO family.

Tables 2-9 and 2-10 show the I/O standards (together with their supply and reference voltages) supported by the MachXO devices. For further information on utilizing the sysIO buffer to support a variety of standards please see the details of additional technical documentation at the end of this data sheet.

Table 2-8. I/O Support Device by Device

	MachXO256	MachXO640	MachXO1200	MachXO2280
Number of I/O Banks	2	4	8	8
Type of Input Buffers	Single-ended (all I/O Banks)	Single-ended (all I/O Banks)	Single-ended (all I/O Banks) Differential Receivers (all I/O Banks)	Single-ended (all I/O Banks) Differential Receivers (all I/O Banks)
Types of Output Buffers	Single-ended buffers with complementary outputs (all I/O Banks)	Single-ended buffers with complementary outputs (all I/O Banks)	Single-ended buffers with complementary outputs (all I/O Banks) Differential buffers with true LVDS outputs (50% on left and right side)	Single-ended buffers with complementary outputs (all I/O Banks) Differential buffers with true LVDS outputs (50% on left and right side)
Differential Output Emulation Capability	All I/O Banks	All I/O Banks	All I/O Banks	All I/O Banks
PCI Support	No	No	Top side only	Top side only

Table 2-9. Supported Input Standards

	VCCIO (Typ.)				
Input Standard	**3.3V**	**2.5V**	**1.8V**	**1.5V**	**1.2V**
Single Ended Interfaces					
LVTTL	Yes	Yes	Yes	Yes	Yes
LVCMOS33	Yes	Yes	Yes	Yes	Yes
LVCMOS25	Yes	Yes	Yes	Yes	Yes
LVCMOS18			Yes		
LVCMOS15				Yes	
LVCMOS12	Yes	Yes	Yes	Yes	Yes
PCI[1]	Yes				
Differential Interfaces					
BLVDS[2], LVDS[2], LVPECL[2], RSDS[2]	Yes	Yes	Yes	Yes	Yes

1. Top Banks of MachXO1200 and MachXO2280 devices only.
2. MachXO1200 and MachXO2280 devices only.

Table 2-10. Supported Output Standards

Output Standard	Drive	V_{CCIO} (Typ.)
Single-ended Interfaces		
LVTTL	4mA, 8mA, 12mA, 16mA	3.3
LVCMOS33	4mA, 8mA, 12mA, 14mA	3.3
LVCMOS25	4mA, 8mA, 12mA, 14mA	2.5
LVCMOS18	4mA, 8mA, 12mA, 14mA	1.8
LVCMOS15	4mA, 8mA	1.5
LVCMOS12	2mA, 6mA	1.2
LVCMOS33, Open Drain	4mA, 8mA, 12mA, 14mA	—
LVCMOS25, Open Drain	4mA, 8mA, 12mA, 14mA	—
LVCMOS18, Open Drain	4mA, 8mA, 12mA, 14mA	—
LVCMOS15, Open Drain	4mA, 8mA	—
LVCMOS12, Open Drain	2mA, 6mA	—
PCI33[3]	N/A	3.3
Differential Interfaces		
LVDS[1, 2]	N/A	2.5
BLVDS, RSDS[2]	N/A	2.5
LVPECL[2]	N/A	3.3

1. MachXO1200 and MachXO2280 devices have dedicated LVDS buffers.
2. These interfaces can be emulated with external resistors in all devices.
3. Top Banks of MachXO1200 and MachXO2280 devices only.

sysIO Buffer Banks

The number of Banks vary between the devices of this family. Eight Banks surround the two larger devices, the MachXO1200 and MachXO2280 (two Banks per side). The MachXO640 has four Banks (one Bank per side). The smallest member of this family, the MachXO256, has only two Banks.

Each sysIO buffer Bank is capable of supporting multiple I/O standards. Each Bank has its own I/O supply voltage (V_{CCIO}) which allows it to be completely independent from the other Banks. Figure 2-18, Figure 2-18, Figure 2-20 and Figure 2-21 shows the sysIO Banks and their associated supplies for all devices.

Figure 2-18. MachXO2280 Banks

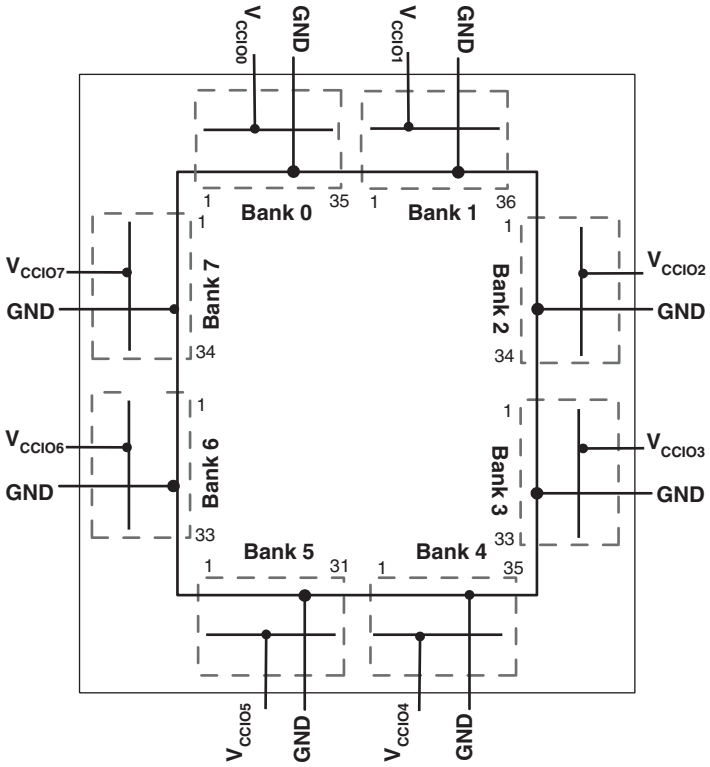


Figure 2-19. MachXO1200 Banks

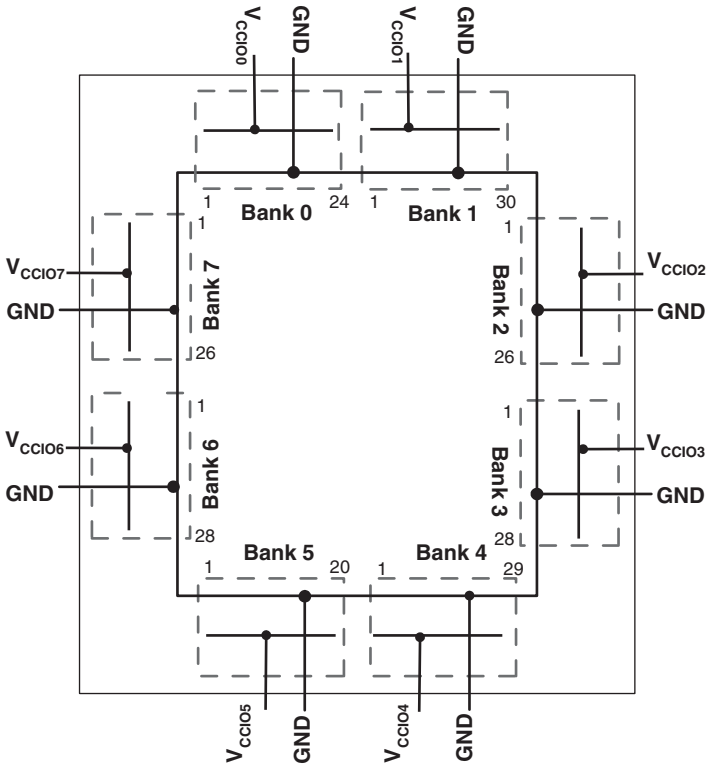


Figure 2-20. MachXO640 Banks

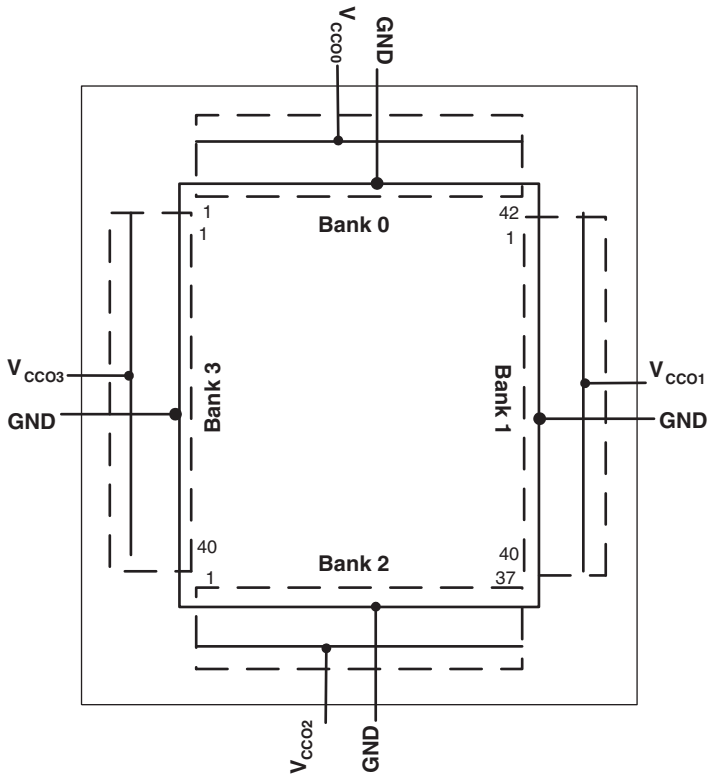
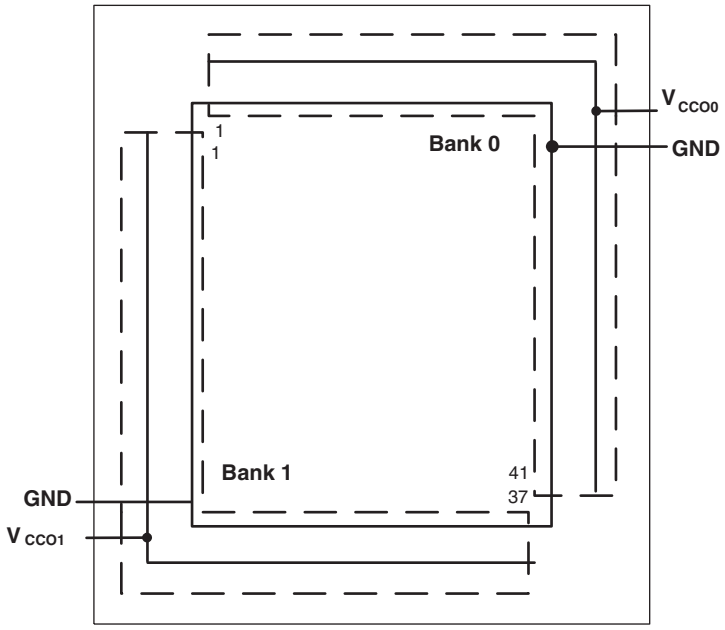


Figure 2-21. MachXO256 Banks



Hot Socketing

The MachXO devices have been carefully designed to ensure predictable behavior during power-up and power-down. Leakage into I/O pins is controlled to within specified limits. This allows for easy integration with the rest of

the system. These capabilities make the MachXO ideal for many multiple power supply and hot-swap applications.

Sleep Mode

The MachXO "C" devices (V_{CC} = 1.8/2.5/3.3V) have a sleep mode that allows standby current to be reduced dramatically during periods of system inactivity. Entry and exit to Sleep mode is controlled by the SLEEPN pin.

During Sleep mode, the logic is non-operational, registers and EBR contents are not maintained, and I/Os are tri-stated. Do not enter Sleep mode during device programming or configuration operation. In Sleep mode, power supplies are in their normal operating range, eliminating the need for external switching of power supplies. Table 2-11 compares the characteristics of Normal, Off and Sleep modes.

Table 2-11. Characteristics of Normal, Off and Sleep Modes

Characteristic	Normal	Off	Sleep
SLEEPN Pin	High	—	Low
Static Icc	Typical <10mA	0	Typical <100uA
I/O Leakage	<10µA	<1mA	<10µA
Power Supplies VCC/VCCIO/VCCAUX	Normal Range	0	Normal Range
Logic Operation	User Defined	Non Operational	Non operational
I/O Operation	User Defined	Tri-state	Tri-state
JTAG and Programming circuitry	Operational	Non-operational	Non-operational
EBR Contents and Registers	Maintained	Non-maintained	Non-maintained

SLEEPN Pin Characteristics

The SLEEPN pin behaves as an LVCMOS input with the voltage standard appropriate to the VCC supply for the device. This pin also has a weak pull-up, along with a Schmidt trigger and glitch filter to prevent false triggering. An external pull-up to VCC is recommended when Sleep Mode is not used to ensure the device stays in normal operation mode. Typically, the device enters sleep mode several hundred nanoseconds after SLEEPN is held at a valid low and restarts normal operation as specified in the Sleep Mode Timing table. The AC and DC specifications portion of this data sheet shows a detailed timing diagram.

Oscillator

Every MachXO device has an internal CMOS oscillator. The oscillator can be routed as an input clock to the clock tree or to general routing resources. The oscillator frequency can be divided by internal logic. There is a dedicated programming bit to enable/disable the oscillator. The oscillator frequency ranges from 18MHz to 26MHz.

Configuration and Testing

The following section describes the configuration and testing features of the MachXO family of devices.

IEEE 1149.1-Compliant Boundary Scan Testability

All MachXO devices have boundary scan cells that are accessed through an IEEE 1149.1 compliant test access port (TAP). This allows functional testing of the circuit board, on which the device is mounted, through a serial scan path that can access all critical logic nodes. Internal registers are linked internally, allowing test data to be shifted in and loaded directly onto test nodes, or test data to be captured and shifted out for verification. The test access port consists of dedicated I/Os: TDI, TDO, TCK and TMS. The test access port shares its power supply with one of the VCCIO Banks (MachXO256: V_{CCIO1}; MachXO640: V_{CCIO2}; MachXO1200 and MachXO2280: V_{CCIO5}) and can operate with LVCMOS3.3, 2.5, 1.8, 1.5, and 1.2 standards.

For more details on boundary scan test, please see information regarding additional technical documentation at the end of this data sheet.

Device Configuration

All MachXO devices contain a test access port that can be used for device configuration and programming.

The non-volatile memory in the MachXO can be configured in two different modes:

- In IEEE 1532 mode via the IEEE 1149.1 port. In this mode, the device is off-line and I/Os are controlled by BSCAN registers.
- In background mode via the IEEE 1149.1 port. This allows the device to remain operational in user mode while reprogramming takes place.

The SRAM configuration memory can be configured in three different ways:

- At power-up via the on-chip non-volatile memory.
- After a refresh command is issued via the IEEE 1149.1 port.
- In IEEE 1532 mode via the IEEE 1149.1 port.

Figure 2-22 provides a pictorial representation of the different programming modes available in the MachXO devices. On power-up, the SRAM is ready to be configured with IEEE 1149.1 serial TAP port using IEEE 1532 protocols.

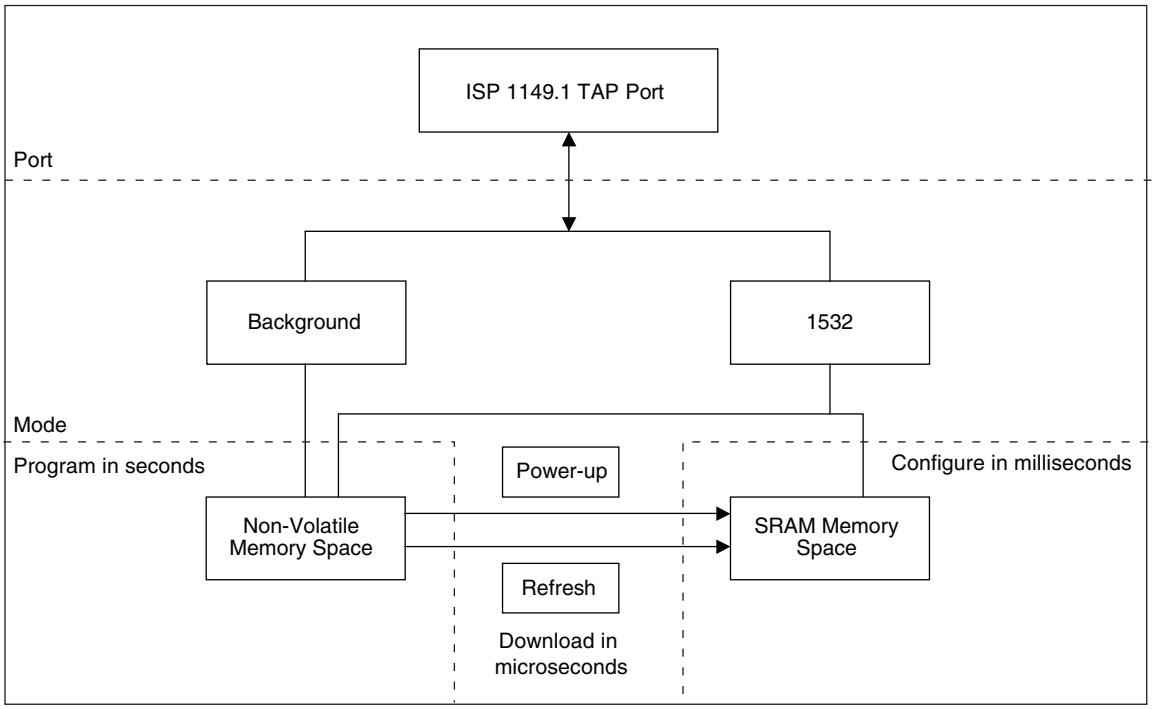
Leave Alone I/O
When using IEEE 1532 mode for non-volatile memory programming, SRAM configuration, or issuing a refresh command, users may specify I/Os as high, low, tristated or held at current value. This provides excellent flexibility for implementing systems where reconfiguration or reprogramming occurs on-the-fly.

TransFR (Transparent Field Reconfiguration)
TransFR (TFR) is a unique Lattice technology that allows users to update their logic in the field without interrupting system operation using a single ispVM command. See TN1087, Minimizing System Interruption During Configuration Using TransFR Technology for details.

Security
The MachXO devices contain security bits that, when set, prevent the readback of the SRAM configuration and non-volatile memory spaces. Once set, the only way to clear the security bits is to erase the memory space.

For more information on device configuration, please see details of additional technical documentation at the end of this data sheet.

Figure 2-22. MachXO Configuration and Programming



Density Shifting

The MachXO family has been designed to enable density migration in the same package. Furthermore, the architecture ensures a high success rate when performing design migration from lower density parts to higher density parts. In many cases, it is also possible to shift a lower utilization design targeted for a high-density device to a lower density device. However, the exact details of the final resource utilization will impact the likely success in each case.

MachXO Family Data Sheet DC and Switching Characteristics

July 2010 Data Sheet DS1002

Absolute Maximum Ratings[1, 2, 3]

	LCMXO E (1.2V)	LCMXO C (1.8V/2.5V/3.3V)
Supply Voltage V_{CC}	-0.5 to 1.32V	-0.5 to 3.75V
Supply Voltage V_{CCAUX}	-0.5 to 3.75V	-0.5 to 3.75V
Output Supply Voltage V_{CCIO}	-0.5 to 3.75V	-0.5 to 3.75V
I/O Tristate Voltage Applied [4]	-0.5 to 3.75V	-0.5 to 3.75V
Dedicated Input Voltage Applied[4]	-0.5 to 3.75V	-0.5 to 4.25V
Storage Temperature (ambient)	-65 to 150°C	-65 to 150°C
Junction Temp. (Tj)	+125°C	+125°C

1. Stress above those listed under the "Absolute Maximum Ratings" may cause permanent damage to the device. Functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.
2. Compliance with the Lattice *Thermal Management* document is required.
3. All voltages referenced to GND.
4. Overshoot and undershoot of -2V to (V_{IHMAX} + 2) volts is permitted for a duration of <20ns.

Recommended Operating Conditions[1]

Symbol	Parameter	Min.	Max.	Units
V_{CC}	Core Supply Voltage for 1.2V Devices	1.14	1.26	V
	Core Supply Voltage for 1.8V/2.5V/3.3V Devices	1.71	3.465	V
V_{CCAUX}[3]	Auxiliary Supply Voltage	3.135	3.465	V
V_{CCIO}[2]	I/O Driver Supply Voltage	1.14	3.465	V
t_{JCOM}	Junction Temperature Commercial Operation	0	+85	°C
t_{JIND}	Junction Temperature Industrial Operation	-40	100	°C
$t_{JFLASHCOM}$	Junction Temperature, Flash Programming, Commercial	0	+85	°C
$t_{JFLASHIND}$	Junction Temperature, Flash Programming, Industrial	-40	100	°C

1. Like power supplies must be tied together. For example, if V_{CCIO} and V_{CC} are both 2.5V, they must also be the same supply. 3.3V V_{CCIO} and 1.2V V_{CCIO} should be tied to V_{CCAUX} or 1.2V V_{CC} respectively.
2. See recommended voltages by I/O standard in subsequent table.
3. V_{CC} must reach minimum V_{CC} value before V_{CCAUX} reaches 2.5V.

MachXO256 and MachXO640 Hot Socketing Specifications[1, 2, 3]

Symbol	Parameter	Condition	Min.	Typ.	Max	Units
I_{DK}	Input or I/O leakage Current	$0 \le V_{IN} \le V_{IH}$ (MAX)	—	—	+/-1000	µA

1. Insensitive to sequence of V_{CC}, V_{CCAUX}, and V_{CCIO}. However, assumes monotonic rise/fall rates for V_{CC}, V_{CCAUX}, and V_{CCIO}.
2. $0 \le V_{CC} \le V_{CC}$ (MAX), $0 \le V_{CCIO} \le V_{CCIO}$ (MAX) and $0 \le V_{CCAUX} \le V_{CCAUX}$ (MAX).
3. I_{DK} is additive to I_{PU}, I_{PD} or I_{BH}.

MachXO1200 and MachXO2280 Hot Socketing Specifications[1, 2, 3, 4]

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
Non-LVDS General Purpose sysIOs						
I_{DK}	Input or I/O Leakage Current	$0 \le V_{IN} \le V_{IH}$ (MAX.)	—	—	+/-1000	µA
LVDS General Purpose sysIOs						
I_{DK_LVDS}	Input or I/O Leakage Current	$V_{IN} \le V_{CCIO}$	—	—	+/-1000	µA
		$V_{IN} > V_{CCIO}$	—	35	—	mA

1. Insensitive to sequence of V_{CC}, V_{CCAUX}, and V_{CCIO}. However, assumes monotonic rise/fall rates for V_{CC}, V_{CCAUX}, and V_{CCIO}.
2. $0 \le V_{CC} \le V_{CC}$ (MAX), $0 \le V_{CCIO} \le V_{CCIO}$ (MAX), and $0 \le V_{CCAUX} \le V_{CCAUX}$ (MAX).
3. I_{DK} is additive to I_{PU}, I_{PW} or I_{BH}.
4. LVCMOS and LVTTL only.

DC Electrical Characteristics

Over Recommended Operating Conditions

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
I_{IL}, I_{IH}[1, 4, 5]	Input or I/O Leakage	$0 \le V_{IN} \le (V_{CCIO} - 0.2V)$	—	—	10	µA
		$(V_{CCIO} - 0.2V) < V_{IN} \le 3.6V$	—	—	40	µA
I_{PU}	I/O Active Pull-up Current	$0 \le V_{IN} \le 0.7\ V_{CCIO}$	-30	—	-150	µA
I_{PD}	I/O Active Pull-down Current	V_{IL} (MAX) $\le V_{IN} \le V_{IH}$ (MAX)	30	—	150	µA
I_{BHLS}	Bus Hold Low sustaining current	$V_{IN} = V_{IL}$ (MAX)	30	—	—	µA
I_{BHHS}	Bus Hold High sustaining current	$V_{IN} = 0.7V_{CCIO}$	-30	—	—	µA
I_{BHLO}	Bus Hold Low Overdrive current	$0 \le V_{IN} \le V_{IH}$ (MAX)	—	—	150	µA
I_{BHHO}	Bus Hold High Overdrive current	$0 \le V_{IN} \le V_{IH}$ (MAX)	—	—	-150	µA
V_{BHT}[3]	Bus Hold trip Points	$0 \le V_{IN} \le V_{IH}$ (MAX)	V_{IL} (MAX)	—	V_{IH} (MIN)	V
C1	I/O Capacitance[2]	V_{CCIO} = 3.3V, 2.5V, 1.8V, 1.5V, 1.2V, V_{CC} = Typ., V_{IO} = 0 to V_{IH} (MAX)	—	8	—	pf
C2	Dedicated Input Capacitance[2]	V_{CCIO} = 3.3V, 2.5V, 1.8V, 1.5V, 1.2V, V_{CC} = Typ., V_{IO} = 0 to V_{IH} (MAX)	—	8	—	pf

1. Input or I/O leakage current is measured with the pin configured as an input or as an I/O with the output driver tri-stated. It is not measured with the output driver active. Bus maintenance circuits are disabled.
2. T_A 25°C, f = 1.0MHz
3. Please refer to V_{IL} and V_{IH} in the sysIO Single-Ended DC Electrical Characteristics table of this document.
4. Not applicable to SLEEPN pin.
5. When V_{IH} is higher than V_{CCIO}, a transient current typically of 30ns in duration or less with a peak current of 6mA can occur on the high-to-low transition. For MachXO1200 and MachXO2280 true LVDS output pins, V_{IH} must be less than or equal to V_{CCIO}.

Supply Current (Sleep Mode)[1, 2]

Symbol	Parameter	Device	Typ.[3]	Max.	Units
I_{CC}	Core Power Supply	LCMXO256C	12	25	µA
		LCMXO640C	12	25	µA
		LCMXO1200C	12	25	µA
		LCMXO2280C	12	25	µA
I_{CCAUX}	Auxiliary Power Supply	LCMXO256C	1	15	µA
		LCMXO640C	1	25	µA
		LCMXO1200C	1	45	µA
		LCMXO2280C	1	85	µA
I_{CCIO}	Bank Power Supply[4]	All LCMXO 'C' Devices	2	30	µA

1. Assumes all inputs are configured as LVCMOS and held at the VCCIO or GND.
2. Frequency = 0MHz.
3. T_A = 25°C, power supplies at nominal voltage.
4. Per Bank.

Supply Current (Standby)[1, 2, 3, 4]

Over Recommended Operating Conditions

Symbol	Parameter	Device	Typ.[5]	Units
I_{CC}	Core Power Supply	LCMXO256C	7	mA
		LCMXO640C	9	mA
		LCMXO1200C	14	mA
		LCMXO2280C	20	mA
		LCMXO256E	4	mA
		LCMXO640E	6	mA
		LCMXO1200E	10	mA
		LCMXO2280E	12	mA
I_{CCAUX}	Auxiliary Power Supply V_{CCAUX} = 3.3V	LCMXO256E/C	5	mA
		LCMXO640E/C	7	mA
		LCMXO1200E/C	12	mA
		LCMXO2280E/C	13	mA
I_{CCIO}	Bank Power Supply[6]	All devices	2	mA

1. For further information on supply current, please see details of additional technical documentation at the end of this data sheet.
2. Assumes all outputs are tristated, all inputs are configured as LVCMOS and held at V_{CCIO} or GND.
3. Frequency = 0MHz.
4. User pattern = blank.
5. T_J = 25ºC, power supplies at nominal voltage.
6. Per Bank. V_{CCIO} = 2.5V. Does not include pull-up/pull-down.

Initialization Supply Current[1, 2, 3, 4]

Over Recommended Operating Conditions

Symbol	Parameter	Device	Typ.[5]	Units
I_{CC}	Core Power Supply	LCMXO256C	13	mA
		LCMXO640C	17	mA
		LCMXO1200C	21	mA
		LCMXO2280C	23	mA
		LCMXO256E	10	mA
		LCMXO640E	14	mA
		LCMXO1200E	18	mA
		LCMXO2280E	20	mA
I_{CCAUX}	Auxiliary Power Supply V_{CCAUX} = 3.3V	LCMXO256E/C	10	mA
		LCMXO640E/C	13	mA
		LCMXO1200E/C	24	mA
		LCMXO2280E/C	25	mA
I_{CCIO}	Bank Power Supply[6]	All devices	2	mA

1. For further information on supply current, please see details of additional technical documentation at the end of this data sheet.
2. Assumes all I/O pins are held at V_{CCIO} or GND.
3. Frequency = 0MHz.
4. Typical user pattern.
5. T_J = 25°C, power supplies at nominal voltage.
6. Per Bank, V_{CCIO} = 2.5V. Does not include pull-up/pull-down.

Programming and Erase Flash Supply Current[1, 2, 3, 4]

Symbol	Parameter	Device	Typ.[5]	Units
I_{CC}	Core Power Supply	LCMXO256C	9	mA
		LCMXO640C	11	mA
		LCMXO1200C	16	mA
		LCMXO2280C	22	mA
		LCMXO256E	6	mA
		LCMXO640E	8	mA
		LCMXO1200E	12	mA
		LCMXO2280E	14	mA
I_{CCAUX}	Auxiliary Power Supply V_{CCAUX} = 3.3V	LCMXO256C/E	8	mA
		LCMXO640C/E	10	mA
		LCMXO1200/E	15	mA
		LCMXO2280C/E	16	mA
I_{CCIO}	Bank Power Supply[6]	All devices	2	mA

1. For further information on supply current, please see details of additional technical documentation at the end of this data sheet.
2. Assumes all I/O pins are held at V_{CCIO} or GND.
3. Typical user pattern.
4. JTAG programming is at 25MHz.
5. T_J = 25°C, power supplies at nominal voltage.
6. Per Bank. V_{CCIO} = 2.5V. Does not include pull-up/pull-down.

sysIO Recommended Operating Conditions

Standard	V_{CCIO} (V)		
	Min.	Typ.	Max.
LVCMOS 3.3	3.135	3.3	3.465
LVCMOS 2.5	2.375	2.5	2.625
LVCMOS 1.8	1.71	1.8	1.89
LVCMOS 1.5	1.425	1.5	1.575
LVCMOS 1.2	1.14	1.2	1.26
LVTTL	3.135	3.3	3.465
PCI[3]	3.135	3.3	3.465
LVDS[1, 2]	2.375	2.5	2.625
LVPECL[1]	3.135	3.3	3.465
BLVDS[1]	2.375	2.5	2.625
RSDS[1]	2.375	2.5	2.625

1. Inputs on chip. Outputs are implemented with the addition of external resistors.
2. MachXO1200 and MachXO2280 devices have dedicated LVDS buffers
3. Input on the top bank of the MachXO1200 and MachXO2280 only.

sysIO Single-Ended DC Electrical Characteristics

Input/Output Standard	V_{IL}		V_{IH}		V_{OL} Max. (V)	V_{OH} Min. (V)	I_{OL}[1] (mA)	I_{OH}[1] (mA)
	Min. (V)	Max. (V)	Min. (V)	Max. (V)				
LVCMOS 3.3	-0.3	0.8	2.0	3.6	0.4	V_{CCIO} - 0.4	16, 12, 8, 4	-14, -12, -8, -4
					0.2	V_{CCIO} - 0.2	0.1	-0.1
LVTTL	-0.3	0.8	2.0	3.6	0.4	2.4	16	-16
					0.4	V_{CCIO} - 0.4	12, 8, 4	-12, -8, -4
					0.2	V_{CCIO} - 0.2	0.1	-0.1
LVCMOS 2.5	-0.3	0.7	1.7	3.6	0.4	V_{CCIO} - 0.4	16, 12, 8, 4	-14, -12, -8, -4
					0.2	V_{CCIO} - 0.2	0.1	-0.1
LVCMOS 1.8	-0.3	$0.35V_{CCIO}$	$0.65V_{CCIO}$	3.6	0.4	V_{CCIO} - 0.4	16, 12, 8, 4	-14, -12, -8, -4
					0.2	V_{CCIO} - 0.2	0.1	-0.1
LVCMOS 1.5	-0.3	$0.35V_{CCIO}$	$0.65V_{CCIO}$	3.6	0.4	V_{CCIO} - 0.4	8, 4	-8, -4
					0.2	V_{CCIO} - 0.2	0.1	-0.1
LVCMOS 1.2 (“C” Version)	-0.3	0.42	0.78	3.6	0.4	V_{CCIO} - 0.4	6, 2	-6, -2
					0.2	V_{CCIO} - 0.2	0.1	-0.1
LVCMOS 1.2 (“E” Version)	-0.3	$0.35V_{CC}$	$0.65V_{CC}$	3.6	0.4	V_{CCIO} - 0.4	6, 2	-6, -2
					0.2	V_{CCIO} - 0.2	0.1	-0.1
PCI	-0.3	$0.3V_{CCIO}$	$0.5V_{CCIO}$	3.6	$0.1V_{CCIO}$	$0.9V_{CCIO}$	1.5	-0.5

1. The average DC current drawn by I/Os between GND connections, or between the last GND in an I/O Bank and the end of an I/O Bank, as shown in the logic signal connections table shall not exceed n * 8mA. Where n is the number of I/Os between Bank GND connections or between the last GND in a Bank and the end of a Bank.

sysIO Differential Electrical Characteristics

LVDS

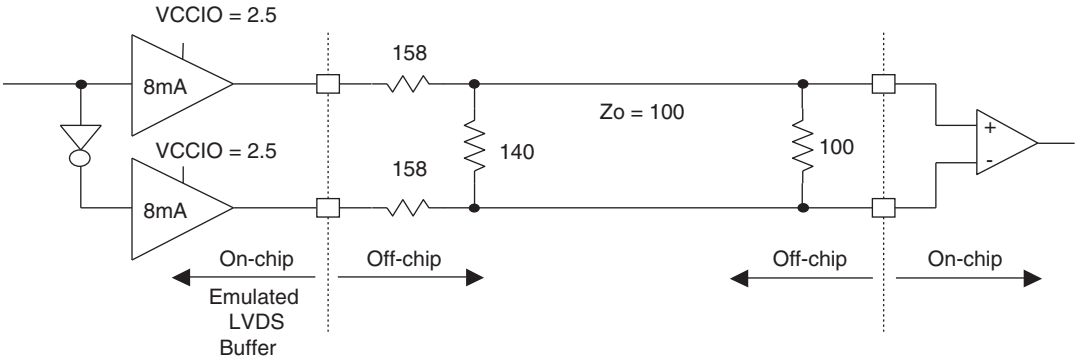
Over Recommended Operating Conditions

Parameter Symbol	Parameter Description	Test Conditions	Min.	Typ.	Max.	Units
V_{INP}, V_{INM}	Input Voltage		0	—	2.4	V
V_{THD}	Differential Input Threshold		+/-100	—	—	mV
V_{CM}	Input Common Mode Voltage	100mV ≤ V_{THD}	V_{THD}/2	1.2	1.8	V
		200mV ≤ V_{THD}	V_{THD}/2	1.2	1.9	V
		350mV ≤ V_{THD}	V_{THD}/2	1.2	2.0	V
I_{IN}	Input current	Power on	—	—	+/-10	µA
V_{OH}	Output high voltage for V_{OP} or V_{OM}	R_T = 100 Ohm	—	1.38	1.60	V
V_{OL}	Output low voltage for V_{OP} or V_{OM}	R_T = 100 Ohm	0.9V	1.03	—	V
V_{OD}	Output voltage differential	(V_{OP} - V_{OM}), R_T = 100 Ohm	250	350	450	mV
ΔV_{OD}	Change in V_{OD} between high and low		—	—	50	mV
V_{OS}	Output voltage offset	(V_{OP} - V_{OM})/2, R_T = 100 Ohm	1.125	1.25	1.375	V
ΔV_{OS}	Change in V_{OS} between H and L		—	—	50	mV
I_{OSD}	Output short circuit current	V_{OD} = 0V Driver outputs shorted	—	—	6	mA

LVDS Emulation

MachXO devices can support LVDS outputs via emulation (LVDS25E), in addition to the LVDS support that is available on-chip on certain devices. The output is emulated using complementary LVCMOS outputs in conjunction with resistors across the driver outputs on all devices. The scheme shown in Figure 3-1 is one possible solution for LVDS standard implementation. Resistor values in Figure 3-1 are industry standard values for 1% resistors.

Figure 3-1. LVDS Using External Resistors (LVDS25E)



Note: All resistors are ±1%.

The LVDS differential input buffers are available on certain devices in the MachXO family.

Table 3-1. LVDS DC Conditions

Over Recommended Operating Conditions

Parameter	Description	Typical	Units
Z_{OUT}	Output impedance	20	Ω
R_S	Driver series resistor	294	Ω
R_P	Driver parallel resistor	121	Ω
R_T	Receiver termination	100	Ω
V_{OH}	Output high voltage	1.43	V
V_{OL}	Output low voltage	1.07	V
V_{OD}	Output differential voltage	0.35	V
V_{CM}	Output common mode voltage	1.25	V
Z_{BACK}	Back impedance	100	Ω
I_{DC}	DC output current	3.66	mA

BLVDS

The MachXO family supports the BLVDS standard through emulation. The output is emulated using complementary LVCMOS outputs in conjunction with a parallel external resistor across the driver outputs. The input standard is supported by the LVDS differential input buffer on certain devices. BLVDS is intended for use when multi-drop and bi-directional multi-point differential signaling is required. The scheme shown in Figure 3-2 is one possible solution for bi-directional multi-point differential signals.

Figure 3-2. BLVDS Multi-point Output Example

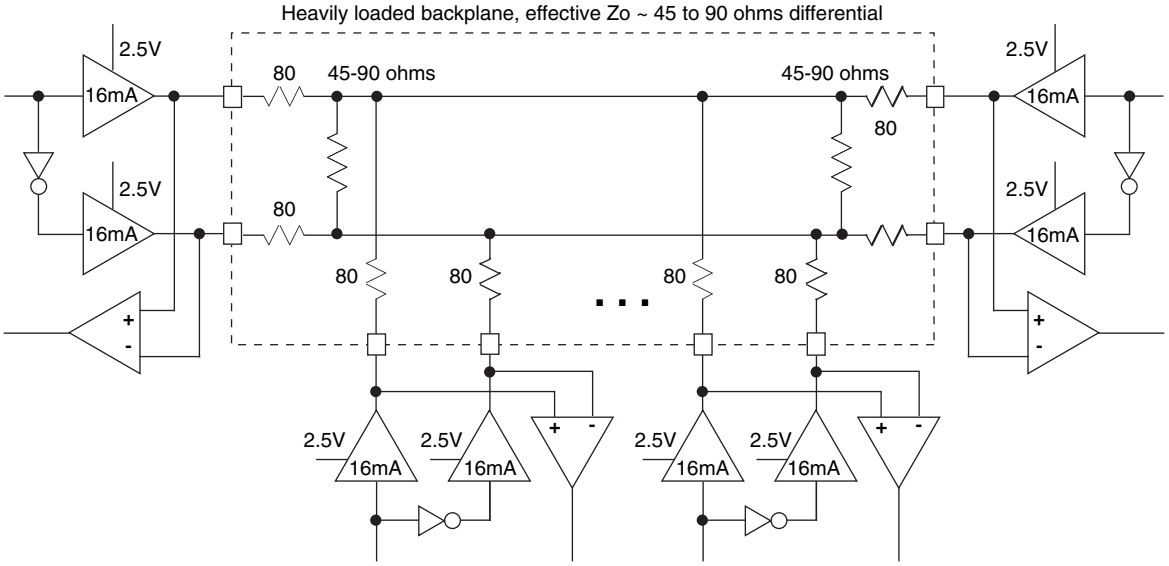


Table 3-2. BLVDS DC Conditions[1]

Over Recommended Operating Conditions

Symbol	Description	Nominal		Units
		Zo = 45	Zo = 90	
Z_{OUT}	Output impedance	100	100	Ohms
R_{TLEFT}	Left end termination	45	90	Ohms
R_{TRIGHT}	Right end termination	45	90	Ohms
V_{OH}	Output high voltage	1.375	1.48	V
V_{OL}	Output low voltage	1.125	1.02	V
V_{OD}	Output differential voltage	0.25	0.46	V
V_{CM}	Output common mode voltage	1.25	1.25	V
I_{DC}	DC output current	11.2	10.2	mA

1. For input buffer, see LVDS table.

LVPECL

The MachXO family supports the differential LVPECL standard through emulation. This output standard is emulated using complementary LVCMOS outputs in conjunction with a parallel resistor across the driver outputs on all the devices. The LVPECL input standard is supported by the LVDS differential input buffer on certain devices. The scheme shown in Figure 3-3 is one possible solution for point-to-point signals.

Figure 3-3. Differential LVPECL

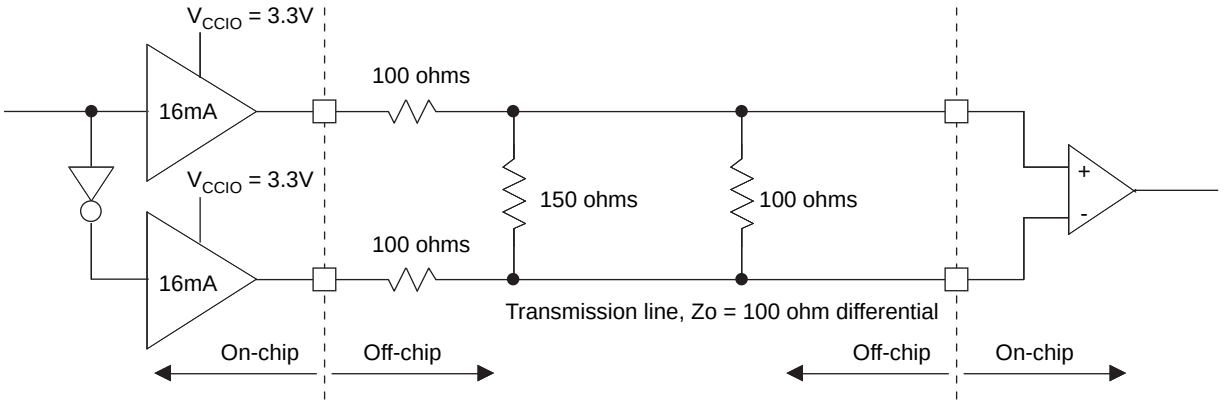


Table 3-3. LVPECL DC Conditions[1]

Over Recommended Operating Conditions

Symbol	Description	Nominal	Units
Z_{OUT}	Output impedance	100	Ohms
R_P	Driver parallel resistor	150	Ohms
R_T	Receiver termination	100	Ohms
V_{OH}	Output high voltage	2.03	V
V_{OL}	Output low voltage	1.27	V
V_{OD}	Output differential voltage	0.76	V
V_{CM}	Output common mode voltage	1.65	V
Z_{BACK}	Back impedance	85.7	Ohms
I_{DC}	DC output current	12.7	mA

1. For input buffer, see LVDS table.

For further information on LVPECL, BLVDS and other differential interfaces please see details of additional technical documentation at the end of the data sheet.

RSDS

The MachXO family supports the differential RSDS standard. The output standard is emulated using complementary LVCMOS outputs in conjunction with a parallel resistor across the driver outputs on all the devices. The RSDS input standard is supported by the LVDS differential input buffer on certain devices. The scheme shown in Figure 3-4 is one possible solution for RSDS standard implementation. Use LVDS25E mode with suggested resistors for RSDS operation. Resistor values in Figure 3-4 are industry standard values for 1% resistors.

Figure 3-4. RSDS (Reduced Swing Differential Standard)

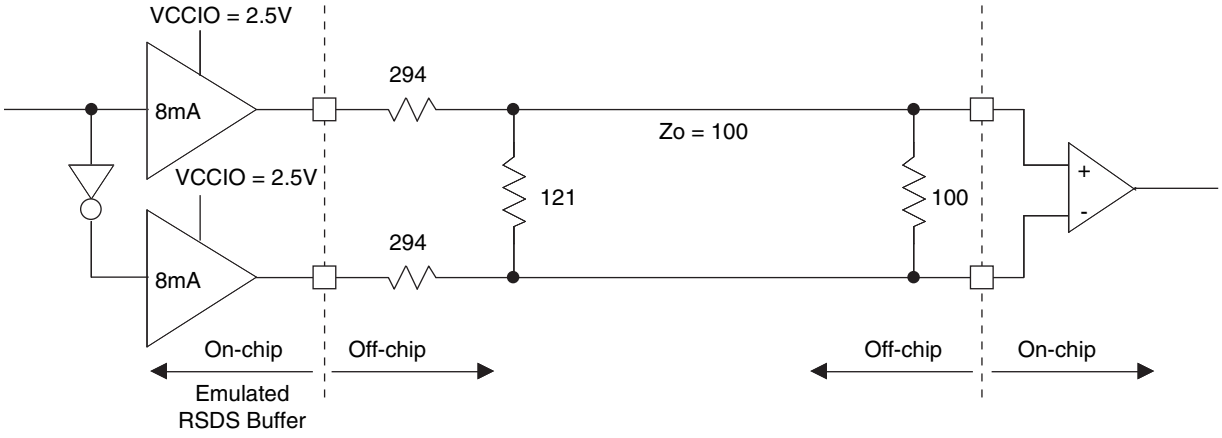


Table 3-4. RSDS DC Conditions

Parameter	Description	Typical	Units
Z_{OUT}	Output impedance	20	Ohms
R_S	Driver series resistor	294	Ohms
R_P	Driver parallel resistor	121	Ohms
R_T	Receiver termination	100	Ohms
V_{OH}	Output high voltage	1.35	V
V_{OL}	Output low voltage	1.15	V
V_{OD}	Output differential voltage	0.20	V
V_{CM}	Output common mode voltage	1.25	V
Z_{BACK}	Back impedance	101.5	Ohms
I_{DC}	DC output current	3.66	mA

Typical Building Block Function Performance[1]

Pin-to-Pin Performance (LVCMOS25 12mA Drive)

Function	-5 Timing	Units
Basic Functions		
16-bit decoder	6.7	ns
4:1 MUX	4.5	ns
16:1 MUX	5.1	ns

Register-to-Register Performance

Function	-5 Timing	Units
Basic Functions		
16:1 MUX	487	MHz
16-bit adder	292	MHz
16-bit counter	388	MHz
64-bit counter	200	MHz
Embedded Memory Functions (1200 and 2280 Devices Only)		
256x36 Single Port RAM	284	MHz
512x18 True-Dual Port RAM	284	MHz
Distributed Memory Functions		
16x2 Single Port RAM	434	MHz
64x2 Single Port RAM	320	MHz
128x4 Single Port RAM	261	MHz
32x2 Pseudo-Dual Port RAM	314	MHz
64x4 Pseudo-Dual Port RAM	271	MHz

1. The above timing numbers are generated using the ispLEVER design tool. Exact performance may vary with device and tool version. The tool uses internal parameters that have been characterized but are not tested on every device.

Rev. A 0.19

Derating Logic Timing

Logic Timing provided in the following sections of the data sheet and the ispLEVER design tools are worst case numbers in the operating range. Actual delays may be much faster. The ispLEVER design tool from Lattice can provide logic timing numbers at a particular temperature and voltage.

MachXO External Switching Characteristics[1]

Over Recommended Operating Conditions

Parameter	Description	Device	-5		-4		-3		Units
			Min.	Max.	Min.	Max.	Min.	Max.	
General I/O Pin Parameters (Using Global Clock without PLL)[1]									
t_{PD}	Best Case t_{PD} Through 1 LUT	LCMXO256	—	3.5	—	4.2	—	4.9	ns
		LCMXO640	—	3.5	—	4.2	—	4.9	ns
		LCMXO1200	—	3.6	—	4.4	—	5.1	ns
		LCMXO2280	—	3.6	—	4.4	—	5.1	ns
t_{CO}	Best Case Clock to Output - From PFU	LCMXO256	—	4.0	—	4.8	—	5.6	ns
		LCMXO640	—	4.0	—	4.8	—	5.7	ns
		LCMXO1200	—	4.3	—	5.2	—	6.1	ns
		LCMXO2280	—	4.3	—	5.2	—	6.1	ns
t_{SU}	Clock to Data Setup - To PFU	LCMXO256	1.3	—	1.6	—	1.8	—	ns
		LCMXO640	1.1	—	1.3	—	1.5	—	ns
		LCMXO1200	1.1	—	1.3	—	1.6	—	ns
		LCMXO2280	1.1	—	1.3	—	1.5	—	ns
t_{H}	Clock to Data Hold - To PFU	LCMXO256	-0.3	—	-0.3	—	-0.3	—	ns
		LCMXO640	-0.1	—	-0.1	—	-0.1	—	ns
		LCMXO1200	0.0	—	0.0	—	0.0	—	ns
		LCMXO2280	-0.4	—	-0.4	—	-0.4	—	ns
f_{MAX_IO}	Clock Frequency of I/O and PFU Register	LCMXO256	—	600	—	550	—	500	MHz
		LCMXO640	—	600	—	550	—	500	MHz
		LCMXO1200	—	600	—	550	—	500	MHz
		LCMXO2280	—	600	—	550	—	500	MHz
t_{SKEW_PRI}	Global Clock Skew Across Device	LCMXO256	—	200	—	220	—	240	ps
		LCMXO640	—	200	—	220	—	240	ps
		LCMXO1200	—	220	—	240	—	260	ps
		LCMXO2280	—	220	—	240	—	260	ps

1. General timing numbers based on LVCMOS2.5V, 12 mA.

Rev. A 0.19

MachXO Internal Timing Parameters[1]

Over Recommended Operating Conditions

Parameter	Description	-5		-4		-3		Units
		Min.	Max.	Min.	Max.	Min.	Max.	
PFU/PFF Logic Mode Timing								
t_{LUT4_PFU}	LUT4 delay (A to D inputs to F output)	—	0.28	—	0.34	—	0.39	ns
t_{LUT6_PFU}	LUT6 delay (A to D inputs to OFX output)	—	0.44	—	0.53	—	0.62	ns
t_{LSR_PFU}	Set/Reset to output of PFU	—	0.90	—	1.08	—	1.26	ns
t_{SUM_PFU}	Clock to Mux (M0,M1) input setup time	0.10	—	0.13	—	0.15	—	ns
t_{HM_PFU}	Clock to Mux (M0,M1) input hold time	-0.05	—	-0.06	—	-0.07	—	ns
t_{SUD_PFU}	Clock to D input setup time	0.13	—	0.16	—	0.18	—	ns
t_{HD_PFU}	Clock to D input hold time	-0.03	—	-0.03	—	-0.04	—	ns
t_{CK2Q_PFU}	Clock to Q delay, D-type register configuration	—	0.40	—	0.48	—	0.56	ns
t_{LE2Q_PFU}	Clock to Q delay latch configuration	—	0.53	—	0.64	—	0.74	ns
t_{LD2Q_PFU}	D to Q throughput delay when latch is enabled	—	0.55	—	0.66	—	0.77	ns
PFU Dual Port Memory Mode Timing								
t_{CORAM_PFU}	Clock to Output	—	0.40	—	0.48	—	0.56	ns
t_{SUDATA_PFU}	Data Setup Time	-0.18	—	-0.22	—	-0.25	—	ns
t_{HDATA_PFU}	Data Hold Time	0.28	—	0.34	—	0.39	—	ns
t_{SUADDR_PFU}	Address Setup Time	-0.46	—	-0.56	—	-0.65	—	ns
t_{HADDR_PFU}	Address Hold Time	0.71	—	0.85	—	0.99	—	ns
t_{SUWREN_PFU}	Write/Read Enable Setup Time	-0.22	—	-0.26	—	-0.30	—	ns
t_{HWREN_PFU}	Write/Read Enable Hold Time	0.33	—	0.40	—	0.47	—	ns
PIO Input/Output Buffer Timing								
t_{IN_PIO}	Input Buffer Delay	—	0.75	—	0.90	—	1.06	ns
t_{OUT_PIO}	Output Buffer Delay	—	1.29	—	1.54	—	1.80	ns
EBR Timing (1200 and 2280 Devices Only)								
t_{CO_EBR}	Clock to output from Address or Data with no output register	—	2.24	—	2.69	—	3.14	ns
t_{COO_EBR}	Clock to output from EBR output Register	—	0.54	—	0.64	—	0.75	ns
t_{SUDATA_EBR}	Setup Data to EBR Memory	-0.26	—	-0.31	—	-0.37	—	ns
t_{HDATA_EBR}	Hold Data to EBR Memory	0.41	—	0.49	—	0.57	—	ns
t_{SUADDR_EBR}	Setup Address to EBR Memory	-0.26	—	-0.31	—	-0.37	—	ns
t_{HADDR_EBR}	Hold Address to EBR Memory	0.41	—	0.49	—	0.57	—	ns
t_{SUWREN_EBR}	Setup Write/Read Enable to EBR Memory	-0.17	—	-0.20	—	-0.23	—	ns
t_{HWREN_EBR}	Hold Write/Read Enable to EBR Memory	0.26	—	0.31	—	0.36	—	ns
t_{SUCE_EBR}	Clock Enable Setup Time to EBR Output Register	0.19	—	0.23	—	0.27	—	ns
t_{HCE_EBR}	Clock Enable Hold Time to EBR Output Register	-0.13	—	-0.16	—	-0.18	—	ns
t_{RSTO_EBR}	Reset To Output Delay Time from EBR Output Register	—	1.03	—	1.23	—	1.44	ns
PLL Parameters (1200 and 2280 Devices Only)								
t_{RSTREC}	Reset Recovery to Rising Clock	1.00	—	1.00	—	1.00	—	ns
t_{RSTSU}	Reset Signal Setup Time	1.00	—	1.00	—	1.00	—	ns

1. Internal parameters are characterized but not tested on every device.

Rev. A 0.19

MachXO Family Timing Adders[1, 2, 3]

Over Recommended Operating Conditions

Buffer Type	Description	-5	-4	-3	Units
Input Adjusters					
LVDS25[4]	LVDS	0.44	0.53	0.61	ns
BLVDS25[4]	BLVDS	0.44	0.53	0.61	ns
LVPECL33[4]	LVPECL	0.42	0.50	0.59	ns
LVTTL33	LVTTL	0.01	0.01	0.01	ns
LVCMOS33	LVCMOS 3.3	0.01	0.01	0.01	ns
LVCMOS25	LVCMOS 2.5	0.00	0.00	0.00	ns
LVCMOS18	LVCMOS 1.8	0.07	0.08	0.10	ns
LVCMOS15	LVCMOS 1.5	0.14	0.17	0.19	ns
LVCMOS12	LVCMOS 1.2	0.40	0.48	0.56	ns
PCI33[4]	PCI	0.01	0.01	0.01	ns
Output Adjusters					
LVDS25E	LVDS 2.5 E	-0.13	-0.15	-0.18	ns
LVDS25[4]	LVDS 2.5	-0.21	-0.26	-0.30	ns
BLVDS25	BLVDS 2.5	-0.03	-0.03	-0.04	ns
LVPECL33	LVPECL 3.3	0.04	0.04	0.05	ns
LVTTL33_4mA	LVTTL 4mA drive	0.04	0.04	0.05	ns
LVTTL33_8mA	LVTTL 8mA drive	0.06	0.07	0.08	ns
LVTTL33_12mA	LVTTL 12mA drive	-0.01	-0.01	-0.01	ns
LVTTL33_16mA	LVTTL 16mA drive	0.50	0.60	0.70	ns
LVCMOS33_4mA	LVCMOS 3.3 4mA drive	0.04	0.04	0.05	ns
LVCMOS33_8mA	LVCMOS 3.3 8mA drive	0.06	0.07	0.08	ns
LVCMOS33_12mA	LVCMOS 3.3 12mA drive	-0.01	-0.01	-0.01	ns
LVCMOS33_14mA	LVCMOS 3.3 14mA drive	0.50	0.60	0.70	ns
LVCMOS25_4mA	LVCMOS 2.5 4mA drive	0.05	0.06	0.07	ns
LVCMOS25_8mA	LVCMOS 2.5 8mA drive	0.10	0.12	0.13	ns
LVCMOS25_12mA	LVCMOS 2.5 12mA drive	0.00	0.00	0.00	ns
LVCMOS25_14mA	LVCMOS 2.5 14mA drive	0.34	0.40	0.47	ns
LVCMOS18_4mA	LVCMOS 1.8 4mA drive	0.11	0.13	0.15	ns
LVCMOS18_8mA	LVCMOS 1.8 8mA drive	0.05	0.06	0.06	ns
LVCMOS18_12mA	LVCMOS 1.8 12mA drive	-0.06	-0.07	-0.08	ns
LVCMOS18_14mA	LVCMOS 1.8 14mA drive	0.06	0.07	0.09	ns
LVCMOS15_4mA	LVCMOS 1.5 4mA drive	0.15	0.19	0.22	ns
LVCMOS15_8mA	LVCMOS 1.5 8mA drive	0.05	0.06	0.07	ns
LVCMOS12_2mA	LVCMOS 1.2 2mA drive	0.26	0.31	0.36	ns
LVCMOS12_6mA	LVCMOS 1.2 6mA drive	0.05	0.06	0.07	ns
PCI33[4]	PCI33	1.85	2.22	2.59	ns

1. Timing adders are characterized but not tested on every device.
2. LVCMOS timing is measured with the load specified in Switching Test Conditions table.
3. All other standards tested according to the appropriate specifications.
4. I/O standard only available in LCMXO1200 and LCMXO2280 devices.

Rev. A 0.19

sysCLOCK PLL Timing

Over Recommended Operating Conditions

Parameter	Descriptions	Conditions	Min.	Max.	Units
f_{IN}	Input Clock Frequency (CLKI, CLKFB)		25	420	MHz
		Input Divider (M) = 1; Feedback Divider (N) <= 4[5, 6]	18	25	MHz
f_{OUT}	Output Clock Frequency (CLKOP, CLKOS)		25	420	MHz
f_{OUT2}	K-Divider Output Frequency (CLKOK)		0.195	210	MHz
f_{VCO}	PLL VCO Frequency		420	840	MHz
f_{PFD}	Phase Detector Input Frequency		25	—	MHz
		Input Divider (M) = 1; Feedback Divider (N) <= 4[5, 6]	18	25	MHz
AC Characteristics					
t_{DT}	Output Clock Duty Cycle	Default duty cycle selected[3]	45	55	%
t_{PH}[4]	Output Phase Accuracy		—	0.05	UI
t_{OPJIT}[1]	Output Clock Period Jitter	f_{OUT} >= 100 MHz	—	+/-120	ps
		f_{OUT} < 100 MHz	—	0.02	UIPP
t_{SK}	Input Clock to Output Clock Skew	Divider ratio = integer	—	+/-200	ps
t_{W}	Output Clock Pulse Width	At 90% or 10%[3]	1	—	ns
t_{LOCK}[2]	PLL Lock-in Time		—	150	µs
t_{PA}	Programmable Delay Unit		100	450	ps
t_{IPJIT}	Input Clock Period Jitter	$f_{OUT} \geq$ 100 MHz	—	+/-200	ps
		f_{OUT} < 100 MHz	—	0.02	UI
t_{FBKDLY}	External Feedback Delay		—	10	ns
t_{HI}	Input Clock High Time	90% to 90%	0.5	—	ns
t_{LO}	Input Clock Low Time	10% to 10%	0.5	—	ns
t_{RST}	RST Pulse Width		10	—	ns

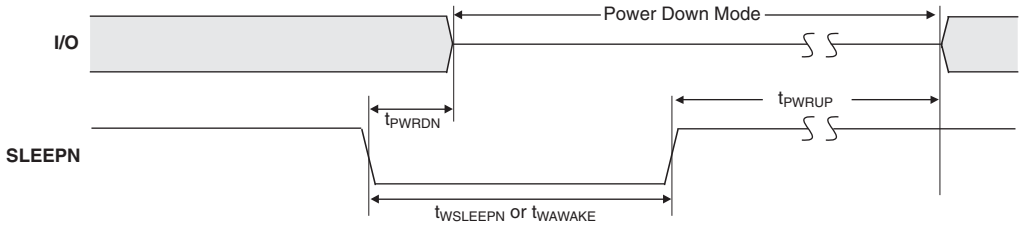
1. Jitter sample is taken over 10,000 samples of the primary PLL output with a clean reference clock.
2. Output clock is valid after t_{LOCK} for PLL reset and dynamic delay adjustment.
3. Using LVDS output buffers.
4. CLKOS as compared to CLKOP output.
5. When using an input frequency less than 25 MHz the output frequency must be less than or equal to 4 times the input frequency.
6. The on-chip oscillator can be used to provide reference clock input to the PLL provided the output frequency restriction for clock inputs below 25 MHz are followed.

Rev. A 0.19

MachXO “C” Sleep Mode Timing

Symbol	Parameter	Device	Min.	Typ.	Max	Units
t_{PWRDN}	SLEEPN Low to Power Down	All	—	—	400	ns
t_{PWRUP}	SLEEPN High to Power Up	LCMXO256	—	—	400	µs
		LCMXO640	—	—	600	µs
		LCMXO1200	—	—	800	µs
		LCMXO2280	—	—	1000	µs
$t_{WSLEEPN}$	SLEEPN Pulse Width	All	400	—	—	ns
t_{WAWAKE}	SLEEPN Pulse Rejection	All	—	—	100	ns

Rev. A 0.19



Flash Download Time

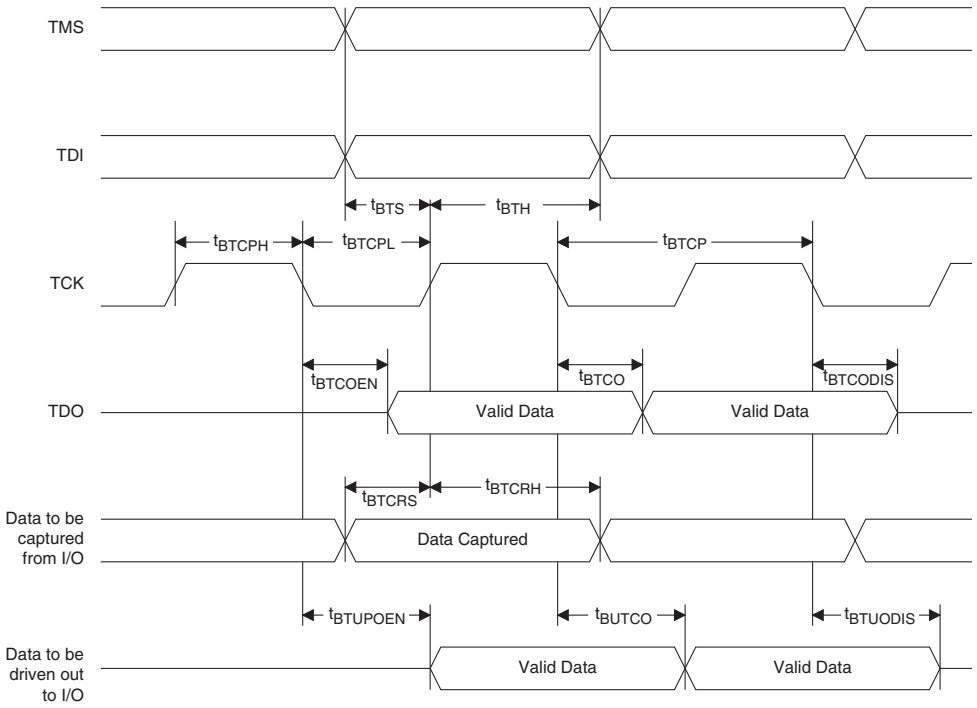
Symbol	Parameter		Min.	Typ.	Max.	Units
$t_{REFRESH}$	Minimum V_{CC} or V_{CCAUX} (later of the two supplies) to Device I/O Active	LCMXO256	—	—	0.4	ms
		LCMXO640	—	—	0.6	ms
		LCMXO1200	—	—	0.8	ms
		LCMXO2280	—	—	1.0	ms

JTAG Port Timing Specifications

Symbol	Parameter	Min.	Max.	Units
f_{MAX}	TCK [BSCAN] clock frequency	—	25	MHz
t_{BTCP}	TCK [BSCAN] clock pulse width	40	—	ns
t_{BTCPH}	TCK [BSCAN] clock pulse width high	20	—	ns
t_{BTCPL}	TCK [BSCAN] clock pulse width low	20	—	ns
t_{BTS}	TCK [BSCAN] setup time	8	—	ns
t_{BTH}	TCK [BSCAN] hold time	10	—	ns
t_{BTRF}	TCK [BSCAN] rise/fall time	50	—	mV/ns
t_{BTCO}	TAP controller falling edge of clock to output valid	—	10	ns
$t_{BTCODIS}$	TAP controller falling edge of clock to output disabled	—	10	ns
t_{BTCOEN}	TAP controller falling edge of clock to output enabled	—	10	ns
t_{BTCRS}	BSCAN test capture register setup time	8	—	ns
t_{BTCRH}	BSCAN test capture register hold time	25	—	ns
t_{BUTCO}	BSCAN test update register, falling edge of clock to output valid	—	25	ns
$t_{BTUODIS}$	BSCAN test update register, falling edge of clock to output disabled	—	25	ns
$t_{BTUPOEN}$	BSCAN test update register, falling edge of clock to output enabled	—	25	ns

Rev. A 0.19

Figure 3-5. JTAG Port Timing Waveforms



Switching Test Conditions

Figure 3-6 shows the output test load that is used for AC testing. The specific values for resistance, capacitance, voltage, and other test conditions are shown in Figure 3-5.

Figure 3-6. Output Test Load, LVTTL and LVCMOS Standards

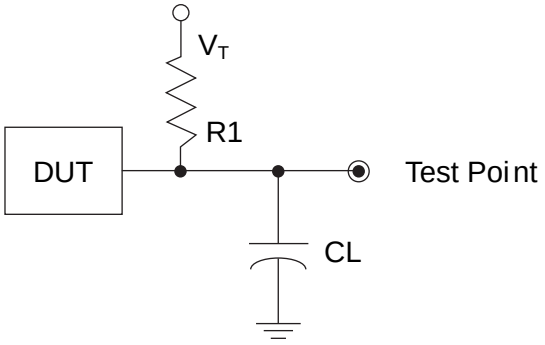


Table 3-5. Test Fixture Required Components, Non-Terminated Interfaces

Test Condition	R_1	C_L	Timing Ref.	V_T
LVTTL and LVCMOS settings (L -> H, H -> L)	∞	0pF	LVTTL, LVCMOS 3.3 = 1.5V	—
			LVCMOS 2.5 = $V_{CCIO}/2$	—
			LVCMOS 1.8 = $V_{CCIO}/2$	—
			LVCMOS 1.5 = $V_{CCIO}/2$	—
			LVCMOS 1.2 = $V_{CCIO}/2$	—
LVTTL and LVCMOS 3.3 (Z -> H)	188	0pF	1.5	V_{OL}
LVTTL and LVCMOS 3.3 (Z -> L)				V_{OH}
Other LVCMOS (Z -> H)			$V_{CCIO}/2$	V_{OL}
Other LVCMOS (Z -> L)			$V_{CCIO}/2$	V_{OH}
LVTTL + LVCMOS (H -> Z)			V_{OH} - 0.15	V_{OL}
LVTTL + LVCMOS (L -> Z)			V_{OL} - 0.15	V_{OH}

Note: Output test conditions for all other interfaces are determined by the respective standards.

Signal Descriptions

Signal Name	I/O	Descriptions
General Purpose		
P[Edge] [Row/Column Number]_[A/B/C/D/E/F]	I/O	[Edge] indicates the edge of the device on which the pad is located. Valid edge designations are L (Left), B (Bottom), R (Right), T (Top). [Row/Column Number] indicates the PFU row or the column of the device on which the PIO Group exists. When Edge is T (Top) or (Bottom), only need to specify Row Number. When Edge is L (Left) or R (Right), only need to specify Column Number. [A/B/C/D/E/F] indicates the PIO within the group to which the pad is connected. Some of these user programmable pins are shared with special function pins. When not used as special function pins, these pins can be programmed as I/Os for user logic. During configuration of the user-programmable I/Os, the user has an option to tri-state the I/Os and enable an internal pull-up resistor. This option also applies to unused pins (or those not bonded to a package pin). The default during configuration is for user-programmable I/Os to be tri-stated with an internal pull-up resistor enabled. When the device is erased, I/Os will be tri-stated with an internal pull-up resistor enabled.
GSRN	I	Global RESET signal (active low). Dedicated pad, when not in use it can be used as an I/O pin.
TSALL	I	TSALL is a dedicated pad for the global output enable signal. When TSALL is high all the outputs are tristated. It is a dual function pin. When not in use, it can be used as an I/O pin.
NC	—	No connect.
GND	—	GND - Ground. Dedicated pins.
V_{CC}	—	VCC - The power supply pins for core logic. Dedicated pins.
V_{CCAUX}	—	VCCAUX - the Auxiliary power supply pin. This pin powers up a variety of internal circuits including all the differential and referenced input buffers. Dedicated pins.
V_{CCIOx}	—	V_{CCIO} - The power supply pins for I/O Bank x. Dedicated pins.
SLEEPN[1]	I	Sleep Mode pin - Active low sleep pin.þ When this pin is held high, the device operates normally.þ This pin has a weak internal pull-up, but when unused, an external pull-up to V_{CC} is recommended. When driven low, the device moves into Sleep mode after a specified time.
PLL and Clock Functions (Used as user programmable I/O pins when not used for PLL or clock pins)		
[LOC][0]_PLL[T, C]_IN	—	Reference clock (PLL) input Pads: [LOC] indicates location. Valid designations are ULM (Upper PLL) and LLM (Lower PLL). T = true and C = complement.
[LOC][0]_PLL[T, C]_FB	—	Optional feedback (PLL) input Pads: [LOC] indicates location. Valid designations are ULM (Upper PLL) and LLM (Lower PLL). T = true and C = complement.
PCLK [n]_[1:0]	—	Primary Clock Pads, n per side.
Test and Programming (Dedicated pins)		
TMS	I	Test Mode Select input pin, used to control the 1149.1 state machine.
TCK	I	Test Clock input pin, used to clock the 1149.1 state machine.
TDI	I	Test Data input pin, used to load data into the device using an 1149.1 state machine.
TDO	O	Output pin -Test Data output pin used to shift data out of the device using 1149.1.

1. Applies to MachXO “C” devices only. NC for “E” devices.

Pin Information Summary

Pin Type		LCMXO256C/E		LCMXO640C/E				
		100 TQFP	100 csBGA	100 TQFP	144 TQFP	100 csBGA	132 csBGA	256 caBGA / 256 ftBGA
Single Ended User I/O		78	78	74	113	74	101	159
Differential Pair User I/O[1]		38	38	17	43	17	42	79
Muxed		6	6	6	6	6	6	6
TAP		4	4	4	4	4	4	4
Dedicated (Total Without Supplies)		5	5	5	5	5	5	5
VCC		2	2	2	4	2	4	4
VCCAUX		1	1	1	2	1	2	2
VCCIO	Bank0	3	3	2	2	2	2	4
	Bank1	3	3	2	2	2	2	4
	Bank2	—	—	2	2	2	2	4
	Bank3	—	—	2	2	2	2	4
GND		8	8	10	12	10	12	18
NC		0	0	0	0	0	0	52
Single Ended/Differential I/O per Bank	Bank0	41/20	41/20	18/5	29/10	18/5	26/11	42/21
	Bank1	37/18	37/18	21/4	30/11	21/4	27/12	40/20
	Bank2	—	—	14/2	24/9	14/2	21/9	36/18
	Bank3	—	—	21/6	30/13	21/6	27/10	40/20

1. These devices support emulated LVDS outputs.þLVDS inputs are not supported.

Pin Type		LCMXO1200C/E				LCMXO2280C/E				
		100 TQFP	144 TQFP	132 csBGA	256 caBGA / 256 ftBGA	100 TQFP	144 TQFP	132 csBGA	256 caBGA / 256 ftBGA	324 ftBGA
Single Ended User I/O		73	113	101	211	73	113	101	211	271
Differential Pair User I/O[1]		27	48	42	105	30	47	41	105	134
Muxed		6	6	6	6	6	6	6	6	6
TAP		4	4	4	4	4	4	4	4	4
Dedicated (Total Without Supplies)		5	5	5	5	5	5	5	5	5
VCC		4	4	4	4	2	4	4	4	6
VCCAUX		2	2	2	2	2	2	2	2	2
VCCIO	Bank0	1	1	1	2	1	1	1	2	2
	Bank1	1	1	1	2	1	1	1	2	2
	Bank2	1	1	1	2	1	1	1	2	2
	Bank3	1	1	1	2	1	1	1	2	2
	Bank4	1	1	1	2	1	1	1	2	2
	Bank5	1	1	1	2	1	1	1	2	2
	Bank6	1	1	1	2	1	1	1	2	2
	Bank7	1	1	1	2	1	1	1	2	2
GND		8	12	12	18	8	12	12	18	24
NC		0	0	0	0	0	0	0	0	0
Single Ended/Differential I/O per Bank	Bank0	10/3	14/6	13/5	26/13	9/3	13/6	12/5	24/12	34/17
	Bank1	8/2	15/7	13/5	28/14	9/3	16/7	14/5	30/15	36/18
	Bank2	10/4	15/7	13/6	26/13	10/4	15/7	13/6	26/13	34/17
	Bank3	11/5	15/7	14/7	28/14	11/5	15/7	14/7	28/14	34/17
	Bank4	8/3	14/5	13/5	27/13	8/3	14/4	13/4	29/14	35/17
	Bank5	5/2	10/4	8/2	22/11	5/2	10/4	8/2	20/10	30/15
	Bank6	10/3	15/6	13/6	28/14	10/4	15/6	13/6	28/14	34/17
	Bank7	11/5	15/6	14/6	26/13	11/5	15/6	14/6	26/13	34/17

1. These devices support on-chip LVDS buffers for left and right I/O Banks.

Power Supply and NC

Signal	100 TQFP[1]	144 TQFP[1]	100 csBGA[2]
VCC	**LCMXO256/640**: 35, 90 **LCMXO1200/2280**: 17, 35, 66, 91	21, 52, 93, 129	P7, B6
VCCIO0	**LCMXO256**: 60, 74, 92 **LCMXO640**: 80, 92 **LCMXO1200/2280**: 94	**LCMXO640**: 117, 135 **LCMXO1200/2280**: 135	**LCMXO256**: H14, A14, B5 **LCMXO640**: B12, B5
VCCIO1	**LCMXO256**: 10, 24, 41 **LCMXO640**: 60, 74 **LCMXO1200/2280**: 80	**LCMXO640**: 82, 98 **LCMXO1200/2280**: 117	**LCMXO256**: G1, P1, P10 **LCMXO640**: H14, A14
VCCIO2	**LCMXO256**: None **LCMXO640**: 29, 41 **LCMXO1200/2280**: 70	**LCMXO640**: 38, 63 **LCMXO1200/2280**: 98	**LCMXO256**: None **LCMXO640**: P4, P10
VCCIO3	**LCMXO256**: None **LCMXO640**: 10, 24 **LCMXO1200/2280**: 56	**LCMXO640**: 10, 26 **LCMXO1200/2280**: 82	**LCMXO256**: None **LCMXO640**: G1, P1
VCCIO4	**LCMXO256/640**: None **LCMXO1200/2280**: 44	**LCMXO640**: None **LCMXO1200/2280**: 63	—
VCCIO5	**LCMXO256/640**: None **LCMXO1200/2280**: 27	**LCMXO640**: None **LCMXO1200/2280**: 38	—
VCCIO6	**LCMXO256/640**: None **LCMXO1200/2280**: 20	**LCMXO640**: None **LCMXO1200/2280**: 26	—
VCCIO7	**LCMXO256/640**: None **LCMXO1200/2280**: 6	**LCMXO640**: None **LCMXO1200/2280**: 10	—
VCCAUX	**LCMXO256/640**: 88 **LCMXO1200/2280**: 36, 90	53, 128	B7
GND[3]	**LCMXO256**: 40, 84, 62, 75, 93, 12, 25, 42 **LCMXO640**: 40, 84, 81, 93, 62, 75, 30, 42, 12, 25 **LCMXO1200/2280**: 9, 41, 59, 83, 100, 76, 50, 26	16, 59, 88, 123, 118, 136, 83, 99, 37, 64, 11, 27	**LCMXO256**: N9, B9, G14, B13, A4, H1, N2, N10 **LCMXO640**: N9, B9, A10, A4, G14, B13, N3, N10, H1, N2
NC[4]			—

1. Pin orientation follows the conventional order from pin 1 marking of the top side view and counter-clockwise.
2. Pin orientation A1 starts from the upper left corner of the top side view with alphabetical order ascending vertically and numerical order ascending horizontally.
3. All grounds must be electrically connected at the board level. For fpBGA and ftBGA packages, the total number of GND balls is less than the actual number of GND logic connections from the die to the common package GND plane.
4. NC pins should not be connected to any active signals, VCC or GND.

Power Supply and NC (Cont.)

Signal	132 csBGA[1]	256 caBGA / 256 ftBGA[1]	324 ftBGA[1]
VCC	H3, P6, G12, C7	G7, G10, K7, K10	F14, G11, G9, H7, L7, M9
VCCIO0	**LCMXO640**: B11, C5 **LCMXO1200/2280**: C5	**LCMXO640**: F8, F7, F9, F10 **LCMXO1200/2280**: F8, F7	G8, G7
VCCIO1	**LCMXO640**: L12, E12 **LCMXO1200/2280**: B11	**LCMXO640**: H11, G11, K11, J11 **LCMXO1200/2280**: F9, F10	G12, G10
VCCIO2	**LCMXO640**: N2, M10 **LCMXO1200/2280**: E12	**LCMXO640**: L9, L10, L8, L7 **LCMXO1200/2280**: H11, G11	J12, H12
VCCIO3	**LCMXO640**: D2, K3 **LCMXO1200/2280**: L12	**LCMXO640**: K6, J6, H6, G6 **LCMXO1200/2280**: K11, J11	L12, K12
VCCIO4	**LCMXO640**: None **LCMXO1200/2280**: M10	**LCMXO640**: None **LCMXO1200/2280**: L9, L10	M12, M11
VCCIO5	**LCMXO640**: None **LCMXO1200/2280**: N2	**LCMXO640**: None **LCMXO1200/2280**: L8, L7	M8, R9
VCCIO6	**LCMXO640**: None **LCMXO1200/2280**: K3	**LCMXO640**: None **LCMXO1200/2280**: K6, J6	M7, K7
VCCIO7	**LCMXO640**: None **LCMXO1200/2280**: D2	**LCMXO640**: None **LCMXO1200/2280**: H6, G6	H6, J7
VCCAUX	P7, A7	T9, A8	M10, F9
GND[2]	F1, P9, J14, C9, A10, B4, L13, D13, P2, N11, E1, L2	A1, A16, F11, G8, G9, H7, H8, H9, H10, J7, J8, J9, J10, K8, K9, L6, T1, T16	E14, F16, H10, H11, H8, H9, J10, J11, J4, J8, J9, K10, K11, K17, K8, K9, L10, L11, L8, L9, N2, P14, P5, R7
NC[3]	—	**LCMXO640**: E4, E5, F5, F6, C3, C2, G4, G5, H4, H5, K5, K4, M5, M4, P2, P3, N5, N6, M7, M8, N10, N11, R15, R16, P15, P16, M11, L11, N12, N13, M13, M12, K12, J12, F12, F13, E12, E13, D13, D14, B15, A15, C14, B14, E11, E10, E7, E6, D4, D3, B3, B2 **LCMXO1200**: None **LCMXO2280**: None	—

1. Pin orientation A1 starts from the upper left corner of the top side view with alphabetical order ascending vertically and numerical order ascending horizontally.
2. All grounds must be electrically connected at the board level. For fpBGA and ftBGA packages, the total number of GND balls is less than the actual number of GND logic connections from the die to the common package GND plane.
3. NC pins should not be connected to any active signals, VCC or GND.

LCMXO256 and LCMXO640 Logic Signal Connections: 100 TQFP

Pin Number	LCMXO256				LCMXO640			
	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
1	PL2A	1		T	PL2A	3		T
2	PL2B	1		C	PL2C	3		T
3	PL3A	1		T	PL2B	3		C
4	PL3B	1		C	PL2D	3		C
5	PL3C	1		T	PL3A	3		T
6	PL3D	1		C	PL3B	3		C
7	PL4A	1		T	PL3C	3		T
8	PL4B	1		C	PL3D	3		C
9	PL5A	1		T	PL4A	3		
10	VCCIO1	1			VCCIO3	3		
11	PL5B	1		C	PL4C	3		T
12	GNDIO1	1			GNDIO3	3		
13	PL5C	1		T	PL4D	3		C
14	PL5D	1	GSRN	C	PL5B	3	GSRN	
15	PL6A	1		T	PL7B	3		
16	PL6B	1	TSALL	C	PL8C	3	TSALL	T
17	PL7A	1		T	PL8D	3		C
18	PL7B	1		C	PL9A	3		
19	PL7C	1		T	PL9C	3		
20	PL7D	1		C	PL10A	3		
21	PL8A	1		T	PL10C	3		
22	PL8B	1		C	PL11A	3		
23	PL9A	1		T	PL11C	3		
24	VCCIO1	1			VCCIO3	3		
25	GNDIO1	1			GNDIO3	3		
26	TMS	1	TMS		TMS	2	TMS	
27	PL9B	1		C	PB2C	2		
28	TCK	1	TCK		TCK	2	TCK	
29	PB2A	1		T	VCCIO2	2		
30	PB2B	1		C	GNDIO2	2		
31	TDO	1	TDO		TDO	2	TDO	
32	PB2C	1		T	PB4C	2		
33	TDI	1	TDI		TDI	2	TDI	
34	PB2D	1		C	PB4E	2		
35	VCC	-			VCC	-		
36	PB3A	1	PCLK1_1**	T	PB5B	2	PCLK2_1**	
37	PB3B	1		C	PB5D	2		
38	PB3C	1	PCLK1_0**	T	PB6B	2	PCLK2_0**	
39	PB3D	1		C	PB6C	2		
40	GND	-			GND	-		
41	VCCIO1	1			VCCIO2	2		
42	GNDIO1	1			GNDIO2	2		

LCMXO256 and LCMXO640 Logic Signal Connections: 100 TQFP (Cont.)

	LCMXO256				LCMXO640			
Pin Number	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
43	PB4A	1		T	PB8B	2		
44	PB4B	1		C	PB8C	2		T
45	PB4C	1		T	PB8D	2		C
46	PB4D	1		C	PB9A	2		
47	PB5A	1			PB9C	2		T
48*	SLEEPN	-	SLEEPN		SLEEPN	-	SLEEPN	
49	PB5C	1		T	PB9D	2		C
50	PB5D	1		C	PB9F	2		
51	PR9B	0		C	PR11D	1		C
52	PR9A	0		T	PR11B	1		C
53	PR8B	0		C	PR11C	1		T
54	PR8A	0		T	PR11A	1		T
55	PR7D	0		C	PR10D	1		C
56	PR7C	0		T	PR10C	1		T
57	PR7B	0		C	PR10B	1		C
58	PR7A	0		T	PR10A	1		T
59	PR6B	0		C	PR9D	1		
60	VCCIO0	0			VCCIO1	1		
61	PR6A	0		T	PR9B	1		
62	GNDIO0	0			GNDIO1	1		
63	PR5D	0		C	PR7B	1		
64	PR5C	0		T	PR6C	1		
65	PR5B	0		C	PR6B	1		
66	PR5A	0		T	PR5D	1		
67	PR4B	0		C	PR5B	1		
68	PR4A	0		T	PR4D	1		
69	PR3D	0		C	PR4B	1		
70	PR3C	0		T	PR3D	1		
71	PR3B	0		C	PR3B	1		
72	PR3A	0		T	PR2D	1		
73	PR2B	0		C	PR2B	1		
74	VCCIO0	0			VCCIO1	1		
75	GNDIO0	0			GNDIO1	1		
76	PR2A	0		T	PT9F	0		C
77	PT5C	0			PT9E	0		T
78	PT5B	0		C	PT9C	0		
79	PT5A	0		T	PT9A	0		
80	PT4F	0		C	VCCIO0	0		
81	PT4E	0		T	GNDIO0	0		
82	PT4D	0		C	PT7E	0		
83	PT4C	0		T	PT7A	0		
84	GND	-			GND	-		

LCMXO256 and LCMXO640 Logic Signal Connections: 100 TQFP (Cont.)

Pin Number	LCMXO256				LCMXO640			
	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
85	PT4B	0	PCLK0_1**	C	PT6B	0	PCLK0_1**	
86	PT4A	0	PCLK0_0**	T	PT5B	0	PCLK0_0**	C
87	PT3D	0		C	PT5A	0		T
88	VCCAUX	-			VCCAUX	-		
89	PT3C	0		T	PT4F	0		
90	VCC	-			VCC	-		
91	PT3B	0		C	PT3F	0		
92	VCCIO0	0			VCCIO0	0		
93	GNDIO0	0			GNDIO0	0		
94	PT3A	0		T	PT3B	0		C
95	PT2F	0		C	PT3A	0		T
96	PT2E	0		T	PT2F	0		C
97	PT2D	0		C	PT2E	0		T
98	PT2C	0		T	PT2B	0		C
99	PT2B	0		C	PT2C	0		
100	PT2A	0		T	PT2A	0		T

* NC for “E” devices.
** Primary clock inputs are single-ended.

LCMXO1200 and LCMXO2280 Logic Signal Connections: 100 TQFP

Pin Number	LCMXO1200				LCMXO2280			
	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
1	PL2A	7		T	PL2A	7	LUM0_PLLT_FB_A	T
2	PL2B	7		C	PL2B	7	LUM0_PLLC_FB_A	C
3	PL3C	7		T	PL3C	7	LUM0_PLLT_IN_A	T
4	PL3D	7		C	PL3D	7	LUM0_PLLC_IN_A	C
5	PL4B	7			PL4B	7		
6	VCCIO7	7			VCCIO7	7		
7	PL6A	7		T*	PL7A	7		T*
8	PL6B	7	GSRN	C*	PL7B	7	GSRN	C*
9	GND	-			GND	-		
10	PL7C	7		T	PL9C	7		T
11	PL7D	7		C	PL9D	7		C
12	PL8C	7		T	PL10C	7		T
13	PL8D	7		C	PL10D	7		C
14	PL9C	6			PL11C	6		
15	PL10A	6		T*	PL13A	6		T*
16	PL10B	6		C*	PL13B	6		C*
17	VCC	-			VCC	-		
18	PL11B	6			PL14D	6		C
19	PL11C	6	TSALL		PL14C	6	TSALL	T
20	VCCIO6	6			VCCIO6	6		
21	PL13C	6			PL16C	6		
22	PL14A	6	LLM0_PLLT_FB_A	T*	PL17A	6	LLM0_PLLT_FB_A	T*
23	PL14B	6	LLM0_PLLC_FB_A	C*	PL17B	6	LLM0_PLLC_FB_A	C*
24	PL15A	6	LLM0_PLLT_IN_A	T*	PL18A	6	LLM0_PLLT_IN_A	T*
25	PL15B	6	LLM0_PLLC_IN_A	C*	PL18B	6	LLM0_PLLC_IN_A	C*
26**	GNDIO6 GNDIO5	-			GNDIO6 GNDIO5	-		
27	VCCIO5	5			VCCIO5	5		
28	TMS	5	TMS		TMS	5	TMS	
29	TCK	5	TCK		TCK	5	TCK	
30	PB3B	5			PB3B	5		
31	PB4A	5		T	PB4A	5		T
32	PB4B	5		C	PB4B	5		C
33	TDO	5	TDO		TDO	5	TDO	
34	TDI	5	TDI		TDI	5	TDI	
35	VCC	-			VCC	-		
36	VCCAUX	-			VCCAUX	-		
37	PB6E	5		T	PB8E	5		T
38	PB6F	5		C	PB8F	5		C
39	PB7B	4	PCLK4_1****		PB10F	4	PCLK4_1****	
40	PB7F	4	PCLK4_0****		PB10B	4	PCLK4_0****	
41	GND	-			GND	-		

LCMXO1200 and LCMXO2280 Logic Signal Connections: 100 TQFP (Cont.)

Pin Number	LCMXO1200				LCMXO2280			
	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
42	PB9A	4		T	PB12A	4		T
43	PB9B	4		C	PB12B	4		C
44	VCCIO4	4			VCCIO4	4		
45	PB10A	4		T	PB13A	4		T
46	PB10B	4		C	PB13B	4		C
47***	SLEEPN	-	SLEEPN		SLEEPN	-	SLEEPN	
48	PB11A	4		T	PB16A	4		T
49	PB11B	4		C	PB16B	4		C
50**	GNDIO3 GNDIO4	-			GNDIO3 GNDIO4	-		
51	PR16B	3			PR19B	3		
52	PR15B	3		C*	PR18B	3		C*
53	PR15A	3		T*	PR18A	3		T*
54	PR14B	3		C*	PR17B	3		C*
55	PR14A	3		T*	PR17A	3		T*
56	VCCIO3	3			VCCIO3	3		
57	PR12B	3		C*	PR15B	3		C*
58	PR12A	3		T*	PR15A	3		T*
59	GND	-			GND	-		
60	PR10B	3		C*	PR13B	3		C*
61	PR10A	3		T*	PR13A	3		T*
62	PR9B	3		C*	PR11B	3		C*
63	PR9A	3		T*	PR11A	3		T*
64	PR8B	2		C*	PR10B	2		C*
65	PR8A	2		T*	PR10A	2		T*
66	VCC	-			VCC	-		
67	PR6C	2			PR8C	2		
68	PR6B	2		C*	PR8B	2		C*
69	PR6A	2		T*	PR8A	2		T*
70	VCCIO2	2			VCCIO2	2		
71	PR4D	2			PR5D	2		
72	PR4B	2		C*	PR5B	2		C*
73	PR4A	2		T*	PR5A	2		T*
74	PR2B	2		C	PR3B	2		C*
75	PR2A	2		T	PR3A	2		T*
76**	GNDIO1 GNDIO2	-			GNDIO1 GNDIO2	-		
77	PT11C	1			PT15C	1		
78	PT11B	1		C	PT14B	1		C
79	PT11A	1		T	PT14A	1		T
80	VCCIO1	1			VCCIO1	1		
81	PT9E	1			PT12D	1		C

LCMXO1200 and LCMXO2280 Logic Signal Connections: 100 TQFP (Cont.)

Pin Number	LCMXO1200				LCMXO2280			
	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
82	PT9A	1			PT12C	1		T
83	GND	-			GND	-		
84	PT8B	1		C	PT11B	1		C
85	PT8A	1		T	PT11A	1		T
86	PT7D	1	PCLK1_1****		PT10B	1	PCLK1_1****	
87	PT6F	0	PCLK0_0****		PT9B	1	PCLK1_0****	
88	PT6D	0		C	PT8F	0		C
89	PT6C	0		T	PT8E	0		T
90	VCCAUX	-			VCCAUX	-		
91	VCC	-			VCC	-		
92	PT5B	0			PT6D	0		
93	PT4B	0			PT6F	0		
94	VCCIO0	0			VCCIO0	0		
95	PT3D	0		C	PT4B	0		C
96	PT3C	0		T	PT4A	0		T
97	PT3B	0			PT3B	0		
98	PT2B	0		C	PT2B	0		C
99	PT2A	0		T	PT2A	0		T
100**	GNDIO0 GNDIO7	-			GNDIO0 GNDIO7	-		

*Supports true LVDS outputs.
**Double bonded to the pin.
***NC for "E" devices.
****Primary clock inputs are single-ended.

LCMXO256 and LCMXO640 Logic Signal Connections: 100 csBGA

LCMXO256					LCMXO640				
Ball Number	Ball Function	Bank	Dual Function	Differen-tial	Ball Number	Ball Function	Bank	Dual Function	Differen-tial
B1	PL2A	1		T	B1	PL2A	3		T
C1	PL2B	1		C	C1	PL2C	3		T
D2	PL3A	1		T	D2	PL2B	3		C
D1	PL3B	1		C	D1	PL2D	3		C
C2	PL3C	1		T	C2	PL3A	3		T
E1	PL3D	1		C	E1	PL3B	3		C
E2	PL4A	1		T	E2	PL3C	3		T
F1	PL4B	1		C	F1	PL3D	3		C
F2	PL5A	1		T	F2	PL4A	3		
G2	PL5B	1		C	G2	PL4C	3		T
H1	GNDIO1	1			H1	GNDIO3	3		
H2	PL5C	1		T	H2	PL4D	3		C
J1	PL5D	1	GSRN	C	J1	PL5B	3	GSRN	
J2	PL6A	1		T	J2	PL7B	3		
K1	PL6B	1	TSALL	C	K1	PL8C	3	TSALL	T
K2	PL7A	1		T	K2	PL8D	3		C
L1	PL7B	1		C	L1	PL9A	3		
L2	PL7C	1		T	L2	PL9C	3		
M1	PL7D	1		C	M1	PL10A	3		
M2	PL8A	1		T	M2	PL10C	3		
N1	PL8B	1		C	N1	PL11A	3		
M3	PL9A	1		T	M3	PL11C	3		
N2	GNDIO1	1			N2	GNDIO3	3		
P2	TMS	1	TMS		P2	TMS	2	TMS	
P3	PL9B	1		C	P3	PB2C	2		
N4	TCK	1	TCK		N4	TCK	2	TCK	
P4	PB2A	1		T	P4	VCCIO2	2		
N3	PB2B	1		C	N3	GNDIO2	2		
P5	TDO	1	TDO		P5	TDO	2	TDO	
N5	PB2C	1		T	N5	PB4C	2		
P6	TDI	1	TDI		P6	TDI	2	TDI	
N6	PB2D	1		C	N6	PB4E	2		
P7	VCC	-			P7	VCC	-		
N7	PB3A	1	PCLK1_1**	T	N7	PB5B	2	PCLK2_1**	
P8	PB3B	1		C	P8	PB5D	2		
N8	PB3C	1	PCLK1_0**	T	N8	PB6B	2	PCLK2_0**	
P9	PB3D	1		C	P9	PB6C	2		
N10	GNDIO1	1			N10	GNDIO2	2		
P11	PB4A	1		T	P11	PB8B	2		
N11	PB4B	1		C	N11	PB8C	2		T
P12	PB4C	1		T	P12	PB8D	2		C
N12	PB4D	1		C	N12	PB9A	2		

LCMXO256 and LCMXO640 Logic Signal Connections: 100 csBGA (Cont.)

LCMXO256					LCMXO640				
Ball Number	Ball Function	Bank	Dual Function	Differen-tial	Ball Number	Ball Function	Bank	Dual Function	Differen-tial
P13	PB5A	1			P13	PB9C	2		T
M12*	SLEEPN	-	SLEEPN		M12*	SLEEPN	-	SLEEPN	
P14	PB5C	1		T	P14	PB9D	2		C
N13	PB5D	1		C	N13	PB9F	2		
N14	PR9B	0		C	N14	PR11D	1		C
M14	PR9A	0		T	M14	PR11B	1		C
L13	PR8B	0		C	L13	PR11C	1		T
L14	PR8A	0		T	L14	PR11A	1		T
M13	PR7D	0		C	M13	PR10D	1		C
K14	PR7C	0		T	K14	PR10C	1		T
K13	PR7B	0		C	K13	PR10B	1		C
J14	PR7A	0		T	J14	PR10A	1		T
J13	PR6B	0		C	J13	PR9D	1		
H13	PR6A	0		T	H13	PR9B	1		
G14	GNDIO0	0			G14	GNDIO1	1		
G13	PR5D	0		C	G13	PR7B	1		
F14	PR5C	0		T	F14	PR6C	1		
F13	PR5B	0		C	F13	PR6B	1		
E14	PR5A	0		T	E14	PR5D	1		
E13	PR4B	0		C	E13	PR5B	1		
D14	PR4A	0		T	D14	PR4D	1		
D13	PR3D	0		C	D13	PR4B	1		
C14	PR3C	0		T	C14	PR3D	1		
C13	PR3B	0		C	C13	PR3B	1		
B14	PR3A	0		T	B14	PR2D	1		
C12	PR2B	0		C	C12	PR2B	1		
B13	GNDIO0	0			B13	GNDIO1	1		
A13	PR2A	0		T	A13	PT9F	0		C
A12	PT5C	0			A12	PT9E	0		T
B11	PT5B	0		C	B11	PT9C	0		
A11	PT5A	0		T	A11	PT9A	0		
B12	PT4F	0		C	B12	VCCIO0	0		
A10	PT4E	0		T	A10	GNDIO0	0		
B10	PT4D	0		C	B10	PT7E	0		
A9	PT4C	0		T	A9	PT7A	0		
A8	PT4B	0	PCLK0_1**	C	A8	PT6B	0	PCLK0_1**	
B8	PT4A	0	PCLK0_0**	T	B8	PT5B	0	PCLK0_0**	C
A7	PT3D	0		C	A7	PT5A	0		T
B7	VCCAUX	-			B7	VCCAUX	-		
A6	PT3C	0		T	A6	PT4F	0		
B6	VCC	-			B6	VCC	-		
A5	PT3B	0		C	A5	PT3F	0		

LCMXO256 and LCMXO640 Logic Signal Connections: 100 csBGA (Cont.)

LCMXO256					LCMXO640				
Ball Number	Ball Function	Bank	Dual Function	Differen-tial	Ball Number	Ball Function	Bank	Dual Function	Differen-tial
A4	GNDIO0	0			A4	GNDIO0	0		
B4	PT3A	0		T	B4	PT3B	0		C
A3	PT2F	0		C	A3	PT3A	0		T
B3	PT2E	0		T	B3	PT2F	0		C
A2	PT2D	0		C	A2	PT2E	0		T
C3	PT2C	0		T	C3	PT2B	0		C
A1	PT2B	0		C	A1	PT2C	0		
B2	PT2A	0		T	B2	PT2A	0		T
N9	GND	-			N9	GND	-		
B9	GND	-			B9	GND	-		
B5	VCCIO0	0			B5	VCCIO0	0		
A14	VCCIO0	0			A14	VCCIO1	1		
H14	VCCIO0	0			H14	VCCIO1	1		
P10	VCCIO1	1			P10	VCCIO2	2		
G1	VCCIO1	1			G1	VCCIO3	3		
P1	VCCIO1	1			P1	VCCIO3	3		

*NC for “E” devices.
**Primary clock inputs are single-ended.

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 132 csBGA

LCMXO640					LCMXO1200					LCMXO2280				
Ball #	Ball Function	Bank	Dual Function	Differential	Ball #	Ball Function	Bank	Dual Function	Differential	Ball #	Ball Function	Bank	Dual Function	Differential
B1	PL2A	3		T	B1	PL2A	7		T	B1	PL2A	7	LUM0_PLLT_FB_A	T
C1	PL2B	3		C	C1	PL3C	7		T	C1	PL3C	7	LUM0_PLLT_IN_A	T
B2	PL2C	3		T	B2	PL2B	7		C	B2	PL2B	7	LUM0_PLLC_FB_A	C
C2	PL2D	3		C	C2	PL4A	7		T*	C2	PL4A	7		T*
C3	PL3A	3		T	C3	PL3D	7		C	C3	PL3D	7	LUM0_PLLC_IN_A	C
D1	PL3B	3		C	D1	PL4B	7		C*	D1	PL4B	7		C*
D3	PL3D	3			D3	PL4C	7			D3	PL4C	7		
E1	GNDIO3	3			E1	GNDIO7	7			E1	GNDIO7	7		
E2	PL5A	3		T	E2	PL6A	7		T*	E2	PL7A	7		T*
E3	PL5B	3	GSRN	C	E3	PL6B	7	GSRN	C*	E3	PL7B	7	GSRN	C*
F2	PL5D	3			F2	PL6D	7			F2	PL7D	7		
F3	PL6B	3			F3	PL7C	7		T	F3	PL9C	7		T
G1	PL6C	3		T	G1	PL7D	7		C	G1	PL9D	7		C
G2	PL6D	3		C	G2	PL8C	7		T	G2	PL10C	7		T
G3	PL7A	3		T	G3	PL8D	7		C	G3	PL10D	7		C
H2	PL7B	3		C	H2	PL10A	6		T*	H2	PL12A	6		T*
H1	PL7C	3			H1	PL10B	6		C*	H1	PL12B	6		C*
H3	VCC	-			H3	VCC	-			H3	VCC	-		
J1	PL8A	3			J1	PL11B	6			J1	PL14D	6		C
J2	PL8C	3	TSALL		J2	PL11C	6	TSALL	T	J2	PL14C	6	TSALL	T
J3	PL9A	3		T	J3	PL11D	6		C	J3	PL14B	6		
K2	PL9B	3		C	K2	PL12A	6		T*	K2	PL15A	6		T*
K1	PL9C	3			K1	PL12B	6		C*	K1	PL15B	6		C*
L2	GNDIO3	3			L2	GNDIO6	6			L2	GNDIO6	6		
L1	PL10A	3		T	L1	PL14A	6	LLM0_PLLT_FB_A	T*	L1	PL17A	6	LLM0_PLLT_FB_A	T*
L3	PL10B	3		C	L3	PL14B	6	LLM0_PLLC_FB_A	C*	L3	PL17B	6	LLM0_PLLC_FB_A	C*
M1	PL11A	3		T	M1	PL15A	6	LLM0_PLLT_IN_A	T*	M1	PL18A	6	LLM0_PLLT_IN_A	T*
N1	PL11B	3		C	N1	PL16A	6		T	N1	PL19A	6		T
M2	PL11C	3		T	M2	PL15B	6	LLM0_PLLC_IN_A	C*	M2	PL18B	6	LLM0_PLLC_IN_A	C*
P1	PL11D	3		C	P1	PL16B	6		C	P1	PL19B	6		C
P2	GNDIO2	2			P2	GNDIO5	5			P2	GNDIO5	5		
P3	TMS	2	TMS		P3	TMS	5	TMS		P3	TMS	5	TMS	
M3	PB2C	2		T	M3	PB2C	5		T	M3	PB2A	5		T
N3	PB2D	2		C	N3	PB2D	5		C	N3	PB2B	5		C
P4	TCK	2	TCK		P4	TCK	5	TCK		P4	TCK	5	TCK	
M4	PB3B	2			M4	PB3B	5			M4	PB3B	5		
N4	PB3C	2		T	N4	PB4A	5		T	N4	PB4A	5		T
P5	PB3D	2		C	P5	PB4B	5		C	P5	PB4B	5		C
N5	TDO	2	TDO		N5	TDO	5	TDO		N5	TDO	5	TDO	
M5	TDI	2	TDI		M5	TDI	5	TDI		M5	TDI	5	TDI	
N6	PB4E	2		T	N6	PB5C	5			N6	PB6C	5		
P6	VCC	-			P6	VCC	-			P6	VCC	-		
M6	PB4F	2		C	M6	PB6A	5			M6	PB8A	5		
P7	VCCAUX	-			P7	VCCAUX	-			P7	VCCAUX	-		
N7	PB5A	2		T	N7	PB6F	5			N7	PB8F	5		
M7	PB5B	2	PCLK2_1***	C	M7	PB7B	4	PCLK4_1***		M7	PB10F	4	PCLK4_1***	
N8	PB5D	2			N8	PB7C	4		T	N8	PB10C	4		T
P8	PB6A	2		T	P8	PB7D	4		C	P8	PB10D	4		C
M8	PB6B	2	PCLK2_0***	C	M8	PB7F	4	PCLK4_0***		M8	PB10B	4	PCLK4_0***	
N9	PB7A	2		T	N9	PB9A	4		T	N9	PB12A	4		T

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 132 csBGA (Cont.)

LCMXO640					LCMXO1200					LCMXO2280				
Ball #	Ball Function	Bank	Dual Function	Differential	Ball #	Ball Function	Bank	Dual Function	Differential	Ball #	Ball Function	Bank	Dual Function	Differential
M9	PB7B	2		C	M9	PB9B	4		C	M9	PB12B	4		C
N10	PB7E	2		T	N10	PB9C	4		T	N10	PB12C	4		T
P10	PB7F	2		C	P10	PB9D	4		C	P10	PB12D	4		C
N11	GNDIO2	2			N11	GNDIO4	4			N11	GNDIO4	4		
P11	PB8C	2		T	P11	PB10A	4		T	P11	PB13C	4		T
M11	PB8D	2		C	M11	PB10B	4		C	M11	PB13D	4		C
P12	PB9C	2		T	P12	PB10C	4			P12	PB15B	4		
P13	PB9D	2		C	P13	PB11C	4		T	P13	PB16C	4		T
N12**	SLEEPN	-	SLEEPN		N12**	SLEEPN	-	SLEEPN		N12**	SLEEPN	-	SLEEPN	
P14	PB9F	2			P14	PB11D	4		C	P14	PB16D	4		C
N14	PR11D	1		C	N14	PR16B	3		C	N14	PR19B	3		C
M14	PR11C	1		T	M14	PR15B	3		C*	M14	PR18B	3		C*
N13	PR11B	1		C	N13	PR16A	3		T	N13	PR19A	3		T
M12	PR11A	1		T	M12	PR15A	3		T*	M12	PR18A	3		T*
M13	PR10B	1		C	M13	PR14B	3		C*	M13	PR17B	3		C*
L14	PR10A	1		T	L14	PR14A	3		T*	L14	PR17A	3		T*
L13	GNDIO1	1			L13	GNDIO3	3			L13	GNDIO3	3		
K14	PR8D	1		C	K14	PR12B	3		C*	K14	PR15B	3		C*
K13	PR8C	1		T	K13	PR12A	3		T*	K13	PR15A	3		T*
K12	PR8B	1		C	K12	PR11B	3		C*	K12	PR14B	3		C*
J13	PR8A	1		T	J13	PR11A	3		T*	J13	PR14A	3		T*
J12	PR7C	1			J12	PR10B	3		C*	J12	PR13B	3		C*
H14	PR7B	1		C	H14	PR10A	3		T*	H14	PR13A	3		T*
H13	PR7A	1		T	H13	PR9B	3		C*	H13	PR11B	3		C*
H12	PR6D	1		C	H12	PR9A	3		T*	H12	PR11A	3		T*
G13	PR6C	1		T	G13	PR8B	2		C*	G13	PR10B	2		C*
G14	PR6B	1			G14	PR8A	2		T*	G14	PR10A	2		T*
G12	VCC	-			G12	VCC	-			G12	VCC	-		
F14	PR5D	1		C	F14	PR6C	2			F14	PR8C	2		
F13	PR5C	1		T	F13	PR6B	2		C*	F13	PR8B	2		C*
F12	PR4D	1		C	F12	PR6A	2		T*	F12	PR8A	2		T*
E13	PR4C	1		T	E13	PR5B	2		C*	E13	PR7B	2		C*
E14	PR4B	1			E14	PR5A	2		T*	E14	PR7A	2		T*
D13	GNDIO1	1			D13	GNDIO2	2			D13	GNDIO2	2		
D14	PR3D	1		C	D14	PR4B	2		C*	D14	PR5B	2		C*
D12	PR3C	1		T	D12	PR4A	2		T*	D12	PR5A	2		T*
C14	PR2D	1		C	C14	PR3D	2		C	C14	PR4D	2		C
B14	PR2C	1		T	B14	PR2B	2		C	B14	PR3B	2		C*
C13	PR2B	1		C	C13	PR3C	2		T	C13	PR4C	2		T
A14	PR2A	1		T	A14	PR2A	2		T	A14	PR3A	2		T*
A13	PT9F	0		C	A13	PT11D	1		C	A13	PT16D	1		C
A12	PT9E	0		T	A12	PT11B	1		C	A12	PT16B	1		C
B13	PT9D	0		C	B13	PT11C	1		T	B13	PT16C	1		T
B12	PT9C	0		T	B12	PT10F	1			B12	PT15D	1		
C12	PT9B	0		C	C12	PT11A	1		T	C12	PT16A	1		T
A11	PT9A	0		T	A11	PT10D	1		C	A11	PT14B	1		C
C11	PT8C	0			C11	PT10C	1		T	C11	PT14A	1		T
A10	GNDIO0	0			A10	GNDIO1	1			A10	GNDIO1	1		
B10	PT7F	0		C	B10	PT9F	1		C	B10	PT12F	1		C
C10	PT7E	0		T	C10	PT9E	1		T	C10	PT12E	1		T

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 132 csBGA (Cont.)

LCMXO640					LCMXO1200					LCMXO2280				
Ball #	Ball Function	Bank	Dual Function	Differential	Ball #	Ball Function	Bank	Dual Function	Differential	Ball #	Ball Function	Bank	Dual Function	Differential
B9	PT7B	0		C	B9	PT9B	1		C	B9	PT12D	1		C
A9	PT7A	0		T	A9	PT9A	1		T	A9	PT12C	1		T
A8	PT6B	0	PCLK0_1***	C	A8	PT7D	1	PCLK1_1***		A8	PT10B	1	PCLK1_1***	
B8	PT6A	0		T	B8	PT7B	1			B8	PT9D	1		
C8	PT5B	0	PCLK0_0***	C	C8	PT6F	0	PCLK1_0***		C8	PT9B	1	PCLK1_0***	
B7	PT5A	0		T	B7	PT6D	0			B7	PT8D	0		
A7	VCCAUX	-			A7	VCCAUX	-			A7	VCCAUX	-		
C7	VCC	-			C7	VCC	-			C7	VCC	-		
A6	PT4D	0		C	A6	PT5D	0		C	A6	PT7B	0		C
B6	PT4C	0		T	B6	PT5C	0		T	B6	PT7A	0		T
C6	PT3F	0		C	C6	PT5B	0		C	C6	PT6D	0		
B5	PT3E	0		T	B5	PT5A	0		T	B5	PT6E	0		T
A5	PT3D	0			A5	PT4B	0			A5	PT6F	0		C
B4	GNDIO0	0			B4	GNDIO0	0			B4	GNDIO0	0		
A4	PT3B	0			A4	PT3D	0		C	A4	PT4B	0		C
C4	PT2F	0			C4	PT3C	0		T	C4	PT4A	0		T
A3	PT2D	0		C	A3	PT3B	0		C	A3	PT3B	0		C
A2	PT2C	0		T	A2	PT2B	0		C	A2	PT2B	0		C
B3	PT2B	0		C	B3	PT3A	0		T	B3	PT3A	0		T
A1	PT2A	0		T	A1	PT2A	0		T	A1	PT2A	0		T
F1	GND	-			F1	GND	-			F1	GND	-		
P9	GND	-			P9	GND	-			P9	GND	-		
J14	GND	-			J14	GND	-			J14	GND	-		
C9	GND	-			C9	GND	-			C9	GND	-		
C5	VCCIO0	0			C5	VCCIO0	0			C5	VCCIO0	0		
B11	VCCIO0	0			B11	VCCIO1	1			B11	VCCIO1	1		
E12	VCCIO1	1			E12	VCCIO2	2			E12	VCCIO2	2		
L12	VCCIO1	1			L12	VCCIO3	3			L12	VCCIO3	3		
M10	VCCIO2	2			M10	VCCIO4	4			M10	VCCIO4	4		
N2	VCCIO2	2			N2	VCCIO5	5			N2	VCCIO5	5		
D2	VCCIO3	3			D2	VCCIO7	7			D2	VCCIO7	7		
K3	VCCIO3	3			K3	VCCIO6	6			K3	VCCIO6	6		

*Supports true LVDS outputs.
**NC for “E” devices.
***Primary clock inputs arer single-ended.

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 144 TQFP

Pin Number	LCMXO640				LCMXO1200				LCMXO2280			
	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
1	PL2A	3		T	PL2A	7		T	PL2A	7	LUM0_PLLT_FB_A	T
2	PL2C	3		T	PL2B	7		C	PL2B	7	LUM0_PLLC_FB_A	C
3	PL2B	3		C	PL3A	7		T*	PL3A	7		T*
4	PL3A	3		T	PL3B	7		C*	PL3B	7		C*
5	PL2D	3		C	PL3C	7		T	PL3C	7	LUM0_PLLT_IN_A	T
6	PL3B	3		C	PL3D	7		C	PL3D	7	LUM0_PLLC_IN_A	C
7	PL3C	3		T	PL4A	7		T*	PL4A	7		T*
8	PL3D	3		C	PL4B	7		C*	PL4B	7		C*
9	PL4A	3			PL4C	7			PL4C	7		
10	VCCIO3	3			VCCIO7	7			VCCIO7	7		
11	GNDIO3	3			GNDIO7	7			GNDIO7	7		
12	PL4D	3			PL5C	7			PL6C	7		
13	PL5A	3		T	PL6A	7		T*	PL7A	7		T*
14	PL5B	3	GSRN	C	PL6B	7	GSRN	C*	PL7B	7	GSRN	C*
15	PL5D	3			PL6D	7			PL7D	7		
16	GND	-			GND	-			GND	-		
17	PL6C	3		T	PL7C	7		T	PL9C	7		T
18	PL6D	3		C	PL7D	7		C	PL9D	7		C
19	PL7A	3		T	PL10A	6		T*	PL13A	6		T*
20	PL7B	3		C	PL10B	6		C*	PL13B	6		C*
21	VCC	-			VCC	-			VCC	-		
22	PL8A	3		T	PL11A	6		T*	PL13D	6		
23	PL8B	3		C	PL11B	6		C*	PL14D	6		C
24	PL8C	3	TSALL		PL11C	6	TSALL		PL14C	6	TSALL	T
25	PL9C	3		T	PL12B	6			PL15B	6		
26	VCCIO3	3			VCCIO6	6			VCCIO6	6		
27	GNDIO3	3			GNDIO6	6			GNDIO6	6		
28	PL9D	3		C	PL13D	6			PL16D	6		
29	PL10A	3		T	PL14A	6	LLM0_PLLT_FB_A	T*	PL17A	6	LLM0_PLLT_FB_A	T*
30	PL10B	3		C	PL14B	6	LLM0_PLLC_FB_A	C*	PL17B	6	LLM0_PLLC_FB_A	C*
31	PL10C	3		T	PL14C	6		T	PL17C	6		T
32	PL11A	3		T	PL14D	6		C	PL17D	6		C
33	PL10D	3		C	PL15A	6	LLM0_PLLT_IN_A	T*	PL18A	6	LLM0_PLLT_IN_A	T*
34	PL11C	3		T	PL15B	6	LLM0_PLLC_IN_A	C*	PL18B	6	LLM0_PLLC_IN_A	C*
35	PL11B	3		C	PL16A	6		T	PL19A	6		T
36	PL11D	3		C	PL16B	6		C	PL19B	6		C
37	GNDIO2	2			GNDIO5	5			GNDIO5	5		
38	VCCIO2	2			VCCIO5	5			VCCIO5	5		
39	TMS	2	TMS		TMS	5	TMS		TMS	5	TMS	
40	PB2C	2			PB2C	5		T	PB2A	5		T
41	PB3A	2		T	PB2D	5		C	PB2B	5		C
42	TCK	2	TCK		TCK	5	TCK		TCK	5	TCK	
43	PB3B	2		C	PB3A	5		T	PB3A	5		T
44	PB3C	2		T	PB3B	5		C	PB3B	5		C
45	PB3D	2		C	PB4A	5		T	PB4A	5		T
46	PB4A	2		T	PB4B	5		C	PB4B	5		C
47	TDO	2	TDO		TDO	5	TDO		TDO	5	TDO	
48	PB4B	2		C	PB4D	5			PB4D	5		
49	PB4C	2		T	PB5A	5		T	PB5A	5		T
50	PB4D	2		C	PB5B	5		C	PB5B	5		C

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 144 TQFP (Cont.)

Pin Number	LCMXO640				LCMXO1200				LCMXO2280			
	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
51	TDI	2	TDI		TDI	5	TDI		TDI	5	TDI	
52	VCC	-			VCC	-			VCC	-		
53	VCCAUX	-			VCCAUX	-			VCCAUX	-		
54	PB5A	2		T	PB6F	5			PB8F	5		
55	PB5B	2	PCLKT2_1***	C	PB7B	4	PCLK4_1***		PB10F	4	PCLK4_1***	
56	PB5D	2			PB7C	4		T	PB10C	4		T
57	PB6A	2		T	PB7D	4		C	PB10D	4		C
58	PB6B	2	PCLKT2_0***	C	PB7F	4	PCLK4_0***		PB10B	4	PCLK4_0***	
59	GND	-			GND	-			GND	-		
60	PB7C	2			PB9A	4		T	PB12A	4		T
61	PB7E	2			PB9B	4		C	PB12B	4		C
62	PB8A	2			PB9E	4			PB12E	4		
63	VCCIO2	2			VCCIO4	4			VCCIO4	4		
64	GNDIO2	2			GNDIO4	4			GNDIO4	4		
65	PB8C	2		T	PB10A	4		T	PB13A	4		T
66	PB8D	2		C	PB10B	4		C	PB13B	4		C
67	PB9A	2		T	PB10C	4		T	PB13C	4		T
68	PB9C	2		T	PB10D	4		C	PB13D	4		C
69	PB9B	2		C	PB10F	4			PB14D	4		
70**	SLEEPN	-	SLEEPN		SLEEPN	-	SLEEPN		SLEEPN	-	SLEEPN	
71	PB9D	2		C	PB11C	4		T	PB16C	4		T
72	PB9F	2			PB11D	4		C	PB16D	4		C
73	PR11D	1		C	PR16B	3		C	PR20B	3		C
74	PR11B	1		C	PR16A	3		T	PR20A	3		T
75	PR11C	1		T	PR15B	3		C*	PR19B	3		C
76	PR10D	1		C	PR15A	3		T*	PR19A	3		T
77	PR11A	1		T	PR14D	3		C	PR17D	3		C
78	PR10B	1		C	PR14C	3		T	PR17C	3		T
79	PR10C	1		T	PR14B	3		C*	PR17B	3		C*
80	PR10A	1		T	PR14A	3		T*	PR17A	3		T*
81	PR9D	1			PR13D	3			PR16D	3		
82	VCCIO1	1			VCCIO3	3			VCCIO3	3		
83	GNDIO1	1			GNDIO3	3			GNDIO3	3		
84	PR9A	1			PR12B	3		C*	PR15B	3		C*
85	PR8C	1			PR12A	3		T*	PR15A	3		T*
86	PR8A	1			PR11B	3		C*	PR14B	3		C*
87	PR7D	1			PR11A	3		T*	PR14A	3		T*
88	GND	-			GND	-			GND	-		
89	PR7B	1		C	PR10B	3		C*	PR13B	3		C*
90	PR7A	1		T	PR10A	3		T*	PR13A	3		T*
91	PR6D	1		C	PR8B	2		C*	PR10B	2		C*
92	PR6C	1		T	PR8A	2		T*	PR10A	2		T*
93	VCC	-			VCC	-			VCC	-		
94	PR5D	1			PR6B	2		C*	PR8B	2		C*
95	PR5B	1			PR6A	2		T*	PR8A	2		T*
96	PR4D	1			PR5B	2		C*	PR7B	2		C*
97	PR4B	1		C	PR5A	2		T*	PR7A	2		T*
98	VCCIO1	1			VCCIO2	2			VCCIO2	2		
99	GNDIO1	1			GNDIO2	2			GNDIO2	2		
100	PR4A	1		T	PR4C	2			PR5C	2		

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 144 TQFP (Cont.)

Pin Number	LCMXO640				LCMXO1200				LCMXO2280			
	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential	Ball Function	Bank	Dual Function	Differential
101	PR3D	1		C	PR4B	2		C*	PR5B	2		C*
102	PR3C	1		T	PR4A	2		T*	PR5A	2		T*
103	PR3B	1		C	PR3D	2		C	PR4D	2		C
104	PR2D	1		C	PR3C	2		T	PR4C	2		T
105	PR3A	1		T	PR3B	2		C*	PR4B	2		C*
106	PR2B	1		C	PR3A	2		T*	PR4A	2		T*
107	PR2C	1		T	PR2B	2		C	PR3B	2		C*
108	PR2A	1		T	PR2A	2		T	PR3A	2		T*
109	PT9F	0		C	PT11D	1		C	PT16D	1		C
110	PT9D	0		C	PT11C	1		T	PT16C	1		T
111	PT9E	0		T	PT11B	1		C	PT16B	1		C
112	PT9B	0		C	PT11A	1		T	PT16A	1		T
113	PT9C	0		T	PT10F	1		C	PT15D	1		C
114	PT9A	0		T	PT10E	1		T	PT15C	1		T
115	PT8C	0			PT10D	1		C	PT14B	1		C
116	PT8B	0		C	PT10C	1		T	PT14A	1		T
117	VCCIO0	0			VCCIO1	1			VCCIO1	1		
118	GNDIO0	0			GNDIO1	1			GNDIO1	1		
119	PT8A	0		T	PT9F	1		C	PT12F	1		C
120	PT7E	0			PT9E	1		T	PT12E	1		T
121	PT7C	0			PT9B	1		C	PT12D	1		C
122	PT7A	0			PT9A	1		T	PT12C	1		T
123	GND	-			GND	-			GND	-		
124	PT6B	0	PCLK0_1***	C	PT7D	1	PCLK1_1***		PT10B	1	PCLK1_1***	
125	PT6A	0		T	PT7B	1		C	PT9D	1		C
126	PT5C	0			PT7A	1		T	PT9C	1		T
127	PT5B	0	PCLK0_0***		PT6F	0	PCLK1_0***		PT9B	1	PCLK1_0***	
128	VCCAUX	-			VCCAUX	-			VCCAUX	-		
129	VCC	-			VCC	-			VCC	-		
130	PT4D	0			PT5D	0		C	PT7B	0		C
131	PT4B	0		C	PT5C	0		T	PT7A	0		T
132	PT4A	0		T	PT5B	0		C	PT6D	0		
133	PT3F	0			PT5A	0		T	PT6E	0		T
134	PT3D	0			PT4B	0			PT6F	0		C
135	VCCIO0	0			VCCIO0	0			VCCIO0	0		
136	GNDIO0	0			GNDIO0	0			GNDIO0	0		
137	PT3B	0		C	PT3D	0		C	PT4B	0		T
138	PT2F	0		C	PT3C	0		T	PT4A	0		C
139	PT3A	0		T	PT3B	0		C	PT3B	0		C
140	PT2D	0		C	PT3A	0		T	PT3A	0		T
141	PT2E	0		T	PT2D	0		C	PT2D	0		C
142	PT2B	0		C	PT2C	0		T	PT2C	0		T
143	PT2C	0		T	PT2B	0		C	PT2B	0		C
144	PT2A	0		T	PT2A	0		T	PT2A	0		T

*Supports true LVDS outputs.
**NC for “E” devices.
***Primary clock inputs arer single-ended.

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 256 caBGA / 256 ftBGA

LCMXO640					LCMXO1200					LCMXO2280				
Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential
GND	GNDIO3	3			GND	GNDIO7	7			GND	GNDIO7	7		
VCCIO3	VCCIO3	3			VCCIO7	VCCIO7	7			VCCIO7	VCCIO7	7		
E4	NC				E4	PL2A	7		T	E4	PL2A	7	LUM0_PLLT_FB_A	T
E5	NC				E5	PL2B	7		C	E5	PL2B	7	LUM0_PLLC_FB_A	C
F5	NC				F5	PL3A	7		T*	F5	PL3A	7		T*
F6	NC				F6	PL3B	7		C*	F6	PL3B	7		C*
F3	PL3A	3		T	F3	PL3C	7		T	F3	PL3C	7	LUM0_PLLT_IN_A	T
F4	PL3B	3		C	F4	PL3D	7		C	F4	PL3D	7	LUM0_PLLC_IN_A	C
E3	PL2C	3		T	E3	PL4A	7		T*	E3	PL4A	7		T*
E2	PL2D	3		C	E2	PL4B	7		C*	E2	PL4B	7		C*
C3	NC				C3	PL4C	7		T	C3	PL4C	7		T
C2	NC				C2	PL4D	7		C	C2	PL4D	7		C
B1	PL2A	3		T	B1	PL5A	7		T*	B1	PL5A	7		T*
C1	PL2B	3		C	C1	PL5B	7		C*	C1	PL5B	7		C*
VCCIO3	VCCIO3	3			VCCIO7	VCCIO7	7			VCCIO7	VCCIO7	7		
GND	GNDIO3	3			GND	GNDIO7	7			GND	GNDIO7	7		
D2	PL3C	3		T	D2	PL5C	7		T	D2	PL6C	7		T
D1	PL3D	3		C	D1	PL5D	7		C	D1	PL6D	7		C
F2	PL5A	3		T	F2	PL6A	7		T*	F2	PL7A	7		T*
G2	PL5B	3	GSRN	C	G2	PL6B	7	GSRN	C*	G2	PL7B	7	GSRN	C*
E1	PL4A	3		T	E1	PL6C	7		T	E1	PL7C	7		T
F1	PL4B	3		C	F1	PL6D	7		C	F1	PL7D	7		C
G4	NC				G4	PL7A	7		T*	G4	PL8A	7		T*
G5	NC				G5	PL7B	7		C*	G5	PL8B	7		C*
GND	GND	-			GND	GND	-			GND	GND	-		
G3	PL4C	3		T	G3	PL7C	7		T	G3	PL8C	7		T
H3	PL4D	3		C	H3	PL7D	7		C	H3	PL8D	7		C
H4	NC				H4	PL8A	7		T*	H4	PL9A	7		T*
H5	NC				H5	PL8B	7		C*	H5	PL9B	7		C*
-	-				VCCIO7	VCCIO7	7			VCCIO7	VCCIO7	7		
-	-				GND	GNDIO7	7			GND	GNDIO7	7		
G1	PL5C	3		T	G1	PL8C	7		T	G1	PL10C	7		T
H1	PL5D	3		C	H1	PL8D	7		C	H1	PL10D	7		C
H2	PL6A	3		T	H2	PL9A	6		T*	H2	PL11A	6		T*
J2	PL6B	3		C	J2	PL9B	6		C*	J2	PL11B	6		C*
J3	PL7C	3		T	J3	PL9C	6		T	J3	PL11C	6		T
K3	PL7D	3		C	K3	PL9D	6		C	K3	PL11D	6		C
J1	PL6C	3		T	J1	PL10A	6		T*	J1	PL12A	6		T*
-	-				VCCIO6	VCCIO6	6			VCCIO6	VCCIO6	6		
-	-				GND	GNDIO6	6			GND	GNDIO6	6		
K1	PL6D	3		C	K1	PL10B	6		C*	K1	PL12B	6		C*
K2	PL9A	3		T	K2	PL10C	6		T	K2	PL12C	6		T
L2	PL9B	3		C	L2	PL10D	6		C	L2	PL12D	6		C
L1	PL7A	3		T	L1	PL11A	6		T*	L1	PL13A	6		T*
M1	PL7B	3		C	M1	PL11B	6		C*	M1	PL13B	6		C*
P1	PL8D	3		C	P1	PL11D	6		C	P1	PL14D	6		C
N1	PL8C	3	TSALL	T	N1	PL11C	6	TSALL	T	N1	PL14C	6	TSALL	T
L3	PL10A	3		T	L3	PL12A	6		T*	L3	PL15A	6		T*
M3	PL10B	3		C	M3	PL12B	6		C*	M3	PL15B	6		C*
M2	PL9C	3		T	M2	PL12C	6		T	M2	PL15C	6		T
N2	PL9D	3		C	N2	PL12D	6		C	N2	PL15D	6		C
VCCIO3	VCCIO3	3			VCCIO6	VCCIO6	6			VCCIO6	VCCIO6	6		
GND	GNDIO3	3			GND	GNDIO6	6			GND	GNDIO6	6		

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 256 caBGA / 256 ftBGA (Cont.)

LCMXO640					LCMXO1200					LCMXO2280				
Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential
J4	PL8A	3		T	J4	PL13A	6		T*	J4	PL16A	6		T*
J5	PL8B	3		C	J5	PL13B	6		C*	J5	PL16B	6		C*
R1	PL11A	3		T	R1	PL13C	6		T	R1	PL16C	6		T
R2	PL11B	3		C	R2	PL13D	6		C	R2	PL16D	6		C
-	-	-			-	-	-			GND	GND	-		
K5	NC				K5	PL14A	6	LLM0_PLLT_FB_A	T*	K5	PL17A	6	LLM0_PLLT_FB_A	T*
K4	NC				K4	PL14B	6	LLM0_PLLC_FB_A	C*	K4	PL17B	6	LLM0_PLLC_FB_A	C*
L5	PL10C	3		T	L5	PL14C	6		T	L5	PL17C	6		T
L4	PL10D	3		C	L4	PL14D	6		C	L4	PL17D	6		C
M5	NC				M5	PL15A	6	LLM0_PLLT_IN_A	T*	M5	PL18A	6	LLM0_PLLT_IN_A	T*
M4	NC				M4	PL15B	6	LLM0_PLLC_IN_A	C*	M4	PL18B	6	LLM0_PLLC_IN_A	C*
N4	PL11C	3		T	N4	PL16A	6		T	N4	PL19A	6		T
N3	PL11D	3		C	N3	PL16B	6		C	N3	PL19B	6		C
VCCIO3	VCCIO3	3			VCCIO6	VCCIO6	6			VCCIO6	VCCIO6	6		
GND	GNDIO3	3			GND	GNDIO6	6			GND	GNDIO6	6		
GND	GNDIO2	2			GND	GNDIO5	5			GND	GNDIO5	5		
VCCIO2	VCCIO2	2			VCCIO5	VCCIO5	5			VCCIO5	VCCIO5	5		
P4	TMS	2	TMS		P4	TMS	5	TMS		P4	TMS	5	TMS	
P2	NC				P2	PB2A	5		T	P2	PB2A	5		T
P3	NC				P3	PB2B	5		C	P3	PB2B	5		C
N5	NC				N5	PB2C	5		T	N5	PB2C	5		T
R3	TCK	2	TCK		R3	TCK	5	TCK		R3	TCK	5	TCK	
N6	NC				N6	PB2D	5		C	N6	PB2D	5		C
T2	PB2A	2		T	T2	PB3A	5		T	T2	PB3A	5		T
T3	PB2B	2		C	T3	PB3B	5		C	T3	PB3B	5		C
R4	PB2C	2		T	R4	PB3C	5		T	R4	PB3C	5		T
R5	PB2D	2		C	R5	PB3D	5		C	R5	PB3D	5		C
P5	PB3A	2		T	P5	PB4A	5		T	P5	PB4A	5		T
P6	PB3B	2		C	P6	PB4B	5		C	P6	PB4B	5		C
T5	PB3C	2		T	T5	PB4C	5		T	T5	PB4C	5		T
M6	TDO	2	TDO		M6	TDO	5	TDO		M6	TDO	5	TDO	
T4	PB3D	2		C	T4	PB4D	5		C	T4	PB4D	5		C
R6	PB4A	2		T	R6	PB5A	5		T	R6	PB5A	5		T
GND	GNDIO2	2			GND	GNDIO5	5			GND	GNDIO5	5		
VCCIO2	VCCIO2	2			VCCIO5	VCCIO5	5			VCCIO5	VCCIO5	5		
T6	PB4B	2		C	T6	PB5B	5		C	T6	PB5B	5		C
N7	TDI	2	TDI		N7	TDI	5	TDI		N7	TDI	5	TDI	
T8	PB4C	2		T	T8	PB5C	5		T	T8	PB6A	5		T
T7	PB4D	2		C	T7	PB5D	5		C	T7	PB6B	5		C
M7	NC				M7	PB6A	5		T	M7	PB7C	5		T
M8	NC				M8	PB6B	5		C	M8	PB7D	5		C
T9	VCCAUX	-			T9	VCCAUX	-			T9	VCCAUX	-		
R7	PB4E	2		T	R7	PB6C	5		T	R7	PB8C	5		T
R8	PB4F	2		C	R8	PB6D	5		C	R8	PB8D	5		C
-	-				VCCIO5	VCCIO5	5			VCCIO5	VCCIO5	5		
-	-				GND	GNDIO5	5			GND	GNDIO5	5		
P7	PB5C	2		T	P7	PB6E	5		T	P7	PB9A	4		T
P8	PB5D	2		C	P8	PB6F	5		C	P8	PB9B	4		C
N8	PB5A	2		T	N8	PB7A	4		T	N8	PB10E	4		T
N9	PB5B	2	PCLK2_1***	C	N9	PB7B	4	PCLK4_1***	C	N9	PB10F	4	PCLK4_1***	C
P10	PB7B	2		C	P10	PB7D	4		C	P10	PB10D	4		C
P9	PB7A	2		T	P9	PB7C	4		T	P9	PB10C	4		T
M9	PB6B	2	PCLK2_0***	C	M9	PB7F	4	PCLK4_0***	C	M9	PB10B	4	PCLK4_0***	C

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 256 caBGA / 256 ftBGA (Cont.)

LCMXO640					LCMXO1200					LCMXO2280				
Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential
-	-				VCCIO4	VCCIO4	4			VCCIO4	VCCIO4	4		
-	-				GND	GNDIO4	4			GND	GNDIO4	4		
M10	PB6A	2		T	M10	PB7E	4		T	M10	PB10A	4		T
R9	PB6C	2		T	R9	PB8A	4		T	R9	PB11C	4		T
R10	PB6D	2		C	R10	PB8B	4		C	R10	PB11D	4		C
T10	PB7C	2		T	T10	PB8C	4		T	T10	PB12A	4		T
T11	PB7D	2		C	T11	PB8D	4		C	T11	PB12B	4		C
N10	NC				N10	PB8E	4		T	N10	PB12C	4		T
N11	NC				N11	PB8F	4		C	N11	PB12D	4		C
VCCIO2	VCCIO2	2			VCCIO4	VCCIO4	4			VCCIO4	VCCIO4	4		
GND	GNDIO2	2			GND	GNDIO4	4			GND	GNDIO4	4		
R11	PB7E	2		T	R11	PB9A	4		T	R11	PB13A	4		T
R12	PB7F	2		C	R12	PB9B	4		C	R12	PB13B	4		C
P11	PB8A	2		T	P11	PB9C	4		T	P11	PB13C	4		T
P12	PB8B	2		C	P12	PB9D	4		C	P12	PB13D	4		C
T13	PB8C	2		T	T13	PB9E	4		T	T13	PB14A	4		T
T12	PB8D	2		C	T12	PB9F	4		C	T12	PB14B	4		C
R13	PB9A	2		T	R13	PB10A	4		T	R13	PB14C	4		T
R14	PB9B	2		C	R14	PB10B	4		C	R14	PB14D	4		C
GND	GND	-			GND	GND	-			GND	GND	-		
T14	PB9C	2		T	T14	PB10C	4		T	T14	PB15A	4		T
T15	PB9D	2		C	T15	PB10D	4		C	T15	PB15B	4		C
P13**	SLEEPN	-	SLEEPN		P13**	SLEEPN	-	SLEEPN		P13**	SLEEPN	-	SLEEPN	
P14	PB9F	2			P14	PB10F	4			P14	PB15D	4		
R15	NC				R15	PB11A	4		T	R15	PB16A	4		T
R16	NC				R16	PB11B	4		C	R16	PB16B	4		C
P15	NC				P15	PB11C	4		T	P15	PB16C	4		T
P16	NC				P16	PB11D	4		C	P16	PB16D	4		C
VCCIO2	VCCIO2	2			VCCIO4	VCCIO4	4			VCCIO4	VCCIO4	4		
GND	GNDIO2	2			GND	GNDIO4	4			GND	GNDIO4	4		
GND	GNDIO1	1			GND	GNDIO3	3			GND	GNDIO3	3		
VCCIO1	VCCIO1	1			VCCIO3	VCCIO3	3			VCCIO3	VCCIO3	3		
M11	NC				M11	PR16B	3		C	M11	PR20B	3		C
L11	NC				L11	PR16A	3		T	L11	PR20A	3		T
N12	NC				N12	PR15B	3		C*	N12	PR18B	3		C*
N13	NC				N13	PR15A	3		T*	N13	PR18A	3		T*
M13	NC				M13	PR14D	3		C	M13	PR17D	3		C
M12	NC				M12	PR14C	3		T	M12	PR17C	3		T
N14	PR11D	1		C	N14	PR14B	3		C*	N14	PR17B	3		C*
N15	PR11C	1		T	N15	PR14A	3		T*	N15	PR17A	3		T*
L13	PR11B	1		C	L13	PR13D	3		C	L13	PR16D	3		C
L12	PR11A	1		T	L12	PR13C	3		T	L12	PR16C	3		T
M14	PR10B	1		C	M14	PR13B	3		C*	M14	PR16B	3		C*
VCCIO1	VCCIO1	1			VCCIO3	VCCIO3	3			VCCIO3	VCCIO3	3		
GND	GNDIO1	1			GND	GNDIO3	3			GND	GNDIO3	3		
L14	PR10A	1		T	L14	PR13A	3		T*	L14	PR16A	3		T*
N16	PR10D	1		C	N16	PR12D	3		C	N16	PR15D	3		C
M16	PR10C	1		T	M16	PR12C	3		T	M16	PR15C	3		T
M15	PR9D	1		C	M15	PR12B	3		C*	M15	PR15B	3		C*
L15	PR9C	1		T	L15	PR12A	3		T*	L15	PR15A	3		T*
L16	PR9B	1		C	L16	PR11D	3		C	L16	PR14D	3		C
K16	PR9A	1		T	K16	PR11C	3		T	K16	PR14C	3		T
K13	PR8D	1		C	K13	PR11B	3		C*	K13	PR14B	3		C*

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 256 caBGA / 256 ftBGA (Cont.)

LCMXO640					LCMXO1200					LCMXO2280				
Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential
J13	PR8C	1		T	J13	PR11A	3		T*	J13	PR14A	3		T*
GND	GND	-			GND	GND	-			GND	GND	-		
K14	PR8B	1		C	K14	PR10D	3		C	K14	PR13D	3		C
J14	PR8A	1		T	J14	PR10C	3		T	J14	PR13C	3		T
K15	PR7D	1		C	K15	PR10B	3		C*	K15	PR13B	3		C*
J15	PR7C	1		T	J15	PR10A	3		T*	J15	PR13A	3		T*
-	-				GND	GNDIO3	3			GND	GNDIO3	3		
-	-				VCCIO3	VCCIO3	3			VCCIO3	VCCIO3	3		
K12	NC				K12	PR9D	3		C	K12	PR11D	3		C
J12	NC				J12	PR9C	3		T	J12	PR11C	3		T
J16	PR7B	1		C	J16	PR9B	3		C*	J16	PR11B	3		C*
H16	PR7A	1		T	H16	PR9A	3		T*	H16	PR11A	3		T*
H15	PR6B	1		C	H15	PR8D	2		C	H15	PR10D	2		C
G15	PR6A	1		T	G15	PR8C	2		T	G15	PR10C	2		T
H14	PR5D	1		C	H14	PR8B	2		C*	H14	PR10B	2		C*
G14	PR5C	1		T	G14	PR8A	2		T*	G14	PR10A	2		T*
GND	GNDIO1	1			GND	GNDIO2	2			GND	GNDIO2	2		
VCCIO1	VCCIO1	1			VCCIO2	VCCIO2	2			VCCIO2	VCCIO2	2		
H13	PR6D	1		C	H13	PR7D	2		C	H13	PR9D	2		C
H12	PR6C	1		T	H12	PR7C	2		T	H12	PR9C	2		T
G13	PR4D	1		C	G13	PR7B	2		C*	G13	PR9B	2		C*
G12	PR4C	1		T	G12	PR7A	2		T*	G12	PR9A	2		T*
G16	PR5B	1		C	G16	PR6D	2		C	G16	PR7D	2		C
F16	PR5A	1		T	F16	PR6C	2		T	F16	PR7C	2		T
F15	PR4B	1		C	F15	PR6B	2		C*	F15	PR7B	2		C*
E15	PR4A	1		T	E15	PR6A	2		T*	E15	PR7A	2		T*
E16	PR3B	1		C	E16	PR5D	2		C	E16	PR6D	2		C
D16	PR3A	1		T	D16	PR5C	2		T	D16	PR6C	2		T
VCCIO1	VCCIO1	1			VCCIO2	VCCIO2	2			VCCIO2	VCCIO2	2		
GND	GNDIO1	1			GND	GNDIO2	2			GND	GNDIO2	2		
D15	PR2D	1		C	D15	PR5B	2		C*	D15	PR6B	2		C*
C15	PR2C	1		T	C15	PR5A	2		T*	C15	PR6A	2		T*
C16	PR2B	1		C	C16	PR4D	2		C	C16	PR5D	2		C
B16	PR2A	1		T	B16	PR4C	2		T	B16	PR5C	2		T
F14	PR3D	1		C	F14	PR4B	2		C*	F14	PR5B	2		C*
E14	PR3C	1		T	E14	PR4A	2		T*	E14	PR5A	2		T*
-	-	-			-	-	-			GND	GND	-		
F12	NC				F12	PR3D	2		C	F12	PR4D	2		C
F13	NC				F13	PR3C	2		T	F13	PR4C	2		T
E12	NC				E12	PR3B	2		C*	E12	PR4B	2		C*
E13	NC				E13	PR3A	2		T*	E13	PR4A	2		T*
D13	NC				D13	PR2B	2		C	D13	PR3B	2		C*
D14	NC				D14	PR2A	2		T	D14	PR3A	2		T*
VCCIO0	VCCIO0	0			VCCIO2	VCCIO2	2			VCCIO2	VCCIO2	2		
GND	GNDIO0	0			GND	GNDIO2	2			GND	GNDIO2	2		
GND	GNDIO0	0			GND	GNDIO1	1			GND	GNDIO1	1		
VCCIO0	VCCIO0	0			VCCIO1	VCCIO1	1			VCCIO1	VCCIO1	1		
B15	NC				B15	PT11D	1		C	B15	PT16D	1		C
A15	NC				A15	PT11C	1		T	A15	PT16C	1		T
C14	NC				C14	PT11B	1		C	C14	PT16B	1		C
B14	NC				B14	PT11A	1		T	B14	PT16A	1		T
C13	PT9F	0		C	C13	PT10F	1		C	C13	PT15D	1		C
B13	PT9E	0		T	B13	PT10E	1		T	B13	PT15C	1		T

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 256 caBGA / 256 ftBGA (Cont.)

LCMXO640					LCMXO1200					LCMXO2280				
Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential
E11	NC				E11	PT10D	1		C	E11	PT15B	1		C
E10	NC				E10	PT10C	1		T	E10	PT15A	1		T
D12	PT9D	0		C	D12	PT10B	1		C	D12	PT14D	1		C
D11	PT9C	0		T	D11	PT10A	1		T	D11	PT14C	1		T
A14	PT7F	0		C	A14	PT9F	1		C	A14	PT14B	1		C
A13	PT7E	0		T	A13	PT9E	1		T	A13	PT14A	1		T
C12	PT8B	0		C	C12	PT9D	1		C	C12	PT13D	1		C
C11	PT8A	0		T	C11	PT9C	1		T	C11	PT13C	1		T
-	-				VCCIO1	VCCIO1	1			VCCIO1	VCCIO1	1		
-	-				GND	GNDIO1	1			GND	GNDIO1	1		
B12	PT7B	0		C	B12	PT9B	1		C	B12	PT12D	1		C
B11	PT7A	0		T	B11	PT9A	1		T	B11	PT12C	1		T
A12	PT7D	0		C	A12	PT8F	1		C	A12	PT12B	1		C
A11	PT7C	0		T	A11	PT8E	1		T	A11	PT12A	1		T
GND	GND	-			GND	GND	-			GND	GND	-		
B10	PT5D	0		C	B10	PT8D	1		C	B10	PT11B	1		C
B9	PT5C	0		T	B9	PT8C	1		T	B9	PT11A	1		T
D10	PT8D	0		C	D10	PT8B	1		C	D10	PT10F	1		C
D9	PT8C	0		T	D9	PT8A	1		T	D9	PT10E	1		T
-	-				VCCIO1	VCCIO1	1			VCCIO1	VCCIO1	1		
-	-				GND	GNDIO1	1			GND	GNDIO1	1		
C10	PT6D	0		C	C10	PT7F	1		C	C10	PT10D	1		C
C9	PT6C	0		T	C9	PT7E	1		T	C9	PT10C	1		T
A9	PT6B	0	PCLK0_1***	C	A9	PT7D	1	PCLK1_1***	C	A9	PT10B	1	PCLK1_1***	C
A10	PT6A	0		T	A10	PT7C	1		T	A10	PT10A	1		T
E9	PT9B	0		C	E9	PT7B	1		C	E9	PT9D	1		C
E8	PT9A	0		T	E8	PT7A	1		T	E8	PT9C	1		T
D7	PT5B	0	PCLK0_0***	C	D7	PT6F	0	PCLK1_0***	C	D7	PT9B	1	PCLK1_0***	C
D8	PT5A	0		T	D8	PT6E	0		T	D8	PT9A	1		T
VCCIO0	VCCIO0	0			VCCIO0	VCCIO0	0			VCCIO0	VCCIO0	0		
GND	GNDIO0	0			GND	GNDIO0	0			GND	GNDIO0	0		
C8	PT4F	0		C	C8	PT6D	0		C	C8	PT8D	0		C
B8	PT4E	0		T	B8	PT6C	0		T	B8	PT8C	0		T
A8	VCCAUX	-			A8	VCCAUX	-			A8	VCCAUX	-		
A7	PT4D	0		C	A7	PT6B	0		C	A7	PT7D	0		C
A6	PT4C	0		T	A6	PT6A	0		T	A6	PT7C	0		T
VCC	VCC	-			VCC	VCC	-			VCC	VCC	-		
B7	PT4B	0		C	B7	PT5F	0		C	B7	PT7B	0		C
B6	PT4A	0		T	B6	PT5E	0		T	B6	PT7A	0		T
C6	PT3C	0		T	C6	PT5C	0		T	C6	PT6A	0		T
C7	PT3D	0		C	C7	PT5D	0		C	C7	PT6B	0		C
A5	PT3E	0		T	A5	PT5A	0		T	A5	PT6C	0		T
A4	PT3F	0		C	A4	PT5B	0		C	A4	PT6D	0		C
E7	NC				E7	PT4C	0		T	E7	PT6E	0		T
E6	NC				E6	PT4D	0		C	E6	PT6F	0		C
B5	PT3B	0		C	B5	PT3F	0		C	B5	PT5D	0		C
B4	PT3A	0		T	B4	PT3E	0		T	B4	PT5C	0		T
D5	PT2D	0		C	D5	PT3D	0		C	D5	PT5B	0		C
D6	PT2C	0		T	D6	PT3C	0		T	D6	PT5A	0		T
C4	PT2E	0		T	C4	PT4A	0		T	C4	PT4A	0		T
C5	PT2F	0		C	C5	PT4B	0		C	C5	PT4B	0		C
-	-	-			-	-	-			GND	GND	-		
D4	NC				D4	PT2D	0		C	D4	PT3D	0		C

LCMXO640, LCMXO1200 and LCMXO2280 Logic Signal Connections: 256 caBGA / 256 ftBGA (Cont.)

LCMXO640					LCMXO1200					LCMXO2280				
Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential	Ball Number	Ball Function	Bank	Dual Function	Differential
D3	NC				D3	PT2C	0		T	D3	PT3C	0		T
A3	PT2B	0		C	A3	PT3B	0		C	A3	PT3B	0		C
A2	PT2A	0		T	A2	PT3A	0		T	A2	PT3A	0		T
B3	NC				B3	PT2B	0		C	B3	PT2D	0		C
B2	NC				B2	PT2A	0		T	B2	PT2C	0		T
VCCIO0	VCCIO0	0			VCCIO0	VCCIO0	0			VCCIO0	VCCIO0	0		
GND	GNDIO0	0			GND	GNDIO0	0			GND	GNDIO0	0		
A1	GND	-			A1	GND	-			A1	GND	-		
A16	GND	-			A16	GND	-			A16	GND	-		
F11	GND	-			F11	GND	-			F11	GND	-		
G8	GND	-			G8	GND	-			G8	GND	-		
G9	GND	-			G9	GND	-			G9	GND	-		
H7	GND	-			H7	GND	-			H7	GND	-		
H8	GND	-			H8	GND	-			H8	GND	-		
H9	GND	-			H9	GND	-			H9	GND	-		
H10	GND	-			H10	GND	-			H10	GND	-		
J7	GND	-			J7	GND	-			J7	GND	-		
J8	GND	-			J8	GND	-			J8	GND	-		
J9	GND	-			J9	GND	-			J9	GND	-		
J10	GND	-			J10	GND	-			J10	GND	-		
K8	GND	-			K8	GND	-			K8	GND	-		
K9	GND	-			K9	GND	-			K9	GND	-		
L6	GND	-			L6	GND	-			L6	GND	-		
T1	GND	-			T1	GND	-			T1	GND	-		
T16	GND	-			T16	GND	-			T16	GND	-		
G7	VCC	-			G7	VCC	-			G7	VCC	-		
G10	VCC	-			G10	VCC	-			G10	VCC	-		
K7	VCC	-			K7	VCC	-			K7	VCC	-		
K10	VCC	-			K10	VCC	-			K10	VCC	-		
H6	VCCIO3	3			H6	VCCIO7	7			H6	VCCIO7	7		
G6	VCCIO3	3			G6	VCCIO7	7			G6	VCCIO7	7		
K6	VCCIO3	3			K6	VCCIO6	6			K6	VCCIO6	6		
J6	VCCIO3	3			J6	VCCIO6	6			J6	VCCIO6	6		
L8	VCCIO2	2			L8	VCCIO5	5			L8	VCCIO5	5		
L7	VCCIO2	2			L7	VCCIO5	5			L7	VCCIO5	5		
L9	VCCIO2	2			L9	VCCIO4	4			L9	VCCIO4	4		
L10	VCCIO2	2			L10	VCCIO4	4			L10	VCCIO4	4		
K11	VCCIO1	1			K11	VCCIO3	3			K11	VCCIO3	3		
J11	VCCIO1	1			J11	VCCIO3	3			J11	VCCIO3	3		
H11	VCCIO1	1			H11	VCCIO2	2			H11	VCCIO2	2		
G11	VCCIO1	1			G11	VCCIO2	2			G11	VCCIO2	2		
F9	VCCIO0	0			F9	VCCIO1	1			F9	VCCIO1	1		
F10	VCCIO0	0			F10	VCCIO1	1			F10	VCCIO1	1		
F8	VCCIO0	0			F8	VCCIO0	0			F8	VCCIO0	0		
F7	VCCIO0	0			F7	VCCIO0	0			F7	VCCIO0	0		

* Supports true LVDS outputs.
** NC for “E” devices.
*** Primary clock inputs are single-ended.

LCMXO2280 Logic Signal Connections: 324 ftBGA

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
GND	GNDIO7	7		
VCCIO7	VCCIO7	7		
D4	PL2A	7	LUM0_PLLT_FB_A	T
F5	PL2B	7	LUM0_PLLC_FB_A	C
B3	PL3A	7		T*
C3	PL3B	7		C*
E4	PL3C	7	LUM0_PLLT_IN_A	T
G6	PL3D	7	LUM0_PLLC_IN_A	C
A1	PL4A	7		T*
B1	PL4B	7		C*
F4	PL4C	7		T
VCC	VCC	-		
E3	PL4D	7		C
D2	PL5A	7		T*
D3	PL5B	7		C*
G5	PL5C	7		T
F3	PL5D	7		C
C2	PL6A	7		T*
VCCIO7	VCCIO7	7		
GND	GNDIO7	7		
C1	PL6B	7		C*
H5	PL6C	7		T
G4	PL6D	7		C
E2	PL7A	7		T*
D1	PL7B	7	GSRN	C*
J6	PL7C	7		T
H4	PL7D	7		C
F2	PL8A	7		T*
E1	PL8B	7		C*
GND	GND	-		
J3	PL8C	7		T
J5	PL8D	7		C
G3	PL9A	7		T*
H3	PL9B	7		C*
K3	PL9C	7		T
K5	PL9D	7		C
F1	PL10A	7		T*
VCCIO7	VCCIO7	7		
GND	GNDIO7	7		
G1	PL10B	7		C*
K4	PL10C	7		T
K6	PL10D	7		C

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
G2	PL11A	6		T*
H2	PL11B	6		C*
L3	PL11C	6		T
L5	PL11D	6		C
H1	PL12A	6		T*
VCCIO6	VCCIO6	6		
GND	GNDIO6	6		
J2	PL12B	6		C*
L4	PL12C	6		T
L6	PL12D	6		C
K2	PL13A	6		T*
K1	PL13B	6		C*
J1	PL13C	6		T
VCC	VCC	-		
L2	PL13D	6		C
M5	PL14D	6		C
M3	PL14C	6	TSALL	T
L1	PL14B	6		C*
M2	PL14A	6		T*
M1	PL15A	6		T*
N1	PL15B	6		C*
M6	PL15C	6		T
M4	PL15D	6		C
VCCIO6	VCCIO6	6		
GND	GNDIO6	6		
P1	PL16A	6		T*
P2	PL16B	6		C*
N3	PL16C	6		T
N4	PL16D	6		C
GND	GND	-		
T1	PL17A	6	LLM0_PLLT_FB_A	T*
R1	PL17B	6	LLM0_PLLC_FB_A	C*
P3	PL17C	6		T
N5	PL17D	6		C
R3	PL18A	6	LLM0_PLLT_IN_A	T*
R2	PL18B	6	LLM0_PLLC_IN_A	C*
P4	PL19A	6		T
N6	PL19B	6		C
U1	PL20A	6		T
VCCIO6	VCCIO6	6		
GND	GNDIO6	6		
GND	GNDIO5	5		
VCCIO5	VCCIO5	5		

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
T2	PL20B	6		C
P6	TMS	5	TMS	
V1	PB2A	5		T
U2	PB2B	5		C
T3	PB2C	5		T
N7	TCK	5	TCK	
R4	PB2D	5		C
R5	PB3A	5		T
T4	PB3B	5		C
VCC	VCC	-		
R6	PB3C	5		T
P7	PB3D	5		C
U3	PB4A	5		T
T5	PB4B	5		C
V2	PB4C	5		T
N8	TDO	5	TDO	
V3	PB4D	5		C
T6	PB5A	5		T
GND	GNDIO5	5		
VCCIO5	VCCIO5	5		
U4	PB5B	5		C
P8	PB5C	5		T
T7	PB5D	5		C
V4	TDI	5	TDI	
R8	PB6A	5		T
N9	PB6B	5		C
U5	PB6C	5		T
V5	PB6D	5		C
U6	PB7A	5		T
VCC	VCC	-		
V6	PB7B	5		C
P9	PB7C	5		T
T8	PB7D	5		C
U7	PB8A	5		T
V7	PB8B	5		C
M10	VCCAUX	-		
U8	PB8C	5		T
V8	PB8D	5		C
VCCIO5	VCCIO5	5		
GND	GNDIO5	5		
T9	PB8E	5		T
U9	PB8F	5		C
V9	PB9A	4		T

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
V10	PB9B	4		C
N10	PB9C	4		T
R10	PB9D	4		C
P10	PB10F	4	PCLK4_1***	C
T10	PB10E	4		T
U10	PB10D	4		C
V11	PB10C	4		T
U11	PB10B	4	PCLK4_0***	C
VCCIO4	VCCIO4	4		
GND	GNDIO4	4		
T11	PB10A	4		T
U12	PB11A	4		T
R11	PB11B	4		C
GND	GND	-		
T12	PB11C	4		T
P11	PB11D	4		C
V12	PB12A	4		T
V13	PB12B	4		C
R12	PB12C	4		T
N11	PB12D	4		C
U13	PB12E	4		T
VCCIO4	VCCIO4	4		
GND	GNDIO4	4		
V14	PB12F	4		C
T13	PB13A	4		T
P12	PB13B	4		C
R13	PB13C	4		T
N12	PB13D	4		C
V15	PB14A	4		T
U14	PB14B	4		C
V16	PB14C	4		T
GND	GND	-		
T14	PB14D	4		C
U15	PB15A	4		T
V17	PB15B	4		C
P13**	SLEEPN	-	SLEEPN	
T15	PB15D	4		
U16	PB16A	4		T
V18	PB16B	4		C
N13	PB16C	4		T
R14	PB16D	4		C
VCCIO4	VCCIO4	4		
GND	GNDIO4	4		

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
GND	GNDIO3	3		
VCCIO3	VCCIO3	3		
P15	PR20B	3		C
N14	PR20A	3		T
N15	PR19B	3		C
M13	PR19A	3		T
R15	PR18B	3		C*
T16	PR18A	3		T*
N16	PR17D	3		C
M14	PR17C	3		T
U17	PR17B	3		C*
VCC	VCC	-		
U18	PR17A	3		T*
R17	PR16D	3		C
R16	PR16C	3		T
P16	PR16B	3		C*
VCCIO3	VCCIO3	3		
GND	GNDIO3	3		
P17	PR16A	3		T*
L13	PR15D	3		C
M15	PR15C	3		T
T17	PR15B	3		C*
T18	PR15A	3		T*
L14	PR14D	3		C
L15	PR14C	3		T
R18	PR14B	3		C*
P18	PR14A	3		T*
GND	GND	-		
K15	PR13D	3		C
K13	PR13C	3		T
N17	PR13B	3		C*
N18	PR13A	3		T*
K16	PR12D	3		C
K14	PR12C	3		T
M16	PR12B	3		C*
L16	PR12A	3		T*
GND	GNDIO3	3		
VCCIO3	VCCIO3	3		
J16	PR11D	3		C
J14	PR11C	3		T
M17	PR11B	3		C*
L17	PR11A	3		T*
J15	PR10D	2		C

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
J13	PR10C	2		T
M18	PR10B	2		C*
L18	PR10A	2		T*
GND	GNDIO2	2		
VCCIO2	VCCIO2	2		
H16	PR9D	2		C
H14	PR9C	2		T
K18	PR9B	2		C*
J18	PR9A	2		T*
J17	PR8D	2		C
VCC	VCC	-		
H18	PR8C	2		T
H17	PR8B	2		C*
G17	PR8A	2		T*
H13	PR7D	2		C
H15	PR7C	2		T
G18	PR7B	2		C*
F18	PR7A	2		T*
G14	PR6D	2		C
G16	PR6C	2		T
VCCIO2	VCCIO2	2		
GND	GNDIO2	2		
E18	PR6B	2		C*
F17	PR6A	2		T*
G13	PR5D	2		C
G15	PR5C	2		T
E17	PR5B	2		C*
E16	PR5A	2		T*
GND	GND	-		
F15	PR4D	2		C
E15	PR4C	2		T
D17	PR4B	2		C*
D18	PR4A	2		T*
B18	PR3D	2		C
C18	PR3C	2		T
C16	PR3B	2		C*
D16	PR3A	2		T*
C17	PR2B	2		C
D15	PR2A	2		T
VCCIO2	VCCIO2	2		
GND	GNDIO2	2		
GND	GNDIO1	1		
VCCIO1	VCCIO1	1		

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
E13	PT16D	1		C
C15	PT16C	1		T
F13	PT16B	1		C
D14	PT16A	1		T
A18	PT15D	1		C
B17	PT15C	1		T
A16	PT15B	1		C
A17	PT15A	1		T
VCC	VCC	-		
D13	PT14D	1		C
F12	PT14C	1		T
C14	PT14B	1		C
E12	PT14A	1		T
C13	PT13D	1		C
B16	PT13C	1		T
B15	PT13B	1		C
A15	PT13A	1		T
VCCIO1	VCCIO1	1		
GND	GNDIO1	1		
B14	PT12F	1		C
A14	PT12E	1		T
D12	PT12D	1		C
F11	PT12C	1		T
B13	PT12B	1		C
A13	PT12A	1		T
C12	PT11D	1		C
GND	GND	-		
B12	PT11C	1		T
E11	PT11B	1		C
D11	PT11A	1		T
C11	PT10F	1		C
A12	PT10E	1		T
VCCIO1	VCCIO1	1		
GND	GNDIO1	1		
F10	PT10D	1		C
D10	PT10C	1		T
B11	PT10B	1	PCLK1_1***	C
A11	PT10A	1		T
E10	PT9D	1		C
C10	PT9C	1		T
D9	PT9B	1	PCLK1_0***	C
E9	PT9A	1		T
B10	PT8F	0		C

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
A10	PT8E	0		T
VCCIO0	VCCIO0	0		
GND	GNDIO0	0		
A9	PT8D	0		C
C9	PT8C	0		T
B9	PT8B	0		C
F9	VCCAUX	-		
A8	PT8A	0		T
B8	PT7D	0		C
C8	PT7C	0		T
VCC	VCC	-		
A7	PT7B	0		C
B7	PT7A	0		T
A6	PT6A	0		T
B6	PT6B	0		C
D8	PT6C	0		T
F8	PT6D	0		C
C7	PT6E	0		T
E8	PT6F	0		C
D7	PT5D	0		C
VCCIO0	VCCIO0	0		
GND	GNDIO0	0		
E7	PT5C	0		T
A5	PT5B	0		C
C6	PT5A	0		T
B5	PT4A	0		T
A4	PT4B	0		C
D6	PT4C	0		T
F7	PT4D	0		C
B4	PT4E	0		T
GND	GND	-		
C5	PT4F	0		C
F6	PT3D	0		C
E5	PT3C	0		T
E6	PT3B	0		C
D5	PT3A	0		T
A3	PT2D	0		C
C4	PT2C	0		T
A2	PT2B	0		C
B2	PT2A	0		T
VCCIO0	VCCIO0	0		
GND	GNDIO0	0		
E14	GND	-		

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
F16	GND	-		
H10	GND	-		
H11	GND	-		
H8	GND	-		
H9	GND	-		
J10	GND	-		
J11	GND	-		
J4	GND	-		
J8	GND	-		
J9	GND	-		
K10	GND	-		
K11	GND	-		
K17	GND	-		
K8	GND	-		
K9	GND	-		
L10	GND	-		
L11	GND	-		
L8	GND	-		
L9	GND	-		
N2	GND	-		
P14	GND	-		
P5	GND	-		
R7	GND	-		
F14	VCC	-		
G11	VCC	-		
G9	VCC	-		
H7	VCC	-		
L7	VCC	-		
M9	VCC	-		
H6	VCCIO7	7		
J7	VCCIO7	7		
M7	VCCIO6	6		
K7	VCCIO6	6		
M8	VCCIO5	5		
R9	VCCIO5	5		
M12	VCCIO4	4		
M11	VCCIO4	4		
L12	VCCIO3	3		
K12	VCCIO3	3		
J12	VCCIO2	2		
H12	VCCIO2	2		
G12	VCCIO1	1		
G10	VCCIO1	1		

LCMXO2280 Logic Signal Connections: 324 ftBGA (Cont.)

LCMXO2280				
Ball Number	**Ball Function**	**Bank**	**Dual Function**	**Differential**
G8	VCCIO0	0		
G7	VCCIO0	0		

* Supports true LVDS outputs.
** NC for “E” devices.
*** Primary clock inputs are single-ended.

Thermal Management

Thermal management is recommended as part of any sound FPGA design methodology. To assess the thermal characteristics of a system, Lattice specifies a maximum allowable junction temperature in all device data sheets. Designers must complete a thermal analysis of their specific design to ensure that the device and package do not exceed the junction temperature limits. Refer to the Thermal Management document to find the device/package specific thermal values.

For Further Information

For further information regarding Thermal Management, refer to the following:

- Thermal Management document
- TN1090 - Power Estimation and Management for MachXO Devices
- Power Calculator tool included with the Lattice ispLEVER design tool, or as a standalone download from www.latticesemi.com/software

MachXO Family Data Sheet Ordering Information

June 2009 Data Sheet DS1002

Part Number Description

LCMXO XXXX X – X XXXXXX X XX

Device Family
MachXO PLD

Logic Capacity
256 LUTs = 256
640 LUTs = 640
1200 LUTs = 1200
2280 LUTs = 2280

Supply Voltage
C = 1.8V/2.5V/3.3V
E = 1.2V

Note: Parts dual marked as described.

Speed
3 = Slowest
4
5 = Fastest

Package
T100 = 100-pin TQFP
T144 = 144-pin TQFP
M100 = 100-ball csBGA
M132 = 132-ball csBGA
B256 = 256-ball caBGA
FT256 = 256-ball ftBGA
FT324 = 324-ball ftBGA

TN100 = 100-pin Lead-Free TQFP
TN144 = 144-pin Lead-Free TQFP
MN100 = 100-ball Lead-Free csBGA
MN132 = 132-ball Lead-Free csBGA
BN256 = 256-ball Lead-Free caBGA
FTN256 = 256-ball Lead-Free ftBGA
FTN324 = 324-ball Lead-Free ftBGA

Grade
C = Commercial
I = Industrial

ES = Engineering Sample
Blank = Production Device

Ordering Information

Note: MachXO devices are dual marked except the slowest commercial speed grade device.þFor example the commercial speed grade LCMXO640E-4F256C is also marked with industrial grade -3I grade.þThe slowest commercial speed grade does not have industrial markings.þ The markings appears as follows:

Dual Mark

Conventional Packaging

Commercial

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO256C-3T100C	256	1.8V/2.5V/3.3V	78	-3	TQFP	100	COM
LCMXO256C-4T100C	256	1.8V/2.5V/3.3V	78	-4	TQFP	100	COM
LCMXO256C-5T100C	256	1.8V/2.5V/3.3V	78	-5	TQFP	100	COM
LCMXO256C-3M100C	256	1.8V/2.5V/3.3V	78	-3	csBGA	100	COM
LCMXO256C-4M100C	256	1.8V/2.5V/3.3V	78	-4	csBGA	100	COM
LCMXO256C-5M100C	256	1.8V/2.5V/3.3V	78	-5	csBGA	100	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO640C-3T100C	640	1.8V/2.5V/3.3V	74	-3	TQFP	100	COM
LCMXO640C-4T100C	640	1.8V/2.5V/3.3V	74	-4	TQFP	100	COM
LCMXO640C-5T100C	640	1.8V/2.5V/3.3V	74	-5	TQFP	100	COM
LCMXO640C-3M100C	640	1.8V/2.5V/3.3V	74	-3	csBGA	100	COM
LCMXO640C-4M100C	640	1.8V/2.5V/3.3V	74	-4	csBGA	100	COM
LCMXO640C-5M100C	640	1.8V/2.5V/3.3V	74	-5	csBGA	100	COM
LCMXO640C-3T144C	640	1.8V/2.5V/3.3V	113	-3	TQFP	144	COM
LCMXO640C-4T144C	640	1.8V/2.5V/3.3V	113	-4	TQFP	144	COM
LCMXO640C-5T144C	640	1.8V/2.5V/3.3V	113	-5	TQFP	144	COM
LCMXO640C-3M132C	640	1.8V/2.5V/3.3V	101	-3	csBGA	132	COM
LCMXO640C-4M132C	640	1.8V/2.5V/3.3V	101	-4	csBGA	132	COM
LCMXO640C-5M132C	640	1.8V/2.5V/3.3V	101	-5	csBGA	132	COM
LCMXO640C-3B256C	640	1.8V/2.5V/3.3V	159	-3	caBGA	256	COM
LCMXO640C-4B256C	640	1.8V/2.5V/3.3V	159	-4	caBGA	256	COM
LCMXO640C-5B256C	640	1.8V/2.5V/3.3V	159	-5	caBGA	256	COM
LCMXO640C-3FT256C	640	1.8V/2.5V/3.3V	159	-3	ftBGA	256	COM
LCMXO640C-4FT256C	640	1.8V/2.5V/3.3V	159	-4	ftBGA	256	COM
LCMXO640C-5FT256C	640	1.8V/2.5V/3.3V	159	-5	ftBGA	256	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO1200C-3T100C	1200	1.8V/2.5V/3.3V	73	-3	TQFP	100	COM
LCMXO1200C-4T100C	1200	1.8V/2.5V/3.3V	73	-4	TQFP	100	COM
LCMXO1200C-5T100C	1200	1.8V/2.5V/3.3V	73	-5	TQFP	100	COM
LCMXO1200C-3T144C	1200	1.8V/2.5V/3.3V	113	-3	TQFP	144	COM
LCMXO1200C-4T144C	1200	1.8V/2.5V/3.3V	113	-4	TQFP	144	COM
LCMXO1200C-5T144C	1200	1.8V/2.5V/3.3V	113	-5	TQFP	144	COM
LCMXO1200C-3M132C	1200	1.8V/2.5V/3.3V	101	-3	csBGA	132	COM
LCMXO1200C-4M132C	1200	1.8V/2.5V/3.3V	101	-4	csBGA	132	COM
LCMXO1200C-5M132C	1200	1.8V/2.5V/3.3V	101	-5	csBGA	132	COM
LCMXO1200C-3B256C	1200	1.8V/2.5V/3.3V	211	-3	caBGA	256	COM
LCMXO1200C-4B256C	1200	1.8V/2.5V/3.3V	211	-4	caBGA	256	COM
LCMXO1200C-5B256C	1200	1.8V/2.5V/3.3V	211	-5	caBGA	256	COM
LCMXO1200C-3FT256C	1200	1.8V/2.5V/3.3V	211	-3	ftBGA	256	COM
LCMXO1200C-4FT256C	1200	1.8V/2.5V/3.3V	211	-4	ftBGA	256	COM
LCMXO1200C-5FT256C	1200	1.8V/2.5V/3.3V	211	-5	ftBGA	256	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO2280C-3T100C	2280	1.8V/2.5V/3.3V	73	-3	TQFP	100	COM
LCMXO2280C-4T100C	2280	1.8V/2.5V/3.3V	73	-4	TQFP	100	COM
LCMXO2280C-5T100C	2280	1.8V/2.5V/3.3V	73	-5	TQFP	100	COM
LCMXO2280C-3T144C	2280	1.8V/2.5V/3.3V	113	-3	TQFP	144	COM
LCMXO2280C-4T144C	2280	1.8V/2.5V/3.3V	113	-4	TQFP	144	COM
LCMXO2280C-5T144C	2280	1.8V/2.5V/3.3V	113	-5	TQFP	144	COM
LCMXO2280C-3M132C	2280	1.8V/2.5V/3.3V	101	-3	csBGA	132	COM
LCMXO2280C-4M132C	2280	1.8V/2.5V/3.3V	101	-4	csBGA	132	COM
LCMXO2280C-5M132C	2280	1.8V/2.5V/3.3V	101	-5	csBGA	132	COM
LCMXO2280C-3B256C	2280	1.8V/2.5V/3.3V	211	-3	caBGA	256	COM
LCMXO2280C-4B256C	2280	1.8V/2.5V/3.3V	211	-4	caBGA	256	COM
LCMXO2280C-5B256C	2280	1.8V/2.5V/3.3V	211	-5	caBGA	256	COM
LCMXO2280C-3FT256C	2280	1.8V/2.5V/3.3V	211	-3	ftBGA	256	COM
LCMXO2280C-4FT256C	2280	1.8V/2.5V/3.3V	211	-4	ftBGA	256	COM
LCMXO2280C-5FT256C	2280	1.8V/2.5V/3.3V	211	-5	ftBGA	256	COM
LCMXO2280C-3FT324C	2280	1.8V/2.5V/3.3V	271	-3	ftBGA	324	COM
LCMXO2280C-4FT324C	2280	1.8V/2.5V/3.3V	271	-4	ftBGA	324	COM
LCMXO2280C-5FT324C	2280	1.8V/2.5V/3.3V	271	-5	ftBGA	324	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO256E-3T100C	256	1.2V	78	-3	TQFP	100	COM
LCMXO256E-4T100C	256	1.2V	78	-4	TQFP	100	COM
LCMXO256E-5T100C	256	1.2V	78	-5	TQFP	100	COM
LCMXO256E-3M100C	256	1.2V	78	-3	csBGA	100	COM
LCMXO256E-4M100C	256	1.2V	78	-4	csBGA	100	COM
LCMXO256E-5M100C	256	1.2V	78	-5	csBGA	100	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO640E-3T100C	640	1.2V	74	-3	TQFP	100	COM
LCMXO640E-4T100C	640	1.2V	74	-4	TQFP	100	COM
LCMXO640E-5T100C	640	1.2V	74	-5	TQFP	100	COM
LCMXO640E-3M100C	640	1.2V	74	-3	csBGA	100	COM
LCMXO640E-4M100C	640	1.2V	74	-4	csBGA	100	COM
LCMXO640E-5M100C	640	1.2V	74	-5	csBGA	100	COM
LCMXO640E-3T144C	640	1.2V	113	-3	TQFP	144	COM
LCMXO640E-4T144C	640	1.2V	113	-4	TQFP	144	COM
LCMXO640E-5T144C	640	1.2V	113	-5	TQFP	144	COM
LCMXO640E-3M132C	640	1.2V	101	-3	csBGA	132	COM
LCMXO640E-4M132C	640	1.2V	101	-4	csBGA	132	COM
LCMXO640E-5M132C	640	1.2V	101	-5	csBGA	132	COM
LCMXO640E-3B256C	640	1.2V	159	-3	caBGA	256	COM
LCMXO640E-4B256C	640	1.2V	159	-4	caBGA	256	COM
LCMXO640E-5B256C	640	1.2V	159	-5	caBGA	256	COM
LCMXO640E-3FT256C	640	1.2V	159	-3	ftBGA	256	COM
LCMXO640E-4FT256C	640	1.2V	159	-4	ftBGA	256	COM
LCMXO640E-5FT256C	640	1.2V	159	-5	ftBGA	256	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO1200E-3T100C	1200	1.2V	73	-3	TQFP	100	COM
LCMXO1200E-4T100C	1200	1.2V	73	-4	TQFP	100	COM
LCMXO1200E-5T100C	1200	1.2V	73	-5	TQFP	100	COM
LCMXO1200E-3T144C	1200	1.2V	113	-3	TQFP	144	COM
LCMXO1200E-4T144C	1200	1.2V	113	-4	TQFP	144	COM
LCMXO1200E-5T144C	1200	1.2V	113	-5	TQFP	144	COM
LCMXO1200E-3M132C	1200	1.2V	101	-3	csBGA	132	COM
LCMXO1200E-4M132C	1200	1.2V	101	-4	csBGA	132	COM
LCMXO1200E-5M132C	1200	1.2V	101	-5	csBGA	132	COM
LCMXO1200E-3B256C	1200	1.2V	211	-3	caBGA	256	COM
LCMXO1200E-4B256C	1200	1.2V	211	-4	caBGA	256	COM
LCMXO1200E-5B256C	1200	1.2V	211	-5	caBGA	256	COM
LCMXO1200E-3FT256C	1200	1.2V	211	-3	ftBGA	256	COM
LCMXO1200E-4FT256C	1200	1.2V	211	-4	ftBGA	256	COM
LCMXO1200E-5FT256C	1200	1.2V	211	-5	ftBGA	256	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO2280E-3T100C	2280	1.2V	73	-3	TQFP	100	COM
LCMXO2280E-4T100C	2280	1.2V	73	-4	TQFP	100	COM
LCMXO2280E-5T100C	2280	1.2V	73	-5	TQFP	100	COM
LCMXO2280E-3T144C	2280	1.2V	113	-3	TQFP	144	COM
LCMXO2280E-4T144C	2280	1.2V	113	-4	TQFP	144	COM
LCMXO2280E-5T144C	2280	1.2V	113	-5	TQFP	144	COM
LCMXO2280E-3M132C	2280	1.2V	101	-3	csBGA	132	COM
LCMXO2280E-4M132C	2280	1.2V	101	-4	csBGA	132	COM
LCMXO2280E-5M132C	2280	1.2V	101	-5	csBGA	132	COM
LCMXO2280E-3B256C	2280	1.2V	211	-3	caBGA	256	COM
LCMXO2280E-4B256C	2280	1.2V	211	-4	caBGA	256	COM
LCMXO2280E-5B256C	2280	1.2V	211	-5	caBGA	256	COM
LCMXO2280E-3FT256C	2280	1.2V	211	-3	ftBGA	256	COM
LCMXO2280E-4FT256C	2280	1.2V	211	-4	ftBGA	256	COM
LCMXO2280E-5FT256C	2280	1.2V	211	-5	ftBGA	256	COM
LCMXO2280E-3FT324C	2280	1.2V	271	-3	ftBGA	324	COM
LCMXO2280E-4FT324C	2280	1.2V	271	-4	ftBGA	324	COM
LCMXO2280E-5FT324C	2280	1.2V	271	-5	ftBGA	324	COM

Conventional Packaging

Industrial

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO256C-3T100I	256	1.8V/2.5V/3.3V	78	-3	TQFP	100	IND
LCMXO256C-4T100I	256	1.8V/2.5V/3.3V	78	-4	TQFP	100	IND
LCMXO256C-3M100I	256	1.8V/2.5V/3.3V	78	-3	csBGA	100	IND
LCMXO256C-4M100I	256	1.8V/2.5V/3.3V	78	-4	csBGA	100	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO640C-3T100I	640	1.8V/2.5V/3.3V	74	-3	TQFP	100	IND
LCMXO640C-4T100I	640	1.8V/2.5V/3.3V	74	-4	TQFP	100	IND
LCMXO640C-3M100I	640	1.8V/2.5V/3.3V	74	-3	csBGA	100	IND
LCMXO640C-4M100I	640	1.8V/2.5V/3.3V	74	-4	csBGA	100	IND
LCMXO640C-3T144I	640	1.8V/2.5V/3.3V	113	-3	TQFP	144	IND
LCMXO640C-4T144I	640	1.8V/2.5V/3.3V	113	-4	TQFP	144	IND
LCMXO640C-3M132I	640	1.8V/2.5V/3.3V	101	-3	csBGA	132	IND
LCMXO640C-4M132I	640	1.8V/2.5V/3.3V	101	-4	csBGA	132	IND
LCMXO640C-3B256I	640	1.8V/2.5V/3.3V	159	-3	caBGA	256	IND
LCMXO640C-4B256I	640	1.8V/2.5V/3.3V	159	-4	caBGA	256	IND
LCMXO640C-3FT256I	640	1.8V/2.5V/3.3V	159	-3	ftBGA	256	IND
LCMXO640C-4FT256I	640	1.8V/2.5V/3.3V	159	-4	ftBGA	256	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO1200C-3T100I	1200	1.8V/2.5V/3.3V	73	-3	TQFP	100	IND
LCMXO1200C-4T100I	1200	1.8V/2.5V/3.3V	73	-4	TQFP	100	IND
LCMXO1200C-3T144I	1200	1.8V/2.5V/3.3V	113	-3	TQFP	144	IND
LCMXO1200C-4T144I	1200	1.8V/2.5V/3.3V	113	-4	TQFP	144	IND
LCMXO1200C-3M132I	1200	1.8V/2.5V/3.3V	101	-3	csBGA	132	IND
LCMXO1200C-4M132I	1200	1.8V/2.5V/3.3V	101	-4	csBGA	132	IND
LCMXO1200C-3B256I	1200	1.8V/2.5V/3.3V	211	-3	caBGA	256	IND
LCMXO1200C-4B256I	1200	1.8V/2.5V/3.3V	211	-4	caBGA	256	IND
LCMXO1200C-3FT256I	1200	1.8V/2.5V/3.3V	211	-3	ftBGA	256	IND
LCMXO1200C-4FT256I	1200	1.8V/2.5V/3.3V	211	-4	ftBGA	256	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO2280C-3T100I	2280	1.8V/2.5V/3.3V	73	-3	TQFP	100	IND
LCMXO2280C-4T100I	2280	1.8V/2.5V/3.3V	73	-4	TQFP	100	IND
LCMXO2280C-3T144I	2280	1.8V/2.5V/3.3V	113	-3	TQFP	144	IND
LCMXO2280C-4T144I	2280	1.8V/2.5V/3.3V	113	-4	TQFP	144	IND
LCMXO2280C-3M132I	2280	1.8V/2.5V/3.3V	101	-3	csBGA	132	IND
LCMXO2280C-4M132I	2280	1.8V/2.5V/3.3V	101	-4	csBGA	132	IND
LCMXO2280C-3B256I	2280	1.8V/2.5V/3.3V	211	-3	caBGA	256	IND
LCMXO2280C-4B256I	2280	1.8V/2.5V/3.3V	211	-4	caBGA	256	IND
LCMXO2280C-3FT256I	2280	1.8V/2.5V/3.3V	211	-3	ftBGA	256	IND
LCMXO2280C-4FT256I	2280	1.8V/2.5V/3.3V	211	-4	ftBGA	256	IND
LCMXO2280C-3FT324I	2280	1.8V/2.5V/3.3V	271	-3	ftBGA	324	IND
LCMXO2280C-4FT324I	2280	1.8V/2.5V/3.3V	271	-4	ftBGA	324	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO256E-3T100I	256	1.2V	78	-3	TQFP	100	IND
LCMXO256E-4T100I	256	1.2V	78	-4	TQFP	100	IND
LCMXO256E-3M100I	256	1.2V	78	-3	csBGA	100	IND
LCMXO256E-4M100I	256	1.2V	78	-4	csBGA	100	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO640E-3T100I	640	1.2V	74	-3	TQFP	100	IND
LCMXO640E-4T100I	640	1.2V	74	-4	TQFP	100	IND
LCMXO640E-3M100I	640	1.2V	74	-3	csBGA	100	IND
LCMXO640E-4M100I	640	1.2V	74	-4	csBGA	100	IND
LCMXO640E-3T144I	640	1.2V	113	-3	TQFP	144	IND
LCMXO640E-4T144I	640	1.2V	113	-4	TQFP	144	IND
LCMXO640E-3M132I	640	1.2V	101	-3	csBGA	132	IND
LCMXO640E-4M132I	640	1.2V	101	-4	csBGA	132	IND
LCMXO640E-3B256I	640	1.2V	159	-3	caBGA	256	IND
LCMXO640E-4B256I	640	1.2V	159	-4	caBGA	256	IND
LCMXO640E-3FT256I	640	1.2V	159	-3	ftBGA	256	IND
LCMXO640E-4FT256I	640	1.2V	159	-4	ftBGA	256	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO1200E-3T100I	1200	1.2V	73	-3	TQFP	100	IND
LCMXO1200E-4T100I	1200	1.2V	73	-4	TQFP	100	IND
LCMXO1200E-3T144I	1200	1.2V	113	-3	TQFP	144	IND
LCMXO1200E-4T144I	1200	1.2V	113	-4	TQFP	144	IND
LCMXO1200E-3M132I	1200	1.2V	101	-3	csBGA	132	IND
LCMXO1200E-4M132I	1200	1.2V	101	-4	csBGA	132	IND
LCMXO1200E-3B256I	1200	1.2V	211	-3	caBGA	256	IND
LCMXO1200E-4B256I	1200	1.2V	211	-4	caBGA	256	IND
LCMXO1200E-3FT256I	1200	1.2V	211	-3	ftBGA	256	IND
LCMXO1200E-4FT256I	1200	1.2V	211	-4	ftBGA	256	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO2280E-3T100I	2280	1.2V	73	-3	TQFP	100	IND
LCMXO2280E-4T100I	2280	1.2V	73	-4	TQFP	100	IND
LCMXO2280E-3T144I	2280	1.2V	113	-3	TQFP	144	IND
LCMXO2280E-4T144I	2280	1.2V	113	-4	TQFP	144	IND
LCMXO2280E-3M132I	2280	1.2V	101	-3	csBGA	132	IND
LCMXO2280E-4M132I	2280	1.2V	101	-4	csBGA	132	IND
LCMXO2280E-3B256I	2280	1.2V	211	-3	caBGA	256	IND
LCMXO2280E-4B256I	2280	1.2V	211	-4	caBGA	256	IND
LCMXO2280E-3FT256I	2280	1.2V	211	-3	ftBGA	256	IND
LCMXO2280E-4FT256I	2280	1.2V	211	-4	ftBGA	256	IND
LCMXO2280E-3FT324I	2280	1.2V	271	-3	ftBGA	324	IND
LCMXO2280E-4FT324I	2280	1.2V	271	-4	ftBGA	324	IND

Lead-Free Packaging

Commercial

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO256C-3TN100C	256	1.8V/2.5V/3.3V	78	-3	Lead-Free TQFP	100	COM
LCMXO256C-4TN100C	256	1.8V/2.5V/3.3V	78	-4	Lead-Free TQFP	100	COM
LCMXO256C-5TN100C	256	1.8V/2.5V/3.3V	78	-5	Lead-Free TQFP	100	COM
LCMXO256C-3MN100C	256	1.8V/2.5V/3.3V	78	-3	Lead-Free csBGA	100	COM
LCMXO256C-4MN100C	256	1.8V/2.5V/3.3V	78	-4	Lead-Free csBGA	100	COM
LCMXO256C-5MN100C	256	1.8V/2.5V/3.3V	78	-5	Lead-Free csBGA	100	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO640C-3TN100C	640	1.8V/2.5V/3.3V	74	-3	Lead-Free TQFP	100	COM
LCMXO640C-4TN100C	640	1.8V/2.5V/3.3V	74	-4	Lead-Free TQFP	100	COM
LCMXO640C-5TN100C	640	1.8V/2.5V/3.3V	74	-5	Lead-Free TQFP	100	COM
LCMXO640C-3MN100C	640	1.8V/2.5V/3.3V	74	-3	Lead-Free csBGA	100	COM
LCMXO640C-4MN100C	640	1.8V/2.5V/3.3V	74	-4	Lead-Free csBGA	100	COM
LCMXO640C-5MN100C	640	1.8V/2.5V/3.3V	74	-5	Lead-Free csBGA	100	COM
LCMXO640C-3TN144C	640	1.8V/2.5V/3.3V	113	-3	Lead-Free TQFP	144	COM
LCMXO640C-4TN144C	640	1.8V/2.5V/3.3V	113	-4	Lead-Free TQFP	144	COM
LCMXO640C-5TN144C	640	1.8V/2.5V/3.3V	113	-5	Lead-Free TQFP	144	COM
LCMXO640C-3MN132C	640	1.8V/2.5V/3.3V	101	-3	Lead-Free csBGA	132	COM
LCMXO640C-4MN132C	640	1.8V/2.5V/3.3V	101	-4	Lead-Free csBGA	132	COM
LCMXO640C-5MN132C	640	1.8V/2.5V/3.3V	101	-5	Lead-Free csBGA	132	COM
LCMXO640C-3BN256C	640	1.8V/2.5V/3.3V	159	-3	Lead-Free caBGA	256	COM
LCMXO640C-4BN256C	640	1.8V/2.5V/3.3V	159	-4	Lead-Free caBGA	256	COM
LCMXO640C-5BN256C	640	1.8V/2.5V/3.3V	159	-5	Lead-Free caBGA	256	COM
LCMXO640C-3FTN256C	640	1.8V/2.5V/3.3V	159	-3	Lead-Free ftBGA	256	COM
LCMXO640C-4FTN256C	640	1.8V/2.5V/3.3V	159	-4	Lead-Free ftBGA	256	COM
LCMXO640C-5FTN256C	640	1.8V/2.5V/3.3V	159	-5	Lead-Free ftBGA	256	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO1200C-3TN100C	1200	1.8V/2.5V/3.3V	73	-3	Lead-Free TQFP	100	COM
LCMXO1200C-4TN100C	1200	1.8V/2.5V/3.3V	73	-4	Lead-Free TQFP	100	COM
LCMXO1200C-5TN100C	1200	1.8V/2.5V/3.3V	73	-5	Lead-Free TQFP	100	COM
LCMXO1200C-3TN144C	1200	1.8V/2.5V/3.3V	113	-3	Lead-Free TQFP	144	COM
LCMXO1200C-4TN144C	1200	1.8V/2.5V/3.3V	113	-4	Lead-Free TQFP	144	COM
LCMXO1200C-5TN144C	1200	1.8V/2.5V/3.3V	113	-5	Lead-Free TQFP	144	COM
LCMXO1200C-3MN132C	1200	1.8V/2.5V/3.3V	101	-3	Lead-Free csBGA	132	COM
LCMXO1200C-4MN132C	1200	1.8V/2.5V/3.3V	101	-4	Lead-Free csBGA	132	COM
LCMXO1200C-5MN132C	1200	1.8V/2.5V/3.3V	101	-5	Lead-Free csBGA	132	COM
LCMXO1200C-3BN256C	1200	1.8V/2.5V/3.3V	211	-3	Lead-Free caBGA	256	COM
LCMXO1200C-4BN256C	1200	1.8V/2.5V/3.3V	211	-4	Lead-Free caBGA	256	COM
LCMXO1200C-5BN256C	1200	1.8V/2.5V/3.3V	211	-5	Lead-Free caBGA	256	COM
LCMXO1200C-3FTN256C	1200	1.8V/2.5V/3.3V	211	-3	Lead-Free ftBGA	256	COM
LCMXO1200C-4FTN256C	1200	1.8V/2.5V/3.3V	211	-4	Lead-Free ftBGA	256	COM
LCMXO1200C-5FTN256C	1200	1.8V/2.5V/3.3V	211	-5	Lead-Free ftBGA	256	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO2280C-3TN100C	2280	1.8V/2.5V/3.3V	73	-3	Lead-Free TQFP	100	COM
LCMXO2280C-4TN100C	2280	1.8V/2.5V/3.3V	73	-4	Lead-Free TQFP	100	COM
LCMXO2280C-5TN100C	2280	1.8V/2.5V/3.3V	73	-5	Lead-Free TQFP	100	COM
LCMXO2280C-3TN144C	2280	1.8V/2.5V/3.3V	113	-3	Lead-Free TQFP	144	COM
LCMXO2280C-4TN144C	2280	1.8V/2.5V/3.3V	113	-4	Lead-Free TQFP	144	COM
LCMXO2280C-5TN144C	2280	1.8V/2.5V/3.3V	113	-5	Lead-Free TQFP	144	COM
LCMXO2280C-3MN132C	2280	1.8V/2.5V/3.3V	101	-3	Lead-Free csBGA	132	COM
LCMXO2280C-4MN132C	2280	1.8V/2.5V/3.3V	101	-4	Lead-Free csBGA	132	COM
LCMXO2280C-5MN132C	2280	1.8V/2.5V/3.3V	101	-5	Lead-Free csBGA	132	COM
LCMXO2280C-3BN256C	2280	1.8V/2.5V/3.3V	211	-3	Lead-Free caBGA	256	COM
LCMXO2280C-4BN256C	2280	1.8V/2.5V/3.3V	211	-4	Lead-Free caBGA	256	COM
LCMXO2280C-5BN256C	2280	1.8V/2.5V/3.3V	211	-5	Lead-Free caBGA	256	COM
LCMXO2280C-3FTN256C	2280	1.8V/2.5V/3.3V	211	-3	Lead-Free ftBGA	256	COM
LCMXO2280C-4FTN256C	2280	1.8V/2.5V/3.3V	211	-4	Lead-Free ftBGA	256	COM
LCMXO2280C-5FTN256C	2280	1.8V/2.5V/3.3V	211	-5	Lead-Free ftBGA	256	COM
LCMXO2280C-3FTN324C	2280	1.8V/2.5V/3.3V	271	-3	Lead-Free ftBGA	324	COM
LCMXO2280C-4FTN324C	2280	1.8V/2.5V/3.3V	271	-4	Lead-Free ftBGA	324	COM
LCMXO2280C-5FTN324C	2280	1.8V/2.5V/3.3V	271	-5	Lead-Free ftBGA	324	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO256E-3TN100C	256	1.2V	78	-3	Lead-Free TQFP	100	COM
LCMXO256E-4TN100C	256	1.2V	78	-4	Lead-Free TQFP	100	COM
LCMXO256E-5TN100C	256	1.2V	78	-5	Lead-Free TQFP	100	COM
LCMXO256E-3MN100C	256	1.2V	78	-3	Lead-Free csBGA	100	COM
LCMXO256E-4MN100C	256	1.2V	78	-4	Lead-Free csBGA	100	COM
LCMXO256E-5MN100C	256	1.2V	78	-5	Lead-Free csBGA	100	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO640E-3TN100C	640	1.2V	74	-3	Lead-Free TQFP	100	COM
LCMXO640E-4TN100C	640	1.2V	74	-4	Lead-Free TQFP	100	COM
LCMXO640E-5TN100C	640	1.2V	74	-5	Lead-Free TQFP	100	COM
LCMXO640E-3MN100C	640	1.2V	74	-3	Lead-Free csBGA	100	COM
LCMXO640E-4MN100C	640	1.2V	74	-4	Lead-Free csBGA	100	COM
LCMXO640E-5MN100C	640	1.2V	74	-5	Lead-Free csBGA	100	COM
LCMXO640E-3TN144C	640	1.2V	113	-3	Lead-Free TQFP	144	COM
LCMXO640E-4TN144C	640	1.2V	113	-4	Lead-Free TQFP	144	COM
LCMXO640E-5TN144C	640	1.2V	113	-5	Lead-Free TQFP	144	COM
LCMXO640E-3MN132C	640	1.2V	101	-3	Lead-Free csBGA	132	COM
LCMXO640E-4MN132C	640	1.2V	101	-4	Lead-Free csBGA	132	COM
LCMXO640E-5MN132C	640	1.2V	101	-5	Lead-Free csBGA	132	COM
LCMXO640E-3BN256C	640	1.2V	159	-3	Lead-Free caBGA	256	COM
LCMXO640E-4BN256C	640	1.2V	159	-4	Lead-Free caBGA	256	COM
LCMXO640E-5BN256C	640	1.2V	159	-5	Lead-Free caBGA	256	COM
LCMXO640E-3FTN256C	640	1.2V	159	-3	Lead-Free ftBGA	256	COM
LCMXO640E-4FTN256C	640	1.2V	159	-4	Lead-Free ftBGA	256	COM
LCMXO640E-5FTN256C	640	1.2V	159	-5	Lead-Free ftBGA	256	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO1200E-3TN100C	1200	1.2V	73	-3	Lead-Free TQFP	100	COM
LCMXO1200E-4TN100C	1200	1.2V	73	-4	Lead-Free TQFP	100	COM
LCMXO1200E-5TN100C	1200	1.2V	73	-5	Lead-Free TQFP	100	COM
LCMXO1200E-3TN144C	1200	1.2V	113	-3	Lead-Free TQFP	144	COM
LCMXO1200E-4TN144C	1200	1.2V	113	-4	Lead-Free TQFP	144	COM
LCMXO1200E-5TN144C	1200	1.2V	113	-5	Lead-Free TQFP	144	COM
LCMXO1200E-3MN132C	1200	1.2V	101	-3	Lead-Free csBGA	132	COM
LCMXO1200E-4MN132C	1200	1.2V	101	-4	Lead-Free csBGA	132	COM
LCMXO1200E-5MN132C	1200	1.2V	101	-5	Lead-Free csBGA	132	COM
LCMXO1200E-3BN256C	1200	1.2V	211	-3	Lead-Free caBGA	256	COM
LCMXO1200E-4BN256C	1200	1.2V	211	-4	Lead-Free caBGA	256	COM
LCMXO1200E-5BN256C	1200	1.2V	211	-5	Lead-Free caBGA	256	COM
LCMXO1200E-3FTN256C	1200	1.2V	211	-3	Lead-Free ftBGA	256	COM
LCMXO1200E-4FTN256C	1200	1.2V	211	-4	Lead-Free ftBGA	256	COM
LCMXO1200E-5FTN256C	1200	1.2V	211	-5	Lead-Free ftBGA	256	COM

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO2280E-3TN100C	2280	1.2V	73	-3	Lead-Free TQFP	100	COM
LCMXO2280E-4TN100C	2280	1.2V	73	-4	Lead-Free TQFP	100	COM
LCMXO2280E-5TN100C	2280	1.2V	73	-5	Lead-Free TQFP	100	COM
LCMXO2280E-3TN144C	2280	1.2V	113	-3	Lead-Free TQFP	144	COM
LCMXO2280E-4TN144C	2280	1.2V	113	-4	Lead-Free TQFP	144	COM
LCMXO2280E-5TN144C	2280	1.2V	113	-5	Lead-Free TQFP	144	COM
LCMXO2280E-3MN132C	2280	1.2V	101	-3	Lead-Free csBGA	132	COM
LCMXO2280E-4MN132C	2280	1.2V	101	-4	Lead-Free csBGA	132	COM
LCMXO2280E-5MN132C	2280	1.2V	101	-5	Lead-Free csBGA	132	COM
LCMXO2280E-3BN256C	2280	1.2V	211	-3	Lead-Free caBGA	256	COM
LCMXO2280E-4BN256C	2280	1.2V	211	-4	Lead-Free caBGA	256	COM
LCMXO2280E-5BN256C	2280	1.2V	211	-5	Lead-Free caBGA	256	COM
LCMXO2280E-3FTN256C	2280	1.2V	211	-3	Lead-Free ftBGA	256	COM
LCMXO2280E-4FTN256C	2280	1.2V	211	-4	Lead-Free ftBGA	256	COM
LCMXO2280E-5FTN256C	2280	1.2V	211	-5	Lead-Free ftBGA	256	COM
LCMXO2280E-3FTN324C	2280	1.2V	271	-3	Lead-Free ftBGA	324	COM
LCMXO2280E-4FTN324C	2280	1.2V	271	-4	Lead-Free ftBGA	324	COM
LCMXO2280E-5FTN324C	2280	1.2V	271	-5	Lead-Free ftBGA	324	COM

Lead-Free Packaging

Industrial

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO256C-3TN100I	256	1.8V/2.5V/3.3V	78	-3	Lead-Free TQFP	100	IND
LCMXO256C-4TN100I	256	1.8V/2.5V/3.3V	78	-4	Lead-Free TQFP	100	IND
LCMXO256C-3MN100I	256	1.8V/2.5V/3.3V	78	-3	Lead-Free csBGA	100	IND
LCMXO256C-4MN100I	256	1.8V/2.5V/3.3V	78	-4	Lead-Free csBGA	100	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO640C-3TN100I	640	1.8V/2.5V/3.3V	74	-3	Lead-Free TQFP	100	IND
LCMXO640C-4TN100I	640	1.8V/2.5V/3.3V	74	-4	Lead-Free TQFP	100	IND
LCMXO640C-3MN100I	640	1.8V/2.5V/3.3V	74	-3	Lead-Free csBGA	100	IND
LCMXO640C-4MN100I	640	1.8V/2.5V/3.3V	74	-4	Lead-Free csBGA	100	IND
LCMXO640C-3TN144I	640	1.8V/2.5V/3.3V	113	-3	Lead-Free TQFP	144	IND
LCMXO640C-4TN144I	640	1.8V/2.5V/3.3V	113	-4	Lead-Free TQFP	144	IND
LCMXO640C-3MN132I	640	1.8V/2.5V/3.3V	101	-3	Lead-Free csBGA	132	IND
LCMXO640C-4MN132I	640	1.8V/2.5V/3.3V	101	-4	Lead-Free csBGA	132	IND
LCMXO640C-3BN256I	640	1.8V/2.5V/3.3V	159	-3	Lead-Free caBGA	256	IND
LCMXO640C-4BN256I	640	1.8V/2.5V/3.3V	159	-4	Lead-Free caBGA	256	IND
LCMXO640C-3FTN256I	640	1.8V/2.5V/3.3V	159	-3	Lead-Free ftBGA	256	IND
LCMXO640C-4FTN256I	640	1.8V/2.5V/3.3V	159	-4	Lead-Free ftBGA	256	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO1200C-3TN100I	1200	1.8V/2.5V/3.3V	73	-3	Lead-Free TQFP	100	IND
LCMXO1200C-4TN100I	1200	1.8V/2.5V/3.3V	73	-4	Lead-Free TQFP	100	IND
LCMXO1200C-3TN144I	1200	1.8V/2.5V/3.3V	113	-3	Lead-Free TQFP	144	IND
LCMXO1200C-4TN144I	1200	1.8V/2.5V/3.3V	113	-4	Lead-Free TQFP	144	IND
LCMXO1200C-3MN132I	1200	1.8V/2.5V/3.3V	101	-3	Lead-Free csBGA	132	IND
LCMXO1200C-4MN132I	1200	1.8V/2.5V/3.3V	101	-4	Lead-Free csBGA	132	IND
LCMXO1200C-3BN256I	1200	1.8V/2.5V/3.3V	211	-3	Lead-Free caBGA	256	IND
LCMXO1200C-4BN256I	1200	1.8V/2.5V/3.3V	211	-4	Lead-Free caBGA	256	IND
LCMXO1200C-3FTN256I	1200	1.8V/2.5V/3.3V	211	-3	Lead-Free ftBGA	256	IND
LCMXO1200C-4FTN256I	1200	1.8V/2.5V/3.3V	211	-4	Lead-Free ftBGA	256	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO2280C-3TN100I	2280	1.8V/2.5V/3.3V	73	-3	Lead-Free TQFP	100	IND
LCMXO2280C-4TN100I	2280	1.8V/2.5V/3.3V	73	-4	Lead-Free TQFP	100	IND
LCMXO2280C-3TN144I	2280	1.8V/2.5V/3.3V	113	-3	Lead-Free TQFP	144	IND
LCMXO2280C-4TN144I	2280	1.8V/2.5V/3.3V	113	-4	Lead-Free TQFP	144	IND
LCMXO2280C-3MN132I	2280	1.8V/2.5V/3.3V	101	-3	Lead-Free csBGA	132	IND
LCMXO2280C-4MN132I	2280	1.8V/2.5V/3.3V	101	-4	Lead-Free csBGA	132	IND
LCMXO2280C-3BN256I	2280	1.8V/2.5V/3.3V	211	-3	Lead-Free caBGA	256	IND
LCMXO2280C-4BN256I	2280	1.8V/2.5V/3.3V	211	-4	Lead-Free caBGA	256	IND
LCMXO2280C-3FTN256I	2280	1.8V/2.5V/3.3V	211	-3	Lead-Free ftBGA	256	IND
LCMXO2280C-4FTN256I	2280	1.8V/2.5V/3.3V	211	-4	Lead-Free ftBGA	256	IND
LCMXO2280C-3FTN324I	2280	1.8V/2.5V/3.3V	271	-3	Lead-Free ftBGA	324	IND
LCMXO2280C-4FTN324I	2280	1.8V/2.5V/3.3V	271	-4	Lead-Free ftBGA	324	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO256E-3TN100I	256	1.2V	78	-3	Lead-Free TQFP	100	IND
LCMXO256E-4TN100I	256	1.2V	78	-4	Lead-Free TQFP	100	IND
LCMXO256E-3MN100I	256	1.2V	78	-3	Lead-Free csBGA	100	IND
LCMXO256E-4MN100I	256	1.2V	78	-4	Lead-Free csBGA	100	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO640E-3TN100I	640	1.2V	74	-3	Lead-Free TQFP	100	IND
LCMXO640E-4TN100I	640	1.2V	74	-4	Lead-Free TQFP	100	IND
LCMXO640E-3MN100I	640	1.2V	74	-3	Lead-Free csBGA	100	IND
LCMXO640E-4MN100I	640	1.2V	74	-4	Lead-Free csBGA	100	IND
LCMXO640E-3TN144I	640	1.2V	113	-3	Lead-Free TQFP	144	IND
LCMXO640E-4TN144I	640	1.2V	113	-4	Lead-Free TQFP	144	IND
LCMXO640E-3MN132I	640	1.2V	101	-3	Lead-Free csBGA	132	IND
LCMXO640E-4MN132I	640	1.2V	101	-4	Lead-Free csBGA	132	IND
LCMXO640E-3BN256I	640	1.2V	159	-3	Lead-Free caBGA	256	IND
LCMXO640E-4BN256I	640	1.2V	159	-4	Lead-Free caBGA	256	IND
LCMXO640E-3FTN256I	640	1.2V	159	-3	Lead-Free ftBGA	256	IND
LCMXO640E-4FTN256I	640	1.2V	159	-4	Lead-Free ftBGA	256	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO1200E-3TN100I	1200	1.2V	73	-3	Lead-Free TQFP	100	IND
LCMXO1200E-4TN100I	1200	1.2V	73	-4	Lead-Free TQFP	100	IND
LCMXO1200E-3TN144I	1200	1.2V	113	-3	Lead-Free TQFP	144	IND
LCMXO1200E-4TN144I	1200	1.2V	113	-4	Lead-Free TQFP	144	IND
LCMXO1200E-3MN132I	1200	1.2V	101	-3	Lead-Free csBGA	132	IND
LCMXO1200E-4MN132I	1200	1.2V	101	-4	Lead-Free csBGA	132	IND
LCMXO1200E-3BN256I	1200	1.2V	211	-3	Lead-Free caBGA	256	IND
LCMXO1200E-4BN256I	1200	1.2V	211	-4	Lead-Free caBGA	256	IND
LCMXO1200E-3FTN256I	1200	1.2V	211	-3	Lead-Free ftBGA	256	IND
LCMXO1200E-4FTN256I	1200	1.2V	211	-4	Lead-Free ftBGA	256	IND

Part Number	LUTs	Supply Voltage	I/Os	Grade	Package	Pins	Temp.
LCMXO2280E-3TN100I	2280	1.2V	73	-3	Lead-Free TQFP	100	IND
LCMXO2280E-4TN100I	2280	1.2V	73	-4	Lead-Free TQFP	100	IND
LCMXO2280E-3TN144I	2280	1.2V	113	-3	Lead-Free TQFP	144	IND
LCMXO2280E-4TN144I	2280	1.2V	113	-4	Lead-Free TQFP	144	IND
LCMXO2280E-3MN132I	2280	1.2V	101	-3	Lead-Free csBGA	132	IND
LCMXO2280E-4MN132I	2280	1.2V	101	-4	Lead-Free csBGA	132	IND
LCMXO2280E-3BN256I	2280	1.2V	211	-3	Lead-Free caBGA	256	IND
LCMXO2280E-4BN256I	2280	1.2V	211	-4	Lead-Free caBGA	256	IND
LCMXO2280E-3FTN256I	2280	1.2V	211	-3	Lead-Free ftBGA	256	IND
LCMXO2280E-4FTN256I	2280	1.2V	211	-4	Lead-Free ftBGA	256	IND
LCMXO2280E-3FTN324I	2280	1.2V	271	-3	Lead-Free ftBGA	324	IND
LCMXO2280E-4FTN324I	2280	1.2V	271	-4	Lead-Free ftBGA	324	IND

MachXO Family Data Sheet Supplemental Information

For Further Information

A variety of technical notes for the MachXO family are available on the Lattice web site.

- TN1091, MachXO sysIO Usage Guide
- TN1089, MachXO sysCLOCK Design and Usage Guide
- TN1092, Memory Usage Guide for MachXO Devices
- TN1090, Power Estimation and Management for MachXO Devices
- TN1086, MachXO JTAG Programming and Configuration User's Guide
- TN1087, Minimizing System Interruption During Configuration Using TransFR Technology
- TN1097, MachXO Density Migration
- AN8066, Boundary Scan Testability with Lattice sysIO Capability

For further information on interface standards refer to the following web sites:

- JEDEC Standards (LVTTL, LVCMOS): www.jedec.org
- PCI: www.pcisig.com

MachXO Family Data Sheet Revision History

July 2010 **Data Sheet DS1002**

Revision History

Date	Version	Section	Change Summary
February 2005	01.0	—	Initial release.
October 2005	01.1	Introduction	Distributed RAM information in family table updated. Added footnote 1 - fpBGA packaging to the family selection guide.
		Architecture	sysIO Buffer section updated.
			Hot Socketing section updated.
			Sleep Mode section updated.
			SLEEP Pin Characteristics section updated.
			Oscillator section updated.
			Security section updated.
		DC and Switching Characteristics	Recommended Operating Conditions table updated.
			DC Electrical Characteristics table updated.
			Supply Current (Sleep Mode) table added with LCMXO256/640 data.
			Supply Current (Standby) table updated with LCMXO256/640 data.
			Initialization Supply Current table updated with LCMXO256/640 data.
			Programming and Erase Flash Supply Current table updated with LCMXO256/640 data.
			Register-to-Register Performance table updated (rev. A 0.16).
			External Switching Characteristics table updated (rev. A 0.16).
			Internal Timing Parameter table updated (rev. A 0.16).
			Family Timing Adders updated (rev. A 0.16).
			sysCLOCK Timingupdated (rev. A 0.16).
			MachXO "C" Sleep Mode Timing updated (A 0.16).
			JTAG Port Timing Specification updated (rev. A 0.16).
		Pinout Information	SLEEPIN description updated.
			Pin Information Summary updated.
			Power Supply and NC Connection table has been updated.
			Logic Signal Connection section has been updated to include all devices/packages.
		Ordering Information	Part Number Description section has been updated.
			Ordering Part Number section has been updated (added LCMXO256C/ LCMXO640C "4W").
		Supplemental Information	MachXO Density Migration Technical Note (TN1097) added.
November 2005	01.2	Pinout Information	Added “Power Supply and NC Connections” summary information for LCMXO1200 and LCMXO2280 in 100 TQFP package.
December 2005	01.3	DC and Switching Characteristics	Supply Current (Standby) table updated with LCMXO1200/2280 data.
		Ordering Information	Ordering Part Number section updated (added LCMXO2280C "4W").
April 2006	02.0	Introduction	Introduction paragraphs updated.
		Architecture	Architecture Overview paragraphs updated.

Date	Version	Section	Change Summary
April 2006 (cont.)	02.0 (cont.)	Architecture (cont.)	"Top View of the MachXO1200 Device" figure updated.
			"Top View of the MachXO640 Device" figure updated.
			"Top View of the MachXO256 Device" figure updated.
			"Slice Diagram" figure updated.
			Slice Signal Descriptions table updated.
			Routing section updated.
			sysCLOCK Phase Lockecd Loops (PLLs) section updated.
			PLL Diagram updated.
			PLL Signal Descriptions table updated.
			sysMEM Memory section has been updated.
			PIO Groups section has been updated.
			PIO section has been updated.
			MachXO PIO Block Diagram updated.
			Supported Input Standards table updated.
			MachXO Configuration and Programming diagram updated.
		DC and Switching Characteristics	Recommended Operating Conditions table - footnotes updated.
			MachXO256 and MachXO640 Hot Socketing Specifications - footnotes updated.
			Added MachXO1200 and MachXO2280 Hot Socketing Specifications table.
			DC Electrical Characteristics, footnotes have been updated.
			Supply Current (Sleep Mode) table has been updated, removed "4W" references. Footnotes have been updated.
			Supply Current (Standby) table and associated footnotes updated.
			Intialization Supply Current table and footnotes updated.
			Programming and Erase Flash Supply Current table and associated footnotes have been updatd.
			Register-to-Register Performance table updated (rev. A 0.19).
			MachXO External Switching Characteristics updated (rev. A 0.19).
			MachXO Internal Timing Parameters updated (rev. A 0.19).
			MachXO Family Timing Adders updated (rev. A 0.19).
			sysCLOCK Timing updated (rev. A 0.19).
			MachXO "C" Sleep Mode Timing updated (A 0.19).
			JTAG Port Timing Specification updated (rev. A 0.19).
			Test Fixture Required Components table updated.
		Pinout Information	Signal Descriptions have been updated.
			Pin Information Summary has been updated. Footnote has been added.
			Power Supply and NC Connection table has been updated.
			Logic Signal Connections have been updated (PCLKTx_x --> PCLKx_x)
		Ordering Information	Removed "4W" references.
			Added 256-ftBGA Ordering Part Numbers for MachXO640.
May 2006	02.1	Pinout Information	Removed [LOC][0]_PLL_RST from Signal Description table.
			PCLK footnote has been added to all appropriate pins.
August 2006	02.2	Multiple	Removed 256 fpBGA information for MachXO640.

Date	Version	Section	Change Summary
November 2006	02.3	DC and Switching Characteristics	Corrections to MachXO "C" Sleep Mode Timing table - value for $t_{WSLEEPN}$ (400ns) changed from max. to min. Value for t_{WAWAKE} (100ns) changed from min. to max.
			Added Flash Download Time table.
December 2006	02.4	Architecture	EBR Asynchronous Reset section added.
		Pinout Information	Power Supply and NC table: Pin/Ball orientation footnotes added.
February 2007	02.5	Architecture	Updated EBR Asynchronous Reset section.
August 2007	02.6	DC and Switching Characteristics	Updated sysIO Single-Ended DC Electrical Characteristics table.
November 2007	02.7	DC and Switching Characteristics	Added JTAG Port Timing Waveforms diagram.
		Pinout Information	Added Thermal Management text section.
		Supplemental Information	Updated title list.
June 2009	02.8	Introduction	Added 0.8-mm 256-pin caBGA package to MachXO Family Selection Guide table.
		Pinout Information	Added Logic Signal Connections table for 0.8-mm 256-pin caBGA package.
		Ordering Information	Updated Part Number Description diagram and Ordering Part Number tables with 0.8-mm 256-pin caBGA package information.
July 2010	02.9	DC and Switching Characteristics	Updated sysCLOCK PLL Timing table.

Section II. MachXO Family Technical Notes

MachXO sysIO Usage Guide

September 2010 Technical Note TN1091

Introduction

The MachXO™ sysIO™ buffers provide the ability to easily interface with other devices using advanced system I/O standards. This technical note describes the sysIO standards available and how they can be implemented using Lattice design software.

sysIO Buffer Overview

The MachXO sysIO interface contains multiple Programmable I/O Cell (PIC) blocks. Each PIC contains two Programmable I/Os (PIOs). Two adjacent PIOs can be joined to provide a differential I/O pair (labeled as "T" and "C"). In the MachXO256, MachXO1200 and MachXO2280 devices, the PIOs are arranged in groups of six (PIOA, PIOB, PIOC, PIOD, PIOE, PIOF) on the top and bottom sides of the device and in groups of four (PIOA, PIOB, PIOC, PIOD) and two (PIOA, PIOB) on the left and right sides of the device. In the MachXO640 device, the PIOs are arranged in groups of six (PIOA, PIOB, PIOC, PIOD, PIOE, PIOF) on the top and bottom sides and in groups of four (PIOA, PIOB, PIOC, PIOD) on the left and right sides of the device.

The larger two devices, MachXO1200 and MachXO2280, support single-ended, differential receiver and differential output sysIO buffers. The two smaller devices, MachXO256 and MachXO640, support single-ended sysIO buffers. For more information on the architecture of the sysIO buffer please refer to the device data sheets.

Supported sysIO Standards

The MachXO sysIO buffer supports both single-ended and differential standards. Single-ended standards can be further subdivided into LVCMOS, LVTTL and other standards. The buffers support the LVTTL, LVCMOS 1.2, 1.5, 1.8, 2.5 and 3.3V standards. In the LVCMOS and LVTTL modes, the buffer has individually configurable options for drive strength, bus maintenance (weak pull-up, weak pull-down or bus-keeper latch) on I/O buffers. MachXO1200 and MachXO2280 devices also support differential standards like LVDS, RSDS, BLVDS and LVPECL. Table 8-1 lists the sysIO standards supported in the MachXO devices.

Table 8-1. sysIO Standards Supported

Standard	V_{CCIO} (V)		
	Min.	Typ.	Max.
LVCMOS 3.3	3.135	3.3	3.465
LVCMOS 2.5	2.375	2.5	2.625
LVCMOS 1.8	1.71	1.8	1.89
LVCMOS 1.5	1.425	1.5	1.575
LVCMOS 1.2	1.14	1.2	1.26
LVTTL	3.135	3.3	3.465
PCI[3]	3.135	3.3	3.465
LVDS[1, 2]	2.375	2.5	2.625
LVPECL[1]	3.135	3.3	3.465
BLVDS[1]	2.375	2.5	2.625
RSDS[1]	2.375	2.5	2.625

1. Inputs on chip. Outputs are implemented with the addition of external resistors.
2. MachXO1200 and MachXO2280 devices have dedicated LVDS buffers
3. Input on the top bank of the MachXO1200 and MachXO2280 only.

sysIO Banking Scheme

The MachXO family has a non-homogeneous I/O banking structure. The MachXO256 has two I/O banks, the MachXO640 has four I/O banks and the two largest devices, the MachXO1200 and the MachXO2280, have eight I/O banks. The figures below show the banking structures in each of the devices. Each sysIO bank has a V_{CCIO} supply voltage.

On the MachXO1200 and MachXO2280 devices, the top and bottom banks of the sysIO buffer pair consist of two single-ended output drivers, two single-ended and one differential input buffer. The sysIO buffers on the top side bank also supports PCI buffers. The left and right side sysIO buffer pairs, along with the two single-ended output and input drivers, a differential input and a differential driver on half the I/I/Os of the bank. On the MachXO640 and MachXO256 devices, all the banks of the sysIO buffer pair consists of two single-ended output drivers (with complementary outputs) and two single-ended and input buffers. All the banks will also support differential output buffers using external resistors. The two pads in the pair are described as “true” and “comp”, where the true pad is associated with the positive side of the differential input buffer and the comp (complementary) pad is associated with the negative side of the differential input buffer.

Figures 1, 2 and 3 show the banking schemes for the MachXO640, MachXO256 and MachXO1200/MachXO2280 devices respectively.

Figure 8-1. MachXO256 sysIO Banking

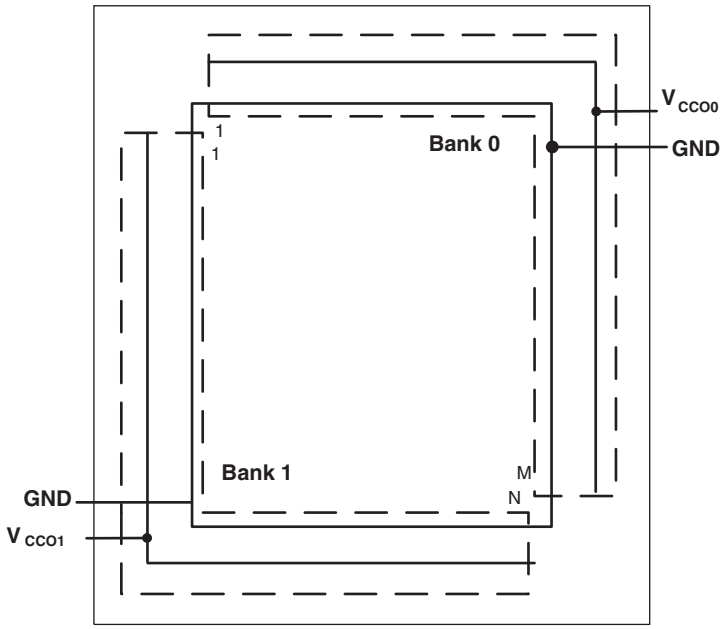


Figure 8-2. MachXO640 sysIO Banking

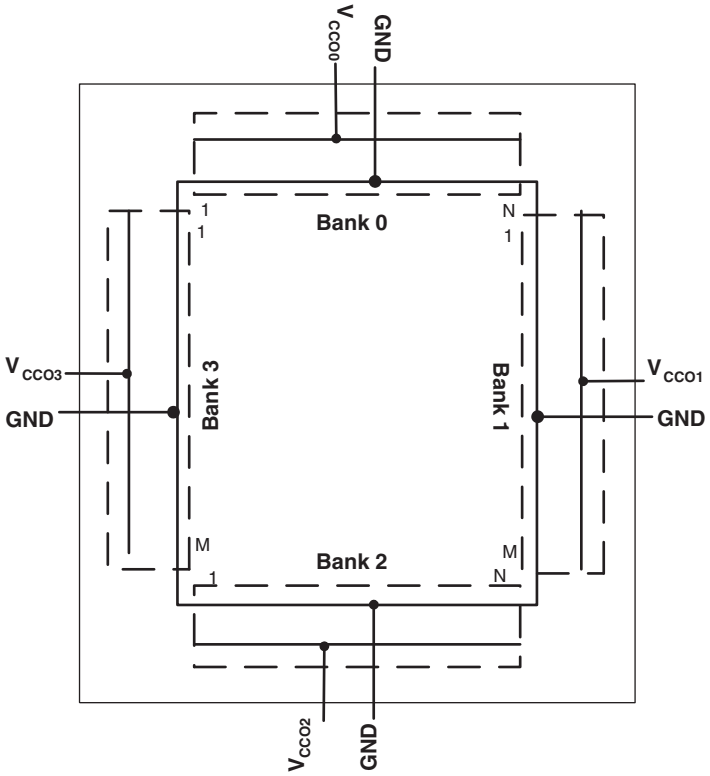
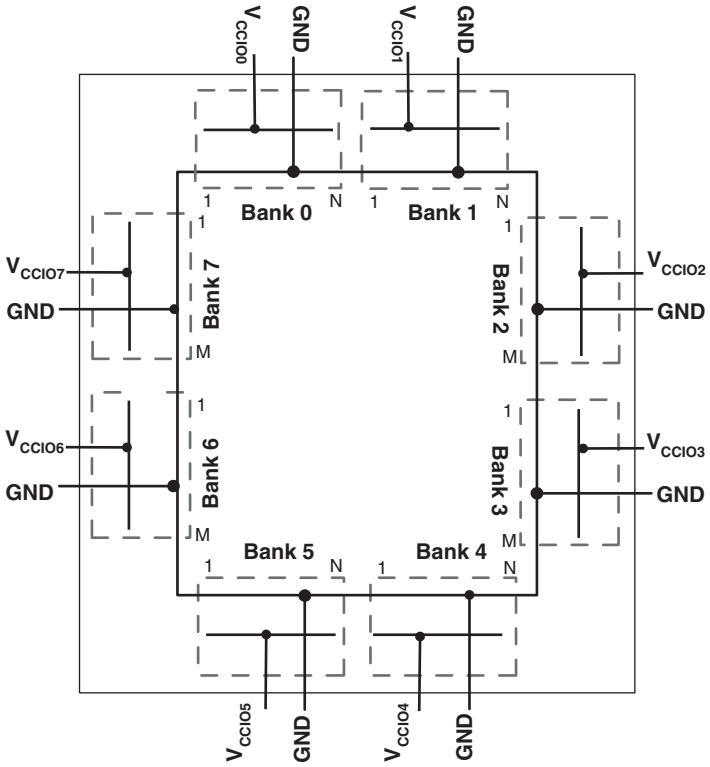


Figure 8-3. MachXO1200 and MachXO2280 sysIO Banking



V_{CCIO} (1.2V/1.5V/1.8V/2.5V/3.3V)

Each bank has a separate V_{CCIO} supply that powers the single-ended output drivers and the ratioed input buffers such as LVTTL, LVCMOS and PCI. LVTTL, LVCMOS3.3, LVCMOS2.5 and LVCMOS1.2 also have fixed threshold options allowing them to be placed in any bank and is independent of bank V_{CCIO}. The V_{CCIO} voltage applied to the bank determines the ratioed input standards that can be supported in that bank. It is also used to power the differential output drivers.

The V_{CCIO} of one of the banks is also used to power the JTAG pins (Bank1 for MachXO256, Bank2 for MachXO640 and Bank5 for MachXO1200 and MachXO2280 devices). Therefore, the threshold of the JTAG pins is determined by the V_{CCIO} of the JTAG bank.

V_{CCAUX} (3.3V)

In addition to the bank V_{CCIO} supplies, devices have a V_{CC} core logic power supply, and a V_{CCAUX} auxiliary supply that powers the differential and referenced input buffers. V_{CCAUX} is required because V_{CC} does not have enough headroom to satisfy the common-mode range requirements of these drivers and input buffers.

Mixed Voltage Support in a Bank

The MachXO sysIO buffer is connected to three parallel ratioed input buffers. These three parallel buffers are connected to V_{CCIO}, V_{CCAUX} and to V_{CC}, giving support for thresholds that track with V_{CCIO}, as well as fixed thresholds for 3.3V (V_{CCAUX}) and 1.2V (V_{CC}) inputs. This allows the input threshold for ratioed buffers to be assigned on a pin-by-pin basis, rather than being tracked with V_{CCIO}. This option is available for all 1.2V, 2.5V and 3.3V ratioed inputs and is independent of the bank V_{CCIO} voltage. For example, if the bank V_{CCIO} is 1.8V, it is possible to have 1.2V and 3.3V ratioed input buffers with fixed thresholds, as well as 2.5V ratioed inputs with tracking thresholds.

Prior to device configuration, the ratioed input thresholds always track the bank V_{CCIO}. This option only takes effect after configuration. Output standards within a bank are always set by V_{CCIO}. Table 8-2 shows the sysIO standards that the user can mix in the same bank.

Table 8-2. Mixed Voltage Support

	Input sysIO Standards					Output sysIO Standards				
V_{CCIO}	**1.2V**	**1.5V**	**1.8V**	**2.5V**	**3.3V**	**1.2V**	**1.5V**	**1.8V**	**2.5V**	**3.3V**
1.2V	Yes			Yes	Yes	Yes				
1.5V	Yes	Yes		Yes	Yes		Yes			
1.8V	Yes		Yes	Yes	Yes			Yes		
2.5V	Yes			Yes	Yes				Yes	
3.3V	Yes			Yes	Yes					Yes

sysIO Standards Supported in Each Bank

Table 8-3. I/O Standards Supported by Various Banks in the MachXO640 and MachXO256

Description	Bank 0	Bank1	Bank2	Bank3
I/O Buffer Type	Single-ended	Single-ended	Single-ended	Single-ended
Output Standards Supported	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 LVDS25E[1] LVPECL[1] BLVDS[1] RSDS[1]	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 LVDS25E[1] LVPECL[1] BLVDS[1] RSDS[1]	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 LVDS25E[1] LVPECL[1] BLVDS[1] RSDS[1]	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 LVDS25E[1] LVPECL[1] BLVDS[1] RSDS[1]
Inputs	All Single-ended	All Single-ended	All Single-ended	All Single-ended
Clock Inputs	All Single-ended	All Single-ended	All Single-ended	All Single-ended

1. These differential standards are implemented by using complementary LVCMOS driver with external resistor pack.
2. MachXO256 only has 2 banks, Banks 0 and Bank 1.

Table 8-4. I/O Standards Supported by I/O Pins on Each Side of the MachXO1200 and MachXO2280

Description	Top	Right	Bottom	Left
I/O Buffer Type	Single-ended	Single-ended and Differential	Single-ended	Single-ended and Differential
Output Standards Supported	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 PCI33 LVDS25E[1] LVPECL[1] BLVDS[1] RSDS[1]	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 LVDS25[2] LVDS25E[1] LVPECL[1] BLVDS[1] RSDS[1]	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 LVDS25E[1] LVPECL[1] BLVDS[1] RSDS[1]	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 LVDS25[2] LVDS25E[1] LVPECL[1] BLVDS[1] RSDS[1]
Inputs	All Single-ended, Differential	All Single-ended, Differential	All Single-ended, Differential	All Single-ended, Differential
Clock Inputs	All Single-ended, Differential (PLL input)	All Single-ended, Differential (PLL input)	All Single-ended, Differential (PLL input)	All Single-ended, Differential (PLL input)
PCI Support	PCI33 with Clamp			
LVDS Output Buffers		LVDS (3.5mA) Buffers		LVDS (3.5mA) Buffers

1. These differential standards are implemented by using complementary LVCMOS driver with external resistor pack.
2. These are supported on half the I/Os of the bank.

LVCMOS Buffer Configurations

All LVCMOS buffer have programmable pull, programmable drive and programmable slew configurations that can be set in the software.

Programmable PULLUP/PULLDOWN/BUSKEEPER

When configured as LVCMOS or LVTTL, each sysIO buffer has a weak pull-up, a weak pull-down resistor and a weak buskeeper (bus hold latch) available. Each I/O can independently be configured to have one of these features or none of them.

Programmable Drive

All LVCMOS and LVTTL single-ended drivers have programmable drive strength. This option can be set for each I/O independently. The table below lists the programmable drive strengths available for each I/O standard. The actual value will vary with the I/O voltage. The user must consider the maximum allowable current per bank and the package thermal limit current when selecting the drive strength.

Table 8-5. Programmable Drive Strength

Single-ended I/O Standard	Programmable Drive (mA)
LVCMOS12	2, 6
LVCMOS15	4, 8
LVCMOS18	4, 8, 12, 14
LVCMOS25	4, 8, 12, 14
LVCMOS33	4, 8, 12, 14
LVTTL	4, 8, 12, 16

Programmable Slew Rate

Each LVCMOS or LVTTL output buffer pin also has a programmable output slew rate control that can be configured for either low noise or high-speed performance. Each I/O pin also has an individual slew rate control. This allows a designer to specify slew rate control on a pin-by-pin basis. This slew rate control affects both the rising edge and the falling edges.

Open-Drain Control

Each LVCMOS and LVTTL output buffer can be configured to function as an open-drain output. The user can implement an open-drain output by turning on the OPENDRAIN attribute in the software.

Programmable PCICLAMP

Each sysIO buffer on the top bank of the MachXO1200 and MachXO2280 devices can be configured to support PCI33. The buffers also have a PCI clamp diode that will be turned on when the sysIO buffer is configured as PCI33.

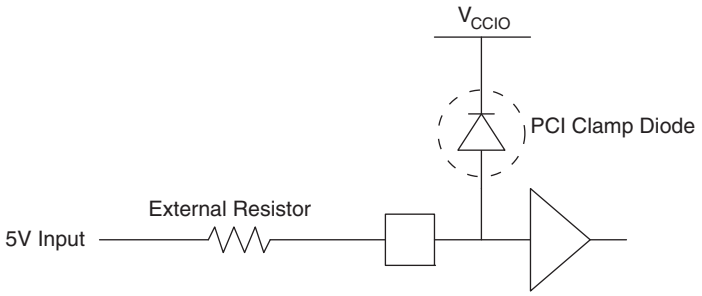
The PCI clamp is mainly used when implementing a 3.3V PCI interface. The PCI Specification revision 2.2 requires the use of clamping diodes for 3.3V operation. For more information on the PCI interface, please refer to the PCI specification, revision 2.2.

The PCI clamp can also be optionally turned on for LVCMOS and LVTTL sysIO buffers on the top bank of the MachXO1200 and MachXO2280 devices. In this case, the PCI clamp is used to implement a 5V input interface.

5V Input Interface Using the PCI Clamp Diode

All the I/Os on the top side of the MachXO1200 and MachXO2280 devices (Banks 0 and 1) have a clamp diode that is used to clamp the voltage at the input to V_{CCIO}. This clamp diode can be used along with an external resistor to make an input 5V tolerant.

Figure 8-4. 5V Input Interface Example



The value of this external resistor will depend on the PCI clamp diode characteristics. The voltage vs. current data across this diode can be found in the device IBIS model.

In order to interface to 5V input, it is recommended that the V_{CCIO} is set between 2.5V to 3.3V.

Below is an example of how to calculate the value of this external resistor when V_{CCIO} is 2.75V.

- Maximum voltage at input pin, V_{INMAX} = 3.75V (see the MachXO Family Data Sheet for more details)
- Bank V_{CCIO} = 2.75V
- Maximum voltage drop across clamp diode, $V_D = V_{INMAX} - V_{CCIO} = 3.75 - 2.75 = 1V$
- The current across the clamp diode at V_D can be found in the power clamp data of the IBIS file. Below is the power clamp portion of the IBIS file, for a LVCMOS3.3 input model with PCI Clamp turned on. When V_D is 1V, the clamp diode current is I_D = 382µA.

Table 8-6. Power Clamp Data from a Typical IBIS Model[1]

Voltage	I (max.)	Units
-1.40	408	µA
-1.30	402	µA
-1.20	395	µA
-1.10	389	µA
-1.00	382	µA
-0.90	380	µA
-0.80	380	µA
-0.70	380	µA
-0.60	380	µA
-0.50	380	µA
-0.40	353	µA
-0.30	51	µA
-0.20	3.8	µA
-0.10	0.23	µA
0.00	0.15	µA

1. Always consult the latest IBIS data available from Lattice. IBIS model information is available by device family on the Lattice web site.

- Assume the maximum output voltage of the driving device is V_{EXT} = 5.25V. The value of the external resistor can then be calculated as follows:

$$R_{EXT} = (V_{EXT} - V_{INMAX})/I_D = (5.25V - 3.75V)/382\mu A = 3.93 \text{ K ohm}$$

If the V_{CCIO} of the bank is increased, it will also increase the value of the external resistor required. Keep in mind that changing the Bank V_{CCIO} will change the value of the input threshold voltage.

Software sysIO Attributes

sysIO attributes can be specified in the HDL, using the Preference Editor GUI in ispLEVER®, the Spreadsheet View in Lattice Diamond™ design software, or in the ASCII Preference file (.perf) directly. Appendices A, B and C list examples of how these can be assigned using each of the methods mentioned above. This section describes in detail each of these attributes.

IO_TYPE

This is used to set the sysIO standard for an I/O. The V_{CCIO} required to set these I/O standards are embedded in the attribute names itself. There is no separate attribute to set the V_{CCIO} requirements. Table 6 and 7 list the available I/O types.

Table 8-7. IO_TYPE Attribute Values for MachXO640 and MachXO256 Devices

sysIO Signaling Standard	IO_TYPE
DEFAULT	LVCMOS25
RSDS	RSDS
Emulated LVDS 2.5V	LVDS25E[1]
Bus LVDS 2.5V	BLVDS25[1]
LVPECL 3.3V	LVPECL33[1]
LVTTL	LVTTL33
3.3V LVCMOS	LVCMOS33
2.5V LVCMOS	LVCMOS25
1.8V LVCMOS	LVCMOS18
1.5V LVCMOS	LVCMOS15
1.2V LVCMOS	LVCMOS12

1. These differential standards are implemented by using complementary LVCMOS driver with external resistor pack.

Table 8-8. IO_TYPE Attribute Values for MachXO1200 and MachXO2280 Devices

sysIO Signaling Standard	IO_TYPE
DEFAULT	LVCMOS25
LVDS 2.5V	LVDS25[2]
RSDS	RSDS
Emulated LVDS 2.5V	LVDS25E[1]
Bus LVDS 2.5V	BLVDS25[1]
LVPECL 3.3V	LVPECL33[1]
LVTTL	LVTTL33
3.3V LVCMOS	LVCMOS33
2.5V LVCMOS	LVCMOS25
1.8V LVCMOS	LVCMOS18
1.5V LVCMOS	LVCMOS15
1.2V LVCMOS	LVCMOS12
3.3V PCI	PCI33[3]

1. These differential standards are implemented by using a complementary LVCMOS driver with external resistor pack.
2. Available on 50% of the I/Os on the right and left side banks.
3. Available on the top side bank.

OPENDRAIN

LVCMOS and LVTTL I/O standards can be set to open-drain configuration by using the OPENDRAIN attribute.

Values: ON, OFF

Default: OFF

DRIVE

The Drive Strength attribute is available for LVTTL and LVCMOS output standards. These can be set or each I/O pin individually. The programmable drive available on a pad will depend on the V_{CCIO}. Table 8-9 shows the drive strength available for different I/O standards and the defaults for each of them.

Table 8-9. Programmable Drive Strength Values at Various V_{CCIO} Voltages

Output Standard	Drive (mA)	Default (mA)
LVTTL	4, 8, 12, 16	8
LVCMOS33	4, 8, 12, 14	8
LVCMOS25	4, 8, 12, 14	12
LVCMOS18	4, 8, 12, 14	8
LVCMOS15	4, 8	8
LVCMOS12	2, 6	6

PULLMODE

The PULLMODE attribute is available for all the LVTLL and LVCMOS inputs and outputs. This attribute can be enabled for each I/O independently.

Values: UP, DOWN, NONE, KEEPER

Default: UP

Table 8-10. PULLMODE Values

PULL Options	PULLMODE Value
Pull up (Default)	UP
Pull Down	DOWN
Bus Keeper	KEEPER
Pull Off	NONE

PCICLAMP

PCI33 inputs and outputs are available on the top bank of the MachXO1200 and MachXO2280 devices. When an I/O is configured as a PCI33 standard, the PCICLAMP is enabled for that buffer. The PCICLAMP is also available for all LVCMOS33 and LVTTL inputs. This is used to implement a 5V input interface.

Values: ON, OFF

Default: ON (for PCI33 input and output)
OFF (for LVCMOS33 and LVTTL inputs)

SLEWRATE

The SLEWRATE attribute is available for all LVTTL and LVCMOS output drivers. Each I/O pin has an individual slew rate control. This allows designers to specify the slew rate control on pin-by-pin basis.

Values: FAST, SLOW

Default: FAST

LOC

This attribute can be used to make pin assignments to the I/O ports in the design. This attributes is only used when the pin assignments are made in HDL source. Users can also assign pins directly using the Preference Editor GUI in ispLEVER or the Spreadsheet View in Diamond. The appendices of this document explain this in greater detail.

Design Considerations and Usage

This section discusses some rules and considerations for designing with the MachXO sysIO buffer.

Banking Rules

- If the V_{CCIO} for any bank is set to 3.3V, it is recommended that it be connected to the same power supply as V_{CCAUX}, thus minimizing leakage.
- If V_{CCIO} for any bank is set to 1.2V, it is recommended that it be connected to the same power supply as V_{CC}, thus minimizing leakage.
- PCI I/O standards with PCI clamps are only available on the top bank (Banks 0 and 1) on the MachXO1200 and MachXO2280 devices.
- PCI I/O standards with PCI clamps are not available on the MachXO640 and MachXO256 devices.
- Only 50% of the I/Os on the left and right banks of the MachXO1200 and MachXO2280 devices support a True LVDS driver. True LVDS receivers are available on all banks for the MachXO1200 and MachXO2280 devices.
- All banks support emulated differential outputs using an external resistor pack and complementary LVCMOS driver.

Zero Hold Time

The user can achieve a zero hold time for his or her inputs by specifying a zero hold time preference in the software. The software will add additional delays to the input path in order to achieve this zero hold time.

Fast Output Path

The MachXO devices have a dedicated fast output I/O connection from the adjacent PFUs to the I/O buffers within the PIO. This connection provides faster output delays for faster clock-to-output propagation delays and pin-to-pin propagation delays. The software will automatically use this fast output path to achieve faster t_{CO} and t_{PD} requirements. You can fulfill the t_{CO} and t_{PD} requirement by assigning these preferences in the Preference Editor in the software.

Dedicated Pins

Global Set Rest (GSR)

The GSR in the MachXO devices is an asynchronous Global Set Reset. This signal is can be programmed to come from either the PFU logic or from the dedicated GSR input pad. When the software does not see any logic associated with the GSR, then it will automatically use the dedicated GSR input path. This provides faster timing. When the reset used is a logic reset, the polarity is programmable. When the dedicated GSR input from the GSR pad is used, the polarity has to be active low.

Tristate All (TSALLPAD)
All MachXO devices have a dedicated TSALLPAD pin that is used to enable or disable the tristate control to all the output buffers. By default, the pin will function as an I/O unless programmed to be a TSALLPAD. This signal also has programmable global polarity control. By default, the polarity is active high. This global tristate control signal can also be generated using user logic.

When the TSALLPAD is enabled, the software will implement the tristate control using the TSALL software primitive. The polarity control of the TSALLPAD will control the polarity of TSALL.

When TSALLPAD is not used in the design, but is required for test purposes, the TSALL primitive can be instantiated in the HDL and the TSALLPAD is connected to the input of this primitive.

Differential I/O Implementation

MachXO devices support a variety of differential standards, as detailed in the following sections.

LVDS (MachXO1200 and MachXO2280)
True LVDS (LVDS25) drivers are available on 50% of the I/Os on the left and right sides of the MachXO1200 and MachXO2280 devices. LVDS input support is available on all sides of the MachXO1200 and MachXO2280 devices.

LVDSE
The single-ended sysIO buffer pairs in all the MachXO devices support LVDS output drivers using complementary LVCMOS drivers with external resistors (LVDS25E) on all four sides of the device.

The MachXO1200 and MachXO2280 devices also support LVDSE inputs on all four sides of the device.

Please refer to the MachXO Family Data Sheet for a detailed explanation of these LVDS implementations.

BLVDS
All single-ended sysIO buffer pairs in all the MachXO devices support Bus-LVDS output driver using complementary LVCMOS drivers with external resistors on all the four sides of the device.

The MachXO1200 and MachXO2280 devices also support BLVDS inputs on all four sides of the device.

Please refer to the MachXO Family Data Sheet for a detailed explanation of BLVDS implementation.

RSDS
All single-ended sysIO buffer pairs in all the MachXO devices support RSDS output driver using complementary LVCMOS drivers with external resistors.

The MachXO1200 and MachXO2280 devices also support RSDS inputs on all four sides of the device.

Please refer to the MachXO Family Data Sheet for a detailed explanation of RSDS implementation.

LVPECL
All single-ended sysIO buffers pairs in all the MachXO devices support LVPECL output driver using complementary LVCMOS drivers with external resistors.

The MachXO1200 and MachXO2280 devices also support LVPECL inputs on all four sides of the device.

Please refer to the MachXO Family Data Sheet for a detailed explanation of LVPECL implementation.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
—	—	Previous Lattice releases.
July 2007	01.4	Correction in VCCIO (1.2V/1.5V/1.8V/2.5V3.3V) text section. VCCIO of Bank2 is used to power the JTAG pin for MachXO640, rather than Bank3.
September 2010	01.5	Updated for Lattice Diamond design software support.
		Updated IBIS information.

Appendix A. HDL Attributes for Synplify® and Precision® RTL Synthesis

Using these HDL attributes, you can assign the sysIO attributes directly in your source. You will need to use the attribute definition and syntax for the synthesis vendor you are planning to you to use. Below are a list of all the sysIO attributes syntax and examples for Precision RTL Synthesis and Synplify. This section only lists the sysIO buffer attributes for these devices. Refer to the Precision RTL Synthesis and Synplify user manuals for a complete list of synthesis attributes. These manuals are available through the ispLEVER or Diamond software Help.

VHDL Synplify/Precision RTL Synthesis

This section lists syntax and examples for all the sysIO attributes in VHDL when using Precision RTL Synthesis and Synplify synthesis tools.

Syntax

Table 8-11. VHDL Attribute Syntax for Precision RTL Synthesis and Synplify

Attribute	Syntax
IO_TYPE	attribute IO_TYPE: string; attribute IO_TYPE of *Pinname*: signal is "*IO_TYPE Value*";
OPENDRAIN	attribute OPENDRAIN: string; attribute OPENDRAIN of *Pinname*: signal is "*OpenDrain Value*";
DRIVE	attribute DRIVE: string; attribute DRIVE of *Pinname*: signal is "*Drive Value*";
PULLMODE	attribute PULLMODE: string; attribute PULLMODE of *Pinname*: signal is "*Pullmode Value*";
PCICLAMP	attribute PCICLAMP: string; attribute PCICLAMP of *Pinname*: signal is "*PCIClamp Value*";
SLEWRATE	attribute PULLMODE: string; attribute PULLMODE of *Pinname*: signal is "*Slewrate Value*";
LOC	attribute LOC: string; attribute LOC of *Pinname*: signal is "*pin_locations*";

Examples

IO_TYPE

--Attribute Declaration******

ATTRIBUTE IO_TYPE: string;

--IO_TYPE assignment for I/O Pin******

ATTRIBUTE IO_TYPE OF portA: SIGNAL IS "PCI33";
ATTRIBUTE IO_TYPE OF portB: SIGNAL IS "LVCMOS33";
ATTRIBUTE IO_TYPE OF portC: SIGNAL IS "LVDS25";

OPENDRAIN

--Attribute Declaration******

ATTRIBUTE OPENDRAIN: string;

--OPENDRAIN assignment for I/O Pin******

ATTRIBUTE OPENDRAIN OF portB: SIGNAL IS "ON";

DRIVE

--Attribute Declaration******

ATTRIBUTE DRIVE: string;

--*DRIVE assignment for I/O Pin*****

ATTRIBUTE DRIVE OF portB: SIGNAL IS “16”;

PULLMODE

--Attribute Declaration******

ATTRIBUTE PULLMODE : string;

--PULLMODE assignment for I/O Pin******

ATTRIBUTE PULLMODE OF portA: SIGNAL IS “DOWN”;

ATTRIBUTE PULLMODE OF portB: SIGNAL IS “UP”;

PCICLAMP

--Attribute Declaration******

ATTRIBUTE PCICLAMP: string;

--PCICLAMP assignment for I/O Pin******

ATTRIBUTE PCICLAMP OF portA: SIGNAL IS “ON”;

SLEWRATE

--Attribute Declaration******

ATTRIBUTE SLEWRATE : string;

-- SLEWRATE assignment for I/O Pin******

ATTRIBUTE SLEWRATE OF portB: SIGNAL IS “FAST”;

LOC

--Attribute Declaration******

ATTRIBUTE LOC : string;

--* LOC assignment for I/O Pin*****

ATTRIBUTE LOC OF input_vector: SIGNAL IS “E3,B3,C3 “;

Verilog Synpilfy

This section lists syntax and examples for all the sysIO attributes in Verilog using the Synplify synthesis tool.

Syntax

Table 8-12. Verilog Synplify Attribute Syntax

Attribute	Syntax
IO_TYPE	*PinType PinName* /* synthesis IO_TYPE="*IO_Type Value*"*/;
OPENDRAIN	*PinType PinName* /* synthesis OPENDRAIN ="*OpenDrain Value*"*/;
DRIVE	*PinType PinName* /* synthesis DRIVE="*Drive Value*"*/;
PULLMODE	*PinType PinName* /* synthesis PULLMODE="*Pullmode Value*"*/;
PCICLAMP	*PinType PinName* /* synthesis PCICLAMP ="*PCIClamp Value*"*/;
SLEWRATE	*PinType PinName* /* synthesis SLEWRATE="*Slewrate Value*"*/;
LOC	*PinType PinName* /* synthesis LOC="*pin_locations* "*/;

Examples

//IO_TYPE, PULLMODE, SLEWRATE and DRIVE assignment

```
output portB /*synthesis IO_TYPE="LVCMOS33" PULLMODE ="UP" SLEWRATE ="FAST"
DRIVE ="14"*/;
output portC /*synthesis IO_TYPE="LVDS25" */;
```

//OPENDRAIN

```
output portA /*synthesis OPENDRAIN ="ON"*/;
```

//PCICLAMP

```
output portA /*synthesis IO_TYPE="PCI33" PCICLAMP ="PCICLAMP"*/;
```

//I/O pin location

```
input [3:0] DATA0 /* synthesis loc="E3,B1,F3"*/;
```

//Register pin location

```
reg data_in_ch1_buf_reg3 /* synthesis loc="R40C47" */;
```

//Vectored internal bus

```
reg [3:0] data_in_ch1_reg /*synthesis loc ="R40C47,R40C46,R40C45,R40C44" */;
```

Verilog Precision RTL Synthesis

This section lists syntax and examples for all the sysIO attributes in Verilog using the Precision RTL Synthesis synthesis tool.

Syntax

Table 8-13. Verilog Precision RTL Synthesis Attribute Syntax

Attribute	Syntax
IO_TYPE	// pragma attribute PinName IO_TYPE IO_TYPE Value
OPENDRAIN	// pragma attribute PinName OPENDRAIN OpenDrain Value
DRIVE	// pragma attribute PinName DRIVE Drive Value
PULLMODE	// pragma attribute PinName PULLMODE Pullmode Value
PCICLAMP	// pragma attribute PinName PCICLAMP PCIClamp Value
SLEWRATE	// pragma attribute PinName SLEWRATE Slewrate Value
LOC	// pragma attribute PinName LOC pin_location

Example

//**IO_TYPE ***

// pragma attribute portA IO_TYPE PCI33
// pragma attribute portB IO_TYPE LVCMOS33
// pragma attribute portC IO_TYPE SSTL25_II

//* Opendrain ***

// pragma attribute portB OPENDRAIN ON
// pragma attribute portD OPENDRAIN OFF

//* Drive ***

// pragma attribute portB DRIVE 20
// pragma attribute portD DRIVE 8

//* Pullmode***

// pragma attribute portB PULLMODE UP

//* PCIClamp***

// pragma attribute portB PCICLAMP ON

//* Slewrate ***

// pragma attribute portB SLEWRATE FAST
// pragma attribute portD SLEWRATE SLOW

//*LOC***

// pragma attribute portB loc E3

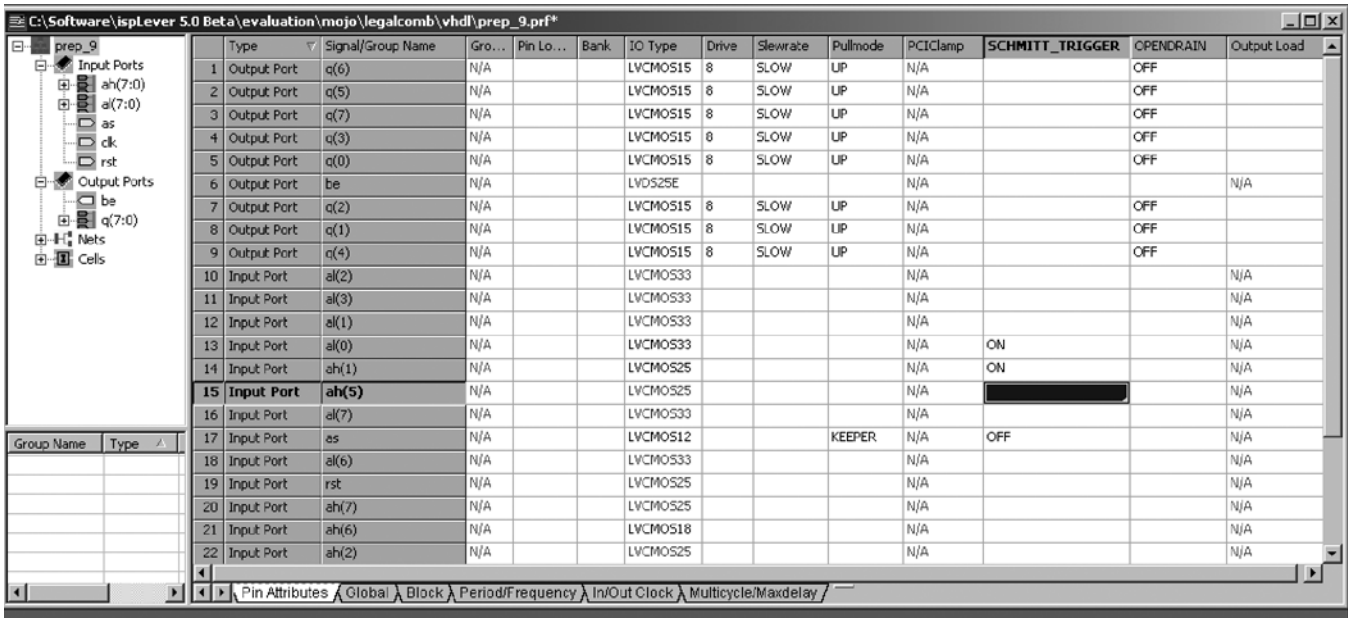
Appendix B. sysIO Attributes Using the ispLEVER Preference Editor or Diamond Spreadsheet View

You can assign the sysIO buffer attributes using the Pre-map Preference Editor GUI available in the ispLEVER or the Spreadsheet View GUI in Diamond. The Pin Attribute sheet in ispLEVER and Port Assignments sheet in Diamond list all the ports in your design and all the available sysIO attributes as preferences. When you click on each of these cells you get a list of all the valid I/O preferences for that port. Each column takes precedence over the next. Therefore, when you choose a particular IO_TYPE, the DRIVE, PULLMODE and SLEWRATE columns will only list the valid combinations for that IO_TYPE. The user can lock the pin locations using the pin location column of the Pin Attribute sheet. Right-click on a cell to list all the available pin locations. The Preference Editor and the Spreadsheet View will also do a DRC check to search for any incorrect pin assignments.

You can enter the DIN/ DOUT preferences using the Cell Attributes sheet of the Preference Editor. All the preferences assigned using the Preference Editor and the Spreadsheet View are written into the preference file (.prf).

Figure 8-5 shows the Pin Attribute sheet and the Cell Attribute sheet view of the Preference Editor. For further information on how to use the Preference Editor, please refer to the ispLEVER Help documentation. You can get to this in the Help menu option of the software.

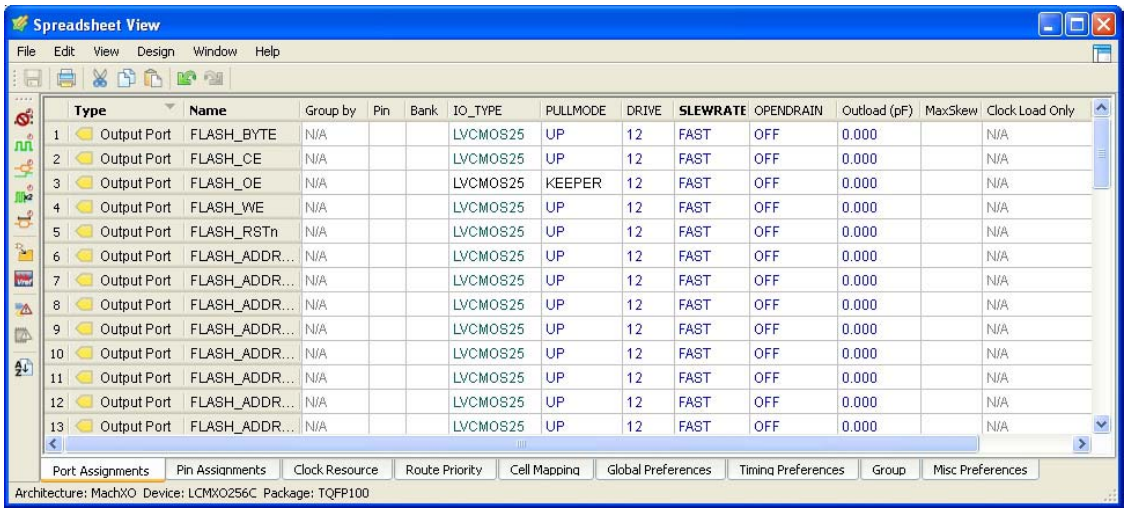
Figure 8-5. Pin Attributes Tab



	Type	Signal/Group Name	Gro...	Pin Lo...	Bank	IO Type	Drive	Slewrate	Pullmode	PCIClamp	SCHMITT_TRIGGER	OPENDRAIN	Output Load
1	Output Port	q(6)	N/A			LVCMOS15	8	SLOW	UP	N/A		OFF	
2	Output Port	q(5)	N/A			LVCMOS15	8	SLOW	UP	N/A		OFF	
3	Output Port	q(7)	N/A			LVCMOS15	8	SLOW	UP	N/A		OFF	
4	Output Port	q(3)	N/A			LVCMOS15	8	SLOW	UP	N/A		OFF	
5	Output Port	q(0)	N/A			LVCMOS15	8	SLOW	UP	N/A		OFF	
6	Output Port	be	N/A			LVDS25E				N/A			N/A
7	Output Port	q(2)	N/A			LVCMOS15	8	SLOW	UP	N/A		OFF	
8	Output Port	q(1)	N/A			LVCMOS15	8	SLOW	UP	N/A		OFF	
9	Output Port	q(4)	N/A			LVCMOS15	8	SLOW	UP	N/A		OFF	
10	Input Port	al(2)	N/A			LVCMOS33				N/A			N/A
11	Input Port	al(3)	N/A			LVCMOS33				N/A			N/A
12	Input Port	al(1)	N/A			LVCMOS33				N/A			N/A
13	Input Port	al(0)	N/A			LVCMOS33				N/A	ON		N/A
14	Input Port	ah(1)	N/A			LVCMOS25				N/A	ON		N/A
15	**Input Port**	**ah(5)**	N/A			LVCMOS25				N/A			N/A
16	Input Port	al(7)	N/A			LVCMOS33				N/A			N/A
17	Input Port	as	N/A			LVCMOS12			KEEPER	N/A	OFF		N/A
18	Input Port	al(6)	N/A			LVCMOS33				N/A			N/A
19	Input Port	rst	N/A			LVCMOS25				N/A			N/A
20	Input Port	ah(7)	N/A			LVCMOS25				N/A			N/A
21	Input Port	ah(6)	N/A			LVCMOS18				N/A			N/A
22	Input Port	ah(2)	N/A			LVCMOS25				N/A			N/A

Figure 8-6 shows the Port Assignment sheet in the Diamond Spreadsheet View. For further information on how to use the Spreadsheet View, please refer to the Reference Guide available in Diamond.

Figure 8-6. Port Assignment Tab



	Type	Name	Group by	Pin	Bank	IO_TYPE	PULLMODE	DRIVE	SLEWRATE	OPENDRAIN	Outload (pF)	MaxSkew	Clock Load Only
1	Output Port	FLASH_BYTE	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
2	Output Port	FLASH_CE	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
3	Output Port	FLASH_OE	N/A			LVCMOS25	KEEPER	12	FAST	OFF	0.000		N/A
4	Output Port	FLASH_WE	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
5	Output Port	FLASH_RSTn	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
6	Output Port	FLASH_ADDR...	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
7	Output Port	FLASH_ADDR...	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
8	Output Port	FLASH_ADDR...	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
9	Output Port	FLASH_ADDR...	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
10	Output Port	FLASH_ADDR...	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
11	Output Port	FLASH_ADDR...	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
12	Output Port	FLASH_ADDR...	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A
13	Output Port	FLASH_ADDR...	N/A			LVCMOS25	UP	12	FAST	OFF	0.000		N/A

Appendix C. sysIO Attributes Using Preference File (ASCII File)

You can also enter the sysIO Attributes directly in the preference (.prf) file as sysIO buffer preferences. The PRF file is an ASCII file containing two separate sections: a schematic section for those preferences created by the Mapper or Translator, and a user section for preferences entered by the user. You can write user preferences directly into this file. The synthesis attributes appear between schematic start and schematic end of the file. You can enter the sysIO buffer preferences after the schematic end line using the preference file syntax. Below are a list of sysIO buffer preference syntax and examples.

IOBUF

This preference is used to assign the attribute IO_TYPE, OPENDRAIN, DRIVE, PULLMODE, PCICLAMP and SLEWRATE.

Syntax
IOBUF [ALLPORTS | PORT <port_name> | GROUP <group_name>] (keyword=<value>)+;

where:

<port_name> = These are not the actual top-level port names, but should be the signal name attached to the port. PIOs in the physical design (.ncd) file are named using this convention. Any multiple listings or wildcarding should be done using GROUPs

Keyword = IO_TYPE, OPENDRAIN, DRIVE, PULLMODE, PCICLAMP, SLEWRATE

Example
IOBUF PORT “port1” IO_TYPE=LVTTL33 OPENDRAIN=ON DRIVE=8 PULLMODE=UP

PCICLAMP =OFF SLEWRATE=FAST;

DEFINE GROUP “bank1” “in*” “out_[0-31]”;

IOBUF GROUP “bank1” IO_TYPE=LVCMOS33;

LOCATE

When this preference is applied to a specified component, it places the component at a specified site and locks the component to the site. If applied to a specified macro instance it places the macro’s reference component at a specified site, places all of the macro’s pre-placed components (that is, all components that were placed in the macro’s library file) in sites relative to the reference component, and locks all of these placed components at their sites. This can also be applied to a specified PGROUP.

Syntax
LOCATE [COMP <comp_name> | MACRO <macro_name>] SITE <site_name>;

LOCATE PGROUP <pgroup_name> [SITE <site_name>; | REGION <region_name>;]

LOCATE PGROUP <pgroup_name> RANGE <site_1> [<site_2> | <count>] [<direction>] | RANGE <chip_side> [<direction>];

LOCATE BUS < bus_name> ROW|COL <number>;

<bus_name> := string

<number> := integer

Note: If the comp_name, macro_name, or site_name begins with anything other than an alpha character (for example, “11C7”), you must enclose the name in quotes. Wildcard expressions are allowed in <comp_name>.

Example
This command places the port Clk0 on the site A4:

LOCATE COMP “Clk0” SITE “A4”;

This command places the component PFU1 on the site named R1C7:

LOCATE COMP “PFU1” SITE “R1C7”;

This command places bus1 on ROW 3 and bus2 on COL4

LOCATE BUS “bus1” ROW 3;

LOCATE BUS “bus2” COL 4;

Memory Usage Guide for MachXO Devices

October 2010 Technical Note TN1092

Introduction

This technical note discusses memory usage for the Lattice MachXO device family. It is intended to be used by design engineers as a guide in integrating the EBR and PFU based memories for these device families in ispLEVER® and Lattice Diamond™ design software.

The architecture of these devices provides resources for memory intensive applications. The sysMEM™ Embedded Block RAM (EBR) complements its distributed PFU-based memory. Single-Port RAM, Dual-Port RAM, Pseudo Dual-Port RAM, FIFO and ROM memories can be constructed using the EBR. LUTs and PFU can implement Distributed Single-Port RAM, Dual-Port RAM and ROM.

The capabilities of the EBR Block RAM and PFU RAM are referred to as primitives and are described later in this document. Designers can utilize the memory primitives in two ways:

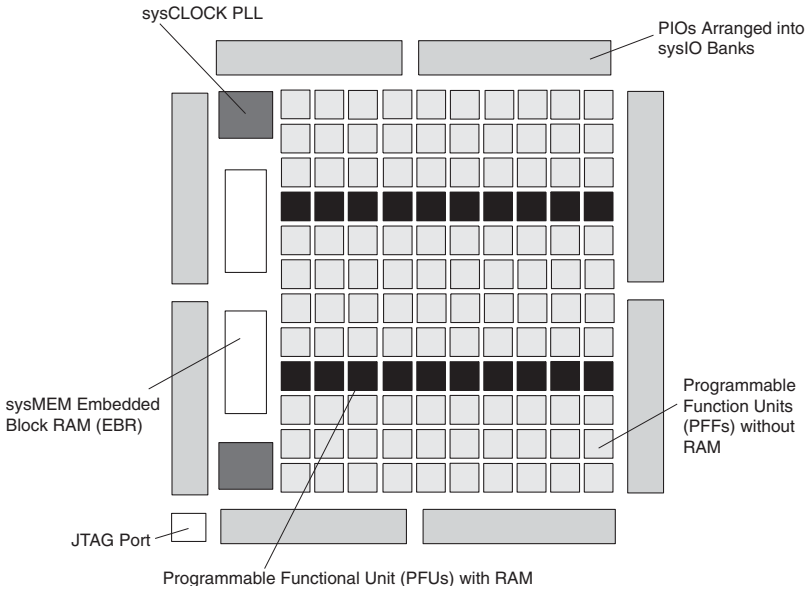
- Via **IPexpress™** – The IPexpress GUI allows users to specify the memory type and size that is required. IPexpress takes this specification and constructs a netlist to implement the desired memory by using one or more of the memory primitives.
- Via the **PMI (Parameterizable Module Inferencing)** – PMI allows experienced users to skip the graphical interface and utilize the configurable memory modules on the fly from the ispLEVER Project Navigator or Diamond. The parameters and the control signals needed either in Verilog or VHDL can be set. The top-level design will have the parameters defined and signals declared so the interface can automatically generate the black box during synthesis.

The remainder of this document discusses these approaches, utilizing IPexpress, PMI inference, memory modules and memory primitives.

MachXO Device Memories

Only the MachXO1200 and MachXO2280 devices contain the sysMEM EBR blocks along with an array of logic blocks called PFUs (or PFFs) surrounded by Programmable I/O Cells (PICs). This is shown in Figure 9-1.

Figure 9-1. Logical View of MachXO1200 and MachXO2280 Devices



The PFU contains the building blocks for logic and Distributed RAM and ROM. The PFF provides the logic building blocks without the distributed RAM This document describes the memory usage and implementation for both Embedded Memory Blocks (EBRs) and Distributed RAM of the PFU. Refer to the MachXO Family Data Sheet for details on the hardware implementation of the EBR and Distributed RAM.

The logic blocks are arranged in a two-dimensional grid with rows and columns as shown in the figures below. The physical location of the EBR and Distributed RAM follows the row and column designation. Since the Distributed RAM is part of the PFU resource, it follows the PFU/PFF row and column designation. The EBR occupies two columns per block to account for the wider port interface.

Utilizing IPexpress

Designers can utilize IPexpress to easily specify a variety of memories in their designs. These modules will be constructed using one or more memory primitives along with general purpose routing and LUTs as required. The available primitives are:

- Single Port RAM (RAM_DQ) – EBR-based
- Dual PORT RAM (RAM_DP_TRUE) – EBR-based
- Pseudo Dual Port RAM (RAM_DP) – EBR-based
- Read Only Memory (ROM) – EBR-Based
- First In First Out Memory (Dual Clock) (FIFO_DC) – EBR-based
- Distributed Single Port RAM (Distributed_SPRAM) – PFU-based
- Distributed Dual Port RAM (Distributed_DPRAM) – PFU-based
- Distributed ROM (Distributed_ROM) – PFU/PFF-based
- RAM Based Shift Register (RAM_Based_Shift_Register) – PFU-based
- Distributed Shift Register (RAM_Based_Shift_Register) - PFU based (see IPexpress Help for details)

IPexpress Flow

For generating any of these memories, create (or open) a project for the MachXO devices.

From Diamond or the ispLEVER Project Navigator, select **Tools > IPexpress**. Alternatively, users can also click on the button in the toolbar shown below when the MachXO devices are targeted in the project.

This opens the IPexpress window as shown in Figure 9-2.

Figure 9-2. IPexpress Main Window, ispLEVER

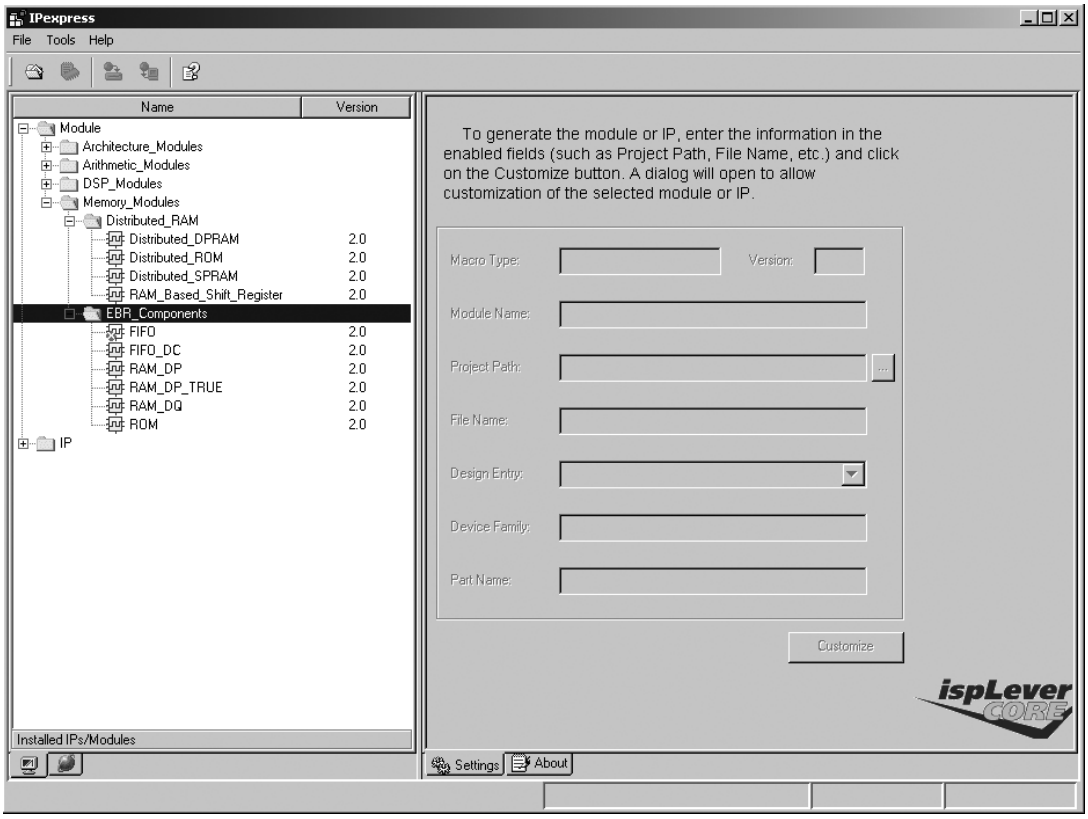
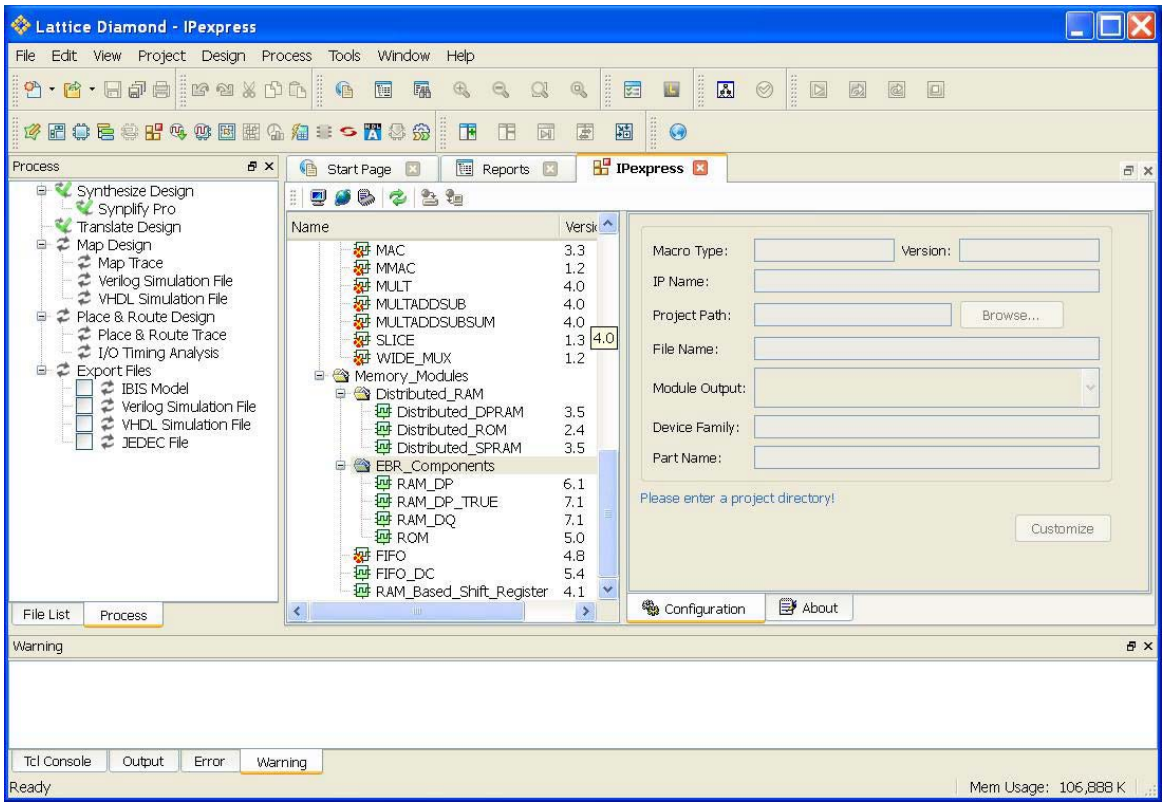


Figure 9-3. IPexpress Main Window, Diamond



The left pane of this window includes the Module Tree. The EBR-based Memory Modules are under the **EBR_Components** and the PFU-based Distributed Memory Modules are under **Storage_Components** as shown in Figure 9-2 and Figure 9-3.

As an example, let us consider generating an EBR-based Pseudo Dual Port RAM of size 512x16. Select RAM_DP under the EBR_Components. The right pane changes as shown in Figure 9-4 and Figure 9-5.

Figure 9-4. Generating Pseudo Dual Port RAM (RAM_DP) Using IPexpress in ispLEVER

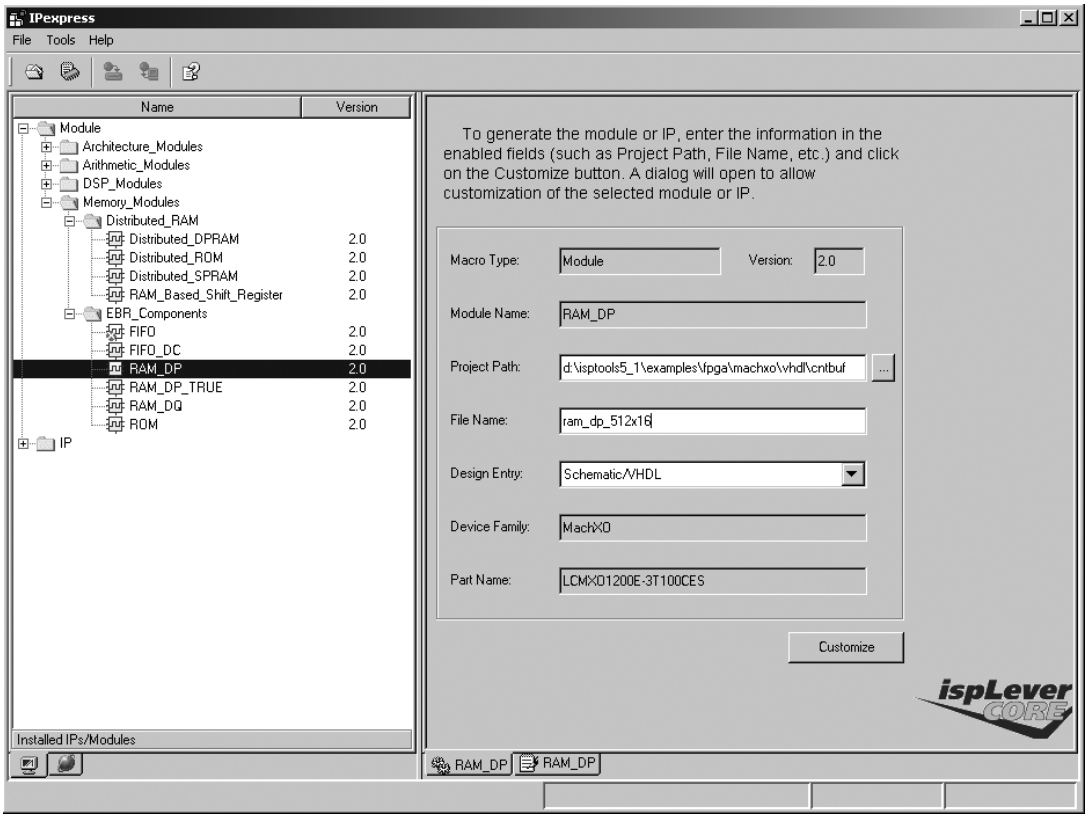
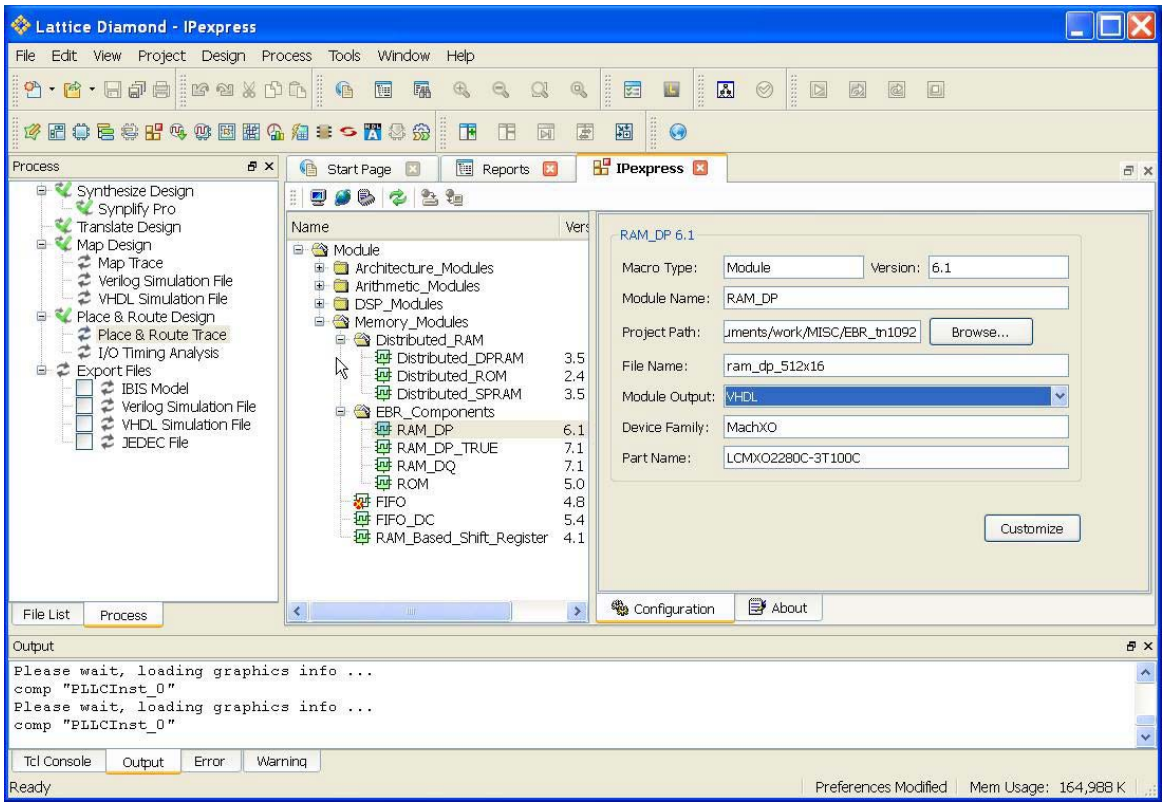


Figure 9-5. Generating Pseudo Dual Port RAM (RAM_DP) Using IPexpress in Diamond



In this right pane, options like the **Device Family**, **Macro Type**, **Category**, and **Module_Name** are device and selected module dependent. These cannot be changed in IPexpress.

Users can change the directory where the generated module files will be placed by clicking the **Browse** button in the **Project Path**.

The **Module Name** text box allows users to specify an entity name for the module they are about to generate. Users must provide this entity name.

Design entry, Verilog or VHDL, by default, is the same as the project type. If the project is a VHDL project, the selected design entry option will be “Schematic/ VHDL”, and “Schematic/ Verilog-HDL” if the project type is Verilog-HDL.

The **Device** pull-down menu allows users to select different devices within the same family, MachXO in this example.

Then click the **Customize** button.

This opens another window where users can customize the RAM (Figure 9-6 and Figure 9-7).

Figure 9-6. Generating Pseudo Dual Port RAM (RAM_DP) Module Customization in ispLEVER

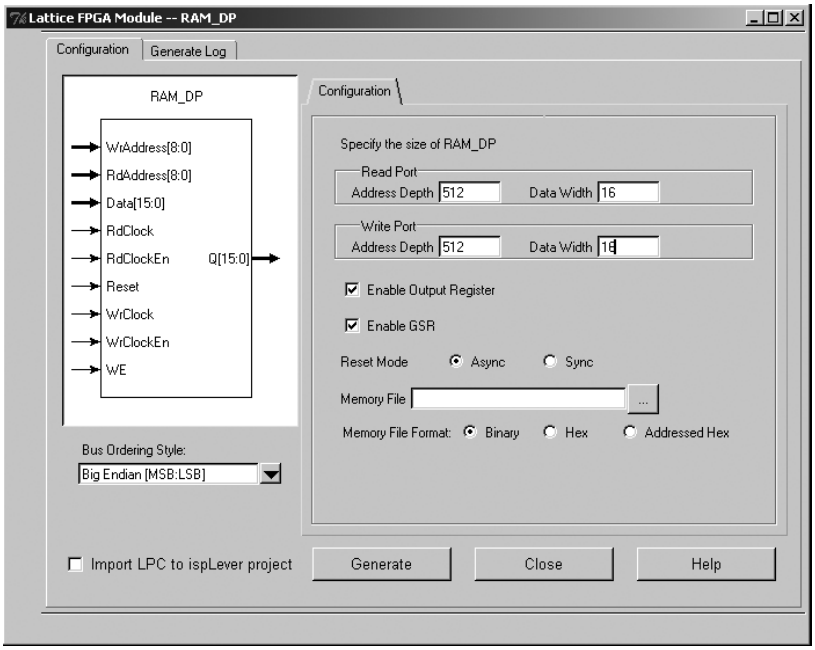
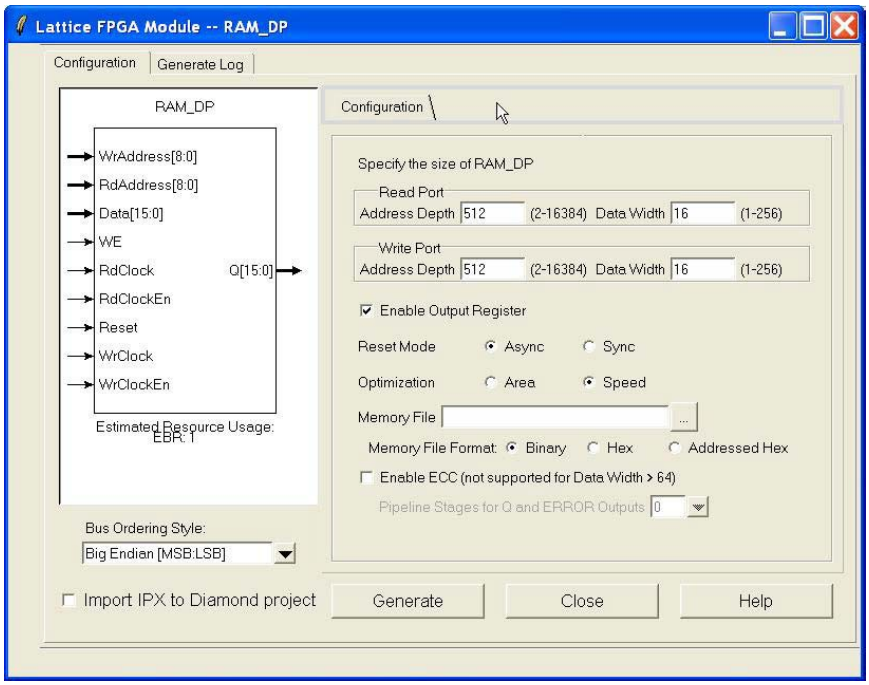


Figure 9-7. Generating Pseudo Dual Port RAM (RAM_DP) Module Customization in Diamond



The left side of this window shows the block diagram of the module. The right side includes the Configuration tab where users can choose options to customize the RAM_DP such as (e.g. specify the address port sizes and data widths).

Users can specify the address depth and data width for the **Read Port** and the **Write Port** in the text boxes provided. In this example we are generating a Pseudo Dual Port RAM of size 512 x 16. Users can also create the RAMs of different port widths in case of Pseudo Dual Port and True Dual Port RAMs.

The Input Data and the Address Control is always registered, as the hardware only supports the clocked write operation for the EBR based RAMs. The check box **Enable Output Registers**, inserts the output registers in the Read Data Port, as the output registers are optional for the EBR-based RAMs.

Users have the option to set the **Reset Mode** as Asynchronous Reset or Synchronous Reset. **Enable GSR** can be checked to be enable the Global Set Reset.

Users can also pre-initialize their memory with the contents specified in the **Memory File**. It is optional to provide this file in the RAM; however for ROM, the Memory File is required. These files can be of Binary, Hex or Addresses Hex format. The details of these formats are discussed in the Initialization File section of this document.

At this point, users can click the **Generate** button to generate the module they have customized. A VHDL or Verilog netlist is then generated and placed in the specified location. Users can incorporate this netlist in their designs.

Another important button is the **Load Parameters** button. IPexpress stores the parameters the user has specified in an <module_name>.lpc file. This file is generated along with the module. Users can click on the Load Parameter button to load the parameters of a previously generated module to re-visit or make changes to them.

Once the module is generated, user can either instantiate the *.lpc or the Verilog-HDL/ VHDL file in top-level module of their design.

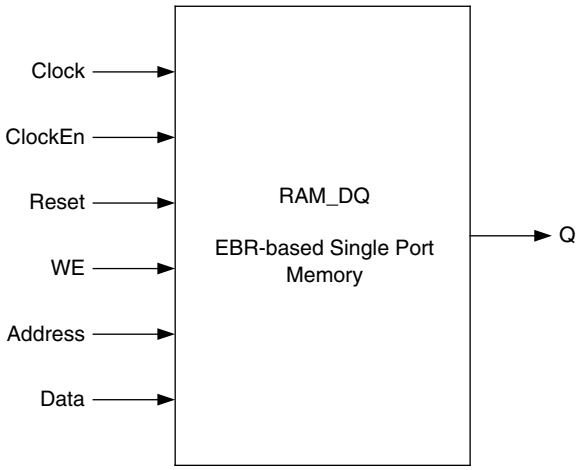
The various memory modules, both EBR and Distributed, are discussed in detail in this document.

Memory Modules

Single Port RAM (RAM_DQ) – EBR Based: The EBR blocks in the MachXO devices can be configured as Single Port RAM or RAM_DQ. IPexpress allows users to generate the Verilog-HDL or VHDL along EDIF netlist for the memory size, as per design requirements.

IPexpress generates the memory module as shown in Figure 9-8.

Figure 9-8. Single Port Memory Module Generated by IPexpress



Since the device has a number of EBR blocks, the generated module makes use of these EBR blocks, or primitives, and cascades them to create the memory sizes specified by the user in the IPexpress GUI. For memory sizes smaller than an EBR block, the module will be created in one EBR block. For memory sizes larger than one EBR block, multiple EBR blocks can be cascaded in depth or width (as required to create these sizes).

In Single Port RAM mode, the input data and address for the ports are registered at the input of the memory array. The output data of the memory is optionally registered at the output.

The various ports and their definitions for the Single Port Memory are listed in Table 9-1. The table lists the corresponding ports for the module generated by IPexpress and for the EBR RAM_DQ primitive.

Table 9-1. EBR-based Single Port Memory Port Definitions

Port Name in the Generated Module	Port Name in the EBR Block Primitive	Description	Active State
Clock	CLK	Clock	Rising Clock Edge
ClockEn	CE	Clock Enable	Active High
Address	AD[x:0]	Address Bus	—
Data	DI[y:0]	Data In	—
Q	DO[y:0]	Data Out	—
WE	WE	Write Enable	Active High
Reset	RST	Reset	Active High
—	CS[2:0]	Chip Select	—

Reset (or RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

Chip Select (CS) is a useful port in the EBR primitive when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. CS is a 3-bit bus, so it can cascade eight memories easily. If the memory size specified by the user requires more than eight EBR blocks, the ispLEVER or Diamond software automatically generates the additional address decoding logic which is implemented in the PFU (external to the EBR blocks).

Each EBR block consists of 9,216 bits of RAM. The values for x (address) and y (data) for each EBR block for the devices are listed in Table 9-2.

Table 9-2. Single Port Memory Sizes for 9K Memories in MachXO

Single Port Memory Size	Input Data	Output Data	Address [MSB:LSB]
8K x 1	DI	DO	AD[12:0]
4K x 2	DI[1:0]	DO[1:0]	AD[11:0]
2K x 4	DI[3:0]	DO[3:0]	AD[10:0]
1K x 9	DI[8:0]	DO[8:0]	AD[9:0]
512 x 18	DI[17:0]	DO[17:0]	AD[8:0]
256 x 36	DI[35:0]	DO[35:0]	AD[7:0]

Table 9-3 shows the various attributes available for the Single Port Memory (RAM_DQ). Some of these attributes are user selectable through the IPexpress GUI. For detailed attribute definitions, refer to Appendix A.

Table 9-3. Single Port RAM Attributes for MachXO

Attribute	Description	Values	Default Value	User Selectable through IPexpress
DATA_WIDTH	Data Word Width	1, 2, 4, 9, 18, 36	18	Yes
REGMODE	Register Mode (Pipelining)	NOREG, OUTREG	NOREG	Yes
RESETMODE	Selects Register Type	ASYNC, SYNC	ASYNC	Yes
CSDECODE	Chip Select Decode	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111 0b000	0b000	No
WRITEMODE	Read/Write Mode	NORMAL, WRITETHROUGH, READBEFOREWRITE	NORMAL	Yes
ENABLE GSR	Enable or Disable Global Set Reset	Enable (1), Disable (0)	1	Yes
INITVAL	Initialization Value	0x00000000000000000000000000000000000 00000000000000000000000000000000000 0000000....0xFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFF FFFFFFF (80-bit hex string)	0x0000000 000000000 000000000 000000000 000000000 000000000 000000000 000000000 000 0000000	—

The Single Port RAM (RAM_DQ) can be configured as NORMAL, READ BEFORE WRITE or WRITE THROUGH modes. Each of these modes affects what data comes out of the port Q of the memory during the write operation followed by the read operation at the same memory location.

Additionally, users can select to enable the output registers for RAM_DQ. Figures 9-9 to 9-14 show the internal timing waveforms for the Single Port RAM (RAM_DQ) with these options.

Figure 9-9. Single Port RAM Timing Waveform - NORMAL Mode, without Output Registers

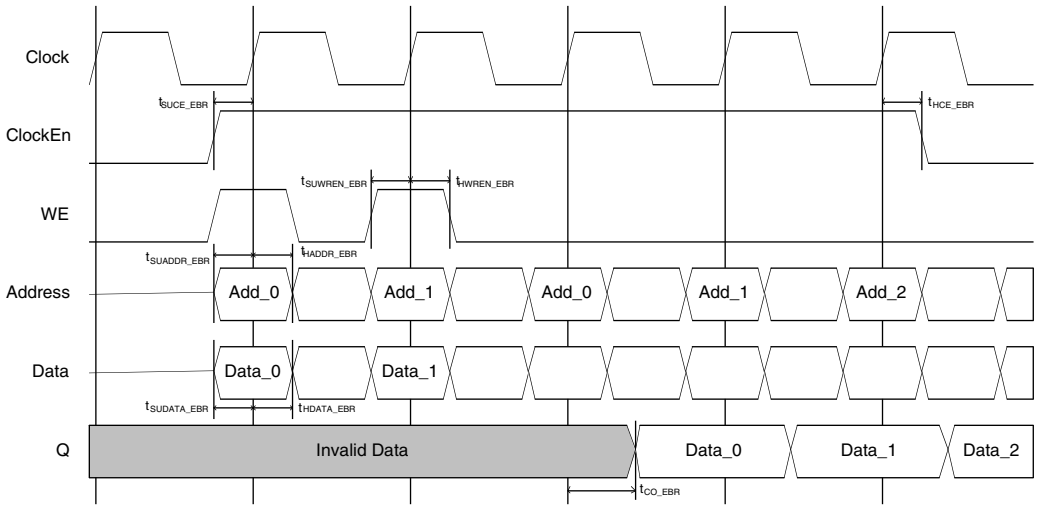


Figure 9-10. Single Port RAM Timing Waveform - NORMAL Mode, with Output Registers

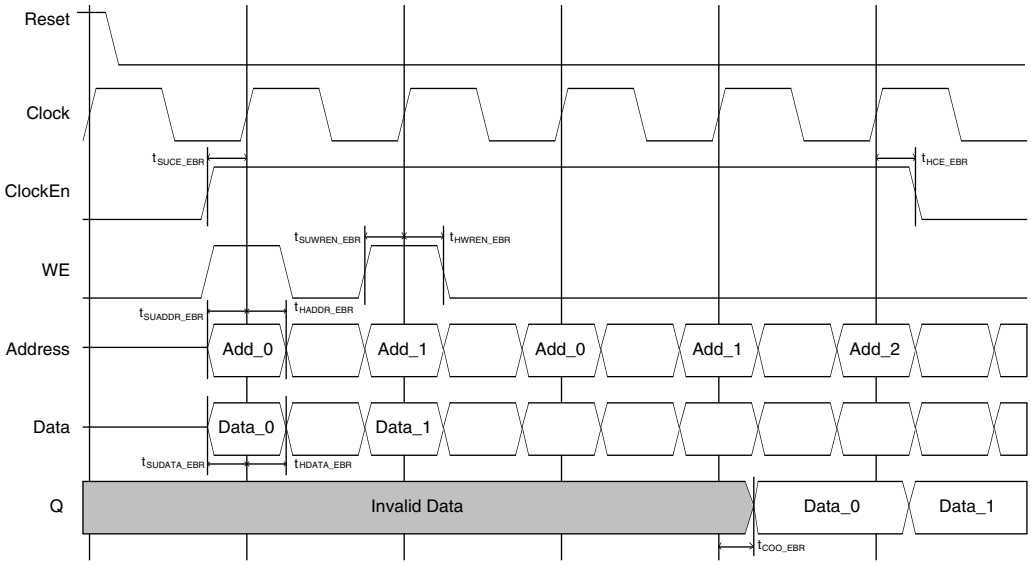


Figure 9-11. Single Port RAM Timing Waveform - READ BEFORE WRITE Mode, without Output Registers

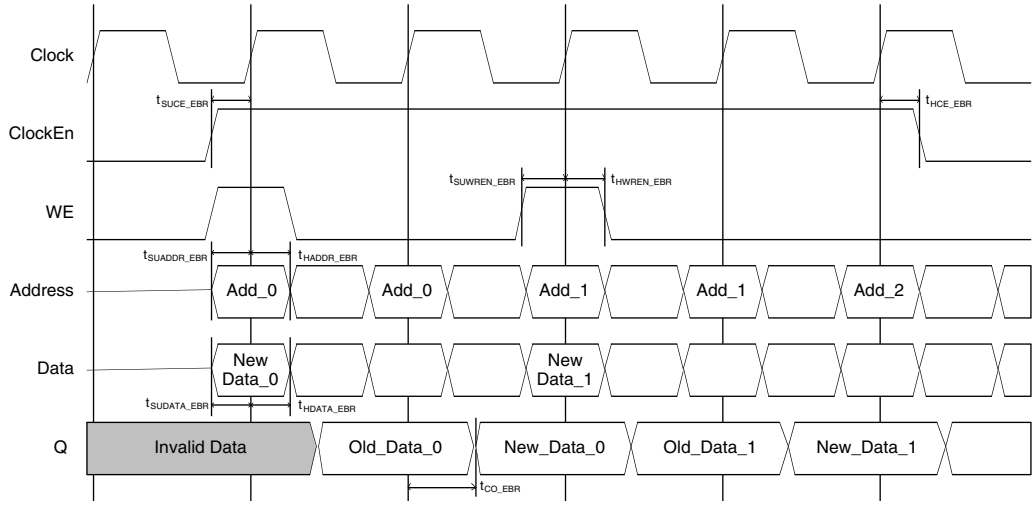


Figure 9-12. Single Port RAM Timing Waveform - READ BEFORE WRITE Mode, with Output Registers

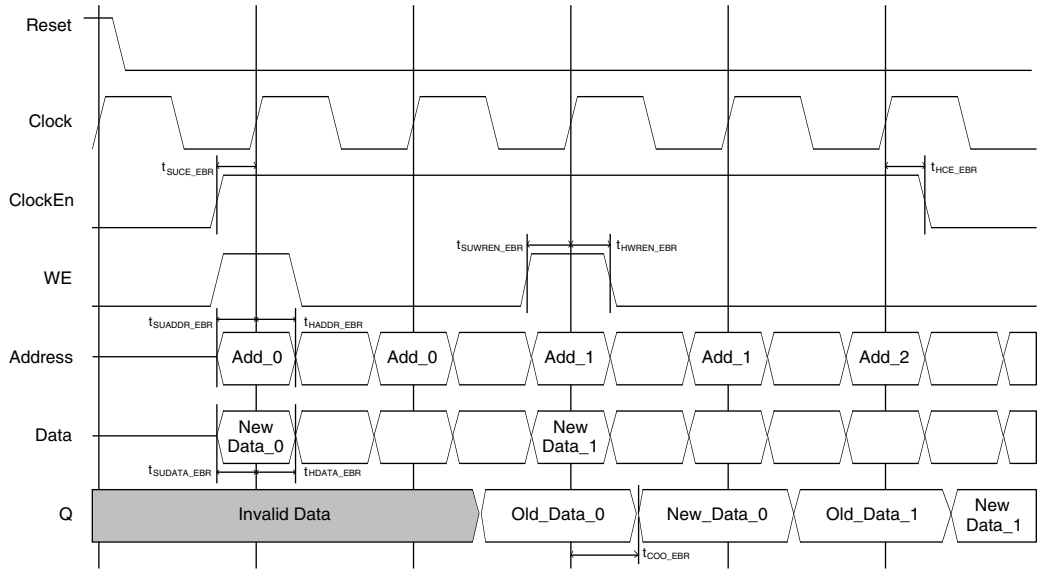


Figure 9-13. Single Port RAM Timing Waveform - WRITE THROUGH Mode, without Output Registers

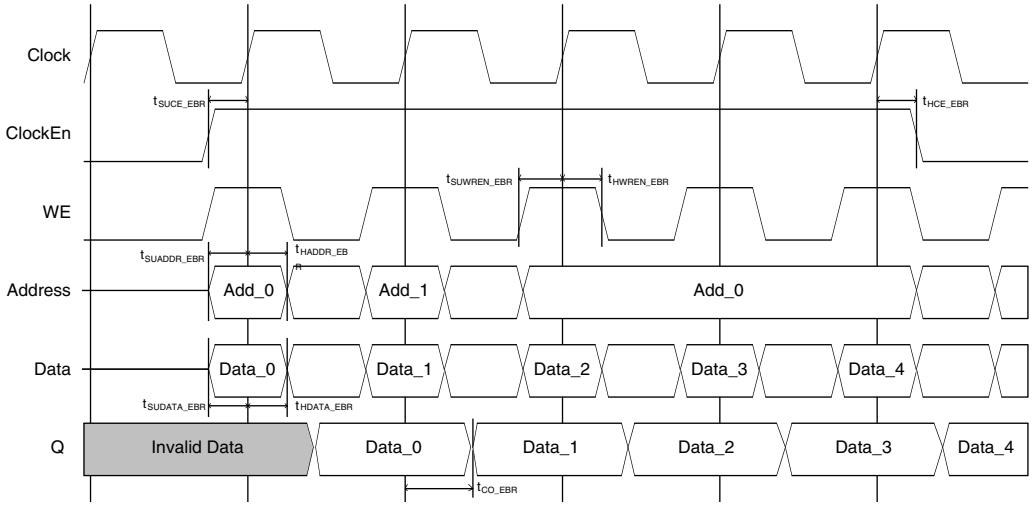
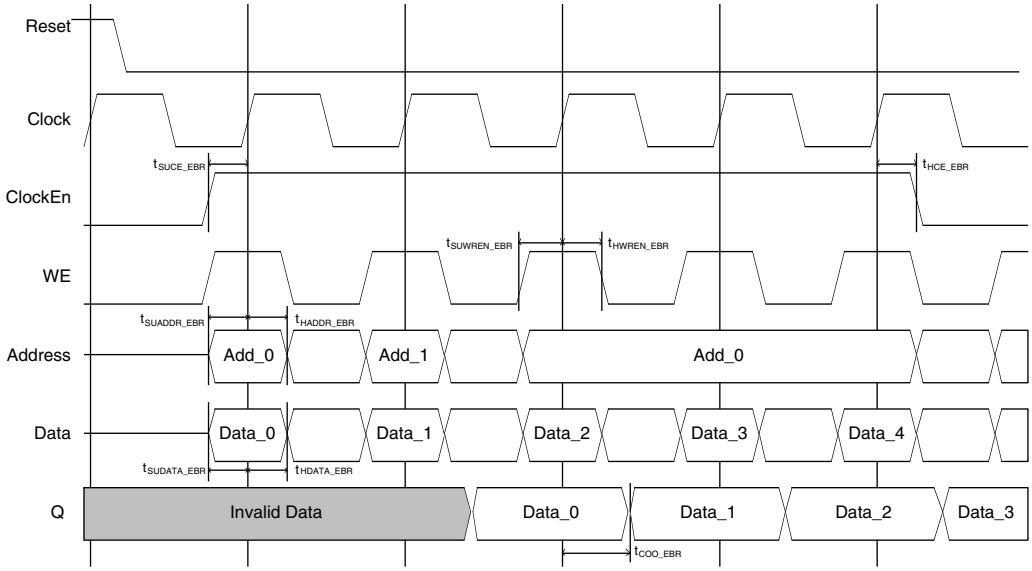


Figure 9-14. Single Port RAM Timing Waveform - WRITE THROUGH Mode, with Output Registers

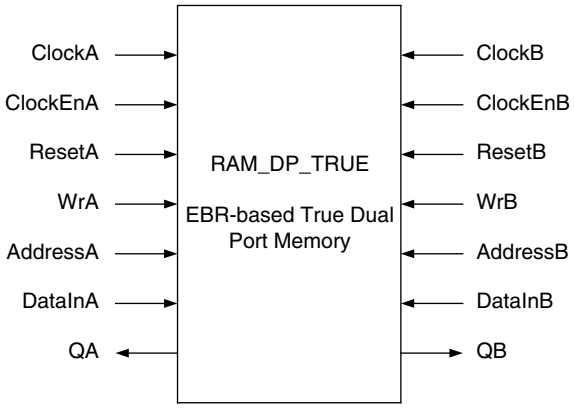


True Dual Port RAM (RAM_DP_TRUE) – EBR Based

The EBR blocks in MachXO devices can be configured as True-Dual Port RAM or RAM_DP_TRUE. IPexpress allows users to generate the Verilog-HDL, VHDL or EDIF netlists for various memory sizes depending on design requirements.

IPexpress generates the memory module as shown in Figure 9-15.

Figure 9-15. True Dual Port Memory Module generated by IPexpress



The generated module makes use of these EBR blocks or primitives. For memory sizes smaller than an EBR block, the module will be created in one EBR block. When the specified memory is larger than one EBR block, multiple EBR blocks can be cascaded in depth or width (as required to create these sizes).

In True Dual Port RAM mode, the input data and address for the ports are registered at the input of the memory array. The output data of the memory is optionally registered at the output.

The various ports and their definitions for Single Port Memory are in Table 9-4. The table lists the corresponding ports for the module generated by IPexpress and for the EBR RAM_DP_TRUE primitive.

Table 9-4. EBR based True Dual Port Memory Port Definitions

Port Name in Generated Module	Port Name in the EBR Block Primitive	Description	Active State
ClockA, ClockB	CLKA, CLKB	Clock for PortA and PortB	Rising Clock Edge
ClockEnA, ClockEnB	CEA, CEB	Clock Enables for Port CLKA and CLKB	Active High
AddressA, AddressB	ADA[x1:0], ADB[x2:0]	Address Bus Port A and Port B	—
DataA, DataB	DIA[y1:0], DIB[y2:0]	Input Data Port A and Port B	—
QA, QB	DOA[y1:0], DOB[y2:0]	Output Data Port A and Port B	—
WrA, WrB	WEA, WEB	Write enable Port A and Port B	Active High
ResetA, ResetB	RSTA, RSTB	Reset for PortA and PortB	Active High
	CSA[2:0], CSB[2:0]	Chip Selects for each port	—

Reset (or RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

Chip Select (CS) is a useful port in the EBR primitive when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. Since CS is a 3-bit bus, it can cascade eight memories easily. However, if the memory size specified by the user requires more than eight EBR blocks, the ispLEVER or Diamond software automatically generates the additional address decoding logic, which is implemented in the PFU external to the EBR blocks.

Each EBR block consists of 9,216 bits of RAM. The values for x's (for address) and y's (data) for each EBR block for the devices are listed in Table 9-5.

Table 9-5. MachXO Dual Port Memory Sizes for 9K Memory

Dual Port Memory Size	Input Data Port A	Input Data Port B	Output Data Port A	Output Data Port B	Address Port A [MSB:LSB]	Address Port B [MSB:LSB]
8K x 1	DIA	DIB	DOA	DOB	ADA[12:0]	ADB[12:0]
4K x 2	DIA[1:0]	DIB[1:0]	DOA[1:0]	DOB[1:0]	ADA[11:0]	ADB[11:0]
2K x 4	DIA[3:0]	DIB[3:0]	DOA[3:0]	DOB[3:0]	ADA[10:0]	ADB[10:0]
1K x 9	DIA[8:0]	DIB[8:0]	DOA[8:0]	DOB[8:0]	ADA[9:0]	ADB[9:0]
512 x 18	DIA[17:0]	DIB[17:0]	DOA[17:0]	DOB[17:0]	ADA[8:0]	ADB[8:0]

Table 9-6 shows the various attributes available for the True Dual Port Memory (RAM_DP_TRUE). Some of these attributes are user-selectable through the IPexpress GUI. For detailed attribute definitions, refer to the Appendix A.

Table 9-6. MachXO Dual Port RAM Attributes

Attribute	Description	Values	Default Value	User Selectable through IPexpress
DATA_WIDTH_A	Data Word Width Port A	1, 2, 4, 9, 18	18	YES
DATA_WIDTH_B	Data Word Width Port B	1, 2, 4, 9, 18	18	YES
REGMODE_A	Register Mode (Pipelining) for Port A	NOREG, OUTREG	NOREG	YES
REGMODE_B	Register Mode (Pipelining) for Port B	NOREG, OUTREG	NOREG	YES
RESETMODE	Selects the Reset type	ASYNC, SYNC	ASYNC	YES
CSDECODE_A	Chip Select Decode for Port A	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	000	NO
CSDECODE_B	Chip Select Decode for Port B	000, 001, 010, 011, 100, 101, 110, 111	000	NO
WRITEMODE_A	Read / Write Mode for Port A	NORMAL, WRITETHROUGH, READBEFOREWRITE	NORMAL	YES
WRITEMODE_B	Read / Write Mode for Port B	NORMAL, WRITETHROUGH, READBEFOREWRITE	NORMAL	YES
ENABLE GSR	Enables Global Set Reset	ENABLE, DISABLE	ENABLED	YES

The True Dual Port RAM (RAM_DP_TRUE) can be configured as NORMAL, READ BEFORE WRITE or WRITE THROUGH modes. Each of these modes affects what data comes out of port Q of the memory during the write operation followed by the read operation at the same memory location. Detailed discussions of the WRITE modes and the constraints of the True Dual Port can be found in Appendix A.

Additionally users can select to enable the output registers for RAM_DP_TRUE. Figures 9-16 to 9-21 show the internal timing waveforms for the True Dual Port RAM (RAM_DP_TRUE) with these options.

Figure 9-16. True Dual Port RAM Timing Waveform - NORMAL Mode, without Output Registers

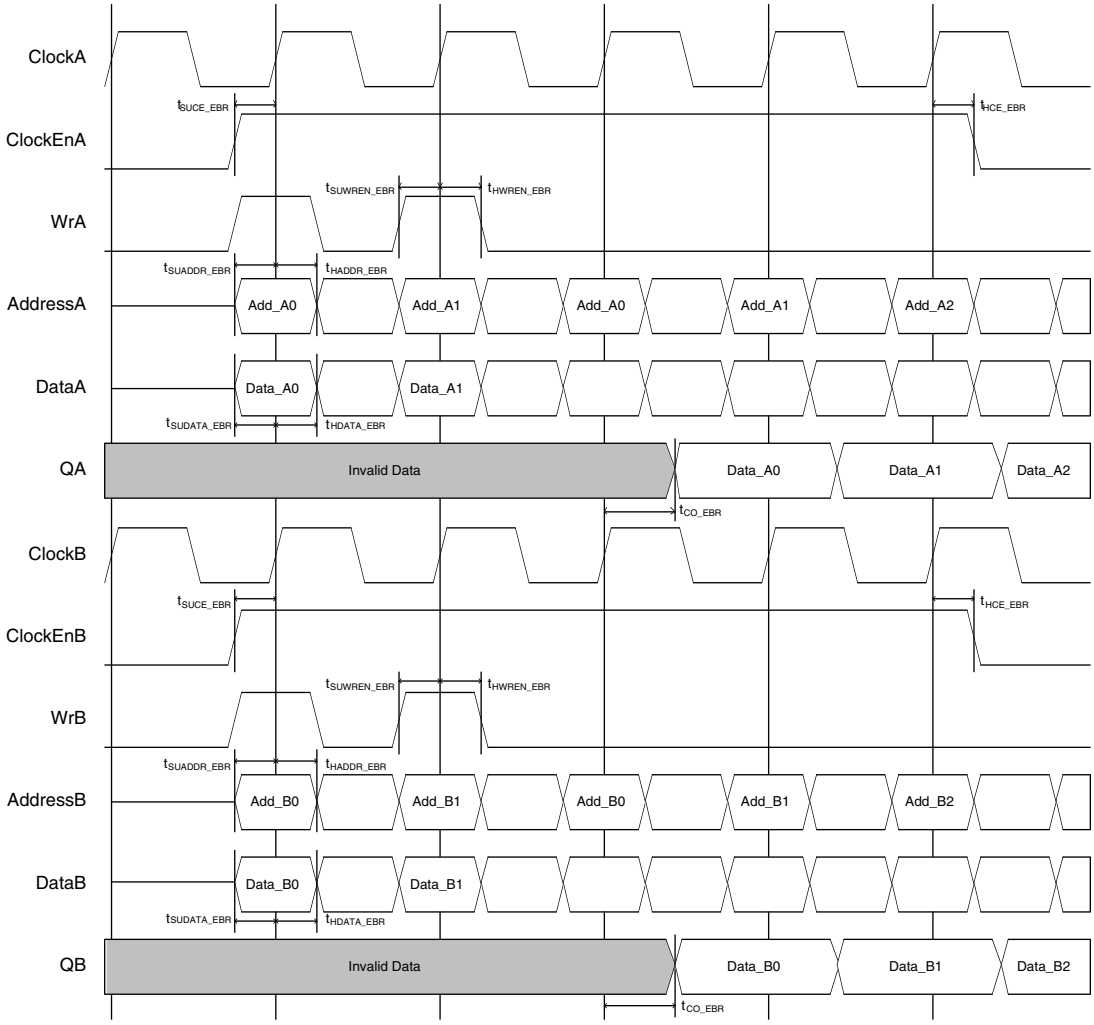


Figure 9-17. True Dual Port RAM Timing Waveform - NORMAL Mode with Output Register

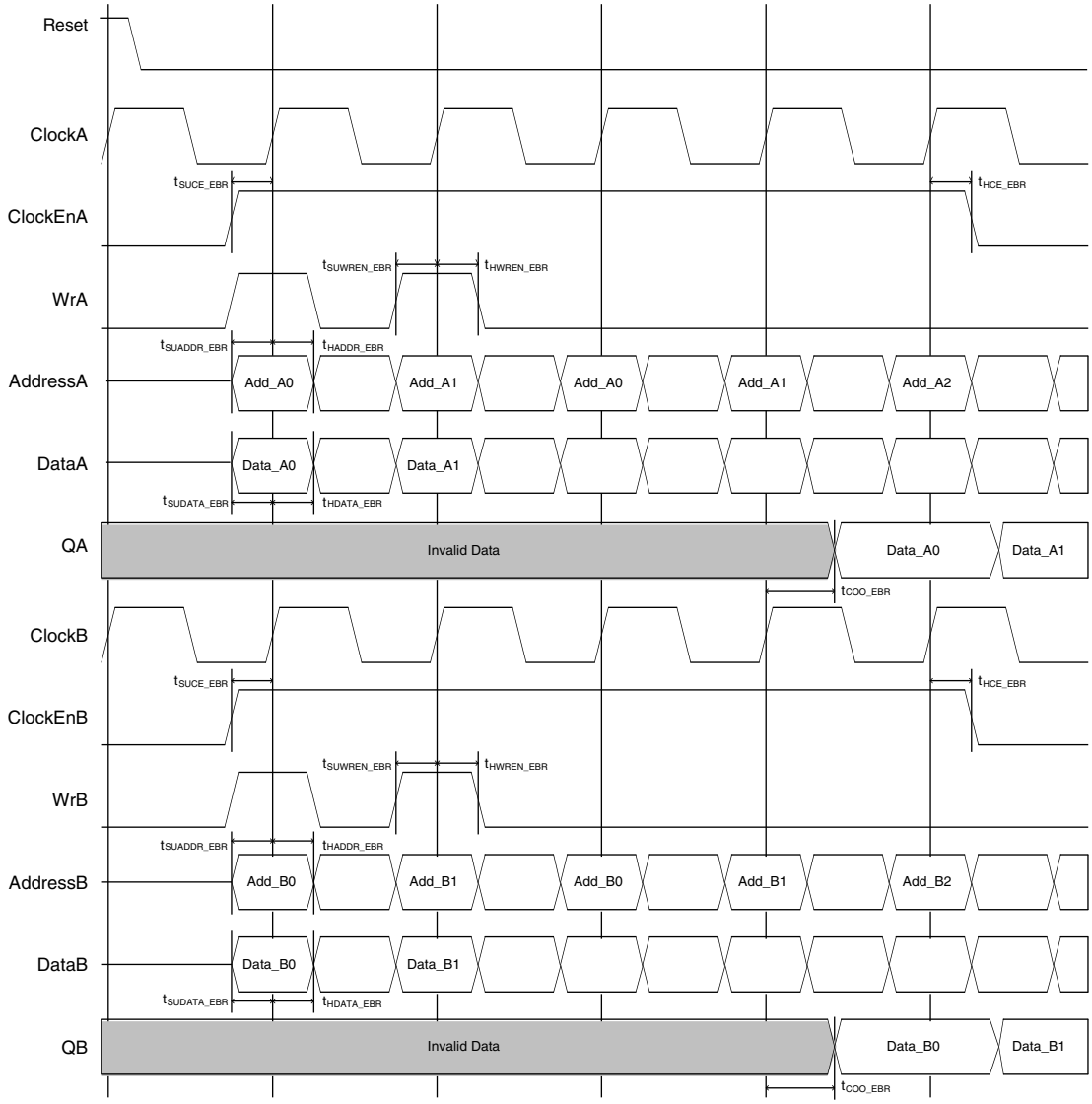


Figure 9-18. True Dual Port RAM Timing Waveform - READ BEFORE WRITE Mode, without Output Registers

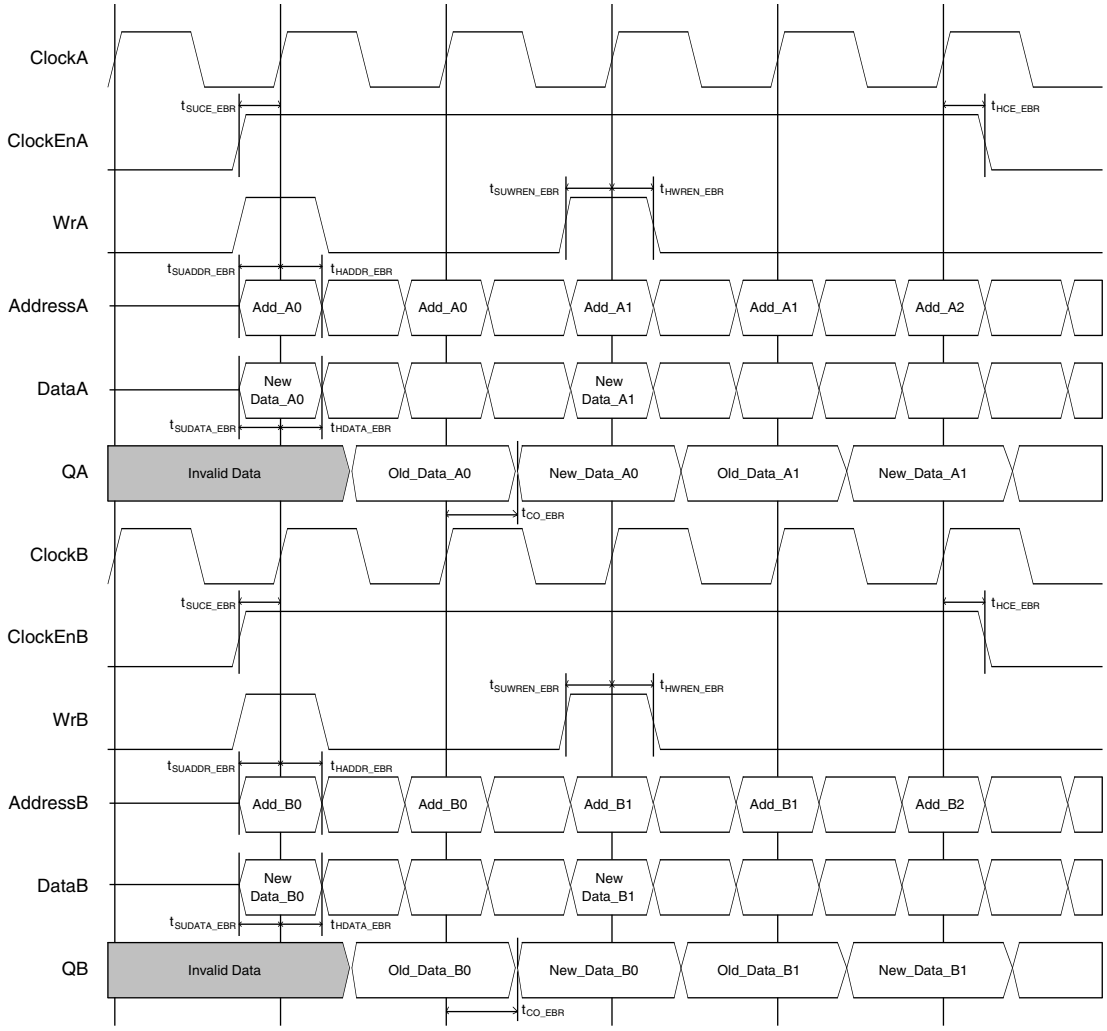


Figure 9-19. True Dual Port RAM Timing Waveform - READ BEFORE WRITE Mode, with Output Registers

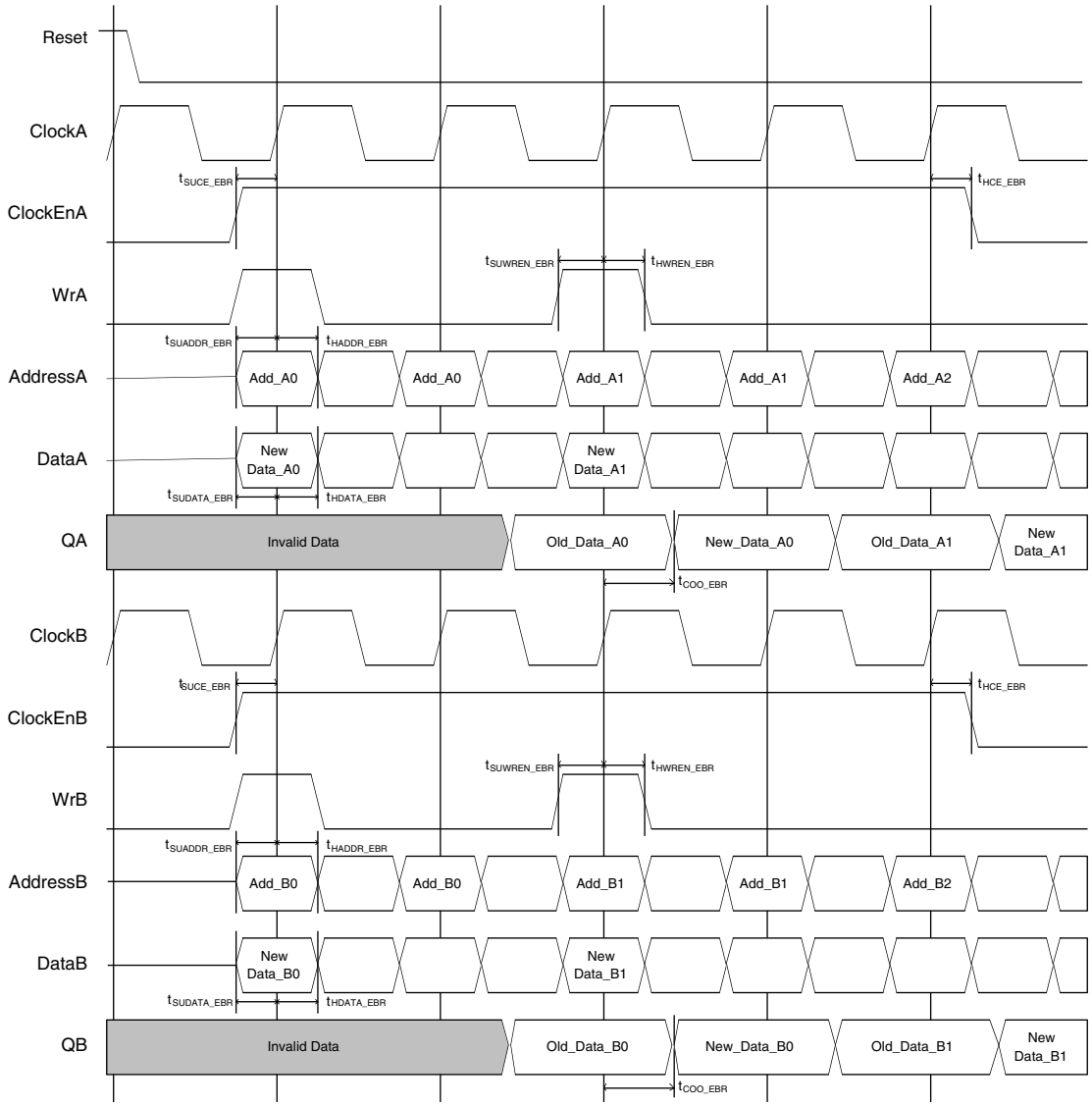


Figure 9-20. True Dual Port RAM Timing Waveform - WRITE THROUGH Mode, without Output Registers

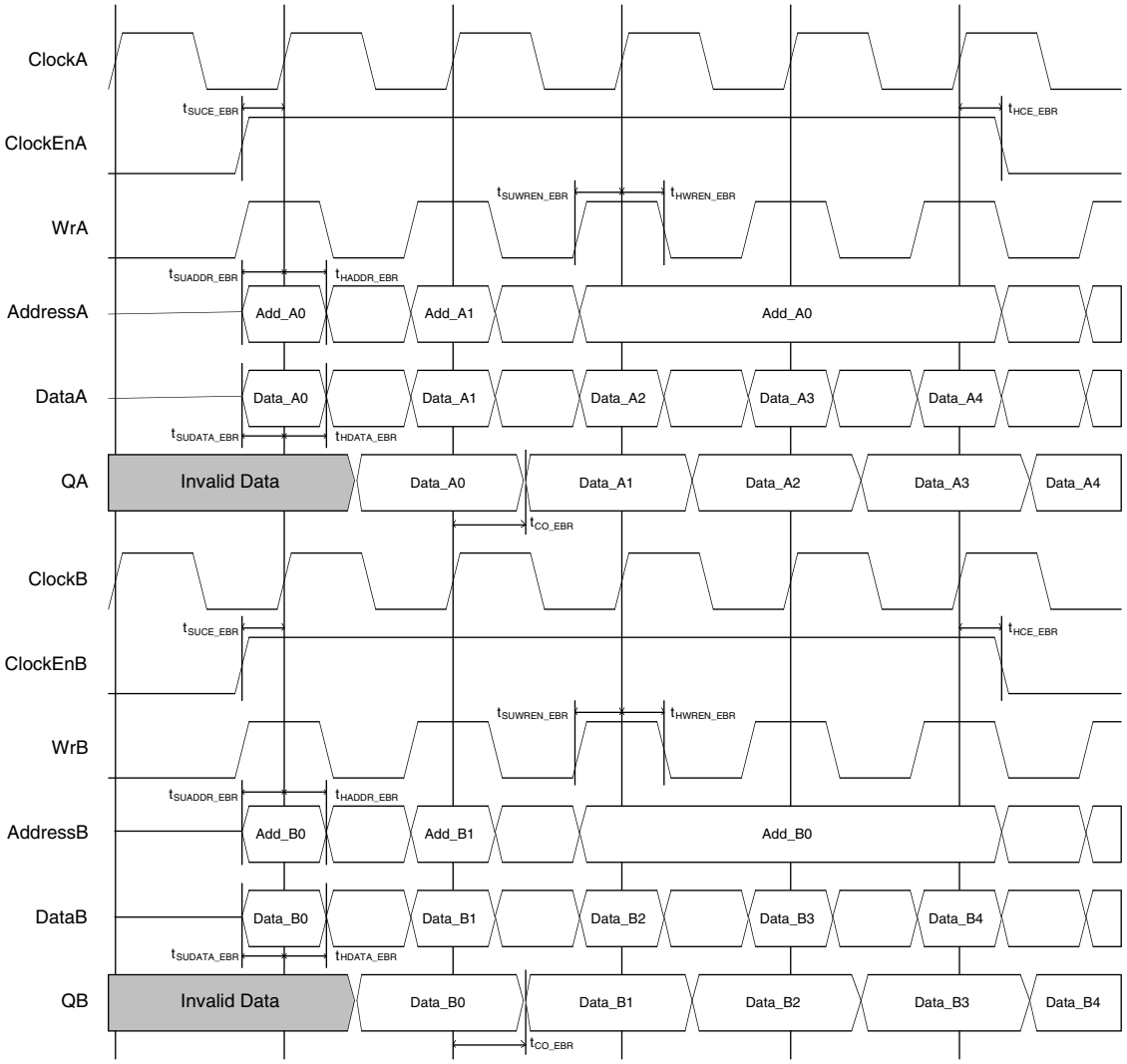
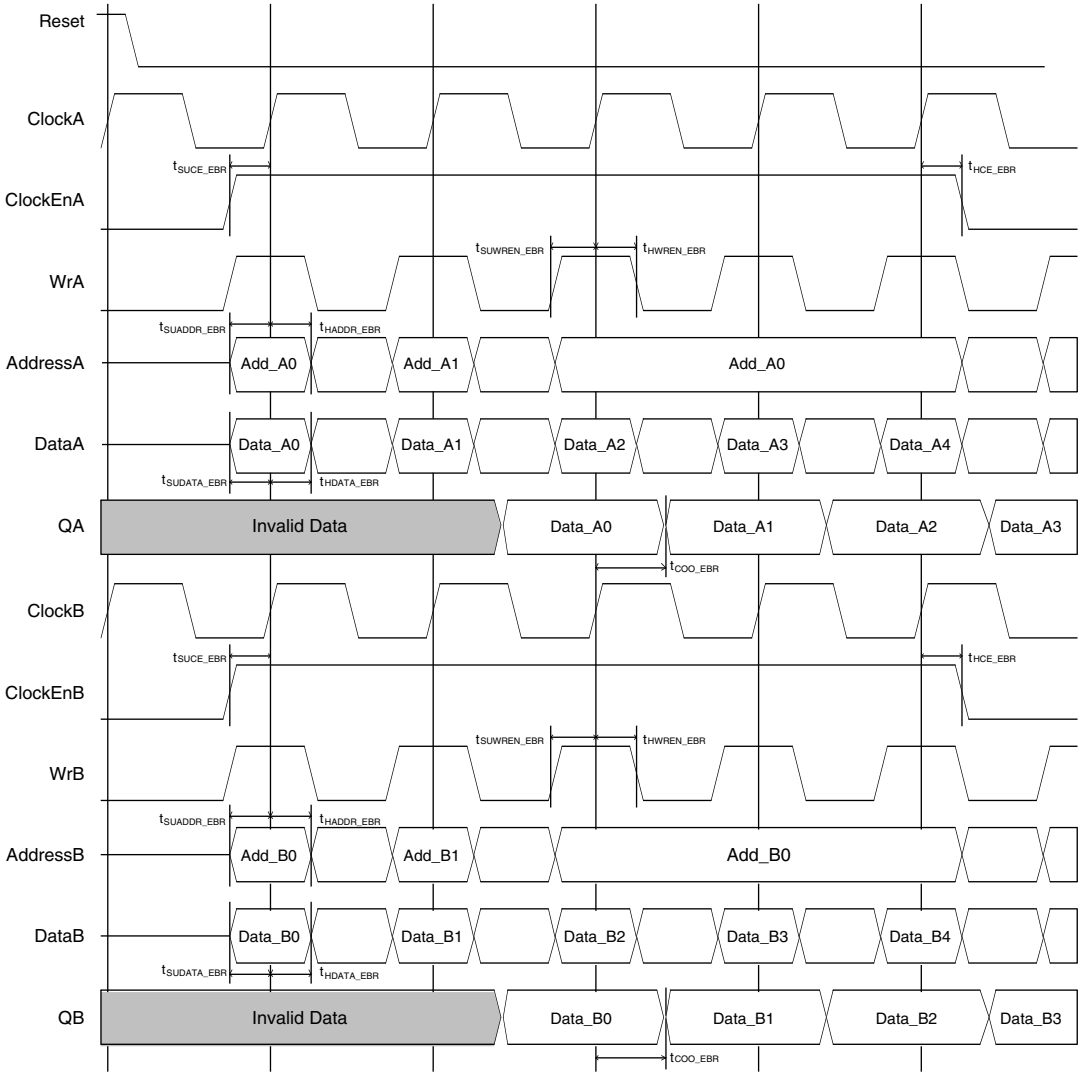


Figure 9-21. True Dual Port RAM Timing Waveform - WRITE THROUGH Mode, with Output Registers

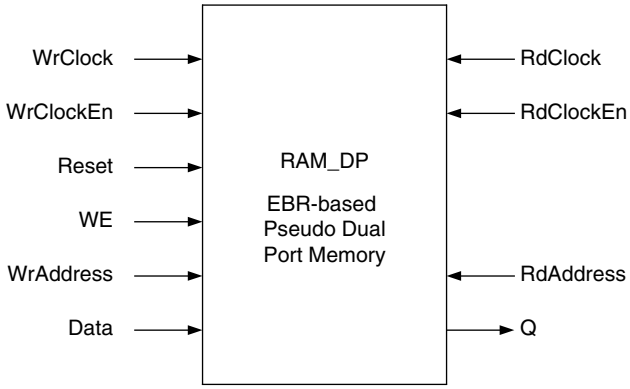


Pseudo Dual Port RAM (RAM_DP)- EBR Based

The EBR blocks in the MachXO devices can be configured as Pseudo-Dual Port RAM or RAM_DP. IPexpress allows users to generate the Verilog-HDL or VHDL along EDIF netlist for the memory size as per design requirements.

IPexpress generates the memory module as shown in Figure 9-22.

Figure 9-22. Pseudo Dual Port Memory Module Generated by IPexpress



The generated module makes use of these EBR blocks or primitives. For the memory sizes smaller than an EBR block, the module will be created in one EBR block. If the specified memory is larger than one EBR block, multiple EBR blocks can be cascaded in depth or width (as required to create these sizes).

In Pseudo Dual Port RAM mode, the input data and address for the ports are registered at the input of the memory array. The output data of the memory is optionally registered at the output.

The various ports and their definitions for the Single Port Memory are listed in Table 9-7. The table lists the corresponding ports for the module generated by IPexpress and for the EBR RAM_DP primitive.

Table 9-7. EBR-Based Pseudo-Dual Port Memory Port Definitions

Port Name in Generated Module	Port Name in the EBR Block Primitive	Description	Active State
RdAddress	ADR[x1:0]	Read Address	—
WrAddress	ADW[x2:0]	Write Address	—
RdClock	CLKR	Read Clock	Rising Clock Edge
WrClock	CLKW	Write Clock	Rising Clock Edge
RdClockEn	CER	Read Clock Enable	Active High
WrClockEn	CEW	Write Clock Enable	Active High
Q	DO[y1:0]	Read Data	—
Data	DI[y2:0]	Write Data	—
WE	WE	Write Enable	Active High
Reset	RST	Reset	Active High
—	CS[2:0]	Chip Select	—

Reset (RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

Chip Select (CS) is a useful port when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. Since CS is a 3-bit bus, it can cascade eight memories easily. However, if the memory size specified by the user requires more than eight EBR blocks, the

Diamond or ispLEVER software automatically generates the additional address decoding logic, which is implemented in the PFU external to the EBR blocks.

Each EBR block consists of 9,216 bits of RAM. The values for x's (for address) and y's (data) for each EBR block for the devices are as per Table 9-8.

Table 9-8. MachXO Pseudo-Dual Port Memory Sizes for 9K Memory

Pseudo-Dual Port Memory Size	Input Data Port A	Input Data Port B	Output Data Port A	Output Data Port B	Read Address Port A [MSB:LSB]	Write Address Port B [MSB:LSB]
8K x 1	DIA	DIB	DOA	DOB	RAD[12:0]	WAD[12:0]
4K x 2	DIA[1:0]	DIB[1:0]	DOA[1:0]	DOB[1:0]	RAD[11:0]	WAD[11:0]
2K x 4	DIA[3:0]	DIB[3:0]	DOA[3:0]	DOB[3:0]	RAD[10:0]	WAD[10:0]
1K x 9	DIA[8:0]	DIB[8:0]	DOA[8:0]	DOB[8:0]	RAD[9:0]	WAD[9:0]
512 x 18	DIA[17:0]	DIB[17:0]	DOA[17:0]	DOB[17:0]	RAD[8:0]	WAD[8:0]
256 x 36	DIA[35:0]	DIB[35:0]	DOA[35:0]	DOB[35:0]	RAD[7:0]	WAD[7:0]

Table 9-9 shows the various attributes available for the Pseudo-Dual Port Memory (RAM_DP). Some of these attributes are user-selectable through the IPexpress GUI. For detailed attribute definitions, refer to Appendix A.

Table 9-9. MachXO Pseudo-Dual Port RAM Attributes

Attribute	Description	Values	Default Value	User Selectable through IPexpress
DATA_WIDTH_W	Write Data Word Width	1, 2, 4, 9, 18, 36	18	YES
DATA_WIDTH_R	Read Data Word Width	1, 2, 4, 9, 18, 36	18	YES
REGMODE	Register Mode (Pipelining)	NOREG, OUTREG	NOREG	YES
RESETMODE	Selects the Reset type	ASYNC, SYNC	ASYNC	YES
CSDECODE_W	Chip Select Decode for Write	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
CSDECODE_R	Chip Select Decode for Read	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
ENABLE GSR	Enables Global Set Reset	ENABLE, DISABLE	ENABLED	YES

Users have the option of enabling the output registers for Pseudo-Dual Port RAM (RAM_DP). Figures 9-23 and 9-24 show the internal timing waveforms for Pseudo-Dual Port RAM (RAM_DP) with these options.

Figure 9-23. PSEUDO DUAL PORT RAM Timing Diagram – without Output Registers

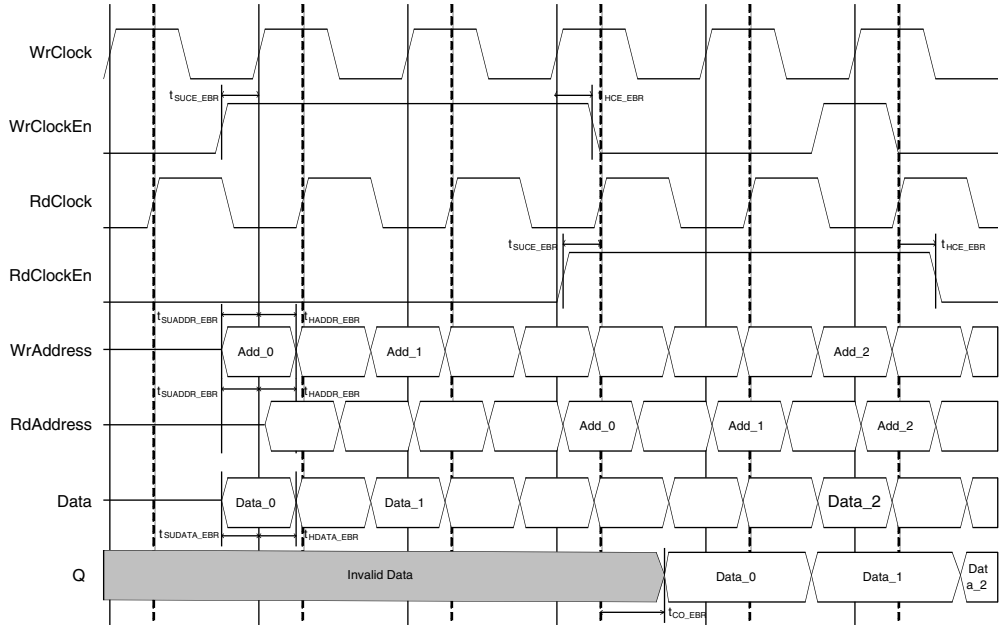
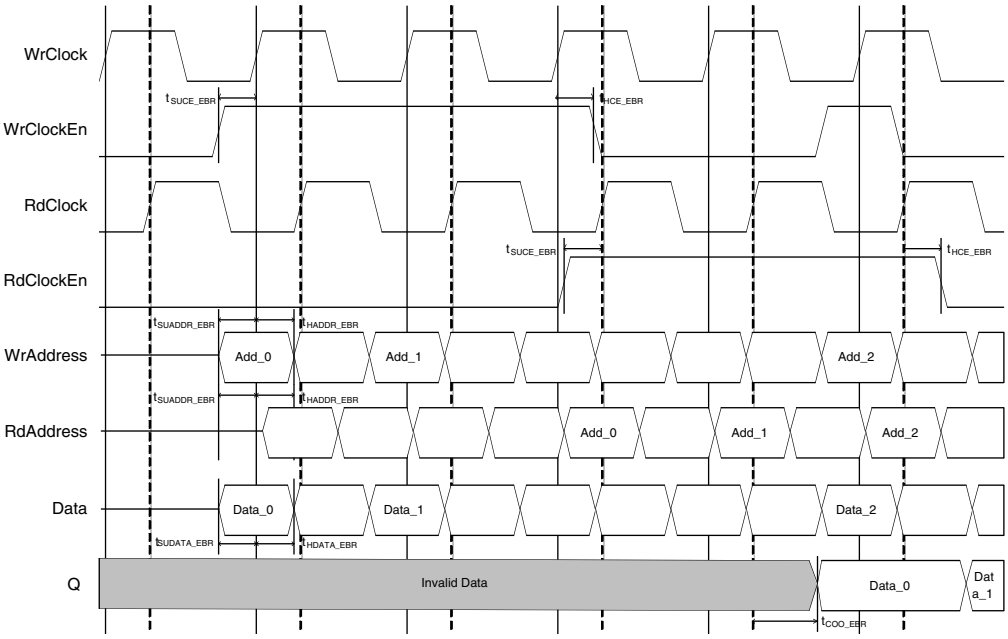


Figure 9-24. PSEUDO DUAL PORT RAM Timing Diagram – with Output Registers

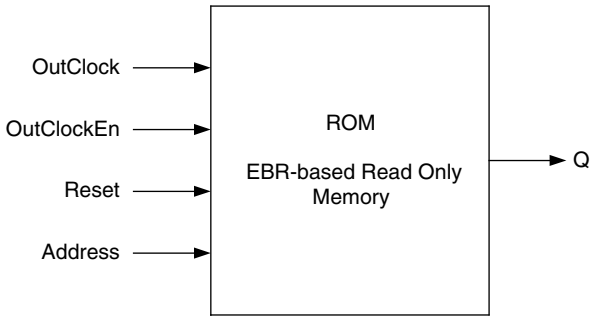


Read Only Memory (ROM) – EBR Based

The EBR blocks in the MachXO devices can be configured as Read Only Memory or ROM. IPexpress allows users to generate the Verilog-HDL or VHDL along EDIF netlist for the memory size, as per design requirement. Users are required to provide the ROM memory content in the form of an initialization file.

IPexpress generates the memory module as shown in Figure 9-25.

Figure 9-25. ROM – Read Only Memory Module Generated by IPexpress



The generated module makes use of these EBR blocks or primitives. For the memory sizes smaller than an EBR block, the module will be created in one EBR block. If the specified memory is larger than one EBR block, multiple EBR blocks can be cascaded, in depth or width (as required to create these sizes).

In ROM mode, the address for the port is registered at the input of the memory array. The output data of the memory is optionally registered at the output.

The various ports and their definitions for the ROM are listed in Table 9-10. The table lists the corresponding ports for the module generated by IPexpress and for the ROM primitive.

Table 9-10. EBR-based ROM Port Definitions

Port Name in Generated Module	Port Name in the EBR Block Primitive	Description	Active State
Address	AD[x:0]	Read Address	—
OutClock	CLK	Clock	Rising Clock Edge
OutClockEn	CE	Clock Enable	Active High
Reset	RST	Reset	Active High
—	CS[2:0]	Chip Select	—

Reset (RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

Chip Select (CS) is a useful port when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. Since CS is a 3-bit bus, it can cascade eight memories easily. However, if the memory size specified by the user requires more than eight EBR blocks, the Diamond or ispLEVER software automatically generates the additional address decoding logic, which is implemented in the PFU external to the EBR blocks.

While generating the ROM using IPexpress, the user must provide the initialization file to pre-initialize the contents of the ROM. These files are the *.mem files and they can be of Binary, Hex or the Addressed Hex formats. The initialization files are discussed in detail in the Initializing Memory section of this document.

Users have the option of enabling the output registers for Read Only Memory (ROM). Figures 9-26 and 9-27 show the internal timing waveforms for the Read Only Memory (ROM) with these options.

Figure 9-26. ROM Timing Waveform – without Output Registers

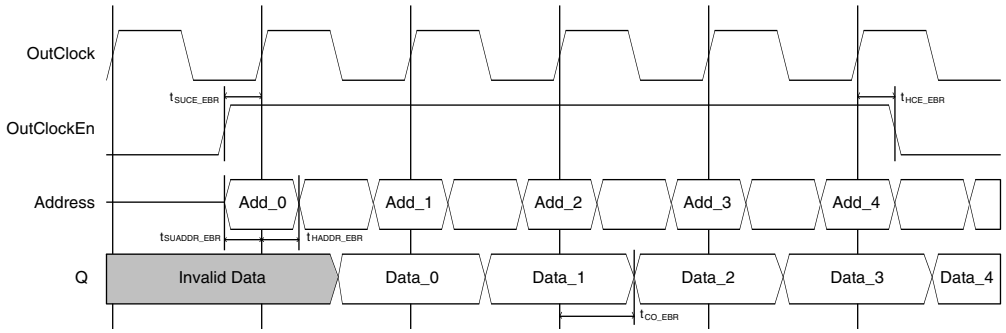
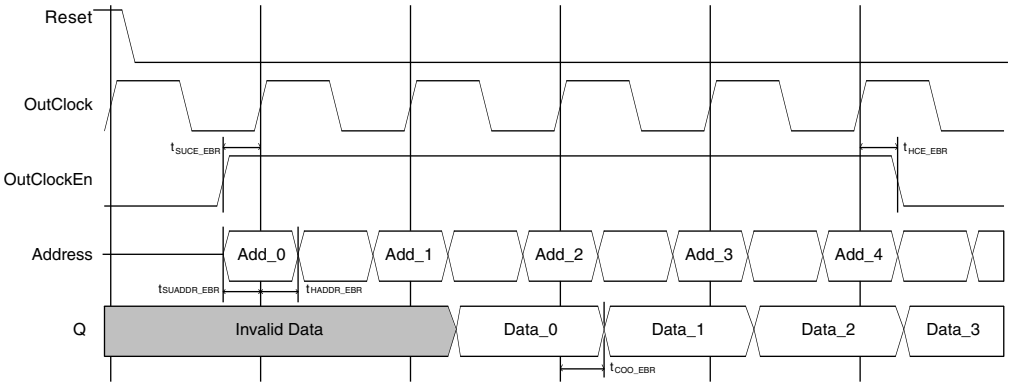


Figure 9-27. ROM Timing Waveform – with Output Registers

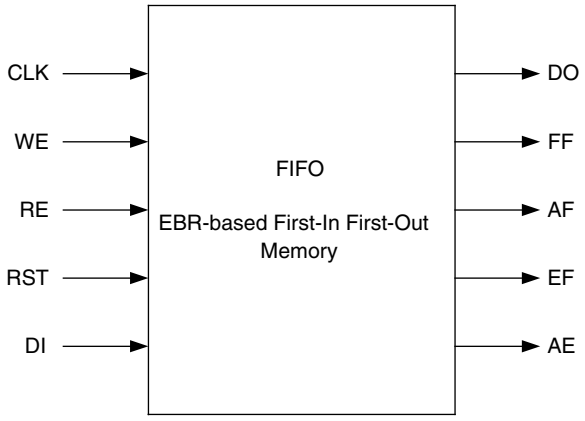


First In First Out (FIFO, FIFO_DC) - EBR Based

The EBR blocks in MachXO devices can be configured as Dual Clock First In First Out Memories, FIFO_DC. IPexpress allows users to generate the Verilog-HDL or VHDL along EDIF netlist for the memory size, according to design requirements.

IPexpress generates the FIFO_DC memory module as shown in Figure 9-28.

Figure 9-28. FIFO Module Generated by IPexpress



The generated module makes use of these EBR blocks or primitives. For the memory sizes smaller than an EBR block, the module will be created in one EBR block. If the specified memory is larger than one EBR block, multiple EBR blocks can be cascaded, in depth or width (as required to create these sizes).

A clock is always required, as only synchronous write is supported. The various ports and their definitions for the FIFO_DC are listed in Table 9-11.

Table 9-11. EBR-based FIFO_DC Memory Port Definitions

Port Name in Generated Module	Port Name in Primitive	Description	Active State
CLKR		Read Port Clock	Rising Clock Edge
CLKW		Write Port Clock	Rising Clock Edge
WE		Write Enable	Active High
RE		Read Enable	Active High
RST		Reset	Active High
DI		Data Input	—
DO		Data Output	—
FF		Full Flag	Active High
AF		Almost Full Flag	Active High
EF		Empty Flag	Active High
AE		Almost Empty	Active High

Reset (RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

The various supported sizes for the FIFO_DC for MachXO are listed in Table 9-12.

Table 9-12. MachXO FIFO_DC Data Widths Sizes

FIFO Size	Input Data	Output Data
8K x 1	DI	DO
4K x 2	DI[1:0]	DO[1:0]
2K x 4	DI[3:0]	DO[3:0]
1K x 9	DI[8:0]	DO[8:0]
512 x 18	DI[17:0]	DO[17:0]
256 x 36	DI[35:0]	DO[35:0]

Table 9-13 shows the various attributes available for the FIFO_DC. Some of these attributes are user-selectable through the IPexpress GUI. For detailed attribute definitions, refer to Appendix A.

Table 9-13. FIFO and FIFO_DC Attributes for MachXO

Attribute	Description	Values	Default Value
DATA_WIDTH_W	Data Width Write Mode	1, 2, 4, 9, 18, 36	18
DATA_WIDTH_R	Data Width Read Mode	1, 2, 4, 9, 18, 36	18
REGMODE	Register Mode	NOREG, OUTREG	NOREG
RESETMODE	Select Reset Type	ASYNC, SYNC	ASYNC
CSDECODE_W	Chip Select Decode for Write Mode	0b00, 0b01, 0b10, 0b11	0
CSDECODE_R	Chip Select Decode for Read Mode	0b00, 0b01, 0b10, 0b11	0
GSR	Enable Global Set Reset	ENABLED, DISABLED	ENABLED
AEPOINTER	Almost Empty Pointer	Ob00000000000000, ..., 0b11111111111111	—
AFPOINTER	Almost Full Pointer	Ob00000000000000, ..., 0b11111111111111	—
FULLPOINTER	Full Pointer	Ob00000000000000, ..., 0b11111111111111	—
FULLPOINTER1	Full Pointer1	Ob00000000000000, ..., 0b11111111111111	—
AFPOINTER1	Almost Full Pointer1	Ob00000000000000, ..., 0b11111111111111	—
AEPOINTER1	Almost Empty Pointer1	Ob00000000000000, ..., 0b11111111111111	—

FIFO_DC Flags

The FIFO_DC have four flags available: Empty, Almost Empty, Almost Full and Full. Almost Empty and Almost Full flags have a programmable range.

The program ranges for the four FIFO_DC flags are specified in Table 9-14.

Table 9-14. FIFO_DC Flag Settings

FIFO Attribute Name	Description	Programming Range	Program Bits
FF	Full flag setting	2^N - 1	14
AFF	Almost full setting	1 to (FF-1)	14
AEF	Almost empty setting	1 to (FF-1)	14
EF	Empty setting	0	5

The only restriction on the flag setting is that the values must be in a specific order (Empty=0, Almost Empty next, followed by Almost Full and Full, respectively). The value of Empty is not equal to the value of Almost Empty (or Full is equal to Almost Full). In this case, a warning is generated and the value of Empty (or Full) is used in place of Almost Empty (or Almost Full). When coming out of reset, the active high flags Empty and Almost Empty are set to high, since they are true.

The user should specify the absolute value of the address at which the Almost Empty and Almost Full flags will go true. For example, if the Almost Full flag is required to go true at the address location 500 for a FIFO of depth 512, the user should specify a value of 500 in IPexpress.

The Empty and Almost Empty flags are always registered to the read clock and the Full and Almost Full flags are always registered to the write clock.

At reset both the write and read counters are pointing to address zero. After reset is de-asserted data can be written into the FIFO_DC to the address pointed to by the write counter at the positive edge of the write clock when the write enable is asserted.

Similarly, data can be read from the FIFO_DC from the address pointed to by the read counter at the positive edge of the read clock when read enable is asserted.

Read Pointer Reset (RPReset) is used to indicate a retransmit, and is more commonly used in “packetized” communications. In this application, the user must keep careful track of when a packet is written into or read from the FIFO_DC.

The data output of the FIFO_DC can be registered or non-registered through a selection in IPexpress. The output registers are enabled by read enable. A reset will clear the contents of the FIFO_DC by resetting the read and write pointers and will put the flags in the initial reset state.

FIFO_DC Operation

If the output registers are not enabled it will take two clock cycles to read the first word out. The register for the flag logic causes this extra clock latency. In the architecture of the emulated FIFO_DC, the internal read enables for reading the data out is controlled not only by the read enable provided by the user but also the empty flag. When the data is written into the FIFO, an internal empty flag is registered using write clock that is enabled by write enable (WrEn). Another clock latency is added due to the clock domain transfer from write clock to read clock using another register which is clocked by read clock that is enabled by read enable.

Internally, the output of this register is inverted and then ANDed with the user-provided read enable that becomes the internal read enable to the RAM_DP which is at the core of the FIFO_DC.

Thus, the first read data takes two clock cycles to propagate through. During the first data out, read enable goes high for one clock cycle, empty flag is de-asserted and is not propagated through the second register enabled by the read enable. The first clock cycle brings the Empty Low and the second clock cycle brings the internal read enable high (RdEn and !EF) and then the data is read out by the second clock cycle. Similarly, the first write data after the full flag has a similar latency.

If the user has enabled the output registers, the output registers will cause an extra clock delay during the first data out as they are clocked by the read clock and enabled by the read enable.

1. First RdEn and Clock Cycle to propagate the EF internally.
2. Second RdEn and Clock Cycle to generate internal Read Enable into the DPRAM.
3. Third RdEn and Clock Cycle to get the data out of the output registers.

Figure 9-29. FIFO_DC without Output Registers (Non-Pipelined)

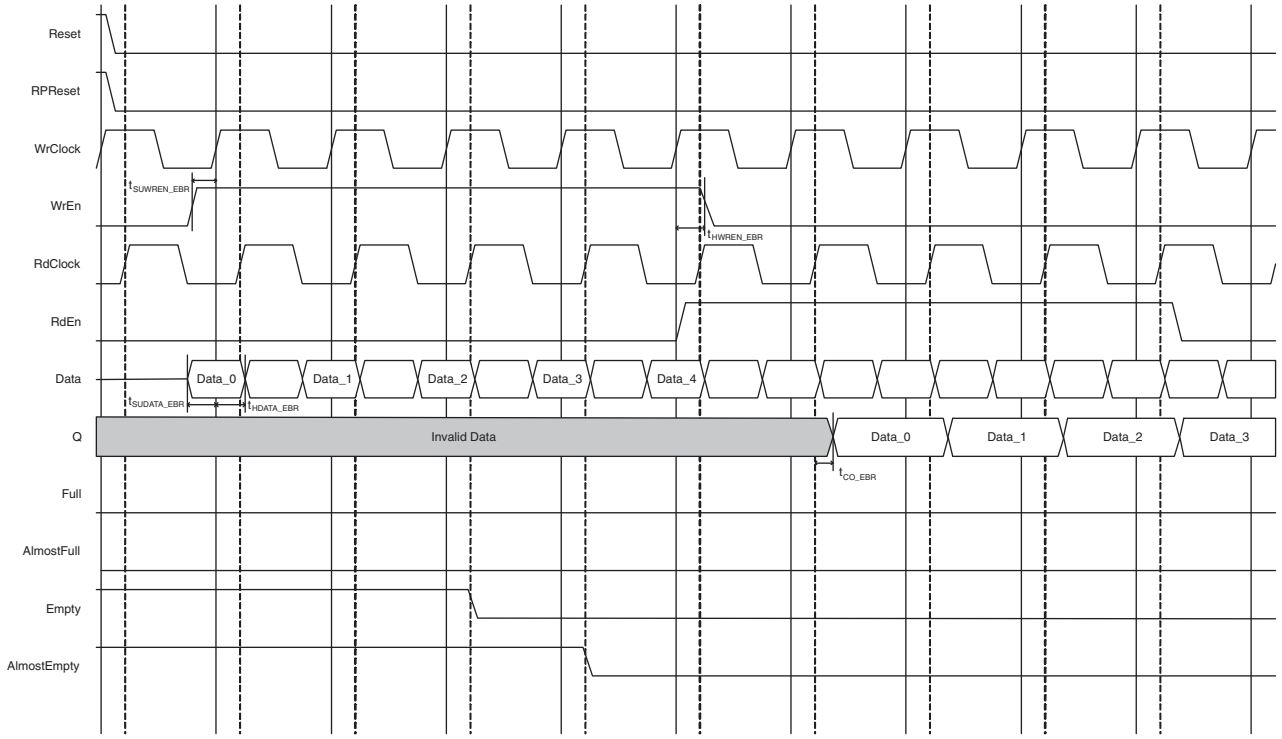
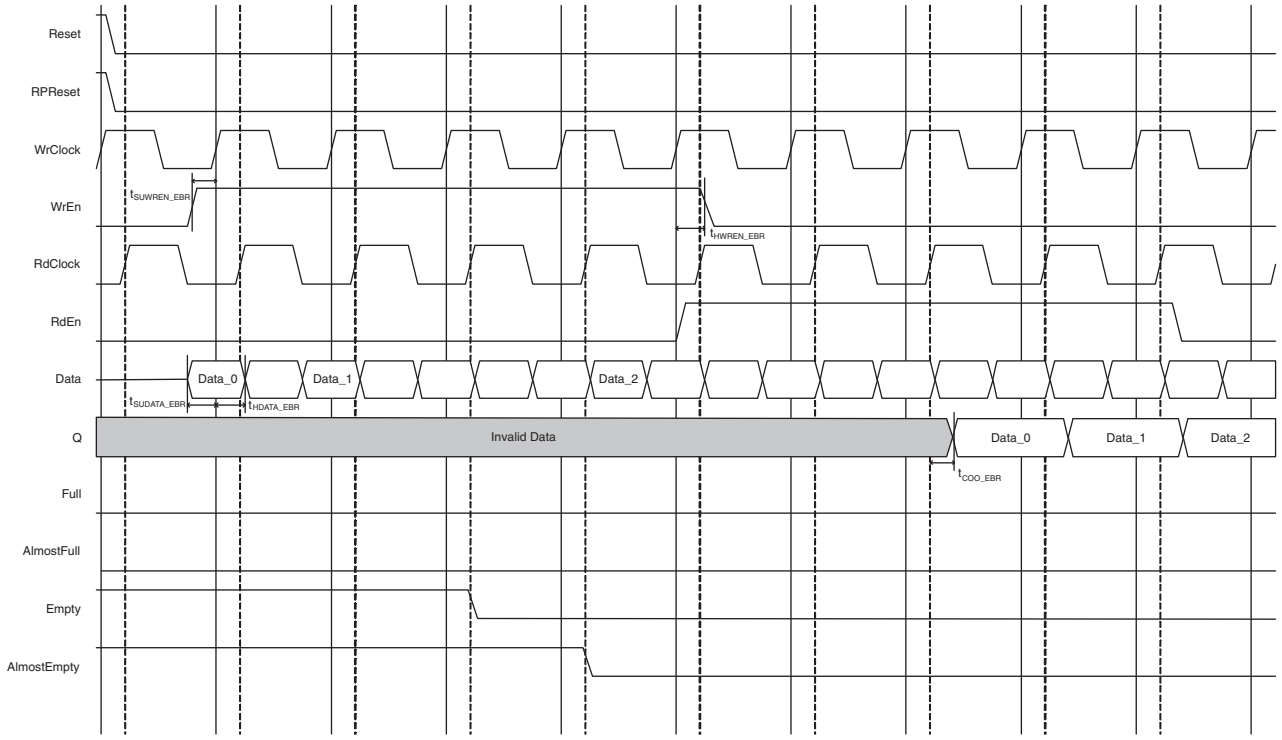


Figure 9-30. FIFO_DC with Output Registers (Pipelined)

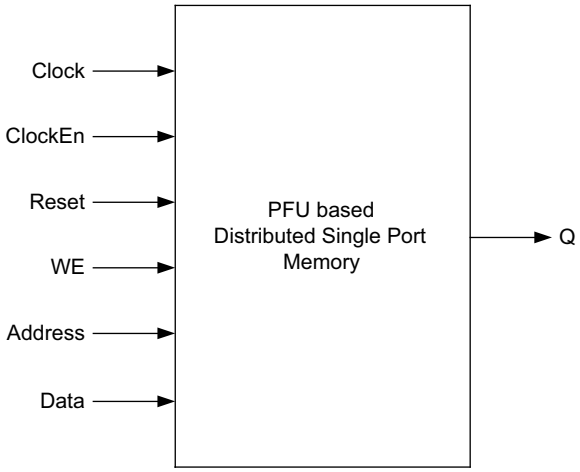


Distributed Single Port RAM (Distributed_SPRAM) – PFU Based

PFU based Distributed Single Port RAM is created using the 4-input LUT (Look-Up Table) available in the PFU. These LUTs can be cascaded to create larger distributed memory sizes.

Figure 9-31 shows the Distributed Single Port RAM module as generated by IPexpress.

Figure 9-31. Distributed Single Port RAM Module Generated by IPexpress



The generated module makes use of the 4-input LUT available in the PFU. Additional logic such as clock and reset is generated by utilizing the resources available in the PFU.

Ports such as Read Clock (RdClock) and Read Clock Enable (RdClockEn) are not available in the hardware primitive. These are generated by the IPexpress when the user wants the to enable the output registers in the IPexpress configuration.

The various ports and their definitions are listed in Table 9-15. The table lists the corresponding ports for the module generated by IPexpress and for the primitive.

Table 9-15. PFU-based Distributed Single Port RAM Port Definitions

Port Name in Generated Module	Port Name in the PFU Primitive	Description	Active State
Clock	CK	Clock	Rising Clock Edge
ClockEn	—	Clock Enable	Active High
Reset	—	Reset	Active High
WE	WRE	Write Enable	Active High
Address	AD[3:0]	Address	—
Data	DI[1:0]	Data In	—
Q	DO[1:0]	Data Out	—

Ports such as Clock Enable (ClockEn) are not available in the hardware primitive. These are generated by IPexpress when the user wishes the to enable the output registers in the IPexpress configuration.

Users have the option of enabling the output registers for Distributed Single Port RAM (Distributed_SPRAM). Figures 9-32 and 9-33 show the internal timing waveforms for the Distributed Single Port RAM (Distributed_SPRAM) with these options.

Figure 9-32. PFU-Based Distributed Single Port RAM Timing Waveform – Without Output Registers

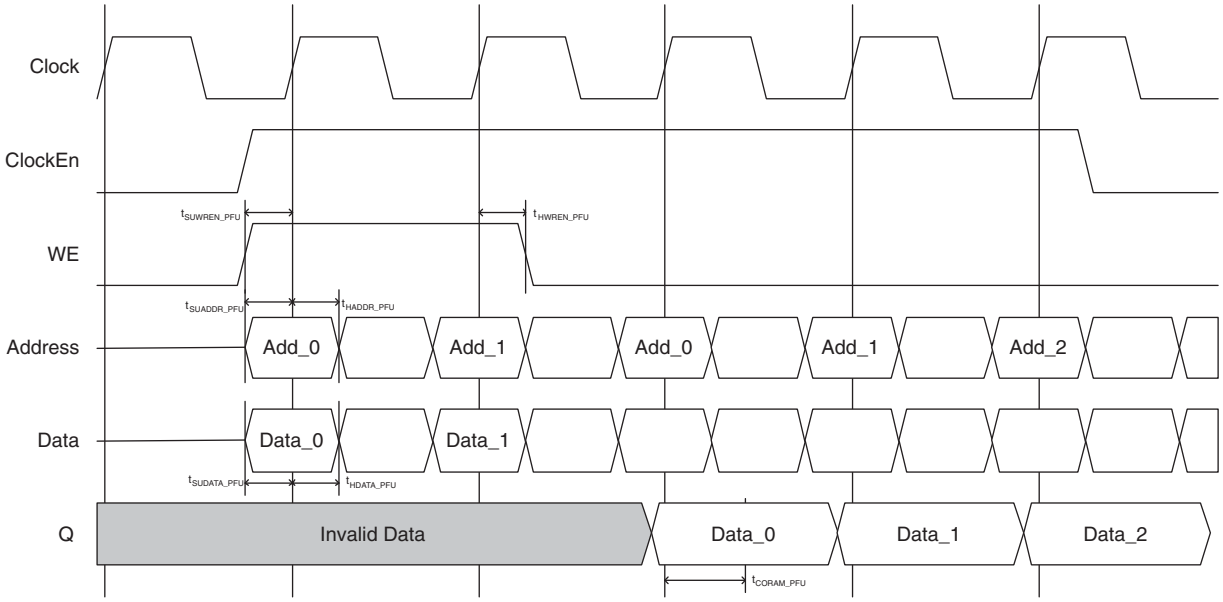
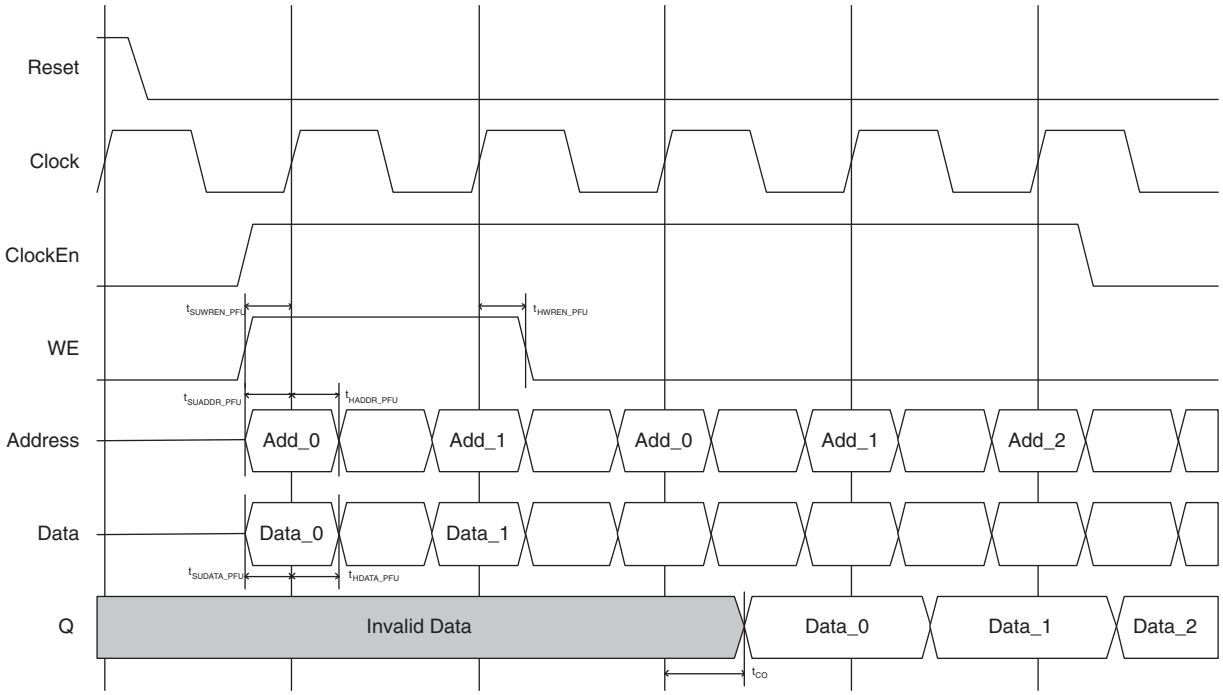


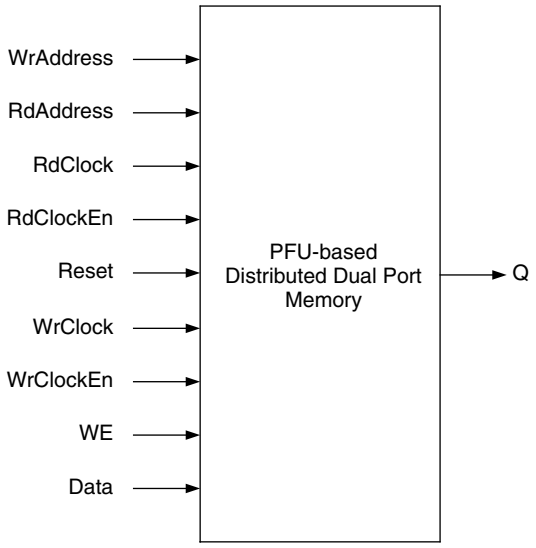
Figure 9-33. PFU-Based Distributed Single Port RAM Timing Waveform - With Output Registers



Distributed Dual Port RAM (Distributed_DPRAM) – PFU Based

PFU-based Distributed Dual Port RAM is also created using the 4-input LUT (Look-Up Table) available in the PFU. These LUTs can be cascaded to create larger distributed memory sizes.

Figure 9-34. Distributed Dual Port RAM Module Generated by IPexpress



The generated module makes use of the 4-input LUT available in the PFU. Additional such as Clock and Reset is generated by utilizing the resources available in the PFU.

The various ports and their definitions are listed in Table 9-16. The table lists the corresponding ports for the module generated by IPexpress and for the primitive.

Table 9-16. PFU based Distributed Dual-Port RAM Port Definitions

Port Name in Generated Module	Port Name in the EBR Block Primitive	Description	Active State
WrAddress	WAD[3:0]	Write Address	—
RdAddress	RAD[3:0]	Read Address	—
RdClock	—	Read Clock	Rising Clock Edge
RdClockEn	—	Read Clock Enable	Active High
WrClock	WCK	Write Clock	Rising Clock Edge
WrClockEn	—	Write Clock Enable	Active High
WE	WRE	Write Enable	Active High
Data	DI[1:0]	Data Input	—
Q	RDO[1:0]	Data Out	—

Ports such as Read Clock (RdClock) and Read Clock Enable (RdClockEn) are not available in the hardware primitive. These are generated by IPexpress when the user wants the to enable the output registers in the IPexpress configuration.

Users have the option of enabling the output registers for Distributed Dual Port RAM (Distributed_DPRAM). Figures 9-35 and 9-36 show the internal timing waveforms for the Distributed Dual Port RAM (Distributed_DPRAM) with these options.

Figure 9-35. PFU-Based Distributed Dual Port RAM Timing Waveform – without Output Registers (Non-Pipelined)

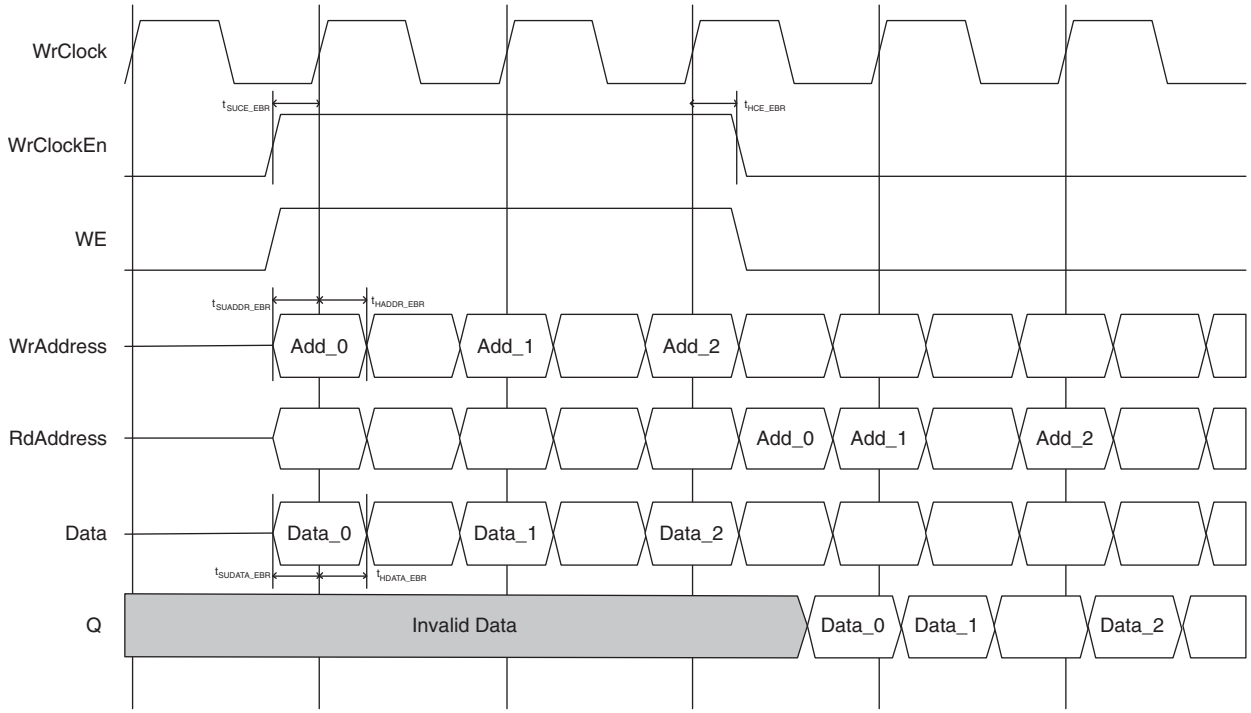
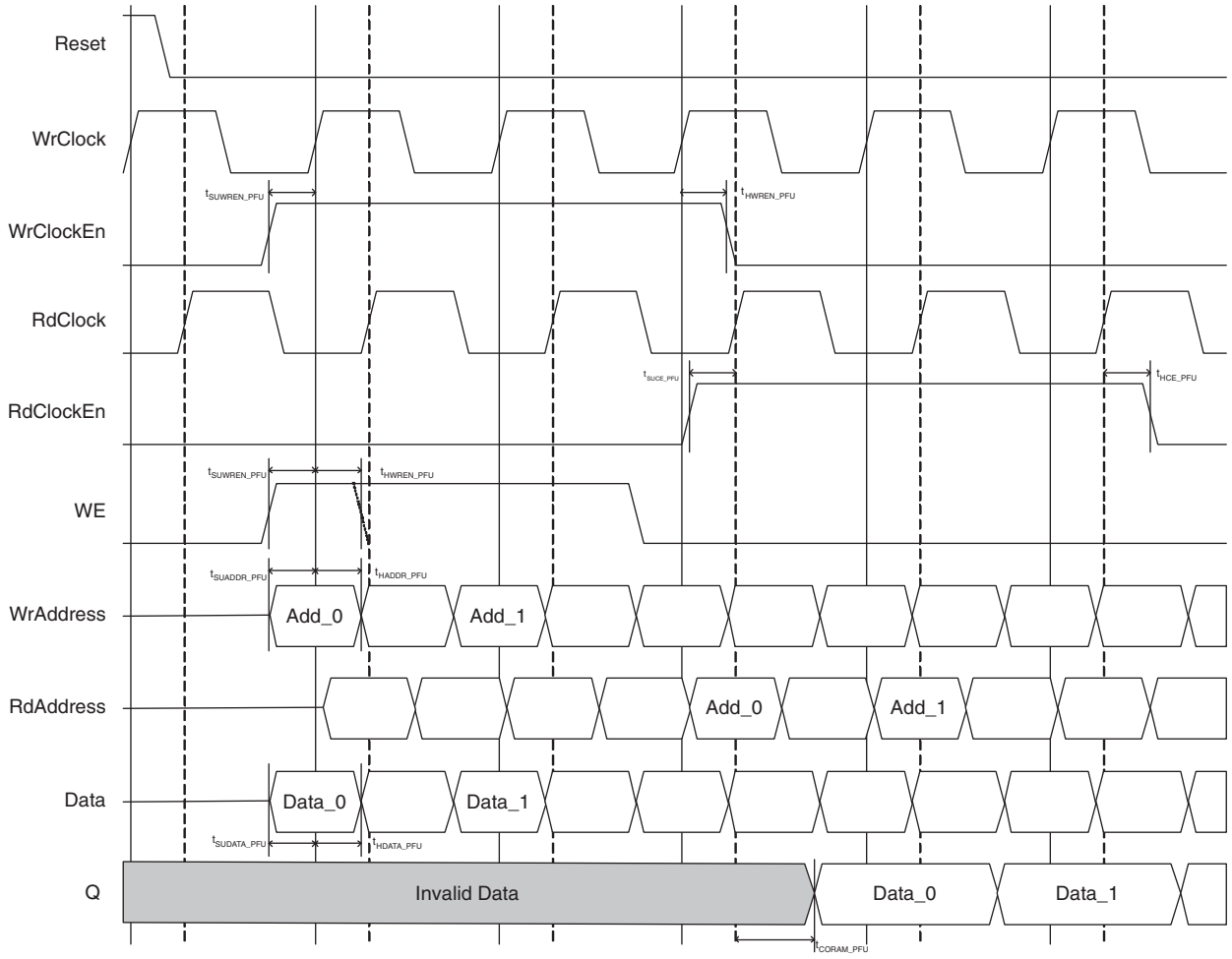


Figure 9-36. PFU-Based Distributed Dual Port RAM Timing Waveform – with Output Registers (Pipelined)

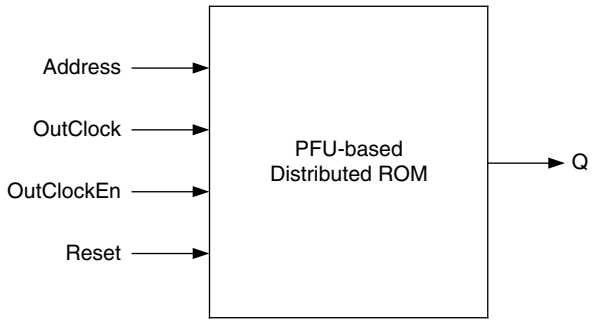


Distributed ROM (Distributed_ROM) – PFU Based

PFU-based Distributed ROM is also created using the 4-input LUT (Look-Up Table) available in the PFU. These LUTs can be cascaded to create larger distributed memory sizes.

Figure 9-37 shows the Distributed ROM module as generated by IPexpress.

Figure 9-37. Distributed ROM Generated by IPexpress



The generated module makes use of the 4-input LUT available in the PFU. Additional logic like clock and reset is generated by utilizing the resources available in the PFU.

Ports such as Out Clock (OutClock) and Out Clock Enable (OutClockEn) are not available in the hardware primitive. These are generated by IPexpress when the user wants the to enable the output registers in the IPexpress configuration.

The various ports and their definitions are listed in Table 9-17. The table lists the corresponding ports for the module generated by IPexpress and for the primitive.

Table 9-17. PFU-based Distributed ROM Port Definitions

Port Name in Generated Module	Port Name in the PFU Block Primitive	Description	Active State
Address	AD[3:0]	Address	—
OutClock	—	Out Clock	Rising Clock Edge
OutClockEn	—	Out Clock Enable	Active High
Reset	—	Reset	Active High
Q	DO	Data Out	—

Users have the option of enabling the output registers for Distributed ROM (Distributed_ROM). Figures 9-38 and 9-39 show the internal timing waveforms for the Distributed ROM with these options.

Figure 9-38. PFU-Based ROM Timing Waveform – Without Output Registers

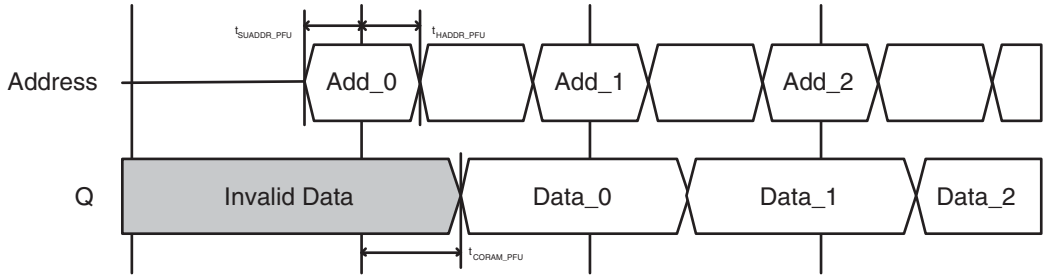
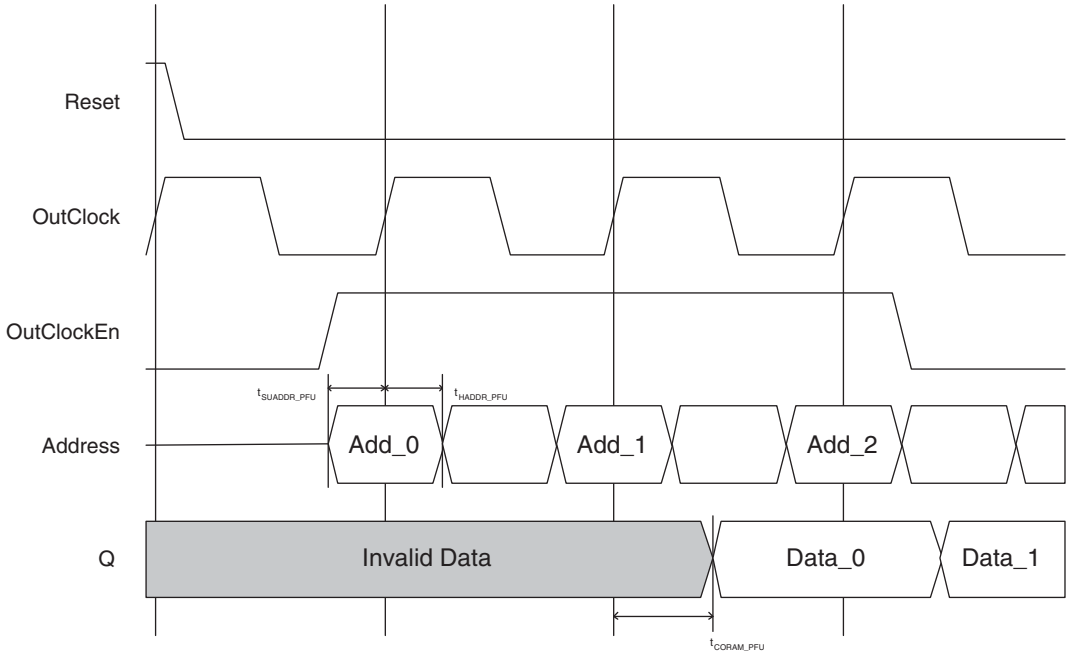


Figure 9-39. PFU-Based ROM Timing Waveform – with Output Registers



Initializing Memory

In the EBR based ROM or RAM memory modes and the PFU based ROM memory mode, it is possible to specify the power-on state of each bit in the memory array. Each bit in the memory array can have one of two values: 0 or 1.

Initialization File Formats

The initialization file is an ASCII file, which can be created or edited using any ASCII editor. IPexpress supports three types of memory file formats:

1. Binary file
2. Hex file
3. Addressed Hex

The file name for the initialization file is *.mem (<file_name>.mem). Each row depicts the value to be stored in a particular memory location and the number of characters (or the number of columns) represents the number of bits for each address (or the width of the memory module).

The initialization file is primarily used for configuring the ROMs. The EBR in RAM mode can also use the initialization file to preload the memory contents.

Binary File

The binary file is a text file of 0's and 1's. The rows indicate the number of words and the columns indicate the width of the memory.

```
Memory Size 20x32
00100000010000000010000001000000
00000001000000010000000100000001
00000010000000100000001000000010
00000011000000110000001100000011
00000100000001000000010000000100
00000101000001010000010100000101
```

```
00000110000001100000011000000110
00000111000001110000011100000111
00001000010010000000100001001000
00001001010010010000100101001001
00001010010010100000101001001010
00001011010010110000101101001011
00001100000011000000110000001100
00001101001011010000110100101101
00001110001111100000111000111110
00001111001111110000111100111111
00010000000100000001000000010000
00010001000100010001000100010001
00010010000100100001001000010010
00010011000100110001001100010011
```

Hex File
The hex file is essentially a text file of hex characters arranged in a similar row-column arrangement. The number of rows in the file is same as the number of address locations, with each row indicating the content of the memory location.

```
Memory Size 8x16
A001
0B03
1004
CE06
0007
040A
0017
02A4
```

Addressed Hex
Addressed hex consists of lines of address and data. Each line starts with an address, followed by a colon, and any number of data. The format of the memfile is address: data data data data ... where address and data are hexadecimal numbers.

- A0: 03 F3 3E 4F
- B2: 3B 9F

The first line puts 03 at address A0, F3 at address A1, 3E at address A2,and 4F at address A3. The second line puts 3B at address B2 and 9F at address B3.

There is no limitation on the values of address and data. The value range is automatically checked based on the values of addr_width and data_width. If there is an error in an address or data value, an error message is printed. Users need not specify data at all address locations. If data is not specified at a certain address, the data at that location is initialized to 0. IPexpress makes memory initialization possible through both the synthesis and simulation flows.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
—	—	Previous Lattice releases.
April 2006	01.2	Updated the Initializing Memory section.
October 2006	01.3	Added dual port memory access notes in Appendix A.
September 2007	01.4	Updated FIFO_DC without Output Registers (Non-pipelined) and FIFO_DC with Output Registers (Pipelined) waveforms.
October 2010	01.5	Updated for Lattice Diamond design software support.

Appendix A. Attribute Definitions

DATA_WIDTH
Data width is associated with the RAM and FIFO elements. The DATA_WIDTH attribute defines the number of bits in each word. It takes the values defined in the RAM size tables in each memory module.

REGMODE
REGMODE, or the Register mode attribute, is used to enable pipelining in the memory. This attribute is associated with the RAM and FIFO elements. The REGMODE attribute takes the NOREG or OUTREG mode parameter that disables and enables the output pipeline registers.

RESETMODE
The RESETMODE attribute allows users to select the mode of reset in the memory. This attribute is associated with the block RAM elements. RESETMODE takes two parameters: SYNC and ASYNC. SYNC means that the memory reset is synchronized with the clock. ASYNC means that the memory reset is asynchronous to clock.

CSDECODE
CSDECODE, or the Chip Select Decode attributes, are associated to block RAM elements. Chip Select (CS) is a useful port when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. CS is a 3-bit bus, so it can cascade eight memories easily. CSDECODE takes the following parameters: “000”, “001”, “010”, “011”, “100”, “101”, “110”, and “111”. CSDECODE values determine the decoding value of CS[2:0]. CSDECODE_W is chip select decode for write and CSDECODE_R is chip select decode for read for Pseudo Dual Port RAM. CSDECODE_A and CSDECODE_B are used for true dual port RAM elements and refer to the A and B ports.

WRITEMODE
The WRITEMODE attribute is associated with the block RAM elements. It takes the NORMAL, WRITETHROUGH, and READBEFOREWRITE mode parameters.

In NORMAL mode, the output data does not change or get updated during the write operation. This mode is supported for all data widths.

In WRITETHROUGH mode, the output data is updated with the input data during the write cycle. This mode is supported for all data widths.

In READBEFOREWRITE mode, the output data port is updated with the existing data stored in the write address, during a write cycle. This mode is supported for x9, x18 and x36 data widths.

WRITEMODE_A and WRITEMODE_B are used for dual port RAM elements and refer to the A and B ports in case of a True Dual Port RAM.

For all modes of the True Dual Port module, simultaneous read access from one port and write access from the other port to the same memory address is not recommended. The read data may be unknown in this situation. Also, simultaneous write access to the same address from both ports is not recommended. When this occurs, the data stored in the address becomes undetermined when one port tries to write a 'H' and the other tries to write a 'L'.

It is recommended that users implement control logic to identify this situation if it occurs and then either:

1. Implement status signals to flag the read data as possibly invalid, or
2. Implement control logic to prevent the simultaneous access from both ports.

GSR
GSR, the Global Set/ Reset attribute, is used to enable or disable the global set/reset for the RAM element.

MachXO sysCLOCK Design and Usage Guide

October 2010 Technical Note TN1089

Introduction

As clock distribution and clock skew management become critical factors in overall system performance, the Phase Locked Loop (PLL) is increasing in importance for digital designers. Lattice incorporates its sysCLOCK™ PLL technology in the MachXO™ device family to help designers manage clocks within their designs. The PLL components in the MachXO device family use the same PLL as the LatticeECP™, LatticeEC™ and LatticeXP™ families.

The number of PLLs for each device are listed in Table 10-1.

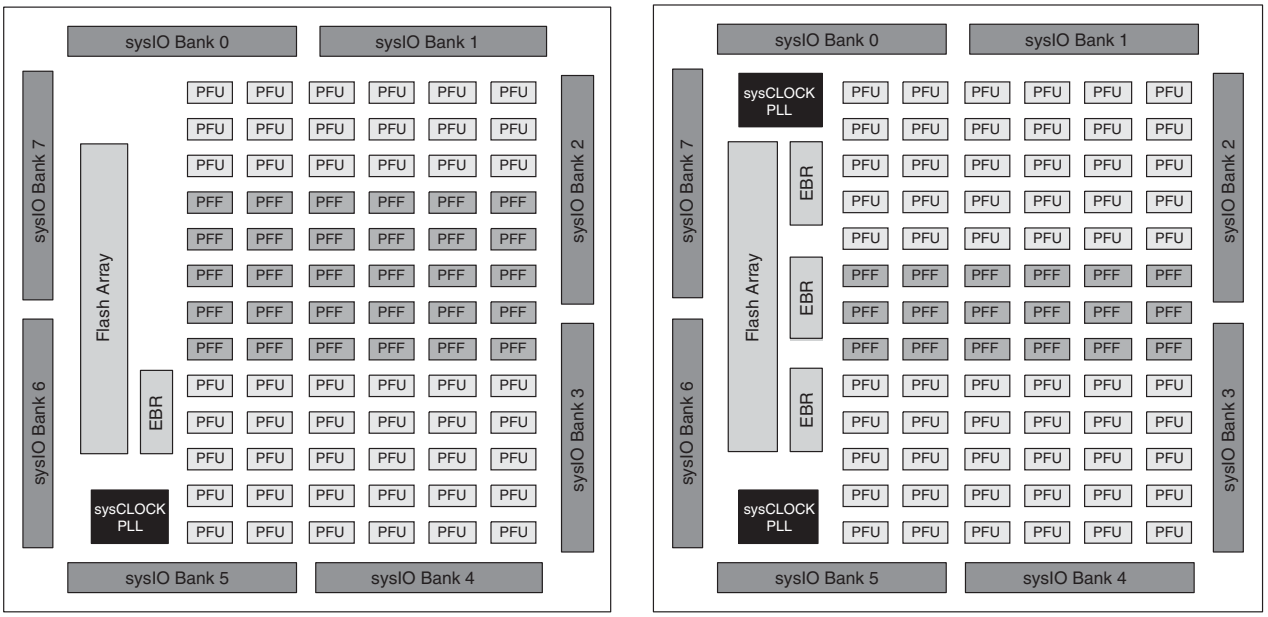
Table 10-1. MachXO Family and PLLs

	MachXO256	MachXO640	MachXO1200	MachXO2280
Number of PLLs	0	0	1	2

MachXO Top Level View

Figure 10-1 shows a chip-level view of the MachXO1200 and MachXO2280.

Figure 10-1. Top Level View of MachXO1200 and MachXO2280

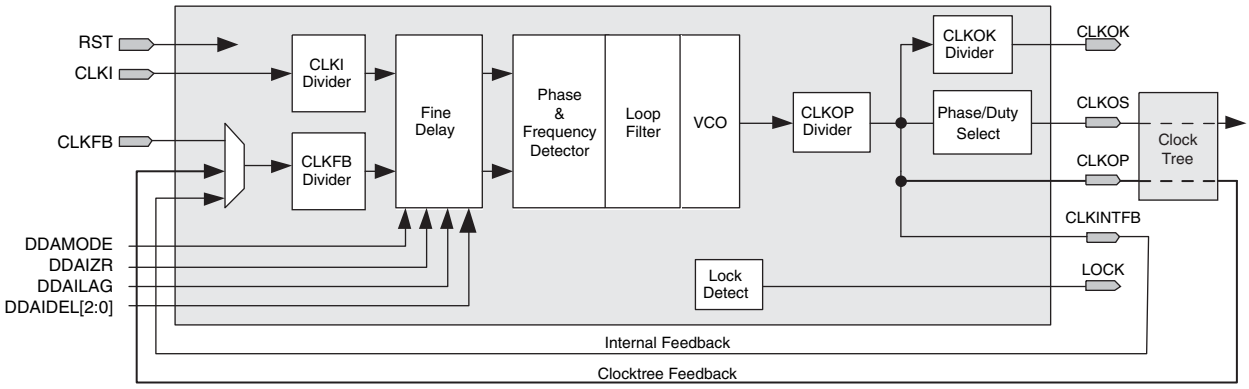


MachXO1200 MachXO2280

sysCLOCK PLL

This technical note describes the features and functions of the PLLs and their configuration in the ispLEVER® design tool. Figure 10-2 shows the block diagram of the PLL.

Figure 10-2. PLL Block Diagram



Features

- Clock synthesis
- Phase shift/duty cycle selection
- Internal, clock tree and external feedback
- Dynamic delay adjustment
- No external components required
- Lock detect output

Functional Description

PLL Divider and Delay Blocks

Input Clock (CLKI) Divider

The CLKI divider is used to control the input clock frequency into the PLL block. The divider setting directly corresponds to the divisor of the output clock. The input and output of the input divider must be within the input and output frequency ranges specified in the MachXO Family Data Sheet.

Feedback Loop (CLKFB) Divider

The CLKFB divider is used to divide the feedback signal. Effectively, this multiplies the output clock, because the divided feedback must speed up to match the input frequency into the PLL block. The PLL block increases the output frequency until the divided feedback frequency equals the input frequency.The input and output of the feedback divider must be within the input and output frequency ranges specified in the MachXO Family Data Sheet.

Delay Adjustment

The delay adjust circuit provides programmable clock delay. The programmable clock delay allows for step delays in increments of 250ps (nominal) for a total of 2.00ns lagging or leading. The time delay setting has a tolerance. See the MachXO Family Data Sheet for details. Under this mode, CLKOP, CLKOS and CLKOK are identically affected. The delay adjustment has two modes of operation:

Static Delay Adjustment: In this mode, the user-selected delay is configured at power-up.

Dynamic Delay Adjustment (DDA): In this mode, a simple bus is used to configure the delay. The bus signals are available to the general purpose FPGA.

Output Clock (CLKOP) Divider

The CLKOP divider serves the dual purposes of squaring the duty cycle of the VCO output and scaling up the VCO frequency into the 420MHz to 840MHz range to minimize jitter. Refer to Table for CLKOP Divider values.

CLKOK Divider

The CLKOK divider feeds the global clock net. It divides the CLKOP signal of the PLL by the value of the divider. It can be set to values of 2, 4, 6,....126,128.

PLL Inputs and Outputs

CLKI Input

The CLKI signal is the reference clock for the PLL. It must conform to the specifications in the MachXO Family Data Sheet in order for the PLL to operate correctly. The CLKI can be derived from a dedicated dual-purpose pin or from routing.

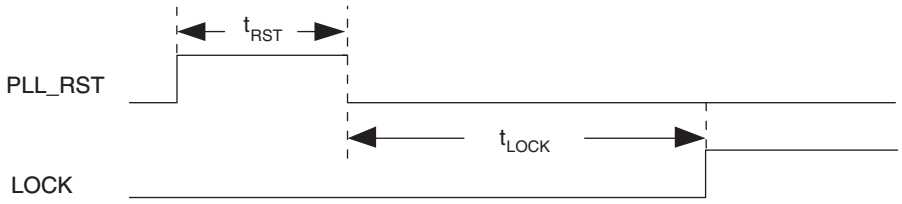
RST Input

The PLL reset occurs under two conditions. At power-up an internal power-up reset signal from the configuration block resets the PLL. The user controlled PLL reset signal RST is provided as part of the PLL module that can be driven by an internally generated reset function or a pin. This RST signal resets all internal PLL counters. When RST goes inactive, the PLL will start the lock-in process, and will take the t_{LOCK} time to complete the PLL lock.

Note: The use of RST is mandatory. RST must be asserted to start the PLL locking process or to re-start the locking process after losing lock.

Figure 10-3 shows the timing diagram of the RST input.

Figure 10-3. RST Input Timing Diagram



CLKFB Input

The feedback signal to the PLL, which is fed through the feedback divider, can be derived from the Primary Clock net (CLKOP), a dedicated dual-purpose pin, directly from the CLKOP divider (CLKINTFB) or from general routing. External feedback allows the designer to compensate for board-level clock alignment.

CLKOP Output

The sysCLOCK PLL main clock output, CLKOP, is a signal available for selection as a primary clock.

CLKOS Output with Phase and Duty Cycle Select

The sysCLOCK PLL auxiliary clock output, CLKOS, is a signal available for selection as a primary clock. The CLKOS is used when phase shift and/or duty cycle adjustment is desired. The programmable phase shift allows for different phase in increments of 45° to 315°. The duty select feature provides duty select in 1/8th of the clock period.

CLKOK Output with Lower Frequency

The CLKOK is used when a lower frequency is desired. It is a signal available for selection as a primary clock.

Dynamic Delay Control I/O Ports

Refer to Table 10-4 for detailed information.

LOCK Output

The LOCK output provides information about the status of the PLL. After the device is powered up and the input clock is valid, the PLL will achieve lock within the specified lock time. Once lock is achieved, the PLL lock signal will be asserted. If, during operation, the input clock or feedback signals to the PLL become invalid, the PLL will lose lock. However, when the input clock completely stops, the LOCK output will remain in its last state, since it is inter-

nally registered by this clock. It is recommended to assert PLL RST to re-synchronize the PLL to the reference clock. The LOCK signal is available to the FPGA routing to implement generation of RST. ModelSim® simulation models take two to four reference clock cycles from RST release to LOCK high.

PLL Attributes

The PLL utilizes several attributes that allow the configuration of the PLL through source constraints. This section details these attributes and their usage.

FIN
The input frequency can be any value within the specified frequency range based on the divider settings.

CLKI_DIV, CLKFB_DIV, CLKOP_DIV, CLKOK_DIV
These dividers determine the output frequencies of each output clock. The user is not allowed to input an invalid combination; determined by the input frequency, the dividers, and the PLL specifications.

FREQUENCY_PIN_CLKOP, FREQUENCY_PIN_CLKOK
These output clock frequencies determine the divider values.

FDEL
The FDEL attribute is used to pass the Delay Adjustment step associated with the Output Clock of the PLL. This allows the user to advance or retard the Output Clock by the step value passed multiplied by 250ps (nominal). The step ranges from -8 to +8 resulting the total delay range to +/- 2ns.

PHASEADJ
The PHASEADJ attribute is used to select Coarse Phase Shift for CLKOS output. The phase adjustment is programmable in 45° increments.

DUTY
The DUTY attribute is used to select the Duty Cycle for CLKOS output. The Duty Cycle is programmable at 1/8 of the period increment.

FB_MODE
There are three sources of feedback signals that can drive the CLKFB Divider: Internal, CLKOP (Clock Tree) and User Clock. CLKOP (Clock Tree) feedback is used by default. Internal feedback takes the CLKOP output at CLKOP Divider output before the Clock Tree to minimize the feedback path delay. The User Clock feedback is driven from the dedicated pin, clock pin or user specified internal logic.

DELAY_CNTL
This attribute is designed to select the Delay Adjustment mode. If the attribute is set to “DYNAMIC” the delay control switches between Dynamic and Static depending upon the input logic of DDAMODE pin. If the attribute is set to “STATIC”, Dynamic Delay inputs are ignored in this mode.

CLKOK Output with Lower Frequency
The CLKOK is used when a lower frequency is desired. It is a signal available for selection as a primary clock.

MachXO PLL Primitive Definitions

A PLL primitive is used for MachXO PLL implementation. The definitions of the PLL I/O ports are shown in Table 10-2. Figure 10-4 shows the MachXO PLL primitive library symbol. The EHXPLLC includes all features available in the PLL including Dynamic Delay Adjustment.

Figure 10-4. MACHXO PLL Primitive Symbols

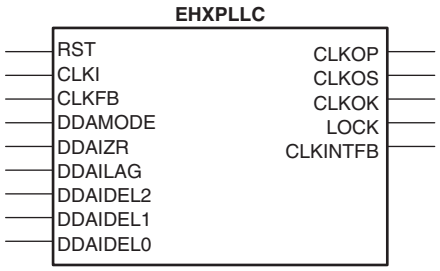


Table 10-2. MachXO PLL I/O Definitions

Signal	I/O	Description
CLKI	I	General routing or dedicated global clock input pad.
CLKFB	I	From general routing, clock tree, internal feedback from CLKOP or dedicated external feedback pad.
RST	I	"1" to reset PLL counters.
CLKOP	O	PLL output clock to clock tree (no phase shift).
CLKOS	O	PLL output clock to clock tree (phase shifted/duty cycle changed).
CLKOK	O	PLL output to clock tree (CLKOK divider, low speed, output).
LOCK	O	"1" indicates PLL LOCK to CLKI, asynchronous signal.
CLKINTFB	O	Internal feedback source. CLKOP divider output before CLOCK TREE.
DDAMODE	I	DDA Mode. "1" Pin control (dynamic), "0": Fuse Control (static).
DDAIZR	I	DDA Delay Zero. "1": delay = 0, "0": delay = on.
DDAILAG	I	DDA Lag/Lead. "1": Lead, "0": Lag.
DDAIDEL[2:0}	I	DDA delay.

PLL Attributes Definitions

The MachXO PLL utilizes several attributes that allow the configuration of the PLL through source constraints. This section details these attributes and their usage.

Table 10-3. MachXO PLL User Attributes

Attributes	MM GUI Access	Attribute Name	Preference Editor Support	Value	Default Value
CLKI Frequency (MHz)	Y	FIN	N	Note 6	100
CLKI Frequency (MHz)	Y	FREQUENCY_PIN_CLKI	N	Note 6	100
CLKOP Frequency (MHz)	Y	FREQUENCY_PIN_CLKOP	N	Note 6	100
CLKOK Frequency (MHz)	Y	FREQUENCY_PIN_CLKOK	N	Note 6	50
CLKOP Frequency Tolerance (%)	Y		N	0.0,0.1,0.2,0.5,1.0,2.0,5.0,10.0	0.0
CLKI Divider Setting	Y	CLKI_DIV[4]	N	1 to 16	1
CLKFB Divider Setting	Y	CLKFB_DIV	N	1 to 16	1
CLKOP Divider Setting	Y	CLKOP_DIV	N	Note 3	8 (Note 2)
CLKOK Divider Setting	Y	CLKOK_DIV	N	2,4,6,..,126,128	2
Fine Delay Adjust	N	FDEL[7]	Y	-8 to 8	0
Coarse Phase Shift Selection (O)	Y	PHASEADJ	N	0, 45, 90...315	0

Table 10-3. MachXO PLL User Attributes (Continued)

Attributes	MM GUI Access	Attribute Name	Preference Editor Support	Value	Default Value
Duty Cycle Selection (1/8 Increment)	Y	DUTY	N	1 to 7	4
Delay Control	Y	DELAY_CNTL[1]	N	DYNAMIC/STATIC	STATIC
Feedback Mode	Y	Note 5	N	INTERNAL/CLKOP/UserClock	CLKOP
CLKOS Select	Y		N		
CLKOK Select	Y		N		

1. DYNAMIC This mode switches delay control between Dynamic and Static depending upon the input logic of DDAMODE pin.
   STATIC This mode sets the delay control to Static Control.
2. CLKOP_DIV value is calculated to maximize the f_{VCO} within the specified range based on FIN, CLKOP_FREQ in conjunction with CLKI_DIV and CLKFB_DIV values.
3. The CLKOP Divider values are 2, 4, 6, 8,…30, 32 if CLKOS is unused and 2, 4, 8, 16, 32 if CLKOS is used.
4. All divider settings are user transparent in Frequency Mode. These are user attributes in Divider Mode.
5. CLKFB source:
   INTERNAL: CLKINTFB (internal feedback path is used).
   CLKOP: Primary Clock net feedback node of CLKOP.
   User Clock:
   - General routing (includes FPGA logic or general I/O)
   - Primary Clock net (includes user connecting CLKOP to CLKFB internally to the chip, or the use of a device clock pin)
   - Dedicated PLL feedback pin
6. Refer to the MachXO Family Data Sheet for current specifications.
7. This attribute is not available in the IPexpress GUI. After reviewing the trace report file, users can determine the amount of delay that will best fit the clocking in their design. Further information on FDEL settings is described in the following section.

FDEL Settings

There are four ways the user can enter the desired FDEL value.

1. Although the FDEL entry is not available in the IPexpress GUI, the module generated by IPexpress includes the attribute with default value “0”. Users can replace it with a desired value.

   Example of source code with default FDEL value:

   ```
   attribute FDEL of ehxpll_mod_0_0 : label is "0";
   generic map (…
      FDEL=>"0",
      …
      ")
   ```

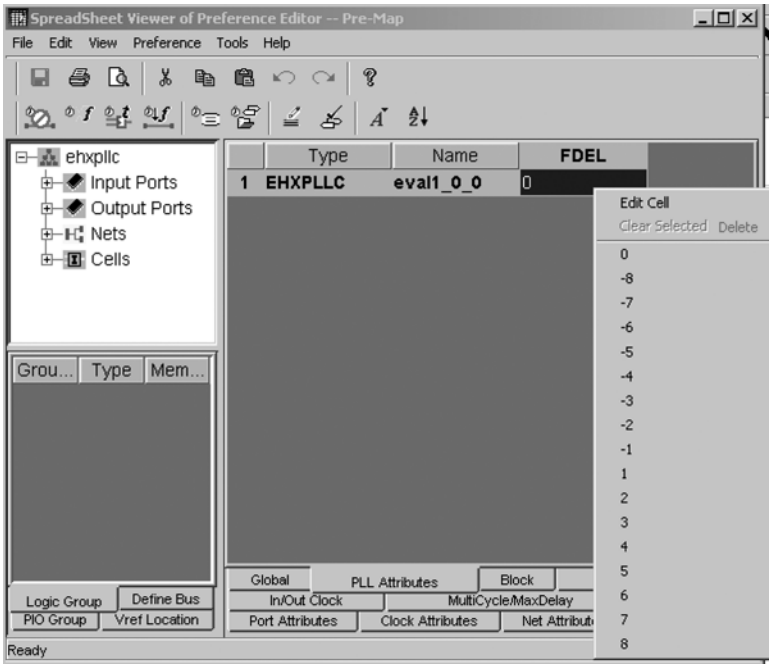
2. Preference File: User may specify the preference in the Preference file.

   Example:

   ```
   ASIC "FDEL_CODE_0_0" TYPE "EHXPLLB" FDEL="-2"  ;
   ```

3. Pre-Map Preference Editor: Users can enter the FDEL value in the Pre-Map Preference Editor as shown in Figure 10-5. Figure 10-5 shows the Pre-Map Preference Editor in the ispLEVER software. To see a similar screen shot for Lattice Diamond™ software, go to Appendix A, Figure 10-16.

Figure 10-5. Pre-Map Preference Editor



Dynamic Delay Adjustment

The Dynamic Delay Adjustment is controlled by the DDAMODE input. When the DDAMODE input is set to “1”, the delay control is done through the inputs, DDAIZR, DDAILAG and DDAIDEL(2:0). For this mode, the attribute “DELAY_CNTL” must be set to “DYNAMIC”. Table 10-4 shows the delay adjustment values based on the attribute/input settings.

In this mode, the PLL may come out of lock due to the abrupt change of phase. RST must be asserted to re-lock the PLL. Upon de-assertion of RST, the PLL will start the lock-in process and will take the t_{LOCK} time to complete the PLL lock.

Table 10-4. Delay Adjustment

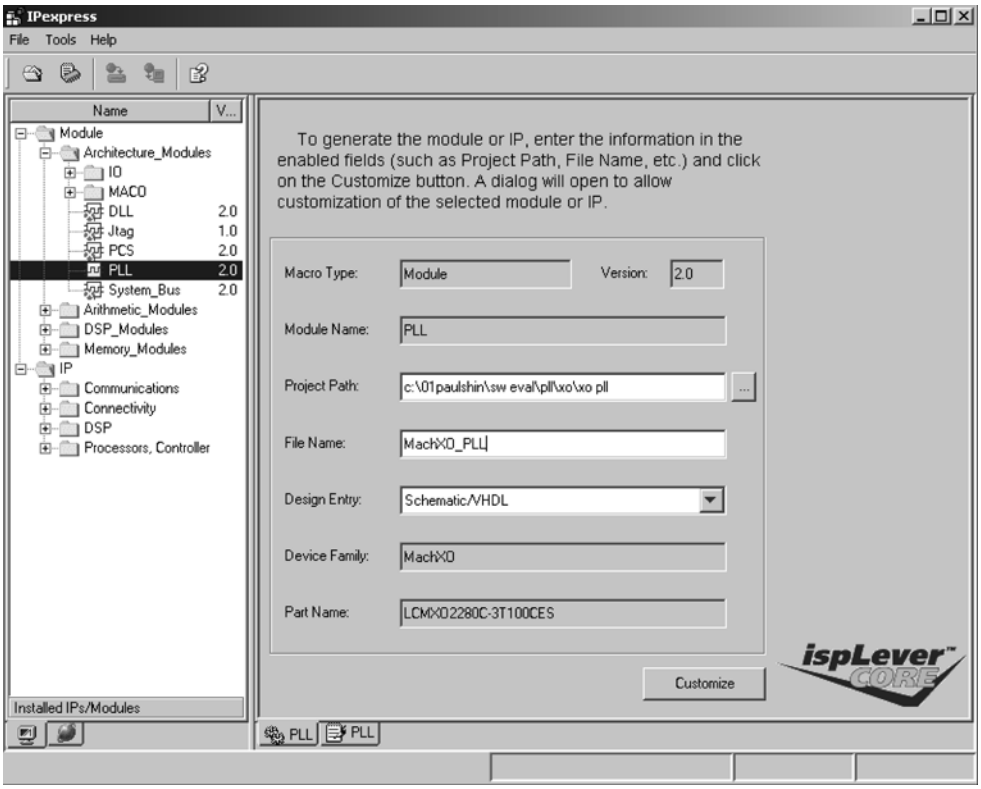
DDAMODE = 1: Dynamic Delay Adjustment			Delay 1 Tdly = 250 ps (Nominal)	DDAMODE = 0
DDAIZR	DDAILAG	DDAIDEL[2:0]		Equivalent FDEL Value
0	1	111	Lead 8 Tdly	-8
0	1	110	Lead 7 Tdly	-7
0	1	101	Lead 6 Tdly	-6
0	1	100	Lead 5 Tdly	-5
0	1	011	Lead 4 Tdly	-4
0	1	010	Lead 3 Tdly	-3
0	1	001	Lead 2 Tdly	-2
0	1	000	Lead 1 Tdly	-1
1	Don't Care	Don't Care	no delay	0
0	0	000	Lag 1 Tdly	1
0	0	001	Lag 2 Tdly	2
0	0	010	Lag 3 Tdly	3
0	0	011	Lag 4 Tdly	4
0	0	100	Lag 5 Tdly	5
0	0	101	Lag 6 Tdly	6
0	0	110	Lag 7 Tdly	7
0	0	111	Lag 8 Tdly	8

MachXO PLL Usage in IPexpress

The MachXO PLL is fully supported by IPexpress in ispLEVER and Diamond design software.

Figure 10-6 shows the main window when PLL is selected. To see screen shots of IPexpress in the Diamond software environment, see Appendix A, Figure 10-17. The only entry required in this window is the module name. Other entries are set to the project settings. The user may change these entries as desired. After entering a module name, click on Customize to open the Configuration Tab window, as shown in Figure 10-7.

Figure 10-6. IPexpress Main Window



Configuration Tab

The Configuration Tab lists all user accessible attributes with default values set. Upon completion of entries, click on Generate to generate source and constraint files. The user may choose to use the .lpc file to load parameters.

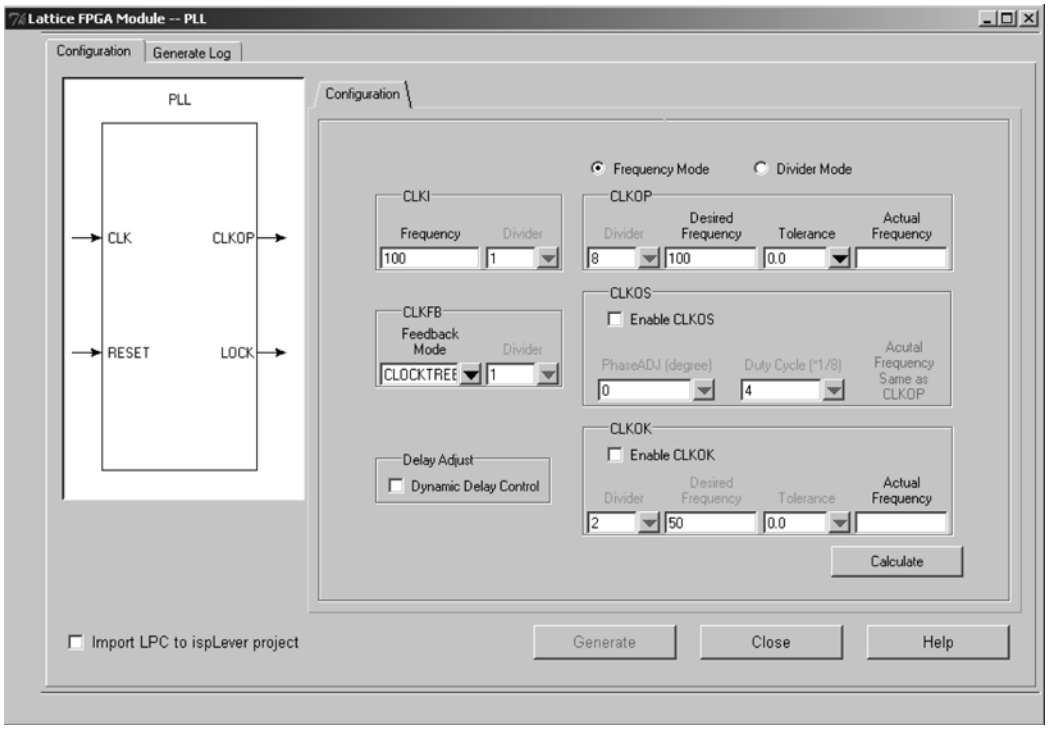
Mode

There are two modes for configuring the PLL in the Configuration Tab, Frequency Mode and Divider Mode.

Frequency Mode

In this mode, the user enters input and output clock frequencies and the software calculates the divider settings for user. If the output frequency the user entered is not achievable, the nearest frequency will be displayed in the Actual text box. After input and output frequencies are entered, clicking the Calculate button will display the divider values. Figure 10-7 shows the Frequency Mode configuration tab.

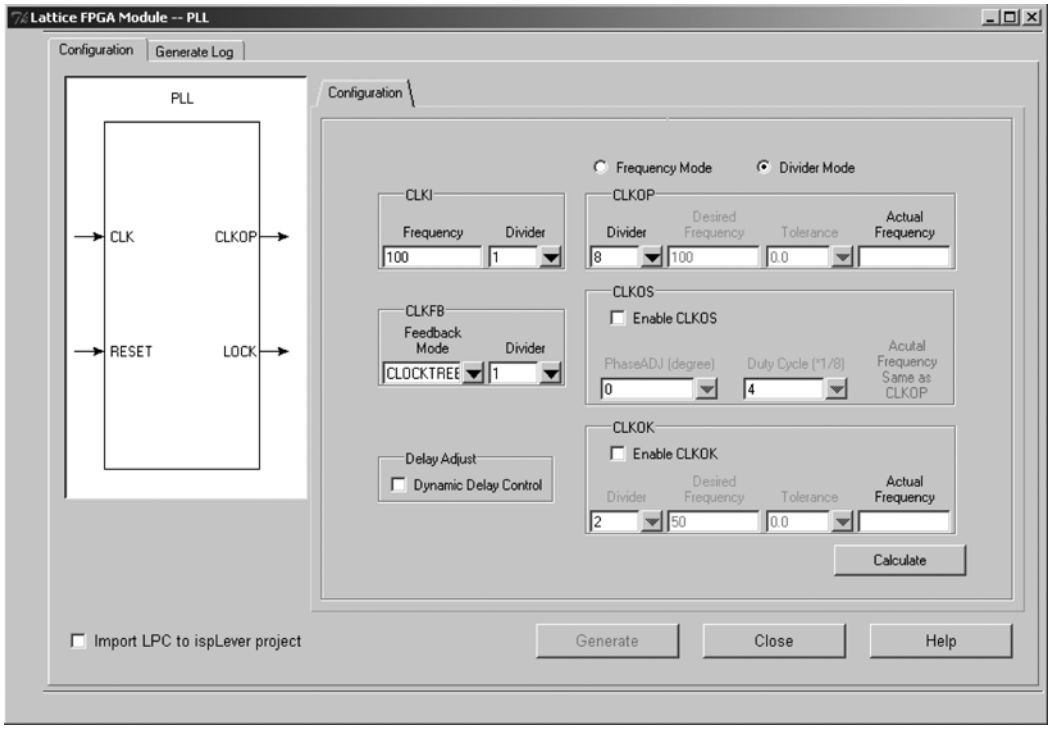
Figure 10-7. MachXO PLL Configuration Tab (Frequency Mode) Screen Capture (also see Diamond screen shot, Appendix A, Figure 10-18)



Divider Mode

In this mode, the user sets the divider settings with input frequency. The user must choose the CLKOP Divider value to maximize the f_{VCO} to achieve optimum PLL performance. After input frequency and divider settings are complete, clicking the Calculate button will display the frequencies. Figure 10-8 shows the Divider Mode Configuration tab.

Figure 10-8. MachXO PLL Configuration Tab (Divider Mode) - also see Diamond screen shot, Appendix A, Figure 10-19



Frequency Programming in Divider Mode for Advanced Users

Table 10-5. Frequency Limits

Parameter	Value
f_{IN}	Refer to the MachXO Family Data Sheet for frequency data.
f_{OUT}	Refer to the MachXO Family Data Sheet for frequency data.
f_{OUTK}	Refer to the MachXO Family Data Sheet for frequency data.
f_{VCO} (Hz)	Refer to the MachXO Family Data Sheet for frequency data.
CLKI Divider	1 to 16
CLKFB Divider	1 to 16
CLKOP Divider	2, 4, 8, 16, 32 (CLKOS used) 2, 4, 6, 8,... 28, 30, 32 (CLKOS not used)
CLKOK Divider	2, 4, 6, 8,..,126, 128
Maximum (N*V)	32
f_{PFD} (f_{IN}/M) (Hz)	Refer to the MachXO Family Data Sheet for frequency data.

Equations for generating Divider Settings and Output Frequency Ranges for Divider Mode Users
The divider names are abbreviated with legacy names as follows:

CLKI DIVIDER: M
CLKFB DIVIDER: N
CLKOP DIVIDER: V
CLKOK DIVIDER: K

f_{VCO} Constraint

From the loop:
$f_{OUT} = f_{IN} * (N/M)$ . . . . . (1)

From the loop:
$f_{VCO} = f_{OUT} * V$ . . . . . (2)

Substitute (1) in (2) yields:
$f_{VCO} = f_{IN} * (N/M) * V$ . . . . . (3)

Arrange (3):
$f_{IN} = (f_{VCO} /(V*N))*M$ . . . . . (4)

From equation (4):
$f_{INMIN} = ((f_{VCOMIN} /(V*N))*M$ . . . . . (5)
$f_{INMAX} = (f_{VCOMAX}/(V*N))*M$ . . . . . (6)

f_{PFD} Constraint

From the loop:
$f_{PFD} = f_{IN}/M$ . . . . . (7)
$f_{IN} = f_{PFD} *M$
$f_{INMIN} = f_{PFDMIN} *M = 25 * M$, (assume $f_{PFDMIN} = 25$) . . . . . (8)

So, equation (5) becomes:
$f_{INMIN} = ((f_{VCOMIN} /(V*N))*M$, if below 25 * M round up to 25 * M . . . . . (9)

From the loop:

$f_{INMAX} = f_{PFDMAX} *M = 420 * M$ . . . . . (10)
Assume f_{INMAX} =420

So, equation (6) becomes:
$f_{INMAX} = (f_{VCOMAX}/(V*N))*M$, if above 420 round down to 420 . . . . . (11)

From equation (1):
$f_{OUTMIN} = f_{INMIN}*(N/M)$, if below 25 * N round up to 25 * N . . . . . (12)
$f_{OUTMAX} = f_{INMAX}*(N/M)$, if above 420 round down to 420 . . . . . (13)

$f_{OUTKMIN} = f_{OUTMIN}/K$
$f_{OUTKMAX} = f_{OUTMAX}/K$

Example: When CLKOS is Not Used

Assume: f_{IN} = 40 MHz, M = 2, N = 3, V = 4

Then:
$f_{OUT} = 40 * 3/2 = 60$ . . . . . within range
$f_{VCO} = 60 * 4 = 240$ . . . . . out of range

f_{PFD} = 40 / 2 = 20 or 60 / 3 = 20 . . . . . .out of range

Assume M =1

Then:

f_{OUT} = 40*3/1 = 120. . . . . .within range
f_{VCO} = 120 * 4 = 480 . . . . . within range but not an optimum value
f_{PFD} = 40 /1 or 120 /3 = 40 . . . . . .within range

In this case, V = 6 will satisfy all conditions.

f_{VCO} = 120*6 = 720 < 840

Oscillator (OSCC)

There is a dedicated oscillator in the MachXO device whose output is made available for user.

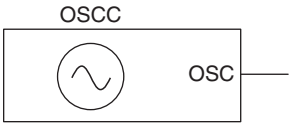
The oscillator frequency range is 18 to 26 MHz. The output of the oscillator can also be routed as an input clock to the clock tree. The oscillator frequency output can be further divided by internal logic (user logic) for lower frequencies, if desired. The oscillator is powered down when not in use.

Primitive Name: OSCC

Table 10-6. OSCC Port Definition

I/O	Name	Description
Output	OSC	Oscillator Clock Output

Figure 10-9. Oscillator Primitive Symbol (OSCC)



Oscillator Usage with VHDL - Example

```
COMPONENT OSCC
        PORT (OSC:OUT   std_logic);
END COMPONENT;

begin

OSCInst0: OSCC
        PORT MAP (
         OSC    =>  osc_int
         );
```

Clock/Control Distribution Network

The MachXO family provides global clocks: four primary clocks and four secondary clocks. These global signals are generated from four 16:1 muxes as shown in Figure 10-10 and Figure 10-11. The available clock sources for the MachXO256 and MachXO640 devices are four dual function clock pins and 12 internal routing signals. The available clock sources for the MachXO2280 devices are four dual function clock pins, six internal routing signals and six PLL outputs.

Dual function I/Os are provided for clocking usage. These I/Os are used as user programmable I/O pins when not in use as PLL or clock pins.

PCLK PIO (Primary Clock Pads)
There are two PCLK PIOs on top and two PIOs on bottom, for a total of four pins for each MachXO device. These pads connect directly to the global clock network.

PLL PIO
There are two pad pairs (one pad pair on the upper side and one pair on the lower side) for the MachXO2280 and one pad pair for the MachXO1200.

PLLFB PIO
Two pad pairs (one pad pair on the upper side and one pair on the lower side) for the MachXO2280 and one pad pair for the MachXO1200.

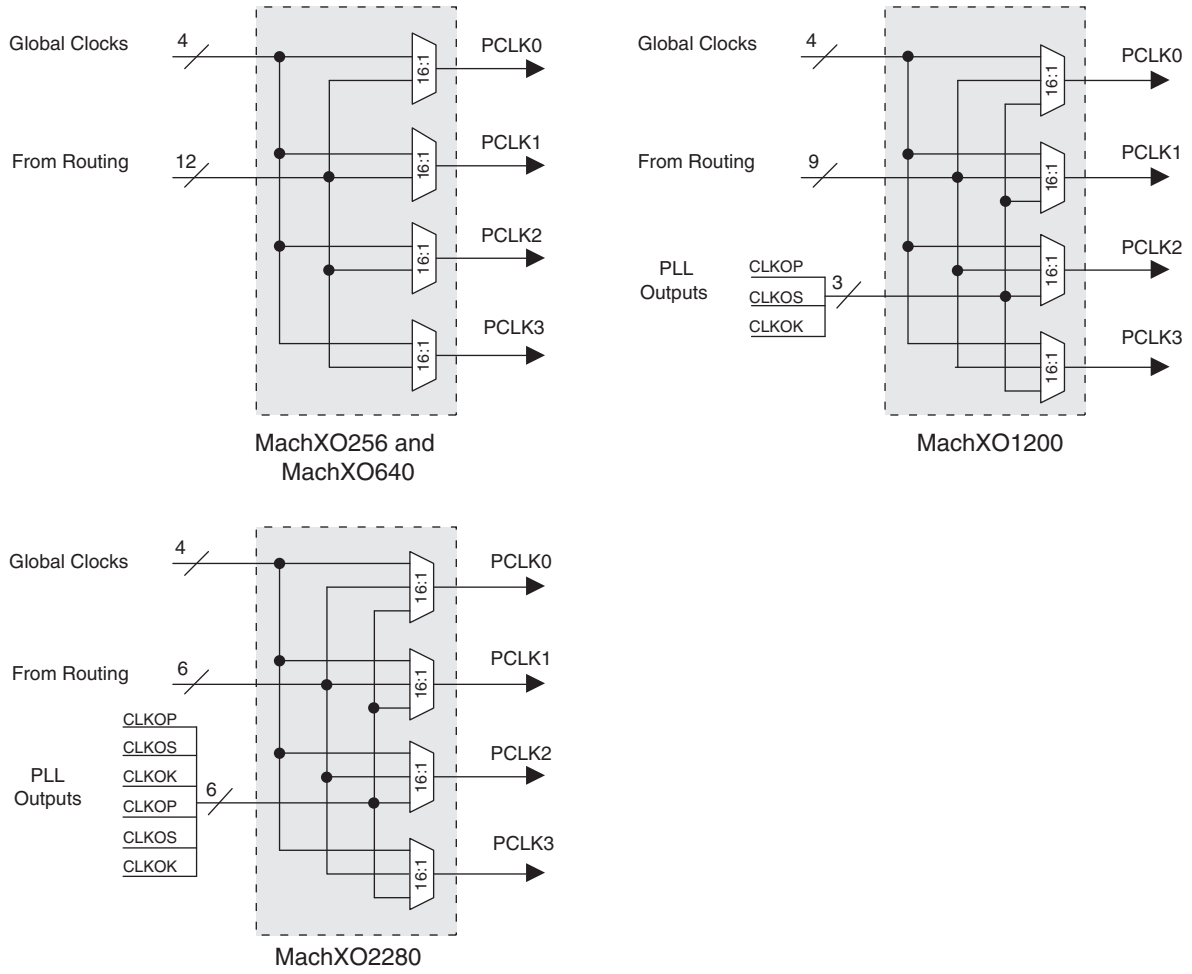
Primary Clock Mux Connectivity

The Primary Clock Mux input sources include:

- Primary clock input pins.
- PLL outputs
- From routing

The Primary Clock Mux outputs feed four clock input switch boxes in a PFU.

Figure 10-10. Primary Clock Net



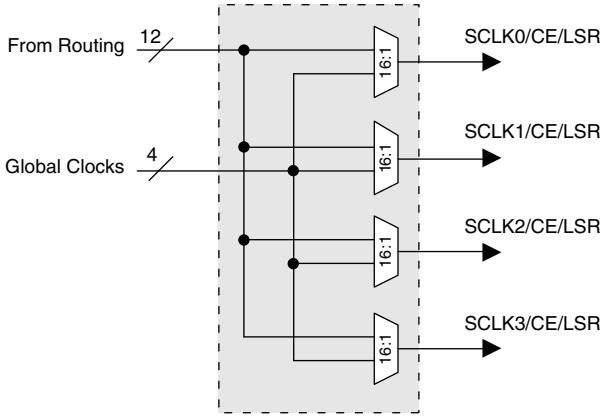
Secondary Clock/CE/LSR Mux Connectivity

The Secondary Clock/CE/LSR Mux input sources include:

- Primary Clock input pins
- From routing

The Secondary Clock/CE/LSR Mux outputs feed four clock input switch boxes and eight control input switch boxes in each PFU. Each slice includes one clock input switch box and two control input switch boxes, one for CE (Clock Enable) and the other for LSR (Local Set/Reset).

Figure 10-11. Secondary Clock/Control Net (All MachXO)



Primary Clock and Secondary Clock/CE/LSR Distribution Network

The Clock Input Switch Box and Control Input Switch Box are described in Figure 10-12 and Figure 10-13. Figure 10-14 shows the entire connectivity of the MachXO clock distribution network for each PFU.

Figure 10-12. Clock Switch Mux to Each PFU Slice

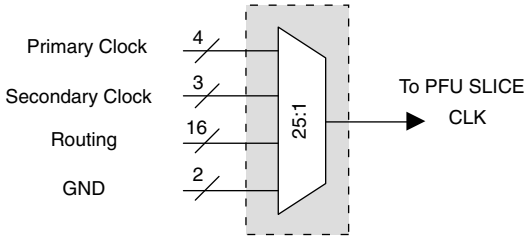


Figure 10-13. Control Switch Mux to Each PFU Slice

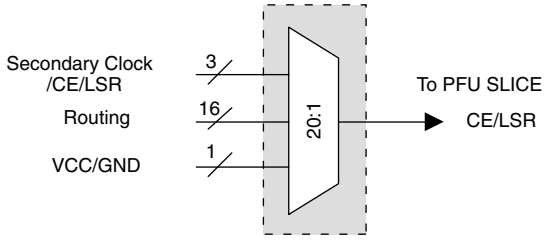
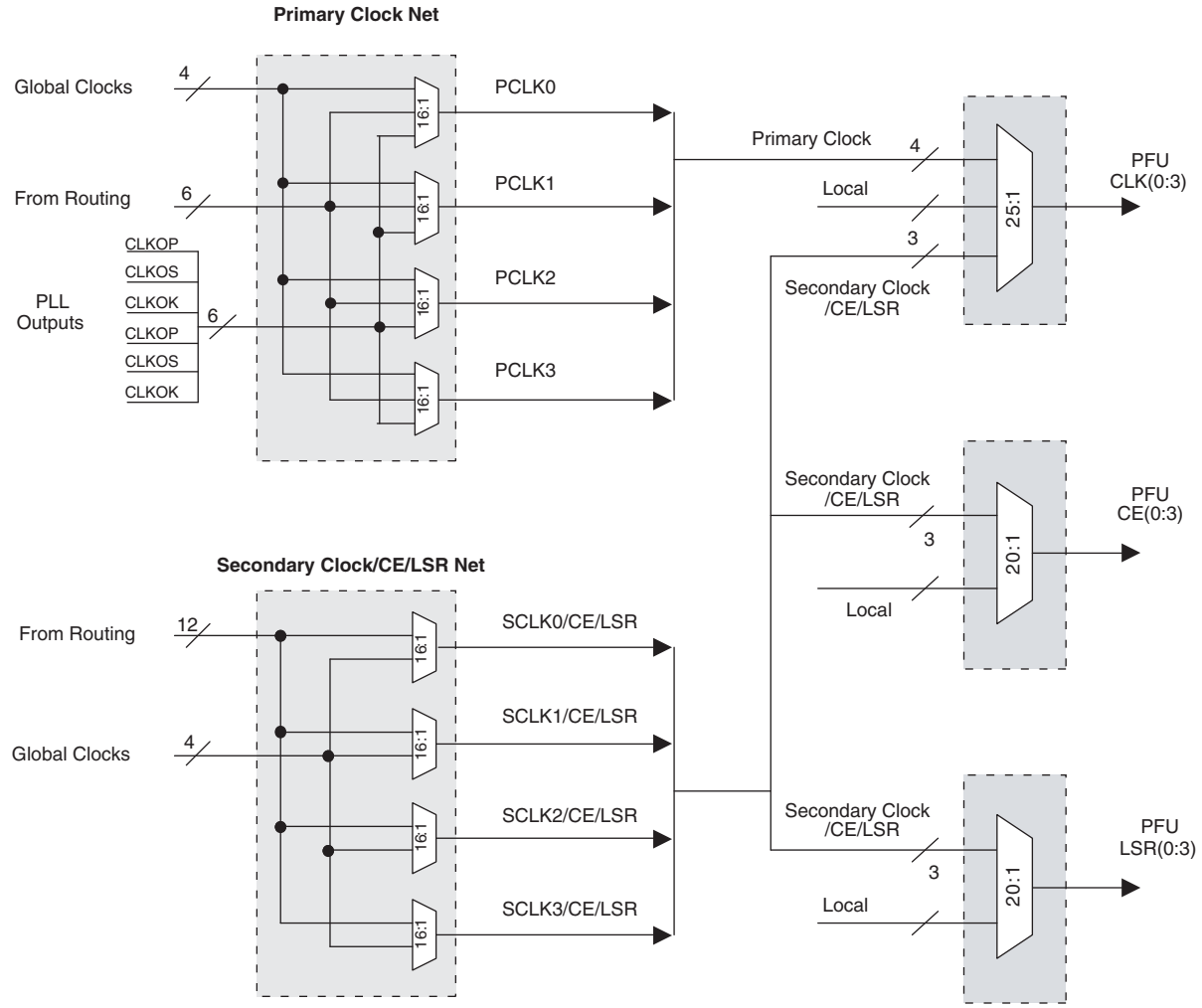


Figure 10-14. Primary Clock and Secondary Clock/CE/LSR Distribution



Maximum Number of Secondary Clocks Available

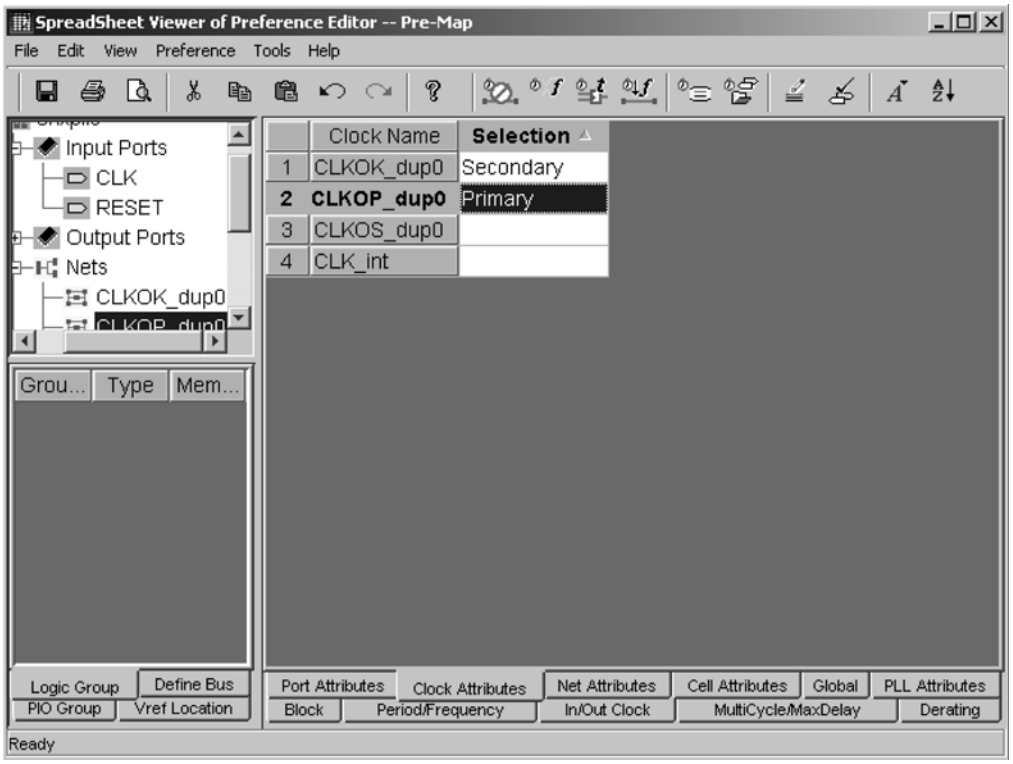
As illustrated in Figure Figure 10-14, there are four secondary clock nets in the clock distribution network but only three of them are reaching the clock input mux at PFU. This limits the number of total secondary clocks available to three.

Post Map Preference Editor Usage

Fine delay adjustment can be entered in the Post-Map Preference Editor after “Place and Run” to determine the required timing for the system design.

The Clock Preference assignments are also entered in the Post-Map Preference Editor. Figure 10-15 shows an example screen shot. To see a similar screen shot for Diamond design software, go to Appendix A, Figure 10-20.

Figure 10-15. Post-Map Preference Editor Example



PCB Layout Recommendations for V_{CCPLL} and GND_{PLL} if Separate Pins are Available

It is best to connect VCC_{PLL} to V_{CC} at a single point using a filter and to create a separate GND_{PLL} plane directly under it (tied via a single point to GND).

Separate islands for both VCC_{PLL} and GND_{PLL} are recommended if applicable.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
July 2005	01.0	Initial release.
October 2005	01.1	OSC frequency range changed.
		Total number of secondary clocks available is 3.
		ispLEVER GUI screen shot updated for version 5.1.
		References to MM/IP Manager replaced with IPexpress
		CLKOS/CLKOK select attributes added.
September 2006	01.2	Minor corrections.
February 2010	01.3	Reconciled LOCK description among MachXO, LatticeXP2, LatticeECP2/M and LatticeECP3.
October 2010	01.4	Updated for Lattice Diamond software support.

Appendix A. Lattice Diamond Design Software Screen Shots

Figure 10-16. Pre-Map Preference Editor in Diamond

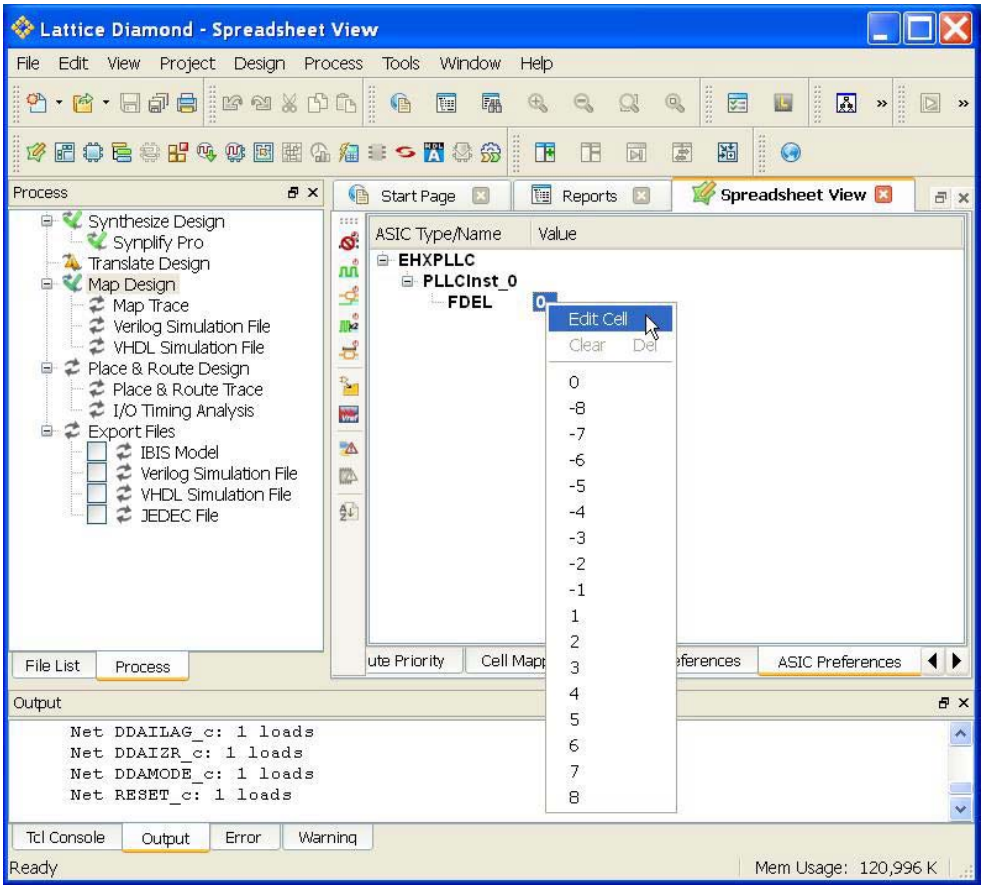


Figure 10-17. IPexpress Main Window in Diamond

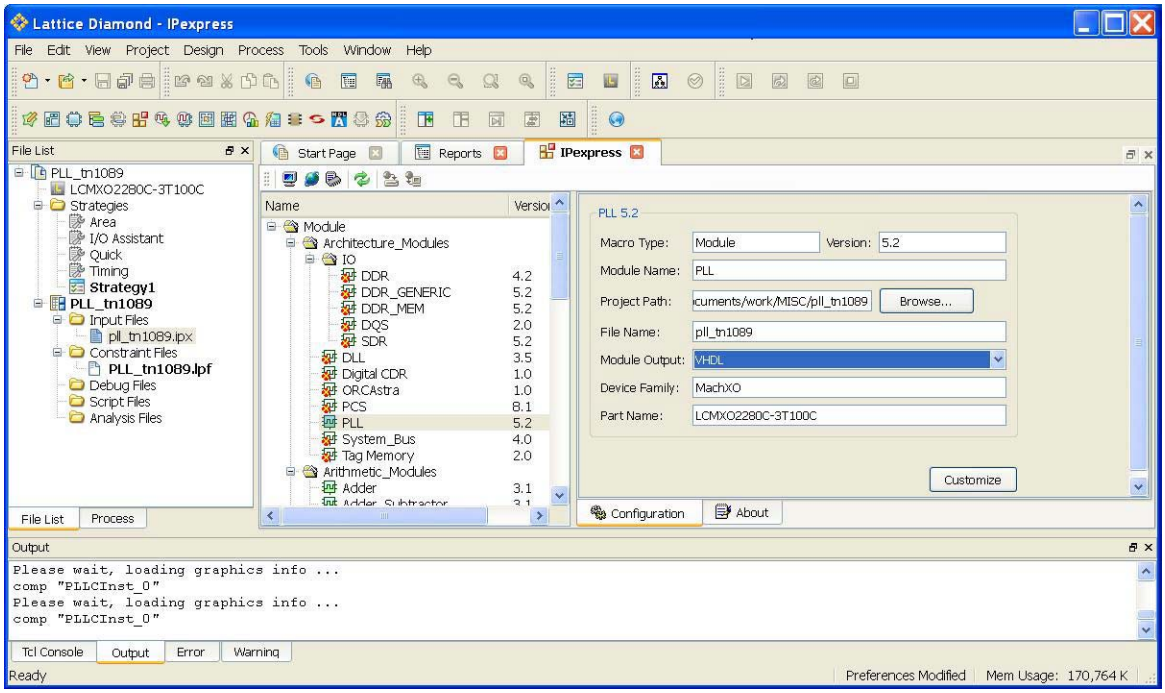


Figure 10-18. MachXO PLL Configuration Tab (Frequency Mode) in Diamond

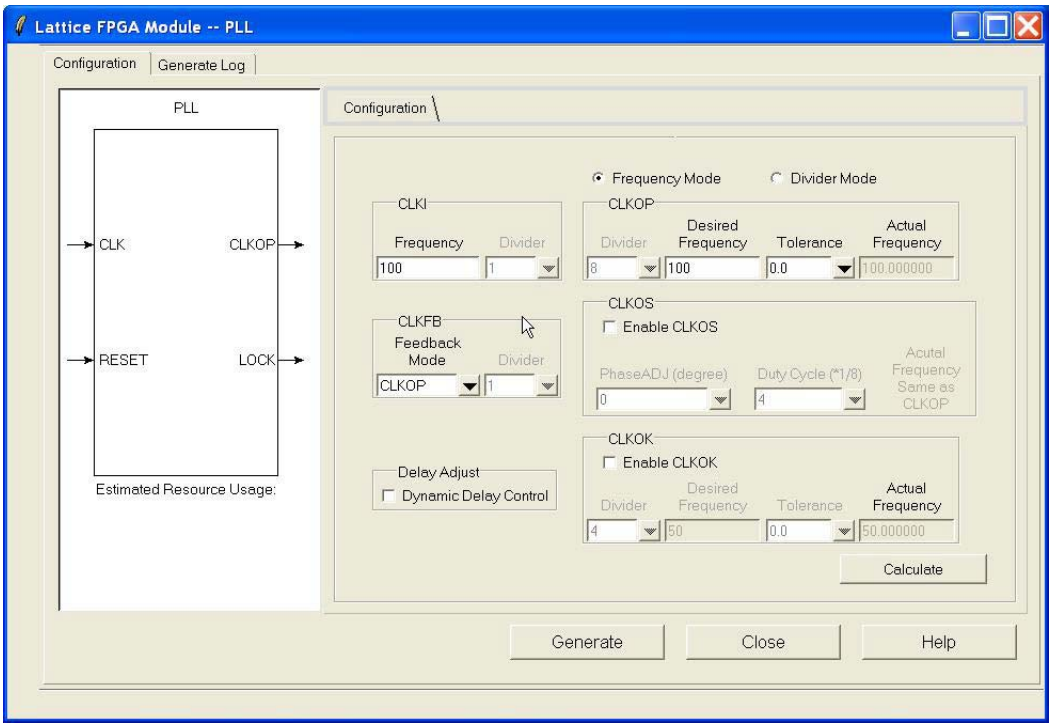


Figure 10-19. MachXO PLL Configuration Tab (Divider Mode) in Diamond

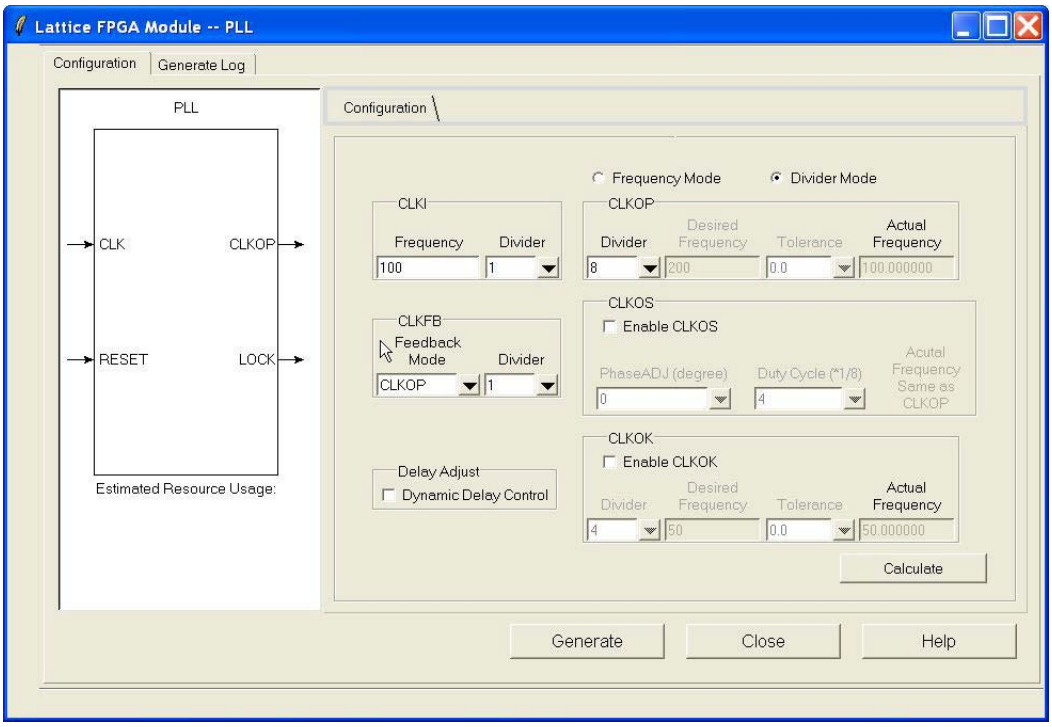
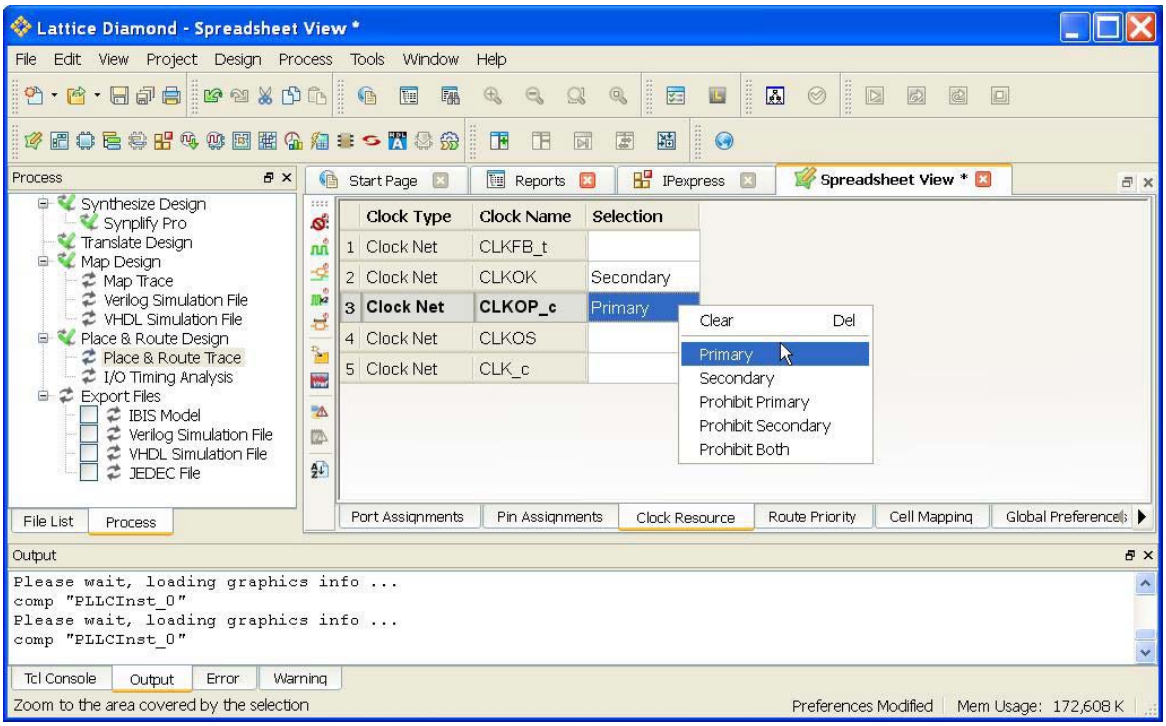


Figure 10-20. Post-Map Preference Editor Example in Diamond



Power Estimation and Management for MachXO Devices

September 2007 Technical Note TN1090

Introduction

One requirement for design engineers using programmable devices is to be able to calculate the power dissipation for a particular device used on a board. Lattice's ispLEVER® design tools include the Power Calculator tool which allows designers to calculate the power dissipation for a given device. This technical note provides users with details for using the Power Calculator to calculate the power consumption of the Lattice MachXO™ family of devices. General guidelines for reducing power consumption are also included.

Power Supply Sequencing and Hot Socketing

The MachXO devices have been designed to ensure predictable behavior during power-up and power-down. Power supplies can be sequenced in any order. During power-up and power-down sequences, the I/Os remain in tristate until the power supply voltage is high enough to ensure reliable operation. In addition, leakage into I/O pins is controlled to within specified limits listed in the MachXO Family Data Sheet. This allows for easy integration with the rest of the system. These capabilities make the MachXO ideal for many multiple power supply and hot-swap applications.

Recommended Power-up Sequence

As described in the previous paragraph, the supplies can be sequenced in any order. However, once internal power is achieved (determined by VCC, VCCAUX) the device releases I/Os from tristate and the management of I/Os becomes the designer's responsibility. Therefore, to simplify a system design it is recommended that supplies be sequenced VCCIO, VCC, VCCAUX. If VCCIO is tied to VCC or VCCAUX, it is recommended that VCCIO and the associated power supply are powered up before the remaining supply.

Please refer to the Hot Socketing section of the MachXO Family Data Sheet for more information.

Power Calculator Hardware Assumptions

Power consumption of the device can be broken down coarsely into the DC portion and the AC portion.

The Power Calculator reports the power dissipation in terms of:

1. DC portion of the power consumption
2. AC portion of the power consumption

The DC power (or the static power consumption) is the total power consumption of the used and the unused resources. These power components are fixed for each resource and depend upon the number of resource units utilized. The DC component also includes the static power dissipation for the unused resources of the device.

The AC portion of the power consumption is associated with the used resources and is the dynamic part of the power consumption. Its power dissipation is directly proportional to the frequency at which the resource is running and the number of resource units used.

Power Calculator

Power Calculator Equations

The power equations used in the Power Calculator are shown below.

Total DC Power (Resource)
= Total DC Power of Used Portion + Total DC Power of Unused Portion
= [DC Leakage per Resource when Used * $N_{RESOURCE}$]
+ [DC Leakage per Resource when Unused * ($N_{TOTAL\ RESOURCE}$ - $N_{RESOURCE}$)]

Where,

$N_{TOTAL\ RESOURCE}$	is the total number of resources in a device
$N_{RESOURCE}$	is the number of resources used in the design

The total DC power consumption for all the resources as per the design data is the Quiescent Power in the Power Calculator.

The AC Power, on the other hand, is governed by the following equation.

Total AC Power (Resource)
= $K_{RESOURCE}$ * f_{MAX} * $AF_{RESOURCE}$ * $N_{RESOURCE}$

Where,

$N_{TOTAL\ RESOURCE}$	is the total number of resources in a device,
$N_{RESOURCE}$	is the number of resources used in the design,
$K_{RESOURCE}$	is the power constant for the resource. The units of this factor are mW/MHz.
f_{MAX}	is the max. frequency at which the resource is running. Frequency is measured in MHz.
$AF_{RESOURCE}$	is the activity factor for the resource group. Activity factor is in terms of the percentage (%) of switching frequency.

Based on the above equations, if we wish to calculate the power consumption of the slice portion, it will be as shown below.

Total DC Power (Slice)
= Total DC Power of Used Portion + Total DC Power of Unused Portion
= [DC Leakage per Slice when Used * N_{SLICE}]
+ [DC Leakage per Slice when Unused * ($N_{TOTAL\ SLICE}$ - N_{SLICE})]

Total AC Power (Slice)
= K_{SLICE} * f_{MAX} * AF_{SLICE} * N_{SLICE}

Power Calculations

The Power Calculator is a powerful tool which allows users to estimate power consumption at three different levels:

1. Estimate of the utilized resources before completing place and route
2. Post place and route design
3. Post place and route, post trace, and post simulation

For first level estimation, the user provides estimates of device usage in the Power Calculator Wizard and the tool provides a rough estimate of the power consumption.

In the second level, a more accurate approach, the user imports the actual device utilization by importing the post place and route netlist (NCD) file.

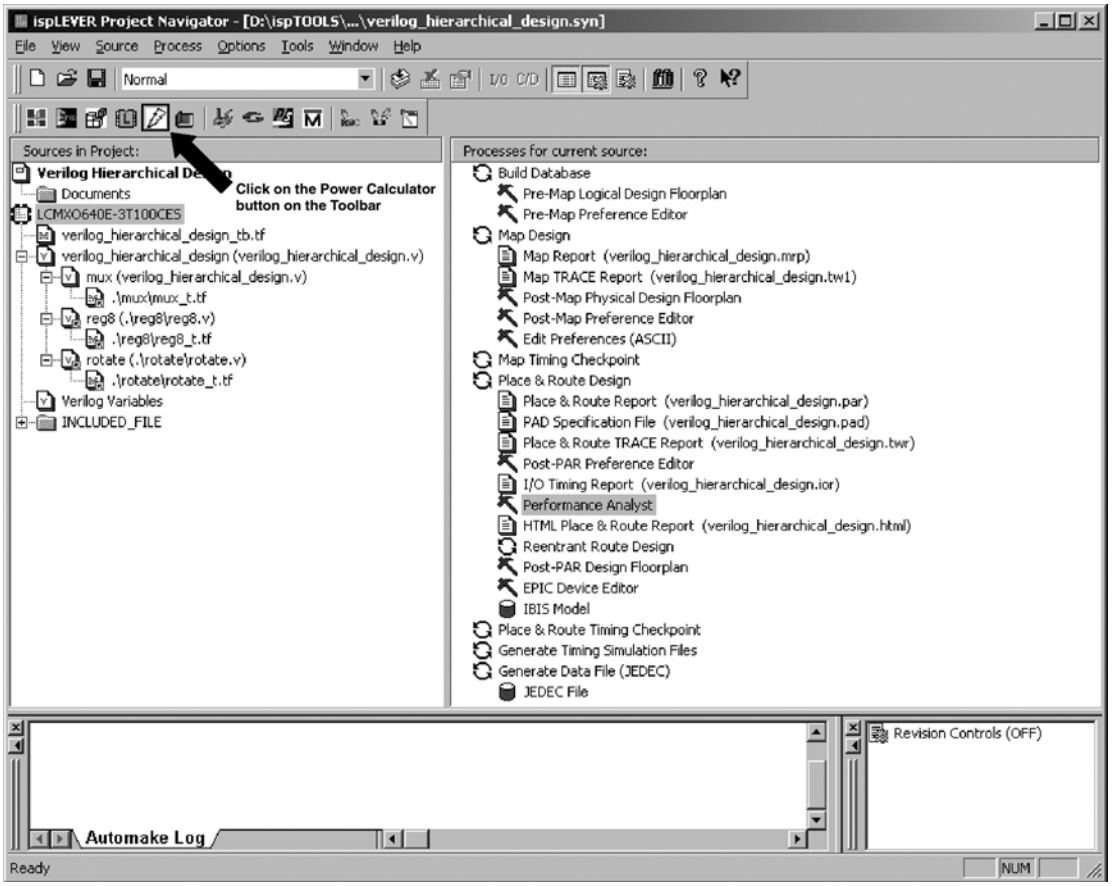
The third and final stage brings even more accuracy to the calculation, by importing the Trace (TWR) file to populate the maximum frequencies (f_{MAX}) into the tool. Users also have the option of importing the information from the post simulation (VCD) file and calculating the activity factor and toggle rates of various components.

These power calculations are discussed in detail in the following sections of this document.

Starting the Power Calculator

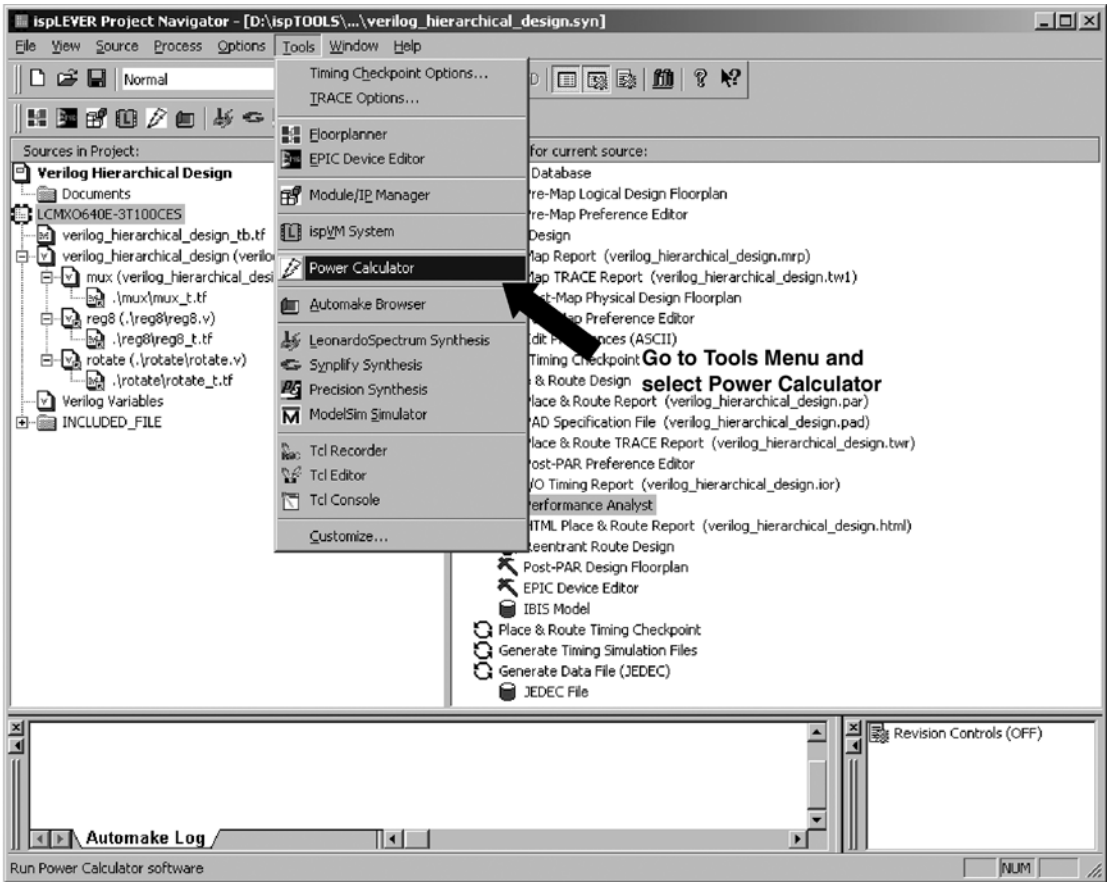
The user may launch the Power Calculator using one of two methods. The first method is by clicking the **Power Calculator button** in the toolbar as shown in Figure 11-1.

Figure 11-1. Starting Power Calculator From the Toolbar



Alternatively, users can launch the Power Calculator by going to the **Tools** menu and selecting **Power Calculator** as shown in Figure 11-2.

Figure 11-2. Starting Power Calculator from Tools Menu

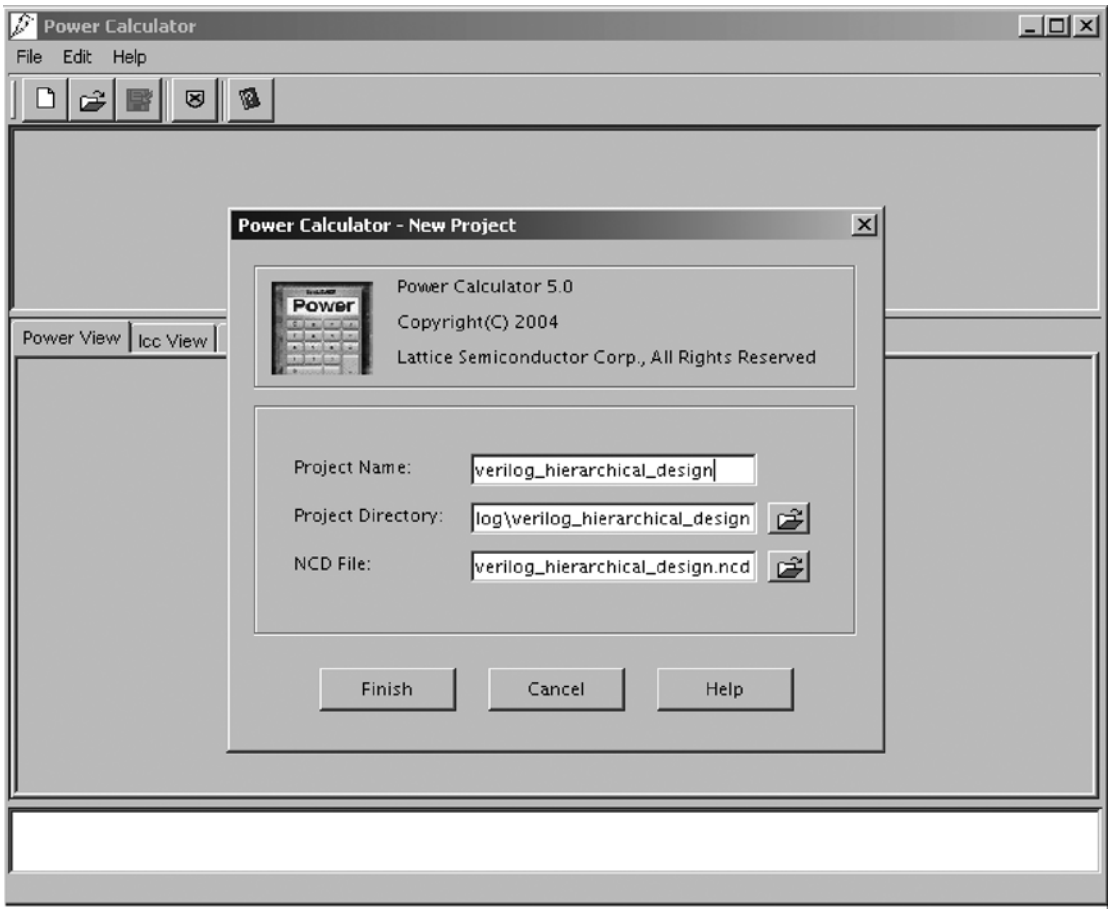


The Power Calculator does not support some of Lattice's older devices. The toolbar button and menu item is only present when supported devices are selected.

Creating a Power Calculator Project

After starting the Power Calculator, the user will see the Power Calculator window. Click on **File -> Menu** and select **New** to get to the Start Project window, as shown in Figure 11-3.

Figure 11-3. Power Calculator Start Project Window (Create New Project)



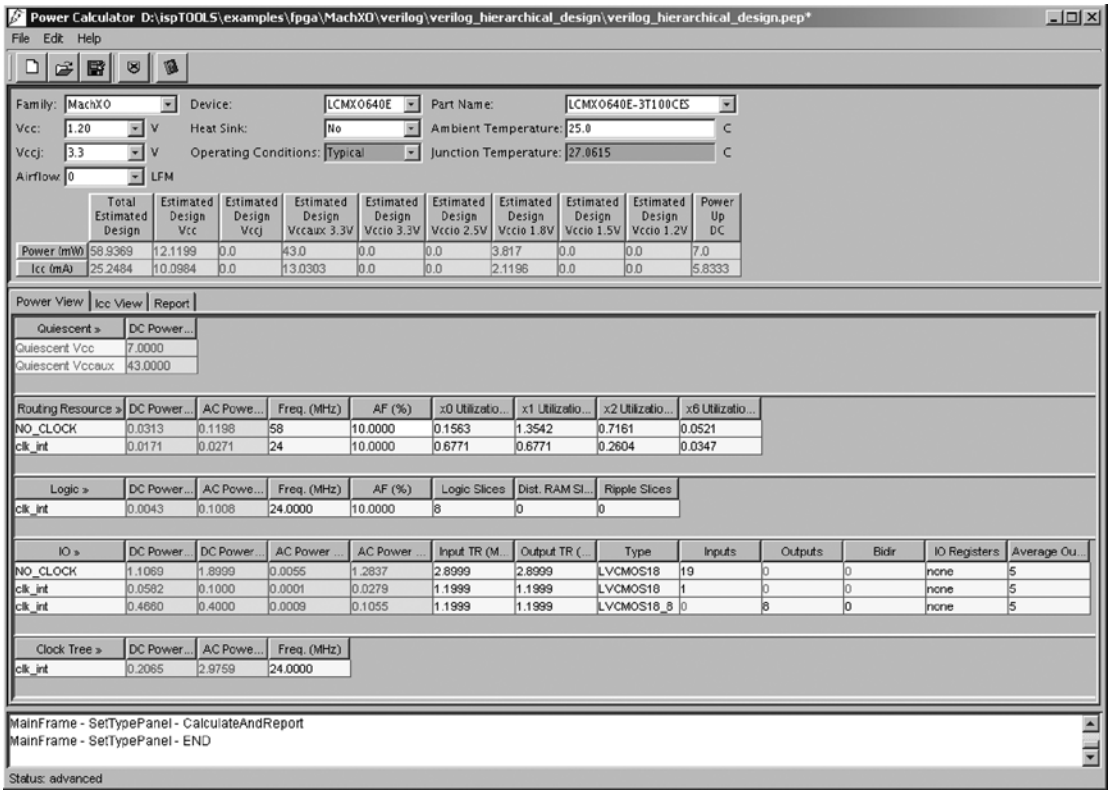
The Start Project window is used to create a new Power Calculator project (*.pep project). Three pieces of data must be input into the Start Project window:

1. The Power Calculator project name by default is the same as the Project Navigator project name. This can be changed, if desired.
2. The Project directory is where the Power Calculator project (*.pep) file will be stored. By default it is stored in the Main Project folder.
3. The NCD file is automatically selected if available or users can browse to the NCD file in a different location.

Power Calculator Main Window

The main Power Calculator window is shown in Figure 11-4.

Figure 11-4. Power Calculator Main Window (Type View)



The top pane of the window shows information about the device family, device and the part number as it appears in the Project Navigator. The V_{CC} used for the power calculation is also listed. Users have a choice of selecting the core voltage, V_{CC} with +5% of the nominal value (or values). The option of selecting V_{CCJ} is also available. Users provide the ambient temperature and the junction temperature is calculated based on that.

Users can also enter values for airflow in Linear Feet per Minute (LFM) along with heat sink to get the junction temperature.

A table in the top part of the Power Calculator summarizes the currents and power consumption associated with each type of power supply for the device. This also takes into consideration the I/O power supplies.

In the middle pane of the window, there are three tabs:

1. **Power View:** The first tab is the Power View. Under this tab, the Power Calculator tool has an interactive spreadsheet type interface with all values in terms of power consumption mW.

   The first column breaks down the design in terms of clock domains. The second and third columns, which are shaded blue in the tool, provide the DC (or static) and AC (or dynamic) power consumption, respectively.

   In case of the I/Os, there are four columns that are shaded blue. These provide the DC and AC power for I/Os for the core voltage, V_{CC} and the I/O voltage supply, $V_{CCIO.}$

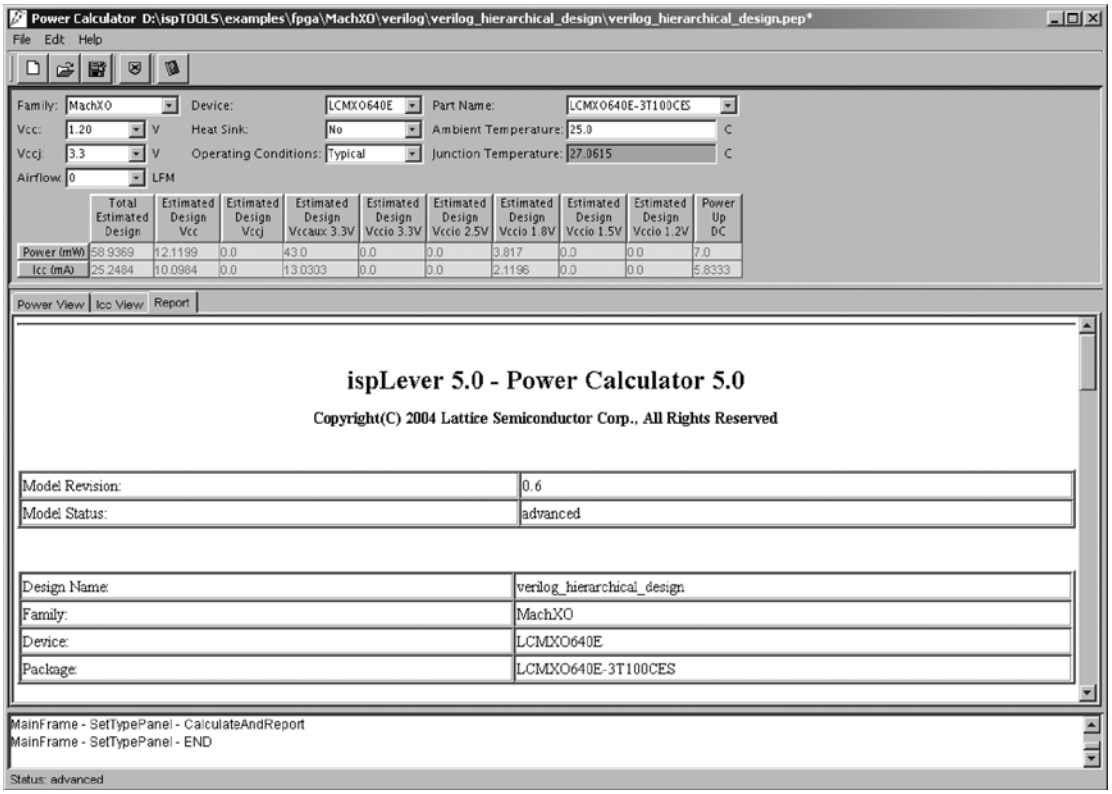
   The first three rows show the Quiescent Power for $V_{CC,}$ V_{CCAUX} and $V_{CCJ.}$ These are DC power numbers for a blank device or a device with no resource utilization.

Some of the cells are shaded yellow in the tool. These cells are editable cells and users can type in values such as frequency, activity factors and resource utilization. The second tab, the Report tab, is the summary of the Power View. This report is in text format and provides the details of the power consumption.

2. **I_{CC} View:** The second tab is the I_{CC} view. This tab is exactly same as the Power View tab, except that all values are in terms of current or mA.
3. **Report:** The third and final tab is the Report View. This is an HTML type of Power Calculator report. It summarizes the contents of the Power and I_{CC} views.

The final pane or the lower pane of the window is the log pane where users can see the log of the various operations in the Power Calculator.

Figure 11-5. Power Calculator Main Window (Power Report View)

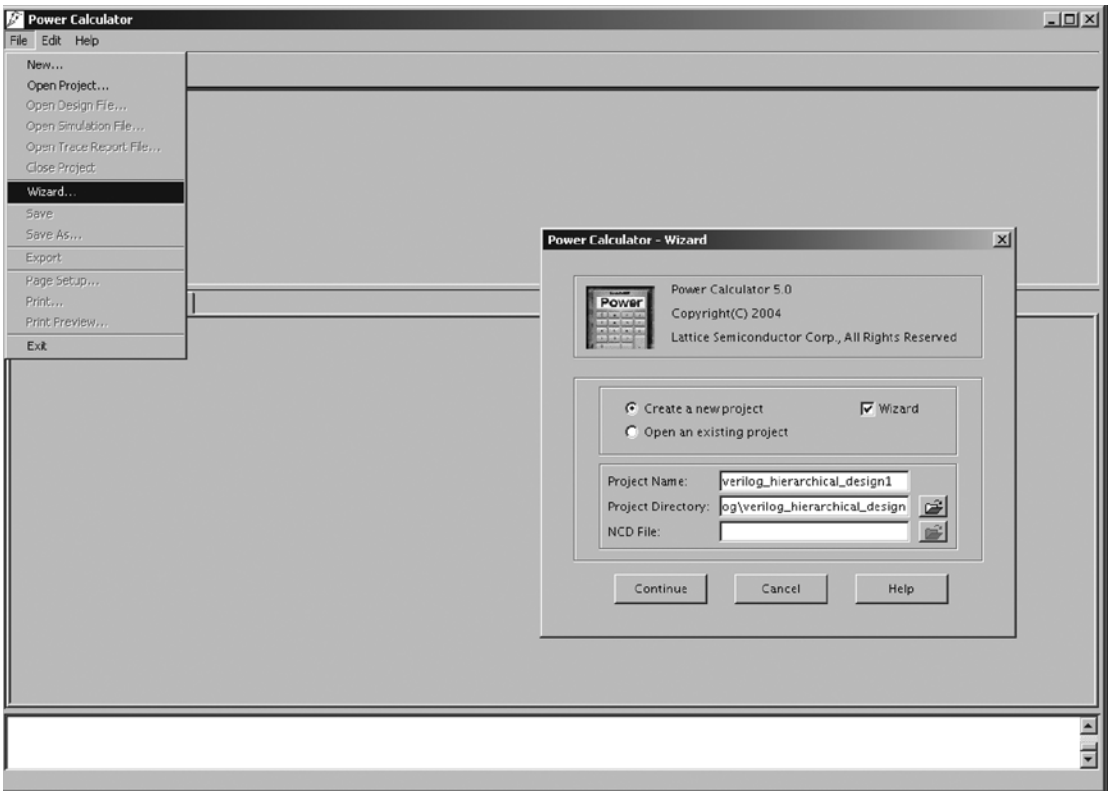


Power Calculator Wizard

The Power Calculator Wizard estimates the power consumption of a given design. This estimation is done before a design is created. The user must understand the logic requirements of the design. The wizard allows the user to provide these parameters and then estimates the power consumption of the device.

To start the Power Calculator in the Wizard mode, go to the **File** menu and select **Wizard**. Alternatively, click on the **Wizard** button to get the **Power Calculator - Wizard window** as shown in Figure 11-6. Select the option **Create a New Project** and check the **Wizard** check box in the **Power Calculator Start Project** window. Users are required to provide the project name and the project folder. Click **Continue**. Since this is power estimation before the actual design, no NCD file is required.

Figure 11-6. Power Calculator Start Project Window (Using the New Project Window Wizard)



The next screen allows users to select the device family, device and appropriate part number for which users plan to estimate power. After making the proper selections, click “Continue” (see Figure 11-7).

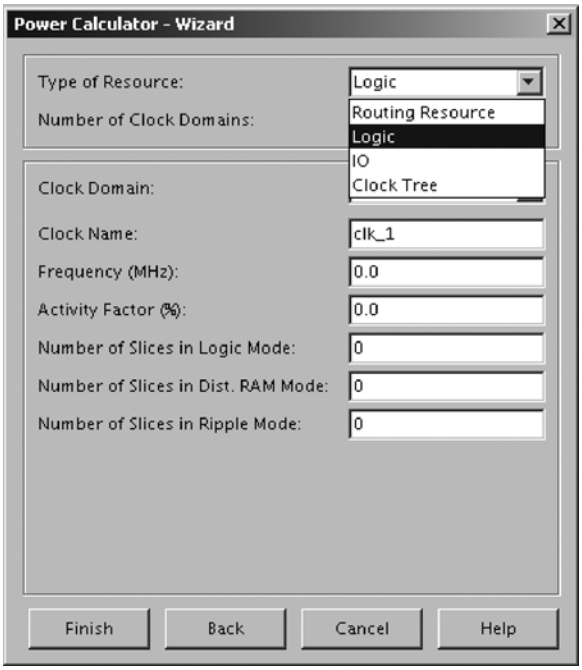
Figure 11-7. Power Calculator Wizard Mode Window - Device Selection

In the following screens, as shown in Figures 8-12, users can select further resources, like I/O types, clock name, frequency at which the clock is running, and other parameters, by selecting the appropriate resource using the pull-down menu:

1. Routing Resources
2. Logic
3. EBR
4. I/O
5. PLL
6. Clock Tree

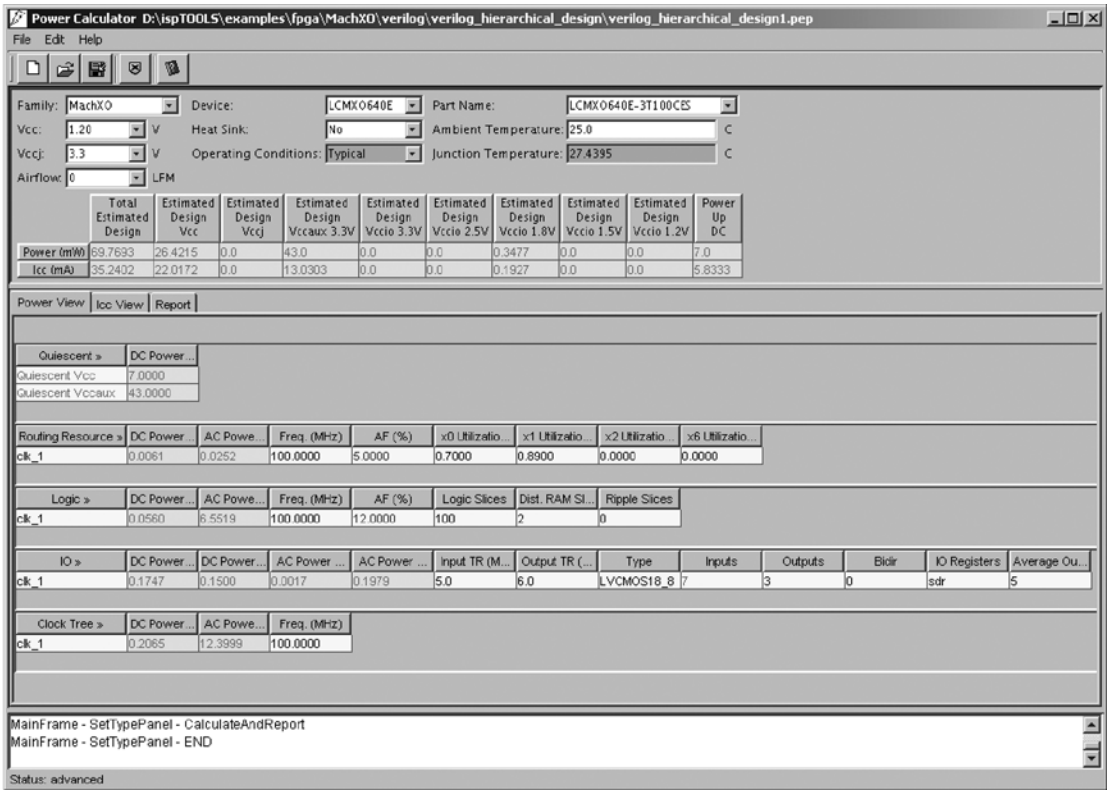
The numbers in these windows refer to the number of clocks and the index corresponds to each clock. By default, the clock names are clk_1, clk_2, and so on. Users can change the name of each clock by typing in the **Clock Name** text box. For each clock domain and resource, users can specify parameters such as frequency, activity factor, etc. In the final window, users must click the **Create** button for each clock-driven resource to include the parameters they have specified for it.

Figure 11-8. Power Calculator Wizard Mode Window - Resource Specification



These parameters are then used in the **Power Type View** window, which can be seen upon clicking on **Finish** (Figure 11-9).

Figure 11-9. Power Calculator Wizard Mode - Main Window



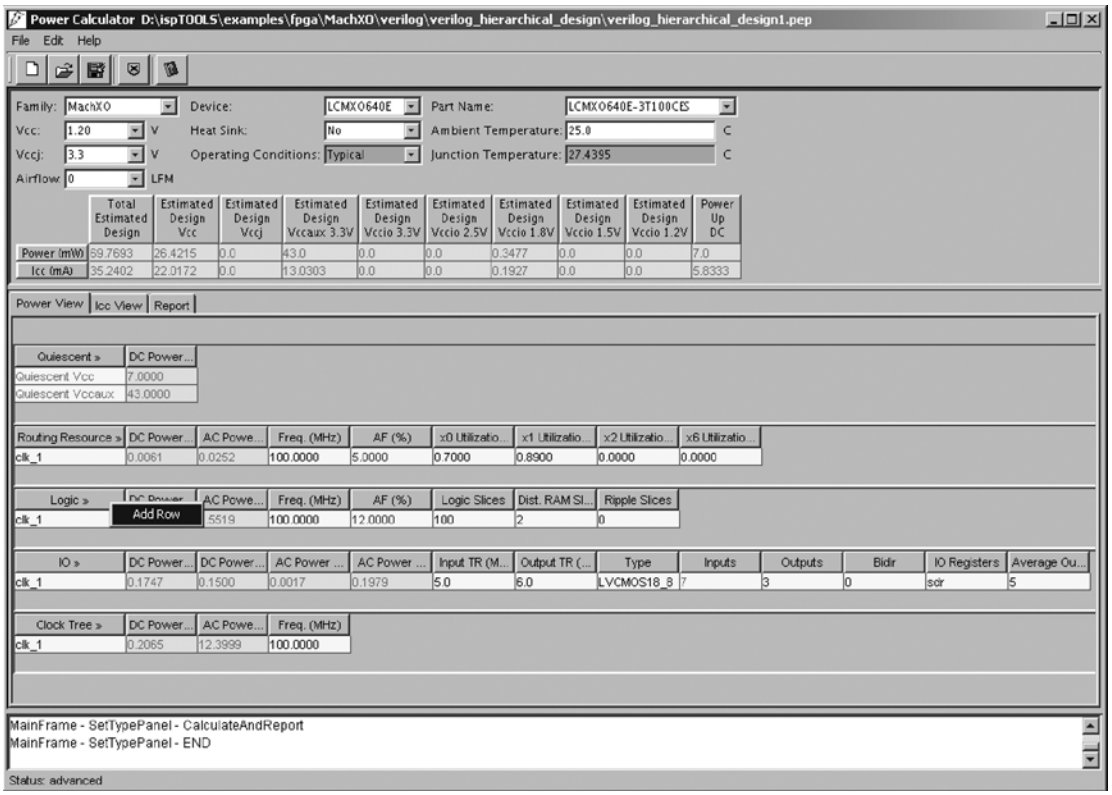
Power Calculator - Creating a New Project Without the NCD File

When starting a new project without the NCD file, begin either by using the **Wizard** (as discussed above) or by selecting the **Create a New Project** option in the **Power Calculator -> Start Project**. Users are required to provide a project name and project directory. After clicking **Continue** the Power Calculator main window will be displayed.

However, in this case there are no resources added. The power estimation row for the routing resources is always available in the Power Calculator. Users are required to add more information like the slice, EBR, I/O, PLL and clock tree utilization to calculate the power consumption.

For example, if the user wants to add the logic resources as shown in Figure 11-10, right-click on **Logic** and select **Add Row** in the pop-up menu.

Figure 11-10. Power Calculator Main Window - Adding Resources



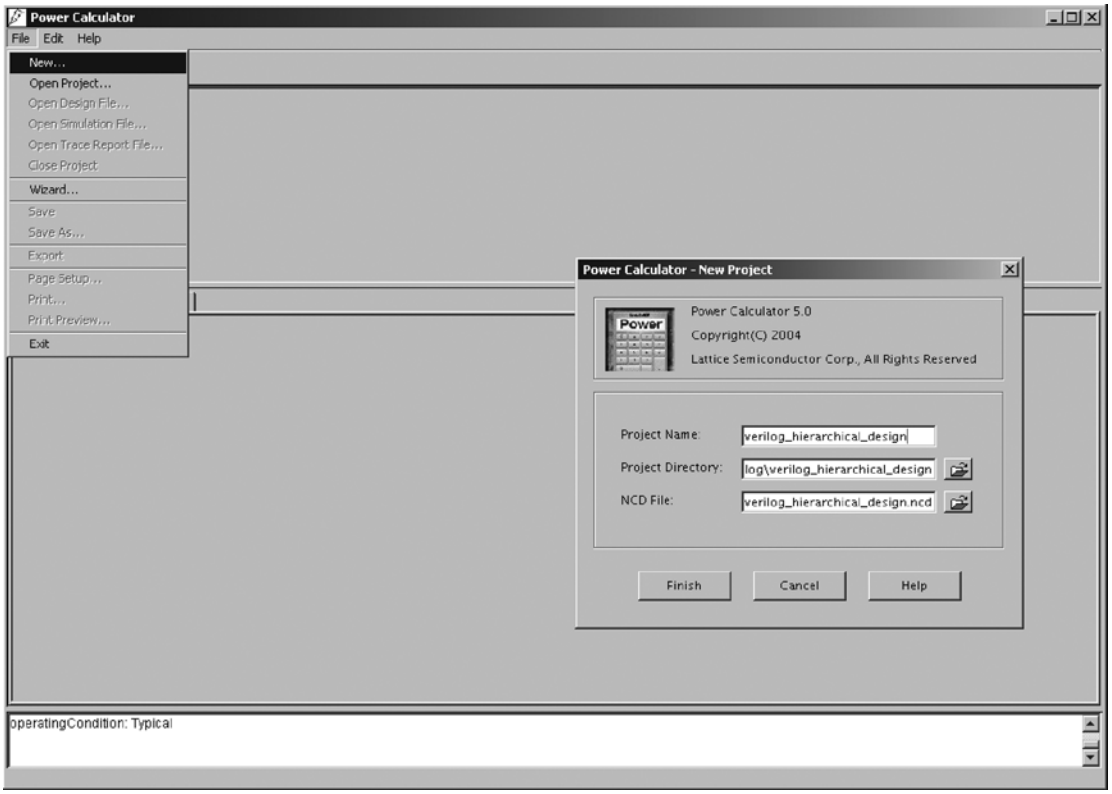
This adds a new row for the logic resource utilization with clock domain as clk_1.

Similarly, users can add other resources like EBR, I/Os, PLLs, routing, etc. Each of these resources is for the AC power estimation and categorized by clock domains.

Power Calculator - Creating a New Project with the NCD File

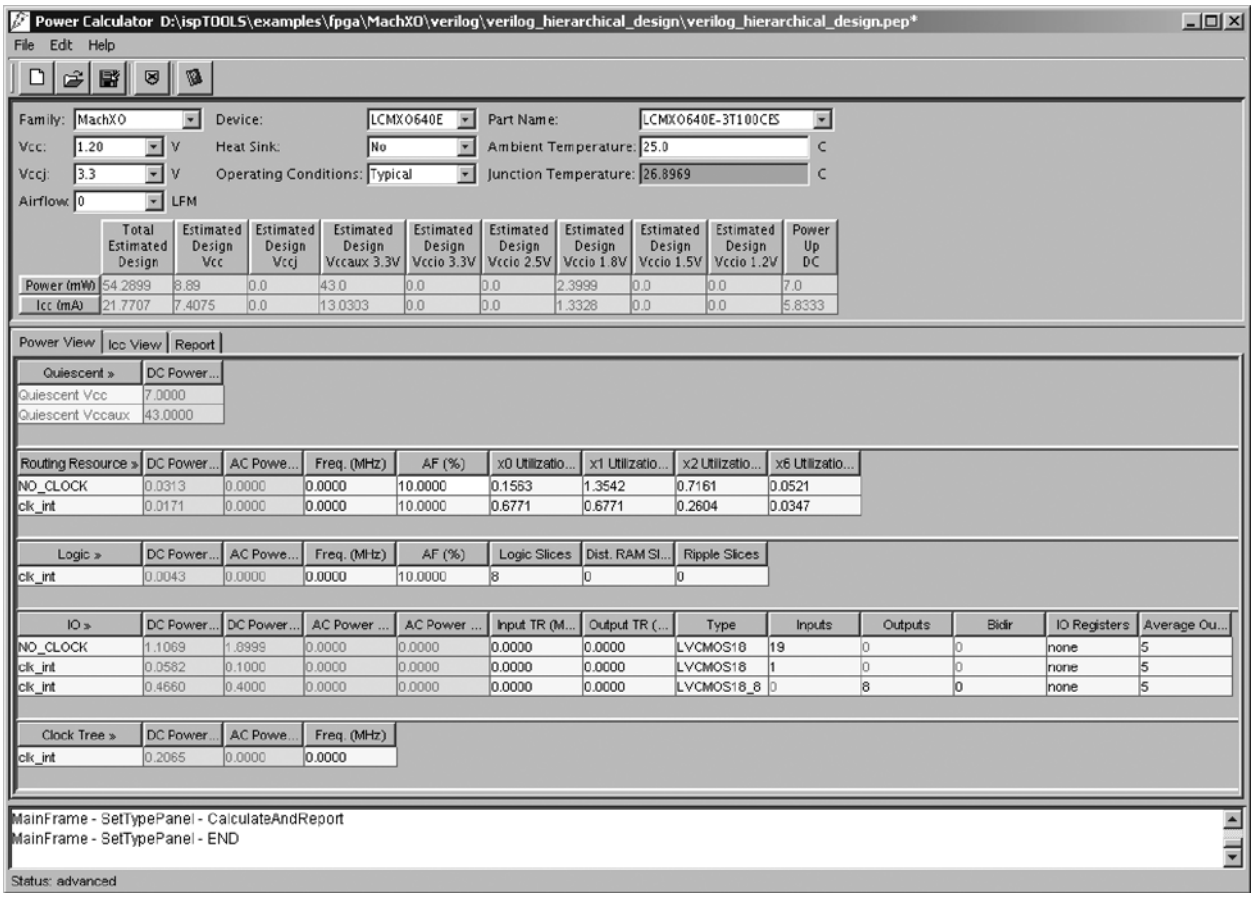
If the post place and routed NCD file is available, the Power Calculator can use this file to import accurate information about the design data and resource utilization and calculate the power. When the Power Calculator is started, the NCD file is automatically placed in the NCD File option, if it is available in the project directory. Otherwise, browse to the NCD file in the Power Calculator.

Figure 11-11. Power Calculator Start Project Window with Post PAR NCD File



The information from the NCD file is automatically inserted into the correct rows. Power Calculator uses the clock names from the design as shown in Figure 11-12.

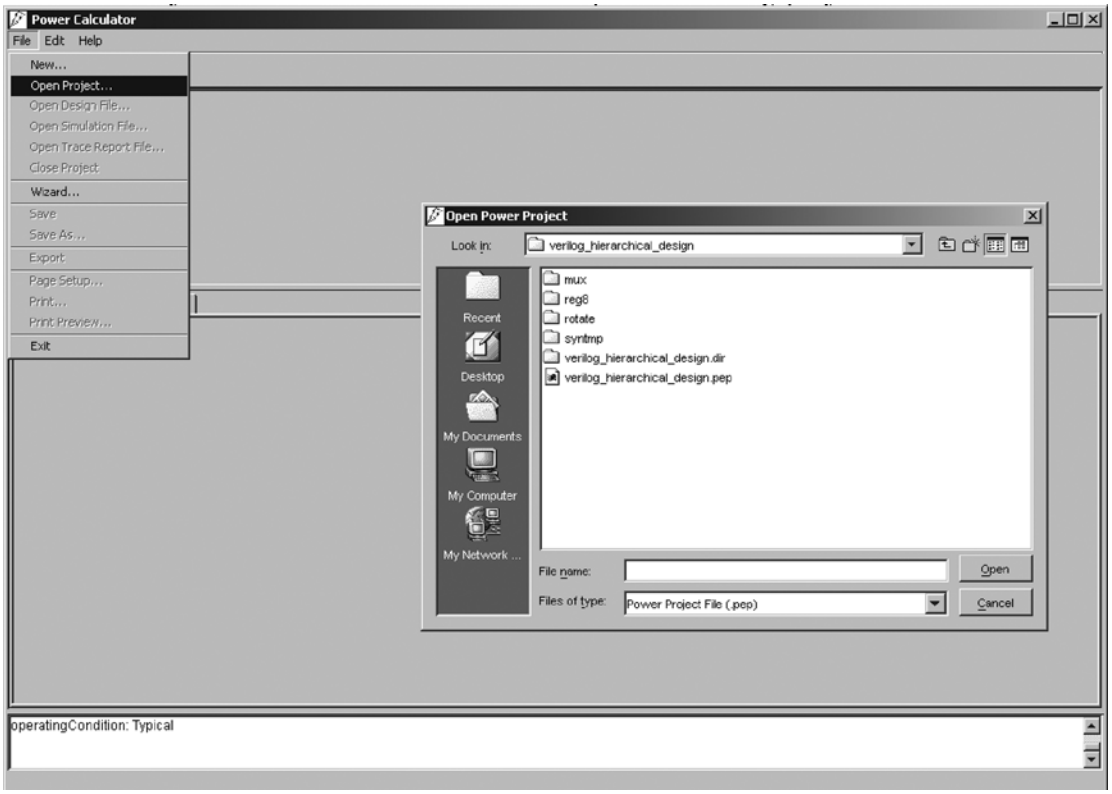
Figure 11-12. Power Calculator Main Window - Resource Utilization Picked up from the NCD File



Power Calculator - Open Existing Project

The Power Calculator - Start Project window also allows users to open an existing project. Select the option **Open Existing Project**, browse to the *.pep project file and click **Continue**. This opens the existing project in similar windows, as discussed above. This is shown in Figure 11-13.

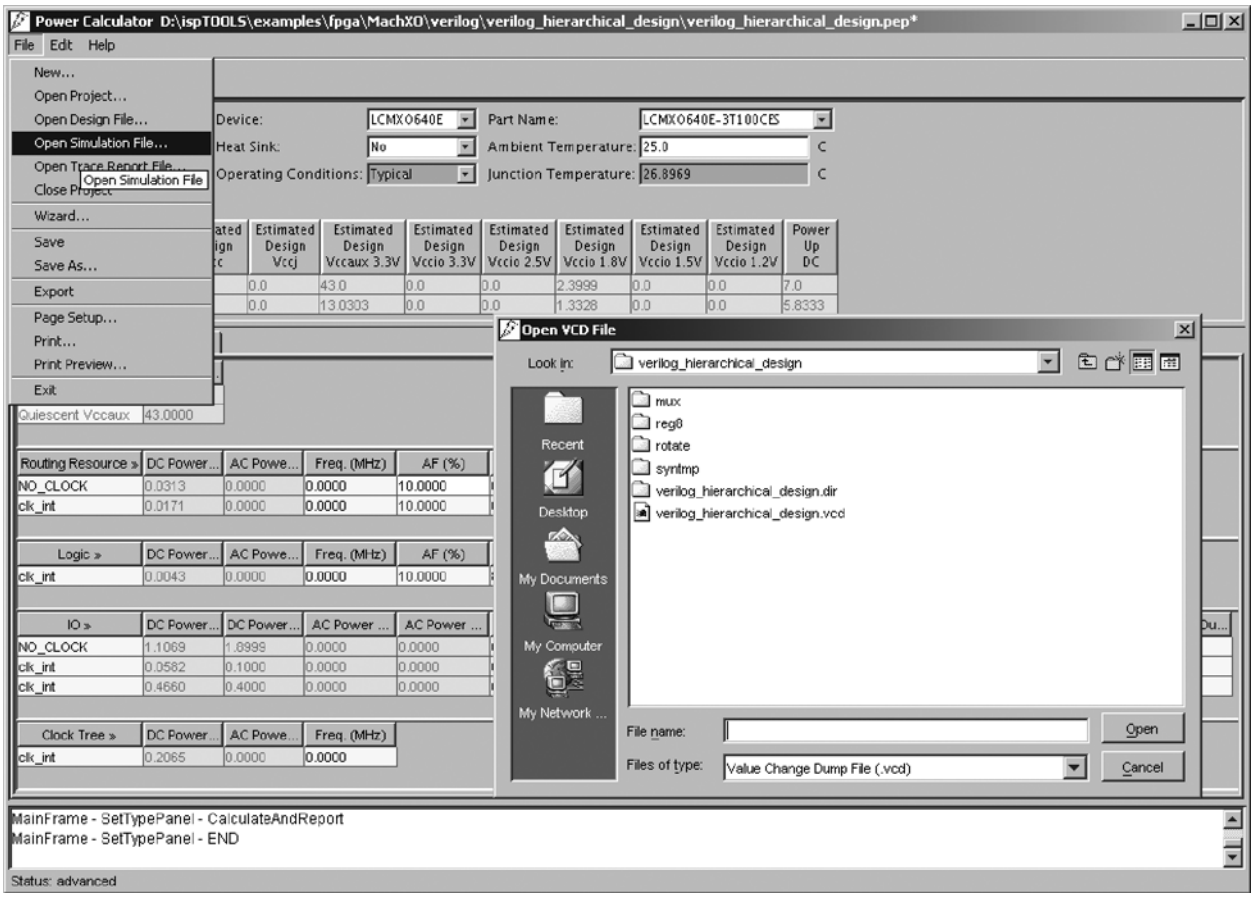
Figure 11-13. Opening an Existing Project in the Power Calculator



Power Calculator - Importing Simulation File (VCD) to the Project

Users can import the post simulation VCD file into the power calculator project to estimate their design's activity factors. Select **Open Simulation File** option under the **File** menu, browse to the VCD file location and select **Open**. All AF and toggle rate fields are populated with the information from this file. This is shown in Figure 11-14.

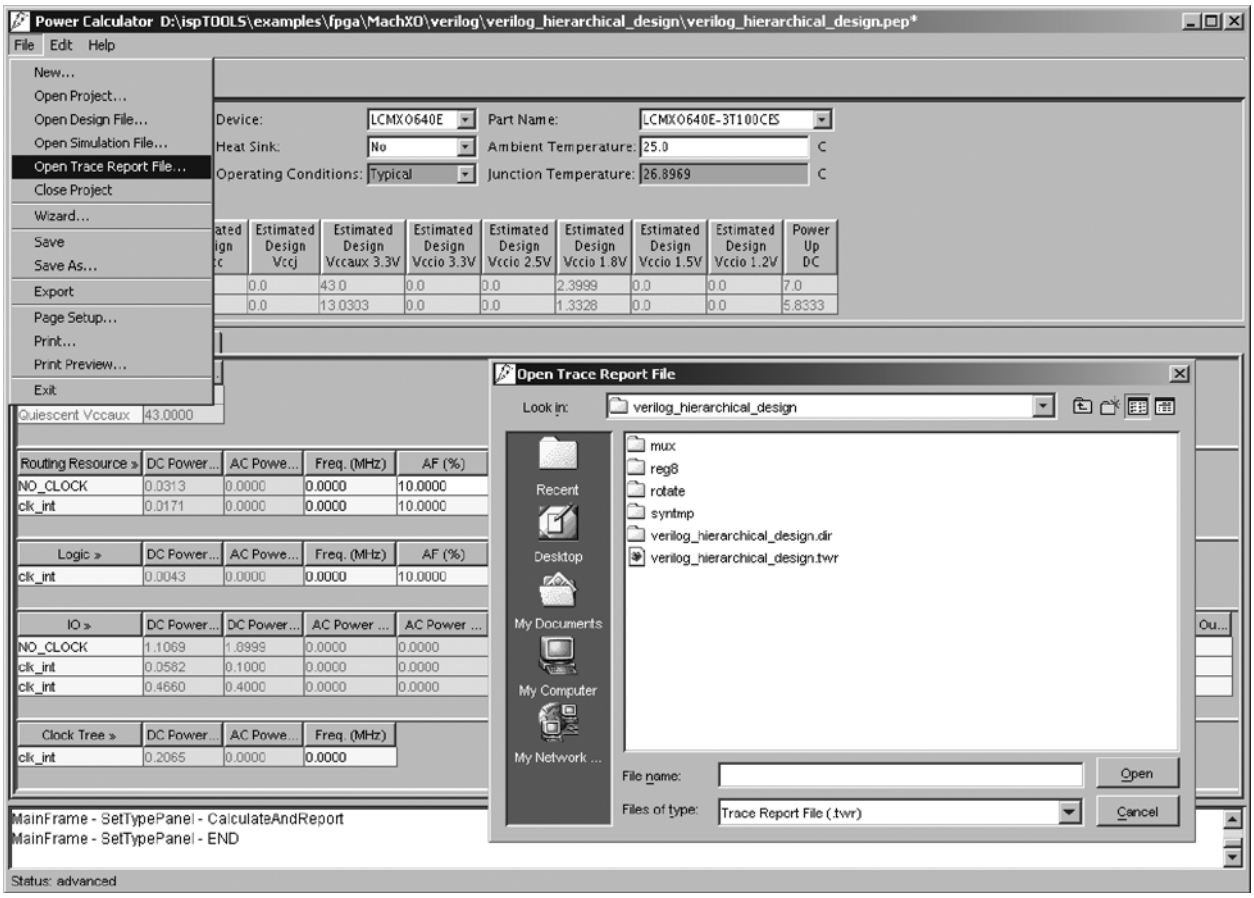
Figure 11-14. Importing Simulation File in the Existing Project in the Power Calculator



Power Calculator - Importing Trace Report File (TWR) to the Project

Users can import the post Trace TWR file into the Power Calculator project to estimate their design's activity factors. Select **Open Trace Report File** option under the **File** menu, browse to the TWR file location and select **Open**. All Freq. (MHZ) fields are populated with the information from this file. This is shown in Figure 11-15.

Figure 11-15. Importing Trace Report File in the Existing Project in the Power Calculator



Activity Factor and Toggle Rate

Activity Factor% (or AF%) is defined as the percentage of frequency (or time) that a signal is active or toggling of the output. Most of the resources associated with a clock domain are running or toggling at some percentage of the frequency at which the clock is running. Users are required to provide this value as a percentage under the AF% column in the Power Calculator tool.

Another term used for I/Os is the I/O Toggle Rate or the I/O Toggle Frequency. The AF% is applicable to the PFU, routing and memory read write ports, etc. The activity of I/Os is determined by the signals provided by the user (in the case of inputs) or as an output of the design (in the case of outputs). So, the rates at which I/Os toggle defines their activity. The Toggle Rate (or TR) in MHz of the output is defined as:

$$\text{Toggle Rate (MHz)} = 1/2 * f_{MAX} * \text{AF\%}$$

Users are required to provide the TR (MHz) value for the I/O instead of providing the frequency and AF% in case of other resources. The AF can be calculated for each routing resource, output or PFU, however it involves long calculations. The general recommendation of a design occupying roughly 30% to 70% of the device is that the AF% used can be between 15% to 25%. This is an average value that can be seen most of the design. The accurate value of an AF depends upon clock frequency, stimulus to the design and the final output.

Ambient and Junction Temperature and Airflow

A common method of characterizing a packaged device's thermal performance is with Thermal Resistance, θ. For a semiconductor device, thermal resistance indicates the steady state temperature rise of the die junction above a given reference for each watt of power (heat) dissipated at the die surface. Its units are °C/W.

The most common examples are θ_{JA}, Thermal Resistance Junction-to-Ambient (in °C/W) and θ_{JC}, Thermal Resistance Junction-to-Case (also in °C/W). Another factor is θ_{JB}, Thermal Resistance Junction-to-Board (in °C/W).

Knowing the reference (i.e. ambient, case or board) temperature, the power and the relevant θ value, the junction temp can be calculated as per the following equations.

$$T_J = T_A + \theta_{JA} * P \text{ . . . . . (1)}$$

$$T_J = T_C + \theta_{JC} * P \text{. . . . . (2)}$$

$$T_J = T_B + \theta_{JB} * P \text{ . . . . . (3)}$$

Where T_J, T_A, T_C and T_B are the junction, ambient, case (or package) and board temperatures (in °C) respectively. P is the total power dissipation of the device.

θ_{JA} is commonly used with natural and forced convection air-cooled systems. θ_{JC} is useful when the package has a high conductivity case mounted directly to a PCB or heatsink. And θ_{JB} applies when the board temp adjacent to the package is known.

Power Calculator utilizes the ambient temperature (°C) to calculate the junction temperature (°C) based on the θ_{JA} for the targeted device, per equation 1 above. Users can also provide the airflow values (in LFM) to get a more accurate value of the junction temperature.

Managing Power Consumption

One of the most critical design factors today is reducing system power consumption, especially for modern handheld devices and electronics. There are several design techniques that designers can use to significantly reduce overall system power consumption. Some of these include:

1. Reducing operating voltage.
2. Operating within the specified package temperature limitations.
3. Using optimum clock frequency reduces power consumption, as the dynamic power is directly proportional to the frequency of operation. Designers must determine if a portion of their design can be clocked at a lower rate which will reduce power.
4. Reducing the span of the design across the device. A more closely placed design utilizes fewer routing resources for less power consumption.
5. Reducing the voltage swing of the I/Os where possible.
6. Using optimum encoding where possible. For example, a 16-bit binary counter has, on average, only 12% Activity Factor and a 7-bit binary counter has an average of 28% Activity Factor. On the other hand, a 7-bit Linear Feedback Shift Register could toggle as much as 50% Activity Factor, which causes higher power consumption. A gray code counter, where only one bit changes at each clock edge, will use the least amount of power, as the Activity Factor would be less than 10%.
7. Minimize the operating temperature, by the following methods:
    a. Use packages that can better dissipate heat. For example, packages with lower thermal impedance.
    b. Place heat sinks and thermal planes around the device on the PCB.
    c. Better airflow techniques using mechanical airflow guides and fans (both system fans and device mounted fans).

Power Calculator Assumptions

The following are the assumptions made in the Power Calculator:

1. The Power Calculator tool is based on equations with constants based on room temperature of 25°C.
2. The user can define the Ambient Temperature (T_A) for device Junction Temperature (T_J) calculation based on the power estimation. T_J is calculated from user-entered T_A and power calculation of typical room temperature.

3. The I/O power consumption is based on output loading of 5pF. Users have ability to change this capacitive loading.
4. The current version of the Power Calculator allows users to get an estimate of the power dissipation and the current for each type of power supplies, that are V_{CC}, V_{CCIO}, V_{CCJ} and V_{CCAUX}.
5. The nominal V_{CC} is used by default to calculate power consumption. Users can choose a lower or higher V_{CC} from a list of available values.
6. The current versions also allow users to enter an airflow in Linear Feet per Minute (LFM) along with the heat sink option to calculate the junction temperature.
7. The default value of the I/O types for the devices is LVCMOS12, 6mA.
8. The Activity Factor is defined as the toggle rate of the registered output. For example, assuming that the input of a flip-flop is changing at every clock cycle, 100% AF of a flip-flop running at 100MHz is 50MHz.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
July 2005	01.0	Initial release.
September 2007	01.1	Document title changed to: “Power Estimation and Management for MachXO Devices.”

MachXO JTAG Programming and Configuration User's Guide

June 2010 Technical Note TN1086

Introduction

The MachXO™ is a reconfigurable programmable logic device. The MachXO uses SRAM memory cells to allow configuring the device to any required functionality. The MachXO also provides non-volatile Flash memory cells to store the configuration data. The data retrieved from the Flash memory rapidly loads the SRAM memory at power-up or at the request of the user. This combination of memory types provides several unique programming and operational capabilities not found in antifuse or SRAM-only based FPGA devices. Some of the capabilities provided by the MachXO:

- "Instant-on" SRAM loading at power-up
- Bitstream security. Since there is no external PROM there is no access to the bitstream used to program the MachXO device.
- "Instant" reconfiguration to a known good state from the Flash memory, i.e. recovery from "soft upset events"
- Reprogramming of the Flash while in normal SRAM operation

This technical note describes how to take advantage of these unique capabilities.

Programming Overview

The MachXO contains two types of memory, SRAM and Flash (refer to Figure 12-1). SRAM contains the active configuration, essentially the "fuses" that define the circuit connections; Flash provides an internal storage space for the configuration data.

The SRAM can be configured in two ways; by using IEEE 1532 mode via the IEEE 1149.1 compliant ispJTAG™ port or from data stored in the on-chip Flash memory. While writing SRAM via the ispJTAG port the state of the FPGA's I/Os is determined by the BSCAN registers. The state of each BSCAN (boundary scan) register can be determined by the user; available options are high, low, tristate (default), or current value (also called leave alone).

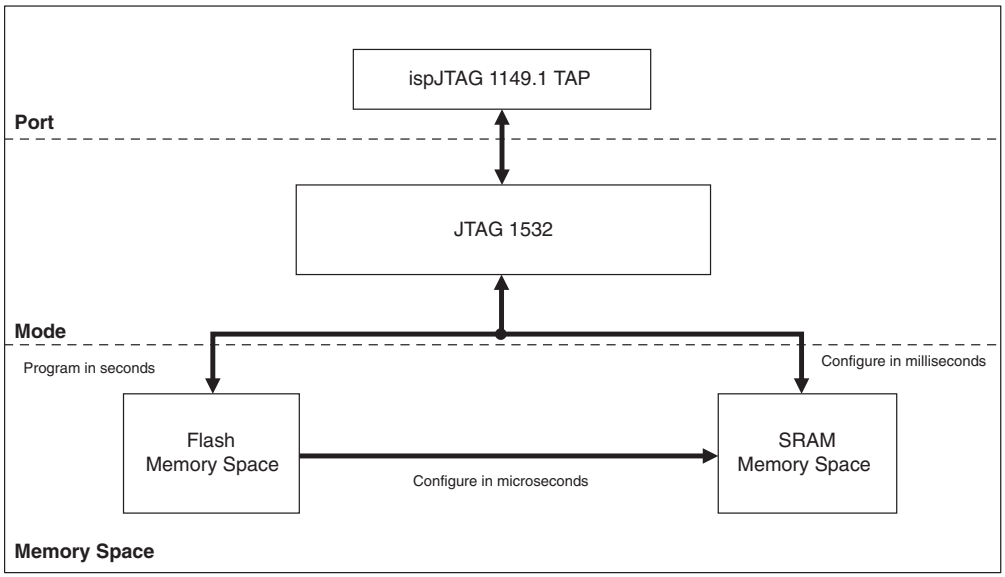
The SRAM may be read without disturbing the operation of the device. This is called transparent, or background readback. Care must be exercised when reading EBR or distributed RAM, as it is possible to cause conflicts with accesses from the user design, causing possible data corruption. It is recommended that read/write enables to the EBR or distributed RAM be turned off any time these resources are being read via the JTAG port.

The on-chip Flash can also be programmed using IEEE 1532 mode via the IEEE 1149.1 compliant ispJTAG port. If the SRAM portion of the device is blank then the Flash will be programmed using direct mode. In direct mode the state of the device's I/Os is determined by the BSCAN registers. The state of each BSCAN register can be determined by the user; available options are high, low, tristate (default), or current value. If the SRAM portion of the device is not blank then the Flash will be programmed in background (transparent) mode.

Both SRAM and Flash memory in the MachXO have multiple security fuses to prevent unauthorized readback of the configuration data. Once set, the only way to clear these security bits is to erase the memory space. A secured device will read out all zeros.

Note that very early production releases of the MachXO1200 and MachXO2280 marked with a "4W" in the part number field did not support writes to, or reads from, the SRAM fabric using JTAG. These parts do, however, support all other programming modes.

Figure 12-1. Programming Block Diagram



ispJTAG

The ispJTAG pins are standard IEEE 1149.1 TAP (Test Access Port) pins. The ispJTAG pins are dedicated pins and are always accessible when the MachXO device is powered up.

Table 12-1. ispJTAG Pins Definition

Pin Name	I/O Type	Pin Type
TDO	Output, weak pull-up	JTAG
TDI	Input, weak pull-up	JTAG
TMS	Input, weak pull-up	JTAG
TCK	Input	JTAG

Note: Weak pull-ups consist of a current source of 30µA to 150µA. The pull-ups for TDO, TDI, and TMS track the associated V_{CCIO}. A pull-down of 4.7K is recommended on TCK.

TDO

The Test Data Output pin is used to shift serial test instructions and data out of the MachXO. TDO is clocked out on the falling edge of TCK. When TDO is not being driven by the internal circuitry, the pin will be in a high impedance state (tristate). An internal pull-up resistor on the TDO pin is provided. The internal resistor is pulled up to the associated V_{CCIO}.

TDI

The Test Data Input pin is used to shift serial test instructions and data into the MachXO. TDI is clocked into the device on the rising edge of TCK. An internal pull-up resistor on the TDI pin is provided. The internal resistor is pulled up to the associated V_{CCIO}.

TMS

The Test Mode Select pin controls test operations on the TAP controller. On the rising edge of TCK, depending on the state of TMS, a transition will be made in the TAP controller state machine. An internal pull-up resistor on the TMS pin is provided. The internal resistor is pulled up to the associated V_{CCIO}.

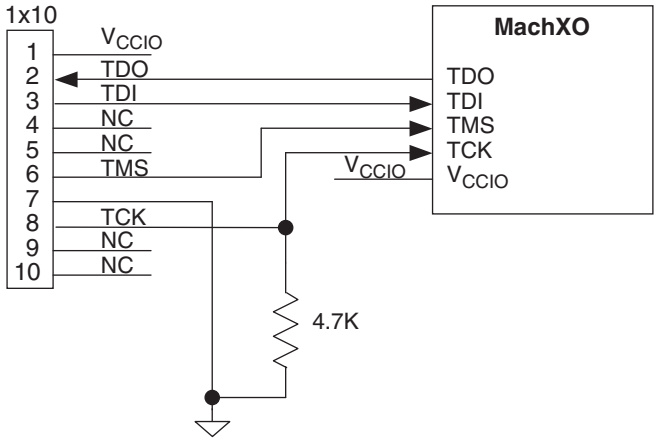
TCK

The test clock pin provides the clock to run the TAP controller, which loads and unloads the data and instruction registers. TCK can be stopped in either the high or low state and can be clocked at speeds up to the frequency indicated in the MachXO Family Data Sheet. The TCK pin does not have a pull-up. A pull-down on the PCB of 4.7 K is recommended to avoid inadvertent clocking of the TAP controller as V_{CC} ramps up.

V_{CC} Supply for JTAG

V_{CC} for the internal JTAG logic is supplied by the associated bank's V_{CCIO}. To determine which bank contains the JTAG pins please refer to the MachXO Family Data Sheet. Valid voltage levels are 3.3V, 2.5V, 1.8V, 1.5V, and 1.2V, but check that the desired voltage is compatible with the ispDOWNLOAD® Cable connected to the download header.

Figure 12-2. Download Header to MachXO Wiring



Note: Place a decoupling capacitor close to the connector's V_{CCIO} pin. Any standard ceramic capacitor value may be used, for example 0.1 µF, 0.01 µF, etc.

Download Cable Pinout

Standard pinouts for the 1x10, 1x8, and 2x5 download headers are shown in the ispDOWNLOAD Cable Data Sheet. All new ispDOWNLOAD Cables have uncommitted "flywire" connections, so they can be attached to any of the header styles. Refer to the ispDOWNLOAD Cable Data Sheet for additional details.

BSDL Files

BSDL files for this device can be found on the Lattice Semiconductor web site. The boundary scan ring covers all of the I/O pins.

Device Wake Up

When configuration is complete (the SRAM has been loaded), the device will wake up in a predictable fashion. The wake up sequence is driven by, and is synchronous to, an internal clock.

Software Options

Preference Options

Preference options are set by opening the Preference Editor within the ispLEVER® or Lattice Diamond™ design software and selecting the global options tab.

Table 12-2. Preference Options for the MachXO

Preference Name	Default Setting [List of All Settings]
CONFIG_SECURE	OFF [off, on]
Usercode Format	Bin[Bin, ASCII, Hex]
Usercode	00000000000000000000000000000000

Security
When CONFIG_SECURE is set to ON the on-chip security fuses will be set and no readback of the general contents (SRAM or Flash) will be supported through the ispJTAG port. The ispJTAG DeviceID area (including the Usercode) and the device Status Register are readable and not considered securable. Default is OFF.

Usercode
Usercode is a user defined 32-bit register, sometimes called the UES (User Electronic Signature). The Usercode can be thought of as a user "notepad". The Usercode is supported as part of the IEEE 1149.1 definition.

Lattice incorporated the Usercode to store such design and manufacturing data as the manufacturer's ID, programming date, programmer make, pattern code, checksum, PCB location, revision number, and product flow. The intent is to assist users with the complex chore of record maintenance and product flow control. In practice, the Usercode can be used for any of a number of ID functions. The Usercode can be easily edited using the Usercode/UES Editor in ispVM® by clicking on **ispTools -> ispVM Editors**.

Note that the Usercode is included in the file transmission checksum, but not the fuse checksum.

Configuring SRAM or Programming Flash

The final step in the design process is to load the JEDEC file created by ispLEVER or Diamond into the MachXO. The JEDEC file can be used to configure the SRAM or program the Flash. Simply connect a Lattice download cable to the ispJTAG connector that is wired to your MachXO and apply power to the FPGA. Then start ispVM and follow these steps:

1. Click on **File -> New**.
2. Click on **Edit -> Add New Device**.
3. **Select** the MachXO device, **Browse** to the JEDEC file created in ispLEVER or Diamond, and then select either SRAM or Flash via the **Device Access Options** drop down box.
4. Click **OK**. You are now ready to program the MachXO by clicking on the green **GO** button on the toolbar.
5. For other options, such as Read and Save, Verify ID, etc., return to **Device Information** by double clicking on the MachXO in **Device List** and use the **Operation** drop-down box.

If you need to create an SVF (Serial Vector File), a file to allow programming from your ATE (Automated Test Equipment), or a file to support VME (ispVM Embedded), simply perform steps 1, 2, and 3 above but click on SVF, ATE, or VME instead of GO.

More information on ispVM and ispVM Embedded can be found by starting ispVM and clicking on Help. Here you will find several tutorials as well as the help facility.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
—	—	Previous releases.
February 2007	01.3	Updated Download Cable Pinout section.
June 2010	01.4	Updated for Lattice Diamond design software support.

HDL Synthesis Coding Guidelines for Lattice Semiconductor FPGAs

October 2005 Technical Note TN1008

Introduction

Coding style plays an important role in utilizing FPGA resources. Although many popular synthesis tools have significantly improved optimization algorithms for FPGAs, it still is the responsibility of the user to generate meaningful and efficient HDL code to guide their synthesis tools to achieve the best result for a specific architecture. This application note is intended to help designers establish useful HDL coding styles for Lattice Semiconductor FPGA devices. It includes VHDL and Verilog design guidelines for both novice and experienced users.

The application note is divided into two sections. The general coding styles for FPGAs section provides an overview for effective FPGA designs. The following topics are discussed in detail:

- Hierarchical Coding
- Design Partitioning
- Encoding Methodologies for State Machines
- Coding Styles for Finite State Machines (FSM)
- Using Pipelines
- Comparing IF Statements and CASE Statements
- Avoiding Non-intentional Latches

The HDL Design with Lattice Semiconductor FPGA Devices section covers specific coding techniques and examples:

- Using the Lattice Semiconductor FPGA Synthesis Library
- Implementation of Multiplexers
- Creating Clock Dividers
- Register Control Signals (CE, LSR, GSR)
- Using PIC Features
- Implementation of Memories
- Preventing Logic Replication and Fanout
- Comparing Synthesis Results and Place and Route Results

General Coding Styles for FPGA

The following recommendations for common HDL coding styles will help users generate robust and reliable FPGA designs.

Hierarchical Coding

HDL designs can either be synthesized as a flat module or as many small hierarchical modules. Each methodology has its advantages and disadvantages. Since designs in smaller blocks are easier to keep track of, it is preferred to apply hierarchical structure to large and complex FPGA designs. Hierarchical coding methodology allows a group of engineers to work on one design at the same time. It speeds up design compilation, makes changing the implementation of key blocks easier, and reduces the design period by allowing the re-use of design modules for current and future designs. In addition, it produces designs that are easier to understand. However, if the design mapping into the FPGA is not optimal across hierarchical boundaries, it will lead to lower device utilization and design performance. This disadvantage can be overcome with careful design considerations when choosing the design hierarchy. Here are some tips for building hierarchical structures:

- The top level should only contain instantiation statements to call all major blocks
- Any I/O instantiations should be at the top level
- Any signals going into or out of the devices should be declared as input, output or bi-directional pins at the top level

- Memory blocks should be kept separate from other code

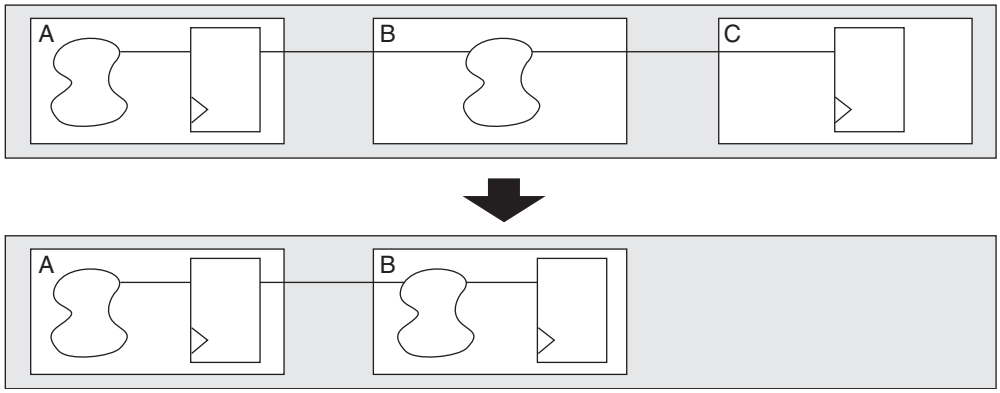
Design Partitioning

By effectively partitioning the design, a designer can reduce overall run time and improve synthesis results. Here are some recommendations for design partitioning.

Maintain Synchronous Sub-blocks by Registering All Outputs

It is suggested to arrange the design boundary such that the outputs in each block are registered. Registering outputs helps the synthesis tool to consider the implementation of the combinatorial logic and registers into the same logic block. Registering outputs also makes the application of timing constraints easier since it eliminates possible problems with logic optimization across design boundaries. Single clock is recommended for each synchronous block because it significantly reduces the timing consideration in the block. It leaves the adjustment of the clock relationships of the whole design at the top level of the hierarchy. Figure 13-1 shows an example of synchronous blocks with registered outputs.

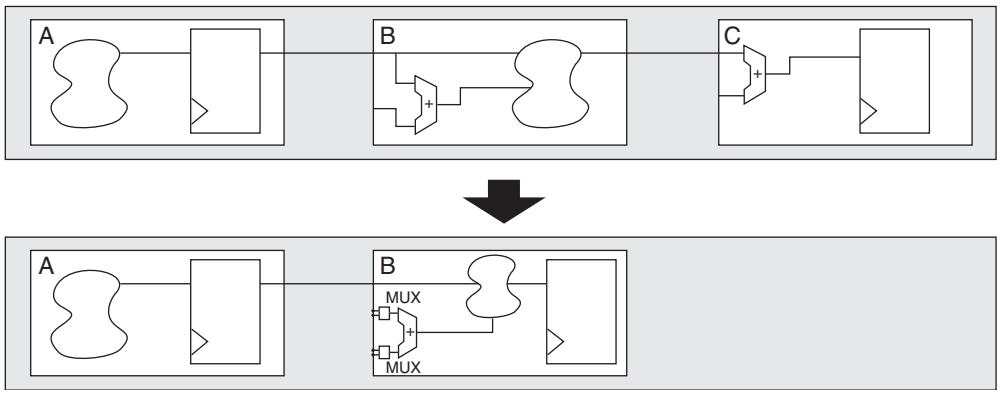
Figure 13-1. Synchronous Blocks with Registered Outputs



Keep Related Logic Together in the Same Block

Keeping related logic and sharable resources in the same block allows the sharing of common combinatorial terms and arithmetic functions within the block. It also allows the synthesis tools to optimize the entire critical path in a single operation. Since synthesis tools can only effectively handle optimization of certain amounts of logic, optimization of critical paths pending across the boundaries may not be optimal. Figure 13-2 shows an example of merging sharable resource in the same block.

Figure 13-2. Merge Sharable Resource in the Same Block



Separate Logic with Different Optimization Goals

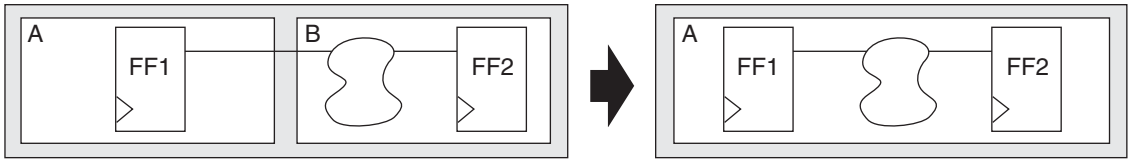
Separating critical paths from non-critical paths may achieve efficient synthesis results. At the beginning of the project, one should consider the design in terms of performance requirements and resource requirements. If there are

two portions of a block, one that needs to be optimized for area and a second that needs to be optimized for speed, they should be separated into two blocks. By doing this, different optimization strategies for each module can be applied without being limited by one another.

Keep Logic with the Same Relaxation Constraints in the Same Block

When a portion of the design does not require high performance, this portion can be applied with relaxed timing constraints such as “multicycle” to achieve high utilization of device area. Relaxation constraints help to reduce overall run time. They can also help to efficiently save resources, which can be used on critical paths. Figure 13-3 shows an example of grouping logic with the same relaxation constraint in one block.

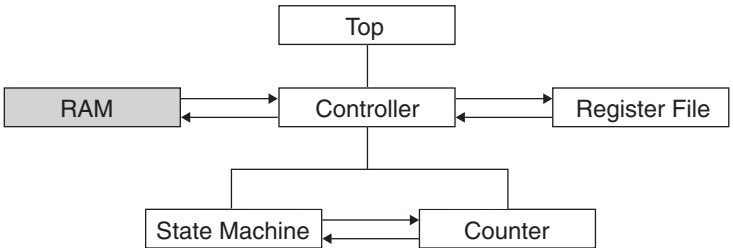
Figure 13-3. Logic with the Same Relaxation Constraint



Keep Instantiated Code in Separate Blocks

It is recommended that the RAM block in the hierarchy be left in a separate block (Figure 13-4). This allows for easy swapping between the RAM behavioral code for simulation, and the code for technology instantiation. In addition, this coding style facilitates the integration of the ispLEVER® IPexpress™ tool into the synthesis process.

Figure 13-4. Separate RAM Block



Keep the Number FPGA Gates at 30 to 80 PFUs Per Block

This range varies based on the computer configuration, time required to complete each optimization run, and the targeted FPGA routing resources. Although a smaller block methodology allows more control, it may not produce the most efficient design since it does not provide the synthesis tool enough logic to apply “Resource Sharing” algorithms. On the other hand, having a large number of gates per block gives the synthesis tool too much to work on and causes changes that affect more logic than necessary in an incremental or multi-block design flow.

State Encoding Methodologies for State Machines

There are several ways to encode a state machine, including binary encoding, gray-code encoding and one-hot encoding. State machines with binary or gray-code encoded states have minimal numbers of flip-flops and wide combinatorial functions, which are typically favored for CPLD architectures. However, most FPGAs have many flip-flops and relatively narrow combinatorial function generators. Binary or gray-code encoding schemes can result in inefficient implementation in terms of speed and density for FPGAs. On the other hand, one-hot encoded state machine represents each state with one flip-flop. As a result, it decreases the width of combinatorial logic, which matches well with FPGA architectures. For large and complex state machines, one-hot encoding usually is the preferable method for FPGA architectures. For small state machines, binary encoding or gray-code encoding may be more efficient.

There are many ways to ensure the state machine encoding scheme for a design. One can hard code the states in the source code by specifying a numerical value for each state. This approach ensures the correct encoding of the state machine but is more restrictive in the coding style. The enumerated coding style leaves the flexibility of state machine encoding to the synthesis tools. Most synthesis tools allow users to define encoding styles either through

attributes in the source code or through the tool's Graphical User Interface (GUI). Each synthesis tool has its own synthesis attributes and syntax for choosing the encoding styles. Refer to the synthesis tools documentation for details about attributes syntax and values.

The following syntax defines an enumeration type in VHDL:

```
type type_name is (state1_name,state2_name,.....,stateN_name)
```

Here is a VHDL example of enumeration states:

```
type STATE_TYPE is (S0,S1,S2,S3,S4);
signal CURRENT_STATE, NEXT_STATE : STATE_TYPE;
```

The following are examples of Synplify® and LeonardoSpectrum® VHDL synthesis attributes.

Synplify:

```
attribute syn_encoding : string;
attribute syn_encoding of <signal_name> : type is "value ";
-- The syn_encoding attribute has 4 values : sequential, onehot, gray and safe.
```

LeonardoSpectrum:

```
-- Declare TYPE_ENCODING_STYLE attribute
-- Not needed if the exemplar_1164 package is used
type encoding_style is (BINARY, ONEHOT, GRAY, RANDOM, AUTO);
attribute TYPE_ENCODING_STYLE : encoding_style;
...
attribute TYPE_ENCODING_STYLE of <typename> : type is ONEHOT;
```

In Verilog, one must provide explicit state values for states. This can be done by using the bit pattern (e.g., 3'b001), or by defining a parameter and using it as the case item. The latter method is preferable. The following is an example using parameter for state values.

```
Parameter state1 = 2'h1, state2 = 2'h2;
...
current_state = state2; // setting current state to 2'h2
```

The attributes in the source code override the default encoding style assigned during synthesis. Since Verilog does not have predefined attributes for synthesis, attributes are usually attached to the appropriate objects in the source code as comments. The attributes and their values are case sensitive and usually appear in lower case. The following examples use attributes in Verilog source code to specify state machine encoding style.

Synplify:

```
Reg[2:0] state; /* synthesis syn_encoding = "value" */;
// The syn_encoding attribute has 4 values : sequential, onehot, gray and safe.
```

In LeonardoSpectrum, it is recommended to set the state machine variable to an enumeration type with enum pragma. Once this is set in the source code, encoding schemes can be selected in the LeonardoSpectrum GUI.

LeonardoSpectrum:

```
Parameter /* exemplar enum <type_name> */ s0 = 0, s1 = 1, s2 = 2, s3 = 3, S4 = 4;
Reg [2:0] /* exemplar enum <type_name> */ present_state, next_state ;
```

In general, synthesis tools will select the optimal encoding style that takes into account the target device architecture and size of the decode logic. One can always apply synthesis attributes to override the default encoding style if necessary.

Coding Styles for FSM

A finite state machine (FSM) is a hardware component that advances from the current state to the next state at the clock edge. As mentioned in the Encoding Methodologies for State Machines section, the preferable scheme for FPGA architectures is one-hot encoding. This section discusses some common issues encountered when constructing state machines, such as initialization and state coverage, and special case statements in Verilog.

General State Machine Description

Generally, there are two approaches to describe a state machine. One is to use one process/block to handle both state transitions and state outputs. The other is to separate the state transition and the state outputs into two different process/blocks. The latter approach is more straightforward because it separates the synchronous state registers from the decoding logic used in the computation of the next state and the outputs. This will make the code easier to read and modify, and makes the documentation more efficient. If the outputs of the state machine are combinatorial signals, the second approach is almost always necessary because it will prevent the accidental registering of the state machine outputs.

The following examples describe a simple state machine in VHDL and Verilog. In the VHDL example, a sequential process is separated from the combinatorial process. In Verilog code, two *always* blocks are used to describe the state machine in a similar way.

VHDL Example for State Machine

```
. . .
architecture lattice_fpga of dram_refresh is
   type state_typ is (s0, s1, s2, s3, s4);
   signal present_state, next_state : state_typ;
begin
   -- process to update the present state
   registers: process (clk, reset)
   begin
      if (reset='1') then
         present_state <= s0;
      elsif clk'event and clk='1' then
         present_state <= next_state;
      end if;
   end process registers;

   -- process to calculate the next state & output
   transitions: process (present_state, refresh, cs)
   begin
      ras <= '0'; cas <= '0'; ready <= '0';
      case present_state is
         when s0 =>
            ras <= '1'; cas <= '1'; ready <= '1';
            if (refresh = '1') then next_state <= s3;
            elsif (cs = '1') then next_state <= s1;
            else next_state <= s0;
            end if;
         when s1 =>
            ras <= '0'; cas <= '1'; ready <= '0';
            next_state <= s2;
         when s2 =>
            ras <= '0'; cas <= '0'; ready <= '0';
            if (cs = '0') then next_state <= s0;
            else next_state <= s2;
            end if;
         when s3 =>
            ras <= '1'; cas <= '0'; ready <= '0';
            next_state <= s4;
         when s4 =>
            ras <= '0'; cas <= '0'; ready <= '0';
            next_state <= s0;
         when others =>
            ras <= '0'; cas <= '0'; ready <= '0';
            next_state <= s0;
      end case;
   end process transitions;
. . .
```

Verilog Example for State Machine

```
. . .
parameter s0 = 0, s1 = 1, s2 = 2, s3 = 3, s4 = 4;

reg [2:0] present_state, next_state;
reg ras, cas, ready;

// always block to update the present state
always @ (posedge clk or posedge reset)
begin
   if (reset) present_state = s0;
   else present_state = next_state;
end

// always block to calculate the next state & outputs
always @ (present_state or refresh or cs)
begin
   next_state = s0;
   ras = 1'bX; cas = 1'bX; ready = 1'bX;
   case (present_state)
      s0 : if (refresh) begin
              next_state = s3;
              ras = 1'b1; cas = 1'b0; ready = 1'b0;
           end
           else if (cs) begin
              next_state = s1; ras = 1'b0; cas = 1'b1; ready = 1'b0;
           end
           else begin
              next_state = s0; ras = 1'b1; cas = 1'b1; ready = 1'b1;
           end
      s1 : begin
              next_state = s2; ras = 1'b0; cas = 1'b0; ready = 1'b0;
           end
      s2 : if (~cs) begin
              next_state = s0; ras = 1'b1; cas = 1'b1; ready = 1'b1;
           end
           else begin
              next_state = s2; ras = 1'b0; cas = 1'b0; ready = 1'b0;
           end
      s3 : begin
              next_state = s4; ras = 1'b1; cas = 1'b0; ready = 1'b0;
           end
      s4 : begin
              next_state = s0; ras = 1'b0; cas = 1'b0; ready = 1'b0;
           end
   endcase
end
. . .
```

Initialization and Default State

A state machine must be initialized to a valid state after power-up. This can be done at the device level during power up or by including a reset operation to bring it to a known state. For all Lattice Semiconductor FPGA devices, the Global Set/Reset (GSR) is pulsed at power-up, regardless of the function defined in the design source code. In the above example, an asynchronous reset can be used to bring the state machine to a valid initialization state. In the same manner, a state machine should have a default state to ensure the state machine will not go into an invalid state if not all the possible combinations are clearly defined in the design source code. VHDL and Verilog have different syntax for default state declaration. In VHDL, if a CASE statement is used to construct a state machine, “When Others” should be used as the last statement before the end of the statement, If an IF-THEN-ELSE statement is used, “Else” should be the last assignment for the state machine. In Verilog, use “default” as the last assignment for a CASE statement, and use “Else” for the IF-THEN-ELSE statement.

When Others in VHDL

```
...
architecture lattice_fpga of FSM1 is
   type state_typ is (deflt, idle, read, write);
   signal next_state : state_typ;
begin
process(clk, rst)
begin
   if (rst='1') then
      next_state <= idle; dout <= '0';
   elsif (clk'event and clk='1') then
      case next_state is
         when idle =>
            next_state <= read; dout <= din(0);
         when read =>
            next_state <= write; dout <= din(1);
         when write =>
            next_state <= idle; dout <= din(2);
         when others =>
            next_state <= deflt; dout <= '0';
      end case;
   end if;
end process;
...
```

Default Clause in Verilog

```
...
// Define state labels explicitly
parameter deflt=2'bxx;
parameter idle =2'b00;
parameter read =2'b01;
parameter write=2'b10;

reg [1:0] next_state;
reg dout;

always @(posedge clk or posedge rst)
   if (rst) begin
      next_state <= idle;
      dout <= 1'b0;
   end
   else begin
      case(next_state)
         idle: begin
                dout <= din[0]; next_state <= read;
               end
         read: begin
                dout <= din[1]; next_state <= write;
               end
         write: begin
                dout <= din[2]; next_state <= idle;
               end
         default: begin
                   dout <= 1'b0; next_state <= deflt;
                  end
```

Full Case and Parallel Case Specification in Verilog

Verilog has additional attributes to define the default states without writing it specifically in the code. One can use “full_case” to achieve the same performance as “default”. The following examples show the equivalent representations of the same code in Synplify. LeonardoSpectrum allows users to apply Verilog-specific options in the GUI settings.

```
…
case (current_state)  // synthesis full_case
   2’b00 : next_state <= 2’b01;
   2’b01 : next_state <= 2’b11;
   2’b11 : next_state <= 2’b00;
…
```

```
…
case (current_state)
   2’b00 : next_state <= 2’b01;
   2’b01 : next_state <= 2’b11;
   2’b11 : next_state <= 2’b00;
   default : next_state <= 2bx;
```

“Parallel_case” makes sure that all the statements in a case statement are mutually exclusive. It is used to inform the synthesis tools that only one case can be true at a time. The syntax for this attribute in Synplify is as follows:

```
// synthesis parallel_case
```

Using Pipelines in the Designs

Pipelining can improve design performance by restructuring a long data path with several levels of logic and breaking it up over multiple clock cycles. This method allows a faster clock cycle by relaxing the clock-to-output and setup time requirements between the registers. It is usually an advantageous structure for creating faster data paths in register-rich FPGA devices. Knowledge of each FPGA architecture helps in planning pipelines at the

beginning of the design cycle. When the pipelining technique is applied, special care must be taken for the rest of the design to account for the additional data path latency. The following illustrates the same data path before (Figure 13-5) and after pipelining (Figure 13-6).

Figure 13-5. Before Pipelining

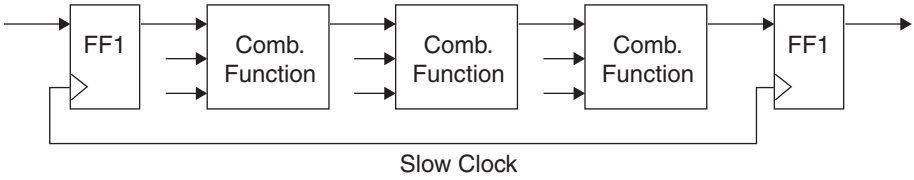
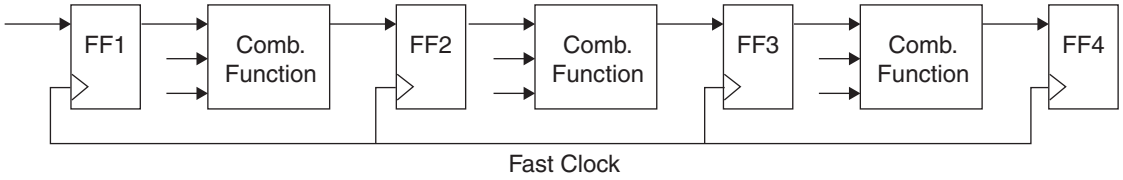


Figure 13-6. After Pipelining



Before pipelining, the clock speed is determined by the clock-to-out time of the source register, the logic delay through four levels of combinatorial logic, the associated routing delays, and the setup time of the destination register. After pipelining is applied, the clock speed is significantly improved by reducing the delay of four logic levels to one logic level and the associated routing delays, even though the rest of the timing requirements remain the same. It is recommended to check the Place and Route timing report to ensure that the pipelined design gives the desired performance.

Comparing IF statement and CASE statement

CASE and IF-THEN-ELSE statements are common for sequential logic in HDL designs. The IF-THEN-ELSE statement generally generates priority-encoded logic, whereas the CASE statement implements balanced logic. An IF-THEN-ELSE statement can contain a set of different expressions while a Case statement is evaluated against a common controlling expression. Both statements will give the same functional implementation if the decode conditions are mutually exclusive, as shown in the following VHDL codes.

```
-- Case Statement — mutually exclusive conditions
process (s, x, y, z)
begin
   O1 <= '0';
   O2 <= '0';
   O3 <= '0';
   case (s) is
      when "00" => O1 <= x;
      when "01" => O2 <= y;
      when "10" => O3 <= z;
   end case;
end process;
```

```
-- If-Then-Else — mutually exclusive conditions
process (s, x, y, z)
begin
   O1 <= '0';
   O2 <= '0';
   O3 <= '0';
   if s = "00" then O1 <= x;
   elsif s = "01" then O2 <= y;
   elsif s = "10" then O3 <= z;
   end if;
end process;
```

However, the use of If-Then-Else construct could be a key pitfall to make the design more complex than necessary, because extra logic are needed to build a priority tree. Consider the following examples:

```
--A: If-Then-Elese Statement: Complex O3 Equations
process(s1, s2, s3, x, y, z)
begin
   O1 <= '0';
   O2 <= '0';
   O3 <= '0';
   if s1 = '1' then
      O1 <= x;
   elsif s2 = '1' then
      O2 <= y;
   elsif s3 = '1' then
      O3 <= z;
   end if;
end process;
```

```
--B: If-Then-Else Statement: Simplified O3 Equation
process (s1, s2, s3, x, y, z)
begin
   O1 <= '0';
   O2 <= '0';
   O3 <= '0';
   if s1 = '1' then
      O1 <= x;
   end if;
   if s2 = '1' then
      O2 <= y;
   end if;
   if s3 <= '1' then
      O3 <= z;
   end if;
 end process;
```

If the decode conditions are not mutually exclusive, IF-THEN-ELSE construct will cause the last output to be dependent on all the control signals. The equation for O3 output in example A is:

```
O3 <= z and (s3) and (not (s1 and s2));
```

If the same code can be written as in example B, most of the synthesis tools will remove the priority tree and decode the output as:

```
O3 <= z and s3;
```

This reduces the logic requirement for the state machine decoder. If each output is indeed dependent of all of the inputs, it is better to use a CASE statement since CASE statements provide equal branches for each output.

Avoiding Non-intentional Latches

Synthesis tools infer latches from incomplete conditional expressions, such as an IF-THEN-ELSE statements without an Else clause. To avoid non-intentional latches, one should specify all conditions explicitly or specify a default assignment. Otherwise, latches will be inserted into the resulting RTL code, requiring additional resources in the device or introducing combinatorial feedback loops that create asynchronous timing problems. Non-intentional latches can be avoided by using clocked registers or by employing any of the following coding techniques:

- Assigning a default value at the beginning of a process
- Assigning outputs for all input conditions
- Using else, (when others) as the final clause

Another way to avoid non-intentional latches is to check the synthesis tool outputs. Most of the synthesis tools give warnings whenever there are latches in the design. Checking the warning list after synthesis will save a tremendous amount of effort in trying to determine why a design is so large later in the Place and Route stage.

HDL Design with Lattice Semiconductor FPGA Devices

The following section discusses the HDL coding techniques utilizing specific Lattice Semiconductor FPGA system features. This kind of architecture-specific coding style will further improve resource utilization and enhance the performance of designs.

Lattice Semiconductor FPGA Synthesis Library

The Lattice Semiconductor FPGA Synthesis Library includes a number of library elements to perform specific logic functions. These library elements are optimized for Lattice Semiconductor FPGAs and have high performance and utilization. The following are the classifications of the library elements in the Lattice Semiconductor FPGA Synthe-

sis Library. The definitions of these library elements can be found in the *Reference Manuals* section of the ispLEVER on-line help system.

- Logic gates and LUTs
- Comparators, adders, subtractors
- Counters
- Flip-flops and latches
- Memory, 4E-specific memory (block RAM function)
- Multiplexors
- Multipliers
- All I/O cells, including I/O flip-flops
- PIC cells
- Special cells, including PLL, GSR, boundary scan, etc.
- FPSC elements

IPepxress, a parameterized module complier optimized for Lattice FPGA devices, is available for more complex logic functions. IPexpress supports generation of library elements with a number of different options such as PLLs and creates parameterized logic functions such as PFU and EBR memory, multipliers, adders, subtractors, and counters. IPexpress accepts options that specify parameters for parameterized modules such as data path modules and memory modules, and produces a circuit description with Lattice Semiconductor FPGA library elements. Output from IPexpress can be written in EDIF, VHDL, or Verilog. In order to use synthesis tools to utilize the Lattice FPGA architectural features, it is strongly recommended to use IPexpress to generate modules for source code instantiation. The following are examples of Lattice Semiconductor FPGA modules supported by IPexpress:

- PLL
- Memory implemented in PFU:
  – Synchronous single-port RAM, synchronous dual-port RAM, synchronous ROM, synchronous FIFO
- Memory implemented with EBR:
  – Quad-port Block RAM, Dual-Port Block RAM, Single-Port Block RAM, ROM, FIFO
- Other EBR based Functions
  – Multiplier, CAM
- PFU based functions
  – Multiplier, adder, subtractor, adder/subtractor, linear feedback shifter, counter
- MPI/System Bus

IPexpress is especially efficient when generating high pin count modules as it saves time in manually cascading small library elements from the synthesis library. Detailed information about IPexpress and its user guide can be found in the ispLEVER help system.

Implementing Multiplexers

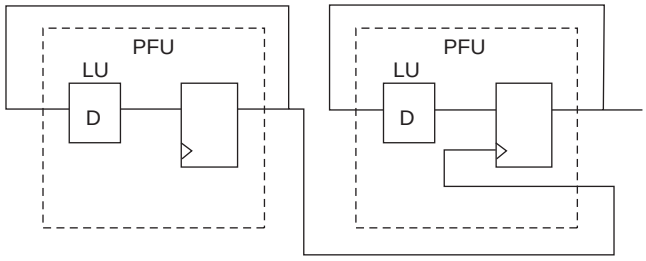
The flexible configurations of LUTs can realize any 4-, 5-, or 6-input logic function like 2-to-1, 3-to-1 or 4-to-1 multiplexers. Larger multiplexers can be efficiently created by programming multiple 4-input LUTs. Synthesis tools camn automatically infer Lattice FPGA optimized multiplexer library elements based on the behavioral description in the HDL source code. This provides the flexibility to the Mapper and Place and Route tools to configure the LUT mode and connections in the most optimum fashion.

```
16:1 MUX
…
process(sel, din)
   begin
      if    (sel="0000") then muxout <= din(0);
      elsif (sel="0001") then muxout <= din(1);
      elsif (sel="0010") then muxout <= din(2);
      elsif (sel="0011") then muxout <= din(3);
      elsif (sel="0100") then muxout <= din(4);
      elsif (sel="0101") then muxout <= din(5);
      elsif (sel="0110") then muxout <= din(6);
      elsif (sel="0111") then muxout <= din(7);
      elsif (sel="1000") then muxout <= din(8);
      elsif (sel="1001") then muxout <= din(9);
      elsif (sel="1010") then muxout <= din(10);
      elsif (sel="1011") then muxout <= din(11);
      elsif (sel="1100") then muxout <= din(12);
      elsif (sel="1101") then muxout <= din(13);
      elsif (sel="1110") then muxout <= din(14);
      elsif (sel="1111") then muxout <= din(15);
      else muxout <= '0';
      end if;
   end process;
…
```

Clock Dividers

There are two ways to implement clock dividers in Lattice Semiconductor FPGA devices. The first is to cascade the registers with asynchronous clocks. The register output feeds the clock pin of the next register (Figure 13-7). Since the clock number in each PFU is limited to two, any clock divider with more than two bits will require multiple PFU implementations. As a result, the asynchronous daisy chaining implementation of clock divider will be slower due to the inter-PFU routing delays. This kind of delays is usually ambiguous and inconsistent because of the nature of FPGA routing structures.

Figure 13-7. Daisy Chaining of Flip-flops



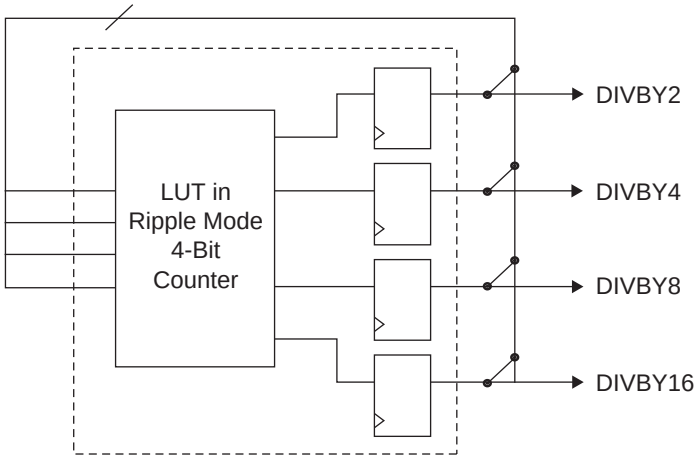
The following are the HDL representations of the design in Figure 13-7.

```
-- VHDL Example of Daisy Chaining FF
...
-- 1st FF to divide Clock in half
CLK_DIV1: process(CLK, RST)
begin
   if (RST='1') then
      clk1 <= '0';
   elsif (CLK'event and CLK='1') then
      clk1 <= not clk1;
   end if;
end process CLK_DIV1;

-- 2nd FF to divide clock in half
CLK_DIV2: process(clk1, RST)
begin
   if (RST='1') then
      clk2 <= '0';
   elsif (clk1'event and clk1='1') then
      clk2 <= not clk2;
   end if;
end process CLK_DIV2;
```

```
//Verilog Example of Daisy Chaining FF
...
always @(posedge CLK or posedge RST)
begin
   if (RST)
      clk1 = 1'b0;
   else
      clk1 = !clk1;
end

always @(posedge clk1 or posedge RST)
begin
   if (RST)
      clk2 = 1'b0;
   else
      clk2 = !clk2;
end

...
```

The preferable way is to fully employ the PFU's natural “Ripple-mode”. A single PFU can support up to 8-bit ripple functions with fast carry logic. Figure 13-8 is an example of 4-bit counter in PFU “Ripple Mode”. In Lattice Semiconductor FPGA architectures, an internal generated clock can get on the clock spine for small skew clock distribution, further enhancing the performance of the clock divider.

Figure 13-8. Use PFU “Ripple Mode”



Here are the HDL representations of the design in Figure 13-8.

```
-- VHDL : “RippleMode” Clock Divider
...

COUNT4: process(CLK, RST)
begin
   if (RST='1') then
      cnt <= (others=>'0');
   elsif (CLK'event and CLK='1') then
      cnt <= cnt + 1;
   end if;
end process COUNT4;

DIVBY4  <= cnt(1);
DIVBY16 <= cnt(3);
```

```
//Verilog : “RippleMode” Clock Divider
...
always @(posedge CLK or posedge RST)
begin
   if (RST)
      cnt = 4'b0;
   else
      cnt = cnt + 1'b1;
end

assign DIVBY4  = cnt[1];
assign DIVBY16 = cnt[3];

...
```

Register Control Signals

The general-purpose latches/FFs in the PFU are used in a variety of configurations depending on device family. For example, the Lattice EC, ECP, SC and XP family of devices clock, clock enable and LSR control can be applied to the registers on a slice basis. Each slice contains two LUT4 lookup tables feeding two registers (programmed asto be in FF or Latch mode), and some associated logic that allows the LUTs to be combined to perform functions such as LUT5, LUT6, LUT7 and LUT8. There is control logic to perform set/reset functions (prgorammable as synchronous/asynchronous), clock select, chip-select and wider RAM/ROM functions. The ORCA Series 4 family of devices clock, clock enable and LSR control can be applied to the registers on a nibble-wide basis. When writing design codes in HDL, keep the architecture in mind to avoid wasting resources in the device. Here are several points for consideration:

- If the register number is not a multiple of 2 or 4 (dependent on device family), try to code the registers in a way that all registers share the same clock, and in a way that all registers share the same control signals.
- Lattice Semiconductor FPGA devices have multiple dedicated Clock Enable signals per PFU. Try to code the asynchronous clocks as clock enables, so that PFU clock signals can be released to use global low-skew clocks.
- Try to code the registers with Local synchronous Set/Reset and Global asynchronous Set/Reset

For more detailed architecture information, refer to the Lattice Semiconductor FPGA data sheets.

Clock Enable

Figure 13-9 shows an example of gated clocking. Gating clock is not encouraged in digital designs because it may cause timing issues such as unexpected clock skews. The structure of the PFU makes the gating clock even more undesirable since it will use up all the clock resources in one PFU and sometimes waste the FF/ Latches resources in the PFU. By using the clock enable in the PFU, the same functionality can be achieved without worrying about timing issues as only one signal is controlling the clock. Since only one clock is used in the PFU, all related logic can be implemented in one block to achieve better performance. Figure 13-10 shows the design with clock enable signal being used.

Figure 13-9. Asynchronous: Gated Clocking

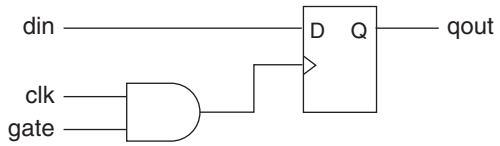
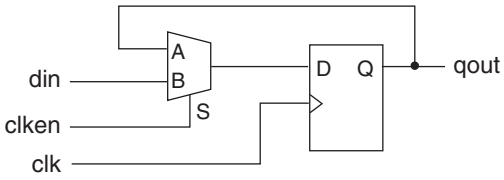


Figure 13-10. Synchronous: Clock Enabling



The VHDL and Verilog coding for Clock Enable are as shown in Figure 13-10.

```
-- VHDL example for Clock Enable

...
Clock_Enable: process(clk)
begin
   if (clk'event or clk='1') then
      if (clken='1') then
         qout <= din;
      end if;
   end if;
end process Clock_Enable;
```

```
// Verilog example for Clock Enable

...
always @(posedge clk)
   qout <= clken ? din : qout;

...
```

The following are guidelines for coding the Clock Enable in Lattice Semiconductor FPGAs:

- Clock Enable is only supported by FFs, not latches.
- Nibble wide FFs and slices inside a PFU share the same Clock Enable
- All flip-flops in the Lattice Semiconductor FPGA library have a positive clock enable signal
- In the ORCA Series 4 architecture, the Clock Enable signal has the higher priority over synchronous set/reset by default. However, it can be programmed to have the priority of synchronous LSR over the priority of Clock Enable. This can be achieved by instantiating the library element in the source code. For example, the library element FD1P3IX is a flip-flop that allows synchronous Clear to override Clock Enable. Users can also specify the priority of generic coding by setting the priority of the control signals differently. The following examples demonstrate coding methodologies to help the synthesis tools to set the higher priority of Clock Enable or synchronous LSR.

```
-- VHDL Example of CE over Sync. LSR
...
COUNT8: process(CLK, GRST)
begin
   if (GRST = '1') then
      cnt <= (others => '0');
   elsif (CLK'event and CLK='1') then
   -- CE Over LSR: Clock Enable has higher priority
      if (CKEN = '1') then
            cnt <= cnt + 1;
      elsif (LRST = '1') then
            cnt <= (others =>'0');
      end if;
   end if;
end process COUNT8;
```

```
// Verilog Example of CE over Sync. LSR
...
always @(posedge CLK or posedge GRST)
begin
   if (GRST)
      cnt = 4'b0;
   else
      if (CKEN)
         cnt = cnt + 1'b1;
      else if (LRST)
         cnt = 4'b0;
end...
```

```
-- VHDL Example of Sync. LSR Over CE
...
COUNT8: process(CLK, GRST)
begin
   if (GRST = '1') then
      cnt <= (others => '0');
   elsif (CLK'event and CLK='1') then
   -- LSR over CE: Sync. Set/Reset has higher priority
      if (LRST = '1') then
         cnt <= (others => '0');
      elsif (CKEN = '1') then
         cnt <= cnt + 1;
      end if;
```

```
// Verilog Example of Sync. LSR Over CE
...
always @(posedge CLK or posedge GRST)
begin
   if (GRST)
      cnt = 4'b0;
   else if (LRST)
      cnt = 4'b0;
   else if (CKEN)
      cnt = cnt + 1'b1;
end
...
```

SET / Reset

There are two types of set/reset functions in Lattice Semiconductor FPGAs: Global (GSR) and Local (LSR). The GSR signal is asynchronous and is used to initialize all registers during configuration. It can be activated either by an external dedicated pin or from internal logic after configuration. The local SET/Reset signal may be synchronous or asynchronous. GSR is pulsed at power up to either set or reset the registers depending on the configuration of the device. Since the GSR signal has dedicated routing resources that connect to the set and reset pin of the flip-flops, it saves general-purpose routing and buffering resources and improves overall performance. If asynchronous reset is used in the design, it is recommended to use the GSR for this function, if possible. The reset signal can be forced to be GSR by the instantiation library element. Synthesis tools will automatically infer GSR if all

registers in the design are asynchronously set or reset by the same wire. The following examples show the correct syntax for instantiating GSR in the VHDL and Verilog codes.

```
-- VHDL Example of GSR Instantiation
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity gsr_test is
          port (rst, clk: in std_logic;
          cntout : out std_logic_vector(1 downto 0));
end gsr_test;

architecture behave of gsr_test is
          signal cnt : std_logic_vector(1 downto 0);
begin

          u1: GSR port map (gsr=>rst);

          process(clk, rst)
          begin
                    if rst = '1' then
                              cnt <= "00";
                    elsif rising_edge (clk) then
                              cnt <= cnt + 1;
                    end if;
          end process;
          cntout <= cnt;
end behave;
```

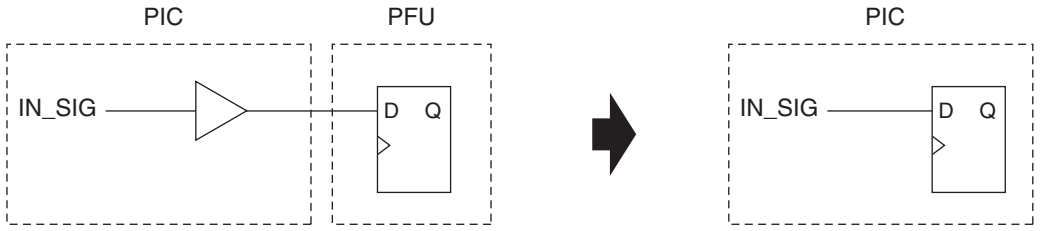
```
// Verilog Example of GSR Instantiation

module gsr_test(clk, rst, cntout);

input clk, rst;
output[1:0] cntout;

reg[1:0] cnt;

GSR u1 (.GSR(rst));

always @(posedge clk or negedge rst)
begin
          if (!rst)
                    cnt = 2'b0;
          else
                    cnt = cnt + 1;
end

assign cntout = cnt;

endmodule
```

Use PIC Features

Using I/O Registers/Latches in PIC

Moving registers or latches into Input/Output cells (PIC) may reduce the number of PFUs used and decrease routing congestion. In addition, it reduces setup time requirements for incoming data and clock-to-output delay for output data, as shown in Figure 13-11. Most synthesis tools will infer input registers or output registers in PIC if possible. Users can set synthesis attributes in the specific tools to turn off the auto-infer capability. Users can also instantiate library elements to control the implementation of PIC resource usage.

Figure 13-11. Moving FF into PIC Input Register



Before Using Input Register **After Using Input Register**

Figure 13-12. Moving FF into PIC Output Register

Before Using Output Register **After Using Output Register**

Inferring Bi-directional I/O

Users can either structurally instantiate the bi-directional I/O library elements, or behaviorally describe the I/O paths to infer bi-directional buffers. The following VHDL and Verilog examples show how to infer bi-directional I/O buffers.

```
-- Inferring Bi-directional I/O in VHDL

library ieee;
use ieee.std_logic_1164.all;

entity bidir_infer is
   port(A, B   : inout std_logic;
        dir    : in std_logic);
end bidir_infer;

architecture lattice_fpga of bidir_infer is
begin
   B <= A when (dir='1') else 'Z';
   A <= B when (dir='0') else 'Z';
end lattice_fpga
```

```
// Inferring Bi-directional I/O in Verilog

module bidir_infer (A, B, DIR);
inout A, B;
input DIR;

assign B = (DIR)  ? A : 1'bz;
assign A = (~DIR) ? B : 1'bz;

endmodule
```

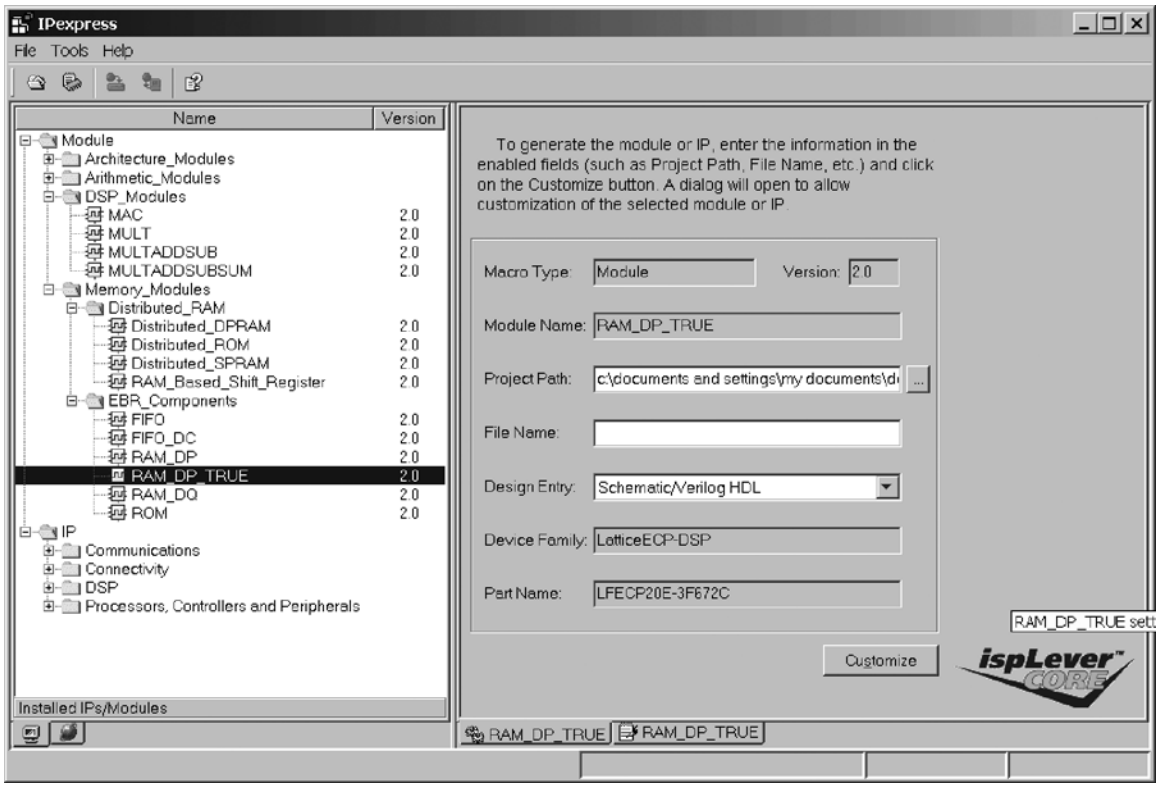
Specifying I/O Types and Locations

Users can either assign I/O types and unique I/O locations in the Preference Editor or specify them as attributes in the VHDL or Verilog source code. The following examples show how to add attributes in the Synplify and Leonardo-Spectrum synthesis tool sets. For a complete list of supported attributes, refer to the HDL Attributes section of the ispLEVER on-line help system.

```
-- VHDL example of specifying I/O type and location attributes for Synplify & Leonardo

    entity cnt is
        port(clk: in std_logic;
             res: out std_logic);
        attribute LEVELMODE: string:
        attribute LEVELMODE of clk : signal is “SSTL2”;
        attribute LOC of clk : signal is “V2”;
        attribute LEVELMODE of res : signal is “SSTL2”;
        attribute LOC of res : signal is “V3”;
    end entity cnt;


-- Verilog example of specifying I/O type and location attributes for Synplify & Leonardo

    module cnt(clk,res);

        input clk /* synthesis LEVELMODE=”SSTL2” LOC=”V2”*/;
        output res /* synthesis LEVELMODE=”SSTL2” LOC=”V3” */;

        ...

        // exemplar begin
            // exemplar attribute clk LEVELMODE SSTL2
            // exemplar attribute clk LOC V2
            // exemplar attribute res LEVELMODE SSTL2
            // exemplar attribute res LOC V3
        // exemplar end

    endmodule
```

Implementation of Memories

Although an RTL description of RAM is portable and the coding is straightforward, it is not recommended because the structure of RAM blocks in every architecture is unique. Synthesis tools are not optimized to handle RAM implementation and thus generate inefficient netlists for device fitting. For Lattice Semiconductor FPGA devices, RAM blocks should be generated through IPexpress as shown in the following screen shot.



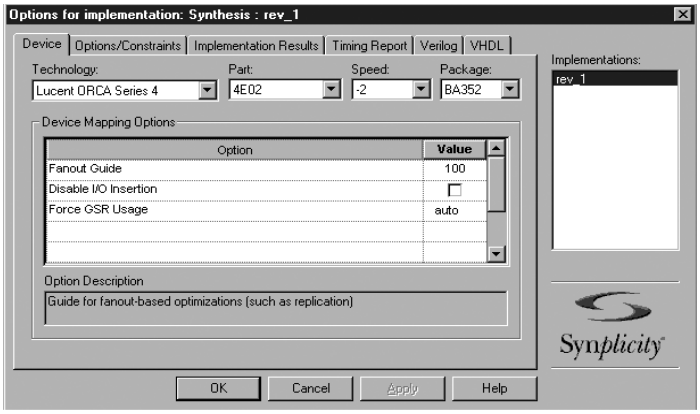
When implementing large memories in the design, it is recommended to construct the memory from the Enhanced Block RAM (EBR) components found in every Lattice Semiconductor FPGA device. When implementing small memories in the design, it is recommended to construct the memory from the resources in the PFU. The memory utilizing resources in the PFU can also be generated by IPexpress.

Lattice Semiconductor FPGAs support many different memory types including synchronous dual-port RAM, synchronous single-port RAM, synchronous FIFO and synchronous ROM. For more information on supported memory types per FPGA architecture, please consult the Lattice Semiconductor FPGA data sheets.

Preventing Logic Replication and Limited Fanout

Lattice Semiconductor FPGA device architectures are designed to handle high signal fanouts. When users make use of clock resources, there will be no hindrance on fanout problems. However, synthesis tools tend to replicate logic to reduce fanout during logic synthesis. For example, if the code implies Clock Enable and is synthesized with speed constraints, the synthesis tool may replicate the Clock Enable logic. This kind of logic replication occupies more resources in the devices and makes performance checking more difficult. It is recommended to control the logic replication in synthesis process by using attributes for high fanout limit.

In the Synplicity® project GUI, under the Implementation Options => Devices tab, users can set the Fanout Guide value to 1000 instead of using the default value of 100. This will guide the tool to allow high fanout signals without replicating the logic. In the LeonardoSpectrum tool project GUI, under Technology => Advanced Settings, users can set the Max Fanout to be any number instead of the default value “0”.



Use ispLEVER Project Navigator Results for Device Utilization and Performance

Many synthesis tools give usage reports at the end of a successful synthesis. These reports show the name and the number of library elements used in the design. The data in these reports do not represent the actual implementation of the design in the final Place and Route tool because the EDIF netlist will be further optimized during Mapping and Place and Route to achieve the best results. It is strongly recommended to use the MAP report and the PAR report in the ispLEVER Project Navigator tool to understand the actual resource utilization in the device. Although the synthesis report also provides a performance summary, the timing information is based on estimated logic delays only. The Place & Route TRACE Report in the ispLEVER Project Navigator gives accurate performance analysis of the design by including actual logic and routing delays in the paths.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

PCB Layout Recommendations for BGA Packages

September 2010 Technical Note TN1074

Introduction

As Ball Grid Array (BGA) packages become increasingly popular and become more populated across the array with higher pin count and smaller pitch, it is important to understand how they are affected by various board layout techniques. This document provides a brief overview of PCB layout considerations when working with BGA packages. It outlines some of the most common problems and provides tips for avoiding them at the design stage. A key challenge of adopting fine-pitch (0.8 mm or less) BGA packages is the design of a route fanout pattern that maximizes I/O utilization while minimizing fabrication cost. This technical note provides an overview of PCB design examples provided by Lattice Semiconductor.

For more information and design examples see the PCB Design Support page at the Lattice Semiconductor web site (www.latticesemi.com/support/pcbdesignsupport.cfm).

BGA Board Layout Recommendations

All Lattice BGA packages have Solder Mask Defined (SMD) pads. In order to evenly balance the stress in the solder joints, Lattice recommends PCB solder pads be SMD with dimensions as similar to the applicable BGA as possible. If Non Solder Mask Defined Pads (NSMD) are used, the optimum pad dimensions will result in an equivalent surface contact area on the PCB as on the BGA component.

Table 14-1. Lattice Semiconductor SMD/NSMD Pad Recommendations[1]

	0.4 mm Ball Pitch	0.5 mm Ball Pitch	0.8 mm Ball Pitch	1.0 mm Ball Pitch			1.27 mm Ball Pitch
	64, 132 ucBGA	56, 64, 100, 132, 144 csBGA	49, 100, 256 caBGA	100 fpBGA, 256 ftBGA (Option 1[2])	256 ftBGA (Option 2[3]), 324 ftBGA, 144, 208, 256, 388, 416, 484, 516, 672, 676, 900, 1152, 1156 fpBGA	1020, 1152, 1704 Organic fcBGA	272, 388 PBGA, 256, 320, 352 SBGA
Nominal BGA package pad opening diameter (mm) - SMD	0.20	0.27	0.40	0.40	0.45	0.50	0.60
PCB Solder Mask Defined (SMD) Pad Recommendations							
Optimum solder mask opening	0.22	0.30	0.40	0.40	0.45	0.50	0.60
PCB Non Solder Mask Defined (NSMD) Pad Recommendations							
Optimum solder land diameter	0.17	0.25	0.35	0.35	0.40	0.45	0.55

1. These Lattice-recommended PCB design values will result in optimum Board Level Reliability (BLR) performance for each corresponding package. Those who use PCB design values which deviate from these recommendations should understand that the BLR performance may be reduced.
2. ispMACH 4000, MachXO, LatticeXP2.
3. LatticeECP3.

Table 14-2. Lattice Semiconductor BGA Package Types

Package Type	Description
PBGA	**Plastic BGA** with 1.27 mm solder ball pitch. Die up configuration.
fpBGA	**Fine Pitch BGA** – Plastic BGA with 1.0 mm solder ball pitch. Die up configuration.
ftBGA	**Fine Pitch Thin BGA** – Thin plastic BGA with 1.0 mm solder ball pitch. Die up configuration.
caBGA	**Chip Array BGA** – Plastic BGA with 0.8 mm solder ball pitch. Die up configuration.
csBGA	**Chip Scale BGA** – Plastic BGA with 0.5 mm solder ball pitch. Die up configuration.
fcBGA	**Flip-Chip BGA** with 1.0 mm solder ball pitch. Die down configuration. May have a ceramic or plastic substrate.
SBGA	**Super BGA** – Similar to PBGA, but with an integrated heatsink plate. This package has 1.27 mm solder ball pitch and die down configuration. SBGA packages offer enhanced thermal dissipation capability.
fpSBGA	**Fine Pitch SBGA** – Super BGA with 1.0 mm solder ball pitch. Die down configuration.
ucBGA	**Ultra Chip BGA** – Saw singulated plastic ball grid array package with 0.4 mm ball pitch.

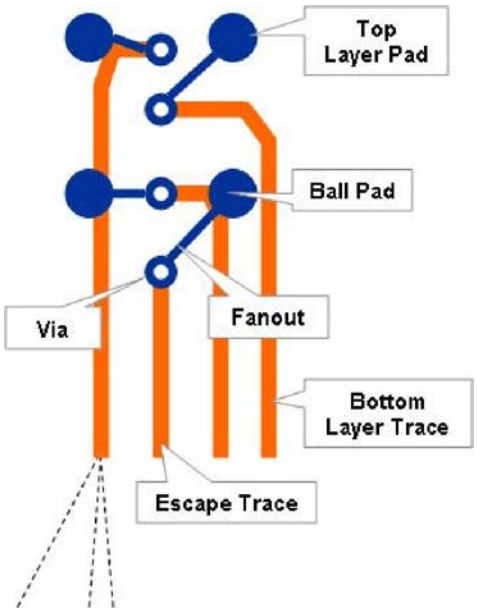
BGA Breakout and Routing Examples

Lattice provides several resources and different design implementations that show BGA breakout and routing of various fine-pitch BGA packages. Different stack up and layer counts are also used to show a range of design rules and fabrication costs. It is important to consult with your board fabrication and assembly houses as to the most economical and reliable process for your application.

Currently there is a wide choice of BGAs from Lattice, with many devices offered in multiple packages and pitches of BGA densities as well as non-BGA options such as TQFP, QFN and others. The BGA pitch or “center to center” ball dimensions include, 1.00 mm BGAs, space-saving 0.5 mm pitch chip scale BGA and 0.4mm pitch ultra chip scale BGA packages. Fine pitch packages offer advantages and disadvantages alike. Finer pitch means that the trace and space limits will have to be adjusted down to match the BGA. Many times a design can get away with small traces underneath the BGA then fan out with a slightly larger trace width. The PCB fabrication facility will need to be aware of your design objectives and check for the smallest trace dimensions supported. Smaller traces take more time to inspect, check and align etc. Etching needs to be closely monitored when trace and space rules reach their lower limit.

The combination of fanout traces, escape vias, and escape traces that allow routing out from under the BGA pin array to the perimeter of the device are collectively referred to as the “BGA breakout”. The fanout pattern will arrange the breakout via, layer, and stack-up to maximize the number of I/Os that can be routed. Fanout patterns are an important consideration for devices over 800 pins and can be follow polar (north/south/east/west) or layer-biased directions. (Source: *BGA Breakouts and Routing*, Charles Pfeil, Mentor Graphics).

Figure 14-1. BGA Breakout Routing Terms



Lattice provides BGA breakout and routing examples for various fine pitch packages (www.latticesemi.com/support/pcbdesignsupport/bgabreakoutroutingexample/index.cfm). Each package example is built to comply with IPC7351 (www.ipc.org) specifications and nomenclature conventions. Some examples include different layout options depending on design and cost goals. For example, the 256-ball chip array BGA (BN256) examples demonstrate a design with fully-utilized I/Os, fine trace width and pitch, on a 6-layer PCB stack-up and a less expensive design with relaxed design rules, and fewer I/O pads routed, on a 4-layer PCB stack up.

Table 14-3. Package Layout Example Summary

Package	Example #	Pitch (mm)	Signal/ Power Layers	Trace/ Width-Space (mm)	Ball Pad (mm)	Ball Mask (mm)	Via Pad (mm)	Via Drill (mm)
MN64	1	0.5	6	.100/.100	.23	.33	.30	.125
UMN64	1	0.4	6	.100/.100	.18	.28	.25	.10
MN100	1	0.5	4	.085/.085	.23	.38	.45	.20
	2	0.5	4	.100/.100	.23	.38	.45	.20
MN132	1	0.5	4	.085/.085	.23	.38	.40	.15
	2	0.5	4	.100/.100	.23	.38	.40	.15
MN144	1	0.5	6	.100/.100	.23	.33	.30	.125
	2	0.5	4	.100/.100	.23	.38	.30	.125
BN256	1	0.8	6	.100/.100	.35	.50	.40	.125
	2	0.8	4	.100/.100	.35	.50	.40	.15

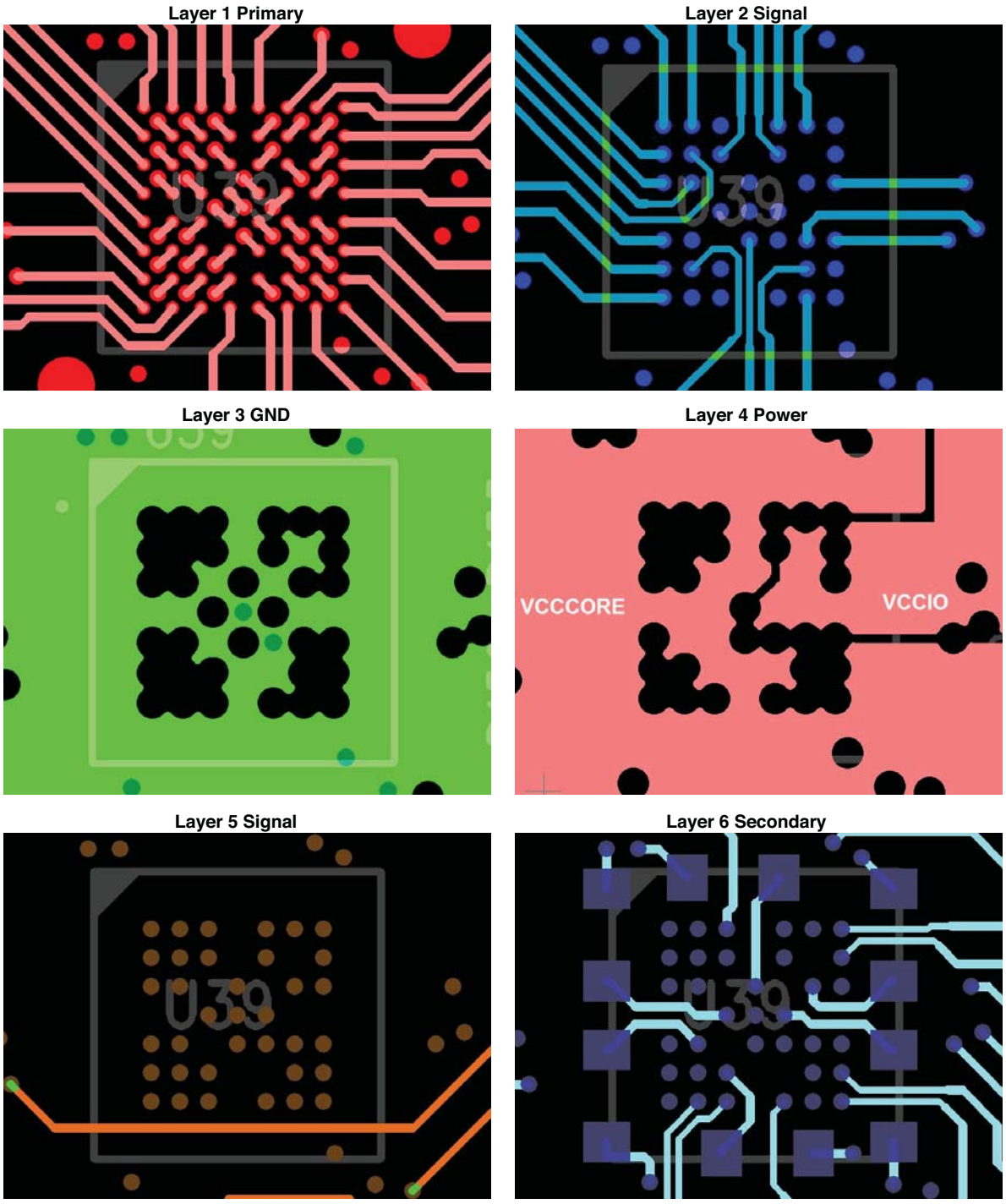
For mechanical dimension details on packages, see the Lattice Package Diagrams document.

In order to show how some of the routing challenges are solved, examples are provided for fine-pitch BGA packages from the MachXO™ and the ispMACH® 4000ZE families. Principles for these apply to other Lattice BGA packaged products.

64-ball csBGA BGA Breakout and Routing Example

This example places an ispMACH 4000ZE CPLD in a 5x5 mm, 0.5 mm pitch, 64-ball csBGA package (LC4064ZE-MN64) in an 6-layer stack up with maximum I/O utilization.

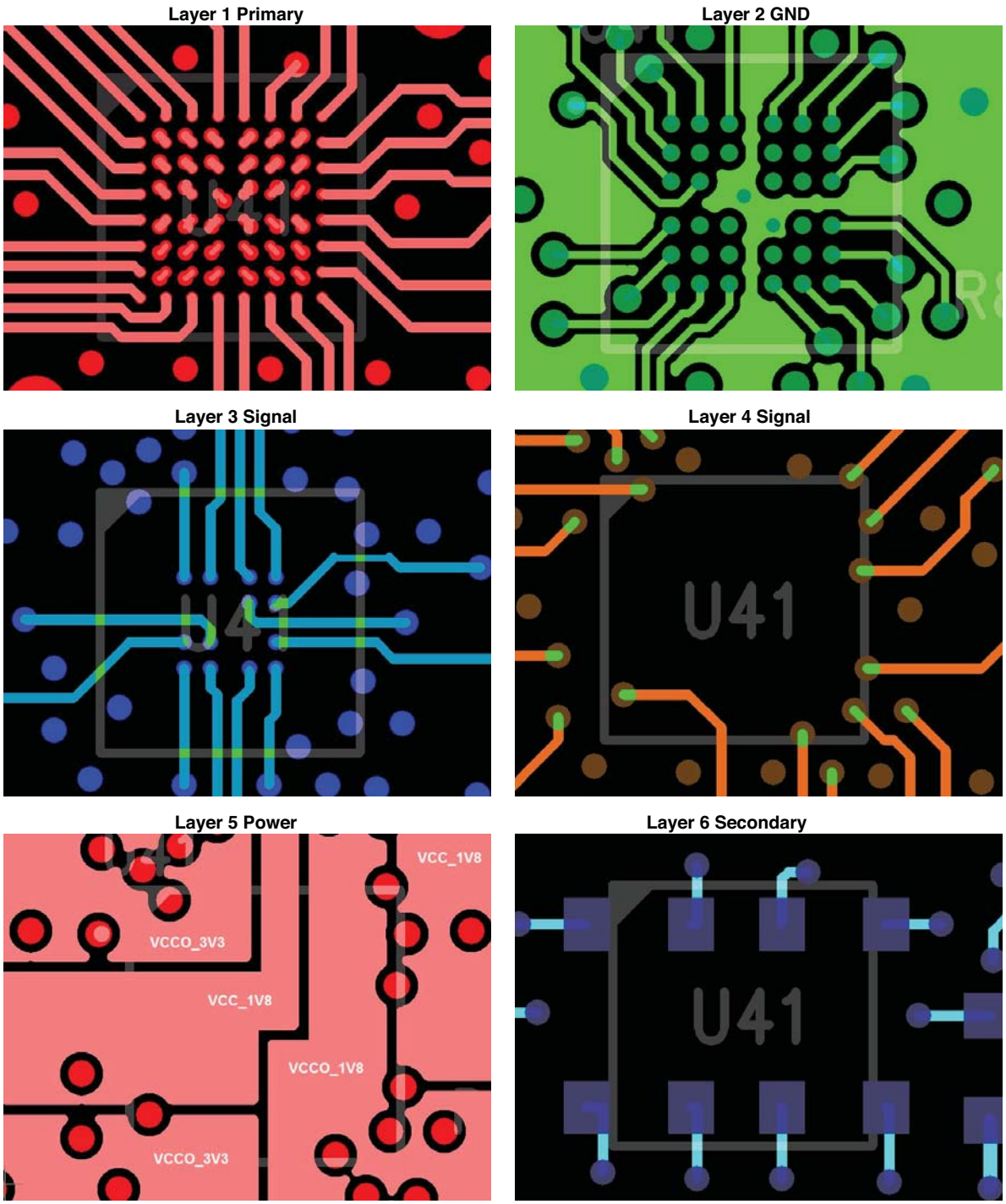
Figure 14-2. CAM Artwork Screen Shots 64-Ball csBGA



64-ball ucBGA BGA Breakout and Routing Example

This example places an ispMACH 4000ZE CPLD in a 4x4 mm, 0.4 mm pitch, 64-ball ucBGA package (LC4064ZE-UMN64) in an n-layer stack up with maximum I/O utilization. This example demonstrates a modified dogbone fanout technique to get access to all pins yet limiting number of layers and via schedules, while setting up layers to use reference planes for high-speed signal traces.

Figure 14-3. CAM Artwork Screen Shots, 64-Ball ucBGA



100-ball csBGA BGA Breakout and Routing Examples

These examples place a MachXO PLD in a 8x8 mm, 0.5 mm pitch, 100-ball csBGA package (LXMXO640-M100/MN100) into two fabrication scenarios. One for a 6-layer stack up with maximum I/O utilization and a 4-layer with about 15% fewer I/Os. The 4-layer (Example #2) design makes maximum use of via and trace geometry to reduce layer count and ease fabrication while still maintaining high I/O usage.

Figure 14-4. CAM Artwork Screen Shots, Example #1, 100-ball csBGA

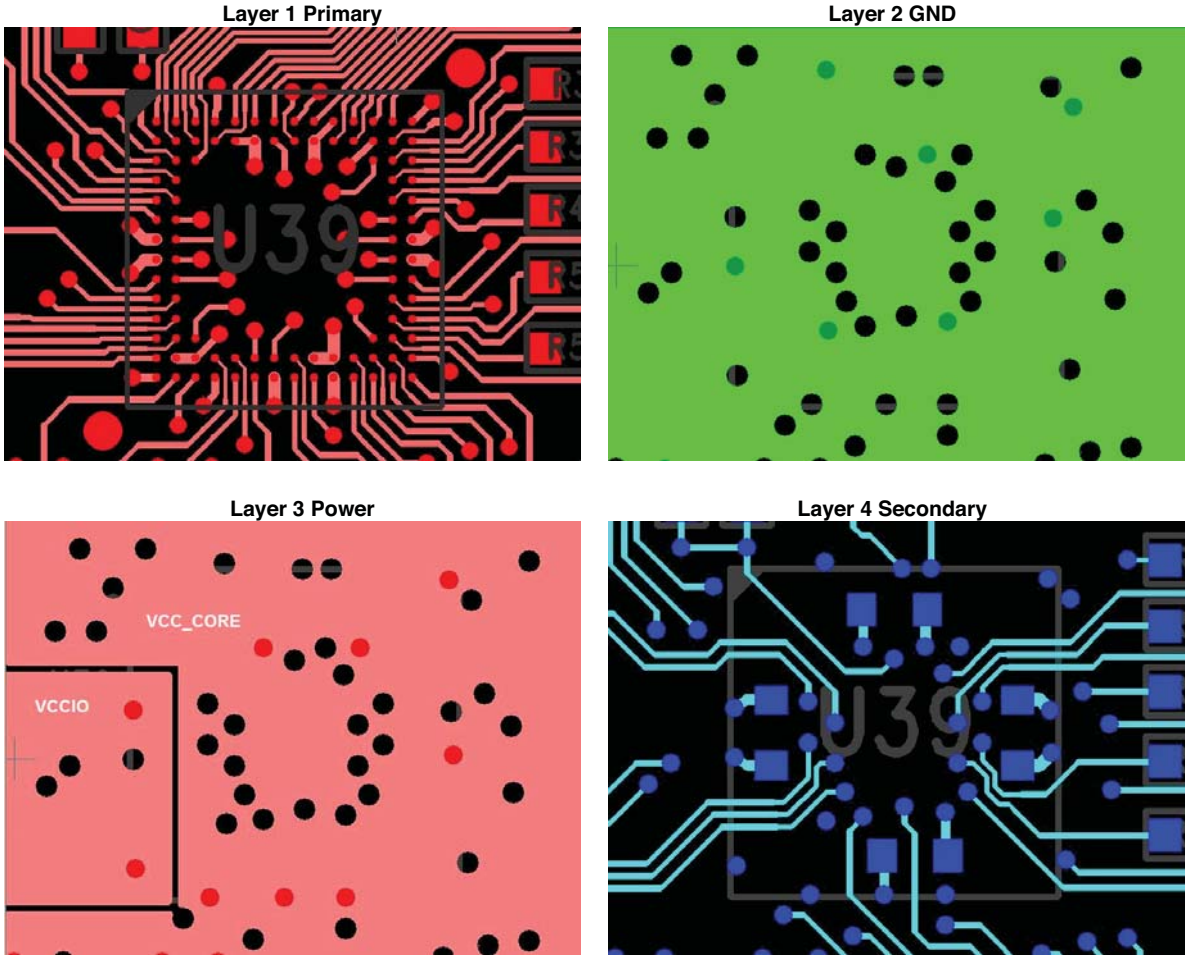
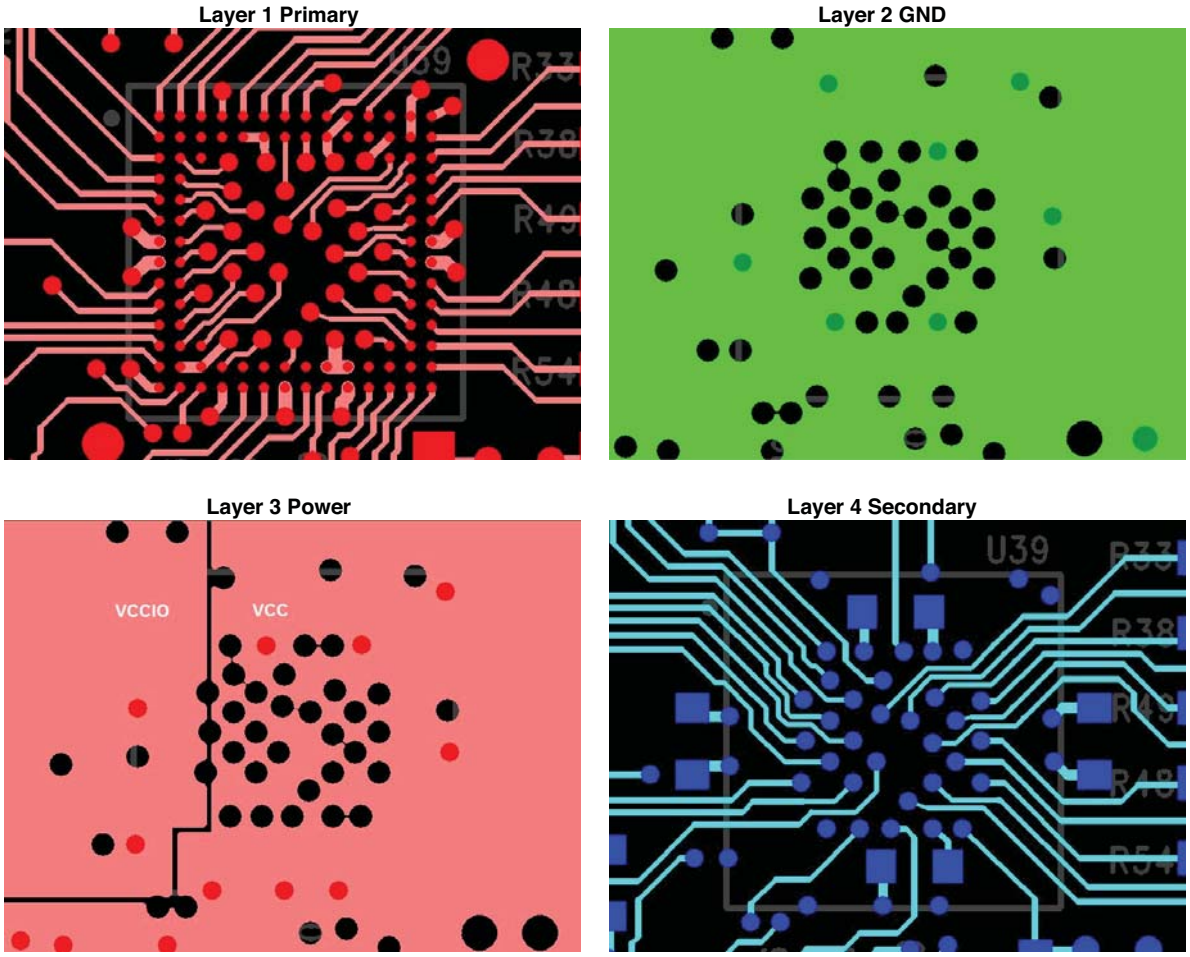


Figure 14-5. CAM Artwork Screen Shots, Example #2, 100-ball csBGA



132-ball csBGA BGA Breakout Examples

These examples place a MachXO PLD in a 8x8 mm, 0.5 mm pitch, 132-ball csBGA package (LCMXO640-M132/MN132) into two fabrication scenarios. One for a 6-layer stack up with maximum I/O utilization and a 4-layer with about 15% fewer I/Os.

Figure 14-6. CAM Artwork Screen Shots, Example #1, 132-ball csBGA

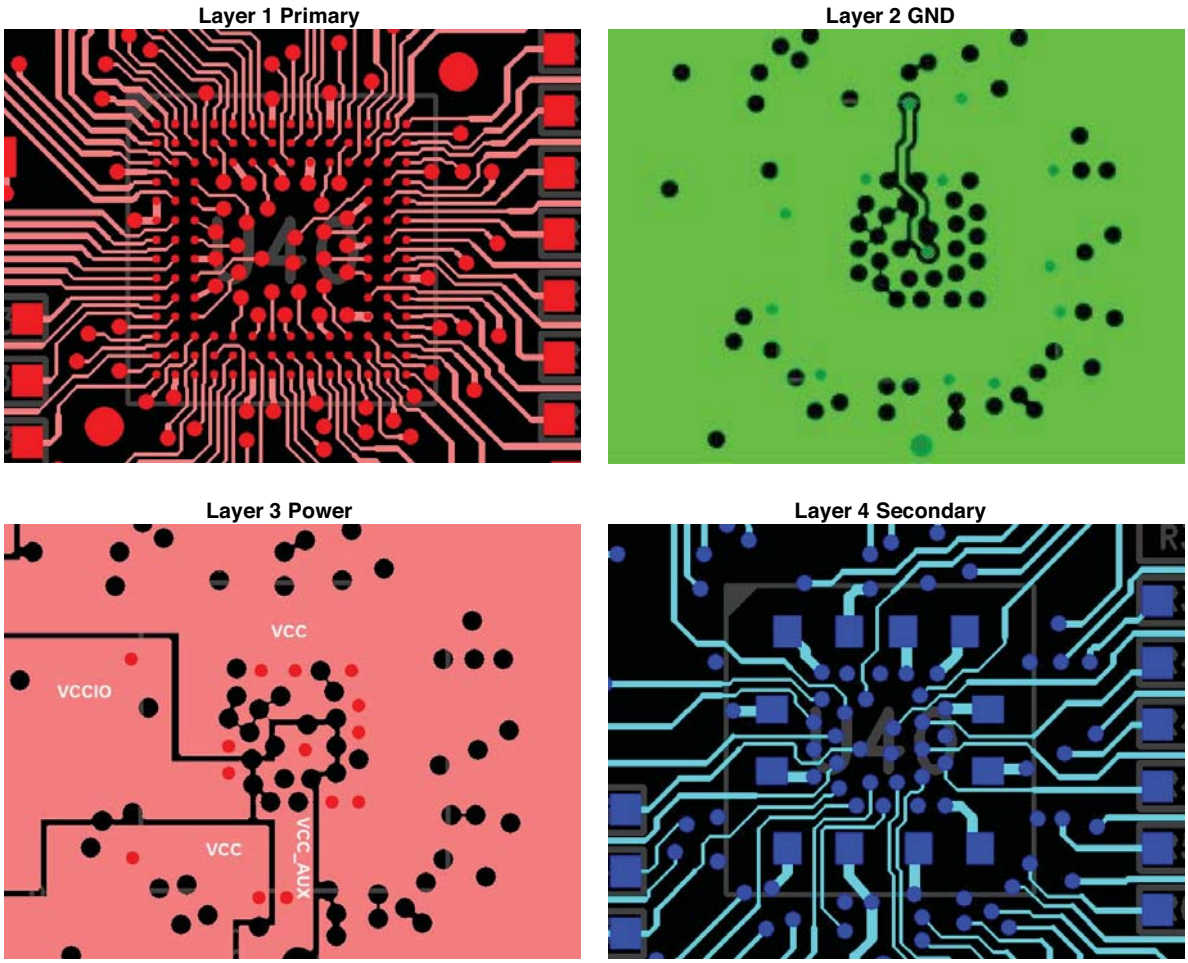
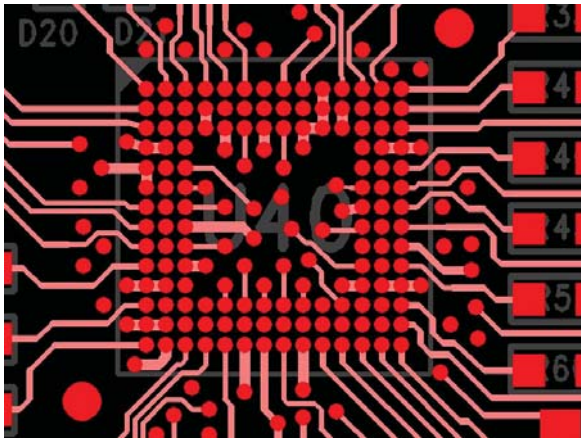
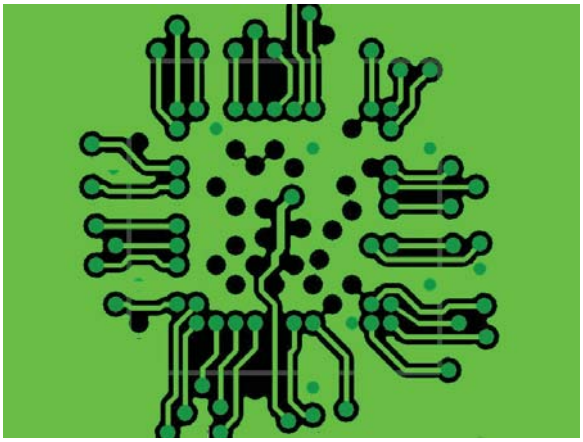


Figure 14-7. CAM Artwork Screen Shots, Example #2, 132-ball csBGA

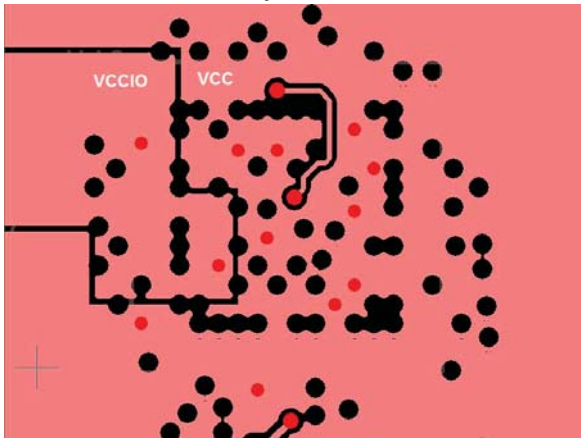
Layer 1 Primary



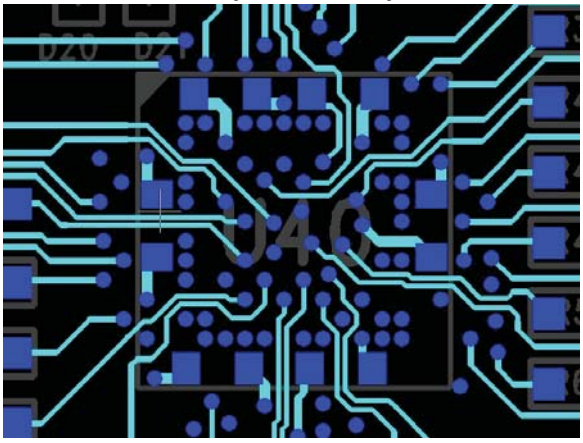
Layer 2 GND



Layer 3 Power



Layer 4 Secondary



144-ball csBGA BGA Breakout Examples

These examples place an ispMACH 4000ZE in a 7x7 mm, 0.5 mm pitch, 144-ball csBGA package (LC4256ZE-MN144) into two fabrication scenarios. One for a 6-layer stack up with maximum I/O utilization and a 4-layer with about 5% fewer I/Os. The 6-layer (Example #1) design avoids uses of micro vias and takes advantage of removed pads on inner layers to route all pins out to 6 layers with good layer structure for high-speed signal integrity.

Figure 14-8. CAM Artwork Screen Shots, Example #1, 144-ball csBGA

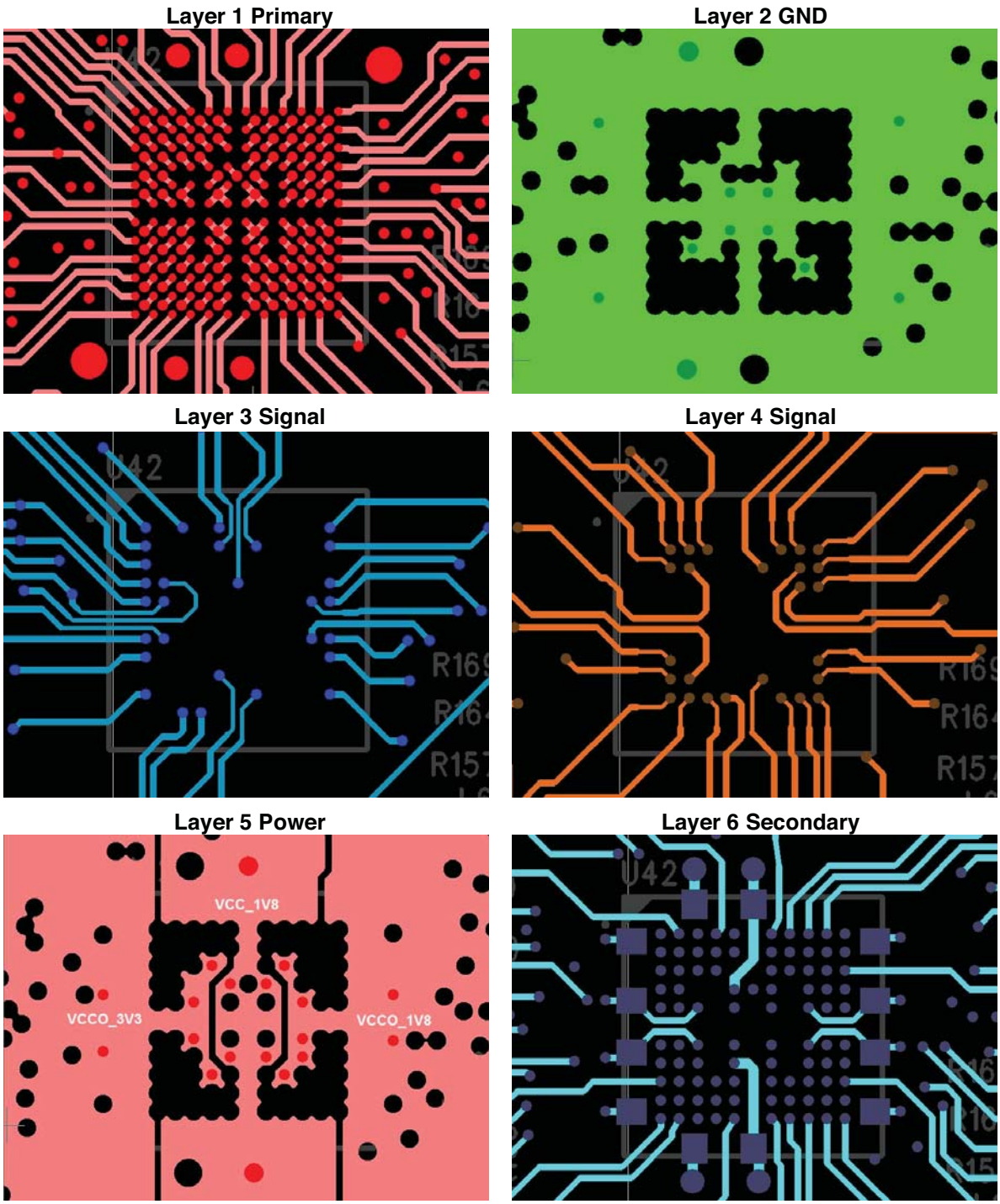
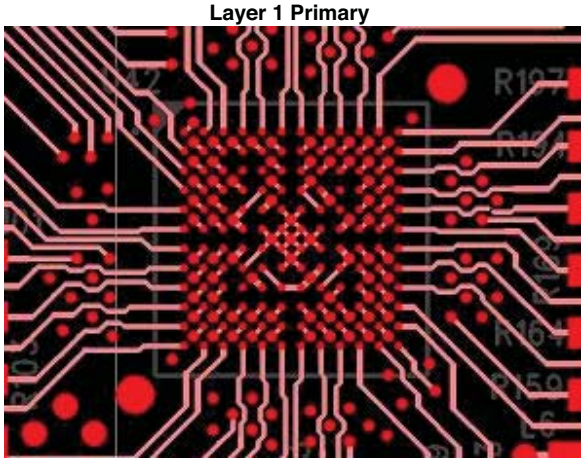
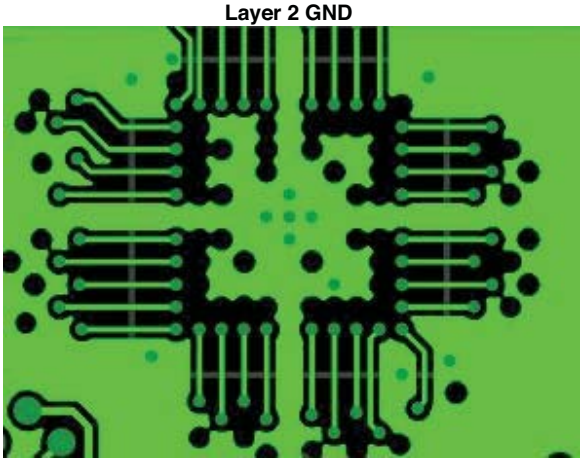


Figure 14-9. CAM Artwork Screen Shots, Example #2, 144-ball csBGA

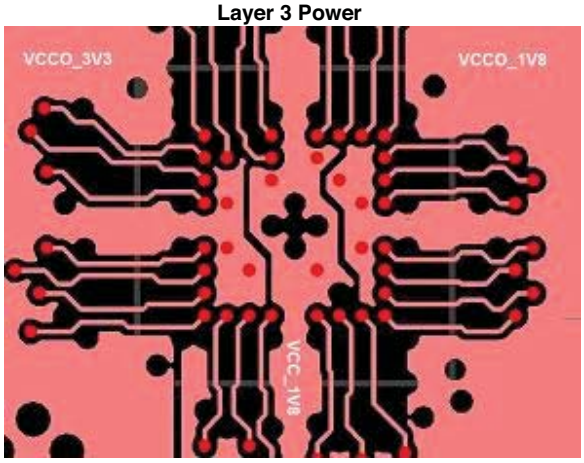
Layer 1 Primary



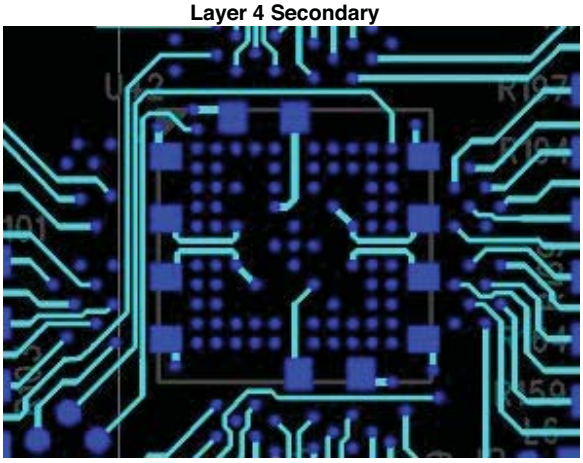
Layer 2 GND



Layer 3 Power



Layer 4 Secondary

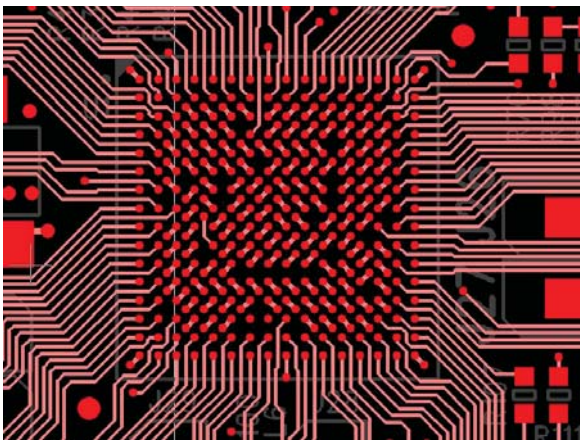


256-ball caBGA BGA Breakout Examples

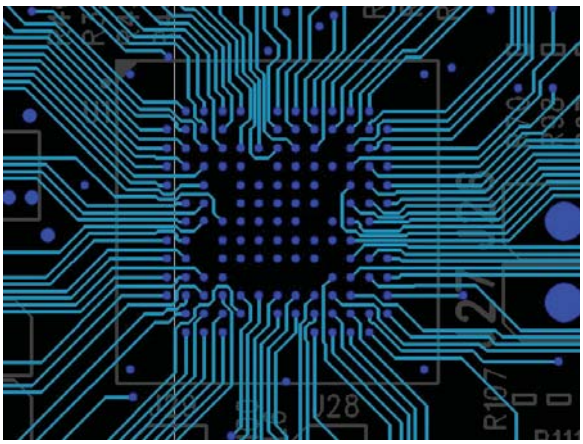
This BGA breakout and routing example places a MachXO PLD in a 14x14 mm, 0.8 mm pitch, 256-ball caBGA package (LCMXO2280-B256/BN256) into two fabrication scenarios. One for a 6-layer stack up with maximum I/O utilization and a 4-layer with about 10% fewer I/Os. The 6-layer design (Example #1), demonstrates the best use of mechanically drill blind vias to place caps near power pins to minimize layers.

Figure 14-10. CAM Artwork Screen Shots, Example #1, 256-Ball caBGA

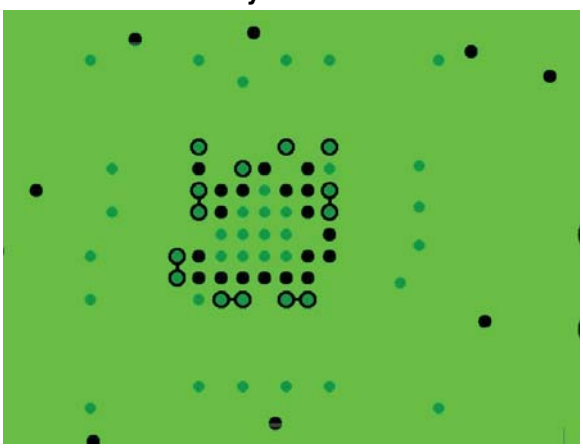
Layer 1 Primary



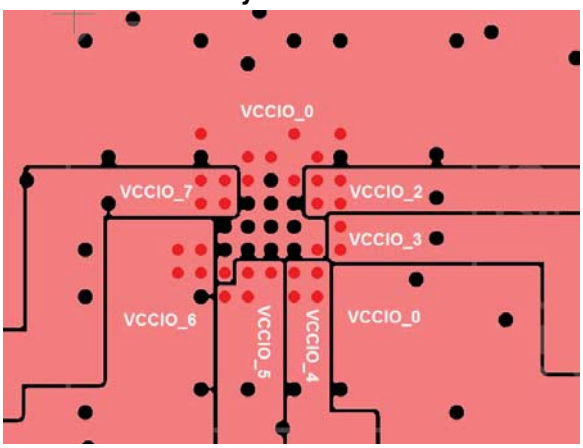
Layer 2 Signal



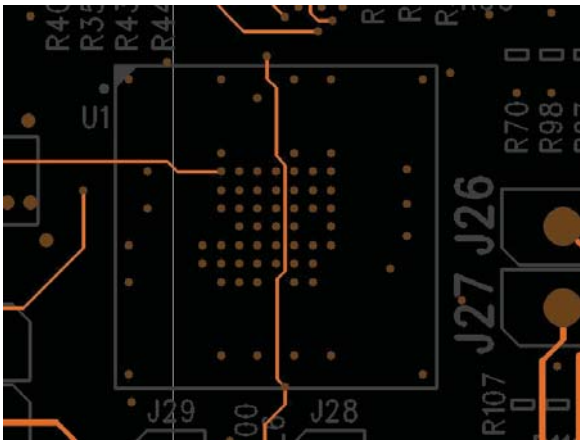
Layer 3 GND



Layer 4 Power



Layer 5 Signal



Layer 6 Secondary

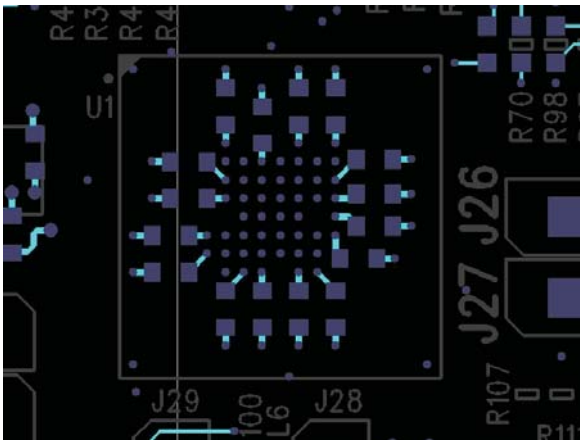
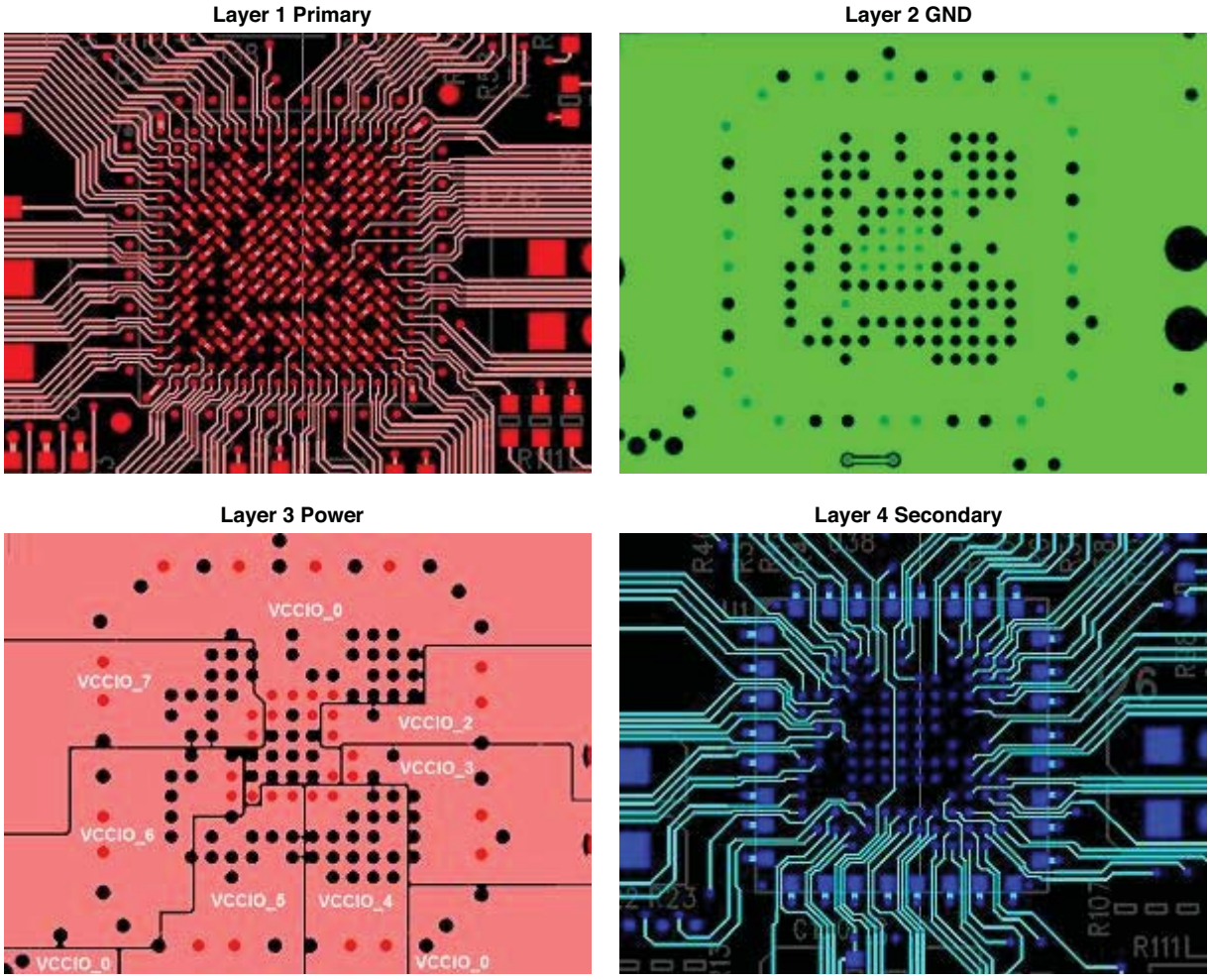


Figure 14-11. CAM Artwork Screen Shots, Example #2, 256-ball caBGA



PCB Fabrication Cost and Design Rule Considerations

PCB fabrication cost is a key consideration for many electronics products. By reviewing the IC device package ball density and pitch, I/O signal requirements of your application, and the manufacturing constraints of your PCB fabrication facility you can better weigh the trade-offs between design decisions.

Choosing the best package for your application involves answering a few questions:

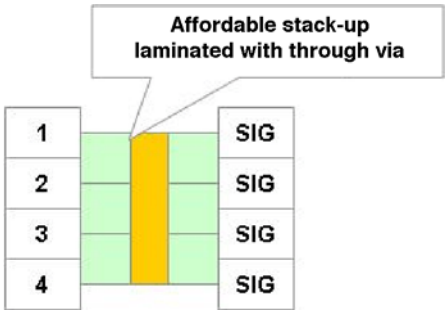
- What is the driving factor in the application? The smallest possible form factor or a low PCB cost?
- How many I/O signals does the application require?
- What PCB layer stack up will provide the best I/O density within budget?
- What layout design rules does the printed-circuit board (PCB) vendor support?
- How many PCB layers does the budget allow?

As the ball pitch becomes smaller with each new BGA generation, new PCB fabrication techniques and signal via type have been developed to handle the complexities. Micro vias, laser vias, filled, buried and blind vias, even buried and plated over vias. Complex boards use a combination of most of these.

Stack-up types, ordered by cost, high-to-low include:

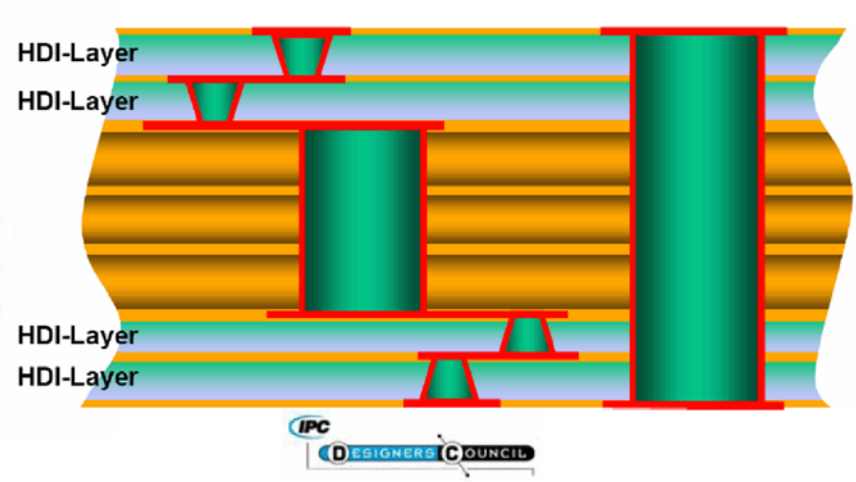
- High-Density Interconnect (HDI) build up with micro vias
- Laminated with blind and buried vias
- Laminated with through vias

Figure 14-12. Stack-up Example



HDI is a “sandwich” with older-style larger geometry in the middle with fully drilled through holes and then a stack of fine geometry of blind, buried or mixed via, laminated both on the top and bottom of the middle stack-up. The laminated layers are thinner than traditional layers and allow finer drilling technology. Staggered micro vias allow vias within close tolerance or connected to a BGA pad to go down to the next layer or more to route away for escape routing or underneath the BGA device for further interconnect. HDI type interconnect is used on complex boards and takes extra steps in the processing flow due to special drilling, plating an laminations. It is a mix of older technology mixed with newer technology that results in a board that is highly routable.

Figure 14-13. HDI Stack-up with Staggered Micro Vias



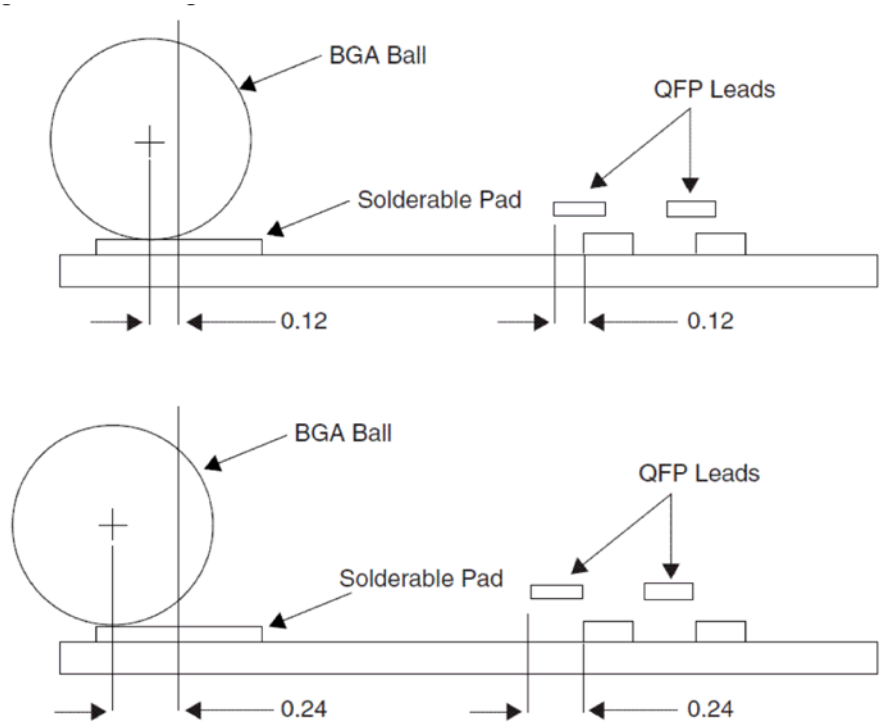
Advantages and Disadvantages of BGA Packaging

As pin counts increase and board space becomes more valuable, it is important to place as many circuits per square inch as possible. BGA offer the best I/O density for a given PCB area. Lattice offers a range of packages from a 4x4 mm 64-ball csBGA to a 33x33 mm 1704-ball fcBGA.

One of the greatest advantages of BGA packaging is that it can be supported with existing placement and assembly equipment. BGAs also offer significantly more misalignment tolerance and less susceptibility to co-planarity issues. Even if the solder paste is misaligned or the device is slightly offset, the BGA will self-center during the

reflow process. This is due to the surface tension of the solder and flux in its molten state pulling each ball into the center of the pad.

Figure 14-14. Misalignment of BGA Balls vs. QPF Leads



Controlling the oven re-flow profile is one of the most important assembly parameters for consistent and reliable BGA placement. Profiles are typically tested on a pre-run. One or two panels are run to dial in the process, then visual and X-ray inspection equipment are used for verification.

BGA packages present numerous benefits previously unobtainable in surface mount packaging technology. BGAs provide higher pin counts in a much smaller area than was previous possible. No longer is the package design limited to connections along the periphery on the outside quadrants of the package edge like a PQFP or TQFP outline. Fully populated ball grid arrays with pitches as small as 0.4 mm are available.

Some BGA devices are arranged with de-populated interconnect near or around the center. These are dependent on the die size and number of pins. The area void of interconnect in the middle of the array has some advantages, it can be used for escape routing vias or tying directly to the ground or power planes.

Although the packages can be quite complex and densely populated, all of these packages receive strict quality and reliability testing and are widely accepted today by designers and PCB fabrication/assembly houses. All of this is due to advances in equipment and technology that have allowed a smooth transition into the assembly flow.

BGA Package Test and Assembly

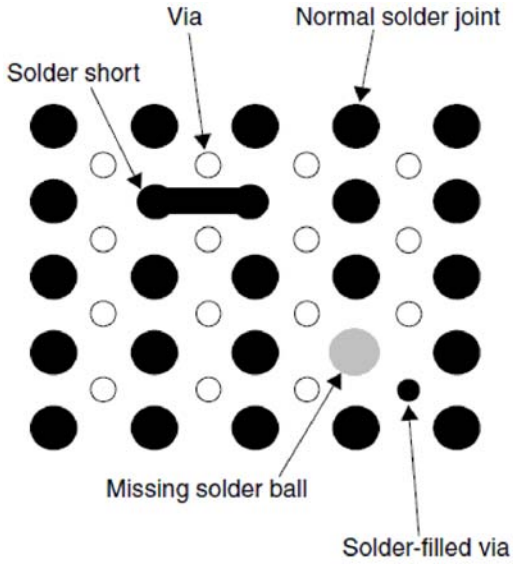
How can a pad/ball/pin be tested that can't be seen? All connections are hidden under the substrate at the ball interconnect, making it impossible to directly probe or test. To address this limitation, Lattice programmable devices provide JTAG, BSCAN, and boundary scan cells that allow electronic test and continuity of each pin with a boundary scan tester. This can be embedded into the system itself or driven externally from a high-speed test head. The boundary scan can test the pins or board for simple continuity tests or full functional test by shifting in test patterns through the JTAG port.

For debug or prototype boards it may be necessary to place test points, open vias, or pads to have access to a given set of pins in order to drive, over-drive or observe a given set of signals. These can be very small, as many

pogo pin type probes are extremely small and can handle GHz range DC frequency. Zero ohm resistors are also commonly used in first-run boards as a way to gain access to a pad or pin.

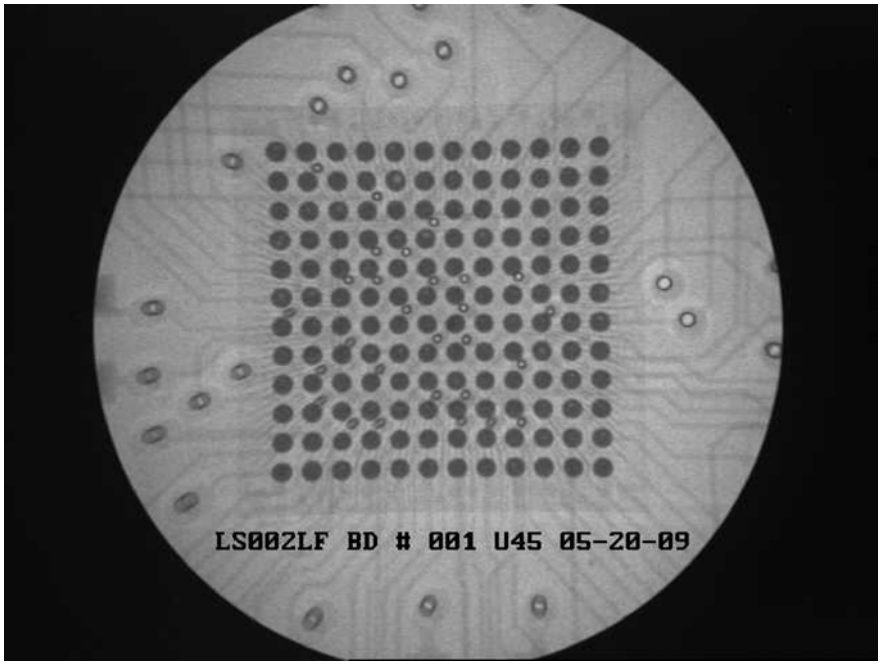
After assembly, BGA solder point quality and integrity is visually inspected with X-ray technology as part of the fabrication process. A special X-ray machine can look through the plastic package, substrate and silicon to directly view the BGA solder balls, vias, traces and pads.

Figure 14-15. Example of How Defects May Appear in an X-Ray Image



The X-ray image in Figure 14-16 shows proper alignment; no voids or defects are noted. Balls, vias and traces are visible.

Figure 14-16. X-Ray Inspection Plot of ispMACH 4000ZE 144-ball csBGA (Photo Courtesy of CEM, Ltd., www.cemltd.com)



PCB cross-sectioning is another method used to verify BGA and PCB quality and reliability. After a new process has been developed or changed or when qualifying a new vendor, it is a good practice to get physical information from the vendor on their BGA reflow. When trace/space and drill or laser tolerances are nearing their limits, board yield can be as be as low as 50% for the bare board fab. Cross-sections give you a good idea if the process is correct but do not guarantee each batch or each board design will behave the same way due to layout dimensions, thermal issues, flux/paste and alignment, etc.

Figure 14-17 shows a BGA cross-section that uses a non-soldermask over bare copper-defined pad. (NSMD) pad.

Figure 14-17. BGA Cross-Section

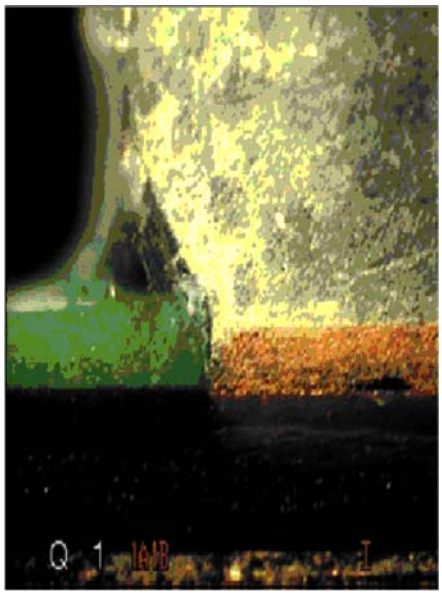
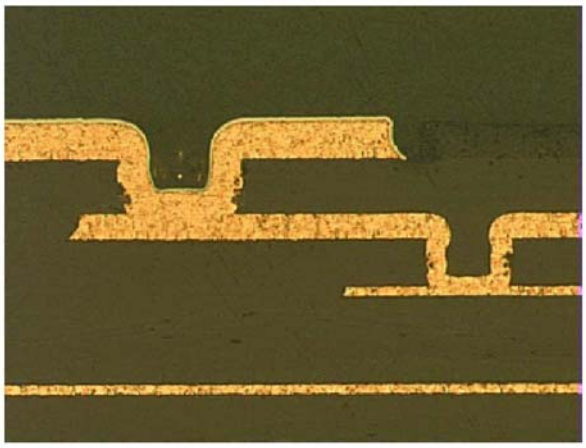


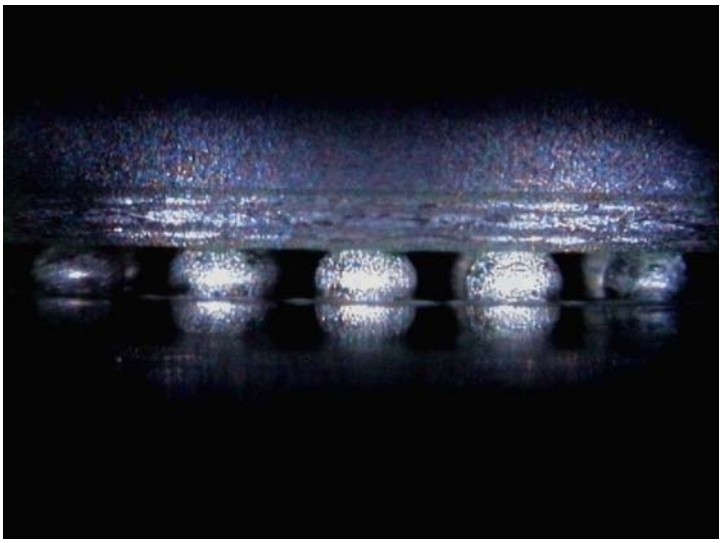
Figure 14-18 shows “offset” micro via stack routing between layers.

Figure 14-18. Cross-Section of Micro Vias



High-resolution video cameras are used for edge inspection to verify ball seating, distortion, solder wetting, flow, contaminates, etc. Figure 14-19 is a video view of a side/edge shot looking at BGA balls soldered down to the ispMACH 4000ZE Pico Evaluation Board (www.latticesemi.com/4000ze-pico-kit), an FR4 4-layer PCB.

Figure 14-19. Edge View Camera Inspection



In the photos samples above, trained technicians and computer camera recognition equipment are used for inspection of the X-ray results, looking for voids, shorts, missing connections, contaminants, alignment or other gross failure mechanisms. For example, in Figure 14-19, the BGA ball connections appear to be squashed downward, with mild distortion, insuring that proper oven profile temperatures were achieved.

These technologies help in the successful placement and long term use of BGAs in the industry's latest products. Further advancements have been made in material content to conform to environmental issues, toxic materials and recycling. Another issue that relates to board design is the physical silkscreen logos and information related to recycling, lead content and other hazardous waste components, strict adherence must be paid to these requirements. Although a documentation and silkscreen issue, it can become a challenge to fit this information on the board in some cases due to component population and must be accounted for in overall board real estate.

PCB Design Support

Lattice provides a collection of PCB design resources at www.latticesemi.com/support/pcbdesignsupport.cfm including schematic libraries, PCB CAM viewers, technical notes, and BGA breakout and routing examples.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
January 2005	01.0	Initial release.
November 2005	01.1	Figures updated.
June 2006	01.2	Removed NSMD content.
September 2006	01.3	Added note to BGA Board Layout Recommendations table.
		Reformatted BGA Package Types section in tabular format.
March 2008	01.4	Revised recommended Solder Mask Defined and Non Solder Mask Defined PCB solder pad dimensions to match industry standards.
May 2009	01.5	Updated BGA Board Layout Recommendations table and BGA Package Types table for 0.4 mm pitch ucBGA package.
February 2010	01.6	Edits to most sections and additional links and graphics added for each example.
March 2010	01.7	Replaced Lattice BGA Naming Conventions table with Lattice Semiconductor BGA Package Types table and SMD/NSMD Pad Recommendations table.
August 2010	01.8	Lattice Semiconductor SMD/NSMD Pad Recommendations table: Specified nominal Solder Mask Opening for each Lattice BGA package, clarified recommended Solder Mask Opening and Solder Pad Diameters and added cautionary note.
September 2010	01.9	Lattice Semiconductor SMD/NSMD Pad Recommendations table - Added 64 csBGA to 0.5mm pitch column.

Section III. MachXO Family Handbook Revision History

MachXO Family Handbook Revision History

December 2010 **Data Sheet**

Revision History

Date	Handbook Revison Number	Change Summary
July 2005	01.0	Initial release.
October 2005	01.1	MachXO Family Data Sheet updated to version 01.1.
		Technical note TN1086 updated to version 01.1.
October 2005	01.2	Technical note TN1092 updated to version 01.1.
		Technical note TN1089 updated to version 01.1.
		Technical note TN1086 updated to version 01.1.
		Technical note TN1008 updated to version 02.1.
		Technical note TN1074 updated to version 01.1.
November 2005	01.3	MachXO Family Data Sheet updated to version 01.2.
June 2006	01.4	MachXO Family Data Sheet updated to version 02.1.
		Technical note TN1086 updated to version 01.2.
September 2006	01.5	MachXO Family Data Sheet updated to version 02.2.
		Technical note TN1074 updated to version 01.2.
September 2006	01.6	Technical note TN1074 updated to version 01.3.
November 2006	01.7	MachXO Family Data Sheet updated to version 02.3.
		Technical note TN1089 updated to verson 01.2.
		Technical note TN1092 upated to version 01.3.
December 2006	01.8	MachXO Family Data Sheet updated to version 02.4.
February 2007	01.9	MachXO Family Data Sheet updated to version 02.5.
November 2007	02.0	MachXO Family Data Sheet updated to version 02.7.
		Technical note TN1086 updated to version 01.3.
		Technical note TN1090 updated to version 01.1.
		Technical note TN1091 updated to version 01.4.
		Technical note TN1092 updated to version 01.4.
June 2009	02.1	MachXO Family Data Sheet updated to version 02.8.
		Technical note TN1074 updated to version 01.5.
March 2010	02.2	Technical note TN1074 updated to version 01.6.
		Technical note TN1089 updated to version 01.3.
March 2010	02.3	Technical note TN1074 updated to version 01.7.
September 2010	02.4	Technical note TN1074 updated to version 01.9.
		Technical note TN1091 updated to version 01.5.
December 2010	02.5	MachXO Family Data Sheet updated to version 02.9.
		Technical note TN1086 updated to version 01.4.
		Technical note TN1089 updated to version 01.4.
		Technical note TN1092 updated to version 01.5.

Note: For detailed revision changes, please refer to the revision history for each document.