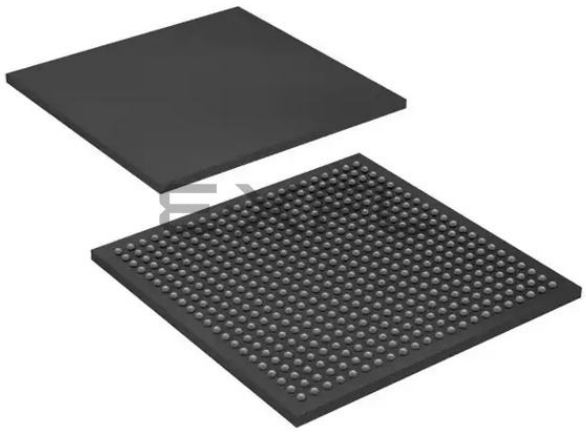


Welcome to E-XFL.COM

Understanding Embedded - FPGAs (Field Programmable Gate Array)

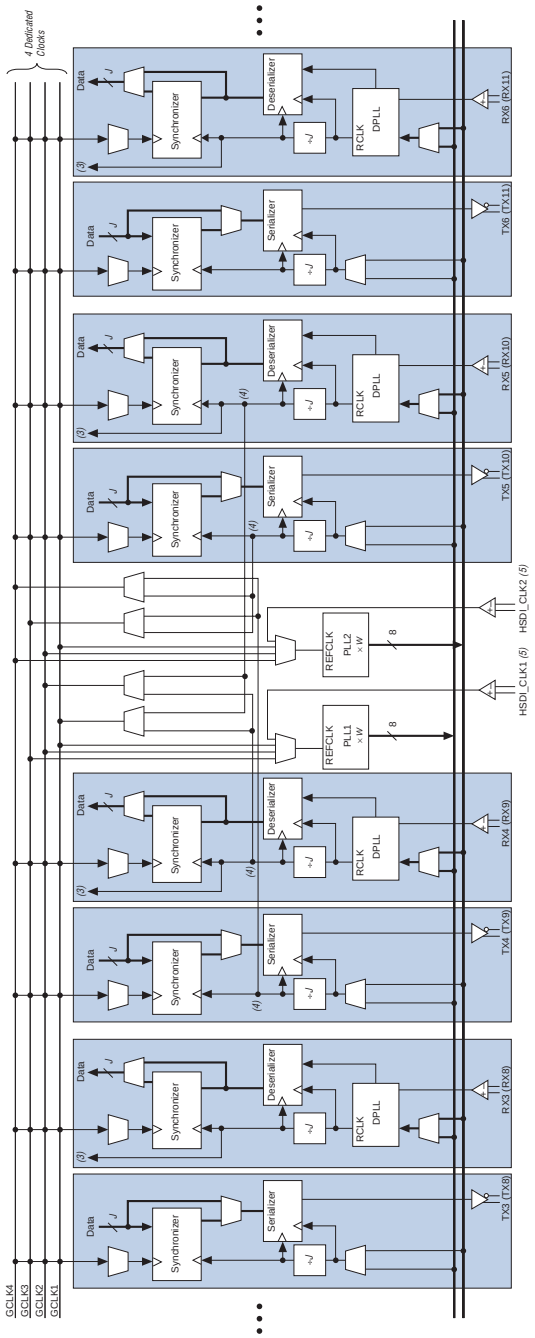
Embedded - FPGAs, or Field Programmable Gate Arrays, are advanced integrated circuits that offer unparalleled flexibility and performance for digital systems. Unlike traditional fixed-function logic devices, FPGAs can be programmed and reprogrammed to execute a wide array of logical operations, enabling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

Applications of Embedded - FPGAs

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

Details	
Product Status	Obsolete
Number of LABs/CLBs	480
Number of Logic Elements/Cells	4800
Total RAM Bits	49152
Number of I/O	303
Number of Gates	120000
Voltage - Supply	1.71V ~ 1.89V
Mounting Type	Surface Mount
Operating Temperature	0°C ~ 85°C (TJ)
Package / Case	484-BBGA, FCBGA
Supplier Device Package	484-FBGA (23x23)
Purchase URL	https://www.e-xfl.com/product-detail/intel/ep1m120f484c7es

Figure 4. Receiver & Transmitter Diagrams for CDR Mode *Notes (1), (2)*



Notes to Figure 4:

(1) EP1M350 devices have 18 individual receiver and transmitter channels. EP1M120 devices have 8 individual receiver and transmitter channels. Receiver and transmitter channel numbers in parenthesis are for EP1M350 devices.
(2) *W* = 1 to 12, 14, 16, 18, or 20
J = 3 to 12, 14, 16, 18, or 20
W does not have to equal *J*.
(3) For every receiver channel in EP1M350 and EP1M120 devices, the ÷*J* recovered clock can drive the priority column interconnect for use as a clock.
(4) The two center channels adjacent to the HSDI PLLs (channels 4 and 5 for EP1M120 devices, channels 9 and 10 for EP1M350 devices) can drive the Mercury device's global clocks.
(5) `HSDI_CLK1` and `HSDI_CLK2` pins must be differential. These clock pins drive HSDI PLLs only. They do not drive to the logic array.

The multiplied reference clock is also used to synchronize and serialize at the transmitter side.

Up to two different serial data rates are supported for input channels or output channels. Received data must be non-return-to-zero (NRZ).

Table 7 defines the support for CDR-mode applications. Table 8 shows the supported data rates for each speed grade.

Table 7. CDR-Mode Applications

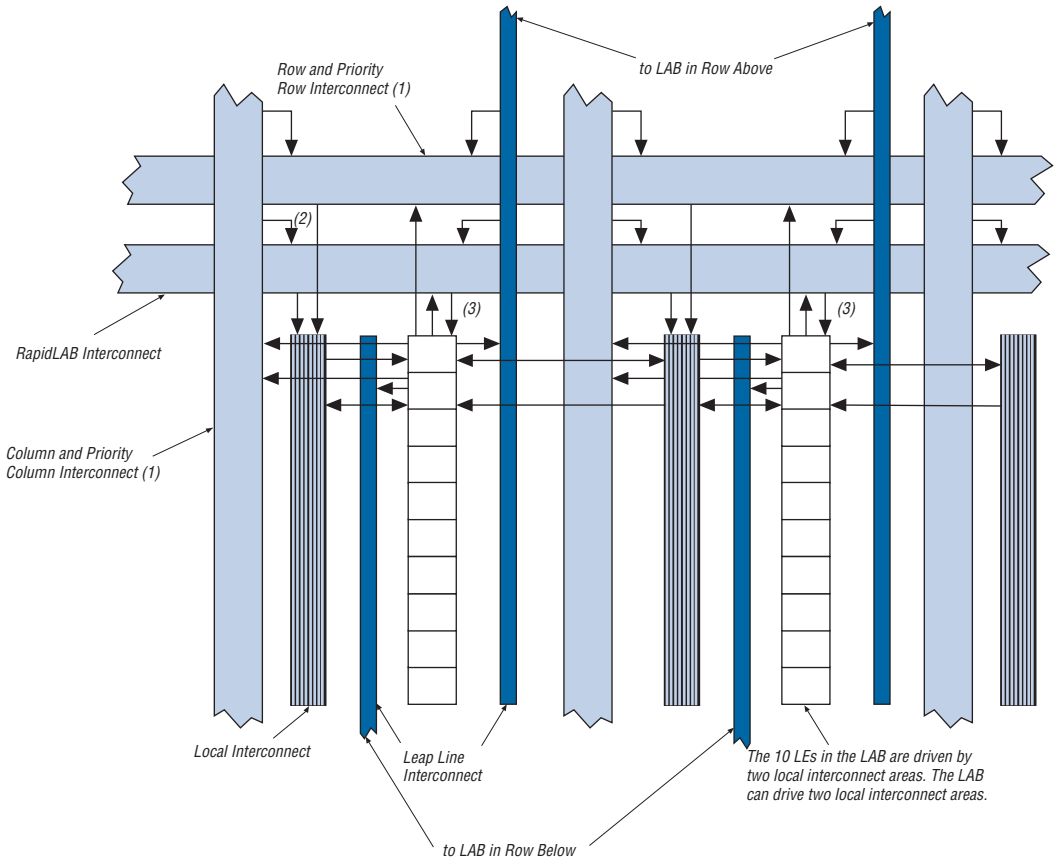
Data Rate	CDR Mode					
	DC-Coupled LVDS	DC-Coupled LVPECL	DC-Coupled 3.3-V PCML	AC-Coupled LVDS *(1)*	AC-Coupled LVPECL *(1)*	AC-Coupled 3.3-V PCML *(1)*
1.0 to 1.25 Gbps	*(2)*	✓	✓	✓	✓	✓
≤ 1.0 Gbps	✓	✓	✓	✓	✓	✓

Notes to Table 7:

(1) The V_{CM} operating range for AC-coupled applications is from 0 to 0.7 V and from 1.8 to 2.4 V.
(2) Use AC-coupled LVDS or another I/O standard. The DC-coupled LVDS I/O standard provides performance up to 1.0 Gbps.

For more information on CDR, see *AN 130: CDR in Mercury Devices*.

Figure 5. Mercury LAB Structure

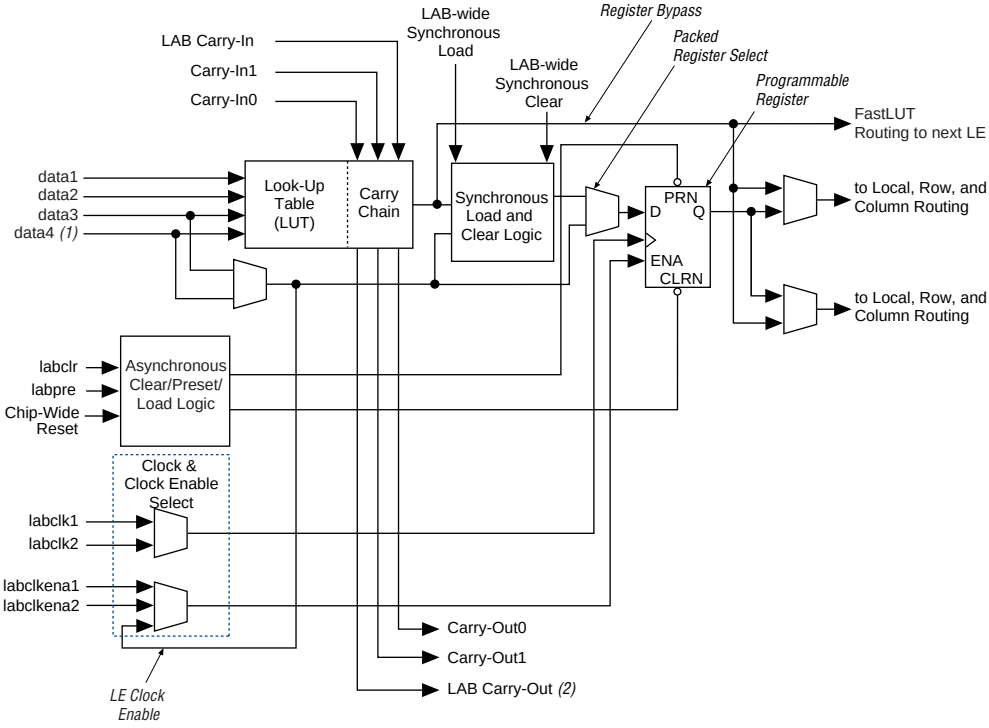


Notes to Figure 5:

(1) Priority column lines drive priority row lines, but not other row lines.
(2) The RapidLAB interconnect can be driven by priority column lines, but not other column lines.
(3) In multiplier mode, the RapidLAB interconnect drives LEs directly.

Mercury devices use an interleaved LAB structure, which allows each LAB to drive two local interconnect areas. Every other LE drives to either the left or right local interconnect area, alternating by LE. The local interconnect can drive LEs within the same LAB or adjacent LABs. This feature minimizes use of the row and column interconnects, providing higher performance and flexibility. Each LAB structure can drive 30 LEs through fast local interconnects.

Figure 7. Mercury LE



Notes to Figure 7:

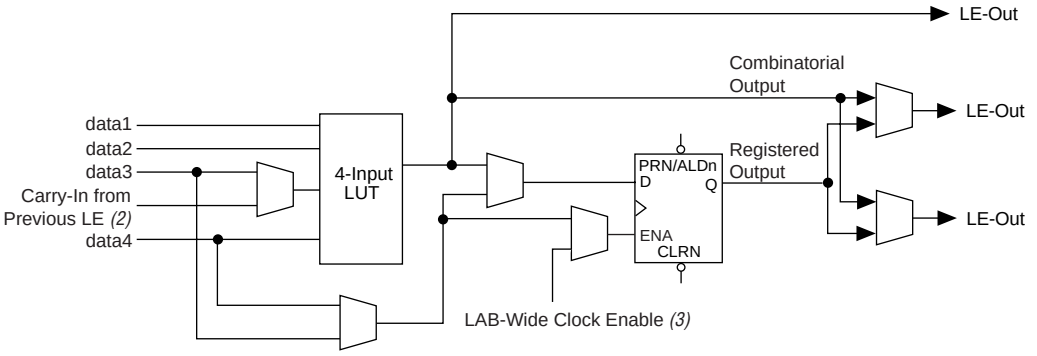
(1) FastLUT interconnect uses the `data4` input.

(2) LAB carry-out can only be generated by LE 4 and / or LE 10.

Each LE's programmable register can be configured for D, T, JK, or SR operation. The register's clock, clock enable, and clear control signals can be driven by global signals, general-purpose I / O pins, or any internal logic. For combinatorial functions, the register is bypassed and the output of the LUT drives directly to the outputs of the LE.

Each LE has four data inputs that can drive the internal LUT. One of these inputs has a shorter delay than the others, improving overall LE performance. This input is chosen automatically by the Quartus II software as appropriate.

Figure 8. Normal-Mode LE *Note (1)*



Notes to Figure 8:

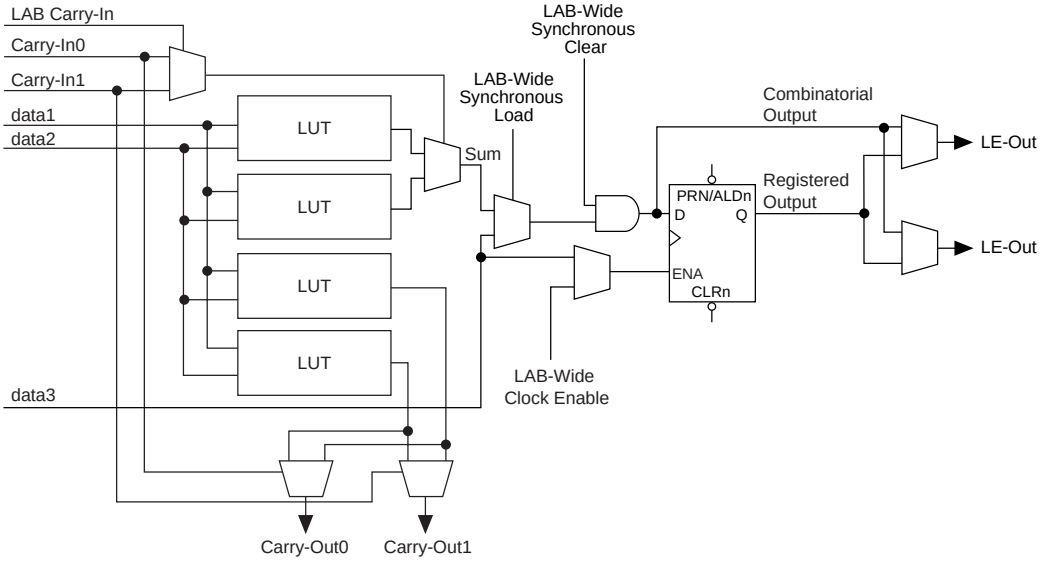
(1) LEs in normal mode support register packing.
(2) When using the carry-in in normal mode, the packed register feature is unavailable.
(3) There are two LAB-wide clock enables per LAB in addition to LE-specific clock enables.

Arithmetic Mode

The arithmetic mode is ideal for implementing adders, accumulators, and comparators. A LE in arithmetic mode contains four 2-input LUTs. The first two 2-input LUTs compute two summations based on a possible carry of `1` or `0`; the other two LUTs generate carry outputs for the two possible chains of the carry-select look-ahead (CSLA) circuitry. As shown in Figure 9, the LAB carry-in signal selects the appropriate carry-in chain (either `carry-in0` or `carry-in1`). The logic level of the chain selected in turn selects which parallel sum is generated as a combinatorial or registered output. For example, when implementing an adder, this output is the signal comprised of the sum `data1` + `data2` + *carry*, where *carry* is `0` or `1`. The other two LUTs use the `data1` and `data2` signals to generate two possible carry-out signals—one for a carry of `1` and the other for a carry of `0`. The `carry-in0` signal acts as the carry select for the `carry-out0` output; `carry-in1` acts as the carry select for the `carry-out1` output. LEs in arithmetic mode can drive out registered and unregistered versions of the LUT output. Figure 9 shows a Mercury LE in arithmetic mode.

The arithmetic mode also offers clock enable, counter enable, synchronous up / down control, synchronous clear, and synchronous load options. The counter enable and synchronous up / down control signals are generated from the data inputs of the LAB local interconnect. The synchronous clear and synchronous load options are LAB-wide signals that affect all registers in the LAB. Consequently, if any of the LEs in a LAB use the counter mode, other LEs in that LAB must be used as part of the same counter or be used for a combinatorial function. The Quartus II software automatically places any registers that are not used by the counter into other LABs.

Figure 9. Arithmetic Mode LE

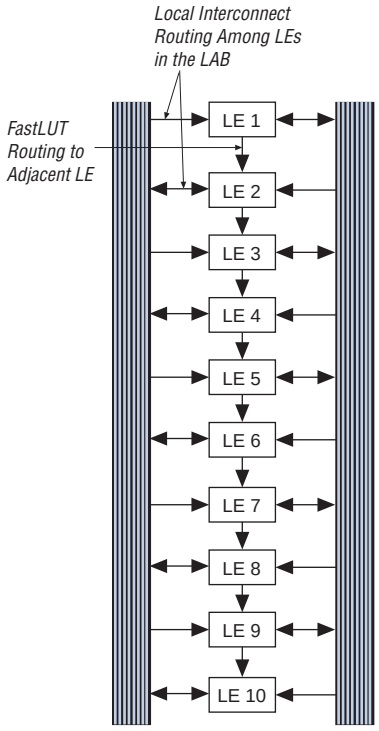


Carry-Select Look-Ahead Chain

The CSLA chain provides a very fast carry-forward function between LEs in arithmetic mode or multiplier mode. The CSLA chain uses the redundant carry calculation to increase the speed of carry functions. The LE can calculate sum and carry values for a possible carry-in of `1` and carry-in of `0` in parallel. The `carry-in0` and `carry-in1` signals from a lower-order bit drive forward into the higher-order bit via the parallel carry chain and feed into both the LUT and the next portion of the CSLA chain. CSLA chains can begin in any LE within a LAB.

For a typical 16 × 16-bit binary tree multiplier, five stages are needed to determine the final product. The Mercury LE multiplier mode allows the partial product formation stage (Stage 1) and the first sum of stages (Stage 2) to be combined in a single stage, shown in Figure 13. This feature, combined with the direct connection between RapidLAB lines and LEs in multiplier mode, allows the fast dedicated implementation of multipliers.

Figure 16. FastLUT Interconnect



ESB rows also have their own interconnect resources to communicate horizontally and vertically with LAB rows. The ESB rows at the top and bottom of the device have their own set of row and priority row interconnect resources. For vertical communication, all LAB column interconnect lines traverse to the ESBs. This includes leap lines, which allow the adjacent LAB rows to communicate with the ESBs.

The row interconnect resources can be driven directly by LEs or ESBs in that row. Further, the column interconnect resources can drive a row line, allowing LEs, IOEs, and ESBs to drive elements in a different row via the column and row resources.

The column interconnect resources can be directly driven by LEs, IOEs, or ESBs within that column. The priority column and leap line resources can be driven directly by LEs. These lines enable high-speed vertical communication in the device for timing-critical paths. The column resources route signals between rows. A column resource can drive row resources directly, allowing fast connections between rows.

Table 9 summarizes how various elements of the Mercury architecture drive each other.

Table 9. Mercury Routing Scheme

Source	Destination										
	LE	Local Interconnect	IOE	ESB Row Interconnect	ESB	Row	Priority Row	RapidLAB Interconnect	Column	Priority Column	Leap Lines
LE	✓ (1)	✓				✓	✓	✓	✓	✓	✓
Local Interconnect	✓		✓								
IOE		✓ (2)				✓ (3)	✓ (3)		✓	✓	
ESB Row Interconnect					✓						
ESB				✓					✓	✓	✓
Row		✓									
Priority Row		✓									
RapidLAB Interconnect	✓ (4)	✓									
Column				✓		✓	✓		✓		
Priority Column				✓			✓	✓	✓	✓	
Leap Lines				✓		✓	✓	✓	✓		

Notes to Table 9:

(1) This direct connection is possible through the FastLUT connection.
(2) IOEs can connect to the adjacent LAB's local interconnects in the associated LAB row.
(3) IOEs can connect to row and priority row interconnects in the associated LAB row.
(4) This connection is used for multiplier mode.

Embedded System Block

The ESB can implement various types of memory blocks, including quad-port, true dual-port, dual- and single-port RAM, ROM, FIFO, and CAM blocks.

The ESB includes input and output registers; the input registers synchronize reads and/or writes, and the output registers can pipeline designs to further increase system performance. The ESB offers a quad port mode, which supports up to four port operations, two reads and two writes simultaneously, with the ability for a different clock on each of the four ports. Figure 17 shows the ESB quad-port block diagram.

ESBs can implement synchronous RAM, which is easier to use than asynchronous RAM. A circuit using asynchronous RAM must generate the RAM write enable (WE) signal while ensuring that its data and address signals meet setup and hold time specifications relative to the WE signal. In contrast, the ESB's synchronous RAM generates its own WE signal and is self-timed with respect to the global clock. Circuits using the ESB's self-timed RAM must only meet the setup and hold time specifications relative to the global clock.

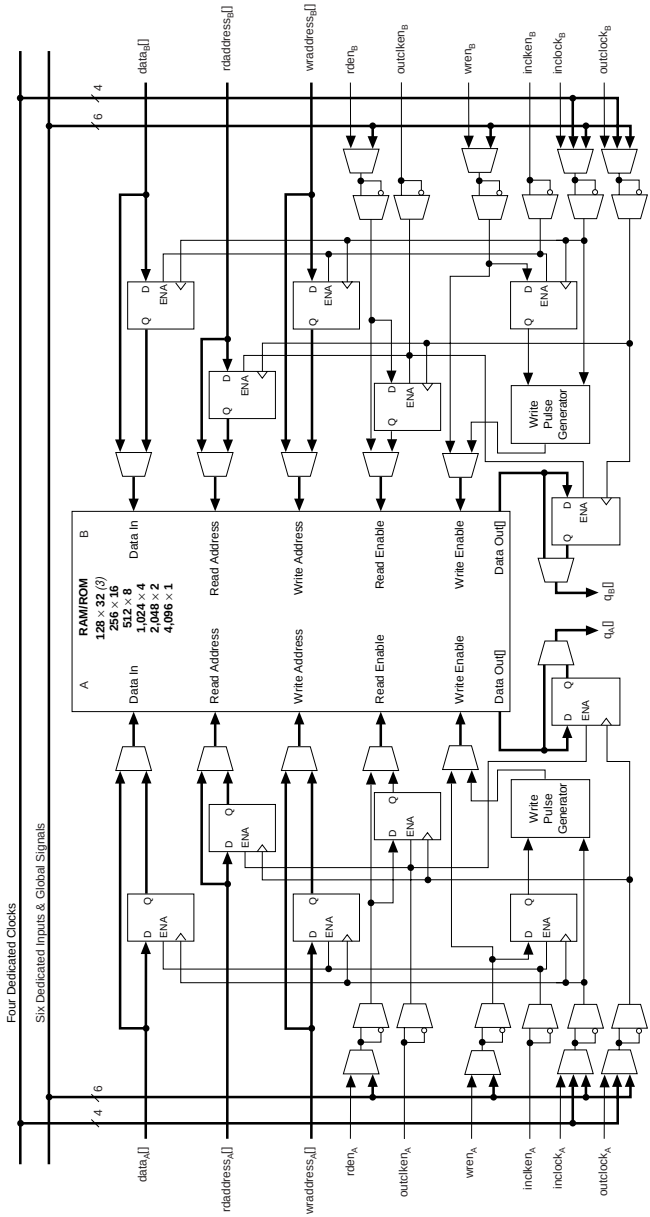
ESBs are grouped together in rows at the top and bottom of the device for fast horizontal communication. The ESB row interconnect can be driven by any ESB in the row. The row interconnect drives the ESB local interconnect, which in turn drives the ESB ports. ESB outputs drive the ESB local interconnect, which can drive row interconnect as well as all types of column interconnect, including leap lines. The leap lines allow fast access between ESBs and the adjacent LAB row.

When implementing memory, each ESB can be configured in any of the following sizes for quad port and true dual-port memory modes: 256 × 16; 512 × 8; 1,024 × 4; 2,048 × 2; or 4,096 × 1. For dual-port and single-port modes, the ESB can be configured for 128 × 32 in addition to the list above. For variable port width RAMs, any port width ratio combination must be 1, 2, 4, 8, or 16. For example, a RAM with data ports of width 1 and 16 or 2 and 32 will work, but not 1 and 32.

The ESB can also be split in half and used for two independent 2,048-bit single-port or dual-port RAM blocks. For example, one half of the ESB can be used as a 128 × 16 memory single-port memory while the other half can be used for a 1,024 × 2 dual-port memory. This effectively doubles the number of RAMs a Mercury device can implement for its given number of ESBs. The Quartus II software automatically merges two logical memory functions in a design into an ESB; the designer does not need to merge the functions manually.

By combining multiple ESBs, the Quartus II software implements larger memory blocks automatically. For example, two 256 × 16 RAM blocks can be combined to form a 256 × 32 RAM block, and two 512 × 8 RAM blocks can be combined to form a 512 × 16 RAM block. Memory performance does not degrade for memory blocks up to 4,096 words deep. Each ESB can implement a 4,096-word-deep memory; the ESBs are used in parallel, eliminating the need for any external control logic and its associated delays. To create a high-speed memory block more than 4,096 words deep, the Quartus II software will automatically combine ESBs with LE control logic.

Figure 19. ESB in Read/Write Clock Mode *Notes (1), (2)*



Notes to Figure 19:

(1) Only half of the ESB, either A or B, is used for dual-port configuration.
(2) All registers can be asynchronously cleared by ESB local interconnect signals, global signals, or the chip-wide reset.
(3) This configuration is supported for dual-port configuration.

Figure 23. Encoded CAM Address Outputs

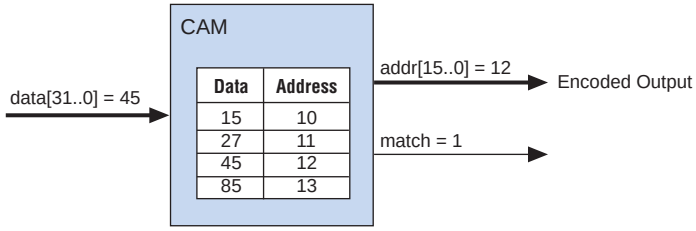
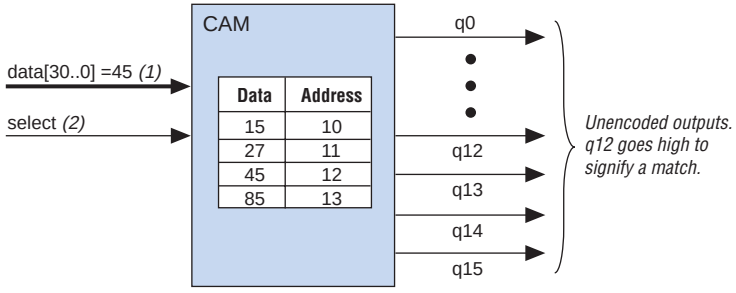


Figure 24. Unencoded CAM Address Outputs



Notes to Figure 24:

(1) For an unencoded output, the ESB only supports 31 input data bits. One input bit is used by the `select` line to choose one of the two banks of 16 outputs.

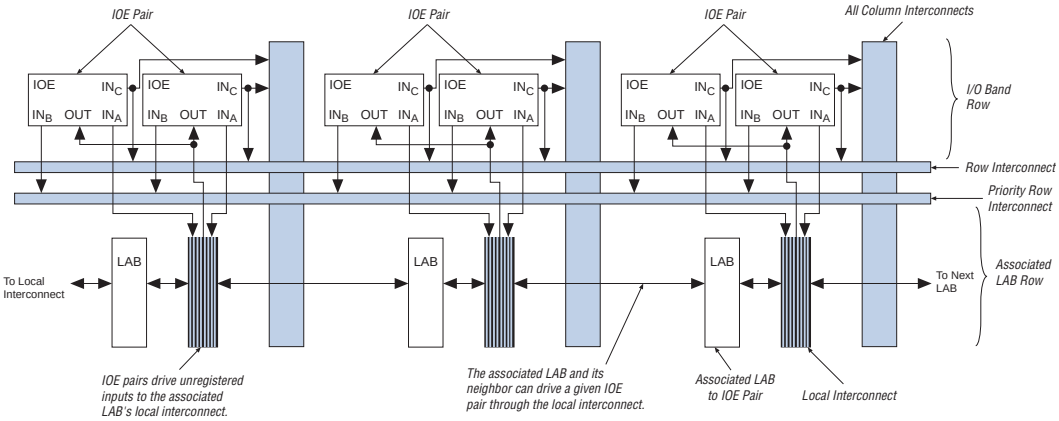
(2) If the `select` input is a 1, then CAM outputs odd words between 1 through 15. If the `select` input is a 0, CAM outputs words even words between 0 through 14.

In single-match mode, it takes two clock cycles to write into CAM, but only one clock cycle to read from CAM. In this mode, both encoded and unencoded outputs are available without external logic. Single-match mode is better suited for designs without duplicate data in the memory.

Each row of I/O pins has an associated LAB row for driving to and from the core of the Mercury device. For a given I/O band row, its associated LAB row is located below it with the exception of the bottom I/O band row. The bottom I/O band is located at the bottom periphery of the device, hence its associated LAB row is located above it. Figure 29 shows an example of an I/O band to associated LAB row interconnect in a Mercury device.

There is a maximum of two IOEs associated with each LAB in the associated LAB row. The local interconnect of the associated LAB drives the IOEs. Since local interconnect is shared with the LAB neighbor, any given LAB can directly drive up to four IOEs. The local interconnect drives the data and OE signals when the IOE is used as an output or bidirectional pin.

Figure 29. IOE Connection to Interconnects and Adjacent LAB *Note (1)*



Note to Figure 29:

(1) IN_A: unregistered input; IN_B: registered / unregistered input; IN_C: registered / unregistered input or OE register output in DDR mode.

The IOEs drive registered or combinatorial versions of input data into the device. The unregistered input data can be driven to the local interconnect (for fast input setup), row and priority row interconnect, and column and priority column interconnects. The registered data can also be driven to the same row and column resources. The OE register output can be fed back through column and row interconnects to implement DDR I/O pins.

The PLLs in Mercury devices are enabled through the Quartus II software. External devices are not required to use these features.

Advanced ClockBoost Multiplication & Division

Each Mercury PLL includes circuitry that provides clock synthesis for up to four outputs (three internal outputs and one external output) using $m / (n \times$ output divider) scaling. When a PLL is locked, the locked output clock aligns to the rising edge of the input clock. The closed loop equation for Figure 31 gives an output frequency $f_{clock0} = (m / (n \times k))f_{IN}$, $f_{clock1} = (m / (n \times p))f_{IN}$, $f_{clock2} = (m / (n \times q))f_{IN}$, and $f_{clock\_ext} = (m / (n \times v))f_{IN}$ or f_{clock1}. These equations allow the multiplication or division of clocks by a programmable number. The Quartus II software automatically chooses the appropriate scaling factors according to the frequency, multiplication, and division values entered.

A single PLL in a Mercury device allows for multiple user-defined multiplication and division ratios that are not possible even with multiple delay-locked loops (DLLs). For example, if a frequency scaling factor of 3.75 is needed for a given input clock, a multiplication factor of 15 and a division factor of 4 can be entered. This advanced multiplication scaling can be performed with a single PLL, making it unnecessary to cascade PLL outputs.

External Clock Outputs

Mercury devices have four low-jitter external clocks available for external clock sources. Other devices on the board can use these outputs as clock sources.

There are three modes for external clock outputs. Multiplication is allowed in all external clock output modes.

- Zero Delay Buffer: The external clock output pin is phase aligned with the clock input pin for zero delay. Programmable phase shift and time delay shift are not allowed in this configuration. Multiplication is allowed with the zero delay buffer mode. The MegaWizard interface for `altclklock` should be used to verify possible clock settings.
- External Feedback: The external feedback input pin is phase aligned with clock input pin. By aligning these clocks, you can actively remove clock delay and skew between devices. Multiplication is allowed with the external feedback mode. This mode has the same restrictions as zero delay buffer mode.

For more information, see the following documents:

- *Application Note 39 (IEEE Std. 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices)*
- Jam Programming & Test Language Specification

Generic Testing

Each Mercury device is functionally tested. Complete testing of each configurable static random access memory (SRAM) bit and all logic functionality ensures 100% yield. AC test measurements for Mercury devices are made under conditions equivalent to those shown in Figure 33. Multiple test patterns can be used to configure devices during all stages of the production flow.

Table 23. LVTTL Specifications *Note (10)*

Symbol	Parameter	Conditions	Minimum	Maximum	Units
V_{CCIO}	Output supply voltage		3.0	3.6	V
V_{IH}	High-level input voltage		1.7	4.1	V
V_{IL}	Low-level input voltage		–0.5	0.7	V
I_I	Input pin leakage current	V_{IN} = 0 V or V_{CCIO}	–10	10	µA
V_{OH}	High-level output voltage	I_{OH} = –4 mA	2.4		V
V_{OL}	Low-level output voltage	I_{OL} = 4 mA		0.45	V

Table 24. LVCMOS Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Units
V_{CCIO}	Power supply voltage range		3.0	3.6	V
V_{IH}	High-level input voltage		1.7	4.1	V
V_{IL}	Low-level input voltage		–0.5	0.7	V
I_I	Input pin leakage current	V_{IN} = 0 V or V_{CCIO}	–10	10	µA
V_{OH}	High-level output voltage	V_{CCIO} = 3.0, I_{OH} = –0.1 mA	V_{CCIO} – 0.2		V
V_{OL}	Low-level output voltage	V_{CCIO} = 3.0, I_{OL} = 0.1 mA		0.2	V

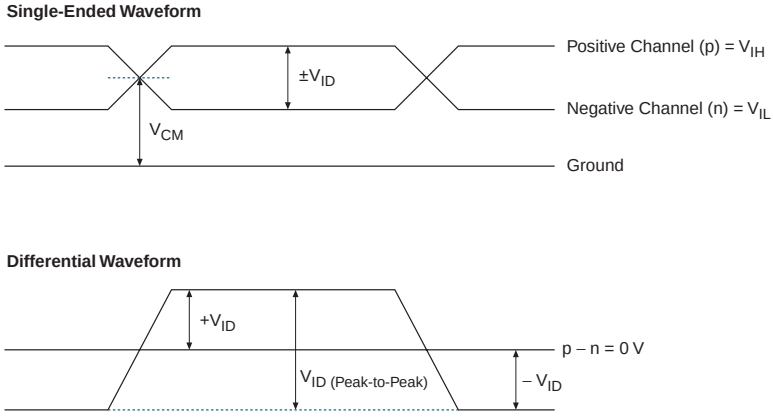
Table 25. 2.5-V I/O Specifications *Note (10)*

Symbol	Parameter	Conditions	Minimum	Maximum	Units
V_{CCIO}	Output supply voltage		2.375	2.625	V
V_{IH}	High-level input voltage		1.7	4.1	V
V_{IL}	Low-level input voltage		–0.5	0.7	V
I_I	Input pin leakage current	V_{IN} = 0 V or V_{CCIO}	10	10	µA
V_{OH}	High-level output voltage	I_{OH} = –0.1 mA	2.1		V
		I_{OH} = –1 mA	2.0		V
		I_{OH} = –2 mA	1.7		V
V_{OL}	Low-level output voltage	I_{OL} = 0.1 mA		0.2	V
		I_{OH} = 1 mA		0.4	V
		I_{OH} = 2 mA		0.7	V

Table 26. 1.8-V I/O Specifications *Note (10)*					
Symbol	**Parameter**	**Conditions**	**Minimum**	**Maximum**	**Units**
V_{CCIO}	Output supply voltage		1.71	1.89	V
V_{IH}	High-level input voltage		$0.65 \times V_{CCIO}$	4.1	V
V_{IL}	Low-level input voltage		–0.5	$0.35 \times V_{CCIO}$	V
I_I	Input pin leakage current	V_{IN} = 0 V or V_{CCIO}	–10	10	µA
V_{OH}	High-level output voltage	I_{OH} = –2 mA	V_{CCIO} – 0.45		V
V_{OL}	Low-level output voltage	I_{OL} = 2 mA		0.45	V

Figures 34 and 35 show receiver input and transmitter output waveforms, respectively, for all differential I/O standards (LVPECL, 3.3-V PCML, LVDS, and HyperTransport technology).

Figure 34. Receiver Input Waveforms for Differential I/O Standards



Tables 44 and 45 describe the Mercury device's external timing parameters.

Table 44. Mercury External Timing Parameters *Notes (1), (2)*

Symbol	Parameter	Conditions
t_{INSU}	Setup time with global clock at IOE register	
t_{INH}	Hold time with global clock at IOE register	
t_{OUTCO}	Clock-to-output delay with global clock at IOE register	C1 = 35 pF
$t_{INSUPLL}$	Setup time with PLL clock at IOE input register	
t_{INHPLL}	Hold time with PLL clock at IOE input register	
$t_{OUTCOPLL}$	Clock-to-output delay with PLL clock at IOE output register	C1 = 35 pF

Table 45. Mercury External Bidirectional Timing Parameters *Notes (1), (2)*

Symbol	Parameter	Conditions
$t_{INSUBIDIR}$	Setup time for bidirectional pins with gobal clock at IOE input register	
$t_{INHBIDIR}$	Hold time for bidirectional pins with global clock at IOE input register	
$t_{OUTCOBIDIR}$	Clock-to-output delay for bidirectional pins with global clock at IOE output register	C1 = 35 pF
$t_{XZBIDIR}$	Synchronous IOE output enable register to output buffer disable delay	C1 = 35 pF
$t_{ZXBIDIR}$	Synchronous IOE output enable register output buffer enable delay	C1 = 35 pF
$t_{INSUBIDIRPLL}$	Setup time for bidirectional pins with PLL clock at IOE input register	
$t_{INHBIDIRPLL}$	Hold time for bidirectional pins with PLL clock at IOE input register	
$t_{OUTCOBIDIRPLL}$	Clock-to-output delay for bidirectional pins with PLL clock at IOE output register	C1 = 35 pF
$t_{XZBIDIRPLL}$	Synchronous IOE output enable register to output buffer disable delay with PLL	C1 = 35 pF
$t_{ZXBIDIRPLL}$	Synchronous IOE output enable register output buffer enable delay with PLL	C1 = 35 pF

Notes to Tables 44 and 45:

(1) These timing parameters are sample-tested only.

(2) All timing parameters are either to and / or from pins, including global clock pins.

Before and during device configuration, all I/O pins are pulled to V_{CCIO} by a built-in weak pull-up resistor.

SRAM configuration elements allow Mercury devices to be reconfigured in-circuit by loading new configuration data into the device. Real-time reconfiguration is performed by forcing the device into command mode with a device pin, loading different configuration data, reinitializing the device, and resuming user-mode operation. In-field upgrades can be performed by distributing new configuration files.

Configuration Schemes

The configuration data for a Mercury device can be loaded with one of five configuration schemes (see Table 52), chosen on the basis of the target application. A configuration device, intelligent controller, or the JTAG port can be used to control the configuration of a Mercury device. When a configuration device is used, the system can configure automatically at system power-up.

By connecting the configuration enable (nCE) and configuration enable output (nCEO) pins on each device, multiple Mercury devices can be configured in any of five configuration schemes.

Table 52. Data Sources for Configuration

Configuration Scheme	Data Source
Configuration device	Configuration device
Passive serial (PS)	MasterBlaster™ or ByteBlasterMV™ download cable or serial data source
Passive parallel asynchronous (PPA)	Parallel data source
Passive parallel synchronous (PPS)	Parallel data source
JTAG	MasterBlaster or ByteBlasterMV download cable or a microprocessor with a Jam STAPL or JBC file

For more information on configuration, see *Application Note 116 (Configuring APEX 20K, FLEX 10K & FLEX 6000 Devices)*.

Device Pin-Outs

See the Altera web site (**http://www.altera.com**) or the ***Altera Digital Library*** for pin-out information.

Revision History

The information contained in the *Mercury Programmable Logic Device Family Data Sheet* version 2.2 supersedes information published in previous versions.

Version 2.2

The following changes were made to the *Mercury Programmable Logic Device Family Data Sheet* version 2.2:

- Updated the condition values (symbols I_I and I_{OZ}) in Table 22.

Version 2.1

The following changes were made to the *Mercury Programmable Logic Device Family Data Sheet* version 2.1:

- Updated Table 8.
- Updated EP1M350 regular I/O banks in Table 13.
- Updated *Note (6)* in Table 14.

Version 2.0

The following changes were made to the *Mercury Programmable Logic Device Family Data Sheet* version 2.0:

- Changed all references to PCML to 3.3-V PCML.
- Updated Table 4.
- Updated "High-Speed Differential Interface" on page 8.
- Added Tables 6 through 8.
- Added Figures 34 and 35.
- Updated I/O specifications in Tables 28 and 29.
- Updated Mercury device capacitance in Table 43.
- Updated EP1M120 device timing in Tables 46 through 49.
- Added EP1M350 device timing in Tables 50 and 51.