Intel - EP3SL340F1517C3N Datasheet

Welcome to E-XFL.COM

Understanding Embedded - FPGAs (Field Programmable Gate Array)

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

Applications of Embedded - FPGAs

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

Details	
Product Status	Obsolete
Number of LABs/CLBs	13500
Number of Logic Elements/Cells	337500
Total RAM Bits	18822144
Number of I/O	976
Number of Gates	-
Voltage - Supply	0.86V ~ 1.15V
Mounting Type	Surface Mount
Operating Temperature	0°C ~ 85°C (TJ)
Package / Case	1517-BBGA, FCBGA
Supplier Device Package	1517-FBGA (40x40)
Purchase URL	https://www.e-xfl.com/product-detail/intel/ep3sl340f1517c3n

SIII5V1-2.2

List of Figures

Table 1–4 lists the Stratix III Hybrid FineLine BGA (HBGA) package sizes.

Table 1–4. Hybrid FineLine BGA Package Sizes

Dimension	780 Pin	1152 Pin
Pitch (mm)	1.00	1.00
Area (mm^2)	1,089	1,600
Length/Width (mm/ mm)	33/33	40/40

Stratix III devices are available in up to three speed grades: –2, –3, and –4, with –2 being the fastest. Stratix III devices are offered in both commercial and industrial temperature range ratings with leaded and lead-free packages. Selectable Core Voltage is available in specially marked low-voltage devices (*L* ordering code suffix).

Table 1–5 lists the Stratix III device speed grades.

Table 1–5. Speed Grades for Stratix III Devices (Part 1 of 2)

Device	Temperature Grade	484 -Pin FineLine BGA	780-Pin FineLine BGA	780-Pin Hybrid FineLine BGA	1152-Pin FineLine BGA	1152-Pin Hybrid FineLine BGA	1517-Pin FineLine BGA	1760-Pin FineLine BGA
EP3SL50	Commercial	–2, –3, –4, –4L	–2, –3,–4, –4L	—	—	—	—	—
	Industrial	–3, –4, –4L	–3, –4, –4L	—	—	—	—	—
EP3SL70	Commercial	–2, –3, –4, –4L	–2, –3, –4, –4L	—	—	—	—	—
	Industrial	–3, –4, –4L	–3, –4, –4L	—	—	—	—	—
EP3SL110	Commercial	—	–2, –3, –4, –4L	—	–2, –3, –4, –4L	—	—	—
	Industrial	—	–3, –4, –4L	—	–3, –4, –4L	—	—	—
EP3SL150	Commercial	—	–2,–3, –4, –4L	—	–2, –3, –4, –4L	—	—	—
	Industrial	—	–3, –4, –4L	—	–3, –4, –4L	—	—	—
EP3SL200	Commercial	—	—	–2,–3, –4, –4L	–2,–3, –4, –4L	—	–2,–3, –4, –4L	—
	Industrial *(1)*	—	—	–3, –4, –4L	–3, –4, –4L	—	–3, –4, –4L	—
EP3SL340	Commercial	—	—	—	—	–2, –3, –4	–2, –3, –4	–2, –3, –4
	Industrial *(1)*	—	—	—	—	–3, –4, –4L	–3, –4, –4L	–3, –4, –4L
EP3SE50	Commercial	–2, –3, –4, –4L	–2, –3, –4, –4L	—	—	—	—	—
	Industrial	–3, –4, –4L	–3, –4, –4L	—	—	—	—	—
EP3SE80	Commercial	—	–2, –3, –4, –4L	—	–2, –3, –4, –4L	—	—	—
	Industrial	—	–3, –4, –4L	—	–3, –4, –4L	—	—	—
EP3SE110	Commercial	—	–2,–3, –4, –4L	—	–2, –3, –4, –4L	—	—	—
	Industrial	—	–3, –4, –4L	—	–3, –4, –4L	—	—	—

C12 column interconnects span a length of 12 LABs and provide the fastest resource for column connections between distant LABs, TriMatrix memory blocks, DSP blocks, and IOEs. C12 interconnects drive LAB local interconnects via C4 and R4 interconnects and do not drive LAB local interconnects directly.

All embedded blocks communicate with the logic array through interconnects similar to LAB-to-LAB interfaces. Each block (for example, TriMatrix memory blocks and DSP blocks) connects to row and column interconnects and has local interconnect regions driven by row and column interconnects. These blocks also have direct link interconnects for fast connections to and from a neighboring LAB.

Table 3–1 shows the Stratix III device's routing scheme.

Table 3–1. Stratix III Device Routing Scheme

Source	Destination															
	Shared Arithmetic Chain	Carry Chain	Register Chain	Local Interconnect	Direct Link Interconnect	R4 Interconnect	R20 Interconnect	C4 Interconnect	C12 Interconnect	ALM	MLAB RAM Block	M9K RAM Block	M144K Block	DSP Blocks	Column IOE	Row IOE
Shared arithmetic chain	—	—	—	—	—	—	—	—	—	✓	—	—	—	—	—	—
Carry chain	—	—	—	—	—	—	—	—	—	✓	—	—	—	—	—	—
Register chain	—	—	—	—	—	—	—	—	—	✓	—	—	—	—	—	—
Local interconnect	—	—	—	—	—	—	—	—	—	✓	✓	✓	✓	✓	✓	✓
Direct link interconnect	—	—	—	✓	—	—	—	—	—	—	—	—	—	—	—	—
R4 interconnect	—	—	—	✓	—	✓	✓	✓	✓	—	—	—	—	—	—	—
R20 interconnect	—	—	—	✓	—	✓	✓	✓	✓	—	—	—	—	—	—	—
C4 interconnect	—	—	—	✓	—	✓	—	✓	—	—	—	—	—	—	—	—
C12 interconnect	—	—	—	✓	—	✓	✓	✓	✓	—	—	—	—	—	—	—
ALM	✓	✓	✓	✓	✓	✓	—	✓	—	—	—	—	—	—	—	—
MLAB RAM block	—	—	—	✓	✓	✓	—	✓	—	—	—	—	—	—	—	—
M9K RAM block	—	—	—	—	✓	✓	—	✓	—	—	—	—	—	—	—	—
M144K block	—	—	—	—	✓	✓	—	✓	—	—	—	—	—	—	—	—
DSP blocks	—	—	—	—	✓	✓	—	✓	—	—	—	—	—	—	—	—
Column IOE	—	—	—	—	—	—	—	✓	✓	—	—	—	—	—	—	—
Row IOE	—	—	—	—	✓	✓	✓	✓	—	—	—	—	—	—	—	—

Notes to Table 3–1:

(1) Except column IOE local interconnects.
(2) Row IOE local interconnects.
(3) Column IOE local interconnects.

MLABs support byte-enable via emulation. There will be increased logic utilization when the byte-enables are emulated.

The default value for the byte-enable signals is high (enabled), in which case writing is controlled only by the write enable signals. The byte-enable registers have no clear port. When using parity bits on the M9K and M144K blocks, the byte-enable controls all nine bits (eight bits of data plus one parity bit). When using parity bits on the MLAB, the byte-enable controls all 10 bits in the widest mode.

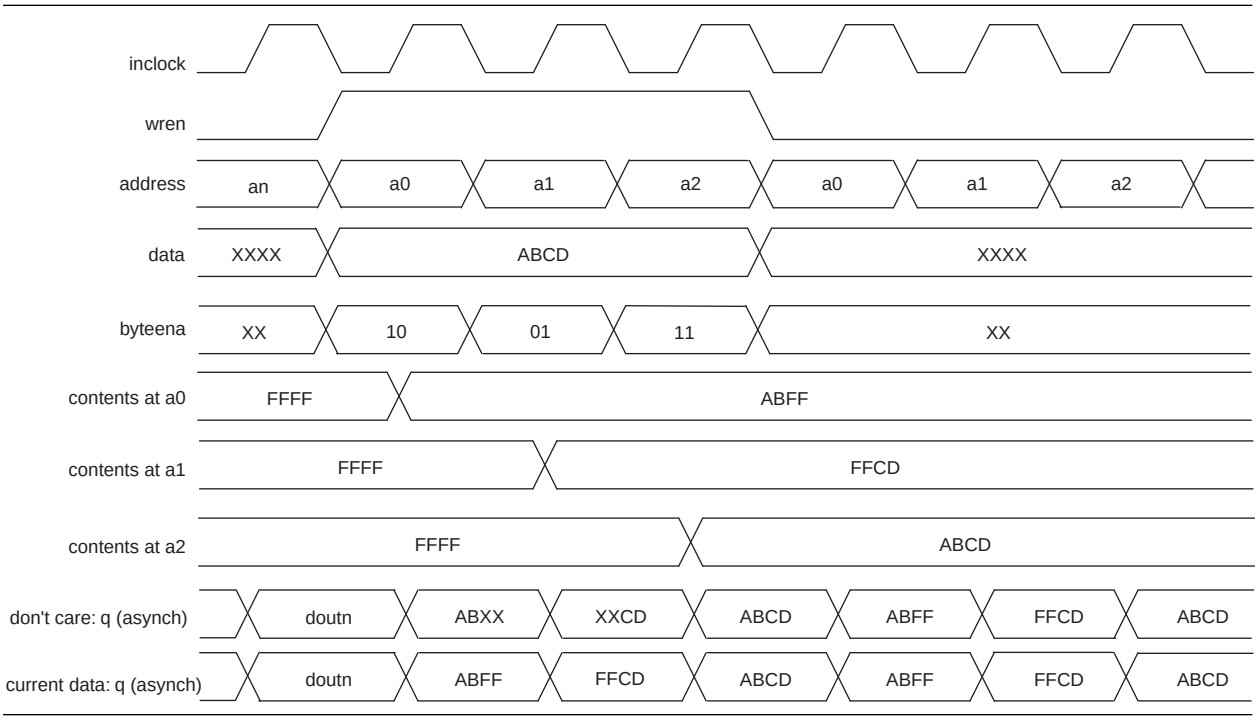
Byte-enables operate in a one-hot fashion, with the LSB of the `byteena` signal corresponding to the least significant byte of the data bus. For example, if you are using a RAM block in ×18 mode, with `byteena = 01`, `data[8..0]` is enabled and `data[17..9]` is disabled. Similarly, if `byteena = 11`, both `data[8..0]` and `data[17..9]` are enabled. Byte-enables are active high.

You cannot use the byte-enable feature when using the ECC feature on M144K blocks.

Figure 4–1 shows how the write enable (`wren`) and byte-enable (`byteena`) signals control the operations of the M9K and M144K.

When a byte-enable bit is de-asserted during a write cycle, the corresponding data byte output can appear as either a "don't care" value or the current data at that location. The output value for the masked byte is controllable via the Quartus II software. When a byte-enable bit is asserted during a write cycle, the corresponding data byte output also depends on the setting chosen in the Quartus II software.

Figure 4–1. Stratix III Byte-Enable Functional Waveform for M9K and M144K



Each Half Block has its own `signa` and `signb` signal. Therefore, all of the `data A` inputs feeding the same DSP Half Block must have the same sign representation. Similarly, all of the `data B` inputs feeding the same DSP Half Block must have the same sign representation. The multiplier offers full precision regardless of the sign representation in all operational modes except for full precision 18 x 18 loopback and Two-Multiplier Adder modes. Refer to "Two-Multiplier Adder Sum Mode" on page 5–21 for details.

When the `signa` and `signb` signals are unused, the Quartus II software sets the multiplier to perform unsigned multiplication by default.

The outputs of the multipliers are the only outputs that can feed into the first-stage adder, as shown in Figure 5–6. There are four first-stage adders in a DSP block (two adders per half DSP block). The first-stage adder block has the ability to perform addition and subtraction. The control signal for addition or subtraction is static and has to be configured upon compile time. The first-stage adders are used by the sum modes to compute the sum of two multipliers, 18 × 18-complex multipliers, and to perform the first stage of a 36 × 36 multiply and shift operation.

Depending on your specifications, the output of the first-stage adder has the option to feed into the pipeline registers, second-stage adder, round and saturation unit, or the output registers.

Pipeline Register Stage

The output from the first-stage adder can either feed or bypass the pipeline registers, as shown in Figure 5–6. Pipeline registers increase the DSP block's maximum performance (at the expense of extra cycles of latency), especially when using the subsequent DSP block stages. Pipeline registers split up the long signal path between the input-registers/multiplier/first-stage adder and the second-stage adder/round-and-saturation/output-registers, creating two shorter paths.

Second-Stage Adder

There are four individual 44-bit second-stage adders per DSP block (2 adders per half DSP block). You can configure the second-stage adders as follows:

- The final stage of a 36-bit multiplier
- A sum of four (18 × 18)
- An accumulator (44-bits maximum)
- A chained output summation (44-bits maximum)

The chained-output adder can be used at the same time as a second-level adder in chained output summation mode.

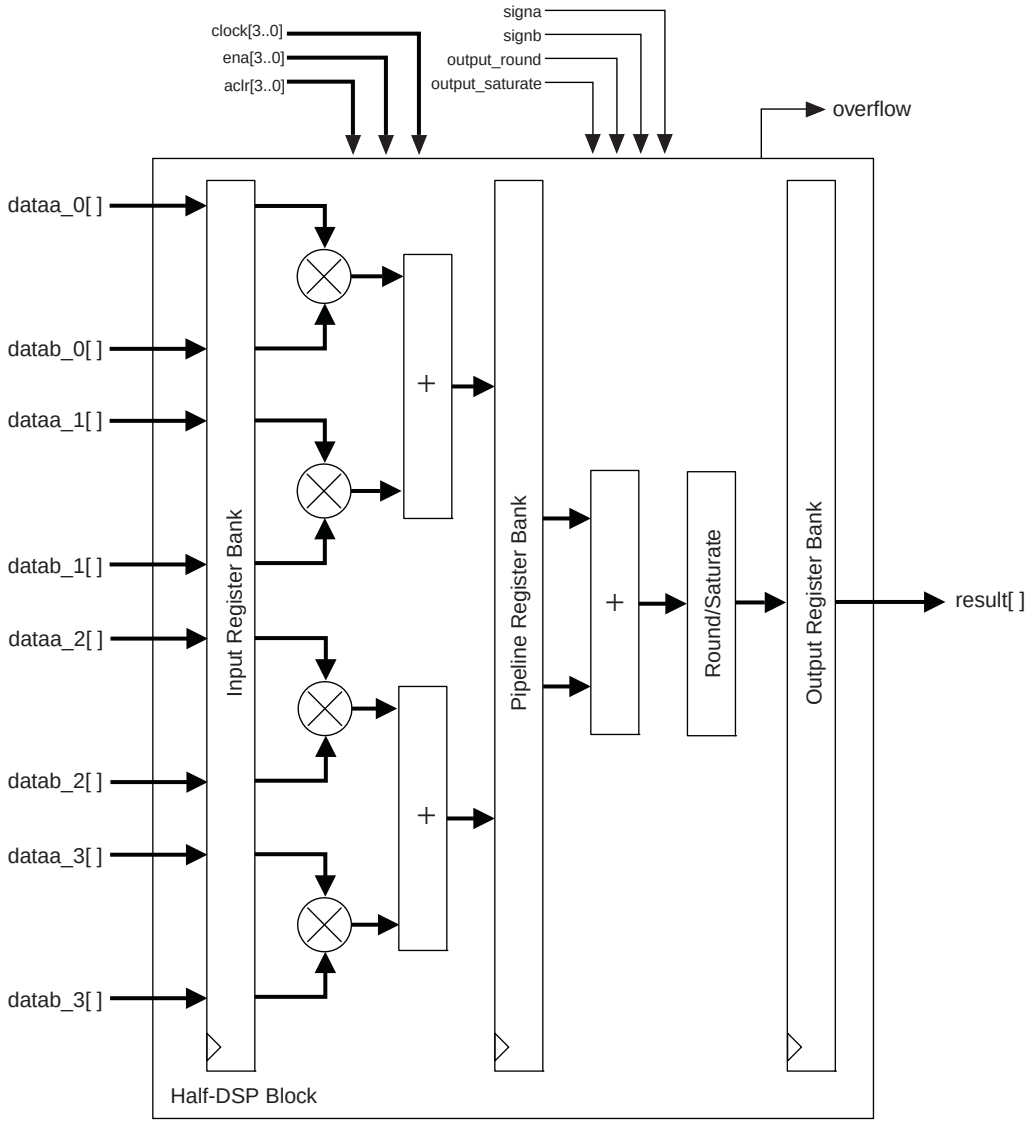
The output of the second-stage adder has the option to go into the round and saturation logic unit or the output register.

You cannot use the second-stage adder independently from the multiplier and first-stage adder.

Four-Multiplier Adder

In the four-multiplier adder configuration shown in Figure 5–17, the DSP block can implement two four-multiplier adders (one four-multiplier adder per half DSP block). These modes are useful for implementing one-dimensional and two-dimensional filtering applications. The four-multiplier adder is performed in two addition stages. The outputs of two of the four multipliers are initially summed in the two first-stage adder blocks. The results of these two adder blocks are then summed in the second-stage adder block to produce the final four-multiplier adder result, as shown by Equation 5–2 and Equation 5–3.

Figure 5–17. Four-Multiplier Adder Mode for Half-DSP Block



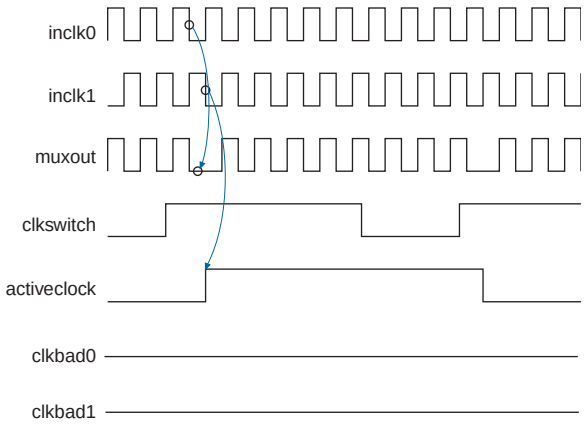
The four-multiplier adder mode supports the round and saturation logic unit. You can use the pipeline registers and output registers within the DSP block to pipeline the multiplier-adder result, increasing the performance of the DSP block.

Manual Override

In the automatic switchover with manual override mode, you can use the clkswitch input for user- or system-controlled switch conditions. You can use this mode for same-frequency switchover or to switch between inputs of different frequencies. For example, if inclk0 is 66 MHz and inclk1 is 200 MHz, you must control the switchover using clkswitch because the automatic clock-sense circuitry cannot monitor clock input (inclk0, inclk1) frequencies with a frequency difference of more than 100% (2×). This feature is useful when the clock sources originate from multiple cards on the backplane, requiring a system-controlled switchover between the frequencies of operation. You should choose the backup clock frequency and set the m, n, c, and k counters accordingly so the VCO operates within the recommended operating frequency range of 600 to 1,300 MHz. The ALTPLL MegaWizard Plug-in Manager notifies users if a given combination of inclk0 and inclk1 frequencies cannot meet this requirement. In the Quartus II software, the VCO value reported is divided by the post scale counter (K).

Figure 6–34 shows an example of a waveform illustrating the switchover feature when controlled by clkswitch. In this case, both clock sources are functional and inclk0 is selected as the reference clock. clkswitch goes high, which starts the switchover sequence. On the falling edge of inclk0, the counter's reference clock, muxout, is gated off to prevent any clock glitching. On the falling edge of inclk1, the reference clock multiplexer switches from inclk0 to inclk1 as the PLL reference, and the activeclock signal changes to indicate which clock is currently feeding the PLL.

Figure 6–34. Clock Switchover Using the clkswitch (Manual) Control *(Note 1)*



Note to Figure 6–34:

(1) Both inclk0 and inclk1 must be running when the clkswitch signal goes high to initiate a manual clock switchover event.

In this mode, the activeclock signal mirrors the clkswitch signal. As both clocks are still functional during the manual switch, neither clkbad signal goes high. Since the switchover circuit is positive-edge sensitive, the falling edge of the clkswitch signal does not cause the circuit to switch back from inclk1 to inclk0. When the clkswitch signal goes high again, the process repeats. clkswitch and automatic switch only work if the clock being switched to is available. If the clock is not available, the state machine waits until the clock is available.

The rselodd bit indicates an odd divide factor for the VCO output frequency along with a 50% duty cycle. For example, if the post-scale divide factor is 3, the high- and low-time count values could be set to 2 and 1, respectively, to achieve this division. This implies a 67%-33% duty cycle. If you need a 50%-50% duty cycle, you can set the rselodd control bit to 1 to achieve this duty cycle despite an odd division factor. The PLL implements this duty cycle by transitioning the output clock from high to low on a falling edge of the VCO output clock. When you set rselodd = 1, you subtract 0.5 cycles from the high time and you add 0.5 cycles to the low time. For example:

- High-time count = 2 cycles
- Low-time count = 1 cycle
- rselodd = 1 effectively equals:
  - High-time count = 1.5 cycles
  - Low-time count = 1.5 cycles
  - Duty cycle = (1.5/3) % high-time count and (1.5/3) % low-time count

Scan Chain Description

The length of the scan chain varies for different Stratix III PLLs. The Top/Bottom PLLs have 10 post-scale counters and a 234-bit scan chain, while the Left/Right PLLs have 7 post-scale counters and a 180-bit scan chain. Table 6–16 lists the number of bits for each component of a Stratix III PLL.

Table 6–16. Top/Bottom PLL Reprogramming Bits (Part 1 of 2)

Block Name	Number of Bits		Total
	Counter	Other *(1)*	
C9 *(2)*	16	2	18
C8	16	2	18
C7	16	2	18
C6 *(3)*	16	2	18
C5	16	2	18
C4	16	2	18
C3	16	2	18
C2	16	2	18
C1	16	2	18
C0	16	2	18
N	16	2	18
M	16	2	18
Charge Pump Current	0	3	3
VCO Post-Scale divider (K)	1	0	1
Loop Filter Capacitor *(4)*	0	2	2
Loop Filter Resistor	0	5	5
Unused CP/LF	0	7	7

Table 6–23. Chapter Revision History (Part 2 of 2)

Date	Version	Changes Made
October 2007	1.2	■ Updated Table 6–13 to remove a reference to gated locks. Updated Table 6–16 and added new rows to it. ■ Modified Figure 6–3 and Figure 6–40. ■ Edited notes for Figure 6–9, Figure 6–10, and Figure 6–17. ■ Replaced Figure 6–41. ■ Added section “Referenced Documents”. ■ Added live links for references.
May 2007	1.1	Changed frequency difference between inclk0 and inclk1 to more than 20% instead of 100% on page 42. Updated Table 6–16, note to Figure 6–17, and Figure 6–19.
November 2006	1.0	Initial Release.

Similarly, in ×16/×18 mode, the I/O bank combines four adjacent ×4 DQS/DQ groups to create a group with a maximum of 19 DQ pins (including parity or DM and QVLD pins) and a pair of DQS/CQ and DQSn/CQn pins. In ×32/×36 mode, the I/O bank combines eight adjacent ×4 DQS DQ groups together to create a group with a maximum of 37 DQ pins (including parity or DM and QVLD pins) and a pair of DQS/CQ and DQSn/CQn pins.

Stratix III modular I/O banks allow easy formation of the DQS/DQ groups. If all the pins in the I/O banks are user I/O pins and are not used for programming, RUP/RDN used for OCT calibration, or PLL clock output pins, you can divide the number of I/O pins in the bank by six to get the maximum possible number of ×4 groups. You can then divide that number by two, four, or eight to get the maximum possible number of ×8/×9, ×16/×18, or ×32/×36, respectively (refer to Table 8–3). However, some of the pins in the I/O bank may be used for other functions.

Table 8–3. DQ/DQS Group in Stratix III Modular I/O Banks

Modular I/O Bank Size	Maximum Possible Number of ×4 Groups *(1)*	Maximum Possible Number of ×8/×9 Groups	Maximum Possible Number of ×16/×18 Groups	Maximum Possible Number of ×32/×36 Groups
24 pins	4	2	1	0
32 pins	5	2	1	0
40 pins	6	3	1	0
48 pins	8	4	2	1

Note to Table 8–3:

(1) Some of the ×4 groups may use R_{UP}/R_{DN} pins. You cannot use these groups if you use the Stratix III calibrated OCT feature, as described in Table 8–1 on page 8–5.

Combining ×16/×18 DQS/DQ groups for ×36 QDR II+/QDR II SRAM Interface

This implementation combines two ×16/×18 DQS/DQ groups to interface with a ×36 QDR II+/QDR II SRAM device. The ×36 read data bus uses two ×16/×18 groups, while the ×36 write data uses another two ×16/×18 groups or four ×8/×9 groups. The CQ/CQn signal traces are split on the board trace to connect two pairs of DQS/CQn pins in the FPGA. This is the only connection on the board that you need to change for this implementation. Other QDR II+/QDR II SRAM interface rules for Stratix III devices also apply for this implementation.

Altera's ALTMEMPHY megafunction does not use the QVLD signal, so you can leave the QVLD signal unconnected as in any QDR II+/QDR II SRAM interfaces in the Stratix III devices.

For more information about the ALTMEMPHY megafunction, refer to the *ALTMEMPHY Megafunction User Guide*.

Rules to Combine Groups

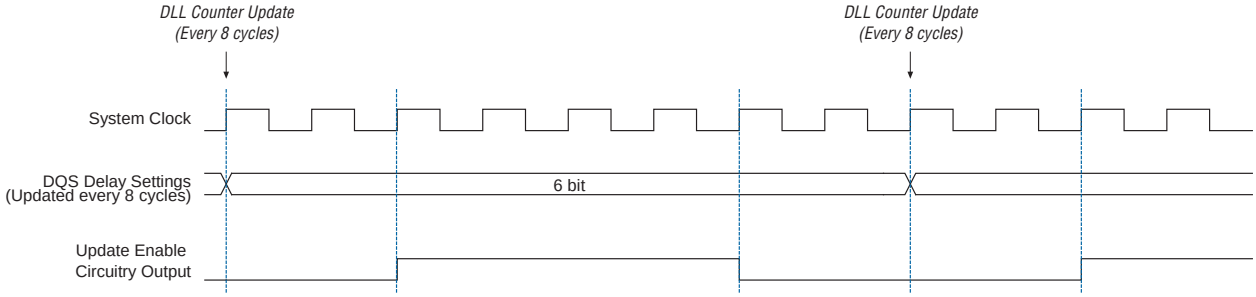
In 780- and 1152-pin package devices, there is at most one ×16/×18 group per I/O sub-bank. You can combine ×16/×18 groups from a single side of the device for a ×36 interface. For devices that do not have four ×16/×18 groups in a single side of the device to form two ×36 groups for read and write data, you can form one ×36 group on one side of the device, and another ×36 group on the other side of the device. For

You can also bypass the DQS delay chain to achieve 0° phase shift.

Update Enable Circuitry

Both the DQS delay settings and phase-offset settings pass through a register before going into the DQS delay chains. The registers are controlled by the update enable circuitry to allow enough time for any changes in the DQS delay setting bits to arrive at all the delay elements. This allows them to be adjusted at the same time. The update enable circuitry enables the registers to allow enough time for the DQS delay settings to travel from the DQS phase-shift circuitry or core logic to all the DQS logic blocks before the next change. It uses the input reference clock or a user clock from the core to generate the update enable output. The ALTMEMPHY megafunction uses this circuit by default. See Figure 8–14 for an example waveform of the update enable circuitry output.

Figure 8–14. Example of a DQS Update Enable Waveform



DQS Postamble Circuitry

For external memory interfaces that use a bi-directional read strobe like DDR3, DDR2, and DDR SDRAM, the DQS signal is low before going to or coming from a high-impedance state. The state where DQS is low, just after a high-impedance state, is called the preamble. The state where DQS is low, just before it returns to a high-impedance state, is called the postamble. There are preamble and postamble specifications for both read and write operations in DDR3, DDR2, and DDR SDRAM. The DQS postamble circuitry ensures that the data is not lost if there is noise on the DQS line during the end of a read operation that occurs while the DQS is in a postamble state.

Stratix III devices have a dedicated postamble register that you can control to ground the shifted DQS signal used to clock the DQ input registers at the end of a read operation. This ensures that any glitches on the DQS input signals at the end of the read postamble time do not affect the DQ IOE registers.

In addition to the dedicated postamble register, Stratix III devices also have an HDR block inside the postamble enable circuitry. These registers are used if the controller is running at half the frequency of the I/Os.

Table 8–12 lists the DQS configuration block bit sequence.

Table 8–12. DQS Configuration Block Bit Sequence

Bit	Bit Name
0..3	dqsbusoutdelaysetting[0..3]
4..6	dqsinputphasesetting[0..2]
7..10	dqsenablectrlphasesetting[0..3]
11..14	dqsoutputphasesetting[0..3]
15..18	dqoutputphasesetting[0..3]
19..22	resyncinputphasesetting[0..3]
23	dividerphasesetting
24	enaoctcycledelaysetting
25	enainputcycledelaysetting
26	enaoutputcycledelaysetting
27..29	dqsenabledelaysetting[0..2]
30..33	octdelaysetting1[0..3]
34..36	octdelaysetting2[0..2]
37	enadataoutbypass
38	enadqsenablephasetransferreg
39	enaoctphasetransferreg
40	enaoutputphasetransferreg
41	enainputphasetransferreg
42	resyncinputphaseinvert
43	dqsenablectrlphaseinvert
44	dqoutputphaseinvert
45	dqsoutputphaseinvert
46	dqsbusoutfinedelaysetting
47	dqsenablefinedelaysetting

IOE Features

This section briefly describes how OCT, programmable delay chains, programmable output delay, slew rate adjustment, and programmable drive strength are useful in memory interfaces.

For more information about the features listed below, refer to the *Stratix III Device I/O Features* chapter.

The DPA circuitry does not require a fixed training pattern to lock to the optimum phase out of the 8 phases. After reset or power up, the DPA circuitry requires transitions on the received data to lock to the optimum phase. The ALTLVDS megafunction provides an optional output port, `rx_dpa_locked` to indicate if the DPA has locked to the optimum phase. When the DPA locks to the optimum phase, the `rx_dpa_locked` signal always stays high unless you assert the `rx_reset` signal of the associated LVDS channel or the `pll_areset` signal of the receiver PLL providing the 8 DPA clock phases.

The `rx_dpa_locked` signal only indicates an initial DPA lock condition to the optimum phase after power up or reset. You must not use the `rx_dpa_locked` signal to validate the integrity of the LVDS link. Use error checkers, for example cyclical redundancy check (CRC) and diagonal interleave parity (DIP4), to validate the integrity of the LVDS link.

An independent reset port (`RX_RESET`) is available to reset the DPA circuitry. You must retrain the DPA circuitry after reset.

Soft-CDR Mode

The Stratix III LVDS channel offers the soft-CDR mode to support the Gigabit Ethernet/SGMII protocols. Clock-data recovery (CDR) is required to extract the clock out of the clock-embedded data to support SGMII. In Stratix III devices, the CDR circuit is implemented in soft-logic as an IP.

In soft-CDR mode, the DPA circuitry selects an optimal DPA clock phase to sample the data and carry on the bit-slip operation and deserialization. The selected DPA clock is also divided down by the deserialization factor, and then forwarded to the PLD core along with the deserialized data. The LVDS block has an output called `DIVCLKOUT` (`rx_divfwdclk` port of the ALTLVDS megafunction) for the forwarded clock signal. This signal is put on the newly introduced periphery clock (PCLK) network. When using soft-CDR mode, the `rx_reset` port should not be asserted when the `rx_dpa_lock` is asserted because the DPA will continually choose new phase taps from the PLL to track parts per million (PPM) differences between the reference clock and incoming data. In Stratix III devices, you can use every LVDS channel in soft-CDR mode and can drive the core via the PCLK network.

Table 11–14. Dedicated Configuration Pins on the Stratix III Device (Part 5 of 5)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
DCLK *(1)*	N/A	Synchronous configuration schemes (PS, FPP, AS)	Input (PS, FPP) Output (AS)	DCLK has an internal pull-up resistor (typically 25 kΩ) that is always active. In AS mode, DCLK is an output from the Stratix III device that provides timing for the configuration interface. After AS configuration, this pin is driven to an inactive state. In schemes that use a configuration device, DCLK will be driven low after configuration is done. In schemes that use a control host, DCLK should be driven either high or low, whichever is more convenient. Toggling this pin after configuration does not affect the configured device.
DATA0 *(1)*	N/A in AS mode. I/O in PS or FPP mode	PS, FPP, AS	Input	Data input. In serial configuration modes, bit-wide configuration data is presented to the target device on the DATA0 pin. In AS mode, DATA0 has an internal pull-up resistor that is always active. After PS or FPP configuration, DATA0 is available as a user I/O pin and the state of this pin depends on the **Dual-Purpose Pin** settings.
DATA[7..1]	I/O	Parallel configuration schemes (FPP)	Inputs	Data inputs. Byte-wide configuration data is presented to the target device on DATA[7..0]. In serial configuration schemes, they function as user I/O pins during configuration, which means they are tri-stated. After configuration, DATA[7..1] are available as user I/O pins and the state of these pin depends on the **Dual-Purpose Pin** settings.

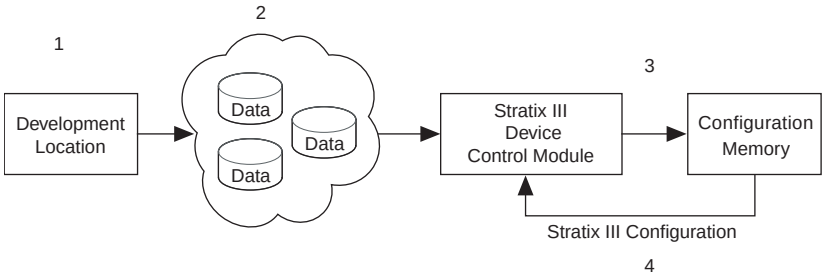
Note to Table 11–14:

(1) To tri-state AS configuration pins in AS configuration scheme, turn on **Enable input tri-state on active configuration pins in user mode** option from the **Device and Pin Options** dialog box. This tri-states DCLK, nCSO, Data0, and ASDO pins. Dual-purpose Pins Setting for Data0 is ignored. To set Data0 to a different setting, for example to use Data0 pin as a regular I/O in user mode, turn off **Enable input tri-state on active configuration pins in user mode** option and set your desired setting from the Dual-purpose Pins Setting menu.

3. The Nios II processor (or user logic) initiates a reconfiguration cycle with the new or updated configuration data.
4. The dedicated remote system upgrade circuitry detects and recovers from any error(s) that might occur during or after the reconfiguration cycle, and provides error status information to the user design.

Figure 12–1 shows the steps required for performing remote configuration updates. (The numbers in the figure below coincide with the steps above.)

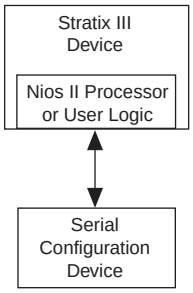
Figure 12–1. Functional Diagram of Stratix III Remote System Upgrade



Stratix III devices only support remote system upgrade in the single device Fast AS configuration scheme.

Figure 12–2 shows the block diagrams for implementing a remote system upgrade with the Stratix III Fast AS configuration scheme.

Figure 12–2. Remote System Upgrade Block Diagram for Stratix III Fast AS Configuration Scheme



You must set the mode select pins (`MSEL[2..0]`) to Fast AS mode to use the remote system upgrade in your system. Table 12–1 lists the `MSEL` pin settings for Stratix III devices in standard configuration mode and remote system upgrade mode. The following sections describe the remote update of remote system upgrade mode.

For more information about standard configuration schemes supported in Stratix III devices, refer to the *Configuring Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook*.

IEEE Std. 1149.1 Boundary-Scan Register

The boundary-scan register is a large serial shift register that uses the TDI pin as an input and the TDO pin as an output. The boundary-scan register consists of three-bit peripheral elements that are associated with Stratix III I/O pins. You can use the boundary-scan register to test external pin connections or to capture internal data.

For the Stratix III family device boundary-scan register lengths, refer to the *Configuring Stratix III Devices* chapter in volume 1 of the *Stratix III Device Handbook*.

Figure 13–3 shows how test data is serially shifted around the periphery of the IEEE Std. 1149.1 device.

Figure 13–3. Boundary-Scan Register

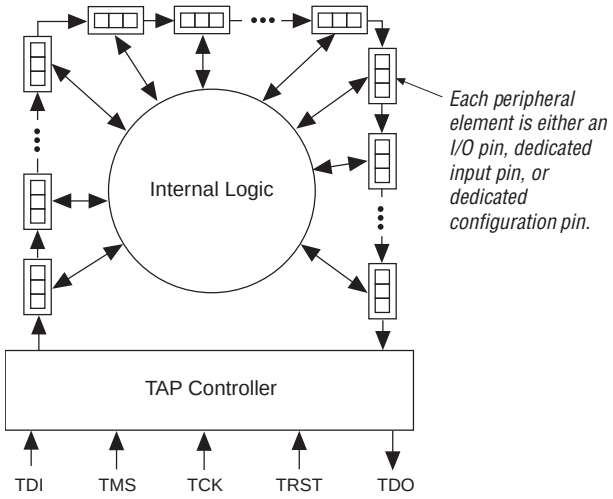


Table 13–2 lists the boundary-scan register length for Stratix III devices.

Table 13–2. Stratix III Boundary-Scan Register Length

Device	Boundary-Scan Register Length
EP3SL50	1506
EP3SL70	1506
EP3SL110	2274
EP3SL150ES	2274
EP3SL150	2274
EP3SL200	2970
EP3SL340	3402
EP3SE50	1506
EP3SE80	2274
EP3SE110	2274
EP3SE260	2970

- Perform a SAMPLE/PRELOAD test cycle prior to the first EXTEST test cycle to ensure that known data is present at the device pins when you enter the EXTEST mode. If the OEJ update register contains a 0, the data in the OUTJ update register is driven out. The state must be known and correct to avoid contention with other devices in the system.
- Do not perform EXTEST testing during ICR. This instruction is supported before or after ICR, but not during ICR. Use the CONFIG_IO instruction to interrupt configuration and then perform testing, or wait for configuration to complete.
- If performing testing before configuration, hold the nCONFIG pin low.

For more information about boundary scan testing, contact Altera® Application at www.altera.com.

Boundary-Scan Description Language (BSDL) Support

The Boundary-Scan Description Language (BSDL), a subset of VHDL, provides a syntax that allows you to describe the features of an IEEE Std. 1149.1 BST-capable device that can be tested. Test software development systems then use the BSDL files for test generation, analysis, and failure diagnostics.

For more information about BSDL files for IEEE Std. 1149.1-compliant Stratix III devices, refer to the *Stratix III BSDL Files* page on the Altera website.

To perform BST on a configured device, you will require a post configuration BSDL file that is customized to your design. This file can be generated with the BSDL Customizer script.

For more information about the BSDL Customizer, refer to the *Altera BSDL Support* page on the Altera website.

Section VI. Packaging Information

This section provides packaging information for the Stratix® III device.

- Chapter 17, Stratix III Device Packaging Information

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the full handbook.