LatticeSC QDRII/II+ SRAM Memory Interface User's Guide

November 2007 Technical Note TN1096

Introduction

Among the most daunting challenges faced by the FPGA designer is the efficient transport of data to external memories. Current applications require large I/O channel bandwidths. In response to these demands, the industry has defined several new memory devices with their associated protocols (e.g., QDR, DDR, RLDRAM), each being optimized for a particular segment of the high-bandwidth market. This User Guide discusses a memory interface for a Second-Generation Quad-Data-Rate SRAM (QDRII/II+ SRAM), implemented in the LatticeSC™ FPGA.

The two data buses (write and read) share a common address bus. The two addresses are independent, and therefore are time-multiplexed. QDRII/II+ SRAMS offer two architectures to provide this: 4-word burst and 2-word burst. The 4-word burst option is simpler and more robust, since the addresses toggle at single-data-rate (SDR). The 2-word burst option addresses, on the other hand, toggle at double-data-rate (DDR), but provide 2X finer granularity. The LatticeSC family can support both options, but this user's guide concentrates on the simpler and more popular 4-word burst device, since granularity is not typically an issue.

QDRII/II+ SDRAM Description

History

The following is a brief timeline of the major events in the emergence of the QDRII/II+ SRAM:

- QDR Consortium Formed – Feb '99
- QDR-I Specifications Released – 2H99
- QDR-II Defined – 2H00
- QDR-I 9Mb Sampled – 1H01
- NEC Joins QDR Consortium – 1Q01
- Samsung Joins QDR Consortium – 2Q01
- Hitachi Signs LOI to Join QDR Consortium – 3Q01
- QDR-II Specifications Released – 4Q01
- QDR-II 18Mb Sampled – 4Q01

Specifications and Performance

The QDRII/II+ SRAM is targeted to applications requiring:

- Very high bandwidth
- Low latency
- Ratio of reads to writes of approximately one-to-one

Table 1 lists some of the features of the QDRII/II+ SRAM solution:

Table 1. QDRII/II+ SRAM Features

QDRII/II+ SRAM Feature	Value	Units
Max Frequency	300	MHz
DLL	Yes (optional)	—
Min DLL Frequency	119	MHz
Initial Latency	1.5	Clocks
Echo Clocks	Yes	—
Max Address Size	19 (2-word burst) 20 (4-word burst)	Bits
Data Width	18/36	Bits
Supply Voltage	1.8	Volts
I/O Protocol	HSTL-15, HSTL-18	—
Refresh Cycles Needed?	No	—

Implementation Challenges

The most difficult aspect of a QDRII/II+ SRAM Memory Interface implementation is the read bus clock design. This clock, CQ/CQ#, is generated by the QDRII/II+ SRAM, and it is in phase with the read data; that is, the clock and the data transitions occur concurrently. The Memory Interface is charged with the task of properly latching the data, based on the CQ clock's transitions.

In order to accomplish this, while achieving maximum timing margins, it is necessary to shift the clock edge to the center of the "eye" (the interval between successive data transitions), which evenly distributes available margin between setup and hold requirements. LatticeSC devices provide PLL/DLL functions that can dynamically perform this operation easily, simply and efficiently.

LatticeSC Features That Solve the Implementation Challenges

The LatticeSC family provides many features that make high-speed systems easier to build. Among them are:

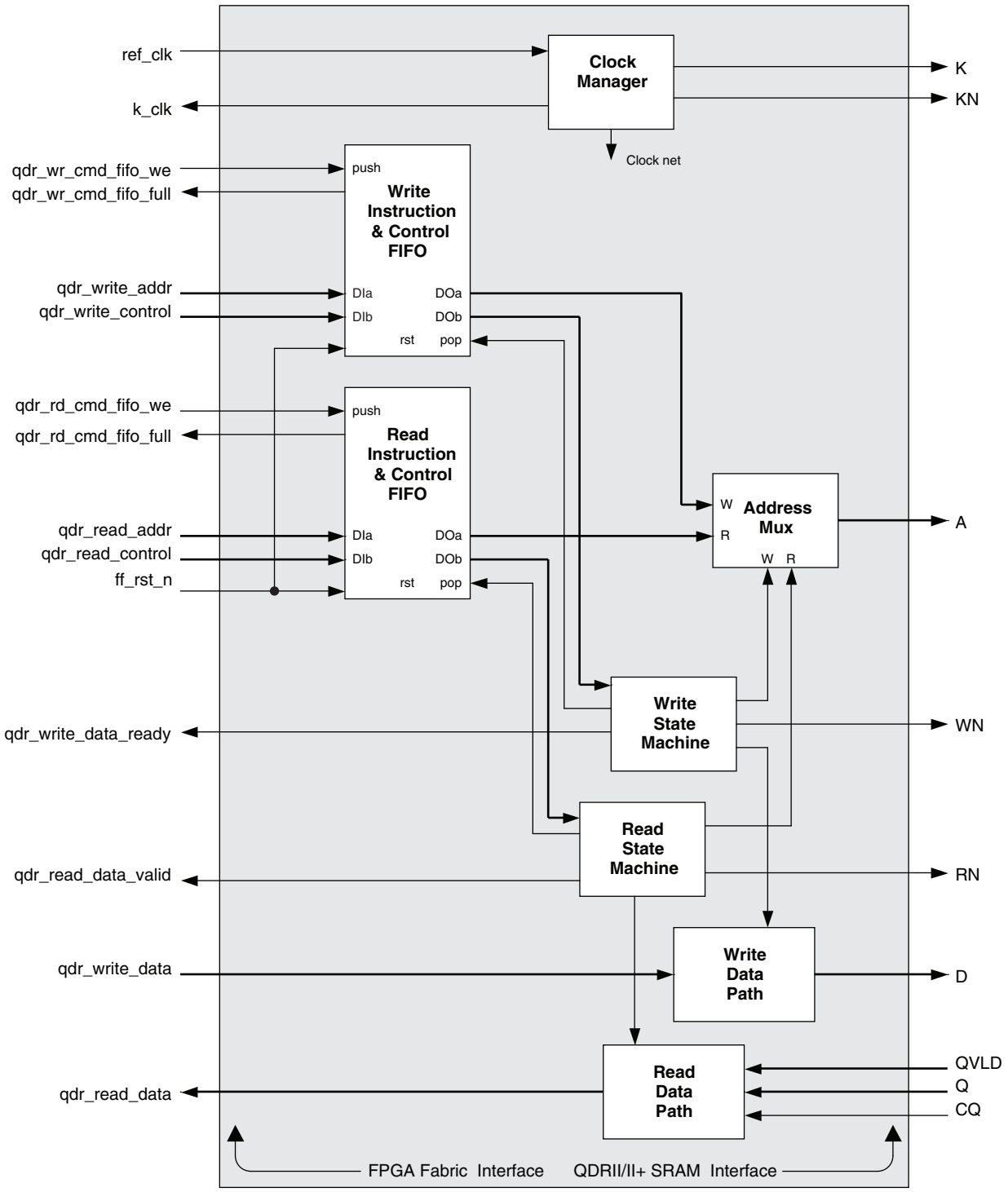
- Per-pin DDR (Double-Data-Rate) capability built in. Usable in both DDR and QDR (Quad-Data-Rate) implementations.
- Per-pin DDR (Double-Data-Rate) incorporates hardwired serial-to-parallel and parallel-to-serial conversion. Useful in SPI4 implementations.
- I/Os that support SSTL (Stub-Series Terminated Logic), used with high-speed DDR SDRAM devices, and HSTL (High-Speed Transceiver Logic), used with high-speed SRAM devices.
- Optional delays built into the I/O paths that can be statically or dynamically set on a per-pin or bus basis. The I/O pin delay blocks match the delay blocks that are internal to the DLLs, so that the DLLs can adaptively determine the optimum delay setting, and then apply that setting to the I/O pin delay blocks. In this way, the DLL can compensate for effects of frequency, voltage, temperature and device-to-device variation.
- PLLs that can generate two high-speed clocks that are in quadrature (90° out of phase with each other).
- Integrated and calibrated I/O pin input parallel terminations that can be used to match the 50-60 Ohm traces typically found on printed circuit boards over process, voltage and temperature variation.
- Integrated and calibrated I/O pin output serial terminations that allow output buffers to match the 50-60 Ohm traces typically found on printed circuit boards over process, voltage and temperature variation.
- High-speed FPGA fabric.

LatticeSC QDRII/II+ Memory Interface Implementation Details

Block Diagram

Figure 1 shows the block diagram for a typical QDRII/II+ Memory Interface implemented in the LatticeSC FPGA.

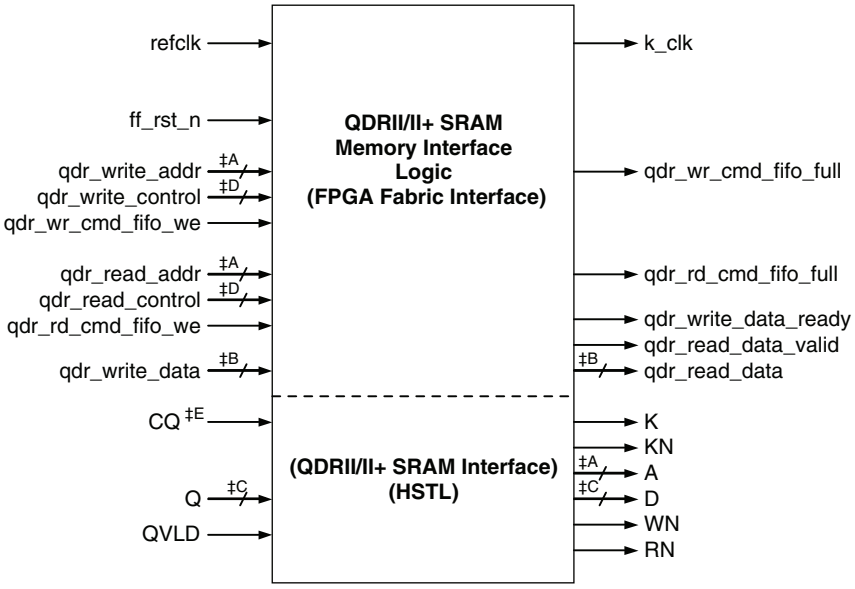
Figure 1. Block Diagram (Typical)



Pinout

Figure 2 shows the pinout for a typical QDRII/II+ Memory Interface implemented in the LatticeSC FPGA. Inputs are on the left, and outputs on the right. The internal FPGA interface is on top, and the external HSTL interface is on the bottom.

Figure 2. QDRII/II+ SRAM Memory Interface Pinout Interface (Typical)



Notes:
- ‡A – 17-20 bits wide
- ‡B – 36 or 72 bits wide
- ‡C – 18 or 36 bits wide
- ‡D – Number and meaning of these signals is implementation-dependent
- ‡E – CQ# is typically not used; instead, data is strobed on both the rising and falling edges of CQ.

Signal List

Table 2 below lists the basic set of interface signals and gives their descriptions.

Table 2. Interface Pin Table

Pin Name	Direction	Width	Description
refclk	In From FPGA	1	Input clock. This input feeds a PLL which supplies all clocks to the unit and the QDRII/II+ SRAM. The PLL may provide frequency multiplication/division.
qdr_rst_n			Active-LOW asynchronous reset
qdr_write_addr		17-20	Write address
qdr_write_control		*	Write control signals. *Note: Number and content is implementation-dependent.
qdr_wr_cmd_fifo_we		1	Accompanies valid data on "qdr_write_addr" and "qdr_write_control".
qdr_read_addr		17-20	Read address.
qdr_read_control		*	Read control signals. *Note: Number and content is implementation-dependent.
qdr_rd_cmd_fifo_we		1	Accompanies valid data on "qdr_read_addr" and "qdr_read_control".
qdr_write_data		36/72	Write data bus.
k_clk	Out to FPGA	1	This is a copy of the unit's internal clock, derived from input signal "refclk".
qdr_wr_cmd_fifo_full		1	Reports the full state of the Write Instruction FIFO.
qdr_rd_cmd_fifo_full		1	Reports the full state of the Read Instruction FIFO.
qdr_write_data_ready		1	Signals that the Memory Interface is accepting the data currently on bus "qdr_write_data".
qdr_read_data_valid		1	Accompanies valid data on bus "qdr_read_data"
qdr_read_data		36/72	Read data bus.
K	Out to SRAM	2	QDRII/II+ SRAM's input clocks "K" and "K#". K is the Memory Interface unit's clock, delayed 90°. KN is the inverse of K.
KN			
A		17-20	QDRII/II+ SRAM's address bus "A". Width is 17 bits in 4-word burst mode, 18 bits in 2-word burst mode.
D		18/36	QDRII/II+ SRAM's write data bus "D".
WN		1	QDRII/II+ SRAM's active-LOW write enable "W#".
RN		1	QDRII/II+ SRAM's active-LOW read enable "R#".
CQ	In from SRAM	1	CQ is the clock for DDR bus "Q". Note: the CQ# signal from the QDRII/II+ SRAM is not used. Instead, both the rising and falling edges of CQ are used to clock incoming data.
Q, QVLD		18/36	QDR_II SRAM's read data bus "Q" and accompanying valid signal "QVLD."

Description of Operation

The QDRII/II+ SRAM features independent read and write buses and control. Also, because the memory is static, there is no need to allocate bus cycles for refresh. Together, this means that the memory is always available for both read and write operations. The only interdependences (for 4-word-burst QDR-SRAMS only) are (1) new operations must of course wait for a previous burst of the same type (read vs. write) to complete, and (2) the read and write operations must share the address bus, so they cannot start on the same clock. Otherwise, there is nothing that can impede the fulfillment of a read or write request.

The first interdependency is satisfied if access requests skip every other clock cycle. The second is satisfied if the reads and writes alternate, each utilizing the address bus during the clock that the other is inactive.

Once this "cadence" is established, data can flow at a constant 100% bandwidth rate, on both the read and write bus. For most applications, the memory must be able to handle sustained maximum-bandwidth transfers, so the memory bandwidth must be equal to or greater than the channel bandwidth. In this case, FIFOs on the read and

write channels are needed only for clock domain transition and buffering of the few data words needed as the channel meshes with the memory's cadence.

A memory interface would typically perform this buffering. Additionally, a memory interface may perform DMA functions, such as extending burst lengths to accommodate larger blocks of read or write data, and buffering multiple block read/write instructions. This instruction buffering also provides the look-ahead necessary to maintain 100% bandwidth operation (i.e., no idle cycles).

Write

A typical write sequence, as it appears on the pins of the QDRII/II+ SRAM, is shown in Figures 3 (4-word burst) and 7 (2-word burst). The accompanying read operation represents the earliest read that will return the data just written by the write operation.

For a 2-word burst device, a write is indicated by signal W# being active LOW during a rising edge on clock K. The first write data word is presented on bus D during this clock edge as well, but the write address is not presented until 1/2 clock later, on the rising edge of clock K#, when the second (final) write data word is also presented on bus D.

For a 4-word burst device, a write is also indicated by signal W# being active LOW during a rising edge on clock K. The write address is also presented during that same clock edge. The first of four write data words is received on the subsequent rising edge of clock K, with the final three write data words being received on the following three rising edges on K#, K and K# respectively.

In a typical memory interface application, a write operation is initiated by driving signal qdr_wcmd_fifo_wenab active HI, and at the same time providing the address on bus qdr_write_addr and the block length on bus qdr_write_block_length. This operation will be held in the write instruction FIFO until its turn comes up, and then the data will be written. The first write data must be waiting on bus qdr_write_data, ready to be accepted, as early as five cycles after qdr_wcmd_fifo_wenab is asserted. When the data is accepted, as indicated by the assertion of qdr_write_data_ready, the second word must be presented on the following cycle. If there are additional bursts of data, they will be accepted on subsequent cycles without interruption until the block is completed. If multiple write operations are stacked up in the write instruction FIFO, the corresponding data will be taken in the same order. If the write instruction FIFO is kept from emptying or new writes are initiated at the same rate as they are executed, then uninterrupted full-bandwidth writes will continue indefinitely.

Read

A typical read sequence, as it appears on the pins of the QDRII/II+ SRAM, is also shown in Figures 3 (4-word burst) and 7 (2-word burst).

For a 2-word burst device, a read is indicated by signal R# being active LOW during a rising edge on clock K. The read address is presented on bus D during this clock edge as well. The first and second read data words are returned coincident with the falling and rising edges on echo clock CQ (note that the falling edge of CQ occurs at the same time as the rising edge on CQ#), starting with the falling edge of CQ that occurs 21/2 cycles after the rising edge of K that clocks in R# LOW.

For a 4-word burst device, a read is also indicated by signal R# being active LOW during a rising edge on clock K. The read address is also presented during that same clock edge. The four read data words are returned coincident with the falling and rising edges on echo clock CQ (note that the falling edge of CQ occurs at the same time as the rising edge on CQ#), starting with the falling edge of CQ that occurs 11/2 cycles after the rising edge of K that clocks in R# LOW.

The first of four read data words is received on the subsequent rising edge of clock K, with the final three read data words being received on the following three rising edges on K#, K and K# respectively.

A read operation is initiated by driving signal qdr_rcmd_fifo_wenab active HI, and at the same time providing the address on bus qdr_read_addr and the block length on bus qdr_read_block_length. This operation will be held in the read instruction FIFO until its turn comes up, and then the requested data will be read. As the read data

becomes available, it will be presented on bus qdr_read_data, accompanied by asserted qdr_read_data_valid. The data for the entire block will be provided uninterrupted. As with writes, if multiple read operations are stored in the read instruction FIFO, the data will be accessed and presented in the same order as requested. If the read instruction FIFO is kept from emptying or new reads are initiated at the same rate as they are executed, then uninterrupted full-bandwidth reads will continue indefinitely in 4-word burst mode.

Fulls and Empties

Both the write and read interfaces utilize fulls and empties to facilitate the throttling of the loading of FIFOs. The signals are similar between the two interfaces.

A full signal informs the FPGA logic that the respective FIFO cannot accept another instruction. A memory interface implementation may also provide an "almost full" indication that enables the pipelined flow to be stopped in time to avoid overflow, while still allowing full-speed pipelined operation. An empty indication may not be provided, since it is primarily used on the memory interface's side of the FIFO to control its emptying, but may be provided to allow the FPGA logic to detect the idle condition.

Figure 3. QDRII/II+ SRAM Interface Timing (4-Word Burst Mode)

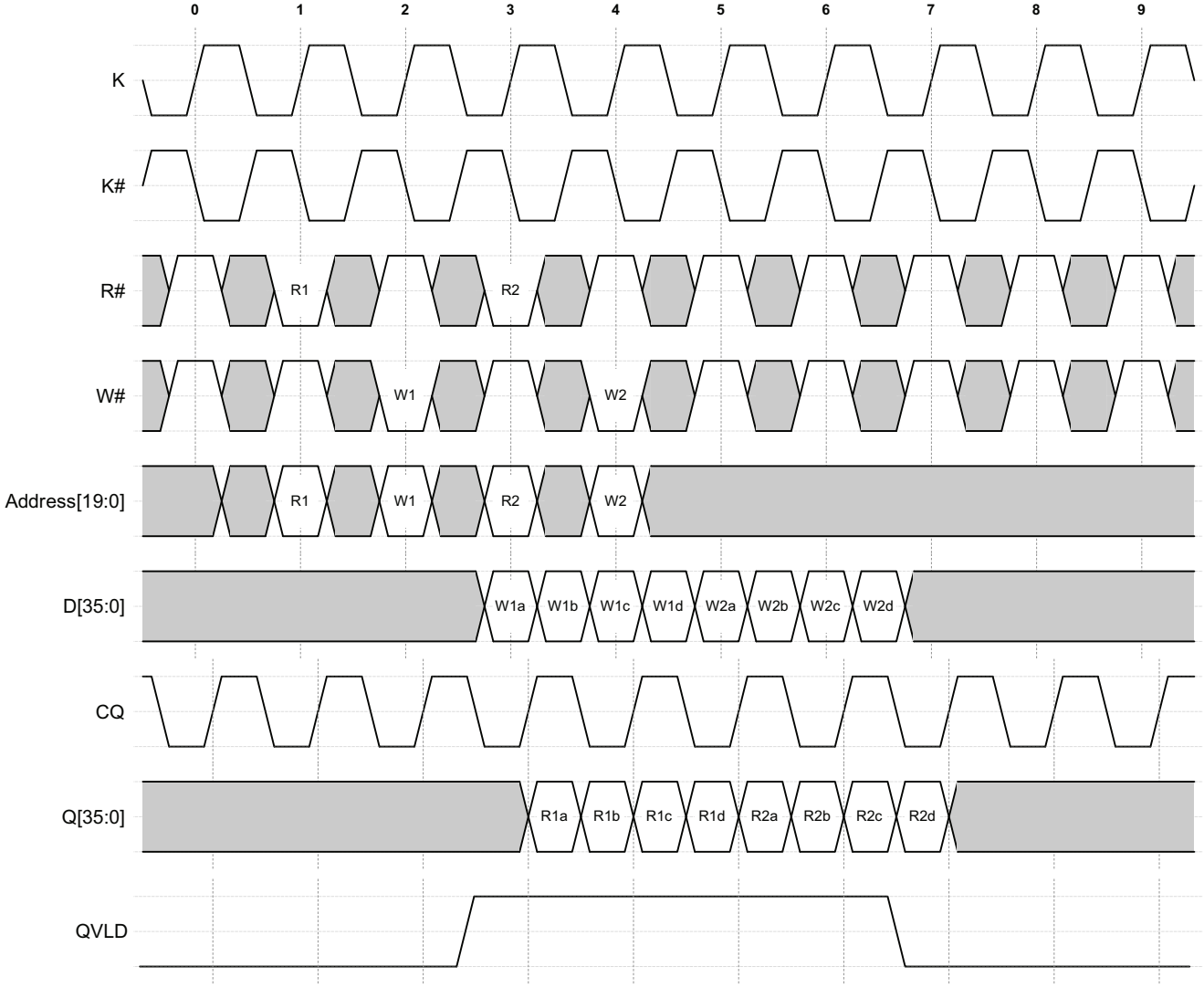


Figure 4. QDRII/II+ SRAM Interface Timing (2-Word Burst Mode)

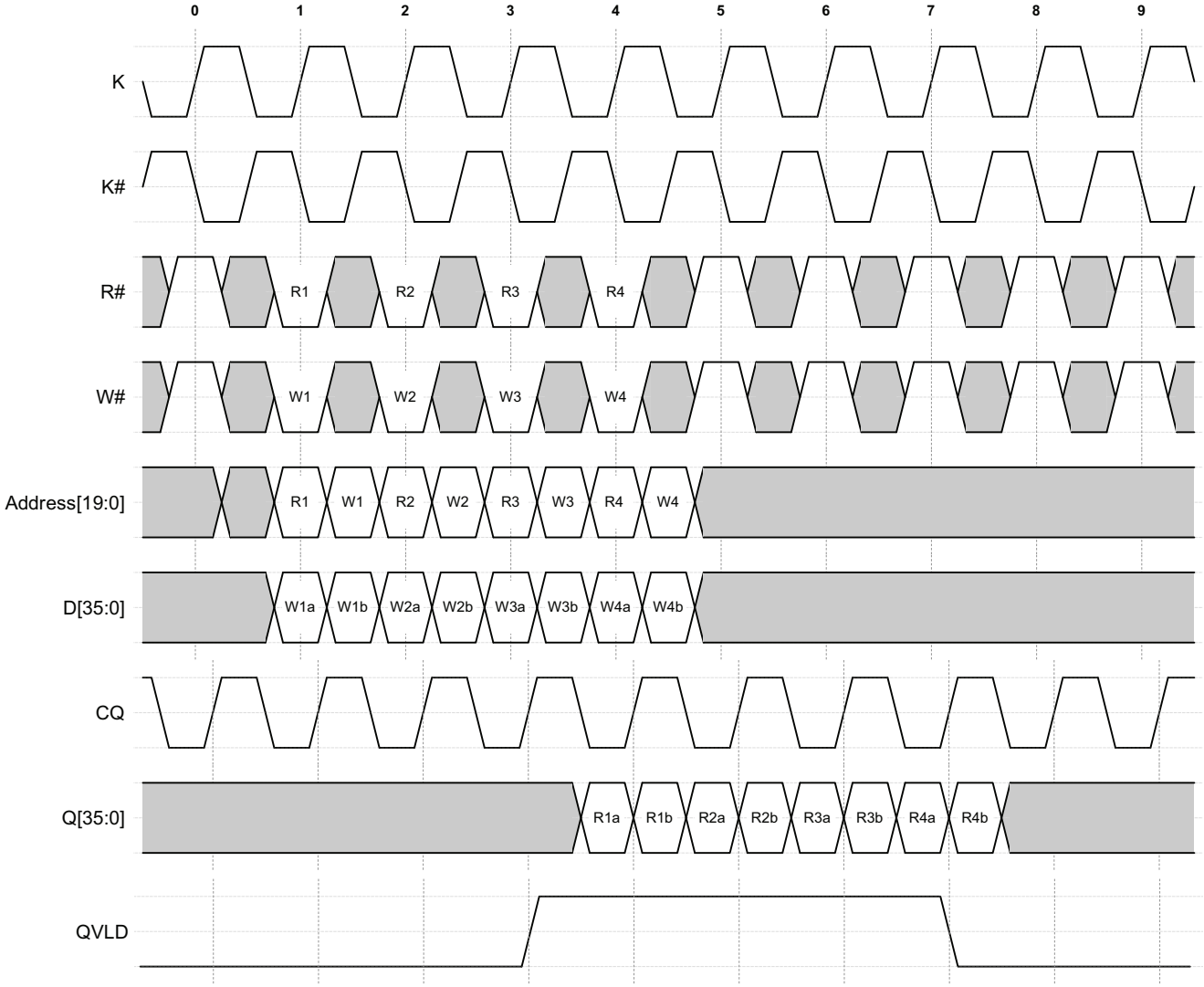


Figure 5. Complete 4-Word Write/Read Sequence, Reading Back Just-Written Data

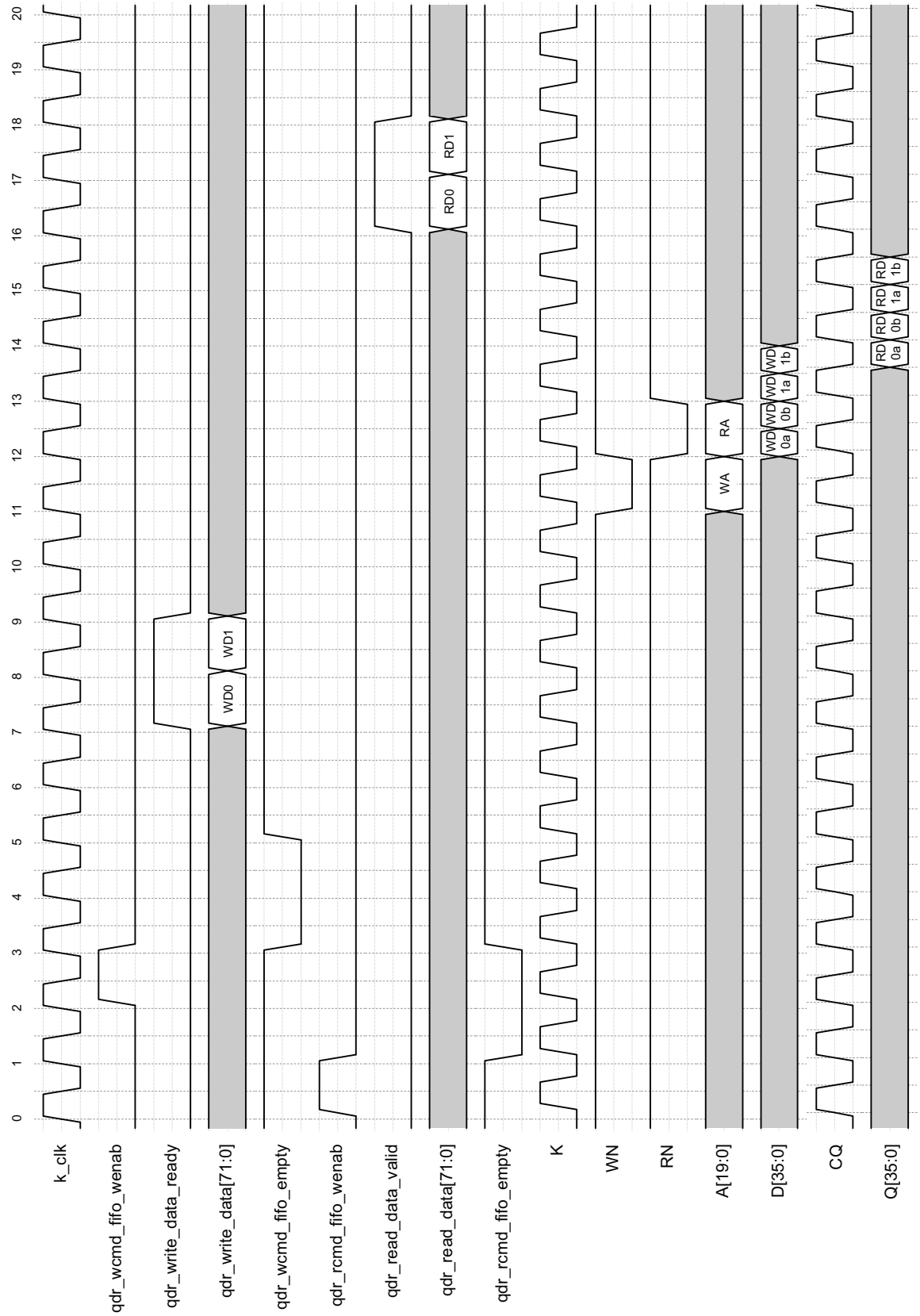
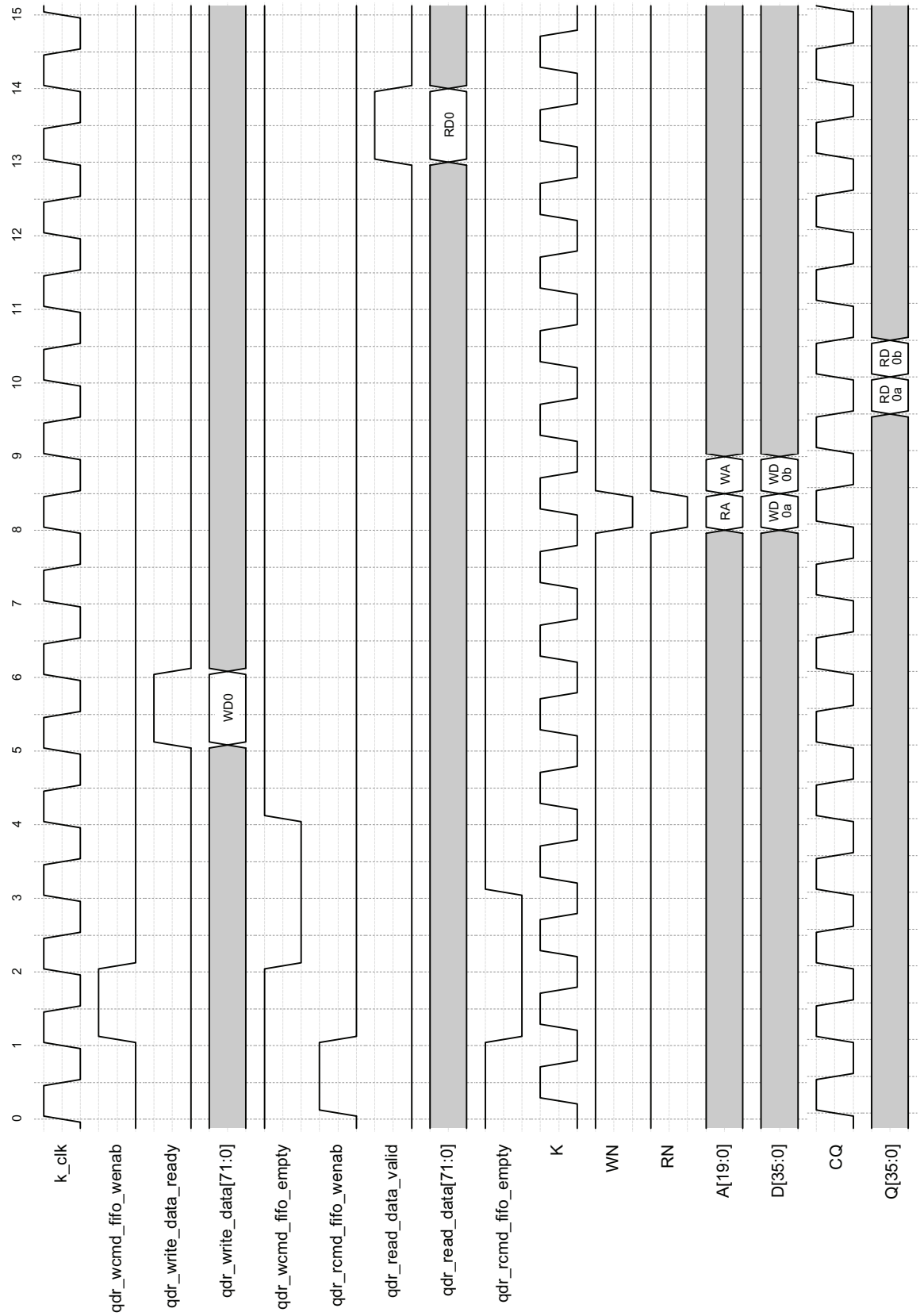


Figure 6. Complete 2-Word Write/Read Sequence, Reading Back Just-Written Data



Write/Read

The timing of a single basic write/read pair of operations having block lengths of one is detailed in Figure 5 (4-word burst) and Figure 6 (2-word burst). Both figures depict a write followed by a read in which the initial state is idle (both the read and write FIFOs are empty), and the read is the earliest that reads back the data just written.

QDRII/II+ SRAM Clocks: K/K#, C/C# and CQ/CQ#

The QDRII/II+ SRAM utilizes three clock pairs:

1. **K/K#** – The K input clock accompanies the input signals (A, D, W# and R#) to the SRAM. It clocks these signals into the device and drives the internal logic. This clock is in a quadrature relationship with its data; that is, it is 90° out of phase. As such, the clock edge transitions are centered in the middle of the data "eye".
2. **C/C#** – The C input clock is synchronous to the K input clock, but can differ slightly in phase by a constant amount, within specified limits. The read output clock CQ for data bus Q is normally aligned to this C clock. This allows the timing of the Q bus to be adjusted - to align multiple SRAM devices, for example. If this capability is not necessary, as in the present case, the C clock can be disabled by driving both C and C# to a constant HIGH. The CQ clock is then aligned to the K clock.
3. **CQ/CQ#** – The CQ output clock, also referred to as the echo clock, is generated by the SRAM to accompany the Q output data bus, and its edge timing is similar or identical to that of the Q bus, not in a quadrature relationship as is the case for the K clock. Therefore, the LatticeSC FPGA is required to effectively shift the CQ clock by 90° before using it to capture the data on the Q bus. The current LatticeSC design derives clocks for both phases of the DDR Q data bus from the CQ clock, and does not utilize the CQ# clock.

LatticeSC I/O Buffers

As can be seen in Figure 8, the QDRII/II+ SRAM Memory Interface utilizes DDR HSTL I/O buffers for both the outputs (for the clock, address, write data and control), and the inputs (for the read data and echo clock). Since the outputs' parallel to serial conversion and the inputs' serial to parallel conversion is performed right in the I/O buffers, there is no need for the FPGA fabric to be able to handle data at the doubled clock rate.

LatticeSC I/O buffers that are used as inputs (input-only or bi-directional) feature the ability to provide internal termination. Two termination configurations are available (Figure 7):

a. Termination directly to VTT via a 60-Ohm impedance.

b. Termination via a Thevenin-equivalent 50-Ohm network across VDDIO and VSS.

Figure 7. LatticeSC Input Buffer Configurations

(a)
Termination to VTT

(b)
Termination to VDDIO/VSS

The input buffers on LatticeSC devices can also insert a predetermined delay. This can be used to align the bits on a data bus, or to align the entire bus to its clock. Individual bits can be assigned a constant delay, and all bits in a bank can be assigned a common dynamic delay value.

The constant delay is used in the QDRII/II+ SRAM Memory Interface design to compensate for varying input delays across the bits in the Read Data Bus Q. The dynamic delay is used to shift the clock for that bus, CQ, so that it is centered in the eye of the data bus, as described in the next section.

In addition to the input buffer features discussed above, the output buffers are configurable to provide either 50 Ω (HSTL-I) or 25 Ω (HSTL-II) output impedance, which is internally compensated for variations in voltage, temperature and device processing.

Clocking Challenges and Solutions

Figure 8 illustrates the clocking network. Several unique features of the LatticeSC architecture are utilized in this design. A PLL [1] is used to perform frequency multiplication of the input clock "refclk", and at the same time to generate a second clock that is 90° lagging, so that the clocks "K" and "K#" to the QDRII/II+ SRAM can transition in the center of the data eye of bus "D".

Both "k_clk" and "k_clk shifted 90°" are typically routed on primary clock nets so that there is very little skew from the ideal 90° offset. The clocks "K" and "K#" are then generated using the same DDR output buffer elements as are used in the buffers for output data and control signals, so that once again very little skew is introduced. These two clocks are generated by simply sending a constant "10" pattern on outputs that are in every other respect identical to the data and control outputs.

A Valid Timing Chain [2] generates a data valid signal at the correct time to line up with the returning read data by duplicating the latency in the external QDRII/II+ SRAM and board routing. This is necessary because there is nothing returned from the QDRII/II+ SRAM with the read data to identify the valid data. Note that for 2-word bursts, the valid is asserted for one full clock (two half-clocks), and for 4-word bursts, it is asserted for two full clocks (four half-clocks). Note also that the number of registers in the timing chain varies to match the read latency (1.5, 2.0 or 2.5) of the QDRII/II+ SRAM. The Valid Timing Chain straddles two clock domains having the same frequency but different phases, and performs the clock domain transition between them. The phase difference represents all the cumulative delays in the external path: board trace delays (in both directions), and delay from K/K# to CQ/CQ#. The clocking scheme described here can accommodate and compensate for approximately 1/2 clock cycle of variation in this delay.

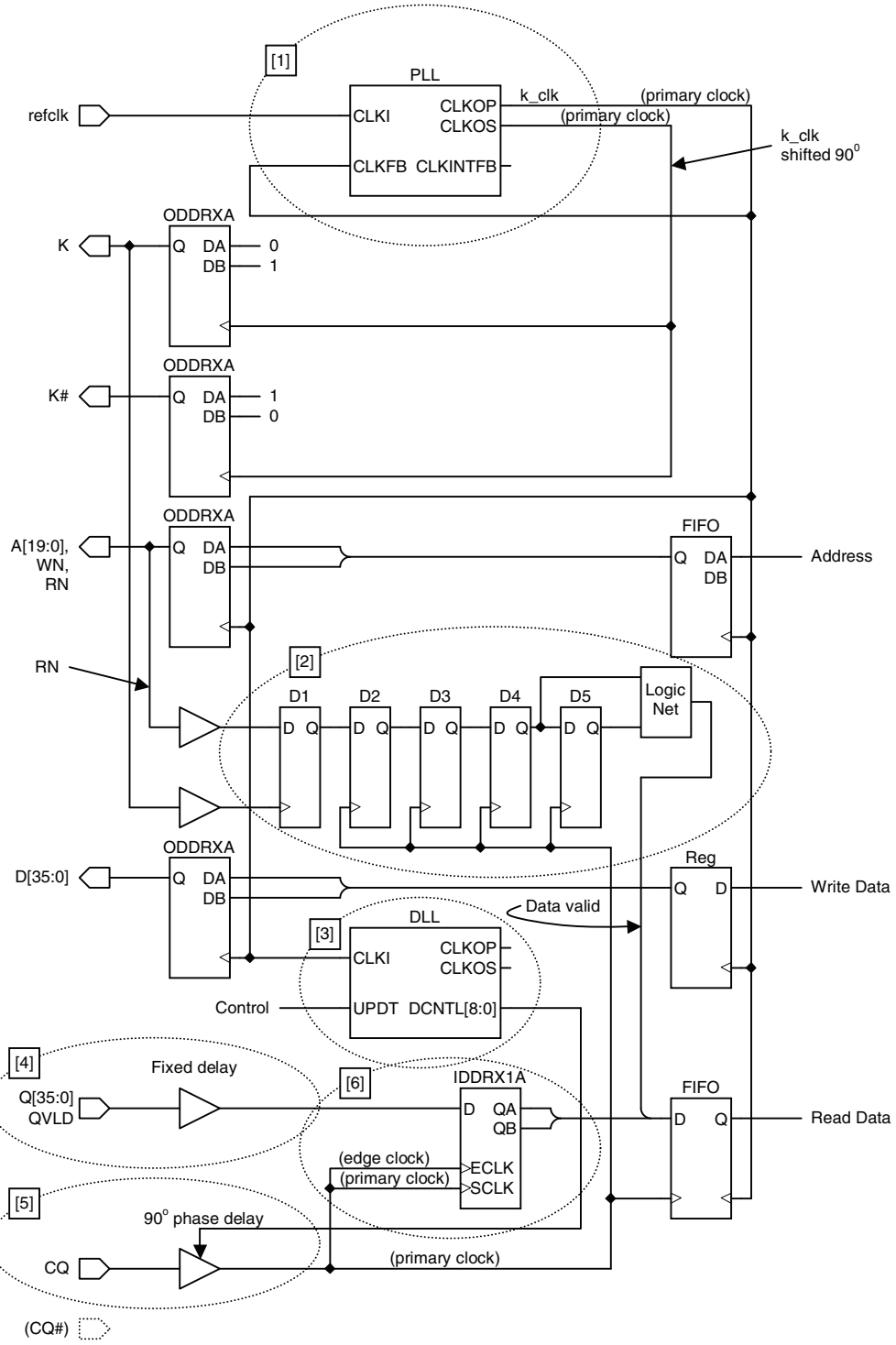
The input registers for the read data bus "Q" and signal "QVLD" [6] require some special clocking, and this need is handled by special hardware capability. The input bus registers have two clock inputs. The first, ECLK, is fed by the edge clock, to receive the data at the earliest time, since the edge clock net has less delay and skew to the I/O registers than the primary clock net. But if this data were to be sent directly to a register clocked by the primary clock, the receiving register's input hold time could be violated. Therefore, the input register also takes a second input clock, SCLK, which is fed the primary clock. The register does not output the data until this clock's edge, avoiding any hold time issues. This clock domain transfer mechanism is built-in to the LatticeSC input buffers, thus allowing operation at highest rates of speed.

For the return read data bus "Q" and its accompanying valid signal QVLD, a DLL [3] is employed to dynamically generate a value that determines the proper delay to cause an effective 90° phase shift on CQ's input buffer [5], so that it too is positioned in the center of the data it captures. This takes advantage of the fact that the DLL and the input buffers contain matching delay blocks, so that the delay selection value generated in the DLL when it generates a 90° shifted clock can also be used in the input buffer to cause the same phase shift. A 9-bit digital bus communicates this delay selection value from the DLL to the "CQ" input buffer.

The read data is then typically transferred from the "CQ" clock domain to the internal clock domain with the assistance of a synchronous FIFO.

For the "Q" data bus and signal QVLD, the delay elements [4] are also used in an "Edge Clock Injection Match" mode. This compensates for the edge clock routing of the "CQ" input, thereby providing an optimal read data edge. Manual changes to the input delay can also be made to each individual "Q" input pin to compensate for differences in board trace delays.

Figure 8. QDRII/II+ SRAM Memory Interface Clock and Data Paths



Performance

Timing Diagrams

Unit timing is shown in Figures 3 to 6. Refer to the read and write operation sections above for details and descriptions.

Timing Budgets

The timing on the signals interconnecting the LatticeSC Memory Interface and the QDRII/II+ SRAM can be difficult to manage, since there are several unusual components to the timing budget. These signals fall into two groups: the signals going to the SRAM under clocking by clock K/K#, and the return signals from the SRAM under clocking by CQ.

All LatticeSC timing specifications apply to all speed grades over process/voltage/temperature variation. The values are subject to change. See the LatticeSC Family Data Sheet for the latest values.

The first group of signals is depicted in Figure 5-1, and the timing budget is calculated in Table 3. There are seven components to the timing budget: [1] clock period, [2] duty cycle distortion (DCD), [3] K/K# clock jitter, [4] PLL phase error skew, [5] package skew, [6] board trace skew, and [7] SRAM receive register input setup/hold.

Parameters marked [a] are LatticeSC device parameters and are valid over all the device's specified operating conditions and speed grades, [b] is a characteristic of the PC board, and [c] are QDRII/II+ SRAM device parameters. Values given for [b] and [c] are typical and must be adjusted for the specific PCB and QDRII/II+ SRAM chosen.

DCD and PLL phase error skew are both significant because they both tend to displace the clocks in the eye, and because of the fact that both edges of the clock are critical for capturing data.

Figure 5-1. Timing Path, DDR Write

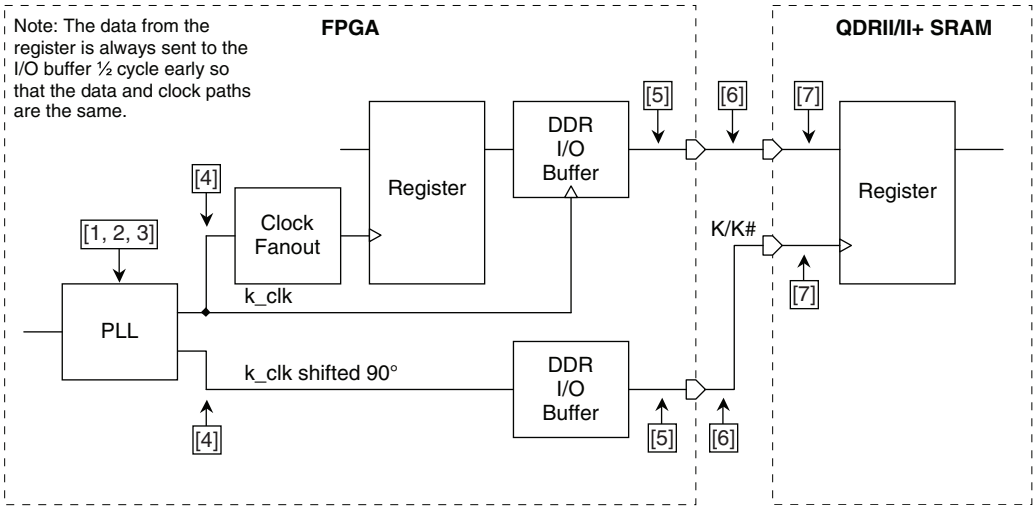


Table 3. Timing Budget – DDR Write Data Path (Preliminary)

Parameter		Units	Value	Long Path	Short Path		Description
fclk		MHz	200				Clock Frequency
tclk		psec	5000				Clock Period
tclk-ddr	[1]	psec	2500				DDR Clock Period (1/2 * tclk)
tdcd	[2]	psec	±125	+125	-125	[a]	Clock Duty Cycle Distortion (45%-55%)
tcj	[3]	psec	±50	+50	-50	[a]	Clock Jitter (0.02 UI)
tphase	[4]	psec	±70	+70	-70	[a]	PLL Phase Error Skew
		psec	±50	+50	-50	[a]	Primary Clock Skew
tpkg	[5]	psec	±40	+40	-40	[a]	LatticeSC Package Skew
tboard	[6]	psec	±80	+80	-80	[b]	Board Trace Skew (K/K# vs. D)
tsd	[7]	psec	+450	+450		[c]	QDRII/II+ SRAM Input Setup
thd	[7]	psec	-450		-450	[c]	QDRII/II+ SRAM Input Hold
Total				+865	-865		Sum of All Components

Note: "Eye" size = (2500 ps) - (865 ps) - (865 ps) = 770 ps

The second group of signals is depicted in Figure 6, and the timing budget is calculated in Table 4 (QDRII/II+ SRAM DLL enabled). There are nine components to the timing budget: [1] clock period, [2] QDRII/II+ SRAM duty cycle distortion (DCD), [3] QDRII/II+ SRAM Clock Output Skew, [4] package skew, [5] board trace skew, [6] edge clock skew, [7] LatticeSC receive register input setup/hold, [8] the SRAM output skew, and [9] the quantization error of the DLL. Item [9] represents the smallest step of delay change in the DLL's internal programmable delay.

Parameters marked [a] are LatticeSC device parameters and are valid over all the device's specified operating conditions and speed grades, [b] is a characteristic of the PC board, and [c] are QDRII/II+ SRAM device parameters. Values given for [b] and [c] are typical and must be adjusted for the specific PCB and QDRII/II+ SRAM chosen.

Figure 6. Timing Path, DDR Read

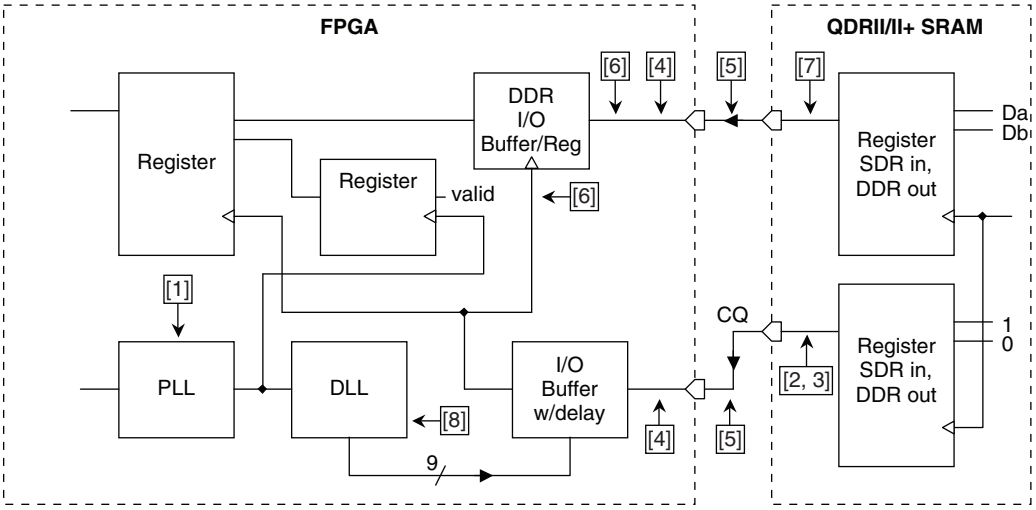


Table 4. Timing Budget - DDR Read Data Path (Preliminary)

Parameter		Units	Value	Long Path	Short Path		Description
fclk		MHz	200				Clock Frequency
tclk		psec	5000				Clock Period
tclk-ddr	[1]	psec	2500				DDR Clock Period (1/2 * tclk)
tdcd	[2]	psec	±125	+125	-125	[b]	Clock Duty Cycle Distortion of QDRII/II+ SRAM (45%-55%)
Tcqhqv	[3]	psec	+350	+350		[b]	QDRII/II+ SRAM CQ High to Q Valid
Tcqhqx	[3]	psec	-350		-350	[b]	QDRII/II+ SRAM CQ High to Q Invalid
tpkg	[4]	psec	±40	+40	-40	[a]	LatticeSC Package Skew
tboard	[5]	psec	±80	+80	-80	[b]	Board Trace Skew
	[6]	psec	+50	+50		[a]	Edge Clock Skew
tsd	[7]	psec	+363	+363		[a]	LatticeSC Input Setup
tsd	[7]	psec	+279		+279	[a]	LatticeSC Input Hold
tramout	[8]	psec	±40	+40	-40	[c]	QDRII/II+ SRAM Output Skew
Tdqe	[9]	psec	±70	+70	-70	[a]	DLL Delay Quantization Error
Total				+1118	-426		Sum of All Components

Note: "Eye" size = (2500 ps) - (1118 ps) - (426 ps) = 956 ps

Usage Overview

Module Manager and I/O Assistant

The LatticeSC family is supported by ispLEVER®, a suite of tools that facilitate the implementation of complex designs such as those that employ a QDRII/II+ SRAM Memory Interface.

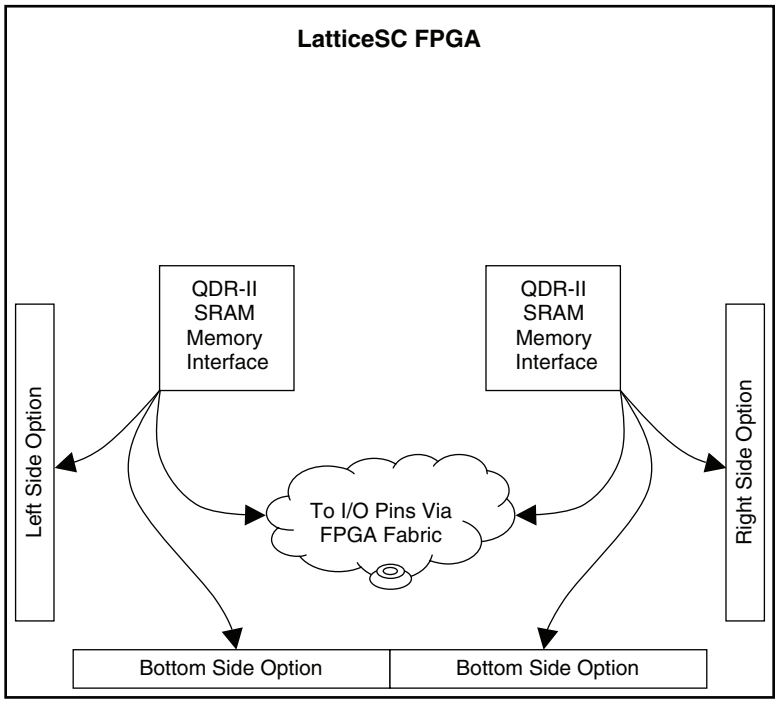
Module Manager is a graphically driven utility that accepts user specifications for pre-optimized logic blocks, and generates all the detailed instructions for synthesizing and simulating that logic. The LatticeSC QDR-SRAM Memory Interface is one of the logic units that Module Manager can generate.

I/O Assistant is another utility in ispLEVER, and it allows the designer to specify physical design restraints such as pin placement and I/O bank assignment, and to do so early in the design cycle, even before the detailed FPGA design is complete. This is necessary in complex designs where these allocations can have significant impact on the design flow.

Pinout Selection and Layout Constraint

In order to achieve maximum operating speed, the QDRII/II+ protocol requires carefully designed I/O timing on the interface between the LatticeSC device and the QDRII/II+ SRAM. For this reason, Lattice drop-in modules will provide pre-optimized pinout assignments, as well as assignments for critical internal components such as PLLs and DLLs (a QDRII/II+ Memory Interface design typically requires one PLL and one DLL). Where possible, optional alternatives are provided so that the Memory Interface can co-exist with various combinations of other IP. One option, the more usual case, places the Memory Interface's interface pins directly adjacent to the unit's internal logic on the left or right side of the chip (see Figure 7). However, in case those pins are unavailable, a second option places the interface pins on the bottom of the chip. There is the third option, of course, of not utilizing pre-assigned pinouts, but then the timing is not pre-optimized, and will change with each rerouting of the design.

Figure 7. QDRII/II+ SRAM Memory Interface Pinout Options



In order to achieve minimum skew between signals, all signals in a bus should be assigned pins within the same I/O bank. If this is not possible, use pins in two banks adjacent to the same corner of the FPGA. These two banks in the LatticeSC device have low skew when driven by a PLL or DLL in that corner.

It is a good design practice to explicitly assign the input clocks, but allow the associated data signals to be assigned by the placement tool. Clocks frequently drive directly into specific internal elements such as a PLL or DLL. These elements typically have a dedicated corresponding I/O pin that, when used, provides optimal performance. The I/O Assistant application facilitates the process of clock/data pin assignment. Using I/O Assistant, the designer can easily assign resources (pins, PLLs, DLLs, etc.) with whatever specificity is appropriate (pin, I/O bank, etc.), even before the design is complete.

All LatticeSC device pins that are tied to the QDRII/II+ SRAM's "Q" bus must be accessible by a single edge clock, which is driven by the conditioned "CQ" clock input.

The choice of pinout is often made or changed late in the design process, after design components have already been created in Module Manager. For this reason, the appropriate pinout is best selected by adding a precompiled set of preferences to the preference file, which allows the pinout to be defined or redefined at the latest possible point. When the Memory Interface is created by Module Manager, a "readme" file is also created. This file contains

all optional pinouts for the target device, and the designer can then copy and paste the appropriate pinout into the preference file.

Board Level Considerations

Trace Layout Guidelines

Good board layout techniques are important on any high-speed design, but QDRII/II+ SRAM designs have some specific requirements as well. Here are some general guidelines:

- All traces are 50 Ohm transmission lines.
- Trace lengths on buses must match to each other and their associated clocks to within 0.5 inch. There are two groups of buses, and trace lengths must be matched to others in that group. The first group includes the Address Bus "A", the Write Data Bus "D", as well as signals "W#" and "R#" (clock = "K/K#"); the second group includes only the Read Data Bus "Q" (clock = CQ).
- All traces are uncoupled; in particular, "K" and "KN" are not a differential pair.
- Power rails, such as "VTT", must be planes, not traces.
- Care must be taken to keep reference voltages, such as the QDRII/II+ SRAM's "VREF", noise-free. This involves robust, wide-bandwidth decoupling, and isolation of quiet, noise-sensitive signals and references from noise sources.
- The physical distance between the LatticeSC device and the QDRII/II+ SRAM needs to be minimized, since trace delays and skews will limit overall speed, as previously discussed.
- Simultaneous Switching Outputs (SSO) are a real concern in QDRII/II+ SRAM designs, since there are usually can be no design constraints on how many signals can switch simultaneously. Thus, for example, for the signals from the LatticeSC device to the QDRII/II+ SRAM, it is possible for all 56 bits in the Address and Write Data Buses to switch simultaneously in the same direction. For this reason, be sure to provide robust, high-current power/ground planes, with plenty of high-frequency bypassing. For more details, refer to the LatticeSC application notes in Section 5.

References

- For more information on the QDRII/II+ standard:
  - QDR Consortium: www.qdrsram.com
- For information on participating vendors:
  - Cypress Semiconductor Corp. www.cypress.com
  - Samsung Semiconductor www.samsung.com
  - NEC Corporation www.nec.com
  - IDT Incorporated www.idt.com
  - Renesas Technology Corp. www.renesas.com
- For more information on the LatticeSC family: www.latticesemi.com
  - LatticeSC Family Data Sheet
  - LatticeSC technical notes

Conclusion

This user's guide discusses an implementation of a QDRII/II+ Memory Interface, to assist designers in determining whether the QDRII/II+ SRAM is appropriate for their application, and if so, to provide understanding of its operation and details for design-in.

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
February 2006	01.0	Initial release.
November 2007	01.1	