SigmaQuad Common I/O Design Guide

Introduction

The 36Mb and 72Mb SigmaQuad product line has five different product families. These are SigmaQuad Common I/O (CIO) DDR Burst of 2 (B2), SigmaQuad Common I/O (CIO) DDR Burst of 4(B4), SigmaQuad Separate I/O (SIO) DDR Burst of 2 (B2), SigmaQuad Type II Burst of 2 (B2), and SigmaQuad Type II Burst of 4 (B4).

This application note will discuss the basic timing and functional differences between the SigmaQuad CIO DDR B2 and B4 product lines. The document will also look at some of the different applications with which SigmaQuad CIO DDR products can be interfaced. Lastly, the document will discuss some design considerations that need to be taken into account when interfacing with SigmaQuad CIO devices.

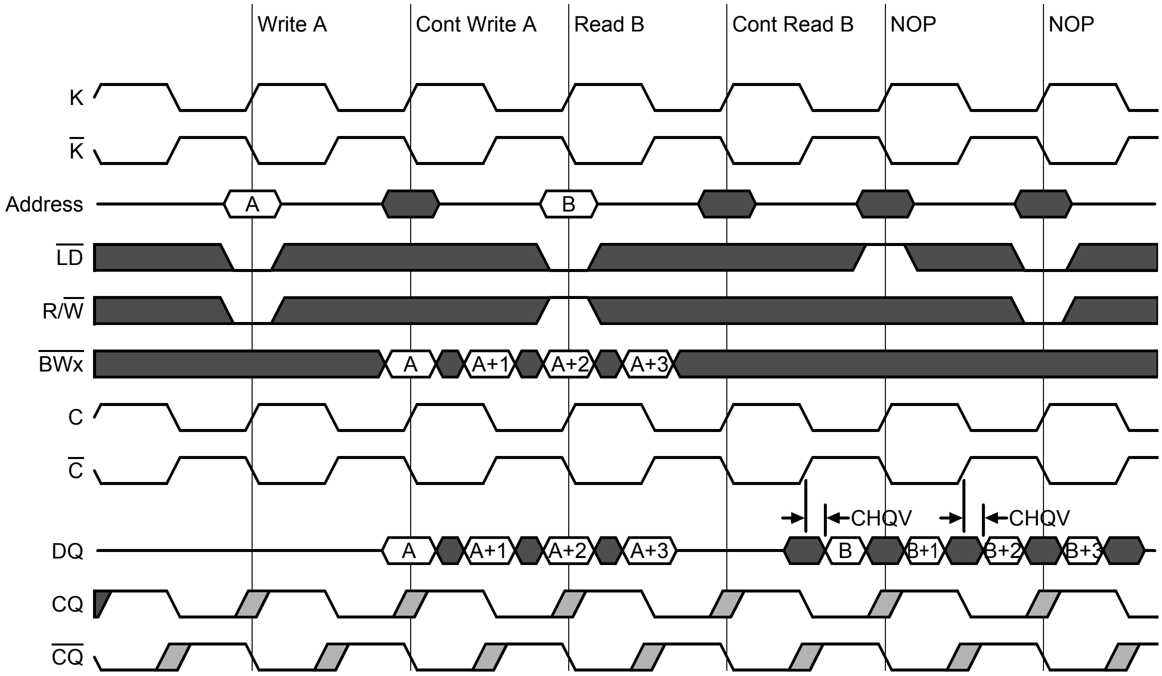
Common I/O devices

Common I/O product lines include the SigmaQuad CIO DDR B2 and SigmaQuad CIO DDR B4 devices. As the name implies, Common I/O devices share a common pin for both the input and output data ports. A benefit of Common I/O devices is that fewer traces are needed for data lines, but this also leads to one of the draw backs, wait states are needed to turn the data bus after a read cycle before the device can be written to.

Read and Write timing for Common I/O devices:

The figure below shows a basic write/read timing diagram for the SigmaQuad CIO DDR B4. For a read command to be loaded, $\overline{\text{LD}}$ needs to be driven low and R$\overline{\text{W}}$ needs to be driven high before the rising edge of K. The first burst read data will be driven out one cycle later on the rising edge of $\overline{\text{C}}$, if C and $\overline{\text{C}}$ are active, or $\overline{\text{K}}$, if C or $\overline{\text{C}}$ are tied high.   The second burst data will be driven out on the next rising edge or C, if C and $\overline{\text{C}}$ are active, or K, if C or $\overline{\text{C}}$ are tied high. The third burst read data will be driven on the next rising edge of $\overline{\text{C}}$, if C and $\overline{\text{C}}$ are active, or $\overline{\text{K}}$, if C or $\overline{\text{C}}$ are tied high.   The fourth and last burst data will be driven out on the next rising edge or C, if C and $\overline{\text{C}}$ are active, or K, if C or $\overline{\text{C}}$ are tied high.

Figure 1: Sigma Quad CIO DDR B4 Write and Read with C and $\overline{\text{C}}$ Active



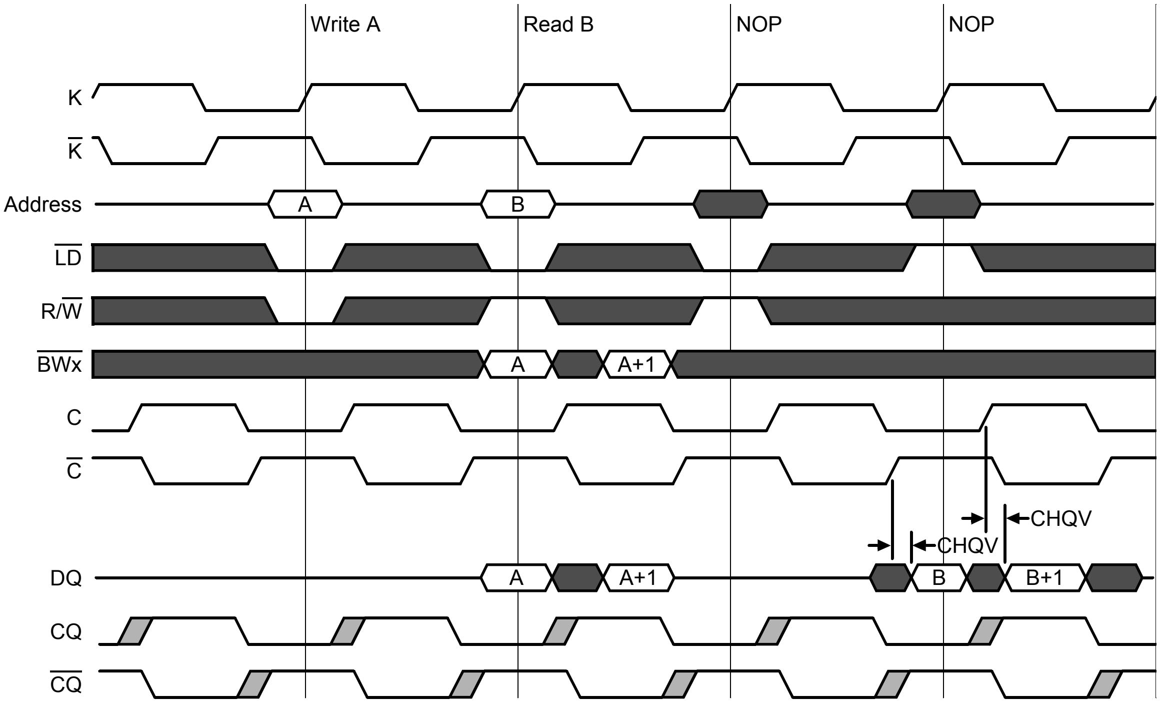
The write command is loaded when $\overline{\text{LD}}$ is driven low and R$\overline{\text{W}}$ is also driven low before the rising edge of K. The write data for the first burst is loaded on the next rising edge or K clock along with the Byte Write and the second write data and byte write are loaded on the following rising edge of $\overline{\text{K}}$ clock when the write address is loaded. The third write data and byte write are loaded on the next rising edge of K and the fourth and last write data and byte write are loaded on the next rising edge of $\overline{\text{K}}$. A deselect or NOP cycle is loaded when $\overline{\text{LD}}$ is driven high before the rising edge of K.

There are some limitations to the read and write cycles for the SigmaQuad CIO B4. The first limitation deals with the write command which is the device is not capable of loading a write command on every rising edge of K. The sequence is write/write cont. If a write/write sequence is attempted, the second write command will be ignored by the device. The second limitation, which deals with the read command is the device is not capable of loading a read command on every rising edge of K. The sequence for loading a read command is read/cont read. If a read/read sequence is attempted then the second read command will be ignored by the device. The transitions between read and write must be accompanied by a NOP cycle. The sequence is read/read cont/NOP/ write. Another consideration is that once the burst sequence, either a read or write, is started it cannot be stopped.

The following figure shows a basic read/write timing diagram for the SigmaQuad CIO DDR B2. For a read command to be loaded, $\overline{\text{LD}}$ needs to be driven low and R$\overline{\text{W}}$ needs to be driven high before the rising edge of K. The first burst read data will be driven out one cycle later on the rising edge of $\overline{\text{C}}$, if C and $\overline{\text{C}}$ are active, or $\overline{\text{K}}$, if C or $\overline{\text{C}}$ are tied high. The second burst data will be driven out on the next rising edge or C, if C and $\overline{\text{C}}$ are active, or K, if C or $\overline{\text{C}}$ are tied high.

The write command is loaded when $\overline{\text{LD}}$ is driven low and R$\overline{\text{W}}$ is also driven low before the rising edge of K. The write data for the first burst is loaded on the next rising edge or K clock along with the Byte Write and the second write data and byte write are loaded on the following rising edge of $\overline{\text{K}}$ clock when the write address is loaded. A deselect or NOP cycle is loaded when $\overline{\text{LD}}$ is driven high before the rising edge of K. There are some limitations to the read and write cycles for the SigmaQuad CIO B2. The first limitation deals with the read-to-write transitions. This transition requires a NOP cycle between the read and write commands. The sequence is read/NOP/write.

SigmaQuad CIO DDR B2 Write and Read with C and $\overline{\text{C}}$ Active



Additional features of the SigmaQuad Common I/O devices

DLL (Delay Lock Loop)

SigmaQuad Separate I/O devices incorporate a DLL to synchronize the output data to the input clocks. This increases the data valid window 30% vs. the same device with the DLL turned off. For further information on this topic, please reference the GSI application note **AN1012—36Mb SigmaQuad Type I vs. Type II Timing Comparison**. There are a few constraints to keep in mind in order for the DLL to function properly. The first is phase jitter. Phase jitter is the maximum allowed cycle-to-,cycle variation of the rising edges of the input clocks. This is defined in the datasheet as t_{KCVar}. For example at an operating frequency of 200 MHz, t_{CYC} of 5.0 ns, from any rising edge of clock to the next rising edge of clock needs to occur between 4.9 ns to 5.1 ns later. If the t_{KCVar} spec is violated too much, then the DLL has the potential to become unlocked.

The next constraint is DLL lock time. Once the input clocks to the SRAM have become stable, it takes 1024 clock cycles before the DLL is able to lock onto the operating frequency. It is ideal to have the voltage supplies DC-stable before supplying the inputs clocks to the SRAM. Also, one thing to keep in mind is that if the incoming clocks are not DC-stable, (i.e., too much cycle-to-cycle variation), the DLL may lock onto the wrong frequency. To avoid this potential undesired issue, use the $\overline{\text{Doff}}$ pin to turn off the DLL until the clocks have stabilized at the desired operating frequency. Then turn the DLL on and the DLL will lock onto the frequency within 1024 cycles.

When the DLL is enabled and the input cycle time is greater than the DLL maximum cycle time ($t_{KHKHmax}$), the datasheet specification for output clock to data valid timing is no longer guaranteed. For DLL input cycle times that are above the max $t_{KHKHmax}$ spec and less than 50 ns, then the clock to data valid spec will be somewhere between 2 ns and 16 ns and will vary cycle to cycle. Once the DLL input cycle time is greater than 50 ns, the clock to data valid specification is around 2 ns. Also, under these conditions the data driven out is still synced to the rising edge of $\overline{C}$ or $\overline{K}$, if C and $\overline{C}$ are tied high.

The SigmaQuad DLL is fine tuned for a particular frequency range across the offered speed grades. For example, for a 200 MHz SigmaQuad B2 device, the minimum t_{CYC} is 5.0 ns and the maximum t_{CYC} is 7.88 ns. This means that the frequency of the input clocks can only be between 200 MHz and 127 MHz, unlike other synchronous SRAM, which can be operated from 250 MHz down to 0 MHz. Outside of this range, the DLL could lose its frequency lock and clock to data out timings are no longer guaranteed.

Output Impedance Control

The ZQ pin on SigmaQuad devices is used to control the drive strength of the output drivers. The output impedance controller can be varied from 30Ω to 70Ω by connecting a resistor that is 5x the required value. To set the output drivers to 50Ω, a resistor value of 250Ω needs to be connected to the ZQ pin. The ZQ circuit provides a pull-up and pull-down tolerance of ±15%.

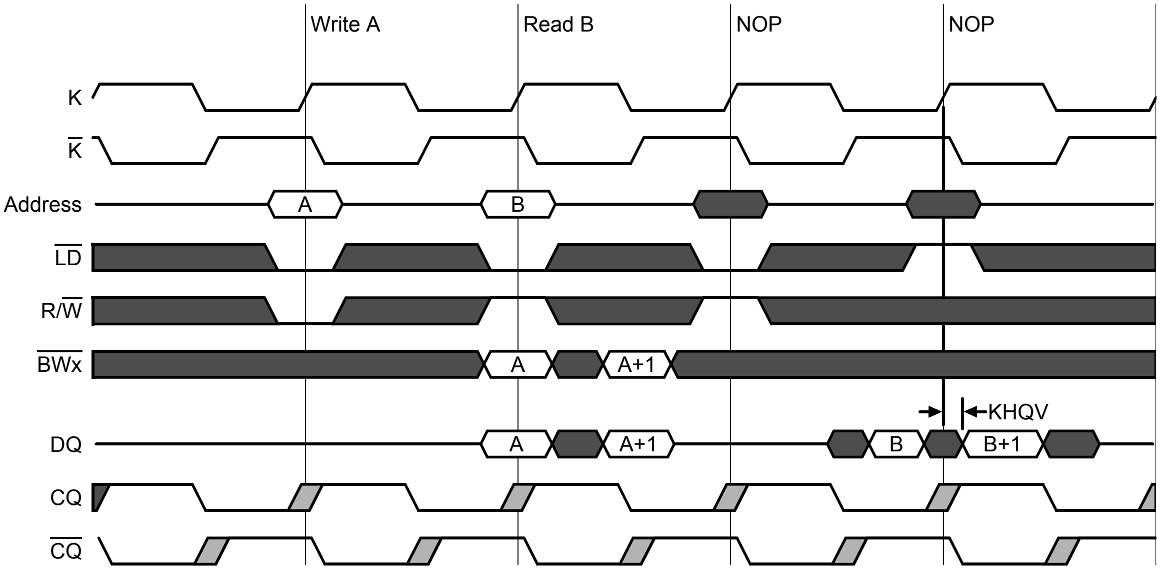
Typical System Implementation

During read operations there are several different methods that can be used to control the return data and predict when to latch-in this data. A few of these methods will be discussed below.

Using K and $\overline{K}$ only

SigmaQuad devices can be configured to operate referencing only K and $\overline{K}$ clocks for both input signals and output signals. This is accomplished by tying C and $\overline{C}$ clocks to V_{DD}. When C and $\overline{C}$ clocks are disabled, both CQ clocks and the I/Os are referenced to K and $\overline{K}$ as shown in the figure on the following page.

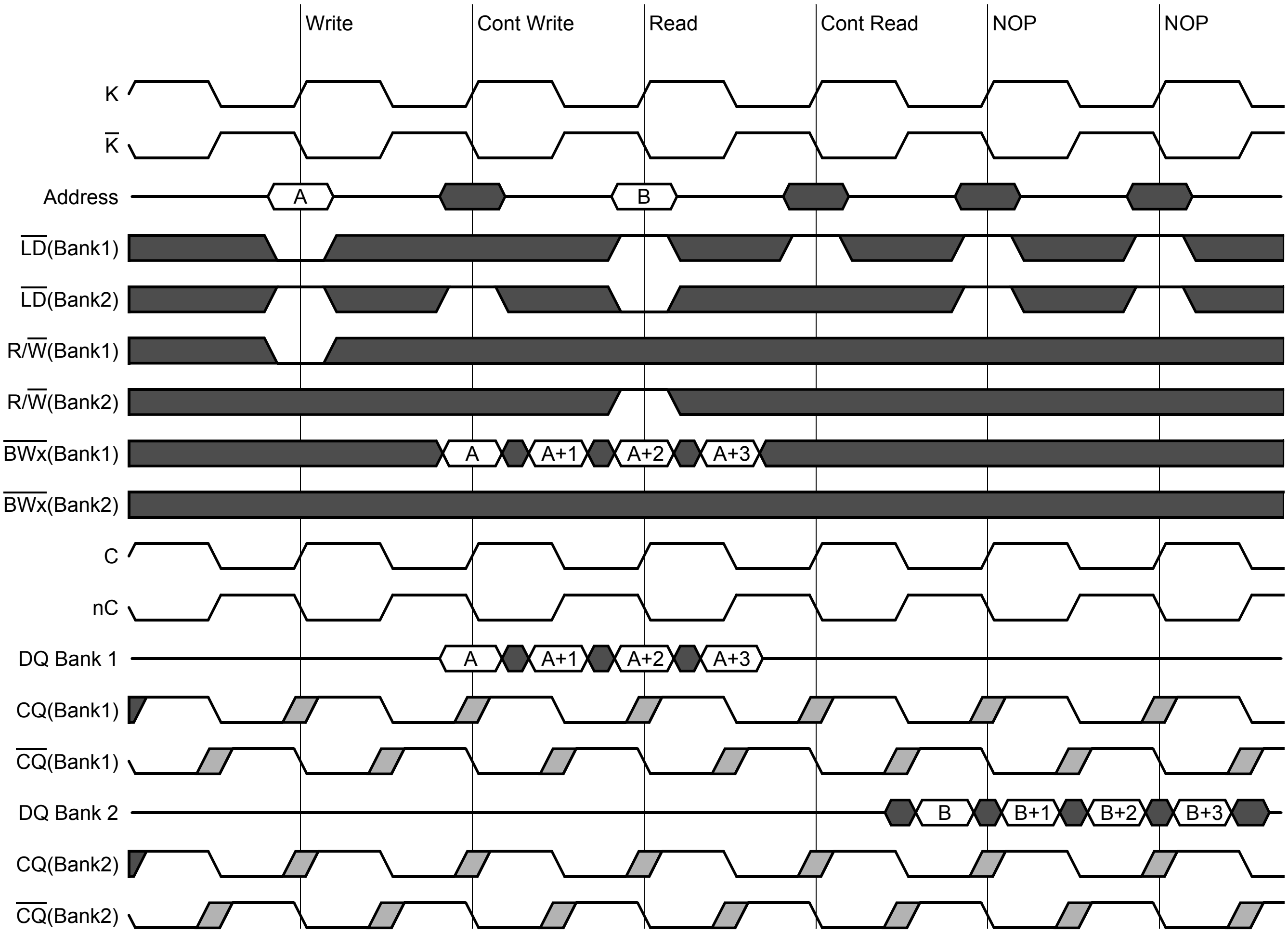
SigmaQuad CIO DDR B2 Write and Read with C and $\overline{\mathrm{C}}$ High



Using C and $\overline{\mathrm{C}}$ to capture data:

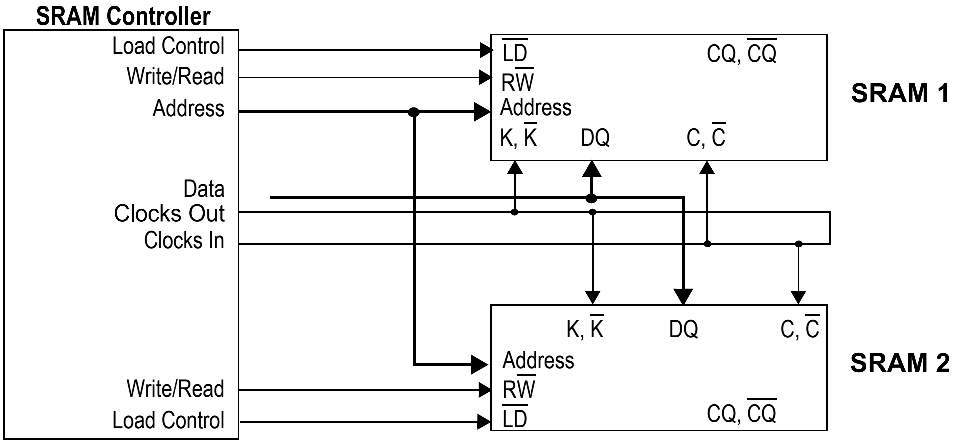
When the SigmaQuad devices are configured with active C and $\overline{\mathrm{C}}$ clocks, the input signals are still referenced to K and $\overline{\mathrm{K}}$, but the I/Os are now referenced to C and $\overline{\mathrm{C}}$. The figure on the following page is a timing diagram with sample depth expansion timing using C and $\overline{\mathrm{C}}$ to clock in return data.

Depth Expansion of SigmaQuad CIO Using C and $\overline{C}$ to Capture Return Data



This clocking setup is designed so that the return data from both of the SRAMs reaches the controller at the same time. This is accomplished by clocking data out from SRAM2 data first and then SRAM1, while assuming equal trace lengths for both the clock pairs and the data. The following figure shows a sample setup utilizing two SigmaQuad devices configured to implement C and $\overline{C}$ controlled reads.

Depth Expansion of SigmaQuad Devices Using C and $\overline{C}$ Clocks to Capture Return Data



There are a few considerations that the system engineer needs to keep in mind—one is the synchronizing of the return clocks and the master controller clock. Another criterion is that the round trip time between SRAM1 and SRAM 2 cannot violate the maximum skew allowed between K and C or $\overline{K}$ and $\overline{C}$. System engineers also need to consider the setup and hold conditions of the controller and add delay emulators by lengthening the return clock's trace with respect to the data trace.

Another consideration is the programmable output impedance pin (ZQ). As stated in the datasheet,s the ZQ pin must be connected to V_{SS} via an external resistor (RQ) to allow the SRAM to monitor and adjust its output driver impedance. The value of RQ must be 5X the value of the intended line impedance driven by the SRAM. For multiple drop SRAM data bus, the output impedance controller needs to be configured differently based on the individual SRAM's position on the bus. For example, if the SRAM is not at the end of the bus then the output drivers will see approximately half the load as the SRAM at the end of the bus. This means that these SRAMs will need to have a lower RQ resistance to allow them to better match the line impedance that they are seeing.

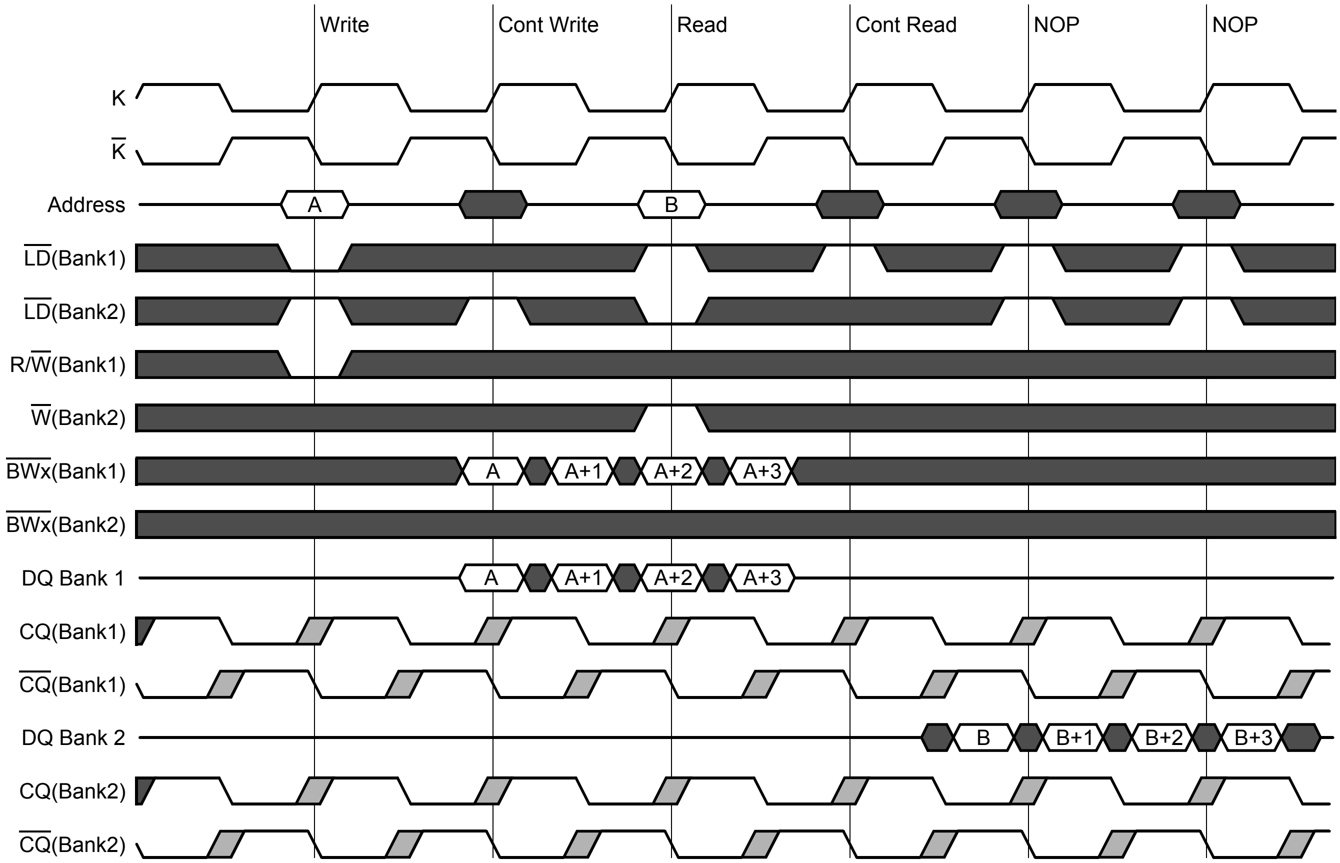
Echo Clocks to Capture Data

As trace lengths between the SigmaQuad devices and the controller get longer and longer, being able to predict and latch-in the return data becomes more difficult.

Echo clocks are generated by the SigmaQuad devices and are designated by CQ and $\overline{CQ}$. The Echo clocks are nominally 180° out of phase with each other and are free running. The echo clock frequency will match that of the input clocks K and $\overline{K}$, if C and $\overline{C}$ are tied high, or C and $\overline{C}$ if active. CQ's rising edge is referenced to the rising edge of C or K, if C and $\overline{C}$ are tied high, and $\overline{CQ}$'s rising edge is referenced to the rising edge of $\overline{C}$ or $\overline{K}$, if C and $\overline{C}$ are tied high.

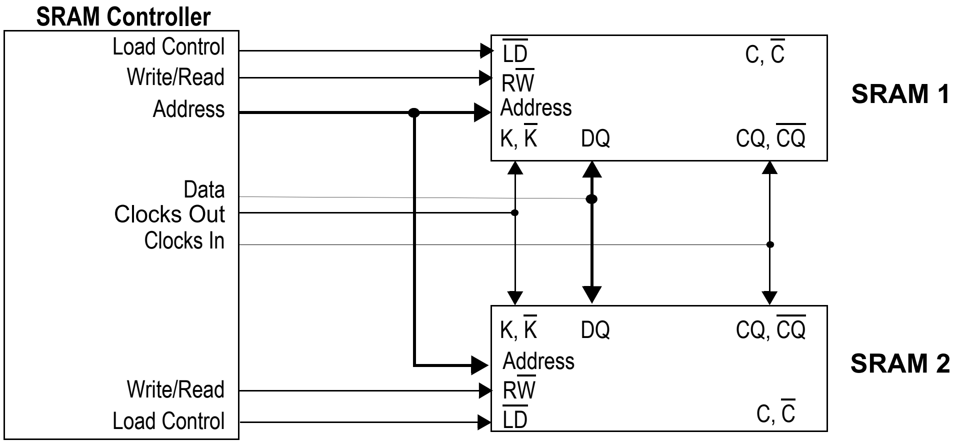
As mentioned before, echo clocks provide a way to predict when to latch-in return data. This is achieved to by providing a constant relationship between the echo clocks and the return data. The following figure is a timing analysis of depth expansion using echo clocks to capture return data when C and $\overline{C}$ are high. This figure shows a simple setup using two SigmaQuad devices with C and $\overline{C}$ tied high and echo clocks used to control when the controller captures data.

Depth Expansion Timing for SigmaQuad CIO Using Echo Clocks to Capture Data



The following figure shows read timing using echo clocks.

Depth Expansion of SigmaQuad Devices Using Echo Clocks as Data in Latch Clocks



There are a few considerations that the system engineer needs to keep in mind—one is the synchronizing of the return clocks and the master controller clock and the other is meeting the setup and hold requirements of the controller with the addition of delay emulators by lengthening the return clock's trace with respect to the data trace.

Summary

This application note has discussed many of the features that are present on SigmaQuad Common I/O devices. These features included depth expansion, different bursting lengths, and clocking schemes. The application note also discussed briefly the use of the DLL. For further information about the DLL features, please review **AN1011—18Mb SigmaQuad Type I vs. Type II Timing Comparison** and **AN1012—36Mb SigmaQuad Type I vs. Type II Timing Comparison**. For further assistance or if there are questions regarding the information discussed in this application note, please contact apps@gsitechnology.com.