Simultaneous Read-Write Operations in Dual-Port SRAM for Flash-Based cSoCs and FPGAs

Introduction

As design complexity grows, greater demands are placed upon embedded memory. Microsemi SmartFusion® customizable system-on-chip (cSoC) and Fusion, IGLOO®, and ProASIC®3 FPGAs provide the flexibility of true dual-port SRAM blocks. The dual-port configuration has two separate blocks (block A and block B) and corresponding clocks (CLKA and CLKB). This allows the user to perform both read and write operations on both blocks A and B. However, when performing simultaneous operations there may be data collisions and undesired data may be obtained at the output. There are four possible simultaneous memory access operations occurring in the same clock cycle at the same address in a dual-port SRAM: read-write, write-write, read-read, and write-read. While a read-read operation on the same address is not affected in any way, proper measures need to be taken for the other three operations to ensure that no data collision occurs.

The intent of this application note is to discuss these three operations in detail. This application note consists of three chapters.

"Chapter 1: IGLOO and ProASIC3L/EL Series FPGAs" – The first chapter describes the simultaneous read-write operations of dual-port SRAM in IGLOO series FPGAs and ProASIC3L/EL series devices.

"Chapter 2: ProASIC3 Series FPGAs " – The second chapter describes the simultaneous read-write operations of dual-port SRAM in ProASIC3 series FPGAs.

"Chapter 3: SmartFusion cSoCs and Fusion FPGAs" – The third chapter describes the simultaneous read-write operations of dual-port SRAM in SmartFusion and Fusion devices.

Dual Port SRAM Overview

The RAM4K9 macro is the dual-port configuration of the RAM block (Figure 1).

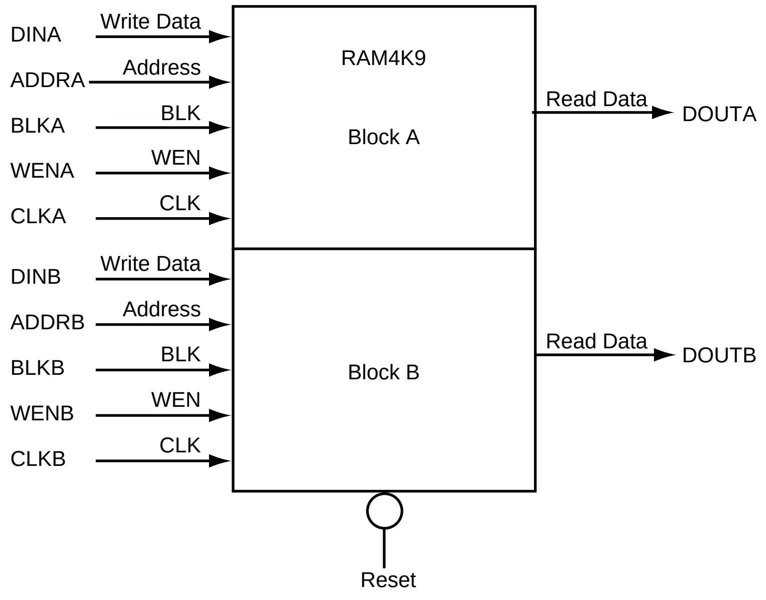

Figure 1 • **RAM4K9 Simplified Configuration**

The RAM4K9 nomenclature refers to both the deepest possible configuration and the widest possible configuration the dual-port RAM block can assume, and does not denote a possible memory aspect ratio. The RAM block can be configured to the following aspect ratios: 4,096×1, 2,048×2, 1,024×4, and 512×9. RAM4K9 is fully synchronous and has the following features:

- Two ports that allow fully independent reads and writes at different frequencies
- Selectable pipelined or nonpipelined read
- Active low block enables for each port
- Toggle control between read and write mode for each port
- Active low asynchronous reset
- Pass-through writes data or holds existing data on output. In pass-through mode, the data written to the write port will immediately appear on the read port.
- Designer software will automatically facilitate falling-edge clocks by bubble-pushing the inversion to previous stages.

The SRAM architecture, modes of operation, and signal descriptions for each family can be found in their respective user’s guides. Similarly, total embedded SRAM for each device can be found in their respective datasheets. Each chapter provides a table with list of supported devices and links to their user’s guides and datasheets.

Chapter 1: IGLOO and ProASIC3L/EL Series FPGAs

This chapter describes the simultaneous read-write operations of dual-port SRAM in IGLOO series FPGAs and ProASIC3/EL series devices. This chapter is applicable only to listed devices in Table 1.

Table 1 • **Supported Devices in Chapter 1**

Series	Datasheet	User's Guide
IGLOO	I*GLOO Low Power Flash FPGAs*	*IGLOO FPGA Fabric User's Guide*
	I*GLOOe Low Power Flash FPGAs*	*IGLOOe FPGA Fabric User's Guide*
	I*GLOO nano Low Power Flash FPGAs*	*IGLOO nano FPGA Fabric User's Guide*
	I*GLOO PLUS Low Power Flash FPGAs*	I*GLOO PLUS FPGA Fabric User's Guide*
ProASIC3	*ProASIC3L Low Power Flash FPGAs*	*ProASIC3L FPGA Fabric User's Guide*
	Radiation-Tolerant ProASIC3 Low Power Spaceflight Flash FPGAs	*RT ProASIC3EL FPGA Fabric User's Guide*
	Military ProASIC3/EL Low Power Flash FPGAs (A3PE600L and A3PE3000L)	*Military ProASIC3EL FPGA Fabric User's Guide* (A3PE600L and A3PE3000L)

Simultaneous Write-Write Operations

Simultaneous write-write is defined as a situation when the two clocks, CLKA and CLKB, turn on very close to one another to initiate a write operation on the same address of the RAM.

The behavior in a write-write situation depends on what occurs at the negative or closing edge of the clock. The word line—access to SRAM cell is enabled by word line—is asserted at the positive edge of the clock and deasserted at the negative edge. This means that once a write operation is commenced with port A, the content of the memory cannot be overwritten with port B until the clock signal (or word line) goes low. Therefore, a small delay is required between the negative edges of CLKA and CLKB in order to successfully perform a simultaneous write. The various behaviors are described in Table 2. Each case is discussed in detail in the following sections.

Table 2 • **List of Simultaneous Write-Write Scenarios**

Case	Description	Output When Data is Read After Write-Write Operations
1	Simultaneous clock on rising edge, CLKB falls before CLKA	Data from CLKA
2	Simultaneous clock on rising edge, CLKB falls after CLKA	Data from CLKB
3	CLKB rises after CLKA, CLKB falls before CLKA	Data from CLKA
4	CLKB rises after CLKA, CLKB falls after CLKA	Data from CLKB
5	CLKB rises when CLKA falls, CLKB falls after CLKA	Data from CLKB
6	CLKB rises after CLKA falls, CLKB falls after CLKA falls	Data from CLKB

Case 1: Simultaneous Clocking of Data from Both CLKA and CLKB, with CLKB Closing Before CLKA

In this case, data is being sent to be written in the same address from both CLKA and CLKB at the same time, with CLKB closing before CLKA. Since the word line is asserted at the positive edge of the clock and deasserted at the negative edge, regardless of whether the clock edge opens simultaneously or asynchronously, data from the later closing edge gets written into the memory block. Therefore, a small delay (t_{C2CWWL}) is required between the negative edges of CLKA and CLKB in order to successfully perform a simultaneous write. In this example, data from CLKA is written into the memory block. The reverse is also true: If CLKA closes before CLKB and both clocks opened simultaneously, then data that is written from CLKB is secured in the RAM address. This is valid for both pipelined and flow-through modes. Figure 2 illustrates simultaneous clocking of data from both CLKA and CLKB, with CLKB closing before CLKA.

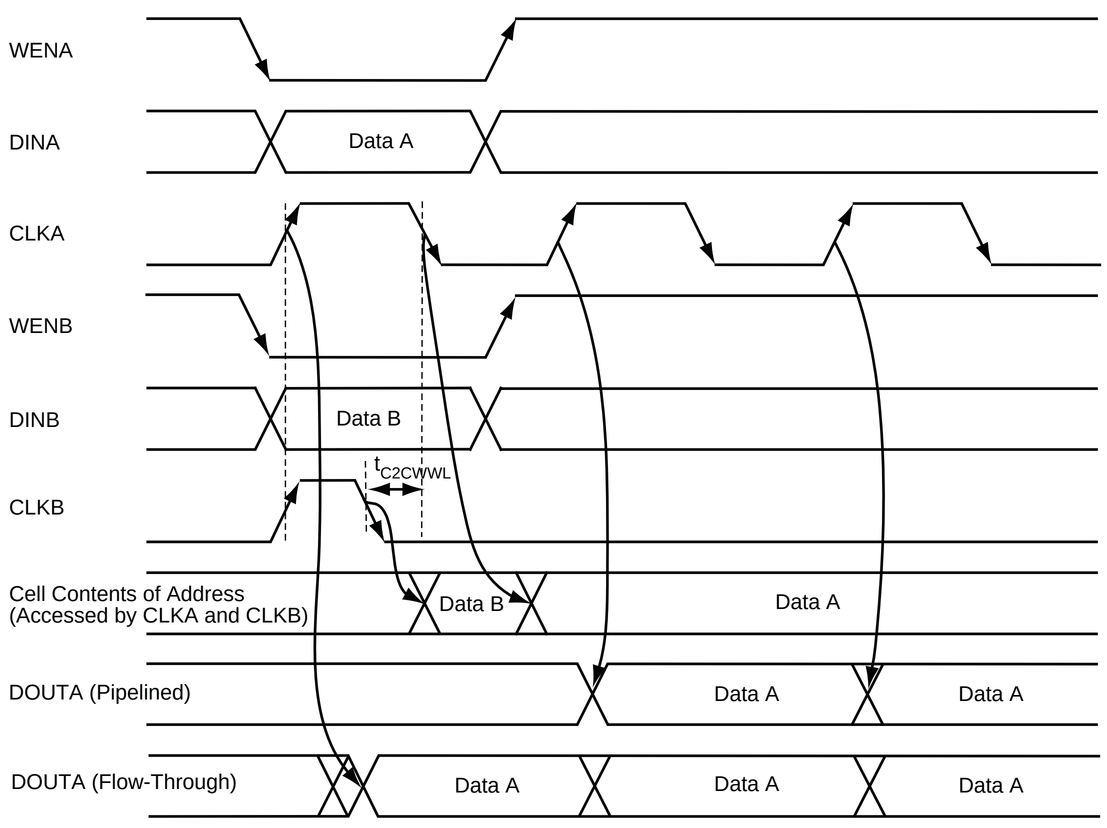

Figure 2 • **Simultaneous Clocking of Data from Both CLKA and CLKB, with CLKB Closing Before CLKA**

Case 2: Simultaneous Clocking of Data from Both CLKA and CLKB, with CLKA Closing Before CLKB

This is the opposite clocking scheme from case 1, with data being sent to be written in the same address from both CLKA and CLKB at the same time, and CLKA closing before CLKB. Since the word line is asserted at the positive edge of the clock and deasserted at the negative edge, regardless of whether the clock edge opens simultaneously or asynchronously, data from the later closing edge gets written into the memory block. Therefore, a small delay (t_{C2CWWL}) is required between the negative edges of CLKA and CLKB in order to successfully perform a simultaneous write. In this example, data from CLKB is written into the memory block. The reverse is case 1, described above. This is valid for both pipelined and flow-through modes. Figure 3 illustrates simultaneous clocking of data from both CLKA and CLKB, with CLKA closing before CLKB..

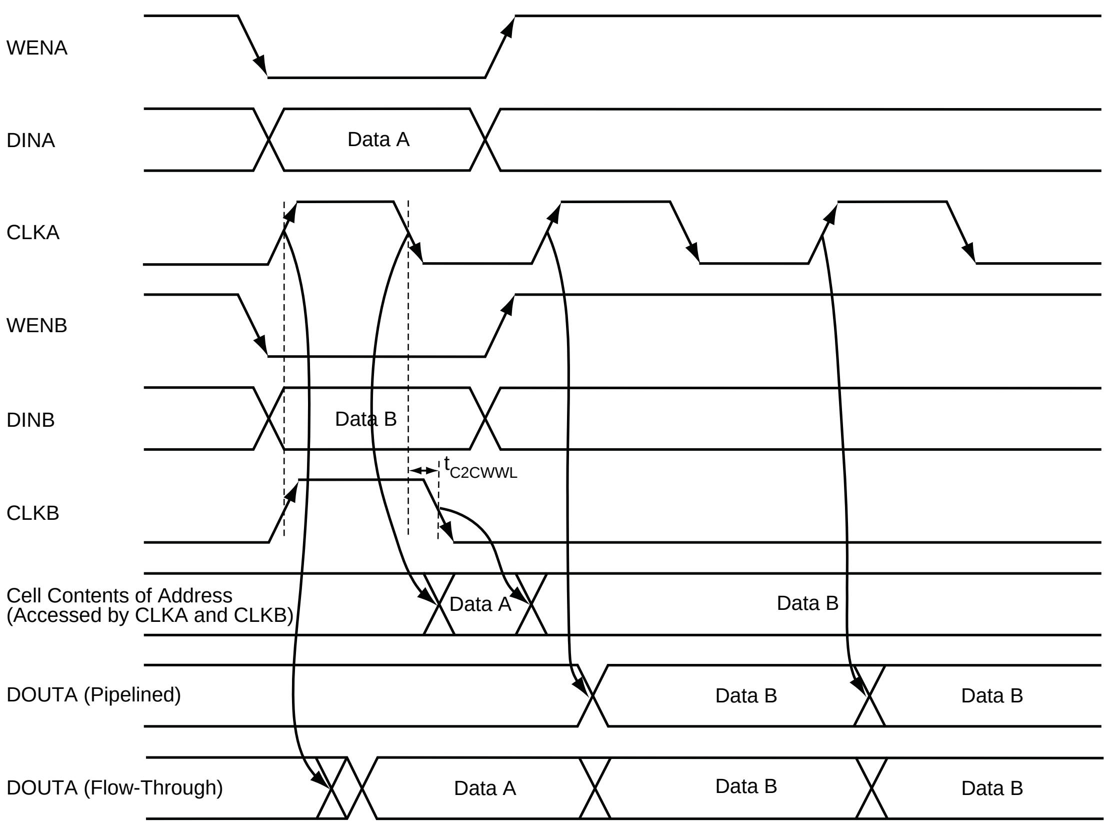

Figure 3 • **Simultaneous Clocking of Data from both CLKA and CLKB, with CLKA Closing Before CLKB**

Case 3: CLKB Opening After CLKA, CLKB Closing Before CLKA

Similar to case 1, data is being sent to be written in the same address from CLKA and then CLKB, with CLKB closing before CLKA. Since the word line is asserted at the positive edge of the clock and deasserted at the negative edge, regardless of whether the clock edge opens simultaneously or asynchronously, data from the later closing edge gets written into the memory block. Therefore, a small delay (t_{C2CWWL}) is required between the negative edges of CLKA and CLKB in order to successfully perform a simultaneous write. In this example, data from CLKA is written into the memory block. The reverse is also true: If CLKA closes before CLKB, regardless of opening edge, then in this case, data that is written from CLKB is secured in the RAM address. This is valid for both pipelined and flow-through modes. Figure 4 illustrates CLKB opens after CLKA and CLKB closes before CLKA.

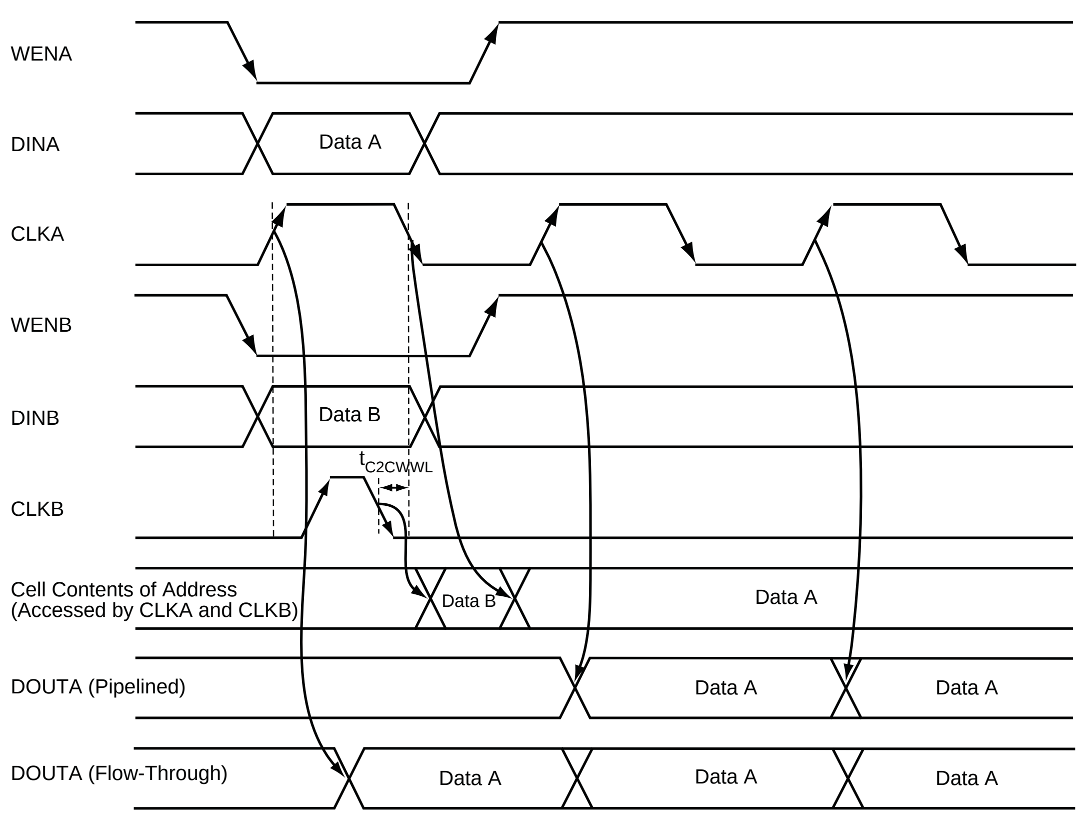

Figure 4 • **CLKB Opens After CLKA, CLKB Closes Before CLKA**

Case 4: CLKB Opening After CLKA, CLKA Closing Before CLKB

Since the word line is asserted at the positive edge of the clock and deasserted at the negative edge, regardless of whether the clock edge opens simultaneously or asynchronously, data from the later closing edge gets written into the memory block. If CLKA closes before CLKB, with a small delay (t_{C2CWWL}) between these two clocks, the last data that is written to the address will be kept. In this case, data that is written from CLKB is secured in the RAM address. It is not dependent on the opening edge of the clocks. As such, whether CLKB opens before CLKA or CLKA opens before CLKB is irrelevant. The reverse is case 3, described above. This is valid for both pipelined and flow-through modes. Figure 5 illustrates CLKB opens after CLKA and CLKB closes after CLKA.

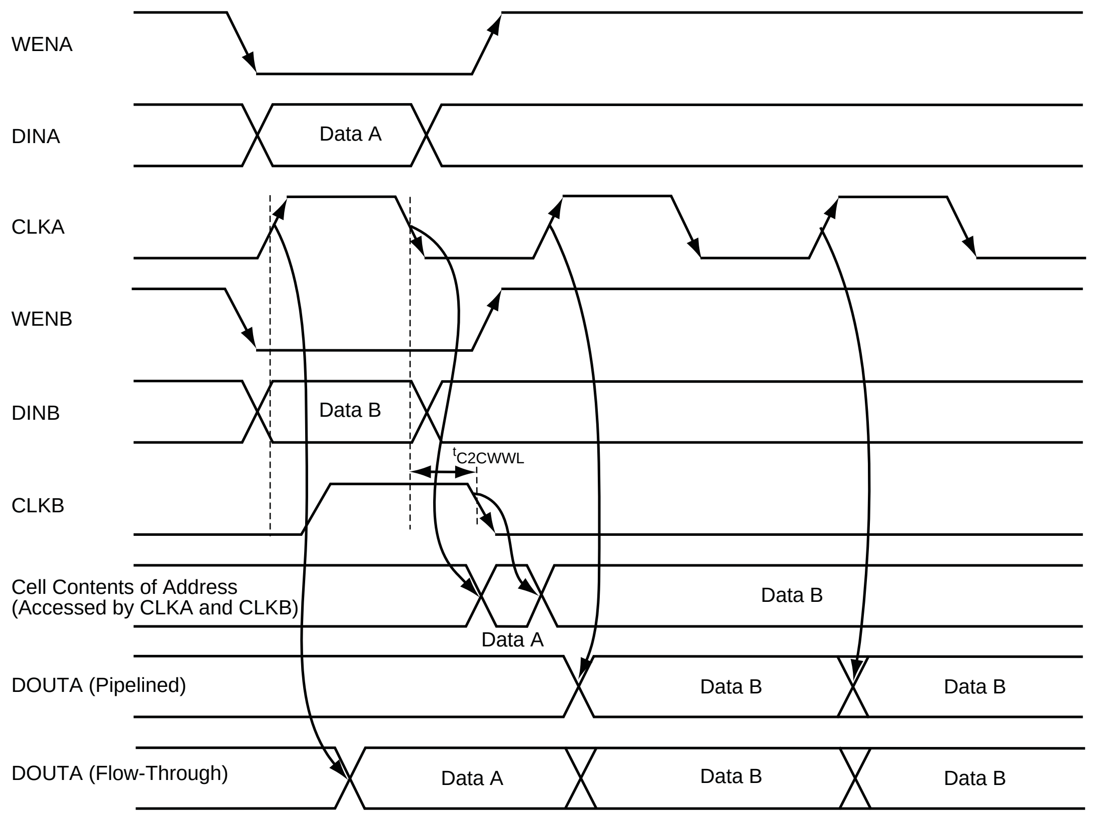

Figure 5 • CLKB Opens After CLKA, CLKB Closes After CLKA

Case 5: CLKB Opening When CLKA Closes, CLKB Closes After CLKA Closes

There is actually no collision in this case; it is normal operation. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data written by CLKB is instantiated, no collisions occur. By intuition, the last data written to the address is stored. In this example, the data written from CLKB is stored in the memory. As a result, in the next cycle where the data is read, data B is read from the memory. The reverse is also true: if data written by CLKA precedes data written by CLKB, data that is written from CLKA is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 6 illustrates CLKB opens when CLKA closes and CLKB closes after CLKA closes.

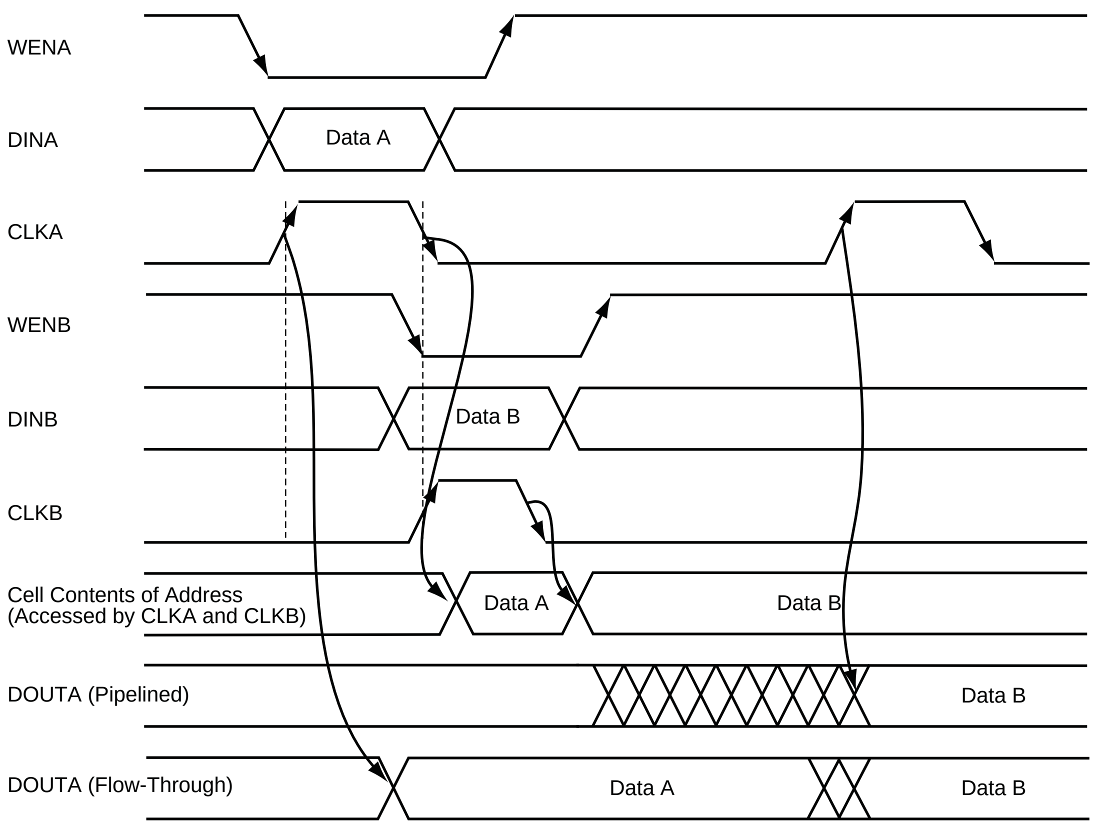

Figure 6 • **CLKB Opens When CLKA Closes; CLKB Closes After CLKA Closes**

Case 6: CLKB Opening After CLKA Closes, CLKB Closes After CLKA Closes

Similar to case 5, there is no collision in this scenario, and it is considered a normal operation case. Since data written by CLKA is completed before data written by CLKB is instantiated, no collisions occur. By intuition, the last data written to the address is stored. In this example, the data written from CLKB is stored in the memory. As a result, in the next cycle where the data is read, data B is read from the memory. The reverse is also true: if data written by CLKA precedes data written by CLKB, data that is written from CLKA is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 7 illustrates CLKB opens after CLKA closes and CLKB closes after CLKA closes.

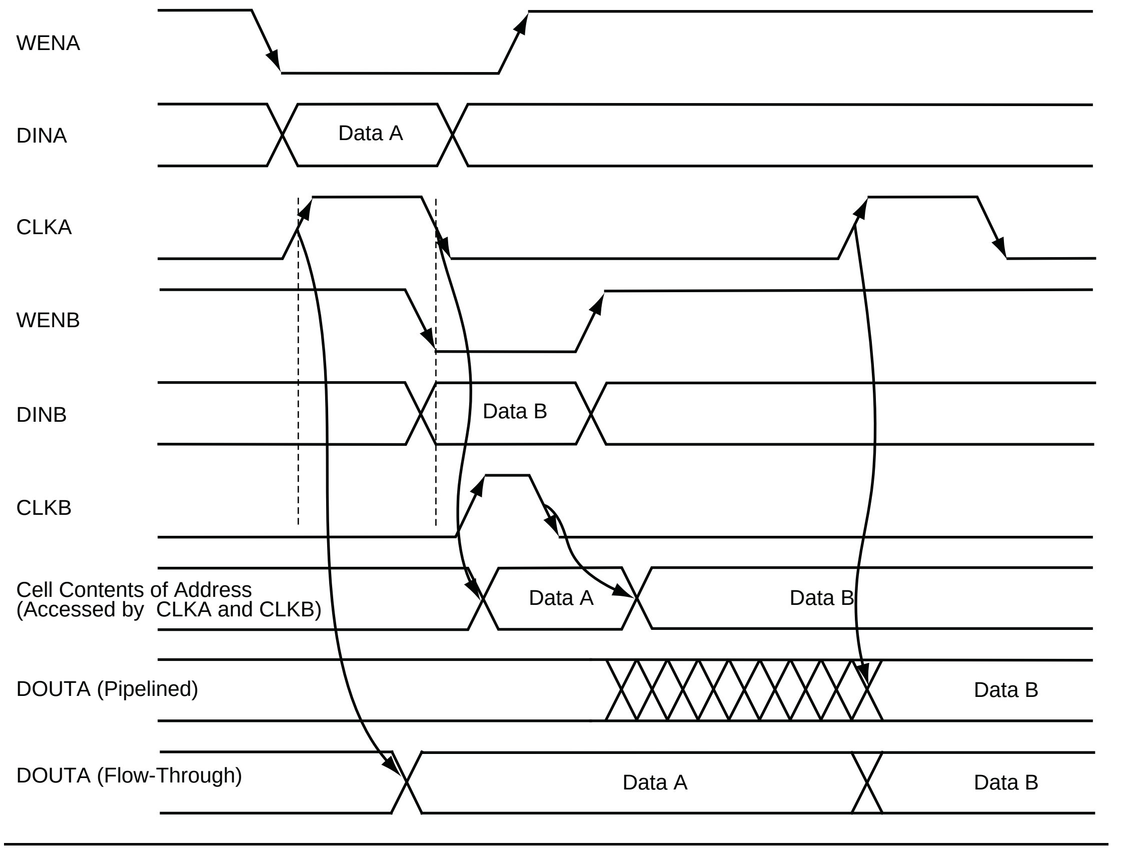

Figure 7 • **Opens When CLKA Closes, CLKB Closes After CLKA Closes**

Simultaneous Read-Write and Write-Read Operations

Simultaneous read-write is defined as the situation when the two clocks, CLKA and CLKB, turn on very close to one another to initiate one write operation and one read operation on the same address of the RAM.

The behavior in a read-write or a write-read situation depends on what occurs at the positive or the opening edge of the clock. Since it takes time for the write operation to occur and also the read operation, a small delay is required between the positive edges of CLKA and CLKB in order to successfully perform a simultaneous write-read or read-write operations. All events except for events having both edges opening simultaneously will have a stable, known output. A write operation is performed for CLKA while a read operation is performed on CLKB. The various behaviors are described in Table 3. Each case will be discussed in detail in the following sections.

Table 3 • **List of Simultaneous Write-Read, Read-Write Scenarios**

Case	Description	Output When Data is Read
1	Simultaneous clock on falling edge, CLKB rises after CLKA	Data from CLKA is read
2	CLKB rises after CLKA, CLKB falls before CLKA	Data from CLKA is read
3	CLKB rises after CLKA, CLKB falls after CLKA	Data from CLKA is read
4	CLKB rises before CLKA, CLKB falls before CLKA	Old data from address is read if delay between CLKA and CLKB is met, else it would be unknown
5	Simultaneous clock on falling edge, CLKB rises before CLKA	Old data from address is read if delay between CLKA and CLKB is met; otherwise it would be unknown
6	CLKB rises when CLKA falls, CLKB falls after CLKA	Data from CLKA is read
7	CLKB rises after CLKA falls, CLKB falls after CLKA	Data from CLKA is read

Case 1: CLKB Opening After CLKA, Both Clock Closing Simultaneously

Case 1 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, data that is written from CLKA is secured in the RAM address and can be read from CLKB. It is independent of the closing edges.

The reverse is also true: If CLKA opens after CLKB and both clocks close simultaneously (with data written from CLKB and read from CLKA), then data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 8 illustrates CLKB opening after CLKA and both clocks closing simultaneously. Data is written from CLKA and read in CLKB.

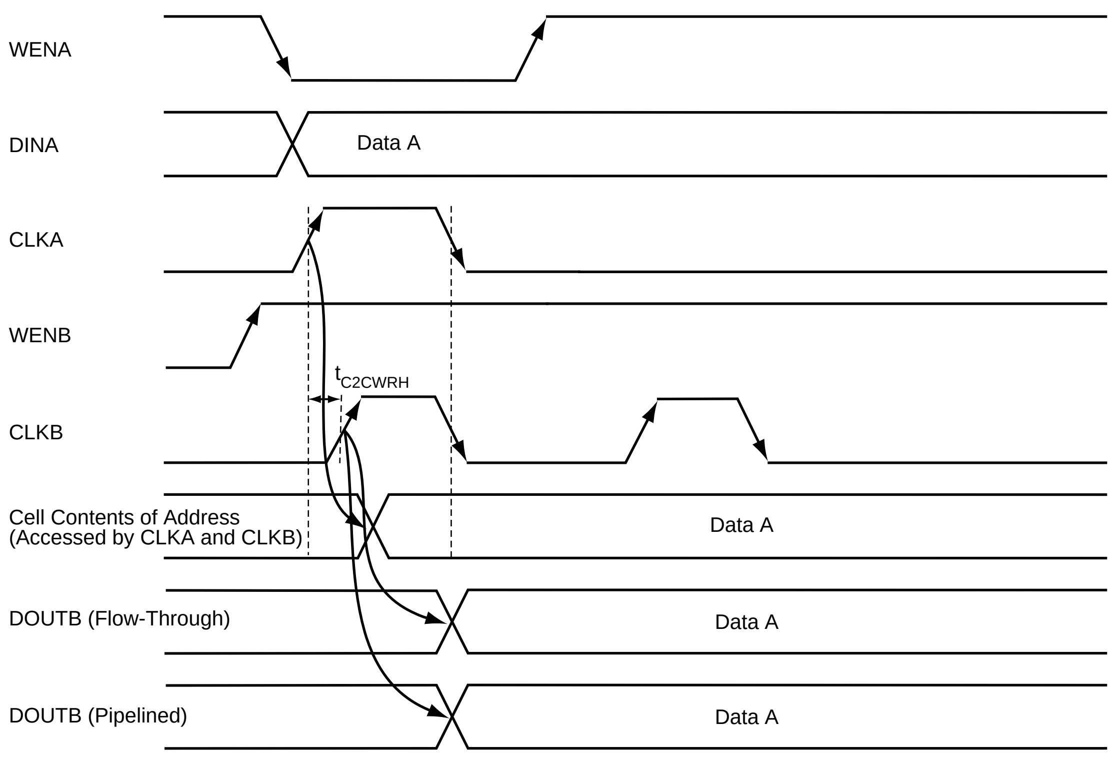

Figure 8 • **CLKB Opens After CLKA, Both Clocks Closing Simultaneously. Data is Written from CLKA and Read in CLKB.**

Case 2: CLKB Opening After CLKA, CLKB Closes Before CLKA

Similar to case 1, case 2 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, data that is written from CLKA is secured in the RAM address and can be read from CLKB. The ability of the data to be read accurately is independent of the closing edges, regardless of whether the clock edges close simultaneously or asynchronously.

The reverse is also true: If CLKA opens after CLKB and CLKA closes before CLKB (with data written from CLKB and read from CLKA), then data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 9 illustrates CLKB opens after CLKA and CLKB closes before CLKA. Data is written from CLKA and read in CLKB.

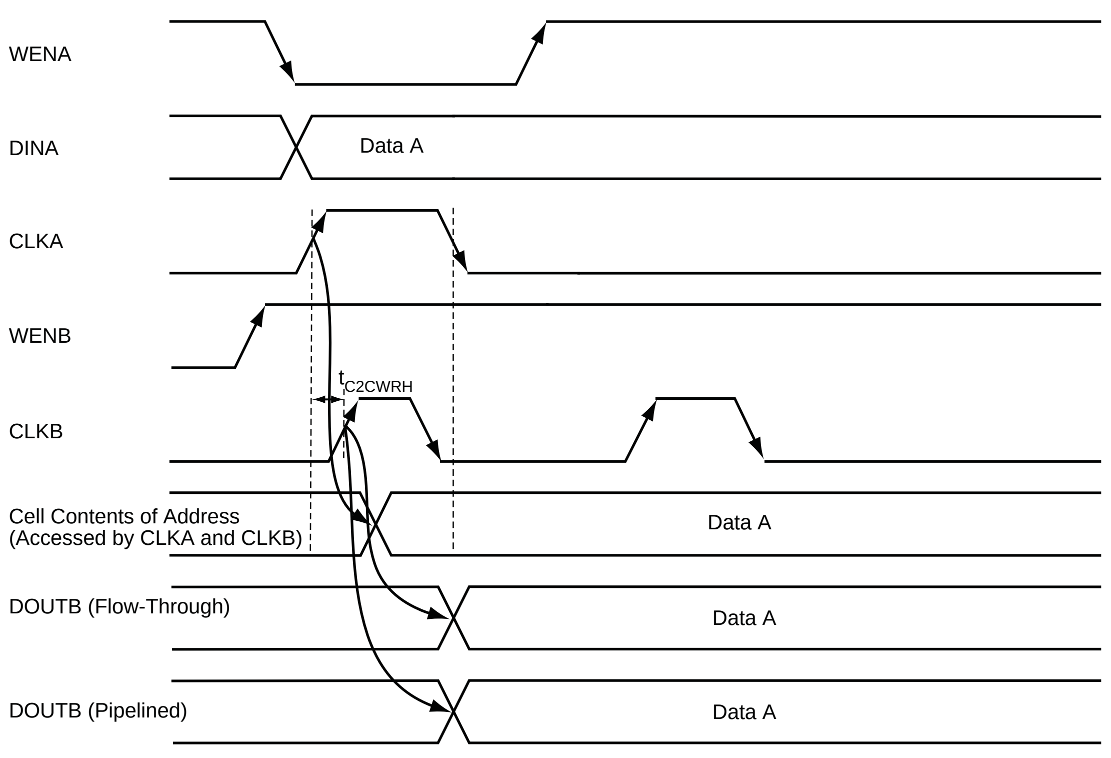

Figure 9 • CLKB Opens After CLKA, CLKB Closes Before CLKA. Data is Written from CLKA and Read in CLKB.

Case 3: CLKB Opening After CLKA, CLKB Closes After CLKA

Similar to case 1, case 3 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock, if data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, in this case, data that is written from CLKA is secured in the RAM address and can be read from CLKB. The ability of the data to be read accurately is independent of the closing edges, regardless of whether the clocks edge closes simultaneously or asynchronously.

The reverse is also true: If CLKA opens after CLKB and CLKA closes before CLKB (with data written from CLKB and read from CLKA), then data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 10 illustrates CLKB opens after CLKA and CLKB closes after CLKA. Data is written from CLKA and read in CLKB.

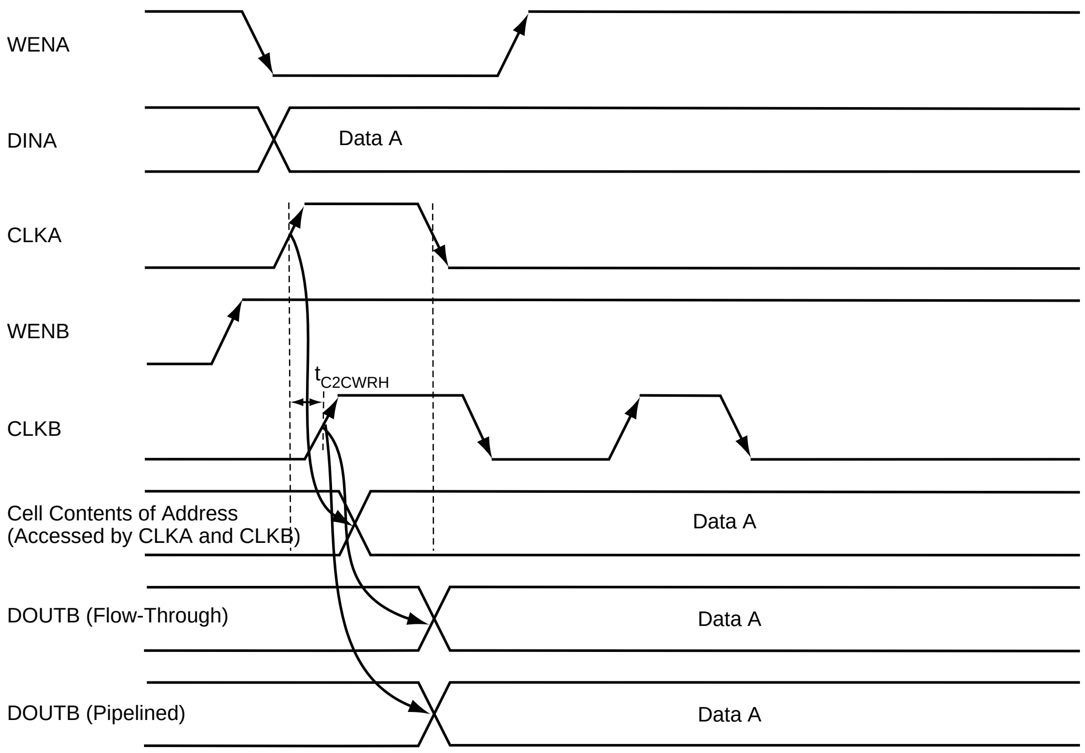

***Figure 10* • CLKB Opens After CLKA, CLKB Closes after CLKA. Data is Written from CLKA and Read in CLKB.**

Case 4: CLKB Opening Before CLKA Opens, CLKB Closes Before CLKA Closes

In this case, it is a read-write operation rather than a write-read operation. Since write and read operations cannot occur instantaneously, a small delay occurs between the read and write operations (t_{C2CRWH}). If this delay is met, the old data from the address is read by CLKB before new data overwrites the old data by CLKA. The reverse is also true: if data being read by CLKA precedes data written by CLKB, the old data in the address is read before new data from CLKB is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 11 illustrates CLKB opens before CLKA opens and CLKB before CLKA closes. Data is written from CLKA and read in CLKB.

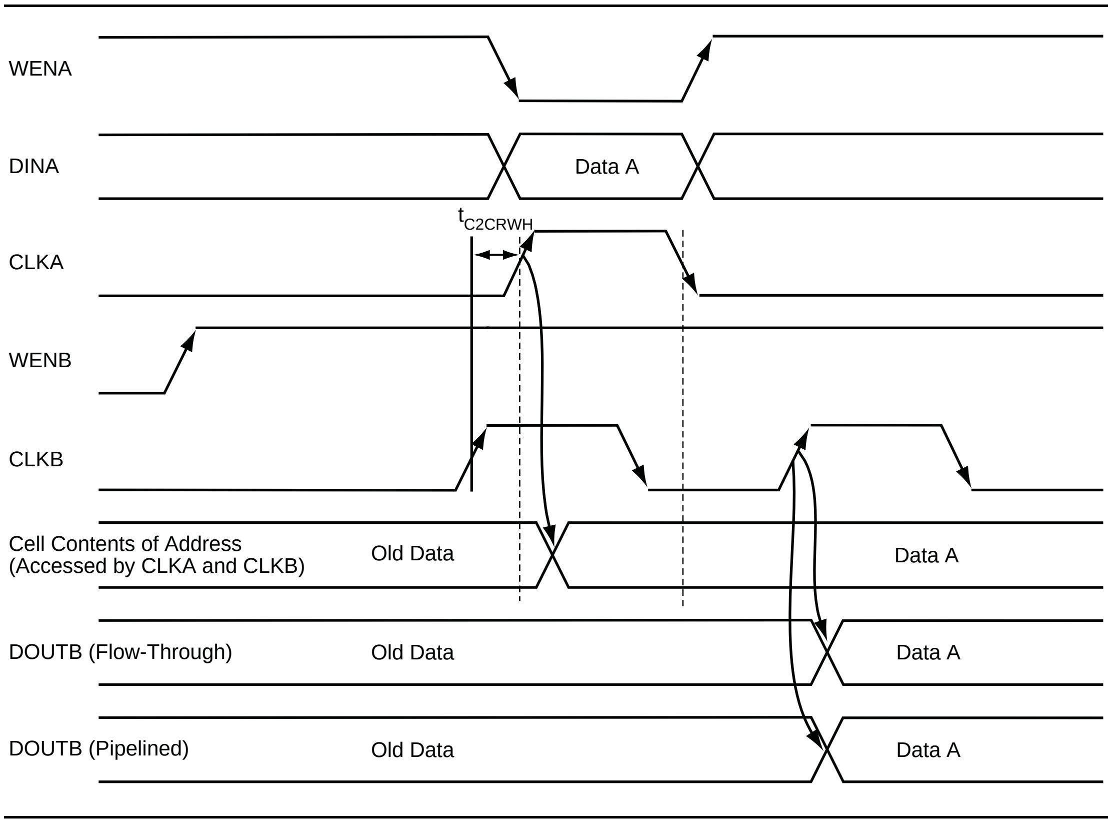

Figure 11 • **CLKB Opens Before CLKA Opens, CLKB Closes Before CLKA Closes. Data is Written from CLKA and Read in CLKB.**

Case 5: CLKB Opening Before CLKA Opens, CLKB Closes Simultaneously with CLKA

Similar to case 4, this is a read-write operation rather than a write-read operation. Since write and read operations cannot occur instantaneously, a small delay occurs between the read and write operations (t_{C2CRWH}). If this delay is met, the old data from the address is read by CLKB before new data overwrites the old data by CLKA. What is read on the read clock (CLKB) is independent of the closing edge. Therefore, whether CLKB closes before, after, or simultaneously with the CLKA edge is irrelevant. The reverse is also true: if data being read by CLKA precedes data written by CLKB, the old data in the address is read before new data from CLKB is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 12 illustrates CLKB opens before CLKA opens and CLKB closes simultaneously with CLKA. Data is written from CLKA and read in CLKB.

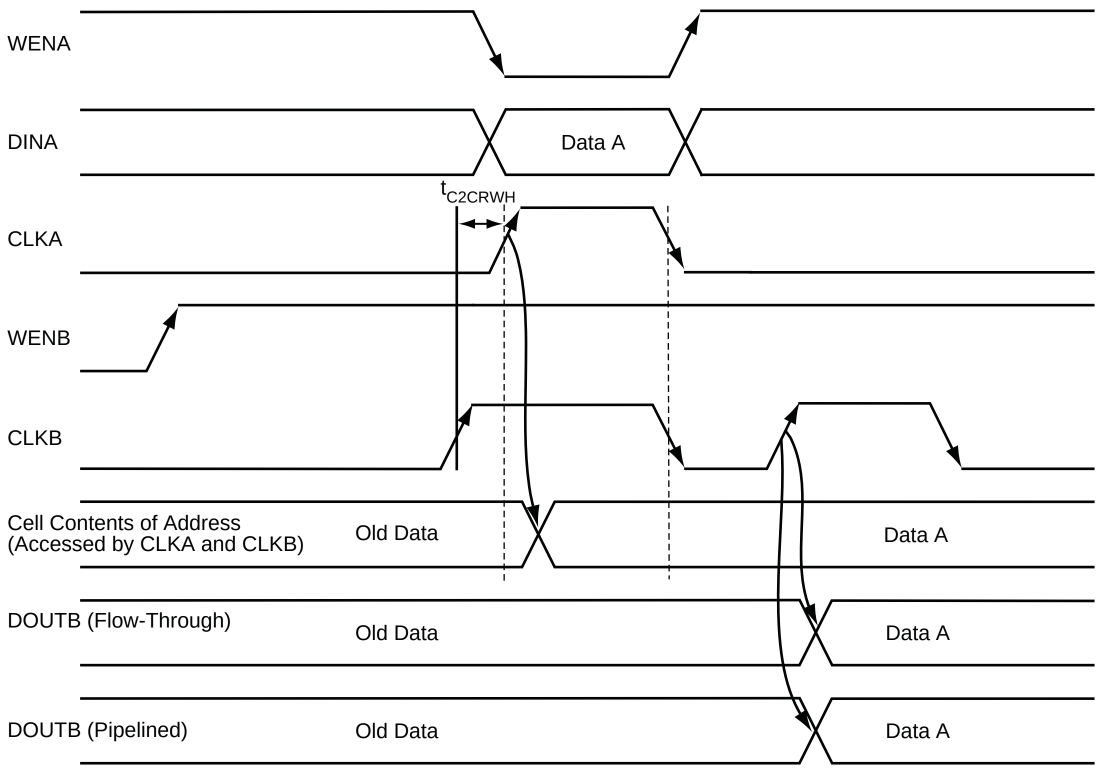

Figure 12 • **CLKB Opens Before CLKA Opens, CLKB Closes Simultaneously with CLKA. Data is Written from CLKA and Read in CLKB.**

Case 6: CLKB Opening when CLKA Closes, CLKB Closes After CLKA

There is no collision in case 6, and it is considered a normal operation case. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data is read by CLKB, no collisions occur. By intuition, the last data written to the address is stored and read. In this example, the data written from CLKA is stored in the memory. As a result, when CLKB opens, the data written by CLKA in the same memory location is read. The reverse is also true: If data written by CLKB precedes data read by CLKA, data that is written from CLKB is secured into the memory and is read by CLKA. This is valid for both pipelined and flow-through modes. Figure 13 illustrates CLKB opens when CLKA closes, CLKB closes after CLKA. Data is written from CLKA and read in CLKB, in pipelined mode.

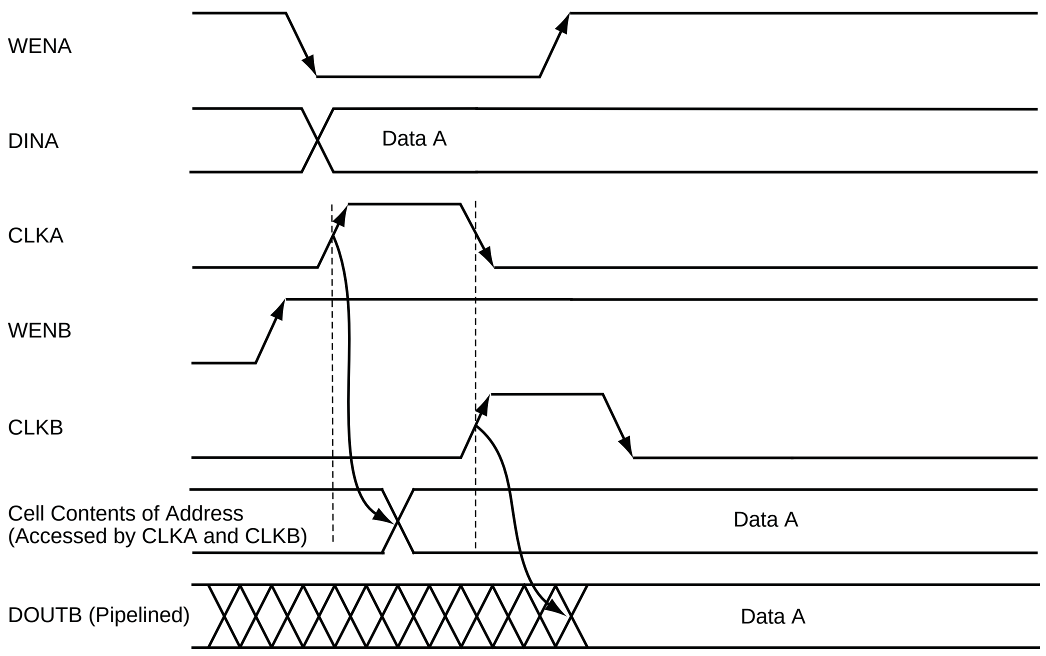

***Figure 13* • CLKB Opens When CLKA Closes; CLKB Closes After CLKA. Data Is Written from CLKA and Read in CLKB, Pipelined Mode.**

Case 7: CLKB Opening After CLKA Closes, CLKB Closes After CLKA

Similar to case 6, this is normal operation and there is no collision. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data is read by CLKB, no collisions occur. By intuition, the last data written to the address is stored and read. In this example, the data written from CLKA is stored in the memory. As a result, when CLKB opens, the data written by CLKA in the same memory location is read. The reverse is also true: if data written by CLKB precedes data read by CLKA, data that is written from CLKB is secured into the memory and is read by CLKA. This is valid for both pipelined and flow-through modes. Figure 14 illustrates CLKB opens after CLKA closes and CLKB closes after CLKA. Data is written from CLKA and read in CLKB.

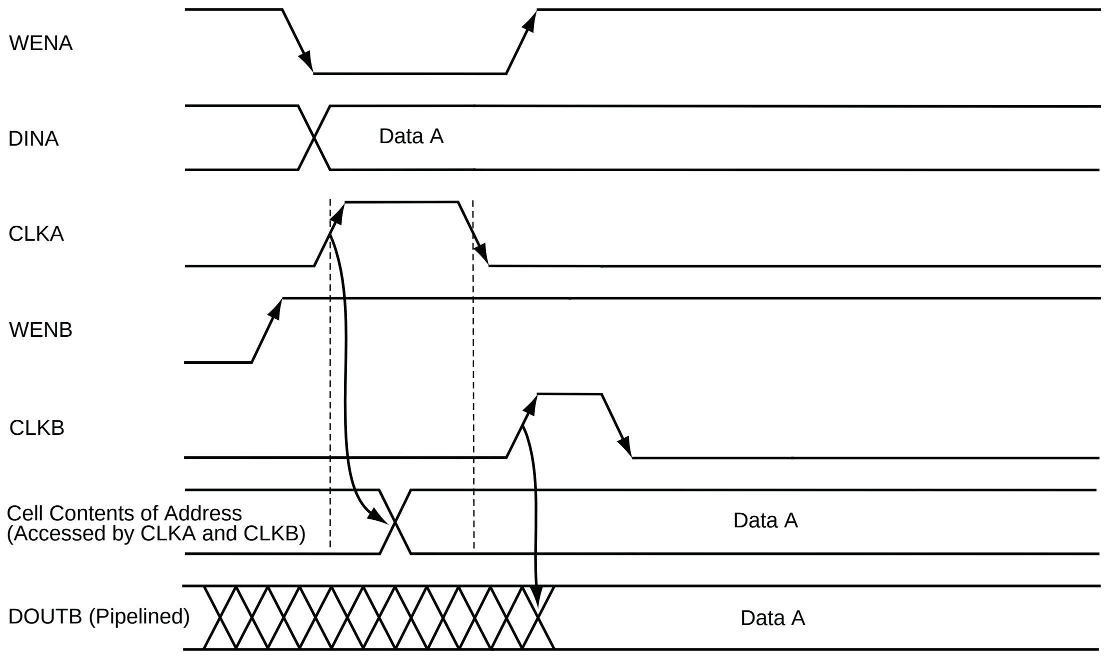

***Figure 14* • CLKB Opens After CLKA Closes, CLKB Closes After CLKA. Data Is Written from CLKA and Read in CLKB.**

The above descriptions of the cases assume that there are no violations in setup/hold times of the described signals (address setup/hold time and data setup/hold time). Two additional clock-to-clock constraints on the positive or opening edge and one clock-to-clock constraint on the negative or closing edge for write-write operation are required:

t_{C2CWWL} – Clock-to-clock constraint for a write-then-write operation

t_{C2CRWH} – Clock-to-clock constraint for a read-then-write operation

t_{C2CWRH} – Clock-to-clock constraint for a write-then-read operation

Values of these timing parameters are available in the datasheets. Refer to the appropriate datasheets for timing information (Table 1 on page 3).

Chapter 2: ProASIC3 Series FPGAs

This section describes the simultaneous read-write operations of dual-port SRAM in ProASIC3 series FPGAs. This section is applicable only to listed devices in **Table 4.**

Table 4 • **Supported Devices in Chapter 2**

Datasheet	User’s Guide
ProASIC3 Flash Family FPGAs	*ProASIC3 FPGA Fabric User’s Guide*
ProASIC3E Flash Family FPGAs	*ProASIC3E FPGA Fabric User’s Guide*
ProASIC3 nano Flash FPGAs	*ProASIC3 nano FPGA Fabric User’s Guide*
Military ProASIC3/EL Low Power Flash FPGAs (A3P250 and A3P1000)	*Military ProASIC3/EL FPGA Fabric User’s Guide* (A3P250 and A3P1000)
Automotive ProASIC3 Flash Family FPGAs	*Automotive ProASIC3 FPGA Fabric User’s Guide*

Simultaneous Write-Write Operations

Simultaneous write-write is defined as the situation when the two clocks, CLKA and CLKB, turn on very close to one another to initiate a write operation on the same address of the RAM.

The behavior in a write-write situation depends on what occurs at both the negative or closing edge of the clock and the positive or opening edge of the clock. Therefore, a small delay is required between the negative edges of CLKA and CLKB as well as the positive edges of CLKA and CLKB in order to successfully perform simultaneous write. In addition, neither clock can fully overlap one another. Example cases are described in Table 5 and discussed further in the next section.

Table 5 • **List of Simultaneous Write-Write Scenarios**

Case	Description	Output When Data is Read After Write-Write Operations
1	CLKB rises after CLKA, CLKB falls after CLKA	Data from CLKB
2	CLKB rises when CLKA falls, CLKB falls after CLKA	Data from CLKB
3	CLKB rises after CLKA falls, CLKB falls after CLKA falls	Data from CLKB

Case 1: CLKB Opening After CLKA, CLKA Closing Before CLKB

Since the write-write situation depends on what occurs at BOTH the open and close edges, a small delay (t_{C2CWWH}) is required between CLKA and CLKB on the opening edge as well as a small delay (t_{C2CWWL}) between the closing of the clocks. The last data that is written to the address is kept. In this example, CLKA opens before CLKB, and closes before CLKB. Hence, data that is written from CLKB is secured in the RAM address. This is valid for both pipelined and flow-through modes. Figure 15 illustrates CLKB opening after CLKA and CLKA closing before CLKB.

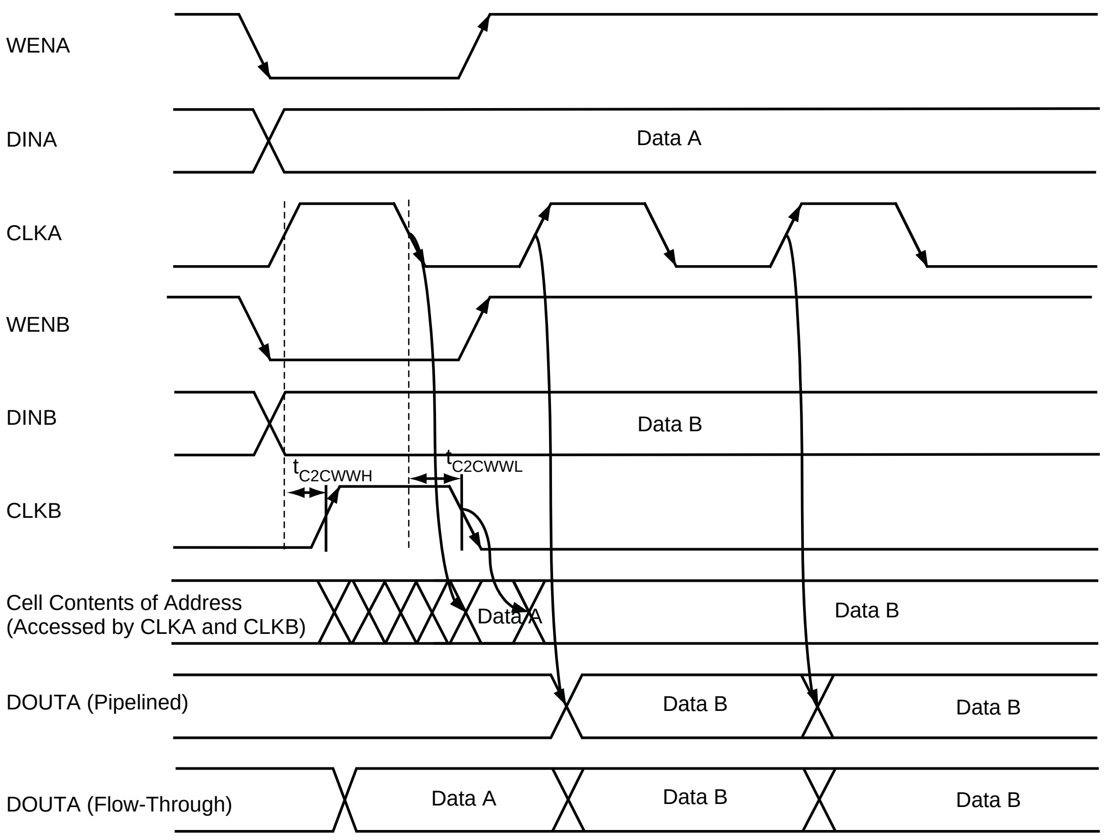

***Figure 15* • CLKB Opening After CLKA, CLKA Closing Before CLKB**

Case 2: CLKB Opening When CLKA Closes, CLKB Closes After CLKA Closes

There is actually no collision in this case; it is normal operation. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data written by CLKB is instantiated, no collisions occur. By intuition, the last data written to the address is stored. In this example, the data written from CLKB is stored in the memory. As a result, in the next cycle where the data is read, data B is read from the memory. The reverse is also true: if data written by CLKA precedes data written by CLKB, data that is written from CLKA is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 16 illustrates CLKB opening when CLKA closes and CLKB closing after CLKA closes.

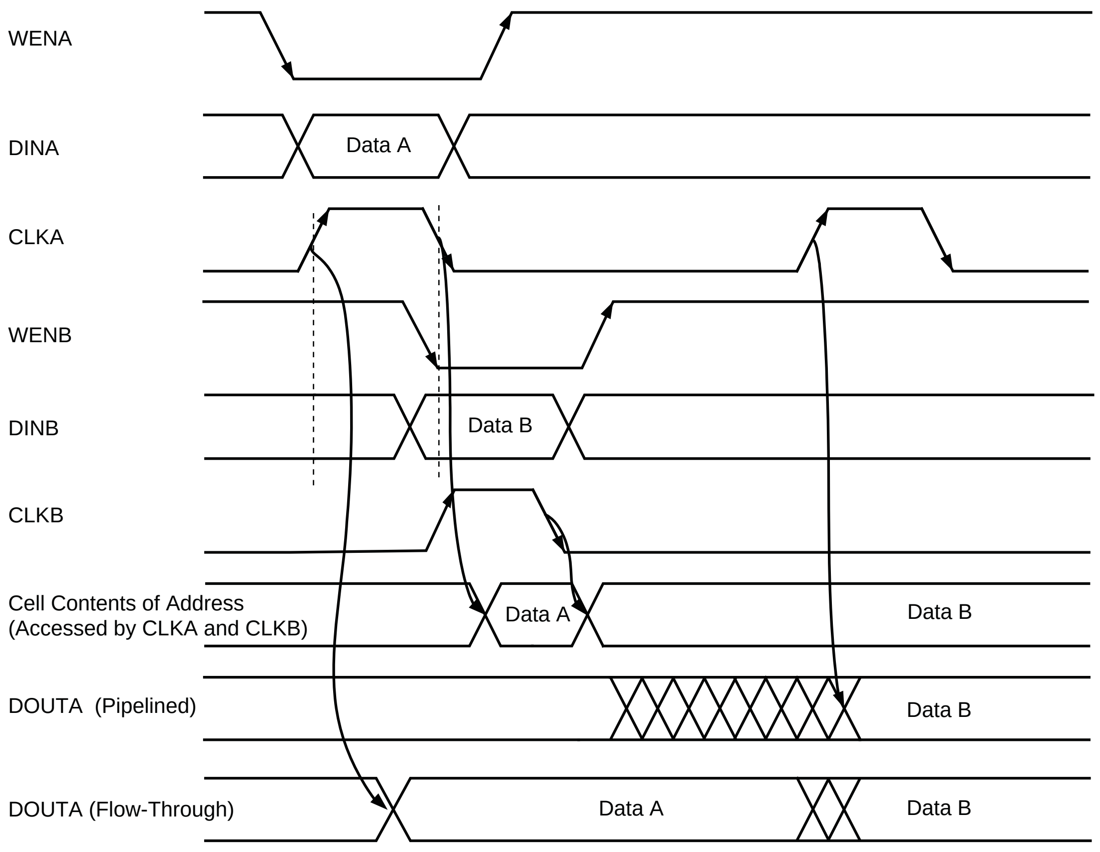

***Figure 16* • CLKB Opens When CLKA Closes, CLKB Closes After CLKA Closes**

Case 3: CLKB Opening After CLKA Closes, CLKB Closes After CLKA Closes

Similar to case 2, there is no collision in this scenario, and it is considered a normal operation case. Since data written by CLKA is completed before data written by CLKB is instantiated, no collisions occur. By intuition, the last data written to the address is stored. In this example, the data written from CLKB is stored in the memory. As a result, in the next cycle where the data is read, data B is read from the memory. The reverse is also true: if data written by CLKA precedes data written by CLKB, data that is written from CLKA is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 17 illustrates CLKB opening after CLKA closes, CLKB closes after CLKA closes.

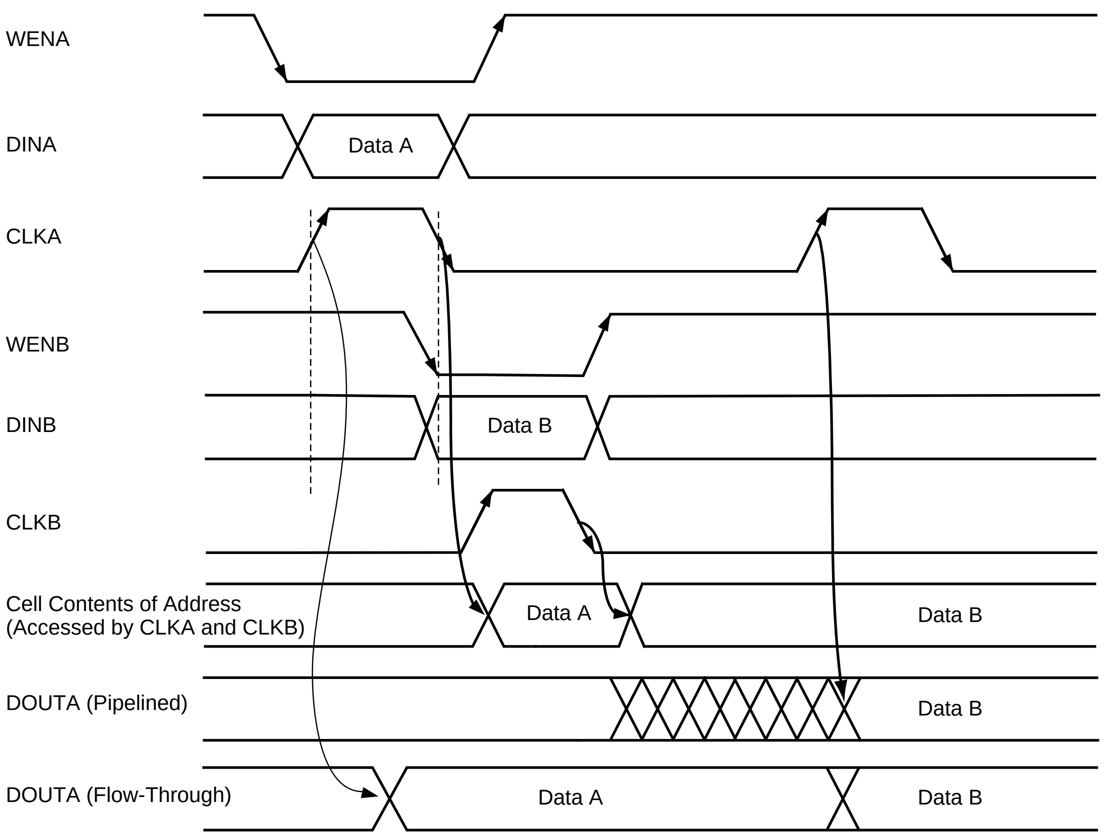

Figure 17 • **CLKB Opens After CLKA Closes, CLKB Closes After CLKA Closes**

Simultaneous Read-Write and Write-Read Operations

Simultaneous read-write is defined as the situation when the two clocks, CLKA and CLKB, will turn on very close to one another to initiate one write operation and one read operation on the same address of the RAM.

The behavior in a read-write or a write-read situation depends on what occurs at the positive or the opening edge of the clock. Since it takes time for the write operation to occur and also the read operation, a small delay is required between the positive edges of CLKA and CLKB in order to successfully perform a simultaneous write-read or read-write operation. All events except for events having both edges opening simultaneously will have a stable, known output. A write operation is performed for CLKA while a read operation is performed on CLKB. The various behaviors are described in Table 6. Each case is discussed in detail in the following sections.

Table 6 • **List of Simultaneous Write-Read, Read-Write Scenarios**

Case	Description	Output When Data is Read
1	Simultaneous clock on falling edge, CLKB rises after CLKA	Data from CLKA is read
2	CLKB rises after CLKA, CLKB falls before CLKA	Data from CLKA is read
3	CLKB rises after CLKA, CLKB falls after CLKA	Data from CLKA is read
4	CLKB rises before CLKA, CLKB falls before CLKA	Old data from address is read if delay between CLKA and CLKB is met; otherwise it would be unknown
5	Simultaneous clock on falling edge, CLKB rises before CLKA	Old Data from address is read if delay between CLKA and CLKB is met, else it would be unknown
6	CLKB rises when CLKA falls, CLKB falls after CLKA	Data from CLKA is read
7	Normal operation; no collision	Data from CLKA is read

Case 1: CLKB Opening After CLKA, Both Clocks Closing Simultaneously

Case 1 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line—access to SRAM cell is enabled by word line—is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, data that is written from CLKA is secured in the RAM address and can be read from CLKB. It is independent of the closing edges.

The reverse is also true: if CLKA opens after CLKB and both clocks close simultaneously (with data written from CLKB and read from CLKA), then data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 18 illustrates CLKB opens after CLKA, both clocks closing simultaneously.

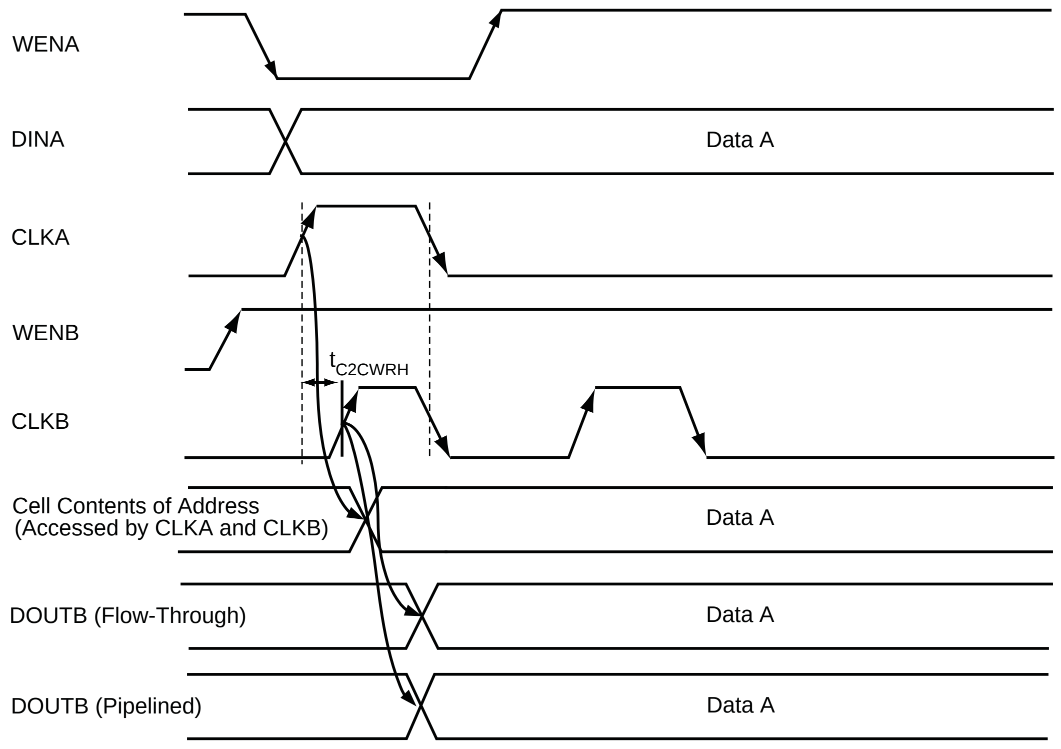

Figure 18 **• CLKB Opens After CLKA, Both Clocks Closing Simultaneously**

Case 2: CLKB Opening After CLKA, CLKB Closes Before CLKA

Similar to case 1, case 2 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, data that is written from CLKA is secured in the RAM address and can be read from CLKB. The ability of the data to be read accurately is independent of the closing edges, regardless of whether the clock edges close simultaneously or asynchronously.

The reverse is also true: If CLKA opens after CLKB and CLKA closes before CLKB (with data written from CLKB and read from CLKA), then data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 19 illustrates CLKB opens after CLKA and CLKB closes before CLKA.

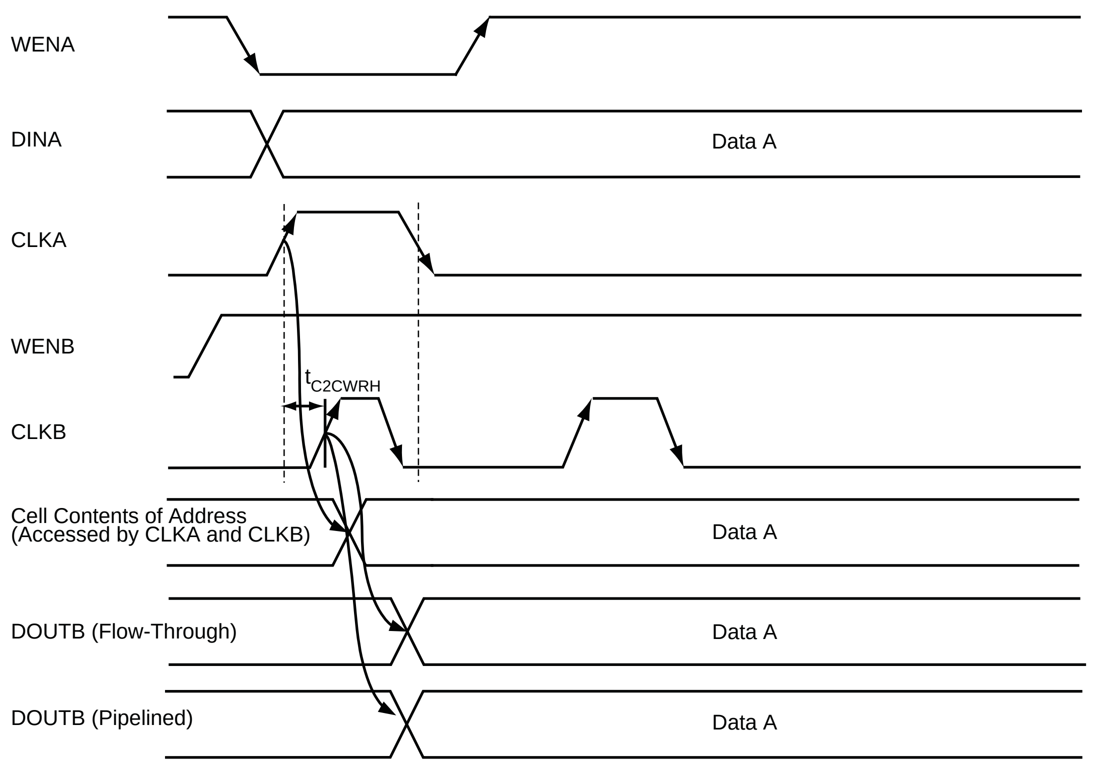

***Figure 19* • CLKB Opens After CLKA, CLKB Closes Before CLKA**

Case 3: CLKB Opening After CLKA, CLKB Closes After CLKA

Similar to case 1, case 3 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, data that is written from CLKA is secured in the RAM address and can be read from CLKB. The ability of the data to be read accurately is independent of the closing edges, regardless of whether the clock edges close simultaneously or asynchronously.

The reverse is also true: if CLKA opens after CLKB and CLKA closes before CLKB (with data written from CLKB and read from CLKA), then data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 20 illustrates CLKB opens after CLKA, CLKB closes after CLKA.

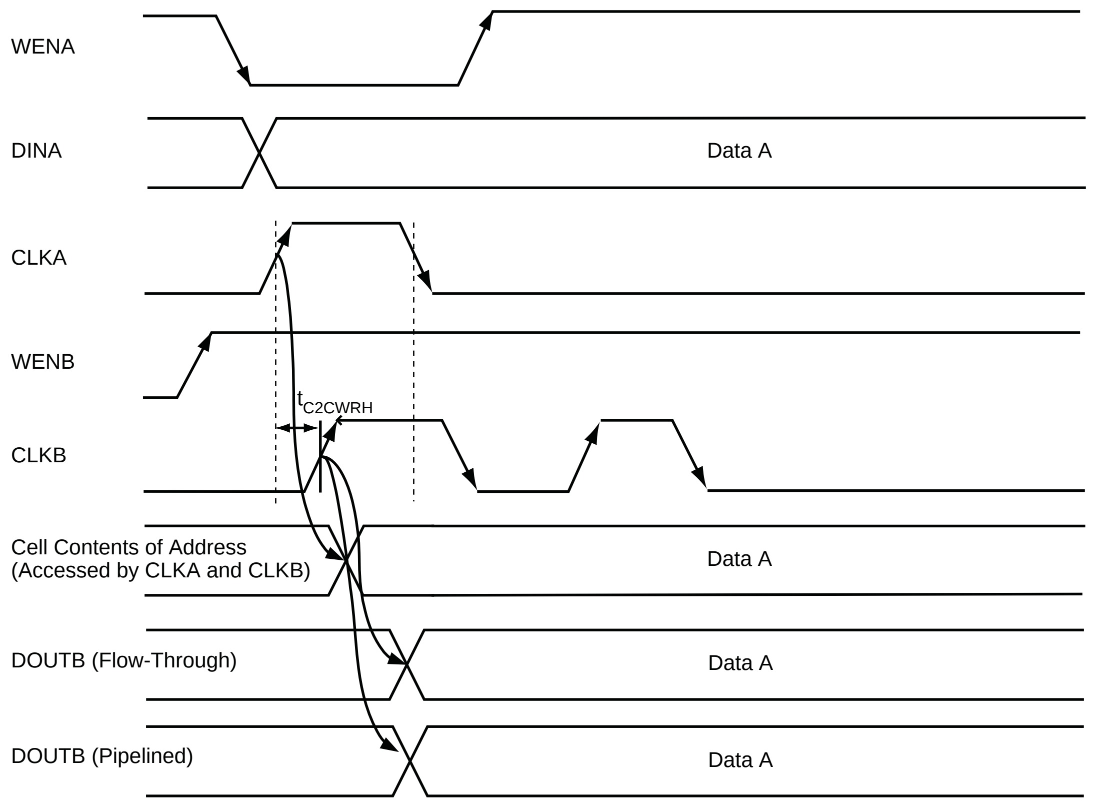

***Figure 20* • CLKB Opens After CLKA, CLKB Closes After CLKA**

Case 4: CLKB Opening Before CLKA Opens, CLKB Closes Before CLKA Closes

In this case, it is a read-write operation rather than a write-read operation. Since write and read operations cannot occur instantaneously, a small delay occurs between the read and write operations (t_{C2CRWH}). If this delay is met, the old data from the address is read by CLKB before new data overwrites the old data by CLKA. The reverse is also true: if data being read by CLKA precedes data written by CLKB, the old data in the address is read before new data from CLKB is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 21 illustrates CLKB opens before CLKA opens and CLKB closes before CLKA closes.

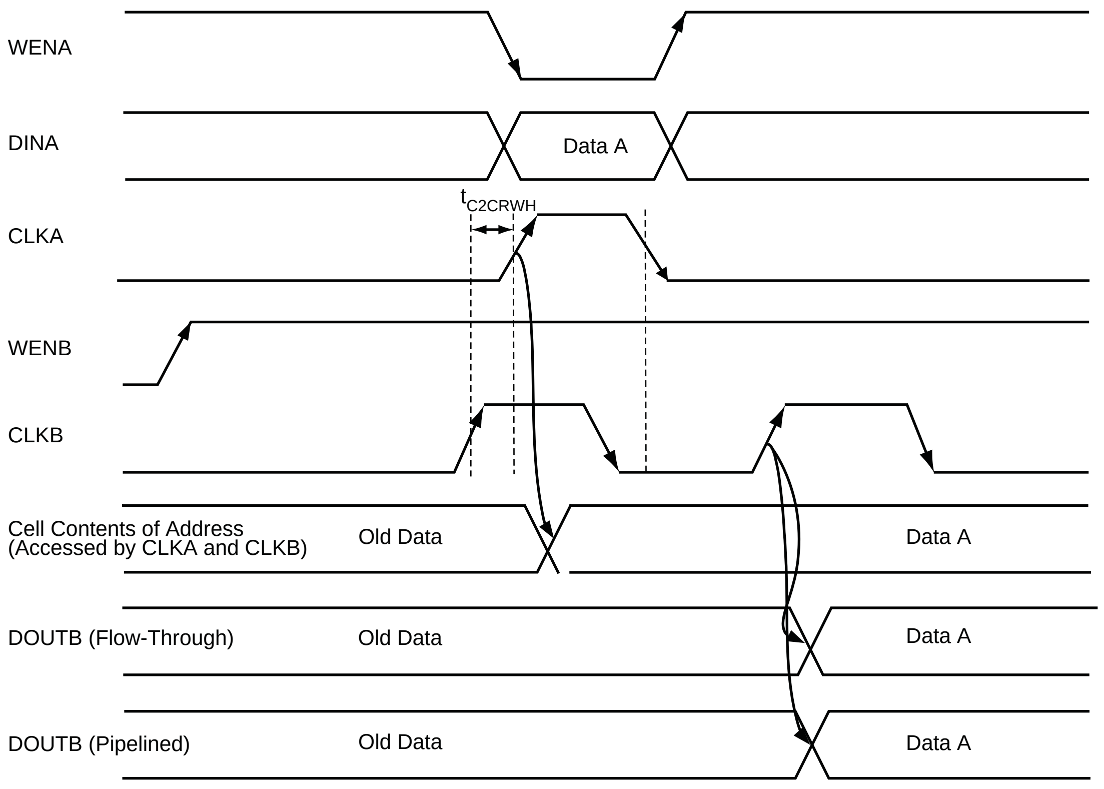

***Figure 21* • CLKB Opens Before CLKA Opens, CLKB Closes Before CLKA Closes**

Case 5: CLKB Opening Before CLKA Opens, CLKB Closes Simultaneously with CLKA

Similar to case 4, this is a read-write operation rather than a write-read operation. Since write and read operations cannot occur instantaneously, a small delay occurs between the read and write operations (t_{C2CRWH}). If this delay is met, the old data from the address is read by CLKB before new data overwrites the old data by CLKA. What is read on the read clock (CLKB) is independent of the closing edge. Therefore, whether CLKB closes before, after, or simultaneously with the CLKA edge is irrelevant. The reverse is also true: if data being read by CLKA precedes data written by CLKB, the old data in the address is read before new data from CLKB is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 22 illustrates CLKB opens before CLKA opens and CLKB closes simultaneously with CLKA.

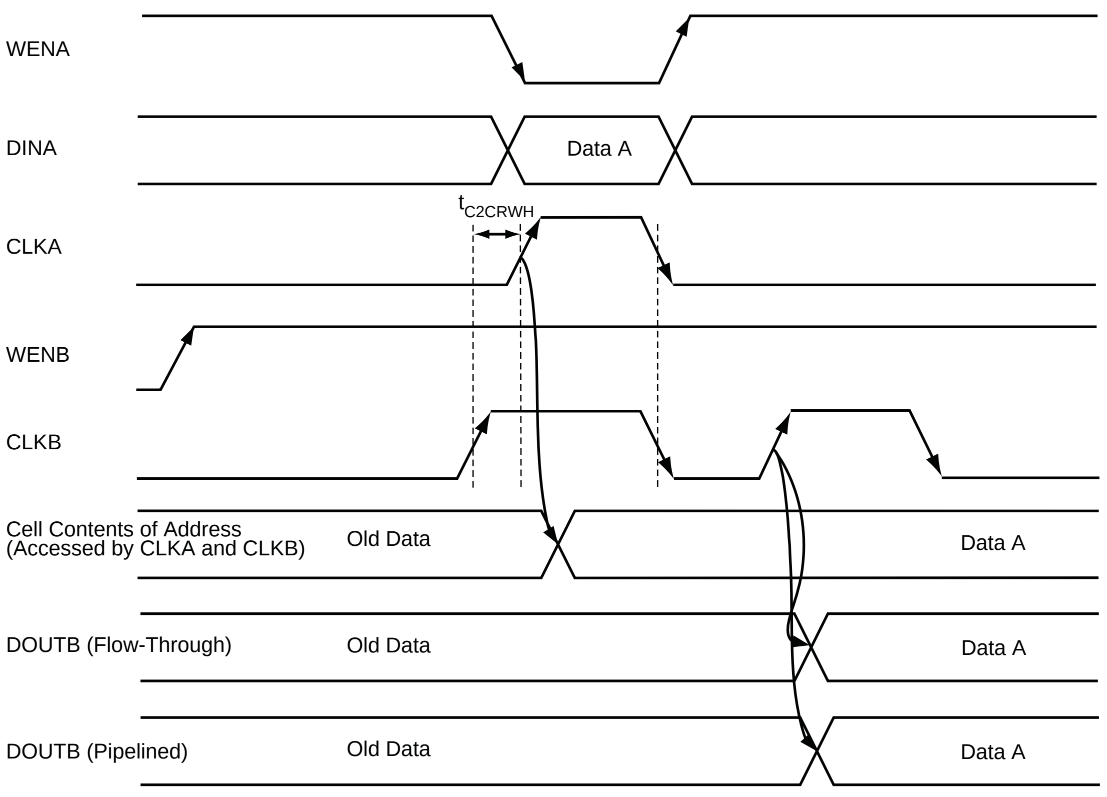

Figure 22 **• CLKB Opens Before CLKA Opens, CLKB Closes Simultaneously with CLKA**

Case 6: CLKB Opening When CLKA Closes, CLKB Closes After CLKA

There is no collision in case 6; it is considered a normal operation case. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data is read by CLKB, no collisions occur. By intuition, the last data written to the address is stored and read. In this example, the data written from CLKA is stored in the memory. As a result, when CLKB opens, the data written by CLKA in the same memory location is read. The reverse is also true: if data written by CLKB precedes data read by CLKA, data that is written from CLKB is secured into the memory and is read by CLKA. This is valid for both pipelined and flow-through modes. Figure 23 illustrates CLKB opening when CLKA closes, CLKB closes after CLKA.

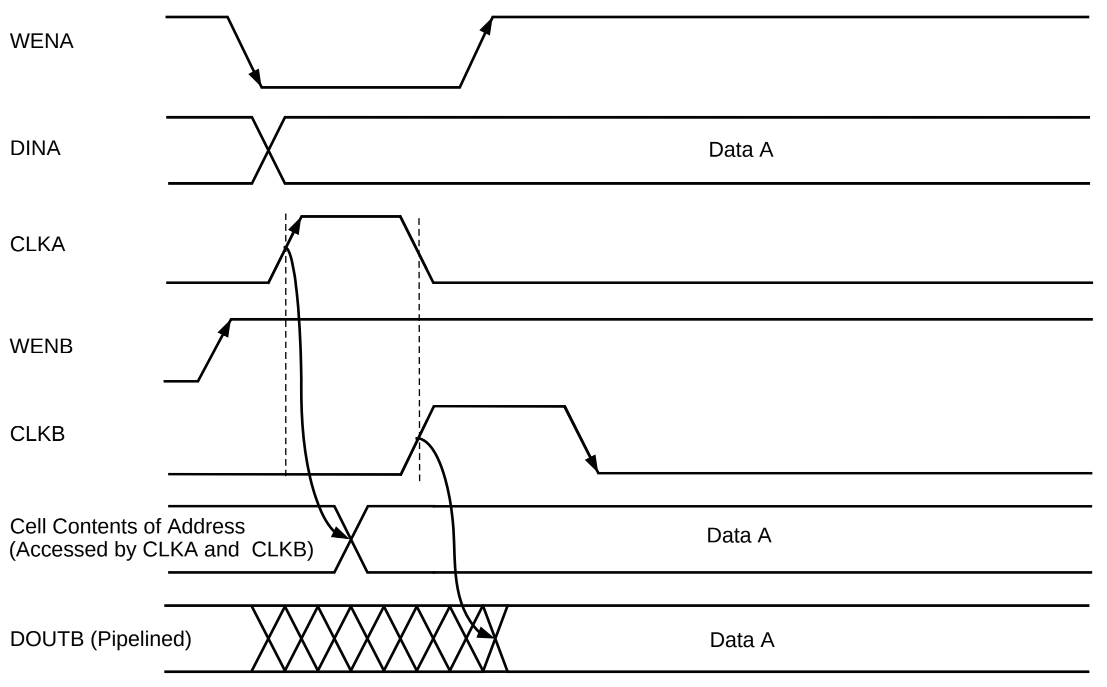

Figure 23 • **CLKB Opens When CLKA Closes, CLKB Closes After CLKA**

Case 7: CLKB Opening After CLKA Closes, CLKB Closes After CLKA

Similar to case 6, this is normal operation and there is no collision. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data is read by CLKB, no collisions occur. By intuition, the last data written to the address is stored and read. In this example, the data written from CLKA is stored in the memory. As a result, when CLKB opens, the data written by CLKA in the same memory location is read. The reverse is also true: if data written by CLKB precedes data read by CLKA, data that is written from CLKB is secured into the memory and is read by CLKA. This is valid for both pipelined and flow-through modes. Figure 24 illustrates CLKB opening after CLKA closes, CLKB closes after CLKA.

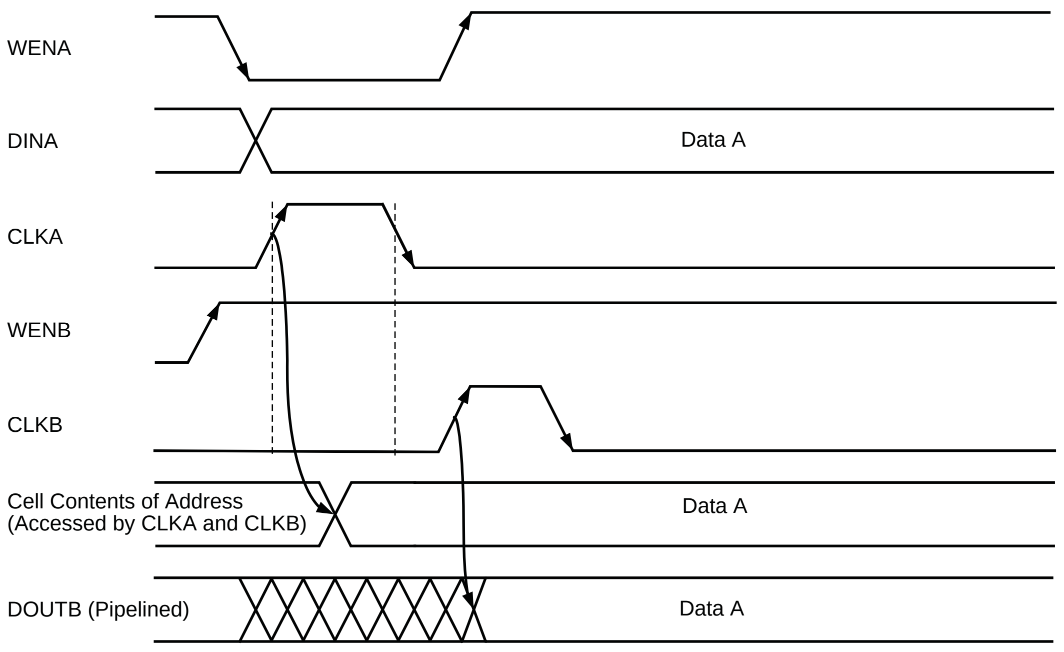

Figure 24 **• CLKB Opens After CLKA Closes, CLKB Closes After CLKA**

Note: The above descriptions of the cases assume that there are no violations in setup/hold times of the described signals (address setup/hold time and data setup/hold time). Three additional clock-to-clock constraints on the positive or opening edge and one clock-to-clock constraint on the negative or closing edge for write-write operation are required:

t_{C2CWWH} – Clock-to-clock constraint for a write-then-write operation between the opening edges

t_{C2CWWL} – Clock-to-clock constraint for a write-then-write operation between the closing edges

t_{C2CRWH} – Clock-to-clock constraint for a read-then-write operation

t_{C2CWRH} – Clock-to-clock constraint for a write-then-read operation

Values of these timing parameters are available in the datasheets. Refer to the appropriate datasheets of ProASIC3 devices for timing information (Table 4 on page 18).

Chapter 3: SmartFusion cSoCs and Fusion FPGAs

This section describes the simultaneous read-write operations of dual-port SRAM in SmartFusion and Fusion devices. This section is applicable only to listed devices in Table 7.

Table 7 • **Supported Devices in Chapter 3**

Series	Datasheet	User's Guide
SmartFusion	*SmartFusion Customizable System-on-Chip (cSoC)*	*SmartFusion FPGA Fabric User's Guide*
Fusion	*Fusion Family of Mixed Signal FPGAs*	*Fusion and Extended Temperature Fusion FPGA Fabric User's Guide*

Simultaneous Write-Write Operations

Simultaneous write-write is defined as the situation when the two clocks, CLKA and CLKB, turn on very close to one another to initiate a write operation on the same address of the RAM.

The behavior in a write-write situation depends on what occurs at the positive or the opening edge of the clock. The word line—access to SRAM cell is enabled by word line—is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. Therefore, only a small delay is required between the positive edges of CLKA and CLKB in order to successfully perform a simultaneous write. The various behaviors are described in Table 8. Each case is discussed in detail in the following sections.

Table 8 • **List of Simultaneous Write-Write Scenarios**

Case	Description	Write-Write Operations
1	Simultaneous clock on falling edge, CLKB rises after CLKA	Data from CLKB
2	CLKB rises after CLKA, CLKB falls before CLKA	Data from CLKB
3	CLKB rises after CLKA, CLKB falls after CLKA	Data from CLKB
4	CLKB rises when CLKA falls, CLKB falls after CLKA	Data from CLKB
5	CLKB rises after CLKA falls, CLKB falls after CLKA falls	Data from CLKB

Case 1: CLKB Opening After CLKA, Both Clocks Closing Simultaneously

In SmartFusion cSoCs and Fusion devices, the word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then CLKB, with a small delay (t_{C2CWWH}) between these two clocks, the last data that is written to the address will be kept. Thus, data that is written from CLKB is secured in the RAM address. The reverse is also true: if CLKA opens after CLKB and both clocks close simultaneously, then data that is written from CLKA is secured in the RAM address. This is valid for both pipelined and flow-through modes. Figure 25 illustrates CLKB opens after CLKA, both clocks closing simultaneously.

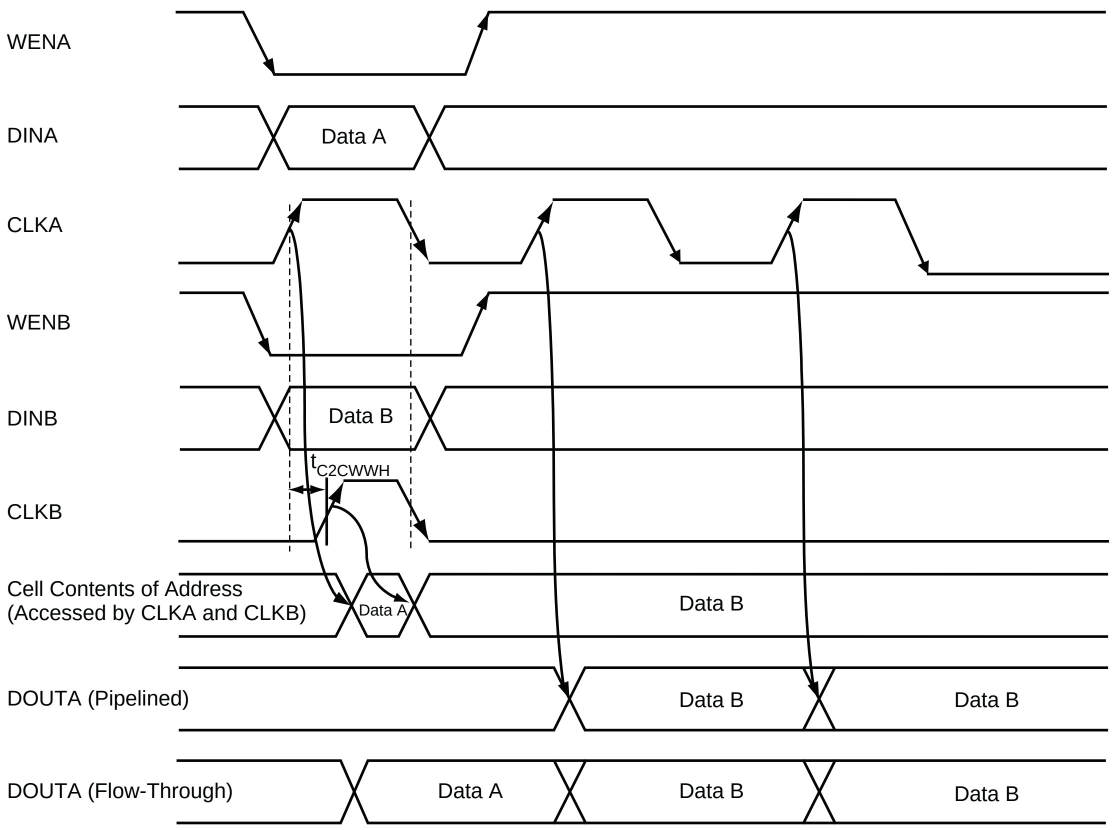

Figure 25 • **CLKB Opens After CLKA, Both Clocks Closing Simultaneously**

Case 2: CLKB Opening After CLKA, CLKB Closing Before CLKA

In Fusion devices, the word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then CLKB, with a small delay (t_{C2CWWH}) between these two clocks, the last data that is written to the address will be kept. Thus, data that is written from CLKB is secured in the RAM address. It is not dependent on the closing edge of the clocks. As such, whether CLKB closes before CLKA or CLKA closes before CLKB is irrelevant. The reverse is also true: if CLKA opens after CLKB and both clocks close simultaneously, then in this case, data that is written from CLKA is secured in the RAM address. This is valid for both pipelined and flow-through modes. Figure 26 illustrates CLKB opens after CLKA and CLKB closes before CLKA.

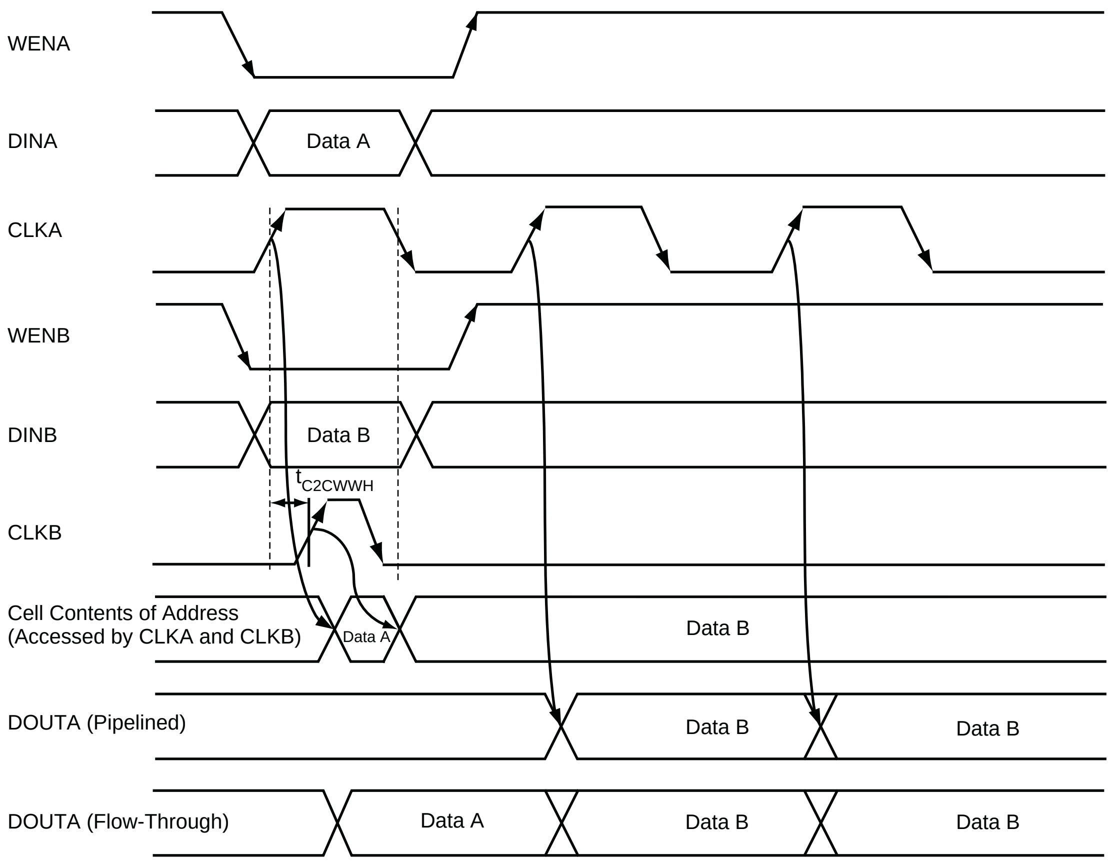

Figure 26 **• CLKB Opens After CLKA, CLKB Closes Before CLKA**

Case 3: CLKB Opening After CLKA, CLKA Closing Before CLKB

Similar to case 2, the word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then CLKB, with a small delay (t_{C2CWWH}) between these two clocks, the last data that is written to the address will be kept. Thus, data that is written from CLKB is secured in the RAM address. It is not dependent on the closing edge of the clocks. As such, whether CLKB closes before CLKA or CLKA closes before CLKB is irrelevant. The reverse is also true: if CLKA opens after CLKB and both clocks close simultaneously, then data that is written from CLKA is secured in the RAM address. This is valid for both pipelined and flow-through modes. Figure 27 illustrates CLKB opens after CLKA and CLKB closes after CLKA.

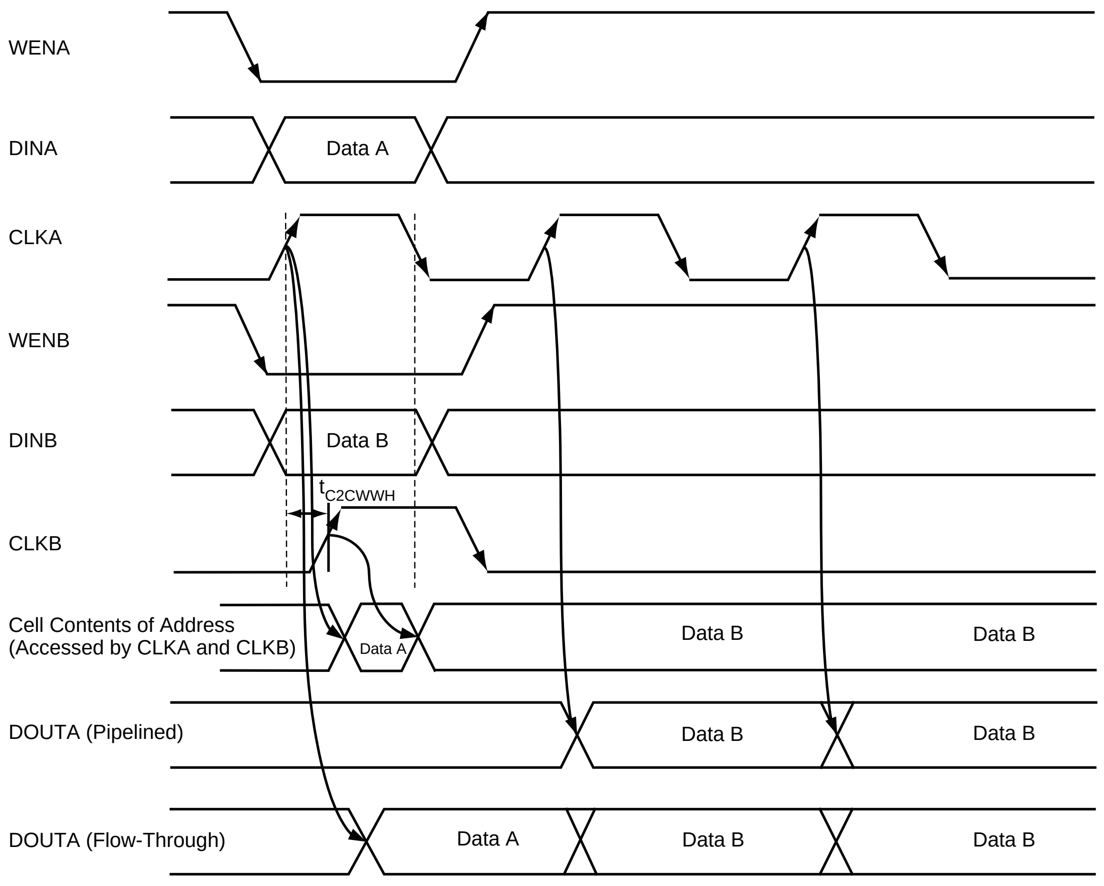

Figure 27 **• CLKB Opens After CLKA, CLKB Closes After CLKA**

Case 4: CLKB Opening When CLKA Closes, CLKB Closes After CLKA Closes

This case is actually what can be known as no collision, or a normal operation case. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data written by CLKB is instantiated, no collisions occur. By intuition, the last data written to the address is stored. In this example, the data written from CLKB is stored in the memory. As a result, in the next cycle where the data is read, data B is read from the memory. The reverse is also true: if data written by CLKA precedes data written by CLKB, data that is written from CLKA is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 28 illustrates CLKB opening when CLKA closes, CLKB closes after C.

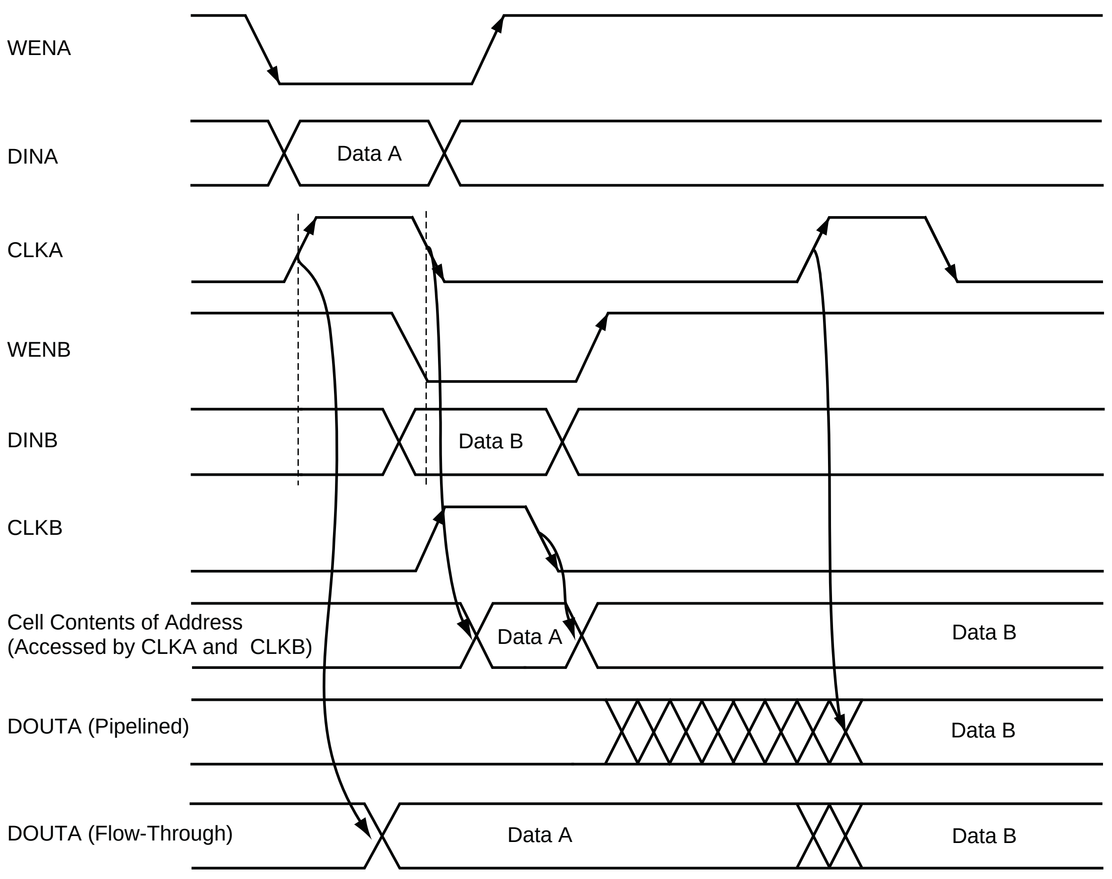

Figure 28 **• CLKB Opens When CLKA Closes, CLKB Closes After CLKA Closes**

Case 5: CLKB Opening After CLKA Closes, CLKB Closes After CLKA Closes

Similar to case 4, there is no collision in this scenario; it is considered a normal operation case. Since data written by CLKA is completed before data written by CLKB is instantiated, no collisions occur. By intuition, the last data written to the address is stored. In this example, the data written from CLKB is stored in the memory. As a result, in the next cycle where the data is read, data B is read from the memory. The reverse is also true: if data written by CLKA precedes data written by CLKB, data that is written from CLKA is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 29 illustrates CLKB opening after CLKA closes and CLKB closing after CLKA closes.

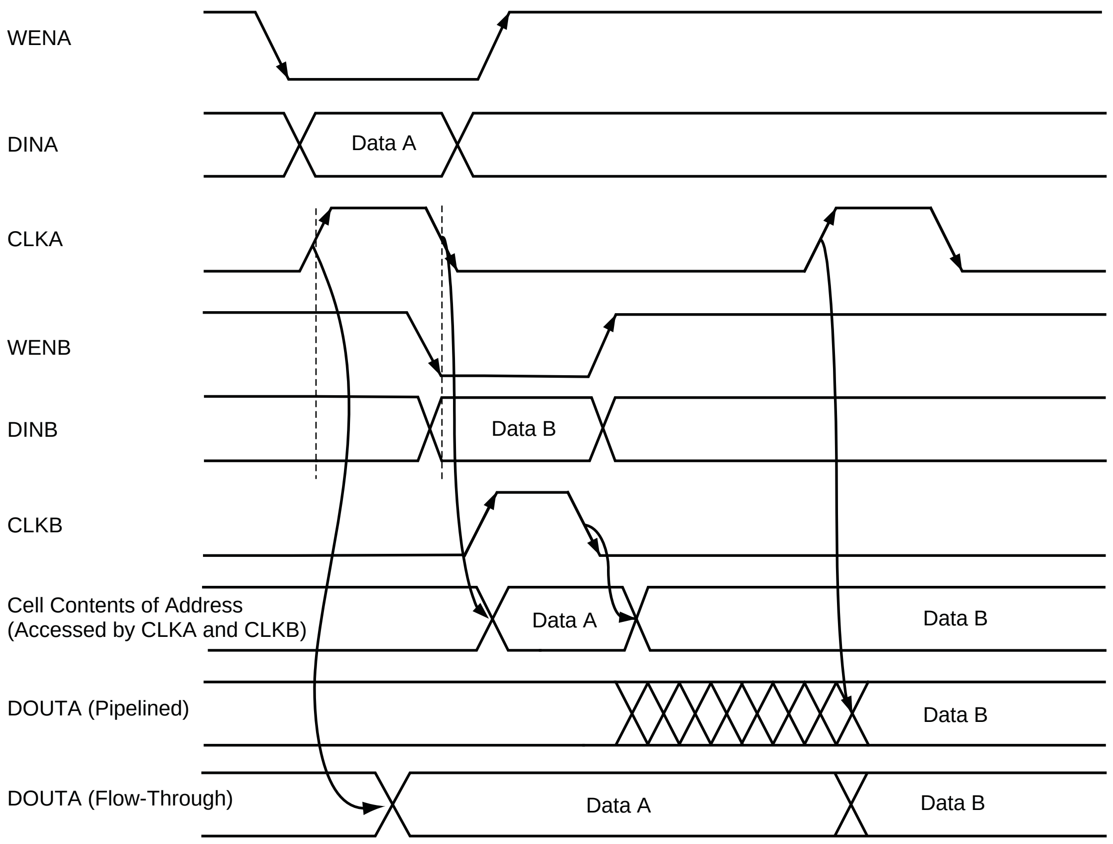

***Figure 29* • CLKB Opens After CLKA Closes, CLKB Closes After CLKA Closes**

Simultaneous Read-Write and Write-Read Operations

Simultaneous read-write is defined as the situation when the two clocks, CLKA and CLKB, turn on very close to one another to initiate one write operation and one read operation on the same address of the RAM.

The behavior in a read-write or a write-read situation depends on what occurs at the positive or the opening edge of the clock. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. Therefore, only a small delay is required between the positive edges of CLKA and CLKB in order to successfully perform a simultaneous write-read or read-write operations. All events except for events having both edges opening simultaneously will have a stable, known output. The various behaviors are described in Table 9. Each case is discussed in detail in the following sections. A write operation is performed for CLKA while a read operation is performed on CLKB.

***Table 9* • List of Simultaneous Write-Read, Read-Write Scenarios**

Case	Description	Output When Data is Read
1	Simultaneous clock on falling edge, CLKB rises after CLKA	Data from CLKA is read
2	CLKB rises after CLKA, CLKB falls before CLKA	Data from CLKA is read
3	CLKB rises after CLKA, CLKB falls after CLKA	Data from CLKA is read
4	CLKB rises before CLKA, CLKB falls before CLKA	Old data from address is read if delay between CLKA and CLKB is met; otherwise it would be unknown.
5	Simultaneous clock on falling edge, CLKB rises before CLKA	Old data from address is read if delay between CLKA and CLKB is met; otherwise it would be unknown.
6	CLKB rises when CLKA falls, CLKB falls after CLKA	Data from CLKA is read
7	CLKB rises after CLKA falls, CLKB falls after CLKA	Data from CLKA is read

Case 1: CLKB Opening After CLKA, Both Clocks Closing Simultaneously

Case 1 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, data that is written from CLKA is secured in the RAM address and can be read from CLKB. It is independent of the closing edges.

The reverse is also true: if CLKA opens after CLKB and both clocks close simultaneously (with data written from CLKB and read from CLKA), then data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 30 illustrates CLKB opens after CLKA, both clocks closing simultaneously.

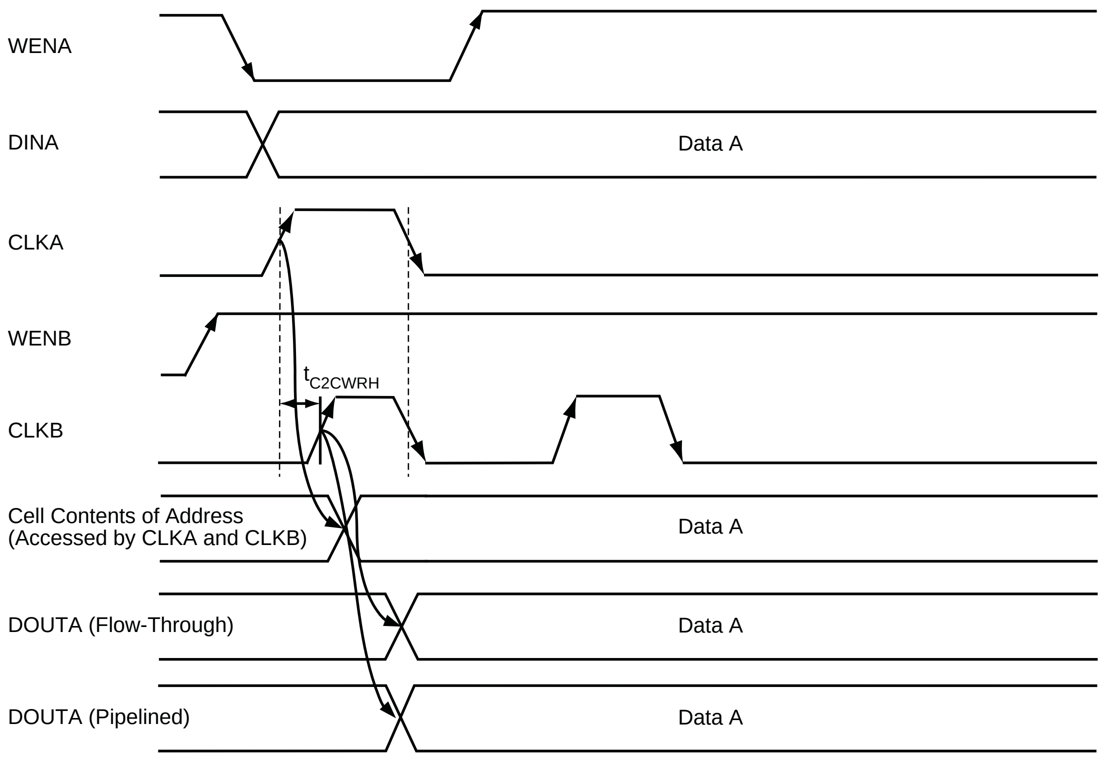

***Figure 30* • CLKB Opens After CLKA, Both Clocks Closing Simultaneously**

Case 2: CLKB Opening After CLKA, CLKB Closes Before CLKA

Similar to case 1, case 2 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock. If data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, data that is written from CLKA is secured in the RAM address and can be read from CLKB. The ability of the data to be read accurately is independent of the closing edges, regardless of whether the clock edges close simultaneously or asynchronously.

The reverse is also true: if CLKA opens after CLKB and CLKA closes before CLKB (with data written from CLKB and read from CLKA), then data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 31 illustrates CLKB opens after CLKA and CLKB closes before CLKA.

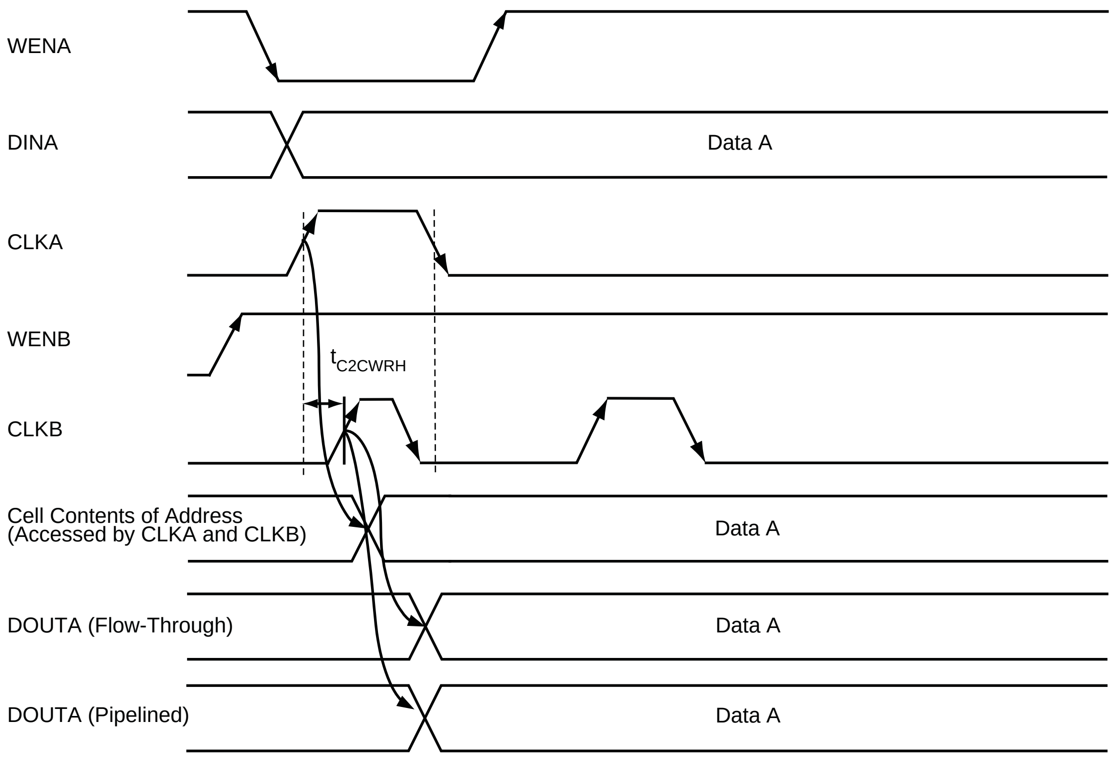

Figure 31 • **CLKB Opens After CLKA, CLKB Closes Before CLKA**

Case 3: CLKB Opening After CLKA, CLKB Closes After CLKA

Similar to case 1, case 3 is a write-read operation occurring in the same clock cycle at the same address in a dual-port SRAM. The word line is pulsed using timing circuits and is High only for a short period of time after the positive edge of a clock, if data is written from CLKA and then read from CLKB, with a small delay (t_{C2CWRH}) between these two clocks, the last data that is written to the address will be read. Thus, data that is written from CLKA is secured in the RAM address and can be read from CLKB. The ability of the data to be read accurately is independent of the closing edges, regardless of whether the clock edges close simultaneously or asynchronously.

The reverse is also true: if CLKA opens after CLKB and CLKA closes before CLKB (with data written from CLKB and read from CLKA), then in this case, data that is written from CLKB is secured in the RAM address and can be read by CLKA. This case is applicable for pipelined and flow-through modes. Figure 32 illustrates CLKB opens after CLKA and CLKB closes after CLKA.

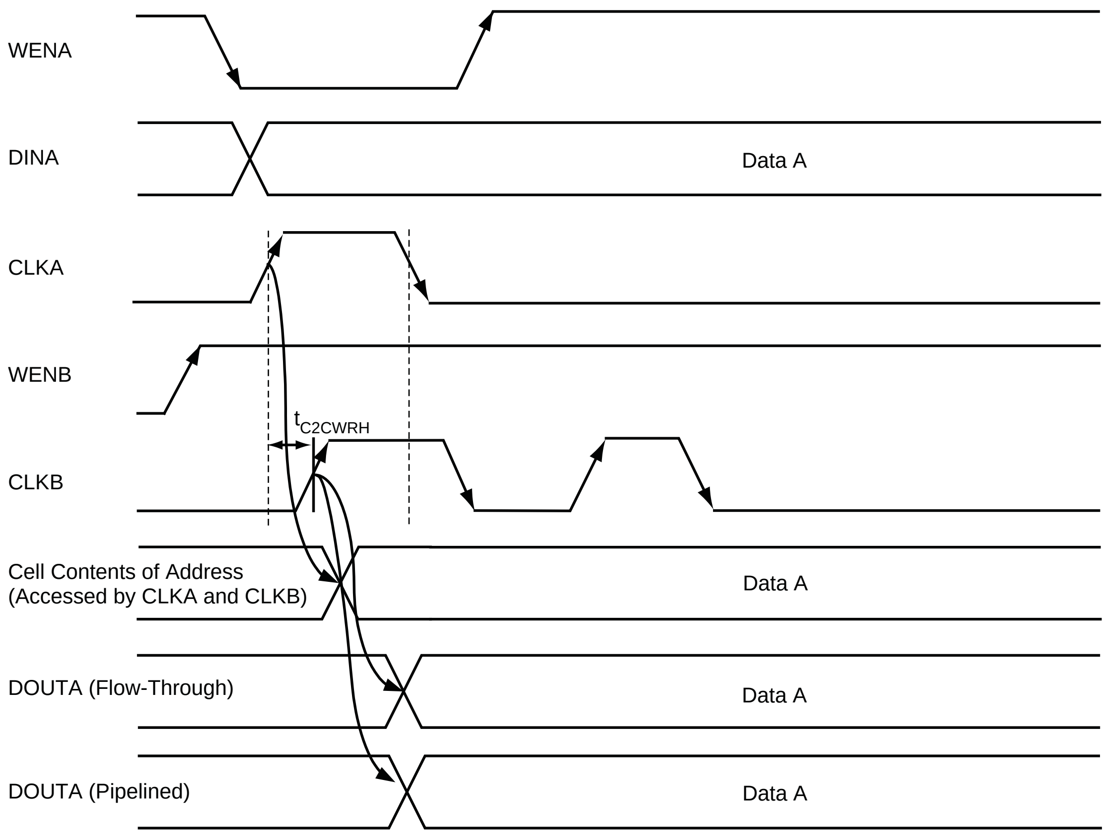

Figure 32 • CLKB Opens After CLKA, CLKB Closes After CLKA

Case 4: CLKB Opening Before CLKA Opens, CLKB Closes Before CLKA Closes

In this case, it is a read-write operation rather than a write-read operation. Since write and read operations cannot occur instantaneously, a small delay occurs between the read and write operations (t_{C2CRWH}). If this delay is met, the old data from the address is read by CLKB before new data overwrites the old data by CLKA. The reverse is also true: if data being read by CLKA precedes data written by CLKB, the old data in the address is read before new data from CLKB is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 33 illustrates CLKB opens before CLKA opens and CLKB closes before CLKA closes.

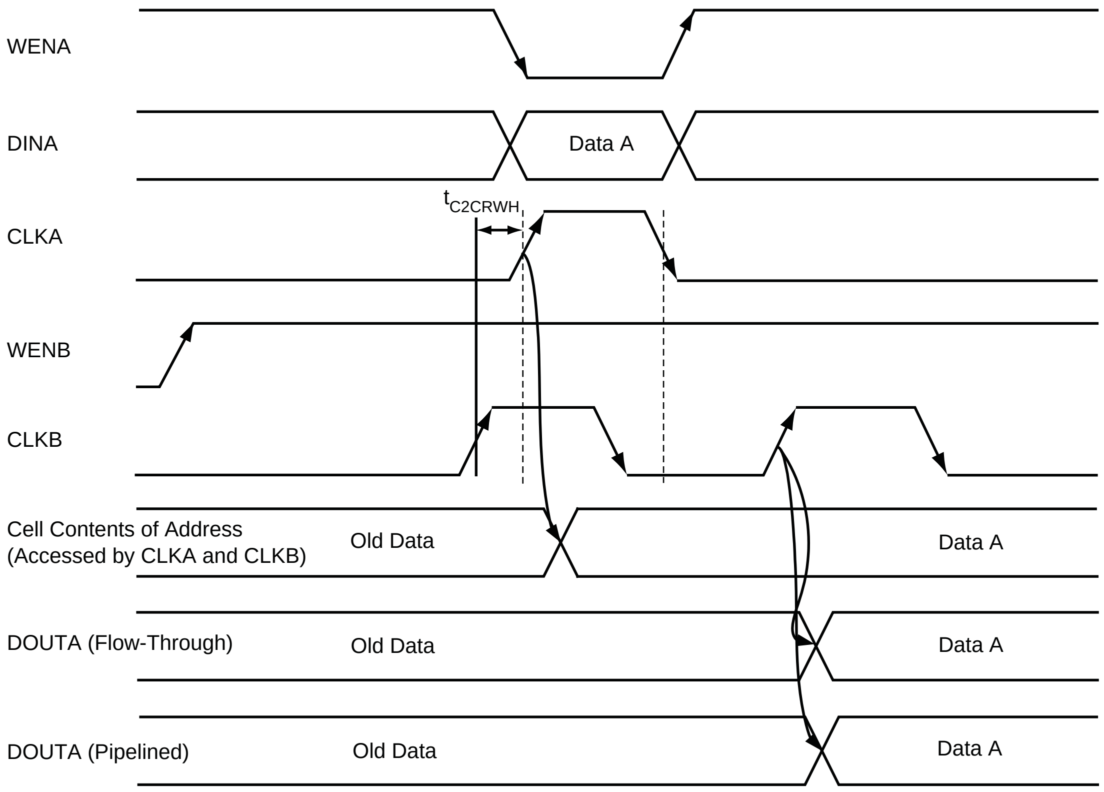

Figure 33 **• CLKB Opens Before CLKA Opens, CLKB Closes Before CLKA Closes**

Case 5: CLKB Opening Before CLKA Opens, CLKB Closes Simultaneously with CLKA

Similar to case 4, this is a read-write operation rather than a write-read operation. Since write and read operations cannot occur instantaneously, a small delay occurs between the read and write operations (t_{C2CRWH}). If this delay is met, the old data from the address is read by CLKB before new data overwrites the old data by CLKA. What is read on the read clock (CLKB) is independent of the closing edge. Therefore, whether CLKB closes before, after, or simultaneously with the CLKA edge is irrelevant. The reverse is also true: if data being read by CLKA precedes data written by CLKB, the old data in the address is read before new data from CLKB is secured into the memory. This is valid for both pipelined and flow-through modes. Figure 34 illustrates CLKB opens before CLKA opens and CLKB closes simultaneously with CLKA.

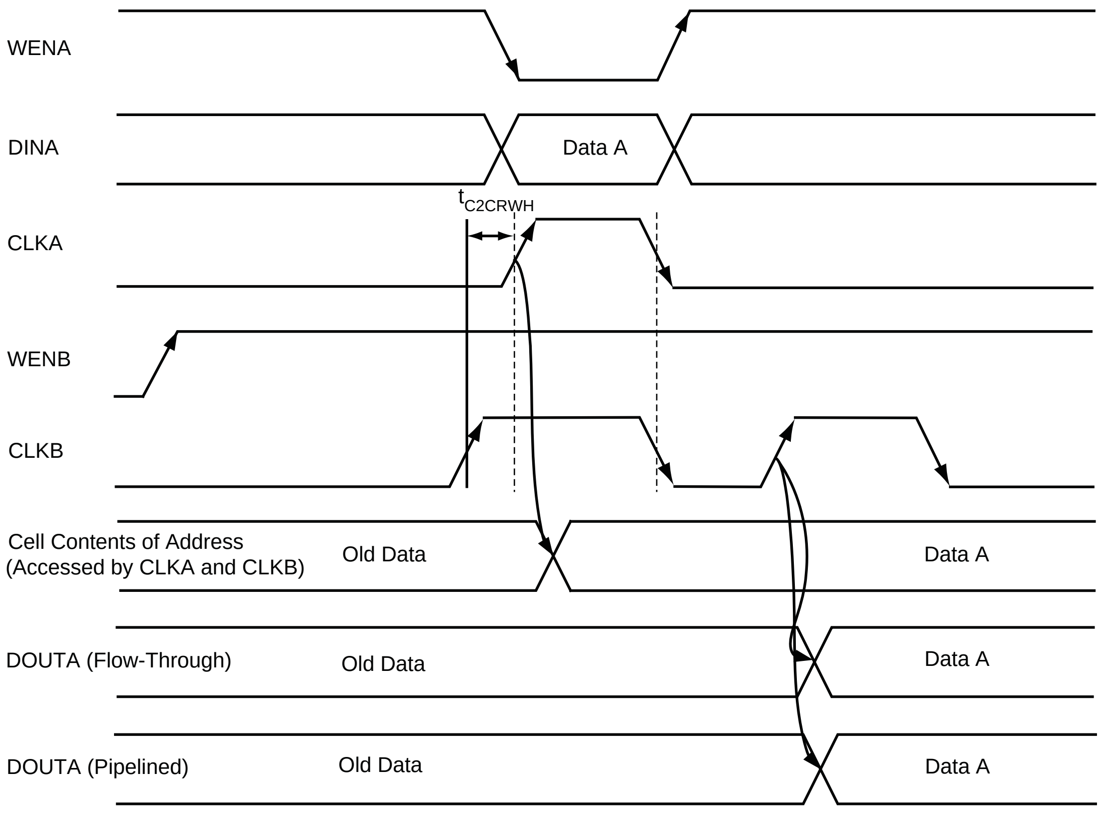

***Figure 34* • CLKB Opens Before CLKA Opens, CLKB Closes Simultaneously with CLKA**

Case 6: CLKB Opening when CLKA Closes, CLKB Closes After CLKA

There is no collision in case 6, and it is considered a normal operation case. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data is read by CLKB, no collisions occur. By intuition, the last data written to the address is stored and read. In this example, the data written from CLKA is stored in the memory. As a result, when CLKB opens, the data written by CLKA in the same memory location is read. The reverse is also true: if data written by CLKB precedes data read by CLKA, data that is written from CLKB is secured into the memory and is read by CLKA. This is valid for both pipelined and flow-through modes. Figure 35 illustrates CLKB opening when CLKA closes and CLKB closes after CLKA.

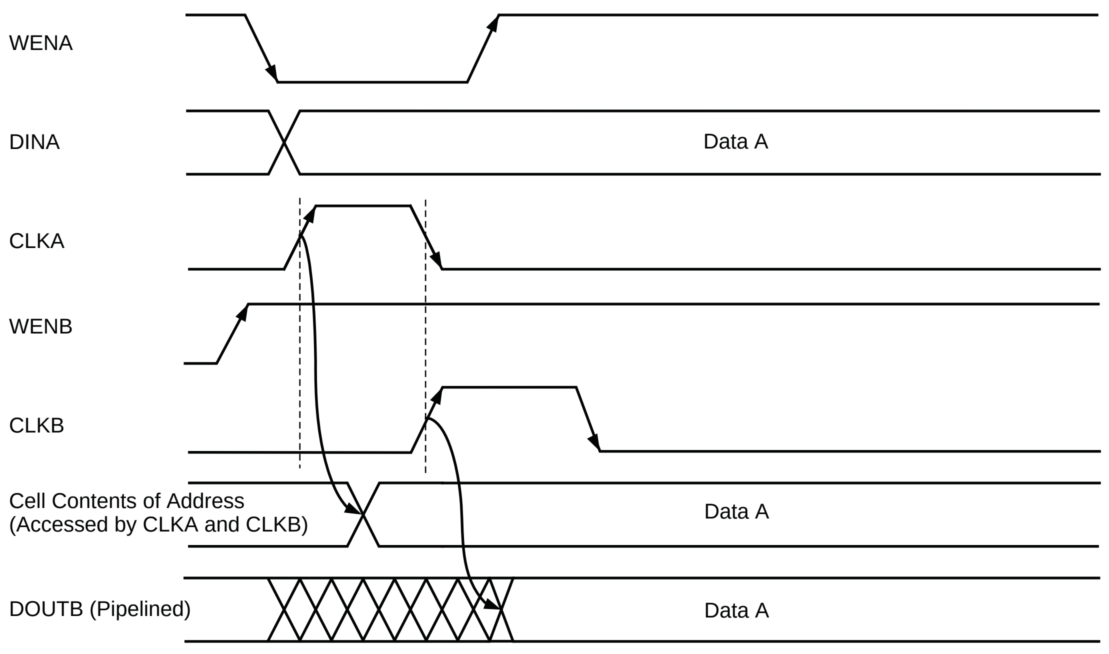

Figure 35 **• CLKB Opens When CLKA Closes, CLKB Closes After CLKA**

Case 7: CLKB Opening After CLKA Closes, CLKB Closes After CLKA

Similar to case 6, this is normal operation and there is no collision. Thus, there is no additional constraint under this condition. Since data written by CLKA is completed before data is read by CLKB, no collisions occur. By intuition, the last data written to the address is stored and read. In this example, the data written from CLKA is stored in the memory. As a result, when CLKB opens, the data written by CLKA in the same memory location is read. The reverse is also true: if data written by CLKB precedes data read by CLKA, data that is written from CLKB is secured into the memory and is read by CLKA. This is valid for both pipelined and flow-through modes. Figure 36 illustrates CLKB opening after CLKA closes and CLKB closes after CLKA.

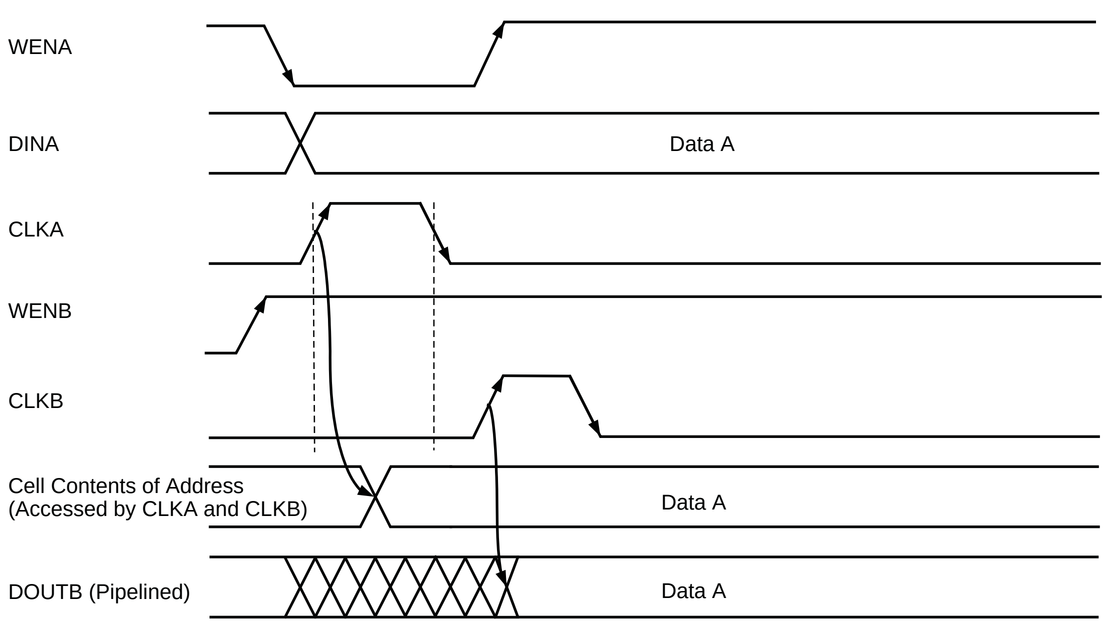

***Figure 36* • CLKB Opens After CLKA Closes, CLKB Closes After CLKA**

Note: The above descriptions of the cases assume that there are no violations in setup/hold times of the described signals (address setup/hold time and data setup/hold time). Three additional clock-to-clock constraints on the positive or opening edge are required:

t_{C2CWWH} – Clock-to-clock constraint for a write-then-write operation

t_{C2CRWH} – Clock-to-clock constraint for a read-then-write operation

t_{C2CWRH} – Clock-to-clock constraint for a write-then-read operation

Values of these timing parameters are available in the datasheets. Refer to the appropriate datasheets for timing information (Table 7 on page 30).

Conclusion

Microsemi suggests that where at all possible, avoid simultaneous read-write activities to ensure correct data written and read from the same address. If it is necessary to implement such configurations, be advised that only the aforementioned cases will provide known output when data is written or read from the same address. Review carefully, prior to implementation, whether or not your design satisfies the criteria discussed here.

Microsemi Corporate Headquarters
One Enterprise Drive, Aliso Viejo CA 92656
Within the USA: (800) 713-4113
Outside the USA: (949) 221-7100
Fax: (949) 756-0308 · www.microsemi.com

51900242-0/8.11