AN1779

Using the MCM69D536/MCM69D618 NetRAMs with Different Speed Computing Elements

Prepared by: Mike Parks, Freescale Semiconductor, Inc. Austin, TX

INTRODUCTION

In many cases, a dual port RAM is used to allow two computing elements to communicate, even though they are running at different speeds. Motorola's MCM69D536 and MCM69D618 synchronous dual port products are well–suited to this task, if the clock driving the slower computing element is derived from the clock driving the faster device. Figure 1 shows the case where the slower side is running at one–third the rate of the faster side.

The MCM69D536 and MCM69D618 NetRAMs employ a standard, four–transistor SRAM cell. These SRAMs accept a single clock that regulates the transfer of data through both the X and the Y ports. Internally, the memory core runs at twice the input clock speed, dividing the bandwidth between the two ports. For example, the MCM69D536TQ6 accepts an input clock of up to 83 MHz. In this case, the memory runs at up to 166 MHz, allowing each of the two ports to transfer 36 bits at a rate of 83 MHz per port.

Given the single clock input, it may seem surprising that the NetRAMs can support an application where the two computing elements run at different speeds. However, the design of the NetRAM interface very cleanly handles reads and writes from a computing element running at a lower speed. When being read, the NetRAM provides data two clock cycles after the read is initiated. The lower speed interface is designed to take advantage of that fact, latching data from the NetRAM two clocks after it initiates a read transaction. Write cycles are implemented even more simply and cleanly, since the NetRAM latches address, date, and $\overline{W}$ signals on the same rising edge of its clock pin during a write transaction. Figure 2 shows the timing diagram for the low speed port interface for the case where it runs at one–third the rate of the high speed port. The high speed computing element is assumed to be running at the same clock speed as the NetRAM, using the data sheet timing.

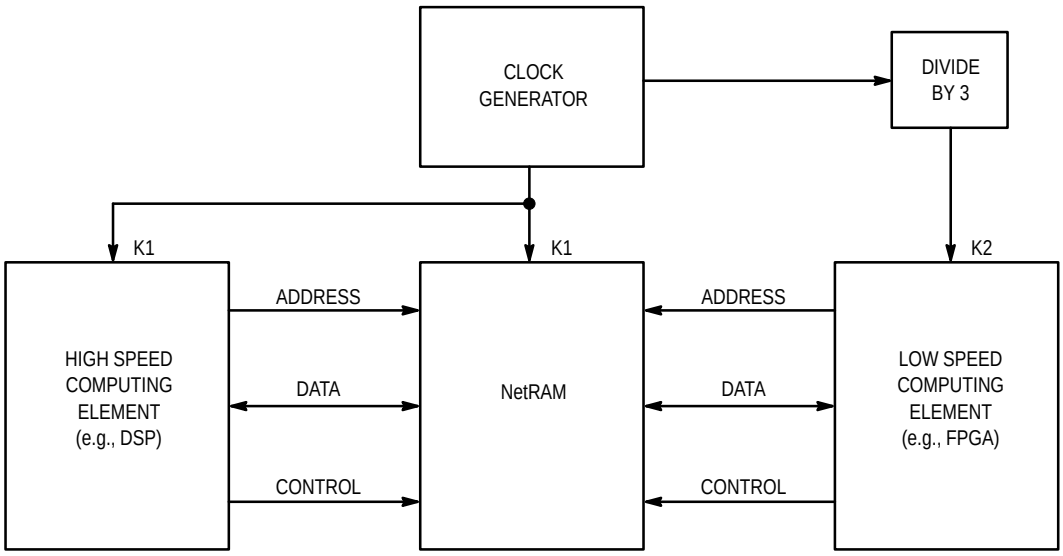

Figure 1. Different Speed Computing Elements

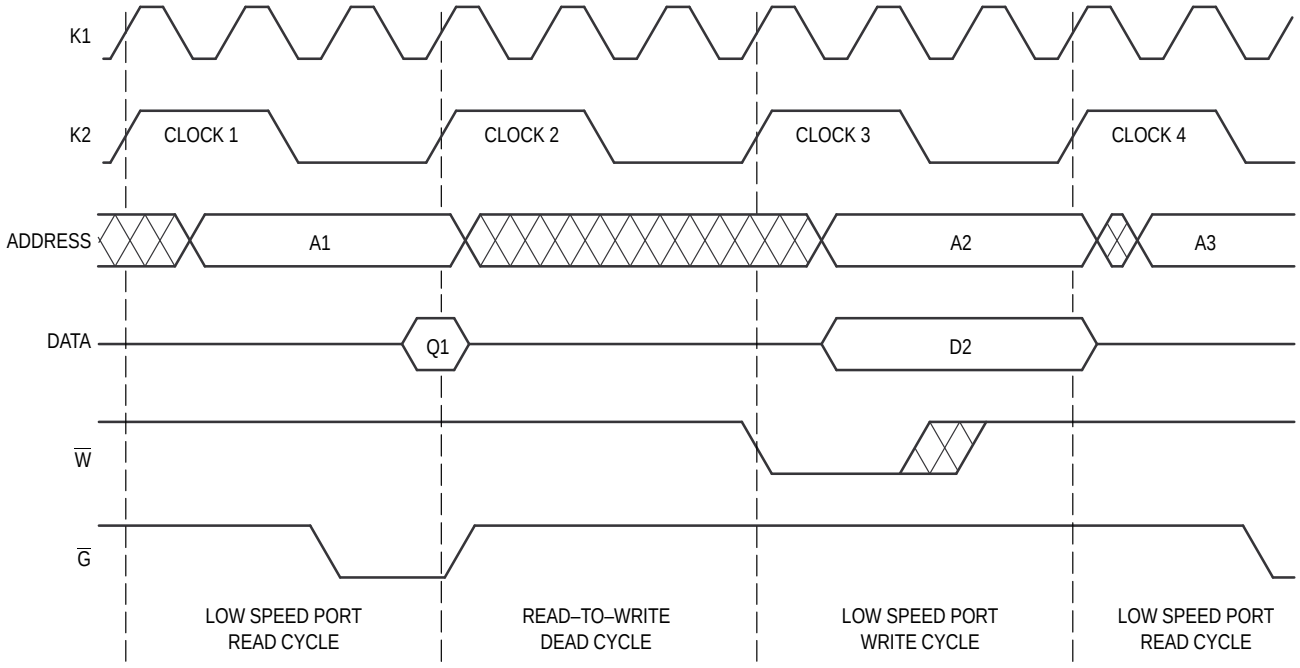

Figure 2. One–Third Speed Port Timing Diagram

GENERAL CASE: 3–TO–1 OR HIGHER RATIO

The first thing to note about Figure 2 is that the NetRAM does not see the low speed clock, K2. K2 drives the lower speed computing element, which generates the address, data input, $\overline{W}$, and $\overline{G}$ signals that are applied to the NetRAM. However, the NetRAM latches the address, data, and $\overline{W}$ signals on the rising edge of the high speed clock, K1. ($\overline{G}$ is an asynchronous input to the NetRAM.) As a result, the phase delay and jitter between K1 and K2 must be considered when designing the interface.

The first read transaction in Figure 2 is initiated on the rising edge of K2 that is labeled clock 1. The low speed computing element outputs the desired address while negating the $\overline{W}$ and $\overline{G}$ signals, and placing the data bus in a high impedance state. On the falling edge of clock 1, the $\overline{G}$ signal is asserted to allow the NetRAM to drive the data bus. The low speed computing element latches this data and negates $\overline{G}$ on the rising edge of clock 2.

In this example, the next bus transaction on the low speed port is a write. This read–to–write transition requires a dead cycle to avoid contention, as both the NetRAM and computing element try to drive the data bus. During this dead cycle, which lasts all of clock 2, both $\overline{W}$ and $\overline{G}$ are negated, the data bus is placed in high impedance, and the value on the address bus is a don't care.

Clock 3 depicts a write cycle. The write is initiated on the rising edge of K2, by the low speed computing element driving the desired address and data, while asserting $\overline{W}$ and negating $\overline{G}$. These values are latched by the NetRAM on the next rising edge of the K1 clock. The $\overline{W}$ signal is negated on the falling edge of clock 3. Depending on the delay in negating $\overline{W}$, note that another write may occur on the next rising edge of K1. However, if the second write does occur, it will be to the same address and with the same data as the first write. In other words, no harm will be done and no special care is needed to avoid this situation.

As illustrated in clock 4, a read transaction can be initiated after a write with no dead cycles. In fact, the read–to–write transition is the only case that does require a dead cycle.

While Figure 2 illustrates the case where the low speed port runs at one–third the rate of the high speed port, any other integer ratio can be used. Higher ratios, such as 4:1, are very straightforward. All of the signals are driven by the same clock edges; they simply are asserted longer. For example, addresses are asserted on one rising edge of K2 and are held until the next rising edge of K2.

SPECIAL CASE: 2–TO–1 RATIO

Using a 2:1 ratio is a special case, however, due to the pipelined nature of the NetRAM read cycles. The low speed computing element interface must be designed in a pipelined fashion as well, since a read initiated on the rising edge of one K2 cycle will be completed on the falling edge of the subsequent K2 cycle. Figure 3 illustrates the same sequence of bus transactions as Figure 2, namely, read–write–read. Note that the first read is initiated in K2 clock 1 and completed in clock 2. No transfer is initiated during clock 2, since a dead cycle is required between reads and writes. A write is initiated and completed during clock 3, while the final read is initiated during clock 4. This second read is completed during clock 5. Either a read or a dead cycle can be initiated during clock 5, since dead cycles are required between read–to–write transitions.

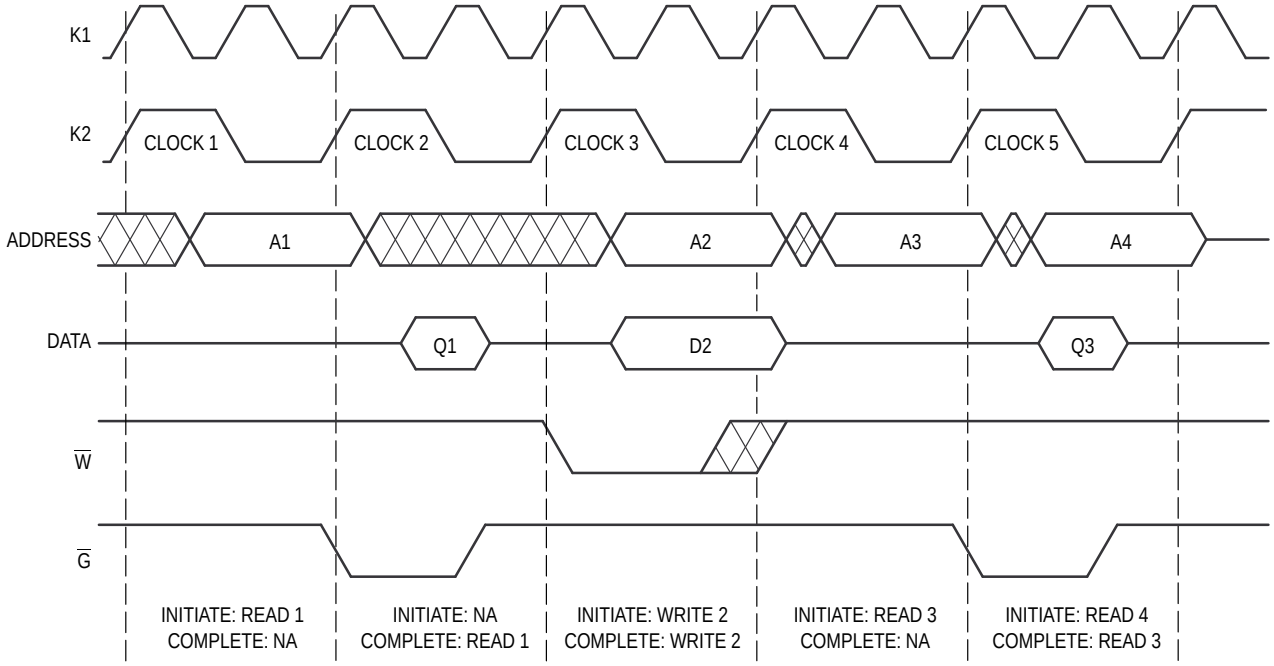

Figure 3. One–Half Speed Port Timing Diagram