Design provides single-port-to-dual-port SDRAM converter

Yu-Chieh Chen, Instrument Technology Research Center, National Applied Research Laboratories, Hsinchu, Taiwan; Edited by Martin Rowe and Fran Granville - March 03, 2011

In multimedia and video applications, DRAM buffers large frames in the image coding and decoding between an image-sensing device and an image-display device (references 1 and 2). It also plays a key role in image-frame-rate converters by matching the displaying format of the monitor (Reference 3). Accessing dual-port SDRAM is a complex task, especially when two data read or write operations are occurring simultaneously into a single-port SDRAM during high-speed operation.

SDRAM supports asynchronous control, performs reading and writing operations, and allows multiple CPUs to read or write in parallel. To reduce production costs and avoid complexity, engineers typically prefer conventional one-port SDRAM. You can, however, use an FPGA to convert a single-port SDRAM to function as a dual-port SDRAM.

The single-port-to-dual-port SDRAM converter comprises two read-FIFO (first-in/first-out) and write-FIFO memory blocks. A pair of read and write controllers each control a read-and-write data bus. Figure 1 shows the architecture of the proposed converter, and Figure 2 shows the operating flow of the read-and-write controller. The write controller first stores the writing data in write-FIFO and executes the SDRAM write command when the WKO (write-keep-out) signal falls to a low level, which means that the read operation is complete. Meanwhile, the RKO (read-keep-out) signal goes high until the write operation is complete.

The procedure prevents the read signal from occurring during the writing operation.
The read-data operation is similar to the write-data operation. The read controller first waits for the RKO signal to fall low if it is high when the read operation starts. The read controller then executes the read command and stores the data from SDRAM to read-FIFO memory. The read controller simultaneously pulls the WKO signal high until the read operation is complete. **Figure 3** shows the hardware logic of the smart controlling strategy, which the following equation also describes:

\[ f_{CONF} = \text{MUX}_{\text{READCOM}} \cdot \text{WRITECOM} \cdot \overline{\text{RKO}} \cdot \overline{\text{WKO}}, \]

where MUX is the multiplexer. For the read example, the SEL (select) signal switches to the read mode and then waits for the RKO signal to fall low if the RKO signal is at a high level when the read command executes *(Reference 1).*

This strategy doubles the high-level period of both the read operation, or RKO signal, and write operation, or WKO signal. These signals rise to logic high before executing each read or write operation. This step ensures that the read operation does not interrupt the write operation and that the write operation does not interrupt the read operation.

**Figure 4** shows the simulation waveform of the converter, which is a part of the image-sensing and displaying sequence. The iSDRAM_Read signal is high, indicating that the SDRAM is performing a read operation. At the same time, the iSDRAM_Write signal is high during the SDRAM-writing operation. The iSDRAM_ReadKeepOut signal goes high before each iSDRAM_Write signal and lasts two times longer than each iSDRAM_Write signal. This setup ensures that the writing command does not execute during the read operation. For the same reason, the read command doesn’t execute during the writing operation.

The write-clock operation is initially ahead of the read-clock operation. After several time delays, however, the write-clock operation catches up with the read-clock operation. At this point, the controller holds the data and waits until the iSDRAM_ReadKeepOut command falls to a low level before executing the read command. Using this read-and-write strategy, the controller resolves the conflict of executing read and write commands during write or read operations.
References