VHDL program enables PCI-bus-arbiter core

Antonio Di Rocco, Selex Communications, Chieti, Italy; Edited By Charles H Small and Fran Granville - September 13, 2007

This Design Idea describes a VHDL implementation of a PCI 2.2-bus arbiter (Figure 1). Any PCI system may have one or more PCI-master devices. Most devices can behave as target hosts, but one must be a PCI-bus initiator, or master. Normally, only microprocessors or high-level DSPs perform both PCI master and target modes, and they may include a PCI arbiter. Listing 1 is a simple VHDL program. It performs an arbitration function by enabling access to the PCI bus depending on the predetermined priorities of each PCI device. The PCI-arbiter core interfaces with 33- and 66-MHz PCI systems, supports as many as six PCI-bus masters, supports “bus parking,” enables a pure rotational-arbitration scheme, supports bus latency and broken masters, and is a synthesizable VHDL source without FGPA- or PLD-library intellectual property.

Listing 1 is a simple VHDL program. It performs an arbitration function by enabling access to the PCI bus depending on the predetermined priorities of each PCI device. The PCI-arbiter core interfaces with 33- and 66-MHz PCI systems, supports as many as six PCI-bus masters, supports “bus parking,” enables a pure rotational-arbitration scheme, supports bus latency and broken masters, and is a synthesizable VHDL source without FGPA- or PLD-library intellectual property.

Figure 1 This PLD/FPGA-based PCI-bus arbiter grants bus requests based on a simple rotational-priority scheme.

The PCI bus supports more than one master device. If only one master requests the bus, that master immediately gets the grant. If several devices simultaneously require the use of the PCI bus to perform a data transfer, they assert their request signal, REQ_N, to the arbiter. The one with highest priority gets the GNT_N grant. After that, the one with the second highest priority has the highest priority, and so on. The PCI_RST assertion resets the arbiter’s priority-shift register to device 0.

The PCI bus has no pullups on the AD bus and C/BE lines. To avoid having these signals float for a long period, PCI designs must implement bus parking, meaning that a master device drives the AD bus and C/BE lines during bus-idle states. The arbiter selects which master will be park master. The arbiter asserts GNT_N of the park master, even though the park master did not assert REQ_N. The constant “Bus parker” in the VHDL code defines the park master. After a device has access to the PCI bus, this device must start the bus access within 16 PCI clock cycles. If this start-up does not happen, the device loses the bus grant, and the device with the next highest priority gets the bus. To check bus latency, the arbiter must check the signals FRAME_N and IRDY_N. The PCI-arbiter core
fits into any PLD or FPGA and consumes few resources.

Also see:

- VHDL code implements 50%-duty-cycle divider
- Design Ideas Submission Guide