Implementing Multicast Using DMA in a PCIe Switch

White Paper

Version 1.0

January 2009

Website: www.plxtech.com
Technical Support: www.plxtech.com/support

1 Introduction

The PCI standard was designed to be a bus based protocol shared amongst several devices. Bus based protocols have inherent broadcast built-in where data on the bus is seen by all the devices. In the case of PCI, broadcast can be implemented when one initiator targets a receiver while other receivers are listening in silent mode. This subset of multicast could be implemented using the Special Cycle command defined in the PCI protocol.

Figure 1 illustrates the broadcast mechanism implemented in the PCI bus. The flow process is as follows:

- the PCI Bus Master starts a transaction with the assertion of FRAME#
- The *Special Cycle* (Broadcast) command is issued in the C/BE[3:0]# lines
- All the slave devices accept the command and data from the Master

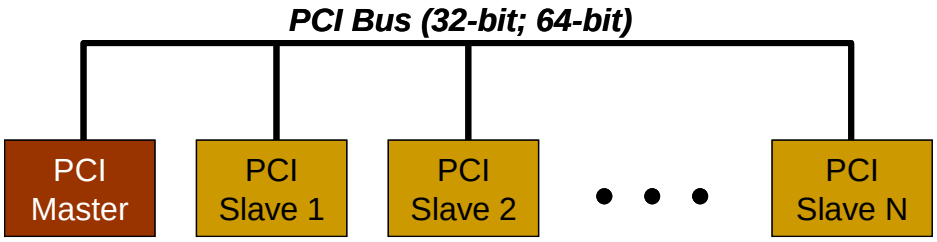

Figure 1. PCI with proprietary Broadcast scheme

This scheme used in the PCI Bus does not work with PCI Express (PCIe) because PCIe is a point-to-point protocol. The addition of multicast in PCIe devices is being addressed today within the PCI-SIG community in the form of an ECN. The ECN describes the function of multicast in a PCIe switch.

Because the multicast ECN is recent to the PCI-SIG, it will take some time before vendors implement multicast compliant devices. Furthermore, large switches will be first to support multicast while the smaller lane switches with multicast support will take longer to become available. However, the multicast function described in the ECN can be implemented today using DMA controllers available in PLX's industry-first PCIe switches with integrated DMA. Using the DMA controller in the PLX PCIe switch provides an efficient and attractive alternative to implementing multicast until the multicast function becomes natively available in a PCIe switch.

In this approach, software will construct a multicast descriptor ring. The multicast descriptor ring will have the same source address, which points to the same transmit buffer, and different destination address for each descriptor based on the destination port in the PCIe switch. The number of descriptors will determine how many devices are in that multicast group. Upon completion, the DMA engine can generate an interrupt upon completion of the multicast ring. This scheme is very flexible in nature compared to that defined by PCI-SIG. The driver updates the transmit buffer after the DMA transfer has been completed.

2 Implementing Multicast in a PCIe System

2.1 DMA Engine

A DMA (Direct Memory Access) engine is typically used to offload the data transfer from CPU's local memory out to devices connected to other side of interconnect. Typical DMA engines are network based DMA engines (E.g. NIC or HBA). The DMA engine is also used to transfer large amounts of data sent from one local memory to remote memory.

PCIe is the de-facto interconnect standard in boards and chips. Integrated DMA in a PCIe switch provides the capability to move large amount of data from local memory to devices attached to the switch; thus returning CPU cycles for time-critical applications. This capability for offloading the CPU plays a bigger role in embedded systems running RTOS.

The latest family of PCIe switches available from PLX (PEX 8619, PEX 8615 and PEX 8609) includes integrated DMA. The DMA engine supports four DMA channels which can be independently programmed and controlled.

As shown in Figure 2, the DMA engine appears as another function on the upstream port. This function has a TYPE 0 configuration header and follows the PCIe driver model. The driver for the DMA engine programs its DMA channels by writing to internal registers in the DMA function.

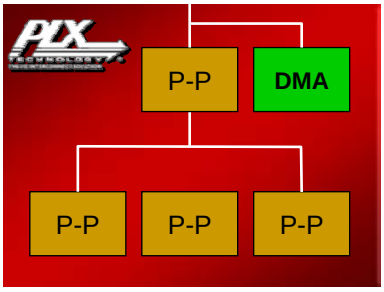

Figure 2. PCIe Switch with Integrated DMA

In the case of multicast, one channel is enough to support the multicast function. Figure 3 shows the descriptor ring format for each DMA channel. Details on the register locations can be found in the device databook. A DMA descriptor is made up of four Double-Words and consists of:

- Destination Address
- Source Address
- Transfer size
- Control

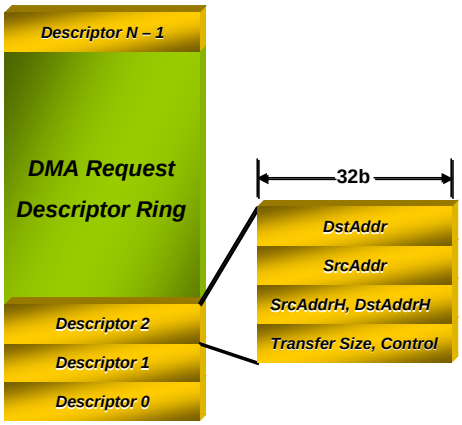

Figure 3. DMA Descriptor and Descriptor Ring

The control field in the descriptor provides the capability to the driver for generating an interrupt. Every descriptor will be treated as one packet or multiple packets. Each descriptor provides the bit map that is being looked at by driver and it inserts a number of descriptors equal to the number of copies to be made.

For implementing multicast, a descriptor ring is set up in host memory for each multicast group supported. The number of descriptors in the ring will determine the number of members in that particular multicast group. Each descriptor will hold the Source Address constant while the Destination Address will be set according to the address range for each of the downstream devices in that multicast group. One descriptor, at the end of the ring, is used as a fence to stop the DMA engine as well as keeping the DMA engine from wrapping around the descriptor ring. Once the descriptor ring is written, the CPU informs the DMA engine about the descriptor ring base address by writing to the internal DMA registers and finally writes to the start bit. The DMA engine reads the descriptor, finds the source pointer and then sends the data towards downstream port. Once it is done, it moves to the next descriptor that points to the same buffer in the local memory but with a destination address to the next down stream port. The DMA engine has the capability of generating an interrupt after each descriptor is done or after the entire descriptor ring is done. Upon completion of a descriptor ring (representing one multicast group) the DMA driver can program the bit so that an interrupt is generated to the host processor. At this point, the driver can also free up the source buffer.

1. CPU Programs Descriptors in RAM
2. CPU enables DMA
3. DMA reads Descriptors in RAM
4. DMA works on 1 descriptor at a time
    a. DMA reads source
    b. Completions arrive in switch
    c. Completions are converted to Writes
    d. DMA Writes to Destination
    e. Repeat for all members of multicast group
    f. Interrupt CPU after descriptor (optional)
    g. Start next descriptor
5. DMA Done – Multicast Group Done
6. Generate interrupt to CPU

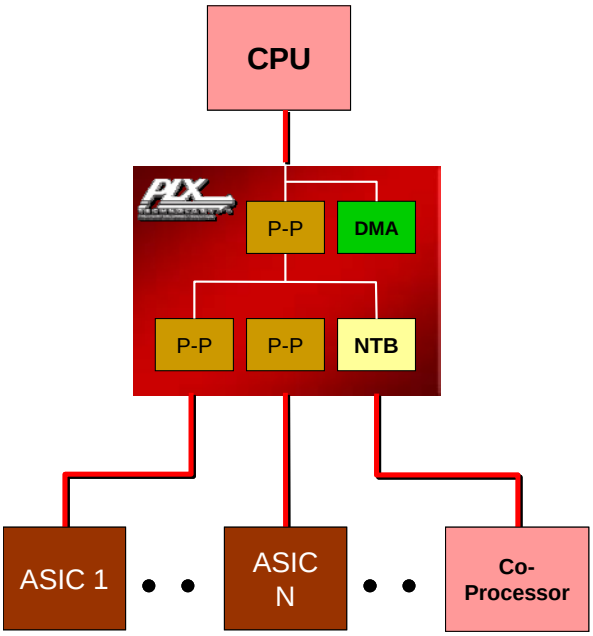

Figure 4. DMA Based Multicast Application

2.2 Multicast Driver

A multicast driver will be required regardless of whether multicast has been implemented in the switch or using the DMA engine as described above. When using the DMA engine to implement multicast, a software driver would implement an API which provides support for this function.

The following are supported by the multicast driver:

1. Configure descriptor base and ring size parameters in DMA engine
2. The descriptors are all zeros – ownership to driver

3. Multicast driver has registered for the interrupt
4. Multicast driver knows the destination addresses based on system wide address map.
5. When the packet needs to be multicasted, driver API call forwards the pointer and number of destination ports (Bit map).
6. Multicast driver writes descriptor at a time and change the ownership bit to DMA engine
7. Once DMA engine is done, it will write the status back into descriptor (Completion) and after the last descriptor, it will generate interrupt to the host.
8. Host can look at status and clean up the buffer.
9. In some cases, if there is an error in reaching one of the ASIC's, a status will be provided in the descriptor.
10. The multicast driver can write the descriptor again to make sure that ASIC receives the packet – or else one of the ASIC's will be out of sync (this cannot be checked with multicast implemented in switch).
11. Multicast driver waits for more packets to be multicasted.

3 Conclusion

The DMA engine in the PLX devices (PEX 8619, PEX 8615 and PEX 8609) can be used to perform the multicast function. The advantages for implementing a DMA based multicast scheme include support for unlimited number of multicast groups (each descriptor ring represents a multicast group), reliable multicast scheme (status for each copy made can be provided by DMA engine), multicast transfers are not limited to posted transactions and most importantly, DMA based multicast provides a PCIe multicast solution today.