System Interface Operations

6

The R10000 System interface provides a gateway between processor, with its associated secondary cache, and the remainder of the computer system.

For convenience, any device communicating with the processor through the System interface is referred to as the external agent.

6.1 Request and Response Cycles

The System interface supports the following request and response cycles:

- Processor requests are generated by the processor, when it requires a system resource.
- External responses are supplied by an external agent in response to a processor request.
- External requests are generated by an external agent when it requires a resource within the processor.
- Processor responses are supplied by the processor in response to an external request.

6.2 System Interface Frequencies

The System interface operates at SysClk frequency, supplied by the external agent. The internal processor clock, PClk, is derived from this same SysClk.

The SysClkDiv mode bits select a PClk to SysClk divisor of 1, 1.5, 2, 2.5, 3, 3.5, or 4, using the formula described in Chapter 7.

6.3 Register-to-Register Operation

The System interface is designed to operate in the following register-to-register fashion with the external agent:

- all System interface outputs are sourced directly from registers clocked on the rising edge of SysClk
- all System interface inputs directly feed registers that are clocked on the rising edge of SysClk

This allows the System interface to run at the highest possible clock frequency.

6.4 System Interface Signals

The R10000 System interface is composed of:

- 3 arbitration signals
- 2 flow-control input signals
- a bidirectional 12-bit command bus
- a bidirectional 64-bit multiplexed address/data bus
- a 3-bit state output bus
- a 5-bit response input bus

6.5 Master and Slave States

At any time, the System interface is either in *master* or *slave* state.

In master state, the processor drives the bidirectional System interface signals and is permitted to issue processor requests to the external agent.

In slave state, the processor tristates the bidirectional System interface signals and accepts external requests from the external agent.

6.6 Connecting to an External Agent

In a uni- or multiprocessor system using dedicated external agents, the System interface connects to a single external agent.

In a multiprocessor system using the cluster bus (see below), the system can connect up to four R10000 processors to an external agent. This external agent is referred to as the cluster coordinator.

6.7 Cluster Bus

In a multiprocessor system using the cluster bus, the cluster coordinator performs the cluster bus arbitration and data flow management. The arbitration scheme assures that either one of the processors or the cluster coordinator is master at any given time, while the remaining devices are slave.

A processor request issued by the master processor is observed as an external request by all slave R10000 processors, as shown in Figure 6-1. Similarly, a processor coherency data response issued by a master processor is observed as an external data response by the slave processors.

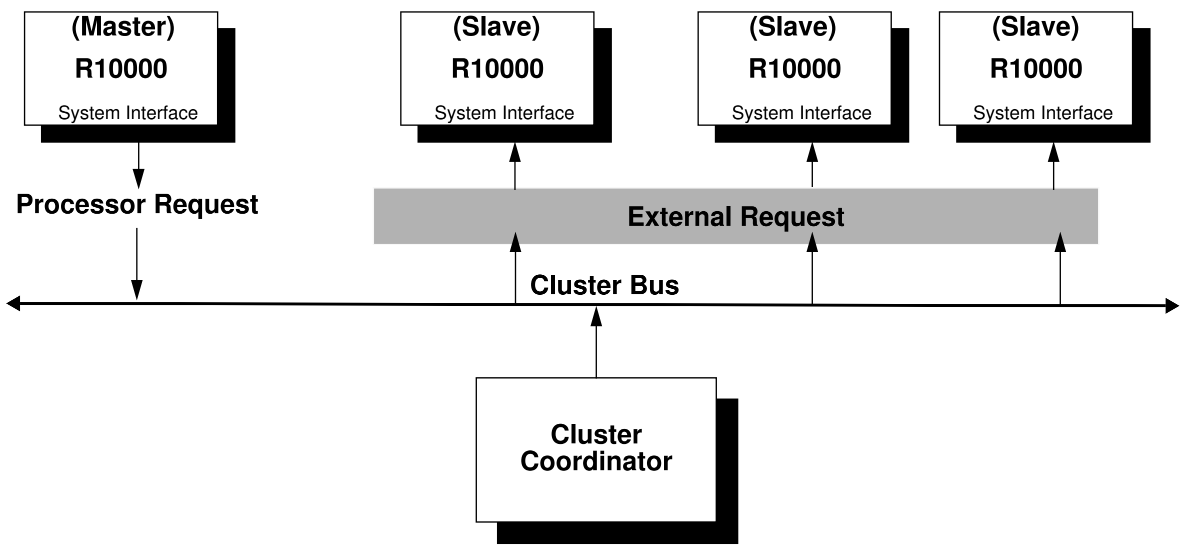

Figure 6-1 Processor Request Master/Slave Status

In a multiprocessor system using the cluster bus, a mode bit specifies whether processor coherent requests are to target the external agent only, or all processors and the external agent. This allows systems with efficient snoopy, duplicate tag, or directory-based coherency protocols to be created.

6.8 System Interface Connections

The major System interface connections required for various system configurations are presented in this section.

Uniprocessor System

Figure 6-2 shows the major System interface connections required for a typical uniprocessor system.

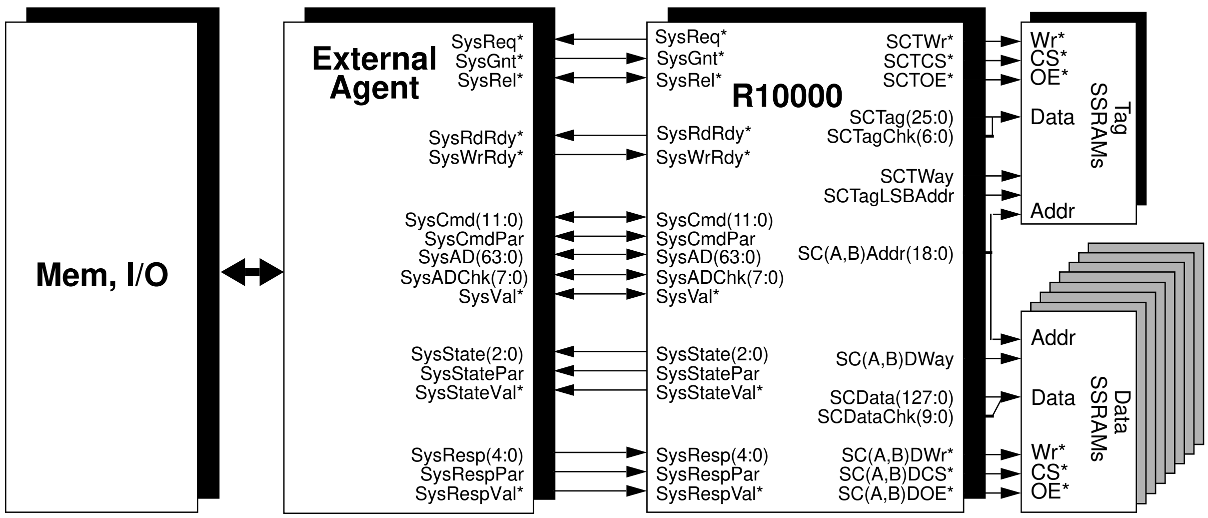

Figure 6-2 System Interface Connections for Uniprocessor System

Multiprocessor System Using Dedicated External Agents

Figure 6-3 shows the major System interface connections required for a typical multiprocessor system using dedicated external agents.

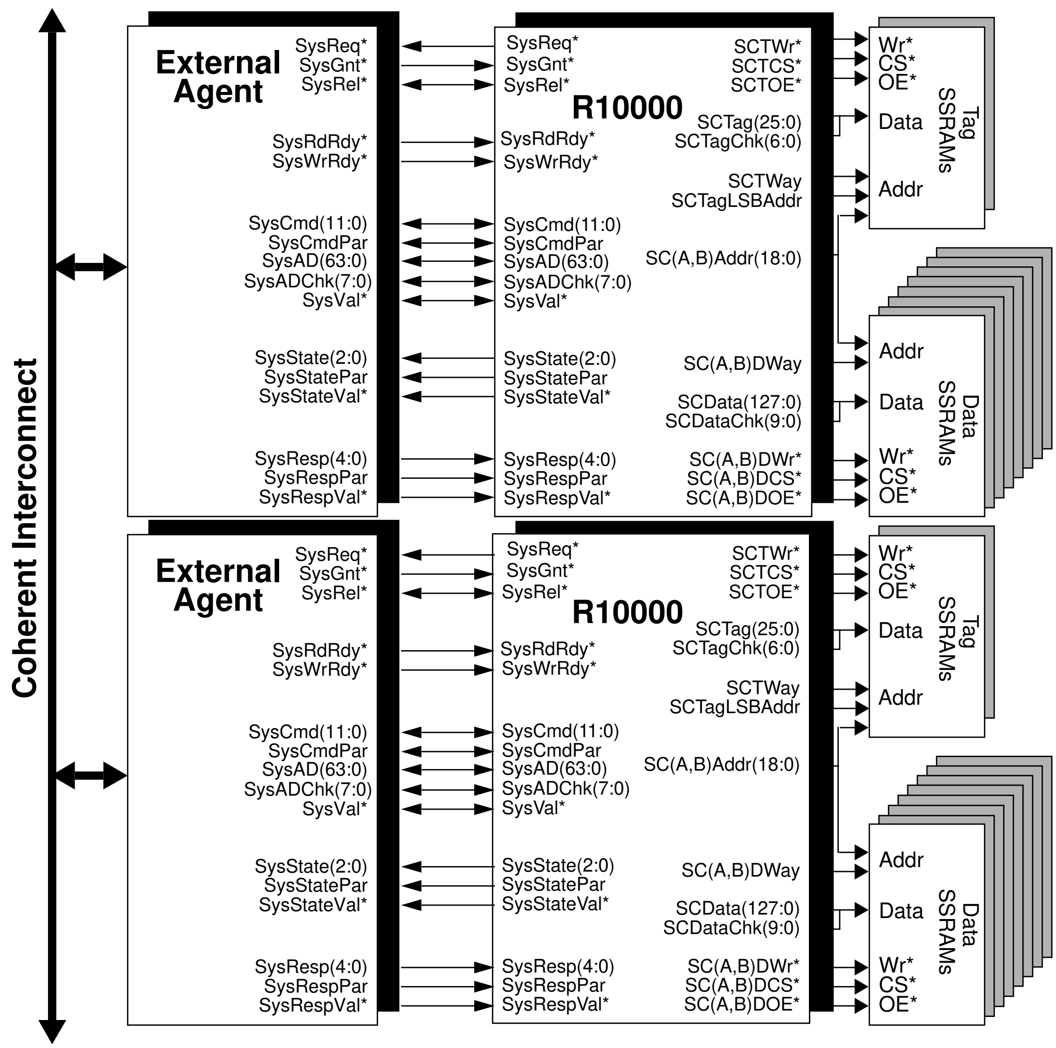

Figure 6-3 System Interface Connections for Multiprocessor using Dedicated External Agents

Multiprocessor System Using the Cluster Bus

Figure 6-4 presents the major System interface connections required for a typical multiprocessor system using the cluster bus.

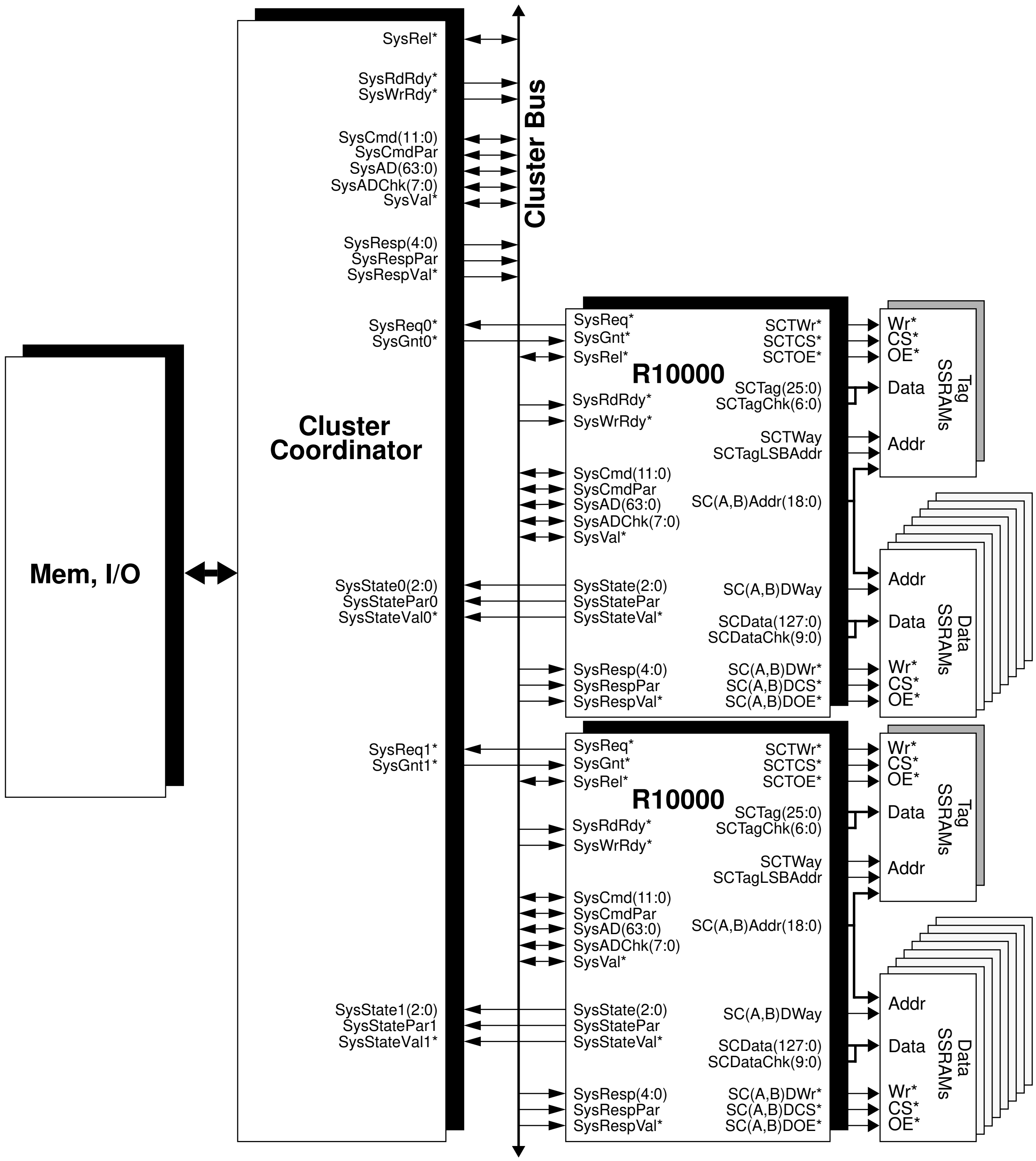

Figure 6-4 System Interface Connections for Multiprocessor Using the Cluster Bus

6.9 System Interface Requests and Responses

The System interface supports the following:

- processor request
- external response
- external request
- processor response

The following sections describe these request and response types, and their operations.

Processor Requests

Processor requests are generated by the processor when it requires a system resource. The following processor requests are supported:

- coherent block read shared request
- coherent block read exclusive request
- noncoherent block read request
- double/single/partial-word read request
- block write request
- double/single/partial-word write request
- upgrade request
- eliminate request

Processor write and eliminate requests do not require or expect a response by the external agent. However, if an external agent detects an error in a processor write or eliminate request, it may use an interrupt to signal the processor. It is not possible to generate precise exceptions for processor write and eliminate requests for which an external agent detects an error.

Processor read and upgrade requests require some type of response by the external agent.

External Responses

External responses are supplied by an external agent or another processor in response to a processor request. The following external responses are supported:

- block data response
- double/single/partial-word data response
- completion response

External Requests

External requests are generated by an external agent when it requires a resource within the processor. The following external requests are supported:

- intervention shared request
- intervention exclusive request
- allocate request number request
- invalidate request
- interrupt request

External intervention and invalidate requests require some type of response by the processor.

Processor Responses

Processor responses are supplied by the processor in response to an external request. The following processor responses are supported:

- coherency state response
- coherency data response

Outstanding Requests and Request Numbers

The processor allows requests and corresponding responses to be split transactions, which enables additional processor and external requests to be issued while waiting for a prior response. The System interface supports a request number field to link requests with their corresponding responses, so responses can be returned out of order.

The processor allows a maximum of eight outstanding requests on the System interface through a 3-bit request number. These outstanding requests may be composed of any mix of processor and external requests.

An individual processor (as opposed to the System interface, above) supports a maximum of four outstanding processor requests at any given time.

Request and Response Relationship

The relationship between processor and external requests, and their acceptable responses, is presented in Table 6-1. The data in this table is given with respect to a single processor, in either a uni- or multiprocessor system (independent of cluster/non-cluster configuration).

Table 6-1 Request and Response Relationship

Request	Acceptable Response Sequences
Processor block read request	External NACK or ERR completion response
	0 or more external block data responses followed by a final external block data response with a coincidental or subsequent external ACK, NACK, or ERR completion response
Processor double/single/partial-word read request	External NACK or ERR completion response
	0 or more external double/single/partial-word data responses followed by a final external double/single/partial-word data response with a coincidental or subsequent external ACK, NACK, or ERR completion response
Processor block write request	None
Processor double/single/partial-word write request	None
Processor upgrade request	External ACK, NACK, or ERR completion response
	0 or more external block data responses followed by a final external block data response with a coincidental or subsequent external ACK, NACK, or ERR completion response
Processor eliminate request	None
External intervention request	Processor coherency state response followed by processor coherency data response (if *DirtyExclusive)* with a coincidental or subsequent external ACK, NACK, or ERR completion response*
External allocate request number request	External ACK, NACK, or ERR completion response*
External invalidate request	Processor coherency state response followed by external ACK, NACK, or ERR completion response*
External interrupt request	None

* External completion response is required to free the request number.

6.10 System Interface Buffers

The processor contains the following five buffers to enhance the performance of the System interface and to simplify the system design:

- incoming buffer
- cached request buffer
- outgoing buffer
- uncached buffer
- cluster request buffer

These buffers are described in the following sections.

Incoming Buffer

The System interface contains an incoming buffer for external block and double/single/partial-word data responses. The four 32-word entries of the incoming buffer correspond to the four possible outstanding processor requests. Block data in each entry of the incoming buffer is stored in subblock order, beginning with a quadword-aligned address.

The incoming buffer eliminates the need for the processor to flow-control the external agent that is providing the external data responses. Regardless of the cache bandwidth or internal resource availability, the external agent may supply external data response data for all outstanding read and upgrade requests at the maximum System interface data rate.

The external agent may issue any number of external data responses for a particular request number before issuing a corresponding external completion response. An external data response remains in the incoming buffer until a corresponding external completion response is received. A former buffered external data response for a particular request number is over-written by a subsequent external data response for the same request number.

An external ACK completion response frees buffered data to be forwarded to the caches and other internal resources while an external NACK or ERR completion response purges any corresponding buffered data. For minimum latency, the external agent should issue an external ACK completion response coincident with the first doubleword of an external data response.

External coherency requests that target blocks residing in the incoming buffer are stalled until the incoming buffer data is forwarded to the secondary cache, and the instruction that caused the secondary miss is satisfied.

Each doubleword of the incoming buffer has an Uncorrectable Error flag. When an external data response provides a doubleword, the processor asserts the corresponding incoming buffer Uncorrectable Error flag if the data quality indicator, SysCmd[5] , is asserted, or if an uncorrectable ECC error is encountered on the system address/data bus and the ECC check indication on SysCmd[0] is asserted.

When the processor forwards block data from an incoming buffer entry after receiving an external ACK completion response, the associated incoming buffer Uncorrectable Error flags are checked, and if any are asserted, a single Cache Error exception is posted. When the processor forwards double/single/partial-word data from an incoming buffer entry after receiving an external ACK completion response, the associated incoming buffer Uncorrectable Error flag is checked, and if asserted, a Bus Error exception is posted.

Cached Request Buffer

The System interface contains a four-entry cached request buffer. This buffer holds the status of the four possible outstanding processor cached requests, including processor block read and upgrade requests. The relative order of the requests is maintained in the cached request buffer.

External coherency requests probe the cached request buffer to detect conflict conditions.

Outgoing Buffer

The System interface contains a five-entry outgoing buffer to provide buffering for the following:

- *DirtyExclusive* blocks that are cast out of the secondary cache because of a block replacement
- various CACHE instructions
- an external intervention request.

Four 32-word *typical* entries are associated with the four possible outstanding processor cached requests allowed by the processor. One 32-word *special* entry is reserved for external intervention requests only. The data is stored in each entry of the outgoing buffer in sequential order, beginning with a secondary cache block-aligned address.

An instruction or data access that misses in the secondary cache but targets an entry in the outgoing buffer is stalled until the outgoing buffer entry is issued as a processor block write request or coherency data response to the System interface bus.

External coherency requests probe the four typical outgoing buffer entries, with the following results:

- If an external intervention request hits a typical entry, that entry is converted from a processor block write request to a processor coherency data response.
- If an external invalidate request hits a typical outgoing buffer entry, that entry is deleted.
- If an external intervention request does not hit a typical outgoing buffer entry, but hits a *DirtyExclusive* block in the secondary cache, the special outgoing buffer entry is used to buffer the processor coherency data response.

A typical outgoing buffer entry containing a block write is ready for issue to the System interface bus when the first quadword is received from the secondary cache. The processor allows data to stream from the secondary cache to the System interface bus through the outgoing buffer.

An outgoing buffer entry containing a coherency data response for a secondary cache block-aligned external intervention request is ready for issue to the System interface bus when the first quadword is received from the secondary cache. In this case, the processor allows data to stream from the secondary cache to the System interface bus through the outgoing buffer. An outgoing buffer entry containing a coherency data response for a non-secondary cache block-aligned external intervention request is ready for issue to the System interface bus when all quadwords are received from the secondary cache.

Each quadword of the outgoing buffer maintains an Uncorrectable Error flag. If an uncorrectable error is encountered while a block is being cast out of the secondary cache, the associated outgoing buffer quadword Uncorrectable Error flag is asserted. When the processor empties an outgoing buffer entry by issuing a processor block write or coherency data response, the outgoing buffer Uncorrectable Error flags are reflected by the data quality indication on SysCmd[5] .

Uncached Buffer

The System interface contains an uncached buffer to provide buffering for uncached and uncached accelerated load and store operations. All operations retain program order within the uncached buffer.

The uncached buffer is organized as a 4-entry FIFO followed by a 2-entry gatherer. Each gathered entry has a capacity of 16 or 32 words, as specified by the SCBlkSize mode bit.

The uncached buffer begins gathering when an uncached accelerated double or singleword block-aligned store is executed. Gathering continues if the subsequent uncached operation executed is an uncached accelerated double or singleword store to a sequential or identical address. Once a second uncached accelerated store is gathered, the gathering mode is determined to be sequential or identical. Gathering continues until one of the following conditions occurs:

- a complete block is gathered
- an uncached or uncached accelerated load is executed
- an uncached or uncached accelerated partial-word store is executed
- an uncached store is executed
- a change in the current gathering mode is observed
- a change in the uncached attribute is observed

When gathering terminates, the data is ready for issue to the System interface bus. A processor uncached accelerated block write request is used to issue a completely gathered uncached accelerated block. One or more disjoint processor uncached accelerated double or singleword write requests are used to issue an incompletely gathered uncached accelerated block.

When gathering in an identical mode, uncached accelerated double or singleword stores may be freely mixed. The uncached buffer packs the associated data into the gatherer. When gathering in sequential mode, uncached accelerated singleword stores must occur in pairs, to prevent an address error exception. For instance, SW, SW, SD, SW, SW is legal. SD, SW, SD, is not.

External coherency requests have no effect on the uncached buffer.

CACHE instructions have no effect on the uncached buffer. SYNC instructions are prevented from graduating if an uncached store resides in the uncached buffer.

Cluster Request Buffer

The System interface contains an 8-entry cluster request buffer. This buffer maintains the status of the eight possible outstanding requests on the System interface. When the System interface is in master state, and it issues the address cycle of processor read or upgrade request, the processor places an entry into the cluster request buffer. When the System interface is in slave state, and an external agent issues an external coherency or allocate request number request, it places an entry into the cluster request buffer.

Once an entry is placed into the cluster request buffer, the associated request number transitions from *free* to *busy*. An entry remains busy until the processor receives an external completion response. Processor requests that are ready to be issued to the System interface bus probe the cluster request buffer to detect conflict conditions.

6.11 System Interface Flow Control

The System interface supports a maximum *request rate* of one request per SysClk cycle, and a maximum *data rate* of one doubleword per SysClk cycle.

Various flow control mechanisms are provided to limit these rates, as described below.

Processor Write and Eliminate Request Flow Control

The processor can only issue a processor write or eliminate request if:

- the System interface is in master state
- SysWrRdy* was asserted two SysClk cycles previously

Processor Read and Upgrade Request Flow Control

The processor can only issue a processor read or upgrade request if:

- the System interface is in master state
- SysRdRdy* was asserted two SysClk cycles previously
- the maximum number of outstanding processor requests specified by the PrcReqMax mode bits is not exceeded
- there is a free request number

Processor Coherency Data Response Flow Control

The processor can only issue a processor coherency data response if:

- the System interface is in master state
- SysWrRdy* was asserted two SysClk cycles previously

External Request Flow Control

When the System interface is in *Slave* state, it is capable of accepting external requests. An external agent may issue external requests in adjacent SysClk cycles.

External Data Response Flow Control

Since the processor has an incoming buffer, an external agent may supply external data response data in adjacent SysClk cycles, without regard to cache bandwidth or internal resource availability.

6.12 System Interface Block Data Ordering

During block data transfers on the System interface SysAD[63:0] bus, even doublewords (Dat0, Dat2,...) always correspond to SCData[127:64], and odd doublewords (Dat1, Dat3,...) always correspond to SCData[63:0].

External Block Data Responses

During the address cycle of processor block read and upgrade requests, the processor specifies a quadword-aligned address. The processor expects the external block data response to be supplied in a subblock order sequence, beginning at the specified quadword-aligned address.

Processor Coherency Data Responses

The address of external intervention requests are internally aligned by the processor to a quadword address. If the processor determines that it must issue a processor coherency data response, it supplies the data in a subblock order sequence beginning at the quadword-aligned address specified by the corresponding external coherency request.

Processor Block Write Requests

During the address cycle of processor block write requests, the processor specifies a cache block-aligned address. During the subsequent data cycles for typical processor block write requests, the processor supplies the data in sequence, beginning with the secondary cache block-aligned address.

6.13 System Interface Bus Encoding

This section presents the encoding of the following four System interface buses:

- SysCmd[11:0]
- SysAD[63:0]
- SysState[2:0]
- SysResp[4:0]

SysCmd[11:0] Encoding

This section describes address and data cycle encodings for the system command bus, SysCmd[11:0] .

SysCmd[11] Encoding

When SysVal* is asserted, SysCmd[11] indicates whether the SysAD[63:0] bus represents an address or a data cycle, as shown in Table 6-2.

Table 6-2 Encoding of SysCmd[11]

SysCmd[11]	Data/Address Cycle Indication
0	SysAD[63:0] address cycle
1	SysAD[63:0] data cycle

SysCmd[10:0] Address Cycle Encoding

During the address cycle of processor read and upgrade requests, SysCmd[10:8] contain the request number, as shown in Table 6-3. The request number provides a mechanism to associate an external response with the corresponding processor request.

Table 6-3 Encoding of SysCmd[10:8] *for Processor Read and Upgrade Requests*

SysCmd[10:8]	Request Number

During the address cycle of processor requests, SysCmd[7:5] contain the command, as shown in Table 6-4.

Table 6-4 Encoding of SysCmd[7:5] *for Processor Requests*

SysCmd[7:5]	Command
0	Coherent block read shared
1	Coherent block read exclusive
2	Noncoherent block read
3	Double/single/partial-word read
4	Block write
5	Double/single/partial-word write
6	Upgrade
7	Special

During the address cycle of processor read requests, SysCmd[4:3] contain the read cause indication, as shown in Table 6-5. This information is useful in handling the associated external response.

Table 6-5 Encoding of SysCmd[4:3] *for Processor Read Requests*

SysCmd[4:3]	Read Cause Indication
0	Instruction access
1	Data typical access
2	Data LL/LLD access
3	Data prefetch access

During the address cycle of processor write requests, SysCmd[4:3] contain the write cause indication, as shown in Table 6-6. This information is useful in handling the associated write data.

Table 6-6 Encoding of SysCmd[4:3] *for Processor Write Requests*

SysCmd[4:3]	Write Cause Indication
0	Reserved
1	Data typical access
2	Data uncached accelerated sequential access
3	Data uncached accelerated identical access

During the address cycle of processor upgrade requests, SysCmd[4:3] contain the upgrade cause indication, as shown in Table 6-7. This information useful in handling the associated external response.

Table 6-7 Encoding of SysCmd[4:3] *for Processor Upgrade Requests*

SysCmd[4:3]	Upgrade Cause Indication
0	Reserved
1	Data typical access
2	Data SC/SCD access
3	Data prefetch access

During the address cycle of processor special requests, SysCmd[4:3] contain the processor special cause indication, as shown in Table 6-8. This information differentiates between the various processor special requests.

Table 6-8 Encoding of SysCmd[4:3] *for Processor Special Requests*

SysCmd[4:3]	Special Cause Indication
0	Reserved
1	Eliminate
2	Reserved
3	Reserved

During the address cycle of processor block read, typical block write, upgrade, and eliminate requests, SysCmd[2:1] contain the secondary cache block former state, as shown in Table 6-9. This information may be useful for system designs implementing a duplicate tag or a directory-based coherency protocol.

Table 6-9 Encoding of SysCmd[2:1] *for Processor Block Read/Write, Upgrade, Eliminate Requests*

SysCmd[2:1]	Secondary Cache Block Former State
0	*Invalid*
1	*Shared*
2	*CleanExclusive*
3	*DirtyExclusive*

During the address cycle of processor double/single/partial-word read and write requests, SysCmd[2:0] contain the data size indication, as shown in Table 6-10.

Table 6-10 Encoding of SysCmd[2:0] *for Processor Double/Single/Partial-Word Read/Write Requests*

SysCmd[2:0]	Data Size Indication
0	One byte valid (Byte)
1	Two bytes valid (Halfword)
2	Three bytes valid (Tribyte)
3	Four bytes valid (Word)
4	Five bytes valid (Quintibyte)
5	Six bytes valid (Sextibyte)
6	Seven bytes valid (Septibyte)
7	Eight bytes valid (Doubleword)

During the address cycle of external intervention and invalidate requests, SysCmd[10:8] contain the request number, as shown in Table 6-11. The request number provides a mechanism to associate a potential processor coherency data response with the corresponding external coherency request.

Table 6-11 Encoding of SysCmd[10:8] *for External Intervention and Invalidate Requests*

SysCmd[10:8]	Request Number

During the address cycle of external requests, SysCmd[7:5] contain the command, as shown in Table 6-12.

Table 6-12 Encoding of SysCmd[7:5] *for External Requests*

SysCmd[7:5]	Command
0	Intervention shared
1	Intervention exclusive
2	Allocate request number
3	Allocate request number
4	NOP
5	NOP
6	Invalidate
7	Special

During the address cycle of external special requests, SysCmd[4:3] contain the external special cause indication, as shown in Table 6-13. This information is used to differentiate between the various external special requests.

Table 6-13 Encoding of SysCmd[4:3] *for External Special Requests*

SysCmd[4:3]	Special Cause Indication
0	Reserved
1	NOP
2	Interrupt
3	Reserved

During external address cycles, SysCmd[0] specifies whether ECC checking and correcting is to be performed for the SysAD[63:0] bus, as shown in Table 6-14. During the address cycle of processor block read, block write, upgrade, and eliminate requests, the processor asserts SysCmd[0] . Consequently, in a multiprocessor system using the cluster bus, ECC checking and correcting is enabled for external coherency requests resulting from processor coherent block read and upgrade requests.

Table 6-14 Encoding of SysCmd[0] *for External Address Cycles*

SysCmd[0]	ECC check indication
0	ECC checking and correcting disable
1	ECC checking and correcting enable

SysCmd[10:0] Data Cycle Encoding

During the data cycles of an external data response or a processor coherency data response, SysCmd[10:8] contain the request number associated with the original request, as shown in Table 6-15.

Table 6-15 Encoding of SysCmd[10:8] *for Data Responses*

SysCmd[10:8]	Request Number

During data cycles, SysCmd[5] indicates the data quality, as shown in Table 6-16.

Table 6-16 Encoding of SysCmd[5] *for Data Cycles*

SysCmd[5]	Data quality indication
0	Data is good
1	Data is bad

During data cycles, SysCmd[4:3] indicate the data type, as shown in Table 6-17. Processor block write and double/single/partial-word write requests use request data and request last data type indications. External data and processor coherency data responses use response data and response last data type indications.

Table 6-17 Encoding of SysCmd[4:3] *for Data Cycles*

SysCmd[4:3]	Data type Indication
0	Request data
1	Response data
2	Request last
3	Response last

During data cycles of an external block data response or processor coherency data response, SysCmd[2:1] contain the state of the cache block, as shown in Table 6-18.

Table 6-18 Encoding of SysCmd[2:1] *for Block Data Responses*

SysCmd[2:1]	Cache Block State
0	Reserved
1	*Shared*
2	*CleanExclusive*
3	*DirtyExclusive*

During data cycles, SysCmd[0] specifies whether ECC checking and correcting is to be performed for the SysAD[63:0] bus, as shown in Table 6-19. During processor data cycles, the processor asserts SysCmd[0]. Consequently, in a multiprocessor system using the cluster bus, ECC checking and correcting will be enabled for external block data responses resulting from processor coherency data responses.

Table 6-19 Encoding of SysCmd[0] *for External Data Cycles*

SysCmd[0]	ECC check indication
0	ECC checking and correcting disable
1	ECC checking and correcting enable

SysCmd[11:0] Map

Table 6-20 presents a map for the SysCmd[11:0] bus.

Table 6-20 SysCmd[11:0] Map

<table>
<tr><th rowspan="2">Cycle Type</th><th rowspan="2" colspan="2">Command</th><th colspan="12">SysCmd[11:0] Bit</th></tr>
<tr><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td rowspan="11">Processor address cycles</td><td colspan="2">Coherent block read shared</td><td rowspan="11">0</td><td colspan="3" rowspan="4">Request number</td><td>0</td><td>0</td><td>0</td><td colspan="2" rowspan="4">Read cause</td><td colspan="2" rowspan="3">Block state</td><td rowspan="3">1</td></tr>
<tr><td colspan="2">Coherent block read exclusive</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td colspan="2">Noncoherent block read</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Double/single/partial-word read</td><td>0</td><td>1</td><td>1</td><td colspan="3">Data size</td></tr>
<tr><td colspan="2">Block write</td><td colspan="3" rowspan="2">0</td><td>1</td><td>0</td><td>0</td><td colspan="2" rowspan="2">Write cause</td><td colspan="2">Block state</td><td>1</td></tr>
<tr><td colspan="2">Double/single/partial-word write</td><td>1</td><td>0</td><td>1</td><td colspan="3">Data size</td></tr>
<tr><td colspan="2">Upgrade</td><td colspan="3">Request number</td><td>1</td><td>1</td><td>0</td><td colspan="2">Upgrade cause</td><td colspan="2">Block state</td><td>1</td></tr>
<tr><td rowspan="4">Special</td><td>Reserved</td><td colspan="3">Reserved</td><td rowspan="4">1</td><td rowspan="4">1</td><td rowspan="4">1</td><td>0</td><td>0</td><td colspan="3">Reserved</td></tr>
<tr><td>Eliminate</td><td colspan="3">0</td><td>0</td><td>1</td><td colspan="2">Block state</td><td>1</td></tr>
<tr><td rowspan="2">Reserved</td><td colspan="3" rowspan="2">Reserved</td><td>1</td><td>0</td><td colspan="3" rowspan="2">Reserved</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td rowspan="3">Processor data cycles</td><td colspan="2">Double/single/partial-word write</td><td rowspan="3">1</td><td colspan="3" rowspan="2">0</td><td colspan="2" rowspan="3">0</td><td>0</td><td colspan="2" rowspan="3">Data type</td><td colspan="2">0</td><td rowspan="3">1</td></tr>
<tr><td colspan="2">Block write</td><td rowspan="2">Data quality</td><td colspan="2" rowspan="2">Block state</td></tr>
<tr><td colspan="2">Coherency data response</td><td colspan="3">Request number</td></tr>
<tr><td rowspan="11">External address cycles</td><td colspan="2">Intervention shared</td><td rowspan="11">0</td><td colspan="3" rowspan="4">Request number</td><td>0</td><td>0</td><td>0</td><td colspan="4" rowspan="7">X</td><td rowspan="2">ECC</td></tr>
<tr><td colspan="2">Intervention exclusive</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td colspan="2" rowspan="2">Allocate request number</td><td>0</td><td>1</td><td>0</td><td rowspan="4">X</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td colspan="2" rowspan="2">NOP</td><td colspan="3" rowspan="2">X</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td colspan="2">Invalidate</td><td colspan="3">Request number</td><td>1</td><td>1</td><td>0</td><td>ECC</td></tr>
<tr><td rowspan="4">Special</td><td rowspan="2">NOP</td><td colspan="3" rowspan="4">X</td><td rowspan="4">1</td><td rowspan="4">1</td><td rowspan="4">1</td><td>0</td><td>0</td><td colspan="2" rowspan="4">X</td><td rowspan="2">X</td></tr>
<tr><td>0</td><td>1</td></tr>
<tr><td>Interrupt</td><td>1</td><td>0</td><td>ECC</td></tr>
<tr><td>NOP</td><td>1</td><td>1</td><td>X</td></tr>
<tr><td rowspan="2">External data cycles</td><td colspan="2">Block data response</td><td rowspan="2">1</td><td colspan="3" rowspan="2">Request number</td><td colspan="2" rowspan="2">X</td><td rowspan="2">Data quality</td><td colspan="2" rowspan="2">Data type</td><td colspan="2">Block state</td><td rowspan="2">ECC</td></tr>
<tr><td colspan="2">Double/single/partial-word data response</td><td colspan="2">X</td></tr>
</table>

SysAD[63:0] Encoding

This section describes the system address/data bus encoding.

SysAD[63:0] Address Cycle Encoding

Table 6-21 presents the encoding of the SysAD[63:0] bus for address cycles.

Table 6-21 Encoding of SysAD[63:0] *for Address Cycles*

SysAD[63:60]	Target Indication
SysAD[63]	Target processor with DevNum = 3
SysAD[62]	Target processor with DevNum = 2
SysAD[61]	Target processor with DevNum = 1
SysAD[60]	Target processor with DevNum = 0
SysAD[59:58]	Uncached attribute
SysAD[57]	Secondary cache block way indication
SysAD[56:40]	Reserved
SysAD[39:0]	Physical address

SysAD[63:60]

During the address cycle of processor noncoherent block read, double/single/partial-word read, block write, double/single/partial-word write, and eliminate requests, the processor always drives a target indication of 0 on SysAD[63:60]. This indicates that the request targets the external agent only. When the CohPrcReqTar mode bit is negated, during the address cycle of processor coherent block read and upgrade requests, the processor also drives a target indication of 0 on SysAD[63:60]. However, when the CohPrcReqTar mode bit is asserted, during the address cycle of processor coherent block read and upgrade requests, the processor drives a target indication of 0xF on SysAD[63:60]. This indicates that the request targets all processors, together with the external agent, on the cluster bus. In multiprocessor systems using the cluster bus, the CohPrcReqTar mode bit is asserted for a snoopy-based coherency protocol, and negated for a duplicate tag or directory-based coherency protocol.

When the processor is in slave state, an external agent uses the target indication field to specify which processors are targets of an external request.

SysAD[59:58] Uncached Attribute

During the address cycle of processor double/single/partial-word read and write requests and during the address cycle of processor *Uncached accelerated* block write requests, the processor drives the uncached attribute onto SysAD[59:58]. See the section titled, Support for Uncached Attribute, in this chapter for more information.

SysAD[57]

During the address cycle of processor block read, typical block write, upgrade, and eliminate requests, SysAD[57] contains the secondary cache block way indication. This information may be useful for system designs implementing a duplicate tag or a directory-based coherency protocol.

SysAD[56:40]

When processor is in master state, it drives SysAD[56:40] to zero during address cycles.

SysAD[39:0]

During the address cycle of processor and external requests, SysAD[39:0] contain the physical address.

Table 6-22 presents the processor request address cycle address alignment.

Table 6-22 Processor Request Address Cycle Alignment

Processor Request Type	Address Alignment	Address Bits Which Are Driven to 0
Block read	Quadword	3:0
Doubleword read/write	Doubleword	2:0
Singleword read/write	Singleword	1:0
Halfword read/write	Halfword	0
Byte, tribyte, quintibyte, sextibyte, septibyte read/write	Byte	-
Block write	Secondary cache block	5:0 (SCBlkSize = 0) 6:0 (SCBlkSize = 1)
Upgrade	Quadword	3:0
Eliminate	Secondary cache block	5:0 (SCBlkSize = 0) 6:0 (SCBlkSize = 1)

Table 6-23 presents the external coherency request address cycle address alignment.

Table 6-23 External Coherency Request Address Cycle Alignment

External Request Type	Address Alignment	Address Bits Which Are Ignored
Intervention	Quadword	3:0
Invalidate	Secondary cache block	5:0 (SCBlkSize = 0) 6:0 (SCBlkSize = 1)

SysAD[63:0] Data Cycle Encoding

During System interface data cycles, when less than a doubleword is transferred on the SysAD[63:0] bus, the valid byte lanes depend on the request address and the MemEnd mode bit.

For example, consider the data cycle for a byte request whose address modulo 8 is 1. When MemEnd is negated (little endian), the SysAD[15:8] byte lane is valid. When MemEnd is asserted (big endian), the SysAD[55:48] byte lane is valid.

SysState[2:0] Encoding

The processor provides a processor coherency state response by driving the targeted secondary cache block tag quality indication on SysState[2], driving the targeted secondary cache block former state on SysState[1:0] and asserting SysStateVal* for one SysClk cycle. Table 6-24 presents the encoding of the SysState[2:0] bus when SysStateVal* is asserted.

Table 6-24 Encoding of SysState[2:0] *when* SysStateVal* *Asserted*

SysState[2]	Secondary cache block tag quality indication
0	Tag is good
1	Tag is bad
SysState[1:0]	Secondary cache block former state
0	*Invalid*
1	*Shared*
2	*CleanExclusive*
3	*DirtyExclusive*

When SysStateVal* is negated, SysState[0] indicates if a processor coherency data response is ready for issue. Table 6-25 presents the encoding of the SysState[2:0] bus when SysStateVal* is negated.

Table 6-25 Encoding of SysState[2:0] *When SysStateVal* Negated*

SysState[2:1]	Reserved
SysState[0]	Processor coherency data response indication
0	Not ready for issue
1	Ready for issue

SysResp[4:0] Encoding

An external agent issues an external completion response by driving the request number associated with the corresponding request on SysResp[4:2], driving the completion indication on SysResp[1:0], and asserting SysRespVal* for one SysClk cycle. Table 6-26 presents the encoding of the SysResp[4:0] bus.

Table 6-26 Encoding of SysResp[4:0]

SysResp[4:2]	Request number
SysResp[1:0]	Completion indication
0	Acknowledge (ACK)
1	Error (ERR)
2	Negative acknowledge (NACK)
3	Reserved

6.14 Interrupts

The processor supports five hardware, two software, one timer, and one nonmaskable interrupt. The Interrupt exception is described in Chapter 17.

Hardware Interrupts

Five hardware interrupts are accessible to an external agent via external interrupt requests.

An external interrupt request consists of a single address cycle on the System interface. During the address cycle, SysAD[63:60] specify the target indication, which allows an external agent to define the target processors of the external interrupt request. If a processor determines it is an external interrupt request target, SysAD[20:16] are the write enables for the five individual *Interrupt* register bits and SysAD[4:0] are the values to be written into these bits, as shown in Figure 6-5. This allows any subset of the *Interrupt* register bits to be set or cleared with a single external interrupt request.

The *Interrupt* register is an architecturally transparent, level-sensitive register that is directly readable as bits 14:10 of the *Cause* register. Since it is level-sensitive, an interrupt bit must remain asserted until the interrupt is taken, at which time the interrupt handler must cause a second external interrupt request to clear the bit.

The processor clears the *Interrupt* register during any of the reset sequences.

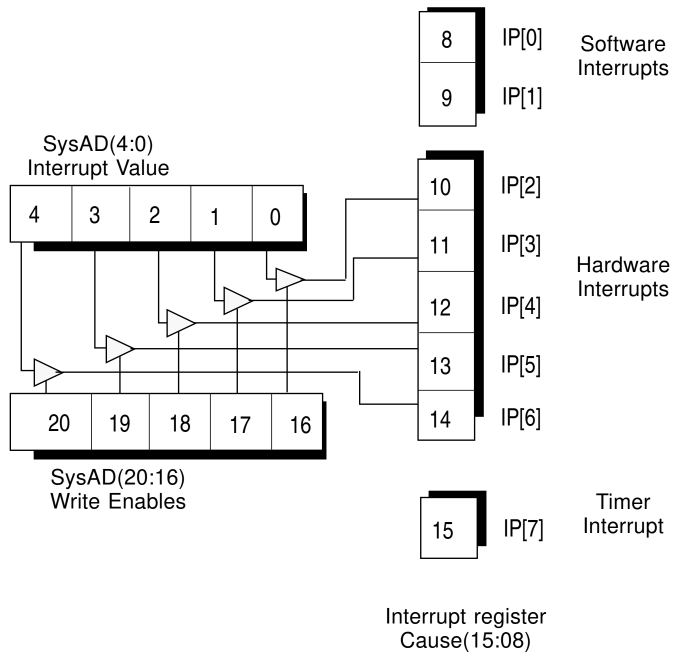

Figure 6-5 Hardware Interrupts

Software Interrupts

The two software interrupts are accessible as bits 9:8 of the *Cause* register, as shown in Figure 6-5. An MTC0 instruction is used to write these bits.

Timer Interrupt

The timer interrupt is accessible as bit 15 of the *Cause* register, IP[7] , as shown in Figure 6-5. This bit is set when one of the following occurs:

- the *Count* register is equal to the *Compare* register
- either one of the two performance counters overflows

Nonmaskable Interrupt

A nonmaskable interrupt is accessible to an external agent as the SysNMI* signal. To post a nonmaskable interrupt, an external agent asserts SysNMI* for at least one SysClk cycle.

The processor recognizes the nonmaskable interrupt on the first SysClk cycle that SysNMI* is asserted. After the nonmaskable interrupt is serviced, an external agent may post another nonmaskable interrupt by first negating SysNMI* for at least one SysClk cycle, and reasserting SysNMI* for at least one SysClk cycle.

6.15 Protocol Abbreviations

The following abbreviations are used in the System interface protocols:

SysCmd[11:0] Abbreviations

Cmd	Unspecified command
BlkRd	Block read request command
RdShd	Coherent block read shared request command
RdExc	Coherent block read exclusive request command
DSPRd	Double/single/partial-word read command
BlkWr	Block write request command
DSPWr	Double/single/partial-word write request command
Ugd	Upgrade request command
Elm	Eliminate request command
IvnShd	Intervention shared request command
IvnExc	Intervention exclusive request command
Alc	Allocate request number command
Ivd	Invalidate request command
Int	Interrupt request command
ExtCoh	External coherency request command
ReqDat	Request data
RspDat	Response data
ReqLst	Request last
RspLst	Response last
Empty	Empty; SysCmd(11:0) and SysAD(63:0) are undefined

SysAD[63:0] Abbreviations

Adr	Physical address
Dat	Unspecified data
Dat<n>	Doubleword n of a block

SysState[2:0] Abbreviations

State	Unspecified state
Ivd	*Invalid*
Shd	*Shared*
ClnExc	*CleanExclusive*
DrtExc	*DirtyExclusive*

SysResp[4:0]	Abbreviations
Rsp	Unspecified completion response
ACK	Acknowledge completion response
ERR	Error completion response
NACK	Negative acknowledge completion response
Master	Abbreviations
EA	External agent
P_n	R10000 processor whose device number is *n*
-	Dead cycle

6.16 System Interface Arbitration

The processor supports a simple System interface arbitration protocol, which relies on an external arbiter. This protocol is used in uniprocessor systems, multiprocessor systems using dedicated external agents, and multiprocessor systems using the cluster bus. System interface arbitration is handled by the SysReq*, SysGnt*, and SysRel* signals (request, grant, and release).

As described earlier in this chapter, the System interface resides in either master or slave state; the processor enters slave state during all of the reset sequences.

When mastership of the System interface changes, there is always one dead SysClk cycle during which the bidirectional signals are not driven; the processor ignores all bidirectional signals during this dead SysClk cycle.

The protocol supports overlapped arbitration which allows arbitration to occur in parallel with requests and responses. This results in fewer wasted cycles when mastership of the System interface changes.

Grant parking is also supported, allowing a device to retain mastership of the System interface as long as no other device requests the System interface.

In multiprocessor systems using the cluster bus, the external arbiter typically implements a round-robin priority scheme.

System Interface Arbitration Rules

The rules for the System interface arbitration are listed below:

- If the System interface is in slave state, and a processor request or coherency data response is ready for issue, and the required resources are available (e.g. a free request number, SysRdRdy* asserted, etc.), the processor asserts SysReq*. The processor will not assert SysReq* unless all of the above conditions are met.
- The processor waits for the assertion of SysGnt*.
- When the processor observes the assertion of SysGnt* it negates SysReq* two SysClk cycles later. Once the processor asserts SysReq*, it does not negate SysReq* until the assertion of SysGnt*, even if the need for the System interface bus is contravened by an external coherency request.
- When the processor observes the assertion of SysRel*, it enters master state two SysClk cycles later, and begins to drive the System interface bus. SysRel* may be asserted coincidentally with or later than SysGnt*.
- Once in master state, the processor does not relinquish mastership of the System interface until it observes the negation of SysGnt*.
- The processor indicates it is relinquishing mastership of the System interface bus by asserting SysRel* for one SysClk cycle, two or more SysClk cycles after the negation of SysGnt*. The processor may issue any type of processor request or coherency data response in the two SysClk cycles following the negation of SysGnt*. This may delay the assertion of SysRel*.

Uniprocessor System

Figure 6-6 shows how the System interface arbitration signals are used in a uniprocessor system. Note that this same configuration would be used in a multiprocessor system using dedicated external agents.

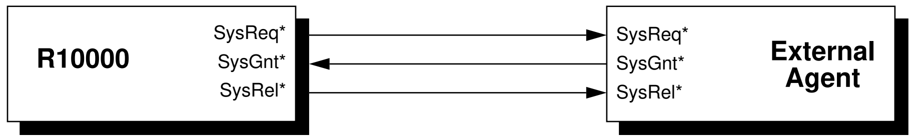

Figure 6-6 Arbitration Signals for Uniprocessor System

Figure 6-7 is an example of the operation of the System interface arbitration in a uniprocessor system. The *Master* row in the following figures indicates which device is driving the System interface bidirectional signals (P_0 and EA in Figure 6-7). When this row contains a dash (-), as shown in Cycle 12 of Figure 6-7, mastership of the System interface is changing and no device is driving the System interface bidirectional signals for this one dead SysClk cycle.

The external agent generally asserts the SysGnt* signal, which allows the processor to issue requests at any time.

When the external agent needs to return an external data response, it negates SysGnt* for a minimum of one cycle, waits for the processor to assert SysRel*, and then begins driving the System interface bus after one dead SysClk cycle.

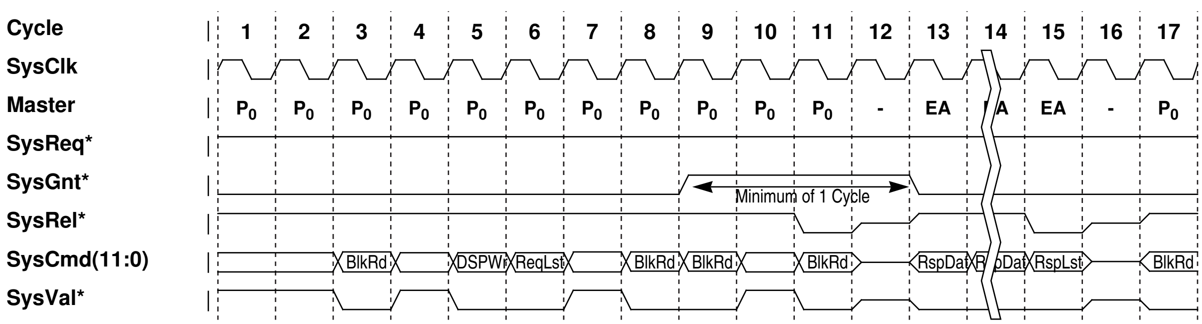

Figure 6-7 Arbitration Protocol for Uniprocessor System

Multiprocessor System Using Cluster Bus

Figure 6-8 shows how the System interface arbitration signals are used in a four-processor system using the cluster bus.

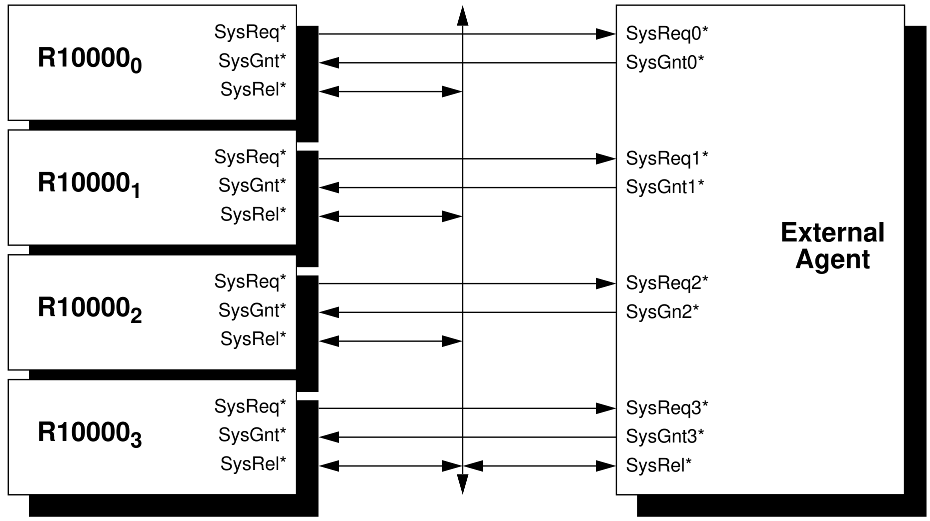

Figure 6-8 Arbitration Signals for Multiprocessor System Using the Cluster Bus

Figure 6-9 is an example of the System interface arbitration in a four-processor system using the cluster bus.

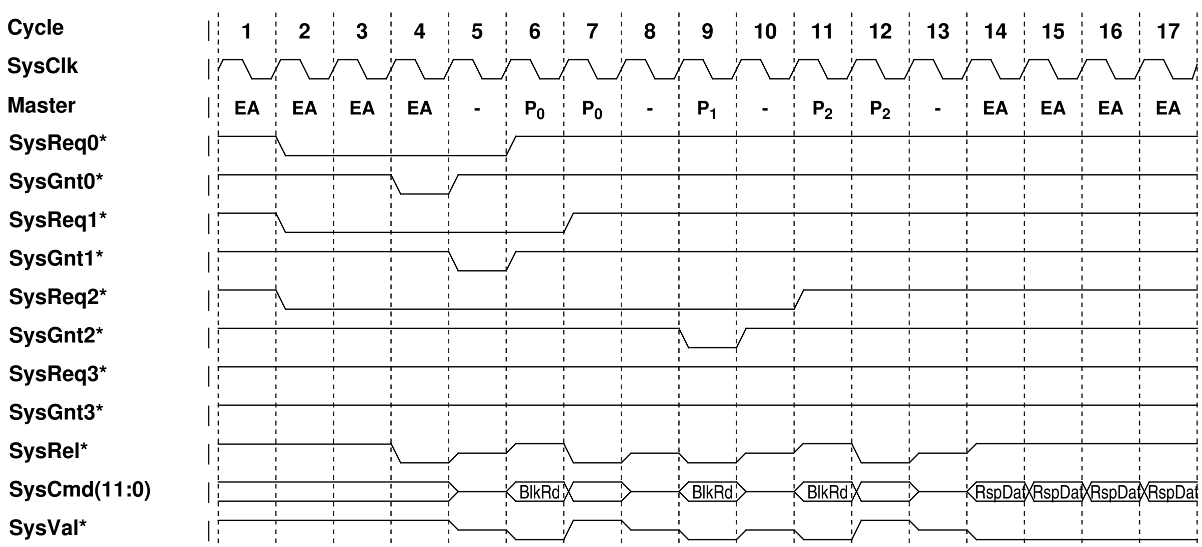

Figure 6-9 Arbitration Protocol for Multiprocessor System Using the Cluster Bus

6.17 System Interface Request and Response Protocol

The following sections detail the System interface request and response protocol. A 32-word secondary cache block size is assumed in the examples below.

Processor Request Protocol

A processor request is generated when the R10000 processor requires a system resource.

The processor may only issue a processor request when the System interface is in master state. If the System interface is in master state, the processor may issue a processor request immediately. Processor requests may occur in adjacent SysClk cycles. If the System interface is not in master state, the processor must first assert SysReq* , and then wait for the external agent to relinquish mastership of the System interface bus by asserting SysGnt* and SysRel* .

When multiple processor requests and/or coherency data responses are ready and meet all issue requirements, the processor uses the following priority:

- block read and upgrade requests have the highest priority, followed by
- processor coherency data responses,
- processor eliminate and typical block write requests,
- processor double/single/partial-word read/write and uncached accelerated block write requests, which have the lowest priority.

Processor Block Read Request Protocol

A processor block read request results from a cached instruction fetch, load, store, or prefetch that misses in the secondary cache.

The processor issues a processor block read request with a single address cycle. The address cycle consists of the following:

- negating SysCmd[11]
- driving a free request number on SysCmd[10:8]
- driving the block read command on SysCmd[7:5]
- driving the read cause indication on SysCmd[4:3]
- driving the secondary cache block former state on SysCmd[2:1]
- asserting SysCmd[0]
- driving the target indication on SysAD[63:60]
- driving the secondary cache block way on SysAD[57]
- driving the physical address on SysAD[39:0]
- asserting SysVal*

The processor may only issue a processor block read request address cycle when the following are true:

- the System interface is in master state
- SysRdRdy* was asserted two SysClk cycles earlier
- there is no conflicting entry in the outgoing buffer
- the maximum number of outstanding processor requests specified by the PrcReqMax mode bits is not exceeded
- there is a free request number
- the processor is not the target of a conflicting outstanding external coherency request

A single processor may have as many as four processor block read requests outstanding on the System interface at any given time.

Figure 6-10 depicts four processor block read requests. Since the System interface is initially in slave state, the processor must first assert SysReq* and then wait until the external agent relinquishes mastership of the System interface by asserting SysGnt* and SysRel* .

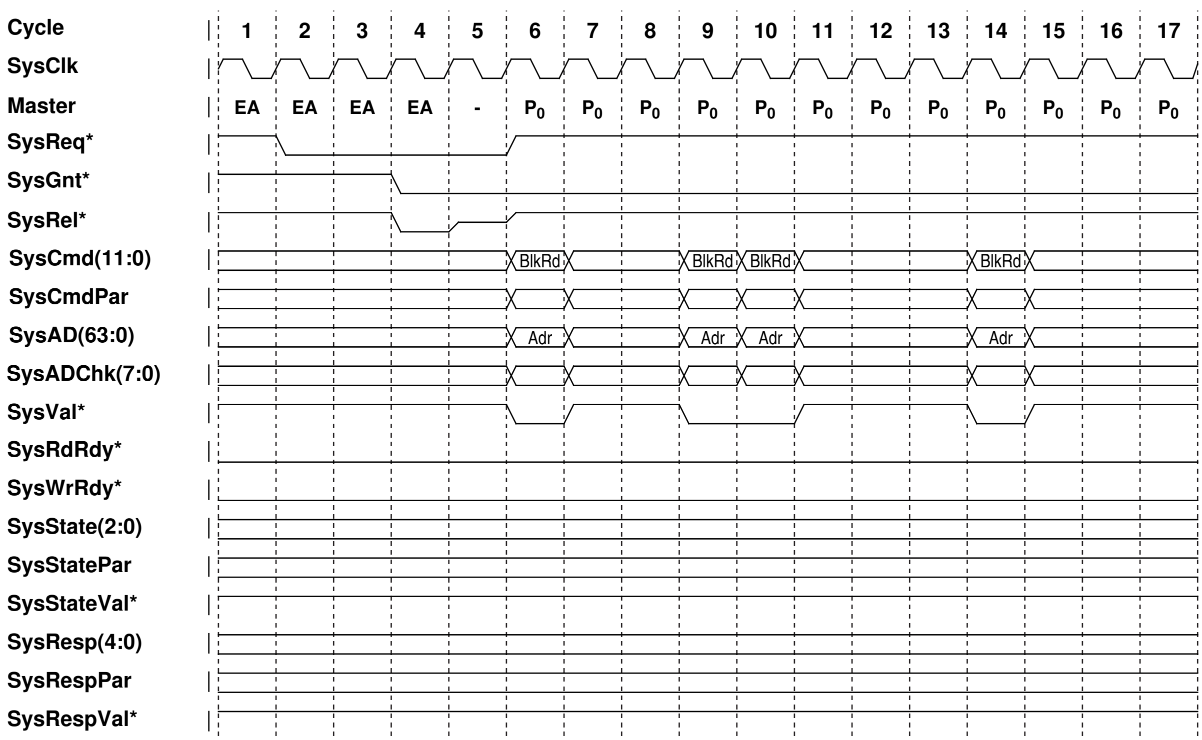

Figure 6-10 Processor Block Read Request Protocol

Processor Double/Single/Partial-Word Read Request Protocol

A processor double/single/partial-word read request results from an uncached instruction fetch or load.

The processor issues a processor double/single/partial-word read request with a single address cycle. The address cycle consists of:

- negating SysCmd[11]
- driving a free request number on SysCmd[10:8]
- driving the double/single/partial-word read command on SysCmd[7:5]
- driving the read cause indication on SysCmd[4:3]
- driving the data size indication on SysCmd[2:0]
- driving the target indication on SysAD[63:60]
- driving the uncached attribute on SysAD[59:58]
- driving the physical address on SysAD[39:0]
- asserting SysVal*

The processor may only issue a processor double/single/partial-word read request address cycle when:

- the System interface is in master state
- SysRdRdy* was asserted two SysClk cycles previously
- the maximum number of outstanding processor requests specified by the PrcReqMax mode bits is not exceeded
- there is a free request number

A single processor may have a maximum of one processor double/single/partial-word read request outstanding on the System interface at any given time.

Figure 6-11 depicts a processor double/single/partial-word read request. Since the System interface is initially in slave state, the processor must first assert SysReq* and then wait until the external agent gives up mastership of the System interface by asserting SysGnt* and SysRel*.

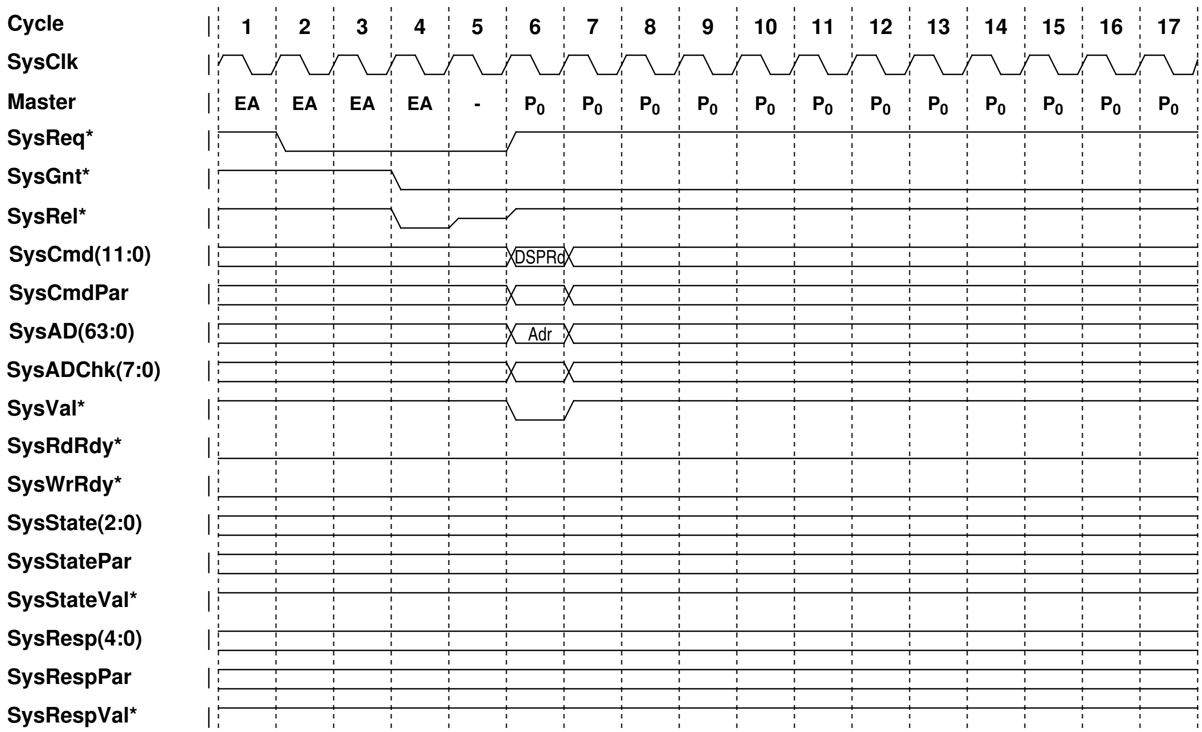

Figure 6-11 Processor Double/Single/Partial-Word Read Request Protocol

Processor Block Write Request Protocol

A processor block write request results from the following:

- replacement of a *DirtyExclusive* secondary cache block due to a load, store, or prefetch secondary cache miss
- a CACHE Index WriteBack Invalidate (S) or Hit WriteBack Invalidate (S) instruction
- a completely gathered uncached accelerated block

As shown in Figure 6-12, the processor issues a processor block write request with a single address cycle followed by 8 or 16 data cycles.

The address cycle consists of the following:

- negating SysCmd[11]
- driving the block write command on SysCmd[7:5]
- driving the write cause indication on SysCmd[4:3]
- asserting SysCmd[0]
- driving the target indication on SysAD[63:60]
- driving the physical address on SysAD[39:0]
- asserting SysVal*

If the processor block write request results from the writeback of a secondary cache block, the *DirtyExclusive* secondary cache block former state is driven on SysCmd[2:1] , and the secondary cache block way is driven on SysAD[57] .

If the processor block write request results from a completely gathered uncached accelerated block, the uncached attribute is driven on SysAD[59:58] .

Each data cycle consists of the following:

- asserting SysCmd[11]
- driving the data quality indication on SysCmd[5]
- driving the data type indication on SysCmd[4:3]
- driving the data on SysAD[63:0]
- asserting SysVal*

The first 7 or 15 data cycles have a request data type indication, and the last data cycle has a request last data type indication.

The processor may negate SysVal* between data cycles of a processor block write request only if the SCClk frequency is less than half of the SysClk frequency.

The processor may only issue a processor block write request address cycle when the following are true:

- the System interface is in master state
- SysWrRdy* was asserted two SysClk cycles previously
- the processor is not the target of a conflicting outstanding external coherency request

Figure 6-12 depicts two adjacent processor block write requests. Since the System interface is initially in slave state, the processor must first assert SysReq* and then wait until the external agent relinquishes mastership of the System interface by asserting SysGnt* and SysRel*.

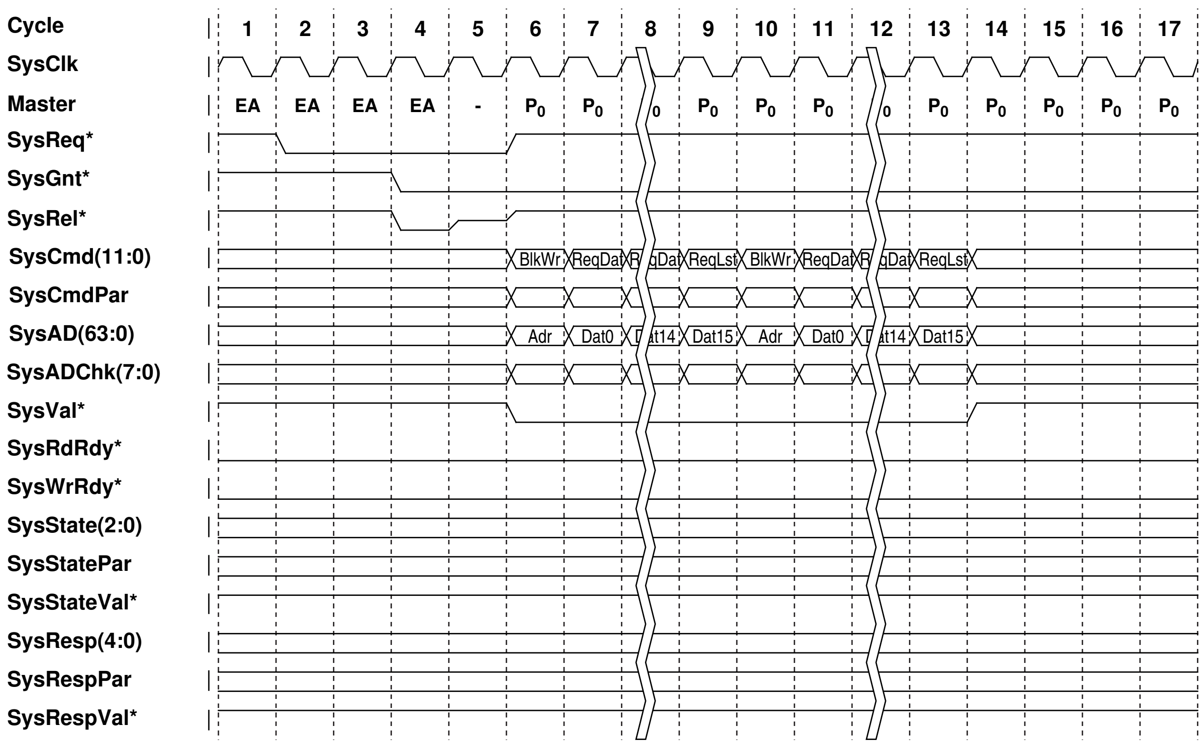

Figure 6-12 Processor Block Write Request Protocol

Processor Double/Single/Partial-Word Write Request Protocol

A processor double/single/partial-word write request results from an uncached store or incompletely gathered uncached accelerated block.

As shown in Figure 6-13, the processor issues a processor double/single/partial-word write request with a single address cycle immediately followed by a single data cycle.

The address cycle consists of the following:

- negating SysCmd[11]
- driving the double/single/partial-word write command on SysCmd[7:5]
- driving the write cause indication on SysCmd[4:3]
- driving the data size indication on SysCmd[2:0]
- driving the target indication on SysAD[63:60]
- driving the uncached attribute on SysAD[59:58]
- driving the physical address on SysAD[39:0]
- asserting SysVal*

The data cycle consists of the following:

- asserting SysCmd[11]
- driving the request last data type indication on SysCmd[4:3]
- driving the write data on SysAD[63:0]
- asserting SysVal*

The processor may only issue a processor double/single/partial-word write request address cycle when the System interface is in master state and SysWrRdy* was asserted two SysClk cycles previously.

Figure 6-13 depicts three processor double/single/partial write requests. Since the System interface is initially in slave state, the processor must first assert SysReq* and then wait until the external agent relinquishes mastership of the System interface by asserting SysGnt* and SysRel* .

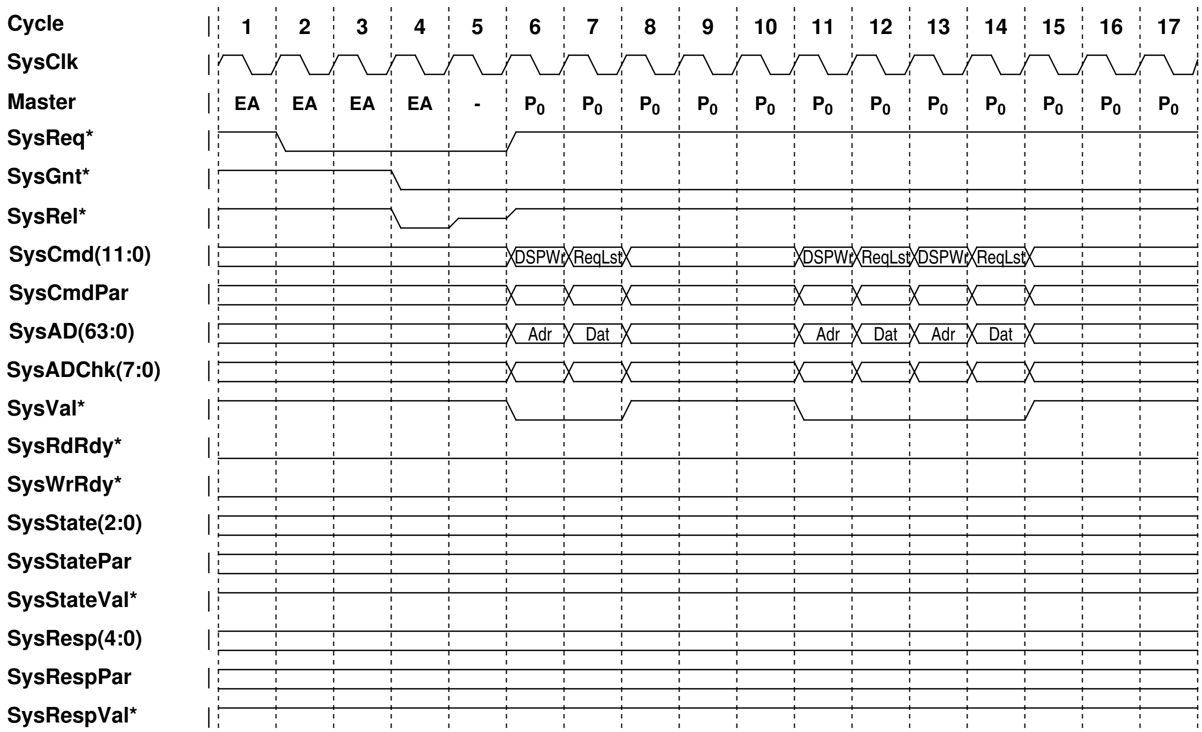

Figure 6-13 Processor Double/Single/Partial-Word Write Request Protocol

Processor Upgrade Request Protocol

A processor upgrade request results from a store or prefetch exclusive that hits a *Shared* block in the secondary cache.

As shown in Figure 6-14, the processor issues a processor upgrade request with a single address cycle. This address cycle consists of the following:

- negating SysCmd[11]
- driving a free request number on SysCmd[10:8]
- driving the upgrade command on SysCmd[7:5]
- driving the upgrade cause indication on SysCmd[4:3]
- driving the secondary cache block former state on SysCmd[2:1]
- asserting SysCmd[0]
- driving the target indication on SysAD[63:60]
- driving the secondary cache block way on SysAD[57]
- driving the physical address on SysAD[39:0]
- asserting SysVal*

The processor may only issue a processor upgrade request address cycle when the following are true:

- the System interface is in master state
- SysRdRdy* was asserted two SysClk cycles previously
- the maximum number of outstanding processor requests specified by the PrcReqMax mode bits is not exceeded
- there is a free request number
- the processor is not the target of a conflicting outstanding external coherency request

A single processor may have as many as four processor upgrade requests outstanding on the System interface at any given time.

Figure 6-14 depicts four processor upgrade requests. Since the System interface is initially in slave state, the processor must first assert SysReq* and then wait until the external agent relinquishes mastership of the System interface by asserting SysGnt* and SysRel* .

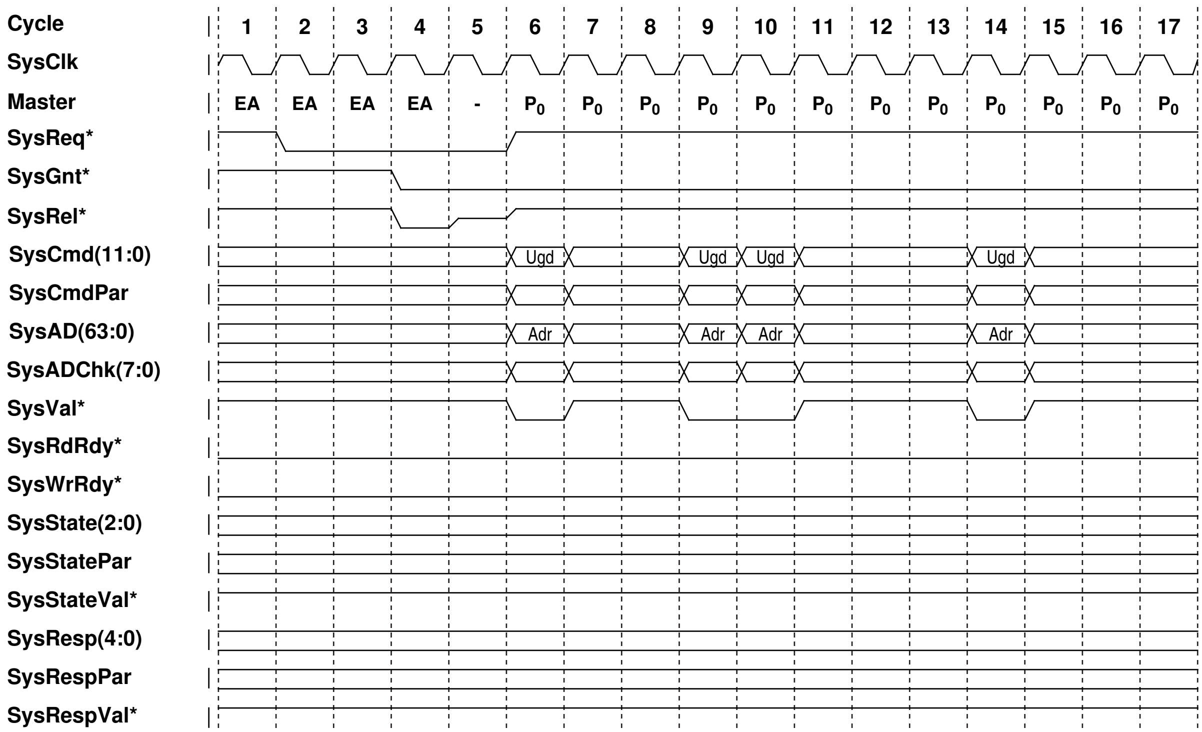

Figure 6-14 Processor Upgrade Request Protocol

Processor Eliminate Request Protocol

A processor eliminate request results from the following:

- a cached instruction fetch, load, store, or prefetch that misses in the secondary cache and forces the replacement of a *Shared* or *CleanExclusive* secondary cache block
- a CACHE Index WriteBack Invalidate (S), Hit Invalidate (S), or Hit WriteBack Invalidate (S) instruction that forces the invalidation of a *Shared* or *CleanExclusive* secondary cache block
- a CACHE Hit Invalidate (S) instruction that forces the invalidation of a *DirtyExclusive* secondary cache block.

A processor eliminate request notifies the external agent that a *Shared*, *CleanExclusive*, or *DirtyExclusive* block has been eliminated from the secondary cache. Such requests are useful for systems implementing a directory-based coherency protocol, and are enabled by asserting the PrcElmReq mode bit.

The processor issues a processor eliminate request with a single address cycle. This address cycle consists of the following:

- negating SysCmd[11]
- driving the special command on SysCmd[7:5]
- driving the eliminate special cause indication on SysCmd[4:3]
- driving the secondary cache block former state on SysCmd[2:1]
- asserting SysCmd[0]
- driving the target indication on SysAD[63:60]
- driving the secondary cache block way on SysAD[57]
- driving the physical address of the eliminated secondary cache block on SysAD[39:0]
- asserting SysVal*

The processor may only issue a processor eliminate request address cycle when the following are true:

- the System interface is in master state
- SysWrRdy* was asserted two SysClk cycles previously
- the PrcElmReq mode bit is asserted
- the processor is not the target of a conflicting outstanding external coherency request

Figure 6-15 depicts three processor eliminate requests. Since the System interface is initially in slave state, the processor must first assert SysReq* and then wait until the external agent relinquishes mastership of the System interface by asserting SysGnt* and SysRel*.

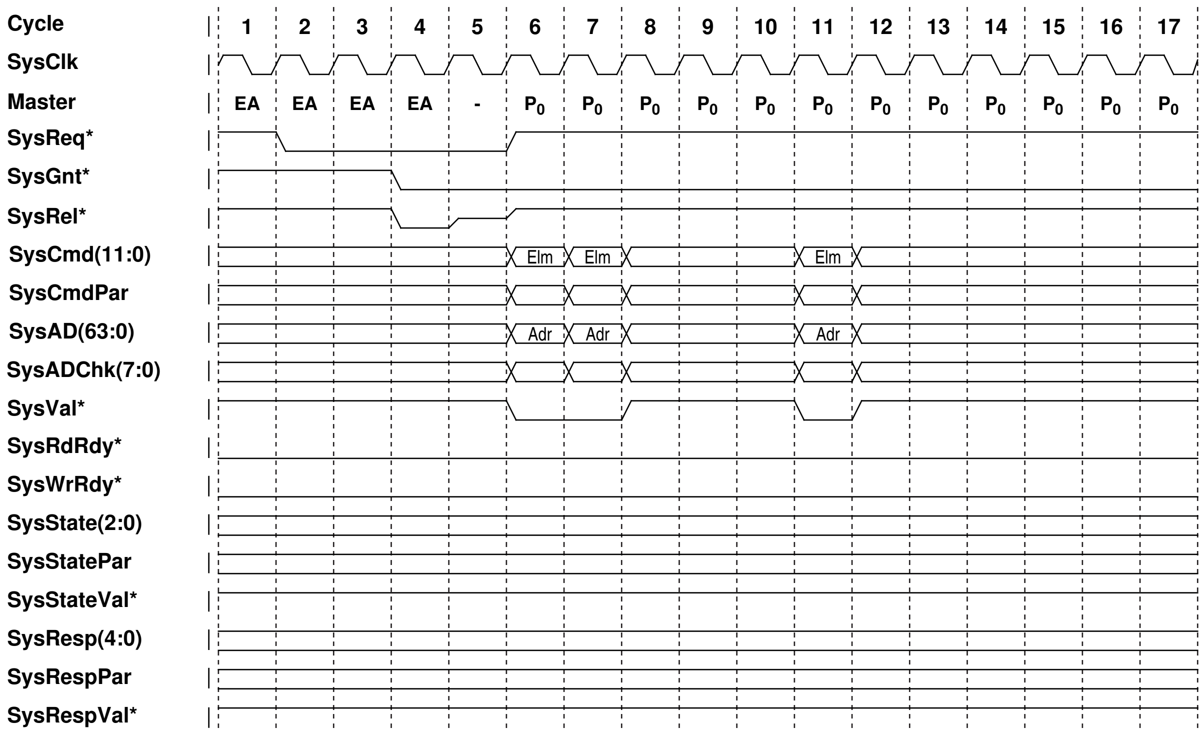

Figure 6-15 Processor Eliminate Request Protocol

Processor Request Flow Control Protocol

The processor provides the signals SysRdRdy* and SysWrRdy* to allow an external agent to control the flow of processor requests. SysRdRdy* controls the flow of processor read and upgrade requests whereas SysWrRdy* controls the flow of processor write and eliminate requests.

The processor can only issue a processor read or upgrade request address cycle to the System interface if SysRdRdy* was asserted two SysClk cycles previously. Similarly, the processor can only issue the address cycle of a processor write or eliminate request to the System interface if SysWrRdy* was asserted two SysClk cycles previously.

To determine the processor request buffering requirements for the external agent, note that the processor can issue any combination of processor requests in adjacent SySClk cycles. Also, since the System interface operates register-to-register with the external agent, a round trip delay of four SysClk cycles occurs between a processor request address cycle which prompts the external agent for flow control, and the flow control actually preventing any additional processor request address cycles from occurring. Consequently, if the maximum number of outstanding processor requests specified by the PrcReqMax mode bits is four, the external agent must be able to accept at least four processor read or upgrade requests. Also, the external agent must be able to accept at least four processor eliminate requests, two processor double/single/partial-word write requests, or one processor block write request.

Figure 6-16 depicts three processor double/single/partial-word write requests and four processor block read requests. After sensing the first processor double/single/partial-word write request, the external agent negates SysWrRdy* . The external agent must have buffering sufficient for one additional processor write request before the flow control takes effect.

The external agent negates SysRdRdy* upon observing the first processor read request. The external agent must have buffering sufficient for three additional processor read requests before the flow control takes effect.

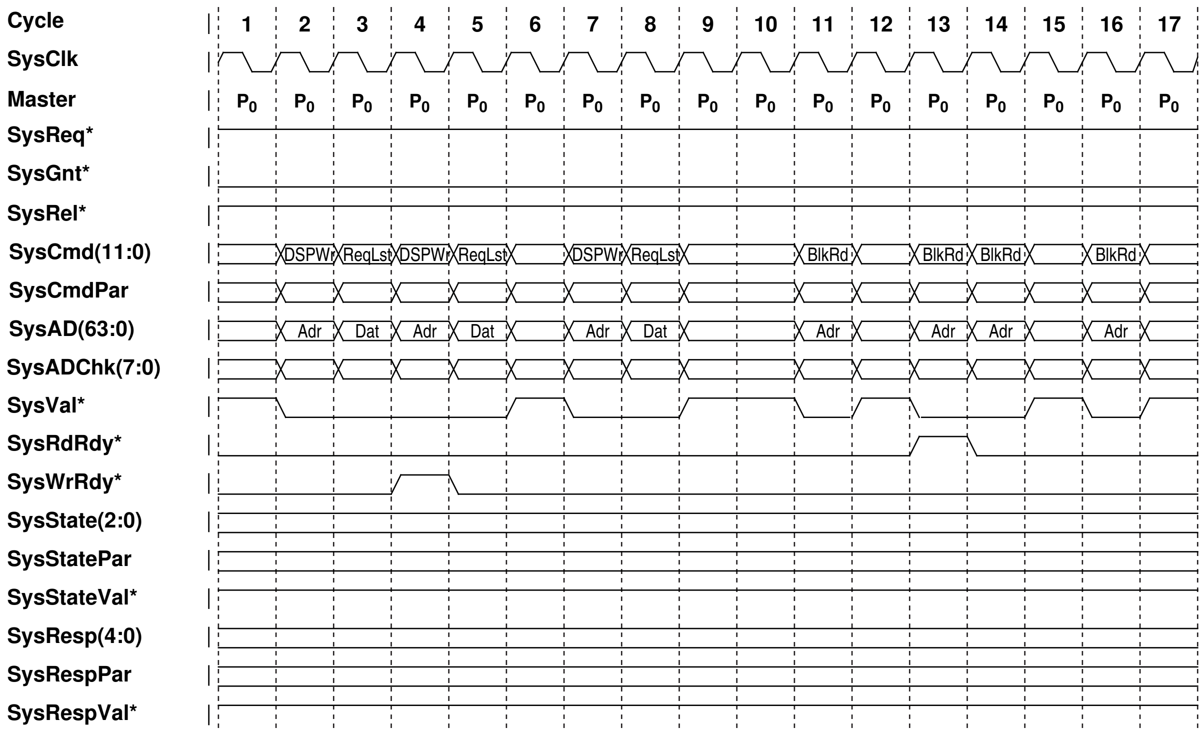

Figure 6-16 Processor Request Flow Control Protocol

External Response Protocol

The processor supports two classes of external responses:

- external data responses provide a double/single/partial-word of data or provide a block of data using the SysAD[63:0] bus
- external completion responses provide an acknowledge, error, or negative acknowledge indication using the SysResp[4:0] bus

An external agent may only issue an external data response to the processor when the System interface is in slave state. If the System interface is not already in slave state, the external agent must first negate SysGnt* and then wait for the processor to assert SysRel*. If the System interface is already in slave state, the external agent may issue an external data response immediately.

External data responses may be accepted by the processor in adjacent SysClk cycles and in arbitrary order, relative to corresponding processor requests.

An external agent may issue an external completion response when the System interface is in either master or slave state. External completion responses may be accepted by the processor in adjacent SysClk cycles and in arbitrary order, relative to the corresponding processor requests.

External Block Data Response Protocol

An external agent may issue an external block data response in response to a processor block read or upgrade request.

An external agent issues an external block data response with 8 or 16 data cycles. Each data cycle consists of the following:

- asserting SysCmd[11]
- driving the request number associated with the corresponding processor request on SysCmd[10:8]
- driving the data quality indication on SysCmd[5]
- driving the data type indication on SysCmd[4:3]
- driving the cache block state on SysCmd[2:1]
- driving the ECC check indication on SysCmd[0]
- driving the data on SysAD[63:0]
- asserting SysVal*

The first 7 or 15 data cycles have a response data type indication, and the last data cycle has a response last data type indication. The external agent may negate SysVal* between data cycles of an external block data response.

External block data response data must be supplied in subblock order, beginning with the quadword-aligned address specified by the corresponding processor request.

External block data responses for processor coherent block read shared or noncoherent block read requests may indicate a state of *Shared*, *CleanExclusive*, or *DirtyExclusive*. External block data responses for processor coherent block read exclusive or upgrade requests may indicate a state of *CleanExclusive* or *DirtyExclusive*.

Figure 6-17 depicts two processor block read requests and the corresponding external block data responses.

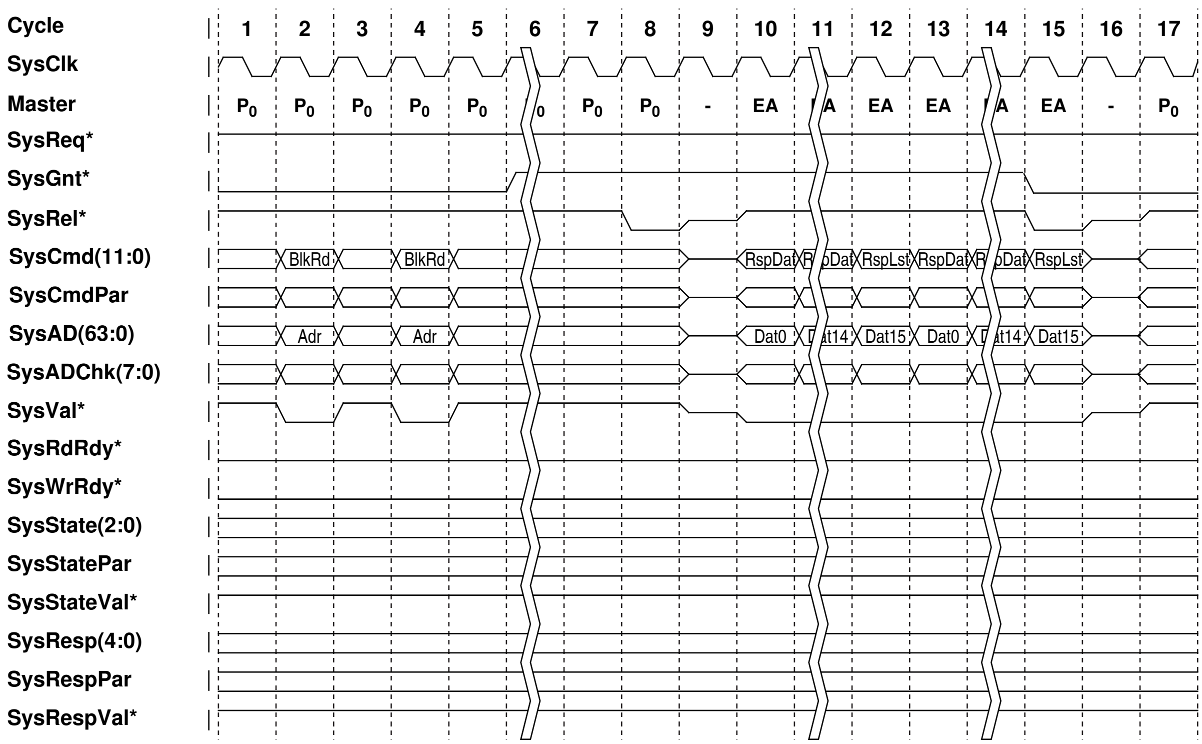

Figure 6-17 External Block Data Response Protocol

External Double/Single/Partial-Word Data Response Protocol

An external agent may issue an external double/single/partial-word data response in response to a processor double/single/partial-word read request.

An external agent issues an external double/single/partial-word data response with a single data cycle; the data cycle consists of:

- asserting SysCmd[11]
- driving the request number associated with the corresponding processor request on SysCmd[10:8]
- driving the data quality indication on SysCmd[5]
- driving the response last data type indication on SysCmd[4:3]
- driving the ECC check indication on SysCmd[0]
- driving the data on SysAD[63:0]
- asserting SysVal*

Figure 6-18 depicts a processor double/single/partial-word read request and the corresponding external double/single/partial-word data response.

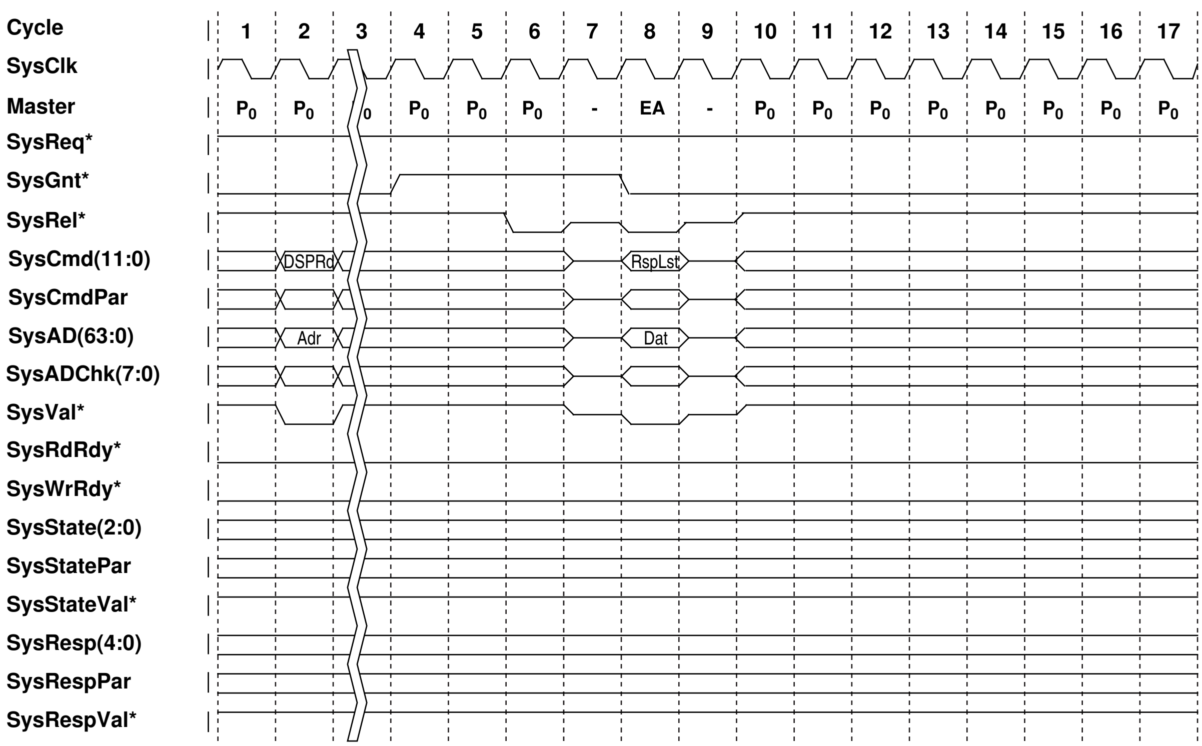

Figure 6-18 External Double/Single/Partial-Word Data Response Protocol

External Completion Response Protocol

An external agent issues an external completion response to provide an acknowledge, error, or negative acknowledge to an outstanding request, and to free the associated request number.

An external agent issues an external completion response by driving the response on SysResp[4:0] and asserting SysRespVal* for one SysClk cycle. SysResp[4:2] contains the request number associated with the corresponding outstanding request and SysResp[1:0] contains an acknowledge, error, or negative acknowledge indication, as described below:

- The external agent issues an external ACK completion response for a processor read or upgrade request to indicate that the request was successful. An external ACK completion response may only be issued for a processor read request if a corresponding external data response is coincidentally or previously issued.
- The external agent issues an external ERR completion response for a processor read or upgrade request to indicate that the request was unsuccessful. Upon receiving an external ERR completion response, the processor takes a Bus Error exception on the associated instruction. If the processor read or upgrade request was caused by a PREFETCH instruction, no exception is taken. Also, if the request was caused by a speculative instruction, no exception is taken.
- The external agent issues an external NACK completion response for a processor read or upgrade request to indicate that the request was not accepted. Upon receiving an external NACK completion response, the processor re-evaluates the associated instruction. Due to the speculative nature of the R10000 processor, the re-evaluation may or may not result in the reissue of a similar processor request.

An external ERR or NACK completion response issued in response to an external intervention, allocate request number, or invalidate has no affect on the processor except to free the request number.

Figure 6-19 depicts a processor upgrade request and a corresponding external completion response.

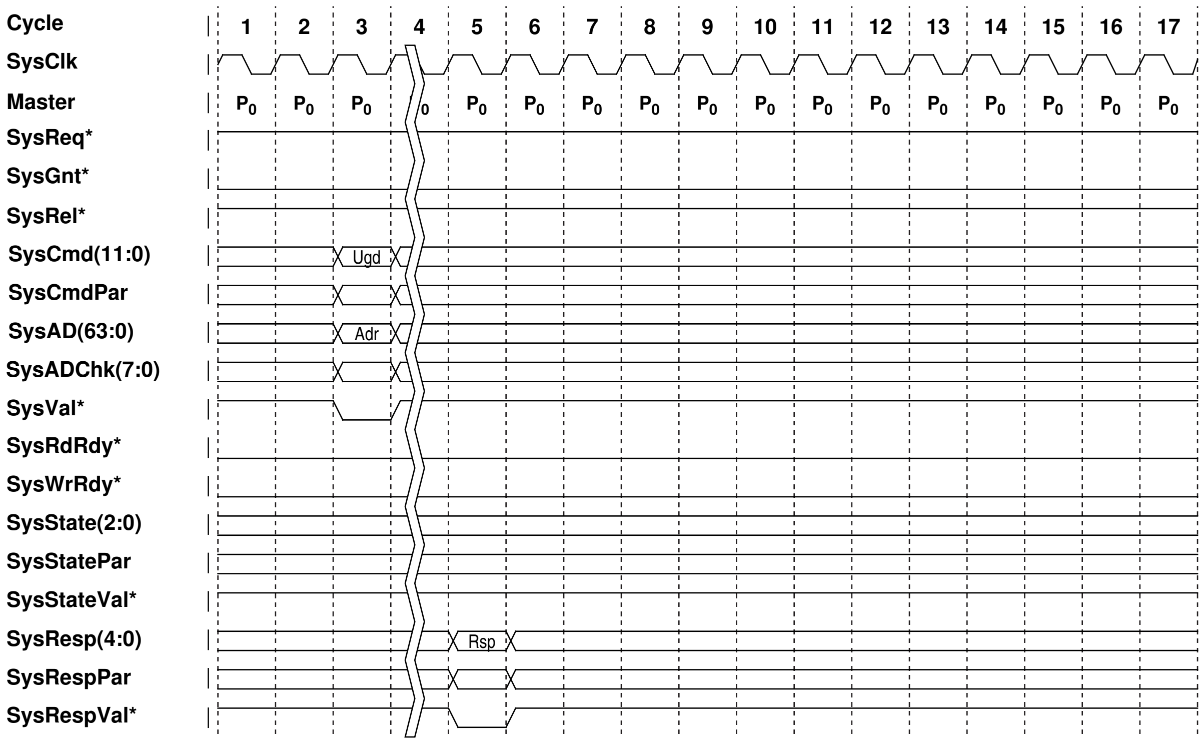

Figure 6-19 External Completion Response Protocol

External Request Protocol

An external agent issues an external request when it requires a resource within the processor.

An external agent may only issue an external request to the processor when the System interface is in slave state. If the System interface is not already in slave state, the external agent must first negate SysGnt* and then wait for the processor to assert SysRel*. If the System interface is already in slave state, the external agent may issue an external request immediately. The total number of outstanding external requests, including interventions, allocate request numbers, and invalidates, cannot exceed eight.

External requests may be accepted by the processor in adjacent SysClk cycles. External intervention and invalidate requests are considered external coherency requests.

External Intervention Request Protocol

An external agent issues an external intervention request to obtain a *Shared* or *Exclusive* copy of a secondary cache block.

An external agent issues an external intervention request with a single address cycle; this address cycle consists of the following:

- negating SysCmd[11]
- driving a request number on SysCmd[10:8]
- driving the intervention command on SysCmd[7:5]
- driving the ECC check indication on SysCmd[0]
- driving the target indication on SysAD[63:60]
- driving the physical address on SysAD[39:0]
- asserting SysVal*

An external agent may only issue an external intervention request address cycle when the System interface is in slave state; typically a free request number is specified. An external agent may have as many as eight external intervention requests outstanding on the System interface at any given time.

Figure 6-20 depicts three external intervention requests. Since the System interface is initially in master state, the external agent must first negate SysGnt* and then wait until the processor relinquishes mastership of the System interface by asserting SysRel* .

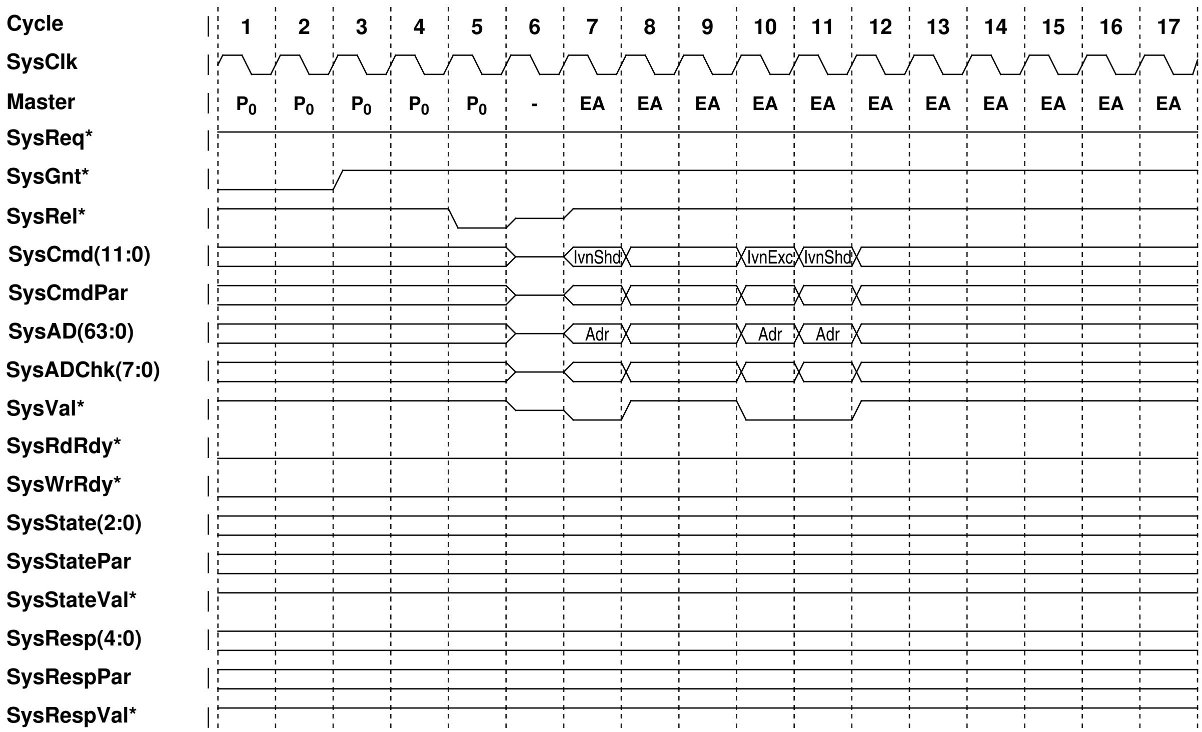

Figure 6-20 External Intervention Request Protocol

External Allocate Request Number Request Protocol

An external agent issues an external allocate request number request to reserve a request number for private use. Once allocated, the processor is prevented from using the request number until an external completion response for the request number is received.

An external agent issues an external allocate request number request with a single address cycle; this address cycle consists of the following:

- negating SysCmd[11]
- driving a free request number on SysCmd[10:8]
- driving the allocate request number command on SysCmd[7:5]
- asserting SysVal*

An external agent may only issue an external allocate request number request address cycle when the System interface is in slave state and there is a free request number. The external agent may have as many as eight external allocate request number requests outstanding on the System interface at any given time.

Figure 6-21 depicts three external allocate request number requests. Since the System interface is initially in master state, the external agent must first negate SysGnt* and then wait until the processor relinquishes mastership of the System interface by asserting SysRel*.

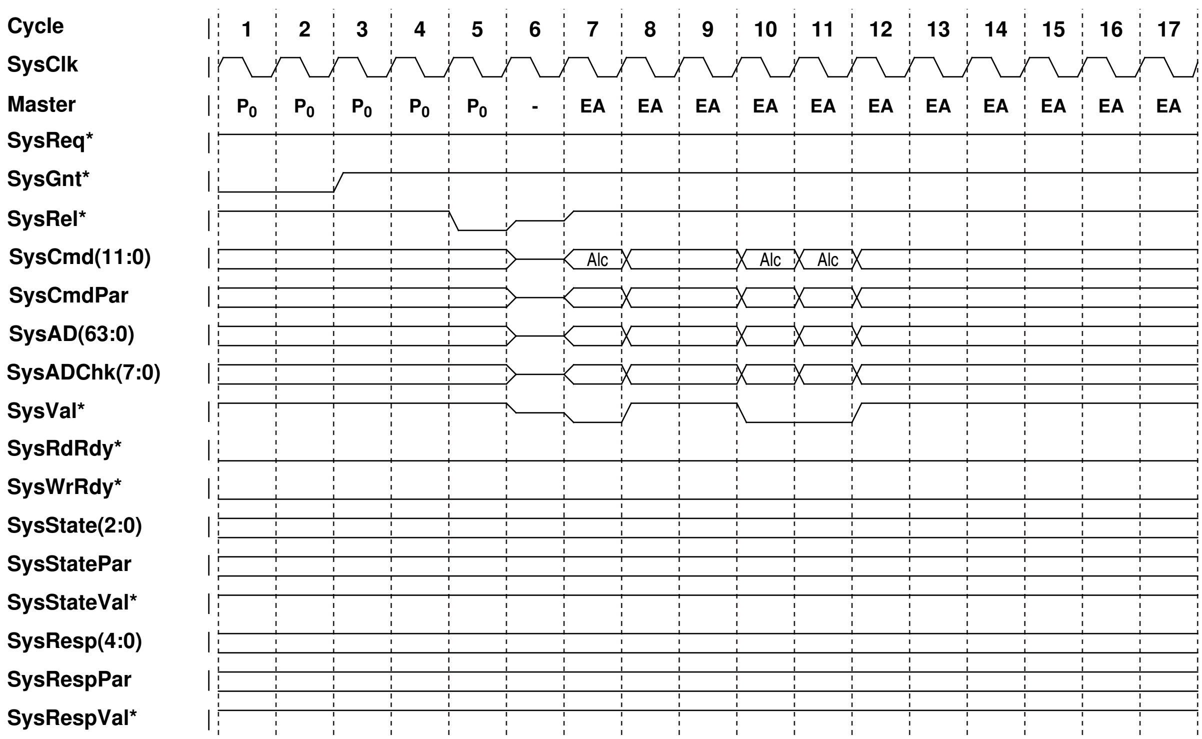

Figure 6-21 External Allocate Request Number Request Protocol

External Invalidate Request Protocol

An external agent issues an external invalidate request to invalidate a secondary cache block.

An external agent issues an external invalidate request with a single address cycle. This address cycle consists of the following:

- negating SysCmd[11]
- driving a request number on SysCmd[10:8]
- driving the invalidate command on SysCmd[7:5]
- driving the ECC check indication on SysCmd[0]
- driving the target indication on SysAD[63:60]
- driving the physical address on SysAD[39:0]
- asserting SysVal*

An external agent may only issue an external invalidate request address cycle when the System interface is in slave state; typically a free request number is specified. An external agent may have as many as eight external invalidate requests outstanding on the System interface at any given time.

Figure 6-22 depicts three external invalidate requests. Since the System interface is initially in master state, the external agent must first negate SysGnt* and then wait until the processor relinquishes mastership of the System interface by asserting SysRel* .

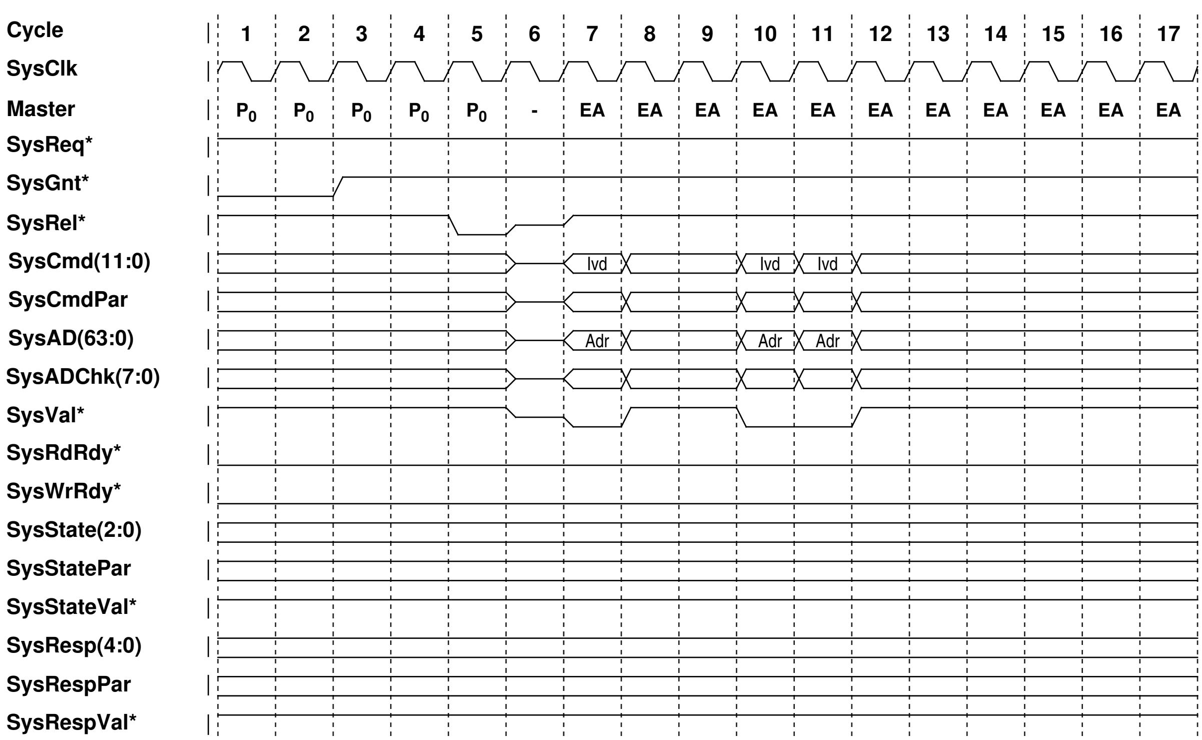

Figure 6-22 External Invalidate Request Protocol

External Interrupt Request Protocol

An external agent issues an external interrupt request to interrupt the normal instruction flow of the processor.

An external agent issues an external interrupt request with a single address cycle. This address cycle consists of the following:

- negating SysCmd[11]
- driving the special command on SysCmd[7:5]
- driving the interrupt special cause indication on SysCmd[4:3]
- driving the ECC check indication on SysCmd[0]
- driving the target indication on SysAD[63:60]
- driving the *Interrupt* register write enables on SysAD[20:16]
- driving the *Interrupt* register values on SysAD[4:0]
- asserting SysVal*

An external agent may only issue an external interrupt request address cycle when the System interface is in slave state.

Figure 6-23 depicts three external interrupt requests. Since the System interface is initially in master state, the external agent must first negate SysGnt* and then wait until the processor relinquishes mastership of the System interface by asserting SysRel* .

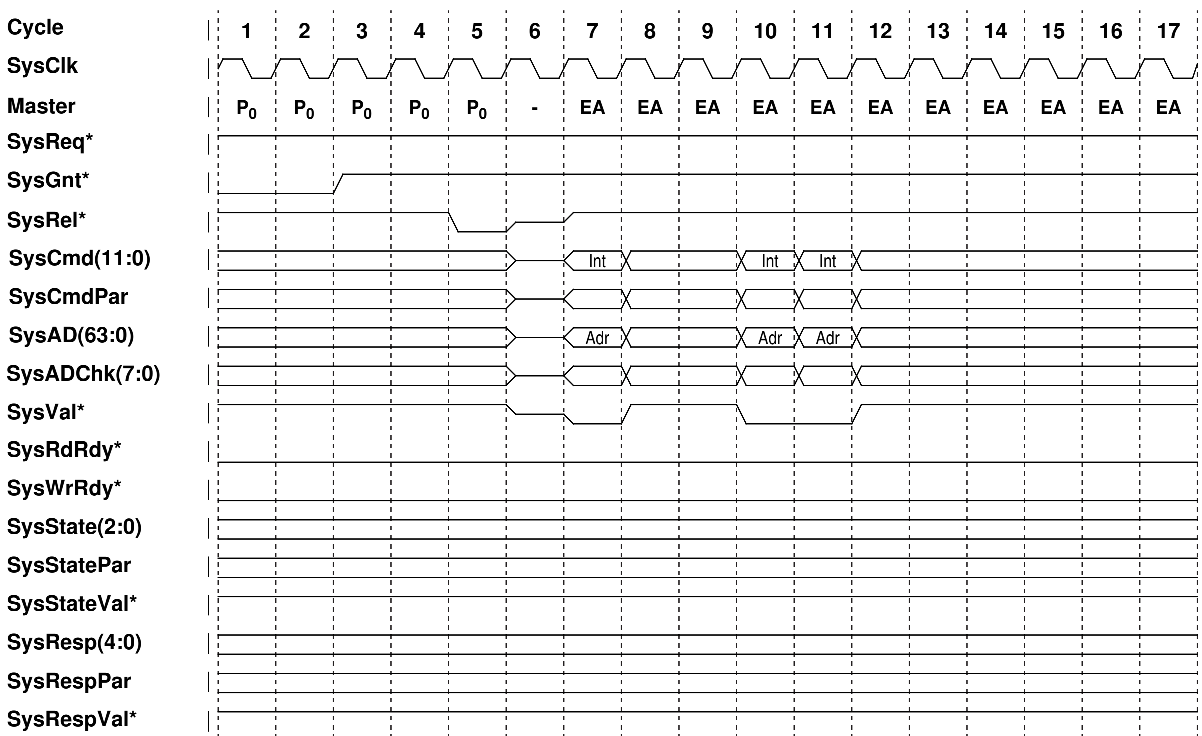

Figure 6-23 External Interrupt Request Protocol

Processor Response Protocol

Processor responses are supplied by the processor in response to external coherency requests that target the processor. The R10000 processor issues a processor coherency state response for each external coherency request that targets the processor. The processor issues a processor coherency data response for each external intervention request that targets the processor and hits a *DirtyExclusive* secondary cache block.

Processor coherency state responses are issued by the processor in the same order that the corresponding external coherency requests are received. Processor coherency state and data responses may occur in adjacent SysClk cycles.

Processor Coherency State Response Protocol

A processor coherency state response results from an external coherency request that targets the processor.

The processor issues a processor coherency state response by driving the secondary cache block tag quality indication on SysState[2], driving the secondary cache block former state on SysState[1:0], and asserting SysStateVal* for one SysClk cycle.

Figure 6-24 depicts two external coherency requests and the resulting processor coherency state responses.

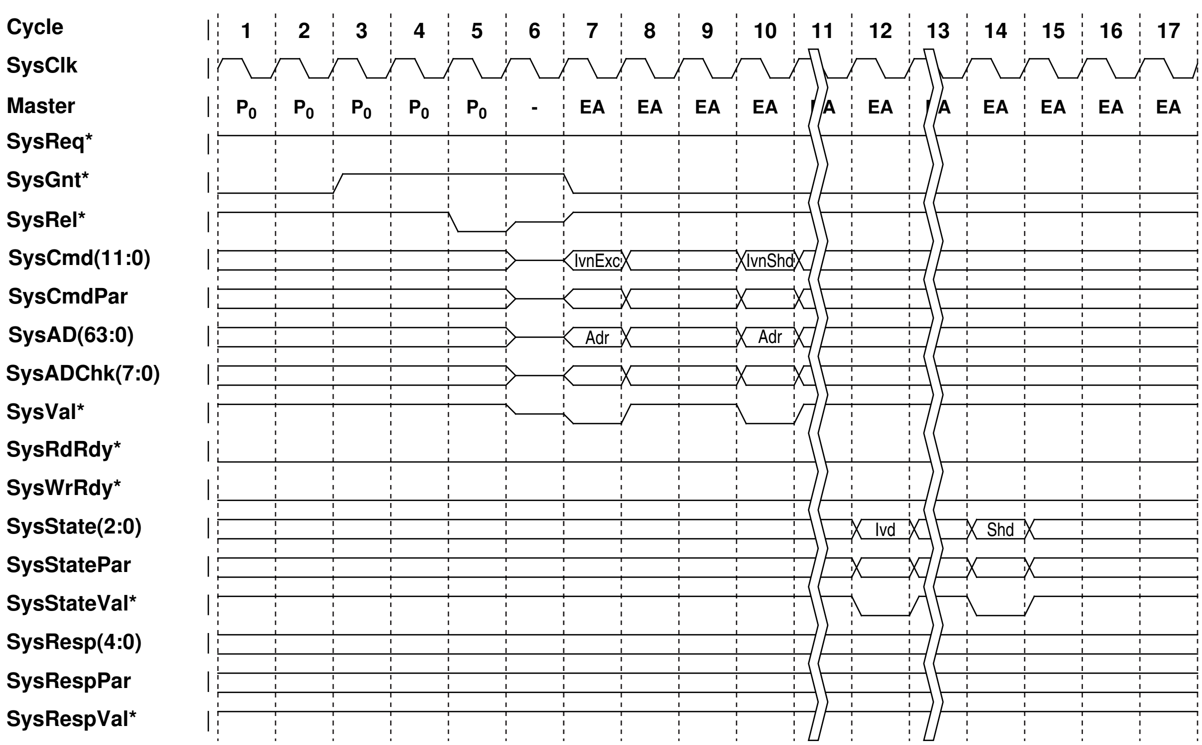

Figure 6-24 Processor Coherency State Response Protocol

Processor Coherency Data Response Protocol

A processor coherency data response results from an external intervention request that targets the processor and hits a *DirtyExclusive* secondary cache block.

The processor issues a processor coherency data response with a single empty cycle followed by either 8 or 16 data cycles. The empty cycle consists of negating SysVal* for a single SysClk cycle. The data cycles consist of the following:

- asserting SysCmd[11]
- driving the request number associated with the corresponding external coherency request on SysCmd[10:8]
- driving the data quality indication on SysCmd[5]
- driving the data type indication on SysCmd[4:3]
- driving the state of the cache block on SysCmd[2:1]
- asserting SysCmd[0]
- driving the data on SysAD[63:0] ,
- asserting SysVal*

The first 7 or 15 data cycles have a response data type indication, and the last data cycle has a response last data indication. The processor may negate SysVal* between data cycles of a processor coherency data response only if the SCClk frequency is less than half of the SysClk frequency.

The processor may only issue a processor coherency data response when the System interface is in master state and SysWrRdy* was asserted two SysClk cycles previously. Note that the empty cycle is considered the issue cycle for a processor coherency data response. If the System interface is not already in master state, the processor must first assert SysReq* , and then wait for the external agent to relinquish mastership of the System interface bus by asserting SysGnt* and SysRel* . If the System interface is already in master state, the processor may issue a processor coherency data response immediately.

When SysStateVal* is negated, SysState[0] indicates if a processor coherency data response is ready to be issued. Processor coherency data response data is supplied in subblock order, beginning with the quadword-aligned address specified by the corresponding external coherency request.

Figure 6-25 depicts one external coherency request and the resulting processor coherency state and data responses.

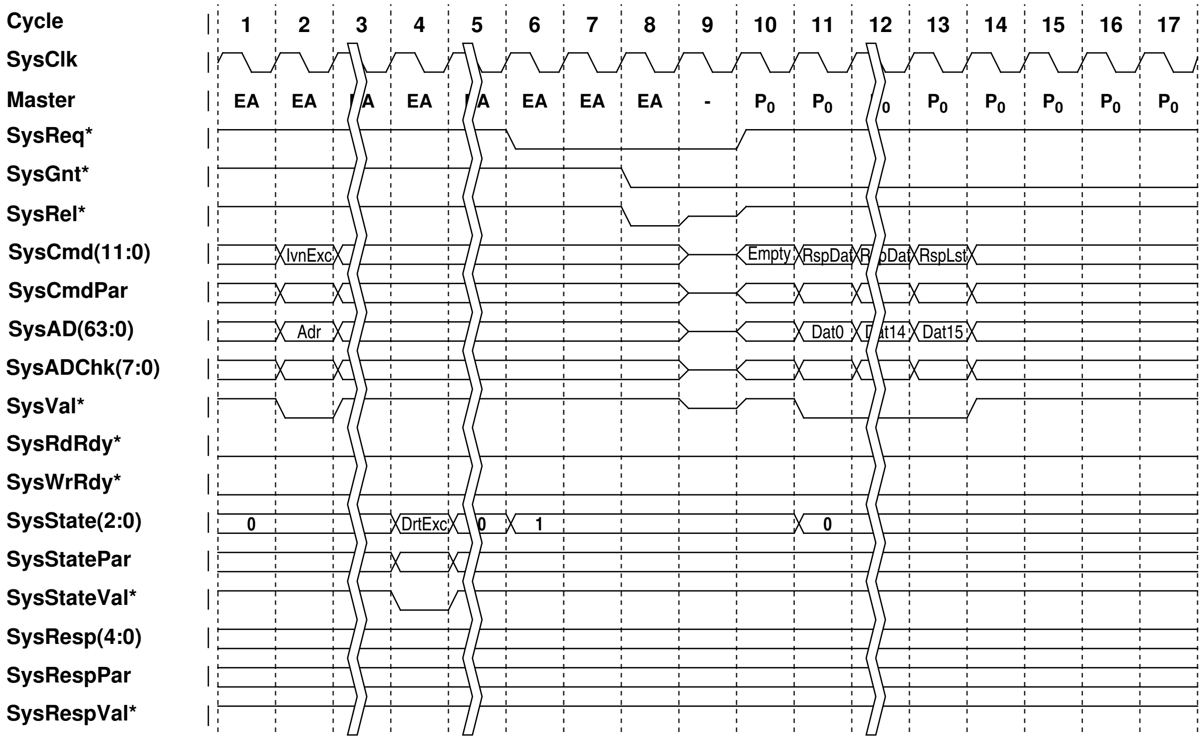

Figure 6-25 Processor Coherency Data Response Protocol

6.18 System Interface Coherency

The System interface supports external intervention shared, intervention exclusive, and invalidate coherency requests. These requests are used by an external agent or other R10000 processors on the cluster bus to maintain cache coherency.

Each external coherency request that targets an R10000 results in a processor coherency state response. Additionally, each external intervention request that targets the R10000 and hits a *DirtyExclusive* secondary cache block results in a processor coherency data response.

External coherency requests and the corresponding processor coherency state responses are handled in FIFO order.

External Intervention Shared Request

An external intervention shared request is used by an external agent to obtain a *Shared* copy of a cache block. If the desired block resides in the processor cache, it is marked *Shared*.

If the secondary cache block's former state was *DirtyExclusive*, the processor issues a processor coherency data response.

External Intervention Exclusive Request

An external intervention exclusive request is used by an external agent to obtain an *Exclusive* copy of a cache block. If the desired block resides in the processor cache, it is marked *Invalid*.

If the secondary cache block's former state was *DirtyExclusive*, the processor issues a processor coherency data response.

External Invalidate Request

An external invalidate request is used by an external agent to invalidate a cache block. If the desired block resides in the processor cache, it is marked *Invalid*.

Under normal circumstances, the secondary cache block former state should not be *CleanExclusive* or *DirtyExclusive*.

External Coherency Request Action

Table 6-27 indicates the action taken for external coherency requests that target the processor.

Table 6-27 Action Taken for External Coherency Requests that Target the R10000 Processor

Secondary Cache Block Former State	Type of External Request	Secondary Cache Block New State	Processor Coherency State Response SysState[1:0]	Processor Coherency Data Response Required?	Processor Coherency Data Response State SysCmd[2:1]
Invalid	Intervention shared	*Invalid*	0	No	N/A
	Intervention exclusive	*Invalid*	0	No	N/A
	Invalidate	*Invalid*	0	No	N/A
Shared	Intervention shared	*Shared*	1	No	N/A
	Intervention exclusive	*Invalid*	1	No	N/A
	Invalidate	*Invalid*	1	No	N/A
CleanExclusive	Intervention shared	*Shared*	2	No	N/A
	Intervention exclusive	*Invalid*	2	No	N/A
	Invalidate*	*Invalid*	2	No	N/A
DirtyExclusive	Intervention shared†	*Shared*	3	Yes	*Shared*
	Intervention exclusive	*Invalid*	3	Yes	*DirtyExclusive*
	Invalidate*	*Invalid*	3	No	N/A

* This should not occur under normal circumstances.

† The processor coherency data response must be written back to memory.

Coherency Conflicts

Coherency conflicts arise when a processor request and an external request target the same secondary cache block. Coherency conflicts may be categorized as either internal or external, and are described in this section.

Internal Coherency Conflicts

A processor request is considered to be pending issue when it is buffered in the processor and has not yet been issued to the System interface bus. Internal coherency conflicts occur when the processor has a processor request pending issue and a conflicting external coherency request is received. Internal coherency conflicts are unavoidable and cannot be anticipated by the external agent since it cannot anticipate when the processor will have processor requests pending issue.

Table 6-28 describes the manner in which the processor resolves internal coherency conflicts.

Table 6-28 Internal Coherency Conflict Resolution

Processor Request Pending Issue	Conflicting External Coherency Request	Resolution
Coherent block read	Intervention shared Intervention exclusive Invalidate	The processor allows the conflicting external coherency request to proceed and provides an *Invalid* processor coherency state response. The processor stalls the processor coherent block read request until the conflicting external coherency request has received an external completion response.
Upgrade	Intervention shared Intervention exclusive Invalidate	The processor allows the conflicting external coherency request to proceed and provides a *Shared* processor coherency state response. Once the conflicting external coherency request has received an external completion response, the processor internally NACKs the processor upgrade request that is pending issue.
Block write	Intervention shared Intervention exclusive	The processor provides a *DirtyExclusive* processor coherency state response and changes the processor block write request that is pending issue into a *DirtyExclusive* processor coherency data response.
	Invalidate	The processor provides a *DirtyExclusive* processor coherency state response and deletes the processor block write request that is pending issue.
Eliminate	Intervention shared Intervention exclusive Invalidate	The processor provides a *Shared* or *CleanExclusive* processor coherency state response and deletes the processor eliminate request that is pending issue.*

* If the processor eliminate request that is pending issue has a *DirtyExclusive* state, a *CleanExclusive* processor coherency state response is provided.

External Coherency Conflicts

A processor request is considered to be pending response when it has been issued to the System interface bus but has not yet received an external data or completion response. External coherency conflicts occur when the processor has a processor request that is pending response and a conflicting external coherency request is received. The processor relies on the external agent to detect and resolve external coherency conflicts.

External coherency conflicts may be avoided if the point of coherence is the processor System interface bus and only one request is allowed to be outstanding for any given secondary cache block. However, in some system designs external coherency conflicts are unavoidable.

Processor block write and eliminate requests are never pending response, and therefore cannot cause external coherency conflicts.

Table 6-29 describes the manner in which the external agent resolves external coherency conflicts.

Table 6-29 External Coherency Conflict Resolution

Processor Requests that are Pending Response	Conflicting External Coherency Request	Resolution
Coherent block read	Intervention shared	The external agent responds to the external coherency requestor that the block is *Invalid*. At some later time, the external agent supplies an external response to the processor coherent block read request that is pending response.[*]
	Intervention exclusive	
	Invalidate	
Upgrade	Intervention shared	The external agent responds to the external coherency requestor that the block is *Shared*. At some later time, the external agent supplies an external response to the processor upgrade request that is pending response.[*]
	Intervention exclusive	The external agent issues the conflicting external coherency request to the processor. The processor allows the conflicting external coherency request to proceed and supplies a *Shared* processor coherency state response. After observing the processor coherency state response, the external agent provides an external ACK completion response for the conflicting external coherency request. At some later time, the external agent supplies an external response for the processor upgrade request that is pending response. This external response may not be an external ACK completion response unless it is associated with an external block data response.
	Invalidate	

* The external agent may choose to issue the conflicting external coherency request to R10000, although it is not required.

External Coherency Request Latency

This section describes the R10000 external coherency request latency. Figure 6-26 depicts the following:

- an external coherency request which targets the processor
- the resulting processor coherency state response
- the potential processor coherency data response

Two external coherency request latency parameters are also defined:

- the processor coherency state response latency, t_{pcsr}, specifies the time from external coherency request to processor coherency state response
- the processor coherency data response latency, t_{pcdr}, specifies the time from the external coherency request to the processor coherency data response if a master, or to the assertion of the processor coherency data response indication on SysState[0] if a slave.

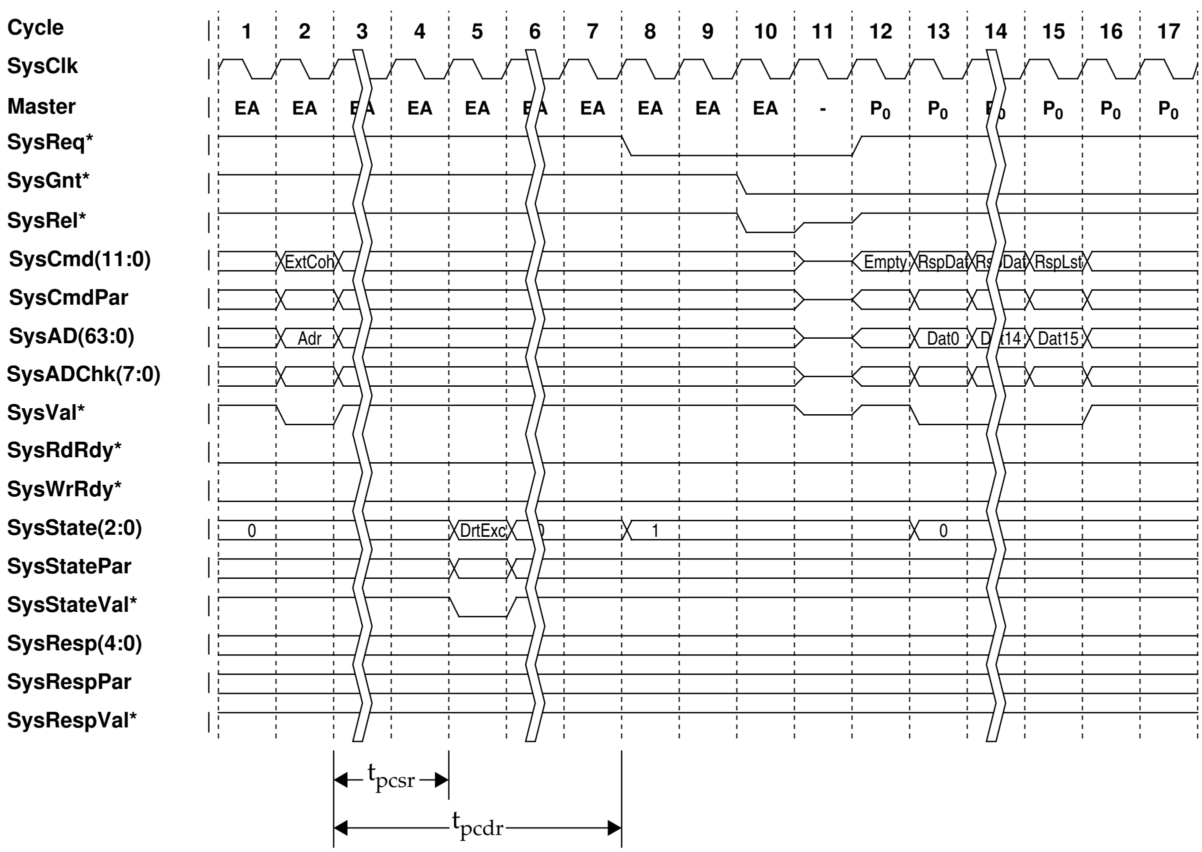

Figure 6-26 External Coherency Request Latency Parameters

The external coherency request latency is presented in Table 6-30.

Table 6-30 External Coherency Request Latency

	Latency * (PClk cycles)					
	Processor Coherency State Response (t_{pcsr})			Processor Coherency Data Response † (t_{pcdr})		
SCClkDiv	Min ‡	Typ ^	Max @	Min §	Typ ƒ	Max ¤
1	5	10	39	8	28	70
1.5	5	13	48	8	33	88
2	5	14	59	8	38	105
2.5	5	16	71	8	43	128
3	5	17	79	8	43	141

* This latency assumes no other previously issued external coherency requests are outstanding. 1 to 3 additional PClk cycles may be required for synchronization with SysClk depending on the SysClkDiv mode bits.

† This value assumes a 32-word secondary cache block size.

‡ This value assumes the external coherency request hits a cached or outgoing buffer entry.

^ This value assumes the external coherency request does not hit a cached or outgoing buffer entry, the secondary cache is not busy, and the external coherency request hits in the MRU way of the secondary cache. If the external coherency request misses in the most-recently used (MRU) way of the secondary cache, 1 to 3 additional PClk cycles are required to query the LRU way of the secondary cache, depending on the SCClkDiv mode bits.

@ This value assumes the external coherency request does not hit a cached or outgoing buffer entry, the secondary cache just commenced an index-conflicting CACHE Hit WriteBack Invalidate (S), and the external coherency request misses in the secondary cache MRU way.

§ This value assumes the external coherency request hits an outgoing buffer entry.

ƒ This value assumes the external coherency request does not hit a cached or outgoing buffer entry, the secondary cache is not busy, the external coherency request hits in the MRU way of the secondary cache, no subset primary data cache blocks are inconsistent, and the external coherency request is secondary cache block-aligned. If the external coherency request misses in the MRU way of the secondary cache, 1 to 3 additional PClk cycles are required to query the LRU way of the secondary cache, depending on the SCClkDiv mode bits.

¤ This value assumes the external coherency request does not hit a cached or outgoing buffer entry, the secondary cache just commenced an index-conflicting CACHE Hit WriteBack Invalidate (S), the external coherency request hits in the LRU way of the secondary cache, all subset primary data cache blocks are inconsistent, and the external coherency request is not secondary cache block-aligned.

SysGblPerf* Signal

The SysGblPerf* signal is provided for systems implementing a relaxed consistency memory model. The external agent asserts this signal when all processor requests are globally performed, thereby allowing the processor to graduate SYNC instructions. The external agent negates this signal when some processor requests are not yet globally performed, thereby preventing the processor from graduating SYNC instructions.

To prevent a SYNC instruction from graduating, the external agent must negate the SysGblPerf* signal no later than the same SysClk cycle in which it issued the external completion response for a processor read or upgrade request which is not yet globally performed. Also, the external agent must negate the SysGblPerf* signal no later than two SysClk cycles after the address cycle of a processor double/single/partial-word write request which has not yet been globally performed.

The SysGblPerf* signal may be permanently asserted in systems implementing a sequential consistency memory model.

6.19 Cluster Bus Operation

A R10000 multiprocessor cluster may be created by directly attaching the System interfaces of 2 to 4 R10000 processors, and providing an external cluster coordinator to handle arbitration and coherency management.

The cluster coordinator arbitrates the multiprocessors using the SysReq*, SysGnt*, and SysRel* signals.

A processor request issued by an R10000 processor in master state is observed as an external request by any R10000 processors in the slave state on the cluster bus. This is described Table 6-31.

Table 6-31 Relationship Between Processor and External Requests for the Cluster Bus

Processor Request	External Request
Coherent block read shared	Intervention shared
Coherent block read exclusive	Intervention exclusive
Noncoherent block read	Allocate request number
Double/single/partial-word read	Allocate request number
Block write	NOP
Double/single/partial-word write	NOP
Upgrade	Invalidate
Eliminate	NOP

In the same manner, a processor coherency data response issued by a processor in the master state is observed as an external block data response by any processors in the slave state.

External coherency requests that target a processor are handled in FIFO order and result in processor coherency state responses. If an external coherency request that targets a processor hits a *DirtyExclusive* secondary cache block, the processor also provides a processor coherency data response.

Figure 6-27 presents an example of a processor read request with four R10000 processors residing on the cluster bus. The CohPrcReqTar mode bit is asserted for a snoopy-based coherency protocol. $R10000_0$ issues a processor coherent read exclusive request. This is observed as an external intervention exclusive request by $R10000_1$, $R10000_2$, and $R10000_3$. $R10000_1$ and $R10000_3$ respond with *Invalid* processor coherency state responses. $R10000_2$ responds with a *DirtyExclusive* processor coherency state response. Based on these processor coherency state responses, the cluster coordinator allows $R10000_2$ to become master of the System interface so that it may provide a processor coherency data response, which will be observed as an external block data response by $R10000_0$. Finally, the cluster coordinator issues an external ACK completion response to forward the external block data response and to free the request number.

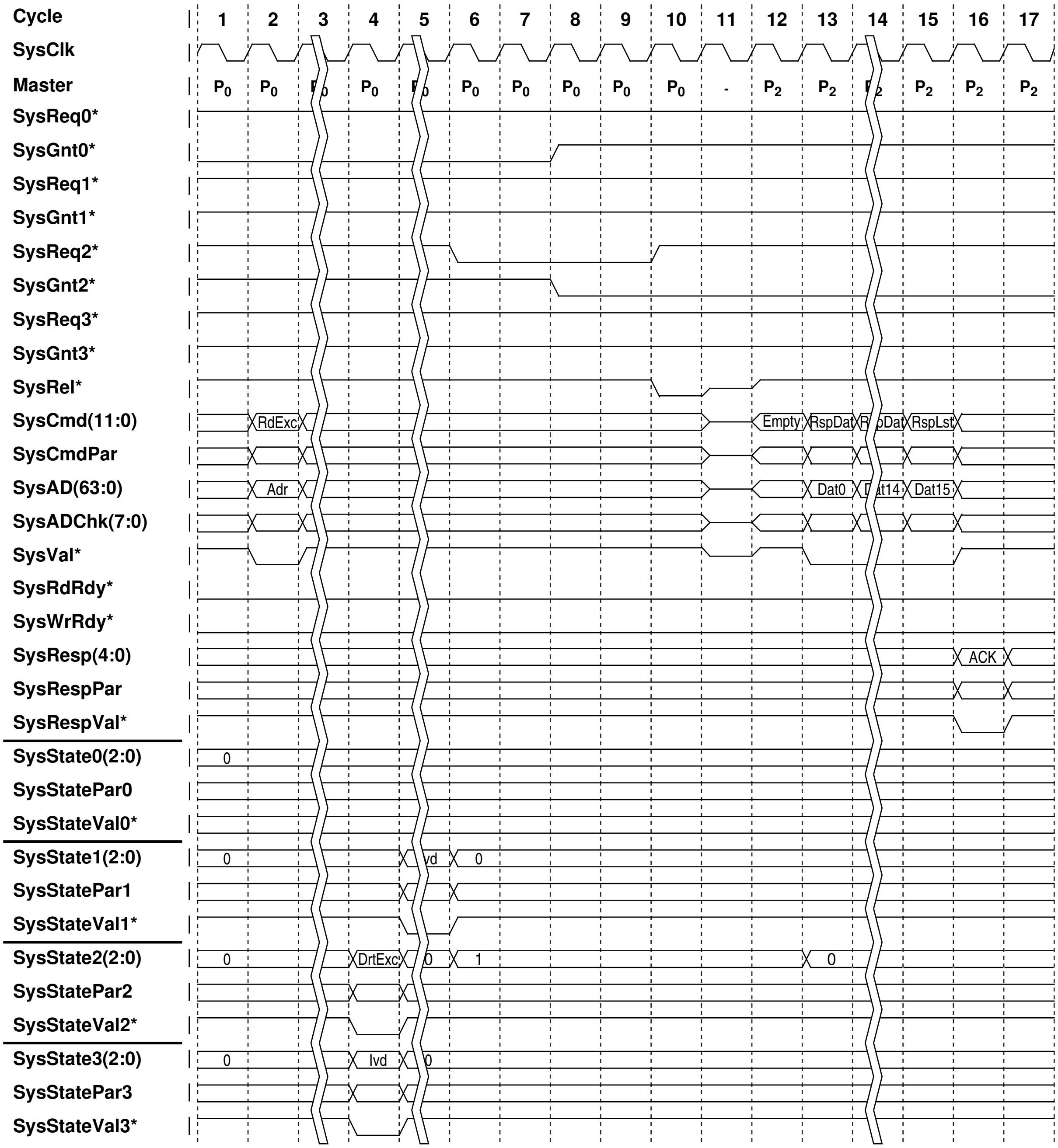

Figure 6-27 R10000 Multiprocessor Cluster Processor Read Request Example

Figure 6-28 presents an example of a processor upgrade request with four R10000 processors residing on the cluster bus. The CohPrcReqTar mode bit is asserted for a snoopy-based coherency protocol. $R10000_0$ issues a processor upgrade request, observed as an external invalidate request by $R10000_1$, $R10000_2$, and $R10000_3$. $R10000_2$ and $R10000_3$ provide *Shared* processor coherency state responses. $R10000_1$ provides an *Invalid* processor coherency state response. Based on these processor coherency state responses, the cluster coordinator issues an external ACK completion response for the processor upgrade request to indicate that the request was successful and to free the request number.

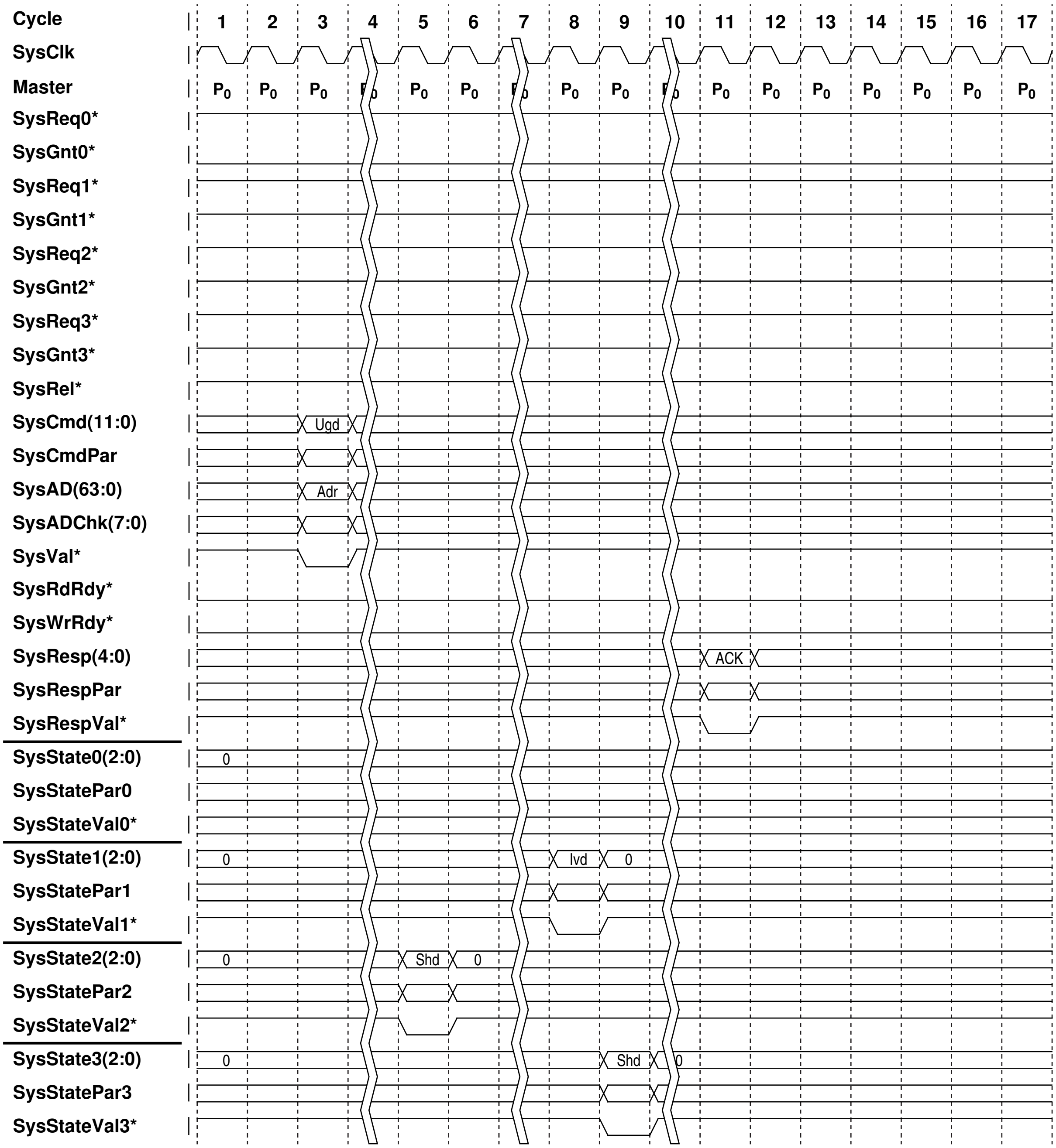

Figure 6-28 R10000 Multiprocessor Cluster Processor Upgrade Request Example

6.20 Support for I/O

The processor assumes a memory-mapped I/O model. Consequentially, no special System interface encodings are provided, or required to designate I/O accesses. It is left to the programmer to ensure that I/O addresses have the appropriate TLB mappings.

The processor supports system designs utilizing hardware or software for coherent I/O. The external coherency requests are useful for creating systems with hardware I/O coherency, and the CACHE instruction is sufficient for creating a system with software I/O coherency.

6.21 Support for External Duplicate Tags

Some system designs implement an external duplicate copy of the secondary cache tags to reduce the coherency request latency and also filter out unnecessary external coherency requests made to the R10000 processor.

For such systems, it must be remembered that blocks may reside in either the secondary cache or in the outgoing buffer. During the address cycle of processor block read requests, the secondary cache block former state is provided. The external agent may use this information to maintain the external duplicate tags.

Typically, in a multiprocessor system using the cluster bus, the cluster coordinator specifies a free request number for an external coherency request. However, in a system using a duplicate-tag or directory-based coherency protocol, where the CohPrcReqTar mode bit is negated, the cluster coordinator may specify a busy request number for an external coherency request, providing each targeted R10000 processor has the request number busy due to an outstanding processor coherency request from another processor.

For example, suppose the processor in master state issues a processor coherent block read or upgrade request. The processors in slave state observe the processor request as an external coherency request that targets the external agent only, causing the associated request number to become busy. The cluster coordinator checks the duplicate tag or directory structure to determine if the block resides in the cache of one of the processors that was in slave state. If necessary, the cluster coordinator issues an external coherency request targeted at one or more of the processors that were in slave state. By using the same request number as the original processor request, this external coherency request does not consume a free request number, and allows a potential processor coherency data response to be supplied as an external block data response to the original processor request.

6.22 Support for a Directory-Based Coherency Protocol

Some system designs implement a directory-based coherency protocol.

For such systems, the processor provides the processor eliminate request cycle. If the PrcElmReq mode bit is asserted, the processor issues a processor eliminate request whenever it intends to eliminate a *Shared*, *CleanExclusive*, or *DirtyExclusive* block from the secondary cache. During the address cycle of the processor eliminate request, the physical address and the secondary cache block former state are provided. The external agent may then use this information to maintain an external directory structure.

6.23 Support for Uncached Attribute

The processor supports a 2-bit user-defined *Uncached Attribute*, which is driven on SysAD[59:58] during the address cycle of the following:

- processor double/single/partial-word read requests
- double/single/partial-word write requests
- block write requests resulting from completely gathered uncached accelerated blocks

For unmapped accesses, the uncached attribute is sourced from VA[58:57].

For mapped accesses, the uncached attribute is sourced from the TLB *Uncached Attribute* field. The TLB *Uncached Attribute* field may be initialized in 64-bit mode using bits 63:62 of the CP0 *EntryLo0* and *EntryLo1* registers.

6.24 Support for Hardware Emulation

When using the R10000 processor in hardware emulation, it is desirable to operate the System interface at a relative low frequency (typically 1 MHz or below). Since the R10000 processor contains dynamic circuitry, an external agent cannot simply provide low frequency SysClk , so a SysCyc* input to the processor allows an external agent to define a virtual system clock, and yet supply a SysClk within the acceptable operating range. The assertion of SysCyc* in a particular SysClk cycle creates a virtual system clock pulse four SysClk cycles later. SysCyc * may be asserted aperiodically.

In a normal system environment, the SysCyc* input should be permanently asserted.

Figure 6-29 depicts the use of SysCyc* to create a virtual SysClk of one-third the normal SysClk frequency.

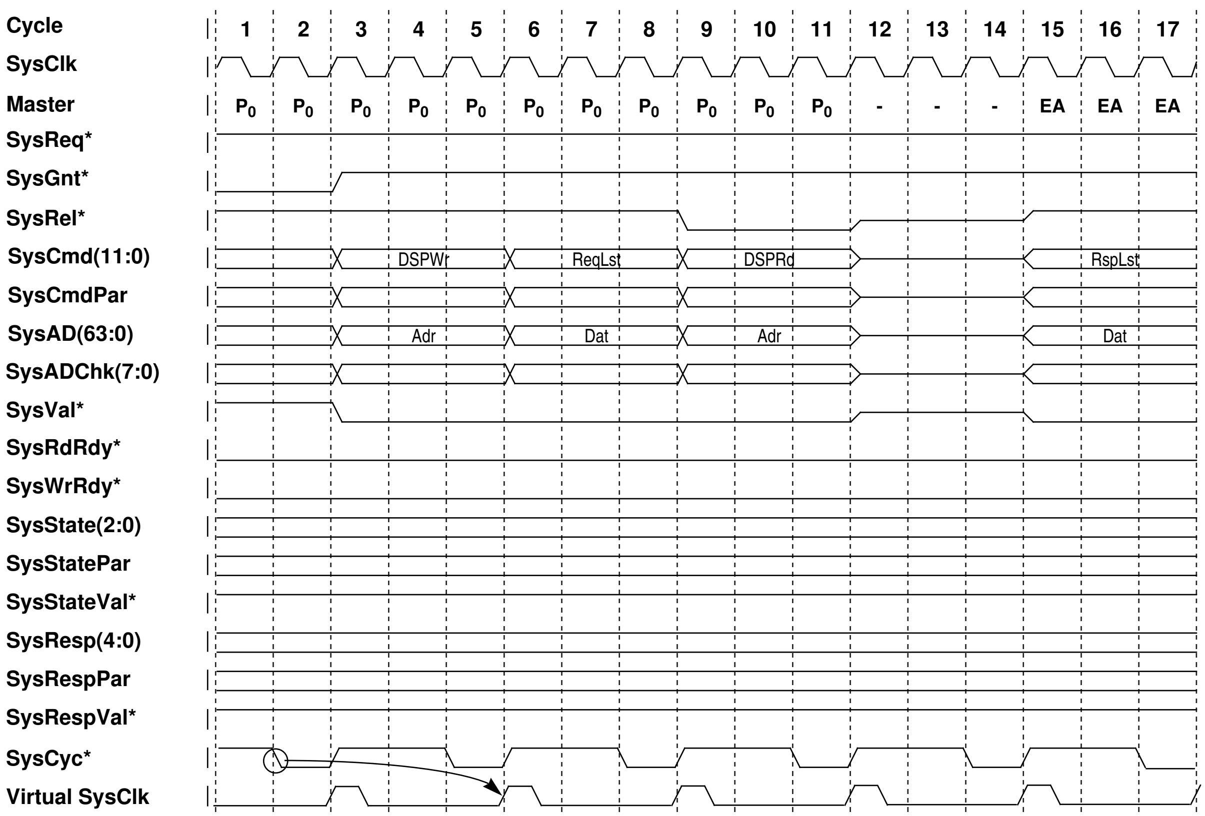

Figure 6-29 Hardware Emulation Protocol