HyperTransport™ Technology

Device Compatibility Checklists

Rev. 1.01 02/2003

The HyperTransport Technology Consortium disclaims all warranties and liability for the use of this document and the information contained herein and assumes no responsibility for any errors that may appear in this document, nor does the HyperTransport Technology Consortium make a commitment to update the information contained herein.

DISCLAMER

This document is provided "AS IS" with no warranties whatsoever, including any warranty of merchantability, noninfringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification or sample. The HyperTransport Technology Consortium disclaims all liability for infringement of property rights relating to the use of information in this document. No license, express, implied, by estoppel, or otherwise, to any intellectual property rights is granted herein.

TRADEMARKS

HyperTransport is a licensed trademark of the HyperTransport Technology Consortium.

Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

About HyperTransport™ Technology

HyperTransport technology is a high-speed, high-performance, point-to-point link for integrated circuits, and is designed to meet the bandwidth needs of tomorrow's computing and communications platforms. HyperTransport technology helps reduce the number of buses while providing a high-performance link for PCs, workstations, and servers, as well as numerous embedded applications and highly scalable multiprocessing systems. It is designed to allow chips inside of PCs, networking and communications devices to communicate with each other up to 48 times faster than with some existing bus technologies.

About the HyperTransport Technology Consortium

The HyperTransport Technology Consortium is a nonprofit corporation managed by its members. The consortium promotes the common business interests of providers to the networking, telecommunications, computer and high-performance embedded application through the conduct of a forum for the future development and adoption of the HyperTransport specification.

AMD, Apple Computers, Broadcom, Cisco Systems, NVIDIA, PMC-Sierra, SGI, SiPackets, Sun Microsystems, and Transmeta are the charter members that comprise the Executive Committee of the HyperTransport Technology Consortium.

Companies interested in the HyperTransport specification are invited to join the consortium. Members of the consortium pay annual dues and receive a royalty-free license to HyperTransport IP, gain access to technical documentation and may attend consortium meetings and events. To become a member, visit the consortium Web site at www.hypertransport.org. Please review the Bylaws of the HyperTransport Technology Consortium, and complete the online membership application.

Revision History: Device Compatibility Checklist

Rev	Date	Comment
1.01	02/2003	Language changes in sections 1 and 2. Document reformatted (separate section for rev 1.05 requirements).
1.00	01/2003	Checklist updated to be current with *HyperTransport™ I/O Link Protocol, Rev 1.05* requirements.
0.05	08/2002	Legal front matter updated; no change to document content.
0.04	08/2002	Pass/Fail columns added throughout.
0.03	07/2002	Compliance requirement descriptions revised. Document reformatted.
0.02	05/2002	Extensive document revision. Front matter legal review.
0.01	08/2001	Document created.

Contents

Preface

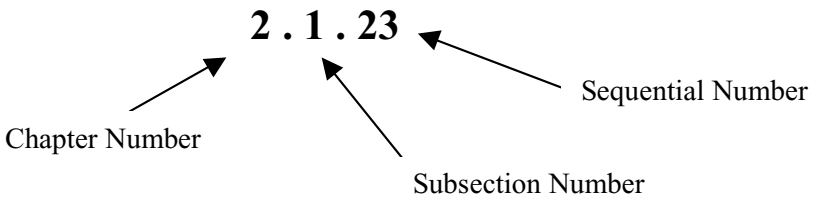
This document is a tool for expediting compatibility with the *HyperTransport™ I/O Link Specification.* This document is not a complete description of protocol requirements and cannot be used as a replacement for the specification. The requirements listed herein start with Rev 1.03 of the I/O link specification, the first rev officially released by the HyperTransport™ Consortium.

For each level of the specification, there is a corresponding chapter. Each chapter contains tables listing the compatibility requirements and the section of the specification where the requirement is found. For folks coding verification, each requirement is also assigned an ID for reference.

Starting with *rev 1.05* of the *HyperTransport™ I/O Link Specification* and for each released revision thereafter, a new section will be added to this document. In that section, any compatibility requirement changes will be listed using the same hierarchy from the main table of contents (Link Layer Requirements and subsections and Protocol Requirements and subsections); or a *No Change* will be flagged in the subsection if appropriate.

Compatibility Requirement ID Decode

Each compatibility requirement in this document is assigned an identifier given as chapter number followed by a period followed by the subsection number followed by a period followed by the sequential number within that subsection; as shown by this example:



1 Link Layer Requirements

1.1 Link: General

Requirement	Section	Requirement Description	Pass Yes/No
1.1.1	2	CAD width for each direction on each link is (independently) 2, 4, 8, 16, or 32 bits. One CLK is provided for each byte or fraction thereof. CTL is clocked by the CLK for the least-significant byte lane.	
1.1.2	2	RESET# and PWROK are always sampled as inputs, and reset all link and protocol state. Devices that drive them do so in an open-drain fashion to enable other drivers.	
1.1.3	3.1	With the exception of CRC Test Mode, CTL only changes on four-byte boundaries after link initialization.	
1.1.4	3.1	Transmitters may assert CTL between a header and its associated data, or during data transfer to insert data-less control packets. When CTL deasserts, data transmission continues from where it left off.	
1.1.5	3.2.1.1	Buffer release packets are only used for nearest-neighbor link interface communication. The receiver must always be able to sink them.	
1.1.6	4.3, 10.1	CRC generation is enabled from link initialization, and disabled by reset or sync flood. All devices must check CRC.	
1.1.7	4.3	Once a transmitter begins driving the sync pattern, it maintains the sync pattern until reset.	
1.1.8	4.8	All request command buffers can hold 8 bytes. All response command buffers can hold 4 bytes. All data buffers can hold 64 bytes.	
1.1.9	4.8	A transmitter cannot issue a command packet unless it has received a buffer release for the appropriate channel. A transmitter cannot issue a command packet with associated data unless it has received both command and data buffer credits for the appropriate channel.	
1.1.10	4.8	Transmit buffer counters are not allowed to overflow. If a transmitter receives more buffer releases of a particular type than it can track, it must discard the extras.	

Requirement	Section	Requirement Description	Pass Yes/No
1.1.11	4.8	Transmitted traffic is not allowed to starve transmission of buffer release messages, which would in turn starve the far transmitter.	
1.1.12	7.5.5, 12.2, 12.4	Links are independently sized by hardware in each direction at cold reset to the maximum width (up to 8 bits) supported by the transmitter and receiver. At warm reset, the link widths are set to the contents of the LinkConfiguration CSR.	
1.1.13	7.5.7, 11.4, 12.5	At cold reset, all links transmitters and receivers come up at a HyperTransport clock rate of 200 MHz. At warm reset, link frequencies are changed to match the contents of the LinkFrequency CSR.	
1.1.14	11.1	All devices must support sync mode clocking.	

1.2 Link: Host, Bridge Secondary Interface

Requirement	Section	Requirement Description	Pass Yes/No
1.2.1	2, 4.1.2	Bridges have separate RESET# and PWROK pins for their primary and secondary interfaces. Hosts may implement separate host and link RESET# and PWROK, but need not. Where 2 sets of pins exist, RESET# assertion and PWROK deassertion propagate asynchronously from primary/host side to secondary side. RESET# deassertion and PWROK assertion on the secondary side must always be driven with correct timing, regardless of timing on the primary/host side.	
1.2.2	12.2	PWROK must not be asserted until power and clocks have been stable for at least 1 ms.	
1.2.3	12.2	RESET# must be asserted at least 1 ms before the assertion of PWROK, and must remain asserted at least 1 ms following the assertion of PWROK.	
1.2.4	12.2	When asserted for a warm reset sequence, RESET# must remain asserted for at least 1 ms.	

1.3 Link: Disconnection (Optional Feature)

Requirement	Section	Requirement Description	Pass Yes/No
1.3.1	4.8	All buffer release fields in a disconnect NOP must be 0.	
1.3.2	8.3	Once asserted, LDTSTOP# must remain asserted for at least 1 us. Links will ignore the deassertion of LDTSTOP# until the link disconnect sequence has been completed on both their transmitter and receiver.	
1.3.3	8.3	Transmitters receiving a disconnect NOP must continue generating CRC until the end of the interval in which the disconnect NOP occurs.	
1.3.4	8.3, 12.2	LDTSTOP# must be deasserted at least 1 us before the deassertion of RESET#, and may not reassert until link initialization is complete.	

2 Protocol Requirements

2.1 Protocol: General

Requirement	Section	Requirement Description	Pass Yes/No
2.1.1	3.2.1, 4.2	Responses to downstream nonposted requests (UnitID = 0) are generated with Bridge = 0 and the UnitID of the responder.	
2.1.2	3.2.1, 4.2	Responses to upstream nonposted requests (UnitID = 0) are generated with Bridge = 1 and the UnitID of the original request.	
2.1.3	3.2.1	Reserved fields in generated packets must be 0 and must be ignored by receivers.	
2.1.4	3.2.1	All outstanding nonposted requests from a single UnitID must have distinct SrcTag values. A SrcTag may not be reused until a response for that request is received, or the link is reset.	
2.1.5	3.2.1.4	Receipt of a reserved command encoding is a protocol error. Receivers may either log and report it as a protocol error, or have undefined behavior.	

Requirement	Section	Requirement Description	Pass Yes/No
2.1.6	3.2.2, 4.4.1	• Sized byte reads must fall within a 4-byte aligned block. • Sized byte writes must fall within a 32-byte aligned block. • Sized DW reads and writes must fall within a 64-byte aligned block.	
2.1.7	4.4.1	Sized byte writes to spaces other than interrupt or system management space must have a nonzero count value. All byte writes count value must be at most 8.	
2.1.8	4.4.1, 4.4.5, 4.5.1	All WrSized, Atomic RMW, and RdResp headers are followed by N+1 DW of data, where N is the value of the Count field. All other packet types do not have associated data.	
2.1.9	4.4.1	Requests with the Compat bit set are always accepted by the compatibility device, and always forwarded by all other targets, regardless of address.	
2.1.10	4.4.1, 4.5.2	Responding agents generate TgtDone responses to nonposted WrSized requests. The Isoc bit from the request is returned in the response	
2.1.11	4.4.1, 4.5.1	Responding agents generate RdResp responses to RdSized requests. The Isoc bit from the request is returned in the response. The RespPassPW bit from the request is returned as the response's PassPW bit. If the read was a byte read, the Count in the response is 0; otherwise, the count field from the request is used.	
2.1.12	4.4.3, 4.5.2	Responding agents generate TgtDone responses to Flush requests. Isoc in the response is 0. PassPW in the response is 1.	
2.1.13	4.4.5, 4.5.1	Atomic RMW requests are always issued with a Count value of 1 or 3. They are always responded to by a RdResp response with a Count value of 1, Isoc = 0, and PassPW = 0.	
2.1.14	4.5.1	The data associated with RdResp responses with the Error and NXZ bits set is always all 1s.	
2.1.15	6.1, 6.3	Devices forwarding requests with the same UnitID and matching nonzero seqIds from the same source to the same (possibly non-HyperTransport) destination, must guarantee that their order is maintained.	
2.1.16	4.4.4, 6.1, 6.3	Devices forwarding a posted request followed by a fence or another request from the same UnitID with PassPW = 0 from the same source to the same and possibly non- HyperTransport destination must guarantee that their order is maintained.	

Requirement	Section	Requirement Description	Pass Yes/No
2.1.17	6.3	Devices forwarding a downstream posted request (UnitID = 0) followed by a downstream response (Bridge = 1) with PassPW = 0 from the same source to the same and possibly non-HyperTransport destination must guarantee their ordering is maintained.	
2.1.18	6.1, 6.4	Devices forwarding a posted request followed by an upstream response (Bridge = 0) with the same UnitID value and PassPW = 0 from the same source to the same and possibly non-HyperTransport destination must guarantee that their order is maintained.	
2.1.19	7.3.2.2, 7.4.4.1	Devices generating (not including host peer-to-peer reflection) response packets with the Error bit set and NXA bit clear set the Signaled Target Abort bit in the Status or Secondary Status CSR as appropriate.	
2.1.20	7.3.2.3, 7.4.4.2	Devices receiving (not including host peer-to-peer reflection) response packets with the Error bit set and NXA bit clear set the Received Target Abort bit in the Status or Secondary Status CSR as appropriate.	
2.1.21	7.3.2.4, 7.4.4.3	Devices receiving (not including host peer-to-peer reflection) response packets with the Error bit set and NXA bit clear set the Received Master Abort bit in the Status or Secondary Status CSR as appropriate.	
2.1.22	7.3.5, 7.4.5, 7.4.6	When comparing HyperTransport addresses to BARs or range registers, the smaller address must be 0 extended before the compare.	
2.1.23	9	Writes to Configuration or I/O space must be nonposted.	
2.1.24	9	Writes to Interrupt or System Management space must be posted.	
2.1.25	9	Accesses to IACK space must only be downstream reads.	

2.2 Protocol: Host, Bridge Secondary Interface

Requirement	Section	Requirement Description	Pass Yes/No
2.2.1	3.2.1, 4.2	Host downstream requests are always generated with a UnitID of 0.	
2.2.2	3.2.1	Hosts reflecting packets peer-to-peer preserve reserved fields.	
2.2.3	4.1.1, 4.9.4, 7.5.3.3	Hosts supporting non-sharing double-hosted chains must accept downstream requests from the far host to their CSRs, and must implement the DoubleEnded and ChainSide bits in the HyperTransport Command register.	
2.2.4	4.1.1	Hosts supporting sharing double-hosted chains must support all the features of non-sharing double-hosted chains, and additionally implement the Device Number and HostHide fields of the HyperTransport Command register.	
2.2.5	6.2	In addition to slave ordering requirements, hosts must perform cache invalidations for received writes before allowing subsequent ordered packets which may be visible to the processor (other writes, reads with side effects, interrupts) to be visible at their destinations.	
2.2.6	6.2.1	Hosts may not generate responses to nonposted requests until all side effects of the request are globally visible.	
2.2.7	7.4	The general reset for host CSRs comes from the primary reset of the device, not the HyperTransport chain reset.	
2.2.8	7.4.4.4	Hosts detecting sync flooding on the HyperTransport chain that was not initiated by the host set the System Error Detected bit in the Secondary Status CSR.	
2.2.9	7.5	All hosts implement a HyperTransport Host Capability Block.	

2.3 Protocol: Slave, Bridge Primary Interface

Requirement	Section	Requirement Description	Pass Yes/No
2.3.1	3.2.1, 4.2	Slave (upstream) requests are always generated with a nonzero UnitID.	
2.3.2	3.2.1	Tunnels forwarding packets preserve all fields, including reserved fields.	
2.3.3	4.1.1	Tunnels must keep track of which link requests targeting them were received from and transmit responses on the receiving link.	
2.3.4	4.9.1, 4.9.2	Slaves only accept downstream requests (UnitID = 0). Upstream requests are forwarded.	
2.3.5	4.9.1, 4.9.2	Slaves only accept downstream responses (Bridge = 1). Upstream responses are forwarded.	
2.3.6	4.4.2	Slaves are not allowed to issue Broadcast requests.	
2.3.7	4.7, 4.8	Slaves may not make accepting a request or issuing a response dependent upon their ability to issue an outgoing request, or upon the receipt of a response to an earlier issued request.	
2.3.8	4.9.3, 7.5.3.2.5, 7.5.3.3.8, 7.5.4.5, 7.5.4.6, 10.1.5	Posted request or response packets that are generated by a slave, or need to be forwarded by a tunnel, to a link that with its LinkCtrl/EndOfChain CSR bit set, or the LinkCtrl/InitializationComplete CSR bit clear while the HyperTransport Command/DropOnUninitializedLink CSR bit is set, are dropped.	
2.3.9	4.9.3, 7.5.3.2.5, 7.5.3.3.8, 7.5.4.5, 7.5.4.6	Nonposted request packets that are generated by a slave, or need to be forwarded by a tunnel, to a link that with its LinkCtrl/EndOfChain CSR bit set, or the LinkCtrl/InitializationComplete CSR bit clear while the HyperTransport Command/DropOnUninitializedLink CSR bit is set, are dropped, and a response is generated. The response is of the appropriate type to the original request, with all the expected fields. Its Error and NXA bits are both set.	
2.3.10	4.9.3, 7.5.3.2.5, 7.5.3.3.8, 7.5.4.5, 7.5.4.6	Request or response packets that are generated by a slave, or need to be forwarded by a tunnel to a link with its LinkCtrl/EndOfChain, LinkCtrl/InitializationComplete, and HyperTransport Command/DropOnUninitializedLink CSR bits clear are stalled until one of them sets.	
2.3.11	4.9.5	Tunnel devices must implement the HyperTransport fairness algorithm to control the bandwidth allocation between forwarded and inserted traffic.	

Requirement	Section	Requirement Description	Pass Yes/No
2.3.12	7.3, 7.4	All slave devices implement either a PCI Device or PCI Bridge configuration header.	
2.3.13	7.3.1.1, 7.3.1.2	Devices may only respond to Memory and I/O Space requests if the appropriate Space Enable bit is set in the Command CSR.	
2.3.14	7.3.1.3, 7.4.1	Devices may only issue Memory and I/O Space requests if the Bus Master Enable bit is set in the Command CSR.	
2.3.15	7.3.1.4, 7.3.2.5	Devices initiating sync floods (not propagating them from a link receiver) can only generate them if the SERR Enable bit is set in the Command CSR. When devices do initiate sync floods, they assert the Signaled System Error bit in the Status CSR.	
2.3.16	7.3.5, 9	BARs must be 64-byte aligned.	
2.3.17	7.5	All slaves implement a HyperTransport Slave Capability Block.	
2.3.18	7.5.3.2.1, 7.5.3.2.2	Slave devices implement configuration space at least for the device number given by the BaseUnitID field in the HyperTransport Command CSR. This device number must contain the HyperTransport Capability Block. The device may also implement one or more additional device configuration spaces for device numbers in the range BaseUnitID+1 through BaseUnitID+UnitCount-1, where UnitCount is also a field in the HyperTransport Command CSR.	
2.3.19	7.5.3.2.3, 7.5.3.2.4	Slave devices send generated requests to the link given by XORing the MasterHost and DefaultDirection bits of the HyperTransport Command CSR.	
2.3.20	7.6	All devices capable of generating HyperTransport interrupts must implement a HyperTransport Interrupt Discovery and Configuration Capability Block.	

3 Rev 1.05 Updates

Link Layer Requirements

3.1 Link:General

Requirement	Section	Requirement Description	Pass Yes/No
3.1.1	3.2.1.3, 4.4.6	Address Extension commands for 64-bit addressing support must immediately precede the affected request packet; only periodic CRC can be inserted between them.	

3.2 Link: Host, Bridge, Secondary Interface

No Change

3.3 Link: Disconnection (Optional Feature)

No Change

Protocol Requirements

3.4 Protocol: General

Requirement	Section	Requirement Description	Pass Yes/No
3.4.1	3.2.1, 4.2	Responses to downstream nonposted requests (UnitID = 0 or is clumped to 0) are generated with Bridge = 0 and the UnitID of the responder.	
3.4..2	3.2.1, 4.2	Responses to upstream nonposted requests (UnitID > 0 or is not clumped to 0) are generated with Bridge = 1 and the UnitID of the original request.	

Requirement	Section	Requirement Description	Pass Yes/No
3.4.3	3.2.1.5	Read and write requests that don't target memory must set the Coherent bit.	
3.4.4	4.4.1	A series of posted requests being forwarded or inserted with the Chain bit set must not have other posted requests injected among them.	
3.4.5	4.4.1, 4.5.2	Responding agents generate TgtDone responses to nonposted WrSized requests. The Isoc bit from the request is returned in the response. Responses to downstream requests carry the two least significant bits of the requestor's UnitID in RqUID.	
3.4.6	4.4.1, 4.5.1	Responding agents generate RdResp responses to RdSized requests. The Isoc bit from the request is returned in the response. The RespPassPW bit from the request is returned as the response's PassPW bit. If the read was a byte read, the Count in the response is 0; otherwise the count field from the request is used. Responses to downstream requests carry the two least significant bits of the requestor's UnitID in the RqUID.	
3.4.7	4.4.5, 4.5.1	Atomic RMW requests are typically issued with a Count value of 1 or 3. They are always responded to by a RdResp response with a Count value of 1, Isoc = 0, and PassPW = 0.	
3.4.8	4.5.1	The data associated with RdResp responses with Master Abort error encoding is always all 1s.	
3.4.9	4.6.1	HyperTransport rev 1.05 compatible devices must implement either the UnitID Reorder bit (see specification section 7.5.10.6) or a UnitID Clumping capability.	
3.4.10	7.3.2.2, 7.4.4.1	Devices generating (not including host peer-to-peer reflection) response packets with Target Abort error encoding set the Signaled Target Abort bit in the Status or Secondary Status CSR as appropriate.	
3.4.11	7.3.2.3, 7.4.4.2	Devices receiving (not including host peer-to-peer reflection) response packets with Target Abort error encoding set the Received Target Abort bit in the Status or Secondary Status CSR as appropriate.	
3.4.12	7.3.2.4, 7.4.4.3	Devices receiving (not including host peer-to-peer reflection) response packets with Master Abort error encoding set the Received Master Abort bit in the Status or Secondary Status CSR as appropriate.	

3.5 Protocol: Host, Bridge, Secondary Interface

Requirement	Section	Requirement Description	Pass Yes/No
3.5.1	3.2.1, 4.2	Host downstream requests are always generated with a Clumped UnitID of 0.	

3.6 Protocol: Slave, Bridge Primary Interface

Requirement	Section	Requirement Description	Pass Yes/No
3.6.1	4.9.1, 4.9.2	Slaves only accept downstream requests (UnitID = 0 or is clumped to 0). Upstream requests are forwarded.	
3.6.2	4.7, 4.8	Slaves may not make accepting a posted request dependent upon their ability to issue an outgoing request nor make acceptance of a nonposted request dependent on their ability to issue an outgoing nonposted request.	