PCI-X Addendum to the PCI Compliance Checklist

Revision 1.0a

August 29, 2000
REVISION | REVISION HISTORY                      | DATE     
--- | --- | --- 
1.0  | Initial Release                      | 3/1/00   
1.0a | Updates for PCI-X Addendum 1.0a      | 8/29/00  

The PCI Special Interest Group 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 PCI Special Interest Group make a commitment to update the information contained herein.

Contact the PCI Special Interest Group office to obtain the latest revision of this checklist.

Questions regarding the this document or membership in the PCI Special Interest Group may be forwarded to:

PCI Special Interest Group  
5440 SW Westgate Drive #217  
Portland, OR 97221  
Phone: 503-291-2569  
Fax: 503-297-1090  
e-mail Administration@pcisig.com  
http://www.pcisig.com

DISCLAIMER

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 PCI SIG disclaims all liability for infringement of proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

All product names are trademarks, registered trademarks, or servicemarks of their respective owners.

Copyright © 1999, 2000 PCI Special Interest Group
Contents

1. INTRODUCTION..............................................................................................................4
2. GENERAL PCI-X PROTOCOL CHECKLIST (XGP)..........................................................5
3. PCI-X INITIATOR CHECKLIST (XIP) ..............................................................................7
4. PCI-X TARGET CHECKLIST (XTP) ...............................................................................9
5. PCI-X SIMPLE DEVICE CHECKLIST (XSP)...............................................................11
6. PCI-X BUS ARBITRATION CHECKLIST (XAP)............................................................12
7. PCI-X CONFIGURATION CHECKLIST (XCP)..............................................................14
8. PCI-X ERROR CHECKLIST (XEP)..................................................................................15
9. PCI-X BUS WIDTH CHECKLIST (XWP).........................................................................17
10. PCI-X SPLIT TRANSACTION CHECKLIST (XST).....................................................19
11. INTEROPERABILITY AND INITIALIZATION CHECKS (XIN).......................................21
12. PCI-X BRIDGE CHECKLIST (XBP)..............................................................................22
13. PCI-X SYSTEM ELECTRICAL CHECKLIST (XME)..................................................23
14. PCI-X ADD-IN CARD ELECTRICAL CHECKLIST (XAE)...........................................24
1. Introduction

This document provides checklists for PCI-X devices, in addition to the checklists defined in the *PCI Compliance Checklist, Revision 2.1*.

The requirements listed in this document are provided as an aid in designing and validating PCI-X devices. These lists are not comprehensive. This document includes only a summary of some of the requirements of the *PCI-X Addendum to the PCI Local Bus Specification, Revision 1.0a* (PCI-X 1.0a). In case of discrepancy between this document and PCI-X 1.0a, PCI-X 1.0a governs. PCI-X devices must meet all of the requirements of PCI-X 1.0a whether or not those requirements are repeated in this document.
2. General PCI-X Protocol Checklist (XGP)

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance.

XGP1. The starting address of Configuration Read and Configuration Write transactions is aligned to a DWORD boundary.  

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP2. For I/O and Memory Read DWORD transactions, the starting address must correspond to the first enabled byte (unless no byte enables are asserted).

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
<th>N/A</th>
</tr>
</thead>
</table>

XGP3. During the attribute phase $C/BE[3::0]#$ and $AD[31::00]$ contain the attributes.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP4. During the attribute phase $C/BE[7::4]#$ and $AD[63::32]$ are reserved and driven high by 64-bit initiators.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
<th>N/A</th>
</tr>
</thead>
</table>

XGP5. The $C/BE#$ bus is reserved (driven high) the clock after the attribute phase.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP6. All burst transactions include the byte count in the attributes.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP7. The byte count always indicates the number of bytes from the first byte of the transaction to the last byte of the Sequence, inclusive.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP8. DWORD transactions do not use a byte count.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP9. Transactions using the I/O Read, I/O Write, Configuration Read, Configuration Write, Interrupt Acknowledge, Special Cycle, and Memory Read DWORD commands are initiated only as 32-bit transactions (REQ64# deasserted).

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
<th>N/A</th>
</tr>
</thead>
</table>

XGP10. Transactions using the I/O Read, I/O Write, Configuration Read, Configuration Write, Interrupt Acknowledge, Special Cycle, and Memory Read DWORD commands and Split Completion Messages are limited to a single data phase.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
<th>N/A</th>
</tr>
</thead>
</table>

XGP11. Byte enables are included in the Requester Attributes for all DWORD transactions. Only bytes for which the byte enable is asserted are affected by the transaction.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP12. For DWORD transactions and Memory Write burst transactions only bytes for which the byte enable is asserted are affected by the transaction.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP13. The $C/BE#$ bus is reserved and driven high during the single data phase of all DWORD transactions, and throughout all data phases of all burst transactions except Memory Write.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP14. DWORD transactions are permitted to have any combination of byte enables, including no byte enables asserted.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

XGP15. Memory Write transactions are permitted to have any combination of byte enables between the starting and ending addresses, inclusive.

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>
XGP16. Byte enables are deasserted for bytes before the starting address and after the ending address (if those addresses are not aligned to the width of the bus), except for Memory Write transactions when a 64-bit initiator’s starting address is in the high 32-bits of the 64-bit bus. In that case $C/BE[7::4]#$ are copied to $C/BE[3::0]#$.  

yes ___  no___

XGP17. For Memory Write transactions the byte count is not affected by whether byte enables are asserted. 

yes ___  no___

XGP18. Device state machines are not affected by the states of DEVSEL#, TRDY#, and STOP# while the bus is idle (FRAME# and IRDY# both deasserted). 

yes ___  no___

XGP19. If the device snoops a transaction, it does not drive any signals during the transaction (same as conventional PCI). 

yes ___  no___  N/A___

XGP20. The device does not drive and receive bus signals at the same time. 

yes ___  no___

XGP21. If the device generates interrupts, it supports message signaled interrupts and supports a 64-bit message address. If the device will be utilized in systems that do not support message signaled interrupts, it also implements hardware interrupts. 

yes ___  no___  N/A___

XGP22. If the device is intended for use on add-in cards, it supports PCI Power Management, as defined in PCI PM 1.1. 

yes ___  no___  N/A___

XGP23. If the device supports power management and it is in D3hot state but RST# remains deasserted, the device maintains its frequency and mode information (from the PCI-X initialization pattern). 

yes ___  no___  N/A___

XGP24. If the system supports PCI power management, it permits devices in D1, D2, and D3hot to signal Split Response to a configuration transaction and initiate the corresponding split completion transaction. (System requirement.) 

yes ___  no___  N/A___

XGP25. Fast back-to-back transactions are not initiated by the device. 

yes ___  no___  N/A___

XGP26. DEVSEL# timing is measured from the address phase if the transaction uses a single address cycle and from the second address phase if the transaction uses a dual address cycle. 

yes ___  no___
3. PCI-X Initiator Checklist (XIP)

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. This checklist applies to all initiator operations.

XIP1. The initiator begins a transaction (other than a configuration transactions) by asserting FRAME#. yes ___ no___

XIP2. The initiator asserts FRAME# or deasserts REQ# within two to six clocks after GNT# is asserted and the bus is idle. yes ___ no___

XIP3. The initiator deasserts FRAME# on the later of the following three conditions:
   1. One clock before the last data phase.
   2. Two clocks after the target asserts TRDY# (or terminates the transaction in some other way).
   3. Eight clocks after the address phase if no target asserts DEVSEL# (Master-Abort). yes ___ no___

XIP4. The initiator asserts IRDY# two clocks after the attribute phase. yes ___ no___

XIP5. The initiator deasserts IRDY# on the later of the following three conditions:
   1. One clock after the last data phase.
   2. Two clocks after the target asserts TRDY# (or terminates the transaction in some other way).
   3. Eight clocks after the address phase if no target asserts DEVSEL# (Master-Abort). yes ___ no___

XIP6. The initiator uses the full address bus to indicate the starting address (including AD[1::0]) of memory and I/O transactions. Note this is the same as conventional I/O transactions but not conventional memory transactions. yes ___ no___

XIP7. The initiator does not initiate a new Sequence using the same Tag until the previous Sequence is complete. yes ___ no___

XIP8. If no target asserts DEVSEL# on or before the Subtractive decode time, the initiator ends the transaction as a Master-Abort. yes ___ no___

XIP9. For write and Split Completion transactions, the initiator must drive data on the AD bus two clocks after the attribute phase. yes ___ no___ N/A___

XIP10. For write and Split Completion transactions, if the transaction is a burst with more than one data phase, the initiator advances to the second data value two clocks after the target asserts DEVSEL#. yes ___ no___ N/A___

XIP11. For write and Split Completion transactions, if the transaction is a burst with more than one data phase and the target inserts wait states, the initiator must toggle between its first and second data values until the target asserts TRDY# (or terminates the transaction). yes ___ no___ N/A___
XIP12. The initiator terminates the transaction when the byte count is satisfied. yes ___ no___

XIP13. Burst transactions do not cross the end of the 64-bit address space. yes ___ no___

XIP14. If a target responds with immediate completion to a burst transaction for more than a single data phase, the initiator does not signal a disconnect at any point except an ADB. (The Sequence terminates when the byte count expires.) yes ___ no___

XIP15. If the initiator intends to disconnect the transaction on the first ADB, and the first ADB is less than four data phases from the starting address, the initiator adjusts the byte count to terminate the transaction on that ADB. yes ___ no___

XIP16. For a burst transaction that would cross the next ADB if the target signals Disconnect at Next ADB four data phases before an ADB or on the first data phase, the initiator deasserts FRAME# two clocks later and disconnects the transaction on the ADB. yes ___ no___

XIP17. The initiator transfers data on each data phase in which the target signals Disconnect at Next ADB the same as if the target had signaled Data Transfer. yes ___ no___

XIP18. The default Latency Timer value for the initiator in PCI-X mode is 64. yes ___ no___

XIP19. The initiator disconnects the current transaction on the next ADB if the Latency Timer expires and GNT# is deasserted at least four data phases before the ADB. yes ___ no___

XIP20. The initiator does not repeat a transaction that the target terminated by signaling Target-Abort. (Same as conventional PCI.) yes ___ no___

XIP21. The device does not set the No Snoop and Relaxed Order attribute bits on any transaction that is not a memory transaction. yes ___ no___

XIP22. If the device initiates Special Cycle transactions, those transactions have no initiator wait states, have only one data phase, and have all byte enables asserted. yes ___ no___ N/A___

XIP23. If the device initiates Special Cycle transaction, those transactions do not set the Received Master-Abort bit in the Status register. yes ___ no___ N/A___

XIP24. The initiator does not use reserved commands (0100b, 0101b, 1000b, and 1001b). yes ___ no___
4. PCI-X Target Checklist (XTP)

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. This checklist applies to all target operations.

XTP1. Memory address ranges (including those assigned through Base Address registers) assigned to the device are no smaller than 128 bytes. yes ___ no___ N/A___

XTP2. The target ignores (must not assert DEVSEL# or change state) any transactions using a reserved command. yes ___ no___

XTP3. The target treats the Alias to Memory Read Block command (1000b) as if it were a Memory Read Block (1110b) and Alias to Memory Write Block command (1001b) as if it were a Memory Write Block (1111b). yes ___ no___

XTP4. The target claims the transaction by asserting DEVSEL# and leaving TRDY# and STOP# deasserted, using decodes A, B, C or Subtractive. yes ___ no___

XTP5. DEVSEL# is never asserted earlier than the clock after the attribute phase. yes ___ no___

XTP6. After the target asserts DEVSEL#, it completes the transaction with one or more data phases by signaling one or more of the following: Split Response, Target-Abort, Single Data Phase Disconnect, Wait State, Data Transfer, Retry, Disconnect at Next ADB. yes ___ no___

XTP7. The target never inserts an odd number of wait states for burst write and Split Completion transactions. yes ___ no___

XTP8. The target does not signal Wait State after the first data phase. yes ___ no___

XTP9. If the target signals Split Response, or Retry, or signals Target-Abort in the first data phase, the target does so within eight clocks of the assertion of FRAME#. yes ___ no___ N/A___

XTP10. If the target signals Single Data Phase Disconnect, or signals Data Transfer or Disconnect at Next ADB in the first data phase, the target does so within 16 clocks of the assertion of FRAME# (except during device initialization time, $T_{rhfa}$, and when the system is copying an expansion ROM image from the device to system memory). yes ___ no___

XTP11. If the PCI-X target signals Data Transfer (with or without preceding wait states), the target disconnects the transaction only on an ADB (until the byte count is satisfied). yes ___ no___

XTP12. If the target has signaled Disconnect at Next ADB, it continues to do so (or signals Target-Abort) until the end of the transaction yes ___ no___ N/A___

XTP13. If the target signals Split Response, Single Data Phase Disconnect or Retry, it does so only on the first data phase (with or without preceding wait states). yes ___ no___ N/A___
XTP14. The target never signals Split Response on burst write or Split Completion transactions.  yes ___ no___

XTP15. The target deasserts DEVSEL#, STOP#, and TRDY# one clock after the last data phase (if they are not already deasserted) and floats them one clock after that.  yes ___ no___

XTP16. The target does not store state information about any type of transaction other than Split Requests. (The target does not perform Delayed Transactions in PCI-X mode.)  yes ___ no___

XTP17. If the target signals Disconnect at Next ADB less than 4 data phases from an ADB but not on the first data phase, the target expects the transaction to continue past the first ADB and disconnect on the next ADB or until the byte count expires.  yes ___ no___

XTP18. If the target signals Disconnect at Next ADB 4 or more data phases from an ADB, the target expects the transaction to disconnect at that ADB.  yes ___ no___

XTP19. If the device responds to Special Cycle transactions, the device does not assert bus signals in response to detecting a Special Cycle transaction.  yes ___ no___ N/A___

XTP20. If the device responds to Interrupt Acknowledge transactions, the device responds to the bus transaction regardless of the address value.  yes ___ no___ N/A___

XTP21. If the target is a bridge, it responds to Split Completions only with Target-Abort, Wait State, Data Transfer, Retry, and Disconnect at Next ADB. If the target is not a bridge, it responds to Split Completions only with Target-Abort, Wait State, and Data Transfer.  yes ___ no___

XTP22. If the target is designed to signal Single Data Phase Disconnect for any memory write transaction with an address four or less data phases before an ADB, the target never signals Data Transfer for any other memory write transaction that begins four or less data phases from that same ADB.  yes ___ no___ N/A___
5. PCI-X Simple Device Checklist (XSP)

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. The checklist applies to all simple devices, as defined in PCI-X 1.0a.

**XSP1.** After initialization time, the device is able to accept a memory and I/O write transaction within the Maximum Completion Time limit of 2us (267 clocks for 133 MHz mode, 200 clocks for 100 MHz mode, and 133 clocks for 66 MHz mode).

**XSP2.** After initialization time, the device accepts memory write transactions even while executing a previous Split Transaction.

**XSP3.** After initialization time, the device accepts all Split Completion transactions that correspond to the device’s outstanding Split Requests without signaling Retry or Disconnect at Next ADB.
6. PCI-X Bus Arbitration Checklist (XAP)

The following checklist applies to bus arbitration. Some rules apply to all initiators and some apply only if the device includes the bus arbiter, as indicated below.

_The following items apply to all initiators. Check here if the device is not an initiator._

**XAP1.** If a device signals Split Response, arbitration within that device fairly allows the initiation of the Split Completion.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
<th>N/A</th>
</tr>
</thead>
</table>

**XAP2.** REQ# and GNT# signals are registered by the initiator.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

**XAP3.** The initiator starts a new transaction (drive the AD bus, etc.) on any clock N only if the initiator’s GNT# was asserted on clock N-2, and either the bus was idle (FRAME# and IRDY# were both deasserted) on clock N-2 or FRAME# was deasserted and IRDY# was asserted on clock N-3.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

**XAP4.** If the device includes multiple sources of initiator activity, all of these sources share a single REQ# and GNT# pair.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
<th>N/A</th>
</tr>
</thead>
</table>

**XAP5.** If GNT# is asserted and the bus is idle for four consecutive clocks, the device (where the bus is parked) actively drives the bus (AD, C/BE#, PAR, and (if a 64-bit device) PAR64) no later than the sixth clock.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

**XAP6.** A parked initiator must assert REQ# if it intends to execute more than a single transaction.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

**XAP7.** If GNT# is deasserted when the Latency Timer expires, the device disconnects the current transaction as soon as possible. (This is typically the next ADB, but in some cases is the following ADB or when the byte count expires.)

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

_The following items apply only if the device includes the bus arbiter. Check here if the device does not include the arbiter._

**XAP8.** The arbiter is fair to all devices.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

**XAP9.** All REQ# and GNT# signals are registered by the arbiter.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

**XAP10.** The arbitration algorithm does not depend on the initiator deasserting REQ# after a target termination (STOP# asserted).

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

**XAP11.** The arbitration algorithm allows the initiator to deassert REQ# on any clock independent of whether GNT# is asserted.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>

**XAP12.** The arbitration algorithm allows the initiator to deassert REQ# without initiating a transaction after GNT# is asserted.

<table>
<thead>
<tr>
<th>yes</th>
<th>no</th>
</tr>
</thead>
</table>
XAP13. The arbiter allows each device a fair opportunity to execute a configuration transaction. To allow a device to execute a configuration transaction, after the arbiter asserts $GNT\#$, the arbiter keeps it asserted for a minimum of five clocks while the bus is idle, or until the initiator asserts $FRAME\#$ or deasserts $REQ\#$. (Note that this rule does not require that every assertion of $GNT\#$ must be for a minimum of five clocks. Only that every device must eventually have a turn to execute a configuration transaction.)

XAP14. If the arbiter deasserts $GNT\#$ to one device, it waits until the next clock to assert $GNT\#$ to another device.

XAP15. In PCI hot-plug systems, the arbiter coordinates with the Hot-Plug Controller to prevent hot-plug operations from interfering with other bus transactions.

XAP16. If no initiators request the bus, the arbiter parks the bus at a device that is capable of being an initiator.
# 7. PCI-X Configuration Checklist (XCP)

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. This checklist applies to all configuration transactions. Rules designated for initiators do not apply to devices that do not initiate configuration transactions. Rules designated for targets do not apply to devices such as host bridges that are never addressed by a configuration transaction.

<table>
<thead>
<tr>
<th>XCP</th>
<th>Description</th>
<th>Yes</th>
<th>No</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCP1</td>
<td>Initiator drives the address for four clocks before asserting FRAME# for configuration transactions when in PCI-X mode.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP2</td>
<td>Initiator includes in the transaction the target device number in AD[15::11] of the address phase.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP3</td>
<td>Initiator includes in the transaction the target bus number in AD[7::0] of the attribute phase.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP4</td>
<td>If the device is not a PCI-X bridge and it is the target of a Type 0 Configuration Write transaction, it stores its device number (from the address phase) and bus number (from the attribute phase) in its PCI-X Status registers. If the device is a PCI-X bridge and it is the target of a Type 0 Configuration Write transaction, it stores its device number (from the address phase) in its PCI-X Status registers.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP5</td>
<td>If the device is not a PCI-X bridge, reading the device’s PCI-X Status register following PCI Reset (before writing to the device) returns 1Fh for the Device Number field and FFh for the Bus Number field.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP6</td>
<td>If the device is not a bridge, it does not respond to Type 1 Configuration transactions.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP7</td>
<td>If the arbiter asserts GNT# to a device on clock N-2, the device starts driving the bus for a configuration transaction (on clock N, N+1, or N+2), and the arbiter deasserts GNT# before clock N+3, the device does not continue the configuration transaction. It floats the bus two clocks after GNT# is deasserted.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP8</td>
<td>The system quiesces all devices on a bus segment before changing the Bus Number in the source bridge. (System requirement)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP9</td>
<td>The system executes a write to the configuration space of each device after the bus segment’s Bus Number has been changed. (System requirement)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP10</td>
<td>All PCI-X devices must have the Capabilities List bit (bit 4) of the Status register set to 1.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP11</td>
<td>All PCI-X devices support the PCI-X Capabilities List Item.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XCP12</td>
<td>When in PCI-X mode, the Fast Back-to-Back bit (bit 7) of the Status register must be set to 0.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
8. PCI-X Error Checklist (XEP)

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. Rules designated for initiators do not apply to devices that do not initiate transactions. Rules designated for targets do not apply to devices that do not check parity.

XEP1. The device asserts \( \text{PERR#} \) (if enabled) on the second clock after \( \text{PAR64} \) and \( \text{PAR} \) are driven if the device is receiving data (i.e., the target of a write or Split Completion and the initiator of a read) and detects a data parity error.

- yes ___  no___ N/A___

XEP2. During read transactions, the target drives \( \text{PAR64} \) (if responding as a 64-bit device) and \( \text{PAR} \) on clock \( N+1 \) for the read data it drives on clock \( N \) and the \( \text{C/BE#} \) bus driven by the initiator on clock \( N-1 \).

- yes ___  no___ N/A___

XEP3. During write and Split Completion transactions, the initiator drives \( \text{PAR64} \) (if initiating as a 64-bit device) and \( \text{PAR} \) on clock \( N+1 \) for the write data and the \( \text{C/BE#} \) bus it drives on clock \( N \).

- yes ___  no___ N/A___

XEP4. The device services data parity error conditions for transactions it initiates in one of the following ways:

- The device and software driver attempt to recover from a data parity error condition.
- The device asserts \( \text{SERR#} \) (if enabled) on a data parity error condition.

- a. ___  b. ___ N/A___

XEP5. The device asserts \( \text{SERR#} \) (if enabled) if it detects a parity error on an attribute phase, independent of whether the device decodes its address during the address phase.

- yes ___  no___ N/A___

XEP6. For Split Transactions, the requester sets the Master Data Parity Error bit in the Status register for data parity errors on either the Split Request or the Split Completion, and upon receipt of a Split Completion Message indicating Split Write Data Parity Error.

- yes ___  no___ N/A___

XEP7. If data parity error recovery is disabled, the device asserts \( \text{SERR#} \) when a data parity error occurs.

- yes ___  no___ N/A___

XEP8. After a device asserts \( \text{PERR#} \), the device drives \( \text{PERR#} \) high for one clock at the end of the bus operation and before the turn-around cycle.

- yes ___  no___ N/A___

XEP9. The target of a write or Split Completion transaction does not check parity while it is inserting initial wait states.

- yes ___  no___ N/A___

XEP10. If a Master-Abort or Target-Abort occurs during a Split Completion, the initiator (completer or bridge) discards the Split Completion. The completer additionally sets the Split Completion Discarded bit and asserts \( \text{SERR#} \) (if enabled) if the original Split Request was a read with side effects. If the initiator is a bridge, the Split Completion Discarded bit must be set and \( \text{SERR#} \) asserted (if enabled) regardless of the address of the original Split Request.

- yes ___  no___ N/A___
XEP11. If a bridge signals Data Transfer on a non-posted write on the originating bus, and the transaction has a data parity error, the bridge discards the transaction.  

yes ___  no___ N/A___

XEP12. If a bridge detects a data parity error on a Split Completion Message, it discards the transaction and asserts **SERR#** (if enabled).  

yes ___  no___ N/A___
9. PCI-X Bus Width Checklist (XWP)

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. Rules designated for initiators do not apply to devices that do not initiate transactions. Rules designated for 64-bit transactions do not apply to devices that are always 32 bits wide.

XWP1. The device implements the width of the address independent of the width of the data transfer.  

XWP2. If the device initiates memory transactions, it is capable of generating 64-bit memory addresses.  

XWP3. If the device requests a memory range through a Base Address register, that Base Address register is 64-bits wide.  

XWP4. If the device initiates a transaction and the address is greater than or equal to 4 GB, the device generates a dual address cycle.  

XWP5. The attribute phase is always a single clock.  

XWP6. Only burst transactions (Split Completions or memory commands other than Memory Read DWORD) use 64-bit data transfers.  

XWP7. AD[63::32] and C/BE[7::4]# are driven high during the attribute phase of transactions from a 64-bit initiator.  

XWP8. A “Reserved” bus is not specified and is ignored by the device receiving the bus (same as conventional PCI).  

XWP9. If the address of a transaction is less than 4 GB, the transaction uses a single address cycle. For 64-bit devices, AD[63::32] and C/BE[7::4]# are reserved during the address phase of these transactions (same as conventional PCI).  

XWP10. If the initiator is 64-bits wide and the address of a transaction is 4 GB or greater, AD[63::00] contain the full address, C/BE[3::0]# contain the Dual Address Cycle command, and C/BE[7::4]# contain the transaction command during the first address phase (same as conventional PCI).  

XWP11. If the initiator is 64-bits wide and the address of a transaction is 4 GB or greater, AD[63::32] and AD[31::00] contain duplicate copies of the upper half of the address, and C/BE[3::0]# and C/BE[7::4]# contain duplicate copies of the transaction command during the second address phase (same as conventional PCI).  

XWP12. If the initiator is 32 bits wide and the address of a transaction is 4 GB or greater, AD[31::00] contain the lower half of the address during the first address phase and contain the upper half of the address during the second address phase (same as conventional PCI).
XWP13. If the initiator is 32-bits wide and the address of a transaction is 4 GB or greater, C/BE[3::0]# contain the Dual Address Cycle command during the first address phase and contain the transaction command during the second address phase (same as conventional PCI).  

XWP14. DEVSEL# timing designations measure from the second address phase of a transaction with a Dual Address Cycle (same as conventional PCI).  

XWP15. Split Completions always have a single address phase both for 64-bit and 32-bit initiators.  

XWP16. A 64-bit initiator asserts REQ64# with the same timing as FRAME# to request a 64-bit data transfer. It de-asserts REQ64# with FRAME# at the end of the transaction (same as conventional PCI).  

XWP17. The device includes a configuration bit in the PCI-X Status register to indicate the width of its interface. If installed on an add-in card the bit indicates the narrower of the widths of the device or the card interfaces.  

XWP18. Allowable disconnect boundaries are unaffected by the width of the data transfer. A 32-bit transfer has twice as many data phases between two ADBs.  

XWP19. If the transfer is 64 bits wide (REQ64# asserted) and AD[2] of the starting byte address is 1 for memory write and Split Completion transactions, the initiator duplicates the upper 32-bits of data on AD[63::32] and AD[31::00], and the upper 4-bits byte enables on C/BE[7::4]# and C/BE[3::0]# for the first data phase.  

XWP20. If the transfer is 64 bits wide (REQ64# asserted), and AD[2] of the starting byte address is 1 for memory write and Split Completion transactions, and the target asserts ACK64# when it asserts DEVSEL#, and the target signals Wait State, the initiator must toggle between the first and second QWORD data phases.  

XWP21. If the transfer is 64 bits wide (REQ64# asserted), and AD[2] of the starting byte address is 1 for memory write and Split Completion transactions, and the target does not assert ACK64# when it asserts DEVSEL#, and the target signals Wait State, the initiator must toggle between the first and second DWORD data phases.  

XWP22. The width of the transaction is established by the state of ACK64# on the first clock that DEVSEL# is asserted. Deassertion of ACK64# and DEVSEL# for Single Data Phase Disconnect does not change the data width established.  

XWP23. The Split Completion address for a DWORD request and for a Split Completion Message is always 0, so the data or Split Completion Message always appears on AD[31::00].
10. **PCI-X Split Transaction Checklist (XST)**

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. Rules designated for initiators do not apply to devices that do not initiate transactions. Rules designated for completers do not apply to devices that never signal Split Response.

| XST1. If the completer signals Split Response, it initiates one or more Split Completion transactions with that Requester ID and Tag. | yes ___ no___ N/A___ |
| XST2. The completer never creates a Split Completion transaction containing both read data and a Split Completion Message. | yes ___ no___ N/A___ |
| XST3. If the completer returns read data, the Completer returns all the data (the full byte count) unless an error occurs. | yes ___ no___ N/A___ |
| XST4. The completer never uses a byte count in the Split Completion other than the full remaining byte count for the Sequence, except to disconnect the Split Completion on the first ADB of the Sequence. | yes ___ no___ N/A___ |
| XST5. If the completer disconnects the Split Completion on the first ADB by adjusting the byte count, the completer sets the Byte Count Modified attribute bit for the affected transactions. | yes ___ no___ N/A___ |
| XST6. Each time the Split Completion resumes after a disconnection at an ADB, the initiator adjusts the byte count (and starting address) to indicate the number of bytes remaining in the Sequence. | yes ___ no___ N/A___ |
| XST7. The requester never terminates a Split Completion transaction with Split Response, Single Data Phase Disconnect, Retry, or Disconnect at Next ADB. | yes ___ no___ N/A___ |
| XST8. The requester correctly handles a Split Response termination and corresponding Split Completion for Memory Read Block commands. | yes ___ no___ N/A___ |
| XST9. The requester correctly handles a Split Response termination and corresponding Split Completion for Memory Read DWORD commands. | yes ___ no___ N/A___ |
| XST10. The requester correctly handles a Split Response termination and corresponding Split Completion for I/O Read commands. | yes ___ no___ N/A___ |
| XST11. The requester correctly handles a Split Response termination and corresponding Split Completion for I/O Write commands. | yes ___ no___ N/A___ |
| XST12. The requester correctly handles a Split Response termination and corresponding Split Completion for Configuration Read commands. | yes ___ no___ N/A___ |
| XST13. The requester correctly handles a Split Response termination and corresponding Split Completion for Configuration Write commands. | yes ___ no___ N/A___ |
| XST14. The requester correctly handles a Split Response termination and corresponding Split Completion for Interrupt Acknowledge commands. | yes ___ no___ N/A___ |
XST15. Completers that signal Split Response return Split Completion even when their Bus Master bit is cleared in configuration space. yes ___ no___ N/A___

XST16. Requesters that have Split Completions outstanding for more than one Sequence at a time function correctly if Split Completions for different Sequences are returned in a different order than the Split Requests. yes ___ no___ N/A___

XST17. Completers generate the Split Completion address and Split Completion attributes according to the rules in Sections 2.10.3 and 2.10.4. yes ___ no___ N/A___

XST18. If the request is a DWORD write transaction, or if the completer encounters an error while executing the request, the completer sends a Split Completion Message to the requester. yes ___ no___ N/A___

XST19. Completers that signal Split Response detect transactions which exceed their address range and send a Split Completion Message indicating the error (Class 2h, Index 00h) after transferring data up to their device boundary. yes ___ no___ N/A___

XST20. The requester accepts all Split Completions from its own Split Requests. yes ___ no___ N/A___
11. **Interoperability and Initialization Checks (XIN)**

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. Rules designated for initiators do not apply to devices that do not initiate transactions. Rules designated for completers do not apply to devices that never signal Split Response.

| XIN1. If placed into a bus with a 33MHz conventional PCI device, the device will operate normally (includes the acceptance of configuration transactions) in conventional PCI mode. | yes ___ no___ |
| XIN2. If the device is a PCI-X 133 device and it is placed in a bus with at least one PCI-X 66 device, the device operates normally (including accepting configuration transactions) in PCI-X 66 mode. | yes ___ no___ N/A___ |
| XIN3. If the device is a PCI-X 66 device, it operates in PCI-X mode at any frequency from 50MHz to 66 MHz. | yes ___ no___ N/A___ |
| XIN4. If the device is a PCI-X 133 device, it operates in PCI-X mode at any frequency from 50MHz to 133 MHz. | yes ___ no___ N/A___ |
| XIN5. A PCI-X 133 device operates properly in a hot-plug system if the system asserts **RST#** and changes the bus frequency to enable the addition of a PCI-X 66 card in an adjacent slot. | yes ___ no___ N/A___ |
| XIN6. A PCI-X device operates properly in a hot-plug system if the system asserts **RST#** and changes the mode to conventional PCI to enable the addition of a conventional PCI card in an adjacent slot. | yes ___ no___ |
| XIN7. A PCI-X system provides the PCI-X initialization pattern on the **DEVSEL#**, **STOP#**, **TRDY#** signals at the rising edge of **RST#**, indicating the operating frequency of the system. | yes ___ no___ N/A___ |
| XIN8. If the device is installed on an add-in card, the 133 MHz Capable bit in the PCI-X Status register is set consistently with the wiring of the PCIXCAP pin on the card edge connector. That is, if the card is a PCI-X 133 card, the bit is set to 1 and PCIXCAP is connected to ground through a capacitor. If the card is a PCI-X 66 card, the bit is cleared to 0 and PCIXCAP is connected to ground through a resistor and capacitor. | yes ___ no___ N/A___ |
12. PCI-X Bridge Checklist (XBP)

The following checklist is to be filled out as a general verification of the IUT’s protocol compliance. This checklist applies only to PCI-X bridge devices (Type 01h header).

XBP1. The bridge accepts immediate read data of two ADQs.  

   yes ___  no___

XBP2. The bridge supports all of the following settings of the Split Transaction Commitment Limit register:
   1. Set to same value as the Split Transaction Capacity register (no overcommitment).
   2. Set to a value greater than the Split Transaction Capacity register (overcommitment).
   3. Set to a value of FFFFh (flood).

   yes ___  no___

XBP3. Both interfaces of the bridge operate in conventional or PCI-X mode independently. All four of the following combinations are supported:
   - PCI-X to PCI-X.
   - PCI-X to conventional PCI.
   - Conventional PCI to PCI-X.
   - Conventional PCI to conventional PCI.

   yes ___  no___

XBP4. The bridge properly generates the following Split Completion Class 1 Error Messages:

<table>
<thead>
<tr>
<th>Index</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00h</td>
<td>Master-Abort</td>
</tr>
<tr>
<td>01h</td>
<td>Target-Abort</td>
</tr>
<tr>
<td>02h</td>
<td>Write Data Parity Error</td>
</tr>
</tbody>
</table>

   yes ___  no___

XBP5. Memory write transactions that cross the bridge in either direction remain in the same Sequence on both sides of the bridge. Their Sequence IDs are not modified by the bridge.

   yes ___  no___

XBP6. The bridge sets the Split Completion Overrun bit for either interface when it terminates a split completion on that interface with Retry or Disconnect at Next ADB because its buffers are full or because of some other manifestation of setting the bridge’s commitment level too high.

   yes ___  no___
13. PCI-X System Electrical Checklist (XME)

The following checklist applies to all systems when operating in PCI-X mode.

XME1. All bussed PCI signals are driven by compliant components. yes ___ no___

XME2. **CLK** cycle time is greater than or equal to 7.5 ns and less than 15 ns for PCI-X 133, or greater than or equal to 15 ns and less than 20 ns for PCI-X 66. yes ___ no___

XME3. **CLK** skew between any two devices in the system is less than 0.5 ns. yes ___ no___

XME4. **CLK** is delivered to all components with at least 3 ns of high/low time for PCI-X 133, or 6 nS for PCI-X 66. yes ___ no___

XME5. (3.3V signaling) P-t-p **CLK** swing is at least 0.4Vcc (0.2Vcc to 0.6Vcc). yes ___ no___

XME6. Tprop from any driver to any other device is less than or equal to \[T_{cyc} - \left( T_{val(max)} + T_{skew(max)} + T_{su(max)} \right)\], which evaluates to 2.0 ns for \(T_{cyc} = 7.5\) ns, 4.5 ns for \(T_{cyc} = 10\) ns, and 9.0 ns for \(T_{cyc} = 15\) ns. yes ___ no___

XME7. Tprop from a driver to any other device is greater than or equal to \(T_{skew(max)} + T_{h(min)} - T_{val(min)}\). yes ___ no___

XME8. The minimum pull-up resister value is 5 kΩ. yes ___ no___

XME9. Reflective noise and crosstalk noise are limited to a maximum of 0.35 Vcc for high logic signals and 0.20 Vcc for low logic signals. yes ___ no___

XME10. If the source bridge requires **IDSEL** inputs to be resistively coupled to **AD** bits, those resistors have a value of a least 2 kΩ. yes ___ no___

XME11. If the source bridge requires **IDSEL** inputs to be resistively coupled to **AD** bits, devices 1-4 connect to **AD[17]** through **AD[20]** respectively. yes ___ no___

XME12. If the system includes slots for add-in cards, the system provides a circuit for sensing the connection of the **PCIXCAP** pin of all add-in cards. yes ___ no___ N/A___
14. PCI-X Add-in Card Electrical Checklist (XAE)

The following checklist applies to all add-in cards when operating in PCI-X mode.

XAE1. PCIXCAP pin is connected to ground through a 0.01 ±10% μF capacitor for PCI-X 133, or through a 10 kΩ ±5% resistor in parallel with a 0.01 ±10% μF capacitor for PCI-X 66.  yes ___ no___

XAE2. Trace length for AD[31::0] is between 0.75 inches and 1.5 inches, inclusive.  yes ___ no___

XAE3. If the card is a 64-bit card, trace length for AD[63::32] is between 1.75 inches and 2.75 inches, inclusive.  yes ___ no___ N/A___

XAE4. Trace length for RST# is between 0.75 inches and 3.0 inches, inclusive.  yes ___ no___

XAE5. Trace spacing, geometries, and materials limit cumulative crosstalk to 5% of the amplitude of the aggressor signal.  yes ___ no___

XAE6. Characteristic impedance of signal traces is 57 ±10% Ω.  yes ___ no___

XAE7. Trace length for CLK is between 2.4 inches and 2.6 inches, inclusive. (Same as conventional PCI.)  yes ___ no___

XAE8. Signal propagation delay of traces is between 150 and 190 ps/inch, inclusive. (Same as conventional PCI.)  yes ___ no___