Role of PCI-X

- *Next generation PCI*: it is an optimized version of PCI.
- PCI-X is intended as the interim technology between PCI and *PCI Express*
- PCI-X is backward-compatible with PCI.
Terminology

- Source Bridge
- Requester and Completer
- Requester ID
- Tag
- Sequence ID
- Sequence
- Source Bridge
- Requester and Completer
- Requester ID
- Tag
- Sequence ID
- Sequence
Detection of PCIX-Capable Functions

- Lowest-common denominator bus
- Hardware detection of card type
- Software detection of function type
Lowest-common denominator bus

• Four device types:
  – 33MHz-capable PCI
  – 66MHz-capable PCI
  – 50-66MHz-capable PCI-X
  – 50-133MHz-capable PCI-X
Hardware detection of card type

- New signal defined: PCIXCAP (B38)
- Source bridge samples PCIXCAP and M66EN during assertion of RST#.
- Bridge drives initialization pattern to PCI-X capable devices and then removes RST#.
- Pattern defines bus protocol and basic operational frequency range.
Software detection of function type

- Check Capabilities List bit in Status register
- Traverse New Capabilities configuration registers sets looking for the register set with New Capability ID 07h.
See next slide
PCI-X Capability Register Set

```
<table>
<thead>
<tr>
<th>31</th>
<th>16</th>
<th>15</th>
<th>8</th>
<th>7</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCI-X Command Register</td>
<td>Pr to next Capability</td>
<td>Capability ID = 07h</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCI-X Status Register</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
Block-Oriented Bus

- Data is transferred in blocks of 128 bytes each.
- While PCI transactions can be disconnected on dword or qword boundaries, PCI-X transactions can only be disconnected on block boundaries.
- Boundary referred to as an ADB.
- Blocks referred to as ADQs (ADB-Delimited Quanta).
Two transaction categories:

- Dword transactions
- Burst transactions (always memory transfers unless Split Completion)
Burst Transactions

- Memory Write
- Memory Write Block
- Memory Read Block
- Alias to Memory Read Block
- Alias to memory Write Block
- Split Completion
Address Phase

- Initiator:
  - provides start byte address
  - provides Command
  - asserts FRAME#
  - may assert REQ64#
Attribute Phase

• Initiator:
  – provides Requester or Completer ID
  – provides Tag number.
  – provides byte transfer count or Byte Enables
  – if a memory transaction, provides RO and NS Attribute bits.
Response Phase

• Initiator awaits connection with target (i.e., the assertion of DEVSEL#).

Data Phase(s)

• Data (or an SCM) is transferred
Example PCI-X Transfer
Example PCI-X Transfer
Example PCI-X Transfer
Example PCI-X Transfer

IRDY# always asserted in fourth clock cycle of all transactions.
Example PCI-X Transfer

Earliest assertion of TRDY# is in clock after DEVSEL# asserted.
Example PCI-X Transfer

Byte Enables Reserved and driven high for all bursts except Memory Write burst.
Example PCI-X Transfer
Example PCI-X Transfer
Example PCI-X Transfer

This illustrates the initiator ending a transaction of four or more data phases:
- due to byte count satisfaction or
- to inform the target that it wishes to disconnect the transaction on the next ADB.
Example PCI-X Transfer
Example PCI-X Transfer
Example PCI-X Transfer
Split Transactions

- Replaces PCI Delayed Transactions
- Target issues Split Response if it can’t meet the 16 clock rule
- Response issued within 8 clocks after FRAME# asserted.
- Memory writes are NEVER split, always posted.
Effects on Initiator

• Transactions ends without completion.
• Requester suspends transaction and moves to its Split Transaction Queue.
• Requester awaits subsequent Split Completion(s) that fulfill request or deliver a Split Completion Error Message.
Effects on Target

- Target memorizes the transaction (including the data if it’s an IO or Configuration write).
- The target then performs the read or write off-line.
- When the target has performed the write, or has accumulated some or all of the requested memory read data, it performs a series of one or more Split Completion transactions.
Split Completion Address Phase

• The Sequence ID issued by the Requester is used as the address, while the transaction type used (i.e., the command) is the Split Completion command.
Split Completion Attribute Phase

- The Completer provides its Completer ID in the Attribute Phase of the Split Completion transaction.
Split Completion Response Phase

- The Completer awaits the assertion of DEVSEL# by the Requester.
Split Completion Data Phase(s)

- The Completer provides either the requested read data or an SCM.