PowerPC™

The 660 Bridge FAQ – 6/20/96

Frequently Asked Questions about the IBM27-82660 PowerPC to PCI Bridge and Memory Controller. This is a living document, which was last updated on the date shown in the title.

Application Note AN-PPC-011

This FAQ contains 660 bridge information which the PowerPC Application Engineering team believes to be of interest to the PowerPC community, and which are too short to be the subject of individual application notes.

MPRA11APU-02
Dale Elson

IBM®

The 660 Bridge FAQ – 6/20/96

Dale Elson

In this document, the IBM27-82660 is referred to as the 660 bridge or the 660. Likewise, the IBM27-82663 and IBM27-82664 are referred to as the 663 and the 664. The *IBM27-82660 PowerPC to PCI Bridge User's Manual* is often referred to as the 660UM.

The 660UM is available in hard copy as SC09–3026–00, and as soft copy as noted in item 1.

1 Where can I get 660 information ?

Date 4/9/96

Description .. Currently, 660 data is located in several places. IBM employes (such as your FAE or other IBM technical representative) can access this information using the addresses marked *Internal*. This information can be accessed externally using the other addresses.

660 Bridge – User's Manual and other information.

Internal ...... /afs/awd/work/PowerPC/support_chips/660_bridge/
.... From AIX, rftp ftp.austin.ibm.com/pub/PPC_support/660_bridge
External ..... ftp://ftp.austin.ibm.com/pub/PPC_support/660_bridge/
.... the IP address is (192.35.232.38)

Reference Designs – Technical Specifications and design files showing how to use the 660 and other PowerPC products.

Internal ...... /afs/awd/work/PowerPC/reference_designs/
.... From AIX, rftp ftp.austin.ibm.com/pub/PPC_support/reference_designs
External ..... ftp://ftp.austin.ibm.com/pub/PPC_support/reference_designs/
.... the IP address is (192.35.232.38)

Application Notes – Technical articles from the PowerPC Application Engineering team about the 660 and other PowerPC products.

Internal ...... /afs/awd/work/PowerPC/app_notes/
.... From AIX, rftp ftp.austin.ibm.com/pub/PPC_support/app_notes
External ..... ftp://ftp.austin.ibm.com/pub/PPC_support/app_notes/
.... the IP address is (192.35.232.38)

2 Can the 660 address and data buses be left tristated ?

Date 4/9/96

Description . . We have not seen any functional problems with the 660 which have been caused by allowing CPU_ADDR[0:31] and CPU_DATA[0:63] to float for an extended amount of time (several seconds).

However, the 660 is a CMOS device, so allowing the inputs to float allows the possibility that the input voltage will float to a level that causes both of the input transistors to conduct some current, thus increasing the power dissipation of these inputs. A more remote possibility is that these inputs will oscillate, which will also increase the power dissipation of the inputs without causing functional problems. As with all electrical devices, system level and handling precautions should be taken to ensure that the 660 is not exposed to ESD events in excess of its published protection limits.

3 What are the access times for 50ns, 60ns, & 70ns EDO DRAM ?

Date 4/9/96

Description . . Additional access times are now available for 50ns, 60ns, and 70ns EDO DRAM (see Table 1 and Table 2). These timings are presented in the same manner as the ones in Section 5 of the 660UM. Notice that the column in Table 2 for 60ns aggressive timing corrects an error in some copies of the 660 UM.

Table 1. General Case EDO DRAM Timing Summary

Transfer	70ns EDO Conservative	70ns EDO Aggressive	60ns EDO Conservative	Note
Memory Timing Register 1	32	32	31	
Memory Timing Register 2	0A	06	06	
Bridge Chipset Options 3	0C	0C	0C	
Initial Read Burst	12-4-4-4	11-3-3-3	11-3-3-3	(3)
Initial Write Burst	5-4-4-4	5-4-3-3	5-4-3-3	
For a pipelined burst transfer immediately following a read:				
Page Hit Read	-5-4-4-4	-5-3-3-3	-5-3-3-3	(3)
Page Hit Write	-3-3-4-4	-3-3-3-3	-3-3-3-3	
Page Miss & Bank Miss Read	-9-4-4-4	-8-3-3-3	-8-3-3-3	(1,3)
Page Miss & Bank Hit Read	-11-3-3-3	-10-3-3-3	-9-3-3-3	(2,3)
Page Miss & Bank Miss Write	-3-3-4-4	-3-3-3-3	-3-3-3-3	(1)
Page Miss & Bank Hit Write	-3-5-4-4	-3-5-3-3	-3-4-3-3	(2)
For a pipelined burst transfer immediately following a write:				
Page Hit Read	-12-4-4-4	-9-3-3-3	-9-3-3-3	(3)
Page Hit Write	-6-3-4-4	-5-3-3-3	-5-3-3-3	
Page Miss & Bank Miss Read	-15-4-4-4	-12-3-3-3	-12-3-3-3	(1,3)
Page Miss & Bank Hit Read	-17-4-4-4	-14-3-3-3	-13-3-3-3	(2,3)
Page Miss & Bank Miss Write	-6-6-4-4	-5-5-3-3	-5-5-3-3	(1)
Page Miss & Bank Hit Write	-6-8-4-4	-5-7-3-3	-5-6-3-3	(2)

See the notes for Table 2.

Table 2. General Case EDO DRAM Timing Summary

Transfer	60ns EDO Aggressive	50ns EDO Conservative	50ns EDO Aggressive	Note
Memory Timing Register 1	11	0C	2C	
Memory Timing Register 2	05	05	02	
Bridge Chipset Options 3	0C	0C	0C	
Initial Read Burst	10-3-3-3	10-3-3-3	10-2-2-2	(3)
Initial Write Burst	5-3-3-3	5-3-3-3	5-4-2-2	
For a pipelined burst transfer immediately following a read:				
Page Hit Read	-5-3-3-3	-5-3-3-3	-5-2-2-2	(3)
Page Hit Write	-3-3-3-3	-3-3-3-3	-3-2-2-2	
Page Miss & Bank Miss Read	-7-3-3-3	-7-3-3-3	-7-2-2-2	(1,3)
Page Miss & Bank Hit Read	-8-3-3-3	-7-3-3-3	-7-2-2-2	(2,3)
Page Miss & Bank Miss Write	-3-3-3-3	-3-3-3-3	-3-3-2-2	(1)
Page Miss & Bank Hit Write	-3-3-3-3	-3-3-3-3	-3-3-2-2	(2)
For a pipelined burst transfer immediately following a write:				
Page Hit Read	-9-3-3-3	-9-3-3-3	-7-2-2-2	(3)
Page Hit Write	-5-3-3-3	-5-3-3-3	-4-2-2-2	
Page Miss & Bank Miss Read	-11-3-3-3	-11-3-3-3	-10-2-2-2	(1,3)
Page Miss & Bank Hit Read	-12-3-3-3	-11-3-3-3	-10-2-2-2	(2,3)
Page Miss & Bank Miss Write	-5-4-3-3	-5-4-3-3	-4-5-2-2	(1)
Page Miss & Bank Hit Write	-5-5-3-3	-5-4-3-3	-4-5-2-2	(2)

1) The RAS# of the new bank is high and has been high (precharging) for the minimum RAS# high time. The bridge places the address on the address lines and asserts RAS#.
2) The access is a page miss, but within the same bank, so the RAS# line must be sent high for at least the minimum RAS# high (precharge) time. The bridge also places the new address on the address lines and asserts RAS#.
3) If asynchronous SRAMs are used with the internal L2 controller, an additional clock cycle is added to the fourth beat of any CPU to memory read burst that causes a cache miss. For example, following a read—a pipelined page hit, cache miss, burst read with EDO DRAM, requires –3–3–3–3 CPU clocks when the L2 uses burst SRAMs and –3–3–3–4 CPU clocks when the L2 uses asynchronous SRAMs. This extra beat is caused by delaying the final TA# by one CPU_CLK to allow the asynchronous SRAM sufficient data hold time for the fourth beat.

4 Is my DRAM fast enough to do 660 RMW cycles in ECC mode ?

Date 4/9/96

Description . . In ECC mode, the 660 can do a read–modify–write cycle to the DRAM if the CPU or a PCI bus-master initiates a memory write of less than 8 bytes.

As shown in Section A.7.5 of the 660UM, when the 660 does a RMW cycle, it does not use the RMW cycle capability that is built into some DRAMs. Instead, it does a read, modifies the data, and then writes back the data (as a pipelined page hit write). This allows RMW cycles with any DRAM that is fast enough to perform read cycles and write cycles. No recalculation of the CAS# pulse width is required.

5 How do interrupt acknowledge cycles work ?

Date 4/11/96

Description . . To perform an interrupt acknowledge operation, the CPU initiates a single-byte read to address BFFF FFF0. This causes the 660 to arbitrate for the PCI bus and then to initiate a single-byte PCI Interrupt Acknowledge transaction with PCI_C/BE[0]# active. PCI_C/BE[0]# is active regardless of the endian mode of the 660. The PCI Interrupt Acknowledge transaction that the 660 generates is similar to those generated by x86 to PCI bridges. The interrupt controller (eg SIO) then claims the transaction and supplies the single-byte interrupt vector on PCI byte lane 0. The 660 then returns the vector to the CPU on the correct byte lane.

There is no physical interrupt vector BCR in the bridge. Other PCI bus masters can initiate interrupt acknowledge transactions.

6 660 User's Manual Corrections –SC09–3026–00, 660umf2.ps, 660umf3.ps, & 660umf4.ps

Date 6/20/96

5.2.2 **General Case DRAM Timing Calculations (6/20/96)**. In items 4, 5b, and 6, the term (+ 1 CLK if EDO) is incorrectly shown on the right side of the equation. This term should be on the left side of each equation, as shown below:

```
4. CAS# pulse width (CPW)
        (    + 1 CLK if EDO) >  T CAS# fall max  . . . . .  * CAS# active out if 664
                                + Tcac . . . . . . . . . . . . . .  * DRAM data access from CAS#
                                + MD setup max  . . . . .  * MEM_DATA setup into 663
                                                           * data sampled 1 clk later
```

The last factor (+ 1 CLK if EDO) is included in the equation only if EDO DRAM is used. Note that CPW must be set to 3 or fewer clocks.

```
5. b) Col addr setup (ASC)
   + CAS# pulse width (CPW)
              (+ 1 CLK if EDO) >  T MA max . . . .  * MA[11:0] valid out of  664
                                  + T 244 max   .  * Buffer delay
                                  + Taa min  . . . .  * DRAM data valid from col addr
                                  + MD setup max * MEM_DATA setup into 663

6. RAS# to CAS# delay (RCD)
  + CAS# pulse width (CPW)
        (+ 1 CLK if EDO)      >   T RAS# fall max  . . . . .  * max 660 RAS# fall time (note 1)
                                  + Trac min  . . .  * DRAM data access from RAS#
                                  + MD setup max/   . . . .  * MEM_DATA setup into 663
```

5.4.1 **Memory Parity (6/20/96)**. Change the last word of the first paragraph from PCI_PERR# to PCI_SERR#.

6 **L2 Cache (6/20/96)**. Change the second sentence to read: The L2 caches *as much of the* system memory space from 0 to 2G *as is populated by DRAM.*

10.3.34 **Single-Bit Error Trigger Level Register (6/20/96)**. Change the second paragraph to read If the single-bit error level trigger register is set to 00h, no *single-bit ECC* error is ever generated to the CPU. *Setting this BCR to 00h does not inhibit the recognition or reporting of multi-bit ECC errors.*

7 PCI Master Data Latency

Date 6/20/96

Description . . The 2.1 PCI specification allows host bridges a Master Data Latency of 8 PCI clocks. While the 660 is in 1:1 CPU:PCI clock mode, it is possible to configure the 660 memory controller to operate the DRAM slowly enough that the master data latency becomes as large as 10 PCI clocks. This does not imply that the 660 is noncompliant, but that it is possible for the designer to use the 660 in a noncompliant manner.

8 PCI Target Initial Latency

Date 6/20/96

Description . . The 2.1 PCI specification allows host bridges a Target Initial Latency (TIL) of 32 PCI clocks under certain conditions. When the 660 is in 2:1 CPU:PCI clock mode, and is used with one busmaster on the CPU bus, the TIL is a maximum of 32 PCI clocks.

However, when the 660 is in 1:1 CPU:PCI clock mode with a single busmaster on the CPU bus, it is possible to configure the 660 memory controller to operate the DRAM slowly enough that the TIL becomes as large as 40 PCI clocks. In systems having more than one CPU busmaster, this latency can increase.

Also, when the 660 is used in 2:1 CPU:PCI clock mode with two busmasters on the CPU bus, it is possible to configure the 660 memory controller to operate the DRAM slowly enough that the TIL exceeds 32 PCI clocks. In systems operating at 1:1 CPU:PCI clock mode, this latency can increase

9 Exception to 660 PCI Revision 2.1 Compliance – Transaction Ordering Rules

Date 6/20/96

Description . . Transaction ordering requirements were added to the PCI specification between revs 2.0 and 2.1. The PCI 2.1 specification states that writes from a given master must be visible (by any other agent in the system) as completing in the order in which they were initiated. This requirement is intended to implement the Producer–Consumer model contained in Appendix E of the PCI 2.1 specification.

There is a theoretical case in which the 660 does not comply with the PCI Producer–Consumer model (Appendix E, Summary of PCI Ordering Rules, Item 5b). Note that to date (6/20/96) there have been no know actual occurrances of this case with the 660. It was recently discovered during a comprehensive review of the PCI 2.1 specification. IBM is not aware of any existing hardware/software combination that produces a situation where this case can be observed.

For the following discussion, please refer to the PCI 2.1 specification sections 3.2.5 and Appendix E. Assume that the CPU is the Producer, and that it is producing the final 4 bytes of a data set by initiating a PCI memory write (via the 660) to a particular PCI agent (the Consumer). Assume that the 660 posts the write, and that before the posted write completes on the PCI bus (for instance if it is retried), the CPU writes the flag to system memory.

Meanwhile, the PCI agent is polling the flag by periodically reading the system memory location. Assume that the PCI agent reads the flag as set, meaning that the data set transfer is complete. At this point, the possibility exists that the PCI agent has not allowed the CPU to PCI (data set) write to complete. If so, then the two CPU transactions have completed out of order, and the flag will indicate that the data transfer has completed, when in fact it has not.

To prevent this, the PCI 2.1 spec requires the bridge to react to the PCI to memory read request by pulling all posted writes through the bridge before completing the read. In other words, when the PCI agent initiates the read (polls the flag), the bridge is to retry (or delay) the read until the posted CPU to PCI memory write (the data set transfer) has completed. This has the effect of ”pulling” the writes through the bridge before the read is executed. The 660 does not implement this function.

Note that PCI devices (such as SCSI, Ethernet, and Token Ring adaptors) that poll the flag in system memory typically also consume the data set by initiating PCI to system memory reads. These devices are not affected.

Also, PCI devices (such as graphics adaptors and non–PCI busmasters) that receive the data set as a PCI memory target typically receive indication that the data is ready (the flag) as a PCI I/O or PCI memory target; they do not poll system memory for the flag. These devices are also not affected.

For affected devices, the following options apply:

Option 1: Add a dummy CPU to PCI write:

Design the device drivers to cause the CPU BIU to execute an access to the PCI bus (of any type) after the data set is written (posted into the 660 by the CPU write to the PCI target) and before the CPU BIU executes the write that sets the flag in system memory. This solution forces the data set write to complete on the PCI bus before the flag write is initiated on the CPU bus. (Remember to insert eieio instructions before and after the dummy write.) This produces the following sequence:

1) The CPU initiates a CPU to PCI write to produce the final part of the data set. The CPU address tenure completes, and the PCI write is posted.

2) Some time later, the 660 initiates this transaction on the PCI bus. The transaction takes an unknown amount of time to complete. If the transaction is retried on the PCI bus, the 660 will continue to reinnitiate the transaction until it completes.

3) After (1) and before (4), the CPU initiates a dummy CPU to PCI write to flush the posted write buffer. The 660 does not allow this second CPU to PCI write to complete on the CPU bus (by being written into the 660 posted write buffer) until the first CPU to PCI transaction (the data set write) actually completes on the PCI bus. This prevents the following transaction (the flag write) from completing before the data set write completes.

4) The CPU initiates the CPU to memory flag write.

Option 2: Change the flag write procedure:

Design the device drivers such that the Consumer receives the flag in some way that ensures that the data set write completes before the flag write completes. One way to do this is to cause the CPU to write the flag to the PCI agent using a PCI IO or memory write, instead of a CPU to memory write.

a) If the flag write is a PCI memory write, then as in item 3 in option 1, this flag write forces the preceeding data set write to complete before the flag write completes.

b) If the flag write is a PCI IO write, then the 660 will not post the flag write, which forces the data set write(s) to complete on the PCI bus before the flag write is initiated on the PCI bus.

Option 3: Change the data set write procedure:

Design the device drivers such that the Consumer receives data set some way other than as a PCI memory target (such as an as an I/O target or a PCI busmaster).

a) While the Consumer is receiving the data set as a PCI IO target, the 660 does not post the IO writes, and so each write completes in order, even if some are retried. This technique forces the data set writes to complete before the flag write completes.

b) If the Consumer is receiving the data set by initiating PCI to memory reads, and is receiving the flag by initiating a PCI to memory read, then as long as the CPU initiates the data set writes to memory before the CPU initiates the flag write to memory, then the ordering of the transactions will be preserved.

PowerPC™

USA and Canada:
IBM Microelectronics Division
1580 Route 52, Bldg. 504
Hopewell Junction, NY 12533-6531
Tel: (800) PowerPC
Fax: (800) PowerFax
http://www.chips.ibm.com
http://www.ibm.com
ftp://ftp.austin.ibm.com/pub/PPC_support

Document Number: MPRA11APU–02