

# ***PowerPC***

## **Preliminary**

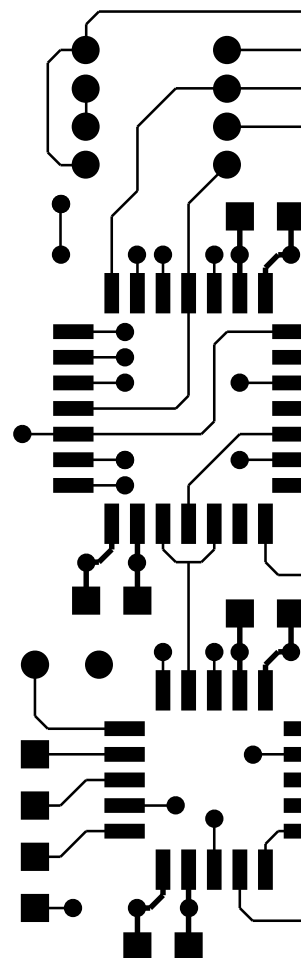
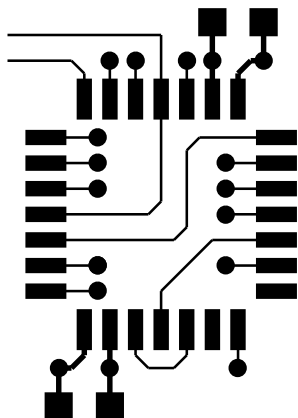
---

### ***100 MHz PPC 603e Multi-Chip Module User's Manual***

#### ***Release 0.1***

*This document provides a detailed technical description of the PPC 603e MCM. It is intended as a first source of information for both hardware and software designers. Where appropriate, other documents are referenced.*

Document Number:  
G5220297-00  
May 1997



© International Business Machines Corporation, 1997. Printed in the United States of America 1997. All Rights reserved.

Note to US Government Users—Documentation related to restricted rights—Use, duplication, or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

IBM Microelectronics, PowerPC, PowerPC 601, PowerPC 603, PowerPC 603e, PowerPC 604, RISCWatch, and AIX are trademarks of the IBM corporation. IBM and the IBM logo are registered trademarks of the IBM corporation. Other company names and product identifiers are trademarks of the respective companies.

This document contains information which is subject to change by IBM without notice. IBM assumes no responsibility or liability for any use of the information contained herein. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. The products described in this document are not intended for use in implantation or other direct life-support applications where malfunction may result in physical harm or injury to persons. INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SOME STATES DO NOT ALLOW DISCLAIMERS OF EXPRESS OR IMPLIED WARRANTIES IN CERTAIN TRANSACTIONS; THEREFORE, THIS STATEMENT MAY NOT APPLY TO YOU.

## Contacts

IBM Microelectronics Division  
1580 Route 52, Bldg. 504  
Hopewell Junction, NY 12533-6531  
Tel: (800) PowerPC

<http://www.chips.ibm.com>  
<http://www.ibm.com>  
[ftp://ftp.austin.ibm.com/pub/PPC\\_support](ftp://ftp.austin.ibm.com/pub/PPC_support)

Dale Elson, Applications Engineer  
IBM PowerPC Embedded Processor Solutions  
[elson@austin.ibm.com](mailto:elson@austin.ibm.com)

## ESD Warning

The planar and MCM contain CMOS devices which are very susceptible to ElectroStatic Discharge (ESD). DO NOT remove them from the antistatic bags until you have connected yourself to an acceptable ESD grounding strap. Work in a static free environment and be sure any person or equipment coming into contact with the cards do not have a static charge. The cards are particularly susceptible until they are placed in a properly designed enclosure. Bench work should be done by persons connected to ESD grounding straps.

**IBM 100 MHz PPC 603e MCM AGREEMENT**

BEFORE READING THE REST OF THE DOCUMENT, YOU SHOULD CAREFULLY READ THE FOLLOWING TERMS AND CONDITIONS. OPENING THE PACKAGE INDICATES YOUR ACCEPTANCE OF THESE TERMS AND CONDITIONS. IF YOU DO NOT AGREE WITH THEM, YOU SHOULD PROMPTLY RETURN THE PACKAGE UNOPENED TO YOUR IBM SALES OFFICE.

International Business Machines Corporation ("IBM") agrees to provide you an 100 MHz PPC 603e MCM in return for your promise to use reasonable efforts to develop a system based on the technology in the MCM. The MCM contains documentation and software listed below:

**Documentation**

IBM27-82660 PowerPC to PCI Bridge and Memory Controller User's Manual  
PowerPC 603e RISC Microprocessor Hardware Specification  
PowerPC 603e RISC Microprocessor Technical Summary  
IBM 64K x 18 Burst SRAM (IBM041814PQK) Data Sheet  
Texas Instruments 3.3V ABT 16-Bit Bus Transceivers with 3-State Output (SN54LVTH16245A, SN54LVTH16245A) Data Sheet  
Integrated Device Technology BiCMOS StaticRAM 240K (16K x 15-Bit) Cache-Tag RAM for PowerPC and RISC Processors (IDT71216) Data Sheet  
Motorola Low Voltage PLL Clock Driver (MPC970/D) Data Sheet

**LICENSE TO SOFTWARE**

The software is licensed not sold. IBM, or the applicable IBM country organization, grants you a license for the software only in the country where you received the software. Title to the physical software and documentation (not the information contained in such documentation) transfers to you upon your acceptance of these terms and conditions. The term "software" means the original and all whole or partial copies of it, including modified copies or portions merged into other programs. IBM retains title to the software. IBM owns, or has licensed from the owner, copyrights to the software provided under this agreement. The terms of this Agreement apply to all of the hardware, software and documentation provided to you as part of the 100 MHz PPC 603e MCM.

With regard to the software provided hereunder, it is understood and agreed that you intend to use the software solely for the purpose of designing PowerPC compatible products, testing your designs, and making your own independent determination of whether you wish to eventually manufacture PowerPC compatible products commercially. In accordance with this understanding, IBM hereby grants you the rights to: a) use, run, and copy the software, but only make such number of copies and run on such number of machines as are reasonably necessary for the purpose of designing PowerPC compatible products and testing such designs; and b) copy the software for the purpose of making one archival or backup copy.

With regard to any copy made in accordance with the foregoing license, you must reproduce any copyright notice appearing thereon. With regard to the software provided hereunder, you may not: a) use, copy, modify or merge the software, except as provided in this license; b) reverse assemble or reverse compile it; or c) sell, sublicense, rent, lease, assign or otherwise transfer it. In the event that you no longer wish to use the software, you will return it to IBM.

**LICENSE TO DESIGN DOCUMENTATION**

With regard to the design documentation provided hereunder, it is understood that you intend to use such documentation solely for the purpose of designing your own PowerPC compatible products, testing your designs, and making your own independent determination of whether you wish to eventually manufacture PowerPC compatible products commercially. In accordance with this understanding, IBM hereby grants you the right to: a) use the design documentation for the purpose of designing PowerPC compatible products and testing such designs; b) make derivative works of the design documentation for the purpose of designing PowerPC compatible products, and testing such designs; and c) make copies of the design documentation and any such derivative works, but only such numbers as are reasonably necessary for designing PowerPC compatible products and testing such designs.

With regard to any copy made in accordance with the foregoing license, you must reproduce any copyright notice appearing thereon. With regard to the design documentation provided hereunder, you may not: a) use, copy, modify, or merge the design documentation as provided in this license; or b) sell, sublicense, rent, lease, assign, or otherwise transfer it.

In the event you no longer wish to use the design documentation or any derivative versions thereof, you must return them to IBM.

**DISCLAIMER OF WARRANTY**

IBM does not represent or warrant that the 100 MHz PPC 603e MCM (which may contain prototype items): a) meets any particular requirements; b) operates uninterrupted; c) is error free; or d) is non-infringing of any patent, copyright, or other intellectual property right of any third party. IBM makes no representation or warranty regarding the performance or compatibility that may be obtained from the use of the MCM or that the MCM is adequate for any use. The MCM may contain errors and may not provide the level of completeness, functionality, support, performance, reliability, or ease of use available with other products, whether or not similar to the MCM. IBM does not represent or warrant that errors or other defects will be identified or corrected.

THE 100 MHz PPC 603e MCM IS PROVIDED "AS IS" WITH ALL FAULTS, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE MCM IS WITH YOU.

Some jurisdictions do not allow exclusion of implied warranties, so the above exclusions may not apply to you.

## **LIMITATION OF REMEDIES**

IBM's entire cumulative liability and your exclusive remedy for damages for all causes, claims or actions wherever and whenever asserted relating in any way to the subject matter of this agreement including the contents of the 100 MHz PPC 603e MCM and any components thereof, is limited to twenty five thousand dollars (\$25,000.00) or its equivalent in your local currency and is without regard to the number of items in the MCM that caused the damage. This limitation will apply, except as otherwise stated in this Section, regardless of the form of the action, including negligence. This limitation will not apply to claims by you for bodily injury or damages to real property or tangible personal property. In no event will IBM be liable for any lost profits, lost savings, or any incidental damages or economic consequential damages, even if IBM has been advised of the possibility of such damages, or for any damages caused by your failure to perform your responsibilities. In addition, IBM will not be liable for any damages claimed by you based on any third party claim. Some jurisdictions do not allow these limitations or exclusions, so they may not apply to you.

## **RISK OF LOSS**

You are responsible for all risk of loss or damage to the 100 MHz PPC 603e MCM upon its delivery to you.

## **IBM TRADEMARKS AND TRADE NAMES**

This Agreement does not give you any rights to use any of IBM's trade names or trademarks. You agree that should IBM determine that any of your advertising, promotional, or other materials are inaccurate or misleading with respect to IBM trademarks or trade names, that you will, upon written notice from IBM, change or correct such materials at your expense.

## **NO IMPLIED LICENSE TO IBM INTELLECTUAL PROPERTY**

Notwithstanding the fact that IBM is hereby providing design information for your convenience, you expressly understand and agree that, except for the rights granted under sections 1 and 2 above, no right or license of any type is granted, expressly or impliedly, under any patents, copyrights, trade secrets, trademarks, or other intellectual property rights of IBM. Moreover, you understand and agree that in the event you wish to be granted any license beyond the scope of the expressly stated herein, you will contact IBM's Intellectual Property Licensing and Services Office (currently located at 500 Columbus Avenue, Thornwood, N.Y.), or such other IBM offices responsible for the licensing of IBM intellectual property, when you seek the license.

## **YOUR ASSUMPTION OF RISK**

You shall be solely responsible for your success in designing, developing, manufacturing, distributing, and marketing any product(s), or portion(s), where use of all or any part of the 100 MHz PPC 603e MCM is involved. You are solely responsible for any claims, warranties, representations, indemnities and liabilities you undertake with your customers, distributors, resellers or others, concerning any product(s) or portion(s) of product(s) where use of all or any part of the MCM is involved. You assume the risk that IBM may introduce other Reference Design that are somehow better than the MCM which is the subject of this Agreement. Furthermore, you accept sole responsibility for your decision to select and use the MCM; for attainment or non-attainment of any schedule, performance, cost, reliability, maintainability, quality, manufacturability or the like, requirements, or goals, self-imposed by you or accepted by you from others, concerning any product(s) or portion(s) of product(s), or for any delays, costs, penalties, charges, damages, expenses, claims or the like, resulting from such non-attainment, where use of all or any part of the MCM is involved.

## **GENERAL**

In the event there is a conflict between the terms of this Agreement and the terms printed or stamped on any item or any ambiguities with respect thereto, including documentation, contained in the 100 MHz PPC 603e MCM, the terms of this Agreement control to the extent IBM is afforded greater protection thereby. IBM may terminate this Agreement if you fail to comply with the terms and conditions of this Agreement. Upon termination of this Agreement, you must destroy all copies of the software and documentation. You are responsible for payment of any taxes, including personal property taxes, resulting from this Agreement. Neither party may bring an action hereunder, regardless of form, more than one (1) year after the cause of the action arose. If you acquired the MCM in the United States, this Agreement is governed by the laws of the State of New York. In the event of litigations, trial shall be in New York without a jury. If you acquired the MCM in Canada, this Agreement is governed by the laws of the Province of Ontario; otherwise, this Agreement is governed by the laws of the country in which you acquired the MCM. All obligations and duties which, by their nature, survive termination or expiration of this Agreement, shall remain in effect beyond termination or expiration of this Agreement, and shall bind IBM, you and your successors and assigns. If any section or paragraph of this Agreement is found by competent authority to be invalid, illegal or unenforceable in any respect for any reason, the validity, legality, and enforceability of any such section or paragraph in every other respect, and the remainder of this Agreement, shall continue in effect so long as it still expresses the intent of the parties. If the intent of the parties cannot be preserved, the parties will attempt to renegotiate this Agreement and failing renegotiation, this Agreement will then be terminated. The headings in this Agreement shall not affect the meaning or interpretation of this Agreement in any way. No failure by IBM in exercising any right, power or remedy under this Agreement shall serve as a waiver of any such right, power or remedy. Neither this Agreement nor any activities hereunder will impair any right of IBM to develop, manufacture, use or market, directly or indirectly, alone or with others, any products or services competitive with those offered or to be offered by you; nor will this Agreement or any activities hereunder require IBM to disclose any business planning information to you. You agree to comply with all applicable government laws and regulations. Any changes to this Agreement must be in writing and signed by the parties.

---

**Table of Contents**

---

<b>Section 1 Introduction</b>	<b>1-1</b>
1.1 The MCM	1-1
1.2 100 MHz PPC 603e MCM	1-2
1.2.1 The CPU	1-3
1.2.2 L2 Cache	1-3
1.2.3 System Memory Interface	1-3
1.2.4 The PCI Interface	1-3
1.2.5 System Clocks	1-4
1.2.6 MCM Performance	1-4
1.3 MCM Technology Overview	1-5
1.3.1 Chip Attachment to MCM	1-5
1.3.2 Substrate Attachment to the Planar	1-6
1.4 Incorporating the MCM into a System Design	1-7
<b>Section 2 Signal Descriptions</b>	<b>2-1</b>
2.1 Pin Descriptions	2-1
2.1.1 Clock Subsystem	2-3
2.1.2 CPU Bus	2-4
2.1.3 PCI Bus	2-7
2.1.4 DRAM	2-8
2.1.5 L2	2-9
2.1.6 System Interface	2-10
2.1.7 660 InterChip Communication	2-11
2.1.8 Test Signals	2-13
2.2 Net Names and MCM Nodes	2-14
2.2.1 Complete Pin List for MCM	2-14
<b>Section 3 CPU and Level Two Cache</b>	<b>3-1</b>
3.1 CPU Busmasters	3-1
3.1.1 603e CPU	3-1
3.2 System Response by CPU Bus Transfer Type	3-2
3.3 System Response by CPU Bus Address Range	3-4
3.3.1 Address Mapping for Non-Contiguous I/O	3-4
3.3.2 Address Mapping for Contiguous I/O	3-6
3.3.3 PCI Final Address Formation	3-6
3.4 CPU to Memory Transfers	3-7
3.4.1 LE Mode	3-7
3.5 CPU to PCI Transactions	3-7
3.5.1 CPU to PCI Read	3-7
3.5.2 CPU to PCI Write	3-8
3.5.2.1 Eight-Byte Writes to the PCI (Memory and I/O)	3-8
3.5.3 CPU to PCI Memory Transactions	3-8
3.5.4 CPU to PCI I/O Transactions	3-8

3.5.5 CPU to PCI Configuration Transactions .....	3-8
3.5.6 CPU to PCI Interrupt Acknowledge Transaction .....	3-9
3.5.7 PCI Lock .....	3-9
3.6 CPU to BCR Transfers .....	3-9
3.7 L2 .....	3-10
3.7.1 Cache Response to CPU Bus .....	3-10
3.7.2 Cache Response to PCI Bus .....	3-11
3.7.3 L2 Configuration .....	3-11
3.7.4 L2 Organization .....	3-11
3.7.5 Other L2 Related BCRs .....	3-11
3.7.6 SRAM .....	3-13
3.7.7 TagRAM .....	3-14
<b>Section 4 DRAM .....</b>	<b>4-1</b>
4.1 Features and Supported Devices .....	4-1
4.1.1 SIMM Nomenclature .....	4-2
4.1.1.1 DRAM Timing .....	4-2
4.1.1.2 DRAM Error Checking .....	4-2
4.2 DRAM Performance .....	4-2
4.2.1 Memory Timing Parameters .....	4-2
4.2.1.1 Memory Timing Register 1 .....	4-3
4.2.1.2 Memory Timing Register 2 .....	4-4
4.2.1.3 RAS# Watchdog Timer BCR .....	4-4
4.2.1.4 DRAM Timing Calculations .....	4-5
4.2.1.5 DRAM Timing Examples .....	4-7
4.2.1.6 70ns DRAM Calculations .....	4-7
4.2.1.7 60ns DRAM Calculations .....	4-8
4.2.1.8 50ns DRAM Calculations .....	4-9
4.2.1.9 60ns EDO DRAM Calculations .....	4-9
4.2.1.10 Aggressive Timing Summary .....	4-10
4.2.1.11 Conservative Timing Summary .....	4-12
4.2.1.12 Page Hit and Page Miss .....	4-14
4.2.1.13 CPU to Memory Access Pipelining .....	4-15
4.2.1.14 Extended Data Out (EDO) DRAM .....	4-15
4.3 System Memory Addressing .....	4-15
4.3.1 DRAM Logical Organization .....	4-15
4.3.1.1 SIMM Topologies .....	4-16
4.3.1.2 Row and Column Address Generation .....	4-17
4.3.1.3 DRAM Pages .....	4-18
4.3.1.4 Supported Transfer Sizes and Alignments .....	4-18
4.3.1.5 Unpopulated Memory Locations .....	4-18
4.3.1.6 Memory Bank Addressing Mode BCRs .....	4-18
4.3.1.7 Memory Bank Starting Address BCRs .....	4-19
4.3.1.8 Memory Bank Extended Starting Address BCRs .....	4-19
4.3.1.9 Memory Bank Ending Address BCRs .....	4-19

4.3.1.10	Memory Bank Extended Ending Address BCR .....	4-20
4.3.1.11	Memory Bank Enable BCR .....	4-20
4.3.1.12	Memory Bank Configuration Example .....	4-20
4.3.1.13	Memory Bank Enable BCR .....	4-21
4.3.1.14	Memory Bank Addressing Mode .....	4-21
4.3.1.15	Starting and Ending Addresses .....	4-22
4.4	Error Checking and Correction .....	4-23
4.4.1	Memory Parity .....	4-23
4.4.1.1	ECC Overview .....	4-24
4.4.1.2	ECC Data Flows .....	4-24
4.4.1.3	Memory Reads .....	4-25
4.4.1.4	Eight-Byte Writes .....	4-25
4.4.1.5	Less-Than Eight-Byte Writes .....	4-26
4.4.1.6	Memory Performance In ECC Mode .....	4-26
4.4.1.7	CPU to Memory Read in ECC Mode .....	4-27
4.4.1.8	CPU to Memory Write in ECC Mode .....	4-27
4.4.1.9	PCI to Memory Read in ECC Mode .....	4-27
4.4.1.10	PCI to Memory Write in ECC Mode .....	4-27
4.4.1.11	Check Bit Calculation .....	4-29
4.4.1.12	Syndrome Decode .....	4-30
4.5	DRAM Refresh .....	4-30
4.5.1	Refresh Timer Divisor Register .....	4-31
4.6	Atomic Memory Transfers .....	4-32
4.6.1	Memory Locks and Reservations .....	4-32
4.6.2	CPU Reservation .....	4-32
4.6.3	PCI Lock .....	4-32
4.6.4	PCI Lock Release .....	4-32
4.7	DRAM Module Loading Considerations .....	4-33
4.8	Related Bridge Control Registers .....	4-33
<b>Section 5</b>	<b>PCI Bus .....</b>	<b>5-1</b>
5.1	PCI Transaction Decoding .....	5-2
5.1.1	PCI Transaction Decoding By Bus Command .....	5-2
5.1.2	PCI Memory Transaction Decoding By Address Range .....	5-2
5.1.3	PCI I/O Transaction Decoding .....	5-3
5.1.4	ISA Master Considerations .....	5-4
5.2	PCI Transaction Details .....	5-5
5.2.1	Memory Access Range and Limitations .....	5-5
5.2.2	Bus Snooping on PCI to Memory Cycles .....	5-5
5.2.3	PCI to PCI Peer Transactions .....	5-5
5.2.4	PCI to System Memory Transactions .....	5-5
5.2.5	PCI to Memory Burst Transfer Completion .....	5-6
5.2.6	PCI to Memory Access Sequence .....	5-6
5.2.7	PCI to Memory Writes .....	5-6
5.2.7.1	Detailed Write Burst Sequence Timing .....	5-6

5.2.8 PCI to Memory Reads .....	5-8
5.2.8.1 Detailed Read Burst Sequence Timing .....	5-8
5.2.9 PCI BE# to CAS# Line Mapping .....	5-9
5.3 Bus Arbitration Logic .....	5-11
5.4 PCI Lock .....	5-12
5.4.1 PCI Busmaster Locks .....	5-12
5.4.2 CPU Bus Locking .....	5-12
5.5 PCI 2.1 Compliance .....	5-13
5.5.1 PCI Target Initial Latency .....	5-13
5.5.2 Transaction Ordering Rules .....	5-13
5.5.2.1 The Problem .....	5-13
5.5.2.2 Solution 1: Add a dummy CPU to PCI write: .....	5-14
5.5.2.3 Solution 2: Change the flag write procedure: .....	5-14
5.5.2.4 Solution 3: Change the data set write procedure: .....	5-14
5.6 Related Bridge Control Registers .....	5-15
<b>Section 6 ROM .....</b>	<b>6-1</b>
6.1 Direct-Attach ROM Mode .....	6-1
6.1.1 ROM Reads .....	6-2
6.1.1.1 ROM Read Sequence .....	6-2
6.1.1.2 Address, Transfer Size, and Alignment .....	6-4
6.1.1.3 Endian Mode Considerations .....	6-4
6.1.1.4 4-Byte Reads .....	6-4
6.1.2 ROM Writes .....	6-4
6.1.2.1 ROM Write Sequence .....	6-5
6.1.2.2 Write Protection .....	6-5
6.1.2.3 Data Flow In Little-Endian Mode .....	6-5
6.1.2.4 Data Flow In Big-Endian Mode .....	6-6
6.2 Remote ROM Mode .....	6-7
6.2.1 Remote ROM Reads .....	6-8
6.2.1.1 Remote ROM Read Sequence .....	6-8
6.2.1.2 Address, Transfer Size, and Alignment .....	6-11
6.2.1.3 Burst Reads .....	6-11
6.2.1.4 Endian Mode Considerations .....	6-11
6.2.1.5 4-Byte Reads .....	6-12
6.2.2 Remote ROM Writes .....	6-12
6.2.2.1 Write Sequence .....	6-12
6.2.2.2 Write Protection .....	6-12
6.2.2.3 Address, Size, Alignment, and Endian Mode .....	6-12
6.3 Related Bridge Control Registers .....	6-13
6.3.1 ROM Write Bridge Control Register .....	6-13
6.3.2 Direct-Attach ROM Lockout BCR .....	6-14
6.3.3 Remote ROM Lockout Bit .....	6-14
6.3.4 Other Related BCRs .....	6-15
<b>Section 7 Clocks .....</b>	<b>7-1</b>
7.1 CPU Clock Physical Design Rules .....	7-2



7.2 Clock Freezing .....	7-5
<b>Section 8 Exceptions .....</b>	<b>8-1</b>
8.1 Interrupts .....	8-1
8.1.1 Planar Interrupt Handler .....	8-1
8.1.2 MCM (660) INT_REQ and INT_CPU# .....	8-1
8.1.3 Interrupt Acknowledge Transactions .....	8-2
8.1.4 NMI_REQ .....	8-2
8.1.5 Interrupt Handling .....	8-2
8.1.6 Planar Interrupt Assignments .....	8-3
8.1.7 Scatter-Gather Interrupts .....	8-4
8.2 Errors .....	8-5
8.2.1 CPU Bus Related Errors .....	8-5
8.2.1.1 CPU Bus Error Types .....	8-5
8.2.1.2 CPU Bus Error Handling Protocol .....	8-6
8.2.2 PCI Bus Related Errors .....	8-6
8.2.2.1 PCI Bus Error Types .....	8-6
8.2.2.2 PCI Bus Error Handling Protocol .....	8-6
8.2.2.3 PCI Bus Data Parity Errors .....	8-6
8.2.3 CPU Bus Transaction Errors .....	8-7
8.2.3.1 CPU Bus Transfer Type or Size Error .....	8-7
8.2.3.2 CPU Bus XATS# Asserted Error .....	8-7
8.2.3.3 CPU to Memory Writes .....	8-8
8.2.3.4 CPU to Memory Reads .....	8-8
8.2.3.5 CPU Data Bus Parity Error .....	8-8
8.2.3.6 L2 Cache Parity Error .....	8-9
8.2.3.7 CPU Bus Write to Locked Flash .....	8-9
8.2.4 CPU to PCI Bus Transaction Errors .....	8-9
8.2.4.1 PCI Bus Data Parity Error While PCI Master .....	8-9
8.2.4.2 PCI Target Abort Received While PCI Master .....	8-10
8.2.4.3 PCI Master Abort Detected While PCI Master .....	8-10
8.2.5 PCI to Memory Transaction Errors .....	8-11
8.2.5.1 PCI to Memory Writes .....	8-11
8.2.5.2 PCI to Memory Reads .....	8-11
8.2.5.3 Out of Bounds PCI Memory Accesses .....	8-11
8.2.5.4 PCI Address Bus Parity Error While PCI Target .....	8-11
8.2.5.5 PCI Bus Data Parity Error While PCI Target .....	8-12
8.2.5.6 Errant Masters .....	8-12
8.2.6 Memory Transaction Errors .....	8-12
8.2.6.1 Memory Select Error .....	8-12
8.2.6.2 System Memory Parity Error .....	8-13
8.2.6.3 System Memory Single-Bit ECC Error .....	8-13
8.2.6.4 System Memory Multi-Bit ECC Error .....	8-14
8.2.7 SERR, I/O Channel Check, and NMI Errors .....	8-14
8.2.7.1 NMI_REQ Asserted Error .....	8-14

8.2.8 Error Reporting Protocol .....	8-15
8.2.8.1 Error Reporting With MCP# .....	8-15
8.2.8.2 Error Reporting With TEA# .....	8-15
8.2.8.3 Error Reporting With PCI_SERR# .....	8-16
8.2.8.4 Error Reporting With PCI_PERR# .....	8-16
8.2.9 Error Status Registers .....	8-16
8.2.10 Error-Related Bridge Control Registers .....	8-16
8.2.11 Special Events Not Reported as Errors .....	8-17
8.3 Resets .....	8-18
8.3.1 CPU Reset .....	8-18
8.3.2 660 Reset .....	8-18
8.3.2.1 Reset State of 660 Pins .....	8-18
8.3.2.2 660 Configuration Strapping .....	8-20
8.3.2.3 660 Deterministic Operation (Lockstep Applications) .....	8-20
8.4 Test Modes .....	8-20
8.4.1 CPU Test .....	8-20
8.4.2 660 Test .....	8-20
8.4.2.1 LSSD Test Mode .....	8-21
<b>Section 9 Set Up and Registers .....</b>	<b>9-1</b>
9.1 CPU Initialization .....	9-1
9.2 660 Bridge Initialization .....	9-1
9.3 PCI Configuration Scan .....	9-2
9.3.1 Multi-Function Adaptors .....	9-2
9.3.2 PCI to PCI Bridges .....	9-2
9.3.3 Indexed BCR Summary .....	9-2
<b>Section 10 System Firmware .....</b>	<b>10-1</b>
10.1 Introduction .....	10-1
10.2 Power On System Test .....	10-1
10.2.1 Hardware Requirements .....	10-1
10.3 Boot Record Format .....	10-1
10.3.1 Boot Record .....	10-2
10.3.1.1 PC Partition Table Entry .....	10-2
10.3.1.2 Extended DOS Partition .....	10-3
10.3.1.3 PowerPC Reference Platform Partition Table Entry .....	10-4
10.3.2 Loading the Load Image .....	10-4
10.4 System Configuration .....	10-6
10.4.1 System Console .....	10-6
10.4.2 System Initialization .....	10-7
10.4.3 Main Menu .....	10-8
10.4.3.1 System Configuration Menu .....	10-9
10.4.3.2 Run a Program .....	10-15
10.4.3.3 Reprogram Flash Memory .....	10-16
10.4.3.4 Exit Options .....	10-17
10.4.4 Default Configuration Values .....	10-17

<b>Section 11 Endian Mode Considerations</b> .....	<b>11-1</b>
11.1 What the 603e CPU Does .....	11-2
11.1.1 The 603e Address Munge .....	11-2
11.1.2 The 603e Data Shift .....	11-2
11.2 What the 660 Bridge Does .....	11-2
11.2.1 The 660 Bridge Address Unmunge .....	11-2
11.2.2 The 660 Bridge Data Swapper .....	11-2
11.3 Bit Ordering Within Bytes .....	11-4
11.4 Byte Swap Instructions .....	11-5
11.5 603e CPU Alignment Exceptions In LE Mode .....	11-5
11.6 Single-Byte Transfers .....	11-6
11.7 Two-Byte Transfers .....	11-10
11.8 Four-Byte Transfers .....	11-12
11.9 Three byte Transfers .....	11-14
11.10 Instruction Fetches and Endian Modes .....	11-15
11.11 Changing BE/LE Mode .....	11-16
11.12 Summary of Bi-Endian Operation and Notes .....	11-18
<b>Section 12 Electromechanical</b> .....	<b>12-1</b>
12.1 MCM Electrical .....	12-1
12.1.1 MCM Electrical Specifications .....	12-1
12.1.2 Absolute Maximum Rating Summary .....	12-2
12.1.3 Recommended Operating Conditions .....	12-2
12.1.4 DC Electrical Characteristics .....	12-2
12.1.5 AC Electrical Characteristics .....	12-3
12.1.6 SCM Package Delay Modeling .....	12-3
12.1.7 MCM Package Delay Modeling .....	12-4
12.1.8 MCM Net Electrical Model .....	12-5
12.1.9 MCM Paper Model .....	12-5
12.2 MCM Thermal .....	12-13
12.2.1 MCM Cooling Requirements .....	12-13
12.2.2 Module-Level Thermal Analysis .....	12-14
12.2.3 Heatsink Performance Requirements [R(hs)] .....	12-14
12.3 MCM Mechanical Drawings .....	12-16
12.3.1 MCM with Cap .....	12-16
12.3.2 MCM Without Cap .....	12-16
12.3.3 MCM Solder Columns .....	12-17
12.3.4 MCM Component Placement .....	12-18
<b>Section 13 MCM Schematics</b> .....	<b>13-1</b>
13.1 Component Placement .....	13-2

<b>Appendix A Example Planar Overview .....</b>	<b>A-1</b>
A.1 The Collateral Material .....	A-1
A.2 Example Planar .....	A-2
A.2.1 System Memory .....	A-3
A.2.2 PCI Bus .....	A-3
A.2.3 Flash ROM .....	A-3
A.2.4 ISA Bus .....	A-3
A.2.5 System I/O EPLD .....	A-3
A.2.6 X-Bus Agents .....	A-3
A.2.7 Power Management .....	A-3
A.3 Other Help .....	A-4
A.3.1 Example Firmware .....	A-4
A.3.2 Design Files .....	A-4
A.3.3 Quickstart Peripheral List .....	A-4
<b>Appendix B Planar I/O Subsystems .....</b>	<b>B-1</b>
B.1 ISA Subsystem .....	B-1
B.1.1 The ISA Bridge .....	B-1
B.1.2 Address Ranges .....	B-2
B.1.3 ISA Bus Concurrency .....	B-2
B.1.4 ISA Busmasters and IGN_PCI_AD31 .....	B-2
B.1.5 DMA .....	B-3
B.1.5.1 Supported DMA Paths .....	B-3
B.1.5.2 DMA Timing .....	B-4
B.1.5.3 Scatter-Gather .....	B-4
B.1.5.4 X-Bus .....	B-4
B.1.5.5 Control Signal Decodes .....	B-4
B.1.5.6 Keyboard/Mouse Controller .....	B-4
B.1.5.7 Real Time Clock (RTC) .....	B-4
B.1.5.8 PCI Adapter Card Presence Detect Register .....	B-5
B.1.5.9 L2 SRAM Identification Register .....	B-5
B.1.5.10 Planar ID Detection Register .....	B-5
B.1.5.11 DRAM Presence Detection .....	B-6
B.1.5.12 DRAM SIMM 1-2 Memory ID Register .....	B-6
B.1.5.13 DRAM SIMM 3-4 Memory ID Register .....	B-6
B.1.6 Miscellaneous .....	B-7
B.1.6.1 Speaker Support .....	B-7
B.2 System EPLD .....	B-7
B.2.1 System Register Support .....	B-7
B.2.1.1 External Register Support .....	B-7
B.2.1.2 Internal Registers .....	B-8
B.2.2 Signal Descriptions .....	B-10
B.2.3 EPLD Design Equations .....	B-12
B.2.3.1 Fit File .....	B-12
B.2.3.2 TDF File .....	B-14

<b>Appendix C Planar Set Up and Registers</b> .....	<b>C-1</b>
C.1 ISA Bridge (SIO) Initialization .....	C-1
C.1.1 Summary of SIO Configuration Registers .....	C-4
C.2 Example Planar Combined Register Listing .....	C-5
C.2.1 Direct Access Registers .....	C-5
C.3 ISA Bus Register Suggestions .....	C-10
<b>Appendix D Planar Electromechanical</b> .....	<b>D-1</b>
D.1 Electrical .....	D-1
D.1.1 Power Requirements .....	D-1
D.1.2 Onboard 3.3V Regulator .....	D-2
D.2 Physical Design Rules .....	D-3
D.2.1 Construction .....	D-3
D.2.2 General Wiring Guidelines .....	D-5
D.3 CPU Bus Nets .....	D-5
D.4 PCI Bus Nets .....	D-5
D.5 Example Planar Mechanical .....	D-6
D.5.1 Example Planar Connectors .....	D-7
D.5.2 Example Planar Connector Locations .....	D-8
D.5.3 Keyboard Connector J14 .....	D-9
D.5.4 Alternate Keyboard Connector J14A .....	D-9
D.5.5 Mouse Connector J15 .....	D-10
D.5.6 Speaker Connector J13 .....	D-10
D.5.7 Power Good LED/KEYLOCK# Connector J12 .....	D-11
D.5.8 HDD LED Connector J11 (1 x 2 Berg) .....	D-11
D.5.9 Reset Switch Connector J10 (1 x 2 Berg) .....	D-11
D.5.10 Fan Connector J9, J35 .....	D-12
D.5.11 3.3V Power Connector J5 .....	D-12
D.5.12 Power Connector J4 .....	D-12
D.5.13 AUX5/ON-OFF Connector J6 .....	D-13
D.5.14 PCI Connectors J19, J25, J26, and J27 .....	D-13
D.5.15 ISA Connectors J29, J30, J31, J32 .....	D-15
D.5.16 DRAM SIMM Connectors J21, J22, J23, and J24 .....	D-16
D.5.17 Power Switch Connector J8 .....	D-18
D.5.18 Power Up Configuration Connector J7 .....	D-18
D.5.19 RISCWatch Connector J2 .....	D-19
D.5.20 Battery Connector BT2 .....	D-19
D.5.21 MCM Board Connector Footprint #1 .....	D-20
D.5.22 MCM Board Connector Footprint #2 .....	D-21
D.6 Enclosure .....	D-22
<b>Appendix E Bill of Materials</b> .....	<b>E-1</b>
E.1 Planar Bill of Materials .....	E-1
<b>Appendix F Planar Schematics</b> .....	<b>F-1</b>
<b>Appendix G Data Sheets</b> .....	<b>G-1</b>

## Figures

Figure 1-1.	MCM Block Diagram .....	1-2
Figure 1-2.	MCM C .....	1-5
Figure 1-3.	Chip Wire Bond and Flip Chip Connection .....	1-5
Figure 1-4.	MCM Flip Chips and Internal Wiring .....	1-6
Figure 1-5.	MCM Solder Columns .....	1-6
Figure 3-1.	Non-Contiguous PCI I/O Address Transformation .....	3-5
Figure 3-2.	Non-Contiguous PCI I/O Address Translation .....	3-6
Figure 3-3.	Contiguous PCI I/O Address Translation .....	3-6
Figure 3-4.	L2 Mapping of System Memory – 512K Configuration .....	3-12
Figure 3-5.	SLC L2 Cache Directory – 512K Configuration .....	3-13
Figure 3-6.	Synchronous SRAM, 512K L2 .....	3-14
Figure 3-7.	Synchronous TagRAM, 512K L2 .....	3-14
Figure 4-1.	CPU to Memory Transfer Timing Parameters .....	4-3
Figure 4-2.	DRAM Logical Implementation .....	4-16
Figure 4-3.	Example Memory Bank Configuration .....	4-22
Figure 4-4.	CPU Read Data Flow .....	4-25
Figure 4-5.	PCI Read Data Flow .....	4-25
Figure 4-6.	CPU 8-Byte Write Data Flow .....	4-25
Figure 4-7.	PCI 8-Byte Write Data Flow .....	4-26
Figure 4-8.	PCI or CPU Read-Modify-Write Data Flow .....	4-26
Figure 4-9.	DRAM Refresh Timing Diagram .....	4-31
Figure 6-1.	ROM Connections .....	6-2
Figure 6-2.	ROM Read Timing Diagram .....	6-3
Figure 6-3.	ROM Connections .....	6-5
Figure 6-4.	ROM Data and Address Flow In Little Endian Mode .....	6-6
Figure 6-5.	ROM Data and Address Flow In Big Endian Mode .....	6-7
Figure 6-6.	Remote ROM Connections .....	6-8
Figure 6-7.	Remote ROM Read – Initial Transactions .....	6-9
Figure 6-8.	Remote ROM Read – Final Transactions .....	6-10
Figure 6-9.	Remote ROM Write .....	6-13
Figure 7-1.	MCM Clocks .....	7-1
Figure 7-2.	MCM Clocks .....	7-4
Figure 7-3.	CPU_CLK to PCI_CLK Skew .....	7-4
Figure 8-1.	Conceptual Block Diagram of INT Logic .....	8-2
Figure 8-2.	Interrupt Handling .....	8-2
Figure 8-3.	PCI Interrupt Connections .....	8-4
Figure 10-1.	Boot Record .....	10-2
Figure 10-2.	Partition Table Entry .....	10-2
Figure 10-3.	Partition Table Entry Format for an Extended Partition .....	10-4
Figure 10-4.	Partition Table Entry for PowerPC Reference Platform .....	10-4
Figure 10-5.	PowerPC Reference Platform Partition .....	10-5
Figure 10-6.	System Initialization Screen .....	10-7

Figure 10-7.	Configuration Utility Main Menu .....	10-8
Figure 10-8.	System Configuration Menu .....	10-9
Figure 10-9.	System Information Screen .....	10-10
Figure 10-10.	Device Configuration Screen .....	10-11
Figure 10-11.	SCSI Devices Screen .....	10-12
Figure 10-12.	Boot Devices Screen .....	10-13
Figure 10-13.	Set Date and Time Screen .....	10-14
Figure 10-14.	Run a Program Screen .....	10-15
Figure 10-15.	Reprogram the Flash Memory Screen .....	10-16
Figure 11-1.	Endian Mode Block Diagram .....	11-3
Figure 11-2.	Example at Address xxxx xxx0 .....	11-6
Figure 11-3.	Example at Address xxxx xxx2 .....	11-7
Figure 11-4.	Double Byte Write Data ab at Address xxxx xxx0 .....	11-10
Figure 11-5.	Word (4-Byte) Write of 0a0b0c0dh at Address xxxx xxx4 .....	11-12
Figure 11-6.	Instruction Alignment Example .....	11-15
Figure 11-7.	Wrong Instruction Read When Unmunger is used .....	11-16
Figure 11-8.	Instruction Stream to Switch Endian Modes .....	11-17
Figure 12-1.	660 Package QFP Paper Model .....	12-3
Figure 12-2.	MCM Net Electrical Model .....	12-5
Figure 12-3.	MCM Thermal Paths .....	12-13
Figure 12-4.	MCM Heat Flows .....	12-13
Figure A-1.	100 MHz PPC 603e Example Planar Block Diagram .....	A-2
Figure B-1.	Typical External Register .....	B-7
Figure D-1.	Signal and Power Layers .....	D-3
Figure D-2.	Typical Wiring Channel Top View .....	D-3
Figure D-3.	PowerPC 603/604 Board Fabrication .....	D-4
Figure D-4.	Power Plane Split .....	D-5
Figure D-5.	The Keyboard Connector .....	D-9
Figure D-6.	The Alternate Keyboard Connector .....	D-9
Figure D-7.	The Mouse Connector .....	D-10
Figure D-8.	1x4 Speaker Connector .....	D-10
Figure D-9.	1x5 Power Good LED Connector .....	D-11
Figure D-10.	1x2 HDD LED Connector .....	D-11
Figure D-11.	1x2 Reset Switch Connector .....	D-11
Figure D-12.	1x2 Fan Connector .....	D-12
Figure D-13.	1x6 3.3V Power Connector J5 .....	D-12
Figure D-14.	1x12 Power Connector .....	D-12
Figure D-15.	AUX5/ON-OFF Connector .....	D-13
Figure D-16.	PCI Connector .....	D-13
Figure D-17.	ISA Connector .....	D-15
Figure D-18.	DRAM SIMM Connector .....	D-16
Figure D-19.	1x2 Power Switch Connector .....	D-18
Figure D-20.	1x2 Power Up Configuration Connector .....	D-18
Figure D-21.	2x8 RISCWatch Connector .....	D-19

---

**Tables**


---

Table 2-1.	Example Signal Table .....	2-1
Table 2-2.	User Supplied Signals .....	2-2
Table 2-3.	Clock Signals .....	2-3
Table 2-4.	CPU Bus Signals .....	2-4
Table 2-5.	PCI Bus Signals .....	2-7
Table 2-6.	DRAM Signals .....	2-8
Table 2-7.	L2 Signals .....	2-9
Table 2-8.	System Interface Signals .....	2-10
Table 2-9.	InterChip Communication Signals .....	2-11
Table 2-10.	Test Signal Descriptions .....	2-13
Table 2-11.	Pin List With Net Names and MCM Nodes .....	2-14
Table 3-1.	TT[0:3] (Transfer Type) Decoding by 660 .....	3-3
Table 3-2.	660 Address Mapping of CPU Bus Transactions .....	3-4
Table 3-3.	CPU to PCI Configuration Mapping .....	3-9
Table 3-4.	L2 Cache Responses to CPU Bus Cycles .....	3-10
Table 3-5.	L2 Operations for PCI to Memory Transactions, 603 Mode ...	3-11
Table 3-6.	Other L2 Related BCRs .....	3-11
Table 4-1.	Memory Timing Parameters .....	4-3
Table 4-2.	Page Mode DRAM Aggressive Timing Summary (1) .....	4-11
Table 4-3.	EDO DRAM Aggressive Timing Summary (1) .....	4-12
Table 4-4.	Page Mode DRAM Conservative Timing Summary (1) .....	4-13
Table 4-5.	EDO DRAM Conservative Timing Summary (1) .....	4-14
Table 4-6.	Supported SIMM Topologies .....	4-16
Table 4-7.	Row Addressing (CPU Addressing) .....	4-17
Table 4-8.	Column Addressing (CPU Addressing) .....	4-17
Table 4-9.	Row Addressing (PCI Addressing) .....	4-17
Table 4-10.	Column Addressing (PCI Addressing) .....	4-17
Table 4-11.	Example Memory Bank Addressing Mode Configuration .....	4-21
Table 4-12.	Example Memory Bank Starting and Ending Address Configuration .....	4-23
Table 4-13.	Bridge Response to Various PCI Write Data Phases .....	4-28
Table 4-14.	Bridge Response to Best Case PCI Write Burst .....	4-28
Table 4-15.	Bridge Response to Case 2 PCI Write Burst .....	4-28
Table 4-16.	Bridge Response to Various PCI Write Bursts .....	4-29
Table 4-17.	Check Bit Calculation .....	4-29
Table 4-18.	Syndrome Decode .....	4-30
Table 4-19.	Typical DRAM Module Maximum Input Capacitance .....	4-33
Table 5-1.	MCM Response to PCI_C[3:0] Bus Commands .....	5-2
Table 5-2.	Mapping of PCI Memory Space, Part 1 .....	5-3
Table 5-3.	Mapping of PCI Memory Space, Part 2 .....	5-3
Table 5-4.	Mapping of PCI Master I/O Transactions .....	5-4
Table 5-5.	PCI to Memory Write Burst Sequence Timing .....	5-7



Table 5-6.	PCI to Memory Read Burst Sequence Timing .....	5-9
Table 5-7.	Active CAS# Lines – PCI to Memory Writes, BE or LE Mode .	5-10
Table 6-1.	ROM Read Data and Address Flow .....	6-4
Table 6-2.	ROM Write Data Flow in Little-Endian Mode .....	6-6
Table 6-3.	ROM Write Data Flow in Big-Endian Mode .....	6-7
Table 6-4.	Remote ROM Read Sequence, CPU Address = FFFX XXX0 .	6-11
Table 6-5.	ROM Write BCR Contents .....	6-14
Table 7-1.	Clock Net Lengths .....	7-2
Table 7-2.	Clock Net Calculations .....	7-3
Table 8-1.	Mapping of PCI Memory Space, Part 1 .....	8-4
Table 8-2.	Invalid CPU Bus Operations .....	8-7
Table 8-3.	664 Pin Reset State .....	8-19
Table 8-4.	663 Pin Reset State .....	8-19
Table 8-5.	Configuration Strapping Options .....	8-20
Table 8-6.	LSSD Test Mode Pin Definitions .....	8-21
Table 9-1.	Configuration Address Assignments .....	9-2
Table 9-2.	660 Bridge Indexed BCR Listing .....	9-2
Table 11-1.	Endian Mode Operations .....	11-1
Table 11-2.	603e LE Mode Address Transform .....	11-2
Table 11-3.	660 Bridge Endian Mode Byte Lane Steering .....	11-3
Table 11-4.	660 Bit Transfer .....	11-4
Table 11-5.	Memory in BE Mode .....	11-7
Table 11-6.	Memory in LE Mode .....	11-8
Table 11-7.	PCI in BE Mode .....	11-8
Table 11-8.	PCI in LE Mode .....	11-9
Table 11-9.	Two Byte Transfer Information .....	11-11
Table 11-10.	Rearranged Two-Byte Transfer Information .....	11-11
Table 11-11.	Four-Byte Transfer Information .....	11-13
Table 11-12.	Rearranged Four-Byte Transfer Information .....	11-14
Table 12-1.	Component Voltage Specifications .....	12-2
Table 12-2.	Electrical Model Range of Values .....	12-4
Table 12-3.	MCM Primary I/O .....	12-6
Table 12-4.	MCM Die Thermal Specifications .....	12-14
Table 12-5.	Thermal Resistance From Cap to Heat Sink .....	12-14
Table 12-6.	Heatsink Performance Requirements .....	12-15
Table A-1.	Quickstart Peripheral List .....	A-4
Table B-1.	DMA Assignments .....	B-3
Table B-2.	DRAM Module Presence Detect Bit Encoding .....	B-5
Table B-3.	Planar ID Encoding .....	B-6
Table B-4.	DRAM Module Presence Detect Bit Encoding .....	B-7
Table B-5.	External Register Support .....	B-8
Table B-6.	Signal Descriptions .....	B-10
Table C-1.	Summary of SIO Register Setup (Configuration Address = 8080 08xx) .....	C-2

Table C-2.	Summary of SIO Configuration Registers .....	C-4
Table C-3.	Combined Register Listing .....	C-6
Table C-4.	Compatible ISA Ports (Not on Reference Board) .....	C-11
Table D-1.	Power Supply Specification .....	D-1
Table D-2.	Approximate Power Consumption .....	D-1
Table D-3.	Specifications for 3.3V Regulator on the Motherboard .....	D-2
Table D-4.	Keyboard Connector Pin Assignments .....	D-9
Table D-5.	Alternate Keyboard Connector Pin Assignments .....	D-9
Table D-6.	Mouse Connector Pin Assignments .....	D-10
Table D-7.	Speaker Connector Pin Assignments .....	D-10
Table D-8.	Power Good LED Connector .....	D-11
Table D-9.	HDD LED Connector .....	D-11
Table D-10.	Reset Switch Connector .....	D-11
Table D-11.	Fan Connector Pin Assignments .....	D-12
Table D-12.	3.3V Power Connector J5 Pin Assignments .....	D-12
Table D-13.	Power Connector J4 Pin Assignments .....	D-12
Table D-14.	AUX5/ON-OFF Connector Pin Assignments .....	D-13
Table D-15.	PCI Connector Pin Assignments .....	D-13
Table D-16.	ISA Connector Pin Assignments .....	D-15
Table D-17.	DRAM SIMM Connector Pin Assignments .....	D-16
Table D-18.	Power Switch Connector Pin Assignments .....	D-18
Table D-19.	Power Up Configuration Connector Pin Assignments .....	D-18
Table D-20.	RISCWatch Connector Pin Assignments .....	D-19
Table D-21.	Height Considerations .....	D-22
Table E-1.	Planar Bill of Materials .....	E-1

---

## About This Book

---

This document is designed for engineers and system designers who are interested in implementing PowerPC systems that use the 100 MHz PPC 603e MCM. The material requires a detailed understanding of computer systems at the hardware and software level.

Power management is beyond the scope of this document. This document was written by Dale Elson.

### Reference Material:

Understanding of the relevant areas of the following documents is required for a good understanding of the 100 MHz PPC 603e MCM:

- *PowerPC 603e RISC Microprocessor User's Manual*, IBM document MPR603EUM-01
- *PowerPC 603e Hardware Specification*, IBM document MPR603EHS-01
- *PowerPC 603e Technical Summary*, IBM document MPR603TSU-04
- *IBM27-82660 PowerPC to PCI Bridge User's Manual*, IBM document number MPR660UMU-01
- *PCI Local Bus Specification*, Revision 2.1, available from the PCI SIG
- *PowerPC Reference Platform Specification*, Version 1.1, IBM document MPRPRPPKG
- *The Power PC Architecture*, second edition, Morgan Kaufmann Publishers (800) 745-7323, IBM document MPRPPCARC-02
- *Intel 82378ZB System I/O (SIO) Data Book*, Intel order number 290473-004.

The following documents are useful as sources of tutorial and supplementary information about the MCM.

- *PowerPC System Architecture*, Tom Shanley, Mindshare Press (800) 420-2677.
- *IBM27-82650 PowerPC to PCI Bridge User's Manual*, IBM document number MPR650UMU-01

### Document Conventions:

The terms 660 and 660 bridge refer to the IBM27-82660.

The terms 660 UM and 660 User's Manual refer to the current version of the *IBM27-82660 PowerPC to PCI Bridge User's Manual*.

The terms 603e UM and 603e User's Manual refer to the *IBM PowerPC 603e RISC Microprocessor User's Manual*.

Kilobytes, megabytes, and gigabytes are indicated by a single capital letter after the numeric value. For example, 4K means 4 kilobytes, 8M means 8 megabytes, and 4G means 4 gigabytes.

The terms DIMM and SIMM are often used to mean DRAM module.

Hexadecimal values are identified (where not clear from context) with a lower-case letter h at the end of the value. Binary values are identified (where not clear from context) with a lower-case letter b at the end of the value.

In identifying ranges of values *from* and *to* are used whenever possible. The range statement from 0 to 2M means from and including zero up to (but not including) two megabytes. The hexadecimal value for the range from 0 to 64K is: 0000h to FFFFh.

The terms *asserted* and *negated* are used extensively. The term *asserted* indicates that a signal is active (logically true), regardless of whether that level is represented by a high or low voltage. The term *negated* means that a signal is not asserted. The # symbol at the end of a signal name indicates that the active state of the signal occurs with a low voltage level.

This image shows a full page of blank graph paper. The grid consists of small, equal-sized squares formed by thin black lines. There are no margins, text, or other markings on the page.This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

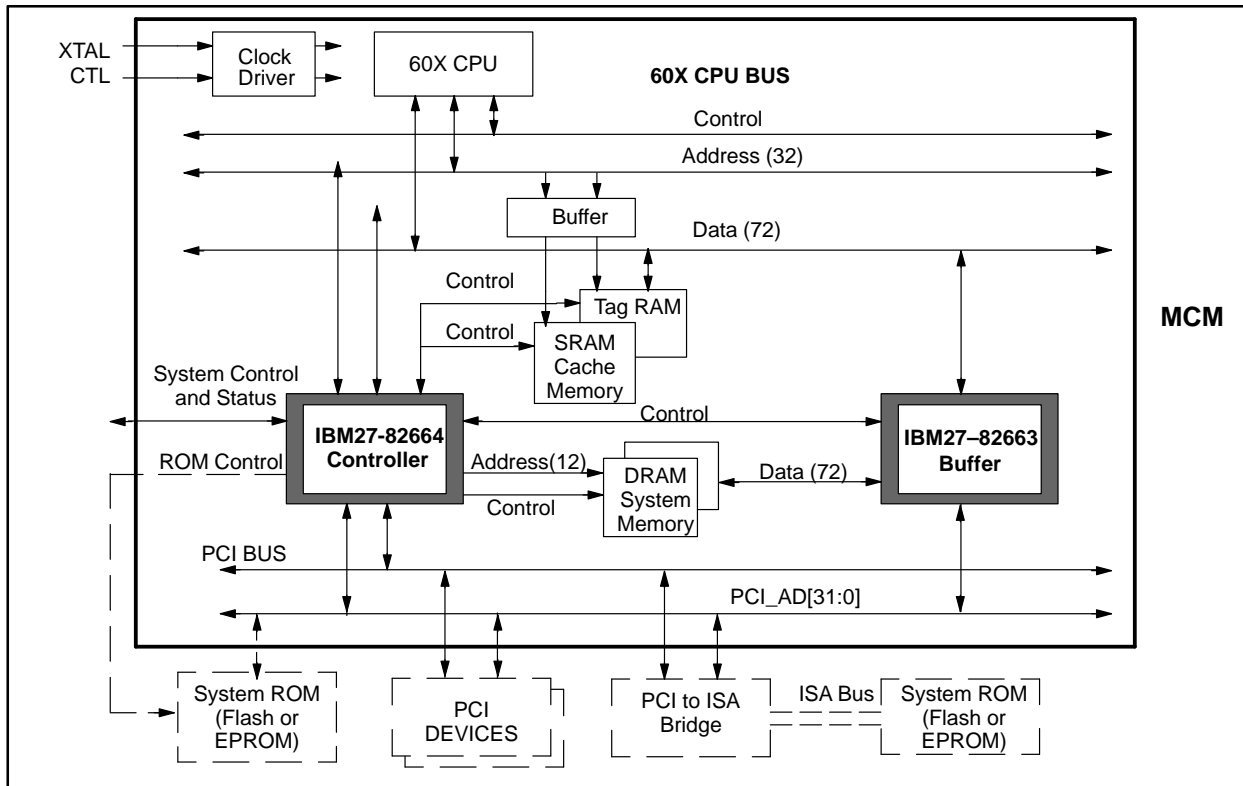
## **Section 1**

### **Introduction**

This document provides a detailed technical description of the 100 MHz PPC 603e Multi-Chip Module (MCM), and is intended to be used by hardware, software, test, and simulation engineers as a first source of information. Software developers should read through the entire document because information relevant to their tasks may be located in hardware sections.

#### **1.1 The MCM**

The MCM consists of the CPU, L2, PCI Bridge, Memory Controller and master clock elements of a PowerPC system. The chips that provide this function are all packaged on a single ceramic substrate.

**1.2 100 MHz PPC 603e MCM****Figure 1-1. MCM Block Diagram**

The MCM (see Figure 1-1) is compliant with the PowerPC Reference Platform Specification Version 1.1, and the PCI Local Bus Specification, revision 2.0/2.1 (see Section 5.5) for host bridges.

The MCM uses the PowerPC 603e™ RISC microprocessor. The IBM27-82660 Bridge chip-set (660 Bridge or 660) interfaces the CPU to the PCI bus and DRAM memory, and provides L2 cache control. The tagRAM and 512K SRAM components are also included on the MCM. The MCM also includes a master clock generator. For a list of the specific dice packaged within the MCM, see Table 1-1.

**Table 1-1. MCM Dice**

Die	Part Number	Vendor
PowerPC 603e Processor		IBM
663 Buffer		IBM
664 Controller		IBM
Clock Driver	MPC970	Motorola
Tag RAM	IDT1216	Integrated Device Technology
Buffer	SCBS143F	Texas Instruments
Synchronous Burst SRAM		IBM

### **1.2.1 The CPU**

The MCM uses a PowerPC 603e (PID6–603e) at CPU bus speeds up to 66MHz. See the PowerPC 603e Technical Summary, Users Manual, and Hardware Specifications for more information. The MCM:

- Runs at 3:2 CPU\_core:CPU\_bus speed ratio up to 99:66 MHz. Consult your IBM representative for available choices of CPU and operating frequency.
- Supports one level of address bus pipelining. Most data writes are posted.
- Reports precise exceptions via TEA#, and reports imprecise exceptions via MCP#.
- Operates the 603e in 64-bit data bus mode.
- Supports bi-endian operation.
- Supplies an ESP connector for RISCWatch debugging and monitor systems.
- Runs in DRTRY# mode.

### **1.2.2 L2 Cache**

The MCM contains a complete L2. The controller is located inside the 660 Bridge. The L2:

- Uses 512K SRAM and a 16K x 15 synchronous tagRAM.
- Is unified, write-thru, direct-mapped, look-aside.
- Caches system memory from 0 to 1G.
- Maintains coherence for 32-byte blocks, the PowerPC 60X coherence unit.
- Supplies data to the CPU on burst read hits and snarfs the data on CPU burst read/write misses. The L2 is not updated during a PCI transaction.
- Ignores CPU single-beat reads, and invalidates on CPU bus single-beat write hits.
- Snoops PCI to memory transactions, invalidates on PCI write hits. The L2 does not supply data to the PCI on read hits.

### **1.2.3 System Memory Interface**

The MCM memory controller, located in the 660, provides the system DRAM memory interface as required by the PowerPC Architecture™. System memory can be accessed from both the CPU bus and the PCI bus. The system memory interface:

- Has a 72-bit wide data path – 64 data bits and 8 bits of optional ECC or parity data
- Supports page mode and EDO (hyper-page mode) DRAM
- Supplies 8 RAS# outputs, 8 CAS# outputs, and 2 WE# outputs, for up to 8 SIMM banks
- Supports 8-byte, 168-pin and 4-byte, 72-pin SIMMs for up to 1G of DRAM
- Allows mixed use of different size SIMMs, including mixed 4-byte and 8-byte SIMMs
- Includes full refresh support, including refresh counter and programmable timer
- Implements burst-mode memory address generation for both CPU and PCI bursts
- Implements both little-endian and big-endian addressing and byte swapping modes
- Provides row and column address multiplexing for 10x10 thru 12x12 RxC DRAMs

### **1.2.4 The PCI Interface**

The MCM allows CPU to PCI access and PCI busmaster to system memory access (with snooping), and handles all PCI related system memory and cache coherency issues.

The MCM supplies a 32-bit PCI host bridge interface that is:

- Compliant with the PCI Specification, revisions 2.0 and 2.1 (see Section 5.5)
- 3.3v and 5v signalling environment compliant.

- Operates up to 33MHz, at 1/2 of the CPU bus frequency.
- Supports a tertiary IO bus bridge, including support for ISA masters.
- Allows system memory block locking by PCI busmasters
- Supports type 0 and type 1 PCI configuration cycles

1.2.5 System Clocks

The MCM uses a Motorola™ MPC970 PLL clock generator to provide the clocks required by the MCM components, and six clocks for system use. A 16.5 MHz quartz crystal (seed clock) provided by the user is used as an input to the clock generator. It then produces the system and PCI clocks.

1.2.6 MCM Performance

Minimum Cycle Times For Pipelined CPU to Memory Transfers at 66 MHz

Responding Device	Read	Write
L2 (9ns Synchronous SRAM)	-2-1-1-1	Snarf
L2 (15ns Asynchronous SRAM)	-3-2-2-2	Snarf
Page DRAM (70ns) Pipelined	-4-4-4-4	-3-3-4-4
EDO DRAM (60ns) Pipelined	-5-3-3-3	-3-3-3-3

Typical PCI to Memory Performance at 66 MHz CPU Clock and 33MHz PCI Clock

Read	8-1-1-1 -1-1-1-1 7-1-1-1 -1-1-1-1 7-1-1-1 -1-1-1-1 ... 7-1-1-1 -1-1-1-1
Write	5-1-1-1 -3-1-1-1 3-1-1-1 -3-1-1-1 3-1-1-1 -3-1-1-1 ... 3-1-1-1 -3-1-1-1



### 1.3 MCM Technology Overview

The IBM Multi-Chip Module (MCM) is shown in Figure 1-2. The MCM consists of a 45mm x 45mm cofired dark ceramic substrate upon which are mounted eleven unpackaged semiconductor dice (chips), capacitors, and resistors. The top of the substrate and the chips are completely covered by an aluminum cap, which is non-hermetically sealed to the substrate. A thermally conductive grease fills the area between the cap and the chips. The MCM is electrically connected to the circuit board (planar) by solder columns.

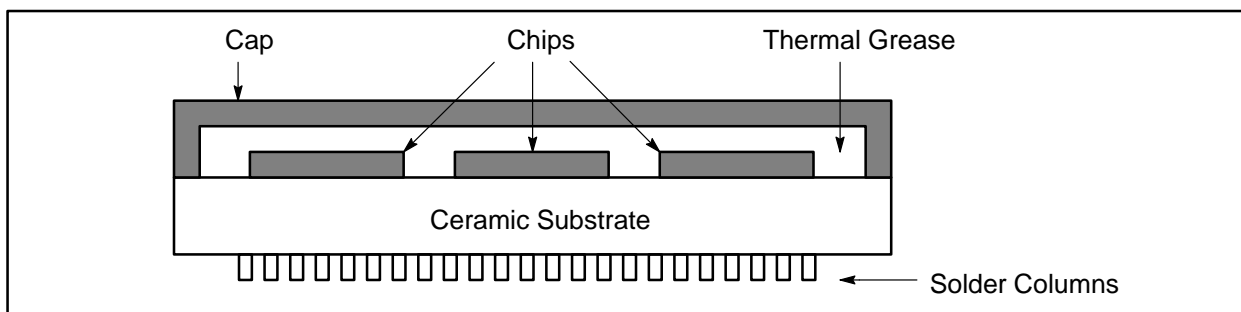


Figure 1-2. MCM C

#### 1.3.1 Chip Attachment to MCM

All die on the MCM are attached to the ceramic substrate using IBM's C4-based flip chip attachment technology. In this technology, C4 solder balls provide an interconnection between the die and the substrate. For those vendor die originally designed for wire bond base interconnection, additional processing steps were applied to allow the C4 interconnection. Figure 1-3 shows a typical wire-bond die. In wire bond technology, the wafer is sliced into individual dies, each die is attached to a wire frame, and (for each chip pad) one end of a wire is bonded to the chip pad, and the other end is bonded to the package pin. The MCM uses the "flip-chip" process. An additional metalization layer is added to the chip, the chip is "flipped" over, and C4 balls are used to connect the chip pads to the MCM substrate pads.

Unpackaged die have a much smaller footprint than packaged die. This reduces the overall size of the MCM circuitry and allows the MCM a much smaller footprint than would be required by the same circuitry implemented on a planar in discrete packages.

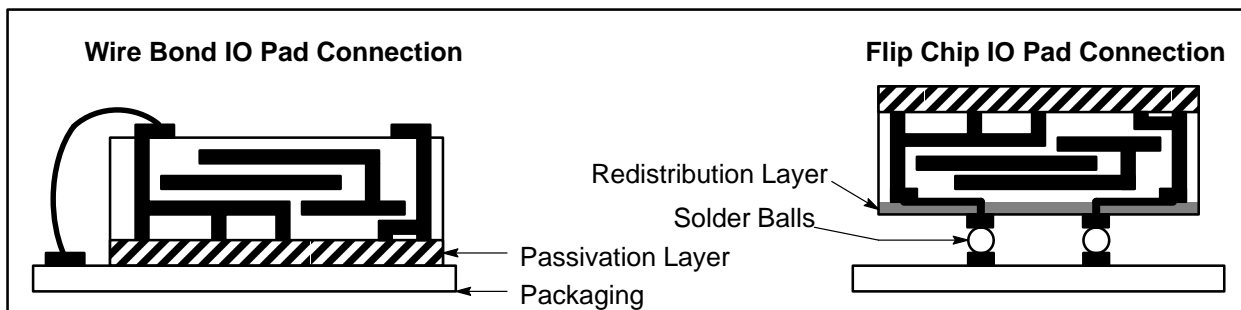


Figure 1-3. Chip Wire Bond and Flip Chip Connection

Figure 1-4 shows chips mounted to the MCM substrate. Figure 1-4 also illustrates the internal structure of the MCM substrate. The MCM acts as the circuit board between the various chips. Structures analogous to printed circuit board traces and vias form the electrical interconnections between the various MCM chips. The trace lengths on the MCM are much shorter than are achievable on a planar. This means that signal propagation (flight times) between the devices are minimized, which can allow faster operating frequencies than can be achieved on a planar. The impedance of the various nets can be more precisely controlled, which contributes to high signal quality and low noise.

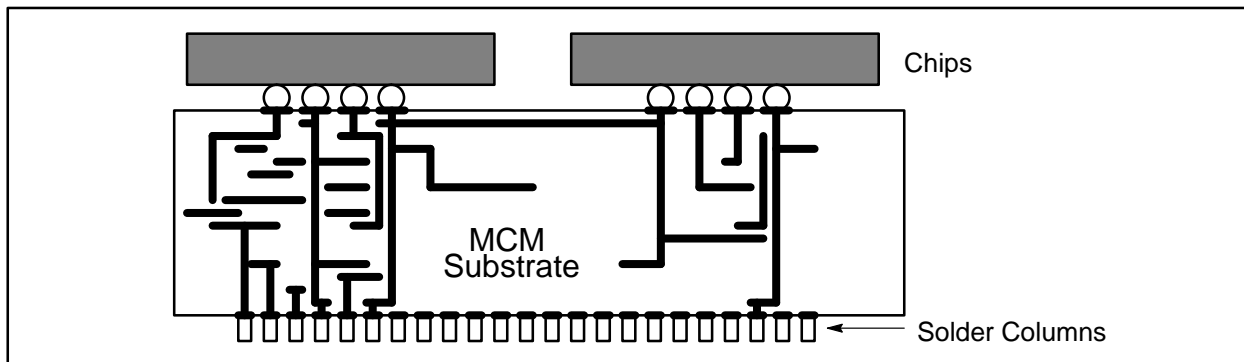


Figure 1-4. MCM Flip Chips and Internal Wiring

### 1.3.2 Substrate Attachment to the Planar

Figure 1-5 shows how solder columns are used to connect the MCM to the planar. Each column is attached to the MCM by an eutectic solder joint. The solder at this point melts at a higher temperature than the solder used for the column itself. The MCM is then soldered to the planar, again using an eutectic solder. The solder column process is robust, and offers a lower impedance. It is also one of the higher density interconnection methods.

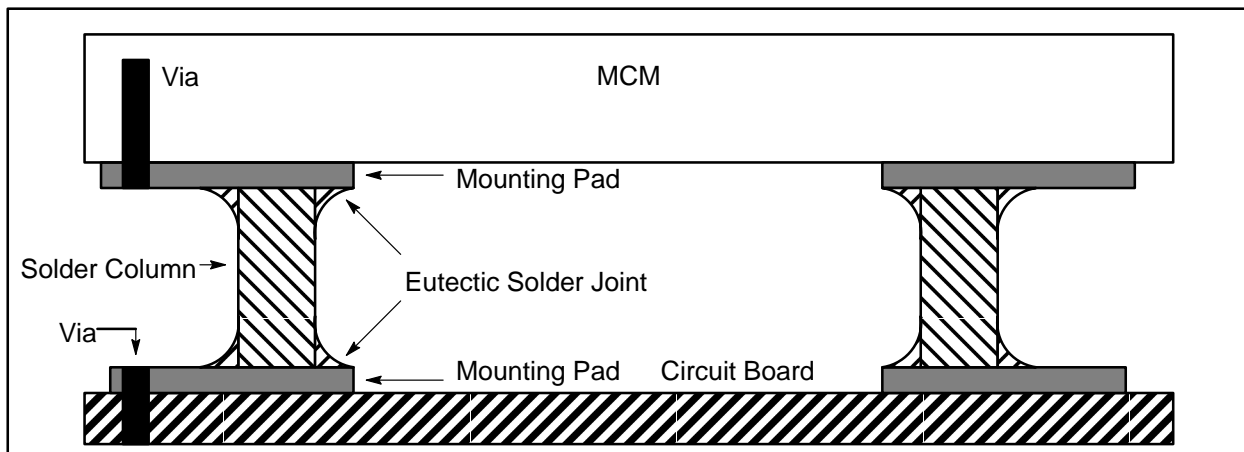


Figure 1-5. MCM Solder Columns

For more information on solder columns, assembly, and rework, see *"Ceramic Ball and Column Grid Array Surface Mount Assembly and Rework"*, IBM document number APD-SBSC-101.0.

## **1.4 Incorporating the MCM into a System Design**

The MCM captures, in one ceramic package, all key components of a PowerPC system design that resides on a CPU's local bus. The MCM's primary ports of interface to the rest of a system are the PCI bus and the system memory (DRAM) port. Other signals that connect the MCM to a system fall into the following groups:

- Processor PLL configuration
- MPC 970 configuration
- Signal nets broken at the MCM level for test purposes and need to be reconnected on the planar
- RISCWatch interface
- 60X bus signals requiring pullup/pulldown connections at the planar
- L2 cache configuration and presence detect
- Clock nets broken at the MCM level to allow net length matching with planar resident clock nets
- ROM control signals
- Local bus signals used for control functions by the planar.

Understanding all of these groups is vital to the use of the MCM. Pay particular attention to those signals broken at the MCM level and which are reconnected at the planar. Those nets are associated with MCM I/O pairs, the names of which are prefaced by the letter "X."

When applying the MCM to a system design, it is recommended that the designer review the schematics for both the MCM (see Section 13) and the example planar (see Appendix F). Although the example planar is a specific design implementation of the MCM, most MCM interface signals are used in a manner that would be common to any PowerPC system design. A review of the planar schematics and MCM and planar component data sheets (see Appendix G) will quickly answer most questions concerning how to interface with the MCM I/O.

Note that it is not intended for the user to interface with all of the MCM I/O. Section 2 contains I/O listings that distinguish between signals intended for application use and those broken out only for manufacturing purposes.

[illegible][illegible]

## Section 2

### Signal Descriptions

This section describes the connectivity of the 100 MHz PPC 603e MCM. Tables are used extensively for this purpose (see Table 2-1 and the following paragraphs for an explanation).

The terms *asserted* and *active* indicate that a signal is logically true, regardless of the voltage level. The terms *negated*, *inactive*, and *deasserted* mean that a signal is logically false. The # symbol at the end of a signal name indicates that the active or asserted state of the signal occurs with a low voltage level. Otherwise the signal is active at a high voltage level.

Table 2-1. Example Signal Table

Signal Name	MCM	Nodes	Description
AACK#	R08	R1.A02 U1.28 U2.109 pu I/O I/O	CPU bus address acknowledge. The CPU bus target (660 or external CPU bus target) asserts AACK# to signal the end of the current address tenure. 660 also asserts AACK# for one clock to signal the end of a broadcast snoop cycle. AACK# is an input to the 664 when a CPU bus target claims the current transaction by means of L2_CLAIM#. AACK# is always an input to the CPU.

The **Signal Name** column contains the name of the signal. In the Example Signal Table (Table 2-1), the AACK# signal is shown.

The **MCM** column shows which pin(s) (solder columns) of the MCM connect to the signal. The AACK# net is connected to column R08 of the MCM.

The **Nodes** columns show the internal MCM devices and pins to which the signal connects. The left column shows the device and pin number, in device.pin format. The right column shows whether the signal is an input or output of the device. An **I** indicates that the signal is an input to the device. An **O** indicates that the signal is an output from that device. A **pu** indicates that the resistor to which the signal is connected is a 10k pullup resistor.

AACK# connects to R1 pin A02, which is a 10k pullup resistor. AACK# also connects to U1 pin 28 and U2 pin 109, both of which are I/Os.

### 2.1 Pin Descriptions

The following sections contain information about the connectivity and function of the MCM pins. See the 660 User's Manual (660 UM), the 603e User's Manual (603e UM), and the SRAM, Tag, and buffer data sheets for more information.

Note that the MCM I/O pins make a connection to every net on the MCM. This was only done to aid in the manufacturing test of the MCM. Of the 1089 I/O pins that the MCM provides, only 296 signals are needed by the user to interface with the MCM. These signals are listed in Table 2-2.

Table 2-2. User Supplied Signals

Type	Signal	Type	Signal
PCI	PCI_AD0:9	MISC. FUNCTION	X_CPU_RDL_OPEN
PCI	PCI_PAR	MISC. FUNCTION	NMI_FROM_ISABRDG
PCI	PCI_AD10:31	MISC. FUNCTION	POWER_GOOD/RESET
PCI	PCI_IRDY#	MISC. FUNCTION	HRESET#
PCI	PCI_LOCK#	RW	TCK
PCI	PCI_PERR#	RW	TDI
PCI	PCI_SERR#	RW	TDO
PCI	PCI_STOP#	RW	TMS#
PCI	PCI_TRDY#	RW	CKSTP_OUT#
PCI	PCI_AD_OE#	RW_P_UP	TRST#
PCI	PCI_C/BE0:3#	P_UP	TT0:3
PCI	PCI_CLK_IN	P_UP	SHD#
PCI	PCI_DEVSEL#	P_UP	SMI#
PCI	PCI_EXT_SEL	P_UP	TAG_CS2
PCI	PCI_OL_OPEN	P_UP	TT2_664
PCI	PCI_OUT_SEL	P_UP	DBG_60X#
PCI	-664_PCI_REQ#	P_UP	L2_CLAIM#
PCI	PCI_FRAME_664#	P_UP	663_TEST#
MEMORY	MA0:11	P_UP	664_TEST#
MEMORY	MDP0:10	P_UP	TAG_MATCH
MEMORY	MD10:63	P_UP	BG_MASTER#
MEMORY	MCE0:7	P_UP	SRAM_ADSP#
MEMORY	MRE0:7	P_UP	TAG_PWRDN#
MEMORY	MWE0:1	P_UP	MWS_P2MRXS
MISC. FUNCTION	TEA#	P_UP	664_STOP_CLK_EN#
MISC. FUNCTION	DRTRY#	P_DOWN	TT4
MISC. FUNCTION	ROM_OE#	P_DOWN	GBL#
MISC. FUNCTION	ROM_WE#	P_DOWN	SRAM_CS#
MISC. FUNCTION	INT_60X#	P_DOWN	TAG_CS1#
MISC. FUNCTION	MCP_60X#	P_DOWN	OE_245_B
MISC. FUNCTION	SRAM_OE#	P_DOWN	DIR_245_A
MISC. FUNCTION	PLL_CFG0:3	P_DOWN	DIR_245_B
MISC. FUNCTION	ROM_LOAD	P_DOWN	TAG_SFUNC
MISC. FUNCTION	60X_AVDD	P_DOWN	CRS_C2PWXS
MISC. FUNCTION	QACK_60X	P_DOWN	663_MIO_TEST
MISC. FUNCTION	QREQ_60X	P_DOWN	664_MIO_TEST
MISC. FUNCTION	X_INT_60X#	CLK	XTAL1:2
MISC. FUNCTION	X_MCP_60X#	CLK	FRZ_CLK
MISC. FUNCTION	X_SRAM_OE#	CLK	FRZ_DATA
MISC. FUNCTION	INT_TO_664	CLK	PCLK_60X
MISC. FUNCTION	X_PCLK_60X	CLK	TAG_BCLK
MISC. FUNCTION	X_TAG_BCLK	CLK	CLK_EXT_FB
MISC. FUNCTION	SRESET_60X#	CLK	CLK_FB_SEL
MISC. FUNCTION	TAG_ADDR_13	CLK	CLK_PLL_EN
MISC. FUNCTION	TAG_DATA_11	CLK	SRAM_BCLK0:3
MISC. FUNCTION	IGN_PCI_AD31	CLK	CLK_COM_FRZ
MISC. FUNCTION	X_SRAM_BCLK0:3	CLK	CLK_REF_SEL
MISC. FUNCTION	X_663_CPU_CLK	CLK	CLK_TTL_CLK
MISC. FUNCTION	X_664_CPU_CLK	CLK	CLK_VCO_SEL

Type	Signal
CLK	663_CPU_CLK
CLK	664_CPU_CLK
CLK	CLK_BCLK_DIV0:1
CLK	CLK_FRZ_STROBE
CLK	CLK_MPC601_CLKS
CLK	CLK_MR/TRISTATE

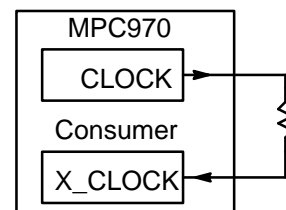
Type	Signal
CLK	664_PCI_CLK
CLK	CLK_PCI_DIV0:1
CLK	USER_PCICLK0:6
DAISEY	DAISY01:20
PWR	VSS
PWR	VDD1:3

## 2.1.1 Clock Subsystem

Table 2-3. Clock Signals

Signal Name	MCM	Nodes <sup>1</sup>	Description (See MPC970 data sheet for more information.)
663_CPU_CLK	G04	U4.38	O MPC970 2x_PCLK output.
X_663_CPU_CLK	F05	U3.157	I The CPU bus clock input of the 663.
664_CPU_CLK	G30	U4.42	O MPC970 BCLK0 output.
X_664_CPU_CLK	H29	U2.121	I The CPU bus clock input of the 664.
PCLK_60X	E18	U4.34	O MPC970 PCLK_EN output.
X_PCLK_60X	D19	U1.212	I The SYSCLK input of the CPU.
SRAM_BCLK0	W30	U4.44	O MPC970 BCLK1 output.
X_SRAM_BCLK0	Y29	U7.51	I The CPU bus clock input of SRAM bank 0.
SRAM_BCLK1	AE30	U4.46	O MPC970 BCLK2 output.
X_SRAM_BCLK1	AF29	U8.51	I The CPU bus clock input of SRAM bank 1.
SRAM_BCLK2	AG04	U4.48	O MPC970 BCLK3 output.
X_SRAM_BCLK2	AF05	U9.51	I The CPU bus clock input of SRAM bank 2.
SRAM_BCLK3	AA04	U4.50	O MPC970 BCLK4 output.
X_SRAM_BCLK3	Y05	U10.51	I The CPU bus clock input of SRAM bank 3.
TAG_BCLK	N30	U4.36	O MPC970 BCLK_EN output.
X_TAG_BCLK	P29	U5.69	I The CLK input of the TAG.
664_PCI_CLK	J30	U4.18	O MPC970 PCI_CLK1 output. Normally connected to PCI_CLK_IN. This clock is intended to be run at one half of the CPU bus frequency. See the PCI Bus section.
PCI_CLK_IN	K29	U2.123	I 660 PCI_CLK input. Normally connected to 664_PCI_CLK.
USER_PCICLK1	AE01	U4.16	O MPC970 PCI_CLK0 output.
USER_PCICLK2	N01	U4.21	O MPC970 PCI_CLK2 output.
USER_PCICLK3	C01	U4.23	O MPC970 PCI_CLK3 output.
USER_PCICLK4	A11	U4.25	O MPC970 PCI_CLK4 output.
USER_PCICLK5	A14	U4.29	O MPC970 PCI_CLK5 output.
USER_PCICLK6	E33	U4.32	O MPC970 PCI_CLK6 output.
<b>Clock Control</b> (See MPC970 data sheet for more information.)			
CLK_601_CLKS	K27	U4.40	I MPC970 input. Tie to GND for normal operation.
CLK_BCLK_DIV0	C16	U4.31	I MPC970 input. Tie to GND for normal operation.
CLK_BCLK_DIV1	B17	U4.27	I MPC970 input. Tie to GND for normal operation.
CLK_COM_FRZ	A18	U4.6	I MPC970 input. Pull high or allow to float for normal operation.
CLK_EXT_FB	A17	U4.14	I MPC970 input. Pull high or allow to float for normal operation.
CLK_FB_SEL	A15	U4.9	I MPC970 input. Pull high or allow to float for normal operation.
CLK_FRZ_STB	F33	U4.4	I MPC970 input. Pull high or allow to float for normal operation.
CLK_MR/3S	A16	U4.2	I MPC970 input. Pull high or allow to float for normal operation.

These IOs are the (nominally 66MHz) CPU bus clocks. They are intended to be used in pairs, where the CLOCK output is connected on the planar to the X\_CLOCK input. The MPC970 drives the CLOCK outputs, and the normal consumer of that clock is connected to the X\_CLOCK input. A series termination resistor and/or EMC components can be used as determined by the application.



These User PCI Clock outputs are intended for use as PCI clocks on the planar. They are nominally one half the frequency of the CPU bus clock. See the PCI Bus section.

Table 2-3. Clock Signals (Continued)

Signal Name	MCM	Nodes <sup>1</sup>		Description (See MPC970 data sheet for more information.)
CLK_PCI_DIV0	B21	U4.20	I	MPC970 input. Pull high or allow to float for normal operation.
CLK_PCI_DIV1	C20	U4.26	I	MPC970 input. Tie to GND for normal operation.
CLK_PLL_EN	D33	U4.7	I	MPC970 input. Pull high or allow to float for normal operation.
CLK_REF_SEL	B13	U4.8	I	MPC970 input. Pull high or allow to float for normal operation.
CLK_TTL_CLK	A26	U4.11	I	MPC970 input. Pull high or allow to float for normal operation.
CLK_VCO_SEL	A13	U4.52	I	MPC970 input. Tie to GND for normal operation.
FRZ_CLK	B11	U4.3	I	MPC970 input. Can be connected to the ISA clock for use, or pulled high.
FRZ_DATA	A22	U4.5	I	MPC970 input. Pull down to GND for normal operation.
XTAL1	U30	U4.12	I/O	MPC970 crystal connection.
XTAL2	T29	U4.13	I/O	MPC970 crystal connection.

1. The following lists the names for the various nodes:

U1 = PPC 603e

U2 = 664

U3 = 663

U4 = MPC970 Clock Driver

U5 = IDT Tag RAM

U6 = "245 Buffer Chip

U7 – U10 = SRAMs

## 2.1.2 CPU Bus

Table 2-4. CPU Bus Signals

Signal Name	MCM	Nodes		Description
A[0:31]	(I/O)	U1, U2, U5, U6	I/O I I	CPU address bus. Represents the physical address of the current transaction. Is valid from the bus cycle in which TS# is asserted through the bus clock in which AACK# is asserted. The CPU drives A[0:31] during transactions mastered by the CPU. The 660 drives A[0:31] while broadcasting CPU address bus snoop cycles in response to a PCI to memory transaction. The CPU and 660 snoop A[0:31] during transactions initiated by other CPU bus agents.
AACK#	R08	R1.A02 U1.28 U2.109	pu I/O I/O	CPU bus address acknowledge. The CPU bus target (660 or external CPU bus target) asserts AACK# to signal the end of the current address tenure. 660 also asserts AACK# for one clock to signal the end of a broadcast snoop cycle. AACK# is an input to the 664 when a CPU bus target claims the current transaction by means of L2_CLAIM#. AACK# is always an input to the CPU.
ABB#	R30	R1.A01 U1.36	pu I/O	CPU bus Address Bus Busy. While asserted, the address bus is busy. The 660 does not touch this signal.
AP0 AP1 AP2 AP3	AJ30 AH29 AG30 AE28	R1,U1 same same same	pu I/O	CPU bus Address Parity. One bit of odd (see DP[0:7]) parity per byte. AP0 maps to A[0:7], AP1 maps to A[8:15], AP2 maps to A[16:23], and AP3 maps to A[24:31]. The 660 does not check address parity.
APE_60X#	AA18	R1.F03 U1.218	pu O	CPU Address Parity Error output. The CPU signals address parity errors using this open drain output. The 660 does not touch this signal.
ARTRY#	P07	R1.A03 U1.32 U2.110	pu I/O I/O	CPU bus Address Retry. ARTRY# is asserted by a CPU bus device (either the target or a snooping master) to signal that the current address tenure needs to be rerun at a later time. The 660 samples ARTRY# (during broadcast snoops and transactions by CPU busmasters) on the second clock after TS# is sampled active. 660 will only assert ARTRY# on the clock after it asserts AACK# (during a PCI retry). See the 603e User's Manual for more information.
BG_664#	F11	U2.134	O	660 CPU_GNT1# – 660 asserts to grant the address bus to CPU1, the MCM CPU.



Table 2-4. CPU Bus Signals (Continued)

Signal Name	MCM	Nodes		Description
BG_60X#	F09	R1.B01 U1.27	pu I	CPU BG# – CPU address bus grant input. Normally connected to BG_664#.
BG_MASTER#	N20	U2.135	O	660 CPU_GNT2# – 660 asserts to grant the bus to a second CPU busmaster.
BR_60X#	F03	R1.A05 U1.219	pu O	CPU BR# – The CPU asserts this output to request the CPU address bus. Normally connected to BR_664#.
BR_664#	D03	U2.127	I	660 CPU_REQ1# – 660 CPU bus request input for CPU1, the MCM CPU.
BR_MASTER#	AG01	R1.H05 U2.128	pu I	660 CPU_REQ2# – 660 CPU bus request input for CPU2, a second CPU busmaster.
CI_60X#	L10	R1.J02 U1.237	pu O	CPU Cache Inhibit output. Asserted to indicate that a single-beat transaction should not be cached. See 603e UM.
CKSTP_IN#	W16	R1.D02 U1.215	pu I	CPU CheckSToP INput. When this signal is asserted, the CPU will enter checkstop.
CKSTP_OUT#	AJ02	R1.J01 U1.216	pu O	CPU CheckSToP OUTput. The CPU asserts this open drain output to indicate that it has detected a checkstop condition and has entered the checkstop state.
CSE0	M09	U1.225	O	CPU Cache Set Element 0 output. See 603e UM.
CSE1	C12	U1.150	O	CPU Cache Set Element 1 output. See 603e UM.
D[0:63]		U1, U3, SRAM	I/O I/O I/O	The 64-bit CPU data bus. D0 is the MSbit. D[0:31] connect to CPU DH[0:31]. D[32:63] connect to CPU DL[0:31]. D[0:63] also connect to the 660 CPU_DATA[0:63], and to the SRAM.
DBB#	C08	R1.C04 U1.145	pu I/O	CPU Data Bus Busy signal. The CPU that validly asserts DBB# is the current CPU busmaster. The 660 does not touch this signal.
DBDIS#	K21	R1.A04 U1.153	pu I	CPU Data Bus DISable input. See 603e UM.
DBG_60X#	G08 (I/O)	U1.26 U2.140	I O	CPU bus Data Bus Grant. 660 asserts DBG# (while ARTRY# is inactive) to signal that the requesting CPU may take ownership of the CPU data bus. DBG# is input-only on the CPU. External agents must not drive this signal.
DBWO_60X#	N16	R1.D05 U1.25	pu I	CPU Data Bus Write Only input. Asserted to allow the CPU to run the data tenure of a write out of order. See 603e UM.
DP[0:7]		U1, U3, SRAM	I/O I/O I/O	CPU Data Parity bus. DP0 maps to D[0:7], DP1 maps to D[8:15], ... DP7 maps to D[56:63]. Odd parity is used; an odd number of bits (including the parity bit) are driven high.
DPE_60X#	AC18 (I/O)	R1.F02 U1.217 U2.133	pu O I	CPU Data Parity Error. The CPU checks data parity and reports any data beat with bad parity two CPU bus clocks after the TA# for that data beat. DPE# is output-only on the CPU, and input-only on 660. 660 samples DPE# to detect L2 parity errors.
DRTRY#	AJ01	R1.H01 U1.156	pu I	CPU Data Retry input. When asserted, means that the CPU must invalidate the data from the previous read data beat. DRTRY# is also sampled at the negation of HRESET# to set the DRTRY# mode. See the application section for details.
GBL#	A12	U1.1 U2.120	I/O O	Global. This signal is asserted during snoop operations to indicate that CPU busmasters must snoop the transaction. 660 does not monitor GBL#.
HRESET#	E32	R1.D04 U1.214	pu I	CPU Hard RESET input. Initiates hardware reset operations. See 603e UM for more information.

Table 2-4. CPU Bus Signals (Continued)

Signal Name	MCM	Nodes		Description
L2_CLAIM#	AL01	U2.132	I	660 CPU_BUS_CLAIM# input. This signal is asserted by an external CPU bus target to claim a CPU bus memory transaction. It inhibits the 660 from driving AACK, TA#, TEA#, and the CPU data bus lines. L2_CLAIM# connects only to 660. L2_CLAIM# is sampled by the 660 on the second CPU_CLK after TS# is sampled active. CPU bus targets can only map to system memory space (0 to 2G) and only to memory space that is not cached by the 660 L2. The 660 L2 caches as much of the space from 0 to 2G as is populated by DRAM. So, if 8M is installed starting at 0, L2_CLAIM# can be asserted from 8M to 2G. If the internal L2 is disabled, then the entire 0 to 2G memory space can be claimed by L2_CLAIM#.
MCP_60X#	R16	R1.H02 U1.186	pu I	CPU MCP# Machine Check Pin input. The CPU enters either a machine check interrupt operation or a check stop state, depending on the state of the machine.
X_MCP_60X#	H27	U2.138	O	660 MCP# Machine Check Pin output. The 660 drives this pin to notify the CPU of a system error from a source which may not be affiliated with the current transaction. The 660 asserts MCP# in the event of a catastrophic or unrecoverable system error. The 660 asserts MCP# for two CPU bus cycles.
PLL_CFG0 PLL_CFG1 PLL_CFG2 PLL_CFG3	D25 B27 A27 C28	U1.213 U1.211 U1.210 U1.208	I I I I	CPU Phase Lock Loop Configuration Inputs. See the 603e UM for more information. Normally set to PLL_CFG[0:3] = 1100 for 99MHz CPU core and 66MHz CPU bus operation.
QACK_60X#	D17	R1.E03 U1.235	pu I	CPU Quiesce Acknowledge input. Indicates that the other CPU bus agents have ceased any bus activity that the CPU has to snoop. QACK# is sampled at the negation of HRESET# to select 32-bit mode. Drive QACK# low during HRESET# with an open collector gate to select 64-bit mode. See the applications section for details.
QREQ_60X#	C18	U1.31	O	CPU QREQ# Quiescence Request output. The CPU is requesting a low power mode. See the 603e UM.
RSRV_60X#	G16	U1.232	O	Represents the state of the Reservation bit. See 603e UM.
SHD#	D27	U2.141	I/O	660 Shared output. The function of this pin is to restore the SHD# net to a high state after it has been asserted. This pin is not used, and should be weakly pulled up.
SMI#	AL02	R1.F05 U1.187	pu I	CPU System Management Interrupt. The CPU initiates an interrupt if MSR[EE] is set (603e), else it ignores the input. Active low, level sensitive, unlatched.
SRESET#	D31	U1.189	I	CPU Soft RESET input. Async falling edge sensitive. Initiates reset exception. See 603e UM for more information.
TA#	AA16 (I/O)	R1.C05 U1.155 U2.111	pu I I/O	CPU bus Transfer Acknowledge. TA# signals that the data is valid. For every CPU clock that TA# is asserted, a data beat completes. For a single-beat cycle, TA# is only one clock.
TBEN_60X#	L16	R1.E01 U1.234	pu I	CPU Time Base Enable input. See 603e UM. Leave this input pulled high for normal operation.
TBST#	U10	R1.C03 U1.192 U2.144	pu I/O I/O	Transfer burst. TBST# indicates a burst transfer of four 64-bit double-words on the 60X CPU bus. The 664 does not assert TBST# during PCI to memory snoop cycles.
TC0 TC1	G28 L30	U1.224 U1.223	O O	CPU Transfer Code outputs. See the 603e UM.
TEA#	N10 (I/O)	R1.D01 U1.154 U2.137	pu I O	CPU bus Transfer Error Acknowledge. Assertion of TEA# terminates the current data tenure, and causes a machine check exception (or checkstop) in the CPU. The 664 asserts TEA# in the event of a catastrophic or unrecoverable system error. TEA# can be masked by setting the mask TEA# bit in the bridge control registers. If XATS# is asserted, the 660 asserts TEA# regardless of the state of MASK_TEA#.

Table 2-4. CPU Bus Signals (Continued)

Signal Name	MCM	Nodes		Description
TLBISYNC#	J16	R1.E02 U1.233	pu I	CPU TLBI sync input. See 603e UM.
TS#	L08	R1.B02 U1.149 U2.143	pu I/O I/O	CPU bus transfer start. TS# is asserted low for one CPU bus clock to signal a valid address on the address bus to start a transaction. TS# is an input to the 664 when a CPU busmaster initiates a CPU bus transaction. TS# is an output of the 664 when it initiates a snoop cycle on behalf of a PCI busmaster accessing system memory.
TSIZ0 TSIZ1 TSIZ2	L06 N06 R06	R1, U1, U2	pu O I/O	CPU bus Transfer Size in number of bytes. The TSIZ lines are valid with the CPU address lines.
TT0 TT1 TT2 TT3 TT4	A19 B19 A20 A21 A23	U1,U2 U1,U2 U1 U1,U2 U1,U2	I/O	CPU bus TT[0:4] Transfer Type. Indicates the type of transaction currently in progress. The TT lines are valid with the CPU address lines. TT2 connects only to the CPU. Normally connect TT2 to TT2_664.
TT2_664	C14	U2.153	I/O	660 TT2. Connect to TT2.
WT_60X#	D15	R1.H03 U1.236	pu O	CPU Write Through output. Asserted to indicate that a single-beat transaction is write-through.
XATS#	K07	R1.B03 U2.129	pu I	CPU bus eXtended Address Transfer Start for PIO operations, which are not supported by the 660. If XATS# is asserted, the 660 generates a TEA# error to the CPU (regardless of the setting of MASK_TEA#). The 603e CPU does not have an XATS# output. This pin is not normally connected.

### 2.1.3 PCI Bus

Table 2-5. PCI Bus Signals

Signal Name	MCM	Nodes		Description
664_PCI_REQ#	AF01	U2.58	O	660 PCI_REQ# PCI bus request. The 660 asserts PCI_REQ# to the PCI bus arbiter to request the PCI bus.
CPU_GNT_664#	AG02	U2.54	I	660 PCI_GNT# PCI bus grant input. PCI_GNT# is driven by the PCI bus arbiter in response to the 660 asserting PCI_REQ# to request the PCI bus.
PCI_AD[31:0]		U2, U3	I/O	660 PCI multiplexed Address–Data bus. A PCI bus transaction consists of an address phase followed by one or more data phases. The address tenure is defined as one PCI bus clock in duration and is coincident with the clock in which PCI_FRAME# is first asserted. After the first clock, the PCI_AD pins carry data. The PCI_AD lines are driven by the initiator during the address phase, and by the originator of the data (initiator or target) during the data phases.
PCI_C/BE[3:0]#	V01 W02 W01 Y01	U2.6 U2.5 U2.4 U2.3	I/O I/O I/O I/O	660 C (bus command) and BE (byte enable) multiplexed lines. During a PCI address phase C[3:0] carry the bus command. During a PCI data phase, PCI_C/BE[3:0]# are active low byte enables; BE0# enables AD[7:0], BE1# enables AD[8:15], BE2# enables AD[16:23], and BE3# enables AD[24:31]. The initiator drives C/BE[3:0]#. If no bus transaction is in progress, then the current PCI busmaster must drive the PCI_C/BE[3:0]# pins.
PCI_DEVSEL#	AA02	U2.204	I/O	660 PCI device select. PCI_DEVSEL# is driven by a PCI target that is claiming the current transaction. The 660 claims PCI memory transactions from 0 to 2G within the installed DRAM space.
PCI_FRAME#	AC01	U2.200	I/O	660 PCI Frame. The current PCI busmaster drives PCI_FRAME#. PCI_FRAME# signals the beginning of the address tenure on the PCI bus and the duration of the data tenure. PCI_FRAME# is deasserted to signal the final data phase of the transaction.

Table 2-5. PCI Bus Signals (Continued)

Signal Name	MCM	Nodes		Description
PCI_IRDY#	AB01	U2.201 U3.167	I/O	660 PCI initiator ready. PCI_IRDY# is driven by the current PCI busmaster. Assertion of PCI_IRDY# indicates that the PCI initiator is ready to complete this data phase.
PCI_LOCK#	AE04	U2.53	I	660 PCI lock. This signal is used to allow PCI masters to establish a resource lock of one cache line of system memory. The 660 never asserts LOCK# as a master, but it honors PCI busmaster locks of system memory.
PCI_PAR	AH01	U2.7	I/O	660 PCI parity. Even parity across AD[31:0] and the C/BE[3:0]# lines. PCI_PAR is valid one PCI bus clock after either an address or data tenure. The PCI device that drove the PCI_AD lines is responsible for driving PCI_PAR. The 660 checks and drives PCI parity.
PCI_PERR#	U01	U2.10	I/O	660 PCI parity error. PCI_PERR# is asserted by the PCI device receiving the data. PCI_PERR# is sampled on the second PCI clock after the PCI_AD lines are sampled.
PCI_SERR#	U02	U2.71	O	660 PCI system error. PCI_SERR# is asserted for one PCI clock when a catastrophic failure is detected while the 660 is a PCI target. PCI_SERR# is not monitored by the 660. The 660 asserts PCI_SERR# for certain PCI bus errors.
PCI_STOP#	AA01	U2.203	I/O	660 PCI stop. The target of the current PCI transaction can assert PCI_STOP# to indicate that the PCI target wants to end the current transaction. Data transfer can still take place if the target also asserts PCI_TRDY#, but that is the final data tenure.
PCI_TRDY#	AC02	U2.02 U3.168	I/O	660 PCI target ready. The target of the current PCI transaction drives PCI_TRDY# to indicate that the PCI target is ready. Data transfer occurs when both PCI_TRDY# and PCI_IRDY# are asserted.

## 2.1.4 DRAM

Table 2-6. DRAM Signals

Signal Name	MCM	Nodes		Description																																																						
MA[11:0]		U2	O	660 multiplexed Memory address. MA[11] is the most significant bit.																																																						
MCE[7:0]#		U2	O	660 CAS[7:0]# Column address selects. CAS[0]# selects memory byte lane 0.																																																						
MD[63:0]			I/O	<div>660 MEM_DATA[63:0] Memory data bus. MD[63:56] is always called memory byte lane 7 and is accessed using CAS[7]#. MEM_DATA[7,15,...63] are always the most significant bit in their memory byte lane.</div> <div>In BE mode, MEM_DATA[63:56] is steered to/from CPU_DATA[56:63]. In LE mode, MEM_DATA[63:56] is steered to/from CPU_DATA[0:7].</div> <table><thead><tr><th>MEM DATA</th><th>Most Signif. Bit</th><th>Mem Byte Lane</th><th>CAS#</th><th>Corresponding BE mode</th><th>CPU Data [ ] in: LE mode</th></tr></thead><tbody><tr><td>63:56</td><td>63</td><td>7</td><td>7</td><td>56:63</td><td>0:7</td></tr><tr><td>55:48</td><td>55</td><td>6</td><td>6</td><td>48:55</td><td>8:15</td></tr><tr><td>47:40</td><td>47</td><td>5</td><td>5</td><td>40:47</td><td>16:23</td></tr><tr><td>39:32</td><td>39</td><td>4</td><td>4</td><td>32:39</td><td>24:31</td></tr><tr><td>31:24</td><td>31</td><td>3</td><td>3</td><td>24:31</td><td>32:39</td></tr><tr><td>23:16</td><td>23</td><td>2</td><td>2</td><td>16:23</td><td>40:47</td></tr><tr><td>15:8</td><td>15</td><td>1</td><td>1</td><td>8:15</td><td>48:55</td></tr><tr><td>7:0</td><td>7</td><td>0</td><td>0</td><td>0:7</td><td>56:63</td></tr></tbody></table>	MEM DATA	Most Signif. Bit	Mem Byte Lane	CAS#	Corresponding BE mode	CPU Data [ ] in: LE mode	63:56	63	7	7	56:63	0:7	55:48	55	6	6	48:55	8:15	47:40	47	5	5	40:47	16:23	39:32	39	4	4	32:39	24:31	31:24	31	3	3	24:31	32:39	23:16	23	2	2	16:23	40:47	15:8	15	1	1	8:15	48:55	7:0	7	0	0	0:7	56:63
MEM DATA	Most Signif. Bit	Mem Byte Lane	CAS#	Corresponding BE mode	CPU Data [ ] in: LE mode																																																					
63:56	63	7	7	56:63	0:7																																																					
55:48	55	6	6	48:55	8:15																																																					
47:40	47	5	5	40:47	16:23																																																					
39:32	39	4	4	32:39	24:31																																																					
31:24	31	3	3	24:31	32:39																																																					
23:16	23	2	2	16:23	40:47																																																					
15:8	15	1	1	8:15	48:55																																																					
7:0	7	0	0	0:7	56:63																																																					
MDP[7:0]		U3	I/O	660 MEM_CHECK[7:0] Memory ECC/parity bus. MDP[0] is always the memory check bit for memory byte lane 0 (MD[7:0]). ECC or even parity is generated and written on memory write cycles when enabled. ECC or even parity across eight bytes is checked on memory read cycles when enabled.																																																						

Table 2-6. DRAM Signals (Continued)

Signal Name	MCM	Nodes		Description
MRE[7:0]#		U2	O	660 RAS[7:0]# Row Address Selects.
MWE0#	M33	U2.176	O	660 WE[1:0]# DRAM Write Enables. WE[1]# and WE[0]# are the same and are used to avoid the need for external buffers.
MWE1#	N33	U2.175	O	

## 2.1.5 L2

Table 2-7. L2 Signals

Signal Name	MCM	Nodes		Description
ABUF[13:28]		U6 SR	O I	Buffered CPU A[13:28], for the SRAM.
DIR_245_A	G22	U6A.1 U6A.24	I I	SRAM address buffer A Direction input. Pull low for normal operation.
DIR_245_B	L22	U6B.1 U6B.24	I I	SRAM address buffer B Direction input. Pull low for normal operation.
OE_245_B#	D29	U6A.25 U6A.48 U6B.25 U6B.48	I I I I	SRAM address buffer (A & B) Output Enable input. Pull low for normal operation.
SRAM_ADS#/ ADDR0	N22	U2.124 SR.2	O I	660 SRAM Address Strobe. Enables latching of new address for the burst SRAM.
SRAM_ADSP#	AK03	SR.1	I	SRAM ADSP# input. Pull high for normal operation.
SRAM_ALE	U22	U2.119	O	660 SRAM Address Latch Enable. This signal is always high when burst SRAMs are used. This signal is not used on the MCM.
SRAM_CNT_EN #/ ADDR1	R22	U2.125 SR.52	O I	660 SRAM CouNT ENable. Enables incrementing of burst address for the burst SRAM.
SRAM_CS#	AD29	SR.5	I	SRAM Chip Select. Pull to GND for normal operation.
SRAM_OE#	AA22	SR.50	I	SRAM Output Enable input.
X_SRAM_OE#	G26	U2.117	O	660 SRAM Output Enable output. Connect to SRAM_OE# for normal operation.
SRAM_WE#	W22	U2.114 SR.3,4	O I	660 SRAM Write Enable.
TA_GATES	N24	U5.23,2 7,28	I	TAG TT1, TAH, & TAIN# tied together and pulled high with 2 resistors. Leave pulled high for normal operation.
TAG_ADDR_13	L26	U5.33	I	TAG A13. Pull high for 256K L2. Connect to A13 for 512K L2.
TAG_A_IN	AC04	U6A	I/O	SRAM buffer spare inputs. Pull to GND or Vdd for normal operation.
TAG_CLR#	AC22	U2.116 U5.70	O I	660 Tag RAM clear. When asserted, all tags are forced to the invalid state. See the L2 Invalidate BCR.
TAG_CS1#	C33	U5.75	I	TAG CS1# Pull this pin to GND for normal operation.
TAG_CS2	H31	U5.76	I	TAG CS2. Pull this pin high for normal operation.
TAG_DATA_11	M27	U5.65	I	TAG D13. Connect to A13 for 256K L2. Pull high for 512K L2.
TAG_DTY	P25	U5.34	O	TAG DTY/S3 output. Not used by MCM.
TAG_MATCH	K31	U5.50 U2.142	O I	TAG match indication (cache hit). This signal is asserted for a cache hit. It is an active high, open drain output of the tag RAM. Pull this signal to 3.3v with a 200 ohm (nominal) resistor.
TAG_PWRDN#	M31	U5.77	I	TAG Power down mode control. Pull this pin high for normal operation.
TAG_SFUNC	C32	U5.22	I	TAG Status Bit Function Control input. Pull to GND for normal operation.
TAG_TA#	N28	U5.51	O	TAG TA# output. Not used by MCM.
TAG_VALID	AC20	U2.115 U5.80	O I	660 Tag valid bit. The 660 asserts TAG_VALID to mark the current block valid in the tag. It is negated during tag writes in response to PCI write hits, etc.
TAG_VLD	L28	U5.54	O	TAG VLD/S1 output. Not used by MCM.

Table 2-7. L2 Signals (Continued)

Signal Name	MCM	Nodes		Description
TAG_WE#	AA20	U2.114 U5.73,74	O I	660 Tag RAM write enable. Asserted to write to the Tag RAM.
TAG_WT	J28	U5.47	O	TAG WT/S3 output. Not used by MCM.
TAG_WT_DTY_IN	L24	U5.5,6 R1.G04	I pu	TAG DTY/S2 input tied to WT/S3 input and pulled high. Leave pulled high for normal operation.
TAOE#	P31	U5.23,27,28 R1, R1	I pu pu	TAG TAOE#, OE_STAT#, and OE_TAG# signals wired together & pulled up. Leave pulled up for normal operation.

## 2.1.6 System Interface

Table 2-8. System Interface Signals

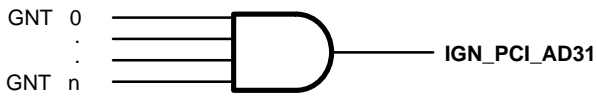
Signal Name	MCM	Nodes		Description
IGN_PCI_AD31	AD01	U2.57	I	660 Ignore PCI_AD[31] input. This signal is required when the Intel™ SIO is the PCI master, because it does not latch ISA_MASTER on posted ISA writes to 0 to 16M. IGN_PCI_AD31 is used to allow ISA busmasters to access system memory when the SIO is used as the ISA bridge. The 664 expects the memory address to appear in the range of 0 to 16M (it actually works over the entire 0-2G range) during the address phase when IGN_PCI_AD31 is asserted and then maps the access to system memory at 0 to 16M. It is usually generated by ANDing all of the active PCI bus grants (see note 1). IGN_PCI_AD31 must be valid on the PCI clock before FRAME# is sampled active.  <div style="text-align: center;">  </div>
664_STOP_CLK_EN#	B29	R1.F04 U2.151	pu I	660 input. Prepares the 660 for stopping the CPU_CLK during power management. This feature is not supported. Leave this pin pulled high for normal operation.
<b>Exceptions</b>				
INT_TO_664	B31	U2.55	I	660 INT_REQ Interrupt request input. The 660 synchronizes this signal from the interrupt controller to the CPU bus clock and passes it through to the CPU as an interrupt.
INT_60X#	U16	R1.D03 U1.188	pu I	CPU INT# Interrupt input. The CPU initiates an interrupt if MSR[EE] is set (603e), else it ignores the input. Active low, level sensitive, unlatched.
X_INT_60X#	F31	U2.139	O	660 INT_CPU# CPU interrupt output. The 660 asserts INT_CPU# to signal the CPU to run an interrupt cycle. The software is expected to eventually run a PCI interrupt acknowledge transaction to get the interrupt vector. 660 can assert INT_CPU# in response to an INT_REQ input. Normally connected to INT_60X#
NMI_FROM_ISABRDG	A31	U2.56	I	660 NMI_REQ input. Non-maskable interrupt request. When detected active (normally from the ISA bridge), an error is reported to the CPU.
POWER_GOOD/RESET#	A28	U2.156	I	660 Power-On-RESET#. While this pin is low, all latches in the 664 enter a pre-defined state. Clocking mode is re-sampled, and ROM write lockout is cleared.
<b>ROM</b>				
ROM_OE#	V31	U2.47	O	660 ROM output enable. ROM_OE# enables direct-attached ROM. This signal is always high during remote ROM operation.
ROM_WE#	T31	U2.60	O	660 ROM write enable. Write enable for flash ROM for direct-attach ROM. This signal is always high during remote ROM operation.

Table 2-8. System Interface Signals (Continued)

Signal Name	MCM	Nodes	Description
Daisy Chain			
Daisy01	E04, D05	The DAISYxx columns are arranged in pairs which are connected to each other. For example, Daisy01 consists of columns E04 and D05, which are connected to each other.	
Daisy02	E06, D07		
Daisy03	E08, D09		
Daisy04	E10, D11		
Daisy05	E12, D13		
Daisy06	E30, F29		
Daisy07	E28, F27		
Daisy08	E26, F25		
Daisy09	E24, F23		
Daisy10	E22, F21		
Daisy11	AK05, AK07		
Daisy12	AK09, AK11		
Daisy13	AK13, AK15		
Daisy14	AK17, AK19		
Daisy15	AK21, AK23		
Daisy16	AD03, AF03		
Daisy17	Y03, AB03		
Daisy18	T03, V03		
Daisy19	M03, P03		
Daisy20	H03, K03		
Power & Ground			
60X_AV <sub>DD</sub>	B02	U1.209	603e Analog VDD. Bypass as recommended in the 603e documentation.
V <sub>DD1</sub>			
V <sub>DD2</sub>			
V <sub>DD3</sub>			
V <sub>SS</sub> (GND)			

### 2.1.7 660 InterChip Communication

These signals provide communication between the 663 and the 664. They are generally not intended for use by the system. Note that CPU\_RDL\_OPEN requires a resistor or other delay component between the 663 and the 664. For more information, see Section 2 of the 660 User's Manual.

Table 2-9. InterChip Communication Signals

Signal Name	MCM	Nodes	Description
663_CPU_PAR_ERR#	G18	U2.192 U3.174	I O 660 CPU_PAR_ERR# signal. CPU Data Bus Parity Error. When asserted, this signal indicates a parity error on the CPU data bus during a write cycle.
AOS_RR_MMRS	W20	U2.69 U3.166	O I 660 All Ones Select/ROM Remote/Mask MEM_RD_SMPL signal. This signal is used to force the data bus to 64 one-bits. While ROM_LOAD is asserted, this signal is used to determine the location of the ROM. When the PCI is burst reading memory, MASK_MEM_RD_SMPL is asserted after the first MEM_RD_SMPL, and stays asserted until the PCI-to-MEM read latch is empty.

Table 2-9. InterChip Communication Signals (Continued)

Signal Name	MCM	Nodes		Description												
C2P_WRL_OPEN	AA14	U2.61 U3.154	O I	660 CPU-to-PCI Write Latch Open signal. When asserted, the CPU-to-PCI write latch accepts new data on each CPU_CLK. When deasserted, the CPU-to-PCI write latch holds its current contents.												
CPU_DATA_OE#	J14	U2.197 U3.146	O I	660 CPU Data Output Enable signal. When asserted, the 663 drives the CPU_DATA bus on the next CPU_CLK.												
CPU_RDL_OPEN	G12	U2.50	O	664 CPU Read Latch Open output. When asserted, the CPU read latch accepts new data on each CPU_CLK. When deasserted, the CPU read latch holds its current contents. This signal is asserted when data is to be sampled from memory or the PCI. When sampling data from memory, this signal is also active on the following CPU_CLK to allow ECC corrections to occur if necessary. If no ECC corrections occur, the same data is provided by the MEM read ECC correction logic.												
X_CPU_RDL_OPEN	G10	U3.148	I	663 CPU Read Latch Open input. See the 660 User's Manual for more information. Add a 200Ω (nominal) series resistor to the CPU_RDL_OPEN net between the 664 and the 663. During a CPU to memory read, if (at the 663) CPU_RDL_OPEN goes low before MEM_RD_SMPL goes low, then the 663 may provide incorrect data to the CPU. The Table shows the minimum required interval between the falling edge of MEM_RD_SMPL and the falling edge of CPU_RDL_OPEN.  <table><tr><td>Case</td><td>663 Requires</td><td>664 Supplies</td></tr><tr><td>1 Worst Case Process, Temperature, &amp; V<sub>DD</sub></td><td>&gt; 1.8ns</td><td>1.3ns</td></tr><tr><td>2 Best Case Process, Worst Case Temp. &amp; V<sub>DD</sub></td><td>&gt; 0.2ns</td><td>0.5ns</td></tr><tr><td>3 Best Case Process, Temperature, &amp; V<sub>DD</sub></td><td>&gt; 0.1ns</td><td>0.3ns</td></tr></table> The worst practical case occurs while the 664 is at Case 2 (provides .5ns difference) and the 663 is at Case 1 (requires 1.8ns difference). This requires that a minimum delay of 1.3ns be added to the CPU_RDL_OPEN signal. A delay of 2.4ns is recommended to allow a conservative margin of error. (Delay = RC = 200Ω * 12pf = 2.4ns). Note that this assumes the CPU_RDL_OPEN and MEM_RD_SMPL nets are both about three inches long and that the resistor is close to the 664. A different resistor value or an R-C combination may be required if the length or capacitance of the two nets are significantly different, or if the resistor placement differs significantly.	Case	663 Requires	664 Supplies	1 Worst Case Process, Temperature, & V <sub>DD</sub>	> 1.8ns	1.3ns	2 Best Case Process, Worst Case Temp. & V <sub>DD</sub>	> 0.2ns	0.5ns	3 Best Case Process, Temperature, & V <sub>DD</sub>	> 0.1ns	0.3ns
Case	663 Requires	664 Supplies														
1 Worst Case Process, Temperature, & V <sub>DD</sub>	> 1.8ns	1.3ns														
2 Best Case Process, Worst Case Temp. & V <sub>DD</sub>	> 0.2ns	0.5ns														
3 Best Case Process, Temperature, & V <sub>DD</sub>	> 0.1ns	0.3ns														
CRS_C2PWXS	A29	U2.65 U3.151	O I	660 CPU Read Select/CPU-to-PCI Write Crossover Select signal. When the CPU read latch is sampling data, this signal controls the CPU read multiplexer. When the CPU-to-PCI write latch is sampling data, this signal controls the CPU-to-PCI write crossover. Pull down with a 1K (nominal) resistor to select (during reset) direct-attach ROM.												
DUAL_CTRL_REF	N12	U2.205 U3.170	O I	660 Control Signal Mux Select signal. For 663 inputs that have two functions this signal selects the function. This signal is generated by dividing CPU_CLK by two. This is a useful first point to check when debugging a "dead" system.												
ECC_LE_SEL	L12	U2.2 U3.149	O I	660 ECC Select/Little-Endian Select signal. This signal indicates use of ECC or byte parity when DUAL_CTRL_REF is high and indicates use of little-endian or big-endian mode when DUAL_CTRL_REF is low.												
MEM_BE[3:0]		U2. U3	O, I	660 Memory Byte Enables signal. The eight BEs for read-modify-write memory cycles are multiplexed on MEM_BE[3:0]. When not running a read-mod-write cycle MEM_BE[3:0] should indicate all data lanes are enabled—MEM_RMW_BE[7:0]# all asserted low.												
MEM_DATA_OE#	L14	U2.196 U3.145	O I	660 Memory Data Output Enable signal. When asserted, the 663 drives the MEM_DATA bus on the next CPU_CLK.												



Table 2-9. InterChip Communication Signals (Continued)

Signal Name	MCM	Nodes		Description
MEM_ERR# (663_MEM_ERR#)	J20	U2.194 U3.171	I O	660 Memory Error signal. When asserted, indicates an uncorrectable multi-bit or parity error has occurred during a memory read. This signal is only valid on the third CPU_CLK after MEM_RD_SMPL is asserted.
MEM_RD_SMPL	J12	U2.49 U3.161	O I	660 Memory Read Sample signal. This signal is used by the ECC logic to determine when to sample ECC results. This signal is also used by the PCI read extension latch and the PCI-to-MEM read latch to load new data.
MEM_WRL_OPEN	AC14	U2.51 U3.150	O I	660 Memory Write Latch Open signal. When asserted, the MEM write latch accepts new data on each CPU_CLK. When deasserted, the MEM write latch holds its current contents.
MWS_P2MRXS	C30	U2.66 U3.152	O I	660 Memory write select/PCI-to-memory read crossover select signal. When the memory write latch is sampling data, this signal controls the memory write multiplexer. Pull up with 10K $\Omega$ (nominal) resistor to select (during reset) 603e in 3:2 core:bus clock mode.
PCI_AD_OE#	N14	U2.195 U3.144	O I	660 PCI Data Output Enable signal. While asserted, the 663 drives the PCI_AD bus. Note: This is an asynchronous input to the 663.
PCI_EXT_SEL	U14	U2.67 U3.153	O I	660 PCI Read Extension Select/PCI Write Extension Select signal. When the PCI is reading from memory, this signal controls the PCI read extension multiplexer.
PCI_OL_OPEN	W14	U2.64 U3.165	O I	660 PCI Other Latches Open signal. This signal controls the latch enables for the PCI-to-MEM read latch, the PCI read extension latch, and the PCI write extension latch.
PCI_OUT_SEL	R14	U2.68 U3.169	O I	660 PCI Output Select signal. When asserted, memory data is routed to the PCI output bus, else CPU data is routed to the PCI output bus. This signal is asynchronous.
ROM_LOAD	U20	U2.70 U3.160	O I	660 ROM Load signal. This signal is used to load data from a ROM one byte at a time until eight bytes are received, then pass the eight bytes to the CPU.
663_SBE#	G20	U2.193 U3.175	I O	660 SBE# Single-Bit Error signal. When asserted, indicates a correctable single-bit error has occurred on the memory data bus. This signal is valid only on the CPU_CLK following the assertion of MEM_RD_SMPL. If the memory is not in ECC mode, this signal is undefined.

## 2.1.8 Test Signals

Table 2-10. Test Signal Descriptions

Signal Name	MCM	Nodes		Description
<b>CPU Test Signals</b>				
LSSD_MODE#	R18	R1.E04 U1.205	pu I	CPU LSSD test input. Leave this input pulled high during normal operation.
L1_TST_CLK#	N18	R1.E05 U1.204	pu I	CPU LSSD test input. Leave this input pulled high during normal operation.
L2_TST_CLK#	L18	R1.F01 U1.203	pu I	CPU LSSD test input. Leave this input pulled high during normal operation.
60X_CLK_OUT	J18	U1.221	O	CPU system CLoCK OUTput. CLK_OUT is a clock test output.
TDO	C26	U1.198	O	CPU JTAG serial scan output.
TMS	A24	R1.J03 U1.200	pu I	CPU JTAG TAP controller mode input. Leave this input pulled high during normal operation.
TDI	A25	R1.J05 U1.199	pu I	CPU JTAG serial scan input. Leave this input pulled high during normal operation.
TCK	B25	R1.J04 U1.201	pu I	CPU JTAG scan clock input. Leave this input pulled high during normal operation.

Table 2-10. Test Signal Descriptions (Continued)

Signal Name	MCM	Nodes		Description
TRST#	C24	R1.H04 U1.202	pu I	CPU JTAG TAP controller reset. Most systems will assert this signal while either HRESET# or TRST# (from the RISCWatch connector) are asserted. See the applications section.
<b>660 Test Signals</b>				
663_MIO_TEST	A30	U3.156	I	Chip level test. Deassert low for normal operation. Do not casually assert this signal.
664_MIO_TEST	K05	U2.154	I	
663_TEST#	G14	U3.155	I	Test mode. Pull to logic high during normal operation. Do not casually assert this signal.
664_TEST#	J04	U2.155	I	

## 2.2 Net Names and MCM Nodes

### 2.2.1 Complete Pin List for MCM

Table 2-11 matches version 5.0 of the schematic. Note that, while all nets on the MCM are brought to the MCM I/O, only a subset is intended for customer use (see Table 2-2). The rest of the I/O signals can be left floating.

Table 2-11. Pin List With Net Names and MCM Nodes

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
A03	PCI_AD20	U2.24,U3.44
A04	PCI_AD21	U2.23,U3.45
A05	PCI_AD23	U2.21,U3.47
A06	PCI_AD24	U2.20,U3.48
A07	PCI_AD26	U2.18,U3.50
A08	PCI_AD27	U2.15,U3.51
A09	PCI_AD29	U2.13,U3.63
A10	PCI_AD30	U2.12,U3.64
A11	USER_PCICLK4	U4.25
A12	GBL#	U1.1,U2.120
A13	CLK_VCO_SEL	U4.52
A14	USER_PCICLK5	U4.29
A15	CLK_FB_SEL	U4.9
A16	CLK_MR/ TRISTATE	U4.2
A17	CLK_EXT_FB	U4.14
A18	CLK_COM_FRZ	U4.6
A19	TT0	U1.191,U2.150
A20	TT2	U1.185
A21	TT3	U1.184,U2.126
A22	FRZ_DATA	U4.5
A23	TT4	U1.180,U2.136
A24	TMS#	R1.J03,U1.200
A25	TDI	R1.J05,U1.199
A26	CLK_TTL_CLK	U4.11
A27	PLL_CFG2	U1.210
A28	POWER_GOOD/RE- SET#	U2.156
A29	CRS_C2PWXS	U2.65,U3.151

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
A30	663_MIO_TEST	U3.156
A31	NMI_FROM_ ISABRDG	U2.56
B02	60X_AVDD	U1.209
B03	PCI_AD18	U2.28,U3.42
B04	VSS	
B05	PCI_AD22	U2.22,U3.46
B06	VDD1	
B07	PCI_AD25	U2.19,U3.49
B08	VSS	
B09	PCI_AD28	U2.14,U3.62
B10	VDD1	
B11	FRZ_CLK	U4.3
B12	VSS	
B13	CLK_REF_SEL	U4.8
B14	VDD1	
B15	VSS	
B16	VSS	
B17	CLK_BCLK_DIV1	U4.27
B18	VSS	
B19	TT1	U1.190,U2.152
B20	VDD1	
B21	CLK_PCI_DIV0	U4.20
B22	VSS	
B23	NC	
B24	VDD1	
B25	TCK	R1.J04,U1.201
B26	VSS	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
B27	PLL_CFG1	U1.211
B28	VDD1	
B29	664_STOP_CLK_EN#	R1.F04 U2.151
B30	VSS	
B31	INT_TO_664	U2.55
B32	VDD1	
C01	USER_PCICLK3	U4.23
C02	PCI_AD17	U2.29,U3.41
C03	NC, DEPOP	
C04	PCI_AD19	U2.25,U3.43
C05	NC, DEPOP	
C06	NC	
C07	NC, DEPOP	
C08	DBB#	R1.C04,U1.145
C09	NC, DEPOP	
C10	PCI_AD31	U2.11,U3.65
C11	NC, DEPOP	
C12	603E_CSE0#	U1.150
C13	NC, DEPOP	
C14	TT2_664	U2.153
C15	NC, DEPOP	
C16	CLK_BCLK_DIV0	U4.31
C17	NC, DEPOP	
C18	QREQ_60X#	U1.31
C19	NC, DEPOP	
C20	CLK_PCI_DIV1	U4.26
C21	NC, DEPOP	
C22	VSS	
C23	NC, DEPOP	
C24	TRST#	R1.H04,U1.202
C25	NC, DEPOP	
C26	TDO	U1.198
C27	NC, DEPOP	
C28	PLL_CFG3	U1.208
C29	NC, DEPOP	
C30	MWS_P2MRXS	U2.66,U3.152
C31	NC, DEPOP	
C32	TAG_SFUNC	U5.22
C33	TAG_CS1#	U5.75
D01	PCI_AD16	U2.30,U3.40
D02	VSS	
D03	BR_664#	U2.127
D04	VDD1	
D05	DAISY01	J1.E04
D06	VSS	
D07	DAISY02	J1.E06
D08	VDD1	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
D09	DAISY03	J1.E08
D10	VSS	
D11	DAISY04	J1.E10
D12	VDD1	
D13	DAISY05	J1.E12
D14	VSS	
D15	WT_60X#	R1.H03,U1.236
D16	VDD1	
D17	QACK_60X#	R1.E03,U1.235
D18	VDD1	
D19	X_PCLK_60X	U1.212
D20	VSS	
D21	NC	
D22	VDD1	
D23	NC	
D24	VSS	
D25	PLL_CFG0	U1.213
D26	VDD1	
D27	SHD#	U2.141
D28	VSS	
D29	OE_245_B	U6A.25A,U6A.48A, U6B.25B,U6B.48B
D30	VDD1	
D31	SRESET_60X#	U1.189
D32	VSS	
D33	CLK_PLL_EN	U4.7
E01	PCI_AD14	U2.32,U3.20
E02	PCI_AD15	U2.31,U3.21
E03	NC, DEPOP	
E04	DAISY01	J1.D05
E05	NC, DEPOP	
E06	DAISY02	J1.D07
E07	NC, DEPOP	
E08	DAISY03	J1.D09
E09	NC, DEPOP	
E10	DAISY04	J1.D11
E11	NC, DEPOP	
E12	DAISY05	J1.D13
E13	NC, DEPOP	
E14	NC	
E15	NC, DEPOP	
E16	NC	
E17	NC, DEPOP	
E18	PCLK_60X	U4.34
E19	NC, DEPOP	
E20	NC	
E21	NC, DEPOP	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
E22	DAISY10	J1.F21
E23	NC, DEPOP	
E24	DAISY09	J1.F23
E25	NC, DEPOP	
E26	DAISY08	J1.F25
E27	NC, DEPOP	
E28	DAISY07	J1.F27
E29	NC, DEPOP	
E30	DAISY06	J1.F29
E31	NC, DEPOP	
E32	HRESET#	R1.D04,U1.214
E33	USER_PCICLK6	U4.32
F01	PCI_AD13	U2.33,U3.19
F02	VDD1	
F03	BR_60X#	R1.A05,U1.219
F04	VSS	
F05	X_663_CPU_CLK	U3.157
F06	VDD1	
F07	MEM_BE1	U2.207,U3.162
F08	VSS	
F09	BG_60X#	R1.B01,U1.27
F10	VDD1	
F11	BG_664#	U2.134
F12	VSS	
F13	NC	
F14	VDD1	
F15	NC	
F16	VSS	
F17	NC	
F18	VSS	
F19	NC	
F20	VDD1	
F21	DAISY10	J1.E22
F22	VSS	
F23	DAISY09	J1.E24
F24	VDD1	
F25	DAISY08	J1.E26
F26	VSS	
F27	DAISY07	J1.E28
F28	VDD1	
F29	DAISY06	J1.E30
F30	VSS	
F31	X_INT_60X#	U2.138
F32	VDD1	
F33	CLK_FRZ_STROBE	U4.4
G01	PCI_AD11	U2.35,U3.17
G02	PCI_AD12	U2.34,U3.18

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
G03	NC, DEPOP	
G04	663_CPU_CLK	U4.38
G05	NC, DEPOP	
G06	MEM_BE3	U2.1,U3.164
G07	NC, DEPOP	
G08	DBG_60X#	U1.26,U2.140
G09	NC, DEPOP	
G10	X_CPU_RDL_OPEN	U3.148
G11	NC, DEPOP	
G12	CPU_RDL_OPEN	U2.50
G13	NC, DEPOP	
G14	663_TEST#	U3.155
G15	NC, DEPOP	
G16	#RSRV_60X	U1.232
G17	NC, DEPOP	
G18	663_CPU_PAR_ERR#	U2.192,U3.174
G19	NC, DEPOP	
G20	663_SBE#	U2.193,U3.175
G21	NC, DEPOP	
G22	DIR_245_A	U6A.1A,U6A.24A
G23	NC, DEPOP	
G24	NC	
G25	NC, DEPOP	
G26	X_SRAM_OE#	U2.117
G27	NC, DEPOP	
G28	TC0	U1.224
G29	NC, DEPOP	
G30	664_CPU_CLK	U4.42
G31	NC, DEPOP	
G32	MCE6#	U2.168
G33	MCE7#	U2.165
H01	PCI_AD10	U2.3, U3.16
H02	VSS	
H03	DAISY20	J1.K03
H04	VDD1	
H05	MEM_BE0	U2.206,U3.161
H06	VSS	
H07	NC	
H08	VDD1	
H09	NC	
H10	VSS	
H11	NC	
H12	VDD1	
H13	NC	
H14	VSS	
H15	NC	
H16	VDD1	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
H17	NC	
H18	VDD1	
H19	NC	
H20	VSS	
H21	NC	
H22	VDD1	
H23	NC	
H24	VSS	
H25	NC	
H26	VDD1	
H27	X_MCP_60X#	U2.138
H28	VSS	
H29	X_664_CPU_CLK	U2.121
H30	VDD1	
H31	TAG_CS2	U5.76
H32	VSS	
H33	MCE5#	U2.169
J01	PCI_AD9	U2.37,U3.15
J02	NC	
J03	NC, DEPOP	
J04	664_TEST#	U2.155
J05	NC, DEPOP	
J06	MEM_BE2	U2.208,U3.163
J07	NC, DEPOP	
J08	NC	
J09	NC, DEPOP	
J10	NC	
J11	NC, DEPOP	
J12	MEM_RD_SMPL	U2.49,U3.147
J13	NC, DEPOP	
J14	CPU_DATA_OE#	U2.197,U3.146
J15	NC, DEPOP	
J16	TLBISYNC#	R1.E02,U1.233
J17	NC, DEPOP	
J18	60X_CLK_OUT	U1.221
J19	NC, DEPOP	
J20	663_MEM_ERR#	U2.194,U3.171
J21	NC, DEPOP	
J22	NC	
J23	NC, DEPOP	
J24	NC	
J25	NC, DEPOP	
J26	VSS	
J27	NC, DEPOP	
J28	TAG_WT	U5.47
J29	NC, DEPOP	
J30	664_PCI_CLK	U4.18

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
J31	NC, DEPOP	
J32	MCE3#	U2.171
J33	MCE4#	U2.170
K01	PCI_AD8	U2.38,U3.14
K02	VDD1	
K03	DAISY20	J1.H03
K04	VSS	
K05	664_MIO_TEST	U2.154
K06	VDD1	
K07	-XATS	R1.B03,U2.129
K08	VSS	
K09	NC	
K10	VDD1	
K11	NC	
K12	VSS	
K13	NC	
K14	VDD1	
K15	NC	
K16	VSS	
K17	NC	
K18	VSS	
K19	NC	
K20	VDD1	
K21	DBDIS#	R1.A04,U1.153
K22	VSS	
K23	NC	
K24	VDD1	
K25	NC	
K26	VSS	
K27	CLK_MPC601_CLKS	U4.40
K28	VDD1	
K29	PCI_CLK_IN	U2.123
K30	VSS	
K31	TAG_MATCH	U2.142mU5.50
K32	VDD1	
K33	MCE2#	U2.172
L01	PCI_AD6	U2.40,U3.12
L02	PCI_AD7	U2.39,U3.13
L03	NC, DEPOP	
L04	ABUF24	U7.22,U8.22,U9.22, U10.22,U6B.41B
L05	NC, DEPOP	
L06	TSIZ0	R1.K05,U1.197, U2.145
L07	NC, DEPOP	
L08	TS#	R1.B02,U1.149, U2.143
L09	NC, DEPOP	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
L10	CI_60X#	R1.J02,U1.237
L11	NC, DEPOP	
L12	ECC_LE_SEL	U2.2,U3.149
L13	NC, DEPOP	
L14	MEM_DATA_OE#	U2.196,U3.145
L15	NC, DEPOP	
L16	TBEN_60X#	R1.E01,U1.234
L17	NC, DEPOP	
L18	L2_TST_CLK#	R1.F01,U1.203
L19	NC, DEPOP	
L20	NC	
L21	NC, DEPOP	
L22	DIR_245_B	U6B.1B,U6B.24B
L23	NC, DEPOP	
L24	TAG_WT_DTY_IN	R1.G04,U5.5,U5.6
L25	NC, DEPOP	
L26	TAG_ADDR_13	U5.33
L27	NC, DEPOP	
L28	TAG_VLD	U5.54
L29	NC, DEPOP	
L30	TC1	U1.223
L31	NC, DEPOP	
L32	MCE0#	U2.174
L33	MCE1#	U2.173
M01	PCI_AD5	U2.41,U3.11
M02	VSS	
M03	DAISY19	J1.P03
M04	VDD1	
M05	ABUF23	U7.21,U8.21,U9.21, U10.21,U6B.40B
M06	VSS	
M07	NC	
M08	VDD1	
M09	CSE	U1.225
M10	VSS	
M11	NC	
M12	VDD1	
M13	NC, DEPOP	
M14	VSS	
M15	NC, DEPOP	
M16	VDD1	
M17	NC, DEPOP	
M18	VDD1	
M19	NC, DEPOP	
M20	VSS	
M21	NC, DEPOP	
M22	VDD3	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
M23	NC	
M24	VSS	
M25	NC	
M26	VDD3	
M27	TAG_DATA_11	U5.65
M28	VSS	
M29	NC	
M30	VDD3	
M31	TAG_PWRDN#	U5.77
M32	VSS	
M33	MWE0#	U2.176
N01	USER_PCICLK2	U4.21
N02	PCI_AD4	U2.42,U3.10
N03	NC, DEPOP	
N04	ABUF22	U7.7,U8.7,U9.7, U10.7,U6B.38B
N05	NC, DEPOP	
N06	TSIZ1	R1.K01,U1.196, U2.146
N07	NC, DEPOP	
N08	NC	
N09	NC, DEPOP	
N10	TEA#	R1.D01,U1.154, U2.137
N11	NC, DEPOP	
N12	DUAL_CTRL_REF	U2.205,U3.170
N13	NC, DEPOP	
N14	PCI_AD_OE#	U2.195,U3.144
N15	NC, DEPOP	
N16	DBWO_60X#	R1.D05,U1.25
N17	NC, DEPOP	
N18	L1_TST_CLK#	R1.E05,U1.204
N19	NC, DEPOP	
N20	BG_MASTER#	U2.135
N21	NC, DEPOP	
N22	SRAM_ADS/ADDR0	U2.124,U7.2,U8.2, U9.2,U10.2
N23	NC, DEPOP	
N24	TA_GATES	R1.G03,R1.G05, U5.23,U5.27,U5.28
N25	NC, DEPOP	
N26	A22	U1.160,U2.99,U5.13, U6B.11B
N27	NC, DEPOP	
N28	TAG_TA#	U5.51
N29	NC, DEPOP	
N30	TAG_BCLK	U4.36
N31	NC, DEPOP	
N32	MDP7	U3.195

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
N33	MWE1#	U2.175
P01	PCI_AD3	U2.43,U3.239
P02	VDD1	
P03	DAISY19	J1.M03
P04	VSS	
P05	ABUF21	U7.6,U8.6,U9.6, U10.6,U6B.37B
P06	VDD1	
P07	ARTRY#	R1.A03,U1.32, U2.110
P08	VSS	
P09	NC	
P10	VDD1	
P11	NC	
P12	VSS	
P13	NC, DEPOP	
P14	VDD1	
P15	NC, DEPOP	
P16	VSS	
P17	NC, DEPOP	
P18	VSS	
P19	NC, DEPOP	
P20	VDD1	
P21	NC, DEPOP	
P22	VSS	
P23	NC	
P24	VDD1	
P25	TAG_DTY	U5.34
P26	VSS	
P27	A30	U1.144,U2.107
P28	VDD1	
P29	X_TAG_BCLK	U5.69
P30	VSS	
P31	TAOE#	R1.G01,R1.G02, U5.66,U5.67,U5.68
P32	VDD1	
P33	MD63	U3.140
R01	PCI_AD1	U2.59,U3.237
R02	PCI_AD2	U2.46,U3.238
R03	NC, DEPOP	
R04	ABUF20	U7.49,U8.49,U9.49, U10.49,U6B.36B
R05	NC, DEPOP	
R06	TSIZ2	R1.K02,U1.195, U2.147
R07	NC, DEPOP	
R08	AACK#	R1.A02,U1.28, U2.109
R09	NC, DEPOP	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
R10	NC	
R11	NC, DEPOP	
R12	NC	
R13	NC, DEPOP	
R14	PCI_OUT_SEL	U2.68,U3.169
R15	VDD1	
R16	MCP_60X#	R1.H02,U1.186
R17	VDD1	
R18	LSSD_MODE#	R1.E04,U1.205
R19	VDD1	
R20	NC	
R21	NC, DEPOP	
R22	SRAM_CNT_EN/ ADDR1	U2.125,U7.52,U8.52, U9.52,U10.52
R23	NC, DEPOP	
R24	A10	U1.170,U2.84,U5.57
R25	NC, DEPOP	
R26	A13	U1.12,U2.87, U6B.23B
R27	NC, DEPOP	
R28	A27	U1.23,U2.104, U6A.9A
R29	NC, DEPOP	
R30	ABB#	R1.A01,U1.36
R31	NC, DEPOP	
R32	MD61	U3.138
R33	MD62	U3.139
T01	PCI_AD0	U2.48,U3.236
T02	VSS	
T03	DAISY18	J1.V03
T04	VDD1	
T05	ABUF19	U7.48,U8.48,U9.48, U10.48,U6B.35B
T06	VSS	
T07	NC	
T08	VDD1	
T09	NC	
T10	VSS	
T11	NC	
T12	VDD1	
T13	NC, DEPOP	
T14	VSS	
T15	NC, DEPOP	
T16	VDD1	
T17	NC, DEPOP	
T18	VDD1	
T19	NC, DEPOP	
T20	VSS	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
T21	NC, DEPOP	
T22	VDD3	
T23	A26	U1.158,U2.103,U5.7, U6B.5B
T24	VSS	
T25	A9	U1.7,U2.83,U5.56
T26	VDD3	
T27	A18	U1.165,U2.93,U5.21, U6B.16B
T28	VSS	
T29	XTAL2	U4.13
T30	VDD3	
T31	ROM_WE#	U2.60
T32	VSS	
T33	MD60	U3.133
U01	PCI_PERR#	U2.10
U02	PCI_SERR#	U2.71
U03	NC, DEPOP	
U04	ABUF18	U7.47,U8.47,U9.47, U10.47,U6B.33B
U05	NC, DEPOP	
U06	ABUF28	U7.26,U8.26,U9.26, U10.26,U6B.47B
U07	NC, DEPOP	
U08	NC	
U09	NC, DEPOP	
U10	TBST#	R1.C03,U1.192, U2.144
U11	NC, DEPOP	
U12	NC	
U13	NC, DEPOP	
U14	PCI_EXT_SEL	U2.67,U3.153
U15	VDD1	
U16	INT_60X#	R1.D03,U1.188
U17	VDD1	
U18	NC	
U19	VDD1	
U20	ROM_LOAD	U2.70,U3.160
U21	NC, DEPOP	
U22	SRAM_ALE	U2.119
U23	NC, DEPOP	
U24	A8	U1.174,U2.82,U5.55
U25	NC, DEPOP	
U26	A17	U1.15,U2.92,U5.29, U6B.17B
U27	NC, DEPOP	
U28	A25	U1.22,U2.102,U5.8, U6B.6B
U29	NC, DEPOP	
U30	XTAL1	U4.12

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
U31	NC, DEPOP	
U32	MD58	U3.131
U33	MD59	U3.132
V01	PCI_C/BE0#	U2.6
V02	VSS	
V03	DAISY18	J1.T03
V04	VDD2	
V05	ABUF17	U7.33,U8.33,U9.33, U10.33,U6B.32B
V06	VSS	
V07	ABUF27	U7.25,U8.25,U9.25, U10.25,U6A.40A
V08	VDD2	
V09	NC	
V10	VSS	
V11	DP7	U1.50,U3.137, U10.20
V12	VDD2	
V13	NC, DEPOP	
V14	VSS	
V15	NC, DEPOP	
V16	VDD2	
V17	NC, DEPOP	
V18	VDD2	
V19	NC, DEPOP	
V20	VSS	
V21	NC, DEPOP	
V22	VDD3	
V23	A31	U1.37,U2.108
V24	VSS	
V25	A7	U1.6,U2.81,U5.46
V26	VDD3	
V27	A16	U1.166,U2.91,U5.30, U6A.11A
V28	VSS	
V29	A24	U1.159,U2.101,U5.9, U6B.8B
V30	VDD3	
V31	ROM_OE#	U2.47
V32	VSS	
V33	MD57	U3.130
W01	PCI_C/BE2#	U2.4
W02	PCI_C/BE1#	U2.5
W03	NC, DEPOP	
W04	ABUF16	U7.32,U8.32,U9.32, U10.32,U6A.38A
W05	NC, DEPOP	
W06	ABUF15	U7.31,U8.31,U9.31, U10.31,U6A.37A
W07	NC, DEPOP	



Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
W08	ABUF14	U7.30,U8.30,U9.30, U10.30,U6B.27B
W09	NC, DEPOP	
W10	DP4	U1.46,U3.77,U9.46
W11	NC, DEPOP	
W12	NC	
W13	NC, DEPOP	
W14	PCI_OL_OPEN	U2.64,U3.165
W15	VDD2	
W16	CKSTP_IN#	R1.D02,U1.215
W17	VDD2	
W18	NC	
W19	VDD2	
W20	AOS_RR_MMRS	U2.69,U3.166
W21	NC, DEPOP	
W22	SRAM_WE#	U2.118,U7.3,U7.4, U8.3,U8.4,U9.3, U9.4,U10.3,U10.4
W23	NC, DEPOP	
W24	A6	U1.175,U2.80,U5.45
W25	NC, DEPOP	
W26	A15	U1.13,U2.90,U5.31, U6A.12A
W27	NC, DEPOP	
W28	A23	U1.21,U2.100,U5.12, U6B.9B
W29	NC, DEPOP	
W30	SRAM_BCLK0	U4.44
W31	NC, DEPOP	
W32	MDP6	U3.214
W33	MD56	U3.123
Y01	PCI_C/BE3#	U2.3
Y02	VDD2	
Y03	DAISY17	J1.AB03
Y04	VSS	
Y05	X_SRAM_BCLK3	U10.51
Y06	VDD2	
Y07	NC	
Y08	VSS	
Y09	DP1	U1.40,U3.219,U7.20
Y10	VDD2	
Y11	DP6	U1.48,U3.116, U10.46
Y12	VSS	
Y13	NC, DEPOP	
Y14	VDD2	
Y15	NC, DEPOP	
Y16	VSS	
Y17	NC, DEPOP	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
Y18	VSS	
Y19	NC, DEPOP	
Y20	VDD2	
Y21	NC, DEPOP	
Y22	VSS	
Y23	NC	
Y24	VDD2	
Y25	A5	U1.5,U2.77,U5.40
Y26	VSS	
Y27	A14	U1.168,U2.89,U5.32, U6B.22B
Y28	VDD2	
Y29	X_SRAM_BCLK0	U7.51
Y30	VSS	
Y31	MA11	U2.177
Y32	VDD2	
Y33	MD55	U3.121
AA01	PCI_STOP#	U2.203
AA02	PCI_DEVSEL#	U2.204
AA03	NC, DEPOP	
AA04	SRAM_BCLK3	U4.50
AA05	NC, DEPOP	
AA06	ABUF26	U7.24,U8.24,U9.24, U10.24,U6B.44B
AA07	NC, DEPOP	
AA08	NC	
AA09	NC, DEPOP	
AA10	DP3	U1.42,U3.34,U8.20
AA11	NC, DEPOP	
AA12	NC	
AA13	NC, DEPOP	
AA14	C2P_WRL_OPEN	U2.61,U3.154
AA15	NC, DEPOP	
AA16	TA#	R1.C05,U1.155, U2.111
AA17	NC, DEPOP	
AA18	APE_60X#	R1.F03,U1.218
AA19	NC, DEPOP	
AA20	TAG_WE#	U2.114,U5.73,U5.74
AA21	NC, DEPOP	
AA22	SRAM_OE#	U7.50,U8.50,U9.50, U10.50
AA23	NC, DEPOP	
AA24	A4	U1.176,U2.76,U5.39
AA25	NC, DEPOP	
AA26	NC	
AA27	NC, DEPOP	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AA28	A21	U1.17,U2.98,U5.14, U6B.12B
AA29	NC, DEPOP	
AA30	A29	U1.30,U2.106
AA31	NC, DEPOP	
AA32	MD53	U3.118
AA33	MD54	U3.119
AB01	PCI_IRDY#	U2.201,U3.167
AB02	VSS	
AB03	DAISY17	J1.Y03
AB04	VDD2	
AB05	ABUF13	U7.29,U8.29,U9.29, U10.29,U6B.26B
AB06	VSS	
AB07	D2	U1.113,U3.178, U7.38
AB08	VDD2	
AB09	DP0	U1.38,U3.196,U7.46
AB10	VSS	
AB11	DP5	U1.47,U3.95,U9.20
AB12	VDD2	
AB13	NC, DEPOP	
AB14	VSS	
AB15	NC, DEPOP	
AB16	VDD2	
AB17	NC, DEPOP	
AB18	VDD2	
AB19	NC, DEPOP	
AB20	VSS	
AB21	NC, DEPOP	
AB22	VDD2	
AB23	NC	
AB24	VSS	
AB25	A3	U1.3,U2.75,U5.37
AB26	VDD2	
AB27	A12	U1.169,U2.86,U5.63
AB28	VSS	
AB29	A20	U1.164,U2.97,U5.15, U6B.13B
AB30	VDD2	
AB31	MA10	U2.178
AB32	VSS	
AB33	MD52	U3.113
AC01	PCI_FRAME_664#	U2.200
AC02	PCI_TRDY#	U2.202,U3.168
AC03	NC, DEPOP	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AC04	TAG_A_IN	U6A.2A,U6A.3A, U6A.5A,U6A.6A, U6A.8A,U6A.13A, U6A.14A,U6A.16A, U6A.17A,U6A.19A, U6A.20A,U6A.22A, U6A.23A
AC05	NC, DEPOP	
AC06	ABUF25	U7.23,U8.23,U9.23, U10.23,U6B.43B
AC07	NC, DEPOP	
AC08	D3	U1.110,U3.179, U7.39
AC09	NC, DEPOP	
AC10	DP2	U1.41,U3.3,U8.46
AC11	NC, DEPOP	
AC12	NC	
AC13	NC, DEPOP	
AC14	MEM_WRL_OPEN	U2.51,U3.150
AC15	NC, DEPOP	
AC16	NC	
AC17	NC, DEPOP	
AC18	DPE_60X#	R1.F02,U1.217, U2.133
AC19	NC, DEPOP	
AC20	TAG_VALID	U2.115,U5.80
AC21	NC, DEPOP	
AC22	TAG_CLEAR#	U2.116,U5.70
AC23	NC, DEPOP	
AC24	A2	U1.178,U2.74,U5.35
AC25	NC, DEPOP	
AC26	A11	U1.11,U2.85,U5.61
AC27	NC, DEPOP	
AC28	A19	U1.16,U2.96,U5.16, U6B.14B
AC29	NC, DEPOP	
AC30	A28	U1.151,U2.105, U6B.2B
AC31	NC, DEPOP	
AC32	MD50	U3.111
AC33	MD51	U3.112
AD01	IGN_PCI_AD31	U2.57
AD02	VDD2	
AD03	DAISY16	J1.AF03
AD04	VSS	
AD05	D5	U1.108,U3.186, U7.41
AD06	VDD2	
AD07	D11	U1.92,U3.207,U7.13
AD08	VSS	
AD09	D17	U1.84,U3.221,U8.35

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AD10	VDD2	
AD11	D23	U1.76,U3.2,U8.45
AD12	VSS	
AD13	D29	U1.68,U3.31,U8.15
AD14	VDD2	
AD15	D35	U1.139,U3.57,U9.39
AD16	VSS	
AD17	D41	U1.129,U3.79,U9.9
AD18	VSS	
AD19	D47	U1.118,U3.94,U9.19
AD20	VDD2	
AD21	D53	U1.101,U3.107, U10.41
AD22	VSS	
AD23	D59	U1.57,U3.126, U10.13
AD24	VDD2	
AD25	A1	U1.2 ,U2.73
AD26	VSS	
AD27	NC	
AD28	VDD2	
AD29	SRAM_CS#	U7.5,U8.5,U9.5, U10.5
AD30	VSS	
AD31	MA9	U2.179
AD32	VDD2	
AD33	MD49	U3.109
AE01	USER_PCICLK1	U4.16
AE02	PCI_LOCK#	U2.53
AE03	NC, DEPOP	
AE04	D4	U1.109,U3.185, U7.40
AE05	NC, DEPOP	
AE06	D10	U1.93,U3.199,U7.12
AE07	NC, DEPOP	
AE08	D16	U1.85,U3.220,U8.34
AE09	NC, DEPOP	
AE10	D22	U1.78,U3.1,U8.44
AE11	NC, DEPOP	
AE12	D28	U1.71,U3.27,U8.14
AE13	NC, DEPOP	
AE14	D34	U1.140,U3.56,U9.38
AE15	NC, DEPOP	
AE16	D40	U1.130,U3.78,U9.8
AE17	NC, DEPOP	
AE18	D46	U1.119,U3.89,U9.18
AE19	NC, DEPOP	
AE20	D52	U1.102,U3.106, U10.40

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AE21	NC, DEPOP	
AE22	D58	U1.56,U3.125, U10.12
AE23	NC, DEPOP	
AE24	A0	U1.179,U2.72
AE25	NC, DEPOP	
AE26	NC	
AE27	NC, DEPOP	
AE28	AP3	R1.B04,U1.226
AE29	NC, DEPOP	
AE30	SRAM_BCLK1	U4.46
AE31	NC, DEPOP	
AE32	MDP5	U3.234
AE33	MD48	U3.108
AF01	664_PCI_REQ#	U2.58
AF02	VSS	
AF03	DAISY16	J1.AD03
AF04	VDD2	
AF05	X_SRAM_BCLK2	U9.51
AF06	VSS	
AF07	D9	U1.94,U3.198,U7.9
AF08	VDD2	
AF09	D15	U1.87,U3.218,U7.19
AF10	VSS	
AF11	D21	U1.80,U3.229,U8.41
AF12	VDD2	
AF13	D27	U1.72,U3.26,U8.13
AF14	VSS	
AF15	D33	U1.141,U3.55,U9.35
AF16	VDD2	
AF17	D39	U1.131,U3.70 ,U9.45
AF18	VDD2	
AF19	D45	U1.123,U3.88 ,U9.15
AF20	VSS	
AF21	D51	U1.105,U3.105, U10.39
AF22	VDD2	
AF23	D57	U1.55,U3.124,U10.9
AF24	VSS	
AF25	D63	U1.64,U3.136, U10.19
AF26	VDD2	
AF27	NC	
AF28	VSS	
AF29	X_SRAM_BCLK1	U8.51
AF30	VDD2	
AF31	MA8	U2.180
AF32	VSS	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AF33	MD47	U3.102
AG01	BR_MASTER#	R1.H05,U2.128
AG02	CPU_GNT_664#	U2.54
AG03	NC, DEPOP	
AG04	SRAM_BCLK2	U4.48
AG05	NC, DEPOP	
AG06	D8	U1.97,U3.197,U7.8
AG07	NC, DEPOP	
AG08	D14	U1.89,U3.210,U7.18
AG09	NC, DEPOP	
AG10	D20	U1.81,U3.228,U8.40
AG11	NC, DEPOP	
AG12	D26	U1.73,U3.25,U8.12
AG13	NC, DEPOP	
AG14	D32	U1.143,U3.54,U9.34
AG15	NC, DEPOP	
AG16	D38	U1.133,U3.69,U9.44
AG17	NC, DEPOP	
AG18	D44	U1.124,U3.87,U9.14
AG19	NC, DEPOP	
AG20	D50	U1.106,U3.104, U10.38
AG21	NC, DEPOP	
AG22	D56	U1.52,U3.117,U10.8
AG23	NC, DEPOP	
AG24	D62	U1.63,U3.135, U10.18
AG25	NC, DEPOP	
AG26	NC	
AG27	NC, DEPOP	
AG28	NC	
AG29	NC, DEPOP	
AG30	AP2	R1.B05,U1.227
AG31	NC, DEPOP	
AG32	MD45	U3.100
AG33	MD46	U3.101
AH01	PCI_PAR	U2.7
AH02	VDD2	
AH03	NC	
AH04	VSS	
AH05	D1	U1.114,U3.177, U7.35
AH06	VDD2	
AH07	D7	U1.98,U3.188,U7.45
AH08	VSS	
AH09	D13	U1.90,U3.209,U7.15
AH10	VDD2	
AH11	D19	U1.82,U3.227, U8.39

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AH12	VSS	
AH13	D25	U1.74,U3.24,U8.9
AH14	VDD2	
AH15	D31	U1.66,U3.33,U8.19
AH16	VSS	
AH17	D37	U1.134,U3.68, U9.41
AH18	VSS	
AH19	D43	U1.125,U3.86,U9.13
AH20	VDD2	
AH21	D49	U1.107,U3.99,U10.35
AH22	VSS	
AH23	D55	U1.51,U3.115,U10.45
AH24	VDD2	
AH25	D61	U1.62,U3.134,U10.15
AH26	VSS	
AH27	NC	
AH28	VDD2	
AH29	AP1	R1.C01, U1.230
AH30	VSS	
AH31	MA7	U2.181
AH32	VDD2	
AH33	MD44	U3.93
AJ01	DRTRY#	R1.H01, U1.156
AJ02	CKSTP_OUT#	R1.J01, U1.216
AJ03	NC, DEPOP	
AJ04	D0	U1.115,U3.176,U7.34
AJ05	NC, DEPOP	
AJ06	D6	U1.99,U3.187,U7.44
AJ07	NC, DEPOP	
AJ08	D12	U1.91,U3.208,U7.14
AJ09	NC, DEPOP	
AJ10	D18	U1.83,U3.226,U8.38
AJ11	NC, DEPOP	
AJ12	D24	U1.75,U3.4,U8.8
AJ13	NC, DEPOP	
AJ14	D30	U1.67,U3.32,U8.18
AJ15	NC, DEPOP	
AJ16	D36	U1.135,U3.67,U9.40
AJ17	NC, DEPOP	
AJ18	D42	U1.126,U3.80,U9.12
AJ19	NC, DEPOP	
AJ20	D48	U1.117,U3.98,U10.34
AJ21	NC, DEPOP	
AJ22	D54	U1.100,U3.114,U10.44
AJ23	NC, DEPOP	
AJ24	D60	U1.58,U3.127,U10.14

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AJ25	NC, DEPOP	
AJ26	NC	
AJ27	NC, DEPOP	
AJ28	NC	
AJ29	NC, DEPOP	
AJ30	AP0	R1.C02,U1.231
AJ31	NC, DEPOP	
AJ32	MD42	U3.91
AJ33	MD43	U3.92
AK01	NC	
AK02	VSS	
AK03	SRAM_ADSP#	U7.1,U8.1,U9.1,U10.1
AK04	VDD2	
AK05	DAISY11	J1.AK07
AK06	VSS	
AK07	DAISY11	J1.AK05
AK08	VDD2	
AK09	DAISY12	J1.AK11
AK10	VSS	
AK11	DAISY12	J1.AK09
AK12	VDD2	
AK13	DAISY13	J1.AK15
AK14	VSS	
AK15	DAISY13	J1.AK13
AK16	VDD2	
AK17	DAISY14	J1.AK19
AK18	VDD2	
AK19	DAISY14	J1.AK17
AK20	VSS	
AK21	DAISY15	J1.AK23
AK22	VDD2	
AK23	DAISY15	J1.AK21
AK24	VSS	
AK25	NC	
AK26	VDD2	
AK27	NC	
AK28	VSS	
AK29	NC	
AK30	VDD2	
AK31	MA6	U2.184
AK32	VSS	
AK33	MD41	U3.90
AL01	L2_CLAIM#	U2.132
AL02	SMI#	R1.F05,U1.187
AL03	NC, DEPOP	
AL04	MRE0#	U2.164
AL05	NC, DEPOP	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AL06	MRE1#	U2.163
AL07	NC, DEPOP	
AL08	MRE2#	U2.162
AL09	NC, DEPOP	
AL10	MRE3#	U2.161
AL11	NC, DEPOP	
AL12	MRE4#	U2.160
AL13	NC, DEPOP	
AL14	MRE5#	U2.159
AL15	NC, DEPOP	
AL16	MRE6#	U2.158
AL17	NC, DEPOP	
AL18	MRE7#	U2.157
AL19	NC, DEPOP	
AL20	MA0	U2.190
AL21	NC, DEPOP	
AL22	MA1	U2.189
AL23	NC, DEPOP	
AL24	MA2	U2.188
AL25	NC, DEPOP	
AL26	MA3	U2.187
AL27	NC, DEPOP	
AL28	MA4	U2.186
AL29	NC, DEPOP	
AL30	MA5	U2.185
AL31	NC, DEPOP	
AL32	MDP4	U3.37
AL33	MD40	U3.83
AM02	VDD2	
AM03	MD1	U3.182
AM04	VSS	
AM05	MD4	U3.189
AM06	VDD2	
AM07	MD7	U3.194
AM08	VSS	
AM09	MD9	U3.201
AM10	VDD2	
AM11	MD12	U3.206
AM12	VSS	
AM13	MD15	U3.213
AM14	VDD2	
AM15	MD17	U3.222
AM16	VSS	
AM17	MD20	U3.225
AM18	VSS	
AM19	MD23	U3.233
AM20	VDD2	

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AM21	MD25	U3.6
AM22	VSS	
AM23	MD28	U3.29
AM24	VDD2	
AM25	MD31	U3.36
AM26	VSS	
AM27	MD33	U3.59
AM28	VDD2	
AM29	MD36	U3.74
AM30	VSS	
AM31	MD39	U3.81
AM32	VDD2	
AN03	MD0	U3.180
AN04	MD2	U3.183
AN05	MD3	U3.184
AN06	MD5	U3.190
AN07	MD6	U3.193
AN08	MDP0	U3.141
AN09	MD8	U3.200
AN10	MD10	U3.202
AN11	MD11	U3.203

Pin <sup>1</sup>	Net Name <sup>2</sup>	MCM Nodes <sup>3</sup>
AN12	MD13	U3.211
AN13	MD14	U3.212
AN14	MDP1	U3.122
AN15	MD16	U3.215
AN16	MD18	U3.223
AN17	MD19	U3.224
AN18	MD21	U3.231
AN19	MD22	U3.232
AN20	MDP2	U3.103
AN21	MD24	U3.5
AN22	MD26	U3.7
AN23	MD27	U3.28
AN24	MD29	U3.30
AN25	MD30	U3.35
AN26	MDP3	U3.82
AN27	MD32	U3.58
AN28	MD34	U3.60
AN29	MD35	U3.73
AN30	MD37	U3.75
AN31	MD38	U3.76

**Notes:**

1. 1077 pins are listed.
2. The net name is the net name from the MCM schematics (see Section 13). Use it when referring to a net. Many pin sites are not populated with a solder column. These sites are indicated with DEPOP in the Net Name column. NC in the Net Name column means that the MCM makes no connection to that site.
3. The MCM Nodes indicate the pins of the devices that the MCM pin is connected to. For example, U2.24 represents pin 24 of device U2.

## **Section 3**

### **CPU and Level Two Cache**

This section discusses topics that are directly related to the CPU, CPU bus, and Level Two (L2) cache, including how the 660 bridge decodes CPU initiated transfers as a function of the transfer type and address range. For more information, refer to the *660 User's Manual*.

The 100 MHz PPC 603e MCM supports CPU bus speeds up to 66MHz, and PCI bus speeds up to 33MHz. The MCM is initially configured with a CPU\_core:CPU\_bus:PCI bus frequency ratio of 99:66:33 MHz. This accomplished by setting the configuration bits for the MPC970 clock driver via off MCM pullups.

#### **3.1 CPU Busmasters**

The MCM uses a single PowerPC 603e CPU, and an internal L2 cache; thus, there are only two busmasters on the CPU bus: the CPU and the 660. CPU bus arbitration is greatly simplified, and the multi-processor capabilities of the 660 are not used. The remaining arbitration on the CPU bus is between the CPU and the snoop broadcasting logic in the 660. Since the 660 parks the CPU bus on CPU1 (the 603e) whenever the bus is idle, CPU latency is minimized.

One level of address bus pipelining is supported, and most data writes are posted. Precise exceptions are reported via TEA#, and imprecise exceptions are reported via MCP#. PIO, or programmed I/O transactions (XATS# type) are not supported.

The MCM is not currently specified for use with external CPU busmasters. For more information on the CPU bus protocol, see the 603e User's Manual. For more information on the multiprocessor capabilities of the 660, see the *660 User's Manual*. For MCM applications involving external CPU bus agents, please contact IBM PowerPC Embedded Processor Systems Application Engineering.

##### **3.1.1 603e CPU**

The MCM operates the 603e in 64-bit data bus mode. The 660 will not operate in 32-bit mode.

The MCM is configured for DRTRY# mode, and should not be configured for no-DRTRY# mode. In DRTRY# mode, data is assumed to have been speculatively presented to the CPU, and so is held for one clock in the 603e BIU before being presented to the CPU core data consumers.

The MCM runs at 3:2 603e internal clock to bus clock ratio at 99MHz:66MHz. CPU PLL\_CFG[0:3] is set to 1100. The 603e may be run at 1:1 core:bus ratio, and the frequency of the CPU bus clock can be varied. See Section 7, Clocks, for more information.

### 3.2 System Response by CPU Bus Transfer Type

All access to the rest of the system is provided to the CPU by the 660. Table 3-1 shows the 660 decoding of CPU bus transfer types. Based on TT[0:3], the 660 responds to CPU bus-master cycles by generating a read transaction, a write transaction, or an address-only response. The 660 ignores TT[4] when it evaluates the transfer type.

The bridge decodes the target of the transaction based on the address range of the transfer as shown in Table 3-2. The transfer type decoding shown in Table 3-1 combines with the target decoding to produce one of the following operations:

- System memory reads and writes
- PCI I/O reads and writes
- PCI configuration reads and writes
- PCI interrupt acknowledge reads
- PCI memory reads and writes
- System ROM reads and writes
- Various bridge control register (BCR) reads and writes.



Table 3-1. TT[0:3] (Transfer Type) Decoding by 660

TT[0:3]	60X Operation	60X Bus Transaction	660 Operation For CPU to Memory Transfers	660 Operation For CPU to PCI Transactions
0000	Clean block or lwarx	Address only	Asserts AACK#. No other response. No PCI transaction.	
0001	Write with flush	SBW(1) or burst	Memory write operation.	PCI write transaction.
0010	Flush block or stwcx	Address only	Asserts AACK#. No other response. No PCI transaction.	
0011	Write with kill	SBW or burst	Memory write operation. L2 invalidates addressed block.	PCI write transaction.
0100	sync or tlbsync	Address only	Asserts AACK#. No other response. No PCI transaction.	
0101	Read or read with no intent to cache	SBR(1) or burst	Memory read operation.	PCI read transaction.
0110	Kill block or icbi	Address only	Asserts AACK#. L2 invalidates addressed block.	Asserts AACK#. No other response.
0111	Read with intent to modify	Burst	Memory read operation.	PCI read transaction.
1000	eieio	Address only	Asserts AACK#. No other response. No PCI transaction.	
1001	Write with flush atomic, stwcx	SBW	Memory write operation.	PCI write transaction.
1010	ecowx	SBW	Asserts AACK# and TA# if the transaction is not claimed by another 60X bus device. No PCI transaction. No other response.	
1011	Reserved		Asserts AACK#. No other response. No PCI transaction.	
1100	TLB invalidate	Address only	Asserts AACK#. No other response. No PCI transaction.	
1101	Read atomic, lwarx	SBR or burst	Memory read operation.	PCI read transaction.
1110	External control in, eciwx	Address only	660 asserts all ones on the CPU data bus. Asserts AACK# and TA# if the transaction is not claimed by another 60X bus device. No PCI transaction. No other response.	
1111	Read with intent to modify atomic, stwcx	Burst	Memory read operation.	PCI read transaction.

**Note:**

1. SBR means Single-Beat Read, and SBW means Single-Beat Write

Transfer types in Table 3-1 that have the same response are handled identically by the bridge. For example, if the address is the same, the bridge generates the same memory read transaction for transfer types 0101, 0111, 1101, and 1111.

The 660 does not generate PCI or system memory transactions in response to address only transfers. The bridge does drive all-ones onto the CPU bus and signals TA# during an eciwx if no other CPU bus agent claims the transfer.

References in the remainder of this document to a CPU read, assume one of the transfer types in Table 3-1 that produce the read response from the 660. Likewise, references to a CPU write refer to those transfer type that produce the write response.

### 3.3 System Response by CPU Bus Address Range

The 660 determines the target of a CPU busmaster transaction based on the CPU bus address range as shown in Table 3-2. The acronym BCR means bridge control register.

**Table 3-2. 660 Address Mapping of CPU Bus Transactions**

CPU Bus Address	Other Conditions	Target Transaction	Target Bus Address	Notes
0 to 2G 0000 0000h to 7FFF FFFFh		System Memory	0 to 2G 0000 0000h to 7FFF FFFFh	(1)(2)
2G to 2G + 8M 8000 0000h to 807F FFFFh	Contiguous Mode	PCI I/O Transaction, BCR Transaction, or PCI Configuration (Type 1) Transaction	0 to 8M 0000 0000h to 007F FFFFh	(3)
	Non-Contiguous Mode		0 to 64K 0000 0000h to 0000 FFFFh	(4)
2G + 8M to 2G + 16M 8080 0000h to 80FF FFFFh		PCI Configuration (Type 0) Transaction	PCI Configuration Space 0080 0000h to 00FF FFFFh	
2G + 16M to 3G – 8M 8100 0000h to BF7F FFFFh		PCI I/O Transaction	16M to 1G – 8M 0100 0000h to 3F7F FFFFh	
3G – 8M to 3G BF80 0000h to BFFF FFFFh		BCR Transactions and PCI Interrupt Ack. Transactions	1G – 8M to 1G 3F80 0000h – 3FFF FFFFh	(3)(6)
3G to 4G – 2M C000 0000h to FFDF FFFFh		PCI Memory Transaction	0 to 1G – 2M 0000 0000h to 3FDF FFFFh	
4G – 2M to 4G FFE0 0000h to FFFF FFFFh	Direct Attach ROM Read, Write, or Write Lockout	BCR Transaction	0 to 2M 0000 0000h to 001F FFFFh (ROM Address Space)	(2)
	Remote ROM	PCI Memory Transaction to I/O Bus Bridge	1G – 2M to 1G 3FE0 0000h to 3FFF FFFFh	(2)

**Notes:**

1. System memory can be cached. Addresses from 2G to 4G are not cacheable.
2. Memory does not occupy the entire address space.
3. Registers do not occupy the entire address space.
4. Each 4K page in the 8M CPU bus address range maps to 32 bytes in PCI I/O space.
5. Registers and memory do not occupy the entire address space. Accesses to unoccupied addresses result in all one-bits on reads and no-ops on writes.
6. A memory read of BFFF FFF0h generates an interrupt acknowledge transaction on the PCI bus.

#### 3.3.1 Address Mapping for Non-Contiguous I/O

Figure 3-1 shows the address mapping that the 660 performs in non-contiguous mode. The I/O map type register (address 8000 0850h) and the bridge chip set options 1 register (index BAh) control the selection of contiguous and non-contiguous I/O. In non-contiguous



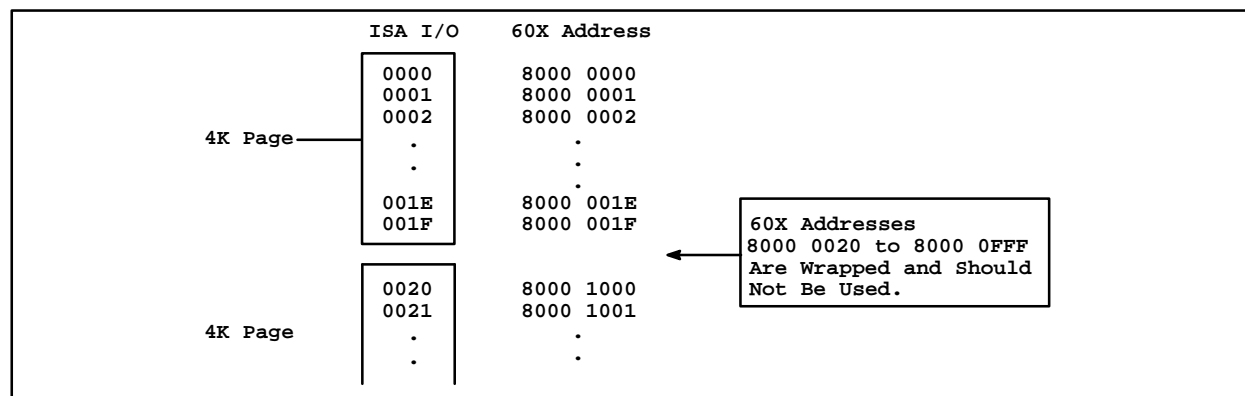


Figure 3-2. Non-Contiguous PCI I/O Address Translation

### 3.3.2 Address Mapping for Contiguous I/O

In contiguous I/O mode, CPU addresses from 2G to 2G + 8M generate a PCI I/O cycle on the PCI bus with PCI\_AD[29:00] unchanged. The low 64K of PCI I/O addresses are forwarded to the ISA bus unless claimed by a PCI agent.

Memory page protection attributes may only be assigned by 4K groups of ports, rather than by 32-port groups as in the non-contiguous mode. This is the power-on default mode. Figure 3-3 gives an example of contiguous I/O partitioning.

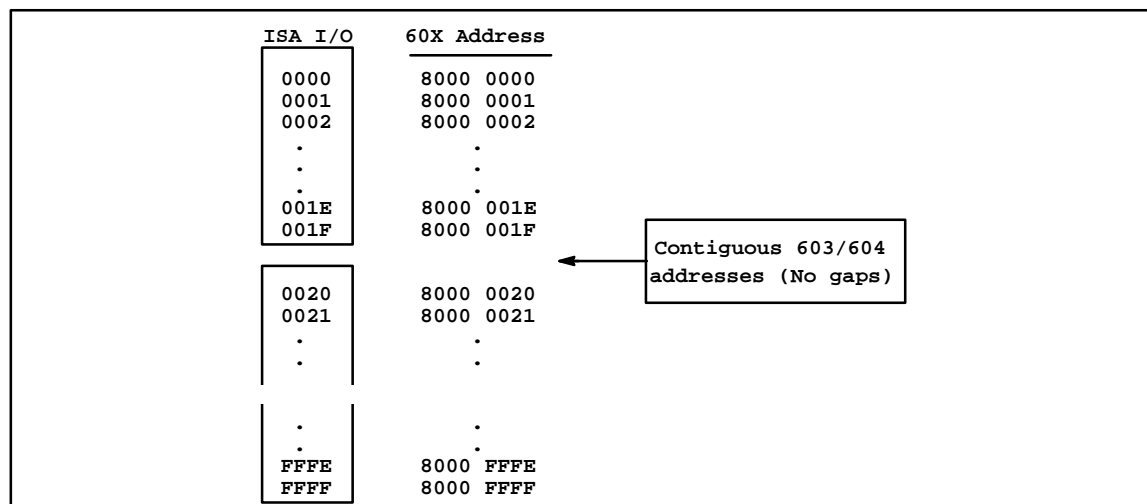


Figure 3-3. Contiguous PCI I/O Address Translation

### 3.3.3 PCI Final Address Formation

The 660 maps 60X CPU bus addresses from 2G to 4G as PCI transactions, error address register reads, or ROM reads and writes. The 660 manipulates 60X bus addresses from 2G to 4G to generate PCI addresses as follows:

- PCI\_AD[31:30] are set to zero.
- PCI\_AD[2:0] are unmunged if little-endian mode is selected.
- After unmunging, PCI\_AD[1:0] are set to 00b except for PCI I/O cycles.

### **3.4 CPU to Memory Transfers**

The system memory address space is from 0 to 2G. Physical memory does not occupy the entire address space. When the CPU reads an unpopulated location, the 660 returns all-ones and completes the transfer normally. When the CPU writes to an unpopulated location, the Bridge signals normal transfer completion to the CPU but does not write the data to memory. The memory select error bit in the error status 1 register (bit 5 in index C1h) is set in both cases.

All CPU to memory writes are posted and can be pipelined.

The 660 supports all CPU to memory bursts, and all single-beat transfer sizes and alignments that do not cross an 8-byte boundary, which includes all memory transfers initiated by the 603/604 CPU.

#### **3.4.1 LE Mode**

The bridge supports all transfer sizes and alignments that the CPU can create in LE mode; however, all loads or stores must be at natural alignments in LE mode (or the CPU will take an alignment exception). Also, load/store multiple word and load/store string word instructions are not supported in the CPU in LE mode.

### **3.5 CPU to PCI Transactions**

Since all CPU to PCI transactions are CPU memory mapped, software must in general utilize the EIEIO instruction which enforces in-order execution, particularly on PCI I/O and configuration transactions. Some PCI memory operations can be sensitive to order of access also. See the 660 User's Manual.

All addresses from 2G to 4G (including ROM space) must be marked non-cacheable. See the PowerPC Reference Platform Specification. The MCM supports all PCI bus protocols during CPU to PCI transactions.

The MCM supports all CPU to PCI transfer sizes that do not cross a 4-byte boundary, and it supports 8-byte CPU to PCI writes that are aligned on an 8-byte boundary. The bridge does not support CPU bursts to the PCI bus.

When the 660 decodes a CPU access as targeted for the PCI, the 660 requests the PCI bus. Once the SIO grants the PCI bus to the 660, the bridge initiates the PCI cycle and releases the bus.

CPU to PCI transactions that the PCI target retries, cause the 660 to deassert its PCI\_REQ# (the Bridge follows the PCI retry protocol). The Bridge stays off of the PCI bus for two PCI clocks before reasserting PCI\_REQ# (or FRAME#, if the PCI bus is idle and the PCI\_GNT# to the Bridge is active).

#### **3.5.1 CPU to PCI Read**

If the CPU to PCI cycle is a read, a PCI read cycle is run. If the PCI read cycle completes, the data is passed to the CPU and the CPU cycle is ended. If the PCI cycle is retried, the CPU cycle is retried. If a PCI master access to system memory is detected before the PCI read cycle is run then the CPU cycle is retried (and no PCI cycle is generated).

### 3.5.2 CPU to PCI Write

If the CPU to PCI cycle is a write, a PCI write cycle is run. CPU to PCI I/O writes are not posted, as per the *PCI Local Bus Specification* version 2.1. If the PCI transaction is retried, the Bridge retries the CPU.

CPU to PCI memory writes are posted, so the CPU write cycle is ended as soon as the data is latched. If the PCI cycle is retried, the Bridge retries the cycle until it completes.

#### 3.5.2.1 Eight-Byte Writes to the PCI (Memory and I/O)

The 660 supports 1-byte, 2-byte, 3-byte, and 4-byte transfers to and from the PCI. The 660 also supports 8-byte memory and I/O writes (writes only, not reads) to the PCI bus. When an 8-byte write to the PCI is detected, it is not posted initially. Instead the CPU waits until the first 4-byte write occurs, then the second 4-byte write is posted. If the PCI retries on the first 4-byte transfer or a PCI master access to system memory is detected before the first 4-byte transfer, then the CPU is retried. If the PCI retries on the second 4-byte transfer, then the 660 retries the PCI write.

### 3.5.3 CPU to PCI Memory Transactions

CPU transfers from 3G to 4G–2M are mapped to the PCI bus as memory transactions.

### 3.5.4 CPU to PCI I/O Transactions

CPU transfers from 2G+16M to 3G–8M are mapped to the PCI bus as I/O transactions. In compliance with the PCI specification, the 660 master aborts all I/O transactions that are not claimed by a PCI agent.

### 3.5.5 CPU to PCI Configuration Transactions

The MCM allows the CPU to generate type 0 and type 1 PCI configuration cycles. The CPU initiates a transfer to the appropriate address. The 660 decodes the cycle and generates a request to the PCI arbiter in the SIO. When the PCI bus is acquired, the 660 enables its PCI\_AD drivers and drives the address onto the PCI\_AD lines for one PCI clock before it asserts PCI\_FRAME#. Predriving the PCI\_AD lines for one clock before asserting PCI\_FRAME# allows the IDSELs to be resistively connected to the PCI\_AD[31:0] bus at the system level.

The transfer size must match the capabilities of the target PCI device for configuration cycles. The MCM supports 1-, 2-, 3-, and 4-byte transfers that do not cross a 4-byte boundary.

Address unmunging and data byte swapping follow the same rules as those for system memory with respect to BE and LE modes of operation. Address unmunging has no effect on the CPU address lines which correspond to the IDSEL inputs of the PCI devices.

The generation of PCI configuration cycles is via the 660 indexed Bridge Control Registers (BCR). This configuration method is described in section 4 of the 660 User's Manual. The IDSEL assignments and their respective PCI\_AD lines are shown in Table 3-3. The addresses used for configuration are assigned as shown in Table 3-3.

Table 3-3. CPU to PCI Configuration Mapping

Device	IDSEL Line	60X Address (1)	PCI Address
PCI to ISA Bridge	A/D 11	8080 08XXh	080 08XX
PCI slot 1	A/D 12	8080 10XXh	080 10XX
PCI slot 2	A/D 13	8080 20XXh	080 20XX
PCI slot 3	A/D 14	8080 40XXh	080 40XX
PCI slot 4	A/D 15	8080 80XXh	080 80XX
PCI slot 5	A/D 16	8081 00XXh	081 00XX
PCI slot 6	A/D 17	8082 00XXh	082 00XX
PCI slot 7	A/D 18	8084 00XXh	084 00XX
PCI slot 8	A/D 19	8088 00XXh	088 00XX
PCI slot 9	A/D 20	8090 00XXh	090 00XX
PCI slot 10	A/D 21	80A0 00XXh	0A0 00XX
PCI slot 11	A/D 22	80C0 00XXh	0C0 00XX

**Notes:**

1. This address is independent of contiguous I/O mode.

### 3.5.6 CPU to PCI Interrupt Acknowledge Transaction

Reading the interrupt acknowledge address (BFFF FFF0h) causes the bridge to arbitrate for the PCI bus and then to execute a standard PCI interrupt acknowledge transaction. The system interrupt controller in the ISA bridge claims the transaction and supplies the 1-byte interrupt vector. There is no physical interrupt vector BCR in the bridge. Other PCI busmasters can initiate interrupt acknowledge transactions, but this may have unpredictable effects.

### 3.5.7 PCI Lock

The 660 does not set PCI locks when acting as the PCI master. The PCI\_LOCK# signal in the 660 supports resource locking of one 32-byte cache sector (block) of system memory. Once a PCI lock is established, the block address is saved. Subsequent accesses to that block from other PCI busmasters or from the CPU bus are retried until the lock is released.

The bridge generates a flush-sector snoop cycle on the CPU bus when a PCI busmaster sets the PCI lock. The flush-sector snoop cycle causes the L1 and L2 caches to invalidate the locked block, which prevents cache hits on accesses to locked blocks. If the L1 contains modified data, the PCI cycle is retried and the modified data is pushed out to memory.

Note: The 60X processors do not have bus-locking functions. Instead, they use the *load reserve* and *store conditional* instructions (lwarx and stwcx) to implement exclusive access. To work with the lwarx and stwcx instructions, the 660 generates a flush-sector operation to the CPU in response to the PCI read that begins a PCI lock.

## 3.6 CPU to BCR Transfers

The 660 can be extensively programmed by means of the Bridge Control Registers (BCR). See the 660 User's Manual for a description of the operation and programming of the 660 BCRs.

### 3.7 L2

The MCM contains an L2 cache controller (located inside the 660), 512K of synchronous SRAM, and a 16K x 15 synchronous tagRAM. This forms a unified, write-thru, direct-mapped, look-aside level 2 cache (L2) that caches CPU memory space from 0 to 1G.

The L2 directory contains 8K entries with one tag per entry. The cache line (block) size is 32 bytes, the PowerPC 60X coherence unit. The L2 maintains coherence for the 32-byte block and neither operates on nor maintains the status of smaller units of memory.

Typical L2 read performance is 3-1-1-1, followed by -2-1-1-1 on pipelined reads. For more information on the operation and capabilities of the L2, see the 660 User's Manual.

#### 3.7.1 Cache Response to CPU Bus

The L2 supplies data to the CPU bus on burst read hits and snarfs the data (updates the SRAM data while the memory controller is accessing the DRAM) on CPU burst read/write misses. It snoops PCI to memory transactions, and it invalidates on PCI write hits. The L2 does not supply data to the PCI on read hits.

**Table 3-4. L2 Cache Responses to CPU Bus Cycles**

TT[0:3]	Type	CPU Bus Cycle	Cache Hit Action	Cache Miss Action
0000		Clean sector	Ignore	Ignore
0001	Single	Write with flush	Invalidate	Ignore
	Burst	Write with flush	Snarf	Snarf
0010		Flush sector	Invalidate	Ignore
0011	Single	Write with kill	Invalidate	Ignore
	Burst	Write with kill	Snarf	Snarf
0101	Single	Read	Ignore	Ignore
	Burst	Read	Claim cycle and supply data	Snarf
0110		Kill sector	Invalidate	Ignore
0111	Always Burst	RWITM	Claim cycle and supply data	Ignore
1000		Reserved	Ignore	Ignore
1001	Always Single	Write with flush atomic	Invalidate	Ignore
1010		External control out	Ignore	Ignore
1011		Reserved	Ignore	Ignore
1100		TLB invalidate	Ignore	Ignore
1101	Single	Read atomic	Ignore	Ignore
	Burst	Read atomic	Claim cycle and supply data	Snarf
1110		External control in	Ignore	Ignore
1111	Always Burst	RWITM Atomic	Claim cycle and supply data	Ignore

The L2 ignores CPU bus single-beat reads, and invalidates on CPU bus single-beat write hits. Table 3-4 shows the actions taken by the L2 cache based on transfer type and single-beat or burst mode.

The CPU cache inhibit (CI#) signal is not used because cache-inhibited bus operations are always single-beat. The 660 does not use TT[4]. Accesses to populated memory are snooped by L2 regardless of the state of GBL#. The 660 only uses GBL# as an output.



### 3.7.2 Cache Response to PCI Bus

The L2 maintains coherency during PCI to memory transactions as shown in Table 3-5. The L2 does not supply data to the PCI bus. The L2 is not updated during a PCI transaction.

Table 3-5. L2 Operations for PCI to Memory Transactions, 603 Mode

PCI Bus Transaction	CPU Bus Broadcast Snoop		L2 Operation	
	Operation	TT[0:4]	L2 Hit	L2 Miss
Memory Read	Single-Beat Read	01010	Ignore	Ignore
Memory Write	Single-Beat Write with Flush	00010	Invalidate Block	Ignore
Initiate Lock (Read)	Single-Beat Write with Flush	00010	Invalidate Block	Ignore

### 3.7.3 L2 Configuration

The L2 controller does not require any software configuration to set the size or organization of the tag and data SRAM. The connection to the CPU address bus determines the size of the tag RAM and data SRAM. Bridge Control Register D4 bit 3 should be set to 1 to select "burst" as the SRAM type.

### 3.7.4 L2 Organization

The L2 directory contains 8K entries with one tag per entry. The cache line (block) size is 32 bytes, the PowerPC 60X coherence unit. The L2 maintains coherence for the 32-byte block and neither operates on nor maintains the status of smaller units of memory. All of the tags together are sometimes called the cache directory.

The L2 uses only A[2:31] to store and access L2 data. A[0:1] are not used or saved in the cache directory because the MCM only caches the lowest 1G of the address space. A[0:1] must equal 00 for the MCM to access the directory.

### 3.7.5 Other L2 Related BCRs

See Table 3-6.

Table 3-6. Other L2 Related BCRs

Bridge Control Register	Index	R/W	Bytes
Error Enable 2	Index C4	R/W	1
Error Status 2	Index C5	R/W	1
Bridge Chip Set Options 3	Index D4	R/W	1
System Control 81C	8000 081C	R/W	1
L2 Invalidate	8000 0814	R	
L2 Error Status	8000 0842	R	
L2 Parity Error Read and Clear	8000 0843	R	
Cache Status	Index B1h	R/W	

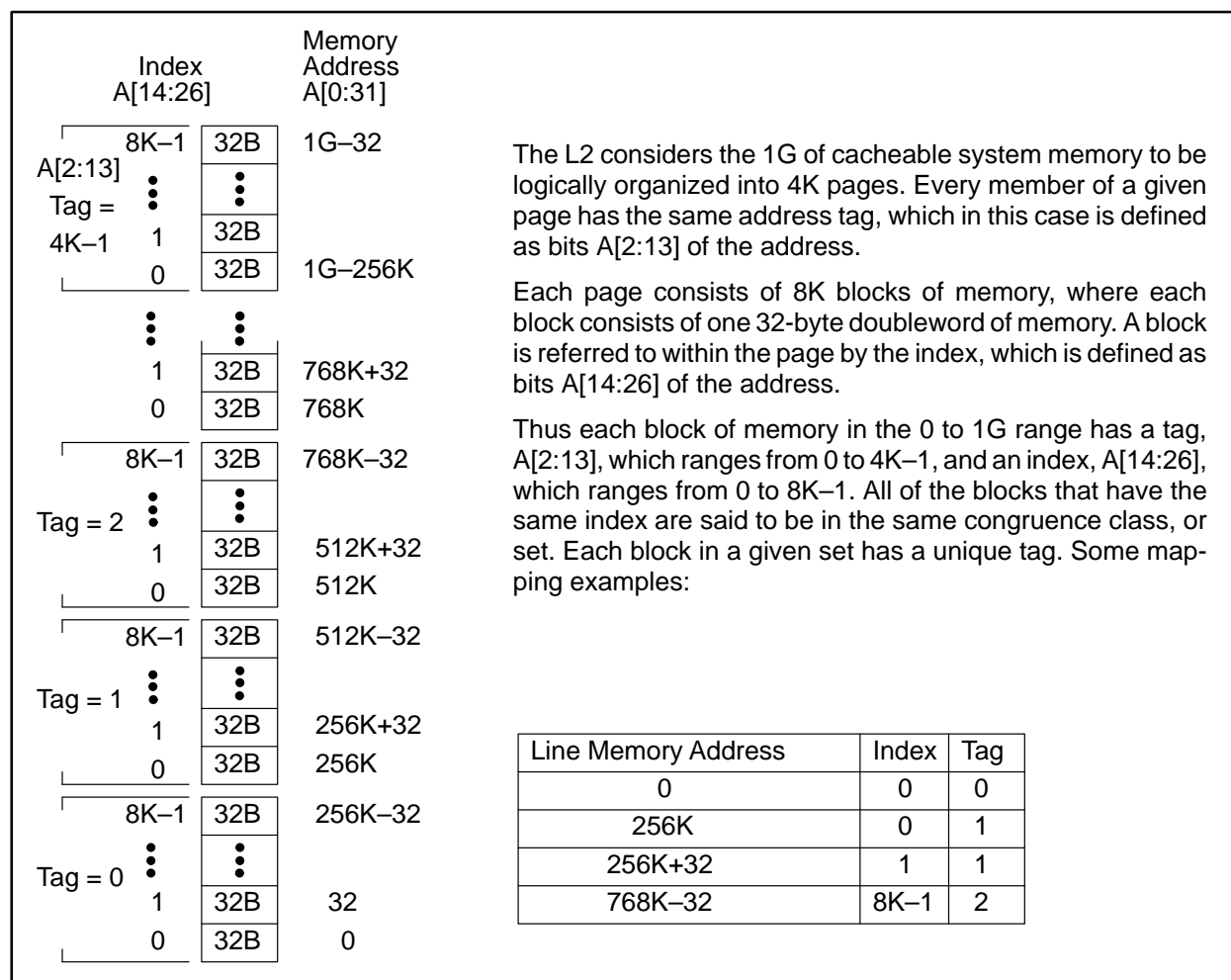
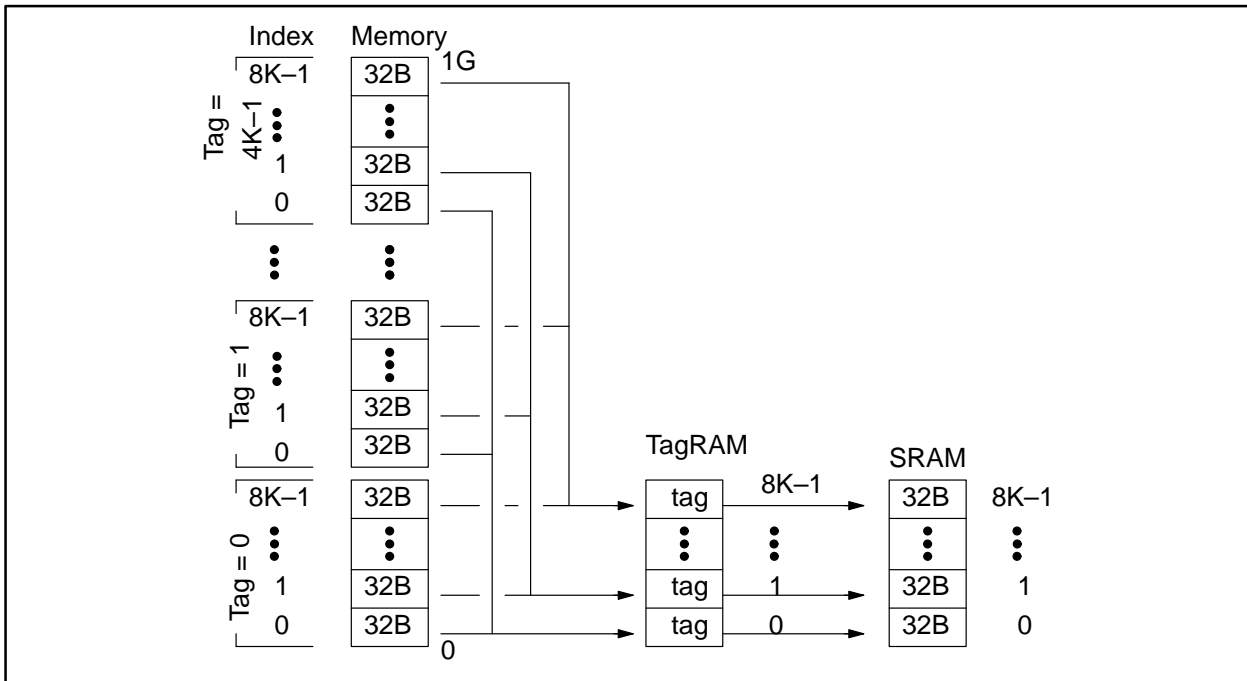
**Figure 3-4. L2 Mapping of System Memory – 512K Configuration**

Figure 3-5 shows how the MCM maps the SRAM to main memory using the index and tag fields of the address. Notice that there are 8k tags and 8k 32-byte blocks in the SRAM, and that the main memory is divided up into 4k pages, each one of which is composed of 8k blocks.

When an address is presented to the cache directory for snooping, the MCM uses the index to select which directory location to access, by presenting A[14:26] to the tagRAM address inputs. The tagRAM then compares the tag in that location (the A[2:13] of the previous cacheable access to that location) to the tag ( A[2:13] ) of the current transaction. If there is a match, and the tag is marked "valid," then there is a cache hit (signalled by TAG\_MATCH).



**Figure 3-5. SLC L2 Cache Directory – 512K Configuration**

Suppose the CPU requests a burst store to location 512K (8 0000h), which is initially invalid (either stale or never accessed). The index is A[14:26], which is 0h, so the accessed tagRAM location is 0h. The tag currently on the address bus is A[2:13], which equals 2. So the MCM stores 2h into tagRAM location 0h. The valid bit is set for that location.

While the CPU is accessing other blocks of memory with different low order addresses, other locations in the tagRAM are being accessed; however, if the CPU again accesses a block of memory with this same low order address, then this tagRAM location will again be accessed, and the tag stored therein will be compared against A[2:13] of the current access. If they are the same, then there is a cache hit.

### 3.7.6 SRAM

The MCM uses four IBM041814 synchronous 64K x 18 SRAMs to implement the SRAM portion of the L2. Figure 3-6 shows the basic connectivity of the MCM SRAM. These are synchronous devices, and each SRAM consumes one of the MPC970 clocks. In burst operation, the address of the data for the initial beat of the burst is latched into the SRAM; the address of the data for the next beat of the burst is incremented internally under the control of the ADV# input (SRAM\_CNT\_EN#).

- Each SRAM is connected to two CPU bus data bytes and the associated two CPU bus parity lines.
- ABUF[13:28] are a buffered copy of CPU A[13:28]. The SRAMs are arranged in parallel, so that an 8-byte doubleword is addressed during each access; thus, ABUF[28] is connected to A0 of the SRAM.
- The 660 asserts SRAM\_WE# to write into the SRAM and asserts SRAM\_OE# to read data out of the SRAM.
- The 660 asserts SRAM\_ADS# to signal the initial beat of the burst.

- Pull SRAM\_CS# low for normal operation. Pull SRAM\_ADSP# high for normal operation.

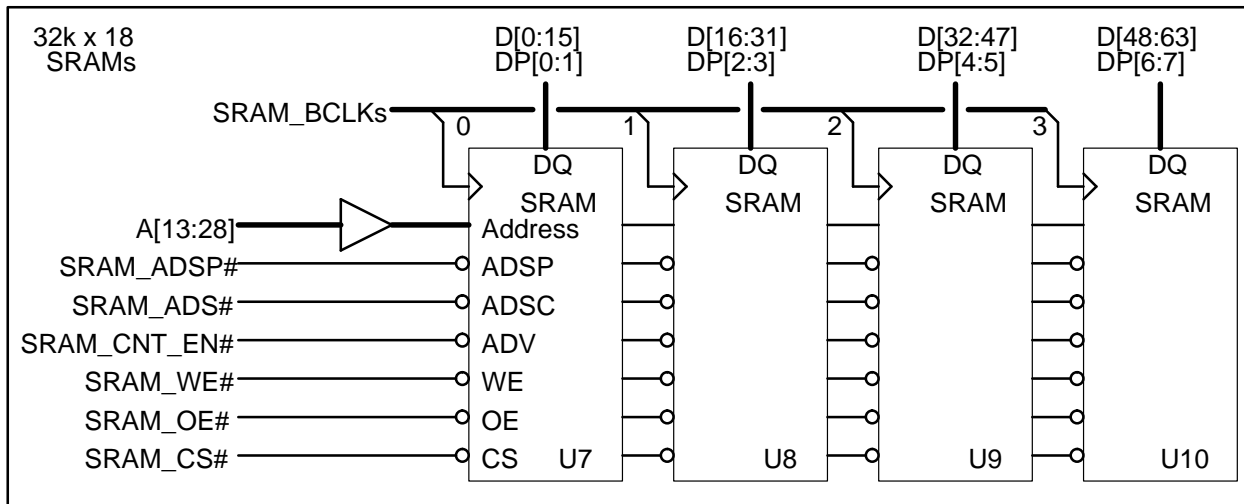


Figure 3-6. Synchronous SRAM, 512K L2

### 3.7.7 TagRAM

The MCM uses an IDT71216 synchronous 16K x 15 cache tagRAM to implement the tags of the L2. Figure 3-7 shows the basic connectivity of the MCM SRAM. The tagRAM is a synchronous device, and so consumes one of the MPC970 clocks.

- CPU address lines A[14:26] form the index of the directory entry. Pull TAG\_ADDR\_13 high for normal operation.
- CPU address lines A[2:13] form the tag of the directory entry. Tie TAG\_DATA\_11 to A13 for normal operation.
- During tagRAM writes, the valid bit associated with the index is set to match the TAG\_VALID input.
- During tagRAM reads, the TAG\_MATCH output is released to the active high (open-drain) state only when A[2:13] matches the contents of tagRAM location A[14:26], and the Valid bit for that location is set to 1. If the the Valid bit is 0 (invalid) or the address stored in the tag does not match the current value of A[2:13], then the TAG\_MATCH output is driven low.

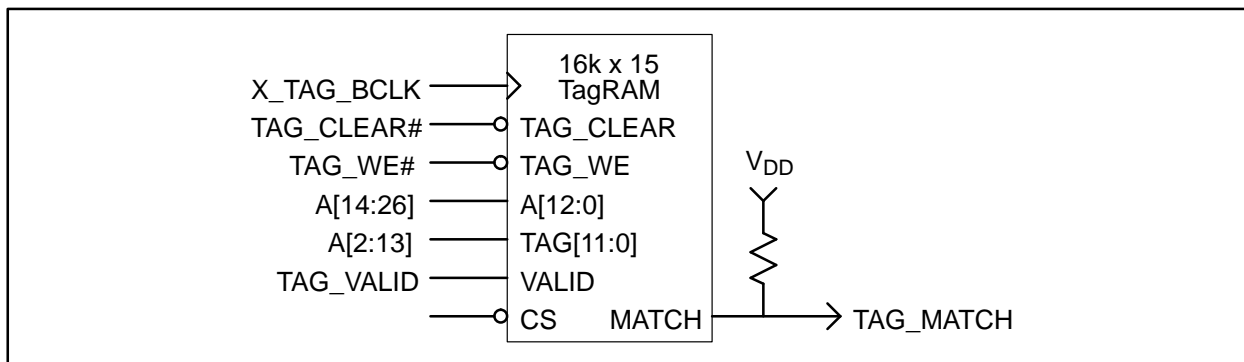


Figure 3-7. Synchronous TagRAM, 512K L2

## Section 4

### DRAM

The memory controller in the 660 bridge controls the system memory DRAM. The system memory can be accessed from both the CPU bus and the PCI bus. Much of the information in this section has been drawn from the 660 User's Manual.

#### 4.1 Features and Supported Devices

- Supports memory operations for the PowerPC Architecture™
- Data bus path 72 bits wide—64 data bits and eight bits of optional ECC or parity data
- Eight SIMM sockets supported
- Eight RAS# outputs, eight CAS# outputs, and two write-enable outputs
- Supports industry-standard 8-byte (168-pin) SIMMs of 8M, 16M, 32M, 64M, and 128M that can be individually installed for a minimum of 8M and a maximum of 1G
- Supports industry-standard 4-byte (72-pin) SIMMs of 4M, 8M, 16M, 32M, 64M, and 128M that must be installed in pairs for a minimum of 8M and a maximum of 1G
- Mixed use of different size SIMMs, including mixed 4-byte and 8-byte SIMMs
- Full refresh support, including refresh address counter and programmable DRAM refresh timer
- Burst-mode memory address generation logic
  - 32-byte CPU bursts to memory
  - Variable length PCI burst to memory
- Little-endian and big-endian addressing and byte swapping modes
- Provides row and column address multiplexing for DRAM SIMMs requiring the following addressing:

SIMM type	SIMM size	Addressing
72-pin	4 Meg	10 x 10
	8 Meg	10 x 10
	16 Meg	11 x 11
	32 Meg	11 x 11
	64 Meg	12 x 12
168-pin	8 Meg	10 x 10
	16 Meg	11 x 10
	32 Meg	12 x 10 or 11 x 11
	64 Meg	12 x 11
	128 Meg	12 x 12

### 4.1.1 SIMM Nomenclature

The term SIMM is used extensively to mean DRAM memory module, without implying the physical implementation of the module, which can be a SIMM, DIMM, or other package.

#### 4.1.1.1 DRAM Timing

The memory controller timing parameters are programmable to allow optimization of timings based on speed of DRAM, clock frequency, and layout topology. Timing must be programmed based on the slowest DRAM installed.

- Support for fast page-mode DRAMs
- Support for extended-data-out (EDO) DRAM (hyper-page mode)
- If 70ns DRAM is used in a system with the CPU bus at 66MHz, the minimum access times for initial (not pipelined) CPU to memory transfers with page mode and EDO DRAM are as follows:

Transfer	EDO DRAM	Page Mode DRAM	Note
Initial Read Burst	10-3-3-3	11-4-4-4	CPU clocks for 32 bytes
Initial Write Burst	5-3-3-3	5-4-4-4	CPU clocks for 32 bytes

- In the same system, the times for a pipelined burst following a read are as follows:

Transfer	EDO DRAM	Page Mode DRAM	Note
Page Hit Read	-5-3-3-3	-5-4-4-4	CPU clocks for 32 bytes
Page Hit Write	-3-3-3-3	-3-3-4-4	CPU clocks for 32 bytes

- In the same system, the times for a pipelined burst following a write are as follows:

Transfer	EDO DRAM	Page Mode DRAM	Note
Page Hit Read	-9-3-3-3	-11-4-4-4	CPU clocks for 32 bytes
Page Hit Write	-5-3-3-3	-6-3-4-4	CPU clocks for 32 bytes

- Other minimum memory timings are as follows:

PCI to memory read at 66MHz CPU and 33MHz PCI

8-1-1-1 -1-1-1-1 7-1-1-1 -1-1-1-1 7-1-1-1 -1-1-1-1 ... 7-1-1-1 -1-1-1-1

PCI to memory write at 66MHz CPU and 33MHz PCI

5-1-1-1 -3-1-1-1 3-1-1-1 -3-1-1-1 3-1-1-1 -3-1-1-1 ... 3-1-1-1 -3-1-1-1

#### 4.1.1.2 DRAM Error Checking

The 660 supports either no parity or one bit per byte parity DRAM SIMMs, in which one parity bit is associated and accessed with each byte. The 660 is BCR programmable to support either no parity, even parity, or ECC data error detection and correction. ECC is implemented using standard parity SIMMs. All installed SIMMs must support the selected error checking protocol.

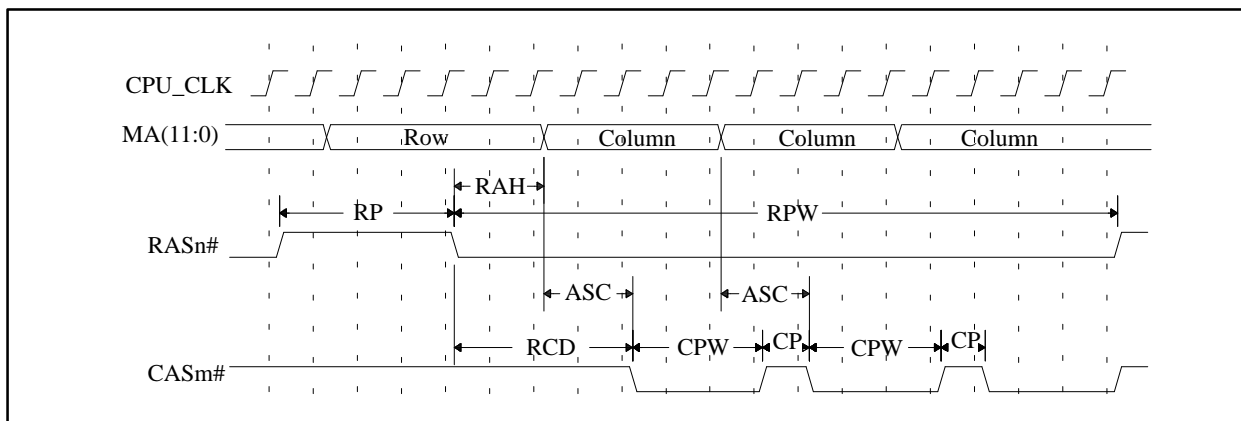
Systems without error checking cost the least. Parity checking mode allows a standard level of error protection with no performance impact. ECC mode allows detection and correction of all single-bit errors and detection of all two-bit errors. ECC mode adds one CPU clock to the latency of CPU to memory reads, and does not effect the timing of 8-byte and 32-byte writes

## 4.2 DRAM Performance

### 4.2.1 Memory Timing Parameters

Most memory controller timing parameters can be adjusted to maximize the performance of the system with the available resources. This adjustment is done by programming vari-

ous memory controller BCRs. Figure 4-1 shows the various programmable memory timing variables. These variables control the number of CPU\_CLKs between various events. The actual amount of time between the events shown will also be affected by various other factors such as clock to output delays. The CPU\_CLK signal shown is not meant to be contiguous, as the number of clocks between various events is programmable.



**Figure 4-1. CPU to Memory Transfer Timing Parameters**

Table 4-1 shows the function, location, and section references for the variables shown in Figure 4-1.

**Table 4-1. Memory Timing Parameters**

Variable	Function	BCR	Section
ASC	Column Address Setup (min)	Memory Timing Register 2	4.2.1.2
CP	CAS# Precharge	Memory Timing Register 2	4.2.1.2
CPW	CAS# Pulse Width (Read & Write)	Memory Timing Register 2	4.2.1.2
RAH	Row Address Hold (min)	Memory Timing Register 1	4.2.1.1
RCD	RAS# to CAS# Delay (min)	Memory Timing Register 2	4.2.1.2
RP	RAS# Precharge	Memory Timing Register 1	4.2.1.1
RPW	RAS# Pulse Width	Memory Timing Register 1	4.2.1.1

Note that ASC, RAH, and RCD are minimums. If  $RAH + ASC$  does not equal RCD, then the larger value will be used such that:

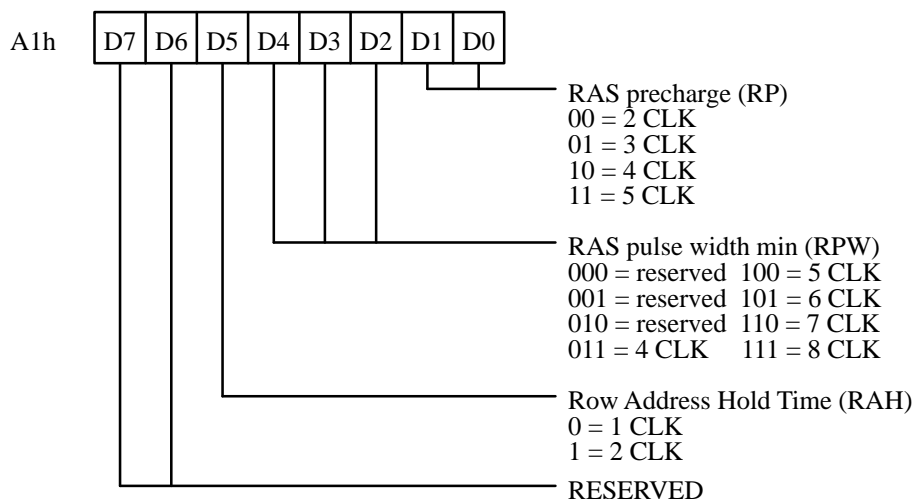
- If  $RCD < RAH + ASC$ , then the actual RCD will be stretched to equal  $RAH + ASC$ .
- If  $RCD > RAH + ASC$ , then the actual RAH will be stretched to equal  $RCD - ASC$ .

#### 4.2.1.1 Memory Timing Register 1

Index	A1	Read/Write	Reset to 3Fh
-------	----	------------	--------------

This BCR determines the timing of RAS# signal assertion for memory cycles. RAS# timing must support the worst-case timing for the slowest DRAM installed in the system. See Section 4.2.1.

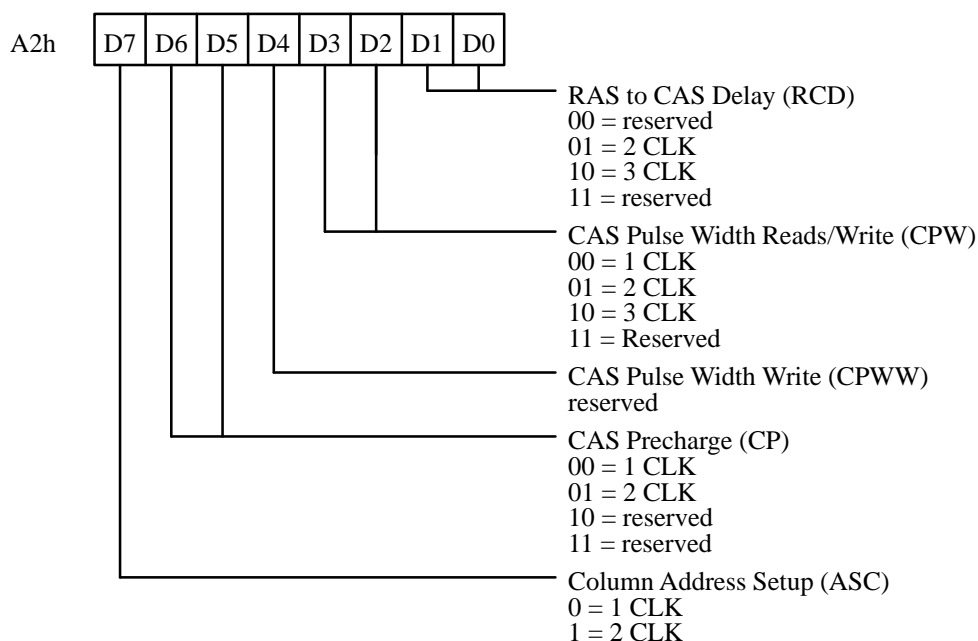
- Bits 1:0 These bits control the number of CPU clocks for RAS# precharge.
- Bits 4:2 These bits control the minimum allowed RAS# pulse width except on refresh. For refresh, the RAS# pulse width is hard-coded to three PCI clocks.
- Bit 5 This bit controls the number of CPU clocks that the row address is held following the assertion of RAS#.



#### 4.2.1.2 Memory Timing Register 2

Index	A2	Read/Write	Reset to AEh
-------	----	------------	--------------

This BCR determines the timing of CAS# signal assertion for memory cycles. CAS# timing must support the worst-case timing for the slowest DRAM installed in the system. See Section 4.2.1.



#### 4.2.1.3 RAS# Watchdog Timer BCR

Index	B6	Read/Write	Reset to 53h
-------	----	------------	--------------

This BCR limits the maximum RAS# active pulse width. The value of this BCR represents the maximum amount of time that any RAS# can remain active in units of eight CPU bus clocks. The timer (down-counter) associated with this BCR is reloaded on the assertion of any RAS# line. On expiration of the timer, the 660 drops out of page mode to deassert the RAS# lines.



In response to the RESET# signal, this register is reset to 53h. This value results in a maximum RAS# active time of just under 10us at 66MHz. This is the value required by most 4-byte and 8-byte SIMMs. The value of the BCR must be reprogrammed if the CPU bus frequency is not 66MHz or a different RAS# pulse width is required.

#### 4.2.1.4 DRAM Timing Calculations

The memory controller of the 660 features programmable DRAM access timing. DRAM timing is programmed into Memory Timing Register 1 (MTR1) and Memory Timing Register 2 (MTR2). See section 4.2.1 for an explanation of the format of these BCRs. All memory controller outputs are switched on the rising edge of the CPU clock, with the exception of signal assertion during refresh operations, which is timed from the PCI clock. The RAS# Watchdog Timer Register, the Refresh Timer Divisor Register and the Bridge Chipset Options 3 (BCO3) BCRs also have an effect on DRAM timing.

The values programmed into these registers are a function of the clock frequencies, the timing requirements of the memory, the amount of memory installed (capacitive loading), the mode of operation (EDO vs. standard), the timing requirements of the 660, the type and arrangement of buffering for the MA (memory address) signals, the clock skew between the 663 and the 664, and the net lengths of the signals to/from the memory (flight time). The calculations below ignore the factors of clock skew and flight time.

This section discusses the timing calculations that are appropriate to 660 memory controller design. The timing recommendations in this section apply to all MCM configurations. Because the MCM uses synchronous SRAM, it is not affected by the 660 DRAM special case timing restrictions.

Each of the nine equations below lists a register or register bits that govern a memory timing parameter. An equation is then provided for calculating the required value based on the timing requirements of the memory and the 660. MTR1[1:0] refers to Memory Timing Register 1 bits 1:0. See Figure 4-1 and Table 4-1.

1. MTR1[1:0] – RAS# precharge (RP). The critical path that determines the RAS# precharge requirement is RAS# rising to RAS# falling. The minimum RAS# precharge time supplied by the 660 must exceed the minimum precharge time required by the DRAM. Make:

$$\text{RAS\# precharge (RP)} > \text{Trp min} \dots\dots\dots * \text{DRAM min RAS\# precharge}$$

2. MTR1[4:2] – RAS# pulse width (RPW). The critical path that determines the RAS# pulse width requirement is RAS# falling to RAS# rising. The minimum RAS# pulse width supplied by the 660 must exceed the minimum RAS# pulse width required by the DRAM plus 5ns. Make:

$$\begin{aligned} \text{RAS\# pulse width (RPW)} > \text{Tras min} \dots\dots\dots * \text{DRAM min RAS\# pulse width} \\ + 5\text{ns} \dots\dots\dots * \text{pulse width shrinks (note 1)} \end{aligned}$$

3. MTR2[6:5] – CAS# precharge (CP). The critical path that determines the CAS# precharge requirement is CAS# rising to CAS# falling. The minimum CAS# precharge time supplied by the 660 must exceed the minimum precharge time required by the DRAM. Make:

CAS# precharge (CP) > T<sub>cp</sub> min ..... \* DRAM min CAS# precharge

4. MTR2[3:2] – CAS# pulse width (CPW). The critical path that determines the CAS# pulse width requirement is the data access time from CAS# plus the setup time into the 663. Thus the minimum CAS# pulse width provided by the 660 must exceed the minimum CAS# pulse width required by the DRAM, plus these factors. Note that the 663 samples memory data on the clock that CAS# is deasserted for standard DRAM and on the clock after CAS# is deasserted for EDO DRAM. Make:

CAS# pulse width (CPW) >	T CAS# fall max	* CAS# active out if 664
(+ 1 CLK if EDO)	+ Tcac . . . . .	* DRAM data access from CAS#
	+ MD setup max	* MEM DATA setup into 663

The 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. MTR2[7] – Column Address Setup (ASC). There are two critical paths that determine the Col addr setup requirement. The minimum column address setup time supplied by the 660 must exceed both constraints. The first is Tasc of the memory. Make:

a) Col Addr Setup (ASC) >	Tasc min	* DRAM min col addr setup time
	+ T MA max . . .	* MA[11:0] valid out of 664
	+ T 244 max . .	* Buffer delay
	– T CAS# max .	* CAS# active out of 664

The second critical path is the data access time from MA plus the setup time into the 663. Make:

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

6. MTR2[1:0] – RAS# to CAS# delay. The minimum RAS# to CAS# delay provided by the 660 must exceed the timing of the critical path that determines the RAS# to CAS# delay, which is the data access time from RAS# plus the setup time into the 663. Make:

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

7. MTR1[5] – Row address hold time. The row address hold time must be set to the RAS#–to–CAS# delay minus the Column address setup. The timing for the row address hold time can also be calculated as follows. Make:

$$\begin{aligned}
 \text{Row addr hold (RAH)} &> \text{Trah min} && * \text{DRAM min row addr hold} \\
 &+ \text{T244 min} && * \text{Buffer delay} \\
 &+ \text{T MA max} && * \text{MA[11:0] valid out of 664} \\
 &- \text{T RAS\# fall max} && * \text{max 660 RAS\# fall time} \\
 &&& \text{(note 1)}
 \end{aligned}$$

**Note 1:** The 664 drivers that drive the RAS# and CAS# signals are slower falling than rising. This causes active high pulse widths to grow by 0 to 5ns and active low pulse widths to shrink by 0 to 5ns.

8. Refresh Timer Divisor. The refresh timing divisor is clocked by the PCI clock. The required value is calculated as follows:

$$\text{Refresh rate} = \text{period}(\text{Tref}) / \text{period}(\text{PCI clock})$$

9. RAS# watchdog timer. The RAS# watchdog timer must be set to limit the max RAS# pulse width:

$$\text{RAS\# watchdog timer} = \text{Tras max} / [\text{period}(\text{CPU clock}) * 8].$$

#### 4.2.1.5 DRAM Timing Examples

This section presents example DRAM timing calculations based on the equations found in Section 4.2.1.4. Except as noted in Section 4.2.1.4, the timing recommendations in this section apply to all 660 configurations.

In the equations below, first the capacitive loads are calculated based on the quantity and types of SIMMs and the buffers used. Next, the timing characteristics are calculated based on the capacitive loads. Finally, the timing requirements and register values are calculated.

#### 4.2.1.6 70ns DRAM Calculations

Ex 1: Assume 70ns standard DRAM memory, four 72-pin DRAM SIMMs, a CPU bus cycle time of 15ns (66.7Mhz), and MA[11:0] buffered by an FCT244 (four SIMMs per driver).

Capacitive loads:

$$\begin{aligned}
 \text{RAS\#} &= 2 * 62\text{pf} + 30\text{pf} = 154\text{pf} \\
 \text{CAS\#} &= 2 * 62\text{pf} + 30\text{pf} = 154\text{pf} \\
 \text{MA (to buffer)} &= 30\text{pf} \\
 \text{MA (to memory)} &= 4 * 161\text{pf} + 40\text{pf} = 684\text{pf}
 \end{aligned}$$

Timing Characteristics:

$$\begin{aligned}
 \text{RAS\#} &= 13.2\text{ns} + .025 * (154\text{pf} - 50\text{pf}) = 16\text{ns} \\
 \text{CAS\#} &= 13.3\text{ns} + .025 * (154\text{pf} - 50\text{pf}) = 16\text{ns} \\
 \text{MA to buffer} &= 13.6\text{ns} + .025 * (30\text{pf} - 50\text{pf}) = 13\text{ns} \\
 \text{MA to memory} &= 4.6\text{ns} + .007 * (684\text{pf} - 50\text{pf}) = 9\text{ns} \\
 \text{663 memory data input setup} &= 6\text{ns}
 \end{aligned}$$

664 Output timings are at 50pf. For loads greater than 50pf, 0.025ns/pf are added.

Timing Requirements and register value calculations:

1.  $4 \text{ CLK} * 15\text{ns} > 50\text{ns}$   
 $60\text{ns} > 50\text{ns} \dots\dots\dots \text{MTR1}[1:0]=10$
2.  $5 \text{ CLK} * 15\text{ns} > 70\text{ns} + 5\text{ns}$   
 $75\text{ns} > 75\text{ns} \dots\dots\dots \text{MTR1}[4:2]=100$

3.  $1 \text{ CLK} * 15\text{ns} > 10\text{ns}$   
 $15\text{ns} > 10\text{ns} \dots\dots\dots \text{MTR2}[6:5]=00$
4.  $3 \text{ CLK} * 15\text{ns} > 16\text{ns} + 20\text{ns} + 6\text{ns}$   
 $45\text{ns} > 42\text{ns} \dots\dots\dots \text{MTR2}[3:2]=10$
5. a.  $1 \text{ CLK} * 15\text{ns} > 0\text{ns} + 13\text{ns} + 9\text{ns} - 16\text{ns}$   
 $15\text{ns} > 6\text{ns}$   
     b.  $(1 + 3)\text{CLK} * 15\text{ns} > 13\text{ns} + 9\text{ns} + 35\text{ns} + 6\text{ns}$   
 $60\text{ns} > 63\text{ns} \dots\dots\dots \text{MTR2}[7]=0 \text{ (see Note 2)}$
6.  $(3 + 3)\text{CLK} * 15\text{ns} > 16\text{ns} + 70\text{ns} + 6\text{ns}$   
 $90\text{ns} > 92\text{ns} \dots\dots \text{MTR2}[1:0]=10 \text{ (see Note 2)}$
7.  $2 \text{ CLK} * 15\text{ns} > 10\text{ns} + 4\text{ns} + 13\text{ns} - 16\text{ns}$   
 $30\text{ns} > 11\text{ns} \dots\dots\dots \text{MTR1}[5]=1$

Results: Memory Timing Register 1 (index A1h) = 32h

Memory Timing Register 2 (index A2h) = 0Ah

**Note 2:** The timing analysis above includes two timing violations (path #5b is violated by 5% and path #6 is violated by 2%). More conservative system designers may wish to use the values MTR1=32h, MTR2=0Eh to ensure all timing requirements are met under all worst-case conditions.

8. Refresh rate =  $15.6\mu\text{s}/30\text{ns} = 520\text{d} = 208\text{h}$   
 Refresh Timer Divisor (index D1h,D0h) = 0208h
9. RAS# watchdog timer =  $10,000\text{ns} / (15\text{ns} * 8) = 83\text{d} = 53\text{h}$   
 RAS# watchdog timer register (index B6h) = 53h (default value).

#### 4.2.1.7 60ns DRAM Calculations

Ex 2: Same assumptions as above, but with 60ns DRAM

1.  $3 \text{ CLK} * 15\text{ns} > 40\text{ns}$   
 $45\text{ns} > 40\text{ns} \dots\dots\dots \text{MTR1}[1:0]=01$
2.  $5 \text{ CLK} * 15\text{ns} > 60\text{ns} + 5\text{ns}$   
 $75\text{ns} > 65\text{ns} \dots\dots\dots \text{MTR1}[4:2]=100$
3.  $1 \text{ CLK} * 15\text{ns} > 10\text{ns}$   
 $15\text{ns} > 10\text{ns} \dots\dots\dots \text{MTR2}[6:5]=00$
4.  $3 \text{ CLK} * 15\text{ns} > 16\text{ns} + 15\text{ns} + 6\text{ns}$   
 $45\text{ns} > 37\text{ns} \dots\dots\dots \text{MTR2}[3:2]=10$
5. a.  $1 \text{ CLK} * 15\text{ns} > 0\text{ns} + 13\text{ns} + 9\text{ns} - 16\text{ns}$   
 $15\text{ns} > 6\text{ns}$   
     b.  $(1 + 3)\text{CLK} * 15\text{ns} > 13\text{ns} + 9\text{ns} + 30\text{ns} + 6\text{ns}$   
 $60\text{ns} > 58\text{ns} \dots\dots\dots \text{MTR2}[7]=0$
6.  $(2 + 3)\text{CLK} * 15\text{ns} > 16\text{ns} + 60\text{ns} + 6\text{ns}$   
 $75\text{ns} > 82\text{ns} \dots\dots\dots \text{MTR2}[1:0]=01 \text{ (see Note 3)}$
7.  $1 \text{ CLK} * 15\text{ns} > 10\text{ns} + 4\text{ns} + 13\text{ns} - 16\text{ns}$   
 $15\text{ns} > 11\text{ns} \dots\dots \text{MTR1}[5]=0$

Results: Memory Timing Register 1 (index A1h) = 11h

Memory Timing Register 2 (index A2h) = 09h

**Note 3:** The timing analysis above includes one timing violation (path #6 is violated by 9%). More conservative system designers may wish to use the values MTR1=31h, MTR2=0Ah to ensure all timing requirements are met under complete worst-case conditions.

8. Refresh rate =  $15.6\mu\text{s}/30\text{ns} = 520\text{d} = 208\text{h}$   
Refresh Timer Divisor (index D1h,D0h) = 0208h
9. RAS# watchdog timer =  $10,000\text{ns} / (15\text{ns} \times 8) = 83\text{d} = 53\text{h}$   
RAS# watchdog timer register (index B6h) = 53h (default value).

#### 4.2.1.8 50ns DRAM Calculations

Ex 3: Same assumptions as above, but with 50ns DRAM

1.  $2 \text{ CLK} \times 15\text{ns} > 30\text{ns}$   
 $30\text{ns} > 30\text{ns} \dots\dots\dots \text{MTR1}[1:0]=00$
2.  $4 \text{ CLK} \times 15\text{ns} > 50\text{ns} + 5\text{ns}$   
 $60\text{ns} > 55\text{ns} \dots\dots\dots \text{MTR1}[4:2]=011$
3.  $1 \text{ CLK} \times 15\text{ns} > 10\text{ns}$   
 $15\text{ns} > 10\text{ns} \dots\dots\dots \text{MTR2}[6:5]=00$
4.  $2 \text{ CLK} \times 15\text{ns} > 16\text{ns} + 13\text{ns} + 6\text{ns}$   
 $30\text{ns} > 35\text{ns} \dots\dots\dots \text{MTR2}[3:2]=01$  (see note 4)
5. a.  $1 \text{ CLK} \times 15\text{ns} > 0\text{ns} + 13\text{ns} + 9\text{ns} - 16\text{ns}$   
 $15\text{ns} > 6\text{ns}$   
b.  $(1 + 2)\text{CLK} \times 15\text{ns} > 13\text{ns} + 9\text{ns} + 25\text{ns} + 6\text{ns}$   
 $45\text{ns} > 53\text{ns} \dots\dots\dots \text{MTR2}[7]=0$  (see note 4)
6.  $(3 + 2)\text{CLK} \times 15\text{ns} > 16\text{ns} + 50\text{ns} + 6\text{ns}$   
 $75\text{ns} > 72\text{ns} \dots\dots\dots \text{MTR2}[1:0]=10$
7.  $2 \text{ CLK} \times 15\text{ns} > 10\text{ns} + 4\text{ns} + 13\text{ns} - 16\text{ns}$   
 $30\text{ns} > 11\text{ns} \dots\dots\dots \text{MTR1}[5]=1$

Results: Memory Timing Register 1 (index A1h) = 2Ch  
Memory Timing Register 2 (index A2h) = 06h

**Note 4:** The timing analysis above includes two timing violations (path #4 is violated by 14% and path #5b is violated by 15%). More conservative system designers may wish to use the values MTR1=0Ch, MTR2=09h to ensure all timing requirements are met under complete worst-case conditions.

8. Refresh rate =  $15.6\mu\text{s}/30\text{ns} = 520\text{d} = 208\text{h}$   
Refresh Timer Divisor (index D1h,D0h) = 0208h
9. RAS# watchdog timer =  $10,000\text{ns} / (15\text{ns} \times 8) = 83\text{d} = 53\text{h}$   
RAS# watchdog timer register (index B6h) = 53h (default value).

#### 4.2.1.9 60ns EDO DRAM Calculations

Ex 4: Same assumptions as above, except using 60ns EDO DRAM

1.  $3 \text{ CLK} \times 15\text{ns} > 40\text{ns}$   
 $45\text{ns} > 40\text{ns} \dots\dots\dots \text{MTR1}[1:0]=01$

2.  $5 \text{ CLK} * 15\text{ns} > 60\text{ns} + 5\text{ns}$   
 $75\text{ns} > 65\text{ns} \dots\dots\dots \text{MTR1}[4:2]=100$
3.  $1 \text{ CLK} * 15\text{ns} > 10\text{ns}$   
 $15\text{ns} > 10\text{ns} \dots\dots\dots \text{MTR2}[6:5]=00$
4.  $(2 + 1)\text{CLK} * 15\text{ns} > 16\text{ns} + 15\text{ns} + 6\text{ns}$   
 $45\text{ns} > 37\text{ns} \dots\dots\dots \text{MTR2}[3:2]=01$
5. a.  $1 \text{ CLK} * 15\text{ns} > 0\text{ns} + 13\text{ns} + 9\text{ns} - 16\text{ns}$   
 $15\text{ns} > 6\text{ns}$   
     b.  $(1 + 2 + 1)\text{CLK} * 15\text{ns} > 13\text{ns} + 9\text{ns} + 30\text{ns} + 6\text{ns}$   
 $60\text{ns} > 58\text{ns} \dots\dots\dots \text{MTR2}[7]=0$
6.  $(2 + 2 + 1)\text{CLK} * 15\text{ns} > 16\text{ns} + 60\text{ns} + 6\text{ns}$   
 $75\text{ns} > 82\text{ns} \dots\dots\dots \text{MTR2}[1:0]=01$  (see Note 5)
7.  $1 \text{ CLK} * 15\text{ns} > 10\text{ns} + 4\text{ns} + 13\text{ns} - 16\text{ns}$   
 $15\text{ns} > 11\text{ns} \dots\dots\dots \text{MTR1}[5]=0$

Results: Memory Timing Register 1 (index A1h) = 11h  
 Memory Timing Register 2 (index A2h) = 05h

**Note 5:** The timing analysis above includes one timing violation (path #6 is violated by 9%). More conservative system designers may wish to use the values MTR1=31h, MTR2=06h to ensure all timing requirements are met under complete worst-case conditions.

8. Refresh rate =  $15.6\mu\text{s}/30\text{ns} = 520\text{d} = 208\text{h}$   
 Refresh Timer Divisor (index D1h,D0h) = 0208h
9. RAS# watchdog timer =  $10,000\text{ns} / (15\text{ns}*8) = 83\text{d} = 53\text{h}$   
 RAS# watchdog timer register (index B6h) = 53h (default value).

#### 4.2.1.10 Aggressive Timing Summary

Table 4-2 contains a summary of recommended general case aggressive page mode DRAM timing, the required control register settings, and the resulting performance of the memory controller. Aggressive timings may generate slight violations of certain worst case timing constraints. In many cases, these violations are of only theoretical interest, since the conditions required to produce the violations are of such low probability.

The top section of Table 4-2 shows the settings of the memory controller BCRs. The next section of the table shows access times from the memory controller idle state, which it enters when it is not servicing a read or write request. The other two sections of the table show access times during back to back burst transfers. The middle section of the table shows access times for the second of any pair of back to back transfers where the first transfer is a read. The lowest section of the table shows access times for the second of any pair of back to back transfers where the first transfer is a write.

Table 4-2. Page Mode DRAM Aggressive Timing Summary (1)

Transfer	70ns Page Aggressive	60ns Page Aggressive	50ns Page Aggressive	Note
Memory Timing Register 1	32	11	2C	
Memory Timing Register 2	0A	09	06	
Bridge Chipset Options 3	08	08	08	
Initial Read Burst	11-4-4-4	10-4-4-4	10-3-3-3	
Initial Write Burst	5-4-4-4	5-3-4-4	5-4-3-3	
For a pipelined burst transfer immediately following a read:				
Page Hit Read	-4-4-4-4	-4-4-4-4	-4-3-3-3	
Page Hit Write	-3-3-4-4	-3-3-4-4	-3-3-3-3	
Page Miss and Bank Miss Read	-8-4-4-4	-7-4-4-4	-7-3-3-3	(2)
Page Miss and Bank Hit Read	-10-4-4-4	-8-4-4-4	-7-3-3-3	(3)
Page Miss and Bank Miss Write	-3-3-4-4	-3-3-4-4	-3-3-3-3	(2)
Page Miss and Bank Hit Write	-3-5-4-4	-3-3-4-4	-3-3-3-3	(3)
For a pipelined burst transfer immediately following a write:				
Page Hit Read	-11-4-4-4	-10-4-4-4	-8-3-3-3	
Page Hit Write	-6-3-4-4	-5-3-4-4	-4-3-3-3	
Page Miss and Bank Miss Read	-14-4-4-4	-13-4-4-4	-11-3-3-3	(2)
Page Miss and Bank Hit Read	-16-4-4-4	-14-4-4-4	-11-3-3-3	(3)
Page Miss and Bank Miss Write	-6-6-4-4	-6-5-4-4	-5-5-3-3	(2)
Page Miss and Bank Hit Write	-6-8-4-4	-6-6-4-4	-6-5-3-3	(3)

**Notes:**

1. Refresh Rate set to 0208h. RAS# watchdog timer BCR set to 53h.
2. 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#.
3. 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#.

Table 4-3 contains a summary of recommended general case aggressive EDO DRAM timing, the required control register settings, and the resulting performance of the memory controller.

Table 4-3. EDO DRAM Aggressive Timing Summary (1)

Transfer	70ns EDO Aggressive	60ns EDO Aggressive	50ns EDO Aggressive	Note
Memory Timing Register 1	32	11	2C	
Memory Timing Register 2	06	05	02	
Bridge Chipset Options 3	0C	0C	0C	
Initial Read Burst	11-3-3-3	10-3-3-3	10-2-2-2	
Initial Write Burst	5-4-3-3	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	
Page Hit Write	-3-3-3-3	-3-3-3-3	-3-2-2-2	
Page Miss & Bank Miss Read	-8-3-3-3	-7-3-3-3	-7-2-2-2	(2)
Page Miss & Bank Hit Read	-10-3-3-3	-8-3-3-3	-7-2-2-2	(3)
Page Miss & Bank Miss Write	-3-3-3-3	-3-3-3-3	-3-3-2-2	(2)
Page Miss & Bank Hit Write	-3-5-3-3	-3-3-3-3	-3-3-2-2	(3)
For a pipelined burst transfer immediately following a write:				
Page Hit Read	-9-3-3-3	-9-3-3-3	-7-2-2-2	
Page Hit Write	-5-3-3-3	-5-3-3-3	-4-2-2-2	
Page Miss & Bank Miss Read	-12-3-3-3	-11-3-3-3	-10-2-2-2	(2)
Page Miss & Bank Hit Read	-14-3-3-3	-12-3-3-3	-10-2-2-2	(3)
Page Miss & Bank Miss Write	-5-5-3-3	-5-4-3-3	-4-5-2-2	(2)
Page Miss & Bank Hit Write	-5-7-3-3	-5-5-3-3	-4-5-2-2	(3)

**Notes:** See Table 4-2 for notes.

#### 4.2.1.11 Conservative Timing Summary

Table 4-4 contains a summary of recommended general case conservative page mode DRAM timing. The table also shows the resulting performance of the memory controller. These conservative timings may be too conservative for many applications. These timings meet all of the worst case timing constraints for the systems described in the examples sections above.



Table 4-4. Page Mode DRAM Conservative Timing Summary (1)

Transfer	70ns Page Conservative	60ns Page Conservative	50ns Page Conservative	Note
Memory Timing Register 1	32	31	0C	
Memory Timing Register 2	0E	0A	09	
Bridge Chipset Options 3	08	08	08	
Initial Read Burst	12-5-5-5	11-4-4-4	10-4-4-4	
Initial Write Burst	5-4-5-5	5-4-4-4	5-3-4-4	
For a pipelined burst transfer immediately following a read:				
Page Hit Read	-5-5-5-5	-4-4-4-4	-4-4-4-4	
Page Hit Write	-3-3-5-5	-3-3-4-4	-3-3-4-4	
Page Miss and Bank Miss Read	-9-5-5-5	-8-4-4-4	-7-4-4-4	(2)
Page Miss and Bank Hit Read	-11-5-5-5	-9-4-4-4	-7-4-4-4	(3)
Page Miss and Bank Miss Write	-3-3-5-5	-3-3-4-4	-3-3-4-4	(2)
Page Miss and Bank Hit Write	-3-5-5-5	-3-4-4-4	-3-3-4-4	(3)
For a pipelined burst transfer immediately following a write:				
Page Hit Read	-15-5-5-5	-11-4-4-4	-11-4-4-4	
Page Hit Write	-6-3-5-5	-6-3-4-4	-5-3-4-4	
Page Miss and Bank Miss Read	-17-5-5-5	-14-4-4-4	-13-4-4-4	(2)
Page Miss and Bank Hit Read	-19-5-5-5	-15-4-4-4	-13-4-4-4	(3)
Page Miss and Bank Miss Write	-6-6-5-5	-6-6-4-4	-6-5-4-4	(2)
Page Miss and Bank Hit Write	-6-9-5-5	-6-7-4-4	-6-5-4-4	(3)

**Notes:** See Table 4-2 for notes.

Table 4-5 contains a summary of recommended general case aggressive EDO DRAM timing, the required control register settings, and the resulting performance of the memory controller.

Table 4-5. EDO DRAM Conservative Timing Summary (1)

Transfer	70ns EDO Conservative	60ns EDO Conservative	50ns EDO Conservative	Note
Memory Timing Register 1	32	31	0C	
Memory Timing Register 2	0A	06	05	
Bridge Chipset Options 3	0C	0C	0C	
Initial Read Burst	12-4-4-4	11-3-3-3	10-3-3-3	(2)
Initial Write Burst	5-4-4-4	5-4-3-3	5-3-3-3	
For a pipelined burst transfer immediately following a read:				
Page Hit Read	-5-4-4-4	-5-3-3-3	-5-3-3-3	(2)
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	-7-3-3-3	(1,3)
Page Miss & Bank Hit Read	-11-3-3-3	-9-3-3-3	-7-3-3-3	(1,4)
Page Miss & Bank Miss Write	-3-3-4-4	-3-3-3-3	-3-3-3-3	(3)
Page Miss & Bank Hit Write	-3-5-4-4	-3-4-3-3	-3-3-3-3	(4)
For a pipelined burst transfer immediately following a write:				
Page Hit Read	-12-4-4-4	-9-3-3-3	-9-3-3-3	(2)
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	-11-3-3-3	(2,3)
Page Miss & Bank Hit Read	-17-4-4-4	-13-3-3-3	-11-3-3-3	(2,4)
Page Miss & Bank Miss Write	-6-6-4-4	-5-5-3-3	-5-4-3-3	(3)
Page Miss & Bank Hit Write	-6-8-4-4	-5-6-3-3	-5-4-3-3	(4)

**Notes:** See Table 4-2 for notes.

#### 4.2.1.12 Page Hit and Page Miss

PowerPC CPU bus memory transfers have the following characteristic behavior. When a CPU issues a memory access followed immediately by another memory access, the second access is typically from the same page of memory. On the other hand, if the second memory access does not immediately follow the first one (so that the CPU bus goes idle) then the second memory access is typically a page miss. Thus the majority of memory accesses following a bus idle condition are page misses. 660 memory performance is optimized by assuming that a CPU to memory transfer from bus idle will be a page miss.

When neither the CPU or the PCI is accessing memory, the memory controller goes to the idle state, and all RAS# lines are precharged (deasserted). Deasserting the RAS# lines at idle begins to satisfy the minimum RAS# precharge time requirement. Assuming that the first access out of idle will be a page miss, this technique allows the memory controller to reduce the time required for the initial beat of the burst DRAM read or write access by three CPU clocks. If the initial access is a page hit, this technique results in an increase in access time of two CPU clocks. A net gain is realized whenever the system is experiencing more page misses from bus idle than page hits from bus idle.

For the first beat of pipelined transactions, the memory controller checks the MA[11:0] memory address for a page hit. If the address is within the same 8K memory page as the previous memory access it is a page hit, the row address currently latched into the DRAM is considered valid, and the bridge accesses the DRAM using CAS# cycles. On page misses, the bridge latches the new row address into the DRAMs before it accesses the DRAM using CAS# cycles.

On CPU to memory bursts, only the address of the first beat of the burst is checked for page hits, because the following three beats are always within the same memory page.

On PCI to memory bursts, the address of each data phase of the burst is checked for page hits.

#### **4.2.1.13 CPU to Memory Access Pipelining**

CPU to memory accesses are pipelined, with the result that during a series of back-to-back CPU to memory accesses, all transfers following the initial transfer are faster. The information from the address tenure of the subsequent transfers is processed by the bridge while the data tenure of the preceding transfer is still active.

Considering a series of CPU to memory read transfers using 60ns EDO DRAM, the initial burst requires 10-3-3-3 CPU clocks. If this transfer is followed immediately (back-to-back) by another CPU to memory transfer, the required cycle time is -5-3-3-3. As long as the transfers are back-to-back, they are pipelined, and can be retired at this pipelined rate.

#### **4.2.1.14 Extended Data Out (EDO) DRAM**

The 660 is designed to support hyper-page-mode DRAM, sometimes called extended data out (EDO) DRAM. Information about the operation of the 660 with EDO DRAM is distributed throughout this section.

### **4.3 System Memory Addressing**

#### **4.3.1 DRAM Logical Organization**

The DRAM system implemented by the 660 is logically arranged as shown in Figure 4-2. Each block shown in Figure 4-2 is a 9 bit DRAM composed of 8 data bits and 1 parity (or check) bit that is accessed whenever the 8 data bits are accessed. The RAS# (aka MRE#) lines strobe in the row address. The CAS# (aka MCE#) lines strobe in the column address. For a block to activate (from idle) for either a read or a write, both the RAS# and CAS# line to it must be activated in the proper sequence. After the initial access, the device can deliver data in fast page mode with only CAS# strobes.

The CAS# lines can be thought of as byte enables, and the RAS# lines as bank enables. The WE# signal goes to all (each of) the devices in the memory array. The OE# of each DRAM device is tied active. This signal is not required to be deasserted at any time, since the DRAMs only enable their output drivers when so instructed by the RAS#/CAS# protocol.

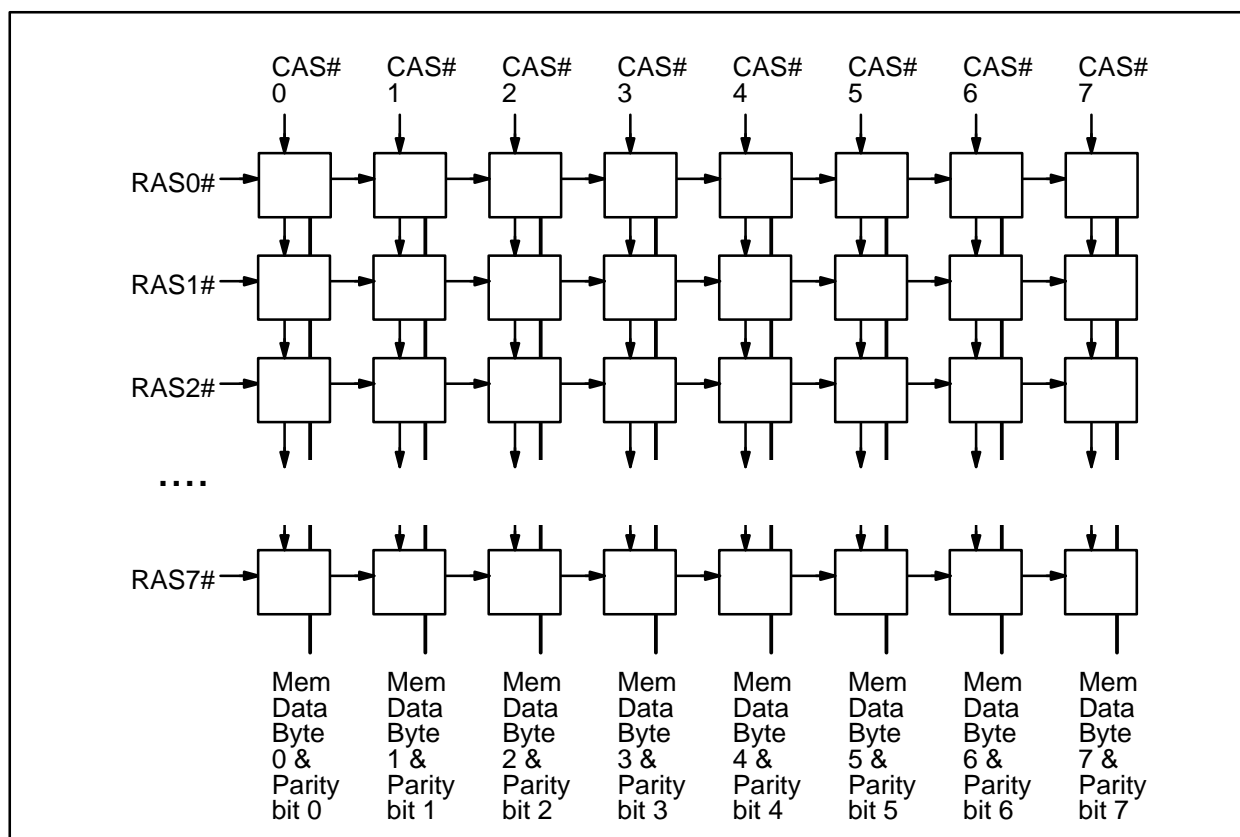


Figure 4-2. DRAM Logical Implementation

#### 4.3.1.1 SIMM Topologies

Table 4-6 shows the various memory module (SIMM) topologies that the 660 supports. Each memory bank can be populated with any supported SIMM.

Table 4-6. Supported SIMM Topologies

SIMM type	Size	Depth	Width	Banks	Addressing	Addressing Mode (1)
4-Byte Wide (72-pin)	4 Meg	1M	4 bytes	1 + 1 empty	10 x 10	2
	8 Meg	1M	4 bytes	2	10 x 10	2
	16 Meg	4M	4 bytes	1 + 1 empty	11 x 11	2
	32 Meg	4M	4 bytes	2	11 x 11	2
	64 Meg	16M	4 bytes	1 + 1 empty	12 x 12	3
8-Byte Wide (168-pin)	8 Meg	1M	8 bytes	1	10 x 10	2
	16 Meg	2M	8 bytes	1	11 x 10	2
	32 Meg	4M	8 bytes	1	11 x 11 or 12 x 10	2
	64 Meg	8M	8 bytes	1	12 x 11	3
	128 Meg	16M	8 bytes	1	12 x 12	3

Note:

1. See BCR(A4 to A7) in Section 4.3.1.6

The 660 supports the 168-pin 8-byte SIMMs shown in Table 4-6, which are each arranged as a single bank of 8-byte-wide DRAM. Each SIMM requires a single RAS# line. These SIMMs do not have to be installed in pairs.

The 660 also supports the 72-pin 4-byte SIMMs shown in Table 4-6, which are each arranged as two banks of 4-byte-wide DRAM, only one bank of which may be accessed at a given time. Each bank requires a RAS# line, and each bank is addressed by the same

address lines. These SIMMs must be installed in pairs (of identical devices), since it is necessary to use two (72-pin) 4-byte SIMMs to construct an 8-byte-wide memory array. Since each 72-pin 4-byte SIMM consists of two banks, this pair of SIMMs also requires two RAS# lines. The 660 addresses a given SIMM based on the value of the associated memory bank addressing mode BCR.

#### 4.3.1.2 Row and Column Address Generation

The 660 formats the row and column addresses presented to the DRAM based on the organization of the DRAM. In memory bank addressing mode 2, the bridge is configured to address devices that require 12x10, 11x11, 11x10, or 10x10 bit (row x column) addressing. In memory bank addressing mode 3, the bridge is configured to address devices that require 12x12 or 12x11 bit (row x column) addressing (no other addressing modes are currently available).

Table 4-7 and Table 4-8 show which CPU address bits are driven onto the memory address bus during CPU to memory transfers. Table 4-9 and Table 4-10 show which PCI\_AD address bits are driven onto the memory address bus during PCI to memory transfers. These address line assignments are not affected by the endian mode of the system. The addressing mode is selected using the memory bank addressing mode BCRs (see Section 4.3.1.6). The addressing mode of each bank of memory is individually configurable.

Table 4-7. Row Addressing (CPU Addressing)

Memory Bank Addressing Mode	Addressing Mode BCR	MA 11	MA 10	MA 9	MA 8	MA 7	MA 6	MA 5	MA 4	MA 3	MA 2	MA 1	MA 0
12x10, 11x10, 10x10, 11x11	010 (Mode 2)	A 7	A 8	A 9	A 10	A 11	A 12	A 13	A 14	A 15	A 16	A 17	A 18
12x12, 12x11	011 (Mode 3)	A 7	A 8	A 9	A 10	A 11	A 12	A 13	A 14	A 15	A 16	A 17	A 18

Table 4-8. Column Addressing (CPU Addressing)

Memory Bank Addressing Mode	Addressing Mode BCR	MA 11	MA 10	MA 9	MA 8	MA 7	MA 6	MA 5	MA 4	MA 3	MA 2	MA 1	MA 0
12x10, 11x10, 10x10, 11x11	010	A 5	A 7	A 19	A 20	A 21	A 22	A 23	A 24	A 25	A 26	A 27	A 28
12x12, 12x11	011	A 5	A 6	A 19	A 20	A 21	A 22	A 23	A 24	A 25	A 26	A 27	A 28

Table 4-9. Row Addressing (PCI Addressing)

Memory Bank Addressing Mode	Addressing Mode BCR	MA 11	MA 10	MA 9	MA 8	MA 7	MA 6	MA 5	MA 4	MA 3	MA 2	MA 1	MA 0
12x10, 11x10, 10x10, 11x11	010	AD 24	AD 23	AD 22	AD 21	AD 20	AD 19	AD 18	AD 17	AD 16	AD 15	AD 14	AD 13
12x12, 12x11	011	AD 24	AD 23	AD 22	AD 21	AD 20	AD 19	AD 18	AD 17	AD 16	AD 15	AD 14	AD 13

Table 4-10. Column Addressing (PCI Addressing)

Memory Bank Addressing Mode	Addressing Mode BCR	MA 11	MA 10	MA 9	MA 8	MA 7	MA 6	MA 5	MA 4	MA 3	MA 2	MA 1	MA 0
12x10, 11x10, 10x10, 11x11	010	AD 26	AD 24	AD 12	AD 11	AD 10	AD 9	AD 8	AD 7	AD 6	AD 5	AD 4	AD 3
12x12, 12x11	011	AD 26	AD 25	AD 12	AD 11	AD 10	AD 9	AD 8	AD 7	AD 6	AD 5	AD 4	AD 3

In the case of 10x10 addressing, MA[9:0] are connected to the DRAM modules. In the case of 11x10 or 11x11, connect MA[10:0], and in the case of 12x11 or 12x12, connect MA[11:0] to the DRAM modules.

#### 4.3.1.3 DRAM Pages

The 660 uses an 8K page size for DRAM page-mode determination.

#### 4.3.1.4 Supported Transfer Sizes and Alignments

The 660 supports all CPU to memory transfer sizes and alignments that do not cross an 8-byte boundary.

#### 4.3.1.5 Unpopulated Memory Locations

Physical memory does not occupy the entire address space assigned to system memory in the memory map. While the Memory Select Error Enable bit = 0:

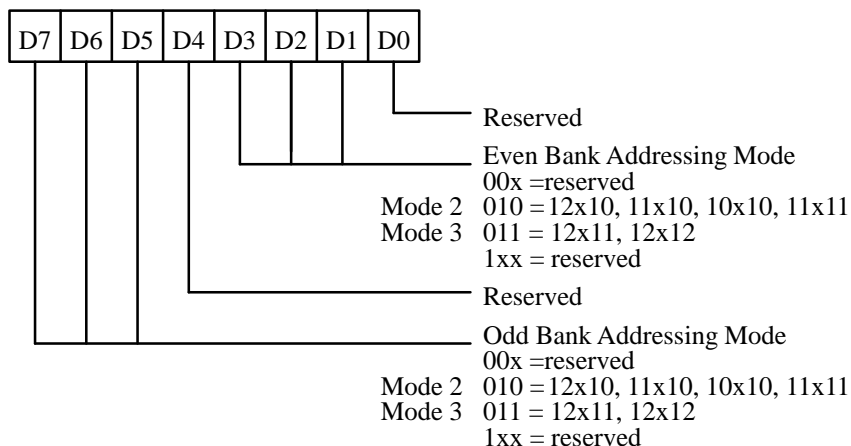
- When the CPU reads an unpopulated location, the bridge returns all-ones, and completes the transfer normally.
- When the CPU writes to an unpopulated location, the bridge signals normal transfer completion to the CPU, but does not write the data to memory.

The memory select error bit in the error status 1 register (index C1h) is set in both cases. Gaps are not allowed in the DRAM memory space, but empty (size=0) memory banks are allowed. While the Memory Select Error bit =1, a read or write to an unpopulated memory location is reported to the initiator as an error.

#### 4.3.1.6 Memory Bank Addressing Mode BCRs

Index	A4 to A7	Read/Write	Reset to 44h (each BCR)
-------	----------	------------	-------------------------

This array of four 8-bit, read/write BCRs defines the format of the row and column addressing of each DRAM memory bank.



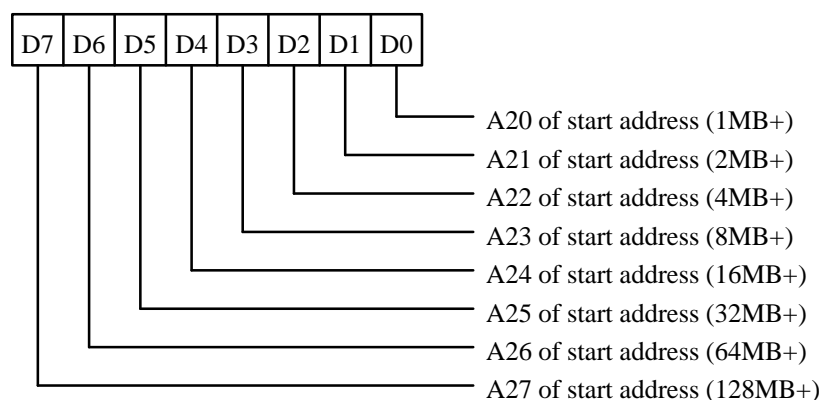
Register	Bits	Memory Bank	Bits	Memory Bank
A4h	3:0	0	7:4	1
A5h	3:0	2	7:4	3
A6h	3:0	4	7:4	5
A7h	3:0	6	7:4	7

### 4.3.1.7 Memory Bank Starting Address BCRs

Index	80 to 87h	Read/Write	Reset to 00h (each BCR)
-------	-----------	------------	-------------------------

This array of eight BCRs (along with the eight extended starting address registers) contains the starting address for each memory bank. Each pair of registers maps to the corresponding RAS# decode. For example, RAS[4]# corresponds to the BCRs at index 84h and 8Ch. The eight least-significant bits of the bank starting address are contained in the starting address register, and the most-significant bits come from the corresponding extended starting address register. The starting address of the bank is entered with the least significant 20 bits truncated. These BCRs must be programmed in conjunction with the ending address and extended ending address registers.

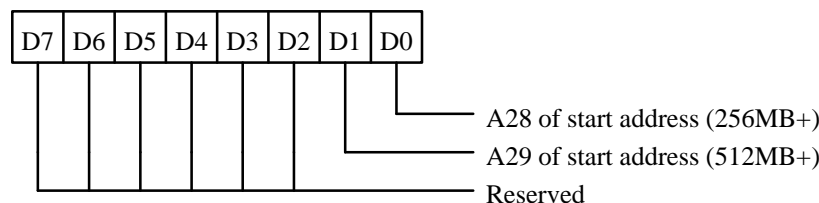
Program the banks in ascending order, such that (for  $n = 0$  to 6) the starting address of bank  $n+1$  is higher than the starting address of bank  $n$ . Each bank must be located in the 0 to 1G address range. See section 4.3.1.12.



### 4.3.1.8 Memory Bank Extended Starting Address BCRs

Index	88 to 8F	Read/Write	Reset to 00h (each BCR)
-------	----------	------------	-------------------------

This array of eight BCRs (along with the eight starting address registers) contains the starting address for each memory bank. These BCRs contain the most-significant address bits of the starting address of the corresponding bank.

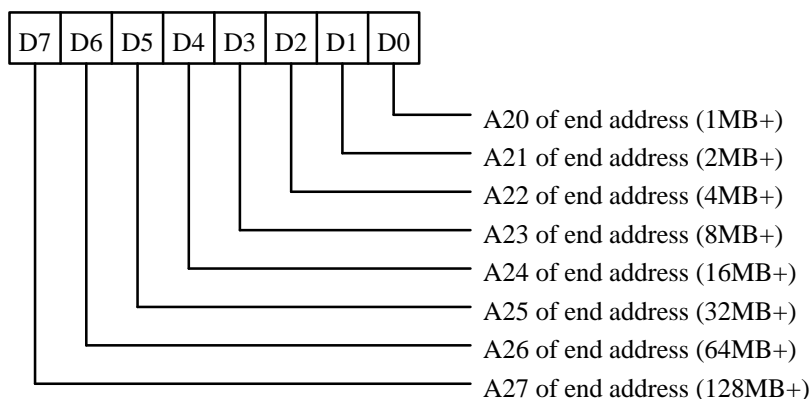


### 4.3.1.9 Memory Bank Ending Address BCRs

Index	90 to 97h	Read/Write	Reset to 00h (each BCR)
-------	-----------	------------	-------------------------

This array of eight BCRs (along with the eight extended starting address registers) contains the ending address for each memory bank. Each pair of registers maps to the corresponding RAS# decode. For example, RAS[4]# corresponds to the BCRs at index 94h and 9Ch. The eight least-significant bits of the bank ending address are contained in the ending address register, and the most-significant bits come from the corresponding extended ending address register. The ending address of the bank is entered as the address of the next high-

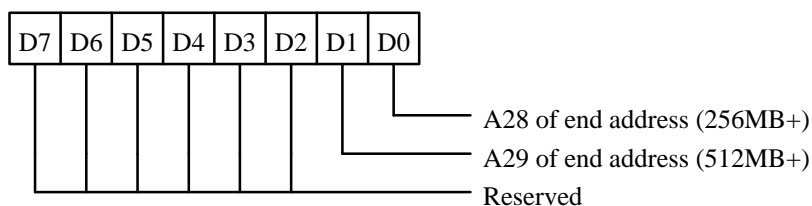
est memory location minus 1, with the least significant 20 bits truncated. Each bank must be located in the 0 to 1G address range. These BCRs must be programmed in conjunction with the ending address and extended ending address registers. See section 4.3.1.12.



#### 4.3.1.10 Memory Bank Extended Ending Address BCR

Index	98 to 9F	Read/Write	Reset to 00 (each BCR)
-------	----------	------------	------------------------

This array of eight 8-bit, read/write registers (along with the eight ending address registers) contains the ending address for each memory bank. These BCRs contain the most-significant address bits of the ending address of its bank.



#### 4.3.1.11 Memory Bank Enable BCR

Index	A0	Read/Write	Reset to 00h
-------	----	------------	--------------

This BCR contains a control enable for each bank of memory. Each bank of memory must be enabled for proper refreshing. For each bit, a 0 disables that bank of memory and a 1 enables it.

This register must be programmed in conjunction with the starting address and ending address registers. If a bank is disabled by this register, the corresponding starting and ending address register entries become don't cares.

#### 4.3.1.12 Memory Bank Configuration Example

In the example memory bank configuration shown in Figure 4-3, the eight memory banks are populated by different size and organization devices. For convenience, this example shows the bridge configured to address each memory bank in order with no gaps in the populated address range but this is not required. Any bank can be placed in any (1MB aligned) non-populated address range from 0 to 1G.



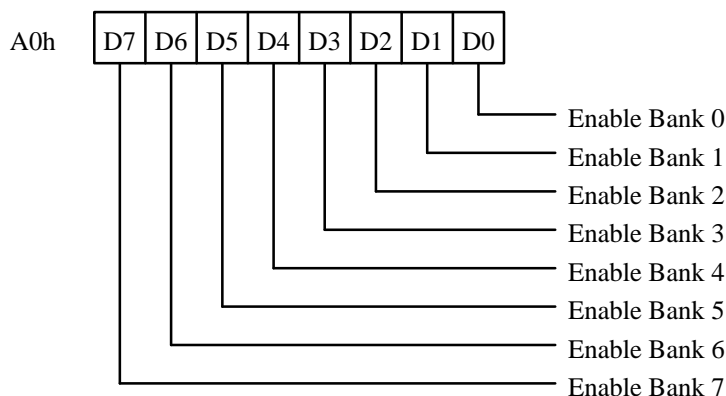


Table 4-11. Example Memory Bank Addressing Mode Configuration

Bank	SIMM Type	SIMM Depth	SIMMs Per Bank	SIMM Bank Topology	SIMM Size	Row x Col	BCR ()	Bits ()	Mode
0	8-Byte	1M	1	8B x 1M x 1 bank	8M	10 x 10	A4	3:1	2
1	4-Byte	4M	2	4B x 4M x 2 bank	32M	11 x 11	A4	7:5	2
2	4-Byte	4M	2	4B x 4M x 2 bank	32M	11 x 11	A5	3:1	2
3	8-Byte	8M	1	8B x 8M x 1 bank	64M	12 x 11	A5	7:5	3
4	none	—	—	—	0M	—	A6	3:1	—
5	4-Byte	4M	2	4B x 4M x 2 bank	32M	11 x 11	A6	7:5	2
6	8-Byte	16M	1	8B x 16M x 1 bank	128M	12 x 12	A7	3:1	3
7	8-Byte	4M	1	8B x 4M x 1 bank	32M	11 x 11 12 x 10	A7	7:5	2

#### 4.3.1.13 Memory Bank Enable BCR

Program indexed BCR A0h (memory bank enable) to EFh to enable all banks except # 4.

#### 4.3.1.14 Memory Bank Addressing Mode

As shown in Table 4-11, memory bank 0 (connected to RAS0#) contains an 8-byte x 1M SIMM (8M) with one bank (one RAS# line). It is addressed with 10 row and 10 column bits. Program bits 3:1 of indexed BCR A4h with 010b (mode 2).

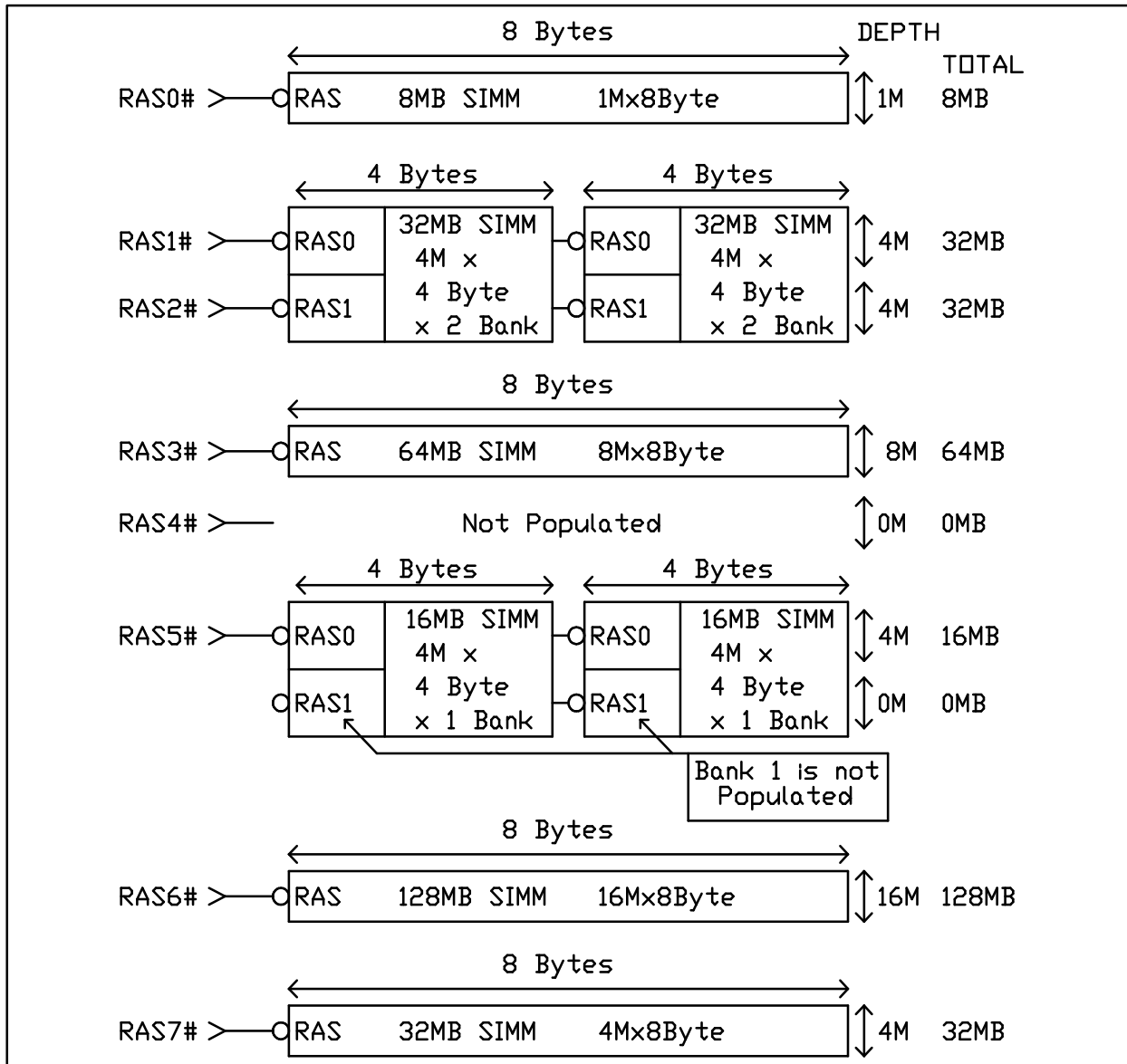


Figure 4-3. Example Memory Bank Configuration

#### 4.3.1.15 Starting and Ending Addresses

As shown in Table 4-12, program indexed BCR 80h with 00h to configure address bits 27:20 of the bank 0 starting address. Program indexed BCR 88h with 00h to configure address bits 29:28. Also program indexed BCR 90h with 00h to configure address bits 27:20 of the bank 0 ending address, and program indexed BCR 98h with 07h to configure address bits 29:28.

The next two physical SIMM units are 4-byte x 4M x 2 bank (16M x 2 bank = 32M) SIMMs, used side by side to achieve 8-byte width. They form banks 1 and 2, each of which is 8-byte x 4M (32M). Note that bank 7 is also 32M, populated by an 8-byte x 8M (32M) SIMM. Using the same configuration, banks 1 and 2 could also be implemented using two 8-byte x 4M (32M) SIMMs. Bank 3 is configured to the same size as bank 1 plus bank 2 and is implemented using a single 8-byte x 8M (64M) SIMM.

Bank 4 is not populated, so set memory bank enable BCR bit D4 to 0. The data in the starting, extended starting, ending, and extended ending address BCRs for bank 4 is ignored. Note that empty memory banks are allowed, but that gaps in the DRAM space are not allowed.

For proper operation, program the bridge to execute memory accesses at the speed of the slowest device. If ECC or parity is selected, all devices must support the capability. And if the bridge is programmed to utilize hyper-page mode, all devices must support extended-data out transfers.

**Table 4-12. Example Memory Bank Starting and Ending Address Configuration**

	SIMM	Starting	Extended		Base		Ending	Extended		Base	
Bank	Size	Address	BCR	Data	BCR	Data	Address	BCR	Data	BCR	Data
0	8M	0000 0000	88	00	80	00	007F FFFF	98	00	90	07
1	32M	0080 0000	89	00	81	08	027F FFFF	99	00	91	27
2	32M	0280 0000	8A	00	82	28	047F FFFF	9A	00	92	47
3	64M	0480 0000	8B	00	83	48	087F FFFF	9B	00	93	87
4	0M	Don't Care	8C	xx	84	xx	Don't Care	9C	xx	94	xx
5	32M	0880 0000	8D	00	85	88	0A7F FFFF	9D	00	95	A7
6	128M	0A80 0000	8E	00	86	A8	127F FFFF	9E	01	96	27
7	32M	1280 0000	8F	01	87	28	147F FFFF	9F	01	97	47

## 4.4 Error Checking and Correction

The 660 provides three levels of memory error checking—no checking, parity checking, and error checking and correction (ECC). If no memory checking is enabled, the system can be configured to use lower-cost, non-parity DRAM.

While the system is configured for parity checking, the 660 performs as follows:

- Uses odd parity checking
- Detects all single bit errors
- Allows full-speed memory accesses

While the system is configured for ECC, the 660 performs as follows:

- Uses an H-matrix and syndrome ECC protocol
- Uses the same memory devices and connectivity as for parity checking
- Detects and corrects all single-bit errors
- Detects all two-bit errors
- The first beat of CPU to memory reads requires one additional CPU clock cycle
- CPU to memory burst writes and 8-byte single-beat writes are full-speed
- CPU to memory single-beat writes of less than eight bytes are implemented as read-modify-write (RMW) cycles
- PCI to memory reads are full-speed
- PCI to memory writes are full-speed while data can be gathered into 8-byte groups before being written to memory. Single beat or ungatherable writes require a read-modify-write cycle

### 4.4.1 Memory Parity

While the 660 memory controller is configured for memory parity checking, the bridge implements an odd parity generation and checking protocol, generating parity on memory

writes and checking parity on memory reads. One parity bit is associated with each data byte and is accessed with it. When a parity error is detected during CPU to memory reads, the error is reported by means of TEA# or MCP#. When a parity error is detected during PCI to memory reads, the error is reported by means of PCI\_SERR#.

The 660 detects all single-bit parity errors, but may not detect multi-bit parity errors. Also, an even number of parity errors will not be detected. For example, an event which causes a parity error in bytes 0, 1, 2, 3, 4, and 5 will not be detected.

#### **4.4.1.1 ECC Overview**

While ECC is enabled, the 660 uses the ECC logic to detect and correct errors in the transmission and storage of system memory data. The bridge implements the ECC protocol using the same connectivity and memory devices as parity checking.

While neither parity or ECC is enabled, the bridge executes memory writes of less than eight bytes in response to busmaster requests to write less than eight bytes of data. The bridge always (whether parity, ECC or neither is enabled) reads data from memory in 8-byte groups, even if the busmaster is requesting less than eight bytes.

While parity is enabled, the bridge also executes memory writes of less than eight bytes in response to busmaster requests to write less than eight bytes of data, since writing a byte to memory also updates the associated parity bit. During memory writes, the bridge generates one parity bit for each byte of data and stores it with that byte of data. This parity bit is a function only of the data byte with which it is associated. During memory reads, the integrity of the data is parity checked by comparing each data byte with its associated parity bit.

However, when ECC is enabled the bridge reads from and writes to system memory only in 9-byte groups (as a 72-bit entity), even though the busmaster may be executing a less-than-8-byte read or write. There is one byte of ECC check byte information for eight bytes of data. During memory writes, the eight check bits are generated as a function of the 64 data bits as a whole, and the check bits are stored with the data bits as a 72-bit entity. During memory reads, all 72 bits are read, and the eight check bits are compared with the 64 data bits as a whole to determine the integrity of the entire 72-bit entity.

In ECC mode, single-bit errors are corrected. When a multi-bit error is detected during CPU to memory reads, the error is reported by means of TEA# or MCP#. When a multi-bit ECC error is detected during PCI to memory reads, the error is reported by means of PCI\_SERR#. Note that the DRTRY# function of the CPU is not used, even when the 660 is in ECC mode (the Bridge will not assert DRTRY#).

Only the data returned to the CPU (or PCI) is corrected for single-bit errors. The corrected data is not written into the DRAM location.

#### **4.4.1.2 ECC Data Flows**

While ECC is enabled, the 660 always reads eight data bytes and one check byte from memory during CPU and PCI reads. During busmaster 8-byte writes to memory, the bridge writes eight data bytes and one check byte to memory. When the busmaster writes less than eight data bytes to memory, it is possible for each check bit to change due to a write to any one of the eight data bytes. The Bridge then executes a read-modify-write (RMW) cycle—reading all eight data bytes from memory, modifying the appropriate bytes with the new data, recalculating all of the check bits, and writing the new data and check bits to memory.

#### 4.4.1.3 Memory Reads

Figure 4-4 shows a simplified data flow in a 660 system during CPU to memory read transfers. Figure 4-5 shows the simplified data flow during PCI to memory reads. The data and check bits flow from system memory into the bridge where the checking logic combines the data with the check bits to generate the syndrome. If there is a single-bit error in the data, the correction logic corrects the data and supplies it to the requesting agent. If there is a multiple-bit error in the data, the bridge signals an error to the requesting agent.

Note that the structure of the bridge as shown in Figure 4-4 through Figure 4-8 is considerably simplified and does not show the posted write buffers and other internal details.

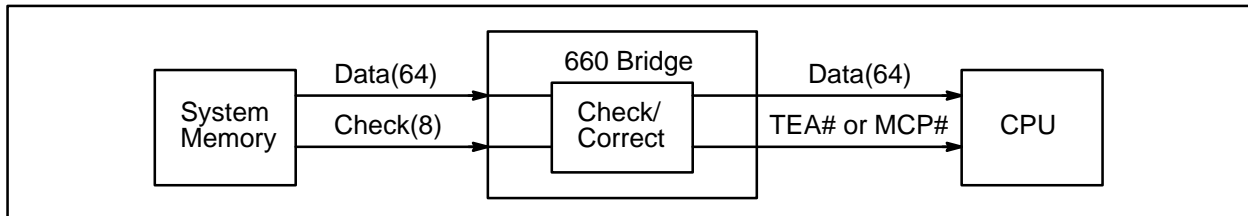


Figure 4-4. CPU Read Data Flow

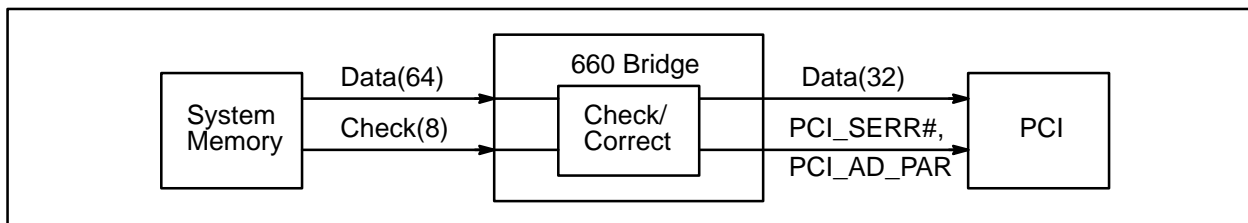


Figure 4-5. PCI Read Data Flow

#### 4.4.1.4 Eight-Byte Writes

Figure 4-6 shows the simplified data flow in a 660 system during 8-byte CPU to memory writes. The data flows from the CPU into the bridge and out onto the memory data bus. The bridge generates the check bits based on the eight bytes of data, and stores them in memory with the data.

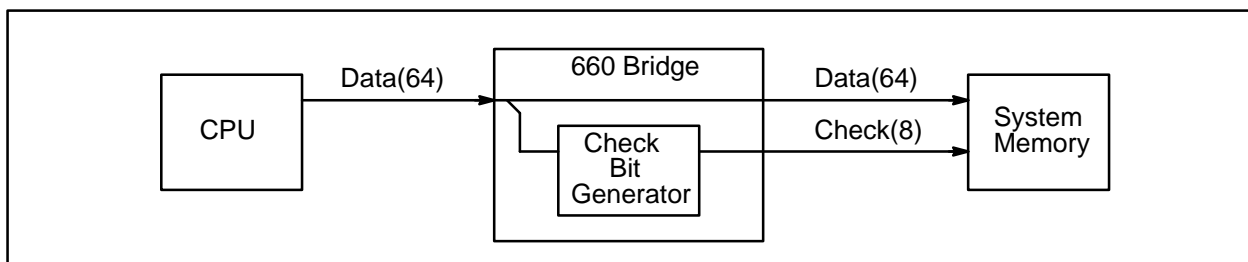
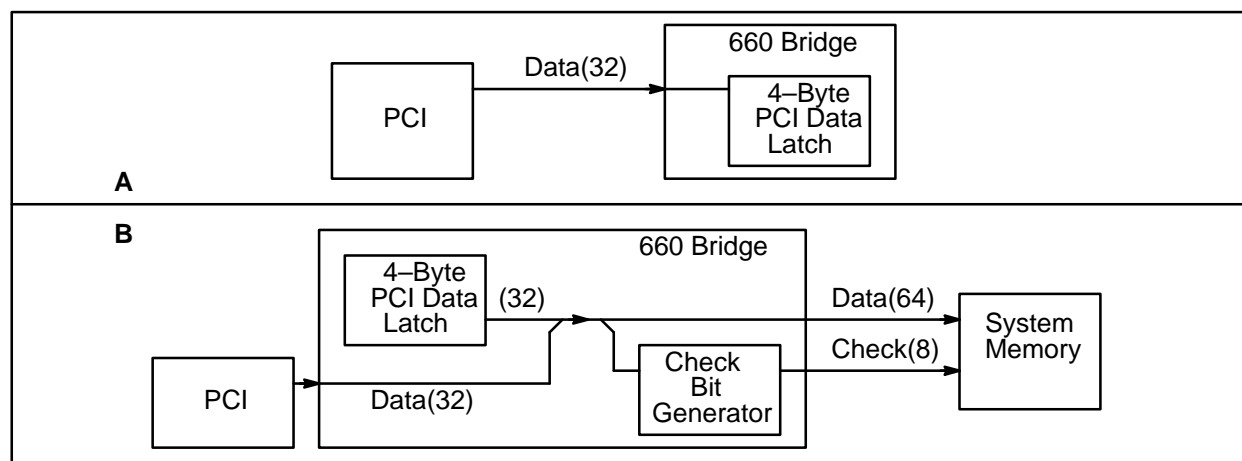
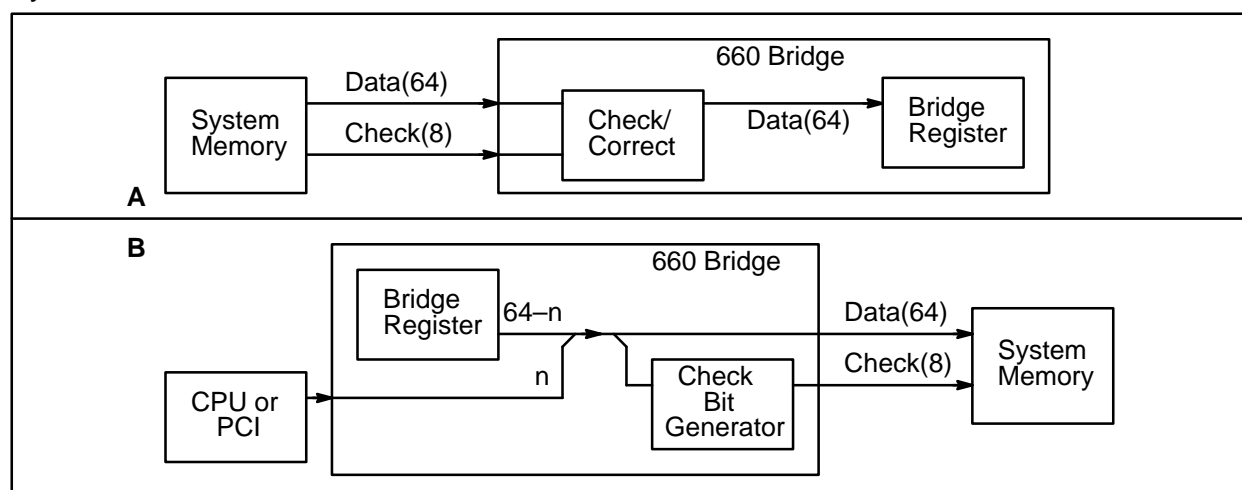


Figure 4-6. CPU 8-Byte Write Data Flow

Figure 4-7 shows the simplified data flow in a 660 based system during gathered PCI to memory 8-byte writes. During the first of the two gathered data phases (A), the data flows from the PCI bus into a 4-byte hold latch in the bridge. On the next data phase (B), the next 4-bytes of PCI data flows into the bridge, where it is combined with the data from the previous data phase. The 8-byte data then flows onto the memory data bus. The bridge generates the check bits based on the 8-byte data, and stores them in memory with the data.

**Figure 4-7. PCI 8-Byte Write Data Flow**

Note that if either or both of the two gathered PCI data phases is a write of less than 4 bytes, the two data phases will still be gathered, and then the bridge will execute a RMW cycle, filling in the unwritten bytes (in the group of 8 bytes) with data from those locations in memory. The same write case while ECC is disabled does not cause a RMW cycle; the bridge merely writes only the indicated bytes, leaving the write enables of the unaccessed bytes deasserted.

**Figure 4-8. PCI or CPU Read-Modify-Write Data Flow**

#### 4.4.1.5 Less-Than Eight-Byte Writes

Figure 4-8 shows the simplified data flow during a CPU or PCI busmaster to memory write of less than eight bytes, during which the bridge executes a RMW cycle. In Figure 4-8(A), the bridge reads in the data and check bits from the addressed memory locations and places this data (corrected as necessary) in a register. In Figure 4-8(B), this data is modified by the bridge, which replaces the appropriate memory data bytes with write data from the busmaster. The bridge then recomputes all of the check bits and writes the new data and check bits to memory.

#### 4.4.1.6 Memory Performance In ECC Mode

Enabling ECC mode on the 660 affects memory performance in various ways, depending on the transaction type. The effect is the same for both EDO and page mode DRAMs.

**4.4.1.7 CPU to Memory Read in ECC Mode**

ECC mode adds one CPU\_CLK to single-beat CPU to memory read transfers and to the first beat of CPU to memory burst read transfers. The other beats of the burst are unaffected. During the extra CPU\_CLK, the 663 holds the data while checking (and if necessary, correcting) it.

The 660 does not use the DRTRY# function, even while in ECC mode. In no-DRTRY# mode, during memory reads the 604 uses the data bus data internally as soon as it samples TA# active. The 660 supports this mode by presenting correct(ed) data to the before driving TA# valid. The Bridge does not speculatively present data to the CPU (using TA#) and then assert DRTRY# if there is a data error.

By allowing the 604 to run in no-DRTRY# mode, the 660 enables the 604 to use data from the L2 cache at the full speed of the L2 cache, without requiring the 604 to insert an additional (internal to the 604) one CPU clock delay on reads.

In DRTRY# mode, during memory reads the 604 holds the data internally for one CPU clock after sampling TA# active. This delay is inserted by the 604 to allow DRTRY# to be sampled before the data is used. Thus during CPU to memory reads in ECC mode while a 604 is in DRTRY# mode, the 660 inserts a one CPU clock delay to check/correct the data, and then the 604 adds a one CPU clock delay to check for DRTRY#. Thus two CPU clocks are added to single-beat CPU to memory read transfers and to the first beat of CPU to memory burst read transfers.

**4.4.1.8 CPU to Memory Write in ECC Mode**

ECC mode adds no additional clock cycles to 8-byte CPU to memory write transfers. Note that all CPU bursts are composed of 8-byte beats. CPU to memory writes of less than eight bytes are handled as RMW cycles, which usually require four additional CPU clocks as compared to 8-byte writes.

**4.4.1.9 PCI to Memory Read in ECC Mode**

ECC mode adds no additional clock cycles to PCI to memory read transactions.

**4.4.1.10 PCI to Memory Write in ECC Mode**

ECC mode has a complex effect on PCI to memory writes. During PCI to memory writes, the bridge attempts to gather adjacent 4-byte data phases into 8-byte memory writes. In response to conditions during a given data phase, the bridge either gathers, writes 8 bytes, or read-modify-writes, as shown in Table 4-13. Gather and 8-byte write operations incur no performance penalties, but RMW cycles add from two to four (usually three) PCI clock cycles to the transaction time. The consequences of ECC mode delays on PCI to memory write bursts are minor and are best understood from a few examples. The following examples assume page hits and no snoop hits.

Table 4-13. Bridge Response to Various PCI Write Data Phases

Bridge Operation	Conditions	Description of Operation
<b>Gather</b>	This data phase is not the last data phase, and This is a 4-byte transfer (BE[3:0]#=0000), and This data phase is to the lower 4-byte word.	The bridge latches the four bytes from this data phase into the low four bytes of a hold register.
<b>Eight-Byte Write</b>	The previous data phase caused a gather, and This is a 4-byte transfer, and This data phase is to the upper 4-byte word.	The bridge combines the (high) four bytes from this data phase with the (low) four bytes from the previous data phase, and writes all eight bytes to memory.
<b>Read-Modify-Write</b>	This is a single phase transaction, or This is the first data phase and is to the upper 4-byte word, or This is the last data phase and is to the lower 4-byte word, or This is a less-than-4-byte transfer.	The bridge Reads eight bytes from memory, modifies the data by replacing the appropriate four bytes with the data from this data phase, and then writes all eight bytes to memory.

Best case (see Table 4-14) is a burst starting on a low word (memory address is 0 mod 8), composed of an even number of data phases, in which all data phases transfer four bytes of data. Notice that the first data phase is gathered and the second data phase causes an 8-byte memory write. The following pairs of data beats follow the same pattern. No page misses, snoop hits, or data beats of less than 4 bytes are encountered. This case adds no PCI clock cycles to the best-case transaction time.

Table 4-14. Bridge Response to Best Case PCI Write Burst

Data Phase	Word	Bridge Operation	Special Conditions	Performance Impact
n (last)	High	8-byte Write	None	None
n – 1	Low	Gather	None	None
...	...	...	...	...
4	High	8-byte Write	None	None
3	Low	Gather	None	None
2	High	8-byte Write	None	None
1 (first)	Low	Setup + Gather	None	None

Table 4-15 shows a case where the first data phase is to a high word (memory address is 4 mod 8), and is composed of an odd number of data phases in which all data phases transfer four bytes of data. Here the total effect is to add three PCI clocks to the transaction time, which is shown during the first data phase in Table 4-15.

Table 4-15. Bridge Response to Case 2 PCI Write Burst

Data Phase	Word	Bridge Operation	Special Conditions	Performance Impact
n (last)	High	8-byte Write	None	None
n – 1	Low	Gather	None	None
...	...	...	...	...
3	High	8-byte Write	None	None
2	Low	Gather	None	None
1 (first)	High	Setup + RMW	None	Add 3 PCI clocks



Table 4-16. Bridge Response to Various PCI Write Bursts

Row	Data Phase	Word	Bridge Operation	Special Conditions	Performance Impact
12	n (last)	Low	RMW	None	Add 3 PCI clocks
11	n – 1	High	8-byte Write	None	None
10	n – 2	Low	Gather	None	None
9	...	...	...	...	None
8	4	High	RMW	None	Add 3 PCI clocks
7	3	Low	RMW	Less than 4-byte transfer	Add 3 PCI clocks
6	...	...	...	...	None
5	10	High	RMW	Less than 4-byte transfer	Add 3 PCI clocks
4	9	Low	Gather	None	None
3	...	...	...	...	None
2	2	High	8-byte Write	None	None
1	1 (first)	Low	Setup + Gather	None	None

Table 4-16 shows the effect of several conditions on the transaction time. Rows 1 through 6 show the effects of a less-than-4-byte transfer at the high word location that occurs during a burst. The term *None* in the column titled *Performance Impact* in row 6 indicates that there are no residual performance penalties due to the events of rows 1 through 5.

Rows 7 through 9 show the effect of a less-than-4-byte transfer at the low word location that occurs during a burst. The term *None* in the column titled *Performance Impact* in row 9 indicates that there are no residual performance penalties due to the events of rows 7 and 8.

Rows 10 through 12 show the effect of a burst that ends at a low word location. The total performance impact from this burst is to add 12 PCI clocks to the transaction time.

Note that the performance penalty for single data phase PCI writes is three additional PCI clocks, whether the destination is the high word or the low word.

#### 4.4.1.11 Check Bit Calculation

The 660 generates the check bits based on Table 4-17 (which is shown using little-endian bit numbering).

Table 4-17. Check Bit Calculation

CB (x)	Data Bits. CB(x) = XOR of Data Bits (0 is LSb)
0	0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,33,34,35,39,41,42,43,47,49,50,51,55,57,58,59,63
1	8,9,10,11,12,13,14,15,24,25,26,27,28,29,30,31,32,34,35,38,40,42,43,46,48,50,51,54,56,58,59,62
2	16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,35,37,40,41,43,45,48,49,51,53,56,57,59,61
3	0,1,2,3,4,5,6,7,16,17,18,19,20,21,22,23,32,33,34,36,40,41,42,44,48,49,50,52,56,57,58,60
4	1,2,3,7,9,10,11,15,17,18,19,23,25,26,27,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47
5	0,2,3,6,8,10,11,14,16,18,19,22,24,26,27,30,40,41,42,43,44,45,46,47,56,57,58,59,60,61,62,63
6	0,1,3,5,8,9,11,13,16,17,19,21,24,25,27,29,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63
7	0,1,2,4,8,9,10,12,16,17,18,20,24,25,26,28,32,33,34,35,36,37,38,39,48,49,50,51,52,53,54,55

**4.4.1.12 Syndrome Decode**

The syndrome consists of an 8-bit quantity which is generated by the 660 as it is comparing the 8 data bytes to the check byte. This syndrome contains the results of the comparison, as shown in Table 4-18, where:

- ne** means that no error has been detected.
- cbx** means that check bit x is inverted (a single-bit error).
- dx** means that data bit x is inverted (a single-bit error).
- blank** means that a multiple bit error has occurred.

Based on the information in the syndrome, the 660 corrects all single-bit errors and signals an error to the requesting agent on multiple-bit errors.

**Table 4-18. Syndrome Decode**

	S0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	S1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	S2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	S3	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
S7 S6 S5 S4																	
0 0 0 0	ne	cb0	cb1		cb2				cb3								
0 0 0 1	cb4			d8			d24			d0			d16				
0 0 1 0	cb5			d9			d25			d1			d17				
0 0 1 1		d40	d41		d42		d44	d43		d45		d46	d47				
0 1 0 0	cb6			d10			d26			d2			d18				
0 1 0 1																	
0 1 1 0		d56	d57		d58		d60	d59		d61		d62	d63				
0 1 1 1				d12			d28			d4			d20				
1 0 0 0	cb7			d11			d27			d3			d19				
1 0 0 1		d32	d33		d34		d36	d35		d37		d38	d39				
1 0 1 0																	
1 0 1 1				d13			d29			d5			d21				
1 1 0 0		d48	d49		d50		d52	d51		d53		d54	d55				
1 1 0 1				d14			d30			d6			d22				
1 1 1 0				d15			d31			d7			d23				
1 1 1 1																	

**4.5 DRAM Refresh**

The memory controller provides DRAM refresh logic for system memory. The memory controller supports CAS#-before-RAS# refresh only, which provides lower power consumption and lower noise generation than RAS#-only refresh. In this refresh mode, MA[11:0] are not required. Refresh of the odd banks of memory is staggered from the refresh of the even banks of memory to further reduce noise (see Figure 4-9).

During refresh,

- WE[1:0] are driven high.
- MEM\_DATA[63:0] are tri-stated.
- MA[11:0] continue to be driven to their previous state.

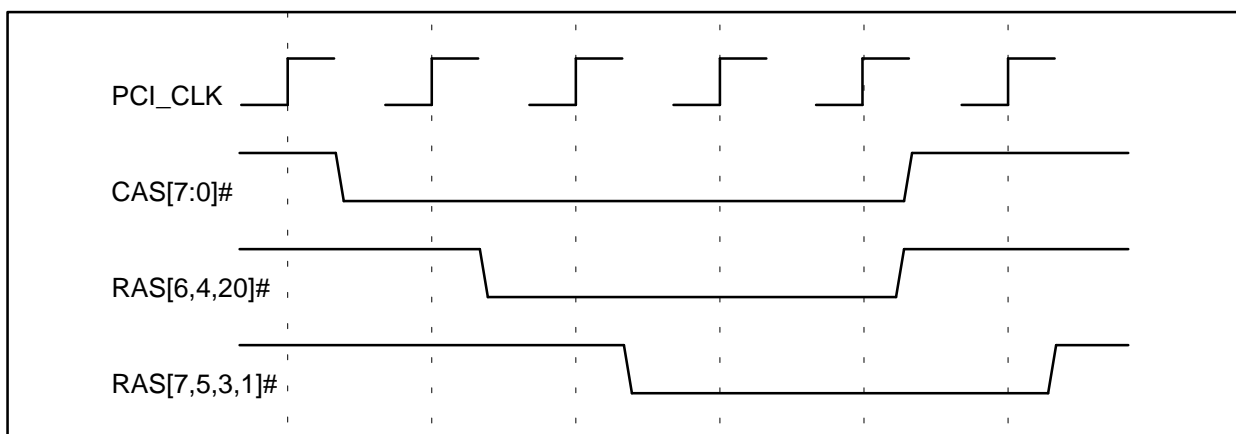


Figure 4-9. DRAM Refresh Timing Diagram

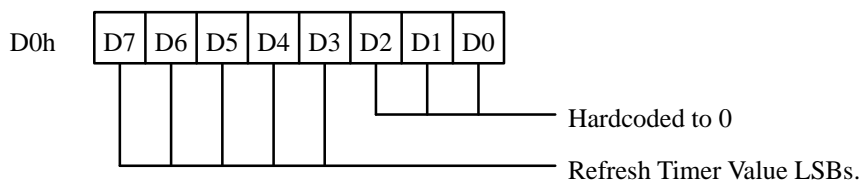
Refresh requests are generated internally by dividing down the PCI\_CLK. The divisor value is programmed in the refresh timer divisor register. This register is initialized to 504, a value that provides a refresh rate of 15.1us when PCI\_CLK rate is 33MHz. For other PCI\_CLK frequencies, the refresh rate register must be properly configured before accessing system memory.

Refresh continues to occur even if CPU\_CLK is stopped.

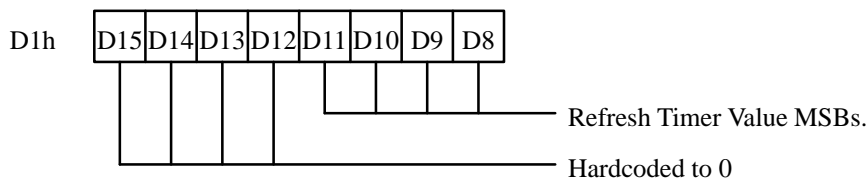
#### 4.5.1 Refresh Timer Divisor Register

Index	D0 to D1	Read/Write	Reset to F8 (D0) and 01 (D1)
-------	----------	------------	------------------------------

The refresh timer register is a 16-bit BCR that determines the memory refresh rate. Typical refresh rates are 15.1 to 15.5 microseconds. If all DRAM in the system supports extended (slow) refresh, the refresh rate can be slower. The refresh timer is clocked by the PCI clock input to the 664 Controller. The reset value of 01F8h provides a refresh rate of 15.1 microseconds while the PCI clock is 33MHz. (01F8h equals 504 times 30ns equals 15.12us.) Bits 3–11 of the timer allow timer values from 8 to 4096.



Bits 7:3 Refresh timer (7:3) : These are the five least-significant bits of the refresh timer value.



Bits 11:8 Refresh timer (11:8) : These are the four most-significant bits of the refresh timer value.

## 4.6 Atomic Memory Transfers

The 660 supports atomic memory transfers by supporting the CPU reservation protocol and the PCI lock protocol.

### 4.6.1 Memory Locks and Reservations

The 660 supports the lwarx and stwcx atomic memory update protocol by broadcasting snoop cycles to the CPU bus during PCI to memory transactions. The bridge does not otherwise take any action, nor does it enforce an external locking protocol for CPU busmasters. See Section 4.6.2.

PCI busmasters can lock and unlock a 32-byte block of system memory in compliance with the PCI specification. This block can be located anywhere within the populated system memory space, aligned on a 32-byte boundary. Only a single lock may be in existence at any given time. The bridge does not implement complete bus locking. See Section 4.6.3.

### 4.6.2 CPU Reservation

The CPU indicates a reservation request by executing a memory read atomic (TT[0:3]=1101). If there is no PCI lock on the addressed block of memory, the bridge allows the transfer, but takes no other action. If there is a PCI lock on that block, the bridge terminates the CPU transfer with ARTRY#, does not access the memory location, and takes no other action.

The CPU removes a reservation by executing a memory read with intent to modify atomic (TT[0:3]=1111) or a memory write with flush atomic (TT[0:3]=1001). The bridge treats these accesses as a normal memory transfers.

### 4.6.3 PCI Lock

The bridge responds to the PCI lock request protocol in compliance with the PCI specification. If an agent requests a lock, and no PCI lock is in effect, the lock is granted. Once a PCI lock is granted, no other PCI locks are granted until the current lock is released.

The 660 prevents CPU busmasters and other PCI busmasters from reading or writing within a block of memory which is locked by a PCI busmaster. CPU busmaster accesses to a locked block are retried with ARTRY#. PCI busmaster accesses to a locked block are retried with the PCI bus retry protocol. PCI and CPU busmaster accesses to other areas of system memory are unrestricted.

PCI to memory transactions cause the bridge to broadcast a snoop cycle to the CPU bus. When a PCI agent is granted a memory block lock, the bridge broadcasts a write with flush (TT[0:3]=0001) cycle on the CPU bus, which causes the L1 and L2 caches to invalidate that sector (if there is an address match). This insures that there will not be a cache hit during a CPU bus accesses to a memory block which is locked by a PCI agent.

### 4.6.4 PCI Lock Release

The bridge responds to the PCI lock release protocol in compliance with the PCI Specification. If an agent releases the lock that it owned, the bridge releases the lock. The bridge generates a normal snoop cycle on the CPU bus.

## 4.7 DRAM Module Loading Considerations

The 660 directly drives up to eight 168 pin DRAM modules, which typically exhibit an input capacitance of less than 20pf. Table 4-19 shows some maximum input capacitance numbers that are typical of the various DRAM modules on the market. System designers may wish to buffer MA[11:0] (and perhaps WE#) if the system design requires the use of more than two banks of 72 pin SIMMs (more than one pair of 2-sided 72 pin SIMMs).

**Table 4-19. Typical DRAM Module Maximum Input Capacitance**

SIMM Type	SIMM Size	Maximum Input Capacitance	
		Address	WE#
72-pin	4 Meg	50pf	50pf
	8 Meg	90pf	95pf
	16 Meg	80pf	95pf
	32 Meg	160pf	190pf
168-pin	8M, 16M, 32M,128M	13pf	13pf
	64M	18pf	18pf

## 4.8 Related Bridge Control Registers

Information on these registers is contained in the *660 Bridge Manual*.

Bridge Control Register	Index	R/W	Bytes
Memory Parity Error Status	8000 0840	R	1
Single-Bit Error Counter	Index B8	R/W	1
Single-Bit Error Trigger Level	Index B9	R/W	1
Bridge Options 2	Index BB	R/W	1
Error Enable 1	Index C0	R/W	1
Error Status 1	Index C1	R/W	1
Single-Bit ECC Error Address	Indx CC – CF	R/W	4
Bridge Chip Set Options 3	Index D4	R/W	1

This image shows a full page of blank graph paper. The grid consists of small, equal-sized squares formed by thin black lines. There are 20 columns and 20 rows of squares, creating a total of 400 square units. The grid covers the entire area of the page, leaving no margins or other markings.This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

## **Section 5**

### **PCI Bus**

The MCM includes a 32-bit PCI bus interface that runs at frequencies up to 33MHz. The MCM PCI interface is a host PCI bridge that is compliant with the PCI Specification, revisions 2.0 and 2.1 (see Section 5.5) for the 3.3v and 5v signalling environments.

The MCM PCI bus interface is designed to run at one half of the CPU bus frequency.

PCI bus activity initiated by the CPU is discussed in section 3. This section describes PCI bus transactions initiated by a (non-660-bridge) PCI busmaster. PCI busmasters can initiate arbitrary length read and write bursts to system memory.

Whenever a PCI busmaster accesses system memory, the 660 broadcasts a snoop cycle onto the CPU bus. See the 660 User's Manual CPU Bus section for details.

The MCM supports ISA Master operations to system memory.

## 5.1 PCI Transaction Decoding

When a PCI busmaster initiates a transaction on the PCI bus, the transaction either misses all of the targets and is master aborted, or it is claimed by a PCI target. The target can be either MCM system memory (via the 660) or another PCI target.

### 5.1.1 PCI Transaction Decoding By Bus Command

Table 5-1 shows the response of the 660 and other agents to various PCI bus transactions initiated by a PCI busmaster other than the 660. The 660 ignores (No response) all PCI bus transactions except PCI memory read and write transactions, which it decodes as possible system memory accesses.

**Table 5-1. MCM Response to PCI\_C[3:0] Bus Commands**

C[3:0]	PCI Bus Command	Can a PCI busmaster Initiate this Transaction?	660 Response to the Transaction	Can Another PCI Target Claim the Transaction?
0000	Interrupt Acknowledge	No. Only the 660 is allowed to initiate.	None	Yes. The ISA bridge is intended to be the target.
0001	Special Cycle	Yes	None	Yes
0010	I/O Read	Yes	None	Yes
0011	I/O Write	Yes	None	Yes
0100	Reserved	No. Reserved	None	n/a
0101	Reserved	No. Reserved	None	n/a
0110	Memory Read	Yes	System memory read	Yes, if no address conflict.
0111	Memory Write	Yes	System memory write.	Yes, if no address conflict.
1000	Reserved	No. Reserved	None	n/a
1001	Reserved	No. Reserved	None	n/a
1010	Configuration Read	No. Only 660.	None	Yes
1011	Configuration Write	No. Only 660.	None	Yes
1100	Memory Read Multiple	Yes	System memory read	Yes, if no address conflict.
1101	Dual Address Cycle	Yes	None	Yes
1110	Memory Read Line	Yes	System memory read	Yes, if no address conflict.
1111	Memory Write and Invalidate	Yes	System memory write	Yes, if no address conflict.

PCI busmasters are not able to access the boot ROM or the BCRs in the 660.

PCI busmasters can not initiate transactions to the CPU bus. CPU bus devices can, however, initiate transfers to the PCI bus. Large amounts of data are typically moved between the PCI bus and memory by bursts initiated by a PCI busmaster.

See Section 8.1.3 and the PCI specification for information on PCI Interrupt Acknowledge transactions.

### 5.1.2 PCI Memory Transaction Decoding By Address Range

When a PCI busmaster transaction is decoded by bus command as a system memory read or write, the 660 checks the address range. Table 5-2 shows the address mapping of PCI busmaster memory accesses to system memory. This is the mapping that the 660 uses when it decodes the bus command to indicate a system memory access.



Table 5-2. Mapping of PCI Memory Space, Part 1

PCI Bus Address	Other Conditions	Target Cycle Decoded	Target Address	Notes
0 to 2G	IGN_PCI_AD31 Deasserted	Not Decoded	N/A	No Response.
	IGN_PCI_AD31 Asserted	System Memory *	0 to 2G	Snooped by caches.
2G to 4G		System Memory *	0 to 2G	Snooped by caches.

\* Memory does not occupy this entire address space. Accesses to unoccupied space are not decoded.

Unless the IGN\_PCI\_AD31 signal is asserted, PCI memory accesses in the 0 to 2G address range are ignored by the 660. There is no system memory access, no snoop cycle, and the 660 does not claim the transaction. When the IGN\_PCI\_AD31 signal is asserted, the 660 maps PCI memory accesses from 0 to 2G directly to system memory at 0 to 2G. PCI memory accesses from 2G to 4G are mapped to system memory from 0 to 2G.

PCI memory accesses that are mapped to system memory cause the 660 to claim the transaction, access system memory, and arbitrate for the CPU bus and broadcast a snoop operation on the CPU bus. A detailed description of the snoop process is presented in the 660 User's Manual.

Table 5-3 gives a more detailed breakdown of the MCM response to PCI memory transactions in the 0 to 2G range. Note that the preferred mapping of PCI memory, so that it can be accessed both by the CPU and by PCI busmasters, is from 16M to 1G–2M.

Table 5-3. Mapping of PCI Memory Space, Part 2

PCI Bus Address	Target Resource	System Memory Address	Snoop Address
2G to 4G	System memory (1)	0 to 2G	0 to 2G
1G–2M to 2G	Reserved (2)	No system memory access. The 660 ignores PCI memory transactions in this range.	No snoop.
16M to 1G–2M	PCI Memory		
0 to 16M	PCI/ISA Memory (3)		

**Notes:**

1. The 660 maps PCI busmaster memory transactions in the 2G to 4G range to system memory, and the CPU is unable to initiate PCI memory transactions to this address range, so do not map devices to this PCI memory address range.
2. The CPU (thru the 660) can not access the 1G–2M to 2G address range, so do not map PCI devices herein unless the CPU will not access them.
3. Transactions initiated on the PCI bus by the ISA bridge on behalf of an ISA busmaster only (IGN\_PCI\_AD31 asserted for an SIO), are forwarded to system memory and broadcast snooped to the CPU bus from 0 to 16M. If this is not an ISA busmaster transaction, then the 660 ignores it. Note that the 660 will also forward PCI transactions from 16M to 2G if IGN\_PCI\_AD31 is asserted during an ISA-bridge-mastered transaction, and that this capability is not normally used.

### 5.1.3 PCI I/O Transaction Decoding

The 660 initiates PCI I/O transactions on behalf of the CPU. Other PCI busmasters are also allowed to initiate PCI I/O transactions. Table 5-4 shows the MCM mapping of PCI I/O transactions. The 660 ignores PCI I/O transactions.

PCI/ISA I/O is mapped to PCI I/O space from 0 to 64K. The ISA bridge subtractively decodes these transactions (and also PCI memory transactions from 0 to 16M). Other devices may actively decode and claim these transactions without contention.

PCI I/O is assigned from 16M to 1G–8M.

**Table 5-4. Mapping of PCI Master I/O Transactions**

PCI Bus Address	Target Resource	Other System Activity
1G–8M to 4G	Reserved (1)	The 660 ignores I/O transactions initiated by PCI busmasters.
16M to 1G–8M	PCI I/O devices	
8M to 16M	Reserved (1)	
64K to 8M	Reserved (2)	
0 to 64K	PCI/ISA I/O	

**Note:**

1. The CPU (thru the 660) can not access this address range, so do not map PCI devices herein unless the CPU will not access them.
2. In contiguous mode, the CPU (thru the 660) can create PCI I/O addresses in the 64K to 8M range. In non-contiguous mode, the CPU can only access PCI addresses from 0 to 64K.

#### 5.1.4 ISA Master Considerations

In systems that implement IGN\_PCI\_AD31 and use an Intel SIO, memory transactions produced on the PCI bus (by the SIO on behalf of an ISA master) are forwarded to system memory at the corresponding address (0 —16M).

If ISA masters are utilized and the SIO is programmed to forward their cycles to the PCI bus, then no other PCI device (e.g. video) is allowed to be mapped at the same addresses because contention would result.

The SIO contains registers to control which ranges of ISA addresses are forwarded to the PCI bus.

The 660 samples IGN\_PCI\_AD31 during PCI busmaster memory transactions from 0 to 2G. If IGN\_PCI\_AD31 is negated, the 660 ignores the transaction, and if IGN\_PCI\_AD31 is asserted, the 660 forwards the transaction to system memory. In theory, the IGN\_PCI\_AD31 signal can be used by any PCI agent for this purpose, but to ensure PR–P compliance, this signal should be asserted only while the ISA bridge is initiating a PCI to memory transaction on behalf of an ISA master. One way to generate IGN\_PCI\_AD31 is to AND together the PCI\_GNT# signals of all of the PCI agents except the SIO and the 660. This will assert IGN\_PCI\_AD31 (during a PCI transaction) only while either the 660 or the SIO is the initiator (and the 660 knows when it is the initiator).

The required connectivity of IGN\_PCI\_AD31 prevents the SIO from initiating peer to peer PCI memory transactions in the 0 to 2G range. ISA masters cannot access any PCI memory. The SIO is allowed to initiate PCI memory transactions from 2G to 4G, and other PCI transaction types (I/O &etc.).

## **5.2 PCI Transaction Details**

Further details of the MCM implementation of various PCI transactions, are found in the 660 User's Manual.

### **5.2.1 Memory Access Range and Limitations**

PCI memory reads and writes by PCI busmasters are decoded by the 660 to determine if they access system memory. PCI memory reads and writes to addresses from 2G to 4G on the PCI bus are mapped by the 660 as system memory reads and writes from 0G to 2G. These PCI to memory transactions are checked against the `top_of_memory` variable to determine if a given access is to a populated bank. The logic of the 660 does not recognize unpopulated holes in the memory banks. PCI accesses to unpopulated locations below the `top_of_memory` are undefined.

PCI accesses to system memory are not limited to 32 bytes. PCI burst-mode accesses are limited only by the size of memory, PCI bus latency restrictions, and the PCI disconnect counter.

### **5.2.2 Bus Snooping on PCI to Memory Cycles**

Each time a PCI (or ISA) busmaster accesses memory, (and once again for each time a PCI burst crosses a cache block boundary) the 660 broadcasts a snoop operation on the CPU bus. If the CPU signals an L1 snoop hit by asserting `ARTRY#`, the 660 retries the PCI transaction. The ISA bridge then removes the grant from the PCI agent, who (according to PCI protocol) releases the bus for at least one cycle and then arbitrates again. Meanwhile, the 660 grants the CPU bus to the CPU, allowing it to do a snoop push. Then the PCI agent again initiates the original transaction.

During the transaction, the 660 L2 cache is monitoring the memory addresses. The L2 takes no action on L2 misses and read hits. If there is an L2 write hit, the L2 marks that block as invalid, does not update the block in SRAM, and does not affect the PCI transaction. L2 operations have no effect on PCI to memory bursts.

### **5.2.3 PCI to PCI Peer Transactions**

Peer to peer PCI transactions are supported consistent with the memory maps of Table 5-1, Table 5-2, Table 5-3, and Table 5-4, which together show the ranges of different bus command transactions that are supported. If the `ISA_MASTER` signal is used with the Intel SIO, then the SIO is not allowed to perform peer to peer PCI memory transactions in the 0 to 2G range. No other transaction types are affected.

### **5.2.4 PCI to System Memory Transactions**

Single and burst transfers are supported. Bursts are supported without special software restrictions. That is, bursts can start at any byte address and end on any byte address and can be of arbitrary length.

As per the PCI specification, the byte enables are allowed to change on each data phase. This has no practical effect on reads, but is supported on writes. The memory addresses linearly increment by 4 on each beat of the PCI burst. All PCI devices must use only linear burst incrementing.

In ECC mode, PCI to memory transactions that result in less than 8-byte writes, cause the memory controller in the 660 to execute a read-modify-write operation, during which 8 by-

tes of memory data are read, the appropriate bytes are modified, the ECC byte is modified, and then the resulting 8-byte doubleword is written to memory.

### 5.2.5 PCI to Memory Burst Transfer Completion

PCI to memory burst transfers continue to normal completion unless one of the following occurs:

- The initiating PCI busmaster disconnects. The 660 handles all master disconnects correctly.
- The 660 target disconnects on a 1 M boundary. The 660 disconnects on all 1M boundaries.
- The 660 target disconnects because the PCI disconnect timer has timed out.
- The CPU retries the snoop cycle that the 660 broadcast on the CPU bus. In this case, the 660 target retries the PCI busmaster. (Note that L2 hits do not affect the PCI to memory transaction. Read hits have no effect on the L2, and write hits cause the L2 to invalidate the block.)
- The 660 will target disconnect the PCI busmaster if the refresh timer times out. In this case, the 660 will disconnect at the end of the current data phase for writes, or at the end of the current cache block, for reads.

### 5.2.6 PCI to Memory Access Sequence

When a PCI access is decoded as a system memory read or write, the memory and CPU bus are requested and, when granted, a snoop cycle to the CPU bus and a memory cycle to system memory are generated. If the processor indicates a snoop hit in the L1 cache (ARTRY# asserted), then the memory cycle is abandoned and the PCI cycle is retried. The CPU then does a snoop push. The L2 cache does not need to do a snoop push because it is write-through, and, therefore, system memory always contains the result of all write cycles. See Section 3 for more L2 information.

### 5.2.7 PCI to Memory Writes

During PCI to memory burst writes, the 660 performs data gathering before initiating the cycle to the memory controller. The data gathering involves combining two PCI write cycles into one memory write cycle if the address of the first write cycle is even.

Minimum initial write access time to 70ns DRAM when the CPU bus is 66MHz and the PCI bus is 33MHz is 5-1-1-1 -3-1-1-1 PCI clocks for 4-4-4-4 -4-4-4-4 bytes of data (14 PCI clocks for 32 bytes of data). Subsequent data phases of the same burst are generally serviced at -3-1-1-1 -3-1-1-1 (12 PCI clocks for 32 bytes of data), giving a peak burst write rate of 32 bytes in 12 PCI clocks, or about 85MBps with a 33MHz PCI clock. This scenario holds while the RAS# timer (10us typical) does not time out, the burst remains within the same 4K memory page, and no refresh is requested (15us typ).

#### 5.2.7.1 Detailed Write Burst Sequence Timing

The detailed write sequence is affected by several factors, such as refresh requests, memory arbitration delays, page and/or bank misses, and cache boundary alignment. Table 5-5 shows the details of the various sequences that a PCI to memory burst write will experience, depending on the address (relative to a cache block boundary) of the first data phase of the transaction. The starting address of the numbering sequence shown on the top row was arbitrarily chosen as xx00, and could be any 32-byte aligned boundary. The times shown in Table 5-5 are in PCI clock cycles, and do not include any cycles that the PCI

master spends acquiring the PCI bus from the PCI bus arbiter. The initial data phase is timed from the assertion of FRAME# to the PCI clock at which the PCI master samples TRDY# active. Subsequent data phase times are from the PCI clock at which the previous TRDY# was sampled active to the PCI clock at which the current TRDY# is sampled active. All the numbers shown in Table 5-5 are for parity (or none) operation. The numbers are also correct for ECC mode operation as long as all the writes are gather-store pairs. Incurring a RMW operation costs 3 PCI\_CLKs.

**Table 5-5. PCI to Memory Write Burst Sequence Timing**

S								S									
00	04	08	0C	10	14	18	1C	20	24	28	2C	30	34	38	3C	40	...
W	1	1	1	Y	1	Z	1	X	1	Z	1	Z	1	Z	1	X	...
	W	1	1	Y	1	Z	1	X	1	Z	1	Z	1	Z	1	X	...
		W	1	1	1	Y	1	X	1	Z	1	Z	1	Z	1	X	...
			W	1	1	Y	1	X	1	Z	1	Z	1	Z	1	X	...
				W	1	1	1	G	1	Z	1	Z	1	Z	1	X	...
					W	1	1	G	1	Z	1	Z	1	Z	1	X	...
						W	1	G	1	Z	1	Z	1	Z	1	X	...
							W	G	1	Z	1	Z	1	Z	1	X	...
								W	1	1	1	Y	1	Z	1	X	...

s indicates a cache block boundary at 0 mod 32. Snoops are broadcast to the CPU bus when a PCI burst crosses one of these boundaries.

w is a function of a 1.5 PCI clock snoop delay and memory arbitration delays. If the CPU is accessing memory when the PCI agent begins the memory write burst, the 660 waits until the CPU completes the current CPU access before allowing the PCI to memory write to proceed. If the RAS# watchdog timer has timed out, the memory controller will precharge the RAS# lines, and if the refresh timer has timed out, the memory controller will do a refresh operation.

- W** (min) = 5 This occurs when the memory controller is idle and no refresh or RAS# timeout occurs.
- W** (typ) = 6 or 7 This occurs if the memory controller is in the middle (beat 3 of 4) of serving a CPU burst transfer when the PCI burst starts, and no refresh or RAS# timeout occurs.
- W** (max) = 23 This occurs when CPU1 is just starting a burst transfer to memory, followed by CPU2 starting a burst transfer to memory, after which a refresh happens to be required.

**X** is a function of snoop delays only. Whenever the memory access crosses a cache block boundary, the Bridge broadcasts a snoop cycle on the CPU bus. (Due to the posted write buffer structure, delays incurred by crossing a page boundary here do not show up until later in the sequence.)

**X** = 3 Always. (The only benefit to disabling PCI snooping or enabling pre-snooping is to reduce this delay to 1. Otherwise neither function increases performance.)

**Y** is a function of memory latency. This page and/or bank miss delay can only be incurred at a page boundary, but shows up here due to the posted write buffer structure. The Bridge has a 4 x 4 posted PCI write buffer, which allows it to accept data phases from the PCI bus while the memory controller is busy servicing page misses. This minimizes the transfer delays caused by these memory overhead functions.

- Y** (typ) = 1 This occurs for a page hit with no refresh. This is also the minimum.
- Y** (mid) = 2 This occurs for a page miss with no refresh.
- Y** (max) = 4 This occurs for a refresh (which also forces a page miss).

**Z** is a function of a subset of the **W** factors (RAS# timeouts and refresh operations). This delay is only incurred due to a RAS# timeout or refresh request that has occurred since the last **W**, **Y**, **Z**, or **G**.

- Z** (typ) = 2 to 3 This occurs for no refresh and no RAS# timeout.
- Z** (max) = 3 to 4 This occurs for either a RAS# timeout or a refresh operation.

**G** is the combination of **X** and **Y**, and is equal to the longer of **X** and **Y**.

### 5.2.8 PCI to Memory Reads

During PCI to memory burst reads, the 660 performs memory pre-fetching when it initiates cycles to the memory controller. The pre-fetching involves loading or pre-loading 32 bytes from the memory for eight 4-byte PCI read cycles. Pre-fetching is only done within the same cache line.

Minimum initial read access time from 70ns DRAM when the CPU bus is 66MHz and the PCI bus is 33MHz, is 8-1-1-1 -1-1-1-1 PCI clocks for 4-4-4-4 -4-4-4-4 bytes of data (15 PCI clocks for 32 bytes of data). Subsequent data phases of the same burst are generally serviced at -7-1-1-1 -1-1-1-1 (14 PCI clocks for 32 bytes of data), giving a peak burst read rate of 32 bytes in 14 PCI clocks, or about 73MBps with a 33MHz PCI clock. This scenario holds while the RAS# timer (10us typical) does not time out and no refresh is requested (15us typ).

#### 5.2.8.1 Detailed Read Burst Sequence Timing

The actual detailed read sequence is affected by several factors, such as the speed of the DRAM, refresh requests, memory arbitration delays, page and/or bank misses, and cache boundary alignment. Table 5-6 shows the details of the various sequences that a PCI to memory burst read will experience, depending on the address (relative to a cache block boundary) of the first data phase of the transaction. The starting address of the numbering sequence shown on the top row was arbitrarily chosen as xx00, and could be any 32-byte aligned boundary. The times shown in Table 5-6 are in PCI clock cycles, and do not include any cycles that the PCI master spends acquiring the PCI bus from the PCI bus arbiter. The initial data phase is timed from the assertion of FRAME# to the PCI clock at which the PCI master samples TRDY# active. Subsequent data phase times are from the PCI clock at which the previous TRDY# was sampled active to the PCI clock at which the current TRDY# is sampled active.

All the numbers shown in Table 5-6 are for ECC, parity, or no-error-checking operation.

Table 5-6. PCI to Memory Read Burst Sequence Timing

S								S									
00	04	08	0C	10	14	18	1C	20	24	28	2C	30	34	38	3C	40	...
N	1	1	1	1	1	1	1	M	1	1	1	1	1	1	1	M	...
	N	1	1	1	1	1	1	M	1	1	1	1	1	1	1	M	...
		N	1	1	1	1	1	M	1	1	1	1	1	1	1	M	...
			N	1	1	1	1	M	1	1	1	1	1	1	1	M	...
				N	1	1	1	M	1	1	1	1	1	1	1	M	...
					N	1	1	M	1	1	1	1	1	1	1	M	...
						N	1	M	1	1	1	1	1	1	1	M	...
							N	M	1	1	1	1	1	1	1	M	...
								N	1	1	1	1	1	1	1	M	...

S indicates a cache block boundary at 0 mod 32. Snoops are broadcast to the CPU bus when a PCI burst crosses one of these boundaries.

N is the number of PCI clocks required from the assertion of FRAME# until the master samples the first TRDY# (from the 660) active, and is a function of snoop and memory arbitration delays. If the CPU is accessing memory when the PCI agent begins the memory read burst, the 660 waits until the CPU completes the current CPU access before allowing the PCI to memory read to proceed. If the RAS# watchdog timer has timed out, the memory controller will precharge the RAS# lines, and if the refresh timer has timed out, the memory controller will do a refresh operation.

- N (min) = 5 This occurs when the memory controller is idle and no refresh or RAS# timeout occurs, and the access produces a page hit.
- N (typ) = 8 or 9 This occurs if the memory controller is in the middle (beat 3 of 4) of serving a CPU burst transfer when the PCI burst starts, and no refresh or RAS# timeout occurs.
- N (max) = 26 This occurs when CPU1 is just starting a burst transfer to memory, followed by CPU2 starting a burst transfer to memory, after which a refresh happens to be required.

M is a function of a 2-clock snoop delay and other delays caused by bridge overhead functions. Whenever the memory access crosses a cache block boundary, the Bridge broadcasts a snoop cycle on the CPU bus.

- M (typ) = 6 or 7 Unless a refresh or RAS# timeout occurs.
- M (typ) = 7 or 8 This occurs for a refresh or RAS# timeout.

The memory controller, running at its own speed, requests up to 4, 8-byte memory reads (into 8, 4-byte buffers in the 663) while the PCI target engine of the 660 is servicing the memory read transaction. Under worst case conditions (slow memory, etc.), the memory controller just keeps up with the PCI bus, and N goes up. Under better conditions, the memory controller gets ahead of the PCI read process, and N decreases.

### 5.2.9 PCI BE# to CAS# Line Mapping

Table 5-7 shows which CAS# lines are activated when a PCI master writes memory. Note that CAS[0]# refers to byte addresses 0 mod 8, CAS[1]# refers to byte addresses 1 mod 8, etc.. For read cycles, eight bytes of memory data are read on each access, but the master receives only the desired 4 bytes. The bytes are read or written to memory independently of BE or LE mode (the endian mode byte swappers are situated between the CPU and the rest of the system, not between the PCI and the rest of the system).

Table 5-7. Active CAS# Lines – PCI to Memory Writes, BE or LE Mode

PCI AD[2]	Byte Enables BE[ ]#				Column Address Selects CAS[ ]#							
	3	2	1	0	0	1	2	3	4	5	6	7
0	1	1	1	1								
0	1	1	1	0	X							
0	1	1	0	1		X						
0	1	1	0	0	X	X						
0	1	0	1	1			X					
0	1	0	1	0	X		X					
0	1	0	0	1		X	X					
0	1	0	0	0	X	X	X					
0	0	1	1	1				X				
0	0	1	1	0	X			X				
0	0	1	0	1		X		X				
0	0	1	0	0	X	X		X				
0	0	0	1	1			X	X				
0	0	0	1	0	X		X	X				
0	0	0	0	1		X	X	X				
0	0	0	0	0	X	X	X	X				
1	1	1	1	1								
1	1	1	1	0					X			
1	1	1	0	1						X		
1	1	1	0	0					X	X		
1	1	0	1	1							X	
1	1	0	1	0					X		X	
1	1	0	0	1						X	X	
1	1	0	0	0					X	X	X	
1	0	1	1	1								X
1	0	1	1	0					X			X
1	0	1	0	1						X		X
1	0	1	0	0					X	X		X
1	0	0	1	1							X	X
1	0	0	1	0					X		X	X
1	0	0	0	1						X	X	X
1	0	0	0	0					X	X	X	X

**Notes:**

X = active. Blank = inactive. Byte enables would normally represent contiguous addresses. This table shows what would happen for all cases.



### **5.3 Bus Arbitration Logic**

The MCM requires an external PCI arbiter such as may be supplied in the ISA bridge. The 660 sends a CPU/660 bus request to the PCI bus arbiter to request ownership of the PCI. The 660 receives a PCI bus grant from the PCI bus arbiter. The 660 follows the PCI specification for host bridges. The PCI arbiter typically parks the PCI bus on the 660.

The 100 MHz PPC 603e MCM example planar uses the Intel SIO as the PCI bus arbiter. The PCI arbiter sees the 660 as one of several PCI agents. The order of priority for PCI arbitration is programmable, and is initially set to be:

1. 660 (the SIO normally parks the bus on the 660)
2. PCI slot 1
3. PCI slot 2
4. PCI slot 3

For more information on arbitration, see Section 9 on planar initialization, and see the PCI Arbitration Controller section of the SIO data book. See the example planar schematics for the connection of the various PCI requests and grants.

There may be concurrency of cycles on the ISA bus (caused by DMA or ISA masters) with PCI or CPU transactions as long as the ISA bus operations are not forwarded to the PCI bus. Forwarding of ISA bus operations must wait for the ISA bridge to grant the PCI bus to its ISA interface.

## 5.4 PCI Lock

The MCM 660 does not set PCI locks, but does honor them. Also see Section 4.6.

### 5.4.1 PCI Busmaster Locks

The `PCI_LOCK#` signal is an input-only to the 660. The 660 provides resource locking of one 32-byte cache sector (block) of system memory. Once a PCI busmaster sets a lock on a 32-byte block of system memory, the 660 saves the block address. Subsequent accesses to that block from other PCI busmasters or from the CPU bus are retried until the lock is released.

The bridge generates a write-with-flush snoop cycle on the CPU bus when a PCI busmaster sets the PCI lock. The write-with-flush snoop cycle causes the L1 and L2 caches to invalidate the locked block, which prevents cache hits on accesses to locked blocks. If the L1 contains modified data (as indicated by a CPU retry), the PCI cycle is retried and the modified data is pushed out to memory.

### 5.4.2 CPU Bus Locking

The 660 does not set PCI locks when acting as the PCI busmaster.

The 60X processors do not have bus-locking functions. Instead, they use the *load reserve* and *store conditional* instructions (`lwarx` and `stwcx`) to implement exclusive access by setting a reservation on a memory block. To work with the `lwarx` and `stwcx` instructions, the 660 snoops all PCI accesses to system memory, which allows the CPU that is holding the reservation to detect a violation of the reservation. In addition, the 660 generates a write-with-flush operation on the CPU bus in response to the PCI read that begins a PCI lock.

## **5.5 PCI 2.1 Compliance**

The MCM contains a highly programmable system component, the 660 bridge. The 660 can be programmed to meet a wide range of application requirements. Along with this flexibility comes the ability to shoot oneself in the foot. This section discusses ways to program the 660 to ensure compliance with the PCI 2.1 specification.

### **5.5.1 PCI Target Initial Latency**

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 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. Program the memory controller for the fastest settings that are consistent with good design practice.

### **5.5.2 Transaction Ordering Rules**

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.

#### **5.5.2.1 The Problem**

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 known actual occurrences 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.

#### **5.5.2.2 Solution 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 reinitiate 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.

#### **5.5.2.3 Solution 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.

- 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.
- 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.

#### **5.5.2.4 Solution 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).

- 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.
- 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.

## 5.6 Related Bridge Control Registers

Information on these registers is contained in the *660 Bridge Manual*.

Bridge Control Register	Index	R/W	Bytes
Memory Controller Misc	8000 0821	R/W	1
PCI/BCR Configuration Address	8000 0CF8	R/W	4
PCI/BCR Configuration Data	8000 0CFC	R/W	4
PCI Type 0 Configuration Addresses IBM27–82650 Compatible	8080 08xx thru 80C0 00xx	R/W	4
PCI Vendor ID	Index 00 – 01	R	2
PCI Device ID	Index 02 – 03	R	2
PCI Command	Index 04 – 05	R/W	2
PCI Device Status	Index 06 – 07	R/W	2
Revision ID	Index 08	R	1
PCI Standard Programming Interface	Index 09	R	1
PCI Subclass Code	Index 0A	R	1
PCI Class Code	Index 0B	R	1
PCI Cache Line Size	Index 0C	R	1
PCI Latency Timer	Index 0D	R	1
PCI Header Type	Index 0E	R	1
PCI Built-in Self-Test (BIST) Control	Index 0F	R	1
PCI Interrupt Line	Index 3C	R	1
PCI Interrupt Pin	Index 3D	R	1
PCI MIN_GNT	Index 3E	R	1
PCI MAX_LAT	Index 3F	R	1
PCI Bus Number	Index 40	R	1
PCI Subordinate Bus Number	Index 41	R	1
PCI Disconnect Counter	Index 42	R/W	1
PCI Special Cycle Address BCR	Index 44 – 45	R	2
Error Enable 1	Index C0	R/W	1
Error Status 1	Index C1	R/W	1
Error Enable 2	Index C4	R/W	1
Error Status 2	Index C5	R/W	1
PCI Bus Error Status	Index C7	R/W	1
CPU/PCI Error Address	Index C8 – CB	R/W	4

This image shows a full page of blank graph paper. The grid consists of small, equal-sized squares formed by thin black lines. There are 20 columns and 20 rows of squares, creating a total of 400 square units. The grid covers the entire area of the page, leaving no margins or other markings.This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

## **Section 6**

### **ROM**

The MCM implements a 2M ROM space from 4G–2M to 4G (on the CPU bus) by means of the 660 bridge. The 660 provides two boot ROM device access methods which minimize pin and package count while still allowing a byte-wide Flash™ ROM device to source 8-byte wide data.

The ROM mode is indicated to the 660 on the strapping pin configuration bits during power-on-reset (POR). See section 8.3.2.2.

The ROM access method used by the example planar is referred to as the direct-attach ROM mode; the ROM attaches directly to the 660 (MCM) using the PCI\_AD lines. This mode is required when using the Intel™ SIO ISA bridge (as the example planar does) because the SIO does not support mapping of the ROM to the ISA bus. The direct-attach mode also supports ROM device writes and write-protect commands.

The example planar uses an AMD AM29F040-120 Flash™ ROM to contain the POST and boot code. This is a 512K device located at 4G–2M. It is recommended that Vital Product Data (VPD) such as the motherboard speed and native I/O complement be programmed into this device. It is possible to program the Flash before or during the manufacturing process.

The other ROM access method (remote ROM mode – see section 6.2) attaches the ROM device to an external PCI agent which supports the PowerPC Reference Platform ROM space map and access protocol. CPU busmaster transfers to ROM space are forwarded to the PCI bus and claimed by the PCI agent, which supplies the ROM device data. This PCI device is typically a PCI to ISA bridge. The ROM device attaches to the ISA bridge through the ISA bus lines, thereby saving a PCI bus load. The 660 supplies write-protect capability in this mode.

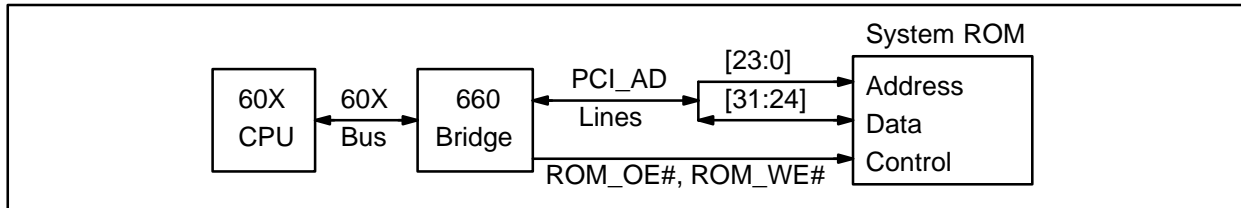
At power-on, the 603/604 CPU comes up in BE Mode with the L1 cache disabled, and begins fetching instructions (using 8-byte single beat reads) at address FFF0 0100 (4G – 1M + 100h). The example planar logic also resets to BE mode.

#### **6.1 Direct-Attach ROM Mode**

The ROM device attaches to the 660 by means of control lines and the PCI\_AD[31:0] lines. When a CPU busmaster reads from the ROM, the 660 masters a BCR transaction, during which it reads the ROM and returns the data to the CPU. CPU writes to the ROM and ROM write-protection operations are also forwarded to the ROM device.

ROM accesses flow from the CPU bus to the 660. As shown in Figure 6-1, the data and address flow from the 660 to the ROM over the PCI\_AD lines. ROM control flows from the 660 to the ROM over control lines that are not a part of the PCI bus.

Although connected to the PCI\_AD lines, the direct-attach ROM is not a PCI agent. The ROM and the PCI agents do not interfere with each other because the ROM is under 660 control, and the 660 does not enable the ROM except during ROM cycles. The 660 accesses the ROM by means of BCR transactions. Other PCI devices cannot read or write the ROM because they cannot generate BCR transactions.



**Figure 6-1. ROM Connections**

### 6.1.1 ROM Reads

When a CPU busmaster reads from memory addresses mapped to ROM space, the 660 arbitrates for the PCI bus and then masters a BCR transaction on the PCI bus. During this transaction, the 660 reads the ROM eight times, accumulates the data, and returns the double-word to the CPU. The 660 then completes the PCI transaction and releases the PCI bus.

The 664 drives the address of the required byte over PCI\_AD[23:0] to the ROM address pins. The ROM drives back the data on PCI\_AD[31:24], where it is received by the 663.

This ROM read discussion assumes that the system is in big-endian mode. For the effects of little-endian mode operation on ROM reads, see Section 6.1.1.3.

#### 6.1.1.1 ROM Read Sequence

Figure 6-2 is a timing diagram of a CPU to ROM read transaction. This case assumes that the PCI bus is parked on the CPU, so that the 660 has a valid PCI bus grant when the CPU starts the CPU bus transfer.

Initially, the CPU drives the address and address attributes onto the CPU bus and asserts TS#. The 660 decodes the CPU transfer as a ROM read transaction. It is possible for TS# to be asserted across either a rising or falling edge of PCI\_CLK. The 660 must only assert (and negate) PCI bus signals on the rising edge of PCI\_CLK, so if TS# is asserted across a rising edge of PCI\_CLK, the 660 waits one CPU\_CLK to synchronize to the PCI bus.

The 660 initiates a BCR transaction by asserting PCI\_FRAME# on the rising edge of PCI\_CLK. Note that the 660 is driving PCI\_AD[23:0] with the ROM address of byte 0 of the 8-byte aligned double-word. The 660 leaves PCI\_AD[31:24] tri-stated, and asserts ROM\_OE# to enable the ROM to drive the data onto these bits. On the next PCI\_CLK, the 660 negates PCI\_FRAME# and asserts PCI\_IRDY#.

The ROM drives the requested data onto its data pins, across PCI\_AD[31:24], and into the 660. Seven PCI\_CLKs after the 660 asserts PCI\_FRAME#, it sends ROM\_LOAD low. On the next clock, the 660 latches in the ROM data on PCI\_AD[31:24], sends ROM\_LOAD# high, and increments the ROM address on PCI\_AD[23:0]. The byte from the ROM is latched into a byte shift register, which accumulates the bytes in an 8-byte double-word. The contents of the shift register move through the 660 and onto the CPU data bus.



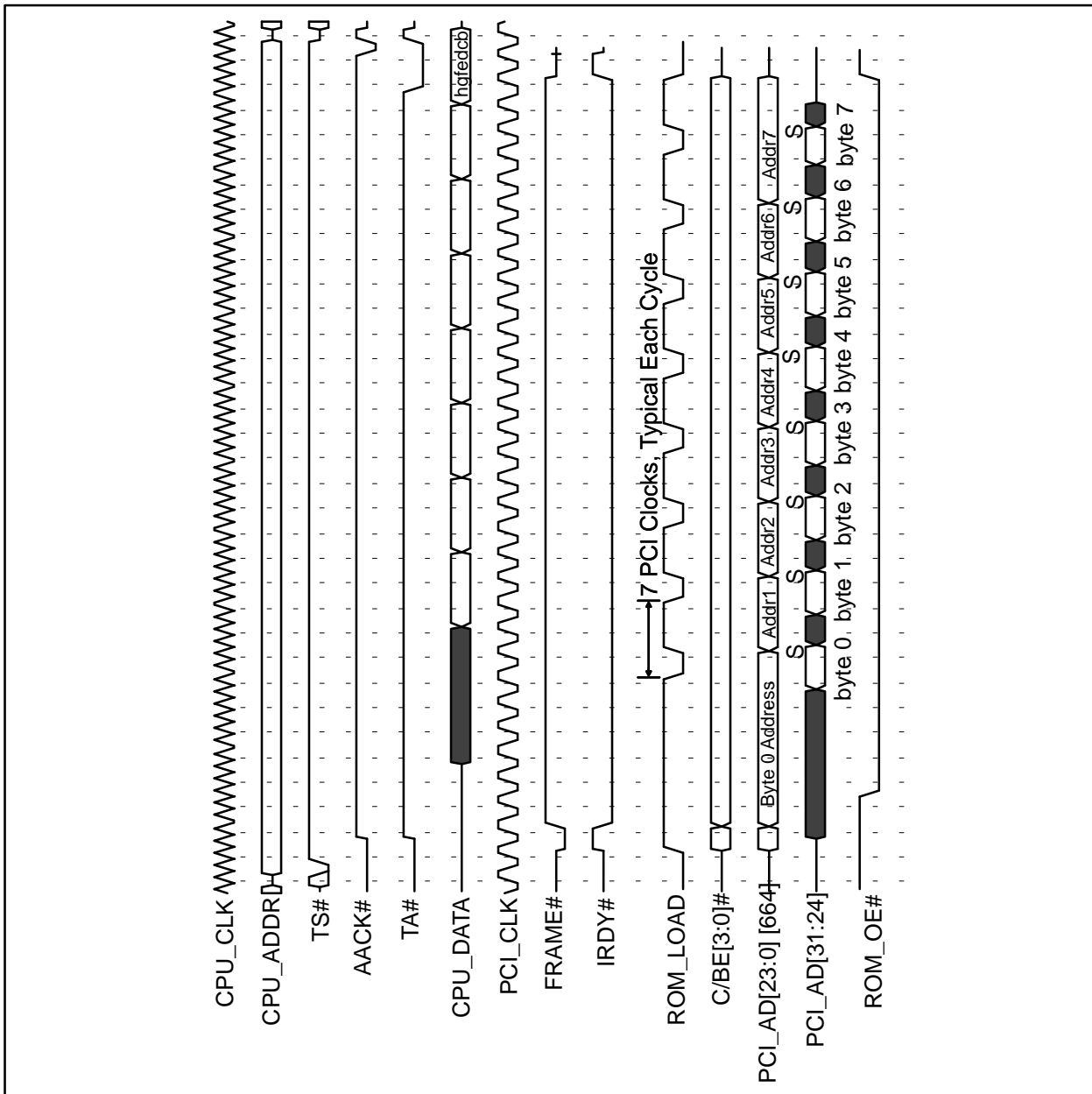


Figure 6-2. ROM Read Timing Diagram

The ROM then drives the next data byte onto PCI\_AD[31:24]. Seven PCI\_CLKs after it negated ROM\_LOAD for the previous byte, the 660 again negates ROM\_LOAD, and also shifts the previous byte of ROM data to the next position. On the next PCI\_CLK, the 660 sends ROM\_LOAD high and increments the ROM address on PCI\_AD[23:0]. This pattern is repeated until all eight bytes have been loaded into the shift register.

After the last byte has been latched into the 660 by the falling edge of ROM\_LOAD, the 660 completes the PCI transaction by deasserting PCI\_IRDY#. It also negates ROM\_OE# to clear the PCI bus. After the last byte of data has had time to propagate through onto the CPU data bus, the 660 signals TA# to the CPU. Table 6-1 shows the data and address flow during the transaction.

On a single-beat transfer, the CPU asserts and negates AACK# concurrently with TA#. For a burst transfer, the 660 asserts TA# for four CPU\_CLKs to return four identical double-words to the CPU. This is the only difference between single-beat and burst ROM reads. Also note that PCI\_DEVSEL#, PCI\_TRDY#, PCI\_STOP#, and ROM\_WE# are negated throughout the transaction.

Table 6-1. ROM Read Data and Address Flow

ROM Access #	ROM Data Byte	ROM Address PCI_AD[23:0]	ROM Data PCI_AD[31:24]	CPU_DATA[0:63] After Shift (BE Mode)	CPU_DATA[0:63] After Shift (LE Mode)
1	Byte 0	XX XXX0h	a	—	—
2	Byte 1	XX XXX1h	b	—	—
3	Byte 2	XX XXX2h	c	—	—
4	Byte 3	XX XXX3h	d	—	—
5	Byte 4	XX XXX4h	e	—	—
6	Byte 5	XX XXX5h	f	—	—
7	Byte 6	XX XXX6h	g	—	—
8	Byte 7	XX XXX7h	h	abcd efgh	hgfe dcba

#### 6.1.1.2 Address, Transfer Size, and Alignment

During ROM reads, system ROM is linear-mapped to CPU memory space from 4G – 2M to 4G (FFE0 0000h to FFFF FFFFh). This address range is translated onto PCI\_AD[23:0] as 0 to 2M (0000 0000h to 001F FFFFh). Since the CPU begins fetching instructions at FFF0 0100h after a reset, the most convenient way to use a 512K device as system ROM with the CPU is to use it from 4G – 512K to 4G. Connecting PCI\_AD[18:0] to ROM\_A[18:0] with no translation implements this. With this connection, the system ROM is aligned with 4G – 2M, but with alias addresses every 512K up to 4G. Other size devices can also be implemented this way.

The CPU read address need not be aligned on an 8-byte boundary. A CPU read from any ROM address of any length that does not cross an 8-byte boundary, returns all eight bytes of that double-word from the ROM. For example, the operations shown in Table 6-1 could have been caused by a CPU memory read to FF80 0100h, FF80 0101h, or FF80 0105h.

#### 6.1.1.3 Endian Mode Considerations

In little-endian mode, the address munging done by the CPU has no effect because PCI\_AD[2:0] are forced to 000 during the address phase by the 660 at the beginning of the transaction. However, in little-endian mode the byte swapper is enabled, so the bytes of ROM data returned to the CPU are swapped as shown in the last column of Table 6-1.

#### 6.1.1.4 4-Byte Reads

The 660 handles 4-byte ROM reads (and all ROM reads of less than 8 bytes) as if they were 8-byte reads. All 8 bytes are gathered by the 660, and all 8 bytes are driven onto the CPU data bus.

### 6.1.2 ROM Writes

The 660 decodes a CPU store word instruction to CPU address FFFF FFF0h as a ROM write cycle. (Note that the 660 treats any access from FFE0 0000h to FFFF FFFE, with

CPU\_A[31]=0, as an access to FFFF FFF0.) The three low-order bytes of the CPU data word are driven onto the ROM address lines, and the high-order byte is driven onto the ROM data lines. For example, a store word instruction with data = 0012 3456h writes 56h to ROM location 00 1234h. Only single-beat, four-byte write transfers (store word) are supported. A ROM write is considered to be a BCR operation. The ROM write BCR is detailed in Section 6.3.1.

The ROM write discussion assumes that the system is in big-endian mode. For the effects of little-endian mode operation on ROM reads, see Section 6.1.2.3. In direct-attach mode, the ROM is attached to the 660 as shown in Figure 6-3.

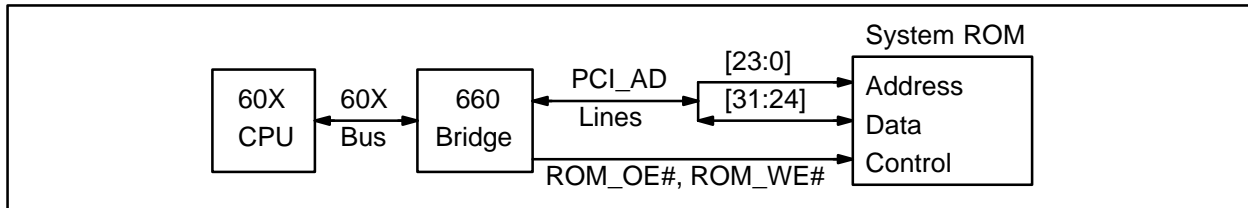


Figure 6-3. ROM Connections

#### 6.1.2.1 ROM Write Sequence

This case assumes that the PCI bus is parked on the CPU. Initially, the CPU drives the address and address attributes onto the CPU bus and asserts TS#. The 660 decodes the CPU transfer as a ROM write transaction, which is a BCR transaction.

The 660 initiates a BCR transaction by asserting PCI\_FRAME# on the rising edge of PCI\_CLK. Note that the 660 is driving PCI\_AD[23:0] with the ROM address and PCI\_AD[31:24] with the ROM data. On the next PCI\_CLK, the 660 negates PCI\_FRAME# and asserts IRDY#. Four PCI\_CLKs after the 660 asserts PCI\_FRAME#, it asserts ROM\_WE# for two PCI\_CLKs.

The 660 completes the PCI transaction by deasserting PCI\_IRDY#. The 660 signals TA# and AACK# to the CPU to signal transfer completion to the CPU. Also note that PCI\_DEVSEL#, PCI\_TRDY#, PCI\_STOP#, and ROM\_OE# are negated throughout the transaction.

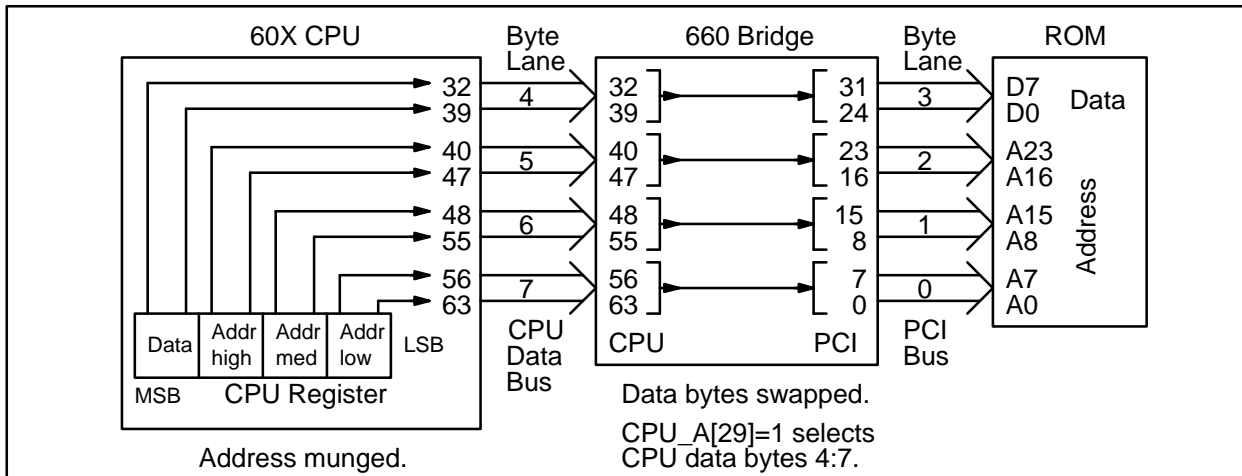
#### 6.1.2.2 Write Protection

Write protection for direct-attach ROM is provided through the ROM lockout BCR (see Section 6.3.2). ROM write-lockout operations are compatible with the 650 bridge.

When a CPU busmaster writes any data to memory address FFFF FFF1h, the 660 locks out all subsequent ROM writes until the 660 is reset. In addition, flash ROM devices can have the means to permanently lock out sectors by writing control sequences. Flash ROM specifications contain details. Note that the 660 treats any access from FFE0 0001 to FFFF FFFF, with CPU\_A[31]=1, as an access to FFFF FFF1.

#### 6.1.2.3 Data Flow In Little-Endian Mode

Figure 6-4 and Table 6-2 show the flow of CPU Data through the 660 to the ROM while the system is in little-endian mode. Note that the CPU Data bus is labeled in big-endian order, the PCI bus is labeled in little-endian order, and the 660 is labeled to match (and the bit significance within the bytes is maintained).



**Figure 6-4. ROM Data and Address Flow In Little Endian Mode**

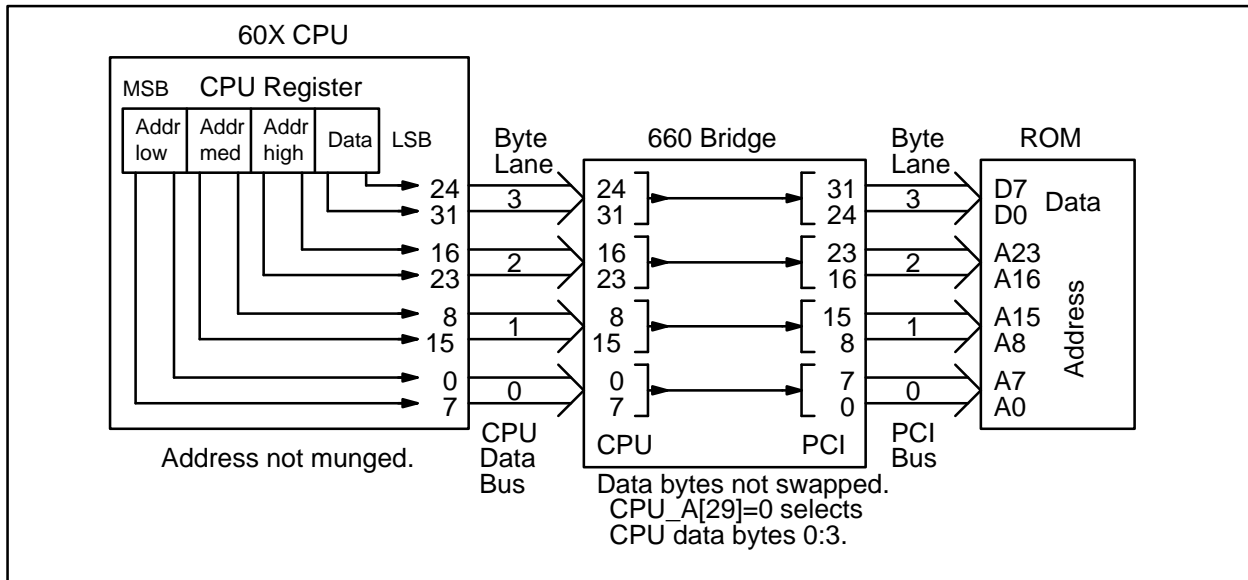
When the CPU executes a store word instruction to FFFF FFF0h, the contents of the source register appear on CPU\_DATA[32:63]. CPU\_ADDR[29] is 1 (after the CPU munges the address), so the 660 selects CPU data byte lanes 4 through 7 as the source of the data. The system is in little-endian mode, so the buffer swaps the data bytes. If the register data is AB012345h, then ABh is written to address 012345h of the ROM. Only single-beat, four-byte write transfers (store word) are supported.

**Table 6-2. ROM Write Data Flow in Little-Endian Mode**

CPU Register	CPU DATA[0:63]	Content	PCI_AD[31:0]	ROM Signal
MSB	32:39	ROM Data	31:24	D[7:0]
	40:47	ROM Address high byte	23:16	A[23:16]
	48:55	ROM Address mid byte	15:8	A[15:8]
LSB	56:63	ROM Address low byte	7:0	A[7:0]

#### 6.1.2.4 Data Flow In Big-Endian Mode

Figure 6-5 and Table 6-3 show the flow of CPU Data through the 660 to the ROM while the system is in big-endian mode. Note that the CPU Data bus is labeled in big-endian order, the PCI bus is labeled in little-endian order, and the 660 is labeled to match (and the bit significance within the bytes is maintained).



**Figure 6-5. ROM Data and Address Flow In Big Endian Mode**

When the CPU executes a store word instruction to FFFF FFF0h, the contents of the source register appear on CPU\_DATA[0:31]. CPU\_ADDR[29]=0, so the 660 selects CPU data byte lanes 0 through 3 as the source of the data. The system is in big-endian mode, so the buffer does not swap the data bytes. If the register data is 452301ABh, then ABh is written to address 012345h of the ROM. Only single-beat, four-byte write transfers (store word) are supported.

**Table 6-3. ROM Write Data Flow in Big-Endian Mode**

CPU Register	CPU DATA[0:63]	Content	PCI_AD[31:0]	ROM Signal
MSB	0:7	ROM Address low byte	7:0	A[7:0]
	8:15	ROM Address mid byte	15:8	A[15:8]
	16:23	ROM Address high byte	23:16	A[23:16]
LSB	24:31	ROM Data	31:24	D[7:0]

## 6.2 Remote ROM Mode

In a system that uses the remote ROM mode, the ROM device attaches to a PCI agent. When a CPU busmaster reads from memory addresses mapped to ROM space, the 660 arbitrates for the PCI bus and then masters a memory read transaction on the PCI bus. The PCI agent claims the transaction and supplies the ROM device data. CPU writes to the ROM and ROM write-protection operations are also forwarded to the PCI agent.

As shown in Figure 6-6, the ROM access flows from the CPU to the 660 over the CPU bus, from the 660 to the PCI agent over the PCI bus, and from the PCI agent to the ROM device. The ROM device attaches to the PCI agent, not to the PCI\_AD lines, so a PCI bus load is saved by the remote ROM method.

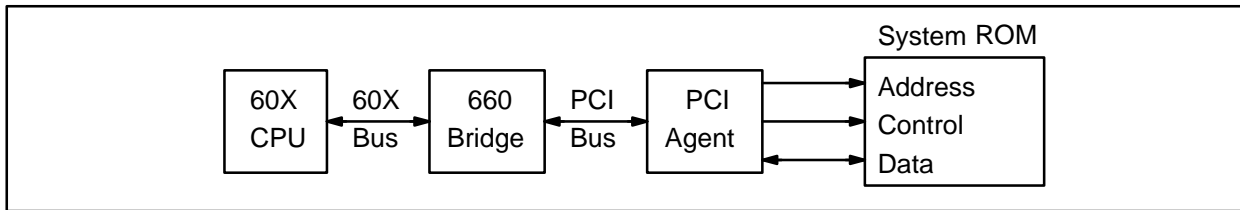


Figure 6-6. Remote ROM Connections

### 6.2.1 Remote ROM Reads

For remote ROM reads, the 660 arbitrates for the PCI bus, initiates eight single-byte PCI accesses, releases the PCI bus, and completes the CPU transfer. The eight single bytes of ROM data are assembled into a double-word in the 663 and passed to the CPU. Figure 6-7 shows the beginning of the operation, including the first two PCI transactions. Figure 6-8 shows the last part of the operation, including the last two PCI transactions.

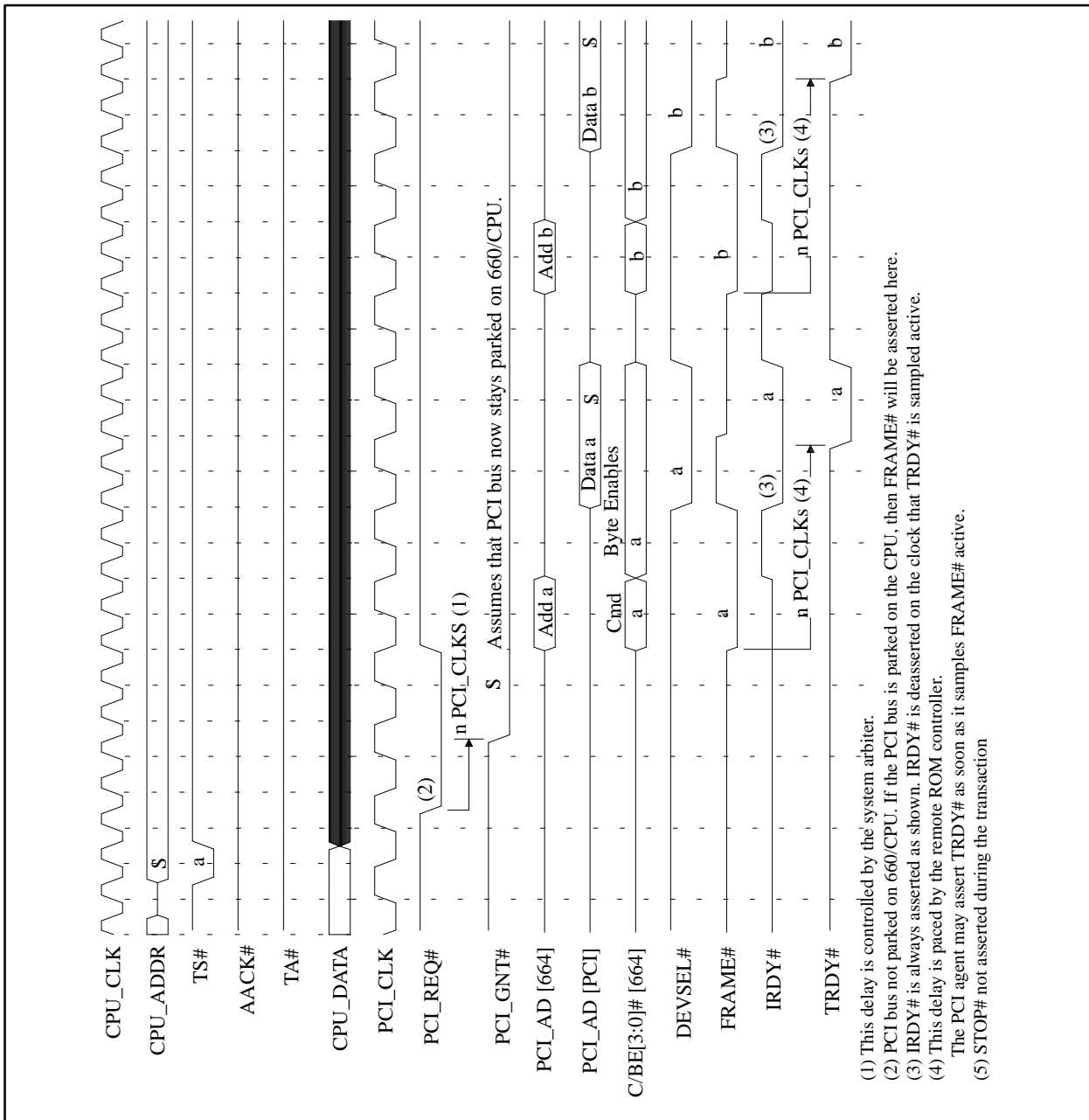
During and following reset, compliant PCI agents are logically disconnected from the PCI bus except for the ability to respond to configuration transactions. These agents have not yet been configured with necessary operational parameters. PCI agents capable of the remote ROM access protocol reset with the ability to respond to remote ROM accesses before being fully configured. The CPU begins reading instructions at FFF0 0100h before it can configure the PCI devices.

The ROM read discussion assumes that the system is in big-endian mode.

#### 6.2.1.1 Remote ROM Read Sequence

In response to a CPU bus read in the 4G – 2M to 4G address range, the 660 requests the PCI bus from the PCI arbiter. When the PCI bus is granted (or if the bus is already parked on the CPU), the 660 initiates a series of PCI memory-read transactions as shown in Table 6-4 for a CPU read from FFE0 0000h to FFFF FFFF. Note that the last column in Table 6-4 shows the effect of little-endian mode operation. See Section 6.2.1.4.

The address of the first transaction is the low-order byte of the double-word pointed to by the CPU address (see Section 6.2.1.2). The 660 expects the low-order byte of ROM data in the 8-byte double-word to be returned on PCI byte lane 0, PCI\_AD[7:0]. As shown in The 660 then masters seven more PCI read transactions, each time receiving back one byte of ROM data and driving it onto the CPU data bus as shown in Table 6-4. Note that the byte enables are incrementing within each 4-byte word pointed to by the PCI address.



**Figure 6-7. Remote ROM Read – Initial Transactions**

At the completion of the eighth PCI read, the 660 drives the assembled double-word onto the CPU data bus. The 660 then signals completion of the transfer to the CPU.

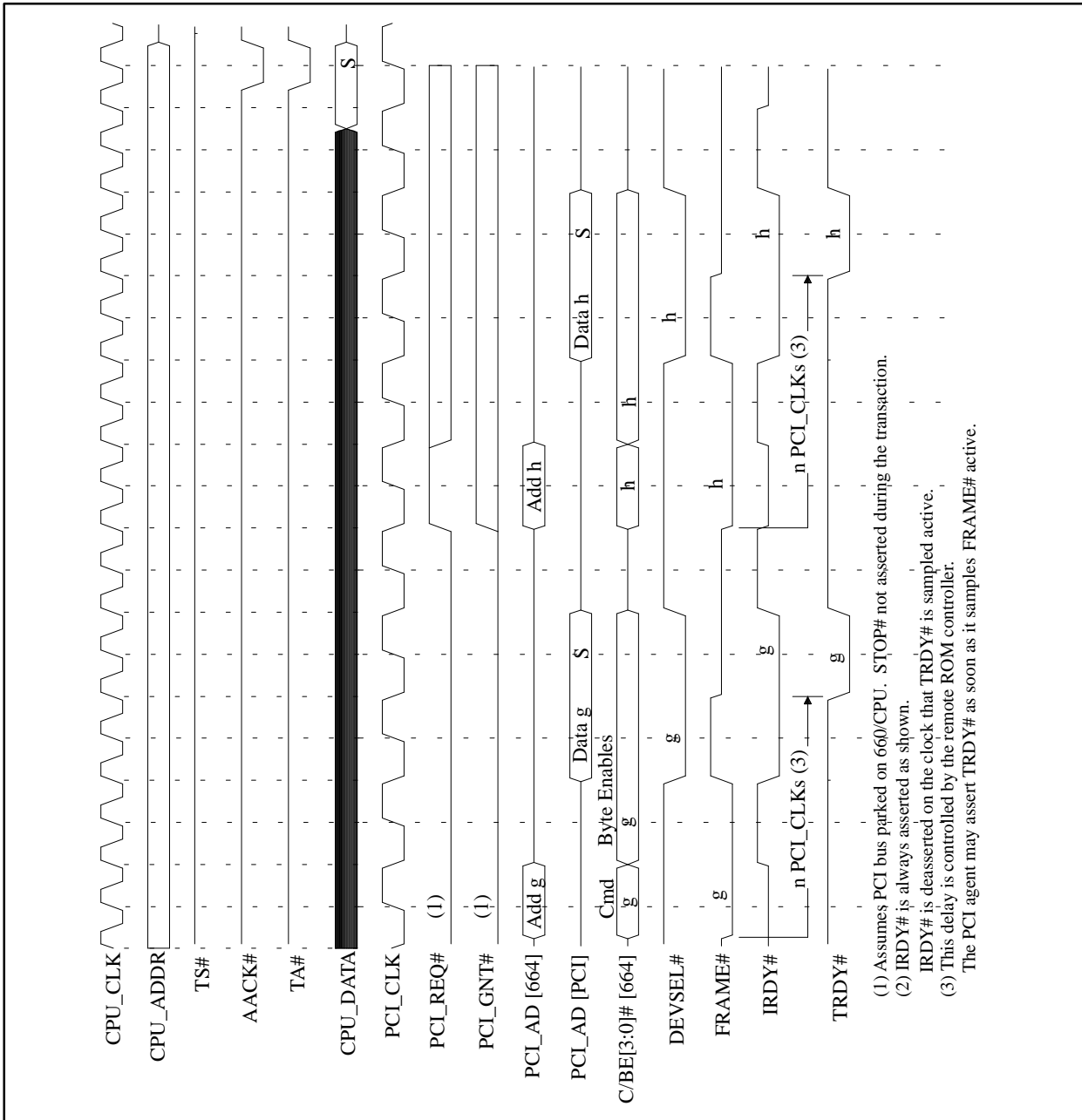


Figure 6-8. Remote ROM Read – Final Transactions

Remote ROM reads are not pipelined. The 660 does not assert AACK# to the CPU until the end of the remote ROM read sequence. The 660 asserts PCI\_REQ# throughout the entire remote ROM read sequence.



Table 6-4. Remote ROM Read Sequence, CPU Address = FFFX XXX0

PCI Access #	PCI Bus Read Memory Address PCI_AD[31:0]	Byte Enables PCI_C/BE[3:0]#	ROM Addr	ROM Data	Big Endian CPU_DATA [0:63]	Little Endian CPU_DATA [0:63]
1	FFFX XXX0h	1110	0	a	—	—
2	FFFX XXX0h	1101	1	b	—	—
3	FFFX XXX0h	1011	2	c	—	—
4	FFFX XXX0h	0111	3	d	—	—
5	FFFX XXX4h	1110	4	e	—	—
6	FFFX XXX4h	1101	5	f	—	—
7	FFFX XXX4h	1011	6	g	—	—
8	FFFX XXX4h	0111	7	h	abcd efgh	hgfe dcba

### 6.2.1.2 Address, Transfer Size, and Alignment

The initial PCI address generated during the remote ROM read sequence is formed by copying the high-order 29 bits of the CPU address, and forcing the three low order bits PCI\_AD[2:0] to 000b. This generates a base address that is aligned on an 8-byte boundary. While reading the lower 4 bytes, the 660 indicates which byte it is requesting using the PCI byte enables C/BE[3:0]#. After the first four bytes of ROM data are read, the 660 increments the address on the PCI\_AD lines by 4 before executing the second four PCI reads.

The CPU read address need not be aligned on an 8-byte boundary. A CPU read from any address (in ROM space) of any length that does not cross an 8-byte boundary within a double-word returns all eight bytes of that double-word data from the ROM. For example, the operations shown in Table 6-4 could have been caused by a CPU memory read to FFF0 0100h, FFF0 0101h, or FFF0 0105h.

Errors occurring during remote ROM reads are handled as usual for the error type. No special rules are in effect.

### 6.2.1.3 Burst Reads

The 660 supports burst reads in remote ROM mode. The 660 supports a pseudo burst mode, which supplies the same eight bytes of data (from the ROM) to the CPU on each beat of a 4-beat CPU burst.

A burst ROM read begins with the 660 executing a single-beat ROM read operation, which assembles eight bytes of ROM data into a double-word on the CPU data bus. For a burst ROM read, the 660 asserts TA# for four CPU\_CLK cycles, with AACK# asserted on the fourth cycle. The same data remains asserted on the CPU data bus for all four of the data cycles.

For a single-beat read, the 660 asserts TA# and AACK# for one CPU\_CLK cycle, and the CPU completes the transfer.

### 6.2.1.4 Endian Mode Considerations

In little-endian mode, the address munging done by the CPU has no effect because PCI\_AD[2:0] are forced to 000 during the address phase by the 660 at the beginning of the transaction. However, in little-endian mode the byte swapper is enabled, so the bytes of ROM data returned to the CPU are swapped as shown in the last column of Table 6-4.

#### **6.2.1.5 4-Byte Reads**

The 660 handles 4-byte ROM reads (and all ROM reads of less than 8 bytes) as if they were 8-byte reads. All 8 bytes are gathered by the 660, and all 8 bytes are driven onto the CPU data bus.

#### **6.2.2 Remote ROM Writes**

While the 660 is configured for remote ROM operation, the 660 forwards all CPU to ROM write transfers to the PCI bus as memory writes. The PCI agent that is controlling the remote ROM acts as the PCI target during CPU to ROM write transfers, executes the write cycle to the ROM, and may provide ROM write-protection.

##### **6.2.2.1 Write Sequence**

A CPU busmaster begins a remote ROM write transaction by initiating a one-byte, single-beat memory write transfer to CPU bus address range 4G – 2M to 4G (FF80 0000h to FFFF FFFFh).

The 660 decodes the CPU transfer, arbitrates for the PCI bus, and initiates a memory write PCI transaction to the same address in the 4G – 2M to 4G address range.

The PCI agent that is controlling the remote ROM (such as the PCI to ISA Bridge), claims the transaction, manages the write cycle to the ROM device, and signals TRDY#.

The 660 then completes the PCI transaction, and signals AACK# and TA# to the CPU. Note that remote ROM writes are neither posted or pipelined.

##### **6.2.2.2 Write Protection**

Write protection can be provided by the PCI agent that controls the ROM. In addition, some flash ROM devices can have the means to permanently lock out sectors by writing control sequences. The 660 also has a write lockout in the Bridge Chipset Options 2 register (bit 0 of index BBh).

##### **6.2.2.3 Address, Size, Alignment, and Endian Mode**

In remote ROM mode, CPU memory writes from 4G – 2M to 4G cause the 660 to generate PCI bus memory write transactions to 4G – 2M to 4G. The 660 does not allow CPU masters to access the rest of the PCI memory space from 2G to 4G.

In remote ROM mode, PCI busmaster memory write transactions from 4G – 2M to 4G are ignored by the 660. However, the PCI agent that controls the ROM responds to these transactions. In contrast, in direct-attach ROM mode, the 660 forwards PCI busmaster memory transactions from 2G to 4G (to populated memory locations) to system memory from 0 to 2G.

Remote ROM writes must be one-byte, single-beat transfers.

The endian mode of the system has no net effect on a ROM write because the transfer size is one byte. The address is munged by the CPU and unmunged by the 660. The data comes out of the CPU on the byte lane associated with the munged address, and then is swapped by the 660 to the byte lane associated with the unmunged address. Thus a ROM write in little-endian mode puts the data byte in the same ROM location as does the same ROM write in big-endian mode.

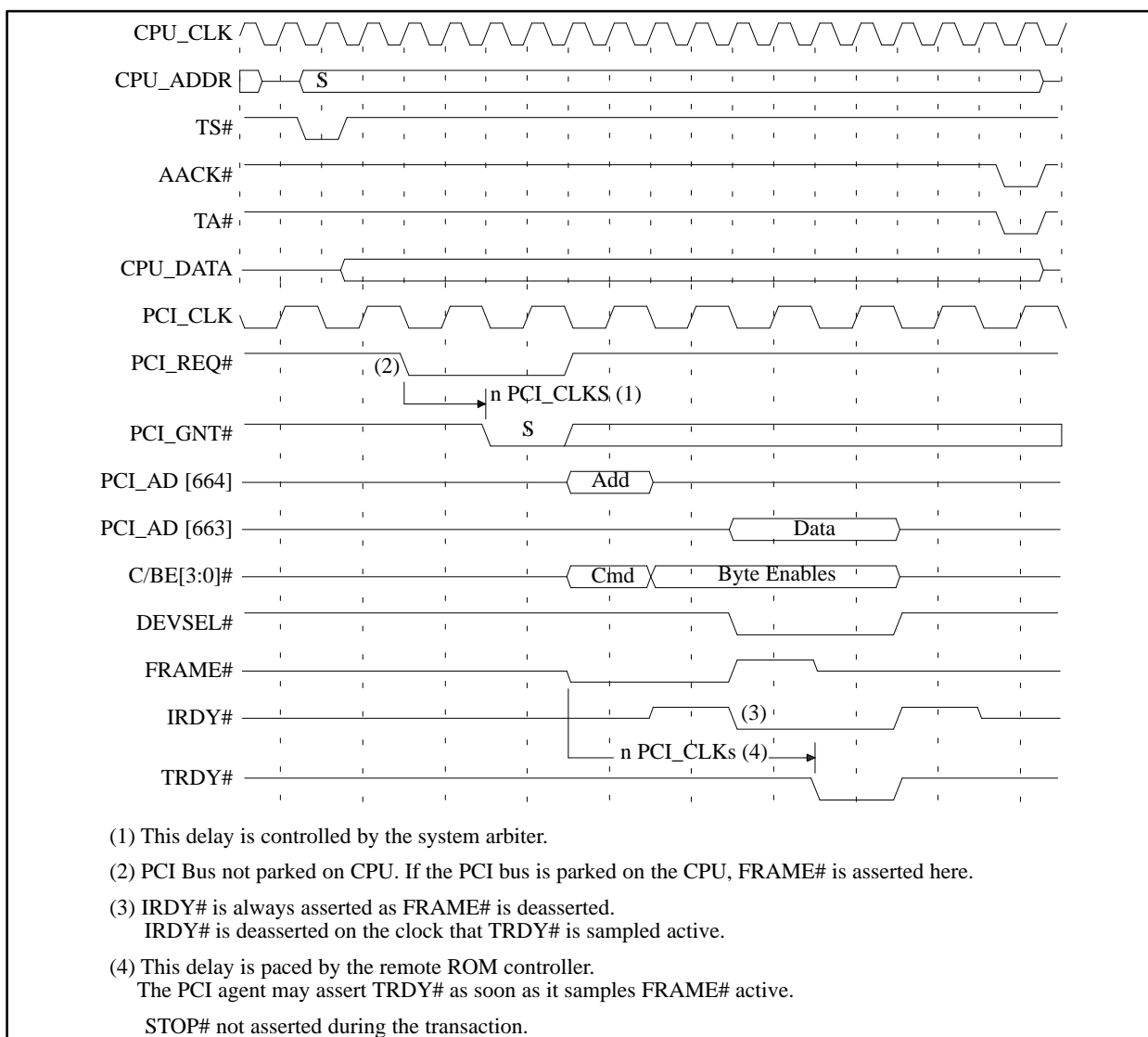


Figure 6-9. Remote ROM Write

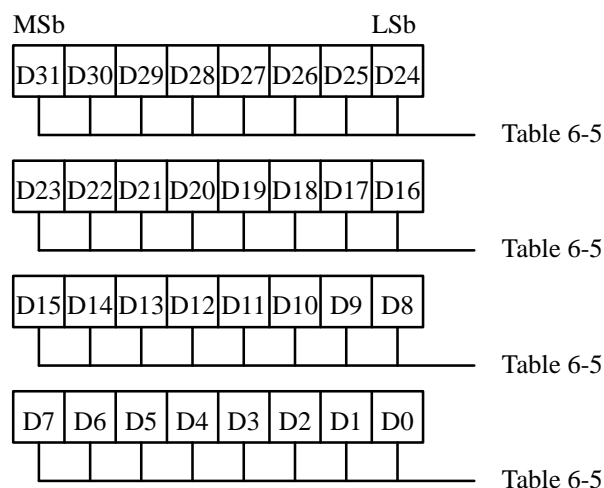
## 6.3 Related Bridge Control Registers

The two BCRs most closely related to the ROM system are the ROM write BCR and the ROM lockout register. Writes to the ROM are accomplished through the ROM write BCR. Write-protection is provided by means of the ROM lockout BCR.

### 6.3.1 ROM Write Bridge Control Register

Direct Access FFFF FFF0h	Write Only	Reset NA
--------------------------	------------	----------

This 32-bit, write-only register is used to program the ROM in direct-attach ROM systems (see section 6.1.2). This register must be written by means of a 4-byte transfer. Bits are shown with little-endian labels.

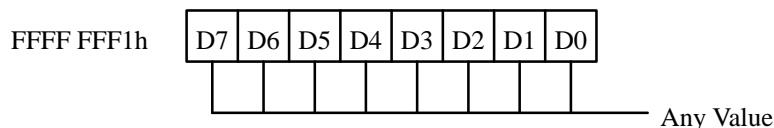
**Table 6-5. ROM Write BCR Contents**

BCR Byte	Content in Little-Endian System	Content in Big-Endian System
MSB	ROM Data	ROM Address low byte
	ROM Address high byte	ROM Address mid byte
	ROM Address mid byte	ROM Address high byte
LSB	ROM Address low byte	ROM Data

### 6.3.2 Direct-Attach ROM Lockout BCR

Direct Access FFFF FFF1h	Write Only	Reset NA
--------------------------	------------	----------

After it has been written once, this 8-bit, write-only register prevents direct-attach ROM writes.



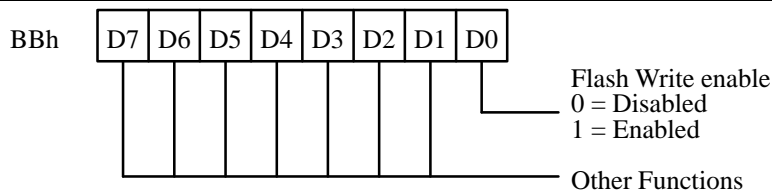
Bits 7:0

Writing any value to the register prevents all future writes to a ROM that is connected directly to the 660 through the PCI\_AD lines.

### 6.3.3 Remote ROM Lockout Bit

The ROM write-protect bit for remote ROM is in the Bridge Chipset Options 2 register (index BBh). While enabled, writes to the remote ROM are forwarded to the PCI memory space. While disabled, writes to the remote ROM are treated as no-ops and an error is signalled. After the first time that the bit is set to 0, it cannot be set back to 1.

Index BBh	Read/Write	Reset to 4Fh
-----------	------------	--------------



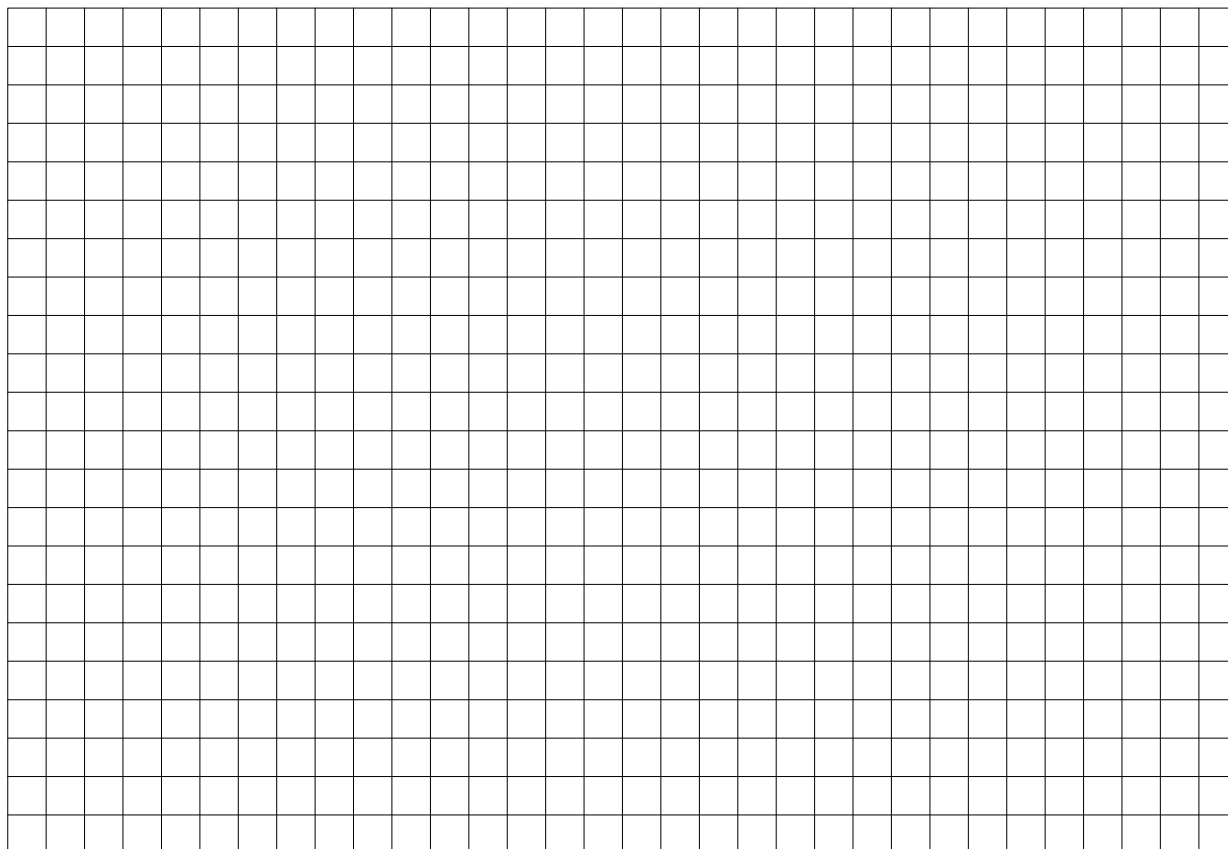
Bit 0

Flash write enable: When the ROM is remotely attached, this bit controls write access to the flash ROM address space (4G – 2M to 4G). When enabled, writes to this space are forwarded to the PCI memory space at the same address. When disabled, writes to this space are treated as no-ops and an error is signalled. After the bit is set to 0 (disabled), it cannot be reset to 1 (enabled).

### 6.3.4 Other Related BCRs

Information on these registers is contained in the *660 Bridge Manual*.

Bridge Control Register	Index	R/W	Bytes
Error Enable 1	Index C0	R/W	1
Error Enable 2	Index C4	R/W	1



---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

## Section 7 Clocks

The 100 MHz PPC 603e MCM provides a separate clock signal for each CPU bus and PCI bus agent in the system. The clock signals are generated by a Motorola™ MPC970. The MPC970 inputs are brought off of the MCM to allow maximum programming flexibility. See the data sheet for more information on MPC970 programming, capabilities, and characteristics. See Figure 7-1.

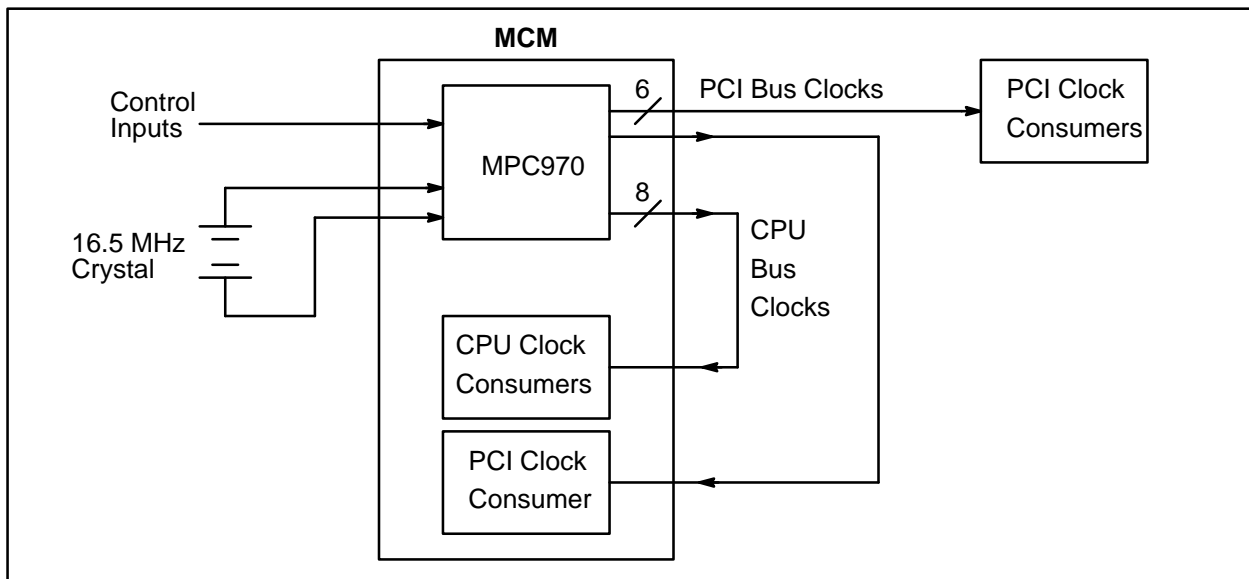


Figure 7-1. MCM Clocks

The example planar configures the MPC970 to produce 8 CPU bus clocks, all of which are consumed by the MCM components. When required, additional CPU bus clocks can be generated using a 'zero delay' clock repeater such as the Motorola MPC930 PLL, using one of the supplied CPU clocks as the seed clock.

The planar also configures the MPC970 to produce seven PCI bus clocks, two of which are consumed by the MCM components. The other PCI clocks are available for use on the planar. The MPC 970 is configured to interface with a 16.5 MHz crystal. An oscillator internal to the component then produces the root frequency used to create all clock outputs.

## 7.1 CPU Clock Physical Design Rules

The MCM and the example planar were physically designed with careful attention to the fact that, at PowerPC operating frequencies, the circuit board itself becomes a component that materially affects circuit behavior. Clock nets are the most critical wiring on the board. Their wiring requirements should be given priority over the requirements of other groups of signals. The following design rules are helpful in designing low-noise and low-skew clock nets:

1. Clock nets are to have a minimum number of vias.
2. No clock wires may be routed closer than one inch to the edge of the board. Consider adding EMC caps to the far end of the clock trace.
3. Clock nets should not have more than two nodes. Daisy chains, stubs, and star fanouts are not allowed.
4. Clock nets are to be routed as much as possible on internal signal planes.
5. Route a ground trace as a shield in the adjacent wiring channel on both sides of the clock trace. It is a good practice to periodically (every inch or so) connect these shield traces to the ground plane. Completely surround the clock trace with shield traces. Avoid impedance bumps.
6. Series (source) termination resistors are required. Choose them according to the MPC970 data sheet recommendations and place them as close as possible to the clock source pin of the MCM.
7. To minimize clock skew on the planar, design the circuit board such that the combined length (MCM plus planar) of each of the clocks is the same. In other words, determine the required total length of the longest clock trace, and then make all of the other traces the same total length (see Figure 7-2).

Inside the MCM, the PCI clock traces run about 1 inch, and the CPU clock traces run about 3.5 inches. Thus the planar runs of the clocks should be adjusted accordingly.

Suppose in Figure 7-2, one of the PCI clocks planar runs is initially the longest, at 8.5 inches. Added to the about 1 inch of MCM run, the total length is about 9.5 inches. Correct design will then stretch all of the other PCI clock lines to 8.5 inches. The required planar run length of the CPU lines is then  $9.5 - 3.5 = 6$  inches. See Table 7-1 and Table 7-2.

**Table 7-1. Clock Net Lengths**

Net Length	Type	Name	Net Topology	
13.7500	CLK	XTAL1	J1.U30	U4.12
11.7071	CLK	XTAL2	J1.T29	U4.13
32.5000	CLK	FRZ_CLK	J1.B11	U4.3
18.6036	CLK	FRZ_DATA	J1.A22	U4.5
8.5866	CLK	PCLK_60X	J1.E18	U4.34
66.3536	CLK	TAG_BCLK	J1.N30	U4.36
29.9142	CLK	CLK_EXT_FB	J1.A17	U4.14
27.0000	CLK	CLK_FB_SEL	J1.A15	U4.9
27.5000	CLK	CLK_PLL_EN	J1.D33	U4.7
67.5000	CLK	SRAM_BCLK0	J1.W30	U4.44



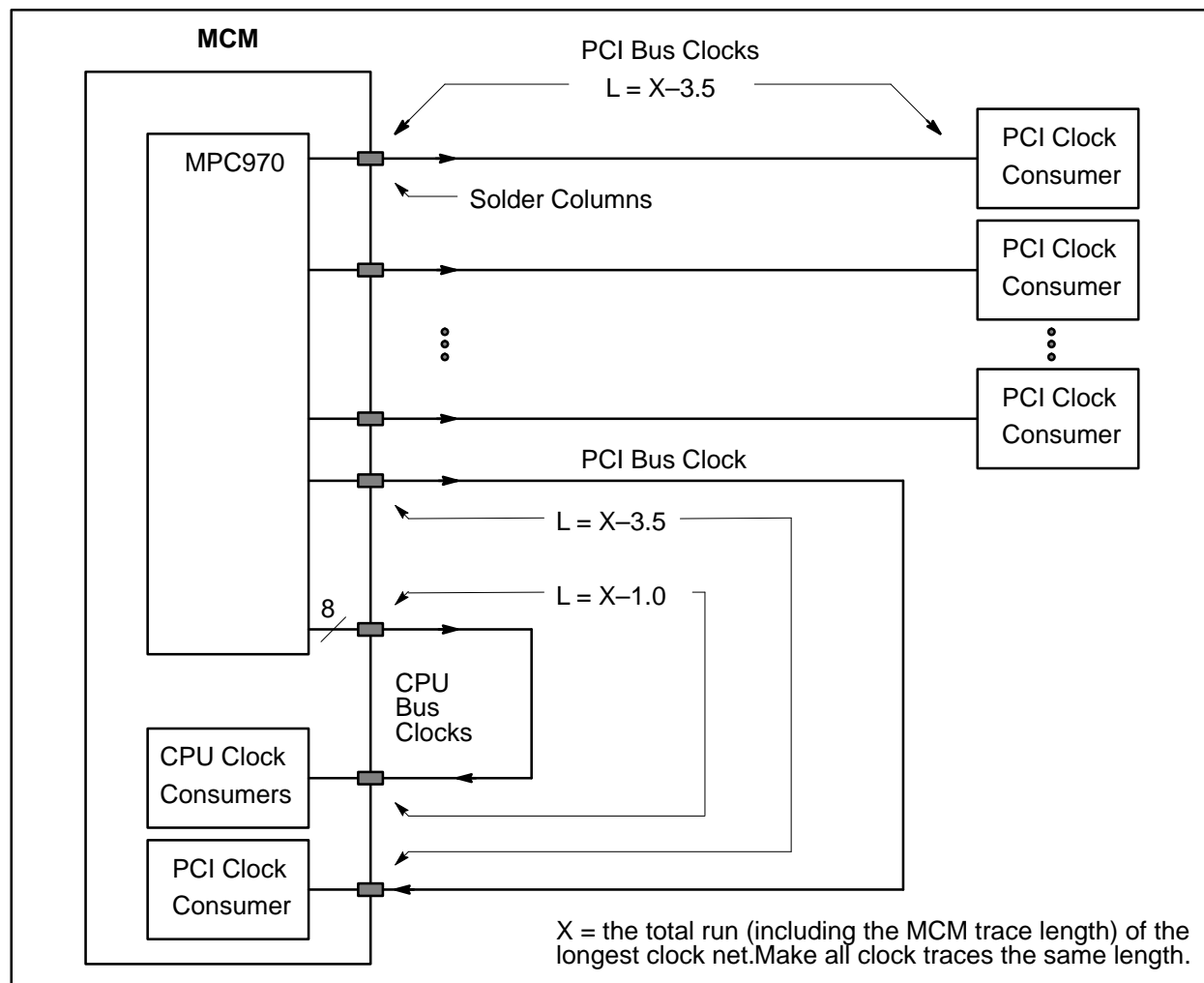
Table 7-1. Clock Net Lengths (Continued)

Net Length	Type	Name	Net Topology	
66.5000	CLK	SRAM_BCLK1	J1.AE30	U4.46
9.1624	CLK	SRAM_BCLK2	J1.AG04	U4.48
8.6624	CLK	SRAM_BCLK3	J1.AA04	U4.50
22.6036	CLK	CLK_COM_FRZ	J1.A18	U4.6
29.6036	CLK	CLK_REF_SEL	J1.B13	U4.8
24.4571	CLK	CLK_TTL_CLK	J1.A26	U4.11
28.0000	CLK	CLK_VCO_SEL	J1.A13	U4.52
1.7500	CLK	663_CPU_CLK	J1.G04	U4.38
63.6036	CLK	664_CPU_CLK	J1.G30	U4.42
24.1036	CLK	CLK_BCLK_DIV0	J1.C16	U4.31
22.2500	CLK	CLK_BCLK_DIV1	J1.B17	U4.27
26.3536	CLK	CLK_FRZ_STROBE	J1.F33	U4.4
14.8536	CLK	CLK_MPC601_CLKS	J1.K27	U4.40
25.3536	CLK	CLK_MR/TRISTATE	J1.A16	U4.2
61.8536	CLK	664_PCI_CLK	J1.J30	U4.18
20.6036	CLK	CLK_PCI_DIV0	J1.B21	U4.20
20.0607	CLK	CLK_PCI_DIV1	J1.C20	U4.26
38.1036	CLK	USER_PCICLK1	J1.AE01	U4.16
29.5000	CLK	USER_PCICLK2	J1.N01	U4.21
42.8536	CLK	USER_PCICLK3	J1.C01	U4.23
31.1036	CLK	USER_PCICLK4	J1.A11	U4.25
25.4571	CLK	USER_PCICLK5	J1.A14	U4.29
29.7500	CLK	USER_PCICLK6	J1.E33	U4.32

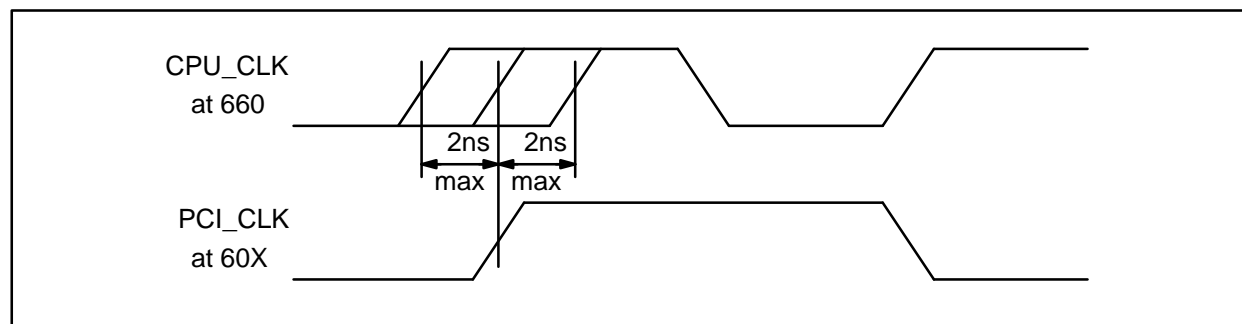
Table 7-2. Clock Net Calculations

Net	MCM Run	Total Run	Required Planar Run	Tolerance (Inch)
CPU Clocks	3.5 (nom)	x	x – 3.5	1
664_PCI_CLK	1 (nom)	x	x – 1	1
PCI_CLK[4:1]	1 (nom)	x	x – 1	1

In our experience, a maximum clock trace length skew of two inches is acceptable. It is the responsibility of the designer to determine the appropriate amount of allowed clock trace length variation for the individual application.

**Figure 7-2. MCM Clocks**

The allowed skew of the PCI\_CLK at any point in the system to the CPU\_CLK at the 660 Bridge is  $\pm 2\text{ns}$ , as shown in Figure 7-3.

**Figure 7-3. CPU\_CLK to PCI\_CLK Skew**

If the design rules laid out in Section 7.1 are followed, the requirements are met.

## **7.2 Clock Freezing**

Some of the PCI clocks may not be required. In this case, the ability of the MPC970 to stop (freeze) the unused clocks is useful to reduce run-time power consumption and EMI emissions.

The MPC970 can freeze any set of output clocks in the low state, while allowing the other clocks to continue running. The freeze command is given to the MPC970 via a two wire synchronous serial interface that originates in the system EPLD. See the clock freeze register description in the System EPLD section for more information on activating this feature.

Firmware can read the presence detect bits of the PCI slots to determine which devices are present. For PCI slots which are not populated, firmware can disable the clocks for those slots.

This function can also be used by power management functions to further reduce power consumption of the motherboard by freezing the clocks of devices which have been placed in a power managed mode.

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

## **Section 8**

### **Exceptions**

This section contains information on interrupts, errors, resets, and test modes. Some of the functionality of each area is implemented on the MCM, and some of that functionality is located on the planar. To simplify the material, both the MCM and the planar parts of each system are presented in this section.

#### **8.1 Interrupts**

##### **8.1.1 Planar Interrupt Handler**

Example planar interrupts are handled primarily by the interrupt controller in the ISA bridge, the 660, the CPU, and the firmware.

There are two 8259 type interrupt controllers located in the ISA bridge. These controllers receive and prioritize example planar interrupts, which can be asserted by motherboard logic, PCI devices, or ISA devices. The interrupt controller then asserts an interrupt to the 660.

The interrupt controller is programmed to handle both ISA and PCI interrupts using the correct protocols, under software control. Much of the operation of the interrupt controller is programmable. See the SIO data book for more information.

##### **8.1.2 MCM (660) INT\_REQ and INT\_CPU#**

The 660 features two interrupt inputs, INT\_REQ and NMI\_REQ.

As shown in Figure 8-1, the 660 inverts INT\_REQ and passes it thru to the CPU as INT\_CPU#. While the 660 is in 601 error reporting mode, it also uses INT\_CPU# to report certain error conditions, but this function is not used by the MCM.

The only reason that the 660 connects to INT\_CPU# is to be able to use it in reporting errors to the 601 CPU. When the bridge is not in 601 error reporting mode, the path through the 660 from INT\_REQ to INT\_CPU# is functionally an inverting latch. The CPU does not need the interrupt to be synchronized to the CPU clock, and typical interrupt controllers feature programmable output polarity, so if the target system is not using a 601, then the interrupt can be wired around the 660, without being connected to the 660. In this case, tie the INT\_REQ input inactive. This could have been done on the example planar; the current connectivity illustrates the use of the 660 pins.

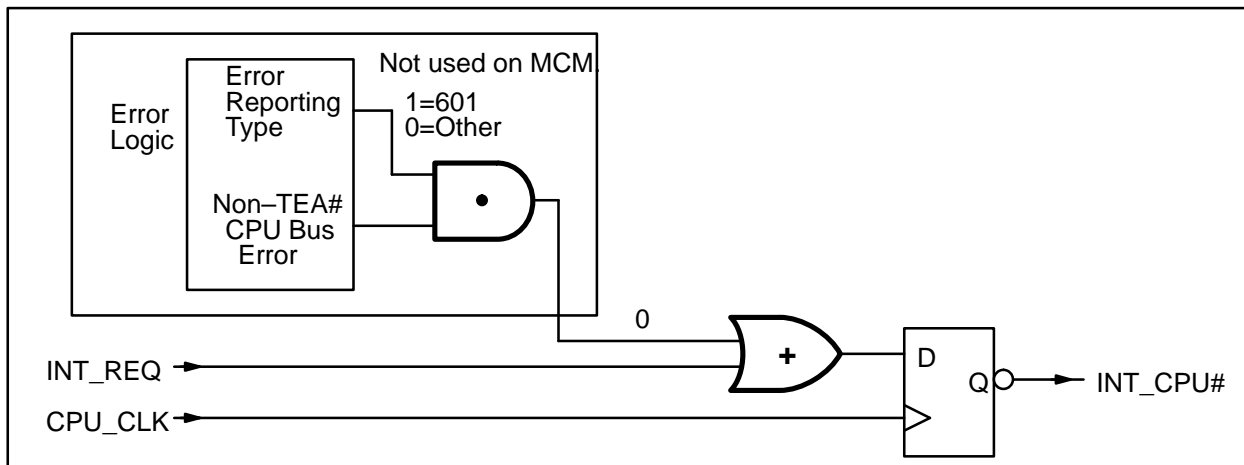


Figure 8-1. Conceptual Block Diagram of INT Logic

### 8.1.3 Interrupt Acknowledge Transactions

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 busmasters can initiate interrupt acknowledge transactions.

### 8.1.4 NMI\_REQ

The 660 considers the NMI\_REQ input to be an error indicator. Note that in Figure 8-1, there is no logical connection between NMI\_REQ and INT\_CPU, except through the error handling logic. See section 8.2.7.1 for more information on NMI\_REQ.

### 8.1.5 Interrupt Handling

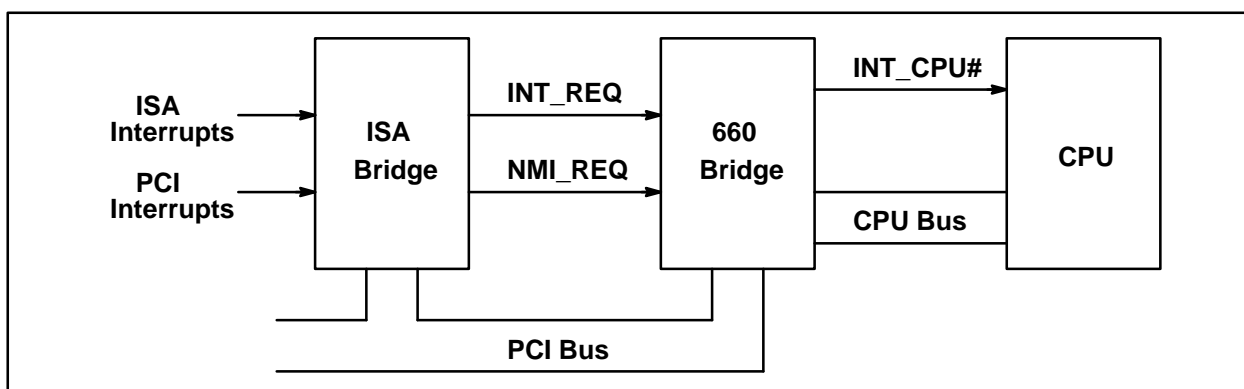


Figure 8-2. Interrupt Handling

As shown in Figure 8-2, the 100 MHz PPC 603e MCM interrupts are routed to the interrupt controller located inside the ISA bridge. When a device signals an interrupt (which is not masked in the interrupt controller), then for error sources, the interrupt controller is pro-

grammed to assert NMI\_REQ. If the input to the interrupt controller signals an interrupt, then:

1. The ISA bridge asserts INT\_REQ to the 660.
2. The 660 asserts INT\_CPU# to the CPU.
3. The CPU recognizes the interrupt signal (INT#) immediately (or as soon as the MSR(EE) interrupt enable bit in the CPU is set to 1), saves its state, and then takes a precise external interrupt exception, branching to either 500h or FFF0 0500h, depending upon the Exception Prefix (EP) bit in the MSR. The MSR(EE) bit is automatically set to 0 at this time.
4. The code at the vector location requests a single-byte read of memory address BFFF FFF0h.
5. In response to the read, the 660 arbitrates for the PCI bus and then generates an interrupt acknowledge transaction on the PCI bus.
6. The ISA bridge decodes and claims the PCI interrupt acknowledge transaction, and returns the 8-bit vector which has been preprogrammed for the active interrupt, and then negates the interrupt output.
7. The 660 accepts the interrupt vector on the PCI bus, returns it to the CPU, and signals TA# to terminate the CPU transfer normally.

Since the CPU does not require that the interrupt signal (INT\_CPU#) be deactivated between interrupts, another interrupt is allowed to occur as soon as software sets the MSR(EE) bit back to 1. For this reason, software should enable interrupts as soon as possible after receiving the vector. Note that the load instruction that fetches the interrupt vector is subject to out-of-order execution; eieio as required. After servicing the interrupt, execute a return from interrupt (RFI) instruction to return to the program that was interrupted. For more information on interrupts, see the Exceptions section of the 603 User's Manual.

Note that other PCI busmasters can initiate interrupt acknowledge transactions, but this may have unpredictable effects.

### **8.1.6 Planar Interrupt Assignments**

In general, program ISA interrupts as edge sensitive. Program PCI interrupts as level sensitive. Interrupts are assigned to priority levels per ISA conventions. Table 8-1 shows the interrupt assignments of the planar. IRQ[0:7] connect to the master controller, and IRQ[8:15] connect to the cascaded controller. Figure 8-3 shows the connection of the PCI interrupts.

Table 8-1. Mapping of PCI Memory Space, Part 1

SIO IRQ #	Connects to	Priority	Assignment or (Comment)
0	no pin	1	Timer 1 Counter 0 (Internal to SIO).
1	Sys I/O EPLD	2	Keyboard
2	no pin	(3-10)	Cascade from controller 2
3	ISA IRQ3	11	(COM 2 or COM 4)
4	ISA IRQ4	12	(COM 1 or COM 3)
5	ISA IRQ5	13	(Parallel LPT 1 or 2)
6	ISA IRQ6	14	(Floppy)
7	ISA IRQ7	15	(Parallel LPT 2 or 3)
8#	RTC	3	TOD (aka Real Time Clock)
9	ISA IRQ9	4	
10	ISA IRQ10	5	(Audio)
11	ISA IRQ11	6	
12/M	ISA IRQ12	7	(Mouse)
13/FERR	—	8	Pulled up.
14	ISA IRQ14	9	(IDE)
15	ISA IRQ15	10	
PIRQ2#	PCI_INTC#		See Figure 8-3.
PIRQ1#	PCI_INTB#		See Figure 8-3.
PIRQ0#	PCI_INTA#		See Figure 8-3.

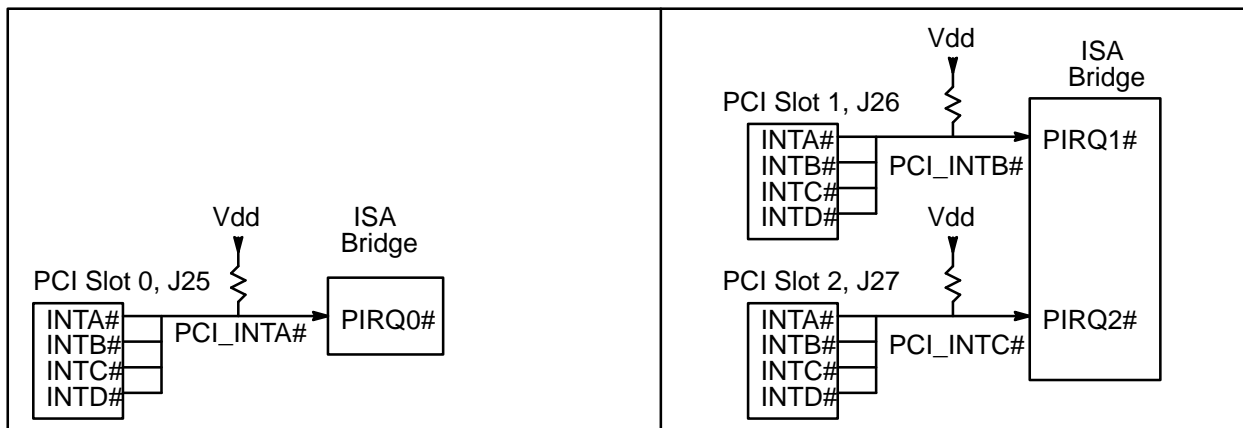


Figure 8-3. PCI Interrupt Connections

### 8.1.7 Scatter-Gather Interrupts

Where possible, set up the scatter-gather function to use the ISA bridge end of process (EOP output) indicator for the termination of ISA bus DMA in which scatter-gather is employed. The planar is initially configured to use this scheme. The EOP signal from the ISA bridge is used as the terminal count (ISA\_TC) signal on the ISA bus.



## **8.2 Errors**

Errors are handled primarily by the 660, the CPU, and the firmware. The errors are reported to the CPU or the PCI and status information is saved in the 660 register set so that error type determination can be done by the CPU. There are two methods which the MCM (660) uses to report errors to the CPU, the TEA# method, and the MCP# method. Errors related to a PCI bus transaction are reported to the PCI by means of the PCI\_PERR# or the PCI\_SERR# signals.

Errors that are detected while the CPU is running a cycle that can be terminated immediately are reported using TEA#. Errors reported in this way are a direct result of the CPU transfer that is currently in progress. For example, when the 660 detects a transfer size error, it terminates the CPU transfer with TEA# instead of with TA#.

Errors that are detected while a CPU transfer is not in progress, and errors that occur because of a CPU transfer but which are detected too late to be reported using TEA#, and errors that are not a direct result of the current CPU transfer, are reported using MCP#. For example, memory parity errors occurring while a PCI busmaster is accessing memory are reported using MCP#.

There are three separate data error checking systems in the 660; CPU bus, memory, and PCI bus. The 660 does not generate or check CPU address bus parity.

Each error that can be detected has an associated mask. If the error is masked, then the detection of that error condition is disabled. There are also assertion masks for the MCP#, TEA#, and PCI\_SERR# signals that prevent reporting of any error by means of that signal (these masks do not affect the detection of the error).

Once an error is detected and the appropriate status, address, and control information is saved, the detection of all subsequent error detection is disabled until the current error is reset. For more information on error handling, see the 660 User's Manual.

### **8.2.1 CPU Bus Related Errors**

#### **8.2.1.1 CPU Bus Error Types**

- **Errors Reported With TEA# (cycle still active)**
  - CPU bus unsupported transfer type
  - CPU bus unsupported transfer size
  - CPU bus XATS# asserted
- **Errors Reported With MCP# (cycle has ended)**
  - CPU data bus parity error
  - CPU bus write to locked flash
  - CPU bus memory select error
  - Memory parity error during CPU to memory transfer
  - Memory single-bit ECC error trigger exceeded during CPU to memory transfer
  - Memory multi-bit ECC error during CPU to memory transfer
  - L2 cache parity error
  - PCI bus data parity error (660 is PCI busmaster) during CPU to PCI transaction
  - PCI target abort received (660 is PCI busmaster) during CPU to PCI transaction
  - PCI master abort generated (660 is PCI busmaster) during CPU to PCI transaction.

**8.2.1.2 CPU Bus Error Handling Protocol**

If the error is masked, do not detect the error.

If the error is detected, perform the following steps:

1. Set status bit indicating error type.
2. Set status bit indicating error during CPU cycle.
3. Save CPU address and control bus values.
4. Report error to the CPU. (Reported by means of TEA# if the CPU cycle is still active or by means of MCP# if the CPU cycle has ended.)

PCI bus data parity errors also cause PCI\_PERR# to be asserted.

There is a status bit (PCI Status Register bit 15) that is set whenever any type of PCI bus parity error is detected. The setting of this status bit is not maskable.

**8.2.2 PCI Bus Related Errors****8.2.2.1 PCI Bus Error Types**

During a PCI to memory transaction (in which the 660 is the PCI target):

- PCI bus address parity error
- PCI bus data parity error
- PCI memory select error
- Memory parity error
- Memory single-bit ECC error trigger exceeded
- Memory multi-bit ECC error

**8.2.2.2 PCI Bus Error Handling Protocol**

If the error is masked, the 660 does not detect the error. If the error is detected, perform the following steps.

1. Set status bit indicating error type.
2. Set status bit indicating error during PCI cycle.
3. Save PCI address and control bus values.
4. Report error to the PCI. If the error is a PCI bus data parity error then report by means of PCI\_PERR#. If the error is not a data parity error then report by means of PCI\_SERR#.
5. If the PCI cycle is still active (not the last data phase), then target abort the cycle.

The 660 can be enabled to report PCI bus data parity errors with PCI\_SERR#. This method should only be used if it is determined that PCI\_PERR# is not supported by some (or all) of the PCI masters in the system.

**8.2.2.3 PCI Bus Data Parity Errors**

While the 660 is the PCI busmaster (during CPU to PCI transactions):

- During reads, the 660 monitors the PCI\_AD (and C/BE# and PCI\_PAR) lines to detect data parity errors during the data phases. If an error is detected, the 660 asserts PCI\_PERR#. Unless masked, the 660 will report the error to the CPU bus using MCP#. This error does not cause the 660 to alter the PCI transaction in any way.
- During writes, the 660 monitors PCI\_PERR# to detect data parity errors that are detected by the target. Unless masked, the 660 will report the error to the CPU bus using MCP#. This error does not cause the 660 to alter the PCI transaction in any way.

While the 660 is the PCI target (during PCI to memory transactions):

- During reads, the 660 does not monitor PCI\_PERR#, and so will not detect a data parity error.
- During writes, the 660 monitors the PCI\_AD (and C/BE# and PCI\_PAR) lines to detect data parity errors during the data phases. If an error is detected, the 660 asserts PCI\_PERR#. The 660 will not report the error to the CPU bus. This error does not cause the 660 to alter the PCI transaction in any way.

### 8.2.3 CPU Bus Transaction Errors

#### 8.2.3.1 CPU Bus Transfer Type or Size Error

This error is generated when the CPU generates a bus operation that is not supported by the 660 (see Table 8-2). An error is not generated if the cycle is claimed by another CPU device (CPU\_BUS\_CLAIM# asserted).

**Table 8-2. Invalid CPU Bus Operations**

TT[0:4]	Operation
1000x	Reserved
1011x	Reserved

Only the following transfer sizes are supported. Other transfer sizes are not supported.

- 1-byte to 8-byte single-beat reads or writes to memory within an 8-byte boundary
- Burst reads or writes to memory (32 bytes, aligned to double-word)
- 1-byte to 4-byte single-beat reads or writes to the PCI bus that do not cross a 4-byte boundary
- 8-byte single-beat writes to the PCI bus within an 8-byte boundary
- All accesses not to memory or PCI with sizes of 1 to 4 bytes within a 4-byte boundary
- ROM reads support the same sizes as memory.

Transfer type and size errors can be controlled by the indexed register set. The mask is at register C0h bit 0. If an error is detected, status bits at register C1h bits 1:0 are set to 10. Register C7h bit 4 is cleared to indicate an error on a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5. Transfer type and size errors are reset by writing a 1 to register C1h bit 0 or register C1h bit 1. The indexed register set uses the same mask and error reset bits for XATS# that it uses for unsupported transfer type errors.

Transfer type and size errors can also be controlled by the 650-compatible register set. The mask cannot be controlled by means of this register set. If an error is detected, the status bit at 8000 0844h bit 0 is cleared. The address is saved at BFFF EFF0h. This error can be reset by reading BFFF EFF0h. Note that the 650-compatible register set does not differentiate between XATS# errors and unsupported transfer type errors.

#### 8.2.3.2 CPU Bus XATS# Asserted Error

This error is generated when the CPU asserts the XATS# signal.

The XATS# error can be controlled by the indexed register set. The mask is at register C0h bit 0. If an error is detected, the status bits at register C1h bits 1:0 are set to 01. Register C7h bit 4 is cleared to indicate an error on a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5. This error can be reset by writing a 1 to register C1h bit 0 or register C1h bit 1.

The indexed register set uses the same mask and error reset bits for XATS# and for unsupported transfer types.

This error can also be controlled by the 650-compatible register set. The mask cannot be controlled by means of this register set. If an error is detected, the status bit at 8000 0844h bit 0 is cleared. The address is saved at BFFF EFF0h. This error can be reset by reading BFFF EFF0h. The 650-compatible register set does not differentiate between XATS# errors and unsupported transfer type errors.

#### **8.2.3.3 CPU to Memory Writes**

During CPU to memory writes, the CPU drives data parity information onto the CPU data bus. Correct parity is then generated in the 660 and written to DRAM memory along with the data. The L2 SRAM is updated (when required) with the data and the parity information that the CPU drove onto the CPU data bus.

During CPU to memory writes, the 660 checks the data parity sourced by the CPU, and normally reports any detected parity errors via TEA#.

#### **8.2.3.4 CPU to Memory Reads**

The MCM is initially configured to check memory data using parity. However, the 660 can be programmed to execute an error checking and correction (ECC) algorithm on the memory data, by generating ECC check bits during memory writes, and checking-correcting the data during memory reads. ECC can be implemented using normal parity DRAM.

Note that for each memory read operation, eight bytes of memory are read, and parity on eight bytes is checked regardless of the transfer size. Therefore, all of memory must be initialized (at least up to the end of any cache line that can be accessed). For the same reasons, memory must be initialized while ECC memory data error checking is in use.

When the CPU reads from memory, the data and accompanying parity information can come from either the L2 SRAM or from DRAM memory. If the data is sourced from the L2, the parity information also comes from the L2.

If the data is sourced by memory, the parity information also comes from memory. The L2 SRAM is updated (when required) using the data and parity from memory.

During CPU to memory reads, the 660 samples the DPE# output of the CPU to determine parity errors, and reports them back to the CPU via MCP#. The particular memory read data beat will be terminated normally with TA#.

#### **8.2.3.5 CPU Data Bus Parity Error**

This error is generated when a parity error on the CPU data bus is detected during a transfer between the CPU and the 660. The full CPU data bus is always checked for parity regardless of which bytes lanes actually carry valid data. The parity is odd, which means that an odd number of bits, including the parity bit, are driven high. The 660 directly checks the parity during CPU write cycles. The 660 detects CPU bus parity errors by sampling the DPE# signal from the CPU during CPU read cycles.

This error is also generated when an L2 cache data parity error (see section 8.2.3.6) is detected.

CPU\_DPAR[0] indicates the parity for CPU\_DATA[0:7]. CPU\_DPAR[1] indicates the parity for CPU\_DATA[8:15] and so on.

This error can be controlled by the indexed register set. The mask is at register C4h bit 2. If an error is detected, the status bit at register C5h bit 2 is set. Register C7h bit 4 is cleared to indicate an error on a CPU cycle. The CPU address is saved in register C8h, the CPU control is saved in register C3h, and the CPU number is saved in register C7h bit 5. This error can be reset by writing a 1 to register C5h bit 2.

This error cannot be controlled by means of the 650-compatible register set.

#### **8.2.3.6 L2 Cache Parity Error**

This error is generated when a parity error is detected during a CPU read from the L2 cache. The parity is checked by the CPU which drives DPE# to the 660. When this error is detected, the 660 indicates both this error and a CPU bus data parity error.

This error can be controlled by the indexed register set. The mask is at register C4h bit 3. If an error is detected, the status bit at register C5h bit 3 is set. Register C7h bit 4 is cleared to indicate error on a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5. This error can be reset by writing a 1 to register C5h bit 3.

This error can also be controlled by means of a register in the 650-compatible register set. The mask cannot be controlled by means of this register set. If an error is detected, the status bits at 8000 0842h bit 0 and 8000 0843h bit 0 is cleared. The address is not saved in a 650-compatible register (register BFFF EFF0h is undefined). This error can be reset by reading 8000 0843h.

#### **8.2.3.7 CPU Bus Write to Locked Flash**

This error is generated when the CPU attempts to write to flash memory when write to flash ROM has been disabled (locked out). If the flash ROM is directly attached to the 660 (see configuration strapping), CPU writes to FFFF FFF0h are detected as an error if writing has been locked out by means of 660 compatible register FFFF FFF1h (see note in Sections 6.1.2 and 6.1.2.2). If the flash is remotely attached then CPU writes to the 4G – 2M to 4G address space are detected as an error if writing has been locked out by means of register BBh bit 0.

This error can be controlled by the indexed register set. The mask is at register C4h bit 0. If an error is detected, the status bit at register C5h bit 0 is set. Register C7h bit 4 is cleared to indicate error on a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5. This error can be reset by writing a 1 to register C5h bit 0.

This error cannot be controlled by means of the 650-compatible register set.

### **8.2.4 CPU to PCI Bus Transaction Errors**

#### **8.2.4.1 PCI Bus Data Parity Error While PCI Master**

This error is generated when a PCI bus data parity error is detected during a CPU to PCI transaction. The 660 checks parity during read cycles and samples PCI\_PERR# during write cycles. The bridge asserts PCI\_PERR# if a parity error is detected on a read cycle. The PCI bus uses even parity, which means that an even number of bits including the parity bit are driven high.

This error can be controlled by the indexed register set. The mask is at register 04h bit 6. If an error is detected, the status bit at register 06h bit 8 is set. Register C7h bit 4 is cleared

to indicate an error on a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5. This error can be reset by writing a 1 to register 06h bit 8.

When this error is detected, the status bit at register 06h bit 15h is set, regardless of the state of the mask at register 04h bit 6. The status bit at register 06h bit 15 is set by all types of PCI bus parity errors. This bit is cleared by writing a 1 to register 06h bit 15.

This error cannot be controlled by means of the 650-compatible register set.

Unless masked, the 660 will report this error to the CPU as a PCI bus data parity error while PCI master, using MCP#.

#### **8.2.4.2 PCI Target Abort Received While PCI Master**

This error is generated when a target abort is received on the PCI bus during a cycle which is mastered by the 660 for CPU access to the PCI bus. The CPU cycle is terminated with a TEA#, the Error Address register is held, and the Illegal Transfer Error register is set.

This error can be controlled by the indexed register set. The mask is at register C0h bit 7. If an error is detected, the status bit at register 06h bit 12 is set. Register C7h bit 4 is cleared to indicate an error on a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5. This error can be reset by writing a 1 to register 06h bit 12.

This error cannot be controlled by means of the 650-compatible register set.

#### **8.2.4.3 PCI Master Abort Detected While PCI Master**

This error is generated when a master abort is detected on the PCI bus during a cycle which is mastered by the 660 for CPU access to the PCI bus. Master aborts occur when no target claims a PCI memory or I/O cycle—PCI\_DEVSEL# is never asserted.

The 660 master aborts if no agent responds with DEVSEL# within eight clocks after the start of a CPU to PCI cycle. The cycle is ended with a TEA# response to the CPU, all 1's data is returned on reads, the Illegal Transfer Error register is set, and the Error Address register is held.

Note that some operating systems intentionally access unused addresses to determine what devices are located on the PCI bus. These operating systems do not expect an error to be generated by these accesses. When using such an operating system it is necessary to leave this error masked.

This error can be controlled by the indexed register set. The mask is at register C4h bit 4. If an error is detected, the status bit at register 06h bit 13 is set. Register C7h bit 4 is cleared to indicate an error on a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5. This error can be reset by writing a 1 to register 06h bit 13.

This error cannot be controlled by means of the 650-compatible register set.

The 660 also checks for bus hung conditions. If a CPU to PCI cycle does not terminate within approximately 60 usec. after the PCI is owned by the CPU, the cycle is terminated with TEA#. This is true for all CPU to PCI transaction types except configuration transactions. This feature may be disabled via a 660 control register.

In the case of configuration cycles that do not receive a DEVSEL# (no device present at that address), the PCI cycle is master aborted, and TA# (normal response) is returned.

Write data is thrown away and all 1's are returned on read cycles. No error register is set and no address is captured in the error address register.

### **8.2.5 PCI to Memory Transaction Errors**

#### **8.2.5.1 PCI to Memory Writes**

During PCI to memory writes, the 660 generates the data parity that is written into DRAM memory. The bridge also checks the parity of the data, and asserts PCI\_PERR# if it detects a data parity error.

#### **8.2.5.2 PCI to Memory Reads**

During PCI to memory reads, the 660 checks the parity of the memory data, and then generates the data parity that is driven onto the PCI bus. If there is a parity error in the data/parity returned to the 660 from the DRAM, the bridge drives PCI\_PAR incorrectly to propagate the parity error (and also reports the error to the CPU via MCP#). The data beat with the bad parity is not target aborted because doing so would slow all data beats for one PCI clock (TRDY# is generated before the data is known good). However, if the agent is bursting and there is another transfer in the burst, the next cycle is stopped with target abort protocol. During PCI to memory reads, the 660 also samples the PCI\_PERR# signal, which other agents can be programmed to activate when they detect a PCI parity error.

#### **8.2.5.3 Out of Bounds PCI Memory Accesses**

If a PCI busmaster runs a cycle to a system memory address above the top of physical memory, no one will respond, and the initiator master aborts the cycle. The initiating busmaster must be programmed to notify the system of master aborts as needed. The system logic does not notify the CPU.

#### **8.2.5.4 PCI Address Bus Parity Error While PCI Target**

This error is generated when a parity error is detected during the address phase of a PCI access where the 660 is the PCI target of a PCI access to system memory.

This error can be controlled by the indexed register set. The mask is at register 04h bit 6. This error does not have an explicit status bit to indicate its occurrence. However, the following status bits are set:

- Register 06h bit 14 is set to indicate that PCI\_SERR# has been asserted. This bit is cleared by writing a 1 to register 06h bit 14. (BCR04 b8 must be 1 to enable SERR# assertion due to PCI bus address parity errors.)
- Register 06h bit 11 is set to indicate signalled target abort if the cycle was target aborted. This bit is cleared by writing a 1 to register 06h bit 11.
- Register 06h bit 15 is set to indicate a PCI bus parity error regardless of the state of the mask at register 04h bit 6. Note that this bit is set by all types of PCI bus parity errors. This bit is cleared by writing a 1 to register 06h bit 15.

Register C7h bit 4 is set to indicate an error on a PCI cycle. The PCI address is saved in register C8h. The PCI control is saved in register C7h.

This error cannot be controlled by means of the 650-compatible register set.

### 8.2.5.5 PCI Bus Data Parity Error While PCI Target

This error is generated when a PCI bus data parity error is detected during a PCI to memory write transaction. The PCI bus uses even parity, which means that an even number of bits including the parity bit are driven high.

This error can be controlled by means of the registers in the indexed register set. The mask is at register 04h bit 6. This error does not have an explicit status bit to indicate its occurrence. However, the following status bits are set:

- Register 06h bit 11 is set to indicate signalled target abort if the cycle was target aborted. This bit is cleared by writing a 1 to register 06h bit 11.
- Register 06h bit 15 is set to indicate a PCI bus parity error regardless of the state of the mask at register 04h bit 6. Note that this bit is set by all types of PCI bus parity errors. This bit is cleared by writing a 1 to register 06h bit 15.
- Register 06h bit 14, which indicates PCI\_SERR#, is set if the mask at register C0h bit 6 is disabled (cleared). Note that the mask at register C0h bit 6 allows the 660 to signal PCI\_SERR# in addition to PCI\_PERR# for this error. This bit is cleared by writing a 1 to register 06h bit 14.

Register C7h bit 4 is set to indicate an error on a PCI cycle. The PCI address is saved in register C8h. The PCI control is saved in register C7h.

This error cannot be controlled by means of the 650-compatible register set.

During PCI to memory reads, the 660 does not monitor PCI\_PERR# to detect data parity errors. Therefore this error is never generated during PCI to memory reads.

This error is not reported to the CPU.

### 8.2.5.6 Errant Masters

Both PCI and ISA masters can access certain planar and ISA bridge registers. For example, various control registers such as the I/O Map Type register, the BE/LE mode bit, the Memory Control registers, etc. are accessible. Faulty code in the PCI or ISA masters can defeat password security, read the NVRAM, and cause the system to crash without recovery. Take care when writing device drivers to prevent these events.

## 8.2.6 Memory Transaction Errors

### 8.2.6.1 Memory Select Error

This error is generated if a device addresses the system memory space (CPU addresses from 0 to 2G and PCI addresses from 2G to 4G) when memory is not present at that address. The 660 only claims PCI accesses by asserting PCI\_DEVSEL# when the access is to an address where memory is present.

The 660 disconnects PCI burst cycles at 1M boundaries. This ensures that a PCI master cannot begin a transfer at an address where memory is present and then burst (incrementing the address) to an address where memory is not present; therefore, the memory select error is never generated on PCI accesses to system memory.

The memory select error can be controlled by the indexed register set. The mask is at register C0h bit 5. If an error is detected, the status bit at register C1h bit 5 is set. Register C7h bit 4 is cleared to indicate an error on a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5. This error can be reset by writing a 1 to register C1h bit 5.



This error cannot be controlled by means of the 650-compatible register set.

#### **8.2.6.2 System Memory Parity Error**

When memory is being operated in parity mode, this error is generated if a parity error is detected during a read from system memory. Memory parity is odd, which means that an odd number of bits including the parity bit are driven high.

MEM\_CHECK[0] indicates the parity for MEM\_DATA[7:0]. MEM\_CHECK[1] indicates the parity for MEM\_DATA[15:8] and so on.

The system memory parity error can be controlled by the indexed register set. The mask is at register C0h bit 2. If an error is detected, the status bit at register C1h bit 2 is set.

If the parity error occurred while the CPU was accessing memory, then register C7h bit 4 is cleared to indicate the error occurred during a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5.

If the parity error occurred while the PCI was accessing memory, then register C7h bit 4 is set to indicate the error occurred during a PCI cycle. The PCI address is saved in register C8h. The PCI control is saved in register C7h.

This error can be reset by writing a 1 to register C1h bit 2. Note that register locations listed above are used to indicate single-bit ECC errors if the memory is being operated in ECC mode.

This error can also be controlled by means of the register in the 650-compatible register set. The mask cannot be controlled by means of this register set. If an error is detected, the status bit at 8000 0840h bit 0 is cleared. The address is saved at BFFF EFF0h. This error can be reset by reading BFFF EFF0h.

#### **8.2.6.3 System Memory Single-Bit ECC Error**

When memory is being operated in ECC mode, single-bit errors are detected and corrected. Since single-bit errors are corrected, generally no error reporting is necessary. But when a single-bit error is detected, the single-bit error counter register (register B8h) is incremented and the system memory address is saved in the single-bit ECC error address register (register CCh). If the count in the single-bit error counter register exceeds the value in the single-bit error trigger level register (B9h), the 660 generates a single-bit ECC trigger exceeded error.

The trigger exceeded error can be controlled by the indexed register set. The mask is at register C0h bit 2. If an error is detected, the status bit at register C1h bit 2 is set.

If the error occurs while the CPU is accessing memory, register C7h bit 4 is cleared to indicate the error occurred during a CPU cycle. The CPU address is saved in register C8h. The CPU control is saved in register C3h and the CPU number is saved in register C7h bit 5.

If the error occurs while the PCI is accessing memory, register C7h bit 4 is set to indicate the error occurred during a PCI cycle. The PCI address is saved in register C8h. The PCI control is saved in register C7h.

This error can be reset by writing a 1 to register C1h bit 2. Note that register locations listed above are used to indicate memory parity errors if the memory is being operated in parity mode.

This error cannot be controlled by means of the 650-compatible register set.

#### 8.2.6.4 System Memory Multi-Bit ECC Error

When memory is being operated in ECC mode, this error is generated if a multi-bit ECC error (uncorrectable) is detected during a read from system memory.

The multi-bit ECC error can be controlled by the indexed register set. The mask is at register C0h bit 3. If an error is detected, the status bit at register C1h bit 3 is set.

If the error occurs while the CPU is accessing memory, then register C7h bit 4 is cleared to indicate the error occurred during a CPU cycle. The CPU bus address is saved in register C8h. The CPU control is saved in register C3h, and the CPU number is saved in bit 7 of register C7h.

If the error occurs while the PCI is accessing memory, then register C7h bit 4 is set to indicate the error occurred during a PCI cycle. The PCI address is saved in register C8h. The PCI control is saved in register C7h.

This error can be reset by writing a 1 to register C1h bit 3.

This error cannot be controlled by means of the 650-compatible register set.

#### 8.2.7 SERR, I/O Channel Check, and NMI Errors

The PCI bus defines a signal called SERR# which any agent can pulse. This signal is to report error events within the devices, not bus parity errors. The signal is wired to the ISA bus bridge on the planar. The ISA bus signal IOCHCK is also wired to the ISA bridge. If either of these lines activate, the ISA bridge asserts NMI to the 660 unless the condition is masked by a register within the ISA bridge. The NMI signal causes the 660 to generate an interrupt to the CPU, and to assert MCP# to the CPU. The ISA bridge contains status registers to identify the NMI source. Software may interrogate the ISA bridge and other devices to determine the source of the error.

##### 8.2.7.1 NMI\_REQ Asserted Error

This error is generated when the NMI input is sampled asserted by the 660. External logic can assert this signal for any type of catastrophic error it detects. The external logic should also assert this signal if it detects PCI\_SERR# asserted. The 660 does not treat NMI\_REQ as an interrupt, but as an error indicator.

NMI is handled somewhat differently from the bus related error sources.

- There are no 660 BCRs associated with NMI\_REQ. The external logic that asserted NMI to the 660 provides mask and status information.
- The NMI\_REQ input contains no edge detection logic. The 660 has no memory of any previous state of NMI\_REQ.
- In general, the assertion of NMI\_REQ has no effect on any other processes in the 660.
- MCP# is generally asserted continuously while NMI\_REQ is sampled valid, however:
  - If an error was detected before the NMI\_REQ was detected, then the error handling logic will not sample the NMI\_REQ input (and thus will not detect it) until the previous error is cleared using the appropriate BCRs.
  - If the NMI\_REQ is deasserted before the previous error is cleared, the NMI\_REQ will be lost.
  - If NMI\_REQ is still asserted when the previous error is cleared, then the NMI\_REQ will be sampled asserted, and MCP# will begin to be asserted.

- The 660 will not assert MCP# while NMI\_REQ is active unless MCP# assertion is enabled in BCR(BA) bit 0. As before, if NMI\_REQ is active when MCP# assertion is enabled, then MCP# will be asserted.
- Unlike with the bus related error sources, when the 660 samples NMI\_REQ valid, it does not disable further error detection. Thus PCI and CPU bus related errors will still be detected and handled in the normal fashion. However, if the detected bus related error causes MCP# to be asserted (for 2 CPU clocks) then at the end of the second CPU clock, MCP# will be and remain deasserted until the current error is cleared using the appropriate BCRs, even if NMI\_REQ is still asserted.

### **8.2.8 Error Reporting Protocol**

In general, when the 660 recognizes an error condition, it sets various status BCRs, saves address and control information (for bus related errors), disables further error recognition (until the current error is cleared), and reports the error to either the CPU or PCI bus.

Unless otherwise noted, the 660 takes no further error handling action, but relies on the CPU/software or PCI agent to take the next step in the error handling procedure. The 660 continues to react appropriately to CPU and PCI bus traffic, the state of the memory controller is unchanged, current and pipelined CPU and PCI transactions are unaffected, and the behavior and state of the 660 is unaffected.

For example, if a memory parity error is reported to the CPU using MCP#, and the CPU does not respond to the MCP#, then the 660 will in all ways continue to behave as if the MCP# had not been asserted. However, various BCRs will contain the error status and address information, and further error recognition will be disabled until the CPU resets the error in the 660 BCRs.

#### **8.2.8.1 Error Reporting With MCP#**

The following errors are reported to the CPU using MCP#:

- NMI errors,
- Errors that occur because of a CPU transfer but which are detected too late to be reported using TEA#, and
- Errors that are not a direct result of the current CPU transfer.

The 660 reports an error with MCP# by asserting MCP# to the CPU bus for 2 CPU clocks. The 660 does not itself take any other action. All current and pipelined CPU and PCI bus transactions are unaffected. The state of the memory controller is unaffected. The assertion of MCP# does not cause any change in the behavior or state of the 660.

#### **8.2.8.2 Error Reporting With TEA#**

CPU bus related errors that are detected while the CPU is running a cycle that can be terminated immediately are reported using TEA#. Errors reported in this way are a direct result of the CPU transfer that is currently in progress. For example, when the 660 detects a transfer size error, it terminates the CPU transfer with TEA# instead of with TA#.

The 660 reports an error with TEA# by asserting TEA# to the CPU in accordance with the PowerPC bus protocol. The data beat on which TEA# is asserted becomes the final data beat. The 660 does not itself take any other action. All other current and pipelined CPU and PCI bus transactions are unaffected. The state of the memory controller is unaffected. The assertion of TEA# does not cause any other change in the behavior or state of the 660.

**8.2.8.3 Error Reporting With PCI\_SERR#**

The 660 asserts PCI\_SERR# for one PCI clock when a non-data-parity error (system error) is detected and the 660 is a PCI target. As is the case with the other error reporting signals, the 660 may perform various operations in response to a detected error condition, but it does not automatically perform further actions just because it asserts PCI\_SERR#

The 660 does not monitor PCI\_SERR# as driven by other PCI agents. PCI\_SERR# is not an input to the 660.

**8.2.8.4 Error Reporting With PCI\_PERR#**

The 660 asserts PCI\_PERR# for one PCI clock to report PCI bus data parity errors that occur while the 660 is receiving data; during PCI to memory writes and CPU to PCI reads. The 660 asserts PCI\_PERR# in conformance to the PCI specification.

**8.2.9 Error Status Registers**

Error status registers in the 660 may be read to determine the types of outstanding errors. Errors are not accumulated while an error is outstanding; however, there will be one TEA# or MCP# for each error that occurs. For example, if an illegal transfer error causes a TEA#, a memory parity error can occur while the CPU is processing the code that handles the TEA#. The second error can occur before the error status registers are read. If so, then the second error status is not registered, but the MCP# from the memory parity error is asserted.

**8.2.10 Error-Related Bridge Control Registers**

Information on these registers is contained in the *660 Bridge Manual*.

Bridge Control Register	Index	R/W	Bytes
System Control 81C	8000 081C	R/W	1
Memory Parity Error Status	8000 0840	R	1
L2 Error Status	8000 0842	R	1
L2 Parity Error Read and Clear	8000 0843	R	1
Unsupported Transfer Type Error	8000 0844	R	1
System Error Address	BFFF EFF0	R	4
PCI Command	Index 04 – 05	R/W	2
PCI Device Status	Index 06 – 07	R/W	2
Single-Bit Error Counter	Index B8	R/W	1
Single-Bit Error Trigger Level	Index B9	R/W	1
Bridge Options 1	Index BA	R/W	1
Error Enable 1	Index C0	R/W	1
Error Status 1	Index C1	R/W	1
CPU Bus Error Status	Index C3	R	1
Error Enable 2	Index C4	R/W	1
Error Status 2	Index C5	R/W	1
PCI Bus Error Status	Index C7	R/W	1
CPU/PCI Error Address	Index C8 – CB	R/W	4
Single-Bit ECC Error Address	Index CC – CF	R/W	4
Bridge Chip Set Options 3	Index D4	R/W	1

**8.2.11 Special Events Not Reported as Errors**

- A PCI to memory cycle at any memory address above that programmed into the top of memory register.

The 660 ignores this cycle and the initiator master aborts. No data is written into system memory on writes, and the data returned on reads is indeterminate. The busmaster must be programmed to respond to a target abort by alerting the host.

- CPU to PCI configuration cycles to which no device responds with a DEVSEL# signal within 8 clocks (no device at this address)

The data returned on a read cycle is all 1's and write data is discarded. This allows software to scan the PCI at all possible configuration addresses, and it is also consistent with the PCI specification.

- A CPU read of the IACK address having a transfer size other than 1, or having other than 4-byte alignment.

These conditions return indeterminate data. The ISA bridge requires the byte enables, CBE#3:0, to be 1110 in order to place the data on the correct byte lane (0). Accesses other than one byte at the address BFFF FFF0h are undefined.

- A read of the IACK address when no interrupt is pending

A DEFAULT 7 vector is returned in this case. This is the same vector that is returned on spurious interrupts.

- Parity error in Flash/ROM.

Parity is not stored in the Flash ROM. Therefore the memory parity error signal and the DPE signal are ignored during ROM reads. The Flash or ROM should include CRC with software checking to insure integrity.

- Write to Flash with TSIZ other than 4.

This will cause indeterminate data to be written into the Flash at an indeterminate address.

- Caching ROM space.

An L1 or CB-L2 cast out will cause indeterminate results.

- Running any cycle to the PCI configuration space with an undefined address.

Some of these could potentially cause damage. See the warning under the PCI configuration cycle section.

- Accessing any ISA device with the wrong data size for that device.

Indeterminate results will occur.

### 8.3 Resets

The RESET# pin of the 664 must be asserted to initialize the 660 before proper operation commences. The 663 does not have a reset pin. Since the operation of the 663 is controlled by the 664, the entire 660 bridge will be properly initialized by the proper assertion of RESET#.

#### 8.3.1 CPU Reset

For information of CPU SRESET# and HRESET#, see the 603e user manual.

#### 8.3.2 660 Reset

The entire 660 is reset to the correct initial state by the proper assertion of the RESET# input of the 664 (POWER\_GOOD/RESET#). The following rules must be followed to ensure correct operation:

1. RESET# must be asserted for at least eight CPU consecutive CPU clocks. This is the minimum RESET# pulse width.
2. Both the CPU and PCI clocks must be running properly during the entire reset interval.
3. Bus activity on the PCI bus must not begin until at least 4 CPU clocks after the deassertion of RESET#.
4. Bus activity on the CPU bus also must not begin until at least 4 CPU clocks after the deassertion of RESET#.
5. Assertion and deassertion of RESET# can be asynchronous for normal operation, but if deterministic operation is required, see section 8.3.2.3.

All 660 outputs reach their reset state by the second CPU clock after RESET# is first sampled active. The rest of the minimum RESET# pulse width is used by the 660 to initialize internal processes, including setting internal registers and determining the CPU to PCI clock ratio.

Except as noted in section 8.3.2.1, all 660 outputs maintain their reset state until an external stimulus (CPU bus activity) forces them to change.

##### 8.3.2.1 Reset State of 660 Pins

The following symbols are used in Table 8-3 and Table 8-4:

—	means the signal is an input. The signal does not have a required state during reset.
Z	means that the pin is tristate (hi-Z) during reset,
U	means the state of the pin during reset is undefined,
1	means that the pin is driven to a logic 1 state (hi)
0	means that the pin is driven to a logic 0 state (low)

Table 8-3. 664 Pin Reset State

664 Signal	State
AACK#	Z
AOS_RR_MMRS	1
ARTRY#	Z
C2P_WRL_OPEN	1
CAS[7:0]#	1
CPU_ADDR[0:31]	Z
CPU_BUS_CLAIM#	—
CPU_CLK	—
CPU_DATA_OE#	1
CPU_GNT1#	1
CPU_GNT2#	1
CPU_PAR_ERR#	—
CPU_RDL_OPEN	1
CPU_REQ1#	—
CPU_REQ2#	—
CRS_C2PWXS	Z
DBG#	0
DPE#	—
DUAL_CTRL_REF	1
ECC_LE_SEL	1
GBL#	Z
IGN_PCI_AD31	—
INT_CPU#	INT_REQ#
INT_REQ	—
MA[11:0]	1
MCP#	Z

664 Signal	State
MEM_BE[3:0]	1
MEM_DATA_OE#	1
MEM_ERR#	—
MEM_RD_SMPL	1
MEM_WRL_OPEN	0
MIO_TEST	0
MWS_P2MRXS	Z
NMI_REQ	—
PCI_AD[31:0]	Z
PCI_AD_OE#	1
PCI_C/BE[3:0]#	Z
PCI_CLK	—
PCI_DEVSEL#	Z
PCI_EXT_SEL	1
PCI_FRAME#	Z
PCI_GNT#	—
PCI_IRDY#	Z
PCI_LOCK#	—
PCI_OL_OPEN	1
PCI_OUT_SEL	1
PCI_PAR	Z
PCI_PERR#	Z
PCI_REQ#	1
PCI_SERR#	Z
PCI_STOP#	Z
PCI_TRDY#	Z
RAS[7:0]#	1

664 Signal	State
RESET#	0
ROM_LOAD	0
ROM_OE#	1
ROM_WE#	1
SBE#	—
SHD#	Z
SRAM_ADS#	0
SRAM_ALE	1
SRAM_CNT_EN#	1
SRAM_OE#	1
SRAM_WE#	1
STOP_CLK_EN#	—
TA#	1
TAG_CLR#	0
TAG_MATCH	Z
TAG_VALID	1
TAG_WE#	1
TBST#	Z
TEA#	1
TEST#	1
TS#	Z
TSIZE[0:2]	Z
TT[0:4]	Z
WE[1:0]#	1
XATS#	—

**Notes:** During reset, INT\_CPU# is driven to the inverse of INT\_REQ. Drive TEST# continuously high and MIO\_TEST continuously low for correct operation.

Table 8-4. 663 Pin Reset State

663 Signal	State
AOS_RR_MMRS	—
C2P_WRL_OPEN	—
CPU_CLK	—
CPU_DATA[00:63]	Z
CPU_DATA_OE#	—
CPU_DPAR[0:7]	Z
CPU_PAR_ERR#	U
CPU_RDL_OPEN	—
CRS_C2PWXS	—
DUAL_CTRL_REF	—
ECC_LE_SEL	—

663 Signals	State
MEM_BE[0:1]	—
MEM_BE[2:3]	—
MEM_CHECK[0:7]	Z
MEM_DATA[63:0]	Z
MEM_DATA_OE#	—
MEM_ERR#	U
MEM_RD_SMPL	—
MEM_WRL_OPEN	—
MIO_TEST	0
MWS_P2MRXS	—

663 Signals	State
PCI_AD[31:0]	Z
PCI_AD_OE#	—
PCI_EXT_SEL	—
PCI_IRDY#	—
PCI_OL_OPEN	—
PCI_OUT_SEL	—
PCI_TRDY#	—
ROM_LOAD	—
SBE#	U
TEST#	1

**Note:** For correct operation, TEST# must always be driven high and MIO\_TEST must always be driven low.

**8.3.2.2 660 Configuration Strapping**

There are two strapping options for 660 system configuration information which is required before the processor can execute (and which, therefore, cannot be programmed into the 660). Configuration strapping is accomplished by attaching a pullup or pulldown resistor to the specified 664 output pin. During reset, the 664 tri-states these outputs, allowing them to assume the level to which they are strapped. When RESET# is deasserted, the 664 reads in the value from these pins.

Table 8-5 shows the strapping options and their associated pins. Pullup resistors should be 10K ohms to 20K ohms. Pull down resistors should be 500 ohms to 2K ohms.

In Table 8-5, 603(e) refers to either a 603 or a 603e.

**Table 8-5. Configuration Strapping Options**

Function	Pull Up/Down	Pin
Location of ROM	Up = Remote ROM — Down = Direct-Attach ROM	CRS_C2PWXS
603(e) in 1:1 or 3:2 CPU core:bus mode	Down = 603(e) not in 1:1 or 3:2 mode, or 601 or 604. Up = 603(e) in 1:1 or 3:2 CPU core:bus mode.	MWS_P2MRXS

**8.3.2.3 660 Deterministic Operation (Lockstep Applications)**

If fully deterministic operation of the chipset following RESET# is required, then the following items must be considered:

- When RESET# is deasserted, some outputs transition to a different but stable state. This results in requirement #3 in Section 8.3.2, that neither CPU or PCI bus activity is allowed to begin during the first four CPU clocks after the deassertion of RESET#.
- When RESET# is deasserted, the refresh counter begins (and continues) to run, counting the interval between refresh cycles to the memory. There are two ways to start the refresh timer deterministically:  
Meet the following timing requirements for RESET#, so that the clock cycle upon which RESET# is deasserted is known:
  - Setup > 4.2ns relative to a rising PCI clock edge.
  - Hold > 0ns relative to a rising PCI clock edge.

Write to the Refresh Timer Divisor Register (index D1–D0) and the Suspend Refresh Timer Register (index D3–D2) to reset the refresh counters. If this is done before any DRAM accesses occur, then no bus activity will have been affected by the unknown state of the counters before this point.

- When RESET# is deasserted, the DUAL\_CTRL\_REF signal begins toggling. The phase of this toggling never effects any bus operations, and therefore need not be known for deterministic operation of the 660. However, if it is still desirable to control the phase of DUAL\_CTRL\_REF, then the following timing requirements must be met for the deassertion of RESET#:
  - Setup > 4.4ns relative to a rising CPU clock edge.
  - Hold > 0ns relative to a rising CPU clock edge.

**8.4 Test Modes****8.4.1 CPU Test**

The three test pins of the 603e (LLSD\_MODE#, L1\_TST\_CLK#, & L2\_TST\_CLK#) are only supported for use during internal factory testing. They are not supported for customer use. Refer to the RISCWatch documentation for use of the JTAG port for debugging and diagnostic purposes.

**8.4.2 660 Test**

Tie the MIO\_TEST input of both the 663 and the 664 low during normal operation. IBM does not support any use of MIO test mode by external customers.



MIO test mode is enabled on the 663 (asynchronously) by asserting MIO\_TEST to the 663. MIO test mode is enabled on the 664 (asynchronously) by asserting MIO\_TEST to the 664. IBM uses LSSD test mode to verify the switching levels of the inputs of the 663 and the 664. The TEST# and MIO\_TEST pins of the 663 and 664 are intended for use by the IBM manufacturing process only. The inclusion of the following information in this section is the total extent to which IBM supports the use of these pins by external customers.

#### 8.4.2.1 LSSD Test Mode

Tie the TEST# input of both the 663 and the 664 high during normal operation. Do not allow these signals to be casually asserted. Caution is advised in the use of LSSD test mode.

LSSD test mode is enabled on the 663 (asynchronously) by asserting TEST# to the 663. In the same way, LSSD test mode is enabled on the 664 (asynchronously) by asserting TEST# to the 664. In LSSD test mode, the 663 and 664 pins shown in Table 8-6 are redefined to become LSSD test mode pins. These pins have LSSD functions only while the 663 (or 664) is in LSSD test mode. Otherwise the pins perform normally. IBM uses LSSD test mode to verify the logical operation of the 663 and the 664.

Table 8-6. LSSD Test Mode Pin Definitions

Test Pin Name	664 Pin	664 Pin Normal Name	663 Pin	663 Pin Normal Name
TEST_ACLK#	194	MEM_ERR#	149	ECC_LE_SEL
TEST_BCLK#	129	XATS#	170	DUAL_CTRL_REF
TEST_CCLK#	133	DPE#	163	MEM_BE[2]
SCAN_IN	192	CPU_PAR_ERR#	145	MEM_DATA_OE#
SCAN_OUT	47	ROM_OE#	174	CPU_PAR_ERR#
RI#	56	NMI_REQ	161	MEM_BE[0]
DI#	151	STOP_CLK_EN#	162	MEM_BE[1]

In LSSD test mode, never assert more than one of TEST\_ACLK#, TEST\_BCLK#, TEST\_CCLK#, and RESET# at the same time, as this may damage the device by provoking excessive internal current flows.

In LSSD test mode, the DI# pin controls the drivers of the 663 (and the 664). Assertion of the DI# pin asynchronously causes all of the 663 (or 664) output drivers (push-pull, tri-state, open-driver, or bi-directional) to be tristated.

In LSSD test mode, the RI# pin controls the receivers of the 663 (and the 664). Assertion of RI# causes all of the 663 (or 664) inputs to report a certain pattern to the internal logic. This has no effect on the external operation of the device that can be used by an external customer.

The 660 must be reset properly after leaving LSSD test mode in order to assure correct normal mode operation.

No further information on the use of the 660 test pins is expected to be released.

This image shows a full page of blank graph paper. The grid consists of small, equal-sized squares formed by thin black lines. There are 20 columns and 20 rows of squares, creating a total of 400 square units. The grid covers the entire area of the page, leaving no margins or other markings.This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

## **Section 9**

### **Set Up and Registers**

The following subsections represent activities carried out early in the boot firmware. On the MCM, all initialization registers are contained in the 603e and the 660 bridge controller chips.

#### **9.1 CPU Initialization**

The 603 CPU exits the reset state with the L1 cache disabled and bus error checking disabled.

All memory pages 2G to 4G must be marked as non-cacheable.

The Segment Register T bit, bit 0, defaults to 0 which is the normal storage access mode. It must be left in this state for the hardware to function. Direct store (PIO) segments are not supported.

Set the bit that controls ARTRY# negation, HID0[7], to 0 to enable the precharge of ARTRY#. It may be necessary set HID0[7] to 1 to disable the precharge of ARTRY# for 100 MHz PPC 603e MCM configurations having a CPU bus agent (such as an added L2) that drives the ARTRY# line. Software must set this bit before allowing any CPU bus traffic to which the CPU agent might respond. Note that PCI to memory transactions cause the 660 bridge to broadcast snoop operations on the CPU bus.

HID0 bit 0, Master Checkstop Enable, defaults to 1 which is the enabled state. Leave it in this state so that checkstops can occur.

MCM errors are reported through the 660 by way of the TEA# and MCP# pins. Because of this, the bus error checking in the CPU must be disabled by setting HID0 bits 2 and 3 to zero.

#### **9.2 660 Bridge Initialization**

Before DRAM memory operations can begin, the software must:

1. Read the SIMM presence detect and SIMM type registers.
2. Set up and check the memory-related registers in the 660 (see the *660 Bridge User's Manual*).
3. Program the timer in the ISA bridge register which controls ISA refresh timing. In SIO compatible bridges it should be programmed to operate in Mode 2 with an interval of approximately 15 usec.
4. Make sure 200 usec has elapsed since starting the refresh timer so that sufficient refresh cycles have occurred to properly start the memory. This will be hidden if

approximately 120 Flash accesses occur after the timer is started and before the memory initialization starts.

5. Initialize all of memory so that all parity bits are properly set. (The CPU may cache unnecessary data; hence, all of memory must be initialized.) The 660 does not require reconfiguration when port 4Dh in the ISA bridge is utilized to reset the native I/O and the ISA slots.
6. Read the L2 cache presence detect bits and set register xx for 512K of synchronous SRAM cache.

### 9.3 PCI Configuration Scan

The 660 bridge in the MCM enables the software to implement a scan to determine the complement of PCI devices present. This is because the system returns all ones rather than an error when no PCI device responds to initialization cycles. The software may read each possible PCI device ID to determine devices present.

**Table 9-1. Configuration Address Assignments**

Device	IDSEL Line	60X Address*	PCI Address
ISA bus bridge (SIO)	A/D 11	8080 08XXh	080 08XX
PCI Slot 1	A/D 12	8080 10XXh	080 10XX
PCI Slot 2	A/D 13	8080 20XXh	080 20XX
PCI Slot 3	A/D 14	8080 40XXh	080 40XX

**Note:** \*This address is independent of contiguous I/O mode.

Software must use only the addresses specified. Using any addresses that causes more than one IDSEL to be asserted (high) can cause bus contention, because multiple PCI agents will be selected.

In systems that contain the Intel SIO chip, it must be configured prior to any other PCI bus agent. The SIO PCI arbiter is automatically enabled upon power-on reset. During power-on reset, the SIO drives the A/D(31:0), C/BE#(3:0), and PAR signals on the PCI bus.

#### 9.3.1 Multi-Function Adaptors

The 660 supports multi-function adaptors. It passes, unmodified, the address of the load or store instruction that causes a PCI configuration cycle. The only exception is that the three low-order bits are unmunged in little endian mode, and the two low-order address bits are set to zero in either endian mode; therefore, addresses may be selected with non-zero CPU address bits (21:23)—corresponding to PCI bits (10:8)—to configure multi-function adaptors. For example, to configure device 3 in slot 1, use address 80C0 03XXh. To configure device 7 in slot 2, use address 8084 07XXh.

#### 9.3.2 PCI to PCI Bridges

The 660 supports both Type 0 and Type 1 configuration cycles.

#### 9.3.3 Indexed BCR Summary

Table 9-2 contains a summary listing of the indexed BCRs in the 660. Access to these registers is described in the *660 Bridge User's Manual*. The values shown in the Set To column are for reference only, and may not apply to a particular application.

**Table 9-2. 660 Bridge Indexed BCR Listing**

Bridge Control Register	Index	R/W	Bytes	Set To (1)
PCI Vendor ID	Index 00 – 01	R	2	1014h
PCI Device ID	Index 02 – 03	R	2	0037h

Table 9-2. 660 Bridge Indexed BCR Listing (Continued)

Bridge Control Register	Index	R/W	Bytes	Set To (1)
PCI Command	Index 04 – 05	R/W	2	—
PCI Device Status	Index 06 – 07	R/W	2	—
Revision ID	Index 08	R	1	02h
PCI Standard Programming Interface	Index 09	R	1	0
PCI Subclass Code	Index 0A	R	1	0
PCI Class Code	Index 0B	R	1	06h
PCI Cache Line Size	Index 0C	R	1	0
PCI Latency Timer	Index 0D	R	1	0
PCI Header Type	Index 0E	R	1	0
PCI Built-in Self-Test (BIST) Control	Index 0F	R	1	0
PCI Interrupt Line	Index 3C	R	1	0
PCI Interrupt Pin	Index 3D	R	1	0
PCI MIN_GNT	Index 3E	R	1	0
PCI MAX_LAT	Index 3F	R	1	0
PCI Bus Number	Index 40	R	1	0
PCI Subordinate Bus Number	Index 41	R	1	0
PCI Disconnect Counter	Index 42	R/W	1	0
PCI Special Cycle Address BCR	Index 44 –45	R	2	0
Memory Bank 0 Starting Address	Index 80	R/W	1	Memory
Memory Bank 1 Starting Address	Index 81	R/W	1	Memory
Memory Bank 2 Starting Address	Index 82	R/W	1	Memory
Memory Bank 3 Starting Address	Index 83	R/W	1	Memory
Memory Bank 4 Starting Address	Index 84	R/W	1	Memory
Memory Bank 5 Starting Address	Index 85	R/W	1	Memory
Memory Bank 6 Starting Address	Index 86	R/W	1	Memory
Memory Bank 7 Starting Address	Index 87	R/W	1	Memory
Memory Bank 0 Ext Starting Address	Index 88	R/W	1	Memory
Memory Bank 1 Ext Starting Address	Index 89	R/W	1	Memory
Memory Bank 2 Ext Starting Address	Index 8A	R/W	1	Memory
Memory Bank 3 Ext Starting Address	Index 8B	R/W	1	Memory
Memory Bank 4 Ext Starting Address	Index 8C	R/W	1	Memory
Memory Bank 5 Ext Starting Address	Index 8D	R/W	1	Memory
Memory Bank 6 Ext Starting Address	Index 8E	R/W	1	Memory
Memory Bank 7 Ext Starting Address	Index 8F	R/W	1	Memory
Memory Bank 0 Ending Address	Index 90	R/W	1	Memory
Memory Bank 1 Ending Address	Index 91	R/W	1	Memory
Memory Bank 2 Ending Address	Index 92	R/W	1	Memory
Memory Bank 3 Ending Address	Index 93	R/W	1	Memory
Memory Bank 4 Ending Address	Index 94	R/W	1	Memory
Memory Bank 5 Ending Address	Index 95	R/W	1	Memory
Memory Bank 6 Ending Address	Index 96	R/W	1	Memory
Memory Bank 7 Ending Address	Index 97	R/W	1	Memory

Table 9-2. 660 Bridge Indexed BCR Listing (Continued)

Bridge Control Register	Index	R/W	Bytes	Set To (1)
Memory Bank 0 Ext Ending Address	Index 98	R/W	1	Memory
Memory Bank 1 Ext Ending Address	Index 99	R/W	1	Memory
Memory Bank 2 Ext Ending Address	Index 9A	R/W	1	Memory
Memory Bank 3 Ext Ending Address	Index 9B	R/W	1	Memory
Memory Bank 4 Ext Ending Address	Index 9C	R/W	1	Memory
Memory Bank 5 Ext Ending Address	Index 9D	R/W	1	Memory
Memory Bank 6 Ext Ending Address	Index 9E	R/W	1	Memory
Memory Bank 7 Ext Ending Address	Index 9F	R/W	1	Memory
Memory Bank Enable	Index A0	R/W	1	Memory
Memory Timing 1	Index A1	R/W	1	0001 0010
Memory Timing 2	Index A2	R/W	1	1000 1010
Memory Bank 0 & 1 Addressing Mode	Index A4	R/W	1	Mode 2
Memory Bank 2 & 3 Addressing Mode	Index A5	R/W	1	—
Memory Bank 4 & 5 Addressing Mode	Index A6	R/W	1	—
Memory Bank 6 & 7 Addressing Mode	Index A7	R/W	1	—
Cache Status	Index B1	R/W	1	—
Refresh Cycle Definition	Index B4	R	1	—
Refresh Timer B5 (Not used – see Indexed BCR D0)	Index B5	R	1	—
RAS Watchdog Timer	Index B6	R/W	1	53h
PCI Bus Timer (Not used)	Index B7	R	1	0
Single-Bit Error Counter	Index B8	R/W	1	0
Single-Bit Error Trigger Level	Index B9	R/W	1	0
Bridge Options 1	Index BA	R/W	1	07h
Bridge Options 2	Index BB	R/W	1	5Fh
Error Enable 1	Index C0	R/W	1	EDh
Error Status 1	Index C1	R/W	1	0
Error Simulation 1	Index C2	R/W	1	0
CPU Bus Error Status	Index C3	R	1	14h
Error Enable 2	Index C4	R/W	1	04h
Error Status 2	Index C5	R/W	1	04h
Error Simulation 2	Index C6	R/W	1	0
PCI Bus Error Status	Index C7	R/W	1	0Fh
CPU/PCI Error Address	Index C8–CB	R/W	4	—
Single-Bit ECC Error Address	Index CC – CF	R/W	4	—
Refresh Timer Divisor	Index D0 – D1	R/W	2	0180h
Suspend Refresh Timer	Index D2 – D3	R/W	2	01F8h
Bridge Chip Set Options 3	Index D4	R/W	1	88h

**Notes:**

1. In this column, a long dash — means that the initialization firmware does not write to this register. The register is either not used, not written to, or the value of it depends on changing circumstances.  
If the word Memory appears, please refer to the System Memory section of the 660 User's Manual.
2. The initialization firmware sets the Memory registers depending on the information reported by the DRAM presence detect registers.

## **Section 10 System Firmware**

### **10.1 Introduction**

The firmware on the 100 MHz PPC 603e MCM planar handles three major functions:

- Test the system in preparation for execution,
- Load and execute an executable image from a bootable device, and
- Allow user configuration of the system.

Section 10.2 briefly discusses the power on system test function.

Section 10.3 details a structure for boot records which can be loaded by the system firmware.

Section 10.4 describes the system configuration utility.

This information is included for reference only. Some of the information in this section concerns the example planar rather than the MCM.

**To obtain a copy of the commented source code of the firmware on diskette, contact your IBM representative. This material is available free of charge with a signed license agreement.**

### **10.2 Power On System Test**

The Power On System Test (POST) code tests those subsystems of the reference board which are required for configuration and boot to ensure minimum operability. Tests also assure validity of the firmware image and of the stored system configuration.

#### **10.2.1 Hardware Requirements**

In addition to the reference board, the firmware requires the following peripherals to be installed as adapter cards:

- |                     |                                 |                     |
|---------------------|---------------------------------|---------------------|
| • Serial Port 1     | Address: 0x3F8 (COM1:)          | Interrupt: IRQ 4    |
| • Serial Port 2     | Address: 0x2F8 (COM2:)          | Interrupt: IRQ 3    |
| • Floppy Controller | Address: 0x3F0 (Primary Floppy) | Mode: PC/AT or PS/2 |
| • IDE Controller    | Address: 0x1F0 (Primary IDE)    |                     |

### **10.3 Boot Record Format**

The firmware will attempt to boot an executable image from devices specified by the user. See Section 10.4 for details on specifying boot devices and order.

The *PowerPC Reference Platform Specification* details a structure for boot records which can be loaded by the system firmware. This specification is described in the following sections.

### 10.3.1 Boot Record

The format of the boot record is an extension of the PC environment. The boot record is composed of a PC compatibility block and a partition table. To support media interchange, the PC compatibility block may contain an x86-type program. The entries in the partition table identify the PowerPC Reference Platform boot partition and its location in the media.

The layout of the boot record must be designed as shown in Figure 10-1. The first 446 bytes of the boot record contain a PC compatibility block, the next four 16-byte entries make up a partition table totalling 64 bytes, and the last 2 bytes contain a signature.

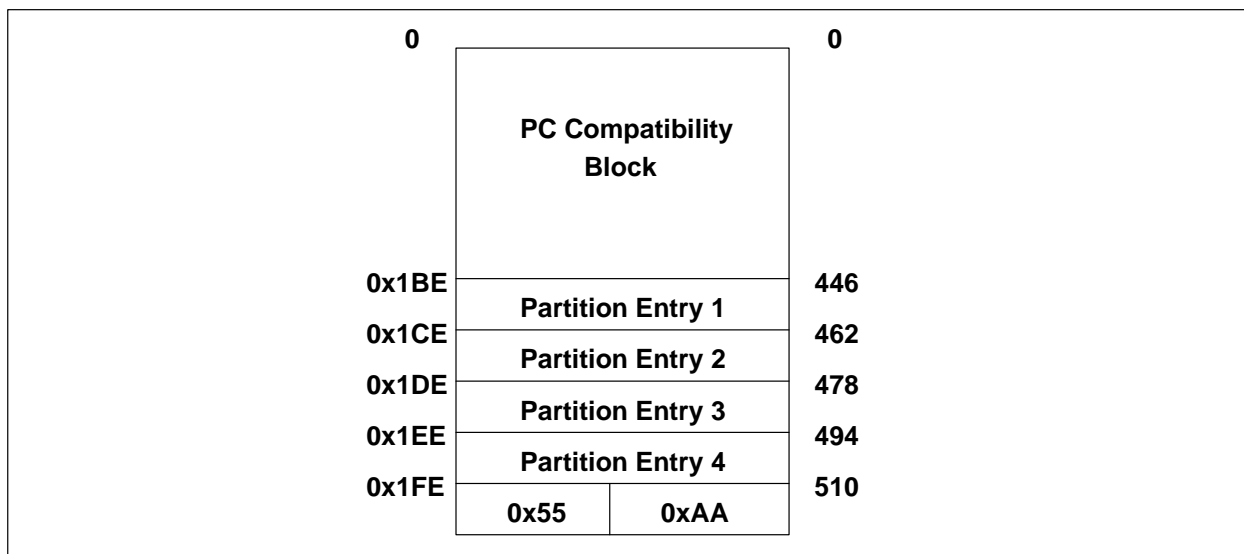


Figure 10-1. Boot Record

#### 10.3.1.1 PC Partition Table Entry

To support media interchange with the PC, the PowerPC Reference Platform defines the format of the partition table entry based on that for the PC. This section describes the format of the PC partition table entry, which is shown in Figure 10-2.

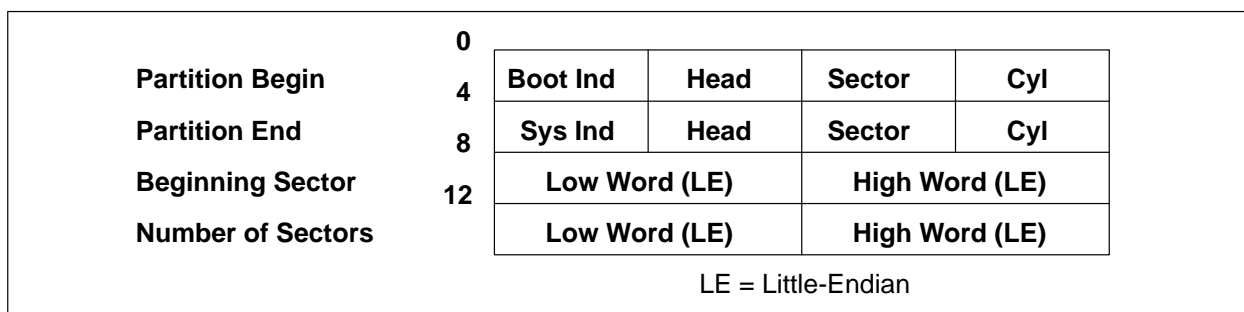


Figure 10-2. Partition Table Entry



- Partition Begin The beginning address of the partition in head, sector, cylinder notation.
- Partition End The end address of the partition in cylinder, head, sector notation.
- Beginning Sector The number of sectors preceding the partition on the disk. That is, the zero-based relative block address of the first sector of the partition.
- Number of Sectors The number of sectors allocated to the partition.

The subfields of a partition table entry are defined as follows:

- Boot Ind Boot Indicator. This byte indicates if the partition is active. If the byte contains 0x00, then the partition is not active and will not be considered as bootable. If the byte contains 0x80, then the partition is considered active.
- Head An eight-bit value, zero-based.
- Sector A six-bit value, one-based. The low-order six bits are the sector value. The high-order two bits are the high-order bits of the 10-bit cylinder value.
- Cyl Cylinder. The low-order eight-bit component of the 10-bit cylinder value (zero-based). The high-order two bits of the cylinder value are found in the sector field.
- Sys Ind System Indicator. This byte defines the type of the partition. There are numerous partition types defined. For example, the following list shows several:

0x00	Available partition
0x01	DOS, 12-bit FAT
0x04	DOS, 16-bit FAT
0x05	DOS extended partition
0x41	PowerPC Reference Platform partition.

### 10.3.1.2 Extended DOS Partition

The extended DOS partition is used to allow more than four partitions in a device. The boot record in the extended DOS partition has a partition table with two entries, but does not contain the code section. The first entry describes the location, size and type of the partition. The second entry points to the next partition in the chained list of partitions. The last partition in the list is indicated with a system indicator value of zero in the second entry of its partition table.

Because of the DOS format limitations for a device partition, a partition which starts at a location beyond the first 1 gigabyte is located by using an enhanced format shown in Figure 10-3.

<b>Partition Begin</b>	0	<b>Boot Ind</b>	-1	-1	-1
<b>Partition End</b>	4	<b>Sys Ind</b>	-1	-1	-1
<b>Beginning Sector</b>	8	<b>32-bit start RBA (zero-based) (LE)</b>			
<b>Number of Sectors</b>	12	<b>32-bit RBA count (one-based) (LE)</b>			

LE = Little-Endian

-1 = All ones in the field.  
RBA = Relative Block Address in units of 512 bytes.

**Figure 10-3. Partition Table Entry Format for an Extended Partition****10.3.1.3 PowerPC Reference Platform Partition Table Entry**

The Power PC Reference Platform partition table entry (see Figure 10-4) is identified by the 0x41 value in the system indicator field. All other fields are ignored by the firmware except for the Beginning Sector and Number of Sectors fields. The CV (Compatible Value – not shown) fields must contain PC-compatible values (i.e. acceptable to DOS) to avoid confusing PC software. The CV fields, however, are ignored by the firmware.

<b>Partition Begin</b>	0	<b>Boot Ind</b>	<b>Head</b>	<b>Sector</b>	<b>Cyl</b>
<b>Partition End</b>	4	<b>Sys Ind</b>	<b>Head</b>	<b>Sector</b>	<b>Cyl</b>
<b>Beginning Sector</b>	8	<b>32-bit start RBA (zero-based) (LE)</b>			
<b>Number of Sectors</b>	12	<b>32-bit RBA count (one-based) (LE)</b>			

RBA = Relative Block Address in units of 512 bytes.      LE = Little-Endian

**Figure 10-4. Partition Table Entry for PowerPC Reference Platform**

The 32-bit start RBA is zero-based. The 32-bit count RBA is one-based and indicates the number of 512-byte blocks. The count is always specified in 512-byte blocks even if the physical sectoring of the target devices is not in 512-byte sectors.

**10.3.2 Loading the Load Image**

This section describes the layout of the PowerPC 0x41 type partition and the process of loading the load image.

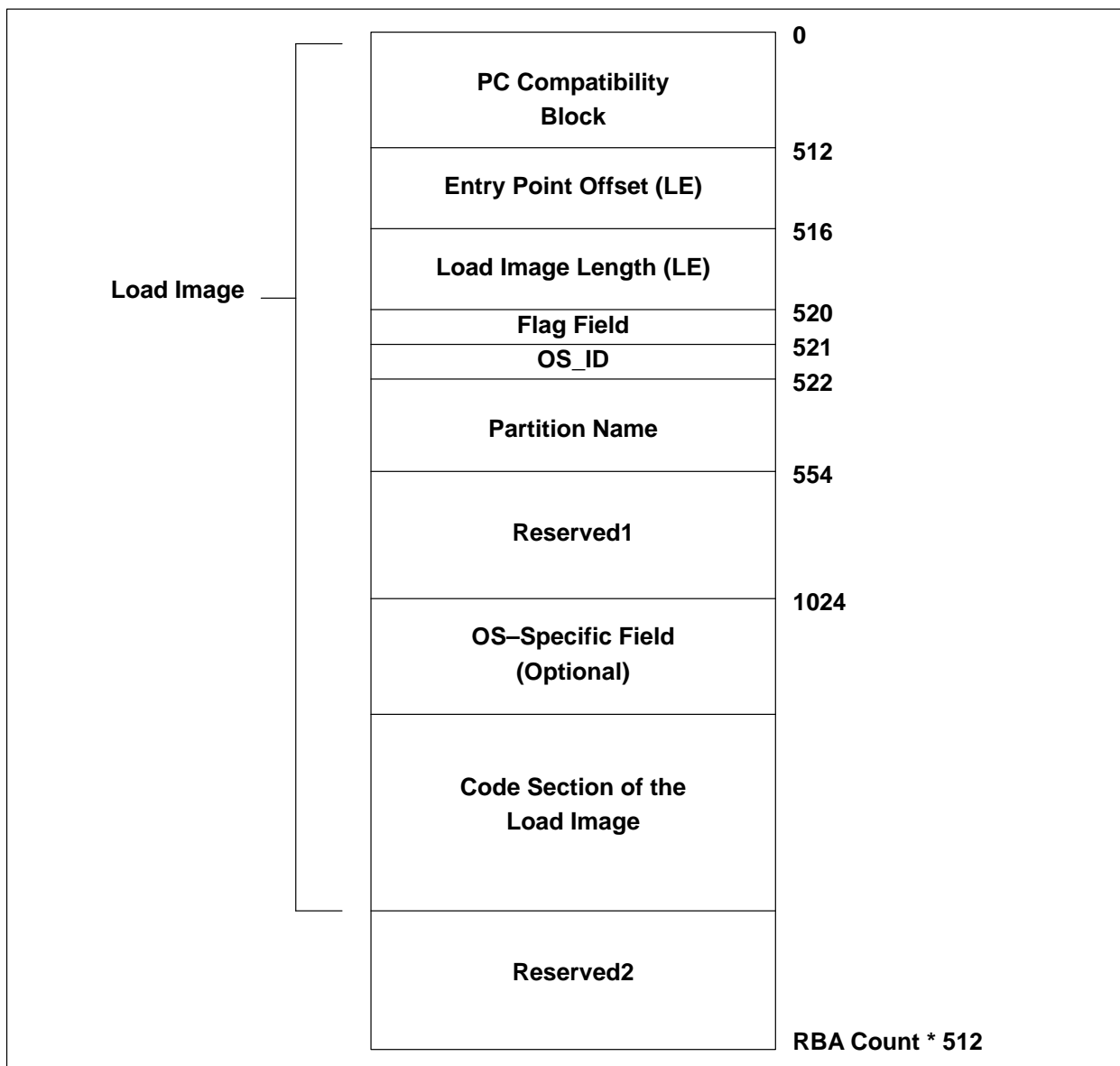


Figure 10-5. PowerPC Reference Platform Partition

The layout for the 0x41 type partition is shown in Figure 10-5. The PC compatibility block in the boot partition may contain an x86-type program. When executed on an x86 machine, this program displays a message indicating that this partition is not applicable to the current system environment.

The second relative block in the boot partition contains the entry point offset, load image length, flag field, operating system ID field, ASCII partition name field, and the reserved1 area. The 32-bit entry point offset (little-endian) is the offset (into the image) of the entry point of the PowerPC Reference Platform boot program. The entry point offset is used to allocate the Reserved1 space. The reserved1 area from offset 554 to Entry Point – 1 is reserved for implementation specific data and future expansion.

The 32-bit load image length (little-endian) is the length in bytes of the load image. The load image length specifies the size of the data physically copied into the system RAM by the firmware.

The flag field is 8 bits wide. The MSb in the field is allocated for the Open Firmware flag. If this bit is set to 1, the loader requires Open Firmware services to continue loading the operating system.

The second MSb is the endian mode bit. If the mode bit is 0, the code in the section is in big-endian mode. Otherwise, the codes is in little-endian mode. The implication of the endian mode bit is different depending on the Open Firmware flag. If the Open Firmware flag is set to 1, the mode bit indicates the endian mode of the code section pointed to by the load image offset, and the firmware has to establish the hardware endian mode according to this bit. Otherwise, this bit is just an informative field for firmware.

The OS\_ID field and partition name field are used to identify the operating system located in the partition. The OS\_ID field has the numeric identification value of the operating system located in the partition. The 32 bytes of partition name field must have the ASCII notation of the partition name. The name and OS\_ID can be used to provide to a user the identification of the boot partition during the manual boot process.

Once the boot partition is identified by the PowerPC Reference Platform boot partition table entry, the firmware:

- Reads into memory the second 512-byte block of the boot partition
- Determines the load image length for reading in the boot image up to but not including the reserved2 space
- Allocates a buffer in system RAM for the load image transfer (no fixed location)
- Transfers the load image into system RAM from the boot device (the reserved2 space is not loaded).

The load image must be fully relocatable, as it may be placed anywhere in memory by the system firmware. Once loaded, the load image may relocate itself anywhere within system RAM.

## 10.4 System Configuration

This section describes the utilities in the system firmware which allow the system to be customized. These utilities allow viewing of the system configuration, as well as the ability to change I/O device configurations, console selection, boot devices, and the date and time. These functions are described in the following sections.

### 10.4.1 System Console

The system console can be either a screen-oriented video display or a line-oriented serial terminal. The example screens shown in this section show the S3 video/keyboard interface. When using a serial terminal, the configuration utilities will prompt for numeric input for each prompt instead of using the arrow keys. All choices and options are the same as for the screen-oriented menus.

The configuration of the reference board as shipped is set for S3 video / Keyboard console. In the case that either the video adapter or the keyboard fails the power-on test, the system

console will default to serial port 1. The baud rate for the serial console is specified in the configuration menus. The value as shipped is 9600 baud.

#### 10.4.2 System Initialization

The logo screen, shown in Figure 10-6, is displayed at power-on. The logo screen is active while the system initializes and tests memory and performs a scan of the SCSI bus to determine what SCSI devices are installed.

```
PowerPC 603/604 Reference Board System Firmware
(C) Copyright 1994 IBM Corp. All Rights Reserved.

#####
# # ##### # # ##### ##### # # #
# # # # # # # # # # #
##### # # # # ##### # ##### #
# # # # # # # ##### #
# # # # # # # # # # #
# # # # # # # # # #
# ##### # # # #####
```

PowerPC 603/604 Reference Board

Press C during memory test for configuration utilities

Testing 8192K of memory: 8192 KB OK

Figure 10-6. System Initialization Screen

While the logo screen is displayed, pressing the 'C' key on the console will enter the system configuration utility. The configuration menu will also be entered if there is no bootable device present, or if the configuration stored in the system non-volatile RAM is not initialized or is corrupt.

### 10.4.3 Main Menu

Figure 10-7 shows the main menu for the system configuration utility. Selections on the menu are highlighted by using the up and down arrow keys on the keyboard, and are chosen with the Enter key. Each choice is detailed in the following sections.

PowerPC 603/604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.
<p style="text-align: center;">Main Menu</p> <p>System Configuration Menu Run a Program Reprogram Flash Memory</p> <p>Save and exit Exit without saving</p>
<p>Press ↑↓ to select item Press Enter to perform action</p>

**Figure 10-7. Configuration Utility Main Menu**

### 10.4.3.1 System Configuration Menu

Figure 10-8 shows the System Configuration menu, which has choices to display and change the default state of the reference board on boot. Each menu item is discussed in the following sections.

<p>PowerPC 604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.</p>
<p>System Configuration Menu</p> <p>System Information Configure I/O Devices View SCSI Devices Set Boot Devices Set Date and Time</p> <p>Previous Menu</p>
<p>Press ↑↓ to select item Press Enter to perform action</p>

Figure 10-8. System Configuration Menu

## System Information

The system configuration option shows the hardware configuration of the system at power-up—including processor, installed options, and firmware revision level. A sample screen is shown in Figure 10-9.

PowerPC 603/604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.	
System Configuration	
System Processor	PowerPC 604
Installed Memory	8 MB
Second-Level Cache	Not Installed
Upgrade Processor	Not Installed
Boot Firmware Revision	1.0
Go to Previous Menu	
Press ↑↓ to select item Press Enter to perform action	

**Figure 10-9. System Information Screen**



## Configure I/O Devices

The configure I/O devices option allows the customization of system I/O ports and the system console. The menu is shown in Figure 10-10. Options are highlighted by using the up and down arrow keys on the keyboard and are changed with the left and right arrow keys. Options on the menu are discussed below.

PowerPC 603/604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.	
Device Configuration	
Select Console Device	[S3 Video / Keyboard]
Set Serial Port 1 Speed	[ 9600 Baud]
Set Serial Port 2 Speed	[ 9600 Baud]
Go to Previous Menu	
Press ↑ ↓ to select item Press ← → to change item	

Figure 10-10. Device Configuration Screen

Any changes made in I/O device configuration are saved when the Save and Exit option on the main menu is selected. Exiting the system configuration utility in any other manner will cause device configuration changes to be lost.

### Select Console Device

The console selection box allows the selection of an option for the system console

- Serial Port 1 or 2      Console input and output will be transmitted and received through a serial port on an adapter card. Console input and output will be transmitted and received at the baud rate selected with Serial Port Speed.
- S3 Video/Keyboard      Console output will be displayed on a video monitor connected to an S3 PCI video adapter; console input will be received from a keyboard connected to the keyboard connector on the reference board

### Set Serial Port 1 or 2 Speed

The serial port speed selection box sets the speed of each serial port. Baud rates for the two serial ports are independent. If a serial port is used as the system console, set this value to match the baud rate of the terminal.

### View SCSI Devices

The SCSI devices screen shows the devices found on the SCSI bus during power-on initialization. The string shown is the SCSI device's response to the SCSI inquiry command. According to the SCSI specification, this data comprises the manufacturer's ID, device model number, and device revision level. A sample screen is shown in Figure 10-11.

PowerPC 603/604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.	
SCSI Devices	
SCSI Device 0	None
SCSI Device 1	None
SCSI Device 2	None
SCSI Device 3	None
SCSI Device 4	None
SCSI Device 5	None
SCSI Device 6	IBM MXT-540SL H
Previous Menu	
Press ↑↓ to select item Press Enter to perform action	

Figure 10-11. SCSI Devices Screen

## Set Boot Devices

The boot devices menu allows the user to select which devices are queried for boot images and in what order they are selected for boot. Allowable selections are one of the two floppy disk drives, any of six SCSI drive ID numbers, either of two IDE disk drives, or no device selected. The default configuration is shown in Figure 10-12. In this configuration, the system will attempt to find a boot image on the first floppy disk drive. If this fails, the system will attempt to boot from the SCSI device programmed to SCSI ID 6. If this fails, the system will attempt to boot from IDE drive zero (master).

PowerPC 603/604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.	
Boot Device Selection	
Set Boot Device 1	[Floppy 1 ]
Set Boot Device 2	[SCSI ID 6 ]
Set Boot Device 3	[IDE Drive 0 ]
Set Boot Device 4	[None ]
Go to Previous Menu	
Press ↑ ↓ to select item Press ← → to change item	

**Figure 10-12. Boot Devices Screen**

If the system fails to find a valid boot image (as discussed in Section 10.3) on any of the selected boot devices, or if no boot device is selected, the user will be prompted to enter the configuration menu to select a valid boot device.

Any changes made in boot device selection is saved when the Save and Exit option on the main menu is selected. Exiting the system configuration utility in any other manner will cause boot device changes to be lost.

## Set Date and Time

The set date and time screen allows the date and time stored in the battery-backed real time clock to be updated. The screen is shown in Figure 10-13. To change the time, the left and right arrow keys are used to select the digit to modify, and the digit is then typed over with the number keys. The date or time will be updated when Enter or either the up or down arrow is pressed. Changing the date or time is immediate, and is not affected by either the Save and Exit or Exit Without Saving options on the main menu.

PowerPC 603/604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.
<p>Set Date and Time</p> <p>Set Date [03/01/94] Set Time [11:30:00]</p> <p>Go to Previous Menu</p>
Press ↑↓ to select item Enter data at cursor

**Figure 10-13. Set Date and Time Screen**

### 10.4.3.2 Run a Program

The Run a Program option on the main menu loads and executes a program from a FAT (DOS) disk or from a CD-ROM in ISO-9660 format. The program is loaded at location 0x00400000 (4 MB) and control is passed with a branch to the first address.

All boot devices specified in the Boot Devices Menu will be searched in order for FAT and CD-ROM file systems, and the first matching file on a boot device will be loaded.

The Run a Program screen is shown in Figure 10-14. To run a program, enter the file name in the Specify Program Filename field and select the Run the Program option.

PowerPC 603/604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.
<p>Run a Program</p> <p>Specify Program Filename [            ]</p> <p>Run the Program</p> <p>Go to Previous Menu</p>
<p>Press ↑↓ to select item</p> <p>Enter data at cursor</p>

Figure 10-14. Run a Program Screen

### 10.4.3.3 Reprogram Flash Memory

The PowerPC 603/604 reference board stores its system firmware in a reprogrammable flash memory on the system board. The reprogram flash memory option on the main menu allows the reprogramming of the flash device with a DOS-formatted diskette. This allows future revisions of the system firmware to be provided on diskette without the need for removal of the device from the board.

If done improperly, reprogramming the flash memory can cause the system to become unusable until external means are available to reprogram the device. Use this option with care.

All boot devices specified in the Boot Devices Menu will be searched in order for FAT and CD-ROM file systems, and the first matching file on a boot device will be loaded.

The Reprogram the Flash Memory screen is shown in Figure 10-15. To reprogram the flash, enter the file name in the Specify Image Filename field and select the Reprogram the Memory option.

PowerPC 603/604 Reference Board System Firmware (C) Copyright 1994 IBM Corp. All Rights Reserved.
<p>Reprogram Flash Memory</p> <p>Image Filename [            ]</p> <p>Reprogram the Memory</p> <p>Go to Previous Menu</p>
<p>Press ↑↓ to select item</p> <p>Enter data at cursor</p>

Figure 10-15. Reprogram the Flash Memory Screen

**10.4.3.4 Exit Options**

The two exit options at the bottom of the main menu leave the system configuration utility. The two options are:

- **Save and Exit**                      Saves any changes made in the Configure I/O Devices and Set Boot Devices screens, and restarts the system.
- **Exit without Saving**              Proceeds with the boot process as if the configuration utility had not been entered. Any changes made in Configure I/O Devices or Set Boot Devices are lost.

**10.4.4 Default Configuration Values**

When the PowerPC 603/604 reference board is shipped from the factory, it has the following default configuration:

- **Console Device**                      S3 Video / Keyboard
- **Serial Port 1**                        9600 Baud
- **Serial Port 2**                        9600 Baud
- **Boot Devices**                        Device 1 - Floppy 1  
    Device 2 - SCSI ID 6  
    Device 3 - IDE Drive 0

These default values also take effect whenever the system configuration in system nonvolatile RAM becomes corrupted.

This image shows a full page of blank graph paper. The grid consists of small, equal-sized squares formed by thin black lines. There are 20 columns and 20 rows of squares, creating a total of 400 square units. The grid covers the entire area of the page, leaving no margins or other markings.This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.



## Section 11

### Endian Mode Considerations

Data represented in memory or media storage is said to be in big endian (BE) order when the most significant byte is stored at the lowest numbered address, and less significant bytes are at successively higher numbered addresses.

Data is stored in little endian (LE) order when it is stored with the order of bytes reversed from that of BE order. In other words, the most significant byte is stored at the highest numbered address. The endian ordering of data never extends past an 8-byte group of storage.

The 100 MHz PPC 603e MCM normally operates with big endian (BE) byte significance, which is the native mode of the PowerPC 603e CPU. Internally, the CPU always operates with big endian addresses, data, and instructions, which is ideal for operating systems such as AIX™, which store data in memory and on media in big endian byte significance. In BE mode, neither the CPU nor the 660 Bridge perform address or data byte lane manipulations that are due to the endian mode. Addresses and data pass 'straight through' the CPU bus interface and the 660.

The CPU also features a mode of operation designed to efficiently process code and operating systems such as WindowsNT™, which store data in memory and on media in LE byte significance. The MCM also supports this mode of operation.

When the MCM is in little endian mode, data is stored in memory with LE ordering. The 660 has hardware to select the proper bytes in the memory and on the PCI bus (via address transforms), and to steer the data to the correct CPU data lane (via a data byte lane swapper). Also, see the *603e CPU* and *660 Bridge User's Manuals*.

Table 11-1 summarizes the operation of the MCM in the two different modes.

**Table 11-1. Endian Mode Operations**

Mode	What the 603e Does	What the 660 Does
Big Endian (BE)	No munge, no shift	No unmunge, no swap
Little Endian (LE)	Address Munged & Data Shifted	Address Unmunged & Data Swapped

In BE mode, the CPU emits the address unchanged, and does not shift the data. This is the native mode of the 603e CPU. In BE mode, the 660 passes the address and data through to the target without any changes (that are due to endian mode).

In LE mode, the CPU transforms (munges) the three least significant address bits, and shifts the data on the byte lanes to match the munged address. In LE mode, the 660 unmunges the address and swaps the data on the byte lanes.

## 11.1 What the 603e CPU Does

### 11.1.1 The 603e Address Munge

The 603e CPU assumes that the significance of memory is BE. When it operates in LE mode, it internally generates the same effective address as the LE code would generate. Since it assumes that the memory is stored with BE significance, it transforms (munges) the three low order addresses when it activates the address pins. For example, in the 1-byte transfer case, address 7 is munged to 0, 6 to 1, 5 to 2, and so on. Table 11-2 shows the address transform rules for the allowed LE mode transfer sizes.

**Table 11-2. 603e LE Mode Address Transform**

Transfer Size	Address Transform
8	None
4	Physical Address[29:31] XOR 100 => A[29:31]
2	Physical Address[29:31] XOR 110 => A[29:31]
1	Physical Address[29:31] XOR 111 => A[29:31]

### 11.1.2 The 603e Data Shift

The data transfer occurs on the byte lanes identified by the address pins and transfer size (TSIZ) pins in either BE or LE mode. In LE mode, the CPU shifts the data from the byte lanes pointed to by the unmunged address, over to the byte lanes pointed to by the munged address. This shift is linear in that it does not rotate or alter the order of the bytes, which are now in the proper set of byte lanes. Note that the individual bytes are still in BE order.

## 11.2 What the 660 Bridge Does

While the MCM is operating properly, data is stored in system memory in the same endian mode as the mode in which the CPU operates. That is, the byte significance in memory is BE in BE mode and it is LE in LE mode. Because of this, hardware is included in the 660 that (in LE mode) will swap the data bytes to the correct byte lanes, and that will transform (or un-munge) the address coming from the 603e.

### 11.2.1 The 660 Bridge Address Unmunge

In LE mode, the 660 unmunges address lines A[29:31]. This unmunge merely applies the same XOR transformation to the three low-order address lines as did the CPU. This effectively reverses the effect of the munge that occurs within the CPU. For example, if the CPU executes a one-byte load coded to access byte 0 of memory in LE mode, it will munge its internal address and emit address A[29:31] = 7h. The 660 will then unmunge the 7 on A[29:31] back to 0, and use this address to access memory.

### 11.2.2 The 660 Bridge Data Swapper

The 660 contains a byte swapper. As shown in Figure 11-1, the byte swapper is placed between the CPU data bus and the memory and PCI data busses. This allows the byte lanes to be swapped between the CPU bus and the PCI bus, or between the CPU bus and memory, but not between the PCI bus and memory. Thus, when a PCI busmaster accesses memory, the MCM does not change either the address or the data location to adjust for endian mode. In either mode, data is stored or fetched from memory at the address presented on the PCI bus.

The 660 cannot tell the endian mode of the CPU directly, and so cannot automatically change endian mode to match the CPU. There is a control bit located in ISA I/O space (port 0092) that the CPU can write to in order to set the endian mode of the motherboard.

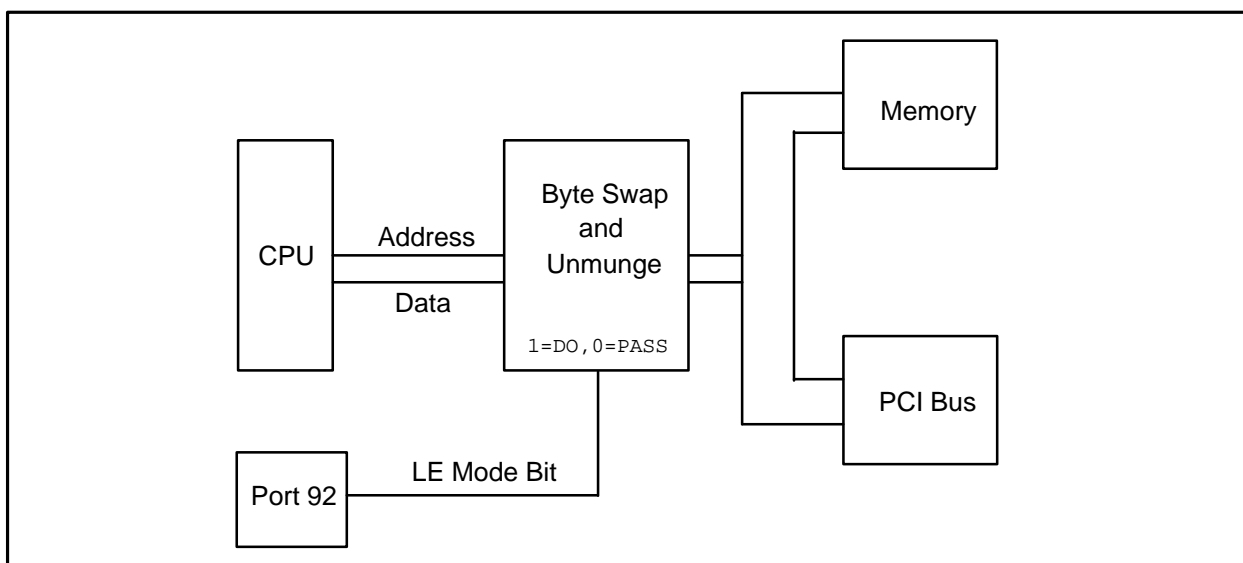


Figure 11-1. Endian Mode Block Diagram

In BE mode, the 660 byte swapper is off, and data passes through it with no changes. In LE mode, the byte swapper is on, and the order of the byte lanes is rotated (swapped) about the center. As shown in Table 11-3, the data on CPU byte lane 0 is steered to memory byte lane 7, the data on CPU byte lane 1 is steered to memory byte lane 6, and so on. During reads, the data flows in the opposite direction over the same paths.

Table 11-3. 660 Bridge Endian Mode Byte Lane Steering

CPU Byte Lane	BE Mode Connection	LE Mode Connection
CPU byte lane 0 (MSB)	Memory byte lane 0, PCI lane 0	Memory byte lane 7, PCI lane 7*
CPU byte lane 1	Memory byte lane 1, PCI lane 1	Memory Byte lane 6, PCI lane 6*
CPU byte lane 2	Memory byte lane 2, PCI lane 2	Memory byte lane 5, PCI lane 5*
CPU byte lane 3	Memory byte lane 3, PCI lane 3	Memory byte lane 4, PCI lane 4*
CPU byte lane 4	Memory byte lane 4, PCI lane 4*	Memory byte lane 3, PCI lane 3
CPU byte lane 5	Memory byte lane 5, PCI lane 5*	Memory byte lane 2, PCI lane 2
CPU byte lane 6	Memory byte lane 6, PCI lane 6*	Memory byte lane 1, PCI lane 1
CPU byte lane 7 (LSB)	Memory byte lane 7, PCI lane 7*	Memory byte lane 0, PCI lane 0

**Note:** \* In this table, PCI byte lanes 3:0 refer to the data bytes associated with PCI\_C/BE[3:0]# when the third least significant bit of the target PCI address (PCI\_AD[29]) is 0, as coded in the instruction. PCI byte lanes [7:4] refer to the data bytes associated with PCI\_C/BE[3:0]# when PCI\_AD[29] is a 1.

### 11.3 Bit Ordering Within Bytes

The LE convention of numbering bits is followed for the memory and PCI busses, and the CPU busses are labeled in BE nomenclature. The various busses are connected to the 660 with their (traditional) native significance maintained (BE for CPU, and LE for PCI and memory), so that MSb connects to MSb and so on. The bit paths between the CPU and memory data busses are shown in Table 11-4 for both BE and LE mode operation.

Table 11-4. 660 Bit Transfer

CPU_DATA[ ]	BE Mode MEM_DATA[ ]	LE Mode MEM_DATA[ ]	CPU_DATA[ ]	BE Mode MEM_DATA[ ]	LE Mode MEM_DATA[ ]
0	7	63	32	39	31
1	6	62	33	38	30
2	5	61	34	37	29
3	4	60	35	36	28
4	3	59	36	35	27
5	2	58	37	34	26
6	1	57	38	33	25
7	0	56	39	32	24
8	15	55	40	47	23
9	14	54	41	46	22
10	13	53	42	45	21
11	12	52	43	44	20
12	11	51	44	43	19
13	10	50	45	42	18
14	9	49	46	41	17
15	8	48	47	40	16
16	23	47	48	55	15
17	22	46	49	54	14
18	21	45	50	53	13
19	20	44	51	52	12
20	19	43	52	51	11
21	18	42	53	50	10
22	17	41	54	49	9
23	16	40	55	48	8
24	31	39	56	63	7
25	30	38	57	62	6
26	29	37	58	61	5
27	28	36	59	60	4
28	27	35	60	59	3
29	26	34	61	58	2
30	25	33	62	57	1
31	24	32	63	56	0

## **11.4 Byte Swap Instructions**

The Power PC architecture defines both word and halfword load/store instructions that have byte swapping capability. Programmers will find these instructions valuable for dealing with the BE nature of this architecture. For example, if a 32-bit configuration register of a typical LE PCI device is read in BE mode, the bytes will appear out of order unless the "load word with byte swap" instruction is used. The byte swap instructions are:

- lhbrx (load half word byte-reverse indexed)
- lwbrx (load word byte-reverse indexed)
- sthbrx (store half word byte-reverse indexed)
- stwbrx (store word byte-reverse indexed)

The byte-reverse instructions should be used in BE mode to access LE devices and in LE mode to access BE devices.

## **11.5 603e CPU Alignment Exceptions In LE Mode**

The CPU does not support a number of instructions and data alignments in the LE mode that it supports in BE mode. When it encounters an unsupportable situation, it takes an internal alignment exception (machine check) and does not produce an external bus cycle. See the latest 603e CPU documentation for details. Examples include:

- LMW instruction
- STMW instruction
- Move assist instructions (LSWI, LSWX, STSWI, STWX)
- Unaligned loads and stores.

11.6 Single-Byte Transfers

Figure 11-2 is an example of byte write data a at address xxxx xxx0.

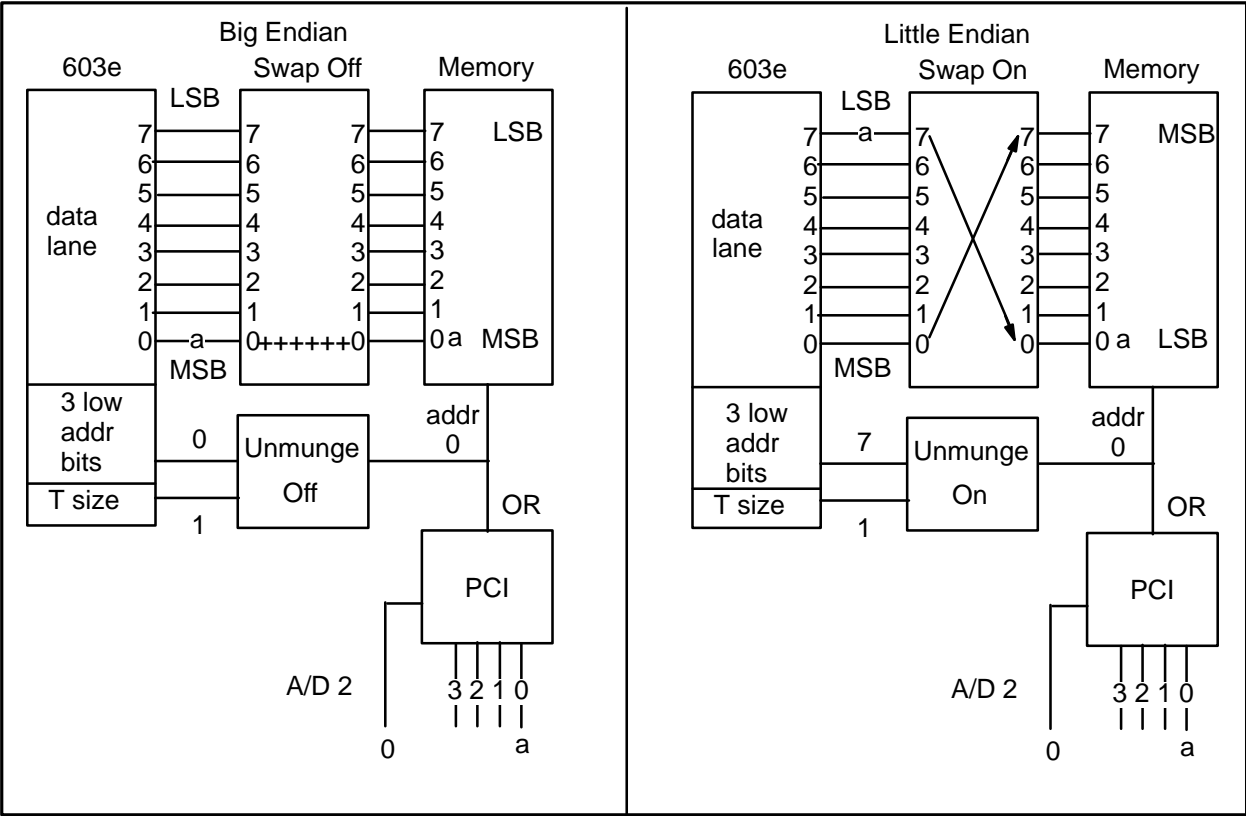


Figure 11-2. Example at Address xxxx xxx0



For single byte accesses to memory in LE mode, Table 11-6 applies.

**Table 11-6. Memory in LE Mode**

603e			603e		BYTE	BYTE	663	
A31	30	29	add	LANE	LANE*	MEM	BYTE	CAS
						LANE		ACTIVE
0	0	0	0	0	MSB 0		7	7
1	0	0	1	1	1		6	6
0	1	0	2	2	2		5	5
1	1	0	3	3	3		4	4
0	0	1	4	4	4		3	3
1	0	1	5	5	5		2	2
0	1	1	6	6	6		1	1
1	1	1	7	7	LSB 7		0	0
MUNGED			ON				SWAP	UNMUNGED

**Note:** \*At the CPU side.

For single byte accesses to PCI in BE mode, Table 11-7 applies.

**Table 11-7. PCI in BE Mode**

603e			603e	BYTE BYTE		PCI BYTE	A/D**	BE#					
A31	30	29	add	LANE	LANE	LANE	2	1	0	3	2	1	0
(0=active byte enable)													
0	0	0	0	0	MSB	0	0	0	0	1	1	1	0
1	0	0	1	1	1	1	0	0	1	1	1	0	1
0	1	0	2	2	2	2	0	1	0	1	0	1	1
1	1	0	3	3	3	3	0	1	1	0	1	1	1
0	0	1	4	4	4	0	1	0	0	1	1	1	0
1	0	1	5	5	5	1	1	0	1	1	1	0	1
0	1	1	6	6	6	2	1	1	0	1	0	1	1
1	1	1	7	7	LSB	7	1	1	1	0	1	1	1
NOT MUNGED						SWAP OFF	NOT UNMUNGED						

**Note:** \*\*AD[0:1] set to 00 for all PCI transactions except I/O cycles.



For single byte accesses to PCI in LE mode, Table 11-8 applies.

**Table 11-8. PCI in LE Mode**

663*									
603e			603e		BYTE	BYTE	PCI BYTE	A/D **	BE#
A31	30	29	add		LANE	LANE	LANE	2 1 0	3 2 1 0
(0=active byte enable)									
0	0	0	0	0	MSB	0	3	1 1 1	0 1 1 1
1	0	0	1	1	1	1	2	1 1 0	1 0 1 1
0	1	0	2	2	2	2	1	1 0 1	1 1 0 1
1	1	0	3	3	3	3	0	1 0 0	1 1 1 0
0	0	1	4	4	4	4	3	0 1 1	0 1 1 1
1	0	1	5	5	5	5	2	0 1 0	1 0 1 1
0	1	1	6	6	6	6	1	0 0 1	1 1 0 1
1	1	1	7	7	LSB	7	0	0 0 0	1 1 1 0
MUNGED					SWAP ON			UNMUNGED	

**Notes:**

\*At the CPU side.

\*\*AD[0:1] set to 00 for all PCI transactions except I/O cycles.

11.7 Two-Byte Transfers

Figure 11-4 gives an example of double byte write data ab at address xxxx xxx0.

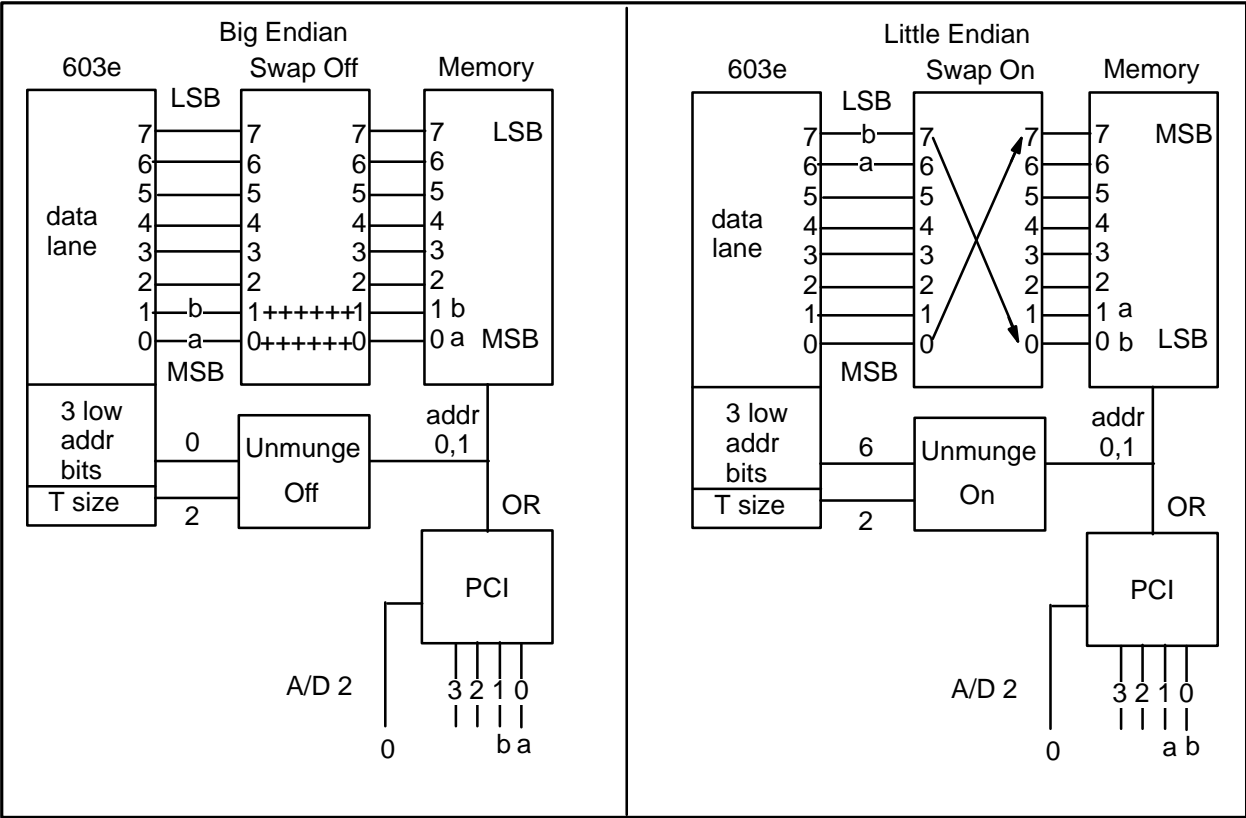


Figure 11-4. Double Byte Write Data ab at Address xxxx xxx0

Table 11-9 and Table 11-10 illustrate all cases that can occur. The columns of Table 11-9 have these meanings:

- The first column indicates target address (e.g. the address of the byte coded into a store half-word instruction).
- The next two columns show the state of the address pins for BE mode.
- The next two columns show the state of the address pins for the same target data when the machine is in LE mode.
- The remaining columns show the CASs and the PCI byte enables associated with the target data.
- The notes indicate which combinations either do not occur at the pins because of internal exceptions, or are not supported externally.

For 2-byte transfers, Table 11-9 holds:

**Table 11-9. Two Byte Transfer Information**

PROG	BE MODE		LE MODE		BE OR LE	BE OR LE		BE OR LE	
TARG	603e	BE	(x or w 110)		Target	CAS# 0:7		PCI CBE#	
ADDR	add	a29:31	Add a29:31		bytes	0	7	AD2	3210
0	0	000	6	110	0-1	0011	1111	0	1100
1	1	001	7	E 111	1-2	E 1001	1111	0	E 1001
2	2	010	4	100	2-3	1100	1111	0	0011
3	3	011	5	E 101	3-4	E 1110	0111	1	E PPPP
4	4	100	2	010	4-5	1111	0011	1	1100
5	5	101	3	E 011	5-6	E 1111	1001	1	E 1001
6	6	110	0	000	6-7	1111	1100	1	0011
7	N	NNN	1	E 001	NNN	E NNNN	NNNN	N	E NNNN

**Notes:**

**N**= not emitted by 60X because it crosses 8 bytes (transforms to 2 singles in BE, machine CH in LE)

**P**= not allowed on PCI (crosses 4 bytes)

**E**= causes exception (does not come out on 603e bus) in LE mode

Table 11-10 contains the same information as found in Table 11-9, but it is arranged to show the CAS and PCI byte enables that activate as a function of the address presented at the pins of the 603e and as a function of BE/LE mode.

**Table 11-10. Rearranged Two-Byte Transfer Information**

2 BYTE XFERS			BE		BE		LE		LE	
60X ADDRESS PINS			CAS#0:7		PCI CBE#		CAS#0:7		PCI CBE#	
0			7	A2	3210		0	7	AD2	3210
0	000		0011	1111	0	1100	1111	1100	1	0011
1	001		1001	1111	0	1001	E NNNN	NNNN	N	E NNNN
2	010		1100	1111	0	0011	1111	0011	1	1100
3	011		1110	0111	0	PPPP	E 1111	1001	1	E 1001
4	100		1111	0011	1	1100	1100	1111	0	0011
5	101		1111	1001	1	1001	E 1110	0111E	0	E PPPP
6	110		1111	1100	1	0011	0011	1111	0	1100
7	111		NNNN	NNNN	N	NNNN	E 1001	1111E	0	E 1001

**Notes:**

**N**= not emitted by 60X because it crosses 8 bytes (transforms to 2 singles in BE, machine CH in LE)

**P**= not allowed on PCI (crosses 4 bytes)

**E**= causes exception (does not come out on 603e bus) in LE mode

## 11.8 Four-Byte Transfers

Figure 11-5 gives an example of Word (4-BYTE) Write of 0a0b0c0dh AT ADDRESS xxxx xxx4.

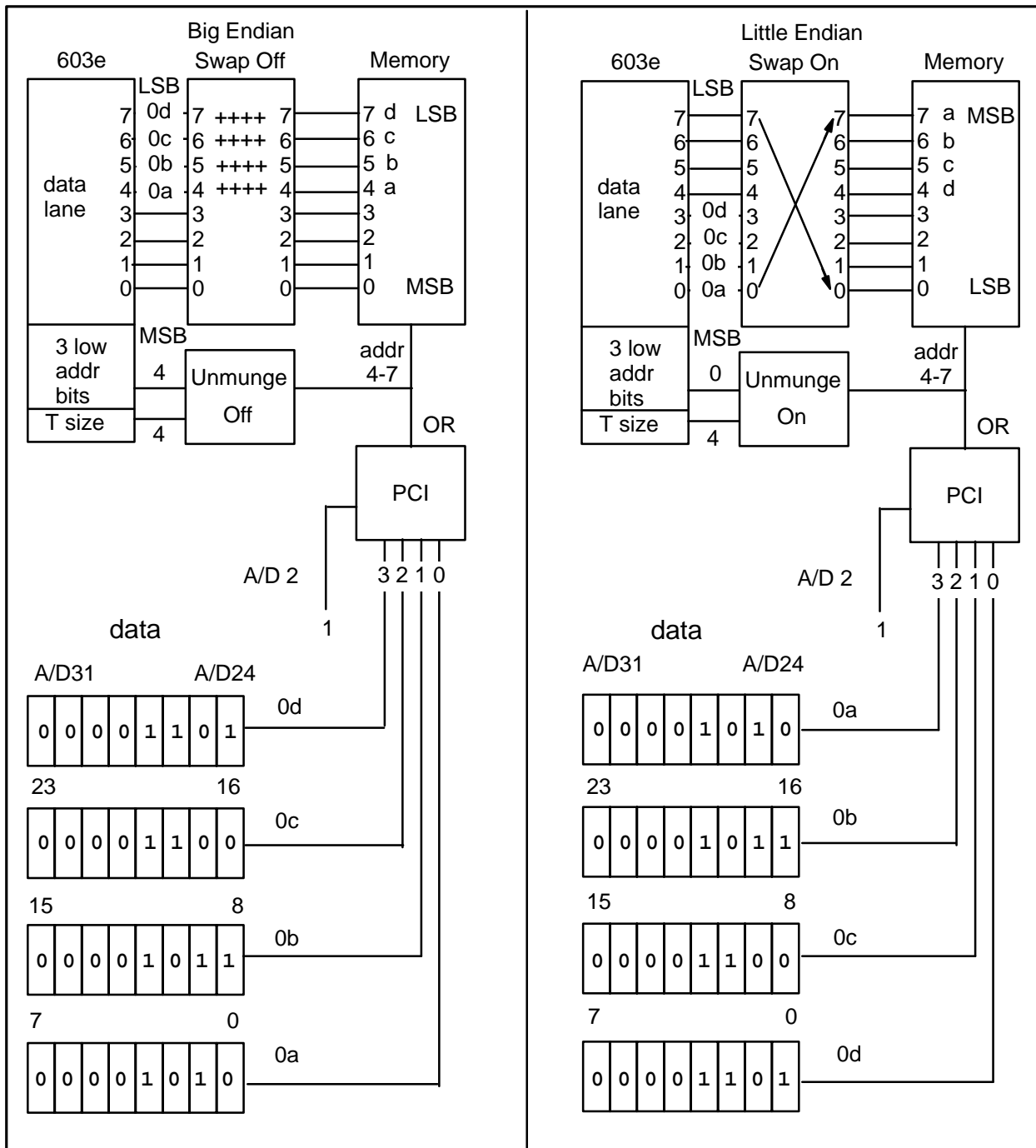


Figure 11-5. Word (4-Byte) Write of 0a0b0c0dh at Address xxxx xxx4

Table 11-11 and Table 11-12 illustrate the cases that can occur. The columns of Table 11-11 have these meanings:

- The first column indicates the target address (e.g. the address of the byte coded into a store word instruction).
- The next two columns show the state of the address pins for BE mode.
- The next two columns show the state of the address pins for the same target data when the machine is in LE mode.
- The remaining columns show the CASs and the PCI byte enables associated with the target data.
- The notes indicate which combinations either do not occur at the 603e pins because of internal exceptions, or are not supported externally.

**Table 11-11. Four-Byte Transfer Information**

PROG TARG ADDR	BE MODE 603e BE add a29:31		LE MODE (x or w 100) add a29:31		BE OR LE Target bytes	BE OR LE CAS# 0:7 0 7		BE OR LE PCI CBE# AD2 3210	
0	0	000	4	100	0-3	0000	1111	0	0000
1	1	001	5	E 101	1-4	E 1000	0111	0	E PPPP
2	2	010	6	E 110	2-5	E 1100	0011	0	E PPPP
3	3	011	7	E 111	3-6	E 1110	0001	1	E PPPP
4	4	100	0	000	4-7	1111	0000	1	0000
5	5	NNN	1	E NNN	N-N	NNNN	NNNN	1	E NNNN
6	6	NNN	2	E NNN	N-N	NNNN	NNNN	1	E NNNN
7	7	NNN	3	E NNN	N-N	NNNN	NNNN	1	E NNNN

**Notes:**

**N**= not emitted by 60X because it crosses 8 bytes (transformed into 2 bus cycles)

**P**= not allowed on PCI (crosses 4 bytes)

**E**= causes exception (does not come out on 603e bus) in LE mode

Table 11-12 contains the same information as found in Table 11-11, but it is arranged to show the CAS and PCI byte enables that activate as a function of the address presented at the pins of the 603e and as a function of BE/LE mode.

Rearranging Table 11-12 for 4-byte transfers:

**Table 11-12. Rearranged Four-Byte Transfer Information**

4 BYTE XFERS		BE		BE		LE		LE	
60X ADDRESS PINS		CAS#0:7		PCI CBE#		CAS#0:7		PCI CBE#	
		0	7	A2	3210	0	7	AD2	3210
0	000	0000	1111	0	0000	1111	0000	0	0000
1	001	1000	0111	0	PPPP	E NNNN	NNNN	0	E NNNN
2	010	1100	0011	0	PPPP	E NNNN	NNNN	0	E NNNN
3	011	1110	0001	0	PPPP	E NNNN	NNNN		E NNNN
4	100	1111	0000	1	0000	0000	1111	1	0000
5	101	NNNN	NNNN	1	NNNN	E 1000	0111	1	E PPPP
6	110	NNNN	NNNN	1	NNNN	E 1100	0011	1	E PPPP
7	111	NNNN	NNNN	1	NNNN	E 1110	0001	1	E PPPP

**Notes:**

N= not emitted by 60X because it crosses 8 bytes (transformed into 2 bus cycles)

P= not allowed on PCI (crosses 4 bytes)

E= causes exception (does not come out on 603e bus) in LE mode

X= not supported in memory controller (crosses 4-byte boundary)

## 11.9 Three byte Transfers

There are no explicit Load/Store three-byte instructions; however, three-byte transfers occur as a result of unaligned four-byte loads and stores as well as a result of move multiple and string instructions.

The TSIZ=3 transfers with address pins = 0, 1, 2, 3, 4, or 5 may occur in BE. All of the other TSIZ and address combinations produced by move multiple and string operations are the same as those produced by aligned or unaligned word and half-word loads and stores.

Since move multiples, strings, and unaligned transfers cause machine checks in LE mode, they are not of concern in the BE design.

### 11.10 Instruction Fetches and Endian Modes

Most instruction fetching is with cache on. Therefore memory is fetched eight bytes wide. Figure 11-6 shows the instruction alignment.

Example: 8 byte instruction fetch I1=abcd, I2=efgh at address xxxx xxx0

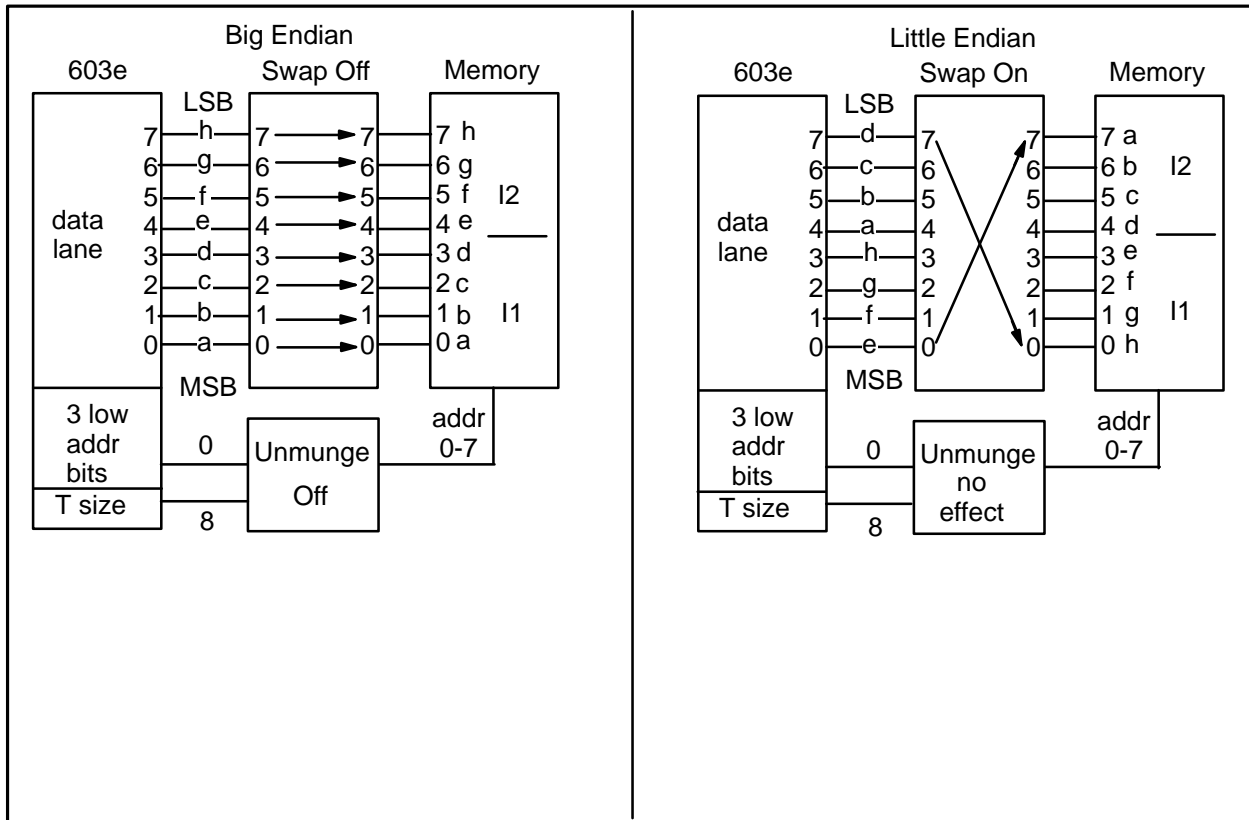


Figure 11-6. Instruction Alignment Example

It is possible to fetch instructions with 4 byte aligned transfers when the cache is turned off. In that case, the 603e does not munge the address in LE mode. The memory controller does not differentiate between instruction and data fetches, but the unmunger is ineffective because the memory is always read 8 byte wide, and data is presented on all 8 byte lanes. If the unmunger were used, the wrong instruction would be read. The net result is illustrated in Figure 11-7.

Example: 4 byte instruction fetch, I2=efgh at address xxxx xxx4

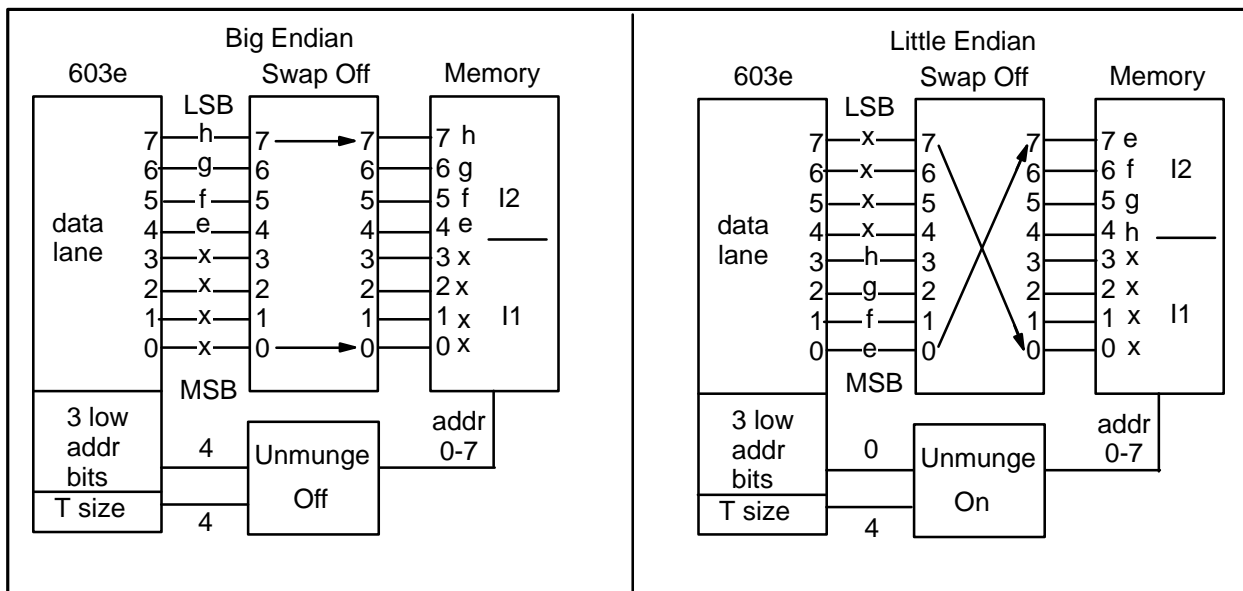


Figure 11-7. Wrong Instruction Read When Unmunger is used

### 11.11 Changing BE/LE Mode

There are two BE/LE mode controls. One is inside the 603e CPU and the other is a register bit on the motherboard. The 603e CPU interior mode is not visible to the motherboard hardware. The BE mode bit referred to in this document is the register bit on the motherboard. It is a bit in I/O space which is memory mapped just like other I/O registers. It defaults to BE mode.

The 603e CPU always powers up in the BE mode and begins fetching to fill its cache. Consequently, at least the first of the ROM code must be BE code. It is beyond the scope of this document to define how the system will know to switch to LE mode; however, great care must be made during the switch in order to synchronize the internal and external mode bits, to flush all caches, and to avoid executing extraneous code.

The following process switches the system from BE to LE mode when used in this system:

1. Disable L1 caching.
2. Disable L2 caching.
3. Flush all system caches.
4. Turn off interrupts immediately after a timer tick so no timer interrupts will occur during the next set of cycles.
5. Mask all interrupts.



6. Set the CPU state and the motherboard to LE (see Figure 11-8). Note that CPU is now in LE mode. All instructions must be in LE order.
7. Put interrupt handlers and CPU data structures in LE format.
8. Enable caches.
9. Enable Interrupts.
10. Start the LE operating system initialization.

Figure 11-8 shows the instruction stream to switch endian modes.

x	mfspr	R2,1008	;Load the HDO register
;Instructions to set the Little-Endian bit in R2			
0	sync		
4	sync		
8	sync		
C	mtspr	1008,R2	;Moves to HID0 register
10	sync		
14	sync		
18	sync		
1c	sync		
20	Store to external Endian control port (X8000 0092)		
;The above instruction must be on a double word boundary			
;So the following instruction is executed first (due to pipeline)			
24	eieio		
; To this point all instructions are in Big Endian format			
; The following instructions look the same in either Endian mode			
28	X38010138		
2C	X38010138		
...	;Enough of these instructions must be executed		
...	;to guarantee the above store has occurred.		
;			
;before any memory or I/O cycles are listed.			
xx	X38010138		

**Figure 11-8. Instruction Stream to Switch Endian Modes**

## 11.12 Summary of Bi-Endian Operation and Notes

- When the 603e CPU is in BE mode, the memory is in BE mode, and data flowing on the PCI is in BE order so that it is recorded on the media in BE order. Byte 0 is the most significant byte.
- When the 603e CPU is in LE mode, the memory is in LE mode, and data flowing on the PCI is in LE order so that it is recorded on the media in LE order. Byte 0 is the least significant byte.
- The PCI bus is addressed in the same manner that memory is when the 603e CPU runs a cycle. The unmunging in LE mode changes the effective low-order address bits (the byte enables and A/D 2). On all but I/O cycles, the two low-order A/D lines are set to zero. On PCI I/O cycles, A/D 1,0 are also transformed by the unmunge operation
- No translations are made when PCI accesses memory so that the byte with address 0 on the PCI flows to byte 0 in memory — 1 to 1, 2 to 2, and so on. For example, if BE0# and BE1# are active and A/D 2 is a 0, then memory byte lanes 0 and 1 are addressed (cas 0 and cas 1 active on writes).
- Note that the LE devices which interpret data structures in the memory require that their control data be arranged in LE order even in BE mode. For example, SCSI scripts in memory must always be arranged in LE order because that is what the device expects.
- Devices such as video may require the bytes to be swapped unless these devices have byte swap capability.

## **Section 12**

### **Electromechanical**

#### **12.1 MCM Electrical**

The electrical characteristics of the MCM IOs are obtained by reference to the electrical characteristics of the individual devices to which each IO is connected. The ratings of the MCM IOs, including the package effects, are equal to or better than the ratings of the individual devices. Please refer to the appropriate sections in the following documents:

- IBM27-82660 User's Manual
- PowerPC 603e (PID6–603e) Hardware Specifications
- TI 74LVT16245 Data Sheet
- IBM IBM041814 SRAM Data Sheet
- IDT IDT71216 TagRAM Data Sheet
- Motorola MPC970 Data Sheet.

The conditions under which the MCM operates must not exceed the Absolute Maximum Ratings of any of the individual devices.

The AC and DC Electrical Specifications, and Timing Specifications of each IO is affected by the combined characteristics of the devices that are attached to the I/O's net. For example, the characteristics of the PCI\_AD lines are a function of the sum of the characteristics of the 663 and the 664. The characteristics of the memory lines are those stated in the 660 User's Manual since these nets connect to only one die on the MCM.

For the specifications in the individual documents to apply, the MCM must be operated within the Recommended Operating Conditions of each of the individual devices.

##### **12.1.1 MCM Electrical Specifications**

The electrical characteristics of the MCM are a function of those same characteristics found in the die that it encompasses, along with the effects of the ceramic substrate which interconnect those die. Characteristics of the individual die themselves, packaged as single chip modules, can be found in their respective data sheets. A number of these data sheets have been included in Appendix G.

A composite electrical model of the MCM will not be presented here. Rather, data summarized from the chip specifications is presented, along with information on the design of the ceramic substrate. This gives the designer the information needed to model the MCM in his application if that is desired.

**12.1.2 Absolute Maximum Rating Summary**

Individual chip data sheets are the source for the following data:

- Supply voltage (Vdd). The MCM is supplied with three Vdd power supply levels. These are listed below along with the parameters that lists their maximum values:
  - Basic 3.3v supply. This level is supplied by the VDD1 pins in the MCM I/O listing (see Table 12-3). It is the 3.3v level for all chips excluding the SRAMs and Tag Ram core. The maximum value is limited to  $V_{cc}=3.9v$  by the 660 Bridge specification.
  - SRAM 3.3v supply. This level is supplied by the VDD2 pins in the MCM I/O listing (see Table 12-3). It is the 3.3v level for the four SRAMs. The maximum value is limited to  $V_{cc}=4.6v$  by the IBM SRAM specification.
  - Tag RAM core 5v supply. The IDT Tag RAM is the only 5v part on the MCM. While its core runs off of  $V_{cc}=5.0v$ , its I/O drivers are sourced from a  $V_{ccq}=3.3v$  level. By its specification, its maximum core voltage is  $V_{cc}=V_{ccq}+4.0v$ , with  $V_{cc}=3.3v$  then  $V_{ccq(max)}=7.3v$ .
- Storage temperature. Storage temperature limits are bounded by the following individual chip limits:
  - Minimum storage temperature =  $-40\text{ }^{\circ}\text{C}$ , per SRAM and clock specifications
  - Maximum storage temperature =  $125\text{ }^{\circ}\text{C}$ , per SRAM, clock, and 660 Bridge specifications.
- Input voltage. The following limits can be summarized from the component data sheets:
  - $V_{in} = -0.3v$  to  $5.5v$ , for I/O sourcing all by the MPC970 clock chip
  - $V_{in} = -0.3v$  to  $3.9v$ , for I/O sourcing the MPC970 clock chip.
- Power dissipation / operating temperature. For a discussion of power dissipation and operating temperature, see Section 12.2.

**12.1.3 Recommended Operating Conditions**

- Supply voltage. The specifications in Table 12-1 meet the requirements stated in the individual component specifications.

**Table 12-1. Component Voltage Specifications**

Level	Min.	Max.	Notes
Vdd1	3.135v	3.465v	Vdd source for all but SRAMs and Tag RAM core
Vdd2	3.135v	3.465v	Vdd source for SRAMs
Vdd3	4.750v	5.250v	Vdd source for Tag RAM core

- High level input voltage ( $V_{ih}$ )
  - $V_{ih} = 2.2v$  minimum, for all nets except those driving the MPC 970 clock
  - $V_{ih} = 2.5v$  minimum, for the MPC970 clock chip nets
- Low level input voltage ( $V_{il}$ )
  - $V_{il} = 0.8v$  maximum, for all nets except those driving the MPC970 clock
  - $V_{il} = 1.0v$  maximum, for the MPC970 clock chip nets

**12.1.4 DC Electrical Characteristics**

The following parameters are normally specified in this category:

- Output high voltage ( $V_{oh}$ )

- Output low voltage (Vol)
- Input and output leakage.

MCM I/O characteristics for these parameters depend on the chip technology that it is connected to. To determine the characteristics of a specific MCM I/O, refer to its schematic to determine chip involvement, then refer to the individual chip data sheets.

### 12.1.5 AC Electrical Characteristics

The single-chip module (SCM) data sheets for the die packaged in the MCM are contained in Appendix G. As a first approximation, it is accurate to use AC characteristics (setup and hold) specifications from the SCM data sheets when doing a system timing analysis of the MCM. The MCM package contains both the semiconductors and the wiring network interconnecting them.

Overall, a system timing budget is much improved by the MCM substrate wiring density; however, when referenced from the package I/O, MCM AC characteristics differ (in theory) from those listed in the SCM data sheets. Actually, such differences are less than 0.5 nano-seconds due to the short net lengths in the MCM ceramic substrate and to the extremely small parasitic capacitances of flip chip packaging.

In order to adjust SCM AC characteristics to values appropriate to the MCM package, subtract out the SCM package delay component of a timing parameter then add back in the corresponding delay for the MCM.

Package modeling information that allows such an analysis is found in Section 12.1.6. Note that the dynamic interfaces between the MCM and a system planar consist of the PCI port and the system memory port. Both interfaces are contained within the 660 bridge components. For this reason, only a paper model of the 660 SCM package is included here for use in package delay comparison.

### 12.1.6 SCM Package Delay Modeling

Figure 12-1 is a paper electrical model of the quad flatpack package that houses the 660 bridge die. Table 12-2 shows the ranges.

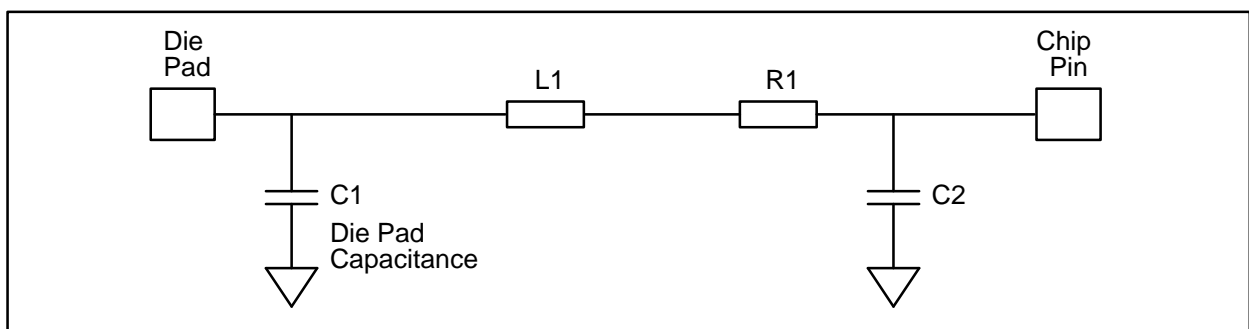


Figure 12-1. 660 Package QFP Paper Model

Table 12-2. Electrical Model Range of Values

Symbol	Condition	663	664	Unit
Inductance ( L1 )	Shortest lead	12	11	nH
	Longest lead	17	14	nH
Resistance ( R1 )	Shortest lead	.17	.13	$\Omega$
	Longest lead	.21	.17	$\Omega$
Capacitance ( C2 )	Shortest lead	1.4	1.2	pf
	Longest lead	2.4	1.9	pf

### 12.1.7 MCM Package Delay Modeling

To provide information needed to correctly model the MCM package, a paper model of the MCM package has been included in Figure 12-2. In this model, a transmission line is used to model the effects of a net trace in ceramic. The length of this transmission line is different for each signal that is considered. Refer to Table 12-3 for a listing of the net lengths for all MCM signals to which the user is expected to interface.

### 12.1.8 MCM Net Electrical Model

Figure 12-2 shows a typical electrical model of the major nets on the MCM. This model applies to the PCI, DRAM, and clock net groups. The segment labeled C4 shows the effects of the chip IO pad. The Ceramic Vias segments show the effects of vias in the MCM ceramic substrate. The Routed Signal segment shows the effects of trace lengths in the ceramic. The Solder Column segment shows the effects of the solder columns. For net lengths see Table 12-3.

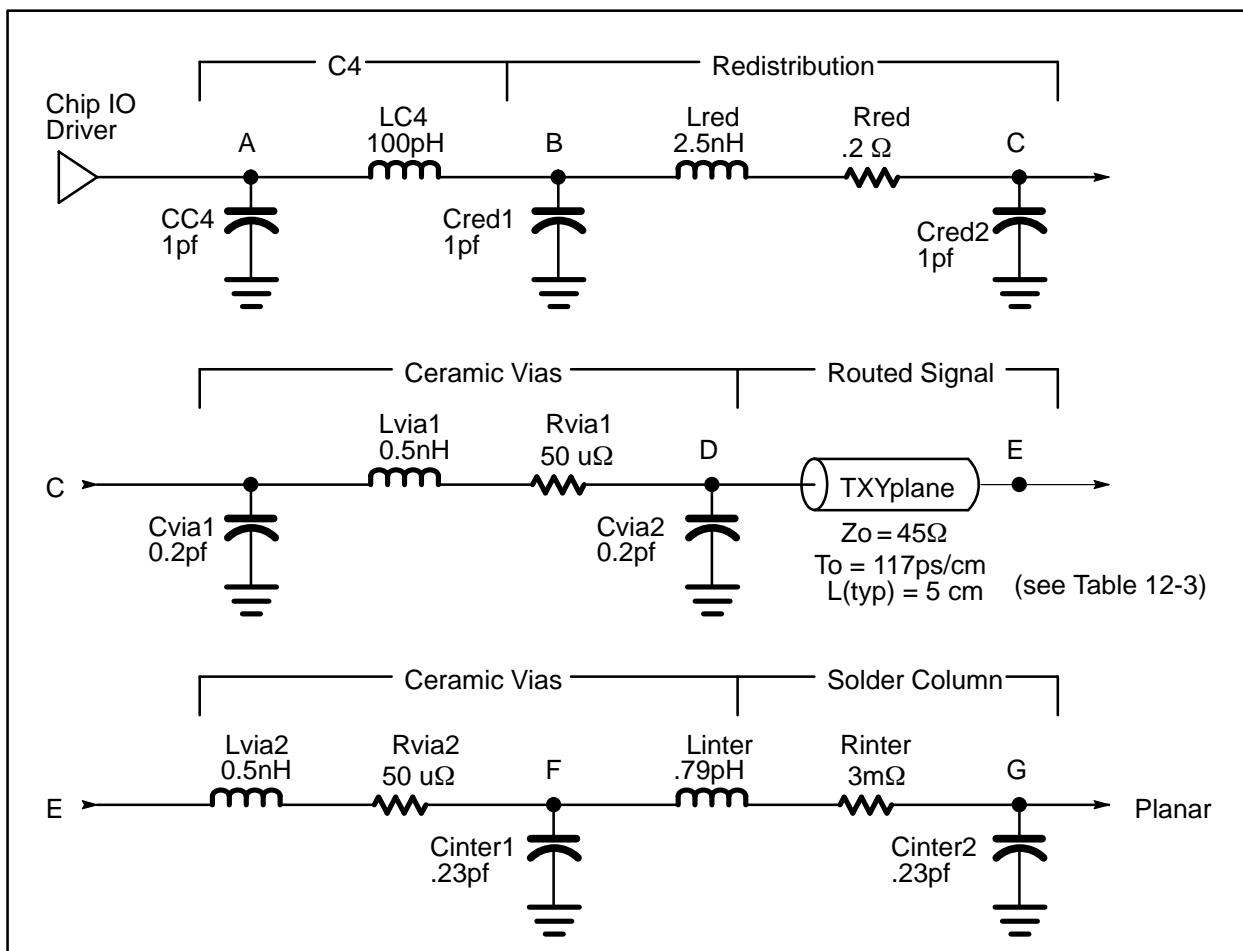


Figure 12-2. MCM Net Electrical Model

### 12.1.9 MCM Paper Model

TBD

Table 12-3. MCM Primary I/O

Net Length	Type	Name	Net Topology			
63.0000	pci	PCI_AD0	J1.T01	U2.48	U3.236	
44.2500	pci	PCI_AD1	J1.R01	U2.59	U3.237	
50.6036	pci	PCI_AD2	J1.R02	U2.46	U3.238	
47.5000	pci	PCI_AD3	J1.P01	U2.43	U3.239	
42.1036	pci	PCI_AD4	J1.N02	U2.42	U3.10	
41.0000	pci	PCI_AD5	J1.M01	U2.41	U3.11	
42.3536	pci	PCI_AD6	J1.L01	U2.40	U3.12	
39.9571	pci	PCI_AD7	J1.L02	U2.39	U3.13	
40.1036	pci	PCI_AD8	J1.K01	U2.38	U3.14	
49.7500	pci	PCI_AD9	J1.J01	U2.37	U3.15	
58.8536	pci	PCI_PAR	J1.AH01	U2.7		
40.1036	pci	PCI_AD10	J1.H01	U2.36	U3.16	
46.9571	pci	PCI_AD11	J1.G01	U2.35	U3.17	
36.1036	pci	PCI_AD12	J1.G02	U2.34	U3.18	
55.0000	pci	PCI_AD13	J1.F01	U2.33	U3.19	
39.8536	pci	PCI_AD14	J1.E01	U2.32	U3.20	
46.9571	pci	PCI_AD15	J1.E02	U2.31	U3.21	
47.9571	pci	PCI_AD16	J1.D01	U2.30	U3.40	
48.6036	pci	PCI_AD17	J1.C02	U2.29	U3.41	
44.2500	pci	PCI_AD18	J1.B03	U2.28	U3.42	
47.5000	pci	PCI_AD19	J1.C04	U2.25	U3.43	
50.6036	pci	PCI_AD20	J1.A03	U2.24	U3.44	
49.4571	pci	PCI_AD21	J1.A04	U2.23	U3.45	
46.8536	pci	PCI_AD22	J1.B05	U2.22	U3.46	
51.0000	pci	PCI_AD23	J1.A05	U2.21	U3.47	
41.3536	pci	PCI_AD24	J1.A06	U2.20	U3.48	
44.2071	pci	PCI_AD25	J1.B07	U2.19	U3.49	
47.6036	pci	PCI_AD26	J1.A07	U2.18	U3.50	
47.6036	pci	PCI_AD27	J1.A08	U2.15	U3.51	
46.4571	pci	PCI_AD28	J1.B09	U2.14	U3.62	
43.7500	pci	PCI_AD29	J1.A09	U2.13	U3.63	
47.5000	pci	PCI_AD30	J1.A10	U2.12	U3.64	
42.2071	pci	PCI_AD31	J1.C10	U2.11	U3.65	
73.5000	pci	PCI_IRDY#	J1.AB01	U2.201	U3.167	
49.1036	pci	PCI_LOCK#	J1.AE02	U2.53		
72.6036	pci	PCI_PERR#	J1.U01	U2.10		
52.8536	pci	PCI_SERR#	J1.U02	U2.71		
56.1036	pci	PCI_STOP#	J1.AA01	U2.203		
64.9571	pci	PCI_TRDY#	J1.AC02	U2.202	U3.168	
44.3536	pci	PCI_AD_OE#	J1.N14	U2.195	U3.144	
46.1036	pci	PCI_C/BE0#	J1.V01	U2.6		
53.6036	pci	PCI_C/BE1#	J1.W02	U2.5		
48.2071	pci	PCI_C/BE2#	J1.W01	U2.4		
49.2500	pci	PCI_C/BE3#	J1.Y01	U2.3		
12.5000	pci	PCI_CLK_IN	J1.K29	U2.123		
54.3536	pci	PCI_DEVSEL#	J1.AA02	U2.204		



Table 12-3. MCM Primary I/O (Continued)

Net Length	Type	Name	Net Topology			
41.6036	pci	PCI_EXT_SEL	J1.U14	U2.67	U3.153	
62.2071	pci	PCI_OL_OPEN	J1.W14	U2.64	U3.165	
50.2071	pci	PCI_OUT_SEL	J1.R14	U2.68	U3.169	
52.8536	pci	664_PCI_REQ#	J1.AF01	U2.58		
55.5000	pci	PCI_FRAME_664#	J1.AC01	U2.200		
42.2500	memory	MA0	J1.AL20	U2.190		
20.2500	memory	MA1	J1.AL22	U2.189		
36.3536	memory	MA2	J1.AL24	U2.188		
36.0000	memory	MA3	J1.AL26	U2.187		
32.7500	memory	MA4	J1.AL28	U2.186		
32.3536	memory	MA5	J1.AL30	U2.185		
31.1036	memory	MA6	J1.AK31	U2.184		
28.6036	memory	MA7	J1.AH31	U2.181		
29.8536	memory	MA8	J1.AF31	U2.180		
25.6036	memory	MA9	J1.AD31	U2.179		
39.8536	memory	MD0	J1.AN03	U3.180		
36.5000	memory	MD1	J1.AM03	U3.182		
45.7500	memory	MD2	J1.AN04	U3.183		
50.0000	memory	MD3	J1.AN05	U3.184		
54.5000	memory	MD4	J1.AM05	U3.189		
53.3536	memory	MD5	J1.AN06	U3.190		
53.2500	memory	MD6	J1.AN07	U3.193		
45.1036	memory	MD7	J1.AM07	U3.194		
39.5000	memory	MD8	J1.AN09	U3.200		
38.7500	memory	MD9	J1.AM09	U3.201		
22.5000	memory	MA10	J1.AB31	U2.178		
20.2500	memory	MA11	J1.Y31	U2.177		
35.0000	memory	MDP0	J1.AN08	U3.141		
43.2500	memory	MDP1	J1.AN14	U3.122		
48.6036	memory	MDP2	J1.AN20	U3.103		
52.0000	memory	MDP3	J1.AN26	U3.82		
58.2500	memory	MDP4	J1.AL32	U3.37		
65.5000	memory	MDP5	J1.AE32	U3.234		
61.0000	memory	MDP6	J1.W32	U3.214		
53.8536	memory	MDP7	J1.N32	U3.195		
54.1036	memory	MD10	J1.AN10	U3.202		
45.3536	memory	MD11	J1.AN11	U3.203		
41.2500	memory	MD12	J1.AM11	U3.206		
42.7500	memory	MD13	J1.AN12	U3.211		
43.1036	memory	MD14	J1.AN13	U3.212		
44.3536	memory	MD15	J1.AM13	U3.213		
46.2500	memory	MD16	J1.AN15	U3.215		
65.8536	memory	MD17	J1.AM15	U3.222		
45.6036	memory	MD18	J1.AN16	U3.223		
47.2500	memory	MD19	J1.AN17	U3.224		
45.7500	memory	MD20	J1.AM17	U3.225		

Table 12-3. MCM Primary I/O (Continued)

Net Length	Type	Name	Net Topology			
51.3536	memory	MD21	J1.AN18	U3.231		
52.1036	memory	MD22	J1.AN19	U3.232		
70.2500	memory	MD23	J1.AM19	U3.233		
51.1036	memory	MD24	J1.AN21	U3.5		
48.5000	memory	MD25	J1.AM21	U3.6		
50.2500	memory	MD26	J1.AN22	U3.7		
50.5000	memory	MD27	J1.AN23	U3.28		
48.7500	memory	MD28	J1.AM23	U3.29		
52.6036	memory	MD29	J1.AN24	U3.30		
52.3536	memory	MD30	J1.AN25	U3.35		
50.0000	memory	MD31	J1.AM25	U3.36		
52.8536	memory	MD32	J1.AN27	U3.58		
50.1036	memory	MD33	J1.AM27	U3.59		
51.9571	memory	MD34	J1.AN28	U3.60		
54.2500	memory	MD35	J1.AN29	U3.73		
53.0000	memory	MD36	J1.AM29	U3.74		
56.8536	memory	MD37	J1.AN30	U3.75		
59.6036	memory	MD38	J1.AN31	U3.76		
56.2500	memory	MD39	J1.AM31	U3.81		
58.8536	memory	MD40	J1.AL33	U3.83		
57.7500	memory	MD41	J1.AK33	U3.90		
56.0000	memory	MD42	J1.AJ32	U3.91		
56.3536	memory	MD43	J1.AJ33	U3.92		
57.6036	memory	MD44	J1.AH33	U3.93		
54.2500	memory	MD45	J1.AG32	U3.100		
56.0000	memory	MD46	J1.AG33	U3.101		
54.3536	memory	MD47	J1.AF33	U3.102		
52.9571	memory	MD48	J1.AE33	U3.108		
54.5000	memory	MD49	J1.AD33	U3.109		
49.8536	memory	MD50	J1.AC32	U3.111		
51.1036	memory	MD51	J1.AC33	U3.112		
50.7500	memory	MD52	J1.AB33	U3.113		
50.2500	memory	MD53	J1.AA32	U3.118		
49.8536	memory	MD54	J1.AA33	U3.119		
54.5000	memory	MD55	J1.Y33	U3.121		
49.8536	memory	MD56	J1.W33	U3.123		
49.8536	memory	MD57	J1.V33	U3.130		
53.3536	memory	MD58	J1.U32	U3.131		
60.7500	memory	MD59	J1.U33	U3.132		
45.5000	memory	MD60	J1.T33	U3.133		
56.8536	memory	MD61	J1.R32	U3.138		
48.6036	memory	MD62	J1.R33	U3.139		
53.2500	memory	MD63	J1.P33	U3.140		
10.2500	memory	MCE0#	J1.L32	U2.174		
10.5000	memory	MCE1#	J1.L33	U2.173		
13.3536	memory	MCE2#	J1.K33	U2.172		

Table 12-3. MCM Primary I/O (Continued)

Net Length	Type	Name	Net Topology			
6.6036	memory	MCE3#	J1.J32	U2.171		
9.5000	memory	MCE4#	J1.J33	U2.170		
7.5000	memory	MCE5#	J1.H33	U2.169		
4.7500	memory	MCE6#	J1.G32	U2.168		
7.5000	memory	MCE7#	J1.G33	U2.165		
64.7500	memory	MRE0#	J1.AL04	U2.164		
67.3536	memory	MRE1#	J1.AL06	U2.163		
65.1036	memory	MRE2#	J1.AL08	U2.162		
60.6036	memory	MRE3#	J1.AL10	U2.161		
57.0000	memory	MRE4#	J1.AL12	U2.160		
54.2500	memory	MRE5#	J1.AL14	U2.159		
52.1036	memory	MRE6#	J1.AL16	U2.158		
53.8536	memory	MRE7#	J1.AL18	U2.157		
13.8536	memory	MWE0#	J1.M33	U2.176		
12.6036	memory	MWE1#	J1.N33	U2.175		
38.7950	funct.	TEA#	J1.N10	R1.D01	U1.154	U2.137
46.2857	funct.	DRTRY#	J1.AJ01	R1.H01	U1.156	
19.2500	funct.	ROM_OE#	J1.V31	U2.47		
17.5000	funct.	ROM_WE#	J1.T31	U2.60		
39.4691	funct.	INT_60X#	J1.U16	R1.D03	U1.188	
40.7150	funct.	MCP_60X#	J1.R16	R1.H02	U1.186	
81.0192	funct.	SRAM_OE#	J1.AA22	U7.50	U8.50	U9.50, U10.50
10.2020	funct.	PLL_CFG0	J1.D25	U1.213		
16.6744	funct.	PLL_CFG1	J1.B27	U1.211		
17.1617	funct.	PLL_CFG2	J1.A27	U1.210		
15.3550	funct.	PLL_CFG3	J1.C28	U1.208		
53.8536	funct.	ROM_LOAD	J1.U20	U2.70	U3.160	
32.4704	funct.	60X_AVDD	J1.B02	U1.209		
27.2459	funct.	QACK_60X#	J1.D17	R1.E03	U1.235	
4.2367	funct.	QREQ_60X#	J1.C18	U1.31		
4.2367	funct.	X_INT_60X#	J1.F31	U2.139		
4.2367	funct.	X_MCP_60X#	J1.H27	U2.138		
4.2367	funct.	X_SRAM_OE#	J1.G26	U2.117		
19.0000	funct.	INT_TO_664	J1.B31	U2.55		
8.5866	funct.	X_PCLK_60X	J1.D19	U1.212		
5.6036	funct.	X_TAG_BCLK	J1.P29	U5.69		
21.9935	funct.	SRESET_60X#	J1.D31	U1.189		
7.3536	funct.	TAG_ADDR_13	J1.L26	U5.33		
6.0000	funct.	TAG_DATA_11	J1.M27	U5.65		
52.1036	funct.	IGN_PCI_AD31	J1.AD01	U2.57		
4.6624	funct.	X_SRAM_BCLK0	J1.Y29	U7.51		
5.1624	funct.	X_SRAM_BCLK1	J1.AF29	U8.51		
9.1624	funct.	X_SRAM_BCLK2	J1.AF05	U9.51		
8.6624	funct.	X_SRAM_BCLK3	J1.Y05	U10.51		
1.7500	funct.	X_663_CPU_CLK	J1.F05	U3.157		

Table 12-3. MCM Primary I/O (Continued)

Net Length	Type	Name	Net Topology			
10.1036	funct.	X_664_CPU_CLK	J1.H29	U2.121		
8.6036	funct.	X_CPU_RDL_OPEN	J1.G10	U3.148		
22.2500	funct.	NMI_FROM_ISABRDG	J1.A31	U2.56		
7.2071	funct.	POWER_GOOD/RESET#	J1.A28	U2.156		
7.2071	funct rw	HRESET#	J1.E32	R1.D04	U1.214	
35.6584	rw	TCK	J1.B25	R1.J04	U1.201	
41.2051	rw	TDI	J1.A25	R1.J05	U1.199	
14.6846	rw	TDO	J1.C26	U1.198		
34.9543	rw	TMS#	J1.A24	R1.J03	U1.200	
58.3719	rw	CKSTP_OUT#	J1.AJ02	R1.J01	U1.216	
58.3719	rw p_up	TRST#	J1.C24	R1.H04	U1.202	
29.4987	p_up	TT0	J1.A19	U1.191	U2.150	
27.6398	p_up	TT1	J1.B19	U1.190	U2.152	
15.3133	p_up	TT2	J1.A20	U1.185		
28.0788	p_up	TT3	J1.A21	U1.184	U2.126	
5.3536	p_up	SHD#	J1.D27	U2.141		
51.7310	p_up	SMI#	J1.AL02	R1.F05	U1.187	
11.6036	p_up	TAG_CS2	J1.H31	U5.76		
21.5000	p_up	TT2_664	J1.C14	U2.153		
35.9554	p_up	DBG_60X#	J1.G08	U1.26	U2.140	
70.1036	p_up	L2_CLAIM#	J1.AL01	U2.132		
21.8536	p_up	663_TEST#	J1.G14	U3.155		
49.6036	p_up	664_TEST#	J1.J04	U2.155		
30.8536	p_up	TAG_MATCH	J1.K31	U2.142	U5.50	
22.0000	p_up	BG_MASTER#	J1.N20	U2.135		
56.1497	p_up	SRAM_ADSP#	J1.AK03	U7.1	U8.1	U9.1, U10.1
5.0000	p_up	TAG_PWRDN#	J1.M31	U5.77		
48.1036	p_up	MWS_P2MRXS	J1.C30	U2.66	U3.152	
40.9072	p_up	664_STOP_CLK_EN	J1.B29	R1.F04	U2.151	
30.7884	p_down	TT4	J1.A23	U1.180	U2.136	
24.5780	p_down	GBL#	J1.A12	U1.1	U2.120	
43.8399	p_down	SRAM_CS#	J1.AD29	U7.5	U8.5	U9.5, U10.5
19.5000	p_down	TAG_CS1#	J1.C33	U5.75		
72.1036	p_down	OE_245_B	J1.D29	U6A.25A	U6A.48A	U6B.25B, U6B.48B
32.2071	p_down	DIR_245_A	J1.G22	U6A.1A	U6A.24A	
43.6036	p_down	DIR_245_B	J1.L22	U6B.1B	U6B.24B	
20.7500	p_down	TAG_SFUNC	J1.C32	U5.22		
52.6036	p_down	CRS_C2PWXS	J1.A29	U2.65	U3.151	
39.1036	p_down	663_MIO_TEST	J1.A30	U3.156		
49.2500	p_down	664_MIO_TEST	J1.K05	U2.154		
13.7500	clk	XTAL1	J1.U30	U4.12		

Table 12-3. MCM Primary I/O (Continued)

Net Length	Type	Name	Net Topology			
11.7071	clk	XTAL2	J1.T29	U4.13		
32.5000	clk	FRZ_CLK	J1.B11	U4.3		
18.6036	clk	FRZ_DATA	J1.A22	U4.5		
8.5866	clk	PCLK_60X	J1.E18	U4.34		
66.3536	clk	TAG_BCLK	J1.N30	U4.36		
29.9142	clk	CLK_EXT_FB	J1.A17	U4.14		
27.0000	clk	CLK_FB_SEL	J1.A15	U4.9		
27.5000	clk	CLK_PLL_EN	J1.D33	U4.7		
67.5000	clk	SRAM_BCLK0	J1.W30	U4.44		
66.5000	clk	SRAM_BCLK1	J1.AE30	U4.46		
9.1624	clk	SRAM_BCLK2	J1.AG04	U4.48		
8.6624	clk	SRAM_BCLK3	J1.AA04	U4.50		
22.6036	clk	CLK_COM_FRZ	J1.A18	U4.6		
29.6036	clk	CLK_REF_SEL	J1.B13	U4.8		
24.4571	clk	CLK_TTL_CLK	J1.A26	U4.11		
28.0000	clk	CLK_VCO_SEL	J1.A13	U4.52		
1.7500	clk	663_CPU_CLK	J1.G04	U4.38		
63.6036	clk	664_CPU_CLK	J1.G30	U4.42		
24.1036	clk	CLK_BCLK_DIV0	J1.C16	U4.31		
22.2500	clk	CLK_BCLK_DIV1	J1.B17	U4.27		
26.3536	clk	CLK_FRZ_STROBE	J1.F33	U4.4		
14.8536	clk	CLK_MPC601_CLKS	J1.K27	U4.40		
25.3536	clk	CLK_MR/TRISTATE	J1.A16	U4.2		
61.8536	clk	664_PCI_CLK	J1.J30	U4.18		
20.6036	clk	CLK_PCI_DIV0	J1.B21	U4.20		
20.0607	clk	CLK_PCI_DIV1	J1.C20	U4.26		
38.1036	clk	USER_PCICLK1	J1.AE01	U4.16		
29.5000	clk	USER_PCICLK2	J1.N01	U4.21		
42.8536	clk	USER_PCICLK3	J1.C01	U4.23		
31.1036	clk	USER_PCICLK4	J1.A11	U4.25		
25.4571	clk	USER_PCICLK5	J1.A14	U4.29		
29.7500	clk	USER_PCICLK6	J1.E33	U4.32		
3.0000	daisy	DAISY01	J1.D05			
3.0000	daisy	DAISY01	J1.E04			
2.5000	daisy	DAISY02	J1.D07			
2.5000	daisy	DAISY02	J1.E06			
2.5000	daisy	DAISY03	J1.D09			
2.5000	daisy	DAISY03	J1.E08			
2.5000	daisy	DAISY04	J1.D11			
2.5000	daisy	DAISY04	J1.E10			
2.5000	daisy	DAISY05	J1.D13			
2.5000	daisy	DAISY05	J1.E12			
2.5000	daisy	DAISY06	J1.E30			
2.5000	daisy	DAISY06	J1.F29			

Table 12-3. MCM Primary I/O (Continued)

Net Length	Type	Name	Net Topology			
3.0000	daisy	DAISY07	J1.E28			
3.0000	daisy	DAISY07	J1.F27			
2.5000	daisy	DAISY08	J1.E26			
2.5000	daisy	DAISY08	J1.F25			
3.2500	daisy	DAISY09	J1.E24			
3.2500	daisy	DAISY09	J1.F23			
4.3536	daisy	DAISY10	J1.E22			
4.3536	daisy	DAISY10	J1.F21			
3.0000	daisy	DAISY11	J1.AK05			
3.0000	daisy	DAISY11	J1.AK07			
3.0000	daisy	DAISY12	J1.AK09			
3.0000	daisy	DAISY12	J1.AK11			
3.2500	daisy	DAISY13	J1.AK13			
3.2500	daisy	DAISY13	J1.AK15			
4.0000	daisy	DAISY14	J1.AK17			
4.0000	daisy	DAISY14	J1.AK19			
3.0000	daisy	DAISY15	J1.AK21			
3.0000	daisy	DAISY15	J1.AK23			
3.0000	daisy	DAISY16	J1.AD03			
3.0000	daisy	DAISY16	J1.AF03			
3.0000	daisy	DAISY17	J1.Y03			
3.0000	daisy	DAISY17	J1.AB03			
3.0000	daisy	DAISY18	J1.T03			
3.0000	daisy	DAISY18	J1.V03			
3.0000	daisy	DAISY19	J1.M03			
3.0000	daisy	DAISY19	J1.P03			
3.0000	daisy	DAISY20	J1.H03			
3.0000	daisy	DAISY20	J1.K03			
3.0000	daisy	DAISY20	J1.K03			

## 12.2 MCM Thermal

Figure 12-3 contains a simplified drawing of the construction of the MCM. The individual devices are mounted to the ceramic substrate, which is mounted to the circuit board by solder columns. The cap covers the devices and the top of the substrate, and is filled with a thermal grease. The cap is non-hermetically sealed to the substrate.

As shown in Figure 12-4, the major heat flow path is from the devices to the thermal grease, to the aluminum cap, and to ambient. A heat sink can be attached to the cap if required.

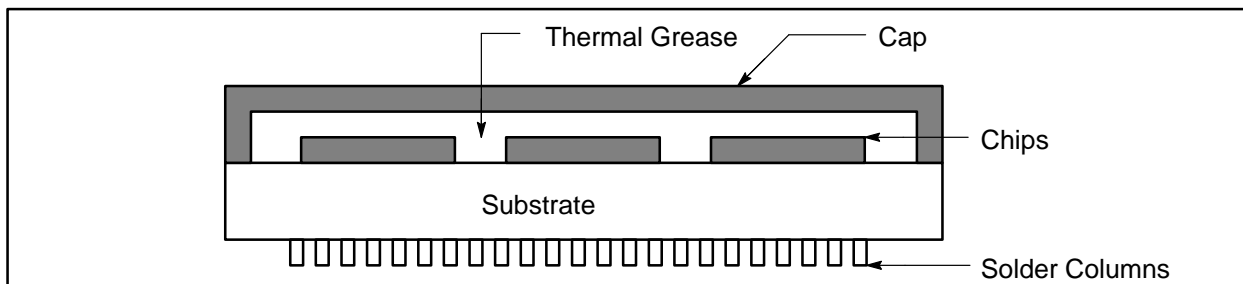


Figure 12-3. MCM Thermal Paths

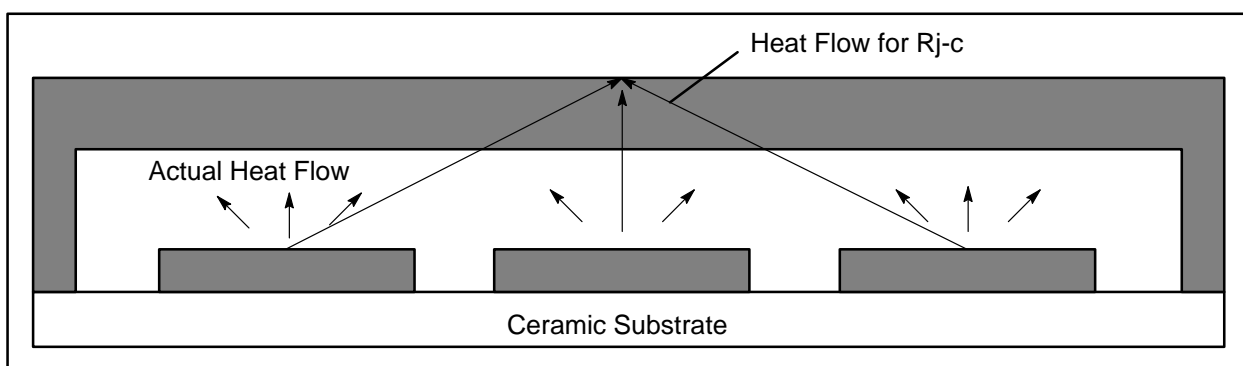


Figure 12-4. MCM Heat Flows

### 12.2.1 MCM Cooling Requirements

The objective of this section is to provide information needed when choosing the proper heatsink for an MCM application. Table 12-4 lists the maximum junction temperatures at which individual die meet their stated performance specifications. MCM cooling requirements are driven by these values. Of these values, the  $T_{j(max)} = 85^{\circ}\text{C}$  requirement placed on the 660 Bridge chipset (82663 and 82664) is the most demanding. A design that keeps the chipset at or below  $85^{\circ}\text{C}$  can easily meet the maximum allowed temperature specifications associated with each of the other die on the MCM. In fact, when this temperature specification is met, all other die on the MCM will be kept at a junction temperature of  $T_j < 90^{\circ}\text{C}$ . The approach that follows is designed to meet this goal.

Table 12-4. MCM Die Thermal Specifications

	603e	663	664	'245	MPC 970	SRAM	Tag RAM
Maximum Junction Temperature (°C)	105	85	85	105	120	105	150
Power Dissipation, Typical (W)	3.2	0.5	1.0	0.21	1.08	0.78	0.53
Power Dissipation, Max (W)	4.0	0.6	1.3	0.26	1.35	0.97	0.66
Typical MCM Total Pd = 9.85 W Maximum MCM Total Pd = 12.3 W							

### 12.2.2 Module-Level Thermal Analysis

The MCM is made up of 12 individual die, each with its own power and thermal path resistances. Thermal modeling has shown that there is little loss in accuracy if this complex thermal arrangement is modeled as single power source driving through a single  $R_{int}$  to the MCM cap. The power source would have a value equal to the sum of the individual die powers ( $P_{mod}$ ), and the module-level thermal resistance ( $R_{int}$ ) would be that of the MCM package taken as a whole. Module power would then be conducted out through the thermal resistance between the cap and ambient air ( $R_{ext}$ ). Given this modeling approach,

$$T_j = (R_{int} \times P_{mod}) + (R_{ext} \times P_{mod}) + T_{amb}$$

The module level internal thermal resistance is as follows:

$$R_{int} = 0.35 \text{ } ^\circ\text{C/W}$$

The external resistance is the sum of the heat sink resistance  $R_{(hs)}$  and the interface resistance between the heat sink and the cap  $R_{(c-hs)}$ ,

$$R_{ext} = R_{(hs)} + R_{(c-hs)}$$

Table 12-5 lists values of  $R_{(c-hs)}$  for two typical interface adhesives. In this analysis, a typical value of  $R_{(c-hs)} = 0.1 \text{ } ^\circ\text{C/W}$  is used.

Table 12-5. Thermal Resistance From Cap to Heat Sink

Parameter	Chromerics T405 Reworkable Adhesive	Al Epoxy 7655 Non-Reworkable
Thermal Conductivity (K, Watts per meter-deg Kelvin)	.4 W/mk	1.06 W/mK
Thickness	0.14 mm	0.1 mm
$\Theta_{c-hs}$ (Thermal Resistance, Cap to Heat Sink)	0.18 $^\circ\text{C/W}$	0.05 $^\circ\text{C/W}$

### 12.2.3 Heatsink Performance Requirements [ $R_{(hs)}$ ]

Using a maximum module power of  $P_{mod} = 12.31 \text{ W}$ , along with the proposed die junction temperature of  $T_j = 85 \text{ } ^\circ\text{C}$ , a relationship between the ambient temperature ( $T_{amb}$ ) and the required heatsink performance is derived. This is presented in Table 12-6 with values given for both  $R_{(ext)}$  and  $R_{(hs)}$ .  $R_{(hs)}$  has been calculated assuming  $R_{(c-hs)} = 0.1 \text{ } ^\circ\text{C/W}$ .



Table 12-6. Heatsink Performance Requirements<sup>1</sup>

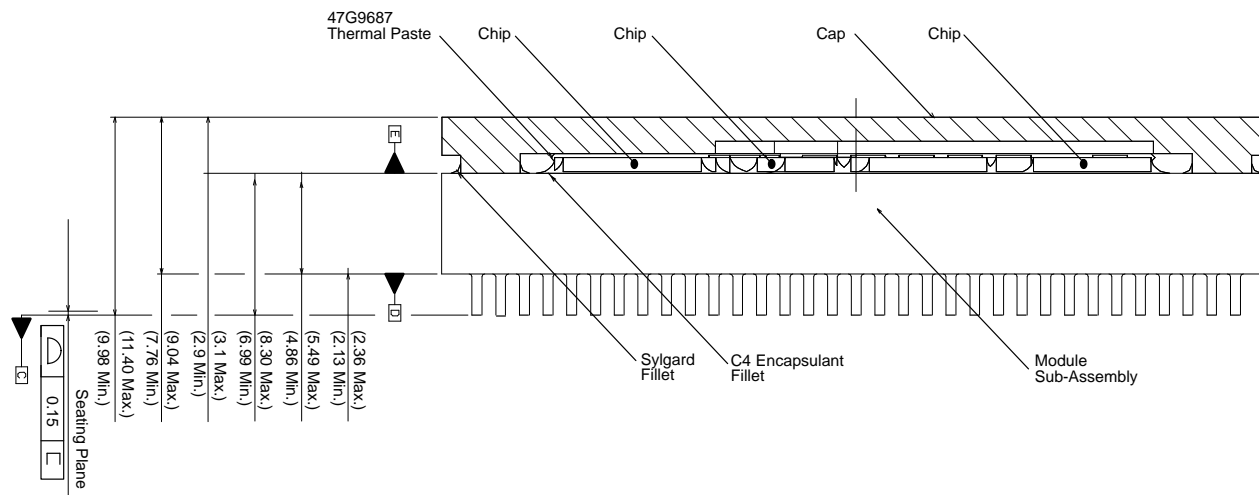
Tamb	Tj–Tamb	R(ext)	R(hs)
25	60	4.52	4.42
30	55	4.11	4.01
35	50	3.71	3.61
40	45	3.30	3.20
45	40	2.90	2.80
50	35	2.49	2.39
55	30	1.67	1.57
60	25	1.27	1.17

**Note:** Pmod = 12.31 W  
Tj = 85 °C  
Rint = 0.35 °C/W

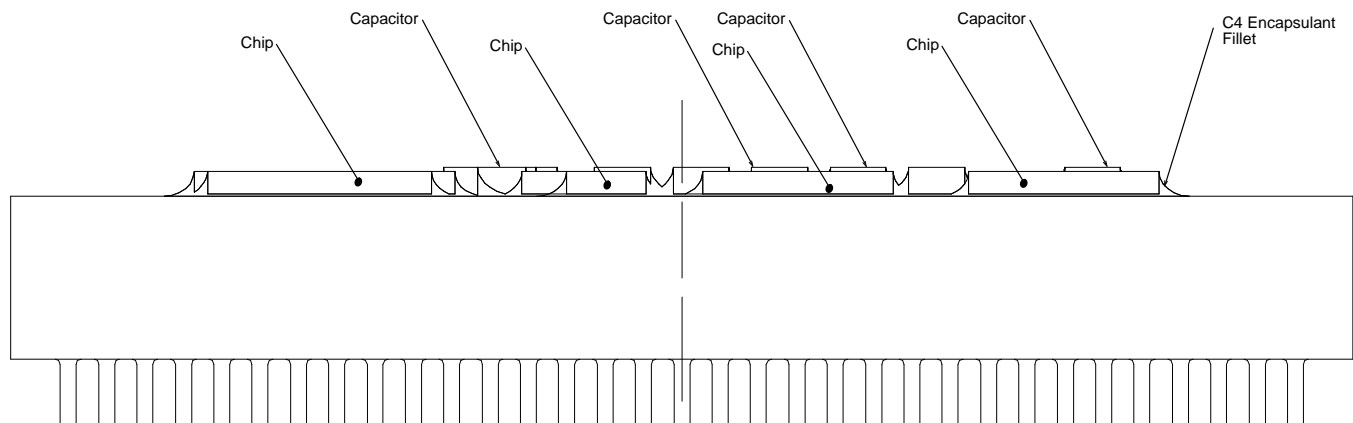
The thermal resistance values listed in Table 12-6 can be obtained from commercially available high performance finned heat sinks. In order to determine the applicability of a specific heatsink, a full three-dimensional modeling analysis of the system package design is recommended.

## 12.3 MCM Mechanical Drawings

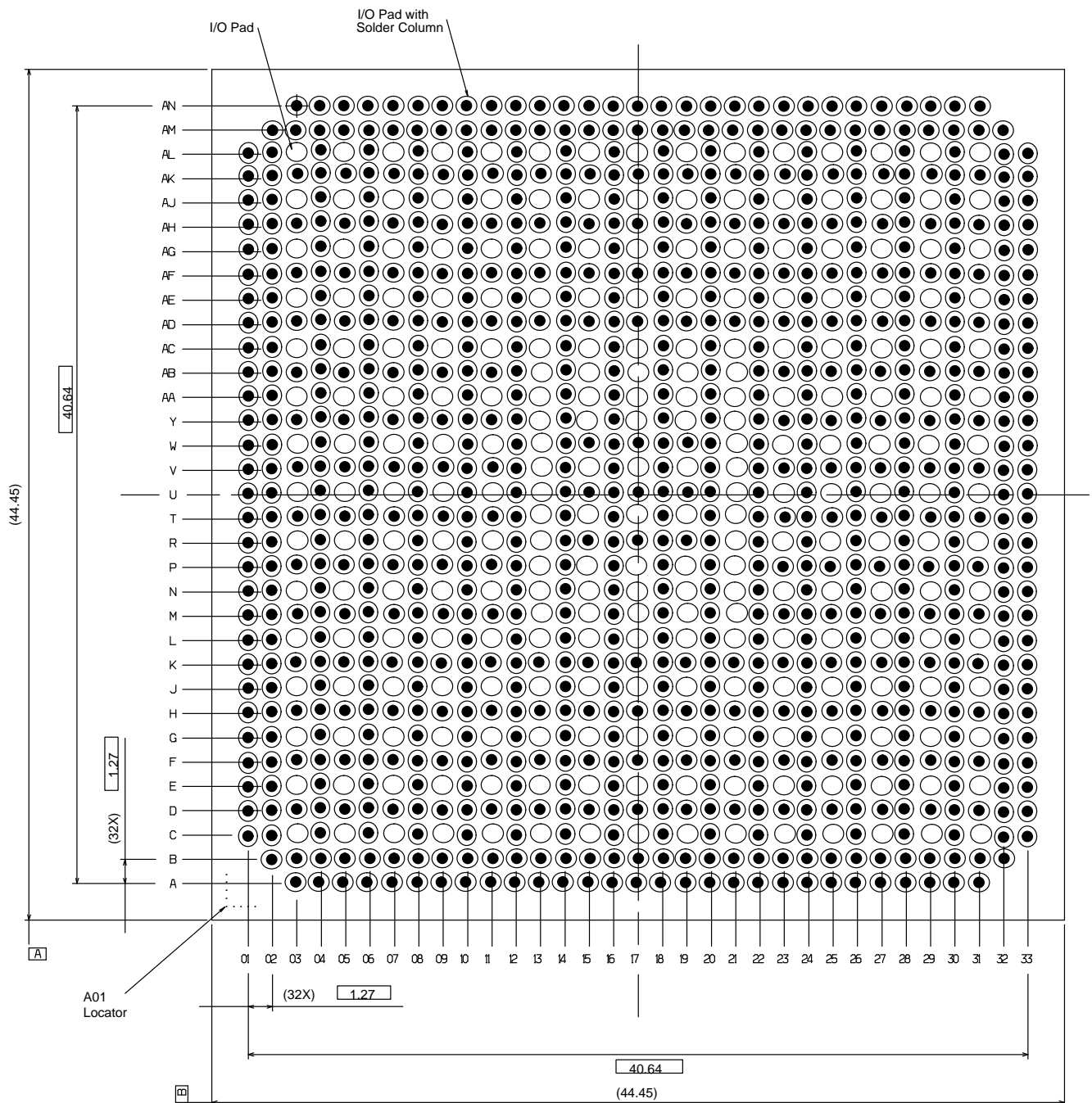
### 12.3.1 MCM with Cap



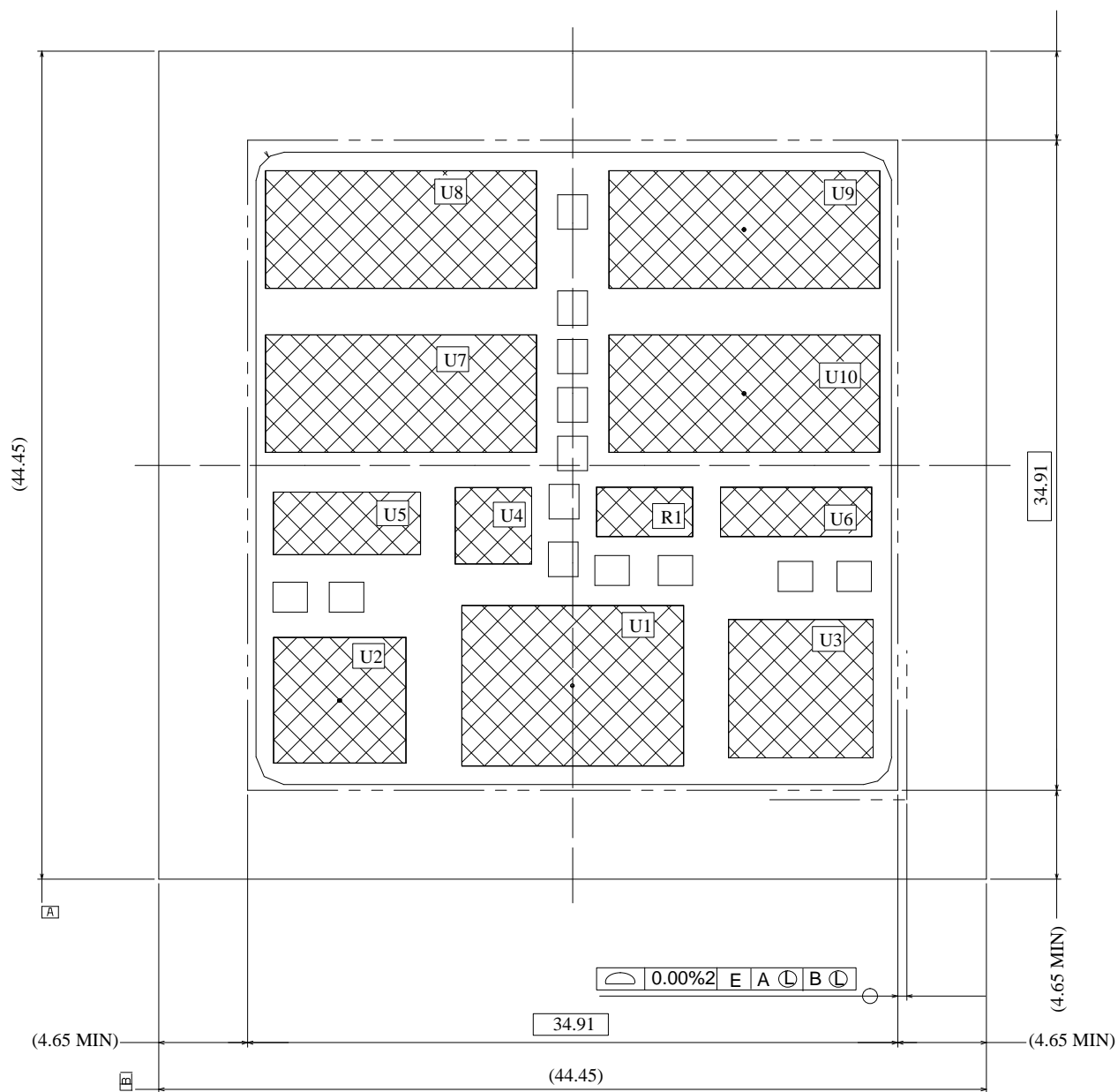
### 12.3.2 MCM Without Cap



## 12.3.3 MCM Solder Columns



## 12.3.4 MCM Component Placement



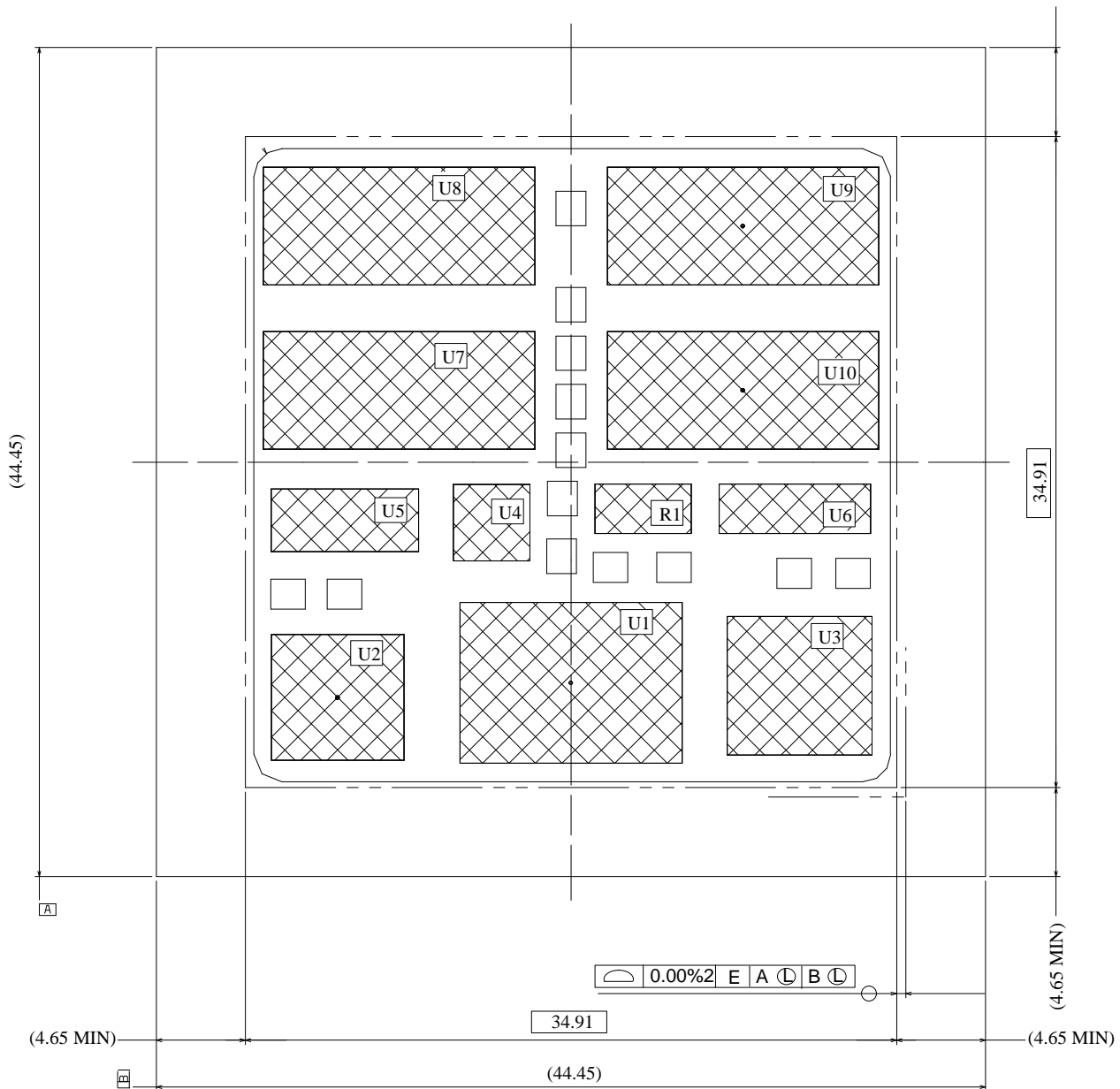
## **Section 13**

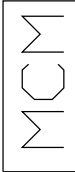
### **MCM Schematics**

This section contains the schematics of the MCM. For schematics of the example planar, see Appendix F.

The schematics are numbered separately from the rest of the MCM technical specification.

## 13.1 Component Placement

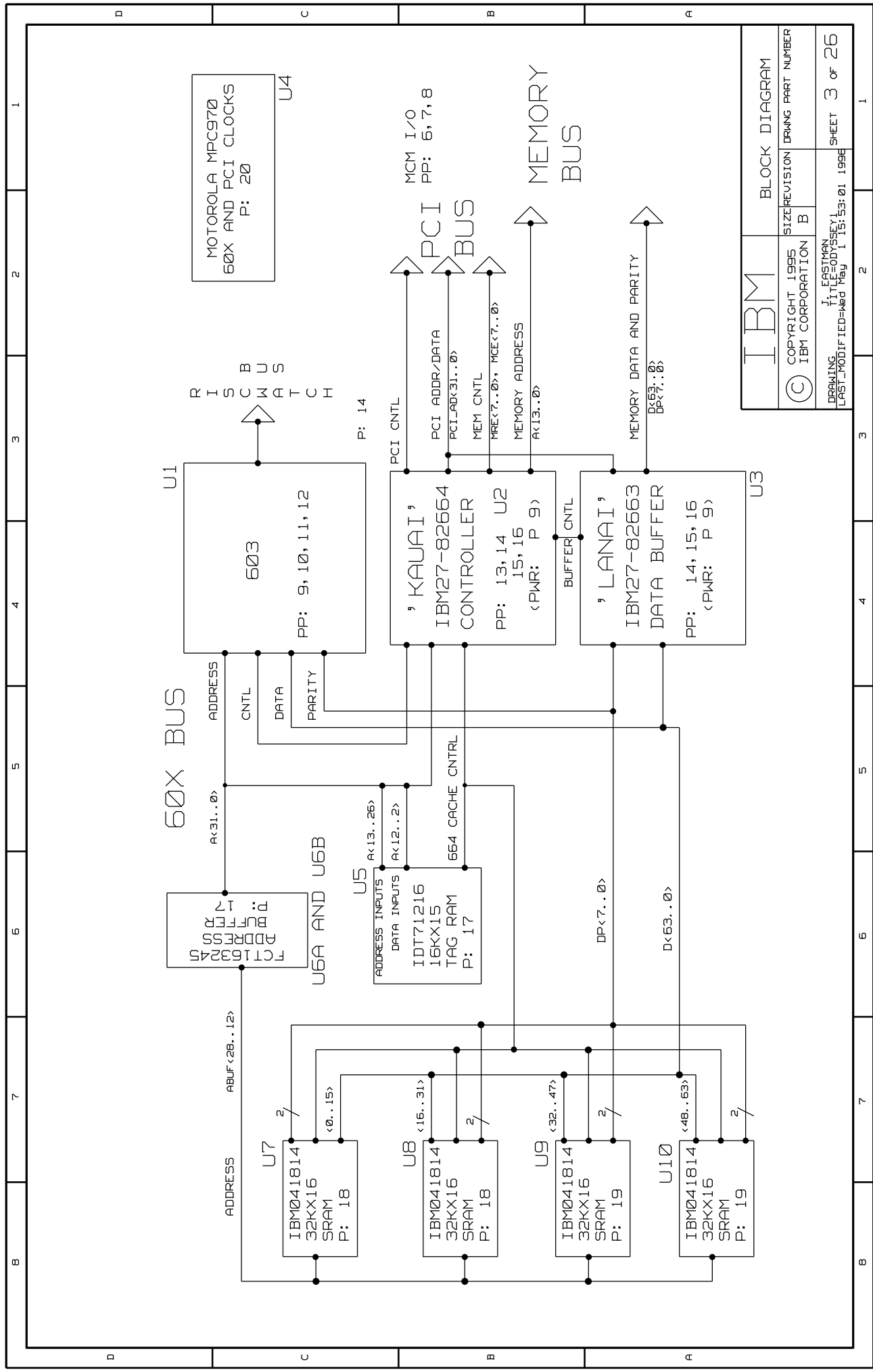




8	7	6	5	4	3	2	1
MCM TEST AND STRESS ENGINEERING				POUGHKEEPSIE, NY			
ODYSSEY							
MULTI-CHIP MODULE (MCM)							
VERSION 5.1							
MAY 2, 1996							
THIS DOCUMENT CONTAINS PRELIMINARY INFORMATION AND IS SUBJECT TO CHANGE WITHOUT NOTICE. IBM ASSUMES NO RESPONSIBILITY OF LIABILITY FOR ANY USE OF THE INFORMATION CONTAINED HEREIN. NOTHING IN THIS DOCUMENT SHALL OPERATE AS AN EXPRESS OF IMPLIED LICENSE OR INDEMNITY UNDER THE INTELLECTUAL PROPERTY RIGHTS OF IBM OR THIRD PARTIES. NO WARRANTY OR GUARANTEE IS GIVEN CONCERNING ANY INFORMATION CONTAINED IN THIS DOCUMENT. TO ENSURE THAT YOU HAVE THE LATEST VERSION OF THIS DOCUMENT, CONTACT ONE OF THE INDIVIDUALS LISTED.							
CONTACT: JED EASTMAN TOM BARDSLEY							
PPC603 512K SYNCH CACHE 663, 664 PCI, MEMORY CNTL							
IBM							
COPYRIGHT 1995 IBM CORPORATION		SIZE/REVISION B		DRAWING J. EASTMAN TITLE=ODYSSEY1 LAST_MODIFIED=Thu May 9 13:41:27 1996		DRAWING PART NUMBER SHEET 1 of 26	
8	7	6	5	4	3	2	1

8	7	6	5	4	3	2	1
D	CONTENTS			CHANGE HISTORY:			
	1. COVER 2. CONTENTS, DESCRIPTION, ECS 3. BLOCK DIAGRAM 4. PHYSICAL DESIGN INFORMATION 5. CONFIGURATION NOTES 6. MCM PINS I - USER INTERFACE 7. MCM PINS II - TEST INTERFACE 8. MCM PINS III - POWER AND GROUND 9. POWER AND DECOUPLING 10. PROCESSOR DATA BUS 11. PROCESSOR DATA BUS 12. PROCESSOR - RISCWATCH, CLOCKS, CONFIGS AND CNTL 13. 664 CONTROLLER - CENTRAL CONTROL PORTION 14. 663/664 BUFFER CONTROL AND INTERFACE 15. 663/664 PCI BUS SECTION AND TEST POINTS 16. 664 BUFFER DATA 17. TAG RAM AND 16245 BUFFERS 18. SRAM 19. SRAM 20. CLOCK 21. CROSS REFERENCE 22. CROSS REFERENCE 23. BLANK PAGE			07/20/95 DRAFT VERSION 1.0 DISTRIBUTED FOR REVIEW 08/03/95 PINS ASSIGNED 08/06/95 PINS RELEASED 08/06/95 VERSION 2.0 RELEASED BROUGHT 664 PCI CLOCK OFF AND BACK ON BROKE OUT PAL 1/0 FOR CONTINGENCY CHANGED USER INTERFACE FROM USER I/O MAJOR POWER PIN ASSIGNMENT CHANGES ADDED PCI-PAR AND TAG-DTLY MODULE 1/0 SRAM-ADSP* USER INTERFACE FROM TEST INTF VERSION 2.1 RELEASED REASSIGNMENT OF DECOUPLING CAPACITORS: VDD1=7, VDD2=2, VDD3=1 REASSIGNMENT OF PINS FOR FOLLOWING NETS: T14: J88 - E28 C160K* PAR - C08 T15: J89 - E28 C160K* PAR - C08 T12: C08 - E14 TEST* K09 - A12 T11: F87 - E16 C500-60K* H15 T10: F89 - F15 CHANGED VOLTAGE: PIN 140 TO DRG* FROM RESVUD ADDITION OF CAPACITORS C11 THRU C15 VERSION 2.2 RELEASED BROUGHT 664 BUS CLOCKS OFF AND BACK ONTO MODULE FOR TUNING ADDED CAPACITORS C14 AND C15 - 15 TOTAL, NOT 15 VERSION 2.3 RELEASED NUMEROUS PIN SHAPS TO EASE BOARD WIRING NAMED DAISY CHAIN NETS MODIFIED LANAI MODULE PINS 111 AND 231 NOW GND AND VDD REASSIGNMENT OF NUMEROUS RESISTORS CHANGED SEVERAL PULLUPS TO 5.0 V FROM 3.3 V MODIFIED 5.0V DAISY CHAIN NETS TO MODULE PIN MOD PIN 322 SUGGESTED FROM VDD1 TO VDD3 VERSION 3.0 RELEASED CHANGED TO MPC970 FROM MPC972 - MAJOR CHANGES TO MODULE 1/0 BROUGHT TAG(11) AND A13 OF TAG OFF MODULE FOR 256K OPTION BROUGHT VOLTAGE CHANGES TO MODULE 1/0 BROKE CPU-LD OPEN NET FOR 100 OHM SERIES RESISTOR FOR LANAI 2.0 VERSION 3.1 RELEASED WIRING OF LEGS CHANGED WIRING OF LEGS CHANGED (X-SRAM-BCLXX) VERSION 3.2 RELEASED ECS BASED ON EMULATOR BOARD DERUG VERSION 3.3 RELEASED 664 TEST* AND 663 TEST* PULL UP RESISTORS MOVED TO TAG RAM 663 PARITY BUSES CORRECTED VERSION 4.0 RELEASED MAJOR CHANGES FROM OUT OF MODULE TO EASE BOARD WIRING VERSION 4.1 RELEASED PINOUT MODIFIED - 3 NETS AFFECTED: MPC-60K*, INT-60K*, AND SRAM-OE* INITIAL PARTS BUILT ACCORDING TO VERSION 5.0 VERSION 5.1 RELEASED WIRING OF CAPS C11, C12, AND C13 NOMENCLATURE, NOT PACKAGED PINS ALL CHIPS WIRING ACCORDING TO 5.1, NOT 5.0 ALL CHIPS TO BE EITHER REDISTRIBUTED OR NON-REDISTRIBUTED KODIAK ABILITY TO USE EITHER REDISTRIBUTED OR NON-REDISTRIBUTED KODIAK			
B	DESIGN DESCRIPTION:			TOTAL PINS: 1077 (PADS)  SIGNAL: 446 VDD1: 64 DAISY CHAIN: 40 VDD2: 54 NC: 326 VDD3: 9 VSS: 128  UNUSED PINS CAN BE DEPOPULATED OF COLUMNS.			
	THE ODYSSEY MCM IS A POWERPC 603E-BASED MULTI-CHIP MODULE WITH 256K OF L2 CACHE. THE LANAI AND KAUAI CHIPS (660 CHIPSET) PROVIDE THE MEMORY CONTROL AND PCI BRIDGE FUNCTION. 256K OF L2 CACHE IS PROVIDED BY 4 64KX18 SRAM DIE. A DISCRETE TAG RAM IS USED IN THE DESIGN. A MOTOROLA MPC970 IS ALSO JOINED TO THE MCM. A CUSTOM RESISTOR CHIP IS USED ALSO.  ODYSSEY IS BASED ON 660 CHIPSET REFERENCE PLATFORM ALL CHIPS OPERATE AT 3.3 V EXCEPT TAG RAM, WHICH OPERATES AT 5.0 V (VDD3).						
A				IBM COPYRIGHT 1995 IBM CORPORATION			
				CONTENTS AND DESCRIPTION SIZE REVISION B DRAWING LAST MODIFIED: May 1 15:52:40 1996 EASTMAN J. TITTLE-ODYSSEY SHEET 2 of 26			

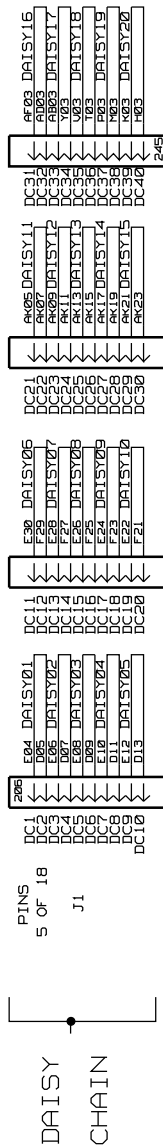
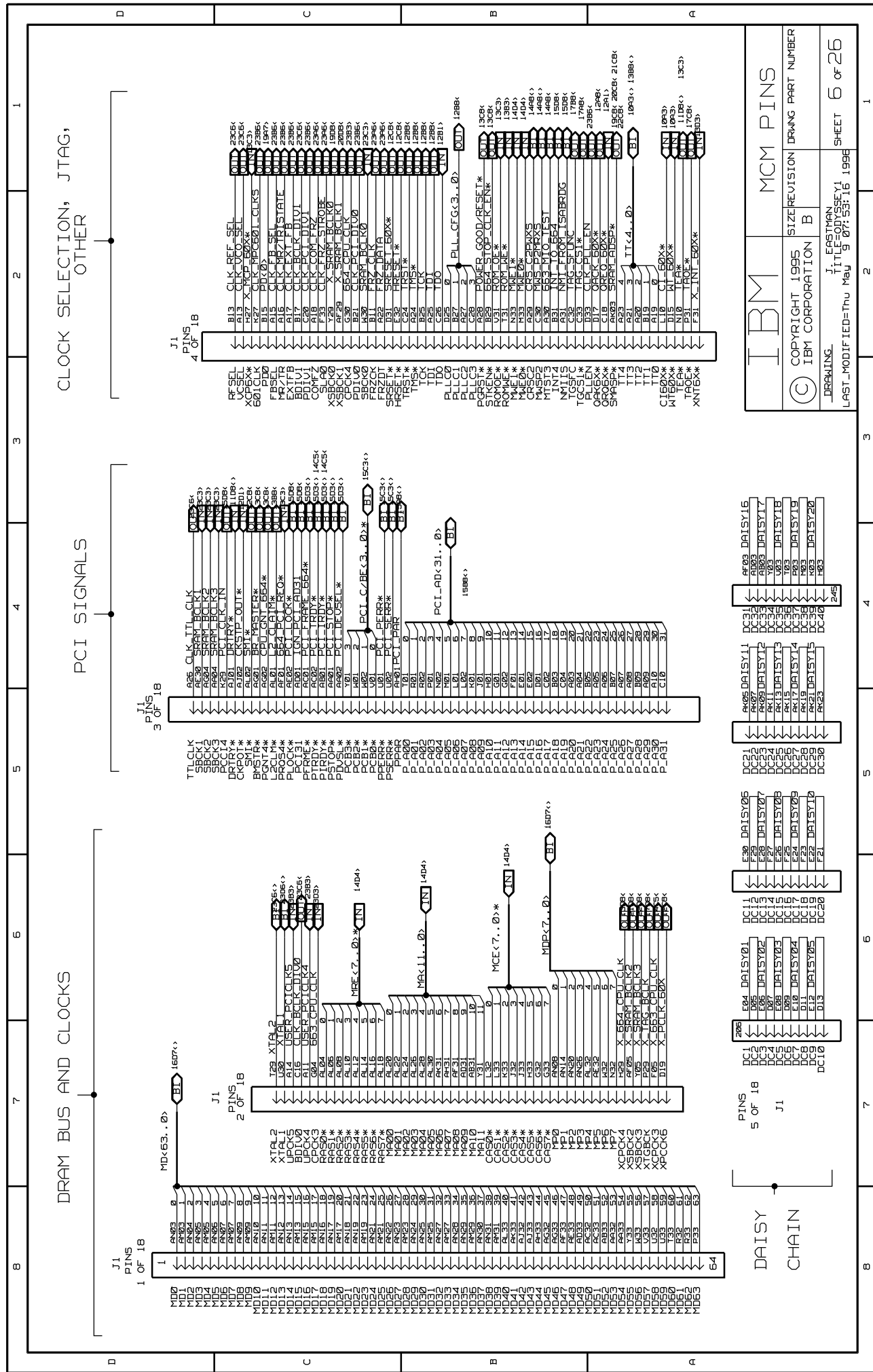




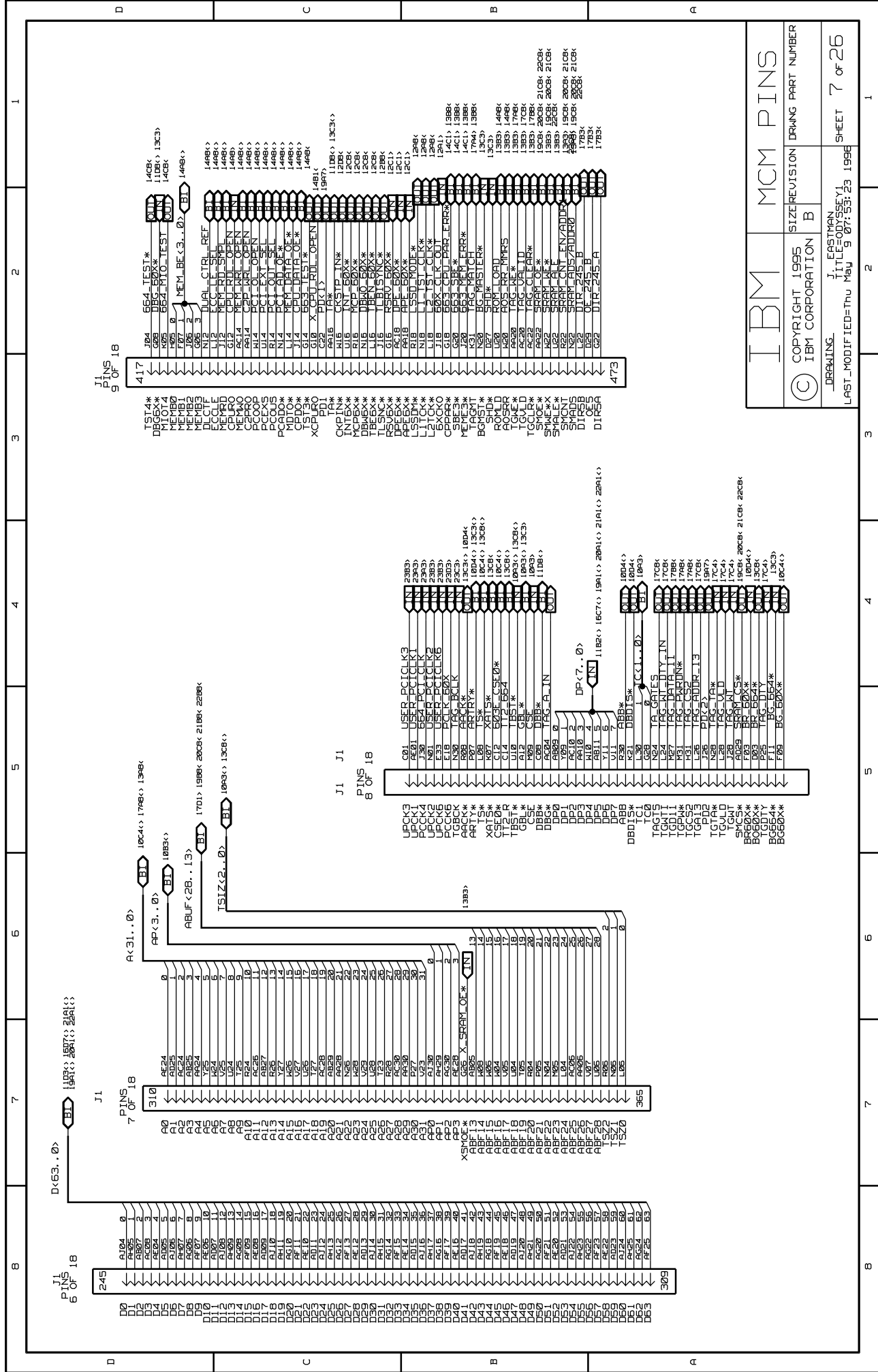
IBM		BLOCK DIAGRAM	
©	COPYRIGHT 1995	SIZE/REVISION	DRAWING PART NUMBER
	IBM CORPORATION	B	
DRAWING	EASTMAN		
LAST MODIFIED	15:53:01	1996	SHEET 3 of 26



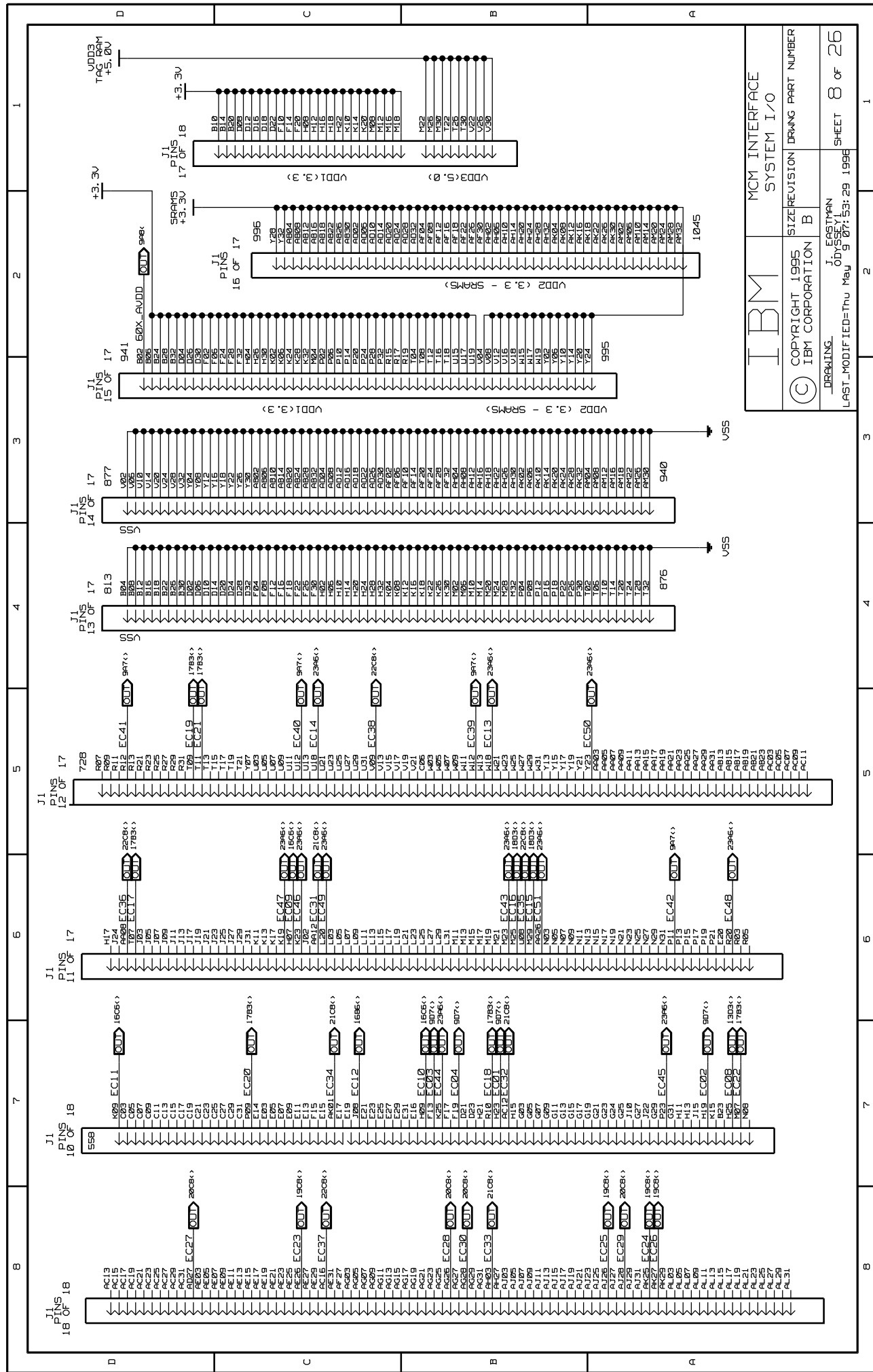
--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

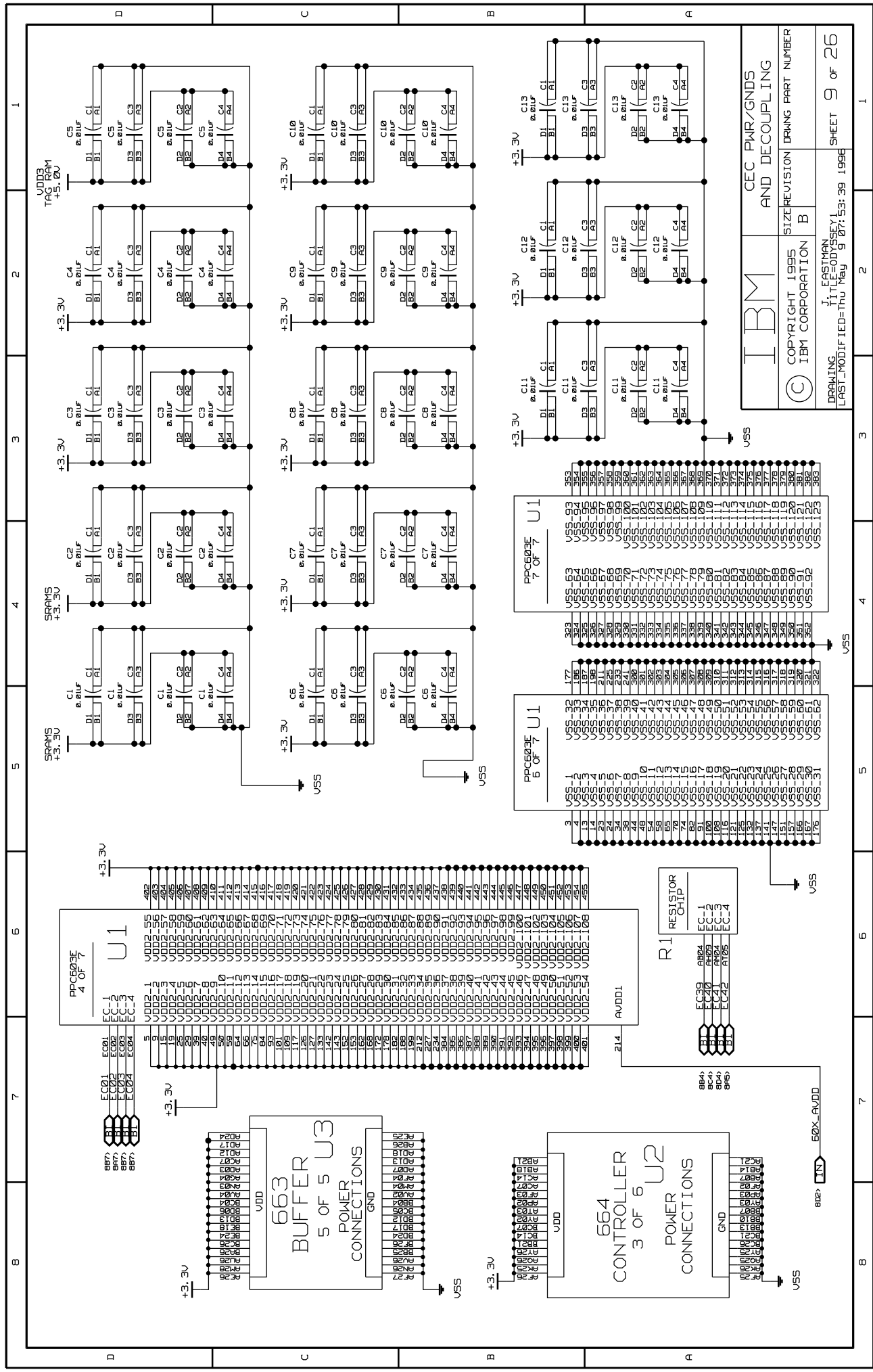


IBM		MCM PINS	
© COPYRIGHT 1995 IBM CORPORATION	SIZE B	REVISION	DRAWING PART NUMBER
DRAWING J. EASTMAN TITLE=ODYSSEY I LAST_MODIFIED=Thu May 9 07:53:16 1996		SHEET 6 of 26	

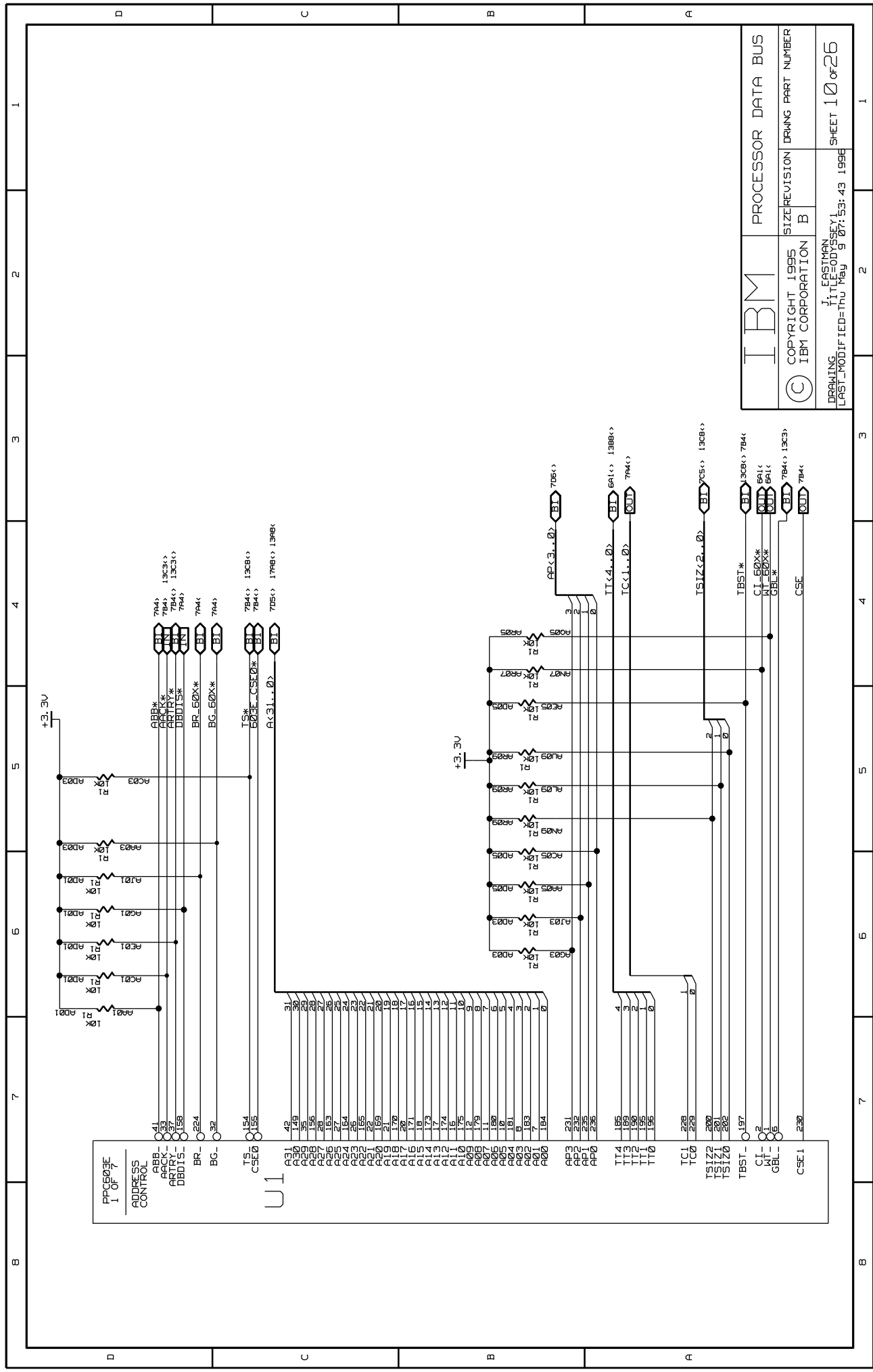


IBM MCM PINS	
© COPYRIGHT 1995 IBM CORPORATION	SIZE/REVISION B
DRAWING J. EASTMAN TITLE=ODYSSEY1 LAST_MODIFIED=Thu May 5 07:53:23 1998	
SHEET 7 of 25	



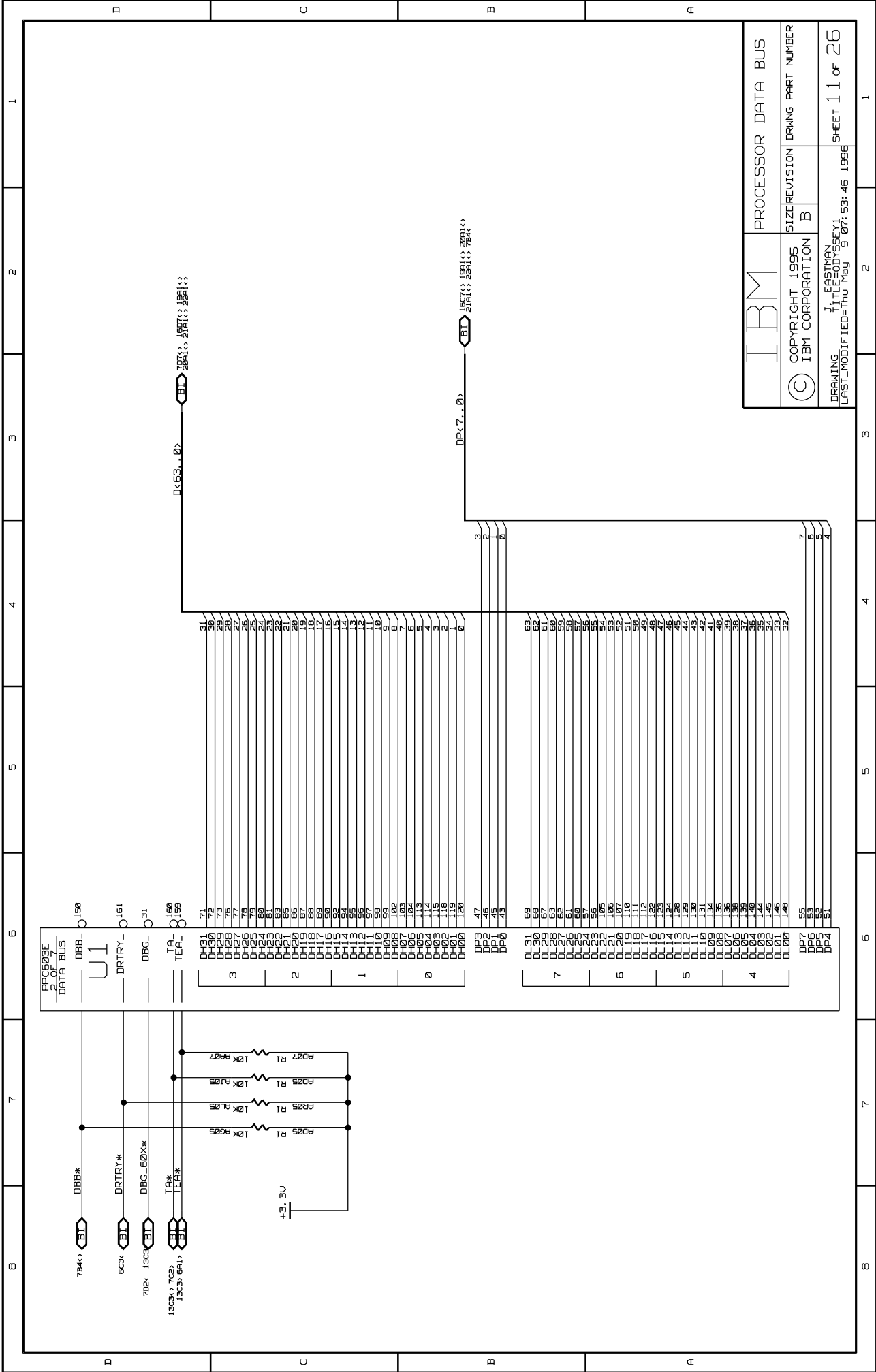


IBM		CEC PWR/GNDS AND DECOUPLING	
COPYRIGHT 1995 IBM CORPORATION		SIZE REVISION B	
DRAWING LAST_MODIFIED IED=Thu May 5 07:53:39 1995		SHEET 9 of 26	

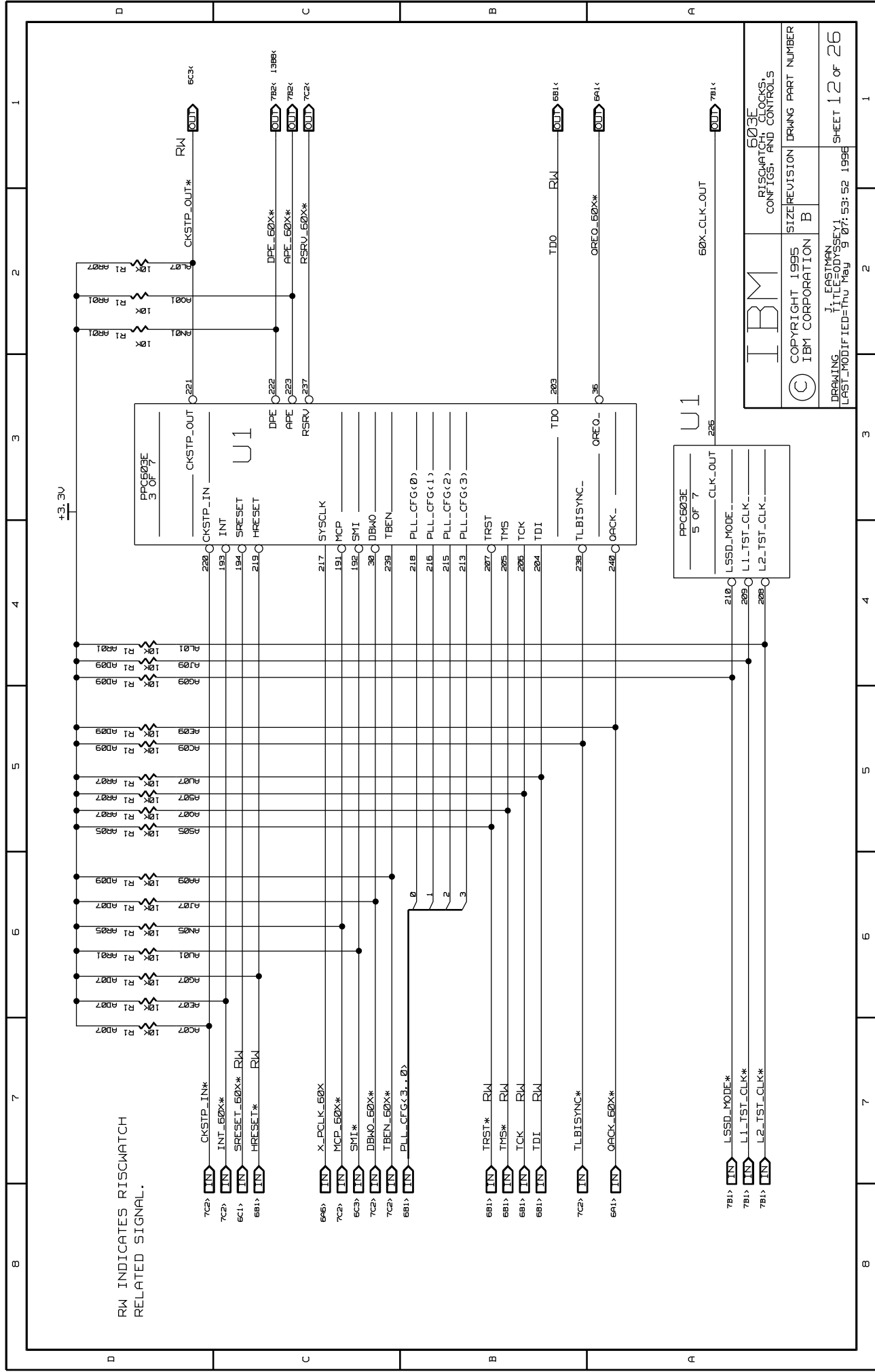


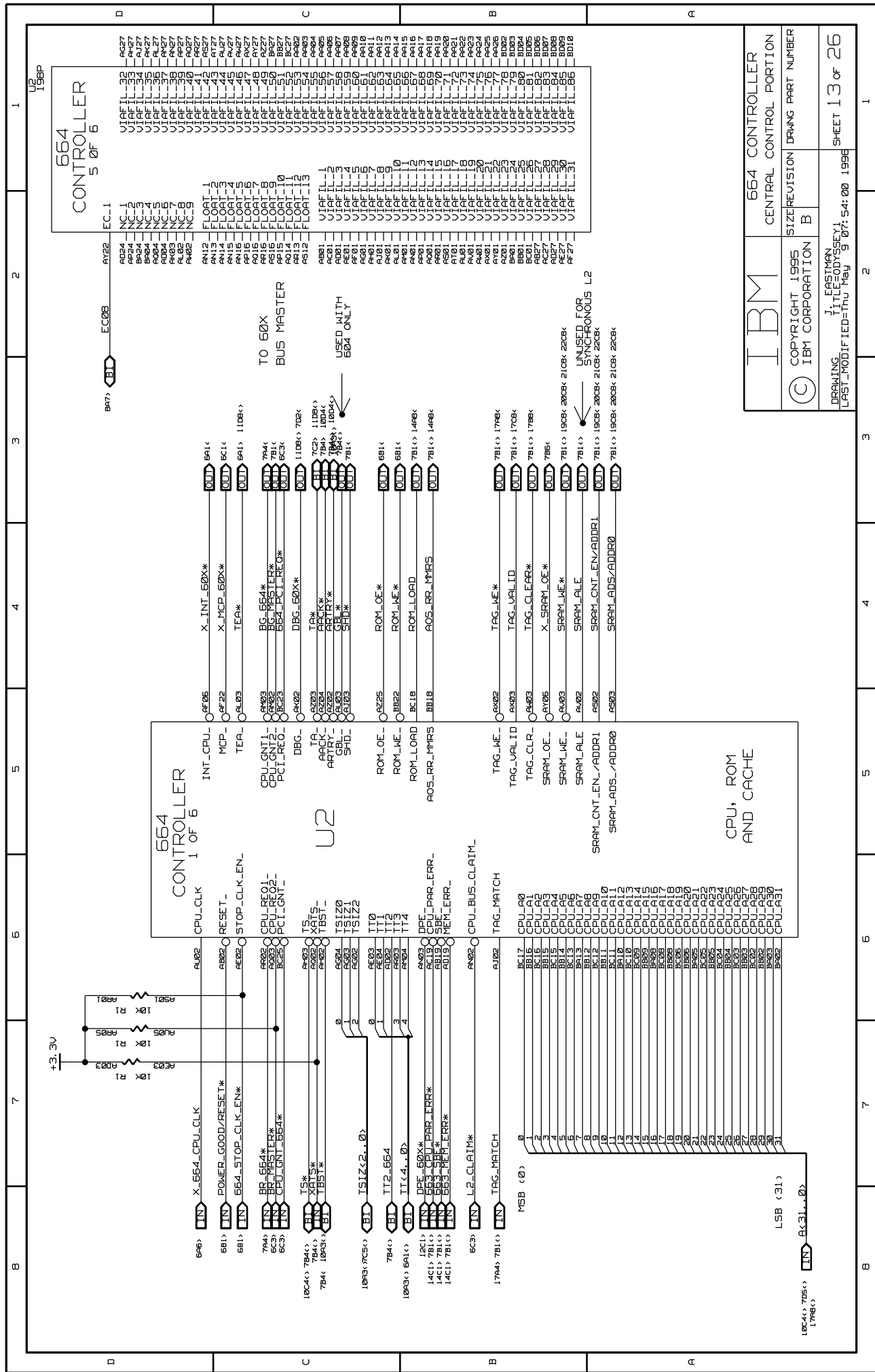
IBM		PROCESSOR DATA BUS	
COPYRIGHT 1995		SIZE/REVISION	
IBM CORPORATION		B	
DRAWING		SHEET 10 of 26	
LAST MODIFIED		5/11/95	
DRAWING		5/11/95	
LAST MODIFIED		5/11/95	

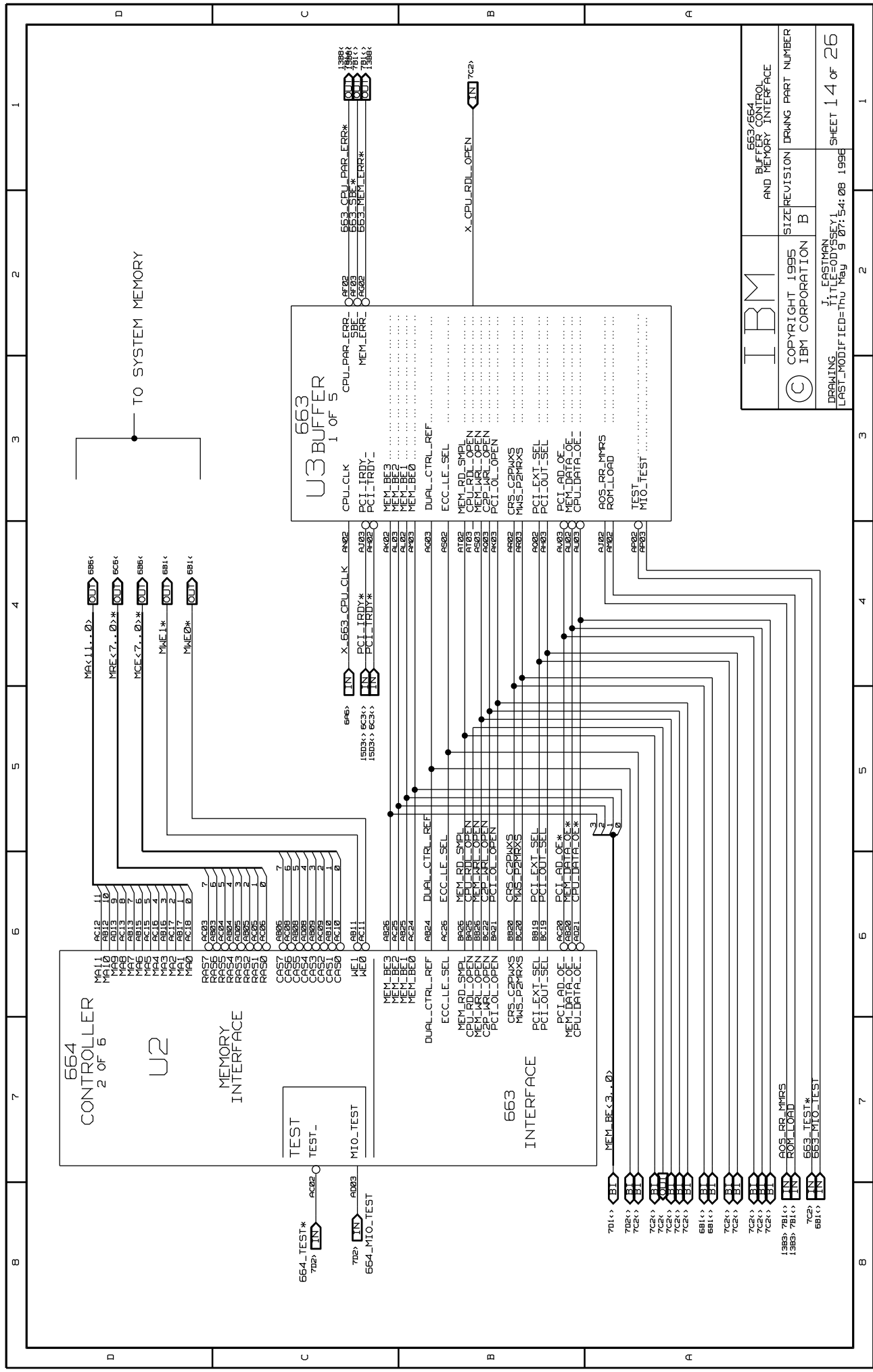




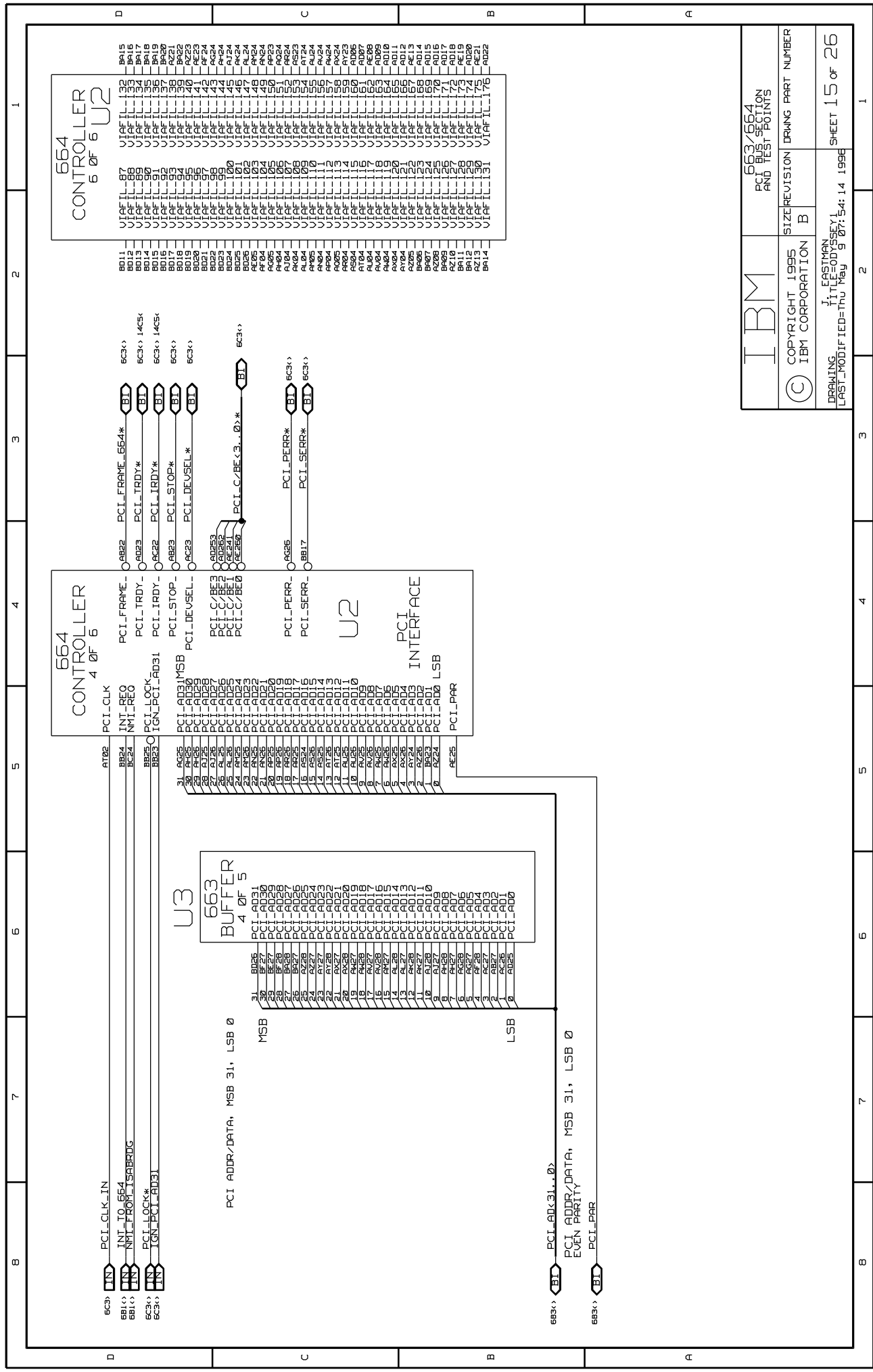
IBM		PROCESSOR DATA BUS	
©	COPYRIGHT 1995 IBM CORPORATION	SIZE	REVISION
		DRAWING PART NUMBER	
DRAWING			
SHEET 11 of 26			
DRAWING J: EASTMAN J: TITLE=ODYSSEY1			
LAST_MODIFIED=THU May 5 07:53:46 1995			

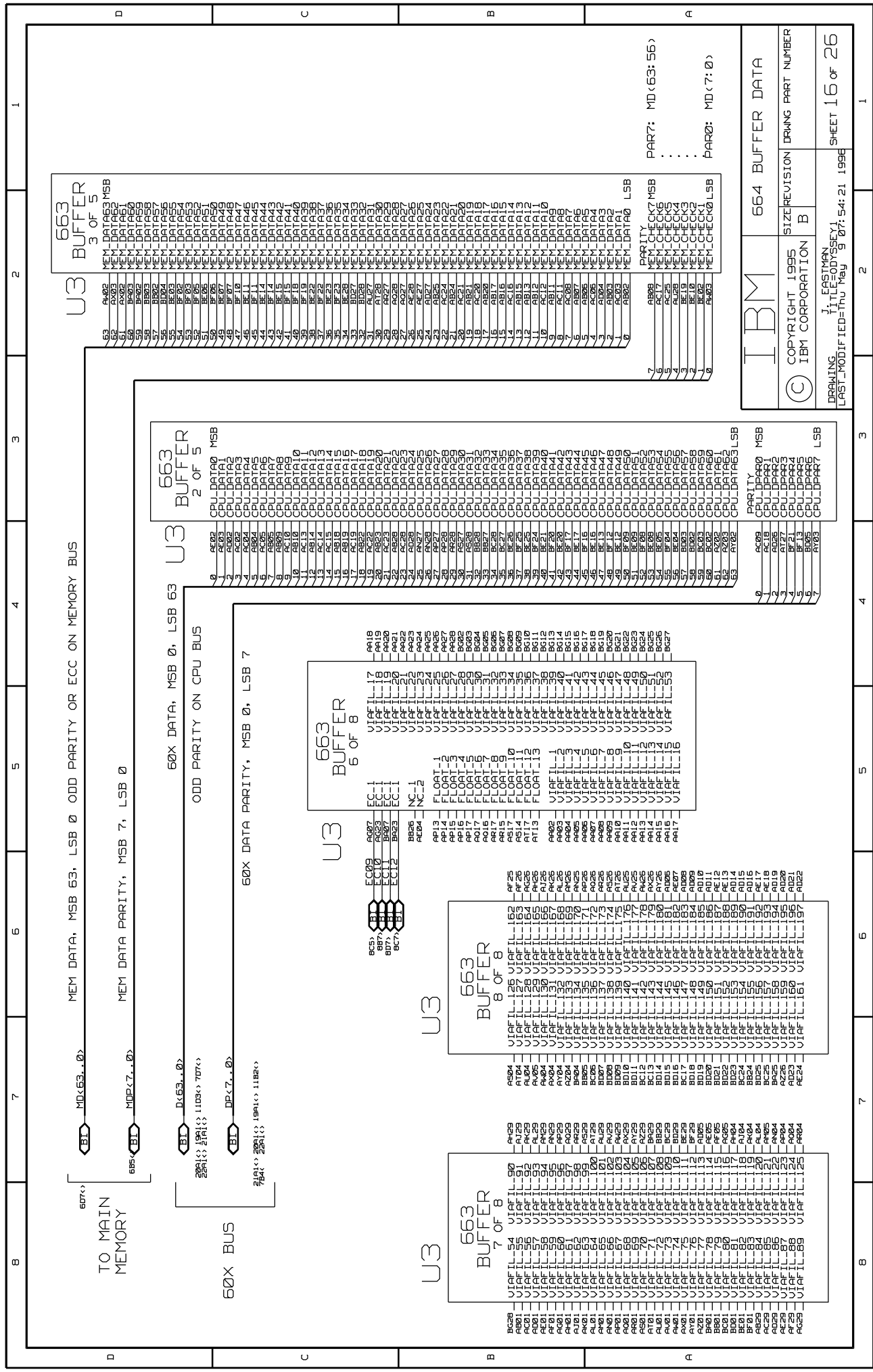






IBM		663/664 BUFFER CONTROL AND MEMORY INTERFACE	
COPYRIGHT 1995 IBM CORPORATION		SIZE/REVISION B	DRAWING PART NUMBER
DRAWING LAST_MODIFIED IED=Thu May 5 07:54:08 1996		SIZE/REVISION B	DRAWING PART NUMBER
DRAWING LAST_MODIFIED IED=Thu May 5 07:54:08 1996		SIZE/REVISION B	DRAWING PART NUMBER





663  
BUFFER  
3 OF 5

663  
BUFFER  
2 OF 5

663  
BUFFER  
5 OF 8

663  
BUFFER  
8 OF 8

663  
BUFFER  
7 OF 8

IBM

664 BUFFER DATA

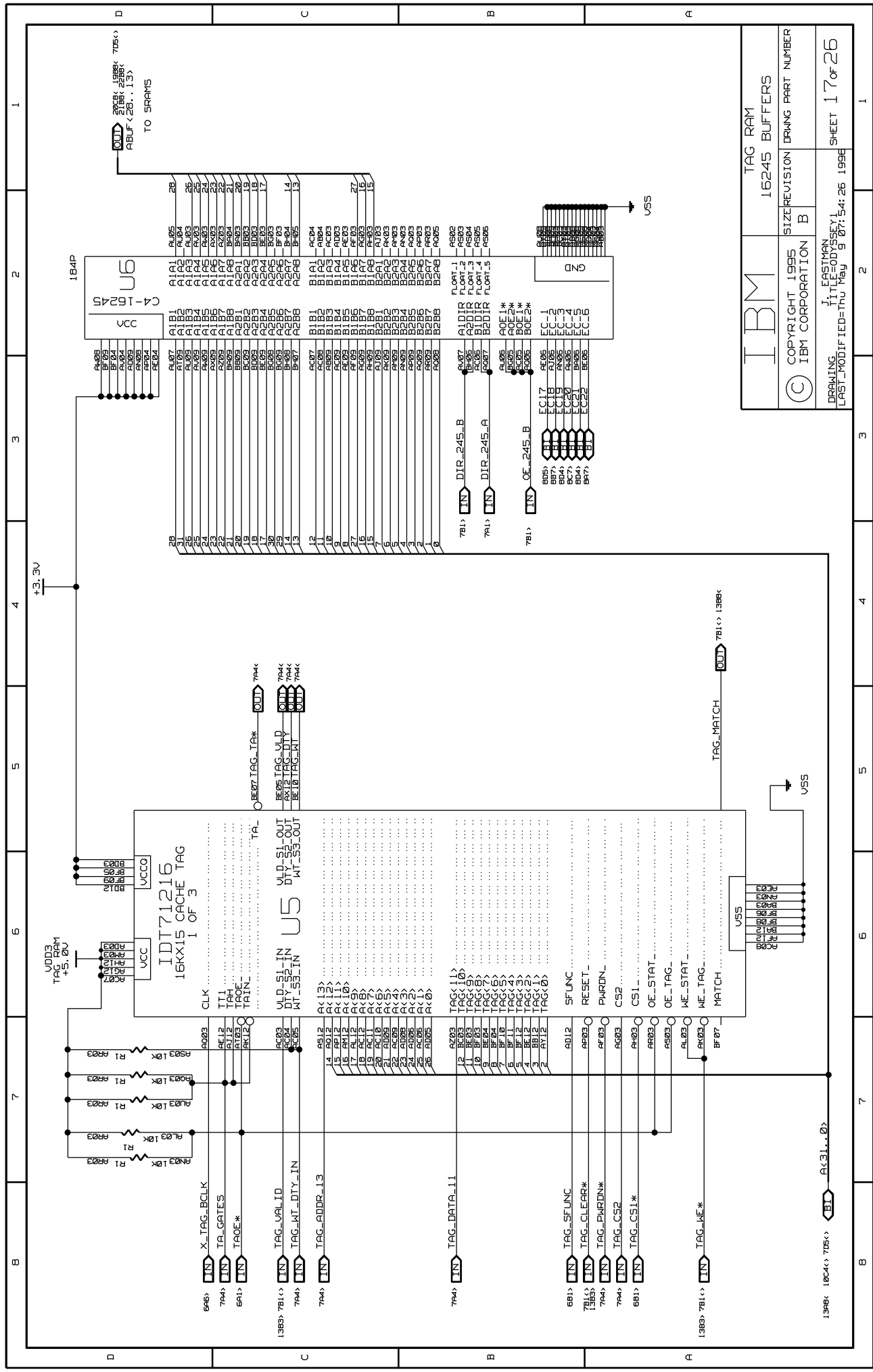
© COPYRIGHT 1995  
IBM CORPORATION

SIZE/REVISION  
B

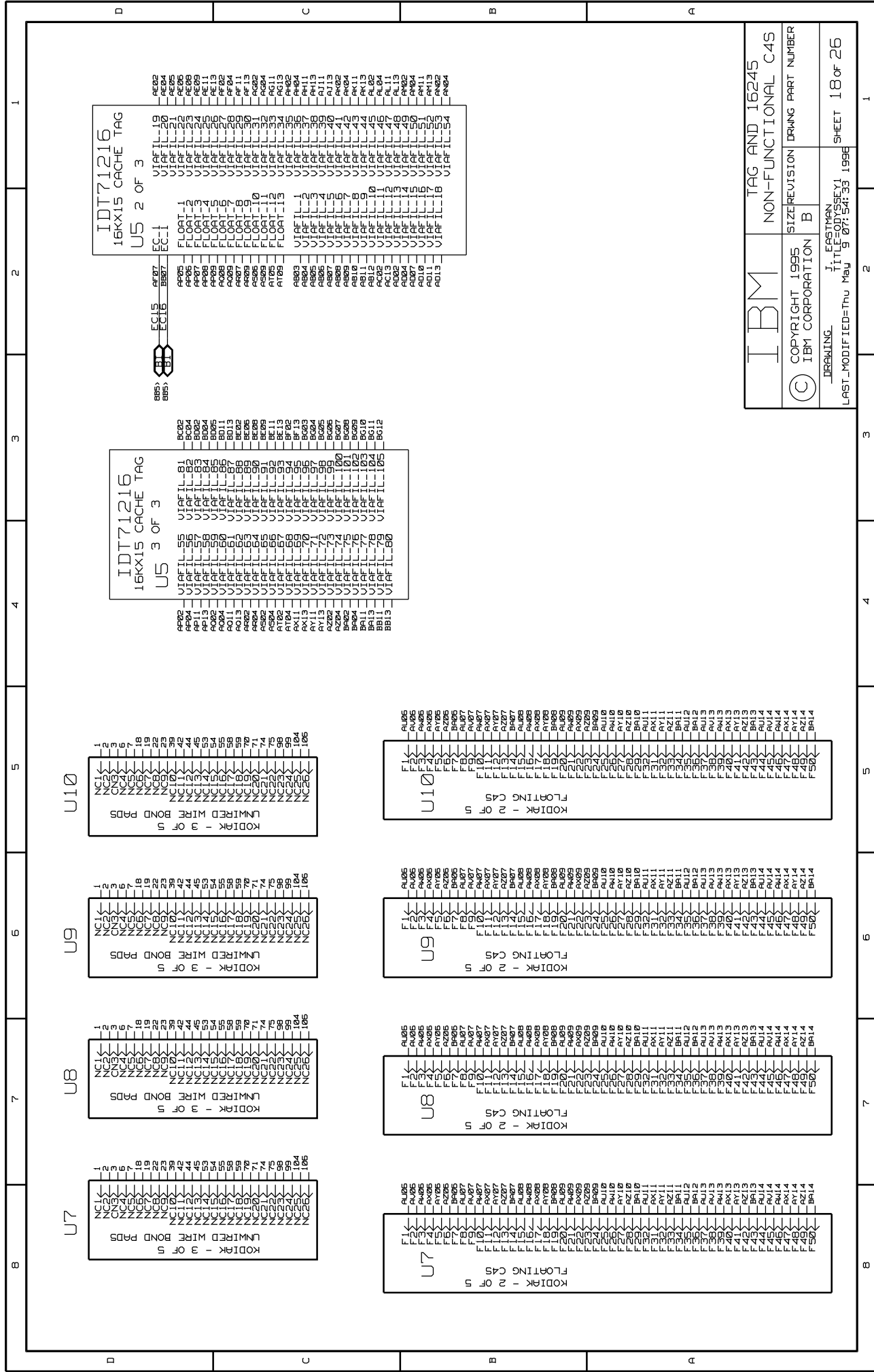
DRAWING PART NUMBER

DRAWING  
LAST MODIFIED: 07-54:21 1996

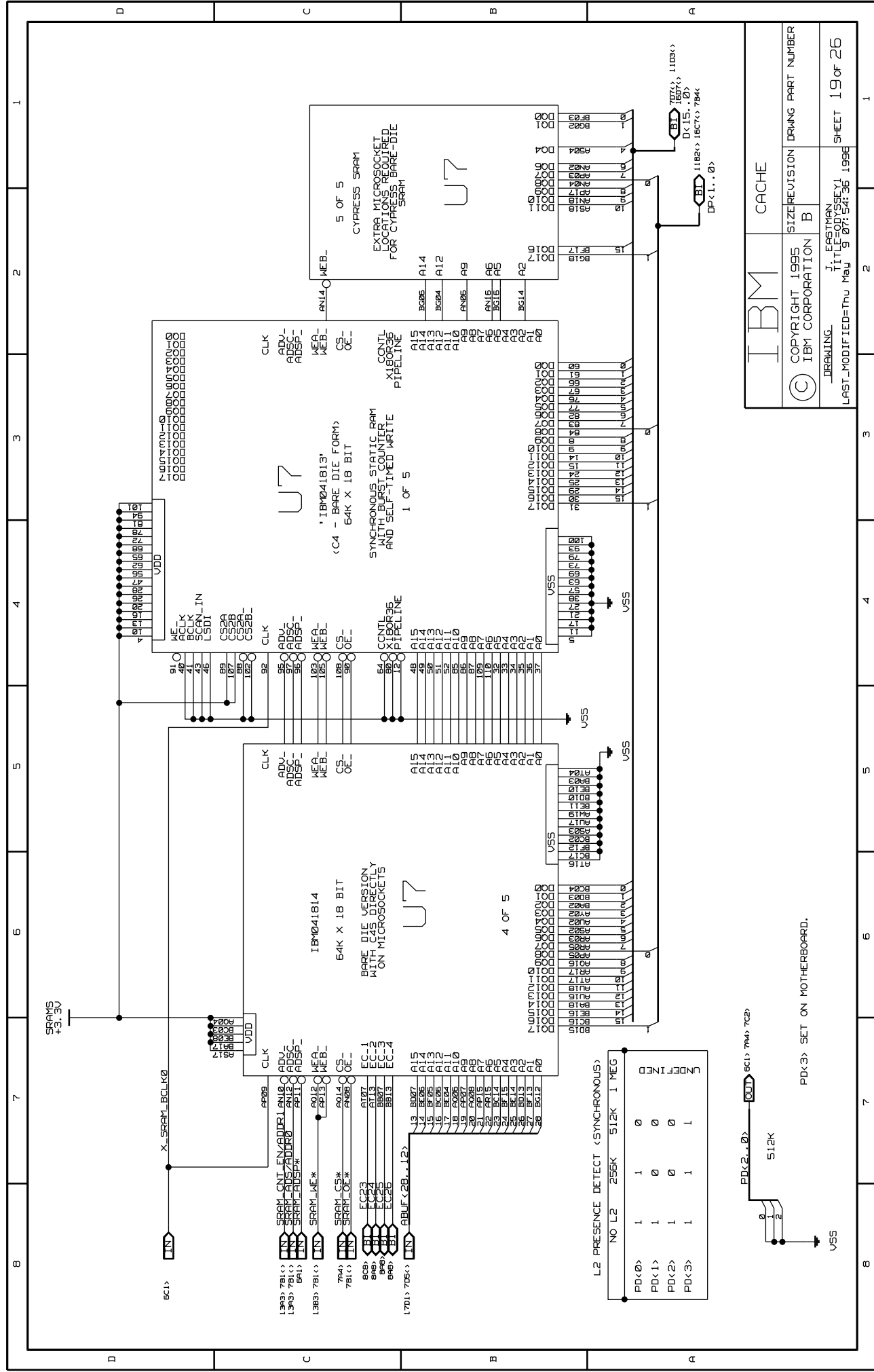
SHEET 16 OF 26

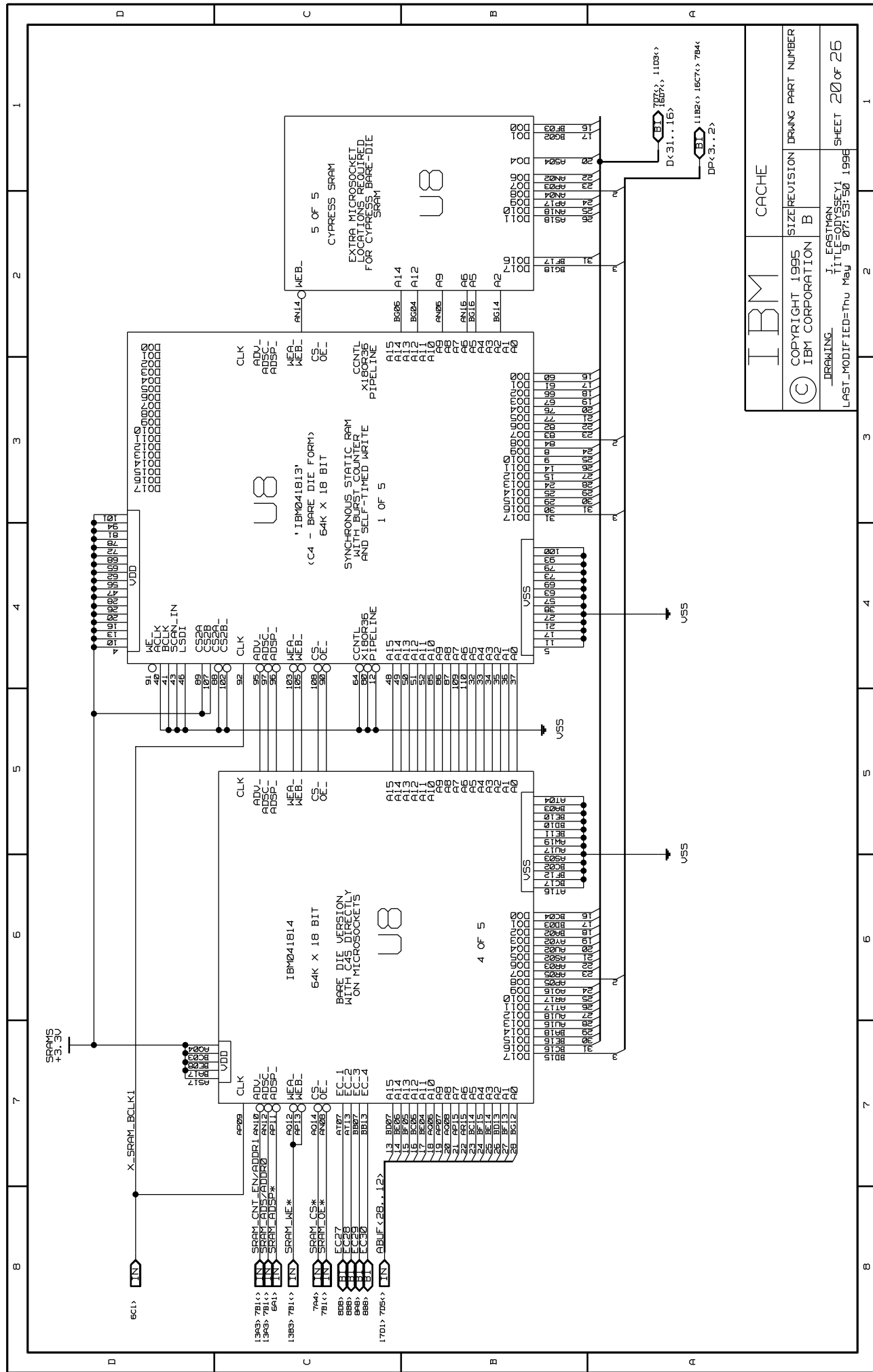


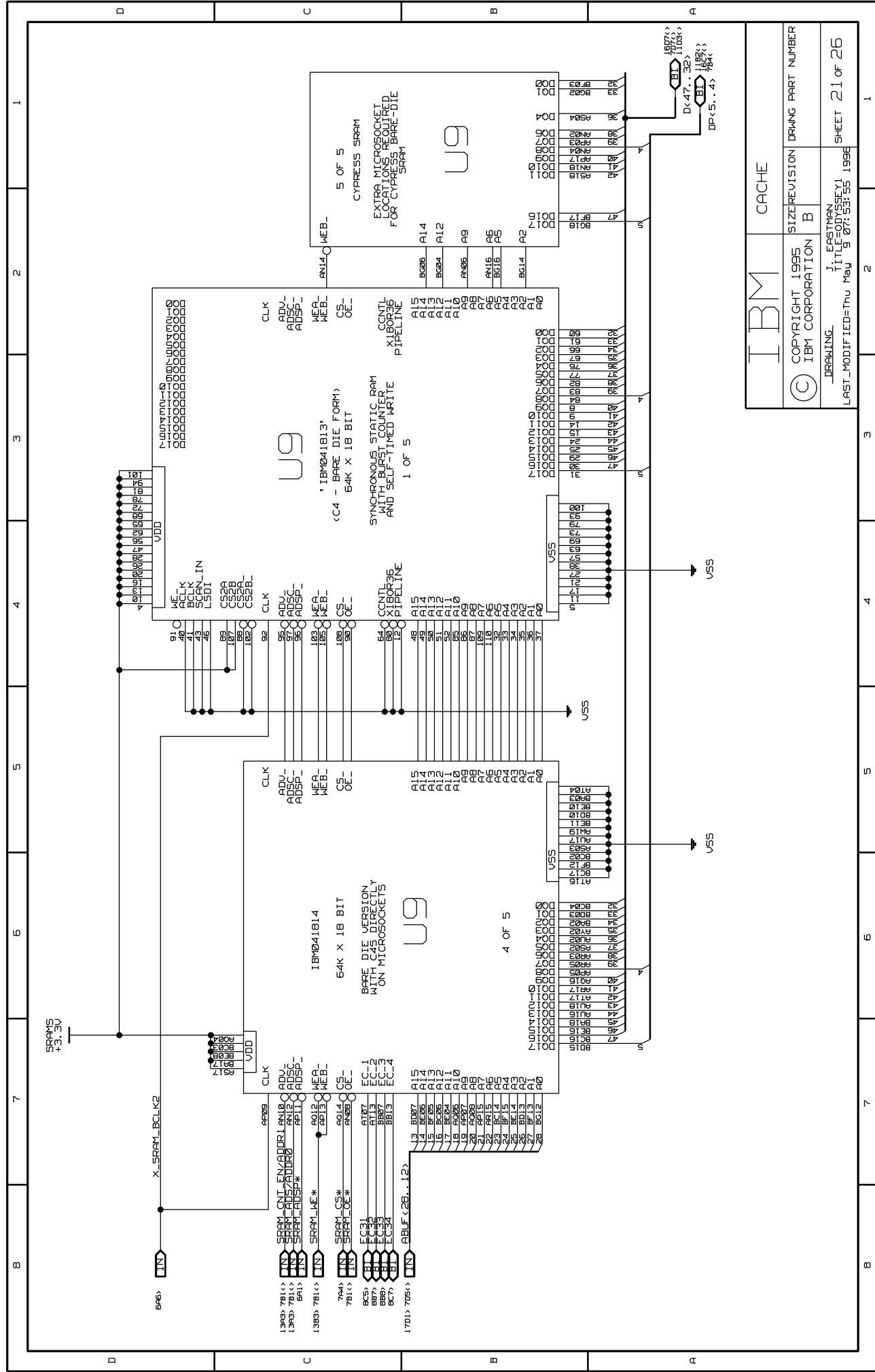
IBM		TAG RAM	
COPYRIGHT 1995		16245 BUFFERS	
IBM CORPORATION		SIZE/REVISION B	
DRAWING		SHEET 17 of 25	
LAST MOD IF IED=THU MAY 5 07:54:26 1996		TITLE=001356V1	

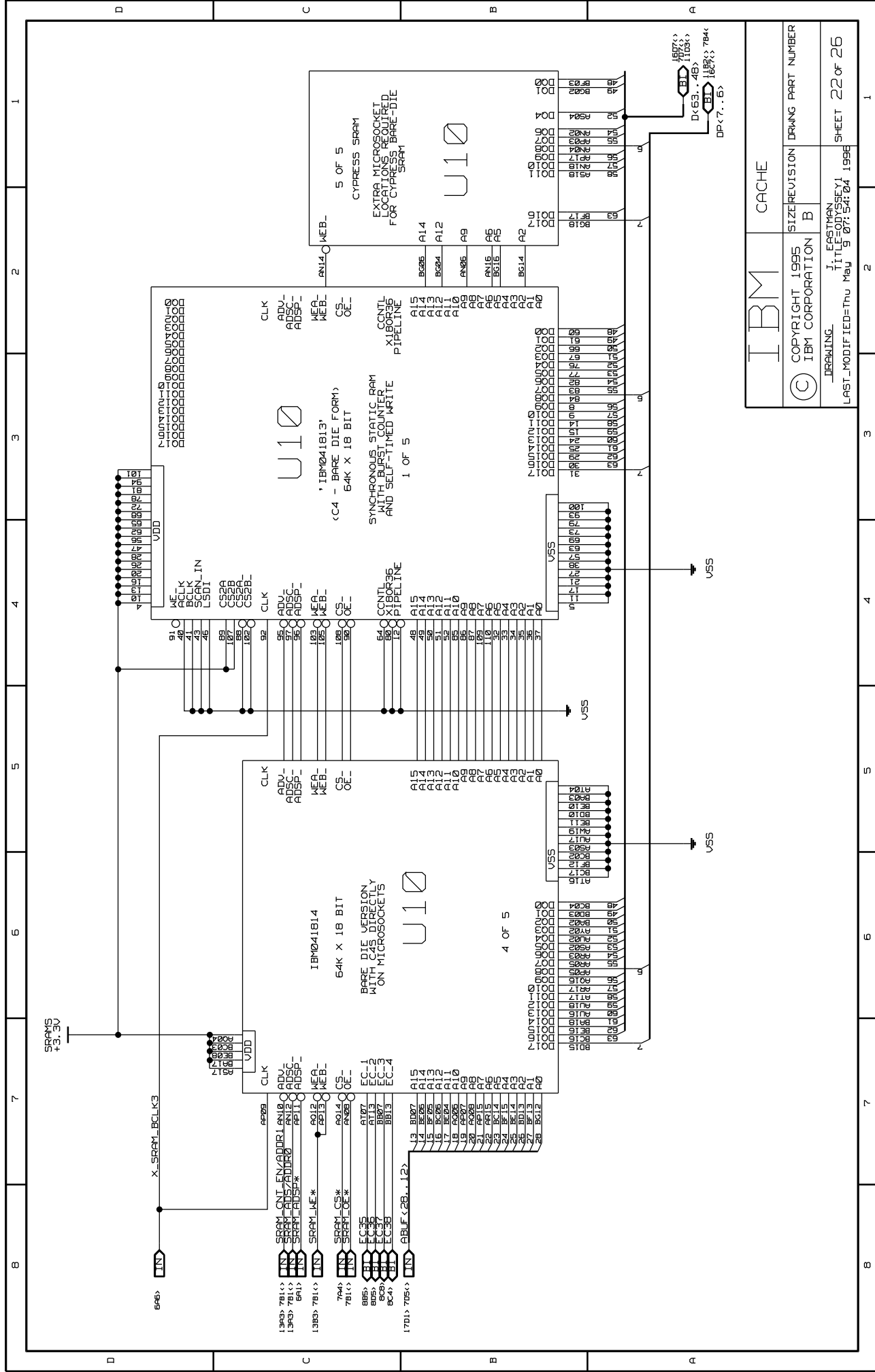










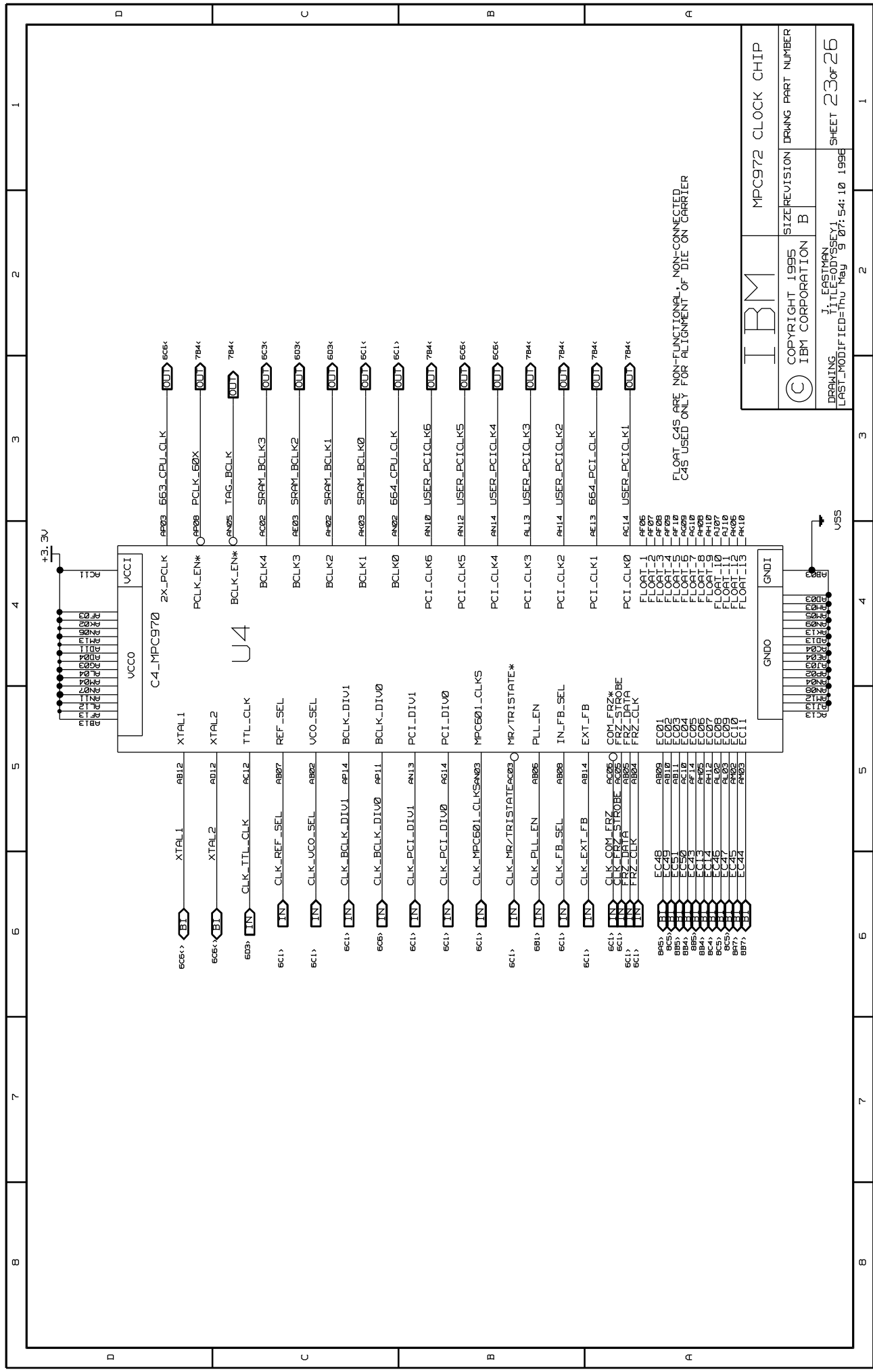


IBM

CACHE

©	COPYRIGHT 1995 IBM CORPORATION	SIZE REVISION B	DRAWING PART NUMBER
DRAWING	J. EASTMAN TITLE=ODYSSEY1 LAST_MODIFIED=Thu May 5 07:54:04 1998		

SHEET 22 of 25



IBM		MPC972 CLOCK CHIP		
©	COPYRIGHT 1995 IBM CORPORATION	SIZE/REVISION	DRAWING PART NUMBER	
		B		
DRAWING J11L5-00DYS55K1		SHEET 23 of 26		
LAST_MODIFIED=THU May 5 07:54:10 1995				

[illegible]

*** Unit Cross-Reference ***									
--- for the entire design ---									
C1	CAP-C4	905	905	905	905				
C2	CAP-C4	906	906	906	906				
C3	CAP-C4	903	903	903	903				
C4	CAP-C4	902	902	902	902				
C5	CAP-C4	901	901	901	901				
C6	CAP-C4	905	905	905	905				
C7	CAP-C4	904	904	904	904				
C8	CAP-C4	903	903	903	903				
C9	CAP-C4	902	902	902	902				
C10	CAP-C4	901	901	901	901				
C11	CAP-C4	903	903	903	903				
C12	CAP-C4	902	902	902	902				
C13	CAP-C4	901	901	901	901				
J1	PT1089	708	802	801	803	803	804	805	807
		808							
R1	RESCH1P	906	1004	1004	1005	1005	1005	1005	
		1006	1006	1006	1006	1006	1006	1006	
		1006	1006	1006	1006	1107	1107	1107	
		1107	1202	1202	1202	1204	1204	1204	
		1205	1205	1205	1205	1205	1205	1205	
		1206	1206	1206	1206	1206	1206	1206	
		1307	1307	1707	1707	1707	1707	1707	
U1	C4-PC003	904	905	905	1003	1106	1203	1203	
U2	C4-PC003	904	905	905	1003	1106	1203	1203	
U3	C4-LAN01	907	1403	1506	1507	1506	1506	1506	
U4	C4-MPC070	2304							
U5	C4-TAG	1705	1802	1804					
U6	C4-LV162M5	1702							
U7	C4-K001AK	1808	1808	1902	1906	1906	1906	1906	
U8	C4-K001AK	1807	1807	2002	2006	2006	2006	2006	
U9	C4-K001AK	1806	1806	2101	2106	2106	2106	2106	
U10	C4-K001AK	1805	1805	2202	2206	2206	2206	2206	

IBM	
©	COPYRIGHT 1995 IBM CORPORATION
SIZE	REVISION
B	
DRAWING PART NUMBER	
SHEET 25 OF 26	

8	7	6	5	4	3	2	1
THIS PAGE INTENTIONALLY LEFT BLANK							
D C B A							

IBM		SIZE	REVISION	DRAWING PART NUMBER
DRAWING		7	COPYRIGHT 1995	
©		IBM CORPORATION	B	
LAST_MODIFIED=Tue Apr 30 22:47:11 1996		SHEET 26 of 26		
ODYSSEY1				



## **Appendix A**

### **Example Planar Overview**

#### **A.1 The Collateral Material**

The information in these appendixes are for reference only, and consists of example planar design information, example firmware, and additional information. It is believed to be correct, but is presented as is, and may contain errors. It is intended to help companies develop their own products using the MCM.

The example planar, along with the firmware and additional information, can be used:

- As a development system.
- To test new boot code, operating systems and/or applications.
- For performance measuring.

The example planar and firmware are:

- Tested for functionality to the level of software internal to IBM as of 12/31/96.
- Prototypes, which may contain prototype ASICs, errata, and/or wiring changes. The example planar may also have errata.

The example planar and firmware are not:

- A fully debugged, complete, and market ready design
- Tested for compliance to FCC and other regulatory requirements
- Intended to be used as a product.
- Sold by IBM.

## A.2 Example Planar

In order to offer more robust support to the MCM user, IBM has designed an example planar for the MCM (see Figure A-1). The example planar is described (schematics, BOM, mechanical drawings, and functional information) in these appendixes. The example planar is a working design that is in use internally at IBM, and that was used in the design and development of the MCM.

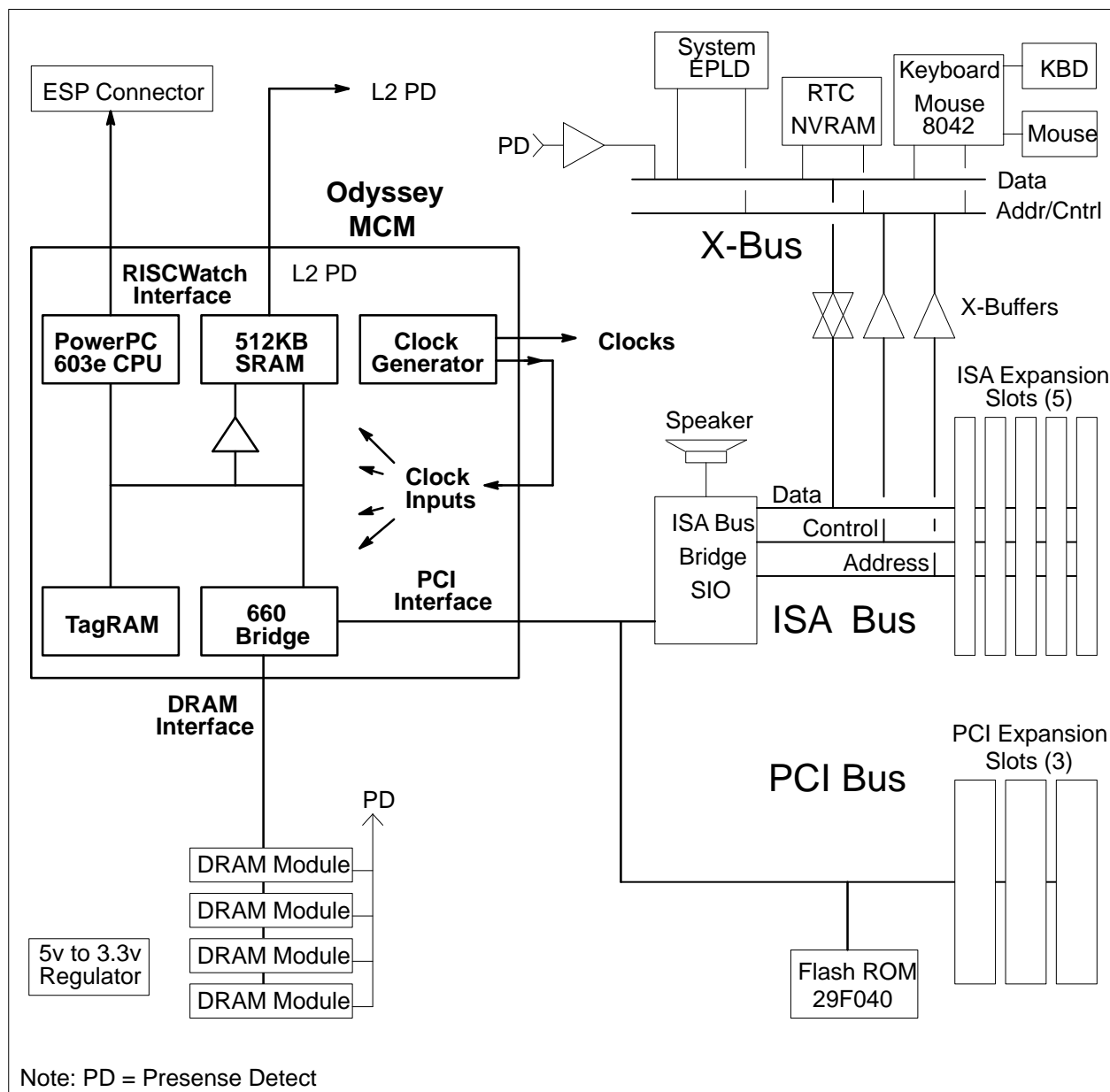


Figure A-1. 100 MHz PPC 603e Example Planar Block Diagram

The example planar is designed to an industry standard BabyAT (8.6 in. by 13 in.) form factor, although the enclosure may need to be modified to provide additional cooling and/or additional space for I/O cards. It requires +5V to power most of the components. The motherboard also requires  $\pm 12$  V to support some of the peripheral features. Components

that require +3.3V (such as the PCI bus agents) are supported using a regulator mounted on the motherboard to convert +5V to +3.3V. It is sized to fit within a BabyAT form factor enclosure.

### **A.2.1 System Memory**

The example planar memory subsystem can support up to 128M of DRAM memory on four 72 pin modules via sockets. Each SIMM socket can support an 8M or 32M 72 pin SIMM. The DRAM subsystem is 72 bits wide: 64 data bits and eight parity bits. One parity bit is generated for each byte of data written. The MCM can also be configured to perform ECC memory data checking and correction using the same standard parity DRAM modules. Or it can be configured to disable DRAM parity checking for systems using non-parity DRAM.

### **A.2.2 PCI Bus**

The MCM includes the interface between the PCI bus and the rest of the system. The example planar provides three standard PCI expansion slots.

### **A.2.3 Flash ROM**

The example planar uses an AMD AM29F040-120 Flash™ ROM to contain the POST and boot code. Vital Product Data (VPD) such as the motherboard speed and native I/O complement be programmed into in this device. It is possible to program the Flash before or during the manufacturing process.

### **A.2.4 ISA Bus**

The ISA bridge function is provided by an Intel 82378ZB chip (SIO). It provides a PCI to ISA bus bridge where the native I/O and the ISA slots reside, and it provides system services such as ISA bus DMA, PCI bus arbitration, and interrupt control. The example planar provides five PCI expansion slots.

### **A.2.5 System I/O EPLD**

The system I/O EPLD is a programmable logic device that uses the X-bus signals and the partial decode signals from the SIO to decode chip selects for various components.

### **A.2.6 X-Bus Agents**

The example planar uses a Dallas Semiconductor™ DS1385S to provide the real time clock (TOD or RTC) function. This device is PC compatible and resides on the X-bus. It features an additional 4K of NVRAM and a replaceable battery.

The example planar uses an Intel 8042AH as a keyboard and mouse controller.

### **A.2.7 Power Management**

Some components are included on the example planar which can be used for power management hardware. Power management is not implemented on the example planar, and is not intended to be a feature of the example planar.

### A.3 Other Help

Several other system, firmware, and miscellaneous issues of interest to the designer are included here. For more information, refer to the IBM PowerPC World Wide Web pages at <http://www.chips.ibm.com> and [ftp://ftp.austin.ibm.com/pub/PPC\\_support](ftp://ftp.austin.ibm.com/pub/PPC_support). Also contact your IBM PowerPC Embedded Processor Solutions Technical Representative for details.

Consult specific manufacturer technical specifications for information on specific I/O devices such as hard drives, CD-ROMs, L2 cache cards, video cards, etc. that comprise a total system.

#### A.3.1 Example Firmware

IBM has developed firmware to support the MCM and example planar. The source code and the binary of this boot & POST firmware is available from IBM. Recommendations in this document for memory mappings, and various software issues may vary from the algorithms implemented in boot code or operating systems.

#### A.3.2 Design Files

The Gerber format physical design files of the example planar are available from IBM.

#### A.3.3 Quickstart Peripheral List

The example planar is intended to use typical PC peripherals. Products from a large number of manufacturers should work satisfactorily.

Table A-1 outlines generic requirements for peripherals and gives examples of some devices that have been used for testing. It is not a recommendation of any particular vendor. Table A-1 does not include cables for a parallel port, indicators, a switch, or a speaker.

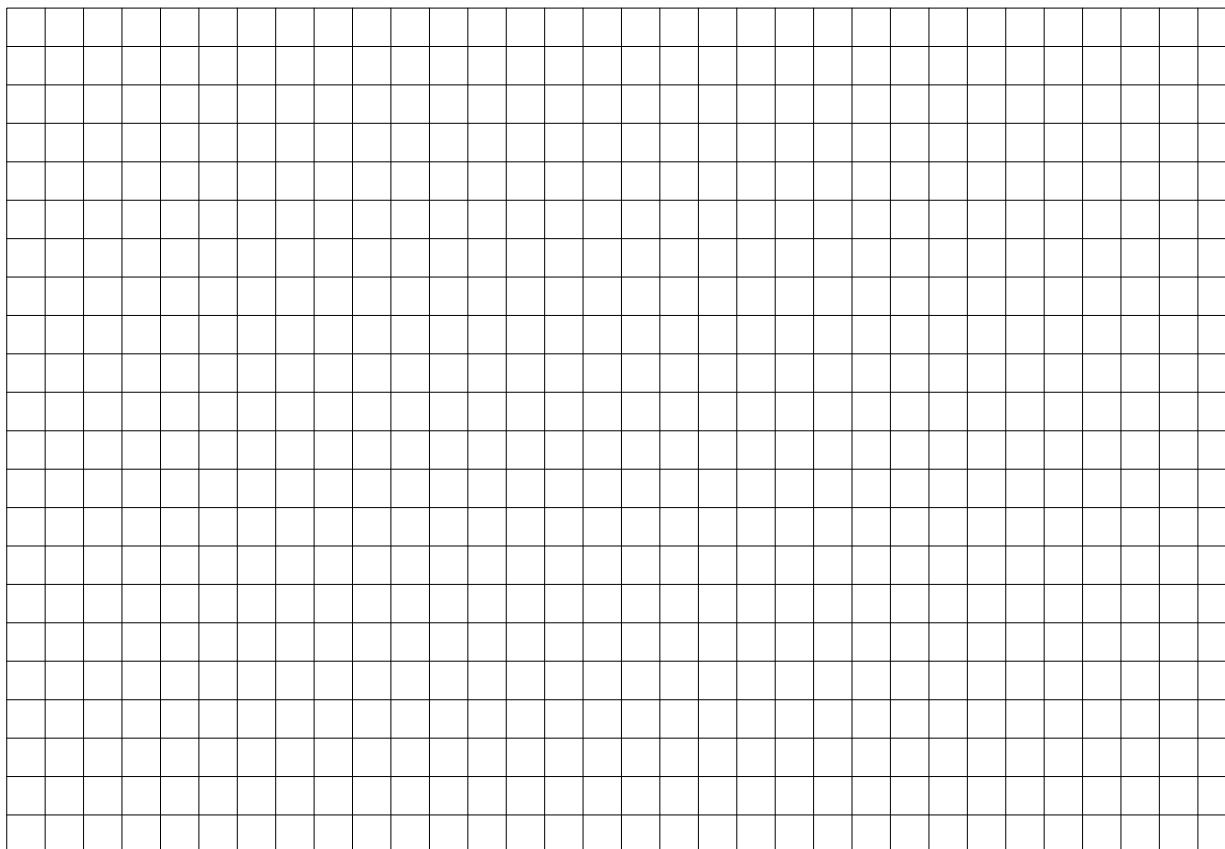
An IBM 3101 asynchronous terminal or equivalent is required for testing with the Bring Up Driver (BUD) code. Settings are 8-bit, no parity, one stop bit, and 9600 baud. VT100 or VT52 emulator terminals may work. It is desirable to also have a video monitor for BUD tests. The boot code will boot with either an async console, a video on motherboard, or both.

**Table A-1. Quickstart Peripheral List**

Generic Description	Example Device
L2 SRAM card, 256KB	Alliance Semiconductor AS7M64P3256-15C
Loctite 384 adhesive, 300ml cartridge	Loctite Corp. 17041
Loctite 384 activator, 10 liter can	Loctite Corp. 17101
Video adapter card, PCI S3	Diamond Stealth (S3) 864
SCSI adaptor card, PCI	NCR 8100S with 609-039-1635 controller
Super I/O adaptor ( IDE, floppy, serial ports, parallel ports, etc.)	Acculogic sIDE-4/HP (110-00139-00E00)
Audio adaptor card	Creative Labs Soundblaster 16
Floppy disk drive, 3.5"x1.44MB	Alps DFR723F, IBM 73G4514, Mitsubishi MF355F-258UG
Hard disk drive, SCSI-2, 8 bit	Quantum LPS270/5405, Maxtor MXT-540SL, IBM WDS-3200 (79F4042)
Hard disk drive, SCSI 1GB	IBM 94G3187
Hard disk drive, IDE 1GB	Maxtor HDMC71260AC
CD-ROM drive, internal SCSI	Toshiba XM-4101BMV
CD-ROM drive, internal SCSI, 4x	Toshiba 5301-4x
CD-ROM drive, internal IDE, 4x	Chinnon CDS5451

**Table A-1. Quickstart Peripheral List (Continued)**

<b>Generic Description</b>	<b>Example Device</b>
Chassis, Baby AT	Olsen Metal Products CC300249-17
Power supply, 200W Energy Star	API-3186S, IBM 06H2968
Box fan	Panaflo FBA08T12M
Box fan shock mounts	IBM 81F7977
Internal cables, floppy, SCSI, and CD-ROM	Standard cables
Speaker, internal 8Ω .5W 2pin	
LED, 2.5 ma drive	
Asynchronous terminal	IBM 3101
Super VGA monitor	IBM 6324, 6325, 6327, 9524, 9525, 9527, 9521
Keyboard, PS/2 compatible	
Mouse, PS/2 compatible	



Handwriting practice lines consisting of 15 horizontal lines.

## **Appendix B**

### **Planar I/O Subsystems**

#### **B.1 ISA Subsystem**

The 100 MHz PPC 603e MCM example planar includes an ISA bus that is interfaced to the PCI bus by the ISA bus bridge. ISA expansion slots are provided by the example planar. The example planar uses a buffered subset of the ISA bus, called the X-bus, to host on-board native I/O, such as the real time clock and the keyboard and mouse controller.

##### **B.1.1 The ISA Bridge**

The ISA bridge function is provided by an Intel 82378ZB chip (SIO). It provides a PCI to ISA bus bridge, with the following major functions:

- Bridge between PCI and ISA
  - 8/16 bit ISA devices
  - 24 bit addressing on ISA
  - Partially decodes native I/O addresses
  - Unclaimed PCI memory address below 16MB forwarded to the ISA bus
  - Unclaimed PCI I/O address below 64K forwarded to the ISA bus
  - Powers up to an open condition (i.e., cycles may be passed to the ISA bus)
  - Generates ISA clock, with a programmable divide ratio of three or four
  - Allows ISA mastering and has programmable decodes that map ISA memory cycles to the PCI bus
  - 32-bit posted memory write data buffer (no I/O buffering)
- Seven channel ISA DMA controller
  - Function of two 83C37s with 32-bit extensions
  - Supports 8-bit or 16-bit devices on the ISA bus
  - Supports 32-bit addressing for ISA to PCI memory transfers
  - 8-byte bidirectional buffer for DMA data
- Timer block (function of 82C54)
- Interrupt Controller (function of two 8259s)
- PCI bus arbiter.
- Functions as PCI target during programming and ISA target cycles, and as busmaster during DMA or ISA master cycles

- Generates ISA\_REFRESH# signal to refresh ISA bus DRAM.

### B.1.2 Address Ranges

The ISA bus address ranges which may be separately enabled in the ISA bus bridge for forwarding to the PCI bus are:

- 0 - 512K
- 512K - 640K
- 640K - 768K
- 768K - 896K ( in 8 ranges of 16 K each)
- 896K - 960K
- 1M - xM (where  $x \leq 16$ ) with the hole (see the SIO data book). The hole may be 64K or 8M.

If an ISA DMA produces an address in the 0-16M range and this address is enabled in the ISA bridge for forwarding to the PCI, the ISA bridge will initiate a PCI transaction which the 660 bridge will forward to system memory.

The 660 uses medium timing when claiming ISA master originated cycles on the PCI. It does not use subtractive decoding. Hence, ISA masters can only communicate with other ISA devices or system memory. They may not communicate with PCI devices.

Warning: The software should not map any PCI memory at PCI addresses which ISA masters can create (those addresses between 0-16M which are programmed for forwarding from ISA to PCI). This is because contention would result between the device mapped at that address and the 660. Alternatively stated, ISA masters should not be allowed to create accesses to system memory using any address between 0 and 16M that is mapped to a PCI device, such as video.

### B.1.3 ISA Bus Concurrency

ISA bus cycles which are not enabled for forwarding, including the hole, remain on the ISA bus. That is, DMA or ISA busmaster cycles on the ISA bus can run concurrently with PCI or CPU cycles.

### B.1.4 ISA Busmasters and IGN\_PCI\_AD31

The ISA bridge supports ISA busmasters. System memory accesses from an ISA busmaster are designed to be mapped to the 0 to 16M range, and the ISA bridge forwards them to the PCI bus at the same range, which is not compliant to the PowerPC Reference Platform specification. Other PCI to system memory accesses however, are correctly mapped to the 2G to 4G range (for system memory address range from 0 to 2G). In some architectures this problem is handled by using the ISA\_MASTER# signal, which is active during the ISA busmaster operation.

However, the ISA bridge allows ISA masters to run posted writes to system memory, without latching in the accompanying ISA\_MASTER# signal. In this situation, the ISA\_MASTER# signal is no longer synchronized to the ISA busmaster operation.



To overcome this challenge, the example system detects PCI memory transactions that are initiated by the SIO. The example planar ANDs together GNT0#, GNT1#, and GNT2#, to generate IGN\_PCI\_AD31, which is active high during the address phase of any PCI transaction that is not initiated by one of the three possible PCI agents (besides the 660 and the SIO). If the PCI transaction was not initiated by the 660, and if it is a memory transaction, then the 660 assumes that it is a system memory transaction initiated on the PCI bus by the SIO on behalf of an ISA busmaster, and so forwards it to the correct system memory address in the 0 to 16M range.

As a consequence of this design, the ISA bridge must be programmed to map ISA DMA (that is bound for system memory) to a PCI memory transaction using the 0 to 2G address range, rather than the apparently correct 2G to 4G range. Since the DMA sourced PCI transaction also causes IGN\_PCI\_AD31 to be asserted during the address phase of a PCI transaction initiated by the ISA bridge, the 660 will not do the usual inversion of the highest order address bit, but will forward the transaction to system memory in the 0 to 2G range.

Another consequence of the design is that the ISA bridge can not initiate peer to peer PCI memory transactions, because no matter what the PCI address is, it will be claimed by the 660 (if the address is that of a populated memory location) and mapped to system memory, possibly causing various inappropriate results.

These are the only limitations on the normal operation of the ISA bridge that are caused by the IGN\_PCI\_AD31 design, and there are no implications for other PCI or ISA agents, which are totally unaffected by the situation.

### B.1.5 DMA

The DMA controller in the ISA bridge consists of the functionality of two 82C37A DMA controllers with 32-bit addressability extensions and enhanced functionality. The DMA request/grant lines are connected on the example planar as shown in Table B-1.

Table B-1. DMA Assignments

DMA Channel	Assignment or Connection
0	ISA connectors
1	ISA connectors
2	ISA connectors
3	ISA connectors
4	Cascade in
5	ISA connectors
6	ISA connectors
7	ISA connectors

#### B.1.5.1 Supported DMA Paths

DMA operations can be performed only:

- From ISA I/O mapped devices to ISA memory mapped devices, and
- From ISA I/O mapped devices to system memory (via the PCI bus). In these transfers, the system memory address must be mapped to the PCI address range 0 to 2G (see Section B.1.4).

The DMA source device can be located on the X-bus. If the DMA target is ISA memory mapped, it can also reside on the X-bus.

- ISA DMA to PCI I/O devices is not allowed.
- ISA DMA to PCI memory devices is not allowed.
- ISA DMA from ISA memory mapped devices is not allowed.

#### **B.1.5.2 DMA Timing**

The DMA controller runs compatible cycles for all ISA to ISA DMA transfers. Type A, type B and type F timing is available only for ISA I/O to system memory (via the PCI) DMA transfers.

#### **B.1.5.3 Scatter-Gather**

The example planar permits the use of independent scatter-gather (SG) operations on DMA channels 0-3 and 5-7. This operation chains together a number of DMA transfers to different memory locations so that they appear as one DMA transfer. The SG command, descriptor table, and status registers are relocatable via a configuration register in the ISA bridge. The termination of an operation may be signaled to the software by configuring any (or all) of the SG channels to use IRQ13 or by configuring the channel to signal end of process (aka Terminal Count) to the DMA device. See the SIO data book for details.

#### **B.1.5.4 X-Bus**

The X-bus is a utility bus, an 8 bit buffered version of the ISA bus, implemented on the reference board to support the native I/O devices that are located on the reference board. X-bus data transceiver U12 is controlled by the ISA bridge via XDIR (UBUSTR) and XDEN# (UBUSOE#). Various devices are located on the X-bus.

#### **B.1.5.5 Control Signal Decodes**

The System I/O EPLD (Chandra) is a programmable logic device that uses the X-bus signals and the partial decode signals from the ISA bridge to decode chip selects for various components. For more information on the System I/O EPLD, see that data sheet.

#### **B.1.5.6 Keyboard/Mouse Controller**

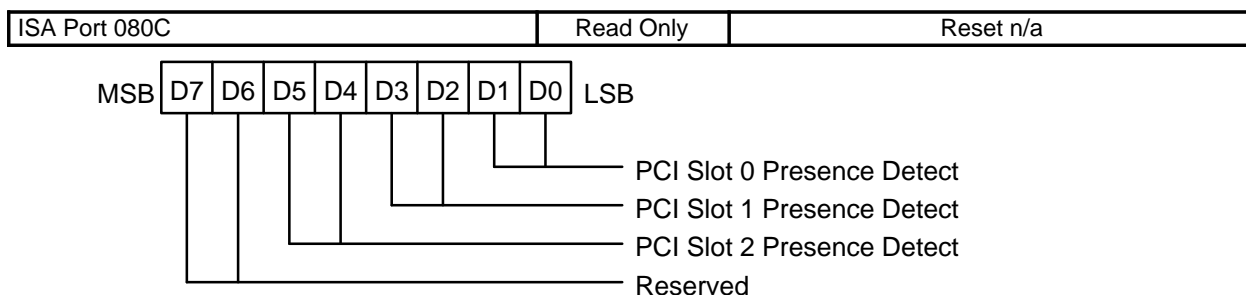
The example planar uses an Intel 8042AH as a keyboard and mouse controller, which resides on the X-bus. The code used is the same version as used in IBM Personal System/2 machines. This microcode may differ from other 8042 type keyboard controllers. These differences are usually only significant when porting AIX to the system (for more information contact your IBM representative. This device contains several registers. See the data sheet for more information.

#### **B.1.5.7 Real Time Clock (RTC)**

The example planar uses a Dallas Semiconductor™ DS1385S to provide the real time clock (TOD or RTC) function. This device is PC compatible and resides on the X-bus. It features an additional 4K of NVRAM and a replaceable battery. This device contains several registers. See the data sheet for more information.

### B.1.5.8 PCI Adapter Card Presence Detect Register

The example planar uses U20 to buffer the PCI adaptor card presence detection bits onto the X-bus under control of the system I/O EPLD. These bits report in pairs, and do not contain any information about the identity of the card. They merely report on its presence.



Bits 7:6 Reserved

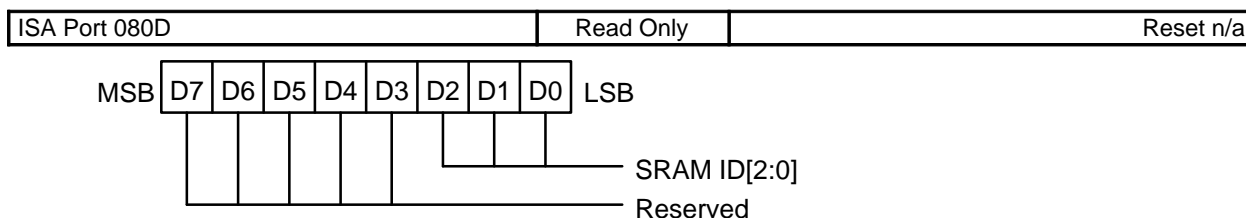
Bits 5:4 PCI Slot 2 Presence Detect Bits. 00 = Present, 11 = No PCI card installed.

Bits 3:2 PCI Slot 2 Presence Detect Bits. 00 = Present, 11 = No PCI card installed.

Bits 1:0 PCI Slot 2 Presence Detect Bits. 00 = Present, 11 = No PCI card installed.

### B.1.5.9 L2 SRAM Identification Register

The example planar uses U19 to buffer the SRAM identification/presence detect bits from the SRAM socket onto the X-bus under control of the system I/O EPLD.



**Table B-2. DRAM Module Presence Detect Bit Encoding**

SRAM ID [2:0]			SRAM Module Identification
ID3	ID2	ID1	
0	0	0	512K Synchronous
0	0	1	256K Synchronous
0	1	0	Reserved
0	1	1	Reserved
1	0	0	512K Asynchronous
1	0	1	256K Asynchronous
1	1	0	1M Asynchronous
1	1	1	L2 SRAM module not present

### B.1.5.10 Planar ID Detection Register

Revision information on the planar (motherboard) is buffered onto the X-bus by U18, under control of the system I/O EPLD.

ISA Port 0852	Read Only	Reset n/a
---------------	-----------	-----------

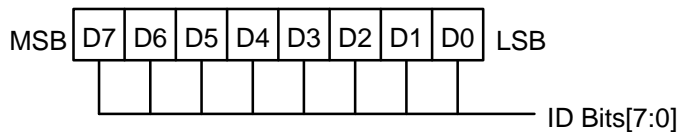


Table B-3. Planar ID Encoding

Planar ID[7:0]	CPU	CPU Internal Clock/ CPU Bus Clock/ PCI Bus Clock	Schematic
0C	603	66/66/33	MPRH02SCU-01
0D	603e	99/66/33	MPRH02SCU-01
0E	604	132/66/33	MPRH02SCU-01
other	—	—	Reserved

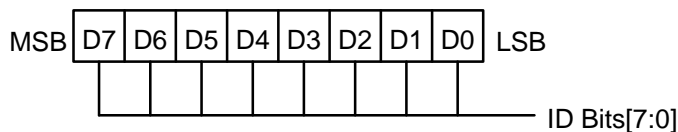
**B.1.5.11 DRAM Presence Detection**

DRAM module presence and identification data is hard coded into the pinout of the SIMM (or DIMM) by shorting particular pins to ground or no-connecting them on the SIMM itself. The example planar uses U17 and U40 to buffer the presence detect bits from the DRAM sockets onto the X-bus under control of signals from the system I/O EPLD. This information appears as the DRAM SIMM 1-2 Memory ID Register and the DRAM SIMM 3-4 Memory ID Register.

**B.1.5.12 DRAM SIMM 1-2 Memory ID Register**

ISA Port 0880	Read Only	Reset n/a
---------------	-----------	-----------

This register indicates the ID bits associated with SIMMs 1 and 2.



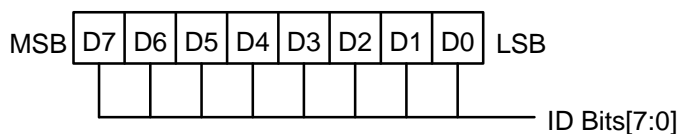
Bits 7:4 SIMM 2 ID bits. See Table B-4.

Bits 3:0 SIMM 1 ID bits. See Table B-4.

**B.1.5.13 DRAM SIMM 3-4 Memory ID Register**

ISA Port 0881	Read Only	Reset n/a
---------------	-----------	-----------

This register indicates the ID bits associated with SIMMs 3 and 4.



Bits 7:4 SIMM 4 ID bits. See Table B-4.

Bits 3:0 SIMM 3 ID bits. See Table B-4.

Table B-4. DRAM Module Presence Detect Bit Encoding

Presence Detect Bits				DRAM Module Identification
PD4	PD3	PD2	PD1	
1	0	0	0	4 MB (1M x 36b) 70ns
1	0	1	1	8 MB (2M x 36b) 70ns
1	0	1	0	16 MB (4M x 36b) 70ns
1	1	0	1	32 MB (8M x 36b) 70ns
1	1	1	1	No DRAM module installed.
Other encodings				Not currently defined.

## B.1.6 Miscellaneous

### B.1.6.1 Speaker Support

The example planar has a connector for a small speaker. The speaker output is driven by the timer 2 signal from the ISA bridge. The intended speaker is a typical PC type, 8 ohms and .5W.

## B.2 System EPLD

**Note:** The System I/O EPLD generates system control register access signals from X-bus I/O port transactions, supports power management, and provides various other system functions.

In this section, the system I/O EPLD is referred to as the EPLD.

The EPLD is a programmed Altera™ EPM5130QC100 electrically programmable logic device. For timing and electrical specifications, see that data sheet.

### B.2.1 System Register Support

The EPLD supports both internal and external registers.

#### B.2.1.1 External Register Support

The EPLD supports a group of external registers, which are latches or other devices that are physically located in a device other than the EPLD. As shown in Figure B-1, the EPLD supplies control signals to the external registers, based on a decode of the address and control lines of the X-bus (or ISA bus). In response to the signals from the EPLD, the external register either reads or writes data to the X-bus.

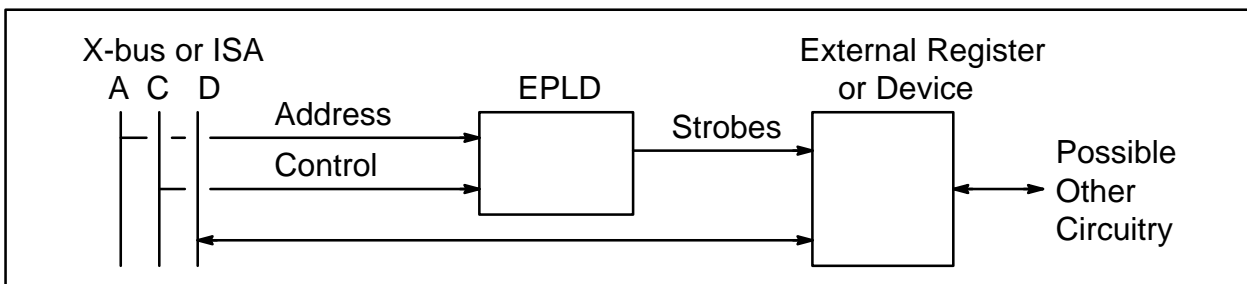


Figure B-1. Typical External Register

For the external registers that the EPLD supports, Table B-5 shows the external register, the ISA I/O port address, and the supplied control signal(s).

Table B-5. External Register Support

ISA Port Address	Register	Read/Write	Register Location	Signal	Strobe or Function
0060 or 0062 or 0064 or 0066	Keyboard Controller Registers	R/W	Keyboard Controller	KBD_CS# (asserted for an access to any of these addresses)	Address Decode
0070	RTC Address Latch Enable	W/O	RTC	RTC_ALE	Write strobe
0071	RTC Data	Write	RTC	RTCWR#	Write strobe
		Read		RTCDs#	Read strobe
0074	NVRAM Address Low Byte	W/O	NVRAM	AS0#	Address Decode
0075	NVRAM Address High Byte	W/O	NVRAM	AS1#	Address Decode
0077	NVRAM Data	Write	NVRAM	NVRAMWE#	Write Strobe
		Read		NVRAMOE#	Read Strobe
080C	Equipment Present	R/O	Board	PRSNT_RD#	Read Strobe
080D	L2 Cache Status Register	R/O	Board	L2_STATUS_RD#	Read Strobe
0852	Planar ID	R/O	Board	PLANAR_ID_RD#	Read Strobe
0880	DRAM Presence Detect 1/2	R/O	Board	DRAM_PD_RD1#	Read Strobe
0881	DRAM Presence Detect 3/4	R/O	Board	DRAM_PD_RD2#	Read Strobe

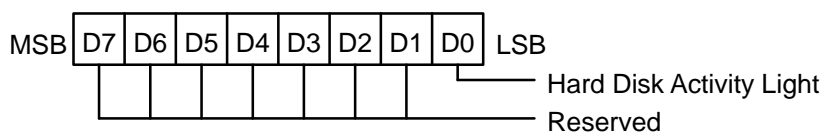
### B.2.1.2 Internal Registers

The EPLD contains a group of internal registers, which are accessed via the ISA bus I/O port address shown for each register.

#### Storage Light Register

ISA Port 0808	Read/Write	Reset to xxxx xxx0
---------------	------------	--------------------

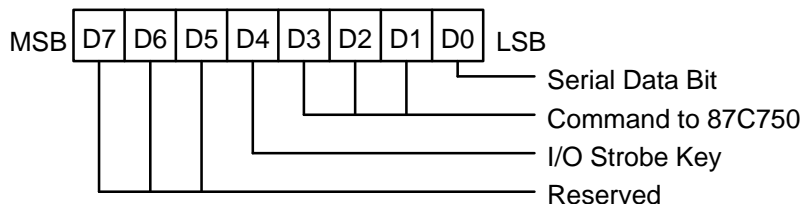
This register controls the HDD\_LED# output of the EPLD. This signal normally controls the hard disc drive activity LED.



Bit 0 . . . Hard Disk Activity Light:  
 0 = Turn light off (negate HDD\_LED#).  
 1 = Turn light on (assert HDD\_LED#).

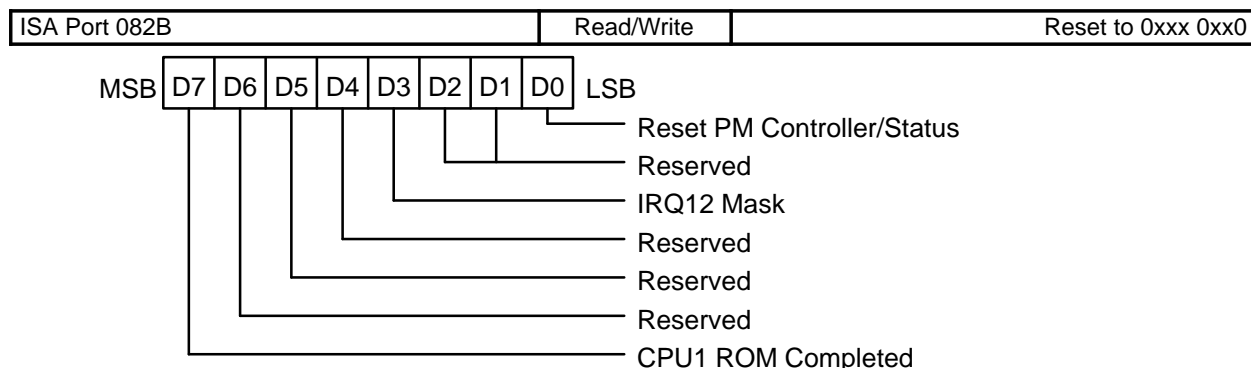
#### Power Management Control Register 1

ISA Port 082A	Read/Write	Reset to xxxx xxxx
---------------	------------	--------------------



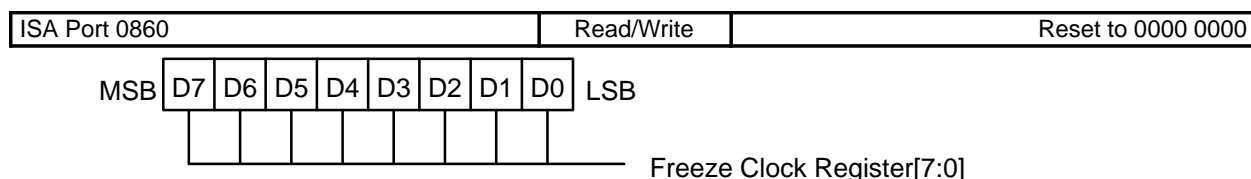
This register is part of the power management system. For a detailed functional description of the operation of this register, see the *603/604 Reference Design Power Management Specification*.

### Power Management Control Register 2



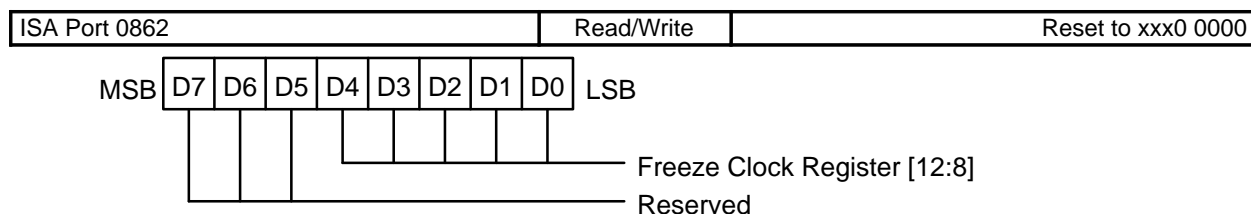
This register is part of the power management system. For a detailed functional description of the operation of this register, see *The 603/604 Reference Design Power Management Specification*.

### Freeze Clock Register (FCR) Low



See FCR high.

### Freeze Clock Register (FCR) High



The freeze clock register (FCR) is a 13 bit register that is accessed by the system via the X-bus as two 8-bit registers, as shown in this and the previous sections. Once triggered, the FCR data is shifted out as serial data on FRZ\_DATA\_OUT.

This function is intended for use with the MPC970 clock chip, as is done in the example planar. The MPC970 contains an input shift register, the input of which (Frz\_Data) is connected to FRZ\_DATA\_OUT. ISA\_CLK is used to clock the data from the EPLD to the MPC970.

EPLD will shift the freeze clock data out of the FCR in response to either one of two triggers:

1. A write to ISA I/O port 862, or

2. A low to high transition on the UNFREEZE input iff bit 0 and bit 1 of the FCR are high.

After triggering the data transfer, wait at least 2.3us for the transfer to complete before accessing the FCR or retriggering the data transfer.

This is a one way serial data transfer between the two devices. Reading the FCR returns the contents of the FCR, and does not cause a read of the data in the MPC970 register.

A 1 in a bit position freezes the corresponding clock output of the MPC970. For details of the data transfer operation and the meaning of the data, see the MPC970 data sheet.

## B.2.2 Signal Descriptions

Table B-6 shows the active signals of the EPLD. Pins not shown should not be connected.

**Table B-6. Signal Descriptions**

Signal Name	Pin	I/O	Description
<b>X-bus Interface Signals</b>			
ECS[2:0]	21, 22, 59	I	Encoded Chip Select [2:0]. Encoded chip selects for peripheral devices supported by the ISA I/O Bridge. Used by EPLD X-bus I/O port address decoders. From SIO.
ECSEN#	60	I	Encoded Chip Select Enable. Asserted to enable the base decoder. Negated to select the option decoder. Used by EPLD X-bus I/O port address decoders. From SIO.
XA[7:0]	61, 64, 65, 66, 67, 70, 71, 72.	I	X-address bus [7:0]. Used by EPLD X-bus I/O port address decoders.
XD[7:0]	31, 32, 33, 92, 73, 74, 34, 95.	I/O 24mA	X-data bus [7:0]. Used by EPLD to transfer data.
XIOW#	16	I	X-bus I/O Write. This signal indicates that the system is writing to an X-bus I/O device. Used by EPLD X-bus I/O port address decoders.
XIOR#	9	I	X-bus I/O Read. This signal indicates that the system is reading from an X-bus I/O device. Used by EPLD X-bus I/O port address decoders.
<b>External Register Support Signals</b>			
AS0#	50	O 6mA	NVRAM address register low byte write strobe. EPLD asserts this signal to write to X-bus port 0074.
AS1#	49	O 6mA	NVRAM address register high byte write strobe. EPLD asserts this signal to write to X-bus port 0075.
DRAM_PD_RD1#	57	O 6mA	DRAM SIMM presence detect read enable 1. EPLD asserts this signal to read X-bus port 0880.
DRAM_PD_RD2#	56	O 6mA	DRAM SIMM presence detect read enable 2. EPLD asserts this signal to read X-bus port 0881.
KYBD_CS#	98	O 6mA	Keyboard chip select. EPLD asserts this signal to read X-bus ports 0060, 0062, 0064, and 0066.
L2_STATUS_RD#	54	O 6mA	L2 cache status register read strobe. EPLD asserts this signal to read X-bus port 080D.
NVRAMOE#	1	O 6mA	NVRAM output enable (read strobe). EPLD asserts this signal to read X-bus port 0076. This normally causes a read of the NVRAM data stored at the location contained in the NVRAM address register.



Table B-6. Signal Descriptions (Continued)

Signal Name	Pin	I/O	Description
<b>External Register Support Signals</b>			
NVRAMWE#	53	O 6mA	NVRAM data write strobe. EPLD asserts this signal to write to X-bus port 0076. This normally causes the data associated with this write to be written into the NVRAM location contained in the NVRAM address register.
PLANAR_ID_RD#	52	O 6mA	Planar ID read. EPLD asserts this signal to read X-bus port 0852.
PRSNT_RD#	41	O 6mA	Equipment present register read. EPLD asserts this signal to read X-bus port 080C.
RTC_ALE	48	O 6mA	Real time clock address latch enable. EPLD asserts this signal to write X-bus port 0070.
RTCDs#	51	O 6mA	Real time clock read strobe. EPLD asserts this signal to read X-bus port 0071.
RTCWR#	35	O 6mA	Real time clock write strobe. EPLD asserts this signal to write to X-bus port 0071.
<b>Interrupt Signals</b>			
IRQ1	86	O	Interrupt request 1. Latched active when IRQ1_IN is asserted. Negated when KYBD_CS# is asserted.
IRQ1_IN	89	I	Keyboard interrupt. EPLD is designed to intercept the keyboard interrupt between the keyboard and the ISA bus bridge. Either connect IRQ1 and IRQ1_IN as shown in the example planar schematics, or disconnect both signals from the system (routing the keyboard interrupt to the ISA bus bridge), or see the <i>603/604 Reference Design Power Management Guide</i> .
IRQ12	36	I	Interrupt request 12 input. Connect to system IRQ12, the mouse interrupt. Also see the <i>603/604 Reference Design Power Management Guide</i> .
<b>System Clock Interface Signals</b>			
FRZ_DATA_OUT	90	O 6mA	Freeze data out. Serial data stream to MPC970 clock chip. See the MPC970 data sheet.
ISA_CLK	20	I	ISA clock. Used to clock the freeze serial data stream to the MPC970 clock chip. See the MPC970 data sheet. Also see the <i>603/604 Reference Design Power Management Guide</i> .
<b>Power Management Signals</b> (not used in release 2.0 or 2.1)			
83CX_RESET	91	O 6mA	Power management controller reset. No-connect or connect as shown in the <i>603/604 Reference Design Power Management Guide</i> .
ACTIVITY#	58	O 6mA	Activity. No-connect or connect as shown in the <i>603/604 Reference Design Power Management Guide</i> .
CMD_STATE#	8	I	Power management controller command state. No-connect or connect as shown in the <i>603/604 Reference Design Power Management Guide</i> .
EXT_ACTVITY#	3	I	External activity. No-connect or pull low or connect as shown in the <i>603/604 Reference Design Power Management Guide</i> .
IO_STROBE#	99	O 6mA	I/O strobe. No-connect or connect as shown in the <i>603/604 Reference Design Power Management Guide</i> .
PROC_RDY	83	I	Power management controller ready. No-connect or connect as shown in the <i>603/604 Reference Design Power Management Guide</i> .
RWD0	96	I/O 24mA	Power management controller serial read/write data bit. No-connect or connect as shown in the <i>603/604 Reference Design Power Management Guide</i> .

Table B-6. Signal Descriptions (Continued)

Signal Name	Pin	I/O	Description
<b>Power Management Signals</b> (not used in release 2.0 or 2.1)			
UNFREEZE	82	I	Unfreeze. No-connect or connect as shown in the <i>603/604 Reference Design Power Management Guide</i> .
<b>Other Signals</b>			
HDD_LED#	100	O 6mA	Hard disk drive activity light. EPLD asserts this signal while bit 0 of the storage light register (port 0808) is 1. This signal normally indicates hard disk drive activity.
RESET#	10	I	System reset. Used by EPLD to reset internal state machines and internal registers.
VCC	18, 19, 43, 44, 68, 69, 93, 94	I	+5V:
GND	12, 13, 37, 38, 62, 63, 87, 88	I	GROUND:

## B.2.3 EPLD Design Equations

### B.2.3.1 Fit File

```
-- MAX+plus II Compiler Fit File
-- Version 5.0 8/5/94
-- Compiled: 05/05/95 14:44:22
```

```
BEGIN
```

```
    DEVICE = "EPM5130WC-1";
    "A0"           : INPUT_PIN = 72      ;
    "A1"           : INPUT_PIN = 71      ;
    "A2"           : INPUT_PIN = 70      ;
    "A3"           : INPUT_PIN = 67      ;
    "A4"           : INPUT_PIN = 66      ;
    "A5"           : INPUT_PIN = 65      ;
    "A6"           : INPUT_PIN = 64      ;
    "A7"           : INPUT_PIN = 61      ;
    "/CMD_STATE"   : INPUT_PIN = 8       ; -- LC8
    "/ECSEN"       : INPUT_PIN = 60      ;
    "ECS0"         : INPUT_PIN = 59      ;
    "ECS1"         : INPUT_PIN = 22      ;
    "ECS2"         : INPUT_PIN = 21      ;
    "/EXT_ACTVTY"  : INPUT_PIN = 3       ; -- LC3
    "IRQ1"         : INPUT_PIN = 89      ; -- LC103
    "IRQ12"        : INPUT_PIN = 36      ; -- LC38
    "ISA_CLK"      : INPUT_PIN = 20      ;
    "PROC_RDY"     : INPUT_PIN = 83      ; -- LC99
    "/RESET"       : INPUT_PIN = 10      ;
    "UNFREEZE"     : INPUT_PIN = 82      ; -- LC98
    "/XIOR"        : INPUT_PIN = 9       ;
    "/XIOW"        : INPUT_PIN = 16      ;
```

"/ACTIVITY"	: OUTPUT_PIN = 58	; -- LC72
"/AS0"	: OUTPUT_PIN = 50	; -- LC56
"/AS1"	: OUTPUT_PIN = 49	; -- LC55
"/DRAM_PD_RD1"	: OUTPUT_PIN = 57	; -- LC71
"/DRAM_PD_RD2"	: OUTPUT_PIN = 56	; -- LC70
"FRZ_DATA_OUT"	: OUTPUT_PIN = 90	; -- LC104
"/HDD_LED"	: OUTPUT_PIN = 100	; -- LC120
"/IO_STROBE"	: OUTPUT_PIN = 99	; -- LC119
"IRQ1_OUT"	: OUTPUT_PIN = 86	; -- LC102
"/KYBD_CS"	: OUTPUT_PIN = 98	; -- LC118
"/L2_STATUS_RD"	: OUTPUT_PIN = 54	; -- LC68
"/NVRAMOE"	: OUTPUT_PIN = 1	; -- LC1
"/NVRAMWE"	: OUTPUT_PIN = 53	; -- LC67
"/PLANAR_ID_RD"	: OUTPUT_PIN = 52	; -- LC66
"/PRSNT_RD"	: OUTPUT_PIN = 41	; -- LC49
"RTC_ALE"	: OUTPUT_PIN = 48	; -- LC54
"/RTCDs"	: OUTPUT_PIN = 51	; -- LC65
"/RTCWR"	: OUTPUT_PIN = 35	; -- LC37
"83CX_RESET"	: OUTPUT_PIN = 91	; -- LC113
"RWD0"	: BIDIR_PIN = 96	; -- LC116
"XD0"	: BIDIR_PIN = 95	; -- LC115
"XD1"	: BIDIR_PIN = 34	; -- LC36
"XD2"	: BIDIR_PIN = 74	; -- LC82
"XD3"	: BIDIR_PIN = 73	; -- LC81
"XD4"	: BIDIR_PIN = 92	; -- LC114
"XD5"	: BIDIR_PIN = 33	; -- LC35
"XD6"	: BIDIR_PIN = 32	; -- LC34
"XD7"	: BIDIR_PIN = 31	; -- LC33
"CLKFF0"	: LOCATION = LC48	;
"CLKFF1"	: LOCATION = LC96	;
"CLKFF2"	: LOCATION = LC95	;
"CLKFF3"	: LOCATION = LC94	;
"CLKFF4"	: LOCATION = LC47	;
"CLKFF5"	: LOCATION = LC46	;
"CLKFF6"	: LOCATION = LC45	;
"CLKFF7"	: LOCATION = LC44	;
"CLKFF8"	: LOCATION = LC43	;
"CLKFF9"	: LOCATION = LC93	;
"CLKFF10"	: LOCATION = LC92	;
"CLKFF11"	: LOCATION = LC91	;
"CLKFF12"	: LOCATION = LC112	;
"CNTR0"	: LOCATION = LC111	;
"CNTR1"	: LOCATION = LC110	;
"CNTR2"	: LOCATION = LC109	;
"CNTR3"	: LOCATION = LC108	;
"DOUBLE_FRZ"	: LOCATION = LC90	;

```

"DOUBLE_SNC"           : LOCATION = LC107 ;
"FREEZEFF"             : LOCATION = LC106 ;
"GEN_STOP_BITFF"       : LOCATION = LC105 ;
"GPCS0"                : LOCATION = LC89 ;
"PWR_REG21"            : LOCATION = LC80;  --IRQ12_MASK
"PWR_REG22"            : LOCATION = LC79 ;
"SHIFT_ENFF"           : LOCATION = LC103 ; -- PIN 89
"SNC_FREEZE"           : LOCATION = LC99 ; -- PIN 83
"SNC_SHIFT_ENFF"       : LOCATION = LC98 ; -- PIN 82
"SNC_UNFREEZE"         : LOCATION = LC101 ; -- PIN 85
"START_SHIFTFF"        : LOCATION = LC100 ; -- PIN 84
"UNFREEZEFF"           : LOCATION = LC97 ; -- PIN 81

```

```
END;
```

### B.2.3.2 TDF File

```

%*****
%***** AHDL SOURCE CODE FOR: SIO/XBUS INTERFACE CONTROL *****
%*****

```

```
SUBDESIGN chandra(
```

```
%%
```

```
%          DEFINE PRIMARY INPUTS AND OUTPUTS          %
```

```
%%
```

```

A[7..0]           :INPUT; % A7 is MSB and A0 is LSB %
ECS[2..0]         :INPUT; % ECS2 is MSB and ECS0 is LSB %
/ECSEN            :INPUT;
/XIOR             :INPUT;
/XIOW             :INPUT;
/RESET            :INPUT;
/CMD_STATE        :INPUT;
PROC_RDY          :INPUT;
/EXT_ACTVTY       :INPUT;
IRQ1              :INPUT;
IRQ12             :INPUT;% Freeze Clock %
ISA_CLK           :INPUT;
UNFREEZE          :INPUT;

```

```

/PLANAR_ID_RD     :OUTPUT;
/PRSNT_RD         :OUTPUT;
/L2_STATUS_RD     :OUTPUT;
/DRAM_PD_RD1      :OUTPUT;
/DRAM_PD_RD2      :OUTPUT;
/KYBD_CS          :OUTPUT;
IRQ1_OUT          :OUTPUT;
RTC_ALE           :OUTPUT;
/RTCDs            :OUTPUT;
/RTCWR            :OUTPUT;
/AS0              :OUTPUT;
/AS1              :OUTPUT;
/NVRAMWE          :OUTPUT;
/NVRAMOE          :OUTPUT;
/HDD_LED          :OUTPUT;

```

```

83CX_RESET      :OUTPUT;
/ACTIVITY       :OUTPUT;
/IO_STROBE      :OUTPUT;
FRZ_DATA_OUT    :OUTPUT;

XD[7..0]        :BIDIR;    % XD7 is MSB and XD0 is LSB %
RWD0            :BIDIR;

)

VARIABLE                                % Misc. Variables%
GPCS0           :NODE;
HDD_LEDFF       :SRFF;
D[7..0]         :NODE;
XD_TRI_OE       :NODE;
LIGHT_STRB      :NODE;
INT1FF          :DFF;      % Keyboard interput latch %
CLRINT1         :NODE;     % Power Management Variables%
PWR_REG1_STRB   :NODE;
PWR_REG2_STRB   :NODE;
PWR_REG2[2..0]  :SRFF;
83CX_CS         :NODE;
RWD0_STRB       :NODE;
IRQ12_MASK      :NODE;     % Freeze Clock Variables %
CLKFF[12..0]    :DFFE;
SHIFT_ENFF      :SRFF;
CNTR[3..0]      :DFF;
SNC_SHIFT_ENFF  :DFFE;
START_SHIFTFF   :SRFF;
CLKFF_WR        :NODE;
CLKFF_STRB      :NODE;
CLKFF_SELL      :NODE;
CLKFF_SELH      :NODE;
GEN_START_BIT   :NODE;
GEN_STOP_BITFF  :DFF;
STOP_SHIFT      :NODE;
UNFREEZEFF      :DFFE;     % Unsynchronized UNFREEZE          %
SNC_UNFREEZE    :DFF;      % Synchronized UNFREEZE          %
DOUBLE_SNC      :DFF;      % Double Synchronized UNFREEZE %
FREEZEFF        :DFFE;     % Unsynchronized FREEZE          %
SNC_FREEZE      :DFF;      % Synchronized FREEZE          %
DOUBLE_FRZ      :DFF;      % Double Synchronized FREEZE %

BEGIN%*****%

% CHANDRA uses SIO's General Purpose Register Decode 0 -- Software sets      %
% to 800-8FF to enable the following decoded signals.                        %

GPCS0 = LCELL (!ECS[2] & ECS[1] & !ECS[0] & /ECSEN);

% Hard Disk Light I/O address range : 0808                                  %

LIGHT_STRB = !A[7] & !A[6] & !A[5] & !A[4] & A[3] & !A[2]
            & !A[1] & !A[0] & GPCS0; HDD_LEDFF.s = XD[0] & LIGHT_STRB;
HDD_LEDFF.r = !XD[0] & LIGHT_STRB;
HDD_LEDFF.clk= GLOBAL(/XIOW);
HDD_LEDFF.clrn= (/RESET);

```

```

/HDD_LED      = !HDD_LEDFF.q;
%%
% Equipment present Read Command 1 (I/O address range: 080C)          %
%%
/PRSNT_RD     = ! (!A[7] & !A[6] & !A[5] & !A[4] & A[3] & A[2]
                  & !A[1] & !A[0] & GPCS0 & !/XIOR);
%%
% Equipment present Read Command 2 (I/O address range: 080D)          %
%%
/L2_STATUS_RD = !(!A[7] & !A[6] & !A[5] & !A[4] & A[3] & A[2]
                  & !A[1] & A[0] & GPCS0 & !/XIOR);
%%
% External/Internal Planar ID I/O address range: 0852                %
%%
/PLANAR_ID_RD = !(!A[7] & A[6] & !A[5] & A[4] & !A[3] & !A[2]
                  & A[1] & !A[0] & GPCS0 & !/XIOR);
%%
% Keyboard Chip Select I/O address range = 0060, 0062, 0064, 0066-    %
%%
/KYBD_CS = !(!ECS[2] & ECS[1] & !ECS[0] & !/ECSEN);

% Keyboard interrupt (INT1) Clear = KBD_CS qualified with XIOR %

CLRINT1 = !ECS[2] & ECS[1] & !ECS[0] & !/ECSEN & !/XIOR; INT1FF.d=VCC;
INT1FF.clk = IRQ1;          % Set Keyboard INT1 latch on rising edge of IN1 %
INT1FF.clrn = /RESET & !CLRINT1;
IRQ1_OUT = INT1FF.q;

%%
% RTC_ALE I/O address 0070                                           %
%%
RTC_ALE = (!A[2] & !A[1] & !A[0] & !ECS[2] & !ECS[1] & !/ECSEN & !/XIOW) ;
%%

% RTCWR I/O address 0071                                           %
%%
/RTCWR = !(!A[2] & !A[1] & A[0] & !ECS[2] & !ECS[1] & !/ECSEN & !/XIOW);
%%

% RTCDS I/O address 0071                                           %
%%
/RTCDS = !(!A[2] & !A[1] & A[0] & !ECS[2] & !ECS[1] & !/ECSEN & !/XIOR);
%%

% Nvram AS0 I/O address 0074                                         %
%%
/AS0 = !(A[2] & !A[1] & !A[0] & !ECS[2] & !ECS[1] & !/ECSEN & !/XIOW);
%%

% Nvram AS1 I/O address range: 0075                                  %
%%
/AS1 = !(A[2] & !A[1] & A[0] & !ECS[2] & !ECS[1] & !/ECSEN & !/XIOW);
%%

% Nvram RTC WER I/O address 0077                                     %
%%

```

```
/NVRAMWE = !(A[2] & A[1] & A[0] & !ECS[2] & !ECS[1] & !/ECSEN & !/XIOW);  
%%  
  
% Nvram RTC OER I/O address range: 0077 %  
%%  
/NVRAMOE = !(A[2] & A[1] & A[0] & !ECS[2] & !ECS[1] & !/ECSEN & !/XIOR)  
; %%  
  
% DRAM_PD_RD1 enables Present Detect I/O address range: 0880 %  
%%  
/DRAM_PD_RD1 = !(A[7] & !A[6] & !A[5] & !A[4] & !A[3] & !A[2] & !A[1]  
                & !A[0] & GPCS0 & !/XIOR);
```

```

%%
% DRAM_PD_RD2 enables Present Detect I/O address range: 0881
%%
/DRAM_PD_RD2 = !(A[7] & !A[6] & !A[5] & !A[4] & !A[3] & !A[2] & !A[1]
               & A[0] & GPCS0 & !/XIOR);

%%
%*****%
%                POWER MANAGEMENT                %
%*****%
% Write Power Control Register 1 I/O address: 082A
%
% (MSB)      Bits 7-5      Reserved
%              Bit 4      I/O Strobe Key              (W/O)
%              Bits 3-1    Data/Command B[3..1] to 83C750 (W/O)
% (LSB)      Bit 0      83C750 D0                      (R/W)
%*****%

83CX_CS =      (!A[7] & !A[6] & A[5] & !A[4] & A[3] & !A[2] & A[1] & GPCS0);
PWR_REG1_STRB = 83CX_CS & !A0;

%%
% -Write ZERO to 83C750
%%
RWD0_STRB      = (PWR_REG1_STRB & !/XIOW & !XD[0]);
RWD0           = TRI (GND, RWD0_STRB);

%%
% 83C750 I/O Strobe Key
%%
/IO_STROBE      = !(XD[4] & PWR_REG1_STRB & !/XIOW & PROC_RDY
                  & !83CX_RESET);

%IO_CHRDY       = TRI (GND, IO_STRB);%

%*****%
% Write Power Control Register 2 I/O address: 082B
%
% (MSB)      Bit 7      CPU1 ROM Completed      (Reset by /RESET,      W/R)
%              Bit 6      Reserved
%              Bit 5      Reserved
%              Bit 4      Reserved
%              Bit 3      IRQ12 Mask              (Reset by /RESET)      (W/R)
%              Bits 2-1    Reserved
% (LSB)      Bit 0      Reset 83C750              (W/O)
%*****%
PWR_REG2_STRB = 83CX_CS & A0;
PWR_REG2[1].s = XD[3] & PWR_REG2_STRB;
PWR_REG2[1].r = !XD[3] & PWR_REG2_STRB;
PWR_REG2[1].clrn = (/RESET);
PWR_REG2[2].s = XD[7] & PWR_REG2_STRB;
PWR_REG2[2].r = !XD[7] & PWR_REG2_STRB;
PWR_REG2[2].clrn = (/RESET);
PWR_REG2[0].s = XD[0] & PWR_REG2_STRB;
PWR_REG2[0].r = !XD[0] & PWR_REG2_STRB;

```



```

PWR_REG2[0].clrn = VCC;
PWR_REG2[1].clk  = GLOBAL(/XIOW);
83CX_RESET      = PWR_REG2[0].q;
IRQ12_MASK      = PWR_REG2[1].q;

%*****%
% This output should be NOred with IRQ1 and EXTACTIV signals externally to%
% provide a real Activity Alert to the 87C350 (because of restriction of %
% CHANDRA's IO pins.)
%
%*****%
/ACTIVITY = !(IRQ1 # !IRQ12_MASK & IRQ12 # !/EXT_ACTVTY);

%*****%
% Freeze Clock Logic - Refer to the Motorola MC88LV970 PLL Clock Driver %
% Specification for information on Freeze Data protocol. %
%
% Freeze Clock Logic contains a 12 Bit serial shift register %
% Decode the low order CLKFF[7..0] on addresses: 0860 -0861 %
% Decode the high order CLKFF[12..8] on addresses: 0862 -0863 %
%*****%CL
KFF_STRB      = (!A[7] & A[6] & A[5] & !A[4] & !A[3] & !A[2] & GPCS0);
CLKFF_WR      = CLKFF_STRB & !/XIOW;
CLKFF_SELL    = CLKFF_STRB & !A[1];
CLKFF_SELH    = CLKFF_STRB & A[1];
CLKFF[12..0].prn = VCC;
CLKFF[12..0].clrn = /RESET;
CLKFF[12..0].ena = START_SHIFTFF.q # CLKFF_WR;
CLKFF[12..0].clk = ISA_CLK;

if START_SHIFTFF.q then % Shift with wraparound %
    CLKFF[11..0].d = CLKFF[12..1].q;
    CLKFF[12].d    = CLKFF[0].q;
else if CLKFF_SELL then % Write to CLKFF[7..0] %
    CLKFF[7..0].d = XD[7..0];
    CLKFF[12..8].d = CLKFF[12..8].q;
    else if CLKFF_SELH then
        CLKFF[7..0].d = CLKFF[7..0].q; % Write to CLKFF[12..8] %
        CLKFF[12..8].d = XD[4..0];
    else
        CLKFF[12..0].d = CLKFF[12..0].q; % Do nothing %
    end if;
end if;
end if;

%*****%
% UNFREEZE Flip Flop is cleared asynchronously when the UNFREEZE signal %
% makes a low to high transition. It is set once shifting has been enabled%
% The CHANDRA ignores the UNFREEZE signal if the 601 PCLK clocks are not %
% frozen. %
%*****%UN-
FREEZEFF.prn   = !SNC_SHIFT_ENFF.q & /RESET;
UNFREEZEFF.clrn = VCC;
UNFREEZEFF.ena = !SHIFT_ENFF.q ;
UNFREEZEFF.d   = !(CLKFF[0].q # CLKFF[1].q);

```

```

UNFREEZEFF.clk = UNFREEZE ;

%*****%
% This signal is the UNFREEZE Flip Flop synchronized to the ISA clock.    %
%*****%
SNC_UNFREEZE.prn = /RESET;
SNC_UNFREEZE.clrn = VCC;
SNC_UNFREEZE.d   = UNFREEZEFF.q;
SNC_UNFREEZE.clk = ISA_CLK;

%*****%
%
% This signal is the UNFREEZE Flip Flop double synchronized to the ISA
clock.%
%*****%
%
DOUBLE_SNC.prn = /RESET;
DOUBLE_SNC.clrn = VCC;
DOUBLE_SNC.d   = SNC_UNFREEZE.q;
DOUBLE_SNC.clk = ISA_CLK;

%*****%
% FREEZE Flip Flop is cleared asynchronously when the UNFREEZE signal    %
% makes a high to low transition. It is set once shifting has been enabled%
% The CHANDRA ignores the UNFREEZE signal if the 601 PCLK clocks are      %
% already frozen.                                                         %
%*****%
FREEZEFF.prn = !SNC_SHIFT_ENFF.q & /RESET;
FREEZEFF.clrn = VCC;
FREEZEFF.ena = !SHIFT_ENFF.q ;
FREEZEFF.d   = CLKFF[0].q & CLKFF[1].q;
FREEZEFF.clk = !UNFREEZE ;

%*****%
% This signal is the FREEZE Flip Flop synchronized to the ISA clock.    %
%*****%
SNC_FREEZE.prn = /RESET;
SNC_FREEZE.clrn = VCC;
SNC_FREEZE.d   = FREEZEFF.q;
SNC_FREEZE.clk = ISA_CLK;

%*****%
%
% This signal is the FREEZE Flip Flop double synchronized to the ISA
clock.%
%*****%
%
DOUBLE_FRZ.prn = /RESET;
DOUBLE_FRZ.clrn = VCC;
DOUBLE_FRZ.d   = SNC_FREEZE.q;
DOUBLE_FRZ.clk = ISA_CLK;
SHIFT_ENFF.s   = CLKFF_SELH;      % Start shifting upon write to 862      %
SHIFT_ENFF.r   = GND;
SHIFT_ENFF.prn = DOUBLE_SNC.q & DOUBLE_FRZ;
SHIFT_ENFF.clrn = !GEN_STOP_BITFF.q & /RESET; %Clr when it reaches 15 & rst%
SHIFT_ENFF.clk = /XIOW;SNC_SHIFT_ENFF.d = SHIFT_ENFF.q;

```

```

% One clock after SHIFT_ENFF%
SNC_SHIFT_ENFF.ena = /RESET;
SNC_SHIFT_ENFF.clrn = /RESET;
SNC_SHIFT_ENFF.clk = ISA_CLK;
CNTR[].clk = ISA_CLK;
CNTR[].clrn = /RESET;          % Counter resets to zero          %

If SNC_SHIFT_ENFF.q Then      % Count while SNC_SHIFT_ENFF is 1    %
    CNTR[].d = CNTR[].q + 1;
Else
    CNTR[].d = CNTR[].q;
End If;

GEN_START_BIT = !CNTR[3].q & !CNTR[2].q & !CNTR[1].q & CNTR[0].q; % 1 %
STOP_SHIFT    = CNTR[3].q & CNTR[2].q & CNTR[1].q & !CNTR[0].q; % 14 %
GEN_STOP_BITFF.d = STOP_SHIFT; %Generate Stop Bit when counter is 15 %
GEN_STOP_BITFF.prn = VCC;
GEN_STOP_BITFF.clrn = /RESET;
GEN_STOP_BITFF.clk = ISA_CLK;
START_SHIFTFF.s    = GEN_START_BIT;
START_SHIFTFF.r    = STOP_SHIFT;
START_SHIFTFF.clrn = /RESET;
START_SHIFTFF.clk  = ISA_CLK;

%*****%
% FRZ_DATA_OUT is a 1 when the counter = 0. It is also a 1 whenever %
% START_SHIFTFF is 1 and CLKFF[0] is a 1 and then when the counter is %
% Fifteen to leave it in the high state. %
%*****%
FRZ_DATA_OUT = !CNTR[3].q & !CNTR[2].q & !CNTR[1].q & !CNTR[0].q
               # GEN_STOP_BITFF.q
               # START_SHIFTFF.q & !CLKFF[0].q;

%*****%
% Read Storage Light Status Register I/O address: 0808 %
% (MSB) Bits 7-1 Reserved %
% (LSB) Bit 0 Hard Disk Active Light (W/R) %
%
% Read Power Control Register 1 I/O address: 082A %
% (MSB) Bits 7-1 Reserved %
% (LSB) Bit 0 83C750 D0 (W/R) %
%
% Read Power Control Register 2 I/O address: 082B %
% (MSB) Bit 7 CPU1 ROM Completed (Reset by /RESET, R/W) %
% Bit 6 Reserved %
% Bit 5 Reserved %
% Bit 4 Reserved %
% Bit 3 IRQ12 Mask (Reset by /RESET, W/R) %
% Bits 2-1 Reserved %
% (LSB) Bit 0 83C750 Status (R/O) %
%
%*****%
XD_TRI_OE = ((LIGHT_STRB
               # PWR_REG1_STRB # PWR_REG2_STRB
               # CLKFF_STRB) & !/XIOR);

```

```
XD[0]      = TRI (D[0], XD_TRI_OE);
D[0]       = HDD_LEDFF & LIGHT_STRB
            # RWD0 & PWR_REG1_STRB
            # PROC_RDY & /CMD_STATE & PWR_REG2_STRB
            # CLKFF[0].q & CLKFF_SELL
            # CLKFF[8].q & CLKFF_SELH;

XD[1]      = TRI (D[1], XD_TRI_OE);
D[1]       = CLKFF[1].q & CLKFF_SELL
            # CLKFF[9].q & CLKFF_SELH;

XD[2]      = TRI (D[2], XD_TRI_OE);
D[2]       = CLKFF[2].q & CLKFF_SELL
            # CLKFF[10].q & CLKFF_SELH;

XD[3]      = TRI (D[3], XD_TRI_OE);
D[3]       = PWR_REG2[1].q & PWR_REG2_STRB
            # CLKFF[3].q & CLKFF_SELL
            # CLKFF[11].q & CLKFF_SELH;

XD[4]      = TRI (D[4], XD_TRI_OE);
D[4]       = CLKFF[4].q & CLKFF_SELL
            # CLKFF[12].q & CLKFF_SELH;

XD[5]      = TRI (D[5], XD_TRI_OE);
D[5]       = CLKFF[5].q & CLKFF_SELL;
XD[6]      = TRI (D[6], XD_TRI_OE);
D[6]       = CLKFF[6].q & CLKFF_SELL;
XD[7]      = TRI (D[7], XD_TRI_OE);
D[7]       = PWR_REG2[2].q & PWR_REG2_STRB
            # CLKFF[7].q & CLKFF_SELL;

END;
```

## **Appendix C**

### **Planar Set Up and Registers**

This section describes the registers and setup considerations that are a function of the example planar, as opposed to the MCM. These settings and registers are not in general required by the MCM in every system. This information is included only for reference.

Designers of systems containing components similar to those found on the example planar may wish to start their development work using these settings.

#### **C.1 ISA Bridge (SIO) Initialization**

The example planar uses an Intel 82378ZB SIO as the ISA bridge.

The SIO chip must be configured prior to any other PCI bus agent. The SIO PCI arbiter is automatically enabled upon power-on reset. During power-on reset, the SIO drives the A/D(31:0), C/BE#(3:0), and par signals on the PCI bus.

The system I/O EPLD uses the decode circuits in the SIO that produce the signals EC-SADDR[2:0] and UBUSCOE# to decode the motherboard register addresses. For this reason, utility bus A and B decode registers must be initialized as shown in Table C-1.

The ISA clock divisor must be set as indicated prior to running any CPU to PCI transactions. If the configuration information is stored in Flash, this should pose no problem.

The SIO must be programmed so that interval timer 1 operates in mode 2 with a period of approximately 15 microseconds. This timer controls the ISA refresh interval. It must be programmed at least 200 microseconds before any access to ISA DRAM is attempted.

PCI memory write cycles destined for ISA can use a 32-bit posted write buffer in the SIO. Bit 2 of the PCI control register controls the enabling of the posted write buffer. The default (power-on reset) state for the posted write buffer is disabled. It is required that the posted write buffer be enabled.

Note that PCI burst transactions are not supported by the SIO. For burst transactions, the SIO will always target abort after the first data phase. The system will not allow the CPU to burst to the SIO (or any other PCI agent). No PCI master should be programmed to attempt burst transactions to the SIO.

The SIO defaults (after power-on reset) to the slow sampling point (bits 4:3 of the PCI Control Register) for its subtractive decode. Of the three choices for the sampling point: slow (5 PCI cycles), typical (4 PCI cycles) and fast (3 PCI cycles), one should be chosen that is one clock after the slowest I/O device on the PCI bus. If the PCI agents are all memory mapped above 16M Byte and all I/O mapped above 64K, then the fast sampling point for

the subtractive decode can be chosen. This insures that no other PCI agent except the SIO will claim these addresses. Configure PCI agents in this manner to improve performance.

The SIO automatically inserts a 4 ISA clock cycle delay between PCI originated back-to-back 8 and 16 bit I/O cycles to the ISA bus. In addition, the ISA Controller Recovery Timer Register (configuration register, address offset=4Ch) enables a number of additional ISA clock cycles of delay to be inserted between these types of back-to-back I/O cycles. The ISA Controller Recovery Timer Register defaults (after power-on reset) to 2 additional ISA clock cycles of delay, making the total delay equal to 6 ISA clock cycles, for both the 8 and 16 bit I/O recovery times. Since none of the native I/O devices on the example planar require such long recovery times, the additional cycles specified by the ISA Controller Timer Register can be disabled. If an ISA card requiring a long recovery time is supported, the driver should insure that the recovery time is met.

Disable scatter/gather mode and GAT mode.

Do not attempt to access DMA channel 4 address and byte count registers.

Always enable the ISA master and DMA buffers. In order to isolate slow ISA Bus I/O devices from the PCI bus, the DMA controller uses the DMA/ISA master Line Buffer. This buffer can operate in single transaction or in 8-byte mode. Bits 0-LE/7-BE and 1-LE/6-BE of the PCI Control register configure the line buffer for DMA and ISA masters separately. It is required that the 8-byte mode be enabled for both (Bits = 1,1).

The registers in Table C-1 must be set in order for the example planar I/O hardware to operate properly. Vendors use LE bit nomenclature, and nomenclature within CPU registers is BE.

**Table C-1. Summary of SIO Register Setup (Configuration Address = 8080 08xx)**

Register	Addr	Bit	Set To	Reset Value	DESCRIPTION
PCI Control Register	40h	2-LE 5-BE	1	0	Enable PCI Memory Posted Write Buffer.
PCI Control Register	40h	1-LE 6-BE	1	0	Enable ISA Master Line buffer.
PCI Control Register	40h	0-LE 7-BE	1	0	Enable DMA Line Buffer
PCI Arbiter Control (Config/PCI)	41h	0-LE 7-BE	0	0	Disable GAT (Guaranteed Access Time Mode). Note: GAT does not work in SIO.
ISA Clock Divisor (Config/PCI)	4Dh	5-LE 2-BE	0	0	Disable Coprocessor Error Support.
ISA Clock Divisor (Config/PCI)	4Dh	4-LE 3-BE	1	0	Enable IRQ12/M Mouse Support.
ISA Clock Divisor (Config/PCI)	4Dh	3-LE 4-BE	*	0	* This bit should be set to 1 before changing or loading the PCI ISA Clock Divisor value. Setting this bit to 1 will assert the RSTDVR signal (which resets the System I/O EPLD and any devices on the ISA bus slots). All these devices will require reconfiguration after this bit has been asserted. Software must guarantee that RSTDVR be asserted for a minimum of 1 ms after the clock divisor value is set.
ISA Clock Divisor (Config/PCI)	4Dh	2:0-LE 5:7-BE	*	0	* Set this field to 000b (divisor = 4). (If PCI clock is slower than 33 MHz, then this field would be 001b (divisor=3).

Table C-1. Summary of SIO Register Setup (Configuration Address = 8080 08xx) (Continued)

Register	Addr	Bit	Set To	Reset Value	DESCRIPTION
Utility Bus Chip Select A (Config/PCI)	4Eh	4-LE 4-BE	1	0	Disable generation of ECSADDR(2:0) and UBU-SOE# for the IDE and Floppy decode.
Utility Bus Chip Select A (Config/PCI)	4Eh	1-LE 6-BE	1	1	Enable keyboard addresses (60h, 62h, 64h, 66h).
Utility Bus Chip Select A (Config/PCI)	4Eh	0-LE 7-BE	1	1	Enable TOD Addresses (70h, 71h).
Utility Bus Chip Select B (Config/PCI)	4Fh	7-LE 0-BE	1	0	Enable access to the motherboard registers in the 0800-08FF address range.
Utility Bus Chip Select B (Config/PCI)	4Fh	6-LE 1-BE	1	1	Enable Port 92h access.
Utility Bus Chip Select B (Config/PCI)	4Fh	5:4-LE 2:3-BE	11	00	Disable generation of default address for Parallel Port.
Utility Bus Chip Select B (Config/PCI)	4Fh	3:2-LE 4:5-BE	11	00	Disable generation of default address for Serial Port B.
Utility Bus Chip Select B (Config/PCI)	4Fh	1:0-LE 6:7-BE	11	00	Disable generation of default address for Serial Port A.
Interrupt Controller 1 - ICW1 (I/O /PCI)	20h	3-LE 4-BE	0	x	Set Interrupt Controller 1 to edge triggered mode.
Interrupt Controller 1 - ICW1 (I/O /PCI)	20h	1-LE 6-BE	0	x	Set Interrupt Controller 1 to cascade mode.
Interrupt Controller 2 - ICW1 (I/O /PCI)	A0h	3-LE 4-BE	0	x	Set Interrupt Controller 2 to edge triggered mode.
Interrupt Controller 2 - ICW1 (I/O /PCI)	A0h	1-LE 6-BE	0	x	Set Interrupt Controller 2 to cascade mode.
NMI Status and Control (I/O /PCI)	61h	3-LE 4-BE	0	0	IOCHK# NMI enabled.
NMI Status and Control (I/O /PCI)	61h	2-LE 5-BE	0	0	PCI SERR# NMI enabled.
NMI Enable and TOD Address (I/O /PCI)	70h	7-LE 0-BE	0	1	NMI interrupt enabled.
DMA Command (I/O /PCI)	08h, D0h	7-LE 0-BE	0	0	DACK# Assert Level set to low.
DMA Command (I/O /PCI)	08h, D0h	6-LE 1-BE	0	0	DREQ Sense Level set to high.

**C.1.1 Summary of SIO Configuration Registers****Table C-2. Summary of SIO Configuration Registers**

Address	Description	Type	Reset Value	Set To *
8080 0800	Vendor Identification	R/O	86h	
8080 0801	Vendor Identification	R/O	80	
8080 0802	Device Identification	R/O	84	
8080 0803	Device Identification	R/O	04	
8080 0804	Command	R/W	07	0F
8080 0805	Command	R/W	00	00
8080 0806	Device Status	R/W	00	
8080 0807	Device Status	R/W	02	
8080 0808	Revision Identification	R/W	00	
8080 0840	PCI Control	R/W	20	21
8080 0841	PCI Arbiter Control	R/W	00	00
8080 0842	PCI Arbiter Priority Control	R/W	04	04
8080 0843	PCI Arbiter Priority Control Extension	R/W	00	00
8080 0844	MEMCS# Control	R/W	00	00
8080 0845	MEMCS# Bottom of Hole	R/W	10	10
8080 0846	MEMCS# Top of Hole	R/W	0F	0F
8080 0847	MEMCS# Top of Memory	R/W	00	00
8080 0848	ISA Address Decoder Control	R/W	01	F1
8080 0849	ISA Address Decoder ROM Block	R/W	00	00
8080 084A	ISA Address Bottom of Hole	R/W	10	10
8080 084B	ISA Address Top of Hole	R/W	0F	0F
8080 084C	ISA Controller Recovery Timer	R/W	56	56
8080 084D	ISA Clock Divisor	R/W	40	10
8080 084E	Utility Bus Chip Select A	R/W	07	07
8080 084F	Utility Bus Chip Select B	R/W	4F	FF
8080 0854	MEMCS# Attribute Register #1	R/W	00	
8080 0855	MEMCS# Attribute Register #2	R/W	00	
8080 0856	MEMCS# Attribute Register #3	R/W	00	
8080 0857	Scatter/Gather Relocation Base	R/W	04	
8080 0860	PIRQ Route Control 0	R/W	80	0F
8080 0861	PIRQ Route Control 1	R/W	80	0F
8080 0862	PIRQ Route Control 2	R/W	80	80
8080 0863	PIRQ Route Control 3 (unused)	R/W	80	80
8080 0880	BIOS Timer Base Address	R/W	78	
8080 0881	BIOS Timer Base Address	R/W	00	

**Note:** \* If the entry in this column is blank, then the boot firmware does not write to this register.



## **C.2 Example Planar Combined Register Listing**

### **C.2.1 Direct Access Registers**

Table C-3 contains a summary listing of the registers that are physically located on the example planar. These registers are generally accessed using single CPU transfers. There is an additional set of registers (see Table 9-2) located in the 660 bridge, which are accessed using pairs of CPU transfers.

Table C-3. Combined Register Listing

ISA Port	Contiguous Mode Addr	Non-Contig Mode Addr	Description	R/W	Set To (6)	Loc (4)	No te
0000	8000 0000	8000 0000	DMA1 CH0 Base and Current Addr	R/W	—	SIO	
0001	8000 0001	8000 0001	DMA1 CH0 Base and Current Cnt	R/W	—	SIO	
0002	8000 0002	8000 0002	DMA1 CH1 Base and Current Addr	R/W	—	SIO	
0003	8000 0003	8000 0003	DMA1 CH0 Base and Current Cnt	R/W	—	SIO	
0004	8000 0004	8000 0004	DMA1 CH2 Base and Current Addr	R/W	—	SIO	
0005	8000 0005	8000 0005	DMA1 CH2 Base and Current Cnt	R/W	—	SIO	
0006	8000 0006	8000 0006	DMA1 CH3 Base and Current Addr	R/W	—	SIO	
0007	8000 0007	8000 0007	DMA1 CH3 Base and Current Cnt	R/W	—	SIO	
0008	8000 0008	8000 0008	DMA1 Status(R) Command(W)	R/W	—	SIO	
0009	8000 0009	8000 0009	DMA1 Soft Request	W	—	SIO	
000A	8000 000A	8000 000A	DMA1 Write Single Mask Bit	W	—	SIO	
000B	8000 000B	8000 000B	DMA1 Write Mode	W	—	SIO	
000C	8000 000C	8000 000C	DMA1 Clear Byte Pointer	W	—	SIO	
000D	8000 000D	8000 000D	DMA1 Master Clear	W	—	SIO	
000E	8000 000E	8000 000E	DMA1 Clear Mask	W	—	SIO	
000F	8000 000F	8000 000F	DMA1 R/W All Mask Register Bits	R/W	—	SIO	
0020	8000 0020	8000 1000	INT1 Control	R/W	—	SIO	
0021	8000 0021	8000 1001	INT1 Mask	R/W	—	SIO	
0040	8000 0040	8000 2000	Timer Counter 1 - Counter 0 Cnt	R/W	—	SIO	
0041	8000 0041	8000 2001	Timer Counter 1 - Counter 1 Cnt	R/W	—	SIO	
0042	8000 0042	8000 2002	Timer Counter 1 - Counter 2 Cnt	R/W	—	SIO	
0043	8000 0043	8000 2003	Timer Counter 1 Command Mode	W	—	SIO	
0060	8000 0060	8000 3000	Reset X-Bus (mse) IRQ12 and Kbd	R	—	SIO	
0061	8000 0061	8000 3001	NMI Status and Control	R/W	—	SIO	
0062	8000 0062	8000 3002	Reserved for Keyboard/Mouse	R/W	—	KBD	(3)
0064	8000 0064	8000 3004	Keyboard/Mouse	R	—	KBD	(3)
0066	8000 0066	8000 3006	Reserved for Keyboard/Mouse	R/W	—	KBD	(3)
0070	8000 0070	8000 3010	TOD Addr and NMI Enable	W	—	SIO	
0071	8000 0071	8000 3011	TOD Read/Write	R/W	—	RTC	(3)
0074	8000 0074	8000 3014	NV RAM Addr Strobe 0	W	—	NVR	(3)
0075	8000 0075	8000 3015	NV RAM Addr Strobe 1	W	—	NVR	(3)
0077	8000 0077	8000 3017	NV RAM Data Port	R/W	—	NVR	(3)
0078	8000 0078	8000 3018	BIOS Timer	R/W	—	SIO	
0079	8000 0079	8000 3019	BIOS Timer	R/W	—	SIO	
007A	8000 007A	8000 301A	BIOS Timer	R/W	—	SIO	
007B	8000 007B	8000 301B	BIOS Timer	R/W	—	SIO	
0080	8000 0080	8000 4000	DMA Page Register Reserved	R/W	—	SIO	
0081	8000 0081	8000 4001	DMA Channel 2 Page Register	R/W	—	SIO	
0082	8000 0082	8000 4002	DMA Channel 3 Page Register	R/W	—	SIO	
0083	8000 0083	8000 4003	DMA Channel 1 Page Register	R/W	—	SIO	
0084	8000 0084	8000 4004	DMA Page Register Reserved	R/W	—	SIO	
0085	8000 0085	8000 4005	DMA Page Register Reserved	R/W	—	SIO	
0086	8000 0086	8000 4006	DMA Page Register Reserved	R/W	—	SIO	
0087	8000 0087	8000 4007	DMA Channel 0 Page Register	R/W	—	SIO	
0088	8000 0088	8000 4008	DMA Page Register Reserved	R/W	—	SIO	

Table C-3. Combined Register Listing (Continued)

ISA Port	Contiguous Mode Addr	Non-Contig Mode Addr	Description	R/W	Set To (6)	Loc (4)	Note
0089	8000 0089	8000 4009	DMA Channel 6 Page Register	R/W	—	SIO	
008A	8000 008A	8000 400A	DMA Channel 7 Page Register	R/W	—	SIO	
008B	8000 008B	8000 400B	DMA Channel 5 Page Register	R/W	—	SIO	
008C	8000 008C	8000 400C	DMA Page Register Reserved	R/W	—	SIO	
008D	8000 008D	8000 400D	DMA Page Register Reserved	R/W	—	SIO	
008E	8000 008E	8000 400E	DMA Page Register Reserved	R/W	—	SIO	
008F	8000 008F	8000 400F	DMA Low Page Register Refresh	R/W	—	SIO	
0090	8000 0090	8000 4010	DMA Page Register Reserved	R/W	—	SIO	
0092	8000 0092	8000 4012	Special Port 92 Register	R/W	—	660	
0094	8000 0094	8000 4014	DMA Page Register Reserved	R/W	—	SIO	(2)
0095	8000 0095	8000 4015	DMA Page Register Reserved	R/W	—	SIO	
0096	8000 0096	8000 4016	DMA Page Register Reserved	R/W	—	SIO	
0098	8000 0098	8000 4018	DMA Page Register Reserved	R/W	—	SIO	
009C	8000 009C	8000 401C	DMA Page Register Reserved	R/W	—	SIO	
009D	8000 009D	8000 401D	DMA Page Register Reserved	R/W	—	SIO	
009E	8000 009E	8000 401E	DMA Page Register Reserved	R/W	—	SIO	
009F	8000 009F	8000 401F	DMA Low Page Register Refresh	R/W	—	SIO	
00A0	8000 00A0	8000 5000	INT2 Control Register	R/W	—	SIO	
00A1	8000 00A1	8000 5001	INT2 Mask Register	R/W	—	SIO	
00C0	8000 00C0	8000 6000	DMA2 CH0 Base and Current Addr	R/W	—	SIO	
00C2	8000 00C2	8000 6002	DMA2 CH0 Base and Current Cnt	R/W	—	SIO	
00C4	8000 00C4	8000 6004	DMA2 CH1 Base and Current Addr	R/W	—	SIO	
00C6	8000 00C6	8000 6006	DMA2 CH1 Base and Current Cnt	R/W	—	SIO	
00C8	8000 00C8	8000 6008	DMA2 CH2 Base and Current Addr	R/W	—	SIO	
00CA	8000 00CA	8000 600A	DMA2 CH2 Base and Current Cnt	R/W	—	SIO	
00CC	8000 00CC	8000 600C	DMA2 CH3 Base and Current Addr	R/W	—	SIO	
00CE	8000 00CE	8000 600E	DMA2 CH3 Base and Current Cnt	R/W	—	SIO	
00D0	8000 00D0	8000 6010	DMA2 Status(R) Command(W)	R/W	—	SIO	
00D2	8000 00D2	8000 6012	DMA2 Soft Request	W	—	SIO	
00D4	8000 00D4	8000 6014	DMA2 Write Single Mask Bit	W	—	SIO	
00D6	8000 00D6	8000 6016	DMA2 Write Mode	W	—	SIO	
00D8	8000 00D8	8000 6018	DMA2 Clear Byte Pointer	W	—	SIO	
00DA	8000 00DA	8000 601A	DMA2 Master Clear	W	—	SIO	
00DC	8000 00DC	8000 601C	DMA2 Clear Mask	W	—	SIO	
00DE	8000 00DE	8000 601E	DMA2 R/W All Mask Register Bits	R/W	—	SIO	
00F0	8000 00F0	8000 7010	Coprocessor Error Reg - Reserved	R/W	—	SIO	
040B	8000 040B	8002 0006	DMA1 Extended Mode	W	—	SIO	
0410	8000 0410	8002 0010	CH0 Scatter/Gather Command	W	—	SIO	
0411	8000 0411	8002 0011	CH1 Scatter/Gather Command	W	—	SIO	
0412	8000 0412	8002 0012	CH2 Scatter/Gather Command	W	—	SIO	
0413	8000 0413	8002 0013	CH3 Scatter/Gather Command	W	—	SIO	
0415	8000 0415	8002 0015	CH5 Scatter/Gather Command	W	—	SIO	
0416	8000 0416	8002 0016	CH6 Scatter/Gather Command	W	—	SIO	
0417	8000 0417	8002 0017	CH7 Scatter/Gather Command	W	—	SIO	
0418	8000 0418	8002 0018	CH0 Scatter/Gather Status	R	—	SIO	

Table C-3. Combined Register Listing (Continued)

ISA Port	Contiguous Mode Addr	Non-Contig Mode Addr	Description	R/W	Set To (6)	Loc (4)	No te
0419	8000 0419	8002 0019	CH1 Scatter/Gather Status	R	—	SIO	
041A	8000 041A	8002 001A	CH2 Scatter/Gather Status	R	—	SIO	
041B	8000 041B	8002 001B	CH3 Scatter/Gather Status	R	—	SIO	
041D	8000 041D	8002 001D	CH5 Scatter/Gather Status	R	—	SIO	
041E	8000 041E	8002 001E	CH6 Scatter/Gather Status	R	—	SIO	
041F	8000 041F	8002 001F	CH7 Scatter/Gather Status	R	—	SIO	
0420	8000 0420	8002 1000	CH0 Scatter/Gather Pointer	R/W	—	SIO	
0421	8000 0421	8002 1001	CH0 Scatter/Gather Pointer	R/W	—	SIO	
0422	8000 0422	8002 1002	CH0 Scatter/Gather Pointer	R/W	—	SIO	
0423	8000 0423	8002 1003	CH0 Scatter/Gather Pointer	R/W	—	SIO	
0424	8000 0424	8002 1004	CH1 Scatter/Gather Pointer	R/W	—	SIO	
0425	8000 0425	8002 1005	CH1 Scatter/Gather Pointer	R/W	—	SIO	
0426	8000 0426	8002 1006	CH1 Scatter/Gather Pointer	R/W	—	SIO	
0427	8000 0427	8002 1007	CH1 Scatter/Gather Pointer	R/W	—	SIO	
0428	8000 0428	8002 1008	CH2 Scatter/Gather Pointer	R/W	—	SIO	
0429	8000 0429	8002 1009	CH2 Scatter/Gather Pointer	R/W	—	SIO	
042A	8000 042A	8002 100A	CH2 Scatter/Gather Pointer	R/W	—	SIO	
042B	8000 042B	8002 100B	CH2 Scatter/Gather Pointer	R/W	—	SIO	
042C	8000 042C	8002 100C	CH3 Scatter/Gather Pointer	R/W	—	SIO	
042D	8000 042D	8002 100D	CH3 Scatter/Gather Pointer	R/W	—	SIO	
042E	8000 042E	8002 100E	CH3 Scatter/Gather Pointer	R/W	—	SIO	
042F	8000 042F	8002 100F	CH3 Scatter/Gather Pointer	R/W	—	SIO	
0434	8000 0434	8002 1014	CH5 Scatter/Gather Pointer	R/W	—	SIO	
0435	8000 0435	8002 1015	CH5 Scatter/Gather Pointer	R/W	—	SIO	
0436	8000 0436	8002 1016	CH5 Scatter/Gather Pointer	R/W	—	SIO	
0437	8000 0437	8002 1017	CH5 Scatter/Gather Pointer	R/W	—	SIO	
0438	8000 0438	8002 1018	CH6 Scatter/Gather Pointer	R/W	—	SIO	
0439	8000 0439	8002 1019	CH6 Scatter/Gather Pointer	R/W	—	SIO	
043A	8000 043A	8002 101A	CH6 Scatter/Gather Pointer	R/W	—	SIO	
043B	8000 043B	8002 101B	CH6 Scatter/Gather Pointer	R/W	—	SIO	
043C	8000 043C	8002 101C	CH7 Scatter/Gather Pointer	R/W	—	SIO	
043D	8000 043D	8002 101D	CH7 Scatter/Gather Pointer	R/W	—	SIO	
043E	8000 043E	8002 101E	CH7 Scatter/Gather Pointer	R/W	—	SIO	
043F	8000 043F	8002 101F	CH7 Scatter/Gather Pointer	R/W	—	SIO	
0481	8000 0481	8002 4001	DMA CH2 High Page	R/W	—	SIO	
0482	8000 0482	8002 4002	DMA CH3 High Page	R/W	—	SIO	
0483	8000 0483	8002 4003	DMA CH1 High Page	R/W	—	SIO	
0487	8000 0487	8002 4007	DMA CH0 High Page	R/W	—	SIO	
0489	8000 0489	8002 4009	DMA CH6 High Page	R/W	—	SIO	
048A	8000 048A	8002 400A	DMA CH7 High Page	R/W	—	SIO	
048B	8000 048B	8002 400B	DMA CH5 High Page	R/W	—	SIO	
04D0	8000 04D0	8002 6010	Interrupt Control 1	R/W	—	SIO	
04D1	8000 04D1	8002 6011	Interrupt Control 2	R/W	—	SIO	
04D6	8000 04D6	8002 6016	DMA2 Extended Mode	W	—	SIO	
0808	8000 0808	8004 0008	HDD Light	R/W	—	EPLD	

Table C-3. Combined Register Listing (Continued)

ISA Port	Contiguous Mode Addr	Non-Contig Mode Addr	Description	R/W	Set To (6)	Loc (4)	Note
080C	8000 080C	8004 000C	Equipment Present	R	—	logic	(3)
080D	8000 080D	8004 000D	L2 Cache Status Reg	R	—	logic	(3)
0814	8000 0814	8004 0014	L2 Flush	W	—	660	
081C	8000 081C	8004 001C	System Control 81C	R/W	(7)	660	
0821	8000 0821	8004 1001	Memory Controller Misc	R/W	—	660	
082A	8000 082A	8004 100A	Power Mgmt Control Reg1	R/W	—	EPLD	(5)
082B	8000 082B	8004 100B	Power Mgmt Control Reg2	R/W	—	EPLD	(5)
0840	8000 0840	8004 2000	Memory Parity Error Status	R	—	660	
0842	8000 0842	8004 2002	L2 Error Status	R	—	660	
0843	8000 0843	8004 2003	L2 Parity Read & Clear	R	—	660	
0844	8000 0844	8004 2004	Unsupported Transfer Type Error	R	—	660	
0850	8000 0850	8004 2010	I/O Map Type	W	—	660	
0852	8000 0852	8004 2012	Board ID	R	—	logic	(3)
0860	8000 0860	8004 3000	Freeze Clock Reg Low	R/W	—	EPLD	(3)
0862	8000 0862	8004 3002	Freeze Clock Reg High	R/W	—	EPLD	(3)
0880	8000 0880	8004 4000	SIMM Presence Detect Slot 1/2	R	—	logic	(3)
0881	8000 0881	8004 4001	SIMM Presence Detect Slot 3/4	R	—	logic	(3)
	8000 0CF8		PCI/BCR Configuration Address	R/W	—	660	
	8000 0CFC		PCI/BCR Configuration Data	R/W	—	660	
	8080 08xx		PCI Type 0 Configuration Addr	R/W	—	660	
	8080 0800		Vendor Identification	R	—	SIO	
	8080 0801		Vendor Identification	R	—	SIO	
	8080 0802		Device Identification	R	—	SIO	
	8080 0803		Device Identification	R	—	SIO	
	8080 0804		Command	R/W	0F	SIO	
	8080 0805		Command	R/W	00	SIO	
	8080 0806		Device Status	R/W	—	SIO	
	8080 0807		Device Status	R/W	—	SIO	
	8080 0808		Revision Identification	R/W	—	SIO	
	8080 0840		PCI Control	R/W	21	SIO	
	8080 0841		PCI Arbiter Control	R/W	00	SIO	
	8080 0842		PCI Arbiter Priority Control	R/W	04	SIO	
	8080 0843		PCI Arbiter Priority Control Extension	R/W	00	SIO	
	8080 0844		MEMCS# Control	R/W	00	SIO	
	8080 0845		MEMCS# Bottom of Hole	R/W	10	SIO	
	8080 0846		MEMCS# Top of Hole	R/W	0F	SIO	
	8080 0847		MEMCS# Top of Memory	R/W	00	SIO	
	8080 0848		ISA Address Decoder Control	R/W	F1	SIO	
	8080 0849		ISA Address Decoder ROM Block	R/W	00	SIO	
	8080 084A		ISA Address Bottom of Hole	R/W	10	SIO	
	8080 084B		ISA Address Top of Hole	R/W	0F	SIO	
	8080 084C		ISA Controller Recovery Timer	R/W	56	SIO	
	8080 084D		ISA Clock Divisor	R/W	10	SIO	
	8080 084E		Utility Bus Chip Select A	R/W	07	SIO	
	8080 084F		Utility Bus Chip Select B	R/W	FF	SIO	

Table C-3. Combined Register Listing (Continued)

ISA Port	Contiguous Mode Addr	Non-Contig Mode Addr	Description	R/W	Set To (6)	Loc (4)	No te
	8080 0854		MEMCS# Attribute Register #1	R/W	—	SIO	
	8080 0855		MEMCS# Attribute Register #2	R/W	—	SIO	
	8080 0856		MEMCS# Attribute Register #3	R/W	—	SIO	
	8080 0857		Scatter/Gather Relocation Base	R/W	—	SIO	
	8080 0860		PIRQ Route Control 0	R/W	0F	SIO	
	8080 0861		PIRQ Route Control 1	R/W	0F	SIO	
	8080 0862		PIRQ Route Control 2	R/W	80	SIO	
	8080 0863		PIRQ Route Control 3 (unused)	R/W	80	SIO	
	8080 0880		BIOS Timer Base Address	R/W	—	SIO	
	8080 0881		BIOS Timer Base Address	R/W	—	SIO	
	8080 10xx		PCI Type 0 Configuration Addr	R/W	—	660	
	8080 20xx		PCI Type 0 Configuration Addr	R/W	—	660	
	8080 40xx		PCI Type 0 Configuration Addr	R/W	—	660	
	8080 80xx		PCI Type 0 Configuration Addr	R/W	—	660	(5)
	8081 00xx		PCI Type 0 Configuration Addr	R/W	—	660	(5)
	8082 00xx		PCI Type 0 Configuration Addr	R/W	—	660	(5)
	8084 00xx		PCI Type 0 Configuration Addr	R/W	—	660	(5)
	8088 00xx		PCI Type 0 Configuration Addr	R/W	—	660	(5)
	8090 00xx		PCI Type 0 Configuration Addr	R/W	—	660	(5)
	80A0 00xx		PCI Type 0 Configuration Addr	R/W	—	660	(5)
	80C0 00xx		PCI Type 0 Configuration Addr	R/W	—	660	(5)
	BFFF EFF0	BFFF EFF0	System Error Addr	R	—	660	(5)
	BFFF FFF0	BFFF FFF0	Interrupt Vector	R	—	660	
	FFFF FFF0	FFFF FFF0	Flash Write Addr/Data	W	—	660	
	FFFF FFF1	FFFF FFF1	Flash Lock Out	W	—	660	

**Notes:**

1. The first 5 hex digits in the contiguous and non-contiguous mode columns represent the memory page number for which the protection attributes may be set in contiguous I/O mode. That is, devices having the same first five digits in this column will have the same attributes in the memory page table.
2. Port 94 may be used by certain video controllers (e.g. Weitek™ 9100). The SIO chip positively decodes this port. Therefore bus contention may arise when both devices claim the PCI cycle to this port address. Bus contention results in invalid data and possibly harm to the hardware.
3. The control signals for these ports are partially decoded by the SIO. The System I/O EPLD completes the decodes, and issues control signals to the registers, which are usually X-bus buffers.
4. KBD = Keyboard / Mouse Controller  
RTC = Real Time Clock, also known as the TOD (Time Of Day clock)  
NVR = Non-Volatile RAM, in the same package as the RTC  
660 = The 660 Bridge.
5. Not used.
6. In the Set To column, a long dash — means that the initialization firmware does not write to this register. The register is either not used, not written to, or the value of it depends on changing circumstances.  
If the word Memory appears, please refer to the System Memory section of the 660 User's Manual.
7. Set register 81C to C0h if an L2 is installed, else leave at reset value.

**C.3 ISA Bus Register Suggestions**

The following port assignments are designed to be compatible with the example planar firmware and Super I/O type chips. These registers and functions are not implemented on the example planar motherboard.

Table C-4. Compatible ISA Ports (Not on Reference Board)

ISA Port	Contiguous Mode Addr	Non-Contig Mode Addr	Description	Note	R/W
01F0	8000 01F0	8000 F010	IDE Data		R/W
01F1	8000 01F1	8000 F011	IDE Error/Features		R/W
01F2	8000 01F2	8000 F012	IDE Sector Count		R/W
01F3	8000 01F3	8000 F013	IDE Sector Number		R/W
01F4	8000 01F4	8000 F014	IDE Cylinder Low		R/W
01F5	8000 01F5	8000 F015	IDE Cylinder High		R/W
01F6	8000 01F6	8000 F016	IDE Drive Head		R/W
01F7	8000 01F7	8000 F017	IDE Status/Command		R/W
<b>0278</b>	8000 0278	8001 3018	Parallel Port 2	(1)	R/W
0279	8000 0279	8001 3019	Parallel Port 2		R/W
027A	8000 027A	8001 301A	Parallel Port 2		R/W
027B	8000 027B	8001 301B	Parallel Port 2		R/W
027C	8000 027C	8001 301C	Parallel Port 2		R/W
027D	8000 027D	8001 301D	Parallel Port 2		R/W
02F8	8000 02F8	8001 7018	Serial Port 2		R/W
02F9	8000 02F9	8001 7019	Serial Port 2		R/W
02FA	8000 02FA	8001 701A	Serial Port 2		R/W
02FB	8000 02FB	8001 701B	Serial Port 2		R/W
02FC	8000 02FC	8001 701C	Serial Port 2		R/W
02FD	8000 02FD	8001 701D	Serial Port 2		R/W
02FE	8000 02FE	8001 701E	Serial Port 2		R/W
02FF	8000 02FF	8001 701F	Serial Port 2		R/W
0370	8000 0370	8001 B010	Secondary Floppy Digital Output		W
0371	8000 0371	8001 B011	Secondary Floppy Digital Output		W
0372	8000 0372	8001 B012	Secondary Floppy Digital Output		W
0373	8000 0373	8001 B013	Secondary Floppy Digital Output		W
0374	8000 0374	8001 B014	Secondary Floppy Digital Output		W
0375	8000 0375	8001 B015	Secondary Floppy Digital Output		W
0376	8000 0376	8001 B016	Secondary Floppy Digital Output		W
0377	8000 0377	8001 B017	Secondary Floppy Digital Output		W
0376	8000 0376	8001 B016	Secondary IDE Alt Status/Device Ctl		R/W
0377	8000 0377	8001 B017	Secondary IDE drive address		W
0378	8000 0378	8001 B018	Parallel Port 1	(1)	W
0379	8000 0379	8001 B019	Parallel Port 1		W
037A	8000 037A	8001 B01A	Parallel Port 1		W
037B	8000 037B	8001 B01B	Parallel Port 1		W
037C	8000 037C	8001 B01C	Parallel Port 1		W
037D	8000 037D	8001 B01D	Parallel Port 1		W
0398	8000 0398	8001 C018	Super I/O Index Address		R/W
0399	8000 0399	8001 C019	Super I/O Data Address		R/W
03BC	8000 03BC	8001 D01C	Parallel Port 3		R/W
03BD	8000 03BD	8001 D01D	Parallel Port 3		R/W
03BE	8000 03BE	8001 D01E	Parallel Port 3		R/W
03F0	8000 03F0	8001 F010	Primary Floppy Digital Output (Media Sense)		W/O
03F1	8000 03F1	8001 F011	Primary Floppy Digital Output		W/O

Table C-4. Compatible ISA Ports (Not on Reference Board) (Continued)

ISA Port	Contiguous Mode Addr	Non-Contig Mode Addr	Description	Note	R/W
03F2	8000 03F2	8001 F012	Primary Floppy Digital Output		W
03F3	8000 03F3	8001 F013	Primary Floppy Digital Output (Also Media Sense)		W
03F4	8000 03F4	8001 F014	Primary Floppy Digital Output		W
03F5	8000 03F5	8001 F015	Primary Floppy Digital Output		W
03F6	8000 03F6	8001 F016	Primary Floppy Digital Output		W
03F7	8000 03F7	8001 F017	Primary Floppy Digital Output		W
03F6	8000 03F6	8001 F016	Primary IDE Alt Status/Device Ctl		R/W
03F7	8000 03F7	8001 F017	Primary IDE Drive Address		R
03F8	8000 03F8	8001 F018	Serial Port 1		R/W
03FA	8000 03FA	8001 F01A	Serial Port 1		R/W
03FB	8000 03FB	8001 F01B	Serial Port 1		R/W
03FC	8000 03FC	8001 F01C	Serial Port 1		R/W
03FD	8000 03FD	8001 F01D	Serial Port 1		R/W
03FE	8000 03FE	8001 F01E	Serial Port 1		R/W
03FF	8000 03FF	8001 F01F	Serial Port 1		R/W

**Note:**

1. This is a preferred location for this function.



## Appendix D

### Planar Electromechanical

#### D.1 Electrical

##### D.1.1 Power Requirements

This section sets out the power supply requirements for the example planar. They are achievable with low-cost PC power supplies (see Table D-1 for specifications and Table D-2 for approximate power consumption).

**Table D-1. Power Supply Specification**

Output	Tolerance	Maximum Ripple P-P
+5V	+5% -4%	50 mV
+12V	+5% -5%	120 mV
-12V	+10% -9%	120 mV
-5V* (1)	+10% -10%	120 mV
3.3V (2)	+5% -4%	50 mV

**Table D-2. Approximate Power Consumption**

Element	+5V (Amp)	+12V (Amp)	-12V (Amp)	-5V (Amp) (1)	3.3V (Amp) (2)
Base motherboard with 1 8M SIMM running typical code	3.0	0.1	0.02	0	0
8M DRAM standby/refresh (each)	0.18	0	0	0	0
32M DRAM standby/refresh (each)	0.16	0	0	0	0
256K SRAM L2 Cache	1.6	0	0	0	0
Each PCI Slot – allocation (3)	3.0 max	0.3 max	0.06 max	0	3.3 max
Each ISA Slot – allocation (3)(4)	4.5 max	1.5 max	0.3 max	0.2 max	0
Main Power Connector Capacity (at 5.0 amp./pin—20 amp. DC return)	20.0	5.0	5.0	0.5	0

**Notes for Table D-1 and Table D-2:**

1. The -5v is not used on the planar, but it is routed from the power supply connector to the ISA slots.
2. The 3.3v to the PCI bus slots comes from the power supply connector. The 3.3v for the CPU and the 660 bridge is a separate supply, generated from the +5v supply by a linear regulator.
3. These power requirements are allocated by the system designer. The currents specified per slot are also the maximum currents which may be consumed by the bus. Because of this, the total currents for all the cards used for either the PCI slot or the ISA slot must not exceed the amounts listed for those slots.
4. ISA slot information is taken from IEEE P996.

Other requirements are:

1. Overshoot on any supply voltage must be less than 10% of nominal and must decay to within the regulation band within 50 msec.
2. In any failure situation, the power supply must shut down before the +5v output reaches 6.5V to give the motherboard a reasonable chance of surviving; however, damage may occur at any voltage above 5.5V.
3. Power\_Good Signal Requirements
  - The signal must be at a TTL down level when power is applied until >100 msec to 500 msec after the 5V supply has reached its minimum regulation level, and at TTL high level thereafter as long as outputs are within regulation.
  - At turn-off, the Power Good signal must drop to a TTL low level before any output drops below its regulation limits.
  - The driver must be capable of driving 400 microamps or sinking 5 milliamps. The rise time/fall time must be less than 1 usec., 10—90%
4. The +5V rise time (10%-90%) shall be 3 msec to 100 msec with a maximum slope of 0.75 volts/msec for voltages above 1.5 volts for all loadings.
5. All supply voltages shall track within 50 msec of each other measured at the 50% point.

### D.1.2 Onboard 3.3V Regulator

There is a 3.3 volt regulator circuit on the example planar to support the CPU and the 660 Bridge. Note that the 3.3v on the PCI bus slots is sourced via the power supply connector, not the onboard 3.3v regulator.

**Table D-3. Specifications for 3.3V Regulator on the Motherboard**

Specification	Value
Output Voltage	3.3 V $\pm$ 3%
Output Current	0.01 A to 5 A
Input voltage	4.75 V to 5.25 V
Pass element maximum case temperature	110 °C
Tracking	In regulation <1ms after +5 reaches 4.75V.
Overcurrent	No current limit feature

## D.2 Physical Design Rules

These guidelines are given to aid designers with the physical design phase of their 100 MHz PPC 603e MCM planar. The guidelines are not intended to replace good physical design practices for the signal types and frequencies discussed, but to supplement standard practice by pointing out sensitive and critical areas. Some discussion of the IBM implementation of the example planar is also included to establish the context for the wiring guidelines.

The example planar was developed by adapting the Harley Reference Design to meet the needs of the MCM. The physical design rules for the example planar are identical to those for Harley except for any changes which are contained in the following areas.

### D.2.1 Construction

The general construction of the example planar is shown in Figure D-1. It is constructed with two high frequency signal layers in the center, two power planes, and two external general purpose signal layers.

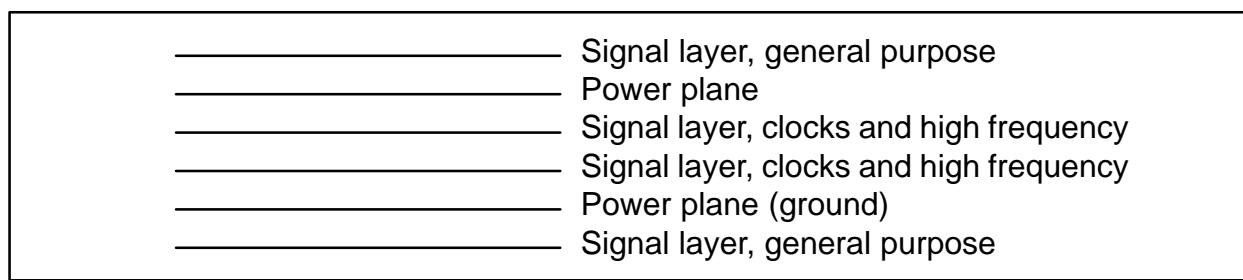


Figure D-1. Signal and Power Layers

A top view of a typical wiring channel, as implemented on the example planar, is shown in Figure D-2 (all dimensions are shown in inches). Minimum trace width is .006" (at 1:1) and minimum space width is .004" (at 1:1).

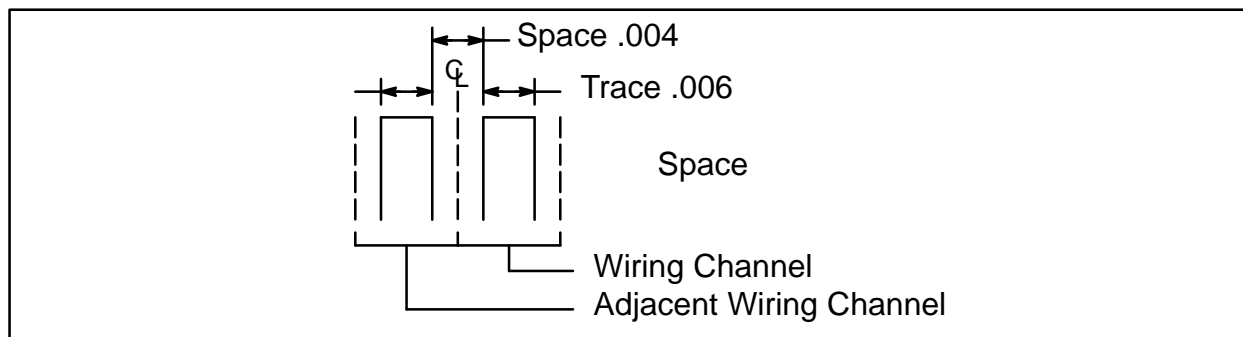





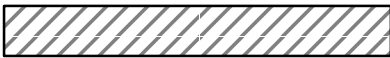





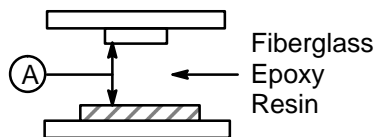


Figure D-2. Typical Wiring Channel Top View

Signal		0.5 oz. Foil
		0.0045 (A)
Power		2 oz. Foil
		0.0062 (A)
Signal		1 oz. Foil
		0.00288 (A)
Signal		1 oz. Foil
		0.0062 (A)
Ground		2 oz. Foil
		0.0045 (A)
Signal		0.5 oz. Foil

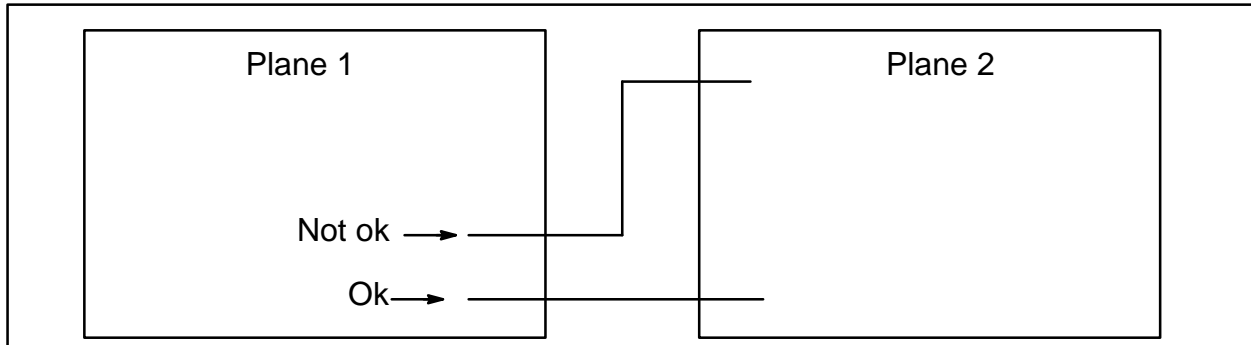
**Notes:**

- (B) All dimensions in inches
- (C) Trace to ground impedance  $70 \Omega \pm 10\%$
- (D) Finished signal trace width: 0.00425
- (E) Overall thickness: 0.060  
after plating: 0.062
- (F) Plating 7070

**Figure D-3. PowerPC 603/604 Board Fabrication**

### **D.2.2 General Wiring Guidelines**

- 1 A power (ground or voltage) plane split occurs where there is a discontinuity in the plane. As shown in Figure D-4, route wires across splits in a power plane in the most perpendicular manner possible. Do not run wires parallel to the split in close proximity to the split.



**Figure D-4. Power Plane Split**

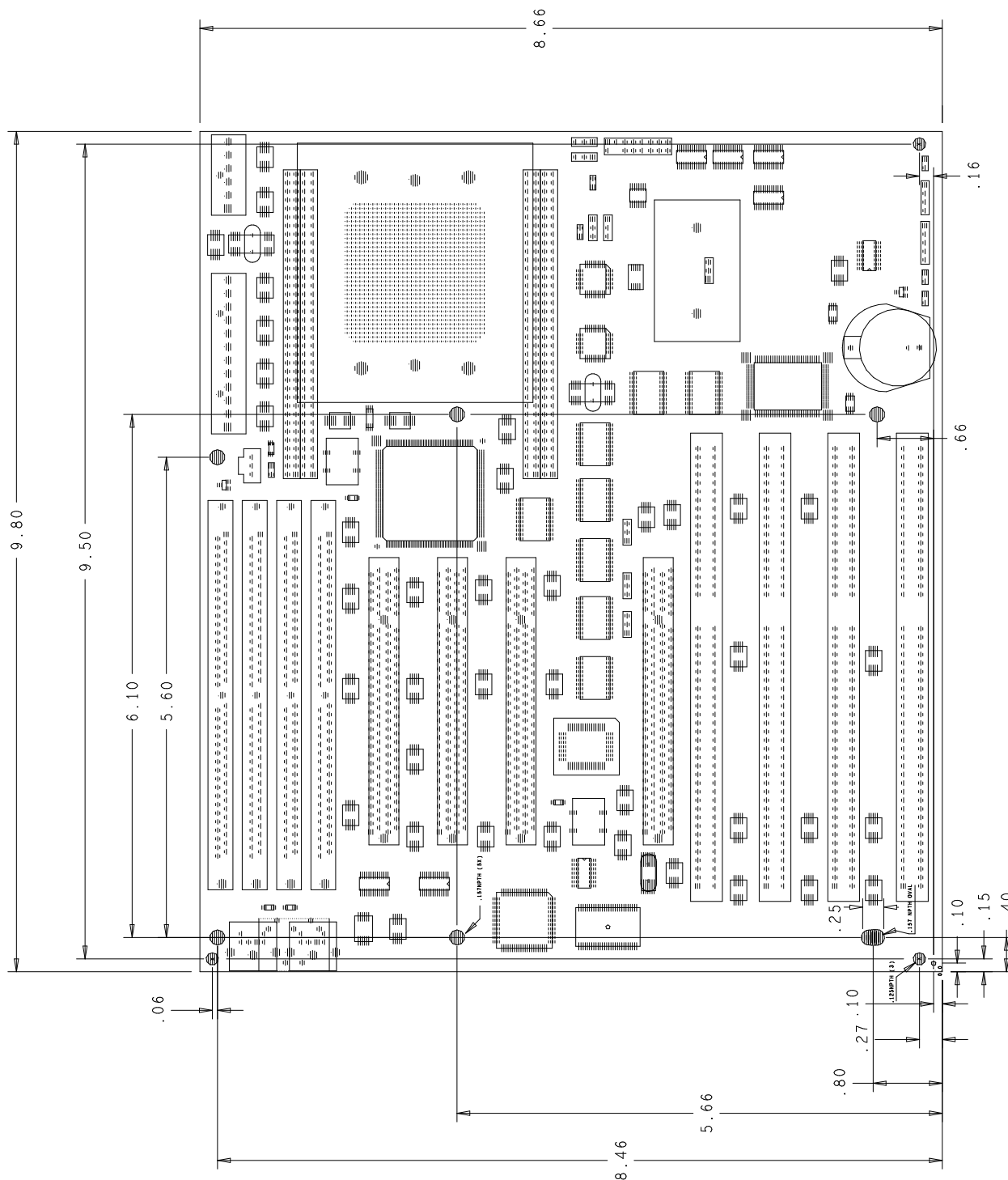
- 2 Minimize the number of wires on the top surface of the board that cross under the clock generator. Do not run any wires close to the crystal connections.
- 3 If a wire is routed near a via which is part of a clock net, there must be at least one vacant wiring channel between the wire and the clock via.
- 4 Several groups of nets require special attention to ensure correct system operation. They are listed below in order of the importance of meeting the design rules that are suggested for that group. For example, ensuring that the clock nets are routed according to the design rules suggested in Section D.3 is more important than ensuring that the PCI bus nets are routed according to the design rules suggested in Section D.4. The ideal system design will meet all of the suggested design rules, but this information is included to guide the designer in case some tradeoffs have to be evaluated.
  - Clock nets.
  - CPU bus nets and timing critical nets.
  - PCI bus nets.
  - Noise sensitive nets.
  - Other nets.

### **D.3 CPU Bus Nets**

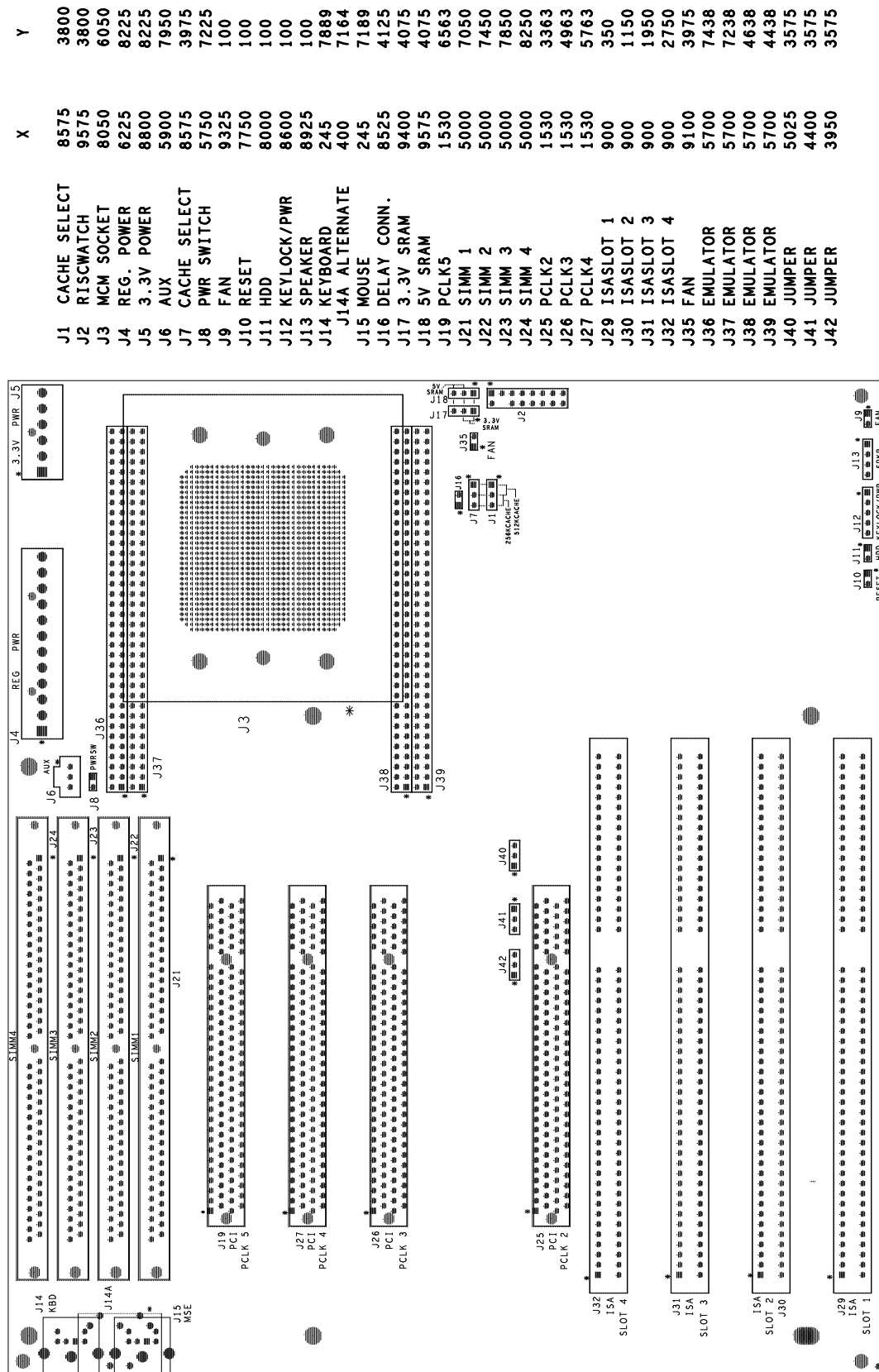
- 1 Make the CPU bus nets either 2-node nets or daisy-chained. Stubs and star fanouts are not allowed.
- 2 Route nets so as to minimize noise reception. Make these nets as short as possible.

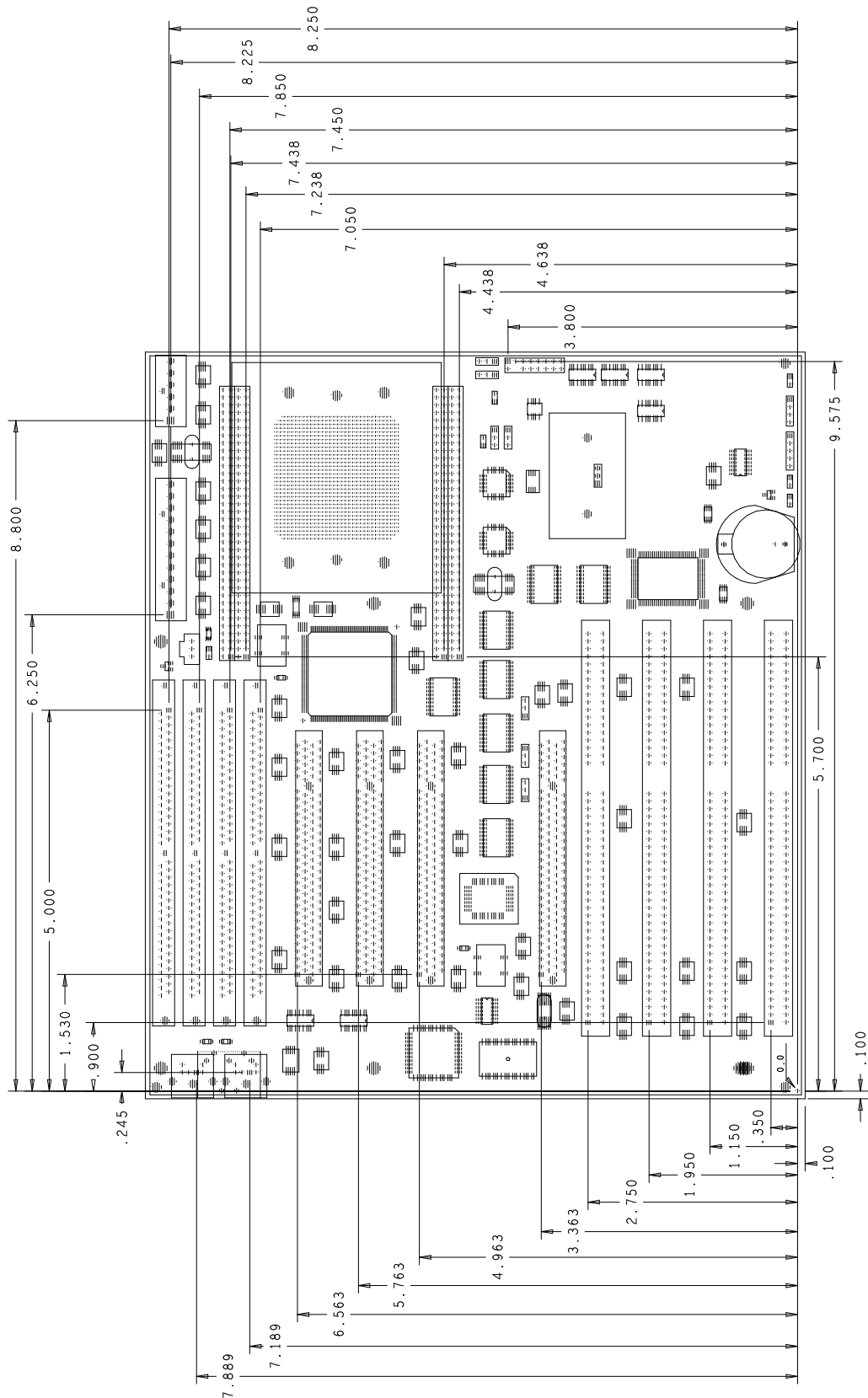
### **D.4 PCI Bus Nets**

- 1 Daisy chain the PCI bus nets. Stubs and star fanouts are not allowed.
- 2 Route these nets so as to minimize noise reception and timing delays. Make these nets as short as possible.

**D.5 Example Planar Mechanical**

## D.5.1 Example Planar Connectors



**D.5.2 Example Planar Connector Locations**



### D.5.3 Keyboard Connector J14

The keyboard connector uses a 6-pin miniature DIN connector (see Figure D-5) Pins are assigned as shown in Table D-4 (as viewed from the back of the machine).

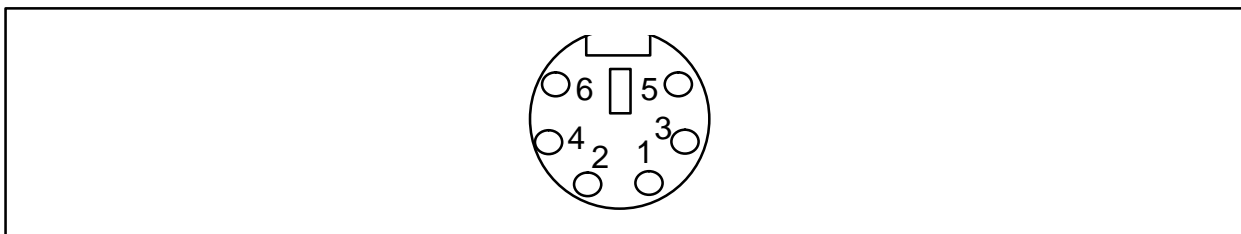


Figure D-5. The Keyboard Connector

Table D-4. Keyboard Connector Pin Assignments

Pin	I/O	Signal Name
1	I/O	DATA
2	NA	n/c
3	NA	GROUND
4	NA	+ 5V DC
5	I/O	CLOCK
6	NA	n/c
M1,2,3	NA	Shield Ground

### D.5.4 Alternate Keyboard Connector J14A

The alternate keyboard connector uses a 5-pin miniature DIN connector (see Figure D-6) Pins are assigned as shown in Table D-5 (as viewed from the back of the machine).

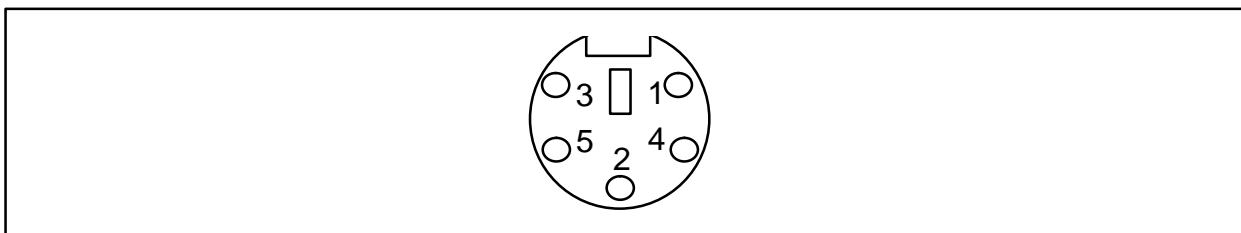


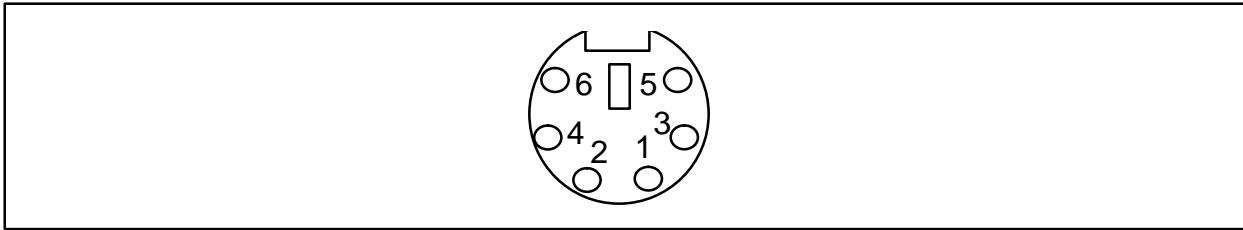
Figure D-6. The Alternate Keyboard Connector

Table D-5. Alternate Keyboard Connector Pin Assignments

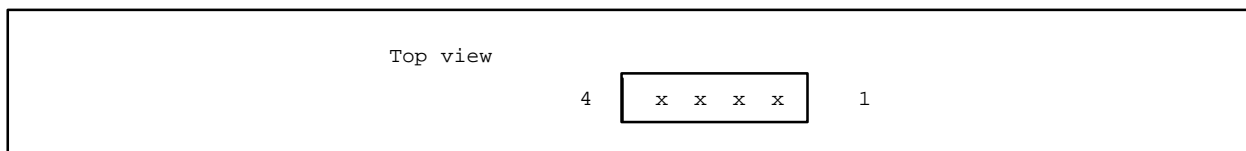
Pin	Signal Name
1	CLOCK
2	DATA
3	n/c
4	GROUND
5	+ 5V DC
M1,2,3	Shield Ground

**D.5.5 Mouse Connector J15**

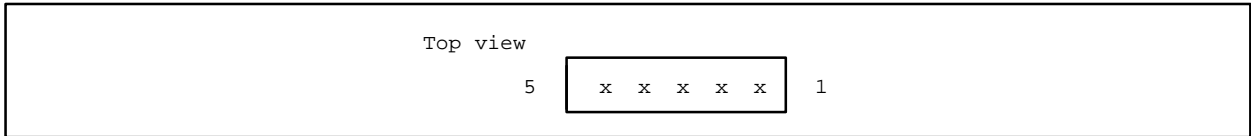
The mouse connector uses a 6-pin miniature DIN connector (see Figure D-7). Pins are assigned as shown in Table D-6 (as viewed from the back of the machine).

**Figure D-7. The Mouse Connector****Table D-6. Mouse Connector Pin Assignments**

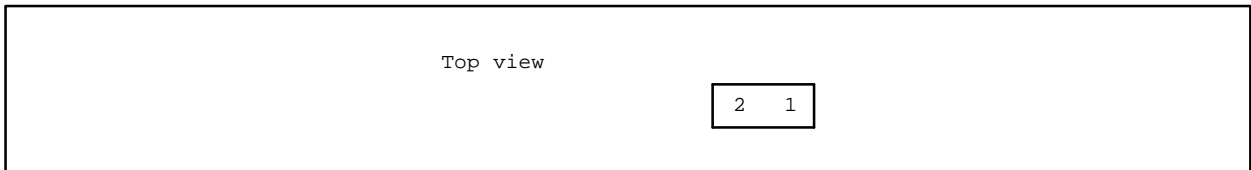
Pin	I/O	Signal Name
1	I/O	DATA
2	NA	n/c
3	NA	GROUND
4	NA	+ 5V DC
5	I/O	CLOCK
6	NA	n/c
M1,2,3	NA	Shield Ground

**D.5.6 Speaker Connector J13****Figure D-8. 1x4 Speaker Connector****Table D-7. Speaker Connector Pin Assignments**

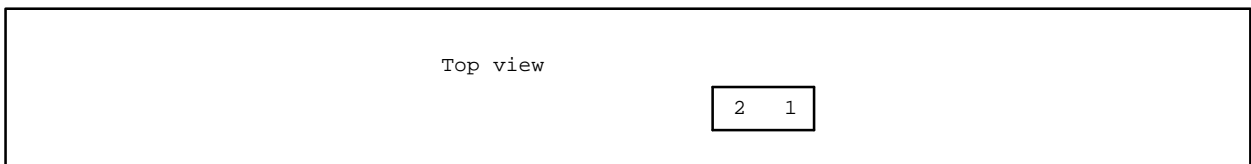
Pin No.	Signal Name
1	MINUS (-) INPUT TO SPEAKER
2	NO CONNECT
3	NO CONNECT
4	GND

**D.5.7 Power Good LED/KEYLOCK# Connector J12****Figure D-9. 1x5 Power Good LED Connector****Table D-8. Power Good LED Connector**

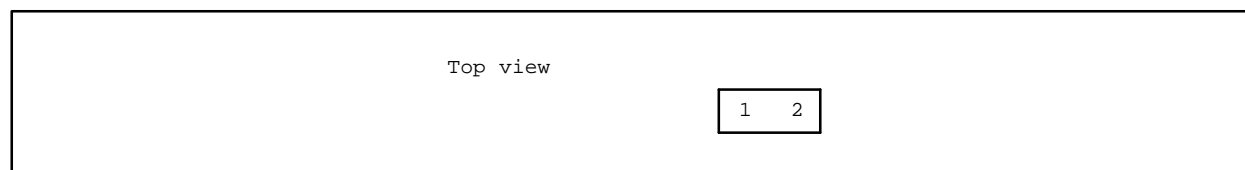
Pin No.	Signal Name
1	LED_POWER_GOOD/RESET#
2	NO CONNECT
3	GND
4	KEYLOCK* (Not Used)
5	GND

**D.5.8 HDD LED Connector J11 (1 x 2 Berg)****Figure D-10. 1x2 HDD LED Connector****Table D-9. HDD LED Connector**

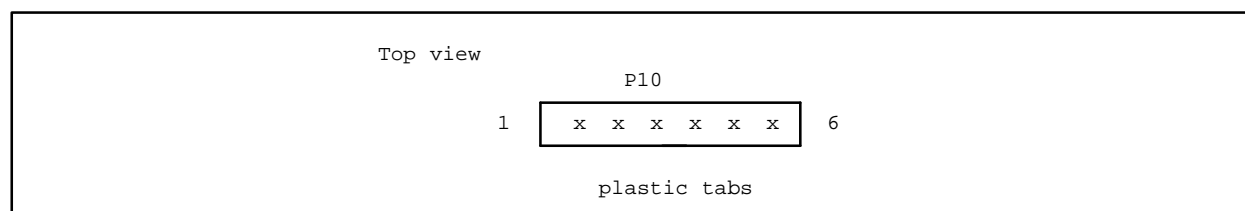
Pin No.	Signal Name
1	+ LED VOLTAGE
2	HDD LED drive signal (low to drive LED)

**D.5.9 Reset Switch Connector J10 (1 x 2 Berg)****Figure D-11. 1x2 Reset Switch Connector****Table D-10. Reset Switch Connector**

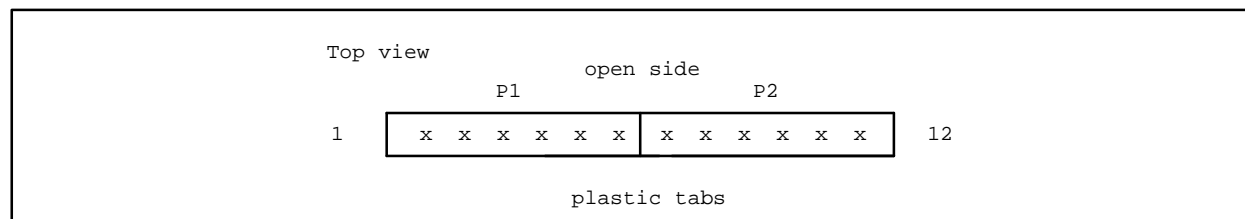
Pin No.	Signal Name
1	RESET N/O CONTACT (close to reset)
2	GROUND

**D.5.10 Fan Connector J9, J35****Figure D-12.1x2 Fan Connector****Table D-11. Fan Connector Pin Assignments**

Pin No.	Signal Name
1	+12 VOLTS
2	GROUND

**D.5.11 3.3V Power Connector J5****Figure D-13.1x6 3.3V Power Connector J5****Table D-12. 3.3V Power Connector J5 Pin Assignments**

Pin No.	Signal Name
1	+3.3 V
2	+3.3 V
3	+3.3 V
4	GROUND
5	GROUND
6	GROUND

**D.5.12 Power Connector J4****Figure D-14.1x12 Power Connector****Table D-13. Power Connector J4 Pin Assignments**

Pin No.	Signal Name
1	POWER GOOD
2	+5 VOLTS
3	+12 VOLTS
4	-12 VOLTS
5, 6, 7, 8	GROUND
9, 10, 11, 12	-5 VOLTS

## D.5.13 AUX5/ON-OFF Connector J6

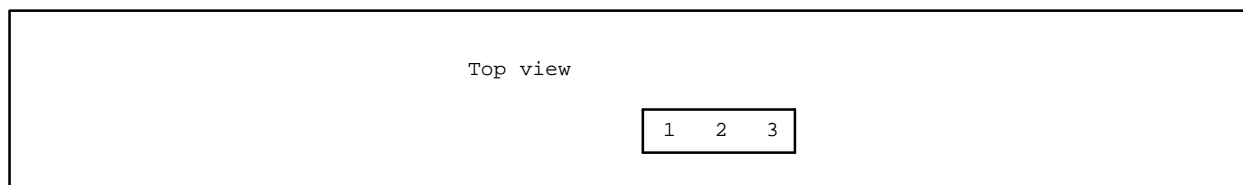


Figure D-15.AUX5/ON-OFF Connector

Table D-14. AUX5/ON-OFF Connector Pin Assignments

Pin No.	Signal Name
1	AUX +5V
2	ON/OFF
3	GROUND

## D.5.14 PCI Connectors J19, J25, J26, and J27

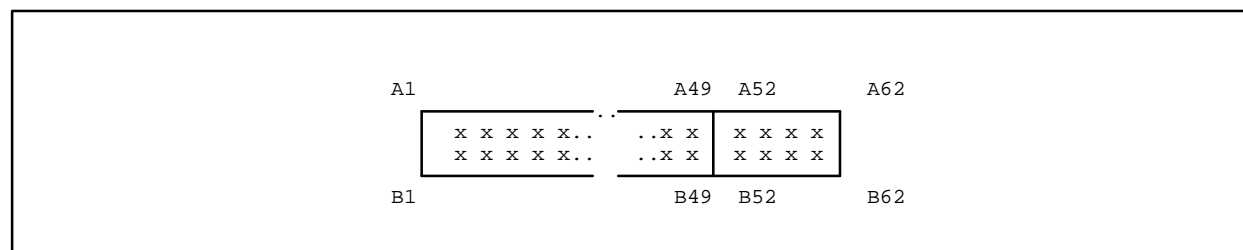


Figure D-16.PCI Connector

Table D-15. PCI Connector Pin Assignments

Pin	Function	Pin	Function
A1	TRST#	B1	–12 VOLTS#
A2	+12 VOLTS	B2	TCK
A3	TMS	B3	GROUND
A4	TDI	B4	TDO
A5	+5 VOLTS	B5	+5 VOLTS
A6	INTA#	B6	+5 VOLTS
A7	INTC#	B7	INTB#
A8	+5 VOLTS	B8	INTD#
A9	RESERVED	B9	PRESENT 1# *
A10	+5 VOLTS	B10	RESERVED
A11	RESERVED	B11	PRESENT 2# *
A12	GROUND	B12	GROUND
A13	GROUND	B13	GROUND
A14	RESERVED	B14	RESERVED
A15	RESET#	B15	GROUND
A16	+5 VOLTS	B16	CLK
A17	GNT#	B17	GROUND
A18	GROUND	B18	REQ#
A19	RESERVED	B19	+5 VOLTS
A20	A/D(30)	B20	A/D(31)

Table D-15. PCI Connector Pin Assignments (Continued)

Pin	Function	Pin	Function
A21	+3.3V	B21	A/D(29)
A22	A/D(28)	B22	GROUND
A23	A/D(26)	B23	A/D(27)
A24	GROUND	B24	A/D(25)
A25	A/D(24)	B25	+3.3V
A26	IDSEL	B26	C/BE#(3)
A27	+3.3V	B27	A/D(23)
A28	A/D(22)	B28	GROUND
A29	A/D(20)	B29	A/D(21)
A30	GROUND	B30	A/D(19)
A31	A/D(18)	B31	+3.3V
A32	A/D(16)	B32	A/D(17)
A33	+3.3V	B33	C/BE#(2)
A34	FRAME#	B34	GROUND
A35	GROUND	B35	IRDY#
A36	TRDY#	B36	+3.3V
A37	GROUND	B37	DEVSEL#
A38	STOP#	B38	GROUND
A39	+3.3V	B39	LOCK#
A40	SDONE	B40	PERR#
A41	SBO#	B41	+3.3V
A42	GROUND	B42	SERR#
A43	PAR	B43	+3.3V
A44	A/D(15)	B44	C/BE#(1)
A45	+3.3V	B45	A/D(14)
A46	A/D(13)	B46	GROUND
A47	A/D(11)	B47	A/D(12)
A48	GROUND	B48	A/D(10)
A49	A/D(9)	B49	GROUND
A50	<Key>	B50	<Key>
A51	<Key>	B51	<Key>
A52	C/BE#(0)	B52	A/D(8)
A53	+3.3V	B53	A/D(7)
A54	A/D(6)	B54	+3.3V
A55	A/D(4)	B55	A/D(5)
A56	GROUND	B56	A/D(3)
A57	A/D(2)	B57	GROUND
A58	A/D(0)	B58	A/D(1)
A59	+5 VOLTS	B59	+5 VOLTS
A60	REQ64#	B60	ACK64#
A61	+5 VOLTS	B61	+5 VOLTS
A62	+5 VOLTS	B62	+5 VOLTS

**Notes:**

\* The two card type bits, B9 and B11, are connected on the riser.

\*\* Int A-D, A6, A7, B7, and B8 are connected together on the riser.

## D.5.15 ISA Connectors J29, J30, J31, J32

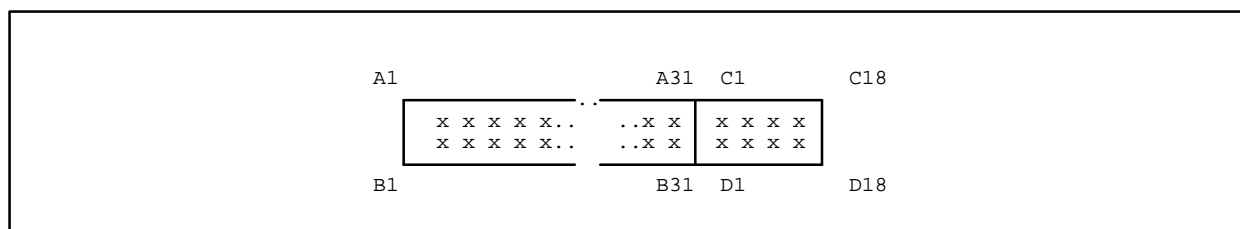


Figure D-17. ISA Connector

Table D-16. ISA Connector Pin Assignments

Pin	Function	Pin	Function
A1	IO CHCK#	B1	GROUND
A2	SD(7)	B2	RESET_DRV
A3	SD(6)	B3	+5 VOLTS
A4	SD(5)	B4	IRQ9
A5	SD(4)	B5	-5 VOLTS
A6	SD(3)	B6	DRQ2
A7	SD(2)	B7	-12 VOLTS
A8	SD(1)	B8	ZERO WS#
A9	SD(0)	B9	+12 VOLTS
A10	IO CHRDY	B10	GROUND
A11	AEN	B11	SMEMW#
A12	SA(19)	B12	SMEMR#
A13	SA(18)	B13	IOW#
A14	SA(17)	B14	IOR#
A15	SA(16)	B15	DACK3#
A16	SA(15)	B16	DRQ3
A17	SA(14)	B17	DACK1#
A18	SA(13)	B18	DRQ1
A19	SA(12)	B19	REFRESH
A20	SA(11)	B20	CLK
A21	SA(10)	B21	IRQ7
A22	SA(9)	B22	IRQ6
A23	SA(8)	B23	IRQ5
A24	SA(7)	B24	IRQ4
A25	SA(6)	B25	IRQ3
A26	SA(5)	B26	DACK2#
A27	SA(4)	B27	T/C
A28	SA(3)	B28	BALE
A29	SA(2)	B29	+5 VOLTS
A30	SA(1)	B30	Oscillator
A31	SA(0)	B31	GROUND
C1	SBHE#	D1	MEMCS16#
C2	LA(23)	D2	IO CS16#
C3	LA(22)	D3	IRQ10
C4	LA(21)	D4	IRQ11
C5	LA(20)	D5	IRQ12

Table D-16. ISA Connector Pin Assignments (Continued)

Pin	Function	Pin	Function
C6	LA(19)	D6	IRQ15
C7	LA(18)	D7	IRQ14
C8	LA(17)	D8	DACK0#
C9	MEMR#	D9	DRQ0
C10	MEMW#	D10	DACK5#
C11	SD(8)	D11	DRQ5
C12	SD(9)	D12	DACK6#
C13	SD(10)	D13	DRQ6
C14	SD(11)	D14	DACK7#
C15	SD(12)	D15	DRQ7
C16	SD(13)	D16	+5 VOLTS
C17	SD(14)	D17	MASTER#
C18	SD(15)	D18	GROUND

## D.5.16 DRAM SIMM Connectors J21, J22, J23, and J24

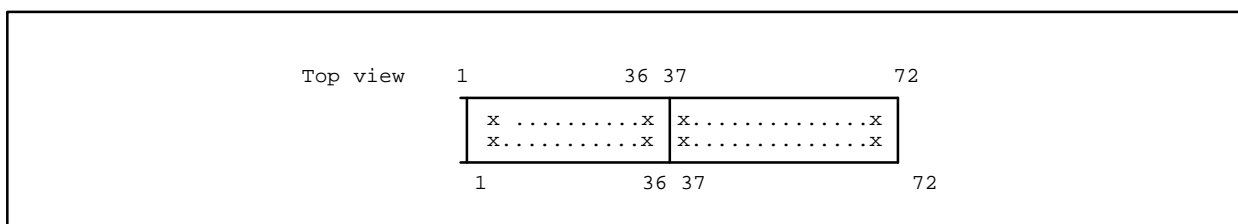


Figure D-18. DRAM SIMM Connector

Table D-17. DRAM SIMM Connector Pin Assignments

SIMM Pin	Motherboard Pin	Function
1	1	GROUND
2	2	DQ0
3	3	DQ18
4	4	DQ1
5	5	DQ19
6	6	DQ2
7	7	DQ20
8	8	DQ3
9	9	DQ21
10	10	+5 V
11	11	CASP
12	12	A0
13	13	A1
14	14	A2
15	15	A3
16	16	A4
17	17	A5
18	18	A6
19	19	A10



Table D-17. DRAM SIMM Connector Pin Assignments (Continued)

SIMM Pin	Motherboard Pin	Function
20	20	DQ4
21	21	DQ22
22	22	DQ5
23	23	DQ23
24	24	DQ6
25	25	DQ24
26	26	DQ7
27	27	DQ25
28	28	A7
29	29	BS0/A11
30	30	+5V
31	31	A8
32	32	A9
33	33	RAS3#
34	34	RAS2#
35	35	DQ26
36	36	DQ8
37	37	DQ17
38	38	DQ35
39	39	GROUND
40	40	CAS0#
41	41	CAS2#
42	42	CAS3#
43	43	CAS1#
44	44	RAS0#
45	45	RAS1#
46	46	BS1
47	47	WE#
48	48	RES1
49	49	DQ9
50	50	DQ27
51	51	DQ10
52	52	DQ28
53	53	DQ11
54	54	DQ29
55	55	DQ12
56	56	DQ30
57	57	DQ13
58	58	DQ31
59	59	+5 V
60	60	DQ32
61	61	DQ14
62	62	DQ33
63	63	DQ15
64	64	DQ34

Table D-17. DRAM SIMM Connector Pin Assignments (Continued)

SIMM Pin	Motherboard Pin	Function
65	65	DQ16
66	66	BS2
67	67	PD1
68	68	PD2
69	69	PD3
70	70	PD4
71	71	BS3
72	72	GROUND

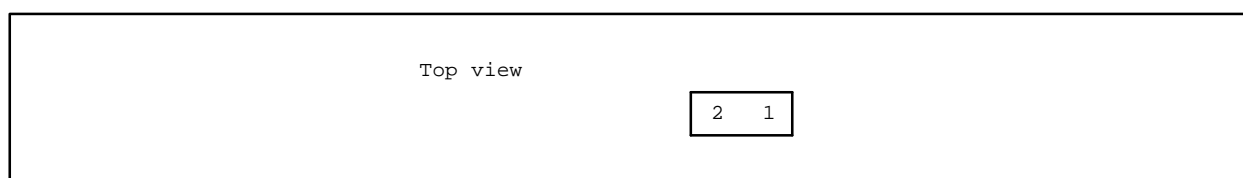
**D.5.17 Power Switch Connector J8**

Figure D-19.1x2 Power Switch Connector

Table D-18. Power Switch Connector Pin Assignments

Pin No.	Signal Name
1	SWITCH_P1
2	SWITCH_GD

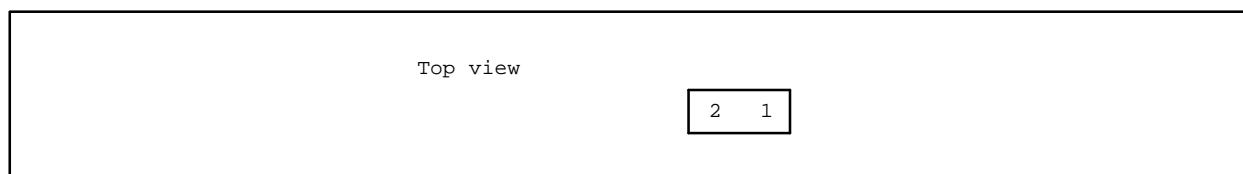
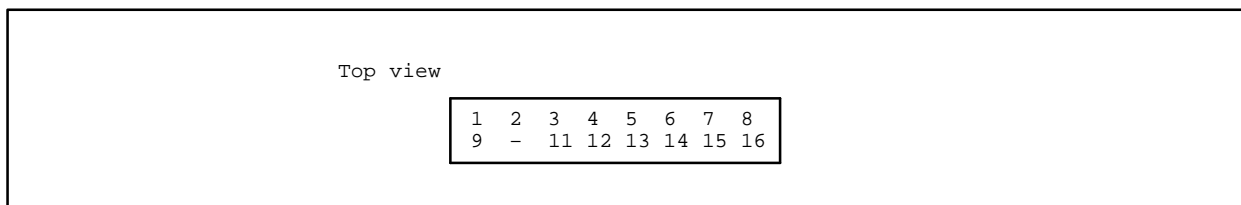
**D.5.18 Power Up Configuration Connector J7**

Figure D-20.1x2 Power Up Configuration Connector

Table D-19. Power Up Configuration Connector Pin Assignments

Pin No.	Signal Name
1	PWR_CFG1
2	PWR_CFG2

**D.5.19 RISCWatch Connector J2****Figure D-21. 2x8 RISCWatch Connector****Table D-20. RISCWatch Connector Pin Assignments**

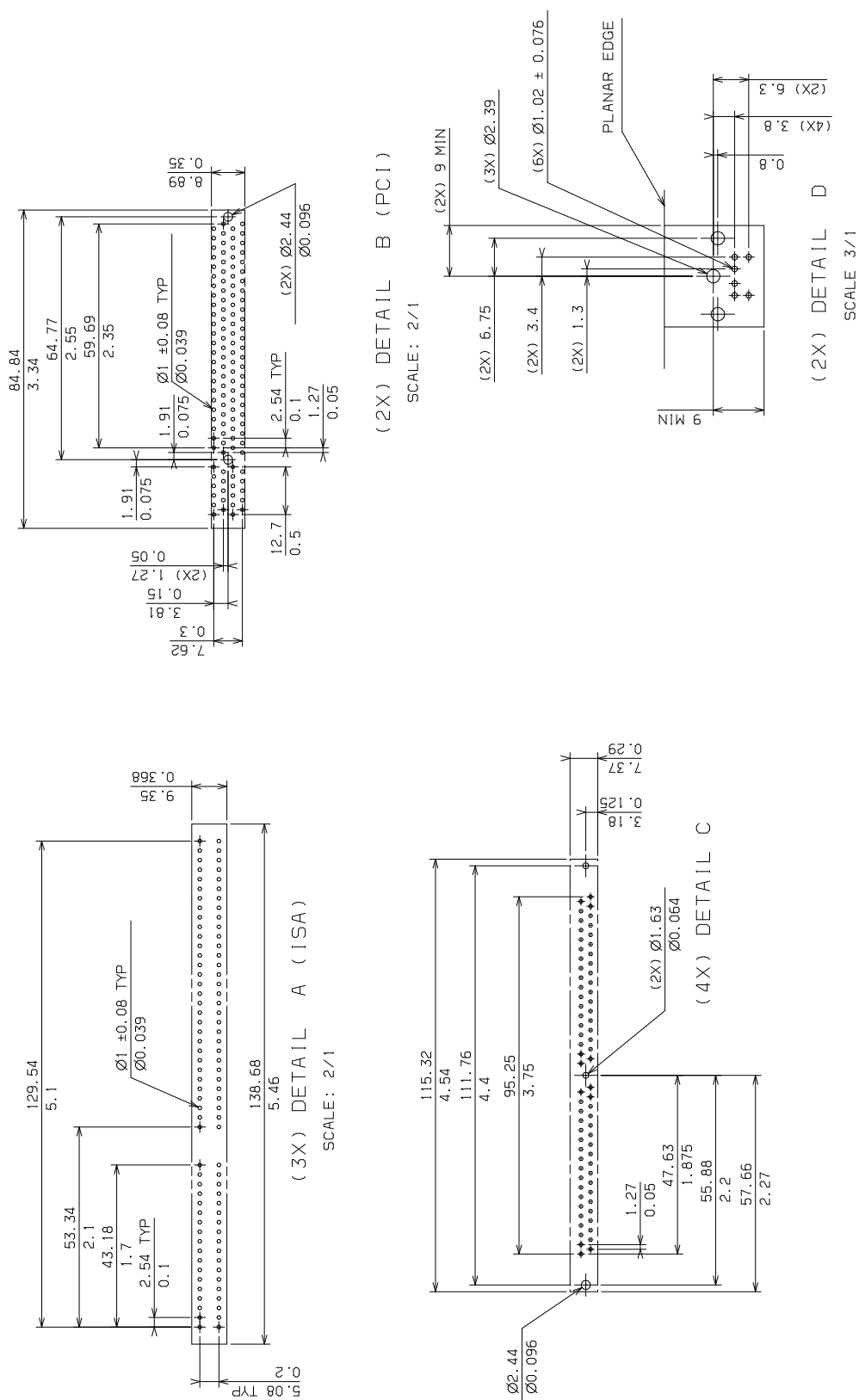
Pin No.	Signal Name
1	CHECK_STOP#
2	HDWR_RESET#
3	RESET_INTERRUPT#
4	CNTL/SCAN_DATA
5	SHIFT_CLK
6	+RUN/-BREAKPOINT
7	SCAN_IN
8	SCAN_OUT
9	GROUND
10	<Key>
11	GROUND
12	RESERVED
13	RESERVED
14	+5 VOLTS
15	OCS_OVERRIDE
16	RESERVED

**D.5.20 Battery Connector BT2**

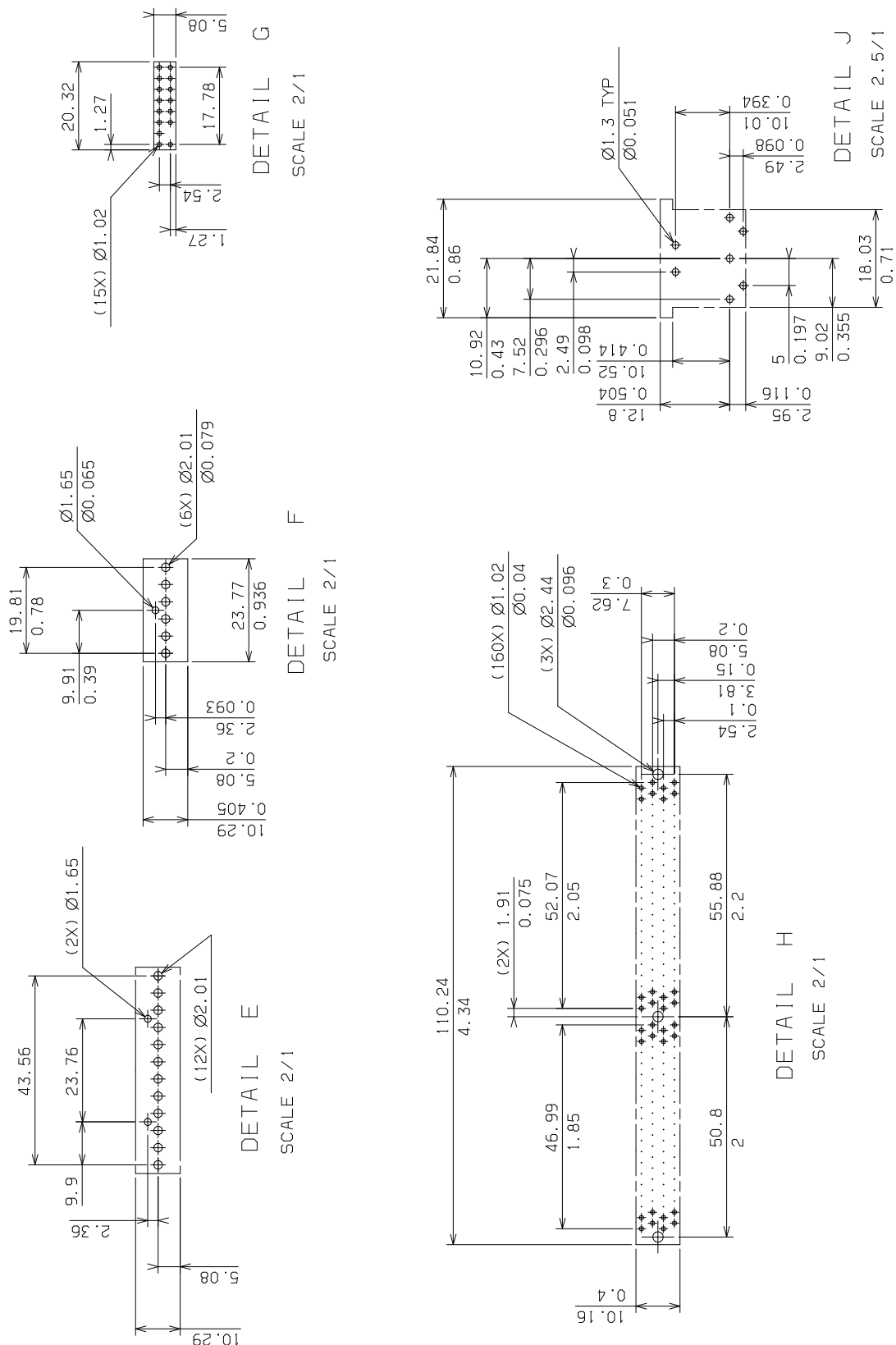
The battery type is CR2032 3 Volt.

Insert the battery with + side up.

### D.5.21 MCM Board Connector Footprint #1



### D.5.22 MCM Board Connector Footprint #2



## D.6 Enclosure

The example planar is mechanically very similar to the Harley board. Enclosures for Harley should work for the example planar, but this has not been verified. The information in this section should be considered as tentative. The experienced designer will verify the appropriateness of an enclosure before specifying it.

The following vendors supply enclosures:

Olson Metal Products Company Inc.  
Attn: Michelle Seay  
1903 N. Austin Street  
Seguin, Texas 78155  
1-800-951-9517 or (210) 379-7000

AT Desktop model number: CC300249  
Medium Tower model number: CC400000

Altex Electronics  
11342 IH-35 North  
San Antonio, Texas  
1-800-531-5367 or FAX (210) 637-3264

Mini Tower model number STC-05  
Medium Tower model number STC-08  
Full Tower model number STC-16

Mega-Tech Marketing Inc.  
3900-D Drossett Dr.  
Austin, Texas

Mini Tower model number A6601  
Full Tower model number A5561

**Note:** IBM makes no recommendations regarding vendors of any components.

In planning the layout of an enclosure, the height of the items in Table D-21 should be considered.

**Table D-21. Height Considerations**

Item	Approximate Height
3.3V regulator heat sink	26.0 mm
SIMMs (seated)	28.4 mm
L2 SRAM SIMM (seated)	28.4 mm
MCM and mounting	

## Appendix E Bill of Materials

### E.1 Planar Bill of Materials

Table E-1. Planar Bill of Materials

No	Part Name	Part Number	Reference Designator	Qty	Manufacturer Name	Manufacturer Part Number	JEDEC TYPE	Part Description
1	10MQ040_SOIC-BASE	87F4917	CR1,CR7	2	SANKEN	SFBP-54YL	D64	40V 1A SCHTK REC
2	29F040ROM_MECH_MECH-BASE	82G6496	X3	1	AMD	AM29F040-120JC	PART	512KX8 FLASH ROM
3	2N7002_SMD-BASE	31F2311	Q1	1	MOTOROLA	2N7002	SOT23	3 PIN FET TRANSISTOR
4	7406_SMD-BASE	17F7776	U3,U28	2	TI	7406	SO14	HEX INVERTER OC
5	74F08_SMD-BASE	61X9236	U1,U29	2	NATL	FO8	SO14	QUAD 2 INPUT & GATE
6	74F11_SMD-BASE	17F7745	U13	1	NATL	F11	SO14	TRI 3 INPUT AND GATE
7	74F125_SMD-BASE	68X2888	U16	1	NATL	74F125SCX	SO14	74F125 BUFFERS 3 STATE
8	74F244_SMD-BASE	6480438	U10,U11,U17-U20,U40	7	PHILIPS	74F244	SO20W	OCTAL BUFF/DRIVER
9	74F245_SMD-BASE	55X8091	U12	1	PHILIPS	74F245	SO20W	OCTAL TRANSCEIVER
10	74F74_SMD-BASE	61X9238	U21	1	NATL	F74	SO14	DUAL"D"FF W/CLEAR & PRESET
11	74HCT14_SMD-BASE	37F9034	U2	1	PHILIPS	74HCT14	SO14	74HCT14 SCHMT TRIGGER INV
12	8042H_SMD-BASE	1054195	U27	1	INTEL	8042H	PLCC44	KEYBOARD CONTROLLER
13	8G4756_SMD-BASE	08G4756	CR6	1	MOTOROLA	MMBD914 LT1	SOT23	70V 200MA SWC DIODE
14	BATTERY_MECH-BASE	15F8409	B1	1	SONY	CR2032	PART	BATTERY 3.3V
15	BERG1X2_DIP-BASE	6181127	J8-J11,J16,J35	6	BERG	69190-502	BERG1X2	1X2 100MIL HEADER VERTICAL
16	BERG1X4_DIP-BASE	6359315	J13	1	BERG	69190-504	BERG1X4	1X4 100MIL HEADER VERTICAL
17	BERG1X5_DIP-BASE	88G4908	J12	1	AMP	104345-3	BERG1X5	BERGSTICK 1X5
18	CAPACITOR-0.001UF,20%	98F1292	C129-C148	20	KYOCERA	08055C102 KAT2A	SMC0805	CAPACITOR

Table E-1. Planar Bill of Materials (Continued)

No	Part Name	Part Number	Reference Designator	Qty	Manufacturer Name	Manufacturer Part Number	JEDEC TYPE	Part Description
19	CAPACITOR-0.01UF,20%	41F0313	C3-C5,C28-C31,C33-C37,C39,C40,C46-C50,C52,C53,C57,C58,C61,C80-C82,C84,C86,C93,C95,C117-C120,C124,C127,C151,C157-C164,C167-C169,C172-C174,C203-C206,C208-C210,C212-C215,C217-C220,C222-C225,C255-C257,C259,C269-C274	81	KYOCERA	0805X103 M2B05	SMC0805	CAPACITOR
20	CAPACITOR-0.1UF,20%	41F0316	C9,C43,C91,C92,C94	5	KYOCERA	0805Y104 Z1B05	SMC0805	CAPACITOR
21	CAPACITOR-10UF,20%	71F7911	C8	1	NEC	TESVEC 1C106M12R	71F7911	CAPACITOR
22	CAPACITOR-2200PF,20%	42G3220	C32,C83,C85,C123,C125	5	MURATA	GRM40X7R 222J050AD	SMC0805	CAPACITOR
23	CAPACITOR-33UF/20%/16V	57G9281	C202,C207,C211,C216,C221	5	KEMET	T491D336 M016AS	CAP33UF3SMT	CAPACITOR
24	CAPACITOR-68PF,20%	62G4724	C87-C90,	4	KYOCERA	08055A680 KAT2	SMC0805	CAPACITOR
25	CAPACITOR-DO_NOT_POP, 20%	-NONE-	C1,C2,C6,C7,C10-C21,C70,C71,C98,C99,C149,C150,C152-C154,C156,C226	27	NONE	NONE	SMC0805	CAPACITOR
26	CHANDRA_SMD-BASE	SKT_100	M1	1	ALTERA	100 QFP SOCKET	100QFP	GTP W/128 MACROCELLS
27	CONNBATT_KELPIE_CONN-BASE	33F8027	M5	1	KELPIE	K17-BH-001	CONN_33F8027	BATTERY HOLDER
28	CONNBERG2X36_DIP-BASE	N_MOLEX	J36-J39	4	MOLEX	15-44-4536	BERG2X36	2X36 MOLEX PCB CONN
29	CONNDIN5-DO_NOT_POP	-NONE-	J14A	1	NONE	NONE	CONN_5DIN	AT KEYBOARD CONNECTOR
30	CONNMDIN6-CONNMDIN6	15F6890	J14,J15	2	AMP	749180-1	CONN_15F6890	6 POS CIRCULAR MINI DIN
31	CONNPOWER2_DIP-BASE	N_BURND	J5	1	BURNDY	GTC6R-1	CONN_1X6	1X6 0 156 CL FRET LCK HDR
32	CONNPOWER_DIP-BASE	55X8085	J4	1	MOLEX	15-48-0212	CONN_55X8085	1X12 0 156 CL FRET LCK HDR
33	CONN_BATTERY-SONY_NO_POP	19G2441	M3	1	SONY		CONN_19G2441	BATTERY HOLDER
34	CONN_BERG1X3_DIP-BASE	1501831	J1,J7,J17,J18,J40-J42	7	BERG	69190-503	BERG1X3	1X3 100MIL HEADER VERTICAL
35	CONN_ISA_DIP-BASE	6137473	J29-J32	4	AMP	645169-4	ISA2X49	2X49 ISA CONNECTOR
36	CONN_PCI_DIP-BASE	72G0316	J19,J25-J27	4	AMP	646255-1	PCI2X605V	2X60 32 BIT PCI CONNECTOR
37	CONN_SIMM72_DIP-BASE	64F5806	J21-J24	4	AMP	822032-4	SIMM72	72 PIN SIMM CONNECTOR
38	CRYSTAL_2PIN-32.768KHZ	03G9527	Y3	1	EPSON	MC405-32.768KHZ-6PF	XTALSMT1	CRYSTAL
39	CRYSTL-DO_NOT_POP	-NONE-	Y2,Y4	2	NONE	NONE	XTALSMT2	CRYSTAL
40	DO-NOT-POP_PLCC20_PLCC20-BASE	-NONE-	S3	1	NONE	NONE	PLCC20	DO-NOT-POP
41	DS1385_SMD-BASE	70G6764	U5	1	DALLAS	DS1385S	SO28W	RTC+ 4KX8SRAM
42	ELCAP-100UF, 10%/16V	75G8253	C110	1	SPRAGUE	293D107X 96R3D2T	SMC2816	CAPACITOR



Table E-1. Planar Bill of Materials (Continued)

No	Part Name	Part Number	Reference Designator	Qty	Manufacturer Name	Manufacturer Part Number	JEDEC TYPE	Part Description
43	ELCAP-33UF, 20%/16V	57G9281	C27,C38,C112-C116,C121,C122,C165,C166,C170,C171,C175-C183,C187-C198,C261-C263	37	KEMET	T491D336 MO16AS	CAP33UF3SMT	CAPACITOR
44	EPM5130FP_SMD-BASE	05H0966	X2	1	ALTERA	EPM5130-QC-100-1	PART	GTP W/128 MACROCELLS
45	ESP_CONN_CONN-BASE	42F6867	J2	1	BERG	79282-516	CONN_42F6867	2X8 PIN HEADER CONN
46	F_BEAD_BEAD-BASE	26F4865	FB1,FB4-FB6	4	TDK	HF50ACB-321611T	SMC1206	FERRITE BEAD
47	IRFZ44_SMD-BASE	03G9500	Q6	1	IR	IRFZ44	TO220X	TRANSISTOR 50V 35A
48	LT1431REG_DIP-BASE	31F2428	U4	1	LINEARTECH	LT1431CS8	SO8	VOLT REGULATOR
49	MTGHOLE-DO_NOT_POP	-NONE-	MH1,MH3-MH5,MH7,MH8	6			157TOOL	
50	OSCLR-14.3181MHZ	87F5263	Y1	1	EPSON	SG-615P-14.31818 MC	OSCSMT4	OSCLR
51	OSCLR-24.0MHZ	87F5265	Y5	1	EPSON	SG-615P-24.0000MC	OSCSMT4	OSCLR
52	PAL_20-SKT_19G5840	19G5840	S1	1	AMP	3-822269-1	PLCC20	SOCKET
53	PAL_MECH_MECH-BASE	05H0958	X5	1	AMD	PALCE16V8 H-5JC-5	PART	PALCE16V8H-5JC-5
54	PART-JUMPER1X2	92F1420	X1,X4	2	AMP	530153-2	PART	JUMPER 1X2
55	PIN1089_DIP-BASE	60G8498	J3	1	IBM	CUSTOM	MCM33X33SKT	1089 PIN CGB SOCKET
56	PLCC32SKT_SMD-BASE	10G7624	M2	1	AMP	821977-1	PLCC32SKT	32 PIN PLCC SOCKET
57	POLYSWITCH_SMD-BASE	34G3113	F1	1	RAYCHEM	SMD100	POLYSWITCH	TAPE RESISTOR
58	POWER1X3_DIP-BASE	65G3724	J6	1	MOLEX	705-43-0037	POWER1X3	AUX POWER CONNECTOR
59	REGHSINK_DIP-BASE	8185352	M4	1	AAVID	533722 B22552	REGHSINK	HEATSINK VOLTREG
60	RESISTOR-0,5%	98F1665	R14,R15,R17-R19,R69,R70,R72,R95,R129,R130,R135,R470,R471,R473	15	PANASONIC	ERJ8GVJ OR	SMC0805	RESISTOR
61	RESISTOR-1.5K,5%	98F1741	R436	1	ROHM	MCR10EZHLJW152	SMC0805	RESISTOR
62	RESISTOR-10,5%	58F1831	R40,R82,R85,R87,R89,R96,R97	7	PANASONIC	ERJ6GVYJ 100S	SMC0805	RESISTOR
63	RESISTOR-100,5%	41F0328	R38,R39,R101,R107,R108,R153,R154,R176,R190,R192,R194,R238,R240,R365,R366,R485	16	PANASONIC	ERJ6VJ 101S	SMC0805	RESISTOR
64	RESISTOR-10K,5%	41F0337	R1,R5-R8,R10,R13,R43,R74-R76,R78,R80,R81,R90-R93,R99,R102,R111-R113,R119,R138-R145,R147,R148,R155-R166,R174,R237,R253-R260,R287-R351,R364,R443,R488-R495,R497,R510	133	PANASONIC	ERJ6GVJ 103S	SMC0805	RESISTOR
65	RESISTOR-12.7K,1%	40G7066	R512	1	PANASONIC	ERJ-6VNF 1272S	SMC0805	RESISTOR

Table E-1. Planar Bill of Materials (Continued)

No	Part Name	Part Number	Reference Designator	Qty	Manufacturer Name	Manufacturer Part Number	JEDEC TYPE	Part Description
66	RESISTOR-150,5%	98F1671	R123	1	ROHM	MCR10EZH MJW151	SMC0805	RESISTOR
67	RESISTOR-1K,5%	41F0333	R12,R21,R28,R29, R31-R36,R41,R44, R47,R55,R77,R100, R230,R239,R280, R505,R513	21	PANASONIC	ERJ6GVJ 102S	SMC0805	RESISTOR
68	RESISTOR-2.7K,5%	09G9748	R261-R272	12	PANASONIC	ERJ-6GVYJ 272S	SMC0805	RESISTOR
69	RESISTOR-200,1%	40G6920	R4	1	PANASONIC	ERJ-6VNF 2000S	SMC0805	RESISTOR
70	RESISTOR-200,5%	98F1424	R73	1	PANASONIC	ERJ-6GEYJ 201V	SMC0805	RESISTOR
71	RESISTOR-22,5%	98F1736	R56,R88,R177, R178	4	PANASONIC	ERJ6GVYJ 220S	SMC0805	RESISTOR
72	RESISTOR-220K,5%	61F2952	R122	1	PANASONIC	ERJ6GVJ 224S	SMC0805	RESISTOR
73	RESISTOR-249,1%	40G7233	R437	1	PANASONIC	ERJ6VNF 2490S	SMC0805	RESISTOR
74	RESISTOR-3.9K,5%	42G3067	R120	1	PANASONIC	ERJ6GVYJ 392S	SMC0805	RESISTOR
75	RESISTOR-300,5%	98F1674	R103,R106,R225, R236,R241,R251, R281-R286	12	ROHM	MCR10EZH MJW301	SMC0805	RESISTOR
76	RESISTOR-33,5%	41F0327	R22-R27,R30,R65, R67,R68,R71,R83, R84,R86,R104, R105,R226-R229, R243,R250	22	PANASONIC	ERJ6GVYJ 330S	SMC0805	RESISTOR
77	RESISTOR-4.7K,5%	41F0336	R231-R235,R273- R279	12	PANASONIC	ERJ-6GVJ 472S	SMC0805	RESISTOR
78	RESISTOR-4.99K,1%	40G7034	R511	1	PANASONIC	ERJ6VNF 4991S	SMC0805	RESISTOR
79	RESISTOR-470K,5%	03G9709	R124	1	ROHM	MCR10EZH LJW474	SMC0805	RESISTOR
80	RESISTOR-5.6K,5%	98F1737	R352-R358	7	PANASONIC	ERJ6GVJ 562S	SMC0805	RESISTOR
81	RESISTOR-510,5%	41F0331	R118,R245	2	PANASONIC	ERJ6GVYJ 511S	SMC0805	RESISTOR
82	RESISTOR-620,5%	61F2960	R20	1	PANASONIC	ERJ6GVJ 621S	SMC0805	RESISTOR
83	RESISTOR-75,5%	61F2961	R79,R359-R362	5	PANASONIC	ERJ6GVJ 750S	SMC0805	RESISTOR
84	RESISTOR-80.6,1%	40G6887	R9	1	PANASONIC	ERJ-6VNF 80R6S	SMC0805	RESISTOR
85	RESISTOR- DO_NOT_POP,5%	-NONE-	R2,R3,R11,R16, R37,R42,R45,R46, R54,R57,R58,R62, R66,R94,R98,R121, R146,R149-R152, R189,R191,R193, R440,R441,R451, R472	28	NONE	NONE	SMC0805	RESISTOR
86	RLS4148_SOIC- BASE	87F4920	CR2	1	ROHM	RLS4148	MELF	40V 1A SCHTK REC
87	S82378ZB_SMD- BASE	82G6542	U7	1	INTEL	S82378ZB	QFP208_5MM	PCI TO ISA BRIDGE CONT
88	XTAL-16.50MHZ	89G3833	Y2A,Y4A	2	ECLIPTEK	ECX-2900- 16.500MHZ	CLP_XTAL	16.5 MHZ CRYSTAL
90	XTAL-DO_NOT_POP	-NONE-	Y3A	1	NONE	NONE	CLP_XTAL	NONE
Total				591				

## **Appendix F**

### **Planar Schematics**

This section contains the schematics of the example planar. For schematics of the MCM, see Section 13.

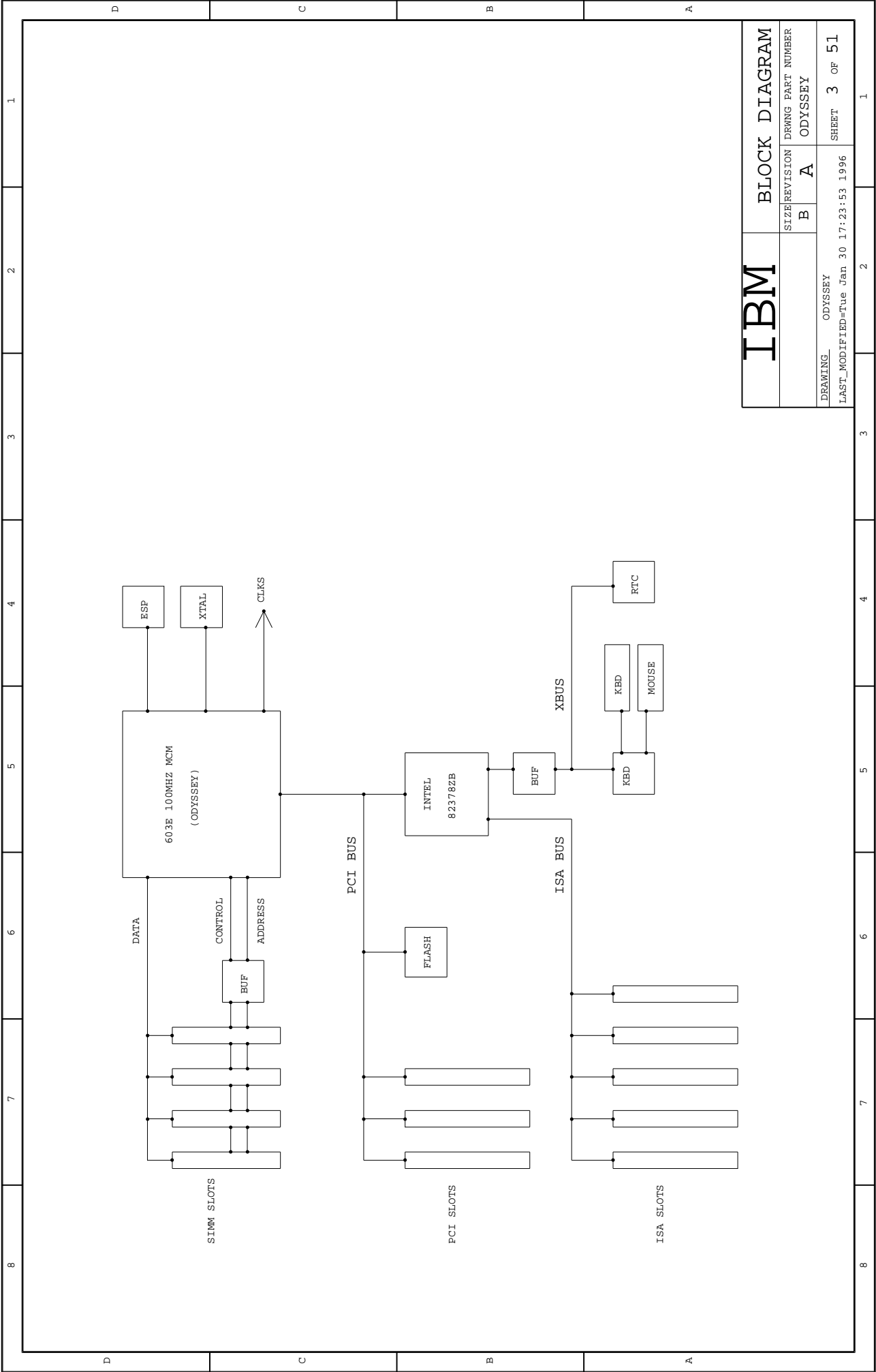
The schematic pages are numbered separately from the rest of the MCM technical specification.

Page 50 of the planar schematics shows a PAL that is used only for development and factory testing of parts of the MCM. This device is not required in user systems, and can be completely removed from the circuit. Note that in normal operation, the PAL is not populated, and the nets that are connected to the PAL outputs are driven by other devices.





[illegible]



IBM		BLOCK DIAGRAM	
SIZE	REVISION	DRAWING PART NUMBER	
B	A	ODYSSEY	
DRAWING		SHEET 3 OF 51	
ODYSSEY		LAST_MODIFIED=Tue Jan 30 17:23:53 1996	

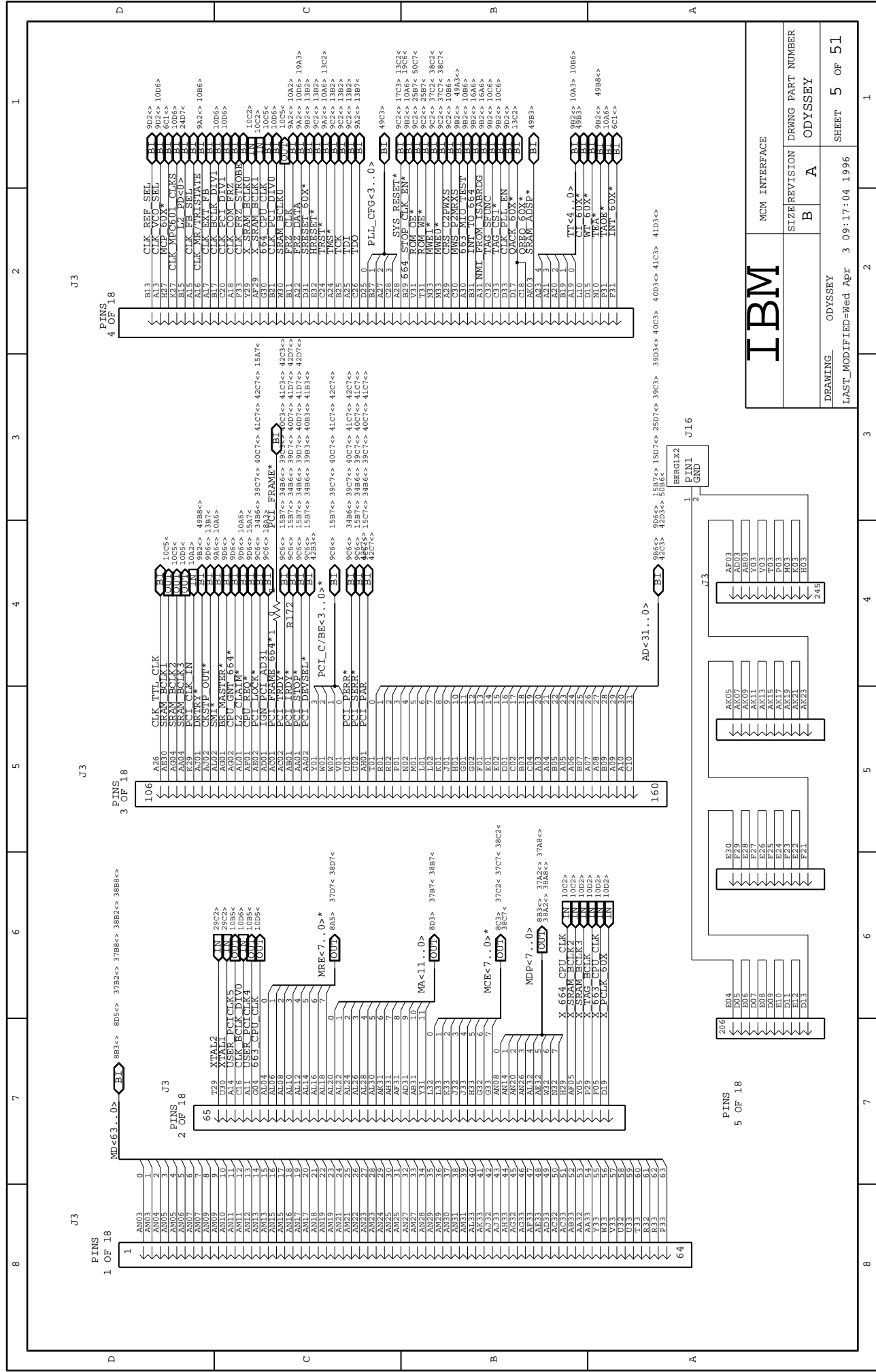
# CHANGE HISTORY

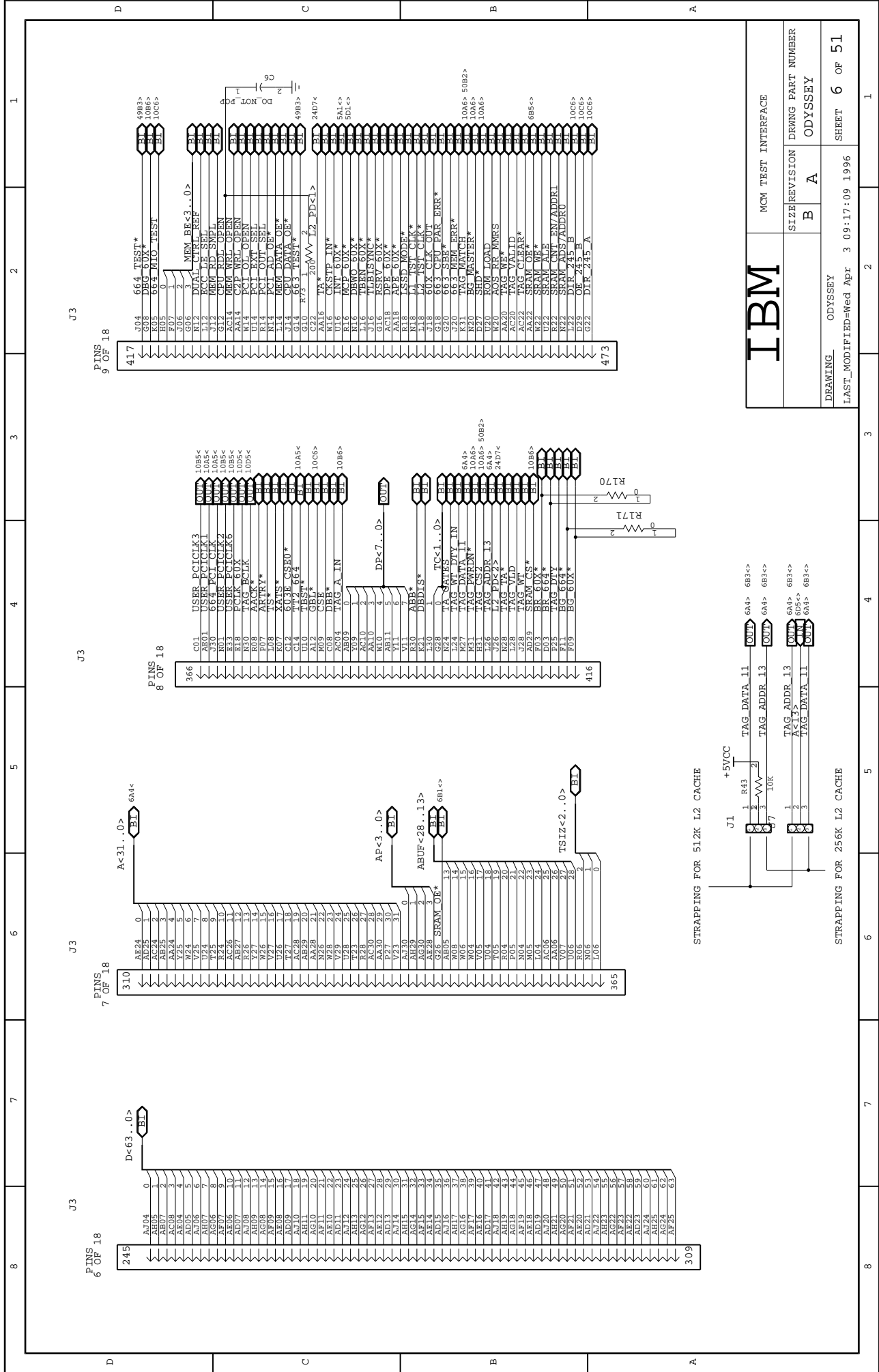
V0.1A 07/28/95 .... DRAFT

IBM		CRITICAL NETS CHANGE HISTORY	
SIZE	REVISION	DWG PART NUMBER	
B	A	ODYSSEY	
DRAWING ODYSSEY		SHEET 4 OF 51	
LAST_MODIFIED=Mon Jan 29 11:20:08 1996			

8	7	6	5	4	3	2	1
<div>IBM</div> <div>CHANGE HISTORY</div> <div>V0.1A 07/28/95 .... DRAFT</div> <div><div>CRITICAL NETS INSTRUCTIONS</div><div>CRITICAL NETS</div><div>CHANGE HISTORY</div><div>DRAWING ODYSSEY</div><div>LAST_MODIFIED=Mon Jan 29 11:20:08 1996</div><div>SHEET 4 OF 51</div></div>							





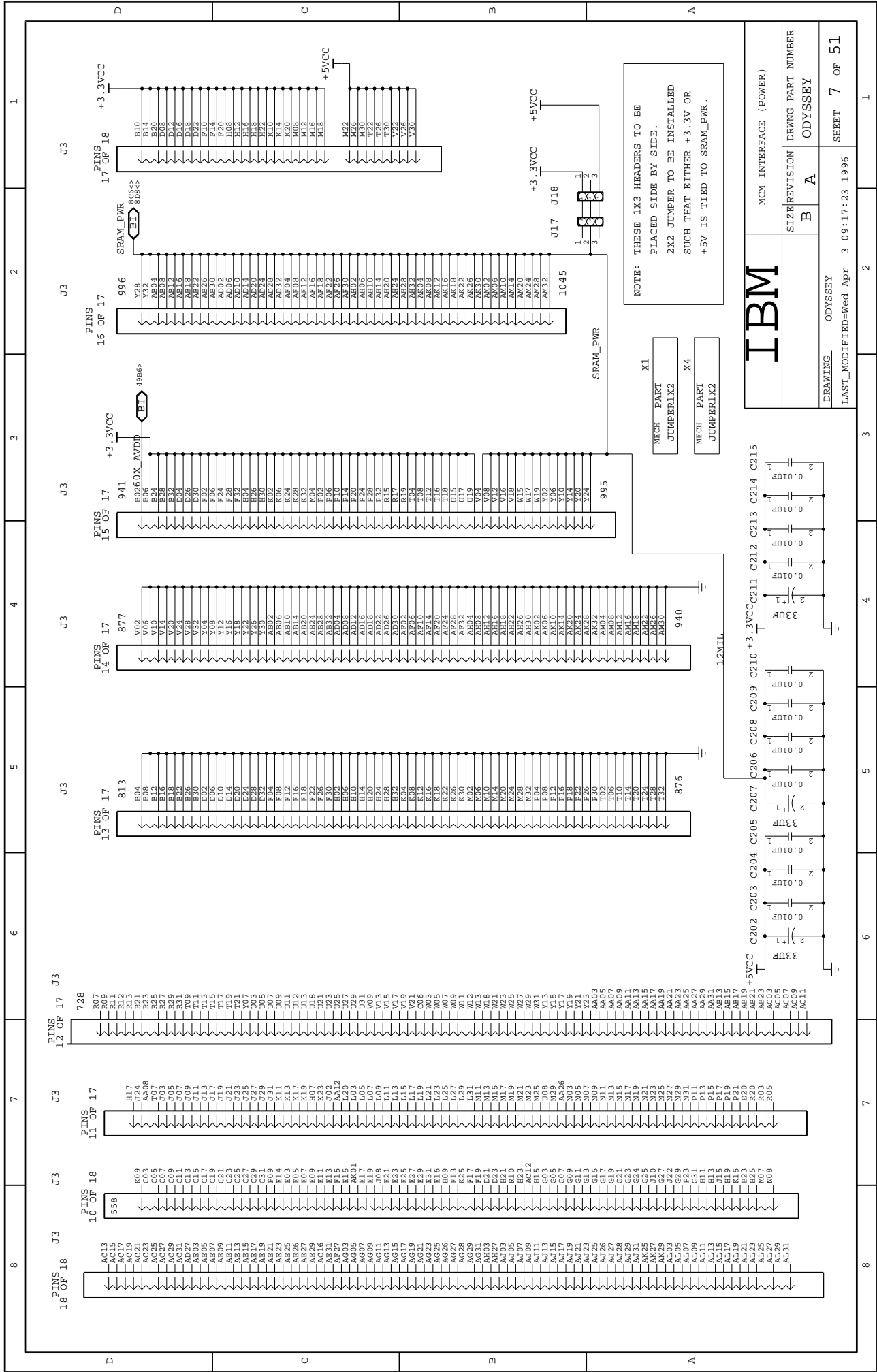


IBM

MCM TEST INTERFACE

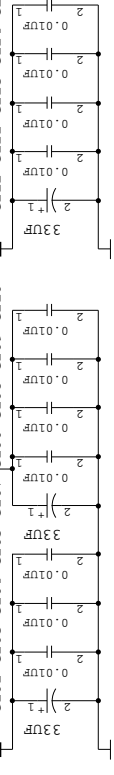
SIZE/REVISION	B	A	ODYSSEY
6	6	6	6

DRAWING	ODYSSEY	LAST_MODIFIED=Wed Apr 3 09:17:09 1996	SHEET 6 OF 51
---------	---------	---------------------------------------	---------------

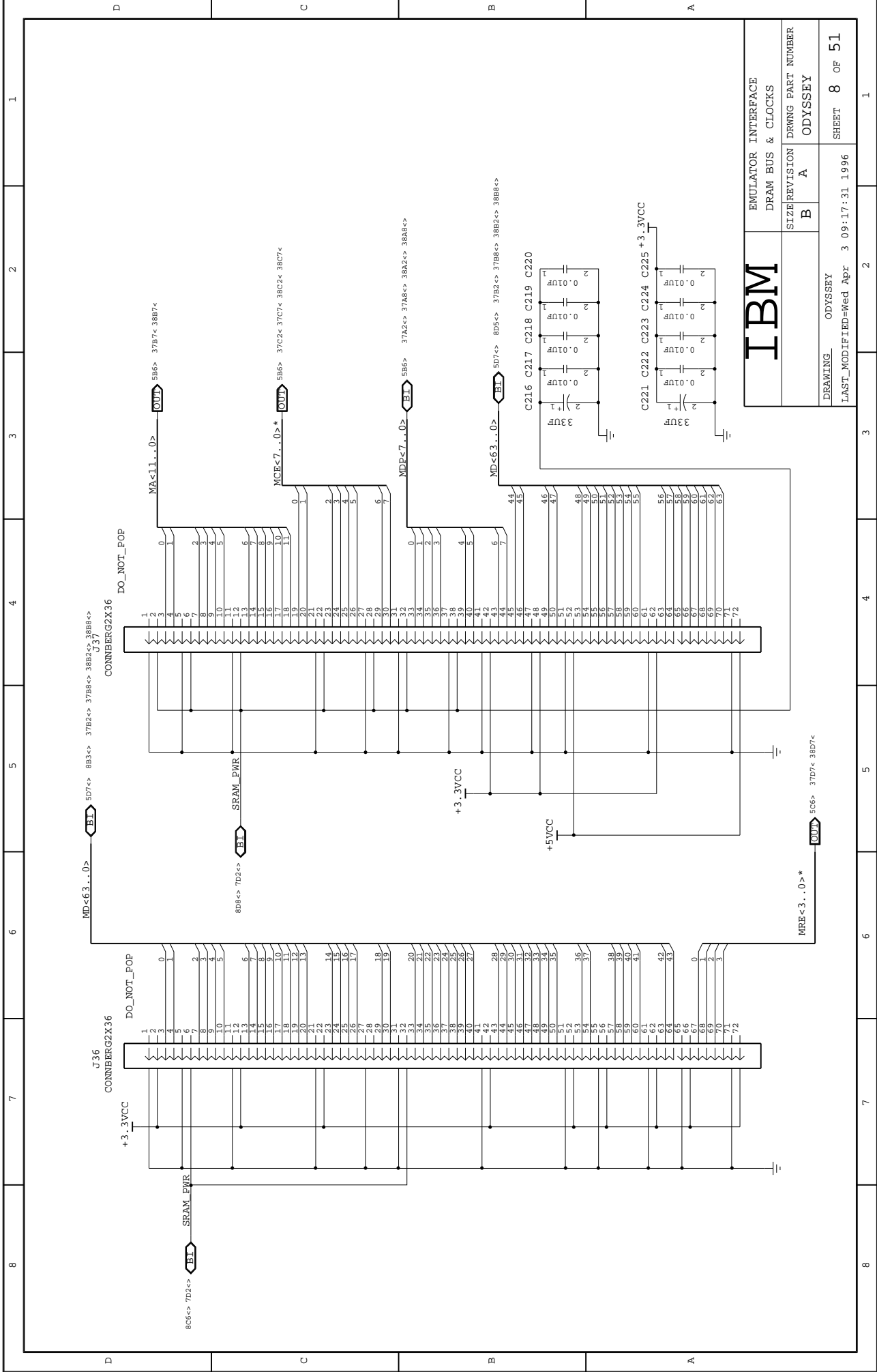


IBM

+5VCC C202 C203 C204 C205 C207 C206 C208 C209 C210 +3.3VCC C211 C212 C213 C214 C215



MCM INTERFACE (POWER)	
SIZE/REVISION	DRAWING PART NUMBER
B A	ODYSSEY
DRAWING	ODYSSEY
LAST_MODIFIED=Wed Apr 3 09:17:23 1996	
SHEET 7 OF 51	

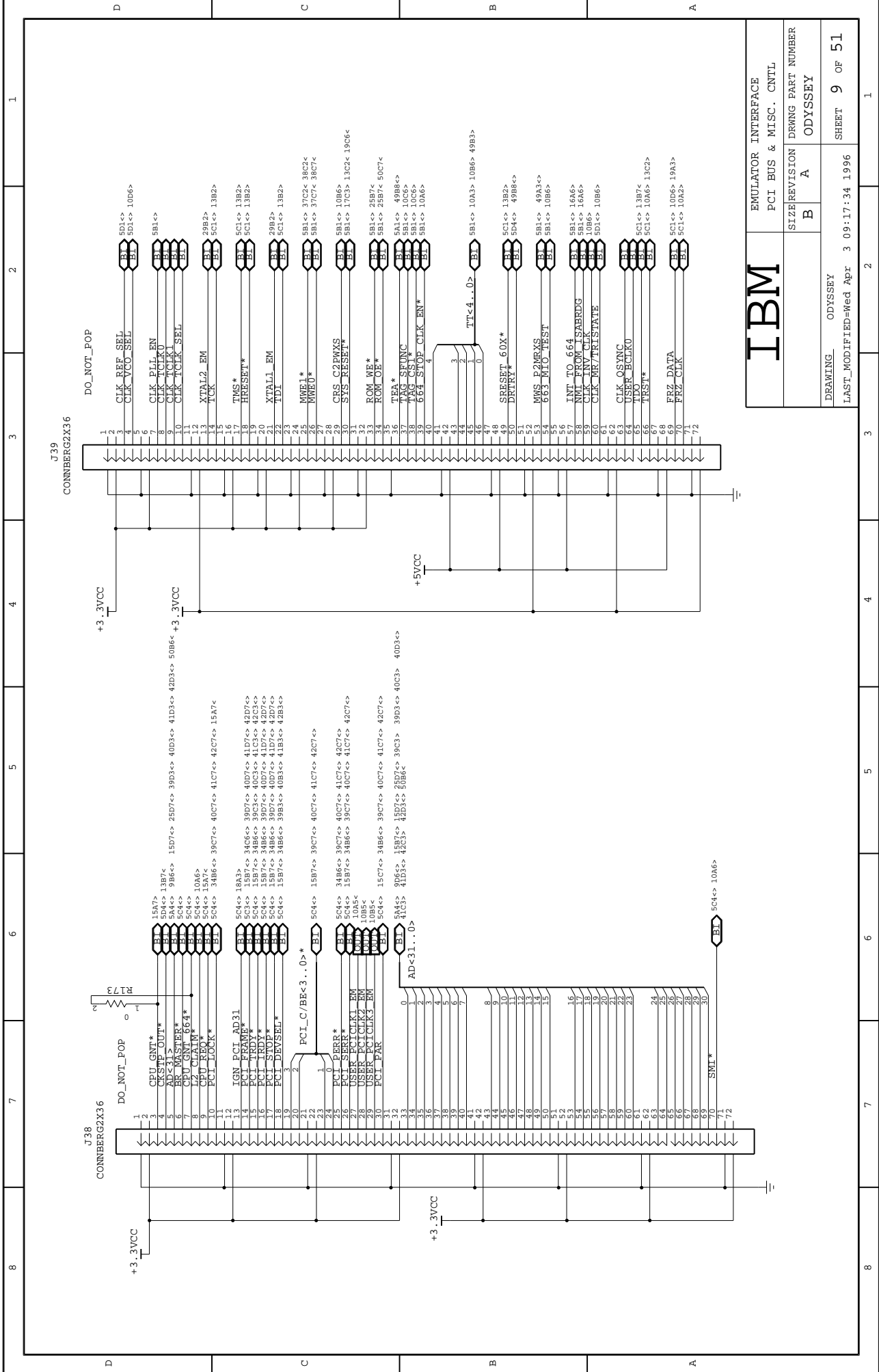


IBM

EMULATOR INTERFACE  
DRAM BUS & CLOCKS

SIZE	REVISION	DRAWING PART NUMBER
B	A	ODYSSEY

DRAWING	ODYSSEY	LAST_MODIFIED=Wed Apr 3 09:17:31 1996
SHEET	8	OF 51

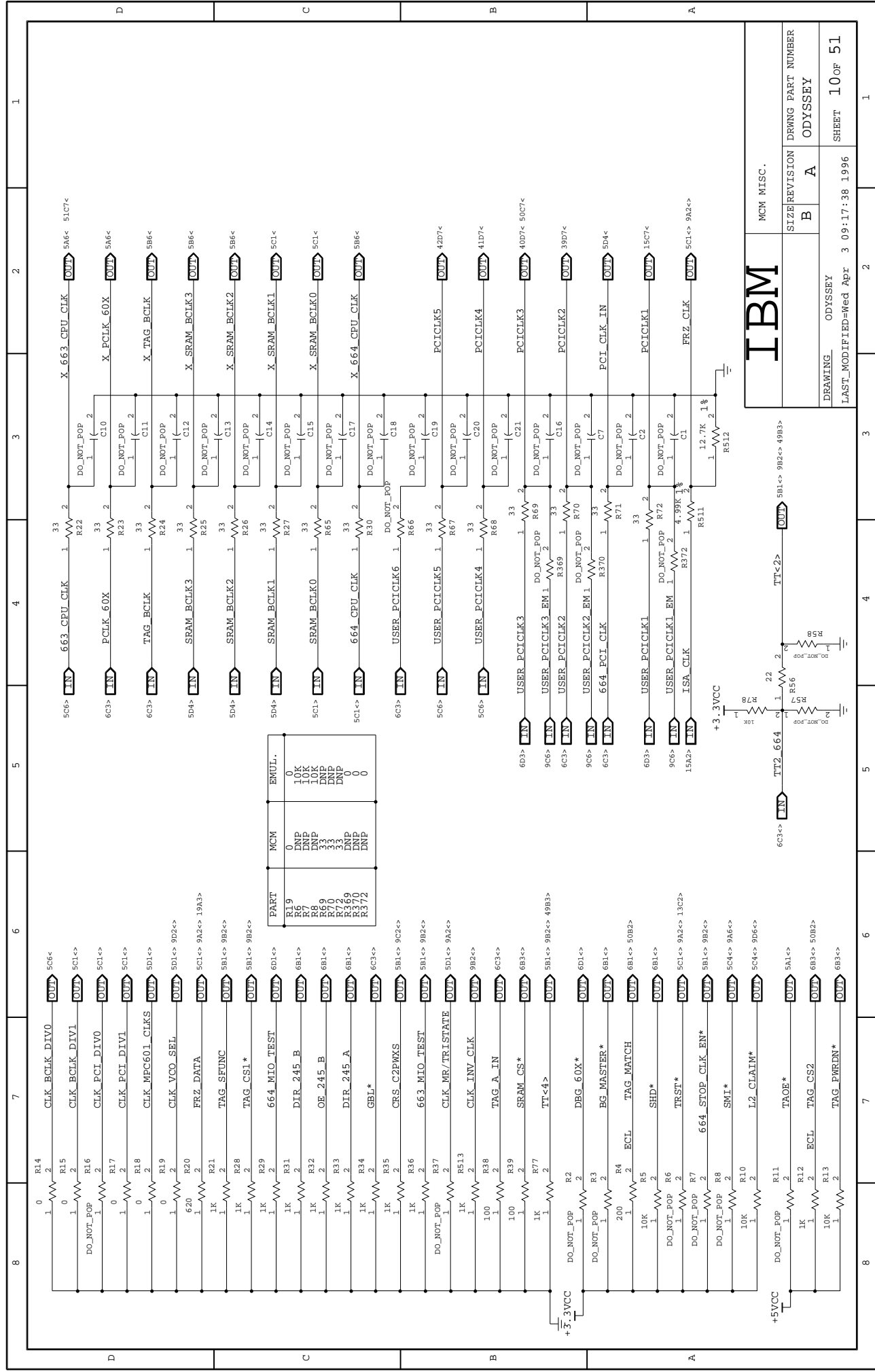


IBM

EMULATOR INTERFACE  
PCI BUS & MISC. CNTL

SIZE/REVISION	DRAWING PART NUMBER
B	A

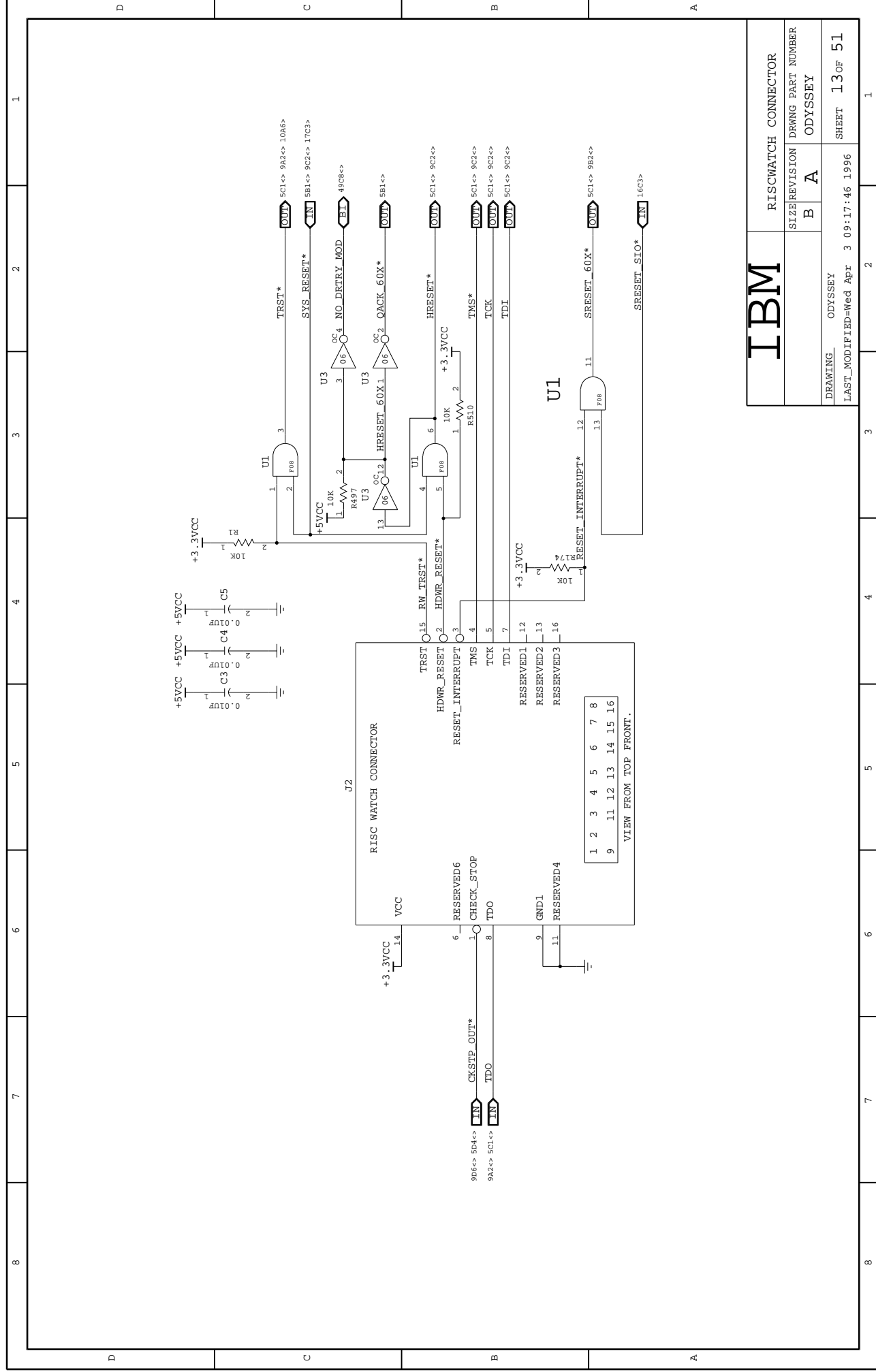
DRAWING	ODYSSEY	LAST MODIFIED=Wed Apr 3 09:17:34 1996
9	OF	51



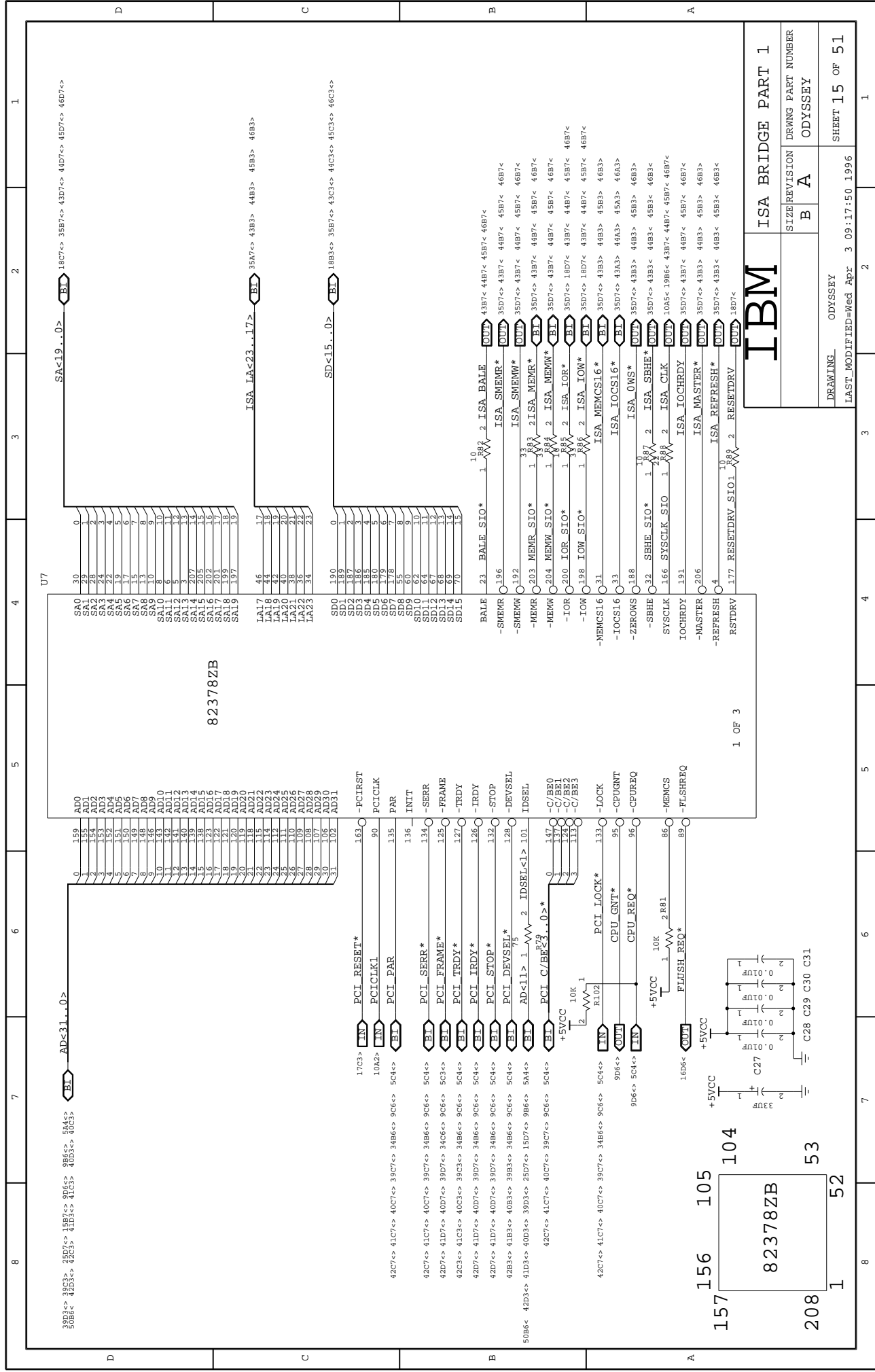
8	7	6	5	4	3	2	1
BLANK							
IBM							
DRAWING		SIZE		REVISION		DRAWING PART NUMBER	
ODYSSEY		B		A		ODYSSEY	
LAST_MODIFIED=Mon Jan 29 16:02:20 1996							
SHEET 11 OF 51							

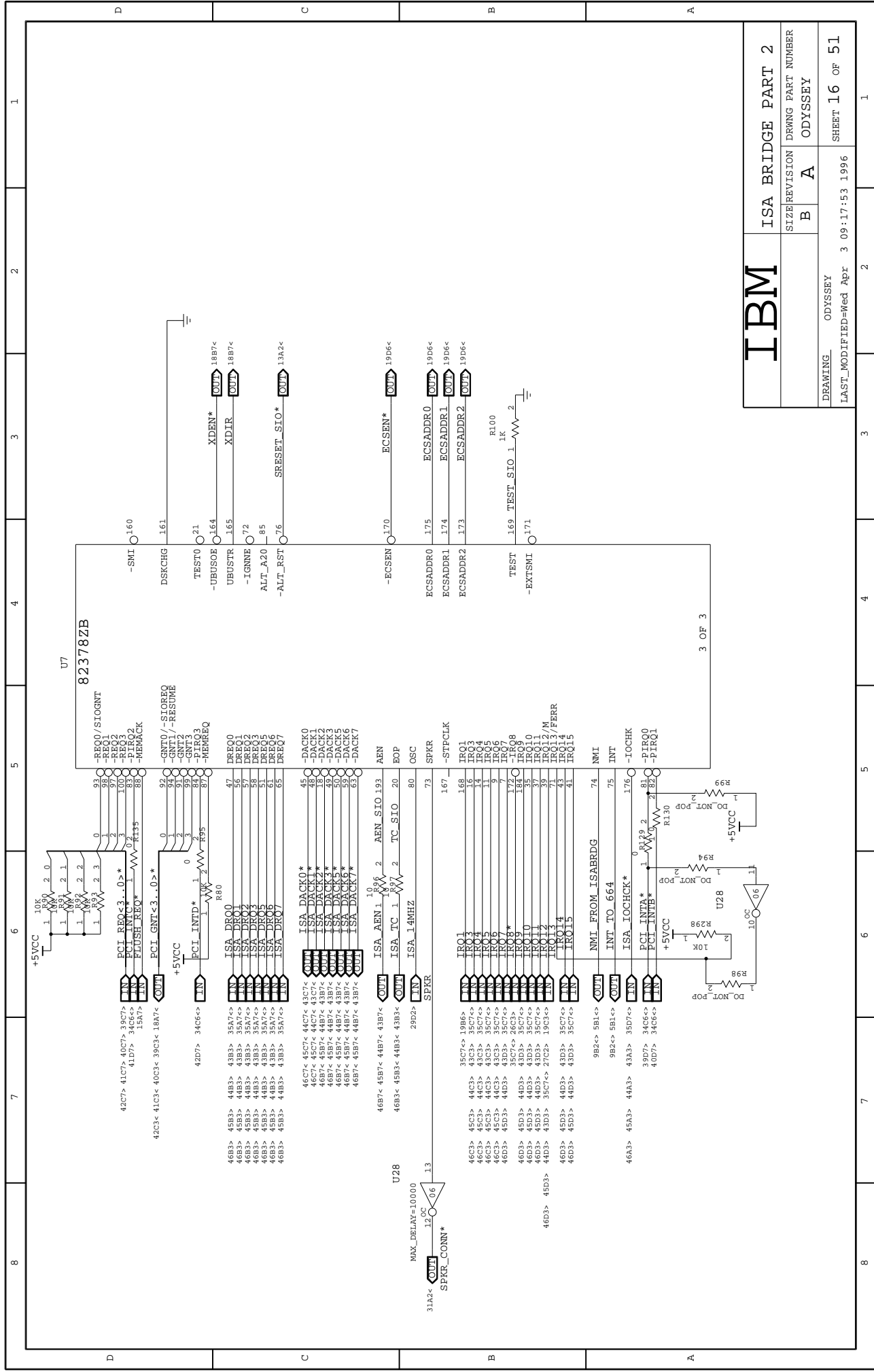
8	7	6	5	4	3	2	1
BLANK							
IBM							
DRAWING		SIZE		REVISION		DRAWING PART NUMBER	
ODYSSEY		B		A		ODYSSEY	
LAST_MODIFIED=Mon Jan 29 16:02:35 1996							
SHEET 12 OF 51							



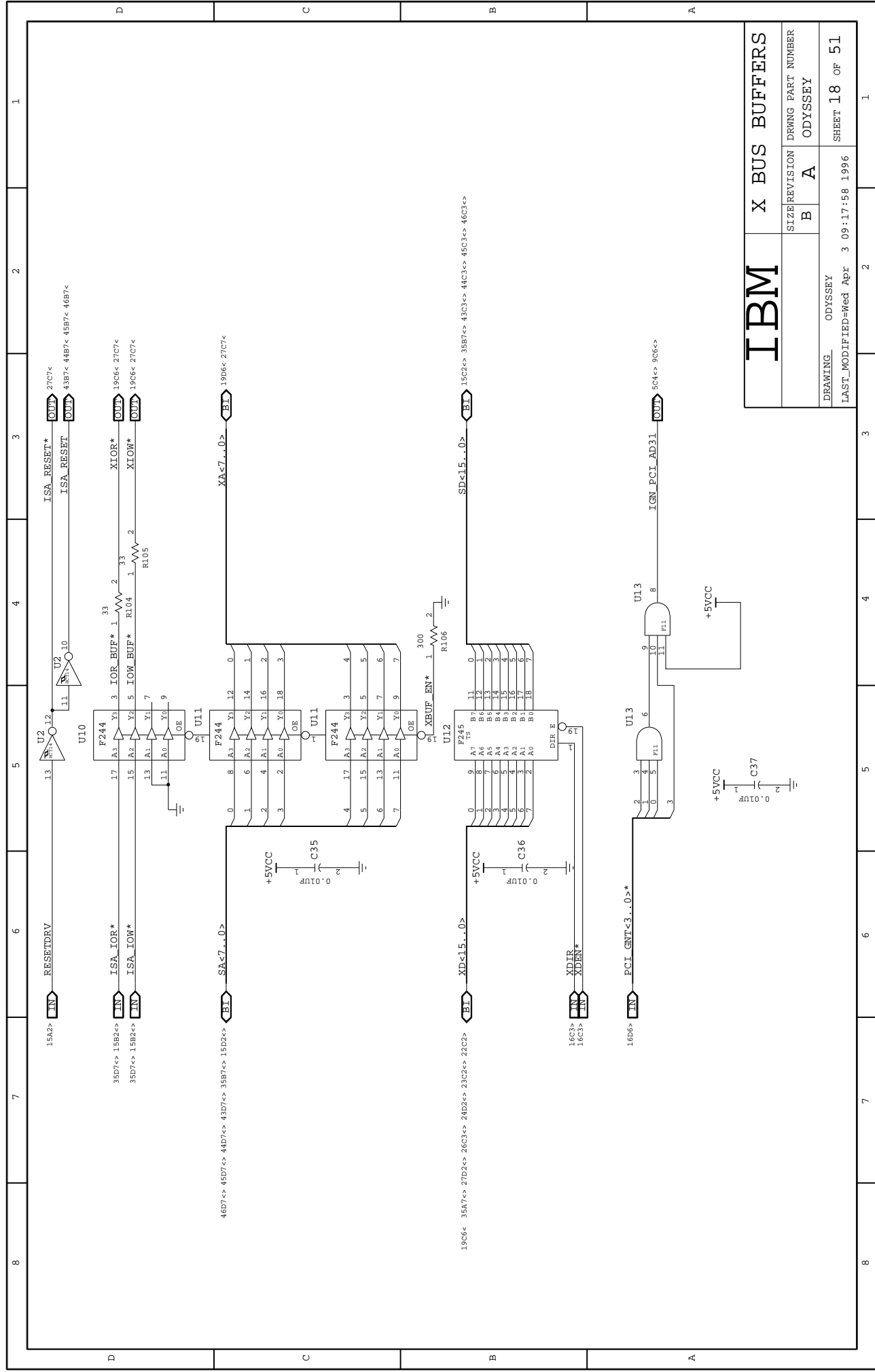


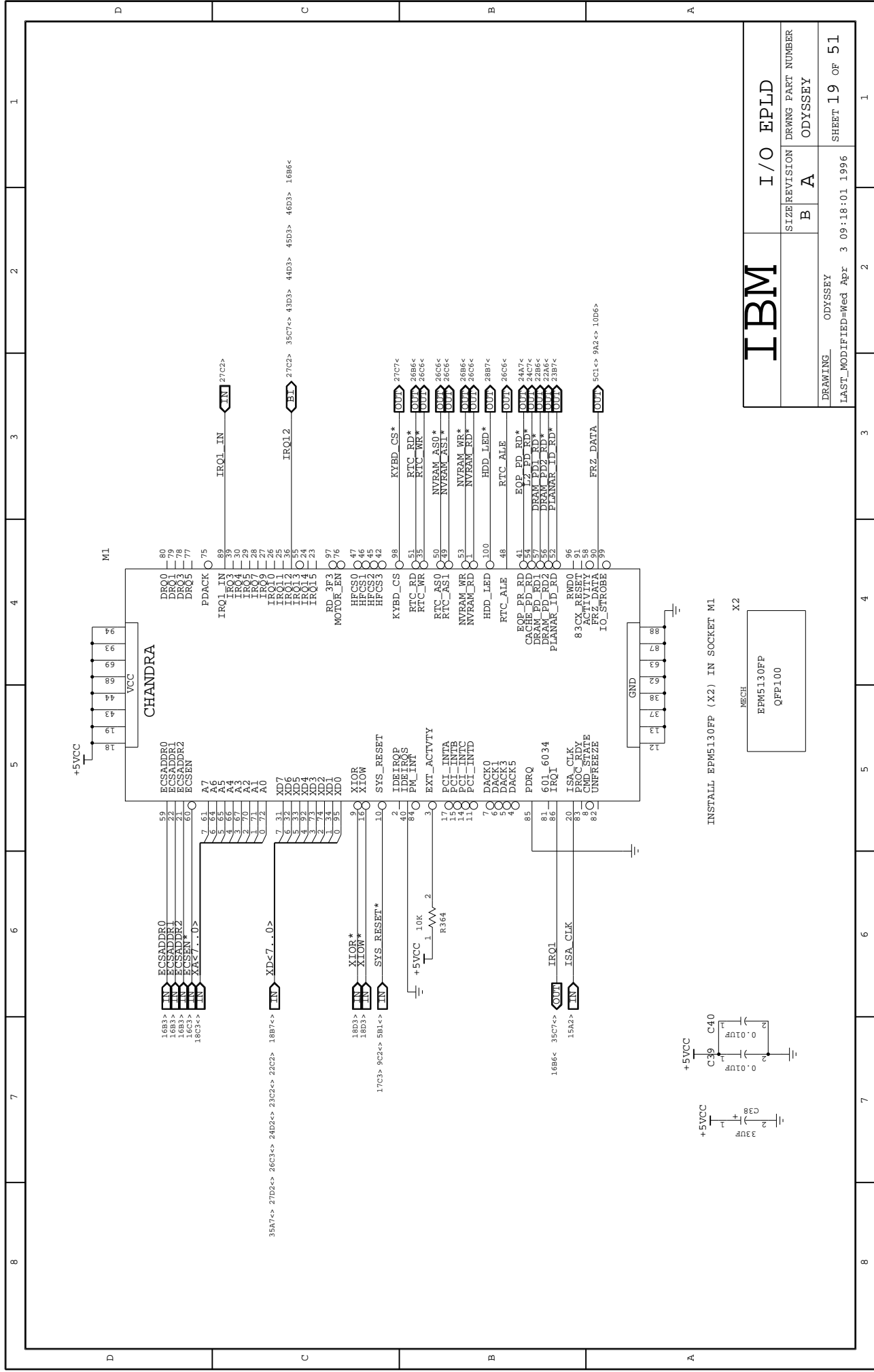
8	7	6	5	4	3	2	1
BLANK							
IBM							
DRAWING		ODYSSEY		LAST_MODIFIED=Mon Jan 29 16:02:54 1996		SHEET 14 OF 51	
SIZE		REVISION		DRAWING PART NUMBER			
B		A		ODYSSEY			











BLANK

IBM

BLANK

SIZE	REVISION	DRWG PART NUMBER
B	A	ODYSSEY

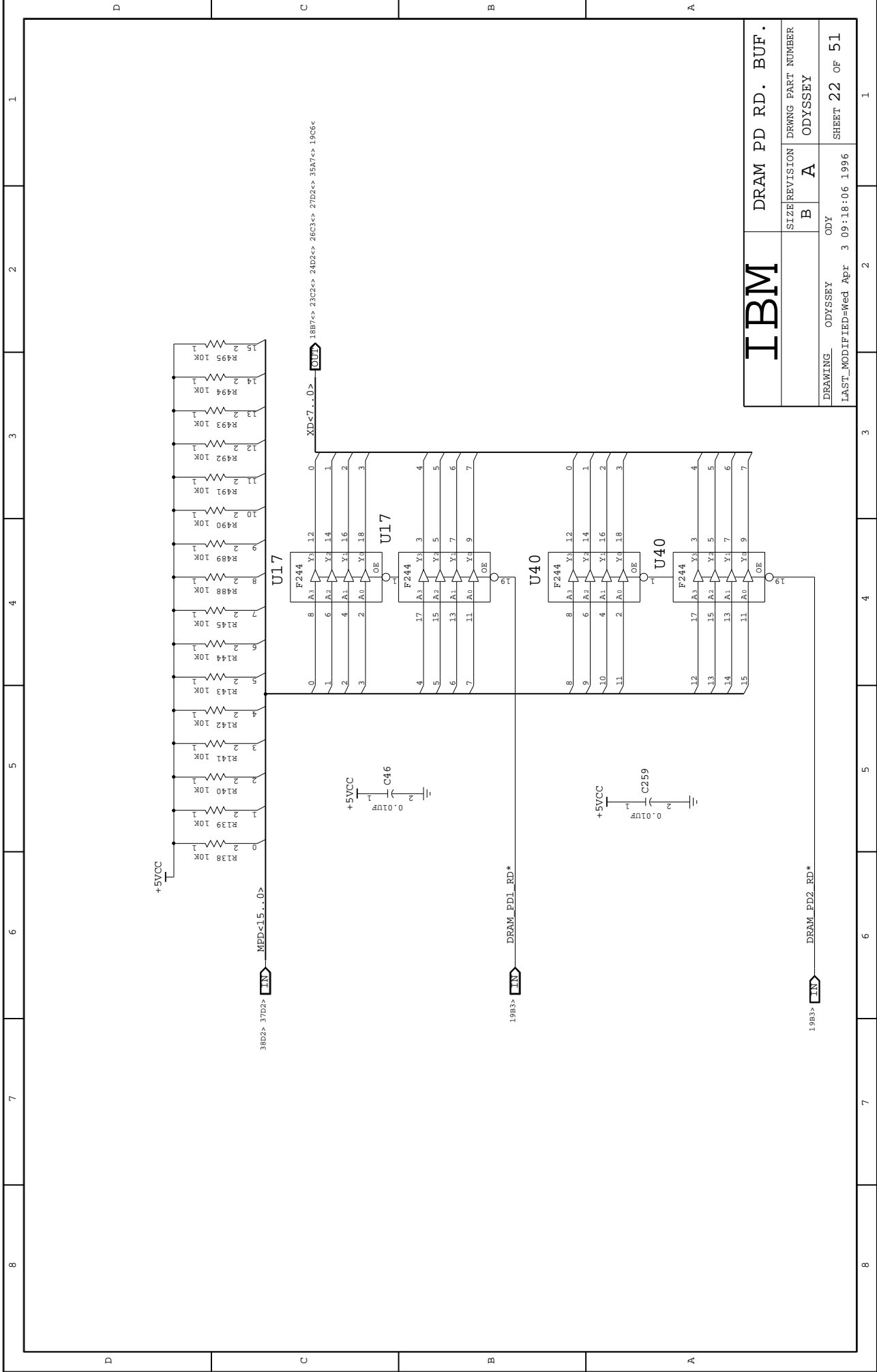
DRWG PART NUMBER  
ODYSSEY

DRAWING	ODYSSEY			
LAST_MODIFIED=Mon Jan 29 16:03:50 1996				
				SHEET 20 OF 51

SHEET 20 OF 51



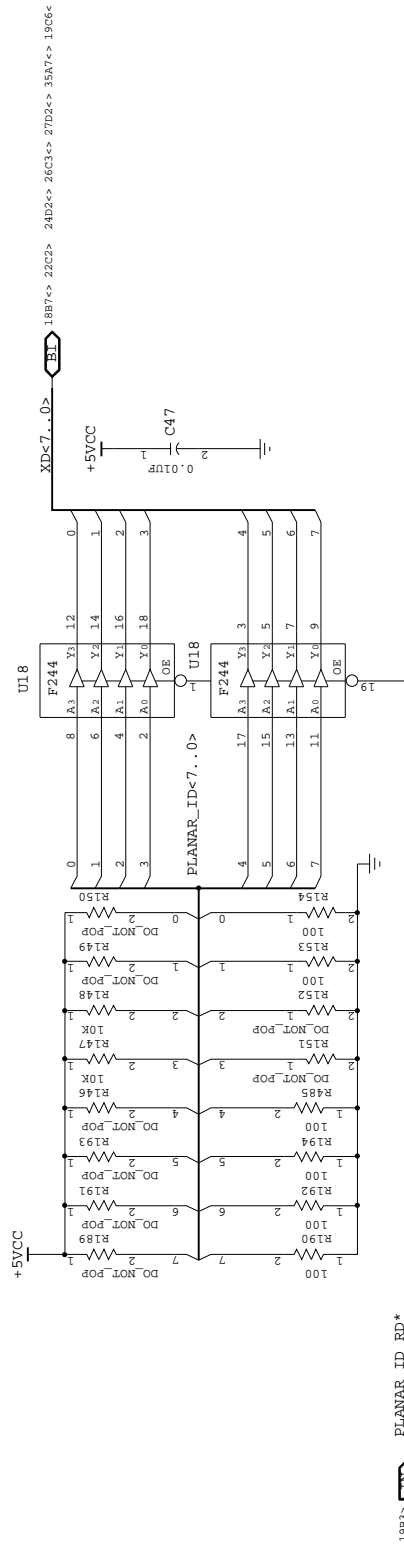




IBM

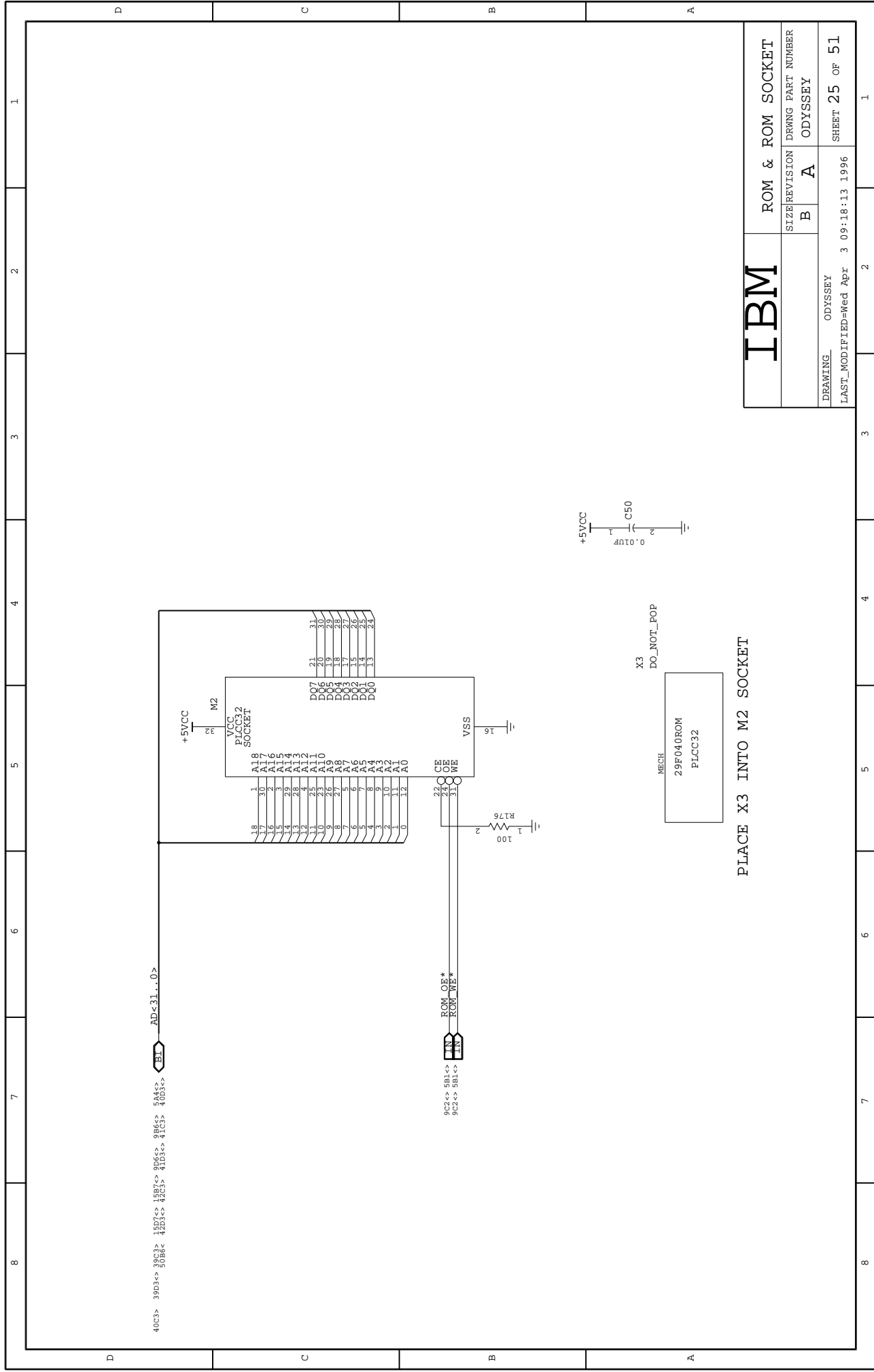
DRAM PD RD. BUF.

SIZE REVISION		DRAWING PART NUMBER	
B	A	ODYSSEY	ODYSSEY
DRAWING		ODYSSEY	ODYSSEY
LAST_MODIFIED=Wed Apr 3 09:18:06 1996		SHEET 22 OF 51	

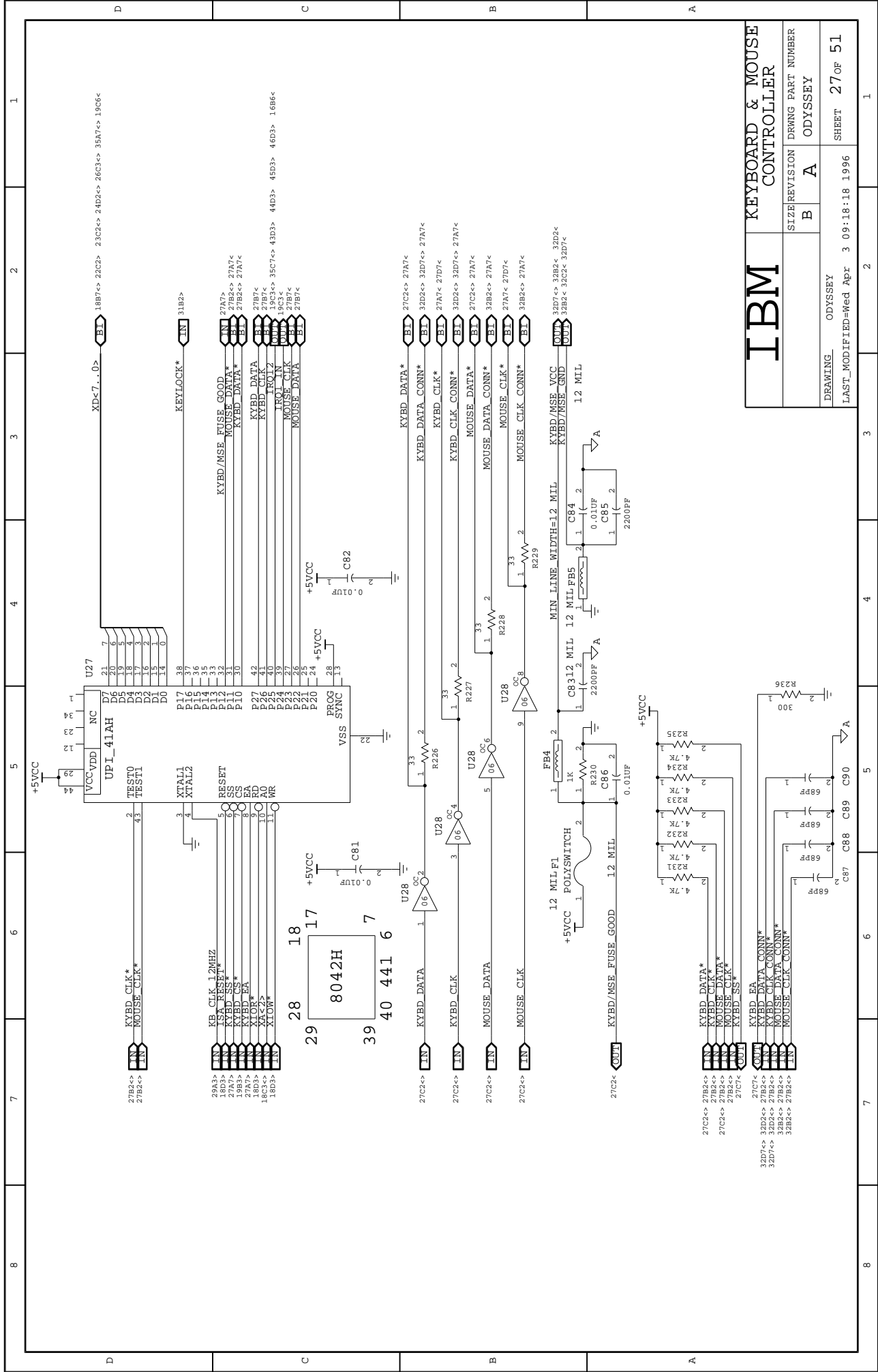


IBM	PLANAR ID RD.	
	SIZE/REVISION	DRAWING PART NUMBER
	B	A ODYSSEY
<u>DRAWING</u> ODYSSEY LAST MODIFIED=Wed Apr 3 09:18:08 1996	SHEET 23 OF 51	



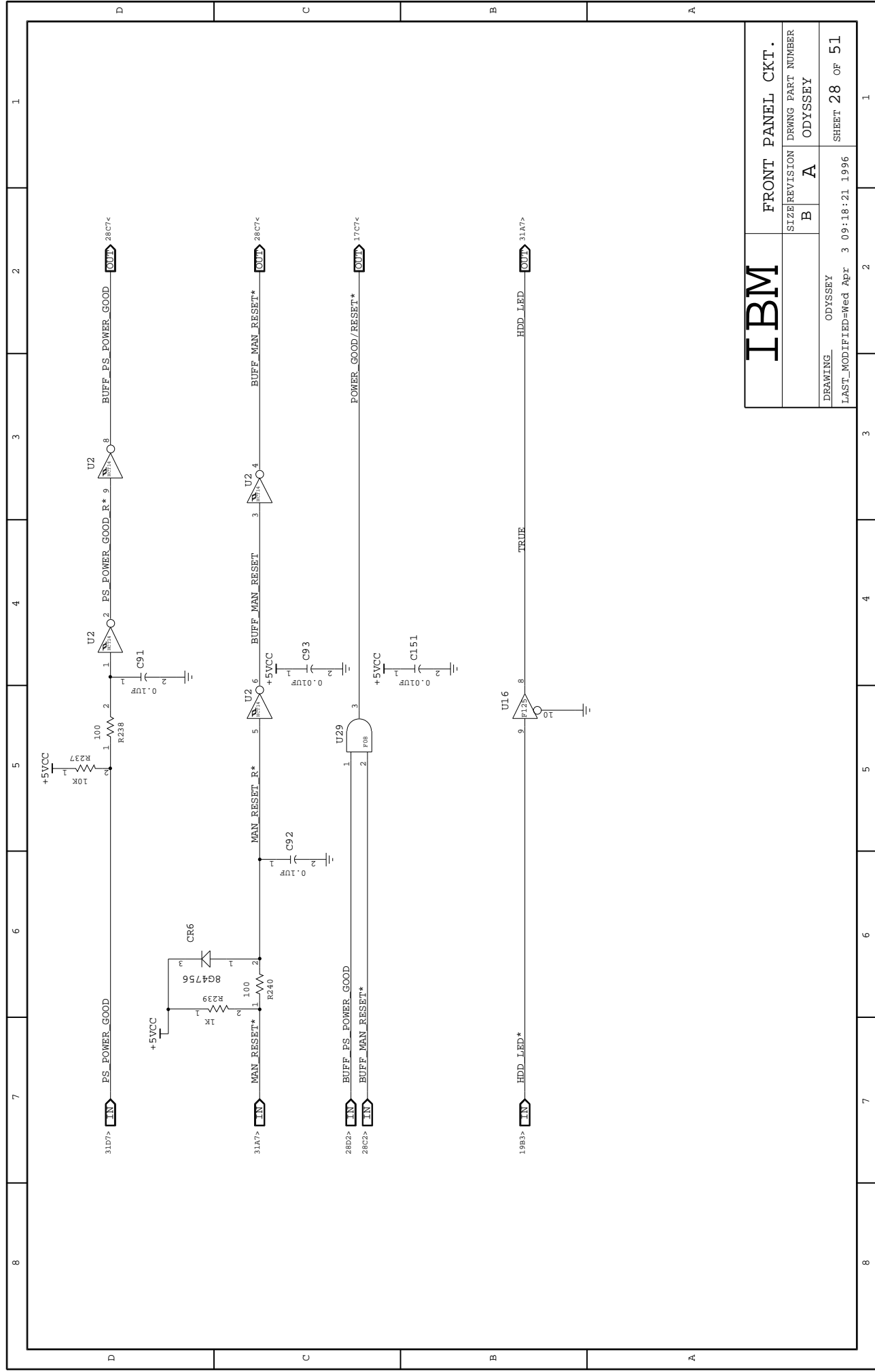






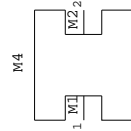
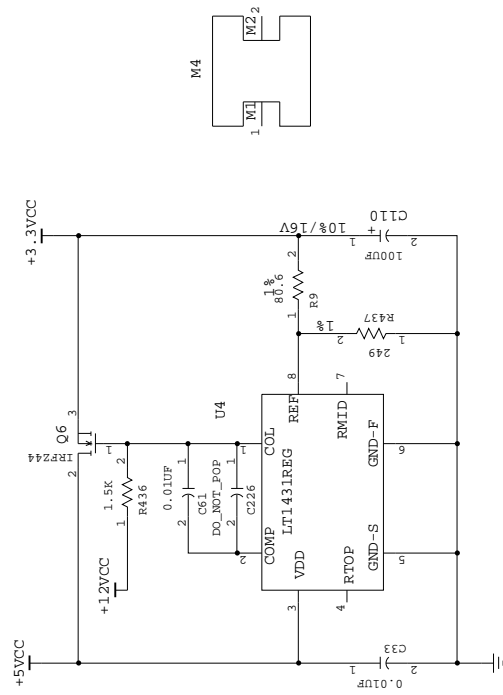
# IBM KEYBOARD & MOUSE CONTROLLER

SIZE/REVISION		DRAWING PART NUMBER	
B	A	ODYSSEY	
DRAWING		SHEET 27 OF 51	
LAST MODIFIED=Wed Apr 3 09:18:18 1996			



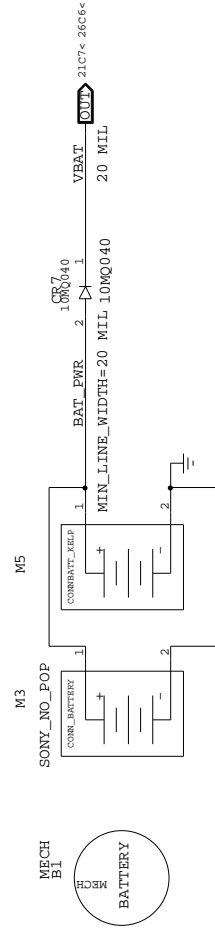




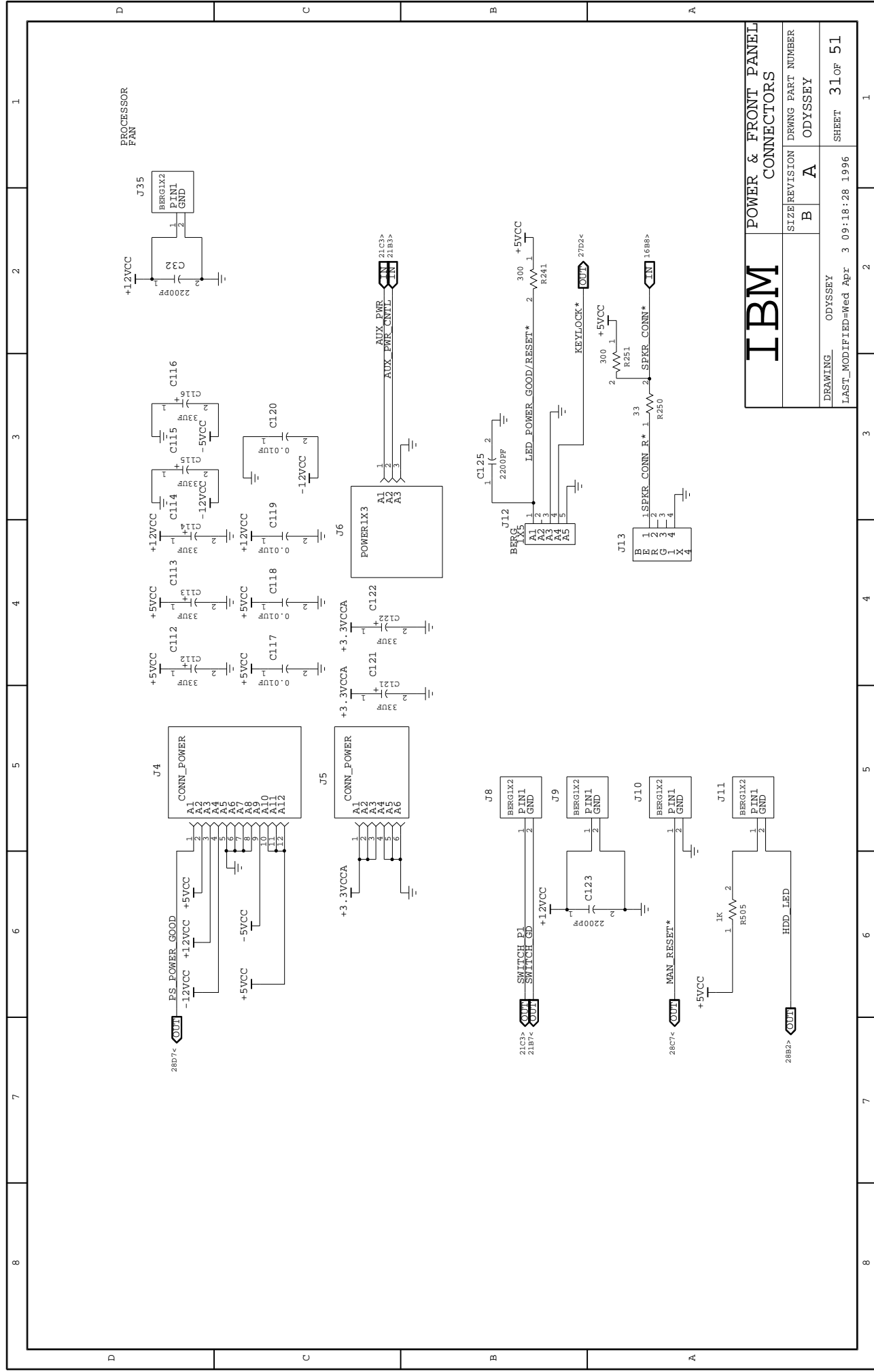


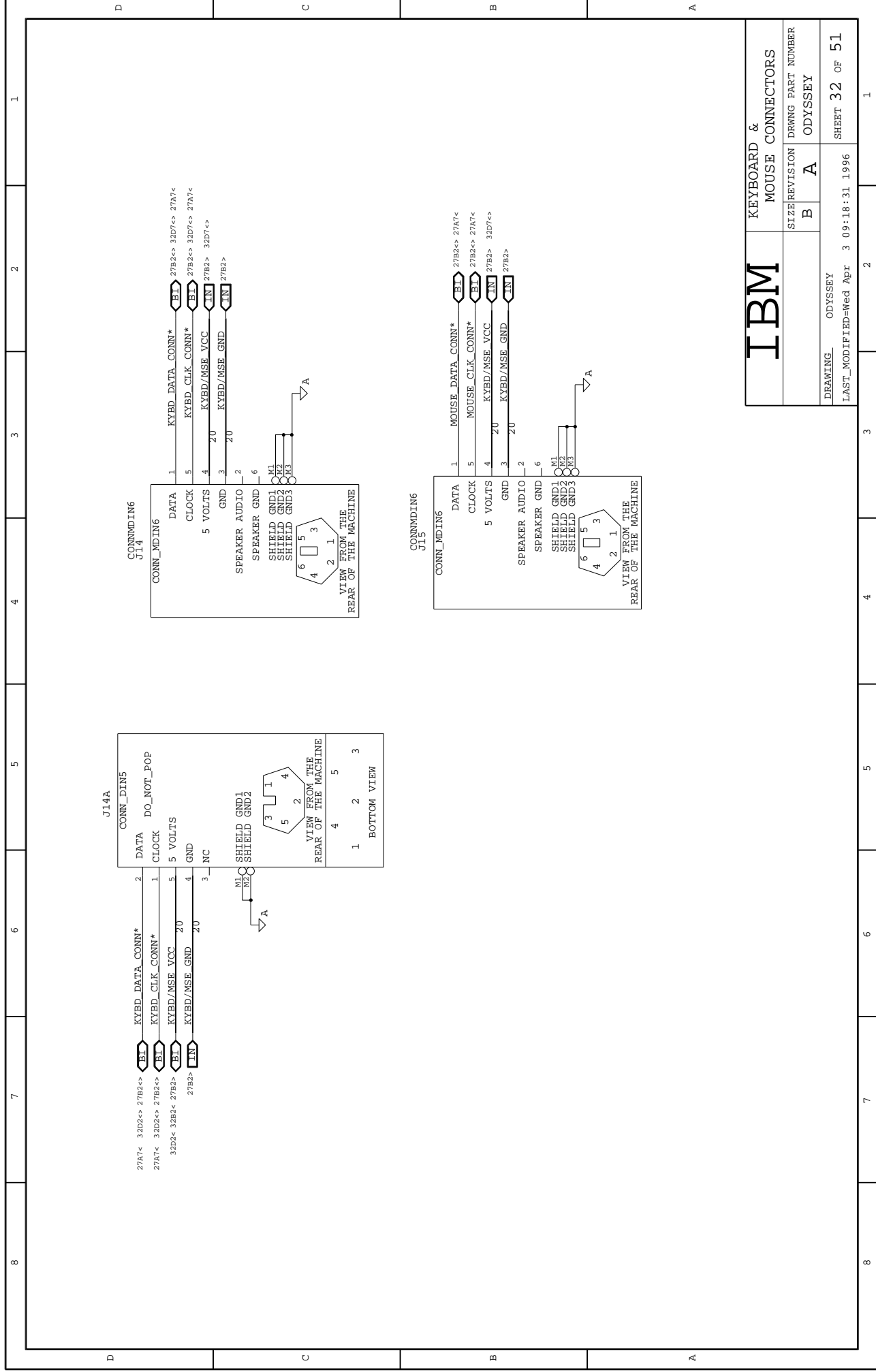
NOTE: VOLTAGE REGULATOR OUTPUT CAN BE VARRIED BY CHANGING R9.

VOLTAGE	R9
2.5V	DO NOT POP
3.3V	80 6 1%
3.6V	110 1%

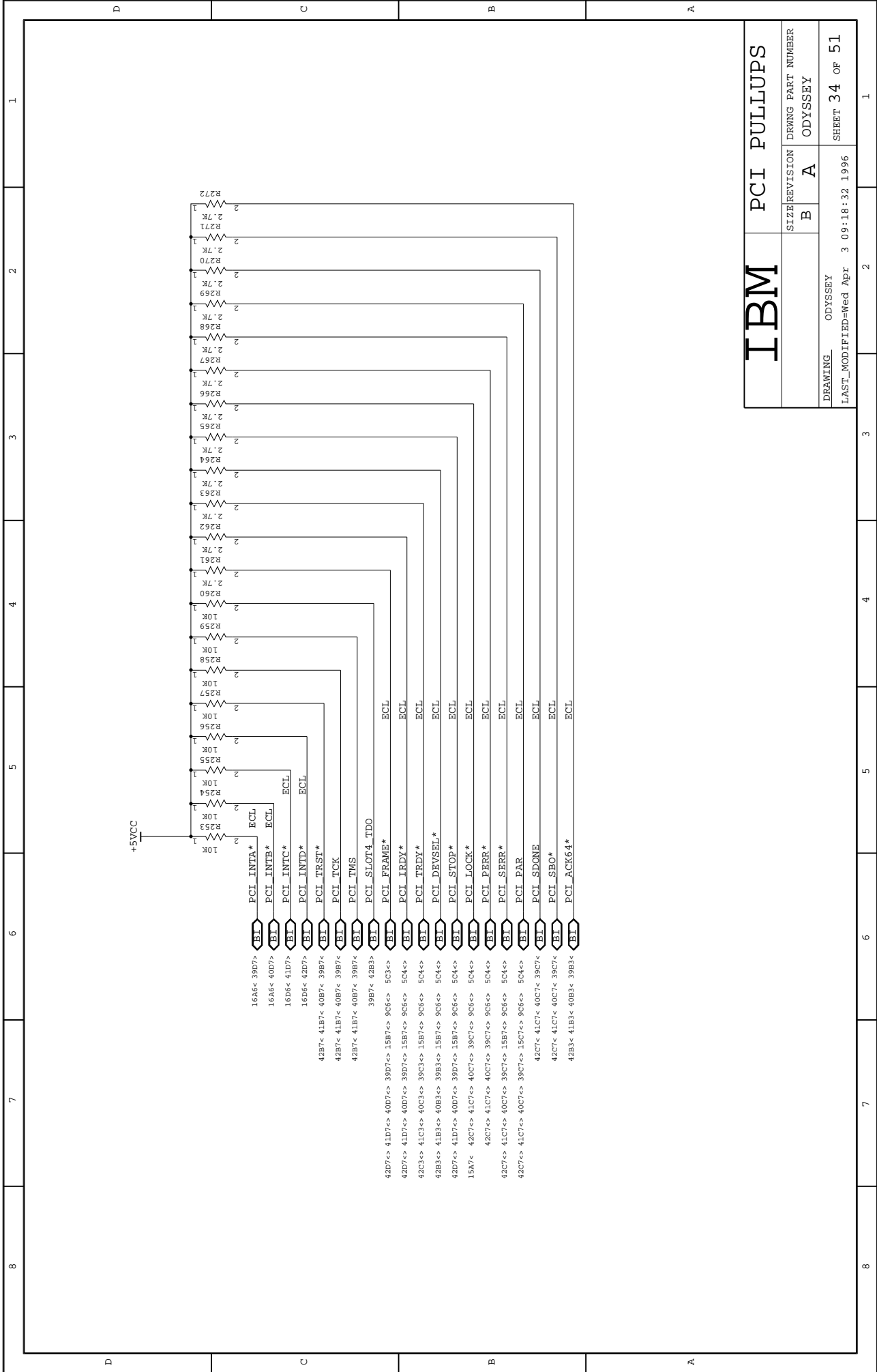


IBM		VOLTAGE REGULATORS	
		SIZE/REVISION	DRAWING PART NUMBER
		B	A
DRAWING ODYSSEY		ODY	
LAST_MODIFIED=Wed Apr 3 09:18:25 1996		SHEET 30 OF 51	



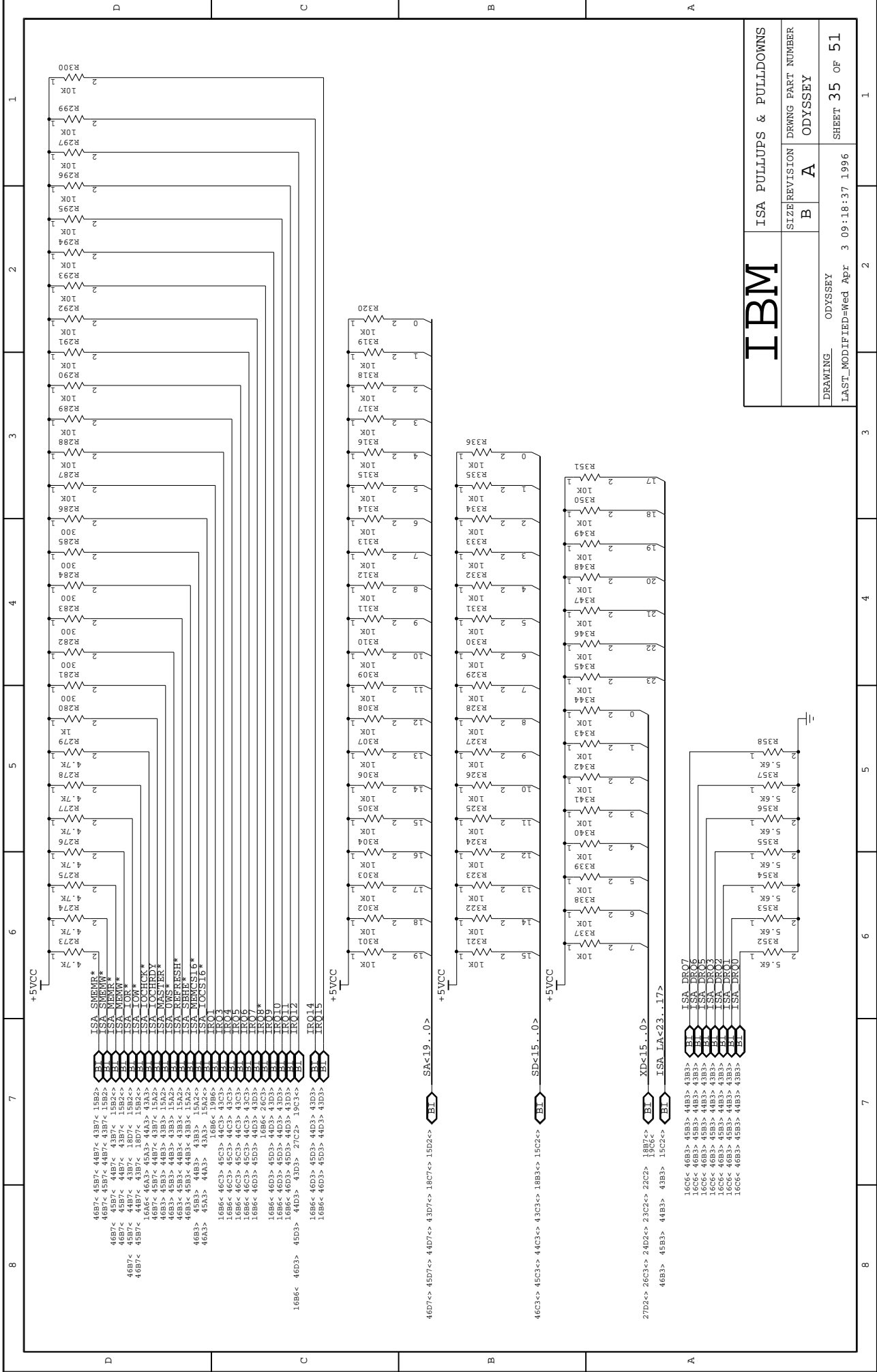






# IBM PCI PULLUPS

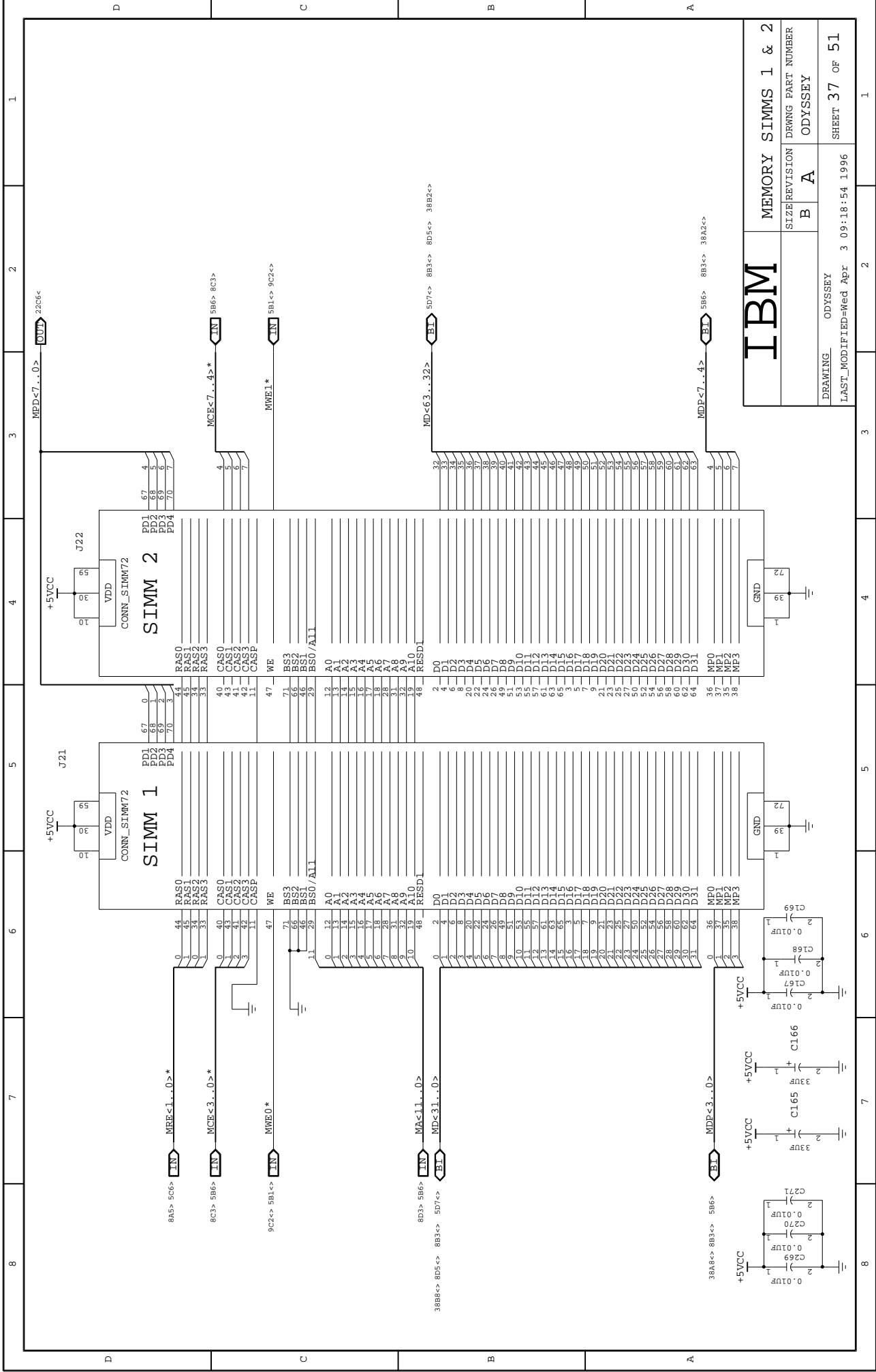
SIZE/REVISION		DRAWING PART NUMBER	
B	A	ODYSSEY	
DRAWING		LAST_MODIFIED=Wed Apr 3 09:18:32 1996	
ODYSSEY		SHEET 34 OF 51	



IBM		ISA PULLUPS & PULLDOWNS		
DRAWING	ODYSSEY	SIZE/REVISION		DRWG PART NUMBER
		B A		ODYSSEY
		LAST_MODIFIED=Wed Apr 3 09:18:37 1996		SHEET 35 OF 51





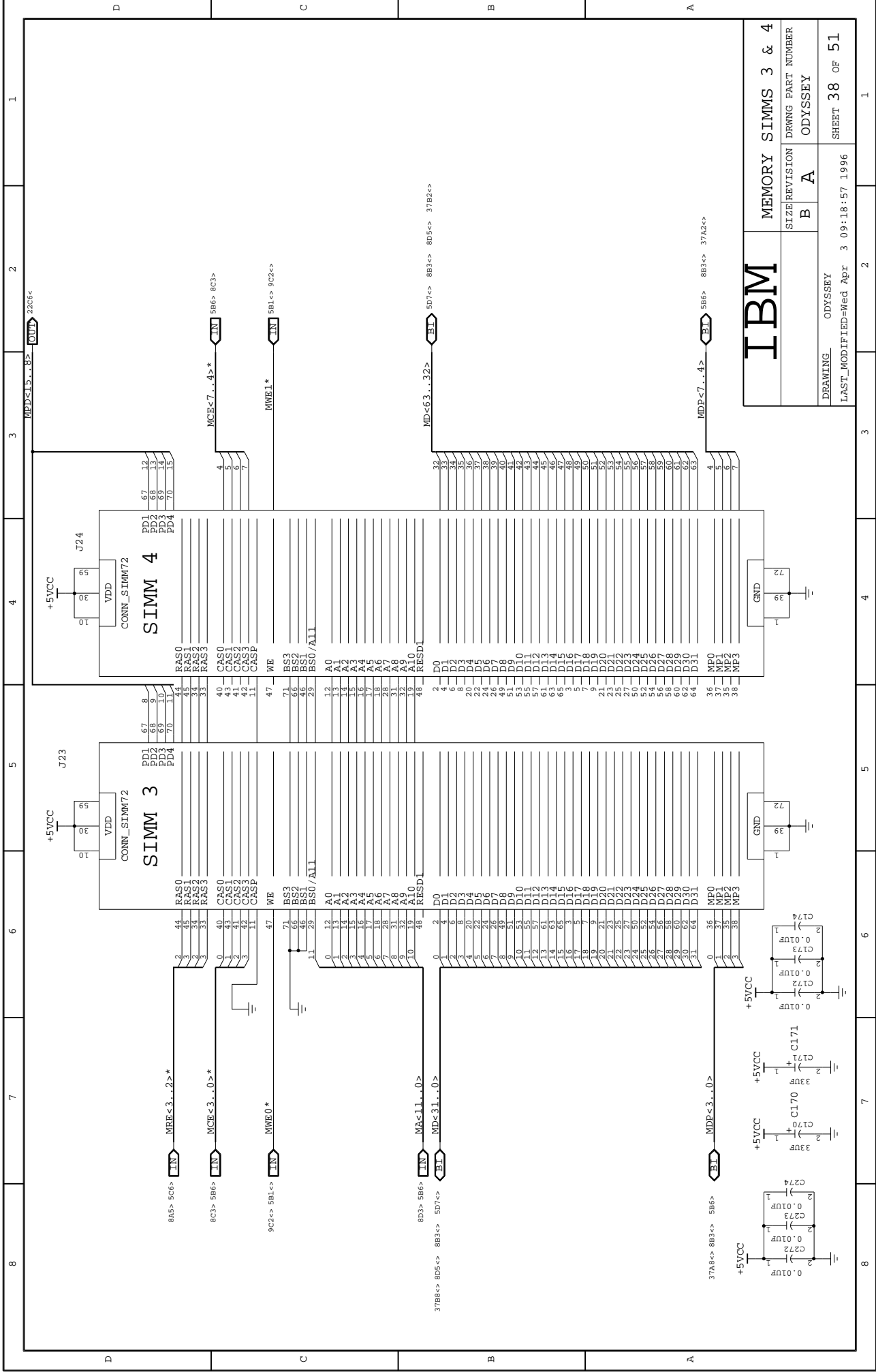


IBM

MEMORY SIMMS 1 & 2

SIZE/REVISION	DRAWING PART NUMBER
B A	ODYSSEY

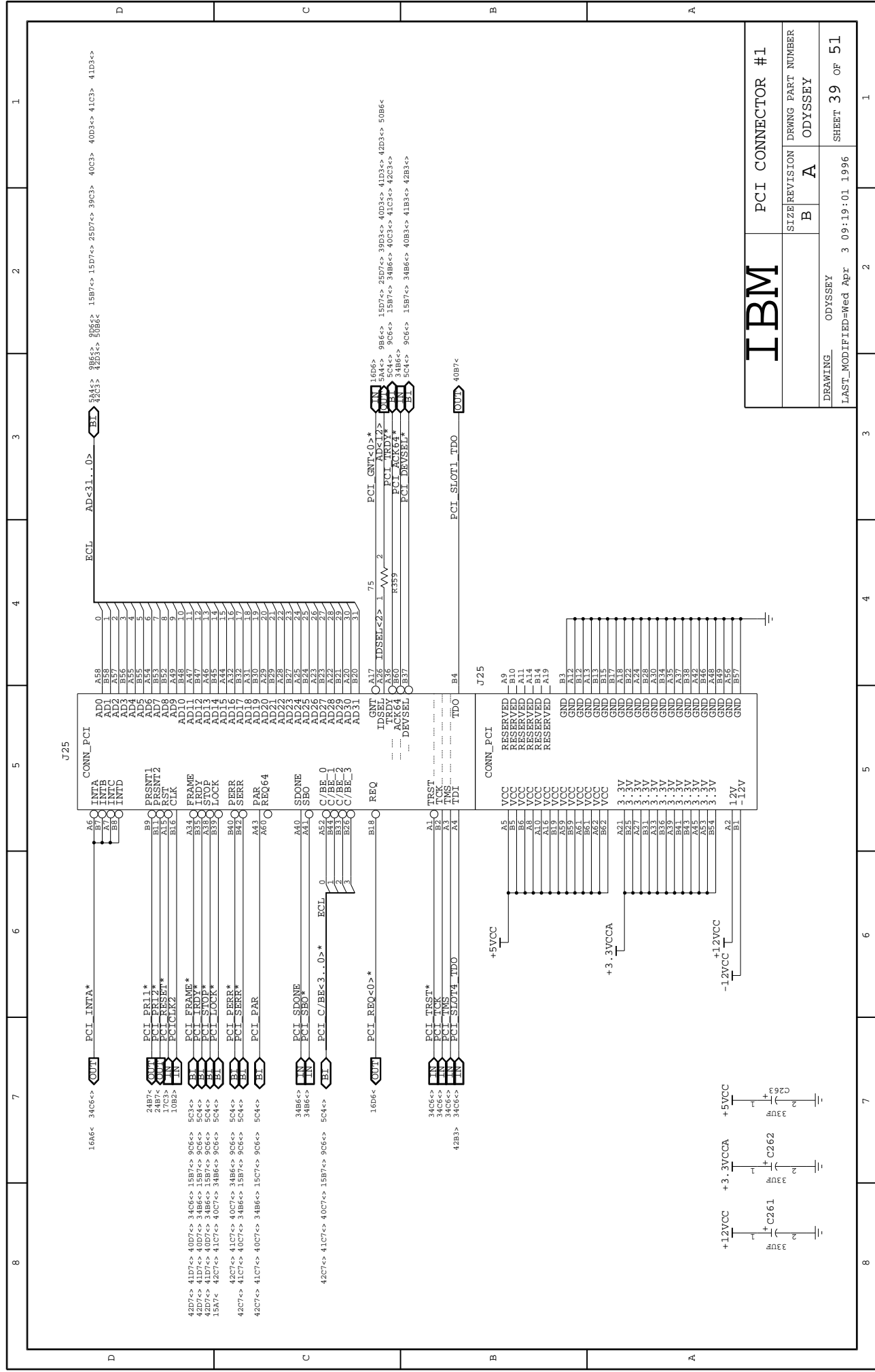
DRAWING	ODYSSEY	LAST_MODIFIED=Wed Apr 3 09:18:54 1996
SHEET 37	OF 51	

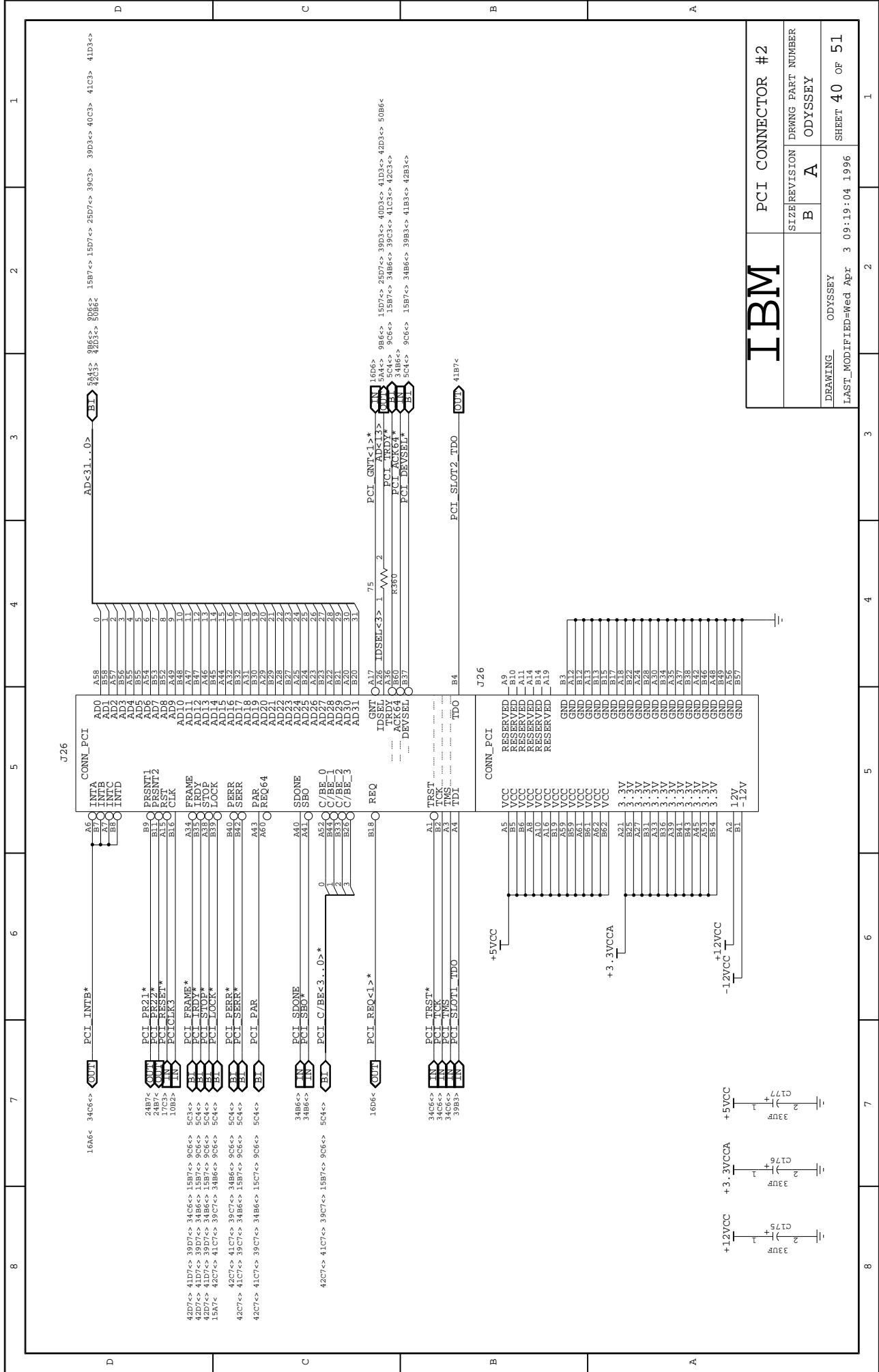


IBM

MEMORY SIMMS 3 & 4

SIZE/REVISION	DRAWING PART NUMBER
B A	ODYSSEY
DRAWING	ODYSSEY
LAST_MODIFIED=Wed Apr 3 09:18:57 1996	SHEET 38 OF 51





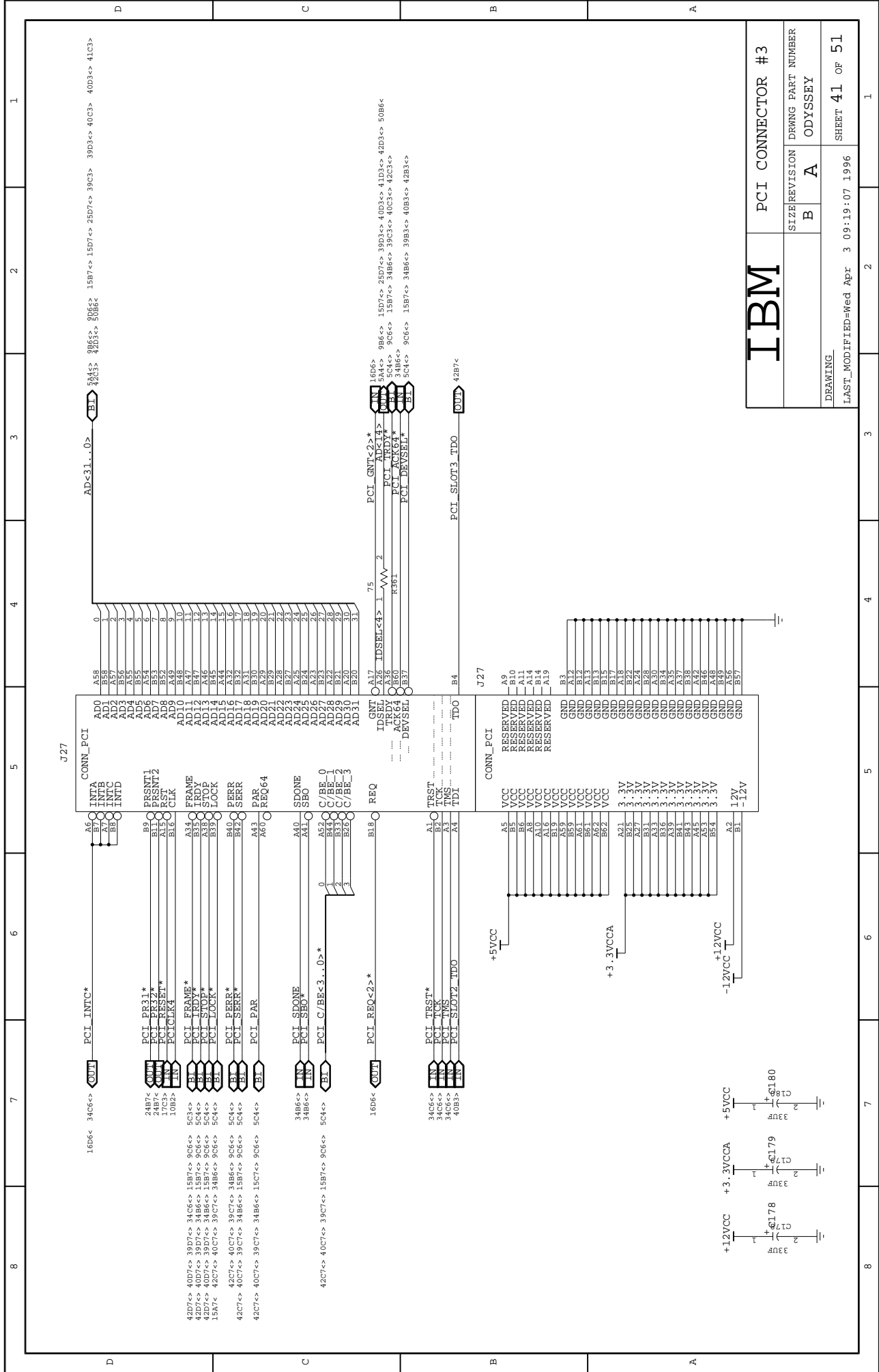
IBM

PCI CONNECTOR #2

SIZE/REVISION	DRAWING PART NUMBER
B	A

DRAWING	ODYSSEY	LAST MODIFIED=Wed Apr 3 09:19:04 1996

SHEET 40 OF 51

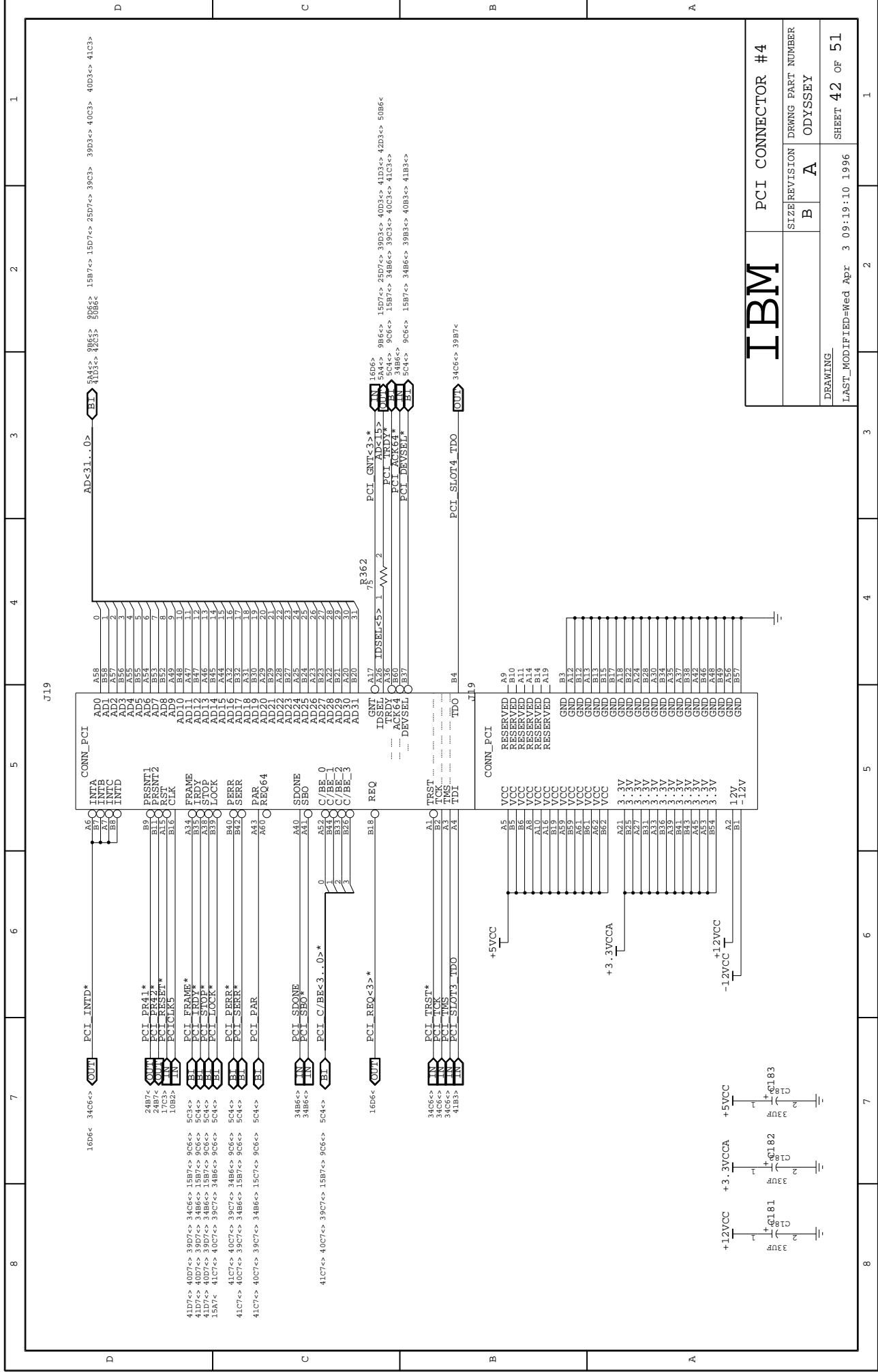


IBM

PCI CONNECTOR #3

SIZE/REVISION	DRAWING PART NUMBER
B	A

DRAWING	LAST MODIFIED=Wed Apr 3 09:19:07 1996
ODYSSEY	SHEET 41 OF 51



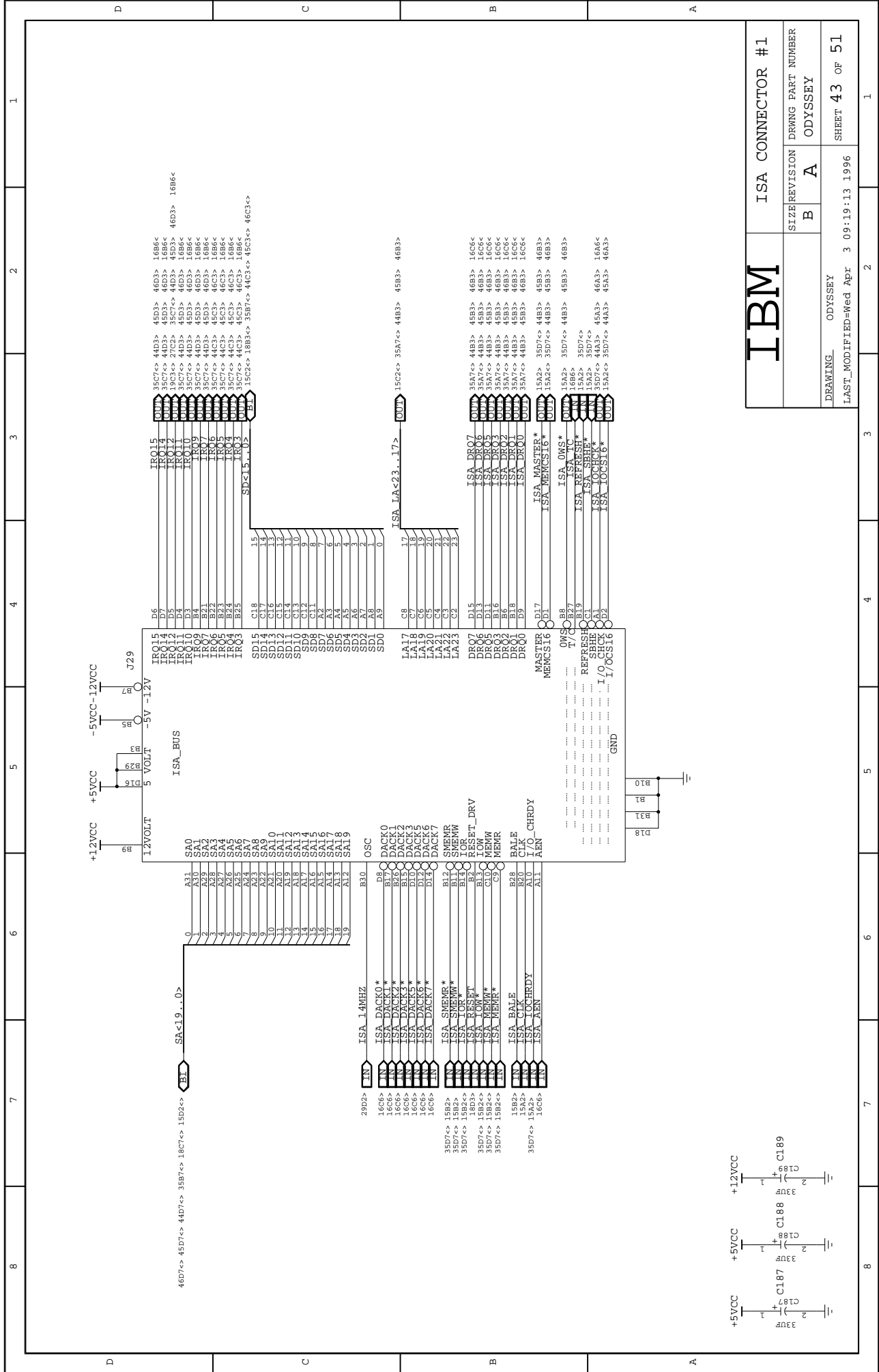
IBM

PCI CONNECTOR #4

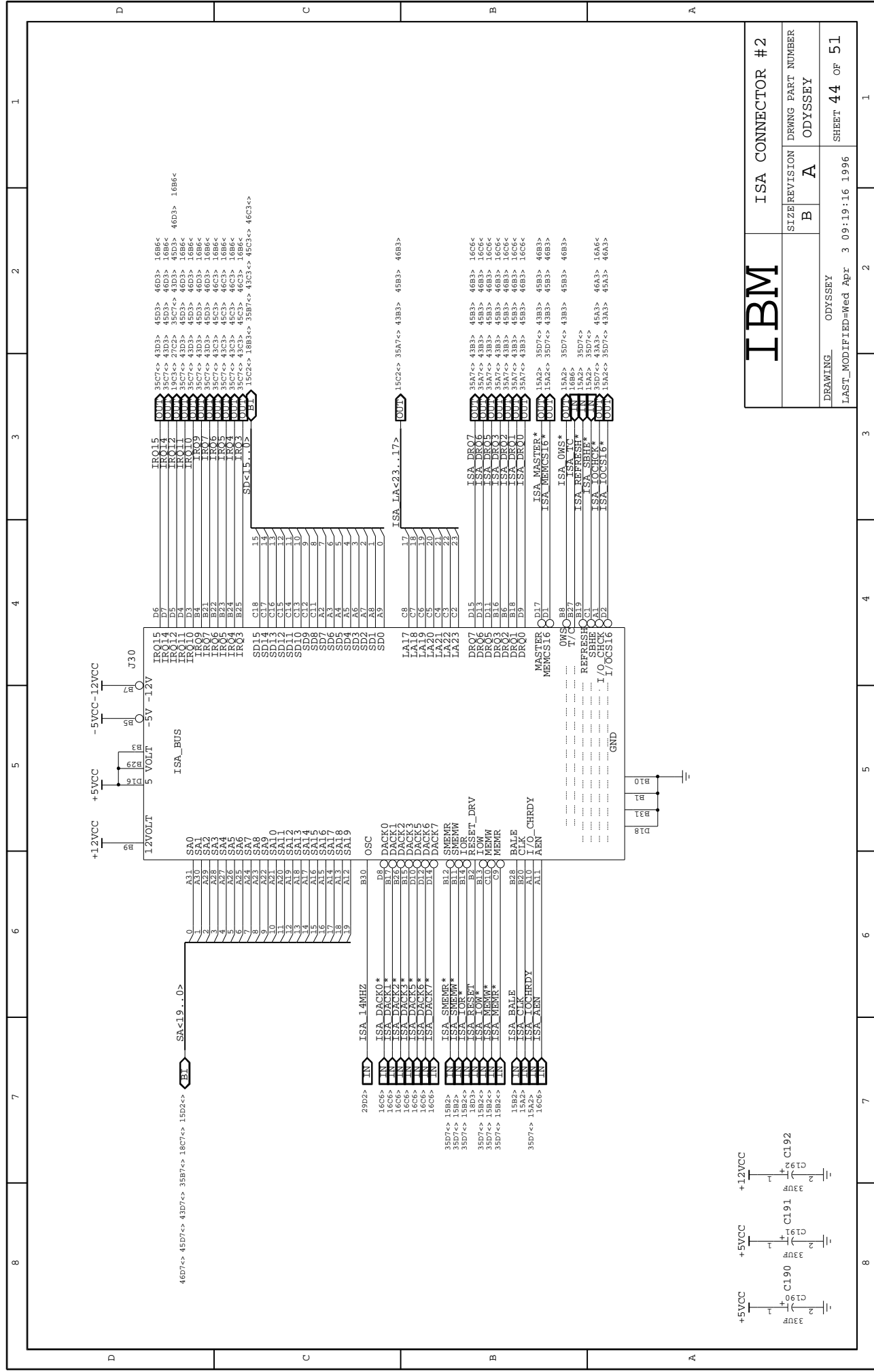
SIZE REVISION B A

DRAWING PART NUMBER ODYSSEY

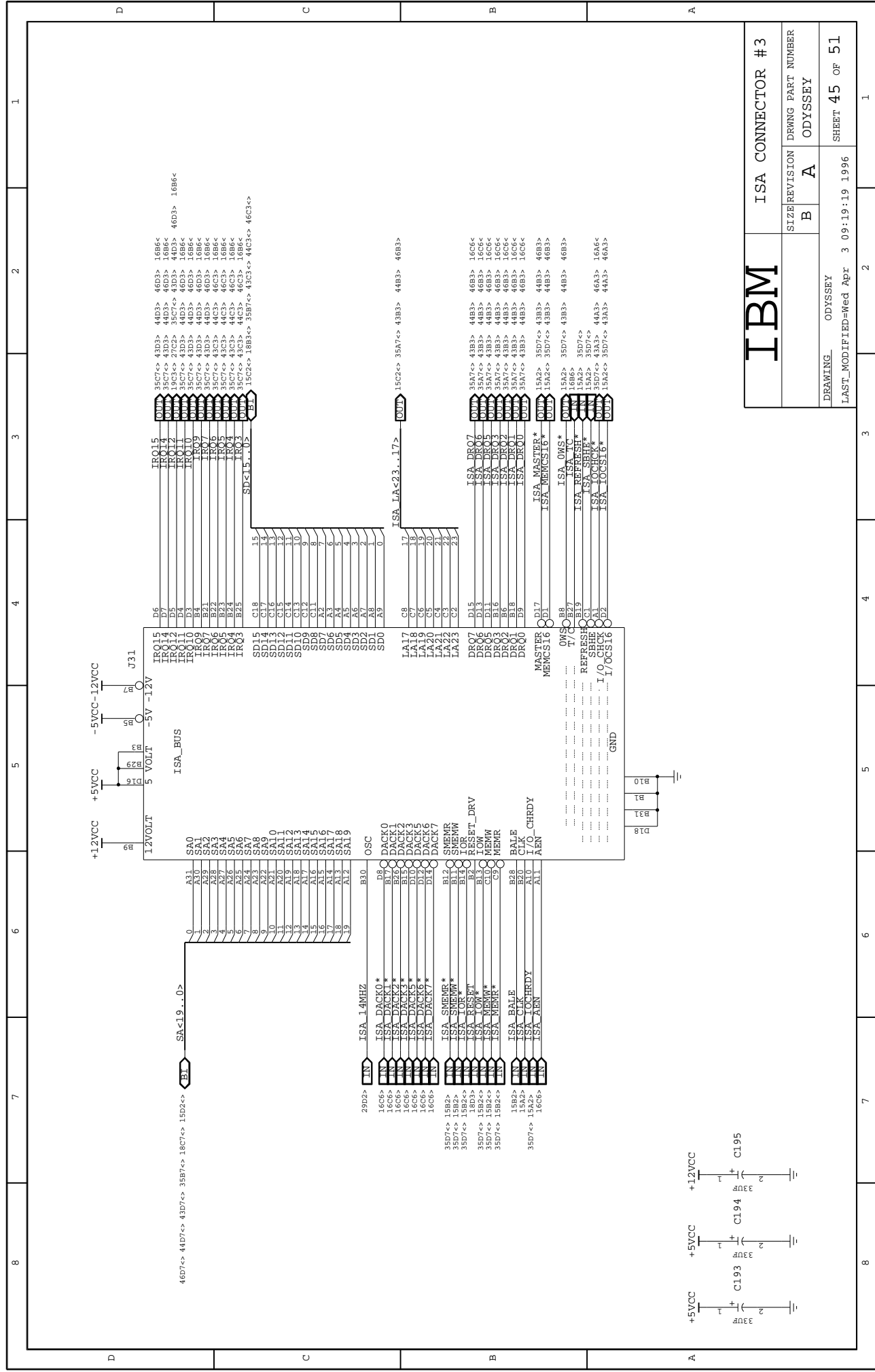
DRAWING LAST\_MODIFIED=Wed Apr 3 09:19:10 1996 SHEET 42 OF 51

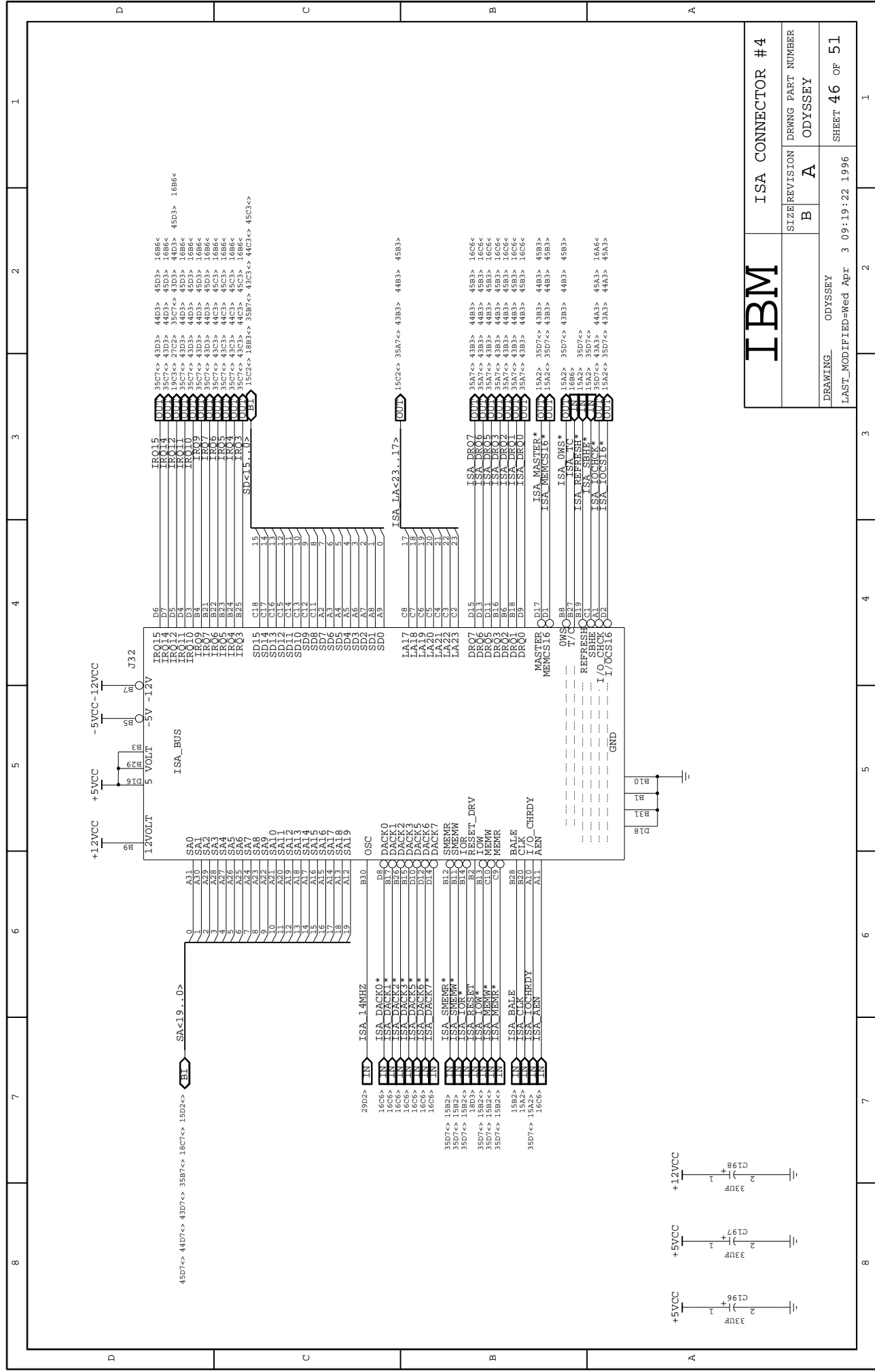


IBM		ISA CONNECTOR #1	
DRAWING	ODYSSEY	SIZE/REVISION	DWRNG PART NUMBER
		B A	ODYSSEY
		LAST_MODIFIED=Wed Apr 3 09:19:13 1996	
		SHEET 43 OF 51	



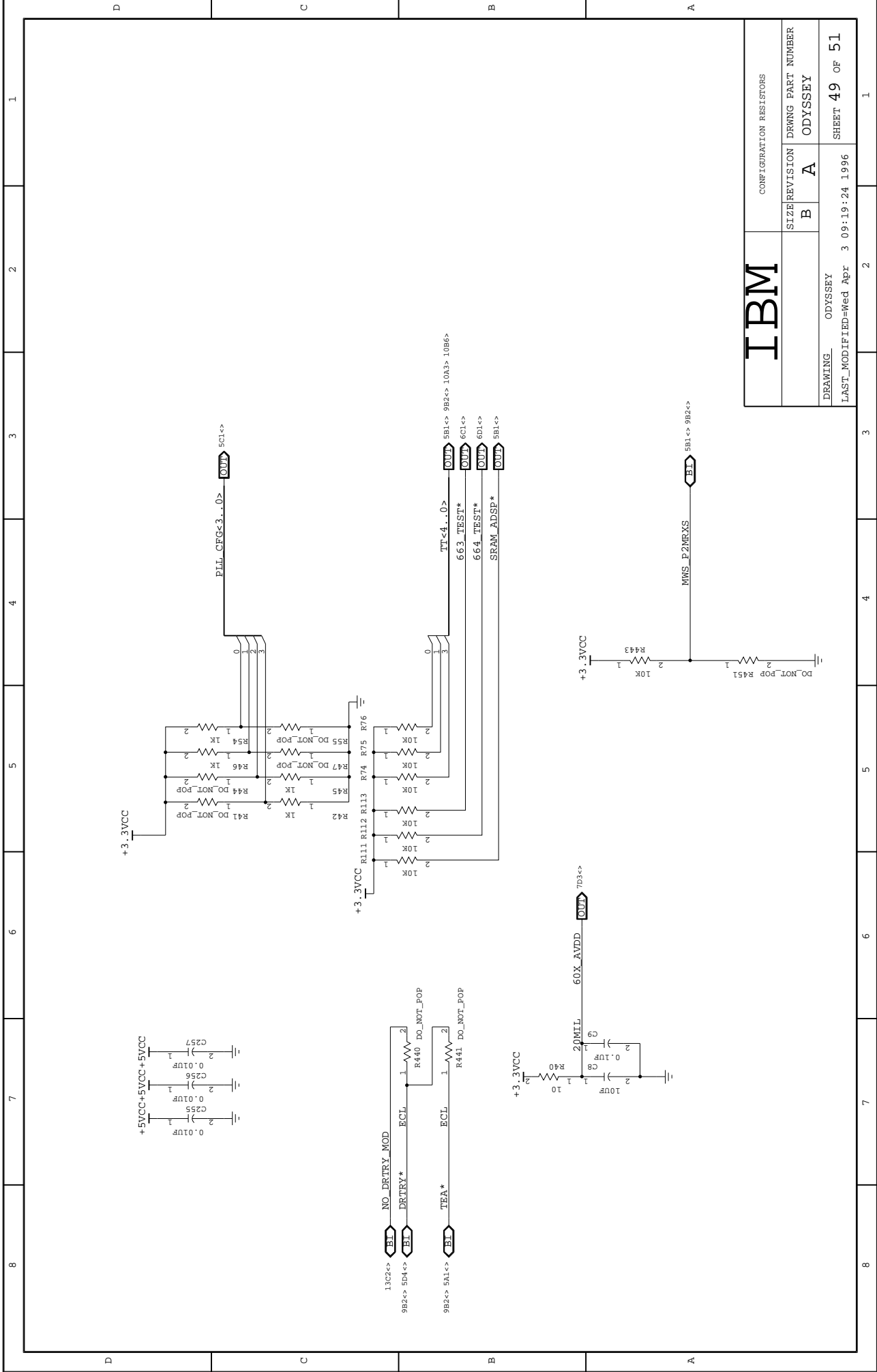






								8			7			6			5			4			3			2			1
D		C		B		A																							
BLANK																													
IBM																													
BLANK																													
DRAWING ODYSSEY														SIZE		REVISION		DRAWING PART NUMBER											
LAST_MODIFIED=Thu Feb 8 15:00:41 1996														B		A		ODYSSEY											
SHEET 47 OF 51																													

8								7		6		5		4		3		2		1							
D																											
C																											
B																											
A																											
BLANK																		IBM									
																		BLANK		DRAWING		SIZE		REVISION		DRAWING PART NUMBER	
																						B		A		ODYSSEY	
																		ODYSSEY		LAST_MODIFIED=Mon Jan 29 16:05:09 1996		SHEET 48 OF 51					
8								7		6		5		4		3		2		1							



IBM

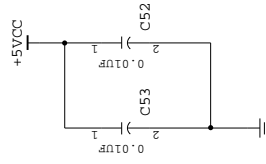
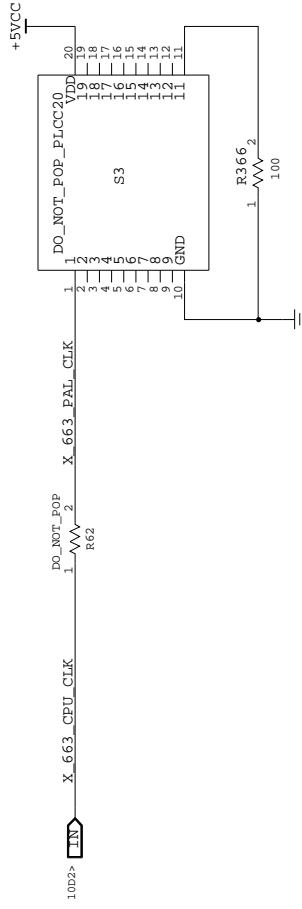
CONFIGURATION RESISTORS

SIZE REVISION B A

DRAWING PART NUMBER ODYSSEY

DRAWING ODYSSEY  
LAST MODIFIED=Wed Apr 3 09:19:24 1996  
SHEET 49 OF 51





IBM	SPARE PAL	
	SIZE	REVISION
	B	A
DRAWING	ODYSSEY	
LAST MODIFIED-Wed Apr 3 09:19:29 1996		SHEET 51 OF 51

[illegible]



[illegible]

[illegible]

8												7												6												5												4												3												2												1																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																			
D												C												B												A																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																								</											

8										7										6										5										4										3										2										1																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
R355 RESISTOR 35A5 R356 RESISTOR 35A5 R357 RESISTOR 35A5 R358 RESISTOR 35A5 R359 RESISTOR 39C4 R360 RESISTOR 40C4 R361 RESISTOR 41C4 R362 RESISTOR 42C4 R364 RESISTOR 19B6 R365 RESISTOR 50B5 R366 RESISTOR 51B4 R367 RESISTOR 50B3 R370 RESISTOR 10A4 R372 RESISTOR 10A4 R436 RESISTOR 30C7 R437 RESISTOR 30B6 R440 RESISTOR 49B7 R441 RESISTOR 49B7 R443 RESISTOR 49A4 R451 RESISTOR 49A4 R470 RESISTOR 50C6 R471 RESISTOR 50C6 R472 RESISTOR 50B3 R473 RESISTOR 50B3 R485 RESISTOR 23B5 R488 RESISTOR 22C4 R489 RESISTOR 22C4 R490 RESISTOR 22C4 R491 RESISTOR 22C3 R492 RESISTOR 22C3 R493 RESISTOR 22C3 R494 RESISTOR 22C3 R495 RESISTOR 22C3 R496 RESISTOR 13C3 R497 RESISTOR 13C3 R505 RESISTOR 31A5 R510 RESISTOR 13B3 R511 RESISTOR 10A4 R512 RESISTOR 10A3 R513 RESISTOR 10B7 S1 PAL_20 50C4 S3 DO-NOT-POP_PLCC20 51C4 U1 F08 13B3 13B3 13C3 36B2 U2 HCT14 18D4 18D5 28C3 28C5 28D3 28D4 U3 06 13C3 U4 171431REG 30B6 U5 171431REG 30B6 U7 5832782B 16C4 16C4 16D4 17B5 U10 F244 17D5 18D5 U11 F244 18C5 18D5 U12 F245 18B5 U13 F11 18A4 18A5 36B2 U16 F125 28B5 36C2 U17 F244 22C4 U18 F244 23B4 23C4 U19 F244 24D3 U20 F244 24B3 24C3 U21 F74 29A5 U22 0642H 17B4 17B5 U28 06 16C4 16B6 16B8 27B5 27B6 U29 F08 28C5 36C2 U40 F244 22A4 22B4 X1 PART 7A3 X2 EPM5130FP 19A4 X3 29F040ROM_MECH 25A4 X4 PART 7A3 X5 PAL_MECH 50A4										Y1 OSCLR 29D4 Y2 CRYSTL 29C4 Y2A XTAL 29C5 Y3 CRYSTAL_2PIN 29B4 Y3A XTAL 29B5 Y4 CRYSTL 29B4 Y4A XTAL 29B5 Y5 OSCLR 29A6																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					

## **Appendix G**

### **Data Sheets**

This section contains data sheets on selected components. IBM does not recommend the use of components from any particular manufacturer (other than IBM). The following data sheets are included as reference material only. Third party material is reproduced and included by permission of the copyright owner.



---

## Features

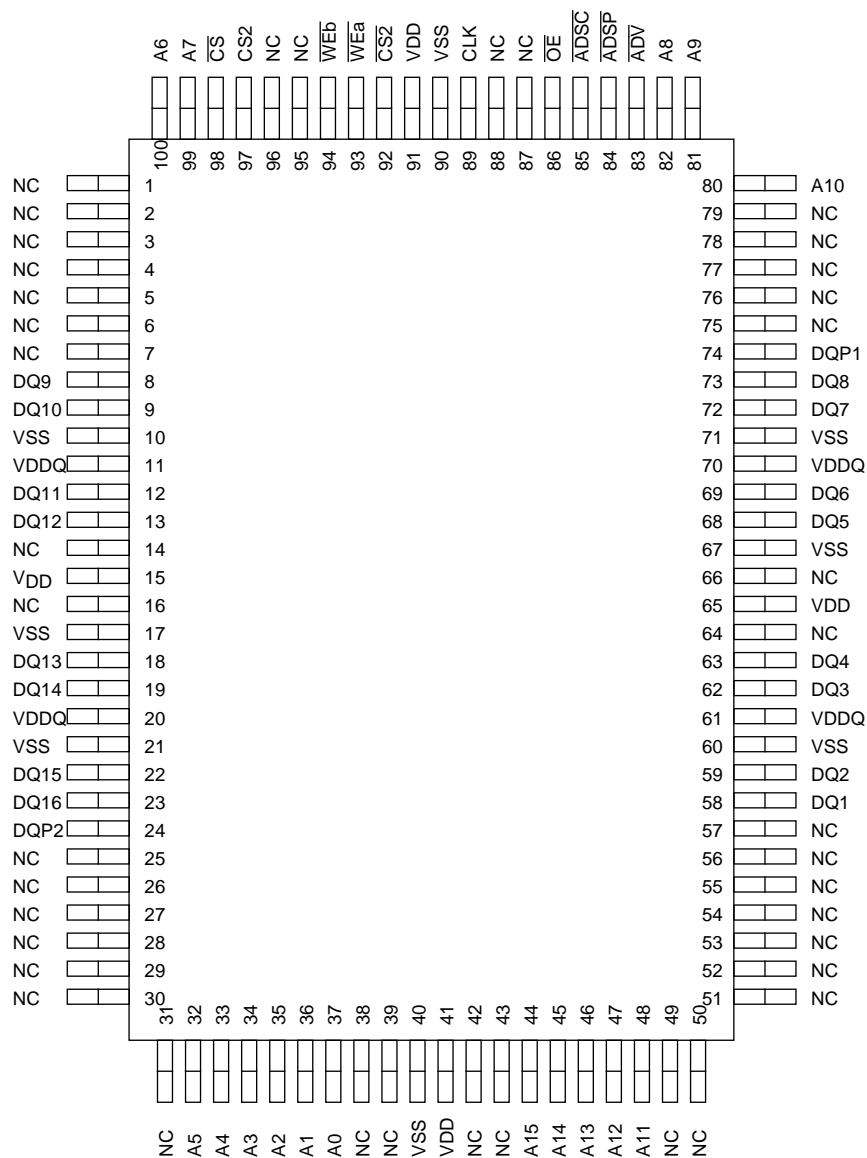
- 64K x 18 Synchronous Burst Mode SRAM
- 0.5 $\mu$  CMOS Technology
- Synchronous Burst Mode of Operation Compatible with PowerPC™ Processors
- LVTTL I/O Compatible with Common I/O
- Single +3.3 V  $\pm$  5% Power Supply and Ground
- Registered Addresses, Data Ins and Control Signals
- 5 V Tolerant I/O
- Asynchronous Output Enable
- Self-Timed Write Operation and Byte Write Capability
- Low Power Dissipation
  - 1.1 W Active at 83MHz
  - 90 mW Standby
- 100 Pin Thin Quad Flat Pack

---

## Description

IBM Microelectronics 1M SRAM is a Synchronous Burstable, high performance CMOS Static RAM that is versatile, wide I/O, and achieves 8 nsec access. A single clock is used to initiate the read/write operation and all internal operations are self-timed. At the rising edge of the Clock, all Addresses, Data Ins and Control Signals are registered internally. Burst mode operation, compatible with PowerPC™ Processor's sequence, is accomplished by integrating input registers, internal 2-bit burst counter and high speed SRAM in a single chip. Burst reads are initiated with either  $\overline{\text{ADSP}}$  or  $\overline{\text{ADSC}}$  being LOW with a valid address during the rising edge of clock. Data from this address plus the three subsequent addresses will be output. The chip is operated with a single +3.3 V power supply and is compatible with LVTTL I/O interfaces.

## X18 TQFP Pin Array Layout

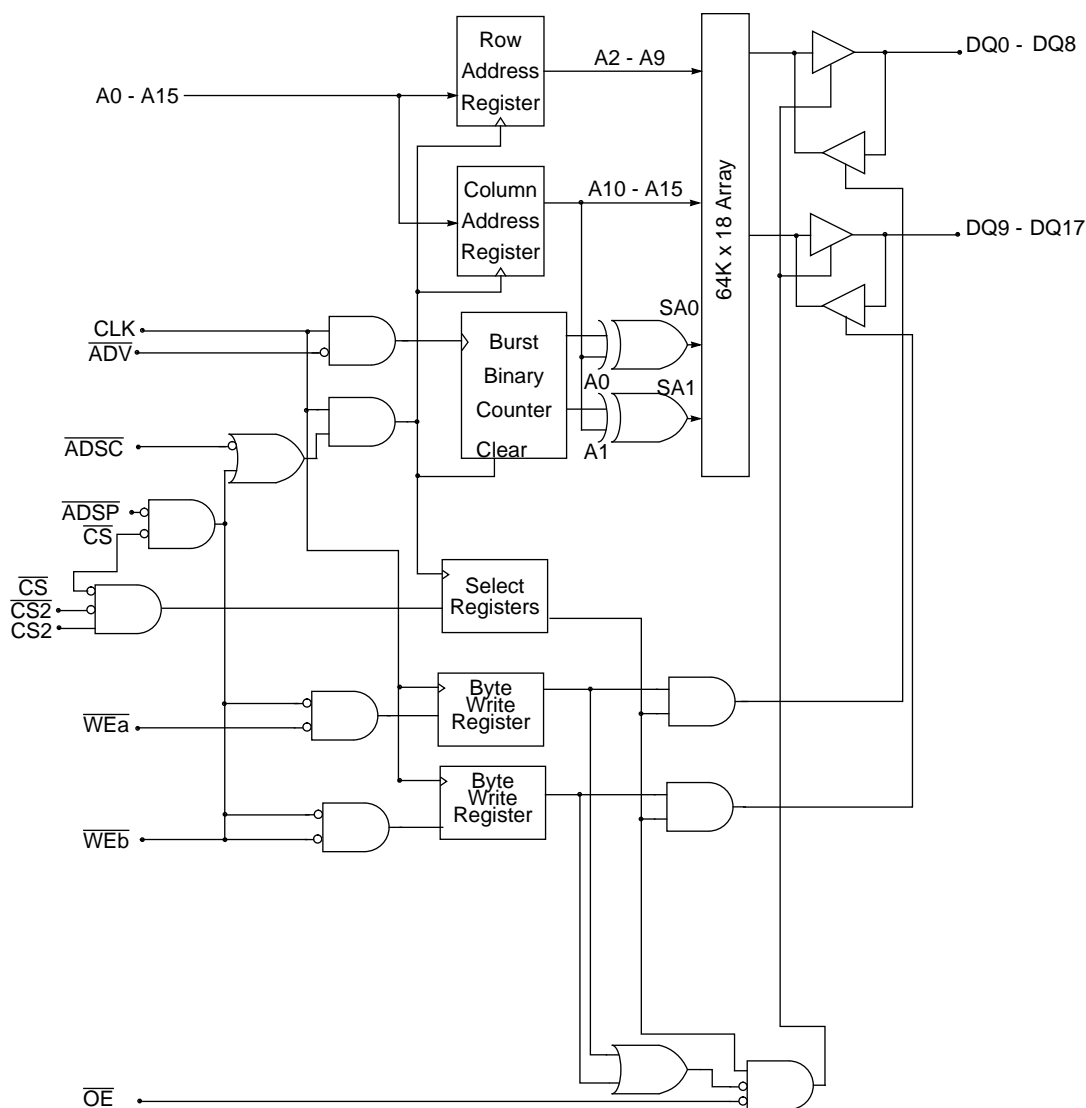


## Pin Description

A0-A15	Address input	$\overline{\text{ADSP}}$	Address Status Processor
DQa - DQb	Data Input/Output (1-8 , 9-16)	$\overline{\text{ADSC}}$	Address status controller
CLK	Clock	$\overline{\text{ADV}}$	Burst Advance Control
$\overline{\text{WEa}}$	Write Enable, Byte a (1 to 8 & DQP1)	$\overline{\text{CS}}$	$\overline{\text{ADSP}}$ - Gated Chip Select
$\overline{\text{WEb}}$	Write Enable, Byte b (9 to 16 & DQP2)	$\text{V}_{\text{DD}}$	Power Supply (+3.3V)
$\overline{\text{OE}}$	Output Enable	$\text{V}_{\text{SS}}$	Ground
$\overline{\text{CS2}}, \text{CS2}$	Chip Selects	$\text{V}_{\text{DDQ}}$	Output Power Supply (+3.3V)
DQP1,DQP2	Parity bits for byte a, and byte b.	NC	No Connect



## Block Diagram



## Ordering Information

Part Number	Organization	Speed	Leads	Notes
IBM041814PQK-8	64K x 18	8 ns Access / 12 ns Cycle	100 pin TQFP	
IBM041814PQK-9	64K x 18	9 ns Access / 12 ns Cycle	100 pin TQFP	
IBM041814PQK-10	64K x 18	10 ns Access / 12 ns Cycle	100 pin TQFP	
IBM041814PQK-11	64K x 18	11 ns Access / 12 ns Cycle	100 pin TQFP	

## Burst SRAM Clock Truth Table

CLK	$\overline{CS}2$	CS2	$\overline{CS}$	$\overline{ADSP}$	$\overline{ADSC}$	$\overline{ADV}$	$\overline{WE}$	$\overline{OE}$	DQ	Operation
L→H	H	X	L	L	X	X	X	X	High-Z	Deselected Cycle
L→H	X	L	L	L	X	X	X	X	High-Z	Deselected Cycle
L→H	H	X	X	X	L	X	X	X	High-Z	Deselected Cycle
L→H	X	L	X	X	L	X	X	X	High-Z	Deselected Cycle
L→H	L	H	L	L	X	X	X	L	Q	Read from External Address, Begin Burst
L→H	L	H	L	L	X	X	X	H	High-Z	Read from External Address, Begin Burst
L→H	L	H	L	H	L	X	H	L	Q	Read from External Address, Begin Burst
L→H	L	H	L	H	L	X	L	X	D	Write to External Address, Begin Burst
L→H	X	X	X	H	H	L	H	L	Q	Read from next Add., Continue Burst
L→H	X	X	X	H	H	L	L	X	D	Write to next Add., Continue Burst
L→H	X	X	X	H	H	H	H	L	Q	Read from Current Add., Suspend Burst
L→H	X	X	X	H	H	H	L	X	D	Write to Current Add., Suspend Burst
L→H	X	X	H	X	L	X	X	X	High-Z	Deselect Cycle
L→H	X	X	H	X	H	L	H	L	Q	Read from next Add., Continue Burst
L→H	X	X	H	X	H	L	L	X	D	Write to next Add., Continue Burst
L→H	X	X	H	X	H	H	H	L	Q	Read from current Add., Suspend Burst
L→H	X	X	H	X	H	H	L	X	D	Write to current Add., Suspend Burst

1. For a write operation preceded by a read cycle,  $\overline{OE}$  must be HIGH early enough to allow Input Data Setup, and must be kept HIGH through Input Data Hold Time.
2.  $\overline{WE}$  refers to  $\overline{WEa}$ ,  $\overline{WEb}$ .
3.  $\overline{ADSP}$  is gated by  $\overline{CS}$ , and  $\overline{CS}$  is used to block  $\overline{ADSP}$  when  $\overline{CS} = V_{IH}$ , as required in applications using Processor Address Pipelining.
4. All Addresses, Data In and Control signals are registered on the rising edge of CLK.

## Burst Sequence Truth Table

External Address	A15-A2	(A1,A0)				Notes
		(0,0)	(0,1)	(1,0)	(1,1)	
1st Access	A15-A2	(0,0)	(0,1)	(1,0)	(1,1)	
2nd Access	A15-A2	(0,1)	(1,0)	(1,1)	(0,0)	
3rd Access	A15-A2	(1,0)	(1,1)	(0,0)	(0,1)	
4th Access	A15-A2	(1,1)	(0,0)	(0,1)	(1,0)	

## Write Enable Truth Table

$\overline{WEa}$	$\overline{WEb}$	Byte Written	Notes
H	H	Read All Bytes	
L	L	Write All Bytes	
L	H	Write Byte A ( $D_{IN} 0 - 8$ )	
H	L	Write Byte B ( $D_{IN} 9 - 17$ )	

## Absolute Maximum Ratings

Parameter	Symbol	Rating	Units	Notes
Power Supply Voltage	$V_{DD}$	-0.5 to 4.6	V	1
Input Voltage	$V_{IN}$	-0.5 to 6.0	V	1
Output Voltage	$V_{OUT}$	-0.5 to $V_{DD}+0.5$	V	1
Operating Temperature	$T_{OPR}$	0 to +70	°C	1
Storage Temperature	$T_{STG}$	-55 to +125	°C	1
Power Dissipation	$P_D$	2.0	W	1
Short Circuit Output Current	$I_{OUT}$	50	mA	1

1. Stresses greater than those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect reliability.

## Recommended DC Operating Conditions ( $T_A=0$ to $70^\circ\text{C}$ )

Parameter	Symbol	Min.	Typ.	Max.	Units	Notes
Supply Voltage	$V_{DD}$	3.135	3.3	3.465	V	1, 4
Input High Voltage	$V_{IH}$	2.2	—	5.5	V	1, 2, 4
Input Low Voltage	$V_{IL}$	-0.3	—	0.8	V	1, 3, 4
Output Current	$I_{OUT}$	—	5	8	mA	4

1. All voltages referenced to  $V_{SS}$ . All  $V_{DD}$  and  $V_{SS}$  pins must be connected.
2.  $V_{IH}(\text{Max})_{DC} = 5.5$  V,  $V_{IH}(\text{Max})_{AC} = 6.0$  V (pulse width  $\leq 4.0\text{ns}$ ).
3.  $V_{IL}(\text{Min})_{DC} = -0.3$  V,  $V_{IL}(\text{Min})_{AC} = -1.5$  V (pulse width  $\leq 4.0\text{ns}$ ).
4. Input Voltage levels are tested to the following DC conditions: 1 microsecond cycle and 200 nanosecond set-up and hold times.

## Capacitance ( $T_A=0$ to $+70^\circ\text{C}$ , $V_{DD}=3.3\text{V} \pm 5\%$ , $f=1\text{MHz}$ )

Parameter	Symbol	Test Condition	Max	Units	Notes
Input Capacitance	$C_{IN}$	$V_{IN} = 0\text{V}$	5	pF	
Data I/O Capacitance (DQ0-DQ17)	$C_{OUT}$	$V_{OUT} = 0\text{V}$	5	pF	

## DC Electrical Characteristics ( $T_A = 0$ to $+70^\circ\text{C}$ , $V_{DD} = 3.3\text{V} \pm 5\%$ )

Parameter	Symbol	Min.	Max.	Units	Notes
Operating Current Average Power Supply Operating Current ( $\overline{OE} = V_{IH}$ , $I_{OUT} = 0$ )	$I_{DD12}$	—	300	mA	2, 3
Standby Current Power Supply Standby Current ( $CS2 = V_{IH}$ or $CS2 = V_{IL}$ or $\overline{CS} = V_{IL}$ All other inputs = $V_{IH}$ or $V_{IL}$ , $I_{OUT} = 0$ Clock @ 83MHz)	$I_{SB}$	—	25	mA	1, 3
Input Leakage Current Input Leakage Current, any input ( $V_{IN} = 0$ & $V_{DD}$ )	$I_{LI}$	—	+1	$\mu\text{A}$	4
Output Leakage Current ( $V_{OUT} = 0$ & $V_{DD}$ , $\overline{OE} = V_{IH}$ )	$I_{LO}$	—	+1	$\mu\text{A}$	
Output High Level Output "H" Level Voltage ( $I_{OH} = -8\text{mA}$ @ 2.4V)	$V_{OH}$	2.4	—	V	
Output Low Level Output "L" Level Voltage ( $I_{OL} = +8\text{mA}$ @ 0.4V)	$V_{OL}$	—	0.4	V	
1. $I_{SB}$ = Stand-by Current. 2. $I_{DD}$ = Selected Current. 3. $I_{OUT}$ = Chip Output Current. 4. The input leakage current for 5.5V input is 200 $\mu\text{A}$ for Clk, Chip Selects, and Output Enable. Other inputs have 100 $\mu\text{A}$ of leakage current at 5.5V					

## AC Test Conditions ( $T_A = 0$ to $+70^\circ\text{C}$ , $V_{DD} = 3.3\text{V} \pm 5\%$ )

Parameter	Symbol	Conditions	Units	Notes
Input Pulse High Level	$V_{IH}$	3.0	V	
Input Pulse Low Level	$V_{IL}$	0.0	V	
Input Rise Time	$T_R$	2.0	ns	
Input Fall Time	$T_F$	2.0	ns	
Input and Output Timing Reference Level		1.5	V	
Output Load Conditions				1
1. See AC Test Loading figure on page 8.				

## AC Characteristics (T<sub>A</sub>=0 to +70°C, V<sub>DD</sub>=3.3V ± 5%, Units in nsec)

Parameter	Symbol	-8		-9		-10		-11		Notes
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Cycle Time	t <sub>CYCLE</sub>	12.0	—	12.0	—	12.0	—	12.0	—	
Clock Pulse High	t <sub>CH</sub>	3.0	—	3.0	—	3.0	—	3.0	—	
Clock Pulse Low	t <sub>CL</sub>	3.0	—	3.0	—	3.0	—	3.0	—	
Clock to Output Valid	t <sub>CQ</sub>	—	8.0	—	9.0	—	10.0	—	11.0	3
Address Status Controller Setup Time	t <sub>ADSCS</sub>	2.5	—	2.5	—	2.5	—	2.5	—	
Address Status Controller Hold Time	t <sub>ADSCH</sub>	0.5	—	0.5	—	0.5	—	0.5	—	
Address Status Processor Setup Time	t <sub>ADSPS</sub>	2.5	—	2.5	—	2.5	—	2.5	—	
Address Status Processor Hold Time	t <sub>ADSPH</sub>	0.5	—	0.5	—	0.5	—	0.5	—	
Advance Setup Time	t <sub>ADVS</sub>	2.5	—	2.5	—	2.5	—	2.5	—	
Advance Hold Time	t <sub>ADVH</sub>	0.5	—	0.5	—	0.5	—	0.5	—	
Address Setup Time	t <sub>AS</sub>	2.5	—	2.5	—	2.5	—	2.5	—	
Address Hold Time	t <sub>AH</sub>	0.5	—	0.5	—	0.5	—	0.5	—	
Chip Selects Setup Time	t <sub>CSS</sub>	2.5	—	2.5	—	2.5	—	2.5	—	
Chip Selects Hold Time	t <sub>CSH</sub>	0.5	—	0.5	—	0.5	—	0.5	—	
Write Enables Setup Time	t <sub>WES</sub>	2.5	—	2.5	—	2.5	—	2.5	—	
Write Enables Hold Time	t <sub>WEH</sub>	0.5	—	0.5	—	0.5	—	0.5	—	
Data In Setup Time	t <sub>DS</sub>	2.5	—	2.5	—	2.5	—	2.5	—	
Data In Hold Time	t <sub>DH</sub>	0.5	—	0.5	—	0.5	—	0.5	—	
Data Out Hold Time	t <sub>CQX</sub>	3.0	—	3.0	—	3.0	—	3.0	—	3
Clock High to Output High-Z	t <sub>CHZ</sub>	—	5.0	—	5.0	—	5.5	—	5.5	1, 2, 4
Clock High to Output Active	t <sub>CLZ</sub>	2.5	—	2.5	—	2.5	—	2.5	—	1, 2, 4
Output Enable to High-Z	t <sub>OHZ</sub>	2.0	5.0	2.0	5.5	2.0	6.0	2.0	6.5	1, 4
Output Enable to Low-Z	t <sub>OLZ</sub>	0.25	—	0.25	—	0.25	—	0.25	—	1, 4
Output Enable to Output Valid	t <sub>OQ</sub>	—	4.0	—	5.0	—	5.0	—	6.0	3

1. Transitions are measured ± 200 mV from steady state voltage.
2. At any given voltage and temperature, T<sub>CHZ</sub> (max) is always less than T<sub>CTZ</sub> (min) for a given device and from device to device. For any read cycle preceded by a write or deselect cycle, the data bus will transition glitch-free from High-Z to new RAM data.
3. See AC Test Loading figure 1 on page 8.
4. See AC Test Loading figure 2 on page 8.

## AC Test Loading

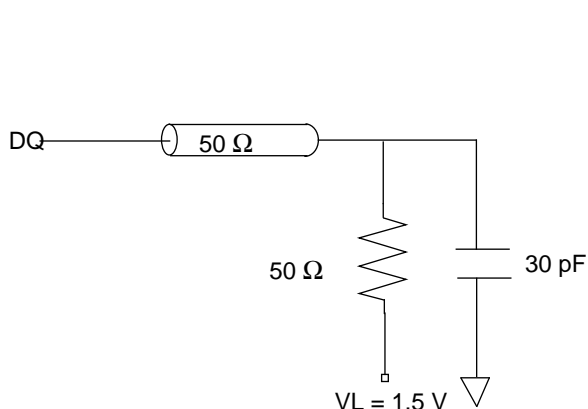


Fig. 1 Test Equivalent Load

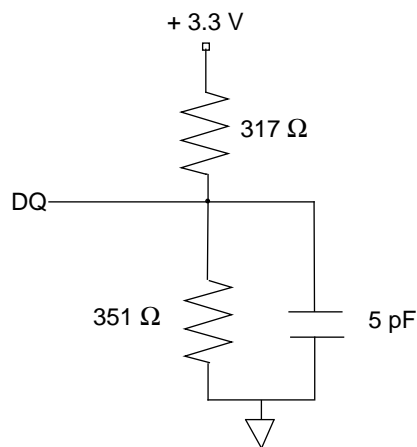
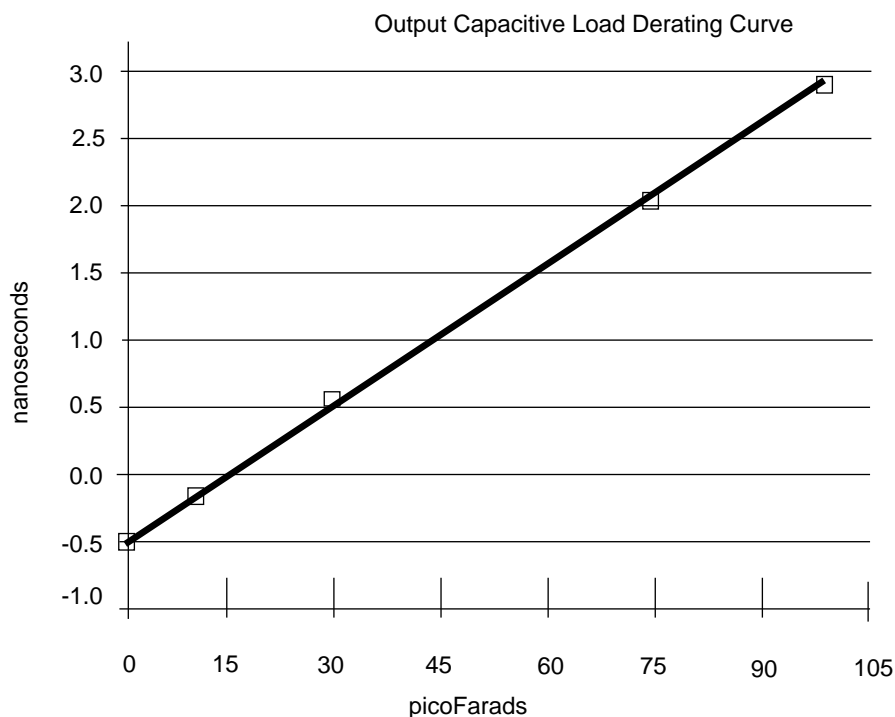
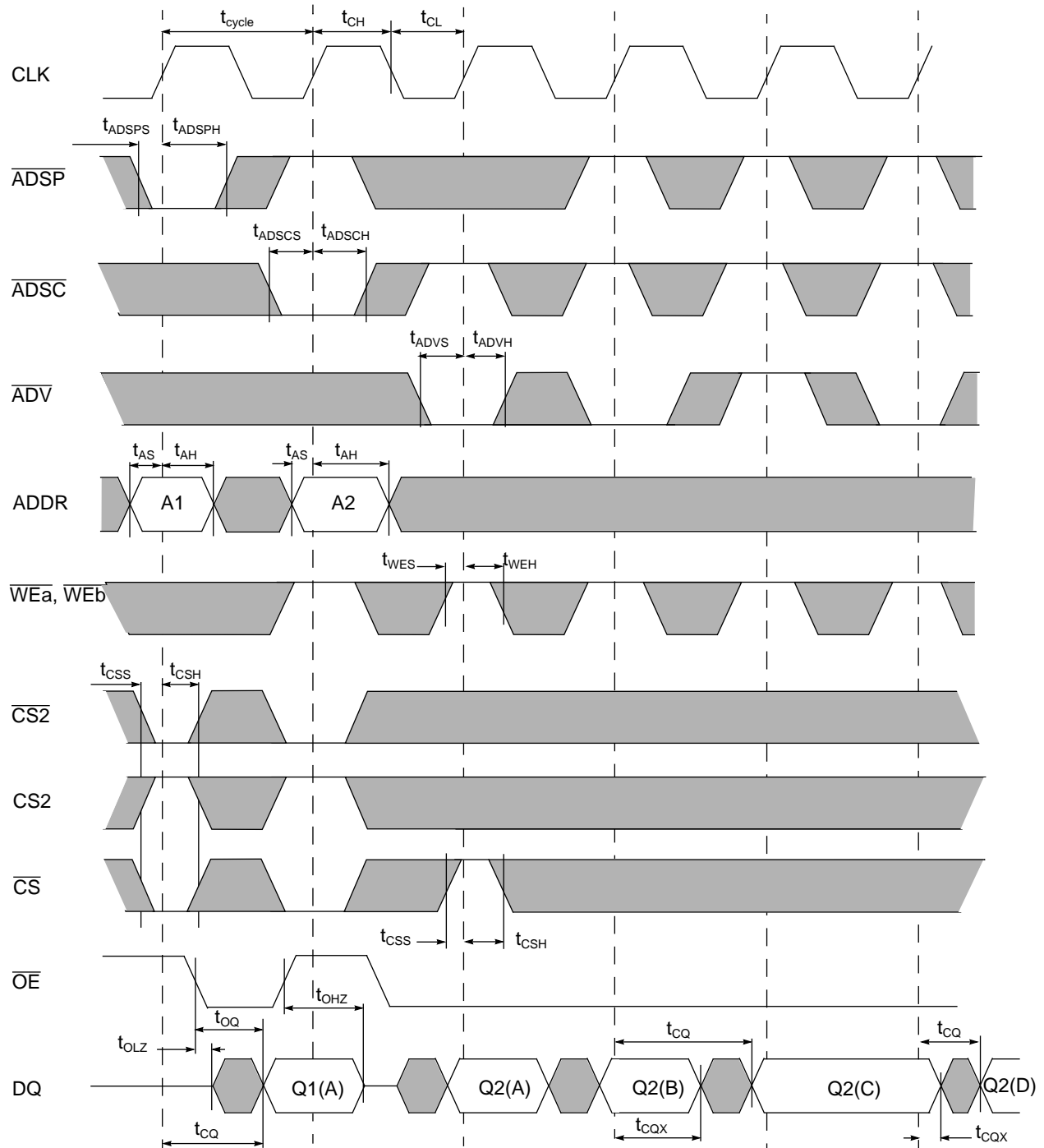


Fig. 2 Test Equivalent Load



The derating curve above is for a purely capacitive load on the output driver. For example, a part specified at 8ns access time will behave as though it has an 8.5 ns access time if a 30 pF load with no DC component was attached to the output driver. The access times guaranteed in the datasheets are based on a 50 ohm terminated test load. For unterminated loads the derating curve should be used. This curve is based on nominal process conditions with worst case parameters  $V_{CC} = 3.14\ \text{V}$ ,  $T_A = 70^\circ\ \text{C}$ .

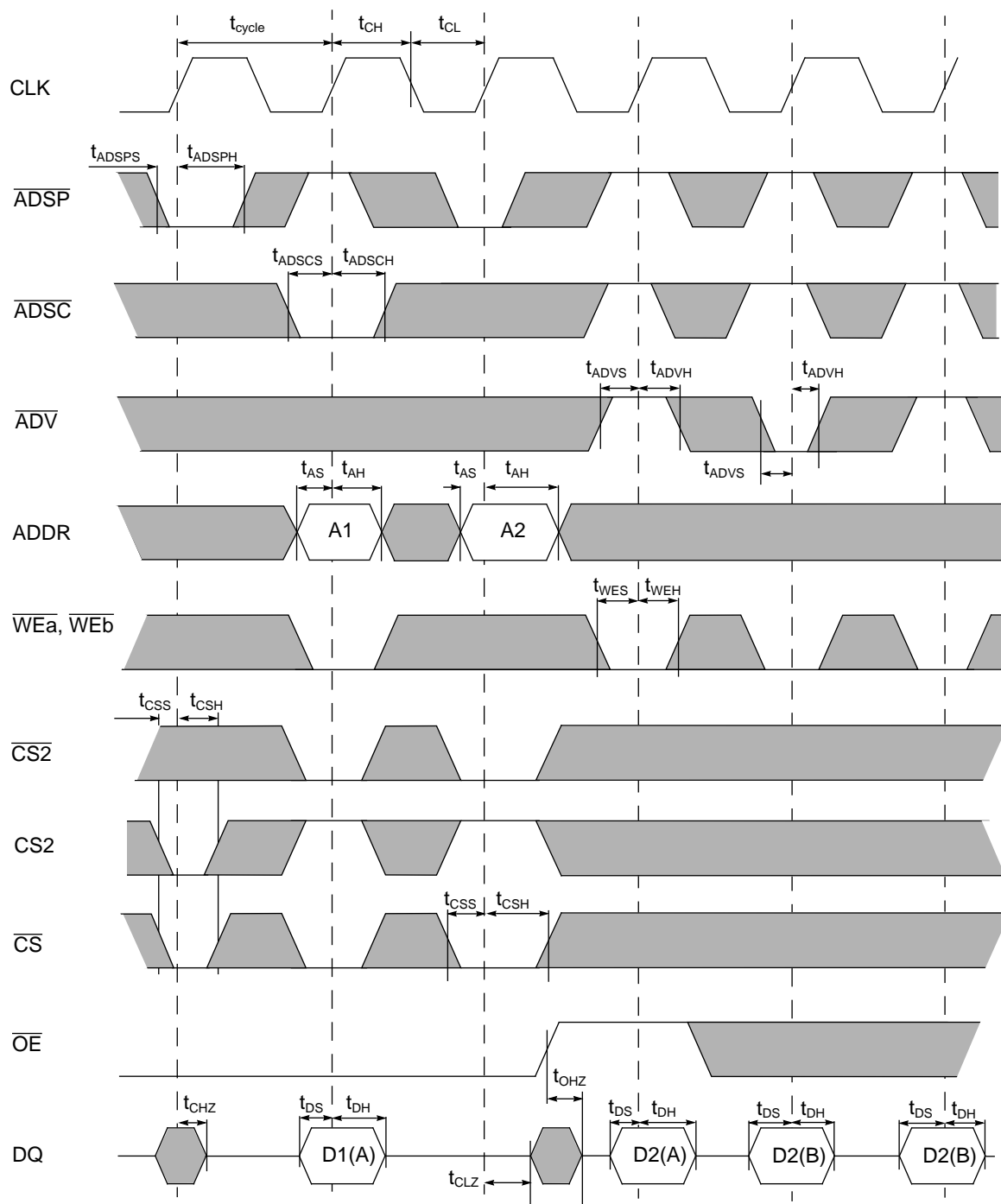
## Timing Diagram (Burst Read)



### Notes:

1. Q1(A) and Q2(A) refer to data written to address A1 and A2.
2. Q2(B), Q2(C) and Q2(D) refer to data written to subsequent internal burst counter addresses.

## Timing Diagram (Burst Write)

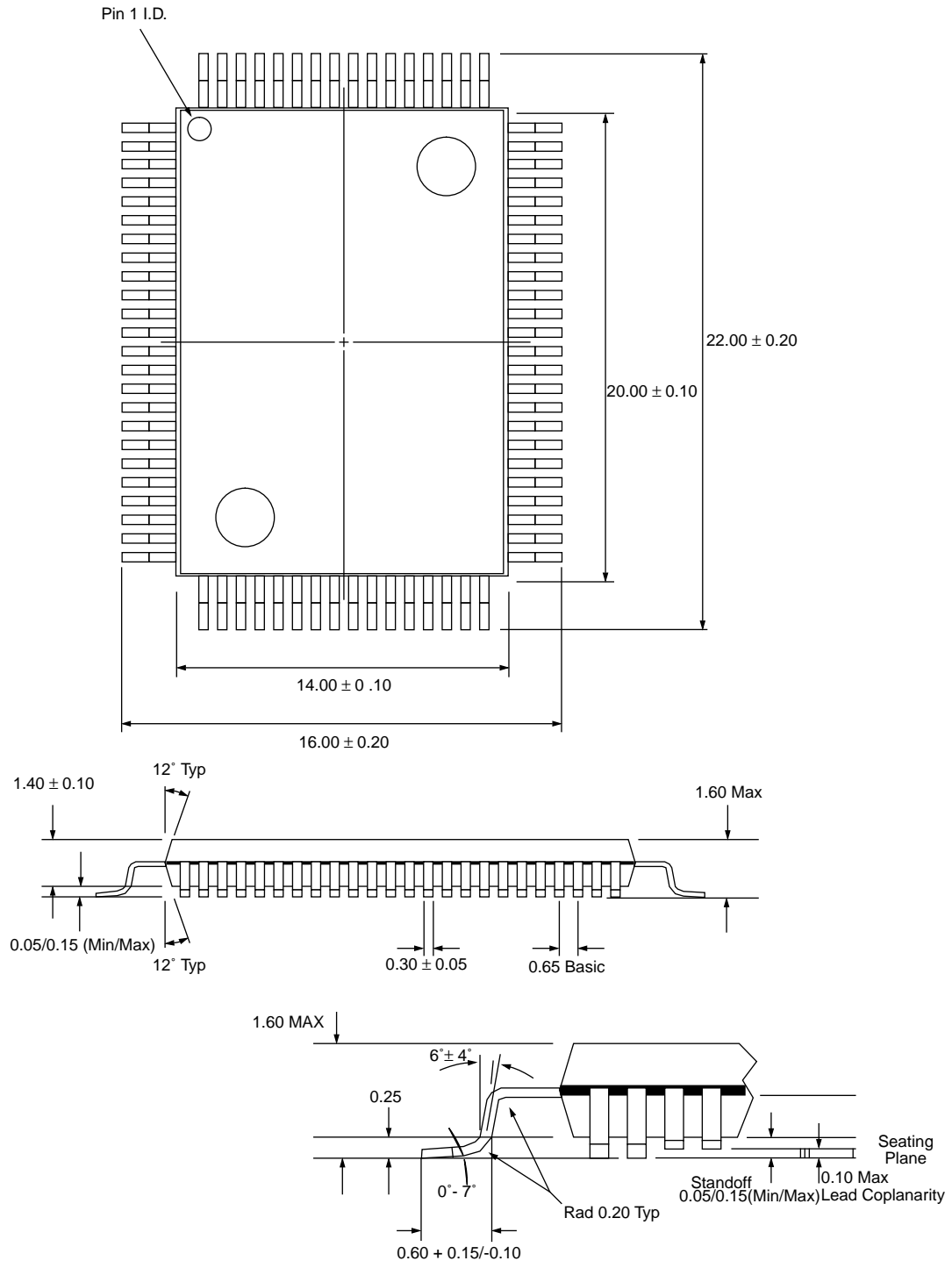


### Notes:

1. D1(A) and D2(A) refer to data written to address A1 and A2.
2. D2(B) refers to data written to a subsequent internal burst counter address.
3.  $\overline{\text{WEa}}$ ,  $\overline{\text{WEb}}$  are don't cares when ADSP is sampled LOW.



## 100 Pin TQFP Package Diagram



## Connect Compatibility for 64K x18 and Future 64K x 16 & 64K x 18

TQFP PIN #	Current Connections (x18)	Future Connections (x16 & x18)	Function
4,27,54,77	NC	V <sub>DDQ</sub>	Output Power Supply
5,26,55,76	NC	V <sub>SS</sub>	Ground
14	NC	Low or High, NC for most vendors but Low or High to comply to the JEDEC standard.	$\overline{FT}$ , FLow thru or Pipeline function, tie Low for flow thru, High for Pipeline
24	DQP2	NC or DQ in x18	Parity bit for second byte
31	NC	Low or High	$\overline{LBO}$ , Linear Burst Order, This pin must be tied low for linear(PowerPC), High for Interleave (Pentium)
64	NC	Low or High. Low allows normal operation.	ZZ , Asynchronous Sleep Mode, Tie to ground for normal function, V <sub>DDQ</sub> for sleep mode (Low power state)
74	DQP1	NC or DQ in x18	Parity bit for first byte
87	NC	Low or $\overline{BWE}$ , Tie low if function not used	Byte Write Enable, Allows individual bytes to be written.
88	NC	Low or $\overline{GW}$ , Tie High if function not used	Global Write Enable, Allows write of all bytes to occur with single pin.

The IBM041812PQK has the pins connected in the manner indicated in the Current Connections (x18) and is also JEDEC complaint. Future Connections refers to the evolution on the JEDEC standard for subsequent part numbers.

## Revision Log

Rev	Contents of Modification
7/95	Initial Release of the 64K x 18 (8/9/10/11) TQFP BURST MODE Application Spec.



© International Business Machines Corp.1996

Printed in the United States of America  
All rights reserved

IBM and the IBM logo are registered trademarks of the IBM Corporation.

This document may contain preliminary information and is subject to change by IBM without notice. IBM assumes no responsibility or liability for any use of the information contained herein. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of IBM or third parties. The products described in this document are not intended for use in implantation or other direct life support applications where malfunction may result in direct physical harm or injury to persons. **NO WARRANTIES OF ANY KIND, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, ARE OFFERED IN THIS DOCUMENT.**

For more information contact your IBM Microelectronics sales representative or visit us on World Wide Web at <http://www.chips.ibm.com>



Integrated Device Technology, Inc.

# BiCMOS StaticRAM 240K (16K x 15-BIT) CACHE-TAG RAM For PowerPC™ and RISC Processors

IDT71216

## FEATURES:

- 16K x 15 Configuration
  - 12 TAG Bits
  - 3 Separate I/O Status Bits (Valid, Dirty, Write Through)
- Match output uses Valid bit to qualify MATCH output
- High-Speed Address-to-Match comparison times
  - 8/9/10/12ns over commercial temperature range
- $\overline{TA}$  circuitry included inside the Cache-Tag for highest speed operation
- Asynchronous Read/Match operation with Synchronous Write and Reset operation
- Separate  $\overline{WE}$  for the TAG bits and the Status bits
- Separate  $\overline{OE}$  for the TAG bits, the Status bits, and  $\overline{TA}$
- Synchronous  $\overline{RESET}$  pin for invalidation of all Tag entries
- Dual Chip selects for easy depth expansion with no performance degradation
- I/O pins both 5V TTL and 3.3V LVTTTL compatible with VCCQ pins
- $\overline{PWRDN}$  pin to place device in low-power mode
- Packaged in a 80-pin Thin Plastic Quad Flat Pack (TQFP)

## DESCRIPTION:

The IDT71216 is a 245,760-bit Cache Tag StaticRAM, organized 16K x 15 and designed to support PowerPC and other RISC processors at bus speeds up to 66MHz. There are twelve common I/O TAG bits, with the remaining three bits used as status bits. A 12-bit comparator is on-chip to allow fast comparison of the twelve stored TAG bits and the current Tag input data. An active HIGH MATCH output is generated when these two groups of data are the same for a given address.

This high-speed MATCH signal, with  $t_{ADM}$  as fast as 8ns, provides the fastest possible enabling of secondary cache accesses.

The three separate I/O status bits (VLD, DTY, and WT) can be configured for either dedicated or generic functionality, depending on the SFUNC input pin. With SFUNC LOW, the status bits are defined and used internally by the device, allowing easier determination of the validity and use of the given Tag data. SFUNC HIGH releases the defined internal status bit usage and control, allowing the user to configure the status bit information to fit his system needs. A synchronous  $\overline{RESET}$  pin, when held LOW at a rising clock edge, will reset all status bits in the array for easy invalidation of all Tag addresses.

The IDT71216 also provides the option for Transfer Acknowledge ( $\overline{TA}$ ) generation within the cache tag itself, based upon MATCH, VLD bit, WT bit, and external inputs provided by the user. This can significantly simplify cache controller logic and minimize cache decision time. Match and Read operations are both asynchronous in order to provide the fastest access times possible, while Write operations are synchronous for ease of system timing.

The IDT71216 uses a 5V power supply on Vcc, with separate VCCQ pins provided for the outputs to offer compliance with both 5.0V TTL and 3.3V LVTTTL Logic levels. The  $\overline{PWRDN}$  pin offers a low-power standby mode to reduce power consumption by 90%, providing significant system power savings.

The IDT71216 is fabricated using IDT's high-performance, high-reliability BiCMOS technology and is offered in a space-saving 80-pin Thin Plastic Quad Flat Pack (TQFP) package.

## PIN DESCRIPTIONS

A0 – A13	Address Inputs	Input
$\overline{CS1}$ , CS2	Chip Selects	Input
$\overline{WET}$	Write Enable - Tag Bits	Input
$\overline{WES}$	Write Enable - Status Bits	Input
$\overline{OET}$	Output Enable - Tag Bits	Input
$\overline{OES}$	Output Enable - Status Bits	Input
$\overline{RESET}$	Status Bit Reset	Input
$\overline{PWRDN}$	Powerdown Mode Control Pin	Input
SFUNC	Status Bit Function Control Pin	Input
TT1	Read/Write Input from Processor	Input
VLD <sub>IN</sub> / S1 <sub>IN</sub>	Valid Bit / S1 Bit Input	Input
DTY <sub>IN</sub> / S2 <sub>IN</sub>	Dirty Bit / S2 Bit Input	Input
WT <sub>IN</sub> / S3 <sub>IN</sub>	Write Through Bit / S3 Bit Input	Input

CLK	System Clock	Input
TAH	$\overline{TA}$ Force High	Input
$\overline{TAOE}$	$\overline{TA}$ Output Enable	Input
$\overline{TAIN}$	Additional $\overline{TA}$ Input	Input
$\overline{TA}$	Transfer Acknowledge	Output
TAG0 – TAG11	Tag Data Input/Outputs	I/O
VLD <sub>OUT</sub> / S1 <sub>OUT</sub>	Valid Bit / S1 Bit Output	Output
DTY <sub>OUT</sub> / S2 <sub>OUT</sub>	Dirty Bit / S2 Bit Output	Output
WT <sub>OUT</sub> / S3 <sub>OUT</sub>	Write Through Bit / S3 Bit Output	Output
MATCH	Match	Output
VCC	+5V Power	Pwr
VCCQ	Output Buffer Power	QPwr
VSS	Ground	Gnd

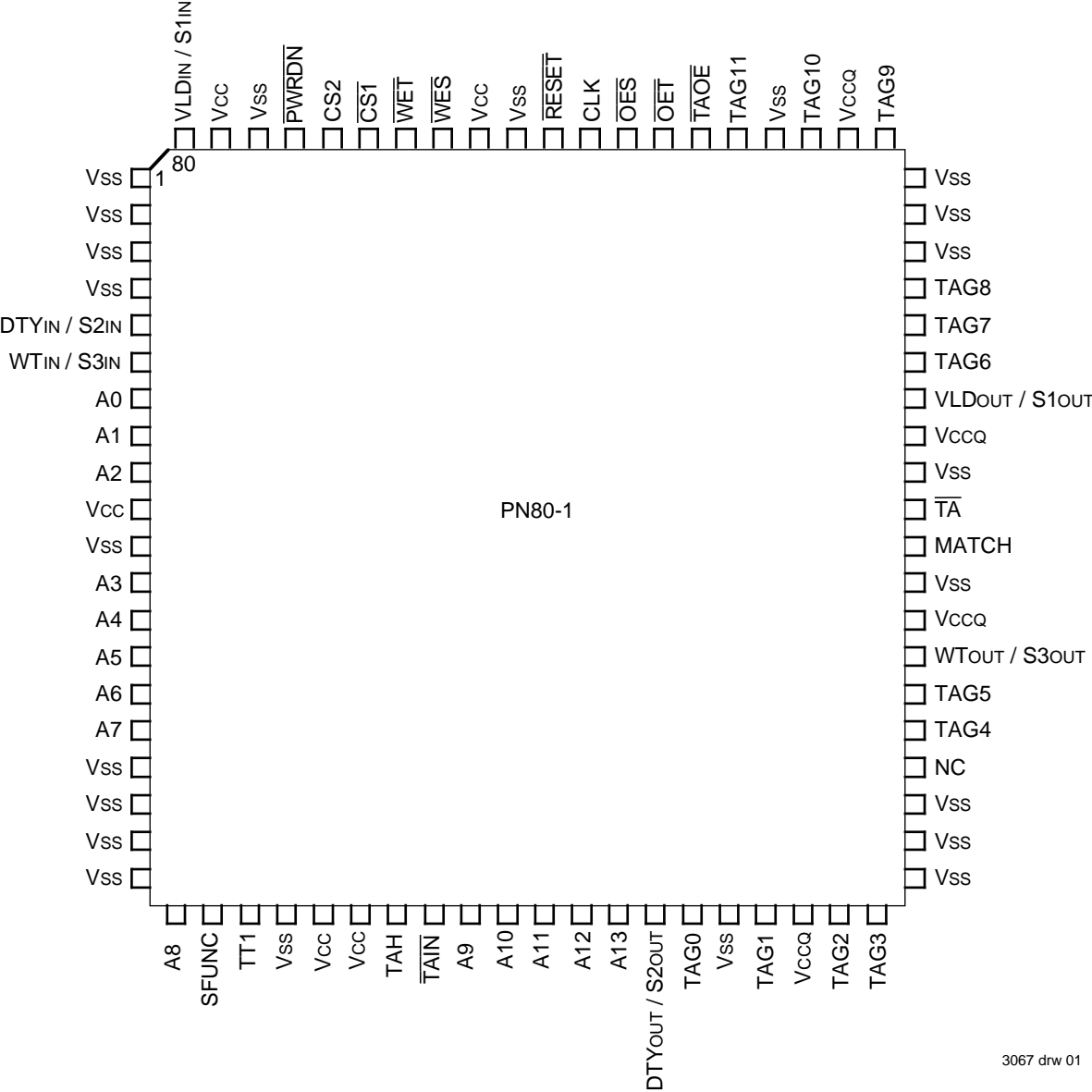
The IDT logo is a registered trademark and CacheRAM is a trademark of Integrated Device Technology, Inc.  
PowerPC is a trademark of International Business Machines, Inc.

3067 tbl 01

## COMMERCIAL TEMPERATURE RANGE

AUGUST 1996

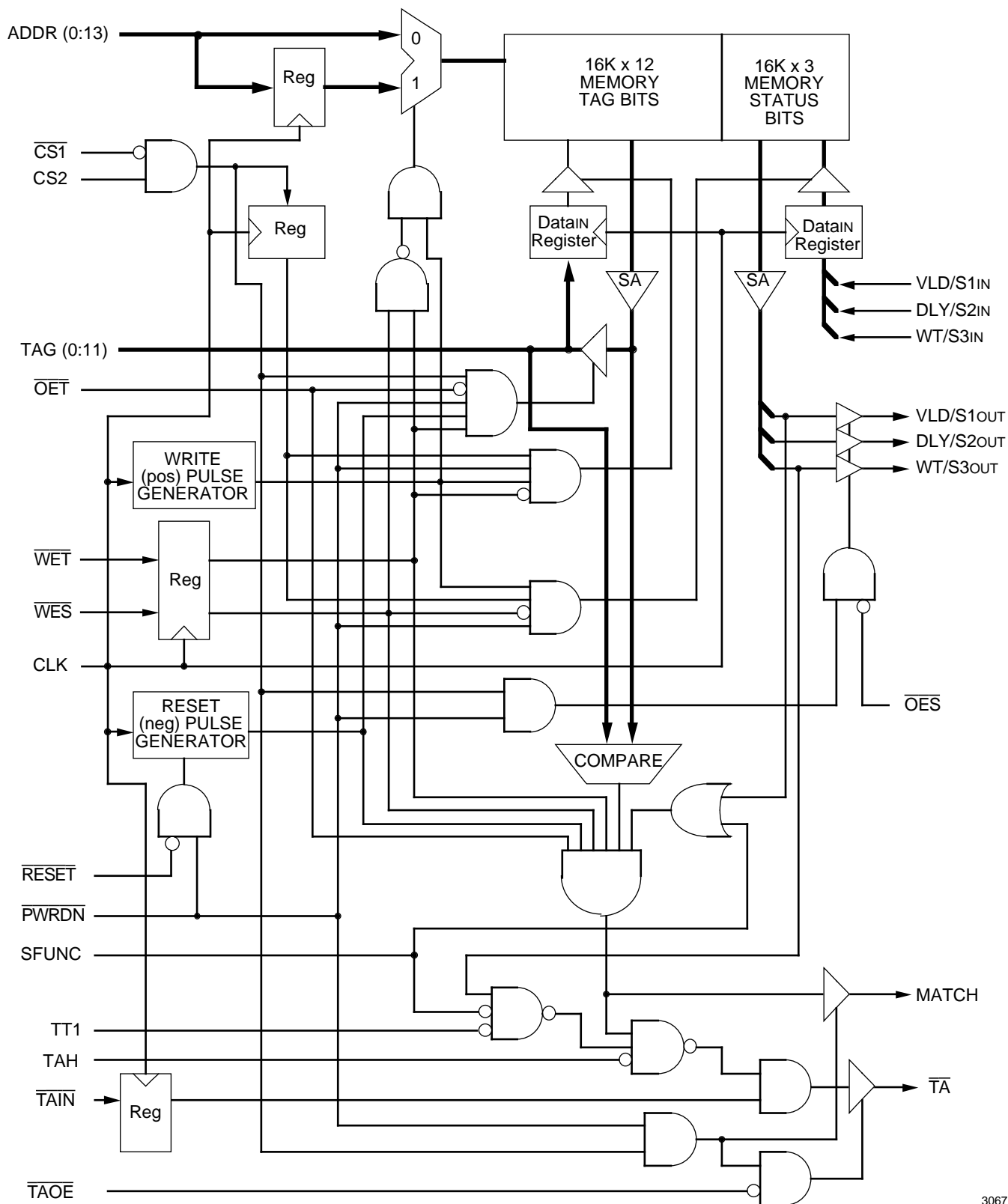
PIN CONFIGURATION



TQFP  
TOP VIEW

3067 drw 01

## FUNCTIONAL BLOCK DIAGRAM



3067 drw 02

## TRUTH TABLES

### CHIP SELECT, RESET, AND POWER-DOWN FUNCTIONS<sup>(1, 2)</sup>

CS1	CS2	RESET	PWRDN	CLK	WET	WES	TAOE	TAG	VLDout	DTYout	WTout	MATCH	TA	OPERATION	POWER
-----	-----	-------	-------	-----	-----	-----	------	-----	--------	--------	-------	-------	----	-----------	-------

#### CHIP SELECT FUNCTION

H	X	X	H	X	X	X	X	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Deselected	Active
X	L	X	H	X	X	X	X	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Deselected	Active
L	H	X	H	X	X	X	X	–	–	–	–	–	–	Selected	Active

#### RESET FUNCTION

L	H	L	H	↑	H	H	L	Hi-Z	L <sup>(3)</sup>	L <sup>(3)</sup>	L <sup>(3)</sup>	L <sup>(3)</sup>	H	Reset Status	Active
L	H	L	H	↑	H	H	H	Hi-Z	L <sup>(3)</sup>	L <sup>(3)</sup>	L <sup>(3)</sup>	L <sup>(3)</sup>	Hi-Z	Reset Status	Active
H	X	L	H	↑	H	H	X	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Reset Status	Active
X	L	L	H	↑	H	H	X	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Reset Status	Active
X	X	L	H	↑	L	X	X	–	–	–	–	–	–	Not Allowed	–
X	X	L	H	↑	X	L	X	–	–	–	–	–	–	Not Allowed	–

#### POWER-DOWN FUNCTION

X	X	X	L	X	H	H	X	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Power-down	Standby
---	---	---	---	---	---	---	---	------	------	------	------	------	------	------------	---------

#### NOTES:

- "H" = V<sub>IH</sub>, "L" = V<sub>IL</sub>, "X" = don't care, "–" = unrelated.
- $\overline{OET}$ ,  $\overline{OES}$ , TT1, TAH,  $\overline{TAIN}$  and SFUNC are "X" for this table.
- OES is LOW.

3067 tbl 02

### READ AND WRITE FUNCTIONS<sup>(1, 2)</sup>

$\overline{OET}$	$\overline{OES}$	WET	WES	CLK	TT1	TAG	VLDin	DTYin	WTin	VLDout	DTYout	WTout	MATCH	OPERATION
------------------	------------------	-----	-----	-----	-----	-----	-------	-------	------	--------	--------	-------	-------	-----------

#### READ FUNCTION

L	X	H	X	X	X	DOUT	–	–	–	–	–	–	DOUT	Read TAG I/O
X	L	X	X	X	X	–	–	–	–	DOUT	DOUT	DOUT	DOUT	Read Status Bits
H	X	X	X	X	X	Hi-Z	–	–	–	–	–	–	DOUT	TAG I/O Disable
X	H	X	X	X	X	–	–	–	–	Hi-Z	Hi-Z	Hi-Z	DOUT	Status Disabled

#### WRITE FUNCTION

H	X	L	X	↑	X	DIN	–	–	–	DOUT	DOUT	DOUT	L	Write TAG I/O
L	X	L	X	↑	X	–	–	–	–	–	–	–	–	Not Allowed
X	L	X	L	↑	X	–	DIN	DIN	DIN	DOUT <sup>(3)</sup>	DOUT <sup>(3)</sup>	DOUT <sup>(3)</sup>	L	Write Status Bits
X	H	X	L	↑	X	–	DIN	DIN	DIN	Hi-Z	Hi-Z	Hi-Z	L	Write Status Bits

#### NOTES:

- "H" = V<sub>IH</sub>, "L" = V<sub>IL</sub>, "X" = don't care, "–" = unrelated.
- This table applies when CS1 is LOW and CS2, RESET, and PWRDN are HIGH.  $\overline{TAOE}$ , TAH,  $\overline{TAIN}$  and SFUNC are "X" for this table.
- DOUT in this case is the same as DIN; that is, the input data is written through to the outputs during the write operation.

3067 tbl 03



## TRUTH TABLES (CONT.)

### MATCH FUNCTION<sup>(1, 2, 3)</sup>

$\overline{CS1}$	CS2	SFUNC	$\overline{OET}$	$\overline{WET}$	$\overline{WES}$	TAG	VLD <sup>(4)</sup>	DTY <sup>(4)</sup>	WT <sup>(4)</sup>	MATCH	OPERATION
H	X	X	X	X	X	Hi-Z	–	–	–	Hi-Z	Deselected
X	L	X	X	X	X	Hi-Z	–	–	–	Hi-Z	Deselected
L	H	X	X	X	X	–	–	–	–	DOUT	Selected
L	H	X	L	H	X	DOUT	–	–	–	L	Read Tag I/O
L	H	X	H	L	X	DIN	–	–	–	L	Write Tag I/O
L	H	X	X	X	L	–	DIN	DIN	DIN	L	Write Status Bits
L	H	L	H	H	H	TAGIN	L	–	–	L	Invalid Data - Dedicated Status Bits
L	H	L	H	H	H	TAGIN	H	–	–	M	Match - Dedicated Status Bits
L	H	H	H	H	H	TAGIN	X	–	–	M	Match - Generic Status Bits

#### NOTES:

3067 tbl 04

- "H" =  $V_{IH}$ , "L" =  $V_{IL}$ , "X" = don't care, "–" = unrelated.
- M = HIGH if TAGIN equals the memory contents at that address; M = LOW if TAGIN does not equal the memory contents at that address.
- PWRDN and RESET are HIGH for this table. TT1, TAH, TAOE, TAIN, OES, and CLK are "X".
- This column represents the stored memory cell data for the given Status bit at the selected address.

### $\overline{TA}$ FUNCTION<sup>(1, 2, 3, 5)</sup>

TAOE	$\overline{TAIN}^{(6)}$	$\overline{OET}$	$\overline{WET}$	$\overline{WES}$	TAH	TT1	SFUNC	VLD <sup>(4)</sup>	DTY <sup>(4)</sup>	WT <sup>(4)</sup>	TAG	MATCH	$\overline{TA}$	OPERATION
H	X	X	X	X	X	X	X	X	–	X	–	–	Hi-Z	$\overline{TA}$ Disabled
L	L	X	X	X	X	X	X	X	–	X	–	X	L	External $\overline{TA}$ Input <sup>(7)</sup>
L	H	L	X	X	X	X	X	X	–	X	DOUT	L	H	Read TAG
L	H	X	L	X	X	X	X	X	–	X	DIN	L	H	Write TAG
L	H	X	X	L	X	X	X	DIN	DIN	DIN	–	L	H	Write Status
L	H	X	X	X	H	X	X	X	–	X	–	X	H	Force $\overline{TA}$ HIGH
L	H	X	X	X	X	X	L	L	–	X	–	L	H	Invalid TAG
L	H	X	X	X	X	L	L	X	–	H	–	X	H	Write Through
L	H	H	H	H	L	X	L	H	–	L	TAGIN	M	$\overline{M}$	Compare
L	H	H	H	H	L	H	L	H	–	X	TAGIN	M	$\overline{M}$	Compare
L	H	H	H	H	L	X	L	H	–	X	TAGIN	M	$\overline{M}$	Compare
L	H	H	H	H	L	X	H	X	–	X	TAGIN	M	$\overline{M}$	Compare

#### NOTES:

3067 tbl 05

- "H" =  $V_{IH}$ , "L" =  $V_{IL}$ , "X" = don't care, "–" = unrelated.
- M = HIGH if TAGIN equals the memory contents at that address; M = LOW if TAGIN does not equal the memory contents at that address.
- PWRDN and RESET are HIGH for this table. CLK and OES are "X".
- This column represents the stored memory cell data for the given Status bit at the selected address.
- $\overline{CS1}$  is LOW, CS2 is HIGH for this table.
- TAIN is a synchronous input; thus the inputs noted in the table must be applied during a rising CLK edge.
- TAIN will be a factor in determining the  $\overline{TA}$  output in all cases except when TAH is HIGH and there is a valid MATCH. In that case,  $\overline{TA}$  will be LOW(Valid).

## RECOMMENDED DC OPERATING CONDITIONS

Symbol	Parameter	Min.	Typ.	Max.	Unit
V <sub>CC</sub>	Supply Voltage	4.75	5.0	5.25	V
V <sub>CCQ</sub>	5V Output Buffers	4.75	5.0	5.25	V
V <sub>CCQ</sub>	3.3V Output Buffers	3.0	3.3	3.6	V
V <sub>SS</sub>	Supply Ground	0	0	0	V
V <sub>IH</sub>	Input High Voltage	2.2	3.0	V <sub>CC</sub> +0.3	V
V <sub>IHQ</sub>	I/O High Voltage	2.2	3.0	V <sub>CCQ</sub> +0.3	V
V <sub>IL</sub>	Input Low Voltage	-0.5 <sup>(1)</sup>	—	0.8	V

**NOTE:** 3067 tbl 06  
1. V<sub>IL</sub> (min.) = -1.5V for pulse width of less than 10ns, once per cycle.

## CAPACITANCE

(T<sub>A</sub> = +25°C, f = 1.0 MHz)

Symbol	Parameter <sup>(1)</sup>	Condition	Max.	Unit
C <sub>IN</sub>	Input Capacitance	V <sub>IN</sub> = 0V	5	pF
C <sub>TAG</sub>	TAG Input/Output Capacitance	V <sub>I/O</sub> = 0V	7	pF
C <sub>OUT</sub>	Output Capacitance	V <sub>OUT</sub> = 0V	7	pF

**NOTE:** 3067 tbl 07  
1. This parameter is determined by device characterization but is not production tested.

## ABSOLUTE MAXIMUM RATINGS<sup>(1)</sup>

Symbol	Rating	Value	Unit
V <sub>TERM</sub>	Terminal Voltage with Respect to GND	-0.5 to +7.0 <sup>(2)</sup>	V
T <sub>A</sub>	Operating Temperature	-0 to +70	°C
T <sub>BIAS</sub>	Temperature Under Bias	-65 to +135	°C
T <sub>STG</sub>	Storage Temperature	-65 to +150	°C
P <sub>T</sub>	Power Dissipation	1.7	W
I <sub>OUT</sub>	DC Output Current	20	mA

**NOTES:** 3067 tbl 08  
1. Stresses greater than those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect reliability.  
2. V<sub>IN</sub> should not exceed V<sub>CC</sub>+0.5V. All pins should not exceed 7.0V. V<sub>CCQ</sub> should never exceed V<sub>CC</sub>, and V<sub>CC</sub> should never exceed V<sub>CCQ</sub> + 4.0V.

## DC ELECTRICAL CHARACTERISTICS OVER THE OPERATING TEMPERATURE AND SUPPLY VOLTAGE RANGE

(V<sub>CC</sub> = 5.0V ± 5%, V<sub>CCQ</sub> = 5.0V ± 5% OR 3.3V ± 0.3V)

Symbol	Parameter	Test Condition	Min.	Max.	Unit
I <sub>LI</sub>	Input Leakage Current	V <sub>CC</sub> = Max., V <sub>IN</sub> = 0V to V <sub>CC</sub>	—	5	μA
I <sub>LO</sub>	Output Leakage Current	$\overline{CS1} \geq V_{IH}$ , CS2 ≤ V <sub>IL</sub> , $\overline{OE} \geq V_{IH}$ , V <sub>CC</sub> = Max. V <sub>OUT</sub> = 0V to V <sub>CCQ</sub> , V <sub>CCQ</sub> = Max.	—	5	μA
V <sub>OL</sub>	Output Low Voltage	I <sub>OL</sub> = 4mA, V <sub>CC</sub> = Min.	—	0.4	V
V <sub>OH</sub>	Output High Voltage	I <sub>OH</sub> = -4mA, V <sub>CC</sub> = Min.	2.4	—	V

3067 tbl 09

## DC ELECTRICAL CHARACTERISTICS OVER THE OPERATING TEMPERATURE AND SUPPLY VOLTAGE RANGE<sup>(1, 2)</sup> (V<sub>CC</sub> = 5.0V ± 5%)

Symbol	Parameter	Test Condition	71216S8		71216S9		71216S10		71216S12		Unit
			Com'l.	Mil.	Com'l.	Mil.	Com'l.	Mil.	Com'l.	Mil.	
I <sub>CC</sub>	Operating Power Supply Current	$\overline{PWRDN} \geq V_{IH}$ Outputs Open, V <sub>CC</sub> = Max., f = f <sub>MAX</sub> <sup>(3)</sup>	330	—	300	—	290	—	280	—	mA
I <sub>SB</sub>	Standby Power Supply Current	$\overline{PWRDN} \leq V_{IL}$ , V <sub>IN</sub> ≥ V <sub>IH</sub> or ≤ V <sub>IL</sub> V <sub>CC</sub> = Max., f = f <sub>MAX</sub> <sup>(3)</sup>	30	—	30	—	30	—	30	—	mA
I <sub>SB1</sub>	Full Standby Power Supply Current	$\overline{PWRDN} \leq V_{IL}$ , V <sub>IN</sub> ≥ V <sub>HC</sub> or ≤ V <sub>LC</sub> <sup>(4)</sup> V <sub>CC</sub> = Max., f = 0 <sup>(3)</sup>	25	—	25	—	25	—	25	—	mA

**NOTES:** 3067 tbl 10  
1. All values are maximum guaranteed values.  
2.  $\overline{CS1} \leq V_{IL}$ , CS2 ≥ V<sub>IH</sub>.  
3. f<sub>MAX</sub> = 1/t<sub>CYC</sub> (all address inputs are cycling at f<sub>MAX</sub>). f = 0 means no address input lines are changing.  
4. V<sub>HC</sub> = V<sub>CC</sub> - 0.2V, V<sub>LC</sub> = 0.2V

## AC ELECTRICAL CHARACTERISTICS

(VCC = 5.0V ± 5%, VCCQ = 5.0V ± 5% OR 3.3V ± 0.3V, TA = 0 to 70°C)

Symbol	Parameter	IDT71216S8		IDT71216S9		IDT71216S10		IDT71216S12		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Read Cycle										
tAAT	Address Access Time Tag Bits	—	10	—	11	—	12	—	14	ns
tACST	Chip Select Access Time Tag Bits	—	8	—	9	—	10	—	12	ns
tCLZ <sup>(1)</sup>	Chip Select to Tag and Status Bits in Low-Z	1	—	1	—	1	—	1	—	ns
tCHZ <sup>(1)</sup>	Chip Select to Tag and Status Bits in High-Z	1	5	1	6	1	6	1	7	ns
tOET	Output Enable to Tag Bits Valid	—	5	—	6	—	6	—	7	ns
tOTLZ <sup>(1)</sup>	Output Enable to Tag Bits in Low-Z	0	—	0	—	0	—	0	—	ns
tOTHZ <sup>(1)</sup>	Output Enable to Tag Bits in High-Z	1	5	1	6	1	6	1	7	ns
tTOH	Tag Bit Hold from Address Change	2	—	2	—	2	—	2	—	ns
tOES	Output Enable to Status Bits Valid	—	5	—	6	—	6	—	7	ns
tOSLZ <sup>(1)</sup>	Output Enable to Status Bits in Low-Z	0	—	0	—	0	—	0	—	ns
tOSHZ <sup>(1)</sup>	Output Enable to Status Bits in High-Z	1	5	1	6	1	6	1	7	ns
tAAS	Address Access Time Status Bits	—	8	—	9	—	10	—	12	ns
tACSS	Chip Select Access Time Status Bits	—	6	—	7	—	8	—	10	ns
tSOH	Status Bit Hold from Address Change	2	—	2	—	2	—	2	—	ns

### NOTE:

1. This parameter is guaranteed with the AC Load (Figure 3) by device characterization, but is not production tested.

3067 tbl 11

## AC ELECTRICAL CHARACTERISTICS <sup>(1)</sup>

(VCC = 5.0V ± 5%, VCCQ = 5.0V ± 5% OR 3.3V ± 0.3V, TA = 0 to 70°C)

Symbol	Parameter	IDT71216S8		IDT71216S9		IDT71216S10		IDT71216S12		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Reset and Power Down Cycles										
tSR	$\overline{\text{RESET}}$ Set-up Time	4	—	4	—	4	—	4	—	ns
tHR	$\overline{\text{RESET}}$ Hold Time	1	—	1	—	1	—	1	—	ns
tSRST	Status Bit Reset Time	—	50	—	60	—	60	—	70	ns
tSHRS	Status Bit Hold from $\overline{\text{RESET}}$ LOW	2	—	2	—	2	—	2	—	ns
tRSMI	$\overline{\text{RESET}}$ LOW to MATCH and $\overline{\text{TA}}$ Invalid	—	9	—	10	—	10	—	12	ns
tRSMV	$\overline{\text{RESET}}$ HIGH to MATCH and $\overline{\text{TA}}$ Valid	—	110	—	120	—	120	—	130	ns
tRSHZ <sup>(2)</sup>	$\overline{\text{RESET}}$ LOW to TAG High-Z	—	9	—	10	—	10	—	12	ns
tRSLZ <sup>(2)</sup>	$\overline{\text{RESET}}$ HIGH to TAG Low-Z	—	90	—	100	—	100	—	110	ns
tPDSR	PWRDN Set-up to $\overline{\text{RESET}}$ LOW	30	—	30	—	30	—	30	—	ns
tRHPL	$\overline{\text{RESET}}$ HIGH to $\overline{\text{PWDRN}}$ LOW	1	—	1	—	1	—	1	—	CLK
tRHWL	$\overline{\text{RESET}}$ HIGH to $\overline{\text{WET}}$ and $\overline{\text{WES}}$ LOW	90	—	95	—	95	—	105	—	ns
tPD <sup>(2)</sup>	$\overline{\text{PWDRN}}$ LOW to Low Power Mode	—	50	—	50	—	50	—	50	ns
tPU <sup>(2)</sup>	$\overline{\text{PWDRN}}$ HIGH to Active Power Mode	0	—	0	—	0	—	0	—	ns
tPDHZ <sup>(2)</sup>	$\overline{\text{PWDRN}}$ LOW to Outputs in High-Z	—	9	—	10	—	10	—	12	ns
tPDLZ <sup>(2)</sup>	$\overline{\text{PWDRN}}$ HIGH to Outputs in Low-Z	0	—	0	—	0	—	0	—	ns
tPUV	$\overline{\text{PWDRN}}$ HIGH to Outputs Valid	—	50	—	50	—	50	—	50	ns
tWHPL <sup>(2)</sup>	$\overline{\text{WET}}$ and $\overline{\text{WES}}$ HIGH to $\overline{\text{PWDRN}}$ LOW	5	—	5	—	5	—	5	—	ns
tPUWL	$\overline{\text{PWDRN}}$ HIGH to $\overline{\text{WET}}$ and $\overline{\text{WES}}$ Active	50	—	50	—	50	—	50	—	ns

### NOTES:

- Power-down mode is intended to be used during extended time periods of device inactivity.
- This parameter is guaranteed with the AC Load (Figure 3) by device characterization, but is not production tested.

3067 tbl 12

## AC ELECTRICAL CHARACTERISTICS <sup>(1)</sup>

(V<sub>CC</sub> = 5.0V ± 5%, V<sub>CCQ</sub> = 5.0V ± 5% OR 3.3V ± 0.3V, T<sub>A</sub> = 0 to 70°C)

Symbol	Parameter	IDT71216S8		IDT71216S9		IDT71216S10		IDT71216S12		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
Write Cycle and Clock Parameters										
tCYC	Clock Cycle Time	15	—	15	—	15	—	16.6	—	ns
tCH <sup>(2, 3)</sup>	Clock Pulse HIGH	4.5	—	4.5	—	4.5	—	5	—	ns
tCL <sup>(2, 3)</sup>	Clock Pulse LOW	4.5	—	4.5	—	4.5	—	5	—	ns
ts	$\overline{\text{WET}}$ , $\overline{\text{WES}}$ , Chip Select, and Input Data Set-up Time	3	—	3	—	3	—	3	—	ns
tH	$\overline{\text{WET}}$ , $\overline{\text{WES}}$ , Chip Select, and Input Data Hold Time	1	—	1	—	1	—	1	—	ns
tSA	Address Set-up Time	3	—	3	—	3	—	3	—	ns
tHA	Address Hold Time	1	—	1	—	1	—	1	—	ns
tWMI	CLK HIGH Write to MATCH and $\overline{\text{TA}}$ Invalid	—	6	—	7	—	7	—	8	ns
tCKLZ <sup>(3)</sup>	CLK HIGH Read to Outputs in Low-Z	1.5	—	1.5	—	1.5	—	1.5	—	ns
tCTV <sup>(4)</sup>	CLK HIGH Read to Tag Bits Valid	—	9	—	10	—	10	—	12	ns
tCSV <sup>(4)</sup>	CLK HIGH Write to Status Outputs Valid	—	8	—	9	—	9	—	10	ns
tCSH <sup>(3)</sup>	Status Output Hold from CLK HIGH Write	0	—	0	—	0	—	0	—	ns
tWHPL	$\overline{\text{WET}}$ and $\overline{\text{WES}}$ HIGH to $\overline{\text{PWRDN}}$ LOW	5	—	5	—	5	—	5	—	ns
tPUWL	$\overline{\text{PWRDN}}$ HIGH to $\overline{\text{WET}}$ and $\overline{\text{WES}}$ Active	50	—	50	—	50	—	50	—	ns

### NOTES:

1. All Write cycles are synchronous and referenced from rising CLK.
2. This parameter is measured as a HIGH time above 2.0V and a LOW time below 0.8V.
3. This parameter is guaranteed with the AC Load (Figure 3) by device characterization, but is not production tested.
4. Addresses are stable prior to CLK transition HIGH.

3067 tbl 14

## AC ELECTRICAL CHARACTERISTICS

(V<sub>CC</sub> = 5.0V ± 5%, V<sub>CCQ</sub> = 5.0V ± 5% OR 3.3V ± 0.3V, T<sub>A</sub> = 0 to 70°C)

Symbol	Parameter	IDT71216S8		IDT71216S9		IDT71216S10		IDT71216S12		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
MATCH and $\overline{\text{TA}}$ Cycles										
tADM	Address to MATCH Valid	—	8	—	9	—	10	—	12	ns
tDAM	Data Input to MATCH Valid	—	8	—	9	—	10	—	12	ns
tCSM	Chip Select to MATCH Valid	—	8	—	9	—	10	—	12	ns
tCMLZ <sup>(1)</sup>	Chip Select to MATCH in Low-Z	1	—	1	—	1	—	1	—	ns
tCMHZ <sup>(1)</sup>	Chip Select to MATCH in High-Z	1	5	1	6	1	6	1	7	ns
tMHA	MATCH Valid Hold from Address	2	—	2	—	2	—	2	—	ns
tMHD	MATCH Valid Hold from Data	2	—	2	—	2	—	2	—	ns
tBHA	$\overline{\text{TA}}$ Valid Hold from Address	2	—	2	—	2	—	2	—	ns
tBHD	$\overline{\text{TA}}$ Valid Hold from Data	2	—	2	—	2	—	2	—	ns
tADB	Address to $\overline{\text{TA}}$ Valid	—	9	—	10	—	11	—	13	ns
tDAB	Data Input to $\overline{\text{TA}}$ Valid	—	9	—	10	—	11	—	13	ns
tCSB	Chip Select LOW to $\overline{\text{TA}}$ Valid	—	9	—	10	—	11	—	13	ns
toEBV	$\overline{\text{TAOE}}$ LOW to $\overline{\text{TA}}$ Valid	—	6	—	6	—	7	—	8	ns
toBLZ <sup>(1)</sup>	$\overline{\text{TAOE}}$ LOW to $\overline{\text{TA}}$ in Low-Z	0	—	0	—	0	—	0	—	ns
toBHZ <sup>(1)</sup>	$\overline{\text{TAOE}}$ HIGH to $\overline{\text{TA}}$ in High-Z	1	5	1	6	1	6	1	7	ns
tBYFH	TAH HIGH to Force $\overline{\text{TA}}$ HIGH	—	5	—	5	—	5	—	6	ns
tBYHV	TAH LOW to $\overline{\text{TA}}$ Valid	—	5	—	5	—	5	—	6	ns
tSB	$\overline{\text{TA}}$ IN Set-up Time	4	—	4	—	4	—	4	—	ns
tHB	$\overline{\text{TA}}$ IN Hold Time	1.5	—	1.5	—	1.5	—	1.5	—	ns
tBIBL	CLK HIGH $\overline{\text{TA}}$ IN LOW to $\overline{\text{TA}}$ LOW	—	6	—	6	—	7	—	8	ns
tBIBV	CLK HIGH $\overline{\text{TA}}$ IN HIGH to $\overline{\text{TA}}$ Valid	—	6	—	6	—	7	—	8	ns
toEMI	$\overline{\text{OET}}$ LOW to MATCH and $\overline{\text{TA}}$ Invalid	—	6	—	7	—	7	—	8	ns
toEMV	$\overline{\text{OET}}$ HIGH to MATCH and $\overline{\text{TA}}$ Valid	—	7	—	8	—	8	—	10	ns
tWRBH <sup>(2)</sup>	W/ $\overline{\text{R}}$ HIGH to $\overline{\text{TA}}$ HIGH	—	6	—	7	—	7	—	8	ns
tWRBV <sup>(2)</sup>	W/ $\overline{\text{R}}$ LOW to $\overline{\text{TA}}$ Valid	—	6	—	7	—	7	—	8	ns
tWMI	CLK HIGH Write to MATCH and $\overline{\text{TA}}$ Invalid	—	7	—	7	—	7	—	8	ns
tWMV <sup>(3)</sup>	CLK HIGH Read to MATCH and $\overline{\text{TA}}$ Valid	—	8	—	9	—	10	—	12	ns

### NOTES:

1. This parameter is guaranteed with the AC Load (Figure 3) by device characterization, but is not production tested.
2. These parameters only apply when SFUNC is LOW and the internal WT bit is HIGH.
3. t<sub>ADM</sub>, t<sub>DAM</sub>, t<sub>CSM</sub> and t<sub>ADB</sub>, t<sub>DAB</sub>, t<sub>CSB</sub> must also be satisfied.

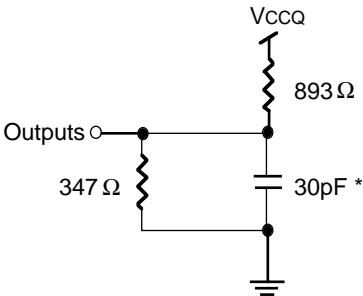
3067 tbl 15

AC TEST CONDITIONS

Input Pulse Levels	GND to 3.0V
Input Rise/Fall Times	3ns
Input Timing Reference Levels	1.5V
Output Timing Reference Levels	1.5V
AC Test Load	See Figs. 1, 2, 3, & 4

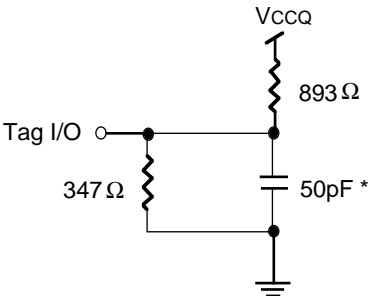
3067 tbl 16

AC TEST LOADS



3067 drw 03

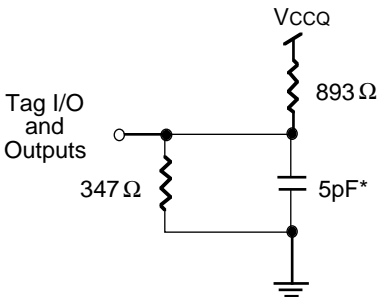
Figure 1. AC Test Load



3067 drw 04

Figure 2. Tag I/O AC Test Load

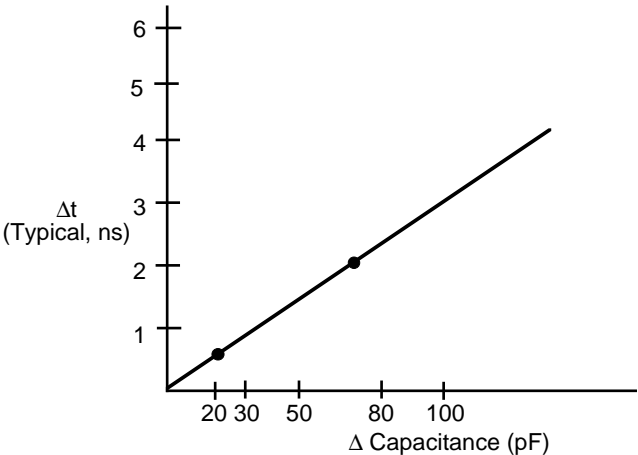
\* Including scope and jig capacitance



3067 drw 05

Figure 3. AC Test Load  
(for tHZ and tLZ parameters )

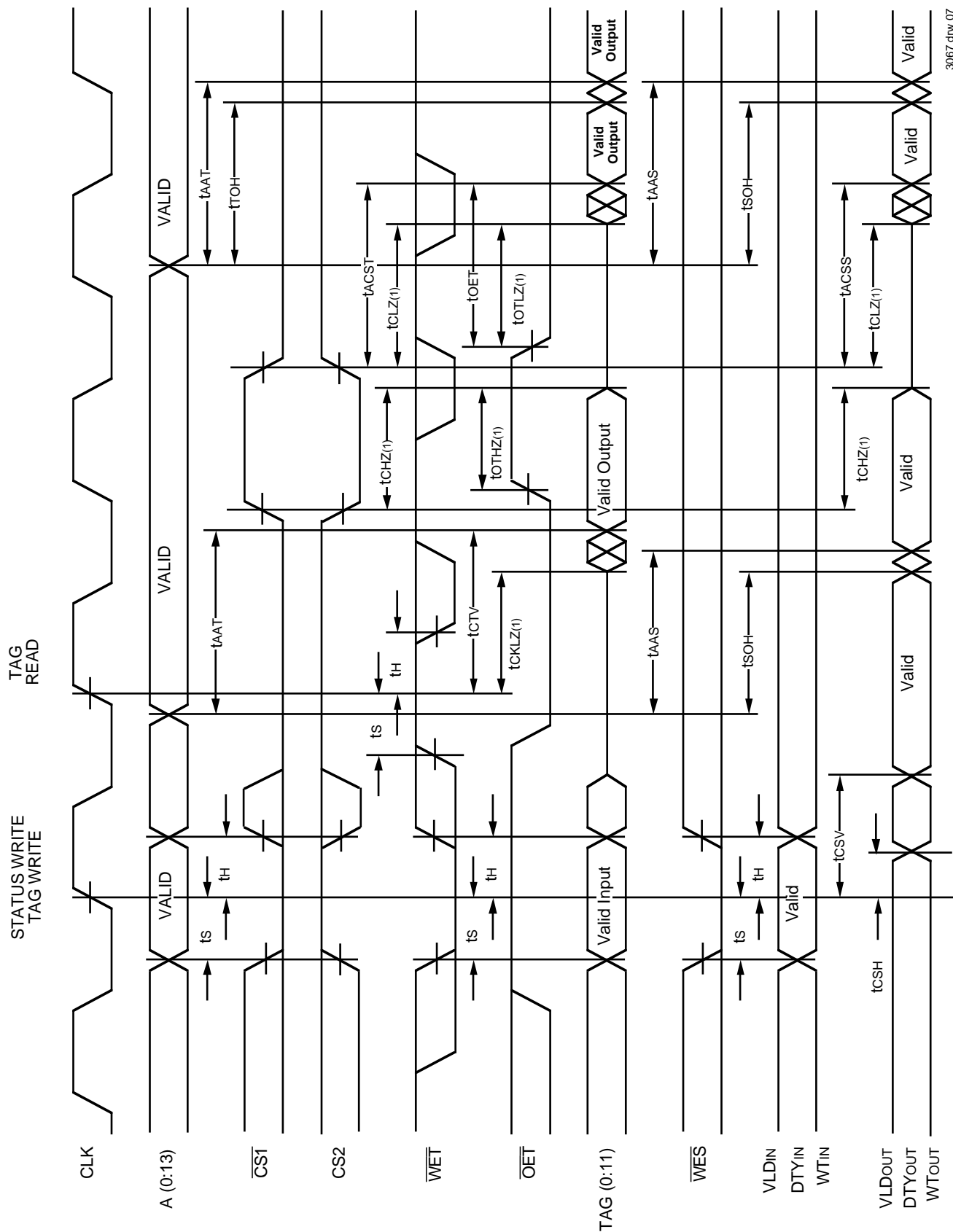
\* Including scope and jig capacitance



3067 drw 06

Figure 4. Lumped Capacitance Load, Typical Derating

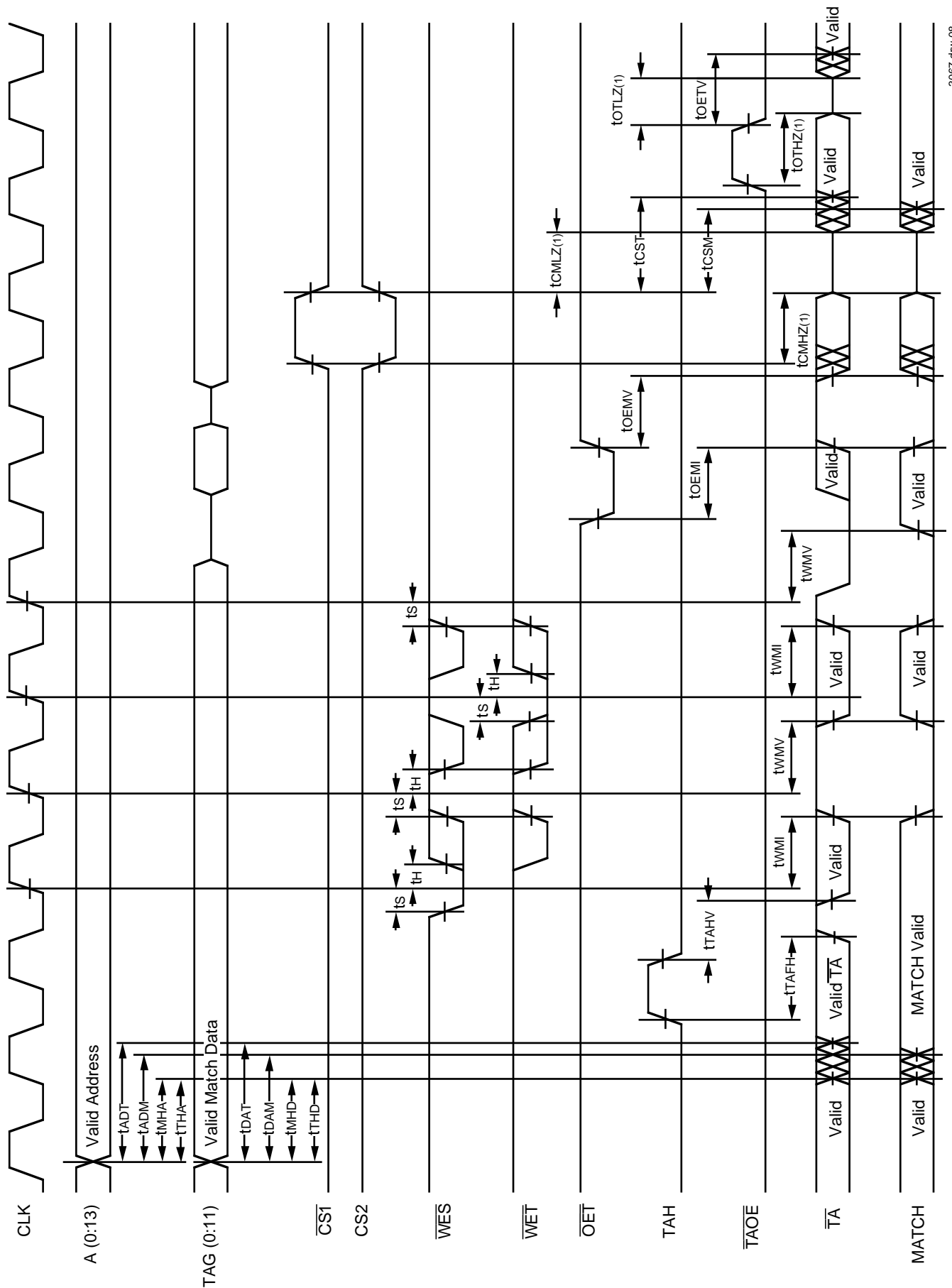
## TIMING WAVEFORMS OF WRITE AND READ CYCLES



**NOTE:**

1. Transition is measured  $\pm 200\text{mV}$  from steady state.

## TIMING WAVEFORMS OF MATCH AND $\overline{\text{TA}}$ FUNCTIONS

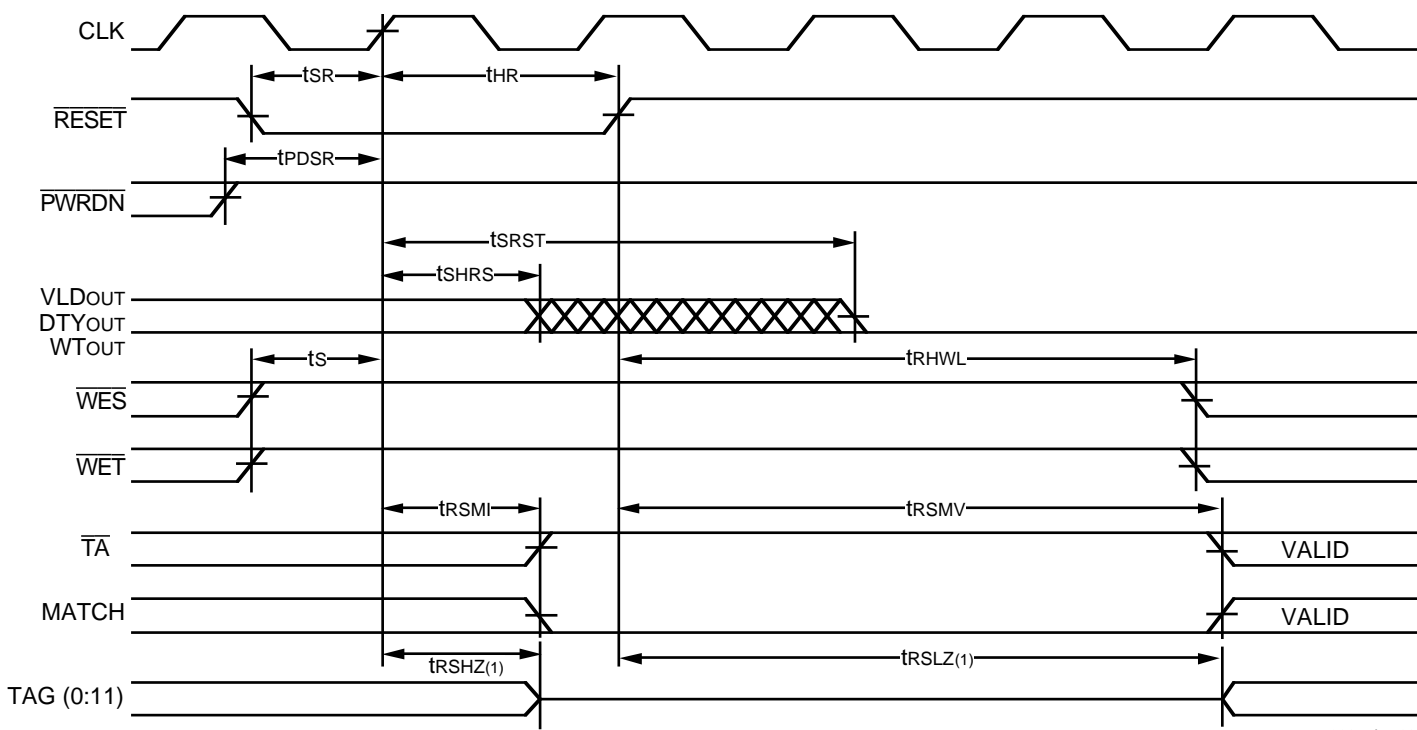


**NOTE:**

1. Transition is measured  $\pm 200\text{mV}$  from steady state.

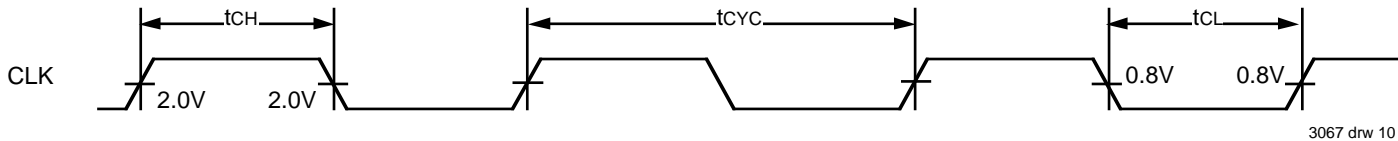


TIMING WAVEFORMS OF RESET FUNCTION



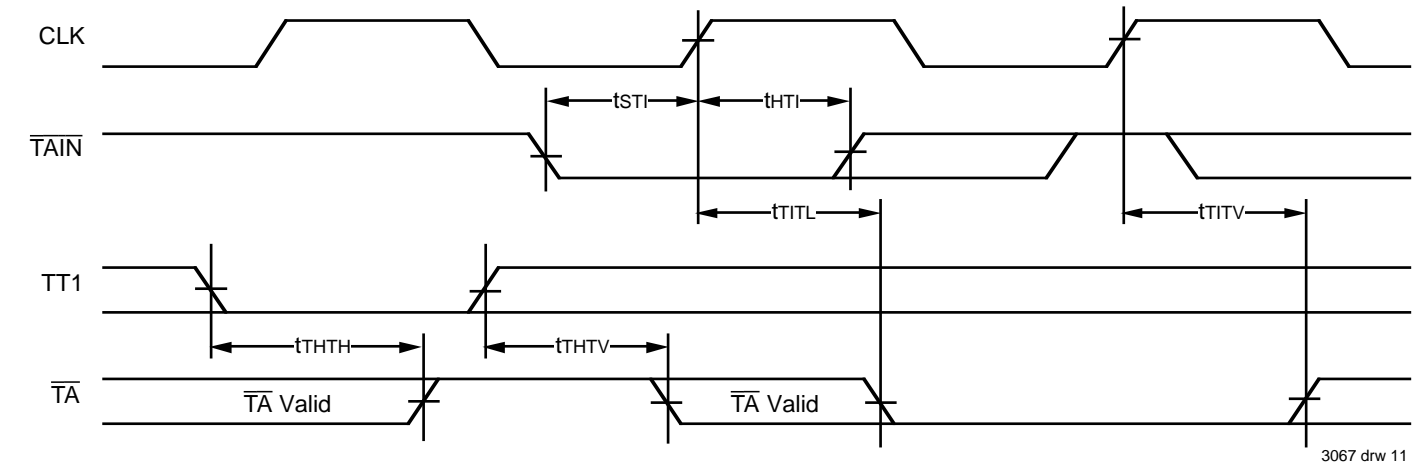
NOTE:  
1. Transition is measured ±200mV from steady state.

CLOCK TIMING WAVEFORM

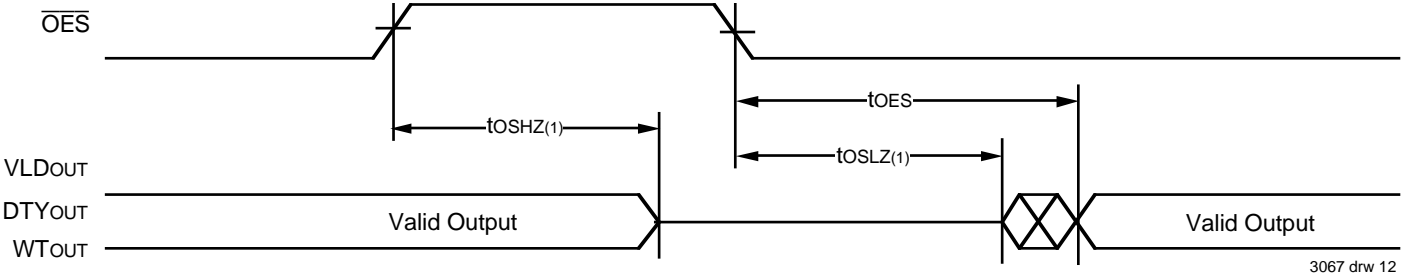


TIMING WAVEFORMS OF TA AND TT1 SIGNAL

Applies when SFUNC is LOW, and the internal WT bit is HIGH

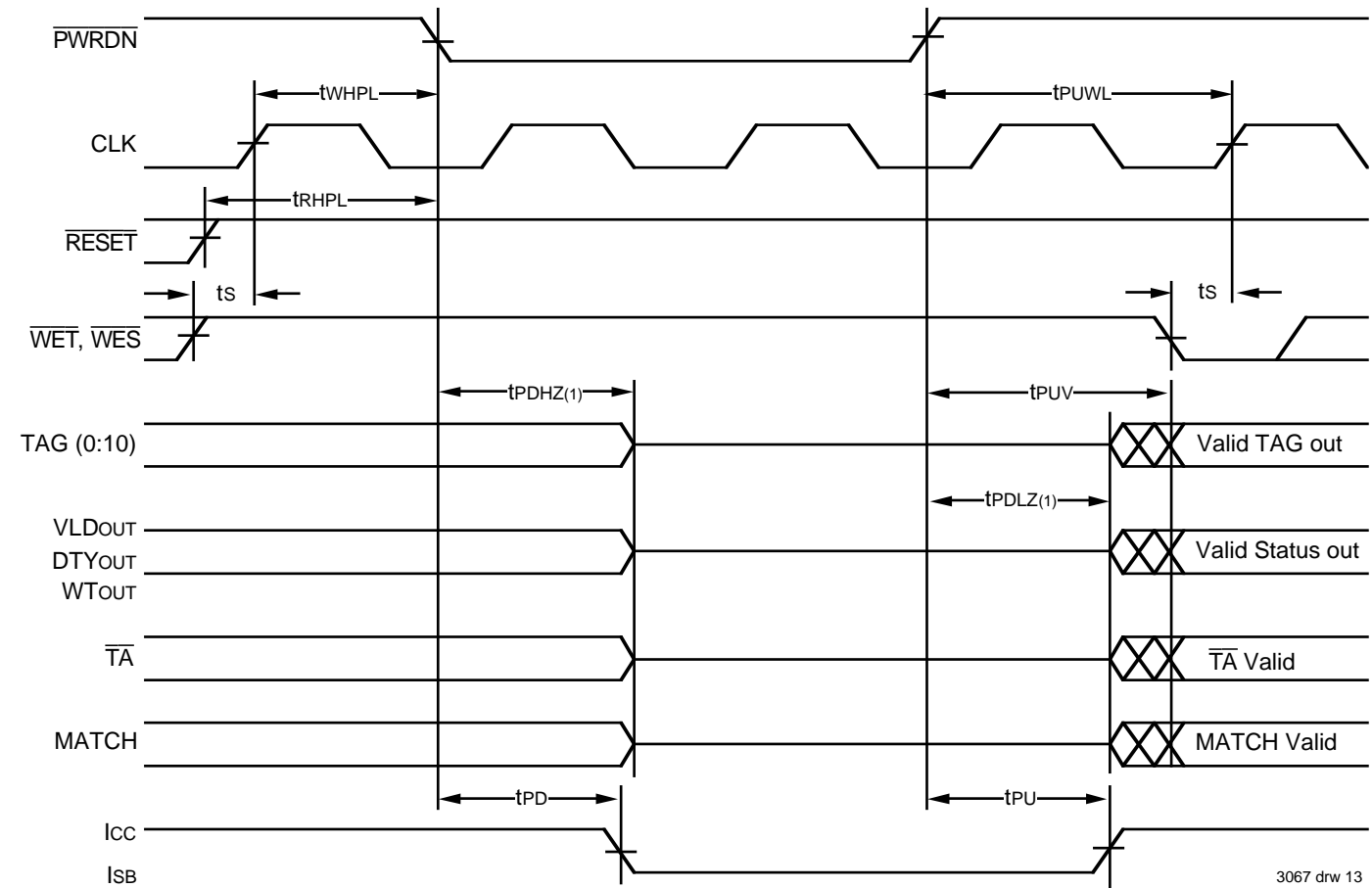


TIMING WAVEFORMS OF  $\overline{\text{OES}}$  FUNCTION



NOTE:  
1. Transition is measured  $\pm 200\text{mV}$  from steady state.

TIMING WAVEFORMS OF POWER DOWN FUNCTION



NOTE:  
1. Transition is measured  $\pm 200\text{mV}$  from steady state.

ORDERING INFORMATION

IDT	71216	S	XX	PF	
Device Type	Power	Speed	Package		
				PF	Plastic Thin Quad Flatpack (PN80-1)
				8	
				9	
				10	
				12	
					Speed in nanoseconds

## Low Voltage PLL Clock Driver

The MPC970 is a 3.3V compatible, PLL based clock driver devices targeted for high performance RISC or CISC processor based systems.

- Fully Integrated PLL
- Output Frequency Up to 250MHz
- Compatible with PowerPC™ and Pentium™ Processors
- Output Frequency Configuration
- On-Board Crystal Oscillator
- 52-Lead TQFP Packaging
- $\pm 50$ ps Typical Cycle-to-Cycle Jitter

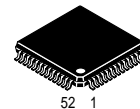
The MPC970 was designed specifically to drive today's PowerPC 601 and Pentium processors while providing the necessary performance to address higher frequency PowerPC 601 as well as PowerPC 603 and PowerPC 604 applications. The 2x\_PCLK output can toggle at up to 250MHz while the remaining outputs can be configured to drive the other system clocks for MPC 601 based systems. As the processor based clock speeds increase the processor bus will likely run at one third or even one fourth the processor clock. The MPC970 supports the necessary waveforms to drive the BCLKEN input signal of the MPC 601 when the processor bus is running at a lower frequency than the processor. The MPC970 uses an advanced PLL design which minimizes the jitter generated on the outputs. The jitter specification is well within the requirements of the Pentium processor and meets the stringent preliminary specifications of the PowerPC 603 and PowerPC 604 processors. The application section of this data sheet deals in more detail with driving PowerPC and Pentium processor based systems.

The external feedback option of the MPC970 provides for a near zero delay between the reference clock input and the outputs of the device. This feature is required in applications where a master clock is being picked up off the backplane and regenerated and distributed on a daughter card. The advanced PLL of the MPC970 eliminates the dead zone of the phase detector and minimizes the jitter of the PLL so that the phase error variation is held to a minimum. This phase error uncertainty makes up a major portion of the part-to-part skew of the device.

For single clock driver applications the MPC970 provides an internal oscillator and internal feedback to simplify board layout and minimize system cost. By using the on-board crystal oscillator the MPC970 acts as both the clock generator and distribution chip. The external component is a relatively inexpensive crystal rather than a more expensive oscillator. Since in single board applications the delay between the input reference and the outputs is inconsequential an internal feedback option is offered. The internal feedback simplifies board design in that the system designer need not worry about noise being coupled into the feedback line due to board parasitics and layout. The internal feedback is a fixed divide by 32 of the VCO. This divide ratio ensures that the input crystals will be  $\leq 20$ MHz, thus keeping the crystal costs down and ensuring availability from multiple vendors.

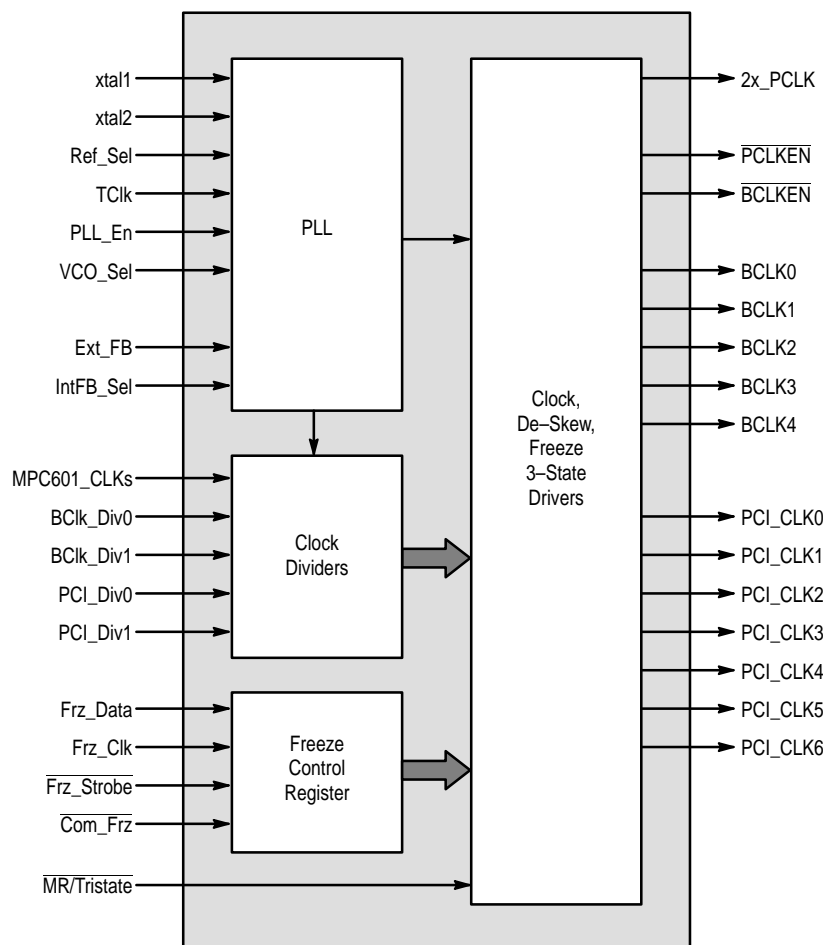
**MPC970**

**LOW VOLTAGE  
PLL CLOCK DRIVER**



52 1  
**FA SUFFIX**  
TQFP PACKAGE  
CASE 848D-03



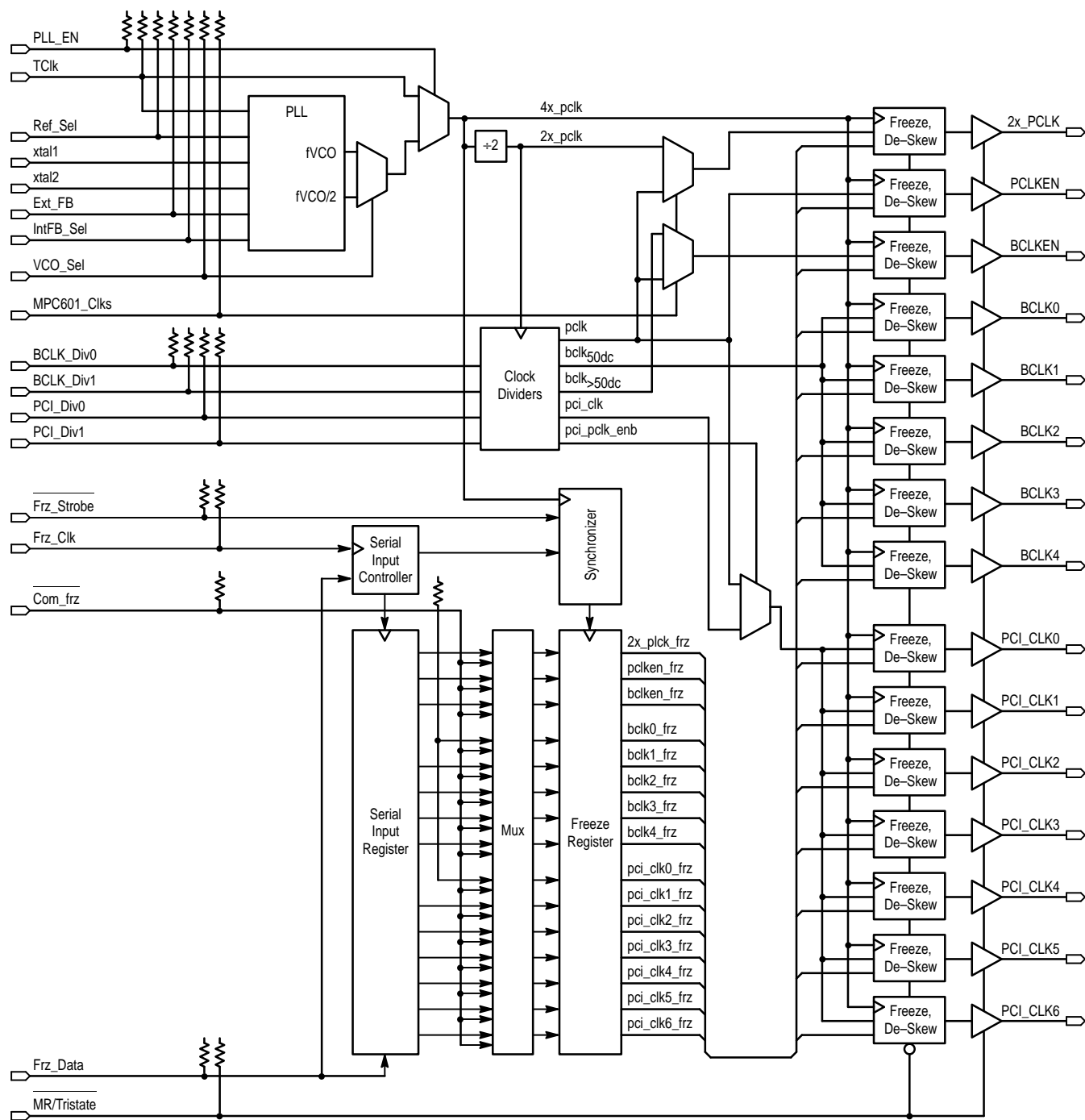


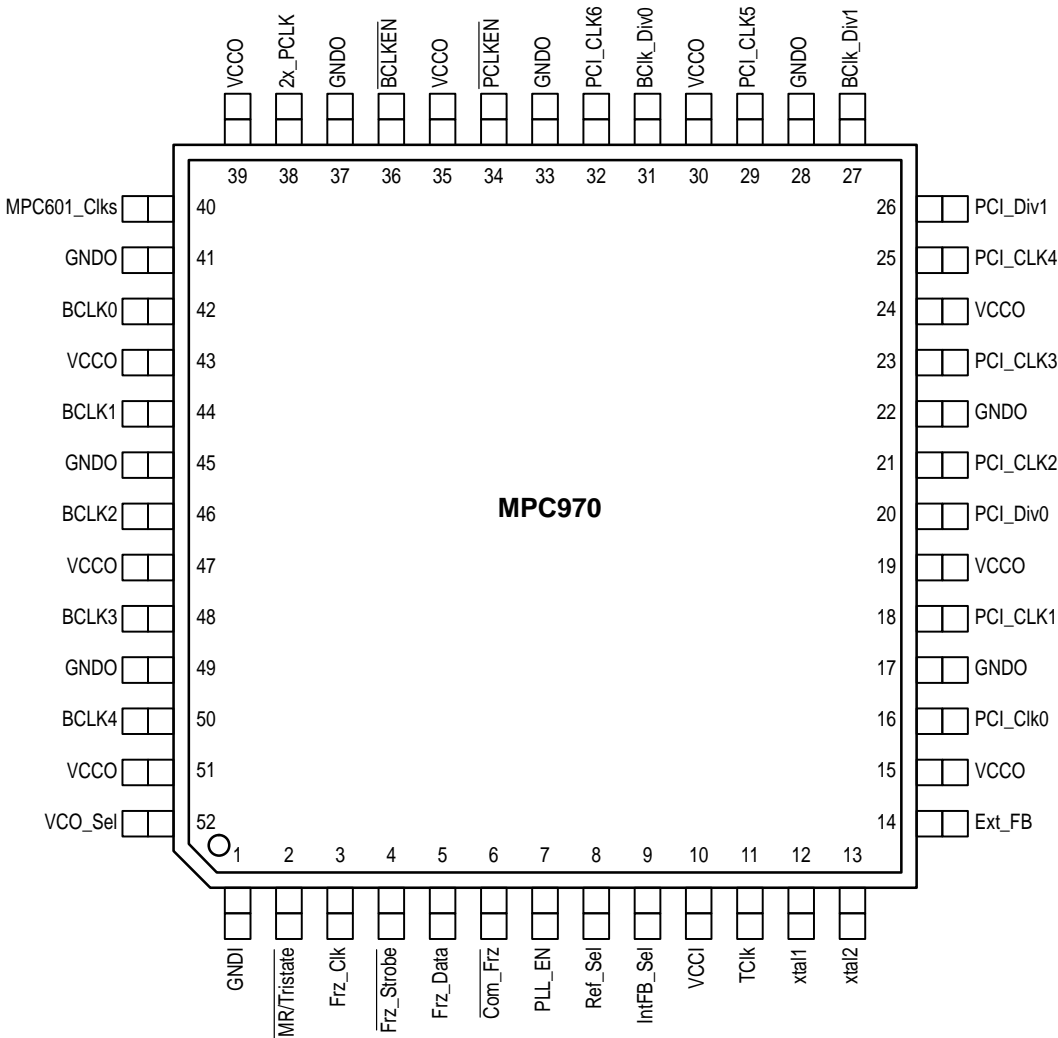
The MPC970 offers a very flexible output enable/disable scheme. This enable/disable scheme helps facilitate system debug as well as provide unique opportunities for system power down schemes to meet the requirements of “green” class machines. The MPC970 allows for the enabling of each output independently via a serial input port or a common enable/disable of all outputs simultaneously via a parallel control pin. When disabled or “frozen” the outputs will be locked in the “LOW” state, however the internal state machines will continue to run. Therefore when “unfrozen” the outputs will activate synchronous and in phase with those outputs which were not frozen. The freezing and unfreezing of outputs occurs only when they are already in the “LOW” state, thus the possibility of runt pulse generation is eliminated. A power-on reset will ensure that upon power up all of the outputs will be active.

For IC and board level testing a MR/Tristate input is provided. When pulled “LOW” all outputs will tristate and all internal flip flops will be reset. In addition the internal PLL can be bypassed and the fanout dividers and output buffers can be driven directly by the TClk input pin. Note that in this mode it will take a number of input clock pulses to cause output transitions as the TClk is fed through the internal dividers.

The MPC970 is fully 3.3V (3.6V for PowerPC 601 designs) compatible and requires no external loop filter components. All inputs accept LVCMOS/LVTTL compatible levels while the outputs provide LVCMOS levels with the capability to drive 50Ω transmission lines. For series terminated lines each MPC970 output can drive two 50Ω lines in parallel thus effectively doubling the fanout of the device.

## SIMPLIFIED BLOCK DIAGRAM





FUNCTION TABLE 1

MPC601_Clks	2x_PCLK	PCLKEN	BCLKEN	BCLK	PCI_CLK
0	VCO/4	VCO/4	VCO/4	X	X
1	VCO/2	VCO/4	BCLK*	X	X

\* Output is purposely delayed vs 2x\_PCLK output.

FUNCTION TABLE 2

PCI_Div1	PCI_Div0	PCI_CLK	BCLK_Div1	BCLK_Div0	BCLK
0	0	BLCK	0	0	PCLKEN
0	1	BCLK/2	0	1	PCLKEN/2
1	0	BCLK/3	1	0	PCLKEN/3
1	1	PCLKEN	1	1	PCLKEN/4

FUNCTION TABLE 3

Control Pin	Logic '0'	Logic '1'
VCO_Sel	fVCO/2	fVCO
Ref_Sel	TCLK	Crystal Osc
PLL_En	Bypass PLL	Enable PLL
IntFB_Sel	Ext Feedback	Int Feedback

**ABSOLUTE MAXIMUM RATINGS\***

Symbol	Parameter	Min	Max	Unit
V <sub>CC</sub>	Supply Voltage	−0.3	4.6	V
V <sub>I</sub>	Input Voltage	−0.3	V <sub>DD</sub> + 0.3	V
I <sub>IN</sub>	Input Current	TBD	TBD	mA
T <sub>Stor</sub>	Storage Temperature Range	−40	125	°C

\* Absolute maximum continuous ratings are those values beyond which damage to the device may occur. Exposure to these conditions or conditions beyond those indicated may adversely affect device reliability. Functional operation under absolute-maximum-rated conditions is not implied.

**DC CHARACTERISTICS** (T<sub>A</sub> = 0 to 70°C)

Symbol	Characteristic	Min	Max	Unit	Condition
V <sub>CC</sub>	Power Supply Voltage	3.0	3.8	V	
I <sub>CC</sub>	Quiescent Power Supply		250	mA	
V <sub>IL</sub>	Input Voltage LOW		0.3V <sub>DD</sub>	V	LVC MOS Inputs
V <sub>IH</sub>	Input Voltage HIGH	0.7V <sub>DD</sub>	TBD	V	LVC MOS Inputs
I <sub>IH</sub>	Input Current HIGH		−100	μA	V <sub>IN</sub> = V <sub>CC</sub>
I <sub>IL</sub>	Input Current LOW	−200		μA	V <sub>IN</sub> = GND
V <sub>OH</sub>	Output Voltage HIGH	V <sub>DD</sub> − 0.2		V	I <sub>OH</sub> = −20mA <sup>1</sup>
V <sub>OL</sub>	Output Voltage LOW		0.2	V	I <sub>OL</sub> = 20mA <sup>1</sup>
I <sub>OZ</sub>	Tristate Output Leakage Current	−10	10	μA	V <sub>OH</sub> = V <sub>CC</sub> or GND
C <sub>IN</sub>	Input Capacitance		4	pF	
C <sub>pd</sub>	Power Dissipation Capacitance			pF	
C <sub>OUT</sub>	Output Capacitance		8	pF	

1. The MPC970 outputs can drive series or parallel terminated 50Ω (or 50Ω to V<sub>CC</sub>/2) transmission lines on the incident edge (see Applications Info section).

**PLL INPUT REFERENCE CHARACTERISTICS** (T<sub>A</sub> = 0 to 70°C)

Symbol	Characteristic	Min	Max	Unit	Condition
t <sub>r</sub> , t <sub>f</sub>	TCLK Input Rise/Falls		3.0	ns	
f <sub>ref</sub>	Reference Input Frequency	10	Note 1	MHz	
f <sub>refDC</sub>	Reference Input Duty Cycle	25	75	%	

1. Maximum input reference is limited by the VCO lock range and the feedback divider.

**AC CHARACTERISTICS** ( $T_A = 0$  to  $70^\circ\text{C}$ )

Symbol	Characteristic	Min	Typ	Max	Unit	Condition
$f_{\text{Xtal}}$	Crystal Oscillator Frequency	10		25	MHz	Note 4
$F_{\text{out}}^2$	Maximum $2x\_PCLK$ Output Frequency			200	MHz	
$t_{\text{DC}}^{1,2}$	Output Duty Cycle	45 35		55 65	%	$F_{\text{out}} < 200\text{MHz}$ $F_{\text{out}} \geq 200\text{MHz}$
$V_{\text{OHAC}}$	AC Output HIGH Voltage	2.4 2.2			V	$F_{\text{out}} < 200\text{MHz}$ $F_{\text{out}} \geq 200\text{MHz}$
$V_{\text{OLAC}}$	AC Output LOW Voltage			0.4 0.6	V	$F_{\text{out}} < 200\text{MHz}$ $F_{\text{out}} \geq 200\text{MHz}$
$t_{\text{pw}}^{1,2}$	$2x\_PCLK$ Pulse Width	1.75	2.27		ns	$F_{\text{out}} = 200\text{MHz}$
$t_{\text{per}}$	Minimum Clock Out Period	4.85	4.91		ns	$F_{\text{out}} = 200\text{MHz}$
$t_{\text{pd}}^3$	SYNC to Feedback Propagation Delay		$X_1 \pm 150$ $X_2 \pm 150$ $X_3 \pm 150$ $X_4 \pm 150$ $X_5 \pm 150$ $X_6 \pm 150$ $X_7 \pm 150$		ps	
$f_{\text{VCO}}$	VCO Lock Range	Feedback = VCO/4 Feedback = VCO/8 Feedback = VCO/12 Feedback = VCO/16 Feedback = VCO/24 Feedback = VCO/32 Feedback = VCO/36 Feedback = VCO/36	TBD TBD TBD TBD TBD TBD TBD TBD	200–700 200–700 200–700 200–700 200–700 200–700 200–700 200–700	TBD TBD TBD TBD TBD TBD TBD TBD	MHz Ext Feedback Ext Feedback Ext Feedback Ext or Int Feedback Ext Feedback Int Feedback Ext Feedback Ext Feedback
$t_{\text{jitter}}^{1,2}$	Output Jitter		$\pm 50$ 110 76	$\pm 100$ 190 210	ps	PLL Jitter $2x\_P$ Period Variation Period Variation (Other)
$t_{\text{skew}}^{1,2}$	Output-to-Output Skew $2x\_PCLK$ , PCLKEN, BCLKEN, BCLK $2x\_PCLK$ , PCLKEN, BCLK PCI_CLK BCLK All			550 550 450 550 800	ps	MPC601_Clk = '0' MPC601_Clk = '1'
$t_{\text{delay}}$	Propagation Delay $2x\_PCLK$ to BCLKEN	100		850	ps	MPC601_Clk = '1'
$t_r, t_f^{1,2}$	Output Rise/Fall Time	0.15		1.5	ns	0.8 to 2.0V
$t_{\text{lock}}$	PLL Lock Time			10	ms	
$t_{\text{PZL}}$	Output Enable Time $\overline{\text{MR}}/\text{Tristate}$ to Outputs			8	ns	
$t_{\text{PHZ}}, t_{\text{PLZ}}$	Output Disable Time $\overline{\text{MR}}/\text{Tristate}$ to Outputs			10	ns	
$f_{\text{MAX}}$	Maximum Frz_Clk Frequency			20	MHz	
$t_s$	Setup Time Frz_Data to Frz_Clk Com_Frz to Frz_Strobe	8 5				
$t_h$	Hold Time Frz_Clk to Frz_Data Frz_Strobe to Com_Frz	8 5				

1. Measured at 1.4V.

2. Drive  $50\Omega$  transmission lines.3.  $X_1, X_2, X_3, X_4, X_5, X_6$ , and  $X_7$  all to be determined. The specs hold only when the 970 or 971 is used in the external feedback mode.

4. See Applications Info section for more crystal information.



## DETAILED PIN DESCRIPTIONS

The following gives a brief description of the functionality of the MPC970 I/O. Unless explicitly stated all inputs are LVCMOS/LVTTL compatible with internal pull up resistors. All outputs are LVCMOS level outputs which are capable of driving two series terminated 50Ω transmission lines on the incident edge.

### xtal1, xtal2

These input pairs are the pins which differentiate the MPC970 and the MPC971. For the MPC970 the xtal1 and xtal2 pins represent the external crystal connections to the internal oscillator. The crystal oscillator is completely self contained, there are no external components required. The oscillator is specified to function for crystals of up to 50MHz. Exact crystal specifications are outlined in the applications section.

### VCO\_Sel

The VCO\_Sel pin allows the user to further divide the internal VCO frequency for the generation of lower frequencies at the outputs. The VCO\_Sel pin should be used to set the VCO into its most optimum range. Refer to the applications section for more details on the VCO frequency range. A logic '1' on the VCO\_Sel pin will bypass the internal ÷2.

### TClk

The TClk input serves a dual purpose; it can be used as either a reference clock input for the PLL from an external frequency source or it can be used as a board level test clock in the PLL bypass mode.

### PLL\_En

The PLL\_En pin allows the TClk input to be routed around the PLL for system test and debug. When pulled low the MPC970 will be placed in the test mode. Note that the TClk input will be routed through the divider chain. For instance in the PowerPC 601 microprocessor clock generation mode the TClk input will toggle twice for each toggle on the 2x\_PCLK output. Depending on the states of the frequency divider select pins this ratio may be higher.

### Frz\_Data

Frz\_Data is the serial data input for the output freeze function of the device. Refer to the applications section for more information on the freeze functionality.

### Frz\_Clk

Frz\_Clk is the serial freeze logic clock input. Refer to the applications section for more information on the freeze functionality.

### Frz\_Strobe

The Frz\_Strobe input is used to freeze or unfreeze all of the outputs simultaneously. Refer to the applications section for more information on the freeze functionality.

### Com\_Frz

The Com\_Frz input allows the user to enable/disable all of the outputs with the control of a single pin. The action will take place upon a high to low transition of the Frz\_Strobe input.

### BCLK\_Div0:1

The BCLK\_Div inputs are used to program the VCO divide ratio for the BCLK outputs. These inputs also set the divide ratio of the BCLKEN output to be equal in frequency to the BCLK outputs when the device is in the MPC601\_Clks mode. The BCLK\_Div inputs set the frequency as follows:

BCLK_Div1	BCLK_Div0	BCLK Frequency
0	0	PCLKEN
0	1	PCLKEN/2
1	0	PCLKEN/3
1	1	PCLKEN/4

In most applications these inputs will be strapped to the appropriate power rails.

### PCI\_Div0:1

The PCI\_Div inputs set the division ratio for the PCI\_CLKs. The PCI\_CLKs are set relative to the BCLK or the PCLKEN output such that you can upgrade the processor bus and maintain the PCI bus frequency in the currently defined ≤ 33MHz range. The PCI\_Div inputs set the PCI\_CLKs as follows:

PCI_Div1	PCI_Div0	PCI_CLK Frequency
0	0	BCLK
0	1	BCLK/2
1	0	BCLK/3
1	1	PCLKEN

In a typical application these inputs will be strapped to the appropriate power rail.

### MPC601\_Clks

The MPC601\_Clks input will configure the outputs to drive the PowerPC 601 microprocessor when pulled HIGH or left open. When pulled LOW it will configure the 2xPCLK, PCLKEN and BCLKEN all into a VCO/4 mode. In this mode the MPC970 will have three more outputs available to drive clock loads on the processor bus for PowerPC 603, PowerPC 604 or Pentium microprocessor based systems.

### Ext\_FB

The Ext\_FB pin is an input to the phase detector of the PLL which is tied to an external feedback output. Typically this feedback will be one of the lowest frequency outputs of the MPC970.

### IntFB\_Sel

The IntFB\_Sel input selects whether the internal feedback signal or an external feedback signal is routed to the phase detector of the PLL. The default mode, pulled HIGH via the internal pull up resistor, is to select the internal feedback.

## MPC970

### MR/Tristate

The MR/Tristate input when pulled LOW will reset all of the internal flip flops and also tristate all of the clock outputs. This input is used primarily for IC and board level test.

### Ref\_Sel

The Ref\_Sel input allows the user to choose between two sources for the PLL reference frequency. For the MPC970, LOW on Ref\_Sel will choose the LVCMOS TCLK input. For the MPC970, a HIGH on Ref\_Sel will choose the crystal oscillator input.

### 2x\_PCLK

In general the outputs are named based on the implementation in a PowerPC 601 microprocessor based system. In the MPC601\_Clk mode the 2x\_PCLK will run at half the internal VCO frequency. With a maximum internal VCO frequency of 1000MHz this output could theoretically toggle at 500MHz, in practice however the output can toggle only as fast as 300MHz. This frequency will be required on future enhancements to the MPC 601 microprocessor. When the MPC970 is taken out of the MPC601\_Clk mode the 2xPCLK will run at a VCO/4 frequency. This divide ratio will place this output frequency in the present and future processor bus speeds of the PowerPC 603, PowerPC 604 and Pentium microprocessors. The 2x\_PCLK output is a 50% duty cycle LVCMOS output.

### PCLKEN

The PCLKEN output is designed to drive the PCLKEN input of the PowerPC 601 microprocessor when the MPC970 is in the MPC601\_Clk mode. The PCLKEN output frequency is one half that of the 2x\_PCLK output, a divide by four of the internal VCO. The PCLKEN output runs at the same frequency regardless of the state of the frequency divide controls. The toggle frequency of this output is well placed for driving the PowerPC 603, PowerPC 604 and Pentium processor buses. The PCLKEN output is a 50% duty cycle LVCMOS output.

### BCLKEN

The BCLKEN output is designed to drive the BCLKEN input of the PowerPC 601 microprocessor when the MPC970 is in the MPC601\_Clks mode. The BCLKEN toggles at the same frequency as the BCLK outputs as described earlier. However when the BCLKEN output is a divide by three or a divide by four of the PCLKEN output the duty cycle is 66/33 and 75/25 respectively per the requirement of the MPC 601 processor. In addition to meet the HOLD time spec for the BCLKEN input of the MPC 601 the BCLKEN output of the MPC970 lags the 2xPCLK output by no less than 100ps. When the MPC970 is not in the MPC601\_Clks mode the BCLKEN output is set at a fixed divide by four from the internal VCO. In addition in this mode the BCLKEN output does NOT lag the other outputs, but rather is synchronous within the Output-to-Output skew spec of the device.

### BCLK0:4

The BCLK outputs are designed to drive the clock loads on the processor bus of either the PowerPC or Pentium microprocessors. The most common practice in "non MPC 601" applications will be to place these outputs in the PCLKEN/1 mode and combine them with the above outputs to drive all of the loads on the processor bus. The division ratios do allow for the swap of these outputs with the PCI\_CLK outputs if more clocks are needed to drive the processor bus. For PowerPC 601 microprocessor based systems the division ratios allow the processor internal speeds to be increased while maintaining reasonable speeds for the L2 cache and the PCI bridge chip. The BCLK outputs are 50% duty cycle LVCMOS outputs.

### PCI\_Clk0:6

As the name would suggest the PCI\_CLK outputs are designed to drive the PCI bus clock loads in a typical microprocessor based system. The division ratios allow for these outputs to remain in the  $\leq 33$ MHz PCI bus speeds for various common processor bus speeds as well as higher future processor bus speeds. These outputs can also be programmed to run at the processor bus speeds if more processor bus clocks are required. The PCI\_CLK outputs are 50% duty cycle LVCMOS outputs.

## APPLICATIONS INFORMATION

### Programming the MPC970

The MPC970 is very flexible in the programming of the frequency relationships of the various outputs as well as the relationships between the input references and outputs. The purpose of this section is to outline the various relationships. Although not exhaustive the hope is that enough information is supplied to allow the customers to tailor the I/O relationships for their specific applications.

The VCO used in the MPC970 is a differential ring oscillator. The VCO exhibits a very wide frequency range to allow for a great deal of flexibility to the end user. Special design techniques were used in the overall PLL design to

keep the relatively high gain of the VCO from significantly impacting the jitter of the PLL.

Table 1 tabulates the various output frequencies for the different modes defined by the division select input pins. In this table the VCO\_Sel pin is high so that the  $\pm 2$  prescaler is bypassed. Note that the  $\pm 32$  feedback is always fed directly from the VCO and is thus unaffected by the level on the VCO\_Sel input. Table 1 shows each of the output frequencies as a function of the VCO frequency. The two VCO ranges can be used to plug in values to get the actual frequencies. When the internal feedback option is used the multiplication factor of the device will equal 32 divided by the

output divide ratio. If the VCO\_Sel pin is "LOW" the multiplication factor will be reduced further by 2. (See "Using the On-Board Crystal Oscillator" section of this datasheet.)

#### Using the External Feedback Feature of the MPC970/71

In applications where the relationship between the output waveforms and the input waveforms are critical the external feedback option will likely be used. Table 1 and Table 2 are still appropriate for establishing the potential output frequency relationships. The input reference frequency for external feedback applications will be equal to the frequency of the feedback signal. As a result the use of the external feedback yields a number of potential input to output frequency multiplication factors which are not available using the internal feedback. Using the external feedback the device can function as a zero delay buffer and could multiply the input from 4 to as much as 48. In practice however the multiplication factor is limited by the loop dynamics of the PLL. The MPC970 PLL was optimized for an input reference frequency or greater than 10MHz. Frequencies lower than 10MHz will tend to pass through the filter and add jitter to the PLL. In addition the PLL was optimized for feedback divide ratios of between 8 and 64. The user should avoid using the device with feedback divide ratios outside of this range. For the external feedback case the feedback divide ratio will include the  $\div 2$  (if VCO\_Sel is LOW) plus the output divider for

the feedback output. If, for instance the MPC970 is to be used as a zero delay buffer the VCO\_Sel pin should be pulled LOW and all of the outputs should be set in a VCO/4 mode. This would produce a feedback ratio of  $\div 8$ . Several potential configurations using the external feedback are pictured in Figure 1 through Figure 4.

The external feedback option of the MPC970 is critical for applications in which more than one clock driver need to be synchronized. The external feedback option ensures that the feed through delay is the same as the feedback delay. This functionality removes propagation delay variation as a factor in the determination of part to part skew. The low jitter PLL used in the MPC970 has a near zero deadband phase detector and very little part to part variability. The result is a very low phase error variability in the product. When coupled with the output to output skew the phase error variability accounts for the part to part skew of the device. From the specification table one sees that the worst case part to part skew of the device is 800ps, assuming that there is zero skew in the multiple reference inputs. For multiple MPC970 applications if the lowest generated output frequency is used as the feedback signal the devices will be guaranteed to be synchronized. For applications where the lowest frequency is not used as the reference or where the internal feedback is used there is no way to guarantee that the multiple devices will be phase synchronized.

**Table 1. Programmable Output Frequency Relationships** (MPC601\_Clk = 'HIGH'; VCO\_Sel = 'HIGH')

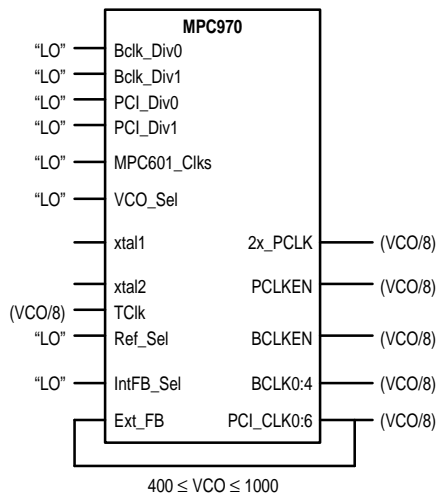
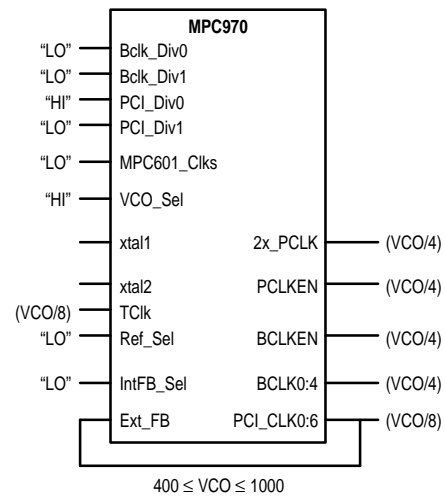
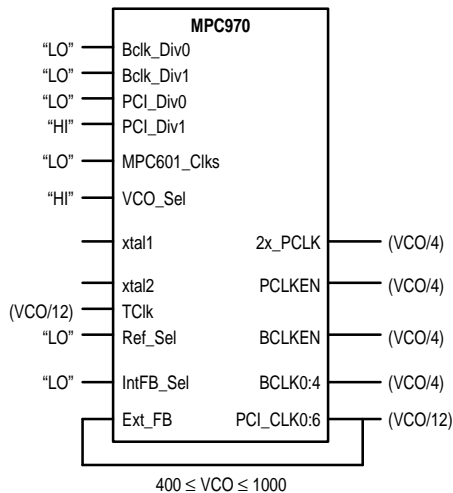
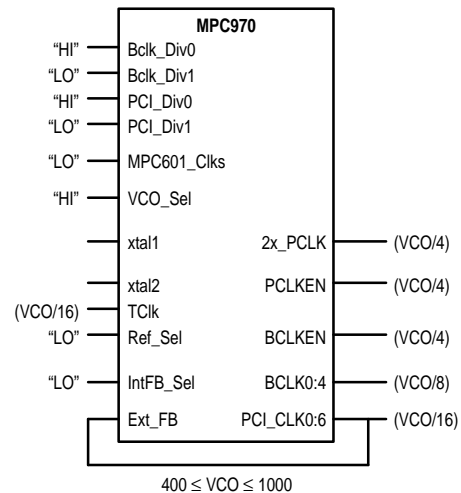
INPUTS				OUTPUTS				
PCI_Div1	PCI_Div0	BCLK_Div1	BCLK_Div0	2x_PCLK	PCLKEN	BCLKEN*	BCLK	PCI_CLK
0	0	0	0	VCO/2	VCO/4	VCO/4	VCO/4	VCO/4
0	0	0	1	VCO/2	VCO/4	VCO/8	VCO/8	VCO/8
0	0	1	0	VCO/2	VCO/4	VCO/12	VCO/12	VCO/12
0	0	1	1	VCO/2	VCO/4	VCO/16	VCO/16	VCO/16
0	1	0	0	VCO/2	VCO/4	VCO/4	VCO/4	VCO/8
0	1	0	1	VCO/2	VCO/4	VCO/8	VCO/8	VCO/16
0	1	1	0	VCO/2	VCO/4	VCO/12	VCO/12	VCO/24
0	1	1	1	VCO/2	VCO/4	VCO/16	VCO/16	VCO/32
1	0	0	0	VCO/2	VCO/4	VCO/4	VCO/4	VCO/12
1	0	0	1	VCO/2	VCO/4	VCO/8	VCO/8	VCO/24
1	0	1	0	VCO/2	VCO/4	VCO/12	VCO/12	VCO/36
1	0	1	1	VCO/2	VCO/4	VCO/16	VCO/16	VCO/48
1	1	0	0	VCO/2	VCO/4	VCO/4	VCO/4	VCO/4
1	1	0	1	VCO/2	VCO/4	VCO/8	VCO/8	VCO/4
1	1	1	0	VCO/2	VCO/4	VCO/12	VCO/12	VCO/4
1	1	1	1	VCO/2	VCO/4	VCO/16	VCO/16	VCO/4

\* BCLK\_En output is delayed relative to other outputs

**Table 2. Programmable Output Frequency Relationships** (MPC601\_Clk = 'LOW'; VCO\_Sel = 'HIGH')

INPUTS				OUTPUTS				
PCI_Div1	PCI_Div0	BCLK_Div1	BCLK_Div0	2x_PCLK	PCLKEN	BCLKEN*	BCLK	PCI_CLK
0	0	0	0	VCO/4	VCO/4	VCO/4	VCO/4	VCO/4
0	0	0	1	VCO/4	VCO/4	VCO/4	VCO/8	VCO/8
0	0	1	0	VCO/4	VCO/4	VCO/4	VCO/12	VCO/12
0	0	1	1	VCO/4	VCO/4	VCO/4	VCO/16	VCO/16
0	1	0	0	VCO/4	VCO/4	VCO/4	VCO/4	VCO/8
0	1	0	1	VCO/4	VCO/4	VCO/4	VCO/8	VCO/16
0	1	1	0	VCO/4	VCO/4	VCO/4	VCO/12	VCO/24
0	1	1	1	VCO/4	VCO/4	VCO/4	VCO/16	VCO/32
1	0	0	0	VCO/4	VCO/4	VCO/4	VCO/4	VCO/12
1	0	0	1	VCO/4	VCO/4	VCO/4	VCO/8	VCO/24
1	0	1	0	VCO/4	VCO/4	VCO/4	VCO/12	VCO/36
1	0	1	1	VCO/4	VCO/4	VCO/4	VCO/16	VCO/48
1	1	0	0	VCO/4	VCO/4	VCO/4	VCO/4	VCO/4
1	1	0	1	VCO/4	VCO/4	VCO/4	VCO/8	VCO/4
1	1	1	0	VCO/4	VCO/4	VCO/4	VCO/12	VCO/4
1	1	1	1	VCO/4	VCO/4	VCO/4	VCO/16	VCO/4

\* BCLK\_En output is coincident with other outputs

**Figure 1. External Feedback Configuration 1****Figure 2. External Feedback Configuration 2****Figure 3. External Feedback Configuration 3****Figure 4. External Feedback Configuration 4**

### Using the On-Board Crystal Oscillator

The MPC970 features an on-board crystal oscillator to allow for seed clock generation as well as final distribution. The on-board oscillator is completely self contained so that the only external component required is the crystal. As the oscillator is somewhat sensitive to loading on its inputs the user is advised to mount the crystal as close to the MPC970 as possible to avoid any board level parasitics. To facilitate co-location surface mount crystals are recommended, but not required. In addition, with crystals with a higher shunt capacitance, it may be necessary to place a 1k resistor across the two crystal leads.

The oscillator circuit is a series resonant circuit as opposed to the more common parallel resonant circuit, this eliminates the need for large on-board capacitors. Because the design is a series resonant design for the optimum frequency accuracy a series resonant crystal should be used (see specification table below). Unfortunately most off the shelf crystals are characterized in a parallel resonant mode. However a parallel resonant crystal is physically no different than a series resonant crystal, a parallel resonant crystal is simply a crystal which has been characterized in its parallel resonant mode. Therefore in the majority of cases a parallel specified crystal can be used with the MPC970 with just a minor frequency error due to the actual series resonant frequency of the parallel resonant specified crystal. Typically a parallel specified crystal used in a series resonant mode will exhibit an oscillatory frequency a few hundred ppm lower than the specified value. For most processor implementations a few hundred ppm translates into kHz inaccuracies, a level which does not represent a major issue.

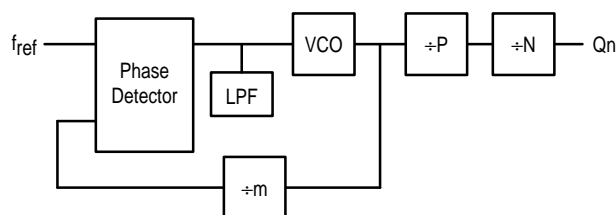
**Table 3. Crystal Specifications**

Parameter	Value
Crystal Cut	Fundamental at Cut
Resonance	Series Resonance*
Frequency Tolerance	±75ppm at 25°C
Frequency/Temperature Stability	±150pm 0 to 70°C
Operating Range	0 to 70°C
Shunt Capacitance	5–7pF
Equivalent Series Resistance (ESR)	50 to 80Ω
Correlation Drive Level	100μW
Aging	5ppm/Yr (First 3 Years)

\* See accompanying text for series versus parallel resonant discussion.

The MPC970 is a clock driver which was designed to generate outputs with programmable frequency relationships and not a synthesizer with a fixed input frequency. As a result the crystal input frequency is a function of the desired output frequency. For a design which utilizes the external feedback to the PLL the selection of the crystal frequency is straight forward; simply chose a crystal which is equal in frequency to the fed back signal. To determine the crystal required to

produce the desired output frequency for an application which utilizes internal feedback the block diagram of Figure 5 should be used. The P and the M values for the MPC970 are also included in Figure 5. The M values can be found in the configuration tables included in this applications section.



$$f_{ref} = \frac{f_{VCO}}{m}, \quad f_{VCO} = f_{Qn} \cdot N \cdot P$$

$$\therefore f_{ref} = \frac{f_{Qn} \cdot N \cdot P}{m}$$

$$m = 32$$

$$P = 1 \text{ (VCO\_Sel='1'), } 2 \text{ (VCO\_Sel='0')}$$

**Figure 5. PLL Block Diagram**

For the MPC970 clock driver, the following will provide an example of how to determine the crystal frequency required for a given design.

Given:

$$2x\_PCLK = 200\text{MHz}$$

$$PCLKEN = 100\text{MHz}$$

$$BCLK = 50\text{MHz}$$

$$PCI\_CLK = 25\text{MHz}$$

$$VCO\_SEL = '1'$$

$$f_{ref} = \frac{f_{Qn} \cdot N \cdot P}{m}$$

From Table 3

$$PCI\_CLK = VCO/16 \text{ then } N = 16$$

or

$$PCLKEN = VCO/4 \text{ then } N = 4$$

From Figure 5

$$m = 32 \text{ and } P = 1$$

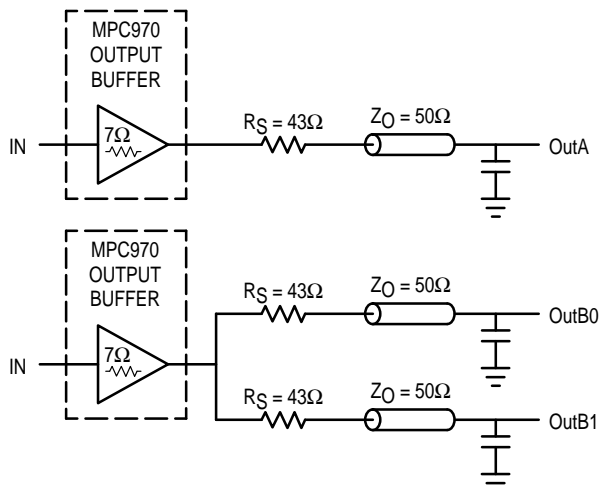
$$f_{ref} = \frac{25 \cdot 16 \cdot 1}{32} = 12.5\text{MHz} \text{ or } \frac{100 \cdot 4 \cdot 1}{32} = 12.5\text{MHz}$$

### Driving Transmission Lines

The MPC970 clock driver was designed to drive high speed signals in a terminated transmission line environment. To provide the optimum flexibility to the user the output drivers were designed to exhibit the lowest impedance possible. With an output impedance of less than 10Ω the drivers can drive either parallel or series terminated transmission lines. For more information on transmission lines the reader is referred to application note AN1091 in the Timing Solutions brochure (BR1333/D).

In most high performance clock networks point-to-point distribution of signals is the method of choice. In a point-to-point scheme either series terminated or parallel

terminated transmission lines can be used. The parallel technique terminates the signal at the end of the line with a  $50\Omega$  resistance to  $V_{CC}/2$ . This technique draws a fairly high level of DC current and thus only a single terminated line can be driven by each output of the MPC970 clock driver. For the series terminated case however there is no DC current draw, thus the outputs can drive multiple series terminated lines. Figure 6 illustrates an output driving a single series terminated line vs two series terminated lines in parallel. When taken to its extreme the fanout of the MPC970 clock driver is effectively doubled due to its capability to drive multiple lines.



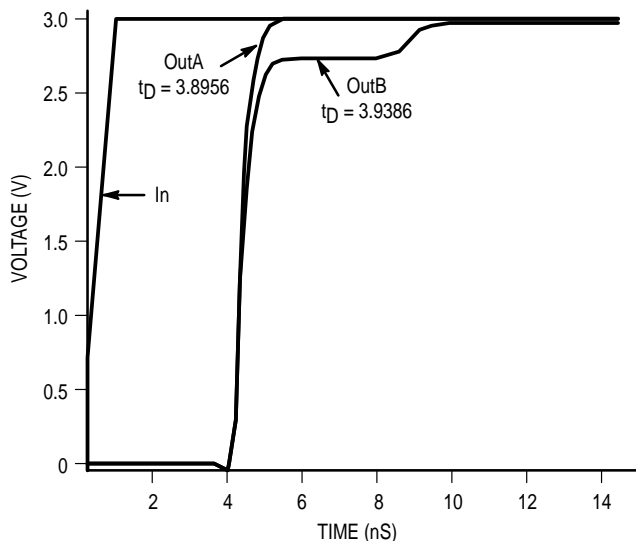
**Figure 6. Single versus Dual Transmission Lines**

The waveform plots of Figure 7 show the simulation results of an output driving a single line vs two lines. In both cases the drive capability of the MPC970 output buffers is more than sufficient to drive  $50\Omega$  transmission lines on the incident edge. Note from the delay measurements in the simulations a delta of only 43ps exists between the two differently loaded outputs. This suggests that the dual line driving need not be used exclusively to maintain the tight output-to-output skew of the MPC970. The output waveform in Figure 7 shows a step in the waveform, this step is caused by the impedance mismatch seen looking into the driver. The parallel combination of the  $43\Omega$  series resistor plus the output impedance does not match the parallel combination of the line impedances. The voltage wave launched down the two lines will equal:

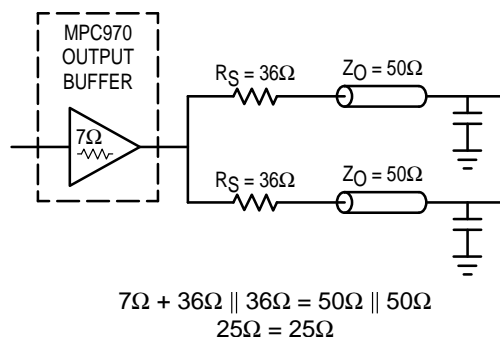
$$V_L = V_S (Z_o / R_s + R_o + Z_o) = 3.0 (25/53.5) = 1.40V$$

At the load end the voltage will double, due to the near unity reflection coefficient, to 2.8V. It will then increment towards the quiescent 3.0V in steps separated by one round trip delay (in this case 4.0ns).

Since this step is well above the threshold region it will not cause any false clock triggering, however designers may be uncomfortable with unwanted reflections on the line. To better match the impedances when driving multiple lines the situation in Figure 8 should be used. In this case the series terminating resistors are reduced such that when the parallel combination is added to the output buffer impedance the line impedance is perfectly matched.



**Figure 7. Single versus Dual Waveforms**



**Figure 8. Optimized Dual Line Termination**

SPICE level output buffer models are available for engineers who want to simulate their specific interconnect schemes. In addition IV characteristics are in the process of being generated to support the other board level simulators in general use.

### Using the Output Freeze Circuitry

With the recent advent of a “green” classification for computers the desire for unique power management among system designers is keen. The individual output enable control of the MPC970 allows designers, under software control, to implement unique power management schemes into their designs. Although useful, individual output control at the expense of one pin per output is too high, therefore a simple serial interface was derived to economize on the control pins.

The freeze control logic provides two mechanisms through which the MPC970 clock outputs may be frozen (stopped in the logic ‘0’ state):

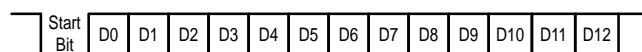
The first freeze mechanism allows serial loading of the 13-bit Serial Input Register, this register contains one programmable freeze enable bit for 13 of the 15 output clocks. The BCLK0 and PCI\_CLK0 outputs cannot be frozen with the serial port, this avoids any potential lock up situation

should an error occur in the loading of the Serial Input Register. The user may programmably freeze an output clock by writing logic '0' to the respective freeze enable bit. Likewise, the user may programmably unfreeze an output clock by writing logic '1' to the respective enable bit.

The second freeze mechanism allows all 15 clocks to be frozen simultaneously by placing a logic '0' on the Com\_Frz input and then issuing a low going pulse on the Frz\_Strobe input. Likewise, all 15 clocks can be simultaneously unfrozen by placing logic '1' on the Com\_Frz input and then issuing a low-going pulse on the Frz\_Strobe input. Note that all 15 clocks are affected by the Frz\_Strobe freeze logic.

The freeze logic will never force a newly-frozen clock to a logic '0' state before the time at which it would normally transition there. The logic simply keeps the frozen clock at logic '0' once it is there. Likewise, the freeze logic will never force a newly-unfrozen clock to a logic '1' state before the time at which it would normally transition there. The logic re-enables the unfrozen clock during the time when the respective clock would normally be in a logic '0' state, eliminating the possibility of 'runt' clock pulses.

The user may write to the Serial Input register through the Frz\_Data input by supplying a logic '0' start bit followed serially by 13 NRZ freeze enable bits. After the 13th freeze enable bit the Frz\_Data signal must be left in (or returned to) a logic '1' state (Figure 9). The period of each Frz\_Data bit equals the period of the free-running Frz\_Clk signal. The Frz\_Data serial transmission should be timed so the MPC970 can sample each Frz\_Data bit with the rising edge of the free-running Frz\_Clk signal.



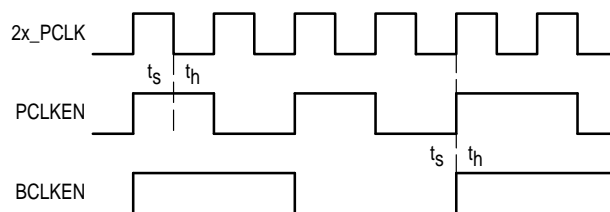
D0 is the control bit for 2x\_PCLK  
D1 is the control bit for PCLKEN  
D2 is the control bit for BCLKEN  
D3-D6 are the control bits for BCLK1-BCLK4  
D7-D12 are the control bits for PCI\_CLK1-PCI\_CLK6

**Figure 9. Freeze Data Input Protocol**

The user can combine the two freeze capabilities to simplify system level implementation. The serial input port can be used to establish the freeze mask to disable the appropriate outputs. The Frz\_Strobe input can then be used to unfreeze the outputs without having to serially load an "all unfrozen" freeze mask.

## Driving the PowerPC 601 Microprocessor

The MPC601 processor requires three clock inputs from the MPC970 clock driver. A 2x\_PCLK input at twice the internal MPC 601 clock rate and the PCLKEN and BCLKEN signals used to mask internal clock edges. The PCLKEN signal always runs at one half the 2x\_PCLK signal while the BCLKEN signal can run at 1x, 1/2x, 1/3x or 1/4x the PCLK input signal depending on the speed of the processor bus. When the BCLKEN signal is running at 1/3 or 1/4 the PCLK input the input duty cycle must be 66/33 and 75/25 respectively. In addition, as shown in Figure 10, to satisfy the BCLKEN to 2x\_PCLK Hold specification the BCLKEN signal must be at least coincident with the 2x\_PCLK edge. To simplify board level implementation it would be desirable that the BCLKEN signal actually lag the 2x\_PCLK by a few hundred picoseconds. The MPC970 insures that its BCLKEN output always lags the 2x\_PCLK input by at least 300ps.



**Figure 10. MPC601 Setup and Hold Times**

Table 4 illustrates some typical MPC 601 system frequencies which can be realized using the MPC970 clock driver.

**Table 4. Common MPC601 System Frequencies**

2x_PCLK	PCLK	BCLK	PCI_CLK
240	120	60(1/2x)	30(1/2x)
240	120	40(1/3x)	20(1/2x)
240	120	30(1/4x)	30(1x)
200	100	50(1/2x)	25(1/2x)
200	100	33(1/3x)	33(1x)
200	100	25(1/4x)	25(1x)
160	80	40(1/2x)	20(1/2x)
160	80	20(1/4x)	20(1x)
132	66	66(1x)	33(1/2x)
132	66	33(1/2x)	33(1x)

### Driving the PowerPC 603, PowerPC 604 and Pentium Microprocessors

The PowerPC 603, PowerPC 604 and Pentium processors differ from the MPC 601 processor in that the processor input clocks are at the same frequency as the processor bus. A typical system for these processors will include 8 – 16 clock loads on the processor bus. When the MPC970 is taken out of the MPC601\_Clk mode there are a total of 8 “non PCI\_CLK” outputs which can be run at the processor bus speeds for these microprocessors. Since each output can drive two series terminated transmission lines the MPC970 can support point to point clock distribution for up to 16 loads on the processor bus. In addition there will be 7 PCI\_CLK outputs which can drive up to 14 loads on the PCI bus.

If more clock loads are present on the processor bus the

PCI\_CLKs can be configured to drive processor bus clock loads in addition to the BCLKs or alternatively the clocking roles of the BCLKs and PCI\_CLKs can be reversed. Table 3 illustrates some useful frequency combinations for driving PowerPC 603, PowerPC 604 or Pentium microprocessor based systems.

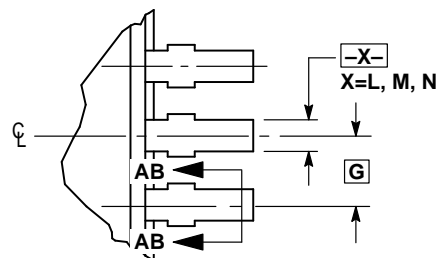
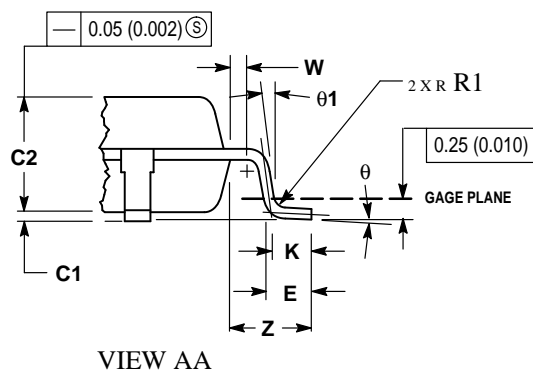
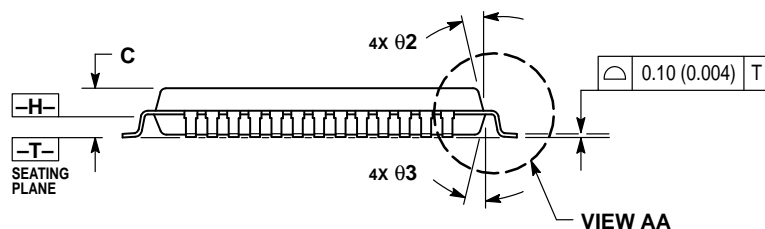
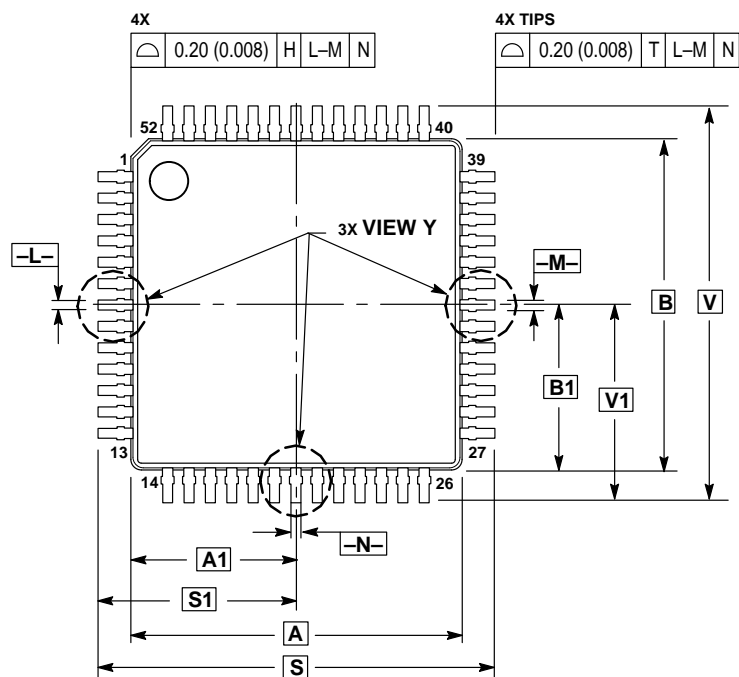
**Table Table 5. Common PowerPC 603, PowerPC 604 and Pentium System Frequencies**

2x_PCLK	PCLK	BCLK	PCI_CLK
80	80	80(1x)	26(1/3x)
75	75	75(1x)	25(1/3x)
66	66	66(1x)	33(1/2x)
66	66	66(1x)	66(1x)
66	66	33(1/2x)	66(PCLKEN)
60	60	60(1x)	30(1/2x)

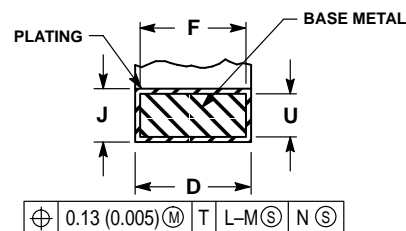


## OUTLINE DIMENSIONS

FA SUFFIX  
TQFP PACKAGE  
CASE 848D-03  
ISSUE C



VIEW Y




SECTION AB-AB  
ROTATED 90° CLOCKWISE

## NOTES:

1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE -H- IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS -L-, -M- AND -N- TO BE DETERMINED AT DATUM PLANE -H-.
5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE -T-.
6. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 (0.010) PER SIDE. DIMENSIONS A AND B DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE -H-.
7. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED 0.46 (0.018). MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION 0.07 (0.003).

DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	10.00 BSC		0.394 BSC	
A1	5.00 BSC		0.197 BSC	
B	10.00 BSC		0.394 BSC	
B1	5.00 BSC		0.197 BSC	
C	—	1.70	—	0.067
C1	0.05	0.20	0.002	0.008
C2	1.30	1.50	0.051	0.059
D	0.20	0.40	0.008	0.016
E	0.45	0.75	0.018	0.030
F	0.22	0.35	0.009	0.014
G	0.65 BSC		0.026 BSC	
J	0.07	0.20	0.003	0.008
K	0.50 REF		0.020 REF	
R1	0.08	0.20	0.003	0.008
S	12.00 BSC		0.472 BSC	
S1	6.00 BSC		0.236 BSC	
U	0.09	0.16	0.004	0.006
V	12.00 BSC		0.472 BSC	
V1	6.00 BSC		0.236 BSC	
W	0.20 REF		0.008 REF	
Z	1.00 REF		0.039 REF	
θ	0°	7°	0°	7°
θ1	0°	—	0°	—
θ2	12° REF		12° REF	
θ3	5°	13°	5°	13°

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

**How to reach us:**

**USA/EUROPE:** Motorola Literature Distribution;  
P.O. Box 20912; Phoenix, Arizona 85036. 1-800-441-2447

**MFAX:** RMFAX0@email.sps.mot.com -TOUCHTONE (602) 244-6609  
**INTERNET:** <http://Design-NET.com>

**JAPAN:** Nippon Motorola Ltd.; Tatsumi-SPD-JLDC, Toshikatsu Otsuki,  
6F Seibu-Butsuryu-Center, 3-14-2 Tatsumi Koto-Ku, Tokyo 135, Japan. 03-3521-8315

**HONG KONG:** Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,  
51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298

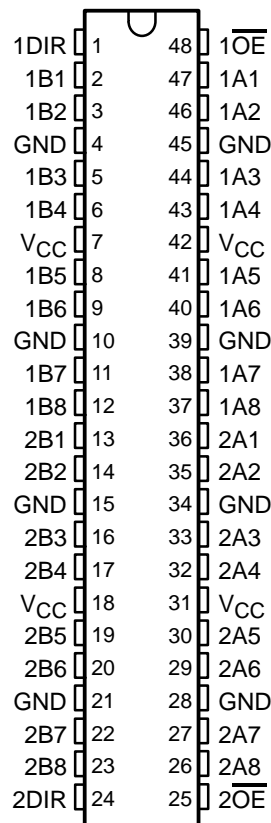


# SN54LVTH16245A, SN74LVTH16245A 3.3-V ABT 16-BIT BUS TRANSCEIVERS WITH 3-STATE OUTPUTS

SCBS143F – MAY 1992 – REVISED AUGUST 1996

- State-of-the-Art Advanced BiCMOS Technology (ABT) Design for 3.3-V Operation and Low-Static Power Dissipation
- Members of the Texas Instruments *Widebus™* Family
- High-Impedance State During Power Up and Power Down
- Support Mixed-Mode Signal Operation (5-V Input and Output Voltages With 3.3-V  $V_{CC}$ )
- Support Unregulated Battery Operation Down to 2.7 V
- Typical  $V_{OLP}$  (Output Ground Bounce)  $< 0.8$  V at  $V_{CC} = 3.3$  V,  $T_A = 25^\circ\text{C}$
- ESD Protection Exceeds 2000 V Per MIL-STD-883, Method 3015; Exceeds 200 V Using Machine Model ( $C = 200$  pF,  $R = 0$ )
- Latch-Up Performance Exceeds 500 mA Per JEDEC Standard JESD-17
- Bus-Hold Data Inputs Eliminate the Need for External Pullup/Pulldown Resistors
- Power Off Disables Inputs/Outputs, Permitting Live Insertion
- Package Options Include Plastic 300-mil Shrink Small-Outline (DL) and Thin Shrink Small-Outline (DGG) Packages and 380-mil Fine-Pitch Ceramic Flat (WD) Packages Using 25-mil Center-to-Center Spacings

SN54LVTH16245A . . . WD PACKAGE  
SN74LVTH16245A . . . DGG OR DL PACKAGE  
(TOP VIEW)



## description

The 'LVTH16245A are 16-bit (dual-octal) noninverting 3-state transceivers designed for low-voltage (3.3-V)  $V_{CC}$  operation, but with the capability to provide a TTL interface to a 5-V system environment.

These devices can be used as two 8-bit transceivers or one 16-bit transceiver. They allow data transmission from the A bus to the B bus or from the B bus to the A bus, depending on the logic level at the direction-control (DIR) input. The output-enable ( $\overline{OE}$ ) input can be used to disable the devices so that the buses are effectively isolated.

Active bus-hold circuitry is provided to hold unused or floating data inputs at a valid logic level.

When  $V_{CC}$  is between 0 and 1.5-V, the devices are in the high-impedance state during power up or power down. However, to ensure the high-impedance state above 1.5-V,  $\overline{OE}$  should be tied to  $V_{CC}$  through a pullup resistor; the minimum value of the resistor is determined by the current-sinking capability of the driver.



Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

Widebus is a trademark of Texas Instruments Incorporated.

UNLESS OTHERWISE NOTED this document contains PRODUCTION DATA information current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

**TEXAS  
INSTRUMENTS**

POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

Copyright © 1996, Texas Instruments Incorporated

SN54LVTH16245A, SN74LVTH16245A  
3.3-V ABT 16-BIT BUS TRANSCEIVERS  
WITH 3-STATE OUTPUTS

SCBS143F – MAY 1992 – REVISED AUGUST 1996

description (continued)

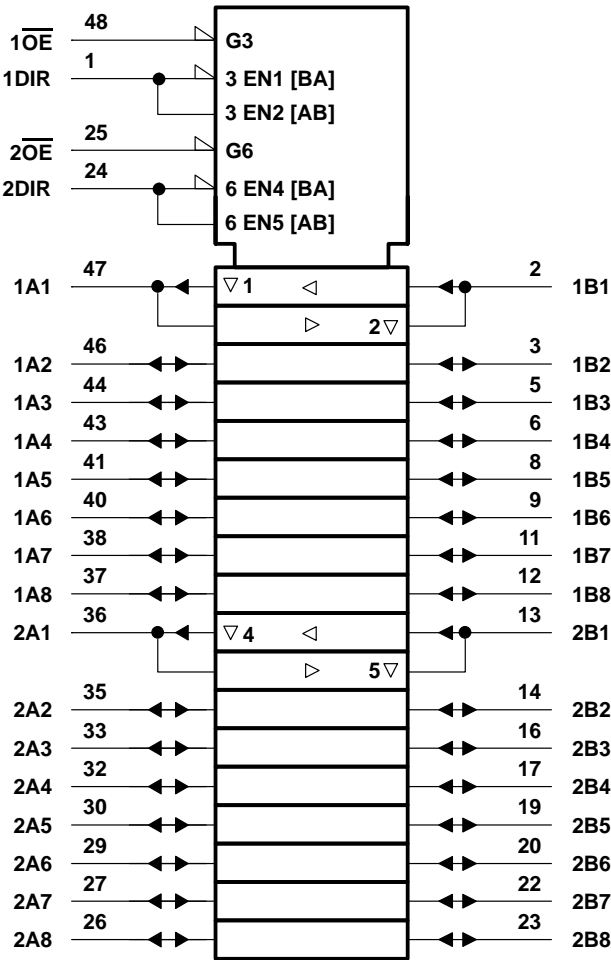
The SN74LVTH16245A is available in TI's shrink small-outline (DL) and thin shrink small-outline (DGG) packages, which provide twice the I/O pin count and functionality of standard small-outline packages in the same printed-circuit-board area.

The SN54LVTH16245A is characterized for operation over the full military temperature range of –55°C to 125°C. The SN74LVTH16245A is characterized for operation from –40°C to 85°C.

FUNCTION TABLE  
(each 8-bit section)

INPUTS		OPERATION
$\overline{\text{OE}}$	DIR	
L	L	B data to A bus
L	H	A data to B bus
H	X	Isolation

logic symbol†

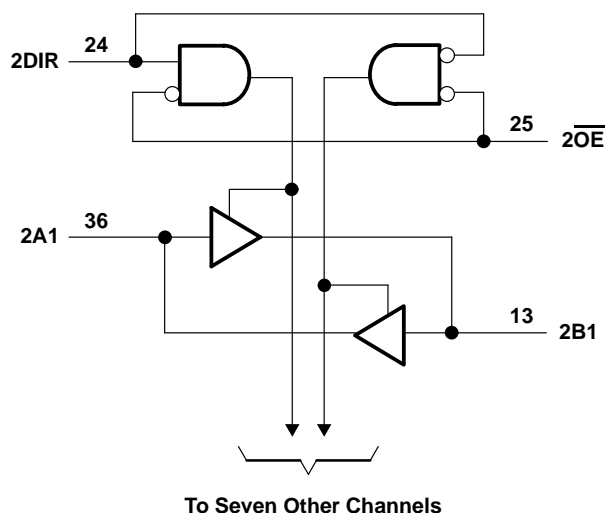
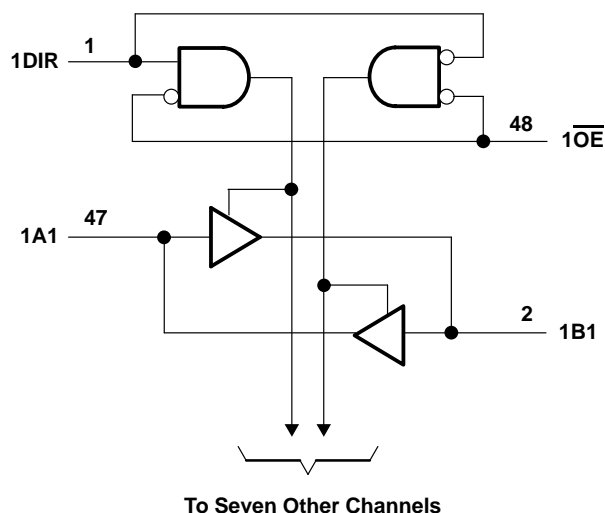


† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

# SN54LVTH16245A, SN74LVTH16245A 3.3-V ABT 16-BIT BUS TRANSCEIVERS WITH 3-STATE OUTPUTS

SCBS143F – MAY 1992 – REVISED AUGUST 1996

## logic diagram (positive logic)



## absolute maximum ratings over operating free-air temperature range (unless otherwise noted)<sup>†</sup>

Supply voltage range, $V_{CC}$	–0.5 V to 4.6 V
Input voltage range, $V_I$ (see Note 1)	–0.5 V to 7 V
Voltage range applied to any output in the high state or power-off state, $V_O$ (see Note 1)	–0.5 V to 7 V
Current into any output in the low state, $I_O$ : SN54LVTH16245A	96 mA
SN74LVTH16245A	128 mA
Current into any output in the high state, $I_O$ (see Note 2): SN54LVTH16245A	48 mA
SN74LVTH16245A	64 mA
Input clamp current, $I_{IK}$ ( $V_I < 0$ )	–50 mA
Output clamp current, $I_{OK}$ ( $V_O < 0$ )	–50 mA
Maximum power dissipation at $T_A = 55^\circ\text{C}$ (in still air) (see Note 3): DGG package	0.85 W
DL package	1.2 W
Storage temperature range, $T_{stg}$	–65°C to 150°C

<sup>†</sup> Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

- NOTES:
1. The input and output negative-voltage ratings may be exceeded if the input and output clamp-current ratings are observed.
  2. This current flows only when the output is in the high state and  $V_O > V_{CC}$ .
  3. The maximum package power dissipation is calculated using a junction temperature of 150°C and a board trace length of 750 mils. For more information, refer to the *Package Thermal Considerations* application note in the *ABT Advanced BiCMOS Technology Data Book*.

# SN54LVTH16245A, SN74LVTH16245A

## 3.3-V ABT 16-BIT BUS TRANSCEIVERS

### WITH 3-STATE OUTPUTS

SCBS143F – MAY 1992 – REVISED AUGUST 1996

#### recommended operating conditions (see Note 4)

			SN54LVTH16245A		SN74LVTH16245A		UNIT
			MIN	MAX	MIN	MAX	
$V_{CC}$	Supply voltage		2.7	3.6	2.7	3.6	V
$V_{IH}$	High-level input voltage		2		2		V
$V_{IL}$	Low-level input voltage			0.8		0.8	V
$V_I$	Input voltage			5.5		5.5	V
$I_{OH}$	High-level output current			–24		–32	mA
$I_{OL}$	Low-level output current			48		64	mA
$\Delta t/\Delta v$	Input transition rise or fall rate	Outputs enabled		10		10	ns/V
$\Delta t/\Delta V_{CC}$	Power-up ramp rate		200		200		$\mu$ s/V
$T_A$	Operating free-air temperature		–55	125	–40	85	°C

NOTE 4: Unused control inputs must be held high or low to prevent them from floating.

# SN54LVTH16245A, SN74LVTH16245A

## 3.3-V ABT 16-BIT BUS TRANSCEIVERS

### WITH 3-STATE OUTPUTS

SCBS143F – MAY 1992 – REVISED AUGUST 1996

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS		SN54LVTH16245A			SN74LVTH16245A			UNIT
			MIN	TYP†	MAX	MIN	TYP†	MAX	
$V_{IK}$	$V_{CC} = 2.7\text{ V}$ , $I_I = -18\text{ mA}$				-1.2			-1.2	V
$V_{OH}$	$V_{CC} = \text{MIN to MAX}^\ddagger$ , $I_{OH} = -100\text{ }\mu\text{A}$		$V_{CC}-0.2$			$V_{CC}-0.2$			V
	$V_{CC} = 2.7\text{ V}$ , $I_{OH} = -8\text{ mA}$		2.4			2.4			
	$V_{CC} = 3\text{ V}$	$I_{OH} = -24\text{ mA}$	2						
		$I_{OH} = -32\text{ mA}$				2			
$V_{OL}$	$V_{CC} = 2.7\text{ V}$	$I_{OL} = 100\text{ }\mu\text{A}$			0.2			0.2	V
		$I_{OL} = 24\text{ mA}$			0.5			0.5	
	$V_{CC} = 3\text{ V}$	$I_{OL} = 16\text{ mA}$			0.4			0.4	
		$I_{OL} = 32\text{ mA}$			0.5			0.5	
		$I_{OL} = 48\text{ mA}$			0.55				
		$I_{OL} = 64\text{ mA}$						0.55	
$I_I$	$V_{CC} = 3.6\text{ V}$ , $V_I = V_{CC}\text{ or GND}$	Control inputs			$\pm 1$			$\pm 1$	$\mu\text{A}$
	$V_{CC} = 0\text{ or MAX}^\ddagger$ , $V_I = 5.5\text{ V}$				10			10	
	$V_{CC} = 3.6\text{ V}$	$V_I = 5.5\text{ V}$			20			20	
		$V_I = V_{CC}$			1			1	
		$V_I = 0$			-5			-5	
$I_{off}$	$V_{CC} = 0$ , $V_I\text{ or }V_O = 0\text{ to }4.5\text{ V}$							$\pm 100$	$\mu\text{A}$
$I_{I(\text{hold})}$	$V_{CC} = 3\text{ V}$	$V_I = 0.8\text{ V}$			75			75	$\mu\text{A}$
		$V_I = 2\text{ V}$			-75			-75	
$I_{OZPU}^\S$	$V_{CC} = 0\text{ to }1.5\text{ V}$ , $V_O = 0.5\text{ V to }3\text{ V}$ , $\overline{OE} = 0$				$\pm 100$			$\pm 100$	$\mu\text{A}$
$I_{OZPD}^\S$	$V_{CC} = 0\text{ to }1.5\text{ V}$ , $V_O = 0.5\text{ V to }3\text{ V}$ , $\overline{OE} = 0$				$\pm 100$			$\pm 100$	$\mu\text{A}$
$I_{CC}^\#$	$V_{CC} = 3.6\text{ V}$ , $I_O = 0$ , $V_I = V_{CC}\text{ or GND}$	Outputs high			0.19			0.19	mA
		Outputs low			5			5	
		Outputs disabled			0.19			0.19	
$\Delta I_{CC}$	$V_{CC} = 3\text{ V to }3.6$ , One input at $V_{CC} - 0.6\text{ V}$ , Other inputs at $V_{CC}\text{ or GND}$				0.3			0.2	mA
$C_i$	$V_I = 3\text{ V or }0$				4			4	pF
$C_{io}$	$V_O = 3\text{ V or }0$				10			10	pF

† All typical values are at  $V_{CC} = 3.3\text{ V}$ ,  $T_A = 25^\circ\text{C}$ .

‡ For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.

§ Unused pins at  $V_{CC}\text{ or GND}$

¶ This parameter is warranted by characterization but not production tested.

# This is the increase in supply current for each input that is at the specified TTL voltage level rather than  $V_{CC}\text{ or GND}$ .

# SN54LVTH16245A, SN74LVTH16245A

## 3.3-V ABT 16-BIT BUS TRANSCEIVERS

### WITH 3-STATE OUTPUTS

SCBS143F – MAY 1992 – REVISED AUGUST 1996

switching characteristics over recommended operating free-air temperature range,  $C_L = 50$  pF (unless otherwise noted) (see Figure 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	SN54LVTH16245A				SN74LVTH16245A				UNIT	
			$V_{CC} = 3.3\text{ V} \pm 0.3\text{ V}$		$V_{CC} = 2.7\text{ V}$		$V_{CC} = 3.3\text{ V} \pm 0.3\text{ V}$			$V_{CC} = 2.7\text{ V}$		
			MIN	MAX	MIN	MAX	MIN	TYP†	MAX	MIN		MAX
t <sub>PLH</sub>	A or B	B or A	1.5	3.5	3.9		1.5	2.3	3.3	3.7		ns
t <sub>PHL</sub>			1.3	3.5	3.7		1.3	2.1	3.3	3.5		
t <sub>PZH</sub>	$\overline{\text{OE}}$	A or B	1.5	4.8	5.6		1.5	2.8	4.5	5.3		ns
t <sub>PZL</sub>			1.6	4.8	5.4		1.6	2.9	4.6	5.2		
t <sub>PHZ</sub>	$\overline{\text{OE}}$	A or B	2.3	5.5	5.8		2.3	3.7	5.1	5.5		ns
t <sub>PLZ</sub>			2.2	5.5	5.7		2.2	3.5	5.1	5.4		

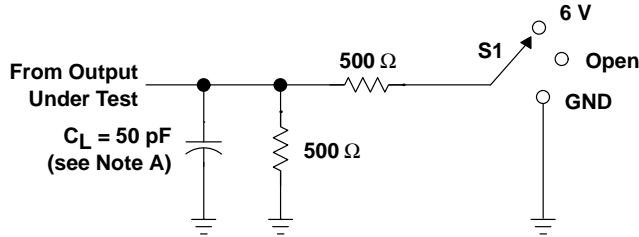
† All typical values are at  $V_{CC} = 3.3\text{ V}$ ,  $T_A = 25^\circ\text{C}$ .



# SN54LVTH16245A, SN74LVTH16245A 3.3-V ABT 16-BIT BUS TRANSCEIVERS WITH 3-STATE OUTPUTS

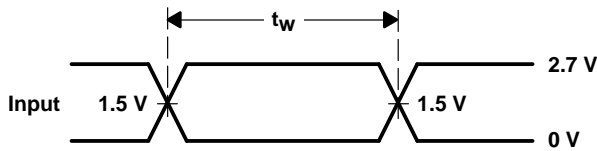
SCBS143F – MAY 1992 – REVISED AUGUST 1996

## PARAMETER MEASUREMENT INFORMATION

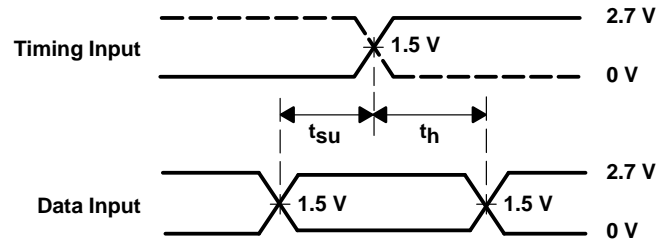


LOAD CIRCUIT

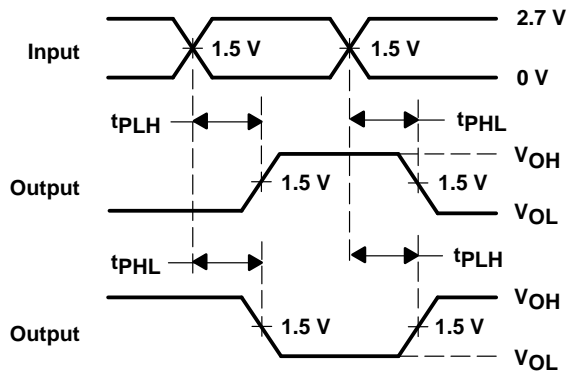
TEST	S1
$t_{PLH}/t_{PHL}$	Open
$t_{PLZ}/t_{PZL}$	6 V
$t_{PHZ}/t_{PZH}$	GND



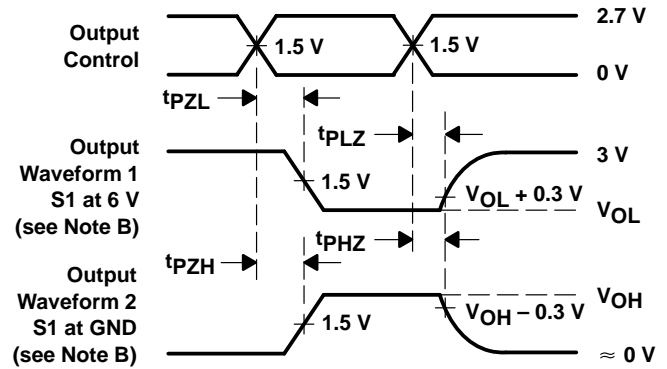
VOLTAGE WAVEFORMS  
PULSE DURATION



VOLTAGE WAVEFORMS  
SETUP AND HOLD TIMES



VOLTAGE WAVEFORMS  
PROPAGATION DELAY TIMES  
INVERTING AND NONINVERTING OUTPUTS



VOLTAGE WAVEFORMS  
ENABLE AND DISABLE TIMES  
LOW- AND HIGH-LEVEL ENABLING

- NOTES:
- A.  $C_L$  includes probe and jig capacitance.
  - B. Waveform 1 is for an output with internal conditions such that the output is low except when disabled by the output control. Waveform 2 is for an output with internal conditions such that the output is high except when disabled by the output control.
  - C. All input pulses are supplied by generators having the following characteristics:  $PRR \leq 10 \text{ MHz}$ ,  $Z_O = 50 \Omega$ ,  $t_r \leq 2.5 \text{ ns}$ ,  $t_f \leq 2.5 \text{ ns}$ .
  - D. The outputs are measured one at a time with one transition per measurement.

Figure 1. Load Circuit and Voltage Waveforms

## **IMPORTANT NOTICE**

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

**TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.**

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.