80960Jx Processor

Specification Update

December 1998

Notice: The 80960 Jx processors may contain design defects or errors known as errata which may cause the product's behavior to deviate from published specifications. Current characterized errata are documented in this specification update.

Order Number: 272852-006

Information in this document is provided in connection with Intel products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Intel's Terms and Conditions of Sale for such products, Intel assumes no liability whatsoever, and Intel disclaims any express or implied warranty, relating to sale and/or use of Intel products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Intel products are not intended for use in medical, life saving, or life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked “reserved” or “undefined.” Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The 80960Jx processors may contain design defects or errors known as errata which may cause the products to deviate from published specifications. Current characterized errata are documented in this specification update.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at http://www.intel.com.

Contents

Revision History

Date	Version	Description
12/15/98	006	Added Document Changes #4: Orientation of Pinout Diagrams for 196-Ball MPGBA Changed and #5: AC Timings Diagrams Do Not Apply to 80960JT Processor. Added 80960JS/JC processor information to the Stepping Register.
05/27/98	005	Removed references to 80960JA/JF and 80960JD data sheets (both combined into new 80960JA/JF/JD/JT data sheet). Added Specification Change # 2: DC Specification Changes Revised Document Change # 3: Index Update for i960® Jx Microprocessor Developer's Manual Revised 80960JD A-0 & A-2 Device and Stepping - Identifier in g0
04/09/98	004	Added new errata #17: Power Supply Sequence Can Damage Internal Diodes. Incorporated 80960JT processor into this specification update. Added Document Change # 3: Index Update for i960® Jx Microprocessor Developer's Manual.
02/03/98	003	Updated "Errata" section to reflect C-0 stepping. Added Specification Clarification #10: Tlx timing specification derating versus AD bus capacitive loading. Updated "Document Changes" section to reflect new Jx user manual.
04/09/97	002	Some improved AC timings HALT Mode ICC current corrections errata #15 modified, Errata does not effect application T_C max data sheet correction
07/01/96	001	This is the new Specification Update document. It contains all identified errata published on or before this date.
11/29/95	2.2	Added new errata #16: V_{IH} level on TRST#/RESET# Greater than Specified in Data Sheets.
10/31/95	2.1	Added new errata #15: Actual Max Tov Greater than Specified in Data Sheets.
3/30/95	2.0	Added the to the errata sheet. Added the A-2 stepping, whose only difference from the A-0 stepping is the fix of the RDYRCV# errata during a Th(hold cycle). A new technote was added to the list of 80960Jx technotes.

Date	Version	Description
3/10/95	1.9	Added new errata section User's Manual Errata containing only application critical users's manual errata. Added new user's manual errata #1: Manual Errata: IPND should not be Modified using a MEM Format Instruction. Two lines of code in errata #4 Fault Stack Alignment were changed as shown: from: sele r5, r10, r8 to: sele r5, r10, r4 from: sele r5, r10, r9 to: sele r5, r10, r5
2/9/95	1.8	Added new errata #13: Data Breakpoints on System Procedure Entries are Lost for Certain Fault Types**.**
1/23/95	1.7	Added new errata #12: "balx" Instruction Does Not Branch When targ and dst Use the Same Register.

Preface

As of July, 1996, Intel's Computing Enhancement Group has consolidated available historical device and documentation errata into this new document type called the Specification Update. We have endeavored to include all documented errata in the consolidation process, however, we make no representations or warranties concerning the completeness of the Specification Update.

This document is an update to the specifications contained in the Affected Documents/Related Documents table below. This document is a compilation of device and documentation errata, specification clarifications and changes. It is intended for hardware system manufacturers and software developers of applications, operating systems, or tools.

Information types defined in Nomenclature are consolidated into the specification update and are no longer published in other documents.

This document may also contain information that was not previously published.

Affected Documents/Related Documents

Title	Order #
i960® Jx Microprocessor Developer's Manual	272483
80960JA/JF Embedded 32-Bit Microprocessor datasheet	272504
80L960JA/JF 3.3 V Embedded 32-Bit Microprocessor datasheet	272744
80960JD Embedded 32-Bit Microprocessor datasheet	272596
80960JA/JF/JD/JT 3.3 V Microprocessor datasheet	273159
80960JS/JC 3.3 V Microprocessor datasheet	273200
AP- 712: DRAM Controller for i960® JA/JF/JD Processors	272674
AP-716: Architectural Comparison 80960Cx/Jx/Hx	272694
AP-727: Interfacing the i960® Jx Processor to NEC SAR	272779

Nomenclature

Errata are design defects or errors. These may cause the 80960Jx processor's behavior to deviate from published specifications. Hardware and software designed to be used with any given stepping must assume that all errata documented for that stepping are present on all devices.

Specification Changes are modifications to the current published specifications. These changes will be incorporated in any new release of the specification.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in any new release of the specification.

Documentation Changes include typos, errors, or omissions from the current published specifications. These will be incorporated in any new release of the specification.

Note: Errata and specification changes remain in the specification update throughout the product's lifecycle, or until a particular stepping is no longer commercially available. Under these circumstances, errata and specification changes removed from the specification update are archived and available upon request. Specification clarifications and documentation changes are removed from the specification update when the appropriate changes are made to the appropriate product specification or user documentation (datasheets, manuals, etc.).

Summary Table of Changes

The following table indicates the errata, specification changes, specification clarifications, or documentation changes which apply to the 80960Jx processor product. Intel may fix some of the errata in a future stepping of the component, and account for the other outstanding issues through documentation or specification changes as noted. This table uses the following notations:

Codes Used in Summary Table

Stepping

X: Errata exists in the stepping indicated. Specification Change or Clarification that applies to this stepping.

(No mark)

or (Blank box): This erratum is fixed in listed stepping or specification change does not apply to listed stepping.

Page

(Page): Page location of item in this document.

Status

Doc: Document change or update will be implemented.

Fix: This erratum is intended to be fixed in a future step of the component.

Fixed: This erratum has been previously fixed.

NoFix: There are no plans to fix this erratum.

Eval: Plans to fix this erratum are under evaluation.

Row

Change bar to left of table row indicates this erratum is either new or modified from the previous version of the document.

Errata

No.	Steppings JA/JF/JD			Steppings JT		Page	Status	ERRATA
	A-0	A-2	C-0	A-0	A-1			
1	X					13	Fixed	RDYRCV# Restriction During Ta, Th and Ti Cycles (A-0 Stepping Only)
2		X				13	Fixed	RDYRCV# Restriction During Ta and Ti Cycles (A-2 Stepping Only)
3	X	X				14	Fixed	System-Local Fault Calls Use System-Supervisor Trace Enable Bit
4	X	X				14	Fixed	Instructions “inten” and “intdis” Not Fully Implemented
5	X	X				15	Fixed	Fault Stack Alignment
6	X	X				17	Fixed	Software Interrupt Erratum
7	X	X				17	Fixed	Pullup on LOCK#/ONCE# Pin Does Not Turn Off
8	X	X				18	Fixed	Software Reinitialization Values in LMMR0, LMMR1, DLMCON
9	X	X				18	Fixed	Power Supply Current (I_{CC}) Higher than Anticipated
10	X	X				18	Fixed	One Cycle Performance Hit Due to Instruction Order with LOAD Instructions
11	X	X				18	Fixed	“divi” Instruction Performance Hit When src2 = dst
12	X	X				18	Fixed	Instructions Executed Between Back-to-Back Interrupts
13	X	X				19	Fixed	“balx” Instruction Does Not Branch When targ and dst Use the Same Register
14	X	X				19	Fixed	Data Breakpoints on System Procedure Entries are Lost for Certain Fault Types
15	X	X				20	Fixed	Actual Max T_{OV} Greater than Specified in Data Sheets
16	X	X				20	Fixed	V_{IH} level on TRST#/RESET# Greater than Specified in Data Sheets
17	X	X	X	X	X	20	NoFix	Power Supply Sequence Can Damage Internal Diodes

Specification Changes

No.	Steppings JA/JF/JD			Steppings JT		Page	Status	SPECIFICATION CHANGES
	A-0	A-2	C-0	A-0	A-1			
1	X[†]	X[†]				22	Doc	AC and DC Specifications Changes
2			X[†]			22	Doc	DC Specification Changes[†]

[†] Applies to JA/JF only.

Specification Clarifications

No.	Steppings JA/JF/JD			Steppings JT		Page	Status	SPECIFICATION CLARIFICATIONS
	A-0	A-2	C-0	A-0	A-1			
								None for this revision of the specification update.

Documentation Changes

No.	Document Revision	Page	Status	DOCUMENTATION CHANGES
1	272483-001	24	Doc	Page 15-09, Section 15 3.5
2	272483-001	24	Doc	Pages B-1, B-6 and B7
3	272483-002	24	Doc	Index Update for i960® Jx Microprocessor Developer’s Manual
4	272159-001 273200-001	25	Doc	Orientation of Pinout Diagrams for 196-Ball MPGBA Changed
5	272159-001	27	Doc	AC Timings Diagrams Do Not Apply to 80960JT Processor

Identification Information

Markings

80960Jx processors may be identified electrically according to device type and stepping.

Stepping Register

The following table lists the devices to which this errata sheet applies:

Device and Stepping	Identifier in g0
80960JA A-0 & A-2	0x08821013
80960JA C-0	0x30821013
80960JF A-0 & A-2	0x08820013
80960JF C-0	0x30820013
80960JD A-0 & A-2	0x08820013
80960JD C-0	0x30830013
80960JT A-0 & A-1	0x0082B013
80960JC A-1	0x30833013
80960JS A-1	0x30823013

Errata

1. RDYRCV# Restriction During Ta, Th and Ti Cycles (A-0 Stepping Only)

Problem: The RDYRCV# pin indicates that data on the address-data (AD) lines can be sampled or removed. If RDYRCV# is not asserted during a Td cycle, then the Td cycle is extended to the next cycle by inserting a wait state (Tw).

Normally the processor ignores this pin during the address state (Ta). On this stepping, however, the processor can recognize the assertion of RDYRCV# during the address state of an access and prematurely terminate the access. For example, in a quad-word load (with RDYRCV# asserted during Ta) the processor will erroneously take the first word of data off the bus during the address cycle, and will then read only three more words of data. This will cause the last word of data to be lost, and the first three will be corrupted. The processor also incorrectly responds to RDYRCV# assertion during the idle (Ti) and (Th) states. Under these conditions, the processor will deassert HOLDA (if asserted), one clock after the RDYRCV# assertion. Hold acknowledge can never be asserted (or reasserted) until one clock after RDYRCV# is sampled high.

Implication: The RDYRCV# generation logic must be designed to only assert RDYRCV# during either the Td or Tr states, and should not be used by a multi-master system for any of the other bus masters.

Workaround: Make sure that the system implementation does not allow RDYRCV# assertion during any state except valid data (Td) and recovery (Tr) states. Typical synchronous state logic for the 80960Jx can be described as “normally-not-ready” (i.e., the pin is only asserted when the system is ready). Normally-not-ready logic will usually satisfy the requirement without modification.

Strict “normally-ready” state logic cannot be used in 80960Jx systems because the logical sense of RDYRCV# changes during the recovery state. Driving RDYRCV# low indefinitely during Tr would, by definition, cause the processor to hang in Tr states indefinitely. If the ready logic resembles a normally-ready system, then change it to normally-not-ready.

If the 80960Jx is the sole bus master, then multiple open-drain ready signals can be wire-OR’ed with a pull-up resistor to drive RDYRCV#. Be sure to satisfy the processor’s setup and hold timing requirements at the end of every bus clock.

If the 80960Jx is part of a multi-master system, then do not share the RDYRCV# signal with the other bus masters. Providing separate ready signals will prevent the 80960Jx from spuriously deasserting HOLDA.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

2. RDYRCV# Restriction During Ta and Ti Cycles (A-2 Stepping Only)

Problem: The RDYRCV# signal indicates that data on the AD lines can be sampled or removed. If RDYRCV# is not asserted during a Td cycle, then the Td cycle is extended to the next cycle by inserting a wait state (Tw).

Normally the processor ignores this pin during the address state (Ta). On this stepping, however, the processor can recognize the assertion of RDYRCV# during the address state of an access and prematurely terminate the access. For example, in a quad-word load (with RDYRCV# asserted during Ta) the processor erroneously takes the first word of data off the bus during the address cycle, and then reads only three more words of data. This causes the last word of data to be lost, and the first three are corrupted.

The processor also incorrectly responds to RDYRCV# assertion during the idle (Ti) state.

Implication: The RDYRCV# generation logic must be designed to only assert RDYRCV# during either the Td or Tr states.

Typical synchronous state logic for the 80960Jx can be described as “normally-not-ready” (i.e., the pin is only asserted when the system is ready). Normally-not-ready logic will usually satisfy the requirement without modification.

Workaround: Make sure that the system implementation does not allow RDYRCV# to be asserted during any state except valid data (Td) and recovery (Tr) states. Typical synchronous state logic for the 80960Jx can be described as “normally-not-ready” (i.e., the pin is only asserted when the system is ready). Normally-not-ready logic will usually satisfy the requirement without modification.

Strict “normally-ready” state logic cannot be used in 80960Jx systems because the logical sense of RDYRCV# changes during the recovery state. Driving RDYRCV# low indefinitely during Tr would, by definition, cause the processor to hang in Tr states indefinitely. If the ready logic resembles a normally-ready system, then change it to normally-not-ready.

If the 80960Jx is the sole bus master, then multiple open-drain ready signals can be wire-OR’ed with a pull-up resistor to drive RDYRCV#. Be sure to satisfy the processor’s setup and hold timing requirements at the end of every bus clock.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

3. System-Local Fault Calls Use System-Supervisor Trace Enable Bit

Problem: When a fault handler is implemented through a system-local fault call, the fault handler IP is found in the system procedure table. All other characteristics of the call are those of a local call: PC.te (trace enable bit in the Process Controls), PC.em (execution mode flag in the Process Controls) remain unchanged and there is no stack switch. A system-local fault call on the 80960Jx has all of the previously mentioned characteristics, except that PC.te is copied from SSP.te (trace control bit in the Supervisor Stack Pointer). This erratum could cause tracing to be toggled when entering a system-local fault call. In addition when the processor is in user mode while executing the fault handler, PC.te will not be restored upon the return from the fault handler.

Implication: Little to none. Use system-supervisor calls or a local-call handler instead of a system-local fault handler.

Workaround: Do not use the system-local fault handlers. If a local fault handler is required, then use a local-call handler and place the handler pointer directly in the fault table. The other option is to use only system-supervisor calls from the system procedure table. If system-local fault handlers must be used, then the designer should make sure that the PC.te in user mode is the same as SSP.te in the System Procedure Table.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

4. Instructions “inten” and “intdis” Not Fully Implemented

Problem: Two new instructions used for interrupt control, inten and intdis, on the 80960Jx were not fully implemented.

Implication: Must use **intctl** instruction instead of either **inten** or **intdis**; however, **intctl** takes more cycles to execute.

Workaround: Do not use **inten** and **intdis** for globally enabling and disabling interrupts. Instead, use the interrupt control instruction **intctl** which can both enable and disable interrupts.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

5. Fault Stack Alignment

Problem: In some situations, when a fault occurs, an unaligned (not quad-word aligned) fault record is written to the stack; as a result, a return-from-supervisor-mode-fault incorrectly restores the arithmetic controls (AC) register and process controls (PC) register. Furthermore, while within the fault handler, the FP incorrectly points to the beginning of the fault record. This case happens when:

1. a local fault handler is selected for the fault or the current execution mode is Supervisor when the fault occurs; and
2. the current stack is offset from a quad-word boundary by 1 to 8 bytes:
   SP = 16 x N + 1
   SP = 16 x N + 2
   SP = 16 x N + 3
   SP = 16 x N + 4
   SP = 16 x N + 5
   SP = 16 x N + 6
   SP = 16 x N + 7
   SP = 16 x N + 8

Where N is a positive integer.

The other possible values of SP do not generate the erratum:

SP = 16 x N and SP = 16 x N + 9 through SP = 16 x N + 15

Code Sequences Affected

Assuming the SP described above, the erratum manifests itself any time a fault is taken and does not result in a stack switch (all cases except system supervisor fault taken from user mode). This forces the fault record to become misaligned with respect to the FP, and the stack pointer (SP) to become non-quad-word aligned. The fault type and the fault IP fields of the fault record are then stored at address FP and FP+4 respectively. The user cannot retrieve information from the fault record at the location expected in memory. Upon return from the fault handler the AC is cleared. The PC is also cleared upon return if the fault handler was executing in supervisor mode.

Implication: Requires extra code in every fault handler and extra time to implement the workaround.

Workaround: The workaround requires special code at the beginning of each fault handler that corrects the unaligned fault record and makes accessing the fault record and returning from the fault handler operate correctly. There is one limitation to the workaround (discussed in further detail below). The workaround code is guaranteed to work on all versions of the 80960Jx present and future. The workaround consists of assembly code that rewrites the entire fault record to an aligned location when the situation is detected. This code should be placed at the beginning of all fault handlers.

RESTRICTION: If the application is composed only of C code compiled by the Intel CTools or GNU C compiler, then this anomaly cannot occur. If an application is compiled by another C compiler or contains customer routines coded directly in 80960 assembly code, then it may be possible for the improper fault record to be generated when a fault occurs. For code generated by the latter method, a code restriction should be followed to ensure that the improper fault record is not generated: if the SP is always quadword aligned (i.e., if SP is always incremented and decremented by multiples of 16), then the proposed workaround is not needed.

```
## Fix Start - Place at beginning of all fault handlers.
## Uses r3 through r15
and not0xf, sp, r3
cmpobesp, r3, 1f## No fix needed if equal

## Problem detected: Fix Unaligned Fault Record
lda 24(sp),sp## Fix the stack to be quadword aligned
lda 32(fp),fp## Fix the frame pointer
lda -72(fp), r3## r3 = base for fault record
not 0, r5## r5 = 0xFFFF FFFF Wrong type of value
ldl 40(r3), r10## r10= 1st fault type (from fault record)
           ## r11= 1st fault IP (from fault record)
subo 24, sp, r12## r12= original value of SP
## Were the fault type and fault IP clobbered by a frame spill?
cmpo r11, r12## is the SP = first fault IP?
           ## <=> has an interrupt occurred?
           ##  No | Yes
sele r5, r10, r4## r4 = fault type* | 0xFFFF FFFF
sele r5, r11, r5## r5 = fault IP* | 0xFFFF FFFF
           ## * = from original fault record
## Get information from the existing fault record
ldq -24(r3),r8## Get 2nd Fault type, Fault Address,
           ## Get saved TC, fault IP (resumption record)
ldq 24(r3),r12## Get Fault Data, Otype, saved PC, AC
## Write new fault record
stl r4, 64(r3)## Store Fault Type & Fault IP
stl r8, (r3)## Write 2nd Fault Address and Fault Type
stl r10, 8(r3)## Write Record
stl r12, 48(r3)## Write Fault Data and Otype
stl r14, 56(r3)## Write Saved PC, AC
## Fix End
1:         ## Start of actual fault handler code.
```

Limitations of the Workaround

The workaround will not work if the following conditions occur:

- An interrupt occurs on the way to the fault handler.
- The local register frame of the fault handler is spilled due to the interrupt.

When this limitation occurs and the fault record is incomplete, the value 0xFFFF FFFF is written in the fault type and fault IP fields of the new fault record (created by the workaround code listed above).

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

6. Software Interrupt Erratum

Problem: When the processor is executing in supervisor mode at process priority N due to a fault or interrupt that comes from user mode at a lower process priority, a software interrupt is posted at priority S, where $S \leq N$. Then a fault or interrupt return to user mode is executed. When the PC is restored due to the return to user mode the process priority is reset to U, where $U < S$. As a result, before the first instruction in user mode can execute, control is transferred to the software interrupt handling mechanism for the interrupt posted at priority S.

The error occurs when the software interrupt handling mechanism updates the interrupt table and Software-Interrupt Priority Register *before* switching to Supervisor mode. Because the processor is in the wrong mode, a type.mismatch fault will occur and control will be transferred to the fault handling mechanism before the first instruction of the software interrupt handler can be issued. Upon return from the fault handler, execution of the interrupt handler will continue; however, subsequent software interrupts may be lost.

Implication: Overhead is needed to make sure software interrupts do not occur in user mode, or the loss of software interrupt capabilities will result.

Workaround: The only workarounds for this erratum are either not allowing software interrupts while the processor is in user mode or avoiding the use of software interrupts altogether.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

7. Pullup on LOCK#/ONCE# Pin Does Not Turn Off

Problem: The pin for the LOCK#/ONCE# signals is provided with a weak pullup device intended to keep the processor from accidentally entering ONCE mode at reset. The pullup device is supposed to turn off after the deassertion of RESET#. However, the pullup remains on, even in the ONCE high impedance mode.

The strength of the pullup device was measured at 140 mA, which means it looks approximately like a 35k resistor attached to V_{CC}.

Implication: Ensure driver strengths will overdrive the pullup transistor.

Workaround: No workaround is necessary; the erratum mostly affects factory test procedures. The LOCK# pulldown transistor and any reasonable driver attached to the pin during ONCE mode have ample strengths to overdrive the pullup transistor.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

8. Software Reinitialization Values in LMMR0, LMMR1, DLMCON

Problem: After software reinitialization, LMMR0.lmte, LMMR1.lmte, and DLMCON.dcen should be zero (refer to the *i960® Jx Microprocessor User's Manual*, Table 11-2). In the current implementation, of both the processor and the user's manual, these bits retain their values prior to the software reinitialization.

Implication: Software should take into account the fact that these values may change on a future stepping of the 80960Jx and should be able to handle both current and future values of these registers.

Workaround: None.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

9. Power Supply Current (I_{CC}) Higher than Anticipated

Problem: The I_{CC} values measured on early silicon samples are approximately 10 - 20% higher than the targeted values in the data sheet.

Implication: User must account for higher current required from the power supply.

Workaround: Intel will publish corrected I_{CC} when device is fully characterized.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

10. One Cycle Performance Hit Due to Instruction Order with LOAD Instructions

Problem: Under certain conditions, a LOAD instruction immediately followed by a REG instruction could cause an extra bus cycle to be added to the execution time. This is due to internal processor bus availability and is non-predictable.

Implication: Instructions may need to be re-ordered to avoid performance loss.

Workaround: For critical code sequences that contain LOAD's followed by a REG instruction, experiment with the order of the instructions that follow the LOAD instruction. This may eliminate the extra cycle. There is no workaround for this problem.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

11. "divi" Instruction Performance Hit When src2 = dst

Problem: Execution of the divi instruction takes approximately 40 extra cycles when the register used in the src2 operand is equal to the dst operand register.

Implication: One less register is available if performance hit is to be avoided.

Workaround: To avoid the ~40 cycle performance hit for **divi**, do not use the same register for src2 and dst. If the application is composed only of C code compiled by the Intel CTools or GNU C compiler version 4.6 or above, then this situation will not occur.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

12. Instructions Executed Between Back-to-Back Interrupts

Problem: The processor can insert instructions from interrupted process code in between back-to-back interrupts under certain conditions. This situation could cause a slightly longer interrupt latency depending on the instruction(s) executed.

This situation occurs when the processor is executing an interrupt handler, and an interrupt(s) is pending at a priority which is lower than the current interrupt handler priority and higher than the interrupted process priority. Upon return from the current interrupt handler, the processor retains the interrupt handler process priority for 4-5 cycles. This prevents lower priority interrupts from being serviced immediately and allows interrupted process code to be executed until the priority is lowered. Extra cycles will be added to the interrupt latency of the pending interrupts based on the instructions executed during the process priority interim.

Implication: Increased interrupt latency may result.

Workaround: There is no workaround for this latency problem.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

13. “balx” Instruction Does Not Branch When targ and dst Use the Same Register

Problem: When the targ and dst operands of balx use the same register, balx does not branch. Typically, the targ register holds the address to branch to, specified by the user, and the dst register is automatically loaded by the processor with the address to return to. When the same register is used for the dst and the targ, the targ register, after being used to calculate the target address, receives the processor-loaded address to return to. Architecturally, using the same registers for targ and dst is permissible and works on other 80960 family members, but on the 80960Jx this causes the balx to become non-functional.

In the following example, balx should compute a target address of “xyz”, load “abc” into r12, and branch to “xyz”. However, “abc” gets loaded into r12 and the processor branches to “abc”.

```
        lda xyz, r12

        balx(r12), r12

abc:addo ....

.

xyz:  ...
```

Implication: This erratum reduces the number of registers available for other use.

Workaround: Avoid using the same register for the targ and the dst registers of a **balx** instruction. If the application is composed only of C code compiled by the Intel CTools or GNU C compiler, then this situation will not occur.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

14. Data Breakpoints on System Procedure Entries are Lost for Certain Fault Types

Problem: This erratum occurs when tracing is enabled (PC.te = 1), and a system supervisor fault call is serviced from supervisor mode or a system local fault call is serviced from either user or supervisor mode. If there is a data breakpoint on the fault handler entry in the system procedure table, then the breakpoint trace fault will be lost.

Implication: A data breakpoint cannot break upon entering a fault handler pointed to in the system procedure table.

Workaround: Avoid setting data breakpoints on fault handler entries in the system procedure table for the above mentioned fault cases.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

15. Actual Max T_{OV} Greater than Specified in Data Sheets

Problem: The actual maximum output valid delay on PQFP and PGA packages for all outputs except ALE/ALE# inactive and DT/R# is 15nS. The original errata discussing the increase on TOV1 to 18 ns was in error and should not have been reported as an errata. The following text for errata number 15 does not apply and designs are not effected.

Problem: The actual maximum output valid delay on PQFP packages for all outputs except ALE/ALE# inactive and DT/R# is 18ns at cold and room temperatures (-6 and 25 degrees Celsius). This exceeds the specified value (15ns) in 80960Jx data sheets.

Implication: In systems requiring output valid times less than 18ns, wait states may have to be added.

Workaround: There are no workarounds.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

16. V_{IH} level on TRST#/RESET# Greater than Specified in Data Sheets

Problem: The input high level voltage (V_{IH}) on PQFP packages for inputs TRST# and RESET# is 2.6V at cold, room and hot temperatures (-6, 25 and 121 degrees Celsius). This exceeds the specified value (2.0V) in 80960Jx data sheets.

Implication: Increased drive is needed to ensure the threshold of the affected signals is exceeded.

Workaround: There are no workarounds.

Status: Fixed. Refer to Summary Table of Changes to determine the affected stepping(s).

17. Power Supply Sequence Can Damage Internal Diodes

Problem: If the voltage on the VCCPLL or VCC(I/O) power supply pins exceeds the V_{CC} pin voltage by 0.5 V at any time, including during the power up and power down sequences, excessive currents can permanently damage on-chip electrostatic discharge (ESD) protection diodes. The damage can accumulate over multiple episodes.

The VCC(I/O) pins appear only on the 132-lead PQFP package.

Pragmatically, this problem only occurs when the VCCPLL, VCC(I/O), and V_{CC} pins are driven by separate power supplies or voltage regulators. Applications that use one power supply for VCCPLL, VCC(I/O), and V_{CC} are not typically at risk. Verify that your application does not allow the VCCPLL voltage to exceed V_{CC} by 0.5 V.

ESD diodes connect the VCCPLL and VCC(I/O) circuitry to V_{CC}. Normally, those diodes are unbiased or reverse-biased, so no current flows. In the event of a positive electrostatic pulse on VCCPLL or VCC(I/O), the diodes protect the phase-locked loop and input/output circuitry by shunting the excess charge to V_{CC}. However, when power supplies forward-bias these diodes for any length of time, the current flow can damage or destroy the diodes.

The VCCPLL low-pass filter recommended in the datasheet does not promote this problem.

The VCC5 power supply pin is also susceptible to excessive current damage, but is adequately protected by the 100 Ω series resistor recommended in the datasheet. See the *80960JA/JF/JD/JT 3.3 V Embedded 32-Bit Microprocessor* datasheet for more details.

Implication: Diode damage can manifest itself as any of the following:

- Resistive short circuits between the VCCPLL, VCC(I/O), and V_{CC} pins
- Compromised ESD protection on the VCCPLL and VCC(I/O) pins

- Unspecified functional or parametric failures resulting from damage to the circuitry near the diodes

Workaround: Use one common power supply or regulator for the VCCPLL, VCC(I/O), and V_{CC} pins. Otherwise, ensure that the V_{CC} pins power-up before VCCPLL and VCC(I/O) and power-down after VCCPLL and VCC(I/O). An alternative for VCCPLL is to limit the VCCPLL diode current flow by providing at least 100 Ω of resistance in series with the VCCPLL pin. Series resistance is not recommended for VCC(I/O).

Status: NoFix. Refer to Summary Table of Changes to determine the affected stepping(s).

Specification Changes

1. AC and DC Specifications Changes

Problem: In the *80960JA/JF Embedded 32-Bit Microprocessor Datasheet* (272504-004), several AC timing specifications and two I_{CC} values have changed.

Table #	Spec	Device	Old Value	New Value
15	TC Max	Jx-33	62.5 ns	125 ns
17		Jx-25	62.5 ns	125 ns
18		Jx-16	62.5 ns	125 ns
14	Halt Mode ICC	Jx-25	30 mA	33 mA
		Jx-16	20 mA	26 mA
17	TOV1	Jx-25	17 ns	15 ns
18	TOV1	Jx-16	18 ns	15 ns
	TBSOF1	Jx-16	36 ns	30 ns
	TBSOF2	Jx-16	36 ns	30 ns

2. DC Specification Changes

Problem: In the *80960JA/JF/JD/JT Embedded 32-Bit Microprocessor Datasheet* (273159-001), two I_{CC} values have changed.

Table #	Spec	Device	Old Value	New Value
22	I_{CC} Active Power Supply	JA/JF-25	260 mA	241 mA
22	I_{CC} Active Power Supply	JA/JF-16	194 mA	154 mA

Specification Clarifications

None for this revision of the Specification Update.

Documentation Changes

1. Page 15-09, Section 15 3.5

Problem: Sentence six of the first paragraph of section 17.3.5 incorrectly states:

"The TAP controller is automatically initialized on power-up."

A new paragraph has been added between the first and second paragraphs:

"The TAP controller is not automatically initialized on power-up. Therefore, it is important that the system reset the TAP controller after power up by asserting the TRST# pin. In addition, the TAP controller can be initialized by applying a high signal level on the TMS input for five TCK periods. Systems that do not use JTAG, or that normally do not apply a clock to TCK should provide a pull-down resistor on TRST# to hold the TAP controller in the Test_Logic_Reset state. A 2.7k value is strong enough to overcome the TRST# pins internal pull-up, but weak enough to allow automatic test equipment to overdrive it during production testing.

"Alternatively, the TRST# pin may be connected to ground if the Test Access Port is never used."

Affected Docs: *i960® Jx Microprocessor User's Manual*, Order #272483

2. Pages B-1, B-6 and B7

Problem: Tables B-1, B-3 and B-4 show a "T" column for a branch prediction bit. Since branch prediction is not implemented on the i960 Jx processor, this bit is ignored.

Affected Docs: *i960® Jx Microprocessor User's Manual*, Order #272483

3. Index Update for *i960® Jx Microprocessor Developer's Manual*

Problem: Page numbering incorrect for numerous references. Corrected Index appended.

Affected Docs: *i960® Jx Microprocessor User's Manual*, Order #272483

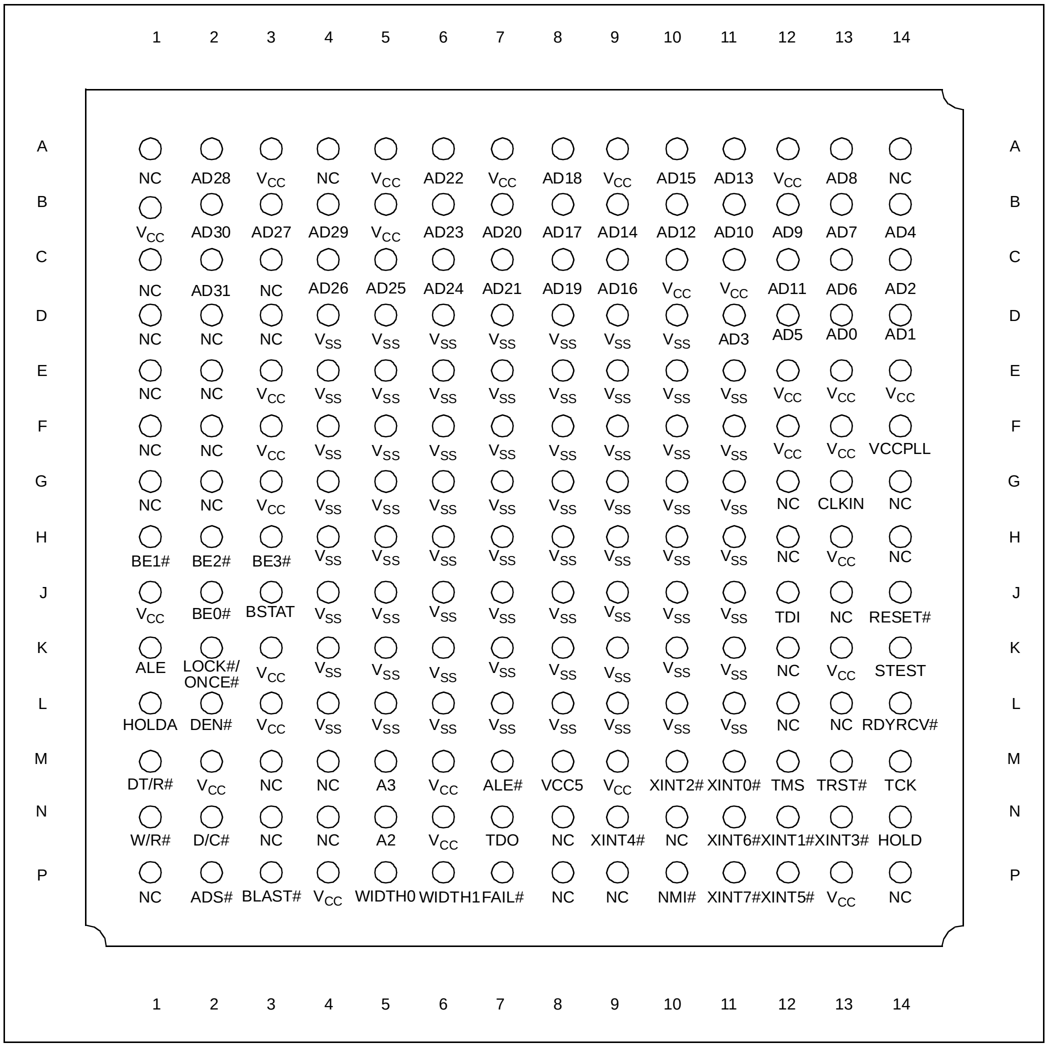
Note: The corrected *i960® Jx Microprocessor User's Manual* can be obtained from Intel's Developer web site at *http://developer.intel.com/design/i960/manuals/272483.htm*.

4. Orientation of Pinout Diagrams for 196-Ball MPGBA Changed

Problem: The two pinout diagrams for the 196-Ball MPBGA package (Figures 6 and 7 in both documents listed below) incorrectly show the orientation of the package. Also, the figure titles incorrectly indicate which is the the top view and which is the bottom view and contain the wrong indication of “Balls Facing Up” and “Balls Facing Down.”

Figure 6 is the top view, “Balls Facing Down,” and should be oriented as shown:

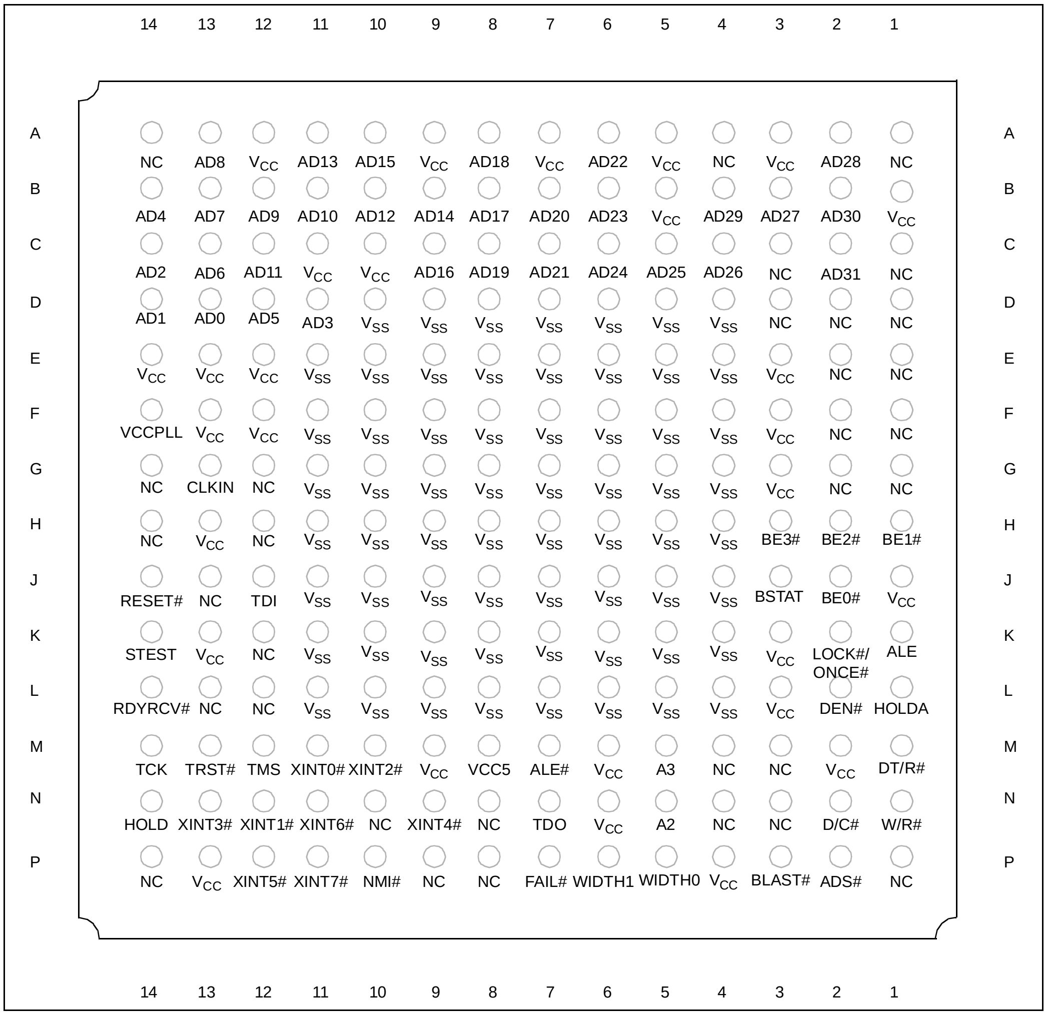
Figure 6. 196-Ball Mini Plastic Ball Grid Array Top View - Balls Facing Down



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	NC	AD28	V_{CC}	NC	V_{CC}	AD22	V_{CC}	AD18	V_{CC}	AD15	AD13	V_{CC}	AD8	NC
B	V_{CC}	AD30	AD27	AD29	V_{CC}	AD23	AD20	AD17	AD14	AD12	AD10	AD9	AD7	AD4
C	NC	AD31	NC	AD26	AD25	AD24	AD21	AD19	AD16	V_{CC}	V_{CC}	AD11	AD6	AD2
D	NC	NC	NC	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	AD3	AD5	AD0	AD1
E	NC	NC	V_{CC}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{CC}	V_{CC}	V_{CC}
F	NC	NC	V_{CC}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{CC}	V_{CC}	VCCPLL
G	NC	NC	V_{CC}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	NC	CLKIN	NC
H	BE1#	BE2#	BE3#	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	NC	V_{CC}	NC
J	V_{CC}	BE0#	BSTAT	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	TDI	NC	RESET#
K	ALE	LOCK#/ ONCE#	V_{CC}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	NC	V_{CC}	STEST
L	HOLDA	DEN#	V_{CC}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	NC	NC	RDYRCV#
M	DT/R#	V_{CC}	NC	NC	A3	V_{CC}	ALE#	VCC5	V_{CC}	XINT2#	XINT0#	TMS	TRST#	TCK
N	W/R#	D/C#	NC	NC	A2	V_{CC}	TDO	NC	XINT4#	NC	XINT6#	XINT1#	XINT3#	HOLD
P	NC	ADS#	BLAST#	V_{CC}	WIDTH0	WIDTH1	FAIL#	NC	NC	NMI#	XINT7#	XINT5#	V_{CC}	NC

Figure 7 is the bottom view, “Balls Facing Up,” and should be oriented as shown:

Figure 7. 196-Ball Mini Plastic Ball Grid Array Bottom View - Balls Facing Up



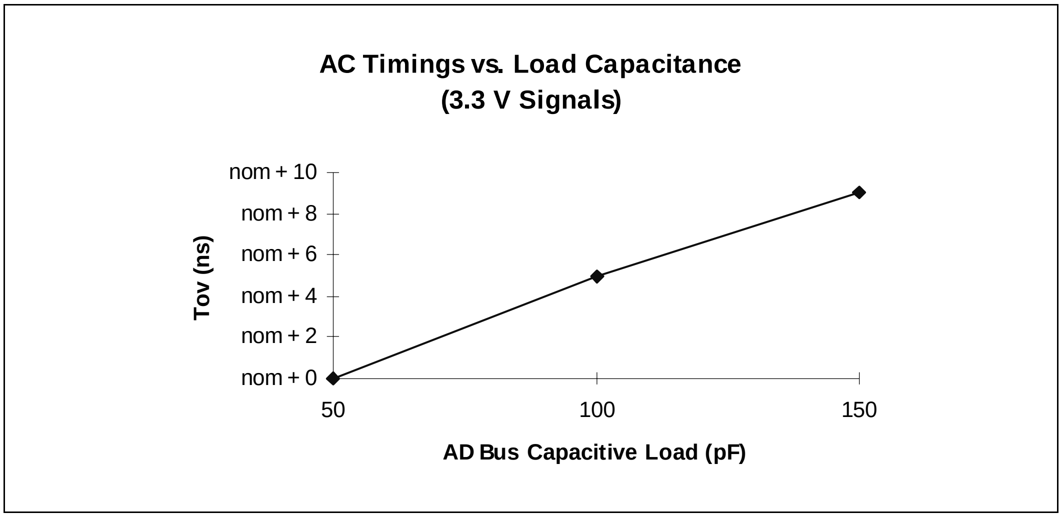
	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
A	NC	AD8	V_{CC}	AD13	AD15	V_{CC}	AD18	V_{CC}	AD22	V_{CC}	NC	V_{CC}	AD28	NC	A
B	AD4	AD7	AD9	AD10	AD12	AD14	AD17	AD20	AD23	V_{CC}	AD29	AD27	AD30	V_{CC}	B
C	AD2	AD6	AD11	V_{CC}	V_{CC}	AD16	AD19	AD21	AD24	AD25	AD26	NC	AD31	NC	C
D	AD1	AD0	AD5	AD3	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	NC	NC	NC	D
E	V_{CC}	V_{CC}	V_{CC}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{CC}	NC	NC	E
F	VCCPLL	V_{CC}	V_{CC}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{CC}	NC	NC	F
G	NC	CLKIN	NC	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{CC}	NC	NC	G
H	NC	V_{CC}	NC	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	BE3#	BE2#	BE1#	H
J	RESET#	NC	TDI	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	BSTAT	BE0#	V_{CC}	J
K	STEST	V_{CC}	NC	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{CC}	LOCK#/ ONCE#	ALE	K
L	RDYRCV#	NC	NC	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{CC}	DEN#	HOLDA	L
M	TCK	TRST#	TMS	XINT0#	XINT2#	V_{CC}	VCC5	ALE#	V_{CC}	A3	NC	NC	V_{CC}	DT/R#	M
N	HOLD	XINT3#	XINT1#	XINT6#	NC	XINT4#	NC	TDO	V_{CC}	A2	NC	NC	D/C#	W/R#	N
P	NC	V_{CC}	XINT5#	XINT7#	NMI#	NC	NC	FAIL#	WIDTH1	WIDTH0	V_{CC}	BLAST#	ADS#	NC	P
	14	13	12	11	10	9	8	7	6	5	4	3	2	1	

Affected Docs: *80960JA/JF/JD/JT 3.3 V Microprocessor* datasheet, Order #273159-001, and *80960JS/JC 3.3 V Microprocessor* datasheet, Order #273200-001.

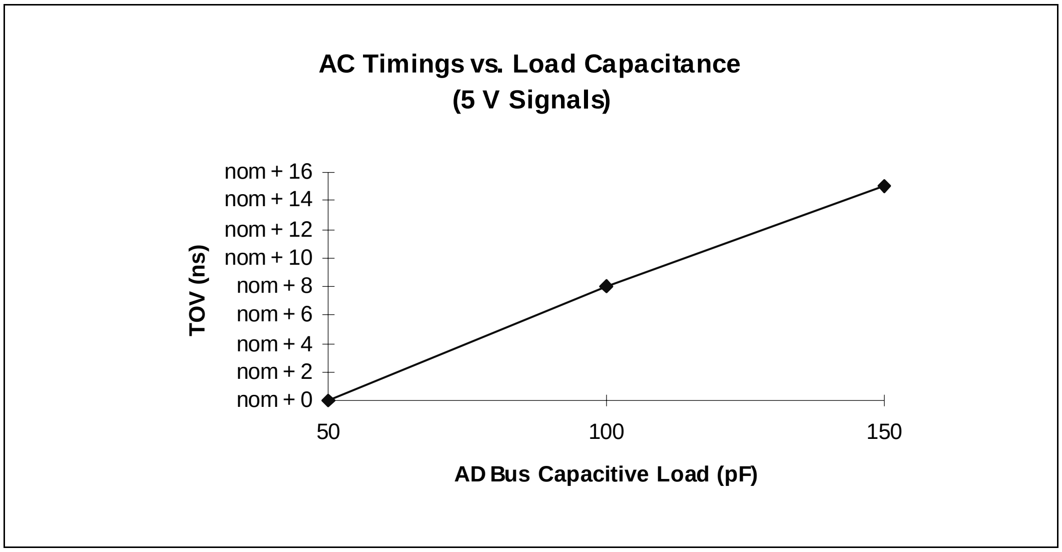
5. AC Timings Diagrams Do Not Apply to 80960JT Processor

Problem: *80960JA/JF/JD/JT 3.3 V Microprocessor* datasheet, the “Output Delay vs. Load Capacitance” diagram and the “TLX vs. AD Bus Load Capacitance” diagrams do not apply to the 80960JT microprocessor (they are correct, however, for the JA/JF/JD microprocessors). The following diagrams and related notes provide the correct derating curve information for the 80960JT microprocessor:

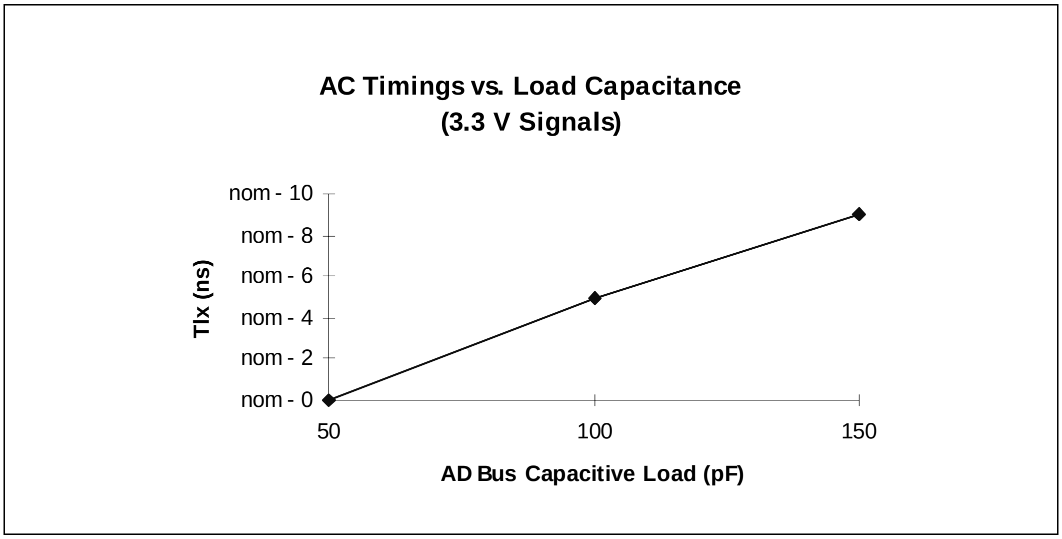
Output Delay or Hold vs. Load Capacitance (3.3 V Signals)



Output Delay or Hold vs. Load Capacitance (5 V Signals)

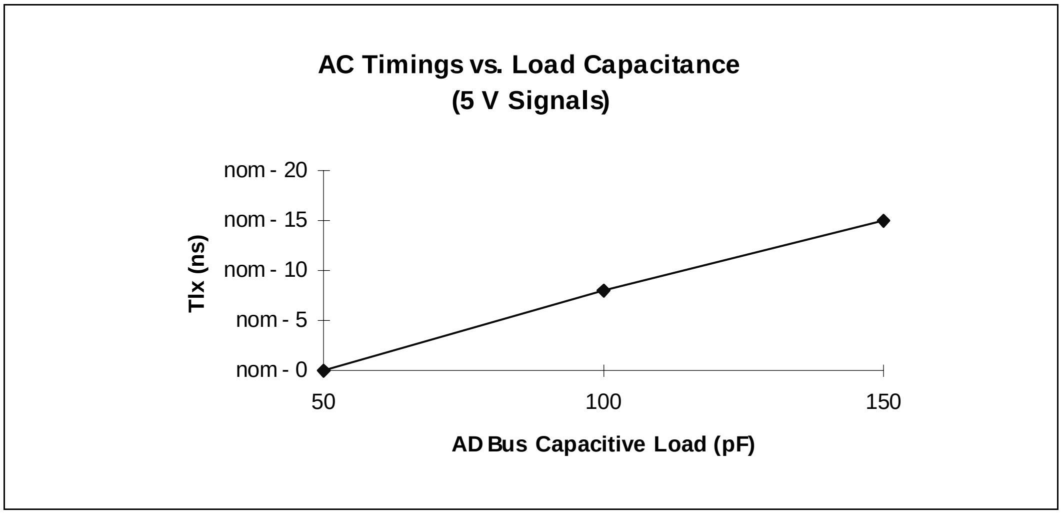


T_{LX} vs. AD Bus Load Capacitance (3.3 V Signals)



Note: The T_{LX} Derating curve applies only when an imbalance in the capacitive load occurs between the AD bus and ALE. The T_{LX} derating is based on a 50 pF load on ALE. The derating applies to ALE and ALE#.

T_{LX} vs. AD Bus Load Capacitance (5 V Signals)



Note: The T_{LX} Derating curve applies only when an imbalance in the capacitive load occurs between the AD bus and ALE. The T_{LX} derating is based on a 50 pF load on ALE. The derating applies to ALE and ALE#.

Affected Docs: *80960JA/JF/JD/JT 3.3 V Microprocessor* datasheet, Order #273159-001