MIPS R4000PC/SC Errata, Processor Revision 2.2 and 3.0

May 10, 1994

MIPS Technologies Inc.
2011 N Shoreline Blvd
PO Box 7311
Mountain View, CA 94039-7311

This document contains information that is proprietary to MIPS Technologies, Inc.

MIPS Technologies, Inc. reserves the right to change any products described herein to improve the function or design. MIPS Technolgies, Inc. does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under patent rights nor imply the rights of others.

Copyright 1993 by MIPS Technologies, Inc. All rights reserved. No part of this document may copied by any means without written permission from MIPS Technologies, Inc.

Note: Change bars in the left column indicate corrections or changes from the last revision of the errata.

1. R4000PC, R4000SC: Master/checker mode is not available.

Workaround: The mode bits to enable Master/Checker operation should not be set to one. These mode bits are the MCMode and DataMaster modebits. Behavior of the R4000 is undefined when Master/Checker mode is enabled.

2. R4000SC: Status output pins do not function.

3. R4000PC, R4000SC: Reduced power mode is not available in the current revision of the R4000. Setting the RP bit in the Status register has no effect on the operation of the R4000.

4. R4000PC, R4000SC (Note: Processor revision 3.0 does not contain this errata): An instruction sequence which contains a load which causes a data cache miss and a jump, where the jump instruction is that last instruction in the page and the delay slot of the jump is not currently mapped, causes the exception vector to be overwritten by the jump address. The R4000 will use the jump address as the exception vector.

Example:

lw	<----	data cache miss
noop	<----	one or two Noops
jr	<----	last instruction in the page (jump or branch instruction)
--------------	<----	page boundary
noop		

Workaround: Workaround: Jump and branch instructions should never be in the last location of a page.

5. R4000SC: When the primary instruction cache is configured with an 8-word line size, the virtual coherency exception (VCE) does not function correctly.

Workaround: Workaround: The primary instruction cache should only be configured with a 4-word line size.

6. R4000PC, R4000SC: The following conditions cause the R4000 to operate incorrectly:

1) An exception is taken from user code
2) On the eret instruction of the exception handler, a CacheErr exception is taken.

The R4000 takes the CacheErr exception correctly but returns to user code instead of the ERET in the exception handler. This will then cause an Address Error exception.

Workaround: Use the following code sequence as the last three instructions of the exception handler:

```
eret
noop
eret
```

The CacheErr handler must add 4 to the ErrorEPC to return to the noop.

7. R4000PC (Note: Processor revision 3.0 does not contain this errata): When an external request is placed between the read and write of a data cache replacement of a dirty cache line and an uncached store is stalled in the WB pipeline stage, the R4000 will send out the command code for an uncached store during the block write of the writeback.

Workaround: Workaround: Interrupts should be signaled through the Int0 through Int5 pins on the R4000PC package. An external null request should not be signaled between the write and read of the replacement of a dirty cache line. In the Revision 2.2 R4000, there are four cycles between the write and read of a dirty cache line replacement.

8. R4000PC, R4000SC: In supervisor mode, with the SX bit (64-bit mode enabled for supervisor mode) in the Status register set to one, the R4000 will generate an Address Error Exception in the "csseg" region: 0xFFFF FFFF C000 0000 to 0xFFFF FFFF DFFF FFFF.

9. R4000PC: If a writeback is delayed by an external request and the processor executes a cache operation which generates a writeback with the retained bit set on the system command bus, the retained bit will also be set for the first writeback.

Workaround: Workaround: Since the retained bit is intended for multiprocessing systems, R4000PC systems should ignore this bit on the system command bus.

10. R4000PC, R4000SC: In kernel mode with the KX bit (64-bit mode enabled for kernel mode) in the Status register set to one, the R4000 fails to generate an Address Error Exception if a load or store is attempted in the region: 0xC000 0FFF F000 0000 to 0xFFFF FFFF 8000. Attempting access to this region should cause an Address Error exception.

11. R4000PC, R4000SC: In the case:

```
lw      rA, (rn)
noop                    (or any non-conflicting instruction)
lw      rn, (rA)        (where the address in rA causes a TLB refill)
-------------------->   end of page
page not mapped
```

where rn and RA are general purpose registers r0 through r31

This code sequence causes the second load instruction to slip due to a load use interlock. When the R4000 crosses the page boundary after the lw, it vectors to 0x8000 0000 and later causes an instruction cache miss. After the instruction cache miss is complete the LW causes another TLB refill. This should vector to 0x8000 0000 but instead goes to 0x8000 0180.

Workaround: Workaround: In the general exception handler, the CAUSE register must be checked to see if a TLB miss is indicated.

12. R4000SC: The following conditions cause the virtual address in the BadVA register to be corrupted by a TLB Probe instruction (TLBP)

The problem occurs if:

1) An exception is generated
2) A TLBProbe is required to handle the exception
3) While handling the exception, a VCED or VCEI occurs, the BadVA may be incorrect.

Workaround: Workaround: The exception handler should jump to uncached space to handle the TLBProbe.

13. R4000SC: The cacheop, Create_DEX_SC (secondary cache operation), when executed on a non-coherent line, will change the state of the secondary line to Clean Exclusive instead of Dirty Exclusive.

Workaround: Workaround: This situation should not create any problem in normal operation since a subsequent store will change the line to Dirty Exclusive.

14. This errata is an update to errata 4. The incorrect behavior does not occur with two Noops between the load and jump. There is a qualifying condition on the load and jump instructions and several similar cases are listed which cause the same error.

R4000PC, R4000SC: An instruction sequence which contains a load which causes a data cache miss and a jump, where the jump instruction is that last instruction in the page and the delay slot of the jump is not currently mapped, causes the exception vector to be overwritten by the jump address. The R4000 will use the jump address as the exception vector. In the first case, the target of the load instruction and source register for the jump instruction must be the same register**.** In all other cases, the condition is independent of the registers used.

Example:

lw	rA,(rn)	<----data cache miss
noop		<---- **one** Noop
jr	rA	<---- jump or branch instruction as the last instruction in the page
----------------------		<---- page boundary
PAGE NOT MAPPED		
lw		<----data cache miss
div		<---- signed, unsigned and doubleword integer divide
beq		<---- branch instruction as the last instruction in the page
----------------------		<---- page boundary
PAGE NOT MAPPED		
sw		<----data cache miss
div		<---- signed, unsigned and doubleword integer

	divide
beq	<---- branch instruction as the last instruction in the page
----------------------	<---- page boundary
PAGE NOT MAPPED	
cacheop	<----data cache miss
div	<---- signed, unsigned and doubleword integer divide
beq	<---- branch instruction as the last instruction in the page
----------------------	<---- page boundary
PAGE NOT MAPPED	

Workaround: Jump and branch instructions should never be in the last location of a page.

15. R4000PC, R4000SC: This errata was deleted. The problem does not occur on the R4000.

16. R4000PC, R4000SC: Please refer to errata 28 for an update to this errata description.

The following code sequence causes the R4000 to incorrectly execute the Double Shift Right Arithmetic 32 (dsra32) instruction. If the dsra32 instruction is executed during an integer multiply, the dsra32 will only shift by the amount in specified in the instruction rather than the amount plus 32 bits.

instruction 1:	mult rs,rt	integer multiply
instruction 2-12:	dsra32 rd,rt,rs	doubleword shift right arithmetic + 32

Workaround: Workaround: A dsra32 instruction placed after an integer multiply should not be one of the 11 instructions after the multiply instruction.

17. R4000SC: This problem occurs when the system interface of the R4000SC is configured with ECC checking. During the writeback of a dirty cache line, the first double word from the secondary cache which follows a double word from the primary cache is driven with an incorrect ECC code on the SysADC bus. The SysADC bus retains the same code for the secondary cache double word as was driven for the primary cache double word.

Workaround: Use parity checking on the system interface bus (SysAD) rather than ECC. Alternatively, mask all checking errors by setting the DE bit of the Diagnostic Status Field in the Status register to one.

18. R4000SC: Under some conditions, stores which address the same primary cache line (lower 12 bits are the same), will prevent the R4000 from responding to an external request until all the secondary cache accesses are complete. The conditions under which this occur are the following:

1) Back to back stores which address the same cache line where the tags match.
2) Back to back stores which address a different address but map to the same cache line

(lower 12 bits of the address are identical). In this case, the stores will result in the replacement of the line. If the accesses to the secondary cache map to the same location, the problem does not occur because the secondary cache interface is idle during the writeback. In this case, the external request will be accepted during the writeback.

3) Successive stores, not necessarily immediately following one another, which map to the same cache line under the conditions listed in 1 and 2 above but where the length of the loop is short enough that the secondary cache bus is never idle. The number of instructions in the loop where the problem can be observed is dependent upon the secondary cache timing parameters programmed by the boot time mode bits.

19. R4000PC, R4000SC: When there is a store followed by a load to the same address and the Watch register contains the address for the load, the Watch exception for the load is not taken.

Workaround: A "noop" placed between the store and the load will enble the Watch exception to be taken correctly.

22 R4000PC, R4000SC: When returning from 32-bit kernel mode to 64-bit user mode, the R4000 interprets the address of the first instruction as a 32-bit mode address. This may cause an incorrect mapping of the address depending upon the virtual address.

Workaround: When operating the R4000 in 64-bit user mode, only use 64-bit kernel mode. (KX=1 in the CP Status register)

23. R4000PC, R4000SC: The 64-bit instruction, daddi, fails to take an overflow exception.

Workaround: There is no workaround for this problem.

24. R4000PC, R4000SC: The R4000 does not take a Reserved Instruction Exception on the "rfe' instruction. The R4000 executes a "noop" and kills the following instruction.

Workaround: There is no workaround for this problem.

25. R4000PC, R4000SC: The "dmtc0" and "dmfc0" instructions do not cause a Reserved Instruction Exception in User or Supervisor mode. These instructions will complete successfully.

Workaround: Workaround: There is no workaround for this problem.

26. R4000SC: This errata text was deleted. Please see errata # 29 for the correct text.

27. When a TLB refill exception occurs on an instruction fetch, the value in the CP0 register BadVAddr might not match CP0 register EPC (or EPC+4 in case of a branch or jump with the delay slot as the first instruction of the next page.

Workaround:

In the first level TLB refill exception, use the TLB probe instruction to check if the virtual address which is in the BadVAddr register already exist in the TLB. If it is in the TLB, then eret (as the BadVAddr was incorrect), else go ahead and write the new TLB entry and eret. By overlapping the TLB probe operation with the other instructions in the handler, and then placing the TLB write instruction in the branch delay slot of a branch likely instruction, the performance overhead for this workaround can be minimized.

Example:

```
        mfc0    k0,   context
        lw      k1,   0(k0)
        lw      k0,   4(k0)
        c0      tlbp                <-- additional instruction due to workaround
        srl     k1,   k1, 3
        mtc0    k1,   tlblo0
        srl     k0,   k0, 3
        mfc0    k1,   index         <-- additional instruction due to workaround
        mtc0    k0,   tlblo1
        bltzl   k1,   1f            <-- additional instruction due to workaround
        c0      tlbwr
1:      nop
        c0      eret
```

If the processor takes a TLB refill exception from the first level exception then it will jump to the “general exception handler”. Inside the “general exception handler”, when the operating system (OS) detects an address outside the expected range in BadVAddr, it should check EPC to make sure it is within a valid range for the process. If EPC is within the valid range, the OS should execute an “eret” instruction. The refill instruction will be re-taken and BadVAddr will contain the correct value.

If the OS is unable to determine the valid address range for the process, the value in EPC should be used to look for a load or store instruction. If EPC does not point to a load or store, the OS should execute an “eret”. The “eret” will then cause another TLB refill exception, which will have a valid BadVAddr. If EPC points to a load or store, the OS must then interpret the instruction to generate the address for the data. If this address matches the address in BadVAddr, the process tried to access data outside the process address space. Otherwise the OS should execute an “eret” causing a TLB refill exception where the value in BadVAddr will be valid.

28. R4000PC, R4000SC: The text from errata 16 should be replaced by the following description.

All extended shifts (shift by n+32) and variable shifts (32 and 64-bit versions) may produce incorrect results under the following conditions:

1) An integer multiply is currently executing
2) These types of shift instructions are executed immediately following an integer divide instruction.

Workaround:

1) Make sure no integer multiply is running wihen these instruction are executed. If this cannot be predicted at compile time, then insert a "mfhi" to R0 instruction immediately

after the integer multiply instruction. This will cause the integer multiply to complete before the shift is executed.

2) Separate integer divide and these two classes of shift instructions by another instruction or a noop.

29. R4000SC: Errata 26 is incorrect. The R4000 always uses sequential ordering regardless of the state of the mode bit which specifies subblock or sequential ordering.

30. R4000PC, R4000SC: When an nmi or softreset exception is taken, both exl and erl bits are set to one in the Status register. The R4000 should perserve the previous state of exl.

31. R4000SC: The SCAPar pins on the secondary cache bus are driven incorrectly. The pins are driven with even parity for the secondary cache address, SCWrB, SCDCSB and SCTCSB pins.

Workaround: These bits are not stored in the tag and are only used in systems which externally compare the address and parity bits.

32. R4000PC, R4000SC: Under the following conditions, the CP0 register, BadVAddr, can be corrupted.

1) There is a data cache miss which causes a jal (jump and link) to be stalled in the DS pipeline stage
2) A floating point dependency causes a slip during the restart after the data for the data cache miss is returned.

Workaround: Workaround:

33. R4000PC, R4000SC: With the following code sequence and conditions, the R4000 with use the general exception vector, 0x80000180, instead of the the TLB Exception vector, 0x800000000.

lw	- takes a watch exception
lw	- takes a watch exception
j	- TLB exception
--------------------->	Page boundary

Workaround: The general exception handler needs to handle refill exceptions directly from user code rather than only within the refill exception handler.

34. R4000PC, R4000SC: The R4000 incorrectly allows xkseg to access up to 0xc000 0100 0000 0000. This region should only extend to 0xc000 00ff 8000 0000.

Workaround:

35. R4000PC, R4000SC: Split secondary cache mode can cause invalid cache error exceptions.

Workaround: Use unified secondary cache mode.

36. R4000PC, R4000SC: The first instruction of the ECC handler cannot write to a general purpose register.

Workaround: The first instruction of the ECC exception handler should be a Noop.

37. R4000PC, R4000SC: The PIDx field of the CP0 register CacheErr, gets the wrong value on Data and Instruction parity errors. ECC errors are will put the correct values in this register, however.

Workaround: Under this failure condition, all possible values of PIDx in the primary cache must be checked for parity errors. In the R4000 with 8K primary caches, there are only two values for the primary cache index that must be checked.

38. R4000PC, R4000SC: A parity error on the primary cache dirty data bit, W, will not be detected.

Workaround: Workaround: There is no workaround for this condition.

39. Under the following conditions, the TLB attributes for a page being refill into the microTLB may be associated with a page mapped in the microTLB.

1) The address of a jump instruction is to the last instruction in a page.
2) The next instruction after the jump target is not mapped in the microTLB.
3) The jump is stalled in the DS pipeline stage.

When the page targeted by the jump is refilled into the microTLB, the coherency bits associated with that page may be incorrect.

Workaround: This problem occurs if the TLB attributes are different for the two pages. Under these conditions, if the TLB attributes are the same, this problem will not occur.

40. R4000 PC, R4000 SC: Processors might not function in "lock-step" properly because the timer in CP0 (Count Register) may not synchronize across multiple processors at reset. As a result, the timers may increment on different clock edges causing the processors to fall out of lock step.

Workaround: Do not use the Count Register as a timer if more than one processor needs to function in lock-step with each other.

41. R4000PC, R4000SC: Under the following condition, the DADDIU instruction can produce an incorrect result.

If this instruction generates a result value that would cause an overflow condition to occur (even though this instruction does not take an overflow exception) then the result value will be correct in bits 0-31 but bit 31 will be replicated through bits 32-63 (so it looks like a 32bit sign-extended value). The overflow condition is defined when the carries out of bits 62 and 63 differ (two's compliment overflow).

Workaround: There is no workaround for this problem.

42. R4000 PC, R4000 SC: The 64bit address space for kernel mode is not correctly decoded. Addresses that should be reported as address error exceptions may cause tlb refill exceptions instead (as they will not be correctly decoded as outside the range of kernel mode address space). The adddress region effected by this is the xkseg region as defined in the MIPS III architecture (address range `0xc000000000000000 - 0xc00000ff7fffffff`). This region is extended to the range `0xc000000000000000 - 0xc00000ffffffffff`.

Workaround: Workaround: There is no workaround for this problem - kernel mode may access this expanded region.

43. R4000 PC, R4000 SC: The 64bit address space for supervisor mode is not correctly decoded. Addresses that should be reported as address error exceptions, may cause tlb refill exceptions instead (as they will not be correctly decoded as outside the range of supervisor mode address space). The address region effected by this is the sseg region as defined in the MIPS III architecture (address range `0xffffffffc0000000 - 0xffffffffdfffffff`). The check to make sure bits 61-32 are all ones is not done. This means the R4000 will permit access to the address regions shown below in supervisor mode:

0xc0000000c0000000 - 0xc0000000dfffffff

0xc0000001c0000000 - 0xc0000001dfffffff

.

.

```
0xfffffffdc0000000 - 0xfffffffddfffffff
0xfffffffec0000000 - 0xfffffffedfffffff
```

Workaround: There is no workaround for this problem - supervisor mode may be able to access kernel address space. This problem is only present in 64bit supervisor mode.

44. R4000SC: The TWr2Dly parameter is always 1 more than the number programmed. Thus, the range of TWr2Dly is 2 to 4 PCycles instead of 1 to 3 PCycles.

Workaround: There is no work around for this problem except taking the bug into account while programming.

45. R4000PC, R4000SC: The processor will not take Cache Error exception due to PTag Parity Error under the following condition:

When performing a Hit_Writeback_Inv_D operation on a primary cacheline, with W_Bit=0. It will invalidate the line, ignoring the error. However, it will take the exception, if the cacheline is inconsistent (W_Bit=1).

Workaround: The effect of this bug is very minimal because, in the worst case, the incorrectly invalidated cacheline will be refetched, when the data is accessed again.

46. R4000PC, R4000SC: NMI does not clear the TLB Shutdown (TS) Bit in the Status Register

(SR[21]).

Workaround: There is no workaround for this problem.

47. When a cache-error exception occurs, the ERL bit is set in the status register to indicate the chip is now executing at error level. However, due to this bug the EXL bit in the status register may also be set. This means that if the processor takes cache-error exception from the "non-exception level", it may return, incorrectly, from the cache-error exception to "exception level (EXL set)". If a cache-error exception occured when the processor was at the "exception level (EXL set)" then it will return, correctly, from the cache-error exception to the previous "exception level".

Workaround: The cache-error exception handler needs to determine if the chip was executing at "exception level" or not, when the cache error exception was taken, and then set the EXL bit in the status register appropriately before executing the ERET.

48. R4000PC, R4000SC: If an NMI is accepted by the processor when it is in any type of stall, the address of the instruction in the WB stage is transferred into EPC and the instruction is killed. However, if the stalled instruction involves a write into the register file, the processor is unable to stop the write. Upon returning from NMI exception handler, the processor re-executes the killed instruction; thus, causing the write into the same register again. The extra write into the register might cause an erroneous result, if the instruction uses same register as the source and the destination.

Workaround: There is no work around for this problem.

49. R4000PC, R4000SC: When data is moved from the Random register to a general purpose register (using MFC0), the result in the destination register, erroneously, ends up to be one less (for example, if the Random register has a value N, the destination register ends up with a value N-1). However, the TLBWR instruction which uses the value from Random register to index into the TLB-array, gets the value N and thus writes into the correct TLB-entry at all times.

Workaround: Adjust the destination data of the instruction MFC0, when the source is Random register.

50. R4000PC, R4000SC: The processor, incorrectly, sets the EXL bit in the status register when an exception is caused by the NMI.

Workaround: There is no work around for this problem.

51. R4000PC, R4000SC: The J and JAL instructions functions incorrectly in certain cases, as described below:

J or JAL instruction causes the processor to unconditionally jump to an address which is formed by shifting left, by two bits, the 26 bit target address provided by the instruction; and by concatenating, at the left end, the high order bits of the address of the delay slot.

The concatenation of the high order bits of the address of the delay slot means that these instructions can only cause a jump within the 256Mbyte region where the delay slot instruction is located.

The bug occurs when a J, or JAL instruction falls in any of the 3 words before the last word in a 256Mbyte region (the words marked j below):

```
0x0ffffff0    j
0x0ffffff4    j
0x0ffffff8    j
0x0ffffffc
------------------------------------- 256Mbyte region boundary
0x10000000
or:
0x1ffffff0    j
0x1ffffff4    j
0x1ffffff8    j
0x1ffffffc
------------------------------------- 256Mbyte region boundary
0x20000000
..., etc.
```

In these cases, the high order bits are taken from the next 256Mbyte region generating a destination address that is 256Mbytes beyond the address that was intended.

Workaround: When assigning addresses, the case of a jump on the boundary of a 256Mbyte region already has to be made a special case as it will form a destination address in the next 256Mbyte region (due to the delay slot already being in the next 256Mbyte region). To workaround this bug, the same address check has to be performed for jumps(j or jal) within the last 4 words of the 256Mbyte region, instead of just the last word.

52. R4000SC: This bug does not apply for the R4000PC.

There are two flavors of this bug:

1) If the instruction just after divide takes an RF exception (tlb-refill, tlb-invalid) and gets an instruction cache miss (both primary and secondary) and the line which is currently in secondary cache at this index had the first data word, where the bits 5..2 are set, then R4000 would get a wrong result for the div.

```
##1
        nop
        div         r8, r9
        -------------------                 # end-of page. -tlb-refill
        nop
##2
        nop
        div         r8, r9
        -------------------                 # end-of page. -tlb-invalid
         nop
```

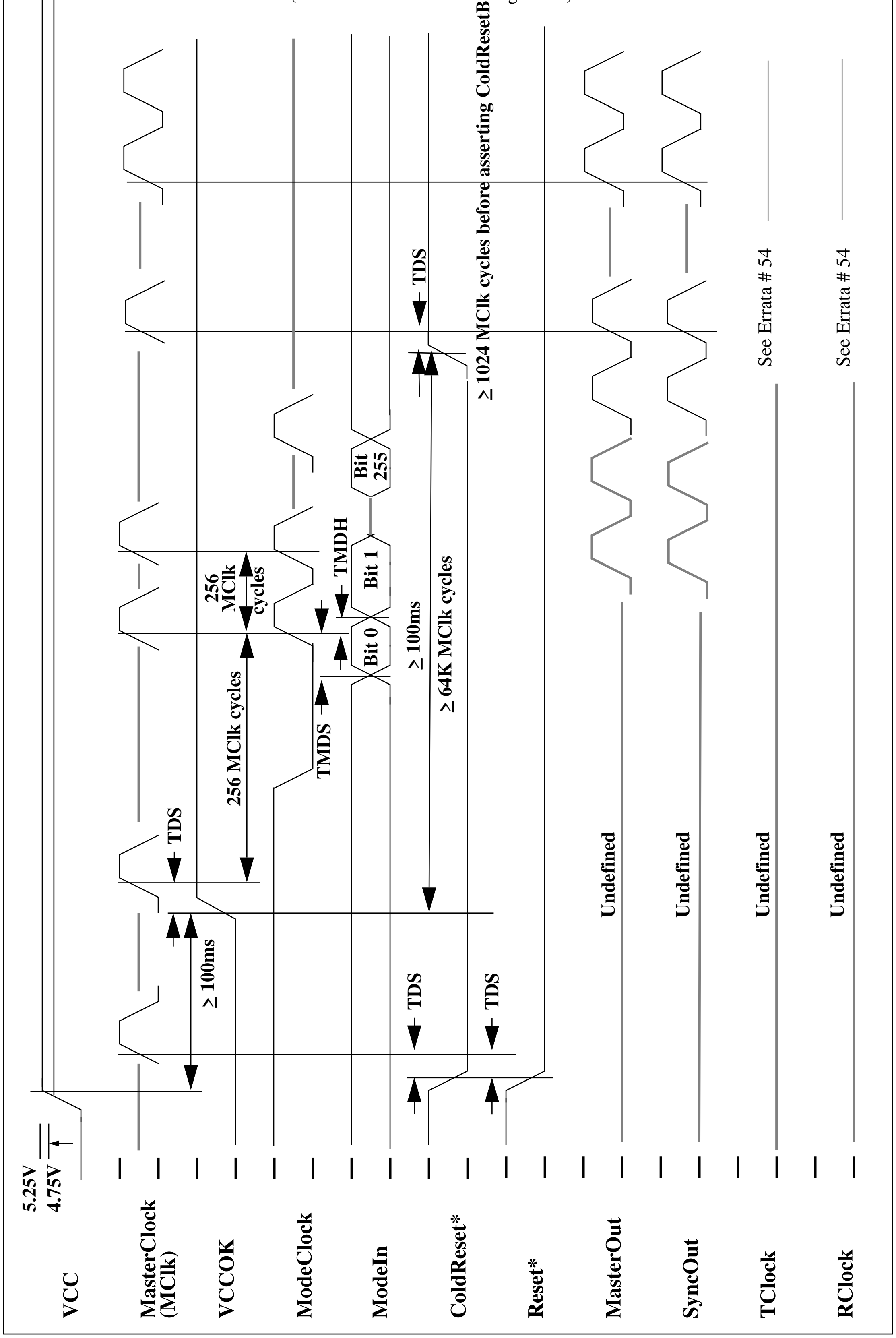
2) If the divide is in the taken branch delay slot, where the target takes RF exception and gets an I-cache miss for the exception vector or where I-cache miss occurs for the target address, under the above mentioned scenarios, the div would get wrong results.

```
##1
        j           r2                      # to next page mapped or unmapped
        div         r8,r9                   # this bug would be there as long as there is an
        nop                                 # ICache miss and the "data pattern" is present
##2
        beq         r0, r0, NextPage        # to Next page
        div         r8,r9
        nop
```

This bug is present for div, divu, ddiv, and ddivu instructions.

Figure 1a: Power-on Reset or Cold Reset - Workaround for Bug# 54

(Waveforms continued on figure 1b)



54. R4000PC, R4000SC: During the power up sequence, the period of TClk and RClk may not be as expected for at least 1024 MasterClock cycles following the de-assertion of ColdResetB. The above bug is true for all system interface divisors. Furthermore, if the system interface divisor other than 2 (i.e. 3,4,6 & 8) is used, TClk and RClk's rising edge may not align to the rising edge of the MasterClock, as specified.

This may not be a problem for systems using only TClk for the external logic. However, certain systems like -

1) systems using MasterClock for the external logic
2) system with external PLL which needs to lock with the internal PLL
3) systems with multiple processors operating in lock-step

this bug may cause problems.

Thus, for system with interface divisor set to two, the only violation will be that the TClk and RClk's period might not be valid within 64 MasterClock after the ColdResetB is de-asserted.

Workaround: Assert a second ColdResetB after at least 1024 MasterClock cycles from the de-assertion of the first ColdResetB. ResetB should remain asserted until at least 64 Master-Clock cycles following the de-assertion of the second ColdResetB. TClk and RClk will then be valid 64 MasterClock cycles after the de-assertion of the second ColdResetB and their edge alignment with respect to MasterClock will be as specified. Figure 1a & 1b show the wave-forms in more detail. Note that the waveforms in Figure 1b are continuation of the those in fig-ure 1a.

55. R4000SC : Processor hangs when “NoMPmode” bit is set to 1.

Setting the Boot-Mode bit# 41, “NoMPmode”, improves the performance of R4x00SC by avoiding invalidation of the existing line in the SCache after the Scache miss.

Workaround: There is no workaround for this bug.

Figure 1b: Power-on Reset or Cold Reset - Workaround for Bug# 54(cont.)