TC86R4400 3.45V 64-bit RISC Microprocessor

Description

The Toshiba TC86R4400 family consists of high-performance 64-bit RISC microprocessors that deliver excellent processing solutions over a wide variety of applications and prices. Systems applications for this family range from inexpensive, highly integrated desktop systems through servers whose CPU performance rivals that of current mainframes, and whose address space requirements are not met by the current generation of microprocessors. Embedded processing applications for the TC86R4400 family of microprocessors range from high-performance laser printer controllers to high-precision avionics controllers.

High integer performance, as well as floating-point performance, has been achieved through techniques such as super-pipelining, on-chip caches, a pipelined floating-point unit, two-level cache memory, and a high-performance on-chip translation lookaside buffer (TLB). The TC86R4400's cache and Memory Management Unit (MMU) offer high performance in handling both large-address-space tasks and a large number of users. These features allow the design of balanced systems, suitable not only for technical and graphics applications, but also for commercial applications like transaction processing with fault-tolerant support. Over a wide range of realistic benchmarks, the TC86R4400 family of microprocessors delivers from 43 to 70 SPEC marks of sustained performance at 100MHz internal or 50MHz external.

The TC86R4400 provides a compatible, timely, and necessary path from 32-bit to true 64-bit computing for users and software developers. The path to the on-chip caches is 64-bits wide. The on-chip floating-point co-processor is a 64-bit floating point unit (FPU). The TC86R4400 provides 64-bit integers, registers, and a flat 64-bit virtual address space. Compatibility with existing 32-bit application code is maintained, however, and an efficient mix of 32-bit and 64-bit programs can run on the same TC86R4400-based machine.

The 64-bit addressing capability of the TC86R4400 allows designers to build operating systems with extensive file mapping, which allows direct access to files without explicit I/O calls. This addressing capability paves the way for the next generation of video technology and documentation with photographs and high-quality images. In addition, CAD applications with huge data bases of complex, structured objects, geographic information systems, and technical number-crunching applications with large data sets will benefit greatly from this capability.

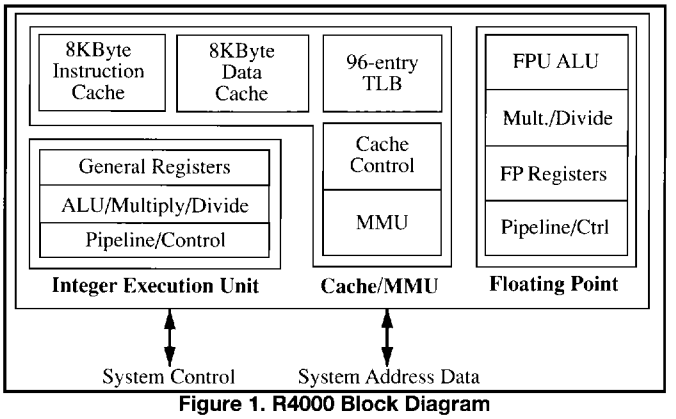


Figure 1. R4000 Block Diagram

Key features of the TC86R4400 family include the following:

- True 64-bit microprocessor with 64-bit integer and floating-point operations, registers, and virtual addresses
- Full compatibility with earlier 32-bit MIPS microprocessors
- Dual instruction issue with no restrictions on the type of instructions issued
- Available in two internal speed grades: 150MHz (3.3V part), and 200MHz (3.45V part).
- On-chip 8KByte instruction cache, 8KByte data cache, and an optional 128-bit secondary cache interface
- On-chip Memory Management Unit (MMU) containing a fully associative TLB whose entries have a variable page size ranging from 4KB to 16MByte
- On-chip ANSI/IEEE-754 standard floating-point unit with precise exceptions
- Thirty-two doubleword (64-bit) general-purpose registers and sixteen doubleword (64-bit) floating-point registers
- 36-bit physical address accessing 64GBytes of physical memory
- Two (for SC) hardware interrupts including NMI
- 64-bit cache-coherent system interface with flexible and high-performance multiprocessing support
- Fault-tolerant support
- Dynamically configurable big-endian or little-endian byte ordering
- Timing flexibility for 128-bit secondary cache interface and 64-bit system interface to allow speed matching of logic and memory components

Superpipeline

Instruction pipelines divide the execution of each instruction into several discrete portions and then execute multiple instructions simultaneously. The instruction pipeline technique can be likened to an assembly line in which the instruction progresses from one specialized stage to the next unit until it is completed or issued.

To achieve the high performance required in a third generation RISC design, the TC86R4400 exploits instruction-level parallelism using a superpipelined micro-architecture. The TC86R4400 implements an eight-stage superpipeline which places no restrictions on instruction issue. Any two instructions can be issued each cycle under normal circumstances. Because instructions can be issued in any order, existing applications can realize the full benefit of the TC86R4400 without requiring recompilation.

The internal pipeline of the TC86R4400 operates at a frequency of 100 MHz, which is twice the external clock frequency. The eight-stage superpipeline of the TC86R4400 is made possible by pipelining cache accesses, shortening register access times, implementing virtually indexed primary caches, and allowing the latency of functional units to span multiple pipeline clock cycles (pcycles).

After extensive simulation of methods of exploiting instruction-level parallelism, superpipelining was chosen because it improves integer performance commensurate with floating-point performance. This method retains the balance of a mainframe CPU without skewing the performance heavily in favor of technical applications. Compared with other methods of exploiting instruction-level parallelism, this approach results in less complex logic, faster cycle times, quicker design cycles, and lower cost.

The execution of a single TC86R4400 CPU instruction consists of the following eight primary steps:

IF Instruction fetch first half. Virtual address is presented to the instruction cache (I-cache) and TLB.

IS Instruction fetch second half. The I-cache outputs the instruction and the TLB generates the physical address.

RF Register fetch. Three activities occur in parallel: the instruction is decoded and a check is made for interlock conditions; an instruction tag check is made to determine if there is a cache hit or not; and operands are fetched from the register file.

EX Instruction execution. One of three activities can occur: if the instruction is a register-to-register operation, an arithmetic, logical, shift, multiply, or divide operation is performed; if the instruction is a load and store, the data virtual address is calculated; if the instruction is a branch, the branch target virtual address is calculated and branch conditions are checked.

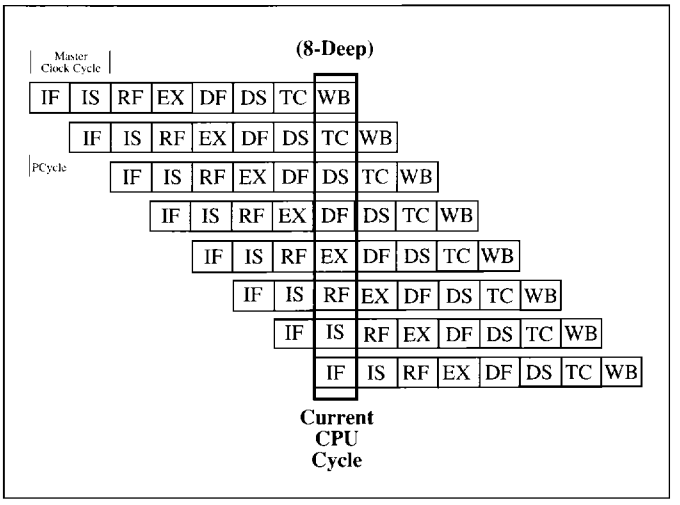
DF Data fetch (D-cache) first half. A virtual address is presented to the D-cache and TLB.

DS Data fetch second half. The D-cache outputs the instruction and the TLB generates the physical address.

TC Tag check. A tag check is performed for loads and stores to determine if there is a hit or not.

WB Write back. The instruction result is written back to the register file.

Because the TC86R4400 uses an eight-stage pipeline, execution of the eight instructions overlaps, as shown in the following figure.



Memory Cache Systems

The TC86R4400 cache management features permit implementation of a high-performance operating system. Highly efficient performance is maintained and the lower pipeline cycle time is exploited using virtually indexed caches.

To be effectively matched with the TC86R4400's superpipeline, the primary cache is virtually indexed and physically tagged. Using a part of the virtual index to address the cache enables the virtual address translation to happen in parallel with the primary cache lookup. The physically tagged cache implies that the cache need not be flushed on context switches.

The environment in which the TC86R4400 cache management operations function has the following principal characteristics:

- The presence or absence of a secondary cache
- A secondary cache that is joint
- Primary and secondary caches that are write back
- Primary caches that are virtually indexed and physically tagged
- Secondary caches that are physically indexed and physically tagged
- Externally initiated cache coherency operations

The operating system environment allows significant amounts of data copying and data initialization. Cache management operations increase the efficiency of such operating systems activity.

To achieve its high performance, the TC86R4400 supports a cache memory hierarchy that increases memory access bandwidth and reduces the latency of load and store instructions. For fast cache access, the TC86R4400 incorporates on-chip instruction and data caches. These high-speed caches keep the pipeline full by feeding the processor at a peak rate of two instructions and two data words every master clock cycle (20 ns).

The TC86R4400SC interfaces to an optional secondary cache. The design of the cache memory hierarchy minimizes miss penalty and miss rates on memory accesses, thereby improving overall system performance. The size of the secondary cache can range from 128KBytes to 4MBytes.

The secondary cache is assumed to consist of one bank of industry-standard static RAM (SRAM) with output enables. The secondary cache consists of a quadword (128 bit) wide data array and a 25-bit wide tag array. Check fields are added to both the data and tag arrays to improve data integrity. The maximum secondary cache size is 4MBytes; the minimum secondary cache size is 128KBytes for a joint cache. The secondary cache is direct-mapped and is addressed with the lower part of the physical address.

The line size and access time of the secondary cache are configurable. The flexibility of the design caches means that the requirements of virtually any system can be met; moreover, design of a secondary cache using standard low-cost SRAMs is permitted.

The TC86R4400 provides all of the secondary cache control circuitry, including Error Correction Code (ECC) protection, on chip. The secondary cache interface consists of a 128-bit data bus, a 25-bit tag bus, an 18-bit address bus and SRAM control signals. To minimize cache miss latency, the interface to the secondary cache is 128 bits wide.

Memory Management

The TC86R4400 contains a flexible, on-chip memory management unit that incorporates a fully associative TLB that performs the virtual to physical address translation. Each entry in the TLB can map from 4KBytes to 16MBytes of virtual memory for a total mapping of up to 1.5GBytes. As a result, it is possible to map very large contiguous regions of virtual memory using a single translation. This feature is particularly useful in mapping large physical regions such as frame buffers and large databases, and in manipulating large data arrays in scientific and engineering number-crunching applications.

In the TC86R4400 TLB, critically needed address translations can be programmed to remain locked in the TLB. This feature is especially useful in realtime systems where predictability of system response is a critical requirement.

Instruction Set

The TC86R4400 implements the extended MIPS instruction set architecture; this improves performance and adds functional capabilities while maintaining complete applications binary compatibility with earlier MIPS microprocessors. The extensions result in better code density, greater multiprocessing support, improved performance for commonly used code sequences in operating system kernels, and faster execution of floating-point intensive applications. All resource dependencies are transparent to the programmer in this instruction set. Every instruction is 32 bits wide to allow easy and fast decode.

New instructions have been defined to take advantage of the 64-bit architecture. When operating as a 32-bit microprocessor, the TC86R4400 takes an exception to these new instructions.

As shown in the illustration, there are three instruction formats: immediate (I-type), jump (J-type), and register (R-type). Using only these three instruction formats simplifies instruction decoding; the compiler can synthesize more complicated (and less frequently used) operations and addressing modes using sequences of these simple instructions.

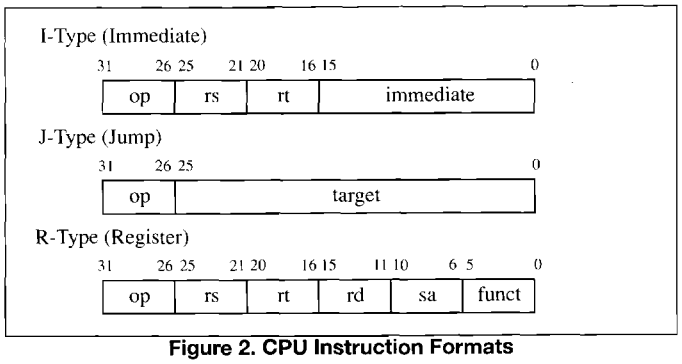


Figure 2. CPU Instruction Formats

The instruction set can be divided into the following groups:

- **Load and Store** instructions that move data between memory and general registers. They are all I-type instructions, because the only addressing mode supported is base register plus 16-bit, signed immediate offset.
- **Computational** instructions that perform arithmetic, logical, shift, multiply, and divide operations on values in registers. They occur in both R-type (both the operands and the result are stored in registers) and I-type (one operand is a 16-bit immediate value) formats.
- **Jump and Branch** instructions that change the control flow of a program. Jumps are always to a paged, absolute address formed by combining a 26-bit target address with the high-order bits of the program counter (J-type format) or register addresses (R-type format). Branches have 16-bit offsets relative to the program counter (I-type). JumpAndLink instructions save a return address in register 31.
- **Coprocessor** instructions that perform operations in the coprocessors. Coprocessor load and store instructions are I-type.
- **Coprocessor 0** instructions that perform operations on CP0 registers to manipulate the memory management and exception handling facilities of the processor (see table following).
- **Special** instructions that perform system calls and breakpoint operations. These instructions are always R-type.
- **Exception** instructions that cause a branch to the general exception-handling vector based upon the result of a comparison. These instructions occur in both R-type (both the operands and the result are registers) and I-type (one operand is a 16-bit immediate value) formats.

CP0 Instructions

Op	Description
DMFC0	Doubleword Move From CP0
DMTC0	Doubleword Move To CP0
MTC0	Move to CP0
MFC0	Move from CP0
TLBR	Read Indexed TLB Entry
TLBWI	Write Indexed TLB Entry
TLBWR	Write Random TLB Entry
TLBP	Probe TLB for Matching Entry
ERET	Exception Return
CACHE	Cache Operation

JTAG Interface

The JTAG boundary scan mechanism is compatible with JTAG specifications; it uses four JTAG interface pins: JTAG serial data in (JDTI), JTAG serial data out (JTDO), JTAG command signal (JTMS), and JTAG serial clock input (JTCK). The JTAG boundary scan mechanism permits testing of the interconnect between the TC86R4400 processor, the attached printed circuit board, and the other components on the board. The JTAG boundary scan mechanism also permits rudimentary low-speed logical testing of the secondary cache RAMs; it does not permit testing of the TC86R4400 processor itself, however. The TC86R4400 processor contains the required JTAG registers—Test Access Port (TAP) controller, JTAG Instruction Register, JTAG Boundary Scan Register, and JTAG Bypass Register—and executes the standard JTAG EXTEST operation associated with external test functionality testing.

The JTAG TAP consists of the four pins mentioned above. Data is serially scanned into one of three registers (Instruction Register, Bypass Register, Boundary Scan Register) from the JTDI pin, and is scanned out from the selected one of these registers onto the JTDO pin. The JTDI input feeds the least significant bit (LSB) of the selected register, and the most significant bit (MSB) of the selected register appears on the JTDO output. The JTMS input controls the state transitions of the main TAP controller state machine. Data on the JTDI and JTMS pins is sampled on the rising edge of the JTCK input clock signal. Data on the JTDO pin changes on the falling edge of the JTCK clock signal.

The TC86R4400 implements the 16-state JTAG TAP controller as defined in the IEEE JTAG specification.The TAP controller state machine can be put in its Reset state in one of two ways. Deassertion of the VCCOk input will reset the TAP controller. Keeping the JTMS input signal asserted through five consecutive rising edges of the JTCK clock input will also send the TAP controller state machine into its Reset state. In either case, keeping JTMS asserted will maintain the Reset state.

The TC86R4400's JTAG Instruction Register is three bits wide and comprises two stages: the shift register stage and the parallel output latch. When the TAP controller is in the Reset state, the value 7 (111) is loaded into the parallel output latch, so that the Bypass Register is selected as the default. When the TAP controller is in the Capture-IR state, the value 4 (100) is loaded into the shift register stage. When the TAP controller is in the Shift-IR state, data is serially shifted into the shift register stage of the Instruction Register from the JTDI input pin, and the MSB of the Instruction Register's shift register state is shifted out onto the JTDO pin. When the TAP controller is in the Update-IR state, the current data in the shift register stage is loaded into the parallel output latch.

The Bypass Register is one bit wide. When the TAP controller is in the Shift-DR (Bypass) state, the data on the JTDI pin is shifted into the Bypass Register, and the Bypass Register's output is shifted out onto the JTDO output pin.

The Boundary Scan Register is 319 bits wide. The three most significant bits control the output enables on the various bidirectional buses. The most significant bit is the JTAG output enable bit for the SysAD, SysADC, SysCmd, and SysCmdP buses. The next most significant bit is the JTAG output enable for the SCData and SCDChk buses. The third most significant bit is the JTAG output enable for the SCTag and SCTChk buses. The remaining 316 bits correspond to the 316 signal pads of the TC86R4400.

When the TAP controller is in the Reset state, the three most significant bits of the Boundary Scan Register are set to "0" (the default JTAG output enable control on all the bidirectional pins is to disable the outputs). When the TAP controller is in the Capture-DR (Boundary Scan) state, the data currently present on all the TC86R4400's input and I/O pins are latched into the Boundary Scan Register. The Boundary Scan Register bits corresponding to output pins are arbitrary in this state and must not be checked during the scan out process. When the TAP controller is in the Shift-DR (Boundary Scan) state, data is serially shifted into the Boundary Scan Register from the JTDI pin, and the contents of the Boundary Scan Register are shifted out onto the JTDO pin. When the TAP controller is in the Update-DR (Boundary Scan) state, the current data in the Boundary Scan Register is latched into its parallel output latch, and the bits corresponding to output pins and those I/O pins whose outputs are enabled (by the three MSBs of the Boundary Scan Register) are enabled onto the TC86R4400's pins.

The scan order of the 316 scan bits of the TC86R4400 is listed on the following page starting from JTDI and ending with JTDO.

CPU Instruction Set Architecture (ISA)

OP	Description	OP	Description
Load and Store Instructions		**Multiply and Divide Instructions**	
LB	Load Byte	MULT	Multiply
LBU	Load Byte Unsigned	MULTU	Multiply Unsigned
LH	Load Halfword	DIV	Divide
LHU	Load Halfword Unsigned	DIVU	Divide Unsigned
LW	Load Word	MFHI	Move From HI
LWL	Load Word Left	MTHI	Move To HI
LWR	Load Word Right	MFLO	Move From LO
SB	Store Byte	MTLO	Move To LO
SH	Store Halfword	**Jump and Branch Instructions**	
SW	Store Word	J	Jump
SWL	Store Word Left	JAL	Jump And Link
SWR	Store Word Right	JR	Jump Register
Arithmetic Instructions (ALU Immediate)		JALR	Jump And Link Register
		BEQ	Branch on Equal
ADDI	Add Immediate	BNE	Branch on Not Equal
ADDIU	Add Immediate Unsigned	BLEZ	Branch on Less than or Equal to Zero
SLTI	Set on Less Than Immediate	BGTZ	Branch on Greater Than Zero
SLTIU	Set on Less Than Immediate Unsigned	BLTZ	Branch on Less Than Zero
ANDI	AND Immediate	BGEZ	Branch on Greater than or Equal to Zero
ORI	OR Immediate	BLTZAL	Branch on Less Than Zero And Link
XORI	Exclusive OR Immediate	BGEZAL	Branch on Greater than or Equal to Zero And Link
LUI	Load Upper Immediate	**Co-processor Instructions**	
Arithmetic Instructions (3-operand, R-type)		LWCz	Load Word to Co-processor z
		SWCz	Store Word from Co-processor z
ADD	Add	MTCz	Move To Co-processor z
ADDU	Add Unsigned	MFCz	Move From Co-processor z
SUB	Subtract	CTCz	Move Control to Co-processor z
SUBU	Subtract Unsigned	CFCz	Move Control From Co-processor z
SLT	Set on Less Than	COPz	Co-processor Operation z
SLTU	Set on Less Than Unsigned	BCzT	Branch on Co-processor z True
AND	AND	BCzF	Branch on Co-processor z False
OR	OR	**Special Instructions**	
XOR	Exclusive OR	SYSCALL	System Call
NOR	NOR	BREAK	Breakpoint
Shift Instructions			
SLL	Shift Left Logical		
SRL	Shift Right Logical		
SRA	Shift Right Arithmetic		
SLLV	Shift Left Logical Variable		
SRLV	Shift Right Logical Variable		
SRAV	Shift Right Arithmetic Variable		

The tables on pages 5 and 6 identify the Instruction Set Architecture (ISA) and the TC86R4400 instructions that are extensions to the instruction set architecture. These instructions result in code space reductions, multiprocessor support, and improved performance

Extensions to the ISA

OP	Description	OP	Description
Load and Store Instructions		**Multiply and Divide Instructions**	
LD	Load Doubleword	DMULT	Doubleword Multiply
LDL	Load Doubleword Left	DMULTU	Doubleword Multiply Unsigned
LDR	Load Doubleword Right	DDIV	Doubleword Divide
LL	Load Linked	DDIVU	Doubleword Divide Unsigned
LLD	Load Linked Doubleword	**Jump and Branch Instructions**	
LWU	Load Word Unsigned	BEQL	Branch on Equal Likely
SC	Store Conditional	BNEL	Branch on Not Equal Likely
SCD	Store Conditional Doubleword	BLEZL	Branch on Less than or Equal to Zero Likely
SD	Store Doubleword	BGTZL	Branch on Greater Than Zero Likely
SDL	Store Doubleword Left	BLTZL	Branch on Less Than Zero Likely
SDR	Store Doubleword Right	BGEZL	Branch on Greater than or Equal to Zero Likely
SYNC	Sync	BLTZALL	Branch on Less Than Zero And Link Likely
Arithmetic Instructions (ALU Immediate)		BGEZALL	Branch on Greater than or Equal to Zero And Link Likely
		BCzTL	Branch on Coprocessor z True Likely
DADDI	Doubleword Add Immediate	BCzFL	Branch on Coprocessor z False Likely
DADDIU	Doubleword Add Immediate Unsigned	**Exception Instructions**	
Arithmetic Instructions (3-operand, R-type)		TGE	Trap if Greater Than or Equal
		TGEU	Trap if Greater Than or Equal Unsigned
DADD	Doubleword Add	TLT	Trap if Less Than
DADDU	Doubleword Add Unsigned	TLTU	Trap if Less Than Unsigned
DSUB	Doubleword Subtract	TEQ	Trap if Equal
DSUBU	Doubleword Subtract Unsigned	TNE	Trap if Not Equal
Shift Instructions		TGEI	Trap if Greater Than or Equal Immediate
DSLL	Doubleword Shift Left Logical	TGEIU	Trap if Greater Than or Equal Immediate Unsigned
DSRL	Doubleword Shift Right Logical	TLTI	Trap if Less Than Immediate
DSRA	Doubleword Shift Right Arithmetic	TLTIU	Trap if Less Than Immediate Unsigned
DSLLV	Doubleword Shift Left Logical Variable	TEQI	Trap if Equal Immediate
DSRLV	Doubleword Shift Right Logical Variable	TNEI	Trap if Not Equal Immediate
DSRAV	Doubleword Shift Right Arithmetic Variable	**Co-processor Instructions**	
DSLL32	Doubleword Shift Left Logical + 32	DMFCz	Doubleword Move From Co-processor z
DSRL32	Doubleword Shift Right Logical + 32	DMTCz	Doubleword Move To Co-processor z
DSRA32	Doubleword Shift Right Arithmetic + 32	LDCz	Load Double Co-processor z
		SDCz	Store Double Co-processor z

in operating system kernel code sequences and in situations where run-time bounds checking is frequently performed.

JTAG Ordering

1. SCDChk[13]
2. SysADC[1]
3. SCDChk[1]
4. SysADC[5]
5. SCDChk[5]
6. Status[0]
7. Status[1]
8. Status[2]
9. Status[3]
10. IvdErr*
11. Status[4]
12. IvdAck*
13. Status[5]
14. Status[6]
15. Status[7]
16. SCDChk[7]
17. SysADC[7]
18. SCDChk[3]
19. SysADC[3]
20. SCDChk[15]
21. VCCOk
22. SCTag[16]
23. SCDChk[11]
24. SCData[63]
25. SysAD[63]
26. SCData[31]
27. SysAD[31]
28. SCData[127]
29. SCTag[17]
30. SCData[95]
31. SCData[62]
32. SysAD[62]
33. SCData[30]
34. SysAD[30]
35. SCData[126]
36. SCTag[18]
37. SCData[94]
38. RClock[1..0] (share the same JTAG bit)
39. SCTag[19]
40. SCData[61]
41. SysAD[61]
42. SCData[29]
43. SysAD[29]
44. SCData[125]
45. Reset*
46. SCTag[20]
47. SCData[93]
48. SCData[60]
49. SysAD[60]
50. SCData[28]
51. SysAD[28]
52. SCData[124]
53. ColdReset*
54. SCTag[21]
55. SCData[92]
56. SCData[59]
57. SysAD[59]
58. SCData[27]
59. SysAD[27]
60. SCData[123]
61. IOIn
62. SCTag[22]
63. SCData[91]
64. SCData[58]
65. SysAD[58]
66. SCData[26]
67. SysAD[26]
68. SCData[122]
69. IOOut
70. SCTag[23]
71. SCData[90]
72. SCData[57]
73. SysAD[57]
74. SCData[25]
75. SysAD[25]
76. SCData[121]
77. -
78. SCTag[24]
79. SCData[89]
80. SCData[56]
81. SysAD[56]
82. SCData[24]
83. SysAD[24]
84. SCData[120]
85. -
86. SCTChk[0]
87. SCData[88]
88. SCDChk[6]
89. SysADC[6]
90. SCDChk[2]
91. SysADC[2]
92. SCDChk[14]
93. NMI*
94. SCTChk[1]
95. SCDChk[10]
96. SCData[55]
97. SysAD[55]
98. SCData[23]
99. SysAD[23]
100. SCData[119]
101. Release*
102. SCTChk[2]
103. SCData[87]
104. SCData[54]
105. SysAD[54]
106. SysAD[22]
107. ModeIn
108. SCData[22]
109. RdRdy*
110. SCData[118]
111. SCData[86]
112. SCData[53]
113. SysAD[53]
114. SCData[21]
115. SysAD[21]
116. SCData[117]
117. ExtRqst*
118. SCTChk[3]
119. SCData[85]
120. SCData[52]
121. SysAD[52]
122. SCData[20]
123. SysAD[20]
124. SCData[116]
125. ValidOut*
126. SCTChk[4]
127. SCData[84]
128. SCData[51]
129. SysAD[51]
130. SCData[19]
131. SysAD[19]
132. SCData[115]
133. ValidIn*
134. SCTChk[5]
135. SCData[83]
136. SCAddr0W,X (share the same JTAG bit)
137. SCAddr0Y,Z (share the same JTAG bit)
138. SCAddr[1]
139. SCData[50]
140. SysAD[50]
141. SCData[18]
142. SysAD[18]
143. SCData[114]
144. Int*[0]
145. SCTChk[6]
146. SCData[82]
147. SCData[49]
148. SysAD[49]
149. SCData[17]
150. SysAD[17]
151. SCData[113]
152. SCAddr/Int*[1]
153. SCAddr[3]
154. SCData[81]
155. SCData[48]
156. SysAD[48]
157. SCData[16]
158. SysAD[16]
159. SCData[112]
160. SCAddr/Int*[2]
161. SCAddr[5]
162. SCData[80]
163. SCAddr[6]
164. SCAddr[7]
165. SCAddr[8]
166. SCAddr[9]
167. SCAddr[10]
168. SCAddr[11]
169. -
170. SCAddr[12]
171. SCAddr[13]
172. SCAddr[14]
173. SCAddr[15]
174. SCAddr[16]
175. SCAddr[17]
176. SCData[64]
177. SCAPar[0]
178. SCAPar[1]/Int*[3]
179. SCData[96]
180. SysAD[0]
181. SCData[0]
182. SysAD[32]
183. SCData[32]
184. SCData[65]
185. SCAPar[2]
186. SCOE*/Int*[4]
187. SCData[97]
188. SysAD[1]
189. SCData[1]
190. SysAD[33]
191. SCData[33]
192. SCData[66]
193. SCDCS*
194. SCTCS*/Int*[5]
195. SCData[98]
196. SysAD[2]
197. SCData[2]
198. SysAD[34]
199. SCData[34]
200. SCTag[0]
201. SCWrW,X* (share the same JTAG bit)
202. SCWrY, Z* (share the same JTAG bit)
203. SCData[67]
204. SCTag[1]
205. SysCmd[0]
206. SCData[99]
207. SysAD[3]
208. SCData[3]
209. SysAD[35]
210. SCData[35]
211. SCData[68]
212. SCTag[2]
213. SysCmd[1]
214. SCData[100]
215. SysAD[4]
216. SCData[4]
217. SysAD[36]
218. SCData[36]
219. SCData[69]
220. SCTag[3]
221. SysCmd[2]
222. SCData[101]
223. SysAD[5]
224. SCData[5]
225. SysAD[37]
226. SCData[37]
227. SCData[70]
228. WrRdy*
229. ModeClock
230. SCData[102]
231. SysAD[6]
232. SCData[6]
233. SysAD[38]
234. SCData[38]
235. SCData[71]
236. SCTag[4]
237. SysCmd[3]
238. SCData[103]
239. SysAD[7]
240. SCData[7]
241. SysAD[39]
242. SCData[39]
243. SCDChk[8]
244. SCTag[5]
245. SysCmd[4]
246. SCDChk[12]
247. SysADC[0]
248. SCDChk[0]
249. SysADC[4]
250. SCDChk[4]
251. SCData[72]
252. SCTag[6]
253. SysCmd[5]
254. SCData[104]
255. SysAD[8]
256. SCData[8]
257. SysAD[40]
258. SCData[40]
259. SCData[73]
260. SCTag[7]
261. SysCmd[6]
262. SCData[105]
263. SysAD[9]
264. SCData[9]
265. SysAD[41]
266. SCData[41]
267. SCData[74]
268. SCTag[8]
269. SysCmd[7]
270. SCData[106]
271. SysAD[10]
272. SCData[10]
273. SysAD[42]
274. SCData[42]
275. SCData[75]
276. SCTag[9]
277. SysCmd[8]
278. SCData[107]
279. SysAD[11]
280. SCData[11]
281. SysAD[43]
282. SCData[43]
283. SCData[76]
284. SCTag[10]
285. SysCmdP
286. SCData[108]
287. SysAD[12]
288. SCData[12]
289. SysAD[44]
290. SCData[44]
291. SCData[77]
292. SCTag[11]
293. Fault*
294. SCData[109]
295. SysAD[13]
296. SCData[13]
297. SysAD[45]
298. SCData[45]
299. SCTag[12]
300. TClock[1..0] (share the same JTAG bit)
301. SCData[78]
302. SCTag[13]
303. SCData[110]
304. SysAD[14]
305. SCData[14]
306. SysAD[46]
307. SCData[46]
308. SCData[79]
309. SCTag[14]
310. SCData[111]
311. SysAD[15]
312. SCData[15]
313. SysAD[47]
314. SCData[47]
315. SCDChk[9]
316. SCTag[15]

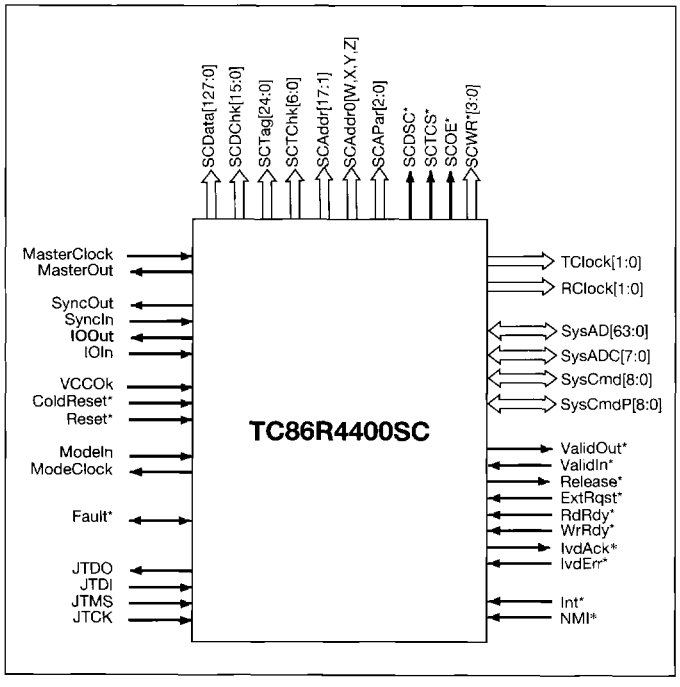
- SCTag_OE (JTAG output enable control for SCTag and SCTChk buses)
- SCData_OE (JTAG output enable control for SCData and SCDChk buses)
- SysAD_OE (JTAG output enable control for SysAD, SysADC, SysCmd and SysCmdP buses)

Logic

The TC86R4400 processor is available in two different configurations: the TC86R4400SC, which include a 128-bit wide secondary cache bus, and the TC86R4400MC.

TC86R4400SC

The TC86R4400SC is available in a 447-pin Pin Grid Array (PGA). This processor supports a secondary cache interface and is ideal in systems where high performance is desired. This component supports a 128KByte to 4MByte secondary cache made from standard SRAMs. This flexibility allows system designers to make price/performance trade-offs in cache subsystem designs.



Pin Descriptions

Pin names, signal types, and descriptions associated with the TC86R4400 appear below.

System Interface

The system interface signals comprise the interface between the TC86R4400 and other components in the system.

ExtRqst*:	**External request**	**Input**
An external agent asserts ExtRqst* to request use of the system interface. The TC86R4400 grants the request by asserting Release*.		
IvdAck*:	**Invalidate acknowledge**	**Input**
An external agent asserts IvdAck* to signal successful completion of a processor invalidate or update request (TC86R4400SC only). Tie to V_{CC}.		
IvdErr*:	**Invalidate error**	**Input**
An external agent asserts IvdErr* to signal unsuccessful completion of a processor invalidate or update request (TC86R4400SC only). Tie to V_{CC}.		
Release*:	**Release interface**	**Output**
In response to the assertion of ExtRqst*, the TC86R4400 asserts Release* to signal the requesting device that the system interface is available.		
RdRdy*:	**Read ready**	**Input**
The external agent asserts RdRdy* to indicate that it can accept processor read, invalidate, or update requests in both secondary-cache and no-secondary-cache mode or can accept a read followed by write request, a read followed by a potential update request, or a read followed by a potential update followed by a write request in secondary cache mode.		
SysAD(63:0):	**System address/data bus**	**Input/Output**
A 64-bit address and data bus for communication between the processor and an external agent.		
SysADC(7:0):	**System address/data check bus**	**Input/Output**
An 8-bit bus containing check bits for the SysAD bus.		
SysCmd(8:0):	**System command/data identifier bus parity**	**Input/Output**
A 9-bit bus for command and data identifier transmission between the processor and an external agent.		
SysCmdP:	**System command/data identifier bus parity**	**Input/Output**
A single, even-parity bit for the SysCmd bus.		
ValidIn*:	**Valid input**	**Input**
An external agent asserts ValidIn* when it is driving a valid address or data on the SysAD bus and a valid command or data identifier on the SysCmd bus.		
ValidOut*:	**Valid output**	**Output**
The TC86R4400 asserts ValidOut* when it is driving a valid address or data on the SysAD bus and a valid command or data identifier on the SysCmd bus.		
WrRdy*:	**Write ready**	**Input**
An external agent asserts WrRdy* when it can accept a processor write request		

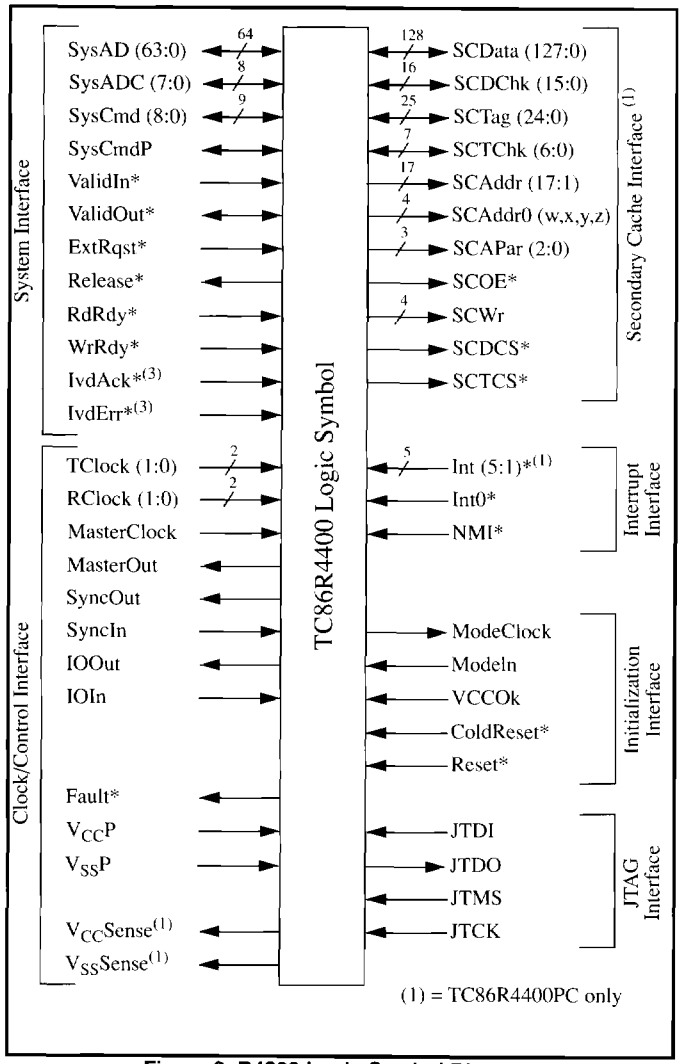


Figure 3. R4000 Logic Symbol Diagram

Clock/Control Interface

The clock/control interface signals comprise the interface for clocking and maintenance functions.

IOOut:	**I/O output**	**Output**
Output slew rate control feedback loop output. Must be connected to IOIn through a delay loop that models the IO path from the TC86R4400 to an external agent.		
IOIn:	**I/O input**	**Input**
Output slew rate control feedback loop input (see IOOut).		
MasterClock:	**Master clock**	**Input**
Master clock input establishes the processor operating frequency.		
MasterOut:	**Master clock out**	**Output**
Master clock output aligned with MasterClock.		
RClock(1:0):	**Receive clocks**	**Output**
Two identical receive clocks that establish the system interface frequency.		
SyncOut:	**Synchronization clock out**	**Output**
Synchronization clock output. Must be connected to SyncIn through an interconnect that models the interconnect between MasterOut, TClock, RClock, and the external agent.		
SyncIn:	**Synchronization clock in**	**Input**
Synchronization clock input.		
TClock(1:0):	**Transmit clocks**	**Output**
Two identical transmit clocks that establish the system interface frequency.		
Fault*:	**Fault**	**Output**
The TC86R4400 asserts Fault* to indicate a mismatch output of boundary comparators, and indication of system interface input parity or ECC errors.		
VccP:	**Quiet VCC for PLL**	**Input**
Quiet Vcc for the internal phase locked loop.		
VccSense:	**VCC sense**	**Input/Output**
This is a special pin used only in component testing and characterization. It provides a separate, direct connection from the on-chip VCC node to a package pin without attaching to the in-package power planes. Test fixtures treat VccSense as an analog output pin: the voltage at this pin directly shows the behavior of the on-chip VCC. Thus, characterization engineers can easily observe the effects of di/dt noise, transmission line reflections, etc. VccSense should be connected to VCC in functional system designs.		
VssP:	**Quiet VSS for PLL**	**Input**
Quiet Vss for the internal phase locked loop.		
VssSense:	**VSS sense**	**Input/Output**
VssSense provides a separate, direct connection from the on-chip VSS node to a package pin without attaching to the in-package ground planes. VssSense should be connected to VSS in functional system designs.		

Secondary Cache Interface

The secondary cache interface signals comprise the interface between the TC86R4400SC and the secondary cache.

SCAddr(17:1)	**Secondary cache address bus**	**Output**
SCAddr0W:	**Secondary cache address LSB**	**Output**
SCAddr0X:	**Secondary cache address LSB**	**Output**
SCAddr0Y:	**Secondary cache address LSB**	**Output**
SCAddr0Z:	**Secondary cache address LSB**	**Output**
The 18-bit address bus for the secondary cache. Bit 0 has four output lines to provide additional drive current.		
SCAPar(2:0):	**Secondary cache address parity bus**	**Output**
A 3-bit bus that carries the parity of the SCAddr bus and the cache control lines SCWR*, SCDCS* and SCTCS*. The individual bit definitions are: SCAPar2 - Even Parity for SCAddr(17:12) and SCWR* SCAPar1 - Even Parity for SCAddr(11:6) and SCDCS* SCAPar0 - Even Parity for SCAddr(5:0) and SCTCS*		
SCData(127:0):	**Secondary cache data bus**	**Input/Output**
A 128-bit bus used to read or write cache data from and to the secondary cache data RAM.		
SCDChk(15:0):	**Secondary cache data ECC bus**	**Input/Output**
A 16-bit bus that carries two 8-bit ECC field covering the 128 bits of SCData from/to secondary cache. SCDChk(15:8) corresponds to SCData(127:64) and SCDChk(7:0) corresponds to SCData(63:0).		
SCDCS*:	**Secondary cache data chip select**	**Output**
Chip select enable signal for the secondary cache data RAM.		
SCOE*:	**Secondary cache output enable**	**Output**
Output enable for the secondary cache data and tag RAM.		
SCTag(24:0):	**Secondary cache tag bus**	**Input/Output**
A 25-bit bus used to read or write cache tags from and to the secondary cache.		
SCTChk(6:0):	**Secondary cache tag ECC bus**	**Input/Output**
A 7-bit bus that carries an Error Checking and Correcting (ECC) field covering the SCTag from and to the secondary cache.		
SCTCS*:	**Secondary cache tag chip select**	**Output**
Chip select enable signal for the secondary cache tag RAM.		
SCWrW*:	**Secondary cache write enable**	**Output**
SCWrX*:	**Secondary cache write enable**	**Output**
SCWrY*:	**Secondary cache write enable**	**Output**
SCWrZ*:	**Secondary cache write enable**	**Output**
Write enable for the secondary cache data and tag RAM		

Interrupt Interface

The interrupt interface signals comprise the interface used by external agents to interrupt the TC86R4400 processor. Int*(5:1) is available only on the TC86R4400PC; Int*(0) and NMI* are available on both configurations.

Int*(5:1):	**Interrupt**	**Input**
Five of six general processor interrupts, bit-wise ORed with bits 5:1 of the interrupt register. (R4000PC only)		
Int*(0):	**Interrupt**	**Input**
One of six general processor interrupts, bit-wise ORed with bit 0 of the interrupt register.		
NMI*:	**Non-maskable interrupt**	**Input**
Non-maskable interrupt, ORed with bit 6 of the interrupt register.		

Initialization Interface

The initialization interface signals comprise the interface by which an external agent initializes the TC86R4400 operating parameters. All of these signals are available on both processor configurations.

ColdReset*:	**Cold reset**	**Input**
This signal must be asserted for a power on reset or a cold reset. The clocks SClock, TClock, and RClock begin to cycle and are synchronized with the de-assertion edge of ColdReset*. ColdReset* must be de-asserted synchronously with MasterOut.		
ModeClock:	**Boot mode clock**	**Output**
Serial boot-mode data clock output at the system clock frequency divided by two hundred and fifty six.		
ModeIn:	**Boot mode data in**	**Input**
Serial boot-mode data input.		
Reset*:	**Reset**	**Input**
This signal must be asserted for any reset sequence. It may be asserted synchronously or asynchronously for a cold reset, or synchronously to initiate a warm reset. Reset* must be de-asserted synchronously with MasterOut.		
VCCOk:	**VCC is OK**	**Input**
When asserted, this signal indicates to the TC86R4400 that the +5 volt power supply has been above 4.75 volts for more than 100 milliseconds and will remain stable. The assertion of VCCOk initiates the initialization sequence.		

JTAG Interface

The JTAG interface signals comprise the interface by which the JTAG boundary scan mechanism is provided.

JTDI:	**JTAG data in**	**Input**
Data is serially scanned in through this pin.		
JTCK:	**JTAG clock input**	**Input**
The TC86R4400 outputs a serial clock on JTCK. On the rising edge of JTCK both JTDI and JTMS are sampled.		
JTDO:	**JTAG data out**	**Output**
Data is serially scanned out through this pin.		
JTMS:	**JTAG command**	**Input**
JTAG command signal, signals that the incoming serial data is command data.		

Signal Summary

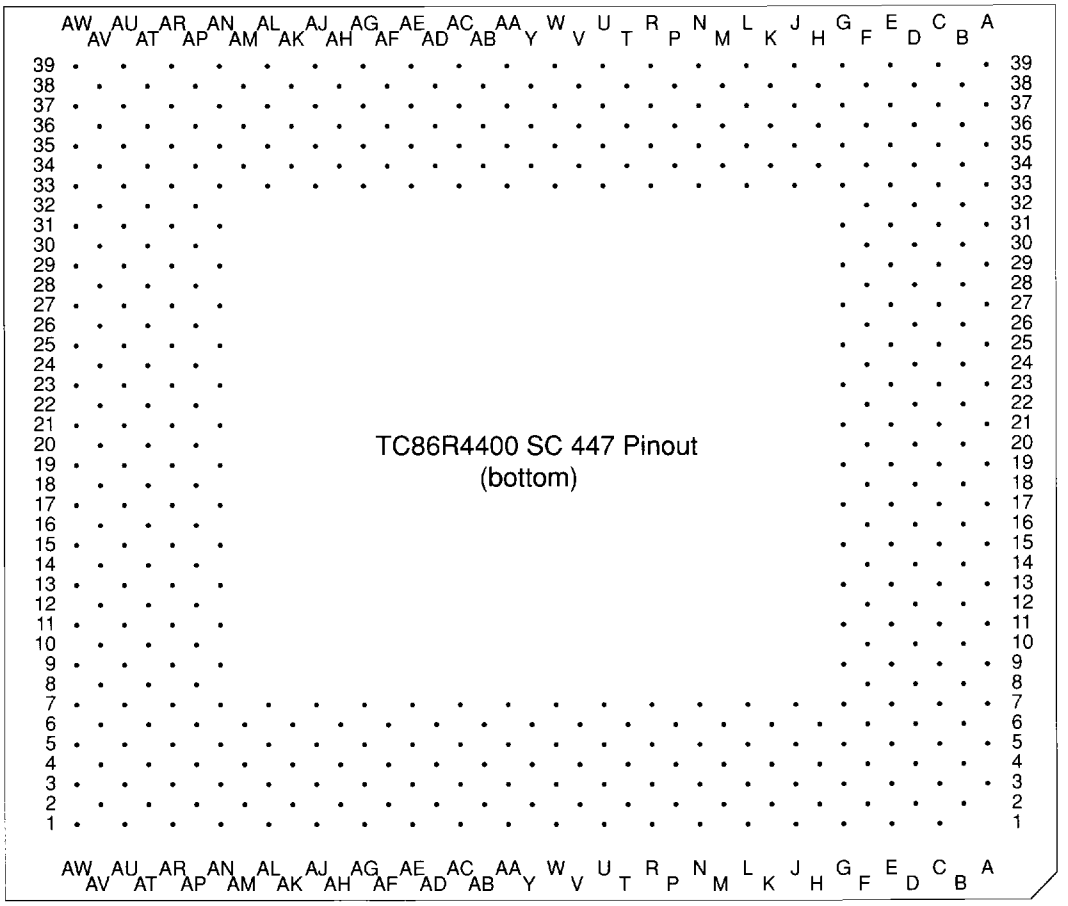
The tables following provide processor signal summaries for the TC86R4400SC.

TC86R4400SC Processor Signal Summary

Description	Name	I/O	Asserted State	3-State
Secondary cache data bus	SCData(127:0)	I/O	High	Yes
Secondary cache data ECC bus	SCDChk(15:0)	I/O	High	Yes
Secondary cache tag bus	SCTag(24:0)	I/O	High	Yes
Secondary cache tag ECCbus	SCTChk(6:0)	I/O	High	Yes
Secondary cache address bus	SCAddr(17:1)	O	High	No
Secondary cache address LSB	SCAddr0Z	O	High	No
Secondary cache address LSB	SCAddr0Y	O	High	No
Secondary cache address LSB	SCAddr0X	O	High	No
Secondary cache address LSB	SCAddr0W	O	High	No
Secondary cache address parity bus	SCAPar(2:0)	O	High	No
Secondary cache output enable	SCOE*	O	Low	No
Secondary cache write enable	SCWrZ*	O	Low	No
Secondary cache write enable	SCWrY*	O	Low	No
Secondary cache write enable	SCWrX*	O	Low	No
Secondary cache write enable	SCWrW*	O	Low	No
Secondary cache data chip select	SCDCS*	O	Low	No
Secondary cache tag chip select	SCTCS*	O	Low	No
System address/data bus	SysAD(63:0)	I/O	High	Yes
System address/data check bus	SysADC(7:0)	I/O	High	Yes
System command/data identifier bus	SysCmd(8:0)	I/O	High	Yes
System command/data identifier bus parity	SysCmdP	I/O	High	Yes
Valid input	ValidIn*	I	Low	No
Valid output	ValidOut*	O	Low	No
External request	ExtRqst*	I	Low	No
Release interface	Release*	O	Low	No
Read ready	RdRdy*	I	Low	No
Write ready	WrRdy*	I	Low	No
Invalidate acknowledge	IvdAck*	I	Low	No
Invalidate error	IvdErr*	I	Low	No
Interrupt	Int*(0)	I	Low	No
Non-maskable interrupt	NMI*	I	Low	No
Boot mode data in	ModeIn	I	High	No
Boot mode clock	ModeClock	O	High	No
JTAG data in	JTDI	I	High	No
JTAG data out	JTDO	O	High	No
JTAG command	JTMS	I	High	No
JTAG clock input	JTCK	I	High	No
Transmit clocks	TClock(1:0)	O	High	No
Receive clocks	RClock(1:0)	O	High	No
Master clock	MasterClock	I	High	No
Master clock out	MasterOut	O	High	No

TC86R4400SC Processor Signal Summary (Continued)

Description	Name	I/O	Asserted State	3-State
Synchronization clock out	SyncOut	O	High	No
Synchronization clock in	SyncIn	I	High	No
I/O output	IOOut	O	High	No
I/O input	IOIn	I	High	No
VCC is OK	VCCOk	I	High	No
Cold reset	ColdReset*	I	Low	No
Reset	Reset*	I	Low	No
Fault	Fault*	O	Low	No
Quiet VCC for PLL	VccP	I	High	No
Quiet VSS for PLL	VssP	I	High	No
Status	Status(7:0)	O	High	No
VCC sense	VccSense	I/O	N/A	No
VSS sense	VssSense	I/O	N/A	No



TC86R4400SC Package Pinout

*1) NoConnect should be completely unconnected.

R4000 Function	SC Pkg Pin	R4000 Function	SC Pkg Pin	R4000 Function	SC Pkg Pin
ColdReset*	AW37	ExtRqst*	AV2	Fault*	C39
IOIn	AV32	IOOut	AV28	Int0*	AL1
IvdAck*	AA35	IvdErr*	AA39	JTCK	U39
JTDI	N39	JTDO	J39	JTMS	G37
MasterClock	AA37	MasterOut	AJ39	ModeClock	B8
ModeIn	AV8	NMI*	AV16	NoConnect *1)	AV24
NoConnect	Y2	PLLCap0	****	PLLCap1	****
RClock0	AM34	RClock1	AL33	RdRdy*	AW7
Release*	AV12	Reset*	AU39	SCAPar0	U5
SCAPar1	U1	SCAPar2	P4	SCAddr1	AL5
SCAddr2	AG1	SCAddr3	AE7	SCAddr4	AC1
SCAddr5	AC5	SCAddr6	AC3	SCAddr7	AA1
SCAddr8	AB4	SCAddr9	AA5	SCAddr10	AA7
SCAddr11	AA3	SCAddr12	W3	SCAddr13	Y6
SCAddr14	W5	SCAddr15	W7	SCAddr16	W1
SCAddr17	U3	SCAddr0W	AN7	SCAddr0X	AN5
SCAddr0Y	AM6	SCAddr0Z	AL7	SCDCS*	M6
SCDChk0	G19	SCDChk1	T34	SCDChk2	AP20
SCDChk3	AD34	SCDChk4	C19	SCDChk5	R37
SCDChk6	AU19	SCDChk7	AE37	SCDChk8	C17
SCDChk9	N37	SCDChk10	AU17	SCDChk11	AG37
SCDChk12	E19	SCDChk13	R35	SCDChk14	AR19
SCDChk15	AE35	SCData0	R3	SCData1	R7
SCData2	L5	SCData3	F8	SCData4	C9
SCData5	F12	SCData6	G15	SCData7	E17
SCData8	G21	SCData9	C25	SCData10	G25
SCData11	E29	SCData12	G31	SCData13	C35
SCData14	K36	SCData15	N35	SCData16	AE3
SCData17	AG5	SCData18	AK4	SCData19	AN9
SCData20	AU9	SCData21	AN13	SCData22	AT14
SCData23	AR17	SCData24	AT22	SCData25	AU25
SCData26	AN27	SCData27	AR29	SCData28	AN31
SCData29	AR35	SCData30	AK36	SCData31	AG35
SCData32	T6	SCData33	L3	SCData34	L7
SCData35	E7	SCData36	G11	SCData37	E13
SCData38	E15	SCData39	G17	SCData40	C23
SCData41	F24	SCData42	E27	SCData43	D30
SCData44	C33	SCData45	E35	SCData46	L35
SCData47	R33	SCData48	AF4	SCData49	AJ3
SCData50	AJ7	SCData51	AP8	SCData52	AT10
SCData53	AR13	SCData54	AR15	SCData55	AT18
SCData56	AU23	SCData57	AT26	SCData58	AR27
SCData59	AN29	SCData60	AP32	SCData61	AN35
SCData62	AJ35	SCData63	AE33	SCData64	V4

TC86R4400SC Package Pinout (Continued)

*1) NoConnect should be completely unconnected.

R4000 Function	SC Pkg Pin	R4000 Function	SC Pkg Pin	R4000 Function	SC Pkg Pin
SCData65	R5	SCData66	N5	SCData67	E5
SCData68	G9	SCData69	E11	SCData70	G13
SCData71	D14	SCData72	C21	SCData73	D22
SCData74	E25	SCData75	G27	SCData76	C31
SCData77	F32	SCData78	J35	SCData79	M34
SCData80	AC7	SCData81	AE5	SCData82	AG7
SCData83	AR5	SCData84	AR9	SCData85	AR11
SCData86	AN15	SCData87	AP16	SCData88	AU21
SCData89	AN23	SCData90	AR25	SCData91	AP28
SCData92	AU31	SCData93	AR33	SCData94	AL35
SCData95	AH34	SCData96	U7	SCData97	N3
SCData98	N7	SCData99	C5	SCData100	E9
SCData101	C11	SCData102	C13	SCData103	F16
SCData104	E21	SCData105	G23	SCData106	C27
SCData107	F28	SCData108	E31	SCData109	G33
SCData110	J37	SCData111	N33	SCData112	AD6
SCData113	AG3	SCData114	AJ5	SCData115	AU5
SCData116	AN11	SCData117	AU11	SCData118	AU13
SCData119	AN17	SCData120	AR21	SCData121	AP24
SCData122	AU27	SCData123	AT30	SCData124	AU33
SCData125	AN33	SCData126	AL37	SCData127	AG33
SCOE*	N1	SCTCS*	J1	SCTChk0	AN21
SCTChk1	AN19	SCTChk2	AU15	SCTChk3	AP12
SCTChk4	AU7	SCTChk5	AR7	SCTChk6	AH6
SCTag0	K4	SCTag1	G7	SCTag2	C7
SCTag3	D10	SCTag4	C15	SCTag5	D18
SCTag6	F20	SCTag7	E23	SCTag8	D26
SCTag9	C29	SCTag10	G29	SCTag11	E33
SCTag12	G35	SCTag13	L33	SCTag14	L37
SCTag15	P36	SCTag16	AF36	SCTag17	AJ37
SCTag18	AJ33	SCTag19	AN37	SCTag20	AU35
SCTag21	AR31	SCTag22	AU29	SCTag23	AN25
SCTag24	AR23	SCWrW*	J5	SCWrX*	J7
SCWrY*	H6	SCWrZ*	G5	Status0	U33
Status1	U35	Status2	V36	Status3	W35
Status4	W37	Status5	AC37	Status6	AC35
Status7	AC33	SyncIn	W39	SyncOut	AN39
SysAD0	T2	SysAD1	M2	SysAD2	J3
SysAD3	G3	SysAD4	C1	SysAD5	A3
SysAD6	A9	SysAD7	A13	SysAD8	A21
SysAD9	A25	SysAD10	A29	SysAD11	A33
SysAD12	B38	SysAD13	E37	SysAD14	G39
SysAD15	L39	SysAD16	AD2	SysAD17	AH2
SysAD18	AL3	SysAD19	AN3	SysAD20	AU1

TC86R4400SC Package Pinout (Continued)

*1) NoConnect should be completely unconnected.

R4000 Function	SC Pkg Pin	R4000 Function	SC Pkg Pin	R4000 Function	SC Pkg Pin
SysAD21	AW3	SysAD22	AW9	SysAD23	AW13
SysAD24	AW21	SysAD25	AW25	SysAD26	AW29
SysAD27	AW33	SysAD28	AV38	SysAD29	AR37
SysAD30	AM38	SysAD31	AH38	SysAD32	R1
SysAD33	L1	SysAD34	H2	SysAD35	E1
SysAD36	C3	SysAD37	A5	SysAD38	A11
SysAD39	A15	SysAD40	A23	SysAD41	A27
SysAD42	A31	SysAD43	A35	SysAD44	C37
SysAD45	E39	SysAD46	H38	SysAD47	M38
SysAD48	AE1	SysAD49	AJ1	SysAD50	AM2
SysAD51	AR1	SysAD52	AU3	SysAD53	AW5
SysAD54	AW11	SysAD55	AW15	SysAD56	AW23
SysAD57	AW27	SysAD58	AW31	SysAD59	AW35
SysAD60	AU37	SysAD61	AR39	SysAD62	AL39
SysAD63	AG39	SysADC0	A17	SysADC1	R39
SysADC2	AW17	SysADC3	AD38	SysADC4	A19
SysADC5	T38	SysADC6	AW19	SysADC7	AC39
SysCmd0	G1	SysCmd1	E3	SysCmd2	B2
SysCmd3	B12	SysCmd4	B16	SysCmd5	B20
SysCmd6	B24	SysCmd7	B28	SysCmd8	B32
SysCmdP	A37	TClock0	H34	TClock1	J33
VCCOk	AE39	ValidIn*	AN1	ValidOut*	AR3
WrRdy*	A7	VccSense	W33	VssSense	U37
VccP	AA33	VssP	Y34	Vcc	A39
Vcc	B6	Vcc	B10	Vcc	B18
Vcc	B26	Vcc	B34	Vcc	D4
Vcc	D8	Vcc	D16	Vcc	D24
Vcc	D32	Vcc	D36	Vcc	F2
Vcc	F14	Vcc	F22	Vcc	F30
Vcc	F38	Vcc	H4	Vcc	H36
Vcc	K6	Vcc	K38	Vcc	P2
Vcc	P34	Vcc	T4	Vcc	T36
Vcc	V6	Vcc	V38	Vcc	Y38
Vcc	AB2	Vcc	AB34	Vcc	AD4
Vcc	AD36	Vcc	AF6	Vcc	AF38
Vcc	AK2	Vcc	AK34	Vcc	AM4
Vcc	AM36	Vcc	AP2	Vcc	AP10
Vcc	AP18	Vcc	AP26	Vcc	AP38
Vcc	AT4	Vcc	AT8	Vcc	AT16
Vcc	AT24	Vcc	AT32	Vcc	AT36
Vcc	AV6	Vcc	AV14	Vcc	AV20
Vcc	AV22	Vcc	AV30	Vcc	AV34
Vcc	AW1	Vcc	AW39	Vss	B4
Vss	B14	Vss	B22	Vss	B30

TC86R4400SC Package Pinout (Continued)

*1) NoConnect should be completely unconnected.

R4000 Function	SC Pkg Pin	R4000 Function	SC Pkg Pin	R4000 Function	SC Pkg Pin
Vss	B36	Vss	D2	Vss	D6
Vss	D12	Vss	D20	Vss	D28
Vss	D34	Vss	D38	Vss	F4
Vss	F6	Vss	F10	Vss	F18
Vss	F26	Vss	F34	Vss	F36
Vss	K2	Vss	K34	Vss	M4
Vss	M36	Vss	P6	Vss	P38
Vss	V2	Vss	V34	Vss	Y4
Vss	Y36	Vss	AB6	Vss	AB36
Vss	AB38	Vss	AF2	Vss	AF34
Vss	AH4	Vss	AH36	Vss	AK6
Vss	AK38	Vss	AP4	Vss	AP6
Vss	AP14	Vss	AP22	Vss	AP30
Vss	AP34	Vss	AP36	Vss	AT2
Vss	AT6	Vss	AT12	Vss	AT20
Vss	AT28	Vss	AT34	Vss	AT38
Vss	AV4	Vss	AV10	Vss	AV18
Vss	AV26	Vss	AV36		

Specifications

Specifications for the TC86R4400-150MHz and -200MHz follow.

Electrical Characteristics

Table 1. Maximum Ratings

TC86R4400SC/MC-150, V_{SS} = 0V (GND)

PARAMETER	SYMBOL	MINIMUM	UNITS
Supply Voltage	V_{CC}	−0.5 ~ +4.0	V
Input Voltage	V_{IN}	−0.5 ~ V_{CC} +0.3	
Storage Temperature	T_{ST}	−65 ~ +125	°C

Table 2. Maximum Ratings

TC86R4400SC/MC-200, V_{SS} = 0V (GND)

PARAMETER	SYMBOL	MINIMUM	UNITS
Supply Voltage	V_{CC}	−0.5 ~ TBD	V
Input Voltage	V_{IN}	−0.5 ~ TBD	
Storage Temperature	T_{ST}	−65 ~ +125	°C

Note: If LSI is used above the maximum ratings, permanent destruction of LSI can result. In addition, it is desireable to use LSI for normal operation under the recommended condition. If these conditions are exceeded, reliability of LSI may be adversely affected.

Table 3. Operating Parameters

TC86R4400SC/MC-150, V_{SS} = 0V (GND)

PARAMETER	SYMBOL	CONDITIONS	Min.	Max.	UNITS
Operating Temperature	T_C	Case Temperature	0	70	°C
Supply Voltage	V_{CC}		3.3	3.6	V
Clock Frequency	f_C		40	75	MHz

Table 4. Operating Parameters

TC86R4400SC/MC-200, V_{SS} = 0V (GND)

PARAMETER	SYMBOL	CONDITIONS	Min.	Max.	UNITS
Operating Temperature	T_C	Case Temperature	0	70	°C
Supply Voltage	V_{CC}		3.28	3.62	V
Clock Frequency	f_C		40	100	MHz

Table 5. DC Characteristics

TC86R4400SC/MC-150

T_C = 0°C~70°C, V_{CC} = 3.3V ± 0.15V

PARAMETER		SYMBOL	CONDITIONS	Min.	Max.	UNITS
Input High Voltage	(*1)	V_{IH}		2.0	V_{CC} + 0.3	V
	(*2)	V_{IHC}		0.8 V_{CC}	V_{CC} + 0.3	
Input Low Voltage	(*1)	V_{IL}		-0.5	0.8	V
	(*2)	V_{ILC}		-0.5	0.2 V_{CC}	
Output High Voltage	(*4)	V_{OH}	I_{OH} =-4mA	2.4		V
	(*3)	V_{OHC}		2.7		
Output Low Voltage	(*4)	V_{OL}	I_{OL} =4mA		0.4	V
	(*3)	V_{OLC}			0.4	
Input Leakage		I_{LI}		-10	10	µA
Input/Output Leakage		I_{LIO}		-20	20	µA
Operating Current		I_{CC}	V_{CC} = 3.45V f_C = 150MHz		3	A
Input Capacitance		C_{IN}			10	pF

Table 6. DC Characteristics

TC86R4400SC/MC-200

T_C = 0°C~70°C, V_{CC} = 3.45V ± 5%

PARAMETER		SYMBOL	CONDITIONS	Min.	Max.	UNITS
Input High Voltage	(*1)	V_{IH}		2.0	V_{CC} + 0.3	V
	(*2)	V_{IHC}		0.8 V_{CC}	V_{CC} + 0.3	
Input Low Voltage	(*1)	V_{IL}		-0.5	0.8	V
	(*2)	V_{ILC}		-0.5	0.2 V_{CC}	
Output High Voltage	(*4)	V_{OH}	I_{OH} =-4mA	2.4		V
	(*3)	V_{OHC}		2.7		
Output Low Voltage	(*4)	V_{OL}	I_{OL} =4mA		0.4	V
	(*3)	V_{OLC}			0.4	
Input Leakage		I_{LI}		-10	10	µA
Input/Output Leakage		I_{LIO}		-20	20	µA
Operating Current		I_{CC}	V_{CC} = 3.45V f_C = 200MHz		TBD	A
Input Capacitance		C_{IN}			10	pF

(*1) *Except MasterClock input

(*2) Applies to MasterClock

(*3) Applies to TClock, RClock, MasterOut and ModeClock output

(*4) Applies to signals except those listed in (*3).

AC Characteristics

Table 7. Clock Timing

TC86R4400SC/MC-150

T_C = 0°C~70°C, V_{CC} = 3.3V ± 0.15V

PARAMETER	SYMBOL	CONDITIONS	Min.	Max.	UNITS
MasterClock High	t_{MCH}	Transition <3.5ns	3		ns
MasterClock Low	t_{MCL}	Transition <3.5ns	3		ns
MasterClock Frequency (*1)	f_{MCK}		40	75	MHz
MasterClock Period	t_{MCP}		13.3	25	ns
Clock Jitter (*2)	t_{MCJ}			±500	ps
MasterClock Rise Time	t_{MCR}			3.5	ns
MasterClock Fall Time	t_{MCF}			3.5	ns

Table 8. Clock Timing

TC86R4400SC/MC-200

T_C = 0°C~70°C, V_{CC} = 3.45V ± 5%

PARAMETER	SYMBOL	CONDITIONS	Min.	Max.	UNITS
MasterClock High	t_{MCH}	Transition <2.5ns	2.5		ns
MasterClock Low	t_{MCL}	Transition <2.5ns	2.5		ns
MasterClock Frequency (*1)	f_{MCK}		25	100	MHz
MasterClock Period	t_{MCP}		10	40	ns
Clock Jitter (*2)	t_{MCJ}			±500	ps
MasterClock Rise Time	t_{MCR}			2.5	ns
MasterClock Fall Time	t_{MCF}			2.5	ns

(*1) Operation of the R4400 is only guaranteed with the Phase Lock Loop enabled

(*2) Refer to Clock Timing section

Table 9. System Interface

TC86R4400SC/MC-150

T_C = 0°C~70°C, V_{CC} = 3.3V ± 0.15V

PARAMETER	SYMBOL	CONDITIONS	Min.	Max.	UNITS
Data Output	t_{DO}	Maximum Slew Rate Modebits [53:56] = 0 Modebits [57:60] = 15	2	7	ns
		Middle Slew Rate Modebits [53:56] = 8 Modebits [57:60] = 8	TBD	TBD	ns
		Minimum Slew Rate Modebits [53:56] = 15 Modebits [57:60] = 0	5	10	ns
Data Setup	t_{DS}		3.5		ns
Data Hold	t_{DH}		1		ns

Table 10. System Interface

TC86R4400SC/MC-200

T_C = 0°C~70°C, V_{CC} = 3.45V ± 5%

PARAMETER	SYMBOL	CONDITIONS	Min.	Max.	UNITS
Data Output	t_{DO}	Maximum Slew Rate Modebits [53:56] = 0 Modebits [57:60] = 15	1	5	ns
		Middle Slew Rate Modebits [53:56] = 8 Modebits [57:60] = 8	TBD	TBD	ns
		Minimum Slew Rate Modebits [53:56] = 15 Modebits [57:60] = 0	5	10	ns
Data Setup	t_{DS}		3.5		ns
Data Hold	t_{DH}		1		ns

Note: Data Output, Data Setup and Data Hold apply to all logic signals driven out of or driven into the R4400 on the system interface. Secondary cache signals are specified in the Secondary Cache Interface

Table 11. Secondary Cache Interface

TC86R4400SC/MC-150

T_C = 0°C~70°C, V_{CC} = 3.3V ± 0.15V

PARAMETER	SYMBOL	CONDITIONS	Min.	Max.	UNITS
PClock to Output	t_{SCO}	Maximum Slew Rate Modebits [53:56] = 0 Modebits [57:60] = 15	2	7	ns
		Middle Slew Rate Modebits [53:56] = 8 Modebits [57:60] = 8	TBD	TBD	ns
		Minimum Slew Rate Modebits [53:56] = 15 Modebits [57:60] = 0	6	12	ns
Data Setup	t_{SCDS}		3.5		ns
Data Hold	t_{SCDH}		1		ns

Table 12. Secondary Cache Interface

TC86R4400SC/MC-200

T_C = 0°C~70°C, V_{CC} = 3.45V ± 5%

PARAMETER	SYMBOL	CONDITIONS	Min.	Max.	UNITS
PClock to Output	t_{SCO}	Maximum Slew Rate Modebits [53:56] = 0 Modebits [57:60] = 15	1	5	ns
		Middle Slew Rate Modebits [53:56] = 8 Modebits [57:60] = 8	TBD	TBD	ns
		Minimum Slew Rate Modebits [53:56] = 15 Modebits [57:60] = 0	5	10	ns
Data Setup	t_{SCDS}		3.5		ns
Data Hold	t_{SCDH}		1		ns

1.3 Timing Diagrams

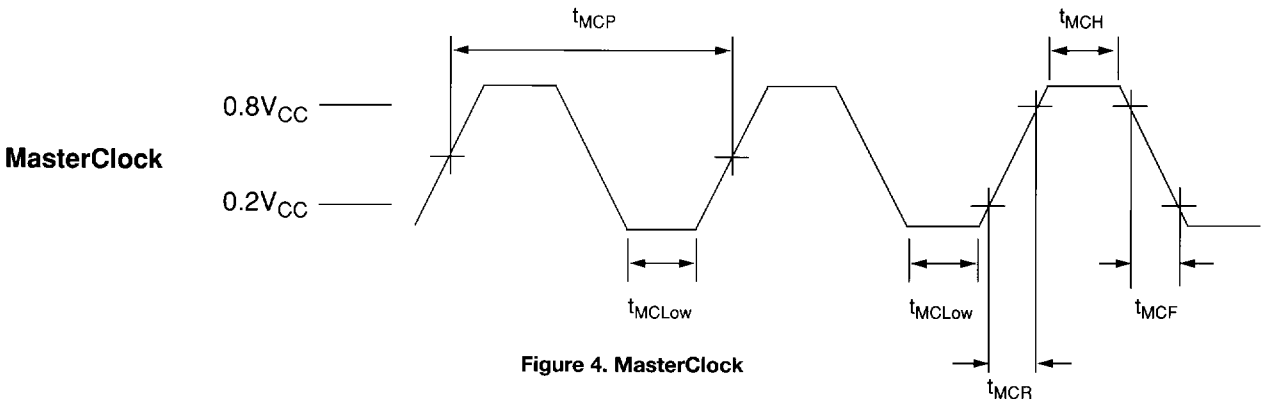


Figure 4. MasterClock

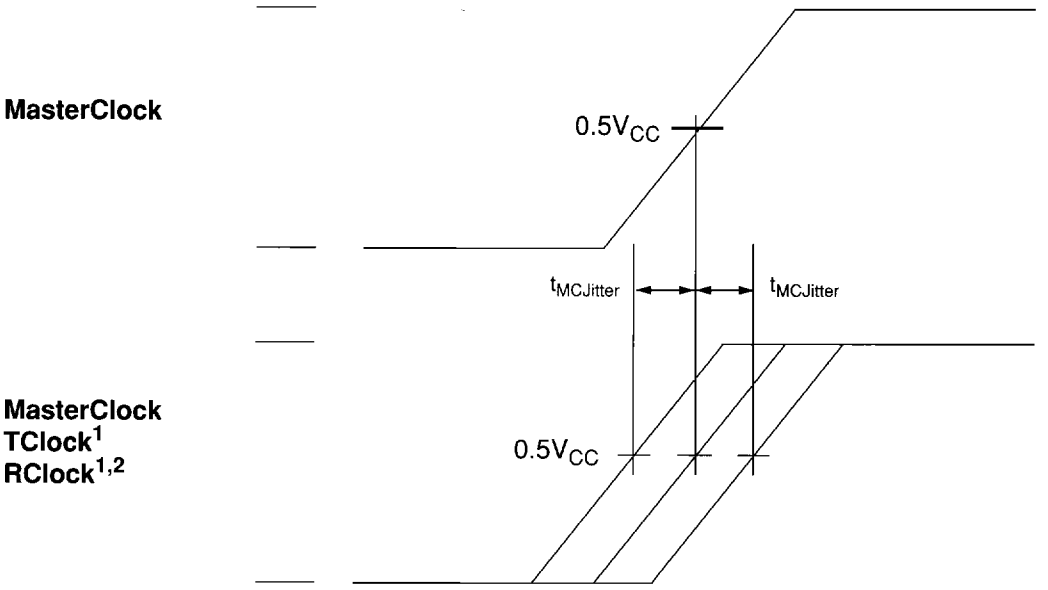


Figure 5. Clock Jitter

Note: (1) When SyncOut shorted to SyncIn by the shortest path, the 50% point of TClock lines up with 50% point of MasterClock.
(2) RClock leads TClock by 90°.
(3) The SyncIn/SyncOut path, TClock and RClock must have the same capacitive loading to match the MasterClock edges.

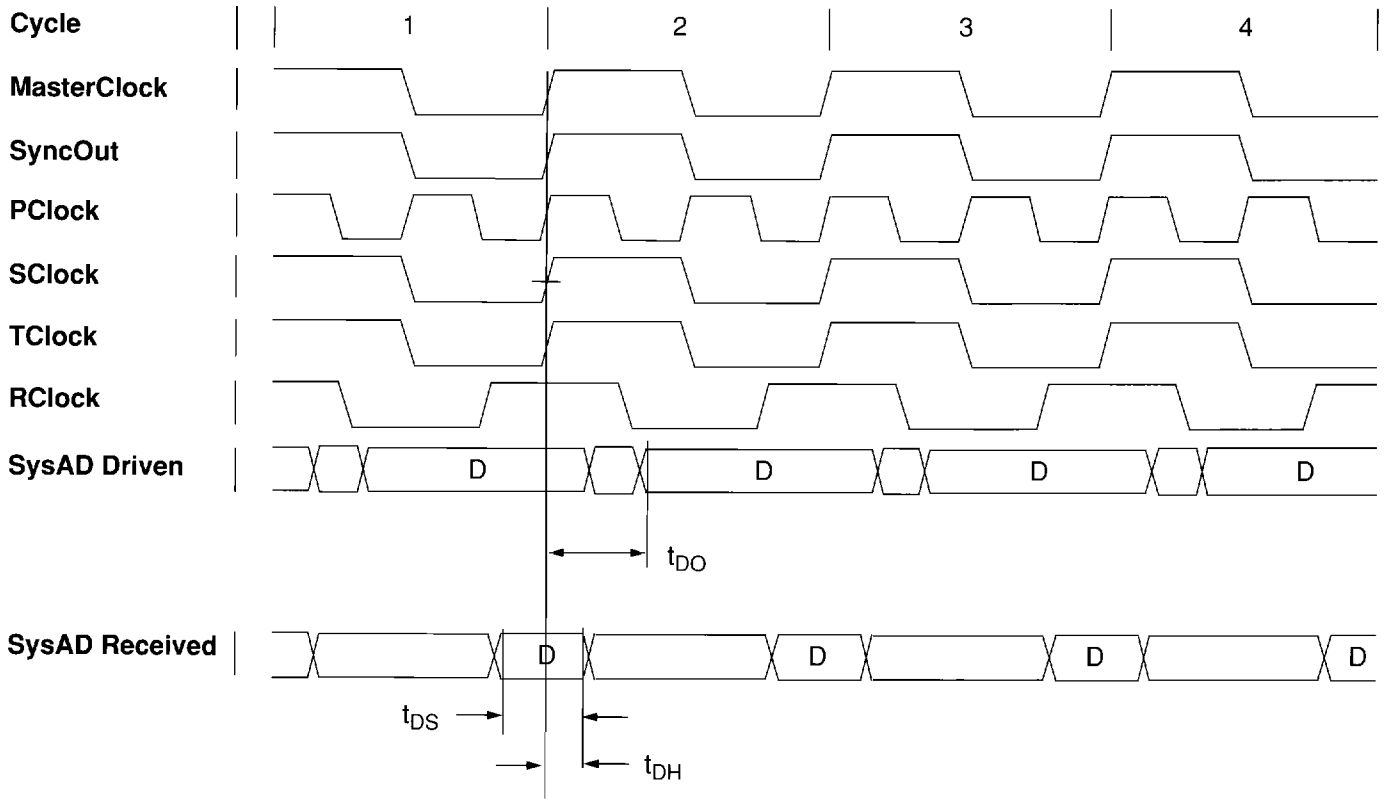


Figure 6. Processor Clock, PClock to SClock Divisor of 2

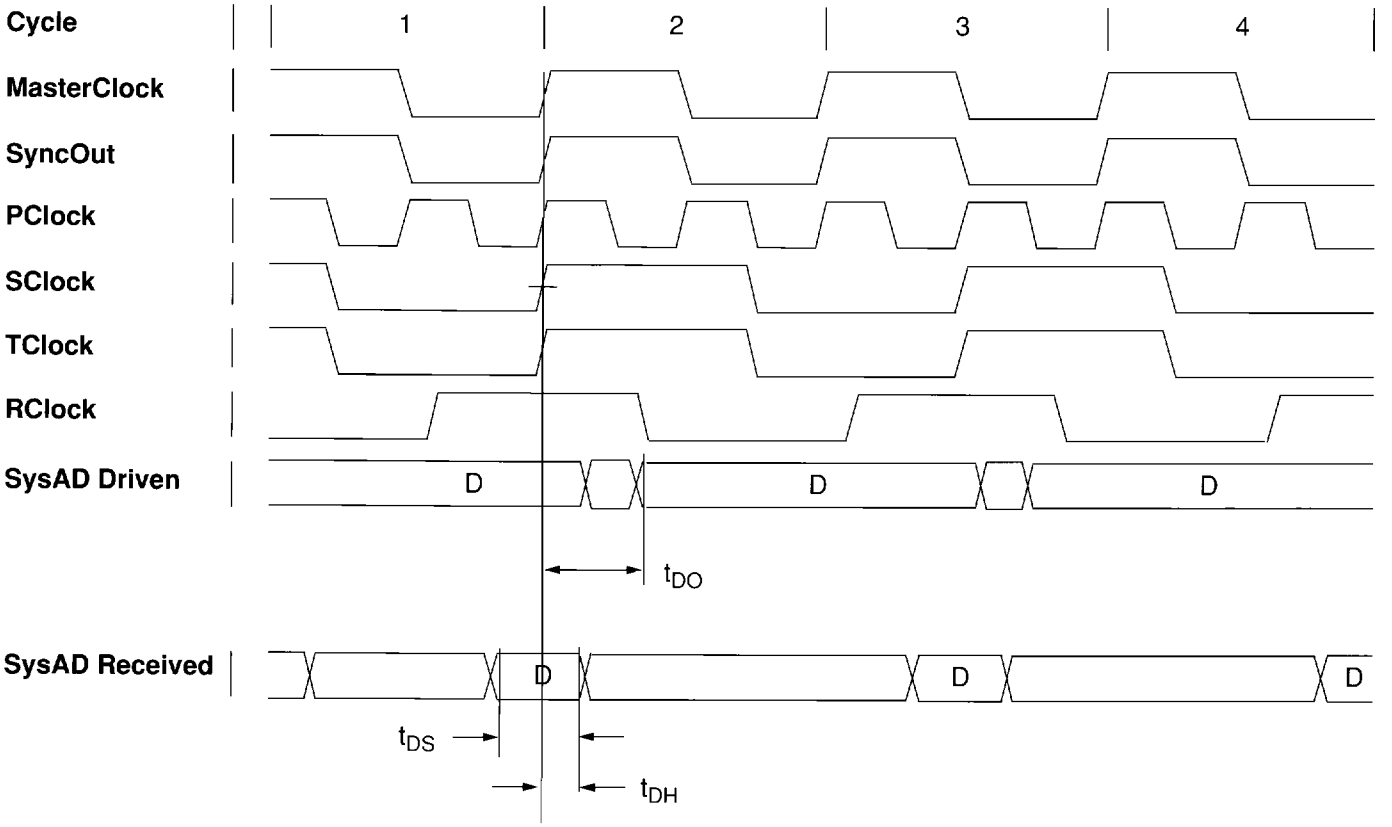


Figure 7. Processor Clock, PClock to SClock Divisor of 3

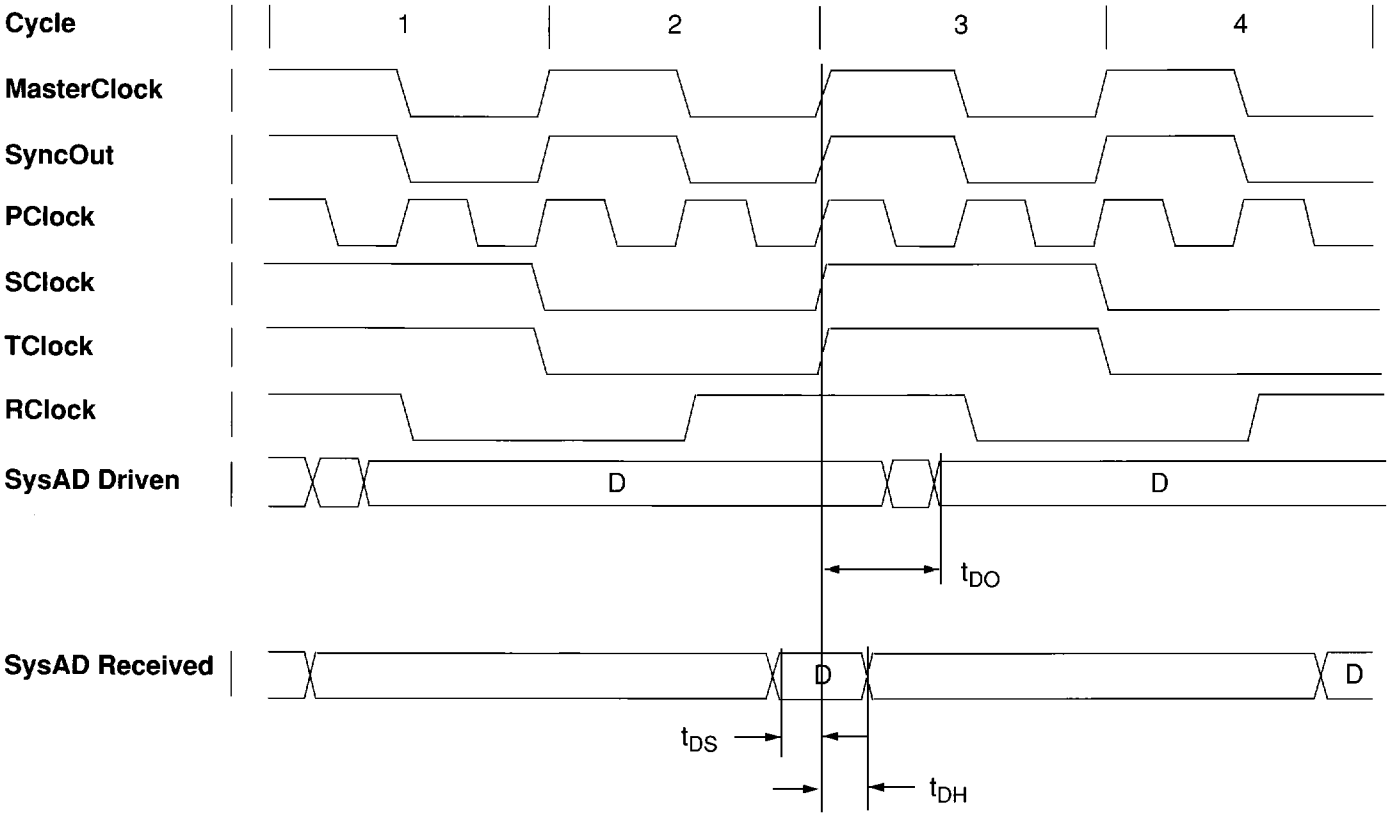
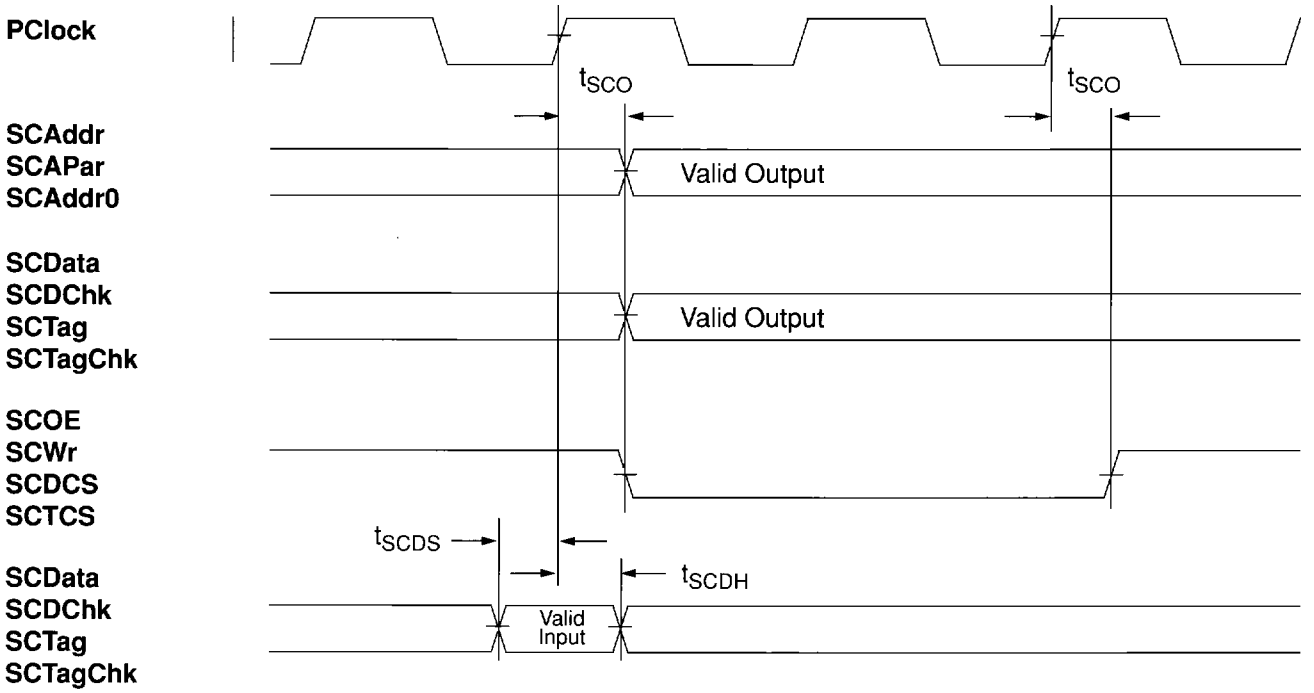
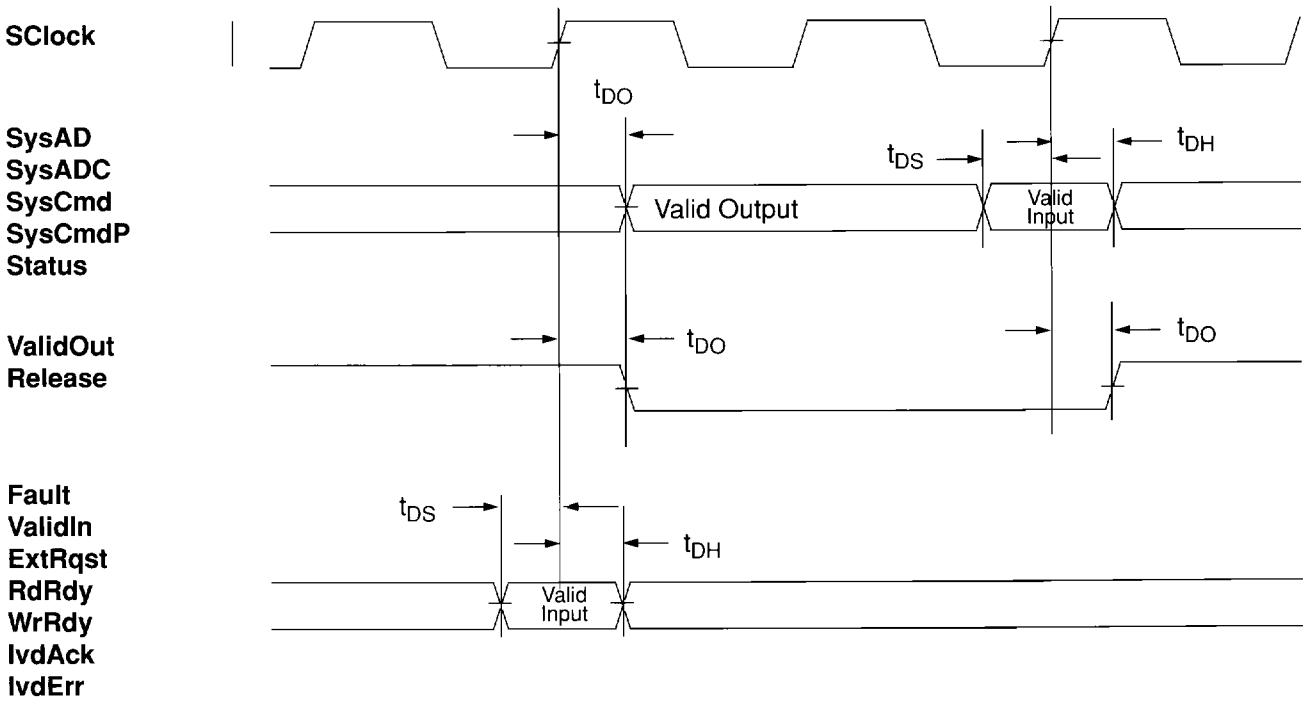


Figure 8. Processor Clock, PClock to SClock Divisor of 4



Note: (1) These waveforms only describe edge to edge timing relationships.
Functional descriptions are contained in previous chapters.

Figure 9. Secondary Cache Edge Timing Relationships



Note: (1) These waveforms only describe edge to edge timing relationships. Functional descriptions are contained in previous chapters.

Figure 10. System Interface Edge Timing Relationship

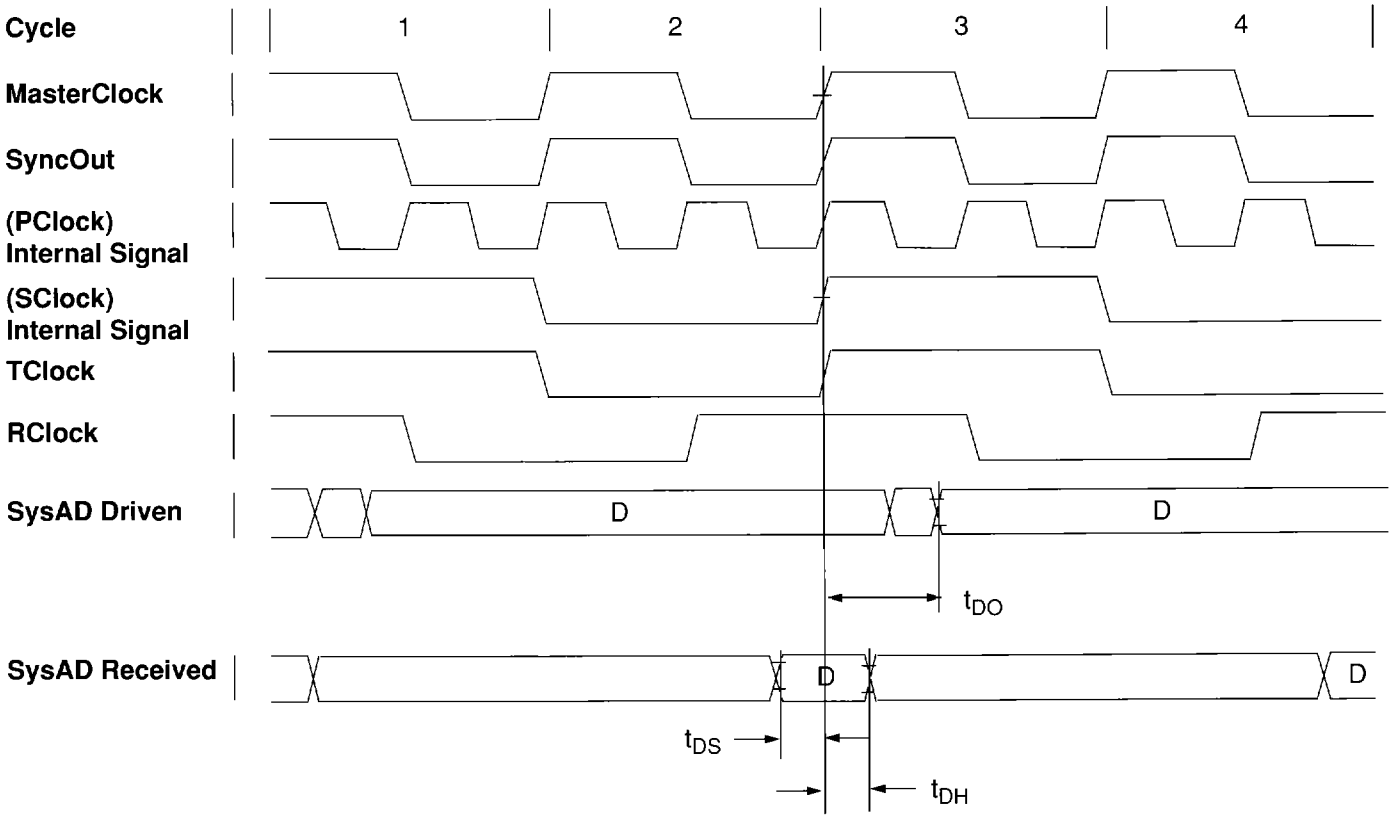


Figure 11. PClock to SClock Divisor of 4

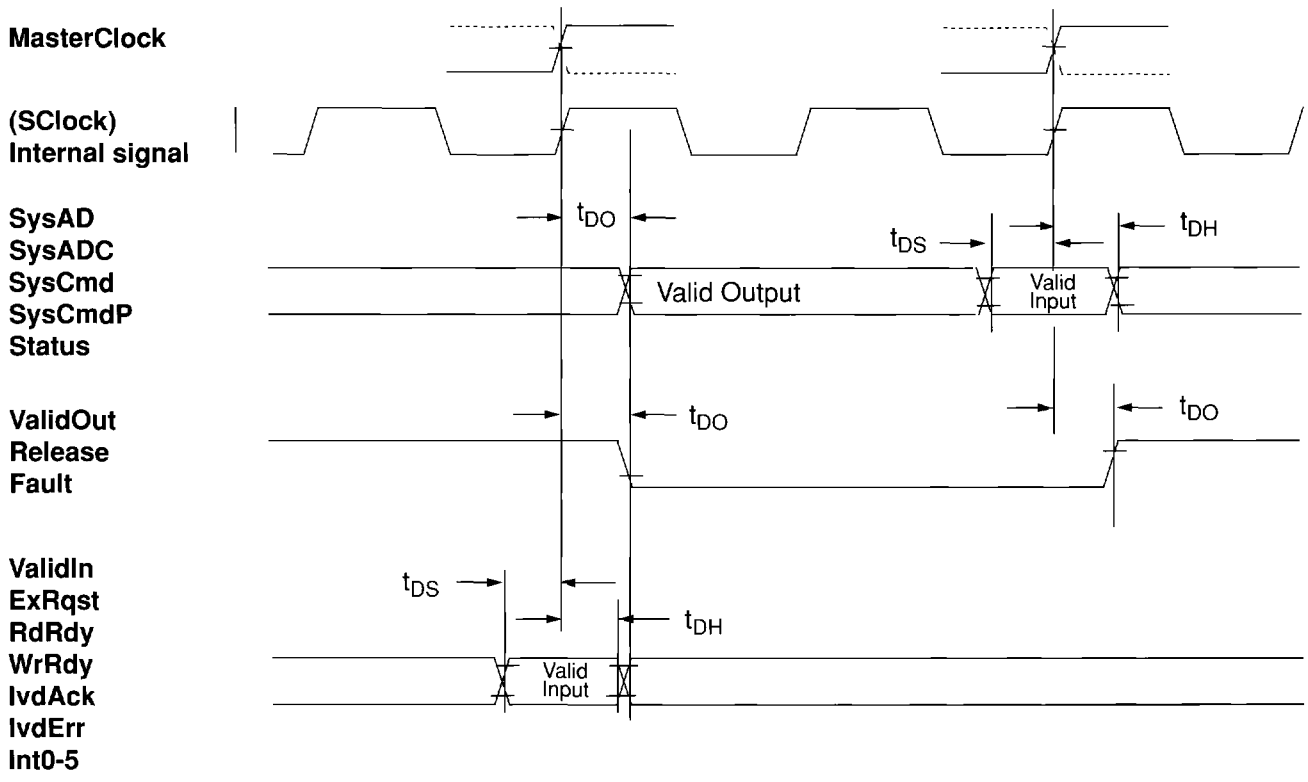


Figure 12. System Interface Timing

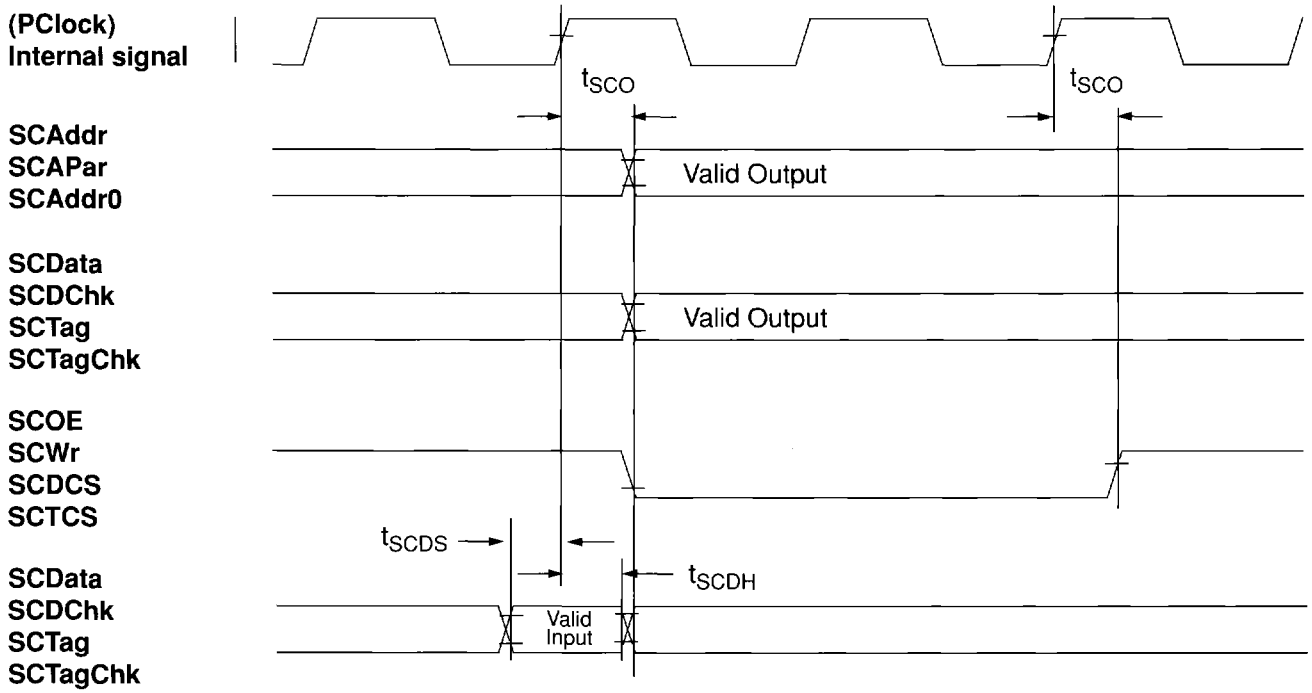
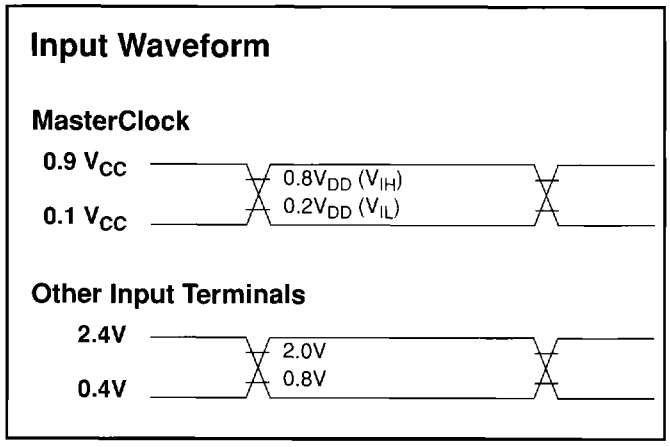
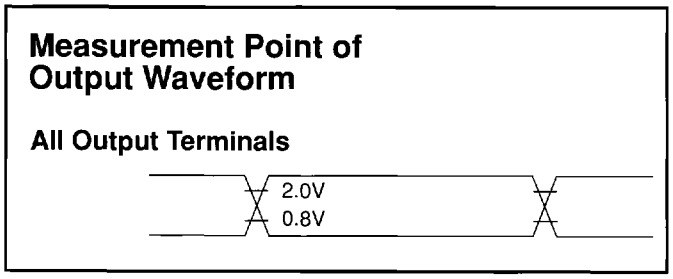


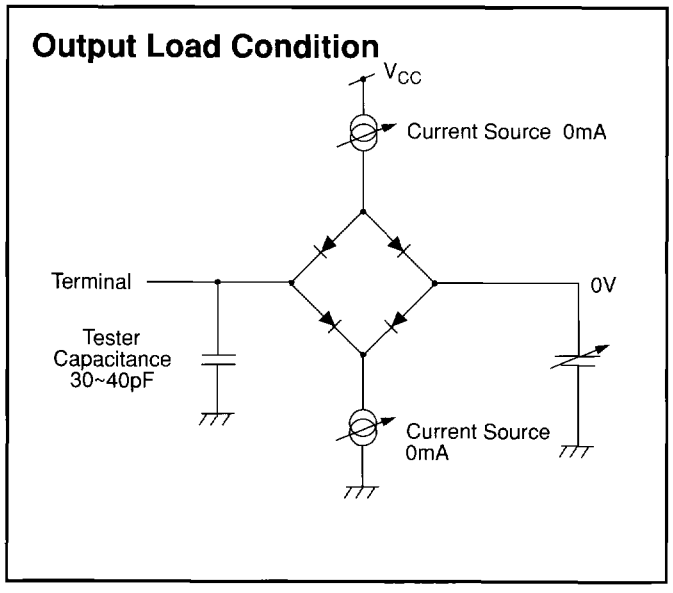
Figure 13. Secondary Cache Timing

TC85R4400

AC Testing Terminal Load Condition







Toshiba TC86R4400 series have products listed in the table below.

Table 13. TC86R4400 series (75MHz version) mark design and packaging type

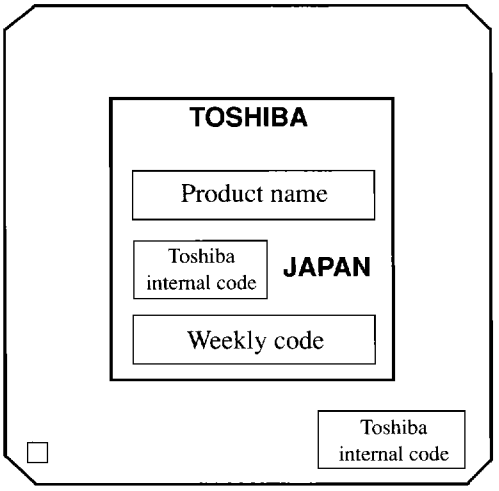
Marking Information		Product Type Number	Description
Product Name	R4400 revision code*		
TC86R4400PC-75	G : Rev 2.1	TC86R4400Y-P7G	MTI R4400PC Rev.2.1 75MHz PGA179
TC86R4400SC-75	G : Rev 2.1	TC86R4400Y-S7G	MTI R4400SC Rev.2.1 75MHz PGA447 (w/o stud)
TC86R4400SC-75	G : Rev 2.1	TC86R4400YS-S7G	MTI R4400SC Rev.2.1 75MHz PGA447 (w/ stud)
TC86R4400MC-75	G : Rev 2.1	TC86R4400Y-M7G	MTI R4400MC Rev.2.1 75MHz PGA447 (w/o stud)
TC86R4400MC-75	G : Rev 2.1	TC86R4400YS-M7G	MTI R4400MC Rev.2.1 75MHz PGA447 (w/ stud)

Table 14. TC86R4400 series (100MHz version) mark design and packaging type

Marking Information		Product Type Number	Description
Product Name	R4400 revision code*		
TC86R4400PC-100	H : Rev 3.0	TC86R4400Y-PAH	MTI R4400PC Rev.3.0 100MHz PGA179
TC86R4400SC-100	H : Rev 3.0	TC86R4400Y-SAH	MTI R4400SC Rev.3.0 100MHz PGA447 (w/o stud)
TC86R4400SC-100	H : Rev 3.0	TC86R4400YS-SAH	MTI R4400SC Rev.3.0 100MHz PGA447 (w/ stud)
TC86R4400MC-100	H : Rev 3.0	TC86R4400Y-MAH	MTI R4400MC Rev.3.0 100MHz PGA447 (w/o stud)
TC86R4400MC-100	H : Rev 3.0	TC86R4400YS-MAH	MTI R4400MC Rev.3.0 100MHz PGA447 (w/ stud)

*1) See "TC86R4400 Series Marking Information" in the next page
*2) Refer to product as product type numbers on issue of purchase order

TC86R4400 Series Marking Information



Product name : General format : TC86R4400XX-YY
XX : Model code specified by PC, SC, or MC
YY : Speed Version (MHz)

(Example) TC86R4400PC-100
PC 100MHz

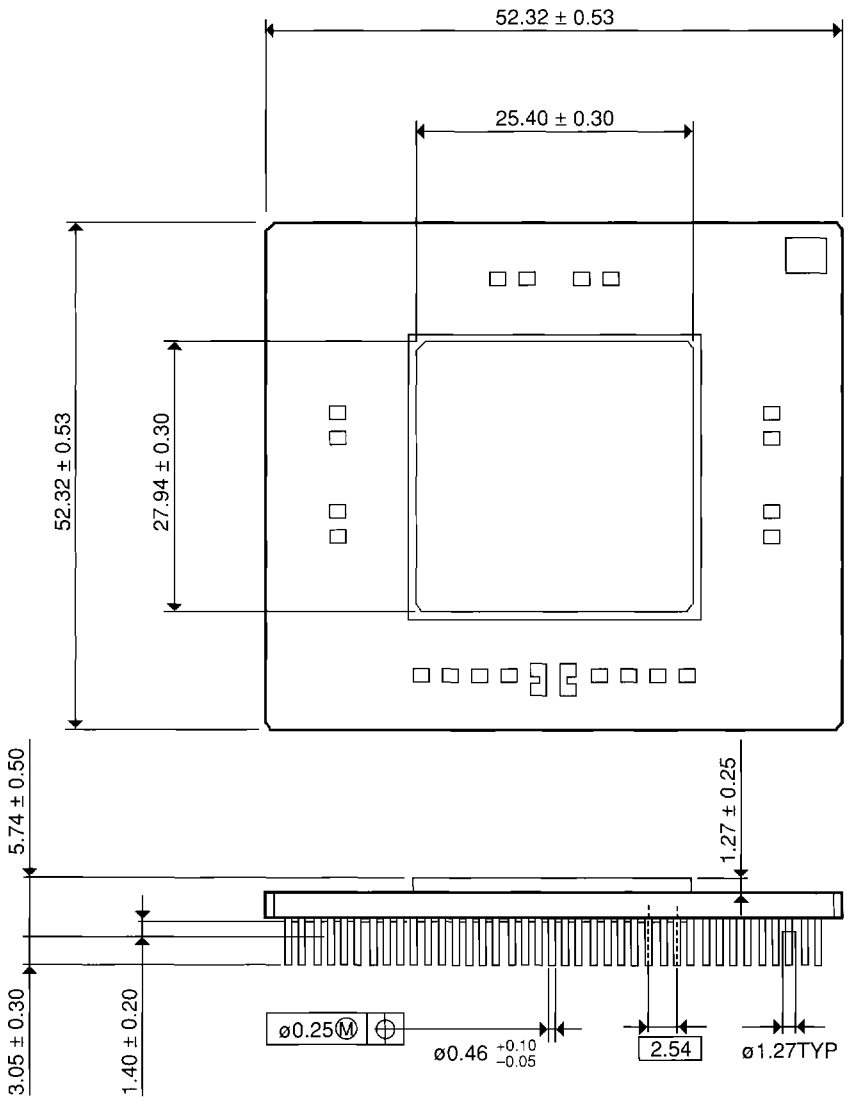
Toshiba internal code: General format (Option) : (This is a Toshiba internal code and may be omitted in some cases)

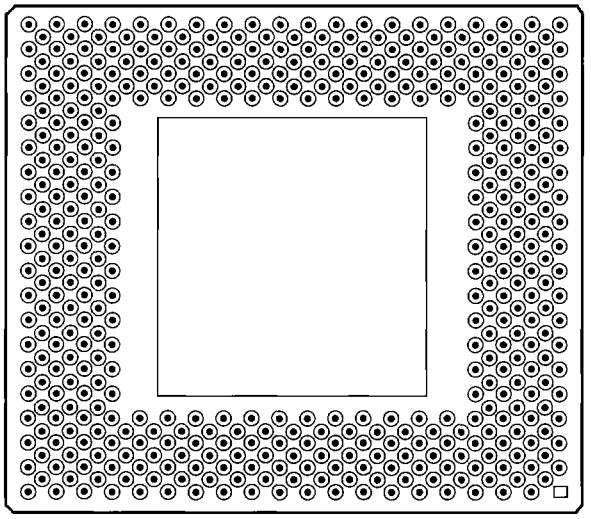
(Example) STUD : with stud package

Weekly code: General format : YYWWABCXXX
YY : Year (2 digits)
WW : Week (2 digits)
B : R4400 revision code
A, C : Toshiba internal code
XXX : Engineering Sample notation (option)

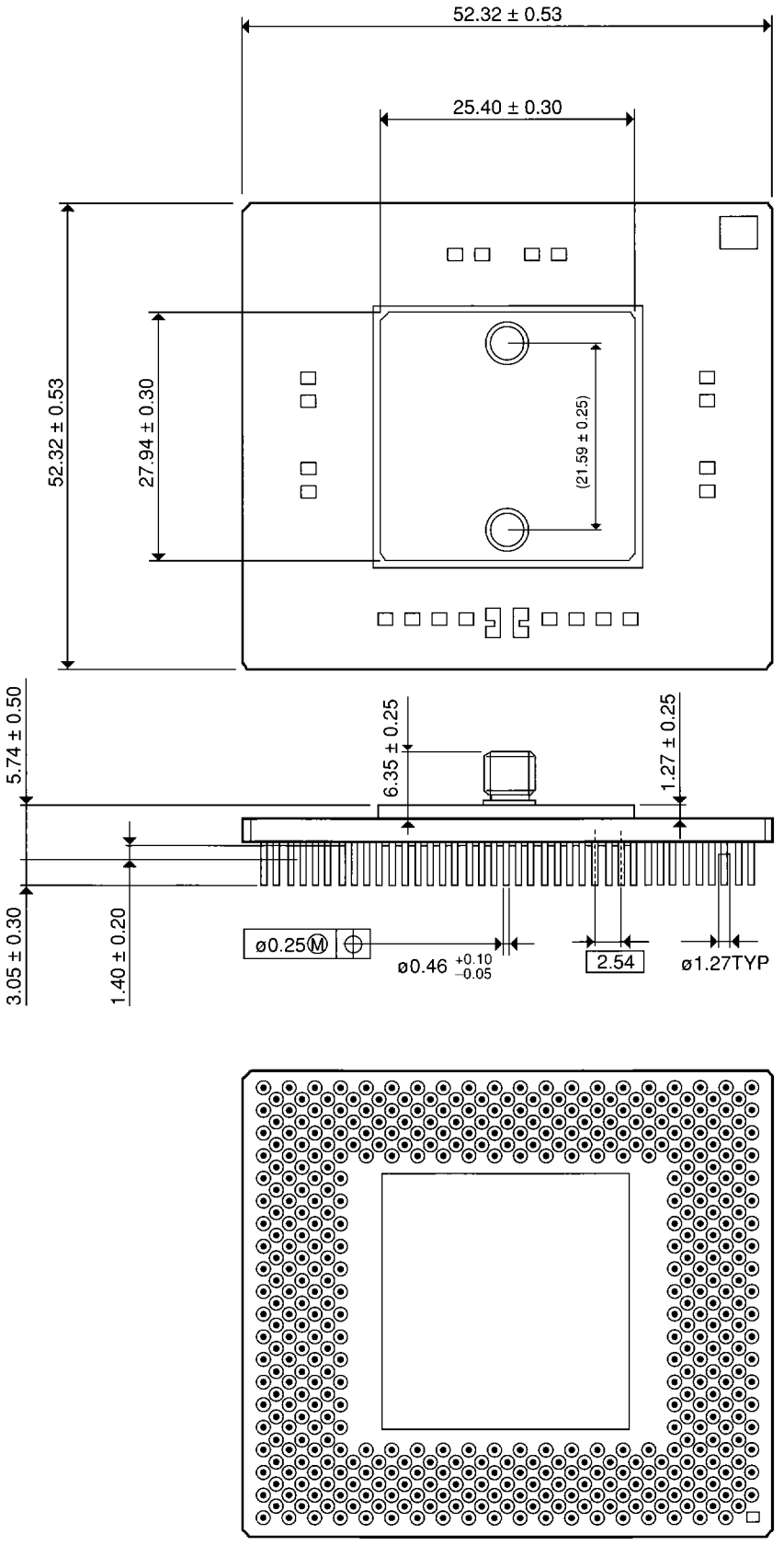
Example 9204YHZ-ES

HPGA447-C-S20D(F)





HPGA447-C-S20D(E)



Application Note

Following is a quick description of the features of the R4400.

1. In addition to divide by 2, 3 and 4, divide by 6 and 8 have been added. TClk for div6 and 8 has its rising edge aligned in the same place as during all other divide by modes. The waveforms for the divide by 2, 3 and 4 clocks are found in the interface spec. The new AC and DC specs will also contain this information. The modebits to set the interface are as follows:

Mode bits 15:17

value
0: div2
1: div3
2: div4
3: div 6
4: div8
5–7: reserved

The SysAd interface divide ration is also visible in the Config register in bits EC(28:30). The value of EC is the same as in the chart above.

2. A new bit was added to CacheErr register to indicate if a cache error occurs while handling external requests. These errors get masked if the processor is executing in an exception handler. In such situations, cache coherency may not hold.

To fix this problem, a new bit was introduced called EW, in the CacheErr register. The description of this bit follows:

EW is the 23rd bit in Cache Err register.

EW bit indicates that a MP critical cache error occurred, but the CacheErr register is not updated as it was busy holding the values of previous cache error.

Following errors can set EW bit.

1.SCacheTag errors due to an external request handling.
2.SCache Data errors due to an external update handling.
3.PCache Tag errors due to an external request handling.

Additional note for case 1:

Case 1 covers only multi bit tag errors. The bit is set for all external requests including intervention and snoops. The reason this was done is because the system would reset the R4000 if it saw the snoop/intervention response having erroneous data bit set. Therefore, for consistency reasons, this is done for all external requests.

3. The R4400 has an uncached store buffer to improve the performance of uncached stores over the previous R4000.

On the R4400, there is a single entry uncached store buffer. This means when an uncached store reaches the WB stage of the pipeline, the store drops into the uncached store buffer and the CPU pipeline may continue to run. If another uncached store reaches the WB stage of the pipeline before the first uncached store has been sent out of the chip, the CPU will stall until the uncached store buffer has completed the first uncached store.

If other useful instructions can be scheduled between successive uncached stores, then considerable performance improvement may be achieved using this mechanism.

To avoid CPU stalls, for sequential stores, there needs to be 7 cycles between uncached stores:

```
sw r2, (r3) # uncached store
1:nop
2:nop
3:nop
4:nop
5:nop
6:nop
7:nop
sw r2, (r3) # uncached store
```

Remember that if any of the instructions between the stores are multiple cycles then the number of instructions between the stores can be reduced further without causing stalls (for example, variable shifts take 2 EX cycles).

If a sequence of uncached stores is executed in a loop, then the 2 killed cycles that are lost in branch latency still count towards the 7 cycles needed between the stores:

```
loop:  sw r2, (r3) # uncached stores
1:     nop
2:     nop
3:     nop
4:     bloop
5:     nop
6:     killed
7:     killed
```

In this case only 4 instructions have to be scheduled (excluding the branch).

The latency between stores is really defined by the fact that back to back uncached stores on the system interface can be sent out at a maximum rate of one store every 4 external cycles (Address, Data, X, X, Address, Data...). This translates to 8 pipeline cycles for a chip running the system interface in divide by 2 mode. If a larger clock divisor is used (div3, div4), then the number of cycles between successive uncached stores goes up by the same ratio (div3 11 pipeline cycles, div4 15 pipeline cycles) if you want to avoid CPU pipeline stalls.

The SYNC instruction is used by the R4400 to ensure that any uncached store in the uncached store buffer is sent out of the chip before any load or store after the SYNC completes.

As external requests have higher priority than pending uncached stores, it is possible for an external agent to see the R4400 complete cached and uncached stores out of program execution order. For example, given the sequence:

```
sw r2, (r3) # uncached store
sw r4, (r5) # cached store
```

Given an external intervention or snoop is sent into the chip when the uncached store has entered the uncached store buffer, but has not yet been sent out of the chip, and the cached store has run successfully through the TC stage (i.e., it has hit in the primary cache.) The external agent will see the state of the internal caches after the cached store but before the result of the uncached store is seen outside the chip. The SYNC instruction may be used to avoid this case if necessary, as shown below:

```
sw r2, (r3) # uncached store
SYNC
sw r4, (r5) # cached store
```

4. The primary caches on the R4400 are 16K for both instruction and data. This change is visible in the Config register in bits DC(6:8) and IC(9:11) which indicate the primary cache size according to the formula 2**(12+DC) and 2**(12+IC).
5. The status pins function in the R4400. The encoding is as follows. Note that the encoding is different than the 2.9.1 version if the interface specification.

   Status <7..4> Processor Internal State or Status <3..0>

   0 Run cycle: Other integer Instruction (not Load/ Store/Conditional Branch)
   1 Run cycle:Load
   2 Run cycle: Untaken Conditional Branch
   3 Run cycle: Taken Conditional Branch
   4 Run cycle: Store
   5 Reserved
   6 MP stall
   7 Run cylce: Integer instruction killed by slip
   8 Stall cycle: Other stall type
   9 Stall cycle: Primary instruction cache
   a Stall cycle: Primary data cache
   b Stall cycle: Secondary cache
   c Run cycle: Other FP Instruction (not Load/Store/ Conditional Branch)
   d Run cycle: Killed by Branch/Jmp[/ERET
   e Run cycle: Instruction killed by exception
   f Run cycle: Floating-point instruction killed by slip
6. Master/Checker operation is also fully functional. These features can be enabled by Boot Time Mode- bits SIMasterMd (bit 18) and SCMasterMd (bit 42).

Master/Checker Mode in Toshiba's TC85R4400

Two R4400's can be configured as a lock-stepped pair to improve data integrity. For such configurations, outputs (and I/O busses during output) are checked between the two chips connected in parallel at bus cycle boundaries. Any discrepancies between them are reported through the assertion of the Fault* pin.

Two boot-time mode bits determine these Master/Checker configurations. For single-chip operations, these bits must be set to 00. This is the "Complete Master" mode, where the R4400's outputs operate normally.

There are two possible lock-step configurations. The simple case pairs a "Complete Master" R4400 (with mode bits set to 00) with a "Complete Listener" R4400, with mode bits set to 11. The latter has output drivers disabled. Since it receives all the same inputs as the former, both R4400's operate identically. On all output cycles, it samples the signals on output and I/O busses and compares with what it expects. Any discrepancies are reported through the Fault* pin.

The second lock-step configuration is called Cross-Coupled Checking. In general, one R4400 is set to drive the data pins of one bus and the other the check pins (parity or EEC) of the same bus. The mode bits to use are 01 and 10. Both chips sample and compare what are on the busses and indicate any discrepancies through their Fault* pins.

Additionally, and for any configuration, the Fault* pin also reports error conditions not covered by R4400 exceptions. (These include input parity error at SysCmd, input addresses and data on SysAD for certain system interface operations.) Note that the fault detection mechanism related to this Fault* pin does not cause any exceptions for the R4400. It continues to run regardless of the state of the pin. External logic must act on the fault and take appropriate action to interrupt or reset the R4400's.

Connections

For lock-step operations, most pins of a pair of R4400's are intended to be connected in parallel. The following discussion refers to signal groups as defined in Chapter 8–Signal Descriptions.

The signal groups to be connected in parallel are:

*System Interface
*Secondary Cache Interface
*Interrupt Interface

Some signals in the following groups are NOT to be connected in parallel:

*Initialization Interface
ModeClock, ModeIn and Reset*
*JTAG Interface
JTDO, JTMS

The remaining signals in these groups can be connected either way, independently or in parallel, according to other considerations.

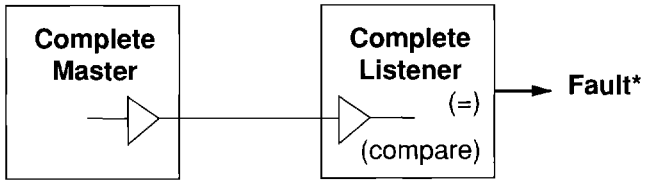
Finally, all of the signals in the Clock/Control Interface are NOT to be connected in parallel except, obviously, $V_{CC}P$ and $V_{SS}P$.

Modes of Operation

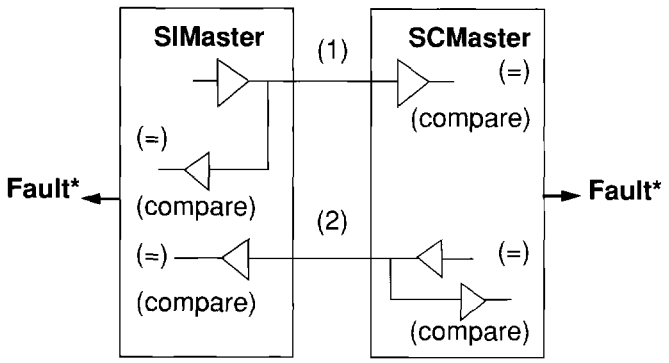
The Master/Checker mode of operation for each R4400 is set via boot-time mode bits: System Interface Master (SIMasterMd) and Secondary Cache Master (SCMaster MD). The following are defined:

SIMasterMd	SCMasterMd	Modes
0	0	Complete Master (required for single-chip operation
1	1	Complete Listener (to pair with Complete Master)
0	1	System Interface Master (SIMaster)
1	0	Secondary Cache Master (SCMaster, to pair with SIMaster)

Schematically, the Master-Listener configuration operates as follows:



And the Cross-Couples Checking configuration operates as follows:



(1) Signals connected in parallel and driven from the SIMaster are:

SysAD(63:0), SysCmd(8:0), and SCAPar(2:0)

(2) Signals connected in parallel and driven from the SCmaster are:

SysADC(7:0), SysCmdP, ValidOut*, Release*, SCAddr(17:1), SCAddr0(w,x,y,z), SCOE*, SCWr(w,x,y,z)*, SCData(127:0), SCDChk(15:0), SCTag(24:0), SCTChk(6:0), SCDCS*, SCTCS*

Fault Detection

Output miscomparison fault detection occurs at the end of bus cycles. The length of these cycles is determined by the appropriate boot-time mode bits.

At the System Interface, outputs are checked at the end of every System Interface cycle whenever the R4400 System Interface is in master state.

At the Secondary Cache Interface, outputs are checked at the end of every read or write cycle.

A special note about the SCAPar(2:0) signals: they are delayed from the rest of the Secondary Cache Interface by exactly one PClock. This includes when they transition, and when they are checked. The transitions follow exactly one PClock after SCAddr transitions, and when the R4400 initially determines its changing between a read and a write cycle without an address change. They do not follow the timing of the SCWr* signals, which are separately settable via the boot-time mode bits.

Fault Reporting

The external fault indication is reported through the Fault* pin. This is a non-persistent signal which operates on the same bus cycles as the System Interface. That is, it transitions at System Interface boundaries, as determined by the PClock-to-SClock divisor from the boot-time mode bits.

There is an internal fault detection latency of 4PClocks. Then the fault signal must synchronize with the System Interface. Any output faults detected and fully-propagated through the internal fault logic within the previous System Interface cycle will be reported in the current cycle.

Note that in the Complete Master mode, output fault reporting is disabled for Secondary Cache Interface pins, but enabled for System Interface pins (SysCmd, SysCmdP, SysAD, SysADC, ValidOut* and Release*.)

Reset Operations

If the Master/Checker mode of the R4400 is Complete Listener, SIMaster or SCMaster, an assertion of Reset* is significant. On boot-up, the first time Reset* is deasserted, the Master/Checker mode is determined by the boot-time mode bits. On the next assertion of Reset*, the mode is forced to be a Forced Complete Master mode. Hence, if Reset* is deasserted, the R4400 will be driving all outputs. Upon yet another Reset* assertion, the R4400 returns to the previous mode as determined by the mode bits. This alternating behavior repeats on subsequent transitions of Reset*.

There is a slight difference between this forced Complete Master mode and the normal Complete Master mode. That is, the Fault* pin continues to report all output faults when the R4400 was in the former mode, not just those of the System Interface.

Fault History Mechanism

Since both output faults and certain input faults are reported through the Fault* pin, there are two fault history bits that record which one or both of them has occurred. These bits are cleared at every deasserting transition of Reset*.

These bits are readable only when the Reset* is asserted. The Fault* pin changes from reporting live faults to indicating which fault history bit was set in the previous time period when Reset* was deasserted. The ModeIn pin acts as the selector. If ModeIn is 0, the Fault* shows the state of the Output Fault History bit (inverted). If ModeIn is 1, it shows the state of the Input Fault History bit (inverted).

There is also a mechanism to reset the fault history bits while the R4400 is running. If the ModeIn pin is asserted, these bits are cleared. This also means that, for the duration when latching fault history bits are desired, ModeIn must be maintained at 0.

The following table summarizes some of the above information:

BOOT/RESET CONTROLS	MODEIN PIN	FAULT HISTORY BITS	FAULT* PIN	MASTER/ CHECKER MODE
V_{CCOK} just asserted (0->1)	Used as mode bits scan data	—	—	—
Reset* just deasserted (0->1)	—	Cleared	—	—
Reset* deasserted (normal operation)	0	Set and latch if fault	Live faults reported	—
Reset* deasserted (normal operation)	1	Cleared	Live faults reported	—
Reset* just asserted (1->0)	—	—	—	Changed, toggling between mode bits and forced master
Reset* just asserted (R4400 in reset)	0	Output Fault History bit connected to Fault* pin	—	
Reset* just asserted (R4400 in reset)	1	Input Fault History bit connected to Fault* pin	—	