9. Error Protection and Handling

This chapter presents the error protection and handling features provided by the R10000 processor.

Two types of errors can occur in an R10000 system:

- correctable
- uncorrectable

The following two sections describe them.

9.1 Correctable Errors

Correctable errors consist of:

- secondary cache tag array correctable ECC errors
- secondary cache data array correctable ECC errors
- System interface address/data bus correctable ECC errors

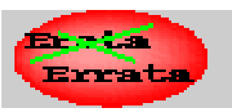
When the processor detects a correctable error, the error is automatically corrected, and normal operation continues. Secondary cache array scrubbing is not performed.

The processor informs the external agent that a correctable error was detected and then corrected by asserting the **SysCorErr*** signal for one **SysClk** cycle.

9.2 Uncorrectable Errors

Uncorrectable errors consist of:

- Primary instruction cache array parity errors
- Primary data cache array parity errors
- Secondary cache tag array uncorrectable ECC errors
- Secondary cache data array uncorrectable ECC errors
- System interface command bus parity errors
- System interface address/data bus uncorrectable ECC errors
- System interface response bus parity errors



When the processor detects an uncorrectable error, a Cache Error exception is posted. In general, the detection of an uncorrectable error does not disrupt any ongoing operations. However, the instruction fetch and load/store units never use data which contains an uncorrectable error.

To inform the external agent, the processor asserts **SysUncErr*** for one **SysClk** cycle whenever any of the following uncorrectable errors are detected:

- Primary instruction cache tag array parity errors
- Primary data cache tag array parity errors
- Secondary cache tag array uncorrectable ECC errors
- System interface command bus parity errors
- System interface address/data bus external address cycle uncorrectable ECC errors
- System interface response bus parity errors. (See page 158 in Errata.)

The processor informs the external agent that an uncorrectable tag error has been detected by asserting **SysUncErr*** for one **SysClk** cycle.

9.3 Propagation of Uncorrectable Errors

The processor assists the external agent in limiting the propagation of uncorrectable errors in the following manner:

- During external block data response cycles, if the data quality indication on **SysCmd(5)** is asserted, or if an uncorrectable ECC error is encountered on the system address/data bus while the ECC check indication on **SysCmd(0)** is asserted, the processor intentionally corrupts the ECC of the corresponding secondary cache quadword after receiving an external ACK completion response.
- During processor data cycles, the processor asserts the data quality indication on **SysCmd(5)** if the data is known to contain uncorrectable errors. The System interface ECC is never intentionally corrupted; the **SysCmd(5)** bit is used to indicate corrupted data.
- If an uncorrectable cache tag error is detected, the processor asserts **SysUncErr*** for one **SysClk** cycle.
- An external coherency request that detects a secondary cache tag array uncorrectable error asserts the secondary cache block tag quality indication on **SysState(2)** during the corresponding processor coherency state response.
- If an external coherency request requires a processor coherency data response, and a primary data cache tag parity error is encountered during the primary cache interrogation, or a secondary cache tag array uncorrectable error is encountered during the secondary cache interrogation, the processor asserts the data quality indication on **SysCmd(5)** for all doublewords of the corresponding processor coherency data response.

9.4 Cache Error Exception

The processor indicates an uncorrectable error has occurred by asserting a Cache Error exception.

The following four internal units detect and report uncorrectable errors:

- instruction cache
- data cache
- secondary cache
- System interface

Each of these four units maintains a unique local *CacheErr* register.

A Cache Error exception is imprecise; that is, it is not associated with a particular instruction. When any of the four units post a Cache Error exception, completed instructions are graduated before the Cache Error exception is taken. If there are Cache Error exceptions posted from more than one of the units, the exceptions are prioritized in the following order:

1. instruction cache
2. data cache
3. secondary cache
4. System interface.

The corresponding local *CacheErr* register is transferred to the CP0 *CacheErr* register and the CP0 *Status* register *ERL* bit is asserted. Instruction fetching begins from 0xa0000100 or 0xbfc00300, depending on the CP0 *Status* register *BEV* bit. The CP0 *ErrorEPC* register is loaded with the virtual address of the next instruction that has not been graduated, so that execution can resume after the Cache Error exception handler completes.

When *ERL*=1, the user address region becomes a 2-Gbyte uncached space mapped directly to the physical addresses. This allows the Cache Error handler to save registers directly to memory without having to use a register to construct the address.

The processor does not support nested Cache Error exception handling. While the CP0 *Status* register *ERL* bit is asserted, any subsequent Cache Error exceptions are ignored. However, the detection of additional uncorrectable errors is not inhibited, and additional Cache Error exceptions may be posted.†

† The hardware does not handle the case of multiple Cache Error exceptions in any special manner; caches are refilled as normal, and data forwarded to the appropriate functional units.

9.5 CP0 *CacheErr* Register *EW* Bit

When a unit detects an uncorrectable error, it records information about the error in its local *CacheErr* register and posts a Cache Error exception. If a subsequent uncorrectable error occurs while waiting for the Cache Error exception to be taken and transfer of the local *CacheErr* register to the CP0 *CacheErr* register to complete, the *EW* bit is set in its local *CacheErr* register. Once the Cache Error exception is taken, the *EW* bit in the CP0 *CacheErr* register is set and the Cache Error exception handler now determines that a second error has occurred.

Once the CP0 *CacheErr* register *EW* bit is set, it can only be cleared by a reset sequence.

9.6 CP0 *Status* Register *DE* Bit

Asserting the CP0 *Status* register *DE* bit suppresses the posting of future Cache Error exceptions. All local *CacheErr* registers are also prevented from being updated. Unlike the R4400 processor architecture, when the *DE* bit is asserted, cache hits are not inhibited when an uncorrectable error is detected. Correctable errors are handled normally when the *DE* bit is set.

NOTE: Be careful when setting this bit, since it may cause erroneous data and/or instructions to be propagated.

9.7 CACHE Instruction

Uncorrectable error protection is suppressed for the Index Load Tag, Index Store Tag, Index Load Data, and Index Store Data CACHE instruction variations. These four variations may be used within a Cache Error exception handler to examine the cache tags and data without the occurrence of further uncorrectable errors.

9.8 Error Protection Schemes Used by R10000

Error protection schemes used in the R10000 processor are:

- parity
- sparse encoding
- ECC

These schemes are described in this section, and listed in Table 9-1.

Table 9-1 Error Protection Schemes Used in the R10000 Processor

Error Detection Used	What is Protected
Parity	Primary caches Secondary cache data System interface buses
Sparse encoding	Primary data cache state mod array
ECC (SECDED)	Secondary cache tag Secondary cache data System interface address/data bus

Parity

Parity is used to protect the primary caches and various System interface buses. The processor uses both odd and even parity schemes:

- in an odd parity scheme, the total number of ones on the protected data and the corresponding parity bit should be odd
- in an even parity scheme, the total number of ones on the protected data and the corresponding parity bit should be even.

Sparse Encoding

A sparse encoding is used to protect the primary data cache state mod array. In such a scheme, valid encodings are chosen so that altering a single bit creates an invalid encoding.

ECC

An error correcting code (ECC) is used to protect the secondary cache tag, the secondary cache data, and the System interface address/data bus. A distinct single-bit error correction and double-bit error detection (SECDED) code is used for each of these three applications.

9.9 Primary Instruction Cache Error Protection and Handling

This section describes error protection and error handling schemes for the primary instruction cache.

Error Protection

The primary instruction cache arrays have the following error protection schemes, as listed in Table 9-2.

Table 9-2 Primary Instruction Cache Array Error Protection

Array	Width	Error Protection
Tag Address	27-bit	Even parity
Tag State	1-bit	Even parity
Data	36-bit	Even parity
LRU	1-bit	None

Error Handling

All primary instruction cache errors are uncorrectable. If an error is detected, the instruction cache unit posts a Cache Error exception and initializes the *D*, *TA*, *TS*, and *PIdx* fields in the local *CacheErr* register (see Chapter 14, *CacheErr Register (27)*, for more information). If an error is detected on the tag address or state array, the processor informs the external agent that an uncorrectable tag error was detected by asserting **SysUncErr*** for one **SysClk** cycle.

9.10 Primary Data Cache Error Protection and Handling

This section describes error protection and error handling schemes for the primary data cache.

Error Protection

The primary data cache arrays have the following error protection schemes, as listed in Table 9-3.

Table 9-3 Primary Data Cache Array Error Protection

Array	Width	Error Protection
Tag Address	28-bit	Even parity
Tag State	3-bit	Even parity
Tag Mod	3-bit	Sparse encoding
Data	8-bit	Even parity
LRU	1-bit	None

Error Handling

All primary data cache errors are uncorrectable. If an error is detected, the data cache unit posts a Cache Error exception and initializes the *EE*, *D*, *TA*, *TS*, *TM*, and *PIdx* fields in the local *CacheErr* register (see Chapter 14, *CacheErr Register (27)*, for more information). If an error is detected on the tag address, state, or mod array, the processor informs the external agent that an uncorrectable tag error was detected by asserting **SysUncErr*** for one **SysClk** cycle.

9.11 Secondary Cache Error Protection and Handling

This section describes error protection and error handling schemes for the secondary cache.

Error Protection

The secondary cache arrays have the following error protection schemes, as listed in Table 9-4.

Table 9-4 Secondary Cache Array Error Protection

Array	Width	Error Protection
Data	128-bit	9-bit ECC + even parity
Tag	26-bit	7-bit ECC
MRU (Way prediction table)	1-bit	None

Error Handling

This section describes error handling for the data array and the tag array. As shown in Table 9-4, errors are not detected for the way prediction table.

Data Array

The 128-bit wide secondary cache data array is protected by a 9-bit wide ECC. An even parity bit is used for rapid detection of correctable (single-bit) errors; when a correctable parity error is detected, the data is sent through the data corrector. The parity bit does not have any logical effect on the processor's ability to either detect or correct errors.

Whenever the processor writes the secondary cache data array, it drives the proper ECC on **SCDataChk(8:0)** and even parity on **SCDataChk(9)**.

Data Array in Correction Mode

The secondary cache operates in correction mode when the **SCCorEn** mode bit is asserted. Whenever the processor reads the secondary cache data array in correction mode, the data is sent through a data corrector.

If a correctable error is detected, in-line correction is automatically made without affecting latency. The processor informs the external agent that a correctable error was detected and corrected by asserting **SysCorErr*** for one **SysClk** cycle.

If an uncorrectable error is detected, the secondary cache unit posts a Cache Error exception and initializes the *D* and *SIdx* fields in the local *CacheErr* register (see Chapter 14, *CacheErr Register (27)*, for more information).

In correction mode, secondary-to-primary cache refill latency is increased by two **PClk** cycles. Multiple processors, operating in a lock-step fashion, remain synchronized in the presence of secondary cache data array correctable errors.

Table 9-5 presents the ECC matrix for the secondary cache data array.

Table 9-5 ECC Matrix for Secondary Cache Data Array

Check Bit		8	7654	3210																																
Data Bit					1111 2222 7654	1111 2222 3210	1111 1111 9876	1111 1111 5432	1111 1100 1098	1111 0000 7654	1111 0000 3210	9999 9876	9999 5432	9988 1098	8888 7654	8888 3210	7777 9876	7777 5432	7766 1098	6666 7654	6666 3210	5555 9876	5555 5432	5544 1098	4444 7654	4444 3210	3333 9876	3333 5432	3322 1098	2222 7654	2222 3210	1111 9876	1111 5432	11 1098	7654	3210
Number of ones per row	54	1	0000	0000	1111	1111	0000	0011	1111	1111	1110	0011	0000	0110	1111	1111	0000	0111	0010	0001	0110	0010	1101	0000	1011	0000	1101	0000	0000	0000	0010	0000	0010	0000	1000	0000
	53	0	1000	0000	1111	1111	1111	1111	0000	0011	0000	1111	1111	1111	0000	0111	0000	0011	1011	0010	0001	0000	0000	1000	0100	1000	1100	1000	0000	0010	1001	0000	1001	0000	0010	0000
	54	0	0100	0000	1000	0011	1111	1111	1111	1111	0000	0011	0000	0101	0000	0011	1111	1111	0101	1100	1000	0000	1000	0100	0000	0100	1100	0100	0011	1000	0100	1000	0100	1000	0101	0000
	53	0	0010	0000	0100	0010	1000	0000	1000	0000	0001	0010	1111	1100	1111	1011	1111	1100	0011	1111	1111	1101	0110	0010	0000	0010	0010	0010	1000	0101	0000	0100	1100	0100	0000	1000
	53	0	0001	0000	0010	0001	0100	0000	0100	0011	1111	1111	1000	0000	1000	0000	1000	0100	1111	1111	1111	1111	0010	0001	0000	0001	0000	0001	1111	1111	1100	0010	0000	0010	1000	0100
	53	0	0000	1000	0001	0000	0010	0011	0010	0000	1000	0001	0100	0100	0100	0000	0100	0110	1011	1111	1111	1100	0011	1111	1101	1111	0011	1111	0100	1100	0000	0001	0000	0001	0100	0010
	54	0	0000	0100	0000	1010	0001	0010	0001	0010	0001	1000	0010	0011	0010	0000	0010	0001	0000	0000	1011	1010	1111	1111	1100	0000	1010	0000	1101	0000	1111	1111	1111	1111	1100	0001
	53	0	0000	0010	0000	0100	0000	1001	0000	1001	0100	0100	0001	0011	0001	0010	0001	0000	0000	1000	0100	1101	1100	0000	1110	0000	1111	1111	1110	0000	1100	0000	1111	1111	1111	1111
	54	0	0000	0001	0000	0001	0000	0100	0000	0100	0010	0000	0000	1011	0000	1101	0000	1011	0100	0111	0000	0100	1110	0000	1111	1111	0110	0000	1100	0011	1111	1111	1100	0000	1111	1111
Number of ones per column		1	1111	1111	3333	3355	3333	3355	3333	3355	3333	3355	3333	3555	3333	3355	3333	3555	3355	5555	5555	5533	5553	3333	5533	3333	5553	3333	5533	3333	5533	3333	5533	3333	5533	3333

Data Array in Noncorrection Mode

When the **SCCorEn** mode bit is negated, the secondary cache operates in noncorrection mode. Whenever the processor reads the secondary cache data array in noncorrection mode, it checks for even parity on **SCDataChk(9)**. If a parity error is detected, it is assumed that a correctable error has occurred, and the secondary cache block is again read through a data corrector. During this re-read, the processor checks the **SCDataChk(8:0)** bus for the proper ECC.

If a correctable error is detected, correction is automatically performed in-line. To inform the external agent that a correctable error had been detected and corrected, the processor asserts **SysCorErr*** for one **SysClk** cycle.

If an uncorrectable error is detected, the secondary cache unit posts a Cache Error exception and initializes the *D* and *SIdx* fields in the local *CacheErr* register.

Secondary cache data array correctable errors are monitored with Performance Counter 0.

Tag Array

The 26-bit-wide secondary cache tag array is protected by a 7-bit-wide ECC. Table 9-6 presents the ECC matrix for the secondary cache tag array.

Table 9-6 ECC Matrix for Secondary Cache Tag Array

Check Bit			0	12	34	56				
Data Bit		2 5	222 432	22 10	11 98	11 76	1111 5432	11 1098	7654	3210
Number of ones per row	11	0	0100	1000	1000	0001	1111	1000	1000	1000
	13	0	1000	0100	0100	0010	1111	1111	0000	0100
	11	1	0010	1000	0001	1000	0000	1111	0100	0010
	11	1	0100	0100	0010	0100	1000	0100	1111	0000
	13	0	1000	0001	1000	1000	0100	0000	1111	1111
	12	1	0010	0010	0100	0100	0010	0010	0010	1111
	14	0	1111	1100	1100	1100	0001	0001	0001	0001
Number of ones per column		3	3331	3311	3311	3311	3333	3333	3333	3333

Whenever the processor reads the secondary cache tag array, it checks the **SCTagChk(6:0)** bus for the proper ECC. If a correctable error is detected, correction is automatically performed in-line, without affecting latency. The processor asserts **SysCorErr*** for one **SysClk** cycle to inform the external agent that a correctable error has been detected and corrected. If an uncorrectable error is detected, the secondary cache unit posts a Cache Error exception and initializes the *TA* and *SIdx* fields in the local *CacheErr* register. The processor asserts **SysUncErr*** for one **SysClk** cycle to inform the external agent that an uncorrectable tag error has been detected.

Whenever the processor writes the secondary cache tag array, it drives the proper ECC on the **SCTagChk(6:0)** bus.

9.12 System Interface Error Protection and Handling

This section describes error protection and error handling schemes for the System interface.

Error Protection

The System interface buses have the following error protection schemes, as listed in Table 9-7.

Table 9-7 System Interface Bus Error Protection

Bus	Width	Error Protection
SysCmd	12-bit	Odd parity
SysAD	64-bit	8-bit ECC
SysState	3-bit	Odd parity
SysResp	5-bit	Odd parity

Error Handling

This section describes error handling on the system command bus, system address/data bus, system state bus, and system response bus.

SysCmd(11:0) Bus

The 12-bit wide system command bus, **SysCmd(11:0)**, is protected by odd parity.

Whenever the processor is in master state and it asserts **SysVal*** to indicate that it is driving valid information on the **SysCmd(11:0)** bus, it also drives odd parity on the **SysCmdPar** signal.

Whenever the processor is in slave state and an external agent asserts **SysVal*** to indicate that it is driving valid information on the **SysCmd(11:0)** bus, the processor checks the **SysCmdPar** signal for odd parity. If a parity error is detected, the processor ignores the **SysCmd(11:0)** and **SysAD(63:0)** buses for one **SysClk** cycle. The System interface unit posts a Cache Error exception and sets the SC bit in the local *CacheErr* register. Additionally, the processor informs the external agent by asserting **SysUncErr*** for one **SysClk** cycle. (See page 168 in Errata.)

> **Caution:** By ignoring the **SysCmd(11:0)** and **SysAD(63:0)** buses, the processor to become unsynchronized with other processors or the external agent on the cluster bus.

SysAD(63:0) Bus

The 64-bit wide system address/data bus, **SysAD(63:0)**, is protected by an 8-bit-wide ECC.

Processor in Master State

Whenever the processor is in master state and it asserts **SysVal*** to indicate it is driving valid information on the **SysAD(63:0)** bus, it also drives the proper ECC on the **SysADChk(7:0)** bus.

Processor in Slave State

Whenever the processor is in slave state, error checking is enabled with the assertion of **SysCmd(0)**, and an external agent asserts **SysVal*** to indicate it is driving valid information on the **SysAD(63:0)** bus, the processor checks the **SysADChk(7:0)** bus for the proper ECC.

Correctable Error Detected

If a correctable error is detected during an external address cycle, or during an external data cycle for a processor read or upgrade request originated by the R10000 processor, correction is automatically performed in-line without affecting latency. The processor asserts **SysCorErr*** for one **SysClk** cycle to inform the external agent that a correctable error has been detected and corrected.

Uncorrectable Error Detected

If an uncorrectable error is detected during an external address cycle, the processor ignores the **SysCmd(11:0)** and **SysAD(63:0)** buses for one **SysClk** cycle, and the System interface unit posts a Cache Error exception and sets the *SA* bit in the local *CacheErr* register. Additionally, the processor informs the external agent by asserting **SysUncErr*** for one **SysClk** cycle. (See page 169 in Errata.)

> **Caution:** By ignoring the **SysCmd(11:0)** and **SysAD(63:0)** buses, this processor may become unsynchronized with other processors or the external agent on the cluster bus.

If an uncorrectable error is detected or the data quality indication on **SysCmd(5)** is asserted during an external data cycle for a processor read or upgrade request originated by the processor, the R10000 asserts the corresponding incoming buffer uncorrectable error flag.

When the processor forwards block data from an incoming buffer entry after receiving an external ACK completion response, the associated incoming buffer uncorrectable error flags are checked, and if any are asserted, the System interface unit posts a single Cache Error exception and initializes the *EE*, *D*, and *SIdx* fields in the local *CacheErr* register.

When the processor forwards double/single/partial-word data from an incoming buffer entry after receiving an external ACK completion response, the associated incoming buffer uncorrectable error flag is checked and, if asserted, the System interface unit posts a Bus Error exception.

Table 9-8 presents the ECC matrix for the System interface address/data bus. This ECC matrix is identical to that used by the R4X00 System interface.

Table 9-8 ECC Matrix for System Interface Address/Data Bus

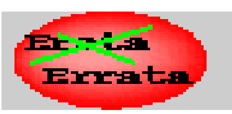
Check Bit			43		52										70		61	
Data Bit		6666 3210	55 98	5555 7654	55 32	5544 1098	4444 7654	4444 3210	3333 9876	3333 5432	3322 1098	2222 7654	2222 3210	1111 9876	1111 5432	11 10	9876	54	3210
Number of ones per row	27	1111	1100	1100	1000	1000	0000	1111	1111	0000	1000	1000	1000	1000	0000	1010	0100	1000	1000
	27	1111	1000	1000	1000	0100	0000	0000	0000	1111	0100	0100	0100	0100	1111	1100	1100	1010	0100
	27	0000	1000	1100	1010	0010	1111	1111	0000	0000	0010	0010	0010	0010	1111	1000	1000	1100	0010
	27	0000	1010	0100	1100	0001	1111	0000	1111	1111	0001	0001	0001	0001	0000	1000	1100	1000	0001
	27	1000	0101	0011	0100	0000	1000	1000	1000	1000	1111	1111	0000	1111	1000	1100	0001	0100	0000
	27	0100	1100	0010	0101	1111	0100	0100	0100	0100	0000	0000	1111	1111	0100	0100	0011	0100	0000
	27	0010	0100	0011	1100	1111	0010	0010	0010	0010	1111	0000	0000	0000	0010	0100	0010	0101	1111
	27	0001	0100	0001	0100	0000	0001	0001	0001	0001	0000	1111	1111	0000	0001	0101	0011	1100	1111
Number of ones per column		3333	5511	3333	5511	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	5511	3333	5511	3333

SysState(2:0) Bus

The 3-bit wide system state bus, **SysState(2:0)**, is protected by odd parity. The processor drives odd parity on the **SysStatePar** signal.

SysResp(4:0) Bus

The 5-bit wide system response bus, **SysResp(4:0)**, is protected by odd parity.



Whenever an external agent asserts **SysRespVal*** to indicate it is driving valid information on the **SysResp(4:0)** bus, the processor checks the **SysRespPar** signal for odd parity. If a parity error is detected, the processor ignores the **SysResp(4:0)** bus for one **SysClk** cycle. The System interface unit posts a Cache Error exception and sets the *SR* bit in the local *CacheErr* register. Additionally, the processor informs the external agent by asserting **SysUncErr*** for one **SysClk** cycle. (See page 170 in Errata.)

> **Caution:** If the processor ignores the **SysResp(4:0)** bus, it may become unsynchronized with other processors or the external agent on the cluster bus. Also, the processor will "hang" if a parity error is detected on the **SysResp[4:0]** bus during an external completion response cycle for a processor double/single/partial-word read request originated by the processor. The external agent may initiate a Soft Reset sequence to obtain the contents of the *CacheErr* register, and the *CacheErr* register will indicate a System interface uncorrectable system response bus error.

Protocol Observation

The processor continuously observes the protocol on the System interface. Table 9-9 presents the supported protocol observations and the associated error handling sequence.

Table 9-9 Protocol Observation

Protocol Observation	Error Handling
External response data cycle with an unexpected request number during an external block data response for a processor block read or upgrade request originated by the processor.	Ignore the external response data cycle
External block data response specifying a *Reserved* cache block state for a processor block read or upgrade request originated by the processor.	Override the cache block state to *CleanExclusive*
External block data response specifying a *Shared* cache block state for a processor coherent block read exclusive or upgrade request originated by the processor.	Override the cache block state to *CleanExclusive*
External completion response specifying a *Reserved* completion indication.	Ignore the external completion response
External ACK completion response for a processor read request originated by the processor that has not received an external data response.	Override the external ACK completion response to a NACK