Memory Management

16

This section describes the R10000 processor memory management, including:

- processor modes and exceptions
- virtual address space
- virtual address translation

16.1 Processor Modes

The R10000 has three operating modes and two addressing modes. All are described in this section.

Processor Operating Modes

The three operating modes are listed in order of decreasing system privilege:

- Kernel mode (highest system privilege): can access and change any register. The innermost core of the operating system runs in kernel mode.
- Supervisor mode : has fewer privileges and is used for less critical sections of the operating system.
- User mode (lowest system privilege): prevents users from interfering with one another.

Selection between the three modes can be made by the operating system (when in Kernal mode) by writing into *Status* register's *KSU* field. The processor is forced into Kernel mode when the processor is handling an error (the *ERL* bit is set) or an exception (the *EXL* bit is set). Table 16-1 shows the selection of operating modes with respect to the *KSU*, *EXL* and *ERL* bits.

Table 16-1 also shows how different instruction sets and addressing modes are enabled by the *Status* register's *XX*, *UX*, *SX* and *KX* bits. A dash ("-") in this table indicates a "don't care." For detailed information on the address spaces available in each mode, refer to section titled, Virtual Address Space, in this chapter.

The R10000 processor was designed for use with the MIPS IV ISA; however, for compatibility with earlier machines, the useable ISAs can be limited to either MIPS III or MIPSI/II.

Table 16-1 Processor Modes

XX 31	KX 7	SX 6	UX 5	KSU 4:3	ERL 2	EXL 1	Description	ISA * III	ISA * IV	Addressing Mode 32-Bit/64-Bit
0	-†	-	0	10	0	0	User mode.	No	No	32
1	-	-	0	10	0	0		No	Yes	32
0	-	-	1	10	0	0		Yes	No	64
1	-	-	1	10	0	0		Yes	Yes	64
-	-	0	-	01	0	0	Supervisor mode.	No	Yes	32
-	-	1	-	01	0	0		Yes	Yes	64
-	0	-	-	00	0	0	Kernel mode.	Yes	Yes	32
-	1	-	-	00	0	0		Yes	Yes	64
-	0	-	-	-	0	1	Exception Level	Yes	Yes	32
-	1	-	-	-	0	1		Yes	Yes	64
-	0	-	-	-	1	X	Error Level.	Yes	Yes	32
-	1	-	-	-	1	X		Yes	Yes	64

* No means the ISA is disabled; Yes means the ISA is enabled.

† Dashes (-) are "don't care."

Addressing Modes

The processor's *addressing mode* determines whether it generates 32-bit or 64-bit memory addresses.

Refer to Table 16-1 for the following addressing mode encodings:

- In Kernel mode the *KX* bit allows 64-bit addressing; all instructions are always valid.
- In Supervisor mode, the *SX* bit allows 64-bit addressing and the MIPS III instructions. MIPS IV ISA is enabled all the time in Supervisor mode.
- In User mode, the *UX* bit allows 64-bit addressing and the MIPS III instructions; the *XX* bit allows the new MIPS IV instructions.

16.2 Virtual Address Space

The processor uses either 32-bit or 64-bit address spaces, depending on the operating and addressing modes set by the *Status* register. Table 16-1 lists the decoding of these modes.

The processor uses the following addresses:

- virtual address VA[43:0]
- region bits VA[63:59]

If a region is mapped , virtual addresses are translated in the TLB. Bits VA[58:44] are not translated in the TLB and are sign extensions of bit VA[43] .

In both 32-bit and 64-bit address mode, the memory address space is divided into many regions, as shown in Figure 16-3. Each region has specific characteristics and uses. The user can access only the *useg* region in 32-bit mode, or *xuseg* in 64-bit mode, as shown in Figure 16-1. The supervisor can access user regions as well as *sseg* (in 32-bit mode) or *xsseg* and *csseg* (in 64-bit mode), shown in Figure 16-2. The kernel can access all regions except those restricted because bits VA[58:44] are not implemented in the TLB, as shown in Figure 16-3.

The R10000 processor follows the R4400 implementation for *data* references only, ensuring compatibility with the NT kernel. If any of the upper 33 bits are nonzero for an instruction fetch, an Address Error is generated. Refer to Table 16-2 for delineation of the address spaces.

User Mode Operations

In User mode, a single, uniform virtual address space—labelled User segment—is available; its size is:

- 2 Gbytes (2^{31} bytes) in 32-bit mode (*useg*)
- 16 Tbytes (2^{44} bytes) in 64-bit mode (*xuseg*)

Figure 16-1 shows User mode virtual address space.

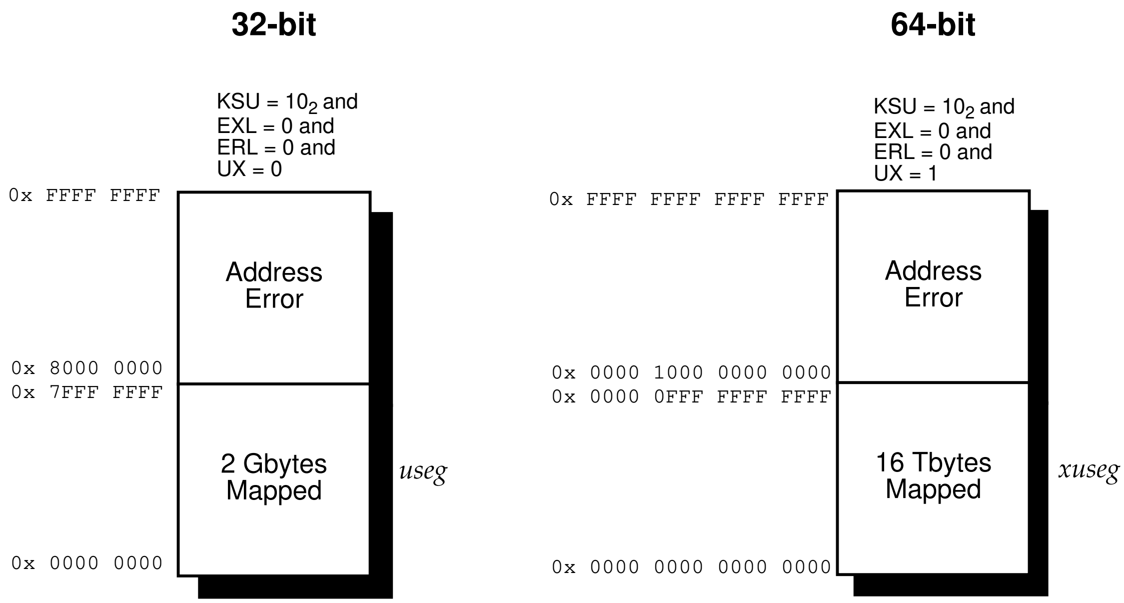

Figure 16-1 User Mode Virtual Address Space

The User segment starts at address 0 and the current active user process resides in either *useg* (in 32-bit mode) or *xuseg* (in 64-bit mode). The TLB identically maps all references to *useg*/*xuseg* from all modes, and controls cache accessibility.

32-bit User Mode (*useg*)

In User mode, when *UX* = 0 in the *Status* register, User mode addressing is compatible with the 32-bit addressing model shown in Figure 16-1, and a 2-Gbyte user address space is available, labelled *useg*.

All valid User mode virtual addresses have their most-significant bit cleared to 0; any attempt to reference an address with the most-significant bit set while in User mode causes an Address Error exception.

The system maps all references to *useg* through the TLB, and bit settings within the TLB entry for the page determine the cacheability of a reference.

64-bit User Mode (*xuseg*)

In User mode, when *UX* =1 in the *Status* register, User mode addressing is extended to the 64-bit model shown in Figure 16-1. In 64-bit User mode, the processor provides a single, uniform virtual address space of 2^{44} bytes, labelled *xuseg*.

All valid User mode virtual addresses have bits 63:44 equal to 0; an attempt to reference an address with bits 63:44 not equal to 0 causes an Address Error exception.

Although the system may be in 32-bit mode, address logic still generates 64-bit values. In this case the high 32 bits must equal the sign bit (31), or an Address Error exception is taken.

Supervisor Mode Operations

Supervisor mode is designed for layered operating systems in which a true kernel runs in processor Kernel mode, and the rest of the operating system runs in Supervisor mode.

The processor operates in Supervisor mode when the *Status* register contains the Supervisor-mode bit-values shown in Table 16-1.

Figure 16-2 shows Supervisor mode address mapping.

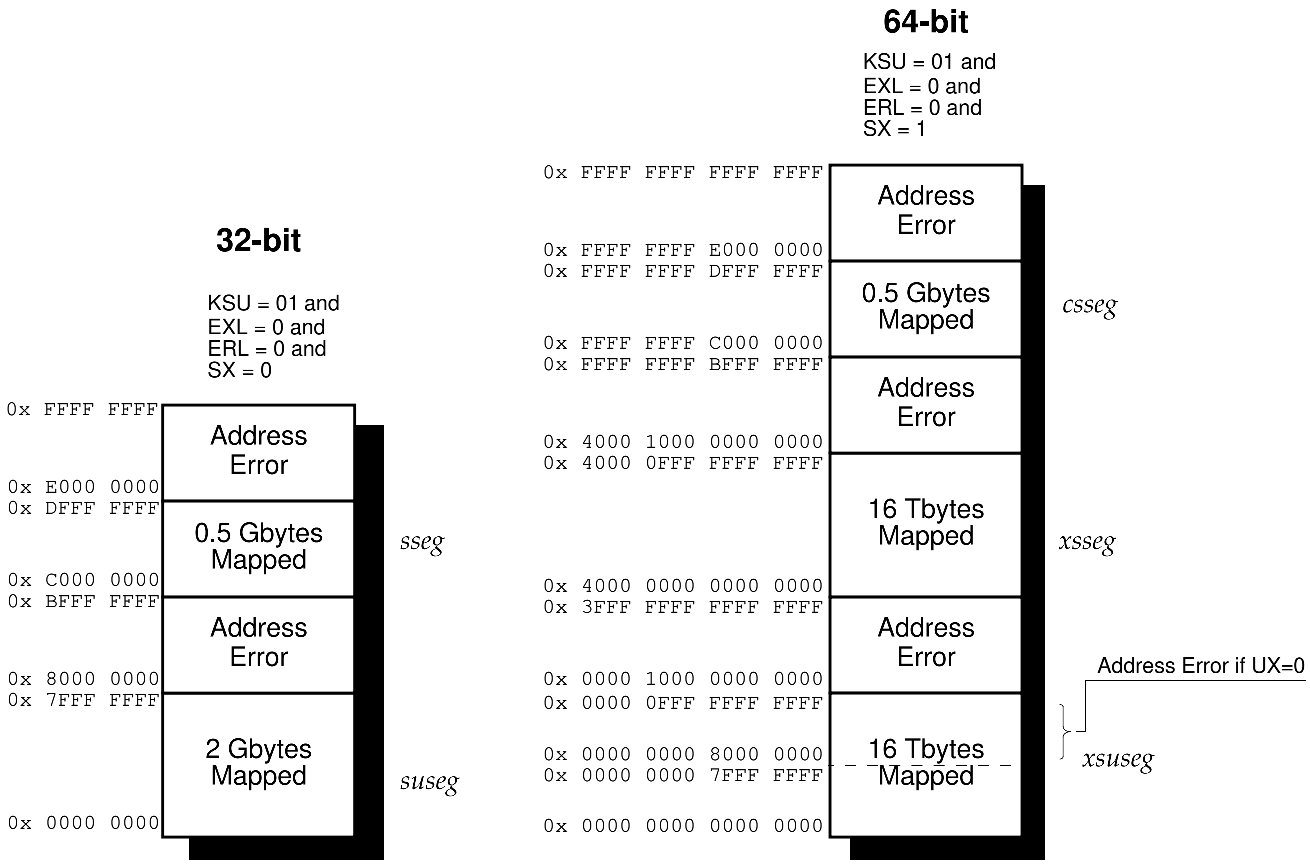

Figure 16-2 Supervisor Mode Address Space

32-bit Supervisor Mode, User Space (*suseg*)

In Supervisor mode, when *SX* = 0 in the *Status* register and the most-significant bit of the 32-bit virtual address is set to 0, the *suseg* virtual address space is selected; it covers the full 2^{31} bytes (2 Gbytes) of the current user address space. The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

This mapped space starts at virtual address 0x0000 0000 and runs through 0x7FFF FFFF.

32-bit Supervisor Mode, Supervisor Space (*sseg*)

In Supervisor mode, when *SX* = 0 in the *Status* register and the three most-significant bits of the 32-bit virtual address are 110_2, the *sseg* virtual address space is selected; it covers 2^{29}-bytes (512 Mbytes) of the current supervisor address space. The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

This mapped space begins at virtual address 0xC000 0000 and runs through 0xDFFF FFFF.

64-bit Supervisor Mode, User Space (*xsuseg*)

In Supervisor mode, when *SX* = 1 in the *Status* register and bits 63:62 of the virtual address are set to 00_2, selection of the *xsuseg* virtual address space is dependent upon the *UX* bit.

- if *UX* = 1, the entire space from 0x0000 0000 0000 0000 through 0000 0FFF FFFF FFFF (16 Tbytes) is selected.
- If *UX* = 0, the address space 0x0000 0000 0000 0000 through 0000 0000 7FFF FFFF (2 Gbytes) is selected. Addressing the space ranging from 0000 0000 8000 0000 through 0000 0FFF FFFF FFFF will cause an address error.

The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

64-bit Supervisor Mode, Current Supervisor Space (*xsseg*)

In Supervisor mode, when *SX* = 1 in the *Status* register and bits 63:62 of the virtual address are set to 01_2, the *xsseg* current supervisor virtual address space is selected. The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

This mapped space begins at virtual address 0x4000 0000 0000 0000 and runs through 0x4000 0FFF FFFF FFFF.

64-bit Supervisor Mode, Separate Supervisor Space (*csseg*)

In Supervisor mode, when *SX* = 1 in the *Status* register and bits 63:62 of the virtual address are set to 11_2, the *csseg* separate supervisor virtual address space is selected. Addressing of the *csseg* is compatible with addressing *sseg* in 32-bit mode. The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

This mapped space begins at virtual address 0xFFFF FFFF C000 0000 and runs through 0xFFFF FFFF DFFF FFFF.

Kernel Mode Operations

The processor operates in Kernel mode when the *Status* register contains the Kernel-mode bit-values shown in Table 16-1.

Kernel mode virtual address space is divided into regions differentiated by the high-order bits of the virtual address, as shown in Figure 16-3.

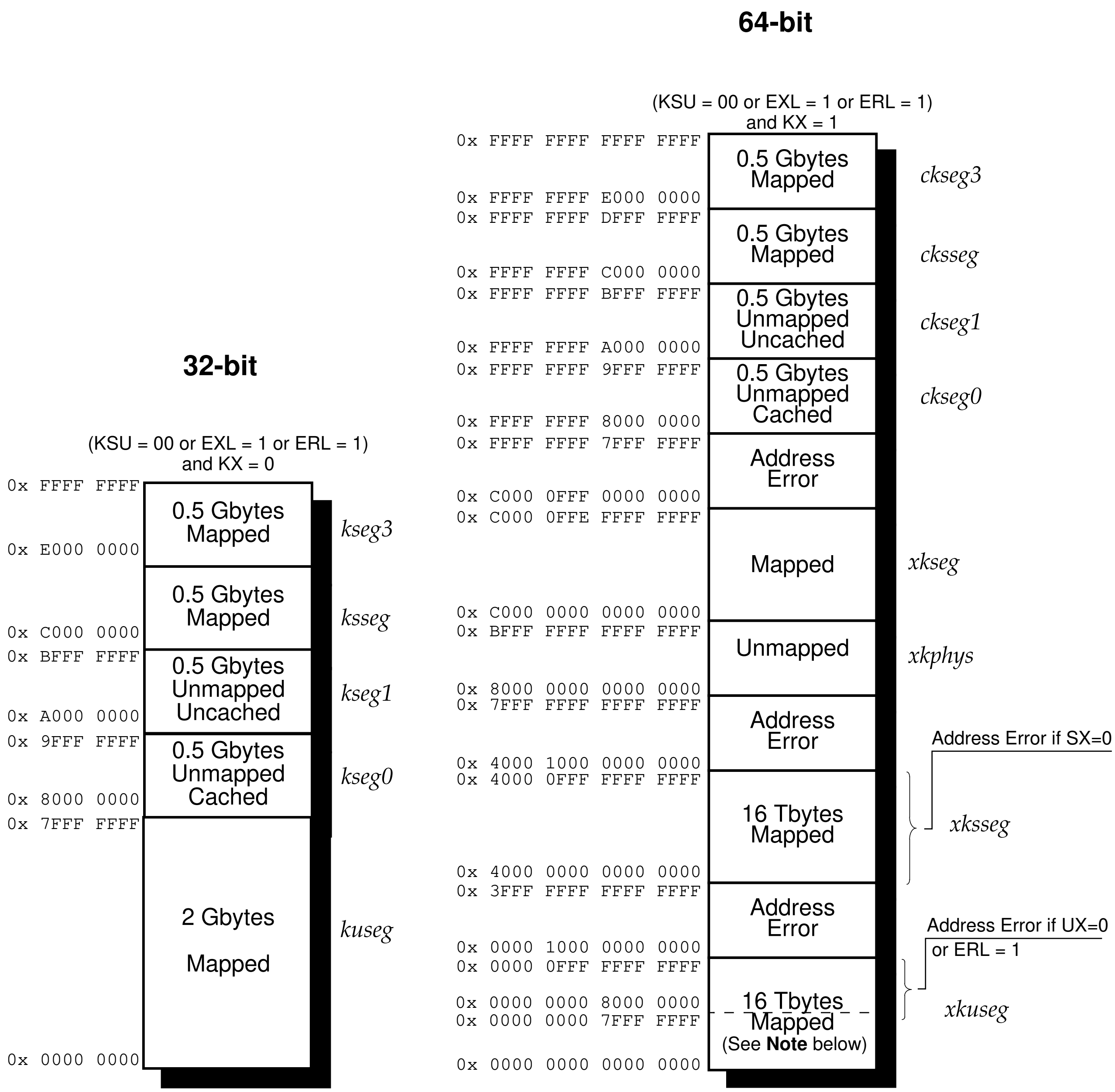

Figure 16-3 Kernel Mode Address Space

NOTE: If *ERL* = 1, the selected 2 Gbyte space becomes uncached and unmapped.

32-bit Kernel Mode, User Space (*kuseg*)

In Kernel mode, when *KX* = 0 in the *Status* register, and the most-significant bit of the virtual address, A31, is cleared, the 32-bit *kuseg* virtual address space is selected; it covers the full 2^{31} bytes (2 Gbytes) of the current user address space. The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

32-bit Kernel Mode, Kernel Space 0 (*kseg0*)

In Kernel mode, when *KX* = 0 in the *Status* register and the most-significant three bits of the virtual address are 100_2, 32-bit *kseg0* virtual address space is selected; it is the 2^{29}-byte (512-Mbyte) kernel physical space. References to *kseg0* are not mapped through the TLB; the physical address is selected by subtracting 0x8000 0000 from the virtual address. The *K0* field of the *Config* register determines cacheability and coherency.

32-bit Kernel Mode, Kernel Space 1 (*kseg1*)

In Kernel mode, when *KX* = 0 in the *Status* register and the most-significant three bits of the 32-bit virtual address are 101_2, 32-bit *kseg1* virtual address space is selected; it is the 2^{29}-byte (512-Mbyte) kernel physical space.

References to *kseg1* are not mapped through the TLB; the physical address is selected by subtracting 0xA000 0000 from the virtual address.

Caches are disabled for accesses to these addresses, and physical memory (or memory-mapped I/O device registers) are accessed directly.

32-bit Kernel Mode, Supervisor Space (*ksseg*)

In Kernel mode, when *KX* = 0 in the *Status* register and the most-significant three bits of the 32-bit virtual address are 110_2, the *ksseg* virtual address space is selected; it is the current 2^{29}-byte (512-Mbyte) supervisor virtual space. The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

References to *ksseg* are mapped through the TLB.

32-bit Kernel Mode, Kernel Space 3 (*kseg3*)

In Kernel mode, when *KX* = 0 in the *Status* register and the most-significant three bits of the 32-bit virtual address are 111_2, the *kseg3* virtual address space is selected; it is the current 2^{29}-byte (512-Mbyte) kernel virtual space. The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

References to *kseg3* are mapped through the TLB.

64-bit Kernel Mode, User Space (*xkuseg*)

In Kernel mode, when *KX* = 1 in the *Status* register and bits 63:62 of the 64-bit virtual address are 00_2, selection of the *xkuseg* virtual address space is dependent upon the *UX* and *ERL* bits.

- if *UX* = 1 and *ERL* = 0, the entire space from 0x0000 0000 0000 0000 through 0000 0FFF FFFF FFFF (16 Tbytes) is selected.
- If *UX* = 0 or *ERL* = 1, the address space 0x0000 0000 0000 0000 through 0000 0000 7FFF FFFF (2 Gbytes) is selected. Addressing the space ranging from 0000 0000 8000 0000 through 0000 0FFF FFFF FFFF will cause an address error. Moreover, if *ERL*=1, the selected 2-Gbyte address space becomes unmapped and uncached.

The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

64-bit Kernel Mode, Current Supervisor Space (*xksseg*)

In Kernel mode, when *KX* = 1 in the *Status* register and bits 63:62 of the 64-bit virtual address are 01_2, selection of the *xksseg* virtual address space is dependent upon the *SX* bit.

- if *SX* = 1, the entire space from 0x4000 0000 0000 0000 through 4000 0FFF FFFF FFFF (16 Tbytes) is selected.
- If *SX* = 0, access to any address in the space ranging from 0x4000 0000 0000 0000 through 4000 0FFF FFFF FFFF causes an address error.

The virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address.

64-bit Kernel Mode, Physical Spaces (*xkphys*)

In Kernel mode, when *KX* = 1 in the *Status* register and bits 63:62 of the 64-bit virtual address are 10_2, the *xkphys* virtual address space is selected; it is a set of eight kernel physical spaces. Each kernel physical space contains either one or four 2^{40}-byte physical pages.

References to this space are not mapped; the physical address selected is taken directly from bits 39:0 of the virtual address. Bits 61:59 of the virtual address specify the *cache algorithm*, described in Chapter 4. If the cache algorithm is either uncached or uncached accelerated (values of 2 or 7) the space contains four physical pages; access to addresses whose bits 56:40 are not equal to 0 cause an Address Error exception. Address bits 58:57 carry the *uncached attribute* (described in Chapter 6), and are not checked for address errors.

If the cache algorithm is neither uncached nor uncached accelerated, the space contains a single physical page, as on the R4400 processor. In this case, access to addresses whose bits 58:40 are not equal to a zero cause an Address Error exception, as shown in Figure 16-4.

0X BFFFFFFF FFFFFFFF	0X BE000100 00000000	Address Error
0X BE0000FF FFFFFFFF	0X BE000000 00000000	Uncached Accelerated
0X BDFFFFFF FFFFFFFF	0X BC000100 00000000	Address Error
0X BC0000FF FFFFFFFF	0X BC000000 00000000	Uncached Accelerated
0X BBFFFFFF FFFFFFFF	0X BA000100 00000000	Address Error
0X BA0000FF FFFFFFFF	0X BA000000 00000000	Uncached Accelerated
0X B9FFFFFF FFFFFFFF	0X B8000100 00000000	Address Error
0X B80000FF FFFFFFFF	0X B8000000 00000000	Uncached Accelerated
0X B7FFFFFF FFFFFFFF	0X B0000100 00000000	Address Error
0X B00000FF FFFFFFFF	0X B0000000 00000000	Reserved*
0X AFFFFFFF FFFFFFFF	0X A8000100 00000000	Address Error
0X A80000FF FFFFFFFF	0X A8000000 00000000	Cacheable Exclusive Write
0X A7FFFFFF FFFFFFFF	0X A0000100 00000000	Address Error
0X A00000FF FFFFFFFF	0X A0000000 00000000	Cacheable Exclusive

0X 9FFFFFFF FFFFFFFF	0X 98000100 00000000	Address Error
0X 980000FF FFFFFFFF	0X 98000000 00000000	Cacheable Noncoherent
0X 97FFFFFF FFFFFFFF	0X 96000100 00000000	Address Error
0X 960000FF FFFFFFFF	0X 96000000 00000000	Uncached
0X 95FFFFFF FFFFFFFF	0X 94000100 00000000	Address Error
0X 940000FF FFFFFFFF	0X 94000000 00000000	Uncached
0X 93FFFFFF FFFFFFFF	0X 92000100 00000000	Address Error
0X 920000FF FFFFFFFF	0X 92000000 00000000	Uncached
0X 91FFFFFF FFFFFFFF	0X 90000100 00000000	Address Error
0X 900000FF FFFFFFFF	0X 90000000 00000000	Uncached
0X 8FFFFFFF FFFFFFFF	0X 88000100 00000000	Address Error
0X 880000FF FFFFFFFF	0X 88000000 00000000	Reserved*
0X 87FFFFFF FFFFFFFF	0X 80000100 00000000	Address Error
0X 800000FF FFFFFFFF	0X 80000000 00000000	Reserved*

* Accessing a reserved space results in undefined behavior.

Figure 16-4 xkphys Virtual Address Space

64-bit Kernel Mode, Kernel Space (*xkseg*)

In Kernel mode, when *KX* = 1 in the *Status* register and bits 63:62 of the 64-bit virtual address are 11_2, the address space selected is one of the following:

- kernel virtual space, *xkseg*, the current kernel virtual space; the virtual address is extended with the contents of the 8-bit ASID field to form a unique virtual address
- one of the four 32-bit kernel mode compatibility spaces (described below).

64-bit Kernel Mode, Compatibility Spaces (*ckseg1:0, cksseg, ckseg3*)

In Kernel mode, when *KX* = 1 in the *Status* register, bits 63:62 of the 64-bit virtual address are 11_2, and bits 61:31 of the virtual address equal –1, the lower two bytes of address, as shown in Figure 16-3, select one of the following 512-Mbyte compatibility spaces.

- *ckseg0*. This 64-bit virtual address space is an unmapped region, compatible with the 32-bit address model *kseg0*. The *K0* field of the *Config* register controls cacheability and coherency.
- *ckseg1*. This 64-bit virtual address space is an unmapped and uncached region, compatible with the 32-bit address model *kseg1*.
- *cksseg*. This 64-bit virtual address space is the current supervisor virtual space, compatible with the 32-bit address model *ksseg*.
- *ckseg3*. This 64-bit virtual address space is kernel virtual space, compatible with the 32-bit address model *kseg3*.

Address Space Access Privilege Differences Between the R4400 and R1000

In the R4400, the 64-bit Supervisor mode can access the entire *xsuseg* space, and the 64-bit Kernel mode can access the entire *xksseg* and *xkuseg* spaces. Access privileges in the R10000 are also dependent on the *UX* and *SX* bits:

- Access to the 64-bit user space in 64-bit Supervisor or Kernel mode (*xsuseg* or *xkuseg*) is controlled by the *UX* bit. If *UX*=0, the 64-bit Supervisor and Kernel modes can only access the 32-bit user space (*suseg* or *kuseg*).
- Access to the 64-bit supervisor space in Kernel mode (*xksseg*) is controlled by the *SX* bit. If *SX*=0, the 64-bit Kernel mode can only access the 32-bit supervisor space (*ksseg*).

An Address Error exception is taken on an illegal access.

The R10000 processor implements the same access privileges for 32-bit processor modes as in the R4400. The Table 16-2 summarizes the access privileges for all processor modes in the R10000 processor.

Table 16-2 Access Privileges for User, Supervisor and Kernel Mode Operations

<table>
<tr><th rowspan="2">64-bit Virtual Address</th><th colspan="3">32-bit Mode</th><th colspan="4">64-bit Mode</th></tr>
<tr><th>User *</th><th>Supervisor</th><th>Kernel</th><th>User</th><th>Supervisor</th><th>Kernel & ERL=0</th><th>Kernel & ERL=1</th></tr>
<tr><td>FFFFFFFF E0000000 TO FFFFFFFF FFFFFFFF</td><td rowspan="11">AddrErr</td><td>AddrErr</td><td rowspan="4">OK</td><td rowspan="10">AddrErr</td><td>AddrErr</td><td rowspan="4">OK</td><td rowspan="4">OK</td></tr>
<tr><td>FFFFFFFF C0000000 TO FFFFFFFF DFFFFFFF</td><td>OK</td><td>OK</td></tr>
<tr><td>FFFFFFFF A0000000 TO FFFFFFFF BFFFFFFF</td><td rowspan="9">AddrErr</td><td rowspan="6">AddrErr</td></tr>
<tr><td>FFFFFFFF 80000000 TO FFFFFFFF 9FFFFFFF</td></tr>
<tr><td>C0000FFF 00000000 TO FFFFFFFF 7FFFFFFF</td><td rowspan="7">AddrErr</td><td>AddrErr</td><td>AddrErr</td></tr>
<tr><td>C0000000 00000000 TO C0000FFE FFFFFFFF</td><td>OK</td><td>OK</td></tr>
<tr><td>80000000 00000000 TO BFFFFFFF FFFFFFFF</td><td>OK</td><td>OK</td></tr>
<tr><td>40001000 00000000 TO 7FFFFFFF FFFFFFFF</td><td>AddrErr</td><td>AddrErr</td></tr>
<tr><td>40000000 00000000 TO 40000FFF FFFFFFFF</td><td>OK</td><td>AddrErr if SX=0</td><td>AddrErr if SX=0</td></tr>
<tr><td>00001000 00000000 TO 3FFFFFFF FFFFFFFF</td><td>AddrErr</td><td>AddrErr</td><td>AddrErr</td></tr>
<tr><td>00000000 80000000 TO 00000FFF FFFFFFFF</td><td rowspan="2">OK</td><td>AddrErr if UX=0</td><td>AddrErr if UX=0</td><td>AddrErr</td></tr>
<tr><td>00000000 00000000 TO 00000000 7FFFFFFF</td><td>OK</td><td>OK</td><td>OK</td><td>OK</td><td>OK</td><td>OK</td></tr>
</table>

* For data references, the upper 32 bits of the virtual addresses are cleared before checking access privilege and TLB translation.

16.3 Virtual Address Translation

Programs can operate using either physical or virtual memory addresses:

- physical addresses correspond to hardware locations in main memory
- virtual addresses are logical values only, and do not correspond to fixed hardware locations

Virtual addresses must first be translated (finding the physical address at which the virtual address points) before main memory can be accessed. This translation is essential for multitasking computer systems, because it allows the operating system to load programs anywhere in main memory independent of the logical addresses used by the programs.

This translation also implements a memory protection scheme, which limits the amount of memory each program may access. The scheme prevents programs from interfering with the memory used by other programs or the operating system.

Cache Pages

Translated virtual addresses retrieve data in blocks, which are called pages. In the R10000 processor, the size of each page may be selected from a range that runs from 4 Kbytes to 16 Mbytes inclusive, in increments of 4 (that is, 4 Kbytes, 16 Kbytes, 64 Kbytes, etc.).

The virtual address bits which select a page (and thus are translated) are called the *page address*. The lower bits which select a byte within the selected page are called the *offset* and are not translated. The number of offset bits varies from 12 to 24 bits, depending on the page size.

Cache Page Size Encodings

Page size is defined in each TLB entry's *PageMask* field. This field is loaded or read using the *PageMask* register, as described in Chapter 14.

Each entry translates a pair of physical pages. The low bit of the virtual address page is not compared, because it is used to select between these two physical pages.

Using the TLB

Translations are maintained by the operating system, using page tables in memory. A subset of these translations are loaded into a hardware buffer called the translation-lookaside buffer or TLB. The contents of this buffer are maintained by the operating system; if an instruction needs a translation which is not already in the buffer, an exception is taken so the operating system can compute and load the needed translation. If all the necessary translations are present, the program is executed without any delays.

The TLB contains 64 entries, each of which maps a pair of virtual pages. Formats of TLB entries are shown in Figure 16-5.

Cache Algorithm Field

The *Cache Algorithm* fields of the TLB, *EntryLo0*, *EntryLo1*, and *Config* registers indicate how data is cached. Cache algorithms are described in Chapter 4.

Format of a TLB Entry

Figure 16-5 shows the TLB entry formats for both 32- and 64-bit modes. Each field of an entry has a corresponding field in the *EntryHi*, *EntryLo0*, *EntryLo1*, or *PageMask* registers, as shown in Chapter 14; for example the *PFN* and uncached attribute (*UC*) fields of the TLB entry are also held in the *EntryLo* registers.

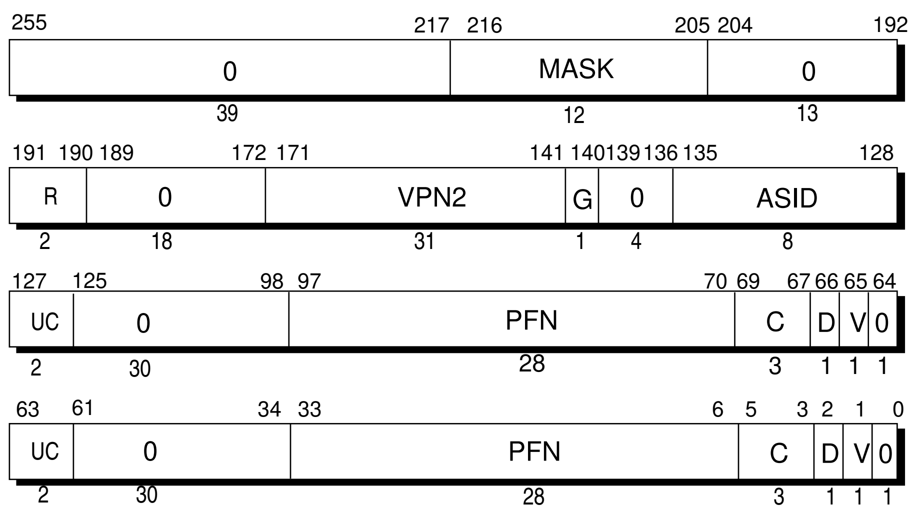

Figure 16-5 Format of a TLB Entry

Address Translation

Because a 64-bit address is unnecessarily large, only the low 44 address bits are translated. The high two virtual address bits (bits 63:62) select between user, supervisor, and kernel address spaces. The intermediate address bits (61:44) must either be all zeros or all ones, depending on the address region. The TLB does not include virtual address bits 61:59, because these are decoded only in the *xkphys* region, which is unmapped.

For data cache accesses, the joint TLB (JTLB) translates addresses from the address calculate unit. For instruction accesses, the JTLB translates the PC address if it misses in the instruction TLB (ITLB). That entry is copied into the ITLB for subsequent accesses. The ITLB is transparent to system software.

Address Space Identification (*ASID*)

Each independent task, or *process*, has a separate address space, assigned a unique 8-bit Address Space Identifier (ASID). This identifier is stored with each TLB entry to distinguish between entries loaded for different processes. The ASID allows the processor to move from one process to another (called a context switch) without having to invalidate TLB entries.

The processor's current ASID is stored in the low 8 bits of the *EntryHi* register. These bits are also used to load the *ASID* field of an entry during TLB refill.

The *ASID* field of each TLB entry is compared to the *EntryHi* register; if the ASIDs are equal or if the entry is global (see below), this TLB entry may be used to translate virtual addresses. The ASID comparison is performed only when a new value is loaded into the *EntryHi* register; the one-bit result of the match is stored in a static Enable latch. (This bit is set whenever a new entry is loaded.)

Global Processes (*G*)

A translation may be defined as *global* so that it can be shared by all processes. This *G* bit is set in the TLB entry and enables the entry independent of its ASID value.

Avoiding TLB Conflict

Setting the *TS* bit in the *Status* register indicates an entry being presented to the TLB matches more than one virtual page entry in the TLB. Any TLB entries that allow multiple matches, even in the *Wired* area, are invalidated before the new entry can be written into the TLB. This prevents multiple matches during address translation.