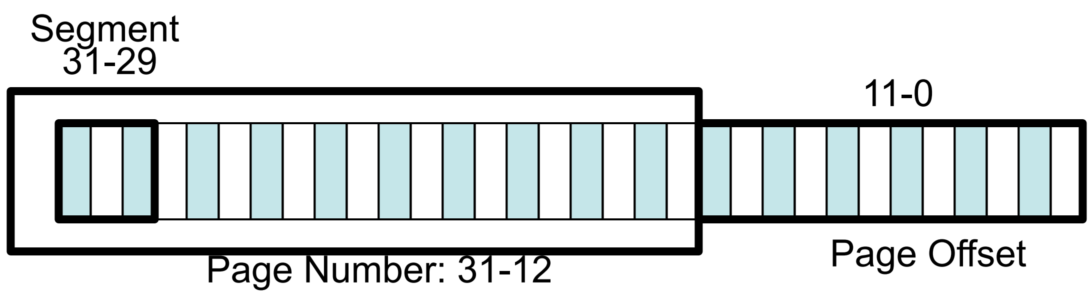
VM Case Study: MIPS R2000

- Topics
  - MIPS (software managed VM)
- Learning Objectives:
  - Describe the MIPS MMU support.
  - Be prepared to undertake assignment 3.

The MIPS R2000

- Introduced in 1987.
- One of the first commercial Reduced Instruction Set Computers (RISC), preceded by the Cray-1 and other supercomputers.
- It has a simple, elegant architecture that has made it a popular target for homework assignments in computer architecture and operating systems.
- Virtual Address Format:
  - Memory addressed in bytes.
  - Bits 31-29 are used to partition the virtual address space into four segments:
    - KUSEG: 0??: user-mapped cached
    - KSEG0: 100: kernel unmapped cached
    - KSEG1: 101: kernel unmapped uncached
    - KSEG2: 11?: kernel mapped cached
  - Each page is 2^{12} = 4096 bytes
  - A virtual address is translated into a physical address by translating the 20-bit virtual page number into a physical page number.



MIPS R2000 TLB Structure

- Associativity: Fully Associative
- Replacement policy: Random
- Size: 64 entries (56 random, 8 "wired")
- Each TLB entry is 64 bits.
- The **TLB_EntryHi** and **TLB_EntryLow** registers are used to read and write TLB entries, defining the contents of TLB entries.

The **TLB_EntryHI** register
- Virtual Page number (20 bits)
- Address Space ID (6 bits)
- reserved (6 bits)

The **TLB_EntryLo** register
- Page Frame Number (20 bits)
- Mode: (4 bits):
  - Non-cacheable, Dirty/write-protect, Valid, and Global
- reserved (8 bits)

Virtual Page Number 63…44	ASID 43…38	reserved 37…32

Physical Page Number 31 … 12	Mode 11 … 8	Reserved 7 … 0

TLB Instructions

- **TLBR**: Read the TLB entry specified by the **index register** into TLB_EntryHi and TLB_EntryLo.
- **TLBWI**: Write the TLB entry specified by the **index register** with the contents of TLB EntryHi and TLB_EntryLo.
- **TLBWR**: Write the TLB entry specified by the **random register** with the contents of TLB_EntryHi and TLB_EntryLo.
- **TLBP**: Probe the TLB for an entry matching the virtual page number, PID, and Context bits that are in TLB_EntryHi, observing the Global mode bit.
  - Sets the P bit if there are no matching entries. Undefined if there are multiple matching entries.

TLB Registers (1)

- The **TLB Index register**: indicates which TLB entry is being manipulated.
  - Index field (6 bits)
  - The P bit indicates the failure of a **TLB Probe** operation.

P	unused	index	Unused
31	30 … 14	13 … 8	7 … 0

- The **TLB Context register**: gives you faulting address and user-page table address.
  - PTEBase: (11 bits) upper bits of the user page table base address. Set by the OS.
  - Bad VPN: (19 bits) Set by the hardware on a TLB miss to the page number (bits 30..12) of the failing virtual address

PTEBase	Bad VPN	0 0
31 … 21	20 .. 2	1-0

TLB Registers (2)

- The **Random** register provides a random numb [8…13] that is used by the TLBWR (TLB Write Random) instruction.
  - This is how random replacement can be implemented.
  - The register is readable, although reading the register is not necessary for TLB management.

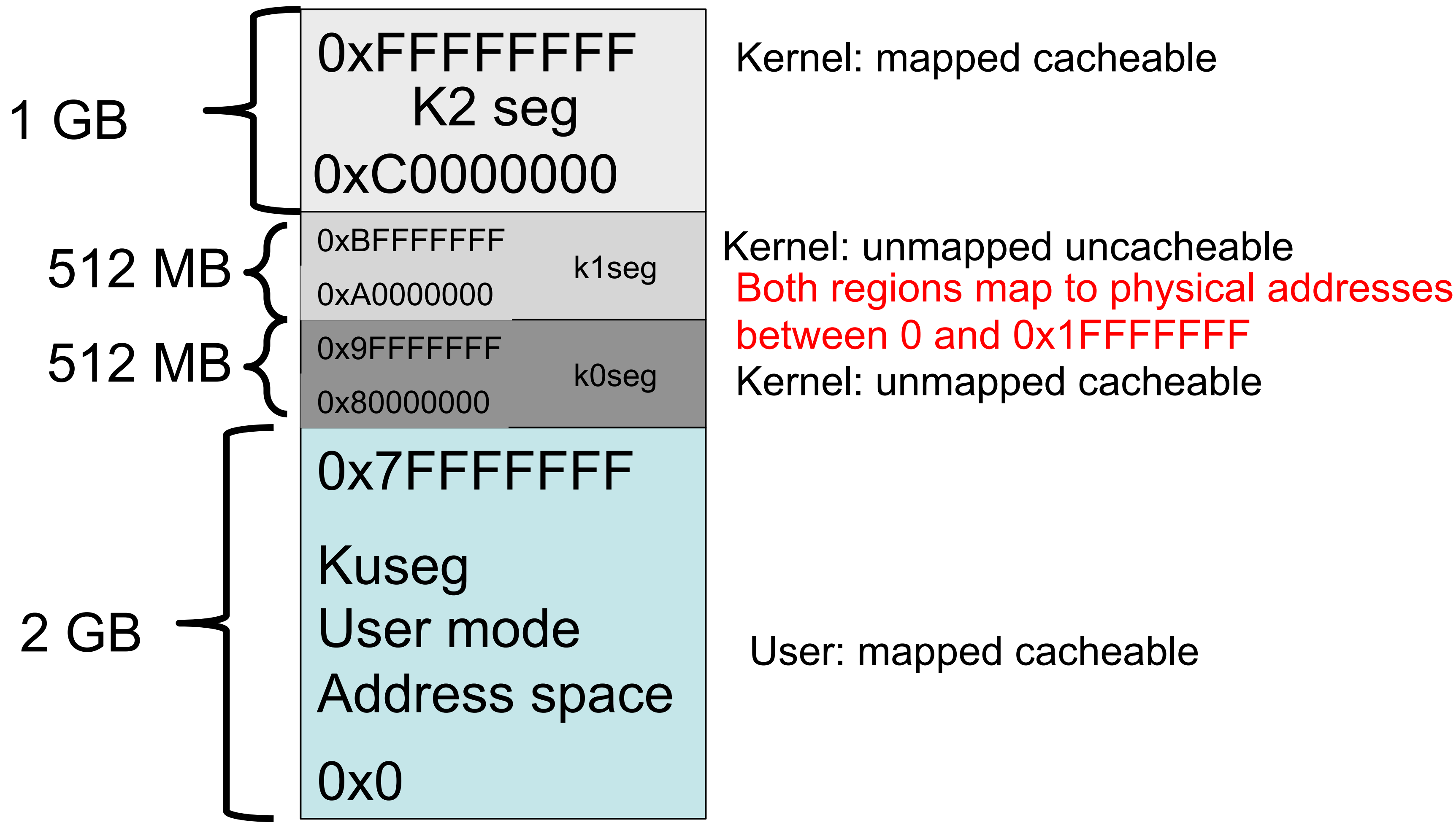
unused	random	Unused
31 … 14	13 … 8	7 … 0

Mach 3.0 TLB Miss handler

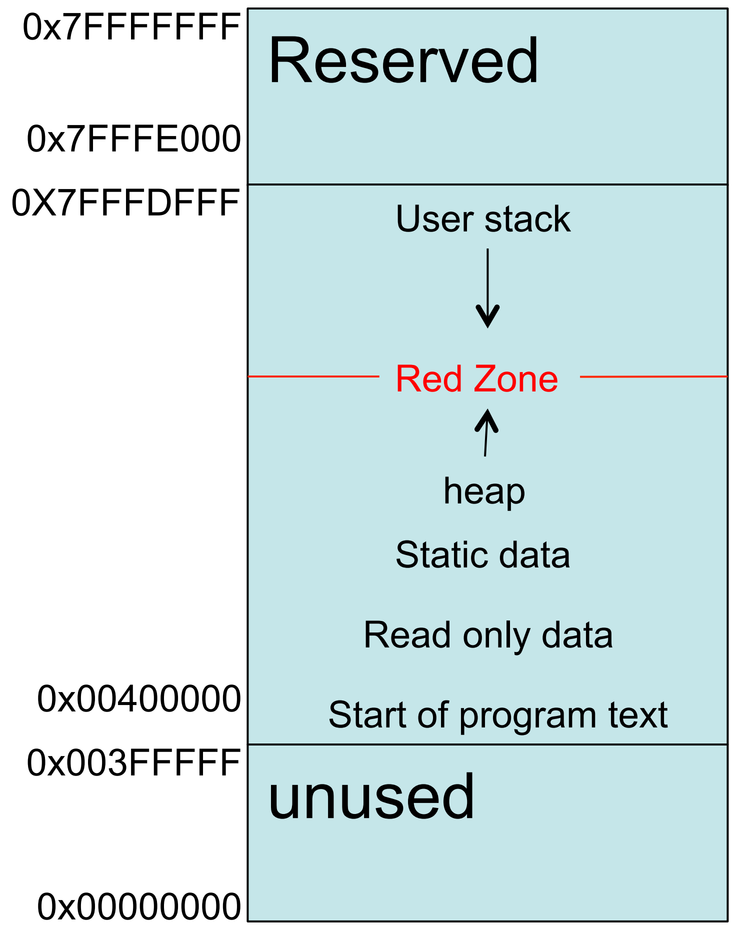
```
NESTED(TRAP_tlbn_umiss, 0, k1)
        mfc0    k0, cO_tlbcxt   (loads context reg into k0)
        mfc0    k1, cp_epc      (load exception PC into k1)
        lw      k0, 0(k0)       (k0 is address of PTE; read it)
        nop                     (load delay)
        mtc0    k0, c0_tlblo    (load entry from PT into TLBlo)
        tlbwr                   (write new entry into the TLB)
        j       k1              (jump to the faulting inst)
        rfe                     (branch slot; back to user mode)
```

- Because page tables are large, they are kept in system virtual memory (k2seg).
- Most of the time, this miss handler won't generate any exceptions, but it can …

Virtual Memory Map



Typical user Address Space Layout



MIPS R2000 Example

Address Translation		**KUSeg Page Table**		**K2Seg Page Table**	
Virtual Addr	Physical Addr	VPN	Translation	VPN	Translation
0x00400000		0	Invalid	C0000	00598
0x00402ADC		…		C0001	006C8
0x004010B0		003FF	Invalid	…	
0x07FFFFF0		00400	0093F	C00FE	Invalid
0x00000128		00401	00940	C00FF	00123
0x80030284		00402	00941	C0100	00987
0xC0001A2F		00403	008F3		
0xB00FF00D		…			
0xDEADBEEF		00500	00CDA		
		00501	00EF9		
		…			
		07FFE	00BC2		
		07FFF	00731		

MIPS R2000 Example

Address Translation		KUSeg Page Table		K2Seg Page Table	
Virtual Addr	Physical Addr	VPN	Translation	VPN	Translation
0x00400000	0x0093F000	0	Invalid	C0000	00598
0x00402ADC	0x00941ADC	…		C0001	006C8
0x004010B0	0x009400B0	003FF	Invalid	…	
0x07FFFFF0	0x00731FF0	00400	0093F	C00FE	Invalid
0x00000128	FAULT	00401	00940	C00FF	00123
0x80030284	0x0030284	00402	00941	C0100	00987
0xC0001A2F	0x006C8A2F	00403	008F3		
0xB00FF00D	0x100FF00D	…			
0xDEADBEEF	FAULT	00500	00CDA		
		00501	00EF9		
		…			
		07FFE	00BC2		
		07FFF	00731		

MIPS R2000 Recap

- The user segment occupies half of the virtual address space.
- System memory is organized into three segments.
- Each segment defines how memory in the segment is accessed.
- System memory can only be accessed when the processor is executing in system mode.
- The virtual page size is 4 KB.
- Q1: How can you implement shared memory between two user processes?
- Q2: How can you implement shared memory between the user and kernel?
- Q3: What kind of fragmentation might you get?
- Q4: What problems do the page tables pose?

MIPS R2000 Recap

- The user segment occupies half of the virtual address space and defines user and system memory.
- System memory is organized into three segments.
- Each segment defines how memory in the segment is accessed.
- System memory can only be accessed when the processor is executing in system mode.
- The virtual page size is 4 KB.
- Q1: How can you implement shared memory between two user processes?
  - Copy PTEs (two page tables contain identical PTEs)
- Q2: How can you implement shared memory between the user and kernel?
  - Kernel can access user memory (in lower portion of address space)
- Q3: What kind of fragmentation might you get?
  - No external (fixed size page); some internal.
- Q4: What problems do the page tables pose?
  - Too big! Requires too many memory references!