School of Computer Science & Engineering

COMP3891/9283 Extended Operating Systems

2026 T2 Week 07

Page Tables Revisited

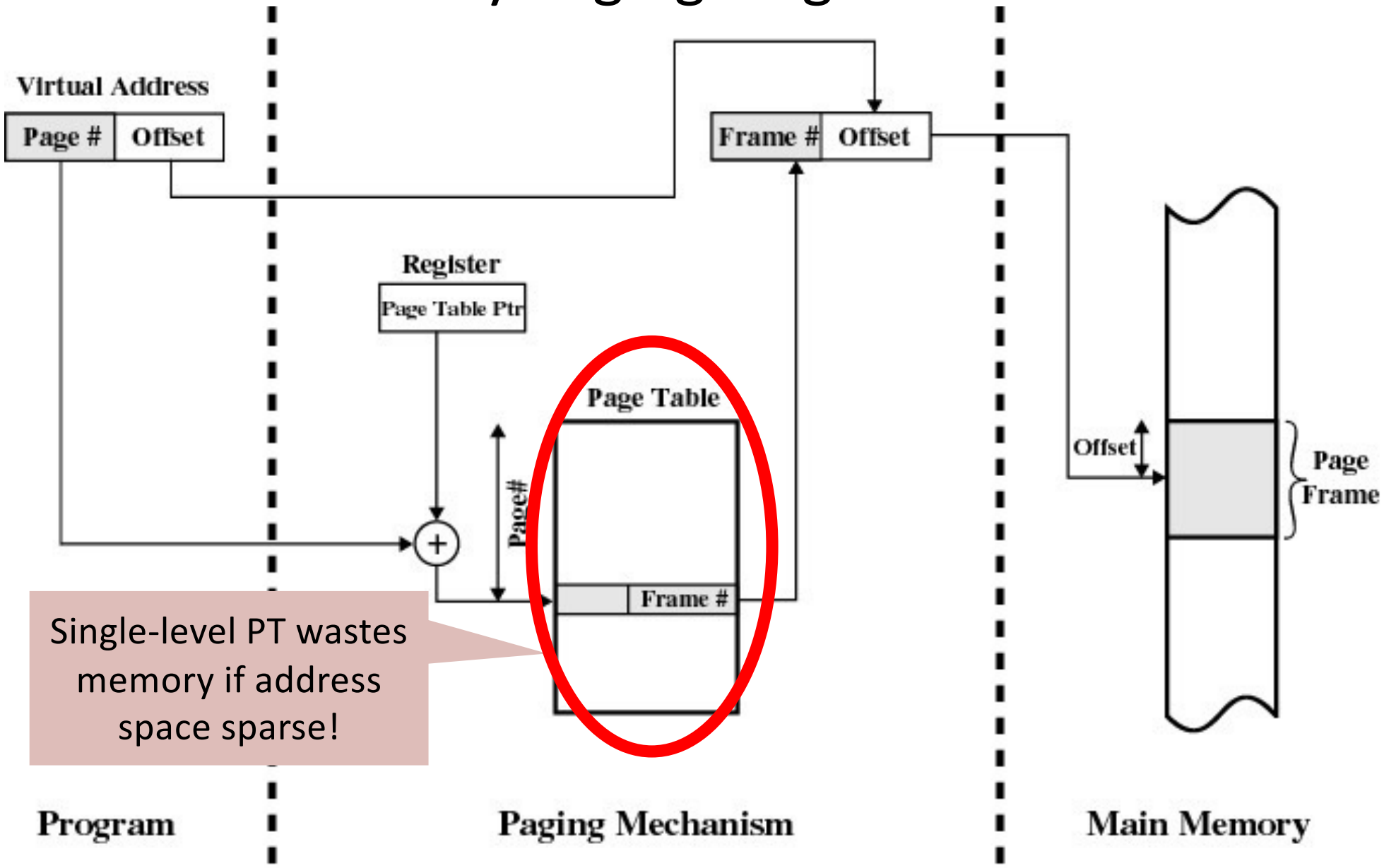
Gernot Heiser

Copyright Notice

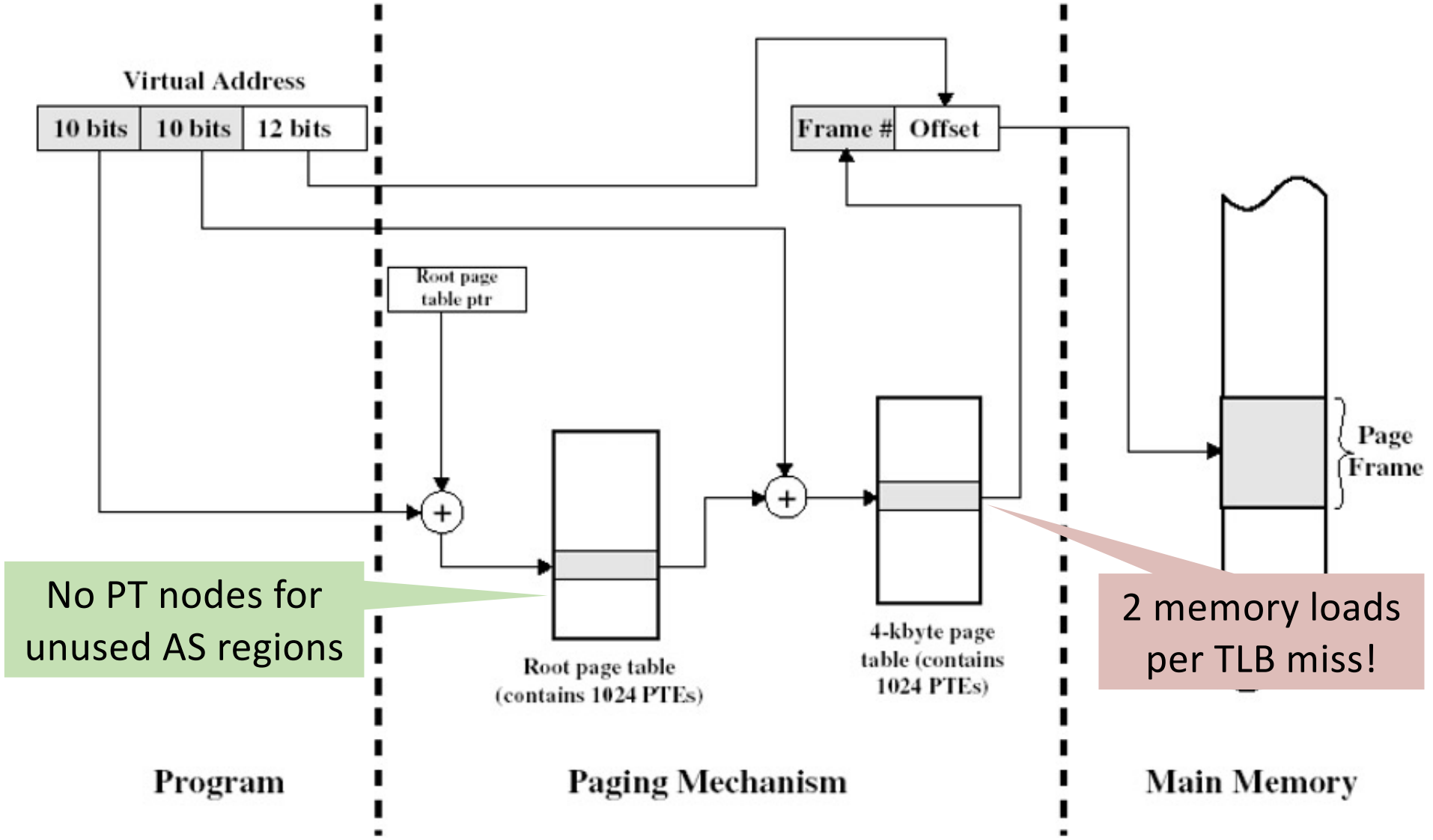
Learning Outcomes

- An understanding of virtual linear array page tables, and their use on the MIPS R3000.
- Exposure to alternative page table structures beyond multi-level and inverted page tables.

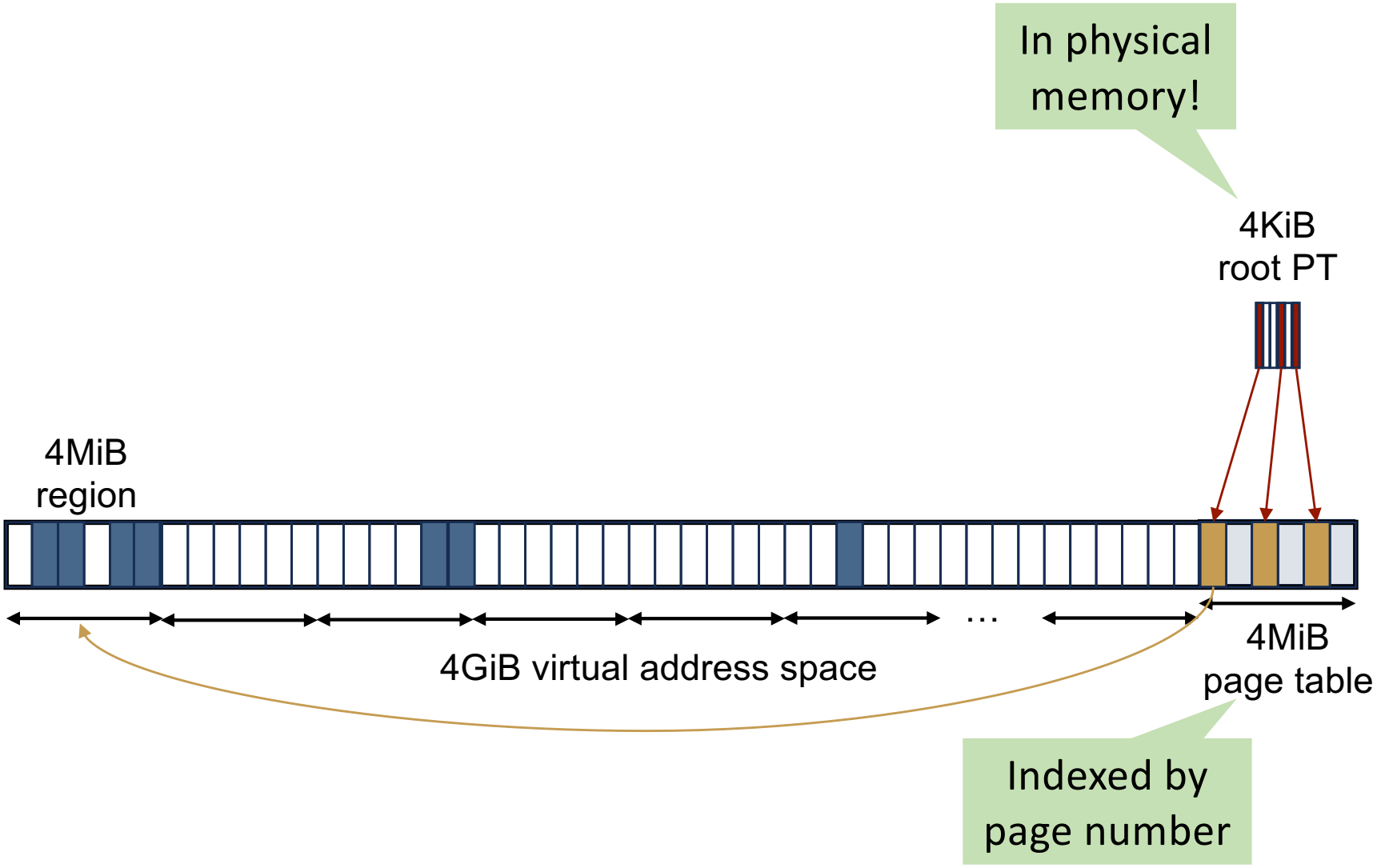
Virtual-Memory Paging: Logical View



Two-level Translation

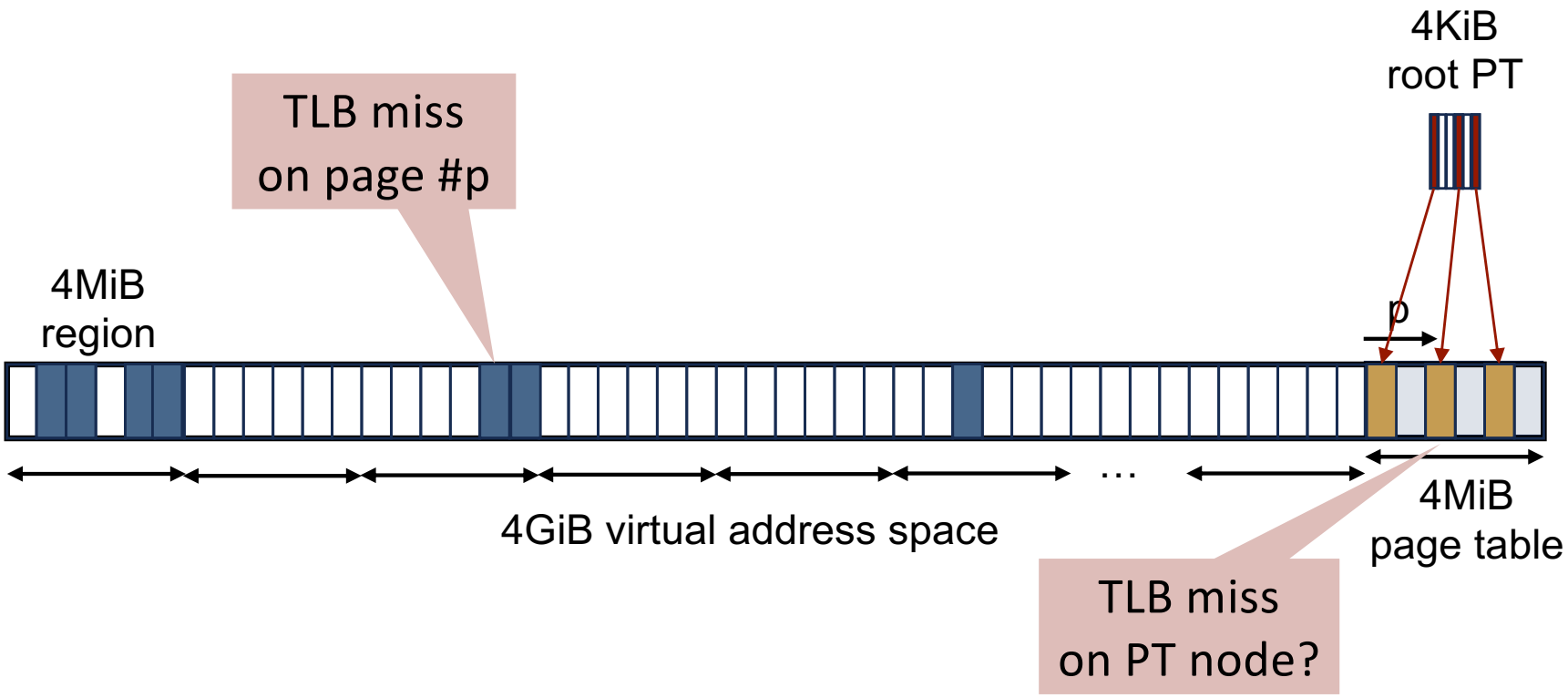


Virtual Linear Array Page Table (VLAPT)



Virtual Linear Array Page Table (VLAPT)

VLAPT generates nested faults!



MIPS R3000 TLB Refill

- R3000 has dedicated exception handler for TLB refill
- Can be optimised for TLB refill only
  - Does not need to check the exception type
  - Does not need to save any registers
    - It uses a specialised assembly routine that only uses k0 and k1.
  - Does not check if PTE exists (if using VLAPT)

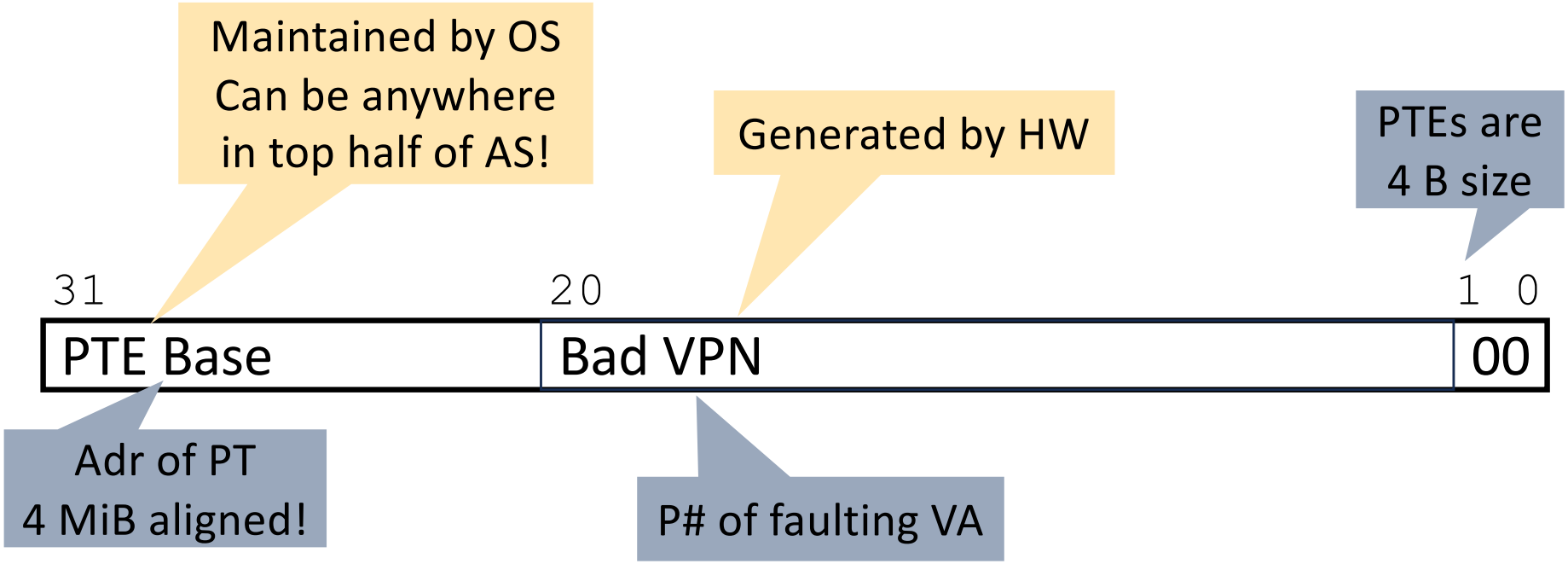
Can be made very fast!

MIPS R3000 TLB Refill Handler

What's this?

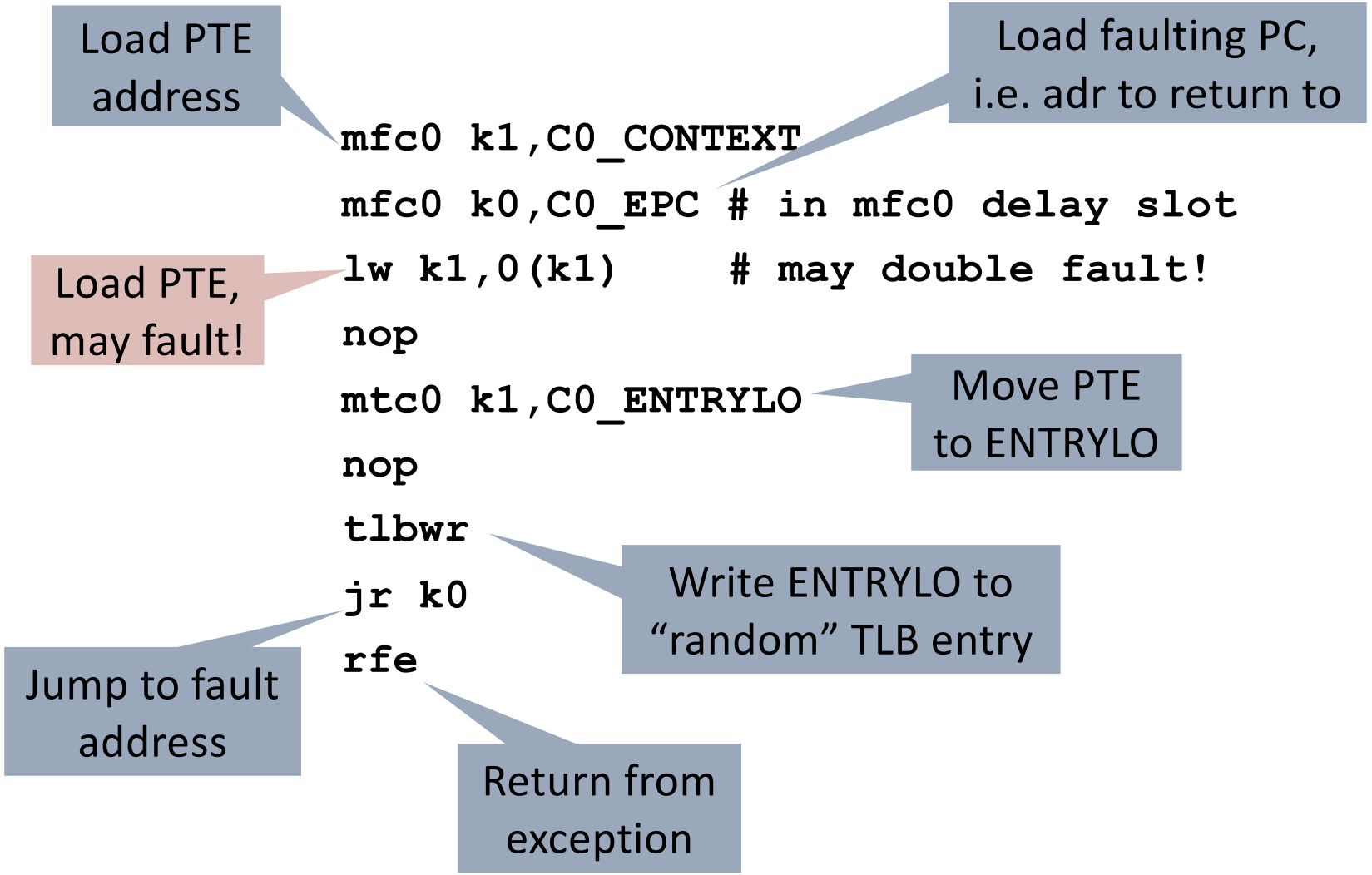
```
mfc0 k1,C0_CONTEXT
mfc0 k0,C0_EPC # in mfc0 delay slot
lw k1,0(k1)    # may double fault!
nop
mtc0 k1,C0_ENTRYLO
nop
tlbwr
jr k0
rfe
```

C0_CONTEXT Register



PTE_Base + Bad_VPN*4 = &PTE

MIPS R3000 TLB Refill Handler



MIPS R3000 TLB Refill Handler

```
mfc0 k1,C0_CONTEXT
mfc0 k0,C0_EPC # in mfc0 delay slot
lw k1,0(k1)    # may double fault!
nop
mtc0 k1,C0_ENTRYLO
nop
tlbwr
jr k0
rfe
```

Load PTE, may fault!

Software-loaded TLB!

- Nested faults go to separate ("general") exception handler
- Normal refill handler doesn't have to worry about this!

Software-Loaded TLB

TLB is fully under control of the OS!

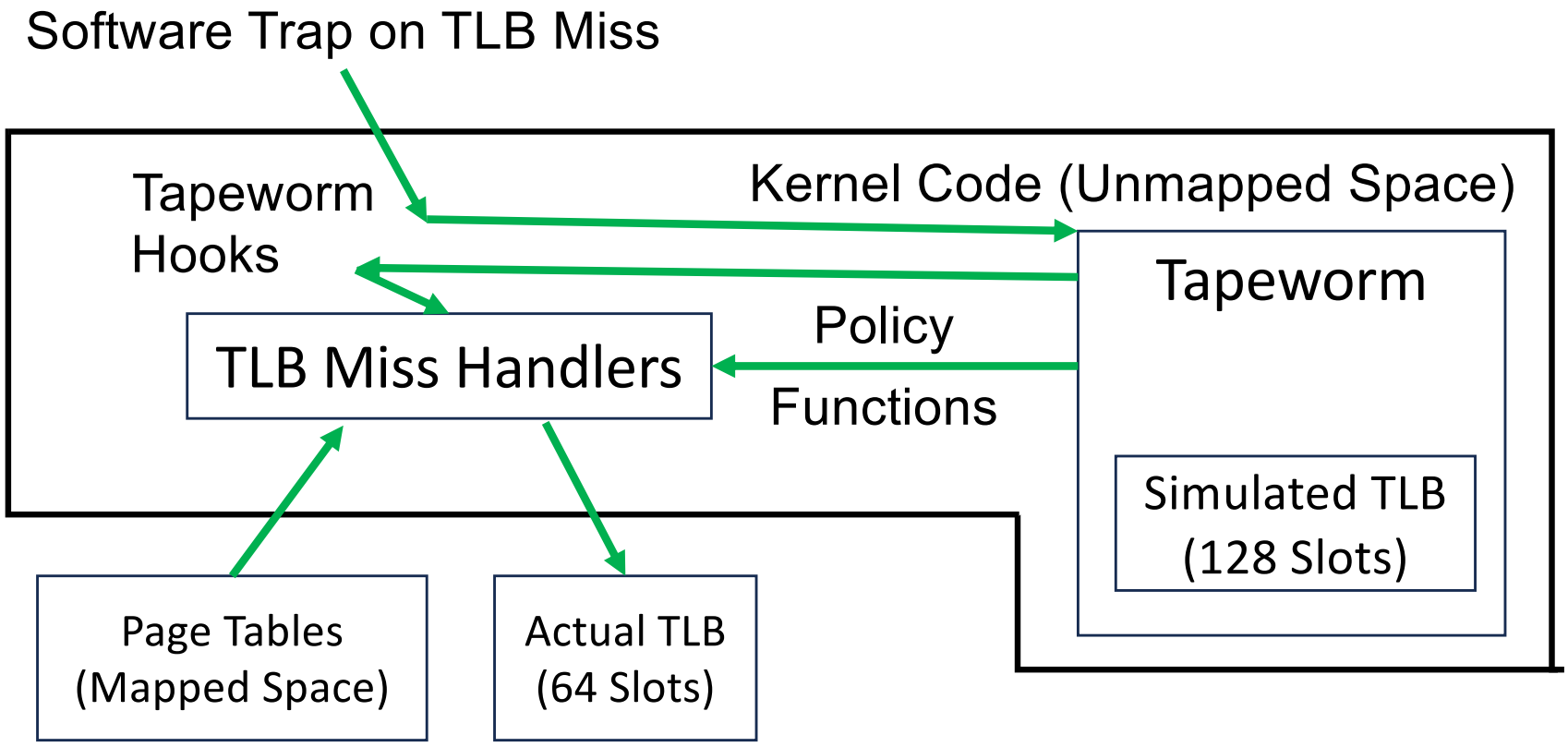
- Pros
  - Can simplify hardware design
  - Avoids imposing a specific page-table structure
- Cons
  - May have slower refill times than hardware-managed TLBs

Evaluation:
David Nagle, Richard Uhlig, Tim Stanley, Stuart Sechrest, Trevor Mudge & Richard Brown:
“Design Tradeoffs for Software-Managed TLBs”
Proceedings of the 20th Annual International Symposium on Computer Architecture (ISCA’93)

Trends at the time

- Operating systems
  - Moving functionality into user processes
  - Making greater use of virtual memory for mapping data structures held within the kernel.
- RAM is increasing
  - TLB capacity is relatively static
  - About to move to 64-bit address spaces
    – what PT structure will work?
- Statement:
  - Trends place greater stress upon the TLB by increasing miss rates and hence, decreasing overall system performance.
  - True/False? How to evaluate?

Tapeworm: In-OS TLB Simulator



Compare 3 OSes on MIPS R2000

- Ultrix – DEC's version of Unix
  - Traditional OS design
  - Most kernel data *physically* addressed (i.e. not in virtual memory)
  - Virtual linear array page table
- OSF/1: New commercial system
  - Most kernel data *virtually* addressed
  - 3-level page table
- Mach 3.0: "Microkernel" OS
  - Small kernel, most OS services in monolithic user-mode Unix server
  - Server data virtually addressed (obviously)
  - 3-level page table

Costs of Different TLB Miss Types (Cycles)

L1 User PTE

L1 Kernel PTE

L2 PTE

L3 PTE

Permission error

Unmapped page

Miss Type	Ultrix	OSF/1	Mach
L1U	16	20	20
L1K	333	355	294
L2	494	511	407
L3	– –	354	286
Modify	375	436	499
Invalid	336	277	267

What is the expected common case?

Measurement Results: TLB Misses

84%

98%

88%

System	Total Run Time (sec)	L1U	L1K	L2	L3	Invalid	Modify	Total
Ultrix	583	9,021,420	135,847	3,828	———	16,191	115	9,177,401
OSF/1	892	9,817,502	1,509,973	34,972	207,163	79,299	42,490	11,691,398
Mach3	975	21,466,165	1,682,722	352,713	556,264	165,849	125,409	24,349,121
Mach3+AFSin	1,371	30,123,212	2,493,283	330,803	690,441	168,429	127,245	33,933,413
Mach3+AFSOut	1,5[illegible]	31,611,047	2,712,979	1,042,527	987,648	168,128	127,505	36,649,834

Observations:

- **OS services in mapped memory increase TLB pressure!**

Measurement Results: Miss-Handling Costs

System	Total TLB Service Time (sec)	L1U	L1K	L2	L3	Invalid	Modify	% of Total Run Time
Ultrix	11.82	8.66	2.71	0.11	———	0.33	0.00	2.03%
OSF/1	51.85	11.78	32.16	1.07	4.40	1.32	1.11	5.81%
Mach3	80.01	25.76	29.68	8.61	9.55	2.66	3.75	8.21%
Mach3+AFSIn	106.56	36.15	43.98	8.08	11.85	2.70	3.81	7.77%
Mach3+AFSOut	134.71	37.93	47.86	25.46	16.95	2.69	3.82	8.86%

Observations:

- **OS services in mapped memory increase TLB pressure!**
- **Results in many nested faults, expensive on SW-Loaded TLB**

HW Support: L2/L1K Miss Handler

Type of PTE Miss	Counts	Previous Total Cost from Table 6 (sec)	New Total Cost (sec)	Time Saved (sec)
Mach3+AFSin				
L1U	30,123,212	36.15	36.15	0.00
L2	330,803	8.08	0.79	7.29
L1K	2,493,283	43.98	2.99	40.99
L3	690,441	11.85	11.85	0.00
Modify	127,245	3.81	3.81	0.00
Invalid	168,429	2.70	2.70	0.00
Total	33,933,413	106.56	58.29	48.28

How Are Things Done Today?

- Hierarchical PTs almost universal
  - 5–6 levels!
  - Only perform through aggressive caching
- Why?

The burden of legacy!
Intel tried (and failed) with Itanium…

- Multi-level TLBs
- Caching PTEs
- …