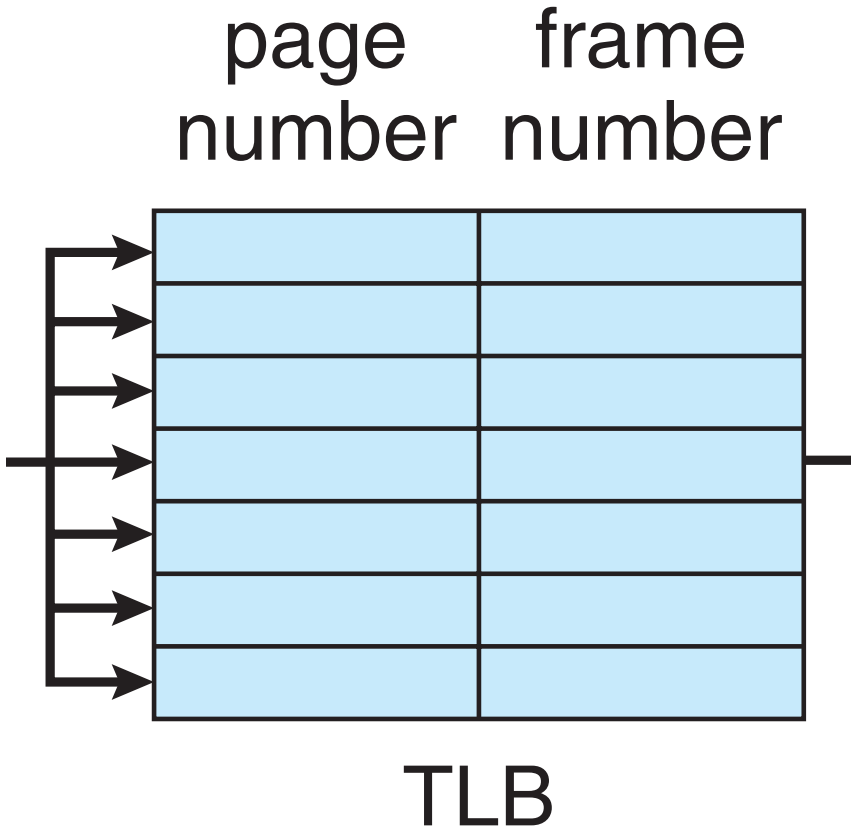


CS 250

OPERATING SYSTEMS

Lecture 9

TLB

Instructor

Dr. Dhiman Saha

Image Source: Google

Space overheads — Issue 1

Storing PT in memory wastes valuable memory space.

> Factor 2 slow down — Issue 2

For every memory reference, paging requires us to perform one extra memory reference in order to first fetch the translation from the page table

Space overheads — Issue 1

Storing PT in memory wastes valuable memory space.

> Factor 2 slow down — Issue 2

For every memory reference, paging requires us to perform one extra memory reference in order to first fetch the translation from the page table

Attempt #6: Translation-lookaside Buffer

Faster Paging

How to speed-up Address Translation?

- How can we speed up address translation?
- Avoid the **extra memory reference** that paging seems to require
- What hardware support is required?
- What OS involvement is needed?

TLB Translation-lookaside buffer

- Hardware support
- Part of the chips memory-management unit
- Basically, a **hardware cache** of popular virtual-to-physical address translations
- A.k.a address-translation cache

Idea

The hardware first checks the TLB to see if the desired translation is held therein.

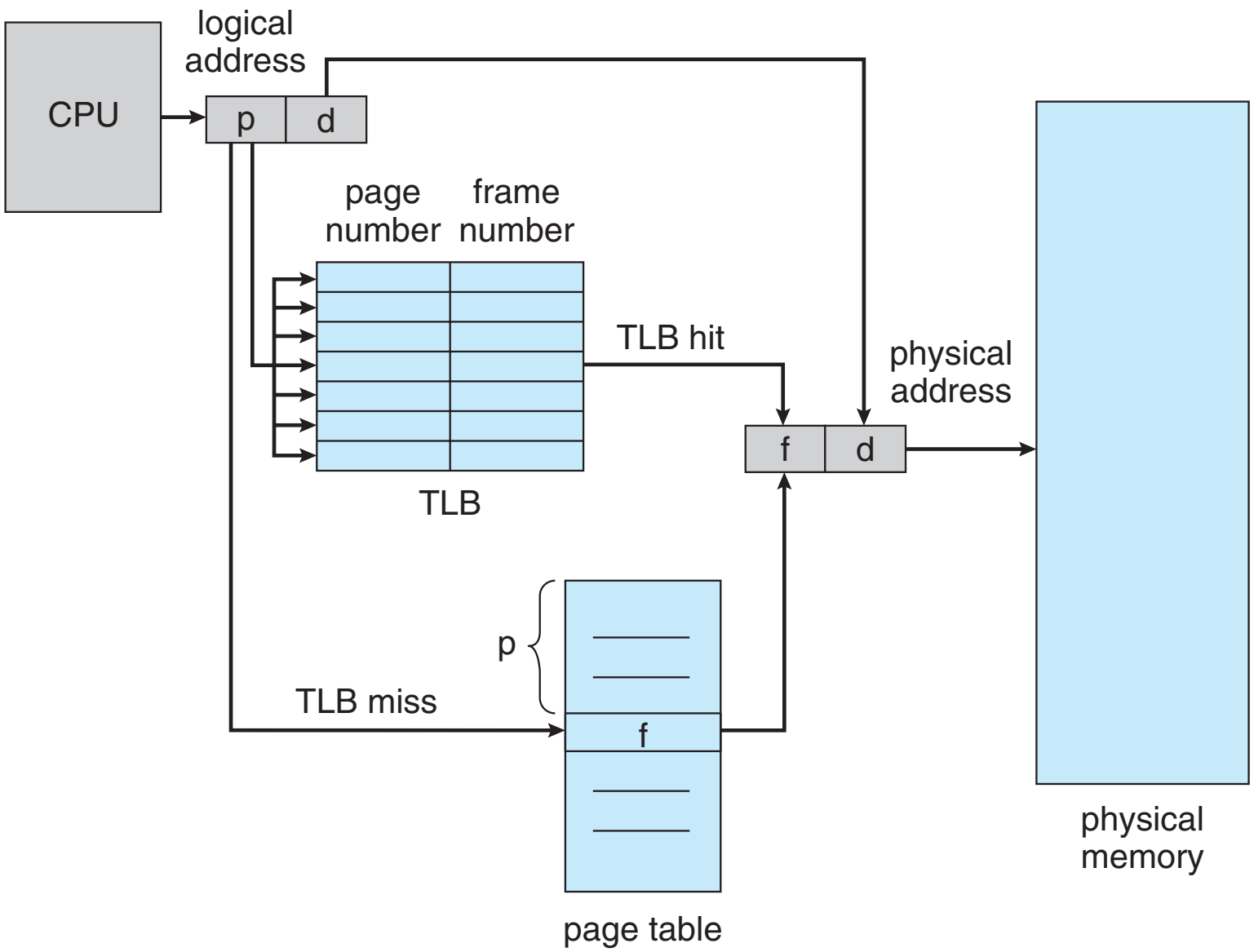
TLB Translation-lookaside buffer

- Hardware support
- Part of the chips memory-management unit
- Basically, a **hardware cache** of popular virtual-to-physical address translations
- A.k.a address-translation cache

Idea

The hardware first checks the TLB to see if the desired translation is held therein.

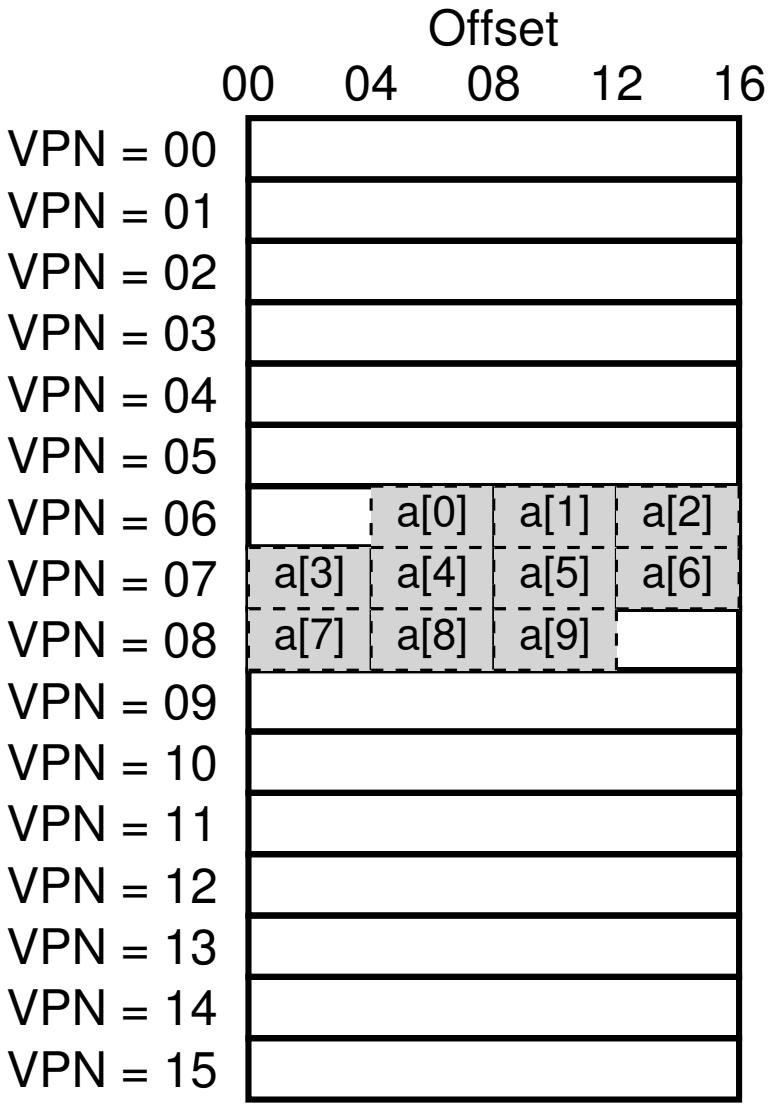
Paging hardware with TLB



HW Managed TLB Control Flow Algorithm

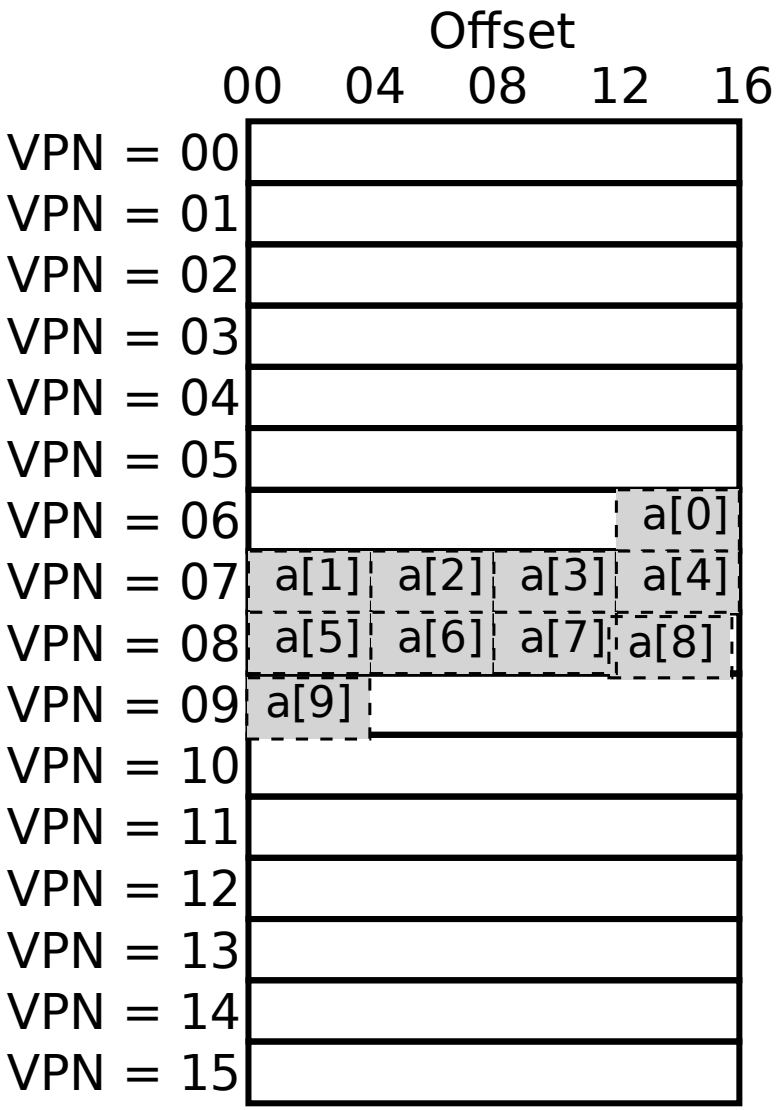
```
1   VPN = (VirtualAddress & VPN_MASK) >> SHIFT
2   (Success, TlbEntry) = TLB_Lookup(VPN)
3   if (Success == True)    // TLB Hit
4       if (CanAccess(TlbEntry.ProtectBits) == True)
5           Offset   = VirtualAddress & OFFSET_MASK
6           PhysAddr = (TlbEntry.PFN << SHIFT) | Offset
7           Register = AccessMemory(PhysAddr)
8       else
9           RaiseException(PROTECTION_FAULT)
10  else                    // TLB Miss
11      PTEAddr = PTBR + (VPN * sizeof(PTE))
12      PTE = AccessMemory(PTEAddr)
13      if (PTE.Valid == False)
14          RaiseException(SEGMENTATION_FAULT)
15      else if (CanAccess(PTE.ProtectBits) == False)
16          RaiseException(PROTECTION_FAULT)
17      else
18          TLB_Insert(VPN, PTE.PFN, PTE.ProtectBits)
19          RetryInstruction()
```

Example Integer Array of 10 elements



What is the TLB activity pattern?

Example — Integer Array of 10 elements



What is the TLB activity pattern?

Points to Ponder

- TLB Hit Vs TLB Miss
- Hit rate
- Effect of page-size
- Spatial Locality
- Temporal Locality

Effective Memory Access Time

How to calculate?

Points to Ponder

- TLB Hit Vs TLB Miss
- Hit rate
- Effect of page-size
- Spatial Locality
- Temporal Locality

Effective Memory Access Time

How to calculate?

Points to Ponder

- TLB Hit Vs TLB Miss
- Hit rate
- Effect of page-size
- Spatial Locality
- Temporal Locality

Effective Memory Access Time

How to calculate?

Points to Ponder

- TLB Hit Vs TLB Miss
- Hit rate
- Effect of page-size
- Spatial Locality
- Temporal Locality

Effective Memory Access Time

How to calculate?

Points to Ponder

- TLB Hit Vs TLB Miss
- Hit rate
- Effect of page-size
- Spatial Locality
- Temporal Locality

Effective Memory Access Time

How to calculate?

Points to Ponder

- TLB Hit Vs TLB Miss
- Hit rate
- Effect of page-size
- Spatial Locality
- Temporal Locality

Effective Memory Access Time

How to calculate?

Who Handles The TLB Miss?

- Hardware Vs Software

CISC — Complex-instruction set computers

- Early systems
- TLB miss handled by hardware
- via a page-table base register

RISC — Reduced-instruction set computers

- Modern architectures
- Software-managed TLB
- Hardware simply raises an exception
- Trap handling mechanism takes over

```
1   VPN = (VirtualAddress & VPN_MASK) >> SHIFT
2   (Success, TlbEntry) = TLB_Lookup(VPN)
3   if (Success == True)    // TLB Hit
4       if (CanAccess(TlbEntry.ProtectBits) == True)
5           Offset   = VirtualAddress & OFFSET_MASK
6           PhysAddr = (TlbEntry.PFN << SHIFT) | Offset
7           Register = AccessMemory(PhysAddr)
8       else
9           RaiseException(PROTECTION_FAULT)
10  else                    // TLB Miss
11      RaiseException(TLB_MISS)
```

Points to Ponder

Does the return-from-trap instruction in a TLB miss needs to be a little different than the return-from-trap when servicing a system call?

Can an infinite chain of TLB misses occur with a OS handled TLB miss?

- Possible Solutions

What are the advantages of a software-managed TLB-miss

Points to Ponder

Does the return-from-trap instruction in a TLB miss needs to be a little different than the return-from-trap when servicing a system call?

Can an infinite chain of TLB misses occur with a OS handled TLB miss?

- Possible Solutions

What are the advantages of a software-managed TLB-miss

Points to Ponder

Does the return-from-trap instruction in a TLB miss needs to be a little different than the return-from-trap when servicing a system call?

Can an infinite chain of TLB misses occur with a OS handled TLB miss?

- Possible Solutions

What are the advantages of a software-managed TLB-miss

TLB Contents

- Typically 32, 64, 128 entries
- Fully Associative. What does this mean?

TLB Entry

VPN | PFN | other bits

Other bits

- Valid. Is this same as page valid bit (Refer next slide)?
- Protection
- ASID (Described next)
- Dirty

TLB Contents

- Typically 32, 64, 128 entries
- Fully Associative. What does this mean?

TLB Entry

VPN | PFN | other bits

Other bits

- Valid. Is this same as page valid bit (Refer next slide)?
- Protection
- ASID (Described next)
- Dirty

TLB Contents

- Typically 32, 64, 128 entries
- Fully Associative. What does this mean?

TLB Entry

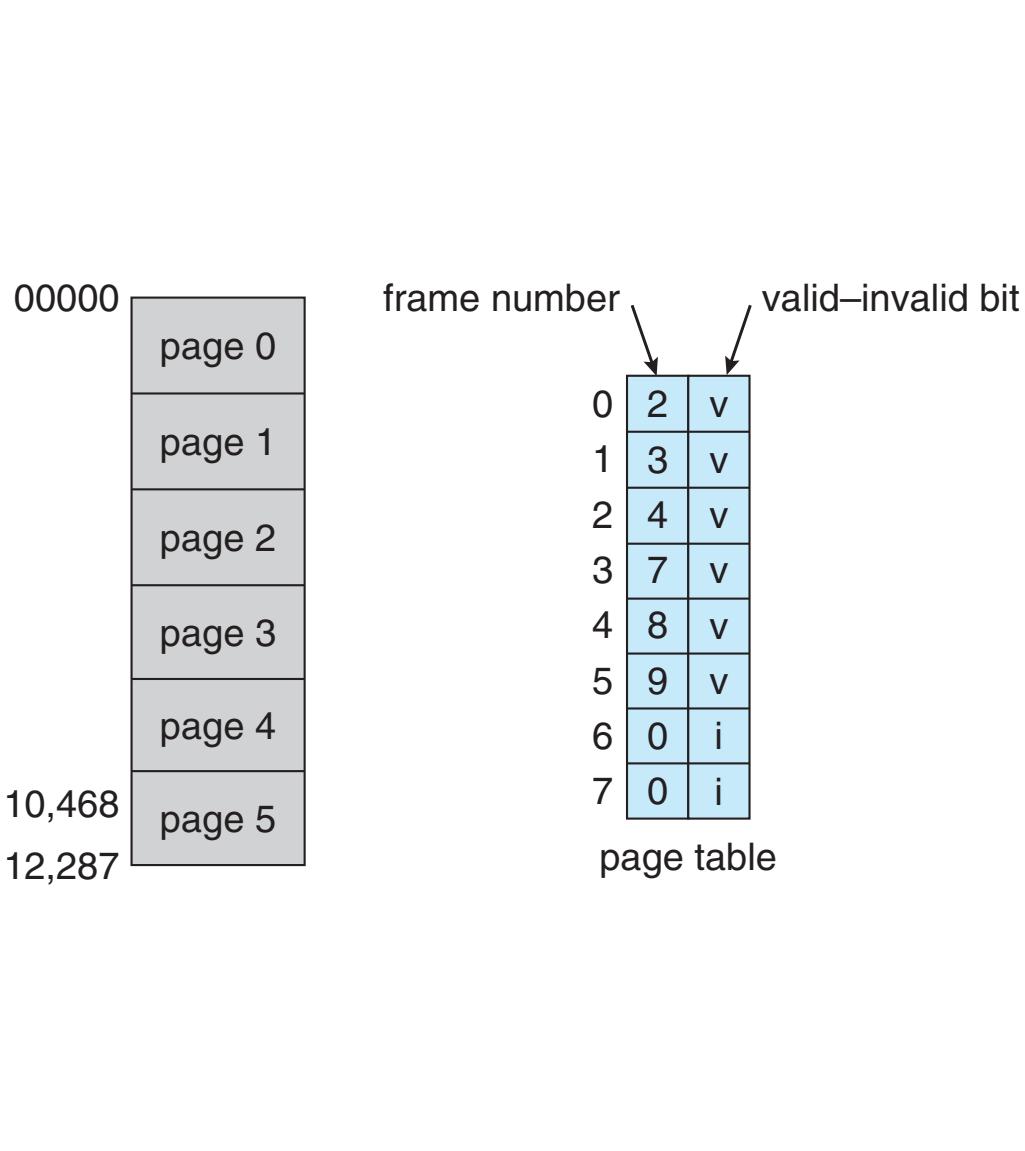
VPN | PFN | other bits

Other bits

- Valid. Is this same as page valid bit (Refer next slide)?
- Protection
- ASID (Described next)
- Dirty

Recall

Valid bit in Paging



How to handle this?

- When context-switching between processes, the translations in the TLB for the last process are **not meaningful to** the about-to-be-run process.

VPN	PFN	valid	prot
10	100	1	rwx
—	—	0	—
10	170	1	rwx
—	—	0	—

Solution #1 Flushing the TLB

How?

- Set all valid bits to 0

When

- In SW use explicit instruction
- In HW while page-table base reg is updated

Issue

- Performance overhead
- Each time a process runs, it **must incur TLB misses** as it touches its data and code pages.

Solution #1 Flushing the TLB

How?

- Set all valid bits to 0

When

- In SW use explicit instruction
- In HW while page-table base reg is updated

Issue

- Performance overhead
- Each time a process runs, it **must incur TLB misses** as it touches its data and code pages.

Solution #1 Flushing the TLB

How?

- Set all valid bits to 0

When

- In SW use explicit instruction
- In HW while page-table base reg is updated

Issue

- Performance overhead
- Each time a process runs, it **must incur TLB misses** as it touches its data and code pages.

Solution #1 Flushing the TLB

How?

- Set all valid bits to 0

When

- In SW use explicit instruction
- In HW while page-table base reg is updated

Issue

- Performance overhead
- Each time a process runs, it **must incur TLB misses** as it touches its data and code pages.

Solution #2 Address Space Identifier (ASID)

VPN	PFN	valid	prot	ASID
10	100	1	rwx	1
—	—	0	—	—
10	170	1	rwx	2
—	—	0	—	—

- Analogous to PID
- Smaller in size
- With ASID the TLB can hold translations from different processes at the same time without any ambiguity.

OS-HW Support

OS must, on a context switch, set some privileged register to the ASID of the current process.

Solution #2 Address Space Identifier (ASID)

VPN	PFN	valid	prot	ASID
10	100	1	rwx	1
—	—	0	—	—
10	170	1	rwx	2
—	—	0	—	—

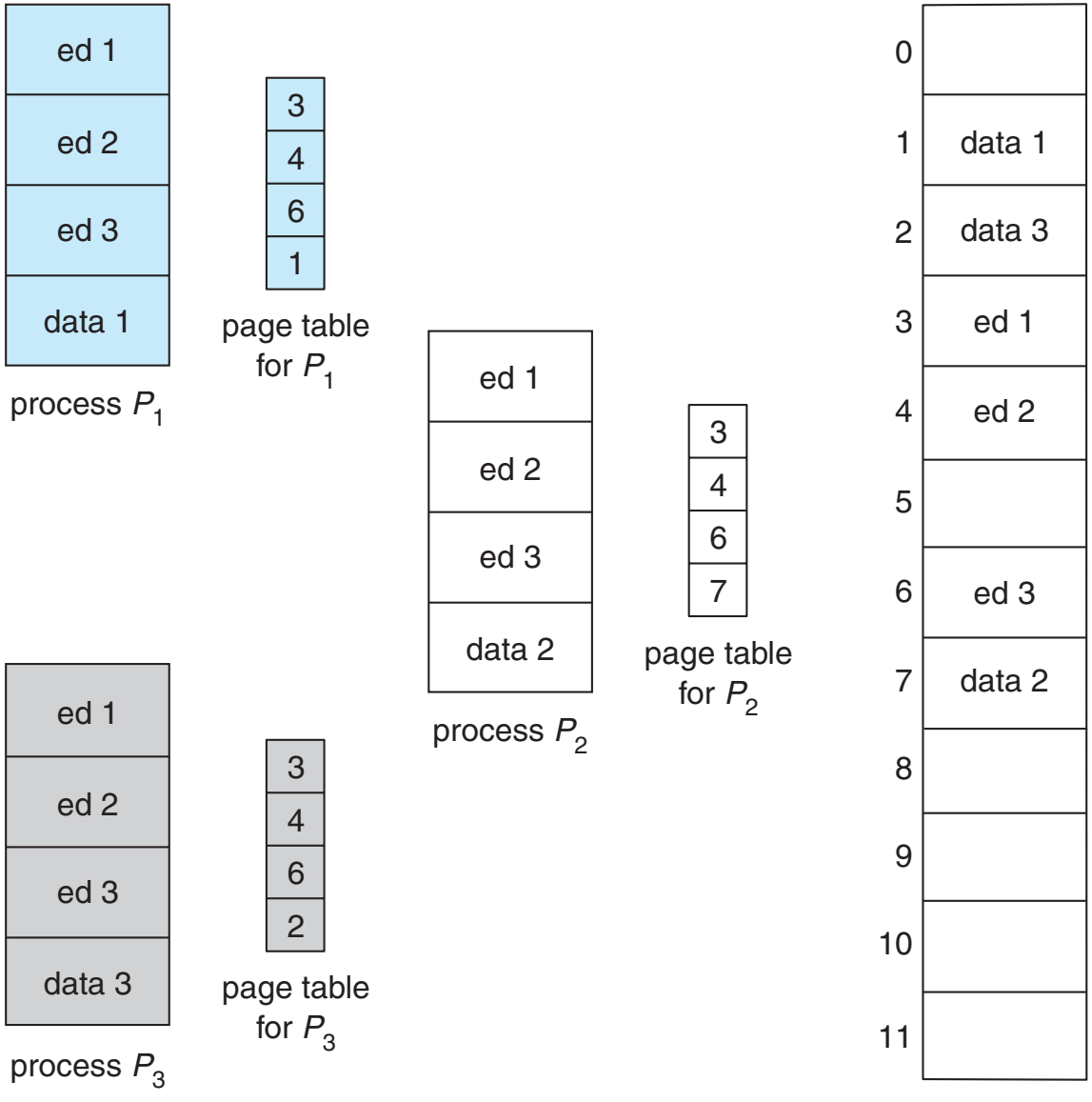
- Analogous to PID
- Smaller in size
- With ASID the TLB can hold translations from different processes at the same time without any ambiguity.

OS-HW Support

OS must, on a context switch, set some privileged register to the ASID of the current process.

Recall

Shared Pages



What would the same reflect in the TLB?

What would the same reflect in the TLB?

VPN	PFN	valid	prot	ASID
10	101	1	r-x	1
—	—	0	—	—
50	101	1	r-x	2
—	—	0	—	—

Which TLB entry should be replaced when we add a new TLB entry?

Goal

Minimize the miss rate (or increase hit rate)

LRU Least Recently Used

Idea

It is likely that an entry that has **not recently been used** is a good candidate for eviction

- Takes advantage of **locality** in the memory-reference stream

Classwork

- Can you devise a program that would perform terribly with LRU?
- What would be the alternative polity?

Random Replacement

- Evicts a TLB mapping at random
- Simple to implement
- Ability to avoid corner-case behaviors like LRU

HW-4/5

Study MIPS TLB Entry

Solves one issue with Paging

- Effective access time is decreased

However

- Issue of TLB Coverage

Does not solve

- Issue of storing large PT in memory

Issues?

Solves one issue with Paging

- Effective access time is decreased

However

- Issue of TLB Coverage

Does not solve

- Issue of storing large PT in memory

Solves one issue with Paging

- Effective access time is decreased

However

- Issue of TLB Coverage

Does not solve

- Issue of storing large PT in memory

Attempt #7: Smaller Page Tables

Next Lecture