CS3281 / CS5281

Virtual Memory (contd.)

CS3281 / CS5281

Spring 2026

*Some lecture slides borrowed and adapted from CMU's
"Computer Systems: A Programmer's Perspective"*

VANDERBILT UNIVERSITY

DRAM Cache Organization

- DRAM cache organization driven by the enormous miss penalty
  - DRAM is about **10x** slower than SRAM (cache)
  - Disk is about **10,000x** slower than DRAM

- Consequences
  - Only a subset of virtual pages are stored in the main memory
  - Highly sophisticated, expensive replacement algorithms
  - Large page (block) size: typically 4 KB
    - CPU caches (SRAM) move data at cache-line granularity (aka "cache blocks"), e.g., 64 B, between SRAM↔DRAM.
    - Virtual memory moves data at page granularity, e.g., 4 KB, between DRAM↔disk. Because the miss penalty to disk is enormous, the transfer unit is much larger than a cache line.

VANDERBILT
UNIVERSITY
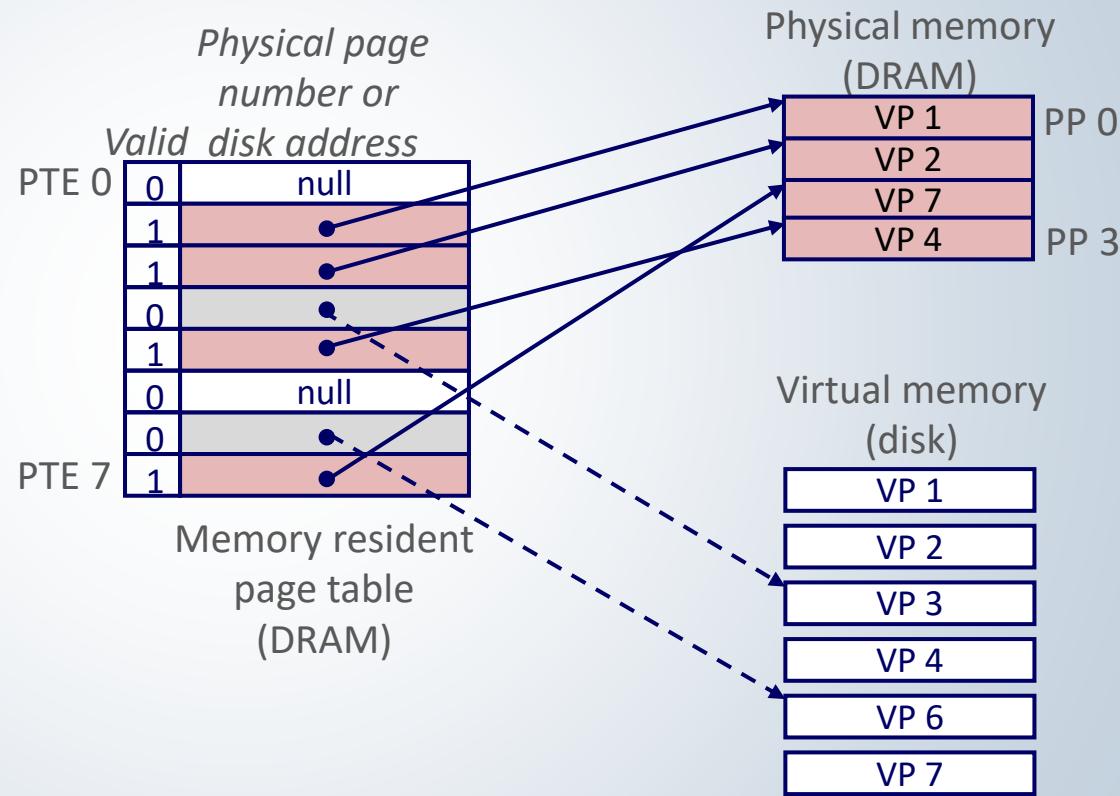
Why pages help: Spatial Locality

- Programs tend to touch nearby addresses soon after one another

- A page fault brings in a whole page (typically 4 KB), not just one byte or one word (32bits) → the next few KB of nearby data/code are already in DRAM.
  - This amortizes the enormous disk miss penalty (disk ≫≫ DRAM), turning one slow I/O into many fast DRAM hits.
- Typical wins:
  - Instruction streams (sequential fetch within functions)
  - Arrays & buffers (for i = 0…N loops)
  - Contiguous allocations

  *Works **only if** locality exists; random sparse access wastes I/O and can cause thrashing.*

VANDERBILT
UNIVERSITY

Enabling Data Structure: Page Table

- A *page table* is an array of page table entries (PTEs) that maps virtual pages to physical pages.
  - Per-process kernel data structure in DRAM

Physical page number or
Valid disk address

PTE 0	0	null
	1	
	1	
	0	
	1	
	0	null
	0	
PTE 7	1	

Memory resident page table (DRAM)

Physical memory (DRAM)

VP 1	PP 0
VP 2	
VP 7	
VP 4	PP 3

Virtual memory (disk)

| VP 1 |
| VP 2 |
| VP 3 |
| VP 4 |
| VP 6 |
| VP 7 |

VM Address Translation
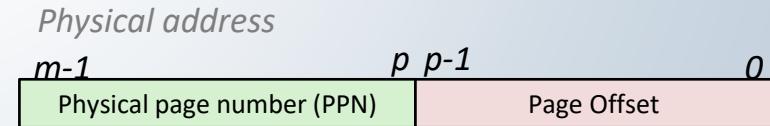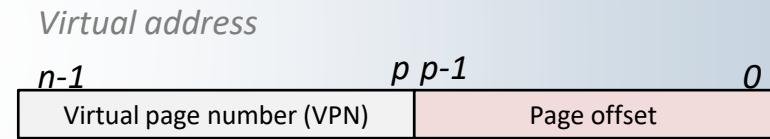
- Virtual Address Space
  - $V = \{0, 1, ..., N{-}1\}$
- Physical Address Space
  - $P = \{0, 1, ..., M{-}1\}$
- Address Translation
  - **MAP: $V \rightarrow P \cup \{\varnothing\}$**
  - For virtual address **a**:
    - **MAP(a) = a'** if data at virtual address **a** is at physical address **a'** in **P**
    - **MAP(a) = $\varnothing$** if data at virtual address **a** is not in physical memory (page fault)

Summary of Address Translation Symbols
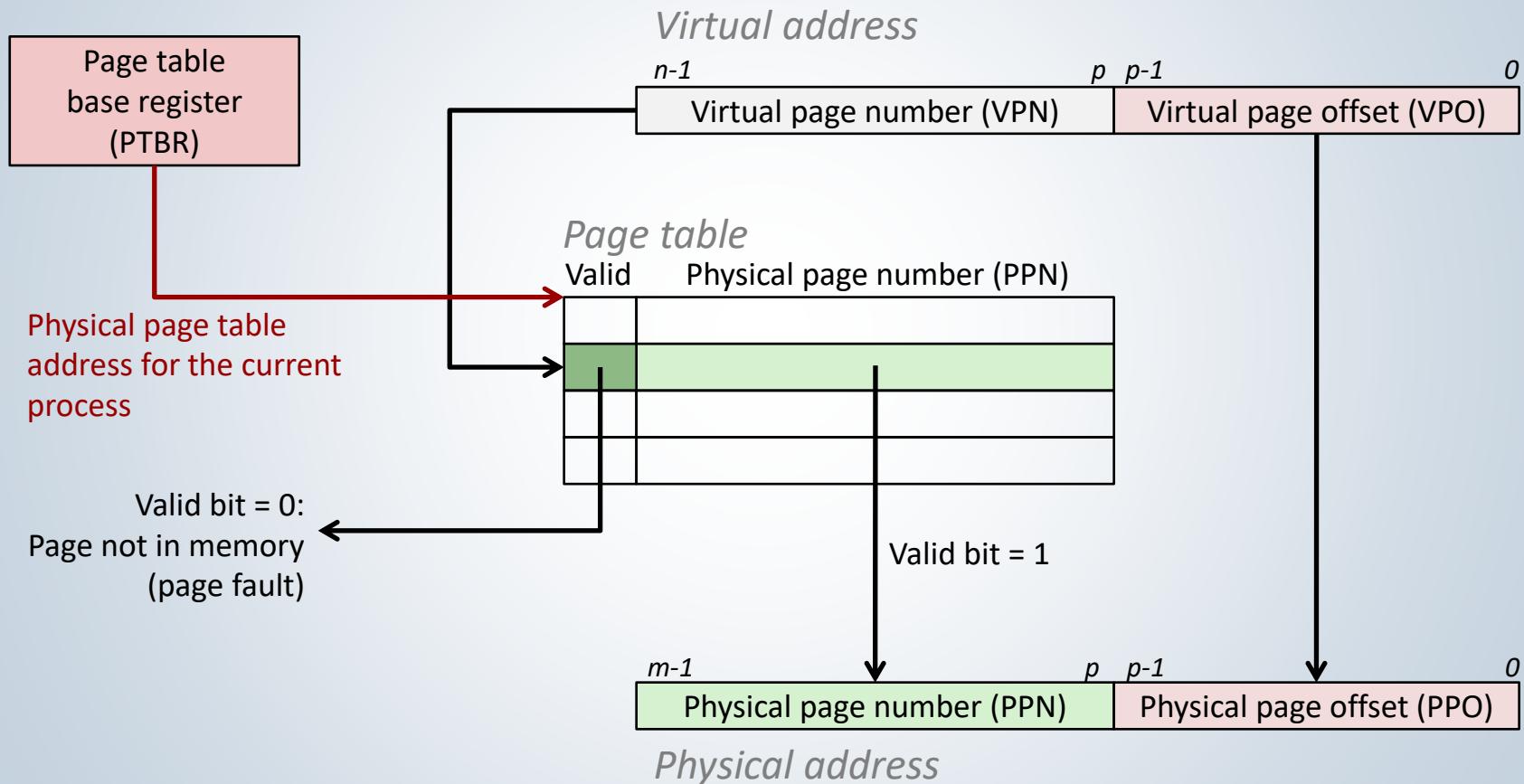
- Basic Parameters
  - **n, m, p:** Number of bits
  - **N = 2^n** : Number of addresses in virtual address space
  - **M = 2^m** : Number of addresses in physical address space
  - **P = 2^p** : Page size (bytes)
- Components of the virtual address (VA)
  - **VPO**: Virtual page offset
  - **VPN**: Virtual page number
- Components of the physical address (PA)
  - **PPO**: Physical page offset (same as VPO)
  - **PPN:** Physical page number

Breaking down virtual addresses

- Basic Parameters
  - $N = 2^n$: Number of addresses in virtual address space
  - $M = 2^m$: Number of addresses in physical address space. $m \leq n$ (usually much less)
  - $P = 2^p$: Page size (bytes)

- Components of the virtual address (VA)
  - Virtual page number (VPN): **n-p** bits
  - Page Offset: **p** bits

- Components of the physical address (PA)
  - Physical page number (PPN): **m-p** bits
  - Page Offset (same offset as VA): **p** bits

Virtual address

$n-1$	p $p-1$	0
Virtual page number (VPN)		Page offset

Physical address

$m-1$	p $p-1$	0
Physical page number (PPN)		Page Offset

VANDERBILT UNIVERSITY

Address Translation with a Page Table

Page table
base register
(PTBR)

Physical page table
address for the current
process

Virtual address

n-1	p	p-1	0
Virtual page number (VPN)		Virtual page offset (VPO)	

Page table

Valid   Physical page number (PPN)

Valid bit = 0:
Page not in memory
(page fault)

Valid bit = 1

Physical address

m-1	p	p-1	0
Physical page number (PPN)		Physical page offset (PPO)	

Virtual memory example

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

Mapping can be anything, which is bigger doesn't really matter!

1. How do we split Virtual Addresses into VPN and Offset?

11	10	9	8	7	6	5	4	3	2	1	0

Virtual memory example

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

Mapping can be anything, which is bigger doesn't really matter!

1. How do we split Virtual Addresses into VPN and Offset?
   - Offset is based on page size: 64-bytes ⇨ 6 bits. All the rest are VPN

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

2. How big are Physical Page Numbers?

VANDERBILT
UNIVERSITY
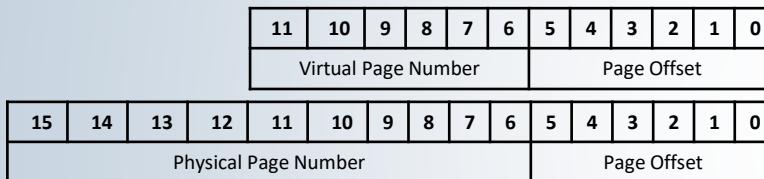
Virtual memory example

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

Mapping can be anything, which is bigger doesn't really matter!

1. How do we split Virtual Addresses into VPN and Offset?
   - Offset is based on page size: 64-bytes ⇨ 6 bits. All the rest are VPN

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

2. How big are Physical Page Numbers? 16-6 = 10 bits

VANDERBILT UNIVERSITY
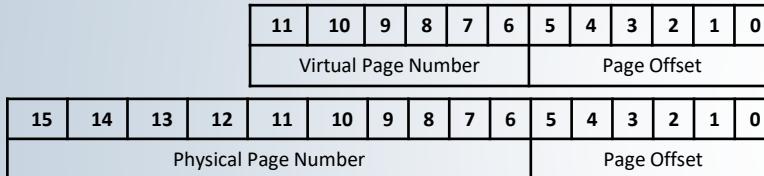
Virtual memory example

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					

- Translate:
- Virtual address: 0x3F0
  - Binary:
  - VPN:
  - Offset:

Virtual memory example
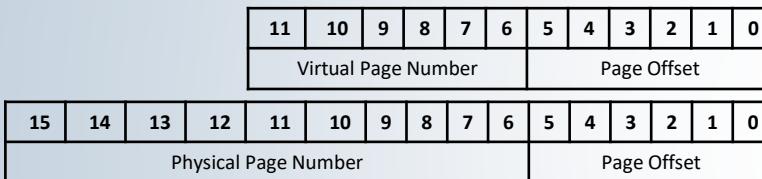
- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					

- Translate:
- Virtual address: 0x3F0
  - Binary: 0b001111110000
  - VPN: 0b001111
  - Offset: 0b110000

Virtual memory example

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					

- Translate:
- Virtual address: 0x3F0
  - Binary: 0b001111110000
  - VPN: 0b001111
  - Offset: 0b110000

VPN	PPN	Valid
0x00	0x123	1
0x01	0x156	1
0x02	0x143	1
0x03	0x16F	1
0x04	0x1FF	0
0x05	0x107	0
0x06	0x100	0
0x07	0x1C0	0
0x08	0x1D8	0
0x09	0x1BF	0
0x0A	0x000	1
0x0B	0x3FF	1
0x0C	0x308	0
0x0D	0x3FD	0
0x0E	0x111	1
0x0F	0x1F0	1

VPN	PPN	Valid
0x10	0x237	1
0x11	0x236	1
0x12	0x2B0	1
0x13	0x280	0
0x14	0x120	0
Continues on…		

- PPN:
- Offset:

VANDERBILT UNIVERSITY

Virtual memory example

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					

- Translate:
- Virtual address: 0x3F0
  - Binary: 0b001111110000
  - VPN: 0b001111
  - Offset: 0b110000

VPN	PPN	Valid
0x00	0x123	1
0x01	0x156	1
0x02	0x143	1
0x03	0x16F	1
0x04	0x1FF	0
0x05	0x107	0
0x06	0x100	0
0x07	0x1C0	0
0x08	0x1D8	0
0x09	0x1BF	0
0x0A	0x000	1
0x0B	0x3FF	1
0x0C	0x308	0
0x0D	0x3FD	0
0x0E	0x111	1
0x0F	0x1F0	1

VPN	PPN	Valid
0x10	0x237	1
0x11	0x236	1
0x12	0x2B0	1
0x13	0x280	0
0x14	0x120	0
Continues on…		

- PPN: 0b01 1111 0000
- Offset: 0b110000

- Physical address:

Virtual memory example

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					

- Translate:

- Virtual address: 0x3F0
  - Binary: 0b001111110000
  - VPN: 0b001111
  - Offset: 0b110000

VPN	PPN	Valid
0x00	0x123	1
0x01	0x156	1
0x02	0x143	1
0x03	0x16F	1
0x04	0x1FF	0
0x05	0x107	0
0x06	0x100	0
0x07	0x1C0	0
0x08	0x1D8	0
0x09	0x1BF	0
0x0A	0x000	1
0x0B	0x3FF	1
0x0C	0x308	0
0x0D	0x3FD	0
0x0E	0x111	1
0x0F	0x1F0	1

VPN	PPN	Valid
0x10	0x237	1
0x11	0x236	1
0x12	0x2B0	1
0x13	0x280	0
0x14	0x120	0
Continues on…		

- PPN: 0b01 1111 0000

- Offset: 0b110000

- Physical address:
  - 0b0111110000110000
  - 0x7C30

ISIS

VANDERBILT UNIVERSITY

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					
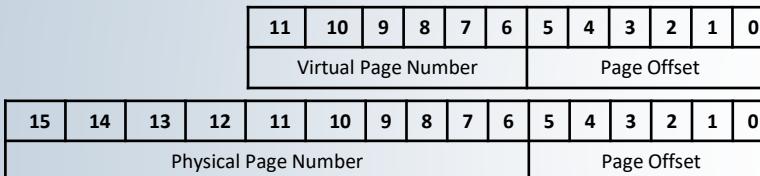
- Translate:
- Virtual address: 0x500
  - Binary:
  - VPN:
  - Offset:

VPN	PPN	Valid
0x00	0x123	1
0x01	0x156	1
0x02	0x143	1
0x03	0x16F	1
0x04	0x1FF	0
0x05	0x107	0
0x06	0x100	0
0x07	0x1C0	0
0x08	0x1D8	0
0x09	0x1BF	0
0x0A	0x000	1
0x0B	0x3FF	1
0x0C	0x308	0
0x0D	0x3FD	0
0x0E	0x111	1
0x0F	0x1F0	1

VPN	PPN	Valid
0x10	0x237	1
0x11	0x236	1
0x12	0x2B0	1
0x13	0x280	0
0x14	0x120	0
Continues on…		

- PPN:

- Offset:

- Physical address:

ISIS

VANDERBILT UNIVERSITY

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					
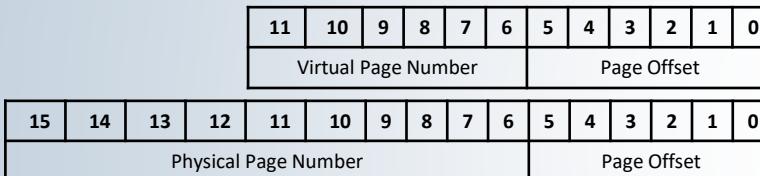
- Translate:
- Virtual address: 0x500
  - Binary: 0b010100000000
  - VPN: 0b010100
  - Offset: 0b000000

VPN	PPN	Valid
0x00	0x123	1
0x01	0x156	1
0x02	0x143	1
0x03	0x16F	1
0x04	0x1FF	0
0x05	0x107	0
0x06	0x100	0
0x07	0x1C0	0
0x08	0x1D8	0
0x09	0x1BF	0
0x0A	0x000	1
0x0B	0x3FF	1
0x0C	0x308	0
0x0D	0x3FD	0
0x0E	0x111	1
0x0F	0x1F0	1

VPN	PPN	Valid
0x10	0x237	1
0x11	0x236	1
0x12	0x2B0	1
0x13	0x280	0
0x14	0x120	0
Continues on…		

- PPN: INVALID
- Offset:
- Physical address:
  - **Page Fault**

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					
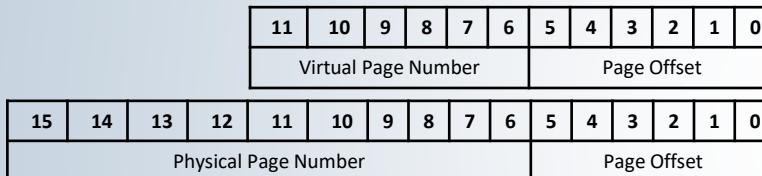
- Translate:

- Virtual address: 0x0D6
  - Binary:
  - VPN:
  - Offset:

VPN	PPN	Valid
0x00	0x123	1
0x01	0x156	1
0x02	0x143	1
0x03	0x16F	1
0x04	0x1FF	0
0x05	0x107	0
0x06	0x100	0
0x07	0x1C0	0
0x08	0x1D8	0
0x09	0x1BF	0
0x0A	0x000	1
0x0B	0x3FF	1
0x0C	0x308	0
0x0D	0x3FD	0
0x0E	0x111	1
0x0F	0x1F0	1

VPN	PPN	Valid
0x10	0x237	1
0x11	0x236	1
0x12	0x2B0	1
0x13	0x280	0
0x14	0x120	0
Continues on…		

- PPN:

- Offset:

- Physical address:

VANDERBILT UNIVERSITY

- Parameters
  - Virtual addresses are 12-bits
  - Physical addresses are 16-bits
  - Page size is 64 bytes

11	10	9	8	7	6	5	4	3	2	1	0
Virtual Page Number						Page Offset					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Physical Page Number										Page Offset					
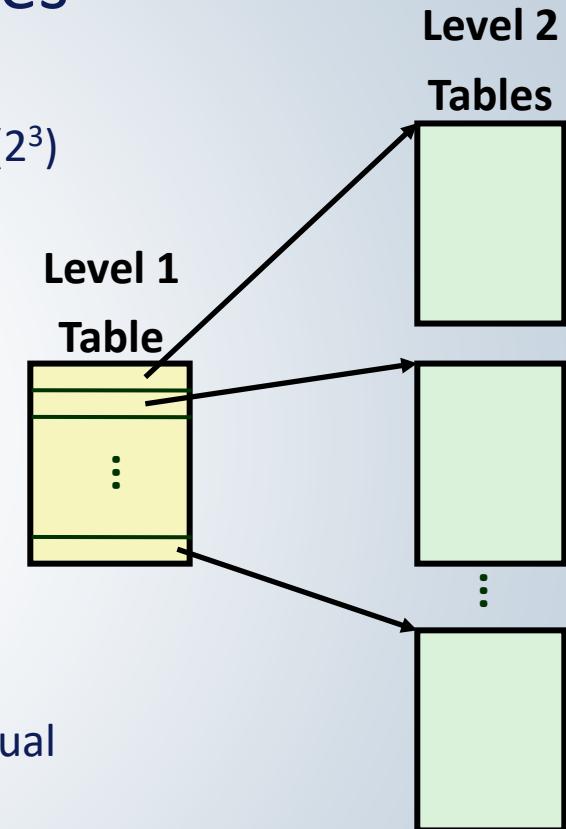
- Translate:
- Virtual address: 0x0D6
  - Binary: 0b000011010110
  - VPN: 0b000011
  - Offset: 0b010110

VPN	PPN	Valid
0x00	0x123	1
0x01	0x156	1
0x02	0x143	1
0x03	0x16F	1
0x04	0x1FF	0
0x05	0x107	0
0x06	0x100	0
0x07	0x1C0	0
0x08	0x1D8	0
0x09	0x1BF	0
0x0A	0x000	1
0x0B	0x3FF	1
0x0C	0x308	0
0x0D	0x3FD	0
0x0E	0x111	1
0x0F	0x1F0	1

VPN	PPN	Valid
0x10	0x237	1
0x11	0x236	1
0x12	0x2B0	1
0x13	0x280	0
0x14	0x120	0
Continues on…		
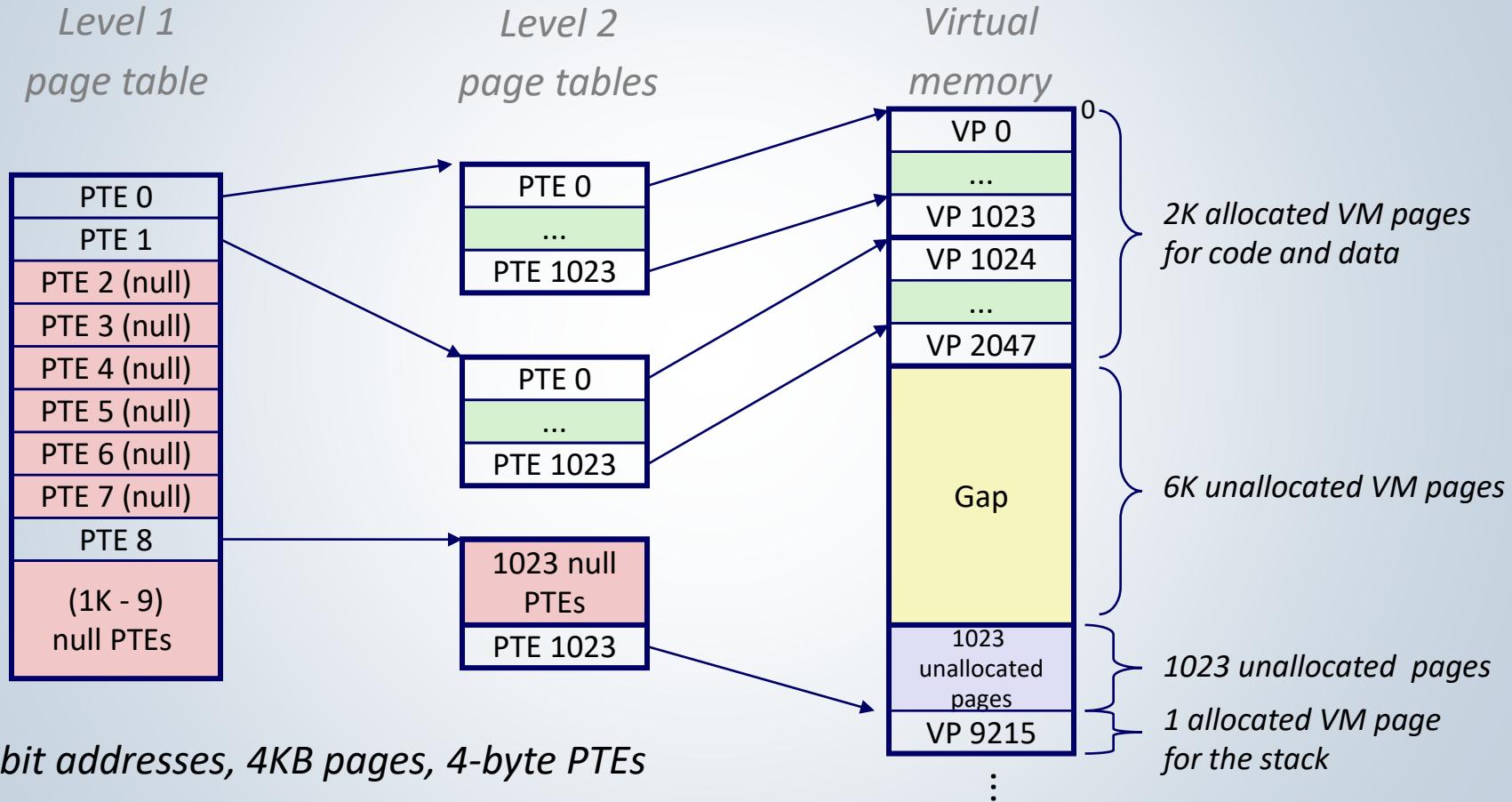
- PPN: 0b010 110 1111
- Offset: 0b010110

- Physical address:
  - 0b0101101111010110
  - 0x5BD6

VANDERBILT UNIVERSITY

Multi-Level Page Tables
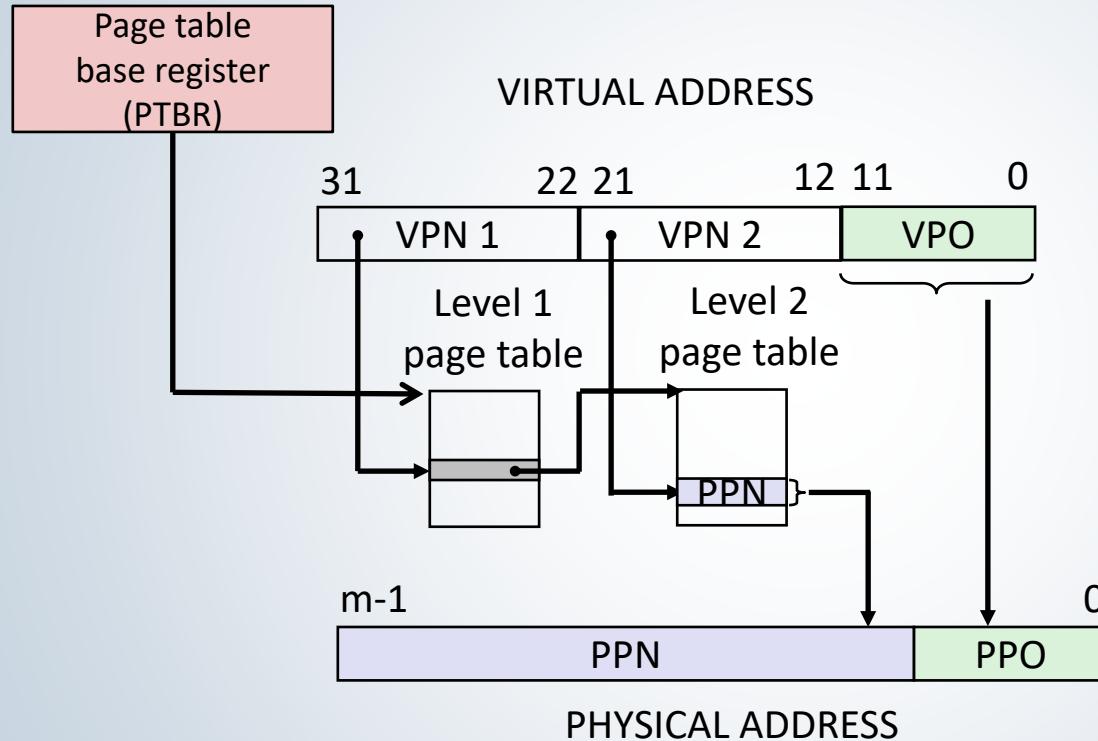
- Suppose:
  - 4KB (2^{12}) page size, 48-bit address space, 8-byte PTE (2^3)

- Problem:
  - Would need a 512 GB page table!
    - $2^{48} * 2^{-12} * 2^3 = 2^{39}$ bytes

- Common solution: Multi-level page table
- Example: 2-level page table
  - Level 1 table: each PTE points to a page table (always memory resident)
  - Level 2 table: each PTE points to a data page (paged in and out like any other data)
  - Huge advantage: Allocate an L2 table when some virtual pages in that region are actually used

Level 2 Tables

Level 1 Table

VANDERBILT UNIVERSITY

Two-Level Page-Table Hierarchy

Level 1 page table

Level 2 page tables

Virtual memory

Level 1 page table
PTE 0
PTE 1
PTE 2 (null)
PTE 3 (null)
PTE 4 (null)
PTE 5 (null)
PTE 6 (null)
PTE 7 (null)
PTE 8
(1K - 9) null PTEs

Level 2 table A
PTE 0
...
PTE 1023

Level 2 table B
PTE 0
...
PTE 1023

Level 2 table C
1023 null PTEs
PTE 1023

Virtual memory:
- VP 0
- ...
- VP 1023
- VP 1024
- ...
- VP 2047
- Gap
- 1023 unallocated pages
- VP 9215

2K allocated VM pages for code and data

6K unallocated VM pages

1023 unallocated pages

1 allocated VM page for the stack

32-bit addresses, 4KB pages, 4-byte PTEs

Translating with a 2-Level Page Table

Page table base register (PTBR)

VIRTUAL ADDRESS

31   22 21   12 11   0

| VPN 1 | VPN 2 | VPO |

Level 1 page table

Level 2 page table

PPN

m-1   0

| PPN | PPO |

PHYSICAL ADDRESS

Multi-level page table (example)

- Consider a 3-level page table. A page table has 64 entries. If the page size is 4K bytes, how much is the size of the virtual address space?

Multi-level page table (example)

- Consider a 3-level page table. A page table has 64 entries. If the page size is 4K bytes, how much is the size of the virtual address space?

- 64 entries per table: 6 bits of index per level

- Page size = 4KB, 2^{12} -> 12 offset bits

- Bits view
  - 3 levels * 6 bits = 18 + offset = 30 bits
  - The virtual address space size is 2^{30} bytes = **1 GiB**

VANDERBILT
UNIVERSITY

Multi-level page table (example)

- Another view:

- L3 table maps 64 pages: 64 x 4KB = 256KB

- L2 table has 64 entries, each mapping to an L3: 64 x 256KB = 16384KB = 16MB

- L1 has 64 entries, each mapping to an L2: 64 x 16MB = 1024MB = 1GB