Virtual Memory

EEL 3713C: Digital Computer Architecture

Quincy Flint

[Ionospheric Radio Lab in NEB]

Outline

1. Memory Problems
   - Not enough memory
   - Holes in address space
   - Programs overwriting
2. What is Virtual Memory?
   - Layer of indirection
   - How does indirection solve above
   - Page tables and translation
3. How do we implement VM?
   - Create and store page tables
   - Fast address translation
4. Virtual Memory and Caches
   - Prevent cache performance degradation when using VM

Page Tables

Page Tables

- The Page Table is the mapping from Virtual to Physical address

Page Tables

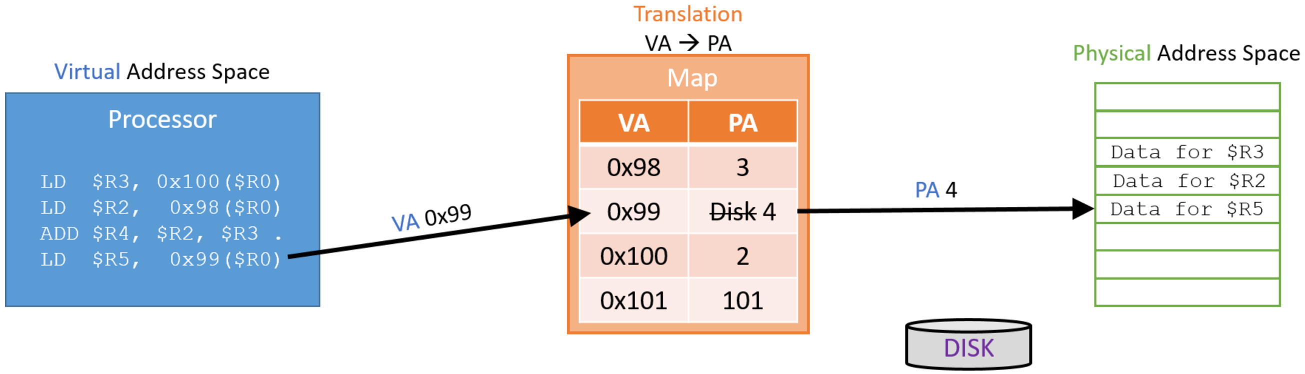
- The Page Table is the mapping from Virtual to Physical address

- So far we have had 1 Page Table Entry (PTE) for every Virtual Address

Page Tables

- The Page Table is the mapping from Virtual to Physical address

- So far we have had 1 Page Table Entry (PTE) for every Virtual Address



```
LD  $R3, 0x100($R0)
LD  $R2,  0x98($R0)
ADD $R4, $R2, $R3 .
LD  $R5,  0x99($R0)
```

VA	PA
0x98	3
0x99	~~Disk~~ 4
0x100	2
0x101	101

```
Data for $R3
Data for $R2
Data for $R5
```

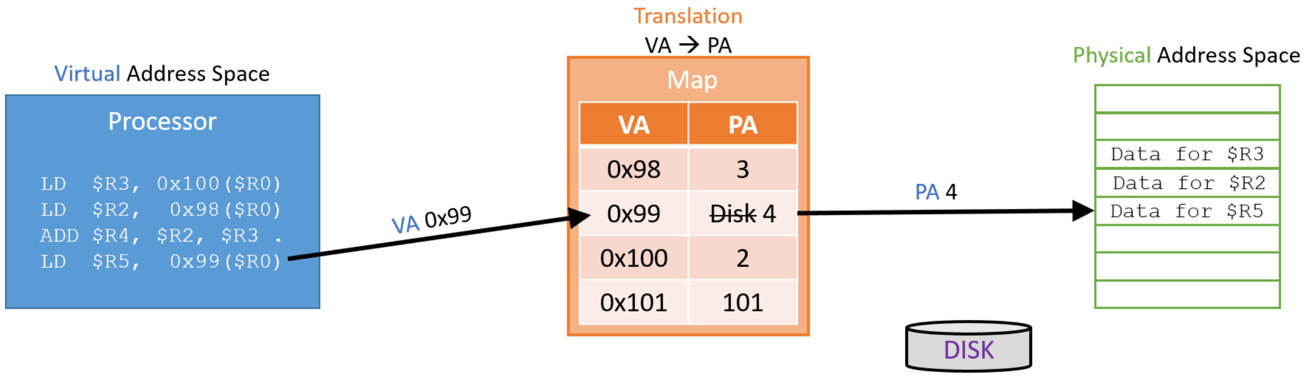
Page Tables

- The Page Table is the mapping from Virtual to Physical address

- So far we have had 1 Page Table Entry (PTE) for every Virtual Address

Q: Given 1 Page Table Entry per Virtual Address, how many entries do we need in our Page Table?

- 1 for each Byte… 2^{32} [4 billion]
- 1 for each Word… 2^{30} [1 billion]
- 1 for each Register… 32
- Undetermined



```
LD  $R3, 0x100($R0)
LD  $R2,  0x98($R0)
ADD $R4, $R2, $R3 .
LD  $R5,  0x99($R0)
```

VA	PA
0x98	3
0x99	~~Disk~~ 4
0x100	2
0x101	101

```
Data for $R3
Data for $R2
Data for $R5
```

Page Tables

- The Page Table is the mapping from Virtual to Physical address

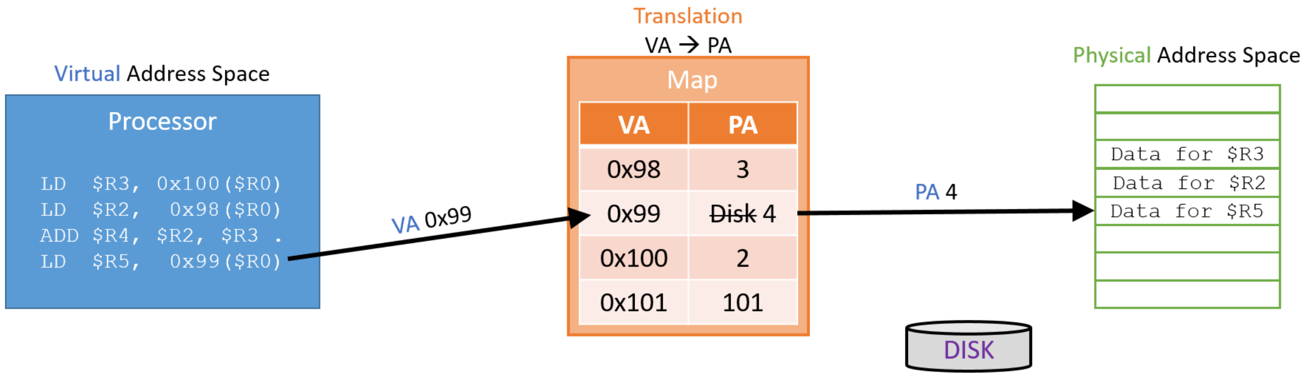
- So far we have had 1 Page Table Entry (PTE) for every Virtual Address

Q: Given 1 Page Table Entry per Virtual Address, how many entries do we need in our Page Table?

- 1 for each Byte… 2^{32} [4 billion]
- 1 for each Word… 2^{30} [1 billion]
- 1 for each Register… 32
- Undetermined

A: 1 for each Word… 2^{30} [1 billion entries]

Memory is word-aligned and we need to access every word. That’s a total of 1GB just for this table!



Virtual Address Space

Processor

```
LD  $R3, 0x100($R0)
LD  $R2,  0x98($R0)
ADD $R4, $R2, $R3 .
LD  $R5,  0x99($R0)
```

Translation
VA → PA

Map

VA	PA
0x98	3
0x99	~~Disk~~ 4
0x100	2
0x101	101

Physical Address Space

```
Data for $R3
Data for $R2
Data for $R5
```

Illustration from the textbook

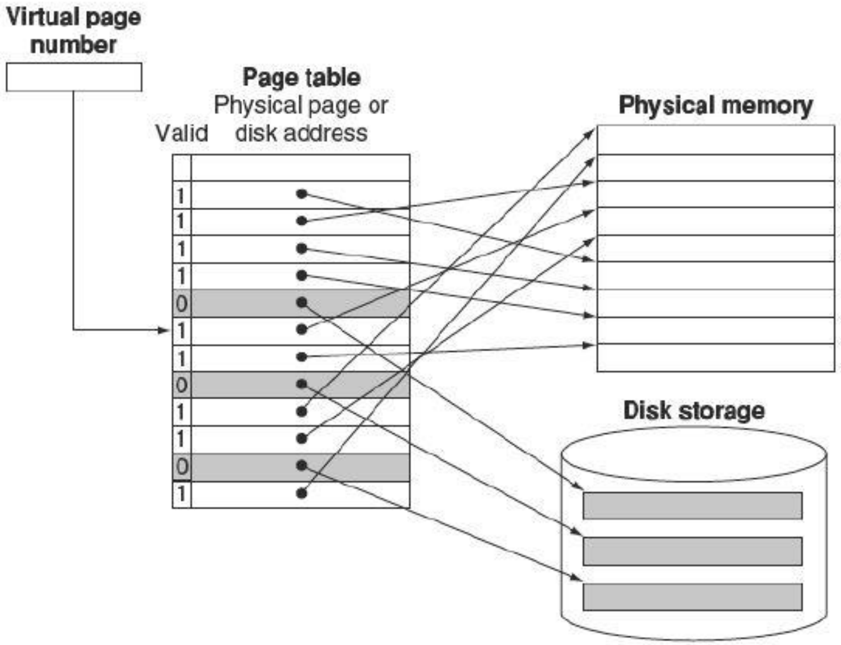


FIGURE 5.28 The page table maps each page in virtual memory to either a page in main memory or a page stored on disk, which is the next level in the hierarchy. The virtual page number is used to index the page table. If the valid bit is on, the page table supplies the physical page number (i.e., the starting address of the page in memory) corresponding to the virtual page. If the valid bit is off, the page currently resides only on disk, at a specified disk address. In many systems, the table of physical page addresses and disk page addresses, while logically one table, is stored in two separate data structures. Dual tables are justified in part because we must keep the disk addresses of all the pages, even if they are currently in main memory. Remember that the pages in main memory and the pages on disk are the same size.

Page Table Size

- We need to translate every address in address space

Page Table Size

- We need to translate every address in address space
  - Our programs have a 32-bit Virtual Address space

Page Table Size

- We need to translate every address in address space
    - Our programs have a 32-bit Virtual Address space
        - 2^{30} entries in our Page Table (as seen so far)
        - Must have a Page Table Entry or we cannot perform translation

Page Table Size

- We need to translate every address in address space
  - Our programs have a 32-bit Virtual Address space
    - 2^{30} entries in our Page Table (as seen so far)
    - Must have a Page Table Entry or we cannot perform translation

- How can we make this more manageable?

Page Table Size

- We need to translate every address in address space
  - Our programs have a 32-bit Virtual Address space
    - 2^{30} entries in our Page Table (as seen so far)
    - Must have a Page Table Entry or we cannot perform translation

- How can we make this more manageable?
  - Segment table into chunks of addresses (**pages**) instead of singular addresses

Page Table Size

- We need to translate every address in address space
  - Our programs have a 32-bit Virtual Address space
    - 2^{30} entries in our Page Table (as seen so far)
    - Must have a Page Table Entry or we cannot perform translation

- How can we make this more manageable?
  - Segment table into chunks of addresses (**pages**) instead of singular addresses

Fine-Grain Approach:
1 address per entry

Translation
VA → PA

Map

VA	PA
0x98	3
0x99	~~Disk~~ 4
0x100	2
0x101	101

Page Table Size

- We need to translate every address in address space
  - Our programs have a 32-bit Virtual Address space
    - 2^{30} entries in our Page Table (as seen so far)
    - Must have a Page Table Entry or we cannot perform translation
- How can we make this more manageable?
  - Segment table into chunks of addresses (**pages**) instead of singular addresses

Fine-Grain Approach:
1 address per entry

Translation
VA → PA

Map

VA	PA
0x98	3
0x99	~~Disk~~ 4
0x100	2
0x101	101

Coarse-Grain Approach:
Many addresses per entry. Can map same number of addresses in smaller area.

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Page Table Size

- We need to translate every address in address space
  - Our programs have a 32-bit Virtual Address space
    - 2^{30} entries in our Page Table (as seen so far)
    - Must have a Page Table Entry or we cannot perform translation

- How can we make this more manageable?
  - Segment table into chunks of addresses (**pages**) instead of singular addresses

Fine-Grain Approach:
1 address per entry

Translation
VA → PA

Map

VA	PA
0x98	3
0x99	~~Disk~~ 4
0x100	2
0x101	101

Coarse-Grain Approach:
Many addresses per entry. Can map same number of addresses in smaller area.

4 KB Page!

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Page Entries, not Word Entries

- The Page Table now manages pages of data, not word entries

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Page Entries, not Word Entries

- The Page Table now manages pages of data, not word entries
  + We need fewer entries to cover the entire address space

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Page Entries, not Word Entries

- The Page Table now manages pages of data, not word entries

  + We need fewer entries to cover the entire address space

  -- BUT less flexible in how we allocate RAM

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Page Entries, not Word Entries

- The Page Table now manages pages of data, not word entries

  + We need fewer entries to cover the entire address space

  -- BUT less flexible in how we allocate RAM

- Today it is common to use 4 kB pages [1,024 words per page]

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Page Entries, not Word Entries

- The Page Table now manages pages of data, not word entries

  + We need fewer entries to cover the entire address space

  -- BUT less flexible in how we allocate RAM

- Today it is common to use 4 kB pages [1,024 words per page]

Q: How many entries do we need in our **Page Table** with **4kB pages** on a **32-bit machine**?

- 2^{32} [4 billion]
- 2^{30} [1 billion]
- 2^{20} [1 million]
- 2^{18} [1/4 million]

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Page Entries, not Word Entries

- The Page Table now manages pages of data, not word entries

  + We need fewer entries to cover the entire address space

  -- BUT less flexible in how we allocate RAM

- Today it is common to use 4 kB pages [1,024 words per page]

Q: How many entries do we need in our **Page Table** with **4kB pages** on a **32-bit machine**?

- 2^{32} [4 billion]
- 2^{30} [1 billion]
- 2^{20} [1 million]
- 2^{18} [1/4 million]

A: 2^{20} [1 million]

We need to address 2^{32} Bytes total but we partition pages into 2^{12} (4 kB) chunks. By simple division we need 2^{20}, or 1 million entries.

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Virtual Address Space

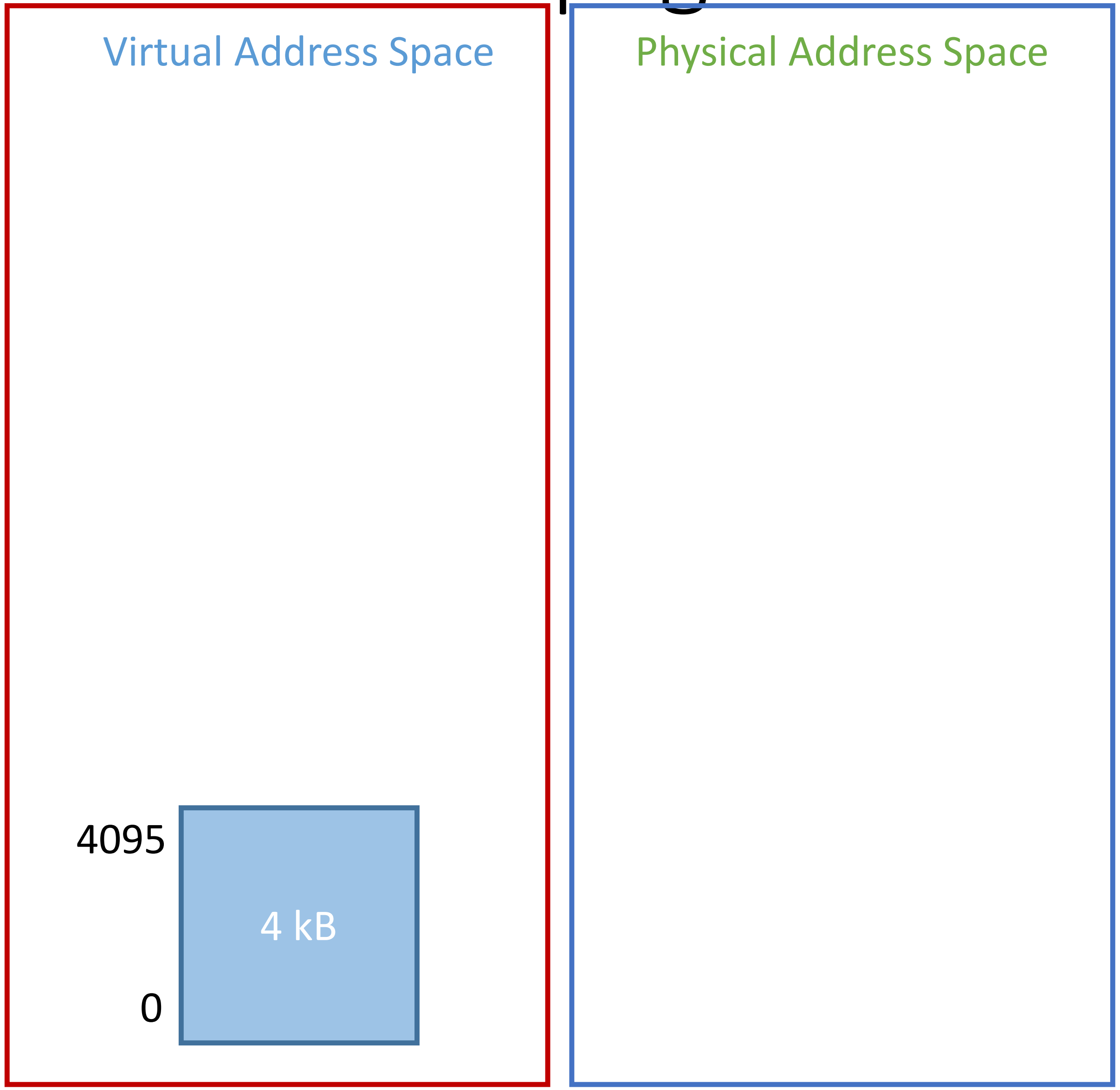
Physical Address Space

How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

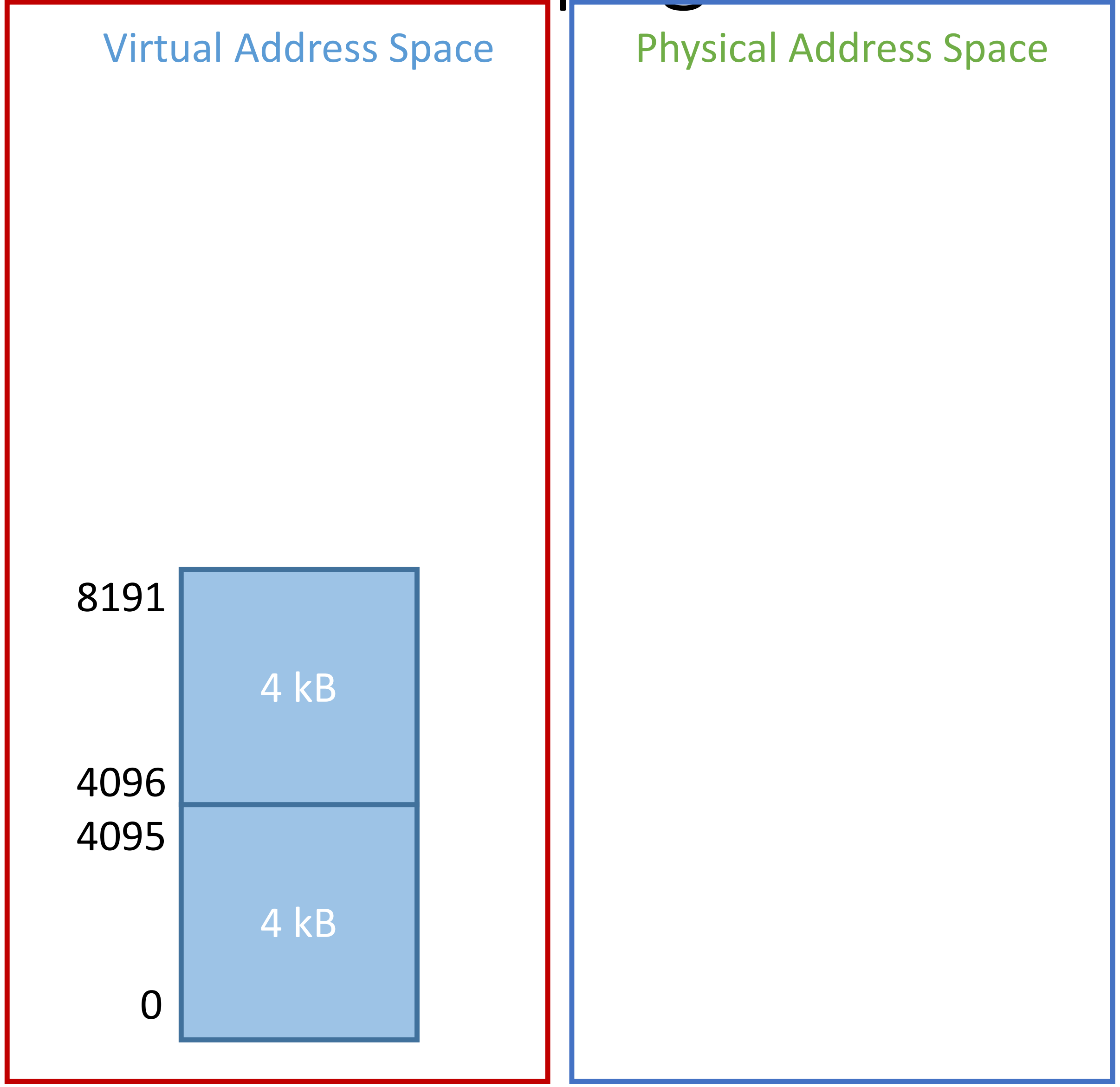


How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

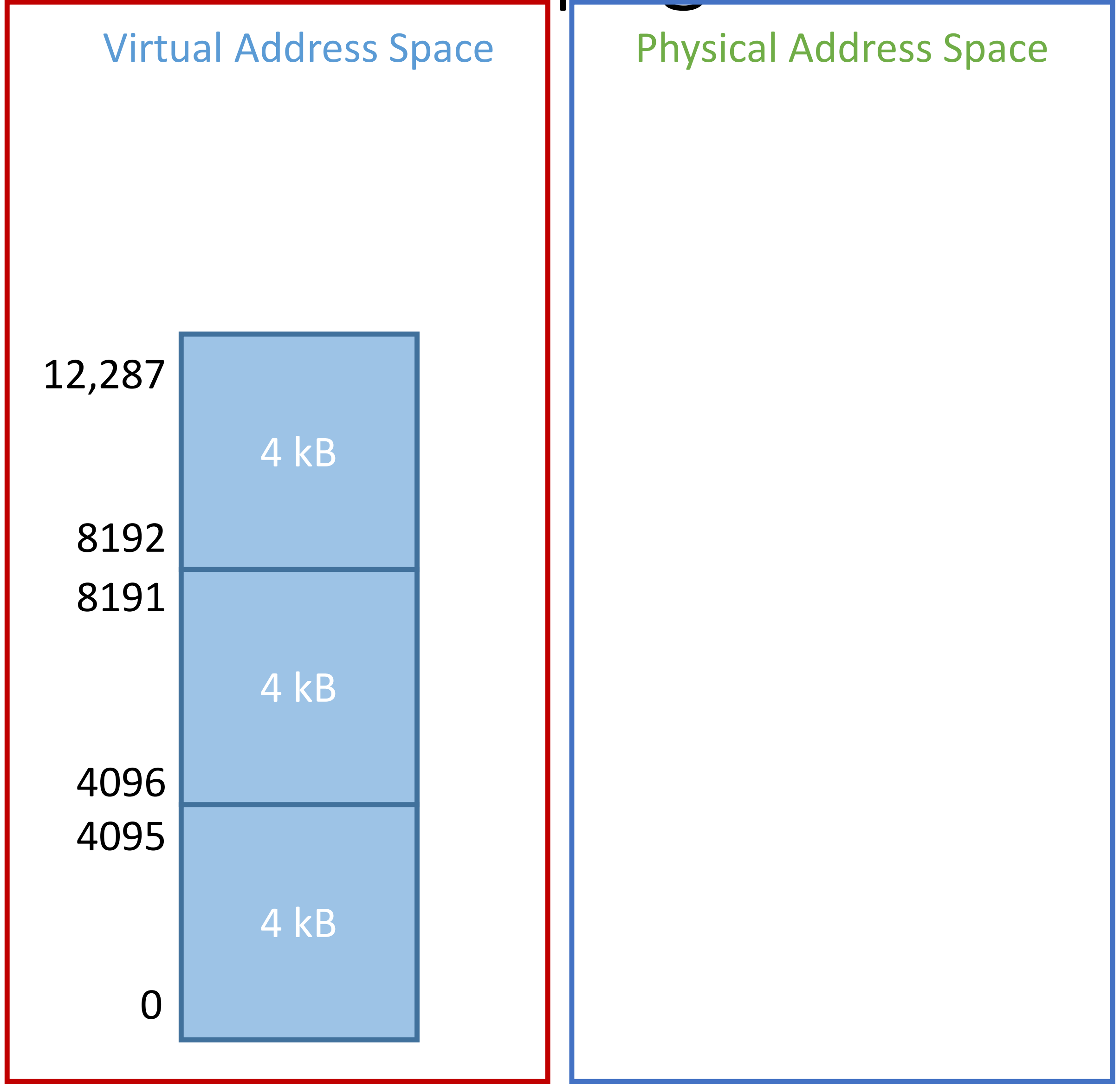


How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

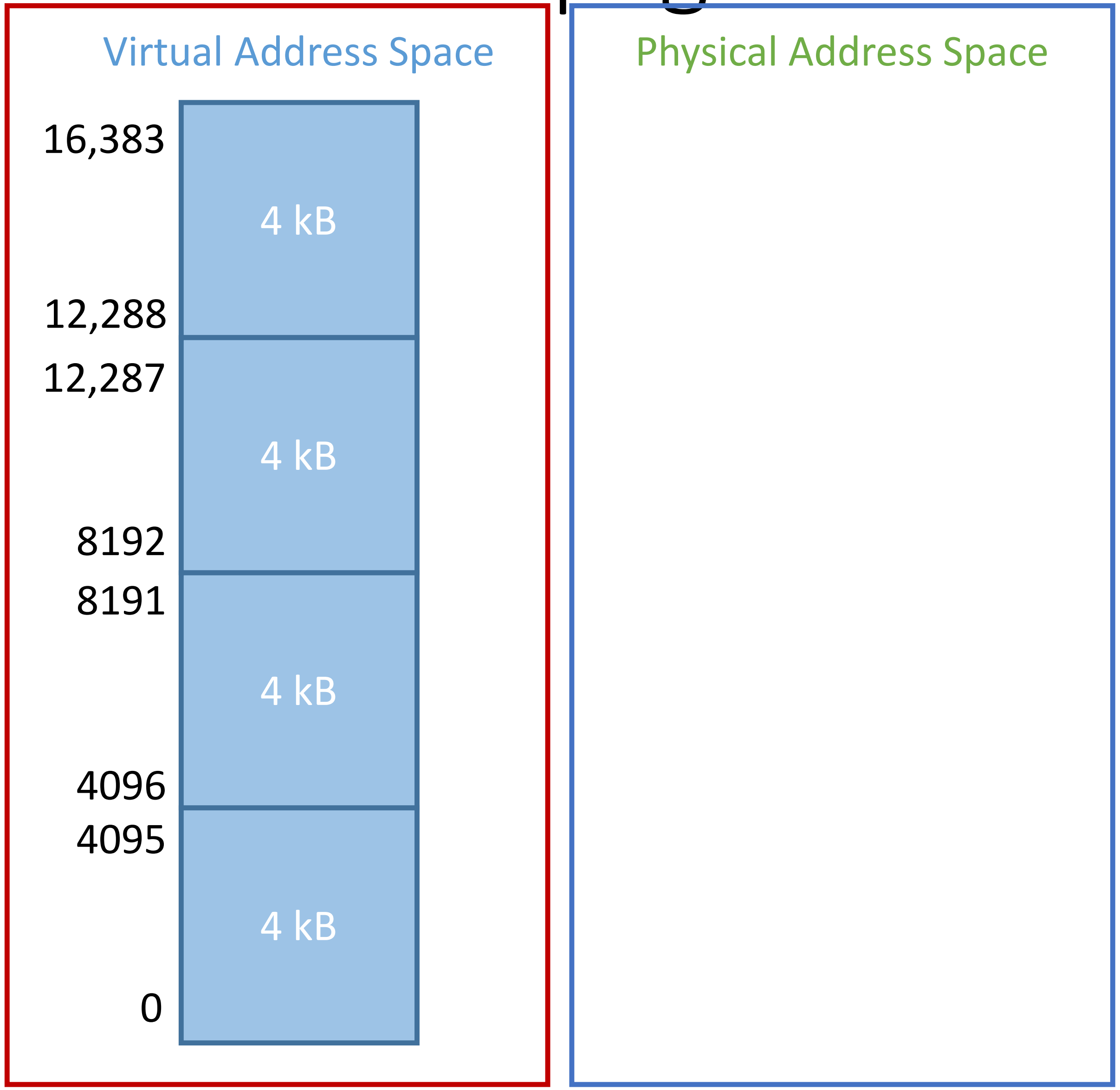


How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

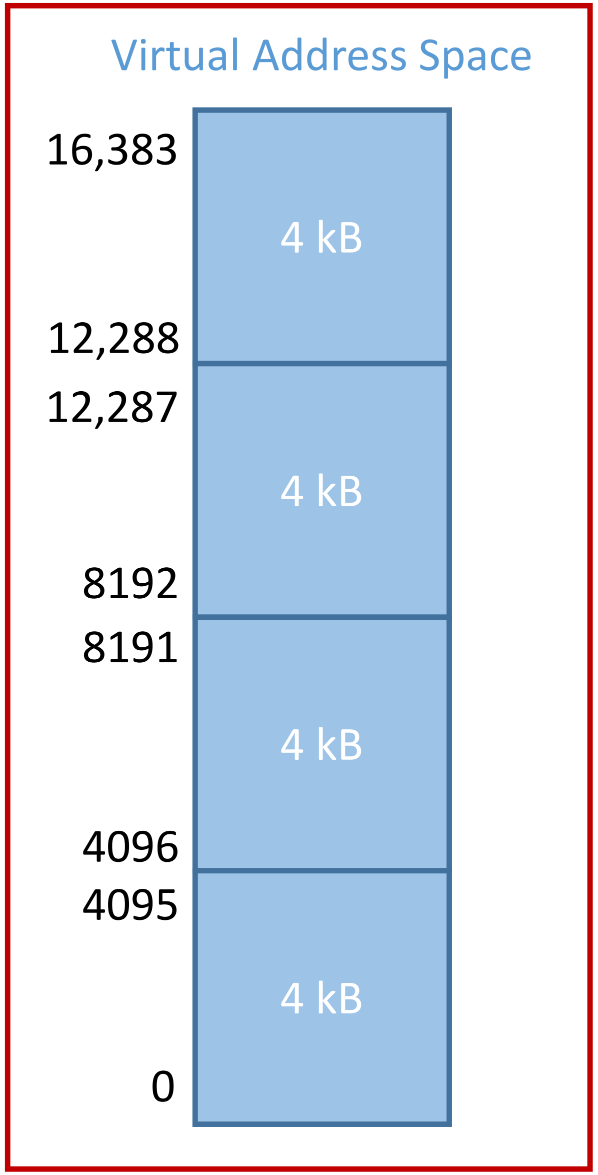


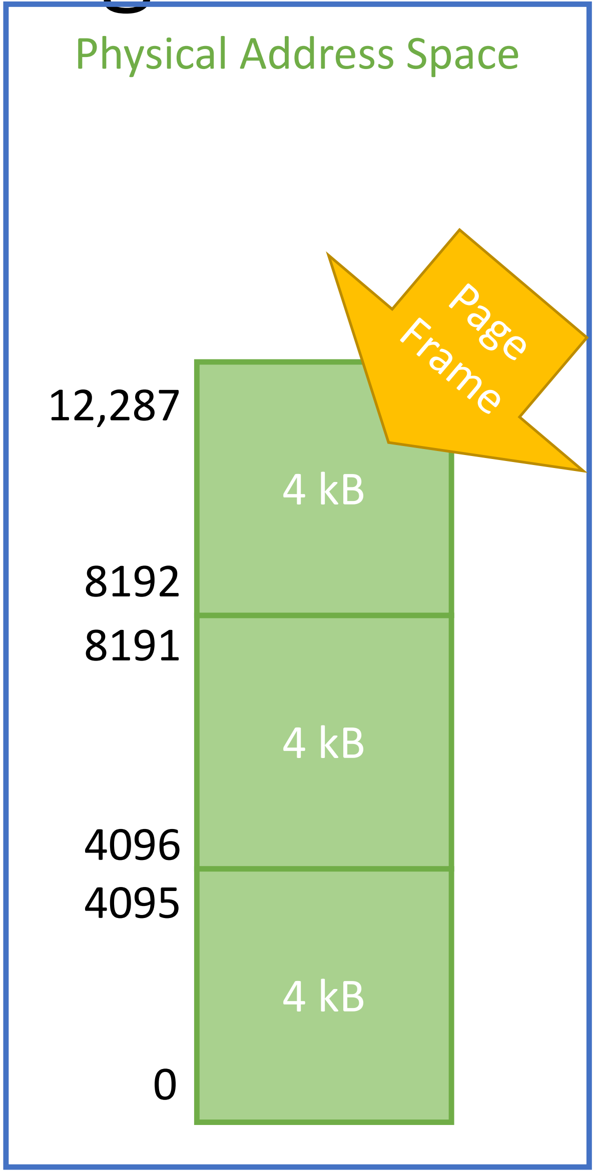
How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191



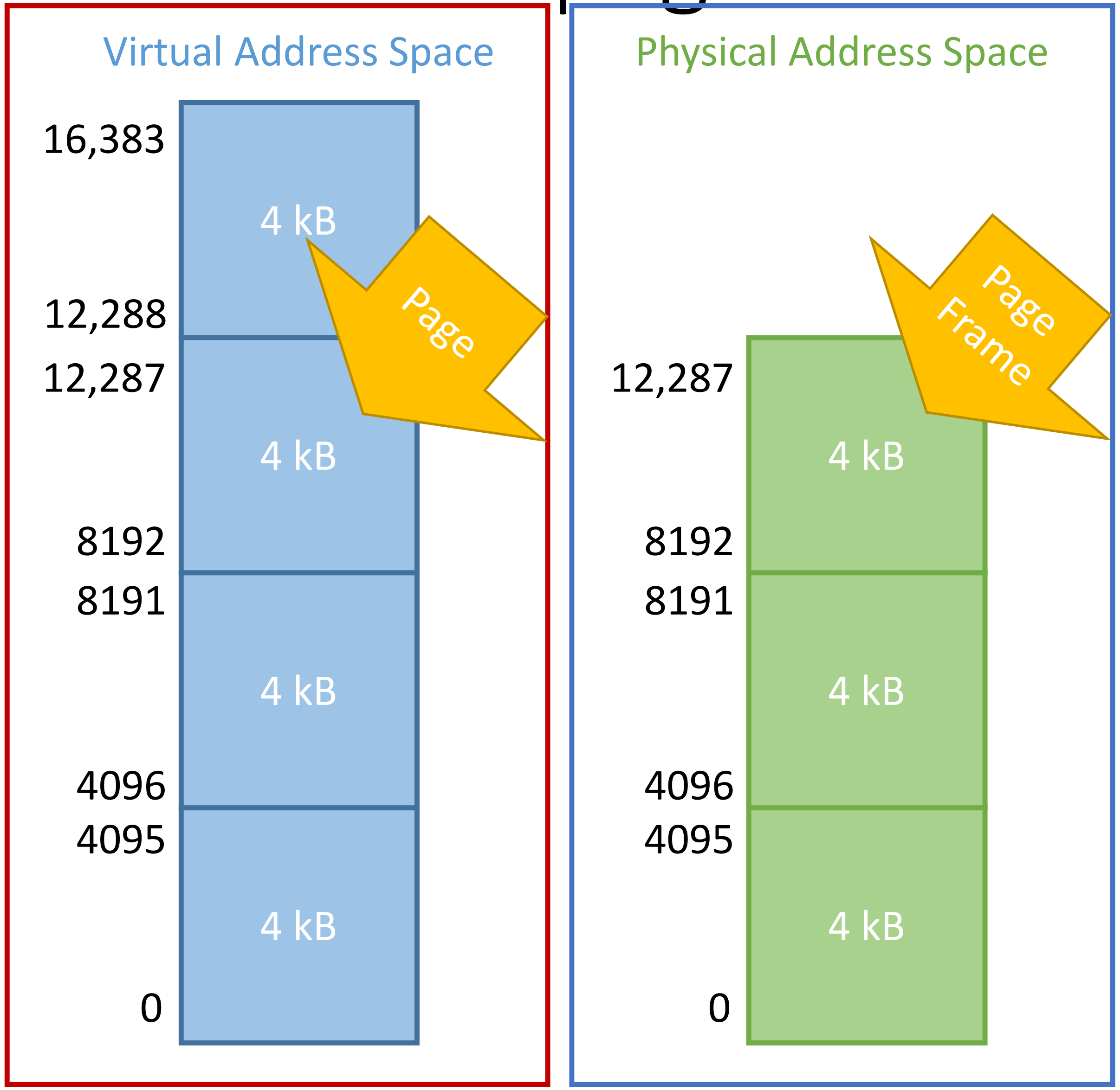


How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191



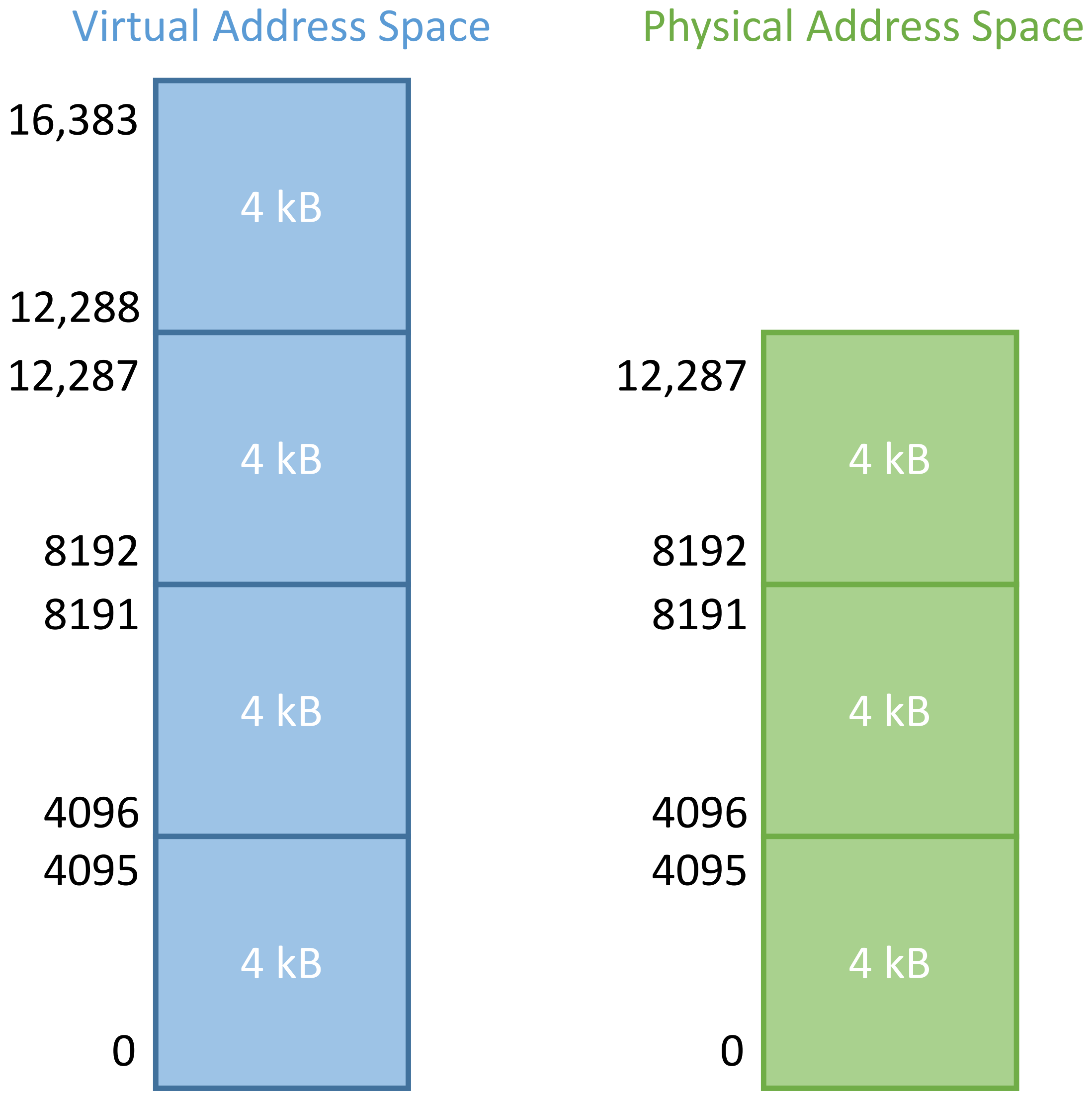
How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Q: What is the PA for VA **4**?
- 4
- 4096
- 4100
- I don't know…



How do we map addresses within a page?

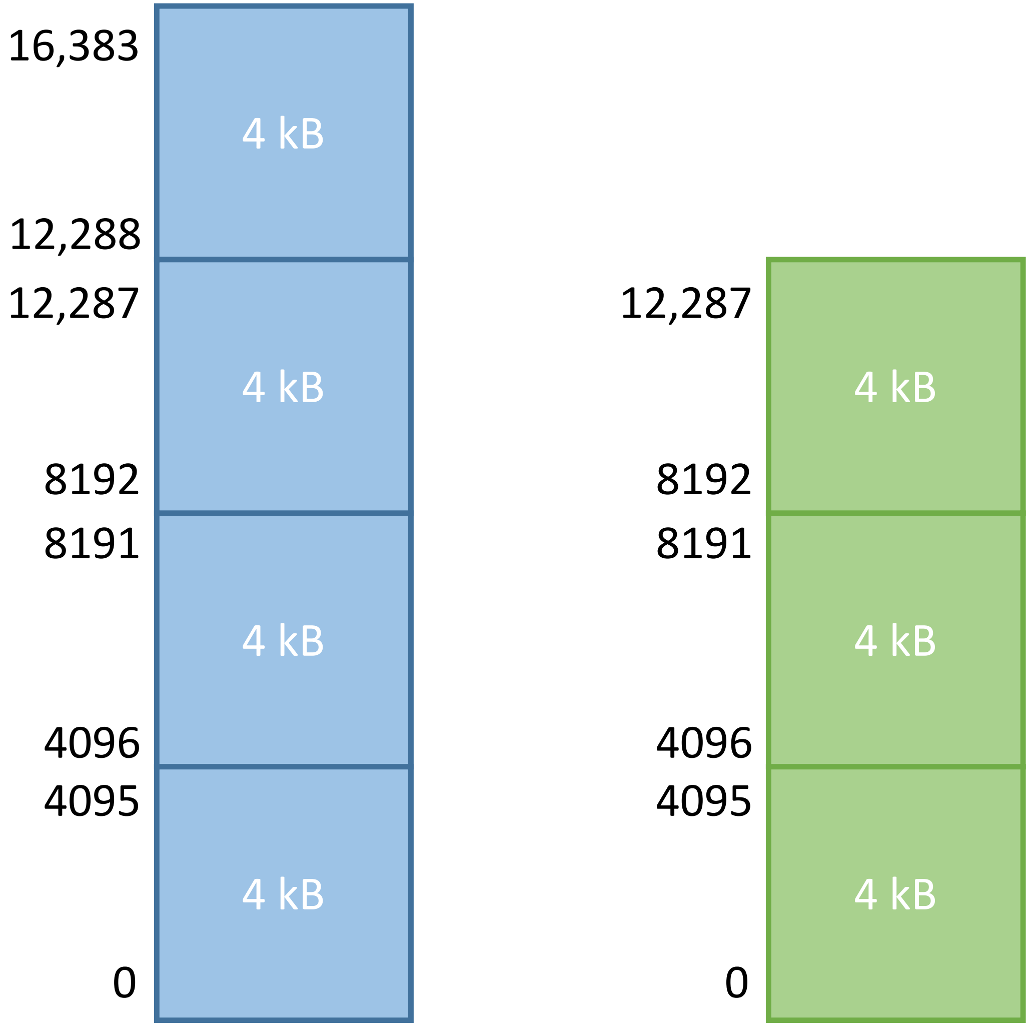
Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Q: What is the PA for VA **4**?

- 4
- 4096
- **4100**
- I don't know…



Virtual Address Space

Physical Address Space

How do we map addresses within a page?

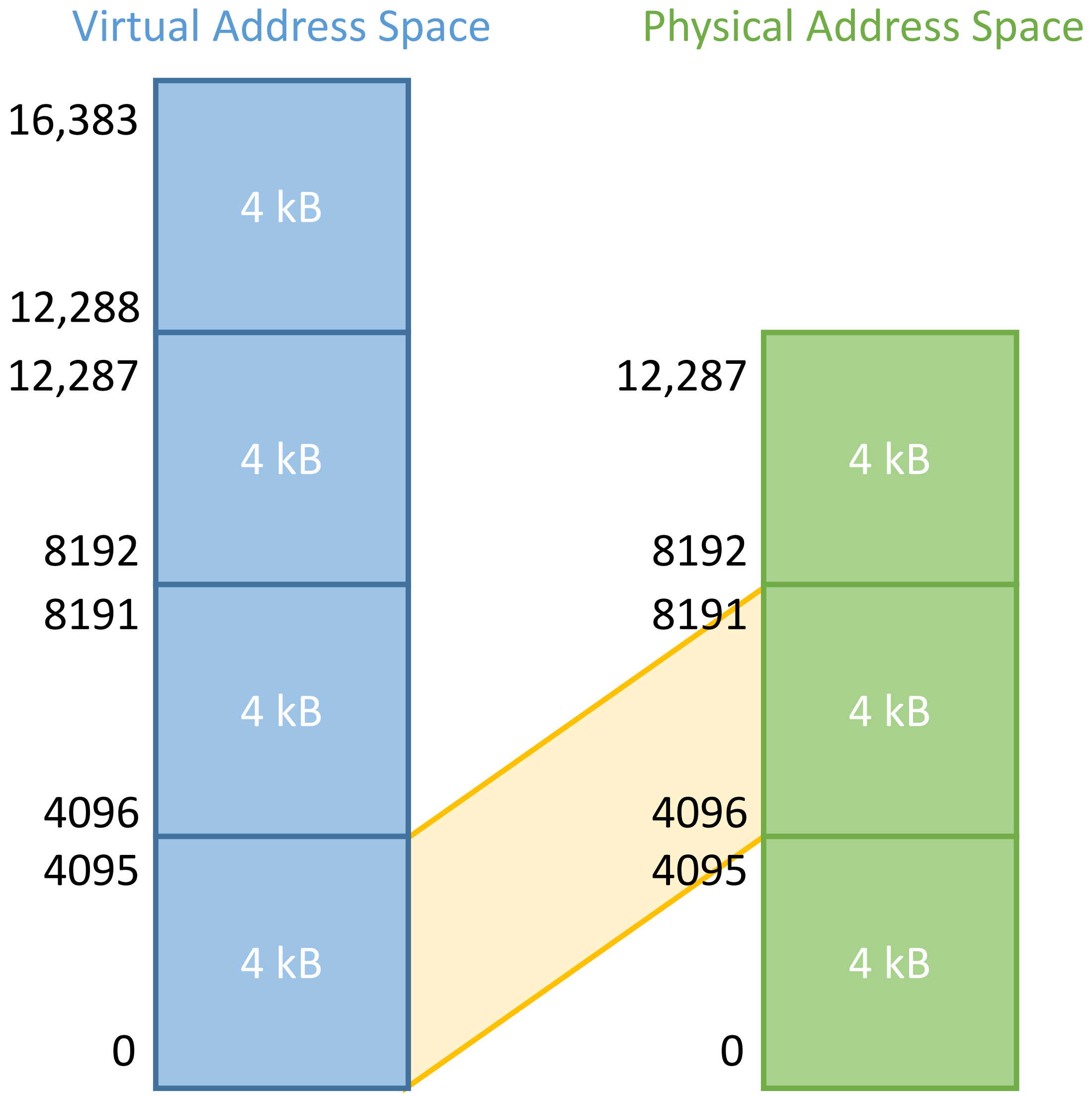
Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Q: What is the PA for VA **4**?

- 4
- 4096
- **4100**
- I don’t know…



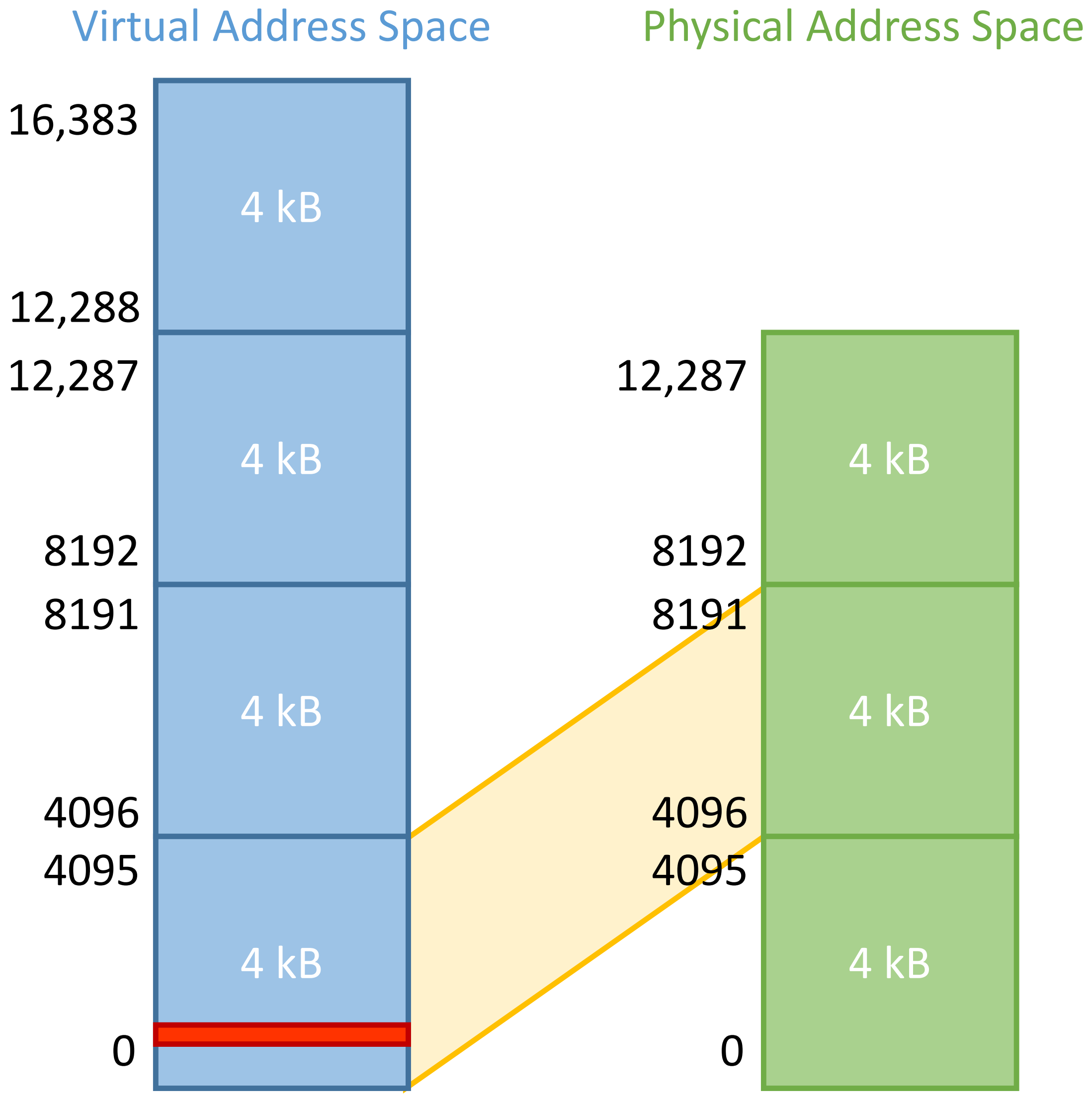
How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Q: What is the PA for VA **4**?
- 4
- 4096
- **4100**
- I don’t know…



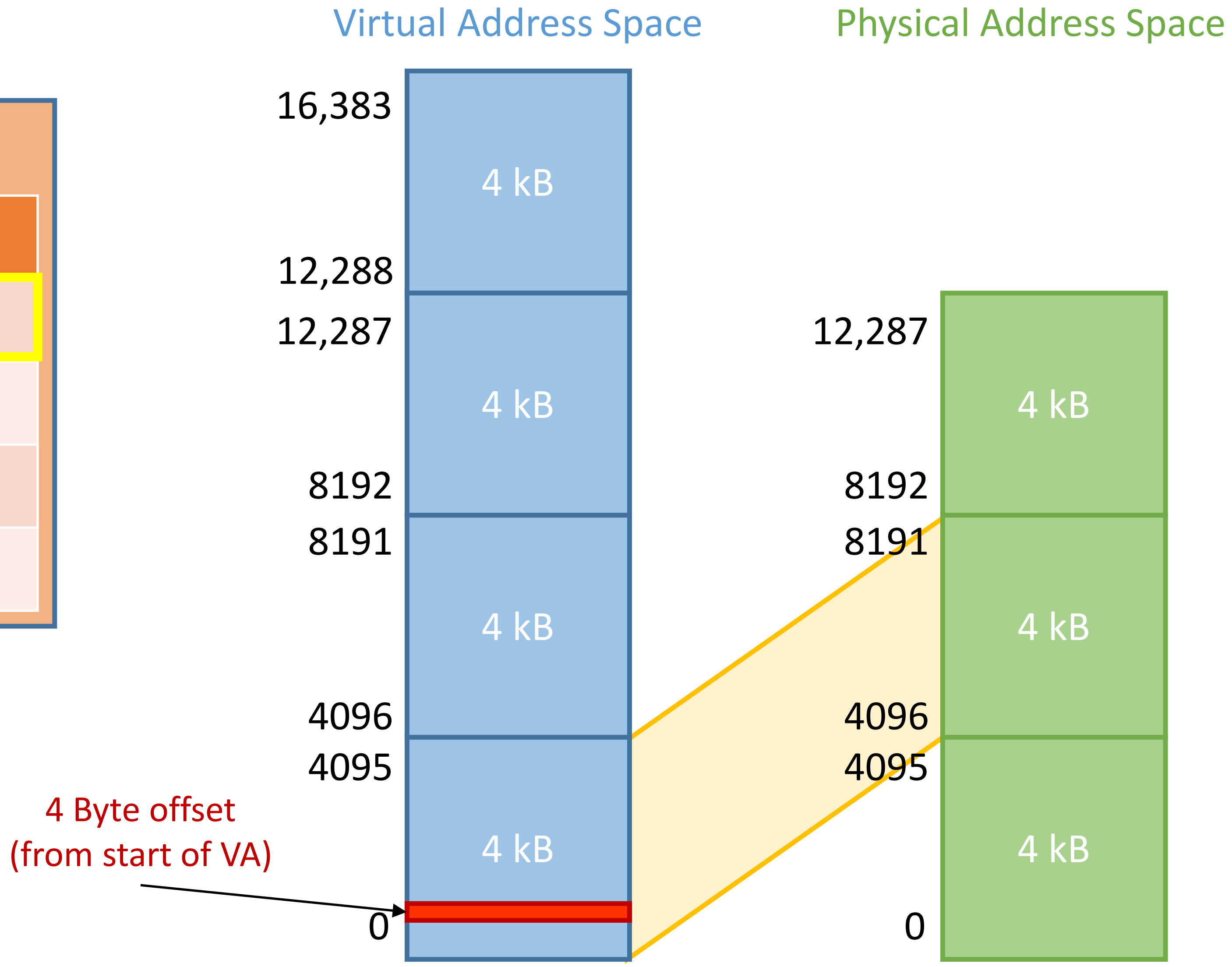
How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Q: What is the PA for VA **4**?
- 4
- 4096
- **4100**
- I don't know…



How do we map addresses within a page?

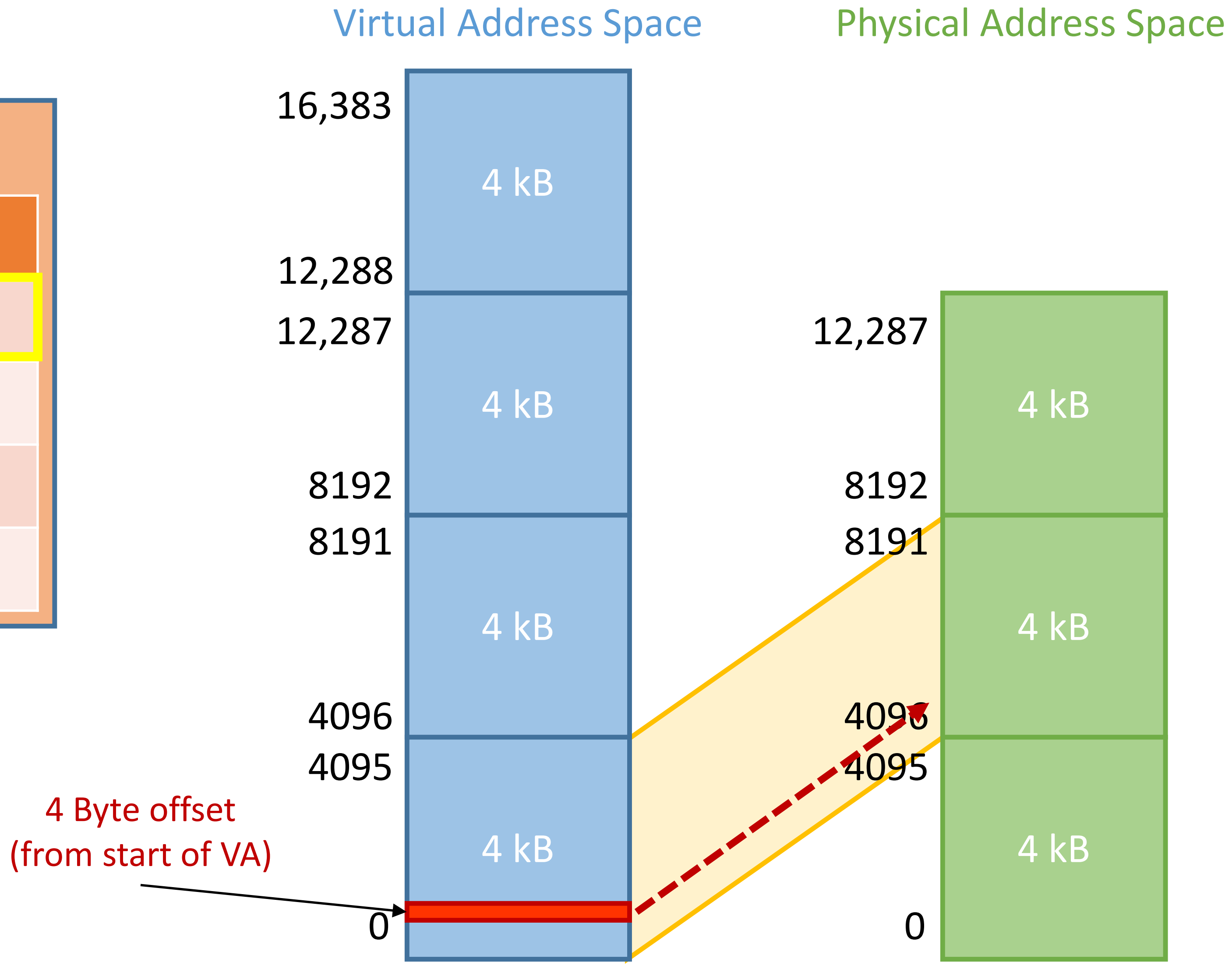
Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Q: What is the PA for VA **4**?

- 4
- 4096
- **4100**
- I don't know…



How do we map addresses within a page?

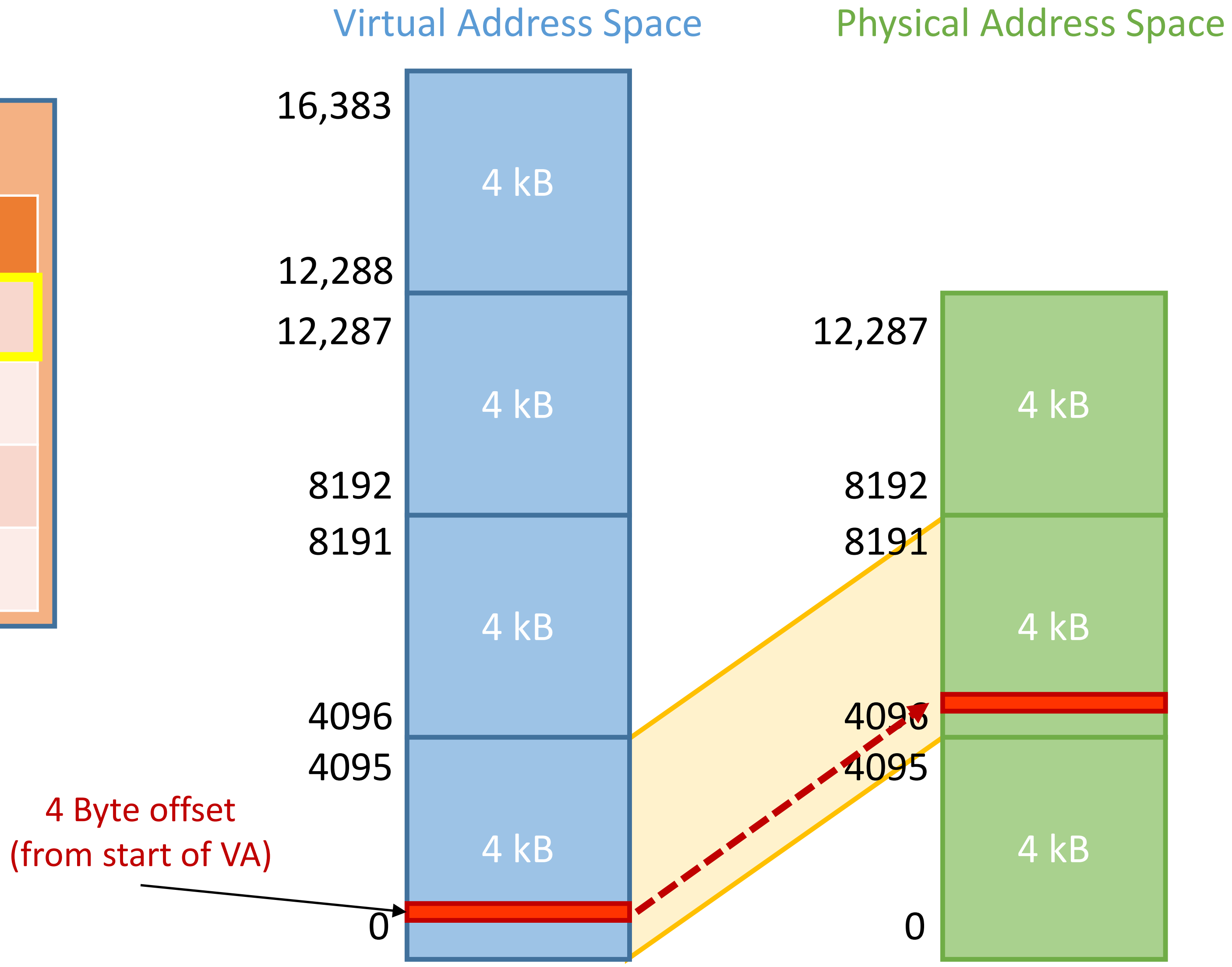
Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Q: What is the PA for VA **4**?

- 4
- 4096
- **4100**
- I don’t know…



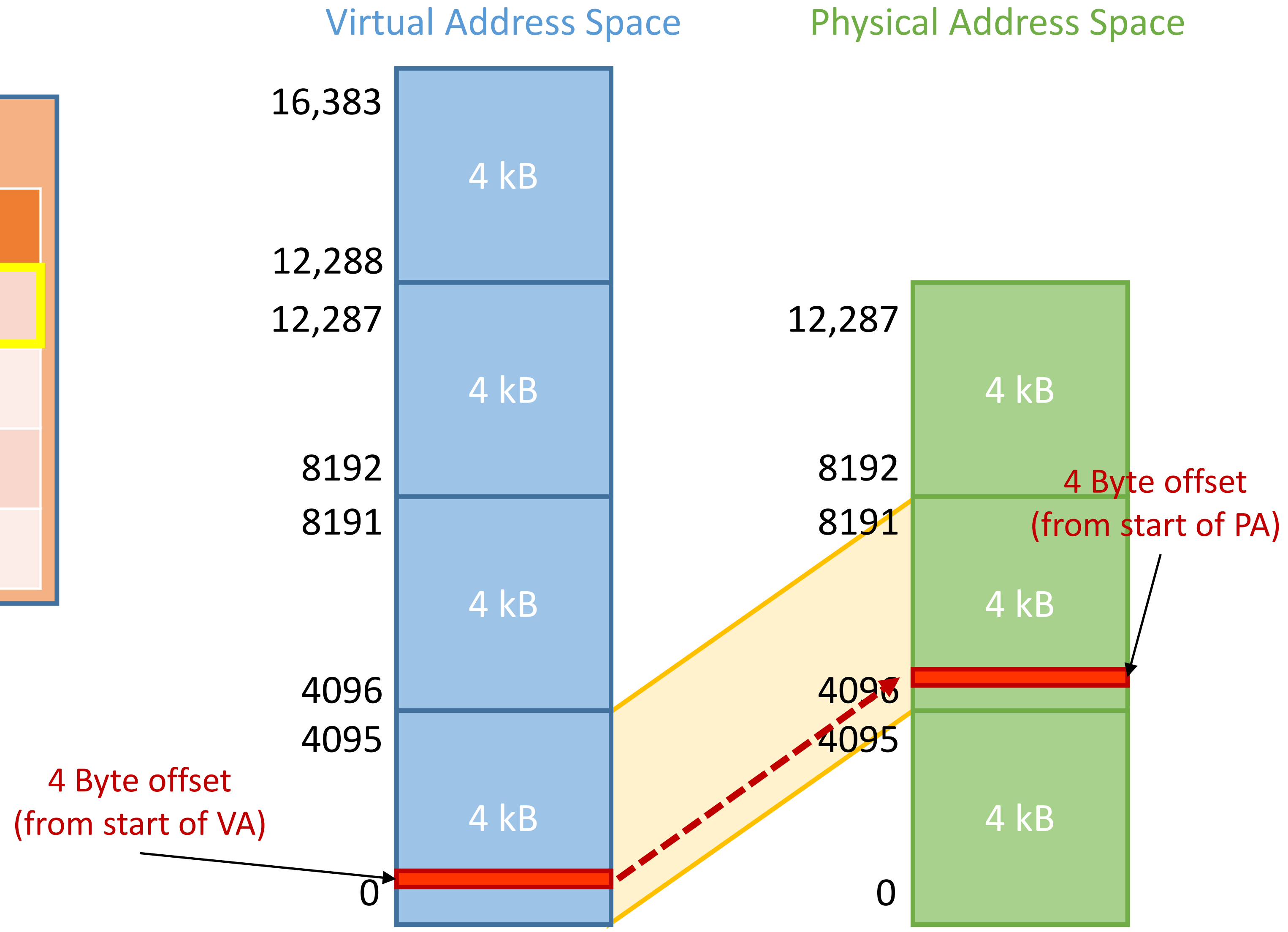
How do we map addresses within a page?

Page Table

VA to PA Mapping

VA	PA
0 – 4095	4096 - 8191

Q: What is the PA for VA **4**?
- 4
- 4096
- **4100**
- I don't know…



Address Translation

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

→

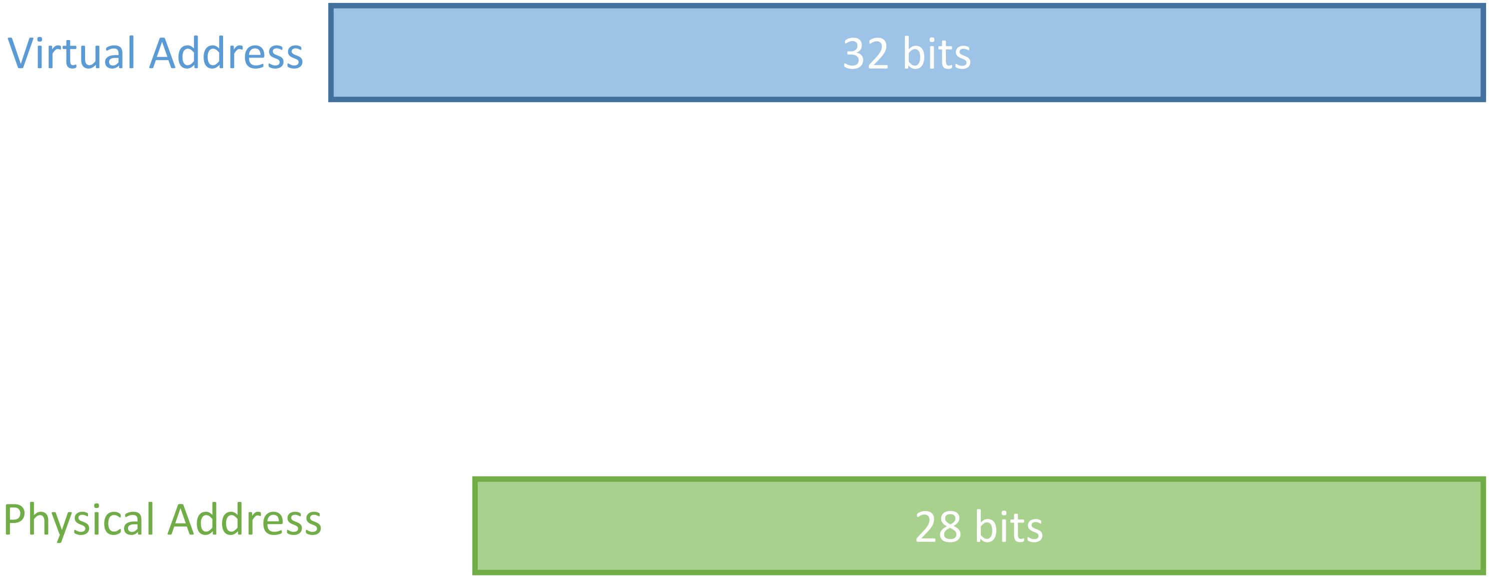
32-bit Virtual Address
28-bit Physical Address

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

⇒

32-bit Virtual Address
28-bit Physical Address

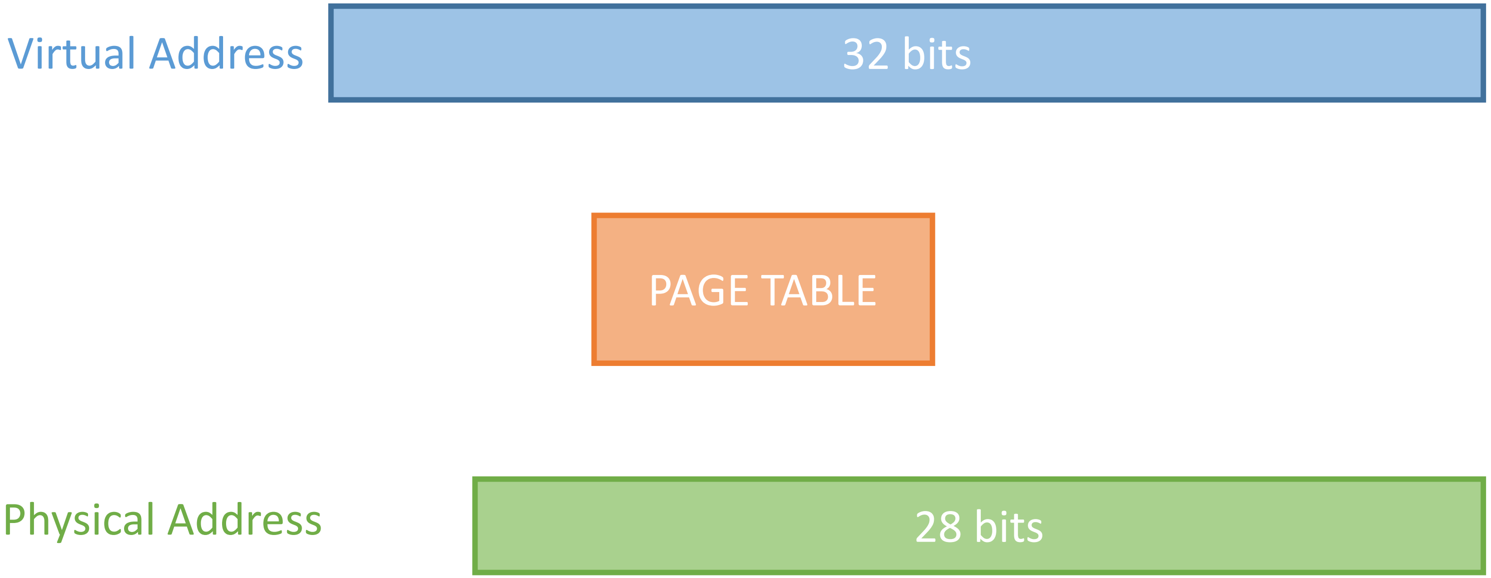


Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

→

32-bit Virtual Address
28-bit Physical Address



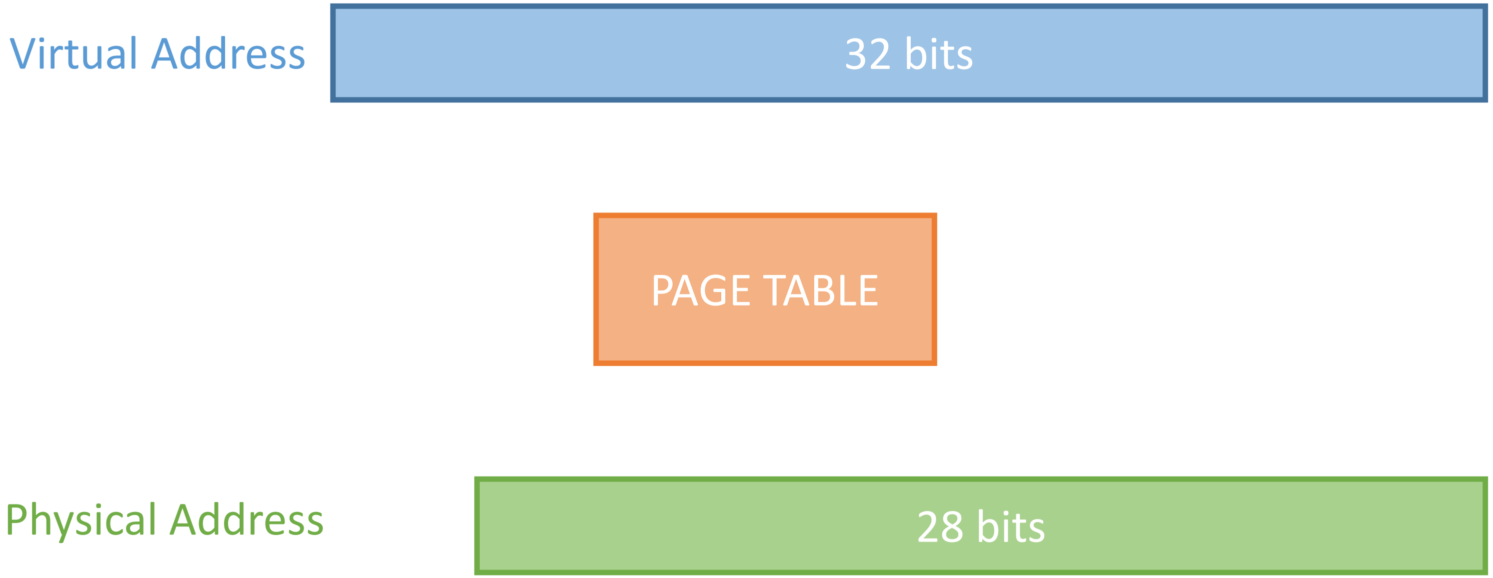
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

⇨

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Virtual Address

32 bits

PAGE TABLE

Physical Address

28 bits

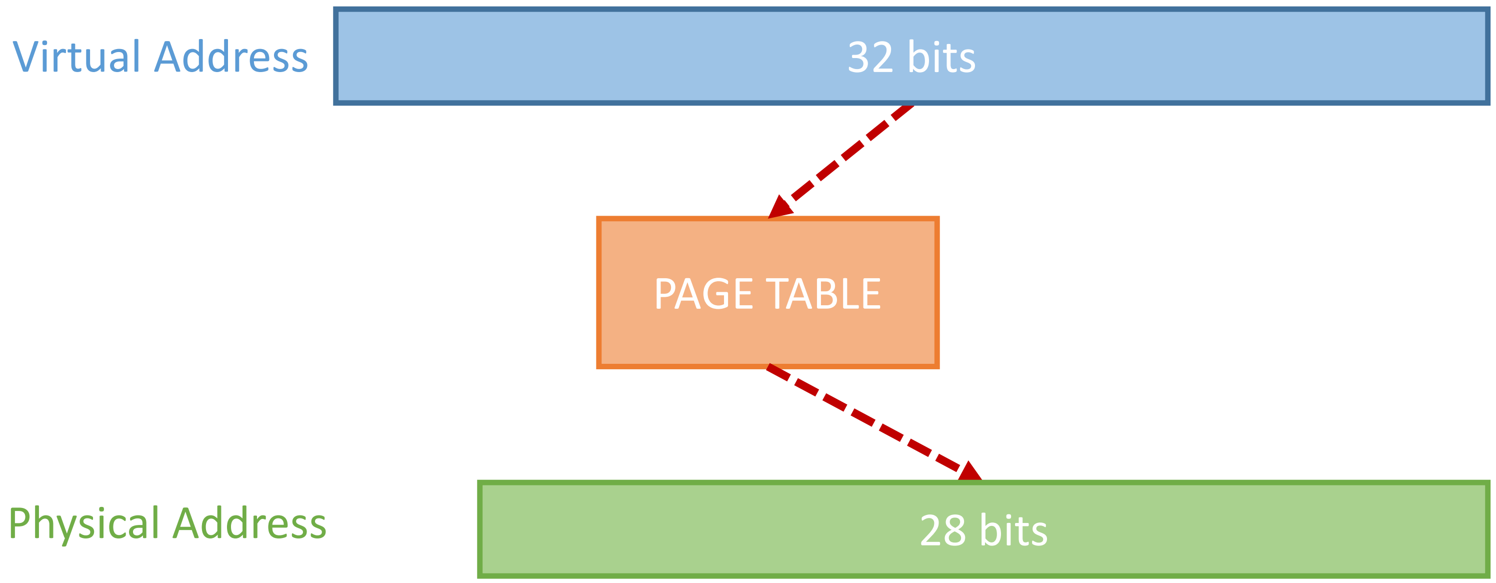
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

⇒

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

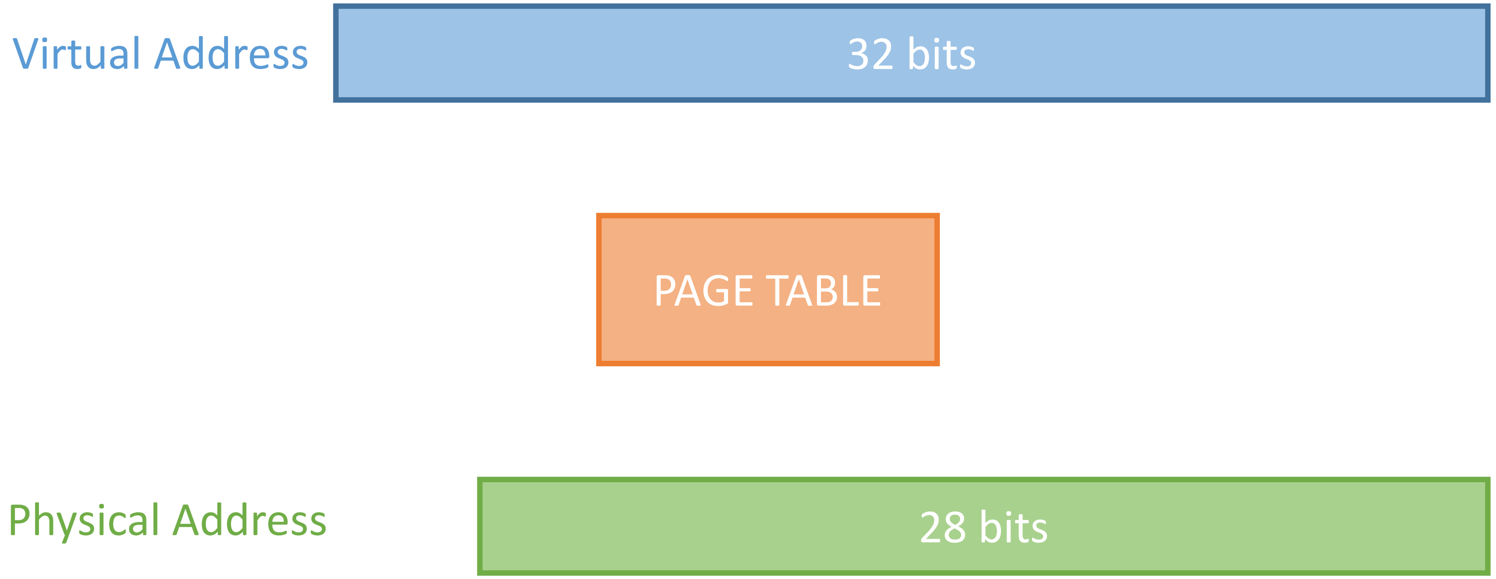


Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



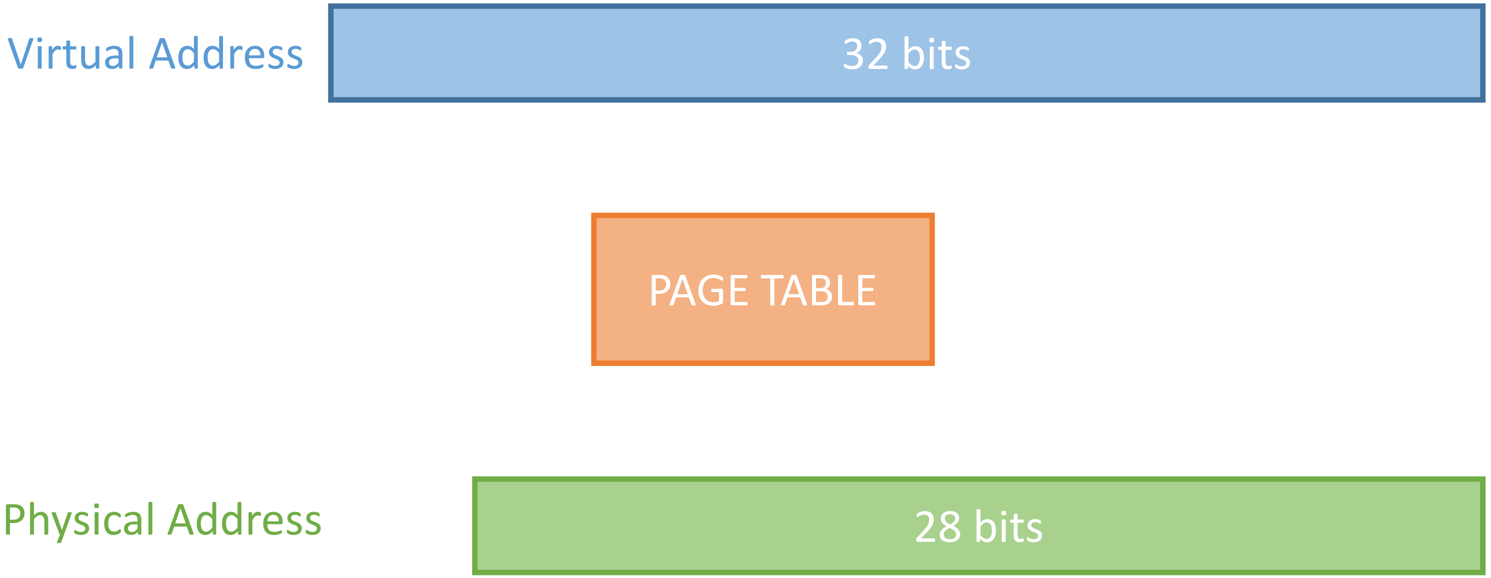
Each Page Table Entry handles
4 kB of address space…

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

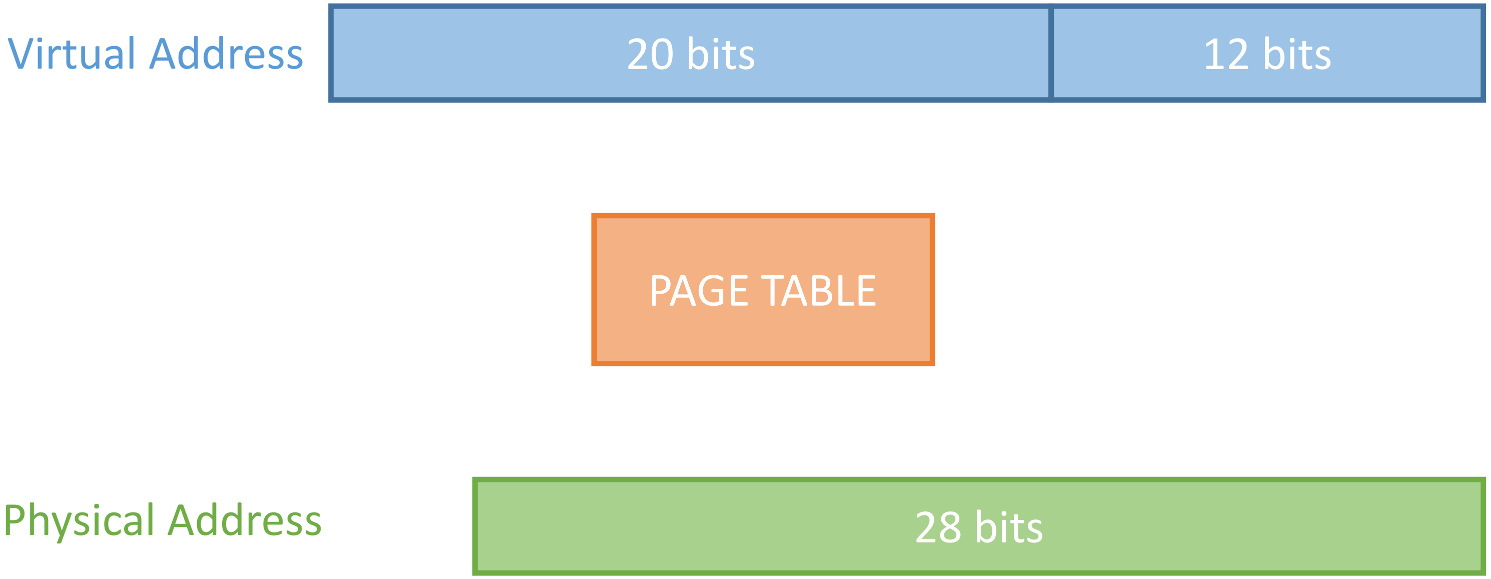
For every page, 4096 addresses [12 bits] do not get translated.

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

For every page, 4096 addresses [12 bits] do not get translated.

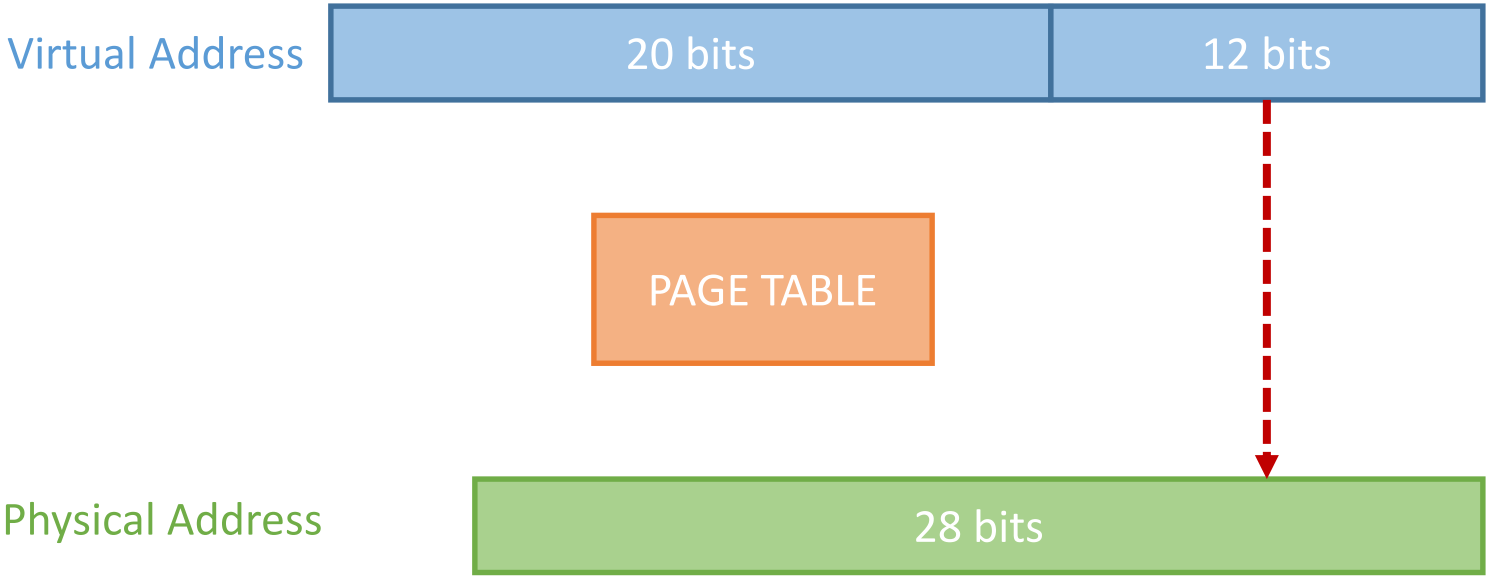
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

⇨

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

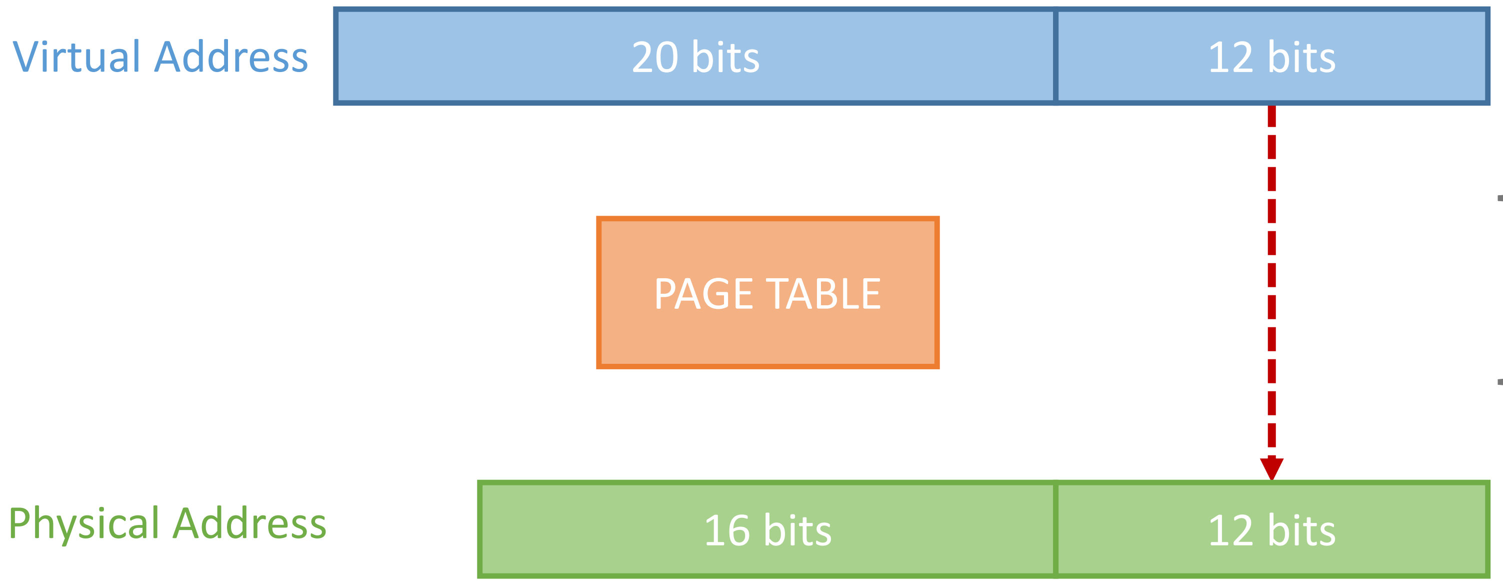
For every page, 4096 addresses [12 bits] do not get translated.

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

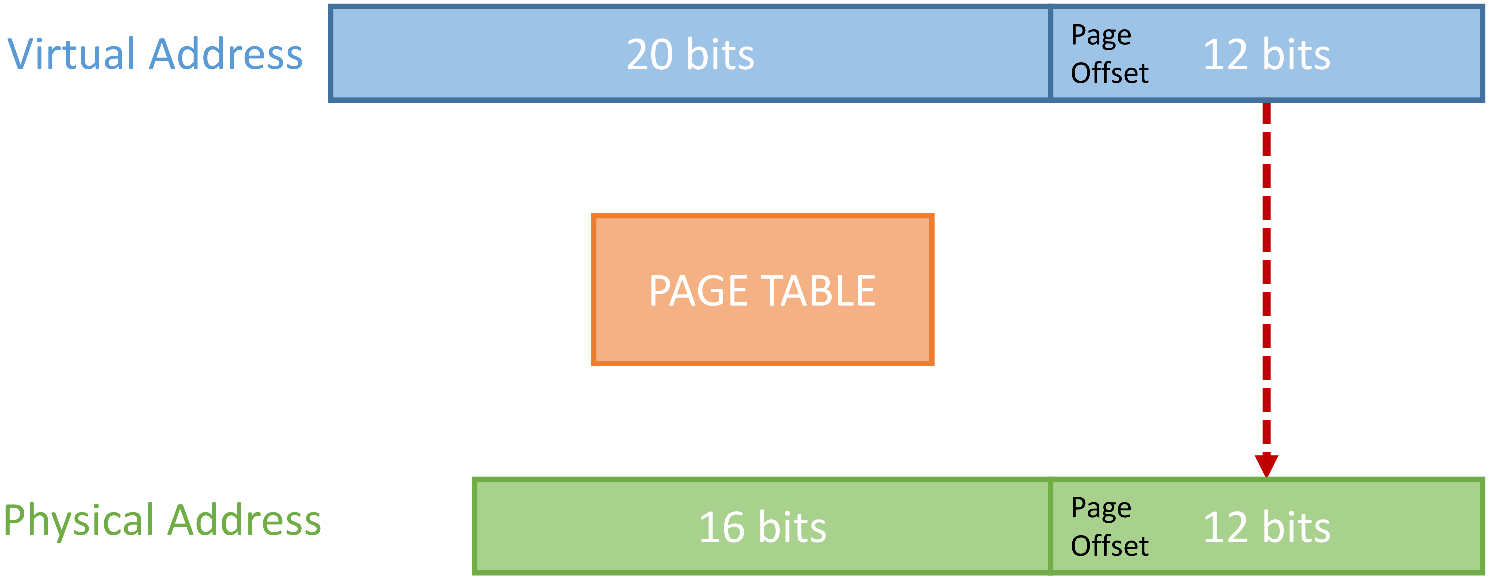
For every page, 4096 addresses [12 bits] do not get translated.

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

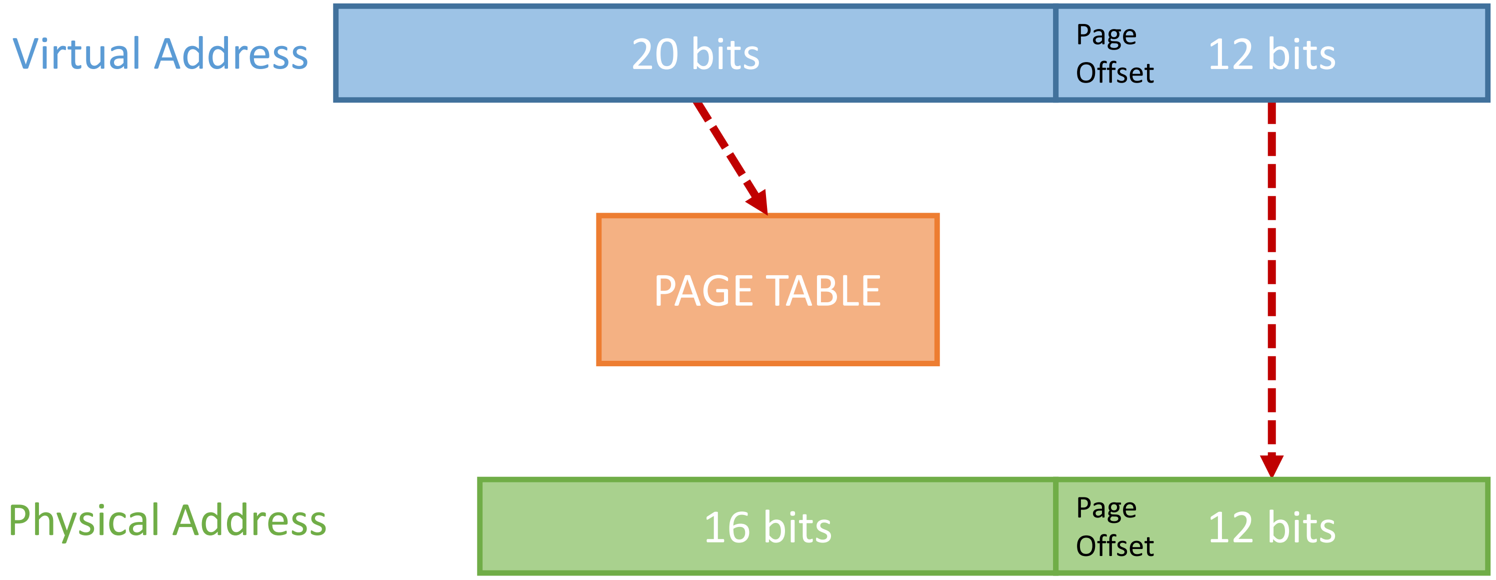
For every page, 4096 addresses [12 bits] do not get translated.

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

For every page, 4096 addresses [12 bits] do not get translated.

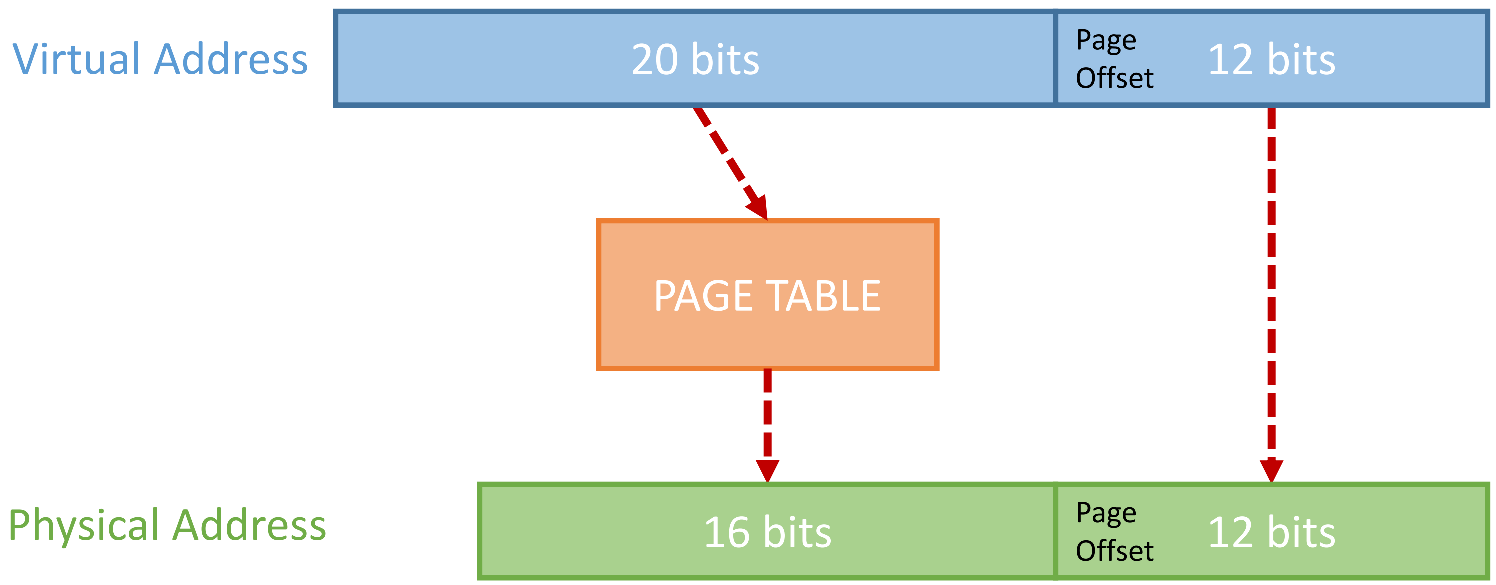
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

⇨

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

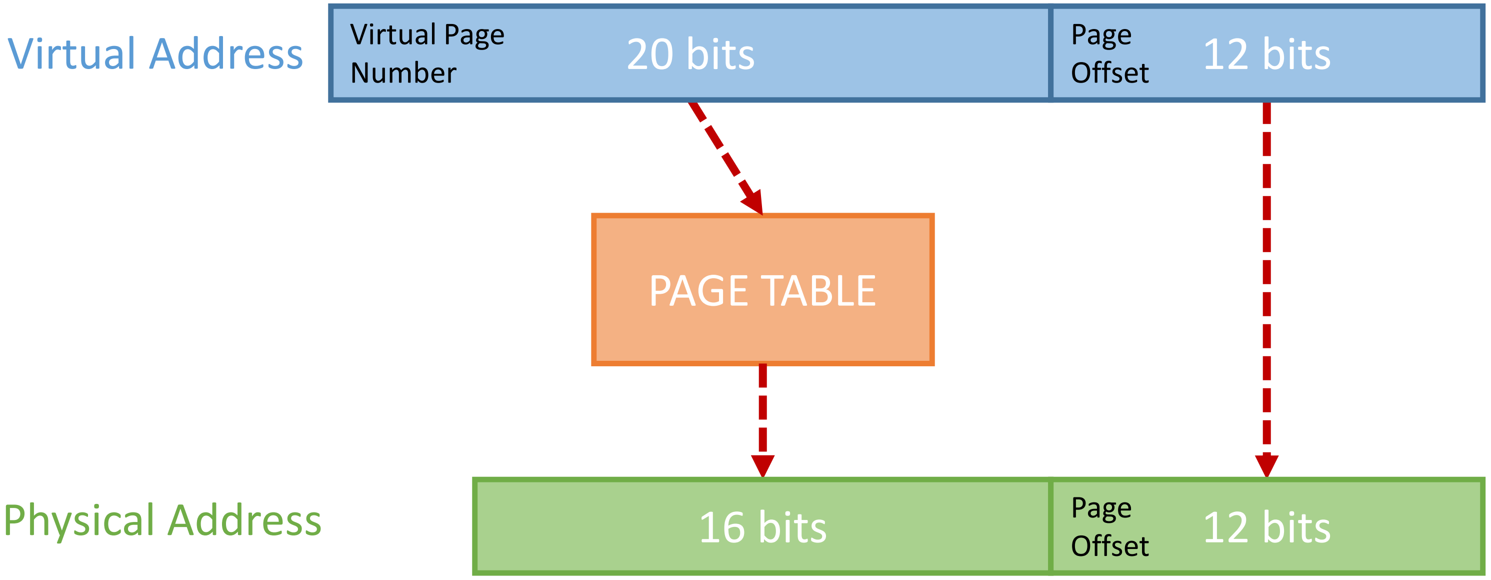
For every page, 4096 addresses [12 bits] do not get translated.

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

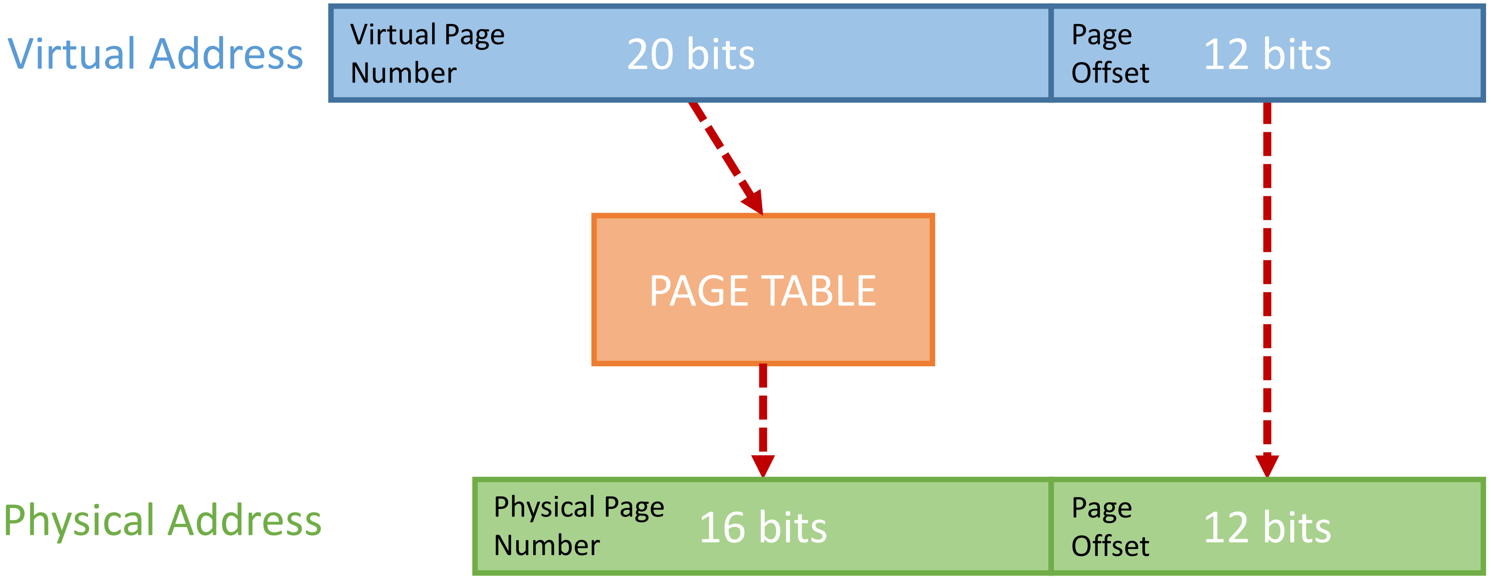
For every page, 4096 addresses [12 bits] do not get translated.

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Each Page Table Entry handles 4 kB of address space…

For every page, 4096 addresses [12 bits] do not get translated.

Illustration from the textbook

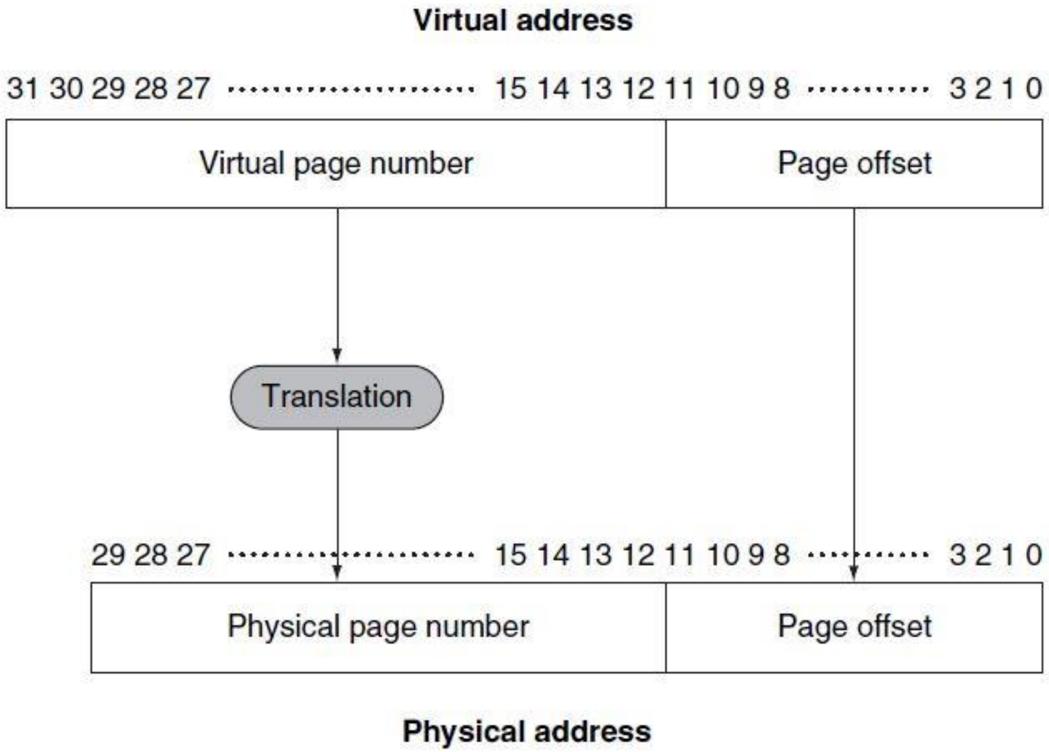


FIGURE 5.26 Mapping from a virtual to a physical address. The page size is 2^{12} = 4 KiB. The number of physical pages allowed in memory is 2^{18}, since the physical page number has 18 bits in it. Thus, main memory can have at most 1 GiB, while the virtual address space is 4 GiB.

Illustration from the textbook

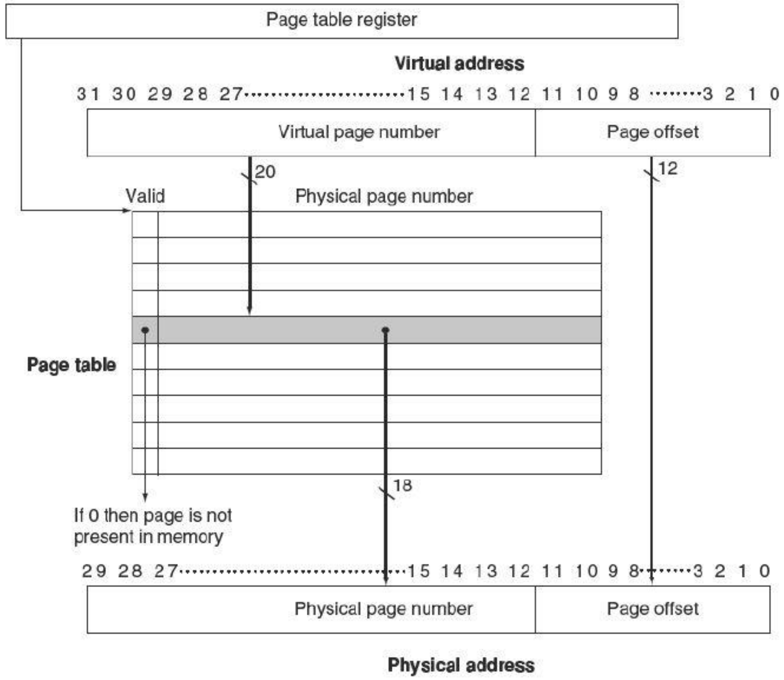
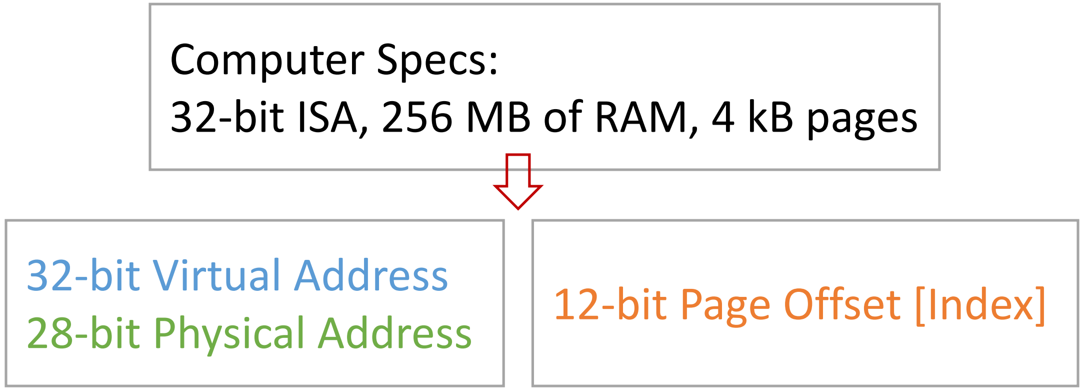


FIGURE 5.27 The page table is indexed with the virtual page number to obtain the corresponding portion of the physical address. We assume a 32-bit address. The page table pointer gives the starting address of the page table. In this figure, the page size is 2^{12} bytes, or 4 KiB. The virtual address space is 2^{32} bytes, or 4 GiB, and the physical address space is 2^{30} bytes, which allows main memory of up to 1 GiB. The number of entries in the page table is 2^{20}, or 1 million entries. The valid bit for each entry indicates whether the mapping is legal. If it is off, then the page is not present in memory. Although the page table entry shown here need only be 19 bits wide, it would typically be rounded up to 32 bits for ease of indexing. The extra bits would be used to store additional information that needs to be kept on a per-page basis, such as protection.

Address Translation



Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

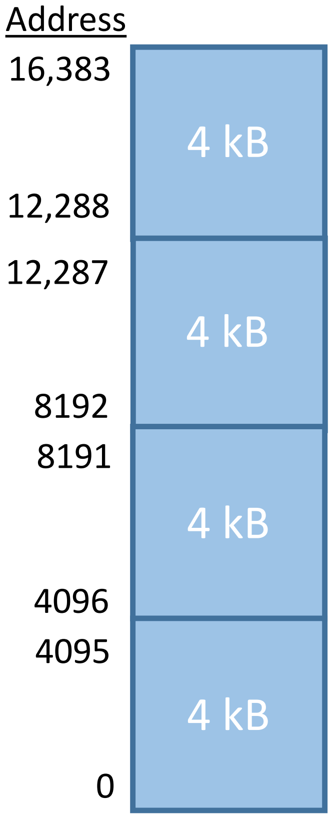
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

Virtual Address Space



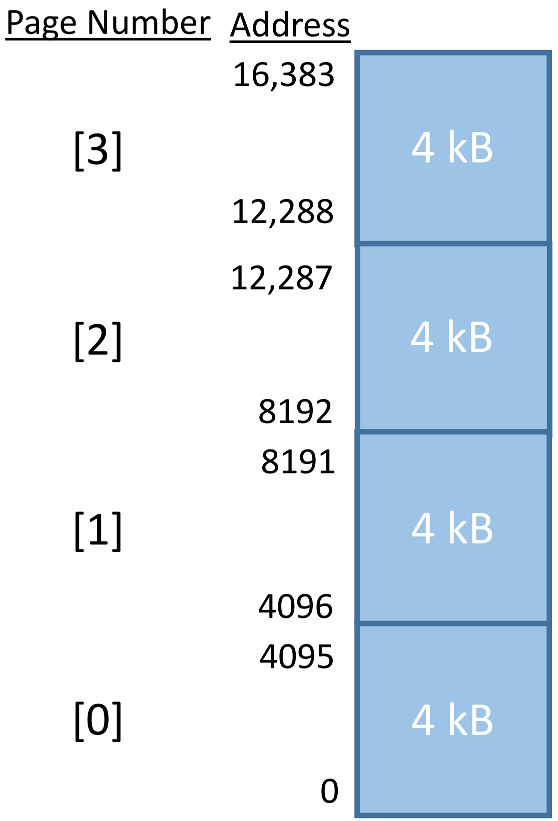
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

Virtual Address Space



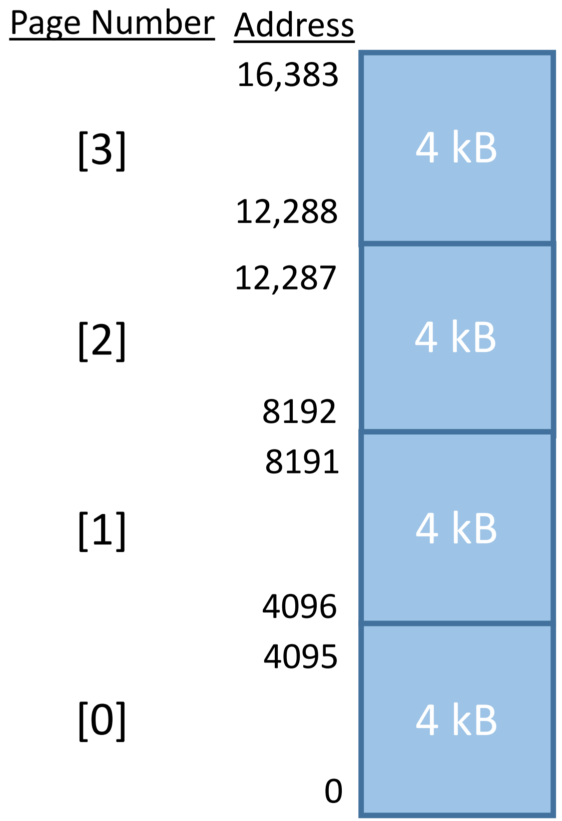
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

Virtual Address Space



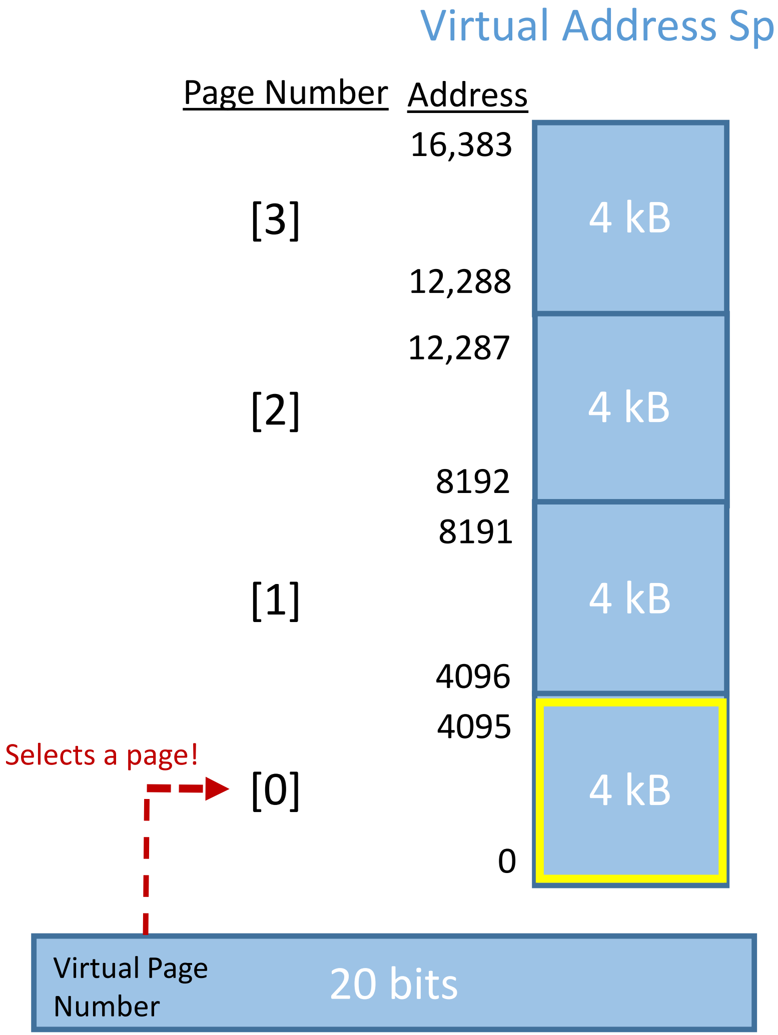
Virtual Page Number 20 bits

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

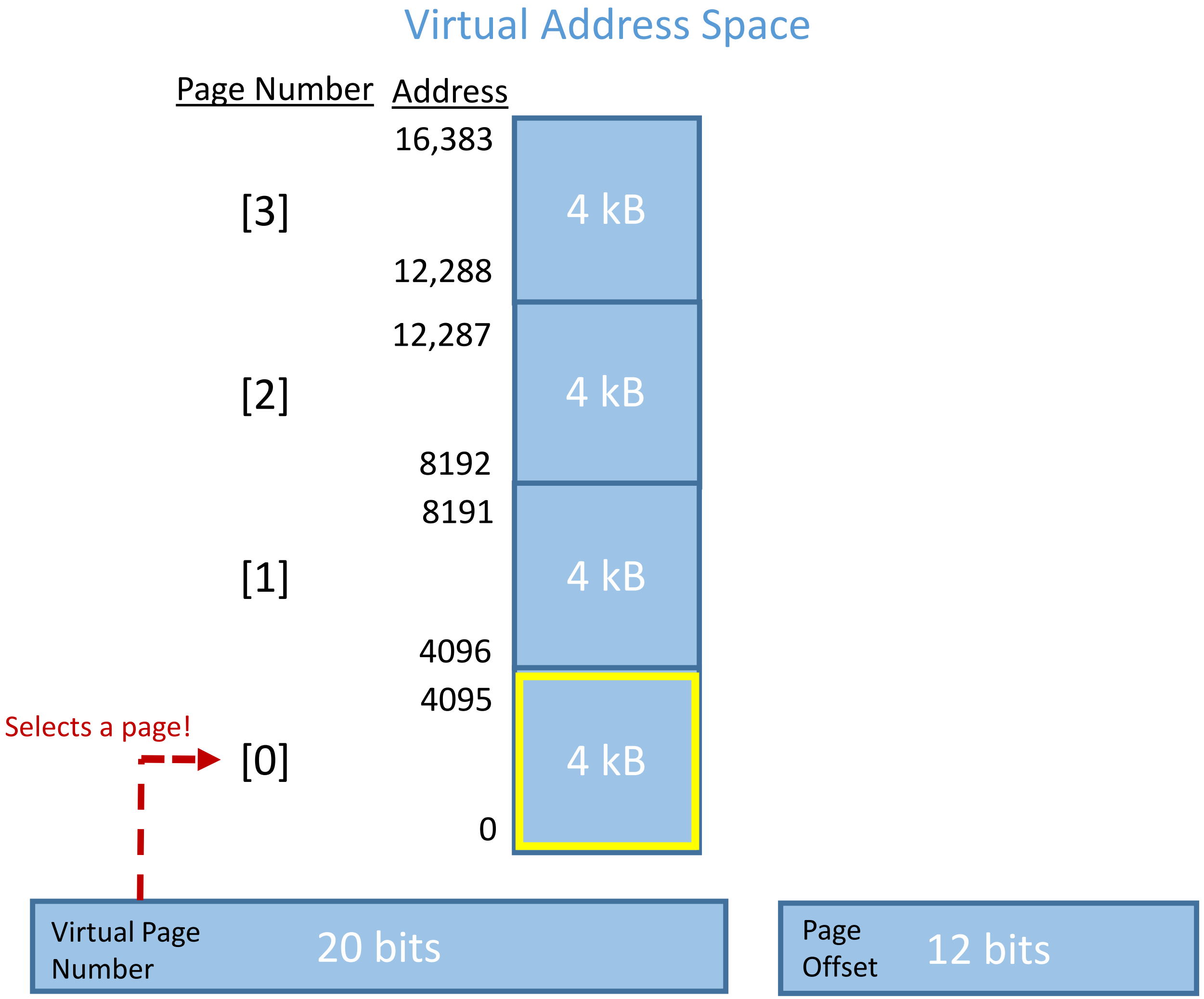


Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



Virtual Address Space

Page Number | Address

16,383
[3] 4 kB
12,288
12,287
[2] 4 kB
8192
8191
[1] 4 kB
4096
4095
[0] 4 kB
0

Selects a page!

Virtual Page Number 20 bits

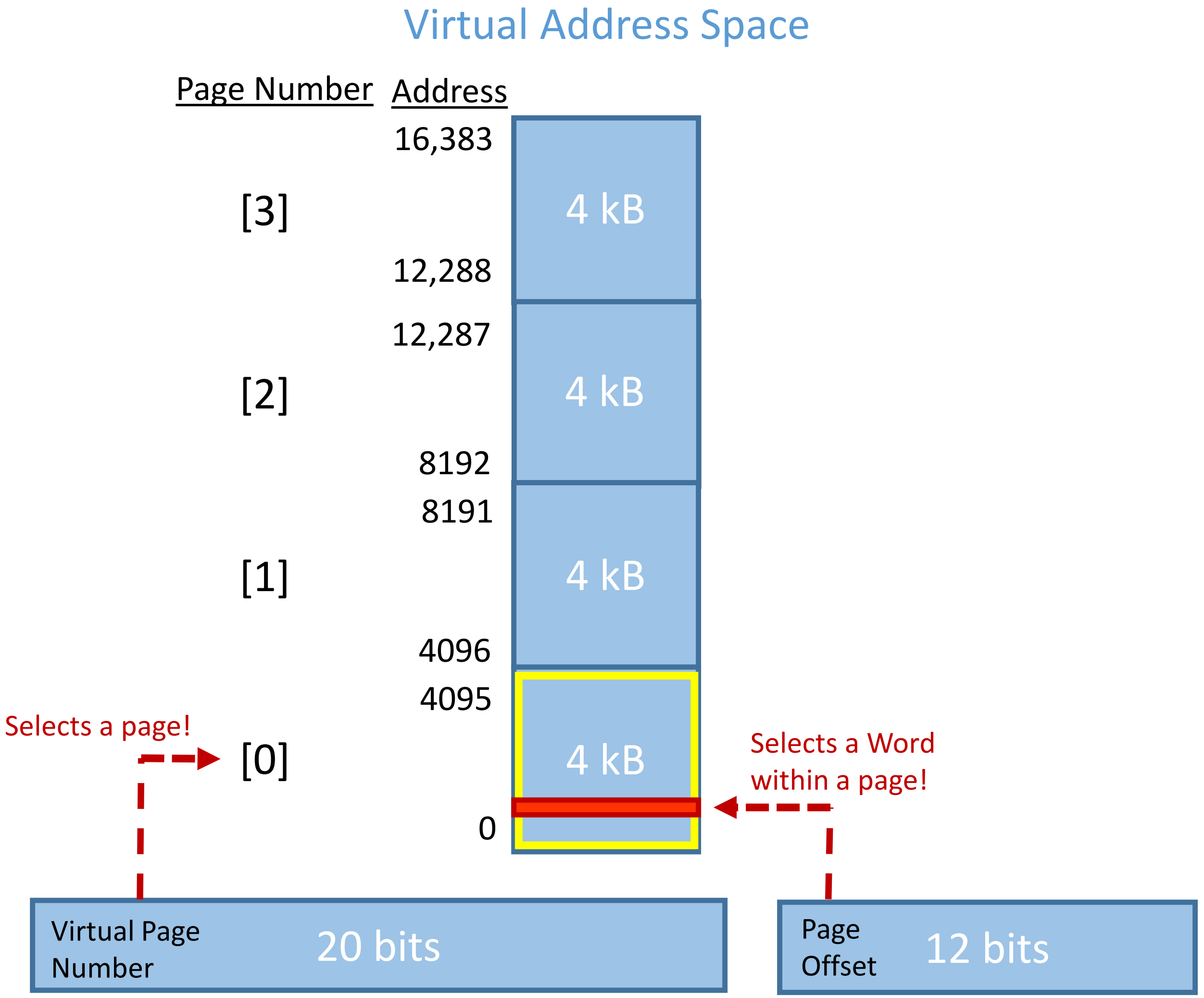
Page Offset 12 bits

Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]



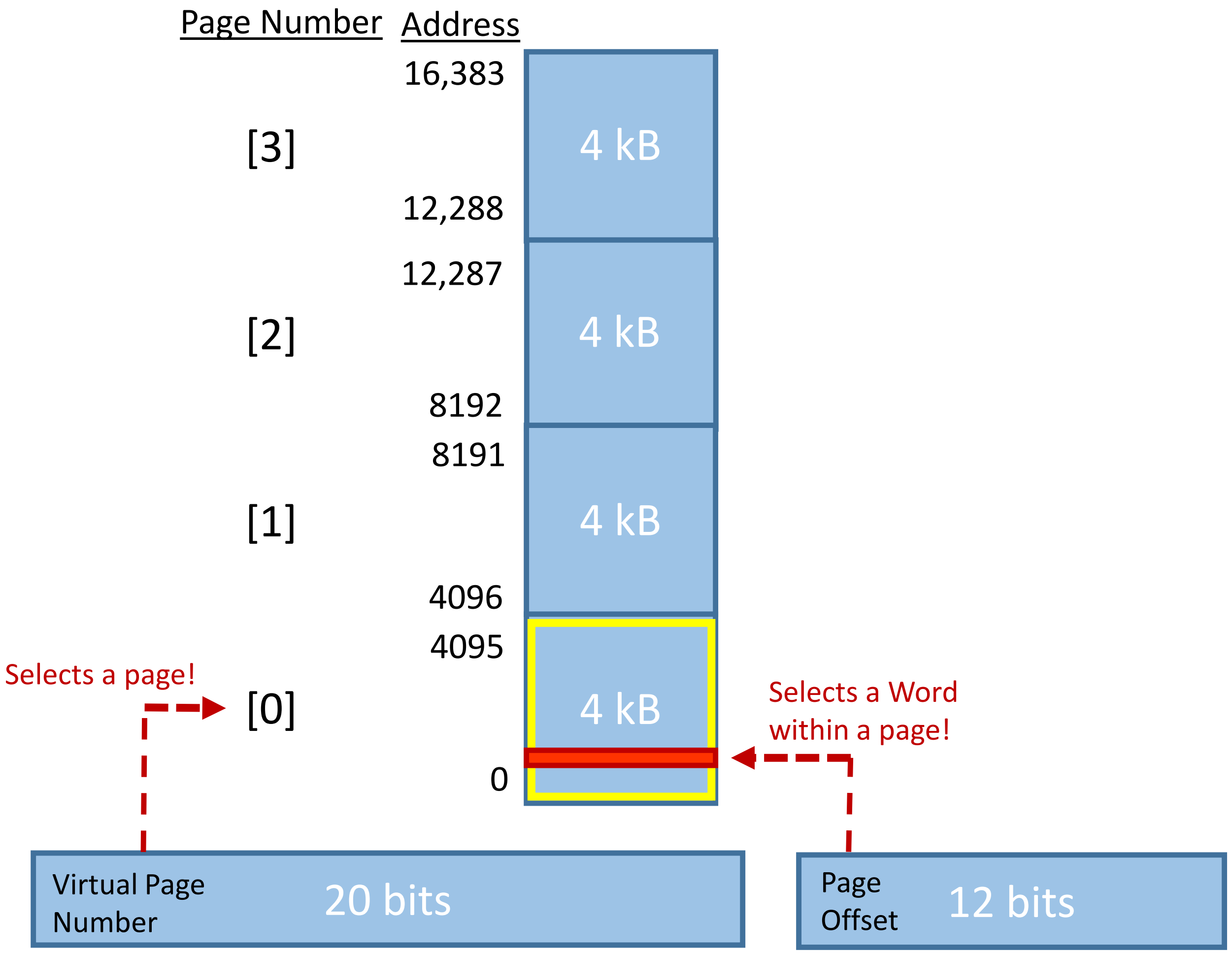
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

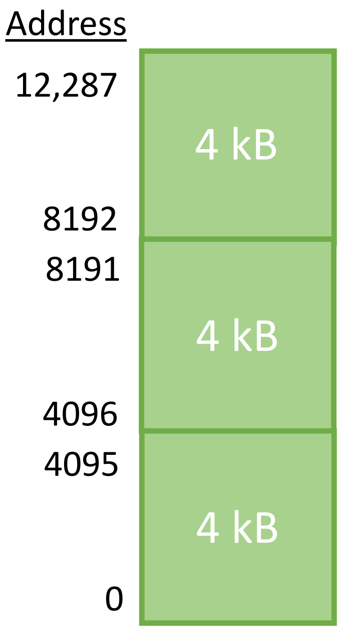
32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

Virtual Address Space



Physical Address Space



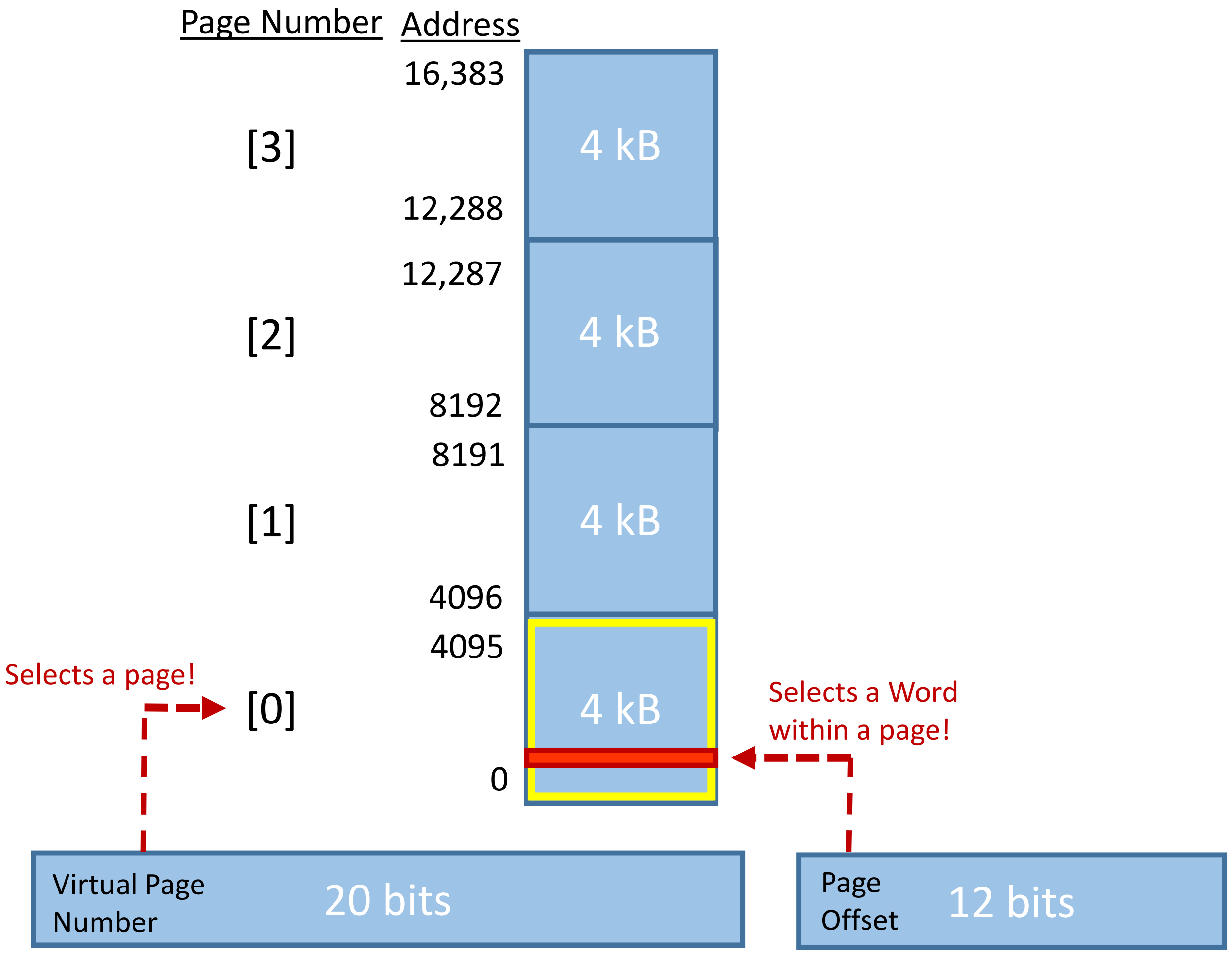
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

Virtual Address Space



Page Number | Address

16,383
[3] 4 kB
12,288
12,287
[2] 4 kB
8192
8191
[1] 4 kB
4096
4095
[0] 4 kB
0

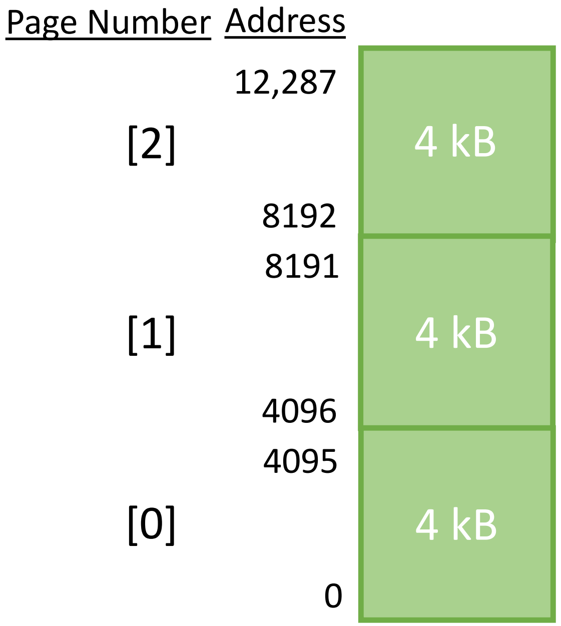
Selects a page!

Selects a Word within a page!

Virtual Page Number 20 bits

Page Offset 12 bits

Physical Address Space



Page Number | Address

12,287
[2] 4 kB
8192
8191
[1] 4 kB
4096
4095
[0] 4 kB
0

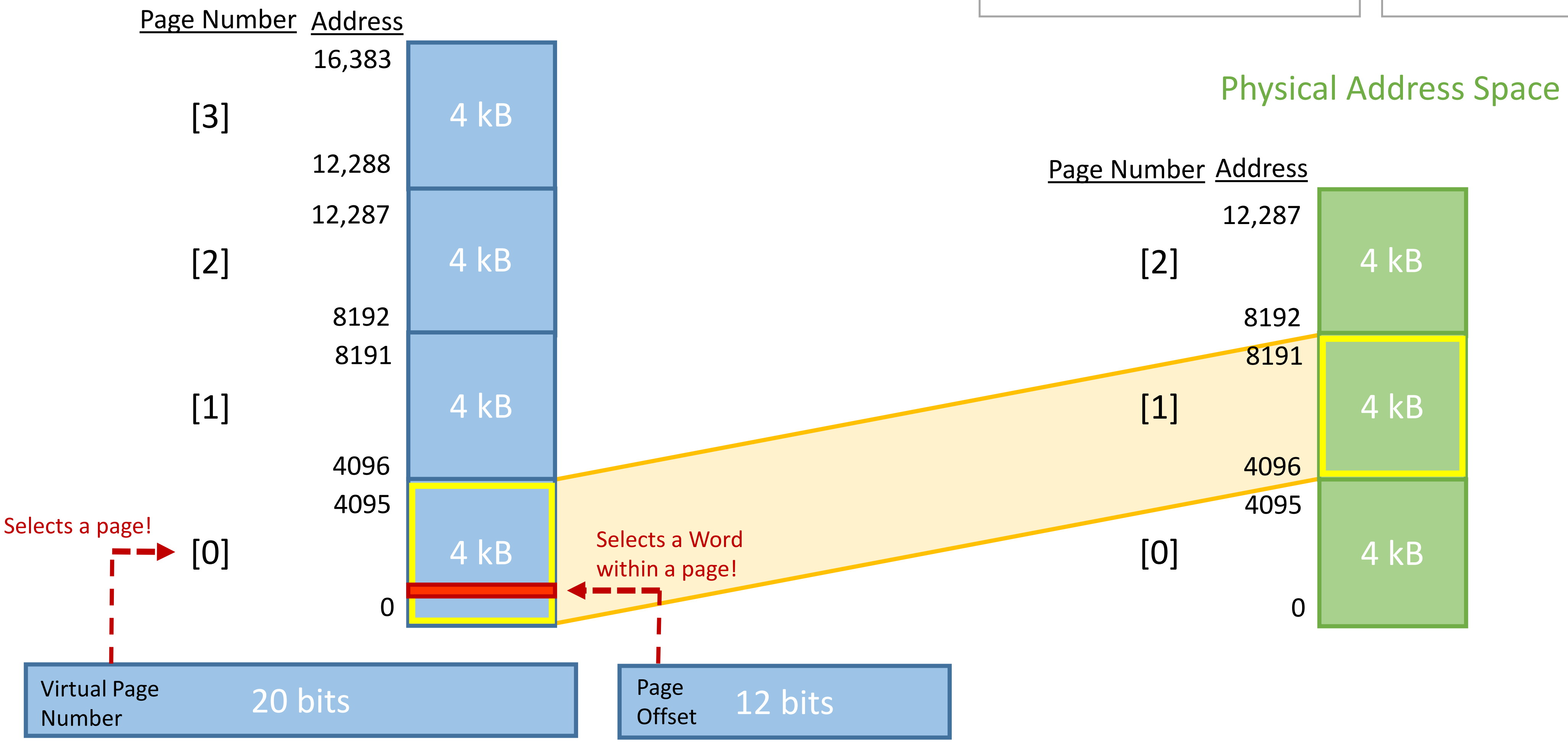
Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

Virtual Address Space

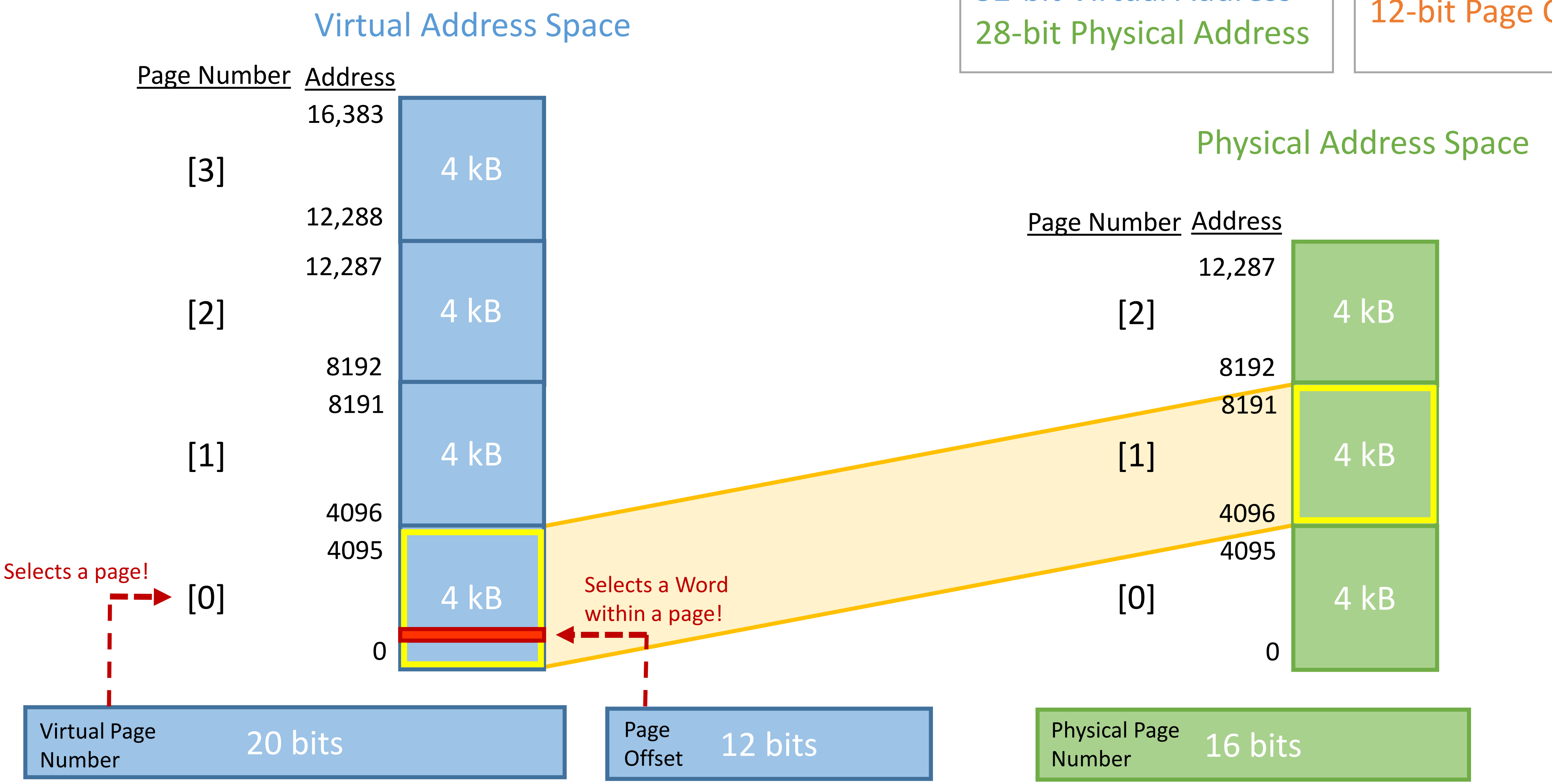


Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

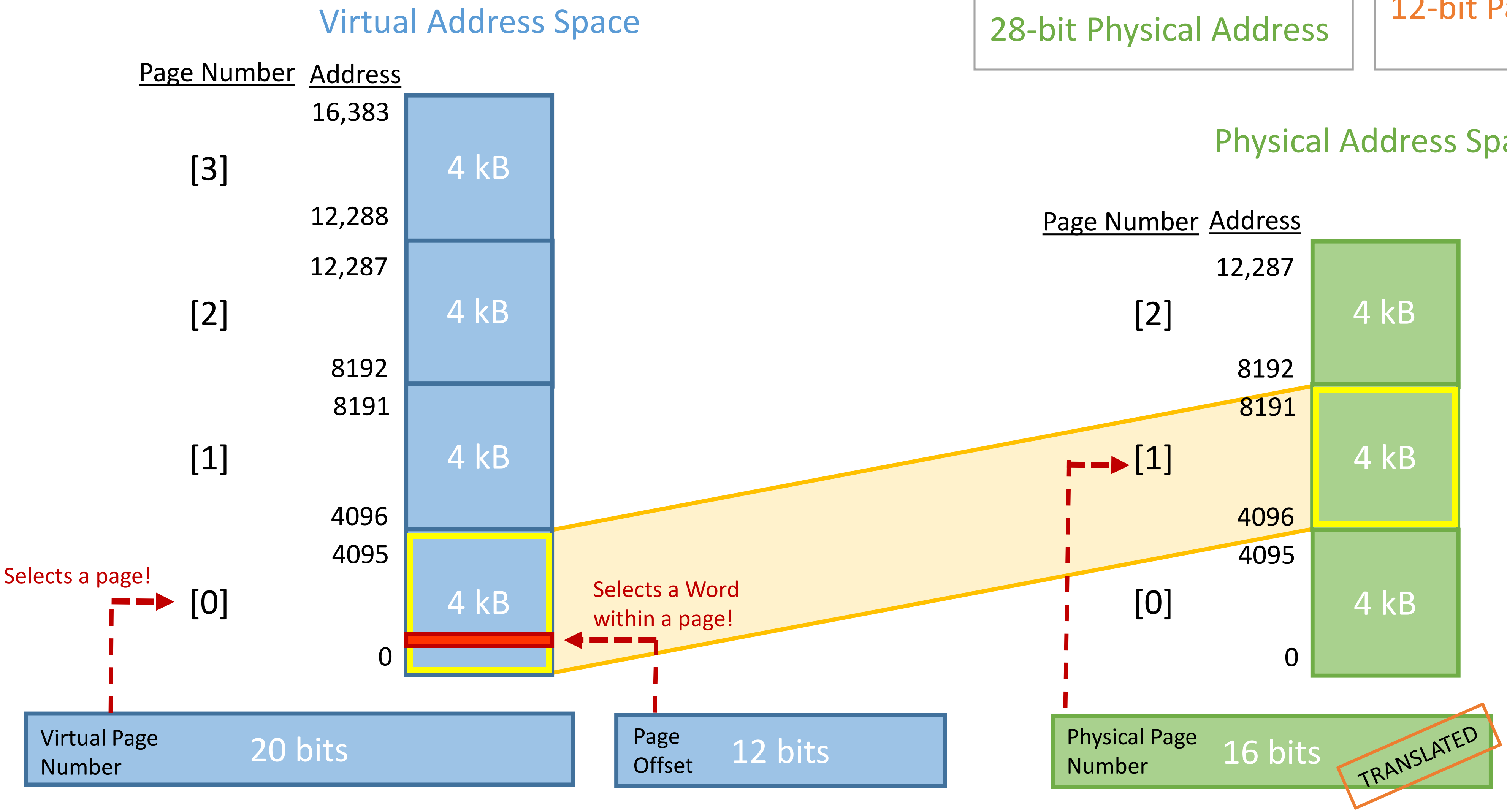


Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

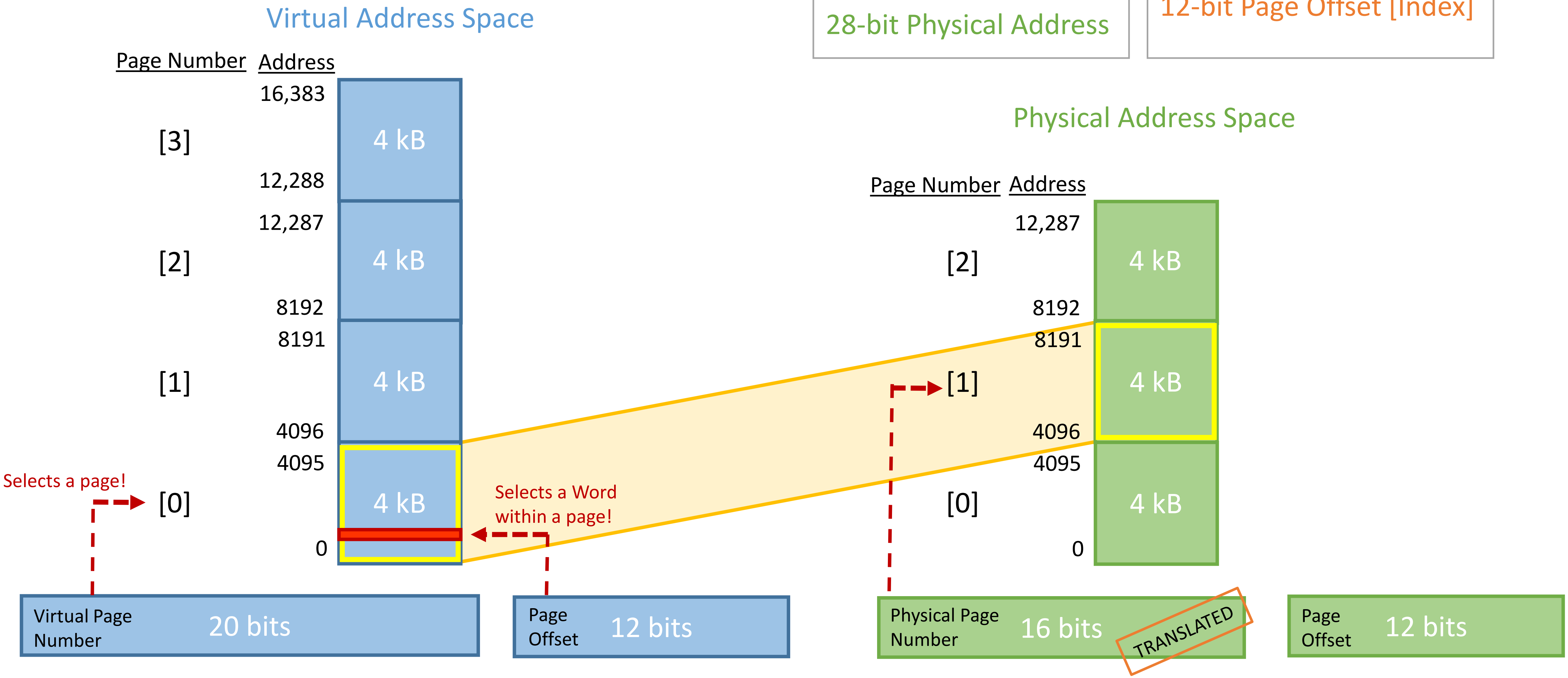


Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

12-bit Page Offset [Index]

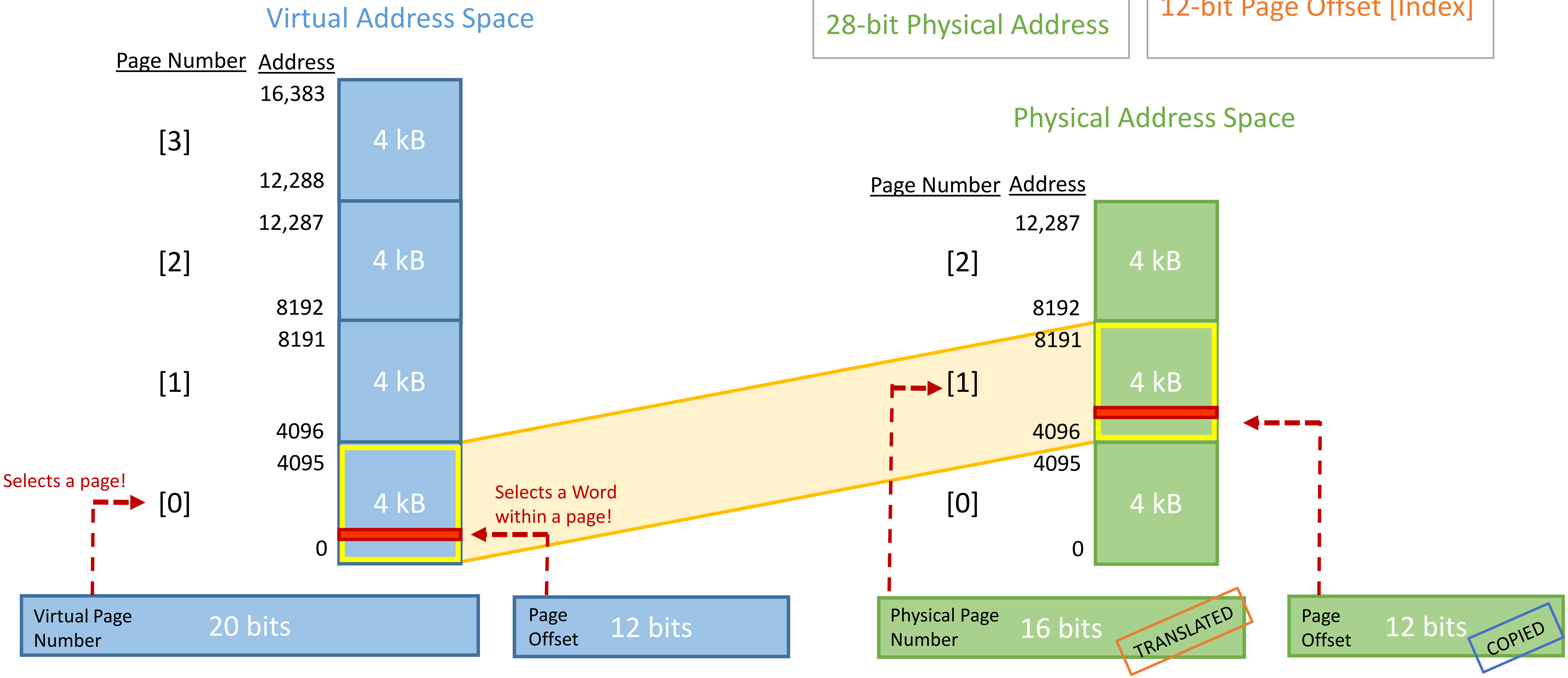


Address Translation

Computer Specs:
32-bit ISA, 256 MB of RAM, 4 kB pages

32-bit Virtual Address
28-bit Physical Address

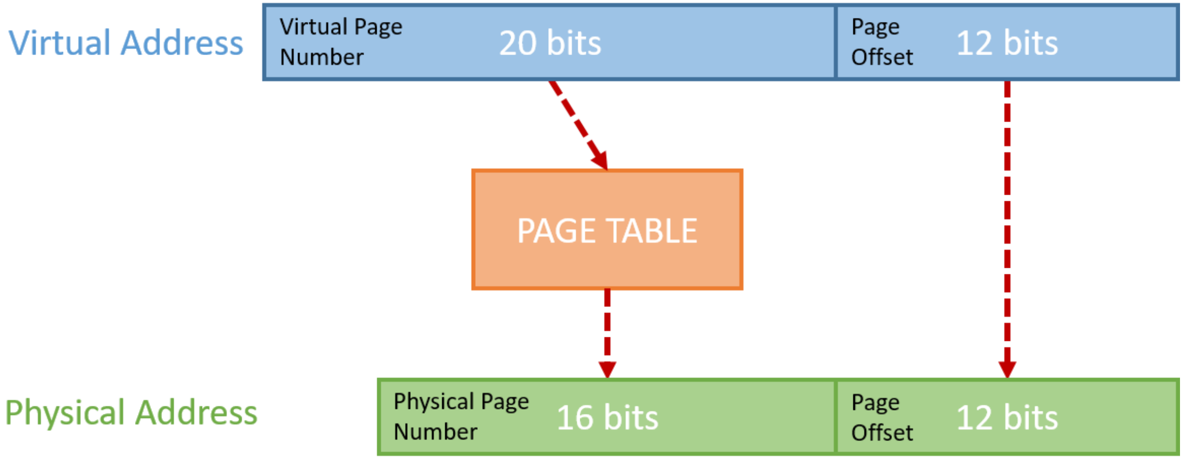
12-bit Page Offset [Index]



Address Translation

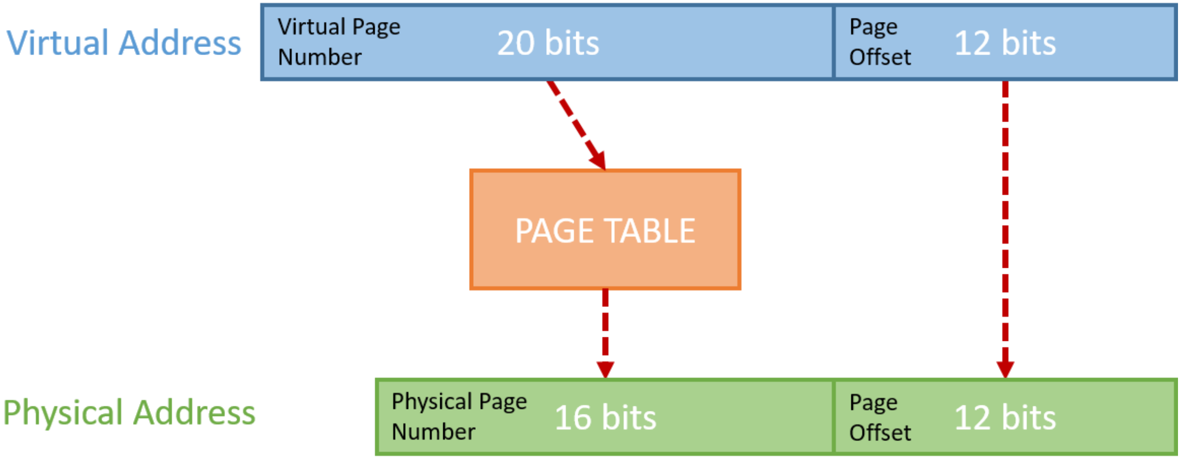
Computer Specs:
32-bit ISA, **256 MB of RAM**, 4 kB pages

Q: Why do we have more Virtual Page Number bits than Physical Page Number bits in this example?



Address Translation

Computer Specs:
32-bit ISA, **256 MB of RAM**, 4 kB pages

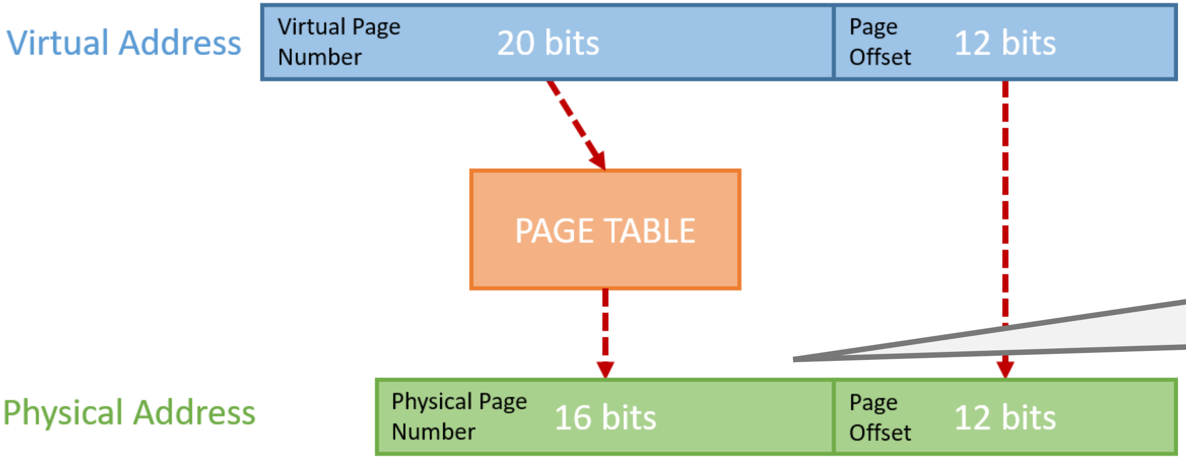


Q: Why do we have more Virtual Page Number bits than Physical Page Number bits in this example?

A: The virtual address space is larger than our physical RAM in this example (here assumed to be 256 MB).

Address Translation

Computer Specs:
32-bit ISA, **256 MB of RAM**, 4 kB pages



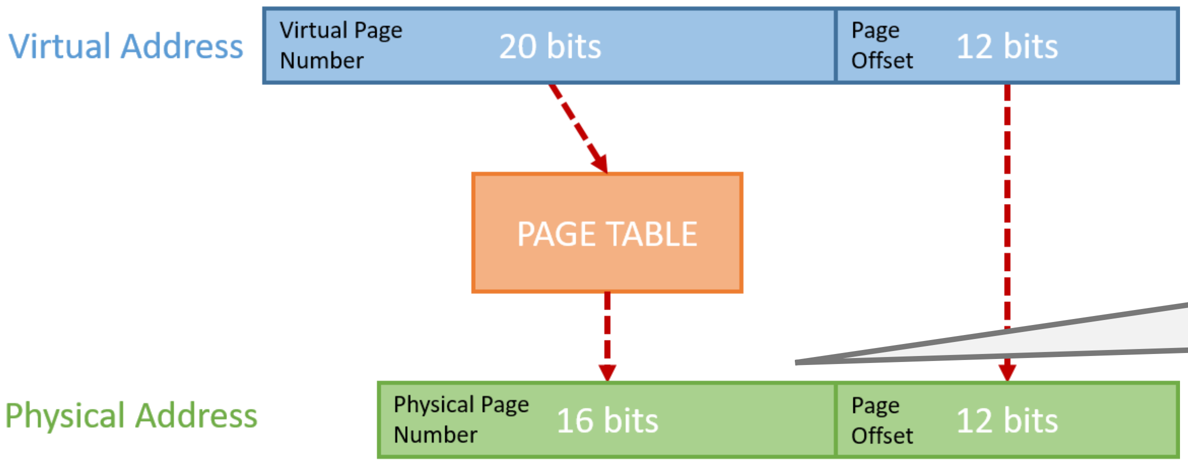
Q: Why do we have more Virtual Page Number bits than Physical Page Number bits in this example?

A: The virtual address space is larger than our physical RAM in this example (here assumed to be 256 MB).

Q: What would change if we now have **8 GB of RAM** installed (instead of 256MB)?

Address Translation

Computer Specs:
32-bit ISA, **256 MB of RAM**, 4 kB pages



Q: Why do we have more Virtual Page Number bits than Physical Page Number bits in this example?

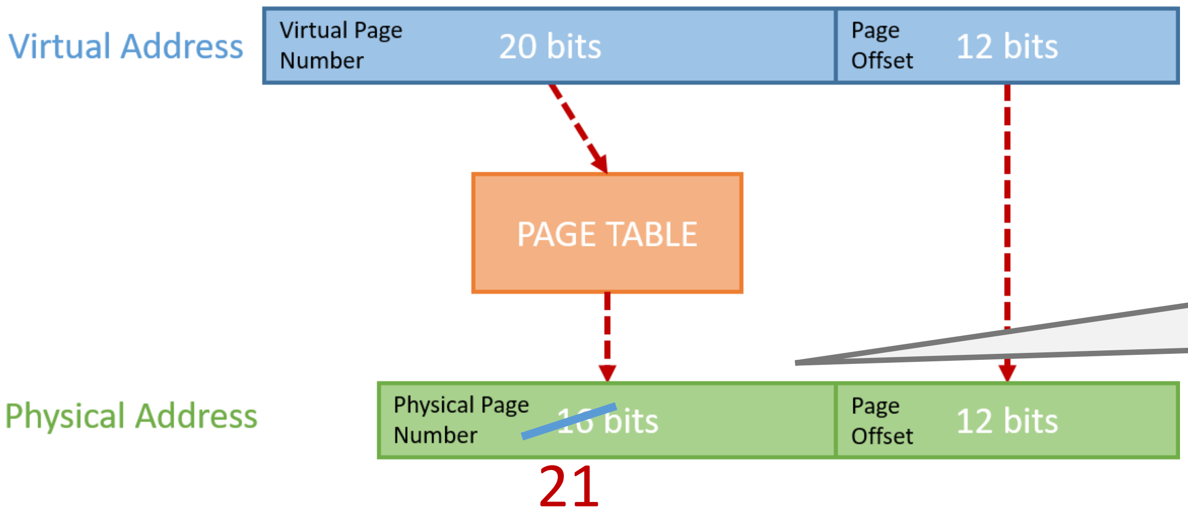
A: The virtual address space is larger than our physical RAM in this example (here assumed to be 256 MB).

Q: What would change if we now have **8 GB of RAM** installed (instead of 256MB)?

A: The physical address space is now larger than our virtual address space. We need 33 bits to address all Words in memory. Our Physical Page is now 21 bits.

Address Translation

Computer Specs:
32-bit ISA, **256 MB of RAM**, 4 kB pages



Q: Why do we have more Virtual Page Number bits than Physical Page Number bits in this example?

A: The virtual address space is larger than our physical RAM in this example (here assumed to be 256 MB).

Q: What would change if we now have **8 GB of RAM** installed (instead of 256MB)?

A: The physical address space is now larger than our virtual address space. We need 33 bits to address all Words in memory. Our Physical Page is now 21 bits.

Address Translation Example

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset

Virtual Address [32 bit]

32 bits

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

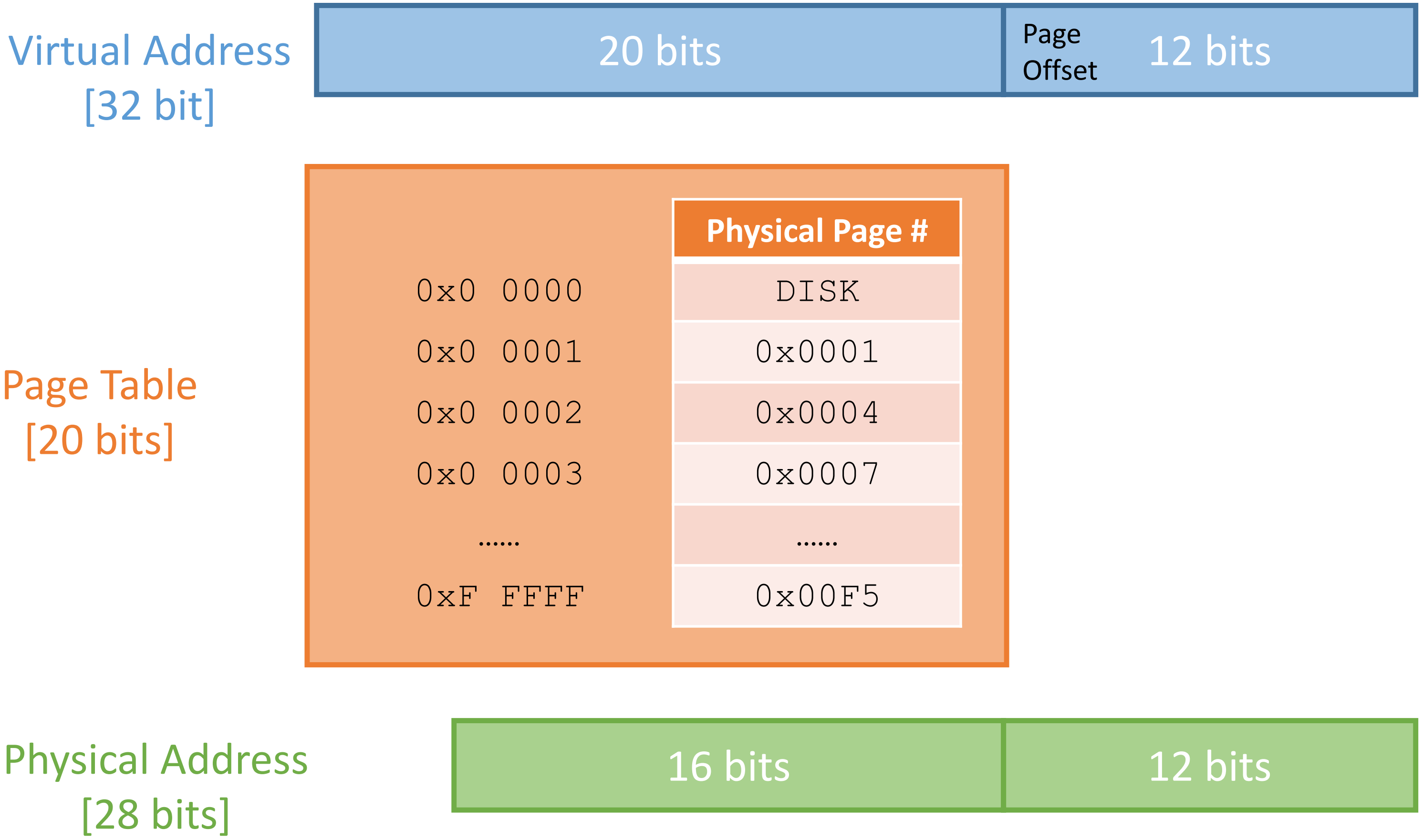
28 bits

Address Translation Example

<u>Computer Specs</u>:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

<u>In our terms</u>:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



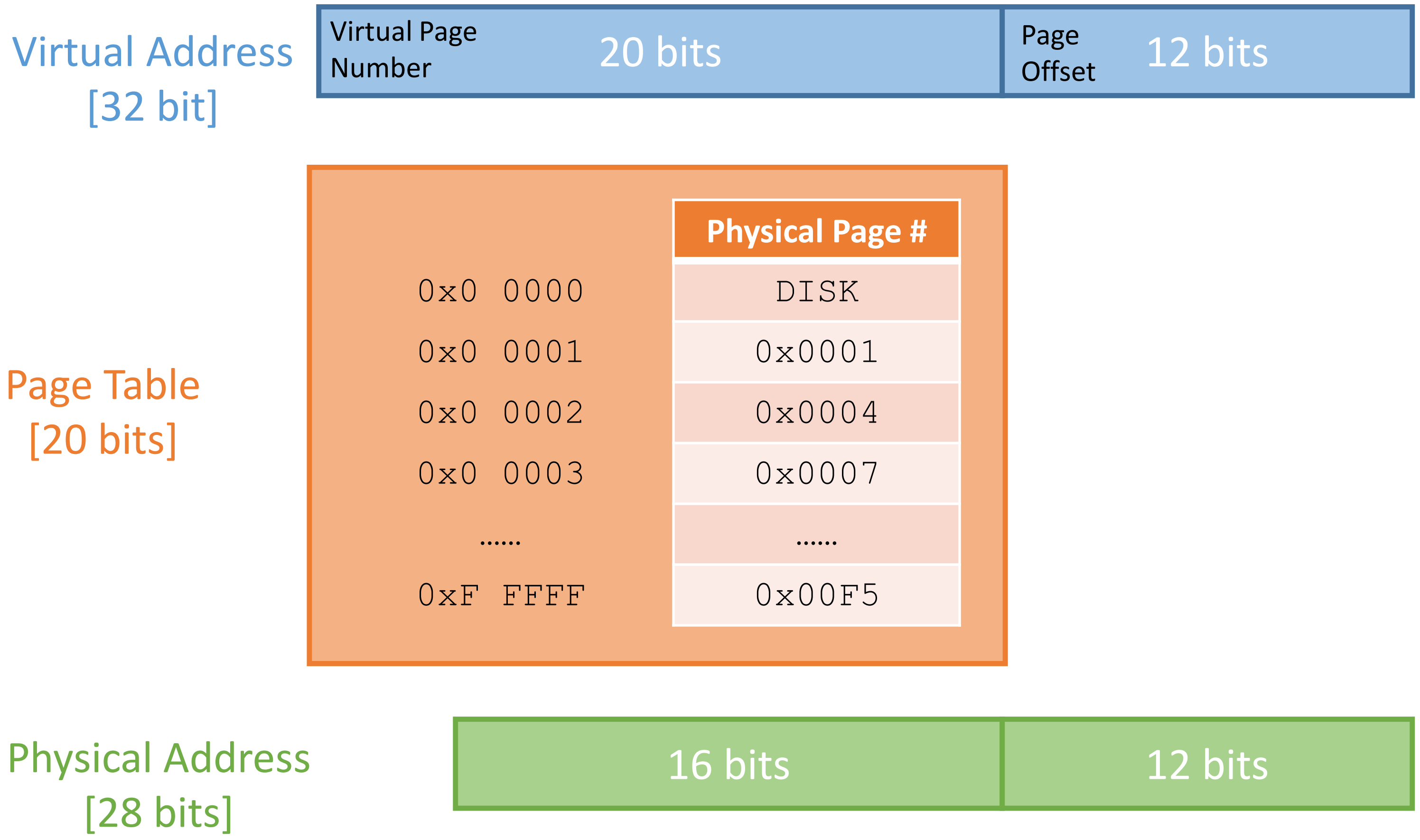
	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Address Translation Example

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number 20 bits	Page Offset 12 bits

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
......	......
0xF FFFF	0x00F5

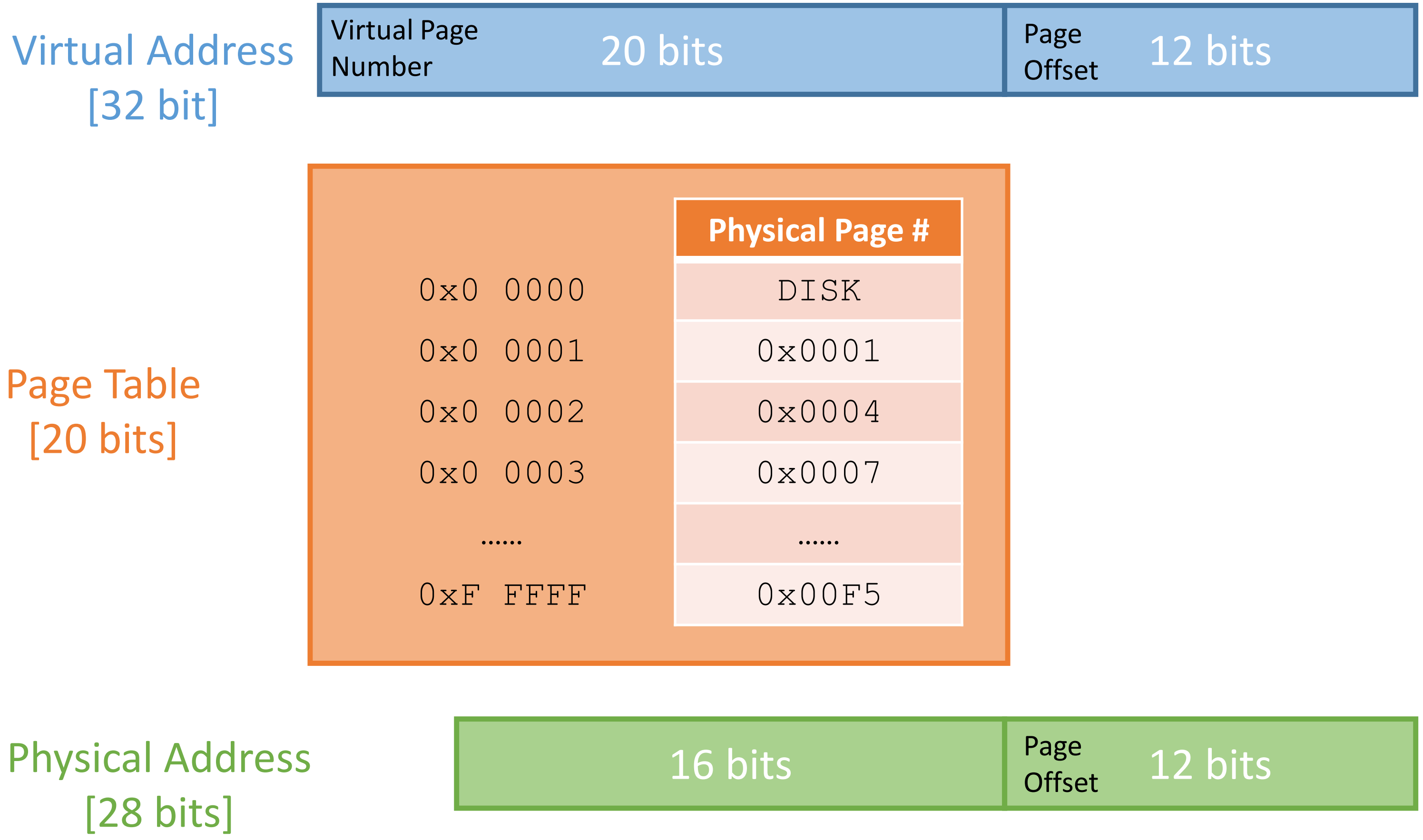
Physical Address [28 bits]

16 bits	12 bits

Address Translation Example

<u>Computer Specs</u>:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

<u>In our terms</u>:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Page Table [20 bits]

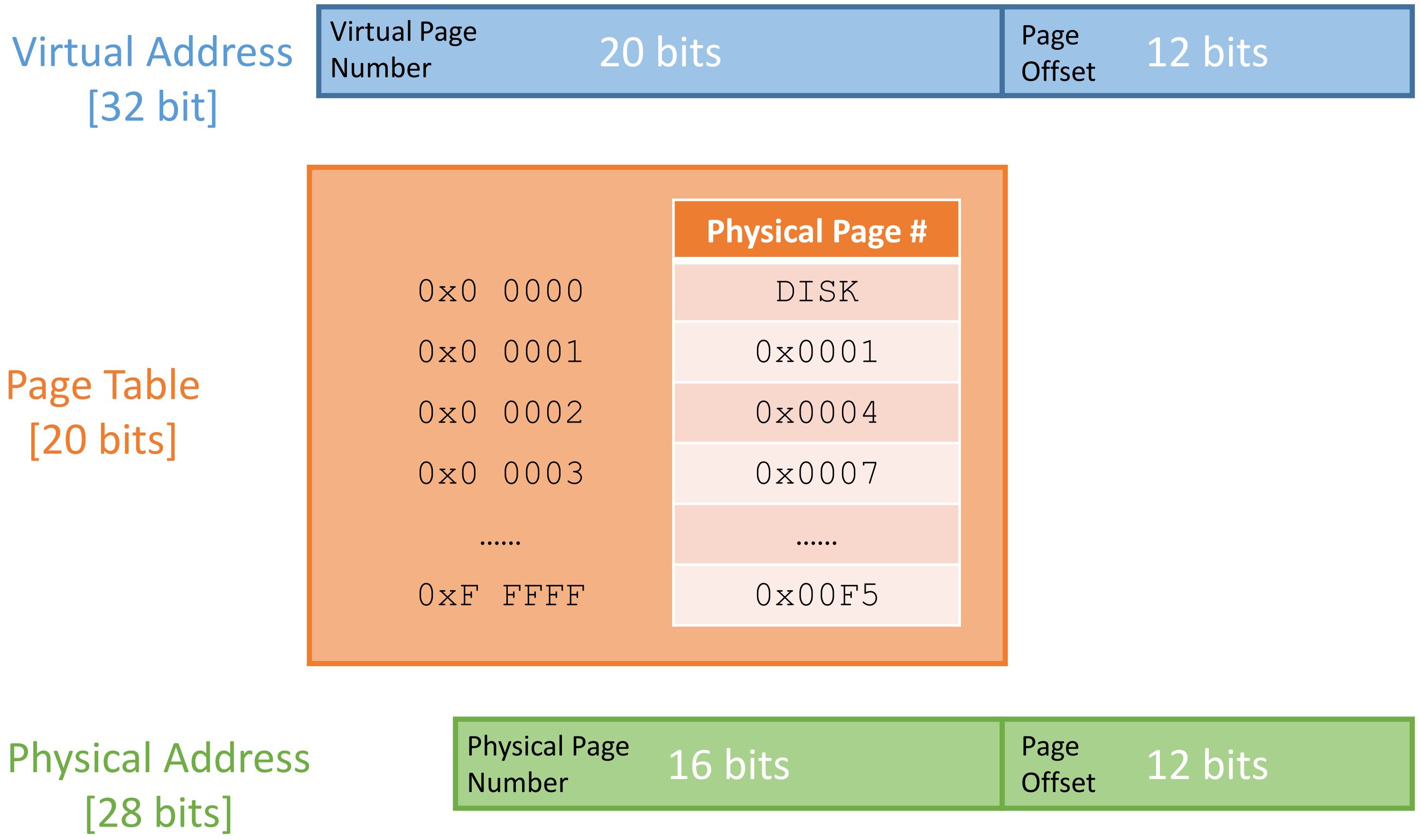
	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

Address Translation Example

<u>Computer Specs</u>:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

<u>In our terms</u>:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number	20 bits	Page Offset	12 bits

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

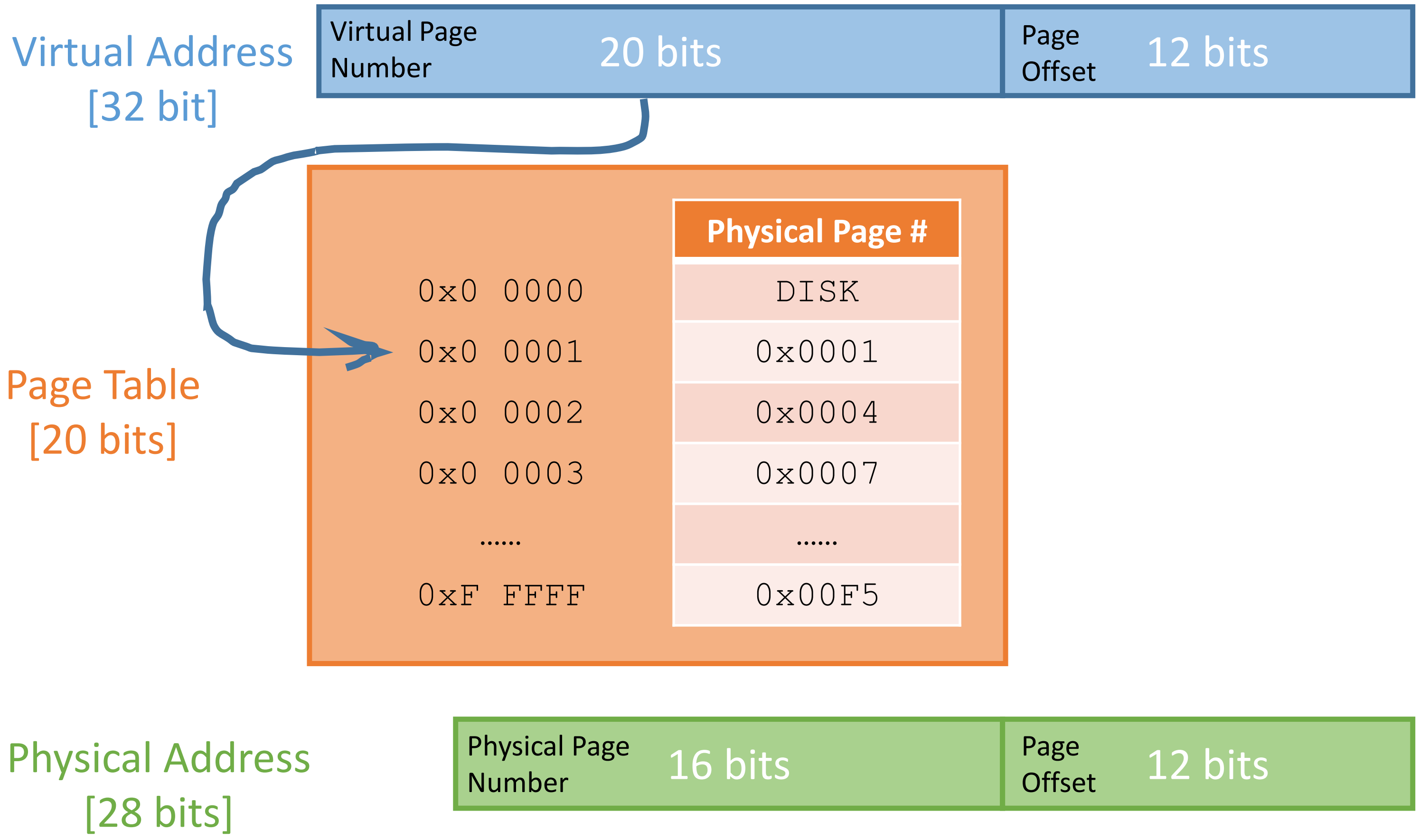
Physical Page Number	16 bits	Page Offset	12 bits

Address Translation Example

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number	Page Offset
20 bits	12 bits

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

Physical Page Number	Page Offset
16 bits	12 bits

Address Translation Example

Computer Specs:

ISA:	32-bit
RAM:	256 MB
Page Size:	4 kB

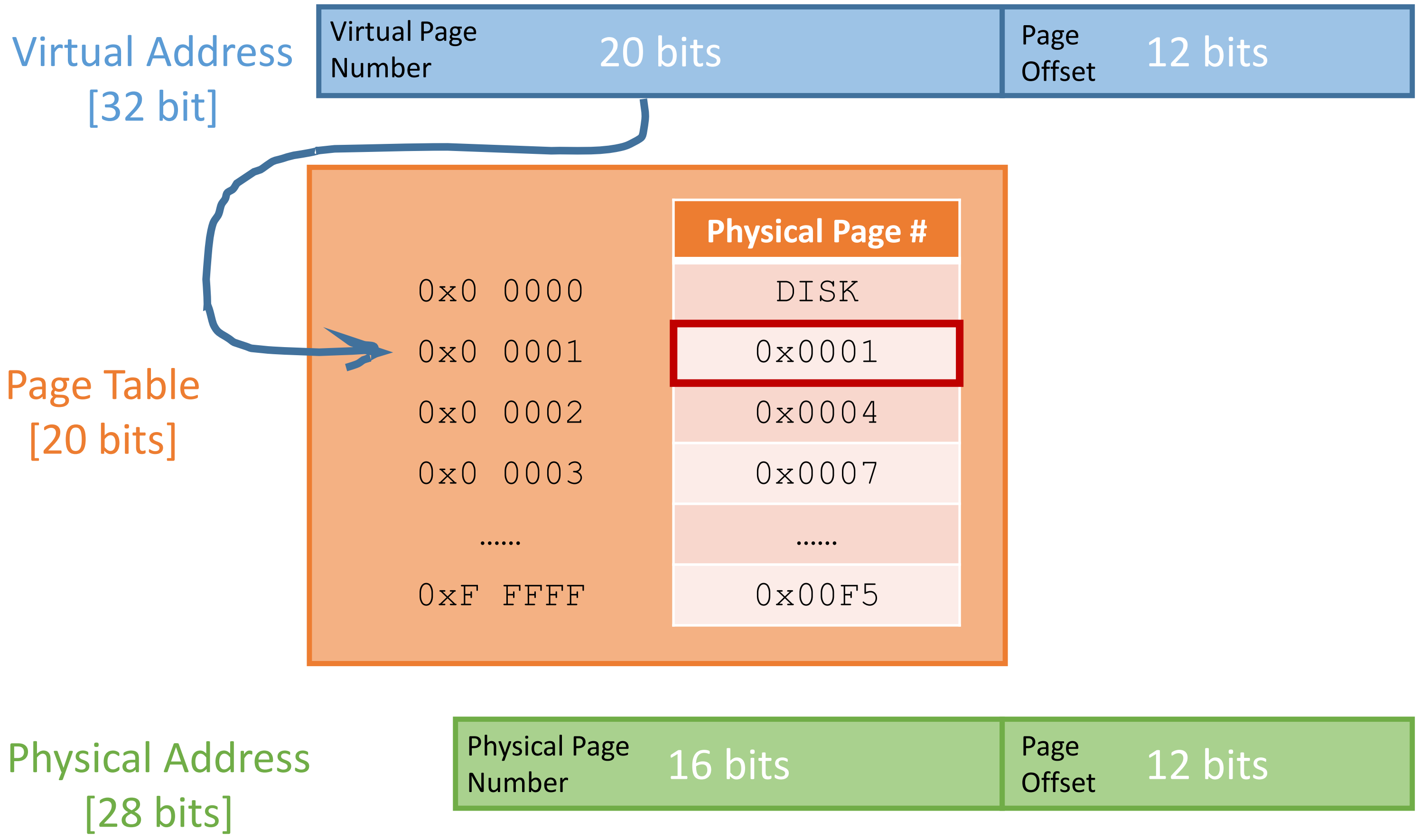
⇨

In our terms:

32-bit Virtual Address

28-bit Physical Address

12-bit Page Offset

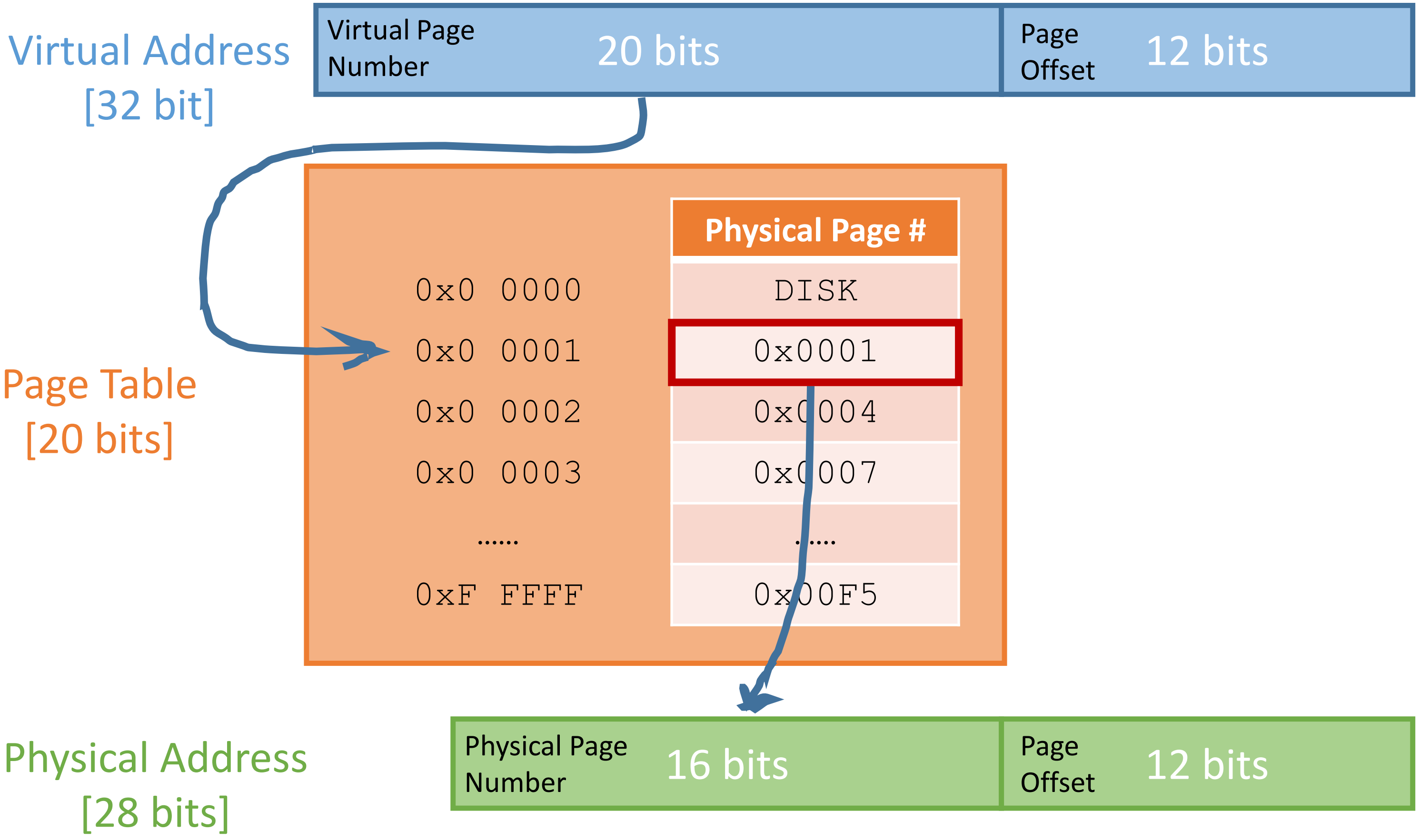


	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Address Translation Example

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number 20 bits	Page Offset 12 bits

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

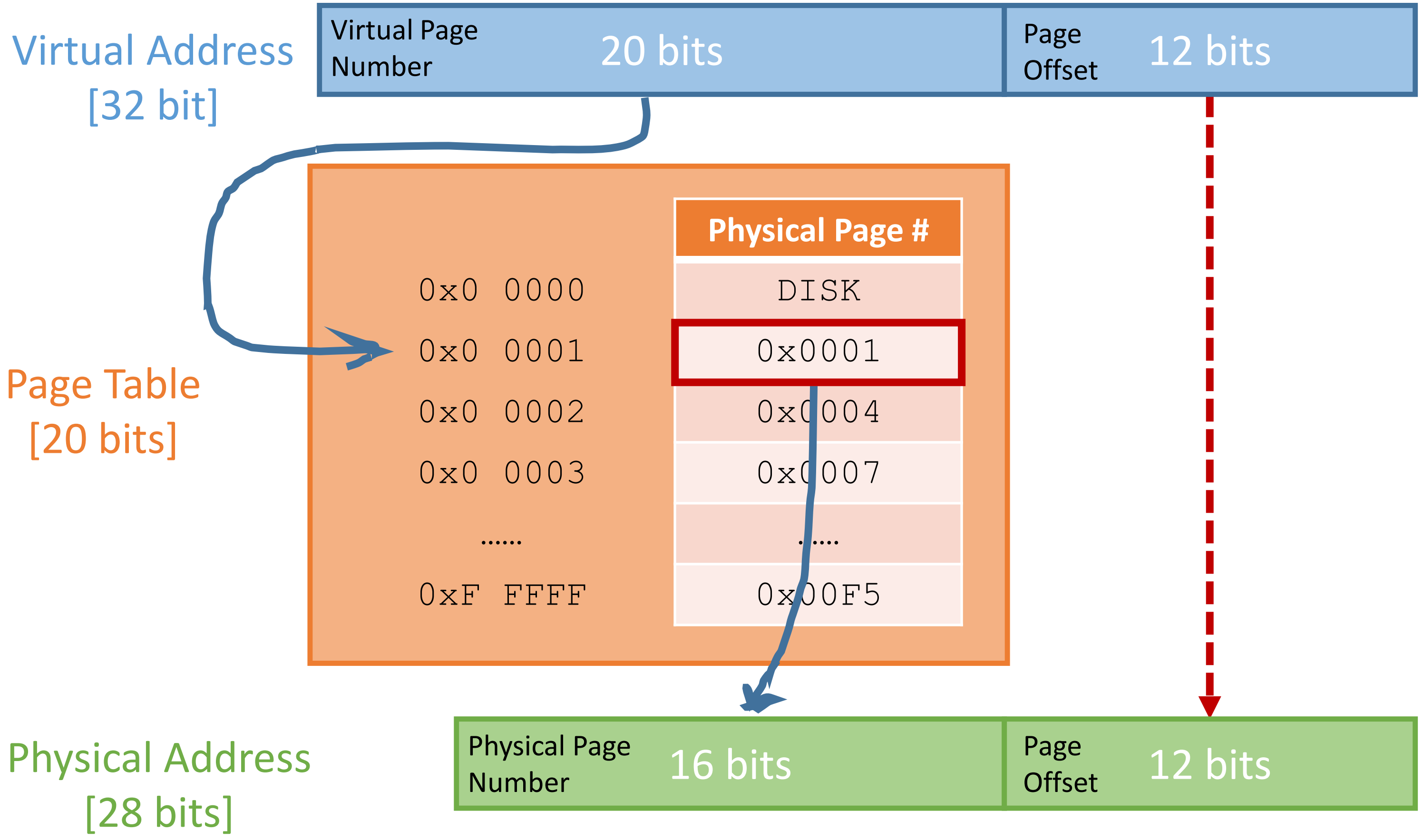
Physical Page Number 16 bits	Page Offset 12 bits

Address Translation Example

<u>Computer Specs</u>:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

<u>In our terms</u>:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number	Page Offset
20 bits	12 bits

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

Physical Page Number	Page Offset
16 bits	12 bits

Address Translation Example

#1

<u>Computer Specs</u>:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

<u>In our terms</u>:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset

Virtual Address
[32 bit]

Virtual Page Number	Page Offset
31 … 12	11 … 0

Page Table
[20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
......	......
0xF FFFF	0x00F5

Physical Address
[28 bits]

Physical Page Number	Page Offset
27 … 12	11 … 0

<u>EXAMPLE</u>:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

Address Translation Example

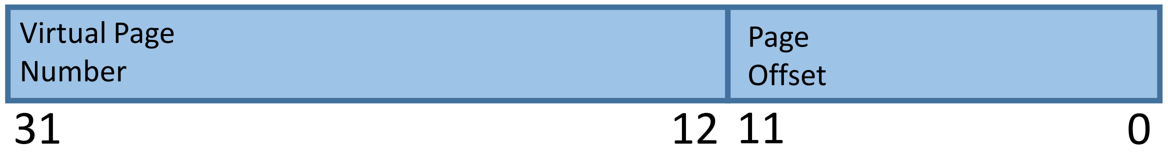
#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset

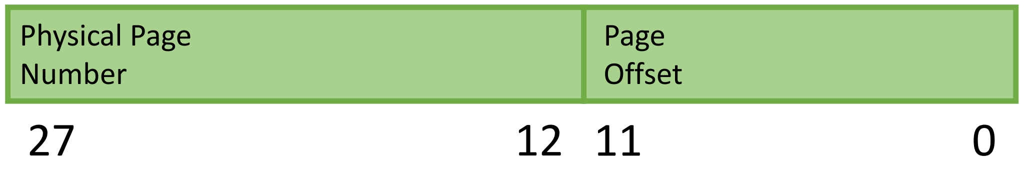
Virtual Address
[32 bit]



Page Table
[20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address
[28 bits]



EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset

Virtual Address [32 bit]

Virtual Page Number	Page Offset
31 … 12	11 … 0

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

Physical Page Number	Page Offset
27 … 12	11 … 0

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

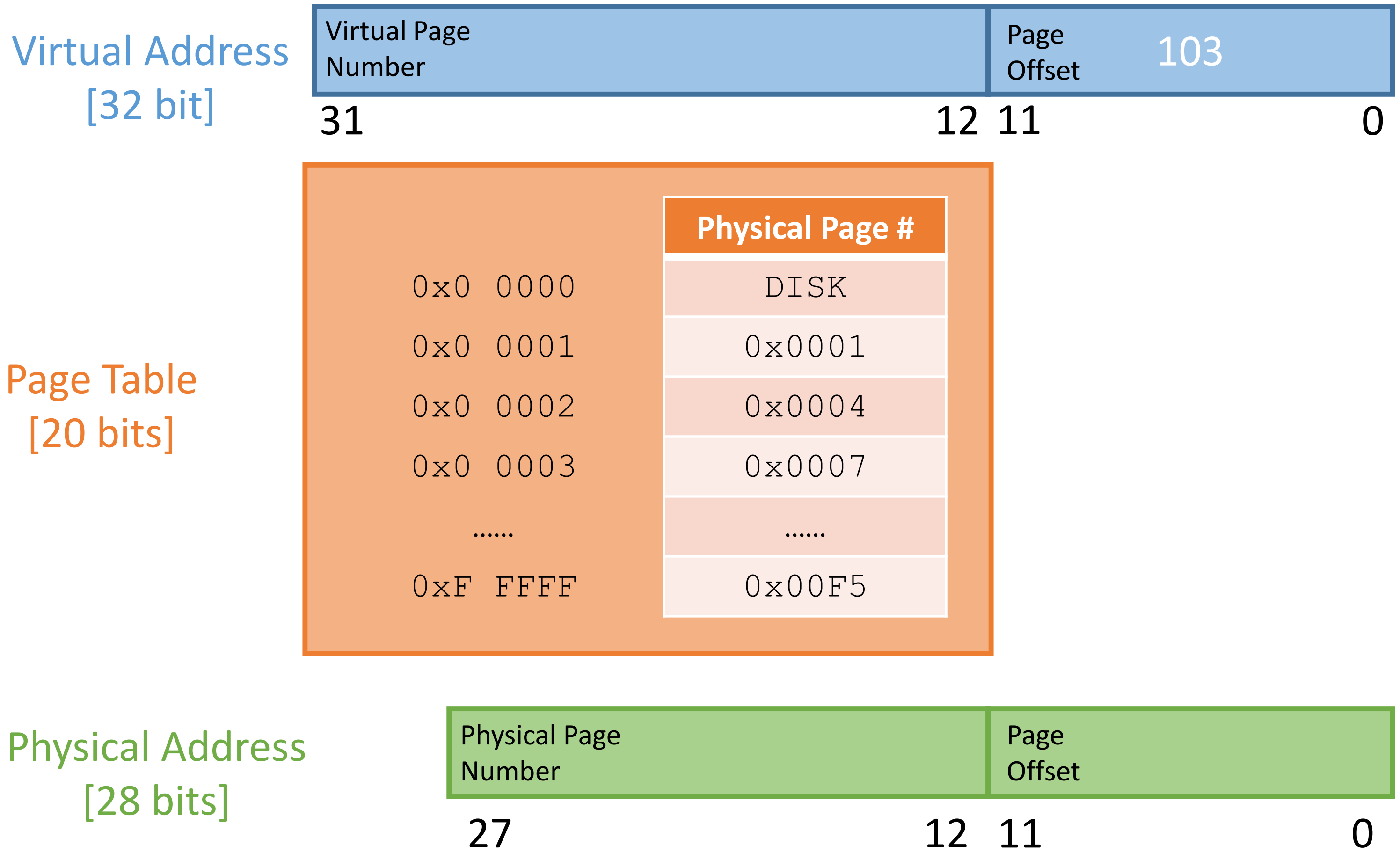
VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number	Page Offset 103
31 … 12	11 … 0

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

Physical Page Number	Page Offset
27 … 12	11 … 0

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

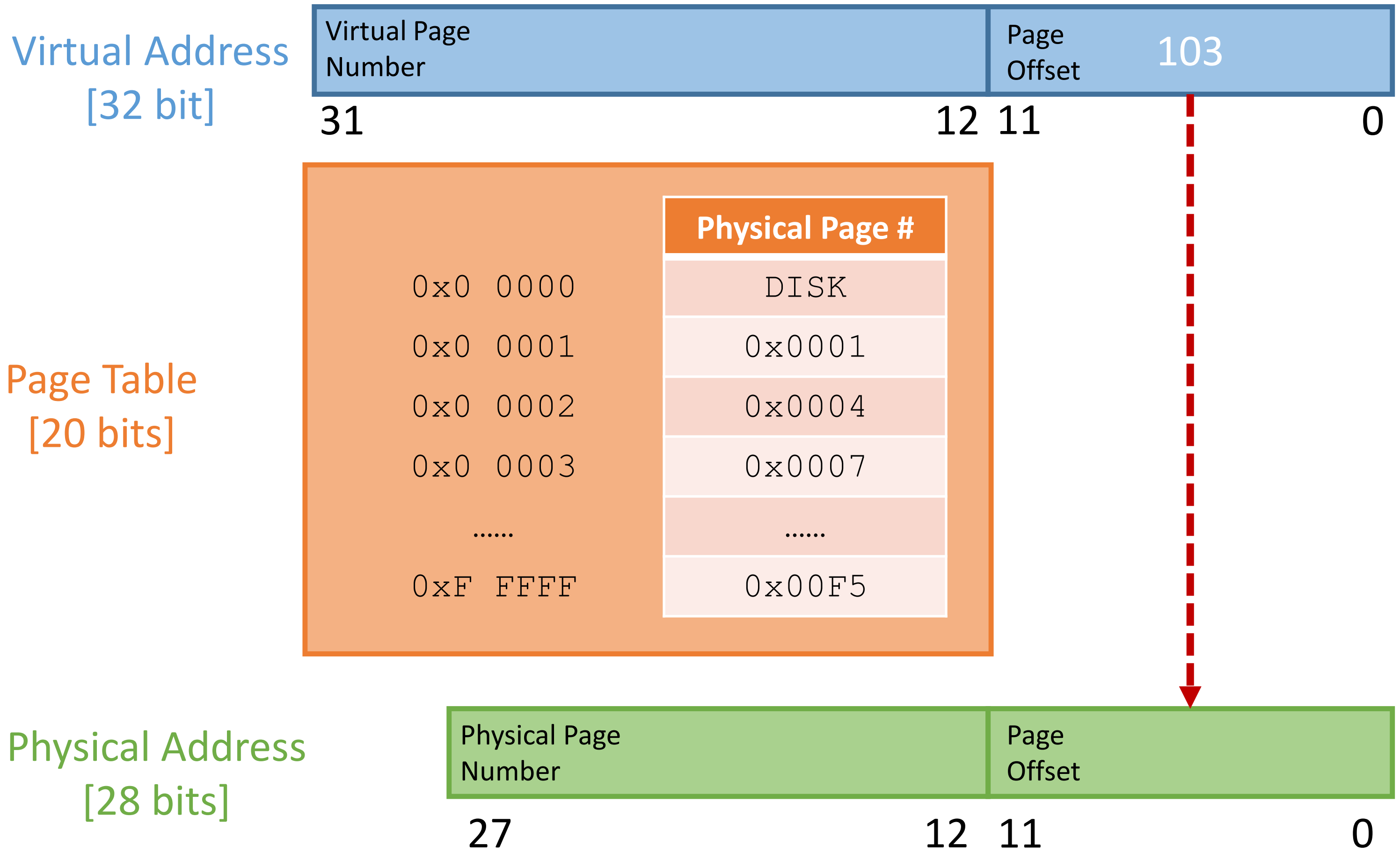
VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

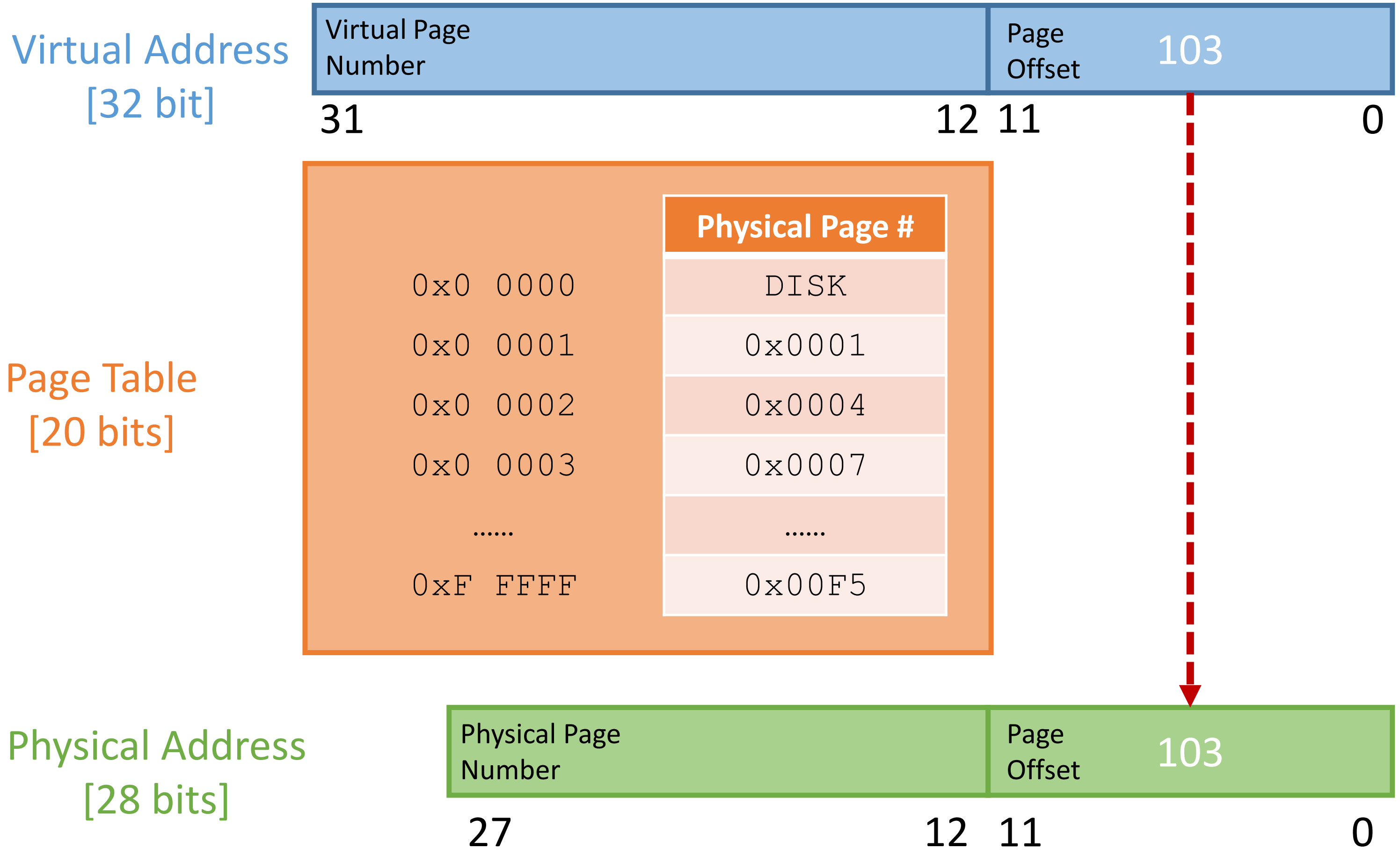
VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
......	......
0xF FFFF	0x00F5

Physical Address [28 bits]

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

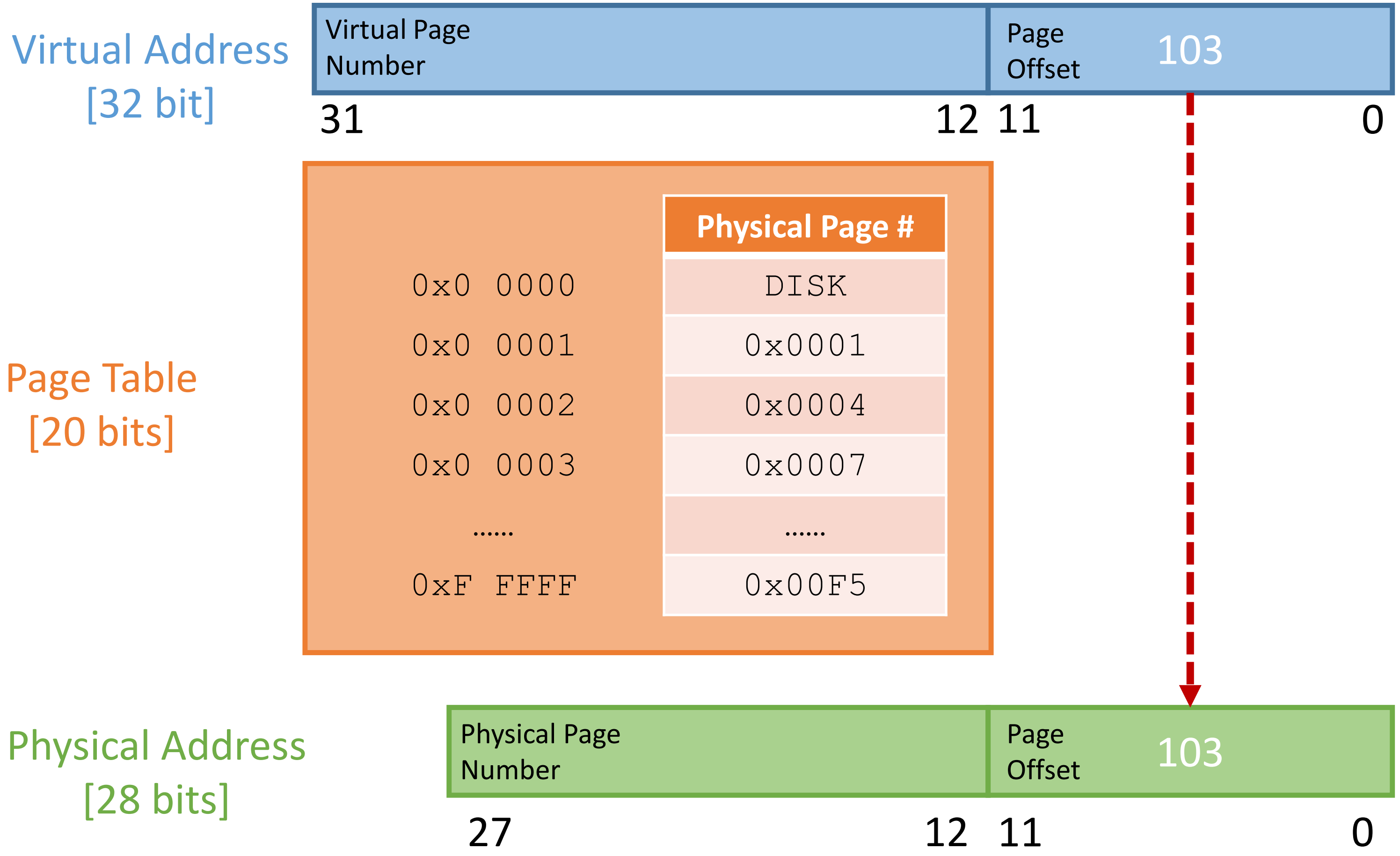
VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

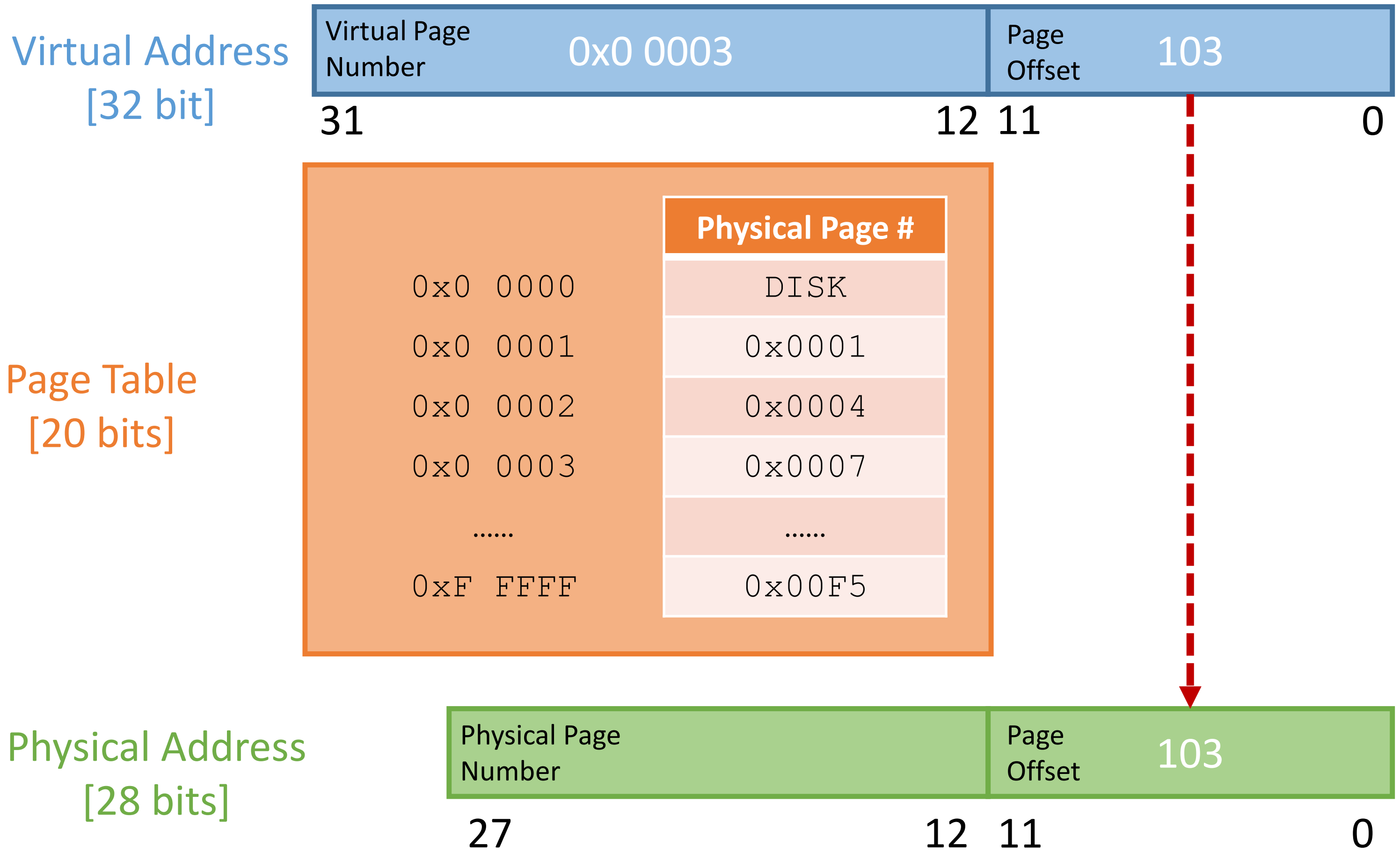
VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number 0x0 0003 | Page Offset 103

31 12 11 0

Page Table [20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address [28 bits]

Physical Page Number | Page Offset 103

27 12 11 0

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

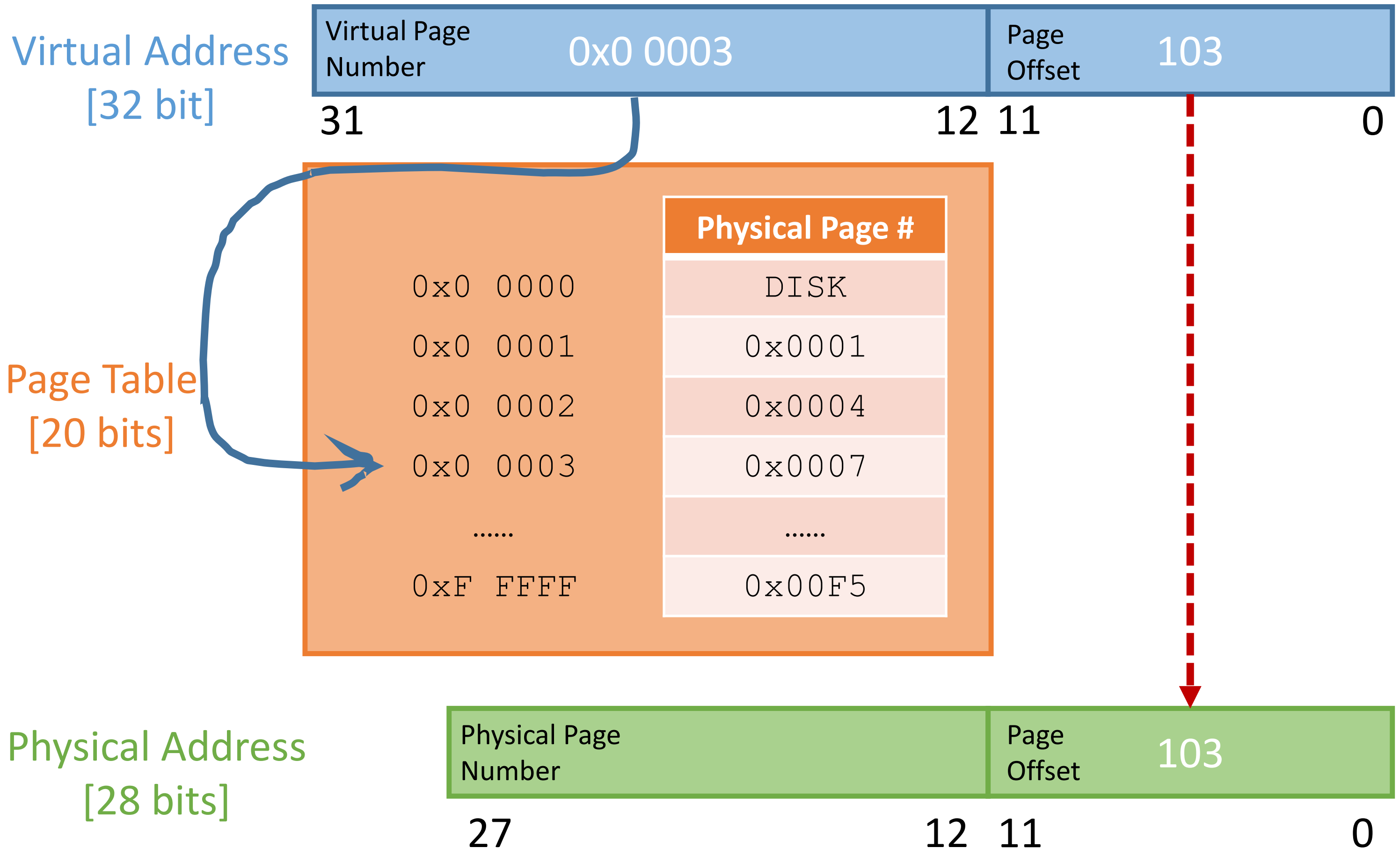
VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

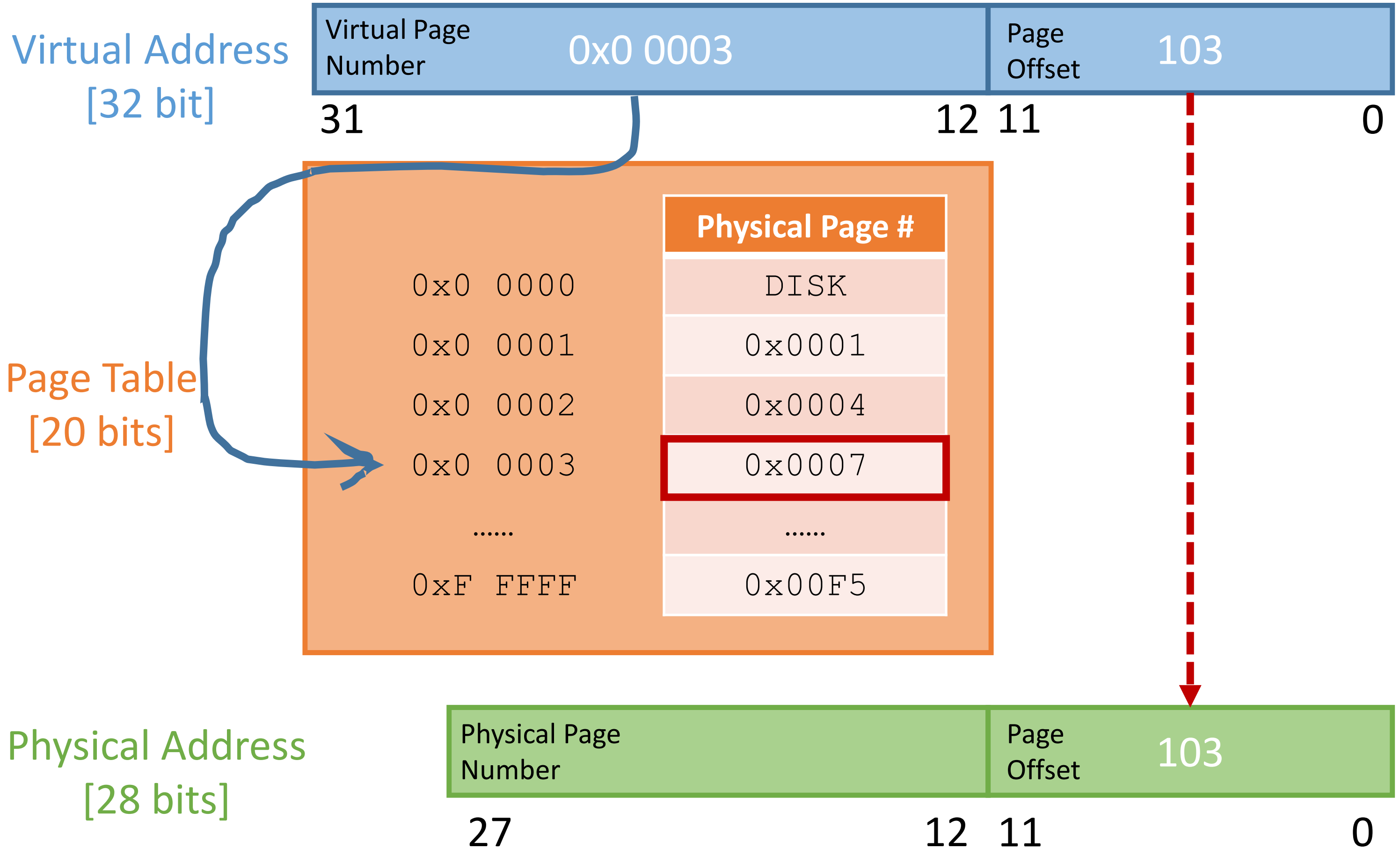
VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

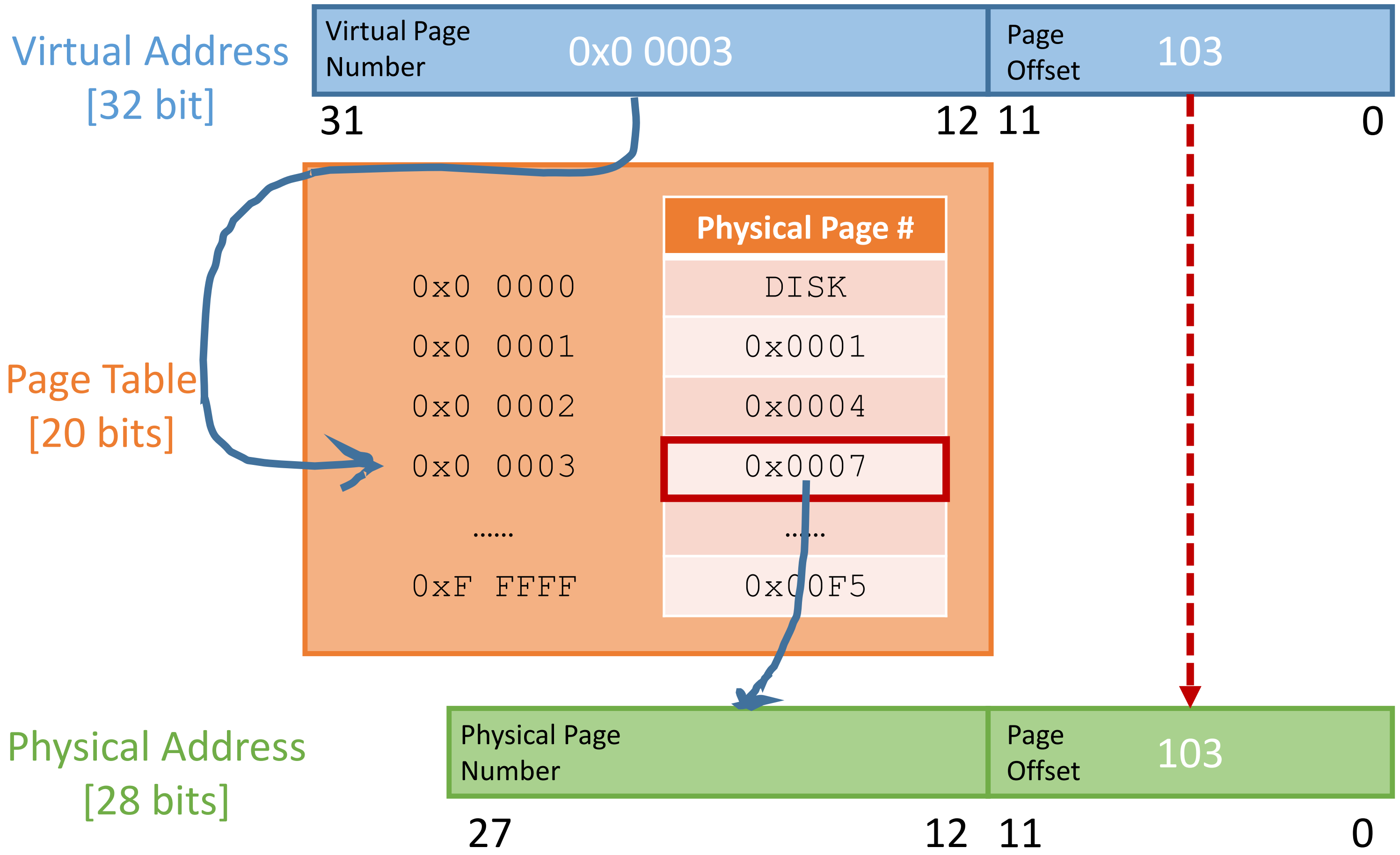
VA: 0x00003 103

Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

VA: 0x00003 103

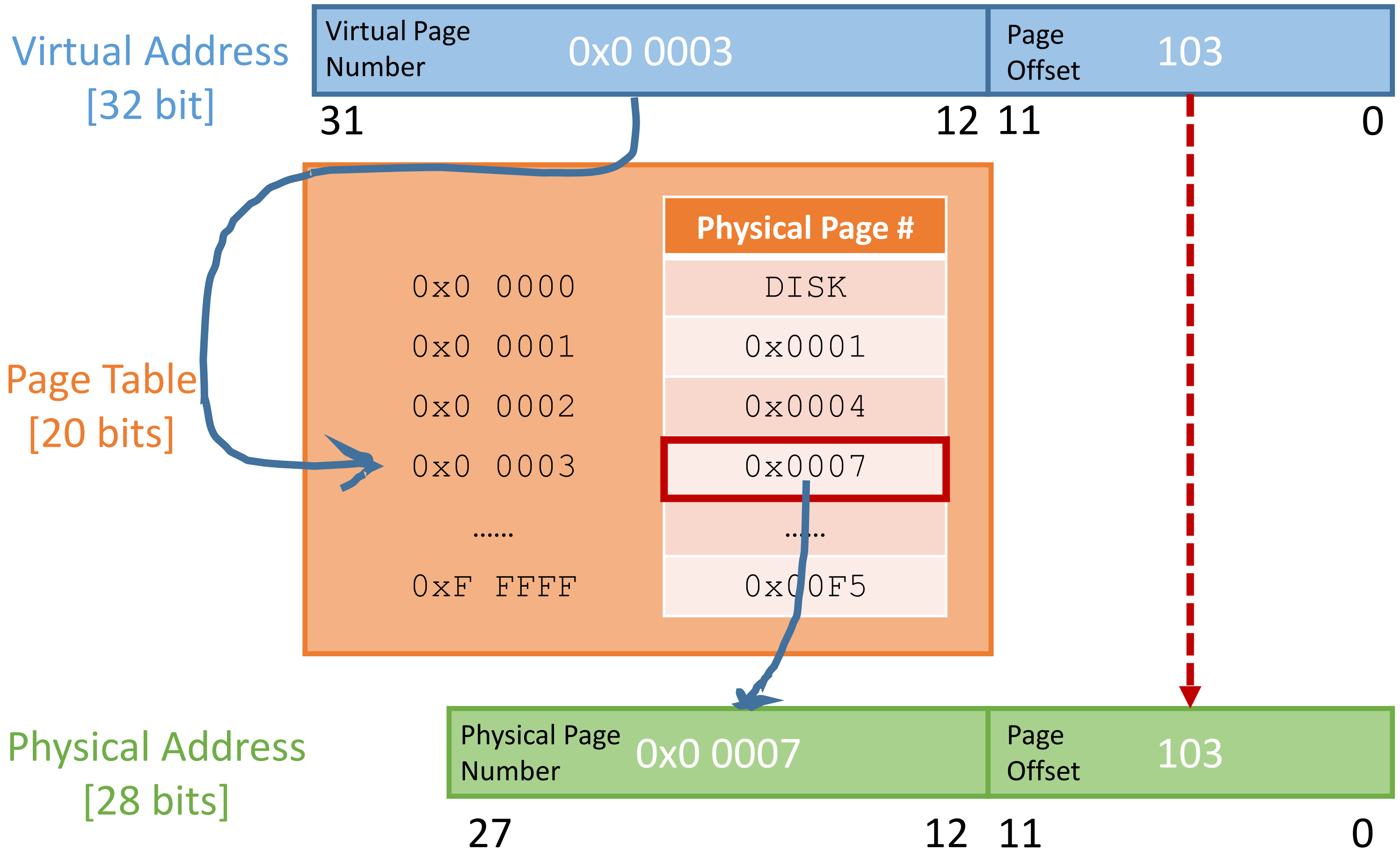
Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
......	......
0xF FFFF	0x00F5

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

VA: 0x00003 103

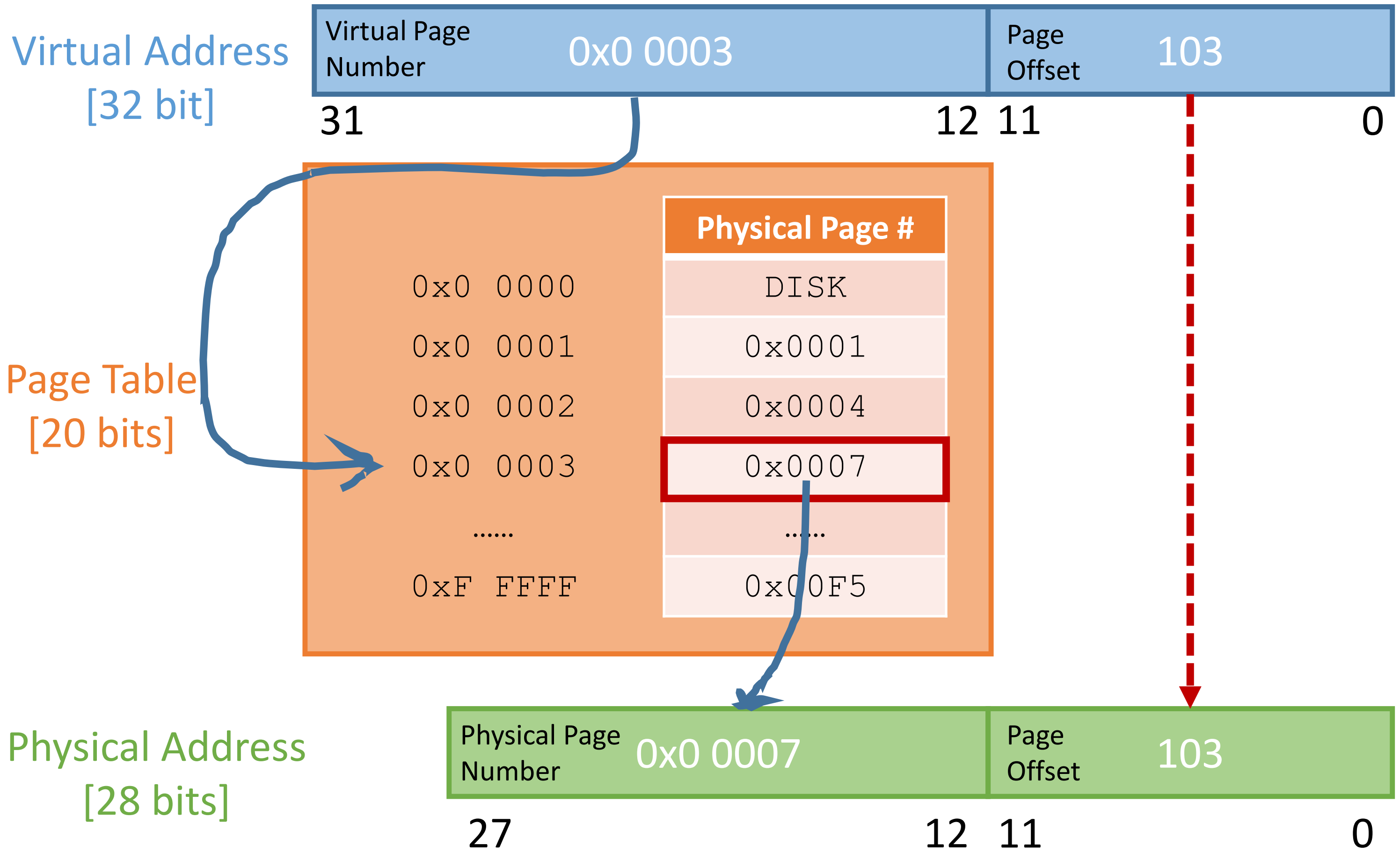
Address Translation Example

#1

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

⇨

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset



	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00003103**

VA: 0x00003 103

⇩

PA: 0x00007 103

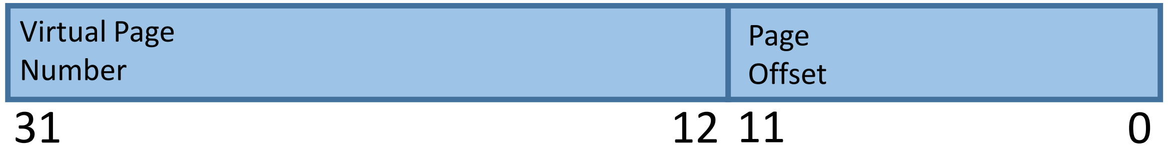
Address Translation Example

#2

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset

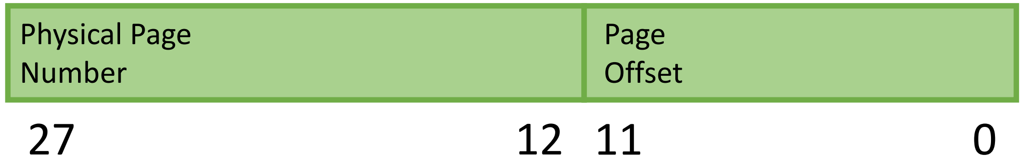
Virtual Address
[32 bit]



Page Table
[20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
……	……
0xF FFFF	0x00F5

Physical Address
[28 bits]



EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00000504**

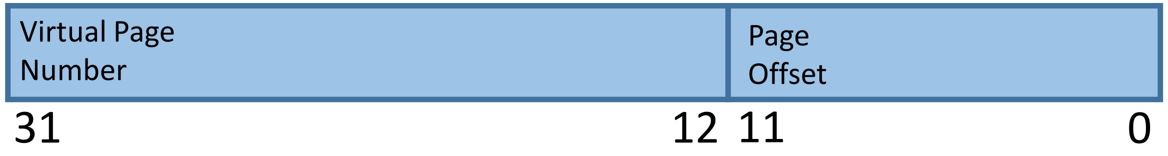
Address Translation Example

#2

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 4 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
12-bit Page Offset

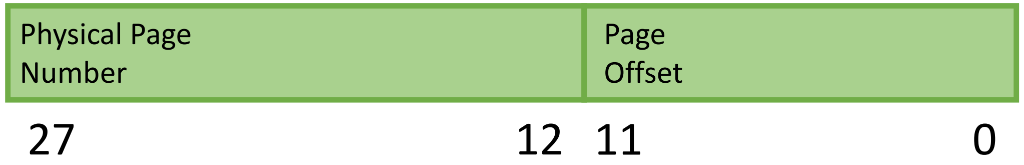
Virtual Address
[32 bit]



Page Table
[20 bits]

	Physical Page #
0x0 0000	DISK
0x0 0001	0x0001
0x0 0002	0x0004
0x0 0003	0x0007
......	......
0xF FFFF	0x00F5

Physical Address
[28 bits]



EXAMPLE:
Using the page table to the left, find the Physical Address associated with Virtual Address **0x00000504**

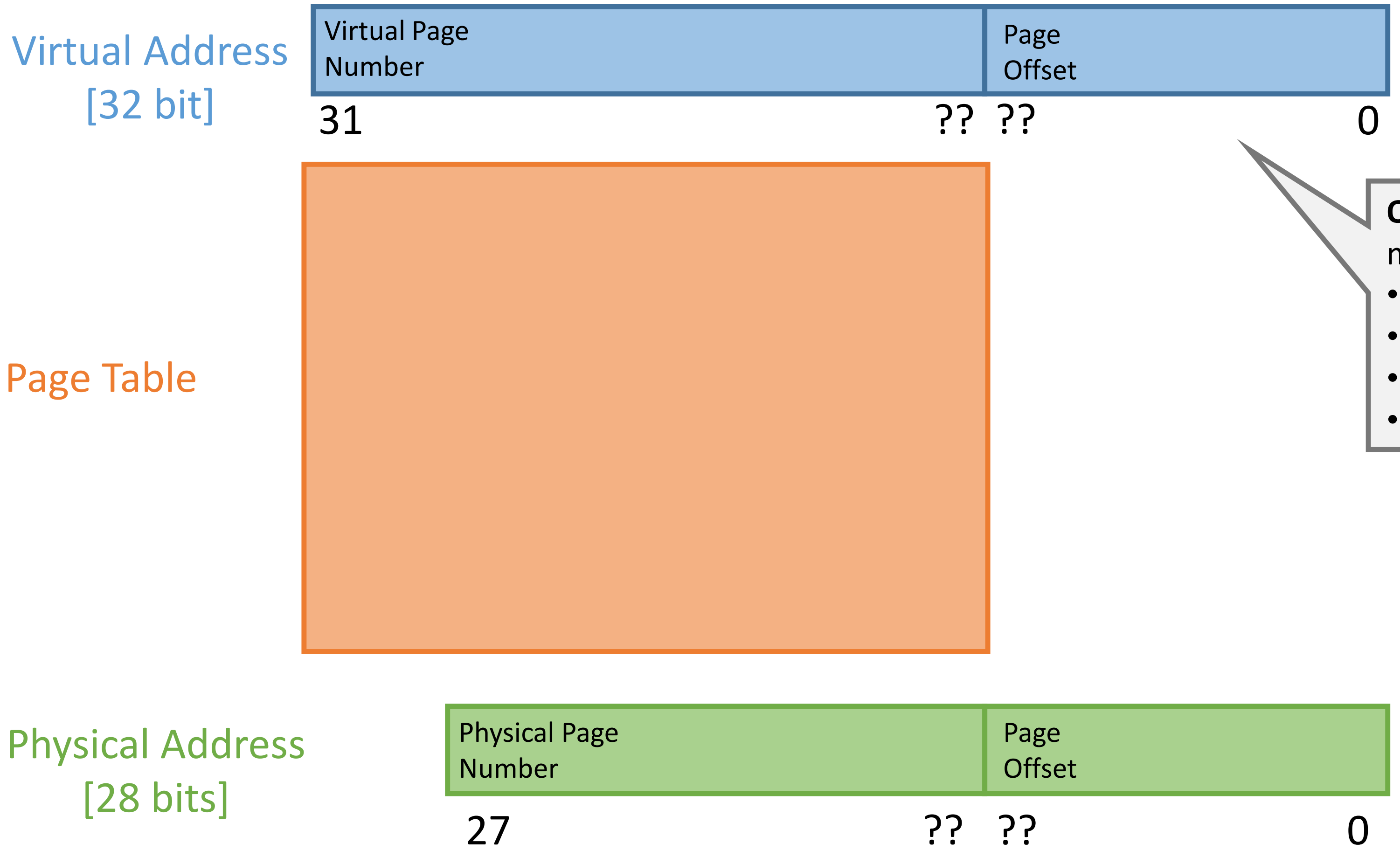
Virtual Address 0x00000 points to the disk, we don’t know from the given information.

Address Translation Example

#3

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 64 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
??-bit Page Offset



Q: What if we use **64 kB pages** instead. How many bits do we need for the **page offset**?

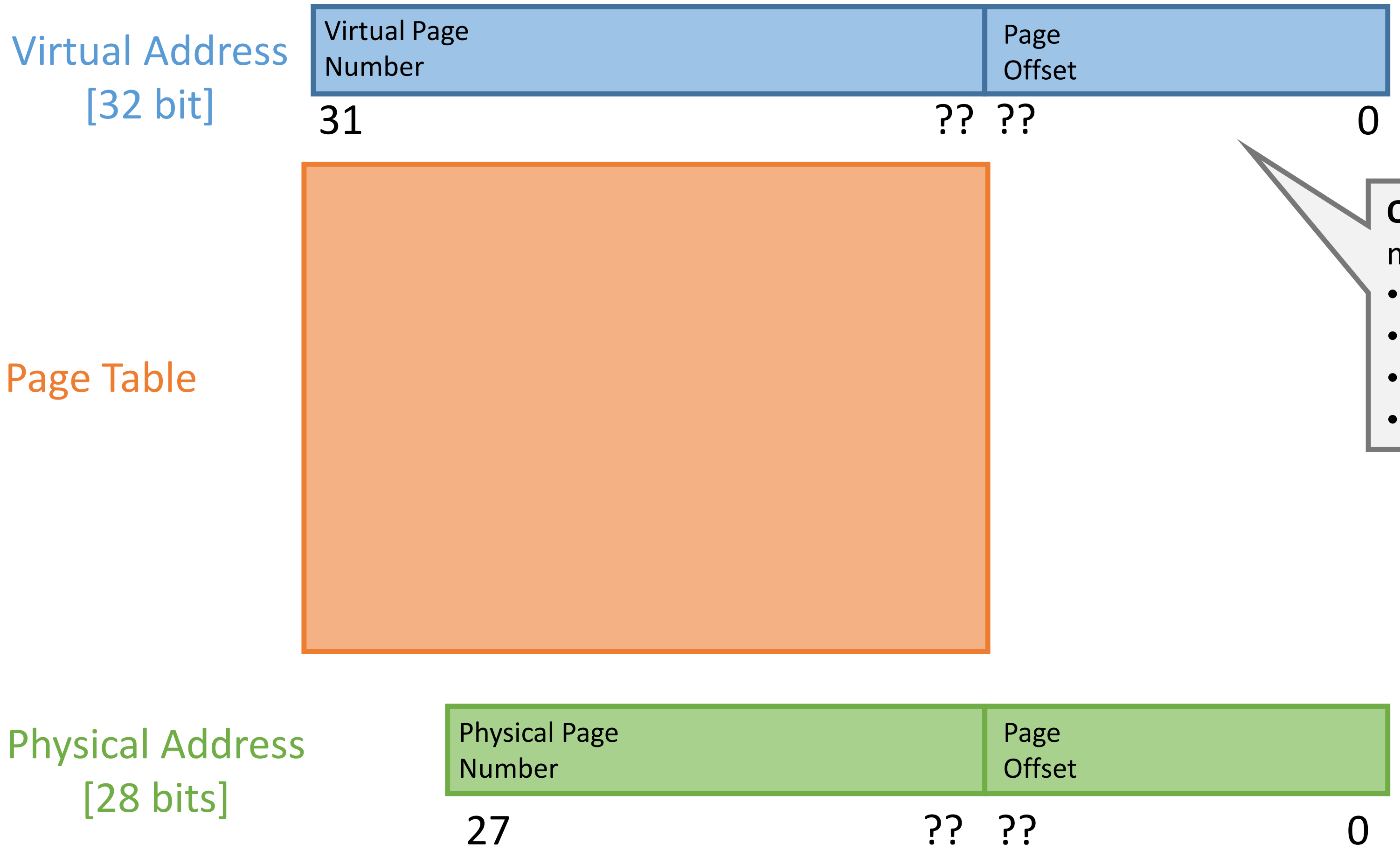
- 12
- 13
- 16
- 18

Address Translation Example

#3

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 64 kB

In our terms:
32-bit Virtual Address
28-bit Physical Address
??-bit Page Offset



Q: What if we use **64 kB pages** instead. How many bits do we need for the **page offset**?

- 12
- 13
- 16
- 18

A: 16

We need 16 bits to index every Word in a Page. This means my *Virtual Page Number* is now *16 bits* (32-16) and my *Physical Page Number* is *12 bits* (28-16).

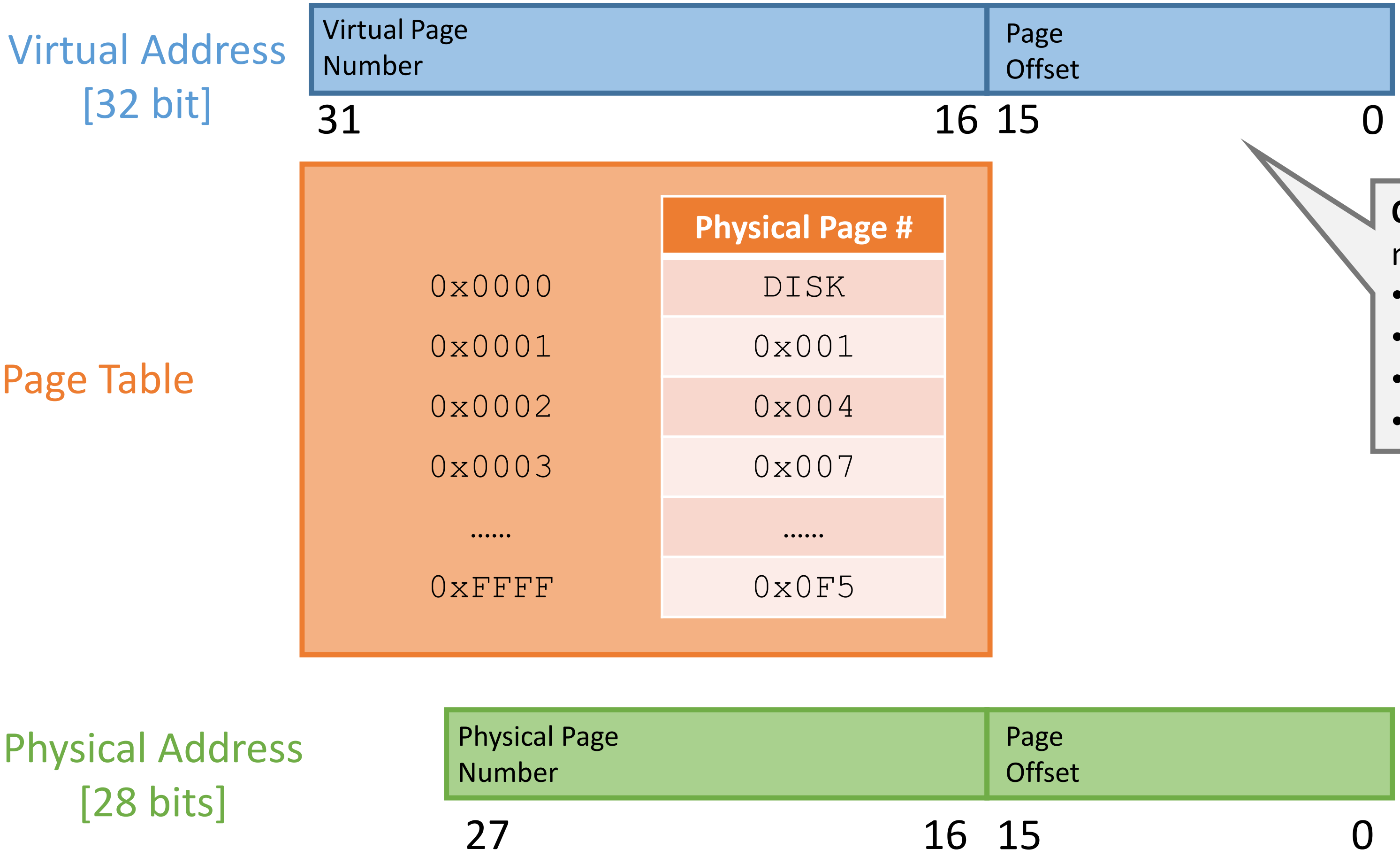
Address Translation Example

#3

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 64 kB

⇨

In our terms:
32-bit Virtual Address
28-bit Physical Address
16-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number	Page Offset
31 … 16	15 … 0

Page Table

	Physical Page #
0x0000	DISK
0x0001	0x001
0x0002	0x004
0x0003	0x007
……	……
0xFFFF	0x0F5

Physical Address [28 bits]

Physical Page Number	Page Offset
27 … 16	15 … 0

Q: What if we use **64 kB pages** instead. How many bits do we need for the **page offset**?

- 12
- 13
- 16
- 18

A: 16

We need 16 bits to index every Word in a Page. This means my *Virtual Page Number* is now *16 bits* (32-16) and my *Physical Page Number* is *12 bits* (28-16).

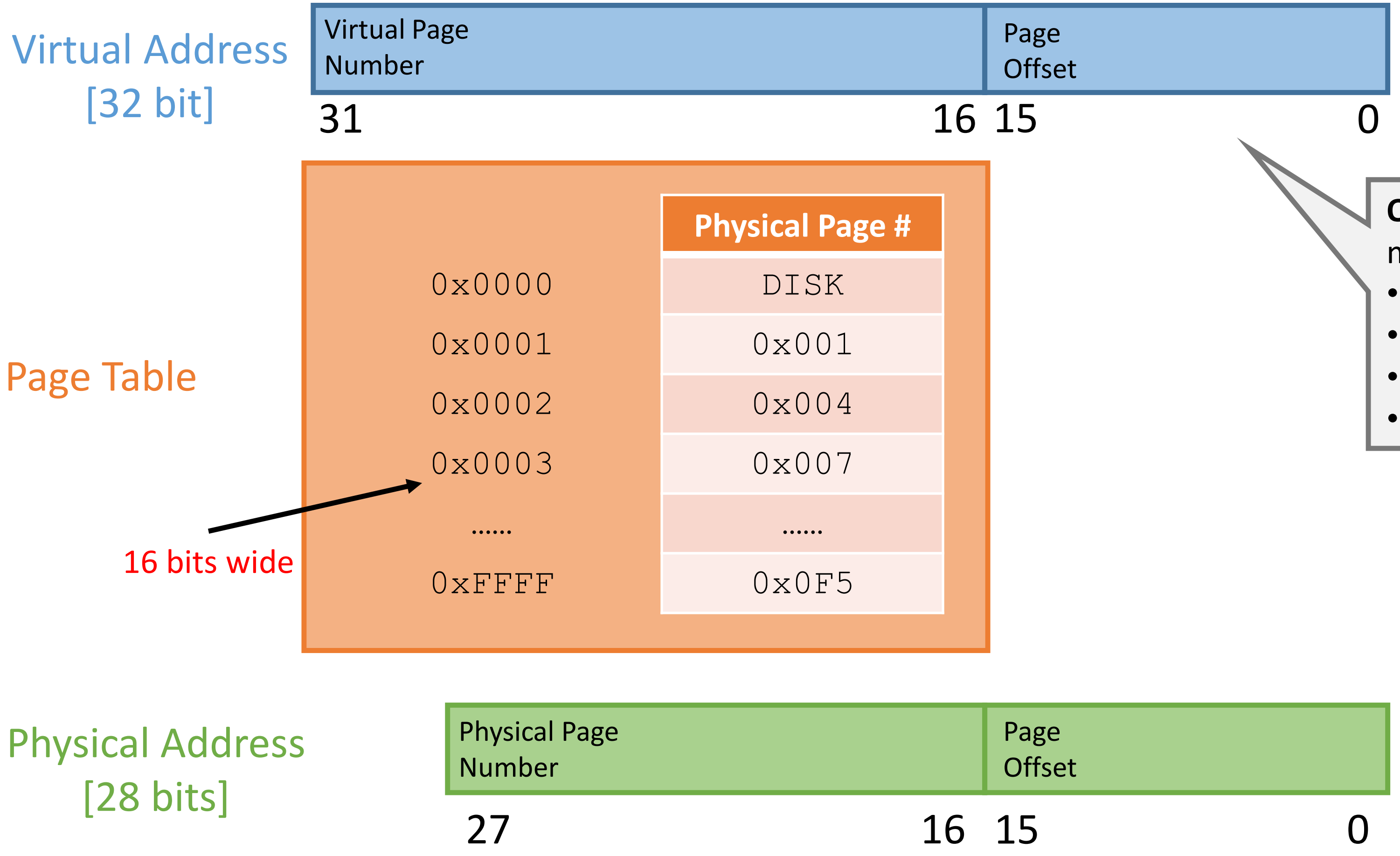
Address Translation Example

#3

Computer Specs:
ISA: 32-bit
RAM: 256 MB
Page Size: 64 kB

⇨

In our terms:
32-bit Virtual Address
28-bit Physical Address
16-bit Page Offset



Virtual Address [32 bit]

Virtual Page Number	Page Offset
31 … 16	15 … 0

Page Table

16 bits wide

	Physical Page #
0x0000	DISK
0x0001	0x001
0x0002	0x004
0x0003	0x007
……	……
0xFFFF	0x0F5

Physical Address [28 bits]

Physical Page Number	Page Offset
27 … 16	15 … 0

Q: What if we use **64 kB pages** instead. How many bits do we need for the **page offset**?

- 12
- 13
- 16
- 18

A: 16

We need 16 bits to index every Word in a Page. This means my *Virtual Page Number* is now *16 bits* (32-16) and my *Physical Page Number* is *12 bits* (28-16).

Address Translation Example

#3

Computer Specs:

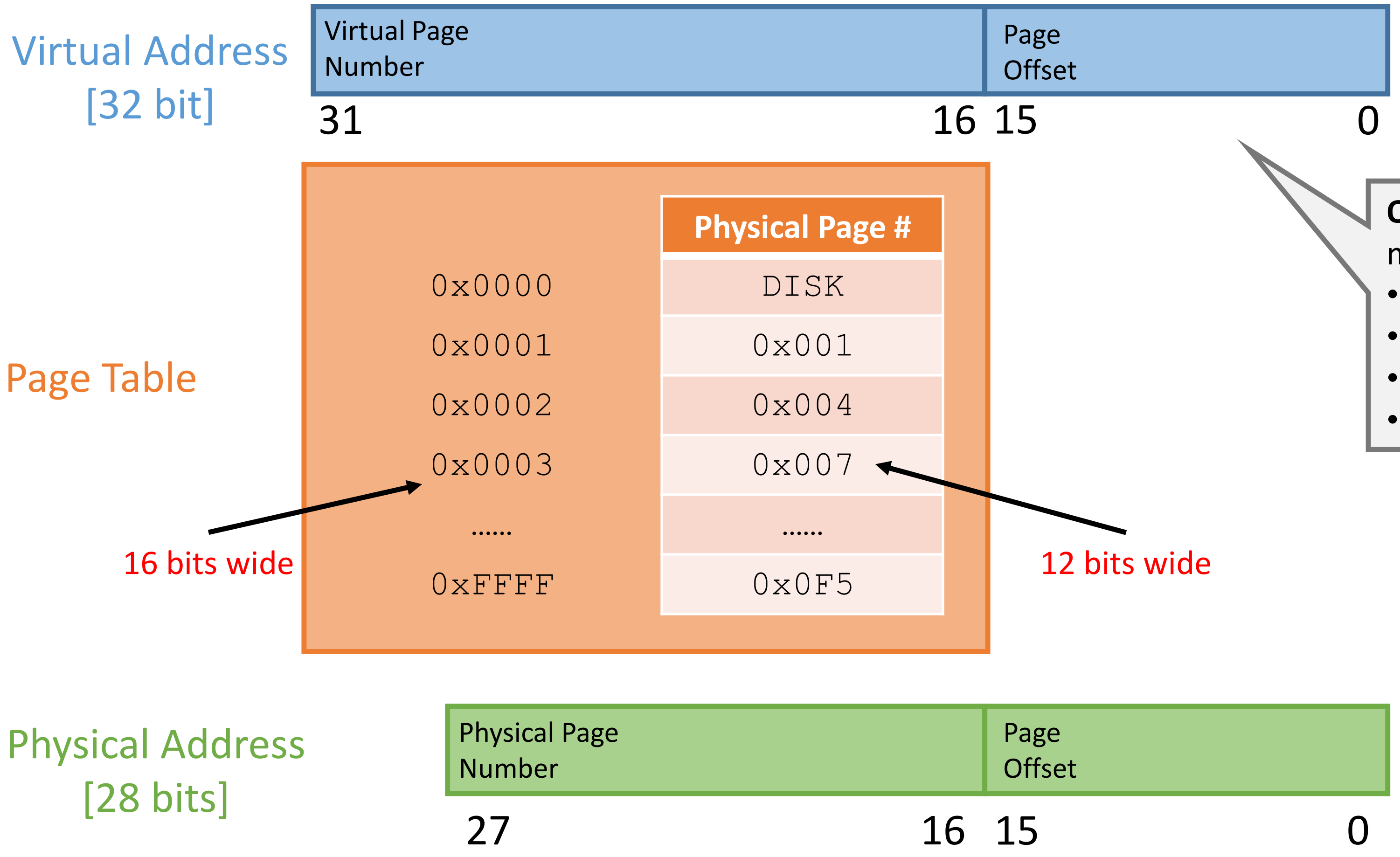
ISA:	32-bit
RAM:	256 MB
Page Size:	64 kB

In our terms:

32-bit Virtual Address

28-bit Physical Address

16-bit Page Offset



	Physical Page #
0x0000	DISK
0x0001	0x001
0x0002	0x004
0x0003	0x007
……	……
0xFFFF	0x0F5

Q: What if we use **64 kB pages** instead. How many bits do we need for the **page offset**?

- 12
- 13
- 16
- 18

A: 16

We need 16 bits to index every Word in a Page. This means my *Virtual Page Number* is now *16 bits* (32-16) and my *Physical Page Number* is *12 bits* (28-16).

Illustration from the textbook

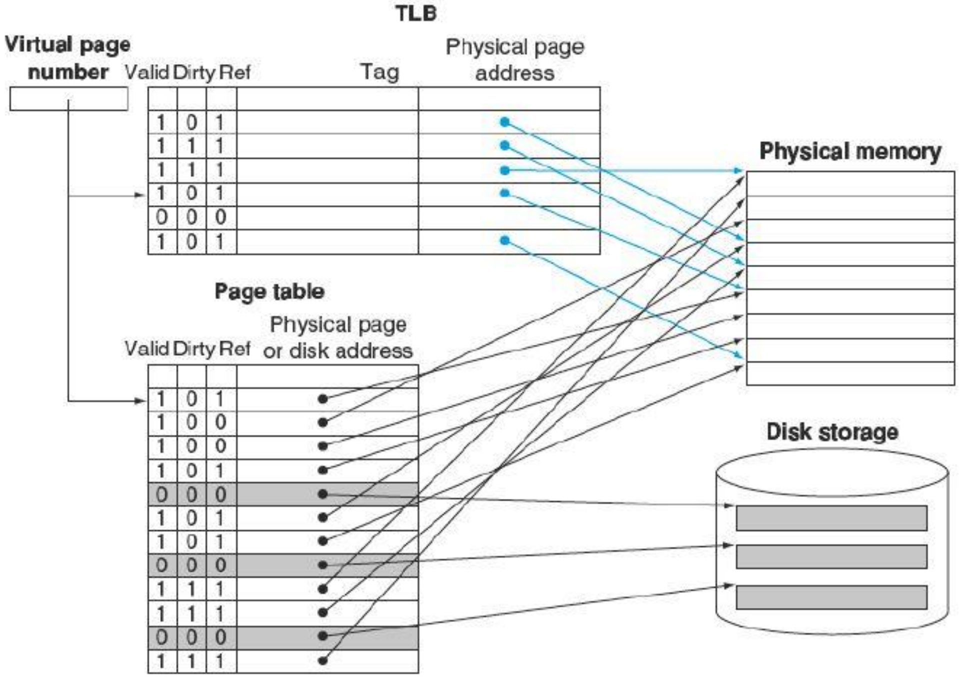


FIGURE 5.29 The TLB acts as a cache of the page table for the entries that map to physical pages only. The TLB contains a subset of the virtual-to-physical page mappings that are in the page table. The TLB mappings are shown in color. Because the TLB is a cache, it must have a tag field. If there is no matching entry in the TLB for a page, the page table must be examined. The page table either supplies a physical page number for the page (which can then be used to build a TLB entry) or indicates that the page resides on disk, in which case a page fault occurs. Since the page table has an entry for every virtual page, no tag field is needed; in other words, unlike a TLB, a page table is *not* a cache.

Page Table Entry Example Format

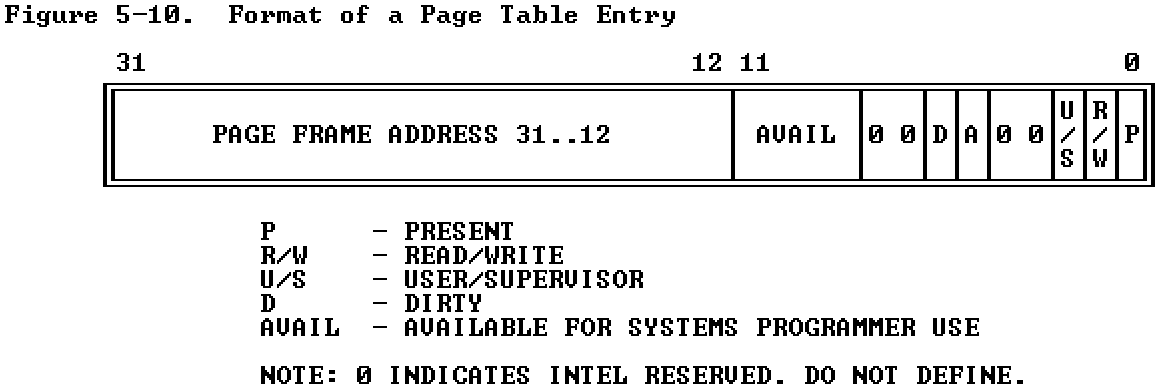


Figure 5-10. Format of a Page Table Entry

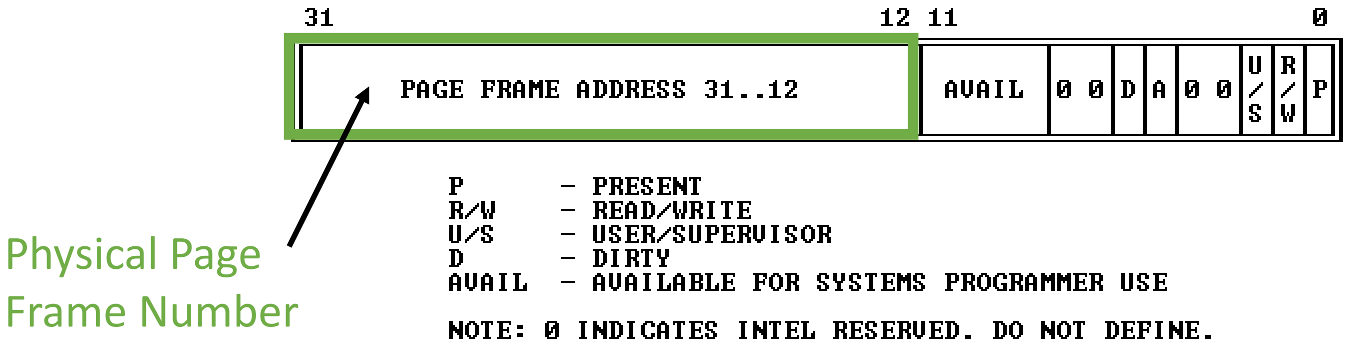
31 12	11							0
PAGE FRAME ADDRESS 31..12	AVAIL	0 0	D	A	0 0	U/S	R/W	P

P - PRESENT
R/W - READ/WRITE
U/S - USER/SUPERVISOR
D - DIRTY
AVAIL - AVAILABLE FOR SYSTEMS PROGRAMMER USE

NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.

Page Table Entry Example Format

Figure 5-10. Format of a Page Table Entry



P	-	PRESENT
R/W	-	READ/WRITE
U/S	-	USER/SUPERVISOR
D	-	DIRTY
AVAIL	-	AVAILABLE FOR SYSTEMS PROGRAMMER USE

NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.

Page Table Entry Example Format

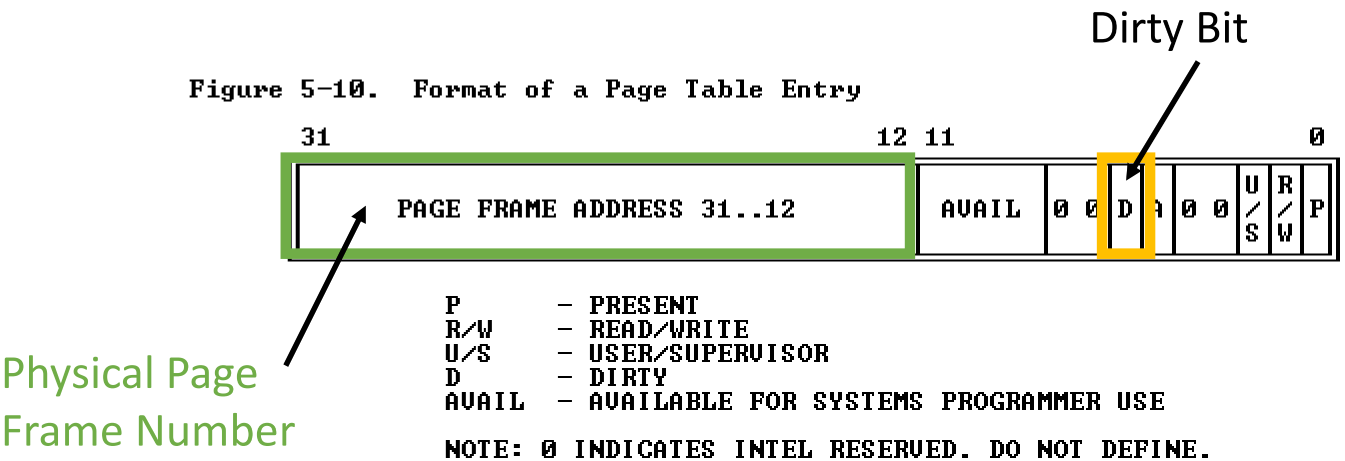


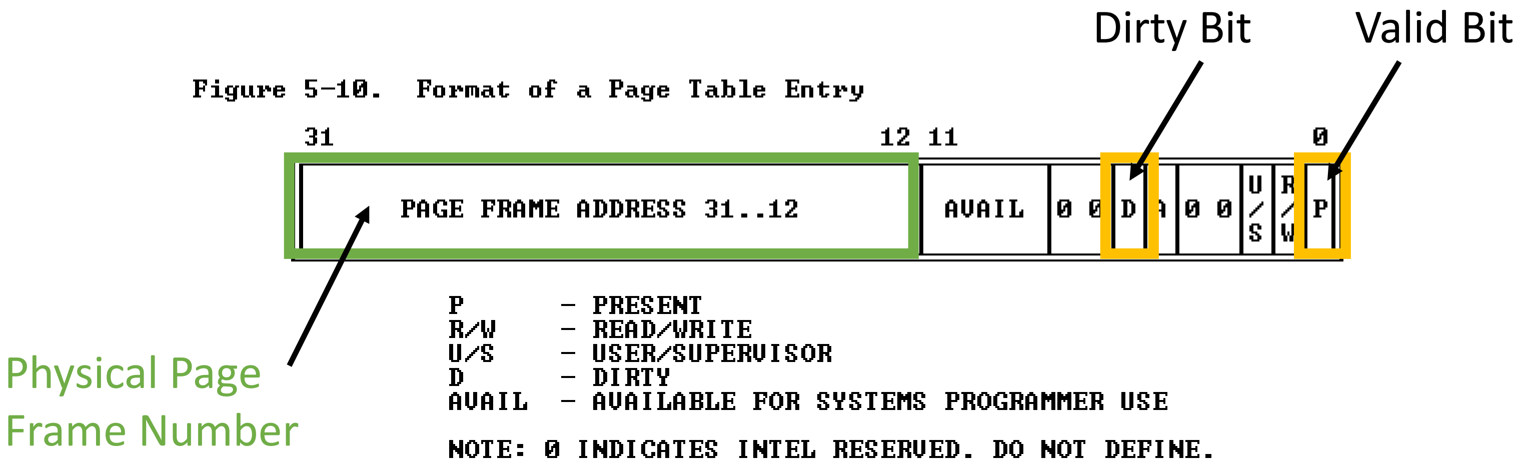
Figure 5-10. Format of a Page Table Entry

P	–	PRESENT
R/W	–	READ/WRITE
U/S	–	USER/SUPERVISOR
D	–	DIRTY
AVAIL	–	AVAILABLE FOR SYSTEMS PROGRAMMER USE

NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.

Page Table Entry Example Format

Figure 5-10. Format of a Page Table Entry



P	– PRESENT
R/W	– READ/WRITE
U/S	– USER/SUPERVISOR
D	– DIRTY
AVAIL	– AVAILABLE FOR SYSTEMS PROGRAMMER USE

NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.

References

- David Black-Schaffer: Lecture Series on Virtual Memory
- Patterson, Hennessy: Computer Organization and Design: the Hardware/Software Interface