

238P Operating Systems, Fall 2018

ELF Header, Real Mode Segmentation, Paging

9 November 2018

Aftab Hussain

University of California, Irvine

Reading ELF Header during boot

ELF structure in xv6 to read in ELF header: ([code](#))

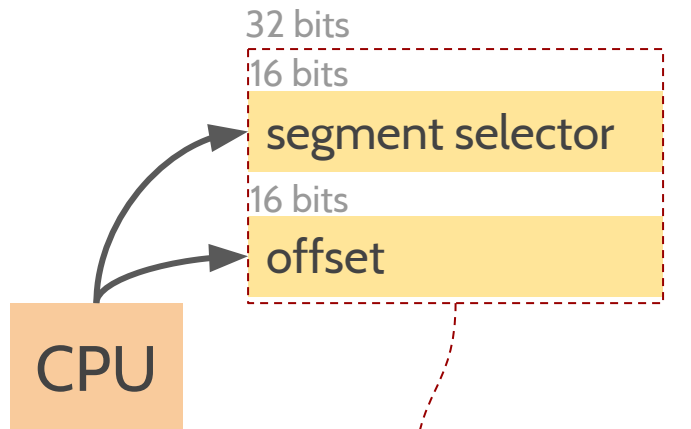
Where the ELF header of the kernel is read: ([code](#))

ELF Header Contents:

[OS 0 to 1: Chapter 5: The Anatomy of a Program.](#)

Review of Segmentation during boot in Real Mode

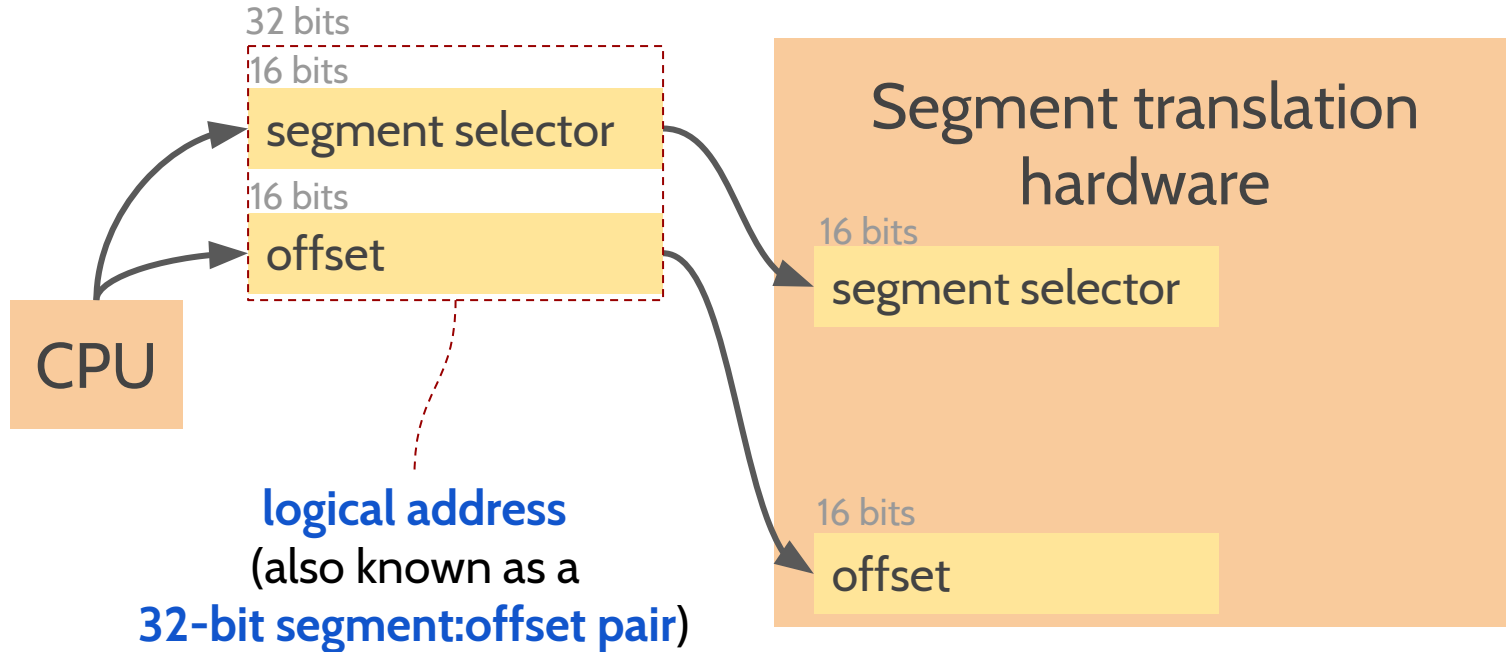
Segmentation in Real Mode



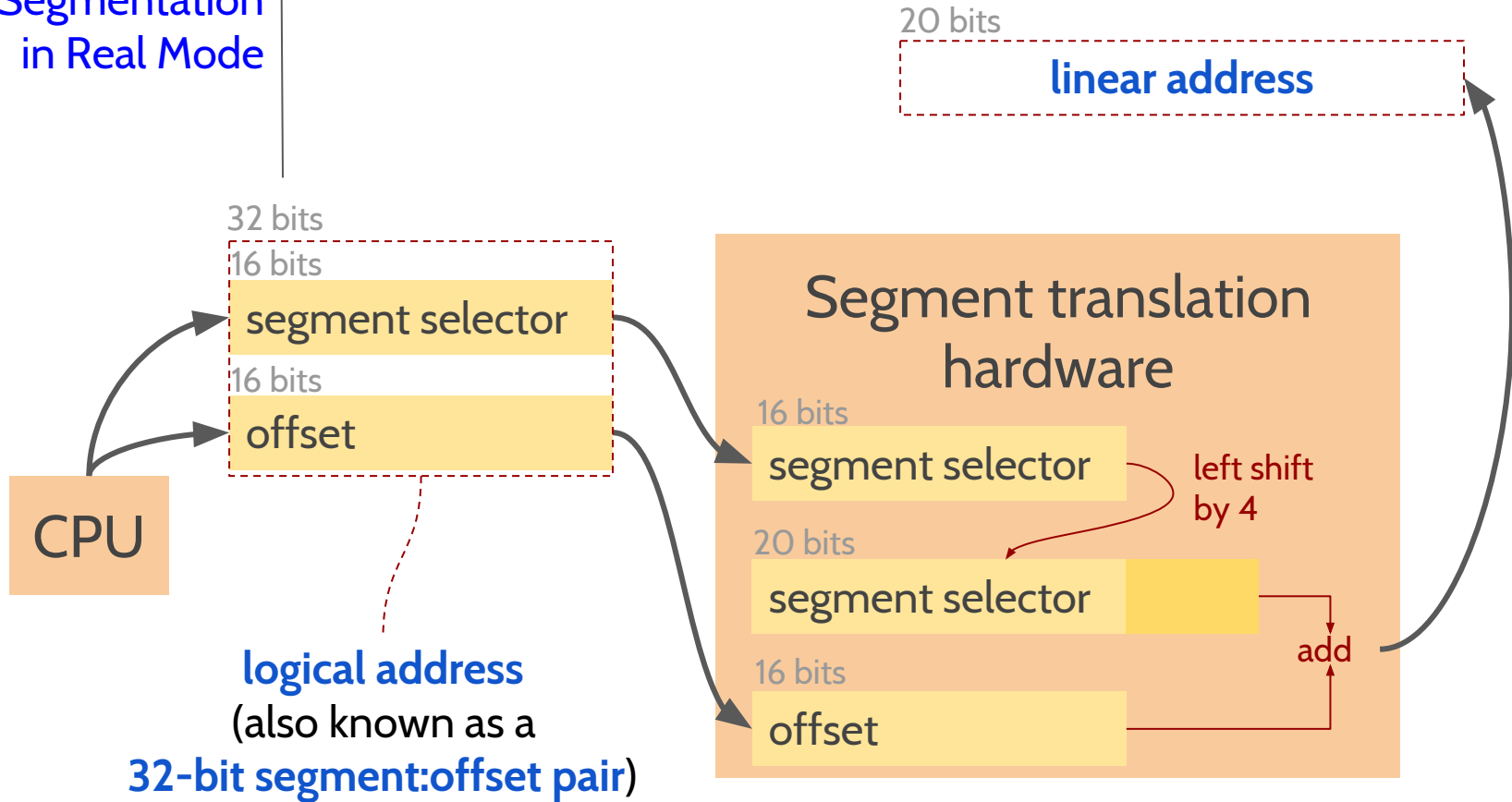
logical address
(also known as a
32-bit segment:offset pair)

xv6 refers to this x86 logical
address as a **virtual address**

Segmentation in Real Mode



Segmentation in Real Mode



Segmentation in Real Mode

directly
corresponds to the
physical address

20 bits

linear address

32 bits

16 bits

segment selector

16 bits

offset

CPU

logical address
(also known as a
32-bit segment:offset pair)

Segment translation
hardware

16 bits

segment selector

left shift
by 4

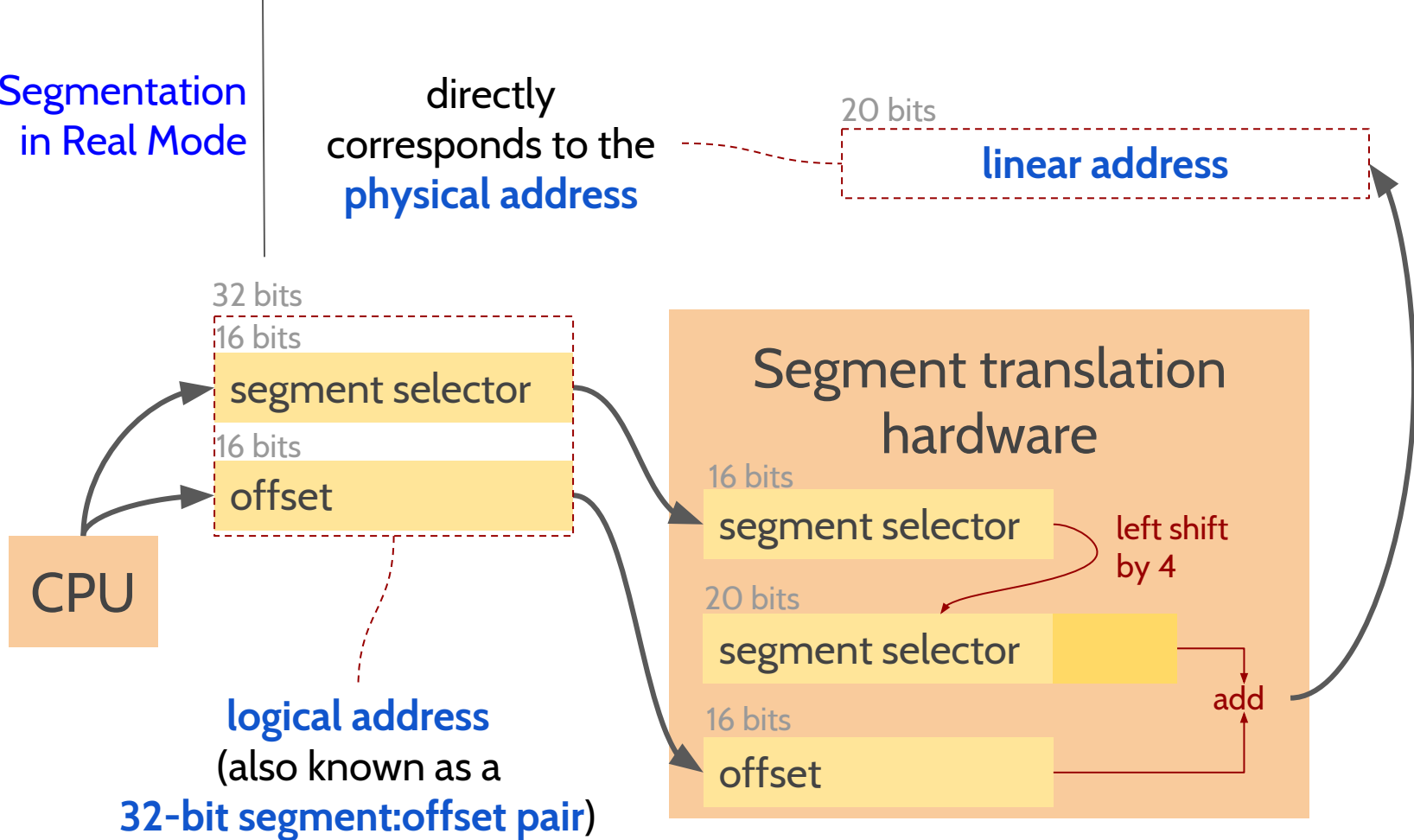
20 bits

segment selector

16 bits

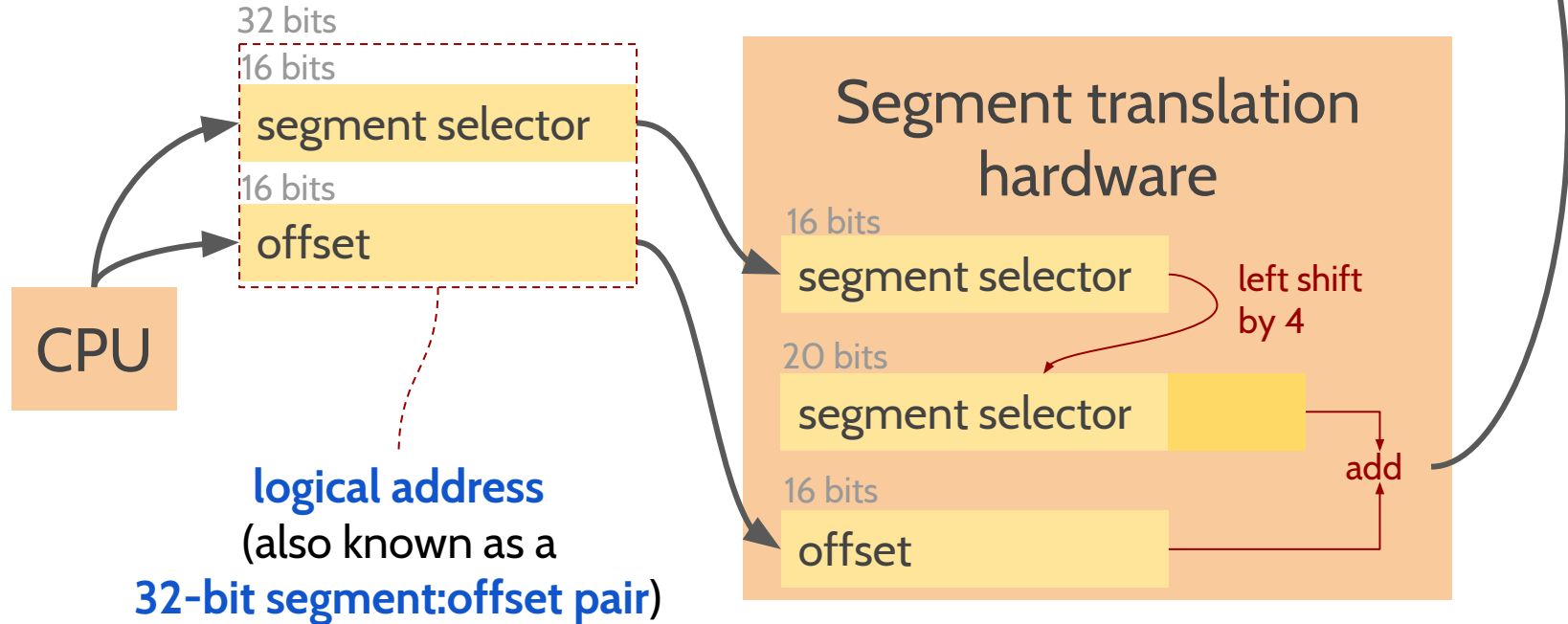
offset

add

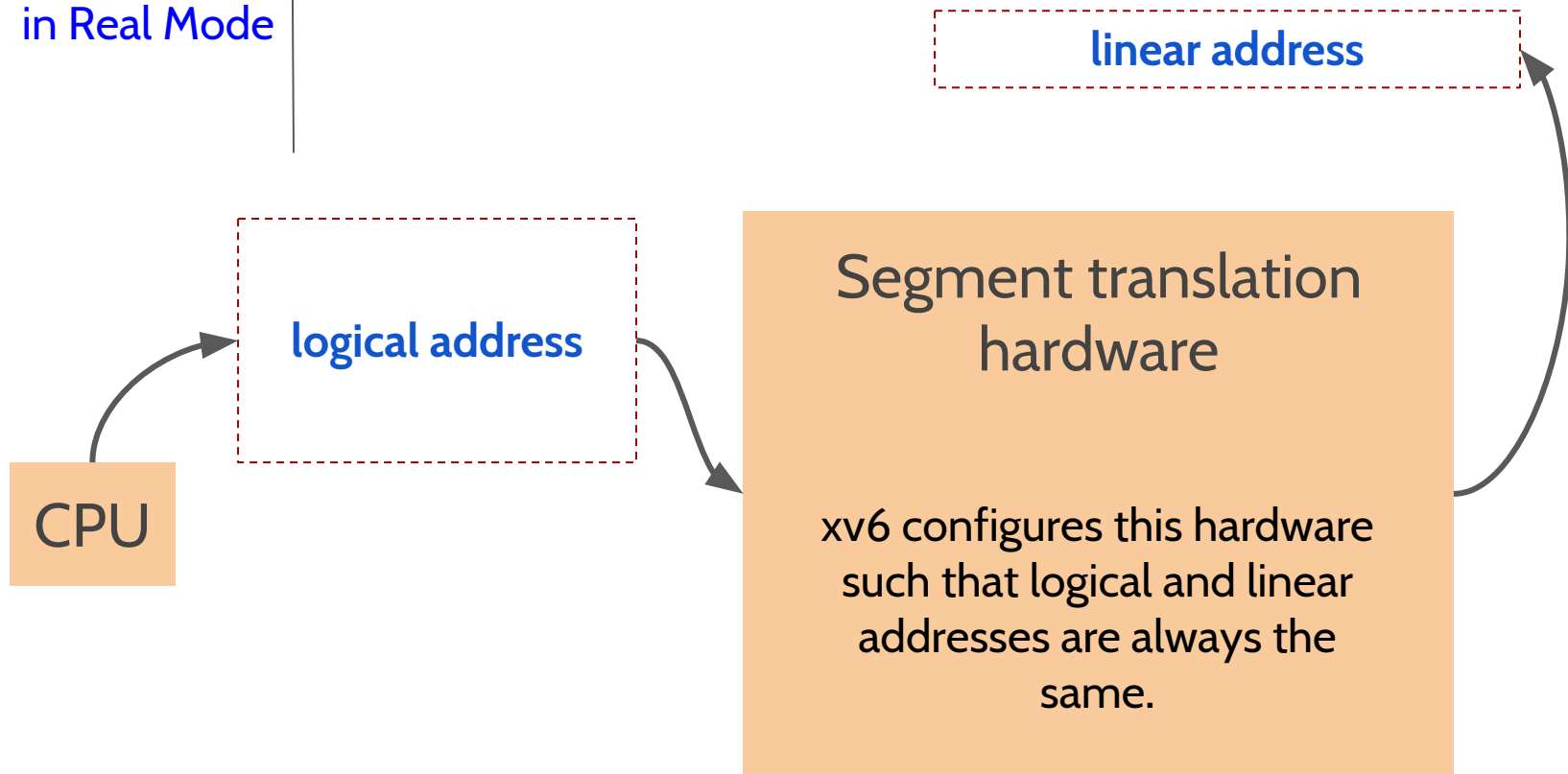


Segmentation in Real Mode

if paging is enabled, this address would go through a further translation process within the paging hardware to generate a physical address



Segmentation in Real Mode



Segmentation in Real Mode

*It follows (without paging) in
xv6,
logical address
= linear address
= physical address*

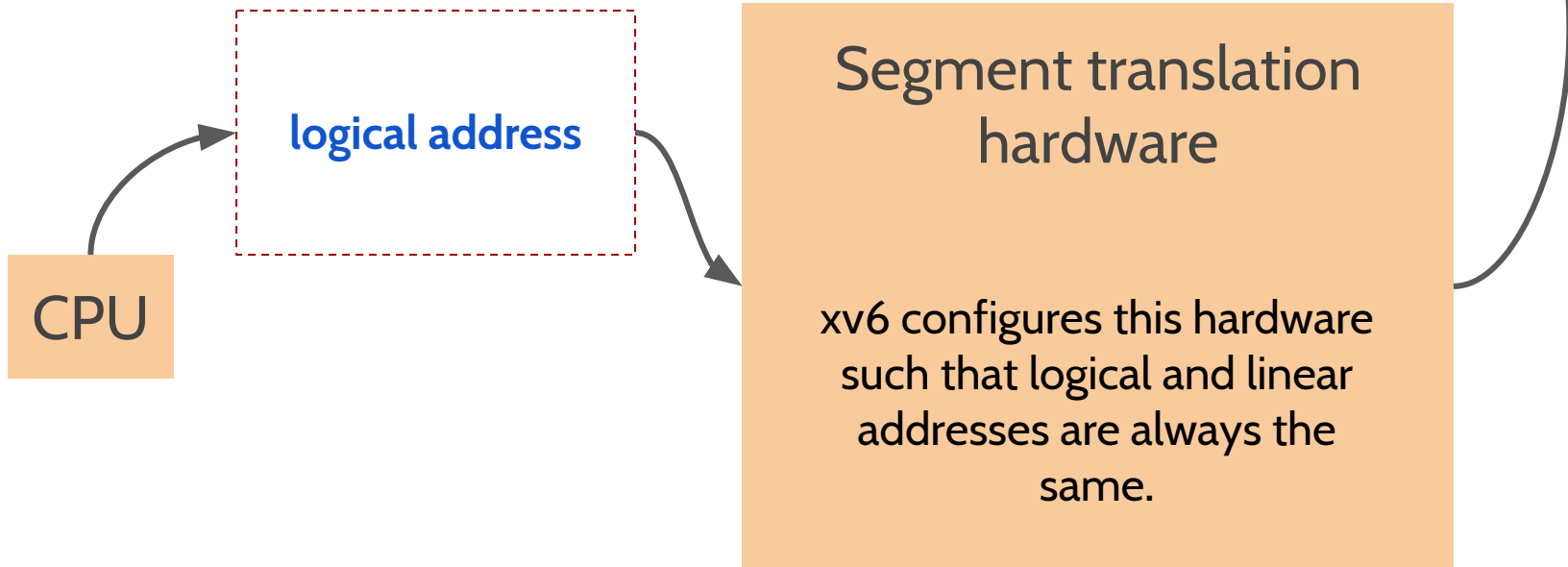
linear address

logical address

Segment translation
hardware

CPU

xv6 configures this hardware
such that logical and linear
addresses are always the
same.



Review of Address Translation using Paging

mov (%EBX), EAX # mov value from the location pointed by EBX into EAX

EAX = 0

EBX = 20 983 809

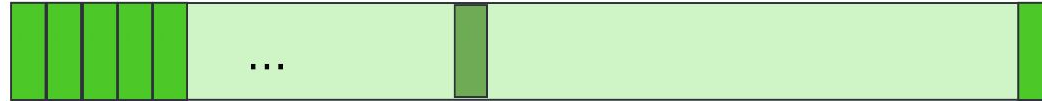
20 983 809 = 00 0000 0101 00 0000 0011 0000 0000 0001



page number

1M (1,048,575)

Virtual Address
Space (or Memory)
of the Process



0 1 2

page number = 5123
or (0b1 0100 0000 0011)

0 1 2 3 4 5 6 7 8 9 10 11 12

Physical
Memory



mov (%EBX), EAX # mov value from the location pointed by EBX into EAX

EAX = 0

EBX = 20 983 809

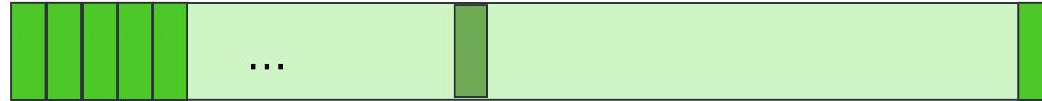
20 983 809 = 00 0000 0101 00 0000 0011 0000 0000 0001



page number

1M (1,048,575)

Virtual Address
Space (or Memory)
of the Process



0 1 2

page number = 5123
or (0b1 0100 0000 0011)

0 1 2 3 4 5 6 7 8 9 10 11 12

Physical
Memory



mov (%EBX), EAX # mov value from the location pointed by EBX into EAX

EAX = 0

EBX = 20 983 809

20 983 809 = 00 0000 0101 00 0000 0011 0000 0000 0001

page number

1M (1,048,575)

Virtual Address
Space (or Memory)
of the Process



CR3 = 0

0 1 2

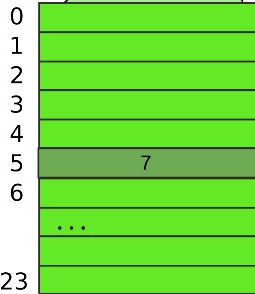
page number = 5123
or (0b1 0100 0000 0011)

0 1 2 3 4 5 6 7 8 9 10 11 12

Physical
Memory



32 bits (4 bytes)



Level 1
(Page Table
Directory)

mov (%EBX), EAX # mov value from the location pointed by EBX into EAX

EAX = 0

EBX = 20 983 809

20 983 809 = 00 0000 010 00 0000 0011 0000 0000 0001

page number

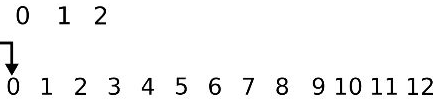
1M (1,048,575)

Virtual Address
Space (or Memory)
of the Process



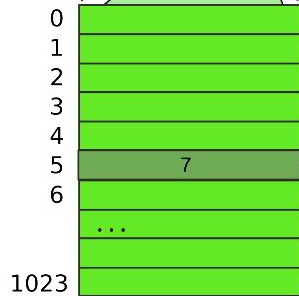
CR3 = 0

page number = 5123
or (0b1 0100 0000 0011)

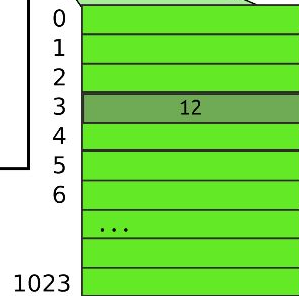


Physical
Memory

32 bits (4 bytes)



Level 1
(Page Table
Directory)



Level 2
(Page Table)

mov (%EBX), EAX # mov value from the location pointed by EBX into EAX

EAX = 0

EBX = 20 983 809

20 983 809 = 00 0000 0101 00 0000 0011 0000 0000 0001

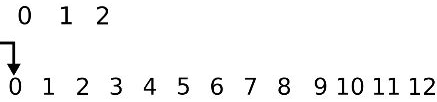
page number

1M (1,048,575)

Virtual Address
Space (or Memory)
of the Process



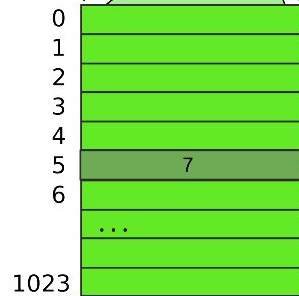
CR3 = 0



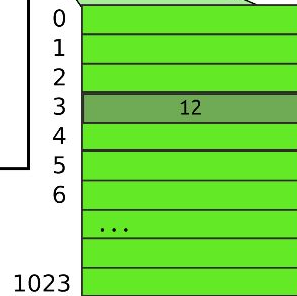
page number = 5123
or (0b1 0100 0000 0011)

Physical
Memory

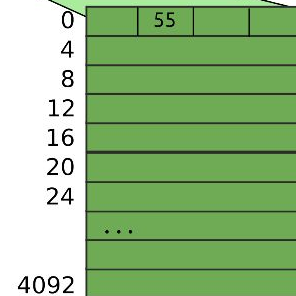
32 bits (4 bytes)



Level 1
(Page Table
Directory)



Level 2
(Page Table)



Page

mov(%EBX), EAX # mov value from the location pointed by EBX into EAX

EAX = 0

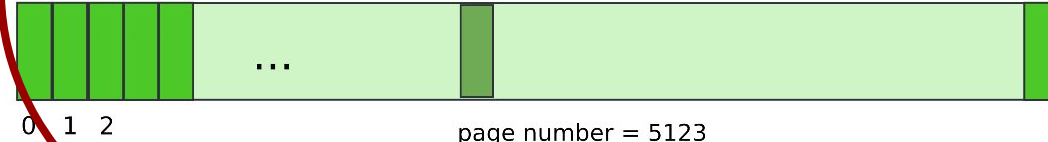
EBX = 20 983 809

20 983 809 = 00 0000 0101 00 0000 001 0000 0000 0001

page number

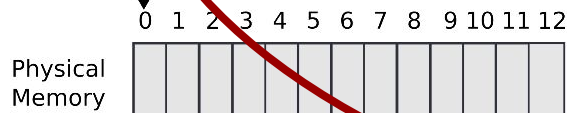
1M (1,048,575)

Virtual Address
Space (or Memory)
of the Process



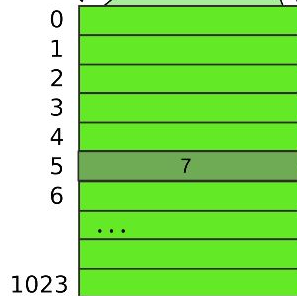
CR3 = 0

page number = 5123
or (0b1 0100 0000 0011)

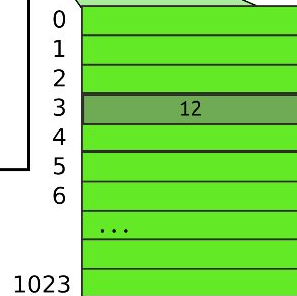


Physical
Memory

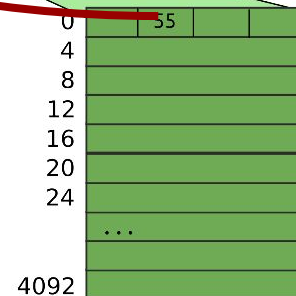
32 bits (4 bytes)



Level 1
(Page Table
Directory)



Level 2
(Page Table)



Page