Operating Systems
Lecture 13: TLB + Advanced Paging

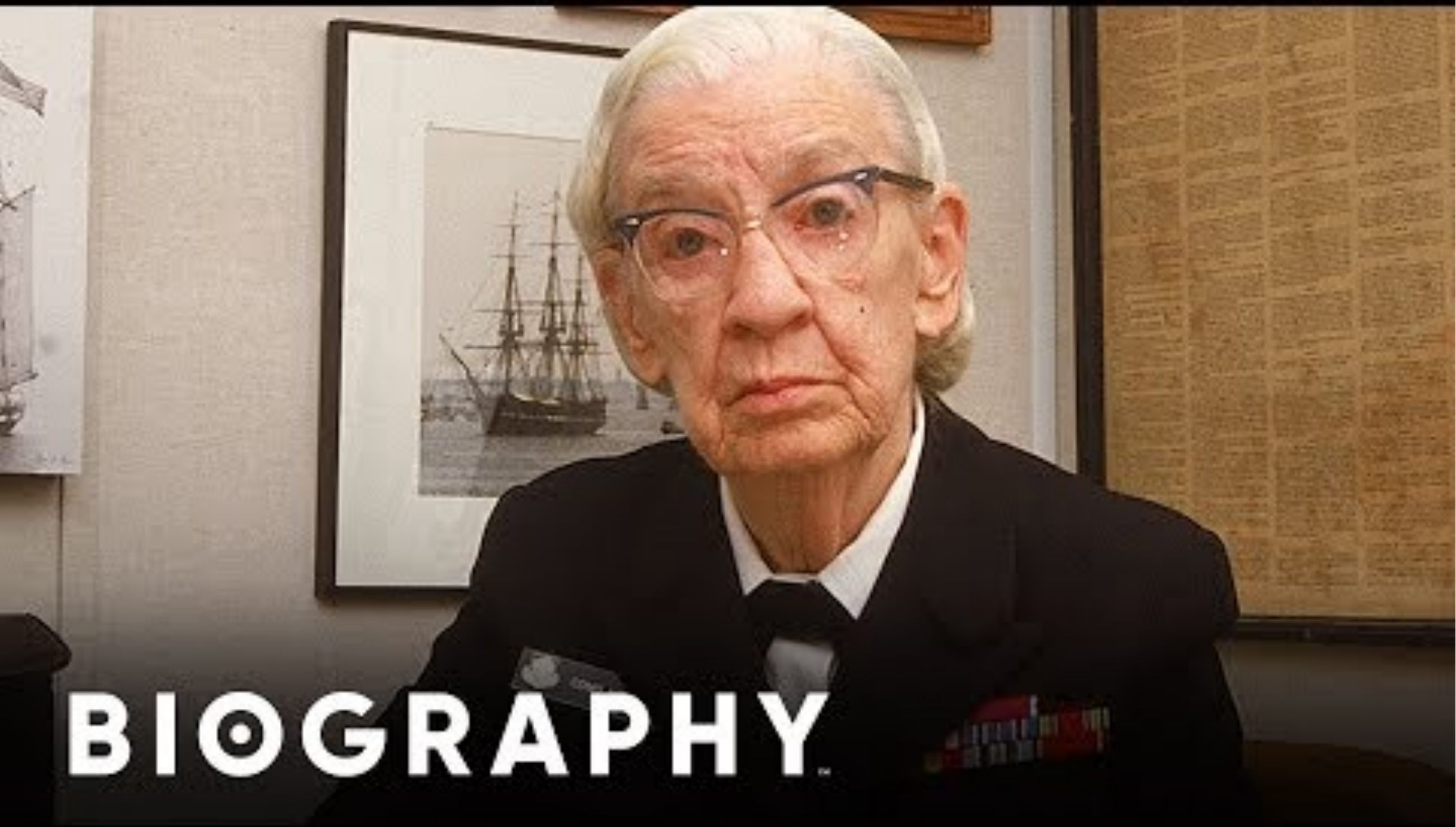
Nipun Batra
Aug 30, 2018

Grace Hopper

Queen of Software

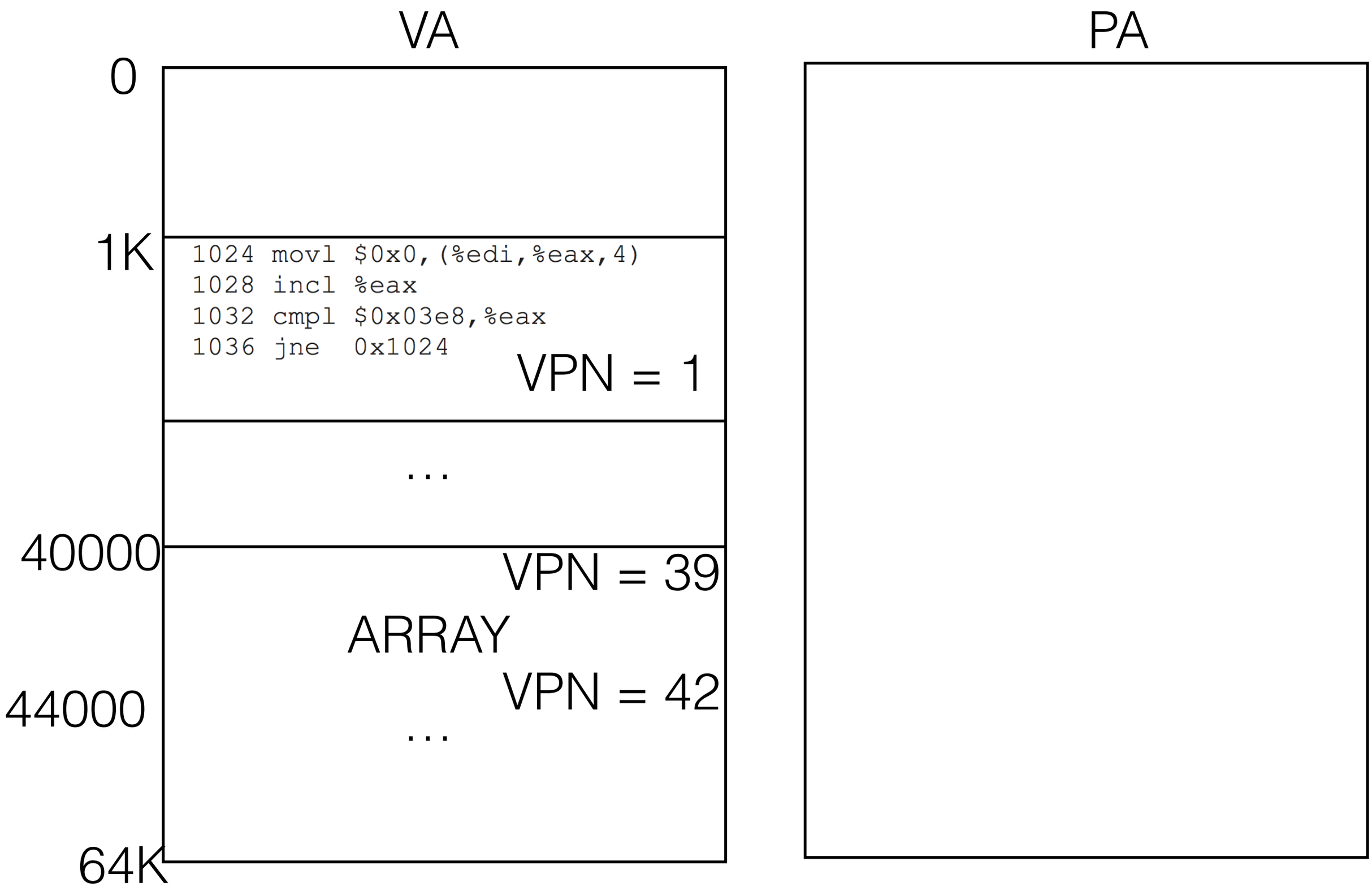
Highlights

- Programmed the World's First Compiler
- Popularized the idea of Machine Independent language
- Invented COBOL (Common Business-Oriented Language)
- Coined the term "Compiler"

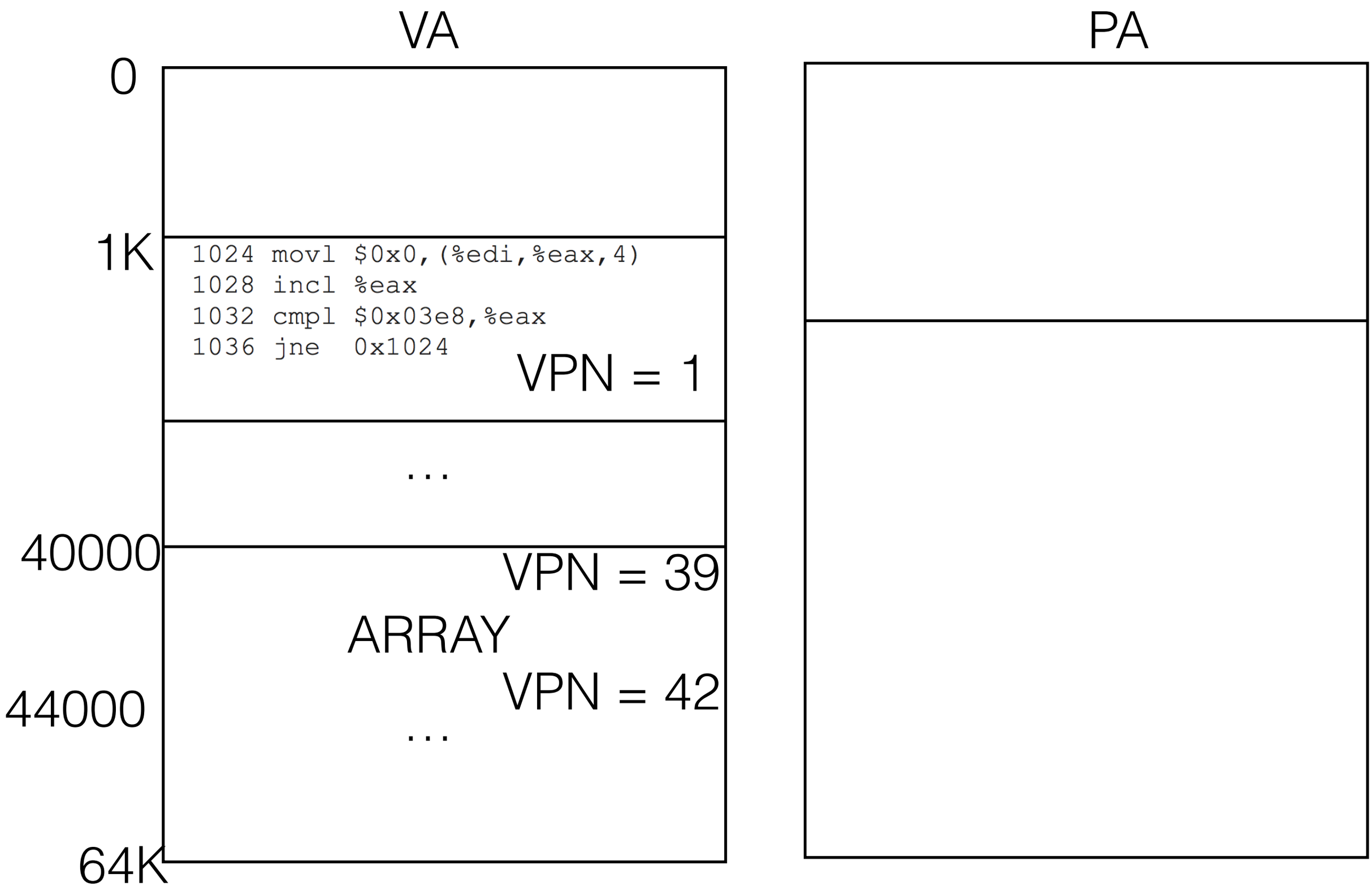


BIOGRAPHY™

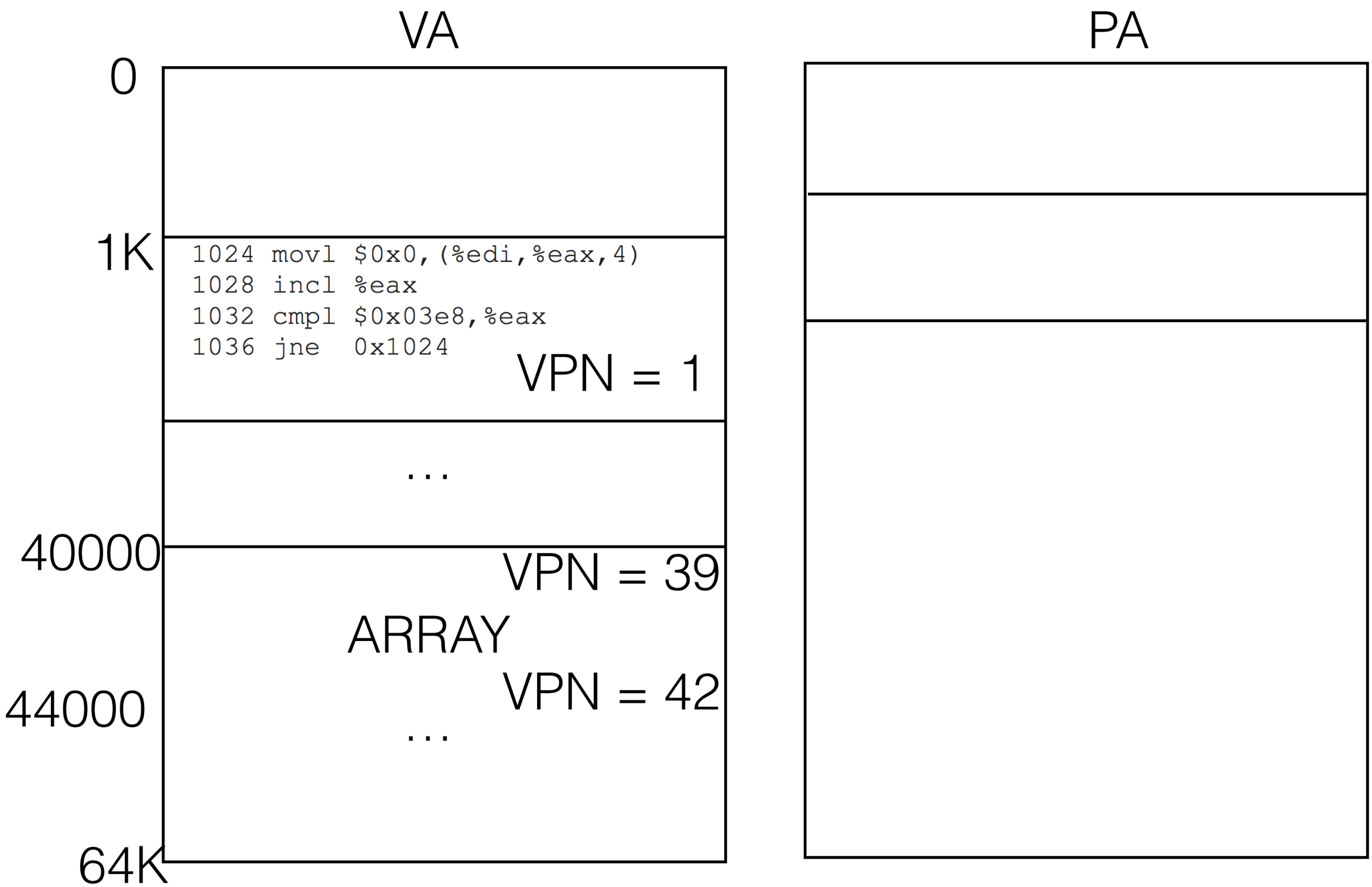
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...

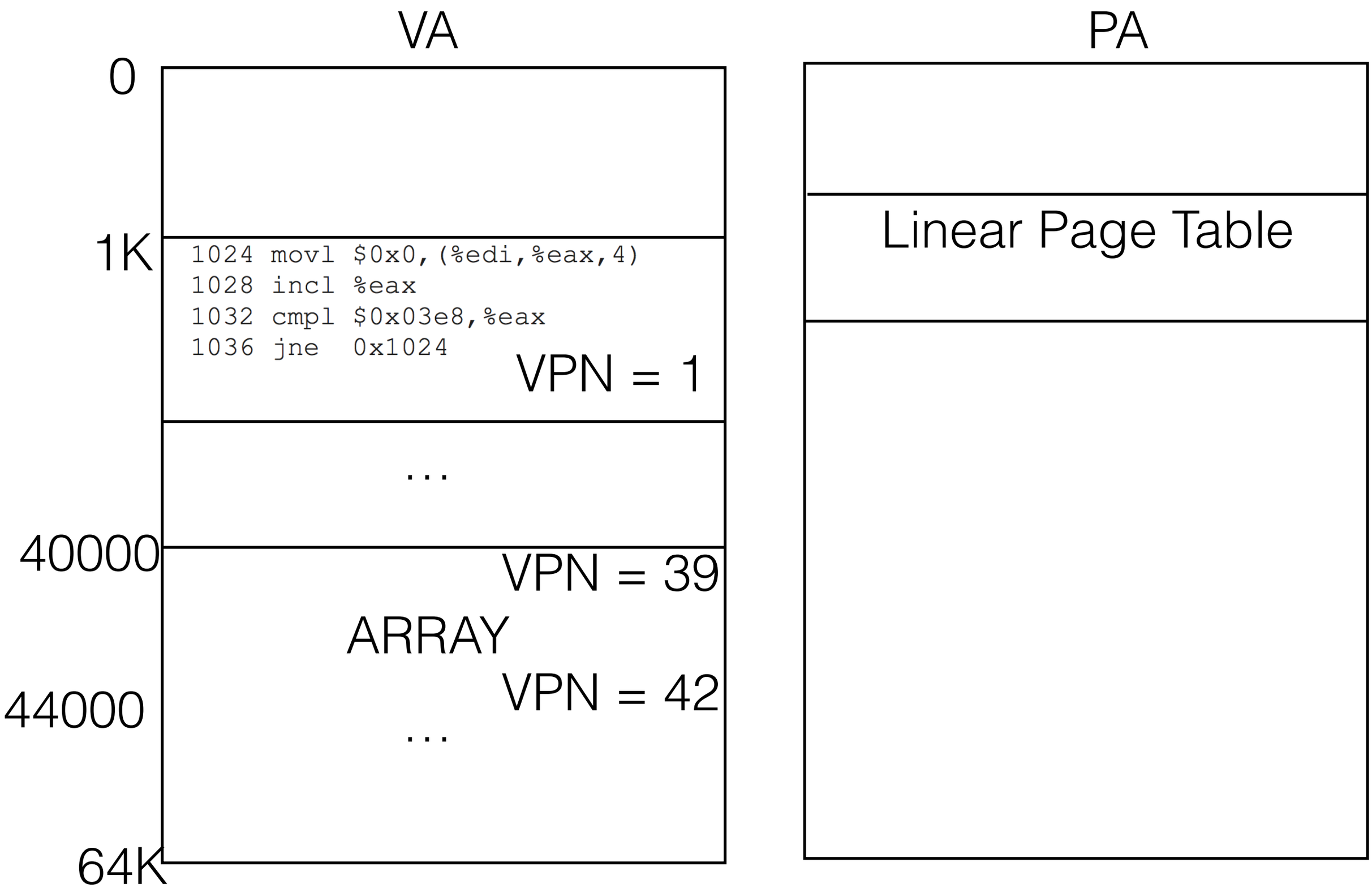
Revision


VA
PA
0
1K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
40000
VPN = 39
ARRAY
VPN = 42
44000
...
64K

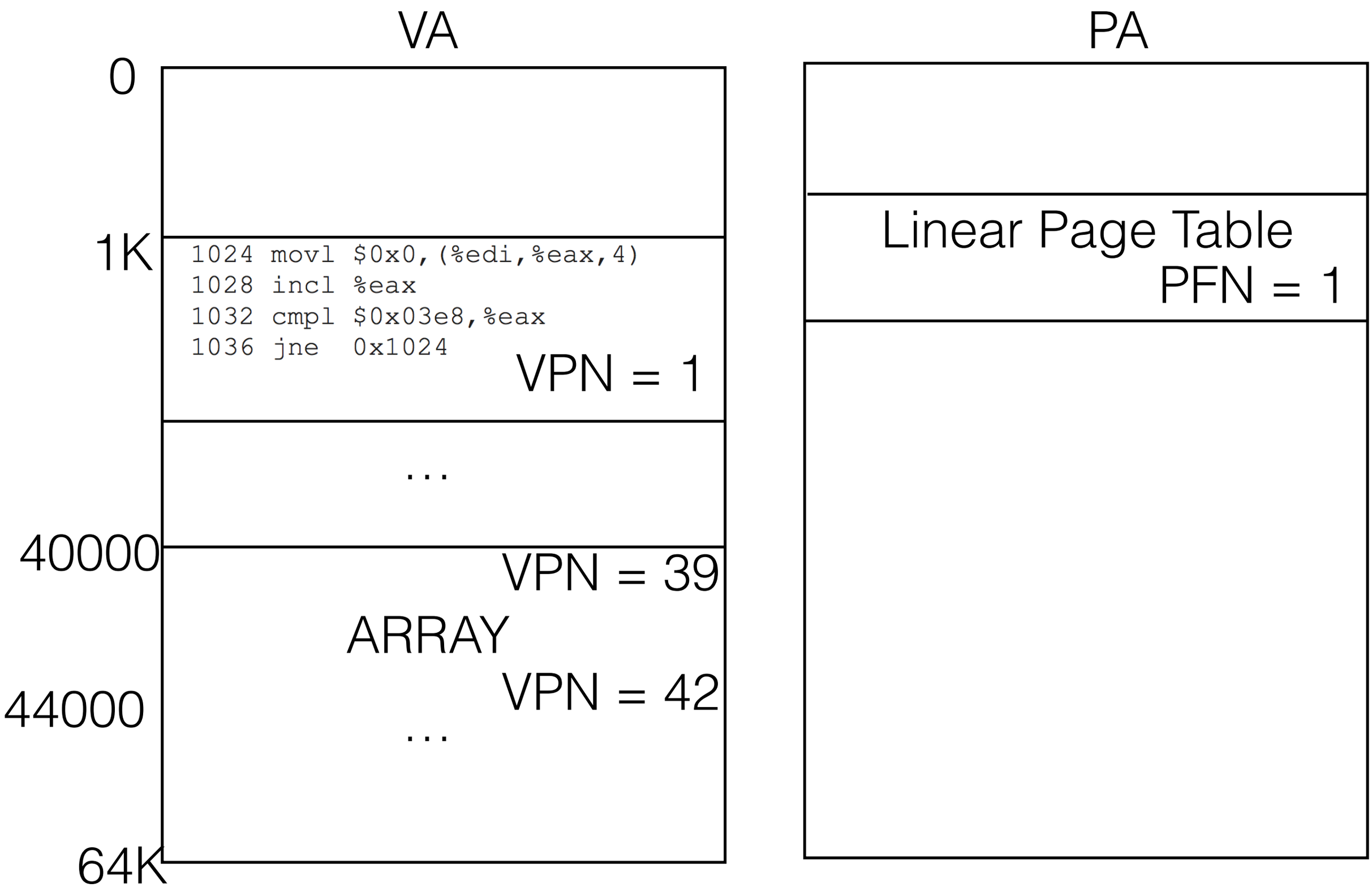
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...

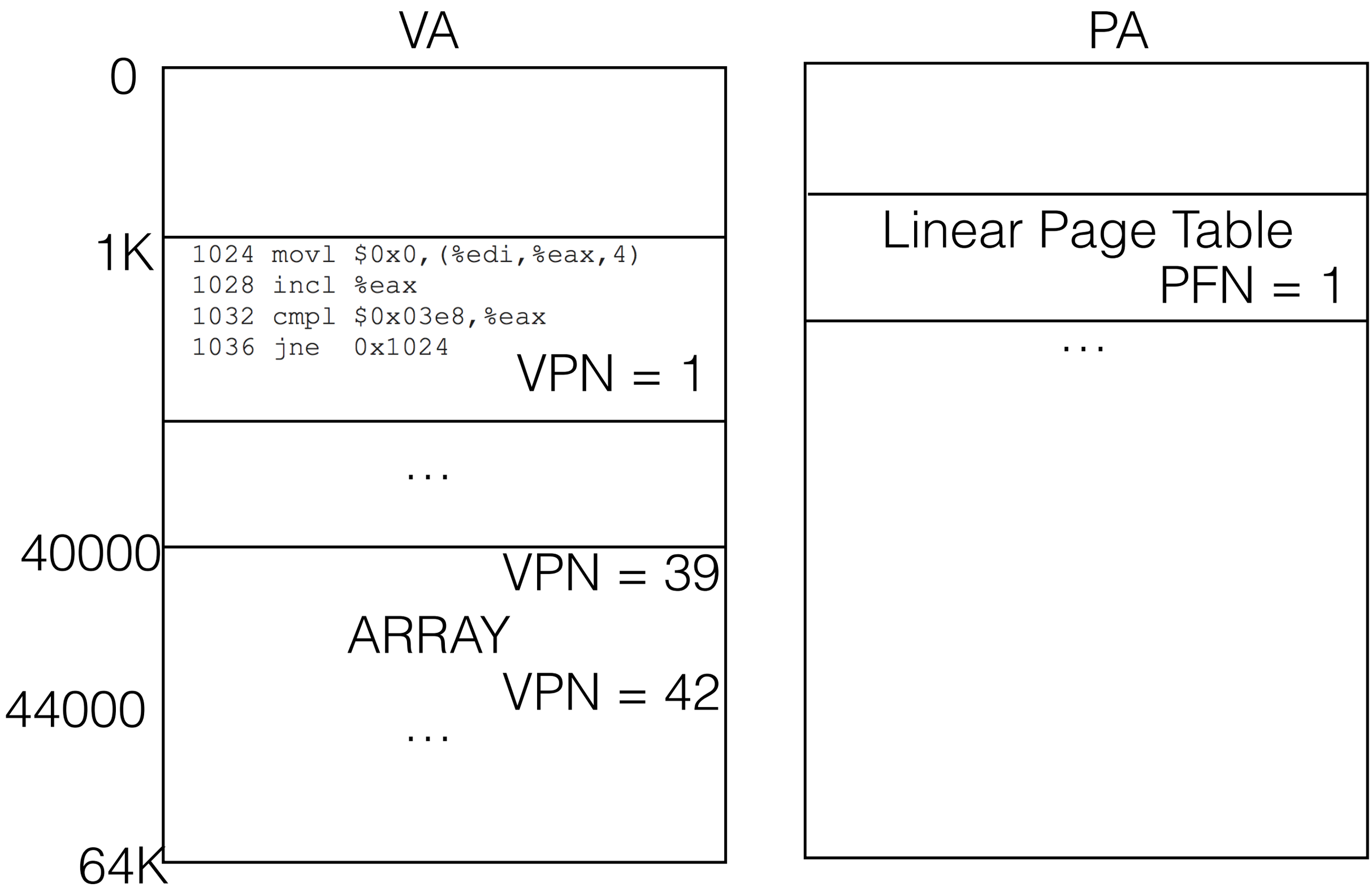
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table

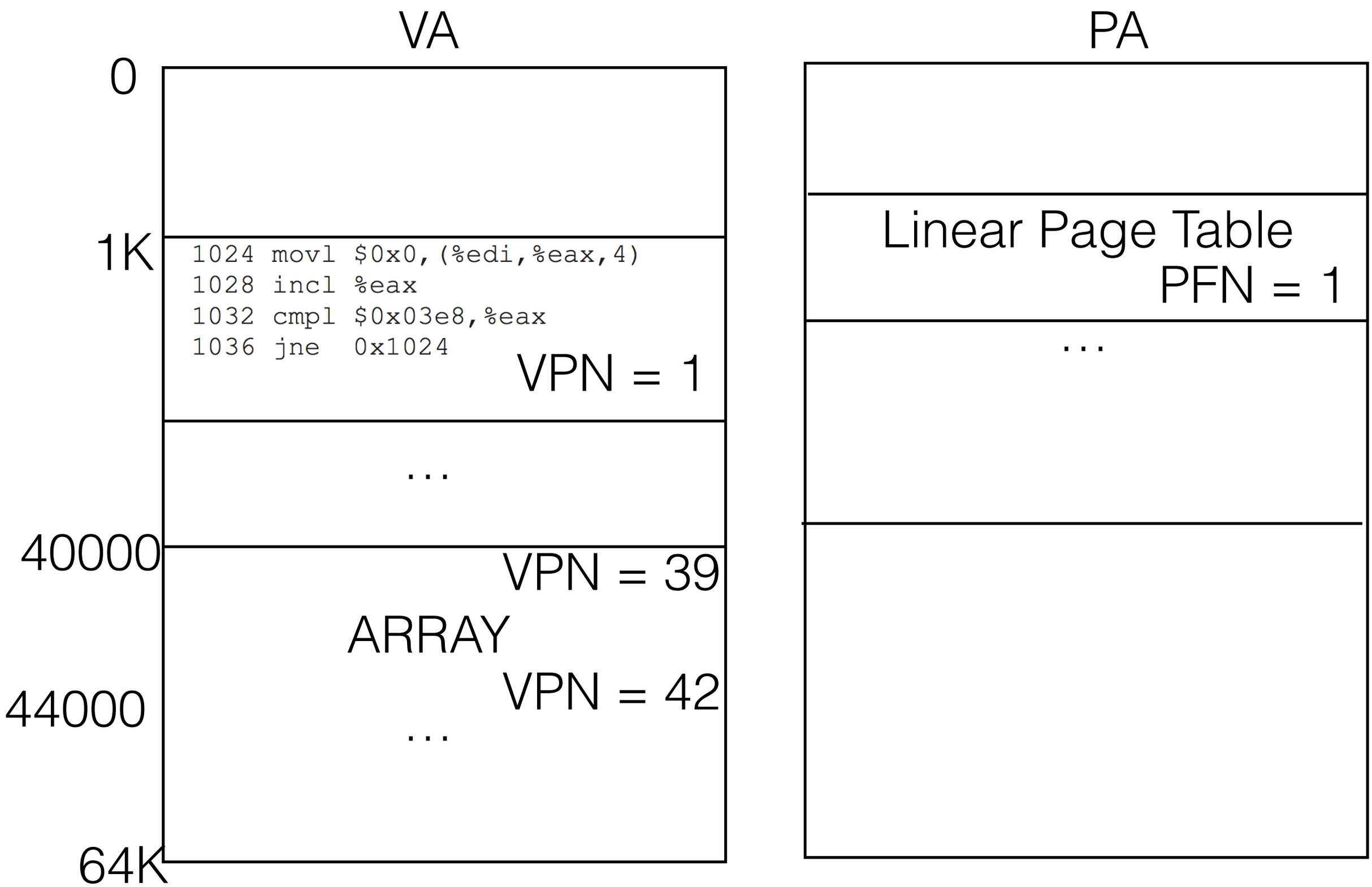
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1

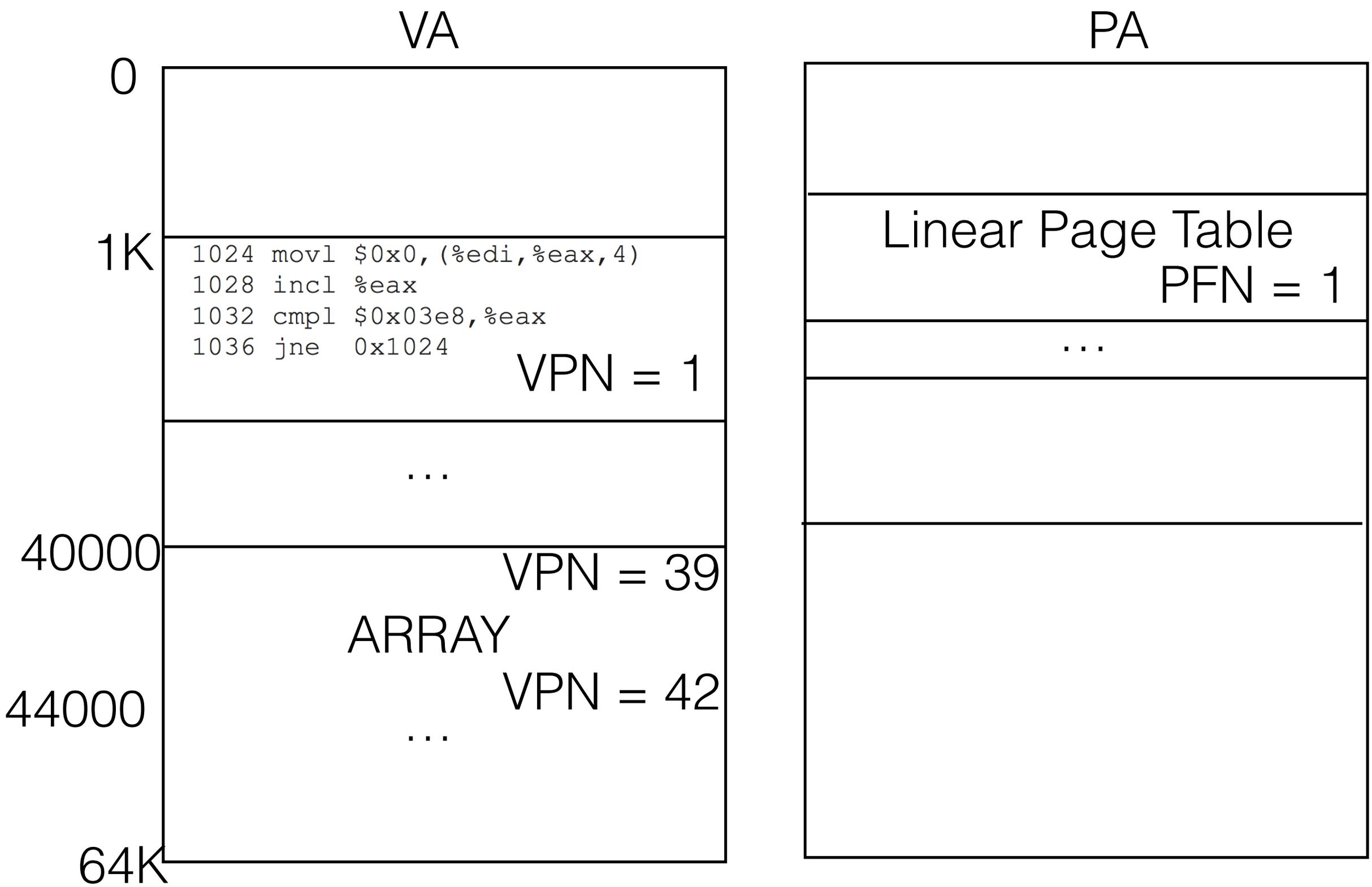
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...

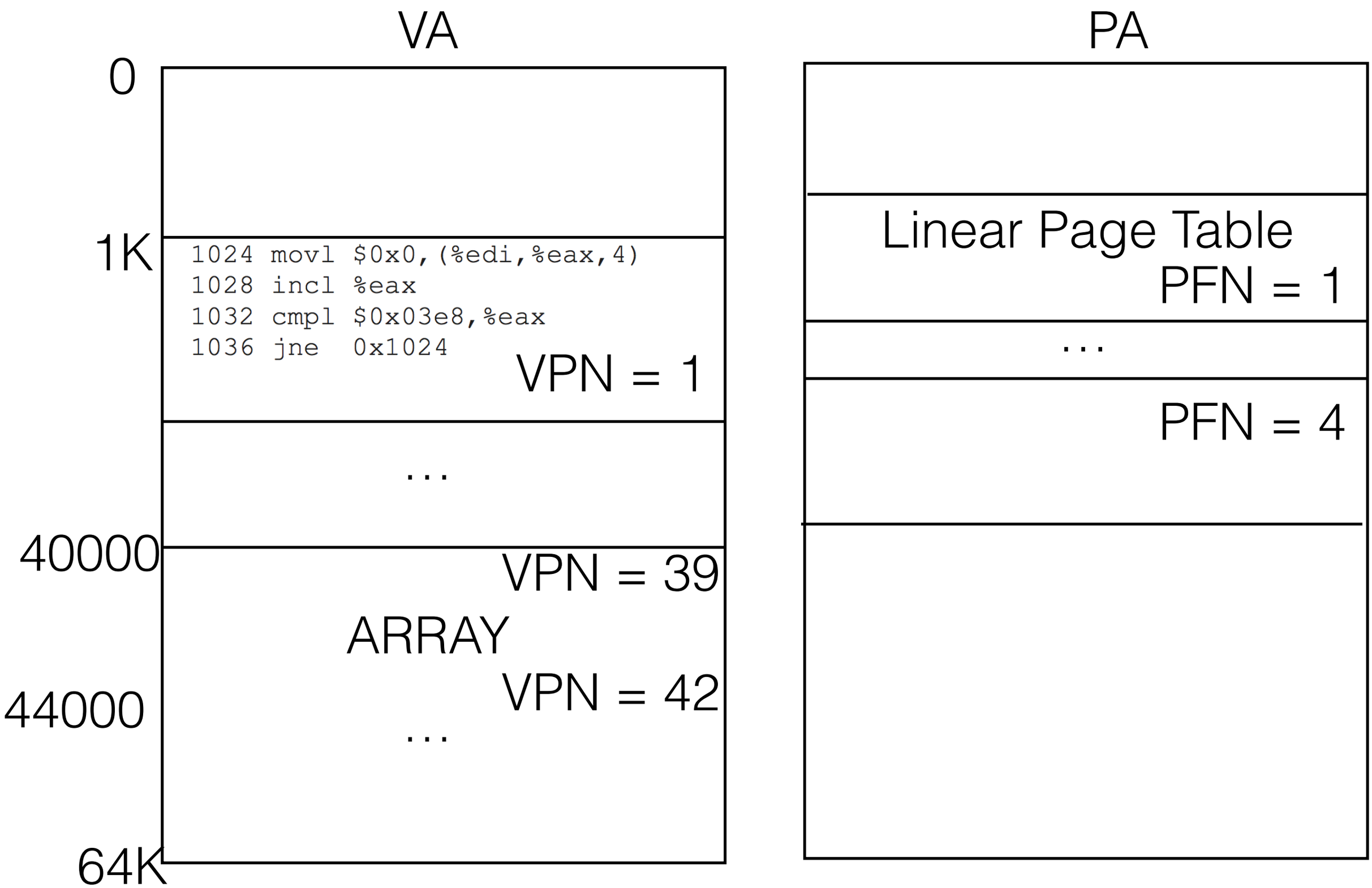
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...

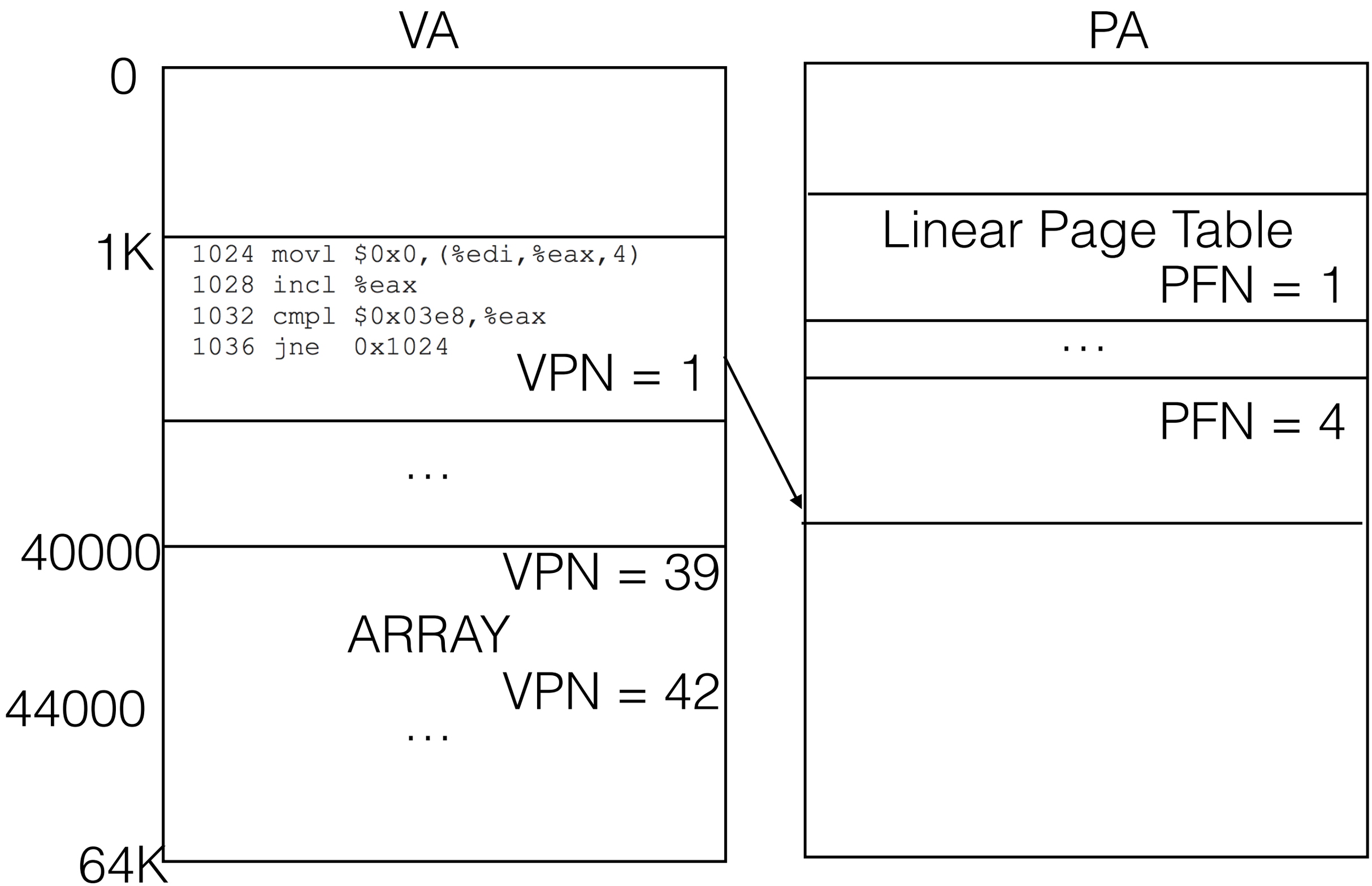
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...

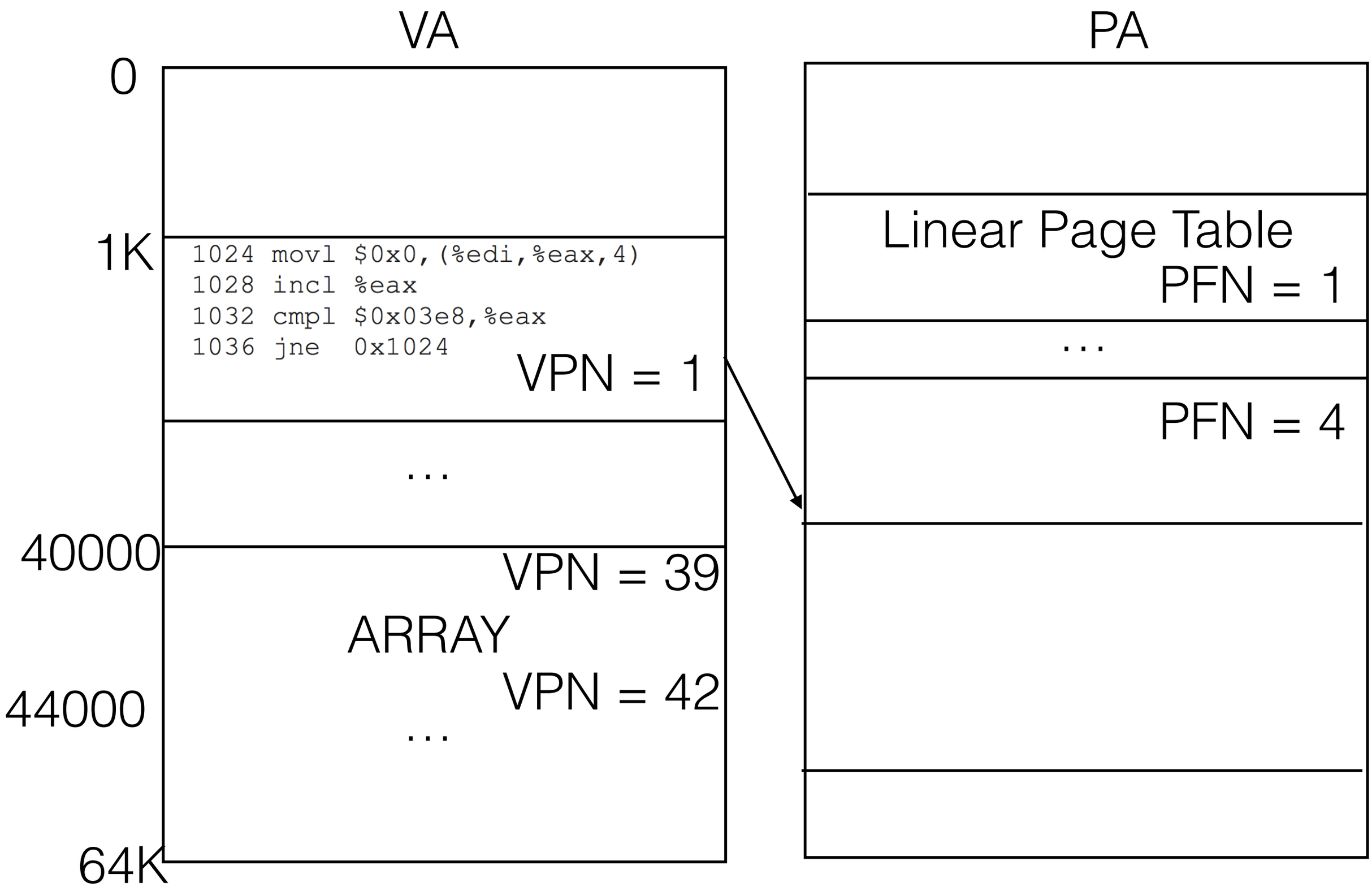
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...
PFN = 4

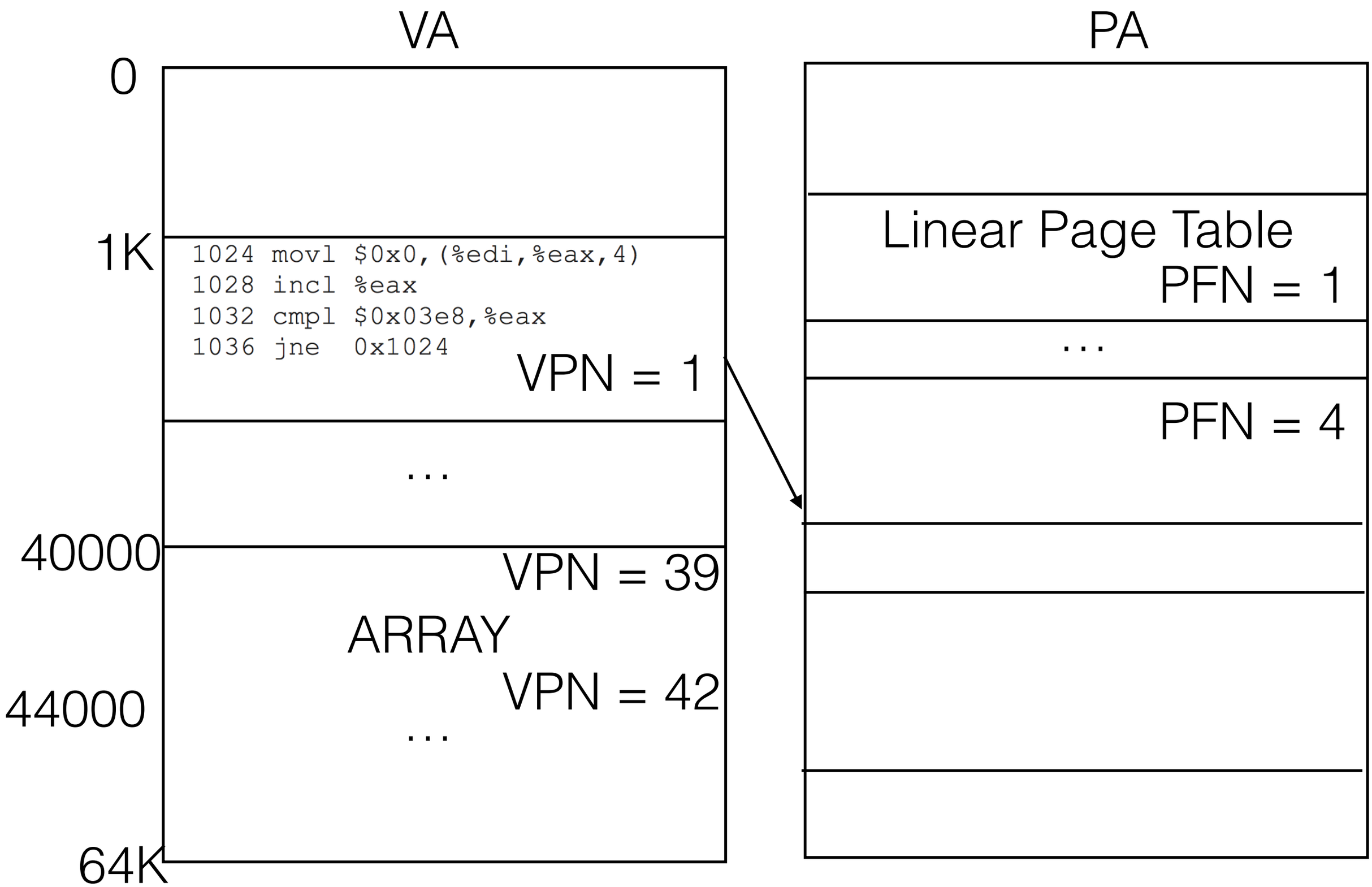
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...
PFN = 4

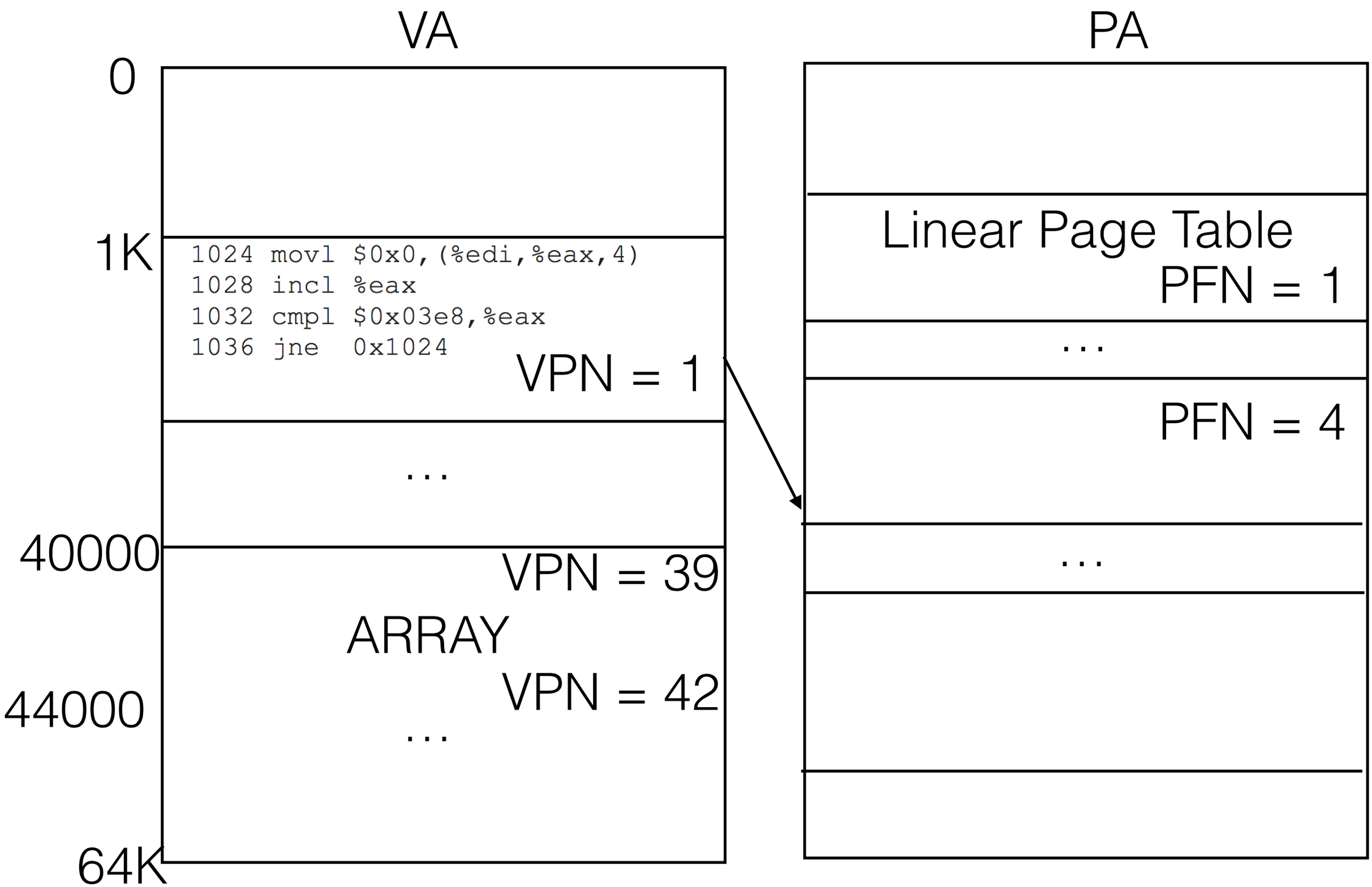
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...
PFN = 4

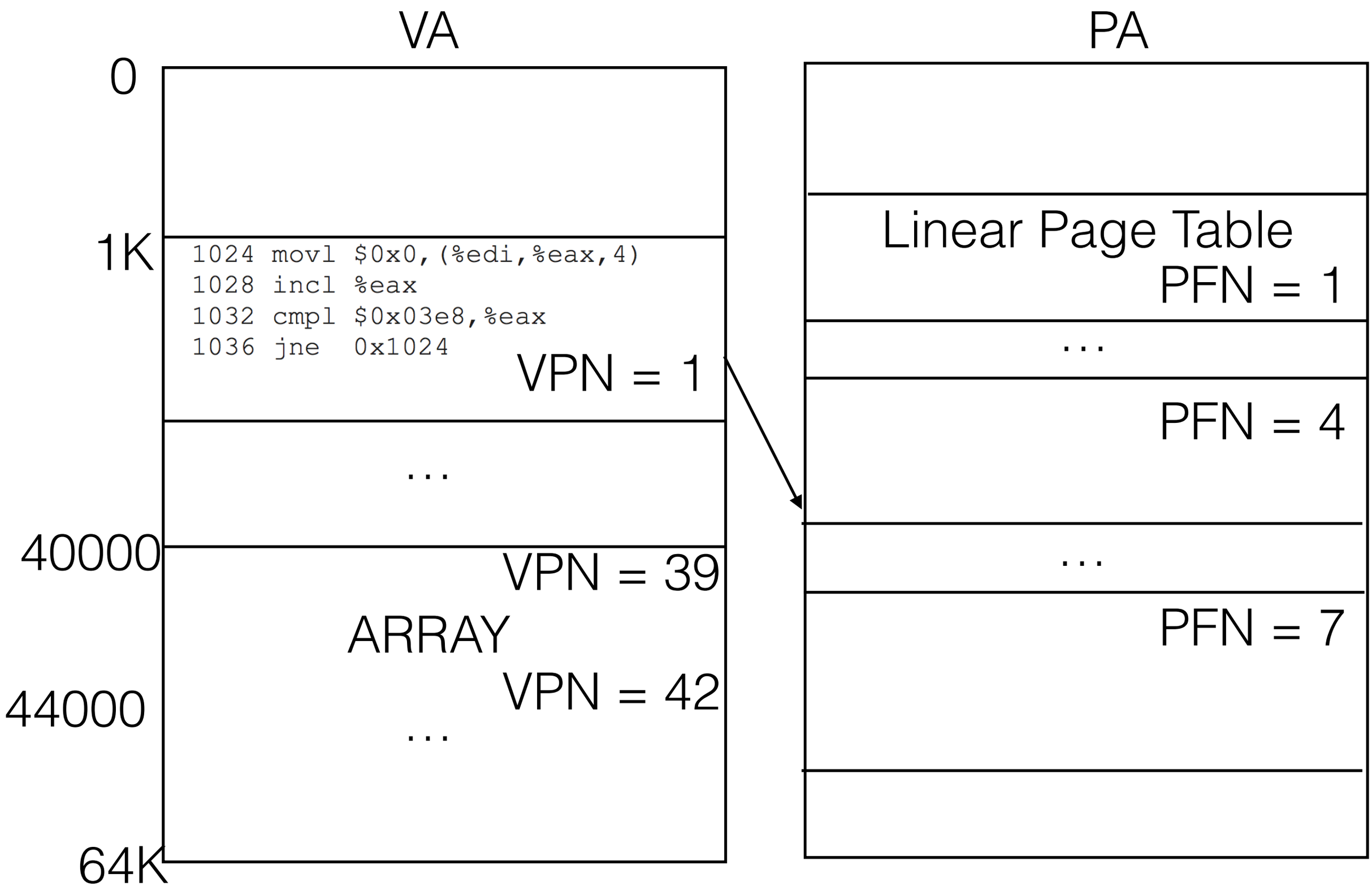
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...
PFN = 4

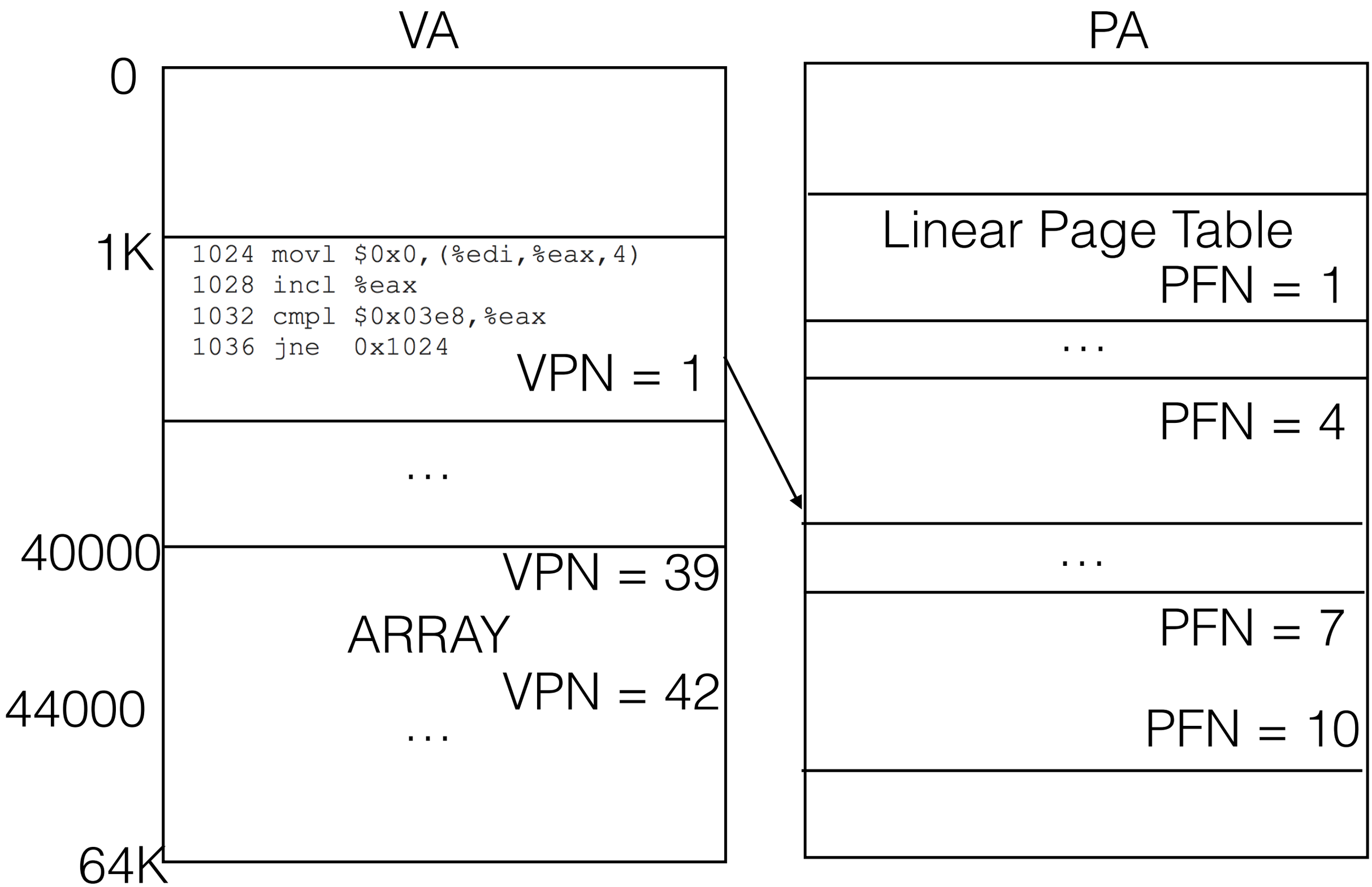
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...
PFN = 4
...

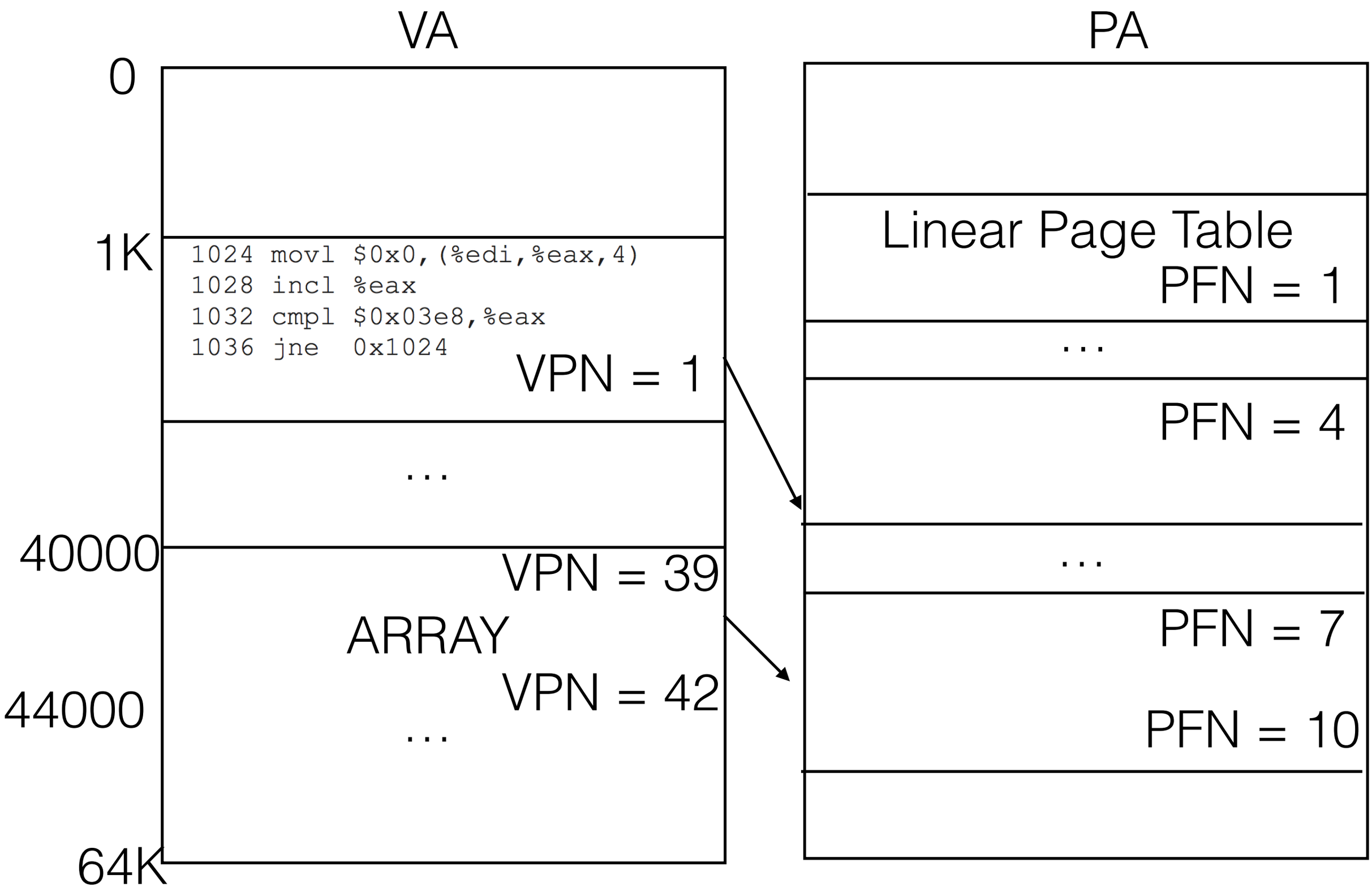
Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...
PFN = 4
...
PFN = 7

Revision

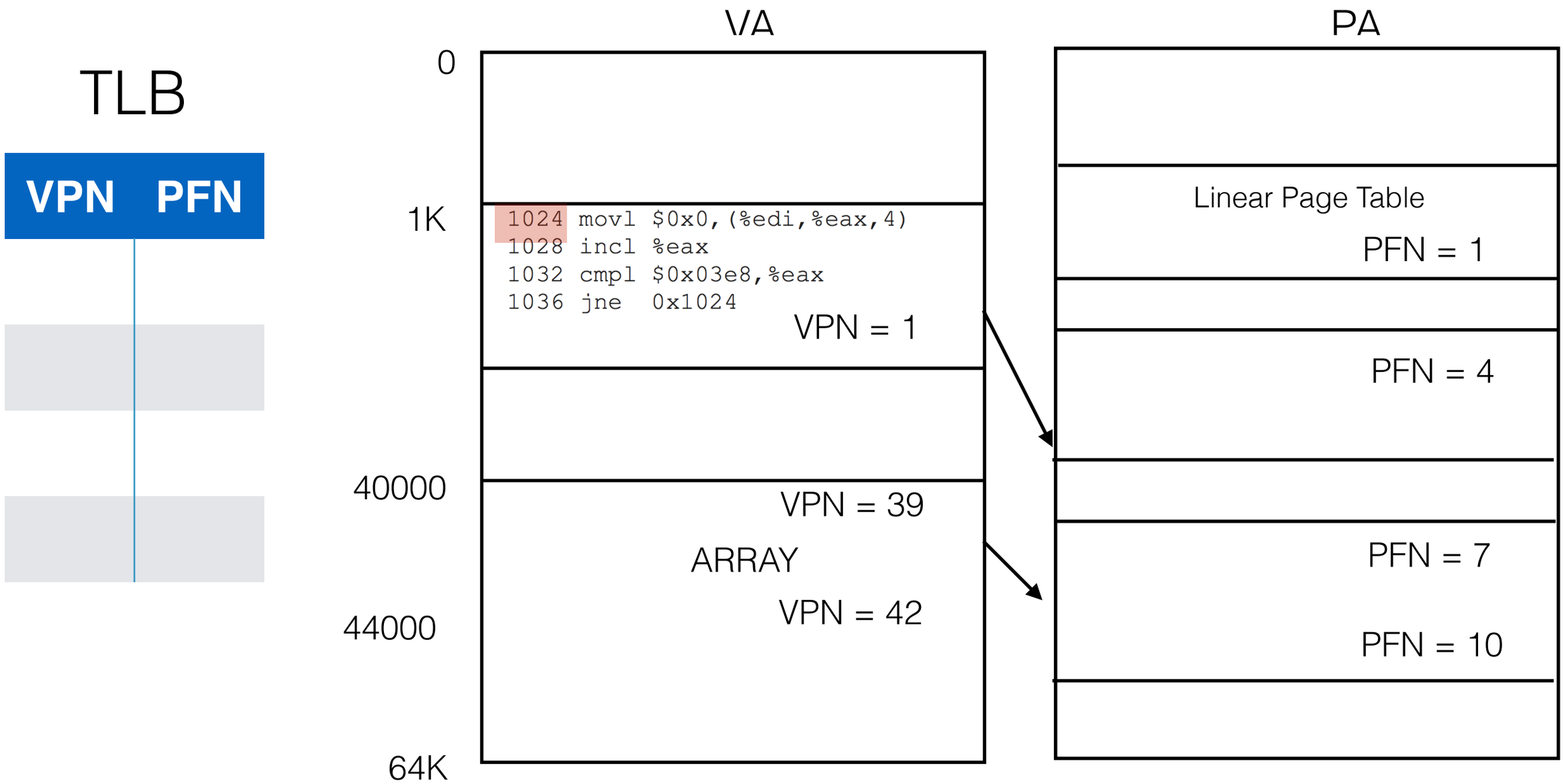

VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...
PFN = 4
...
PFN = 7
PFN = 10

Revision


VA
PA
0
1K
40000
44000
64K
1024 movl $0x0,(%edi,%eax,4)
1028 incl %eax
1032 cmpl $0x03e8,%eax
1036 jne 0x1024
VPN = 1
...
VPN = 39
ARRAY
VPN = 42
...
Linear Page Table
PFN = 1
...
PFN = 4
...
PFN = 7
PFN = 10

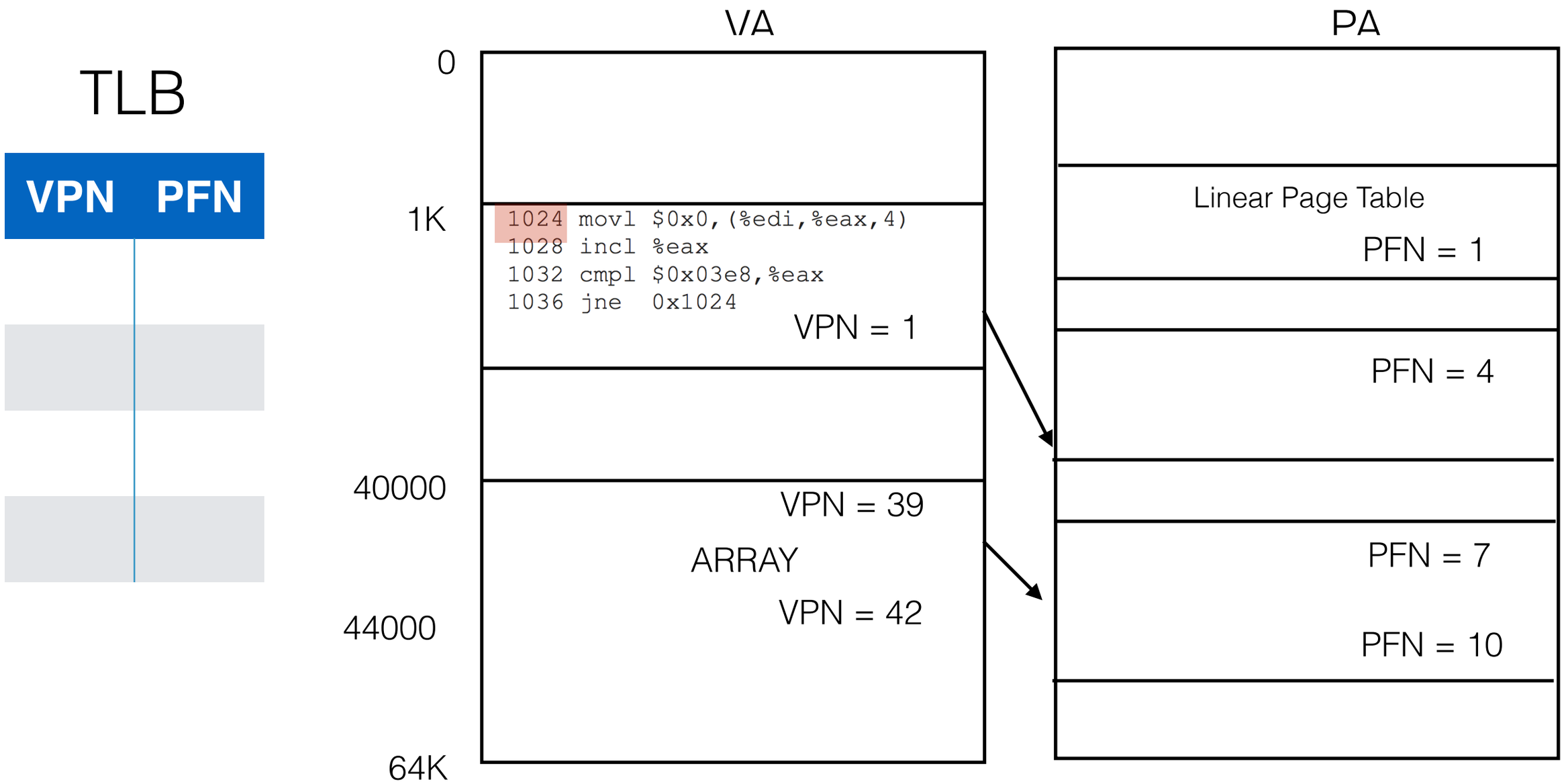
Worked Out Example

FETCH VA 1024



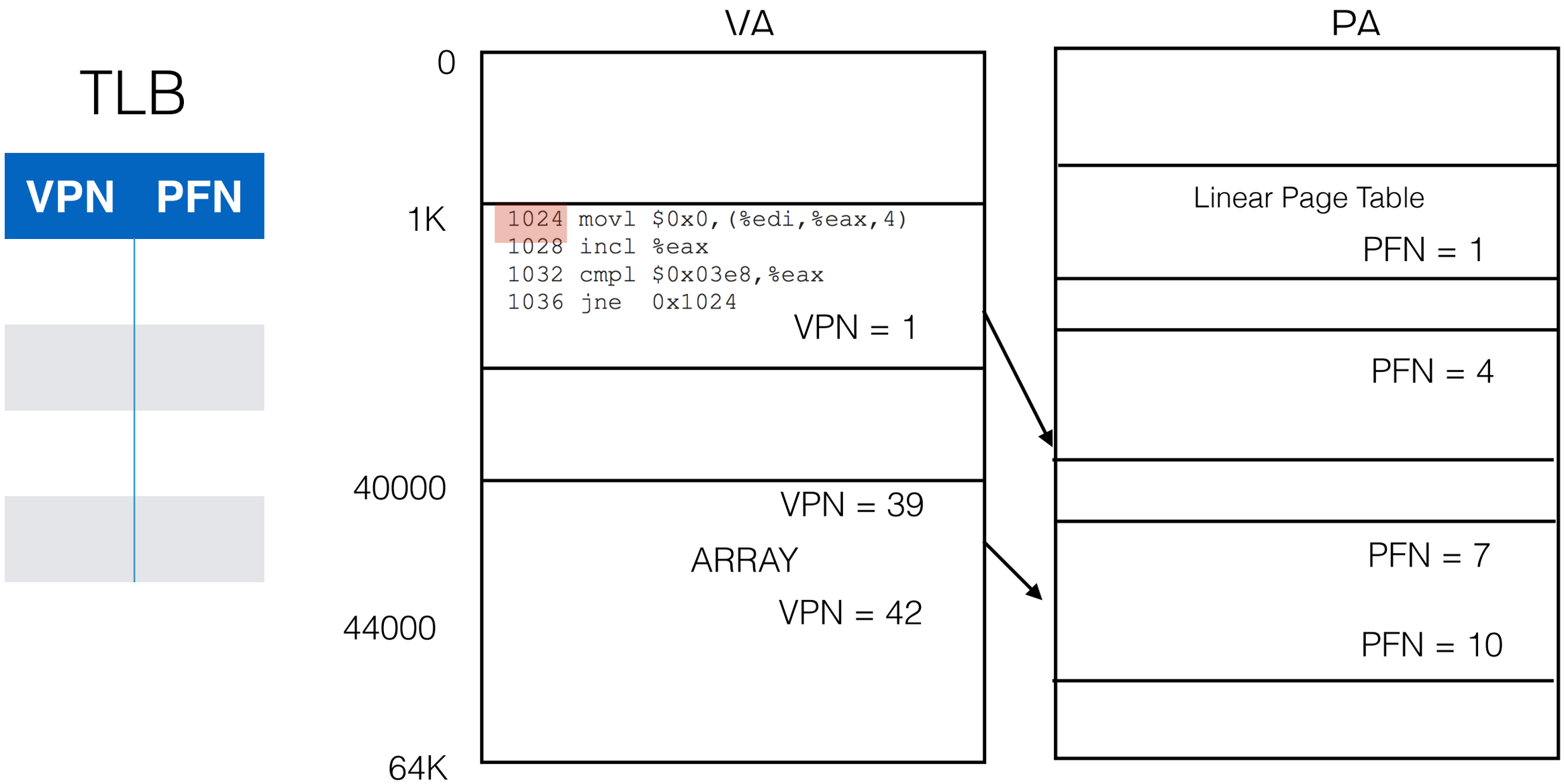
Worked Out Example

Get VPN for VA 1024. VPN = 1



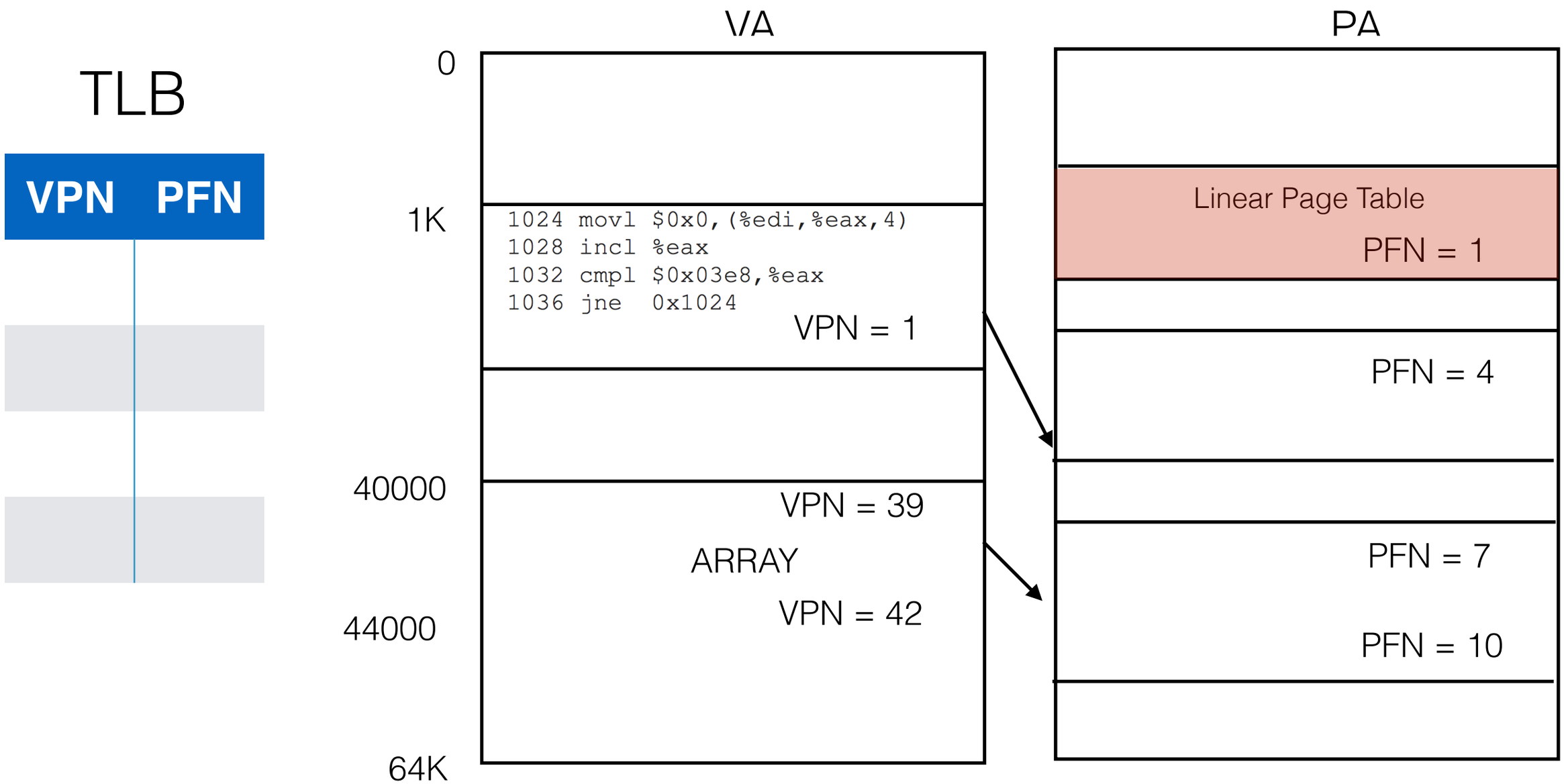
Worked Out Example

LOOK IN TLB for VPN = 1. Not found. TLB Miss!



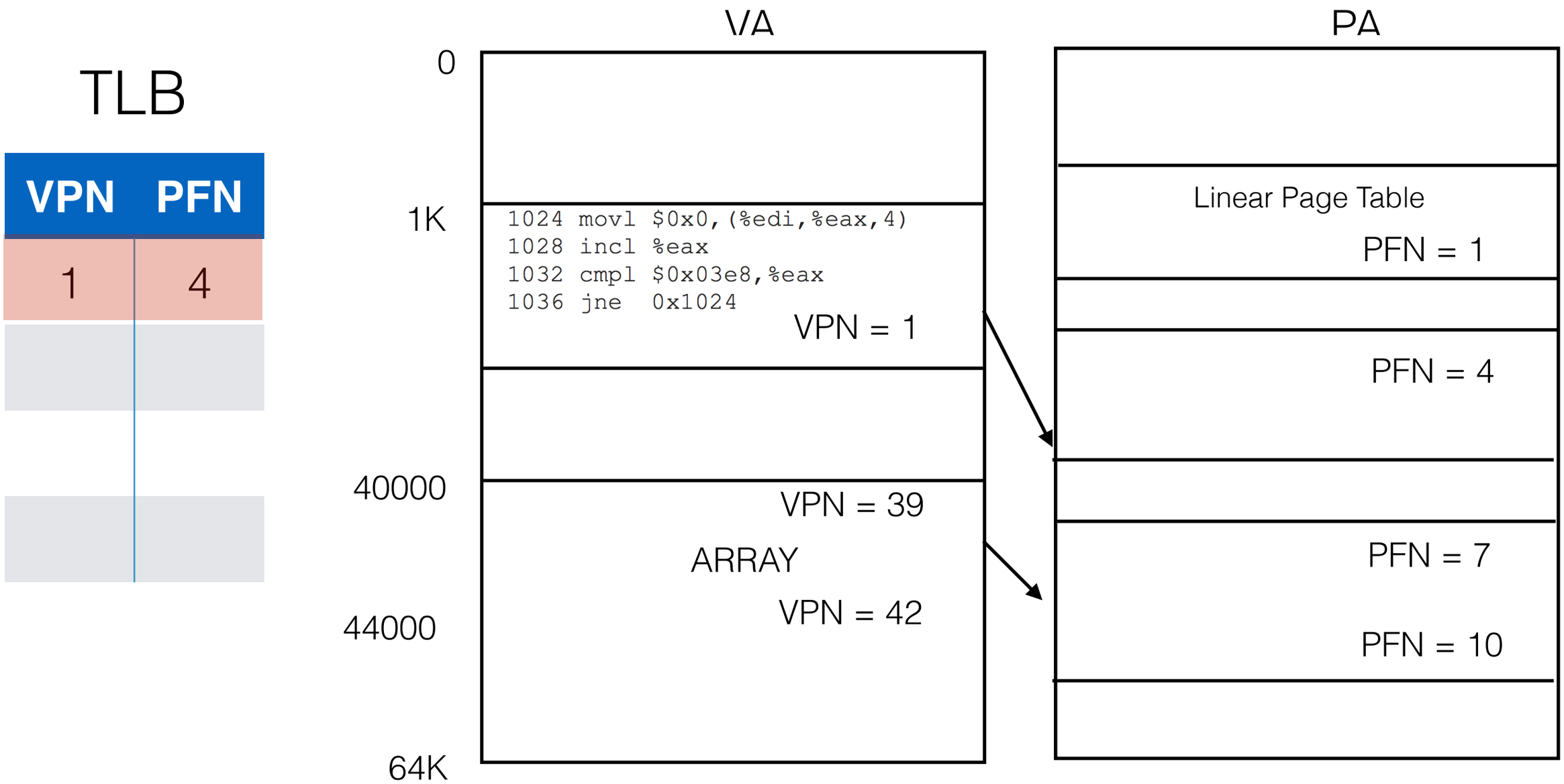
Worked Out Example

Find PFN for VPN = 1 by accessing Page Table



Worked Out Example

Add entry to TLB

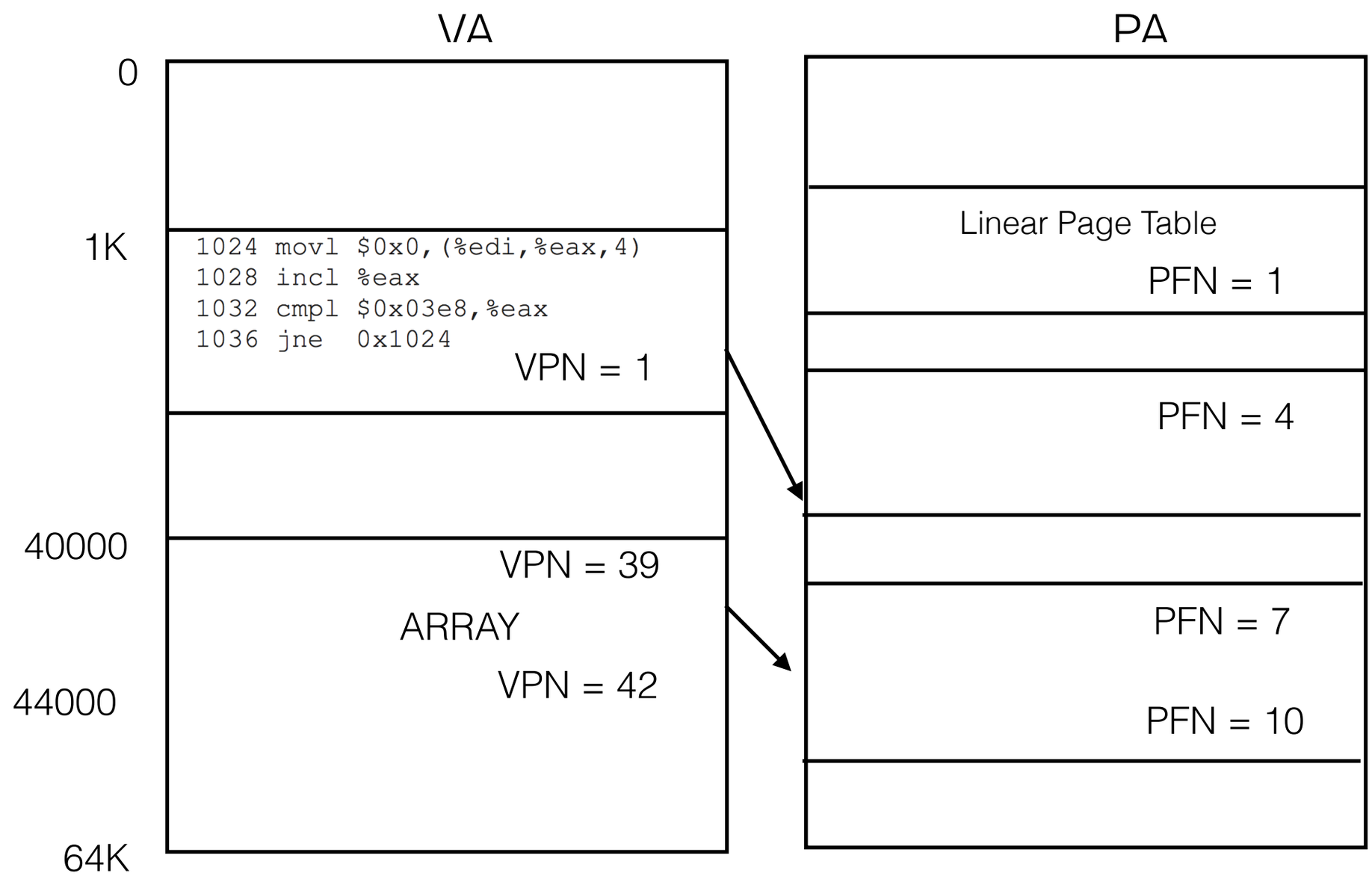


Worked Out Example

Search for translation of VPN = 1 on TLB

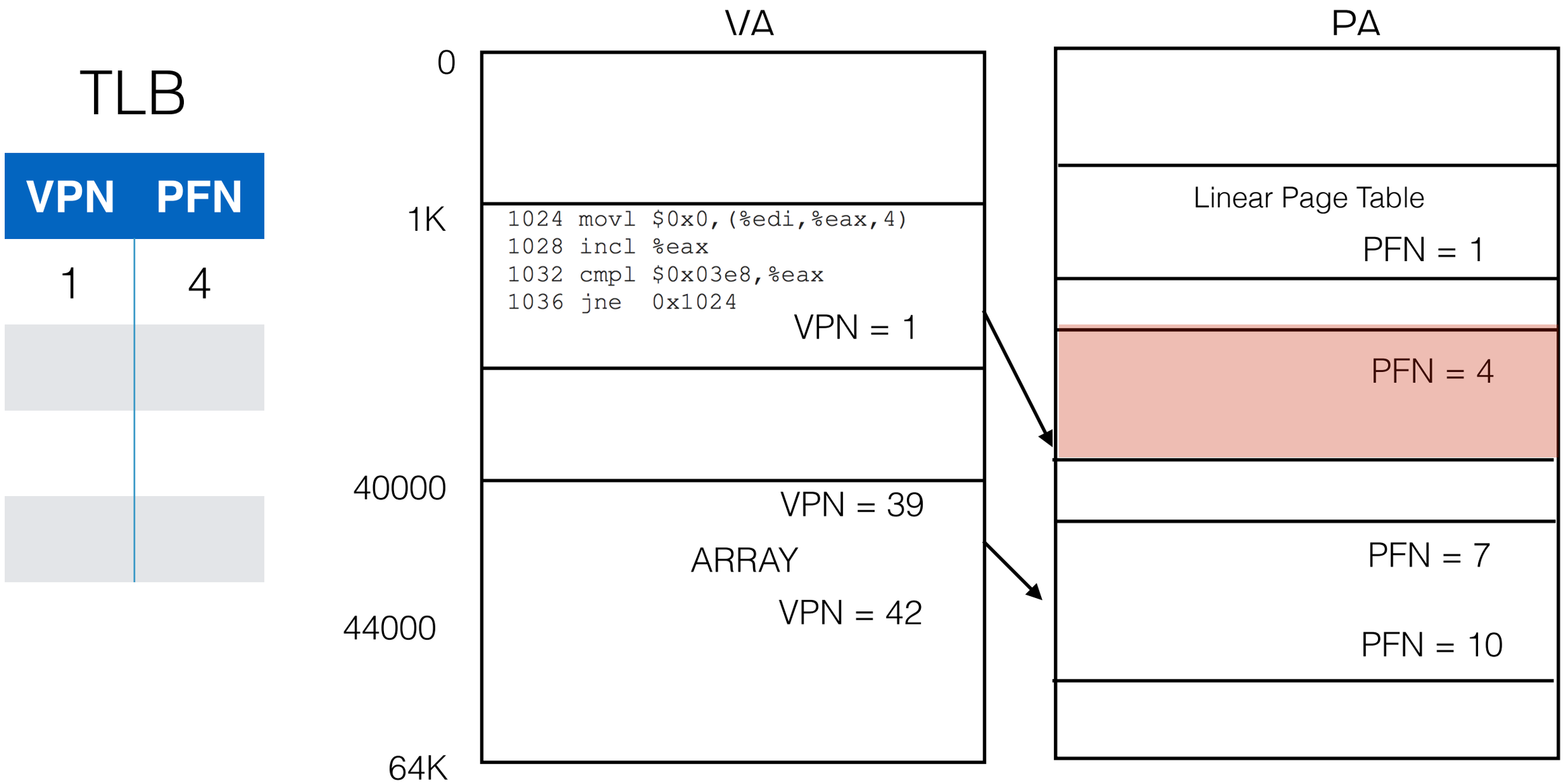
TLB

VPN	PFN
1	4



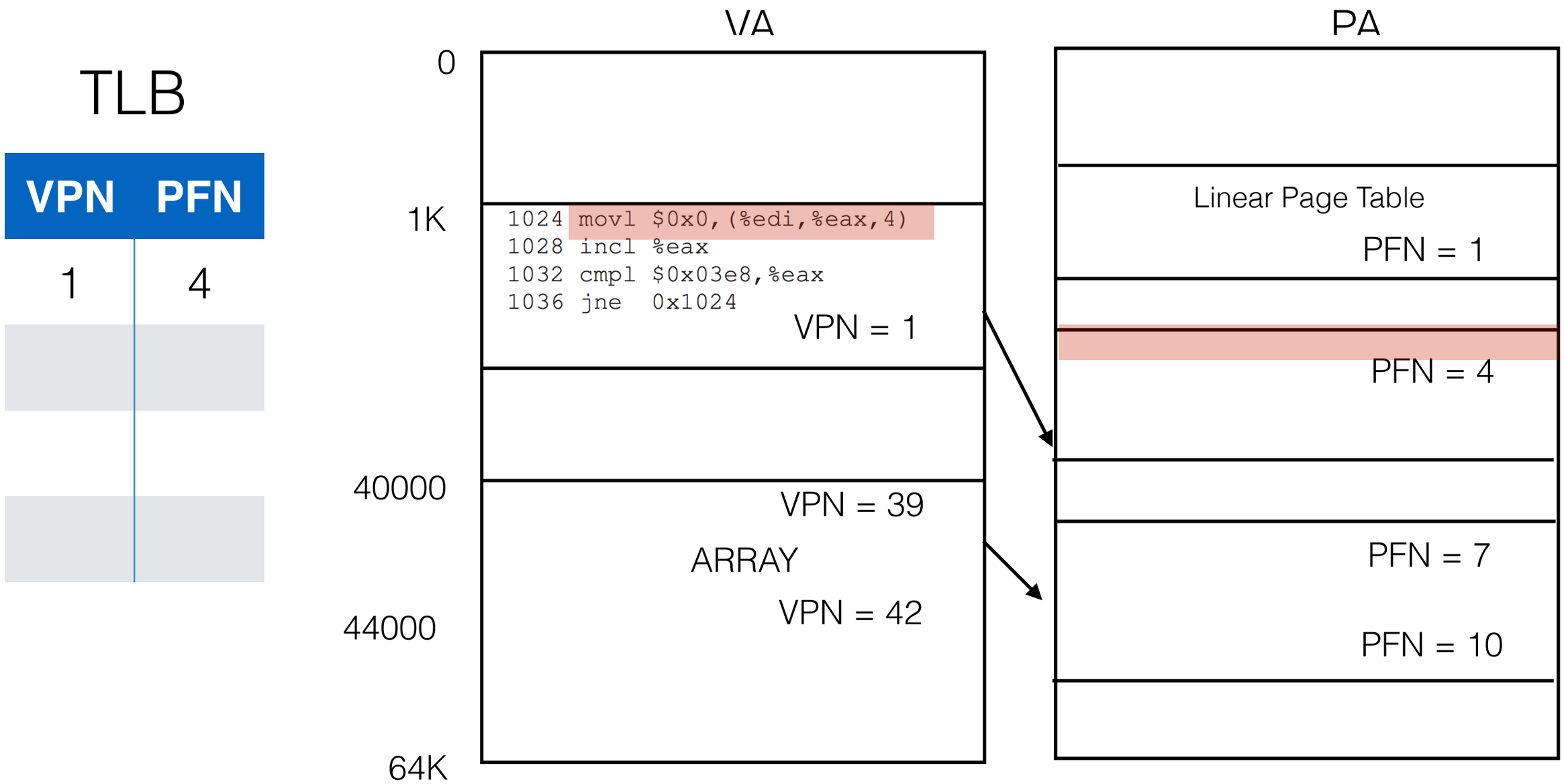
Worked Out Example

Goto PFN 4 and create PA by adding offset



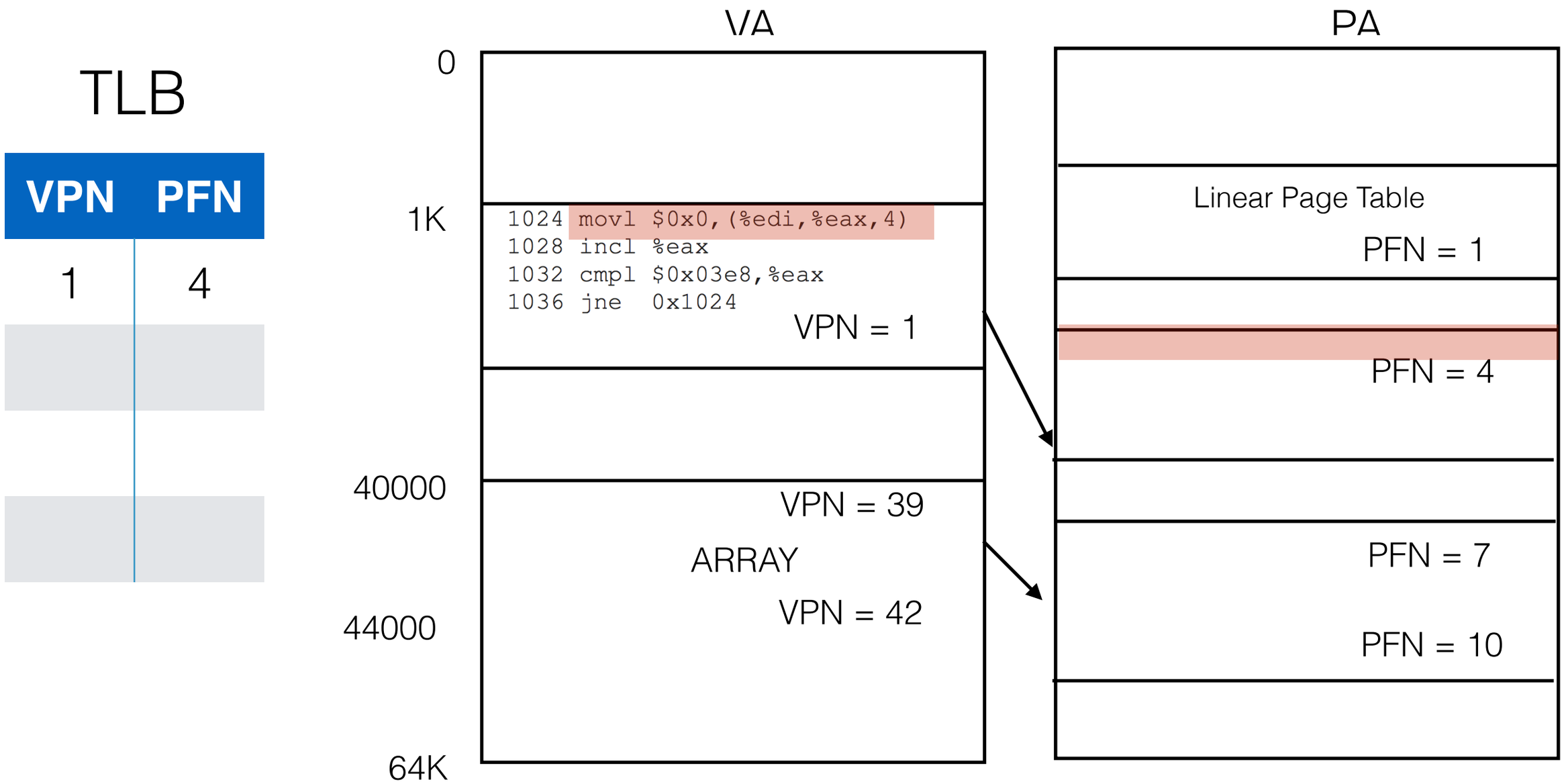
Worked Out Example

READ INSTRUCTION at PA(1024)



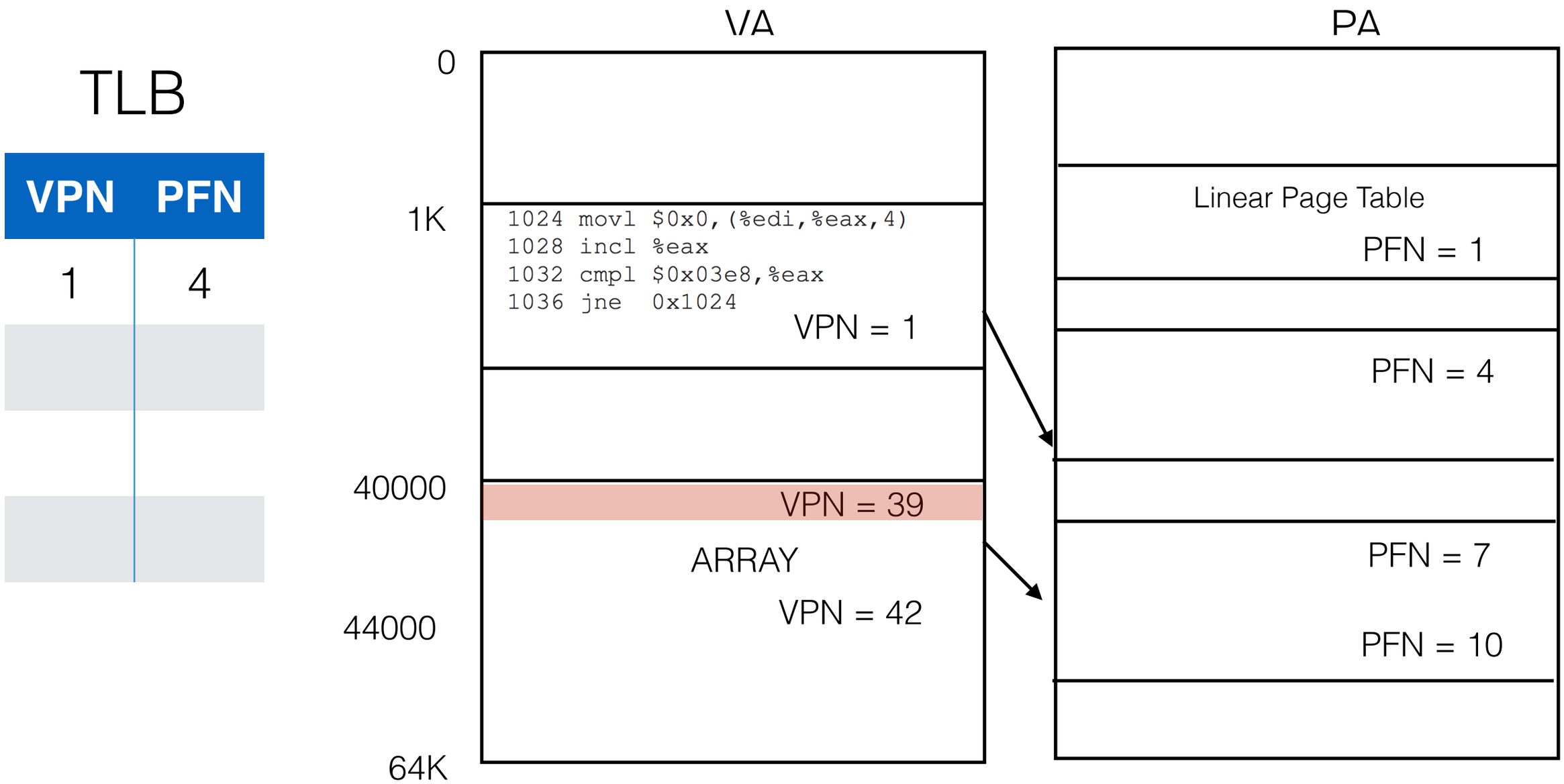
Worked Out Example

READ INSTRUCTION at PA(1024)



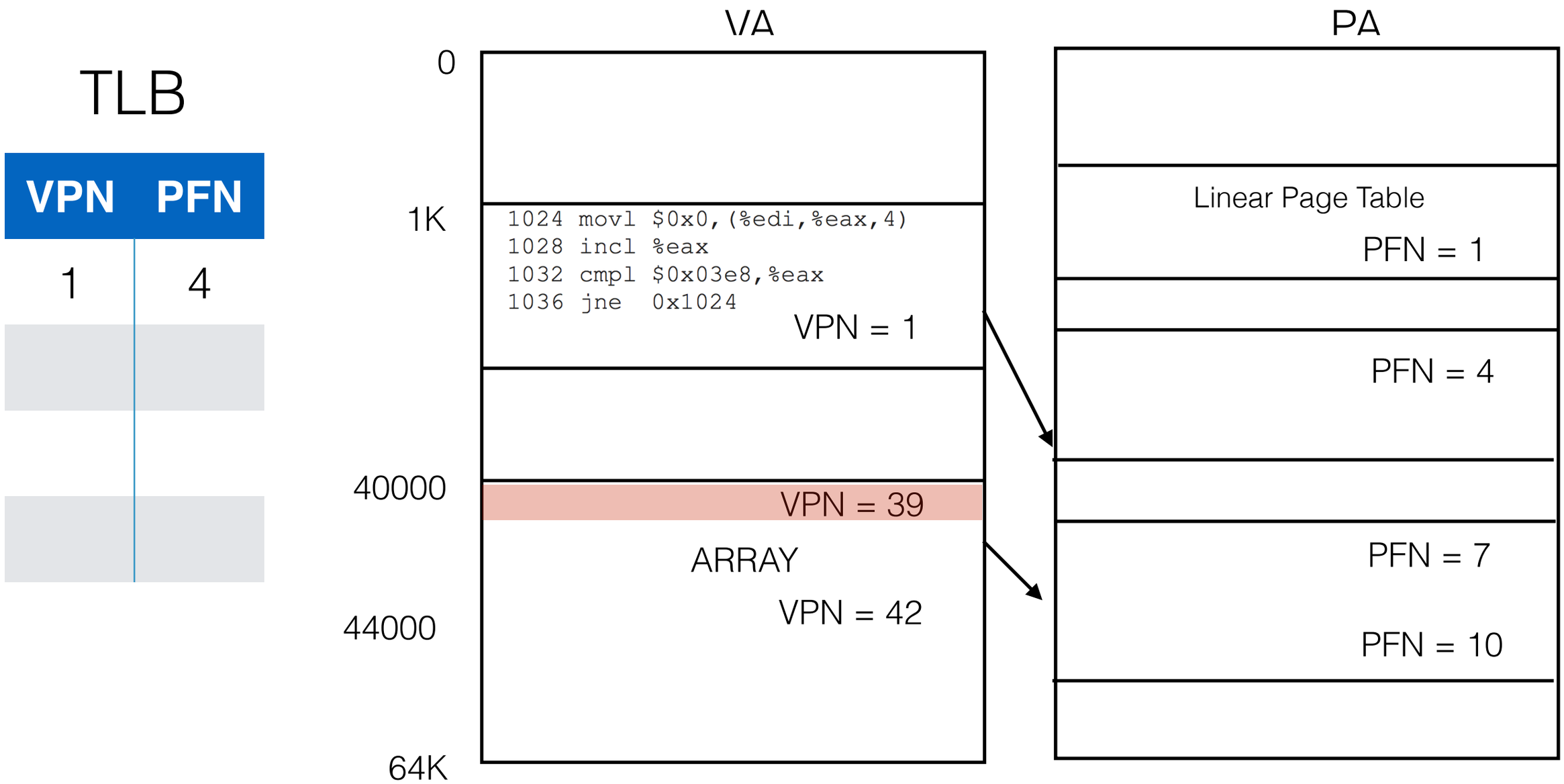
Worked Out Example

Find EDI + 4*EAX -> VA = 40000



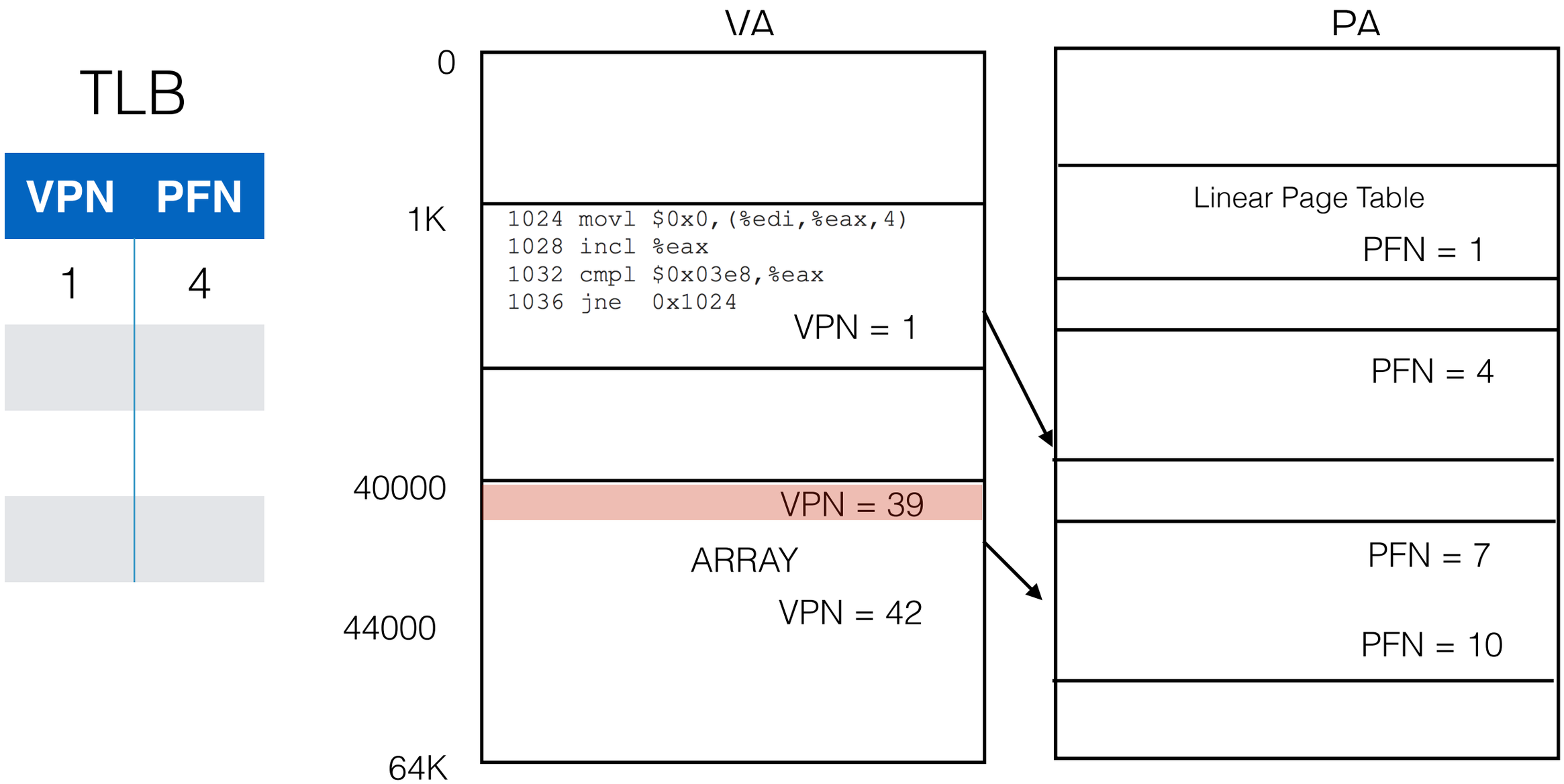
Worked Out Example

Find VPN for VA 40000. VPN = 39



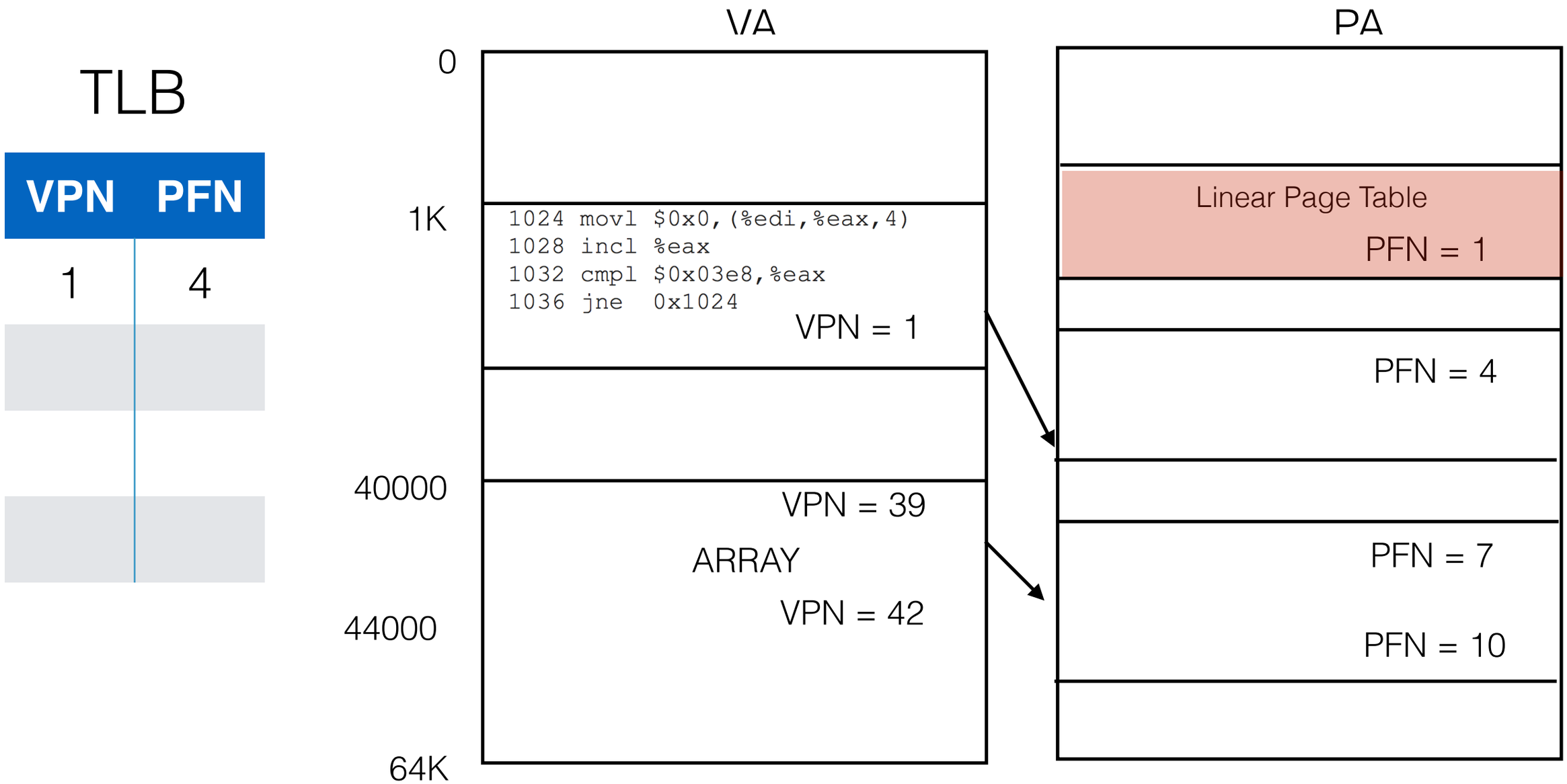
Worked Out Example

Check TLB for VPN = 39. Miss!



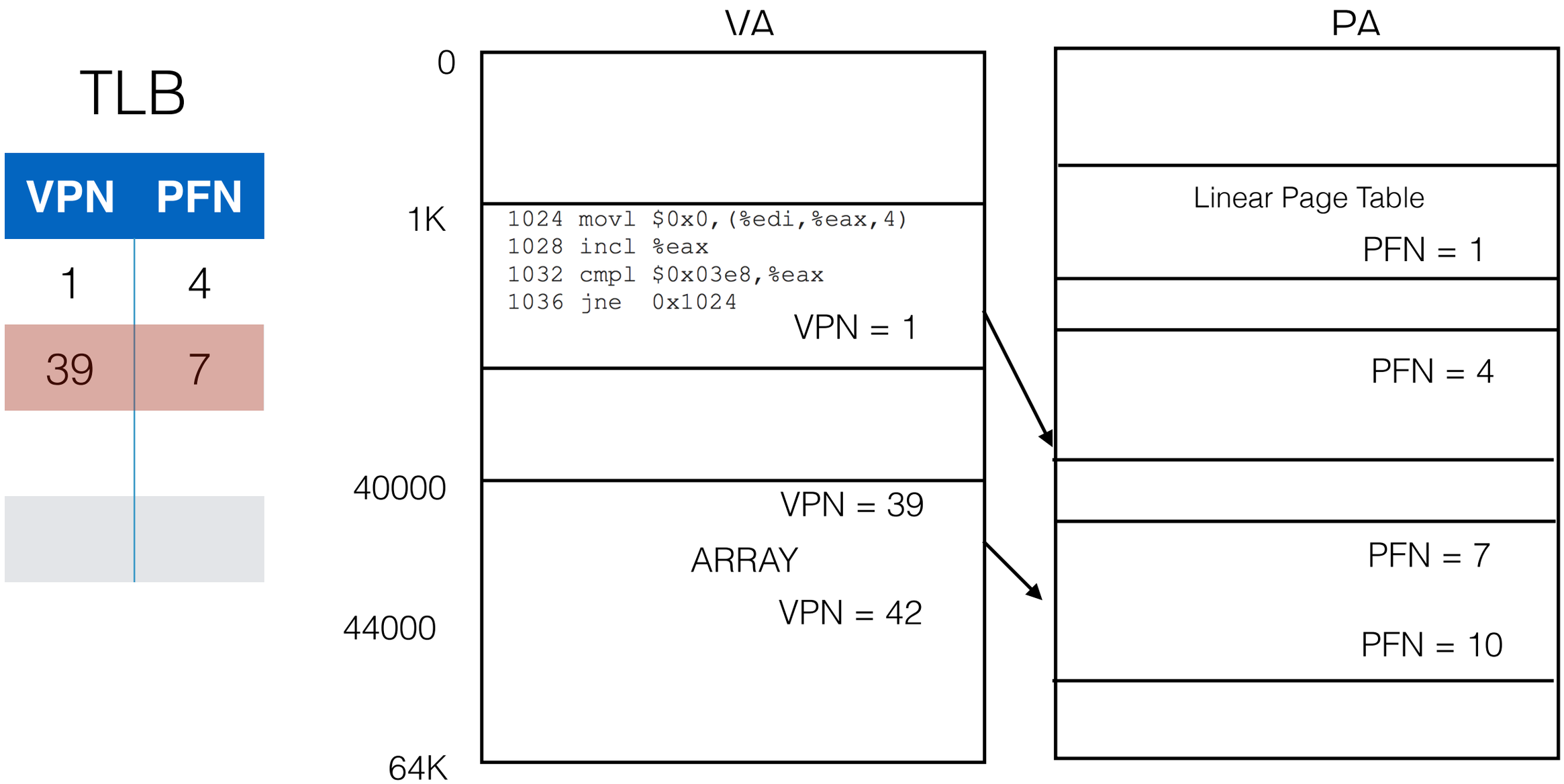
Worked Out Example

Get PFN for VPN = 39 from Page Table



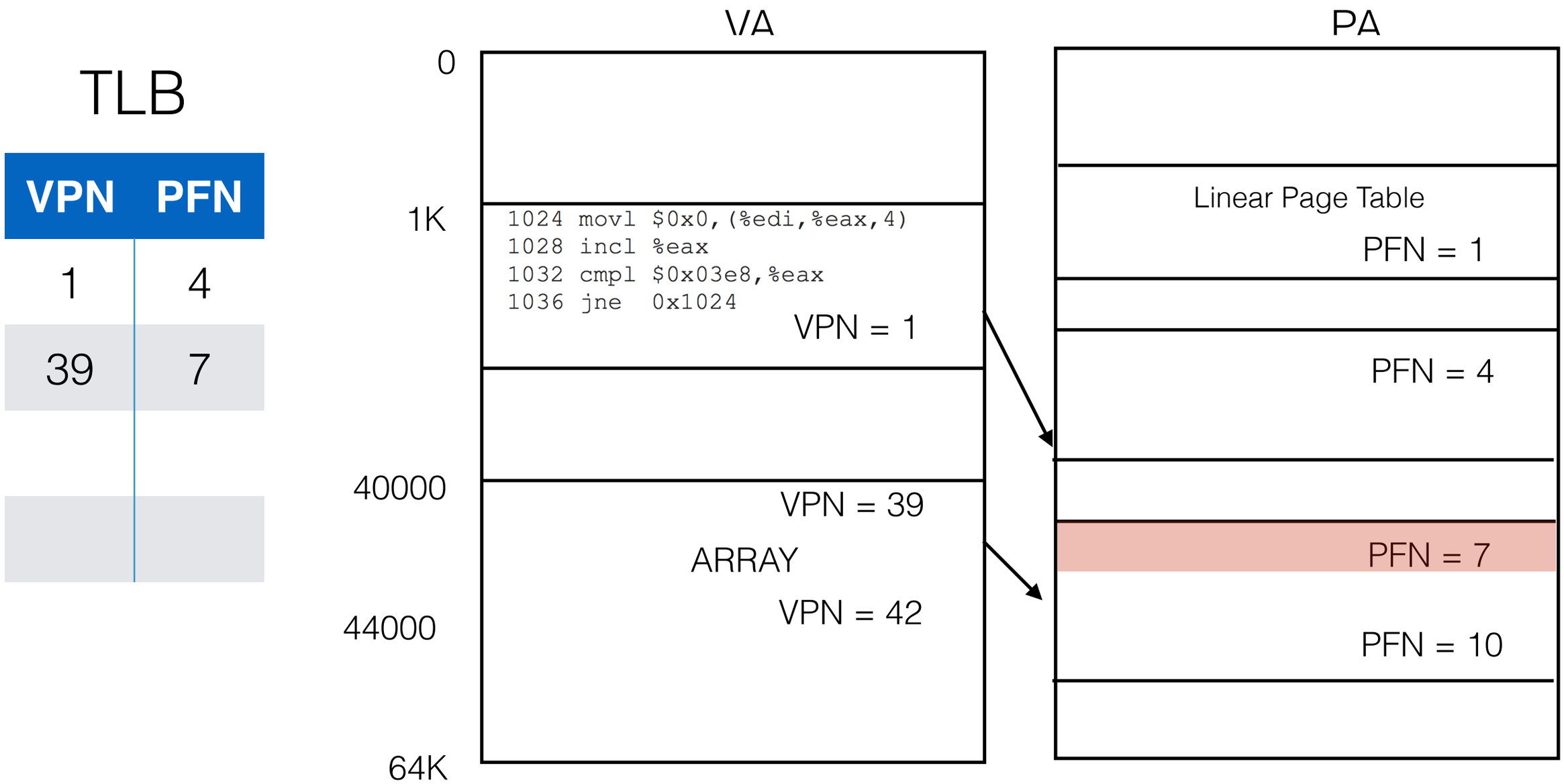
Worked Out Example

Store translation in TLB



Worked Out Example

Find PFN of VPN = 39 from TLB. Add offset to get PA.

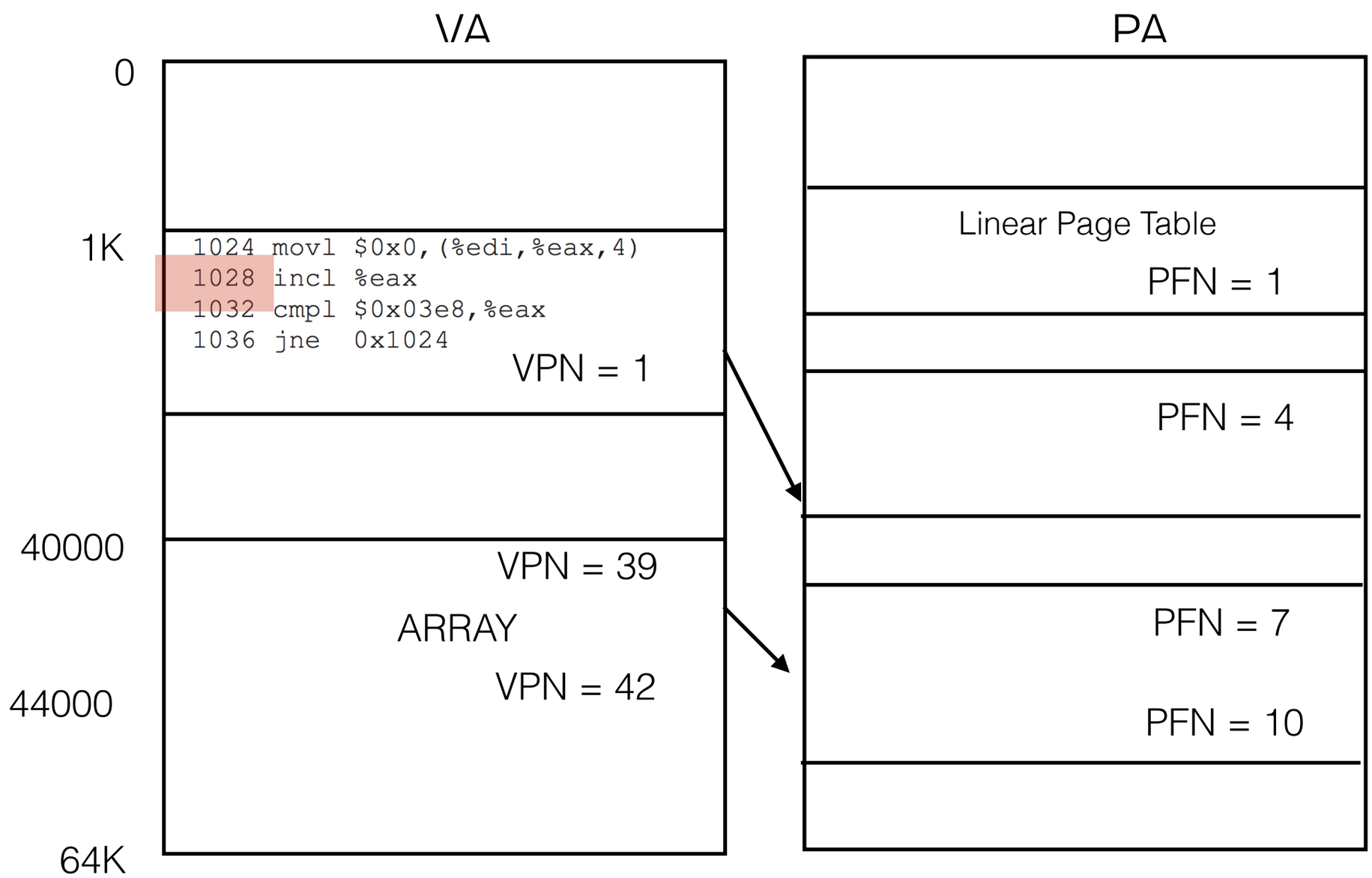


Worked Out Example

FIND PA FOR VA = 1028

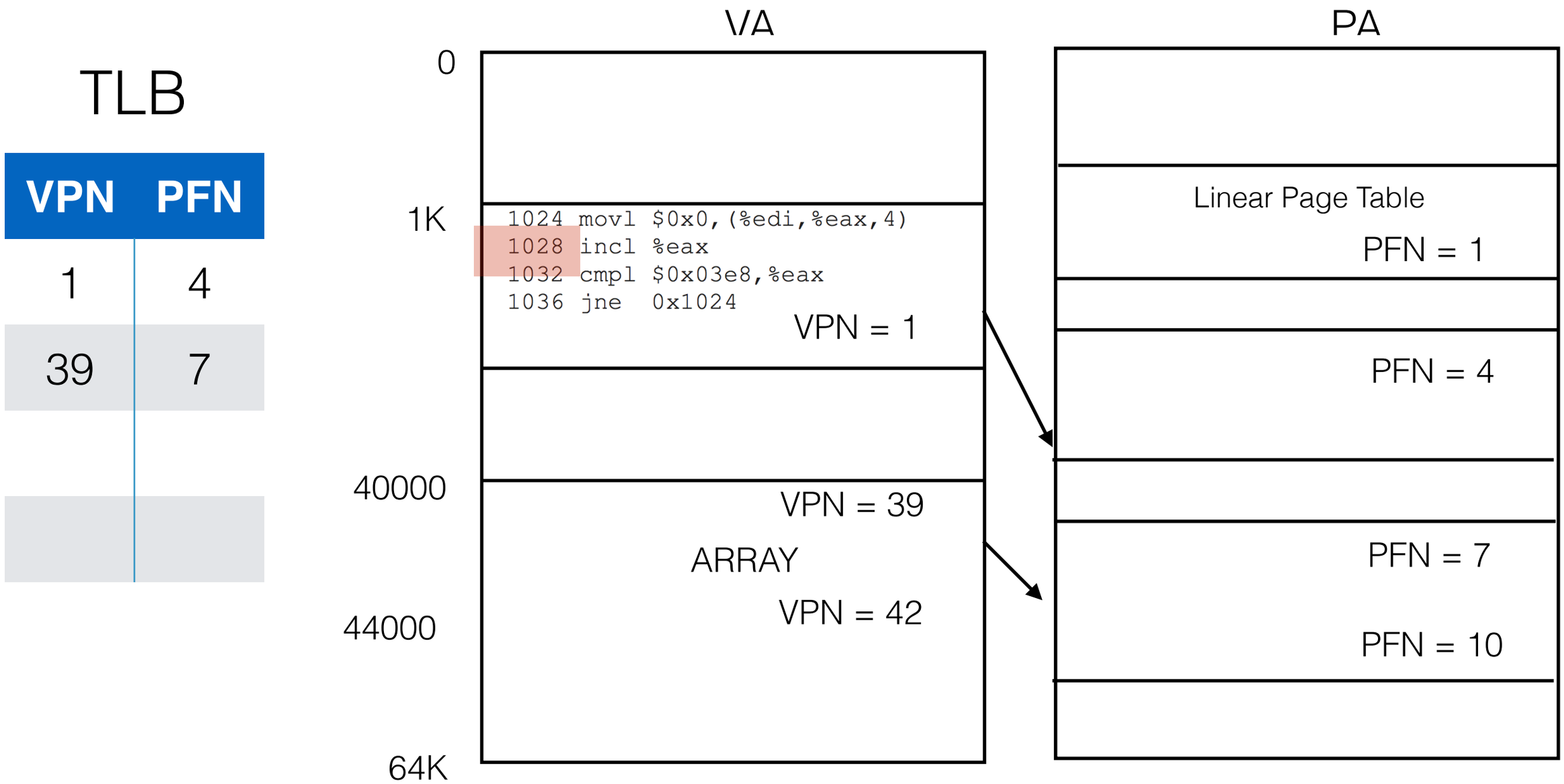
TLB

VPN	PFN
1	4
39	7



Worked Out Example

VPN = 1. Find Translation in TLB for VPN = 1. Found!

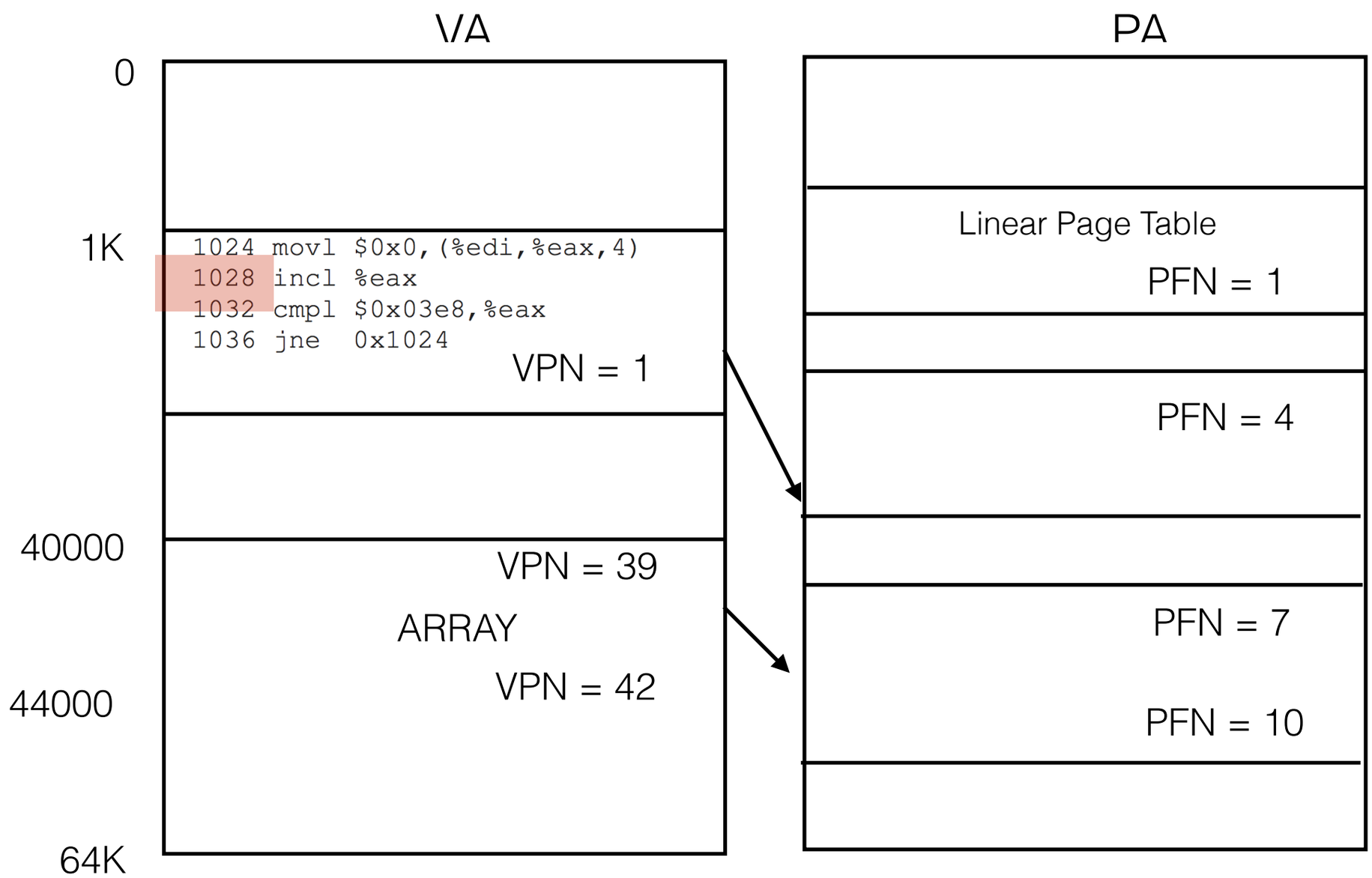


Worked Out Example

PFN = TLB[1] = 4

TLB

VPN	PFN
1	4
39	7

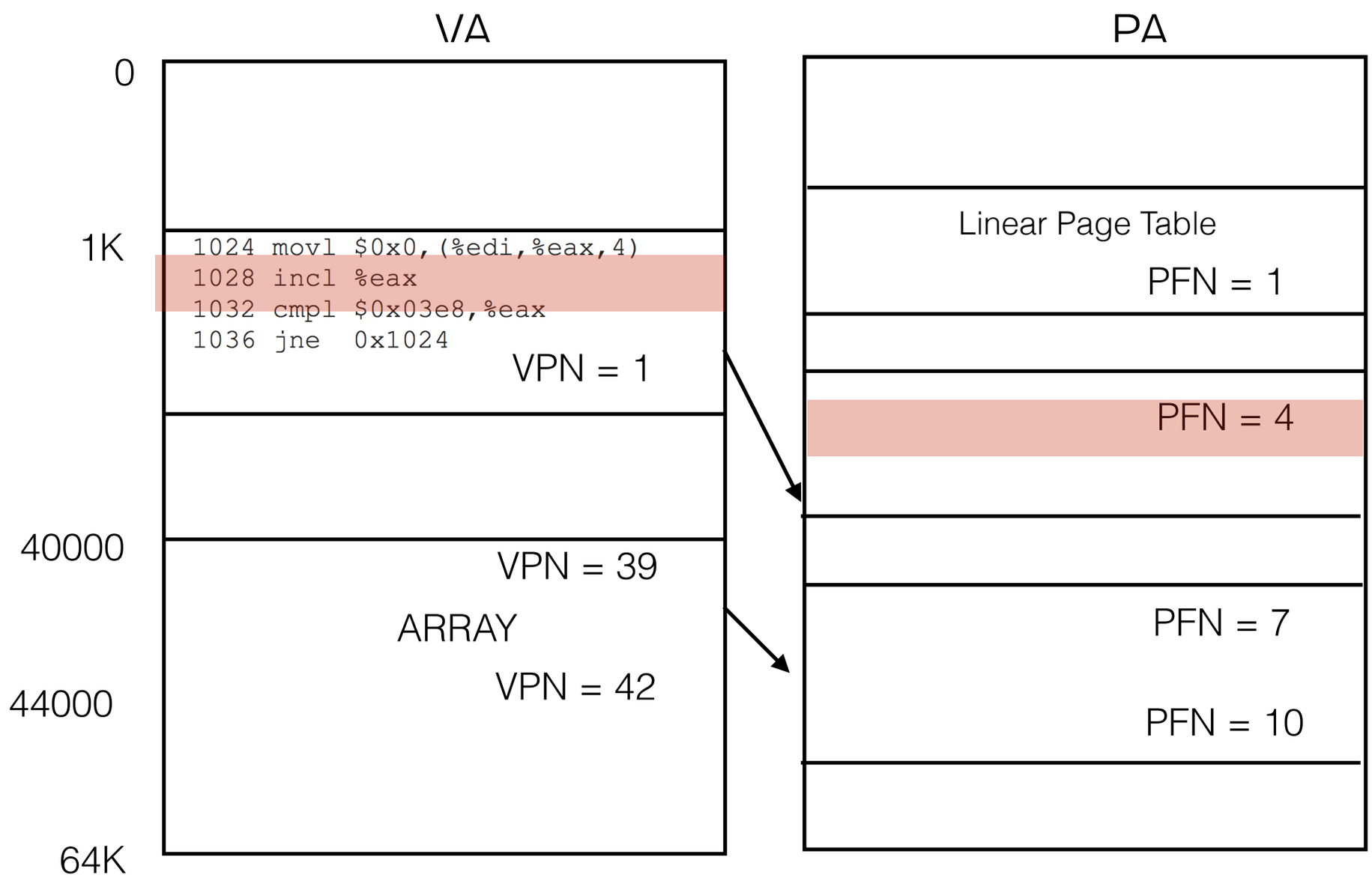


Worked Out Example

Get PA by adding offset to PFN = 4 and execute

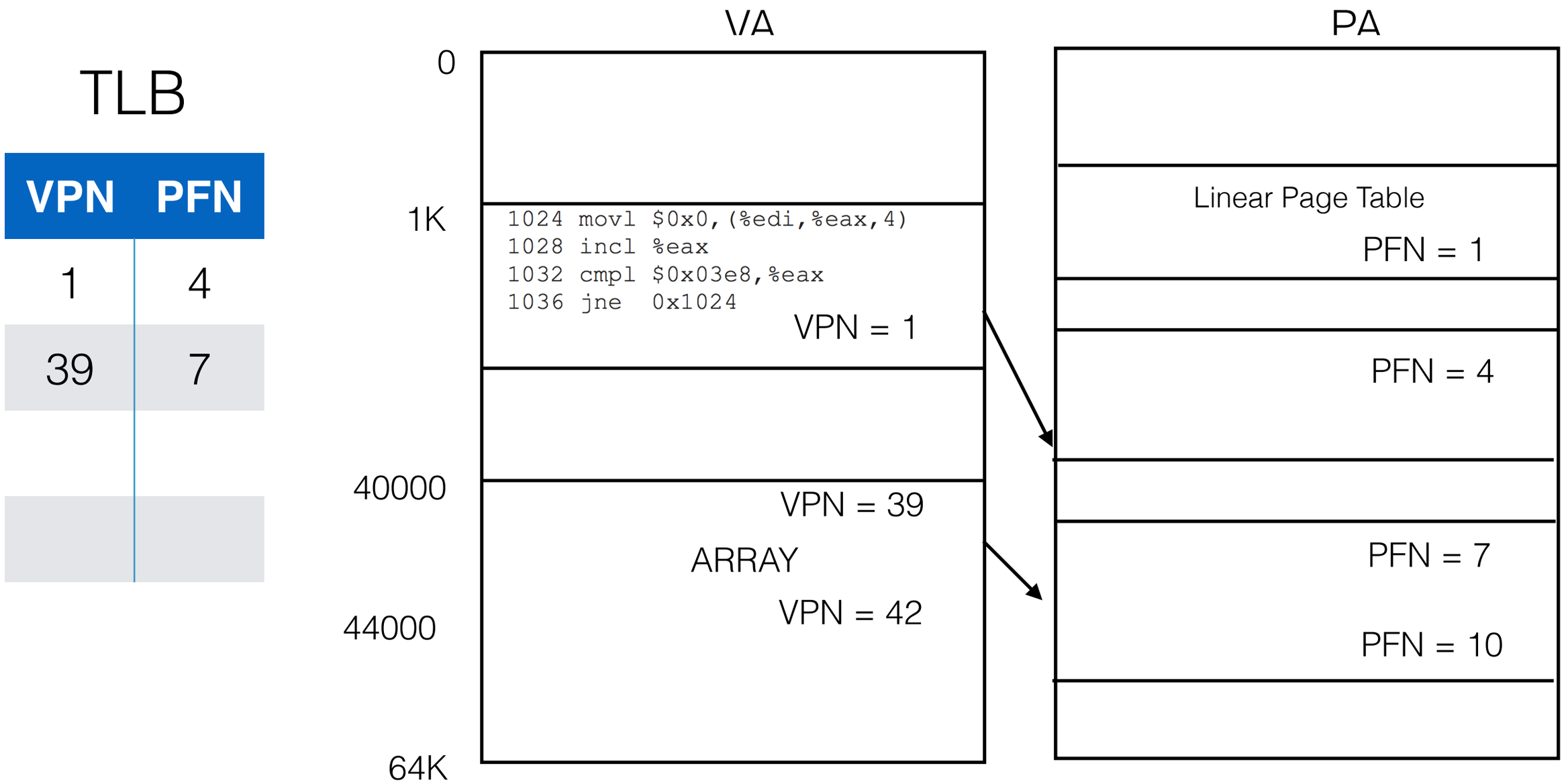
TLB

VPN	PFN
1	4
39	7



Worked Out Example

TLB miss for VPN = 40…



Spatial and Temporal Locality

Spatial and Temporal Locality

1. Hit rate = TLB Hit/(TLB Hit + TLB Miss)

Spatial and Temporal Locality

1. Hit rate = TLB Hit/(TLB Hit + TLB Miss)
2. Spatial locality -> TLB has good hit rate

Spatial and Temporal Locality

1. Hit rate = TLB Hit/(TLB Hit + TLB Miss)
2. Spatial locality -> TLB has good hit rate
    1. Arrays elements are spatially close (EDX + 4*EAX)

Spatial and Temporal Locality

1. Hit rate = TLB Hit/(TLB Hit + TLB Miss)
2. Spatial locality -> TLB has good hit rate
    1. Arrays elements are spatially close (EDX + 4*EAX)
    2. Instructions are spatially close (1024, …)

Spatial and Temporal Locality

1. Hit rate = TLB Hit/(TLB Hit + TLB Miss)
2. Spatial locality -> TLB has good hit rate
    1. Arrays elements are spatially close (EDX + 4*EAX)
    2. Instructions are spatially close (1024, …)
3. Temporal locality -> TLB has a good hit rate

Spatial and Temporal Locality

1. Hit rate = TLB Hit/(TLB Hit + TLB Miss)
2. Spatial locality -> TLB has good hit rate
    1. Arrays elements are spatially close (EDX + 4*EAX)
    2. Instructions are spatially close (1024, …)
3. Temporal locality -> TLB has a good hit rate
    1. Loop. Re-using same instructions which exist in TLB

Memory Cycle Rate Example

Memory Cycle Rate Example

Memory Cycle Rate Example

1. Hit = 1 clock cycle

Memory Cycle Rate Example

1. Hit = 1 clock cycle
2. Miss = 30 clock cycles

Memory Cycle Rate Example

1. Hit = 1 clock cycle
2. Miss = 30 clock cycles
3. Miss rate = 1%

Memory Cycle Rate Example

1. Hit = 1 clock cycle
2. Miss = 30 clock cycles
3. Miss rate = 1%
4. Cycle rate = .99*1 + .01*(30 + 1) =1.3 cycles

Context Switch

TLB

VPN	PFN
1	4
39	7

Context Switch

TLB

P1 running

VPN	PFN
1	4
39	7

Context Switch

TLB

P1 running

VPN	PFN
1	4
39	7

Context Switch

TLB

P1 running

VPN	PFN
1	4
39	7

P2 running

Context Switch

TLB

P1 running

VPN	PFN
1	4
39	7
…	…
1	30

P2 running

Context Switch

TLB

P1 running

VPN	PFN
1	4
39	7
…	…
1	30

P2 running

What will VPN 1 be mapped to?

Context Switch

TLB

P1 running

VPN	PFN
1	4
39	7
…	…
1	30

P2 running

Context Switch

TLB

P1 running

VPN	PFN
1	4
39	7
…	…
1	30

P2 running

What will VPN 1 be mapped to?

Context Switch

TLB

P1 running

VPN	PFN	VALID	PERMISSION	ASID
1	4	1	R	1
39	7	1	R	1
…	…	0	RW	
1	30	1	RWX	2

P2 running

TLB Entry Replacement

TLB Entry Replacement

TLB Entry Replacement

1. Why to replace TLB entries?

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?
    1. Newer translation found

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?
    1. Newer translation found
    2. Context switch

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?
    1. Newer translation found
    2. Context switch
3. How to replace?

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?
    1. Newer translation found
    2. Context switch
3. How to replace?
    1. Remove at random

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?
    1. Newer translation found
    2. Context switch
3. How to replace?
    1. Remove at random
    2. Remove least recently used (LRU)

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?
    1. Newer translation found
    2. Context switch
3. How to replace?
    1. Remove at random
    2. Remove least recently used (LRU)
        1. Corner case:

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?
    1. Newer translation found
    2. Context switch
3. How to replace?
    1. Remove at random
    2. Remove least recently used (LRU)
        1. Corner case:
            1. TLB size = N

TLB Entry Replacement

1. Why to replace TLB entries?
    1. Limited space
2. When to replace?
    1. Newer translation found
    2. Context switch
3. How to replace?
    1. Remove at random
    2. Remove least recently used (LRU)
        1. Corner case:
            1. TLB size = N
            2. N+1 page accesses in loop

Memory Virtualisation Thus Far

Memory Virtualisation Thus Far

Memory Virtualisation Thus Far

- Base & Bounds

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation
  - Pros: Still relatively simple, 3 registers, lesser fragmentation

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation
  - Pros: Still relatively simple, 3 registers, lesser fragmentation
  - Cons: Still contiguous block of memory for segment

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation
  - Pros: Still relatively simple, 3 registers, lesser fragmentation
  - Cons: Still contiguous block of memory for segment
- Paging

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation
  - Pros: Still relatively simple, 3 registers, lesser fragmentation
  - Cons: Still contiguous block of memory for segment
- Paging
  - Pros: Very low chances of segmentation

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation
  - Pros: Still relatively simple, 3 registers, lesser fragmentation
  - Cons: Still contiguous block of memory for segment
- Paging
  - Pros: Very low chances of segmentation
  - Cons: Slow, lots of memory accesses; memory overhead/ process is huge!

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation
  - Pros: Still relatively simple, 3 registers, lesser fragmentation
  - Cons: Still contiguous block of memory for segment
- Paging
  - Pros: Very low chances of segmentation
  - Cons: Slow, lots of memory accesses; memory overhead/ process is huge!
- Paging + TLB

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation
  - Pros: Still relatively simple, 3 registers, lesser fragmentation
  - Cons: Still contiguous block of memory for segment
- Paging
  - Pros: Very low chances of segmentation
  - Cons: Slow, lots of memory accesses; memory overhead/ process is huge!
- Paging + TLB
  - Pros: Improves the address translation speed (spatial & temporal locality)

Memory Virtualisation Thus Far

- Base & Bounds
  - Pros: Very quick, 2 registers
  - Cons: Contiguous block of memory -> fragmentation
- Segmentation
  - Pros: Still relatively simple, 3 registers, lesser fragmentation
  - Cons: Still contiguous block of memory for segment
- Paging
  - Pros: Very low chances of segmentation
  - Cons: Slow, lots of memory accesses; memory overhead/ process is huge!
- Paging + TLB
  - Pros: Improves the address translation speed (spatial & temporal locality)
  - Cons: Limited in size, memory overhead/process still huge

Reducing Memory Overheads of Paging

Reducing Memory Overheads of Paging

- 32 bit address space with 4 KB pages

Reducing Memory Overheads of Paging

- 4 KB pages ->12 bits?

Reducing Memory Overheads of Paging

-
-
- Remaining bits = 32 - 12 = 20

Reducing Memory Overheads of Paging

- 20 bit VPN

Reducing Memory Overheads of Paging

- # pages = 2^20

Reducing Memory Overheads of Paging

-
-
-
  -
  -
- 4 bytes per translation -> 4 * 2^20 MB = **4 MB/process**

Reducing Memory Overheads of Paging

Reducing Memory Overheads of Paging

Solution 0: Decrease the size of VA space

Reducing Memory Overheads of Paging

Solution 0: Decrease the size of VA space

- 12 bit offset for 4 K pages

Reducing Memory Overheads of Paging

Solution 0: Decrease the size of VA space

-
- 30 bit address space

Reducing Memory Overheads of Paging

Solution 0: Decrease the size of VA space

-
-
- 18 bit VPNs

Reducing Memory Overheads of Paging

Solution 0: Decrease the size of VA space

-
-
-
- 4 bytes per translation -> 4 * 2^18 MB = **1 MB/process**

Reducing Memory Overheads of Paging

Solution 0: Decrease the size of VA space

-
-
-
- 4 bytes per translation -> 4 * 2^18 MB = **1 MB/process**

  - 32 bit address space —> 4 GB

Reducing Memory Overheads of Paging

Solution 0: Decrease the size of VA space

-
-
-
- 4 bytes per translation -> 4 * 2^18 MB = **1 MB/process**
-
- 30 bit address space —> 1 GB

Reducing Memory Overheads of Paging

Reducing Memory Overheads of Paging

Solution 1: Increase the page size

Reducing Memory Overheads of Paging

Solution 1: Increase the page size

- 32 bit address space with 16 KB pages

Reducing Memory Overheads of Paging

Solution 1: Increase the page size

- 16 KB pages ->14 bits?

Reducing Memory Overheads of Paging

Solution 1: Increase the page size

-
-
- Remaining bits = 32 - 14 = 18

Reducing Memory Overheads of Paging

Solution 1: Increase the page size

- 18 bit VPN

Reducing Memory Overheads of Paging

Solution 1: Increase the page size

- # pages = 2^18

Reducing Memory Overheads of Paging

Solution 1: Increase the page size

-
-
-
  -
  -
- 4 bytes per translation -> 4 * 2^18 MB = **1 MB/process**

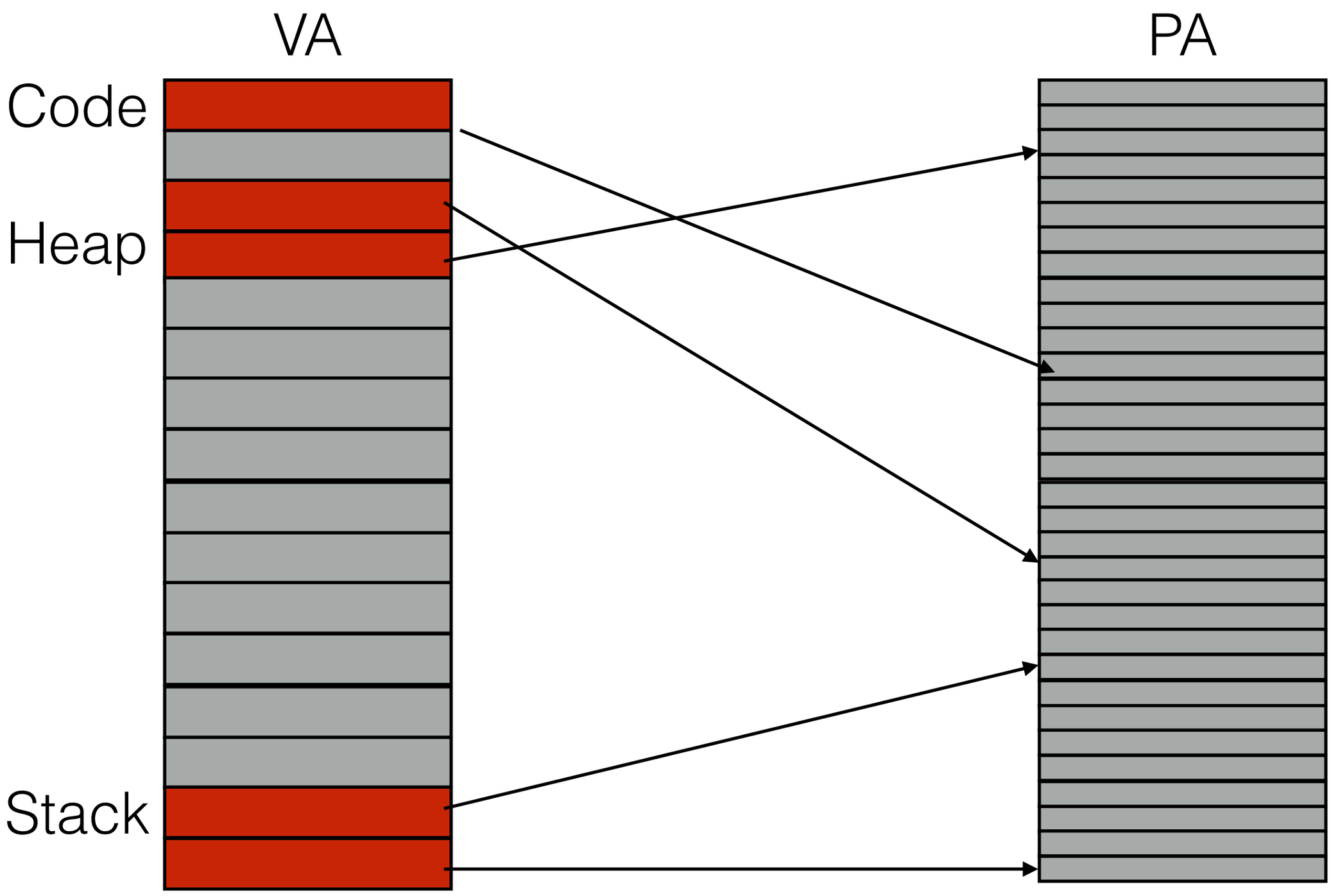
Reducing Memory Overheads of Paging

Solution 1: Increase the page size

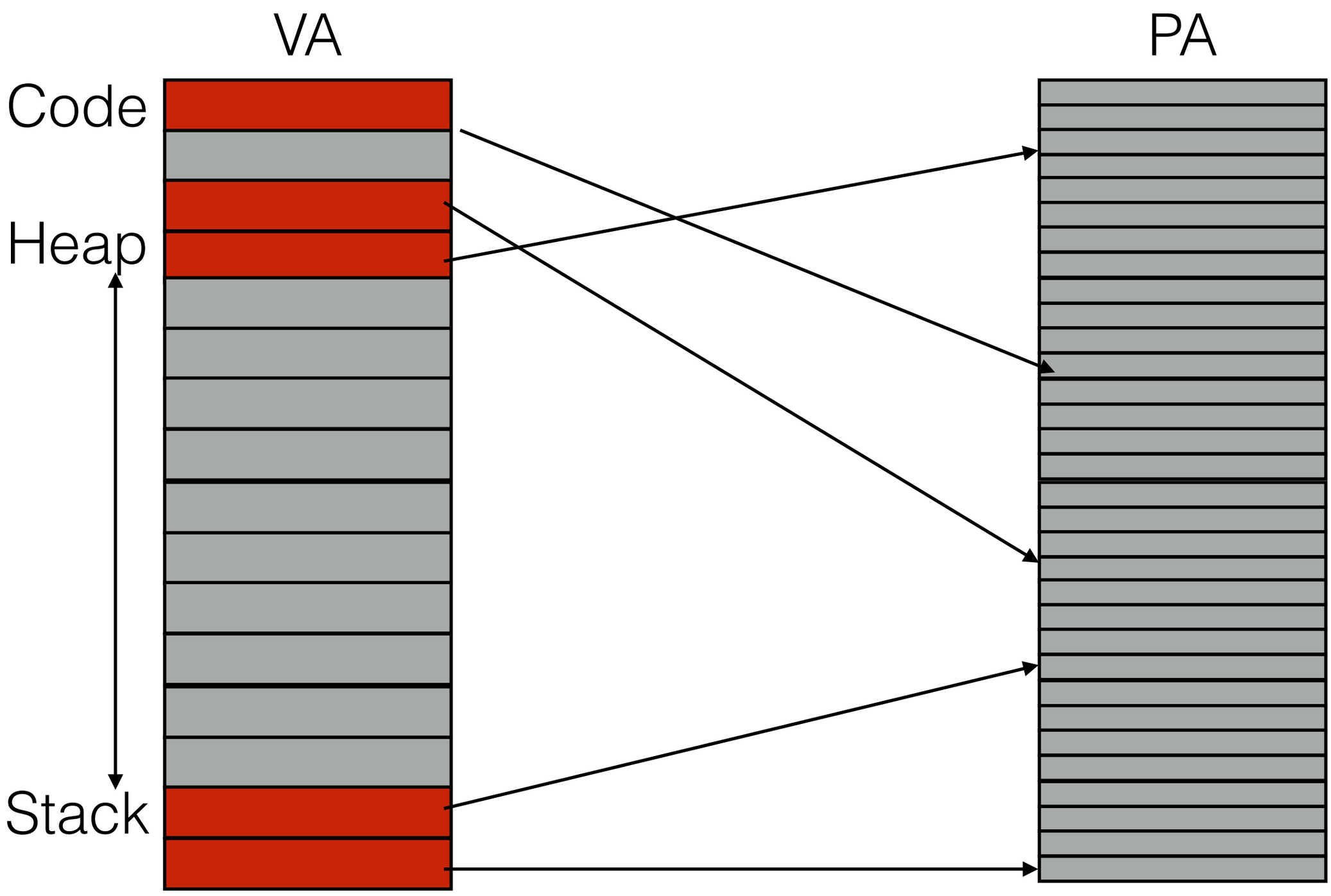
-
-
-
  -
  -
- 4 bytes per translation -> 4 * 2^18 MB = **1 MB/process**

Larger page size —> Fragmentation

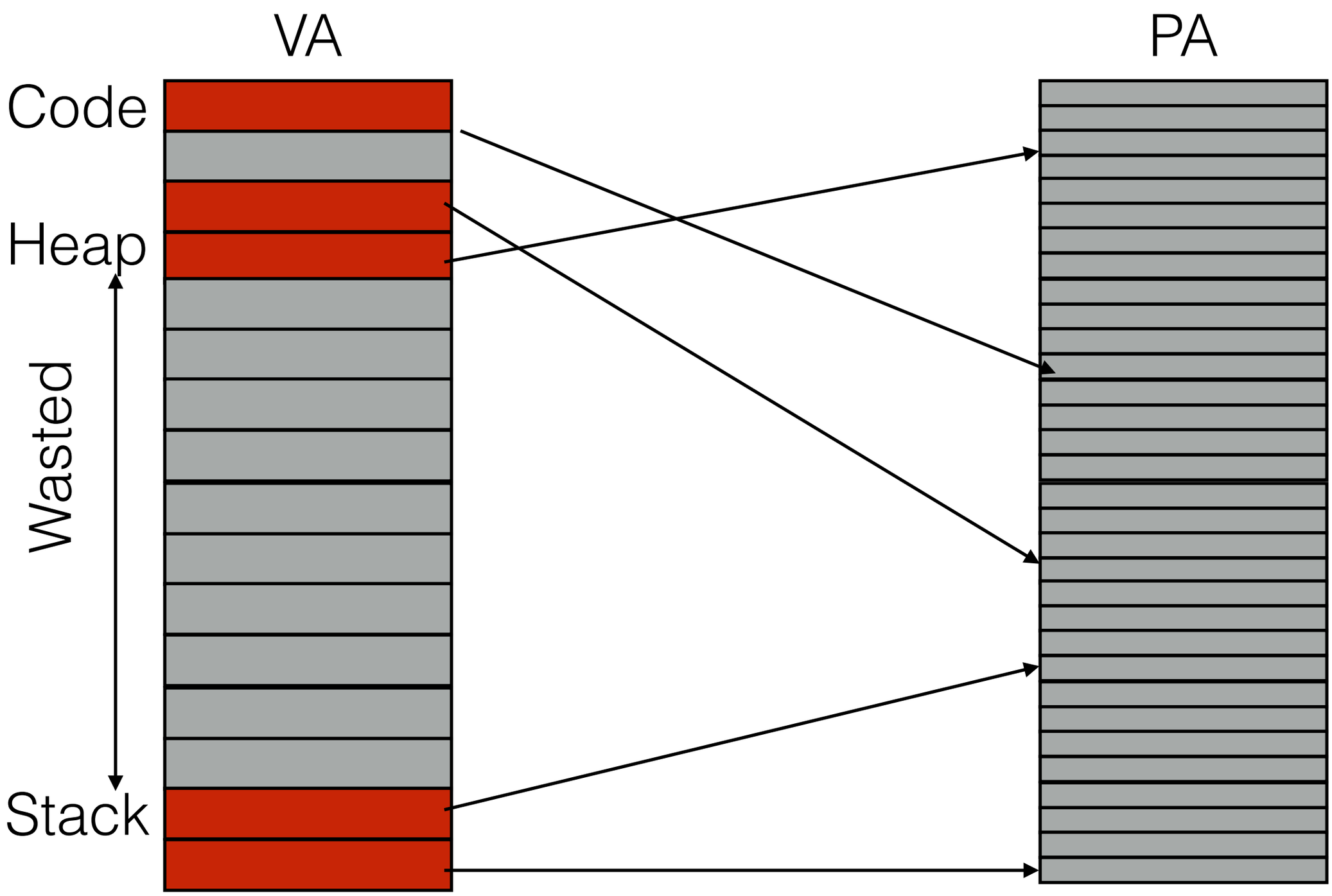
Reducing Memory Overheads of Paging



Reducing Memory Overheads of Paging



Reducing Memory Overheads of Paging



Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Linear Page Table

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Linear Page Table

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Linear Page Table

Lookup = O(1)

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Linear Page Table

Lookup = O(1)
Space = 16*Size

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

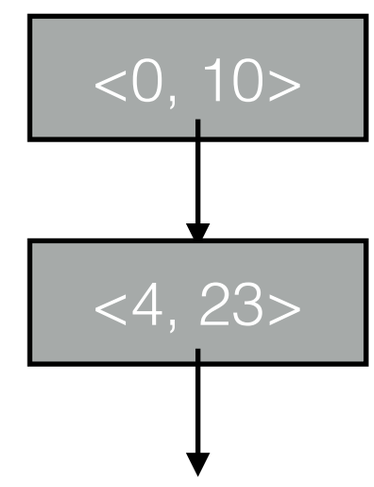
Convert

Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Convert



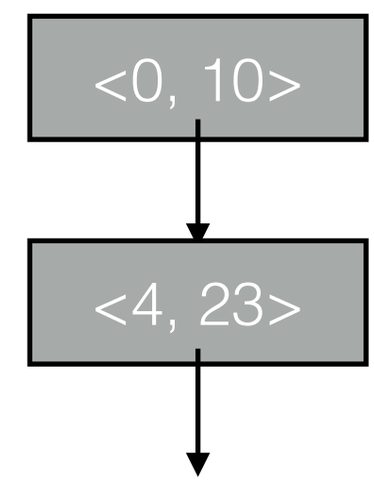
Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Linked List

Convert



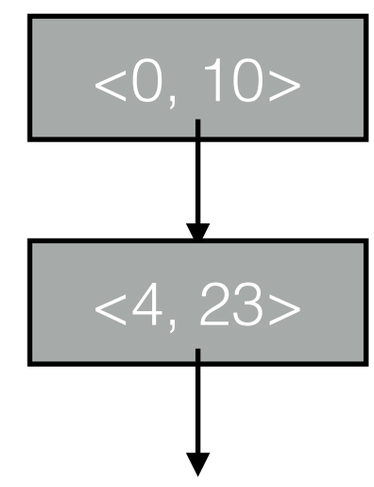
Reducing Memory Overheads of Paging

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Linked List

Convert



Reducing Memory Overheads of Paging

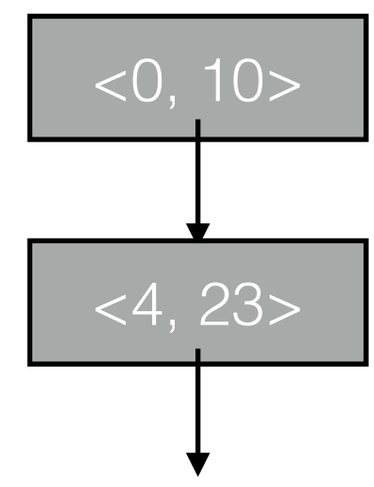
PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Linked List

Lookup = O(n) for n

Convert



Reducing Memory Overheads of Paging

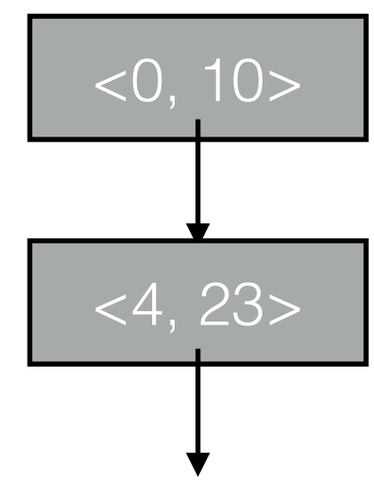
PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

Linked List

Lookup = O(n) for n valid pages (n<<16)

Convert



Reducing Memory Overheads of Paging

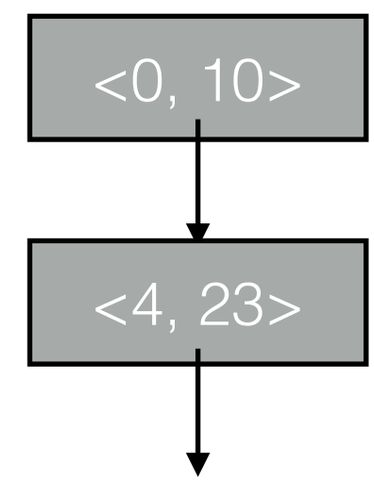
PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Wasted

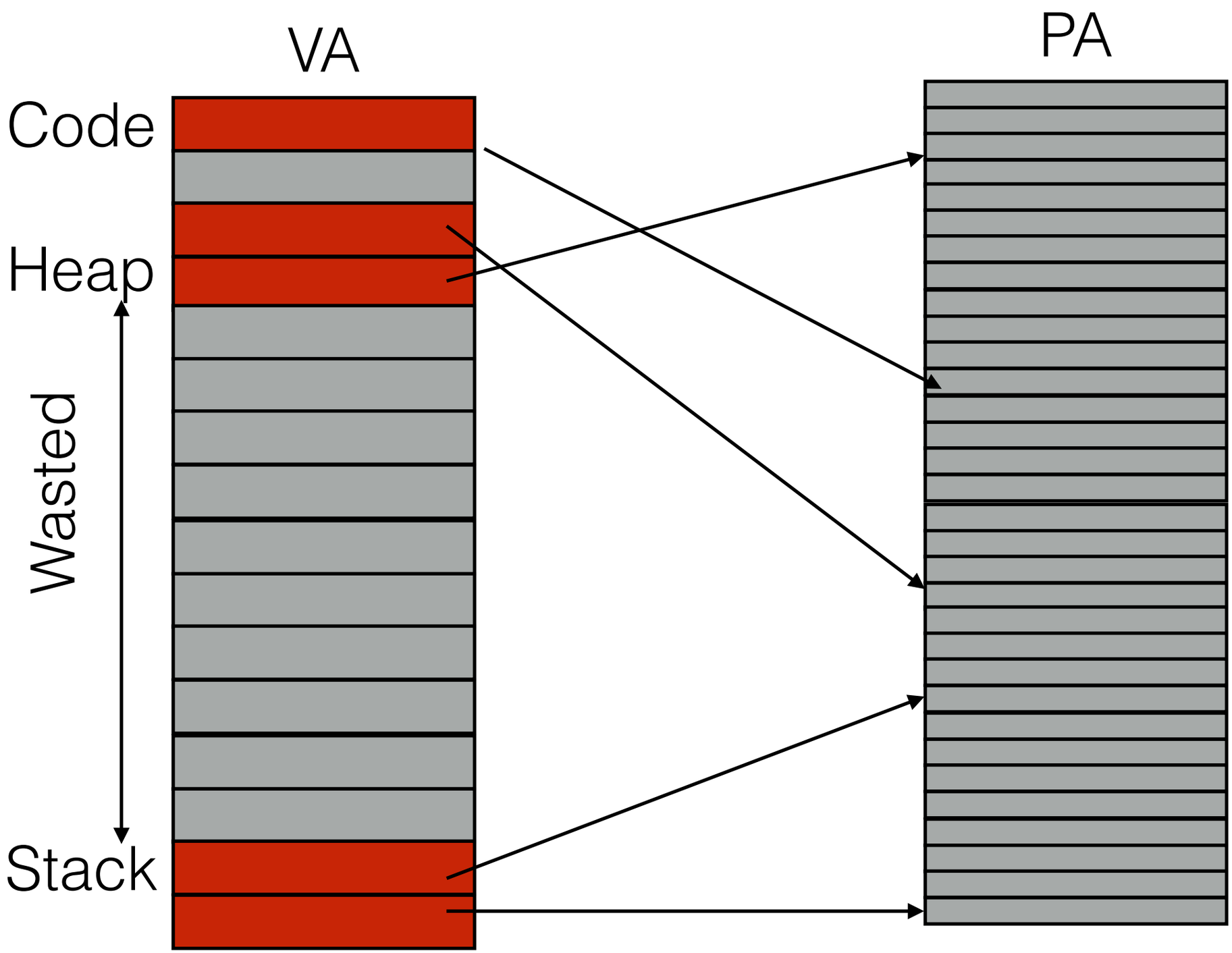
Linked List

Lookup = O(n) for n valid pages (n<<16)
Space = n*Size

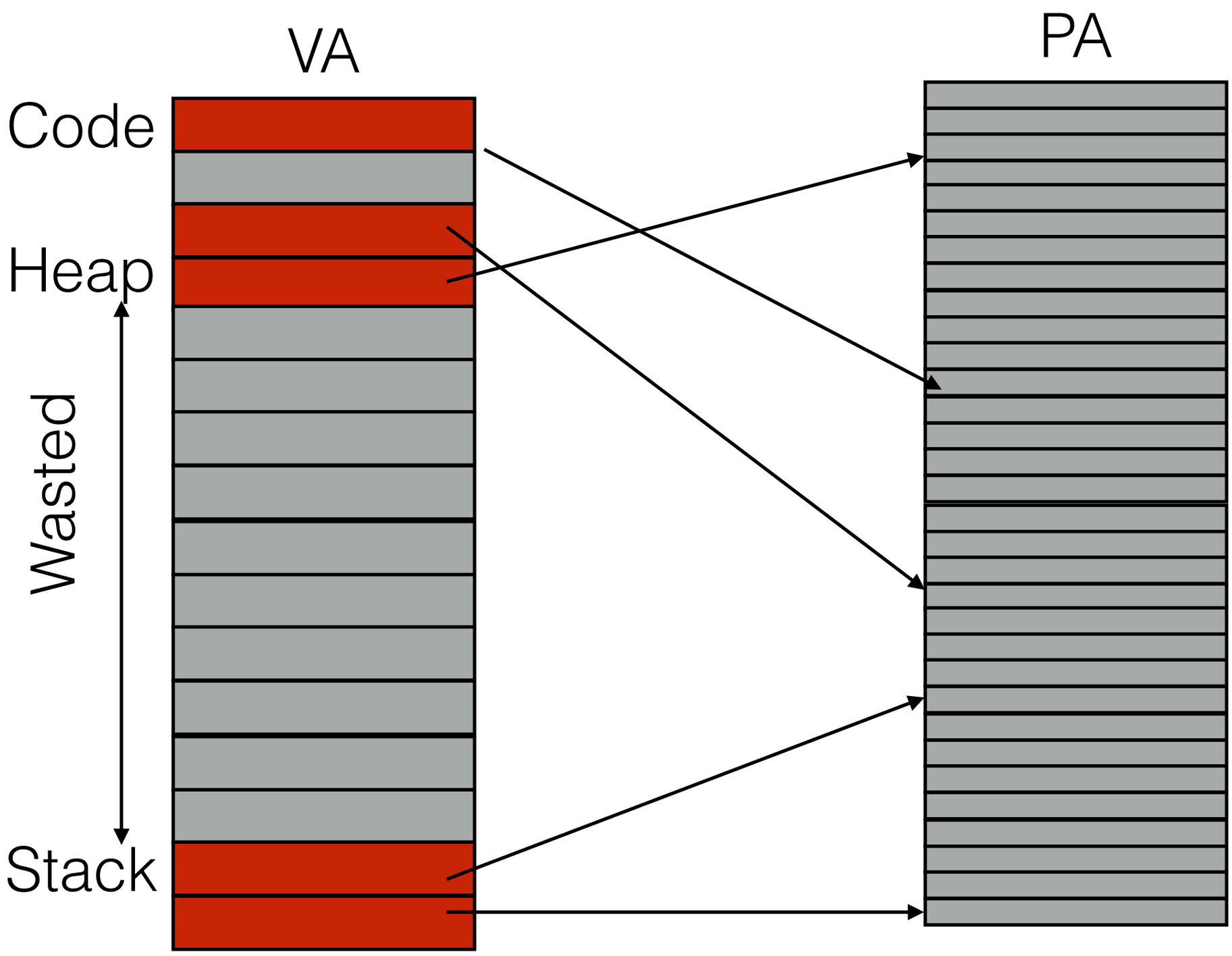
Convert



Reducing Memory Overheads of Paging

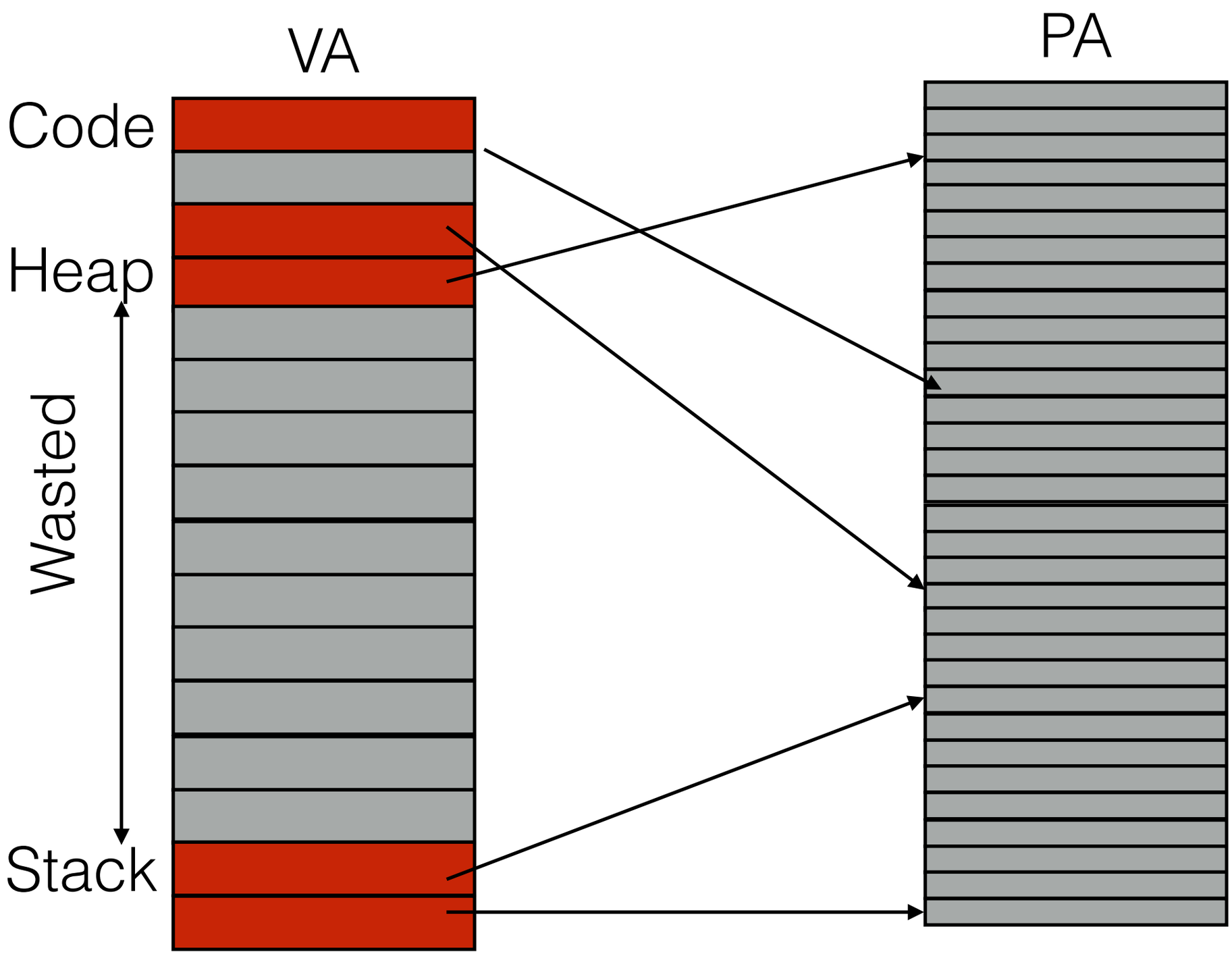


Reducing Memory Overheads of Paging



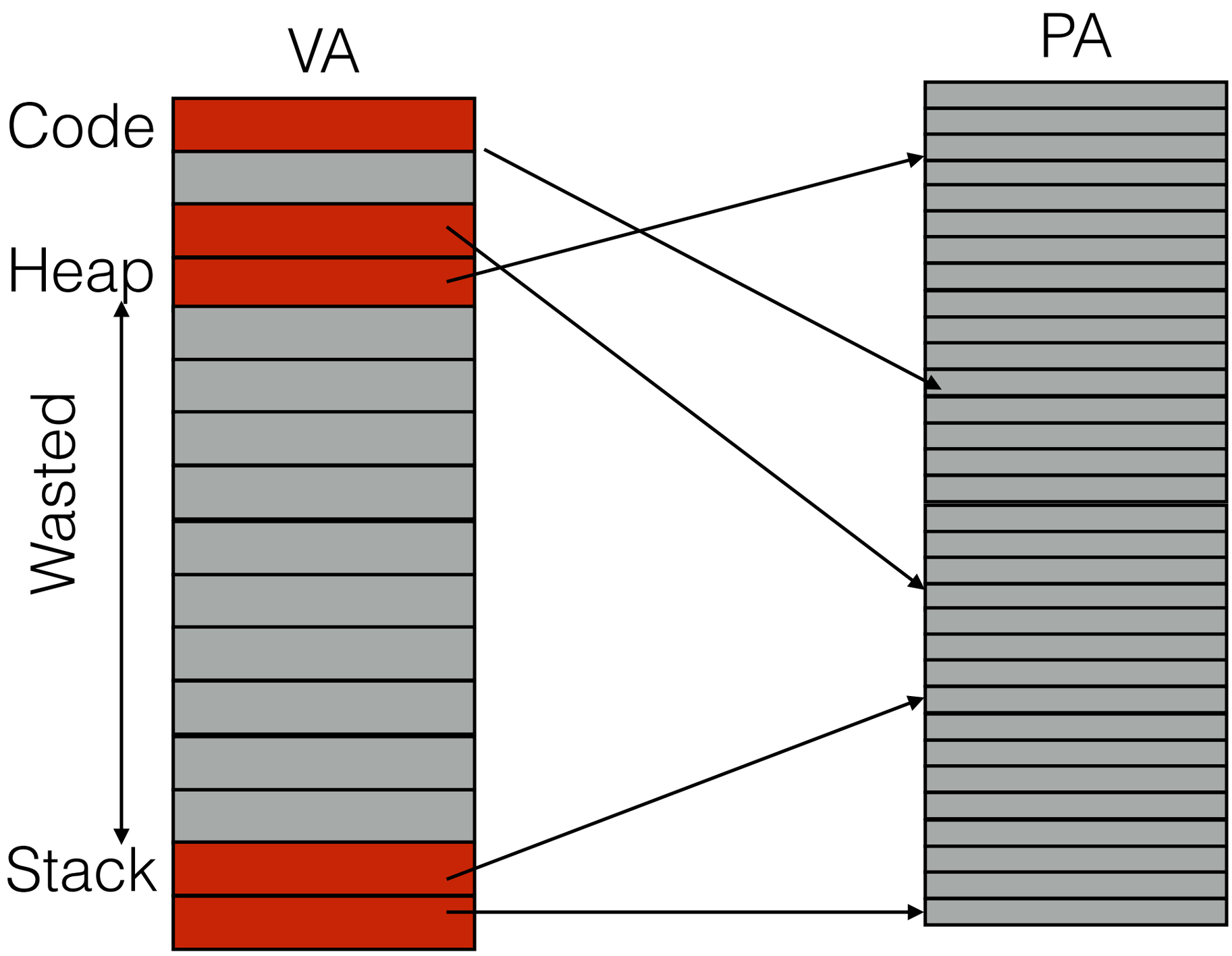
How did we fill holes in PA space earlier?

Reducing Memory Overheads of Paging



How did we fill holes in PA space earlier?

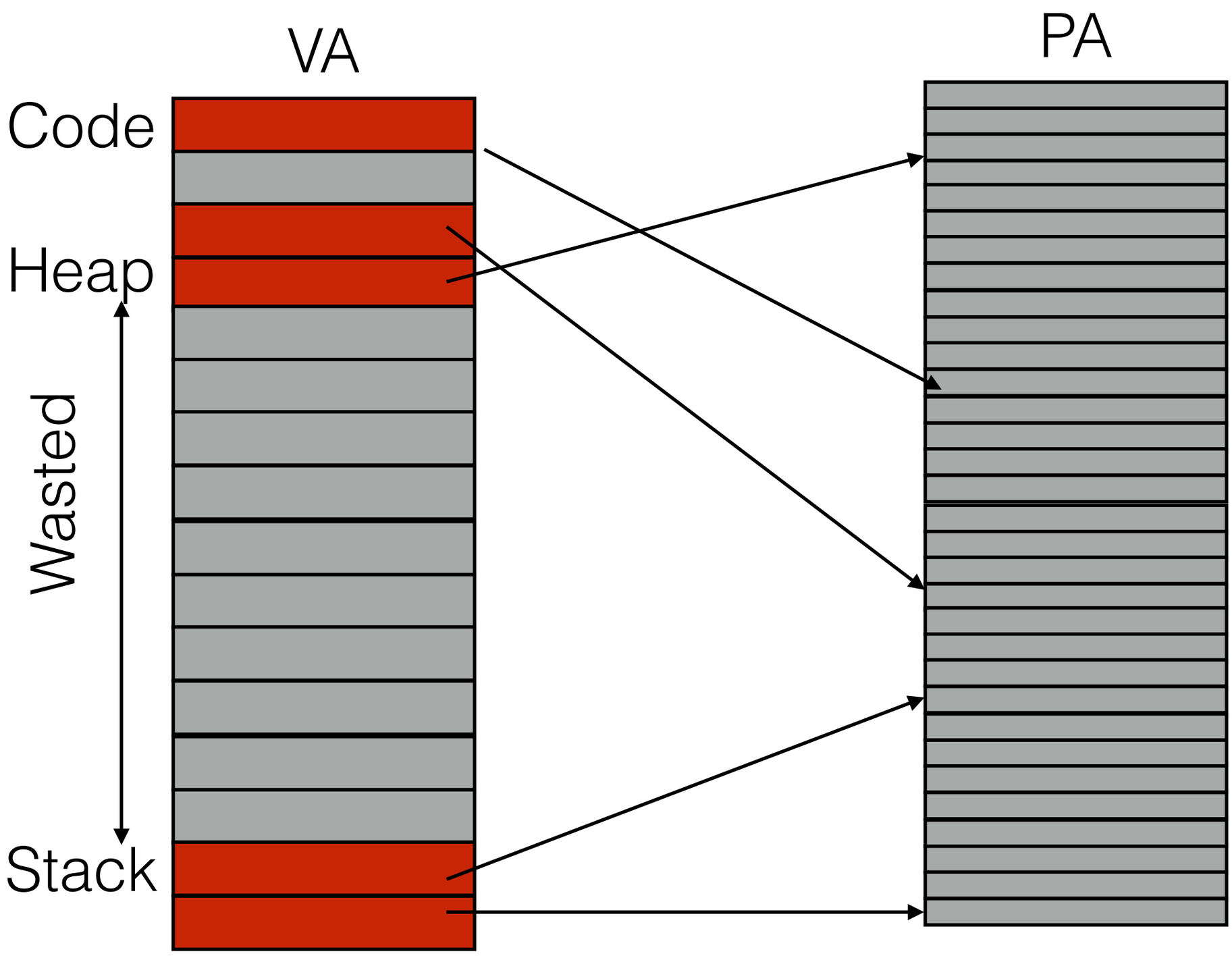
Reducing Memory Overheads of Paging



How did we fill holes in PA space earlier?

1. Segmentation

Reducing Memory Overheads of Paging



How did we fill holes in PA space earlier?

1. Segmentation
2. Paging

Reducing Memory Overheads of Paging - Segmentation + PT

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds
  - Base & Bounds stored in :

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds
  - Base & Bounds stored in :
    - MMU

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds
  - Base & Bounds stored in :
    - MMU
  - What did Base store in regular segmentation?

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds
  - Base & Bounds stored in :
    - MMU
  - What did Base store in regular segmentation?
    - PA where segment resided

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds
  - Base & Bounds stored in :
    - MMU
  - What did Base store in regular segmentation?
    - PA where segment resided
  - What would Base refer here?

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds
  - Base & Bounds stored in :
    - MMU
  - What did Base store in regular segmentation?
    - PA where segment resided
  - What would Base refer here?
    - PA of PT for segment

Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds
  - Base & Bounds stored in :
    - MMU
  - What did Base store in regular segmentation?
    - PA where segment resided
  - What would Base refer here?
    - PA of PT for segment
  - What would Bounds refer to here?

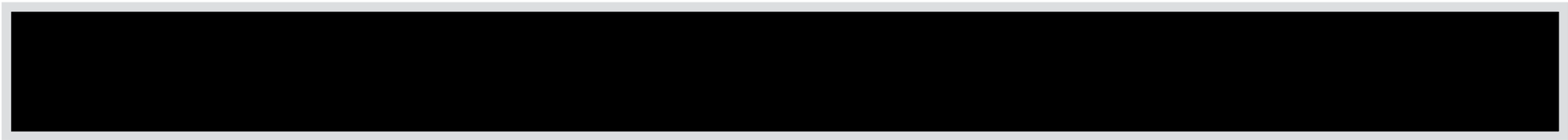
Reducing Memory Overheads of Paging - Segmentation + PT

Idea: use different page tables for heap, stack, etc

- Each PT can be different size
- Each PT has base & bounds
  - Base & Bounds stored in :
    - MMU
  - What did Base store in regular segmentation?
    - PA where segment resided
  - What would Base refer here?
    - PA of PT for segment
  - What would Bounds refer to here?
    - Number of valid pages/End of page table

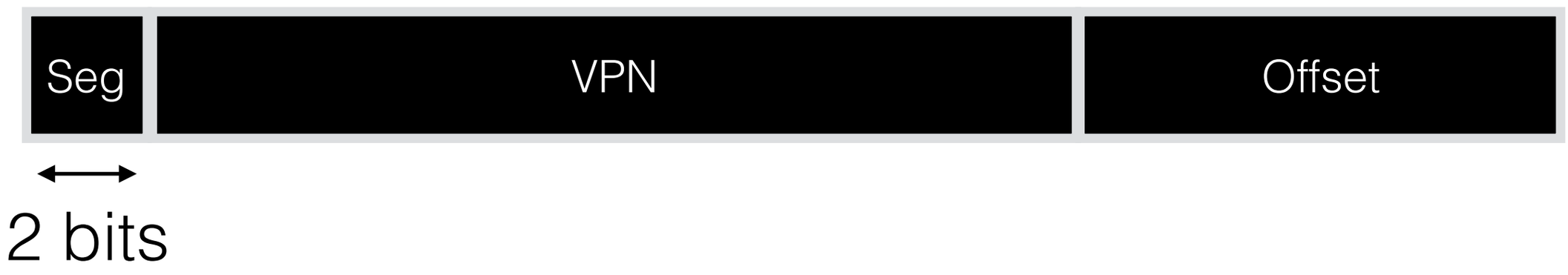
Reducing Memory Overheads of Paging - Segmentation + PT

32 bit VA space



32 bit VA space with 4KB pages

32 bit VA space with 4KB pages for 4 segments



Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	…
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	…
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

PFN	Valid	…
10	1	..
..	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	…
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT

PFN	Valid	…
10	1	..
..	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	…
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0

PFN	Valid	…
10	1	..
..	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0
Bounds = 2

PFN	Valid	...
10	1	..
..	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0
Bounds = 2

PFN	Valid	...
10	1	..
..	1	

PFN	Valid	...
23	1	..
..	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0
Bounds = 2

PFN	Valid	...
10	1	..
..	1	

Heap PT

PFN	Valid	...
23	1	..
..	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0
Bounds = 2

PFN	Valid	...
10	1	..
..	1	

Heap PT
Base = 4

PFN	Valid	...
23	1	..
..	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0
Bounds = 2

PFN	Valid	...
10	1	..
..	1	

Heap PT
Base = 4
Bounds = 2

PFN	Valid	...
23	1	..
..	1	

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0
Bounds = 2

PFN	Valid	...
10	1	..
..	1	

Heap PT
Base = 4
Bounds = 2

PFN	Valid	...
23	1	..
..	1	

Stack PT

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0
Bounds = 2

PFN	Valid	...
10	1	..
..	1	

Heap PT
Base = 4
Bounds = 2

PFN	Valid	...
23	1	..
..	1	

Stack PT
Base = ?

Reducing Memory Overheads of Paging - Segmentation + PT

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Code PT
Base = 0
Bounds = 2

PFN	Valid	...
10	1	..
..	1	

Heap PT
Base = 4
Bounds = 2

PFN	Valid	...
23	1	..
..	1	

Stack PT
Base = ?
Bounds = ?

Reducing Memory Overheads of Paging - Segmentation + PT

Reducing Memory Overheads of Paging - Segmentation + PT

Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:

Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:
  - Leads to memory saving (Large gaps between segments)

Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:
  - Leads to memory saving (Large gaps between segments)
- Cons:

Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:
  - Leads to memory saving (Large gaps between segments)
- Cons:
  - Uses segmentation

Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:
  - Leads to memory saving (Large gaps between segments)
- Cons:
  - Uses segmentation
  - Assumes certain usage pattern of address space

Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:
  - Leads to memory saving (Large gaps between segments)
- Cons:
  - Uses segmentation
  - Assumes certain usage pattern of address space
  - Sparsely used segments (sub-segment internal fragmentation) have same space waste issue

Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:
  - Leads to memory saving (Large gaps between segments)
- Cons:
  - Uses segmentation
    - Assumes certain usage pattern of address space
    - Sparsely used segments (sub-segment internal fragmentation) have same space waste issue
      - How to address this?

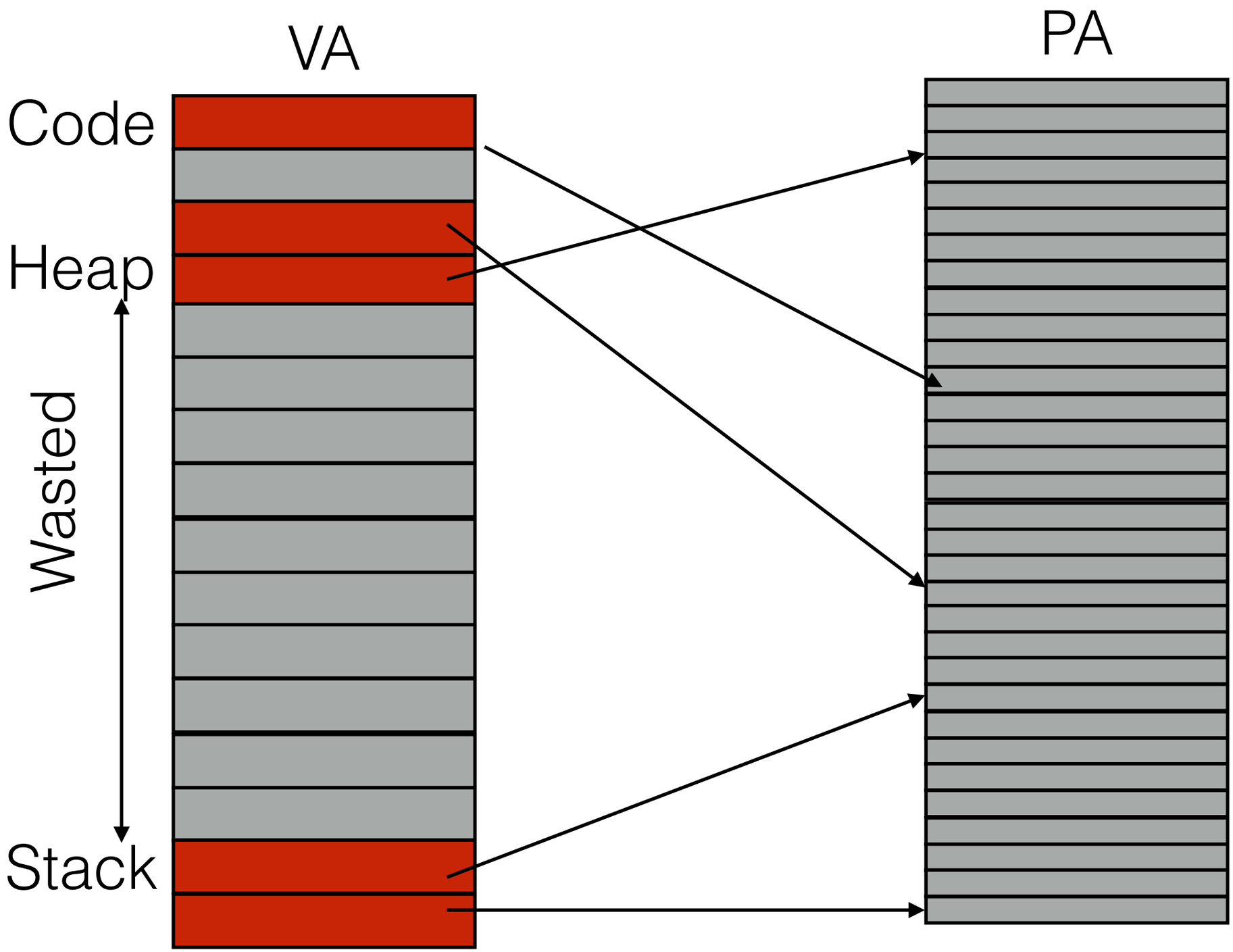
Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:
  - Leads to memory saving (Large gaps between segments)
- Cons:
  - Uses segmentation
  - Assumes certain usage pattern of address space
  - Sparsely used segments (sub-segment internal fragmentation) have same space waste issue
    - How to address this?
      - LinkedList! Getting complex now.

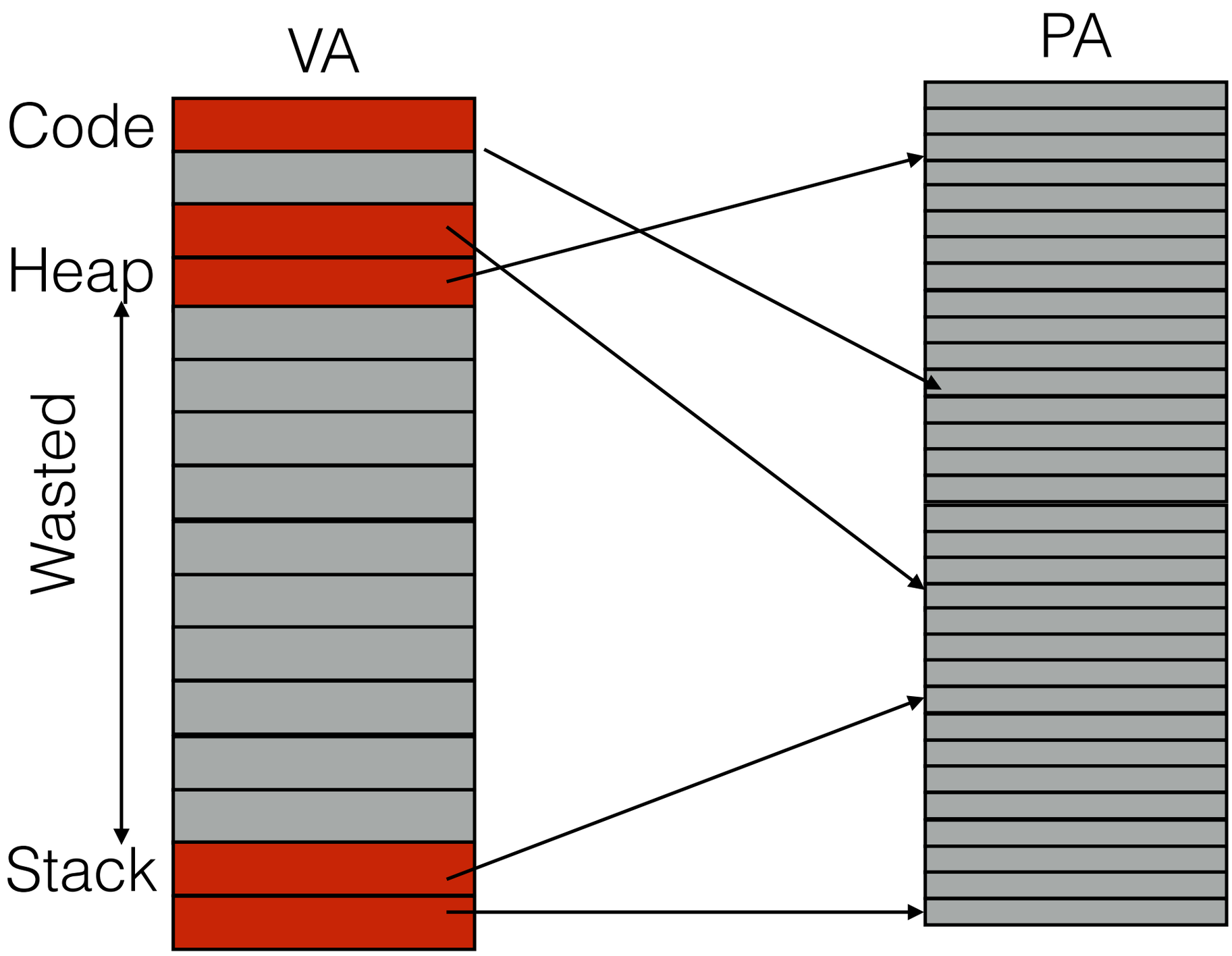
Reducing Memory Overheads of Paging - Segmentation + PT

- Pros:
  - Leads to memory saving (Large gaps between segments)
- Cons:
  - Uses segmentation
  - Assumes certain usage pattern of address space
  - Sparsely used segments (sub-segment internal fragmentation) have same space waste issue
    - How to address this?
      - LinkedList! Getting complex now.
    - Variable size page tables —> Can lead to fragmentation

Reducing Memory Overheads of Paging

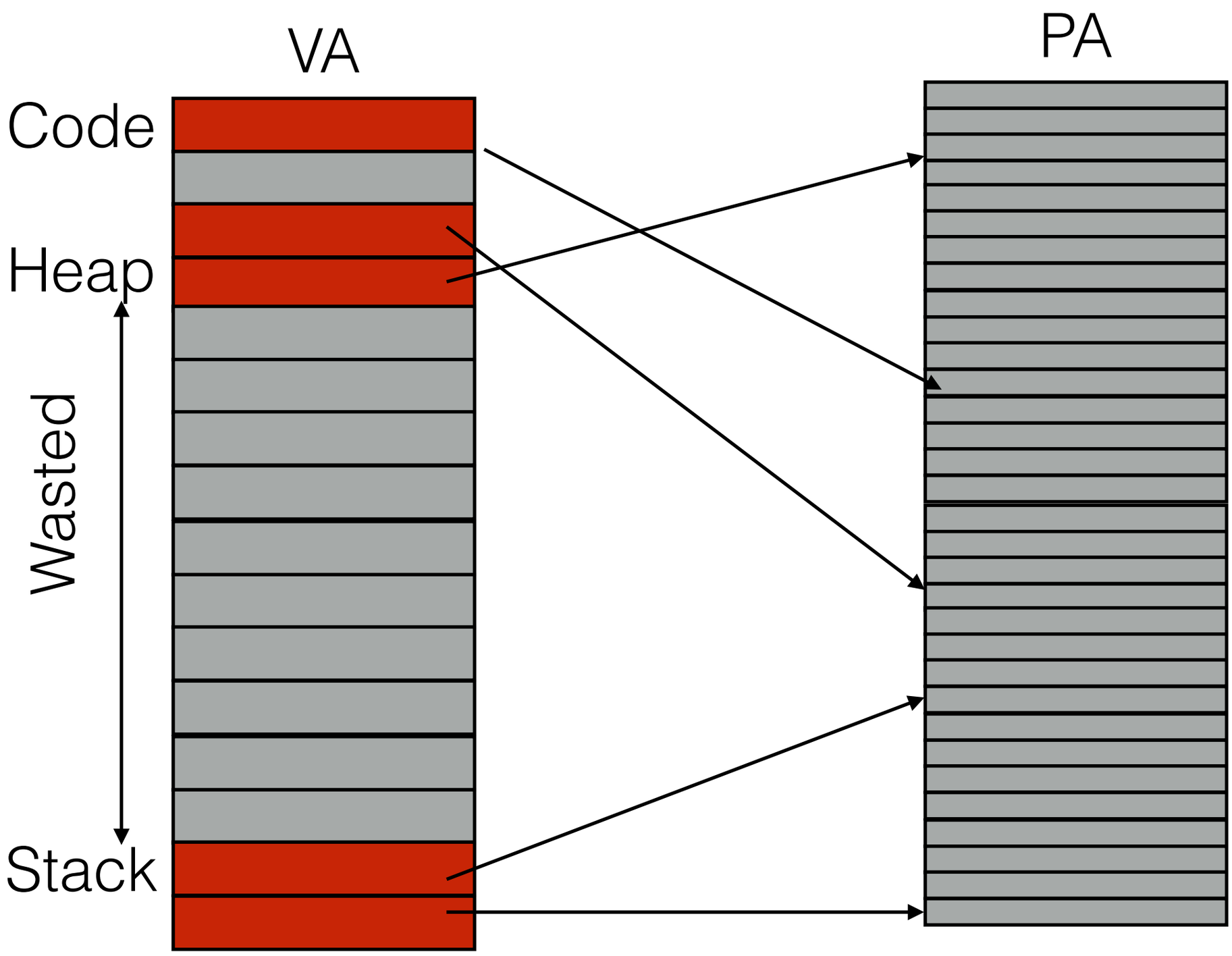


Reducing Memory Overheads of Paging



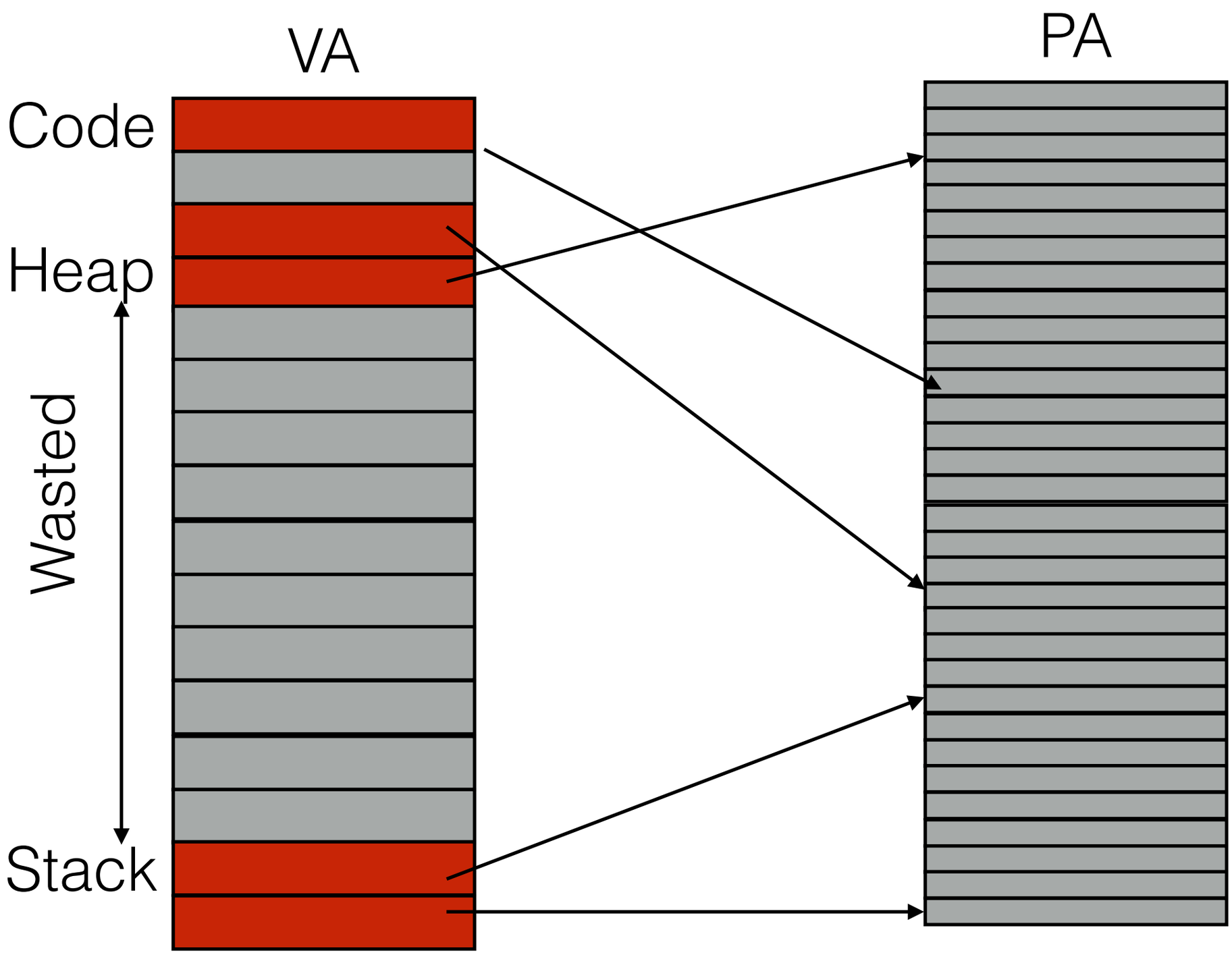
How did we fill holes in PA space earlier?

Reducing Memory Overheads of Paging



How did we fill holes in PA space earlier?

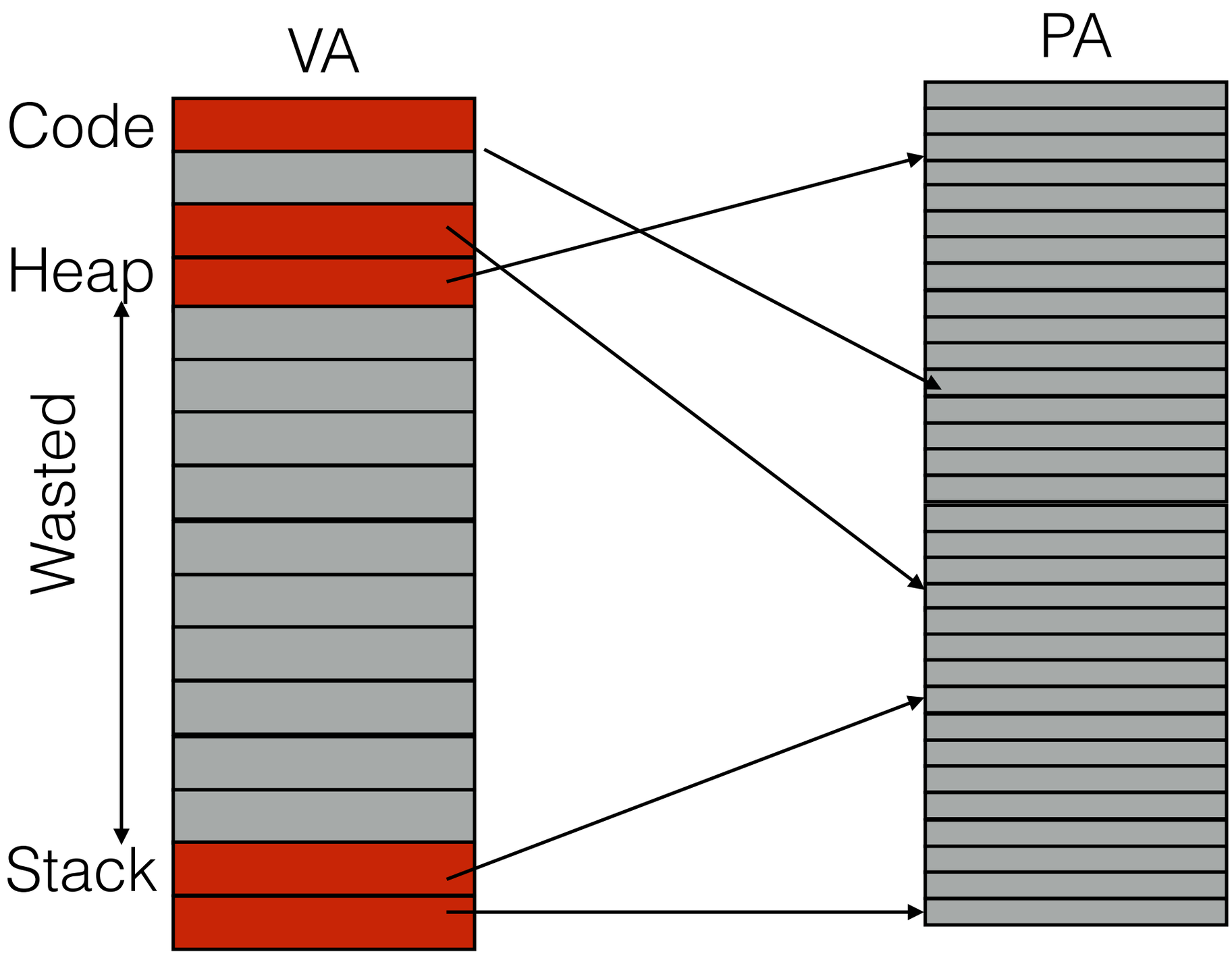
Reducing Memory Overheads of Paging



How did we fill holes in PA space earlier?

1. Segmentation

Reducing Memory Overheads of Paging



How did we fill holes in PA space earlier?

1. Segmentation
2. **Paging**

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

Idea: break PT itself into pages

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

Idea: break PT itself into pages

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

Idea: break PT itself into pages

- A Page Directory refers to pieces

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

Idea: break PT itself into pages

- A Page Directory refers to pieces
- Only have pieces with >0 valid entries

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	...
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Page Directory Entry (PDE)

Total entries = 16

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	...
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	…
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN	Valid	…
10	1	..
-	0	
-	0	
23	1	…

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	...
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	…

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	…

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	

PFN = 201

23	1	
-	0	
-	0	
-	0	

PFN = 202

-	0	
-	0	
-	0	
-	0	

PFN = 203

-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	

PFN = 201

23	1	
-	0	
-	0	
-	0	

PFN = 202

-	0	
-	0	
-	0	
-	0	

PFN = 203

-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	

PFN = 201

23	1	
-	0	
-	0	
-	0	

PFN = 202

-	0	
-	0	
-	0	
-	0	

PFN = 203

-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

PFN = 201

PFN = 202

PFN = 203

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	

PFN = 201

PFN	Valid	...
23	1	
-	0	
-	0	
-	0	

PFN = 202

PFN	Valid	...
-	0	
-	0	
-	0	
-	0	

PFN = 203

PFN	Valid	...
-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN = 201

PFN = 202

PFN = 203

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 202

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

PFN = 201

PFN = 202

PFN = 203

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 202

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN = 203

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	...
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 202

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN = 203

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	…
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	…
201	1	..
202	0	
203	0	
204	1	…

PFN = 201

PFN	Valid	…
10	1	..
-	0	
-	0	
23	1	…

PFN = 202

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 203

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 204

PFN	Valid	…
-	0	..
-	0	
28	1	
4	1	…

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	…
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	…
201	1	..
202	0	
203	0	
204	1	…

PFN = 201

PFN	Valid	…
10	1	..
-	0	
-	0	
23	1	…

PFN = 202

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 203

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 204

PFN	Valid	…
-	0	..
-	0	
28	1	
4	1	…

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	…
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	…
201	1	..
202	0	
203	0	
204	1	…

PFN = 201

PFN	Valid	…
10	1	..
-	0	
-	0	
23	1	…

PFN = 202

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 203

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 204

PFN	Valid	…
-	0	..
-	0	
28	1	
4	1	…

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	…
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	…
201	1	..
202	0	
203	0	
204	1	…

PFN = 201

PFN	Valid	…
10	1	..
-	0	
-	0	
23	1	…

PFN = 202

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 203

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 204

PFN	Valid	…
-	0	..
-	0	
28	1	
4	1	…

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	...
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

Total entries = 16 + 4

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 202

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN = 203

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	...
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

Total entries = 16 + 4

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 202

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN = 203

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

	PFN	Valid	…
PFN = 200	10	1	..
	-	0	
	-	0	
	-	0	
PFN = 201	23	1	
	-	0	
	-	0	
PFN = 202	-	0	
	-	0	
	-	0	
	-	0	
PFN = 203	-	0	
	-	0	
	28	1	
	4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	…
201	1	..
202	0	
203	0	
204	1	…

Total entries = 16 + 4

PFN = 201

PFN	Valid	…
10	1	..
-	0	
-	0	
23	1	…

PFN = 202

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 203

PFN	Valid	…
-	0	..
-	0	
-	0	
-	0	…

PFN = 204

PFN	Valid	…
-	0	..
-	0	
28	1	
4	1	…

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN = 201

PFN = 202

PFN = 203

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

Total entries = 16 + 4

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 202

PFN = 203

PFN	Valid	...
-	0	..
-	0	
-	0	
-	0	...

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN = 201

PFN = 202

PFN = 203

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 202

PFN = 203

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Total entries = 16 + 4

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN = 201

PFN = 202

PFN = 203

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 203

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Total entries = 16 + 4

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN = 201

PFN = 202

PFN = 203

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Total entries = 16 + 4

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN = 201

PFN = 202

PFN = 203

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Reducing Memory Overheads of Paging - Segmentation + PT

PFN = 200

PFN = 201

PFN = 202

PFN = 203

PFN	Valid	...
10	1	..
-	0	
-	0	
-	0	
23	1	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
-	0	
28	1	
4	1	

Total entries = 16

Page Directory Entry (PDE)

PFN = 200

PFN	Valid	...
201	1	..
202	0	
203	0	
204	1	...

PFN = 201

PFN	Valid	...
10	1	..
-	0	
-	0	
23	1	...

PFN = 204

PFN	Valid	...
-	0	..
-	0	
28	1	
4	1	...

Total entries = 16 + 4 -8

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

- Multi-level paging allocates memory proportional to requirement

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

- Multi-level paging allocates memory proportional to requirement
  - Sparse address spaces well supported

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

- Multi-level paging allocates memory proportional to requirement
  - Sparse address spaces well supported
- Previously, we needed 4 MB contiguous space for PT memory

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

- Multi-level paging allocates memory proportional to requirement
  - Sparse address spaces well supported
- Previously, we needed 4 MB contiguous space for PT memory
  - Now?

Reducing Memory Overheads of Paging - PT + PT (Multi-level Paging)

- Multi-level paging allocates memory proportional to requirement
  - Sparse address spaces well supported
- Previously, we needed 4 MB contiguous space for PT memory
  - Now?
    - Can place Page Tables anywhere in memory with multi-level paging…