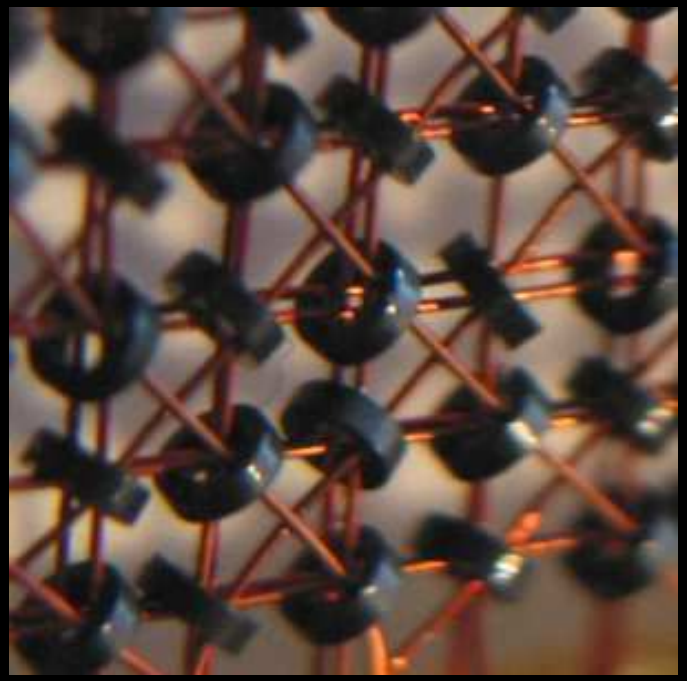
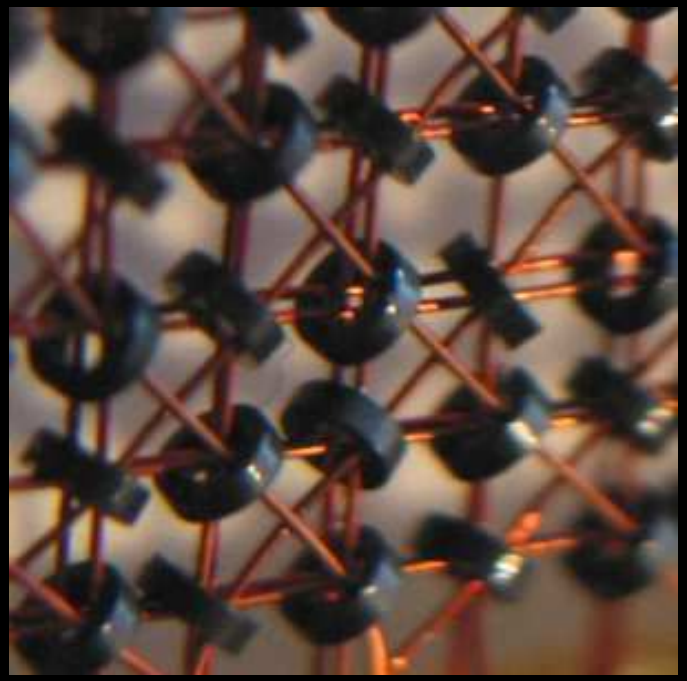
OS Memory Management



OS Memory Management



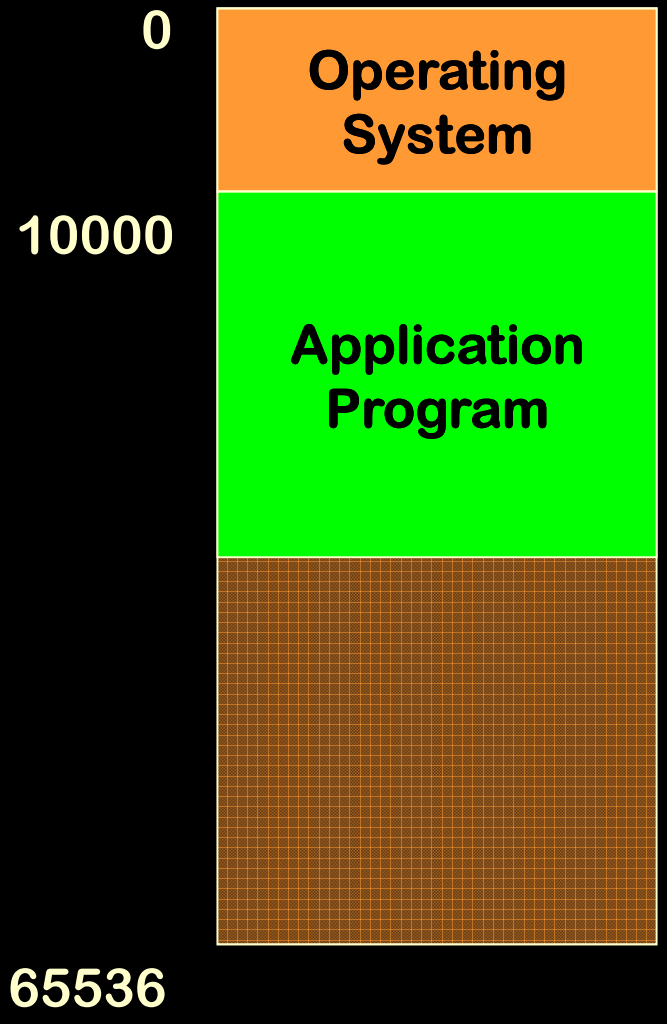
Quoth the raven: "No more core"

OS Manages Resources

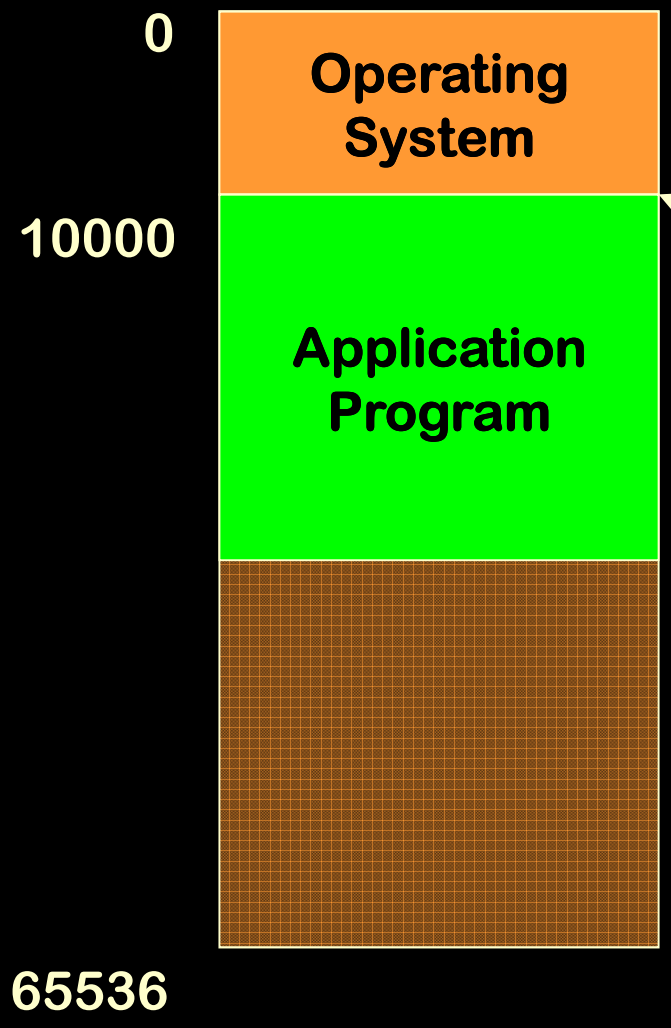
- **Memory**
- CPU
  - Processes
- I/O Devices
- Information
  - Files

Sharing Nicely

Single Contiguous Memory

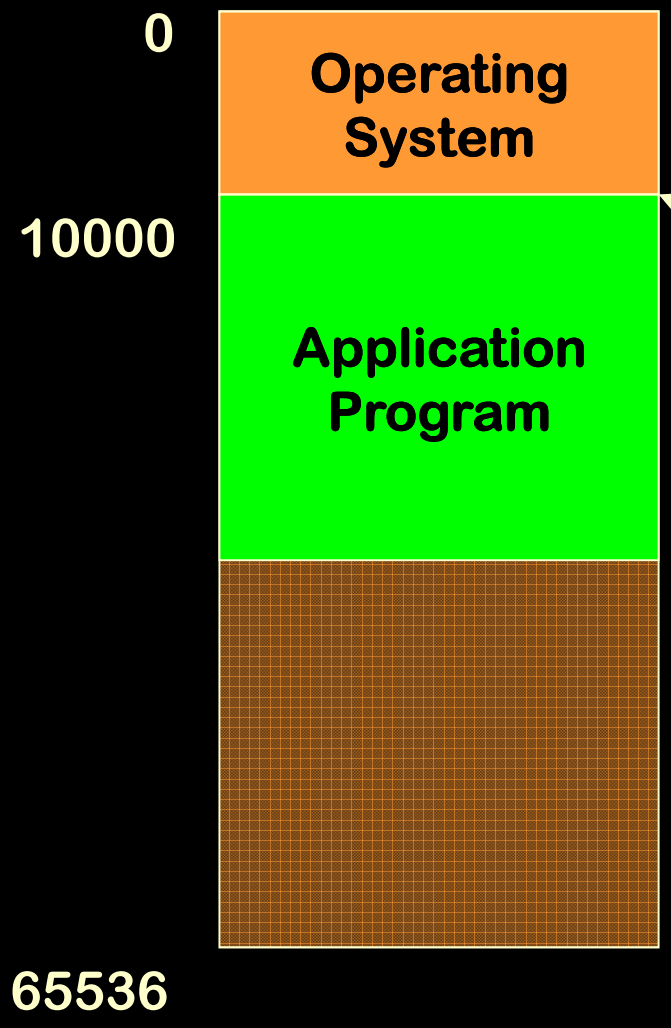


Single Contiguous Memory



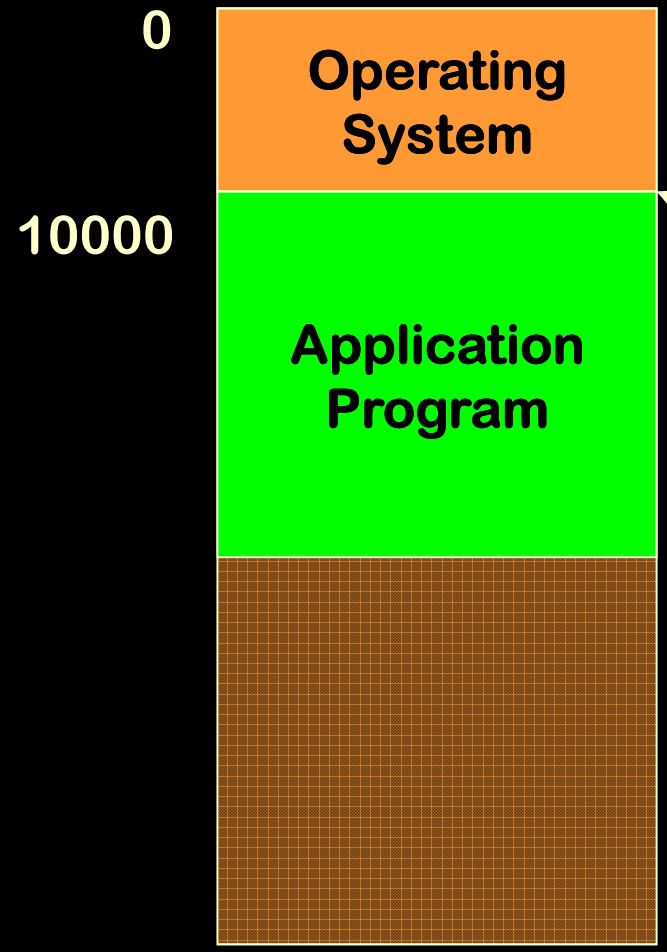

```
        br main
x:      .word 0
main:   lda  x,d
        adda 1,i
        sta  x,d
        ...
```

Single Continuous Memory




```
        br main
x:      .word 0
main:   lda  3,d
        adda 1,i
        sta  3,d
        ...
```

Single Continuous Memory

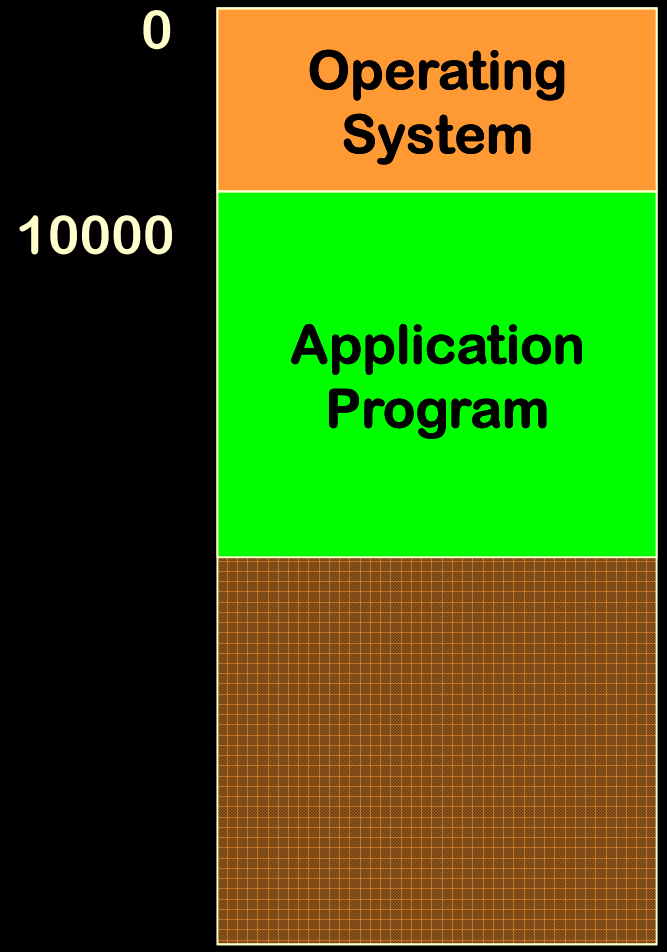



```
        br main
x:      .word 0
main:   lda  3,d
        adda 1,i
        sta  3,d
        ...
```

Logical Address = 3

Physical Address = 10003

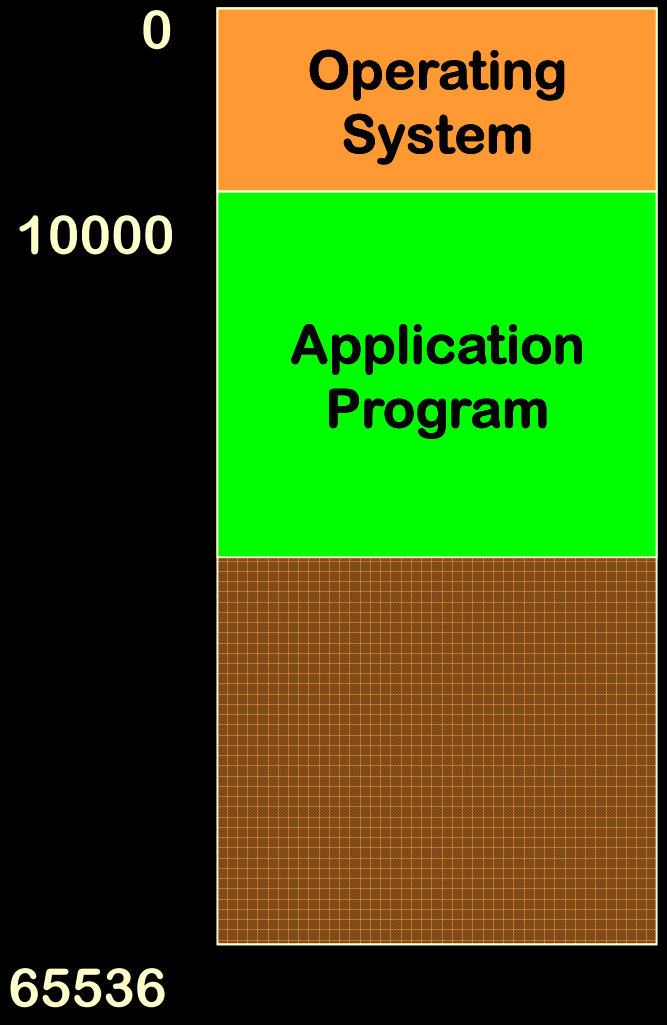
Managing Memory



10000 Base Register

Logical Address = 3
Physical Address = 10003

Managing Memory



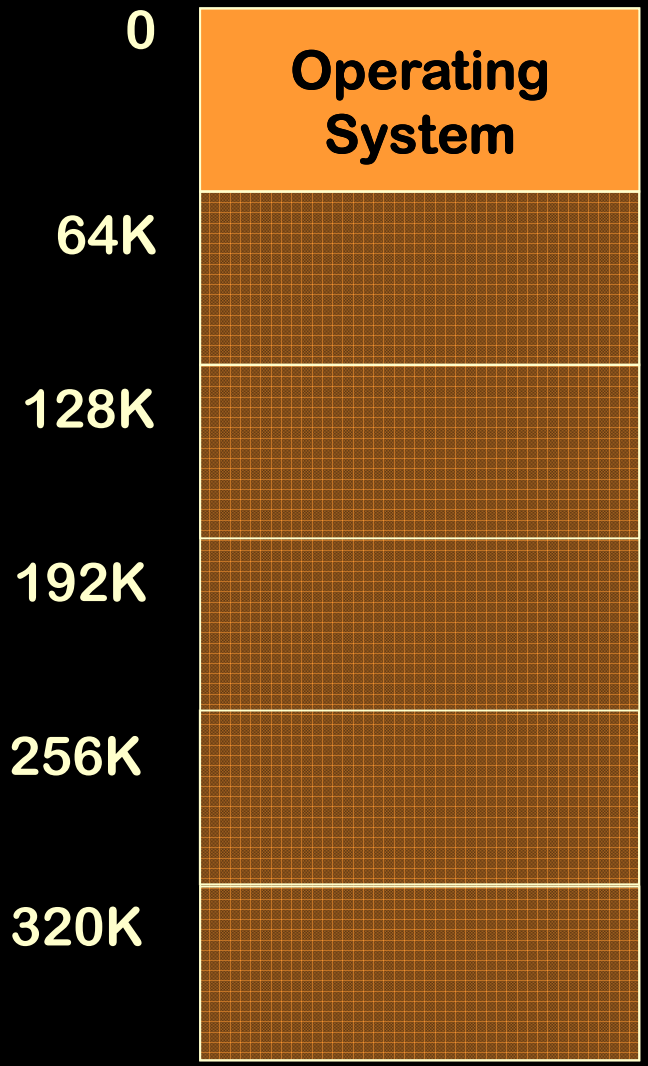
10000 Base Register

+ Logical Address

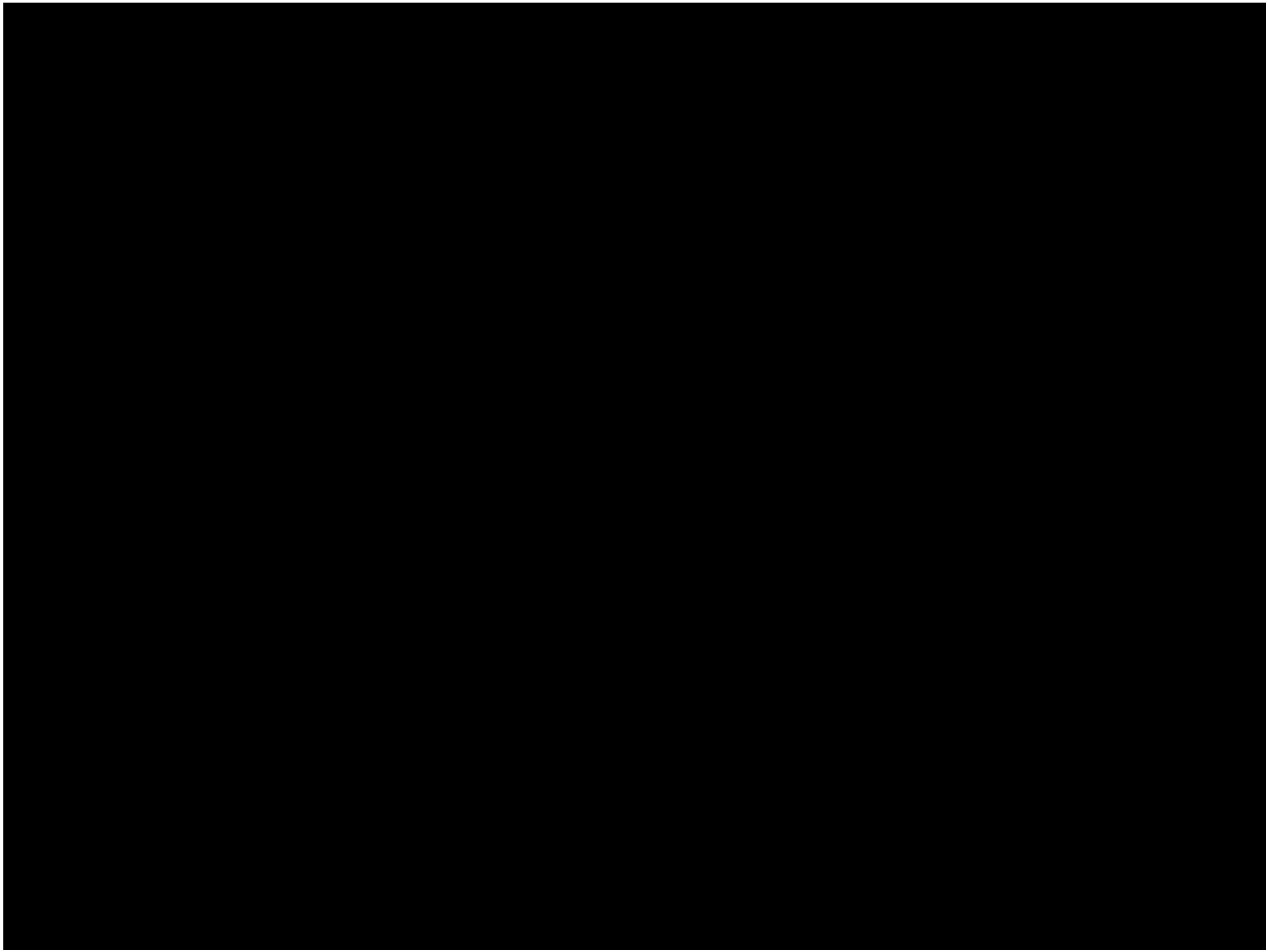
= Physical Address

Logical Address = 3
Physical Address = 10003

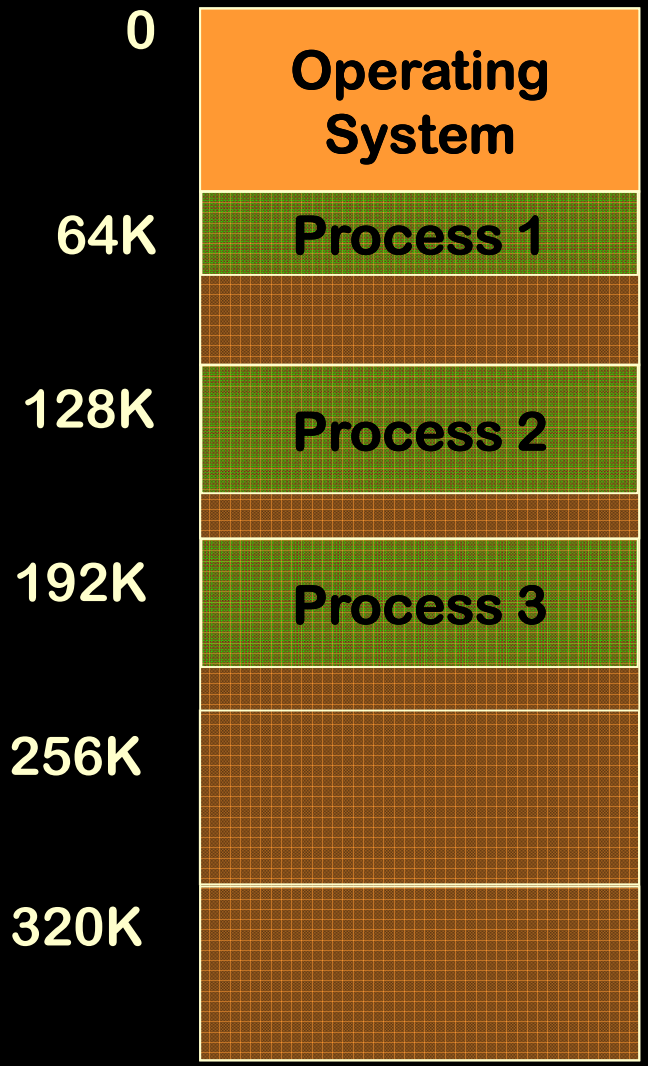
Fixed Partitions



Base Register

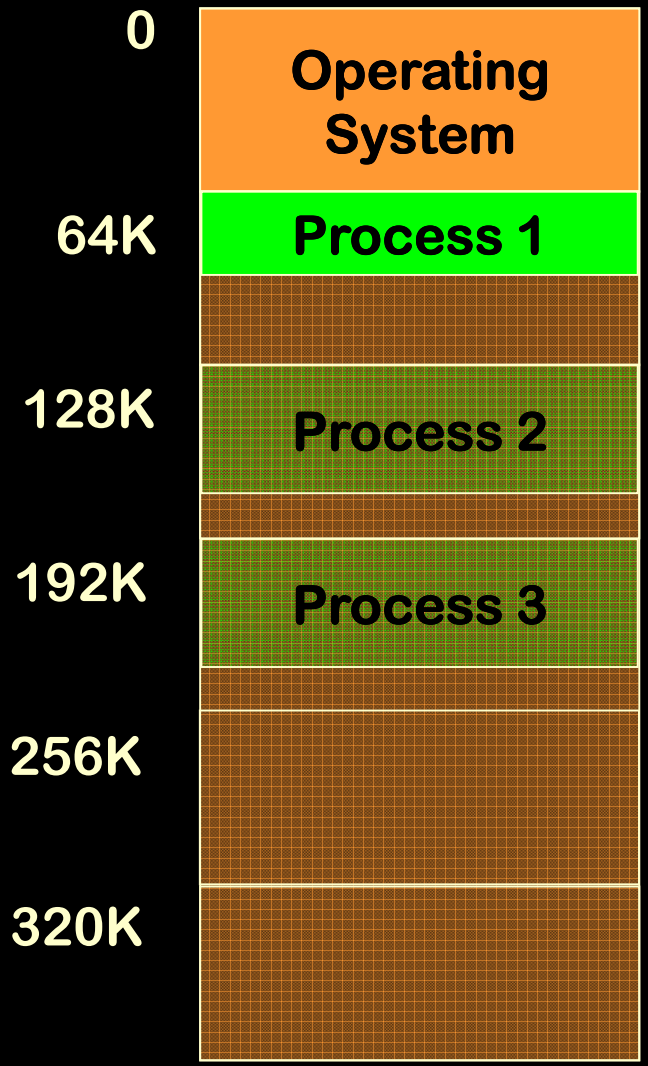


Fixed Partitions



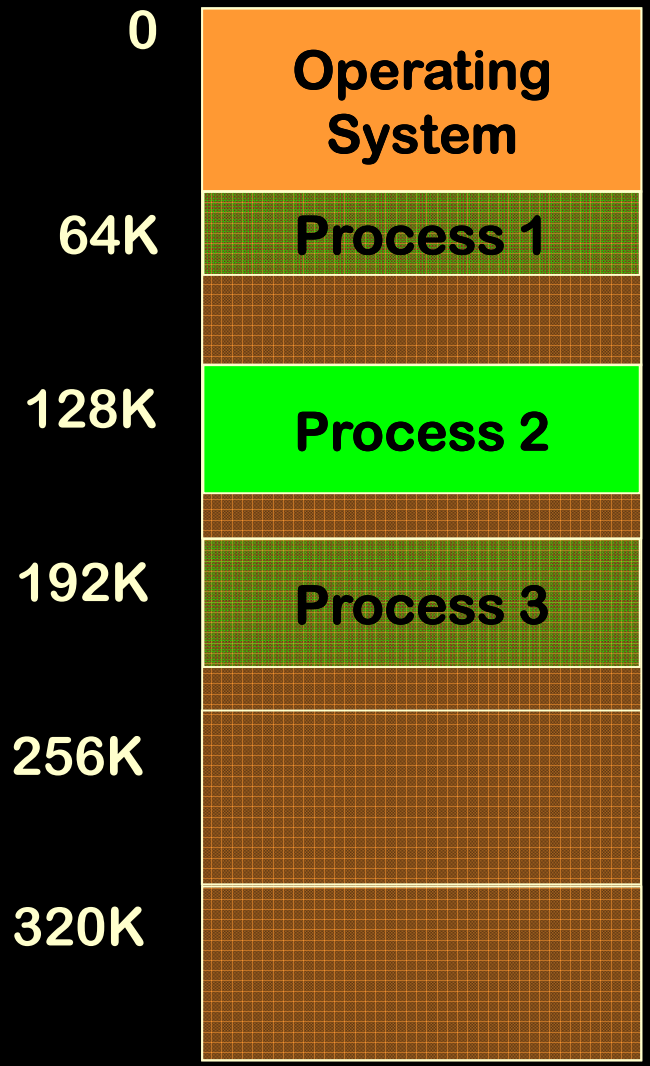
Base Register

Fixed Partitions



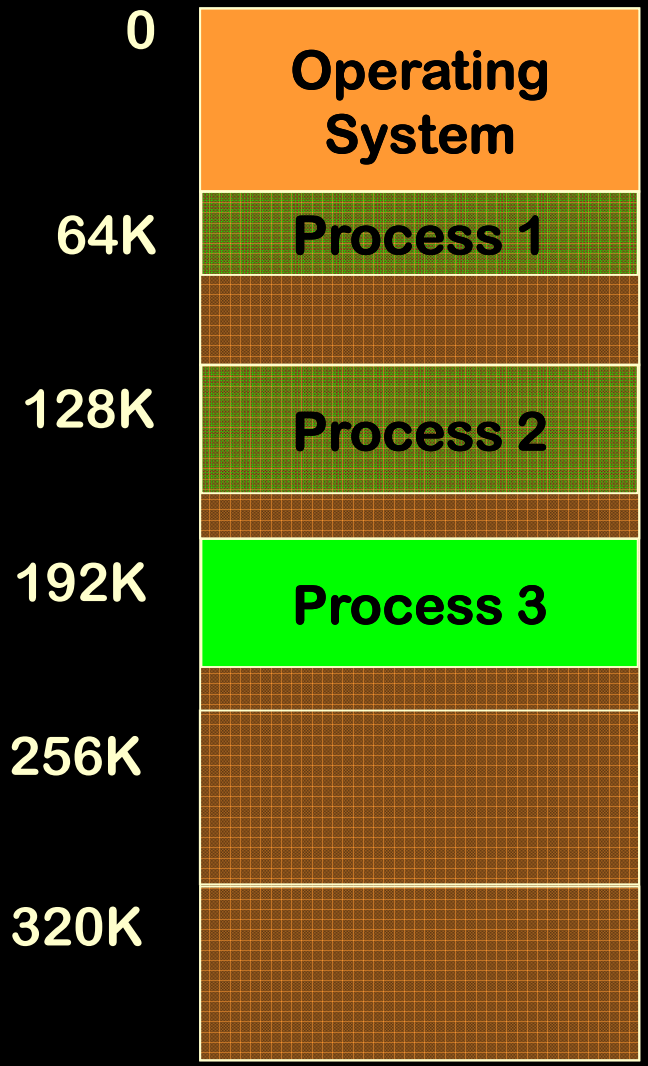
65536 Base Register

Fixed Partitions



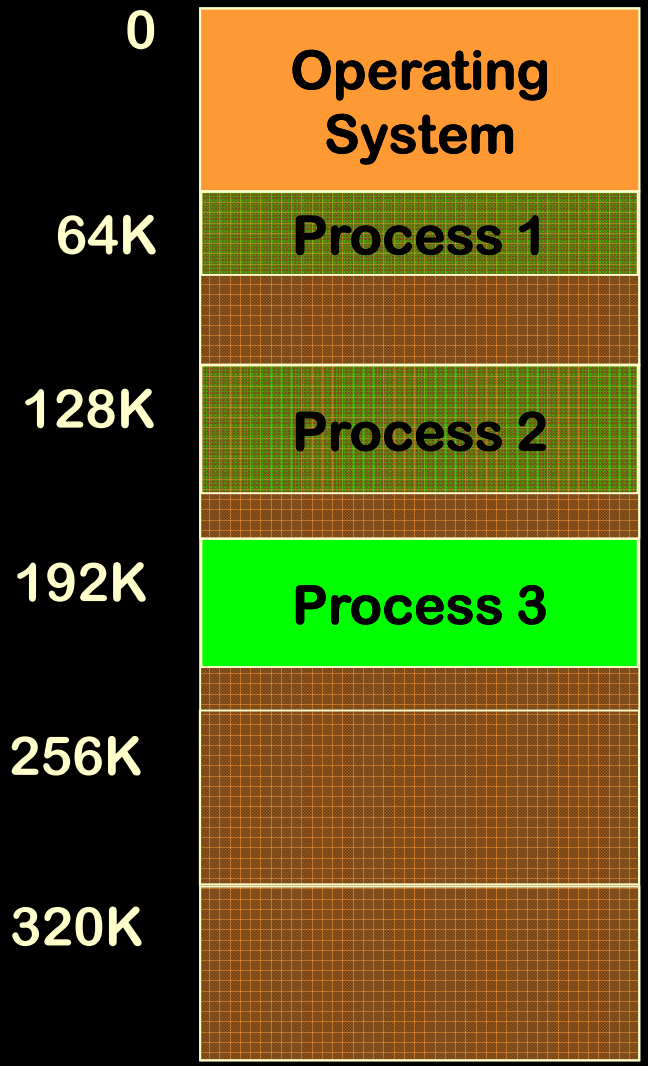
131072 Base Register

Fixed Partitions



196608 Base Register

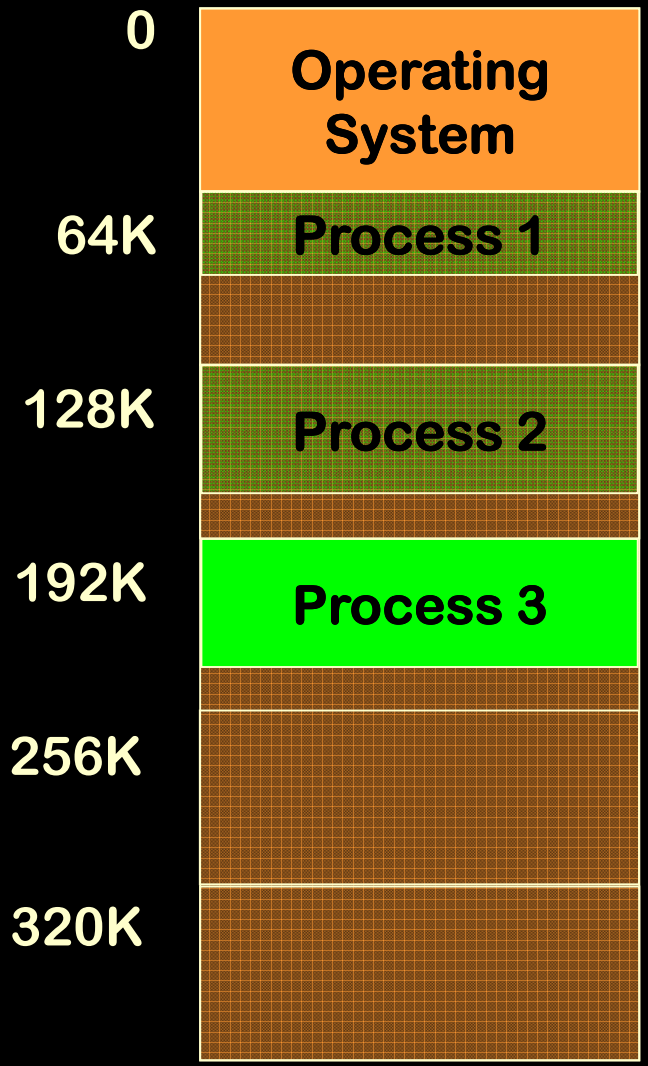
Fixed Partitions



196608 **Base Register**

Process	Base
1	64K
2	128K
...	

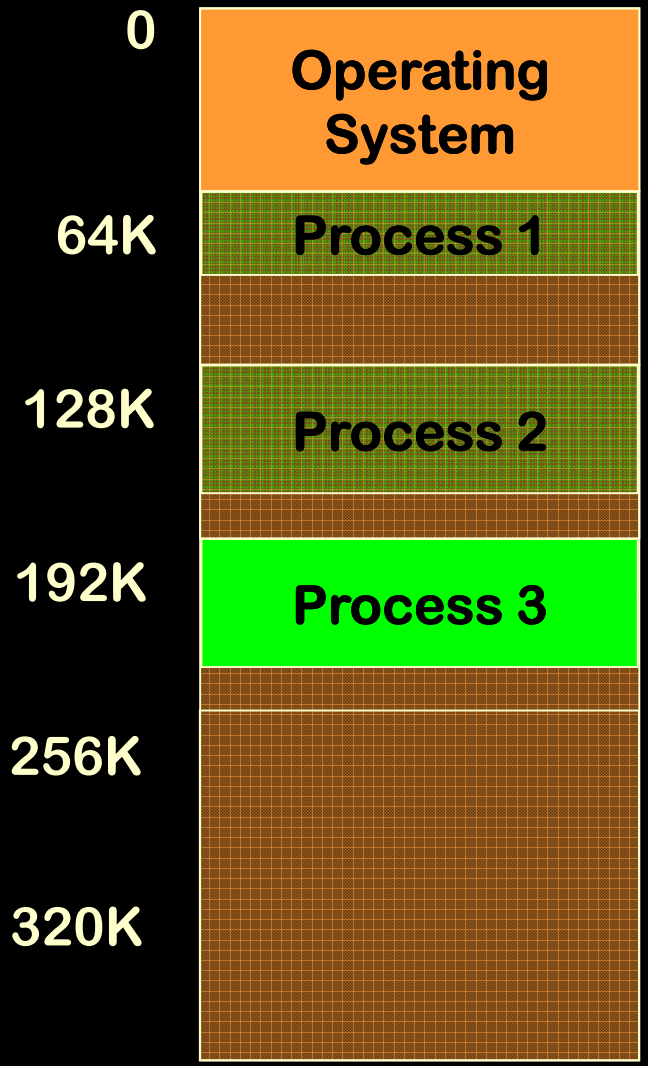
Fixed Partitions



196608 Base Register

Big Process

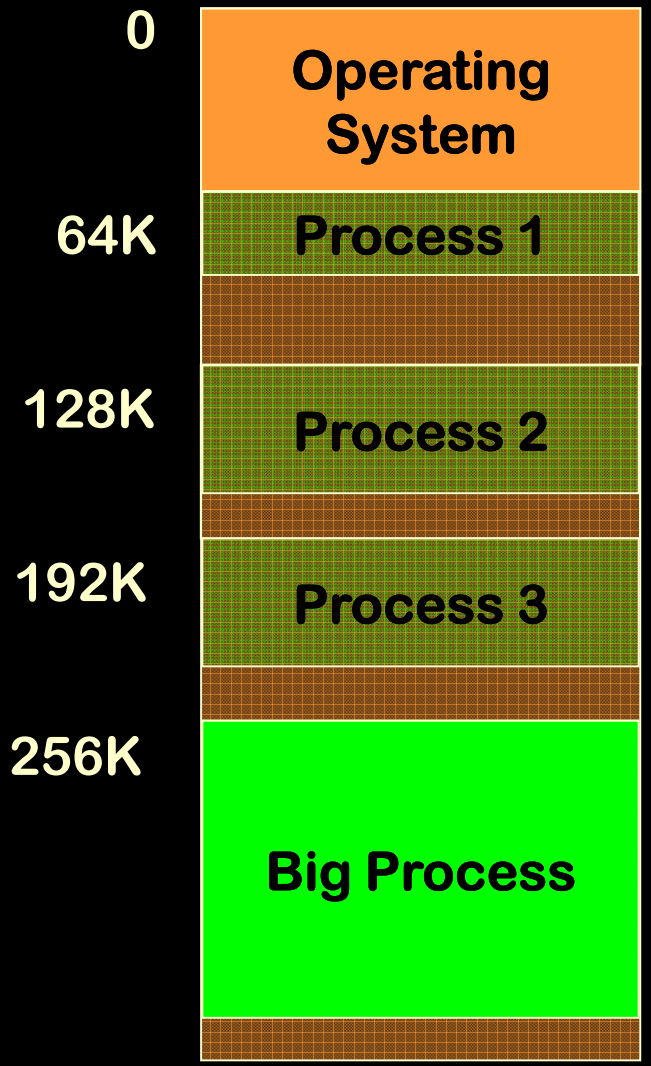
Fixed, Variable-size Partitions



196608 Base Register

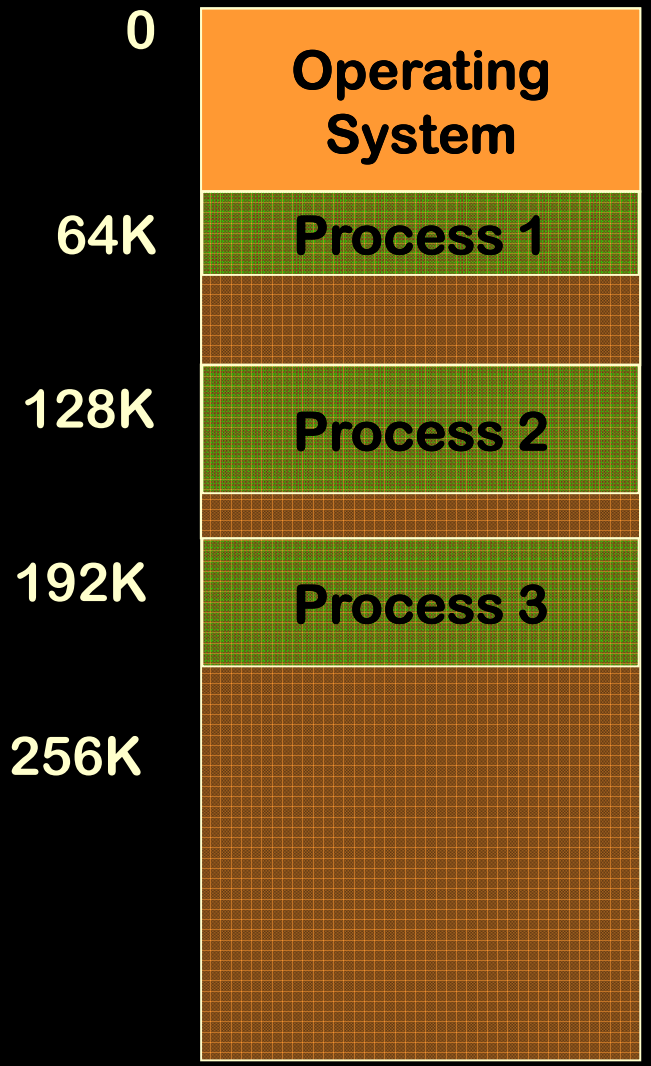
Big Process

Fixed, Variable-size Partitions



262144 Base Register

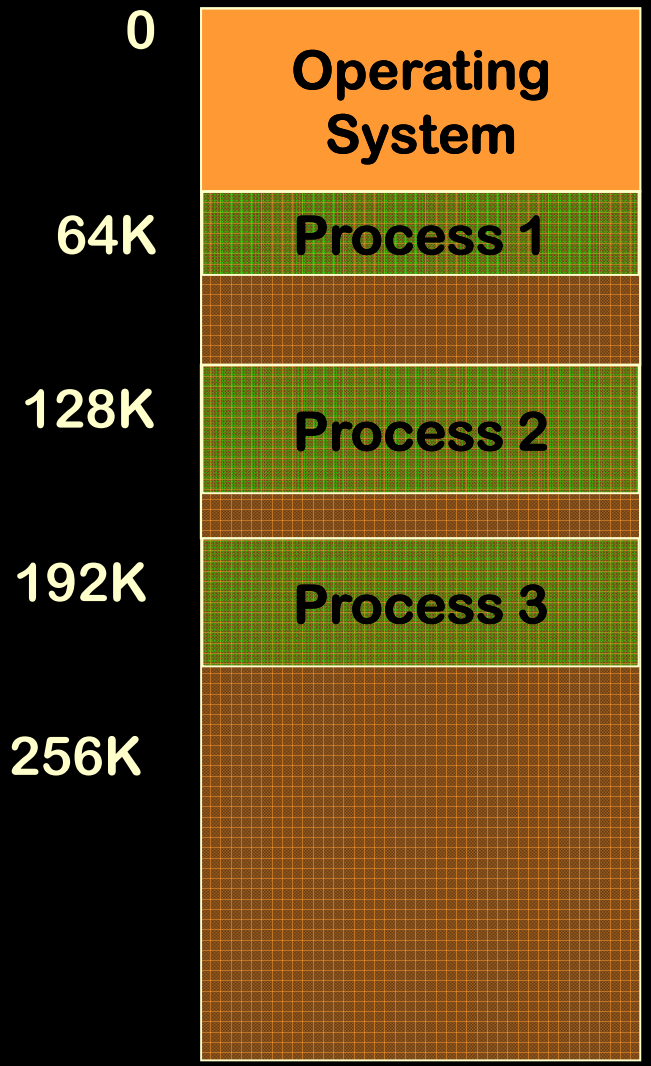
Fixed, Variable-size Partitions



Should I load it?

Process 6

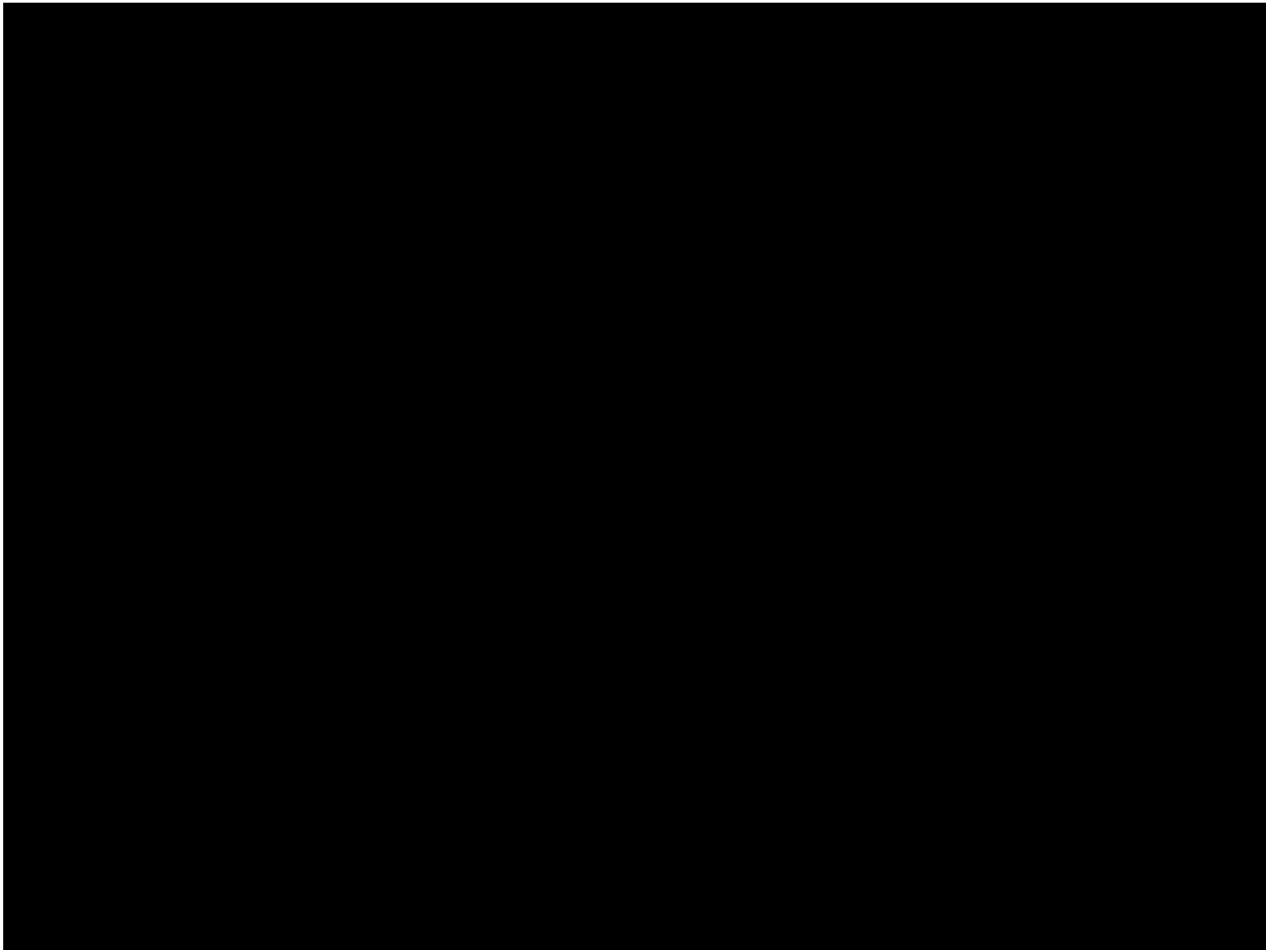
Fixed, Variable-size Partitions



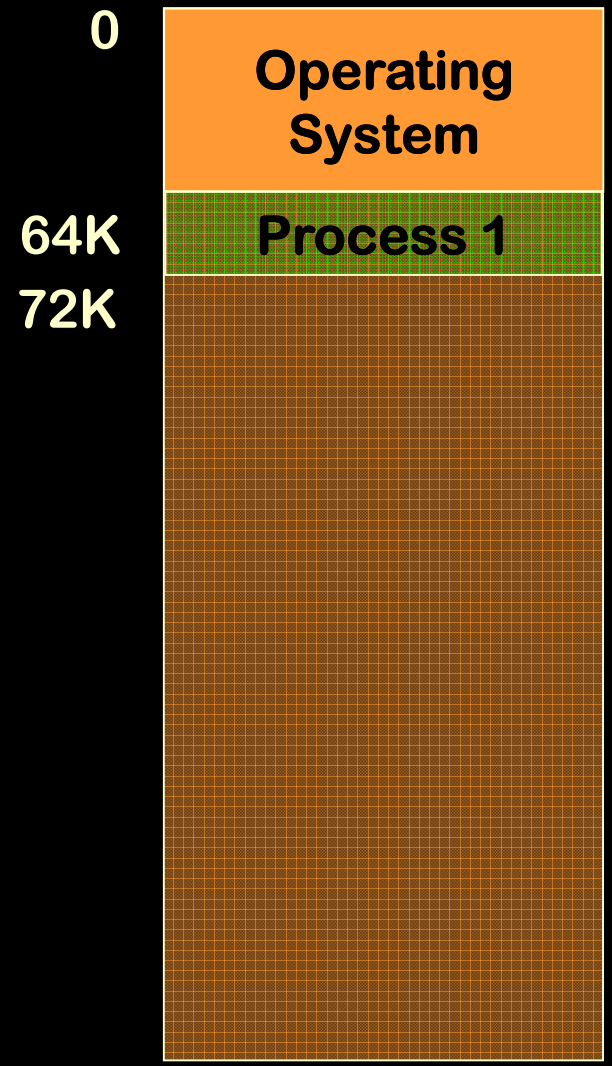
Who gets loaded?

Process 6

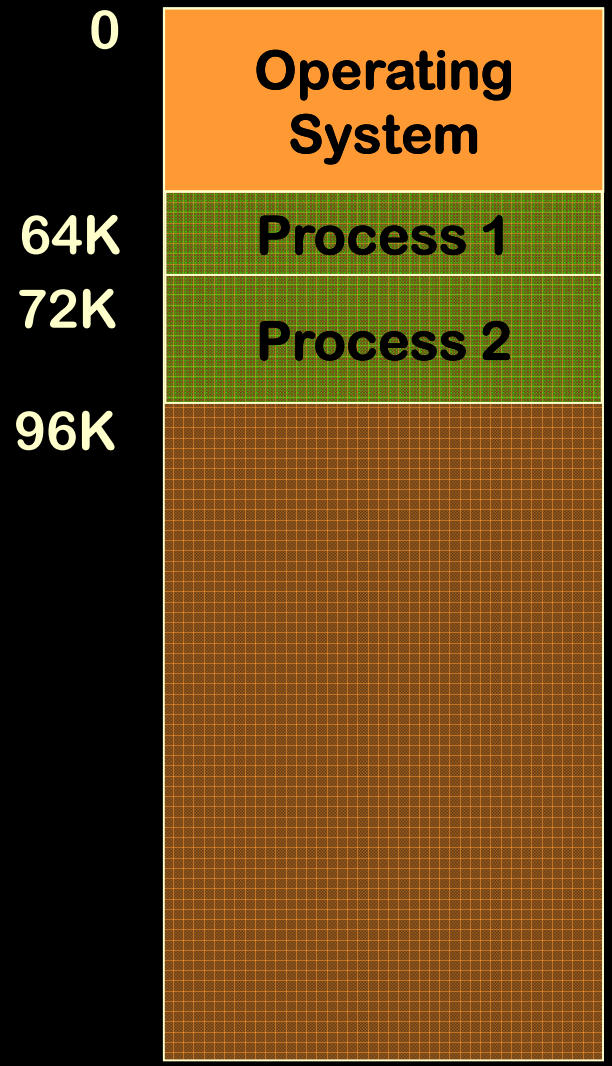
Another Big Process



Variable Partitions

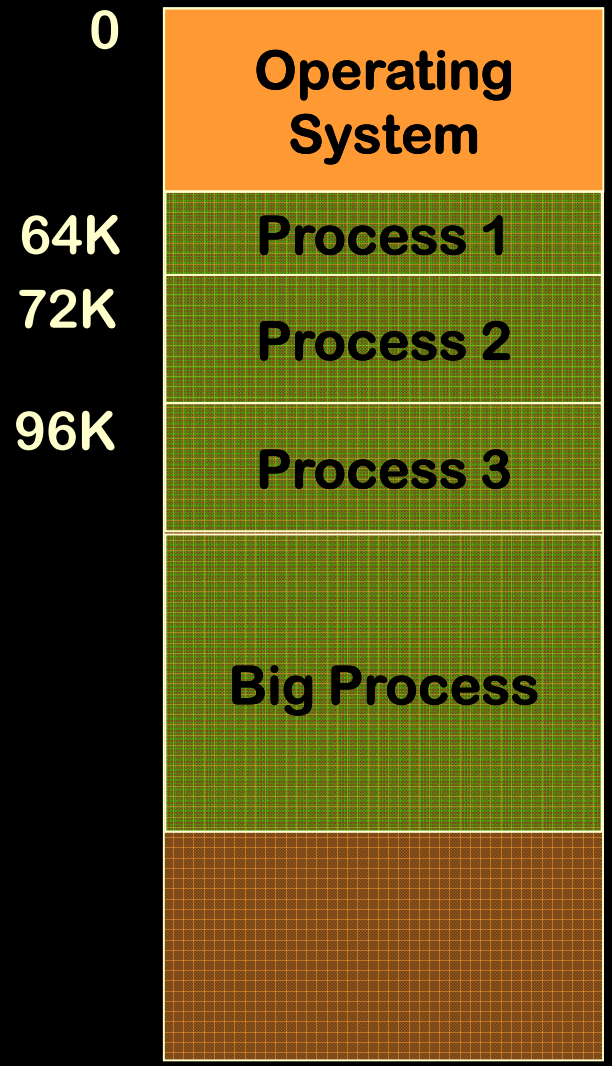


Variable Partitions



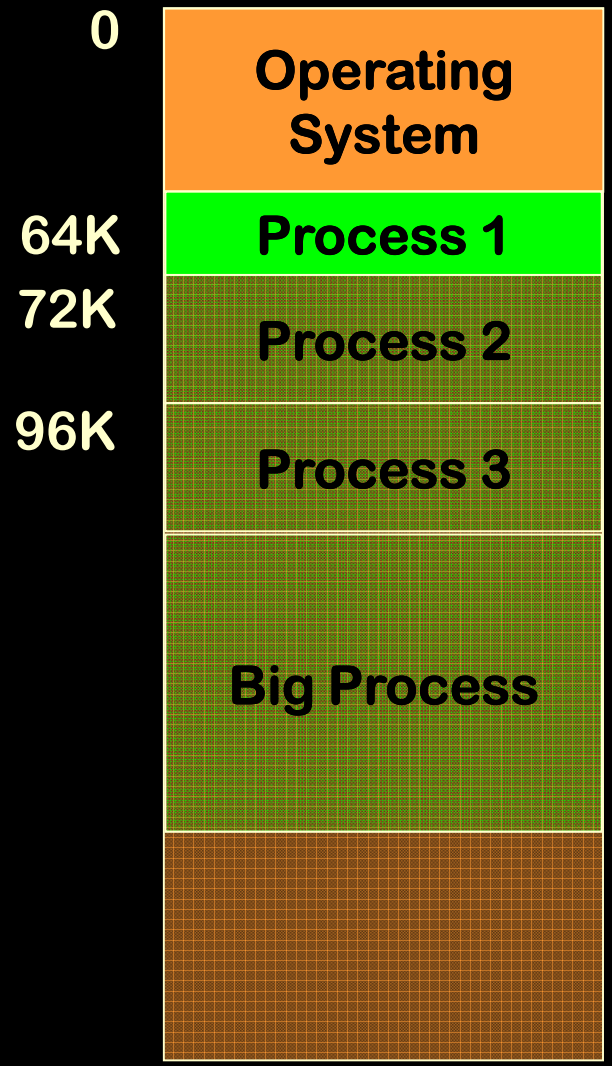
Partitions are multiples of a minimum "block size"

Variable Partitions



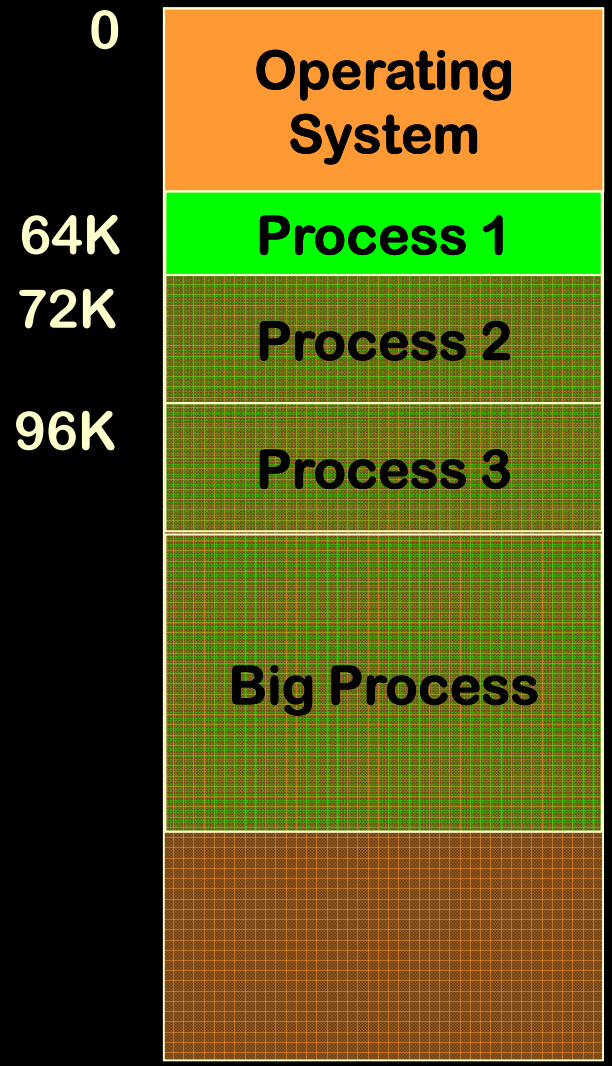
Partitions are multiples of a minimum "block size"

Variable Partitions



65536 Base Register

Variable Partitions



65536 Base Register

8192 Bounds Register

Variable Partitions

73728 Base Register

24576 Bounds Register

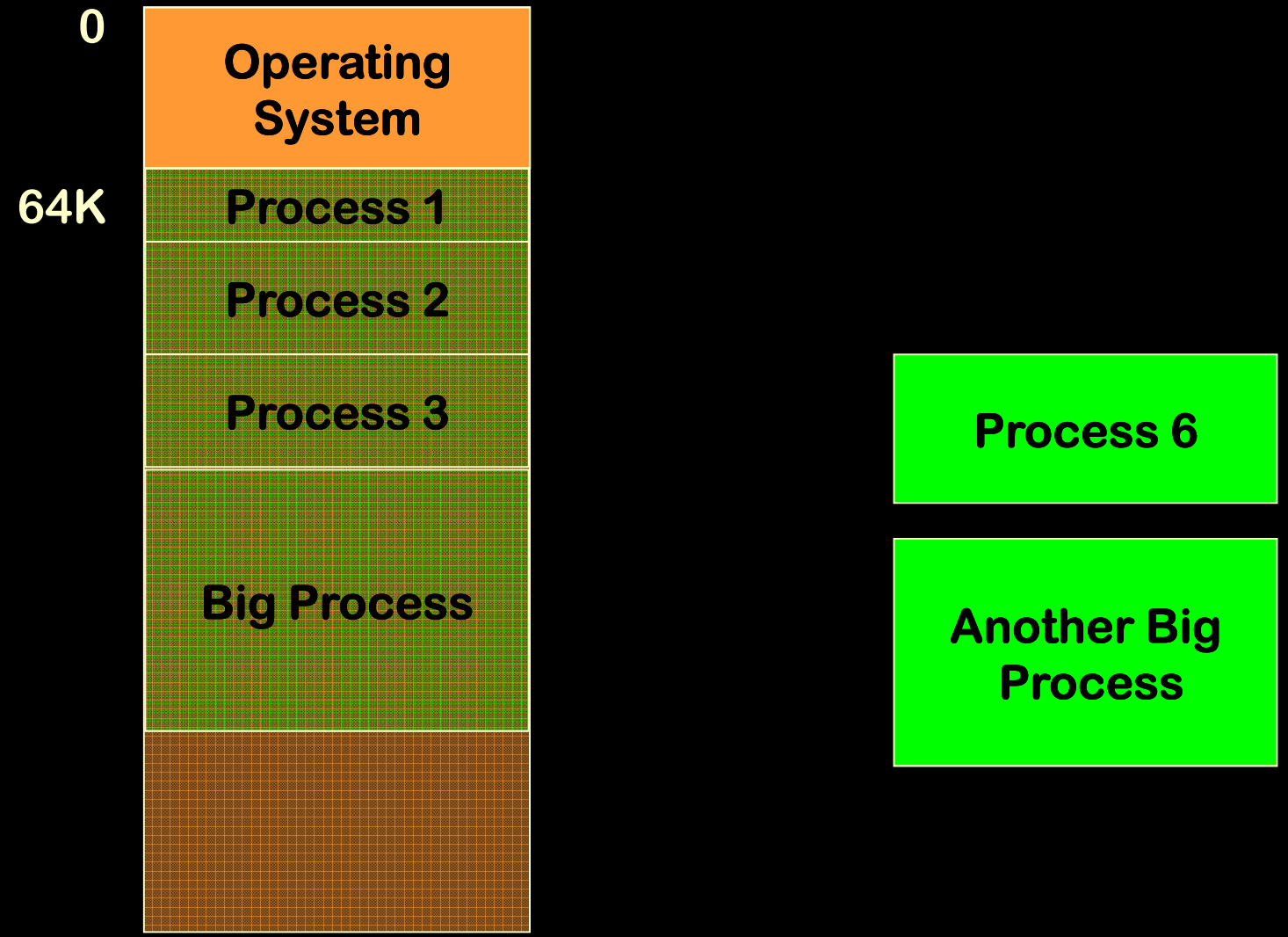
Variable Partitions

73728 Base Register

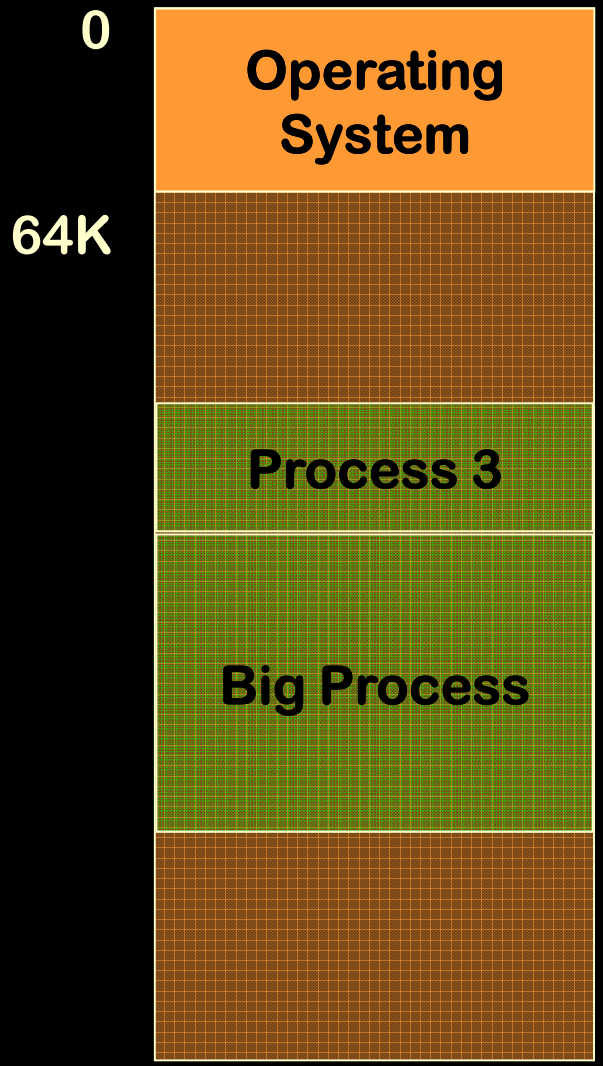
24576 Bounds Register

Process	Base	Bound
1	64K	8K
2	72K	24K
...		

Variable Partitions



Variable Partitions

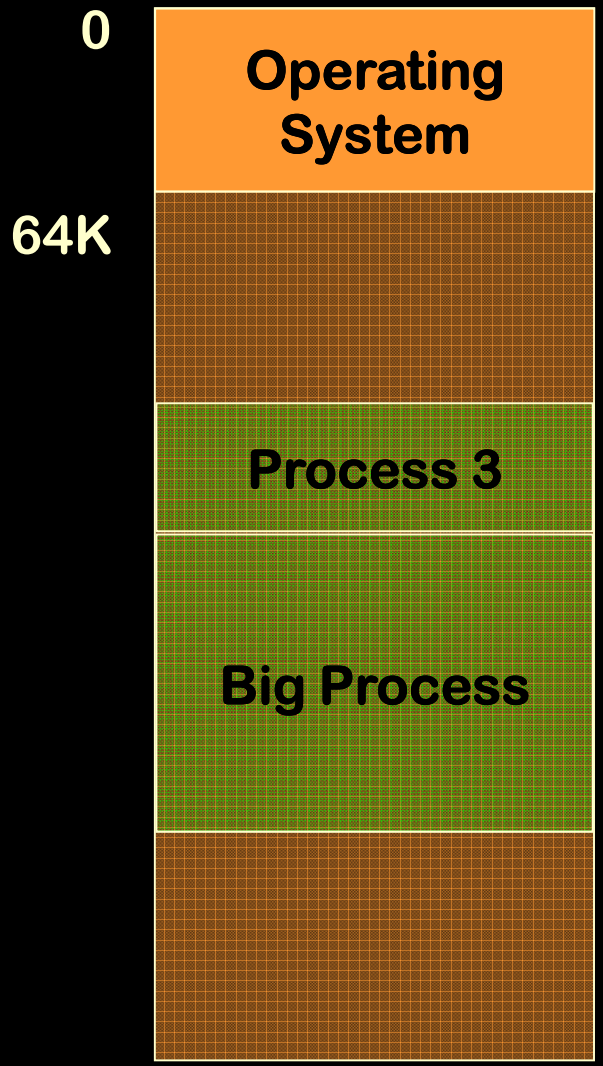


Where do I load #6?

Process 6

Another Big Process

Variable Partitions



Best fit?
First fit?
Biggest (worst fit)?

Process 6

Another Big Process

Variable Partitions

Compaction?

Another Big Process

Where to load: what's "Fair"?

- First fit
  - Simple to implement, may waste space

Where to load: what's "Fair"?

- First fit
  - Simple to implement, may waste space
- Best fit
  - May leave lots of small useless space

Where to load: what's "Fair"?

- First fit
  - Simple to implement, may waste space
- Best fit
  - May leave lots of small useless space
- Biggest (worst fit)
  - May waste even more space

Where to load: what's "Fair"?

- First fit
  - Simple to implement, may waste space
- Best fit
  - May leave lots of small useless space
- Biggest (worst fit)
  - May waste even more space
- Reserve large partitions for large processes
  - Smaller ones wait

Where to load: what's "Fair"?

- First fit
  - Simple to implement, may waste space
- Best fit
  - May leave lots of small useless space
- Biggest (worst fit)
  - May waste even more space
- Reserve large partitions for large processes
  - Smaller ones wait
- Move processes around so bigger ones will fit
  - Takes time; EVERYONE waits

Who to load: what's "Fair"?

- Load the first process you can find that fits
  - Bigger processes may wait a LONG time

Who to load: what's "Fair"?

- Load the first process you can find that fits
  - Bigger processes may wait a LONG time

Who to load: what's "Fair"?

- Load the first process you can find that fits
  - Bigger processes may wait a LONG time
- Load the first process in line as soon as it fits
  - Smaller processes wait even though they could be loaded

Who to load: what's "Fair"?

- Load the first process you can find that fits
  - Bigger processes may wait a LONG time
- Load the first process in line as soon as it fits
  - Smaller processes wait even though they could be loaded
- Reserve large partitions for large processes
  - Smaller ones wait
    - Load small one(s) if no big ones waiting

- D block

Who to load: what's "Fair"?

- Load the first process you can find that fits
  - Bigger processes may wait a LONG time
- Load the first process in line as soon as it fits
  - Smaller processes wait even though they could be loaded
- Reserve large partitions for large processes
  - Smaller ones wait
    - Load small one(s) if no big ones waiting
- Reserve small partitions for small processes
  - Bigger ones wait even if room available
    - Load big one(s) if no small ones waiting

Summary

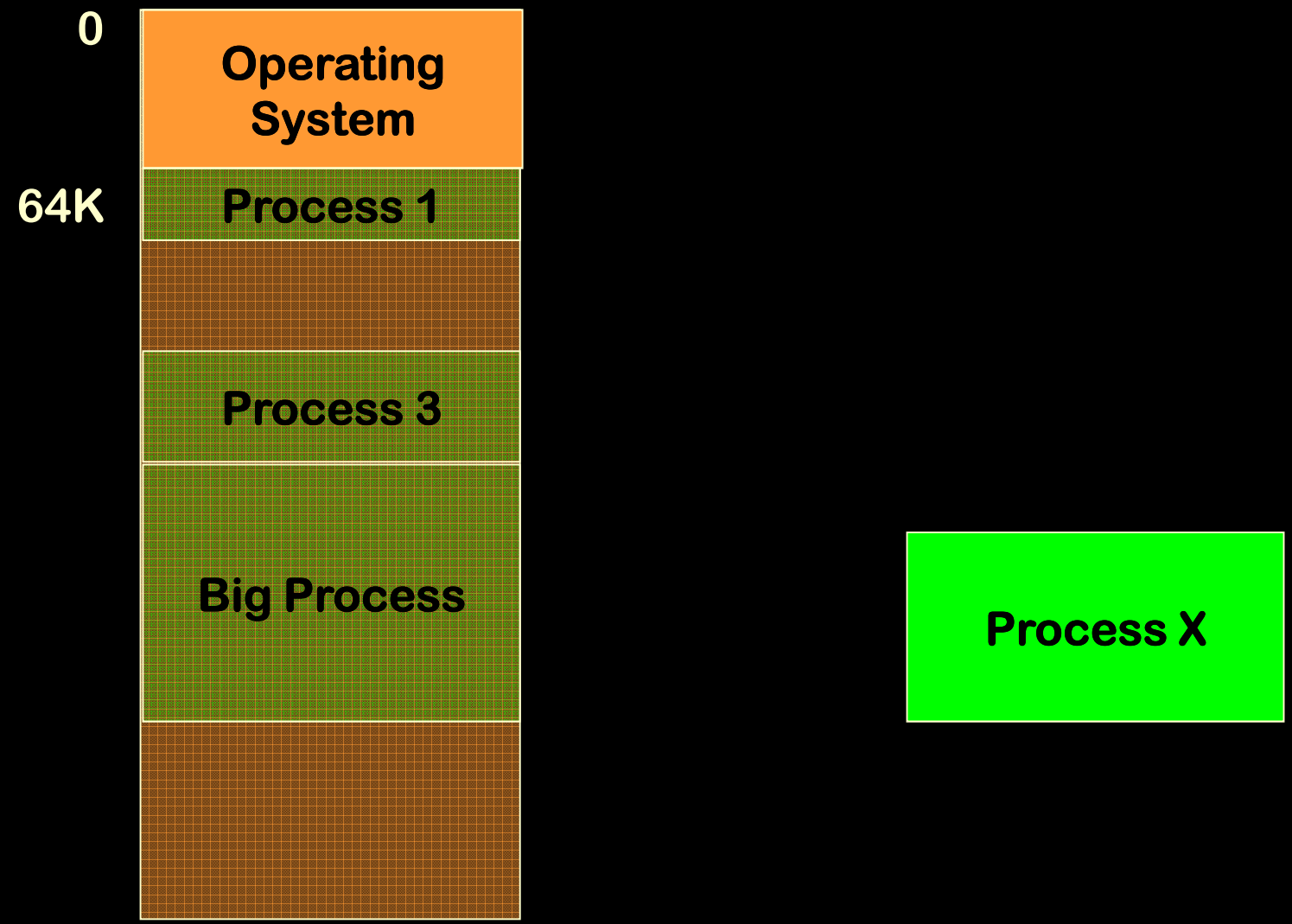
Scheme	Features	Costs
One Partition	OS and program separate	• Base register to map “logical” address to physical address

Summary

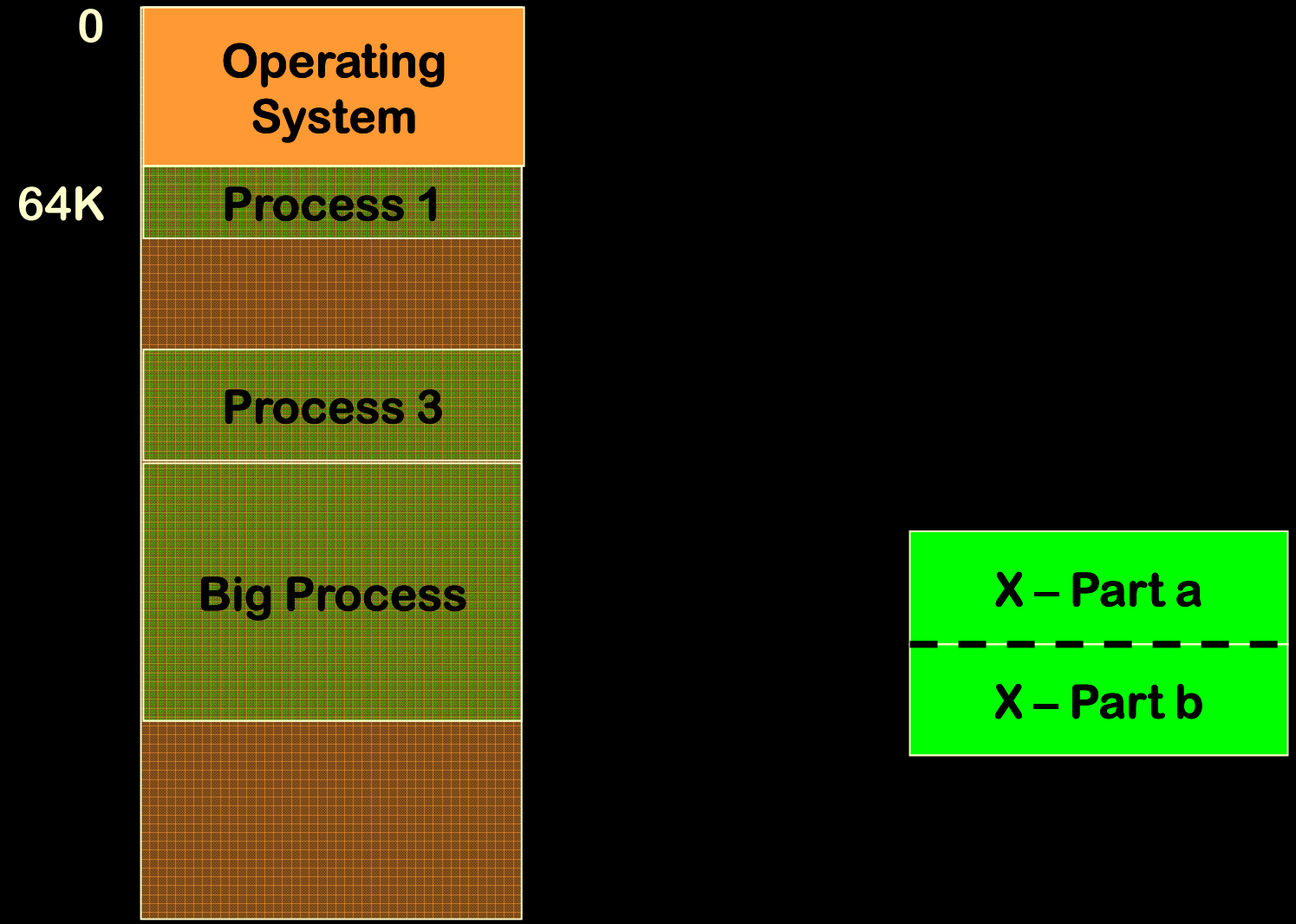
Scheme	Features	Costs
One Partition	OS and program separate	• Base register to map “logical” address to physical address
Fixed Partitions	Multiple processes	• Available memory may not be in one place

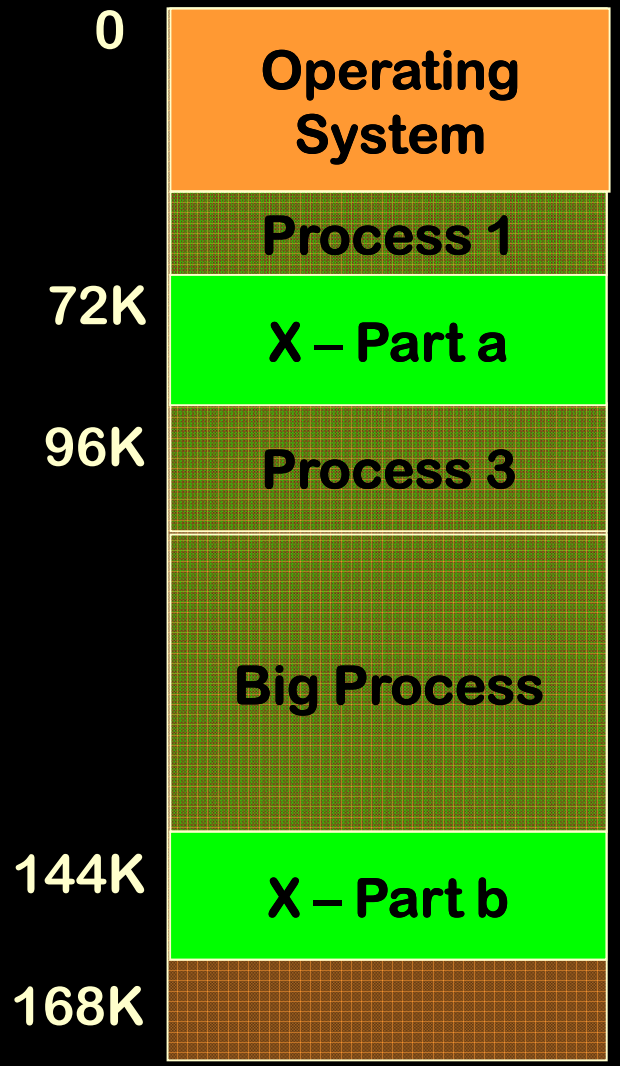
Summary

Scheme	Features	Costs
One Partition	OS and program separate	• Base register to map “logical” address to physical address
Fixed Partitions	Multiple processes	• Available memory may not be in one place
Variable Partitions	More efficient use of space	• Bounds register


0
Operating System
64K
Process 1
Process 3
Big Process
Process X

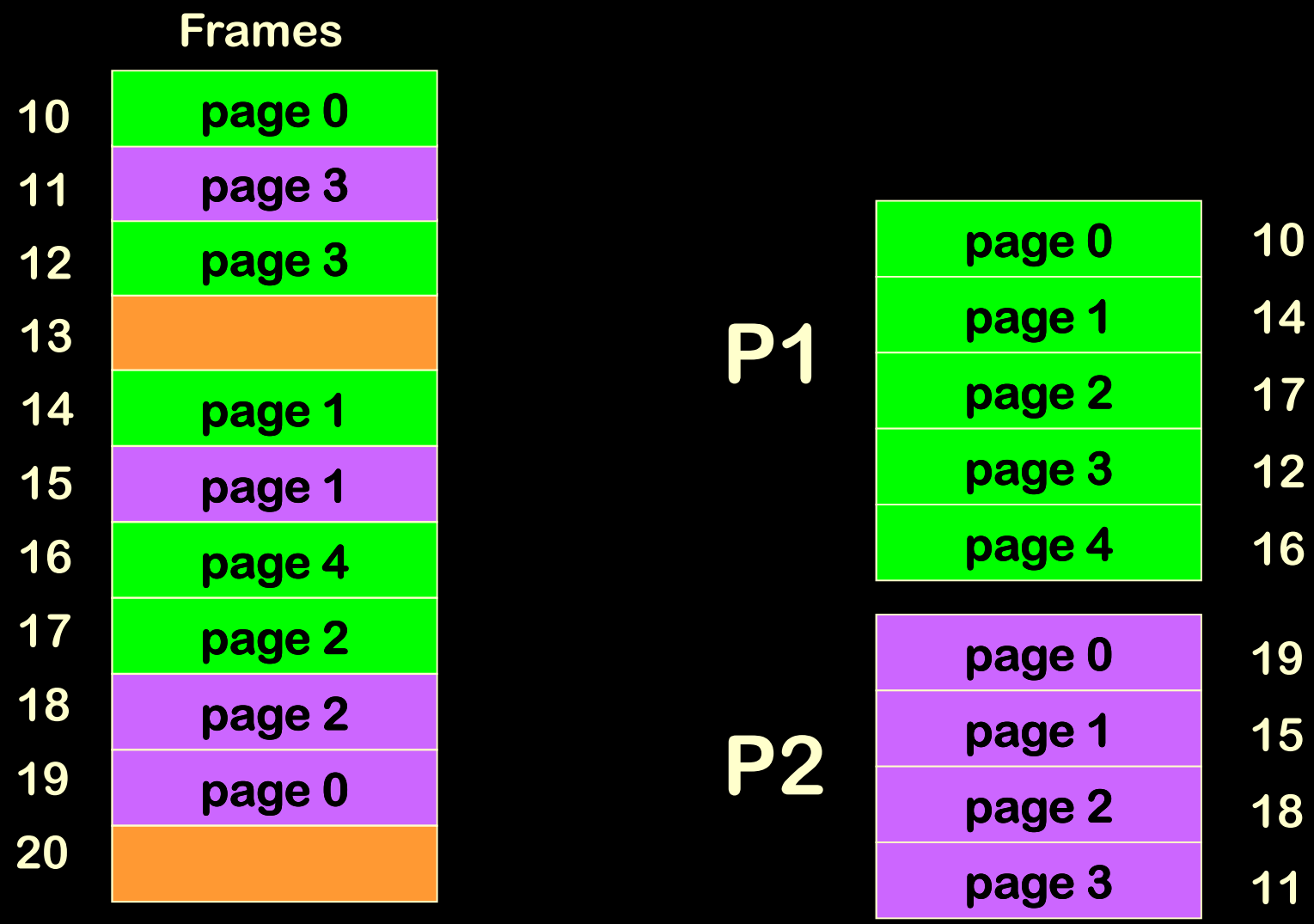
Split 'em up!


0
Operating System
64K
Process 1
Process 3
Big Process
X – Part a
X – Part b

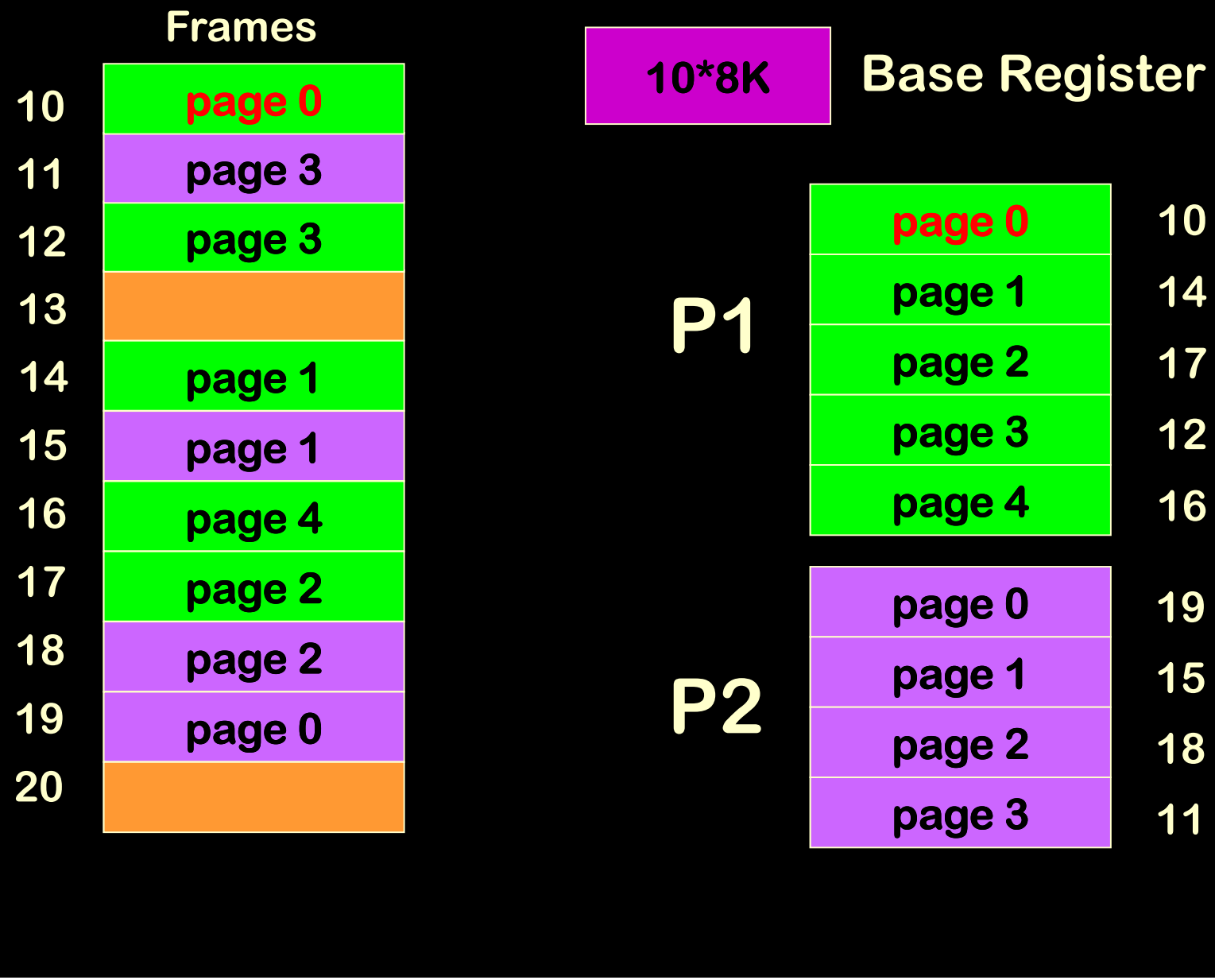


Logical Address	Physical Address
0	72K
8K	80K
16K	88K
23K	95K
24K	144K
32K	152K
40K	160K
47K	167K
48K	168K

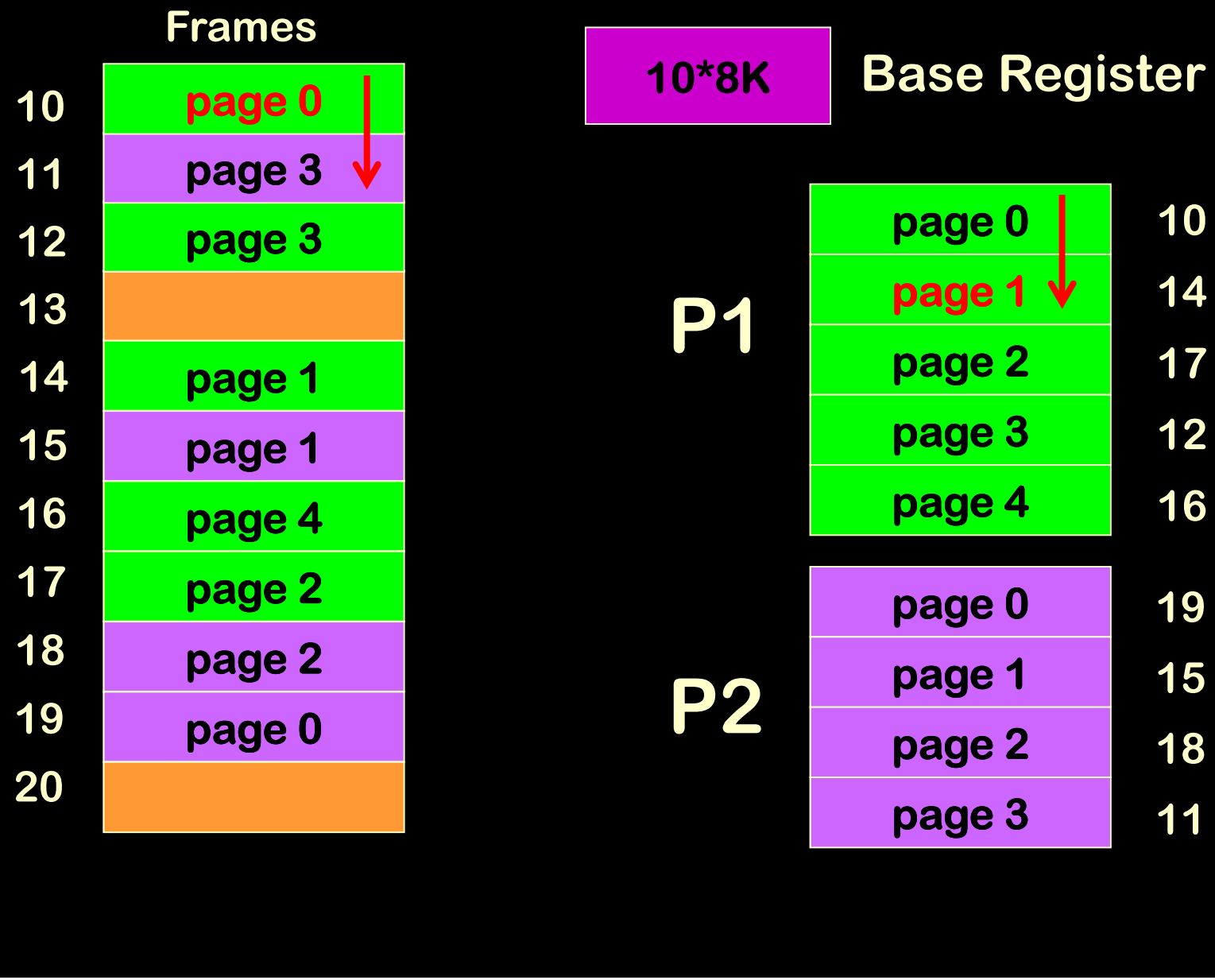
Paged Memory Management



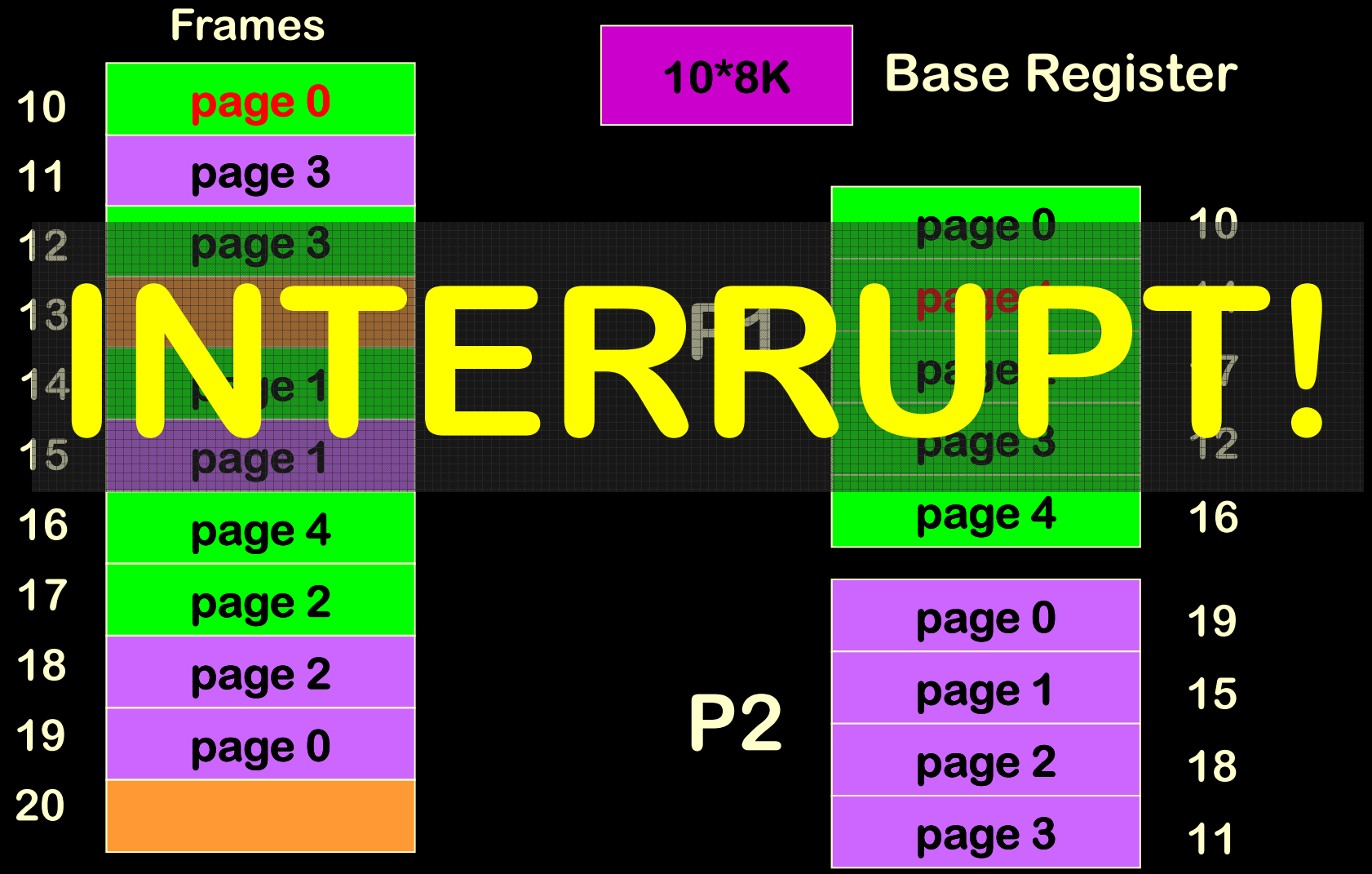
Paged Memory Management



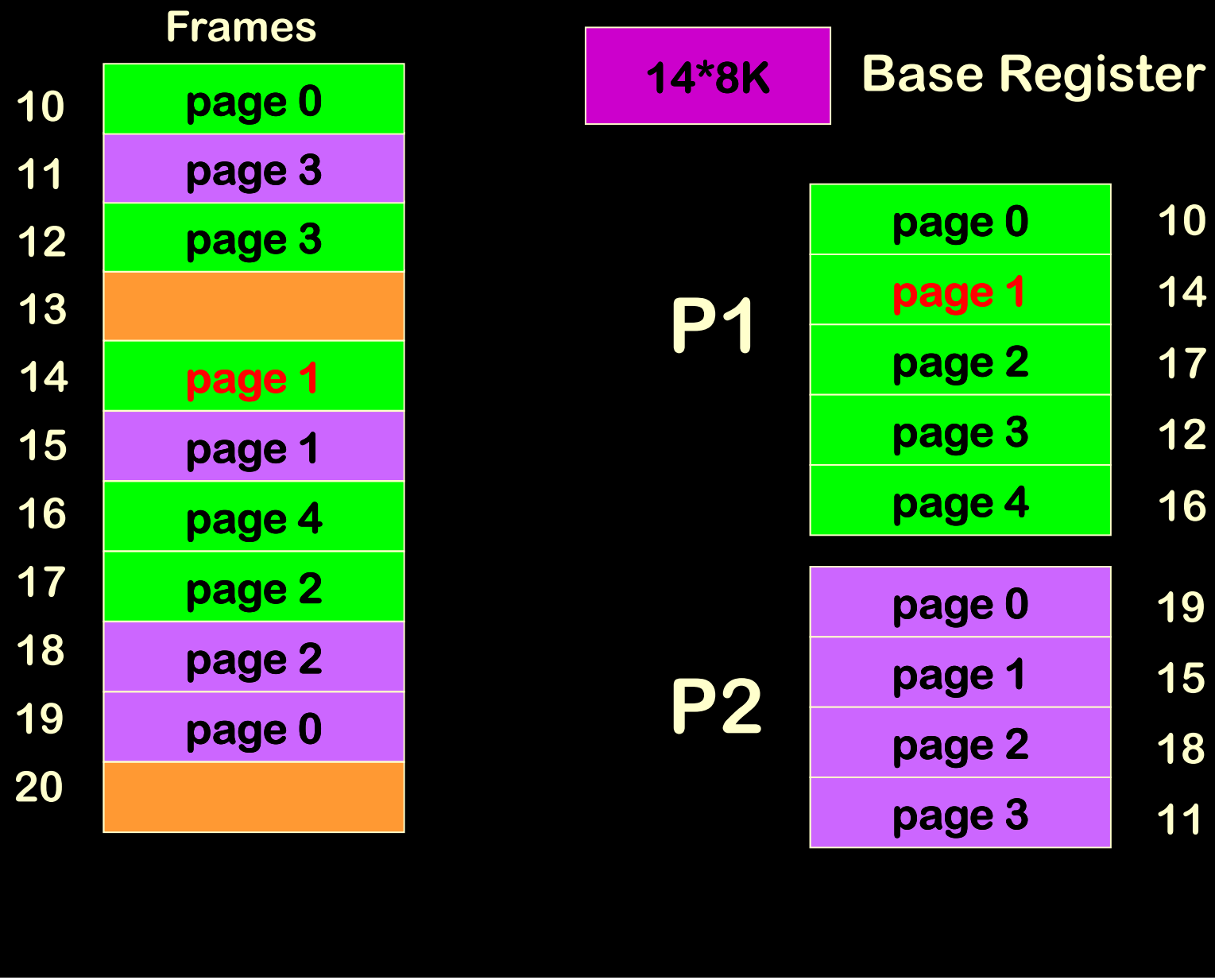
Paged Memory Management



Paged Memory Management

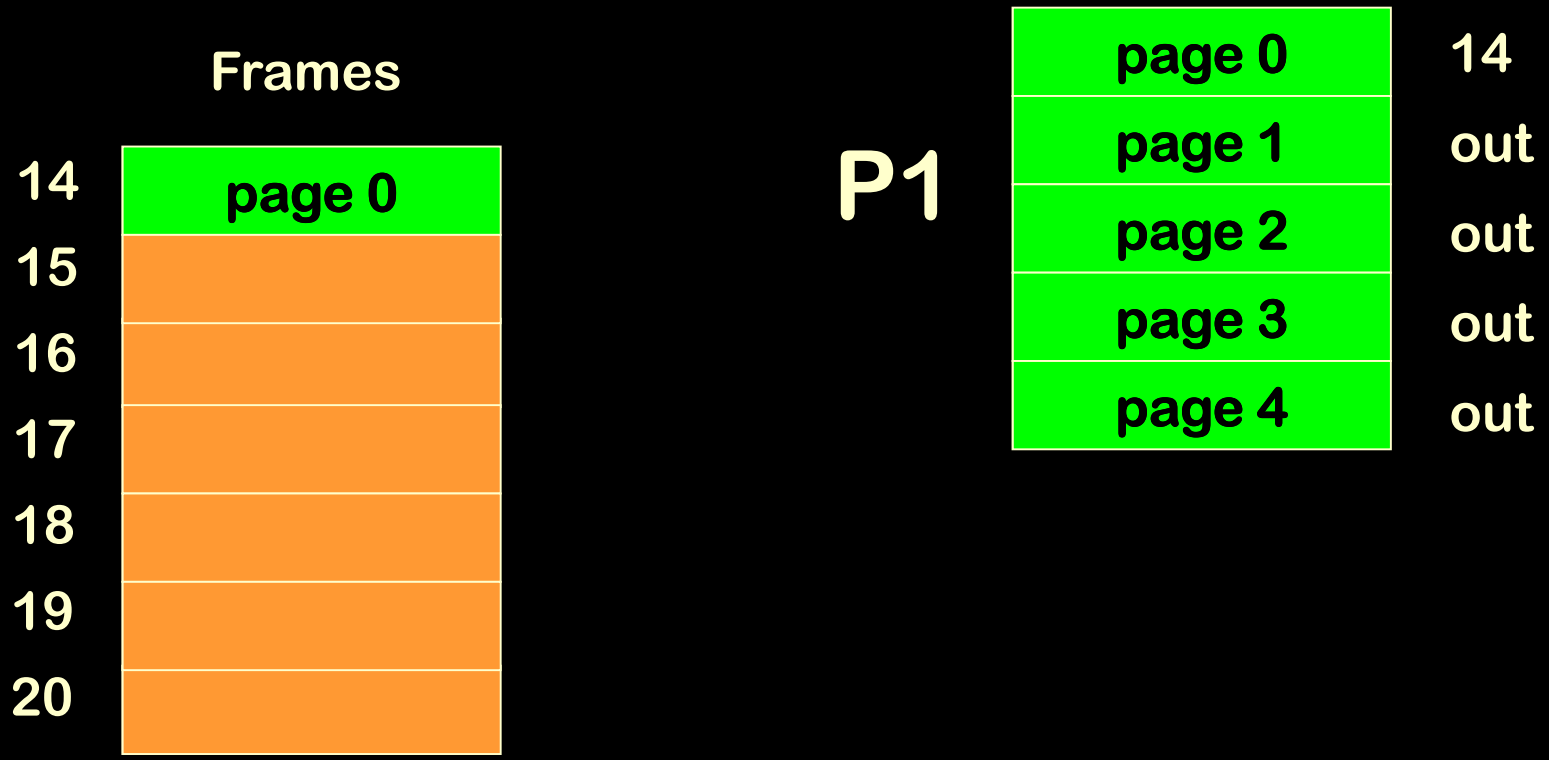


Paged Memory Management



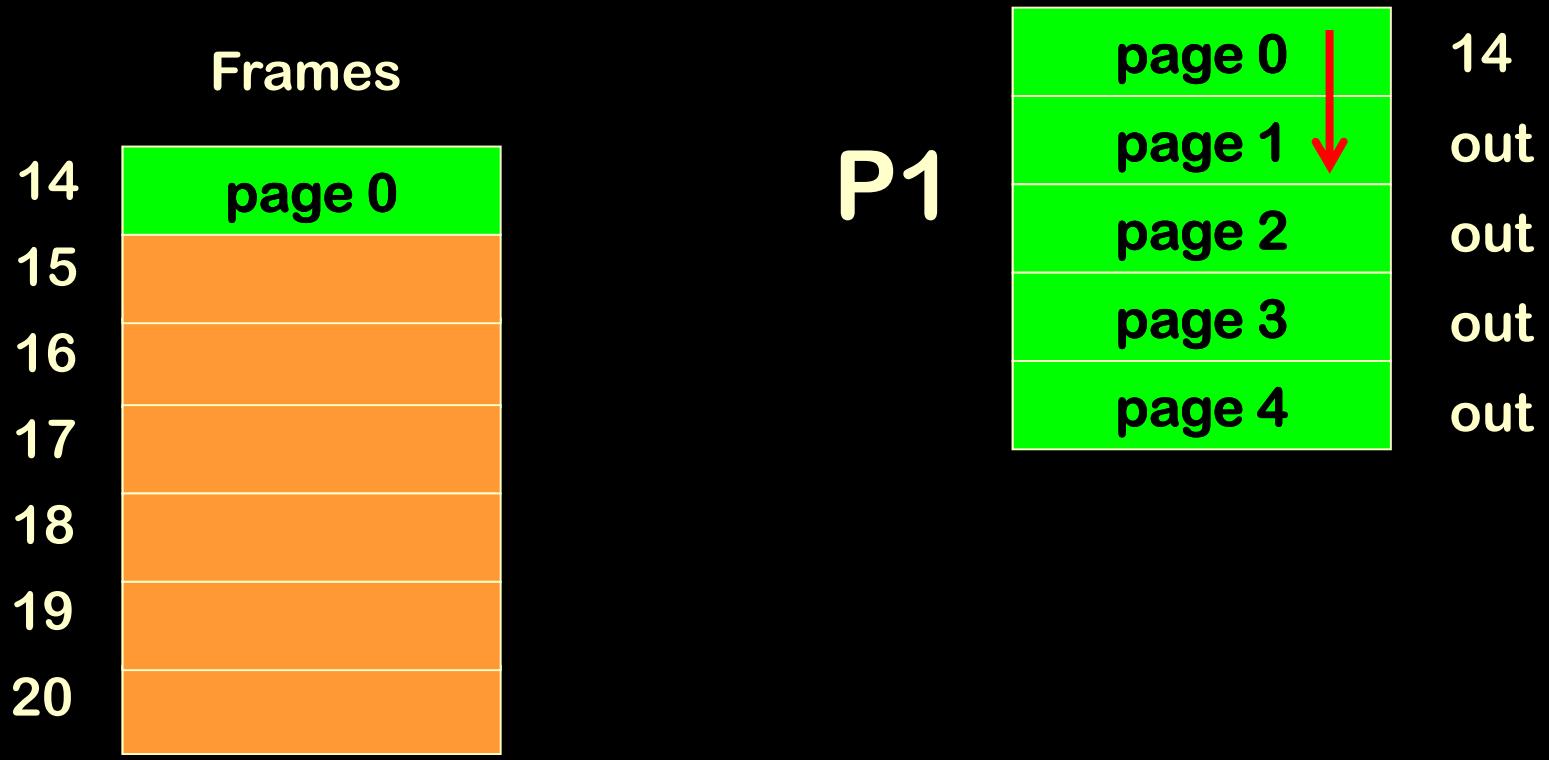
Demand Paging

- Bring in pages as needed



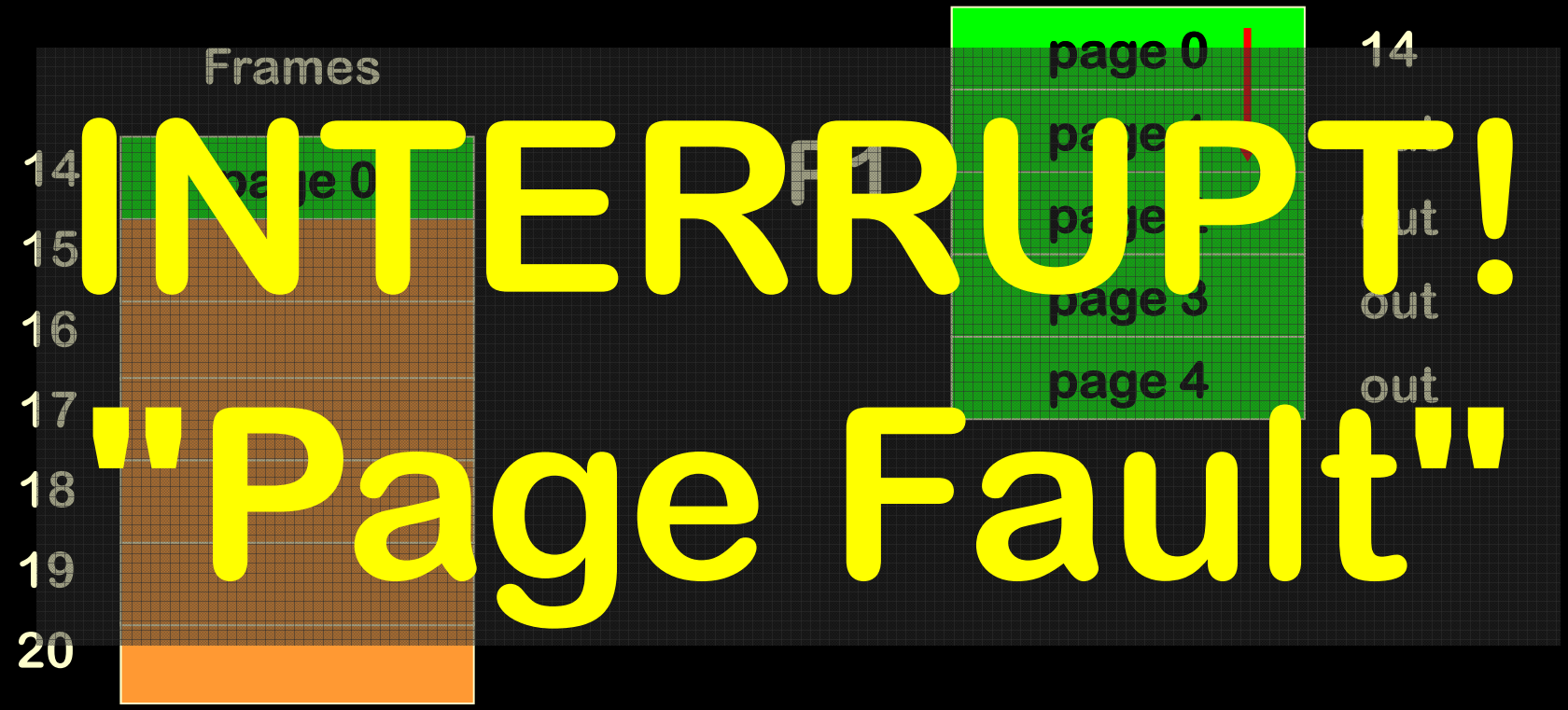
Demand Paging

- Bring in pages as needed



Demand Paging

- Bring in pages as needed

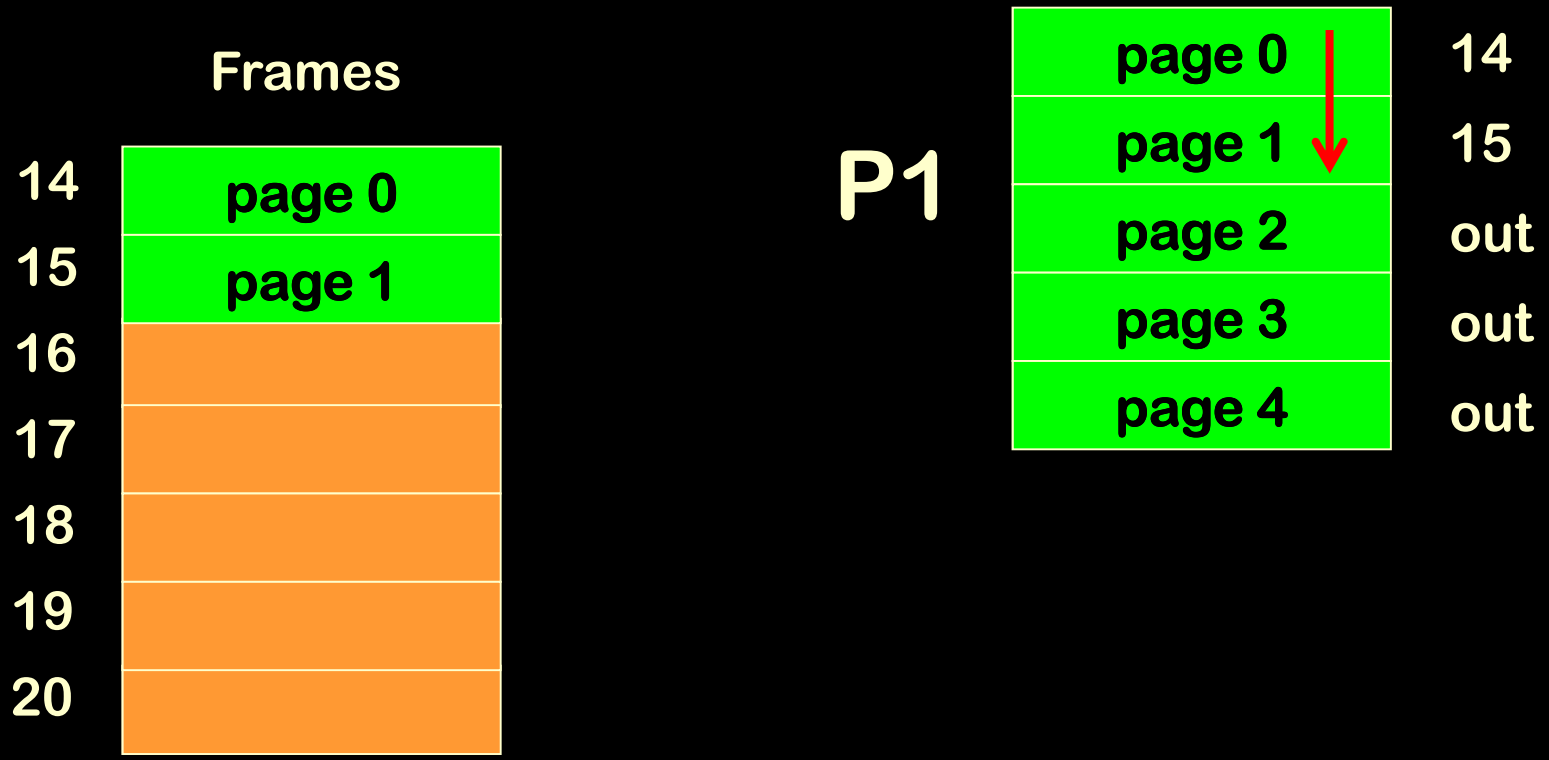


Page Fault

- Page needed for a process isn't currently in memory
- Read it from disk into a free frame

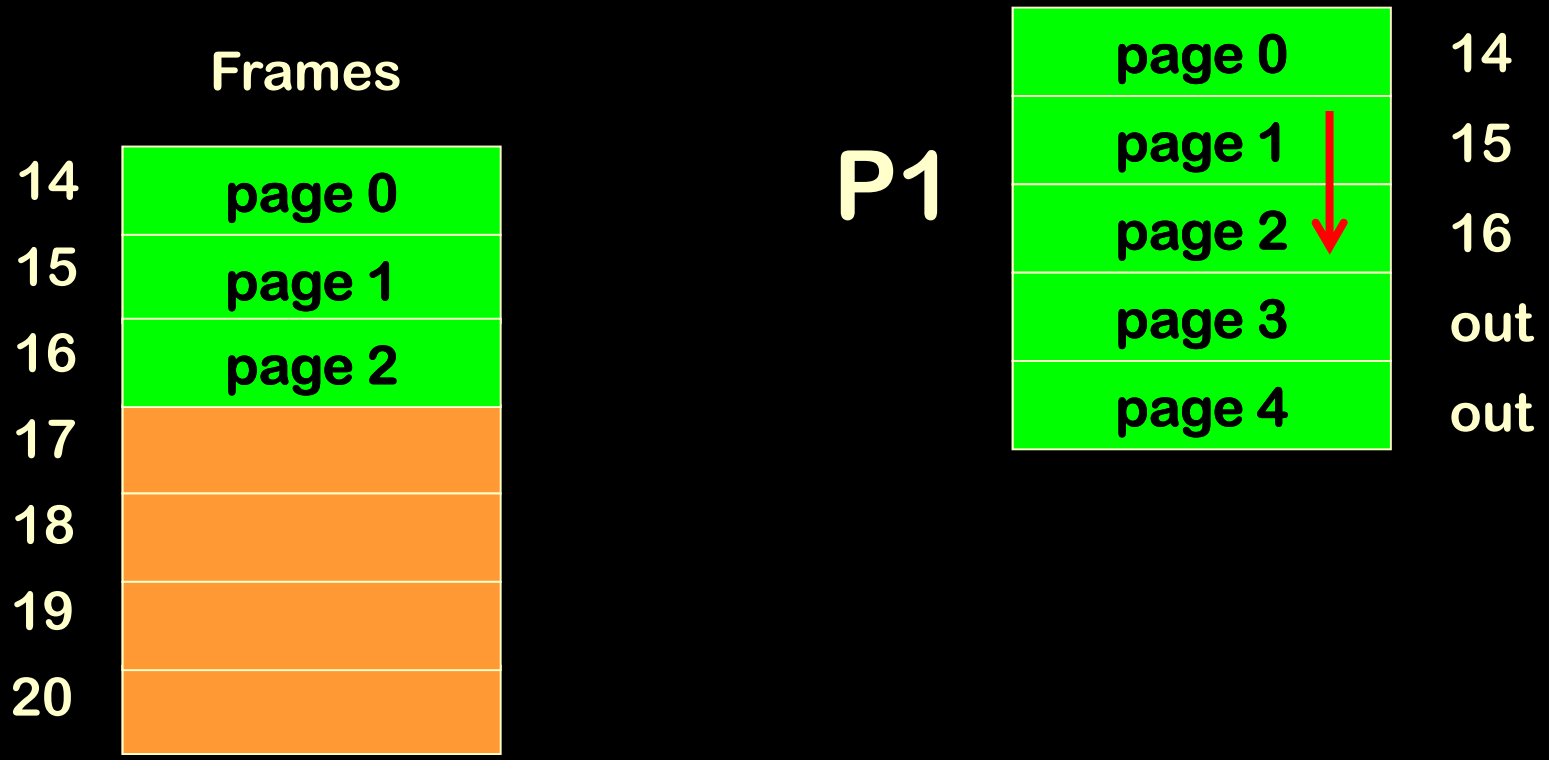
Demand Paging

- Bring in pages as needed



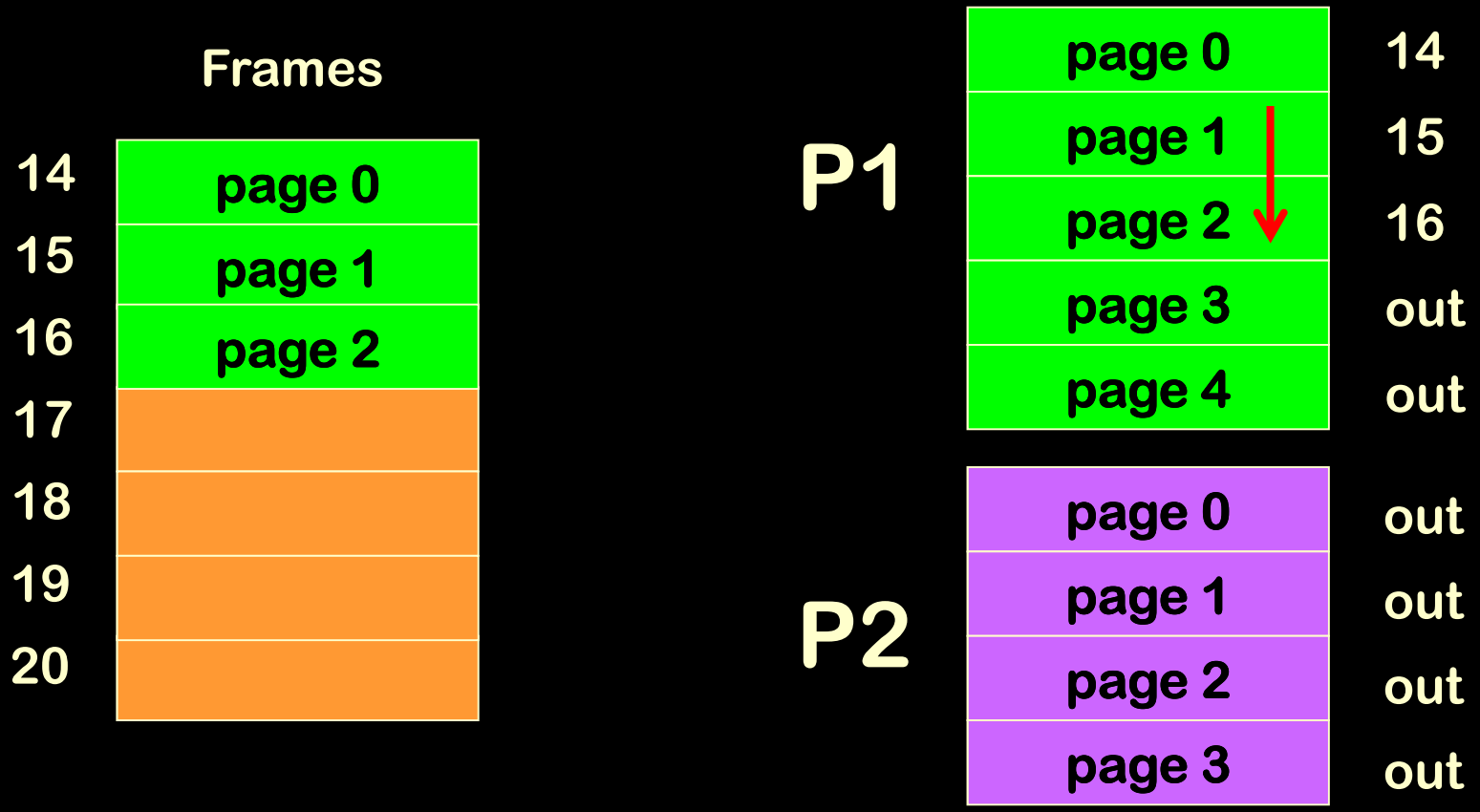
Demand Paging

- Bring in pages as needed



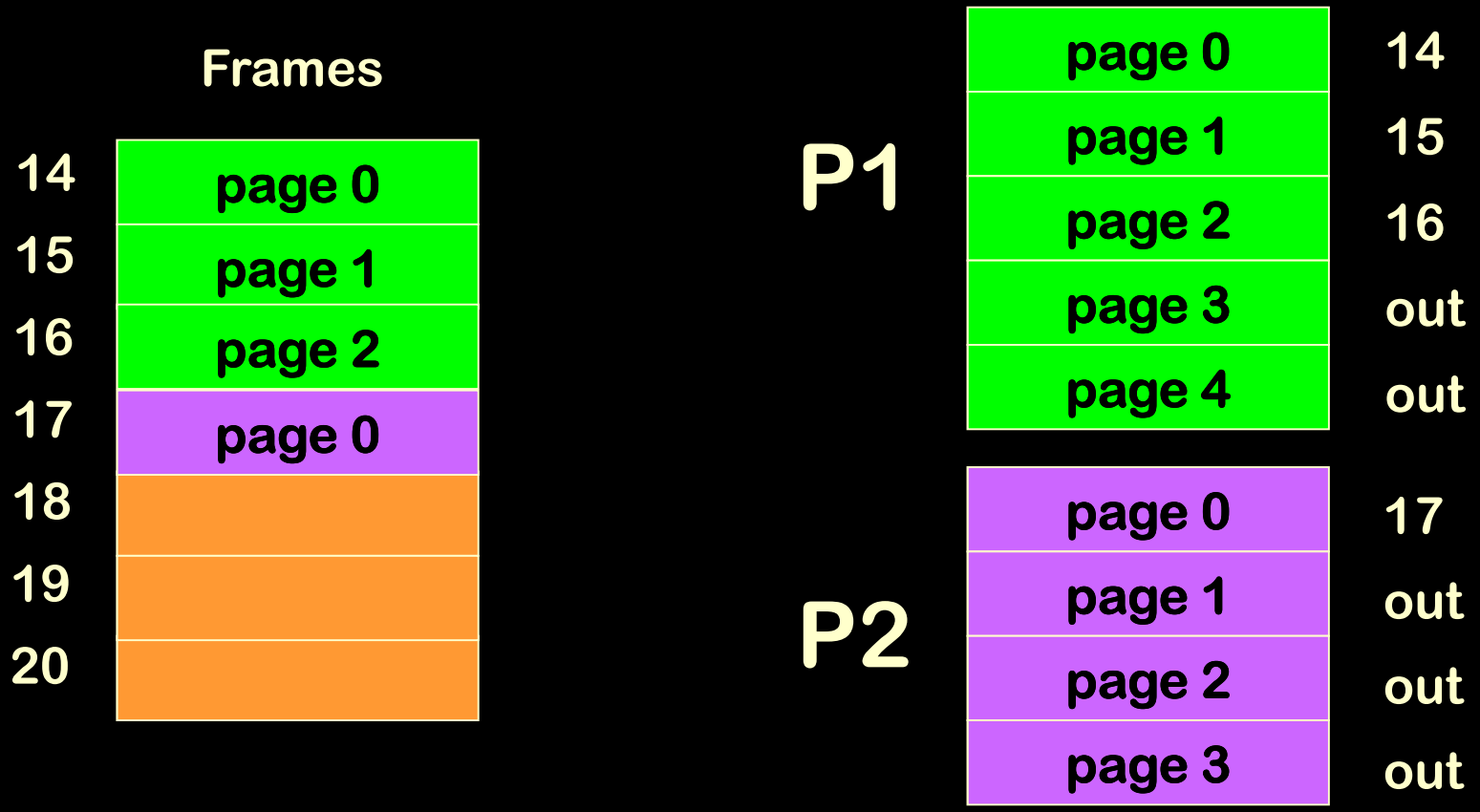
Demand Paging

- Bring in pages as needed



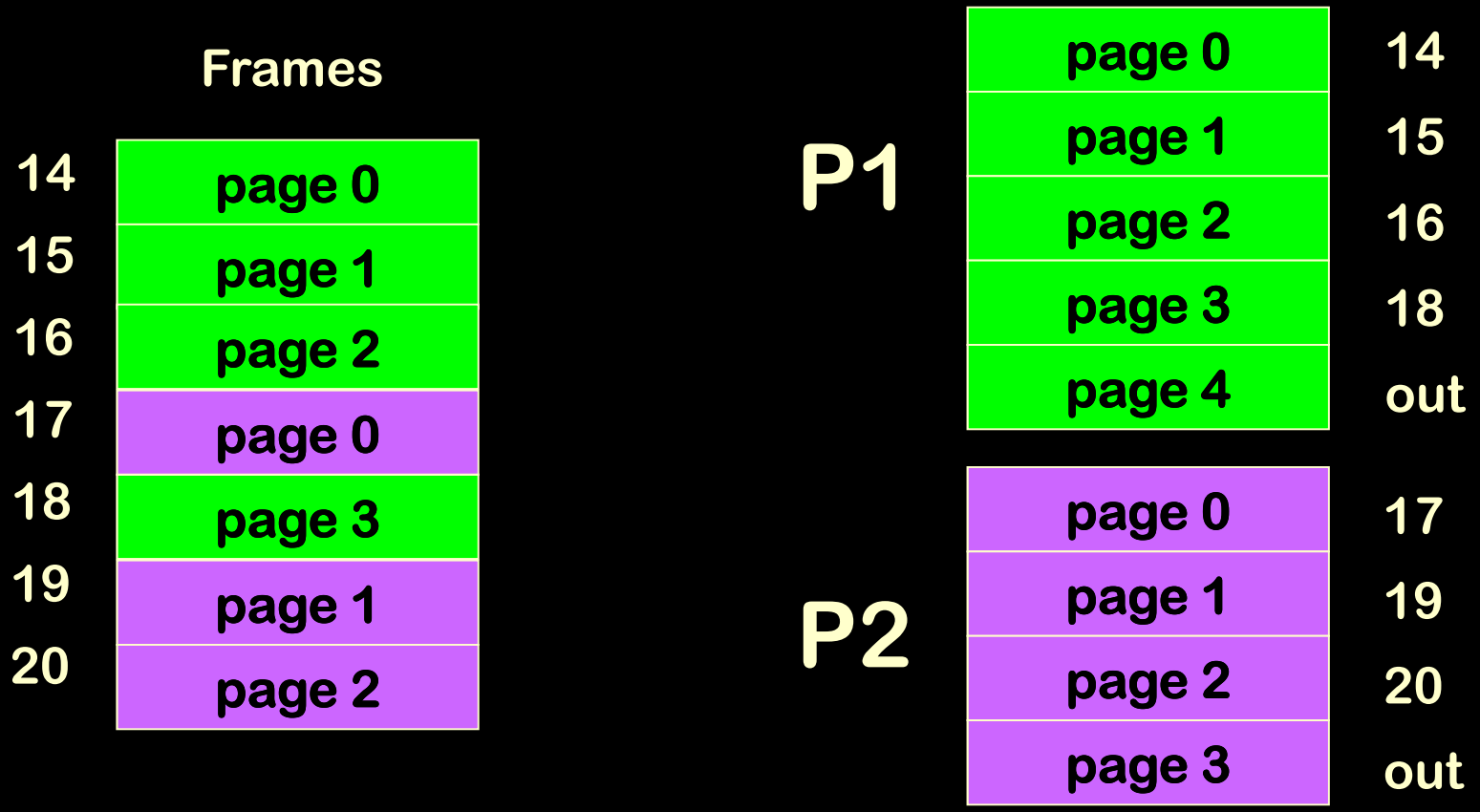
Virtual Memory

- Not all of a process needs to be in memory at the same time



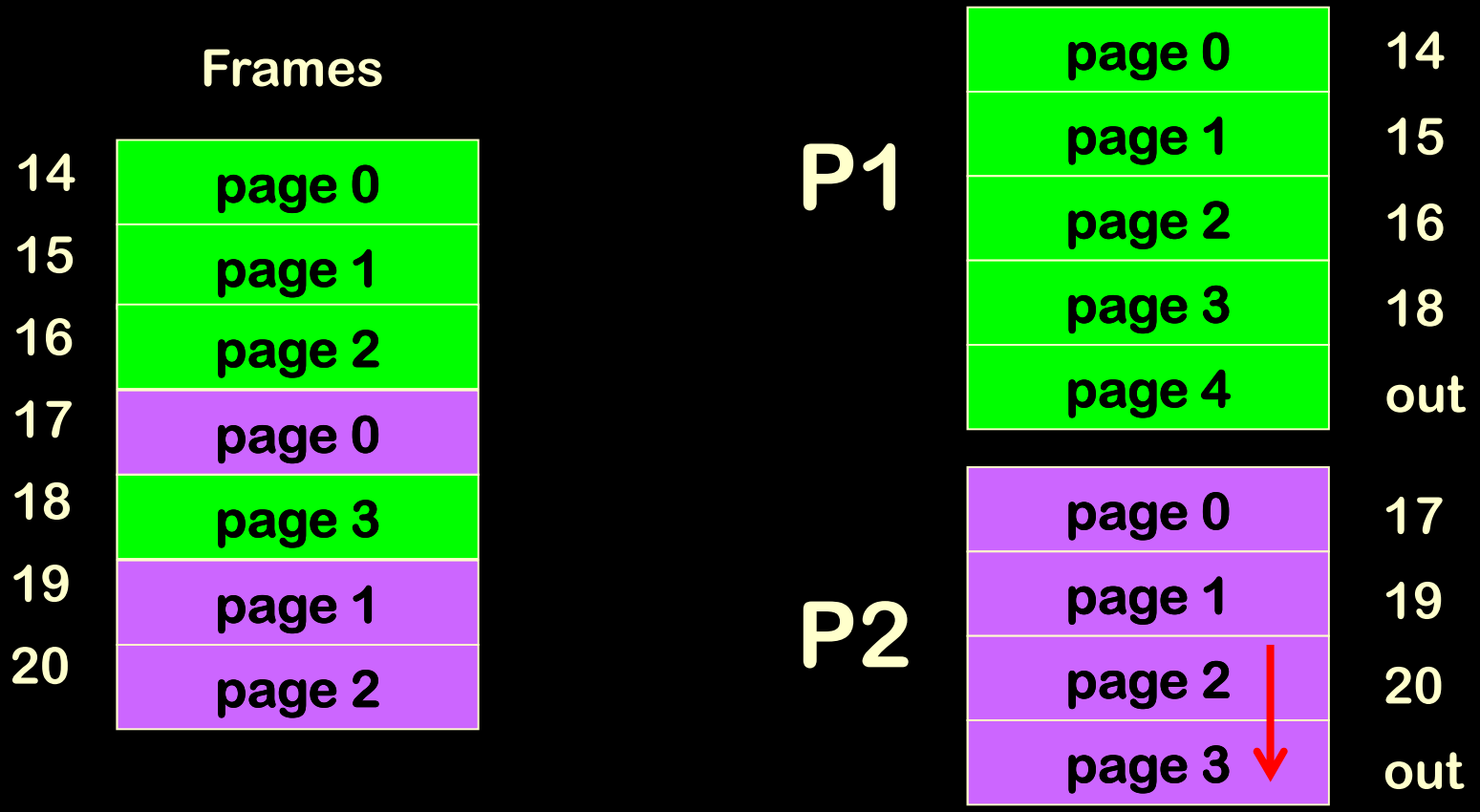
Virtual Memory

- Not all of a process needs to be in memory at the same time



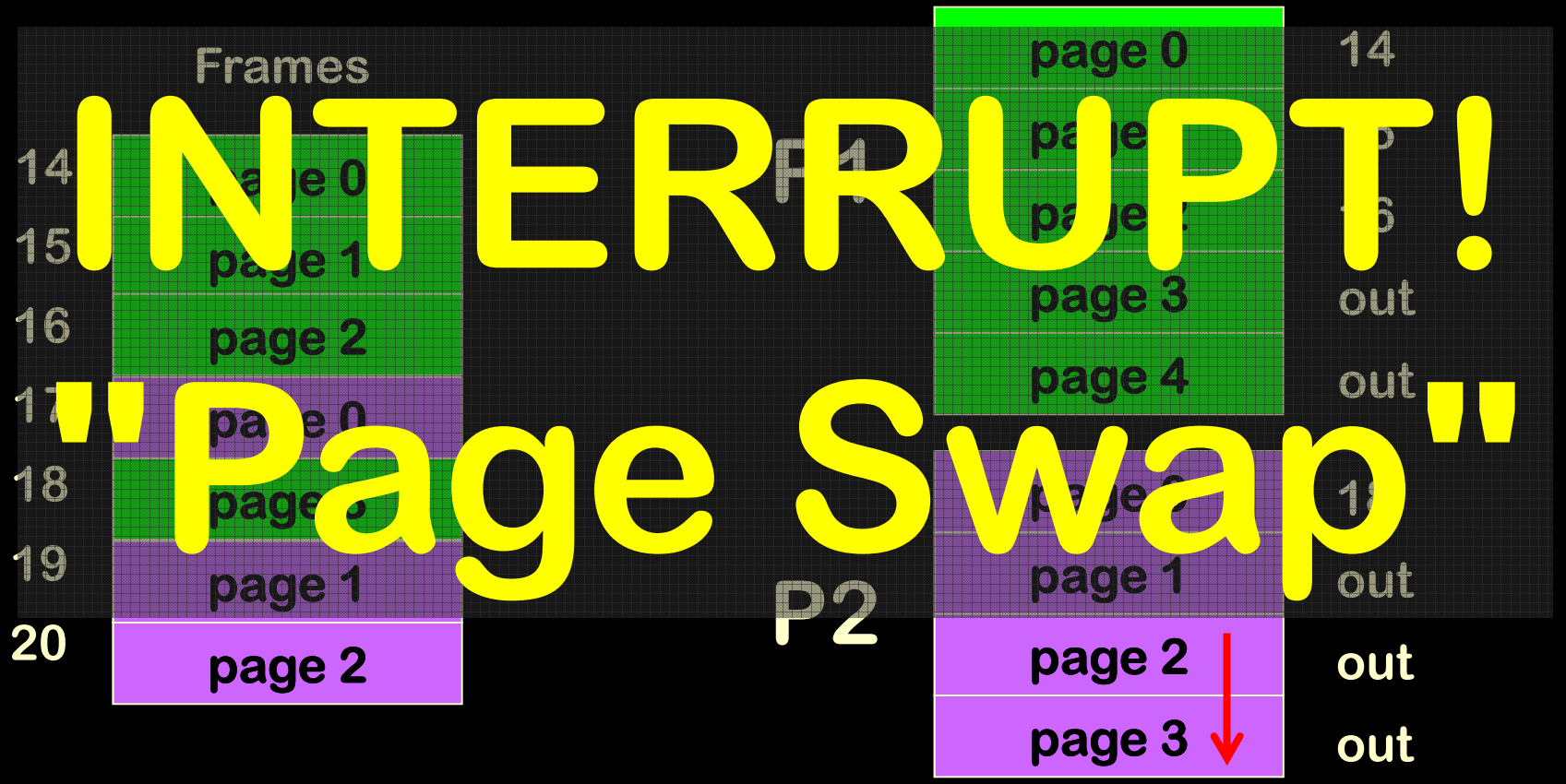
Virtual Memory

- Not all of a process needs to be in memory at the same time



Virtual Memory

- Not all of a process needs to be in memory at the same time



Page Swap

- Page needed for a process isn't in memory, and there's no free frame
- In general, can't just wait!
- Which page should we replace?

Page Swap – what's "Fair"?

Page Swap – what's "Fair"?

- Random

Page Swap – what's "Fair"?

- Random
- First in, First Out (FIFO)
  - Need to keep track when loaded

Page Swap – what's "Fair"?

- Random
- First in, First Out (FIFO)
  - Need to keep track when loaded
- Least Recently Used (LRU)
  - Need to keep track when used

Page Swap – what's "Fair"?

- Random
- First in, First Out (FIFO)
  - Need to keep track when loaded
- Least Recently Used (LRU)
  - Need to keep track when used
- Clean, then Dirty
  - "Dirty" page has been changed

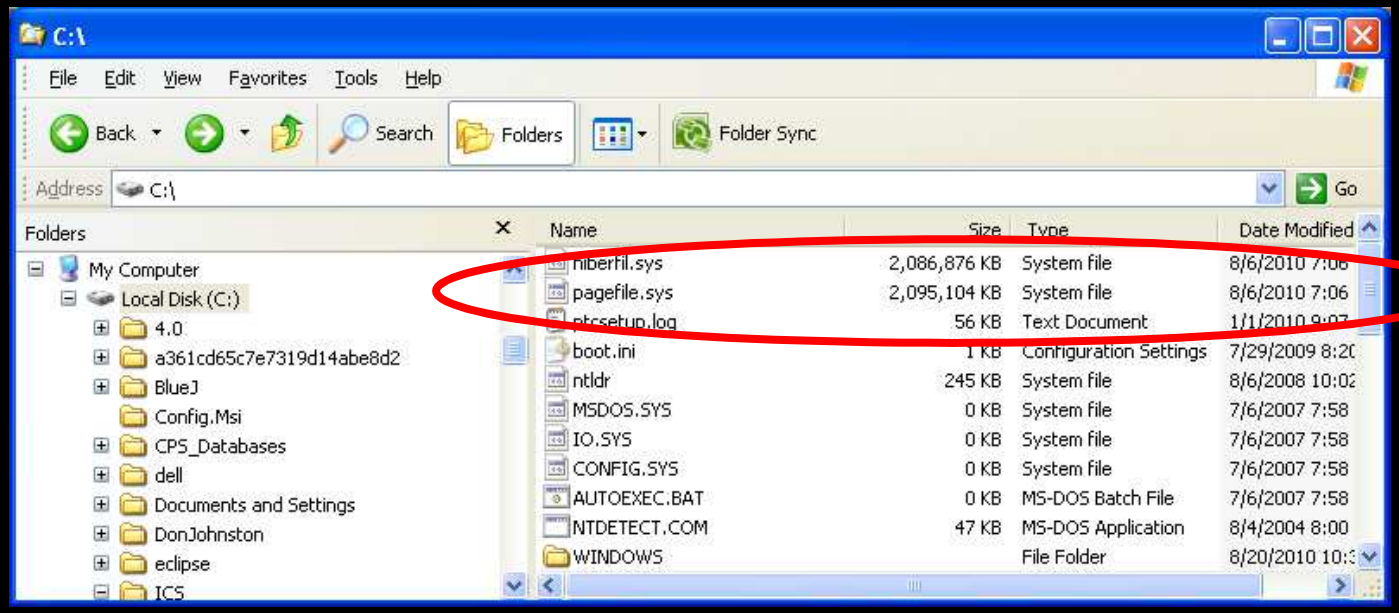
$$x = x + 1$$

Virtual Memory

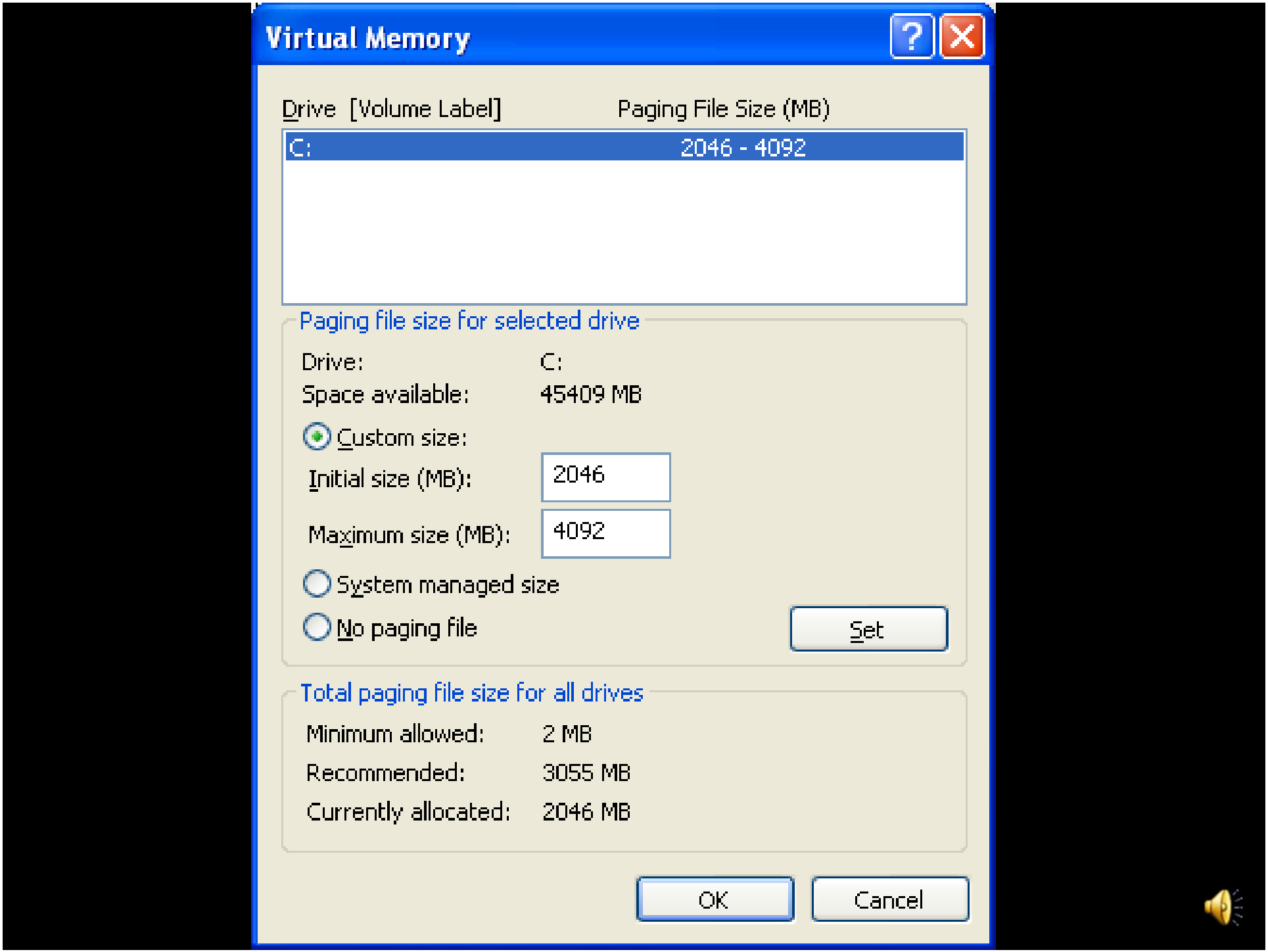
- Run programs that are bigger than the physical memory available

Virtual Memory

- Run programs that are bigger than the physical memory available
- Store out-of-memory pages in a "page file" with priority access on disk

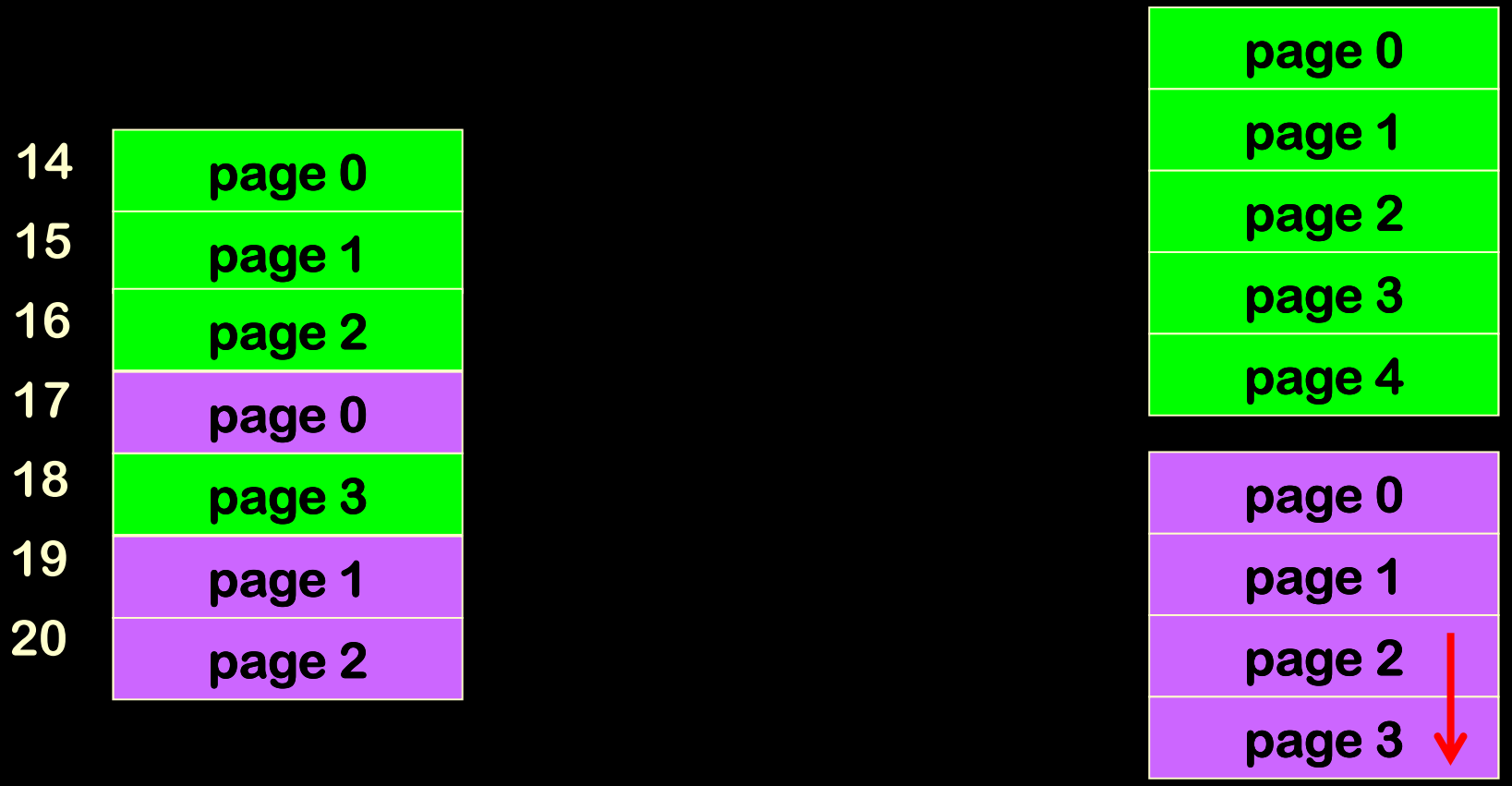


- H block, E block


Virtual Memory
Drive [Volume Label]
Paging File Size (MB)
C:
2046 - 4092
Paging file size for selected drive
Drive:
C:
Space available:
45409 MB
Custom size:
Initial size (MB):
2046
Maximum size (MB):
4092
System managed size
No paging file
Set
Total paging file size for all drives
Minimum allowed:
2 MB
Recommended:
3055 MB
Currently allocated:
2046 MB
OK
Cancel

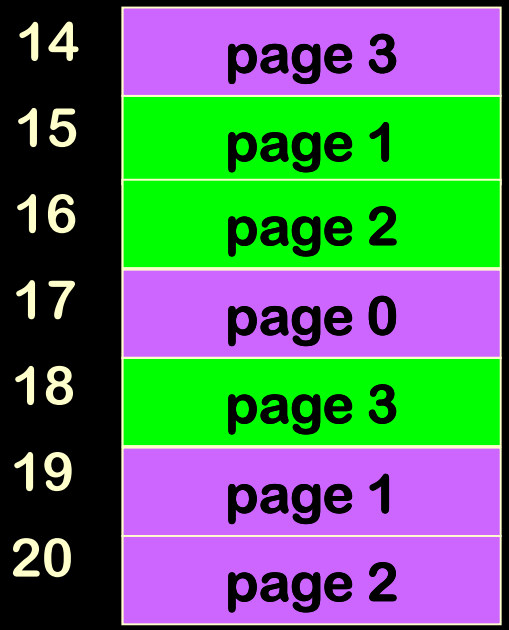
Thrashing

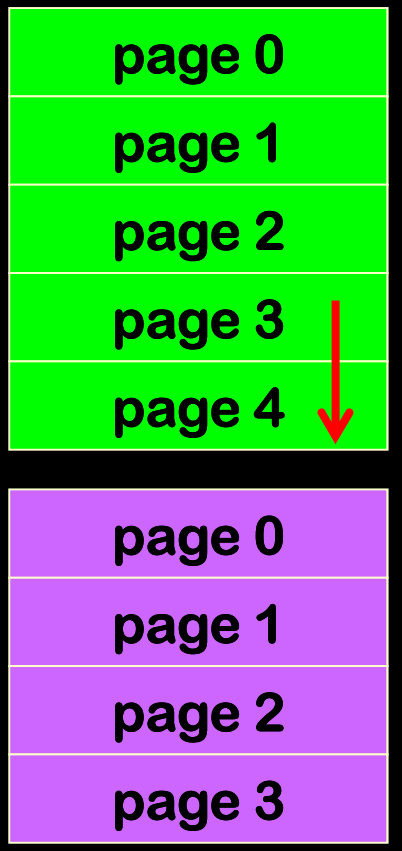
- Too frequent Page Swaps



Thrashing

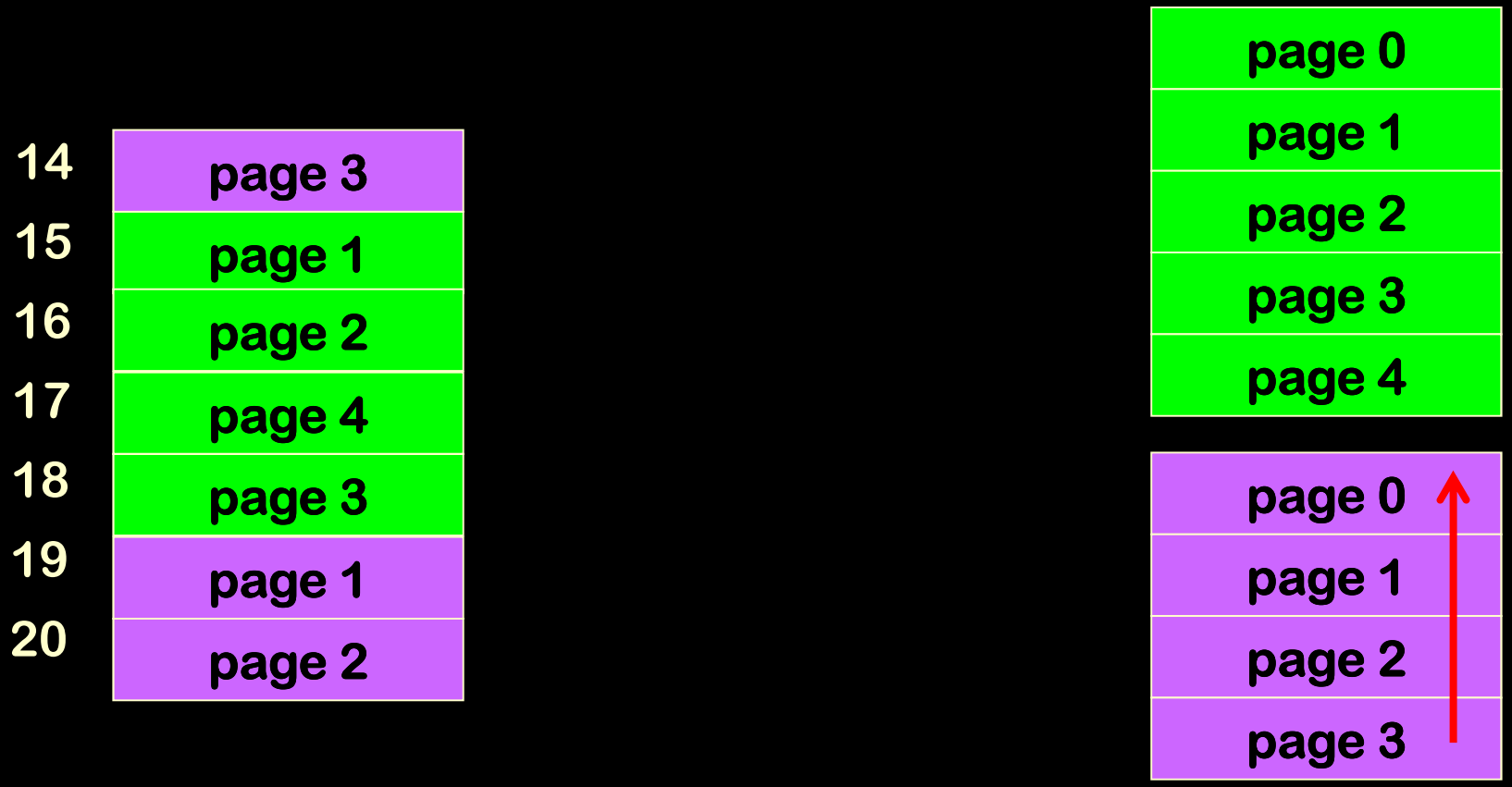
- Too frequent Page Swaps





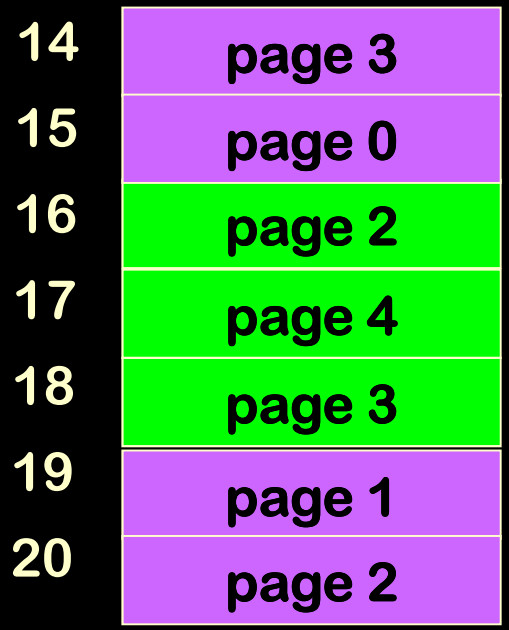
Thrashing

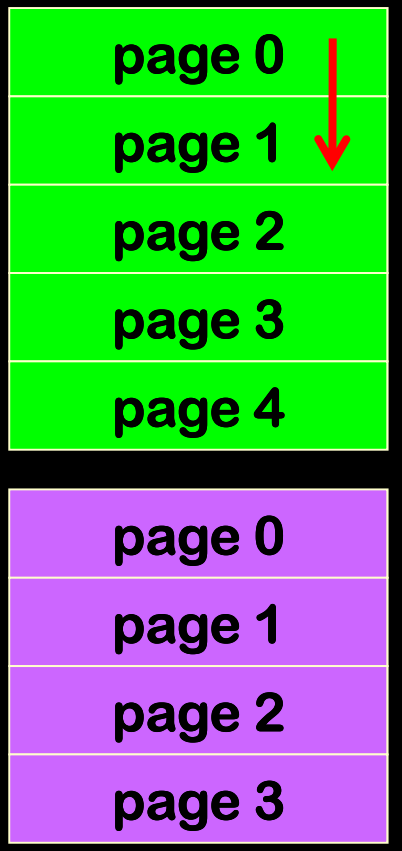
- Too frequent Page Swaps



Thrashing

- Too frequent Page Swaps

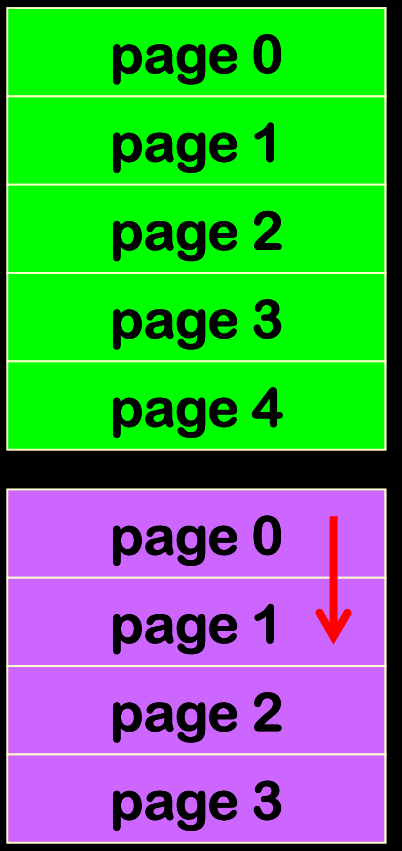




Thrashing

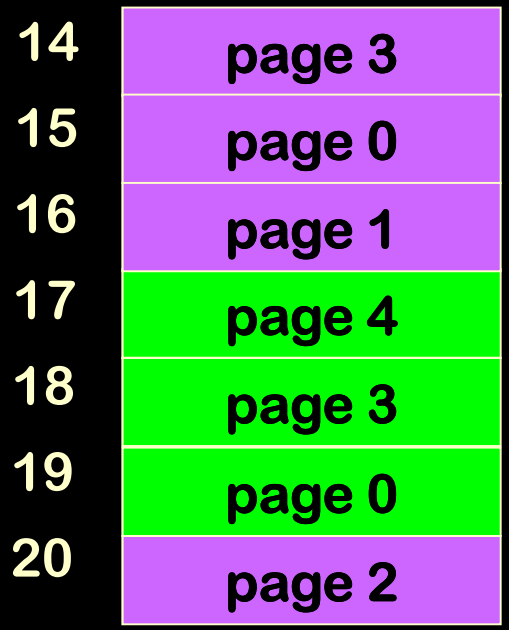
- Too frequent Page Swaps

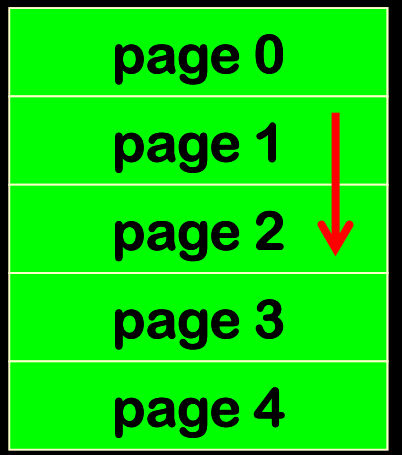
14	page 3
15	page 0
16	page 2
17	page 4
18	page 3
19	page 0
20	page 2

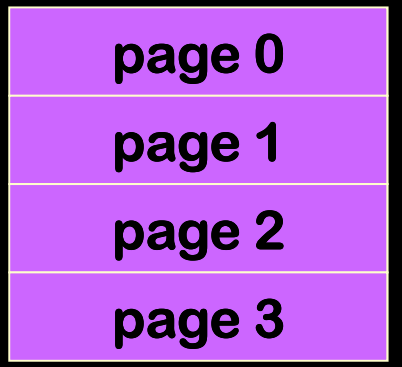


Thrashing

- Too frequent Page Swaps

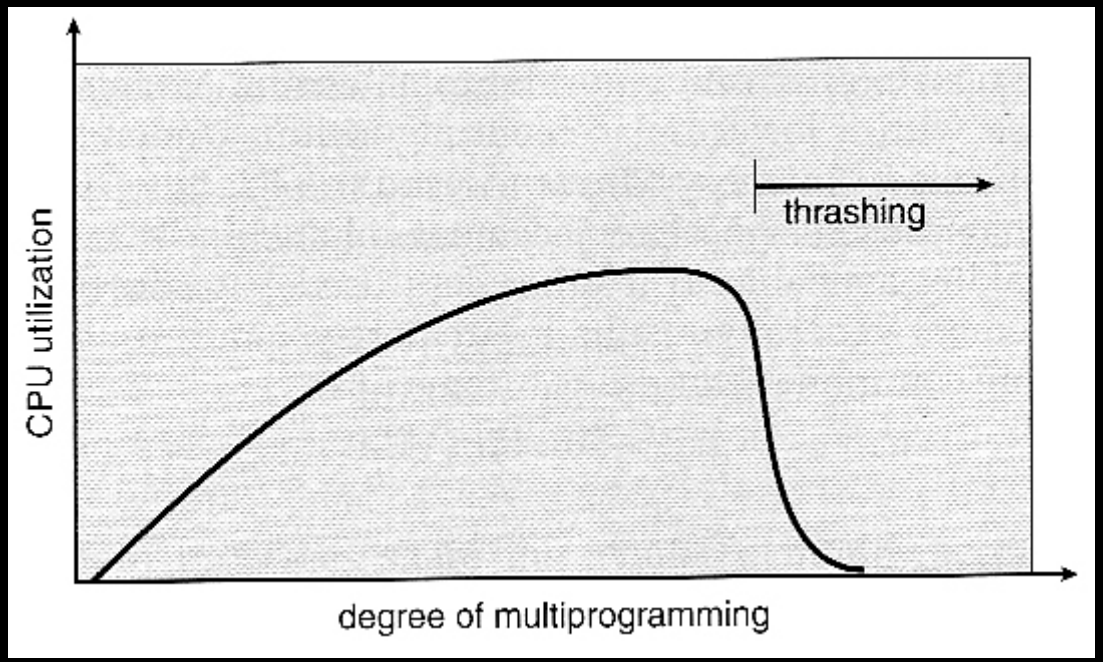






Thrashing

- Too frequent Page Swaps
  - Not enough real memory
  - Poor program design



Summary

Scheme	Features	Costs
One Partition	OS and program separate	• Base register to map “logical” address to physical address

Summary

Scheme	Features	Costs
One Partition	OS and program separate	• Base register to map “logical” address to physical address
Fixed Partitions	Multiple processes	• Available memory may not be in one place

Summary

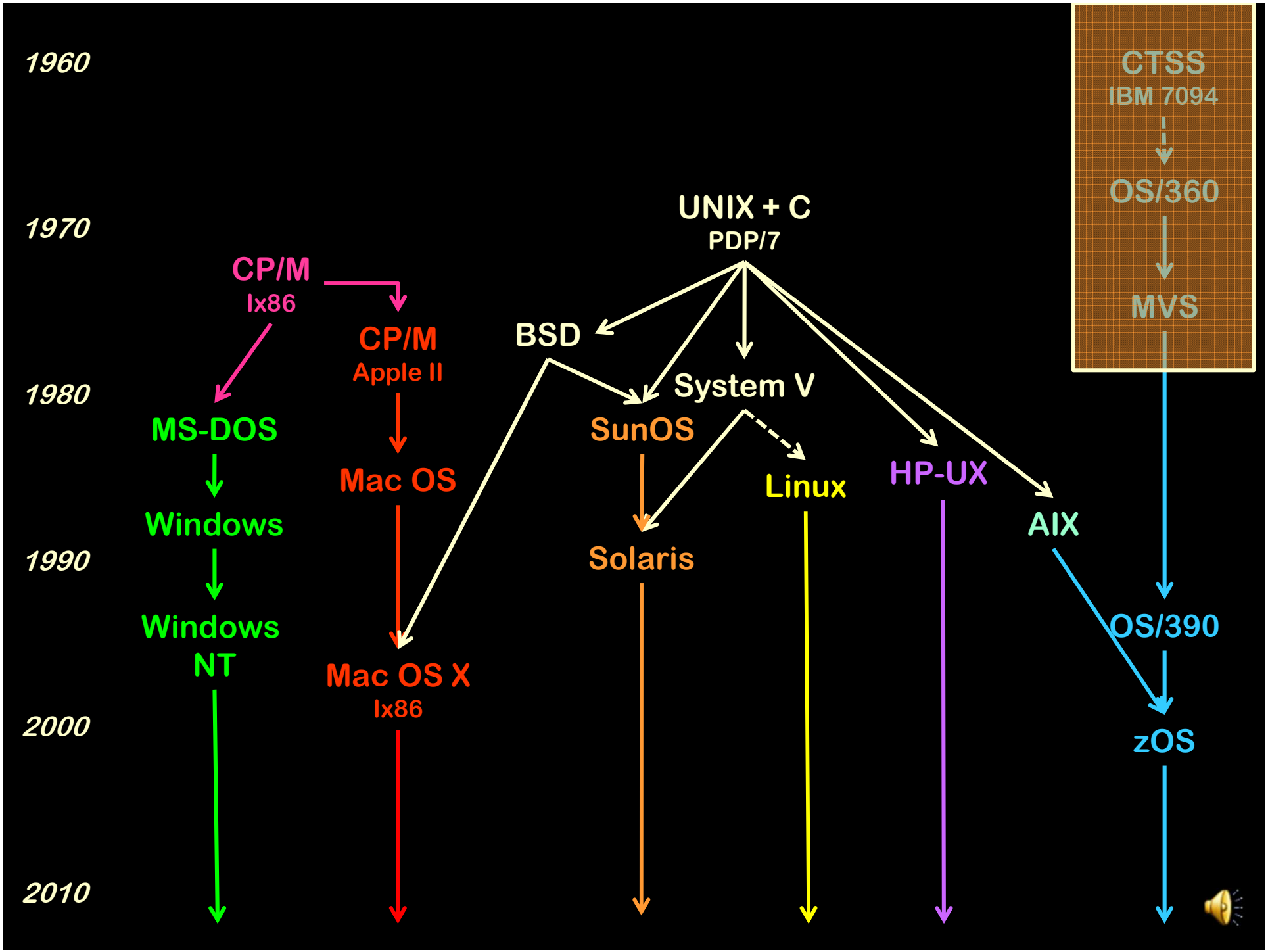
Scheme	Features	Costs
One Partition	OS and program separate	• Base register to map “logical” address to physical address
Fixed Partitions	Multiple processes	• Available memory may not be in one place
Variable Partitions	More efficient use of space	• Bounds register

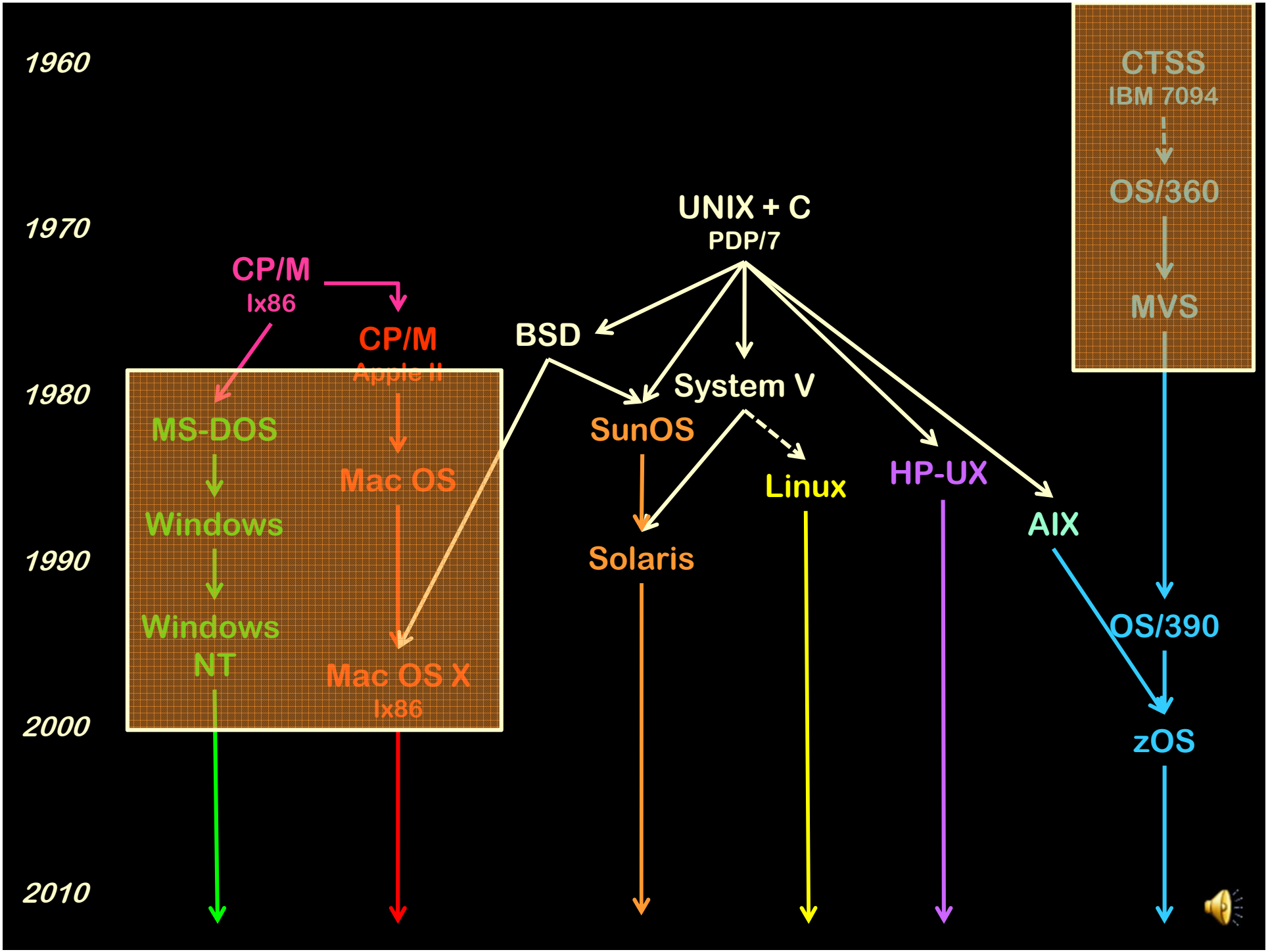
Summary

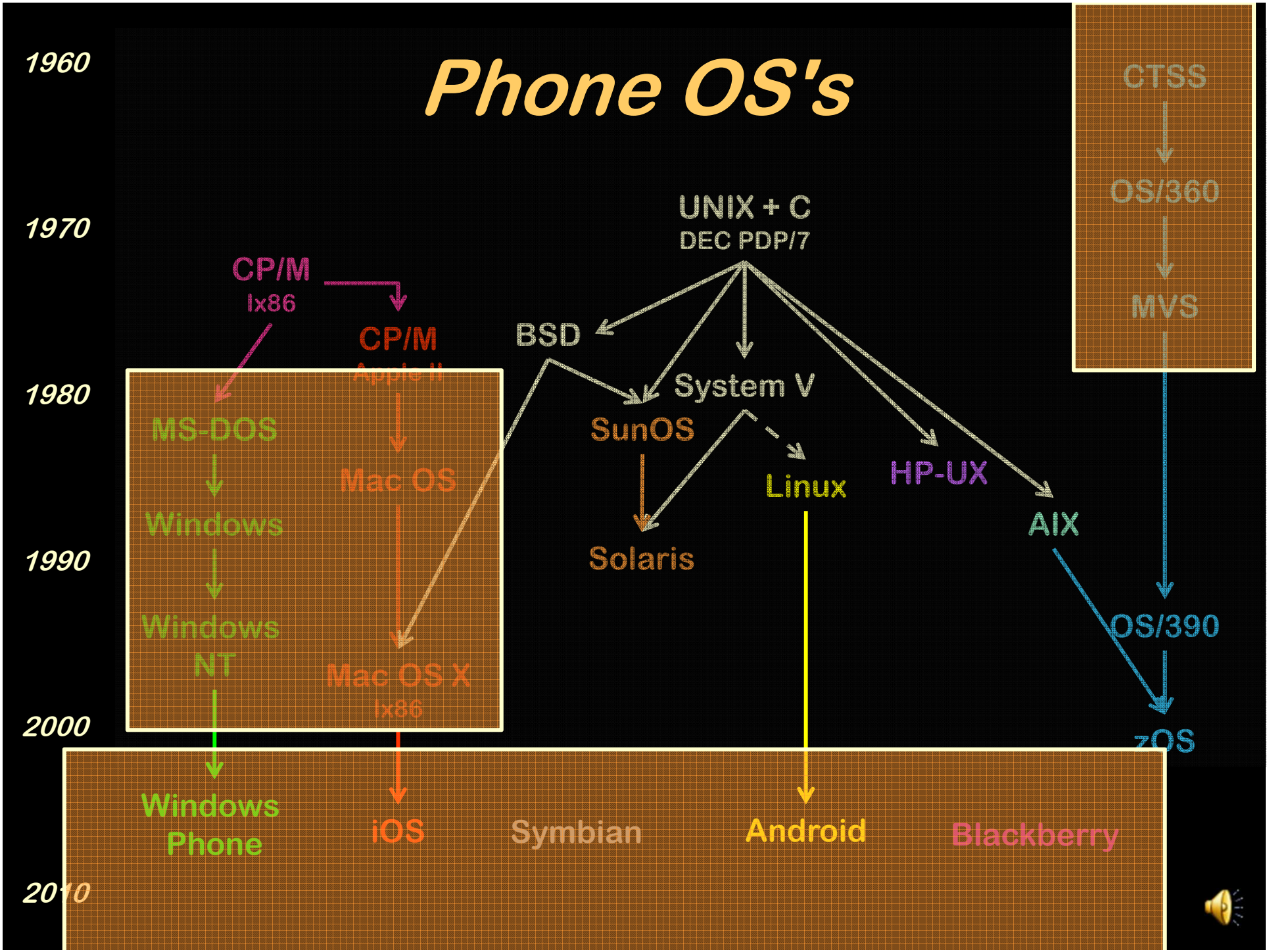
Scheme	Features	Costs
One Partition	OS and program separate	• Base register to map “logical” address to physical address
Fixed Partitions	Multiple processes	• Available memory may not be in one place
Variable Partitions	More efficient use of space	• Bounds register
Paged Memory	Most efficient use of space	• Program can still be “too big” • Page Map Table + interrupts

Summary

Scheme	Features	Costs
One Partition	OS and program separate	• Base register to map “logical” address to physical address
Fixed Partitions	Multiple processes	• Available memory may not be in one place
Variable Partitions	More efficient use of space	• Bounds register
Paged Memory	Most efficient use of space	• Program can still be “too big” • Page Map Table + interrupts
Virtual Memory	Allow larger processes than physical memory	• Requires “demand paging” • Can “thrash”


1960
1970
1980
1990
2000
2010
CTSS
IBM 7094
OS/360
MVS
UNIX + C
PDP/7
CP/M
Ix86
CP/M
Apple II
BSD
System V
MS-DOS
SunOS
Mac OS
Linux
HP-UX
AIX
Windows
Solaris
Windows
NT
OS/390
Mac OS X
Ix86
zOS


1960
1970
1980
1990
2000
2010
CTSS
IBM 7094
OS/360
MVS
UNIX + C
PDP/7
CP/M
Ix86
CP/M
Apple II
BSD
System V
MS-DOS
Windows
Windows
NT
Mac OS
Mac OS X
Ix86
SunOS
Solaris
Linux
HP-UX
AIX
OS/390
zOS


Phone OS's
1960
1970
1980
1990
2000
2010
CTSS
OS/360
MVS
UNIX + C
DEC PDP/7
CP/M
Ix86
CP/M
BSD
System V
SunOS
MS-DOS
Mac OS
Windows
Linux
HP-UX
AIX
Solaris
Windows
NT
OS/390
Mac OS X
Ix86
zOS
Windows
Phone
iOS
Symbian
Android
Blackberry