Memory Management Technique: Paged, Segmented and Paged-Segmented

CS303 Operating Systems
Autumn 2018
8Nov2018

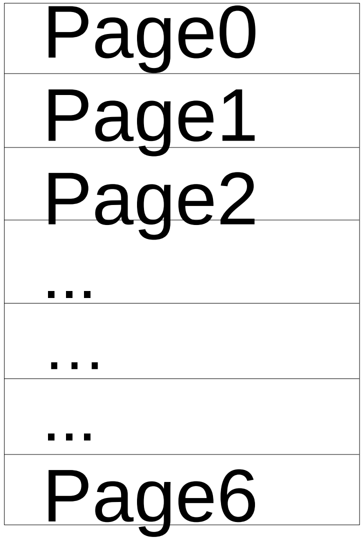
MVT Illustration: Compaction is invoked relatively depending on the adv.

- If the available free partition exceeds the req. process mem. By few bytes
  - Better go for it than carrying the costly mem. Compaction operation
  - Might result in very small amount of internal frag.
    - Its less expensive at times to live with int. frag. Of few KB
      - Than carrying out the compaction
        - And space complexity wise as well

Paged Memory Management (PMM)

- Overcomes the need for compaction
- In PPM
    - Job/Process/Task is divided into pages
    - MM is also divided into frames
    - Size of pages/frames is equal
        - Pagesize: P bytes

Paged Memory Management (PMM) (… Cont.)



Page0

Page1

Page2

...

…

...

Page6

process

PMM Technique essentials

- MM essentially resolves a LA into the Physical Address
- Now since pages of processes get loaded into frames of memory in a opportunistic mapping
  - We need to
    - First resolve which LA is in which Process-Page
    - Also get the offset address of this instruction in corrsponding process-page
    - Then resolve which page is loaded into which frame
    - Since the page-size and frame-sizes are identical, offset remains the same
  - Thus an LA is first resolved into page-number and offset, which then helps in table lookup, resulting in the LA to final Physical Address in the memory
  -

Paged Memory Management (PMM)

- Drawbacks of MFT and MVT:
  - If two or more instances of an application get launched
    - What would happen?
      - Processes get created as many times the appli. is requested
      - MM gets populated
        - Over utilisation of resources i.e. MM
      - Several partitions exist with same text/code section

- Accross all instances of the same application:
  - Since text section remains the same
  - It would be nice to share the code section
    - Avoid duplicate entries in MM

Paged Memory Management (PMM)

- In comparison with MFT and MVT,
  - Sizes of process-page and MM-frame being equal, external fragmentation is NOT possible
  - PMM permits sharing of **reentrant-code and non-self-modifying code** across users
    - Reentrant code: possible to invoke the function/s with different arguments from different call points
    - Self-modifying: code that modifies its own behavior at it is running
      - SDI Vs. MDI
      - Illustrate SDI and MDI concept
      - Illustrate GUI with common menu based OS

- Drawbacks of PMM:
    - Still internal fragmentation is possible
      - What is the max. size of internal fragmentation?

Paged Memory Management (PMM)

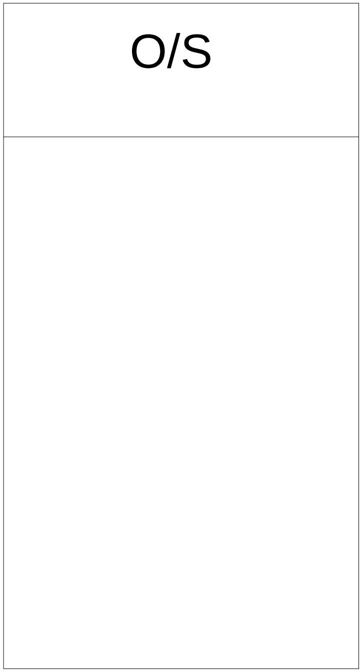
- Overcomes the need for compaction
- In PPM
    - Job/Process/Task is divided into pages
    - MM is also divided into frames
    - Size of pages/frames is equal
        - Pagesize: P bytes

PM Illustration

Page Map Table

Page	Frame

Main Memory



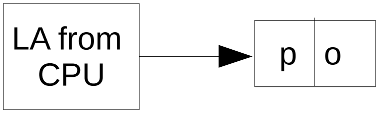
LA from CPU

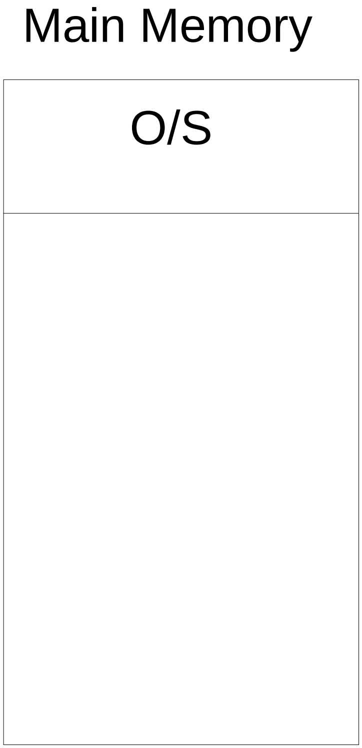
LA: generated logical address
P: page size = frame size
p = LA **div** P; o = LA **mod** P

PA= f*P + o (when page/frame begins with 0 index)

Page Map Table

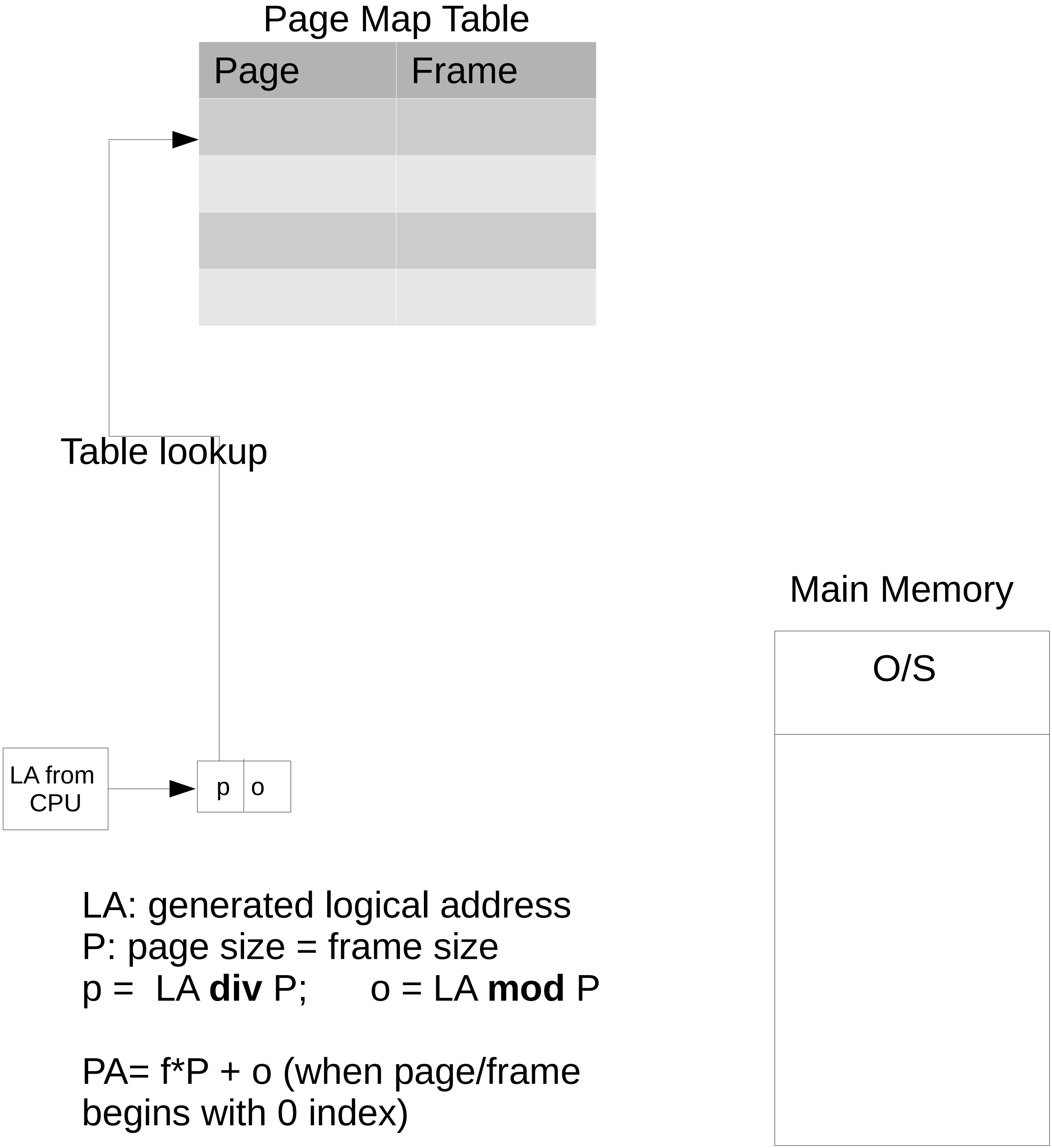
Page	Frame





LA: generated logical address
P: page size = frame size
p = LA **div** P; o = LA **mod** P

PA= f*P + o (when page/frame begins with 0 index)



Page Map Table

Page	Frame

Table lookup

LA from CPU

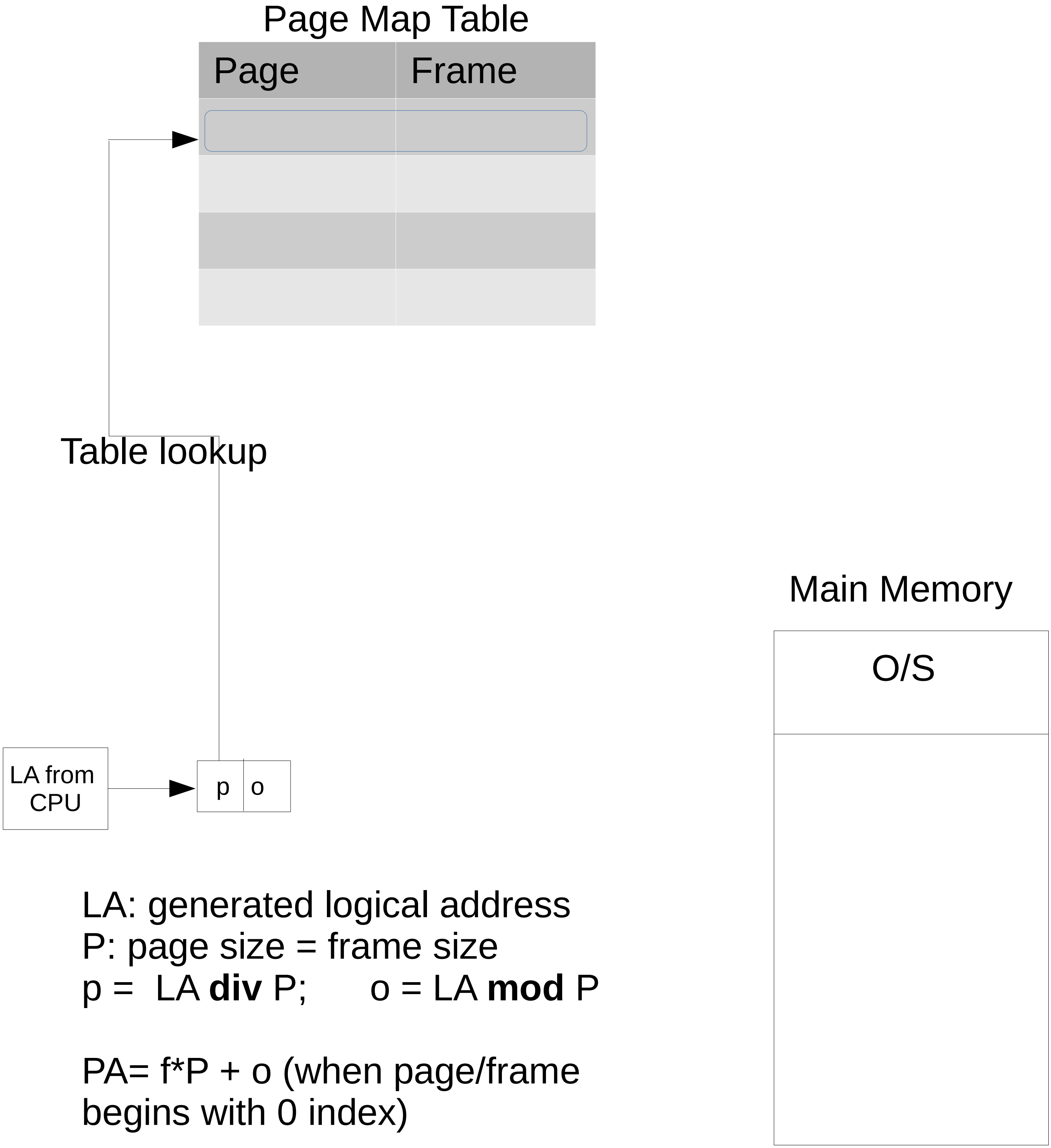
p o

Main Memory

O/S

LA: generated logical address
P: page size = frame size
p = LA **div** P; o = LA **mod** P

PA= f*P + o (when page/frame begins with 0 index)



Page Map Table

Page	Frame

Table lookup

LA from CPU

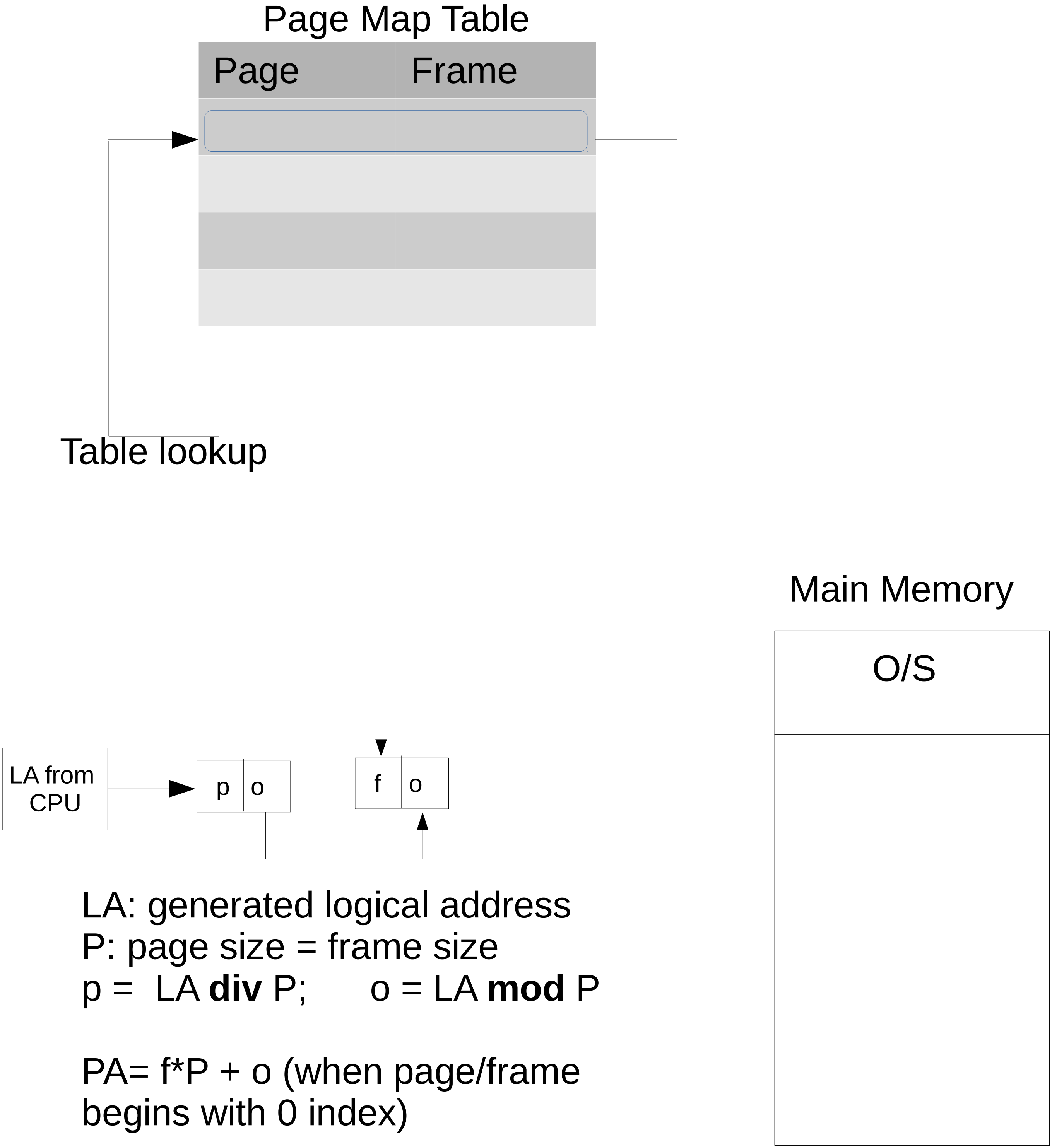
p o

Main Memory

O/S

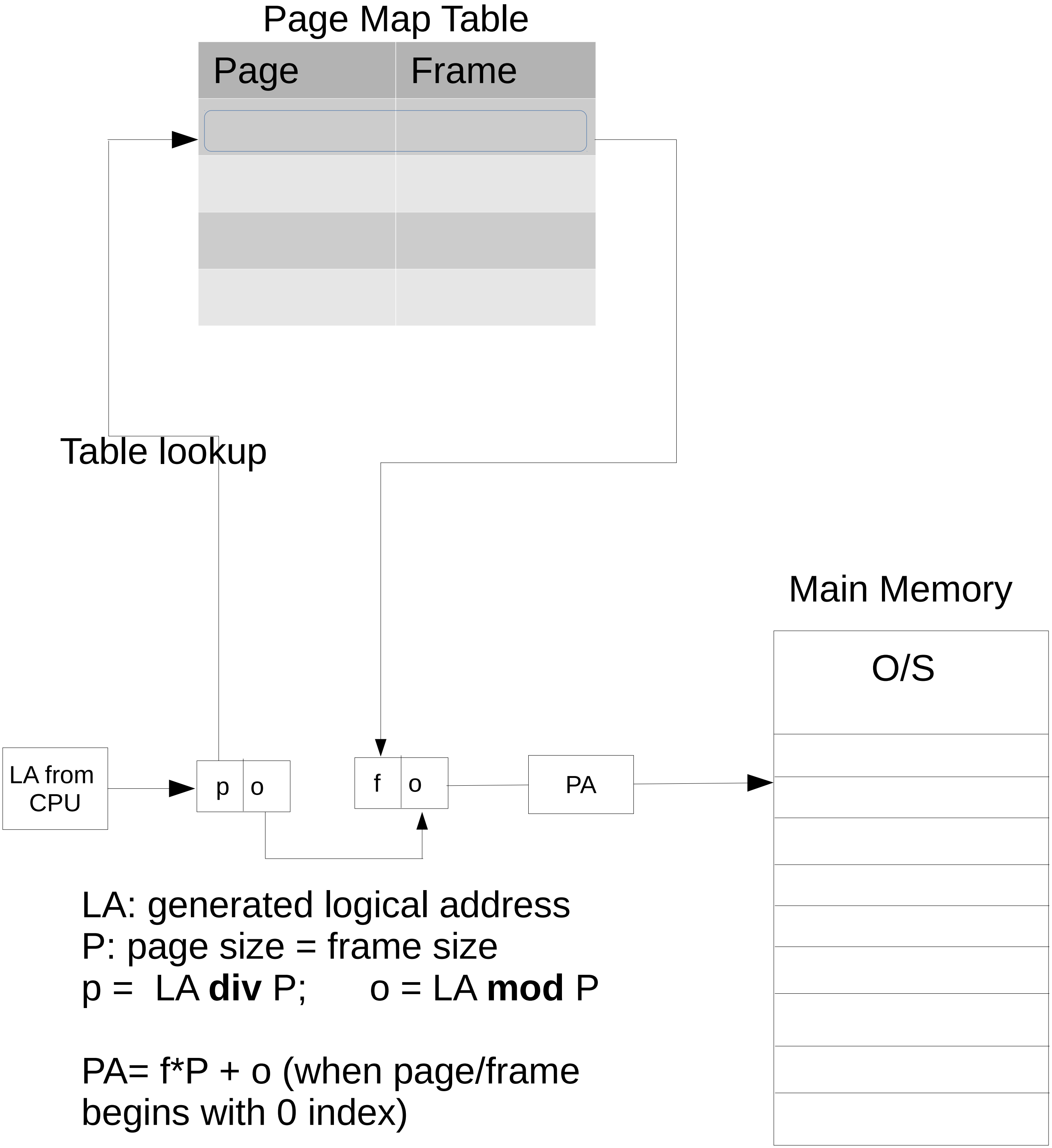
LA: generated logical address
P: page size = frame size
p = LA **div** P; o = LA **mod** P

PA= f*P + o (when page/frame begins with 0 index)



LA: generated logical address
P: page size = frame size
p = LA **div** P; o = LA **mod** P

PA= f*P + o (when page/frame begins with 0 index)



Page Map Table

Page	Frame

Table lookup

Main Memory

O/S

LA from CPU

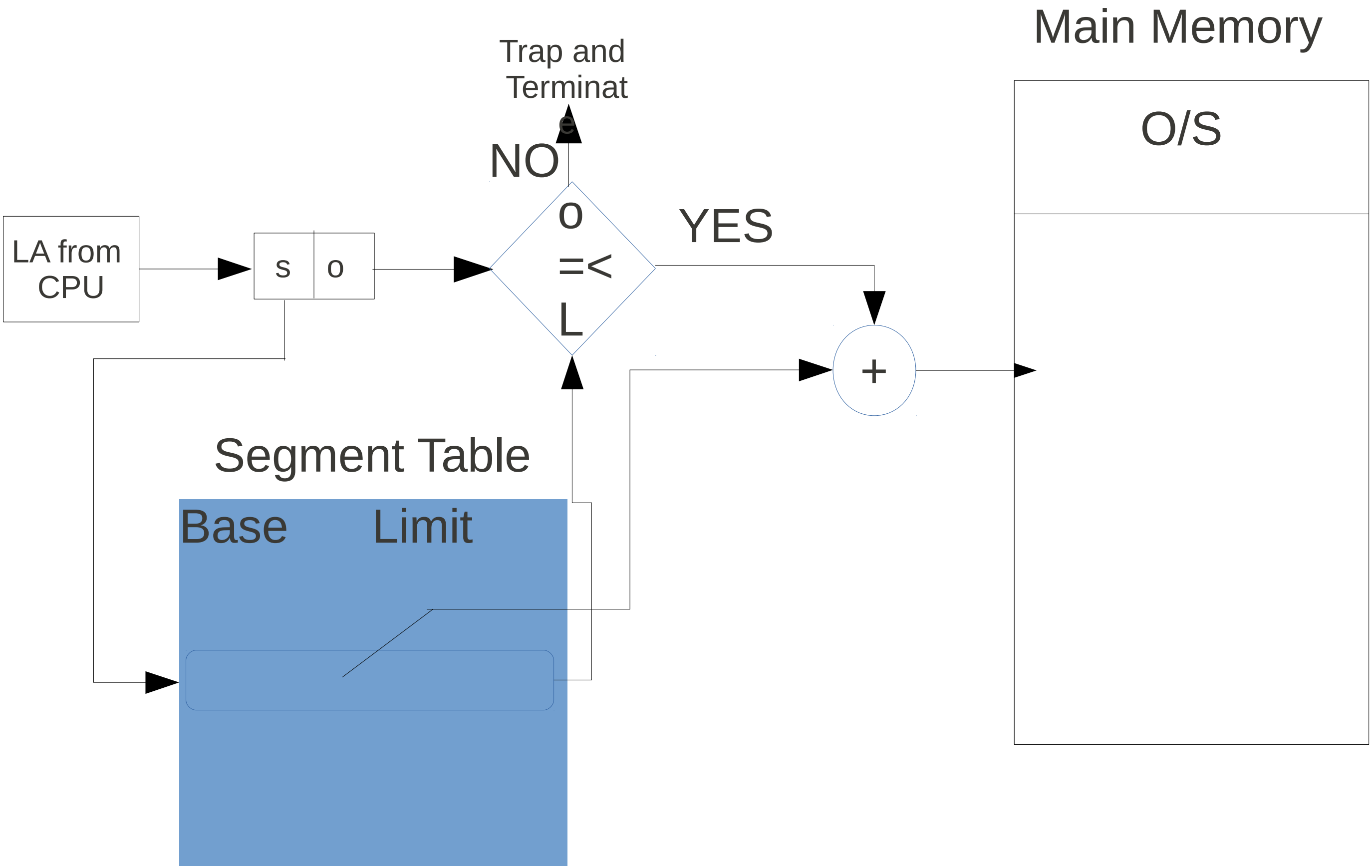
p o

f o

PA

LA: generated logical address
P: page size = frame size
p = LA **div** P; o = LA **mod** P

PA= f*P + o (when page/frame begins with 0 index)


Main Memory
Trap and
Terminat
e
NO
O/S
LA from
CPU
s
o
o
=<
L
YES
+
Segment Table
Base
Limit