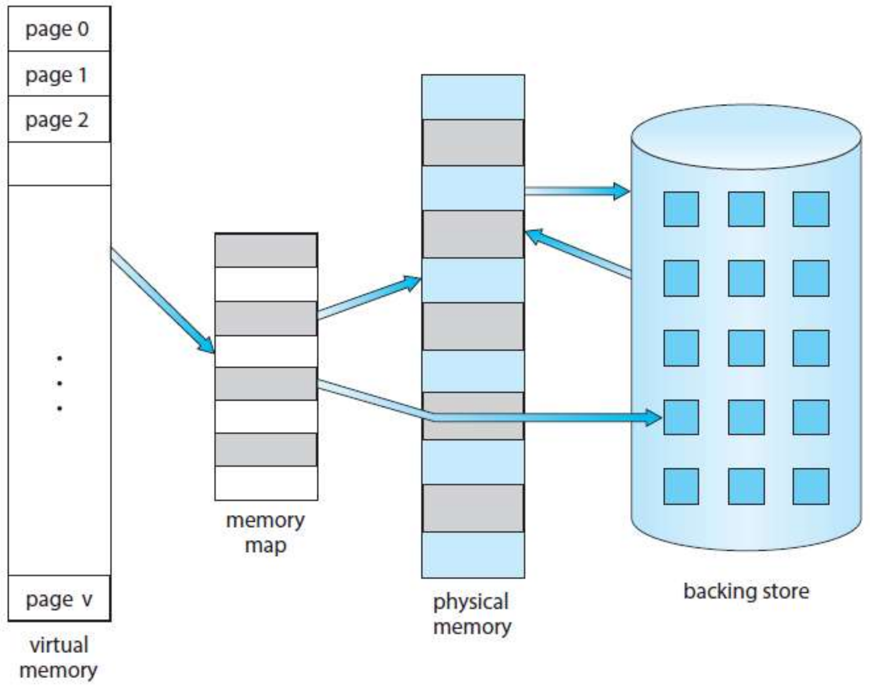
Demand Paging – Basic Concepts

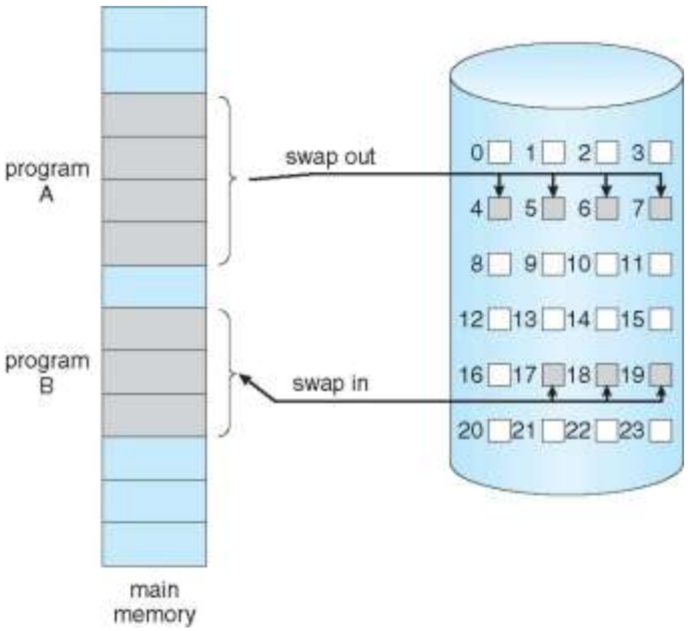
Demand Paging

- How to load program to memory – Two options
  - Could bring entire process into memory at load time
  - We may not need the entire program in memory initially
- Bring a page into memory only when it is needed – **Demand Paging**
  - Less I/O needed, no unnecessary I/O
  - Less memory needed
  - Faster response
  - More users

Demand Paging

- It is similar to paging system with swapping.
  - invalid reference ⇒ abort
  - Not-in-memory ⇒ bring to memory
- Lazy swapper – never swaps a page into memory unless page will be needed.
  - Swapper that deals with pages is a pager.





Basic Concepts

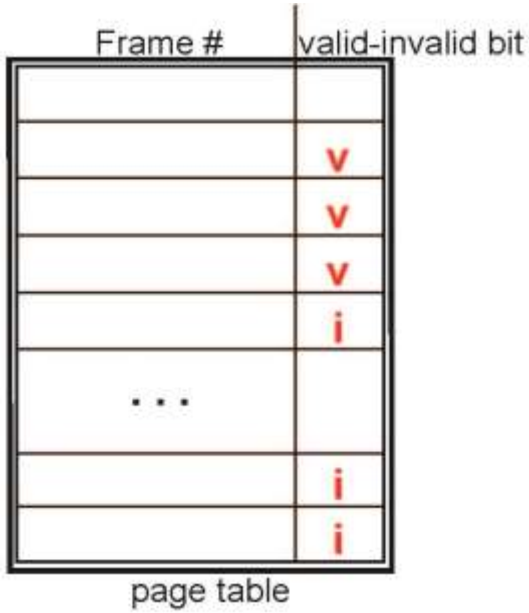
- General concept – Load the page into main memory only when it is needed.
  - Some pages will be in main memory
  - Some may be still in secondary storage.
- Need some form of hardware support to distinguish between the two.

Basic Concepts

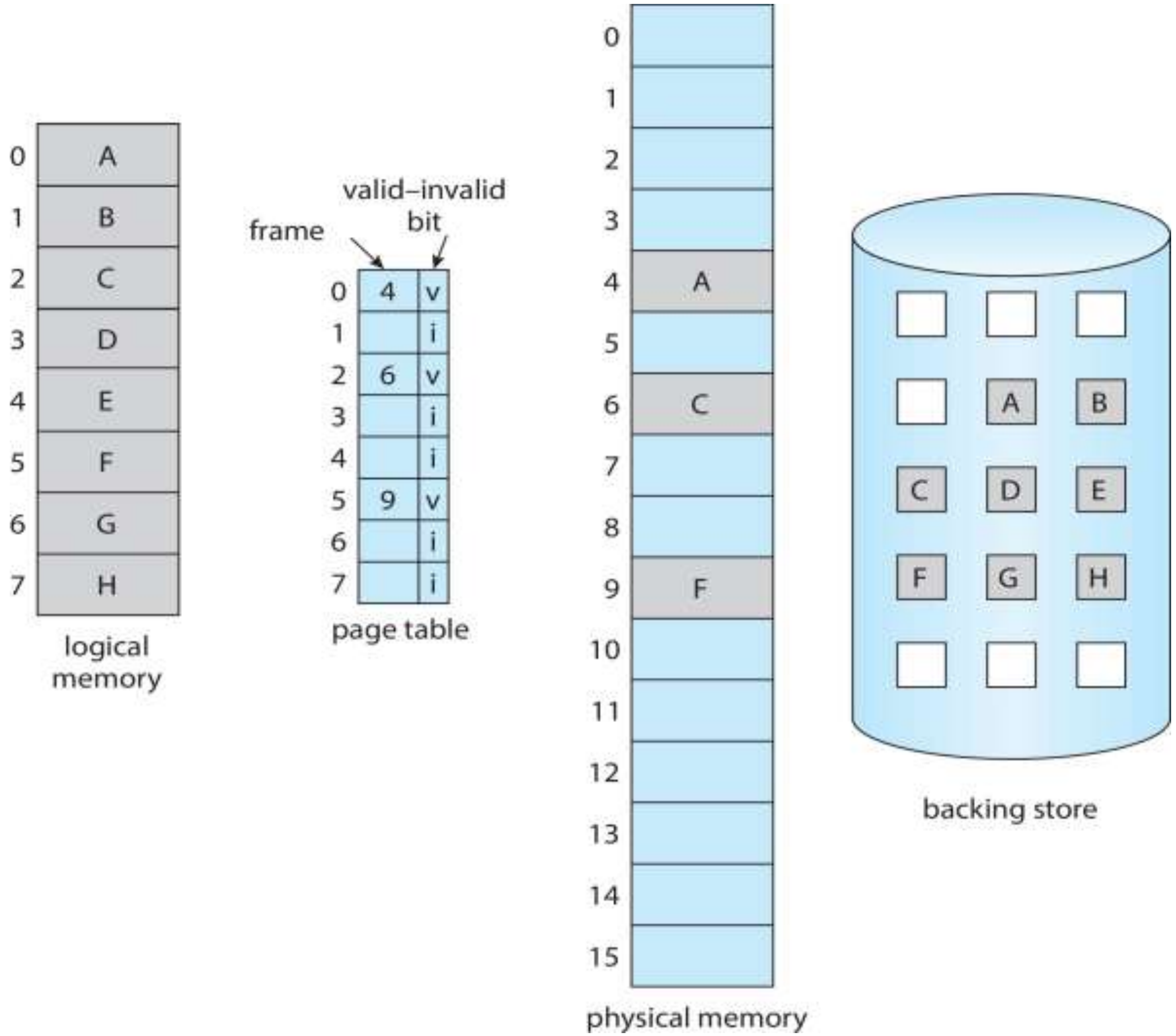
- If pages referred are in main memory
  - Works like non demand paging
- If pages referred are not in main memory
  - Need to detect and load the page into memory from storage.
    - Without changing program behavior
    - Without programmer needing to change code
- **Solution** - Use page table with valid-invalid bit

Page table with Valid – Invalid Bit

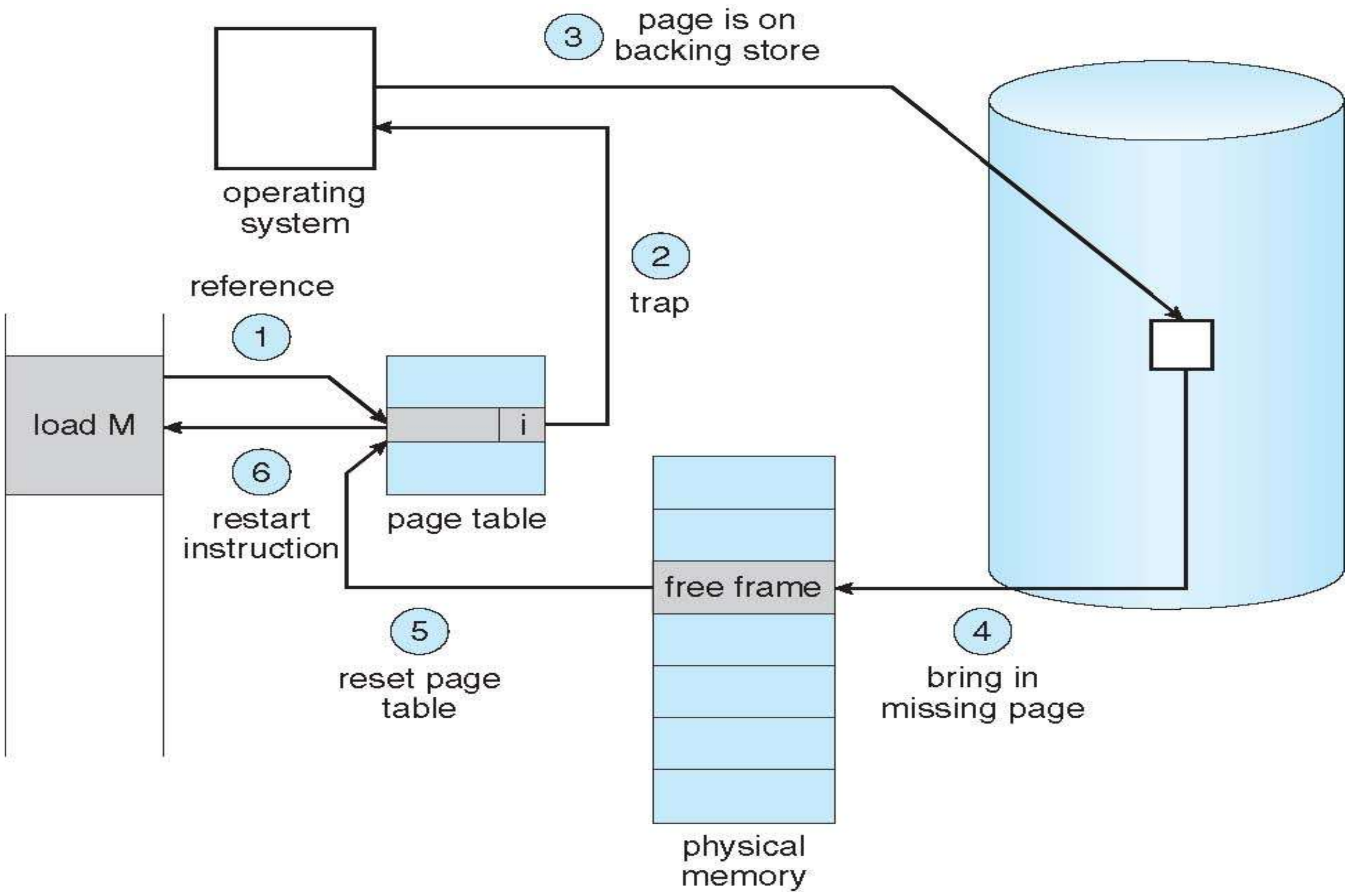
- With each page table entry a valid–invalid bit is associated
  (v ⇒ in-memory, i ⇒ not-in-memory)
- Initially valid–invalid bit is set to i on all entries
- During MMU address translation, if valid–invalid bit in the page table entry is i ⇒ page fault



Page Table When Some Pages Are Not in Main Memory



Steps in handling page fault



Steps in handling Page Fault

1. Access to a page marked invalid causes a ***page fault***
2. Page fault causes trap to operating system
3. Operating system looks at another table to decide:
   a) Invalid reference $\Rightarrow$ abort
   b) Just not in memory (go to step 4)
4. Find free frame (what if there is none?)
5. Swap page into frame via scheduled disk operation
6. Reset tables to indicate page now in memory
   Set validation bit = *v*
7. Restart the instruction that caused the page fault.

Aspects of Demand Paging

- Pure demand paging: start process with no pages in memory.
  - OS sets instruction pointer to first instruction of process, non-memory-resident -> page fault
  - And for every other process pages on first access
  - Actually, a given instruction could access multiple pages -> multiple page faults.
- Hardware support needed for demand paging
  - Page table with valid / invalid bit
  - Secondary memory (swap device with swap space)
  - Instruction restart

Free Frame List

- When a page fault occurs, the operating system must bring the desired page from secondary storage into main memory.
- Most operating systems maintain a free-frame list → a pool of free frames for satisfying such requests.

- Operating system typically allocate free frames using a technique known as zero-fill-on-demand → the content of the frames zeroed-out before being allocated.
- When a system starts up, all available memory is placed on the free-frame list.