**Memory Management – Unit II – Part II**

* Concerned with management of primary memory

**Four functions of Memory management**

* Keeping track of the status of each location of primary memory, i.e., each location is either allocated or unallocated
* Determining allocation policy for memory, i.e., deciding to whom it should go, how much, when, and where.
* Allocation technique – once it is decided to allocate memory, the specific locations must be selected and allocation information updated.
* Deallocation technique and policy – handling the reclaiming of memory.

**Different Techniques/ Types of MM**

* Single Contiguous MM
* Partitioned MM
* Relocation partitioned MM
* Paged MM
* Demand Paged MM
* Segmented MM
* Segmented and Demand paged MM
* Other MM Schemes

**Four sections of MM techniques**

* An overview of the approach and concepts employed
* A description of any special hardware facilities required or recommended.
* A description of the particular software algorithms and processing required
* A discussion of the advantages and disadvantages of the particular strategy

**Demand Paged Memory Management**

Virtual Memory technique is to produce the illusion of an extremely large memory ( Size of the VM = RAM + Hard disk / peripheral device) . It follows the virtual memory techniques. When a job is initially scheduled for execution, usually only its first page is loaded. All other pages needed by the jobs are subsequently loaded on demand and guarantees that an unnecessary page is not loaded. A copy of a entire job’s address space is stored on a secondary storage device.

The entire job’s address space is not required because

1. User written error handling routines are used only when an error occurs in the data or computation.
2. Certain options of the programs are mutually exclusive
3. Many tables are assigned a fixed amount of address space even though only small amount of the table is actually used.
4. Many routines are commonly used at mutually exclusive routines during a run.

To handle demand page, the Page Map Table (PMT) hardware to include a status bit (Y, page is in memory, N page is in hard disk.) if status is N it generates page interrupt. The OS must process this interrupt by loading the required page and adjusting the PMT entries. This scheme is called **demand paged memory management**.

Moving pages between memory and secondary storage device is called **Page swapping or page removal or page replacement or page turning or page cannibalizing.** The phenomenon of excessively moving pages back and forth between memory and secondary storage has been called **Thrashing**.

The four Functions of Demand paged memory management environment L

1. Keeping track of status – this is accomplished through three sets of tables
2. Page Map Table – one per address space
3. Memory Block Table – one per system
4. File Map Table – one per address space
5. The policy of who gets memory who gets memory and when – partially determined by the job scheduler, but on a dynamic basis, it is also determined by the demand paged interrupts.
6. Allocation – when a block must be allocated, an available block must be found and the status of the block altered.
7. Deallocation – if it is not possible to find an available block for allocation, one of the allocated memory blocks must be deallocated and reassigned. When a job terminates, all the blocks it was using become available.

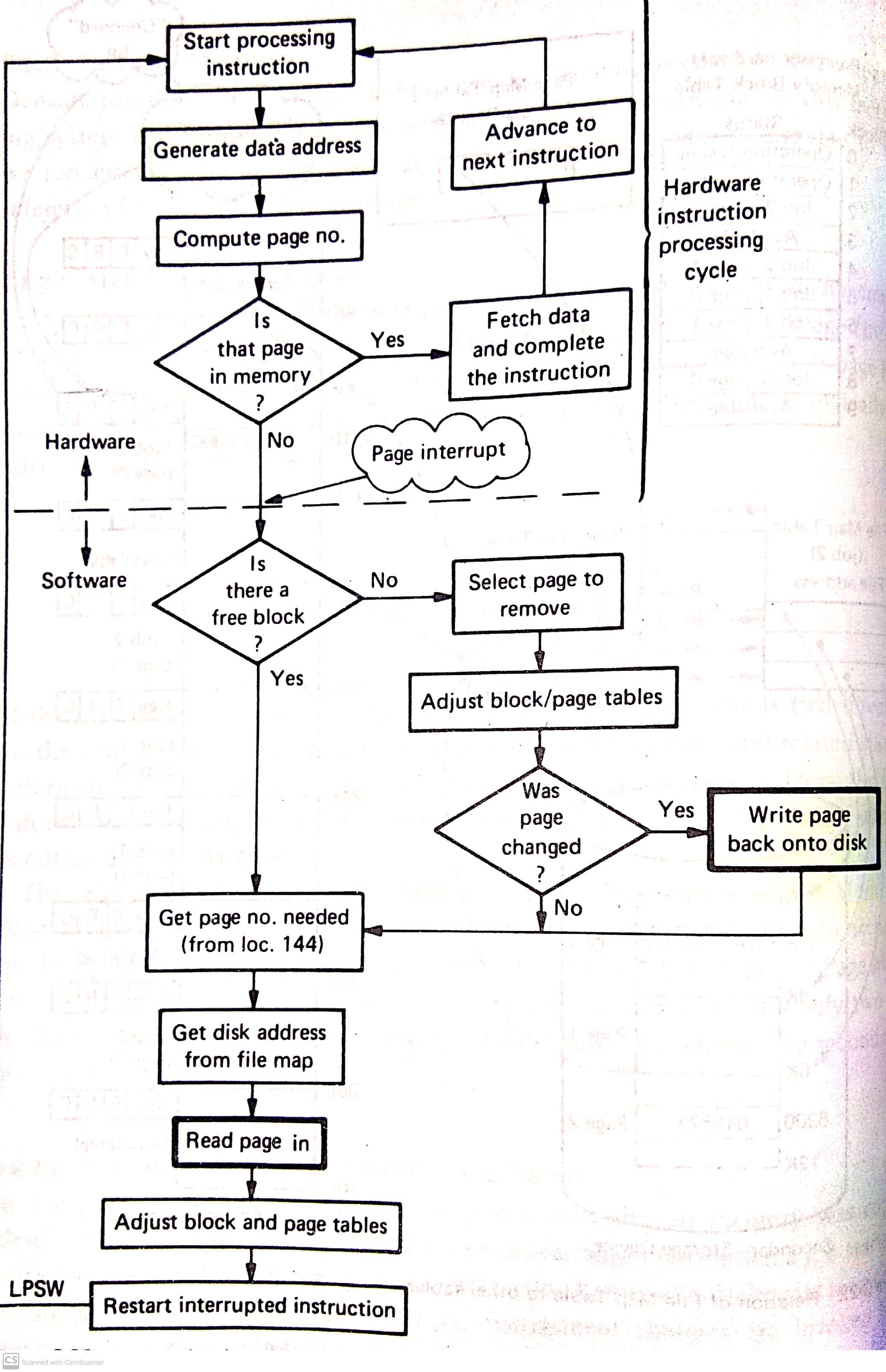
**Hardware Support**

The address mapping hardware via PMT needed for demand paging. Three key additions to the hardware are required

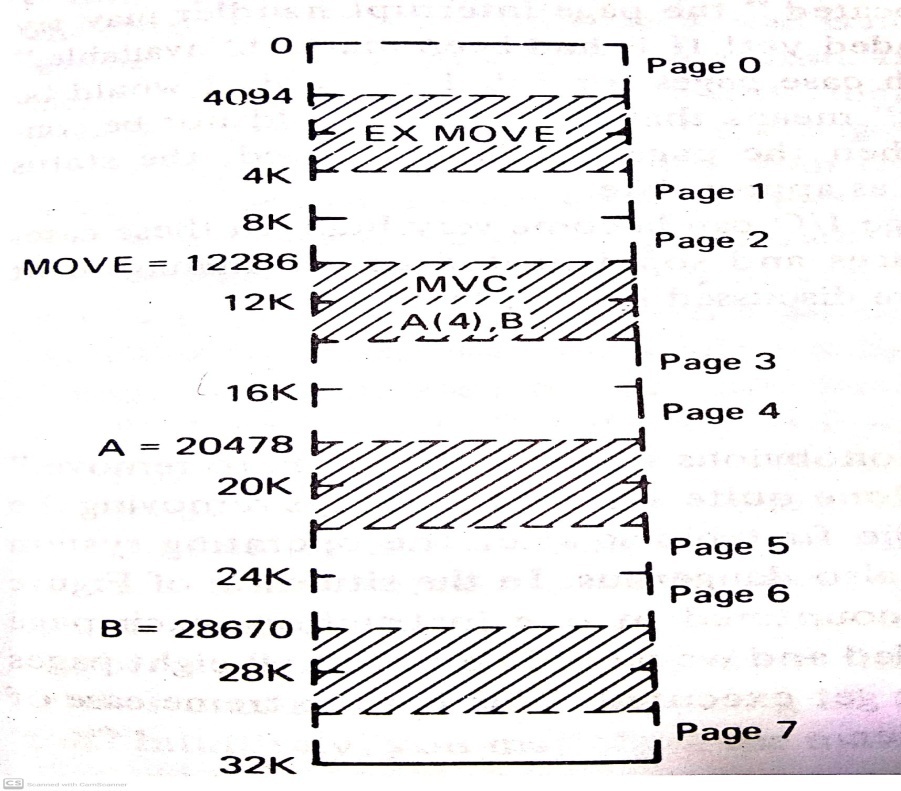
1. A status bit in the PMT
2. Interrupt action to transfer control to the OS if the job attempts to access a page not in memory.
3. Record of individual page usage to assist the OS in determining which page to remove, if necessary.

**Software Algorithm**

* **File Map Requirement** : PMT entry 32 bit , 0-11 bits: Block Number, 12th bit Interrupt bit, 13-15 bits –unused, 16-31 bits File address
* **Overview of Page Interrupt Processing**: In demand paging there is close interaction between the hardware and software is shown in the figure 1. In some situation all pages of an address space is required to execute a instruction as shown in the second figure. To execute the instruction EX MOVE.



**Figure 1. Interaction between hardware and software**



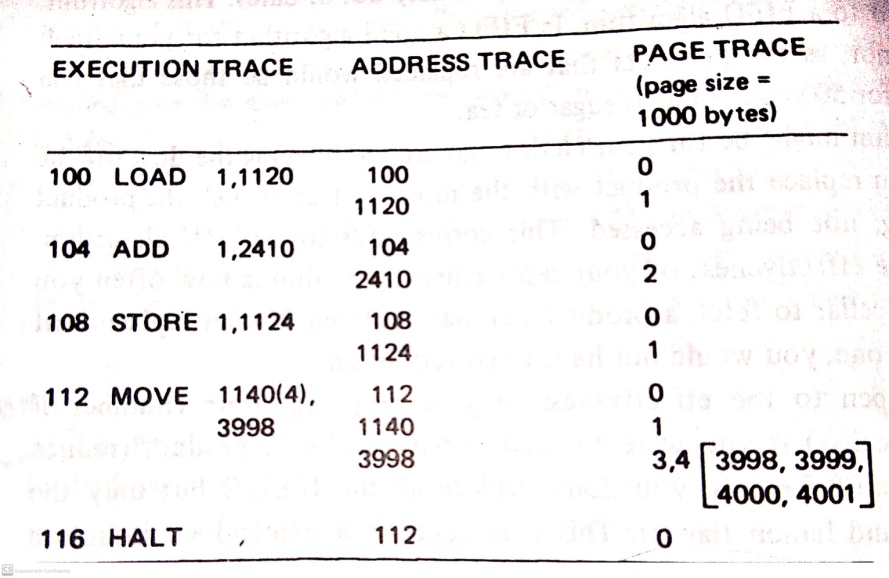
**Figure 2. Situation that can arise eight page interrupt**

**Page Removal Algorithms**

1. FIFO (First In First Out)
2. LRU (Least Recently Used)
3. Tuple-coupling

**Model of Page Removal performance and Program behaviour**

In order to evaluate the effectiveness of alternate page removal algorithms, it is necessary to use a performance model. The abstract of the program model as shown in the figure 3.



**Figure 3. Execution, Address and Page Trace**

FIFO page trace analysis is shown in figure 4. A page trace P, specific memory size M , a replacement algorithm R.

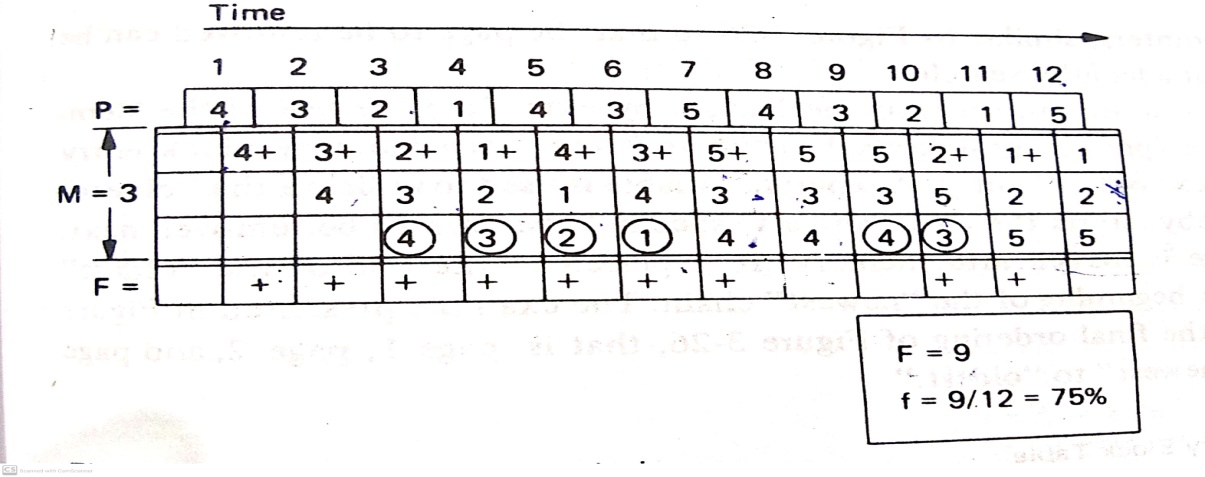


Figure 4 : FIFO page trace analysis

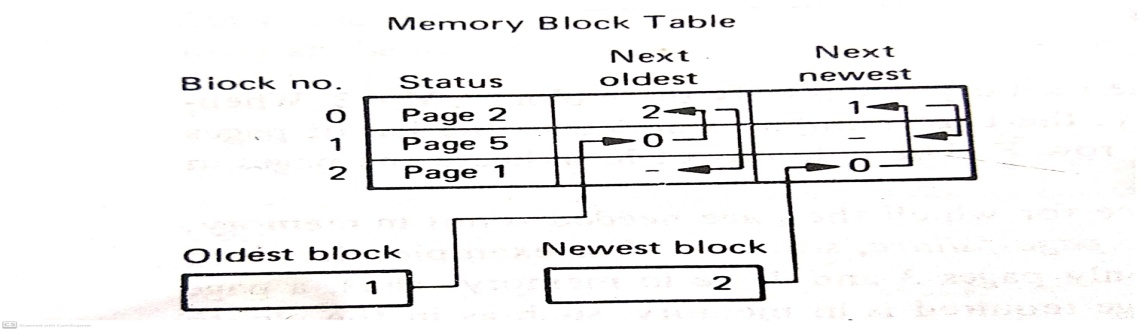
****

Figure 5 : Page Removal Record Keeping

**FIFO Removal**

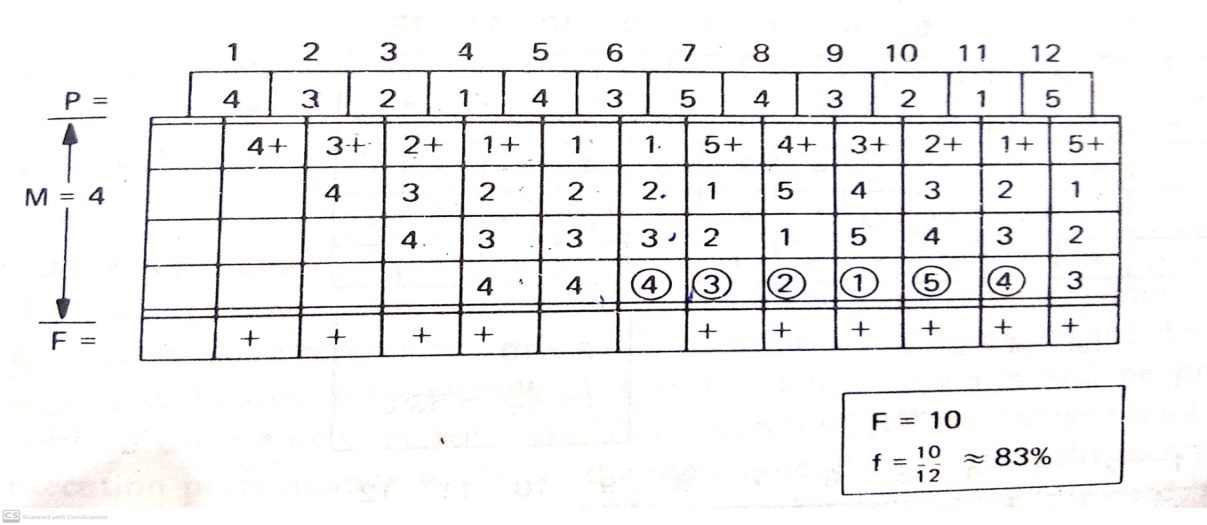
****

Figure 6 : FIFO Anomaly

In Figure 4 and 5 the page fault ratio is increasing in FIFO even though memory size increasing so the next page removal LRU to overcome the FIFO anomaly.

**LRU Removal**

It removes the page that has been in memory for the longest time., LRU removal for memory two memory sizes M=3 and M=4

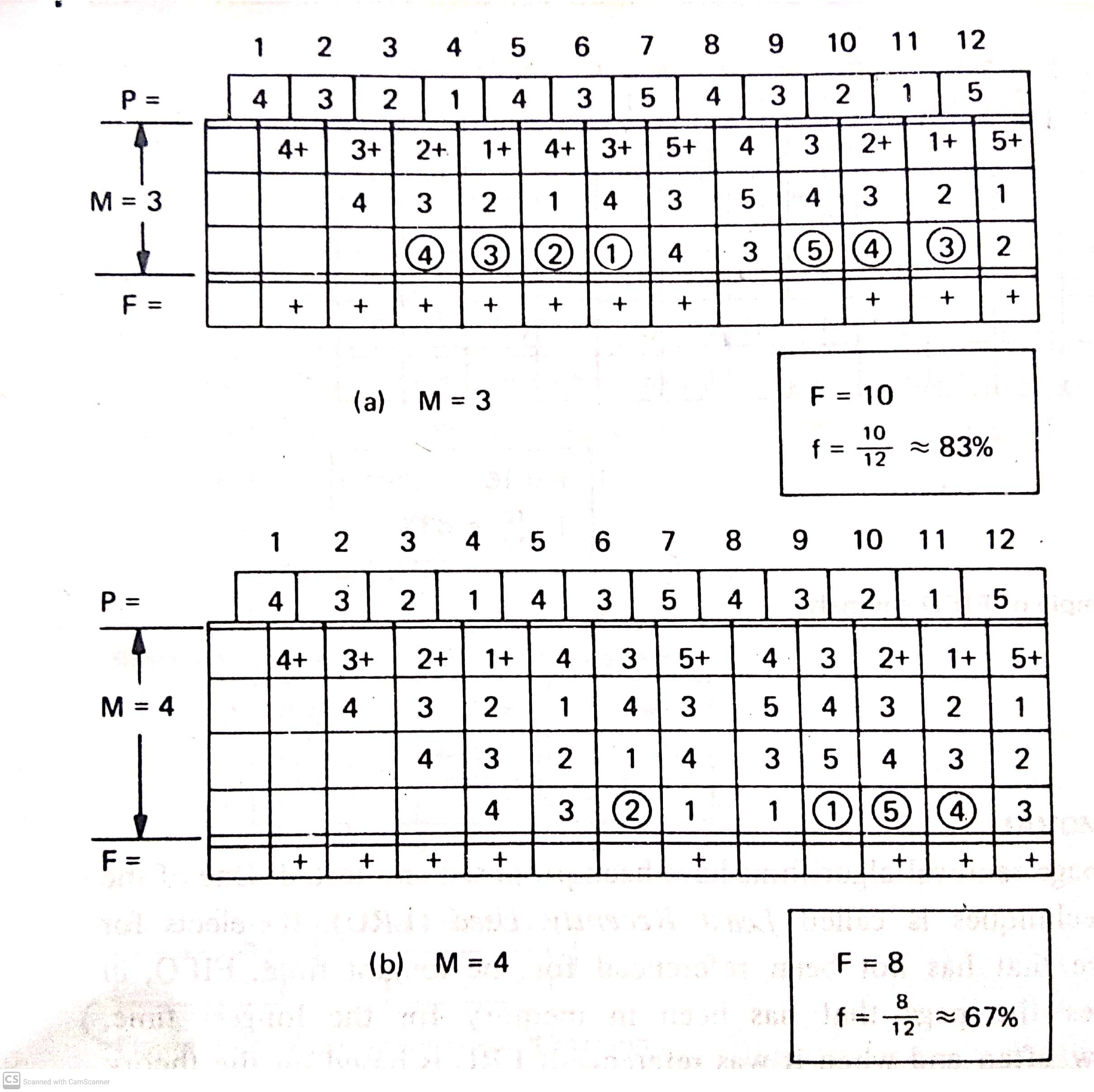


Figure 7: Example of LRU

**LRU Approximation**

An LRU removal algorithm should update the page removal status information after every page reference.

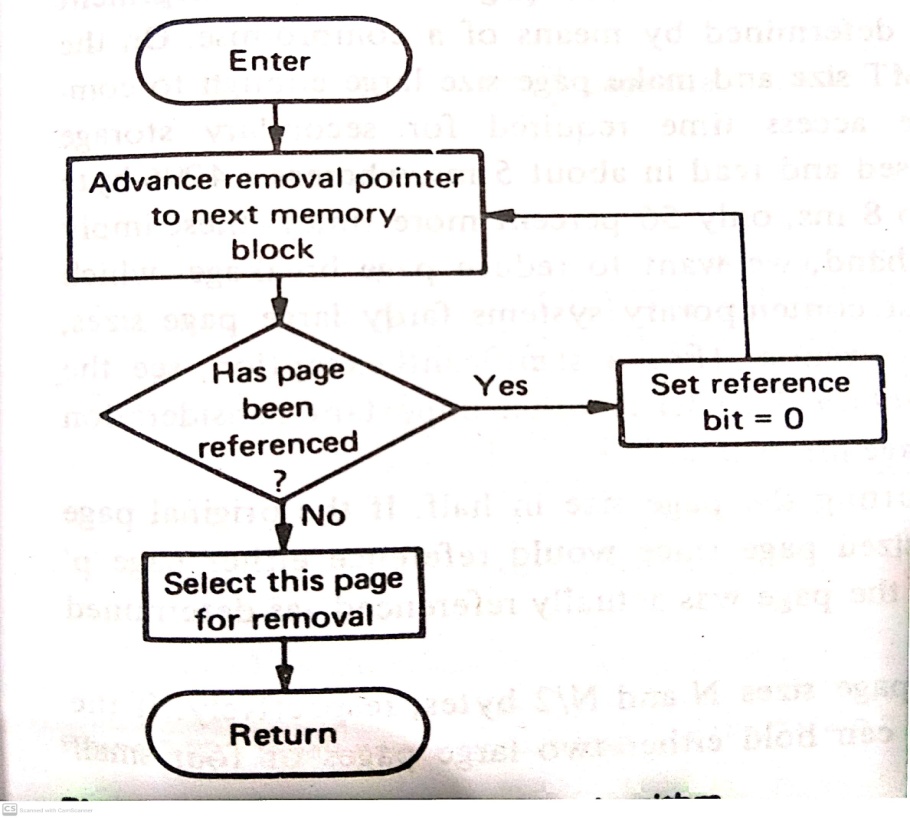


Figure 8 : LRU Approximation Algorithm

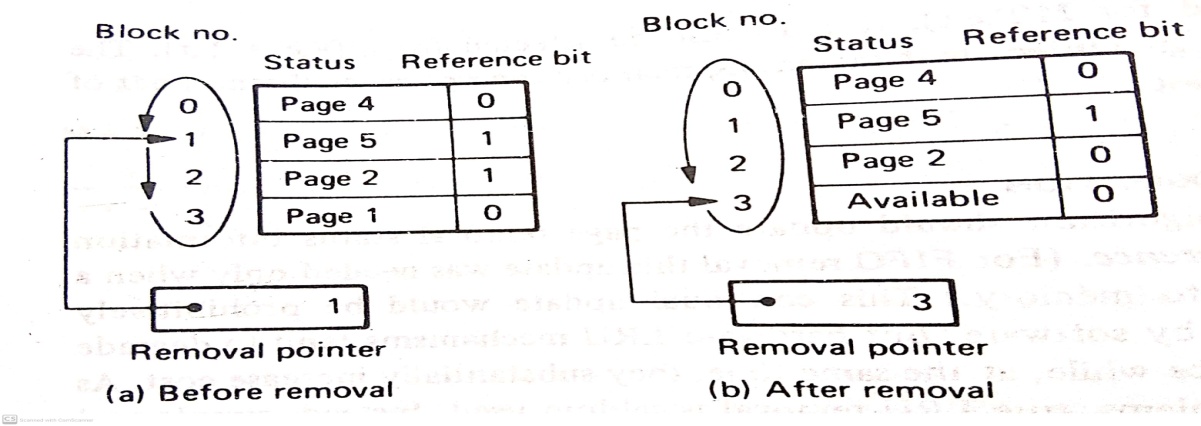


Figure 9 LRU Approximation

**Page Size Anomaly**

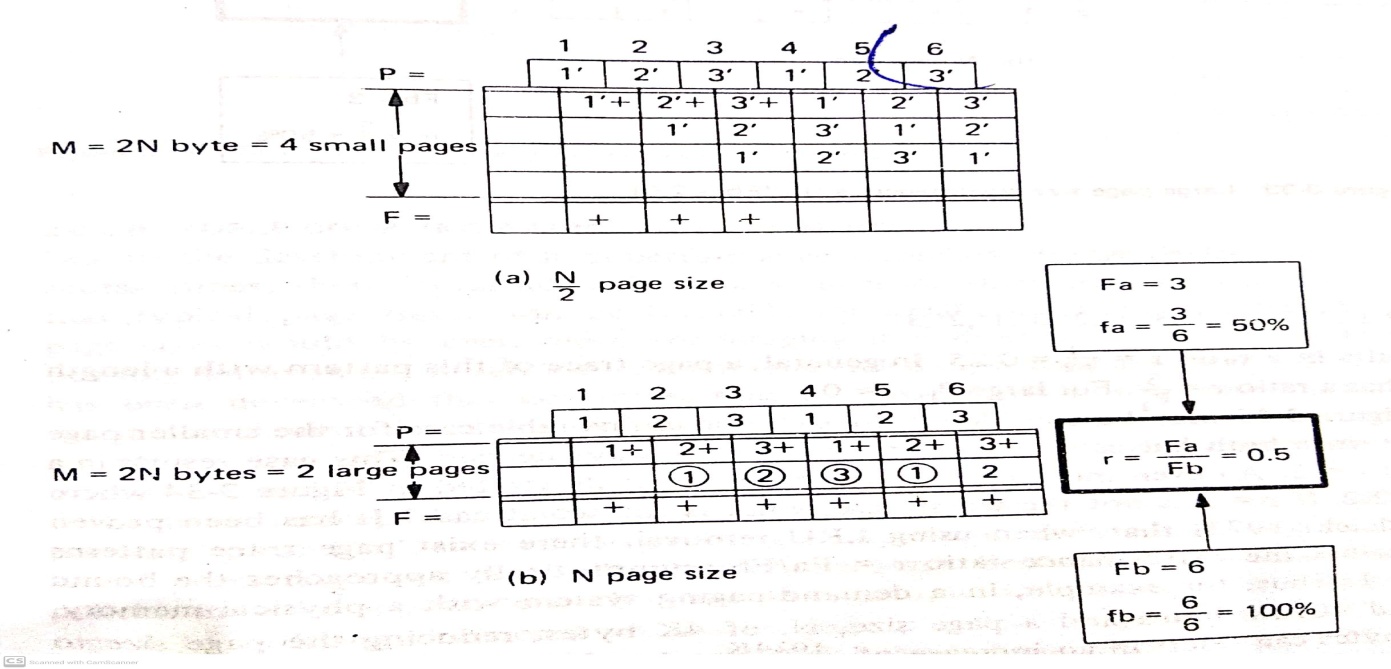
****

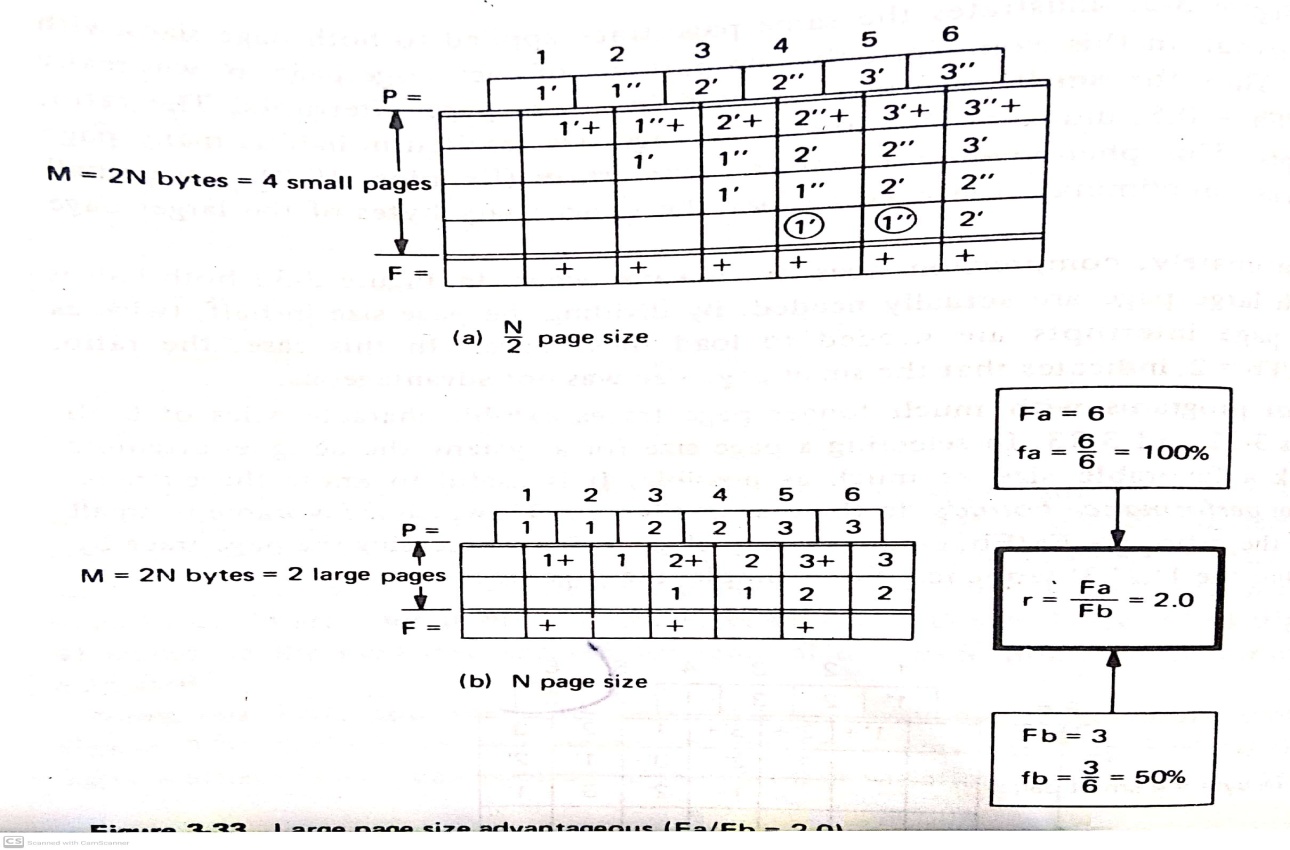
Figure 10 L Small Size advantageous 

Figure 11LLarge page size advantageous

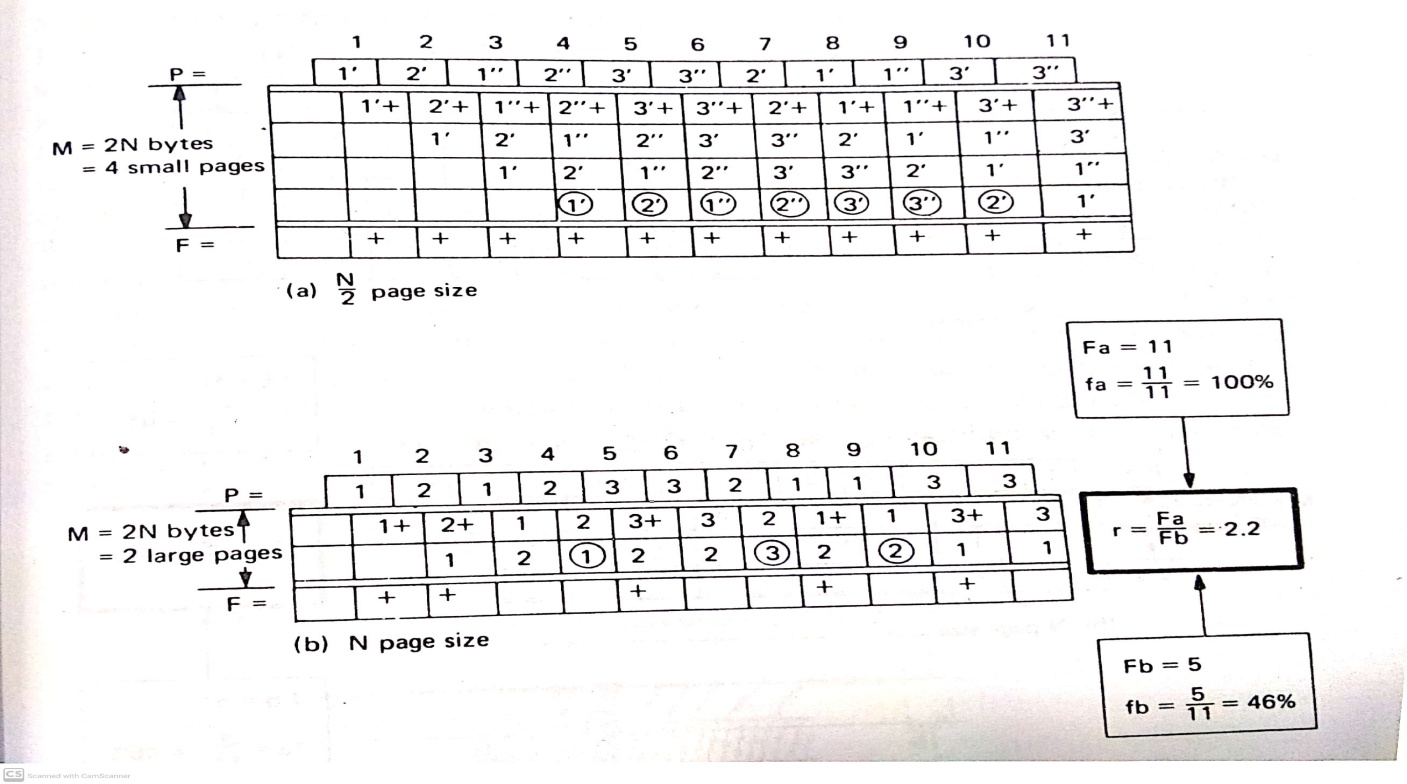


Fig12. Small Page size unfavourable

**Tuple-Coupling Approach to Page Removal**

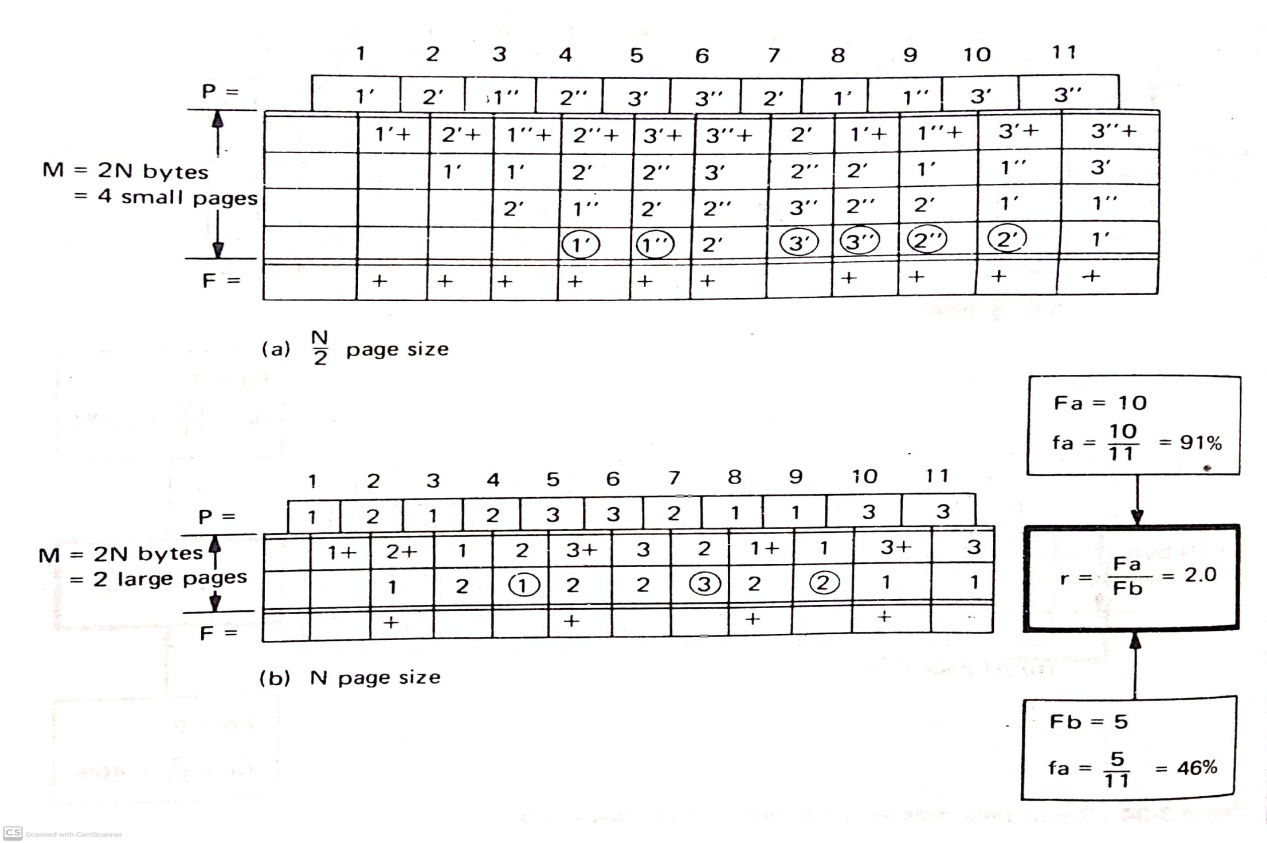


Fig13: LRU removal with tuple-coupling

**Performance and Program Behaviour Consideration**

Demand paging can provide good system performance by making it possible to multiprogram more jobs at the same time

**Address Reference Patterns**

To understand the address reference pattern, it is helpful to imagine a tiny light bulb connected to each byte of memory. This light bulb is turned on briefly whenever the corresponding byte is referenced either as data or as an instruction. The program’s reference pattern is not random. Very definite patterns correspond to different phases of the management.

**Locality**

Every operation of a virtual memory system is dependent upon the degree of locality of reference in programs. It can be divided into two classes

Temporal Locality

Spatial Locality

**Temporal Locality** refers once a location –data or instruction – is referenced, it is often referenced again very soon. Example : Loops, frequently used variables and subroutines

**Spatial Locality** refers to the probability that once a location is referenced, a nearby location will be referenced soon, example : linear data structure( array)

The degree of locality varies from program to program

**Advantages**

1. Large Virtual Memory
2. More efficient use of memory
3. Unconstrained multiprogramming

**Disadvantages**

1. The number of tables and amount of processor overhead for handling page interrupts are greater than in the case of the simple paged memory management
2. Extreme thrashing situations increasing the overhead of processor

**Segmented Memory Management**

A segment can be defined as a logical grouping of information such as a subroutine, array or data area. Each job’s address space actually consists of a collection of segments. Segmentation is the technique for managing these segments.

In a segmented environment all address space references require two components (1) a segment specifier and (2) the location within the segment

CALL [X]|<Y>

LOAD 1, [A|/6

STORE 1,[B]|<C>

Segmented memory management can offer several advantages

1. Eliminate fragementation
2. Provide Virtual Memory
3. Allow Dynamic growing segments or automatic bounds checking
4. Dynamic linking and loading
5. Facilities shared segments
6. Enforced controlled access

The four functions of memory management

1. Keeping track of status is done through four main sets of tables
2. Segment Map Tables – One per address space
3. Unallocated Area Table – One in system
4. Active Reference Table – One per address space
5. Active Segment Table – One in system
6. The policy of who gets memory and when may be determined statically by the job scheduler if virtual memory use is limited. In general VM are determined on a dynamic basis by segment demand interrupts.
7. Allocation
8. Deallocation

**Hardware Support**

**Formation of Segmented Address** is to generate a two component segment number ad byte number

**Address Mapping** is required because segments are of arbitrary size, it is necessary to check that the reference byte address is within the segment’s range.

**Software Algorithm**

1. Reference segment S, Byte B
2. Check is segment S in Memory if yes go to next step else segment exception interrupt
3. Check is B<=Segment length if yes continue else Segment out of bounds interrupts
4. Check is type of access allowed? (execute, read, write) yes carry on else segment protection interrupt
5. Set referenced bit always, set changed bit if write reference
6. Get location of segment L, add byte location Bto get physical memory Address A= L+B
7. Access location A.

**Advantages**

Six advantages

**Disadvantages**

1. Considerable compaction overhead
2. There is difficulty in managing variable size segments on secondary storage
3. Maximum size of segment is limited by the size of main memory
4. Constraints to prevent segment thrashing

**Segmented and Demand Paged memory management**

Instead of treating each segment as a single contiguous entity, each one can be sub divided into pages. By sphysically manipulating these pages, rather than the entire segment, problems of compaction, secondary storage handling and limitation on segment size are removed.

**Hardware Support**

The 24 bit effective address is split into three parts (1) Segment Number (2) Page Number and (3) byte number. It supports an address of up to 256 segments where each segment may consist of up to 16 pages. The mapping from virtual address to physical address is accomplished by a series of tables depicted in the **figure segment page mapping hardware**

Translation Lookaside Buffer(TLB) an associative scratch pad memory is automatically used by the hardware to eliminate most SMT and PMT access.

**Software Algorithm**

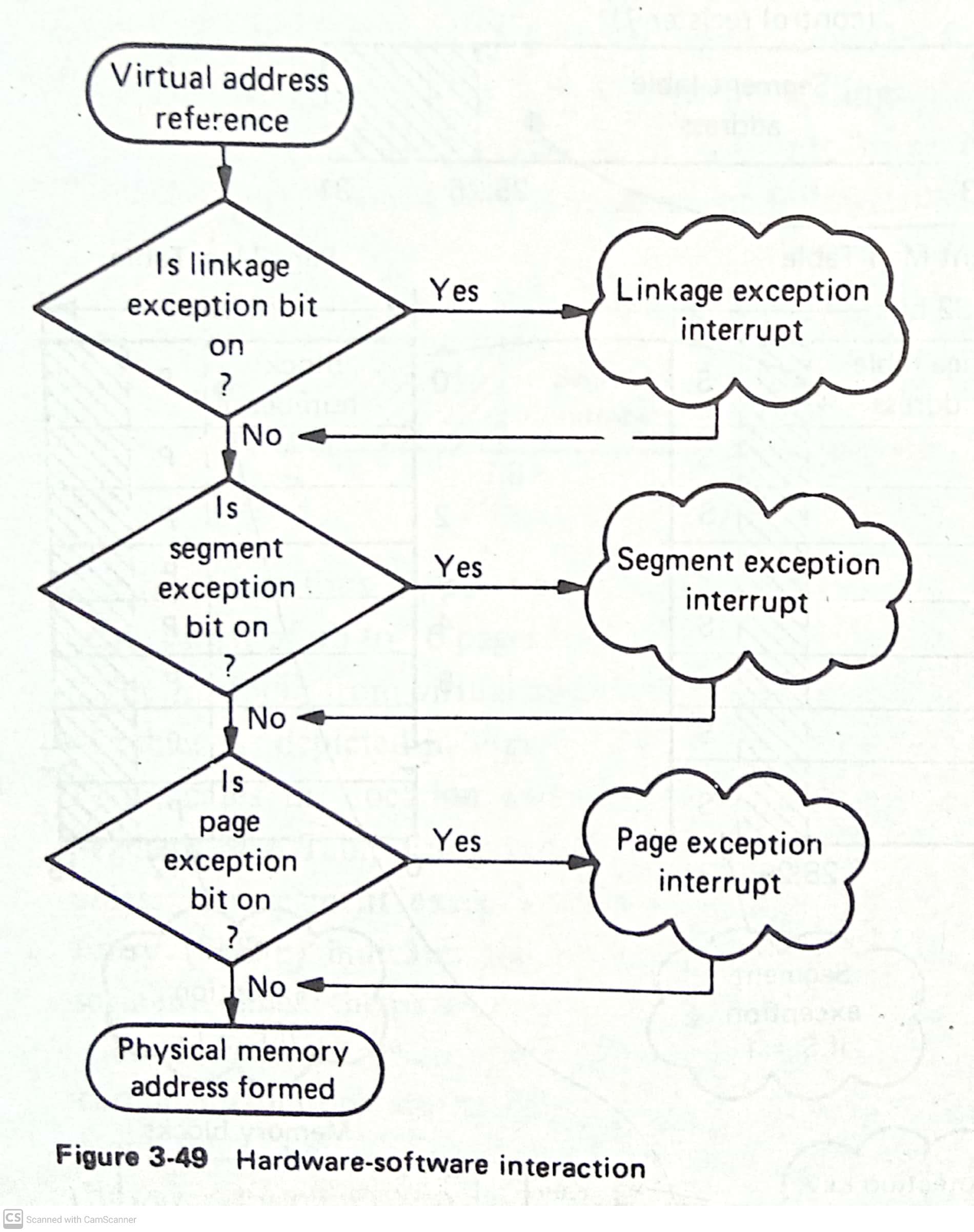
The algorithm presented in both demand paged and segmented memory management are directly applicable with small modifications. In the **figure Hardware software interaction** illustrates between the address translation hardware and the software.

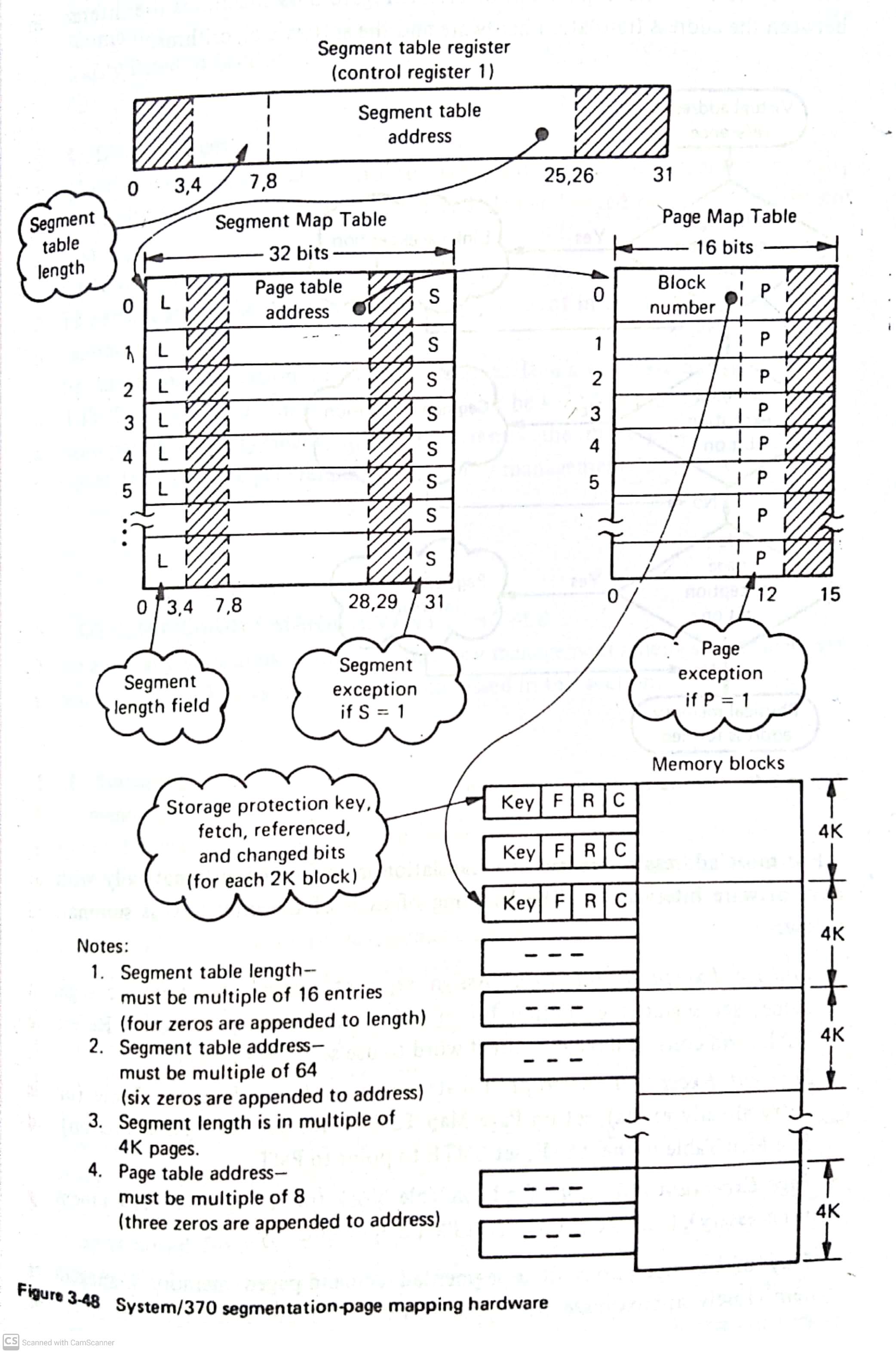
Handling of each of the interrupts are

1. Linkage exception interrupt

2. Segment exception interrupt

3. Page exception interrupt





**Advantages**

The combination of segmentation and demand paging provides all the advantages.

-Optimal utilization of all resources

- supports virtual memory

- avoids fragmentation

**Disadvantages**

Increased hardware cost, processor overhead, and complexity

**Other Memory management schemes**

**Swapping**

Many time sharing operating systems use swapping technique. One or more jobs can be removed from main memory and swapped onto secondary storage is called **roll-out**. When the high priority job has completed the low priority jobs may be reloaded into main memory is called **roll-in**

**Overlays**

A more refined form of the above swapping technique, which swaps only portion of the job’s address space is called overlay management. Overlay’s normally used in conjunction with single contiguous , partitioned or relocatable .partitioned management.. Overlay memory management is depicted in the figure.

