1**) How to speed up the work of page memory organization.**

Principles of page organization. When organizing a page, the virtual address space is divided into a number of pages of equal size. The physical address space is also broken down into individual parts the size of a page. Each such part is called a page frame.Cited in fig. 7.10 main memory contains only one page frame.

Features of page organization and memory addressing will be considered on a concrete example. We will assume (Fig. 7.10) that:

• • the virtual address space has a size of 64 K (IK = 210) and is divided into 16 pages of 4K each;

• • The physical address space of main memory is 32K in size and contains 8 4K page frames each.

Therefore, you need to map the 16-bit virtual address to the 15-bit physical address of main memory. As means for display of virtual addresses on physical addresses it is necessary to use (fig. 7.11):

• table of pages, each row of which contains:

■ presence bit, which indicates the presence / absence (1/0) of the page in main memory;

■ 3-bit code corresponding to one of the eight numbers (0, 1, ..., 7) of the page memory frame;

• Memory Management Unit (MMU), which includes:

■ register to store the table of pages;

■ input register R n, to store a 16-bit virtual address, which is divided into two parts.

If the program accesses a page that is not in the main memory, you must not only call the desired page from disk, but also free up space for it by sending another page to disk. Choosing such a page at random can lead to a number of mistakes. For example, if you delete a page with a command that caused an error, another error will occur when you try to call the next command (due to deleting the page with the failed command). Therefore, you need an algorithm to determine which page you want to delete from memory.

2) **Is it necessary to implement virtual memory in the system if it is known that the total amount of Memory required for all active processes will never exceed the amount of available physical Memory? If so, which features of the virtual memory system should be implemented and which should not?**

**3) Why should page size be a power of 2?**

An offset (lower bits of the virtual address) is added to the physical page number.

The use in paragraph 3) of the fact that the page size is equal to degree 2, allows you to use the concatenation operation (join) instead of a longer addition operation. This reduces the time to obtain a physical address, thereby increasing the performance of the computer.

The performance of the system with page memory organization is affected by the time costs associated with the processing of page interrupts and the conversion of VA into physical. With frequent page breaks, the system can spend most of its time wasting time swapping pages. To reduce the frequency of page breaks, you should increase the page size. In addition, increasing the page size reduces the size of the page table, and thus reduces memory consumption. On the other hand, if the page is large, then a large and dummy area in the last virtual page of each program. On average, each program loses half of the page volume, which in the amount of a large page can be significant. The time to convert a virtual address to a physical one is largely determined by the access time to the page table. In this regard, the table of pages tend to be placed in "fast" storage devices. This could be, for example, a set of special registers or memory that uses associative data search and caching to reduce access time.

4) **List the possible advantages of page-segment organization of memory compared to pure segmentation and purely page organization.**

Consider the main advantages of page memory organization compared to segmentation. They are determined primarily by the fact that all pages have the same length.

• Implementing memory allocation and freeing is simplified. All pages are equal in terms of process, so you can maintain a list of free pages and, if necessary, select the first page from this list, and return the page to the list after release. You can't do this with segments, because each segment can only be used for its intended purpose (trying to use a segment for a different purpose will most likely result in the need for a segment of a different length).

• Implementing data exchange with disk is also simplified. To organize such an exchange, the area on the disk that is used to store information about the pages downloaded from memory (support space, backing store) can also be divided into blocks of fixed size equal to the frame size.

Page organization of memory is not without its drawbacks.

• First of all, this approach causes internal fragmentation, due to the fact that the page size is always fixed, and if you need to allocate a memory block of a specific length, its size will be a multiple of page size. On average, the amount of unused memory is about half a page for each allocated memory block (similar to a segment). This fragmentation can be reduced by reducing the number and size of the blocks that are allocated.

• Page tables should be larger than segment tables. Thus, to allocate a continuous memory range of 100 KB, you will need one element of the segment table that describes the segment allocated for this range. On the other hand, if we use 4 KB pages to describe this range, we will need 25 page table elements - one element for each page.