Virtual memory

Virtual memory is a concept used in some large computer systems that permit the user to construct programs as though a large memory space were available, equal to the totality of auxiliary memory. Each address that is referenced by the cpu goes through an address mapping from the so called virtual address to a physical address in the main memory. Virtual memory is used to give programmers the illusion that they have a very large memory at their disposal, even though the computer

actually has a relatively small main memory. A virtual memory system provides a mechanism for translating program generated addresses into correct main memory locations. This is done dynamically, while programs are being executed in the cpu.

Address Space and Memory Space

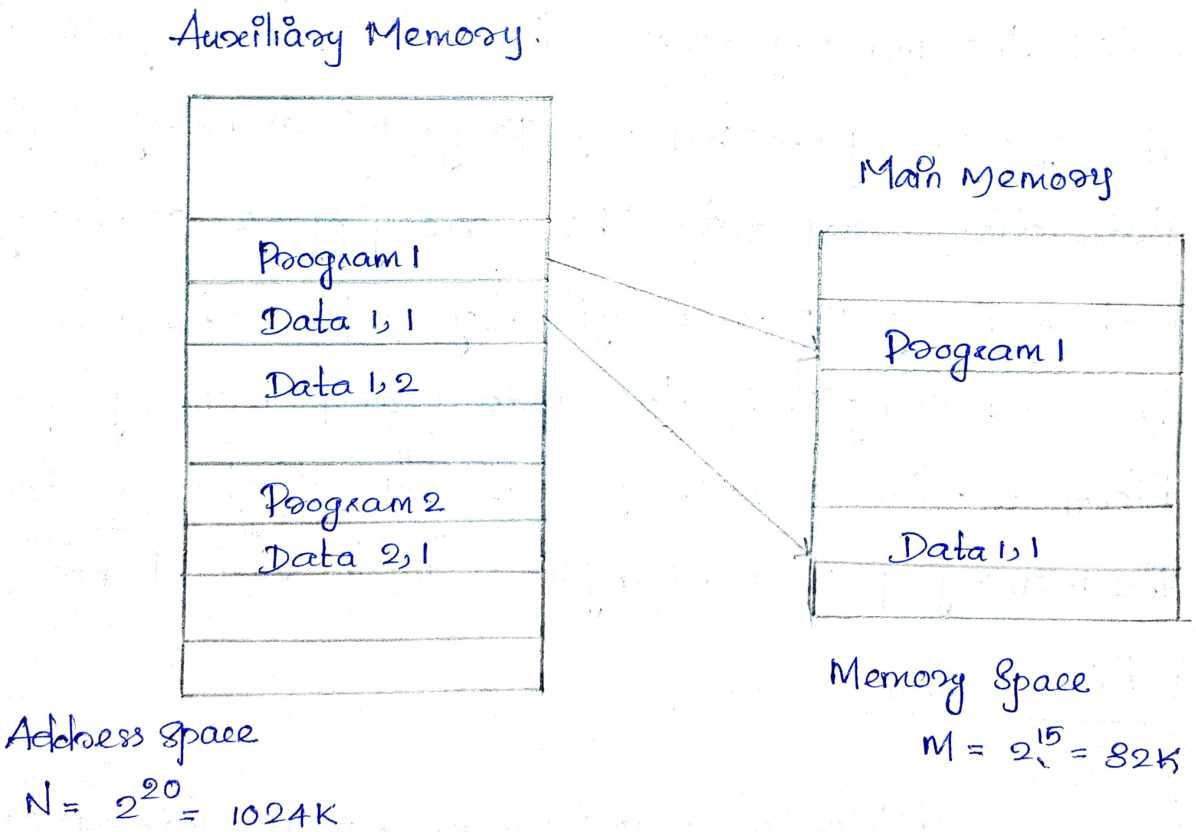
An address used by a programmer will be called a virtual address, and the set of such addresses the address space. An address in main memory is called a location or physical address. The set of such, locations is called the memory space. Thus the address space is the set of addresses generated by programs as they reference instructions and data. The memory space consists of the actual main memory locations directly addressable for processing. In most computers, the address and memory space are identical. The address space is allowed to be larger than the memory space in computers with virtual memory.

Illustration

Consider a computer with a main memory capacity of 32K words (K = 1024). 15 bits are needed to specify a physical address in memory since $32K = 2^{15}$. Suppose that the computer has available auxiliary memory for storing $2^{20} = 1024$ K words. Denoting the address space by N and the memory space by M,

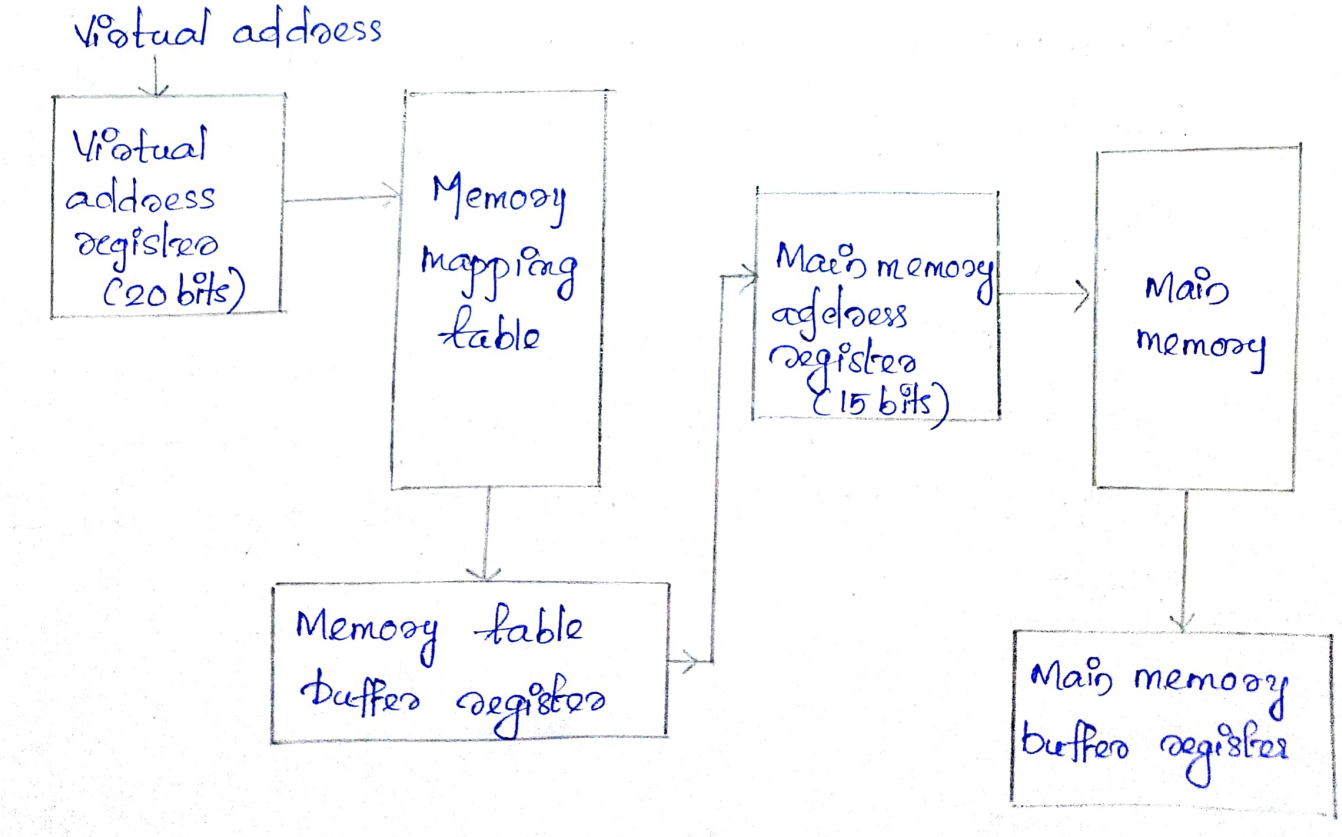
then we have N = 1024K and M = 32K.

In a multiprogram computer system, programs and data are transferred to and from auxiliary memory and main memory. Suppose that program 1 is currently being executed in the CPU. Program 1 and a portion of its associated data are moved from auxiliary memory into main memory. Portions of programs and data need not to be in contiguous locations in memory since information is being moved in and out, and empty spaces may be available in scattered locations in memory.



Relation between address and memory space in a virtual memory system

In a virtual memory system, programmers are ~~told~~ assumed that they have the total address space at their disposal. The address field of the instruction code has a sufficient number of bits to specify all virtual addresses. In the example, the address field of an instruction code will consist of 20 bits, but physical memory addresses must be specified with only 15 bits. Thus CPU will reference instructions and data with a 20 bit address, but the information at this address must be taken from physical memory because access to auxiliary storage for individual words will be prohibitively long. A table is then needed to map a virtual address of 20 bits to a physical address of 15 bits. The mapping is a dynamic operation, which means that every address is translated immediately as a word is referenced by CPU. The mapping table may be stored in a separate memory or in main memory.



Memory table for mapping a virtual address

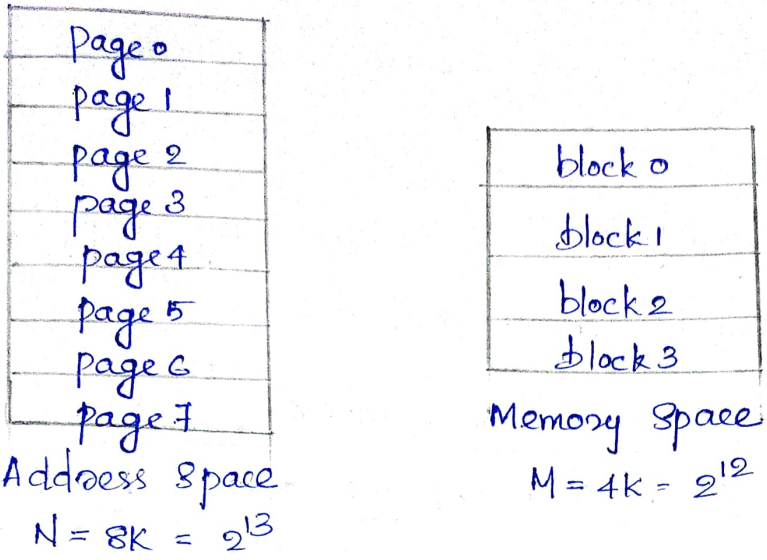
Address Mapping using pages:

The address space and memory space are each divided into group of fixed size. The physical memory is broken down into groups of equal size called blocks. The address space (virtual memory) is divided into group of same size called pages.

eg: If a page or block consist of 1K words, then using the previous example, address space is divided into 1024 pages and main memory is divided into 32 blocks. A page refers to the organization of address space, while a block refers to the organization of memory space. The programs are also considered to be split into pages. Portions of programs are moved from auxiliary memory to main memory in records equal to the size of a page. The term 'page frame' is also used to denote a block.

Paging

Consider a computer with an address space of 8K and a memory space of 4K. If we split each page into groups of 1K words we obtain eight pages and four blocks.



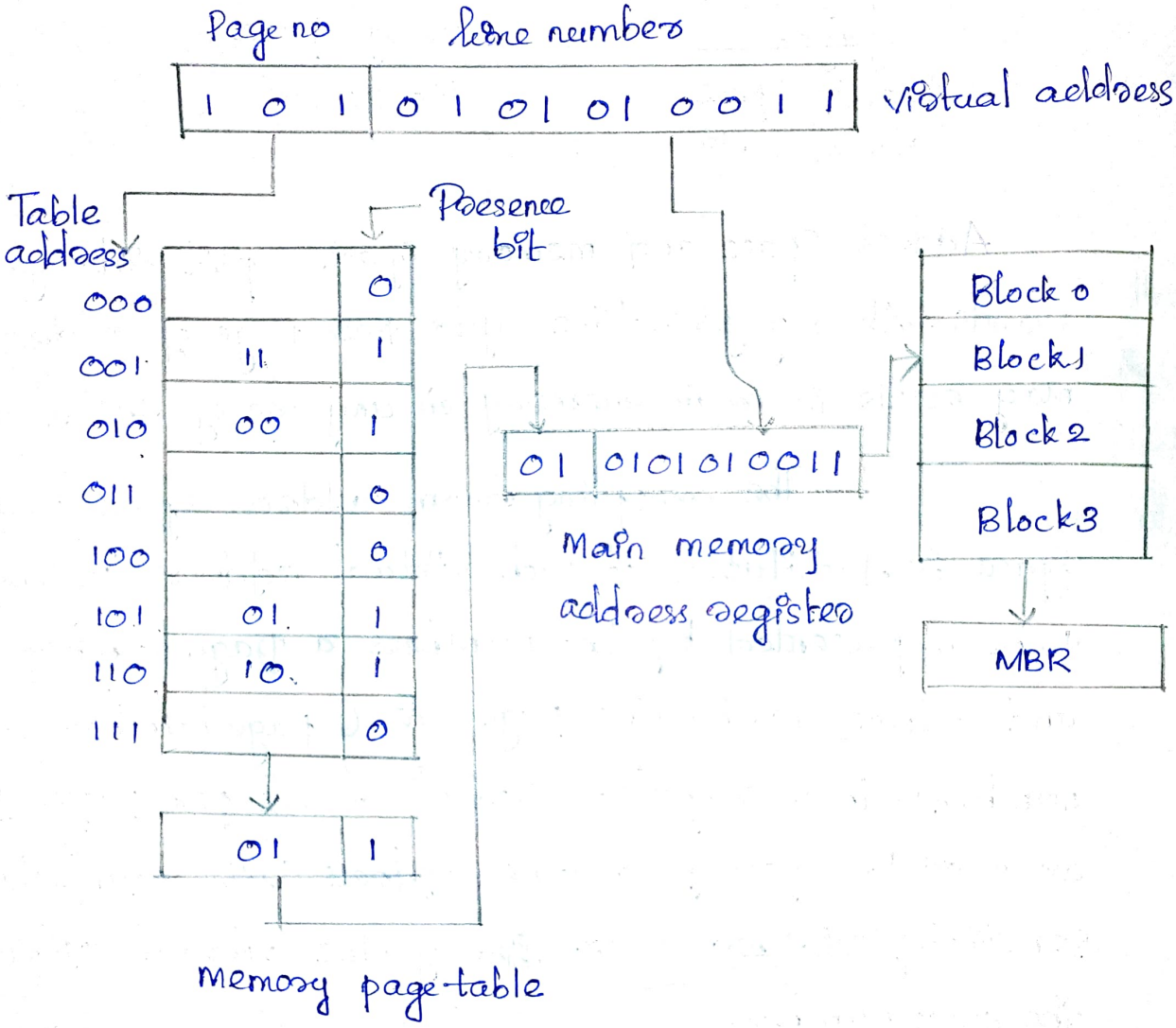
Address space and memory space split into groups of 1K words. At any given time, upto four page of address space may reside in main memory in any one of the four blocks.

The mapping from address space to memory space is facilitated if each virtual address is considered to be represented by two numbers a page number address and a line within the page. ie, (page number and line number). In a computer with 2^p words per page, p bits are used to specify a line address (line number) and the remaining higher order bits of the virtual address specify the page number.

In the example of above figure a virtual address has 13 bits. Since each page consists of 2^{10} = 1024 words, the high-order three bits of a virtual address will specify one of the eight pages and the lower order 10 bits give the line address within the page. The line address is in address space and memory space is the same, the only

mapping required is from a page number to a block number.

The organization of the memory mapping table in a paged system is shown below.



The memory page table consists of eight words one for each page. The address in the page table denotes the page number and the content of the word gives the block number where that page is stored in main memory. The table shows that pages 1, 2, 5 and 6 are now available in main memory in block 3, 0, 1 and 2 respectively. A

Presence bit in each location indicates whether the page has been transferred from auxiliary memory into main memory. A 0 in the presence bit indicates that this page is not available in main memory. The CPU references a word in memory with a virtual address of 13 bits. The three highest order bits of the virtual address specify a page number and also an address for the memory page table.

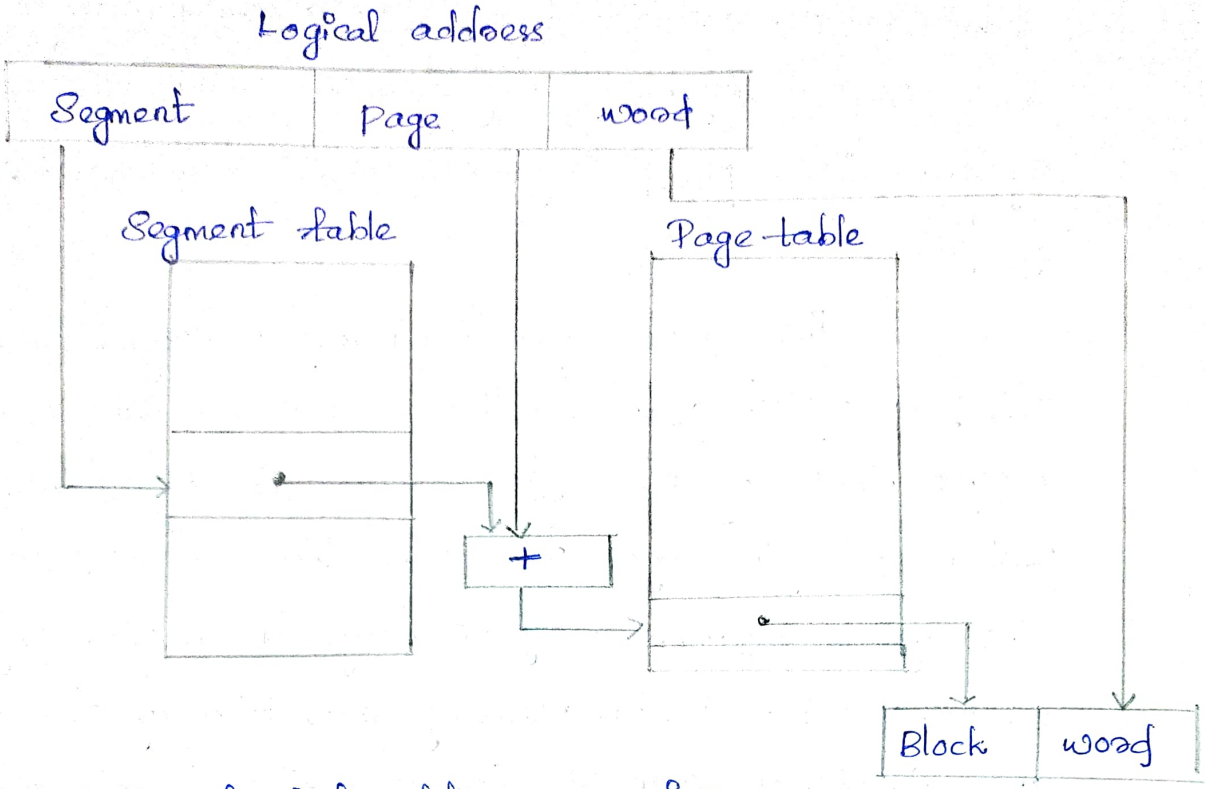
The content of the word in the memory page table at the page number address is read out into the memory table buffer register. If the presence bit is a 1, the ~~table~~ block number thus read is transferred to the two high order bits of the main memory address register. The line number from the virtual address is transferred into the 10 low order bits of the memory address register. A read signal to main memory transfers the content of the word to the main memory buffer register ready to be used by the CPU. If the presence bit in the word read from the page table is 0, it signifies that the content of the word referenced by the virtual address does not reside in main memory. A call to the operating system is then generated to fetch the required page from auxiliary memory and place it into main memory before resuming computation.

//

Segmented Page Mapping (Segmentation)

The property of logical space that it uses variable-length segments. The length of each segment is allowed to grow and contract according to the needs of the program being executed. One way of specifying the length of a segment is by associating with it a number of equal size pages.

Consider the logical address shown in the following figure.



Logical to physical address mapping

The logical address is partitioned into three fields. The Segment field specifies a Segment number. The Page field specifies the page within the Segment and the word field gives the specific word within the page. A page field of k bits can specify upto 2^k pages. A Segment number may be associated with just one page or with as many as 2^k pages. Thus the length of a Segment would vary according to the number of pages that are assigned to it.

The mapping of the logical address into a physical address is done by means of two tables, Segment table and page table. The Segment number of the logical address specifies the address for the Segment table. The entry in the Segment table is a pointer address for a page table base.

The page table base is added to the page number given in the logical address. The produces a pointer address to an entry in the page table. The value found in the page table provides the block number in physical memory. The concatenation of the block field with the word field produces the final physical mapped address.

The two mapping tables may be stored in two separate small memories or in main memory. In either case, a memory reference from the cpu will require three accesses to memory, one from the segment table, one from the page table and the third from main memory. This would slow the system significantly when compared to a conventional system that requires only one reference to the memory.

To avoid this speed penalty a fast associative memory is used to hold the most recently referenced table entries, processed in the cpu. Later when the memory block has been assigned and the transfer completed, the original program can resume its operation.