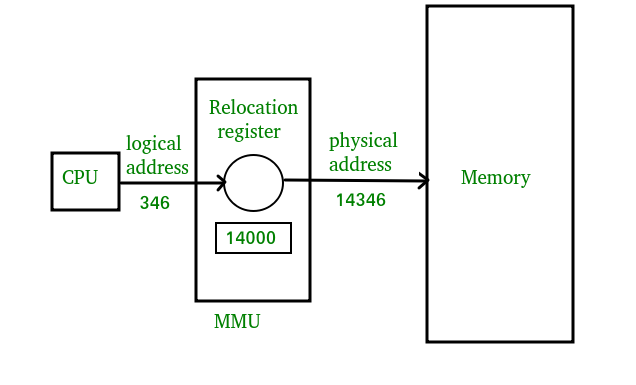
# Logical vs Physical Address in Operating System

Logical Address is generated by CPU while a program is running. The logical address is virtual address as it does not exist physically therefore it is also known as Virtual Address. This address is used as a reference to access the physical memory location by CPU. The term Logical Address Space is used for the set of all logical addresses generated by a programs perspective.  
The hardware device called Memory-Management Unit is used for mapping logical address to its corresponding physical address.

Physical Address identifies a physical location of required data in a memory. The user never directly deals with the physical address but can access by its corresponding logical address. The user program generates the logical address and thinks that the program is running in this logical address but the program needs physical memory for its execution therefore the logical address must be mapped to the physical address bu MMU before they are used. The term Physical Address Space is used for all physical addresses corresponding to the logical addresses in a Logical address space.

Differences Between Logical and Physical Address in Operating System

1. The basic difference between Logical and physical address is that Logical address is generated by CPU in perspective of a program whereas the physical address is a location that exists in the memory unit.
2. Logical Address Space is the set of all logical addresses generated by CPU for a program whereas the set of all physical address mapped to corresponding logical addresses is called Physical Address Space.
3. The logical address does not exist physically in the memory whereas physical address is a location in the memory that can be accessed physically.
4. Identical logical addresses are generated by Compile-time and Load time address binding methods whereas they differs from each other in run-time address binding method. Please refer [this](https://www.geeksforgeeks.org/memory-management-mapping-virtual-address-physical-addresses/) for details.
5. The logical address is generated by the CPU while program is running whereas the physical address is computed by the Memory Management Unit (MMU).

Comparison Chart:

| Paramenter | LOGICAL ADDRESS | PHYSICAL ADDRESS |
| --- | --- | --- |
| Basic | generated by CPU | location in a memory unit |
| Address Space | Logical Address Space is set of all logical addresses generated by CPU in reference to a program. | Physical Address is set of all physical addresses mapped to the corresponding logical addresses. |
| Visibility | User can view the logical address of a program. | User can never view physical address of program. |
| Generation | generated by the CPU | Computed by MMU |
| Access | The user can use the logical address to access the physical address. | The user can indirectly access physical address but not directly. |

## **High and Low Memory:**

The difference between logical and kernel virtual addresses is highlighted on 32-bit systems that are equipped with large amounts of memory. With 32 bits, it is possible to address 4 GB of memory. Linux on 32-bit systems has, until recently, been limited to substantially less memory than that, however, because of the way it sets up the virtual address space.

The kernel (on the x86 architecture, in the default configuration) splits the 4-GB virtual address space between user-space and the kernel; the same set of mappings is used in both contexts. A typical split dedicates 3 GB to user space, and 1 GB for kernel space.[[1](https://www.oreilly.com/library/view/linux-device-drivers/0596005903/ch15.html" \l "ftn.linuxdrive3-CHP-15-FNOTE-1)] The kernel's code and data structures must fit into that space, but the biggest consumer of kernel address space is virtual mappings for physical memory. The kernel cannot directly manipulate memory that is not mapped into the kernel's address space. The kernel, in other words, needs its own virtual address for any memory it must touch directly. Thus, for many years, the maximum amount of physical memory that could be handled by the kernel was the amount that could be mapped into the kernel's portion of the virtual address space, minus the space needed for the kernel code itself. As a result, x86-based Linux systems could work with a maximum of a little under 1 GB of physical memory.

In response to commercial pressure to support more memory while not breaking 32-bit application and the system's compatibility, the processor manufacturers have added "address extension" features to their products. The result is that, in many cases, even 32-bit processors can address more than 4 GB of physical memory. The limitation on how much memory can be directly mapped with logical addresses remains, however. Only the lowest portion of memory (up to 1 or 2 GB, depending on the hardware and the kernel configuration) has logical addresses;[[2](https://www.oreilly.com/library/view/linux-device-drivers/0596005903/ch15.html" \l "ftn.linuxdrive3-CHP-15-FNOTE-2)] the rest (high memory) does not. Before accessing a specific high-memory page, the kernel must set up an explicit virtual mapping to make that page available in the kernel's address space. Thus, many kernel data structures must be placed in low memory; high memory tends to be reserved for user-space process pages.