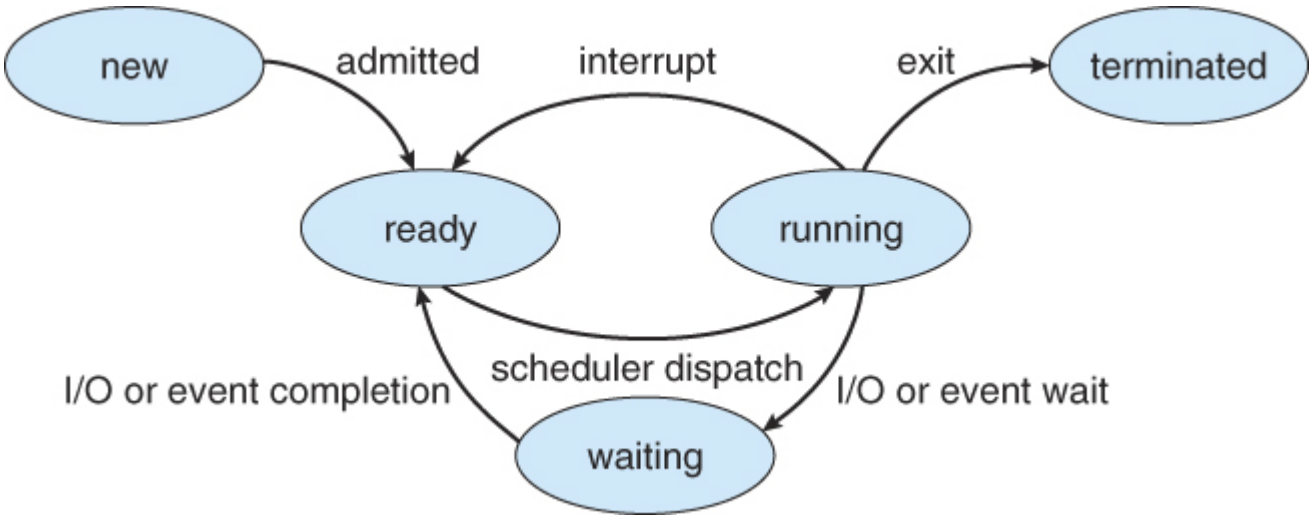
3. Processes in Operating Systems

In an operating system, a process is a fundamental concept that represents a running program. It is an instance of a program being executed, and it consists of the program's code, data, and resources allocated by the operating system. Processes are essential for multitasking, allowing multiple programs to run concurrently on a single computer.

Process States

A process can be in one of several states throughout its lifetime:

1. **New**: The process is being created.
2. **Ready**: The process is waiting to be assigned to a processor.
3. **Running**: The process is being executed by a processor.
4. **Waiting** (or **Blocked**): The process is waiting for some event to occur, such as I/O completion or a signal from another process.
5. **Terminated**: The process has finished execution.

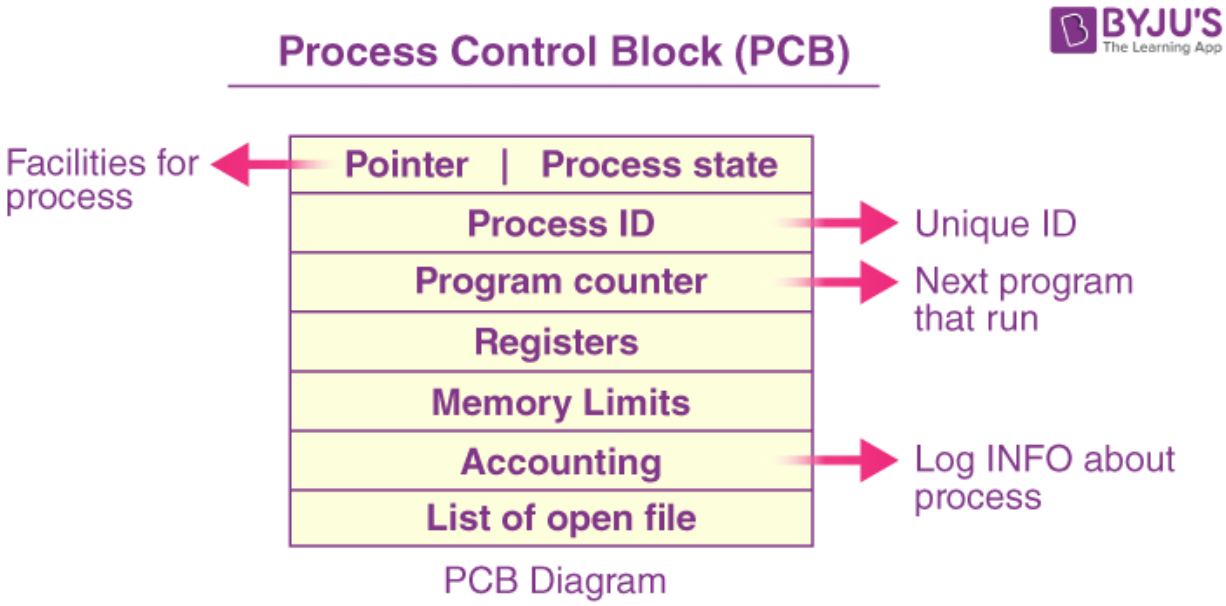


Process Control Block (PCB)

The operating system maintains a data structure called the Process Control Block (PCB) for each process. The PCB contains information about the process, including:

- Process ID (PID)
- Process state
- Program counter
- CPU registers

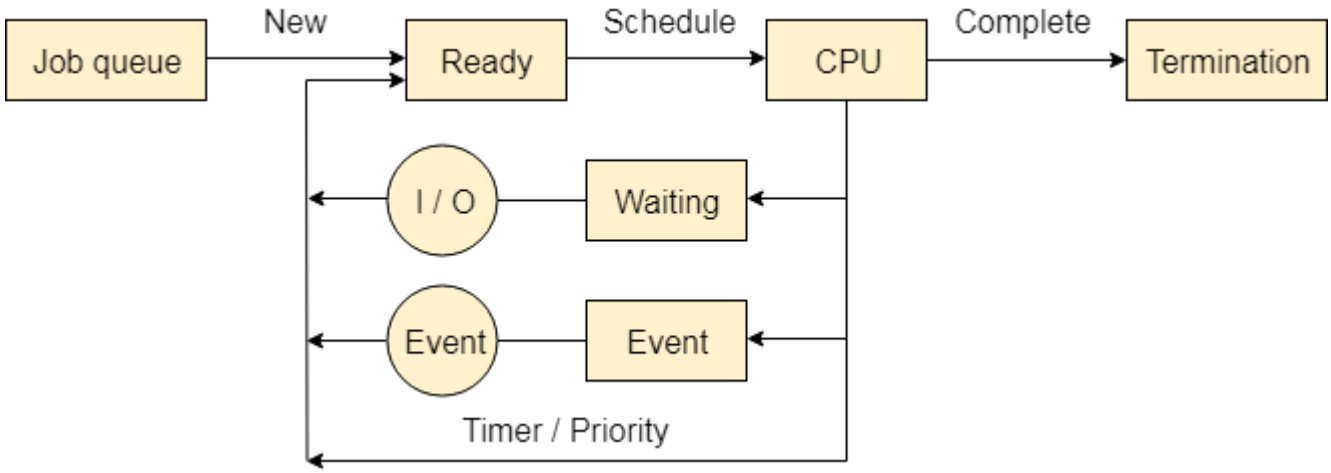
- CPU scheduling information
- Memory management information
- I/O status information
- Accounting information



Process Queue

The operating system maintains several queues to manage processes efficiently:

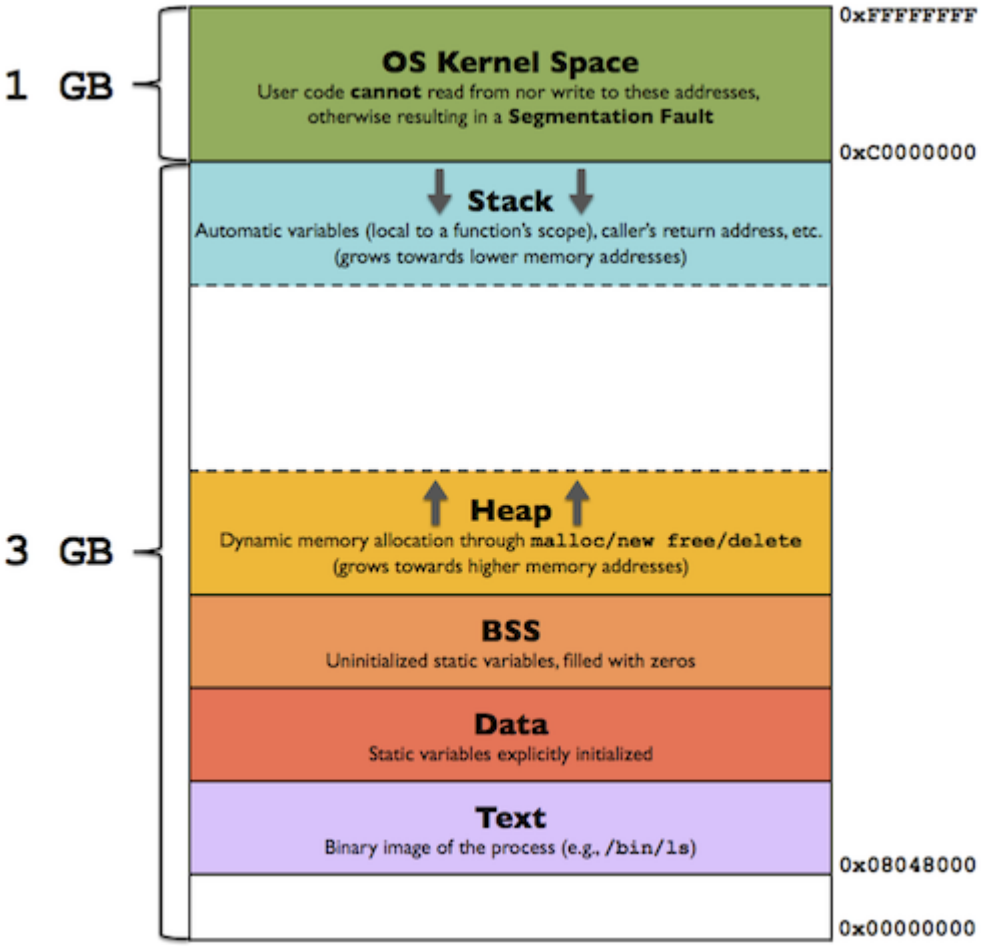
1. **Job Queue**: This queue holds all the processes in the system.
2. **Ready Queue**: This queue holds the processes that are ready to be executed and waiting for the CPU.
3. **Device Queues**: These queues hold the processes waiting for a particular I/O device.



Process Layout in Memory

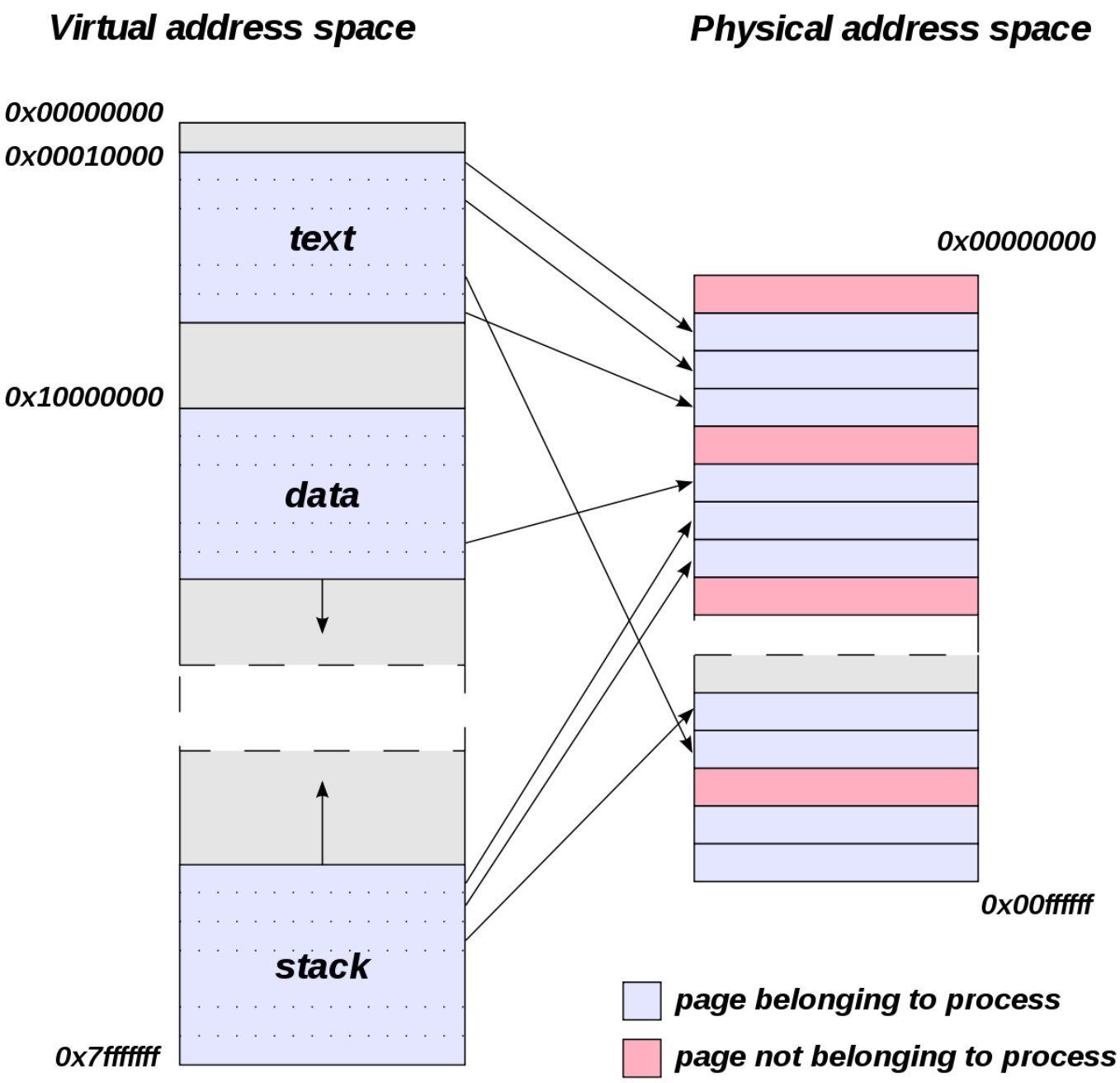
A process's memory layout typically consists of the following segments:

1. **Text**: This segment contains the program's executable code.
2. **Data**: This segment contains the initialized and uninitialized data variables.
3. **Heap**: This segment is used for dynamic memory allocation during runtime.
4. **Stack**: This segment is used for local variables, function parameters, and return addresses.



Processes and Memory: Paging

Paging is a memory management technique used by operating systems to efficiently manage memory for processes. In paging, the physical memory is divided into fixed-size blocks called frames, and the process's virtual memory is divided into blocks of the same size called pages. The operating system maintains a page table for each process, which maps the virtual pages to physical frames.



Servers, Services, and Daemons (Unix/Linux)

In Unix and Linux systems, processes that run in the background and provide specific services are called daemons. These processes are usually started during system startup and continue running until the system is shut down. Some examples of daemons include:

- **httpd**: The Apache web server daemon that handles HTTP requests.
- **sshd**: The SSH daemon that enables secure remote access to the system.
- **cron**: The daemon responsible for executing scheduled tasks.

Services are programs that provide specific functionalities to other programs or users. They can be started, stopped, and managed using the appropriate commands or system utilities.

See Also 4. How a computer starts up (Raspberry Pi)