Chapter : Process Management

Chapter : Processes

- Process Concept
- Process Scheduling
- Operations on Processes

Process

- Process – a program in execution
- Program becomes process when executable file loaded into memory
- Process execution must progress in sequential manner
- A process includes:
  - program counter
  - Stack
  - heap
  - data section

Process

- **Process memory** is divided into four sections for efficient working :
- The **Text section** is made up of the compiled program code
- The **Data section** is made up of the global and static variables
- The **Heap** is used for the dynamic memory allocation
- The **Stack** is used for local variables.

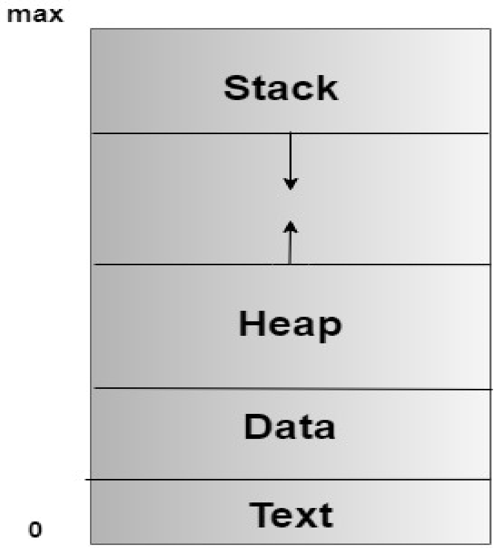
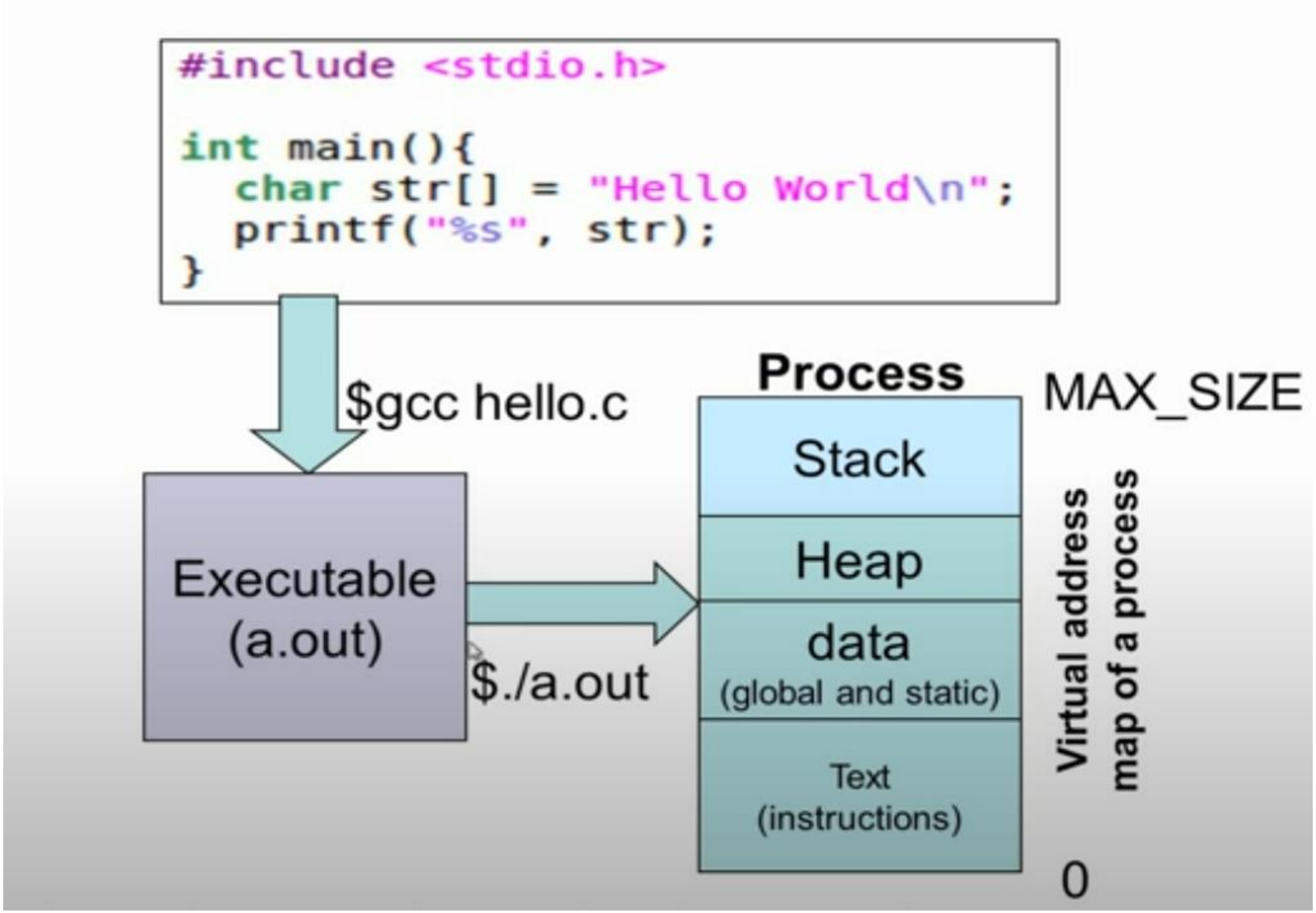


Figure: Process in the Memory



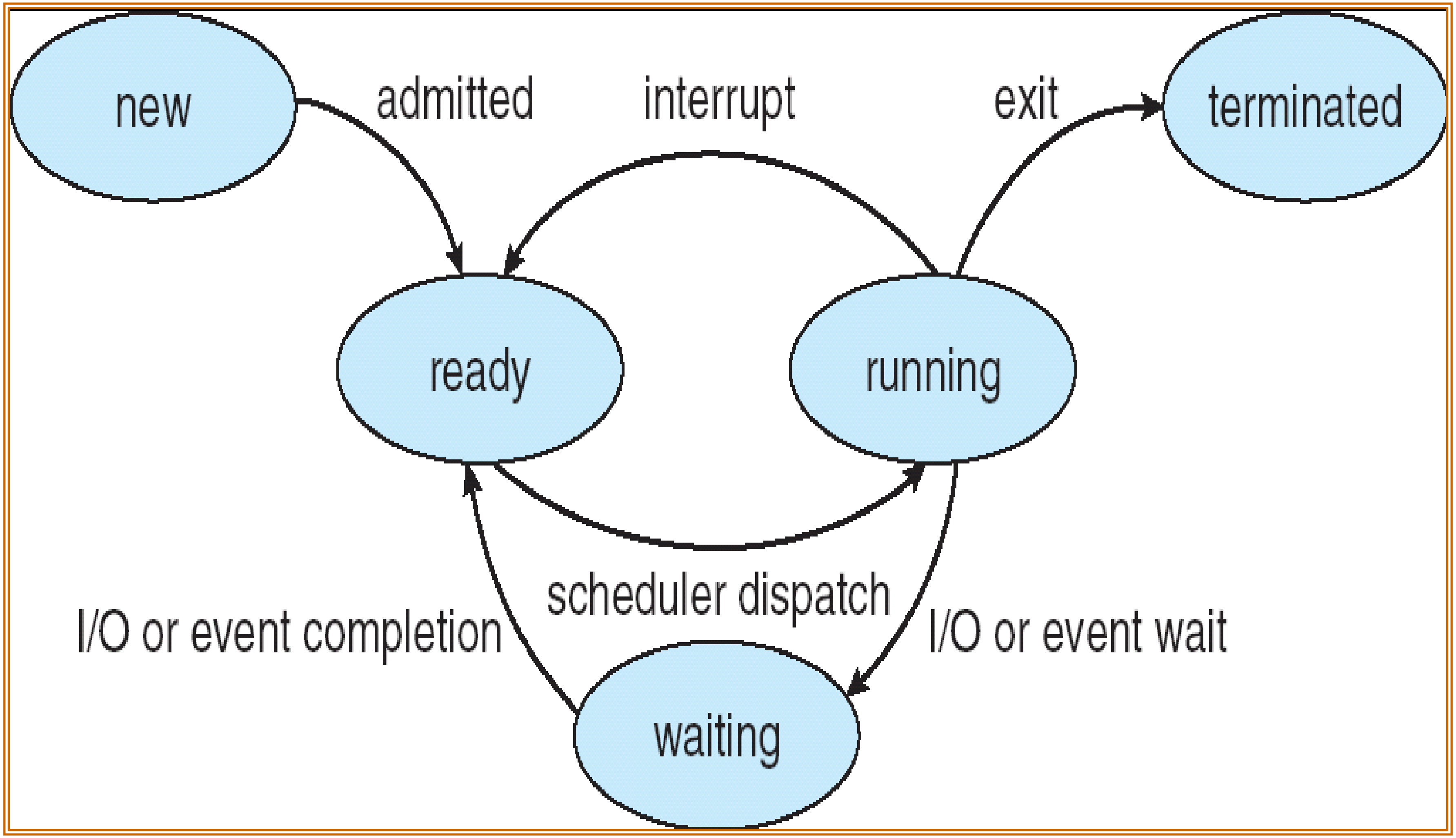
```
#include <stdio.h>

int main(){
  char str[] = "Hello World\n";
  printf("%s", str);
}
```

Process States

- As a process executes, it changes *state*
  - **new**: The process is being created
  - **ready**: The process is waiting to be assigned to a processor and is ready to get executed
  - **running**: Instructions are being executed
  - **waiting**: The process is waiting for some event to occur/ waiting for resources
  - **terminated**: The process has finished execution

Process States



Process Control Block (PCB)

- **Process Control Block**
- (PCB, also called Task Controlling Block) is a data structure in the operating system kernel containing the information needed to manage a particular **process**.
- All the information about the process we have present in PCB i.e. Process Control Block
- A data structure used in the OS to represent one single process

Process Control Block

The role of the process control block arises as an identification card for each process. The Operating System doesn't know which process is which, until Operating System refers through the PCB of every process

.

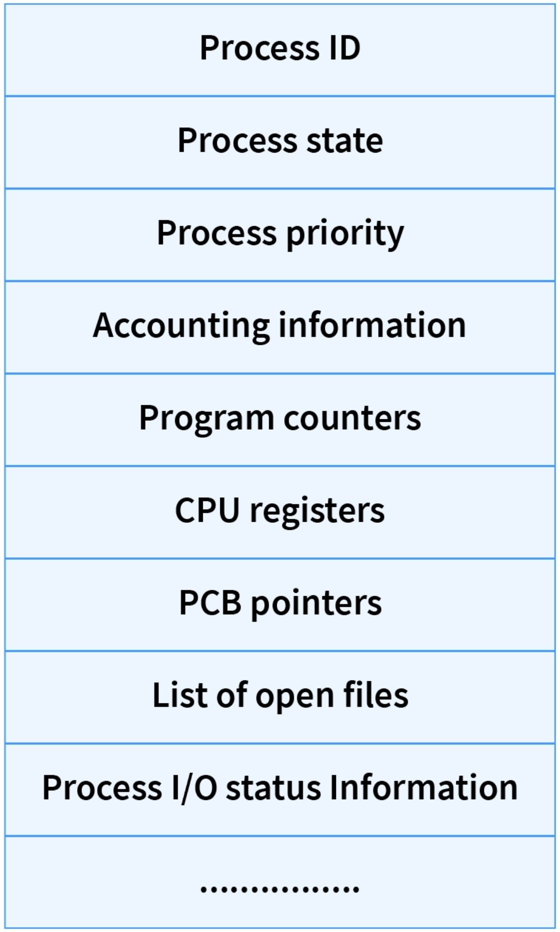
For Example:, there are MS word processes, pdf processes, printing processes, and many background processes are running currently on the CPU.

How will OS identify and manage each process without knowing the identity of each process?

So, here PCB comes into play as a data structure to store information about each process.

Therefore, whenever a process (like print command), a process control block (PCB) is created for that process in the operating system which is used by the operating system to execute and manage the processes when the operating system is free

Process Control Block (PCB)



Process ID

Process state

Process priority

Accounting information

Program counters

CPU registers

PCB pointers

List of open files

Process I/O status Information

....

Process Control Block (PCB)

Information associated with each process:

1. **Process State-** new, ready….etc.
2. **Pointer-** to the parent process
3. **Program Counter-** next instruction to be executed.
4. **Process ID- Unique identification** number in OS
5. **CPU Registers-** Various CPU registers **where process** need to be **stored for execution** for running state.
6. **CPU Scheduling Information-** Process Priority and other info. required for scheduling

Process Control Block (PCB)

7. **Memory-Management Information-** Registers, Page tables (where process is saved)used by OS

8. **Accounting Information-** For how long CPU and other resources are allocated to process

9. **I/O Status Information-** List of devices allocated to the process

Process Control Block

- **Process State**
- This specifies the process state i.e. new, ready, running, waiting or terminated.
- **Process ID**
- This shows the number of the particular process.
- **Program Counter**
- This contains the address of the next instruction that needs to be executed in the process.
- **Registers**
- This specifies the registers that are used by the process. They may include accumulators, index registers, stack pointers, general purpose registers etc.

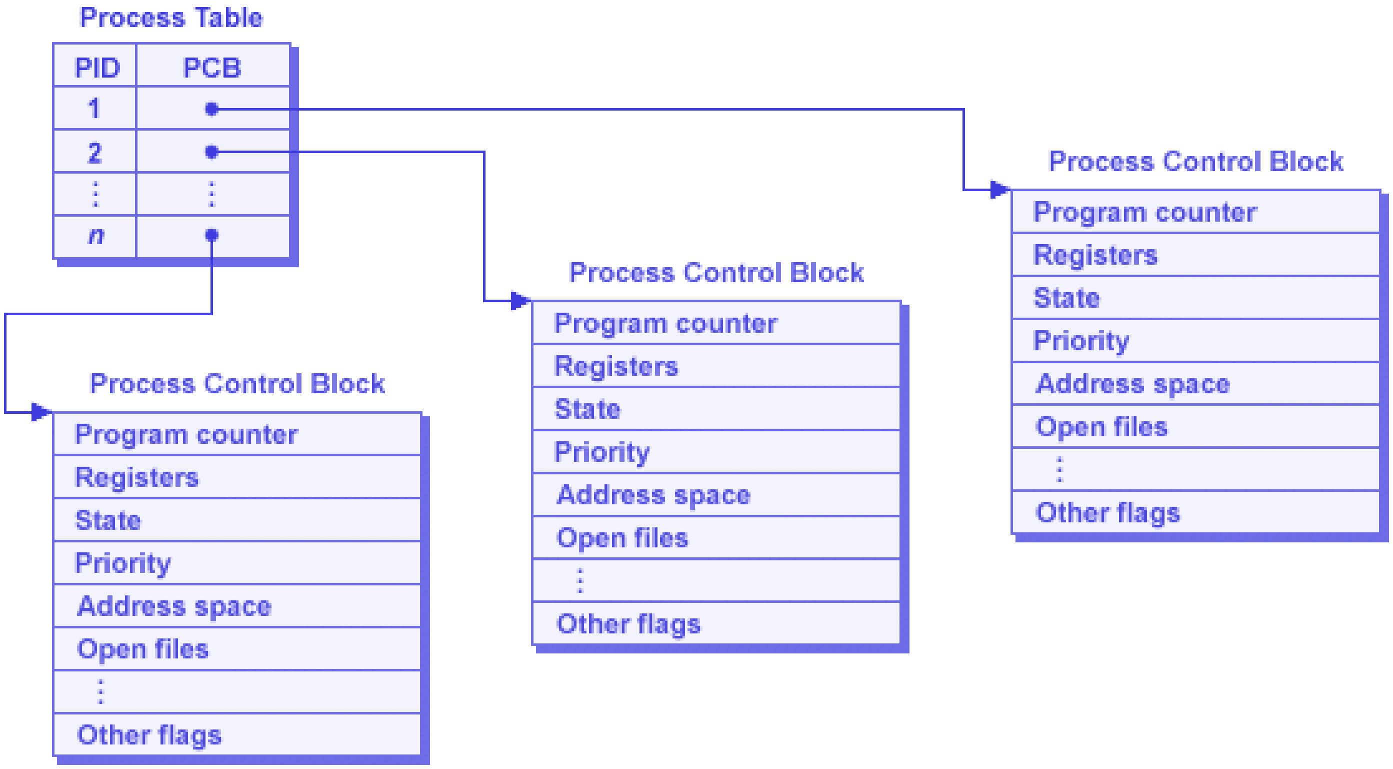
Process Control Block

- **List of Open Files**
- These are the different files that are associated with the process
- **CPU Scheduling Information**
- The process priority, pointers to scheduling queues etc. is the CPU scheduling information that is contained in the PCB. This may also include any other scheduling parameters.
- **Memory Management Information**
- The memory management information includes the page tables or the segment tables depending on the memory system used.

Process Control Block

- **I/O Status Information**
- This information includes the list of I/O devices used by the process, the list of files etc.
- **Accounting information**
- The time limits, account numbers, amount of CPU used, process numbers etc. are all a part of the PCB accounting information.
- Location of the Process Control Block
- The process control block is kept in a memory area that is protected from the normal user access. This is done because it contains important process information. Some of the operating systems place the PCB at the beginning of the kernel stack for the process as it is a safe location.

Operating System maintain PCB of each process using link list data structure



Process Table

PID	PCB
1	●
2	●
⋮	⋮
n	●

Process Control Block

Program counter
Registers
State
Priority
Address space
Open files
⋮
Other flags

Process Control Block

Program counter
Registers
State
Priority
Address space
Open files
⋮
Other flags

Process Control Block

Program counter
Registers
State
Priority
Address space
Open files
⋮
Other flags

Process

System must provide mechanisms for:

- Process creation
- Process termination

Creation

- This is the initial step of process execution activity. Process creation means the construction of a new process for the execution. This might be performed by system, user or old process itself. There are several events that leads to the process creation. Some of the such events are following:
- When we start the computer, system creates several background processes.
- A user may request to create a new process.
- A process can create a new process itself while executing.

Process Creation

- **Parent** process create **children** processes, which, in turn create other processes, forming a **tree** of processes
- Generally, process identified and managed via a **process identifier** (**pid**)
- Resource sharing options
  - Parent and children share all resources
  - Children share subset of parent’s resources
  - Parent and child share no resources
- Execution options
  - Parent and children execute concurrently
  - Parent waits until children terminate

Process Termination

- Process termination occurs when the process is terminated The exit() system call is used by most operating systems for process termination.

- Some of the causes of process termination are as follows –

A process may be terminated after its execution is naturally completed. This process leaves the processor and releases all its resources.

A child process may be terminated if its parent process requests for its termination.

A process can be terminated if it tries to use a resource that it is not allowed to. For example - A process can be terminated for trying to write into a read only file.

- If an I/O failure occurs for a process, it can be terminated. For example - If a process requires the printer and it is not working, then the process will be terminated.

- In most cases, if a parent process is terminated then its child processes are also terminated. This is done because the child process cannot exist without the parent process.

- If a process requires more memory than is currently available in the system, then it is terminated because of memory scarcity.