PROCESS

1

What is Process?

- Process is a program under execution.
- Process is an abstraction of a running program.
- Process is an instance of an executing program, including the current values of the program counter, registers & variables.
- Each process has its own virtual CPU.

2

2

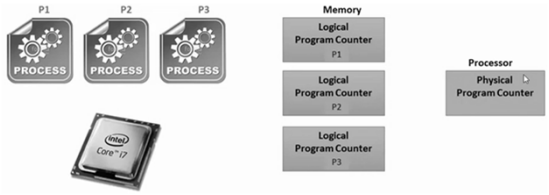
Multiprogramming

- The real CPU switches back and forth from process to process.
- This rapid switching back and forth is called multiprogramming.
- The number of process loaded simultaneously in memory is called degree of multiprogramming

3

3

Multiprogramming execution

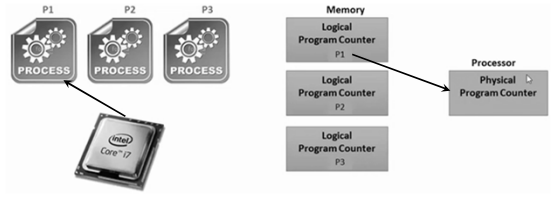


- There are 3 processes, 1 processor (CPU), 3 logical program counter (one for each process) in memory and 1 physical program counter in processor.
- Here CPU is free.
- No data in physical program counter.

4

4

Multiprogramming execution

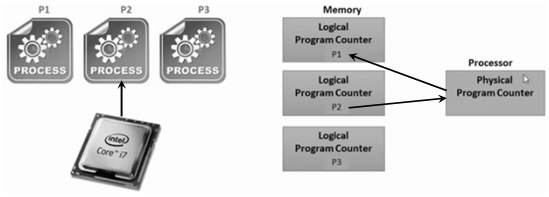


- CPU is allocated to process P1 (process P1 is running).
- Data of process P1 is copied from its logical program counter to the physical program counter.

5

5

Multiprogramming execution

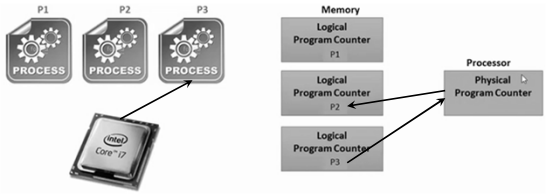


- CPU switches from process P1 to process P2.
- CPU is allocated to process P2 (process P2 is running).
- Data of process P1 is copied back to its logical program counter.
- Data of process P2 is copied from its logical program counter to the physical program counter.

6

6

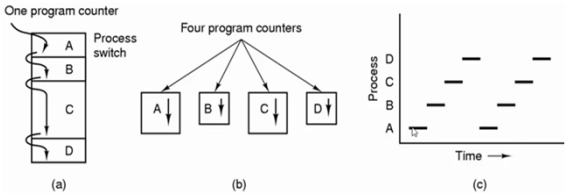
Multiprogramming execution



- CPU switches from process P2 to process P3.
- CPU is allocated to process P3 (process P3 is running).
- Data of process P2 is copied back to its logical program counter.
- Data of process P3 is copied from its logical program counter to the physical program counter.

7

Process Model



- Fig.(a) Multiprogramming of four programs in memory.
- Fig.(b) Conceptual model of 4 independent, sequential processes, each with its own flow of control ()i.e., its own logical program counter) and each one running independently of the other ones.
- Fig.(c) over a long period of time interval, all the processes have made progress, but at any given instant only one process is actually running.

8

PROCESS CONTROL BLOCK (PCB)

Process state –
- running, waiting, etc

Program counter –
- location of instruction to next execute

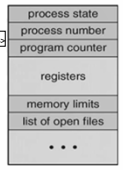
CPU registers –
- contents of all process-centric registers

CPU scheduling information-
- priorities, scheduling queue pointers

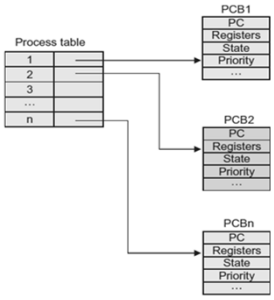
Memory-management information –
- memory allocated to the process

Accounting information –
- CPU used, clock time elapsed since start, time limits

I/O status information –
- I/O devices allocated to process, list of open files

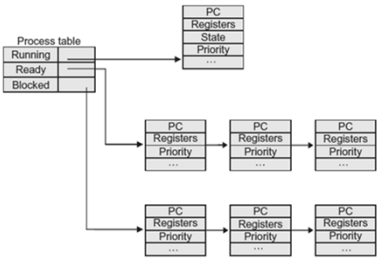


9



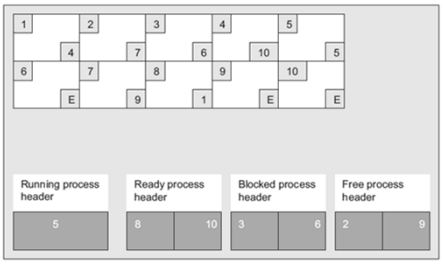
Process Table and the PCB

10



Separate queues for different states of processes

11



PCB queues in memory

12

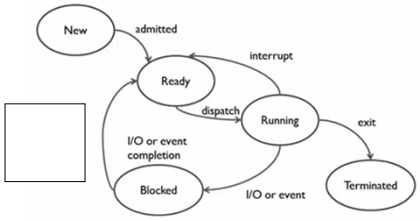
PROCESS STATES

- As a process executes, it changes **state**
  - **new**: The process is being created
  - **ready**: The process is waiting to be assigned to a processor
  - **running**: Instructions are being executed
  - **waiting**: The process is waiting for some event to occur (Waiting = blocked)
  - **terminated**: The process has finished execution

13

13

PROCESS STATES



14

14

PROCESS SCHEDULING

- Maximize CPU use, quickly switch processes onto CPU for time sharing
- System queues:
  - **Job queue** – set of all processes in the system
  - **Ready queue** – set of all processes residing in main memory, ready and waiting to execute
  - **Device queues** – set of processes waiting for an I/O device
- Processes migrate among the various queues

15

15

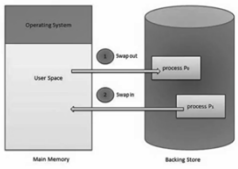
SCHEDULERS

- **Long-term scheduler** (or **job scheduler**)
  - Selects which processes should be brought into the ready queue - invoked infrequently
- **Short-term scheduler** (or **CPU scheduler**)
  - Selects which process should be executed next and allocates CPU- invoked very frequently ⇒ (must be fast)
  - Sometimes the only scheduler in a system
- **Medium-term scheduler** can be added
  - Remove process from memory, store on disk, bring back in from disk to continue execution: **swapping**

16

16

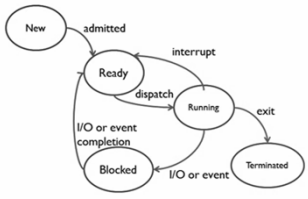
MEDIUM TERM SCHEDULING (SWAPPING)



17

17

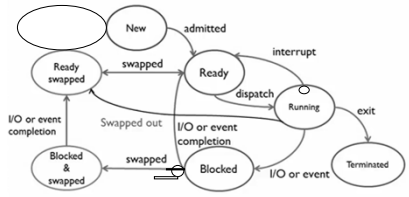
MEDIUM TERM SCHEDULING (SWAPPING)



18

18

MEDIUM TERM SCHEDULING (SWAPPING)



19

PROCESS DEFINITION

- Processes can be described as either type:
  - **I/O-bound process** – spends more time doing I/O than computations, many short CPU bursts
  - **CPU-bound process** – spends more time doing computations; few very long CPU bursts
- Long-term scheduler strives for good ***process mix***
- The long-term and medium term scheduler control the **degree of multiprogramming** (number and type of active programs)

20

CONTEXT SWITCH

- When CPU switches to another process
  - save the state of the old process and
  - load the saved state for the new process via a context switch
- **Context** of a process represented in the PCB
- Context-switch time is overhead
  - The system does no useful work while switching
  - The more complex the OS and the PCB
    - Longer the context switch
- Time dependent on hardware support
  - Some hardware provides multiple sets of registers per CPU -> multiple contexts loaded at once

21

OPERATIONS ON PROCESSES

- Creation
- Termination
- Block
- Wake up
- Change priority
- Dispatch

22

PROCESS CREATION

- **Parent** process creates **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

23

PROCESS TERMINATION

- **Voluntary**
  - Normal exit
  - Internal error or exception
    - Example: exit if no input file is found
- **Involuntary**
  - Fatal error
    - Example: divide by zero/ illegal memory access
  - Explicitly killed by another process
    - Example: task manager

24

PROCESS TERMINATION

Process termination

- Process executes last statement and asks the operating system to delete it
- Output data from child to parent
- Process' resources are deallocated by operating system

- Parent may terminate execution of children processes
  - Child has exceeded allocated resources
  - Task assigned to child is no longer required
  - If parent is exiting
    - Some operating systems do not allow child to continue if its parent terminates
      - All children terminated - **cascading termination**

- If no parent waiting, then terminated process is a zombie
- If parent terminated, processes are orphans

25

25

INTERPROCESS COMMUNICATION

- Processes within a system may be ***independent*** or ***cooperating***
- Cooperating process can affect or be affected by other processes, including sharing data
- Reasons for cooperating processes:
  - Information sharing
  - Computation speedup
  - Modularity
  - Convenience
- Cooperating processes need **interprocess communication (IPC)**
- Two models of IPC
  - **Shared memory**
  - **Message passing**

26

26