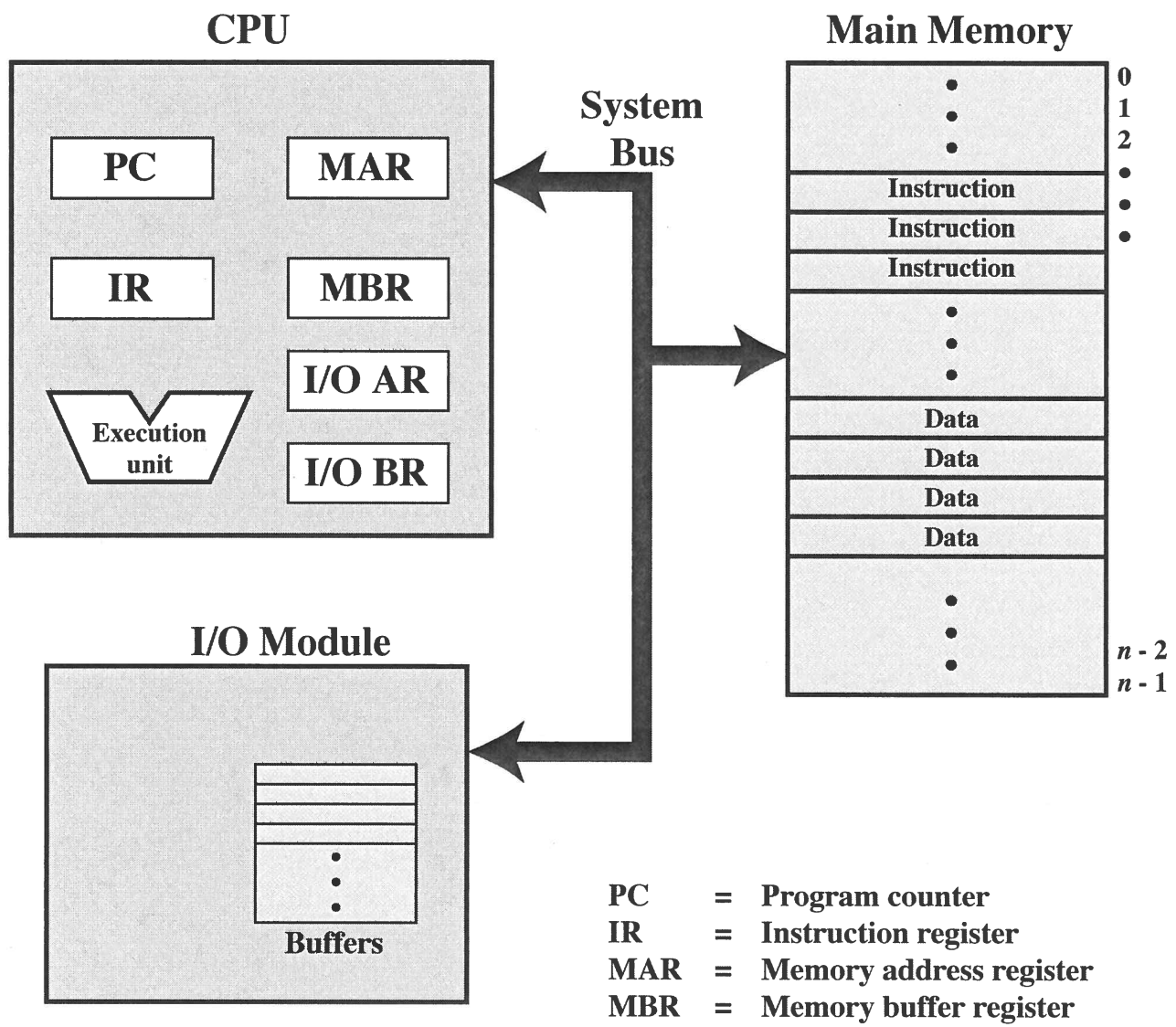


PC = Program counter
IR = Instruction register
MAR = Memory address register
MBR = Memory buffer register
I/O AR = Input/output address register
I/O BR = Input/output buffer register

Process Control Block

- Contains the process elements
- It is possible to interrupt a running process and later resume execution as if the interruption had not occurred
- Created and managed by the operating system
- Key tool that allows support for multiple processes

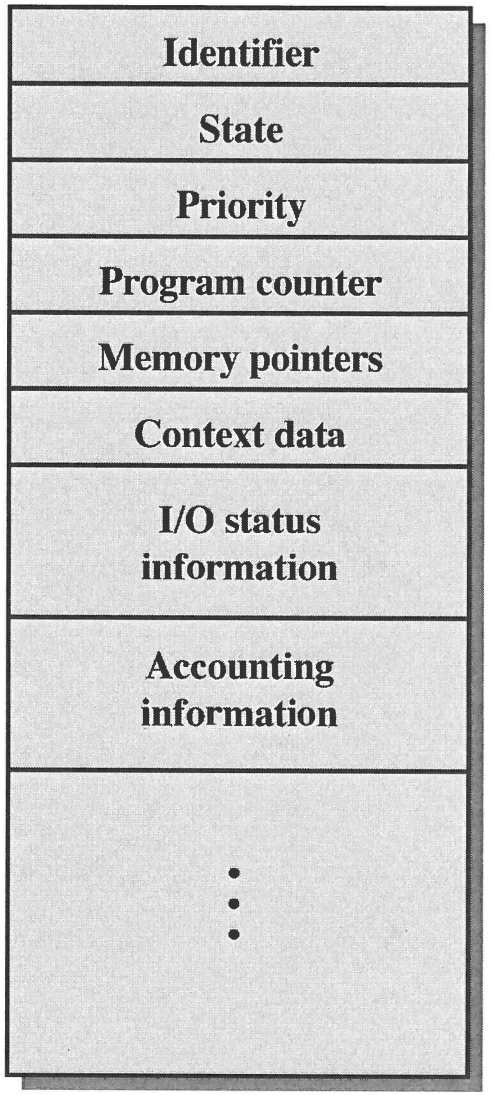
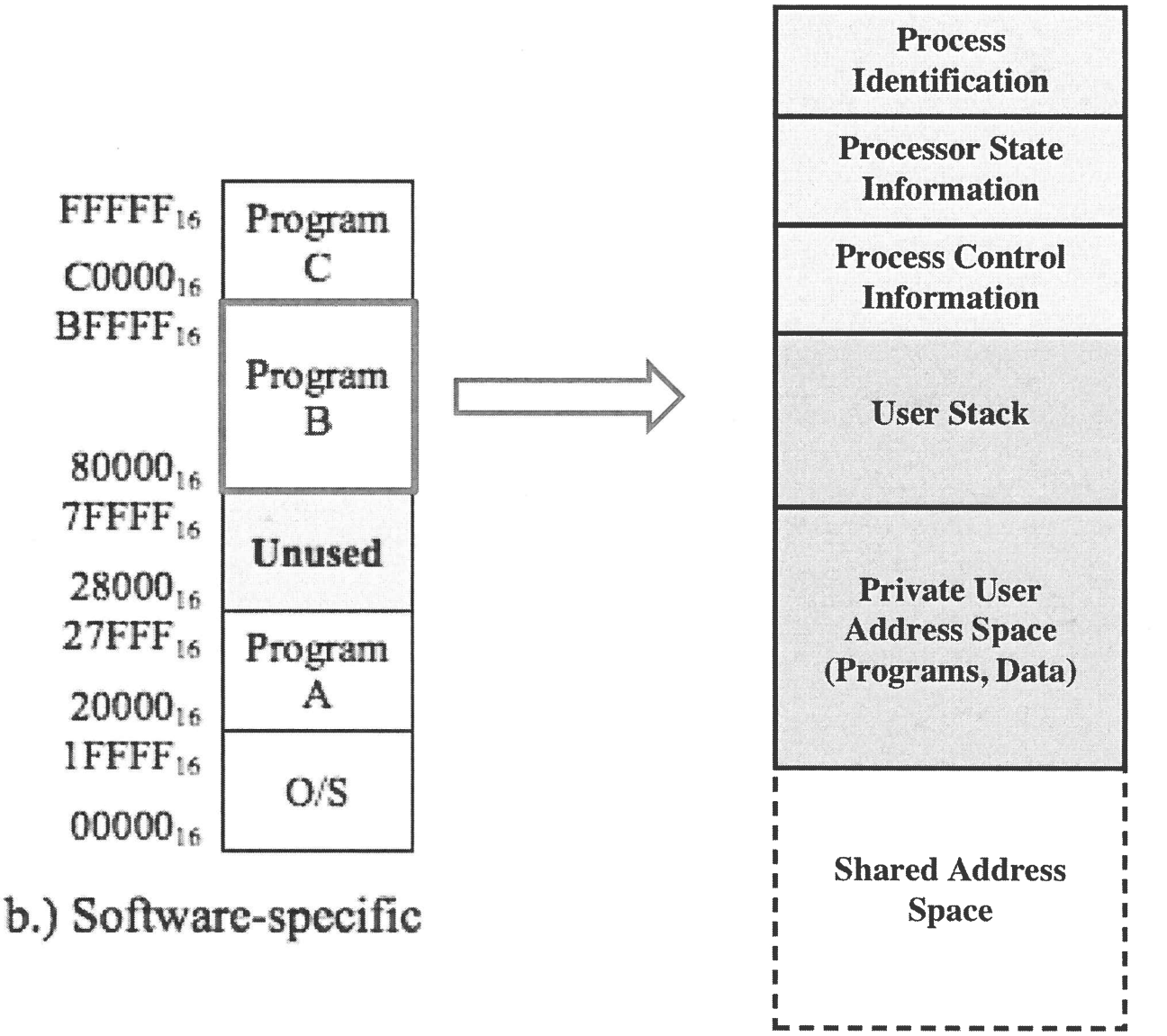
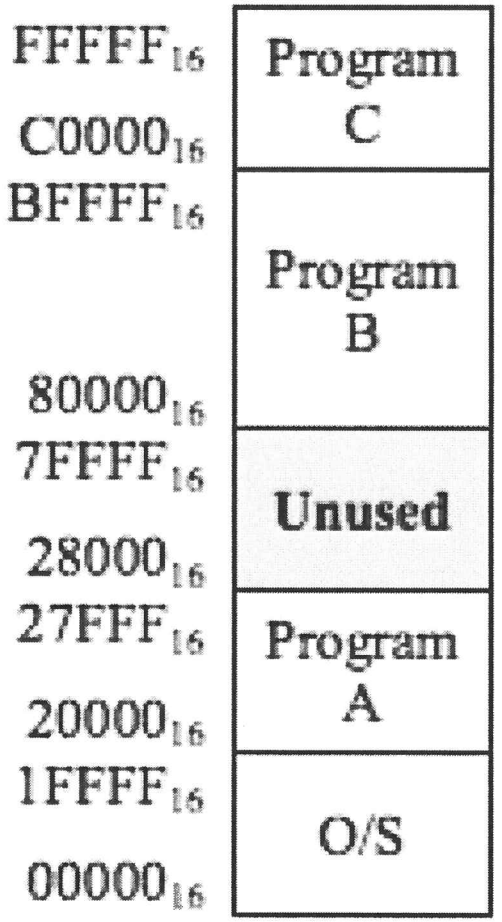


Figure 3.1 Simplified Process Control Block

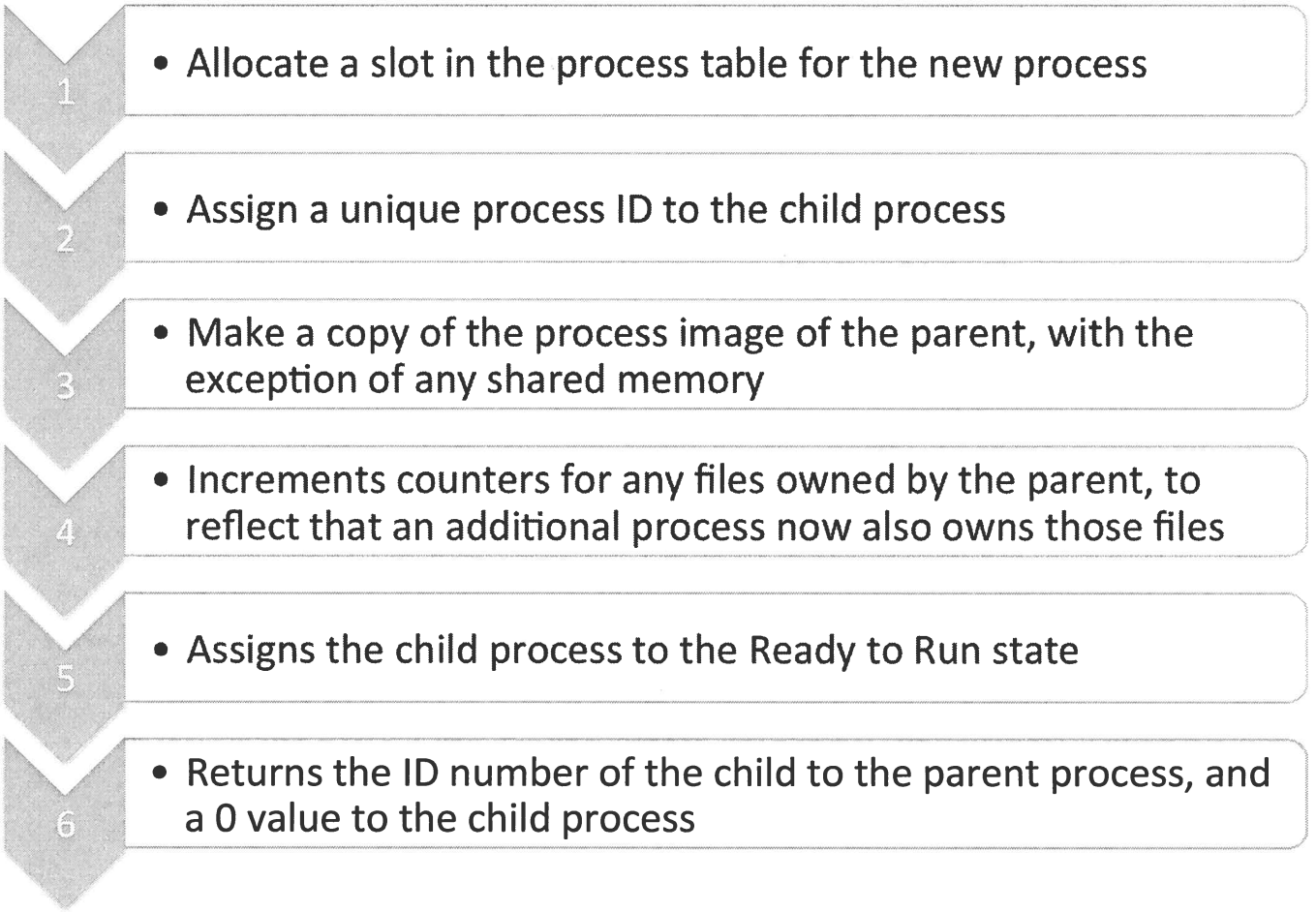

FFFFF16
C0000₁₆
BFFFF16
80000₁₆
7FFFF16
28000₁₆
27FFF16
20000₁₆
1FFFF16
00000₁₆
Program C
Program B
Unused
Program A
O/S
b.) Software-specific
Process Identification
Processor State Information
Process Control Information
User Stack
Private User Address Space (Programs, Data)
Shared Address Space
Program B (Process B)



b.) Software-specific

Process Creation

- Process creation is by means of the kernel system call, fork ( )
- This causes the OS, in Kernel Mode, to:



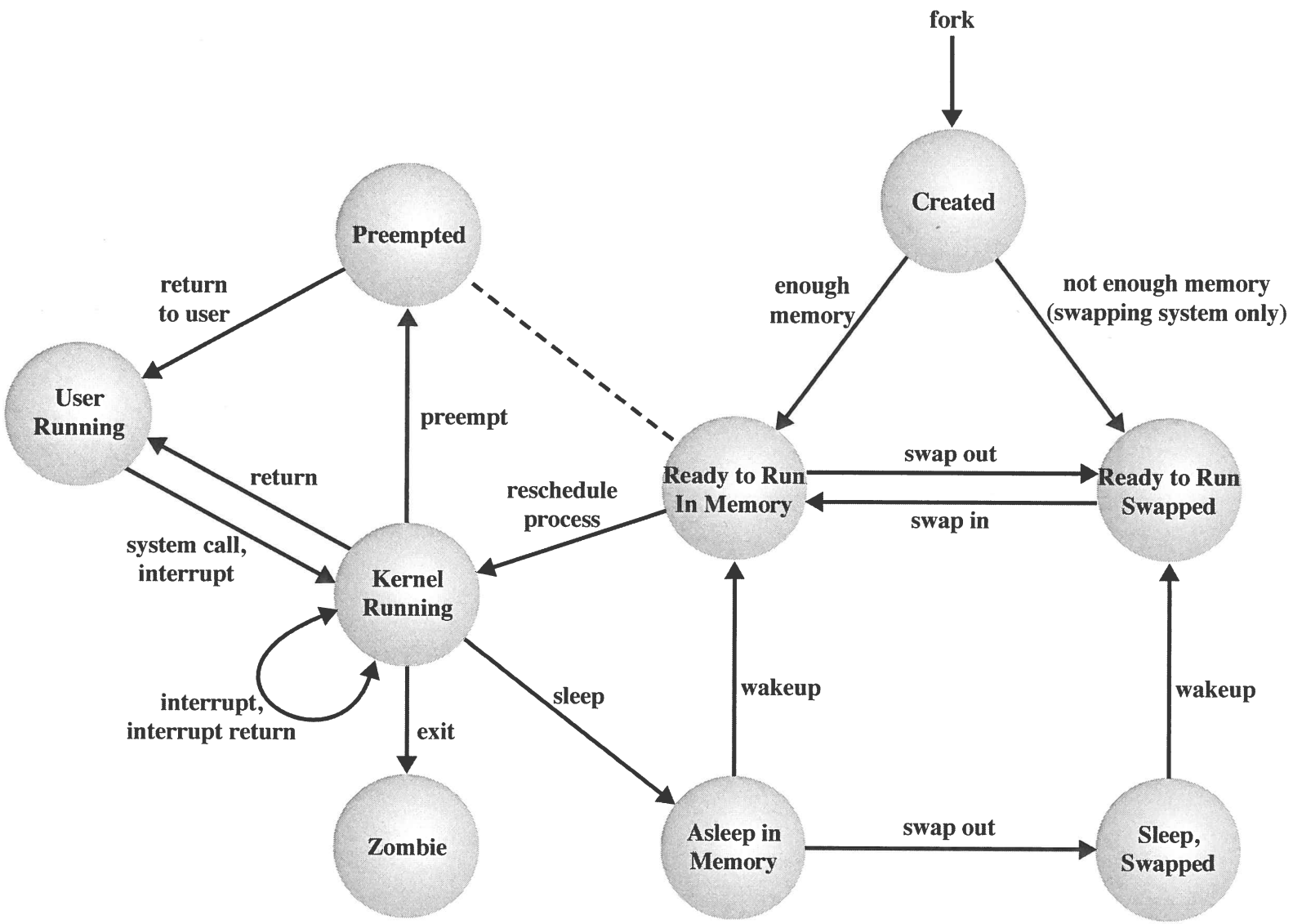


Figure 3.17 UNIX Process State Transition Diagram

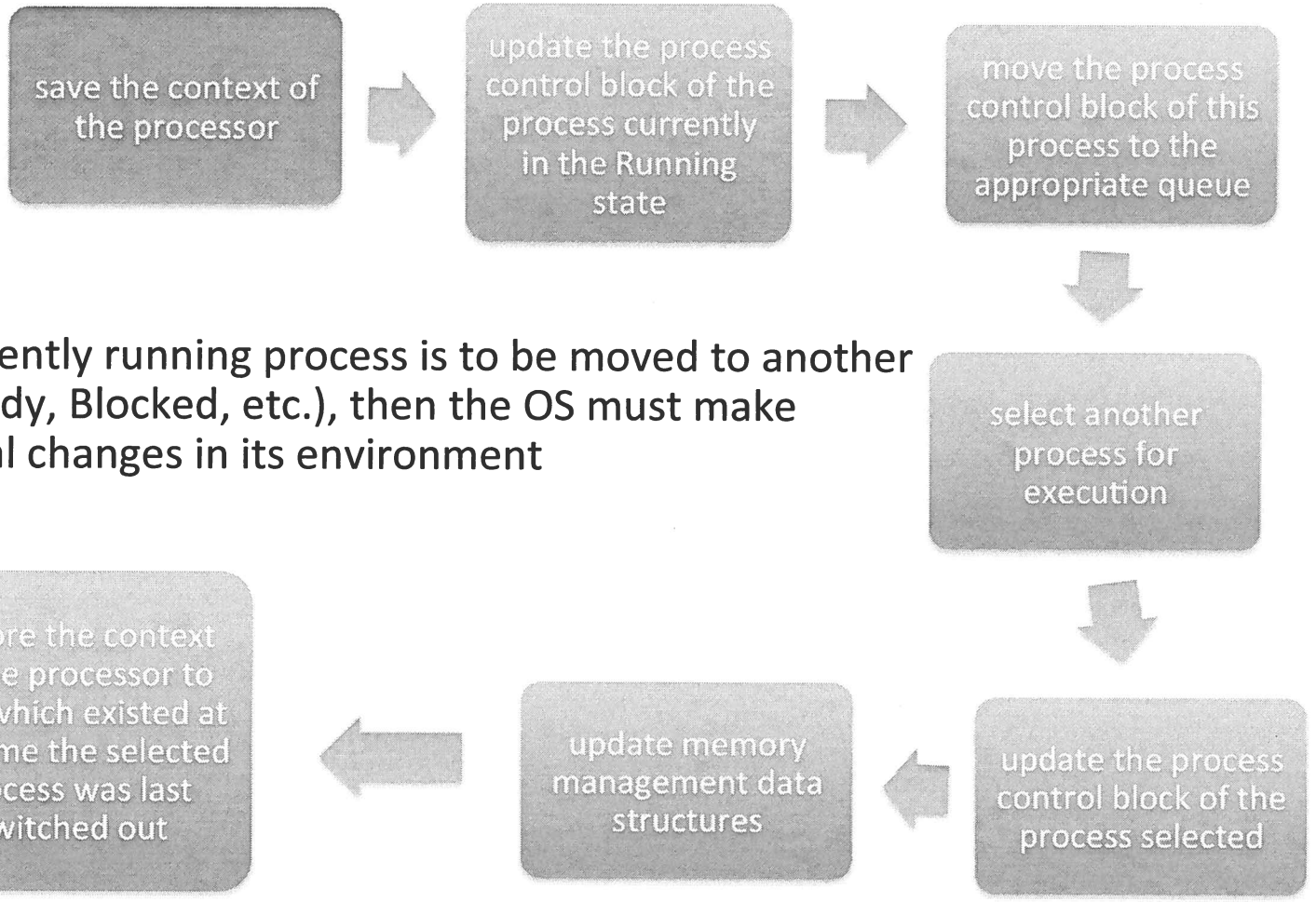
Table 3.10 UNIX Process Image

(Table is located on page 144 in the textbook)

User-Level Context	
Process text	Executable machine instructions of the program
Process data	Data accessible by the program of this process
User stack	Contains the arguments, local variables, and pointers for functions executing in user mode
Shared memory	Memory shared with other processes, used for interprocess communication
Register Context	
Program counter	Address of next instruction to be executed; may be in kernel or user memory space of this process
Processor status register	Contains the hardware status at the time of preemption; contents and format are hardware dependent
Stack pointer	Points to the top of the kernel or user stack, depending on the mode of operation at the time or preemption
General-purpose registers	Hardware dependent
System-Level Context	
Process table entry	Defines state of a process; this information is always accessible to the operating system
U (user) area	Process control information that needs to be accessed only in the context of the process
Per process region table	Defines the mapping from virtual to physical addresses; also contains a permission field that indicates the type of access allowed the process: read-only, read-write, or read-execute
Kernel stack	Contains the stack frame of kernel procedures as the process executes in kernel mode

Change of Process State

- The steps in a full process switch are:



If the currently running process is to be moved to another state (Ready, Blocked, etc.), then the OS must make substantial changes in its environment

Table 3.8

Mechanisms for Interrupting the Execution of a Process

Mechanism	Cause	Use
Interrupt	External to the execution of the current instruction	Reaction to an asynchronous external event
Trap	Associated with the execution of the current instruction	Handling of an error or an exception condition
Supervisor call	Explicit request	Call to an operating system function

System Interrupts

Interrupt

- Due to some sort of event that is external to and independent of the currently running process
  - clock interrupt
  - I/O interrupt
  - memory fault
- Time slice
  - the maximum amount of time that a process can execute before being interrupted

Trap

- An error or exception condition generated within the currently running process
- OS determines if the condition is fatal
  - moved to the Exit state and a process switch occurs
  - action will depend on the nature of the error