Processes and Resources

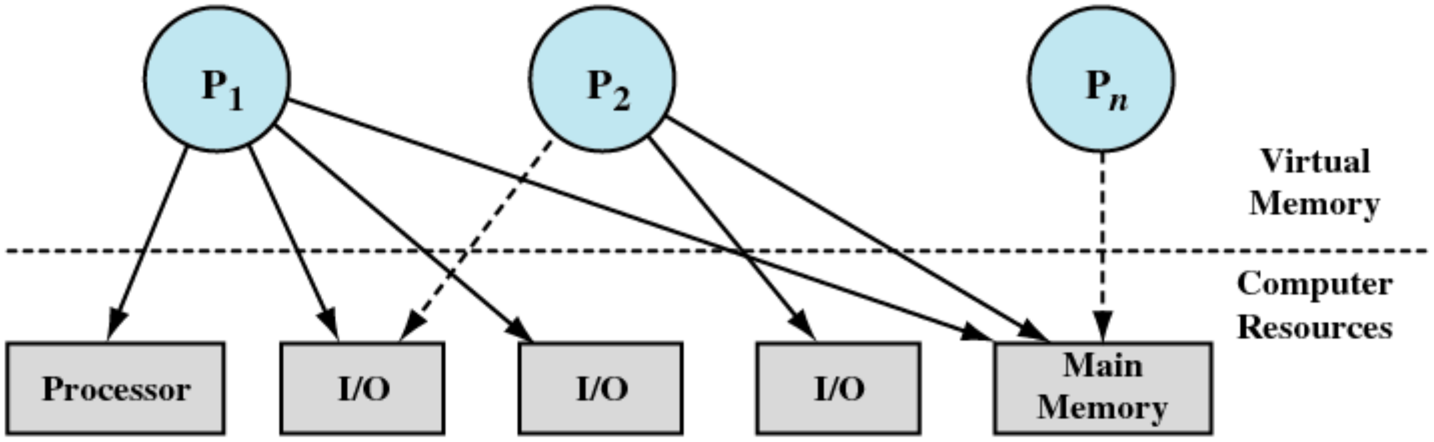


Figure 3.10 Processes and Resources (resource allocation at one snapshot in time)

P_2 is blocked, waiting for the I/O allocated to P_1
P_n has been swapped out (it is thus suspended)

Operating System Control Structures

- Information about the current status of each process and resource
- Tables are constructed for each entity the operating system manages

Memory Tables

- Keep track of
  - allocation of main memory to processes
  - allocation of secondary memory to processes
  - protection attributes for access to shared memory regions
  - information needed to manage virtual memory

I/O Tables

- Used by OS to manage I/O devices
  - I/O device is available or assigned
  - status of I/O operation
  - location in main memory being used as the source or destination of the I/O transfer

File Tables

- Keep track of
  - existence of files
  - location on secondary memory
  - current Status
  - attributes

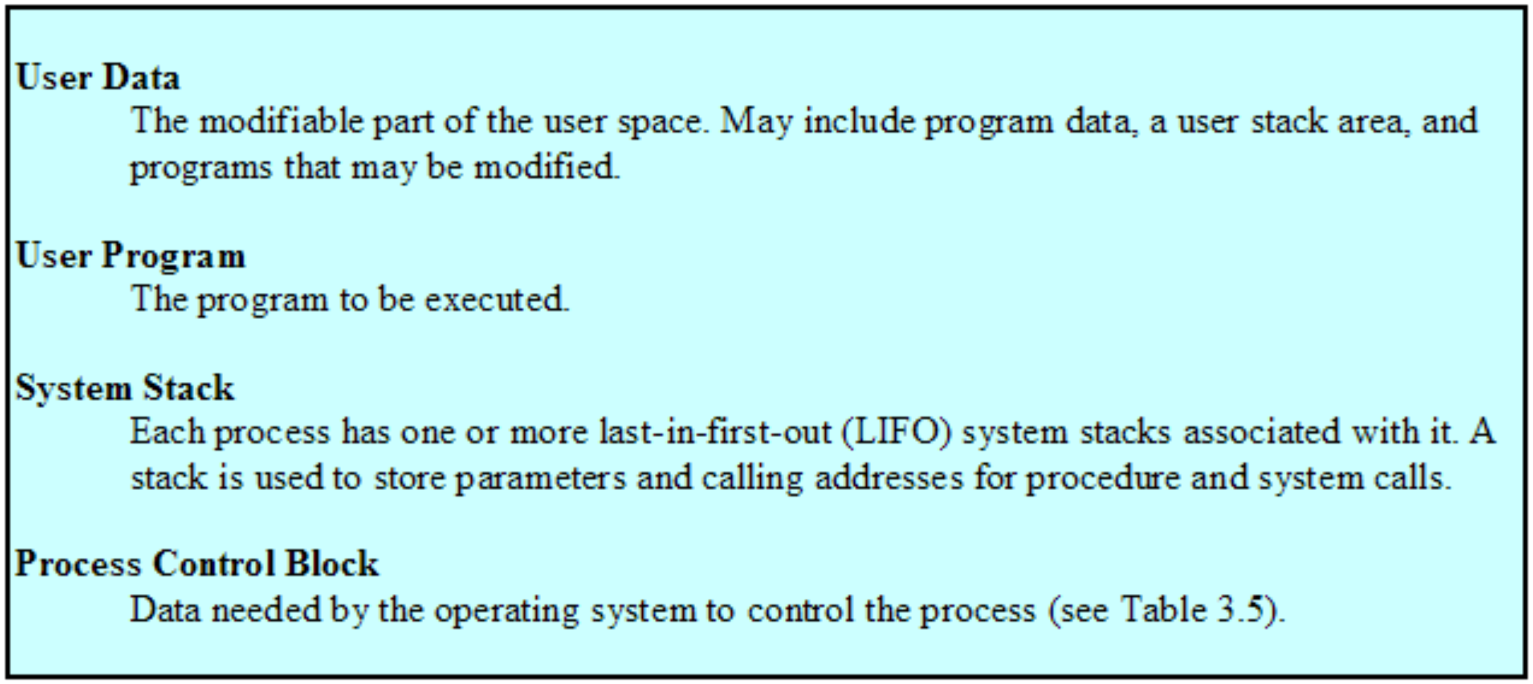
  - sometimes this information is maintained by a file management system

Process Table

- Mange processes
  - where process is located
  - attributes in the process control block
    - Program
    - Data
    - Stack

Process Image

Table 3.4 Typical Elements of a Process Image



User Data
The modifiable part of the user space. May include program data, a user stack area, and programs that may be modified.

User Program
The program to be executed.

System Stack
Each process has one or more last-in-first-out (LIFO) system stacks associated with it. A stack is used to store parameters and calling addresses for procedure and system calls.

Process Control Block
Data needed by the operating system to control the process (see Table 3.5).

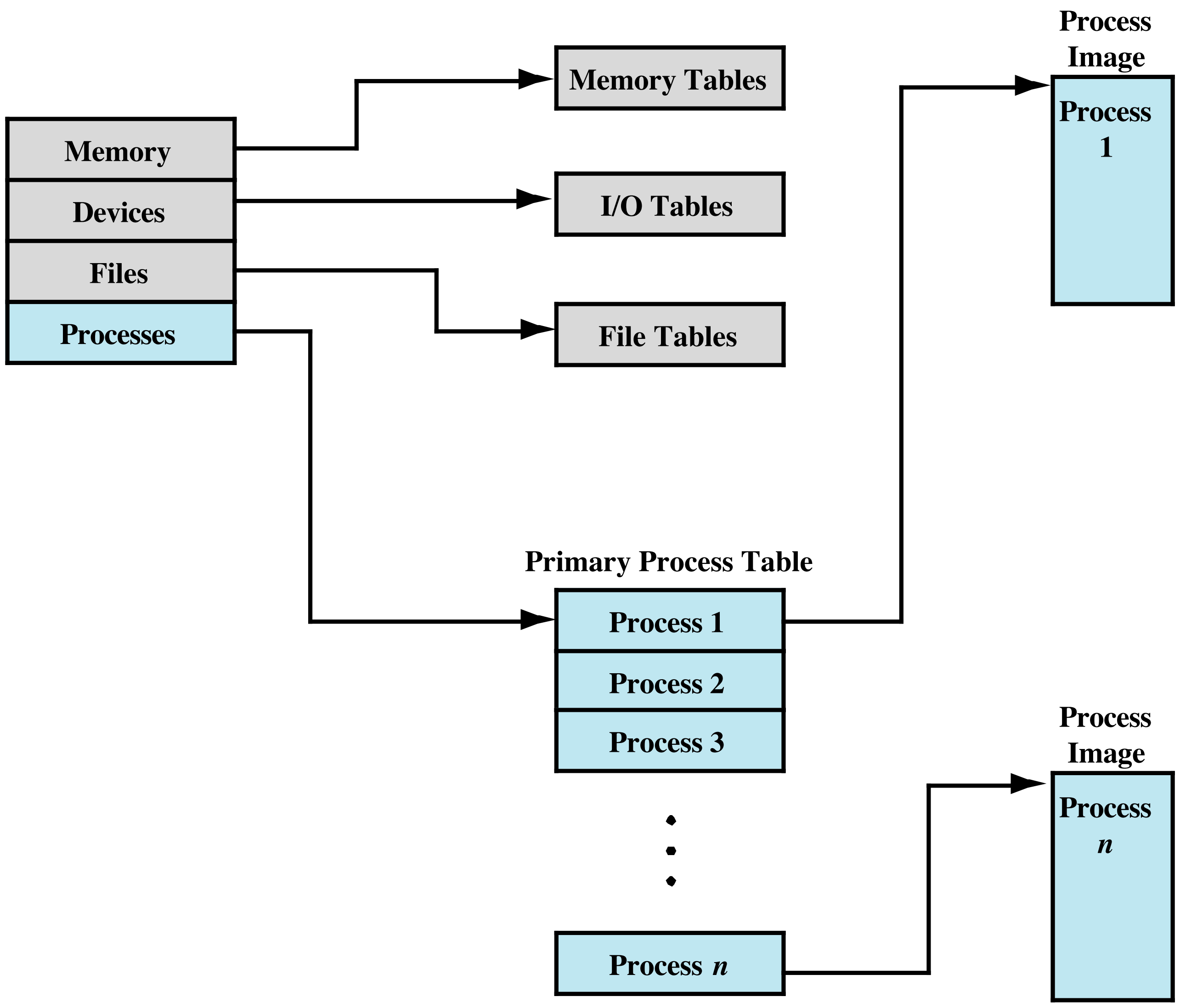


Figure 3.11 General Structure of Operating System Control Table

Process Control Block

- From Table 3.5
- Process identification
  - Identifiers
    - Numeric identifiers that may be stored with the process control block include
      - Identifier of this process
      - Identifier of the process that created this process (parent process)
      - User identifier

Process Control Block

- Processor State Information
  - User-Visible Registers
    - A user-visible register is one that may be referenced by means of the machine language that the processor executes while in user mode. Typically, there are from 8 to 32 of these registers, although some RISC implementations have over 100.

Process Control Block

- Processor State Information
  - Control and Status Registers

    These are a variety of processor registers that are employed to control the operation of the processor. These include

    - *Program counter:* Contains the address of the next instruction to be fetched
    - *Condition codes:* Result of the most recent arithmetic or logical operation (e.g., sign, zero, carry, equal, overflow)
    - *Status information:* Includes interrupt enabled/disabled flags, execution mode

Process Control Block

- Processor State Information
  - Stack Pointers
    - Each process has one or more last-in-first-out (LIFO) system stacks associated with it. A stack is used to store parameters and calling addresses for procedure and system calls. The stack pointer points to the top of the stack.

Process Control Block

- Process Control Information
  - Scheduling and State Information

    This is information needed by the OS to perform its scheduling function

    - *Process state:* defines the readiness of the process to be scheduled for execution (e.g., running, ready, waiting, halted).
    - *Priority:* One or more fields may be used to describe the scheduling priority of the process. In some systems, several values are required (e.g., default, current, highest-allowable)
    - *Scheduling-related information:* This will depend on the scheduling algorithm used. Examples are the amount of time that the process has been waiting and the amount of time that the process executed the last time it was running.
    - *Event:* Identity of event the process is awaiting before it can be resumed

Process Control Block

- Process Control Information
  - Data Structuring
    - A process may be linked to other process in a queue, ring, or some other structure.
      - E.g., all processes in a waiting state for a particular priority level may be linked in a queue. A process may exhibit a parent-child relationship with another process. The process control block may contain pointers to other processes to support these structures.

Process Control Block

- Process Control Information
  - Interprocess Communication
    - Various flags, signals, and messages may be associated with communication between two independent processes. Some or all of this information may be maintained in the process control block.
  - Process Privileges
    - Processes are granted privileges in terms of the memory that may be accessed and the types of instructions that may be executed. In addition, privileges may apply to the use of system utilities and services.

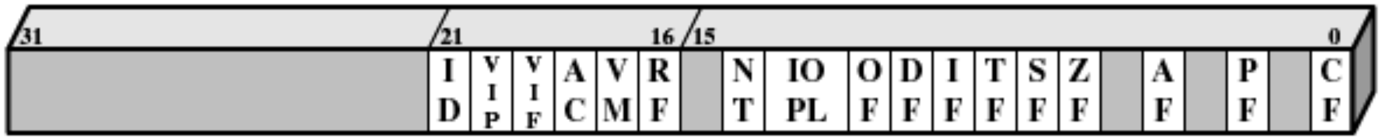
Process Control Block

- Process Control Information
  - Memory Management
    - This section may include pointers to segment and/or page tables that describe the virtual memory assigned to this process.
  - Resource Ownership and Utilization
    - Resources controlled by the process may be indicated, such as opened files. A history of utilization of the processor or other resources may also be included; this information may be needed by the scheduler.

Processor State Information

- Consists of contents of processor registers
  - User-visible registers
  - Control and status registers
  - Stack pointers
- Program status word (PSW)
  - contains status information
  - E.g. consider the EFLAGS register on Pentium machines

Pentium II EFLAGS Register

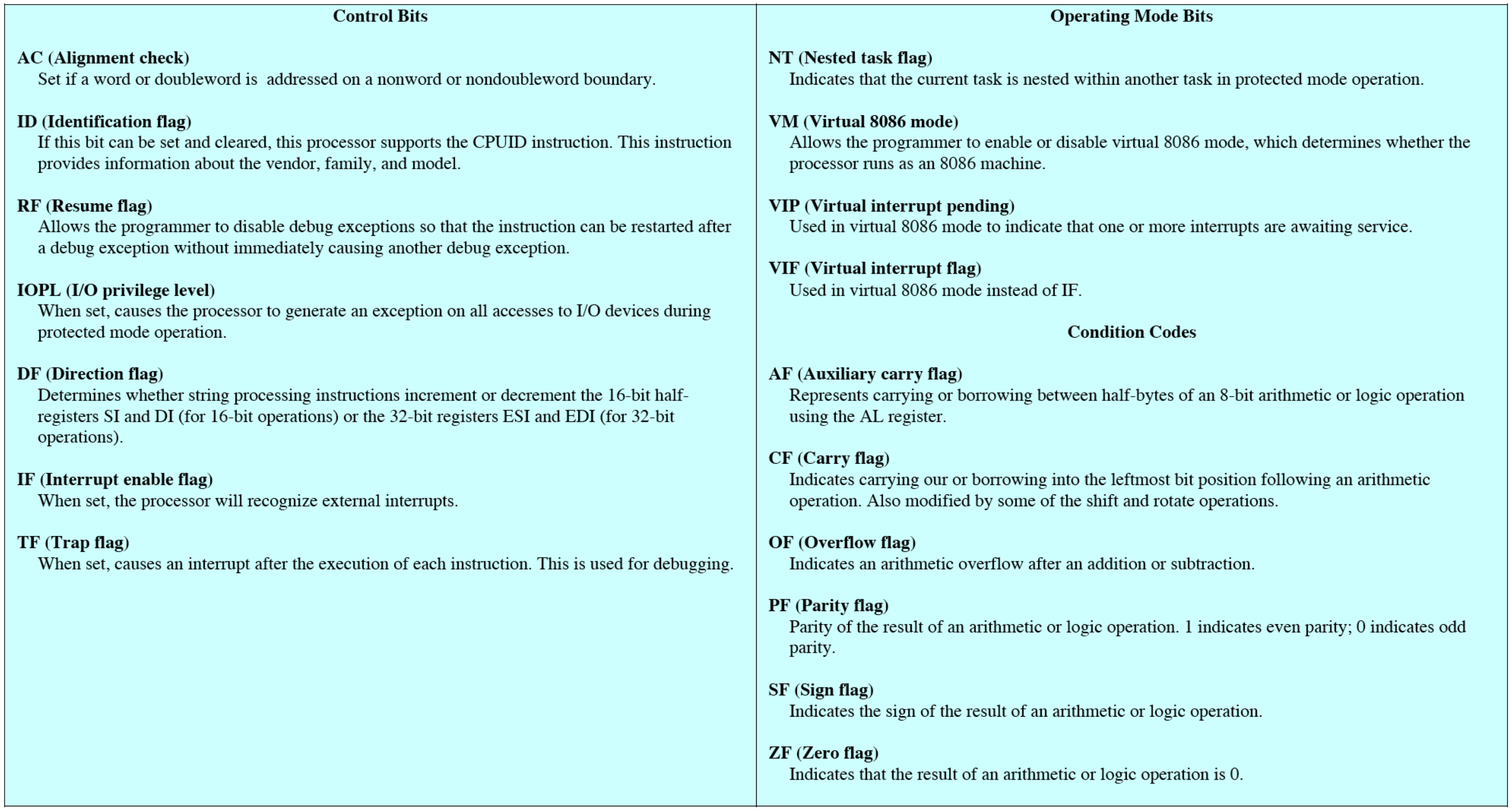


ID	=	Identification flag
VIP	=	Virtual interrupt pending
VIF	=	Virtual interrupt flag
AC	=	Alignment check
VM	=	Virtual 8086 mode
RF	=	Resume flag
NT	=	Nested task flag
IOPL	=	I/O privilege level
OF	=	Overflow flag
DF	=	Direction flag
IF	=	Interrupt enable flag
TF	=	Trap flag
SF	=	Sign flag
ZF	=	Zero flag
AF	=	Auxiliary carry flag
PF	=	Parity flag
CF	=	Carry flag

Figure 3.12 Pentium II EFLAGS Register

Pentium II EFLAGS Register

Table 3.6 Pentium EFLAGS Register Bits



Control Bits	Operating Mode Bits
AC (Alignment check) Set if a word or doubleword is addressed on a nonword or nondoubleword boundary.	**NT (Nested task flag)** Indicates that the current task is nested within another task in protected mode operation.
ID (Identification flag) If this bit can be set and cleared, this processor supports the CPUID instruction. This instruction provides information about the vendor, family, and model.	**VM (Virtual 8086 mode)** Allows the programmer to enable or disable virtual 8086 mode, which determines whether the processor runs as an 8086 machine.
RF (Resume flag) Allows the programmer to disable debug exceptions so that the instruction can be restarted after a debug exception without immediately causing another debug exception.	**VIP (Virtual interrupt pending)** Used in virtual 8086 mode to indicate that one or more interrupts are awaiting service.
IOPL (I/O privilege level) When set, causes the processor to generate an exception on all accesses to I/O devices during protected mode operation.	**VIF (Virtual interrupt flag)** Used in virtual 8086 mode instead of IF.
DF (Direction flag) Determines whether string processing instructions increment or decrement the 16-bit half-registers SI and DI (for 16-bit operations) or the 32-bit registers ESI and EDI (for 32-bit operations).	**Condition Codes**
IF (Interrupt enable flag) When set, the processor will recognize external interrupts.	**AF (Auxiliary carry flag)** Represents carrying or borrowing between half-bytes of an 8-bit arithmetic or logic operation using the AL register.
TF (Trap flag) When set, causes an interrupt after the execution of each instruction. This is used for debugging.	**CF (Carry flag)** Indicates carrying our or borrowing into the leftmost bit position following an arithmetic operation. Also modified by some of the shift and rotate operations.
	OF (Overflow flag) Indicates an arithmetic overflow after an addition or subtraction.
	PF (Parity flag) Parity of the result of an arithmetic or logic operation. 1 indicates even parity; 0 indicates odd parity.
	SF (Sign flag) Indicates the sign of the result of an arithmetic or logic operation.
	ZF (Zero flag) Indicates that the result of an arithmetic or logic operation is 0.

Modes of Execution

- User mode
  - Less-privileged mode
  - User programs typically execute in this mode
- System mode, control mode, or kernel mode
  - More-privileged mode
  - Kernel of the operating system

Process Creation

- Process creation steps:
  - Assign a unique process identifier
  - Allocate space for the process
  - Initialize process control block
  - Set up appropriate linkages
    - e.g. add new process to linked list used for scheduling queue
  - Create or expand other data structures
    - e.g. maintain an accounting file

When to Switch a Process

- Clock interrupt
  - process has executed for the maximum allowable time slice
- I/O interrupt
- Memory fault
  - memory address is in virtual memory so it must be brought into main memory

When to Switch a Process

- Trap
  - error or exception occurred
  - may cause process to be moved to Exit state
- Supervisor call
  - such as file open
    - e.g. user process calls OS function to open file

Change of Process State

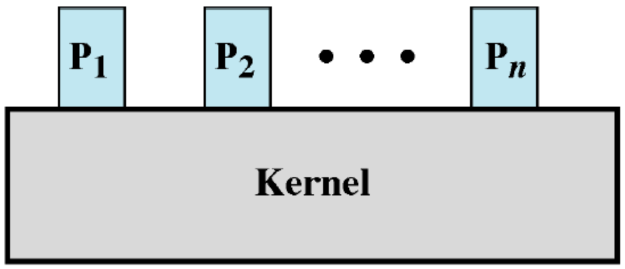
- Outgoing process
  - save context of processor including program counter and other registers
  - update the process control block of the process that is currently in the Running state
  - move process control block to appropriate queue – ready; blocked; ready/suspend
- Select another process for execution

Change of Process State

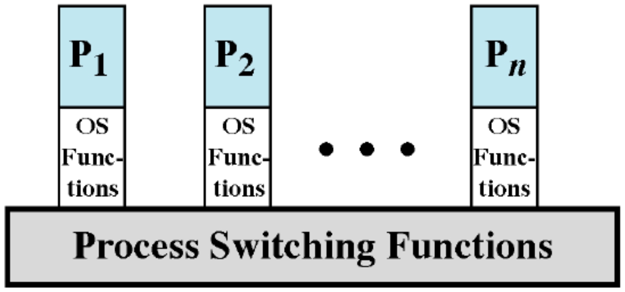
- Incoming process
  - update the process control block of the process selected
  - update memory-management data structures
  - restore context of the selected process

Execution of the OS

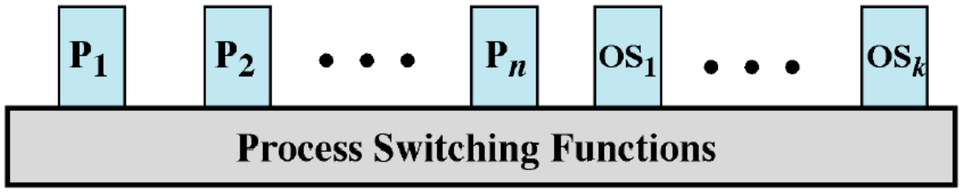
- Non-process Kernel
  - execute kernel outside of any process
  - OS code is executed as a separate entity that operates in privileged mode
- Execution Within User Processes
  - OS software within context of a user process
  - Process executes in privileged mode when executing OS code



(a) Separate kernel

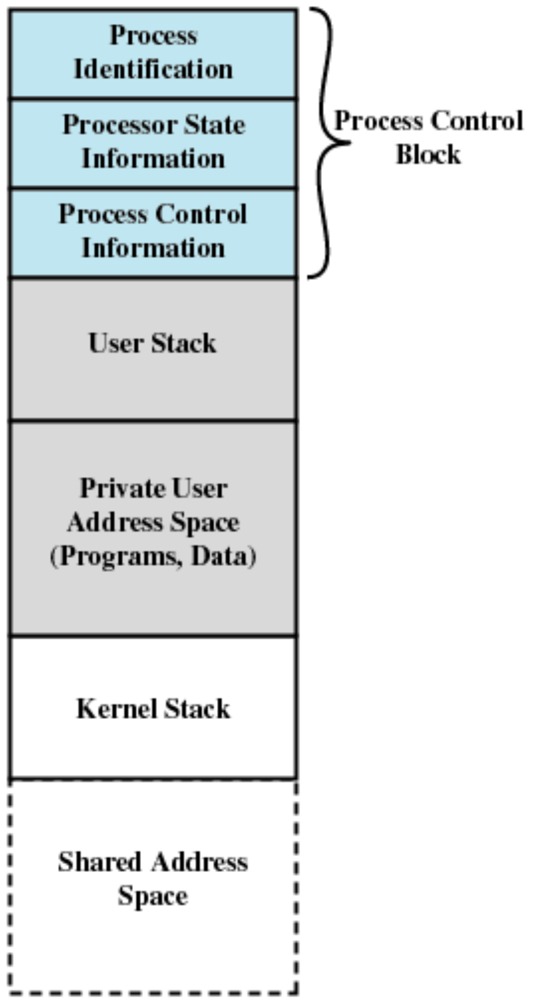


(b) OS functions execute within user processes



(c) OS functions execute as separate processes

Relationship between OS and User Process


Process Identification
Processor State Information
Process Control Information
Process Control Block
User Stack
Private User Address Space (Programs, Data)
Kernel Stack
Shared Address Space

part (b) of previous figure

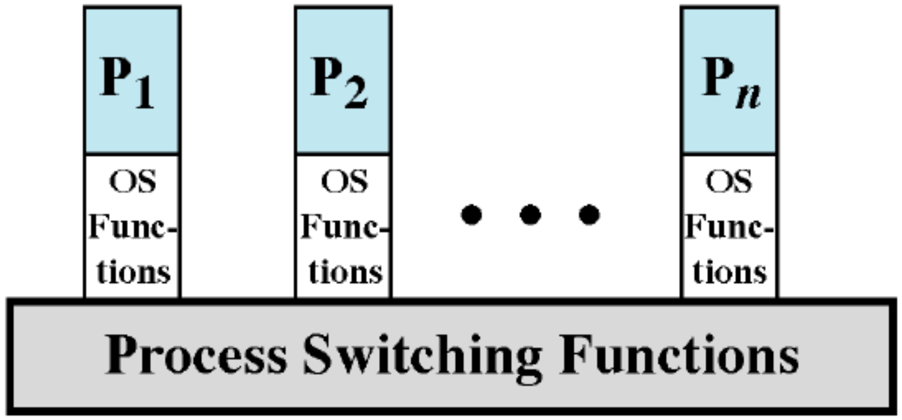
Figure 3.16 Process Image: Operating System Executes Within User Space

Execution of the Operating System

- Process-Based Operating System
  - part (c) of previous figure
  - implement OS as a collection of system processes
  - useful in multi-processor or multi-computer environment

UNIX SVR4 Process Management

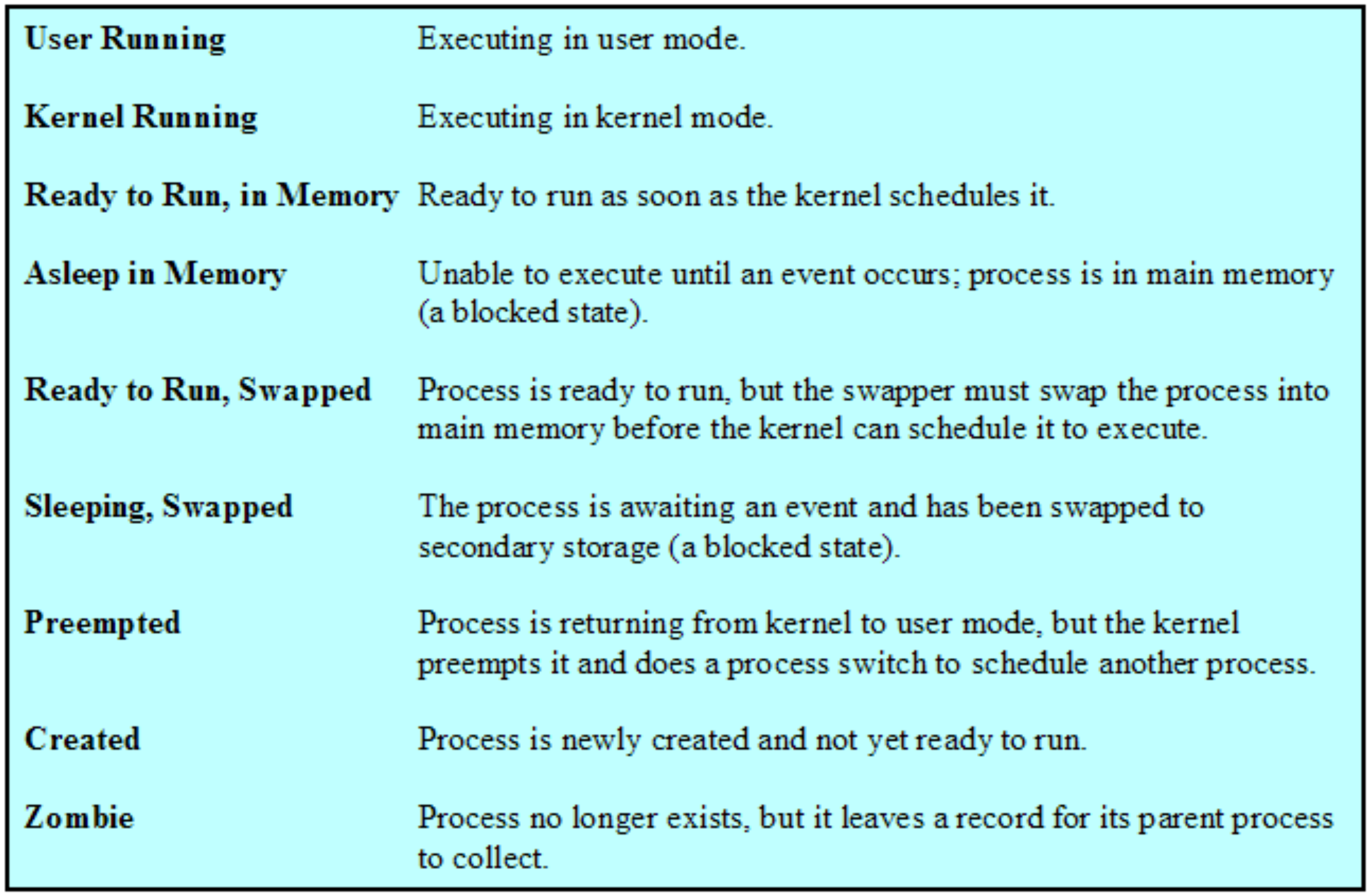
- Most of the operating system executes within the environment of a user process



(b) OS functions execute within user processes

UNIX Process States

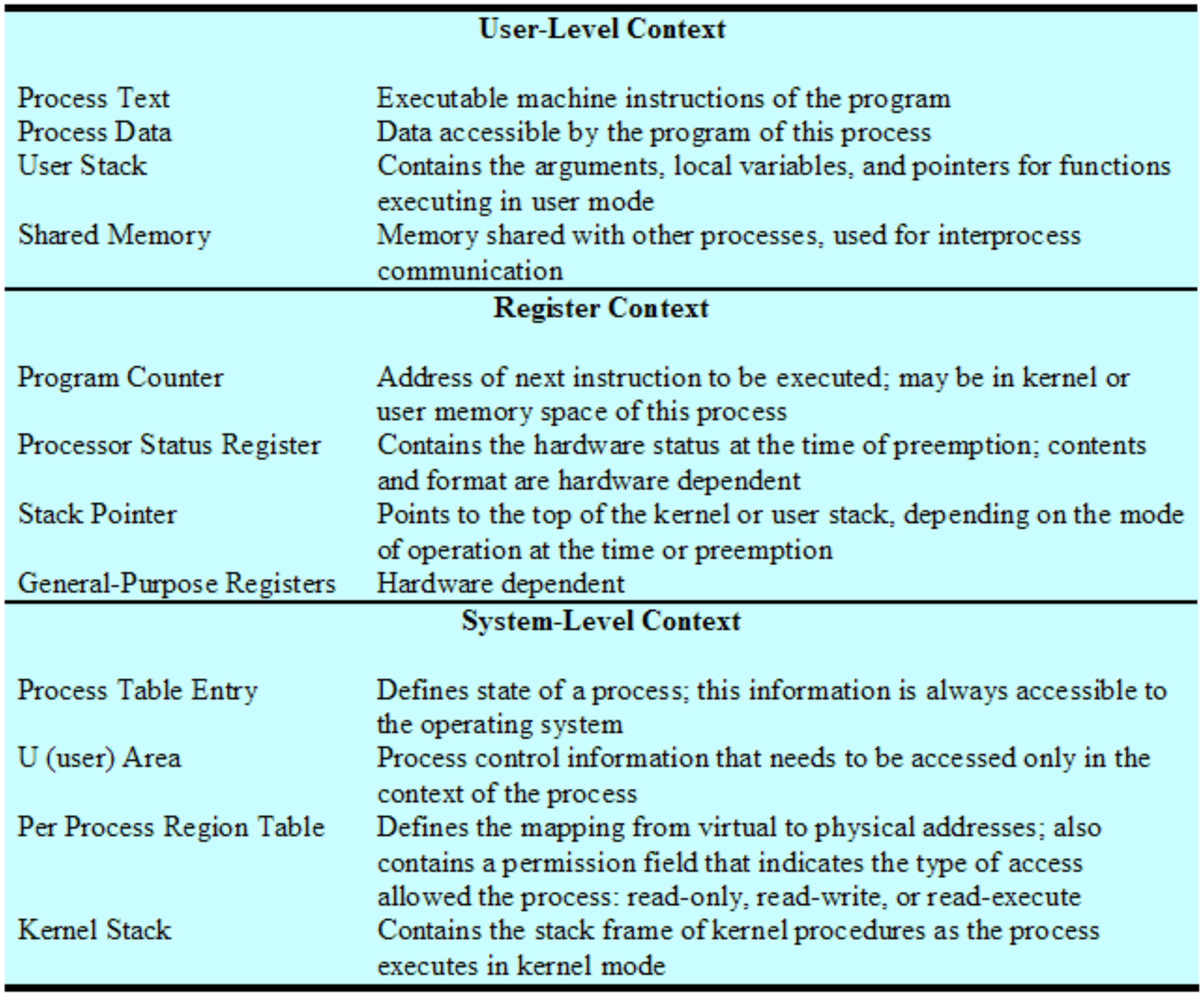
Table 3.9 UNIX Process States



User Running	Executing in user mode.
Kernel Running	Executing in kernel mode.
Ready to Run, in Memory	Ready to run as soon as the kernel schedules it.
Asleep in Memory	Unable to execute until an event occurs; process is in main memory (a blocked state).
Ready to Run, Swapped	Process is ready to run, but the swapper must swap the process into main memory before the kernel can schedule it to execute.
Sleeping, Swapped	The process is awaiting an event and has been swapped to secondary storage (a blocked state).
Preempted	Process is returning from kernel to user mode, but the kernel preempts it and does a process switch to schedule another process.
Created	Process is newly created and not yet ready to run.
Zombie	Process no longer exists, but it leaves a record for its parent process to collect.

UNIX Process Image

Table 3.10 UNIX Process Image



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

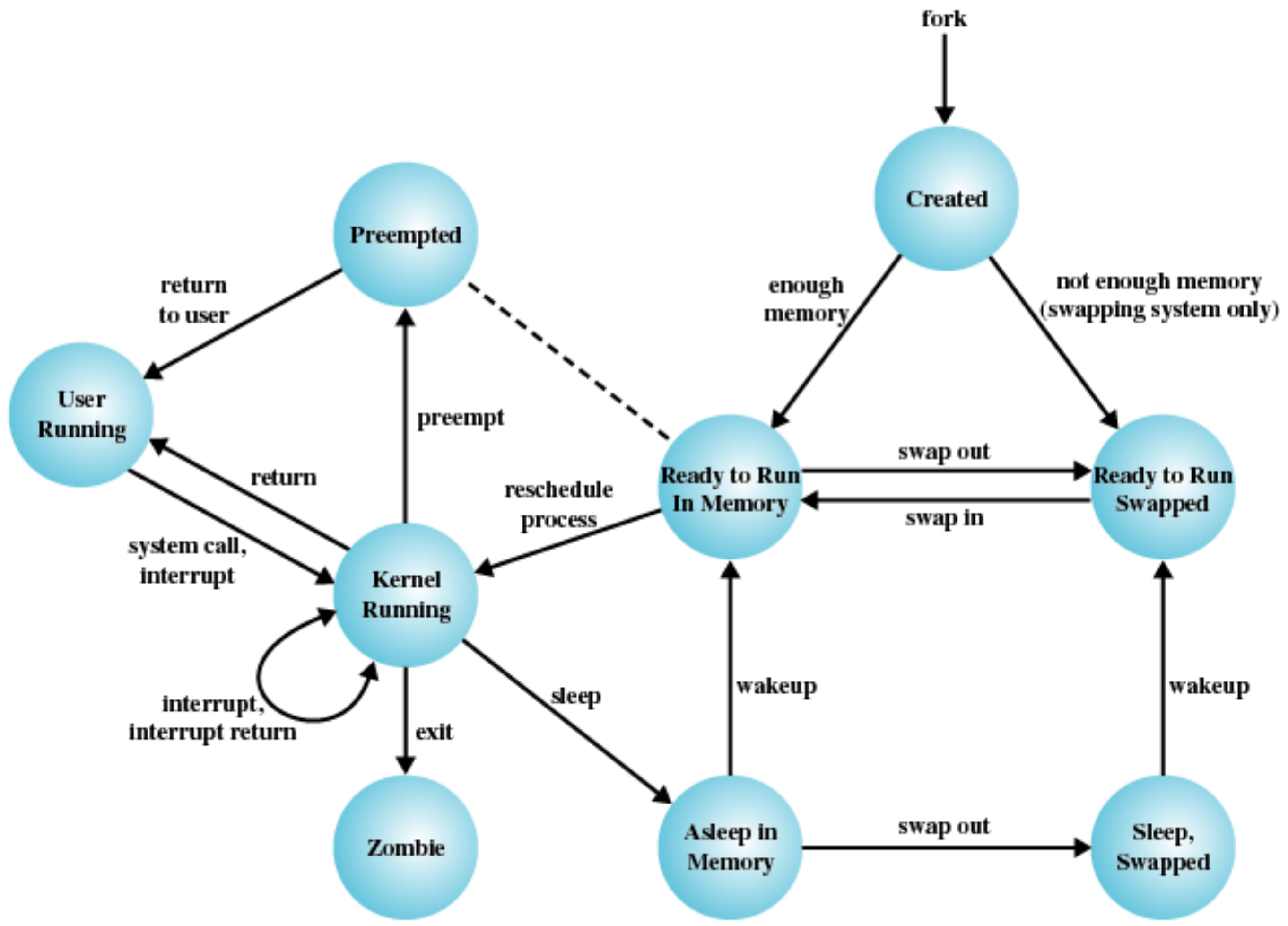


Figure 3.17 UNIX Process State Transition Diagram