Chapter 3: Processes

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Process Concept

- **Process** – a program in execution; process execution must progress in sequential fashion
- Multiple parts
    - The program code, also called **text section**
    - Current activity including **program counter**, processor registers

Assignment Project Exam Help

    - **Stack** containing temporary data

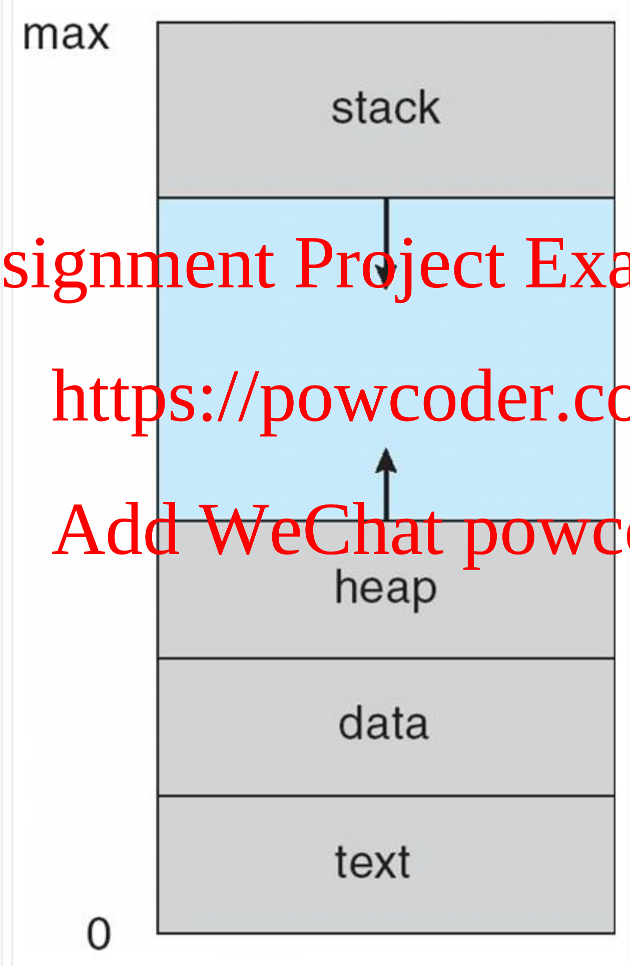
https://powcoder.com

        - Function parameters, return addresses, local variables
    - **Data section** containing global variables

Add WeChat powcoder

    - **Heap** containing memory dynamically allocated during run time
- Program is ***passive*** entity stored on disk (**executable file**), process is ***active***
    - Program becomes process when executable file loaded into memory
- One program can be several processes
    - Consider multiple users executing the same program

Process in Memory



Assignment Project Exam Help

https://powcoder.com

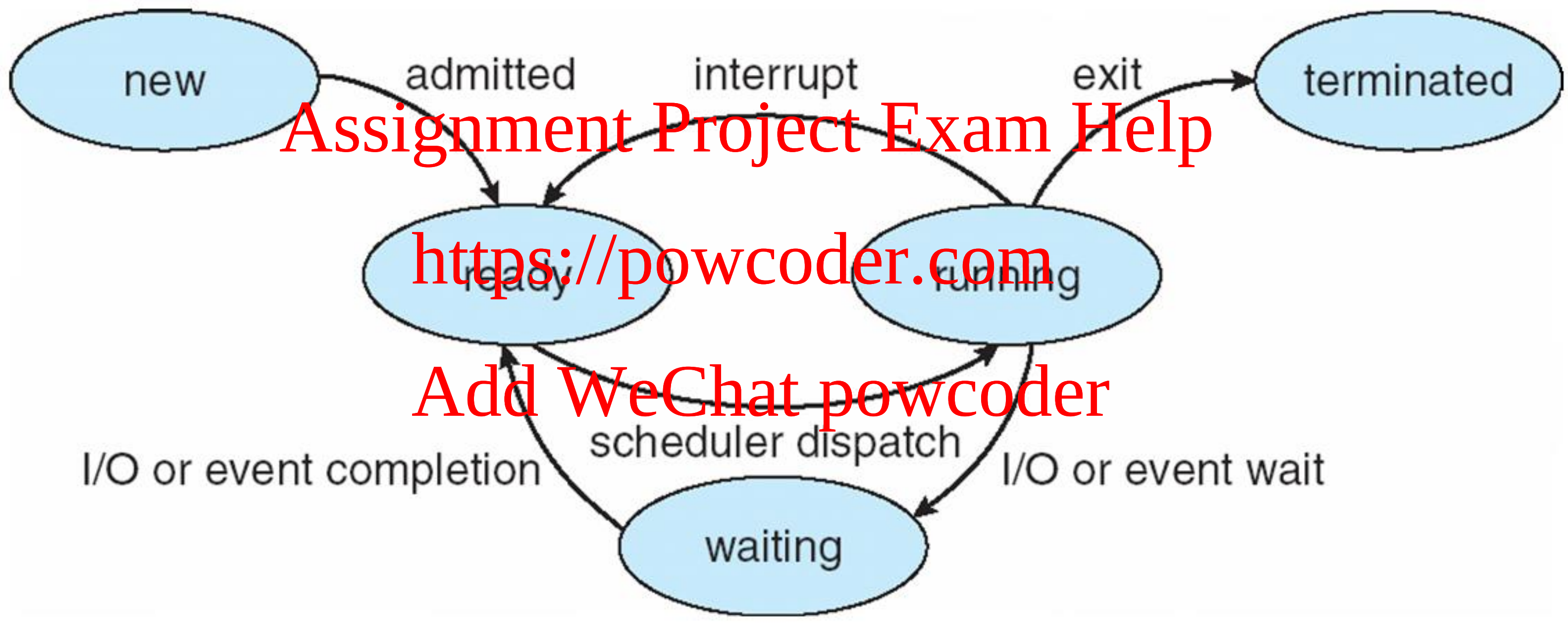
Add WeChat powcoder

Process State

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

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Diagram of Process State

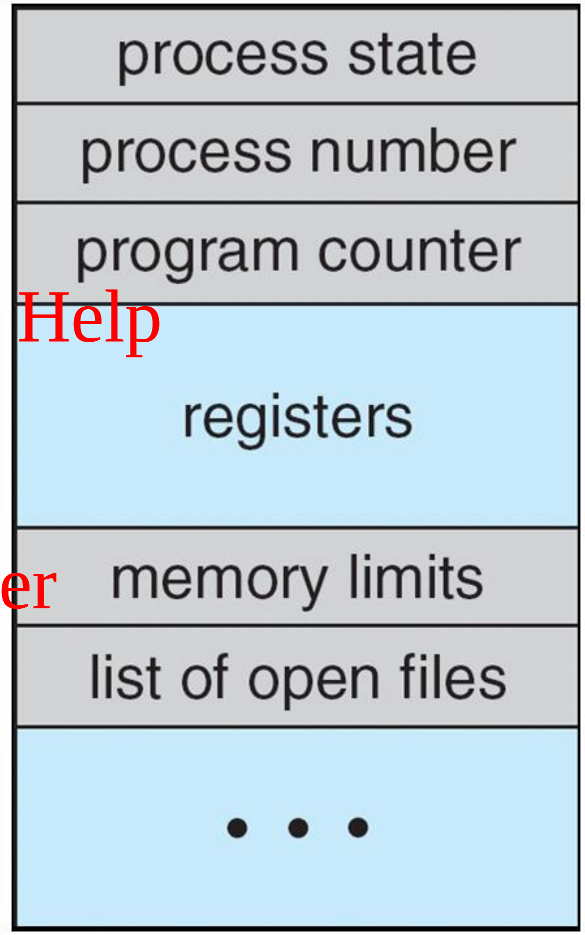


Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Process Control Block (PCB)

Information associated with each process (also called **task control block**)

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



process state

process number

program counter

registers

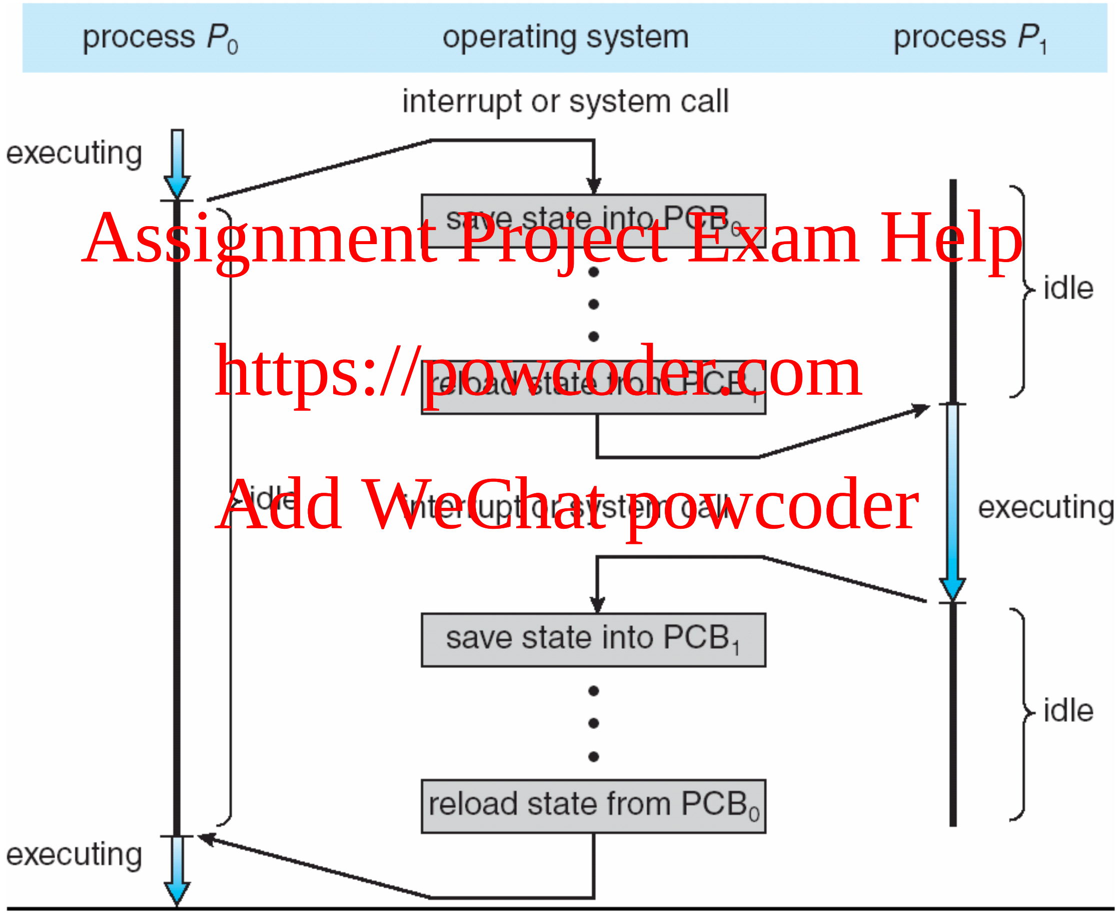
memory limits

list of open files

• • •

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

CPU Switch From Process to Process



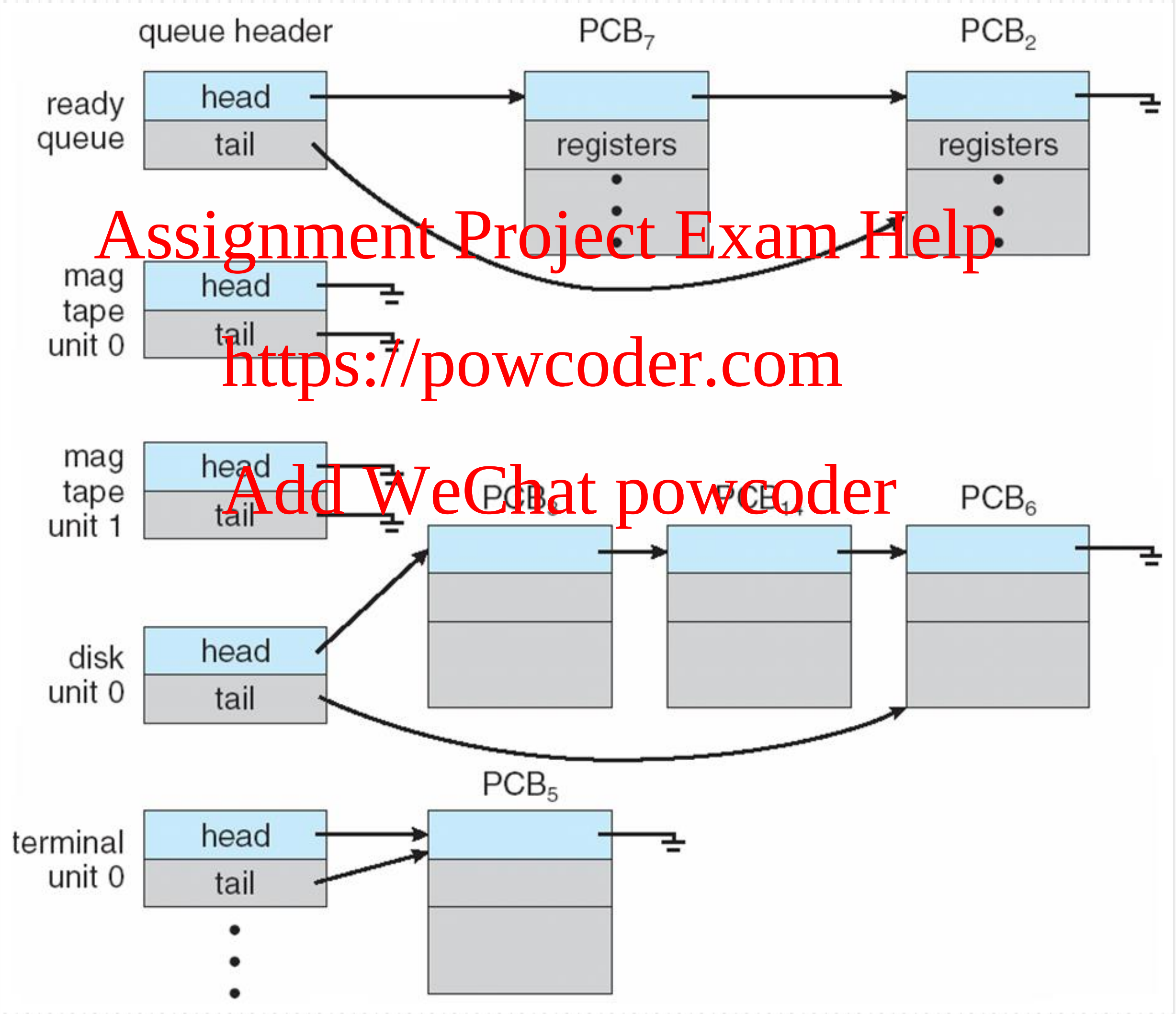
Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Process Scheduling

- Maximize CPU use, quickly switch processes onto CPU for time sharing
- **Process scheduler** selects among available processes for next execution on CPU
- Maintains **scheduling queues** of processes
  - **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

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Ready Queue And Various I/O Device Queues



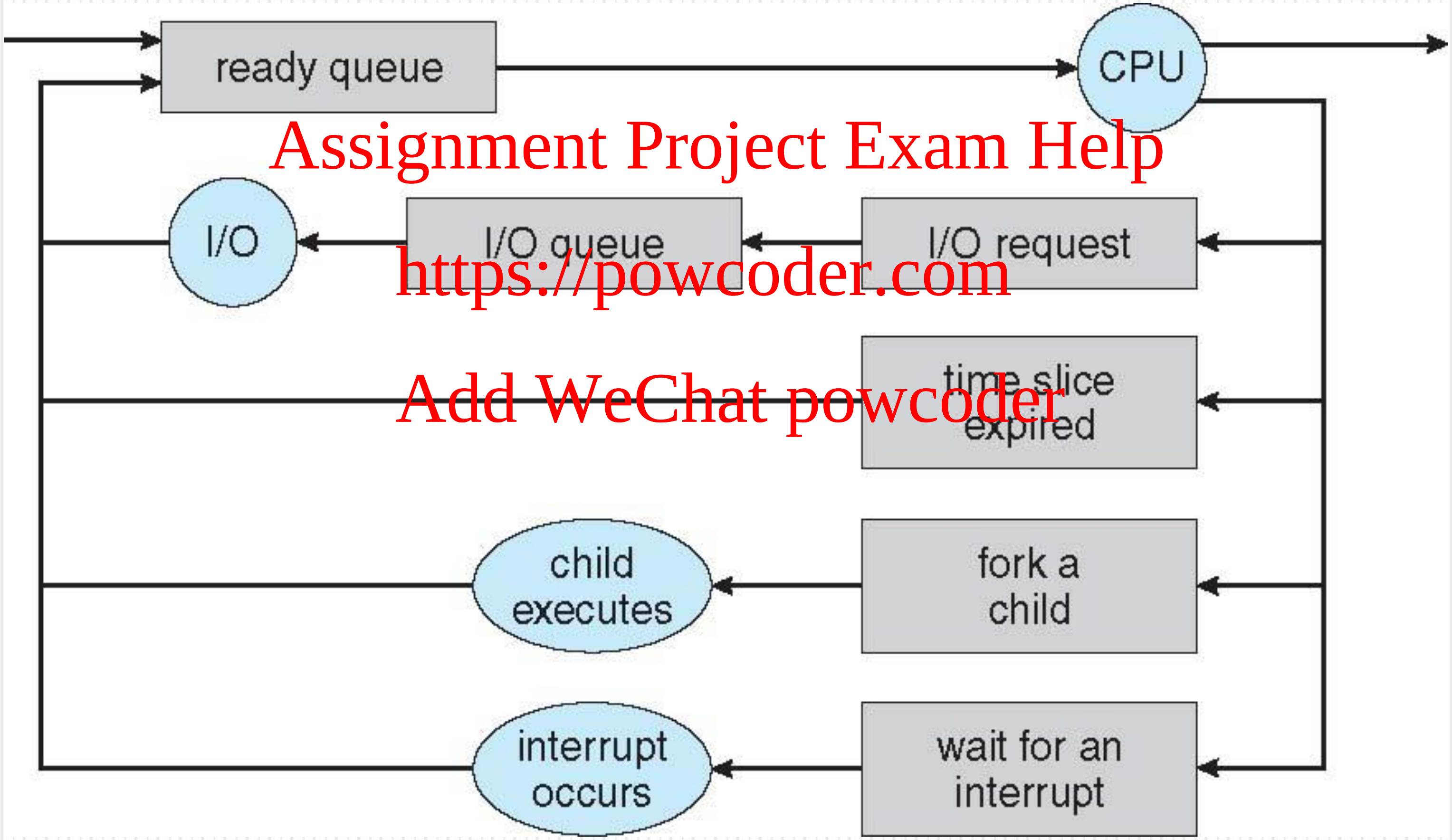
Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Representation of Process Scheduling

- **Queueing diagram** represents queues, resources, flows



Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Schedulers

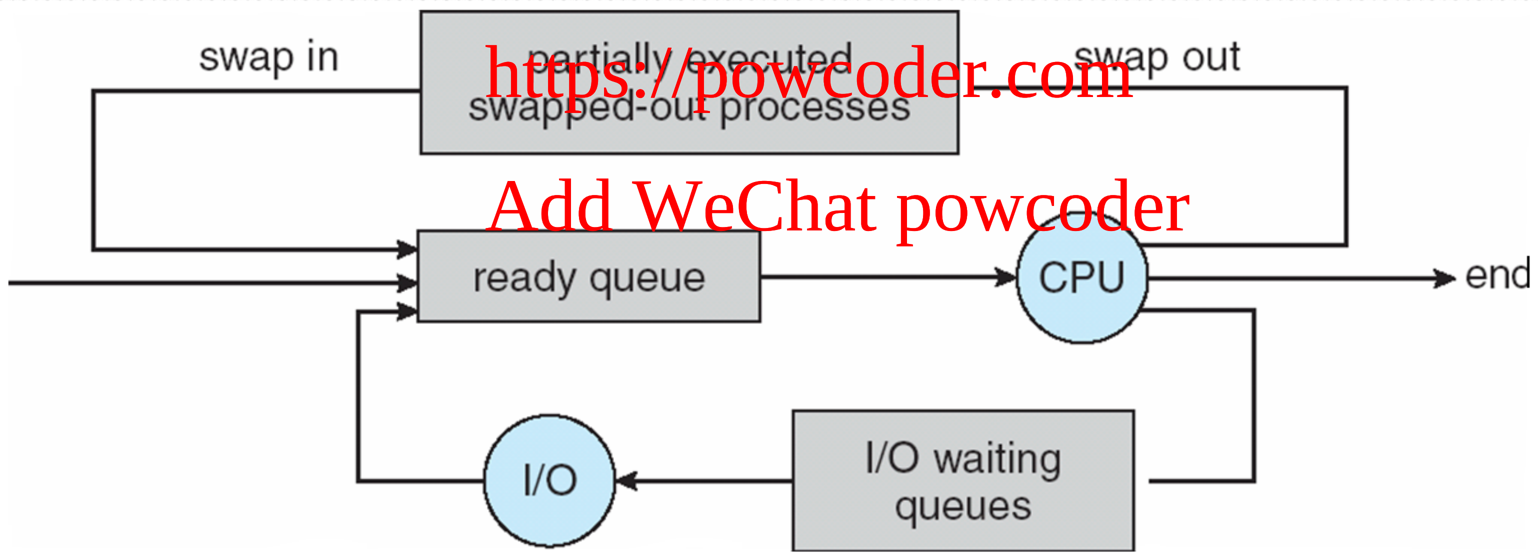
- **Long-term scheduler** (or **job scheduler**) – selects which processes should be brought into the ready queue - The long-term scheduler controls the **degree of multiprogramming -** Long-term scheduler is invoked very infrequently (seconds, minutes) ⇒ (may be slow)
- **Short-term scheduler** (or **CPU scheduler**) – selects which process should be executed next and allocates CPU - Sometimes the only scheduler in a system - Short-term scheduler is invoked very frequently (milliseconds) ⇒ (must be fast)
- Processes can be described as either:
  - **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***

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Addition of Medium Term Scheduling

- **Medium-term scheduler** can be added if degree of multiple programming needs to decrease
  - Remove process from memory, store on disk, bring back in from disk to continue execution: **swapping**

Assignment Project Exam Help



https://powcoder.com

Add WeChat powcoder

Context Switch

- When CPU switches to another process, the system must **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

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Context Switch

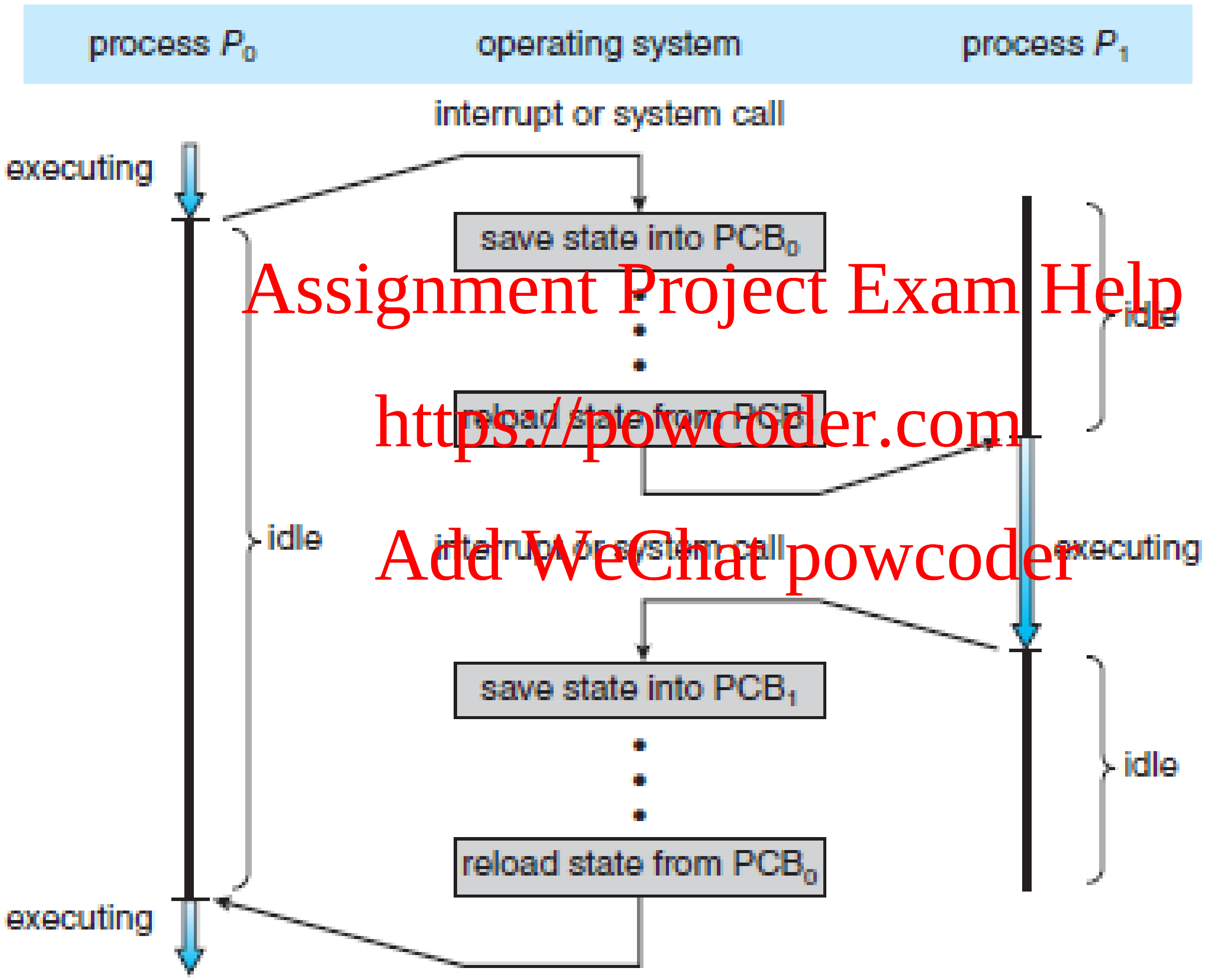


Figure 3.6 Diagram showing context switch from process to process.

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

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**)

Assignment Project Exam Help

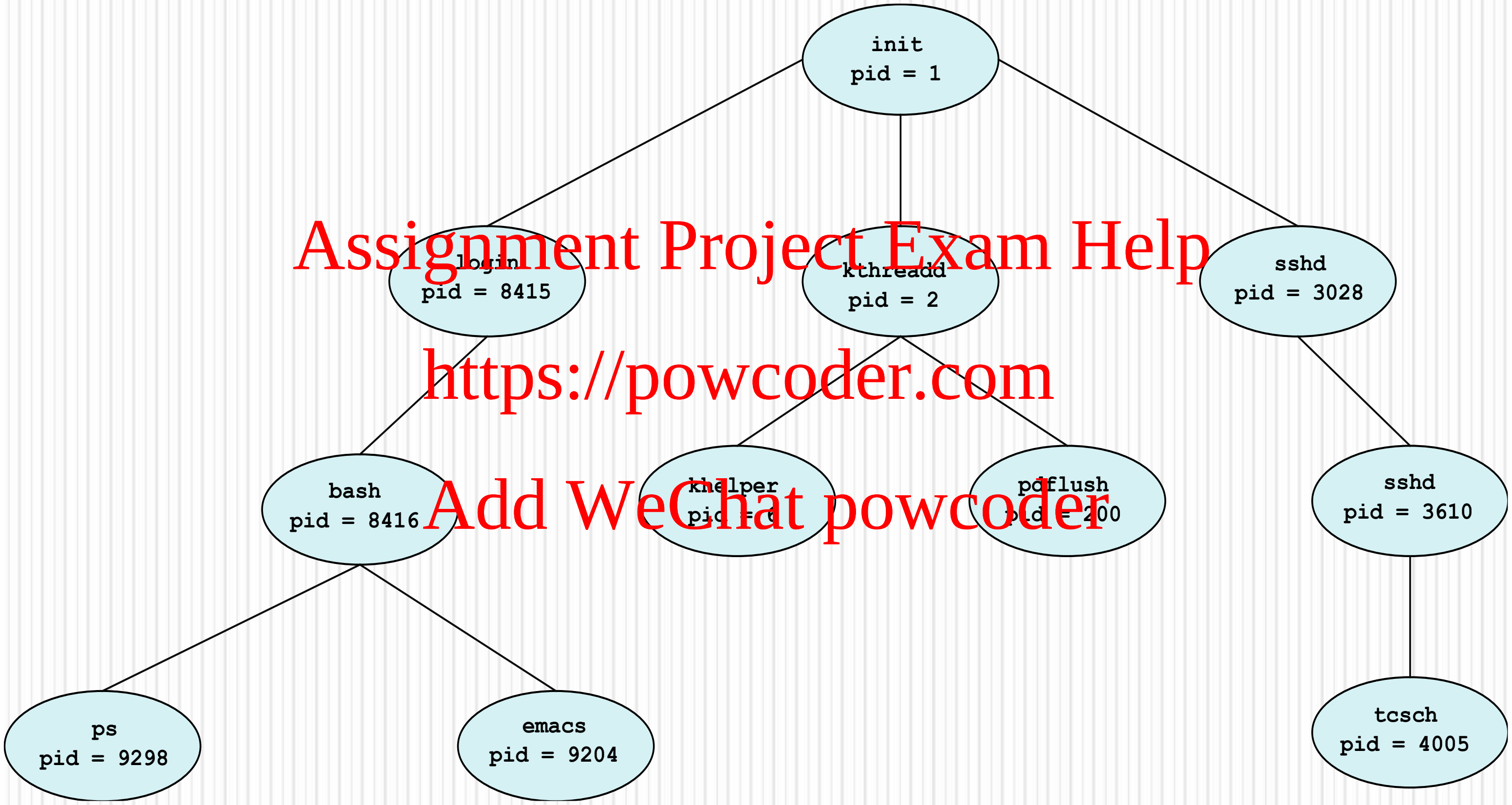
https://powcoder.com

Add WeChat powcoder

- 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

A Tree of Processes in Linux



Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Process Creation (Cont.)

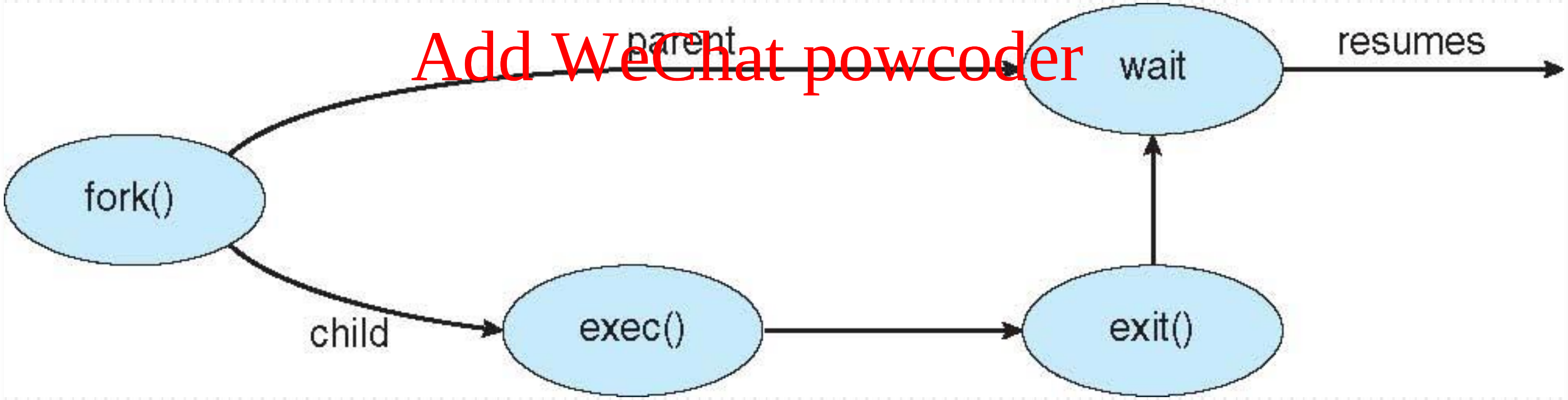
- Address space
  - Child duplicate of parent
  - Child has a program loaded into it

Assignment Project Exam Help

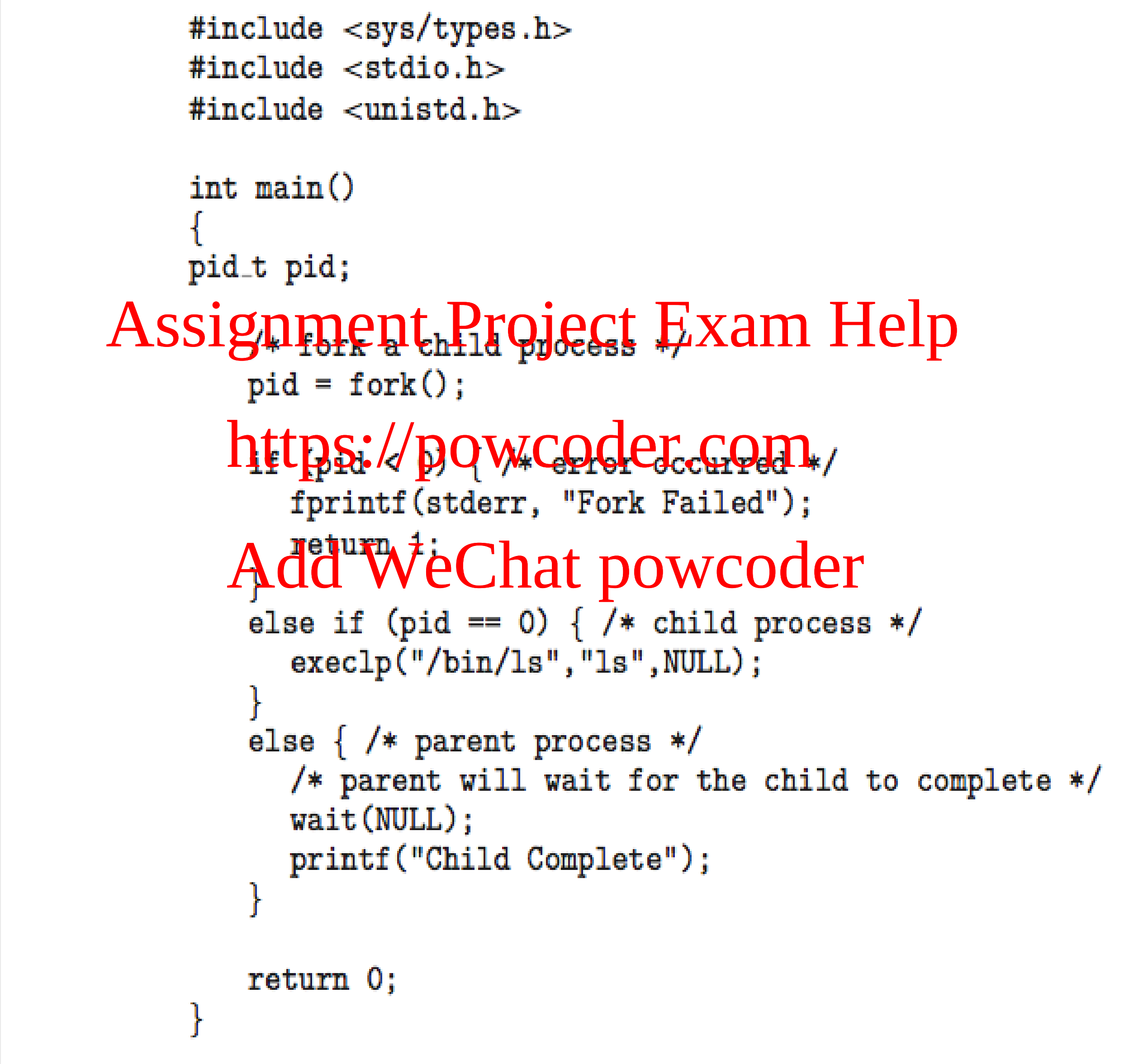
https://powcoder.com

Add WeChat powcoder

- UNIX examples



C Program Forking Separate Process



```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

int main()
{
pid_t pid;

   /* fork a child process */
   pid = fork();

   if (pid < 0) { /* error occurred */
      fprintf(stderr, "Fork Failed");
      return 1;
   }
   else if (pid == 0) { /* child process */
      execlp("/bin/ls","ls",NULL);
   }
   else { /* parent process */
      /* parent will wait for the child to complete */
      wait(NULL);
      printf("Child Complete");
   }

   return 0;
}
```

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Process Termination

- Process executes last statement and asks the operating system to delete it (`exit()`)
  - Output data from child to parent (via `wait()`)
  - Process' resources are deallocated by operating system
- Parent may terminate execution of children processes (`abort()`)
  - 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**

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Process Termination

- Wait for termination, returning the pid:

```
pid t_pid; int status;
pid = wait(&status);
```

Assignment Project Exam Help

- If no parent waiting, then terminated process is a **zombie**

https://powcoder.com

- If parent terminated, processes are **orphans**

Add WeChat powcoder

Interprocess Communication

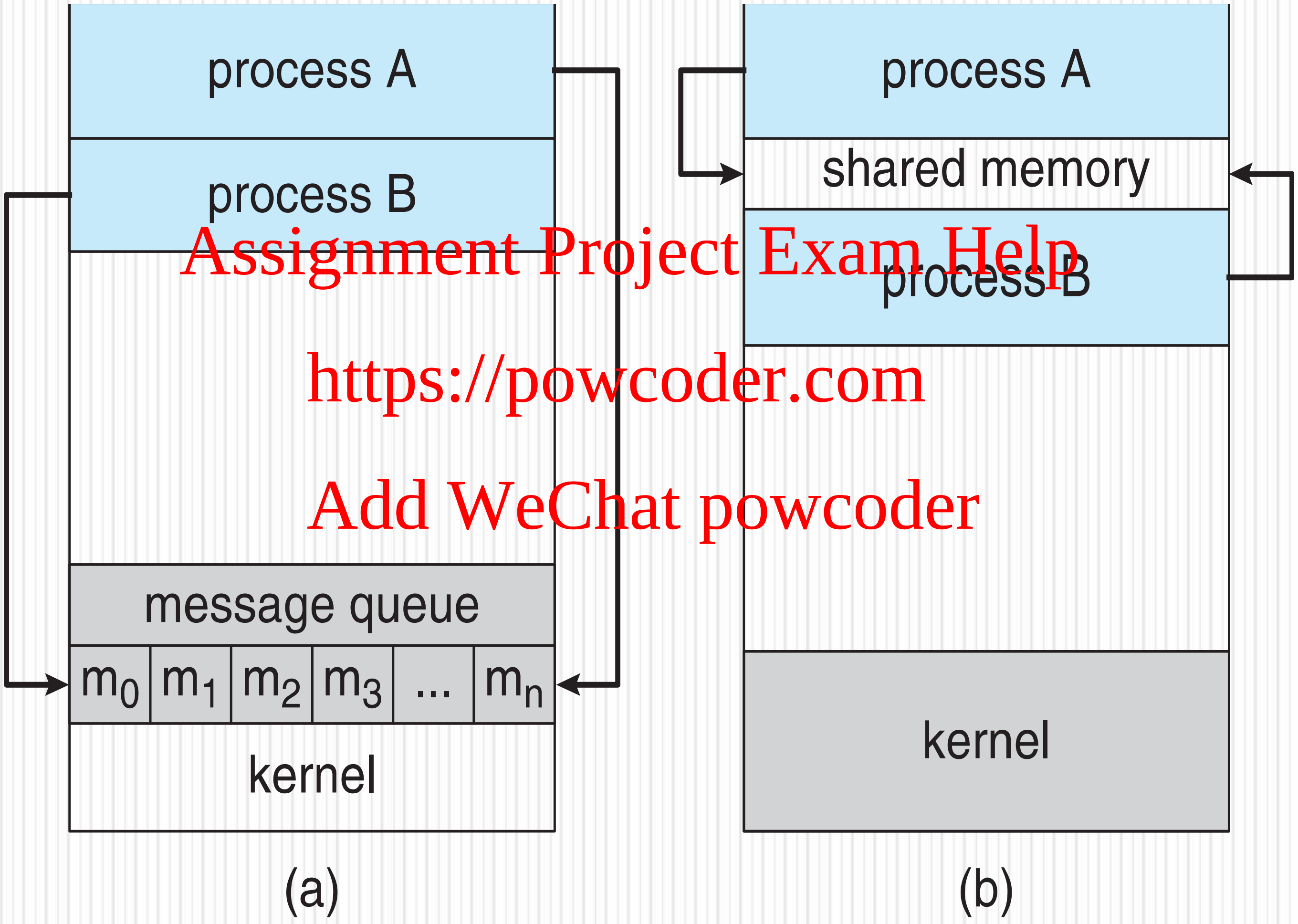
- Processes within a system may be ***independent*** or ***cooperating***
- Cooperating process can affect or be affected by other processes, including sharing data
- Reasons for cooperating processes:
  - Information sharing
  - Computation speedup
  - Modularity
  - Convenience
- Cooperating processes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

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Communications Models



Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Message Passing Vs. Shared Memory

Message passing:

- Useful for exchanging smaller amounts of data
- Easier to implement
- Implemented using system calls
- Every message requires kernel intervention - slower

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Shared memory:

- Useful for exchanging larger amounts of data
- Requires synchronization
- System calls are only required to set up shared-memory region
- Once set up, all accesses are routine memory accesses, no further kernel assistance required - faster

Cooperating Processes

- ***Independent*** process cannot affect or be affected by the execution of another process

Assignment Project Exam Help

- ***Cooperating*** process can affect or be affected by the execution of another process

https://powcoder.com

Add WeChat powcoder

- Advantages of process cooperation
  - Information sharing
  - Computation speed-up
  - Modularity
  - Convenience

Producer-Consumer Problem

- Paradigm for cooperating processes, *producer* process produces information that is consumed by a *consumer* process
  - **unbounded-buffer** places no practical limit on the size of the buffer
  - **bounded-buffer** assumes that there is a fixed buffer size

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Bounded-Buffer – Shared-Memory Solution

- Shared data

```
#define BUFFER_SIZE 10
typedef struct {
    . . .
} item;

item buffer[BUFFER_SIZE];
int in = 0;
int out = 0;
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- Solution is correct, but can only use BUFFER_SIZE-1 elements

Bounded-Buffer – Producer

```
item next_produced;
while (true) {
   /* produce an item in next produced */
   while (((in + 1) % BUFFER_SIZE) == out)
      ; /* do nothing */
   buffer[in] = next_produced;
   in = (in + 1) % BUFFER_SIZE;
}
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Bounded Buffer – Consumer

```
item next_consumed;
while (true) {
     while (in == out)
          ; /* do nothing */
     next_consumed = buffer[out];
     out = (out + 1) % BUFFER_SIZE;

     /* consume the item in next
consumed */
}
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Interprocess Communication – Message Passing

- Mechanism for processes to communicate and to synchronize their actions
- Message system – processes communicate with each other without resorting to shared variables
- IPC facility provides two operations:
  - **`send`**(*message*) – message size fixed or variable
  - **`receive`**(*message*)
- If *P* and *Q* wish to communicate, they need to:
  - establish a ***communication link*** between them
  - exchange messages via send/receive
- Implementation of communication link
  - physical (e.g., shared memory, hardware bus)
  - logical (e.g., direct or indirect, synchronous or asynchronous, automatic or explicit buffering)

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Implementation Questions

- How are links established?
- Can a link be associated with more than two processes?
- How many links can there be between every pair of communicating processes?
- What is the capacity of a link?
- Is the size of a message that the link can accommodate fixed or variable?
- Is a link unidirectional or bi-directional?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Direct Communication

- Processes must name each other explicitly:
  - **send** (*P, message*) – send a message to process P
  - **receive**(*Q, message*) – receive a message from process Q

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

- Properties of communication link
  - Links are established automatically
  - A link is associated with exactly one pair of communicating processes
  - Between each pair there exists exactly one link
  - The link may be unidirectional, but is usually bi-directional

Indirect Communication

- Messages are directed and received from mailboxes (also referred to as ports)
  - Each mailbox has a unique id
  - Processes can communicate only if they share a mailbox

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

- Properties of communication link
  - Link established only if processes share a common mailbox
  - A link may be associated with many processes
  - Each pair of processes may share several communication links
  - Link may be unidirectional or bi-directional

Indirect Communication

- Operations
  - create a new mailbox
  - send and receive messages through mailbox
  - destroy a mailbox

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

- Primitives are defined as:

`send`(*A*, *message*) – send a message to mailbox A

`receive`(*A*, *message*) – receive a message from mailbox A

Indirect Communication

- Mailbox sharing
  - P_1, P_2, and P_3 share mailbox A
  - P_1, sends; P_2 and P_3 receive
  - Who gets the message?

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

- Solutions
  - Allow a link to be associated with at most two processes
  - Allow only one process at a time to execute a receive operation
  - Allow the system to select arbitrarily the receiver. Sender is notified who the receiver was.

Synchronization

- Message passing may be either blocking or non-blocking
- Blocking is considered synchronous
  - **Blocking send** has the sender block until the message is received
  - **Blocking receive** has the receiver block until a message is available
- Non-blocking is considered asynchronous
  - **Non-blocking** send has the sender send the message and continue
  - **Non-blocking** receive has the receiver receive a valid message or null

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Solution to Producer-Consumer Problem using Blocking Send( ) and Receive( )

```
message next_produced;
while (true){
        /* produce an item in next produced */
    send(next_produced);
}

 message next_consumed;
 while (true) {
    receive(next_consumed);
    /* consume the item in next consumed */
 }
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Buffering

- Queue of messages attached to the link; implemented in one of three ways

  1. Zero capacity – 0 messages
     Sender must wait for receiver (rendezvous)
  2. Bounded capacity – finite length of *n* messages
     Sender must wait if link full
  3. Unbounded capacity – infinite length
     Sender never waits

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Communications in Client-Server Systems

- Sockets
- Remote Procedure Calls

Assignment Project Exam Help

https://powcoder.com

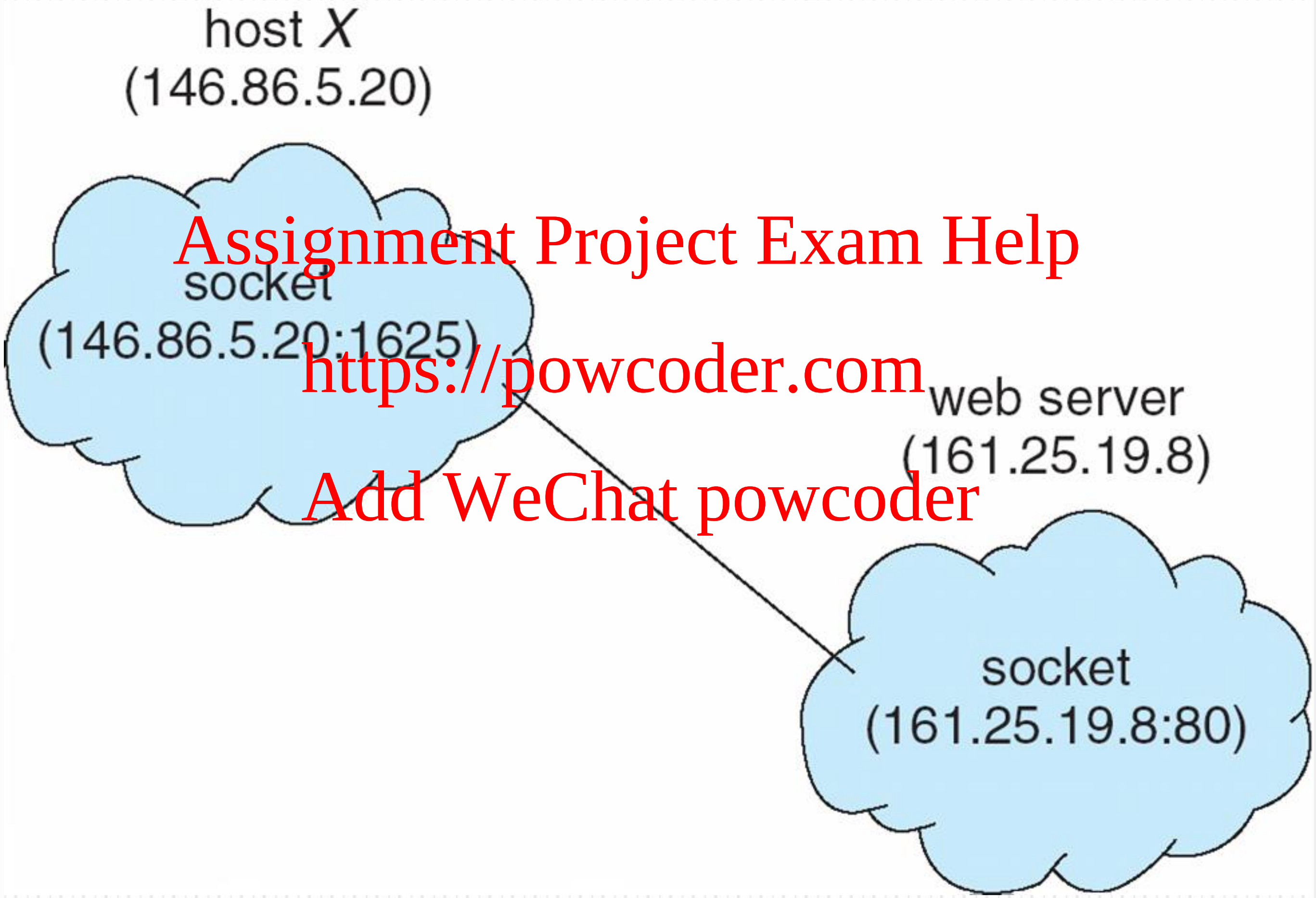
Add WeChat powcoder

Sockets

- A **socket** is defined as an endpoint for communication
- Concatenation of IP address and **port** – a number included at start of message packet to differentiate network services on a host
- The socket **161.25.19.8:1625** refers to port **1625** on host **161.25.19.8**
- Communication consists between a pair of sockets
- All ports below 1024 are ***well known***, used for standard services

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Socket Communication



Assignment Project Exam Help

https://powcoder.com

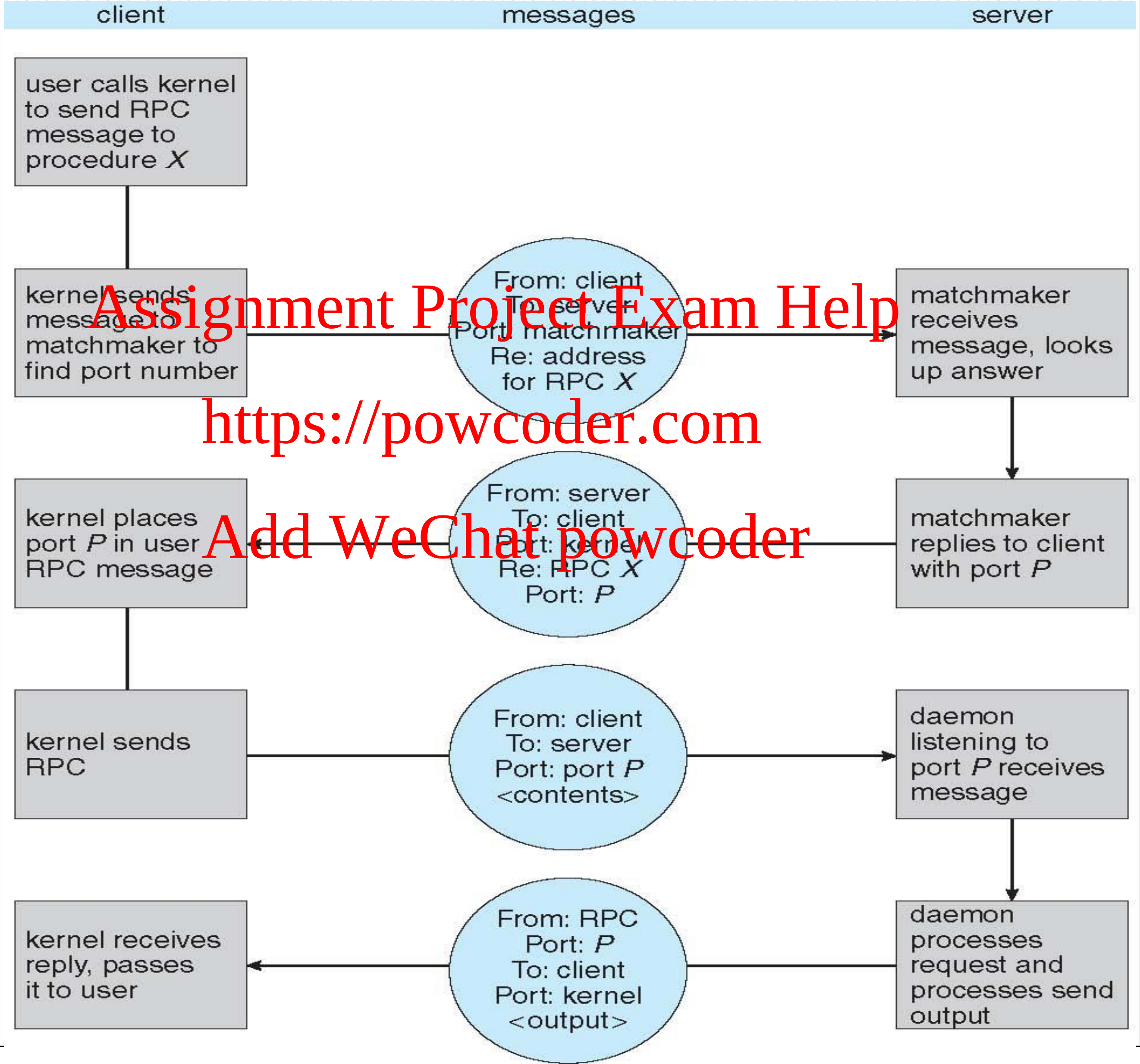
Add WeChat powcoder

Remote Procedure Calls

- Remote procedure call (RPC) abstracts procedure calls between processes on networked systems
  - Again uses ports for service differentiation
- **Stubs** – client-side proxy for the actual procedure on the server
- The client-side stub locates the server and **marshalls** the parameters
- The server-side stub receives this message, unpacks the marshalled parameters, and performs the procedure on the server
- Data representation handled via **External Data Representation** (**XDL**) format to account for different architectures
- Remote communication has more failure scenarios than local
  - Messages can be delivered ***exactly once*** rather than ***at most once***
- OS typically provides a rendezvous (or **matchmaker**) service to connect client and server

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Execution of RPC



client

messages

server

user calls kernel to send RPC message to procedure *X*

kernel sends message to matchmaker to find port number

From: client
To: server
Port: matchmaker
Re: address for RPC *X*

matchmaker receives message, looks up answer

kernel places port *P* in user RPC message

From: server
To: client
Port: kernel
Re: RPC *X*
Port: *P*

matchmaker replies to client with port *P*

kernel sends RPC

From: client
To: server
Port: port *P*
<contents>

daemon listening to port *P* receives message

kernel receives reply, passes it to user

From: RPC
Port: *P*
To: client
Port: kernel
<output>

daemon processes request and processes send output

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

End of Chapter 3

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder