Quizlet

Chapter 5 Operating Systems

Study online at quizlet.com/_2pqks7

1. **3 approaches to achieving mutual exclusion**
   1.] Software - let the application enforce it themselves through coding
   2.] Hardware - disable interrupts or have special machine instructions
   3.] Operating System or language: let the OS or programming language offer support for this capability

2. **3 types of process interaction**
   1.] process not aware of each other
   2.] process indirectly aware of each other
   3.] process directly aware of each other

3. **Basic commands in Message Passing are**
   receive and send

4. **binary semaphore**
   -can only take a value of 1 or 0
   -wait() - if semaphore value is 1, set the value to zero and keep executing the process. If semaphore value is 0, then block the process.
   -signal() - check if any process in the queue is blocked and remove it from queue. If queue is empty set the value to one.

5. **The combinations are possible in message passing for synchronization**
   1.] Blocking send, blocking receive. Both sender and receiver are blocked until message is delivered
   2.] Nonblocking send, block receiver. Sender continues sending messages while reciver is block until he receives a message.
   3.]Nonblocking send, Nonblocking receive. Niether party is waiting

6. **Concurrency occurs in which of the following contexts**
   Multiple Applications, Structured Applications, OS Applications

7. **Concurrent**
   -running at the same time, refers to concurrent activities

8. **Concurrent occurs in**
   Multiprogramming, Multiprocessors, and distributed os

9. **Cooperation among process by sharing**
   Same problem of mutual exclusion, deadlock and starvation. A new problem is data coherence, which means the problem of keeping data in a consistent state.

10. **Cooperation by communication**
    There is no mutual exluxion here, since no resources is being shared. Its all about communicating between processors. Note that starvation and deadlock is still present though. Deadlock in a sense, two processes waiting for each other message. Starvation in sense that (p1,p2,p3) p3 is not getting any message back.

11. **Counting sempaphore**
    -general semaphore with n values

12. **Difference between monitors and semaphores**
    Monitors are more easier to control

13. **Direct Addressing**
    specify destination of process

14. **Hardware support for mutual exclusion**
    1.] Interrupt disabling
    2.] Special machine instructions

15. **Hardware Support: Interrupt disabling**
    problems in interrupt disabling is that it does not work in a multiprocessor environment, since multiple proceses execute at once regardless of interrupts.

16. **Hardware support: Special Machine Instructions**
    A special machine instruction could be designed that carries out two operations atomically, without interruption. Works on uniprocessor and share memory multiprocessor.

17. **In case of competing resources, 3 control problem must be faced**
    First is the need for mutual exclusion. The enforcement of mutual exclusion creates two additional problems. One is deadlock. Final control problem is starvation. A process competing for a resources never receives it due to scheduling of other processes.

18. **Indirect Addressing**
    send to mailbox and let receiving process pickup whenever necssary

19. **Message Passing has two requirements...**
    One is synchronization and the other is communication.

20. **A monitor is a software module with each of the following characteristics**

A monitor is a software module with these chief characteristics:
Local data variables are accessible only by the monitor
Process enters monitor by invoking one of its procedures
Only one process may be executing in the monitor at a time, others are suspended waiting for the monitor to become available

21. **Monitor uses what for synchronization**

condition variables

22. **Passing control back and forth in software solutions is called...**

coroutine

23. **Process directly aware of each other**

Process that communicate each other by process ID and that are designed to work jointly on some activity. Such process exhibits cooperation

24. **Process indirectly aware of each other**

Process that are not necassarliy aware of each other by the respective process IDs but that share access to some object, such as IO buffer. Such process exhibit cooperation in sharing the common object

25. **Process unaware of each other**

These are independent process that are not intended to work togather. These can either be batch jobs or interactive sessions. Processes are not working togather, but OS needs to be concerned about the competion for resources. For example, two independent application may both want to access the same disk or file or printer. OS must regulate these accesses

26. **Producer/Consumer Problem**

In binary semaphores, the values must be saved inside. You can make a counter semaphore too, but make sure where your placement of semaphore is done. You don't want to deadlock the system.

27. **Reader/Writer Examle**

Read slide

28. **Requirements for mutual exclusion**

1.] Enforcement-only one processor at a time
2.] Non-interference - a process that halts in non critical section does not affect other processers
3.] No deadlock or starvation needing to enter cricitical section
4.] When critical section open, any processor can go
5.] No assumption should be made about speed of process
6.] Finite time inside critical section

29. **Semaphore**

-wait() decrements a value
-signal() increments a value
-the value may be initialized to nonnegative number

30. **Some of the problems with concureny**

Sharing of resources, allocation of resources, debugging

31. **Strong semaphore**

-queue uses FIFO ordering. There is no starvation

32. **The various schemes for specifying processes in send and receive primitives fall into two categories:**

direct addressing and indirect addressing

33. **Weak Semaphore**

- No ordering in queue. Probably would have a starvation

34. **What are OS design concerns in concurrency?**

-allocation of resources(Deadlock), keep track of various process using pcb, protect data and resources from interference of another process, process must be independ of results

35. **What are some problems with Hardwrae support: Special Machine Instructions**

1.] Busy-waiting consumes processor times
2.] Starvation possible
3.] Deadlock possible if higher priority process comes in. p2 is higher priority and p1 is low priority. p1 is interrupted because p2 is high priority, but p1 can't go inside because of mutual exlusion but its higher priority so it hs to go