

Resource (1)

- A **resource** is a commodity needed by a process.
- Resources can be either:
 - **serially reusable:** e.g., CPU, memory, disk space, I/O devices, files.
acquire → use → release
 - **consumable:** produced by a process, needed by a process; e.g., messages, buffers of information, interrupts.
create → acquire → use
Resource ceases to exist after it has been used, so it is not released.

Resource (2)

- Resources can also be either:
 - **preemptible:** e.g., CPU, central memory or
 - **non-preemptible:** e.g., tape drives.
- And resources can be either:
 - **shared** among several processes or
 - **dedicated** exclusively to a single process.

Using Semaphore to Share Resource

	Process P();	0	→	Process Q();	
2	→	{ A.Wait();	6	→	{ A.Wait();
3	→	B.Wait();			B.Wait();
		use both resource			use both resource
4	→	B.Signal();			B.Signal();
5	→	A.Signal(); }			A.Signal(); }

External Semaphore A(1), B(1);

- 2 External Semaphore A(0), B(1);
- 3 External Semaphore A(0), B(0);
- 4 External Semaphore A(0), B(1);
- 5 External Semaphore A(1), B(1);

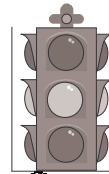
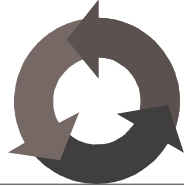
But Deadlock can Happen!

1	→	Process P();	1	→	Process Q();
2	→	{ A.Wait();	3	→	{ B.Wait();
		B.Wait();			A.Wait();
		use both resources			use both resources
		B.Signal;			A.Signal();
		A.Signal;			B.Signal(); }

DEADLOCK

- 1 External Semaphore A(1), B(1);
- 2 External Semaphore A(0), B(1);
- 3 External Semaphore A(0), B(0);

Deadlock



Mechanisms for Deadlock Control

Oct. 24, 2007

CSE660: Introduction to Operating Systems

5

Deadlock Definition

What is a deadlock?

- A process is **deadlocked** if it is waiting for an event that will never occur.
- Typically, but not necessarily, more than one process will be involved together in a deadlock (the *deadly embrace*).

Is deadlock the same as starvation (or infinitely postponed)?

- A process is **infinitely postponed** if it is delayed repeatedly over a *long* period of time while the attention of the system is given to other processes. i.e. logically the process may proceed but the system never gives it resources.

Oct. 24, 2007

CSE660: Introduction to Operating Systems

6

Conditions for Deadlock

- What conditions should exist in order to lead to a deadlock
- Real life analogy such as
 - "You take the monitor, I grab the keyboard"
 - Cars in intersection

Necessary and Sufficient Conditions for Deadlock

- Mutual exclusion**
 - Processes claim **exclusive** control of the resources they require
- Wait-for condition**
 - Processes hold resources already allocated to them while waiting for additional resources
- No preemption condition**
 - Resources cannot be removed from the processes holding them until used to completion
- Circular wait condition**
 - A circular chain of processes exists in which each process holds one or more resources that are requested by the next process in the chain