# Deadlocks

Under the normal mode of operation, a process may utilize a resource in only the following sequence:

**1. Request**. The process requests the resource. If the request cannot be granted immediately (for example, if the resource is being used by another process), then the requesting process must wait until it can acquire the resource.

1. **Use. The process can operate on the resource (for example, if the resource is a printer, the process can print on the printer).**
2. **Release. The process releases the resource.**

**Conditions for deadlock:**

1. **Mutual exclusion. At least one resource must be held in a nonsharable mode; that is, only one process at a time can use the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.**
2. **Hold and wait. A process must be holding at least one resource and waiting to acquire additional resources that are currently being held by other processes.**
3. **No preemption. Resources cannot be preempted; that is, a resource can be released only voluntarily by the process holding it, after that process has completed its task.**
4. **Circular wait. A set { *P*0 , *P*1 , ..., *Pn* } of waiting processes must exist such that *P*0 is waiting for a resource held by *P*1, *P*1 is waiting for a resource held by *P*2, ..., *Pn*−1 is waiting for a resource held by *Pn*, and *Pn* is waiting for a resource held by *P*0.**

Deadlocks can be described more precisely in terms of a directed graph called a **system resource-allocation graph**. This graph consists of a set of vertices *V* and a set of edges *E*. The set of vertices *V* is partitioned into two different types of nodes: *P* = {*P*1, *P*2, ..., *Pn*}, the set consisting of all the active processes in the system, and *R* = {*R*1, *R*2, ..., *Rm*}, the set consisting of all resource types in the system.

A directed edge from process *Pi* to resource type *Rj* is denoted by *Pi* → *Rj* ; it signifies that process *Pi* has requested an instance of resource type *Rj* and is currently waiting for that resource. A directed edge from resource type *Rj* to process *Pi* is denoted by *Rj* → *Pi*; it signifies that an instance of resource type *Rj* has been allocated to process *Pi* . A directed edge *Pi* → *Rj* is called a **request edge**; a directed edge *Rj* → *Pi* is called an **assignment edge**.

We can deal with the deadlock problem in one of three

ways:

* We can use a protocol to prevent or avoid deadlocks, ensuring that the

system will ***never*** enter a deadlocked state.

* We can allow the system to enter a dead locked state, detect it, and recover.
* We can ignore the problem altogether and pretend that deadlocks never occur in the system.

Design:

For the second part of the lab, we designed a vending machine (which is implement using FSM) which dispensed to product: Phone case or earbuds. The price of the phone-case is $10 and earbuds is $15. The inputs to the vending machine include the product selection (i.e sel\_item) signal and money inserted by the customer. The machine accepts denominations of $5, $10, $20 and $50 only. If sel\_item signal is ‘0’ than phone-case is selected otherwise earbuds is selected. The customer first needs to select which product they want to purchase and then they start inserting the money. If the vending machine have received money greater than equal to price of the selected product, then it will give out the change and dispense the product, else it will ask customer to add more money. The seven-segment display is used to display the current sum of money that have been added to machine.

Diagram

Description automatically generated  
**Figure 1: The above diagram shows block diagram for Vending Machine FSM.**

The clock signal is used to synchronous the input in the machine and reset signal is activated when product is dispensed and change out is given. Select\_segement and sum\_display is used to display the coins inserted in machine to seven-segement display. Moreover, coins and sel\_item are the input to machine.