Semaphores

Centurion University
Shaping Lives...
Empowering Communities...

- Semaphores are typically used for mutual exclusion, access control to shared resources, and basic synchronization
- Can be described as counters used to control access to shared resources by multiple processes [threads].
  - eg: Multiple request drivers accessing a shared bus
- Implemented as a bucket with fixed number of keys
  - Processes using semaphores must first procure a key before executing
  - All others must wait until a sufficient number of keys is returned to the bucket.

Centurion
UNIVERSITY
Shaping Lives...
Empowering Communities...

Semaphore Methods

Semaphore provides following built-in methods:

Method	Use
new()	Create a semaphore with specified number of keys.
put()	Return one or more keys back.
get()	Obtain one or more keys.
try_get()	Try to get one or more keys without blocking.

Centurion
UNIVERSITY
Shaping Lives
Empow

Semaphore example

Example 7-25 Semaphores controlling access to hardware resource

```
program automatic test;
  semaphore sem;              // Create a semaphore
  initial begin
    sem = new(1);             // Allocate with 1 key
    fork
      sequencer;              // Spawn two threads that
      sequencer;              //   do bus transactions
    join
  end

  task sequencer;
    repeat($urandom%10)       // Random wait, 0-9 cycles
      @bus.cb;
    sendTrans;                // Execute the transaction
  endtask

  task sendTrans;
    sem.get(1);               // Get the key to the bus
    @bus.cb;                  // Drive signals onto bus
    bus.cb.addr <= t.addr;
    ...
    sem.put(1);               // Put it back when done
  endtask
endprogram
```