COMPUTER SCIENCE

Process Synchronization

Lecture 05

Dr. KHALEEL KHAN SIR

TOPICS TO BE COVERED

- Semaphore

Session-II : 12/11/2022

→ Bounded is NOT Satisfied with TSL → (Multi-Process Solution)

* H/W In the text-Book of Galvin

Additional Logic

Piece of Code in entry

II: SWAP : Privileged/ATOMIC Instn

→ Lock-based ; LOCK-KEY

→ ATOMIC

```
void SWAP(Bool *a, Bool *b)
{
        Bool t;
Atomic {  t = *a;
          *a = *b;
          *b = t;
}
```

Guarantee

M.E ✓

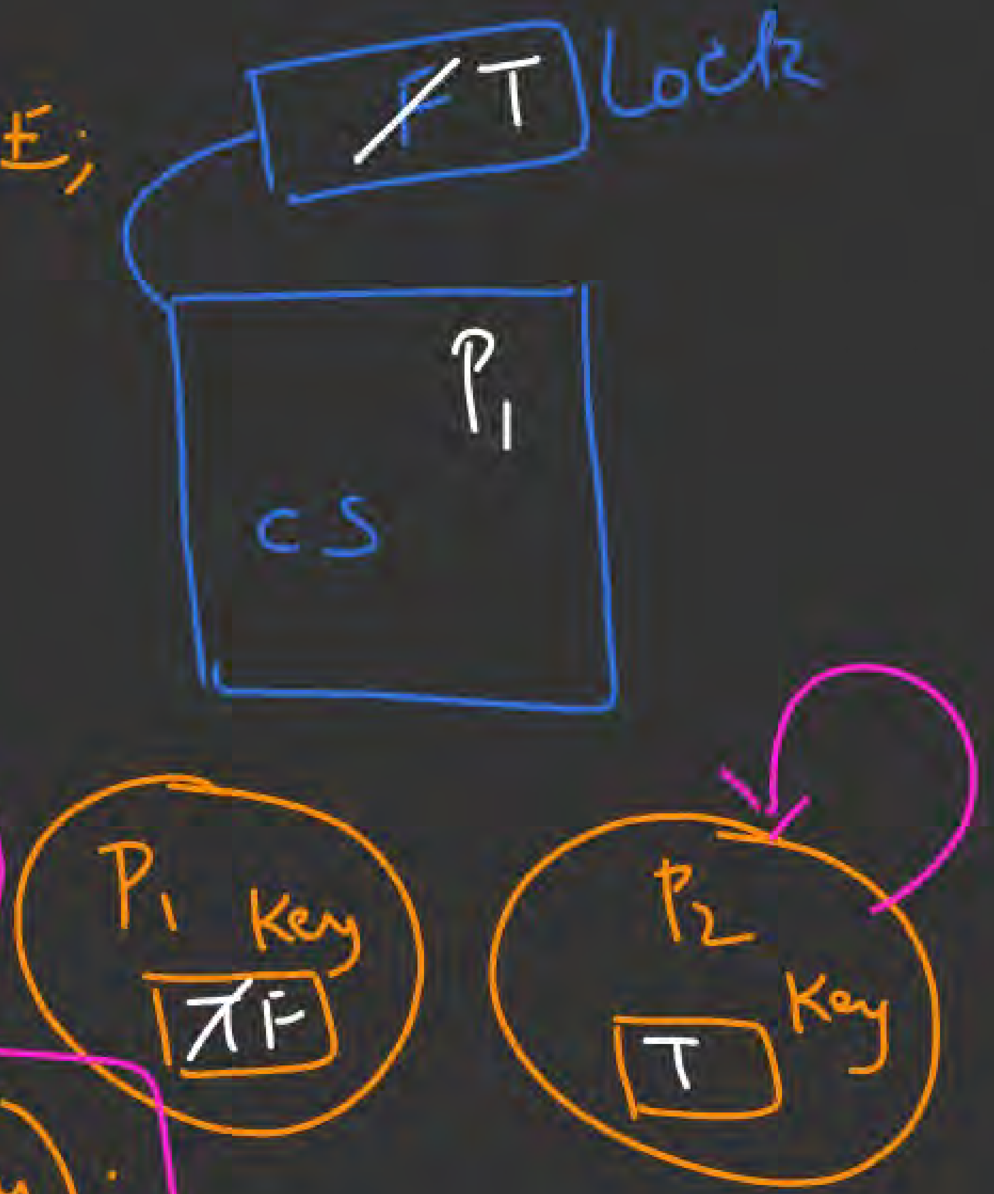
Progress: ✓

Bounded waiting:

UM: Bool lock = FALSE;

```
void Process(int i)
{
    Bool Key = TRUE;
    while(1)
    {
      a) Non_CS();
      b) do
         {
            SWAP(&lock, &Key);
         } while(Key == TRUE);
      c) <CS>
      d) Lock = FALSE;
    }
}
```

Logic



t_1: P_1 : a, b; SWAP

t_2: P_2 : a, b: SWAP

Tanenbaum

Apart from Busy-waiting Causing wastage of cpu-time, they also Suffer from the problem of Priority-Inversion:

<TSL + SWAP + PET + DEK>

: Pre Emptive Priority based Scheduling

H, L : Priority

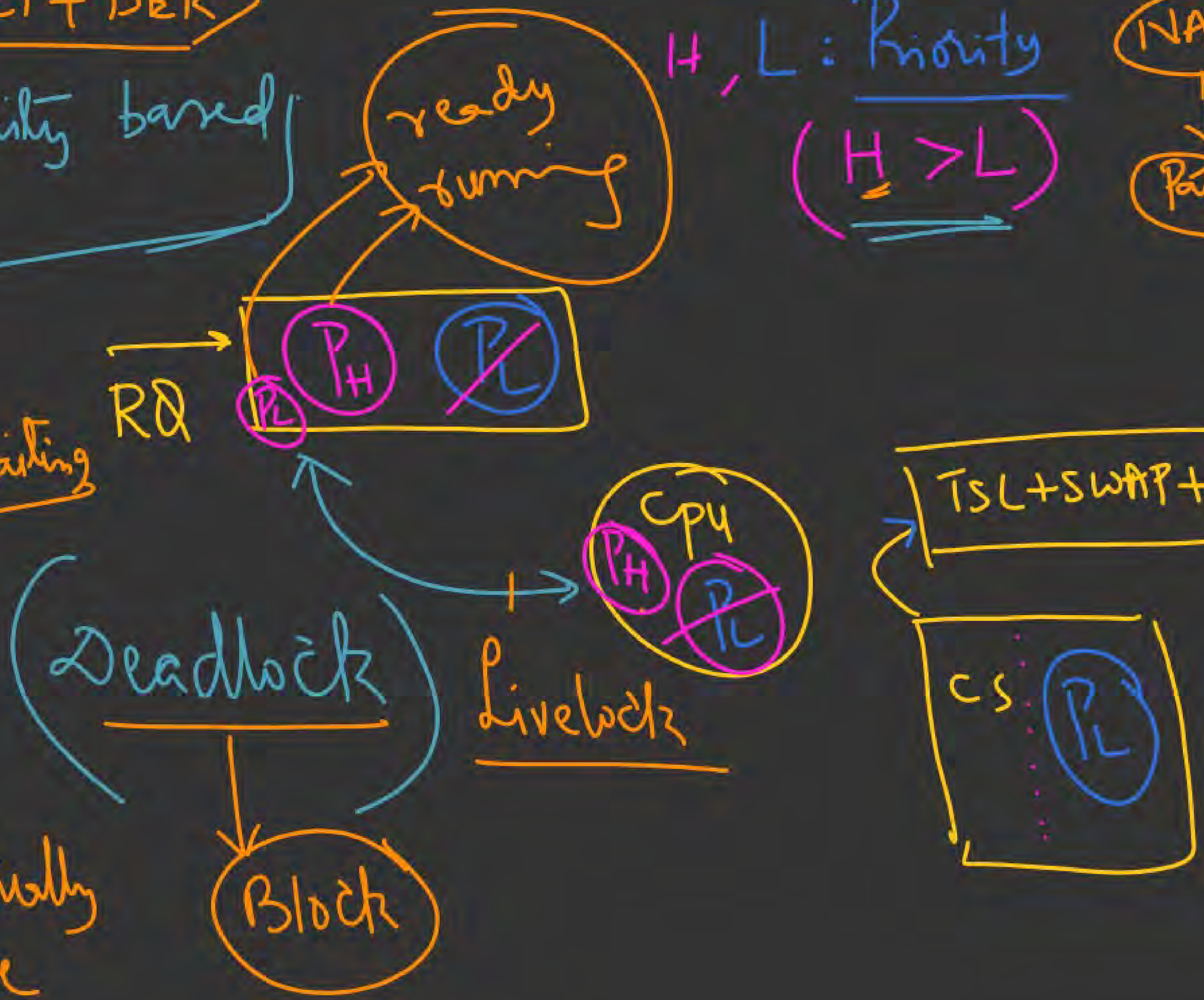
(H > L)

NASA → Pathfinder

Lock:

- Spinlock: Busy-waiting
- Deadlock
- Livelock

Conceptually Same



CPU: ✓

I. There is no Problem arising if P_H is NOT intrstd in CS;

II P_H is also Intrstd in CS

Priority-Inheritance Protocol

⊛ III : **OS-Based** : (Blocking Mechanisms
Non-Busy-waiting)

- → Sleep-wakeup
- → SEMAPHORE
- → *MONITOR
  - thru P.L

5) Sleep() & wakeup()
- → OS primitives (K M)
- → Multi-process soln
- → They are used in UM like Sys-Calls.

(i) Inconsistency

Does this Code has a Deadlock

G

```
#define N 100
integer Buffer[N];
int Count = 0;
   <No. of Data items
      in Buffer>
```

```
void Producer(void)
{
  int itemp, in=0;
  while (1)
  {
    a) itemp=Produce_item();
    b) if (Count == N) | Sleep();
    c) Buffer[in]=itemp;
    d) in =(in+1) % N;
    e) Count= Count+1;
    f) if (Count == 1) Wakeup(Consumer);
  }
}
```

```
void Consumer(void)
{
    int itemc, out=0;
    while (1)
    {
      a) if (Count == 0) | Sleep();
      b) itemc= Buffer[out];
      c) out=(out+1) % N;
      d) Count = Count-1;
      e) if (Count == (N-1)) Wakeup
                            (Producer);
      f) Process_item(itemc);
    }
}
```

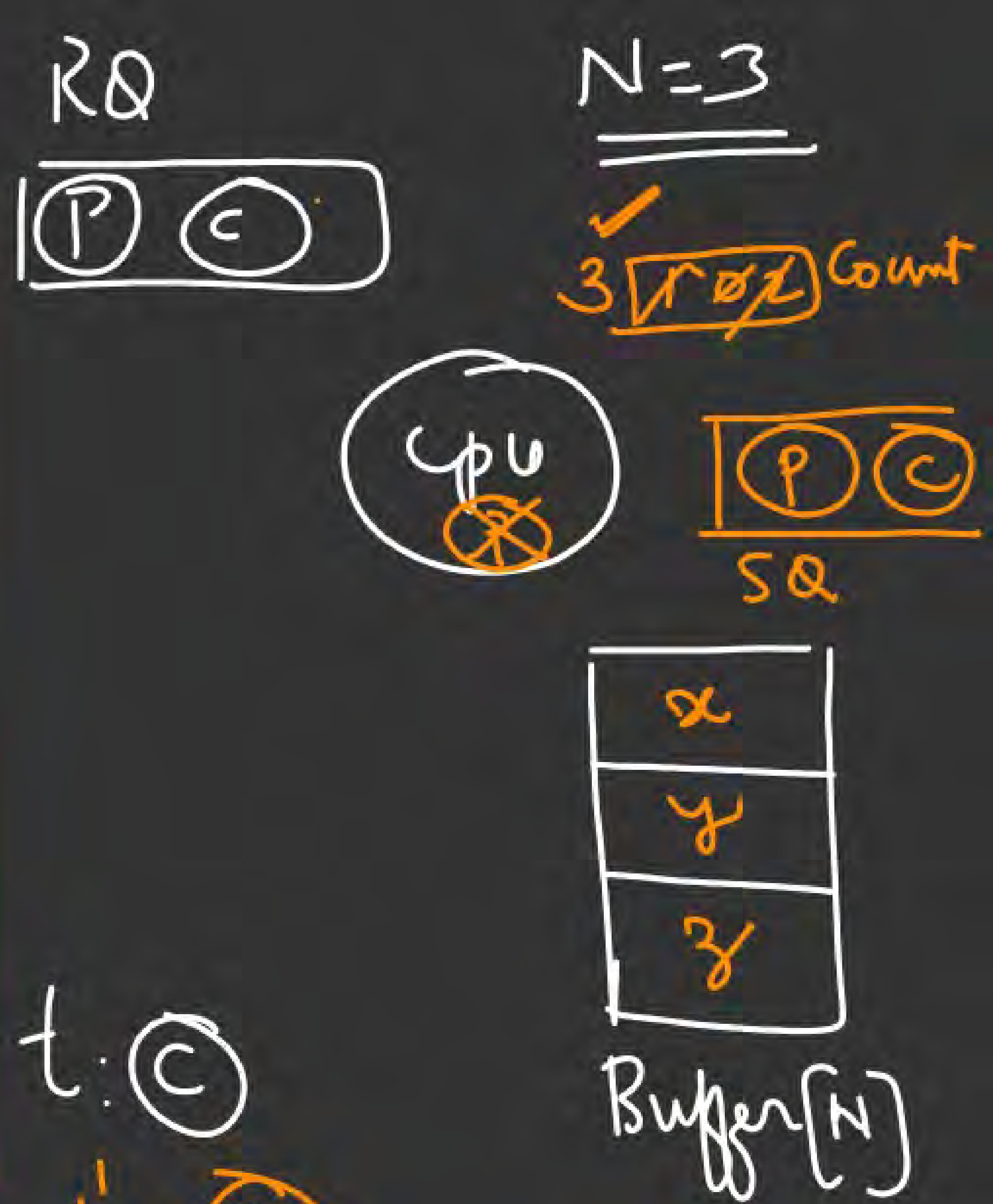
?

Cooper

PH

Compet

Inconsistency



t : C

t' : P .... Sleep

t'' : C : Sleep();

PrePmption

Deadlock

XI: SEMAPHORES:

→ OS based Mechanism

→ Proposed by Dijkstra;

→ Blocking Mechanism

→ * General purpose utility

→ Semaphore is implemented as an A.D.T

Wait ⟷ Signal | release

DOWN ⟷ up

P ⟷ V

→ Semaphore operations are Atomic (KM)

Defn : is a variable (ADT: SEM) that takes only Integer values;

BINARY (0|1)

General COUNTING ⟨−∞ to +∞⟩

THANK YOU!