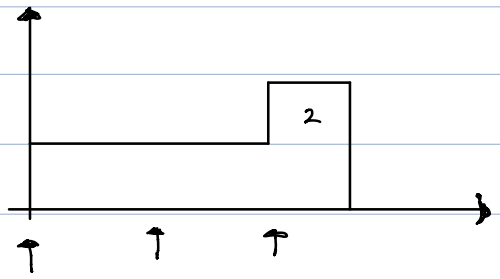
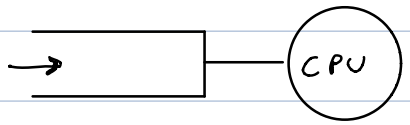
Questions Regarding Project

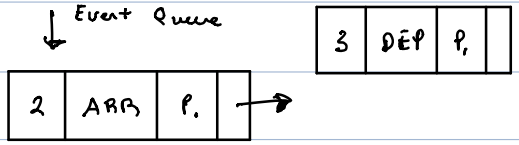


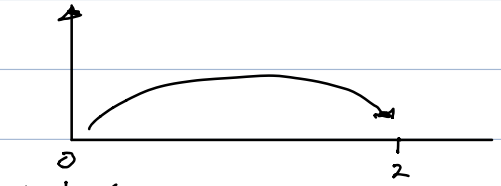
Periodically sample the Queue

Ready Queue Logic



Clock : Current time



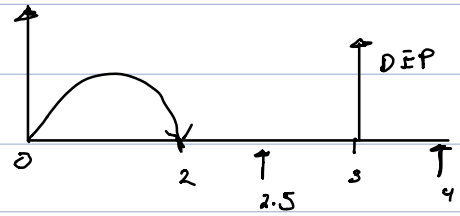


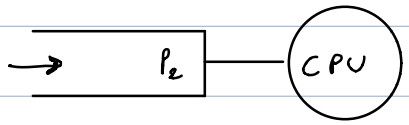
Clock ∅

ARR P_1

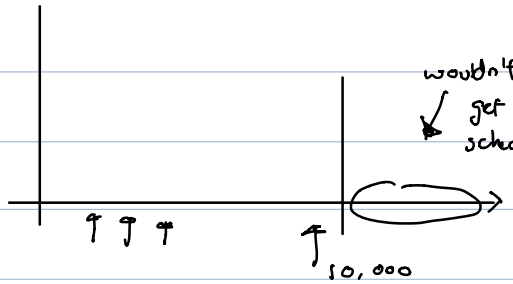
Based on current time

- With something being non-preemptive it should know the exact time

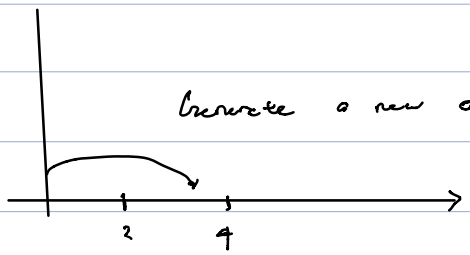




- Two Approaches



- Based on the Arrival, then create a new event



```
Signal (Semaphore *s)
    s->value ++;
    if (s->value <= 0){
        remove P from s->list;
        wake up (P)
    }
}
```

FIFO
No starvation

```
while (1) {
    wait(s)
    C.S.
    signal(s)
    R.S.
}
```

S = 1 S = 0

Deadlock

Situation in which 2 or more process wait for an event that can only happen by one of the waiting processes

e>//

P_1	P_2	P_1 P_2 may be
wait(S);	wait(Q);	deadlocked
wait(Q);	wait(S);	
C.S	C.S	
signal(Q);	signal(S);	
signal(S);	signal(Q);	

Priority Inversion

Higher is waiting on lower priority process

1 High — 5 Medium — ~~3~~ 1 Low (Effectively gets preemptive)

→ [R]

↑ Running and holding resource

↓ wouldn't be able to release resource

Priority Intertience

A process accessing a resource needed by a higher priority one, would inherit the higher priority one, until done

- Doesn't get preempted by any process

• Classical Problems of Synchronization

1 • Bounded buffer problem

pool of n buffers

each hold one item

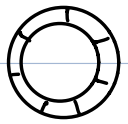
producer → produces an item and adds it to the buffer

consumer → removes an item from the buffer and consumes it.

- Can't produce item if bucket is full
- Can't consume from empty pool

N number of elements

mutex → access to buffer



empty → counts # of empty buffers

full → counts # of full buffers

Full = 0

Empty = n

Producer :

```
do {
    produce an item;
    wait (empty);
    wait(mutex);
    add item to buffer
```

Consumer :

```
do {
    wait (full)
    wait(mutex)
    get-item
    signal (mutex)
```

signal (mutex)

signal (full)

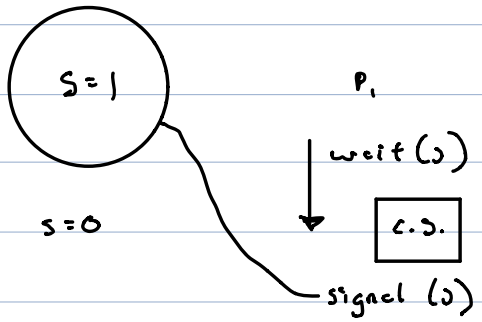
} while (1)

signal (empty)

consume

} while (1)

• Semaphores that are binary are mutexes



wait will be done n times without stopping.

signal full will increment by 1

FCFS

$\rho = \lambda T_s$

$= 5 \cdot 0.06$

$= 0.3$

$T_q = q$