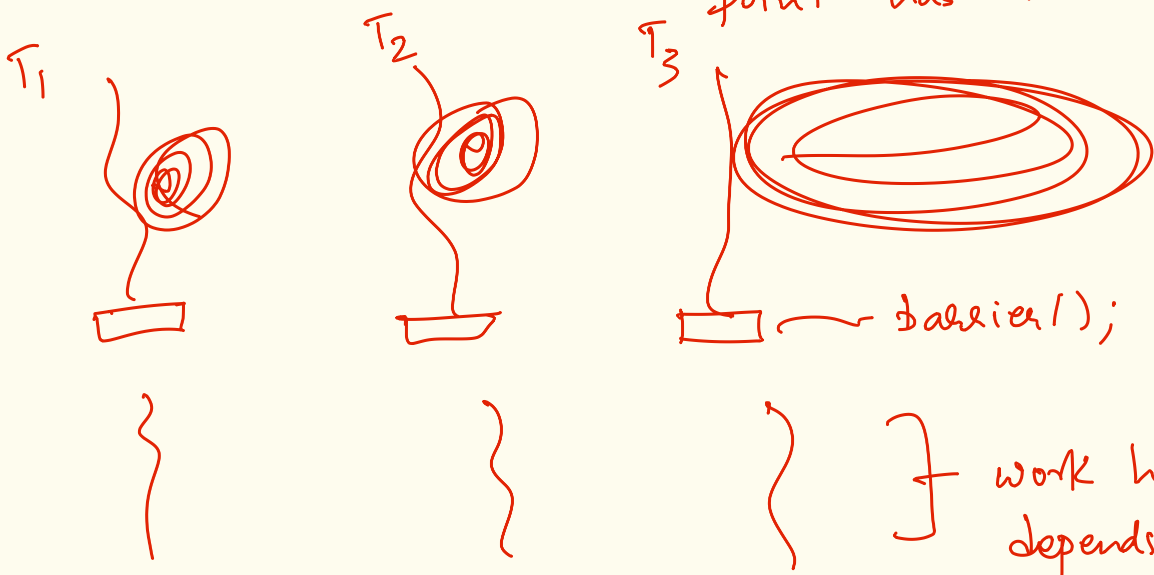
CS219 Lecture 29

\# spinlocks, disable interrupts, disable pre-emption

lock; spinlock, mutex

condition variables, semaphores

\# barrier ~ primitive to hold all threads till a condition/execution point has reached.



work here depends on state of completion of all threads before barrier.

(*) implement a barrier.

- init ~ how many instances should reach the barrier!
- barrier ~ wait for barrier condition.

(*) barrier using semaphores.

- init
- down
- up

semaphore S_1, S_2 $\quad$ S_1.init(K);

S_2.init(0);

```
barrier() {
    down(S1);
    if (S1 == 0)  S2.init(K);
    down(S2);
}
```

a reset of semaphore will also check for wakeup condⁿ.

← sleep / block till all threads reach barrier;

H.W — correctness of nested barriers.

barrier(B);	barrier(B);	barrier(B);
barrier(B);	barrier(B);	barrier(B);

1. verify that above implementation has / does not have a problem.

2. if problem, what solution?

   if no problem, really?

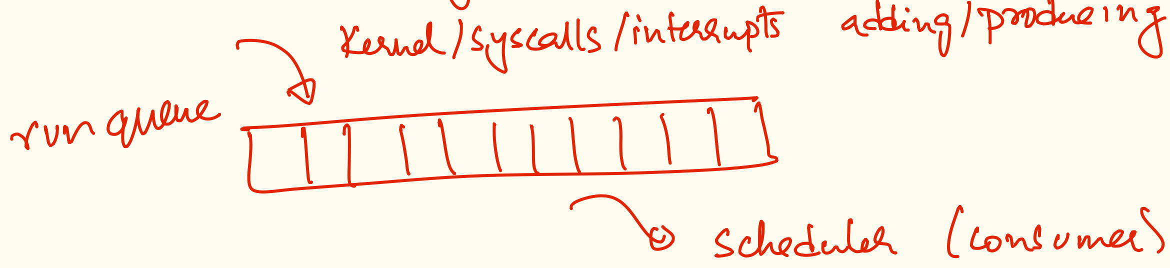
assume.

S_2.init(K)

S_1.init(K)

as part of the if($S_1 == 0$) condition.

Producer- consumer synchronization problem.



- ~ producers add to queue till size of queue
- ~ consumers consume / dequeue from queue if queue not empty.

} atomic / synchronized

implement. : MAX. is size of queue.

using Semaphores
a solution of the prod- con synchronization problem:

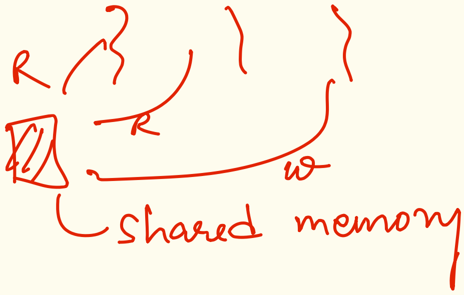
```
semaphores Sp(MAX);   Sc(0);

producer() {                      consumer() {

    down(Sp);                         down(Sc);
  spinlock(L);                      spinlock(L);
    K= (index ++ % MAX);            index=(index - 1) % MAX;
  spinunlock(L);                    spinunlock(L);
    addqueue(K);                      dequeue (index);
    up(Sc);
                                      up(Sp);
}                                 }
```

or use a pindex
& cindex

reader-writer synchronization problem.



~ if reader; allow other readers
; do not allow other writers

if writer; do not allow readers
or other writers.

(H.W2) implement RW lock using condition variables.

Reader Lock

Writer Lock

Writer Unlock

Reader Unlock