Exe 13

Deadlock

Please consider the below execution flow.

Initially: a=1, b=1, c=1			
Step	**Thread 1**	**Thread 2**	**Thread3**
Step1	**P(a)**	**P(b)**	**P(b)**
Step2	**P(b)**	**P(c)**	**V(b)**
Step3	**V(a)**	**V(c)**	**P(c)**
Step4	**P(c)**	**P(a)**	**P(a)**
Step5	**V(b)**	**V(b)**	**V(c)**
Step6	**V(c)**	**V(a)**	**V(a)**

1. Does it cause deadlock? Please show at least two execution flow that will cause deadlock (You just need to answer the states of involved threads. e.g. After T1 finishes step1 and T2 finishes step2) .
2. If we remove **Thread3**, will it cause deadlock? Please draw the **progress graph** and explain your reason.

Spin lock

Sam modifies the ticket lock by adding one line "**`IDLE(...);`**" in the while-loop. Note(**`IDLE(n)`** will consume **`c*n`** CPU cycles where c is a user-defined constant

The original ticket lock	A modified ticket lock

`void lock(lock_t *lock) {`
`  int my turn =`
`  FetchAndAdd(&lock->ticket);`
`  while (lock->turn != myturn)`
`  ;`
`}`
	`void lock(lock_t *lock) {`
	`  int my turn =`
	`  FetchAndAdd(&lock->ticket);`
	`  while (lock->turn != myturn)`
	`  IDLE(myturn - lock->turn);`
	`}`

1. What's the **advantage** after adding the "**`IDLE(...);`**" line?
2. What's the disadvantage after adding the "**`IDLE(...);`**" line?
   HINT: Consider what if an inappropriate constant ***C*** is chosen.
3. For setting **IDLE** time, why to use "**`myturn - lock->turn`**" rather than a fixed value?

Schedule

Suppose we use following MLFQ scheduling policy with the give assumption.
1. There are 3 priority queues **Q0, Q1, Q2**. **Q0** has the lowest priority, **Q2** has the highest priority.
2. **FIFO** is used in each queue (Not **RR**).
3. The time slices are 8ms(**Q0**), 4ms(**Q1**), 2ms(**Q2**).
4. Priority boosting is not supported.
5. Scheduling is carried out only at completion of processor time slices.
Give the information of the following jobs of workload (Note: No I/O issue are involved.)

Job	Arrival Time	Runtime
A	0ms	7ms
B	3ms	5ms
C	6ms	13ms
D	12ms	8m

1. Please fill the table showing the state of CPU and the processing state of the CPU.

Timeline	0	2	3	6
CPU	A->2	A->6	A->6	B->8
Q2	A(7)		B(5)	B(5)C(13)
Q1		A(5)	A(4)	
Q0				A(1)

2. Compute the response time and turnaround time. Is the turnaround time good as expected? If not, try to tell the reason.