Solutions to practice questions on Processes and Interrupts

There are either software interrupts or hardware interrupts.
Interrupts could be due to:

- a program requesting the use of hardware devices. The hardware will send an interrupt signal to the CPU to update the OS with the status of the request;
- OS activities, e.g. a hardware clock could trigger a timer interrupt to inform the OS of a pre-scheduled activity;
- hardware events, e.g. a device failing or a user typing on a keyboard;
- software exceptions and errors, e.g. a division by zero.

Solutions to practice questions on Processes and Interrupts

Q1: All n interrupts could be due to the program making IO requests.

max: n
min: 0

None of the interrupts could be due to the programs requests. Instead, they could all be triggered for other reasons, e.g. timer interrupts from a hardware clock to initiate scheduled OS activities

Q2: All n interrupts could be due to the program making IO requests.

max: n
min: 0

None of the interrupts could be due to the programs requests. Instead, they could all be triggered for other reasons, e.g. hardware failure or interrupts from users typing on a Keyboard.

Q3: All n interrupts must be due to the program making IO requests. In particular,

max: n
min: n

since the program successfully terminates and all hardware interrupts are in service to the program's request, hardware failures couldn't have occurred.

Solutions to practice questions on Processes and Interrupts

Q4: In this question the program could have failed and terminated as a consequence. So, "n-1" of the interrupts could have been due to OS activities, and the "nth" interrupt could have been because the program failed (e.g. a division by zero occurred).

max: 1
min: 0

So, since the program issues no IO requests, at most 1 of the "n" interrupts – i.e. the nth interrupt – is due to the program.

Of course, none of the interrupts could be due to the program; e.g. OS timer interrupts.

Q5: All of the interrupts are due to either:

max: n
min: n

- the program making IO requests;
- the program making IO requests and failing.

Solutions to practice questions on Processes and Interrupts

No IO requests!

Q6: "n-1" interrupts must not have been due to the program, followed by the software failing (i.e. the "nth" interrupt). Note: the software must fail.

max: 1
min: 1

The question said so!

Q7: "n-1" interrupts could be due to OS activities, with the program failing and causing the "nth" interrupt.

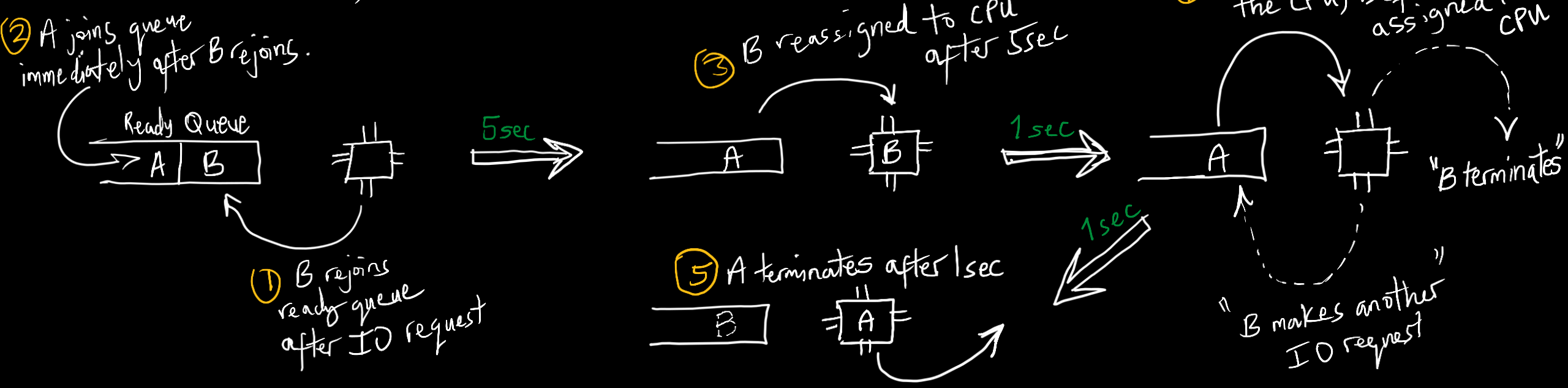
All of the interrupts could be due to OS activities.

max: 1
min: 0

Solutions to practice questions on Processes and Interrupts

Q8: A arrives in the ready queue immediately after B has made an IO request and re-entered the queue. After 5 sec the IO request has been serviced and B is reassigned to the CPU. B will execute for, at most, a further 1 sec before it either terminates or returns to the queue. After which A is assigned to the CPU, executes for 1 sec, and then terminates. So, at most, A takes 7 sec to execute.

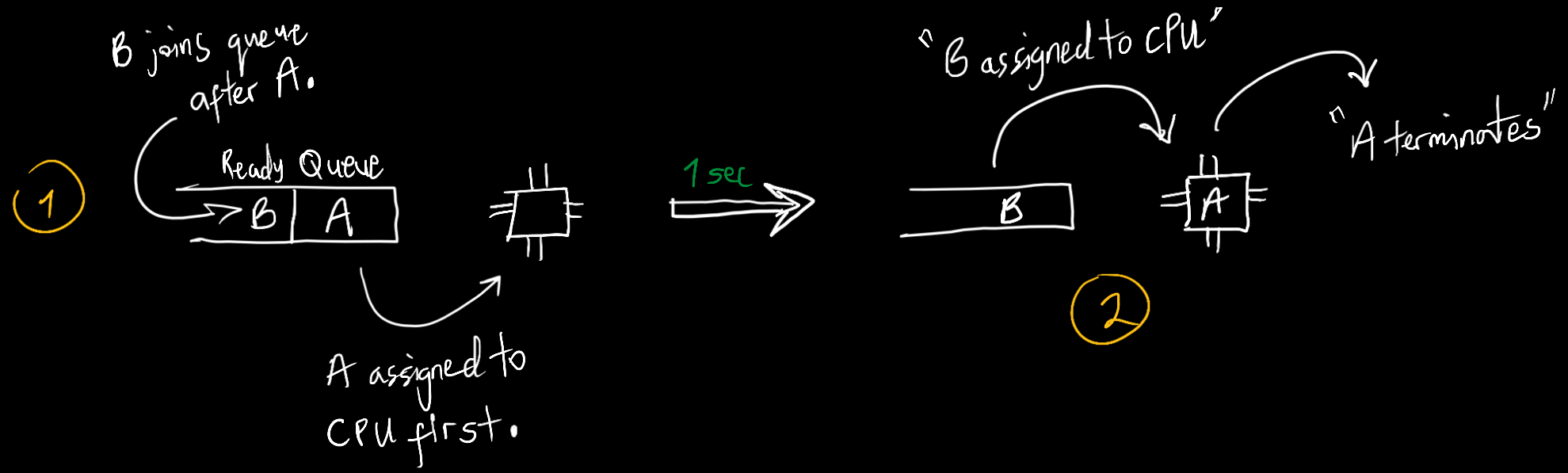
max: 7s



Solutions to practice questions on Processes and Interrupts

Q8: A could arrive in the ready queue before B, so A gets assigned to the CPU first. It executes for 1sec before terminating.

min: 1s



Solutions to practice questions on Processes and Interrupts

Q9: A could arrive in the ready queue immediately after B is assigned to the CPU.

B could take, at most, 1sec before the CPU is reassigned to A (i.e. B makes an IO request at 1sec). Then A executes for 1sec before terminating. In total, A has spent 1+1 = 2sec executing in the system.

max: 2s
min: 1s

Or A could have arrived in the ready queue first. So it gets the CPU first and only spends 1sec before terminating.

Solutions to practice questions on Processes and Interrupts

Q10: The solution is the same as Q9, because the programs use only a negligible amount of RAM. So, the "suspend" states are not entered into.