Process Synchronization in XV6

#	2

Process Synchronization in XV6

FRQ Repo: `https://github.com/susantoscott/project2-frq`

Part 1: 3 Processes - a Parent and 2 Children

- **What is the effect of the parent process calling wait() two times?**
  - The parent ensures that both it's children reaches that point, instead of just 1.
- **Do you always get the same order of execution?**
  - No.
- **Does Child 1 always execute (print Child 1 Executing) before Child 2?**
  - No.
- **What do you notice?**
  - Now Child 2 will always execute before Child 1 if Child 2 is successfully created.
- **Can we guarantee that Child 1 always execute before Child 2?**
  - No.

Part 3: Condition Variables

- **After seeing what the two system calls do, why do you think we had to add system calls for the operations on condition variables? Why not just have these operations as functions in ulib.c as we did for spinlock?**

    - Spinlocks can be implemented in `ulib.c` because they’re so easy to perform, just busy-loop until a condition is met. This is not the same for CVs, which require the OS scheduler’s help to put processes to sleep and wake them up.

Part 4: Using the Condition Variables

- **Does Child 1 always execute before Child 2?**
    - Yes.

Part 5: Lost Wakeups

- **Is it always the case that Child 1 executes before Child 2?**
    - Yes.
- **Do you observe deadlocks?**
    - No.