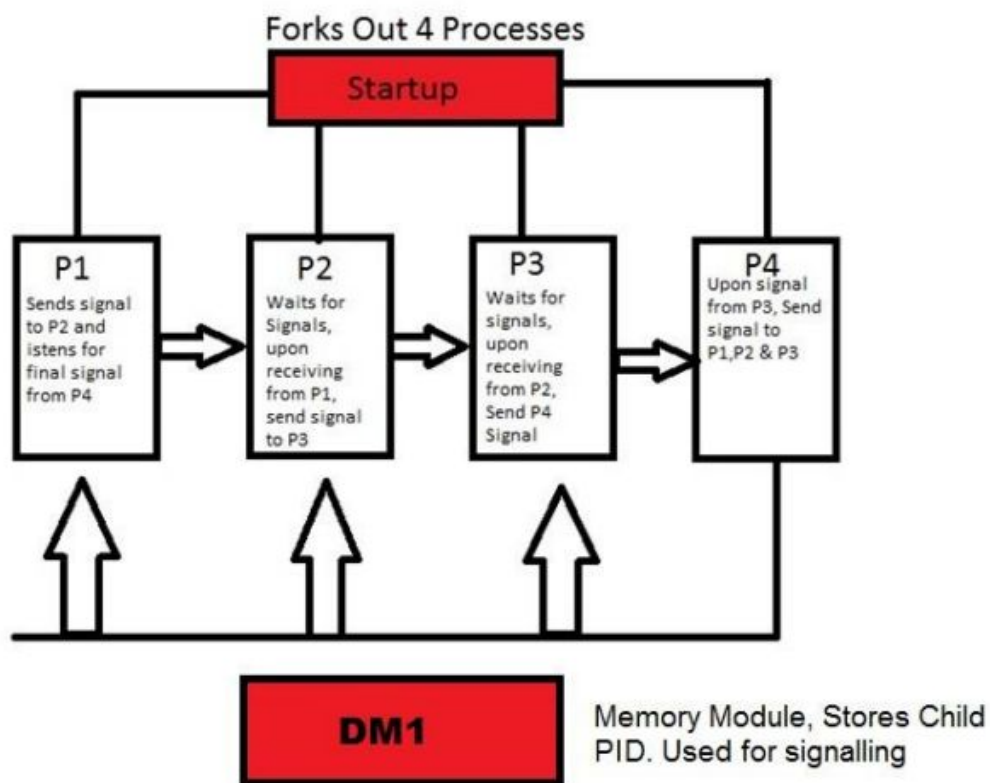


Description of problem:

A startup program forks 4 child processes. Upon forking processes, the process IDs (PID) of the child processes are stored in a data module. P1 starts off by accessing the data module and finding the PID of P2 and then sends it a signal. P2 repeats the process but only upon when receiving a signal from P1. P3 listens for P2's signal and upon receiving, it sends a signal to P4. After P4 receives the signal, it then sends out signals to P1, P2 and P3.

Diagram:



Explanation:

- Startup forks 4 processes via the `_os_exec` & `_os_fork` commands.
- Startup then creates a data module via the `_os_datmod`.
- P1, P2, P3 & P4 all then link to this data module via the `_os_link` command.
- The processes use signal handlers in order to know when to send their signal to the next process.
- Each process has its own delay in order to prevent the process from dying before it can send its signal.

Sample output from test side:

```
[1]$ Startup P2: Signal From P1 received  
P3: Signal From P2 received  
P4: Signal From P3 received  
P1: Signal From P4 received  
P2: Signal From P4 received  
P3: Signal From P4 received
```

Explanation of Output:

Startup forks the processes. P1 then links to the data module and finds P2s PID. It then sends a signal which is handled by P2s signal handler. The process is repeated up to P4 and upon reaching P4s signal handler, P4 sends signals to P1, P2 & P3