DHRUV GARG 16 BCE 1190

INTERPROCESS COMMUNICATION

LAB 4

CODE

```
/* sig_talk.c --- Example of how 2 processes can talk
*/

/* to each other using kill() and signal() */

/* We will fork() 2 process and let the parent send a
few signals to it`s child  */


#include <stdio.h>

#include <signal.h>

#include <stdlib.h>


main()

{ int pid;

  /* get child process */

    if ((pid = fork()) < 0) {

       perror("fork");

       exit(1);

    }


   if (pid == 0)

    { /* child */

      for(;;); /* loop for ever */

    }


  else /* parent */

    {  /* pid hold id of child */

      printf("\nPARENT: sending sleep signal\n\n");

      kill(pid,SIGSTOP);


      printf("\nParent has put the child to sleep for 10
seconds\n\n");

      sleep(10); /* pause for 10 secs */

      printf("\nPARENT: sending wake up signal\n\n");

      kill(pid,SIGCONT);

      printf("\nParent has woken up the child
process\n\n");

    }

}
```

Why are SIGSTOP AND SIGCONT used to put a child process to sleep and wake it up?

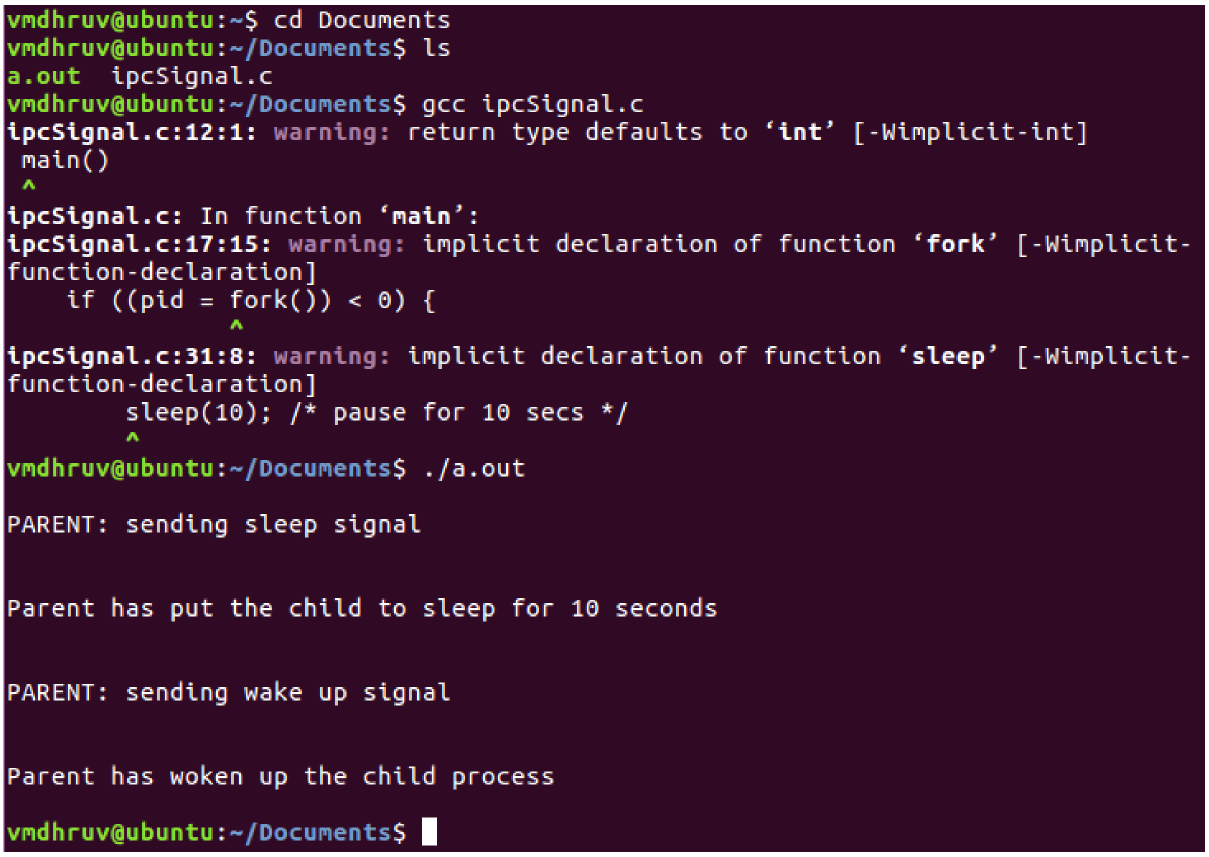
SIGSTOP: When SIGSTOP is sent to a process, the usual behaviour is to pause that process in its current state.

SIGCONT: The paused process will only resume execution if it is sent the SIGCONT signal. SIGCONT tells a process to “pick up where you left off”.

SIGSTOP and SIGCONT are used for job control in the Unix shell, among other purposes.

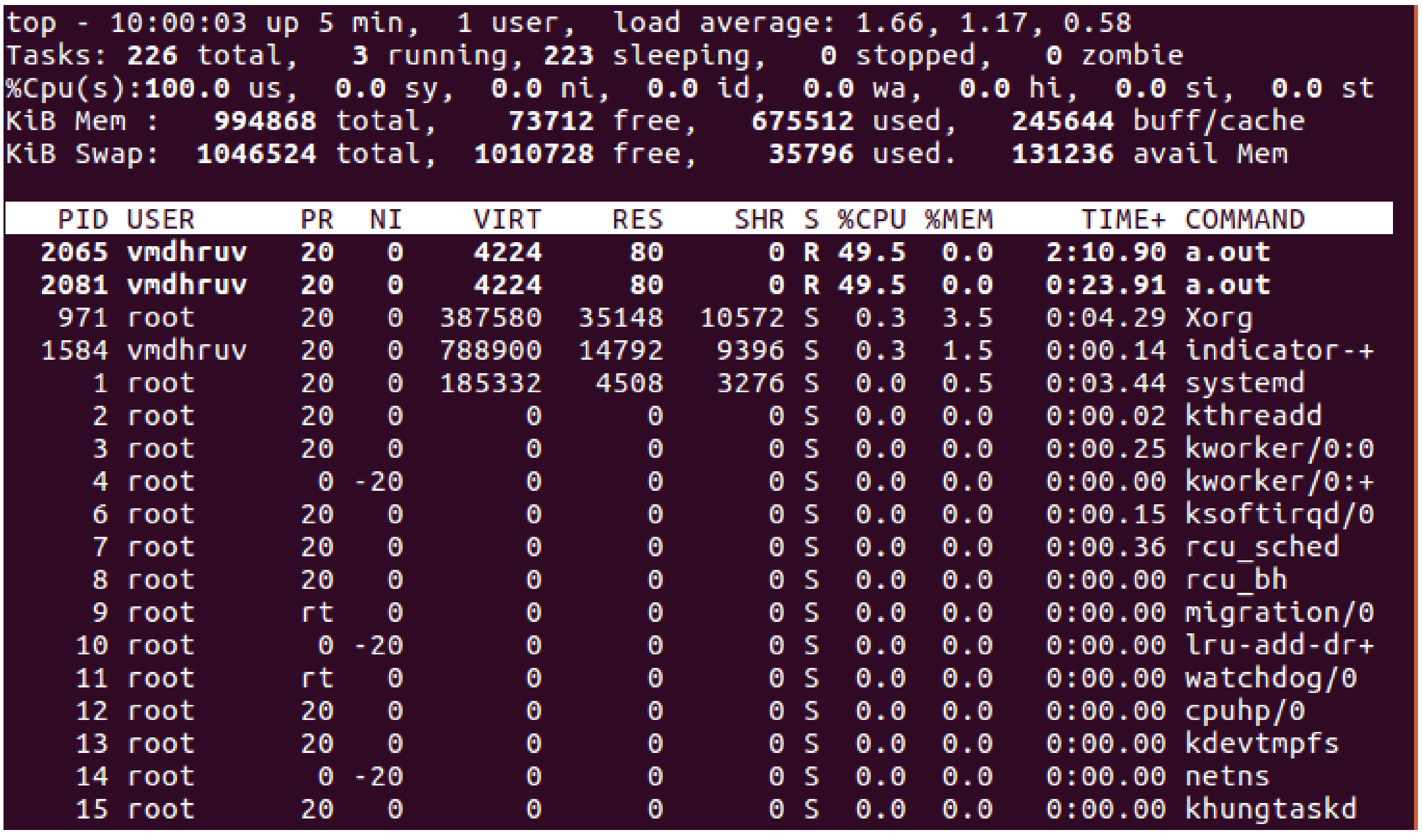
SCREENSHOTS

Running the program. The code executes successfully – creating, putting to sleep and waking up the child process.



```
vmdhruv@ubuntu:~$ cd Documents
vmdhruv@ubuntu:~/Documents$ ls
a.out  ipcSignal.c
vmdhruv@ubuntu:~/Documents$ gcc ipcSignal.c
ipcSignal.c:12:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
 main()
 ^
ipcSignal.c: In function ‘main’:
ipcSignal.c:17:15: warning: implicit declaration of function ‘fork’ [-Wimplicit-
function-declaration]
    if ((pid = fork()) < 0) {
               ^
ipcSignal.c:31:8: warning: implicit declaration of function ‘sleep’ [-Wimplicit-
function-declaration]
        sleep(10); /* pause for 10 secs */
        ^
vmdhruv@ubuntu:~/Documents$ ./a.out

PARENT: sending sleep signal


Parent has put the child to sleep for 10 seconds


PARENT: sending wake up signal


Parent has woken up the child process

vmdhruv@ubuntu:~/Documents$ 
```

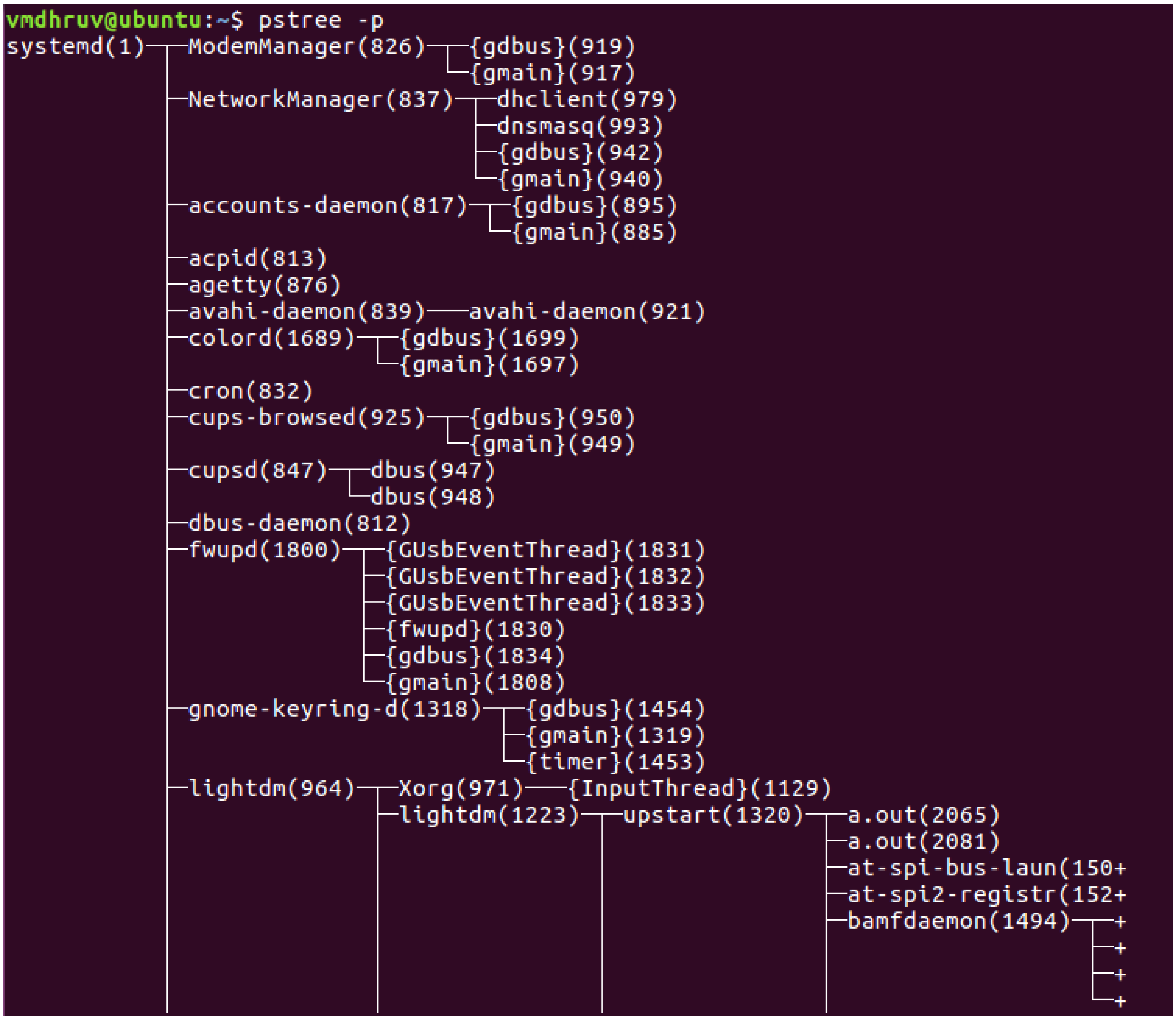
On running the code two times, we can see that there are two “a.out” processes exist in the current running processes table.



```
top - 10:00:03 up 5 min,  1 user,  load average: 1.66, 1.17, 0.58
Tasks: 226 total,   3 running, 223 sleeping,   0 stopped,   0 zombie
%Cpu(s):100.0 us,  0.0 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :   994868 total,    73712 free,   675512 used,   245644 buff/cache
KiB Swap:  1046524 total,  1010728 free,    35796 used.   131236 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 2065 vmdhruv   20   0    4224     80      0 R 49.5  0.0   2:10.90 a.out
 2081 vmdhruv   20   0    4224     80      0 R 49.5  0.0   0:23.91 a.out
  971 root      20   0  387580  35148  10572 S  0.3  3.5   0:04.29 Xorg
 1584 vmdhruv   20   0  788900  14792   9396 S  0.3  1.5   0:00.14 indicator-+
    1 root      20   0  185332   4508   3276 S  0.0  0.5   0:03.44 systemd
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.02 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:00.25 kworker/0:0
    4 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:+
    6 root      20   0       0      0      0 S  0.0  0.0   0:00.15 ksoftirqd/0
    7 root      20   0       0      0      0 S  0.0  0.0   0:00.36 rcu_sched
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
   10 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 lru-add-dr+
   11 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 watchdog/0
   12 root      20   0       0      0      0 S  0.0  0.0   0:00.00 cpuhp/0
   13 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kdevtmpfs
   14 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 netns
   15 root      20   0       0      0      0 S  0.0  0.0   0:00.00 khungtaskd
```

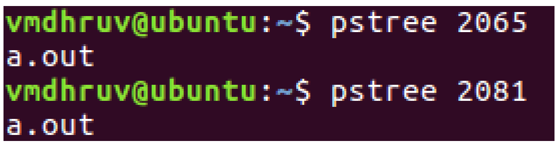
DHRUV GARG 16 BCE 1190

Using "pstree –p" command we can **visualize entire process tree**. We can find the two "a.out" processes running by their unique PIDs.



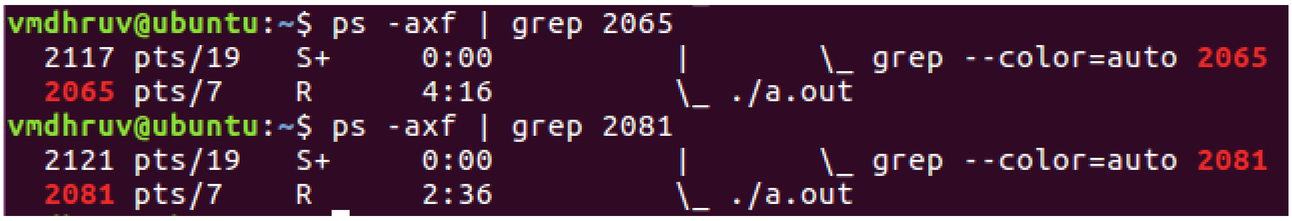
```
vmdhruv@ubuntu:~$ pstree -p
systemd(1)─┬─ModemManager(826)─┬─{gdbus}(919)
           │                   └─{gmain}(917)
           ├─NetworkManager(837)─┬─dhclient(979)
           │                     ├─dnsmasq(993)
           │                     ├─{gdbus}(942)
           │                     └─{gmain}(940)
           ├─accounts-daemon(817)─┬─{gdbus}(895)
           │                      └─{gmain}(885)
           ├─acpid(813)
           ├─agetty(876)
           ├─avahi-daemon(839)───avahi-daemon(921)
           ├─colord(1689)─┬─{gdbus}(1699)
           │              └─{gmain}(1697)
           ├─cron(832)
           ├─cups-browsed(925)─┬─{gdbus}(950)
           │                   └─{gmain}(949)
           ├─cupsd(847)─┬─dbus(947)
           │            └─dbus(948)
           ├─dbus-daemon(812)
           ├─fwupd(1800)─┬─{GUsbEventThread}(1831)
           │             ├─{GUsbEventThread}(1832)
           │             ├─{GUsbEventThread}(1833)
           │             ├─{fwupd}(1830)
           │             ├─{gdbus}(1834)
           │             └─{gmain}(1808)
           ├─gnome-keyring-d(1318)─┬─{gdbus}(1454)
           │                       ├─{gmain}(1319)
           │                       └─{timer}(1453)
           ├─lightdm(964)─┬─Xorg(971)───{InputThread}(1129)
           │              ├─lightdm(1223)─┬─upstart(1320)─┬─a.out(2065)
           │              │               │               ├─a.out(2081)
           │              │               │               ├─at-spi-bus-laun(150+
           │              │               │               ├─at-spi2-registr(152+
           │              │               │               ├─bamfdaemon(1494)─┬─+
           │              │               │               │                  ├─+
           │              │               │               │                  ├─+
           │              │               │               │                  └─+
```

To list the name of the process by the PID



```
vmdhruv@ubuntu:~$ pstree 2065
a.out
vmdhruv@ubuntu:~$ pstree 2081
a.out
```

Listing the child process PIDs and run time. The PIDs 2065 and 2085 represent parent processes, while 2117 and 2121 are child processes. (For some reason, the run time for both child processes is shown as 0:00).



```
vmdhruv@ubuntu:~$ ps -axf | grep 2065
  2117 pts/19   S+     0:00          |       \_ grep --color=auto 2065
  2065 pts/7    R      4:16          \_ ./a.out
vmdhruv@ubuntu:~$ ps -axf | grep 2081
  2121 pts/19   S+     0:00          |       \_ grep --color=auto 2081
  2081 pts/7    R      2:36          \_ ./a.out
```