University of Calgary

CPSC 457: Principles of Operating System, Winter 2018

Assignment 2

For
Coskun Sahin, Dr.Pavol Federl

By
BenKun Chen
30005337
benkun.chen@ucalgary.ca
Tutorial Section: T02

Q1 - Written question (4 marks)

A. DMA(Direct Memory Access is a special piece of hardware on most modern systems provided by DMA controllers, which allows hardware subsystems transfer an entire block of data to access main memory without any CPU intervention. There will be only one interrupt been generated per-block in order to tell the device driver that the operation has succeed.

B. Multiprogramming is rudimentary version of parallel programming, which allows several processes to utilize CPU effectively by making another process running on CPU when CPU is waiting for the I/O of some jobs.

C. The concept of multiprogramming would not be practical on a PC that does not support DMA. The DMA is used for bulk data movement such as disk I/O. Since processor is always busy processing on control flow of bits, other processes will have no chance to utilize the CPU.

Q2 - Written question (3 marks)

A. The sequence of a wrapped system call invokes the actual system call in the kernel is that the wrapper system call first does some formatting and then calls it's system call. The system then puts code for which ever the system call's name in register to be able to trapped by a special instruction in order to let system switches from user mode to kernel mode. Then, the predefined trap handler gets invoked. The handler will ask OS to save application state and do requested operation. After it finishes, the OS then returns to the user mode and then increment SP.

B. It is not essential that a wrapper of a system call is named the same as the underlying system call. The reason is that the wrapper of a system call in different OSes can have different names. Unlike the underlying system call's names, wrapper's name is usually simple and easy to understand and read. A wrapper of a system call can also calls multiple underlying system calls when executing(e.g. read(), printf()). This makes wrapper of a system call's name more general than a underlying system call.

Q3 - Written question (4 marks)

A. It is not possible to go from blocked state to running state. In a blocked state, the process is waiting for some event to occur. After the state gets unblocked, the process will be sent to ready state in order to be executed by the CPU. Eventually, the process scheduler decides to pass the process time slice to running state.

B. It is also not possible to go from ready state to blocked state. When a process is in ready state, the process is ready to be executed by the CPU. Only the CPU can send the blocking request to blocked state, in which, ready state must go from running state to blocked state.

Q4 - Written question (3 marks)

A context switch is the operation which allows the illusion of sharing of a single CPU (or limited number of CPUs) among many processes. It makes the process to store and restore it's states when later resume from a same point. When OS switches between processes A and B, OS saves A's state in A's PCB and restores B's state from B's PCB or vice versa. To explain the actions happened inside the system, the context switch first suspended the progression of one process and stored the CPU's state for that process somewhere in the memory. It then retrieved the context of the next process from memory and restoring it in the CPU's register and returned to the location indicated by the program counter in order to resume the process.

Q5 - Programming question (10 marks)

See file "scan.sh".

Q6 - Programming question (15 marks)

See file "scan.cpp".

Q7 – Written question (3 marks)

Bash script

```
[benkun.chen@zone51-eb Desktop]$ strace -c ./scan.sh jpg 5
./cpsc441/a2/TestFile/files/a.jpg 737026
./img012.jpg 233606
./cpsc231/Lecture/airport1_sketch.jpg 188161
./cpsc231/Lecture/airport1.jpg 96462
Total size: 1255255
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 87.56    0.014599        2920         5         1 wait4
  3.95    0.000659           7        90        52 open
  1.85    0.000308           8        41           mmap
  1.23    0.000205           4        50         6 close
  0.98    0.000163           4        37           fstat
  0.76    0.000127          32         4           clone
  0.73    0.000121          10        12           mprotect
  0.71    0.000119           4        27        10 stat
  0.36    0.000060          10         6         4 execve
  0.28    0.000047           4        11           read
  0.26    0.000044           2        19           rt_sigprocmask
  0.22    0.000037           2        16           rt_sigaction
  0.17    0.000028          14         2           munmap
  0.14    0.000024           4         6         2 access
  0.10    0.000017           2         8           brk
  0.07    0.000012           2         5           getuid
```

```
  0.07    0.000011           4         3           pipe
  0.07    0.000011           2         5           getgid
  0.07    0.000011           2         5           geteuid
  0.06    0.000010           3         3         2 ioctl
  0.06    0.000010           2         5           getegid
  0.05    0.000009           3         3         1 fcntl
  0.05    0.000008           3         3           lseek
  0.04    0.000007           4         2           arch_prctl
  0.03    0.000005           3         2           prlimit64
  0.02    0.000004           4         1           dup2
  0.02    0.000004           2         2           getpid
  0.02    0.000004           4         1           sysinfo
  0.02    0.000003           3         1           rt_sigreturn
  0.02    0.000003           3         1           getpgrp
  0.01    0.000002           2         1           uname
  0.01    0.000002           2         1           getppid
------ ----------- ----------- --------- --------- ----------------
100.00    0.016674                   378        78 total
```

```
[benkun.chen@zone51-eb Desktop]$ time ./scan.sh jpg 5
./cpsc441/a2/TestFile/files/a.jpg 737026
./img012.jpg 233606
./cpsc231/Lecture/airport1_sketch.jpg 188161
./cpsc231/Lecture/airport1.jpg 96462
Total size: 1255255

real    0m0.093s
user    0m0.009s
sys     0m0.029s
```

C++ program

```
[benkun.chen@zone51-eb Desktop]$ strace -c ./a jpg 5
Found 4 files:
        ./cpsc441/a2/TestFile/files/a.jpg : 737026
        ./img012.jpg : 233606
        ./cpsc231/Lecture/airport1_sketch.jpg : 188161
        ./cpsc231/Lecture/airport1.jpg : 96462
Total size: 1255255 bytes.
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 19.12    0.000152           5        28        23 open
 18.87    0.000150          19         8         3 stat
 13.84    0.000110           8        14           mmap
 10.94    0.000087           9        10           mprotect
  7.55    0.000060          60         1           wait4
```

```
  6.79    0.000054           9         6           write
  6.04    0.000048           8         6           read
  4.65    0.000037           5         7           fstat
  4.53    0.000036           5         7           close
  4.40    0.000035          35         1           clone
  1.13    0.000009           9         1           munmap
  0.75    0.000006           2         3           brk
  0.63    0.000005           5         1           pipe2
  0.50    0.000004           4         1           arch_prctl
  0.25    0.000002           2         1           fcntl
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
------ ----------- ----------- --------- --------- ----------------
100.00    0.000795                    97        27 total
```

```
[benkun.chen@zone51-eb Desktop]$ time ./a jpg 5
Found 4 files:
        ./cpsc441/a2/TestFile/files/a.jpg : 737026
        ./img012.jpg : 233606
        ./cpsc231/Lecture/airport1_sketch.jpg : 188161
        ./cpsc231/Lecture/airport1.jpg : 96462
Total size: 1255255 bytes.

real    0m0.072s
user    0m0.010s
sys     0m0.013s
```

Since the C++ program used the C/ C++ function “qsort()”, which greatly boost up the efficiency of the program. As a result, it has less system calls and running time than the bash script.

Q8 – Programming question (10 marks)

See file “sum.cpp”.