University of St Andrews

MARTINMAS 2023-24 EXAMINATION DIET

SCHOOL OF COMPUTER SCIENCE

MODULE CODE: **CS3104**

MODULE TITLE: Operating Systems

EXAM DURATION: 3 hours

EXAM INSTRUCTIONS: (a) Answer all three questions

(b) Each question carries 20 marks

(c) Answer questions in the Script booklet.

YOU MUST HAND IN THIS EXAM PAPER AT THE END OF THE EXAM.

PLEASE DO NOT TURN OVER THIS EXAM PAPER UNTIL YOU ARE INSTRUCTED TO DO SO.

1. Kernel Programming

(a) One role of an operating system is to allocate and share hardware resources, including CPU time. Describe two other operating system roles. [2 marks]

(b) Modern systems like Android and IOS feature application sandboxes. Describe two existing features in Linux that can be used for sandboxing. [2 marks]

(c) (i) Explain, with a brief example, what can go wrong if an operating system never flushes the translation lookaside buffer. [2 marks]

(ii) The APPEP CPU features 16Kib pages and 3-level paging. Every page table has 64 entries. Compute the indices and page offset used when translating the address 0xff123abc. [3 marks]

(iii) Demonstrate the behaviour of the clock page replacement algorithm on the following sequence of page accesses, assuming that there are 3 available frames.

Page number: 1, 2, 3, 2, 1, 2, 5, 1, 2, 3, 4

[3 marks]

(d) (i) The "quantum en-randulator" is a hardware device that performs random coin flips. An excerpt of the hardware documentation is provided in Figure 1. The driver provides a new syscall `flip_coin()` to retrieve a fresh coin flip and an interrupt handler. Complete the code for the driver.

```
struct waitlist waiting_tasks;

/* returns 0/1 for heads/tails */
int flip_coin_syscall() {
        * Your code here *
}
void enrandulator_interrupt() {
        wake_all(&waiting_tasks);
}
```

You may assume the following functions exist

```
struct task *current_task();
void add_task_to_waitlist(struct task *task,
                          struct waitlist *waitlist);
void remove_from_waitlist(struct task *task,
                          struct waitlist *waitlist);
void schedule();
```

Processes are not runnable when they are on a waitlist. More credit will be awarded for solutions that avoid busy-waiting and concurrency issues. State any assumptions that you make about the existing functions. [5 marks]

(ii) A future version of the en-randulator is DMA-enabled and can flip and report a large batch of coins at once. Briefly outline both a new hardware interface and a new syscall interface for the improved device. Explain how it would improve the performance of certain applications. [3 marks]

[Total marks 20]

APPEP CORPORATION QUANTUM EN-RANDULATOR

Section 13.3.7 Device Operation

The Quantum en-randulator uses quantum technology to flip coins and return their face value to the operating system. The device only flips coins when requested. To request N flips the value N should be written to the REQUEST_FLIPS register. The value N will be added to the number of outstanding flips requested by previous writes to REQUEST_FLIPS.

As flip data becomes available the results are added to a FIFO queue. Reading from the FLIP_DATA register will either dequeue a flip from the FIFO (returning 0/1) or return -1 to indicate that the FIFO is empty.

An interrupt is raised after a new flip is added to the FIFO queue.

The REQUEST_FLIPS register is 32-bits wide and has address 0xFFEF0030.
The FLIP_DATA register is 32-bits wide and has address 0xFFEF0040.

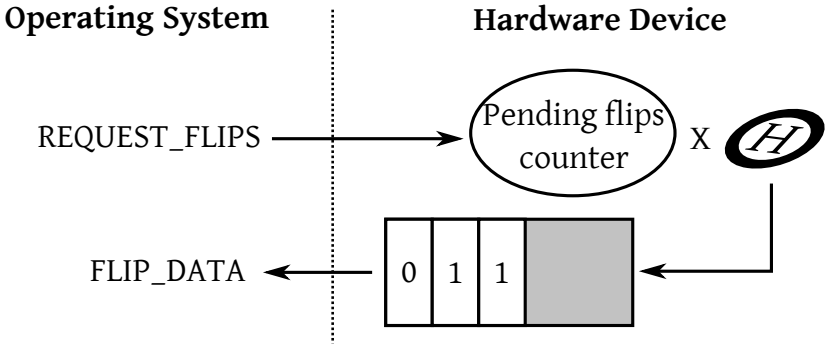

Figure 1: Hardware Documentation

2. Interrupts and File Systems

(a) (i) Hardware devices, the timer and the `int` instruction are all sources of interrupts on x86. State one other interrupt source. [1 mark]

(ii) Explain what happens to the state of an x86 processor when an interrupt occurs. In your answer include the effect on:

- the instruction pointer,
- the stack,
- the registers,
- the CPU mode.

Do not describe the execution of the interrupt service routine. [4 marks]

(iii) The following is an excerpt from an x86 interrupt handler. Explain all the bugs in the implementation.

```
isr:
   popq %rbx
   movq %rbx, last_interrupt_addr

   movq total_interrupts, %rax
   addq $1, %rax
   movq %rax, total_interrupts

   ret
```

[3 marks]

(iv) Explain fully one concurrency problem that can occur when a spinlock is taken both inside and outside of an interrupt handler. How is this issue avoided? [2 marks]

(b) (i) Describe the difference between internal and external fragmentation. [2 marks]

(ii) Give an example of where large block sizes are beneficial. Give an example where they might be a hindrance. [4 marks]

(iii) What is the maximum file size supported by a file system with 16 direct blocks, single, double and triple indirection? Assume a block size of 512 bytes. Disk block numbers are stored in 4 bytes. [4 marks]

[Total marks 20]

3. Scheduling and Concurrency

(a) Explain the difference between a process and a thread. In your answer give an example of each. [2 marks]

Consider a set of processes, P1 to P6, with burst times of 7, 5, 2, 1, 3, 3, respectively. The processes arrive in the order P1, P2, P3, P4, P5, P6 at times 0, 1, 5, 6, 10, 10.

(i) Draw Gantt charts for the execution of these processes using FCFS, SJF and RR (with a quantum of 2) scheduling. [6 marks]

(ii) What is the turn around time for each process for each algorithm? [1 mark]

(iii) Compute the average waiting time for each schedule. Which algorithm gives the minimal average waiting time? [1 mark]

(b) Give an example of a classic race condition. [2 marks]

(c) Explain the difference between progress and bounded waiting in the context of critical sections. Given an example to illustrate your answer. [2 marks]

(d) What is a test and set instruction? How can it be used to implement mutual exclusion? Consider using a fragment of code to illustrate your explanation. [3 marks]

(e) Describe how semaphores work using code. You can assume the existence of a `thread_block()` and a `thread_wakeup()` function. [3 marks]

[Total marks 20]

***** END OF PAPER *****