答題注意事項/Instructions to answer the questions

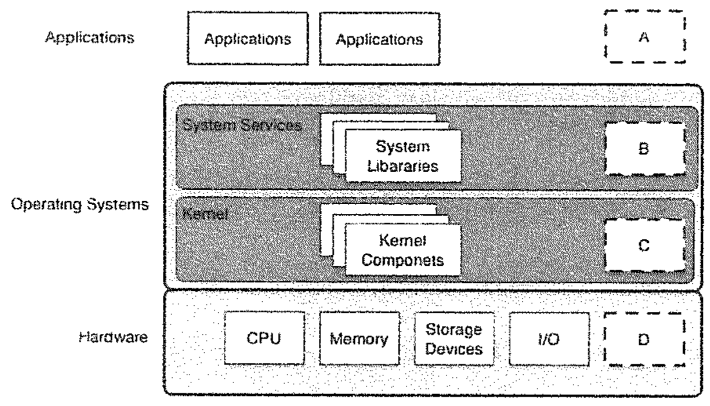
1) 僅答案卷的【非選擇題作答區】第一頁為作答頁，其餘為草稿用，不計分。
Only the first page of Answer Sheet is used to record the final answers and all the other pages are used for sketch.

2) 考生必須將以下表格抄錄至答案卷的【非選擇題作答區】第一頁，不在此一表格的內容一律視為計算過程，不計分。
Before answering the questions, you have to duplicate the following table to Page 1 and only Page 1. The content outside of this table are considered as sketch.

題號	子題 (a)	子題 (b)		子題 (c)		子題 (d)				子題 (e)
1										(此處不作答)
2										(此處不作答)
3										
4										
5										
6				c1	c2	d1		d2		(此處不作答)
										(此處不作答)
7		b1	b2	c1	c2	d1	d2	d3	d4	(此處不作答)
										(此處不作答)
8		(此處不作答)		(此處不作答)		(此處不作答)				(此處不作答)

以上表格必須完整複製於答案卷【非選擇題作答區】第一頁，超出第一頁的部分也視為計算過程。
The above table must be completed duplicated on Page 1 of Answer Sheet. The table not on Page 1 of Answer Sheet is considered as sketch.

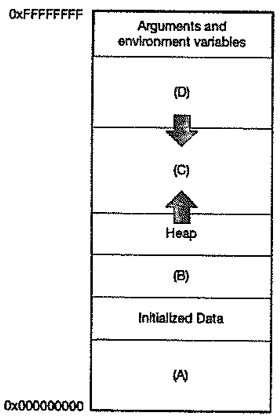
1 (10 pts) Operating systems can be classified into several types according to the architecture of sub-systems. The following figure shows the architecture of operating systems. Four dashed rectangles (marked by A, B, C, D) are located at applications layer, system services layer, kernel layer, and hardware layer.



Please answer the location of the selected sub-systems in difference types of operating systems.
(a) (2 pts) In MS-DOS (monolithic operating systems), where will the memory management services be located?
(b) (3 pts) In Linux (layered operating system), where will the standard I/O libraries are located?
(c) (2 pts) In Container (virtualized operating systems), where will the scheduling services are located?
(d) (3 pts) In Mach/QNX (micro-kernel operating systems), where will the device driver be located?

見背面

2. (10 pts) A running process requires four different types of memory context. Please specify the locations of each context showing in the figure on the right.
   (a) (2 pts) Stack segment
   (b) (3 pts) Text segment
   (c) (2 pts) Uninitialized data segment
   (d) (3 pts) Free space



3. (10 pts) The following table shows the time attributes of five processes in a uni-processor computer, including PID, priority, arrival time, CPU burst, and IO burst. Assuming that the IO burst occurs on different devices and always starts after first time unit. The less the priority value, the higher its priority. Please answer the following sub-questions.

PID	Priority	Arrival Time	CPU Burst	IO Burst
1	2	0	4	0
2	1 (Highest)	2	5	3
3	3 (Lowest)	3	3	0
4	3	4	5	3
5	1	10	4	2

(a) (2 pts) When round-robin scheduling algorithm is used and the quantum size is 3, what's the completion time of PID 5 process?

(b) (2 pts) When non-preemptive priority scheduling algorithm is used, what's the completion time of PID 5 process?

(c) (2 pts) When preemptive SJF scheduling algorithm is used, what's the completion time of PID 5 process?

(d) (2 pts) When preemptive LISC scheduling algorithm is used and PID 5 process arrives at time 16, what's the completion time of PID 5 process?

(e) (2 pts) When preemptive LIF scheduling algorithm is used and the CPU burst of PID 5 process becomes 8, what's the completion time of PID 5 process?

4. (10 pts) Suppose that a process needs to read and write a large file from the storage sub-system. Please answer the **five** sub-questions using one of three values: (**A**) – less than, (**B**) – greater than, and (**C**) – similar to.

Assume blocking I/O is adopted as the I/O model, the file to be stored has very large size, and the delay of system load can be ignored here. The file system has 4K-byte data blocks.

The following table shows the setting of the process and the sub-questions to be answered. The first column defines the buffering model, the second column defines the parameters of the buffering model, and the third column shows the

接 次 頁

function calls used to store the data. The first line of below table stands for the baseline, which considers the combination of unbuffered I/O, 4K buffer size, and no disk synchronization, to be compared to.

Setting			**Sub-Questions**
Unbuffered IO	Buf Size 4KB	write()	(Baseline to be compared to)
Unbuffered IO	Buf Size 4KB	fsync() after write ()	**(a) (2 pts) How is the Clock time compared with that of baseline?**
Unbuffered IO	Buf Size 8KB	fsync () after write ()	**(b) (2 pts) How is the User CPU time compared with that of baseline?**
Buffered IO	line-at-a-time	puts()	**(c) (2 pts) How is the System CPU time compared with that of baseline?**
Buffered IO	fully buffer	puts()	**(d) (2 pts) How is the User CPU time compared with that of baseline?**
Buffered IO	fully buffer	puts()+fflush()+fsync ()	**(e) (2 pts) How is the System CPU time compared with that of baseline?**

5. (10 pts) Please read the code segment and answer the **five** sub-questions.
   (a) (2 pts) Please answer the line number where the process will be suspended when it first starts?
   (b) (2 pts) When the process is suspended and a SIGUSR2 signal is sent to the process, how does the process react? (**A**): terminated, (**B**): resume, (**C**): no action, or (**D**) Add SIGUSR2 to the signal mask.
   (c) (2 pts) What's the signal mask when SIGUSR2 is caught by the process: (**A**) SIGUSR1and SIGUSR2, (**B**) SIGUSR1, (**C**) SIGINT and SIGUSR2, and (**D**) Empty.
   (d) (2 pts) When SIGINT is sent and caught, from which line will the process be resumed?
   (e) (2 pts) When sigaction() on line 17, 19, and 21 are replaced by signal(), what's the signal mask after resumed by catching SIGINT? (**A**) SIGUSR1 and SIGUSR2, (**B**) SIGUSR1, (**C**) SIGINT and SIGUSR2, and (**D**) Empty.

```
1 %% include all the head files
2
3 static void sigHandling(int);
4
5 int
6 main(void)
7 {
8       sigset_t   newmask, oldmask, waitmask;
9       struct sigaction    handling, saveintr;
10
11      pr_mask("program start: ");
12
13      handling.sa_handler = sigHandling;
14      sigemptyset(&handling.sa_mask);
15      handling.sa_flags = 0;
16
17      if (sigaction(SIGUSR1, &handling, &saveintr) < 0)
18         return(-1);
19      if (sigaction(SIGUSR2, &handling, &saveintr) < 0)
20         return(-1);
21      if (sigaction(SIGINT, &handling, &saveintr) < 0)
22         return(-1);
23
24      sigemptyset(&waitmask);
25      sigaddset(&waitmask, SIGUSR1);
26      sigemptyset(&newmask);
27      sigaddset(&newmask, SIGINT);
28
29      /*
30       * Block SIGINT and save current signal mask.
```

見背面

```
     */
    if (sigprocmask(SIG_BLOCK, &newmask, &oldmask) < 0)
        err_sys("SIG_BLOCK error");

    /*
     * Critical region of code.
     */
    pr_mask("in critical region: ");

    /*
     * Pause, allowing all signals except SIGUSR1.
     */
    if (sigsuspend(&waitmask) != -1)
        err_sys("sigsuspend error");

    pr_mask("after return from sigsuspend: ");

    /*
     * Reset signal mask which unblocks SIGINT.
     */
    if (sigprocmask(SIG_SETMASK, &oldmask, NULL) < 0)
        err_sys("SIG_SETMASK error");

    /*
     * And continue processing ...
     */
    pr_mask("program exit: ");

    exit(0);
}

static void
sigHandling(int signo)
{
    pr_mask("\n in sigHandling:");
    printf("Received signal %d\n", signo);
}
```

6. (20 pts) Consider a SMP processor (Symmetric Multiprocessor) adopting the Write-Invalidate *Snooping Cache Coherence* protocol. Each processor has a 4KiB direct-mapped, physically-addressed L1 cache with 16-byte cache lines. Three integer arrays A, B and C are placed contiguously in the memory in the order of A, B and C. The starting address of array A is 0x0000A000 (assuming 32-bit memory address). Processor P0 and P1 execute the following code segment (the L1 cache is empty initially):

```
Int  A[300], B[300], C[300]
for (i=4*Pn; i< 4*(Pn+1);i++)    /* Pn is the processor id: 0, 1 */
    C[i] = A[i]+B[i];
```

(a) (5 pts) What is the size (bits) of the tag array?
(b) (5 pts) Could increasing the cache associativity reduce cache misses running the above codes?
(c) (5 pts) How many coherence misses could occur in the worst case(c1)? How many of them are false-sharing misses(c2)?
(d) (5 pts) if the cache line size becomes 32 bytes (the cache size is still 4KiB), how many coherence misses could occur in the worst case (d1)? How many of them are false-sharing misses(d2)?

7. (25 pts) Consider a 5-stage (IF, ID, EX, MEM, WB) RISC-V pipelined processor (Figure 1) composed of logic blocks with the following latency (any unspecified block latency is treated as 0):

接次頁

I-Mem / D-Mem R/W	Register File R/W	Mux	ALU	Adder / Shifter/Comparator	PC R/W	Single gate	Imm Gen	Control/ ALU Control	Pipeline Register R/W	Forwarding Unit/Hazard Detection Unit
280 ps	180 ps	25 ps	200 ps	50 ps	30 ps	5 ps	50 ps	80 ps	30 ps	50 ps

31 funct7	rs2	rs1	funct3	rd	opcode 0
7 bits	5 bits	5 bits	3 bits	5 bits	7 bits

R-type instruction format (add/sub rd, rs1, rs2)

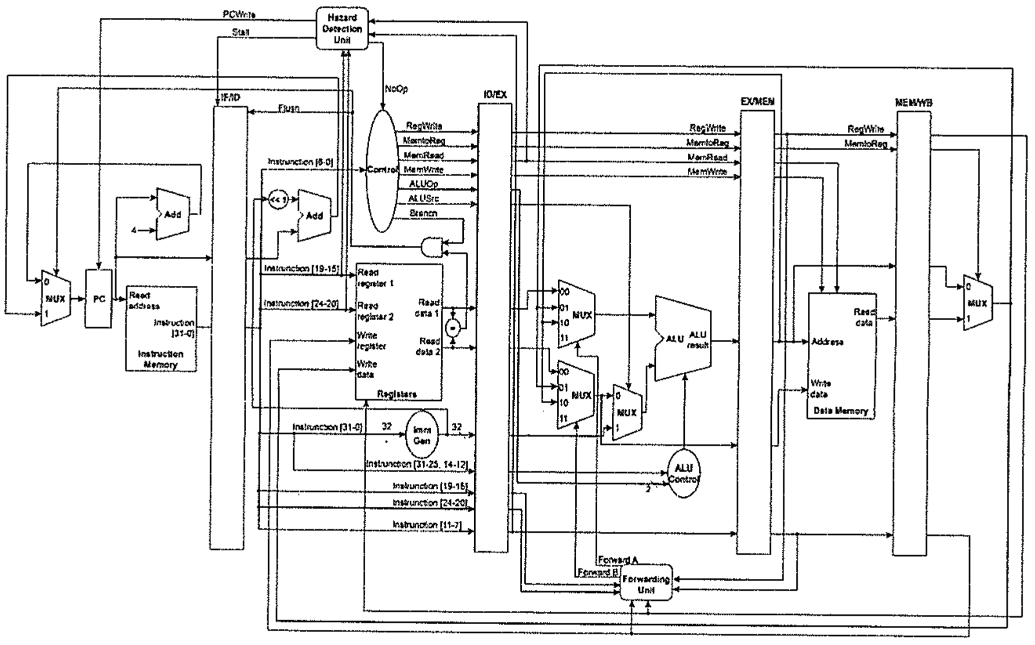


Figure 1

Given the following code segment:

```
#1 lw     x10, 100(x5)
#2 add    x2, x10, x2
#3 add    x2, x2, x4
#4 sub    x6, x2, x6
```

(a) (10 pts) Which pipeline stage determines the clock cycle time?

(b) (5 pts) When the first instruction (lw) is in the MEM stage, which instructions are executing in the ID (b1) and EX (b2) stage? (請填寫 instruction number, 不要寫 instruction)

(c) (5 pts) When the sub instruction (#4) is in the EX state, what value should be set for the control signal ForwardA (c1) and ForwardB (c2), respectively?

(d) (5 pts) What are the control signal values for RegWrite (d1), MemtoReg(d2), MemRead(d3), and MemWrite (d4) for the sub instruction?

8. (5 pts) Which one(s) of the following statements are correct? (全對才給分)

(1) A processor with higher MIPS (Million Instruction per Second) has higher performance than a processor with lower MIPS.

(2) The LRU (Least Recently Used) cache replacement policy has been proved to be the optimal one.

見背面

(3) Vector instructions could be beneficial to applications with good data-level parallelism.
(4) A SMT (Simultaneous Multithreading) processor can increase CPU utilization by allowing instructions from several independent threads to execute each cycle.