國立臺北科技大學101學年度碩士班招生考試

系所組別：2300 資訊工程系碩士班

第一節　作業系統與計算機組織　試題

注意事項：

1. 本試題共 10 題，配分共 100 分。
2. 請標明大題、子題編號作答，不必抄題。
3. 全部答案均須在答案卷之答案欄內作答，否則不予計分。

一、 (12%) For the following questions regarding *process management*, please indicate whether each statement is true or false. If a statement is incorrect, please explain the reasons.

1. The primary distinction among short-term, medium-term, and long-term schedulers lies in frequency of execution. (2%)
2. Shortest job first (SJF) scheduling algorithm will not result in starvation. (2%)
3. First-come, first-served (FCFS) scheduling algorithm always favors short processes. (2%)
4. Multithreaded programs can always provide better performance than a single-threaded solution. (2%)
5. A multithreaded program using multiple user-level threads achieves better performance on a multiprocessor system than on a single-processor system. (2%)
6. A multithreaded program using multiple kernel threads can provide better performance than a single-threaded program on a single-processor system. (2%)

二、(8%) Among the following statements about *memory management*, please indicate whether each statement is true or false. If a statement is incorrect, please explain the reasons.

1. Pure segmentation has the problem of internal fragmentation. (2%)
2. Pure segmentation requires more memory overhead than pure paging to maintain the address translation structures. (2%)
3. Pure paging avoids the problem of external fragmentation. (2%)
4. Both pure paging and pure segmentation have the ability to share code across processes. (2%)

三、(6%) Describe the differences among the layered, modular, and microkernel approaches to operating system structures. Please compare their advantages and disadvantages.

四、(6%) Assume that we have a demand-paged memory. The page table is held in registers. It takes 8 milliseconds to service a page fault if an empty page is available or if the replaced page is not modified and 20 milliseconds if the replaced page is modified. Memory access time is 100 nanoseconds.

Assume that the page to be replaced is modified 70 percent of the time. What is the maximum acceptable page-fault rate for an effective access time of no more than 200 nanoseconds?

五、(8%) Regarding the following descriptions about *file management*, please indicate whether each statement is true or false. If a statement is incorrect, please correct it.

1. The inode in UNIX is a variation of the indexed allocation method. (2%)
2. The contiguous allocation algorithm suffers from the problems of internal fragmentation and size declaration. (2%)
3. One of the advantages of the journaling file systems is that if the file system crashes, all remaining transactions in the log do not need to be performed again. (2%)
4. The benefit of the variant of linked allocation that uses a file-allocation table (FAT) to chain together the blocks of a file is that: random-access time is improved. (2%)

六、(10%) Answer the following questions regarding *process coordination*.

1. What is the meaning of the term *busy waiting*? Can busy waiting be avoided altogether? Why spinlocks are not appropriate for single-processor systems yet are often used in multiprocessor systems. Please explain your answers. (6%)

2. Please describe the possible issues if we want to provide synchronization mechanisms in a *distributed system*. (4%)

注意：背面尚有試題

七、(16%) The following C code segments in Figure 1 compute matrix multiplications of two matrices A and B, where A is a 4×183 row matrix, and B is a 183×1 matrix.

c-1	for (i = 0; i < 183; i++)
	  for (j = 0; j < 4; j++)
	  C[j] += A[j][i] * B[i];

c-2	for (j = 0; j < 4; j++)
	  for (i = 0; i < 183; i++)
	  C[j] += A[j][i] * B[i];

Figure 1. The C code segments for matrix multiplication

1. For the **C codes statements** above, please translate these two C code segments into corresponding **MIPS assembly code**s. Assume that the variable **i**, **j** are assigned to registers $s0 and $s1, respectively, and the base address of the array **A, B, C** are in registers $s2, $s3, and $s4, respectively. Assume that all the values in the above variables and arrays have already defined and initialized. (8%)

2. What are the accuracies (in percentage) of "**always-not-taken**" predictors for the **outer** and **inner** loops in the above two code segments, respectively? (4%)

3. What are the accuracies of the **two-bit predictor** in the following Figure 2 for the branches for the **outer** and **inner** loops in the above two code segments, respectively? Assume that the predictor starts off in the **top-right state** in Figure 2 (predict taken). (4%)

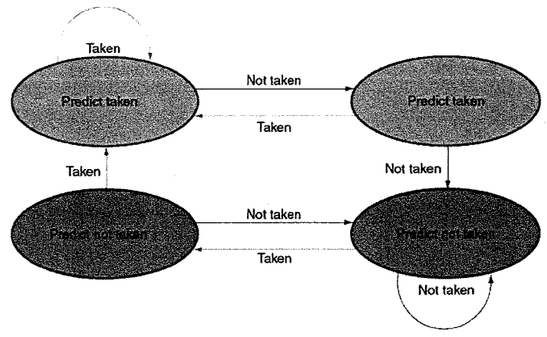

Figure 2. The two-bit predictor

八、(20%) For a typical MIPS processor with a five-stage pipeline, including: (1).**IF**-Instruction fetch; (2).**ID**-Instruction decode and register fetch; (3).**EX**-Execution or calculate effective address; (4).**MEM**-Access data memory; and (5).**WB**-Write back to registers. Assume that the register write is done in the first-half of a clock cycle, and the register reads can be done in the second-half of the cycle. Then, the following code segment will be executed using this pipeline.

i1: lw $1, 10($2)
i2: sub $3, $2, $1
i3: add $4, $8, $1
i4: and $1, $6, $5
i5: beq $5, $1, *L1*
i6: add $6, $7, $5
i7: sub $5, $1, $6
L1: lw $3, 12($5)

where *in* indicates the *n*th instruction in this code segment.

1. Please find all data dependencies in the above code segment in terms of *in.* (3%)

2. Identify all types of hazards in the above code segment in terms of *in.* (3%)

3. If we stall the pipeline when a data hazard happens (**no forwarding**), how many cycles will it take to complete the above code segment, and how many cycles are we stalling due to data hazards? Please complete the resulting pipeline and illustrate the pipeline execution diagram. (5%)

4. Suppose the above MIPS code segment can be now executed using the logic of the data hazard detection and forwarding unit depicted in Figure 3. Please determine whether the above MIPS instructions are correctly executed? If so, please give your explanations. If not so, how can we revise and correct the logic in Figure 1 and so that the codes can be correctly executed? (4%)

```
If   (MEM/WB.RegWrite
       and (MEM/WB.RegisterRd ≠ 0 )
       and (MEM/WB.RegisterRd = ID/EX.RegisterRs ) )
Then
       ForwardA = 01

If   (MEM/WB.RegWrite
       and (MEM/WB.RegisterRd ≠ 0 )
       and (MEM/WB.RegisterRd = ID/EX.RegisterRt ) )
Then
       ForwardB = 01
```

Figure 3. The logic of the data hazard detection and forwarding unit

5. If we now have suitable data hazard detection and forwarding units to perform full forwarding (i.e. forward all results that can be forwarded), how many cycles will it take to complete the above code segment, and how many cycles are we still need stalling due to data hazards? Please complete the resulting pipeline and illustrate the pipeline execution diagram. (5%)

九、(5%) Given the following hardware components in Figure 4, including one 6-bit register, one 6-bit ALU, one 12-bit shift register, and a control unit. Thus, given the hardware components in Figure 4, can you design a combinative hardware for both multiplication and division computations on the above-mentioned integers? Please sketch your design (hints: Multiplication needs Multiplicand, Multiplier, and Product; Division needs Dividend, Divisor, Quotient, and Remainder).

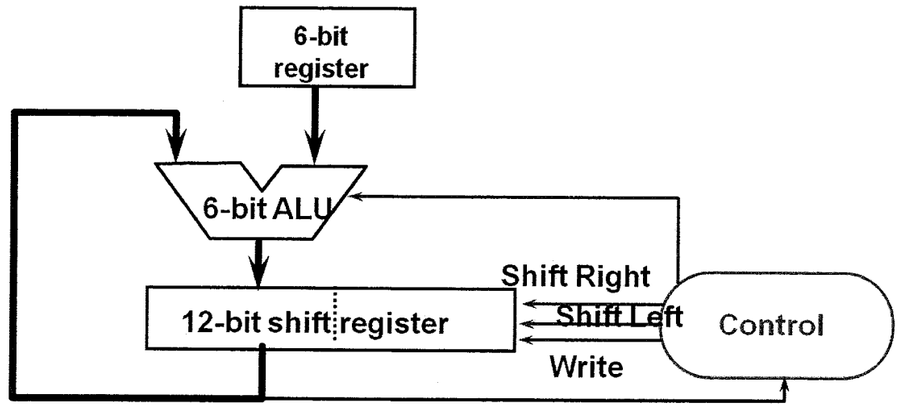

Figure 4. The hardware components to perform both multiplication and division computations

十、(9%) Suppose we have one set associative cache below, and which comprises of **32 words**. Below is a sequence of memory address references as word addresses:

Address Sequence	0, 55, 66, 18, 48, 101, 71,183 ,64, 212, 247, 208, 80, 96, 87

Please indicate each reference as a hit or a miss for the following associative cache. Assuming that LRU (Least Recently Used) is used for each replacement algorithm and all the caches are initially empty.

1. This cache is a **four-way set associative cache** with **two-word blocks**, please indicate each reference as a hit or a miss, and determine its miss rate. (5%)
2. For the above associative cache, how many misses are **conflict misses**? (2%)
3. For each of above associative cache, how many misses are **compulsory misses**? (2%)