考試科目	計算機系統 (計算機組織與結構)	所別	資訊科學	考試時間	4 月 21 日 上 [✓ 下] 午 第 1 節 星期 日

一、Computer Organization and Design

1. **[Overflow Detection]** (a) Assume that A and B are 32-bit integers. It is trivial to show that overflow can not occur when adding operands with different signs. Therefore, we only need to consider the following four conditions. Fill in the blanks to indicate when overflow actually occurs. (2%)

Operation	Operand A	Operand B	Result	Overflow? (Yes or No)
A+B	>=0	>=0	>=0	
A+B	>=0	>=0	< 0	
A+B	< 0	< 0	>=0	
A+B	< 0	< 0	< 0	

(b) Repeat part (a) for subtraction. What conditions need to be analyzed for subtraction? Why? (4%)

(c) A simple check for overflow during addition is to see if the CarryIn to the most significant bit is not the same at the CarryOut of the most significant bit. Prove that this check is the same as the results shown in (a) and (b) by building a table that shows all the possible combinations of Sign (of A, B, and Result) and CarryIn to the sign bit position and derive the CarryOut, Overflow and related information.(4%)

2. **[Computer Arithmetic]** An elegant approach to multiplying signed numbers is called *Booth's algorithm*.

(a) Demonstrate *Booth's algorithm* using the example: 3x-2 represented by 4-bit numbers. List all the steps involved and the intermediate values of the product. (6%)

(b) To compute *axb* where *a* is the multiplier and *b* is the multiplicand (both are 32-bits), one can recast Booth's algorithm in terms of the bit value of the multiplier (a_i refers to the i-th bit of a):

a_i	a_{i-1}	Operation
0	0	Do nothing
0	1	Add b
1	0	Subtract b
1	1	Do nothing

With the above table and the fact that shifting the multiplicand left respect to the Product register is equivalent to multiplying by a power of 2, show that Booth's algorithm does in fact perform 2's complement multiplication of a and b. (4%)

3. **[Cache]** Suppose a computer's address size is *k* bits (using byte addressing), the cache size is *S* bytes, the block size is *B* bytes, and the cache is *A*-way set associative. Assume

考 試 科 目	計算機系統(計算機組織與結構)	所 別	資訊科學	考 試 時 間	4 月 21 日上午第 1 節 星期 日 (下)

that B is a power of two, so $B=2^b$. Figure out what the following quantities are in terms of S, B, A, b and k:

(a) the number of sets in the cache (3%)

(b) the number of index bits in the address (3%)

(c) the number of bits needed to implement the cache. (4%)

4. **[Integrated Circuit Cost]** The cost of an integrated circuit can be expressed through three simple equations:

   Cost per die = Cost per wafer/(Dies per wafer x yield)

   Dies per wafer = Wafer area/Die area

   Yield $=1/(1+\text{Defects per area} \times \text{Die area}/\alpha)^{\alpha}$

   (a) What is the yield assuming $0.8/cm^2$ defect density and a die area of 90 mm^2 for $\alpha=2$? (4%)

   (b) What is the approximate relationship between cost and die area? (Hint: Cost = f((Die area)^x) for some x. Determine x in terms of α) (4%)

   (c) What implication does the above relationship have for designers (assume that $\alpha>=2$)? (2%)

5. **[Disk Array I/O]** Assume that we have the following two magnetic disk configurations: a single disk and an array of 4 disks.

   i. Each disk has 64 sectors per track, each sector holds 1000 bytes, and the disk revolves at 10000 rpm.

   ii. Seek time is 6ms.

   iii. The delay of the disk controller is 1 ms per transaction, either for a single disk or for the array.

   Assume that the performance of the I/O system is limited only by the disks and the controller. Remember that the consecutive sectors on a single disk will be spread on one sector per disk in the array.

   (a) How much time does each I/O operation take for the single disk if the workload consists of 4 KB reads from sequential sectors? (2%)

   (b) How much time does each I/O operation take for the disk array if the workload consists of 4 KB reads from sequential sectors? (2%)

   (c) Compare the I/O performance per second of these two disk organizations, assuming that the requests are random reads, half of which are 4 KB and half of which are 16 KB of data from sequential sectors. (6%)

   (For simplicity, assume that the sectors may be read in any order. The rotational latency is one-half the revolution time for the single disk read of 16 sectors and the disk array read of 4 sectors.)

備 考	試 題 隨 卷 繳 交

考試科目	計算机系統(作業系統)	所別	資訊科學系	考試時間	4月21日(六) 星期日 [✓ 下]午第1節

二. There are five problems in this computer systems: operating systems examination. Please do all of them. The weights for each problem is indicated after the problem number.

1. (a) (3 points) What resources are used when a thread is created? How do they differ from those used when a process is created?

   (b) (3 points) What are two differences between user-level threads and kernel-level threads?

   (c) (4 points) Describe the actions taken by a kernel to context switch between kernel-level threads?

2. The *dining-philosophers* problem is considered to be a classic synchronization problem. It is a simple representation of the need to allocate several resources among several processes in a deadlock- and starvation-free manner. One simple solution is to represent each chopstick by a semaphore. A philosopher tries to grab the chopstick by executing **wait** operation on that semaphore; she (or he) releases her chopsticks by executing the **signal** operation on the appropriate semaphores. Thus, the shared data are

```
semaphore chopstick[5];
```

where all the elements of **chopstick** are initialized to 1. The structure of philosopher *i* is shown as the following:

```
do{
   wait(chopstick[i]);
   wait(chopstick[i+1] % 5);
     .....
     eat
     .....
   signal(chopstick[i]);
   signal(chopstick[(i+1) % 5]);
     .....
     think
     .....
   } while(1);
```

(a) (3 points) Why we reject the above program codes to solve the *dining-philosophers* problem?

(b) (7 points) Please show a correct program structure to solve the *dining-philosophers* problem using high-level synchronization construct **monitor**?

備考	試題隨卷繳交

考試科目	計算機系統(作業系統)	所別	資訊科學系	考試時間	4 月 21 日上 [✓ 下] 午第 1 節 星期 日

3. A page-replacement algorithm should minimize the number of page faults. We can do this minimization by distributing heavily used pages evenly over all of memory, rather than having them compete for a small number of page frames. We can associate with each page frame a counter of the number of pages that are associated with that frame. Then, to replace a page, we search for the page frame with the smallest counter. Define a page-replacement algorithm using this basic idea. Specifically address the problems of:

   i. what the initial value of the counters is,
   ii. when counters are increased,
   iii. when counters are decreased,
   iv. how the page to be replaced is selected.

   (a) (5 points) How many page faults occur for your algorithm for the following reference string, for five page frames?

   $$1, 2, 3, 4, 5, 3, 4, 1, 6, 7, 8, 7, 8, 9, 7, 8, 9, 5, 4, 5, 4, 2$$

   (b) (5 points) What is the minimum number of page faults for an optimal page-replacement strategy for the reference string in part (a) with five page frames? Is it the page faults number the same as (a)'s, please comments on (a) and (b)'s results?

4. (a) (5 points) Consider a system that supports the strategies of contiguous, linked, and indexed allocation. What criteria should be used in deciding which strategy is best utilized for a particular file?

   (b) (5 points) A UNIX i-node has 10 disk addresses for data blocks, as well as the addresses of single, double, and triple indirect blocks. If each of these holds 512 disk addresses, what is the size of the largest file that can be handled, assuming that a disk block is 2K?

5. Consider the operating systems that support three domains ($user_1$, .., $user_3$) and three objects: file F_1, file F_2, file F_3. $user_1$ is the owner of file F_1, $user_2$ is the owner of file F_2, $user_3$ is the owner of file F_3. Furthermore, the operating systems allow $user_1$ to do the domain **switch** to $user_2$ and $user_2$ to do the domain **switch** to $user_3$. The *control* right in $access(user_2, user_3)$ was also included in the access matrix.

   (a) (5 points) How would you denote this protection scheme in access matrix?

   (b) (5 points) How would you implement of (a)'s access matrix in access control list(ACL)?

備考	試題隨卷繳交