所別：資訊工程學系碩士班 科目：作業系統與計算機組織
軟體工程研究所碩士班

1. There is an unpipelined processor that has a 1 ns clock cycle and that uses 4 cycles for ALU operations and 5 cycles for memory operations. Assume that the relative frequencies of these operations are 40%, 20%, and 40%, respectively. Suppose that due to clock skew and setup, pipelining the processor adds 0.2 ns of overhead to the clock. Ignoring any latency impact, how much speedup in the instruction execution rate will we gain from a pipeline implementation?(10 pts)

2. A computer system has L1 and L2 caches. The local hit rates for L1 and L2 are 95% and 80%, respectively. The miss penalties are 8 and 60 cycles, respectively. (a) Assume a CPI(Cycles per Instruction) of 1.2 without any cache miss and an average of 1.1 memory accesses per instruction, what is effective CPI after cache misses are factored in? (10 pts)(b) Taking the two levels of caches as a single cache memory, what are its miss rate and miss penalty?(5 pts)

3. [13 pts]Engineers in your company developed two different hardware implementations, M1 and M2, of the same instruction set, which has three classes of instructions: I (Integer arithmetic), F (Floating-point arithmetic), and N (Non-arithmetic). M1's clock rate is 1.2GHz and M2's clock cycle time is 1ns. The average CPI for the three instruction classes on M1 and M2 are shown below:

Class	CPI for M1	CPI for M2
I	3.2	3.8
F	5.6	4.2
N	2.4	2.0

Please answer the following questions:

a. What are the peak performances of M1 and M2 in MIPS? (2pt.)

b. If 50% of all instructions executed in a program are from class N and the rest are divided equally among F and I, which machine is faster and by what factor? (2pt.)

c. The designers of M1 plan to redesign the machine to improve its performance. With the instruction mix given in question b, please evaluate each of the following 4 redesign options and rank them according to their performance improvement. (5pts.)

   1. Using a faster floating-point unit which doubles the speed of floating-point arithmetic execution.
   2. Adding a second integer ALU to reduce the integer CPI to 1.6.
   3. Using faster logic that allows a clock rate of 1.5GHz with the same CPIs.
   4. The CPIs given in the table include the effect of instruction cache misses at an average rate of 5%. Each cache miss adds 10 cycles to the effective CPI of the instruction causing the miss. A new redesign option is to use a larger instruction cache that would reduce the miss rate from 5% to 2%.

d. If you prefer the M2 implementation and would like to work out test programs that run faster on M2 than on M1. Let x and y be the fraction of instructions belonging to class I and F respectively. What kind of relationship between x and y will you maintain? (4pts.)

注意：背面有試題

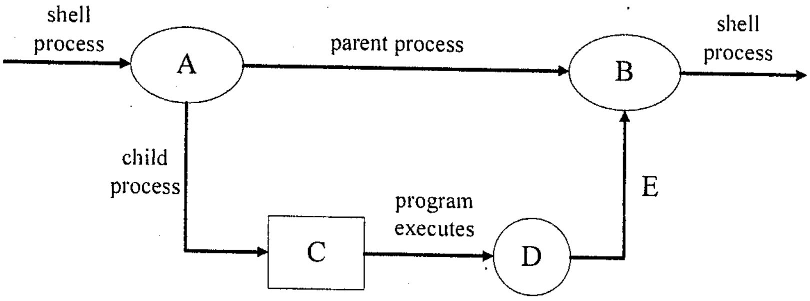
4. [12 pts] Please answer the following questions about memory hierarchy.

a. Please write short C codes to demonstrate the locality of memory access. (3 pts.)

b. What are TLB and page table? Please describe clearly and systematically how a memory access is completed by the processor cache/main memory/TLB/page table/hard disk. (6pts.)

c. A computer system has a cache memory with 128K bytes. The 32-bit memory address format is as follows: Tag bits: 31~15, Index (or Set) bits: 14~4, Offset bits:3~0. Please derive the number of degrees of set associativity in this cache. (3pts.)

5.(10pts) The following figure shows the procedure of process control system calls for UNIX systems. What are A, B, C, D and E in the figure?



6.(10pts) Please give the advantages of implementing kernel-based threads and user-mode threads, respectively.

(後面還有題目)

注意：背面有試題

所別：資訊工程學系碩士班 科目：作業系統與計算機組織
軟體工程研究所碩士班

7.(10pts)

a.Describing what are spinlocks ?
Explain why spinlocks are not appropriate for uniprocessor systems yet may be sutible for multiprocessor systems ?

b.Describing what are dispatcher objects and the states of them, to explain it with the state chaging in a mutex lock example.

8.(20pts)

A program is showed as follow, Please answer the questions:

a. What happen after //Line--101 &
what is the meaning after //Line--103 , wait

b. After compiling and execution with different loop count value as showed in //case--1 , //case--2 , //case--3 , Describing why the results are different in each case ?

d. The global variable "value" is shared Read/Write by parent and child processes or not ? What means of the term copy-on-write ?

e. If you want to let the "value" be shared by thread runner0 and runner1 correctly in each process , what need to do ? If it is need to shared in both parent and child thread processes to update "value" correctly in any delay loop count , what need to do ?

```
bash-2.03$ cat fork-thread.c
#include <pthread.h>
#include <stdio.h>
int value = 0;
void  *runner0(void *param);
void  *runner1(void *param);

int main(int argc, char *argv[])
{
int pid;
pthread_t tid[2];
pthread_attr_t attr;

  if (argc !=5){
    fprintf(stderr, "usage:a.out <int> <int> <int> <int> \n");
    return -1;
  }

  if (atoi(argv[1]) <= 0 | atoi(argv[2]) <= 0 | atoi(argv[3]) <= 0 | atoi(argv[4
 ]) <= 0 ) {
     fprintf(stderr, "%d %d %d %d must be >= 0\n", atoi(argv[1]), atoi(argv[2])
 , atoi(argv[3]), atoi(argv[4]) );
     return -1;
  }

  value = value + 30 ;

  printf("Before Fork: initial value = %d\n", value);
// Line--101 , fork
  pid = fork();
  if (pid == 0) {
// Line--102 , thread
      pthread_attr_init(&attr);
      pthread_create(&tid[0], &attr, runner0, argv[1]);
      pthread_create(&tid[1], &attr, runner1, argv[2]);
      pthread_join(tid[0],NULL);
      pthread_join(tid[1],NULL);
      printf("my pid = %d CHILD: value = %d \n", getpid(), value);
  }
  else if (pid > 0) {
       value = value + 100;
       printf("PARENT: value = %d\n", value);
       pthread_attr_init(&attr);
       pthread_create(&tid[0], &attr, runner0, argv[3]);
       pthread_create(&tid[1], &attr, runner1, argv[4]);
       pthread_join(tid[0],NULL);
       pthread_join(tid[1],NULL);
       printf("Before wait PARENT: value = %d \n", value);
// line--103 , wait
       wait(NULL);
       printf("my pid=%d After wait.PARENT: value=%d\n", getpid(), value);
       }
}
void *runner0(void *param) {
    int i, upper0 = atoi(param) * 100000;
```

注意：背面有試題

所別：資訊工程學系碩士班 科目：作業系統與計算機組織
　　　軟體工程研究所碩士班

```
     int j, value0;
     value0 = value;
// Line--104 , loop delay
     for (i = 1; i < upper0 ;i++) {
     j = i ;
     }
     value0 = value0 - 10;
     value = value0;

     printf("my pid=%d RUNNER0:value=%d upper0=%s\n",getpid(),value,param);
     pthread_exit(0);
     }

void *runner1(void *param) {
     int i, upper1 = atoi(param) * 100000;
     int j, value1;
     value1 = value;
// Line--105 , loop delay
     for (i = 1; i < upper1 ;i++) {
     j = i ;
     }
     value1 = value1 - 20;
     value = value1;

     printf("my pid=%d RUNNER1:value=%d upper1=%s\n",getpid(),value,param);
     pthread_exit(0);
     }

bash-2.03$ cc -pthread  fork-thread.c

// case--1
bash-2.03$ ./a.out 10 10 10 10
Before Fork: initial value = 30
PARENT: value = 130
my pid=3145 RUNNER0:value=120 upper0=10
my pid=3146 RUNNER0:value=20 upper0=10
my pid=3145 RUNNER1:value=100 upper1=10
my pid=3146 RUNNER1:value=0 upper1=10
Before wait PARENT: value = 100
my pid = 3146 CHILD: value = 0
my pid=3145 After wait,PARENT: value=100
bash-2.03$

// case--2
bash-2.03$ ./a.out 100 10 10 100
Before Fork: initial value = 30
PARENT: value = 130
my pid=3149 RUNNER0:value=120 upper0=10
my pid=3150 RUNNER1:value=10 upper1=10
my pid=3150 RUNNER0:value=20 upper0=100
my pid=3149 RUNNER1:value=100 upper1=100
my pid = 3150 CHILD: value = 20
Before wait PARENT: value = 100
my pid=3149 After wait,PARENT: value=100

// case--3
bash-2.03$ ./a.out 100 10 100 10
Before Fork: initial value = 30
PARENT: value = 130
my pid=3152 RUNNER1:value=10 upper1=10
my pid=3151 RUNNER1:value=f10 upper1=10
my pid=3152 RUNNER0:value=20 upper0=100
my pid = 3152 CHILD: value = 20
my pid=3151 RUNNER0:value=120 upper0=100
Before wait PARENT: value = 120
my pid=3151 After wait,PARENT: value=120
```