Department of Computer Science
Tsing Hua University
CS4100 Computer Architecture
Midterm, Fall 2006
Nov. 7, 2006

If you do not understand a question, please raise your hand. Keep your eyes on your own paper and do not talk during the exam.

1. (20 %) Answer TRUE or FALSE.
   (a) In addition to having mnemonic names such as $t1, the 32 MIPS integer registers can be also referred to by their numbers, starting from $1 up to and including $32. F
   (b) There is at least one MIPS instruction that can directly store a result of an ALU calculation into memory. F
   (c) The *jr* instruction can be used with different registers. T
   (d) In a recursive program $ra does not always need to be stored on the stack. T
   (e) If we only have one or two parameters to send to a non-recursive function, then we can use registers and do not need to use the stack. T
   (f) The stack grows towards address 0. T
   (g) Overflow will never occur if two signed numbers of different signs are added. T
   (h) A carry-out (i.e., having the value 1) at the most significant bit after an addition of two singed numbers always indicates overflow. F
   (i) IEEE 754 single precision floating-point numbers always have an implicit 1 placed to the left of the binary point. F
   (j) In the IEEE 754 single precision floating-point number format, just like in mathematics, any number to the 0th power is 1. F

2. (6%) Explain the following terms:
   (a) Wafer, Die, and Yield
   (b) SPEC2000
   Wafer:晶圓，由矽所組成，用來製造 IC 的原料，圓形薄片。
   Die: wafer 經過切割後，形成的方形薄片，上有許多電路。
   Yield: 良率，指製造 IC 時，能正常運作的 IC 佔全部 IC 的比例
   SPEC2000: 由好幾個程式組成的 benchmark，可以檢測 CPU 的效能，並給予一個評分；我們可以由此評分來比較不同 CPU 的效能。

   配分方式：
   一個半對得一分，四個都有半對得四分，有兩個全對再加一分。EX：如果有兩個全對，一個半對，分數是：3+1=4 分
   只有翻譯而沒有解釋，像 wafer 只寫晶圓，die 只寫晶片，都算半對，如果有多做解釋，但解釋不太正確，也是半對。
3. Two different compilers were used for the following C code. Compiler 1 generated the MIPS code in Figure (i) while Compiler 2 generated the MIPS code in Figure

(ii).

```
int foo(v[], k, n, m)
{
    sum = 0;
    for(i=k; i!=n; i=i+m)
        sum = sum + v[i];
    return sum;
}
```

```
         add $t0, $zero, $zero
         add $t1, $a1, $zero
 loop:   beq $t1, $a2, return
         sll $t2, $t1, 2
         add $t2, $t2, $a0
         lw $t3, 0($t2)
         add $t0, $t0, $t3
         add $t1, $t1, $a3
         j   loop
 return: add $v0, $t0, $zero
         jal $ra
```

(i) Code generate by Compiler 1

```
         add $v0, $zero, $zero
         sll $a1, $a1, 2
         sll $a2, $a2, 2
         sll $a3 ,$a3, 2
         add $t1, $a1, $zero
         beq $t1, $a2, return
 loop:   add $t2, $t1, $a0
         lw $t3, 0($t2)
         add $v0, $v0, $t3
         add $t1, $t1, $a3
         bne $t1, $a2, loop
 return:  jal $ra
```

(ii) Code generate by Compiler 2

(a) (7%) The following table lists the number of cycles for each type of instruction to take in the machine that you are using. Compute the average number of clocks per instruction (CPI) for each MIPS code when the function `foo(v,0,50,1)`is invoked.

Instr. Type	Instruction Cycles
arithmetic	2
logical	2
data transfer	5
conditional branch	3
unconditional jump	1

Code 1:

# of cycles = 153*2+50*2+50*5+51*3+51*1

= 306+100+250+153+51

= 860

# of instructions = 153+50+50+51+51 = 355

CPI = 860 / 355 = 2.42

Code 2:

# of cycles = 152*2+3*2+50*5+51*3+1*1

= 304+6+250+153+1

= 714

# of instructions = 152+3+50+51+1 = 257

CPI = 714 / 257 = 2.78

(b) (3%) If the machine that you are using has a processor operating at 2GHz, how long does it take to execute the function `foo(v,0,50,1)` for each MIPS code?

Code 1: $860 / 2*10^{9} = 4.3*10^{-7}$ (s)

Code 2: $714 / 2*10^{9} = 3.57*10^{-7}$ (s)

配分方式：

有五種 instruction，兩種 compiler，每種 compiler 的每個 instruction 數目各一分，EX：compiler 1 的五種 instruction 數目各為 153、50、50、51、51，每一個 instruction 數目各一分。

如果前面的 instruction 數目錯誤，得到錯誤的答案，後面的答案也因而錯掉，不會扣分。

計算錯誤扣一分。

4. (10%) You are going to enhance a computer, and there are two possible improvements: either make multiply instructions run four times faster than before, or make memory access instructions run two times faster than before. You repeatedly run a program that takes 100 seconds to execute. Of this time, 20% is used for multiplication, 50% for memory access instructions, and 30% for other tasks.

(a) What will the speedup be if you improve only multiplication?

(b) What will the speedup be if you improve only memory access?

(c) What will the speedup be if both improvements are made?

(a) 100 / (30+50+20/4) = 100/85 = 1.18

(b) 100 / (30 +50/2+20) = 100/75 = 1.33

(c) 100 / (30 +50/2 +20/4) = 100/60 = 1.67

配分方式：

總共三小題，對一個小題給四分，對兩個給七分，三個全對十分。

以第一題為例，對的標準為要至少寫出(100/85) or (20/17) or (1.18) or 加速 15s 只要有以上答案都視為正確，如果只有寫出 85s 或 85%都扣一分，其

他小題以此類推。

5. (10%) Consider the following MIPS code segment. To the left of each instruction we show the memory address (in base 16) where each instruction is stored.

Memory Address	**Label**	**Instruction**
0x1000 0010	back:	add $s2, $s3, $s4
…		
0x1000 0100		bne $s1, $zero, back

Below is the binary format for the *bne* instruction of the above code segment. Complete the binary representation of the instruction by writing the binary values for rs, rt, and immediate. (Recall that $17 = $s1 and $0 = $zero.)

31 opcode 26	25 rs 21	20 rt 16	15 immediate 0
000101	10001	00000	1111111111000011

rs => 3分 rt=>3分 immediate => 4分

若rs 和rt 寫顛倒 各扣1分

6. (8%) There are four instruction-set-architecture design principles: 1. Simplicity favors regularity, 2. Smaller is faster, 3. Make the common case fast, 4. Good design demands good comprises. Please give one MIPS example for each of the principle.

Simplicity favors regularity:

a. keeping all instructions a single size,
b. always requiring three register operands in arithmetic instructions
c. keeping the register fields in the same place in each instruction format.

Smaller is faster:

a. The desire for speed is the reason that MIPS has 32 registers rather than many more.
b. Using register (register is faster than memory)

Make the common case fast:

a. PC-relative addressing for conditional branches and immediate address for constant operands.
b. jal
c $zero
d. addi

Good design demands good compromises:

a. providing for larger address and constants in instructions and keeping all instructions the same length.

每小題 2 分，若有其他的答案會斟酌給分。

7. (10%) *beq $s1 $s2 L* (branch on equal); *bne $s1 $s2 L* (branch on not equal); *slt $s1 $s2 $s3* (set on less than), are three instructions designed in the MIPS. Please write a sequence of instructions to perform branch on (a) great than, (b) greater than and equal, (c) less than (d) less than and equal, using these instructions.

   (a) slt $s1 $s2 $s3   2 分
       beq $s1 $zero L
   (b) slt $s1 $s2 $s3   3 分
       beq $s1 $zero L
       beq $s2 $s3 L
   (c) slt $s1 $s2 $s3   2 分
       bnq $s1 $zero L
   (d) slt $s1 $s2 $s3   3 分
       bnq $s1 $zero L
       beq $s2 $s3 L

   如果各個小題 code 有小錯誤   會扣一分

8. Consider the following bit pattern:

   1011 1111 1110 0000 0000 0000 0000 0000

   What is the value that this pattern represents, assuming that it is:

   (a) (3%) A two's complement integer?

   (b) (3%) A single precision floating-point number?

   (a) $-1.075 * 10^9$

   (b) -1.75

9. (10%) (a) Please show how to perform the following 6-bit singed multiply, -7×-8, using 2’s complement representation. (b) Show how booth multiplier works. Just write down the partial products to be added or subtracted.