武汉大学计算机学院

2020-2021 学年第二学期 2019 级弘毅班

《计算机系统基础 2》期末考试试卷（A 卷）

姓名_____ 学号_____

（注：①闭卷考试；②考试时间为 120 分钟；③所有解答必须写在答题纸上。）

Problem 1. (10pts):

Single Choice.

1) Two characteristics of typical RISC instruction set architectures are:

a) Fixed instruction length, few registers

b) Fixed instruction length, only load/store instructions can access memory

c) Fixed instruction length, arithmetic instructions can access memory

d) Variable instruction length, many registers

e) Variable instruction length, arithmetic instructions can access memory

2) Using Hamming Error Correction Code, how many check bits are needed for 64-bit data to find two errors and correct one error?

a) 6

b) 7

c) 8

d) 9

e) 10

3) Say we are designing a computer where all the instructions are 32 bits and all instructions are of the format destination register, source register, immediate as shown below:

Opcode	Destination register	Source register	Immediate

If there are exactly 200 different instructions on this machine and there are 32 registers, how many bits would you expect the immediate field would hold?

a) 7

b) 9

c) 11

d) 12

e) 14

4) For a given branching pattern NT, T, T, NT, T, if a 2-bit branching predictor is used (as shown in the figure below) and the initial state is the bottom-left state, then what is the accuracy?

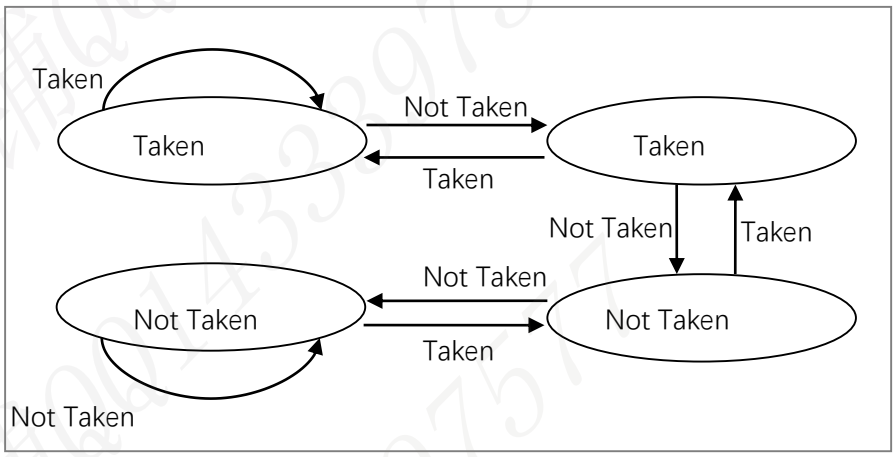
a) 20%

b) 40%

c) 60%

d) 80%

e) 100%



5) When comparing two sequences of transfers from a disk: (i) a single large transfer of 5MB; (ii) a sequence of 5 1-MB transfers all on the same track but at random sector numbers. Why is the single large transfer more efficient? Assume the wait time and controller overhead are negligible. Circle all that apply:

a) Reduction in total seek time

b) Reduction in total rotational delay

c) Reduction in total transfer time

d) Reduction in cache misses

e) The 5 smaller transfers are actually more efficient, thus none of the above

Problem 2: CPU Performance (10pts)

There are two different processors P1 and P2, which execute the same instruction set. P1 has a 3GHz clock rate and a CPI of 1.5. P2 has a 2.5GHz clock rate and a CPI of 1.0.

a) Which processor has the higher performance based on the number of instructions executed per second?

b) If each processor takes 10 seconds to execute a program, calculate the number of clock cycles and instructions of these two programs.

c) If you try to reduce the execution time of the above program by 20%, but this will result in an increase of 20% in the CPI, what clock rate should we have to get this time reduction?

Problem 3: RISC_V Assembly (22pts)

1. Convert the following C code into RISC_V assembly code by filling in the blanks below. The C code is used to recursively calculate the 2^ (n), and is defined as following:

power(0) = 1
power(1) = 2
power(n) = 2*power(n-1);

n = 0

Here's a C function to do this; your assembly-language program will follow this logic closely.

```
int power(int n)
{
    if (n==0) {
        return(1);
    } else {
        return(2*power(n-1));
    }
}
```

a) RISC_V assembly code, Argument n in $x10, Result in $x1.

```
Loop: addi   sp ,   sp,    ____(1)____
      sd     x1 ,   ____(2)____ (sp)
      sd     x10 ,  0(sp)
      bne    ____(3)____ ,  x0,  ____(4)____
      addi   x10 ,  x0,  ____(5)____
      addi   sp ,   sp,  ____(6)____
      jalr   x0,  ____(7)____
L1:   addi   x10 ,  x10,  ____(8)____
      jal    x1,  ____(9)____
      ld     x10 ,  0(sp)
      ld     x1 ,   ____(10)____
      addi   sp ,   sp,  ____(11)____
      add    x10 ,  x10,  ____(12)____
      jalr   x0,  ____(13)____
```

b) For each function call in program, show the content of the stack after the function call is made. Assume the stack pointer is originally at address 0x7ffffffc.

2. Write the RISC_V assembly code that creates the 32-bit constant 0x20014924 and stores that value to register x19.

a) If the current value of the pc is 0x20000000, can you use jal to get to the pc address as stored in register x19? You must clearly show your work to receive credit.

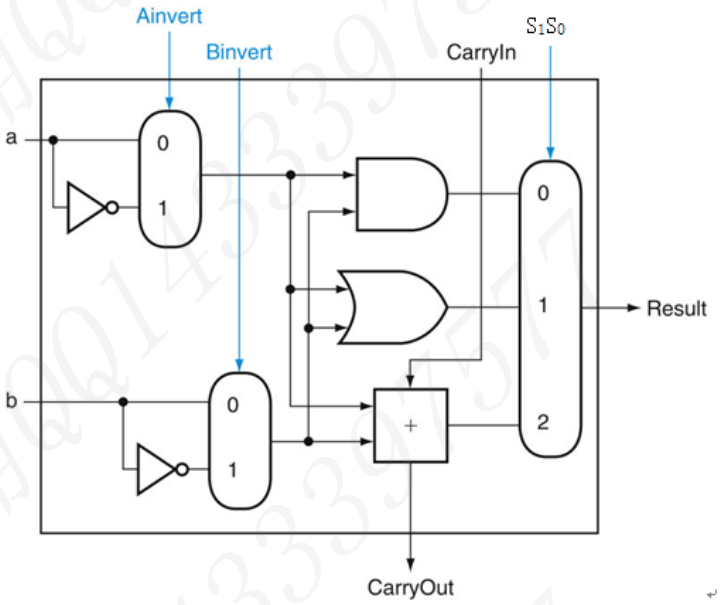
b) If the current value of the PC is 0x1FFFF000, can you use beq to get to the pc address as stored in register x19? You must clearly show your work to receive credit.

Problem 4: Arithmetic (8 pts)

The following diagram is an ALU.

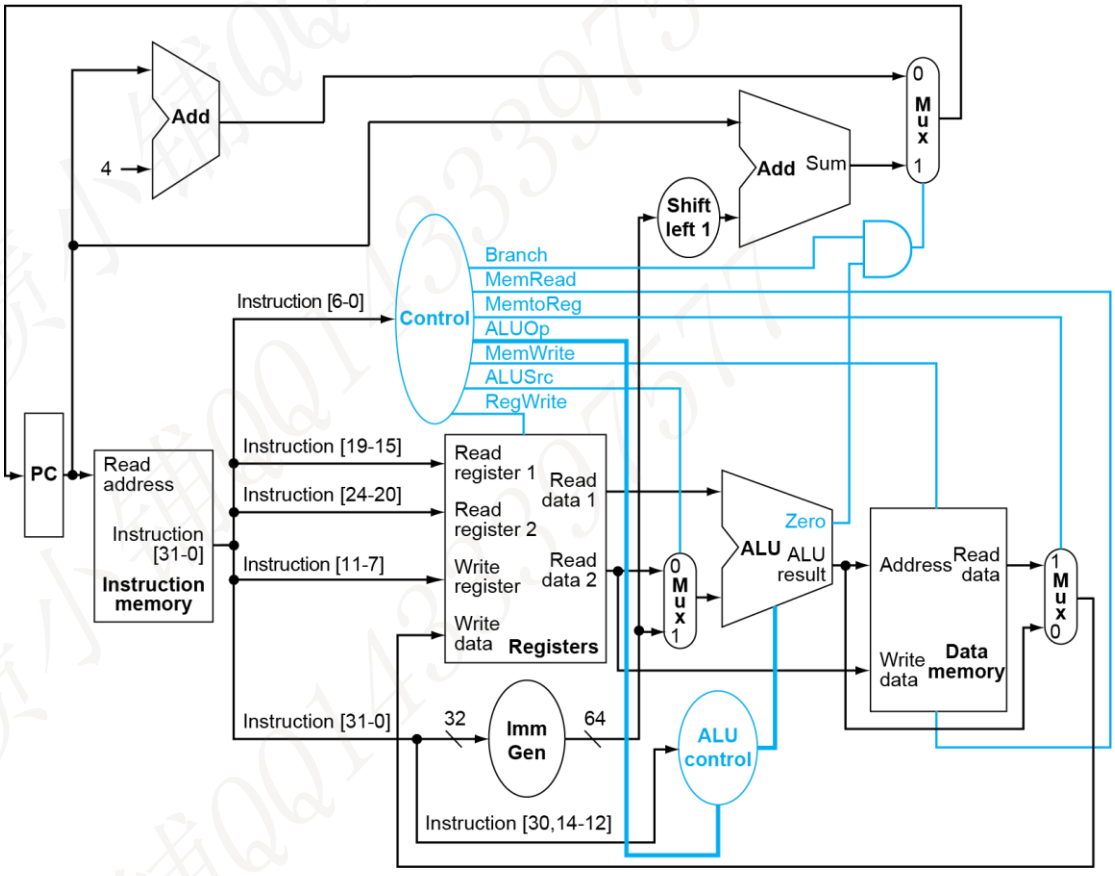
a) Please indicate input signals, output signals, and control signals.

b) Please list realized the combination of control signals which realized the arithmetic operation and logic operation.



Problem 5: Single-Cycle (10 pts)

Consider the following RISC_V single-cycle datapath:



We want to provide hardware support for a new instruction: ss rs1, rs2, imm.
Interpretation: Mem[Reg[rs1]] = Reg[rs2] + immediate

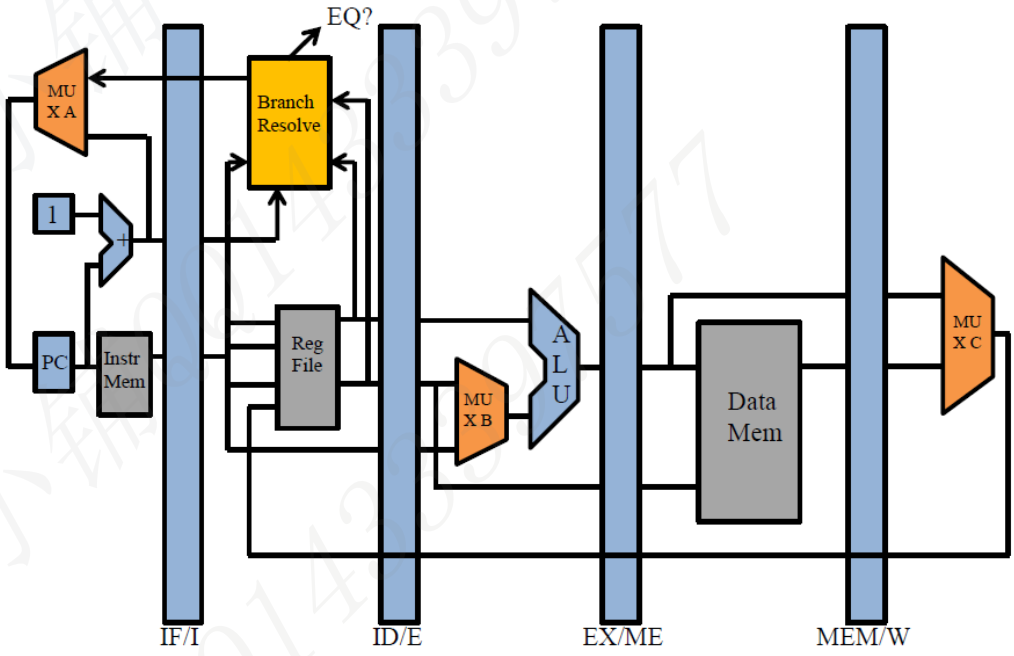
a) Indicate in the above diagram new hardware components that need to be added to the RISC_V. datapath to support the new instruction. You may only add wires,

MUXes, additional inputs to existing MUXes and control signals for those MUXes. Add as little of hardware as possible, label each new or modified hardware component in the diagram and describe that hardware below (include inputs, outputs, etc)

b) Give a step by step description of the RISC_V operation when executing the new instruction. For each Step, give the following information: (1) Single-sentence description of what the cycle is about. (2) Register updates.
Ex:
Step 1 Fetch instruction PC = PC+4

Problem 6: Pipeline (15 pts)

Consider the RISC_V 5-stage pipeline. We make one modification to the pipeline. A "branch resolving logic" is added in the ID stage as shown below. It has a built-in adder and comparator. The values of RegA and RegB as well as the sign-extended offset and PC+4 are passed on to the branch resolving logic. The PC can be overwritten by the logic if a branch is taken (i.e. EQ = 1).
By doing this we are able to resolve branches in the ID stage, instead of MEM stage. Note that branches are still predicted as not taken. Assume the data forwarding paths don't change



Assume that the following RISC_V code is executed on the pipelined processor:

```
        sub x12, x12, x11
label1: beq x12, x0, label2 # not taken once, then taken
        ld x13, 0(x12)
        beq x13, x0, label1 # taken
        add x11, x13, x11
label2: sd x11,0(x12)
```

a) The above design introduces a new data hazard. Explain the hazard. Use code

sequences to show all cases where this data hazard is exhibited. How many nops are required in each case if this hazard is solved with detect-and-stall?

b) The number of stall cycles for the new data hazard can be reduced using forwarding. Explain the necessary new forwarding paths to be added to minimize the number of stall cycles, specifying which pipeline register you are forwarding from and which stage you are forwarding to.

c) Draw the pipeline execution diagram for this code, assuming there are no delay slots.

Problem 7: Caches (15 pts)

1. Consider a cache with the following configuration: write-allocate, total size is 16 bytes, block size is 4 bytes, and 2-way associative. The memory address size is 16 bits and byte-addressable. The replacement policy is LRU. The cache is empty at the start. For the following memory accesses, indicate whether the reference is a hit or miss, and the type of a miss (compulsory, conflict, capacity):

(a) Write "Hit" or "Miss" for the third column and mark an "X" to identify the type of miss, if hit, do not write anything for the last three columns. The first two lines are already given.

Address	Type	Hit or Miss	Compulsory	Capacity	Conflict
0x2A	Load	Miss	X		
0x23	Load				
0x14	Load				
0x29	Load				
0x18	Load				
0x21	Load				
0x2B	Load				
0x1C	Load				
0x2D	Load				
0x17	Load				
0x19	Load				
0x1D	Load				

(b) Compare the number of bytes read and written from/to the memory when the cache is write-back verses write-through, you do not need to drain the write-back cache to memory at the end of the memory access sequence.

Number of bytes read and written from/to memory when the cache is write through:

Bytes read: _____

Bytes written: _____

Number bytes read and written from/to memory when the cache is a write back:

Bytes read: _____
Bytes written: _____

2. A program with the following instruction breakdown is run on the RISC_V processor:

add/sub	25%
beq	20%
sd	25%
ld	30%

Assume that the I-Cache hit rate is 85%, D-Cache hit rate is 90%. A branch predictor is used and 75% branches are predicted correctly. Memory access takes 100 ns.
The processor is a standard 5-stage pipeline with full forwarding. The clock frequency is 100MHz. Assume 35% of lw are followed immediately by a dependent instruction. Calculate the CPI and show your work.

Problem 8: VM（10 pts）

Consider a byte-addressable system with 8KB of physical memory. Virtual addresses are 16 bits long. Each page is 512B in size. The system has an eight-way set associative, 16 entry TLB. A portion of the single-level page and the TLB are shown below, all the numbers are in hexadecimal in the table.

TLB			
Index	Tag	PPN	Valid
0	09	4	1
	12	2	1
	10	0	1
	08	5	1
	05	7	1
	13	1	0
	10	3	0
	18	3	0
1	04	1	0
	0C	1	0
	12	0	0
	08	1	0
	06	7	0
	03	1	0
	07	5	0
	02	2	0

Page Table					
VPN	PPN	Valid	VPN	PPN	Valid
00	6	1	10	0	1
01	5	0	11	5	0
02	3	1	12	2	1
03	4	1	13	4	0
04	2	0	14	6	0
05	7	1	15	2	0
06	1	0	16	4	0
07	3	0	17	6	0
08	5	1	18	1	1
09	4	0	19	2	0
0A	3	0	1A	5	0
0B	2	0	1B	7	0
0C	5	0	1C	6	0
0D	6	0	1D	2	0
0E	1	1	1E	3	0
0F	0	0	1F	1	0

a) How many bits are the TLB tags and TLB indexes? Calculate a total size of TLB .

b) Given the virtual addresses to be read, please complete the following table to show the following for each read:
(1) TLB hit/miss (write H or M, or NA if TLB is not accessed)
(2) Physical Address (write NA if Page Table not accessed, or PF if page fault)

Virtual Address	TLB Hit/Miss/NA	Physical Address
0x013C		
0x0E23		
0x0128		
0x04FF		
0x14DF		
0x1DDE		

#bits used t