2013 学年春夏学期《计算机组成》复习纲要

教材：《Computer Organization & Design》3rd Edition

总则：

- 各章的 Real Stuff 不要求。
- 附录 A：A.3、A.4、A.8、A.9 不要求，A.10 的部分指令可能会考。
- 附录 B：逻辑门之前的内容(如触发器)不要求。
- 附录 C：微程序控制器不要求。

章节	内容	备注
第一章 概论 第四章 性能	计算机历史 软硬件组成 性能评价	● CPI、MIPS、FLOPS ● RISC、CISC ● 每章的 Fallacies&Pitfalls 结合具体内容选讲 ● 每章的 Real Stuff 不要求
第二章 指令是硬件机器的语言——计算机指令系统	指令系统 汇编反汇编 算术、逻辑指令 转移指令、子程序 寻址方式 C 语言手工编译	● 汇编指令转化成机器码(汇编) ● 机器码转化成汇编(反汇编) ● 典型指令：R-、LW、SW、BEQ、J、…其它参考附录 A。 ● C 语言程序、函数的手工编译，比如：strlen，strcpy，indexOf，数组求和、求最大最小等。
第三章 计算机中数的表示、转换与运算	数据表示 整数加减运算 整数乘除运算 浮点表示，加减运算	● 补码、移码、原码 ● 整数加减算法分析、优化 ● 加法器设计 ● 乘、除算法分析。指令不要求 ● 浮点数转换、四则运算。指令不要求
第五章 处理器——数据通路与控制器的设计	单个组件设计 MIPS 指令系统 ALU 与 ALU 控制器 单时钟数据通道 多时钟数据通道 控制器设计	● 基本组件：MUX、译码器、扩展、…… ● 单时钟：增加指令 ● 多时钟：增加指令 ● 控制器、有限状态机(FSM)
第七章 存贮体系结构	存储器概论， 位扩展字扩展 Cache 虚拟存储	● SRAM、DRAM、ROM/PROM/EPROM、刷新 ● 字位扩展 ● 映射策略、TAG 计算 ● 替换策略、写策略、3C 失配、平均时间 ● 页表、TLB
第八章 接口处理器和外部设备	I/O 概论 磁盘系统 总线系统、仲裁 数据通讯：轮询、中断、DMA	● 以概念为主 ● 磁盘计算 ● 轮询中断 DMA 时间计算

✧ 题目、题型以实际试卷为准。本样卷仅为参考

✧ 期末考试为英文、闭卷，可带手写 A4 笔记一页(单面或双面)，包括图表均不可打印复印。

浙江大学 2011～2012 学年 春夏 学期

《 计算机组成 》课程期末考试试卷（A）

课程号： 21186031 ，开课学院：计算机学院/软件学院 任课老师：

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式： 闭 卷，允许带 一页 A4 纸手写笔记 入场，笔记署名，不得互借

交卷方式： 试卷名字朝外对折整齐，草稿纸、笔记与试卷一起上交 。

考试日期： 2012 年 06 月 18 日(10:30～12:30) ，考试时间： 120 分钟

诚信考试，沉着应考，杜绝违纪。

考生姓名：_____ 学号：_____ 所属院系：_____

题序	一.10	二.20	三.25	四.30	五.15	总 分.100
得分						
评卷人						

I. True or False(10x1%; √/×)

eg.	1	2	3	4	5	6	7	8	9	10
√										

eg: TLB: translation-lookaside buffer

1、 Good design demands no compromise

2、 callee-saved register: A register saved by the routine making a procedure call.

9、 asynchronous bus: A bus that uses a handshaking protocol for coordinating usage rather than a clock; can accommodate a wide variety of devices of differing speeds.

10、 The advantage of polling is that it can save a lot of processor time.

II. Choose 1 best answer. (10x2%)

eg.	1	2	3	4	5	6	7	8	9	10
C										

eg. 1KB means (____) bytes。
A: 1　　B: 1000　　C: 1024　　D: 1024*1024

1、 Today's computers are built on 2 key principles: (____)
①Instruction are represented as numbers.
②Programs can be stored in memory to be read or written just like numbers.
③Make the common case fast.
④Every instruction can be conditionally executed.
A: ①③　　B: ②④　　C: ①②　　D: ③④

2、 According to the IEEE754 single precision, (____) is -∞.
A: 0xFF00_0000　　B: 0xFF100_0000
C: 0xFF80_0000　　D: 0xFFFFF_FFFF

9、 (____) is not a BUS.
A: PCI　　B: CPI　　C: ISA　　D: SCSI

10、 The major disadvantage of a bus is (____).
A: versatility　　B: Low cost
C: To create a communication bottleneck
D: Slower data access

III. (25%):

1)(10%) Put the corresponding letters for each 32-bit value in order from least to greatest.
(**Hint:** the question isn't asking you to write down what each one is, it only asks for the relative order!)
A: 0xFF000000 (IEEE754 single precision)
B: 0xFF000000 (2's complement)
C: 0xFF000000 (sign-magnitude)
D: 0xFF000000 (biased notation移码)
E: 0xF0000000 (2's complement)
F: 0xF0000000 (1's complement)

Least←							→Greatest

2)(15%): Suppose **float** $s1>$s0>0; $s3=0xFF000000, $s4=0x00800000. Try to do the MIPS programming for making:

$s2 = $s1 + $s0.

IV、(30%): Memory

1)(15%): Consider a memory system with the following properties:

	R/W time	**Size**
Cache	2ns	256KB
DRAM	20ns	4GB

Disk	20ms	400GB

We use TAG at Cache and Page Table in virtual memory for data addressing. Try to make a quantitative analysis(定量分析) for the questions:

1. Why does Cache and Virtual take different way? Shell we use Page-Table (Block-Table) for Cache?
2. Why don't use direct-mapping or set-associative in virtual memory?

2)(15%) 1G main memory, byte-addressing, 128KB Cache. Now a data locate at **0x123456**(byte-addressing), will mapping to which cache unit in different situation below, and how about its TAG and Total cache size?

0x123456	**The data will Mapping to (*block(s)*)**	**TAG**		**Tatal Size**
		TAG for the data(Hex)	bits	
Direct-mapped, 16 bytes/block				
Direct-mapped, 64 bytes/block				
2-Way set associative 16 bytes/block				
4-Way set associative 32 bytes/block				

V、(15%)Design: Multicycle CPU implementation

MIPS is a register-register architecture, where arithmetic source and destinations must be registers. But let's say we wanted to add a register-memory instruction to the multicycle datapath:

Addm rd, rs, rt # rd = rs + Mem[rt]

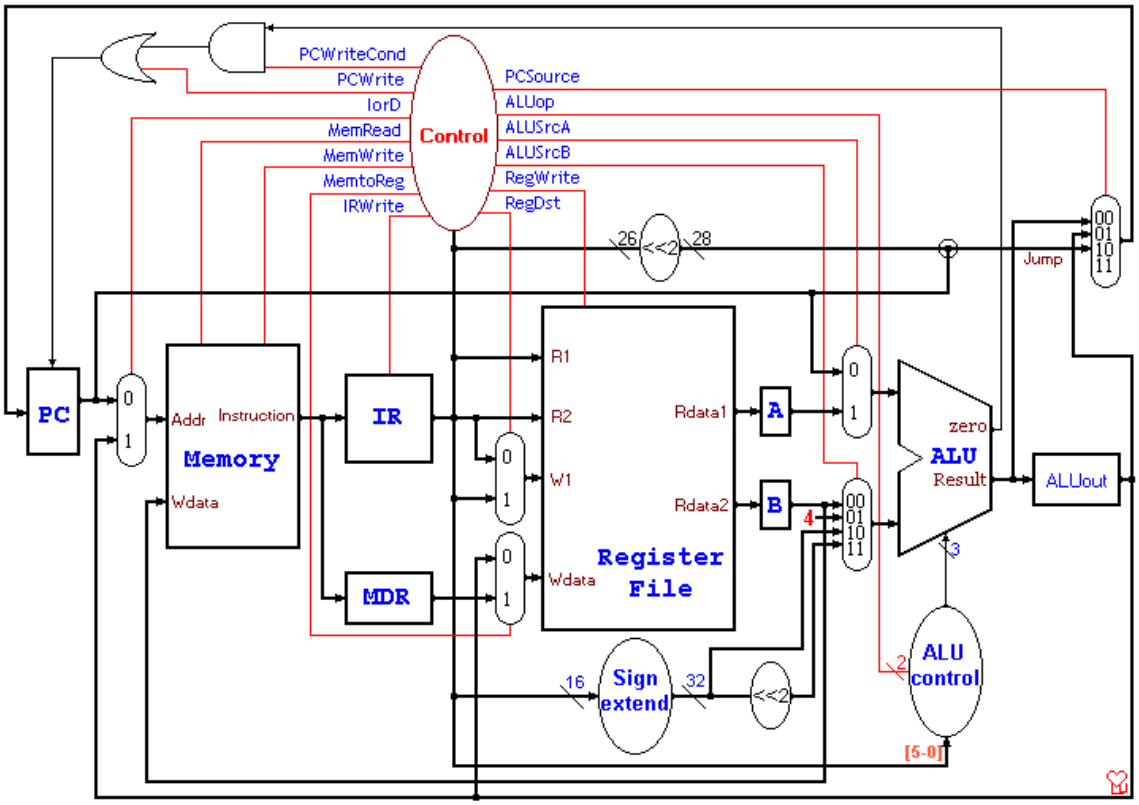
1.(3%)Machine code in Binary;

1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
OpCode																															
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0

2.(4%) show what changes are needed to support addm in the multicycle datapath.

3.(8%) Complete this finite state machine diagram for the addm instruction. Be sure to include any new control signals you may have added.

0、	1、	2、
3、	4、	5、


PCWriteCond
PCWrite
IorD
MemRead
MemWrite
MemtoReg
IRWrite
Control
PCSource
ALUop
ALUSrcA
ALUSrcB
RegWrite
RegDst
26
<<2
28
Jump
00
01
10
11
PC
0
1
Addr
Instruction
Memory
Wdata
IR
MDR
R1
R2
W1
Wdata
Register
File
Rdata1
Rdata2
A
B
4
zero
ALU
Result
ALUout
3
2
ALU
control
[5-0]
16
Sign
extend
32
<<2