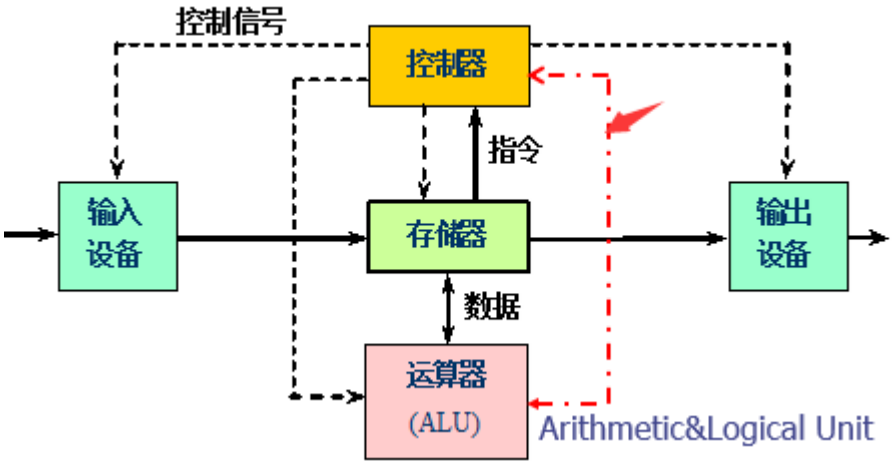
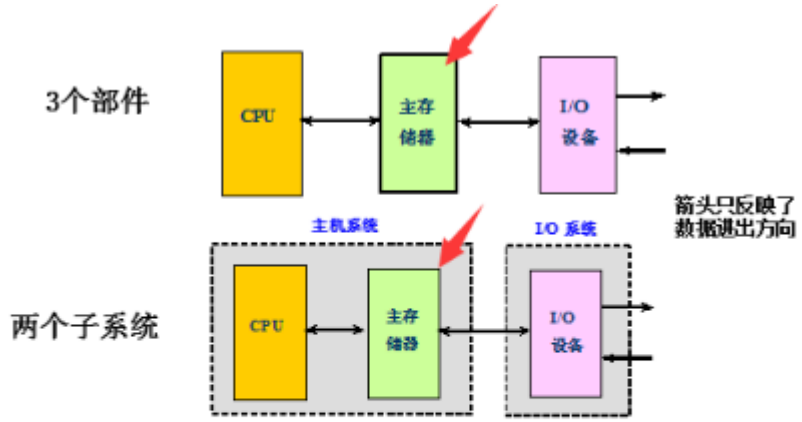
针对《计算机组成原理》教材勘误点

By：Holly 2023.6.14

1. **P2:图 1.2(a)中添加图中红线（运算单元的状态反馈和控制，如结果是不是等于零等）**

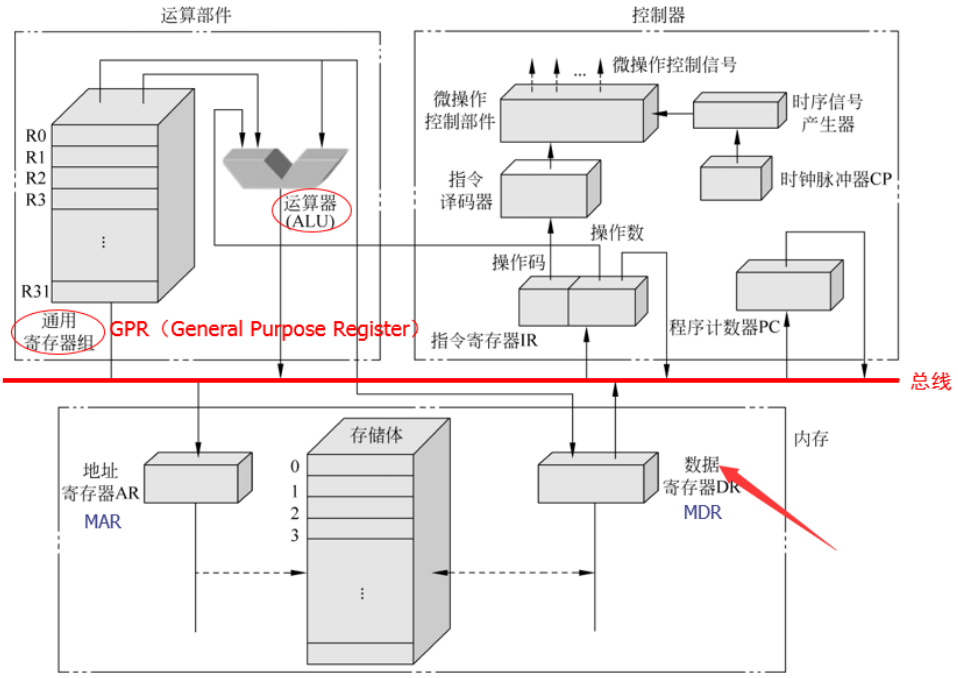


2. **P3 页建议把图中存储器就设定为主存储器。**



CPU	＝运算器	＋	控制器 （＋ 寄存器组）
存储器	＝主存（内存）	＋	辅存（外存）（属于IO设备）
主机	＝CPU	＋	主存储器
I/O设备	＝输入设备	＋	输出设备

3. **P4 图 1.4 见图所示建议用“数据/指令”更好些**



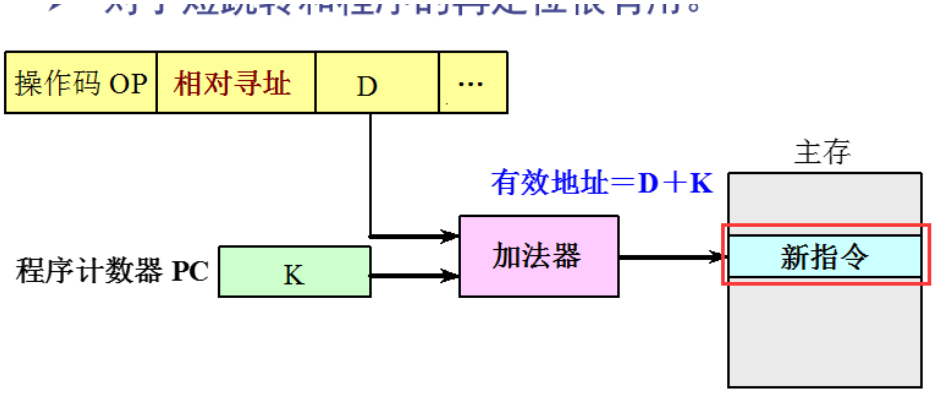
4. **P25 中例 2.15 第 4 个应该是 01101（现在是 01100）**

5. **P29 表 2.3 第一列两个内容调换一下，另外最后一行增加“规格化最小负数 1、1、111…1、10…00、-2^(-1)*2^(2^m-1)”内容更完整些。**

典型数据	浮点形式 数符 阶符 阶码(m) 尾数(n)	真值
非规格化 最小正数	0 0 00…0 00…01	$+2^{-n}\times 2^{-2^m}$
规格化 最小正数	0 0 00…0 10…00	$+2^{-1}\times 2^{-2^m}$
最大正数	0 1 11…1 11…11	$+(1-2^{-n})\times 2^{+(2^m-1)}$
规格化 最大负数	1 0 00…0 11…11	$-2^{-n}\times 2^{-2^m}$
非规格化 最大负数	1 0 00…0 01…11	$-(2^{-1}+2^{-n})\times 2^{-2^m}$
最小负数	1 0 11…1 00…00	$-1\times 2^{+(2^m-1)}$

6. **P30 页介绍单精度尾数 M 部分时，采用原码，用规格化表示（这里不准确，也有部分用非规格化表示（最小正数和最大负数部分），具体见我的文章 https://www.bilibili.com/read/cv19379004）**

7. **P175 的相对寻址这里其实是 PC+偏移量到内存中的代码段去取指，但图示是取数据，对初学者容易搞混淆，建议这部分把寻址先分成数据寻址和指令寻址，因为数据寻址中也有相对寻址（基址+偏移），指令寻址也有相对寻址（PC+偏移），图 7.8 把操作数改成新指令更合适些。**



8. **7.4 节介绍指令类型时，对于 IO 指令建议介绍下 IO 地址空间的概念，如果是存储器 IO，这时候一般不需要设计独立的 IO 指令。**

9. **P184 页图 7.14 建议位标从右到左排序，最左边是 MSB，最右边是 LSB，这和常规保持一致，目前是反的。**

10. **P187 表 7.5 改动处用红线标记了，依据是查了 MIPS 手册**

Jump And Link　　jal　　J　　R[31]=PC+8;PC=JumpAddr

指令举例	指令名称	含义
J　name	跳转	$PC_{27\cdots0}$← name<<2
JAL　name	跳转并链接	Regs[R31]←PC+8；$PC_{27\cdots0}$←name<<2；
JALR　R3	寄存器跳转并链接	Regs[R31]←PC+8；PC← Regs[R3]
JR　R5	寄存器跳转	PC← Regs[R5]
BEQZ R4，name	等于零时分支	if(Regs[R4]== 0)　PC←PC+4+name<<2； ((PC+4)－2^{17})≤name＜((PC+4)+2^{17})
BNE　R3，R4，name	不相等时分支	if(Regs[R3]!= Regs[R4]) PC←PC+4+name<<2； ((PC+4)－2^{17})≤name＜((PC+4)+2^{17})
MOVZ R1，R2，R3	等于零时移动	if(Regs[R3]==0)　Regs[R1]← Regs[R2]

11. **P197 页 8.11 图命名中建议把 R 类指令换成“运算类”指令更好些，虽然在模型机里所有运算类指令都是 R 封装，但是这里是指令功能实现，建议用“运算类”更好些。这部分整体都有这个问题，其他指令都是用功能，运算类指令却用 R 类来代替，感觉不对称。或者用采用 R 封装的运算类指令。**

12. **P198 表 8.1 下面文字 funct（6 位），现在是 5 位。**

13. **P199 表 8.3 中红 X 为改正地方。**

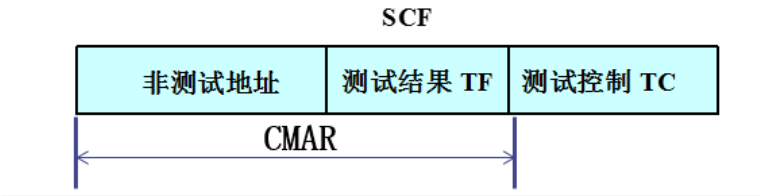
ALU控制器的输入								ALU控制器的输出 A2 A1 A0
ALUOp		funct字段						
ALUOp1	ALUOp0	F5	F4	F3	F2	F1	F0	
0	×	×	×	×	×	×	×	010
1	×	×	×	0	0	0	0	010
1	×	×	×	0	0	1	0	110
1	×	×	×	0	1	0	0	000
1	×	×	×	0	1	0	1	001
1	×	×	×	1	0	1	0	111

14. **P201 页逻辑表达式错了，见下图：**

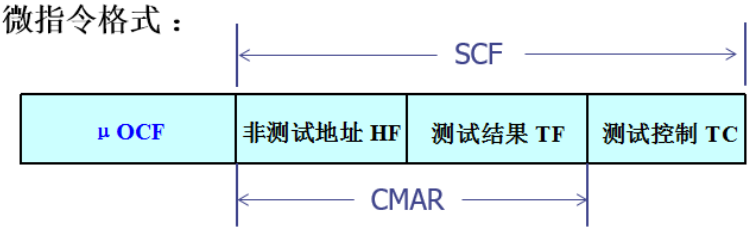
$$ALUSrcA = \overline{Op5} \bullet \overline{Op4} \bullet \overline{Op3} \bullet \overline{Op2} \bullet \overline{Op1} \bullet \overline{Op0} + Op5 \bullet \overline{Op4} \bullet \overline{Op3} \bullet \overline{Op2} \bullet Op1 \bullet Op0$$
$$+ Op5 \bullet \overline{Op4} \bullet OP3 \bullet \overline{Op2} \bullet Op1 \bullet Op0$$

15. **P235 页关于 SCF 的定义感觉和其他资料有出入，目前改成这样了：**

微地址格式

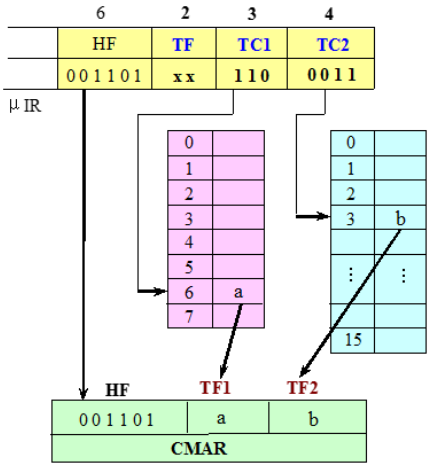


微指令格式：



对应的图 9.11：

例如：微地址8位，微程序能实现4路并行转移。可控制微程序的测试条件共23个，分成2组，第一组8个，第二组15个。



- 微地址8位，所以最多支持256条微指令。
- 4路并行转移，所以TF是2位。

  非测试位数：8-2=6位。
- TC1、TC2是测试结果TF1和TF2的测试控制（测试条件）字段。
  - ✓ 第一组8个，所以TC1:3位；
  - ✓ 第一组15个，所以TC2:4位；
- 测试结果TF和非测试位段HF，直接送CMAR。

16. P238 页表 9.3 第二行 OP 位段应该是 63 对应编码 111 111（见 P191 页操作码分配部分），P239 表 9.4OP 对应操作码 35 和 43 也错了。

17. P239 页表 9.5 中减法操作在第 8 章中是通过 funct 位段控制的，不是通过 ALUop=11 实现的。

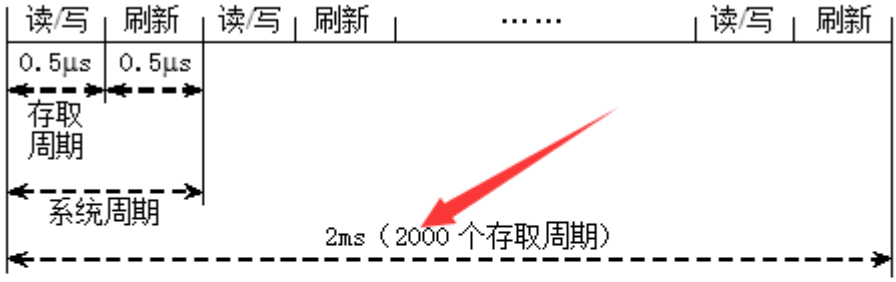
字段名	字段值	控制信号（激活）	功能
IRWrite 1-bit	0		无操作
	1		对IR进行写入操作
ALUCtrl 2-bit	Add	ALUOp=00	使ALU进行加操作
	Subt（删掉）	ALUOp=11	使ALU进行减操作（删掉，第八章没设计该功能）
	funct code	ALUOp=10	用机器指令的funct字段来决定ALU的操作
SRC1 1-bit	PC+4	ALUSrcA=0	选PC+4为ALU的第一个输入
	A	ALUSrcA=1	选寄存器A为ALU的第一个输入
SRC2 2-bit	B	ALUSrcB=00	选寄存器B为ALU的第二个输入
	Extend	ALUSrcB=01	选“Imm”部件的输出作为ALU的第二个输入
	Extshft	ALUSrcB=10	选“左移两位”部件的输出作为ALU的第二个输入

18. P265 图 10.25 文字中：应该是两个数据输出端口和一个数据输入端口。

19. P267 最后一行括号里（为 0）去掉，只有 X 和 Y 都为了才是，但根据上下文，应该是判断其一是不是。

20. P285 页介绍 DRAM 的刷新建议修改为：

分散刷新：



优点：没有死区。

缺点：刷新过于频繁。（2ms刷新2000次，相较于其他方法的128次）

系统存取周期是存储芯片存取周期的两倍，降低了访问存储器的速度。

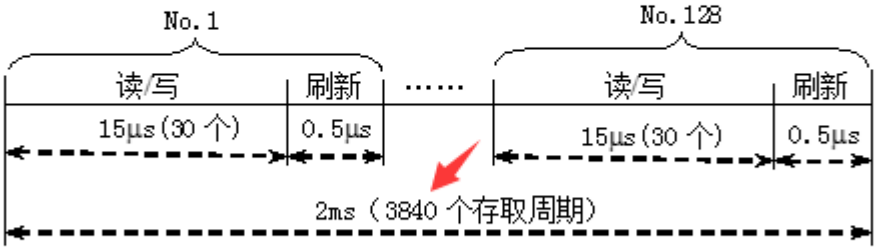
异步刷新：

异步式刷新

- 把刷新操作平均分配到整个最大刷新间隔内进行。
- 相邻两行的刷新间隔为：

  最大刷新间隔时间÷行数 = 2ms/128=15.625μs

  15μs读写+0.5μs刷新，有0.125μs浪费



优点：避免了长时间的“死区”问题，避免了频繁刷新。

缺点：存取周期没有集中刷新多。