

第六章 中央处理器

2. 简单回答下列问题。(参考答案略)

- (1) CPU的基本组成和基本功能各是什么？
- (2) 取指令部件的功能是什么？
- (3) 控制器的功能是什么？
- (4) 为什么对存储器按异步方式进行读写时需要WMFC信号？按同步方式访问存储器时，CPU如何实现存储器读写？
- (5) 单周期处理器的CPI是多少？时钟周期如何确定？为什么单周期处理器的性能差？元件在一个指令周期内能否被重复使用？为什么？
- (6) 多周期处理器的设计思想是什么？每条指令的CPI是否相同？为什么在一个指令周期内某个元件可被重复使用？
- (7) 单周期处理器和多周期处理器的控制逻辑设计的差别是什么？
- (8) 硬布线控制器和微程序控制器的特点各是什么？
- (9) 为什么CISC大多用微程序控制器实现，RISC大多用硬布线控制器实现？
- (10) 水平型微指令和垂直型微指令的基本概念和优缺点是什么？
- (11) CPU检测内部异常和外部中断的方法有什么不同？

3. 在书中图6.9中，假定总线传输延迟和ALU运算时间分别是20ps和200ps，寄存器建立时间为10ps，寄存器保持时间为5ps，寄存器的锁存延迟（Clk-to-Q time）为4ps，控制信号的生成延迟（Clk-to-signal time）为7ps，三态门接通时间为3ps，则从当前时钟到达开始算起，完成以下操作的最短时间是多少？

- (1) 将数据从一个寄存器传送到另一个寄存器
- (2) 将程序计数器PC加1

参考答案：

(1) 寄存器的锁存延迟与控制信号的生成延迟的时间重叠，且Clk-to-signal time > Clk-to-Q time，所以完成寄存器传送的时间延迟为：

$$7+3+20+10=40\text{ps}。$$

(2) 分两个阶段：

$$\text{PC}+1 \rightarrow \text{Z} : 7+3+20+200+10=240\text{ps}；$$

$$\text{Z} \rightarrow \text{PC} : 7+3+20+10=40\text{ps}$$

寄存器保持时间用来作为时间约束。

4. 图6.30给出了某CPU内部结构的一部分，MAR和MDR直接连到存储器总线（图中省略）。在两个总线之间的所有数据传送都需经过算术逻辑部件ALU。ALU可实现的部分功能及其控制信号如下：

MOVa: $F=A$; MOVb: $F=B$;
a+1: $F=A+1$; b+1: $F=B+1$
a-1: $F=A-1$; b-1: $F=B-1$

其中A和B是ALU的输入，F是ALU的输出。假定JSR（转子指令）指令占两个字，第一个字是操作码，第二个字给出子程序的起始地址，返回地址保存在主存的栈中，用SP（栈指示器）指向栈顶，按字编址，每次从主存读取一个字。请写出读取并执行JSR指令所要求的控制信号序列（提示：当前指令地址在PC中）。

参考答案：

假定采用同步方式（若为异步，则只需在read和Write后加一个等待信号WMFC）
分三个阶段：

1. 取指令操作码：PCout, MOVb, MARin

Read, b+1, PCin

MDRout, MOVb, IRin

2. 取子程序首址：PCout, MOVb, MARin

Read, b+1, Yin

MDRout, MOVb, PCin

3. 保存返址至栈：SPout, MOVb, MARin

Yout, MOVb, MDRin

Write, SPout, b-1, SPin

(注：若按最长的存储访问时间作为CPU时钟周期，则上述每个阶段都需三个时钟周期)
能否用更少的时钟周期完成上述功能？不能！以下是另一种方式

1. 取指令操作码：PCout, MOVb, MARin

Read, b+1, Yin

MDRout, MOVb, IRin

2. 取子程序首址：Yout, MOVb, MARin

Read, a+1, Yin (用b+1也行)

MDRout, MOVb, PCin

3. 保存返址至栈：SPout, MOVb, MARin

Yout, MOVb, MDRin

Write, SPout, b-1, SPin

5. 假定某计算机字长16位，CPU内部结构如书中图6.9所示，CPU和存储器之间采用同步方式通信，按字编址。采用定长指令字格式，指令由两个字组成，第一个字指明操作码和寻址方式，第二个字包含立即数Imm16。若一次存储访问所花时间为2个CPU时钟周期，每次存储访问存取一个字，取指令阶段第二次访存将Imm16取到MDR中，请写出下列指令在指令**执行阶段**的控制信号序列，并说明需要几个时钟周期。

(1) 将立即数Imm16加到寄存器R1中，此时，Imm16为立即操作数。

即： $R[R1] \leftarrow R[R1] + Imm16$

(2) 将地址为Imm16的存储单元的内容加到寄存器R1中，此时，Imm16为直接地址。

即： $R[R1] \leftarrow R[R1] + M[Imm16]$

- (3) 将存储单元Imm16的内容作为地址所指的存储单元的内容加到寄存器R1中。此时，Imm16为间接地址。即： $R[R1] \leftarrow R[R1] + M[M[Imm16]]$

参考答案：

(1) MDRout, Yin

R1out, add, Zin

Zout, R1in

需3个时钟周期

(2) MDRout, MARin

Read1, (R1out, Yin也可以放在该控制信号所在的时钟周期中)

Read2, R1out, Yin

MDRout, add, Zin

Zout, R1in

需5个时钟周期

(3) MDRout, MARin

Read1

Read2

MDRout, MARin

Read1, (R1out, Yin)

Read2, R1out, Yin

MDRout, add, Zin

Zout, R1in

需8个时钟周期

6. 假定P.177图6.23单周期数据通路对应的控制逻辑发生错误，使得在任何情况下控制信号RegWr、RegDst、Branch、MemWr、ExtOp总是为0，则哪些指令不能正确执行？为什么？

参考答案：

若RegWr=0，则所有需写结果到寄存器的指令（如：R-Type指令、load指令等）都不能正确执行，因为寄存器不发生写操作；

若Regdst=0，则所有R-Type指令都不能正确执行，因为目的寄存器指定错误；

若Branch=0，则Branch指令可能出错，因为永远不会发生转移；

若MemWr=0，则Store指令不能正确执行，因为存储器不能写入所需数据；

若ExtOp=0，则需要符号扩展的指令（如Beq、lw/sw等）发生错误。

7. 假定P.177图6.23单周期数据通路对应的控制逻辑发生错误，使得在任何情况下控制信号RegWr、RegDst、Branch、MemWr、ExtOp总是为1，则哪些指令不能正确执行？为什么？

参考答案：

若RegWr=1，则所有不需写结果到寄存器的指令（如：sw、beq等）都不能正确执行；

若Regdst=1，则lw和ori等指令不能正确执行，因为目的寄存器指定错误；

若Branch=1，则非Branch指令可能出错，因为可能会发生不必要的转移；

若MemWr=1，则除Store指令外其他指令都不能正确执行，因为存储器总会写入数据；

若ExtOp=1，则需要零扩展的指令（如ori等）会发生错误。

8. 在MIPS指令集中需要增加一条swap指令，可以使用软件方式用若干条已有指令来实现伪指令，也可以通过改动硬件来实现。

（1）写出用伪指令方式实现“swap \$rs, \$rt”时的指令序列

（2）假定用硬件实现时会使一条指令的执行时间增加10%，则swap指令在程序中占多大的比例才值得用硬件方式来实现？

参考答案：

（1） swap指令可用以下三条指令实现。

xor \$rs, \$rs, \$rt

xor \$rt, \$rs, \$rt

xor \$rs, \$rs, \$rt

（若使用额外寄存器\$rtmp，则\$rtmp内容会被破坏，所以伪指令一般不能用额外寄存器）

add \$rtmp, \$rs, \$zero

add \$rs, \$rt, \$zero

add \$rt, \$rtmp, \$zero

（2）假定该指令占x%，其他指令占(1-x)%

则用硬件实现该指令时，程序执行时间为原来的 $1.1 \cdot (x+1-x) = 1.1$ 倍

用软件实现该指令时，程序执行时间为原来的 $3x+1-x = (2x+1)$ 倍

当 $1.1 < 2x+1$ 时，硬件实现才有意义

由此可知， $x > 5\%$

9. 假定P.185图6.32多周期数据通路对应的控制逻辑发生错误，使得在任何情况下控制信号PCWr、IRWr、RegWr、BrWr、PCSource、MemWr、MemtoReg、PCWrCond总是为0，则哪些指令不能正确执行？为什么？

参考答案：

若PCWr=0，则所有指令都不正确，因为无法更新PC

若IRWr=0，则所有指令都不能正确执行，因为IR中不能写入指令

若RegWr=0，则所有需要写结果到寄存器的指令（如：R-Type指令、load指令等）都不能正确执行，因为寄存器不发生写操作

若PCSource=00，则除j之外的其他指令都不能正确得到下条指令地址

若MemWr=0，则Store指令不能正确执行，因为存储器不能写入数据

若MemtoReg=0，则所有Load指令执行错误，因为寄存器写入的是ALU输出

若PCWrCond=0，则Branch指令不能正确执行，因为不能写入转移目标地址到PC

10. 假定P.185图6.32多周期数据通路对应的控制逻辑发生错误，使得在任何情况下控制信号PCWr、IRWr、RegWr、BrWr、PCSource、MemWr、MemtoReg、PCWrCond总是为1，则哪些指令不能正确执行？为什么？

参考答案：

若PCWr=1，则程序执行顺序失控，因为每个时钟都会更新PC

若IRWr=1，则所有指令都可能不能正确执行，因为写入IR的可能不是当前指令

若RegWr=1，则所有不需写结果到寄存器的指令（如：sw、beq等）都不能正确执行

若PCSource=01，则j和Branch指令不能正确得到下条指令地址

若MemWr=1，则除Store指令外的所有指令都不能正确执行

若MemtoReg=1，则除Load外的所有指令执行错误

若PCWrCond=1，则除Branch外的其他指令可能不能正确执行

13. 对于多周期MIPS处理器，假定将访问数据的过程分成两个时钟周期可使时钟频率从4.8GHz提高到5.6GHz，但会使得lw和sw指令增加时钟周期数。已知基准程序CPUint 2000中各类指令的频率为：Load: 25%，Store: 10%，Branch: 11%，Jump: 2%，ALU: 52%。以基准程序CPUint 2000为标准计算时钟频率提高后处理器的性能提高了多少？若将取指令过程再分成两个时钟周期，则可进一步使时钟频率提高到6.4GHz，此时，时钟频率的提高是否也能带来处理器性能的提高？为什么？

参考答案：

SPEC CPUint 2000混合指令频率为：

Load: 25%

Store: 10%

Branch: 11%

Jump: 2%

ALU: 52%

假设M1、M2和M3分别表示时钟频率为4.8GHz、5.6GHz和6.4GHz的多周期处理器，从有限状态图分析，得知：

M1中上述各类指令的CPI分别为5、4、3、3、4

M2中上述各类指令的CPI分别为6、5、3、3、4

M3中上述各类指令的CPI分别为7、6、4、4、5

$CPIM1 = 25\% \times 5 + 10\% \times 4 + 11\% \times 3 + 2\% \times 3 + 52\% \times 4 = 4.12$

$CPIM2 = 25\% \times 6 + 10\% \times 5 + 11\% \times 3 + 2\% \times 3 + 52\% \times 4 = 4.47$

$CPIM3 = 25\% \times 7 + 10\% \times 6 + 11\% \times 4 + 2\% \times 4 + 52\% \times 5 = 5.47$

（实际上就是每个指令都加一个时钟周期，所以 $CPIM3 = CPIM1 + 1 = 5.47$ ）

$MIPSM1 = 4.8G / 4.12 = 1165$

$MIPSM2 = 5.6G / 4.47 = 1253$

$MIPSM3 = 6.4G / 5.47 = 1170$

由此可见，数据存取改为双周期的做法效果较好。进一步把取指令改为双周期的做法反而使MIPS数变小了，所以不可取。因为数据存取只涉及到load/Store指令，而指令存取涉及到所有指令，使得CPI显著提高。

15. 微程序控制器容量为 1024×48 位，微程序可在整个控存内实现转移，反映所有指令执行状态转换的有限状态机中有4个分支点，微指令采用水平格式，微地址由专门的下地址字段确定。请设计微指令的格式，说明各字段有多少位？为什么？

参考答案：

微程序控制器容量为 1024×48 位，说明微地址占10位，微指令字共48位，其中10位下地址字段用来给出下条微地址；转移控制字段需要对5种情况进行控制，需3位。例如，

000：取指令微指令首地址

100：根据分支1处的条件选择下条微地址

101：根据分支2处的条件选择下条微地址

110：根据分支3处的条件选择下条微地址

111：根据分支4处的条件选择下条微地址

剩下的 $48 - 10 - 3 = 35$ 位用来表示微操作码字段。

（如果采用计数器法，则转移控制字段需要对6种情况进行控制，比上述5种情况多一种：即顺序执行下条微指令，此时，也要3位。）

16. 以下异常或中断各在哪个时钟周期中检测？

除数为0、溢出、无效指令、无效指令地址、无效数据地址、缺页、访问越权、中断

参考答案：

- a. “除数为 0”异常在取数/译码(ID/Reg)周期进行检测
- b. “溢出”异常在 R-Type 指令的执行(Exe)周期进行检测
- c. “无效指令”异常在取数/译码(ID/Reg)周期进行检测
- d. “无效指令地址”、“缺页”和“访问越权”异常在取指令(IF)周期检测
- e. “无效数据地址”、“缺页”和“访问越权”异常在存储器访问(Mem)周期检测
- f. “中断”可在每条指令的最后一个周期(WB)进行检测