高等计算机体系结构，2020年春季

作业2： 单周期vs.多周期微体系结构

主讲教师：栾钟治

助讲教师：杨海龙；助教：孙庆骁

作业下发时间：2020年3月27日

作业回收时间：2020年4月10日

# MIPS单周期微体系结构分析 75分

图1为基本的单周期MIPS实现。

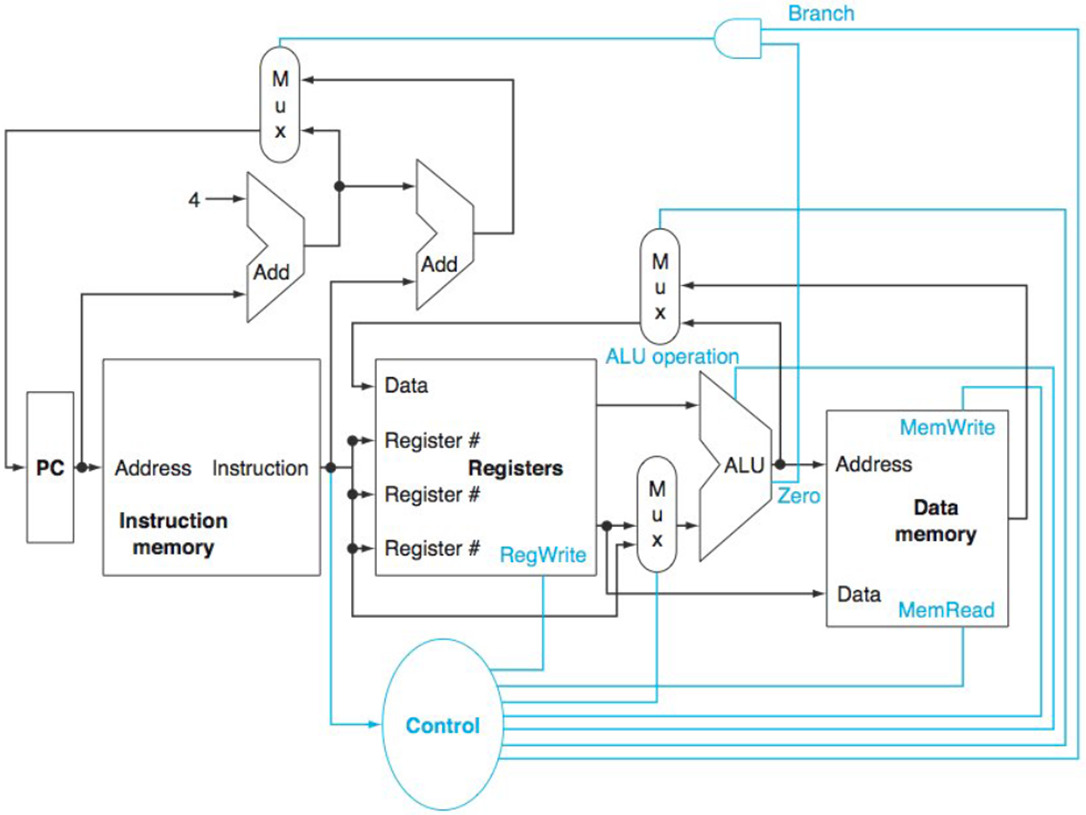


图1

1.1 不同单元有不同的延迟时间。在图1中有七种主要单元。对一条指令而言，关键路径(产生最长延迟的那条路径)上各个单元的延迟时间决定了该指令的最小延迟。假设个单元的延迟时间如下表所示，回答下列3个问题。9分

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **指令存储器** | **加法器** | **多选器** | **ALU** | **寄存器堆** | **数据存储器** | **控制** |
| 400ps | 100ps | 30ps | 120ps | 200ps | 350ps | 100ps |

(a) 对一条MIPS的与指令(AND)而言，关键路径是什么？

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 多路数据选择器(MUX)**

**关键路径延迟：400+200+30+120+30 = 780ps**

(b) 对一条MIPS的装载指令(LW)而言，关键路径是什么？

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 数据存储器(DM) -> 多路数据选择器(MUX)**

**关键路径延迟：400+200+30+120+350+30 = 1130ps**

(c) 对一条MIPS的相等则分支指令(BEQ)而言，关键路径是什么？

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 多路数据选择器(MUX)**

**关键路径延迟：400+200+30+120+30+200 = 780ps**

1.2 图1中基本的单周期MIPS实现仅能实现某些指令。可以在这个指令集中加入新的指令，但决定是否加入取决于给处理器的数据通路和数据通路增加的复杂度。对于下表中的新指令而言，试回答下列3个问题。18分

|  |  |
| --- | --- |
| **指令** | **解释** |
| add3 Rd,Rs,Rt,Rx | Reg[Rd]=Reg[Rs]+Reg[Rt]+Reg[Rx] |

(a) 对上述指令而言，哪些已有的单元还可以被使用？

**指令存储器(IM)、寄存器堆(RF)现存的读取数据端口和数据输出端口、算术逻辑单元(ALU)**

(b) 对上述指令而言，还需要增加哪些功能单元？

**寄存器的另一个读取端口（用于读取Rx）和第二个ALU（用于将Rx与Rs+Rt相加）**

(c) 为了支持这些指令，需要在控制单元增加哪些信号？

**需要一个控制信号来告诉新的ALU做什么，或者如果我们扩展了现有的ALU，我们需要添加一个新的ADD3操作。**

当设计者考虑改进处理器数据通路时，往往要考虑性能与成本的折中。假设我们从图1的数据通路出发，其中指令存储器(Instruction Memory)、加法器(Add)、多选器(Mux)、ALU、寄存器堆(Registers)、数据寄存器(Data Memory)和控制单元(Control)的延迟分别为400ps、100ps、30ps、120ps、200ps、350ps和100ps，相应的成本分别为1000、30、10、100、200、2000和500。试根据表中的改进分别回答下列问题。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **改进** | **延迟** | **成本** | **优势** |
| **a.** | 更快的加法器 | 加法单元-20ps | 每个加法单元+20 | 把已有的加法器用更快的加法器替代 |
| **b.** | 更大的寄存器堆 | 寄存器堆+100ps | 寄存器堆+200 | 需要更少的load和store指令。这将导致指令数减少5% |

(d) 改进前后的时钟周期分别是多少？

**改进前的最长关键路径是：**

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 数据存储器(DM) -> 多路数据选择器(MUX)**

**改进后方案a时钟周期为：1130ps（改进后关键路径不变，因为加法器不再关键路径上所以不变）**

**改进后方案b时钟周期为：1130+100=1230ps:（寄存器在关键路径上）**

(e) 改进后将获得多大的加速比？（加速比：旧时钟周期/新时钟周期）

**a方案加速比为：1130/1130 = 1**

**b方案加速比为：（1/0.95）\*（1130/1230）=0.97性能实际上有点减速。**

(f) 比较改进前后的性能/价格比，进行这样的改进是否有意义？(按照性价比=性能/价格来计算)

**改进前的全部器件的成本为：1000+2\*30+3\*10+100+200+2000+500 = 3890**

**a方案改进后的成本为：3890+20\*2 = 3930**

**相对成本：(3890/3930) = 1.01**

**性能/价格比：1.01/1 = 1.01**

**改进没有意义，为了相同的性能付出更多的代价。**

**b方案改进后的成本为：3890+200 = 4090**

**相对成本：(4090/3890) = 1.05**

**性能/价格比：1.05/0.95 = 1.08**

**改进没有意义，并且速度下降，因此成本/性能变差。**

1.3 下表给出了实现处理器数据通路的逻辑单元延迟。试根据下表情况分别回答下列问题。15分

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **指令存储器** | **加法器** | **多选器** | **ALU** | **寄存器堆** | **数据存储器** | **符号扩展** | **左移两位** |
| 400ps | 100ps | 30ps | 120ps | 200ps | 350ps | 20ps | 2ps |

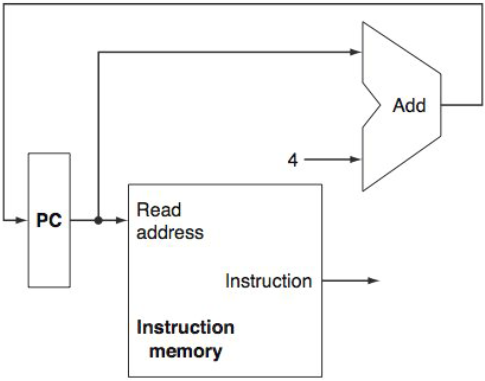


图2

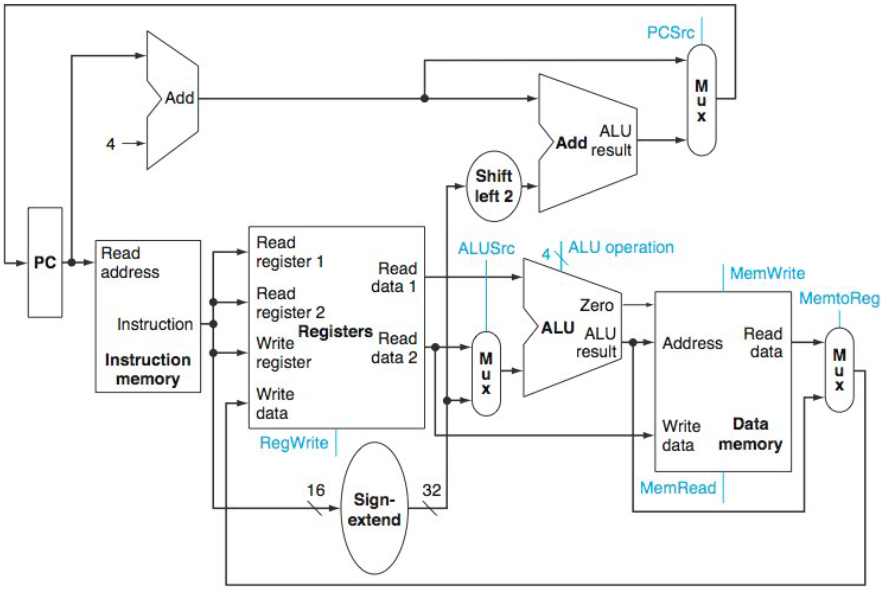


图3

(a) 如果处理器只需做连续取指这一件事(见图2)，那么时钟周期是多少？

**取指的关键路径为指令存储器(IM),则时钟周期为400ps。**

(b) 考虑一个与图3类似的数据通路，但是假设处理器只需处理无条件相对跳转指令，那么时钟周期是多少？

**无条件相对跳转指令的关键路径为：**

**指令存储器(IM) -> 符号扩展(SE) -> 移位器(Shift) ->加法器(ALU) -> 多路选择器(MUX)**

**时钟周期为；400+20+2+100+30 = 552ps**

(c) 同样考虑一个与图3类似的数据通路，但这次假设只需处理有条件相对跳转指令，那么时钟周期是多少？(请注意图3中ALU的零输出端不是与数据存储器连接，该输出与选择PC值来源的多选器的控制有关)

提示：图3中靠右侧的加法器延迟应当按照ALU来计算

**有条件相对跳转指令的关键路径为：**

**指令存储器(IM) -> 寄存器堆(RF)-> 多路选择器(MUX) -> 算术逻辑单元(ALU) -> 多路选择器(MUX)**

**时钟周期为：400+200+30+120+30 = 780ps**

根据下表的两种数据通路的逻辑单元，分别回答下列问题。

|  |  |
| --- | --- |
|  | **单元** |
| **a.** | 执行加4的加法器(对PC) |
| **b.** | 数据存储器 |

(d) 哪些类型的指令需要该单元？

**a. 所有的命令除了跳转指令(jal,jalr,j,jr)。**

**b. LW和RW涉及数据读取存储的指令。**

(e) 对哪些类型的指令而言，该单元位于关键路径上？

**a.没有，因为基本上IM单元占据关键路径。**

**b. LW和RW指令。**

(f) 假设仅需支持beq指令和add指令，讨论该单元的延迟变化对处理器时钟周期的影响。假设其他单元的延迟不变。

**BEQ指令的关键路径为：**

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 多路数据选择器(MUX)**

**时钟周期为：780ps**

**ADD指令的关键路径为：**

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 多路数据选择器(MUX)**

**时钟周期为：780ps**

1. **只要加法器的延迟不超过780ps，BEQ指令、ADD指令就不受影响。两个加法器和一个多选器的延迟时间为230ps,想要影响关键路径，只有加法器超过780-230=550ps的延迟时间才会收到影响。**
2. **BEQ指令和ADD指令都不需要访问数据存储器，所以不受影响。**

1.4 本题讨论数据通路中不同的单元延迟对整个数据通路时钟周期的影响，以及指令如何利用不同的数据通路单元。根据下面的两种延迟情况，分别回答下列问题。15分

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **指令存储器** | **加法器** | **多选器** | **ALU** | **寄存器堆** | **数据存储器** | **符号扩展** | **左移两位** |
| 500ps | 150ps | 100ps | 180ps | 220ps | 1000ps | 90ps | 20ps |

(a) 如果仅需支持ALU类指令(如add、and等)，处理器的时钟周期是多少？

**关键路径为：**

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 多路数据选择器(MUX)**

**时钟周期为：500+220+100+180+100 = 1100ps**

(b) 如果仅需支持lw类指令，时钟周期是多少？

**关键路径为：**

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 数据存储器(DM) ->多路数据选择器(MUX)**

**时钟周期为：500+220+100+180+1000+100 = 2100ps**

(c) 如果必须支持add、beq、lw和sw指令，时钟周期是多少？

**BEQ指令的关键路径为：**

**指令存储器(IM) -> 寄存器堆(RF) -> 多路数据选择器(MUX) -> 算术逻辑单元(ALU) -> 多路数据选择器(MUX)**

**时钟周期：500+220+100+180+100 = 1100ps**

**因此结合上面的计算内容，最长时钟周期为2100ps。**

假设各类型指令所占比例如下表所示，试根据下表的情况分别回答下列问题。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **add** | **addi** | **not** | **beq** | **lw** | **sw** |
| 30% | 15% | 5% | 20% | 20% | 10% |

(d) 数据存储器平均用了多少时钟周期？

**由于只有LW和SW命令执行时候使用数据存储器，所以20% + 10% = 30%**

**1000/2320\*0.3 = 0.129310**

**平均使用0.129310个时钟周期。**

(e) 符号扩展电路的输入平均用了多少时钟周期？在未用到该输入的其他时间，符号扩展电路在做什么？

**符号扩展电路会在ADDI、BEQ、LW、SW指令中使用，所以符号扩展电路的输入**

**15% + 20% + 20% + 10% = 65%**

**500/2320\*0.65 = 0.140086**

**在未用到该输入的其他时间，符号扩展电路也在做符号扩展，因为电路没有控制信号，数据一来就会马上执行。**

(f) 如果可以将数据通路上某个单元的延迟减少10%，应该减少哪个单元的延迟？改进后整个处理器的加速比是多少？

**应该减少数据存储器的延迟，因为数据存储器在所有指令的关键路径上，且延迟时间最大。**

**2100/2000 = 1.050**

1.5本题讨论处理器时钟周期与控制单元设计之间的相互影响。根据下表的数据通路单元延迟情况分别回答下列问题。18分

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **指令存储器** | **加法器** | **多选器** | **ALU** | **寄存器堆** | **数据存储器** | **符号扩展** | **左移两位** | **ALU控制** |
| 500ps | 150ps | 100ps | 180ps | 220ps | 1000ps | 90ps | 20ps | 55ps |

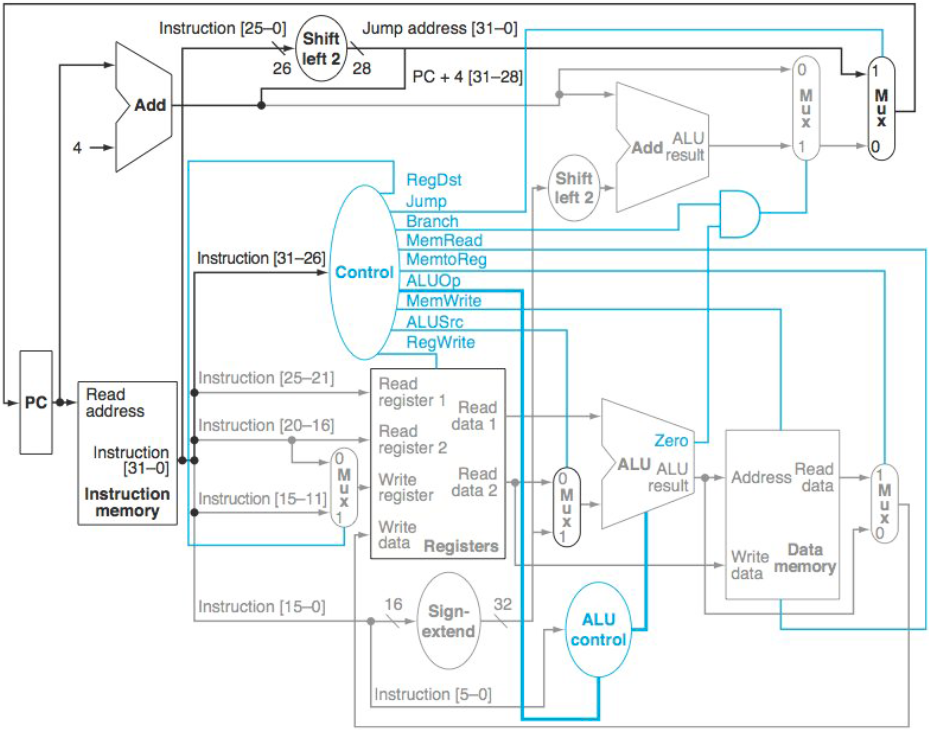


图4

(a) 为了避免增加图4中数据通路的关键路径长度，留给控制单元产生MemWrite信号的时间有多少？

**关键路径为：**

**指令存储器（IM）-> 多路数据选择器（MUX）-> 寄存器堆（RF） -> 多路数据选择器（MUX） -> 算术逻辑单元（ALU） -> 数据存储器（DM）-> 多路数据选择器（MUX）**

**关键路径：500ps + 100ps + 220ps + 100ps + 180ps + 1000ps + 100ps = 2200ps**

**在读取IM后，控制单元才能开始生成MemWrite。 它必须在时钟周期结束之前完成生成该信号：2200-500=1700ps**

(b) 图4中哪个控制信号最不关键，控制单元需要在多长时间内产生该信号以避免其成为关键路径？

**I-Mem读取完成后，所有控制信号开始生成。 信号可能具有的最大松弛时间是在周期结束之前，并且仅在周期结束时才需要MemWrite和RegWrite，因此它们具有最大的松弛时间。**

**生成两个信号而不增加关键路径的时间2200-500=1700ps**

(c) 图4中哪个控制信号最关键，控制单元需要在多长时间内产生该信号以避免其成为关键路径？

**ALUSrc信号最关键**

**500+100+220-500 = 320ps**

**RegDst，Jump和MemtoReg在周期结束之前需要一个Mux延迟，因此它们比MemWrite和RegWrite更为关键。Branch在周期结束之前需要两个Mux延迟，MemRead在周期结束之前需要Mux+DM延迟,ALUOp需要ALU+DM+Mux延迟，ALUSrc需要Mux+ALU+DM+Mux。**

**在ALUOp和ALUSrc之间，如果ALU控制的延迟大于Mux，则ALUOp比ALUSrc更为重要。**

假设控制单元产生控制信号的时间如下表所示，试根据表中的情况回答下列问题(各部件的延迟与前面相同)。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **指令存储器** | **加法器** | **多选器** | **ALU** | **寄存器堆** | **数据存储器** | **符号扩展** | **左移两位** | **ALU控制** |
| 500ps | 150ps | 100ps | 180ps | 220ps | 1000ps | 90ps | 20ps | 55ps |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RegDst** | **Jump** | **Branch** | **MemRead** | **MemtoReg** | **ALUOp** | **MemWrite** | **ALUSrc** | **RegWrite** |
| 720ps | 730ps | 600ps | 400ps | 700ps | 200ps | 710ps | 200ps | 800ps |

(d) 处理器的时钟周期为多少？

**松弛时间计算方法：时钟周期-IM生成信号延迟-控制信号延迟-周期结束前需要的延迟单元。**

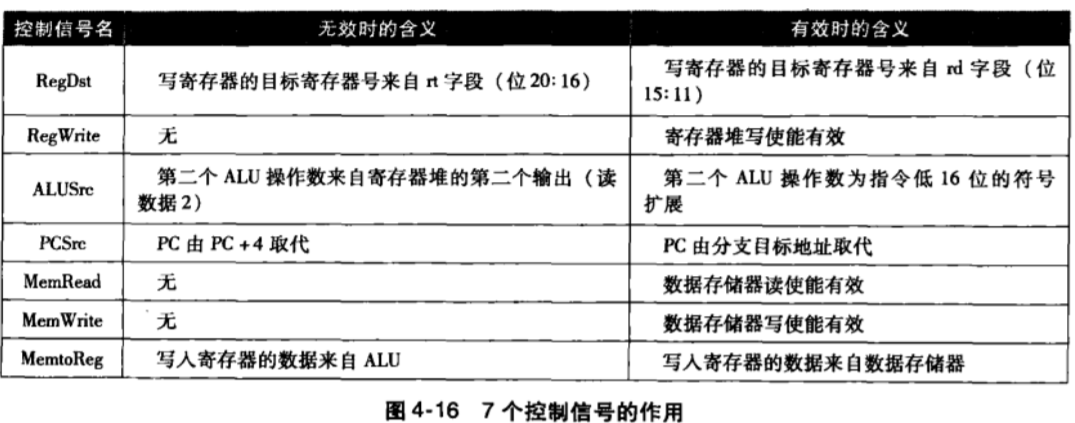
RegDst，Jump和MemtoReg在周期结束之前需要一个Mux延迟，因此它们比MemWrite和RegWrite更为关键。Branch在周期结束之前需要两个Mux延迟，MemRead在周期结束之前需要Mux+DM延迟,ALUOp需要ALU+DM+Mux延迟，ALUSrc需要Mux+ALU+DM+Mux。

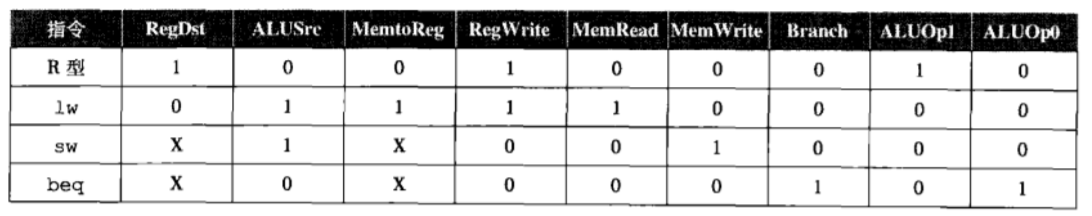
在ALUOp和ALUSrc之间，如果ALU控制的延迟大于Mux，则ALUOp比ALUSrc更为重要。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| RegDst | Jump | Branch | MemRead | MemtoReg | ALUOp | MemWrite | ALUSrc | RegWrite |
| 880ps | 870ps | 900ps | 20ps | 900ps | 200ps | 990ps | 120ps | 900ps |

**综上时钟周期为500ps + 100ps + 220ps + 100ps + 180ps + 1000ps + 100ps = 2200ps。**

**（时钟周期+负松弛值最大值的绝对值）**





(e) 如果你可以加速控制信号的产生，但加快一个控制信号5ps的代价是处理器成本增加1元。那么为了最大化性能你会加速哪些控制信号？这种性能改进的最小代价是多少？

**没有一个信号需要加快。**

**按照上面的设计结果LW指令不需要加快任何控制信号，因为全部的控制信号生成时间都小于各单元需要使用的时间。其次，不是关键路径上面的控制信号改进也不会提升性能，没有必要改进，因此改进最小代价为0。**

(f) 如果一个处理器的成本已经很高，那么我们需要在维持处理器性能的同时降低其成本，而不是像第(e)问中所作的那样为提高它的性能而买单。如果你可以使用更慢的逻辑来实现对信号的控制，并且单个控制信号每减慢5ps，处理其成本就可以节省1元，那么在保持处理器性能的同时，你会减慢哪些控制信号，并且减慢多少来降低成本？

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| RegDst | Jump | Branch | MemRead | MemtoReg | ALUOp | MemWrite | ALUSrc | RegWrite |
| 880ps | 870ps | 900ps | 20ps | 900ps | 200ps | 990ps | 120ps | 900ps |

**总共可节省 5780元**

# MIPS多周期微体系结构分析 25分

1.1假设在多周期MIPS处理器的下列控制信号中存在固定为0缺陷，那么哪些指令将会失效？为什么？(数据通路参考图5，其中不包含j指令；图5中控制器的内部结构如图6 ) 9分

(a) MemtoReg

**LW命令会失效，因为从内存都出的数据无法写入寄存器。**

(b) ALUOp0

**该OP码位为1的指令将变为该Op码位为0的对应OP操作。**

(c) PCSrc

**PC选择指令（J型指令）将会收到影响。**

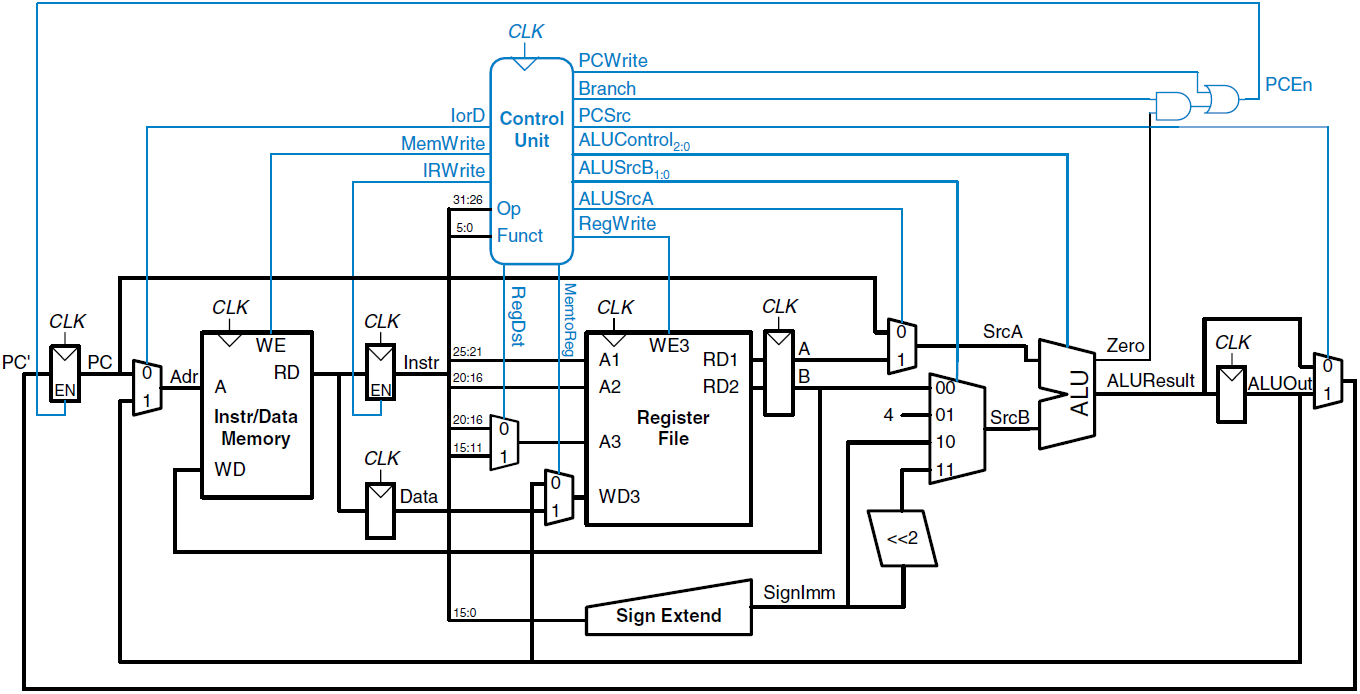


图5

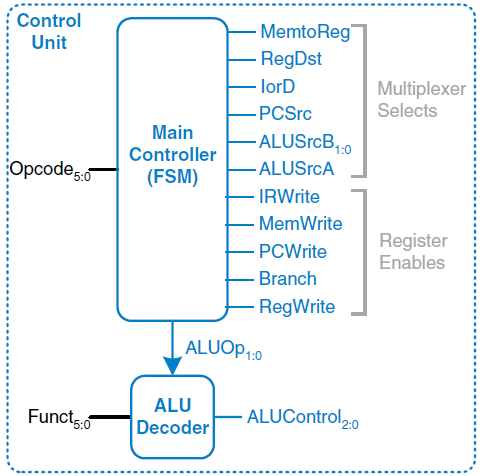


图6

2.2 假设多周期MIPS处理器各个部件的延迟如下表所示(假设存储器和寄存器堆的写速度与读速度相等，数据通路参考图5)。12分

|  |  |  |
| --- | --- | --- |
| **Element** | **Parameter** | **Delay(ps)** |
| **Register clk-to-Q** | tpcq | 30 |
| **Register setup** | tsetup | 20 |
| **Multiplexer** | tmux | 25 |
| **ALU** | tALU | 200 |
| **Memory read** | tmem | 250 |
| **Register file read** | tRFread | 150 |
| **Register file setup** | tRFsetup | 20 |

(a) 通过提高哪个部件的速度(即减小该部件的延迟)可以对整个处理器的速度有最大的提升？

**Memory read**

(b) 在避免不必要浪费的前提之下，该部件的延迟应减小到多少？

**200ps，和ALU的速度相同。**

(c) 提升之后的处理器周期是多少？

**200ps，即最大片段时延。**

(d) 有一种寄存器堆，它比现有的寄存器堆功耗低40%，但是延迟是现有寄存器堆的两倍，请分析一下使用这种寄存器堆是否有意义。

**有意义。不同情况下，计算机的设计需求不一样，有可能这种型号的计算机需要更低功耗的寄存器堆而不考虑延迟的问题，比如嵌入式系统、传感器等系统中的微处理器就要求功耗尽量的低，**

**而对计算机的计算性能可以有所牺牲。**

2.3 在多周期MIPS处理器上运行下面的程序需要多少个周期？这个程序的CPI是多少？ 4分

addi $s0, $0, 5 # sum = 5

while:

beq $s0, $0, done # if result > 0, execute the while block

addi $s0, $s0, -1 # while block: result = result - 1

j while

done:

**CPI为计算机执行一条指令所需的时钟周期数**

**R/I 型指令涉及 IF、ID、EX、WB 四个周期。**

**Branch 型指令涉及 IF、ID、EX 三个周期。**

**J 型指令涉及 IF、ID 两个周期。**

**故指令数目 为：1 + 3\*5 + 1 = 17条**

**总周期数目 为：4 + (4+3+2)\*5 + 3 = 52个**

**则 CPI = 52/17 = 3.058**