高等计算机体系结构，2019 年春季

作业 2： 单周期 vs.多周期微体系结构（参考答案）

主讲教师：栾钟治

助讲教师：杨海龙；助教：许崇杨，左佩璇

作业下发时间：2019 年 3 月 18 日

作业回收时间：2019 年 4 月 1 日

1 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。

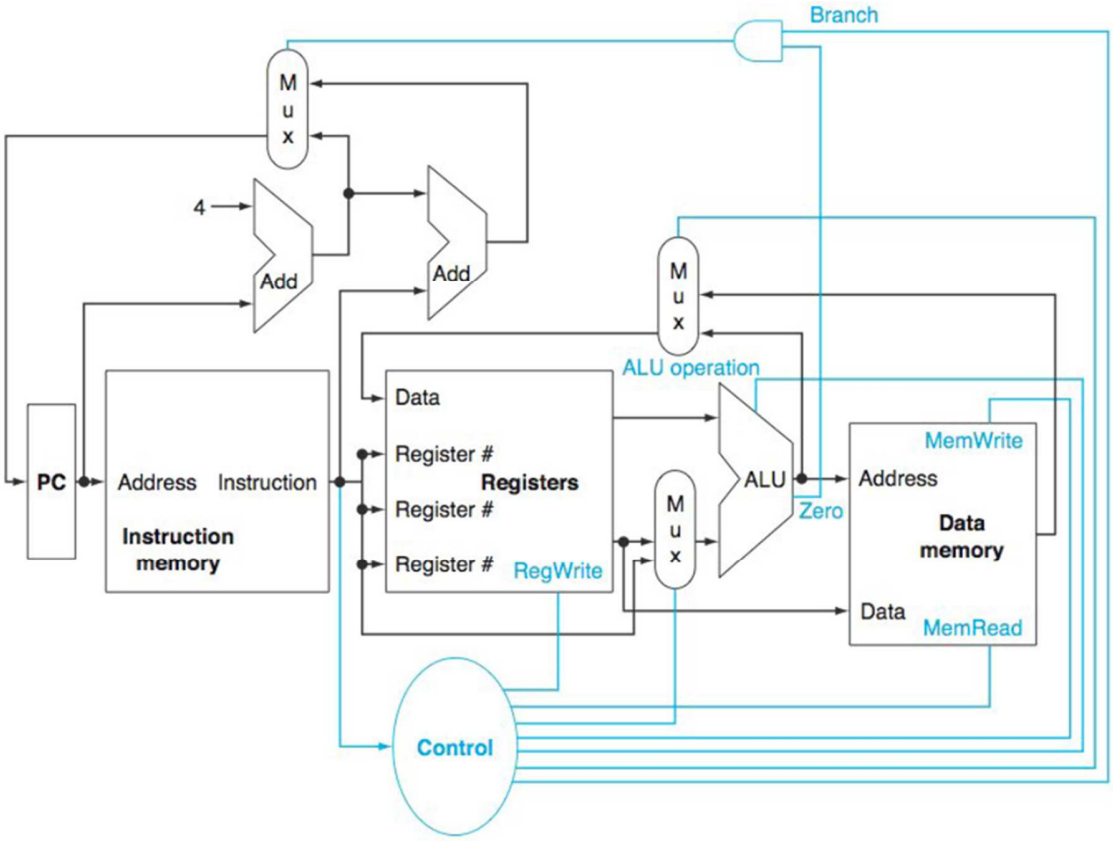

图 1

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
a.	400ps	100ps	30ps	120ps	200ps	350ps	100ps
b.	500ps	150ps	100ps	180ps	220ps	1000ps	65ps

(a) 对一条 MIPS 的与指令(AND)而言，关键路径是什么？

参考答案：
a.关键路径为：I-Mem、Regs(Read)、Mux、ALU、Mux、Regs(Write)
b.关键路径为：I-Mem、Regs(Read)、Mux、ALU、Mux、Regs(Write)
解析：对于 AND 指令(and rd, rs, rt)，存在这样一条长路径：读指令、读寄存器堆、通过 ALUMux 多选器、进行 ALU 运算、通过 RegMux 多选器、写寄存器堆(即 I-Mem、Regs(Read)、Mux、ALU、Mux、Regs(Write))。另外一条长路径与之类似，但是这条路径是在寄存器堆进行读操作时通过控制器的，即：I-Mem、Control、Mux、ALU、Mux、Regs(Write)，但由于控制器的速度快于寄存器堆，因而前者为关键路径，其它的路径都短于这两条路径。

(b) 对一条 MIPS 的装载指令(LW)而言，关键路径是什么？

参考答案：
a.关键路径为：I-Mem、Regs(Read)、Mux、ALU、D-Mem(Read)、Mux、Regs(Write)
b.关键路径为：I-Mem、Regs(Read)、Mux、ALU、D-Mem(Read)、Mux、Regs(Write)
解析：对于 LW 指令，存在这样一条长路径：读指令、读寄存器堆获得基址、使用多选器选择立即数作为 ALU 的输入、使用 ALU 计算地址、访问数据存储器、使用多选器选择存储器输出作为寄存器的数据输入、写寄存器堆，故有路径 I-Mem、Regs(Read)、Mux、ALU、D-Mem(Read)、Mux、Regs(Write)。还有一条与之类似的长路径，但是这条路径是通过控制器而不是寄存器堆的 (用于生成 ALUMux 的控制信号)，由于控制器的速度快于寄存器堆，于是前者为关键路径，除这两条之外的路径都是比较短的路径。

(c) 对一条 MIPS 的相等则分支指令(BEQ)而言，关键路径是什么？

参考答案：
a.由于控制器的速度快于寄存器堆，故关键路径为：I-Mem、Regs(Read)、Mux、ALU、Mux
b.由于控制器的速度快于寄存器堆，故关键路径为：I-Mem、Regs(Read)、Mux、ALU、Mux
解析：这条指令有两种长路径——决定分支条件以及计算新 PC 值。对于决定分支条件的路径，需要读指令、读寄存器堆或使用控制单元、使用 ALUMux、使用 ALU 比较两个值、使用 ALU 的零输出端来控制选择新 PC 值的多选器。对于计算新 PC 值的路径，其中一条是 PC 值加 4(Add)、加偏移量 offset(Add)、选择这个值作为新的 PC 值(Mux)；另一条是读指令(为了取得偏移量)、使用分支加法单元和相应的多选器。但是这两条计算 PC 值中的路径都比决定分支条件的路径要短，这是因为从表中可以看到指令存储器的速度要慢于执行 PC+4 的加法器、ALU 的速度要慢于分支加法器。

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
a.	add3 Rd,Rs,Rt,Rx	Reg[Rd]=Reg[Rs]+Reg[Rt]+Reg[Rx]
b.	sll Rt,Rd,Shift	Reg[Rd]=Reg[Rt]<<Shift(左移)

(a) 对上述指令而言，哪些已有的单元还可以被使用？

参考答案：
a. 除分支加法器、数据存储器之外的所有单元
b. 除分支加法器、数据存储器之外的所有单元，此外，ALU 可以选择继续使用也可以选择不继续使用，具体要视第②问的回答，如果增加一个专门的移位器，则不需要使用 ALU

(b) 对上述指令而言，还需要增加哪些功能单元？

参考答案：
a.寄存器堆增加一个输出端 和一个相应的读地址输入端，以读出 Rx；增加一个加法器(或为现有的 ALU 增加一个输入端)，加法器的一个输入端连接至寄存器堆新增的输出端，另一个输入端连接至 ALU 的输出端。如果采用增加一个加法器的方案，则还需增加寄存器堆数据输入选通器的一个输入端，并连接至加法器的输出端
b.增加一个移位器(或为现有的 ALU 增加移位运算功能)，移位器的输入端连接至寄存器堆的一个输出端。如果采用增加一个移位器的方案，则还需增加寄存器堆数据输入选通器的一个输入端，并连接至移位器的输出端

(c) 为了支持这些指令，需要在控制单元增加哪些信号？

参考答案：
a.如果增加一个加法器，则需增加寄存器堆数据输入选通器的控制信号，以实现数据通道 3 选 1；如果为现有的 ALU 增加一个输入端，则需增加针对三输入端的 ALU 的功能控制信号定义，使其可控制新增的 ADD3 操作
b.如果增加一个移位器，则需增加寄存器堆数据输入选通器的控制信号，以实现数据通道 3 选 1(如同时考虑 a 和 b 中的两条指令，则为 4 选 1)；如为现有的 ALU 增加移位运算功能，则需增加 ALU 的功能控制信号定义，使其可控制新增的移位操作

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？

参考答案：
a. 由于加法单元不在关键路径上，因此对加法器的改进不影响时钟周期。
b. 寄存器堆位于关键路径上，因而，使用更大的寄存器堆后，时钟周期变为 1330ps + 2 × 100ps = 1530ps。
解析：时钟周期是由关键路径决定的，这里的关键路径为：I-Mem(读指令)、Regs(Read)(由于寄存器堆的延迟大于控制器，因而寄存器堆位于关键路径上)、Mux(选择 ALU 的输入)、ALU、Data Memory(Read)、Mux(选择存储器写入到寄存器堆中的数据)、Regs(Write)(数据写入寄存器堆)，该路径的延迟为 400ps + 200ps + 30ps + 120ps + 350ps + 30ps + 200ps = 1330ps。

(e) 改进后将获得多大的加速比？

参考答案：
a.加速比由时钟周期本身的变化以及需要执行的时钟周期数目共同决定，对加法器的改进不影响时钟周期，并且，需要执行的时钟周期数目也不变，因此加速比为 1.000
b.需要的指令数减少 5%，需要的时钟周期数目也相应减少 5%，同时，时钟周期由 1330ps 增加为 1530ps，因而加速比为 (1/0.95)×(1330/1530) = 0.915

(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？

参考答案：
a. 原来的处理器的总成本为 1000(I-Mem) + 200(Regs) + 500(Control) + 100(ALU) + 2000(D-Mem) + 2×30(2 个加法单元) + 3×10(3 个多选器) = 3890，更换加法器之后的总成本为 3890 + 2 × 20 = 3930，相对成本为 3930 / 3890 = 1.010，性能/价格比为 1.000 / 1.010 = 0.990，成本增加但性能没有提升。
b. 使用更大的寄存器堆的成本为 3890 + 200 = 4090，相对成本为 4090 / 3890 = 1.051，性能/价格比为 0.915 / 1.051 = 0.871，说明用更大的投入反而换来了性能的下降。

1.3 下表给出了实现处理器数据通路的逻辑单元延迟。试根据下表两种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**
a.	400ps	100ps	30ps	120ps	200ps	350ps	20ps	2ps
b.	500ps	150ps	100ps	180ps	220ps	1000ps	90ps	20ps

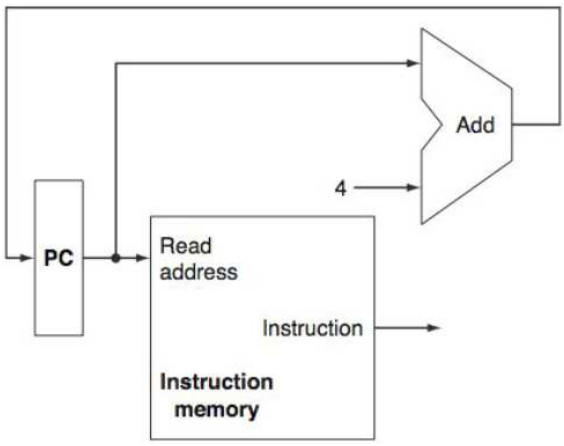

图 2

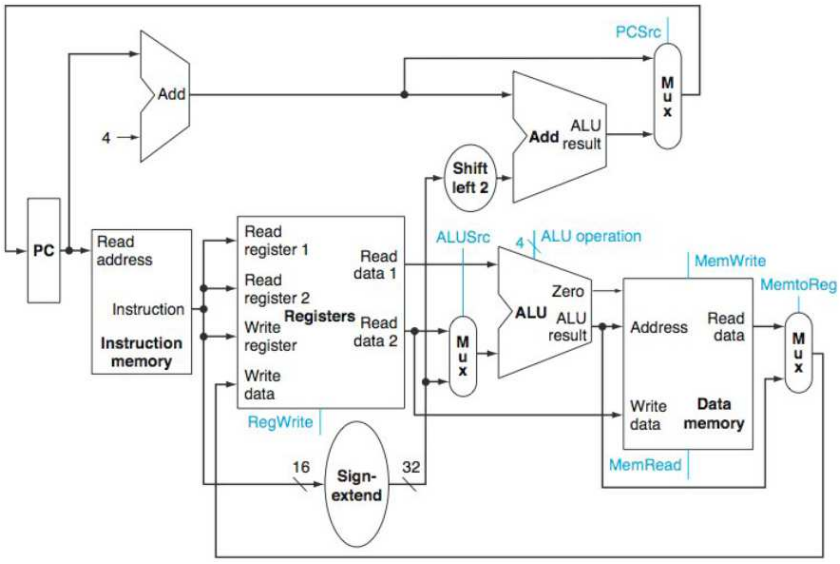

图 3

(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？

参考答案：

a. 由于指令存储器慢于加法器，因此，时钟周期决定于指令存储器的延迟，时钟周期为 400ps。

b. 时钟周期为 500ps。

(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？

参考答案：

a.关键路径为 I-Mem、Sign-extend、Shift-left-2、Add(ALU)、Mux，因此，时钟周期为 400ps + 20ps + 2ps + 120ps + 30ps = 572ps。

b.时钟周期为 500ps + 90ps + 20ps + 180ps + 100ps = 890ps。

(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

参考答案：
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)
a.根据题目中给出的延迟，后者为关键路径，因此时钟周期为 400ps + 200ps + 30ps + 120ps + 30ps = 780ps。
b.根据题目中给出的延迟，后者为关键路径，因此时钟周期为 500ps + 220ps + 100ps + 180ps + 100ps = 1100ps。
解析：对于有条件相对跳转指令，除存在长路径 I-Mem、Sign-extend、Shift-left-2、Add(ALU)、Mux 外，还存在长路径 I-Mem、Registers(Read)、Mux、ALU、Mux，关键路径为这两条路径中较长的一个。

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？

参考答案：
a.所有指令。
b.与存取有关的指令，如：Lw, Sw 等。

(e) 对哪些类型的指令而言，该单元位于关键路径上？

参考答案：
a.所有指令的关键路径都不会包含这个加法器，因为指令存储器的速度慢于加法器，而所有的指令都必须执行读指令这一操作。
b.与存取有关的指令，因为只有与存取有关的指令会用到该单元。

(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。

参考答案：
a. beq 指令的关键路径为 I-Mem、Regs(Read)、Mux、ALU、Mux，延迟为 780ps，add 指令的关键路径为 I-Mem、Regs(Read)、Mux、ALU、Mux、Regs(Write)，延迟为 980ps。这里只需讨论该单元延迟变化相比于较长的关键路径的影响，较长的关键路径为 add 指令，其延迟为 980ps，而执行加 4 的加法器所在的路径为 Add、Add(ALU)、Mux，其延迟为 100ps + 120ps + 30ps = 250ps，若要该单

元延迟变化而导致该路径成为关键路径，从而影响时钟周期，则执行加 4 的加法器延迟要大于 980ps － 150ps = 830ps，才会影响时钟周期。
b. 数据存储器未被 beq 或 add 指令使用，因此，它的延迟不影响时钟周期。

	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
a.	400ps	100ps	30ps	120ps	200ps	350ps	20ps	2ps

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
a.	400ps	100ps	30ps	120ps	200ps	350ps	20ps	0ps
b.	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？

参考答案：
a.时钟周期为 400ps + 200ps + 30ps + 120ps + 30ps + 200ps = 980ps
b.时钟周期为 500ps + 220ps + 100ps + 180ps + 100ps + 220ps = 1320ps
解析：关键路径为 I-Mem、Registers(Read)、Mux(选择 ALU 输入)、ALU、Mux(选择寄存器写入端)、Registers(Write)

(b) 如果仅需支持 lw 类指令，时钟周期是多少？

参考答案：
a.时钟周期为 400ps + 200ps + 30ps + 120ps + 350ps + 30ps + 200ps = 1330ps
b.时钟周期为 500ps + 220ps + 100ps + 180ps + 1000ps + 100ps + 220ps = 2320ps
解析：关键路径为 I-Mem、Registers(Read)、Mux(选择 ALU 输入)、ALU、D-Mem(Read)、Mux(选择写入寄存器堆的数据)、Registers(Write)

(c) 如果必须支持 add、beq、lw 和 sw 指令，时钟周期是多少？

参考答案：
a. 时钟周期为 1330ps
b. 时钟周期为 2320ps
解析：lw 指令的关键路径最长，相比较而言，sw 指令少使用了一个多选器并且不用向寄存器堆写数据，add 和 beq 少使用了数据存储器

假设各类型指令所占比例如下表所示，试根据下表的两种情况分别回答下列问题。

	add	addi	not	beq	lw	sw
a.	30%	15%	5%	20%	20%	10%
b.	25%	5%	5%	15%	35%	15%

(d) 数据存储器平均用了多少时钟周期?

参考答案:
a.平均有 20% + 10% = 30% 的时钟周期里，会用到数据存储器
b.平均有 35% + 15% = 50% 的时钟周期里，会用到数据存储器
解析：只有 lw 和 sw 指令会用到数据存储器

(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？

参考答案:
符号扩展电路实际上在每个周期都有计算结果，但是它的输出在 add 和 not 指令中被忽略了，符号扩展电路的输入只在 addi 指令(提供 ALU 需要的立即数)、beq 指令(提供计算 PC 需要的偏移量)、lw 指令和 sw 指令(提供寻址过程中需要的偏移量)中是需要的
a. 结果为 15% + 20% + 20% + 10% = 65%
b. 结果为 5% + 15% + 35% + 15% = 70%

(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？

参考答案:
lw 指令有最长的关键路径为：I-Mem、Registers(Read)、Mux、D-Mem(Read)、ALU、Mux、Registers(Write)，决定着时钟周期的长度。
a.在路径中，由于指令存储器的延迟值最大，因此，如将它的延迟从 400ps 减小到 360ps(即减少 10%)，则时钟周期由 1330ps 减小到 1290ps，加速比为 1330/1290=1.031。
b.在路径中，由于数据存储器的延迟值最大，因此，如将它的延迟从 1000ps 减小到 900ps(即减少 10%)，则时钟周期由 2320ps 减小到 2220ps，于是加速比为 2320/2220=1.045。

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。根据下表的两种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
a.	400ps	100ps	30ps	120ps	200ps	350ps	20ps	0ps	50ps
b.	500ps	150ps	100ps	180ps	220ps	1000ps	90ps	20ps	55ps

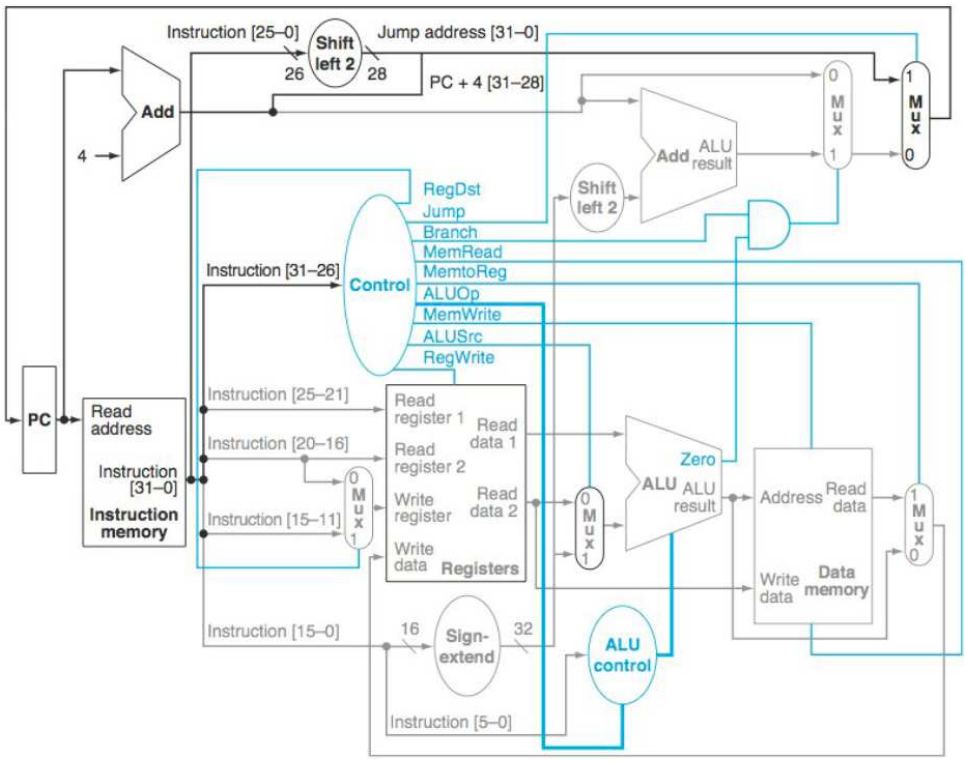

图 4

(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？

参考答案：

a.关键路径延迟为 400ps + 200ps + 30ps + 120ps + 350ps + 30ps + 200ps = 1330ps，留给控制单元产生 MemWrite 信号的最长时间为 1330ps – 400ps – 350ps = 580ps

b.关键路径延迟为 500ps + 220ps + 100ps + 180ps + 1000ps + 100ps + 220ps = 2320ps，留给控制单元产生 MemWrite 信号的最长时间为 2320ps – 500ps – 1000ps = 820ps

解析：假设控制单元的延迟为 0，则 lw 具有最长的关键路径，于是得到关键路径 I-Mem、Regs(Read)、Mux、ALU、D-Mem(Read)、Mux、Regs(Write)。控制单元在读取完指令存储器之后才可以产生 MemWrite 控制信号，并且必须在时钟周期结束之前产生 MemWrite 信号，但是，由于 MemWrite 信号是数据存储器的写使能信号，因此在产生该信号之后还应当至少留出写存储器的时间 D-Mem(Write)。

(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？

参考答案：

a.Jump 信号具有最长的松弛时间，为 1330ps – 400ps – 30ps = 900ps

b.Jump 信号具有最长的松弛时间，为 2320ps – 500ps – 100ps = 1720ps

解析：所有的控制信号都必须在指令读取之后生成，同时一个信号最晚必须在时钟周期结束之前到来，对于 MemWrite、RegWrite 和 Jump 信号，由于它们都只在时钟周期的最后才会用到，因而相比于其它控制信号会拥有更长的松弛时间，由于两种情况下均是数据存储器的延迟>寄存器堆>多选器，因而 Jump 具有最长的松弛时间。

这个题目里面没有考虑 PC 的延迟。

(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？

参考答案：

	最关键信号	产生该信号可用的时间
a.	ALUOp (50ps > 30ps)	200ps + 30ps - 50ps = 180ps
b.	ALUSrc (100ps > 55ps)	220ps

解析：为了不影响关键路径，控制信号必须在数据到达之前产生以便不影响数据的通过，最早出现在关键路径上的控制信号是 ALUOp 和 ALUSrc，ALUSrc 的松弛时间为 Regs(Read)，ALUOp 的松弛时间为 Regs(Read) + Mux - ALU Ctrl，拥有较小松弛时间的信号更为关键，可见二者对于 ALU 计算的影响取决于 ALU Ctrl 与 Mux 的延迟大小。

假设控制单元产生控制信号的时间如下表所示，试根据表中的两种情况回答下列问题(各部件的延迟与前面相同)。

	RegDst	Jump	Branch	MemRead	MemtoReg	ALUOp	MemWrite	ALUSrc	RegWrite
a.	720ps	730ps	600ps	400ps	700ps	200ps	710ps	200ps	800ps
b.	1600ps	1600ps	1400ps	500ps	1400ps	400ps	1500ps	400ps	1700ps

(d) 处理器的时钟周期为多少？

参考答案：

	对时钟周期影响最大的控制信号	理想的时钟周期(由第(a)问得来)	实际的时钟周期
a.	MemWrite (+130ps)	1330ps	1460ps
b.	MemWrite (+680ps)	2320ps	3000ps

解析：为了便于后面的计算，我们首先计算各个控制信号的松弛时间，如下表所示：

	RegDst	Jump	Branch	MemRead	MemtoReg	ALUOp	MemWrite	ALUSrc	RegWrite
a	700ps	900ps	870ps	350ps	700ps	180ps	580ps	200ps	730ps
b	1500ps	1720ps	1620ps	500ps	1500ps	265ps	820ps	220ps	1600ps

若实际产生信号的时间小于松弛时间，则该信号的产生不会影响时钟周期，反之，该信号会影响时钟周期，并且显然是会使时钟周期变大，由此可以计算出新的时钟周期(下表中的数据为实际时间减去松弛时间)。

	RegDst	Jump	Branch	MemRead	MemtoReg	ALUOp	MemWrite	ALUSrc	RegWrite
a.	20ps	-170ps	-270ps	50ps	0ps	20ps	130ps	0ps	70ps
b.	100ps	-120ps	-220ps	0ps	-100ps	135ps	680ps	180ps	100ps

(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？

参考答案：

	对时钟周期有影响的信号	代价

a.	RegDst (+20ps) MemRead (+50ps) ALUOp (+20ps) MemWrite (+130ps) RegWrite (+70ps)	290/5=58
b.	RegDst (+100ps) ALUOp (+135ps) MemWrite (+680ps) ALUSrc (+180ps) RegWrite (+100ps)	1195/5=239

解析：应当只加速影响了时钟周期的控制信号，目标是将这些控制信号对时钟周期的影响至少变为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？

参考答案：

	RegDst	Jump	Branch	MemRead	MemtoReg	ALUOp	MemWrite	ALUSrc	RegWrite
a.	20ps	-170ps	-270ps	50ps	0ps	20ps	130ps	0ps	70ps
b.	100ps	-120ps	-220ps	0ps	-100ps	135ps	680ps	180ps	100ps

上表中具有最大正值的信号决定了周期，则在保证性能的前提之下，可以将所有的信号都减慢到具有跟该信号相同的值，于是最小化成本的方法是按照下表减慢响应信号的速度：

	RegDst	Jump	Branch	MemRead	MemtoReg	ALUOp	MemWrite	ALUSrc	RegWrite
a.	110ps	300ps	400ps	80ps	130ps	110ps	0ps	130ps	60ps
b.	580ps	800ps	900ps	680ps	780ps	545ps	0ps	500ps	580ps

2 MIPS 多周期微体系结构分析 25 分

2.1 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

参考答案：
lw 指令将会失效，MemtoReg 固定为 0 将导致无法选择 DR 作为寄存器堆写数据的来源，而 lw 指令需要将 DR 中的数据写入寄存器堆。

(b) ALUOp0

参考答案：
beq 指令将会失效，ALUOp0 固定为 0 将导致 ALU 无法进行 beq 需要的减法运算(相应的 ALUOp 为 01)。

(c) PCSrc

参考答案：

beq 指令将会失效，PCSrc 固定为 0 将导致无法选择 ALUOut 的输出作为 PC 值的来源，而 beq 指令需要保存在 ALUOut 中的 ALU 的运算结果来修改 PC 的值。

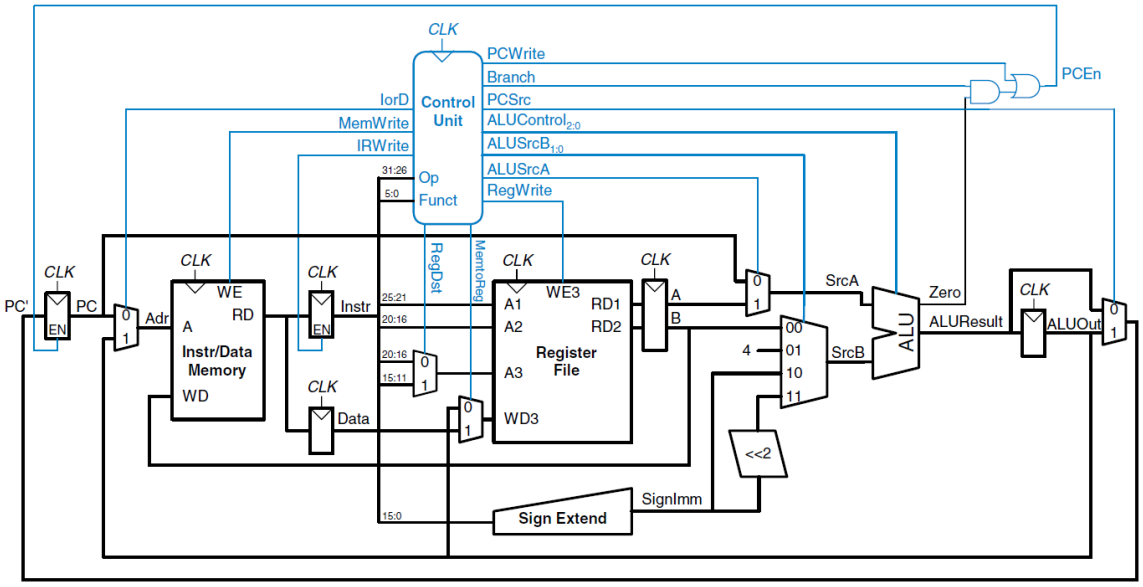

图 5

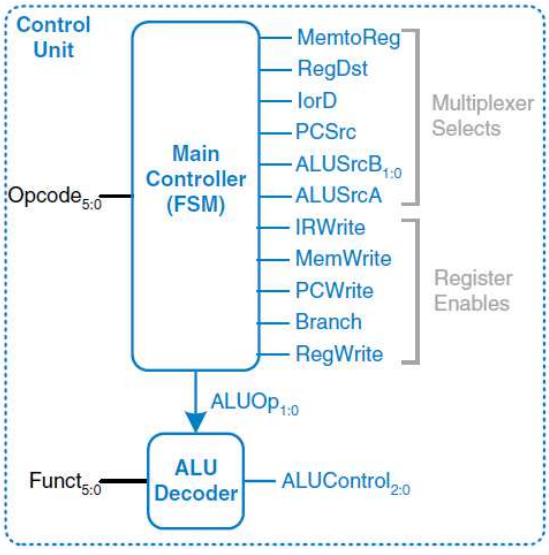

图 6

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	t_{pcq}	30

Register setup	t_{setup}	20
Multiplexer	t_{mux}	25
ALU	t_{ALU}	200
Memory read	t_{mem}	250
Register file read	t_{RFread}	150
Register file setup	$t_{RFsetup}$	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？

参考答案：
由于寄存器堆的速度快于存储器(tRFread < tmem)且 tsetup = tRFsetup，于是有 Tc = tpcq + tmux + max(tALU + tmux, tmem) + tsetup，由表中数据可得 tALU + tmux = 225ps < tmem，于是为了减小时钟周期，应当提高存储器的速度。

(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？

参考答案：
应将存储器的延迟减小到 225ps。

(c) 提升之后的处理器周期是多少？

参考答案：
原来处理器的时钟周期为 30ps + 25ps + 250ps + 20ps = 325ps，提升存储器速度之后处理器的时钟周期为 30ps + 25ps + 225ps + 20ps = 300ps。

(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。

参考答案：
寄存器堆的延迟变为 tRFread=300ps
考虑与寄存器堆有关的两个时间，一个是读寄存器堆操作，时间为 30ps + 300ps + 20ps = 350ps；另一个是写寄存器堆操作，时间为 30ps + 25ps + 300ps + 20ps = 375ps。而原来的时钟周期为 325ps，于是处理器的时钟周期将变为 375ps，计算性能/功耗比为(325/375)/0.6 = 1.44，说明功耗的下降幅度大于性能的下降幅度，使用这种寄存器堆是有一定意义的。

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:

参考答案：
addi 和 beq 都是 I 型指令，j 是跳转指令，addi 需要 4 个时钟周期，beq 需要 3 个时钟周期，j 需要 3 个时钟周期。对程序进行分析可知，第一个 addi 指令执行了 1 遍，beq 指令执行了 6 遍(跳出循环时判断条件不成立也执行了一遍)，第二个 addi 指令执行了 5 遍，j 指令执行了 5 遍，于是共需要 4 + 6×3 + 5×4 + 5×3 = 57 个周期。
CPI 为 57 / (1+6+5+5) = 3.35。