7.2 选择题

（1）[2013] 某 CPU 主频为 1.03GHz，采用 4 级指令流水线，每个流水段的执行需要 1 个时钟周期。假定 CPU 执行了 100 条指令，在其执行过程中，没有发生任何流水线阻塞，此时流水线的吞吐率为（C）。

A．$0.25\times10_9$ 条指令 / 秒 B．$0.97\times10_9$ 条指令 / 秒

C．$1.0\times10_9$ 条指令 / 秒 D．$1.03\times10_9$ 条指令 / 秒

分析：采用 4 级指令流水线执行 100 条指令，需要 4 + (n - 1) = 4 + (100 - 1) = 103 个时钟周期 T。流水线吞吐率指每秒执行指令条数，即 100 / 103T = f * 100 / 103 = 1.03GHz * 100 / 103 = 1 * 10^9 条指令/秒，选 C

（2）[2009] 某计算机的指令流水线由 4 个功能段组成，指令流经各功能段的时间（忽略各功能段之间的缓存时间）分别为 90ns、80ns、70ns 和 60ns，则该计算机的 CPU 时钟周期至少是（A） 。

A．90ns B．80ns C．70ns D．60ns

分析：时钟周期应该以各功能段的最长执行时间为准，选 A

（3）[2018] 若某计算机最复杂指令的执行需要完成 5 个子功能，分别由功能部件 A ～ E 实现，各功能部件所需时间分别为 80ps、50ps、50ps、70ps 和 50ps，采用流水线方式执行指令，流水段寄存器延迟时间为 20ps，则 CPU 时钟周期至少为（D） 。

A．60ps B．70ps C．80ps D．100ps

分析：时钟周期应该以各功能段的最长执行时间为准，还要加上流水段寄存器延时时长，选 D

（4）[2016] 在无转发机制的 5 段基本流水线中，下列指令序列存在数据冲突的指令对是（B） 。

I1: ADD R1, R2, R3; (R2)+(R3) → R1 I2: ADD R5, R2, R4; (R2)+(R4) → R5

I3: ADD R4, R5, R3; (R5)+(R3) → R4 I4: ADD R5, R2, R6; (R2)+(R6) → R5

A．I1 和 I2 B．I2 和 I3 C．I2 和 I4 D．I3 和 I4

分析：数据冲突是指当前指令要用到先前指令的操作结果，而这个结果尚未产生或尚未送达指定位置，会导致当前指令无法继续执行。指令 I3 的源寄存器 R5 需要等待 I2 指令中结果写回后才能取正确的寄存器值，存在数据相关，可能存在数据冲突，选 B

（5）[2019] 在采用“取指、译码 / 取数、执行、访存、写回”5 段流水线的处理器中，执行如下指令序列， 其中 s0、s1、s2、s3 和 t2 表示寄存器编号。

I1: add s2, s1, s0 // R[s2] ← R[s1]+R[s0]

I2: load s3,0(t2) // R[s3] ← M[R[t2]+0]

I3: add s2, s2, s3 // R[s2] ← R[s2]+R[s3]

I4: store s2,0(t2) // M[R[t2]+0] ← R[s2]

下列指令对中，不存在数据冒险的是（C） 。

A．I1 和 I3 B．I2 和 I3 C．I2 和 I4 D．I3 和 I4

分析：I3 的源寄存器 S2 为 I1 的目标寄存器，存在数据冒险。I3 的源寄存器 S3 为 I2 的目标寄存器，存在数据冒险。I4 的源寄存器 S2 是 I3 的目标寄存器，存在数据冒险。只有 I2 和 I4 没有数据冒险，I2 是访存，I4 是写入，并不冲突，所以选 C

（6）[2010] 下列选项中，不会引起指令流水线阻塞的是（A） 。

A．数据旁路（转发） B．数据相关 C．条件转移 D．资源冲突

分析：引起流水线阻塞的原因有：资源冲突、数据冲突、控制相关。数据旁路技术是为了解决数据相关，直接将执行结果送到其他指令所需要的地方，使流水线发生停顿，因此不会引起流水线阻塞，选 A

（7）[2011] 下列给出的指令系统特点中，有利于实现指令流水线的是（D） 。

Ⅰ．指令格式规整且长度一致

Ⅱ．指令和数据按边界对齐存放

Ⅲ．只有 Load/Store 指令才能对操作数进行存储访问

A．仅Ⅰ、Ⅱ B．仅Ⅱ、Ⅲ C．仅Ⅰ、Ⅲ D．Ⅰ、Ⅱ、Ⅲ

分析：指令定长、指令和数据对齐、采用 Load/Store 架构均能有效地简化流水复杂度，选 D

（8）[2017] 下列关于指令流水线数据通路的叙述中，错误的是（A） 。

A．包含生成控制信号的控制部件

B．包含算术逻辑运算部件（ALU）

C．包含通用寄存器组和取指部件

D．由组合逻辑电路和时序逻辑电路组合而成

分析：数据通路中包括程序计数器、指令存储器、流水接口寄存器、通用寄存器组、ALU、数据存储器等，不包括控制部件，选 A

（9）[2017] 下列关于超标量流水线特性的叙述中，正确的是（C） 。

Ⅰ．能缩短流水线功能段的处理时间

Ⅱ．能在一个时钟周期内同时发射多条指令

Ⅲ．能结合动态调度技术提高指令执行并行性

A．仅Ⅱ B．仅Ⅰ、Ⅲ C．仅Ⅱ、Ⅲ D．Ⅰ、Ⅱ和Ⅲ

分析：超标量指 CPU 中有多条流水线，每个时钟周期内可以完成多条指令。这种做法并不能缩短流水线功能段地处理时间，选 C

（10）[2020] 下列给出的处理器类型中，理想情况下 CPI 为 1 的是（B） 。

Ⅰ．单周期 CPU Ⅱ．多周期 CPU Ⅲ．基本流水线 CPU Ⅳ．超标量流水线 CPU

A．Ⅰ和Ⅱ B．Ⅰ和Ⅲ C．Ⅰ、Ⅲ、Ⅳ D．Ⅲ、Ⅳ

分析：单周期 CPU 和基本流水线 CPU 地理想 CPI 为 1.多周期 CPU 的 CPI 大于 1，超标量流水线 CPU 的 CPI 小于 1，选 B

7.8 如果采用气泡流水线执行下述程序，请给出类似图 7.18 所示的流水线时空图。注意时空图中最后一个时钟周期第 5 条指令进入 ID 段。

```
addi $s0, $s0, 4
lw   $s1, ($s0)
add  $s2, $s2, $s1
and  $s3, $s1, $s2
sub  $s4, $s2, $s2
```

CLKs	取指令IF	译码ID	执行EX	访存MEM	写回WB
1	and $1, $1, $2				
2	sub $2, $1, $0	and $1, $1, $2			
3	add $3, $1, $1	sub $2, **$1**, $0	and **$1**, $1, $2		
4	add $3, $1, $1	sub $2, **$1**, $0		and **$1**, $1, $2	
5	add $3, $1, $1	sub $2, $1, $0			and $1, $1, $2
6	or $4, $5, $1	add $3, $5, $1	sub $2, $1, $0		

图 7.18 插入气泡解决数据相关的流水线时空图

答：

CLKs	取指IF	译码ID	执行EX	访存MEM	写回WB
1	addi $s0, $s0, 4				
2	lw $s1, ($s0)	addi $s0, $s0, 4			
3	add $s2, $s2, $s1	lw $s1, ($s0)	addi $s0, $s0, 4		
4	add $s2, $s2, $s1	lw $s1, ($s0)		addi $s0, $s0, 4	
5	add $s2, $s2, $s1	lw $s1, ($s0)			addi $s0, $s0, 4
6	and $s3, $s1, $s2	add $s2, $s2, $s1	lw $s1, ($s0)		
7	and $s3, $s1, $s2	add $s2, $s2, $s1		lw $s1, ($s0)	
8	sub $s4, $s2, $s2	add $s2, $s2, $s1			lw $s1, ($s0)
9		sub $s4, $s2, $s2	add $s2, $s2, $s1		

7.9 如果采用重定向流水线执行 7.8 中程序，请给出类似图 7.18 所示的流水线时空图。注意时空图中最后一个时钟周期第 5 条指令进入 ID 段。

答：

CLKs	取指IF	译码ID	执行EX	访存MEM	写回WB
1	addi $s0,$s0,4				
2	lw $s1,($s0)	addi $s0,$s0,4			
3.	add $s2,$s2,$s1	lw $s1,($s0)	addi $s0,$s0,4		
4.	and $s3,$s1,$s2	add $s2,$s2,$s1	lw $s1,($s0)	addi $s0,$s0,4	
5.	and $s3,$s1,$s2	add $s2,$s2,$s1		lw $s1,($s0)	addi $s0,$s0,4
6.	sub $s4,$s2,$s2	and $s3,$s1,$s2	add $s2,$s2,$s1		lw $s1,($s0)
7.		sub $s4,$s2,$s2	and $s3,$s1,$s2	add $s2,$s2,$s1	

7.10 假设重定向流水线中所有分支跳转指令均在 EX 段执行，无分支预测、无分支延迟槽技术，尝试计算下述程序的执行周期。

```
add $s0,$0,100              #  i=100
while_loop:
    beq $s0,$0,done         # while (i>0)
    addi $s0,$s0,-1         #  i=i-1
    j while_loop            # 继续循环
done:
```

答：beq 指令判断 i 是否为 0，只有最后一次才会跳转，此时因为顺序预取会损失两个时钟周期，j 指令无条件跳转，每次跳转会因为顺序预取损失两个时钟周期。循环体中 3 条指令重复执行 100 次，循环退出额外执行了以此 beq 指令，所以需要执行 1 + 100 * 3 + 1 = 302 条指令，而 j 指令造成流水线停顿 100 * 2 = 200 个时钟周期，

因此程序执行周期数 = 5 + (n - 1) + 200 + 2 = 5 + (302 - 1) + 200 + 2 = 508

7.12 假设重定向流水线中所有分支跳转指令均在 EX 段执行，设无分支预测、无分支延迟槽技术，尝试计算下述程序的执行周期。

```
addi $s0,$0,0              # i=0
addi $s1,$0,0              # sum=0
for_loop:
   slti $t1,$s0,10         #  $t1=(i<10)?1:0

   beq $t1,$0,done         #  for(i=0;i<10;i++)
   addi $s1,$s1,$s0        #  sum=sum+i
   addi $s0,$s0,1          #  i++
   j for_loop              #  继续 for 循环
done:
```

答：beq 指令判断 i 是否为 0，只有最后一次才会跳转，此时因为顺序预取会损失两个时钟周期，j 指令无条件跳转，每次跳转会因为顺序预取损失两个时钟周期。循环体中 5 条指令重复执行 10 次，循环退出额外执行了 slti 和 beq 两条指令，所以需要执行 2 + 10 * 5 + 2 = 54 条指令，而 j 指令造成流水线停顿 10 * 2 = 20 个时钟周期，
因此程序执行周期数 = 5 + (n - 1) + 20 + 2 = 5 + (54 - 1) + 22 = 80

7.15 某 16 位计算机中，带符号整数用补码表示，数据 cache 和指令 cache 分离。表 7.8 中给出了指令系统中的部分指令格式，其中 Rs 和 Rd 表示寄存器，mem 表示存储单元地址。另外，（x）表示寄存器 x 或存储单元 x 的内容。

表 7.8 16 位计算机指令功能

名称	指令的汇编格式	指令功能
加法指令	ADD Rs，Rd	(Rs)+(Rd) → Rd
算术 / 逻辑左移	SHL Rd	2*(Rd) → Rd
算术右移	SHR Rd	(Rd)/2 → Rd
取数指令	LOAD Rd, mem	(mem) → Rd
存数指令	STORE Rs, mem	(Rs) → mem

该计算机采用 5 段流水线方式执行指令，各流水段分别是取指令（IF）、译码 / 读寄存器（ID）、执行 / 计算有效地址（EX）、访问存储器（M）和结果写回寄存器（WB），流水线采用“按序发射，按序完成”方式，没有采用转发技术处理数据相关问题，并且同一个寄存器的读和写操作不能在同一个时钟周期内进行。

请回答下列问题。

（1）若 int 型变量 x 的值为 −513，存放在寄存器 R1 中，则执行指令“SHL R1”后，R1 的内容是多少？（用十六进制表示）

答：x 的补码为 1111 1101 1111 1111B，十六进制为 FDFFH，右移一位得到 1111 1110 1111 1111B，十六进制为 FEFFH

（2）若某个时间段中，有连续的 4 条指令进入流水线，在其执行过程中没有发生任何阻塞，则执行这 4 条指令所需的时钟周期数为多少？

答：至少为 5 + (n − 1) = 5 + (4 − 1) = 8 个时钟周期

（3）若高级语言程序中某赋值语句为 x=a+b，x、a 和 b 均为 int 型变量，它们的存储单元地址分别表示为 [x]、[a] 和 [b]。该语句对应的指令序列及其在指令流水线中的执行过程如图 7.32 所示，则这 4 条指令

执行过程中，I3 的 ID 段和 I4 的 IF 段被阻塞的原因各是什么？

```
I1    LOAD    R1, [a]
I2    LOAD    R2, [b]
I3    ADD     R1, R2
I4    STORE   R2, [x]
```

	时间单元													
指令	1	2	3	4	5	6	7	8	9	10	11	12	13	14
I1	IF	ID	EX	M	WB									
I2		IF	ID	EX	M	WB								
I3			IF				ID	EX	M	WB				
I4							IF				ID	EX	M	WB

图 7.32　指令序列及其执行过程示意图

答：因为 I3 和 I1、I2 都存在数据相关，需要等待 I1 和 I2 将结果写回寄存器后 I3 才可以访问到寄存器的内容。I4 被阻塞的原因是 I4 前一条指令 I3 在 ID 段被阻塞，所以 I4 的 IF 段被阻塞。

（4）若高级语言程序中某赋值语句为 x=x*2+a，x 和 a 均为 unsigned int 型变量，它们的存储单元地址分别表示为 [x]、[a]，则执行这条语句至少需要多少个时钟周期？要求模仿图 7.32 画出这条语句对应的指令序列及其在流水线中的执行过程示意图。

答：

指令序列：

```
I1    LOAD    R1, [x]
I2    LOAD    R2, [a]
I3    SHL     R1
I4    ADD     R1, R2
I5    STORE   R2, [x]
```

指令	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
I1	IF	ID	EX	M	WB												
I2		IF	ID	EX	M	WB											
I3			IF			ID	EX	M	WB								
I4						IF				ID	EX	M	WB				
I5										IF				ID	EX	M	WB

7.16 某程序中有如下循环代码段 p：“for(int i = 0; i <N; i++) sum+=A[i];”。假设编译时变量 sum 和 i 分别分配在寄存器 R1 和 R2 中，常量 N 在寄存器 R6 中，数组 A 的首地址在寄存器 R3 中。程序段 p 的起始地址为 08048100H，对应的汇编代码和机器代码如表 7.9 所示。

表 7.9 程序段 p 对应的汇编代码和机器代码

编号	地址	机器代码	汇编代码	注释
1	08048100H	00022080H	loop: sll R4,R2,2	(R2)<<2 → R4
2	08048104H	00083020H	add R4,R4,R3	(R4)+(R3) → R4
3	08048108H	8C850000H	load R5,0(R4)	((R4)+0) → R5
4	0804810CH	00250820H	add R1,R1,R5	(R1)+(R5) → R1
5	08048110H	20420001H	add R2,R2,1	(R2)+1 → R2
6	08048114H	1446FFFAH	bne R2,R6,loop	if(R2)!=(R6) goto loop

执行上述代码的计算机 M 采用 32 位定长指令字，其中分支指令 bne 采用图 7.33 所示格式。

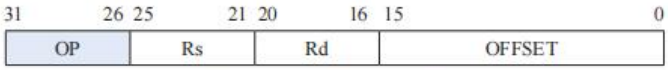

图 7.33 分枝指令 bne 的格式

图 7.33 中，OP 为操作码；Rs 和 Rd 为寄存器编号；OFFSET 为偏移量，用补码表示。请回答下列问题，并说明理由。

（1）M 的存储器编址单位是什么？

答：M 采用 32 位定长指令字，一条指令占 4B，表中各指令的地址均为 32 位，且相邻指令的地址差了 4 个地质单元，正好对应指令长度 4B，所以编址单位为字节。

（2）已知 sll 指令实现左移功能，数组 A 中每个元素占多少位？

答：左移两位相当于乘以 4，根据汇编代码可知数组间的数据间隔为 4 个地址单位，而计算机按字节编址，所以数组 A 中每个元素占 4B

（3）bne 指令的 OFFSET 字段的值是多少？已知 bne 指令采用相对寻址方式，当前 PC 的内容为 bne 指令地址，通过分析表 7.9 中的指令地址和 bne 指令内容，推断出 bne 指令的转移目标地址计算公式。

答：bne 指令的机器代码为 1446 FFFAH，根据指令格式，后 2B 的内容为 offset 字段，所以指令的 offset 字段为 FFFAH，用补码表示，值为-6。当系统执行到 bne 指令时，PC 自动加 4，PC 的内容就为 0804 8118H，而跳转的目标是 0804 8100H，两者相差了 18H，即-24 个字节距离，-24 / -6 = 4，可知 bne 指令的转移目标地址计算公式为(PC) + 4 + offset * 4

（4）若 M 采用“按序发射、按序完成”的 5 级指令流水线——IF（取指令）、ID（译码及取数）、EX（执行）、MEM（访存）、WB（写回寄存器），且硬件不采取任何转发措施，分支指令的执行均引起 3 个时钟周期的阻塞，则 p 中哪些指令的执行会因数据相关而发生流水线阻塞？哪条指令的执行会发生控制冒险？为什么指令 1 的执行不会因为与指令 5 数据相关而发生阻塞？

答：数据相关阻塞的指令为 2、3、4、6 条，第 6 条指令会发生控制冒险。当前循环的第 5 条指令与下次循环的第 1 条指令虽然有数据相关，但由于第 6 条指令后有 3 个时钟周期的阻塞，因而消除了该数据相关