7.2

(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.0.25x10^9 条指令/秒 B.0.97x10^9 条指令/秒 C.1.0x10^9 条指令/秒 D.1.03 x10^9 条指令/秒

仅当第一条指令时指令流水线有空闲，故 100 条指令需要(100+3)个时钟周期，所以吞吐率为 1.03G/103 * 100 = 1.0 * 10^9 条，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

即最长时间，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

最长时间+20ps=100ps，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

I2 写入 R5，I3 读 R5 数据，会发生读写冲突选 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

A.I1 写入 s2，I3 读 s2 数据，会发生读写冲突

B.I2 写入 s2，I3 读 s2 数据，会发生读写冲突

D.I3 写入 s2，I4 读 s2 数据，会发生读写冲突

只有 I2,I4 之间不发生冲突，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

数据旁路使用重定向的方法，不会引起指令流水线阻塞，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

Ⅰ.指令格式规整且长度一致：规整的指令格式可以使流水线中的各个阶段的处理逻辑更加简单和统一，减少了指令译码和执行的复杂度。长度一致的指令可以使每个流水线阶段的处理时间相等，避免了流水线阻塞和冲突。

Ⅱ.指令和数据按边界对齐存放：按边界对齐存放指令和数据可以简化内存访问和加载操作，减少了访存阶段的复杂性。边界对齐存放还可以提高内存访问的效率，减少指令和数据的读取时间。

Ⅲ.只有 Load/Store 指令才能对操作数进行存储访问：这种特点通常与精简指令集(RISC)体系结构相关。Load/Store 指令的特点是只能从内存中加载数据到寄存器或将寄存器中的数据存储回内存，而其他指令只能在寄存器之间进行操作。这种特点简化了指令流水线的设计，减少了数据冒险和相关性问题。

综上所述，选项 D 中的特点Ⅰ、Ⅱ和Ⅲ都有利于实现指令流水线。

(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

指令流水线数据通路通常由多个部件组成，包括取指部件、译码部件、执行部件、访存部件和写回部件等。这些部件负责执行指令流水线的各个阶段。不包含生成控制信号的控制部件，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

超标量流水线是一种流水线处理器的设计技术，具有以下特点：

Ⅱ. 能在一个时钟周期内同时发射多条指令：超标量流水线具有多个功能单元和执行单元，可以在一个时钟周期内并行执行多条指令。这样可以提高指令的执行速度和效率。

Ⅲ. 能结合动态调度技术提高指令执行并行性：超标量流水线通过动态调度技术，可以在运行时根据指令之间的依赖关系和资源可用性进行指令调度，以提高指令执行的并行性和效率。动态调度技术可以根据运行时的情况选择合适的指令并行执行，充分利用处理器资源。

Ⅰ. 能缩短流水线功能段的处理时间：这个叙述是错误的。超标量流水线并不能直接缩短流水线功能段的处理时间，它通过并行执行多条指令来提高整体的执行效率，但并不直接缩短功能段的处理时间。

综上所述，正确的叙述是 C.仅Ⅱ、Ⅲ。

(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

单周期 CPU 的 CPI 理想 CPI 就为 1，多周期 CPU 的理想 CPI<1，超标量流水线 CPU 的理想 CPI>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	取值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	and $s3,$s1,$s2	add $s2,$s2,$s1			lw $s1,($s0)
9	sub $s4,$s2,$s2	and $s3,$s1,$s2	add $s2,$s2,$s1		
10	sub $s4,$s2,$s2	and $s3,$s1,$s2		add $s2,$s2,$s1	
11	sub $s4,$s2,$s2	and $s3,$s1,$s2			add $s2,$s2,$s1
12	next instr	sub $s4,$s2,$s2	and $s3,$s1,$s2		

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。

CLKs	取值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	Next Intsr	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:
```

CLKs	取值IF	译码ID	执行EX	访存MEM	写回WB
1	add $s0,$0,100				
2	beq $s0,$0,done	add $s0,$0,100			
3	addi $s0,$s0,-1	beq $s0,$0,done	add $s0,$0,100		
4	j while_loop	addi $s0,$s0,-1	beq $s0,$0,done	add $s0,$0,100	
5	Next Instr	j while_loop	addi $s0,$s0,-1	beq $s0,$0,done	add $s0,$0,100
6	Next Instr	Next Instr	j while_loop	addi $s0,$s0,-1	beq $s0,$0,done
7	beq $s0,$0,done			j while_loop	addi $s0,$s0,-1
8	addi $s0,$s0,-1	beq $s0,$0,done			j while_loop
9	j while_loop	addi $s0,$s0,-1	beq $s0,$0,done		
10	Next Instr	j while_loop	addi $s0,$s0,-1	beq $s0,$0,done	
11	Next Instr	Next Instr	j while_loop	addi $s0,$s0,-1	beq $s0,$0,done
12	beq $s0,$0,done			j while_loop	addi $s0,$s0,-1
..	..	..	..	..	..
502	beq $s0,$0,done			j while_loop	addi $s0,$s0,-1
503	addi $s0,$s0,-1	beq $s0,$0,done			j while_loop
504	j while_loop	addi $s0,$s0,-1	beq $s0,$0,done		
505	Next Instr			beq $s0,$0,done	
506	Next Instr	Next Instr			beq $s0,$0,done

执行周期=506

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:
```

CLKs	取值IF	译码ID	执行EX	访存MEM	写回WB
1	addi $s0,$0,0				
2	addi $s1,$0,0	addi $s0,$0,0			
3	slti $t1,$s0,10	addi $s1,$0,0	addi $s0,$0,0		
4	beq $t1,$0,done	slti $t1,$s0,10	addi $s1,$0,0	addi $s0,$0,0	
5	addi $s1,$s1,$s0	beq $t1,$0,done	slti $t1,$s0,10	addi $s1,$0,0	addi $s0,$0,0
6	addi $s0,$s0,1	addi $s1,$s1,$s0	beq $t1,$0,done	slti $t1,$s0,10	addi $s1,$0,0
7	j for_loop	addi $s0,$s0,1	addi $s1,$s1,$s0	beq $t1,$0,done	slti $t1,$s0,10
8	Next Instr	j for_loop	addi $s0,$s0,1	addi $s1,$s1,$s0	beq $t1,$0,done
9	Next Instr	Next Instr	j for_loop	addi $s0,$s0,1	addi $s1,$s1,$s0
10	slti $t1,$s0,10			j for_loop	addi $s0,$s0,1
11	beq $t1,$0,done	slti $t1,$s0,10			j for_loop
..	..	..	..	..	..
73	slti $t1,$s0,10			j for_loop	addi $s0,$s0,1
74	beq $t1,$0,done	slti $t1,$s0,10			
75	addi $s1,$s1,$s0	beq $t1,$0,done	slti $t1,$s0,10		
76	addi $s0,$s0,1	addi $s1,$s1,$s0	beq $t1,$0,done	slti $t1,$s0,10	
77	Next Instr			beq $t1,$0,done	slti $t1,$s0,10
78	Next Instr	Next Instr			beq $t1,$0,done

执行周期数=78

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(E)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)

-513 = [1000 0010 0000 0001]$_{原}$=[1111 1101 1111 1111]$_{补}$ = R1 ， SHL R1 后

R1 = [1111 1101 1111 1110] = FDFE H

(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?

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

I3 的 ID 段被阻塞的原因：因为 I_3 与 I_2 和 I_1 都存在数据相关，需等到 I_2 和 I_1 将结果写回寄存器后，I_3 才能读寄存器内容，所以 I_3 的 ID 段被阻塞，流水线中插入了 3 个气泡。I_4 的 IF 段被阻塞的原因：因为 I_4 的前一条指令 I_3 在 ID 段被阻塞，所以 I_4 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。

I1　LOAD　R1，[x]

I2　LOAD　R2，[a]

I3　ADD　R1，R1

I4　ADD　R1，R2

I5　SOTRE　R1，[x]

	时间单元																
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。

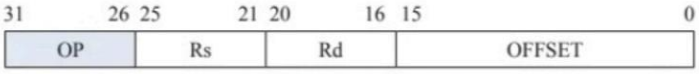

图 7.33　分枝指令 bne 的格式

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?

M 采用 32 位定长指令字，一条指令占 4B，表中指令也均为 32 位，相邻指令地址差也为 4 个地址单位，正好对应指令长度 4B，所以计算机是按字节编制的

(2)　已知 sll 指令实现左移功能，数组 A 中每个元素占多少位?

左移两位相当于乘 4，根据提供的汇编代码可知，每次变化 4，所以占 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。

1446FFFA H 后 16 位是 FFFA ，转化成原码是-0110 = -6，所以 OFFSET 的值是-6

PC = 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?

因为数据相关而发生流水线阻塞的是第 2，3，4，6 条

会发生控制冒险的是第 6 条

因为指令 5 执行后的指令 6 会有三个时钟周期的阻塞，恰好消除了数据相关