1. SEQ 模型：根据 Y-86 模型完成下表

		CALL Dest	JXX Dest
Fetch	icode:ifun	icode:ifun <- M_1[PC]	icode:ifun <- M_1[PC]
	rA,rB		
	valC	valC <- M_8[PC+1]	valC <- M_8[PC+1]
	valP	valP <- PC+9	valP <- PC+9
Decode	valA,srcA		
	valB,srcB		
Execute	valE		
	Cond Code		
Memory	valM		
Write Back	dstE		
	dstM		
PC Update	PC		

2. 已知 valA,valB 为从寄存器 rA,rB 中读出的值， valC 为指令中的常数值，valM 为访存得到的数据，valP 为 PC 自增得到的值，完成SEQ处理器中下面的HCL逻辑：

```
Stage: Execute
word aluA = [
    icode in { IRRMOVQ, IOPQ } : _____;
    icode in { IIRMOVQ, IRMMOVQ, IMRMOVQ } : _____;
    icode in { ICALL, IPUSHQ } : _____;
    icode in { IRET, IPOPQ } : _____;
];
```

```
Stage: PC Update
int new_pc = [
   icode == ICALL : _____;
   icode == IJXX && Cnd: _____;
   icode == IRET: _____;
   1: _____;
];
```

3．判断以下说法是否正确

（ ） (1) 流水线的深度越深，总吞吐率越大，因此流水线应当越深越好。

（ ） (2) 流水线的吞吐率取决于最慢的流水级，因此流水线的划分应当尽量均匀。

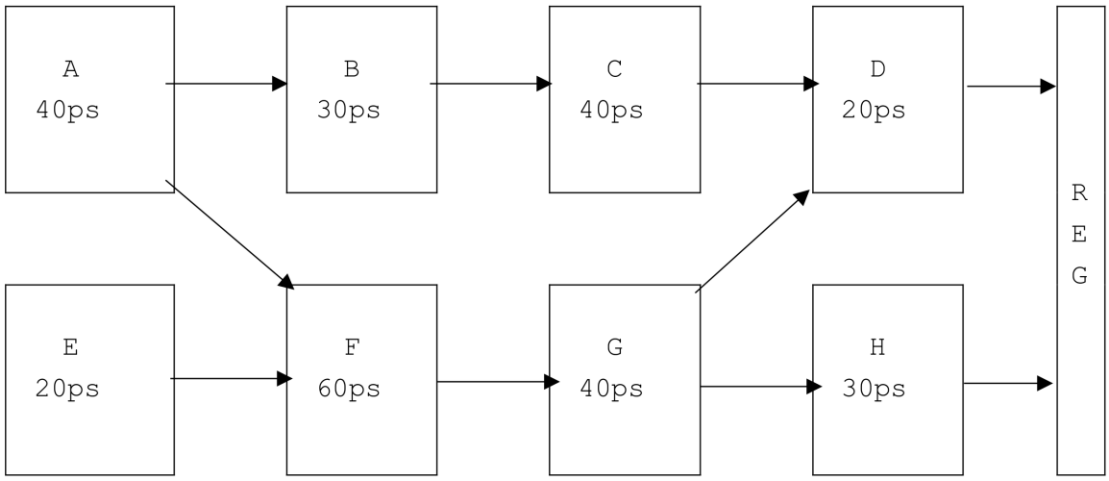
（ ） (3) 假设寄存器延迟为 20ps，那么总吞吐率不可能达到或超过 50 GIPS。

（ ） (4) 数据冒险总是可以只通过转发来解决。

（ ） (5) 数据冒险总是可以只通过暂停流水线来解决。

4．一条三级流水线，包括延迟为50ps，100ps，100ps 的三个流水级，每个寄存器的延迟为10ps。那么这条流水线的总延迟是_____ps，吞吐率是_____GIPS。

5．A~H 为8个基本逻辑单元，下图中标出了每个单元的延迟，以及用箭头标出了单元之间的数据依赖关系。寄存器的延迟均为10ps。



(1) 计算目前的电路的总延迟

(2) 通过插入寄存器，可以对这个电路进行流水化改造。现在想将其改造为两级流水线，为了达到尽可能高的吞吐率，问寄存器应插在何处?获得的吞吐率是多少?

(3) 现在想将其改造为三级流水线，问最优改造所获得的吞吐率是多少?

6．一个只使用流水线暂停、没有数据前递的Y86流水线处理器，为了执行以下的语句，至少需要**停顿**多少个周期?一共需要运行多少个周期?

irmovq $1, %rax
irmovq $2, %rbx
addq %rax, %rcx
addq %rbx, %rdx
halt
	rrmovl %eax, %edx
	mrmovl (%ecx), %eax
	addl %edx, %eax
	halt
		irmovl $0x40, %eax
		mrmovl (%eax), %ebx
		subl %ebx, %ecx
		halt

7．考虑Y86中的控制逻辑。jXX总是预测分支跳转。

(1)写出流水线需要处理ret的条件（ret对应的常量为IRET）：

(2)发现(1)中的条件后，流水线寄存器应如何设置?(选填stall, bubble, normal)

	Fetch	Decode	Execute	Memory	Writeback
ret					

(3)写出流水线需要处理jXX分支错误的条件（jXX对应的常量为IJXX）：

(4)发现(3)中的条件后，流水线寄存器应如何设置?(选填stall, bubble, normal)

	Fetch	Decode	Execute	Memory	Writeback
JXX错误					

(5)写出流水线需要处理load/use hazard(加载/使用冒险)的条件:

(6)发现(5)中的条件后，流水线寄存器应如何设置?(选填stall, bubble, normal)

	Fetch	Decode	Execute	Memory	Writeback
load/use					

(7)在Y86流水线中，是否存在一组指令序列可能同时满足上面的三个条件?是否可能同时出现上面的两种条件?

(8)考察以下两组指令序列，写出发现冒险时应该如何设置流水线寄存器。

... popq %rsp ret			je L1 addq %rdx, %rax .L1: ret		
	Fetch	Decode	Execute	Memory	Writeback
load/use+ret					
	Fetch	Decode	Execute	Memory	Writeback
jXX+ret					

(9)根据以上各小问，补全以下pipe.hcl中的控制逻辑。

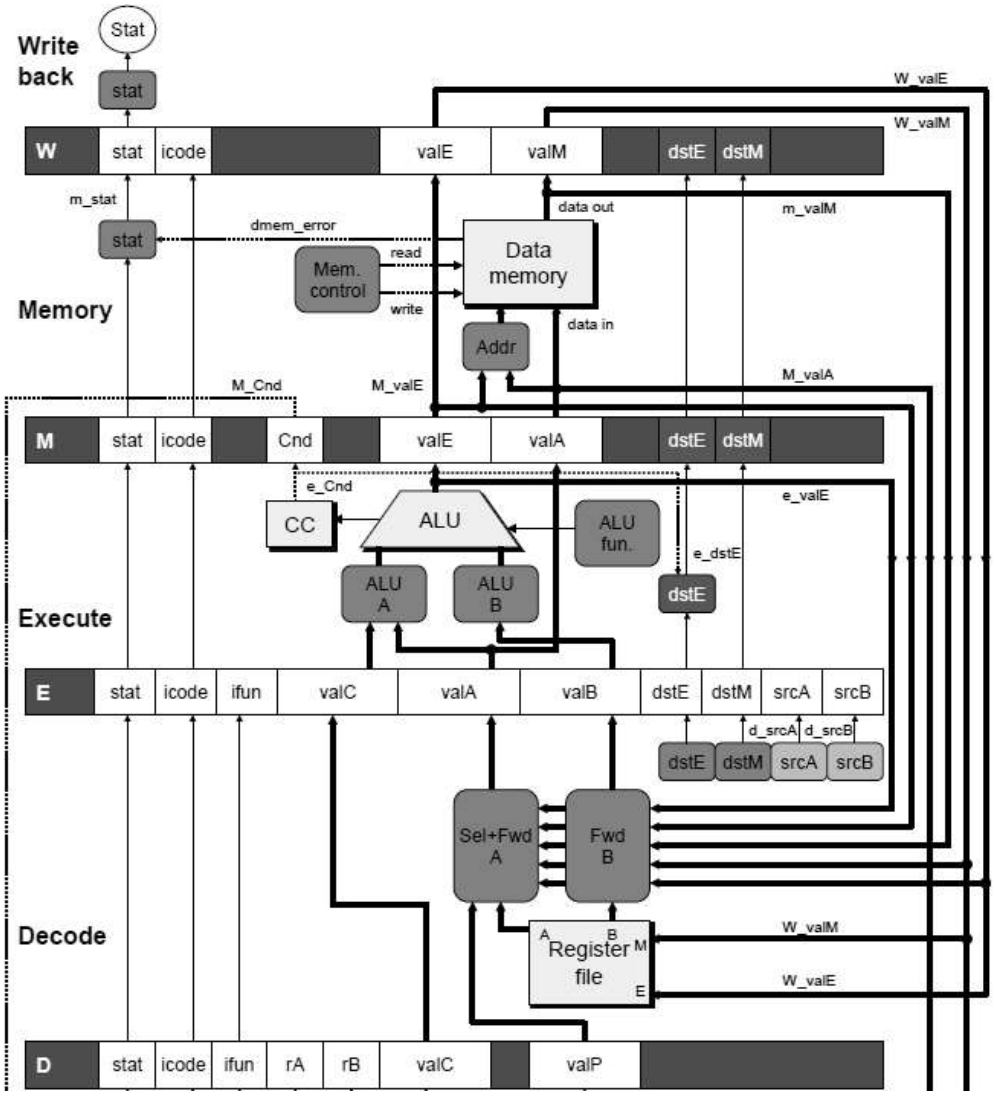
```
bool F_bubble = 0;
bool F_stall =
    E_icode in { IMRMOVQ, _______} &&
    E_dstM in { _______, _______ }  ||
  # Stalling at fetch while ret passes through pipeline
    IRET in { _______, _______, _______ };

bool D_bubble =
  # Mispredicted branch
    (E_icode == IJXX && !_______) ||
  # Stalling at fetch while ret passes through pipeline
  # but not condition for a load/use hazard
    !(______ in { IMRMOVQ, IPOPQ } &&
       ______ in { d_srcA, d_srcB }) &&
    ______ in { D_icode, E_icode, M_icode };
bool D_stall =
    # Conditions for a load/use hazard
    E_icode in { IMRMOVQ, IPOPQ } &&
     E_dstM in { d_srcA, d_srcB };
```

后面大题来自 18、19 年期中。

得分

第四题（15 分）

这是一款 Y86-32 流水线处理器的结构图（局部），请以此为基础，依次回答下列问题。



1、该处理器设计采用了前递（forwarding）技术，一定程度上解决了数据相关的问题，在上图中体现在 Sel+FwdA 和 FwdB 部件上。前者输出的信号会存到流水线寄存器 E 的 valA 域（即 E_valA 信号），请补全该信号的 HCL 语言描述。

```
int E_valA = [
    D_icode in { ICALL, IJXX } :_____ ; # ①
    d_srcA == e_dstE :_____ ;# ②
    d_srcA == M_dstM :_____ ;# ③
```

```
    d_srcA == M_dstE : M_valE       ;
    d_srcA == W_dstM : W_valM       ;
    …
];
```

2、如果在该处理器上运行下面的程序，每条指令在不同时钟周期所处的流水线阶段如下表所示。在这种情况下，哪条指令的执行结果会有错误？写出该指令的地址：_____。

```
demo1.ys
0x000: irmovl $128, %edx
0x006: irmovl $3, %ecx
0x00c: rmmovl %ecx, 0(%edx)
0x012: irmovl $10, %ebx
0x018: mrmovl 0(%edx),%eax
0x01e: addl %ebx, %eax
0x020: halt
```

1	2	3	4	5	6	7	8	9	10	11	12
F	D	E	M	W							
	F	D	E	M	W						
		F	D	E	M	W					
			F	D	E	M	W				
				F	D	E	M	W			
					F	D	E	M	W		
						F	D	E	M	W	

3、如需检测出这个情况，需要增加逻辑电路，用 HCL 语言表达如下：

E_icode in {IMRMOVL,IPOPL} && _____ in { _____ }

4、当新增的电路检测出这个情况后，应对各流水线寄存器进行不同的设置，以便在尽可能少影响性能的前提下解决该问题。请填写下表，可选的设置包括 normal/bubble/stall 三种。

F	D	E	M	W

5、如果遇到下面程序代码所展示的情况，该处理器运行时仍然存在问题。因此，还需要新增检测电路。当新增的电路检测出这个情况后，应对各流水线寄存器进行不同的设置，以便在尽可能少影响性能的前提下解决该问题。请填写下表，可选的设置包括 normal/bubble/stall 三种。

```
demo2.ys
…
0x018: rmmovl %ecx, 0(%edx)
0x01e: irmovl $10, %ebx
0x024: popl %esp
0x026: ret
```

F	D	E	M	W

得分

第四题（15 分）

请分析 Y86-64 ISA 中新加入的一族算术指令： irOpq V, rA, rB，其格式如下：

C	Fn	rA	rB	V（8 字节）

与 Opq 类似，这族指令由四个指令组成，分别是 iraddq, irsubq, irandq 和 irxorq。其功能为：计算 R[rA] OP V 并将结果存入 R[rB]中，这里 OP 根据 Fn 的取值分别取+, -, &和^，且此过程会设置条件码寄存器。

（1）若在教材所描述的 SEQ 处理器上执行这条指令，请按下表补全每个阶段的操作。需说明的信号可能会包括：icode, ifun, rA, rB, valA, valB, valC, valE, valP, Cnd; the register file R[], data memory M[], Program counter PC, condition codes CC。其中对存储器的引用必须标明字节数。如果在某一阶段没有任何操作，请填写 none 指明。

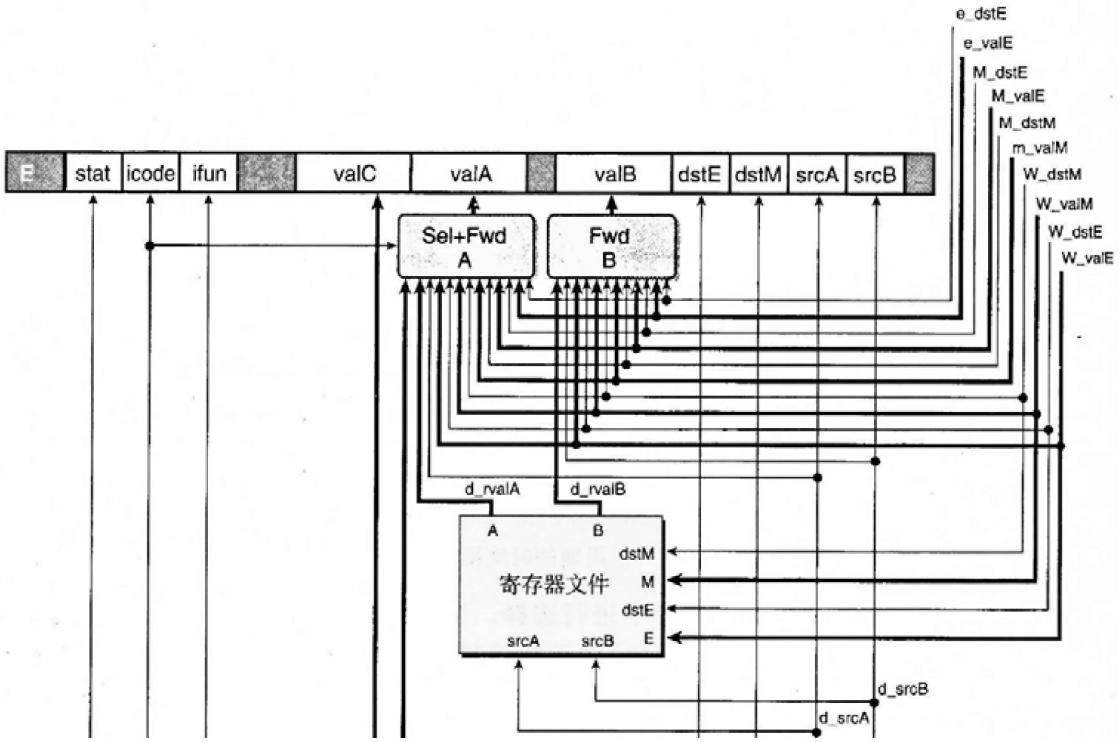
Stage	irOpq V, rA, rB
Fetch	icode : ifun ← M_1[PC]
Decode	valA ← R[rA] valB ← R[rB]
Execute	
Memory	none
Write Back	
Update PC	PC ← valP

（2）考虑如下一段 Y86-64 代码片段：

```
Loop:  mrmovq (%rdi), %r10      # line 1
       rmmovq %r10, (%rsi)      # line 2
       andq %r10, %r10          # line 3
```

```
        jle Npos                # line 4
        irmovq $1, %r10         # line 5
        addq %r10, %rax         # line 6
Npos:   irmovq $1, %r10         # line 7
        subq %r10, %rdx         # line 8
        irmovq $8, %r10         # line 9
        addq %r10, %rdi         # line 10
        addq %r10, %rsi         # line 11
        andq %rdx, %rdx         # line 12
        jg Loop                 # line 13
        ret                  # line 14
```

1. 这段代码中存在一些指令间的数据相关，其中行 5 与行 6 的数据相关可以采用数据前递（Forwarding）技术解决，在下图中体现在 Sel+FwdA 和 FwdB 部件上。前者输出的信号会存到流水线寄存器 E 的 valA 域（即 E_valA 信号），请选出该信号正确的 HCL 语言描述：_____



```
long d_valA = [
    D_icode in { ICALL, IJXX }: D_valP;
    _________;
    _________;
    _________;
    _________;
    _________;
    1: d_rvalA;
```

① d_srcA == e_dstE : e_valE
② d_srcA == M_dstE : M_valE
③ d_srcA == M_dstM : m_valM
④ d_srcA == W_dstE : W_valE
⑤ d_srcA == W_dstM : W_valM

A ①②③④⑤　　B ①③②⑤④　　C ④⑤②③①　　D ⑤④③②①

2. 同样是数据相关，上述代码中行 1 与行 2 的情况不能用以上方法解决。为了检测这种情况，需要增加逻辑电路，用 HCL 语言表达如下：
   E_icode in {_____} &&_____ in {_____}

3. 假设该代码片段在教材所描述的 PIPE 处理器上运行，不考虑该片段代码前后代码的影响以及高速缓存（cache）失效的情况，假设%rdx 初值为 10，%rdi 指向的内存中数组的元素均为正数，处理器设计使用总是选择（always taken）的预测策略。该代码片段预计运行_____周期，若使用新增加的 irOpq 指令来优化这段代码，可以节省_____周期的运行时间。