ICS 第五次小班课习题

【流水线的基本原理】

1. 判断下列说法的正确性

(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。

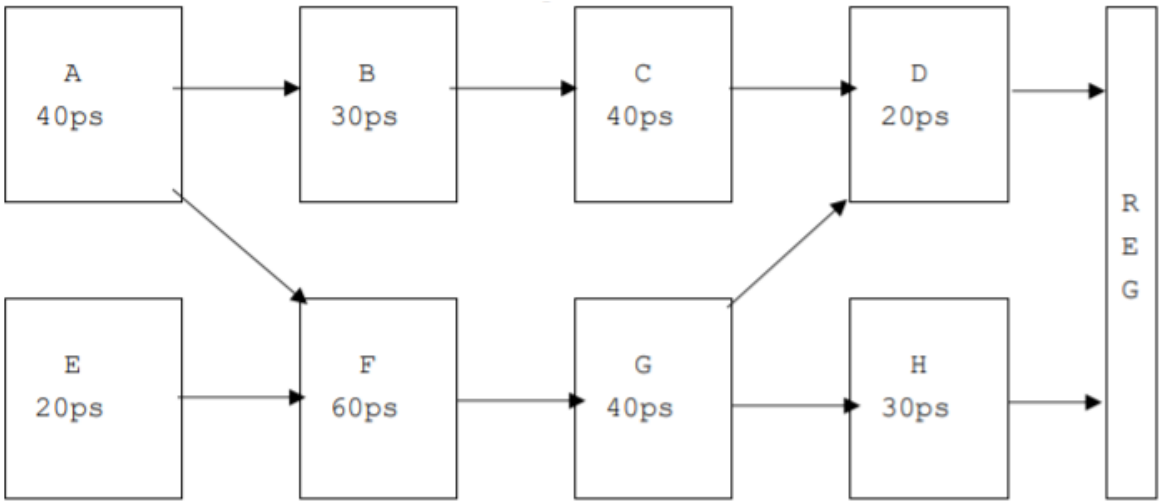
(3) ( )假设寄存器延迟为 20ps，那么总吞吐率不可能达到或超过 50 GIPS。

(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。

2. 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。

3. 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？

【流水线处理器】

4. 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，至少需要累计停顿多少个周期?

irmovl $1, %eax
irmovl $2, %ebx
addl %eax, %ecx
addl %ebx, %edx
halt
	rrmovl %eax, %edx
	mrmovl (%ecx), %eax
	addl %edx, %eax
	halt
		irmovl $0x40, %eax
		mrmovl (%eax), %ebx
		subl %ebx, %ecx
		halt

(1)	(2)	(3)

5. 考虑 Y86 中的 ret 与 jXX 指令。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) 写出发现上述条件以后，流水线寄存器应设置的状态

	Fetch	Decode	Execute	Memory	Writeback
处理 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) 写出发现上述条件以后，流水线寄存器应设置的状态

	Fetch	Decode	Execute	Memory	Writeback
分支错误					

(5) 写出下一条指令地址 f_pc 的控制逻辑

```
int f_pc = [
    M_icode == IJXX && !M_Cnd : __________;
    W_icode == IRET : __________;
    1 : F_predPC;
];

#  已知有如下的代码，其中 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：
int f_predPC = [
    f_icode in { IJXX, ICALL } : f_valC;
    1 : f_valP;
];
int d_valA = [
    D_icode in { ICALL, IJXX } : D_valP; # Use incremented PC
    # ...省略部分数据前递代码
    1 : d_rvalA;  # Use value read from register file
];
```

6. (2016 期中流水线)

7. (2018 期中流水线)

【程序性能优化】

1．有如下的定义：

```
// 以下都是局部变量
int i, j, temp, ians;
int *p, *q, *r;
double dans;

// 以下都是全局变量
int iMat[100][100];

double dMat[100][100];

// 以下都是函数
int foo(int x);
```

判断编译器是否会自动将下列左侧代码优化为右侧代码：

(1)

ians = 0;
for (j = 0; j < 100; j++)
  for (i = 0; i < 100; i++)
  ians += iMat[i][j];
	ians = 0;
	for (i = 0; i < 100; i++)
	  for (j = 0; j < 100; j++)
	  ians += iMat[i][j];

(2)

dans = 0;
for (j = 0; j < 100; j++)
  for (i = 0; i < 100; i++)
  dans += dMat[i][j];
	dans = 0;
	for (i = 0; i < 100; i++)
	  for (j = 0; j < 100; j++)
	  dans += dMat[i][j];

(3)

for (i = 0; i < foo(100); i++)
  ians += iMat[0][i];
	temp = foo(100);
	for (i = 0; i < temp; i++)
	  ians += iMat[0][i];

(4)

*p += *q; *p += *r;	temp = *q + *r; *p += temp;

2．阅读下列 C 代码以及它编译生成的汇编语言

```
long func() {
    long ans = 1;
    long i;
    for (i = 0; i < 1000; i += 2) {
        ans = ans ? (A[i] ? A[i+1]);
    }
    return ans;
}
```

```
func:
    movl    $0, %edx
    movl    $1, %eax
    leaq    A(%rip), %rsi
    jmp .L2
.L3:
    movq    8(%rsi,%rdx,8), %rcx        // 2 cycles
    ??      (%rsi,%rdx,8), %rcx         // k + 1 cycles
    ??      %rcx, %rax                  // k cycles
    addq    $2, %rdx                    // 1 cycles
.L2:
    cmpq    $999, %rdx                  // 1 cycles
    jle .L3
    rep ret
```

该程序每轮循环处理两个元素。在理想的机器上（执行单元足够多），每条指令消耗的时间周期如右边所示。

(1) 当问号处为乘法时，k = 8。此时这段程序的 CPE 为 _____。

(2) 当问号处为加法时，k = 1。此时这段程序的 CPE 为 _____。