计算机组成
(2019级)

习题课

➢1

计算机组成习题参考答案
——组合逻辑

➢2

一、填空题

❖1．PN结的单向导电性即反向偏置时截止，正向偏置时导通。

❖2．TTL与非门的两个状态通常称为关态和开态，当输入全为高电平时对应的是开态，此时输出为低电平；当输入有一为低电平时，对应的是关态，此时输出为高电平。

❖3．TTL与非门的额定输出逻辑低电平V_{OL}=0.35伏额定输出逻辑高电平V_{OH}=3.0伏，。（设电源电压V_{CC}=+5V）

❖4．对于ECL、TTL、CMOS集成电路，按静态功耗低和高的顺序依次为CMOS、TTL、ECL，按工作速度快慢的顺序依次为ECL、TTL、CMOS，按抗干扰能力强弱的顺序依次为CMOS、TTL、ECL。

❖5．逻辑变量和函数只有0、1两种取值，而且它们只是表示两种不同的逻辑状态。

❖6．逻辑函数 $F=\overline{A}C+\overline{B}D$，其反函数 $\overline{F}=(A+\overline{C})(B+\overline{D})$ ，其对偶式 $F^*=(\overline{A}+C)(\overline{B}+D)$.

➢3

一、填空题

❖7．函数$F=AB+AC+\overline{C}D+ACDE$最简与或式是 $F=AB+AC+\overline{C}D$

❖8．从结构看，组合逻辑电路由门电路构成，不含存储电路，也不含反馈电路，信号从输入开始单向传输到输出。对于组合逻辑电路，任何时刻电路的输出仅由当时的输入信号决定。

❖9．将加在电路若干输入端中的某一个输入端的信号变换成相应的一组二进制代码输出的过程叫做编码。

❖10．将二进制代码所表示的信息翻译成对应输出的高低电平信号的过程称为译码；n位二进制译码器有n个输入，有2^n个输出，工作时译码器只允许有一个输出有效。

❖11．输出低电平有效的二-十进制译码器的输入8421BCD码A_3~A_0为0111时，其输出 $\overline{Y}_0 \sim \overline{Y}_9$=1111111011。

➢4

二、选择题

❖1. 硅二极管导通和截止的条件是（③）。

① $U_D \geq 0.7V$，$U_D < 0.5V$

② $U_D \geq 0.5V$，$U_D < 0.7V$

③ $U_D \geq 0.7V$，$U_D < 0.7V$

④ $U_D \geq 0.5V$，$U_D < 0.5V$

❖2. 用电位关系描述晶体三极管工作时的三种状态，正确的可靠工作条件是（④）。

① 截止区：VBE＜0V，放大区：VBC＞0V,饱和区：VBC>0V且VBC＞0V

② 截止区：VBE＜VT，放大区：VBE＞VT且VBC＜0V；饱和区：VBC＜0V

③ 截止区：VBE＜0V，放大区：VBE＞0.6V，饱和区：VCE≤0.3V

④ 截止区：VBE≤0V；VBC＜0V；放大区：VBE>0，且VBC＜0V；饱和区：VBE≥0.7V，且VBC＞0V

➢5

二、选择题

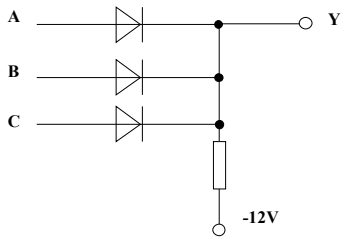
❖3. 二极管门电路如下图所示，假定输入变量的值可在0V和3V两种电平下变化，输出和输入之间的正逻辑关系为（②）。

① 与逻辑

② 或逻辑

③ 与非

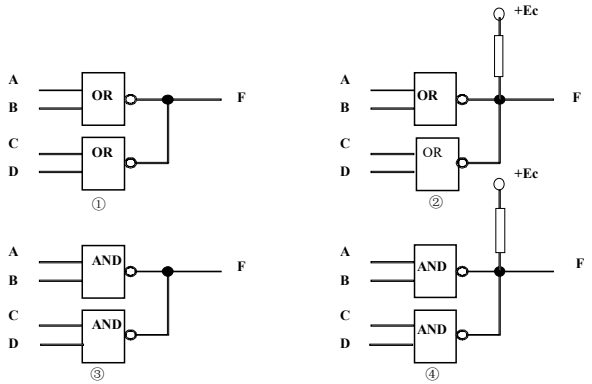
④ 或非



➢6

二、选择题

❖4. 为实现$F = \overline{AB} \cdot \overline{CD}$，下列电路接法正确的是（④）。



➢7

二、选择题

❖5. 函数$F = AB + \overline{A}C + \overline{B}C + \overline{C}D + \overline{D}$的最简与或式为（①）。

①1 ②0 ③AB ④以上均不是

▪解析：$F = AB + \overline{A}C + \overline{B}C + \overline{C}D + \overline{D}$

$= AB + (\overline{A} + \overline{B})C + \overline{C}D + \overline{D}$（利用反演律）

$= AB + \overline{AB}C + \overline{C}D + \overline{D}$（利用吸收律3）

$= AB + C + \overline{C}D + \overline{D} = AB + C + D + \overline{D}$

$= AB + C + 1 = 1$

▪吸收律3：$A + \overline{A}B = A + B$

➢8

二、选择题

❖6．逻辑函数$F=\overline{A+B\overline{C}}(A+B)$，当ABC的取值为（②）时，F=1。

① 000　② 011　③ 101　④ 111

解析：将各输入分别代入逻辑函数中，可知当ABC=011时，F=1。

❖ 7．函数$A\oplus B$与$\overline{A}\oplus\overline{B}$（③）。

① 互为反函数　② 互为对偶式

③ 相等　④ 答案都不正确

- 解析：因为 $A\oplus B=\overline{A}B+A\overline{B}$ ，$\overline{A}\oplus\overline{B}=\overline{\overline{A}B}+\overline{A\overline{B}}=A\overline{B}+\overline{A}B$ ，所以 $A\oplus B=\overline{A}\oplus\overline{B}$

二、选择题

❖8．组合逻辑电路的竞争-冒险是由于（③）引起的。

① 电路不是最简

② 电路有多个输出

③ 构成电路的逻辑元件存在传输延迟

④ 电路使用不同的门电路

❖ 9．能实现从多个输入端中选出一路作为输出的电路称为（③）。

① 触发器　② 计数器　③ 数据选择器　④ 译码器

❖ 10．只考虑本位数而不考虑低位来的进位的加法称为（②）。

① 全加　② 半加　③ 全减　④ 半减

❖11．如需要判断两个二进制数的大小或相等，可使用（④）电路。

① 译码器　② 编码器

③ 数据选择器　④ 数值比较器

三、问答与计算题

❖1．将逻辑函数 $F=ABC+\overline{A}BD$ 写成标准与或表达式。

解：利用基本公式　$A+\overline{A}=$可以把任何一个逻辑函数化为标准与或表达式。

$$
\begin{aligned}
F &= ABC + \overline{A}BD \\
&= ABC\ (D+\overline{D}) + \overline{A}BD\ (C+\overline{C}) \\
&= ABCD + ABC\overline{D} + \overline{A}BCD + \overline{A}B\,\overline{C}\,D \\
&= m_{15} + m_{14} + m_7 + m_5 \\
&= \sum m\,(5,7,14,15)
\end{aligned}
$$

三、问答与计算题

❖2．推导出函数$F=\overline{AB+BC}+A\overline{C}+A\overline{B}$ 的最简与或式。

解：

$$
\begin{aligned}
&F=\overline{AB}\cdot\overline{BC}+A\overline{C} \\
&=(\overline{A}+\overline{B})(\overline{B}+\overline{C})+A\overline{C}\text{（利用反演律）} \\
&\text{展开，} \\
&F=\overline{A}\overline{B}+\overline{A}\overline{C}+\overline{B}+A\overline{C} \\
&\text{（利用吸收律2：}\overline{B}(\overline{B}+\overline{C})=\overline{B}\text{）} \\
&F=(\overline{A}\overline{B}+\overline{B})+(\overline{A}\overline{C}+A\overline{C}) \\
&=\overline{B}+\overline{C}
\end{aligned}
$$

三、问答与计算题

❖3．列出下述问题的真值表，利用最小项推导法写出其逻辑函数表达式，利用公式简化法进行简化，然后，写出完整的Verilog HDL程序。

有3个温度检测器，当的超过60°时，温度检测器输出信号为1；低于60°时，输出为0。当两个或以上的温度检测器输出为1时，总控制器的输出为1，以控制调设备，使温度降低到60°以下。

三、问答与计算题

有3个温度检测器，当的超过60°时，温度检测器输出信号为1；低于60°时，输出为0。当两个或以上的温度检测器输出为1时，总控制器的输出为1，以控制调设备，使温度降低到60°以下。

- 解：设A、B、C分别表示温度检测器的输出信号，Y表示总控制器的输出信号，根据电路的功能要求列出真值表如下表所示：
- 利用最小项推导法：使输出为1的输入组合写成乘积项的形式，其中取值为1的输入用原变量表示，取值为0的输入用反变量表示，然后把这些乘积项加起来。

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$$Y = BC + AC + AB + ABC$$
$$= BC + AC + AB = AB + AC + BC$$

完整的 VerilogHDL 程序：

```
module   and_or(Y,A,B,C);
         output   Y;
         input    A,B,C;
         assign   Y=(A&&B)||(A&&C)||(B&&C);
endmodule
```


三、问答与计算题

❖4．用公式法证明下列等式：

$$\overline{B}C\overline{D}+B\overline{C}D+ACD+\overline{A}B\overline{C}D+\overline{A}\overline{B}CD+B\overline{C}\overline{D}+BCD=B+C$$

证明：

$$\overline{B}C\overline{D}+B\overline{C}D+ACD+\overline{A}B\overline{C}D+\overline{A}\overline{B}CD+B\overline{C}\overline{D}+BCD$$

- 合并：$=\overline{B}C\overline{D}+BD(C+\overline{C})+(A+\overline{A}\overline{B})CD+(\overline{A}+1)B\overline{C}\overline{D}$
- 由反演律：$=\overline{B}C\overline{D}+BD+(A+\overline{A}+\overline{B})CD+B\overline{C}\overline{D}$
- 由互补律：$=\overline{B}C\overline{D}+BD+BCD+B\overline{C}\overline{D}=BD+\overline{B}C+B\overline{C}\overline{D}$
- 由反演率：$=BD+\overline{B}C+B(\overline{C}+\overline{D})=\overline{B}C+B\overline{C}+B=B+C$

或：

$$\overline{B}C\overline{D}+B\overline{C}D+ACD+\overline{A}B\overline{C}D+\overline{A}\overline{B}CD+B\overline{C}\overline{D}+BCD$$
$$=\overline{B}C\overline{D}+BD(C+\overline{C})+(A+\overline{A}\overline{B})CD+(\overline{A}+1)B\overline{C}\overline{D}$$
$$=\overline{B}C\overline{D}+BD+(A+\overline{A}+\overline{B})CD+B\overline{C}\overline{D}$$
$$=\overline{B}C\overline{D}+BD+CD+B\overline{C}\overline{D}=\overline{B}C\overline{D}+BD+(CD+B\overline{C}\overline{D})\quad\text{（吸收律3）}$$
$$=\overline{B}C\overline{D}+BD+CD+B=(\overline{B}C\overline{D}+B)+BD+CD=C\overline{D}+B+BD+CD$$
$$=(C\overline{D}+CD)+(B+BD)=B+C$$

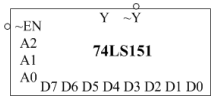
三、问答与计算题

❖5．用8选1数据选择器CT74151实现下列函数：

❖ F(A, B, C, D, E) = ∑m(1, 2, 3, 4, 7, 8, 10, 13, 14, 17, 19, 20, 21, 23, 24, 26, 28, 30, 31)

解答：

- CT74151为8选1数据选择器，其输出$Y(A_2,A_1,A_0)=m_0D_0+m_1D_1+m_2D_2+m_3D_3+m_4D_4+m_5D_5+m_6D_6+m_7D_7$
- 题目中的函数F(A, B, C, D, E) = ∑m(1, 2, 3, 4, 7, 8, 10, 13, 14, 17, 19, 20, 21, 23, 24, 26, 28, 30, 31)，具有5个输入端以及最多32个最小项，因而需要1个32选1数据选择器
- 32选1数据选择器应由4片74151扩展而来，扩展后的32选1数据选择器的输出为：$Y(A_4,A_3,A_2,A_1,A_0)=m_0D_0+m_1D_1+m_2D_2+m_3D_3+m_4D_4+\ldots\ldots+m_{30}D_{30}+m_{31}D_{31}$
- 结合题目中的函数，扩展后的32选1数据选择器的输入端D_0~D_{31}的值应分别为：01111001101001100101110110101011

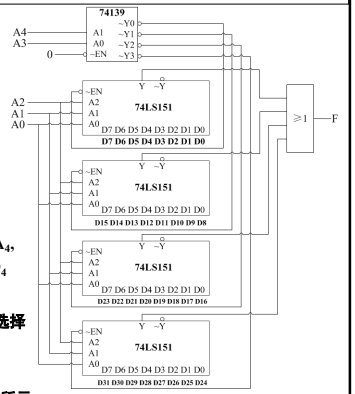


三、问答与计算题

$F(A,B,C,D,E)=\sum m(1,2,3,4,7,8,10,13,14,17,19,20,21,23,24,26,28,30,31)$

4片74151扩展为32选1数据选择器的方法：

- 32选1数据选择器输入的低三位A_2、A_1、A_0接每一片74151的A_2、A_1、A_0输入端，高两位A_4、A_3用做片选，因此还需要增加一个低输出有效的2-4译码器74139，4个输出端分别连接到4片74151的使能端
- 扩展后74139与4片74151的连接方式以及地址信号、数据输入端的对应关系如右图所示



- 结合扩展后的32选1数据选择器的输出函数：$Y(A_4, A_3, A_2, A_1, A_0) = m_0D_0 + m_1D_1 + m_2D_2 + m_3D_3 + m_4D_4 + \ldots\ldots + m_{30}D_{30} + m_{31}D_{31}$
- 以及题目中所要求的函数，扩展后的32选1数据选择器的输入端D_0~D_{31}的值应分别为：
  01111001101001100101110110101011
- 可以得到每一个数据输入端应赋予的值，如右图所示

➢17

三、问答与计算题

❖6．在三进制数系统中，存在三个数字：0, 1, 2。表1定义了一个三进制数的半加器。

（1）设计一个实现此半加器的电路。要求用二进制编码表示三进制数，例如每个三进制数用2位表示。令$A = a_1a_0$，$B = b_1b_0$，$Sum = s_1s_0$，进位信号Carry是二进制信号。编码方案为：$00=(0)_3$，$01=(1)_3$，$10=(2)_3$，要求电路成本最低

（2）使用上述描述的方法设计一个三进制全加器电路。

A B	Sum	Carry
00 00	00	0
00 01	01	0
00 10	10	0
01 00	01	0
01 01	10	0
01 10	00	1
10 00	10	0
10 01	00	1
10 10	01	1

表1 三进制半加器

➢18

三、问答与计算题

解答：

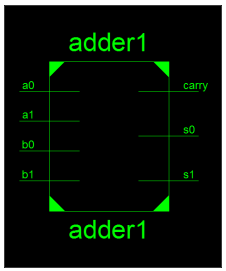
（1）使用最小项表示法可得：

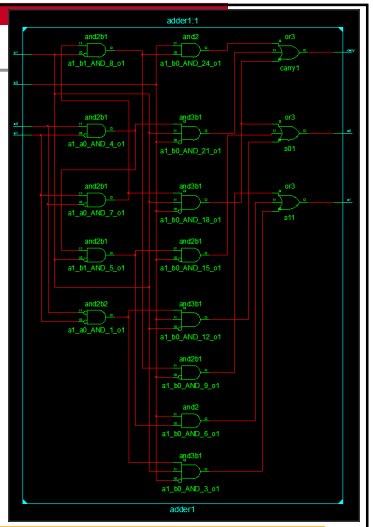
- $s_1 = m_2 + m_5 + m_8 = (!a1 \&\& !a0 \&\& b1 \&\& !b0) || (!a1 \&\& a0 \&\& !b1 \&\& b0) || (a1 \&\& !a0 \&\& !b1 \&\& !b0)$
- $s_0 = m_1 + m_4 + m_{10} = (!a1 \&\& !a0 \&\& !b1 \&\& b0) || (!a1 \&\& a0 \&\& !b1 \&\& !b0) || (a1 \&\& !a0 \&\& b1 \&\& !b0)$
- $Carry = m_6 + m_9 + m_{10} = (!a1 \&\& a0 \&\& b1 \&\& !b0) || (a1 \&\& !a0 \&\& !b1 \&\& b0) || (a1 \&\& !a0 \&\& b1 \&\& !b0)$

➢19

三、问答与计算题

（1）使用ISE综合之后得到的顶层模块如下图所示，模块内部的电路结构如右图所示（采用结构级描述方法）：



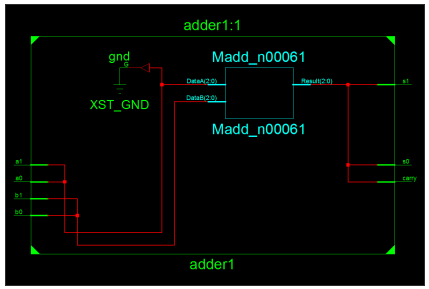


➢20

三、问答与计算题

（1）若使用行为级描述方法来实现，则不能像结构描述那样得到具体的电路：

- assign {carry,s1,s0}={a1,a0}+{b1,b0};



三、问答与计算题

（2）三进制全加器的真值表如下所示，其中low表示来自低位的进位：

A	B	low	Sum	Carry
00	00	0	00	0
00	01	0	01	0
00	10	0	10	0
01	00	0	01	0
01	01	0	10	0
01	10	0	00	1
10	00	0	10	0
10	01	0	00	1
10	10	0	01	1
00	00	1	01	0
00	01	1	10	0
00	10	1	00	1
01	00	1	10	0
01	01	1	00	1
01	10	1	01	1
10	00	1	00	1
10	01	1	01	1
10	10	1	10	1

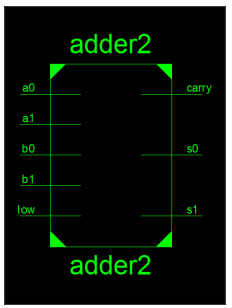
三、问答与计算题

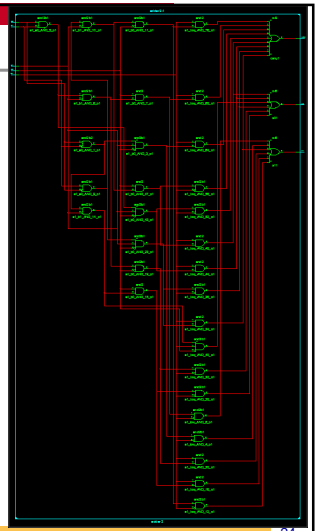
（2）同样使用最小项表示法可得：

- $s_1 = m_4 + m_{10} + m_{16} + m_3 + m_9 + m_{21}$ = (!a1 && !a0 && b1 && !b0 && !low) || (!a1 && a0 && !b1 && b0 && !low) || (a1 && !a0 && !b1 && !b0 && !low) || (!a1 && !a0 && !b1 && b0 && low) || (!a1 && a0 && !b1 && !b0 && low) || (a1 && !a0 && b1 && !b0 && low)
- $s_0 = m_2 + m_8 + m_{20} + m_1 + m_{13} + m_{19}$=(!a1 && !a0 && !b1 && b0 && !low) || (!a1 && a0 && !b1 && !b0 && !low) || (a1 && !a0 && b1 && !b0 && !low) || (!a1 && !a0 && !b1 && !b0 && low) || (!a1 && a0 && b1 && !b0 && low) || (a1 && !a0 && !b1 && b0 && low)
- $Carry = m_{12} + m_{18} + m_{20} + m_5 + m_{11} + m_{13} + m_{17} + m_{19} + m_{21}$ = (!a1 && a0 && b1 && !b0 && !low) || (a1 && !a0 && !b1 && b0 && !low) || (a1 && !a0 && b1 && !b0 && !low) || (!a1 && !a0 && b1 && !b0 && low) || (!a1 && a0 && !b1 && b0 && low) || (!a1 && a0 && b1 && !b0 && low) || (a1 && !a0 && !b1 && !b0 && low) || (a1 && !a0 && !b1 && b0 && low) || (a1 && !a0 && b1 && !b0 && low)

三、问答与计算题

（2）使用ISE综合之后得到的顶层模块如下图所示，模块内部的电路结构如右图所示（采用结构级描述方法）：

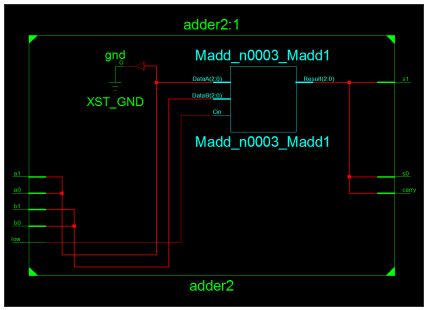




三、问答与计算题

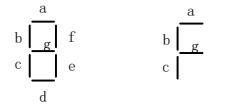
（2）同样的，若使用行为级描述方法来实现，则不能像结构描述那样得到具体的电路：

- assign {carry,s1,s0}={a1,a0}+{b1,b0}+low;



三、问答与计算题

❖7．7段数码管是由7个独立的发光管构成的，每个发光管有一个驱动控制信号。当驱动控制信号为高电平(逻辑1)时，则信号对应的发光管发光。现需设计7段数码管的控制电路，使之能够根据4位输入x[3:0]显示0～9，A～F共16个数字或图案。7段数码管控制电路输出信号为各数码管的驱动控制信号，即a，b，c，d，e，f，g。数码管各段的定义和16进制数“F”（对应abcdefg的二进制输出为1110001）的显示如下图所示。



a）各段定义

b）16进制数F的显示

（1）请给出7段数码管控制电路的输入输出信号真值表。

（2）根据真值表写出各输出信号的逻辑表达式，并化简。

（3）采用结构描述法，用Verilog语言实现上述的数码管控制器。

三、问答与计算题

（1）请给出7段数码管控制电路的输入输出信号真值表。

解答：①若显示数字“1”时f、e亮：

x3	x2	x1	x0	a	b	c	d	e	f	g	显示数字
0	0	0	0	1	1	1	1	1	1	0	0
0	0	0	1	0	0	0	0	1	1	0	1
0	0	1	0	1	0	1	1	0	1	1	2
0	0	1	1	1	0	0	1	1	1	1	3
0	1	0	0	0	1	0	0	1	1	1	4
0	1	0	1	1	1	0	1	1	0	1	5
0	1	1	0	1	1	1	1	1	0	1	6
0	1	1	1	1	0	0	0	1	1	0	7
1	0	0	0	1	1	1	1	1	1	1	8
1	0	0	1	1	1	1	0	1	1	1	9
1	0	1	0	1	1	1	0	1	1	1	A
1	0	1	1	0	1	1	1	1	0	1	B
1	1	0	0	1	1	1	1	0	0	0	C
1	1	0	1	0	0	1	1	1	1	1	D
1	1	1	0	1	1	1	1	0	0	1	E
1	1	1	1	1	1	1	0	0	0	1	F

三、问答与计算题

（1）请给出7段数码管控制电路的输入输出信号真值表。

解答：②若显示数字“1”时b、c亮：

x3	x2	x1	x0	a	b	c	d	e	f	g	显示数字
0	0	0	0	1	1	1	1	1	1	0	0
0	0	0	1	0	1	1	0	0	0	0	1
0	0	1	0	1	0	1	1	0	1	1	2
0	0	1	1	1	0	0	1	1	1	1	3
0	1	0	0	0	1	0	0	1	1	1	4
0	1	0	1	1	1	0	1	1	0	1	5
0	1	1	0	1	1	1	1	1	0	1	6
0	1	1	1	1	0	0	0	1	1	0	7
1	0	0	0	1	1	1	1	1	1	1	8
1	0	0	1	1	1	1	0/1	1	1	1	9
1	0	1	0	1	1	1	0	1	1	1	A
1	0	1	1	0	1	1	1	1	0	1	B
1	1	0	0	1	1	1	1	0	0	0	C
1	1	0	1	0	0	1	1	1	1	1	D
1	1	1	0	1	1	1	1	0	0	1	E
1	1	1	1	1	1	1	0	0	0	1	F

三、问答与计算题

（2）根据真值表写出各输出信号的逻辑表达式，并化简。

解答：以下的解题过程均假设显示数字“1”时f、e亮，采用最小项表示法：

- a = m0 + m2 + m3 + m5 + m6 + m7 + m8 + m9 + m10 + m12 + m14 +m15 = (!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && x1 && !x0) || (!x3 && !x2 && x1 && x0) || (!x3 && x2 && !x1 && x0) || (!x3 && x2 && x1 && !x0) || (!x3 && x2 && x1 && x0) || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0) || (x3 && x2 && !x1 && !x0) || (x3 && x2 && !x1 && x0) || (x3 && x2 && x1 && !x0) || (x3 && x2 && x1 && x0)
- b = m0 + m4 + m5 + m6 + m8 + m9 + m10 + m11 + m12 + m14 + m15 = (!x3 && !x2 && !x1 && !x0) || (!x3 && x2 && !x1 && !x0) || (!x3 && x2 && !x1 && x0) || (!x3 && x2 && x1 && !x0) || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0) || (x3 && x2 && !x1 && !x0) || (x3 && x2 && !x1 && x0) || (x3 && x2 && x1 && !x0) || (x3 && x2 && x1 && x0)

三、问答与计算题

（2）根据真值表写出各输出信号的逻辑表达式，并化简。

解答：以下的解题过程均假设显示数字“1”时f、e亮，采用最小项表示法：

- c = m0 + m2 + m6 + m8 + m10 + m11 + m12 + m13 + m14 + m15 = (!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && x1 && !x0) || (!x3 && x2 && x1 && !x0) || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0) || (x3 && x2 && !x1 && !x0) || (x3 && x2 && !x1 && x0) || (x3 && x2 && x1 && !x0) || (x3 && x2 && x1 && x0)
- d = m0 + m2 + m3 + m5 + m6 + m8 + m9 + m11 + m12 + m13 + m14 = (!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && x1 && !x0) || (!x3 && !x2 && x1 && x0) || (!x3 && x2 && !x1 && !x0) || (!x3 && x2 && x1 && !x0) || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && x0) || (x3 && x2 && !x1 && !x0) || (x3 && x2 && !x1 && x0) || (x3 && x2 && x1 && !x0)

三、问答与计算题

（2）根据真值表写出各输出信号的逻辑表达式，并化简。

解答：以下的解题过程均假设显示数字“1”时f、e亮，采用最小项表示法：

- e = m0 + m1 + m3 + m4 + m5 + m6 + m7 + m8 + m9 + m10 + m11 + m13 = (!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && !x1 && x0) || (!x3 && !x2 && x1 && x0) || (!x3 && x2 && !x1 && !x0) || (!x3 && x2 && !x1 && x0) || (!x3 && x2 && x1 && !x0) || (!x3 && x2 && x1 && x0) || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0) || (x3 && x2 && !x1 && x0)
- f = m0 + m1 + m2 + m3 + m4 + m7 + m8 + m9 + m10 + m13 = (!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && !x1 && x0) || (!x3 && !x2 && x1 && !x0) || (!x3 && !x2 && x1 && x0) || (!x3 && x2 && !x1 && !x0) || (!x3 && x2 && x1 && x0) || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0) || (x3 && x2 && !x1 && !x0)

三、问答与计算题

（2）根据真值表写出各输出信号的逻辑表达式，并化简。

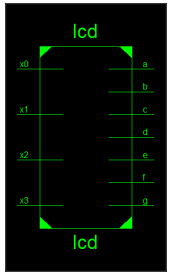
解答：以下的解题过程均假设显示数字“1”时f、e亮，采用最小项表示法：

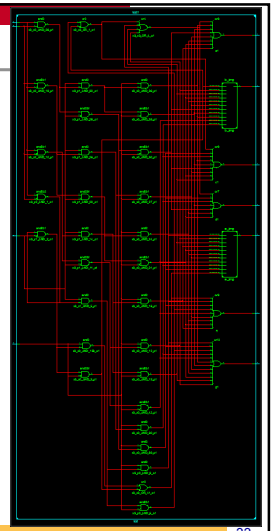
- g = m2 + m3 + m4 + m5 + m6 + m8 + m9 + m10 + m11 + m13 + m14 + m15 = (!x3 && !x2 && x1 && !x0) || (!x3 && !x2 && x1 && x0) || (!x3 && x2 && !x1 && !x0) || (!x3 && x2 && !x1 && x0) || (!x3 && x2 && x1 && !x0) || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0) || (x3 && x2 && !x1 && x0) || (x3 && x2 && x1 && !x0) || (x3 && x2 && x1 && x0)

三、问答与计算题

（3）采用结构描述法，用Verilog语言实现上述的数码管控制器。

解答：根据前面得到的各输出的最小项表达式，可以写出结构级描述的Verilog代码，参考代码见下页，下图是ISE综合之后的顶层模块图，右图是模块内部的电路结构。





➢33

三、问答与计算题

参考Verilog代码（结构描述法）：

```
module lcd(
  input x3, input x2, input x1, input x0,
  output a, output b, output c, output d, output e, output f, output g
  );

assign a=(!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && x1 && !x0) || (!x3 && !x2 && x1 && x0)
                || (!x3 && x2 && !x1 && x0) || (!x3 && x2 && x1 && !x0) || (!x3 && x2 && x1 && x0)
                || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0)
                || (x3 && x2 && !x1 && !x0) || (x3 && x2 && x1 && !x0) || (x3 && x2 && x1 && x0);
assign b=(!x3 && !x2 && !x1 && !x0) || (!x3 && x2 && !x1 && !x0) || (!x3 && x2 && !x1 && x0)
                || (!x3 && x2 && x1 && !x0) || (!x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0)
                || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0) || (x3 && x2 && !x1 && !x0)
                || (x3 && x2 && x1 && !x0) || (x3 && x2 && x1 && x0);
assign c=(!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && x1 && !x0) || (!x3 && x2 && x1 && !x0)
                || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0)
                || (x3 && x2 && !x1 && !x0) || (x3 && x2 && x1 && !x0) || (x3 && x2 && x1 && x0);
assign d=(!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && x1 && !x0) || (!x3 && !x2 && x1 && x0)
                || (!x3 && x2 && !x1 && x0) || (!x3 && x2 && x1 && !x0) || (x3 && !x2 && !x1 && !x0)
                || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && x0) || (x3 && x2 && !x1 && !x0)
                || (x3 && x2 && !x1 && x0) || (x3 && x2 && x1 && !x0);
assign e=(!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && x1 && x0) || (!x3 && !x2 && x1 && x0)
                || (!x3 && x2 && !x1 && !x0) || (!x3 && x2 && x1 && !x0) || (!x3 && x2 && x1 && x0)
                || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0)
                || (x3 && x2 && !x1 && !x0) || (x3 && x2 && !x1 && x0);
assign f=(!x3 && !x2 && !x1 && !x0) || (!x3 && !x2 && !x1 && x0) || (!x3 && !x2 && x1 && !x0)
                || (!x3 && !x2 && x1 && x0) || (!x3 && x2 && !x1 && !x0) || (!x3 && x2 && x1 && x0)
                || (x3 && !x2 && !x1 && !x0) || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0)
                || (x3 && x2 && !x1 && x0);
assign g=(!x3 && !x2 && x1 && !x0) || (!x3 && !x2 && x1 && x0) || (!x3 && x2 && !x1 && !x0)
                || (!x3 && x2 && !x1 && x0) || (!x3 && x2 && x1 && !x0) || (x3 && !x2 && !x1 && !x0)
                || (x3 && !x2 && !x1 && x0) || (x3 && !x2 && x1 && !x0) || (x3 && !x2 && x1 && x0)
                || (x3 && x2 && !x1 && x0) || (x3 && x2 && x1 && !x0) || (x3 && x2 && x1 && x0);

endmodule
```


➢34

计算机组成习题参考答案
——时序逻辑

➢35

一、填空题

❖1.根据在CP控制下，逻辑功能的不同，常把时钟触发器分为RS、D、JK等3种类型。

❖2.由与非门构成的基本RS触发器约束条件是 $\overline{R_D}+\overline{S_D}=1$（或$R_DS_D=0$）。

❖3.钟控JK触发器的特性方程为 $Q^{n+1}=J\overline{Q^n}+\overline{K}Q^n$。

❖4.时序逻辑电路按触发器时钟端的连接方式不同可以分为同步时序逻辑和异步时序逻辑两类。

❖5.n级移位寄存器可以存放n位二进制数据。

❖6.集成计数器的模值是固定的，但可以用反馈复位法和预置法来改变它们的模值。

❖7.由8级触发器构成的十进制计数器模值为100。

❖8.通过级联方法，把两片4位二进制计数器CT74161连接成为8位二进制计数器后，其最大模值是256。

➢36

二、选择题

❖1.若JK触发器的原始状态为0，欲在CP作用后保持0状态，则激励函数JK的值应是（C）。

A．J=1，K=1

B．J=0，K=0

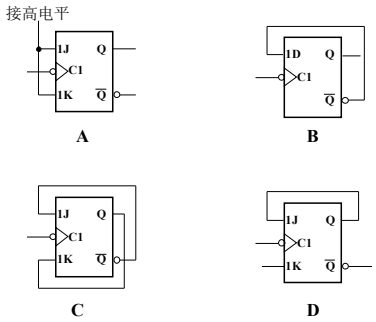
C．J=0，K=x

D．J=x，K=x

➢37

二、选择题

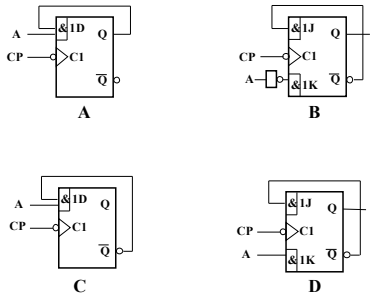
❖2.下列电路中，只有（D）不能实现 $Q^{n+1}=\overline{Q^n}$ 。



➢38

二、选择题

❖3.如下各触发器电路中，能实现 $Q^{n+1}=\overline{Q}^n+AQ^n$ 功能的电路是（B）。



➢39

二、选择题

❖4.用8级触发器可以记忆（D）种不同的状态。

A．8　　B．16　　C．128　　D．256

❖5.同步计数器是指（②）的计数器。

① 由同类型的触发器构成

② 各触发器时钟端连在一起，统一由系统时钟控制

③ 可用前级的输出做后级触发器的时钟

④ 可用后级的输出做前级触发器的时钟

❖6.由10级触发器构成的二进制计数器，其模值为（④）。

① 10　　② 20　　③ 1000　　④ 1024

➢40

二、选择题

❖7.同步4位二进制减法计数器的借位方程是 $B=\overline{Q_4}\overline{Q_3}\overline{Q_2}\overline{Q_1}$，则可知B的周期和正脉冲宽度为（②）。

① 16个CP周期和2个CP周期

② 16个CP周期和1个CP周期

③ 8个CP周期和8个CP周期

④ 8个CP周期和4个CP周期

❖8.已知Q_3、Q_2、Q_1、Q_0是同步十进制计数器的触发器输出，若以Q_3作进位C，则C的周期和正脉冲宽度是（②）。

① 10个CP脉冲，正脉冲宽度为1个CP周期

② 10个CP脉冲，正脉冲宽度为2个CP周期

③ 8个CP脉冲，正脉冲宽度为1个CP周期

④ 8个CP脉冲，正脉冲宽度为2个CP周期

➢41

二、选择题

❖9.一个4位移位寄存器原来的状态为0000，如果串行输入始终为1，则经过4个移位脉冲后寄存器的内容为（④）。

① 0001　② 0111　③ 1110　④ 1111

❖10.可以用来实现并/串转换和串/并转换的器件是（②）。

① 计数器　② 移位寄存器

③ 存储器　④ 全加器

❖11.用反馈复位法来改变8位二进制加法计数器的模值，可以实现（④）模值范围的计数器。

① 1~15　② 1~16　③ 1~32　④ 1~256

➢42

二、选择题

❖12.用Verilog HDL设计同步清除的计数器时，在always语句的敏感参数表中（①）。

① 需要列出时钟信号和清除信号标示符的有效边沿

② 只需要列出时钟信号标示符的有效边沿

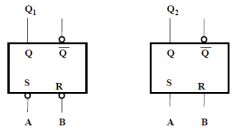
③ 只需要列出清除信号标示符的有效边沿

④ 只需要列出时钟信号或者清除信号标示符的有效边沿

➢43

三、分析与设计题

❖ 1.已知触发器的逻辑符号如图3.1所示，输入波形如图3.2所示，其中FF_1是由与非门构成的基本RS触发器，FF_2是由或非门构成的基本RS触发器，设触发器的初态均为0。试分别简要说明2个电路在不同的输入取值下的逻辑功能，并根据A、B输入波形画出Q_1、$\overline{Q_1}$、Q_2、$\overline{Q_2}$的输出波形图。



FF_1　FF_2

图3.1 基本RS触发器

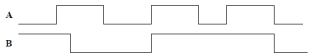


图3.2 A、B输入波形

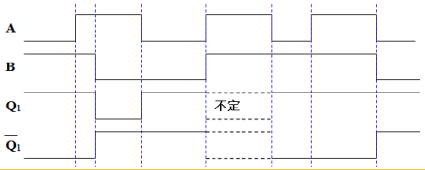
➢44

三、分析与设计题

解：

FF_1是由与非门构成的基本RS触发器，其中A是置1端 $\overline{S}_D$，B是置0端 $\overline{R}_D$，低电平有效，触发器的特性方程为 $Q^{n+1}=S_D+\overline{R}_DQ^n=\overline{A}+BQ^n$。

当A=0、B=1时，即$\overline{S}_D$=0、$\overline{R}_D$=1，触发器置1；当A=1、B=1时，即 $\overline{S}_D$=1、 $\overline{R}_D$=1，触发器保持不变；当A=1、B=0时，即 $\overline{S}_D$=1、$\overline{R}_D$=0，触发器置0，上述3个输入组合没有违反约束条件，根据它们的功能一步一步画出Q_1和 $\overline{Q}_1$的波形，而且 Q_1和 $\overline{Q}_1$的波形符合互非特性。当A=0、B=0时，即$\overline{S}_D$=0、$\overline{R}_D$=0，违反了触发器的约束条件，但根据与非门的特性可知，Q_1和 $\overline{Q}_1$均为高电平，破坏触发器输出互非特性。另外，如果A和B由低电平同时变为高电平时，因门的传输延迟时间的差异而产生竞争，使输出状态不能确定。根据上述分析画出FF_1的时序图。

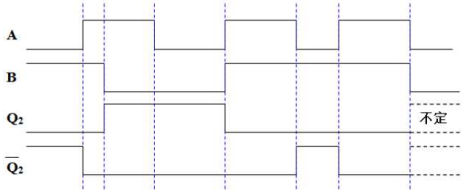


➢45

三、分析与设计题

FF_2是由或非门构成的基本RS触发器，其中A是置1端S_D，B是置0端R_D，高电平有效，触发器的特性方程为 $Q^{n+1}=S_D+\overline{R}_DQ^n=A+\overline{B}Q^n$。

当A=0、B=1时，即S_D=0、R_D=1，触发器置0；当A=0、B=0时，即S_D=0、R_D=0，触发器保持不变；当A=1、B=0时，即S_D=1、 R_D=0，触发器置1，上述3个输入组合没有违反约束条件，根据它们的功能一步一步画出Q_2和 $\overline{Q}_2$的波形，而且Q_2和$\overline{Q}_2$的波形符合互非特性。当A=1、B=1时，违反了触发器的约束条件，但根据或非门的特性可知，Q_2和 $\overline{Q}_2$ 均为低电平，破坏触发器输出互非特性。另外，如果A和B由高电平同时变为低电平时，因门的传输延迟时间的差异而产生竞争，使输出状态不能确定。根据上述分析画出FF_2的时序图。



➢46

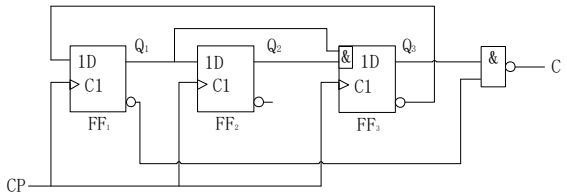
三、分析与设计题

❖2.分析下图所示电路，要求：

（1）写出分析过程，包括各级触发器的驱动方程和状态方程；

（2）画出状态转换表、状态转换图和时序图；

（3）说明电路特点。

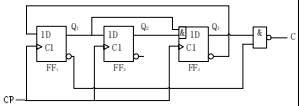


➢47

三、分析与设计题

解：

（1）各级触发器的驱动方程：



$$D_1=\overline{Q_3^n}$$
$$D_2=Q_1^n$$
$$D_3=Q_1^nQ_2^n$$

D触发器的特性方程为：

$$\begin{cases}Q^{n+1}=D(CP=1)\\ Q^{n+1}=Q^n(CP=0)\end{cases}$$

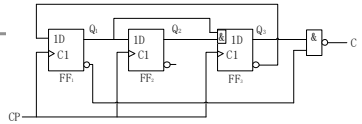
代入可得各级触发器的状态方程：

$$Q_1^{n+1}=\overline{Q_3^n}$$
$$Q_2^{n+1}=Q_1^n$$
$$Q_3^{n+1}=Q_1^nQ_2^n$$
$$C=\overline{Q_3^n}\overline{Q_1^n}$$

➢48

三、分析与设计题

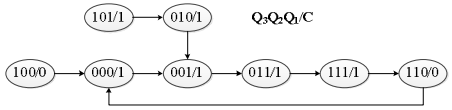
（2）状态转换表：



Q_3^n	Q_2^n	Q_1^n	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	C
0	0	0	0	0	1	1
0	0	1	0	1	1	1
0	1	0	0	0	1	1
0	1	1	1	1	1	1
1	0	0	0	0	0	0
1	0	1	0	1	0	1
1	1	0	0	0	0	0
1	1	1	1	1	0	1

$$Q_1^{n+1} = \overline{Q_3^n}$$
$$Q_2^{n+1} = Q_1^n$$
$$Q_3^{n+1} = Q_1^n Q_2^n$$
$$C = \overline{Q_3^n \overline{Q_1^n}}$$

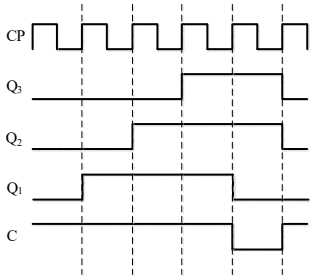
状态转换图：

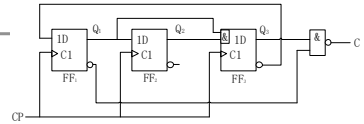


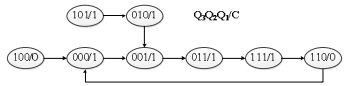
➢49

三、分析与设计题

（2）时序图：







（3）电路特点：具有自启动特性的同步五进制加法计数器

➢50

三、分析与设计题

❖3.集成4位二进制计数器CT74161的逻辑符号如图3.3所示，其功能表如表3.1所示，触发器输出低位到高位的次序是Q_0至Q_3，输出C＝ET $Q_3^n Q_2^n Q_1^n Q_0^n$。试用一片CT74161采用输出C预置法实现十二进制计数器，画出电路连接图。

表3.1 CT74161的功能表

$\overline{R}_D$	$\overline{LD}$	EP	ET	CP	功能
0	×	×	×	×	复位
1	0	×	×	↑	预置
1	1	0	0	↑	保持
1	1	0	1	↑	保持
1	1	1	0	↑	保持
1	1	1	1	↑	计数

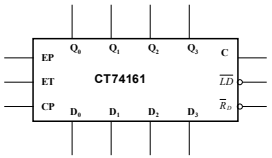


图3.3 4位二进制计数器CT74161的逻辑符号

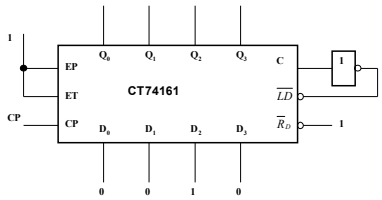
➢51

三、分析与设计题

解：已知CT74161的模值是16，改变后的模值是12，则预置数据值为：

$$16 - 12 = (4)_{10} = (0100)_2$$

由此得出的模12计数器电路如下图所示。



➢52

三、分析与设计题

❖ 4.分析如图3.4所示的状态转换图表示什么类型的状态机。并根据图3.4采用Verilog HDL设计六进制计数器电路。要求在程序中加上必要的注释。

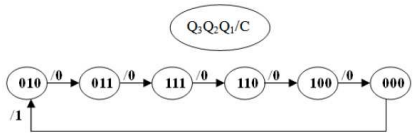


图3.4 六进制计数器电路的状态转换图

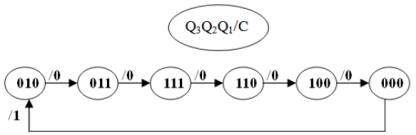
➢53

三、分析与设计题

解：

从状态转换图可以看出，电路没有输入，只进位输出C，某时刻的输出只与当时的状态有关，所以是Moore型状态机。

用Verilog HDL 按照题中所示的状态转换图设计六进制数器电路源程序见下页



➢54

三、分析与设计题

在源程序中，clk 是时钟输入端，上升沿有效；q是计数器的3位状态输出端，cout 是进位输出端。在程序中还设置了6个参数 (parameter) st0 、st1 、st2 、st3 、st4 和st5，分别代表题中所示状态转换图中的各个状态

```
module cnt6 (clk, q, cout);
   input clk;
   output reg[2:0] q;
   output reg cout;
   parameter[2:0] st0 = 'b010, st1 = 'b011, st2 = 'b111, st3 = 'b110, st4 = 'b100,
st5 = 'b000;//parameter也要声明位宽
   always@(posedge clk)
      begin
         case (q)//状态的转移
            st0 :      q = st1;
            st1 :      q = st2;
            st2 :      q = st3;
            st3 :      q = st4;
            st4 :      q = st5;
            st5 :      q = st0;
            default : q = st0;
         endcase
         if (q == st5)  cout = 1;//产生进位输出
         else           cout = 0;
      end
endmodule
```


➢55

三、分析与设计题

❖5.使用D触发器和与非门设计一个4人抢答逻辑电路，具体要求为：

（1）每个参赛者控制一个按钮，用按钮发出抢答信号；

（2）竞赛主持人控制另一个按钮，用于电路复位；

（3）竞赛开始后，先按动按钮者将对应的一个发光二极管点亮，此后其他3人再按动按钮对电路不起作用。

- 提示：抢答逻辑电路通常用于智力竞赛的抢答比赛中。由于参赛者按动按钮发出的信号不能自行保持，而且按动的动作可能有先后、长短之别，所以需要4个触发器分别保存4个参赛者按动按钮发出的信号。由于只要求触发器具有置1（抢答）、置0（复位）功能即可，所以采用RS、D、JK触发器均可，对结构类型也无特定要求。
- 此外，最先产生的抢答信号还应控制其它后来产生的抢答信号无法改变自身触发器的值，这样在抢答结束时就可根据各触发器的状态判断是哪位选手最先按下抢答按钮的。因此，需要将4个触发器的反相输出分别引入到一个与非门的输入端，并与输入时钟信号相与，再将与非门的输出作为各触发器的时钟信号。

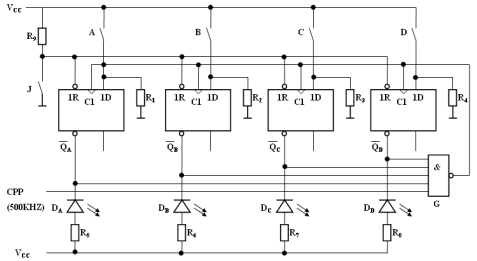
➢56

三、分析与设计题

解：

4人抢答逻辑电路如图所示，图中A、B、C、D四个按钮各由一名参赛者控制，按钮J由主持人控制。

电路工作过程：当A、B、C任何一个按钮首先按下时，对应的触发器被置1；这个触发器的 端随之变为低电平，将与非门G封锁；其余的触发器不再有CP信号输入，无法再置1。



➢57

三、分析与设计题

❖6.设计一个自动售货机控制器，每次可以任意投入一枚五分或1角的硬币。货物价格为20分，当投入足够的钱后，售货机吐出货物并找零钱。请完成下列任务：

（1）画出实现上述功能的状态机；

（2）列出二进制编码的状态转换表和输出逻辑真值表，给出次态每一位编码的逻辑函数表达式和输出逻辑函数表达式，并化简。

➢58

三、分析与设计题

解：（1）首先定义状态机：

假设各个状态定义为：

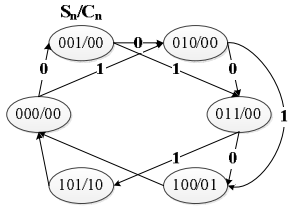
- $S_0(000)$：未投入硬币
- $S_1(001)$：已投入5分
- $S_2(010)$：已投入1角
- $S_3(011)$：已投入1角5分
- $S_4(100)$：已投入2角
- $S_5(101)$：已投入2角5分

输入定义：

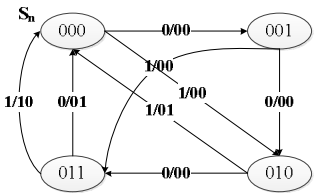
- $I_0(0)$：投入5分硬币
- $I_1(1)$：投入1角硬币

输出定义：

- $C_0(00)$：不吐出货物
- $C_1(01)$：吐出货物
- $C_2(10)$：吐出货物并找零5分



Moore型状态机

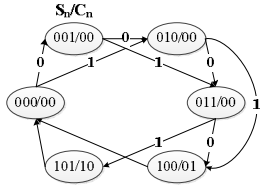


Mealy型状态机

➢59

三、分析与设计题

（2）二进制编码的状态转换表和输出逻辑真值表1：



当前状态($s_2s_1s_0$)/输出(c_1c_0)	输入(I)	次态($s_2's_1's_0'$)/输出(c_1c_0)
000/00	0	001/00
000/00	1	010/00
001/00	0	010/00
001/00	1	011/00
010/00	0	011/00
010/00	1	100/01
011/00	0	100/01
011/00	1	101/10
100/01	X	000/00
101/10	X	000/00

➢60

三、分析与设计题

（2）次态每一位编码的逻辑函数表达式1：

- 次态与当前状态和输入有关（以$s_2s_1s_0$I构成最小项）
- $s_2' = m_5 + m_6 + m_7$ = (!s2 && s1 && !s0 && I) || (!s2 && s1 && s0 && !I) || (!s2 && s1 && s0 && I)
- $s_1' = m_1 + m_2 + m_3 + m_4$ = (!s2 && !s1 && !s0 && I) || (!s2 && !s1 && s0 && !I) || (!s2 && !s1 && s0 && I) || (!s2 && s1 && !s0 && !I)
- $s_0' = m_0 + m_3 + m_4 + m_7$ = (!s2 && !s1 && !s0 && !I) || (!s2 && !s1 && s0 && I) || (!s2 && s1 && !s0 && !I) || (!s2 && s1 && s0 && I)

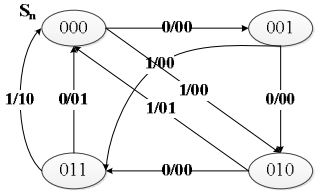
输出的逻辑函数表达式1：

- 输出只与当前状态有关（以$s_2s_1s_0$构成最小项）
- $c_1 = m_5$ = s2 && !s1 && s0
- $c_0 = m_4$ = s2 && !s1 && !s0

当前状态($s_2s_1s_0$)/输出(c_1c_0)	输入(I)	次态($s_2's_1's_0'$)/输出(c_1c_0)
000/00	0	001/00
000/00	1	010/00
001/00	0	010/00
001/00	1	011/00
010/00	0	011/00
010/00	1	100/01
011/00	0	100/01
011/00	1	101/10
100/01	X	000/00
101/10	X	000/00

三、分析与设计题

（2）二进制编码的状态转换表和输出逻辑真值表2：



当前状态($s_2s_1s_0$)	输入(I)	输出(c_1c_0)	次态($s_2's_1's_0'$)
000	0	00	001
000	1	00	010
001	0	00	010
001	1	00	011
010	0	00	011
010	1	01	000
011	0	01	000
011	1	10	000

三、分析与设计题

（2）次态每一位编码的逻辑函数表达式2：

- 次态与当前状态和输入有关（以$s_2s_1s_0$I构成最小项）
- $s_2' = 0$
- $s_1' = m_1 + m_2 + m_3 + m_4$ = (!s2 && !s1 && !s0 && I) || (!s2 && !s1 && s0 && !I) || (!s2 && !s1 && s0 && I) || (!s2 && s1 && !s0 && !I)
- $s_0' = m_0 + m_3 + m_4$ = (!s2 && !s1 && !s0 && !I) || (!s2 && !s1 && s0 && I) || (!s2 && s1 && !s0 && !I)

输出的逻辑函数表达式2：

- 输出与当前状态和输入都有关（以$s_2s_1s_0$I构成最小项）
- $c_1 = m_7$ = !s2 && s1 && s0 && I
- $c_0 = m_5 + m_6$ = (!s2 && s1 && !s0 && I) || (!s2 && s1 && s0 && !I)

当前状态($s_2s_1s_0$)	输入(I)	输出(c_1c_0)	次态($s_2's_1's_0'$)
000	0	00	001
000	1	00	010
001	0	00	010
001	1	00	011
010	0	00	011
010	1	01	000
011	0	01	000
011	1	10	000

计算机组成习题参考答案
——主存储器

第1题

❖说明存取时间与存取周期的区别。

- 存取时间：读或者写操作所用的时间
- 存取周期：两次访问存储单元的最小时间间隔

❖什么是存储器的带宽？若某存储器的数据总线宽度为64位，存取周期为100ns，则该存储器的带宽是多少？

- 存储器带宽：单位时间内访问的存储量
- 计算：$64/(100\times10^{-9})=640Mb/s$

第2题

❖某机字长32位，其存储容量是64KB，按字编址其寻址范围是多少？若主存以字节编制，试画出主存字地址和字节地址的分配情况。

- 容量为64KB，即按字节编址，寻址范围就是64K字节
- 字长为32位，因此，按字编址，寻址范围就是64K×8/32=16K字
- 每个字包含4个字节，主存字地址和字节地址的分配情况：

字地址	字节地址			
0	0	1	2	3
4	4	5	6	7
……	……	……	……	……
65532	65532	65533	65534	65535

第3题

❖一个容量为16K×32位的存储器，分别需要几条地址线和数据线？

❖如果该存储器采用二维地址结构，且行地址和列地址的位数相同，则译码器输出的行选择线和列选择线分别有多少条？

❖若选用下列不同规格的存储芯片来实现该存储器，需要各存储芯片的数目以及它们的排列方式分别是怎样的？

1K×4

2K×8

4K×4

16K×1

4K×8

8K×8

第3题

❖一个容量为16K×32位的存储器，分别需要几条地址线和数据线？

- 地址线的决定因素：存储器存储字单元的数量
- $16K=2^{14}$，于是地址线为14根
- 数据线的决定因素：存储器存储单元的数据宽度
- 数据线为32根

第3题

❖一个容量为16K×32位的存储器，分别需要几条地址线和数据线？

❖如果该存储器采用二维地址结构，且行地址和列地址的位数相同，则译码器输出的行选择线和列选择线分别有多少条？

- 存储容量16K决定需要14条地址线
- 数据宽度32位决定需要32条数据线
- 行地址位数=列地址位数
- $2^x \times 2^x = 16K \rightarrow x = 7$
- 对应的行选择线$2^7 = 128$，列选择线$2^7 = 128$

第3题

❖一个容量为16K×32位的存储器，分别需要几条地址线和数据线？

❖如果该存储器采用二维地址结构，且行地址和列地址的位数相同，则译码器输出的行选择线和列选择线分别有多少条？

❖若选用下列不同规格的存储芯片来实现该存储器，需要各存储芯片的数目以及它们的排列方式分别是怎样的？

1K×4

2K×8

4K×4

16K×1

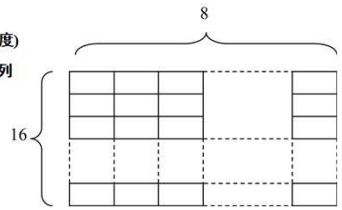
4K×8

8K×8

第3题

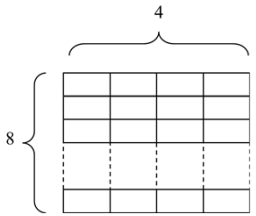
1K×4→16K×32

- 需要进行字扩展(深度)和位扩展(宽度)
- 将全部存储器芯片看成一个二维阵列
- 字扩展需要的行数：16K/1K=16
- 位扩展需要的列数：32/4=8
- 总的存储器芯片数量=16×8=128
- 或者
- 所有存储器芯片的容量之和=目标存储器容量
- 存储器芯片数量=目标存储器容量 / 单个芯片容量
- 存储器芯片数量=(16K×32)/(1K×4)=16×8=128

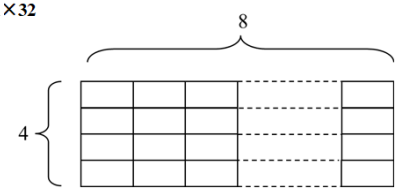


第3题

2K×8→16K×32

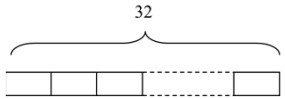


4K×4→16K×32

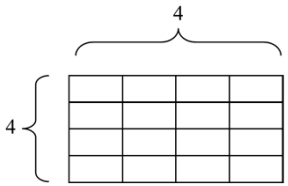


第3题

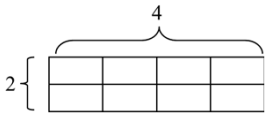
16K×1→16K×32



4K×8→16K×32



8K×8→16K×32



➢73

第4题

❖现有一容量为256K×8的DRAM存储芯片，试回答：

该芯片包含多少个字单元？

该芯片包含多少个二进制存储单元电路(存储位元)？

该芯片的刷新地址计数器应该是多少位？

若该DRAM芯片的存取周期为0.25us，试问采用集中刷新、分散刷新及异步刷新三种方式的刷新间隔各为多少？

➢74

第4题

❖现有一容量为256K×8的DRAM存储芯片，试回答：

该芯片包含多少个字单元？

- 256K字单元

该芯片包含多少个二进制存储单元电路(存储位元)？

- $256K \times 8 = 2^{21}$

该芯片的刷新地址计数器应该是多少位？

- 刷新地址计数器产生的是用于DRAM刷新的行地址，因此，它的位数就是行地址的位数。而DRAM芯片行地址和列地址通常会共享同一组管脚，因此，行地址和列地址的位数是相等的，是整个芯片地址数量的一半
- $256K = 2^{18}$→行地址为9位→刷新地址计数器为9位

若该DRAM芯片的存取周期为0.25us，试问采用集中刷新、分散刷新及异步刷新三种方式的刷新间隔各为多少？

➢75

第4题

❖现有一容量为256K×8的DRAM存储芯片，试回答：

该芯片包含多少个字单元？

该芯片包含多少个二进制存储单元电路(存储位元)？

该芯片的刷新地址计数器应该是多少位？

若该DRAM芯片的存取周期为0.25us，试问采用集中刷新、分散刷新及异步刷新三种方式的刷新间隔各为多少？

- 该芯片共有$2^9 = 512$行，刷新是按行进行的
- 每个刷新周期内，所有行必须至少被刷新一次
- 集中刷新：在刷新周期的某一时间段集中刷新所有行，因此，刷新的间隔时间即为刷新周期，一般取2ms
- 分散刷新：刷新分散到每个存取周期，每个存取周期刷新一行。故刷新的间隔时间为 $512 \times 0.25us = 128us$
- 异步刷新：只要保证在一个刷新周期内将存储芯片所有行刷新一遍即可。因此，刷新的间隔时间仍为刷新周期，一般取2ms

➢76

第5题

❖画出1K×4位的存储器芯片组成一个64K×8位的存储器逻辑框图。要求64K分成4个页面，每个页面分16组，指出共需多少片存储器芯片。

1K×4芯片组成64K×8存储器

需要进行字扩展和位扩展

字扩展：64K/1K＝64

位扩展：8/4 =2

芯片数：(64K×8)/(1K×4)=128

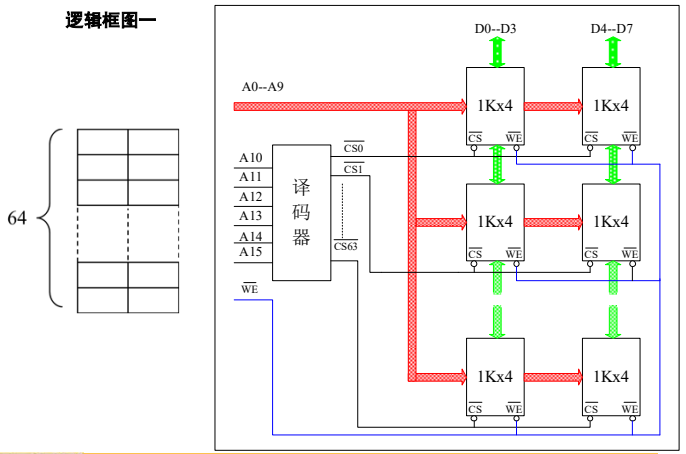
将64K字空间分为4个页面

整个存储器分成4个16K×8的小存储器

➢77

第5题

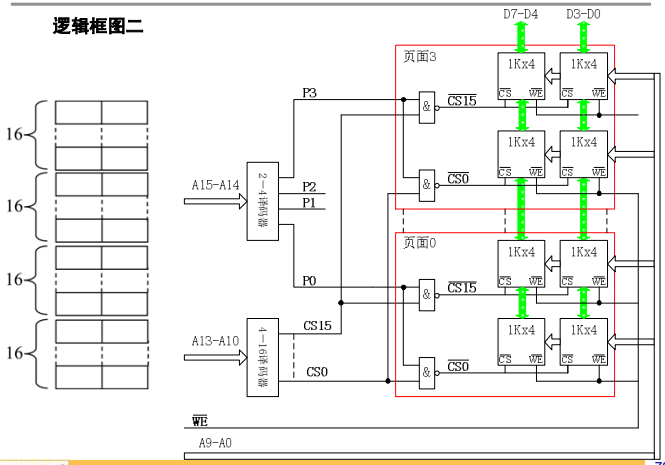
逻辑框图一



➢78

第5题

逻辑框图二



➢79

第6题

❖设有一个64K×16位的RAM芯片，问该芯片共有多少个基本单元电路(简称存储基元)？欲设计一种具有上述同样多存储基元的芯片，要求对芯片字长的选择应满足地址线和数据线的总和为最小，试确定这种芯片的地址线和数据线，并说明有几种解答。

该芯片的存储基元总数=64K×16位=1024K=2^{20}(个)

如要满足地址线和数据线总和最小，应尽量把存储元安排在字向，因为地址位数和字数成2的幂的关系，可较好地压缩线数

设地址为n位，数据为b位，则：$2^n \times b = 2^{20}$，即：$b = 2^{20-n}$

(n,b)的组合有(20,1)，(19,2)，(18,4)，(17,8)，……

n+b最小的组合为(20,1)和(19,2)，所以有两种解答。

➢80

第7题

❖某8位微型计算机地址码为18位，若使用4K×4的RAM芯片组成模块板结构的存储器，问：

该机所允许的最大主存空间是多少？

- 最大主存空间=2^{18}×8=256K×8位=256KB

若每个模板为32K×8位，共需多少模板块？

- (256K×8)/(32K×8)=8

每个模板块内共有几片RAM芯片？

- (32K×8)/(4K×4)=16

共有多少片RAM？

- 16×8=128或(256K×8)/(4K×4)=128

CPU如何选择各模板块？

➢81

第7题

❖某8位机地址码为18位，若使用4K×4的RAM芯片组成模块板结构的存储器，问：

CPU如何选择各模板块？

- CPU地址→(模板号，模板内部偏移)
- 模板块数→模板号
- 8个模板→3位模板号
- 模板存储单元数量→模板内部偏移位数
- 32K→15位偏移
- CPU地址→(3位模板号，15位模板内部偏移)
- CPU地址高3位选择8个模板(3-8译码器)
- CPU地址范围与模板对应关系如表所示

CPU地址范围（高3位）	模板
00000H-07FFFH（00,0）	0
08000H-0FFFFH（00,1）	1
10000H-17FFFH（01,0）	2
18000H-1FFFFH（01,1）	3
20000H-27FFFH（10,0）	4
28000H-2FFFFH（10,1）	5
30000H-37FFFH（11,0）	6
38000H-3FFFFH（11,1）	7

➢82

第8题

❖设CPU有16根地址线，8根数据线，并用MREQ#作访存控制信号，R/W#作读写命令信号，现有存储芯片ROM(2K×8, 4K×4, 8K×8)和RAM(1K×4, 2K×8, 4K×8)及74138译码器和其他门电路。试选择合适芯片，并画出CPU和芯片连接图。要求：

最小4K地址为系统程序区，4096~16383地址范围为用户程序区

指出选用的存储芯片类型及数量

画出片选逻辑

➢83

第8题

16根地址线，8根数据线

- CPU具有64K×8位的寻址能力
- 主存储器容量上限是64K×8位
- 位扩展宽度：8位

4K地址为系统程序区，4096～16383为用户程序区

- ROM地址空间：0000H～0FFFH，容量：4K×8
- RAM地址空间：1000H～3FFFH，容量：12K×8

ROM扩展(4K×8系统程序区)

- 4K×4，2K×8均可，不能选8K×8
- 选择4K×4，2片，位扩展

RAM扩展(12K×8用户程序区)

- 1K×4，2K×8，4K×8均可
- 选择4K×8，3片，字扩展

➢84

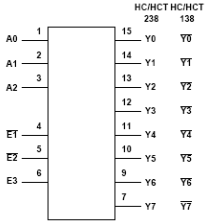
第8题

存储器芯片地址空间

- **共需4个片选**

0000H～0FFFH	ROM(2片4K×4)	2片选连接在一起
1000H～1FFFH	RAM(4K×8)	独立片选
2000H～2FFFH	RAM(4K×8)	独立片选
3000H～3FFFH	RAM(4K×8)	独立片选

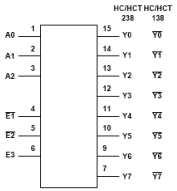
使用74138译码器



➢85

第8题

74138的真值表



INPUTS						OUTPUTS							
ENABLE			ADDRESS										
E3	~E2	~E1	A2	A1	A0	~Y0	~Y1	~Y2	~Y3	~Y4	~Y5	~Y6	~Y7
×	×	H	×	×	×	H	H	H	H	H	H	H	H
L	×	×	×	×	×	H	H	H	H	H	H	H	H
×	H	×	×	×	×	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	L	H	H	H	H	H	L	H	H	H	H
H	L	L	H	L	L	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	H	L	H	H
H	L	L	H	H	L	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

➢86

第8题

译码器输出直接作为片选信号

- **E3、E2#、E1#用做访问控制信号**
- **E2：H**
- **E1#、E0#：MREQ#**
- **A2、A1、A0用做读写命令信号**
- **A2：L**
- **A1、A0：CPU地址线A13，A12**

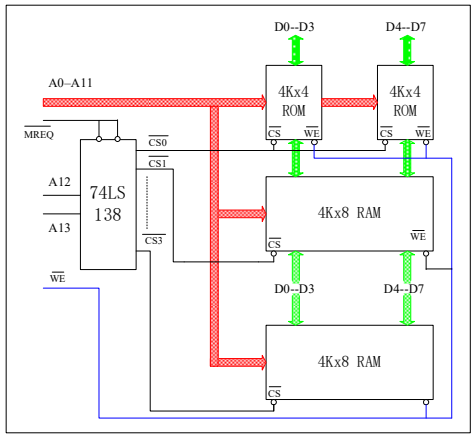
译码器输出：

138输出	连接关系
Y0#	ROM片选
Y1#	RAM片选
Y2#	RAM片选
Y3#	RAM片选

➢87

第8题

片选逻辑



➢88

第9题

❖CPU假设同第8题，现有8片8K×8位的RAM芯片与CPU相连，试回答：

用74138译码器画出CPU与存储芯片的连接图

写出每片RAM的地址范围

如运行时发现不论往哪片RAM写入数据后，以A000H为起始地址的存储芯片都有与其相同的数据，分析故障原因

根据前面的连线图，若出现地址A13与CPU断线，并搭接到高电平上，将出现什么后果？

➢89

第9题

8片8K×8位的RAM芯片与CPU相连

- 主存储器总容量为64K×8
- CPU的数据线为8位，地址线为16位
- 8片RAM只能是字扩展
- 每片RAM占用8KB的地址空间
- 每片RAM的片选信号均为独立

存储器芯片地址空间如表所示

0000H～1FFFH	RAM(8K×8)
2000H～3FFFH	RAM(8K×8)
4000H～5FFFH	RAM(8K×8)
6000H～7FFFH	RAM(8K×8)
8000H～9FFFH	RAM(8K×8)
A000H～BFFFH	RAM(8K×8)
C000H～DFFFH	RAM(8K×8)
E000H～FFFFH	RAM(8K×8)

74LS138译码器

- CPU地址A15, A14, A13对应74LS138的A2, A1, A0
- Y0#…Y7#对应8个RAM芯片的片选

➢90

第9题

如运行时发现不论往哪片RAM写入数据后，以A000H为起始地址的存储芯片都有与其相同的数据，分析故障原因

- CPU地址A15, A14, A13对应74LS138的A2, A1, A0
- Y0#…Y7#对应8个RAM芯片的片选
- 以A000H为起始地址的存储芯片始终被选中
- Y5#在写入操作过程中，恒为低电平

Y0#	0000H～1FFFH	RAM(8K×8)
Y1#	2000H～3FFFH	RAM(8K×8)
Y2#	4000H～5FFFH	RAM(8K×8)
Y3#	6000H～7FFFH	RAM(8K×8)
Y4#	8000H～9FFFH	RAM(8K×8)
Y5#	A000H～BFFFH	RAM(8K×8)
Y6#	C000H～DFFFH	RAM(8K×8)
Y7#	E000H～FFFFH	RAM(8K×8)

➢91

第9题

根据前面的连线图，若出现地址A13与CPU断线，并搭接到高电平上，将出现什么后果？

- 74138译码器的输入将只存在4种可能：001、011、101、111
- 000、001→001
- 010、011→011
- 100、101→101
- 110、111→111

INPUTS						OUTPUTS							
ENABLE			ADDRESS										
E3	~E2	~E1	A2	A1	A0	~Y0	~Y1	~Y2	~Y3	~Y4	~Y5	~Y6	~Y7
×	×	H	×	×	×	H	H	H	H	H	H	H	H
L	×	×	×	×	×	H	H	H	H	H	H	H	H
×	H	×	×	×	×	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	L	H	H	H	H	H	L	H	H	H	H
H	L	L	H	L	L	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	H	L	H	H
H	L	L	H	H	L	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

➢92

第9题

根据前面的连线图，若出现地址A13与CPU断线，并搭接到高电平上，将出现什么后果？

- 对0000～1FFF读写操作实际访问的是RAM1，依次类推
- 即只能正确访问地址中A13=1的RAM芯片1、3、5、7，而访问不到地址中A13=0的RAM芯片0、2、4、6

0000H～1FFFH	—
2000H～3FFFH	RAM1
4000H～5FFFH	—
6000H～7FFFH	RAM3
8000H～9FFFH	—
A000H～BFFFH	RAM5
C000H～DFFFH	—
E000H～FFFFH	RAM7

➢93

计算机组成习题参考答案

--MIPS汇编

➢94

第1题：填空

(a) 将10进制数35转换为8位二进制数是 00100011

(b) 将二进制数00010101转换为16进制数是 0x15

(c) 将10进制数-35转换为8位二进制补码是 11011101

(d) 将8进制数 204 转换为10进制数是 132

(e) 请判断以下两个补码表示的二进制数做二进制加法后是否溢出：是

```
  01101110
  00011010
  10001000
```

(f) 对8位16进制数0x88做符号扩展成16位数是：0xFF88

(g) 下列代码段存储在内存中，起始地址为0x00012344，分支指令执行后PC的两个可能的值分别是：0x00012344 和 0x 00012354。同时，请在注释位置用伪代码形式对每条指令做出描述。

```
loop:   lw      $t0, 0($a0)     # t0 = Mem[a0]
        addi    $a0, $a0, 4     # a0 = a0 +4
        andi    $t1, $t0, 1     # t1 = t0 & 1 "Extract LSD"
        beqz    $t1, loop       # if  t0 is an even go to loop
```


➢95

第2题

❖ 将下列汇编语言指令翻译成机器语言代码，以16进制表示

```
loop:   addu    $a0, $0, $t0        # 0x00082021
        ori     $v0, $0, 4          # 0x34020004
        syscall                     # 0x0000000C
        addi    $t0, $t0, -1        # 0x2108FFFF
        bnez    $t0, loop           # 0x1500FFFB
        andi    $s0, $s7, 0xffc0    # 0x32F0FFC0
        or      $a0, $t7, $s0       # 0x01F02025
        sb      $a0, 4($s6)         # 0xA2C40004
        srl     $s7, $s7, 4         # 0x0017B902
```


➢96

第3题

❖ 写一个MIPS汇编程序，要求对内存以“example100”为标签（label）的数据段中前100个字（words）的数据求和，并将结果存入紧跟在这100个字之后的内存中。

伪代码：

```
$a0 = &example100;          # “&” means “Address of”
$t0 = 0;
For ($t1= 100;  $t1 > 0;  $t1= $t1- 1)
  {
  $t0 = $t0 + Mem($a0);
  $a0 = $a0 + 4;
  }
Mem($a0) = $t0;
```


➢97

```
Label          Op-Code Dest, S1, S2        Comments
               .data
example100:    .space   400
result:        .word
               .globl   main
               .text
main:
               la       $a0, example100    # Load address pointer
               li       $t0, 0             # Clear sum
               li       $t1, 100           # Initialize loop count
loop:
               lw       $t2, 0($a0)        # $t2 = Mem(a0)
               add      $t0, $t0, $t2      # $t0 = $t0 + $t2
               addi     $a0, $a0, 4        # Inc. address pointer
               addi     $t1, $t1, -1       # Dec. loop count
               bgtz     $t1, loop          # if ($t1 > 0) branch
               sw       $t0, 0($a0)        # Store the result
               li       $v0, 10            # End of program
               syscall
```


➢98

第4题

❖ 写一段MIPS汇编语言代码，将内存中“SRC”标签开始的100个字的一块数据转移到内存中另一块以“DEST”标签开始的空间中。

伪代码

```
$a1= &SRC;          # “&” means “Address of”
$a2= &DEST;
for ($t0 = 100; $t0 > 0; $t0 =$t0 -1)
      {$t1 = Mem($a1);
       Mem($a2) = $t1;
        $a1= $a1 + 4;
        $a2= $a2 + 4;
  }
```


➢99

```
Label          Op-Code Dest, S1, S2        Comments
               .data
SRC:           .space   400
DEST:          .space   400
               .globl   main
               .text
main:
               la       $a1, SRC           # $a1 = &SRC
               la       $a2, DEST          #$a2 = &DEST
               li       $t0, 100           #$t0 = 100
loop:          lw       $t1, 0($a1)        #$t1= Mem($a1)
               sw       $t1, 0($a2)        #Mem($a2) = $t1
               addi     $a1, $a1,4         #$a1 = $a1+4
               addi     $a2, $a2,4         #$a2 = $a2+4
               addi     $t0, $t0,  -1      # $t0 = $t0 - 1
               bgtz     $t0, loop          #Branch if $t0 > 0
               li       $v0, 10
               syscall
```


➢100

第5题

- 写一个MIPS函数ABS，通过$a0传入一个32位整数，将这个数的绝对值存回$a0。再写一段主程序，调用两次ABS并输出结果，每次传给ABS的数不同。

函数伪代码

```
Function ABS($a0);
if ($a0 < 0) $a0 = $0 - $a0;
return;
```


➢101

```
Label   Op-Code  Dest. S1, S2        Comments
        .text
ABS:    bgez     $a0, return        # If ($a0 >= 0) done
        sub      $a0, $0, $a0       # $a0 = 0 - $a0
return: jr       $ra                #Return
########################################
        .globl   main
        .text
main:   li       $a0, -9876
        jal      ABS
        li       $v0, 1             # Output result
        syscall
        li       $a0, 9876
        jal      ABS
        li       $v0, 1             # Output result
        syscall
        li       $v0,10             # End of program
        syscall
```


➢102

第6题

- 写一个函数FIB(N, &array)向内存中的一个数组（array）存入斐波那契数列的前N个元素。N和array的地址分别通过$a0和$a1传递进来。斐波那契数列的前几个元素是：1，1，2，3，5，8，13，.....

伪代码

```
Mem($a1) = 1;
Mem($a1 + 4) = 1;
for ($a0 = $a0 - 2; $a0 > 0; $a0 = $a0-1)
  {
  Mem($a1+8) = Mem($a1) + Mem($a1+4);
  $a1 = $a1 + 4;}
return;
```


➢103

```
Label   Op-Code  Dest. S1, S2       Comments
fib:    li       $t0, 1
        sw       $t0, 0($a1)
        sw       $t0, 4($a1)
        addi     $a0, $a0, -2
loop:
        lw       $t0, 0($a1)
        lw       $t1, 4($a1)
        add      $t0, $t0, $t1
        sw       $t0, 8($a1)
        addi     $a1, $a1, 4
        addi     $a0, $a0, -1
        bgtz     $a0, loop
        jr       $ra
```


➢104

第7题

❖ 写一个函数，从$a0，$a1，和$a2中接受传递过来的3个32位整数，按从小到大排序后存回$a0-$a2。

伪代码

```
Function Order($a0,$a1,$a2);
If ($a0 > $a1) exchange $a0 and $a1;
if ($a1 > $a2) exchange $a1 and $a2 else return;
If ($a0 > $a1) exchange $a0 and $a1;
return;
```



```
Label    Op-Code  Dest, S1, S2      Comments
         .text
order:
         ble      $a0, $a1, next
         move     $t0, $a1
         move     $a1, $a0
         move     $a0, $t0
next:
         ble      $a1, $a2, done
         move     $t0, $a2
         move     $a2, $a1
         move     $a1, $t0

         ble      $a0, $a1, done
         move     $t0, $a1
         move     $a1, $a0
         move     $a0, $t0
done:    jr       $ra
```


计算机组成习题参考答案

——单周期处理器设计

一、题目

❖ 在基本的单周期MIPS实现中，不同的指令使用不同的硬件单元。

❖ 根据如下指令回答下列3个问题。

	指令	解释
a.	add Rd,Rs,Rt	Reg[Rd]=Reg[Rs]+Reg[Rt]
b.	lw Rt,Offs(Rs)	Reg[Rt]=Mem[Reg[Rs]+Offs]

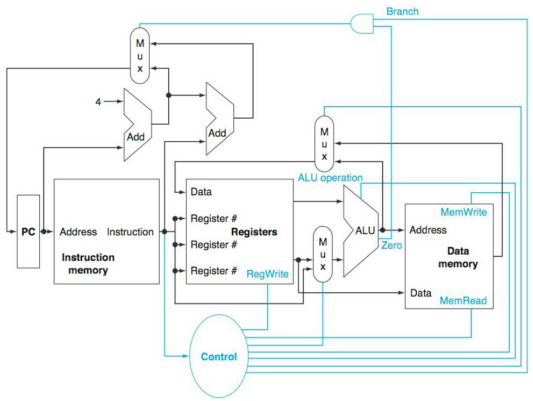
① 对上述指令而言，图1中的控制单元要产生哪些控制信号？

② 对上述指令而言，要用到哪些功能单元？

③ 哪些功能单元会产生输出，但输出不会被以上指令用到？对以上指令而言，哪些功能单元不产生任何输出？

一、题图

❖ 图1



➢109

一、解答 1

	指令	解释
a.	add Rd,Rs,Rt	Reg[Rd]=Reg[Rs]+Reg[Rt]
b.	lw Rt,Offs(Rs)	Reg[Rt]=Mem[Reg[Rs]+Offs]

① 对上述指令而言，图1中的控制单元要产生哪些控制信号？

- 控制单元产生的信号为：

	RegWrite	MemRead	ALUMux	MemWrite	ALUOp	RegMux	Branch
a.	1	0	0(Reg)	0	Add	0(ALU)	0
b.	1	1	1(Imm)	0	Add	1(Mem)	0

- ALUMux代表连接在ALU输入端的多选器的控制信号。其中，0(Reg)代表多选器选择的是寄存器堆的输出；1(Imm)代表多选器选择的是指令存储器中的立即数。
- RegMux代表连接在寄存器堆的Data输入端的多选器的控制信号。其中，0(ALU)代表多选器选择的是ALU的输出，1(Mem)代表多选器选择的是数据存储器的输出。

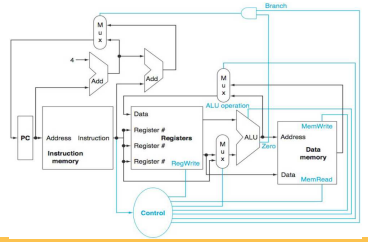
➢110

一、解答2

	指令	解释
a.	add Rd,Rs,Rt	Reg[Rd]=Reg[Rs]+Reg[Rt]
b.	lw Rt,Offs(Rs)	Reg[Rt]=Mem[Reg[Rs]+Offs]

② 对上述指令而言，会用到哪些功能单元？

- 对于指令a. 除数据存储器和分支加法器(图1中靠右侧的Add)之外的所有单元
- 对于指令b. 除分支加法器之外的所有单元



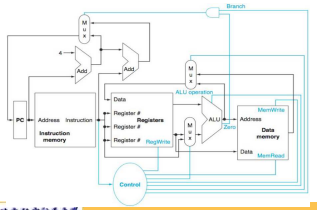
➢111

一、解答3

	指令	解释
a.	add Rd,Rs,Rt	Reg[Rd]=Reg[Rs]+Reg[Rt]
b.	lw Rt,Offs(Rs)	Reg[Rt]=Mem[Reg[Rs]+Offs]

③ 哪些功能单元会产生输出，但输出不会被以上指令用到？对以上指令而言，哪些功能单元不产生任何输出？

	产生输出但没有用到	没有输出
a.	分支加法器	数据存储器
b.	分支加法器，寄存器堆的第二个输出端 (图1中寄存器堆两个输出端中下面的一个)	无



➢112

一、题目

- 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。
- 假设每个单元的延迟时间如下表所示，回答下列3个问题。

	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

④ 对一条MIPS的与指令(AND)而言，关键路径是什么？

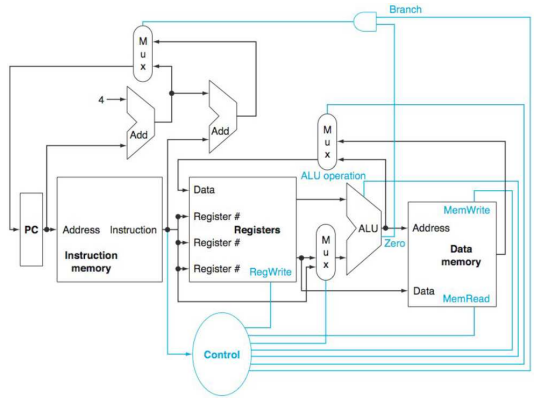
⑤ 对一条MIPS的装载指令(LW)而言，关键路径是什么？

⑥ 对一条MIPS的相等则分支指令(BEQ)而言，关键路径是什么？

➢113

一、题图

- 图1



➢114

一、解答4

	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

④ 对一条MIPS的与指令(AND)而言，关键路径是什么？

- 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。

➢115

一、解答5

	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

⑤ 对一条MIPS的装载指令(LW)而言，关键路径是什么？

- 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。

➢116

一、解答6

	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

⑥ 对一条MIPS的相等则分支指令(BEQ)而言，关键路径是什么？

- 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。

➢117

二、题目

- 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。

	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)

① 对上述指令而言，哪些已有的单元还可以被使用？

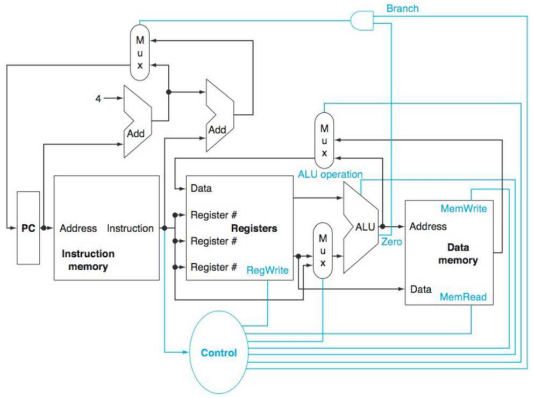
② 对上述指令而言，还需要增加哪些功能单元？

③ 为了支持这些指令，需要在控制单元增加哪些信号？

➢118

二、题图

- 图1



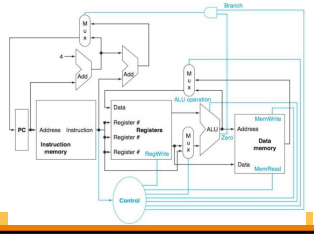
➢119

二、解答1

	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)

① 对上述指令而言，哪些已有的单元还可以被使用？

- 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



➢120

二、解答2

	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)

② 对上述指令而言，还需要增加哪些功能单元？

- 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

二、解答3

	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)

③ 为了支持这些指令，需要在控制单元增加哪些信号？

- 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

二、题目

❖ 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%

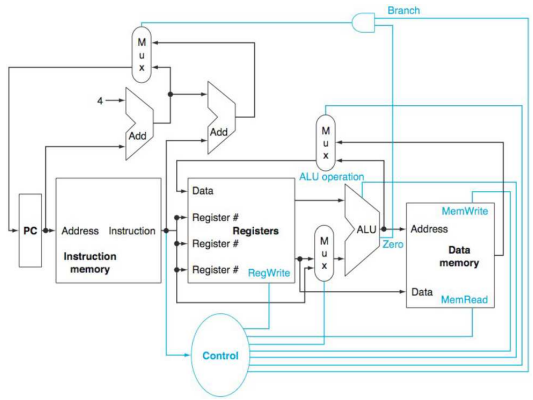
④ 改进前后的时钟周期分别是多少？

⑤ 改进后将获得多大的加速比？

⑥ 比较改进前后的性能/价格比，进行这样的改进是否有意义？

二、题图

❖ 图1



二、解答4

	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%

④ 改进前后的时钟周期分别是多少？

- 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。

二、解答5

	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%

⑤ 改进后将获得多大的加速比？

- 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

二、解答6

	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%

⑥ 比较改进前后的性能/价格比，进行这样的改进是否有意义？

- 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。

三、题目

❖ 下表给出了实现处理器数据通路的逻辑单元延迟。试根据下表的两种情况分别回答下列问题。

	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

① 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？

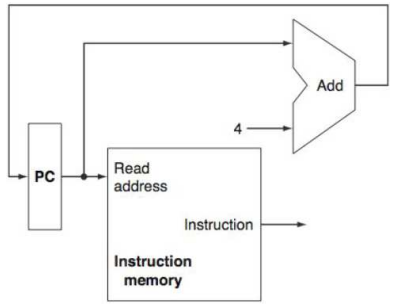
② 考虑一个与图3类似的数据通路，但是假设处理器只需处理无条件相对跳转指令，那么时钟周期是多少？

③ 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

三、题图

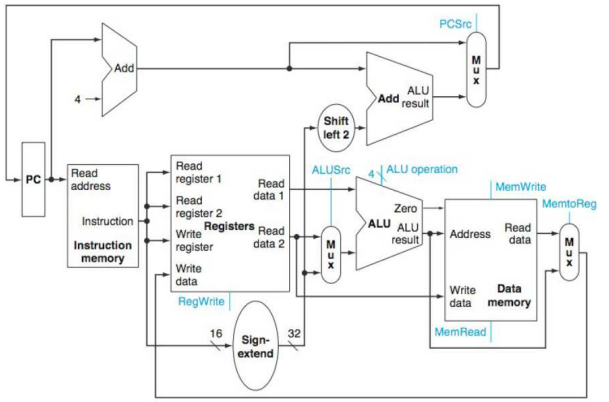
❖图2



➢129

三、题图

❖图3



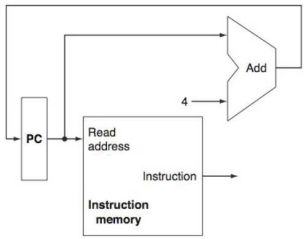
➢130

三、解答1

	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

① 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？

- 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。



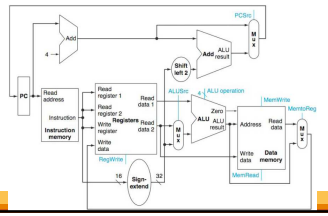
➢131

三、解答2

	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

② 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？

- 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。



➢132

三、解答3

	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

③ 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)

- 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。
- 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。

三、题目

❖ 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

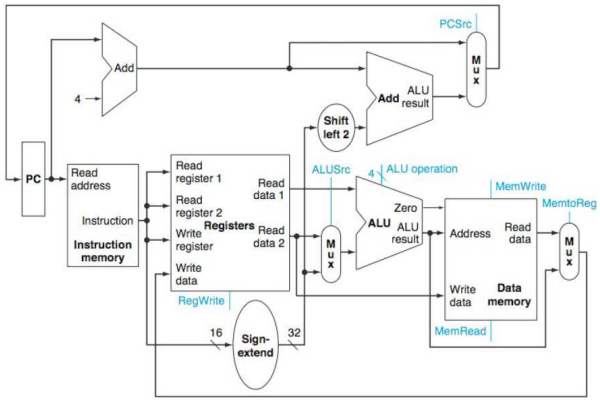
④ 哪些类型的指令需要该单元？

⑤ 对哪些类型的指令而言，该单元位于关键路径上？

⑥ 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。

三、题图

❖图3

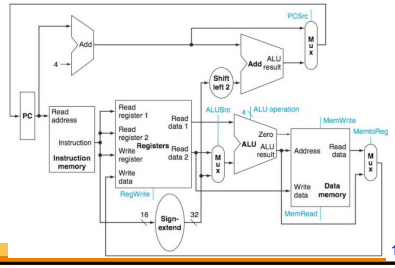


三、解答4

	单元
a.	执行加4的加法器(对PC)
b.	数据存储器

④ 哪些类型的指令需要该单元？

- a.所有指令。
- b.与存取有关的指令，如：Lw, Sw等。

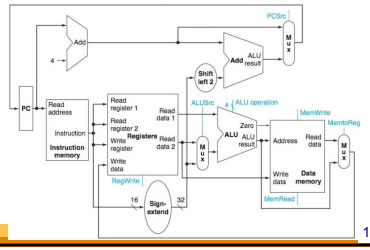


三、解答5

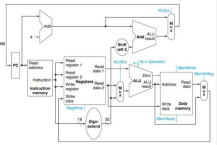
	单元
a.	执行加4的加法器(对PC)
b.	数据存储器

⑤ 对哪些类型的指令而言，该单元位于关键路径上？

- 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。



三、解答6



	单元
a.	执行加4的加法器(对PC)
b.	数据存储器

⑥ 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。

- 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、Mux，其延迟为100ps＋30ps＝150ps，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

四、题目

❖ 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。

	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

① 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？

② 如果仅需支持lw类指令，时钟周期是多少？

③ 如果必须支持add、beq、lw和sw指令，时钟周期是多少？

四、解答1

	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

① 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？

- 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)

四、解答2

	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

② 如果仅需支持lw类指令，时钟周期是多少？

- 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)

➢141

四、解答3

	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

③ 如果必须支持add、beq、lw和sw指令，时钟周期是多少？

- 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

➢142

四、题目

❖ 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%

④ 数据存储器平均用了多少时钟周期？

⑤ 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？

⑥ 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？

➢143

四、解答4

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

④ 数据存储器平均用了多少时钟周期？

- 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

➢144

四、解答5

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

⑤ 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？

- 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%

四、解答6

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

⑥ 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？

- 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。

	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

五、题目

❖ 在制造硅芯片时，材料的缺陷和制造错误会导致电路失效。一个非常普遍的问题是一根线上的信号会对相邻线上的信号产生影响，这被称为串扰。有一类串扰问题是这样的，某些线上的信号为常值(如电源线)，该线附近的线也被固定为0(stuck-at-0)或1(stuck-at-1)。试根据下表的两种缺陷(信号来自图4)分别回答下列问题。

	有问题的信号
a.	指令存储器，输出信号第7位
b.	控制单元，输出信号MemtoReg

① 设这样测试处理器的缺陷：先给PC、寄存器堆、数据和指令存储器中设置一些值(可以自己选择)，执行一条指令，然后读出PC、寄存器堆和存储器中的值；最后检查这些值以判断处理器中是否存在缺陷。你能设计这样一个方案检查该信号上是否有固定为0缺陷吗？

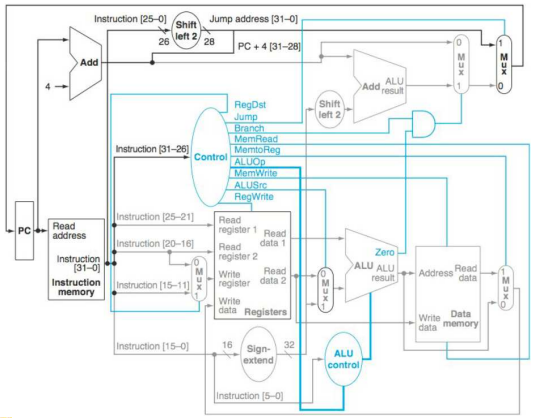
五、题目

② 条件同第①问，但是这次检查固定为1缺陷。你能只设计一个测试方案同时检查固定为0缺陷和固定为1缺陷吗？如果可以，请解释如何实现；如果不能，请说明理由。

③ 如果我们知道一个处理器在该信号上有一个固定为1缺陷，它还能用吗？为了使这个处理器仍然可用，我们必须将原来能在正常MIPS处理器上运行的程序做一些变换，使之可以在这个处理器上运行。假设指令存储器和数据存储器都很大，足够容纳变换后的程序。提示：将因为该缺陷不能用的指令替换为一系列能用的指令，这一系列指令与原指令功能相同。

五、题图

❖ 图4



➢149

五、解答1

	有问题的信号
a.	指令存储器，输出信号第7位
b.	控制单元，输出信号MemtoReg

① 假设这样测试处理器的缺陷：先给PC、寄存器堆、数据和指令存储器中设置一些值(可以自己选择)，执行一条指令，然后读出PC、寄存器堆和存储器中的值；最后检查这些值以判断处理器中是否存在缺陷。你能设计这样一个方案检查该信号上是否有固定为0缺陷吗？

- a.为了测试是否有固定为0缺陷，我们需要一个指令，该指令可以将待测信号置为1，并且与0值有着不同的输出，指令存储器的第7位输出只用于指令的立即数或者偏移量部分，因而可以采用指令ADDI $1,$0,128，该指令可以将128放置到寄存器$1中，如果指令存储器的第7位输出有固定为0缺陷，那么寄存器$1中的值就会为0而不是128。
- b.能够将输出信号MemtoReg置为1的只有load类指令，我们可以将数据存储器中的每个字都置为0，然后执行LW $1,1024($0)，如果寄存器$1中的值不是0而是1024，说明输出信号MemtoReg有固定为0缺陷。

➢150

五、解答2

	有问题的信号
a.	指令存储器，输出信号第7位
b.	控制单元，输出信号MemtoReg

② 条件同第①问，但是这次检查固定为1缺陷。你能只设计一个测试方案同时检查固定为0缺陷和固定为1缺陷吗？如果可以，请解释如何实现；如果不能，请说明理由。

- 检查固定为0缺陷需要能够将该信号置为1的指令，而检查固定为1缺陷需要能够将该信号置为0的指令，由于在一个周期中一个信号不可能既为0又为1，因而不能同时检查固定为0缺陷和固定为1缺陷。若要检查固定为1缺陷，与固定为0缺陷类似，可采用如下方法：
- a.执行指令ADDI $1,$0,0，如果指令存储器的第7位输出有固定为1缺陷，那么寄存器$1中的值就会为128而不是0。
- b.这个信号的固定为1缺陷不能准确检查出来，因为所有能够将MemtoReg信号设置为0的指令都会同时将MemRead信号设置为0，这样会导致写入到寄存器$1中的数据是不确定的(与我们在检测固定为0缺陷时刻意放置的数据没有关系)，有可能最后出现在寄存器$1中的数据是与原来寄存器中的数据相同的，从而检测不出来固定为1缺陷。

➢151

五、解答3

	有问题的信号
a.	指令存储器，输出信号第7位
b.	控制单元，输出信号MemtoReg

③ 如果我们知道一个处理器在该信号上有一个固定为1缺陷，它还能用吗？为了使这个处理器仍然可用，我们必须将原来能在正常MIPS处理器上运行的程序做一些变换，使之可以在这个处理器上运行。假设指令存储器和数据存储器都很大，足够容纳变换后的程序。提示：将因为该缺陷不能用的指令替换为一系列能用的指令，这一系列指令与原指令功能相同。

- a.要避开指令存储器输出信号的固定为1缺陷是有可能的，但是比较麻烦，我们必须找到在输出信号第7位上为0的所有指令(一般是I型指令)，然后将这些指令中的立即数进行相应的替代。例如，对于一个第7位为0的LW $1,0($0)指令，需要用LI $1,128、SUB $1,$0,$1和LW $1,128($1)代替。
- b. MemtoReg信号的固定为1缺陷是不能回避的，MemtoReg信号的固定为1缺陷，会阻止除load类指令外的所有指令向寄存器堆中写入数据，而load类指令只能将数据从存储器载入到寄存器中，不能模仿ALU的运算操作。

➢152

五、题目

❖ 根据下表的缺陷分别回答下列问题。

	缺陷
a.	固定为1
b.	如果指令的第31~26位全为0，则固定为0，否则无缺陷

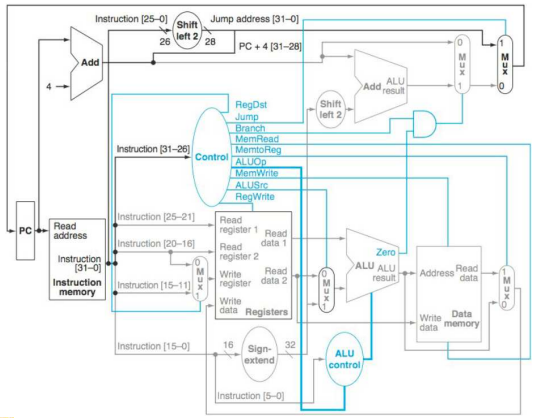
④ 条件同第①问，这次检测控制信号MemRead是否存在上表中的缺陷？

⑤ 条件同第①问，这次检测控制信号Jump是否存在上表中的缺陷？

⑥ 使用第①问中描述的测试方案，可以一次对几个不同的信号进行测试，但一般来说不可能同时测试到所有信号。试着设计一系列方案对所有多选器输出的上表中的缺陷进行测试(五个多选器输出的每一位都要测试到)。尽量使用较少的测试方案。

五、题图

❖图4



五、解答4

	缺陷
a.	固定为1
b.	如果指令的第31~26位全为0，则固定为0，否则无缺陷

④ 条件同第①问，这次检测控制信号MemRead是否存在上表中的缺陷？

- a.如果MemRead存在固定为1缺陷，那么在每一条指令执行的时候都会读取数据存储器，然而对于非load类指令，从存储器读出的数据会被多选器抛弃，这导致我们无法设计出检测该缺陷的方法，因为即使存在该缺陷，处理器也是正常工作的。
- b.为了测试该缺陷我们需要一个操作码为0同时MemRead信号为1的指令，但是操作码为0的指令都是ALU的R型指令，不是load指令，因而它们的MemRead信号都为0，因此，即使存在该缺陷，处理器也是正常工作的，因而我们无法设计出检测该缺陷的方法。

五、解答5

	缺陷
a.	固定为1
b.	如果指令的第31~26位全为0，则固定为0，否则无缺陷

⑤ 条件同第①问，这次检测控制信号Jump是否存在上表中的缺陷？

- a.如果Jump信号存在固定为1缺陷，那么每一条指令执行时都会按照执行J指令时的方法更新PC的值，要检测该缺陷，可以将一条非跳转指令(例如ADD $1,$0,$0)放在指令存储器中第一条指令的位置，该指令执行之后PC的值应当是0x00000004，若PC的值变为0x00002080则说明Jump信号存在固定为1缺陷。
- b.为了测试该缺陷我们需要一个操作码为0同时Jump信号为1的指令，然而跳转指令的操作码都不是0，因此我们无法设计出检测该缺陷的方法，此时处理器是正常工作的。

五、解答6

	缺陷
a.	固定为1
b.	如果指令的第31~26位全为0，则固定为0，否则无缺陷

⑥ 使用第①问中描述的测试方案，可以一次对几个不同的信号进行测试，但一般来说不可能同时测试到所有信号。试着设计一系列方案对所有多选器输出的上表中的缺陷进行测试(五个多选器输出的每一位都要测试到)。尽量使用较少的测试方案。

- 涉及到5个控制信号：RegDst、Jump、Branch、MemtoReg和ALUSrc，分别对每一个信号设计测试方案进行测试即可，但是注意有些信号的缺陷a或者缺陷b是无法检测出来的。
- a.RegDst信号的固定为1缺陷可用lw指令测试，首先可以将数据存储器全写为1，执行lw $1,0x1000($0)，如果载入的字出现在了$2中而不是$1中说明RegDst存在固定为1缺陷。
- Jump的固定为1缺陷测试方法同(5)a。

五、解答6

	缺陷
a.	固定为1
b.	如果指令的第31~26位全为0，则固定为0，否则无缺陷

⑥ 使用第①问中描述的测试方案，可以一次对几个不同的信号进行测试，但一般来说不可能同时测试到所有信号。试着设计一系列方案对所有多选器输出的上表中的缺陷进行测试(五个多选器输出的每一位都要测试到)。尽量使用较少的测试方案。

- Branch的固定为1缺陷可以用非分支指令测试，也可以用测试Jump时同样的方法，ADD $1,$0,$0指令会使ALU的Zero输出为1，导致分支条件“满足”，若执行之后PC的值变为0x00002084而不是0x00000004则说明Branch存在固定为1缺陷。
- MemtoReg信号的固定为1缺陷无法准确检测，原因参见(2)b。
- ALUSrc的固定为1缺陷也可以用ADD $1,$0,$0指令测试，若执行之后$1的值为0x00000820而不是0，则说明存在缺陷。

五、解答6

	缺陷
a.	固定为1
b.	如果指令的第31~26位全为0，则固定为0，否则无缺陷

⑥ 使用第①问中描述的测试方案，可以一次对几个不同的信号进行测试，但一般来说不可能同时测试到所有信号。试着设计一系列方案对所有多选器输出的上表中的缺陷进行测试(五个多选器输出的每一位都要测试到)。尽量使用较少的测试方案。

- 涉及到5个控制信号：RegDst、Jump、Branch、MemtoReg和ALUSrc，分别对每一个信号设计测试方案进行测试即可，但是注意有些信号的缺陷a或者缺陷b是无法检测出来的。
- b.操作码为0的指令都是ALU的R型指令，对于这些指令Jump、Branch、MemtoReg以及ALUSrc本就应该为0，所以这些信号的缺陷无法检测。
- RegDst的缺陷可以用ADD $1,$2,$3来测试，若执行完之后$2+$3的结果出现在了$3而不是$1中，说明存在缺陷。

六、题目

❖ 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。

	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

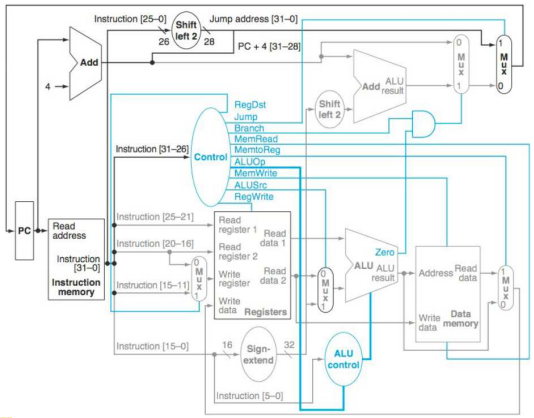
① 为了避免增加图4中数据通路的关键路径长度，留给控制单元产生MemWrite信号的时间有多少？

② 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？

③ 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？

六、题图

❖图4



➢161

六、解答1

	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

① 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？

- 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)。

➢162

六、解答2

	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

② 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？

- 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。

➢163

六、解答3

	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

③ 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？

	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。

➢164

六、题目

❖ 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

④ 处理器的时钟周期为多少？

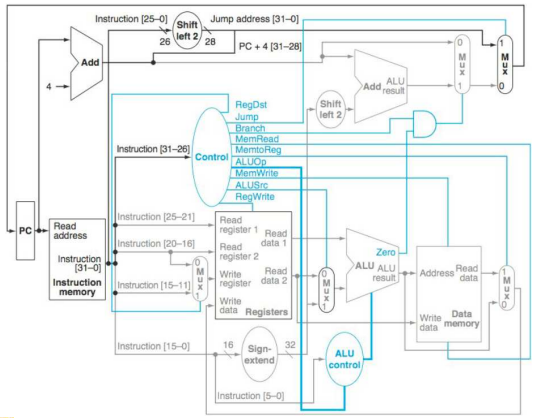
⑤ 如果你可以加速控制信号的产生，但加快一个控制信号5ps的代价是处理器成本增加1元。那么为了最大化性能你会加速哪些控制信号？这种性能改进的最小代价是多少？

⑥ 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⑤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？

➢165

六、题图

❖图4



➢166

六、解答4

	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

	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

④ 处理器的时钟周期为多少？

- 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

	对时钟周期影响最大的控制信号	理想的时钟周期(由第(1)问得来)	实际的时钟周期
a.	MemWrite (+130ps)	1330ps	1460ps
b.	MemWrite (+680ps)	2320ps	3000ps

➢167

六、解答5

	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

⑤ 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？

	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。

➢168

六、解答6

	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

⑥ 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⑤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？

	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

计算机组成习题参考答案

--流水线处理器设计

1、题目

❖ 本习题讨论流水线对处理器时钟周期的影响。表中给出了数据通路中不同阶段延迟的两种情况，试根据这分别回答下列问题。

	IF	ID	EX	MEM	WB
a.	300ps	400ps	350ps	500ps	100ps
b.	200ps	150ps	120ps	190ps	140ps

1) 流水线处理器与非流水线处理器的时钟周期分别是什么？

2) lw 指令在流水线处理器和非流水线处理器中的总延迟分别是多少？

3) 如果可以将原流水线数据通路的一级划分为两，每级的延迟是原级的一半，会选择哪一级进行划分？划分后处理器的时钟周期为多少？

1、解答1

❖ 本习题讨论流水线对处理器时钟周期的影响。表中给出了数据通路中不同阶段延迟的两种情况，试根据这分别回答下列问题。

	IF	ID	EX	MEM	WB
a.	300ps	400ps	350ps	500ps	100ps
b.	200ps	150ps	120ps	190ps	140ps

1) 流水线处理器与非流水线处理器的时钟周期分别是什么？

- a.流水线处理器：500ps（MEM），非流水线处理器：300 + 400 + 350 + 500 + 100 = 1650ps
- b.流水线处理器：200ps（IF），非流水线处理器：200 + 150 + 120 + 190 + 140 = 800ps
- 解析：流水线处理器的时钟周期受限于处理速度最慢的一级，非流水线处理器的时钟周期受限于处理时间最长的指令（一般为lw指令）。

1、解答2

❖ 本习题讨论流水线对处理器时钟周期的影响。表中给出了数据通路中不同阶段延迟的两种情况，试根据这分别回答下列问题。

	IF	ID	EX	MEM	WB
a.	300ps	400ps	350ps	500ps	100ps
b.	200ps	150ps	120ps	190ps	140ps

2) lw 指令在流水线处理器和非流水线处理器中的总延迟分别是多少？

- a.流水线处理器：500 × 5 = 2500ps，非流水线处理器：300 + 400 + 350 + 500 + 100 = 1650ps
- b.流水线处理器：200 × 5 = 1000ps，非流水线处理器：200 + 150 + 120 + 190 + 140 = 800ps
- 解析：lw指令需要完整执行IF、ID、EX、MEM、WB五级。

1、解答3

❖ 本习题讨论流水线对处理器时钟周期的影响。表中给出了数据通路中不同阶段延迟的两种情况，试根据这分别回答下列问题。

	IF	ID	EX	MEM	WB
a.	300ps	400ps	350ps	500ps	100ps
b.	200ps	150ps	120ps	190ps	140ps

3) 如果可以将原流水线数据通路的一级划分为两，每级的延迟是原级的一半，会选择哪一级进行划分？划分后处理器的时钟周期为多少？

- a.选择MEM级，划分后得到MEM1和MEM2级的延迟分别为250ps，处理器 的时钟周期为400ps（ID）
- b.选择IF级，划分后得到IF1和IF2级的延迟分别为100ps，处理器的时钟周期 为190ps（MEM）
- 解析：应当选择原来处理速度最慢的一级进行划分，得到新的具有六级流水 线的处理器，此时处理器的时钟周期仍然受限于处理速度最慢的一级。

1、题目

❖ 假设处理器执行的指令比例下表两种情况所示，试根据每种情况分别回答下列问题。

	ALU	beq	lw	sw
a.	50%	25%	15%	10%
b.	30%	25%	30%	15%

4)假设没有阻塞和冒险，数据存储器的利用率是多少（占总周期数的百分比）？

5)假设没有阻塞和冒险，寄存器堆的写寄存器端口的利用率是多少？

1、解答4

❖ 假设处理器执行的指令比例下表两种情况所示，试根据每种情况分别回答下列问题。

	ALU	beq	lw	sw
a.	50%	25%	15%	10%
b.	30%	25%	30%	15%

4)假设没有阻塞和冒险，数据存储器的利用率是多少（占总周期数的百分比）？

- a. 15% + 10% = 25%
- b. 30% + 15% = 45%
- 解析：需要用到数据存储器的指令为lw和sw。

1、解答5

- 假设处理器执行的指令比例下表两种情况所示，试根据每种情况分别回答下列问题。

	ALU	beq	lw	sw
a.	50%	25%	15%	10%
b.	30%	25%	30%	15%

5)假设没有阻塞和冒险，寄存器堆的写寄存器端口的利用率是多少？

- a. 50% + 15% = 65%
- b. 30% + 30% = 60%
- 解析：需要用到寄存器堆的写寄存器端口的指令为ALU和lw。

2、题目

- 本习题讨论数据相关如何影响基本五级流水线的运行。试根据下表的两种指令序列情况分别回答下列问题。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $6, $2, $2
	sw $6, 50($1)
		b.	lw $5, -16($5)
			sw $5, -16($5)
			add $5, $5, $5

1)指出指令序列中存在的相关及其类型。

2)假设该流水线处理器没有转发，指出指令序列中存在的冒险并加入nop指令以消除冒险。

3)假设该流水线处理器中有充分的转发。指出指令序列中存在的冒险并加入nop指令以消除冒险。

2、解答1

- 本习题讨论数据相关如何影响基本五级流水线的运行。试根据下表的两种指令序列情况分别回答下列问题。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $6, $2, $2
	sw $6, 50($1)
		b.	lw $5, -16($5)
			sw $5, -16($5)
			add $5, $5, $5

1)指出指令序列中存在的相关及其类型。

- a. I1和I3：RAW $1，I1和I2：WAR $6，I2和I3：RAW $6
- b. I1和I2、I3：RAW $5，I1、I2和I3：WAR $5，I1和I3：WAW $5
- 解析：三种数据相关类型为：RAW（Read After Write，写后读），WAR（Write After Read，读后写）和WAW（Write After Write，写后写）。
- a中存在的读写关系：I1写$1，I3读$1；I1读$6，I2写$6；I2写$6，I3读$6。
- b中存在的读写关系：I1写$5，I2读$5，I3读$5；I1读$5，I2读$5，I3写$5；I1写$5，I3写$5。

2、解答2

- 本习题讨论数据相关如何影响基本五级流水线的运行。试根据下表的两种指令序列情况分别回答下列问题。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $6, $2, $2
	sw $6, 50($1)
		b.	lw $5, -16($5)
			sw $5, -16($5)
			add $5, $5, $5

2)假设该流水线处理器没有转发，指出指令序列中存在的冒险并加入nop指令以消除冒险。

- 解析：WAR和WAW不会引起冒险，只有RAW会引起冒险，因而加入nop指令时只需要考虑RAW。lw指令写RF是在WB，sw指令读RF是在ID，add指令读RF是在ID，写RF是在WB，因此至少需要将sw相对于lw推迟两级，将sw相对于add推迟两级，加入nop后的指令序列如下表。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $6, $2, $2
	nop
	nop
	sw $6, 50($1)
		b.	lw $5, -16($5)
			nop
			nop
			sw $5, -16($5)
			add $5, $5, $5

2、解答3

- 本习题讨论数据相关如何影响基本五级流水线的运行。试根据下表的两种指令序列情况分别回答下列问题。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $6, $2, $2
	sw $6, 50($1)
		b.	lw $5, -16($5)
			sw $5, -16($5)
			add $5, $5, $5

3)假设该流水线处理器中有充分的转发。指出指令序列中存在的冒险并加入nop指令以消除冒险。

- 解析：具有了转发之后，ALU可以将计算结果转发至下一条指令的EX，lw可以将MEM结果转发至再下一条指令的EX，这样一来，只需要将sw相对于lw推迟一级即可，而sw通过转发机制可以直接跟在add之后，加入nop后的指令序列如下表。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $6, $2, $2
	sw $6, 50($1)
		b.	lw $5, -16($5)
			nop
			sw $5, -16($5)
			add $5, $5, $5

2、题目

- 根据下表的两种时钟周期情况，分别回答下列问题。

	无转发	充分的转发	仅ALU至ALU的转发
a.	300ps	400ps	360ps
b.	200ps	250ps	220ps

4)该指令序列在无转发和充分的转发时总执行时间分别是多少？后者相对于前者的加速比是多少。

5)如果仅有ALU至ALU的转发（没有从MEM到EX的转发），如何加入nop指令以消除可能的冒险？

6)该指令序列在仅有ALU至ALU的转发时总执行时间分别是多少？与无转发的情况相比，加速比是多少？

2、解答4

- 根据下表的两种时钟周期情况，分别回答下列问题。

	无转发	充分的转发	仅ALU至ALU的转发
a.	300ps	400ps	360ps
b.	200ps	250ps	220ps

4)该指令序列在无转发和充分的转发时总执行时间分别是多少？后者相对于前者的加速比是多少。

- a.无转发：(7＋2)×300＝2700ps，充分的转发：7×400ps＝2800ps，加速比：2700/2800＝0.96
- b.无转发：(7＋2)×200＝1800ps，充分的转发：(7＋1)×250＝2000ps，加速比：1800/2000＝0.90
- 解析：无转发时，a中的指令序列需要插入2个nop，b中的指令序列需要插入2个nop（详见2)问）；有充分的转发时，a中的指令序列不需要再插入nop，b中的指令序列需要插入1个nop（详见3)问）。

原始指令序列			无转发			充分的转发		

a.	lw $1, 40($6)
	add $6, $2, $2
	sw $6, 50($1)
		b.	a.	lw $1, 40($6)
				add $6, $2, $2
				nop
				nop
				sw $6, 50($1)
					b.	lw $5, -16($5)
						nop
						nop
						sw $5, -16($5)
						add $5, $5, $5
							a.	lw $1, 40($6)
								add $6, $2, $2
								sw $6, 50($1)
									b.	lw $5, -16($5)
										nop
										sw $5, -16($5)
										add $5, $5, $5

2、解答5

- 根据下表的两种时钟周期情况，分别回答下列问题。

	无转发	充分的转发	仅ALU至ALU的转发
a.	300ps	400ps	360ps
b.	200ps	250ps	220ps

5)如果仅有ALU至ALU的转发（没有从MEM到EX的转发），如何加入nop指令以消除可能的冒险？

- 解析：若只有ALU至ALU的转发，一条ALU指令可以向下一条指令的EX转发ALU计算结果，但不能向再下一条指令转发，因此实际上此时无法转发lw中MEM的数据，sw相对于lw仍然至少需要推迟两级，但是sw此时通过ALU至ALU的转发机制可以直接跟在add之后，加入nop后的指令序列如下表。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $6, $2, $2
	nop
	sw $6, 50($1)
		b.	lw $5, -16($5)
			nop
			nop
			sw $5, -16($5)
			add $5, $5, $5

2、解答6

❖ 根据下表的两种时钟周期情况，分别回答下列问题。

	无转发	充分的转发	仅ALU至ALU的转发
a.	300ps	400ps	360ps
b.	200ps	250ps	220ps

6)该指令序列在仅有ALU至ALU的转发时总执行时间分别是多少？与无转发的情况相比，加速比是多少？

- a.无转发：(7 + 2) ×300 = 2700ps，仅有ALU至ALU的转发：(7 + 1) ×360ps = 2880ps，加速比：2700/2880 = 0.94
- b.无转发：(7 + 2) ×200 = 1800ps，仅有ALU至ALU的转发：(7 + 2) ×220 = 1980ps，加速比：1800/1980 = 0.91
- 解析：无转发时，a中的指令序列需要插入2个nop，b中的指令序列需要插入2个nop（详见2)问）；仅有ALU至ALU的转发时，a中的指令序列需要插入1个nop，b中的指令序列需要插入2个nop（详见5)问）。

原始指令序列		无转发		仅有ALU至ALU的转发	

a.	lw $1, 40($6)
	add $6, $2, $2
	sw $6, 50($1)
		a.	lw $1, 40($6)
			add $6, $2, $2
			nop
			nop
			sw $6, 50($1)
				a.	lw $1, 40($6)
					add $6, $2, $2
					nop
					sw $6, 50($1)

b.	lw $5, -16($5)
	sw $5, -16($5)
	add $5, $5, $5
		b.	lw $5, -16($5)
			nop
			nop
			sw $5, -16($5)
			add $5, $5, $5
				b.	lw $5, -16($5)
					nop
					nop
					sw $5, -16($5)
					add $5, $5, $5

3、题目

❖ 本习题讨论指令集对流水线设计的影响。试根据下表的两条新指令回答下列问题。

a.	bezi (Rs), Label	if Mem[Rs] = 0 then PC = PC + Offs
b.	swi Rd, Rs(Rt)	Mem[Rs+Rt] = Rd

1)为了将这条新指令增加到MIPS指令集，必须对流水线数据通路做什么改动？

2)需要在第1)问的数据通路上增加哪些控制信号？

3)对新指令的支持是否会引入新的冒险？已有冒险导致的阻塞是否会更加严重？

3、解答1

❖ 本习题讨论指令集对流水线设计的影响。试根据下表的两条新指令回答下列问题。

a.	bezi (Rs), Label	if Mem[Rs] = 0 then PC = PC + Offs
b.	swi Rd, Rs(Rt)	Mem[Rs+Rt] = Rd

1)为了将这条新指令增加到MIPS指令集，必须对流水线数据通路做什么改动？

- a.需要在ALU前的MUX增加一个输入端0；需要在WB中增加一个比较数据存储器值与0的比较器（或者专门为比较这个过程增加新的一级流水，但是不建议这样做）；需要在PC前的MUX控制信号中加入比较器的结果，并且此时MUX的控制信号必须一直阻塞到WB之后才能进行选择。
- b.需要为寄存器堆加入一个新的读寄存器地址写入端和读出端；需要在EX中ALU之后加入一个MUX来选择要写入数据存储器的值是Rd（对于swi指令）还是Rt（对于sw指令）。

3、解答2

❖ 本习题讨论指令集对流水线设计的影响。试根据下表的两条新指令回答下列问题。

a.	bezi (Rs), Label	if Mem[Rs] = 0 then PC = PC + Offs
b.	swi Rd, Rs(Rt)	Mem[Rs+Rt] = Rd

2)需要在第1)问的数据通路上增加哪些控制信号？

- a.需要为ALU前的MUX增加1位控制信号位，使其可以选择出新添加的输入端0；需要为PC前的MUX控制信号加入比较器的结果，来决定是否需要修改PC。
- b.需要为新加入的MUX增加控制信号。

3、解答3

❖ 本习题讨论指令集对流水线设计的影响。试根据下表的两条新指令回答下列问题。

a.	bezi (Rs), Label	if Mem[Rs] = 0 then PC = PC + Offs
b.	swi Rd, Rs(Rt)	Mem[Rs+Rt] = Rd

3)对新指令的支持是否会引入新的冒险？已有冒险导致的阻塞是否会更加严重？

- a.会。bezi指令会引入一个新的控制冒险，它的PC直到WB才能确定是否需要修改，而且无法通过缩短分支延迟的方法提前判断出分支是否执行，因而会使已有冒险导致的阻塞更加严重。
- b.不会。swi指令不改变任何寄存器的值，不会引起数据冒险，同时它也不是分支指令，不会引起控制冒险。

4、题目

❖ 本习题讨论转发、冒险检测和指令集设计之间的关系。分别根据下表的两个指令序列回答下列问题。假设其在一个五级流水线上执行。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $2, $3, $1
	add $1, $6, $4
	sw $2, 20($4)
	and $1, $1, $4
		b.	add $1, $5, $3
			sw $1, 0($2)
			lw $1, 4($2)
			add $5, $5, $1
			sw $1, 0($2)

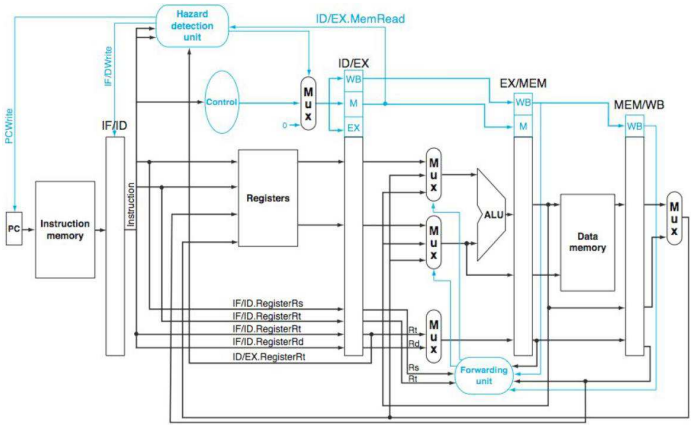
1)如果没有转发或冒险检测电路，请插入nop指令以保证正确执行。

2)重做第1)问，这次仅当通过改变或重排序指令也不能避免冒险时才插入nop指令。假设可以使用寄存器R7作为临时寄存器。

3)如果处理器中存在转发，但忘了实现冒险检测单元（以为实现了），代码执行时会发生什么情况？

4）如果没有转发，对图中的冒险检测单元来说还需要哪些新的输入输出信号？以该指令序列为例，说明为什么需要这些信号。

4、题图



4、解答1

❖ 本习题讨论转发、冒险检测和指令集设计之间的关系。分别根据下表的两个指令序列回答下列问题。假设其在一个五级流水线上执行。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $2, $3, $1
	add $1, $6, $4
	sw $2, 20($4)
	and $1, $1, $4
		b.	add $1, $5, $3
			sw $1, 0($2)
			lw $1, 4($2)
			add $5, $5, $1
			sw $1, 0($2)

1)如果没有转发或冒险检测电路，请插入nop指令以保证正确执行。

	指令序列		指令序列

a.	lw $1, 40($6)
	nop
	nop
	add $2, $3, $1
	add $1, $6, $4
	nop
	sw $2, 20($4)
	and $1, $1, $4
		b.	add $1, $5, $3
			nop
			nop
			sw $1, 0($2)
			lw $1, 4($2)
			nop
			nop
			add $5, $5, $1
			sw $1, 0($2)

4、解答2

❖ 本习题讨论转发、冒险检测和指令集设计之间的关系。分别根据下表的两个指令序列回答下列问题。假设其在一个五级流水线上执行。

	指令序列		指令序列
a.	I1：lw $1, 40($6) I2：add $2, $3, $1 I3：add $1, $6, $4 I4：sw $2, 20($4) I5：and $1, $1, $4	b.	I1：add $1, $5, $3 I2：sw $1, 0($2) I3：lw $1, 4($2) I4：add $5, $5, $1 I5：sw $1, 0($2)

2)重做第1)问，这次仅当通过改变或重排序指令也不能避免冒险时才插入nop指令。假设可以使用寄存器R7作为临时寄存器。

	指令序列		指令序列
a.	I1：lw $7, 40($6) I3：add $1, $6, $4 nop I2：add $2, $3, $7 I5：and $1, $1, $4 nop I4：sw $2, 20($4)	b.	I1：add $7, $5, $3 I3：lw $1, 4($2) nop I2：sw $7, 0($2) I4：add $5, $5, $1 I5：sw $1, 0($2)

➢193

4、解答3

❖ 本习题讨论转发、冒险检测和指令集设计之间的关系。分别根据下表的两个指令序列回答下列问题。假设其在一个五级流水线上执行。

	指令序列		指令序列
a.	I1：lw $1, 40($6) I2：add $2, $3, $1 I3：add $1, $6, $4 I4：sw $2, 20($4) I5：and $1, $1, $4	b.	I1：add $1, $5, $3 I2：sw $1, 0($2) I3：lw $1, 4($2) I4：add $5, $5, $1 I5：sw $1, 0($2)

3)如果处理器中存在转发，但忘了实现冒险检测单元（以为实现了），代码执行时会发生什么情况？

- a.I2获得的$1的值是I1的lw指令计算出的访存地址（$6+40），此时$1的值还没有被lw指令更新。
- b.I4获得的$1的值是I3的lw指令计算出的访存地址（$2+4），此时$1的值还没有被lw指令更新。
- 转发单元可以解决EX冒险和MEM冒险，但是无法解决load-use型数据冒险；冒险检测单元用来检测load-use型数据冒险，在load指令与紧随其后利用他的结果的指令间插入阻塞。

➢194

4、解答4

❖ 本习题讨论转发、冒险检测和指令集设计之间的关系。分别根据下表的两个指令序列回答下列问题。假设其在一个五级流水线上执行。

	指令序列		指令序列

a.	lw $1, 40($6)
	add $2, $3, $1
	add $1, $6, $4
	sw $2, 20($4)
	and $1, $1, $4
		b.	add $1, $5, $3
			sw $1, 0($2)
			lw $1, 4($2)
			add $5, $5, $1
			sw $1, 0($2)

4）如果没有转发，对图中的冒险检测单元来说还需要哪些新的输入输出信号？以该指令序列为例，说明为什么需要这些信号。

- 输入信号：冒险检测单元需要在EX中检查R型指令和lw指令的Rd寄存器，在MEM中检查目标寄存器号，因此需要添加ID/EX流水线寄存器的Rd和EX/MEM的输出寄存器作为输入信号，图中已经有ID/EX的Rt这个输入信号了，不需要再次添加。
- 输出信号：不需要额外的输出信号。

➢195

计算机组成习题参考答案

--Cache和虚拟存储

➢196

第一题

某计算机的存储系统由Cache和主存组成。若所访问的字在Cache中，则存取它需要10ns；将所访问的字从主存装入Cache需要60ns。假定Cache的命中率为0.9（缺失时，从主存装入cache，再从cache读取），计算该存储系统访问一个字的平均存取时间。

答：0.9*(10)+0.1*(10+60)=16ns

➢197

第二题

假设一4路组相联Cache，数据存储空间大小64KB，块大小为16字节，主存地址32位，主存一个字包含4个字节，Cache采用写回策略，每个数据块包括1位有效位，Cache每个字用 1 位脏位来表示是否被修改。

1) CPU 如何解释主存地址（主存地址格式）
2) 计算实现该Cache所需总存储容量

➢198

第二题

假设一4路组相联Cache，数据存储空间大小64KB，块大小为16字节，主存地址32位，主存一个字包含4个字节，Cache采用写回策略，每个数据块包括1位有效位，Cache每个字用 1 位脏位来表示是否被修改。

1) CPU 如何解释主存地址（主存地址格式）

18	10	4
组内块地址 (tag)	组地址	块内偏移

- 主存容量：2^{32}= 4G Bytes
- Cache容量：64K Bytes
- 块(Block)大小：16 Bytes
- Way：4 ways (Cache每组含4个Block)
- Cache组数：64KB/(16B×4)=2^{10}= 1024组
- 主存每组块数= 4G Bytes/(16Bytes×1024组) = 2^{18}块/组
- 主存地址：32位，高18位为组内块地址，中间10位为组地址，低4位为块内地址

➢199

第二题

假设一4路组相联Cache，数据存储空间大小64KB，块大小为16字节，主存地址32位，主存一个字包含4个字节，Cache采用写回策略，每个数据块包括1位有效位，Cache每个字用 1 位脏位来表示是否被修改。

1) CPU 如何解释主存地址（主存地址格式）
2) 计算实现该Cache所需总存储容量

- Cache的Tag：23位= 1位有效位+4位脏位(4个字)+18位组内块地址
- 每Cache行组成：23位Tag+128位数据(16 Bytes)
- 实现Cache的总存储容量：(23 + 128)×(64K/16) = 604K bits = 75.5K Bytes

18	10	4
组内块地址 (tag)	组地址	块内偏移

➢200

第三题

计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计.

1）计算上述取数过程的命中率；

2）计算采用Cache后的加速比。

➢201

第三题

计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计.

1）计算上述取数过程的命中率；

- 主存容量32K字，Cache容量4K字
- 块大小＝64字
- 组内Cache块数＝4块
- 组数＝4K/(64×4)＝16
- 主存块数＝32K/64＝512
- 主存块/组＝512/16＝32

➢202

第三题

计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计.

1）计算上述取数过程的命中率；

➢4352个主存字分配在68个主存块中（4352/64=68）

- 主存块0：主存字0~63
- 主存块1：主存字64~127
- ……
- 主存块63：主存字4032~4095
- 主存块64：主存字4096~4159
- 主存块65：主存字4160~4223
- 主存块66：主存字4224~4287
- 主存块67：主存字4288~4351

➢203

第三题

计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计.

1）计算上述取数过程的命中率；

- 主存块⇔Cache组
- 68个主存块
- 16个Cache组
- 每Cache组4块

Cache组	主存块
0	0，16，32，48，64
1	1，17，33，49，65
2	2，18，34，50，66
3	3，19，35，51，67
4	4，20，36，52
…	…
13	13，29，45，61
14	14，30，46，62
15	15，31，47，63

➢204

第三题

初始状态

组	块0	块1	块2	块3
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				

➢205

第三题

第一次循环：块0~块63

组	块0	块1	块2	块3
0	0	16	32	48
1	1	17	33	49
2	2	18	34	50
3	3	19	35	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

块64~67?

➢206

第三题

第一次循环：块64~块67

组	块0	块1	块2	块3
0	0➔64	16	32	48
1	1➔65	17	33	49
2	2➔66	18	34	50
3	3➔67	19	35	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

块0~3?

➢207

第三题

第二次循环：块0~块3

组	块0	块1	块2	块3
0	64	16➔0	32	48
1	65	17➔1	33	49
2	66	18➔2	34	50
3	67	19➔3	35	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢208

第三题

第二次循环：块4~块15

组	块0	块1	块2	块3
0	64	16➔0	32	48
1	65	17➔1	33	49
2	66	18➔2	34	50
3	67	19➔3	35	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢209

第三题

第二次循环：块16~块19

组	块0	块1	块2	块3
0	64	16➔0	32➔16	48
1	65	17➔1	33➔17	49
2	66	18➔2	34➔18	50
3	67	19➔3	35➔19	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢210

第三题

第二次循环：块20~块31

组	块0	块1	块2	块3
0	64	16➔0	32➔16	48
1	65	17➔1	33➔17	49
2	66	18➔2	34➔18	50
3	67	19➔3	35➔19	51
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢211

第三题

第二次循环：块32~块35

组	块0	块1	块2	块3
0	64	16➔0	32➔16	48➔32
1	65	17➔1	33➔17	49➔33
2	66	18➔2	34➔18	50➔34
3	67	19➔3	35➔19	51➔35
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢212

第三题

第二次循环：块36~块47

组	块0	块1	块2	块3
0	64	16→0	32→16	48→32
1	65	17→1	33→17	49→33
2	66	18→2	34→18	50→34
3	67	19→3	35→19	51→35
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢213

第三题

第二次循环：块48~块51

组	块0	块1	块2	块3
0	64→48	16→0	32→16	48→32
1	65→49	17→1	33→17	49→33
2	66→50	18→2	34→18	50→34
3	67→51	19→3	35→19	51→35
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢214

第三题

第二次循环：块52~块63

组	块0	块1	块2	块3
0	64→48	16→0	32→16	48→32
1	65→49	17→1	33→17	49→33
2	66→50	18→2	34→18	50→34
3	67→51	19→3	35→19	51→35
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢215

第三题

第二次循环：块64~块67

组	块0	块1	块2	块3
0	64→48	16→0→64	32→16	48→32
1	65→49	17→1→65	33→17	49→33
2	66→50	18→2→66	34→18	50→34
3	67→51	19→3→67	35→19	51→35
4	4	20	36	52
5	5	21	37	53
6	6	22	38	54
7	7	23	39	55
8	8	24	40	56
9	9	25	41	57
10	10	26	42	58
11	11	27	43	59
12	12	28	44	60
13	13	29	45	61
14	14	30	46	62
15	15	31	47	63

➢216

第三题

计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计.

1）计算上述取数过程的命中率；

规律：

- 第1次循环(单位：块)
  - 主存块0~63：未命中
  - 主存块64~67：未命中，替换
  - 未命中次数：68
- 第2次循环~第10次循环(单位：块)
  - 映射到组0~3的20个主存块：未命中，替换
  - 主存块：0~3，16~19，32~35，48~51，64~67
  - 其余48个主存块：全部命中
  - 未命中次数：20×9
  - 命中次数：48×9

第三题

计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计.

1）计算上述取数过程的命中率；

块未命中vs. 块命中(单位：块)

- 未命中次数：68＋20×9＝248
- 命中次数：48×9＝432

若块未命中

- 1次字未命中，63次字命中

若块命中

- 64次字命中

CPU读存储器总计(单位：字)

- Cache未命中次数：1×248＝248
- Cache命中次数：63×248＋64×432＝43272

命中率：

- 43272/(248+43272)=99.43%

第三题

计算机系统包含32K字的主存，Cache容量4K字，每组4 Blocks，每Block 64个字。假设Cache开始是空的，CPU顺序从存储单元0，1，2到4351中读取字，然后再重复这样的取数9次，Cache速度是主存速度的10倍，采用LRU替换算法，假定块替换的时间忽略不计.

1）计算上述取数过程的命中率；

2）计算采用Cache后的加速比。

解1： $T_m=10T_C$

$$S_p=\frac{T_m}{T}=\frac{T_m}{HT_C+(1-H)T_m}=9.51$$

解2： $T_m=10T_C$

- 未命中时访问(字)时间为$10T_c+T_c=11T_c$
- 加速比=全部主存访问时间/(Cache未命中访问时间+Cache命中访问时间)
  $=(10T_c\times(248+43272))/(11T_c\times248+T_c\times43272)=9.46$

$T_i = h_i{\cdot}t_i + m_i{\cdot}(t_i + T_{i+1}) = t_i + m_i \cdot T_{i+1}$, $T=T_C+(1-H)\cdot T_m$

$S_p= T_m/T_C+(1-H)\cdot T_m=9.46$

第四题

考虑一个Cache，其存取时间为2.5ns，行大小为64字节，命中率H=0.95。主存使用块传送方式，第一个字（4字节）存取时间为50ns，其后每个字存取时间为5ns。

1）出现一次Cache缺失的存取时间是多少？假设此时Cache等待，直到该行从主存传送到Cache，然后再从Cache读取；

2）假设行大小增大到128字节，命中率提升到0.97，是否会降低平均存取时间。

第四题

考虑一个Cache，其存取时间为2.5ns，行大小为64字节，命中率H=0.95。主存使用块传送方式，第一个字（4字节）存取时间为50ns，其后每个字存取时间为5ns。

1）出现一次Cache缺失的存取时间是多少？假设此时Cache等待，直到该行从主存传送到Cache，然后再从Cache读取；

行大小为64字节=16字

出现一次Cache缺失的存取时间为：50ns+15×5ns+2.5ns = 127.5ns

第四题

考虑一个Cache，其存取时间为2.5ns，行大小为64字节，命中率H=0.95。主存使用块传送方式，第一个字（4字节）存取时间为50ns，其后每个字存取时间为5ns。

1）出现一次Cache缺失的存取时间是多少？假设此时Cache等待，直到该行从主存传送到Cache，然后再从Cache读取；

2）假设行大小增大到128字节，命中率提升到0.97，是否会降低平均存取时间。

原条件下，平均存取时间为$T = H \times T_c+(1-H) \times T_m = 0.95 \times 2.5ns+0.05 \times 127.5ns = 8.75ns$（这里的$T_m$不是简单的内存访问时间，而是cache未命中时存储系统的访问时间）

行大小增加到128字节=32字后，出现一次Cache缺失的存取时间为50ns+31×5ns+2.5ns = 207.5ns，平均存取时间为$T = 0.97 \times 2.5ns+0.03 \times 207.5ns = 8.65ns$

可见平均存取时间降低了

块大小与缺失率的关系：

#一般而言，增加块大小将降低缺失率(因为空间局部性)，但块大小达到一定程度时，缺失率会随块大小的继续增加而上升(因为块数量下降带来块替换的增加)；

#单纯增加块大小带来缺失代价(缺失损失)的增大。

第五题

给定一个32位的虚拟地址空间和一个24位的物理地址，对于下面不同的分页大小P，请确定虚拟页号（VPN）、虚拟页内偏移量（VPO）、物理页号（PPN）和物理页内偏移量（PPO）的位数。

P	#VPN位数	#VPO位数	#PPN位数	#PPO位数
1KB				
2KB				
4KB				
8KB				

答：

P	#VPN位数	#VPO位数	#PPN位数	#PPO位数
1KB	22	10	14	10
2KB	21	11	13	11
4KB	20	12	12	12
8KB	19	13	11	13

第六题

假定一个计算机系统中有一个TLB和一个L1 Data Cache。该系统按字节编址，虚拟地址16位，物理地址12位；页大小为128字节，TLB采用4路组相联映射，共有16个页表项；L1 Data Cache采用直接映射方式，块大小为4字节，共16行。在系统运行到某一时刻，TLB、页表和L1 Data Cache中的部分内容（用十六进制表示）如下图所示。

组号	标记	实页号	有效位	标记	实页号	有效位	标记	实页号	有效位	标记	实页号	有效位
0	03	—	0	09	1D	1	00	—	0	07	10	1
1	13	2D	1	02	—	0	04	—	0	0A	—	0
2	02	—	0	08	—	0	06	—	0	03	—	0
3	07	—	0	63	12	1	0A	34	1	72	—	0

(a) TLB 内容(4 路组相联，4 组，16 个页表项)

请回答下列问题：

（1）虚拟地址中哪几位表示虚拟页号、哪几位表示页内偏移量？虚拟页号中哪几位表示TLB标记？哪几位表示TLB组索引？

（2）物理地址中哪几位表示物理页号、哪几位表示页内偏移量？在访问Cache时，物理地址如何划分成标记字段、行索引字段和块内地址字段？

（3）CPU 从地址067AH中取出的值是多少？要求对CPU读取地址067AH中内容的过程进行详细说明。

第六题

虚页号	实页号	有效位
000	08	1
001	03	1
002	14	1
003	02	1
004	—	0
005	16	1
006	—	0
007	07	1
008	13	1
009	17	1
00A	09	1
00B	—	0
00C	19	1
00D	—	0
00E	11	1
00F	0D	1

（b）部分页表内容（前 16 项）

行索引	标记	有效位	字节3	字节 2	字节 1	字节 0
0	19	1	12	56	C9	AC
1	—	0	—	—	—	—
2	1B	1	03	45	12	CD
3	—	0	—	—	—	—
4	32	1	23	34	C2	2A
5	0D	1	46	67	23	3D
6	—	0	—	—	—	—
7	10	1	12	54	65	DC
8	24	1	23	62	12	3A
9	—	0	—	—	—	—
A	2D	1	43	62	23	C3
B	—	0	—	—	—	—
C	12	1	76	83	21	35
D	16	1	A3	F4	23	11
E	33	1	2D	4A	45	55
F	—	0	—	—	—	—

（C） L1 Data Cache 内容（直接映射，16 行，块大小 4 字节）

请回答下列问题：

（1）虚拟地址中哪几位表示虚拟页号、哪几位表示页内偏移量？虚拟页号中哪几位表示TLB标记？哪几位表示TLB组索引？

（2）物理地址中哪几位表示物理页号、哪几位表示页内偏移量？在访问Cache时，物理地址如何划分成标记字段、行索引字段和块内地址字段？

（3） CPU 从地址067AH中取出的值是多少？要求对CPU读取地址067AH中内容的过程进行详细说明。

第六题

请回答下列问题：

（1）虚拟地址中哪几位表示虚拟页号、哪几位表示页内偏移量？虚拟页号中哪几位表示TLB标记？哪几位表示TLB组索引？

TLB分为4组，所以TLB组索引为2位

16位虚拟地址中低7位为页内偏移量，高9位为虚页号；虚页号中高7位为TLB标记，低2位为TLB组索引

7	2	7
TLB标记	TLB组索引	页内偏移
虚页号		

第六题

请回答下列问题：

（1）虚拟地址中哪几位表示虚拟页号、哪几位表示页内偏移量？虚拟页号中哪几位表示TLB标记？哪几位表示TLB组索引？

（2）物理地址中哪几位表示物理页号、哪几位表示页内偏移量？在访问Cache时，物理地址如何划分成标记字段、行索引字段和块内地址字段？

Cache有16行，所以Cache行索引为4位

12位物理地址中低7位为页内偏移量，高5位为物理页号。12位物理(主存)地址中，低2位为块内地址，中间4位为Cache行索引，高6位为标记

5	7	
物理页号	页内偏移	
6	4	2
Cache标记	Cache行索引	块内地址

第六题

请回答下列问题：

（1）虚拟地址中哪几位表示虚拟页号、哪几位表示页内偏移量？虚拟页号中哪几位表示TLB标记？哪几位表示TLB组索引？

（2）物理地址中哪几位表示物理页号、哪几位表示页内偏移量？在访问Cache时，物理地址如何划分成标记字段、行索引字段和块内地址字段？

（3） CPU 从地址067AH中取出的值是多少？要求对CPU读取地址067AH中内容的过程进行详细说明。

提示：

程序请求存取页的时候，先检查TLB表，页表项在TLB中，生成物理地址；不在TLB中，访问页表。

不在TLB中，访问页表，不在页表中，说明信息不在主存中，从磁盘读取页。在页表中，生成物理地址。

根据映射规则和生成的物理地址，查找cache，不在cache中从内存装入cache，在cache中从cache中取出相应的内容。

第六题

7	2	7
TLB标记	TLB组索引	页内偏移
虚页号		

请回答下列问题：

（1）虚拟地址中哪几位表示虚拟页号、哪几位表示页内偏移量？虚拟页号中哪几位表示TLB标记？哪几位表示TLB组索引？

（2）物理地址中哪几位表示物理页号、哪几位表示页内偏移量？在访问Cache时，物理地址如何划分成标记字段、行索引字段和块内地址字段？

（3）CPU从地址067AH中取出的值是多少？要求对CPU读取地址067AH中内容的过程进行详细说明。

6	4	2
Cache标记	Cache行索引	块内地址

- 地址067AH=0000 0110 0111 1010B，所以，虚页号为0000 0110 0B，映射到TLB的第00组
- 将0000 011B=03H与TLB第0组的四个标记比较，虽然和其中一个相等，但对应的有效位为0，其余都不相等，所以TLB缺失，需要访问主存中的页表
- 直接查看0000 0110 0B=00CH处的页表项，有效位为1，取出物理页号19H=1100 1B，和页内偏移111 1010B拼接成物理地址：1100 1111 1010B
- 根据中间4位1110直接找到Cache第14行(即第E行)，有效位为1，且标记为33H=11 0011B，正好等于物理地址高6位，故命中
- 根据物理地址最低两位10，取出字节2中的内容4AH=0100 1010B

第七题

在组相联映射方式下：

主存： 2M Bytes

Cache：32K Bytes

Block：256 Bytes

Way ： 8 Ways（Cache每组包含8个Block）

请计算：

Cache 组数

主存每组块数

主存地址分为哪几个部分，每部分的位数

Cache的Tag的位数：

此外，试问在组相联映射方式下，会否出现Cache不满，但新块需启动替换才能调入的现象？

第七题

在组相联映射方式下：

主存： 2M Bytes

Cache：32K Bytes

Block：256 Bytes

Way ： 8 Ways（Cache每组包含8个Block）

请计算：

Cache 组数：32K/256/8=16组

主存每组块数：2M/256/16=512块

主存地址分为哪几个部分，每部分的位数：地址21位

Tag：9	Set#：4	Offset：8

Cache的Tag的位数：9

此外，试问在组相联映射方式下，会否出现Cache不满，但新块需启动替换才能调入的现象？

会！

计算机组成习题参考答案

--总线与I/O

一

某计算机的I/O 设备采用异步串行方式传送字符信息，字符信息的格式为：

- 1 位起始位
- 7 位数据位
- 1 位检验位
- 1 位停止位

若要求每秒钟传送 480 个字符，那么该 I/O 设备的数据传输率应该为多少 bps（位 / 秒）？

➢233

一

某计算机的I/O 设备采用异步串行方式传送字符信息，字符信息的格式为：

- 1 位起始位
- 7 位数据位
- 1 位检验位
- 1 位停止位

若要求每秒钟传送 480 个字符，那么该 I/O 设备的数据传输率应该为多少 bps（位 / 秒）？

参考答案：

- 传送 1 个字符实际在总线上需传输的二进制位是： 1(起始)+7(数据)+1（检验）+1（停止）= 10bits
- 现在要求每秒钟传送 480 个字符，即每秒需传输的二进制位是：480 X 10 = 4800bits
- 也即该 I/O 设备的数据传送率为：4800bps

➢234

二

某系统对输入数据进行取样处理，每抽取一个输入数据，CPU就要中断处理一次，将取样的数据放至存储器中保留的缓冲区内，该中断处理需P秒。此外，缓冲区内每存储N个数据，主程序就将其取出进行处理，处理过程需Q秒。请问，该系统可以跟踪到每秒多少次中断请求？

➢235

二

某系统对输入数据进行取样处理，每抽取一个输入数据，CPU就要中断处理一次，将取样的数据放至存储器中保留的缓冲区内，该中断处理需P秒。此外，缓冲区内每存储N个数据，主程序就将其取出进行处理，处理过程需Q秒。请问，该系统可以跟踪到每秒多少次中断请求？

- 本题的问题可转化为：抽取并处理N个数据（也即跟踪N次中断），需要多少时间？
- 中断抽取需时：N*P
- 主程序处理需时：Q
- 共需时：N*P+Q
- 每秒可跟踪的中断请求为：N/(N*P+Q)

➢236

三

假设某设备向CPU传送信息的最高频率是40000次/秒，而相应的中断处理程序其执行时间为40us，试问该外设是否可用程序中断方式与主机交换信息？为什么 ？

➢237

三

假设某设备向CPU传送信息的最高频率是40000次/秒，而相应的中断处理程序其执行时间为40us，试问该外设是否可用程序中断方式与主机交换信息？为什么 ？

参考解答：

设备向CPU传送信息的最高频率40000次/秒，也就是说，该设备每隔1/40000=25μs需要向CPU传送一次信息。

但如采用程序中断方式，需40μs（>25μs）才能处理一次数据，会造成数据丢失，所以不能用程序中断方式与主机交换信息。

➢238

四

设磁盘存储器转速为3000RPM（转/分），分8个扇区，每扇区存储1K字节（1KB），主存与磁盘存储器数据传送的宽度为16位（即每次传送16位）。假设一条指令最长执行时间是25us，是否可采用一条指令执行结束时响应DMA请求的方案，为什么？若不行应采取什么方案 ？

➢239

四

设磁盘存储器转速为3000RPM（转/分），分8个扇区，每扇区存储1K字节（1KB），主存与磁盘存储器数据传送的宽度为16位（即每次传送16位）。假设一条指令最长执行时间是25us，是否可采用一条指令执行结束时响应DMA请求的方案，为什么？若不行应采取什么方案 ？

参考解答：

磁盘的转速：3000转/60分=50转/秒

即磁盘每秒传送：1KBX8X50=400KB信息

因主存与磁盘的数据传送宽度为16位，若采用DMA方式，每秒需DMA请求的次数为：400KB/2B=200K次。也即每隔1/200K≈5μs需要有一次DMA请求。

如按一条指令执行结束（25μs）响应DMA请求，会造成数据丢失。

因此，可采用按照每隔存取周期结束响应DMA请求的方案。

➢240

五

设磁盘的磁头数为H，每个盘面的磁道数为C，每条磁道上有S个扇区，每扇区可存储N个字节，试列出磁盘容量的计算公式。

设磁盘的转速为W（RPM），每条磁道上存储N个字节，试列出磁盘数据传输率（单位：bit/s）的计算公式。

➢241

五

设磁盘的磁头数为H，每个盘面的磁道数为C，每条磁道上有S个扇区，每扇区可存储N个字节，试列出磁盘容量的计算公式。

H * C * S * N (Bytes)

设磁盘的转速为W（RPM），每条磁道上存储N个字节，试列出磁盘数据传输率（单位：bit/s）的计算公式。

(N*8)/(60/W) (bit/s)

➢242

六

分要点简述：调用中断服务程序和调用子程序的主要区别

在程序中断方式中，磁盘申请中断的优先级高于打印机。 当打印机正在打印时，磁盘申请中断，试问是否要将打 印机的打印操作停下来，等磁盘操作结束后，打印机才能继续打印？为什么？

➢243

六

分要点简述：调用中断服务程序和调用子程序的主要区别

参考解答：

- 中断服务程序与主程序是相互独立的，它们之间没有确定的关系。子程序调用时，转入的子程序与CPU正在执行的程序是同一程序的两个部分。
- 除软中断外，通常中断产生是随机的。而子程序调用是由子程序调用指令（如Call指令）引起的，是编程时预先安排好的。
- 中断服务程序的入口地址由向量地址找到。而子程序入口地址通常是由Call指令中的地址码给出。
- 两种调用都需保护程序断点，但前者通过中断隐指令完成，而后者由Call指令本身完成。
- 处理中断服务程序时，对多个同时发生的中断需进行仲裁，而调用子程序时一般没有这种操作。

➢244

六

在程序中断方式中，磁盘申请中断的优先级高于打印机。 当打印机正在打印时，磁盘申请中断，试问是否要将打 印机的打印操作停下来，等磁盘操作结束后，打印机才 能继续打印？为什么？

参考解答：

- 通常，打印机的打印动作只受打印机本身控制，与CPU无直接关系，因此，当打印机正在打印时，即使有优先级更高的磁盘请求中断，打印机也不会停止打印。
- 但是，如果CPU正在执行打印机的中断服务程序，即打印机可能正在接收数据，此时，若磁盘请求中断，CPU就要中断正在运行的打印机中断服务程序，向打印机的数据传送会受到影响。

七

- **假设：磁盘分16个扇区，每扇区存储512个字节，转速为6000 RPM。主存与磁盘之间的数据传送采用DMA单字传送方式，单字长为64位。一条指令最长执行时间是20微秒。**
- **请问：**
- **（1）该磁盘的数据传输率（bytes/s）是多少？**
- **（2） 是否可采用一条指令执行结束时响应DMA请求的方案，为什么？若不行，可采取什么方案?**

七

- **假设：磁盘分16个扇区，每扇区存储512个字节，转速为6000 RPM。主存与磁盘之间的数据传送采用DMA单字传送方式，单字长为64位。一条指令最长执行时间是20微秒。**
- **请问：**
- **（1）该磁盘的数据传输率（bytes/s）是多少？**

磁盘的转速：6000转/60秒 = 100 转/秒，该磁盘的数据传输率为：512B × 16 × 100 = 800 KB/s，即磁盘每秒传送800 KB信息。

- **（2） 是否可采用一条指令执行结束时响应DMA请求的方案，为什么？若不行，可采取什么方案?**

采用DMA单字（64位）传送方式，每秒需DMA请求的次数为：800KB÷8B = 100K次。也即每隔1/100K ≈ 10us需要有一次DMA请求。

如按一条指令执行结束（20us）响应DMA请求，会造成数据丢失。因此，不能采用一条指令执行结束时响应DMA请求的方案。

可采用在每个存取周期结束时响应DMA请求的方案。

八、

假设一32位处理器总线时钟频率为400MHz，支持多种总线事务。其中最短的总线事务为存储器读事务，需要4个总线时钟周期，第一个时钟周期传送地址和读命令，第4个时钟周期取数；最长的总线事务是突发传送8次数据，需要11个总线时钟周期完成，第一个时钟周期传送地址和读命令，第4个时钟周期开始连续传送8个数据，每个时钟周期传送一次数据。

1）该总线是同步总线还是异步总线；

同步

2）总线的最大数据传输率为多少；

32b x 400MHz = 1600MB/s

3）若处理器一直持续发起最短总线事务，则此时总线数据传输率是多少？

32b x 400MHz / 4 = 400MB/s

4）若处理器一直持续发起最长总线事务，则此时总线数据传输率是多少？

32b x 8 x 400MHz / 11 ≈ 1163.6MB/s

九、

某计算机字长为32位，CPU主频为500MHz，CPI为5（即执行每条指令平均需5个时钟周期）。假定某外设的数据传输率为0.5MB/S，采用中断方式与主机进行数据传送，每次传送32位，对应的中断服务程序包含18条指令，中断服务的其他开销相当于2条指令的执行时间。请回答下列问题，要求给出计算过程。

1. 中断方式下CPU用于该外设I/O的时间占CPU时间的百分比是多少？
2. 若该外设的数据传输率为5MB/S，改用DMA方式传送数据，假定每次DMA传送块大小为5000字节，且DMA预处理和后处理的总开销为500个时钟周期，则CPU用于该外设I/O的时间占整个CPU时间的百分比是多少？（假设DMA与CPU之间没有访存冲突）。

➢249

九、

某计算机字长为32位，CPU主频为500MHz，CPI为5（即执行每条指令平均需5个时钟周期）。假定某外设的数据传输率为0.5MB/S，采用中断方式与主机进行数据传送，每次传送32位，对应的中断服务程序包含18条指令，中断服务的其他开销相当于2条指令的执行时间。请回答下列问题，要求给出计算过程。

1. 中断方式下CPU用于该外设I/O的时间占CPU时间的百分比是多少？

❖在中断方式下，每32位(4B)被中断一次，故每秒中断次数：

0.5MB/4B=0.5×10^6/4=0.125×10^6次

- 因为中断服务程序包含18条指令，中断服务的其他开销相当于2条指令的执行时间，且执行每条指令平均需5个时钟周期，所以，1秒内用于中断的时钟周期数为

(18+2)×5×0.125×10^6=0.125×10^8

- 所以CPU用于该外设I/O的时间占整个CPU时间的百分比是：

0.125*10^8/500*10^6=2.5%

➢250

九

2. 若该外设的数据传输率为5MB/S，改用DMA方式传送数据，假定每次DMA传送块大小为5000字节，且DMA预处理和后处理的总开销为500个时钟周期，则CPU用于该外设I/O的时间占整个CPU时间的百分比是多少？（假设DMA与CPU之间没有访存冲突）。

❖在DMA方式下，每秒进行DMA操作次数：

5MB/5000B=5×10^6/5000=1×10^3次

- 因为DMA预处理和后处理的总开销为500个时钟周期，所以1秒钟之内用于DMA操作的时钟周期数为

500×1×10^3=5×10^5

- 故在DMA方式下，占整个CPU时间的百分比是

(5×10^5)/(500×10^6))×100%=0.1%

➢251