计算机组成原理

----课堂例题汇总

【例】 假设一台计算机主频为1GHZ，在其上运行由2×10^5条指令组成的目标代码，程序主要由4类指令组成，他们所占的比例和各自的CPI如下表所示，求程序的CPI和MIPS及CPU时间T_{CPU}。

指令类型	CPI	指令比例
算术和逻辑	1	60%
Load/Store	2	18%
转移	4	12%
Cache缺失访存	8	10%

【解】：$CPI = 1\times60\% + 2\times18\% + 4\times12\% + 8\times10\% = 2.24$

$$MIPS = f/(CPI \times 10^6) = 1\times10^9/(2.24\times10^6) = 446.4$$

$$T_{CPU} = 2\times10^5 \times CPI / f = (2\times10^5 \times 2.24 / 10^9) = 4.48\times10^{-4} \text{(秒)}$$

【例】：已知机器字长16位，求十进制数N=-65的原码，反码和补码。

【解】：首先将数N转换成二进制数（数值位15位）：

N＝-000,0000,0100,0001

原码： 1000,0000,0100,0001B=8041H

反码： 1111,1111,1011,1110B=FFBEH

补码： 1111,1111,1011,1111B=FFBFH

【小问题】如果用移码表示，则数N的编码是多少？

【例】：已知机器字长16位，数N的机器码编码为8001H，求其机器码分别为原码，反码和补码时，其十进制真值分别是多少？

【解】：首先将数N编码转换成二进制数：

N＝1000, 0000, 0000, 0001

原码： 符号位=1，数值位=000, 0000, 0000, 0001

即N原码表示时，其十进制真值为：-1

反码： 1000, 0000, 0000, 0001 转换成原码：1111, 1111, 1111, 1110

可得，N反码表示时，其十进制真值为：-32766

补码： 1000, 0000, 0000, 0001 转换成原码：1111, 1111, 1111, 1111

可得，N补码表示时，其十进制真值为：-32767

【练】设机器字长16位，定点整数，若数X的移码为7FFEH，问：

(1) 其原码，反码，补码各是多少？

(2) 数X的真值是多少？

【解】 先求补码，把数X的移码符号位取反，即得X的补码

(1) $[X]_{移}$= 7FFEH =0111，1111，1111，1110 B

$[X]_{补}$= 1111，1111，1111，1110B = FFFEH

又 ∵ 负数 $[X]_{补}=[X]_{反}+1$

∴ $[X]_{反}= [X]_{补}-1$=FFFDH

$[X]_{原}$=1000，0000，0000，0010B =8002H

(2) 由（1）中的原码，可得数X的真值为：-2

【练】设机器字长为8位,用于表示定点小数，其中已知符号位占1位，数值位7位；若数 N=-23/128，求N的原码，反码，补码（用二进制表示）。

【解】 先求数N的二进制表示，∵ 分子 23=10111,分母 128=2^7

∴ N=−0.0010111

因此，数N的原码，反码，补码分别为：

$[X]_{原}$= 1.0010111

$[X]_{反}$= 1.1101000

$[X]_{补}$= 1.1101001

【例】：将十进制数20.59375转换成32位浮点数的二进制格式来存储。

【解】：首先分别将整数和分数部分转换成二进制数：

$$20.59375 = 10100.10011$$

然后移动小数点，使其在第1，2位之间，即：

$$10100.10011 = 1.010010011 \times 2^4 \qquad e = 4$$

$S = 0$，$M = 010010011$

∵ $e = E - 127$，∴ $E = 4 + 127 = 131 = 1000,0011$

最后得到32位浮点数的二进制存储格式为：

0100,0001,1010,0100,1100,0000,0000,0000＝ 41A4C000H

【例】若单精度浮点数*N*的IEEE754格式为41360000H，求其32位浮点数的十进制值。

【解】：将N展开成二进制： 0100, 0001, 0011, 0110, 0000, 0000, 0000, 0000

数符：0； 阶码：1000, 0010

尾数：011, 0110, 0000, 0000, 0000, 0000

指数e＝阶码－127＝10000010－01111111 ＝00000011=$(3)_{10}$

包括隐藏位1的尾数：

1.*M*＝1.011 0110 0000 0000 0000 0000＝1.011011

则： $N=(-1)^s\times 1.M\times 2^e = +(1.011011)\times 2^3$

$=+1011.011=(11.375)_{10}$

【练】已知x=-0.01111,y=+0.11001,用变形补码计算:

（1）x+y;　　（2）x-y;

并判断是否溢出，若溢出，指出是正溢（上溢）还是负溢（下溢）

【解】：$[x]_{补}$=11.10001，$[y]_{补}$=00.11001，$[-y]_{补}$=11.00111

（1） x+y

$[x]_{补}+[y]_{补}$=11.10001+00.11001=00.01010

即 x+y=+0.0101

（2） x-y

$[x]_{补}+[-y]_{补}$=11.10001+11.00111=10.11000

即x+y的结果为溢出，负溢（或 下溢）

【例】 已知x=-0.1101，y=0.0101，用原码一位乘法计算x×y。

【解】：$[x]_{原}=1.1101$， $[y]_{原}=0.0101$

乘积的符号位为：$x_f \oplus y_f=1 \oplus 0=1$

令 $x'=|x|=0.1101$，

$|y|=0101$ （只取数值位）

	部分积	乘数	说明
	00.0000	0101	部分积P_0=0
+|x|	00.1101		y_4=1，+|x|
	00.1101	0101	
→	00.0110	1010	右移1位，得P_1
+0	00.0000		y_3=0，+0
	00.0110	1010	
→	00.0011	0101	右移1位，得P_2
+|x|	00.1101		y_2=1，+|x|
	01.0000	0101	
→	00.1000	0010	右移1位，得P_3
+0	00.0000		y_1=0，+0
	00.1000	0010	
→	00.**0100**	**0001**	右移1位，得P_4=|x|×|y|

即 $[x \times y]_{原}$=1.01000001， 所以 x×y=-0.01000001

【练】 已知x=+13，y=-5，用原码一位乘法计算x×y。设x，y的数值位均为4位。

【解】：$[x]_{原}$=0,1101， $[y]_{原}$=1,0101

乘积的符号位为：$x_f \oplus y_f = 0 \oplus 1 = 1$

令 $x' = |x| = 0,1101$,

$|y| = 0101$ （只取数值位）

	部分积	乘数	说明
	00,0000	0101	部分积P_0=0
+|x|	00,1101		y_4=1，+|x|
	00,1101	0101	
→	00,0110	1010	右移1位，得P_1
+0	00,0000		y_3=0，+0
	00,0110	1010	
→	00,0011	0101	右移1位，得P_2
+|x|	00,1101		y_2=1，+|x|
	01,0000	0101	
→	00,1000	0010	右移1位，得P_3
+0	00,0000		y_1=0，+0
	00,1000	0010	
→	00,**0100**	**0001**	右移1位，得P_4=|x|×|y|

即 $[x \times y]_{原}$=1,01000001， 所以 x×y=-1000001 （即-65_{10}）

【例】 已知x=-0.1101，y=0.0101，用补码一位乘法计算x×y。

【解】：$[x]_{补}=1.0011$

$[-x]_{补}=0.1101$

$[y]_{补}=0.0101$

	部分积	乘数	说明
	00.0000	001010	部分积$[P_0]_{补}$=0，附加位y_{n+1}=0
$+[-x]_{补}$	00.1101	y_{n+1}	$y_n y_{n+1}$=10，$+[-x]_{补}$
	00.1101	001010	
	00.0110	100101	右移1位，得$[P_1]_{补}$
$+[x]_{补}$	11.0011		$y_n y_{n+1}$=01，$+[x]_{补}$
	11.1001	100101	
	11.1100	110010	右移1位，得$[P_2]_{补}$
$+[-x]_{补}$	00.1101		$y_n y_{n+1}$=10，$+[-x]_{补}$
	00.1001	110010	
	00.0100	111001	右移1位，得$[P_3]_{补}$
$+[x]_{补}$	11.0011		$y_n y_{n+1}$=01，$+[x]_{补}$
	11.0111	111001	
	11.1011	111100	右移1位，得$[P_4]_{补}$
$+[0]_{补}$	00.0000		$y_n y_{n+1}$=00，$+[0]_{补}$
	11.1011	111100	最后一步不移位，得$[x \times y]_{补}$

$[x \times y]_{补}$=**1.10111111** 所以 $x \times y$=**-0.01000001**

【例】 设 x ＝＋15， y ＝－13，用带求补器的补码阵列乘法器求出乘积 x · y ＝？

【解】 设最高位为符号位，则输入数据为： $[x]_{补}$ =0,1111 $[y]_{补}$ = 1,0011

符号位单独运算，算前求补输出为： |x|=1111， |y|=1101

阵列运算：

```
                    1  1  1  1
  ×)                1  1  0  1
-------------------------------
                    1  1  1  1
                 0  0  0  0
              1  1  1  1
           1  1  1  1                 0
-------------------------------------
        1  1  0  0  0  0  1  1
```

算后求补输出为： 00111101， 最后乘积 $[x \cdot y]_{补}$=1,00111101

计算结果真值： $x \cdot y = -11000011_2 = -195_{10}$

【例】 设 x =−0.1011， y =+0.0011，用带求补器的补码阵列乘法器求出乘积 $x \cdot y$。

【解】 设最高位为符号位，则输入数据为： $[x]_{补}$ =1.0101 $[y]_{补}$ = 0.0011

符号位单独运算，算前求补输出为：|x|=1011， |y|=0011

阵列运算：

```
                          1  0  1  1
   ×)                     0  0  1  1
-------------------------------------
                          1  0  1  1
                       1  0  1  1
                    0  0  0  0
                 0  0  0  0                0
-------------------------------------
0  0  1  0  0  0  0  1
```

补足位数！！

算后求补输出为： 11011111， 最后乘积 $[x \cdot y]_{补}$=1.11011111

计算结果真值： x · y = -0.00100001

【例】 设 x =2^{010}×0.11011011， y =2^{100} ×(−0.10101100)， 求 x + y 。

【解】： 阶码采用双符号位，尾数采用双符号位，则它们的浮点表示分别为： $[x]_{浮}$= 00 010，00.11011011 ； $[y]_{浮}$= 00 100，11.01010100

(1) 求阶差并对阶

$\triangle E = E_x - E_y = [E_x]_{补} + [-E_y]_{补}$= 00 010 + 11 100 = 11 110

即△E为−2，x的阶码小，应使 M_x右移两位， E_x加2。

$[x]_{浮}$＝00 100， 00.00110110(11)

其中(11)表示M_x右移2位后移出的最低两位数。

(2) 尾数求和

$$\begin{array}{rr} & 00.00110110(11) \\ + & 11.01010100 \\ \hline & 11.10001010(11) \end{array}$$

(3) 规格化处理

尾数运算结果的符号位与最高数值位为同值，应执行左规处理，结果为11.00010101(10)， 阶码为00 011。

(4) 舍入处理 采用末位恒置1法，则有:11.00010101

(5) 判断溢出 阶码符号位为00，不溢出，故得最终结果为：

$$x + y = -0.11101011 \times 2^{011}$$

【练】 两浮点数$x = 2^{01}\times0.1101$，$y = 2^{11}\times(-0.1010)$。设尾数在计算机中以补码表示，数值位4位，阶码以补码表示，舍入处理采用截去法，求$x+y$。

【解】：将x,y转换成浮点数据格式

$[x]_{浮}$ = 00 01, 00.1101，　　$[y]_{浮}$ = 00 11, 11.0110

对阶：阶差为11-01=10，即2，因此将x的尾数右移两位，得

$[x]_{浮}$ = 00 11, 00.001101

尾数求和，得：　　$[x+y]_{浮}$ = 00 11, 11.100101

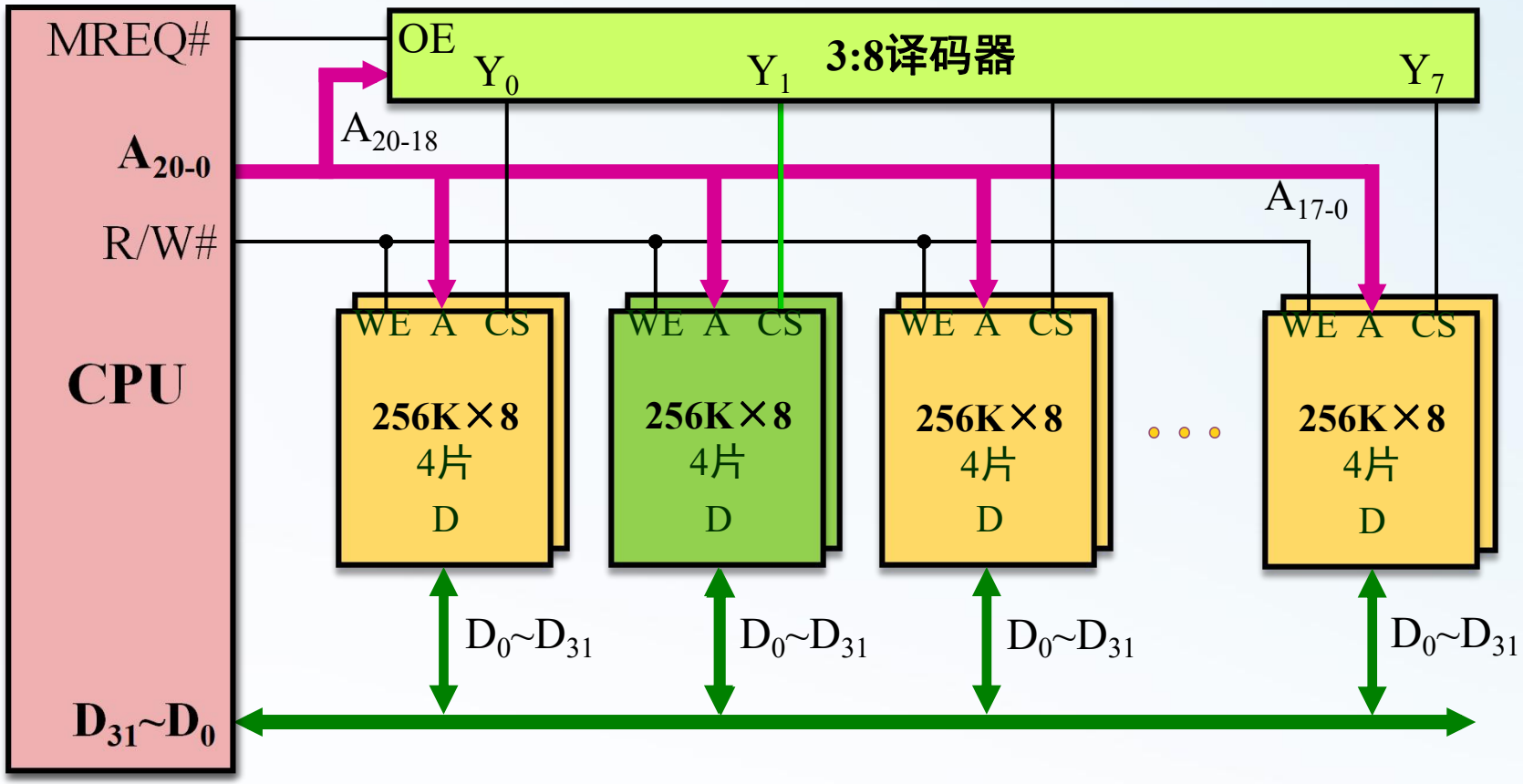
规格化：由于符号位和第一位数相等，不是规格化数，向左规格化，得

$[x+y]_{浮}$ = 00 10, 11.001010

舍入处理：截去法。　$[x+y]_{浮}$ = 00 10, 11.0010

溢出判断：数据无溢出，因此结果为　　$x+y = -0.1110\times 2^{10}$

字位同时扩展



【小练习】：给出各组存储芯片的地址范围。

填空题 20分

设置

给出对应各组（以译码器输出线标识）的地址范围：

	开始地址		末地址
首组Y0：	[填空1] H	------	[填空2] H
第6组Y6：	[填空3] H	------	[填空4] H

正常使用填空题需3.0以上版本雨课堂

作答

地址空间分配表

地址 / 片号	选片 A_{20} $A_{19}A_{18}$	片内 A_{17} A_{16} A_{15}……A_1 A_0	总地址 (十六进制)	说明
0	0 00	00,0000,0000,0000,0000 11,1111,1111,1111,1111	000000 03FFFF	最低地址 最高地址
.....	.....	.....	.....	最低地址 最高地址
6	1 10	00,0000,0000,0000,0000 11,1111,1111,1111,1111	180000 1BFFFF	最低地址 最高地址
7	1 11	00,0000,0000,0000,0000 11,1111,1111,1111,1111	1C0000 1FFFFF	最低地址 最高地址

【例】 CPU地址总线为16位，数据总线为8位，控制总线中，与主存有关的有MREQ#(允许访存，低电平有效)，R/W#（读写控制信号）使用4K×4位的SRAM芯片，设计一个存储容量为16K×8位的主存储器，画出主存储器与CPU的连接图。

【分析】 需要SRAM芯片数=(16K×8)/(4K×4)=8片

∵ 数据总线为8位，而SRAM芯片为4K×4位；

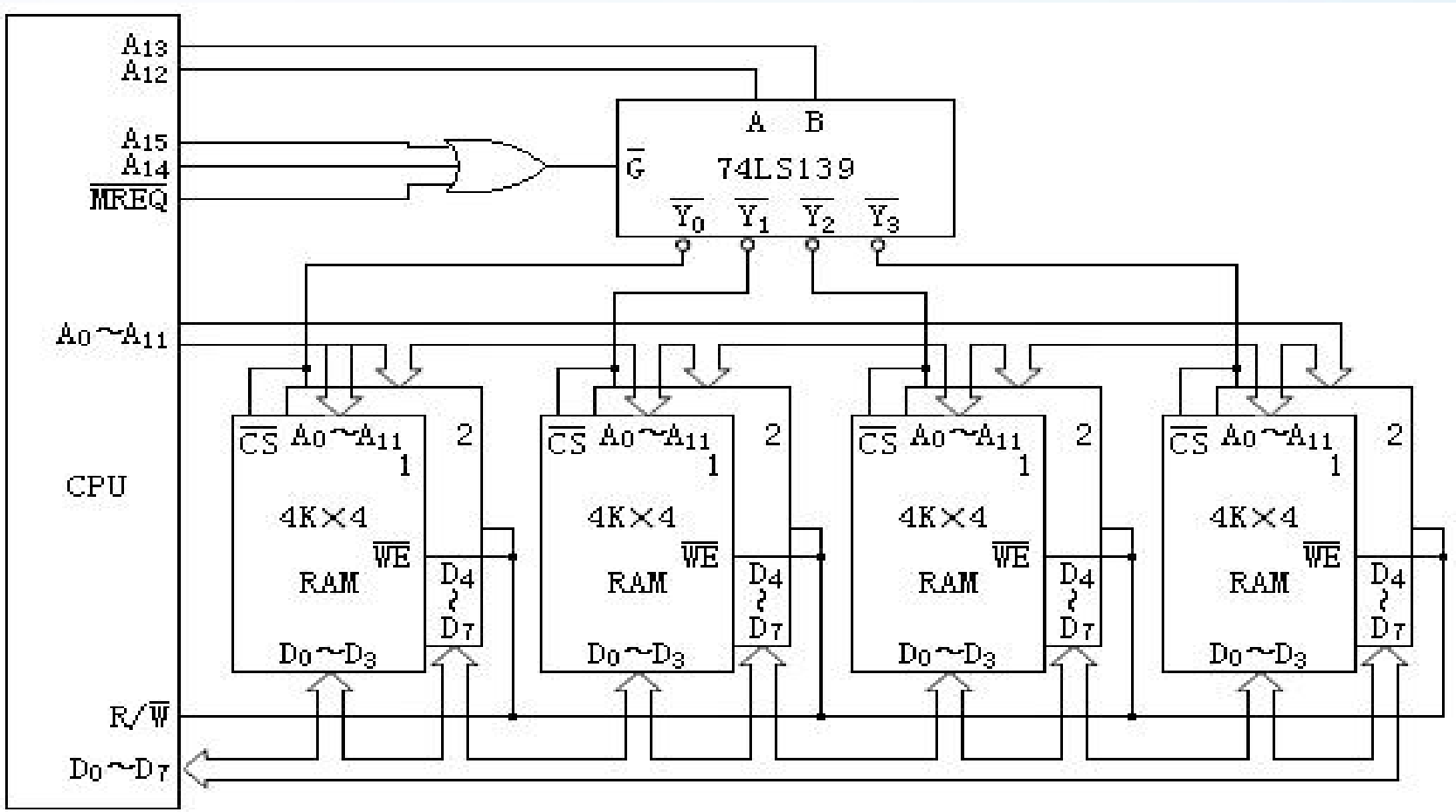
∴ 需要两片SRAM为一组。

又∵ $4K=2^{12}$ ， 故参与片内译码的地址线为12位，即A_{11}--A_0 ；剩余的4位地址线用于片间译码。

可以采用多种译码方案。

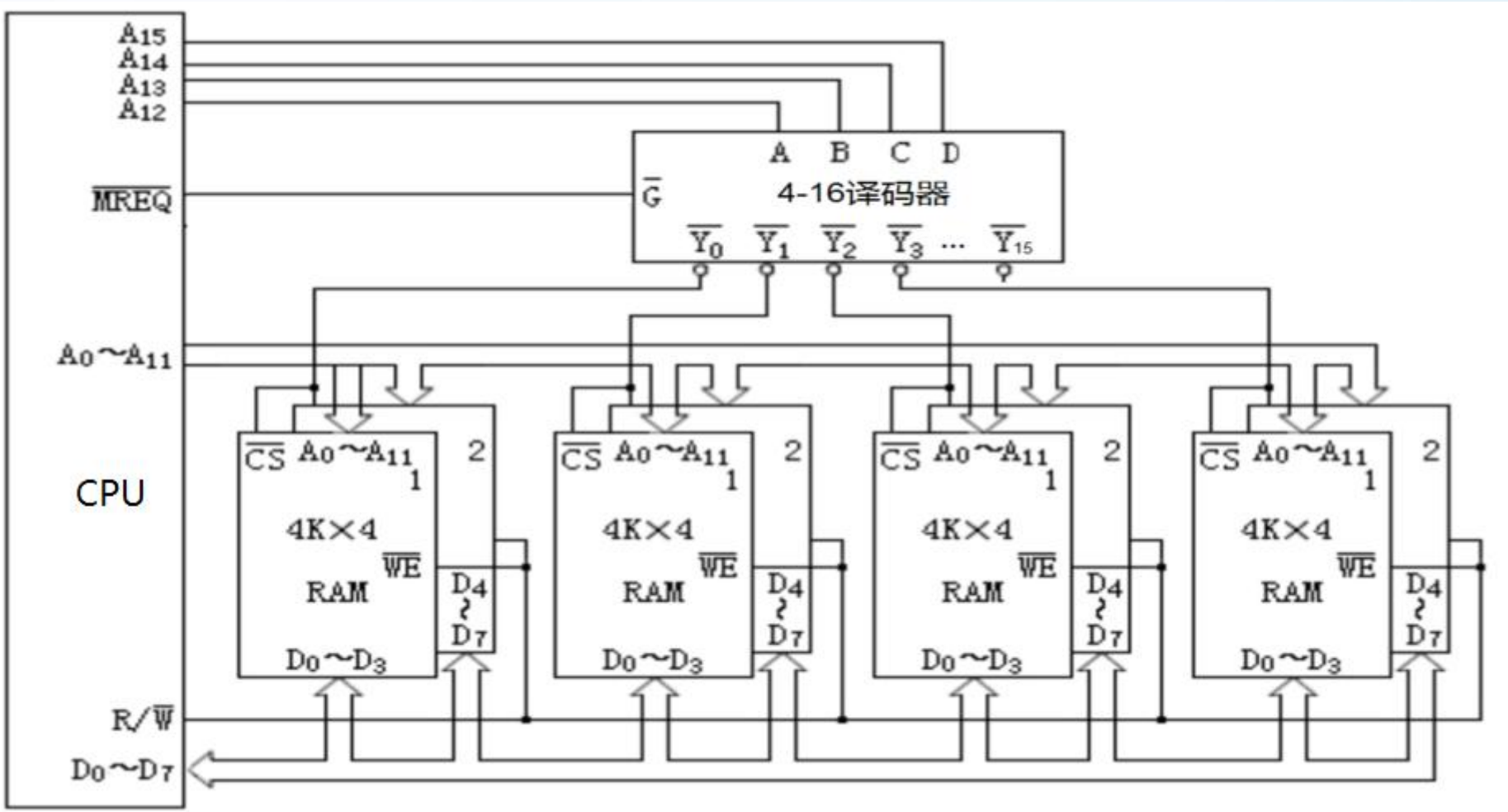
方案一： 采用2-4译码器

地址线$A_{13}A_{12}$用于片间译码，$A_{15}A_{14}$=0。



方案二： 采用4-16译码器

四根地址线$A_{15}A_{14}A_{13}A_{12}$ 都用于片间译码。



【思考】针对上述方案，试考虑：

1. 各组RAM的地址范围？

2. 若当前要访问如下地址单元，是否可行？若可行，指出所访问单元属于哪一组RAM。

（1） 45F9H； （2） 1570H； （3） 31FFH。

【解】

1. 各组RAM的地址范围：

第0组 0000H---0FFFH； 第1组 1000H---1FFFH；

第2组 2000H---2FFFH； 第3组 3000H---3FFFH；

2. （1） 45F9H； 不可行。

（2） 1570H； 可以访问，属于第1组RAM；

（3） 31FFH。 可以访问，属于第3组RAM。

【练习】 CPU的地址总线16根(A_{15}—A_0，A_0为低位)；双向数据总线8根(D_7—D_0)，控制总线中与主存有关的信号有： MREQ#，R/W#。

主存地址空间分配如下：

0—8191为系统程序区，由只读存储芯片组成；

8192—32767为用户程序区；

最后(最大地址)2K地址空间为系统程序工作区。

现有如下存储器芯片：

EPROM：8K×8位(控制端仅有CS)；

SRAM：16K×1位，2K×8位，4K×8位，8K×8位。

请从上述芯片中选择适当芯片设计该计算机主存储器，画出主存储器逻辑框图，注意画出选片逻辑(可选用门电路及3：8译码器74LS138)与CPU 的连接，说明选哪些存储器芯片，选多少片。

【解】：（1）主存地址空间分布如图所示。

16根地址线寻址 —— 64K

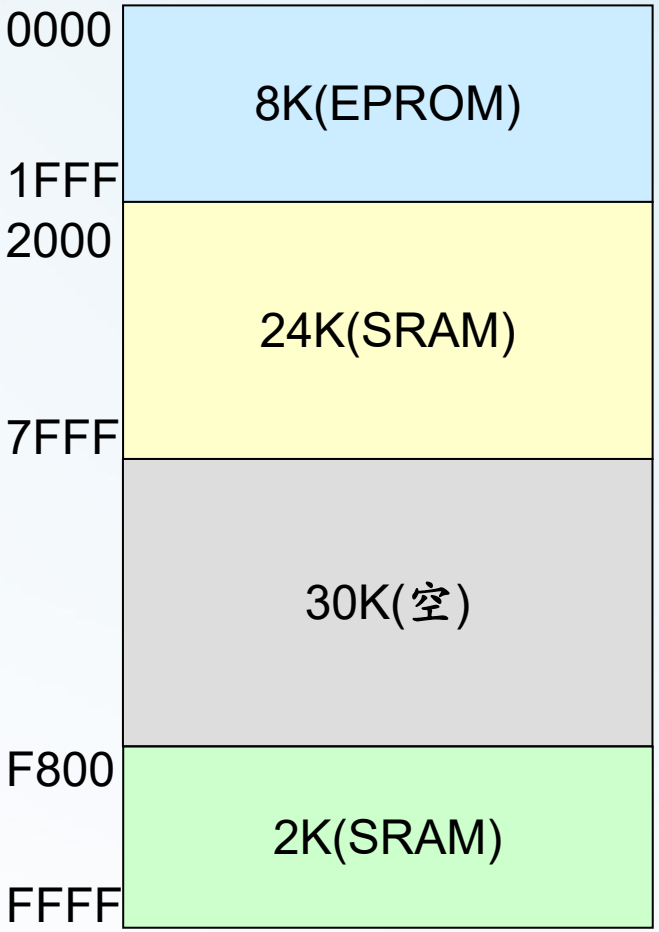
地址范围：0000 ~ FFFFH(65535)

根据条件，选定芯片：

EPROM：8K×8位 1片 系统程序区

SRAM： 8K×8位 3片 用户程序区

2K×8位 1片 系统程序工作区



地址空间分配表

	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
ROM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
RAM1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RAM2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
RAM3	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RAM4	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

(2) 连接电路

片内寻址：

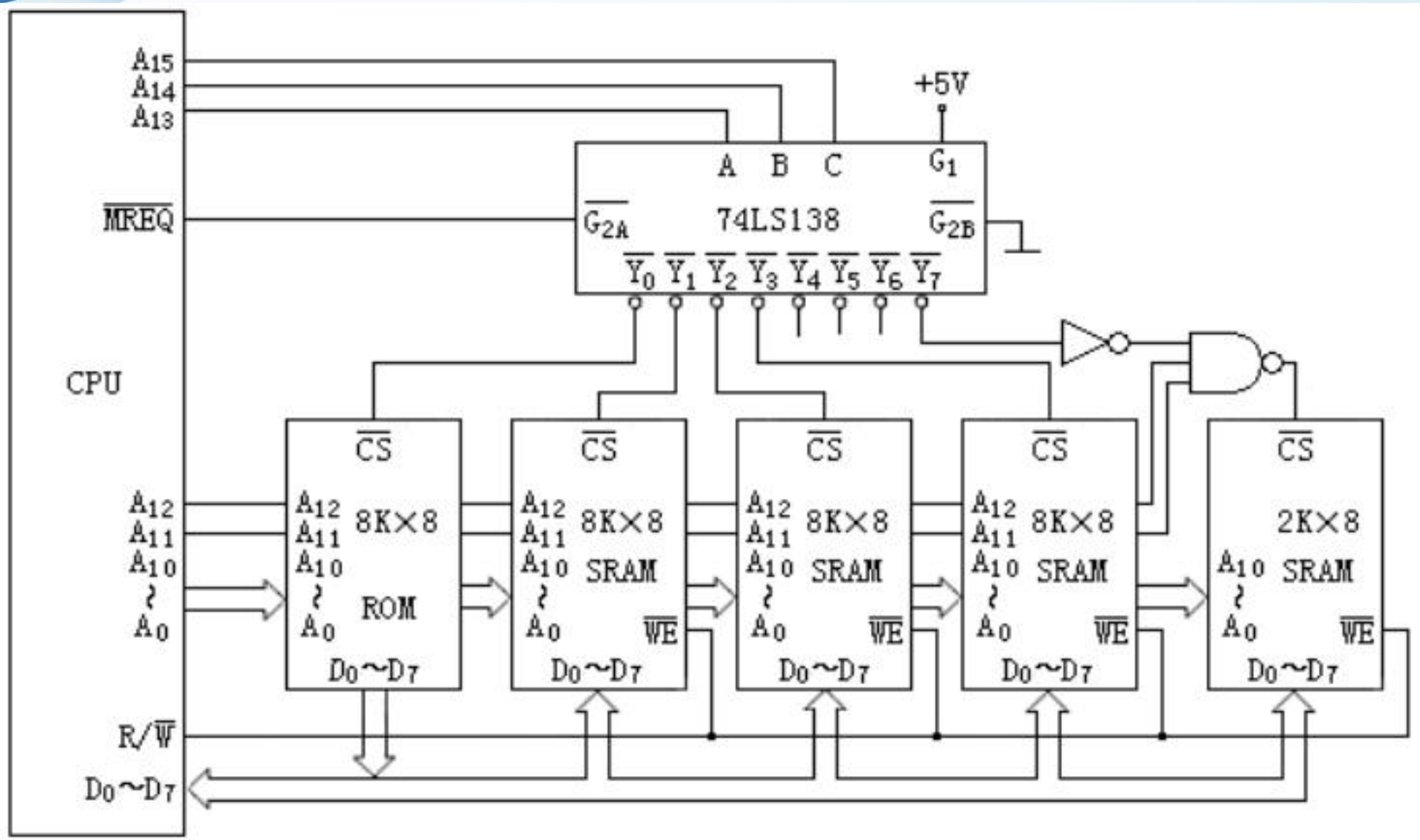
8K芯片——片内13根 A_{12}~A_0

2K芯片——片内11根 A_{10}~A_0

片间寻址：

	$A_{15}A_{14}A_{13}$	译码线
前32K	0 0 0	Y_0#
	0 0 1	Y_1#
	0 1 0	Y_2#
	0 1 1	Y_3#
最后2K	1 1 1 加 $A_{12}A_{11}$	
	1 1	

EPROM: 0000H~1FFFH
SRAM1: 2000H~3FFFH
SRAM2: 4000H~5FFFH
SRAM3: 6000H~7FFFH
SRAM4: F800H~FFFFH


A15
A14
A13
MREQ
CPU
A12
A11
A10
~
A0
R/W
D0~D7
+5V
A B C
G1
G2A
74LS138
G2B
Y0 Y1 Y2 Y3 Y4 Y5 Y6 Y7
CS
8K×8
ROM
SRAM
WE
2K×8

设置

【练习】CPU地址总线为20位，数据总线为8位，读/写控制信号为R/W#，访存允许信号为MREQ#。ROM芯片大小为128K×8位，起始地址为00000H；RAM区域大小为256KB，使用128K×4位的SRAM芯片，有WE#和CS#信号控制端，起始地址为80000H。问：

（1） RAM区域需要 [填空1] 片SRAM， 分为 [填空2] 组；

各组RAM的地址范围是：

1组首地址：[填空3] H-----末地址： [填空4] H

2组首地址：[填空5] H----- 末地址：[填空6] H

作答

设置

(2)给出连接图

作答

【解】：ROM芯片大小为128K×8位，起始地址为00000H；故其末地址为：1FFFFH（注：128K-1=2^{17}-1）

RAM区域大小为256KB，使用128K×4位的SRAM芯片，需要SRAM芯片数量为：256KB/128K×4位=4，因为数据总线为8位，故SRAM芯片需2片一组进行工作，因此，SRAM分为2组。

RAM芯片规格为：128K×4位，由此可知，其片内译码需要地址线数为17位（128K=2^{17}），系统地址线为20位，因此系统最高3位地址线可以用于片间译码，可使用3-8译码器。

地址空间分配表

	A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	.....	A_1	A_0
ROM	0	0	0	0	0	.....	0	0
	0	0	0	1	1	.....	1	1
RAM1	1	0	0	0	0	.....	0	0
	1	0	0	1	1	.....	1	1
RAM2	1	0	1	0	0	.....	0	0
	1	0	1	1	1	.....	1	1

使用3-8译码器，$A_{19}A_{18}A_{17}$作为输入。

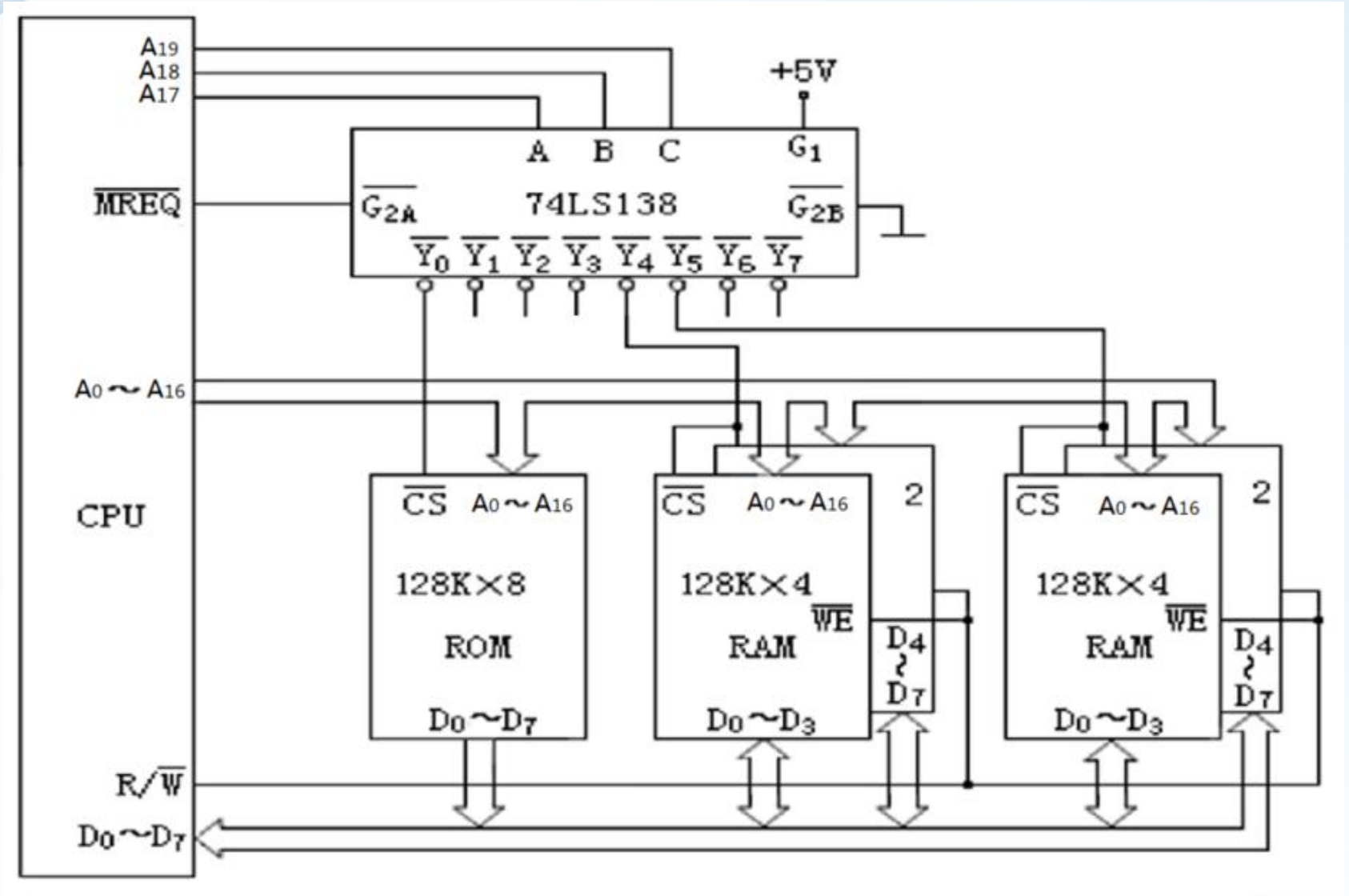
已知其起始地址为80000H。因此各组SRAM芯片地址范围为：

第一组：80000H---9FFFFH　其最高三位为 100

即其片选端接3-8译码器的Y4#输出端

第二组：A0000H---BFFFFH　其最高三位为 101

即其片选端接3-8译码器的Y5#输出端

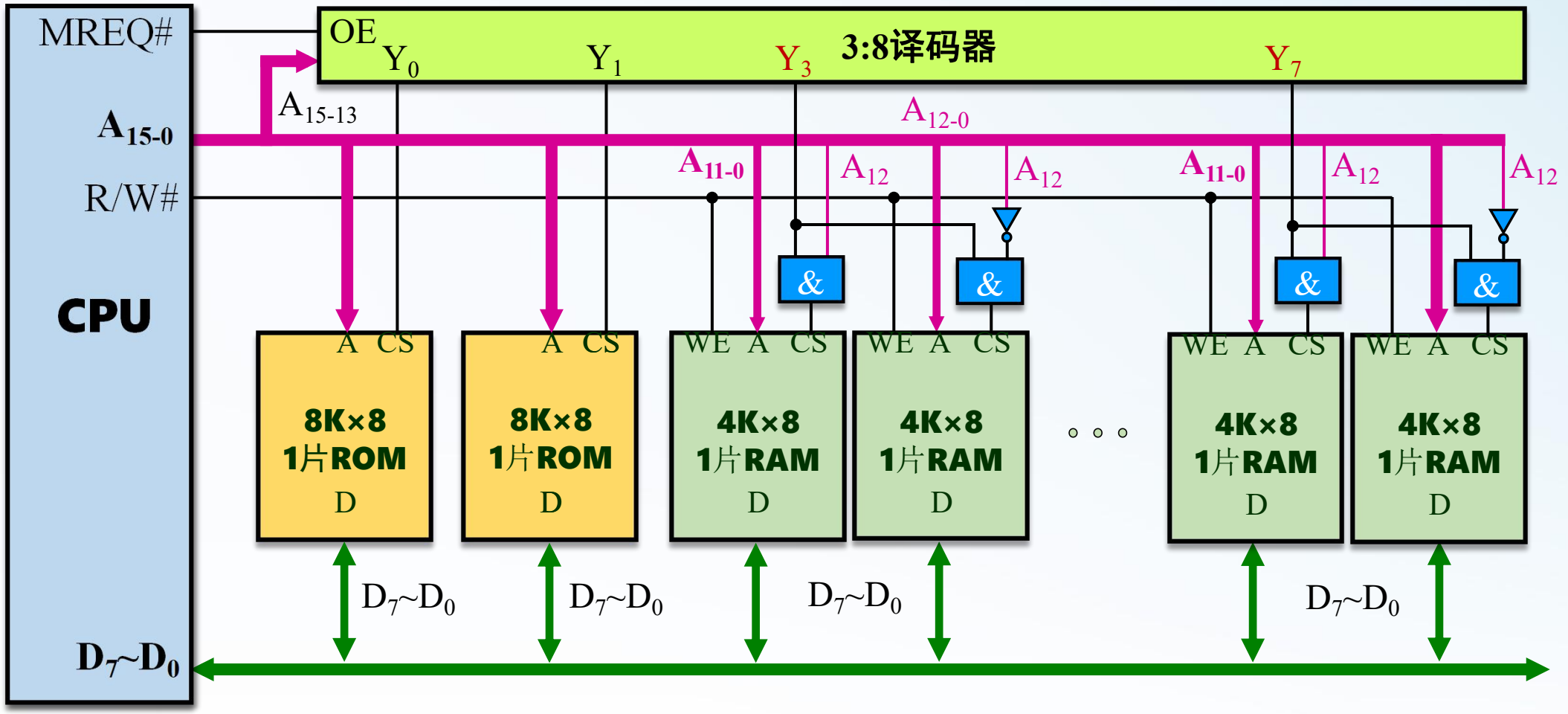

A19
A18
A17
MREQ
A0～A16
CPU
R/W
D0～D7
+5V
A B C G1
G2A 74LS138 G2B
Y0 Y1 Y2 Y3 Y4 Y5 Y6 Y7
CS A0～A16
128K×8
ROM
D0～D7
CS A0～A16
128K×4
WE
RAM
D0～D3
2
D4～D7
CS A0～A16
128K×4
WE
RAM
D0～D3
2
D4～D7

【练习】CPU地址总线为16位，数据总线为8位，读/写控制信号为R/W#，访存允许信号为MREQ#。SRAM芯片，有WE#和CS#信号控制端。现需设计56KB的存储器，其中ROM容量16KB，其首地址为0000H，RAM部分容量为40KB，其首地址为6000H片间译码采用3-8译码器，试根据给出的ROM、RAM芯片规格，完成存储系统设计：

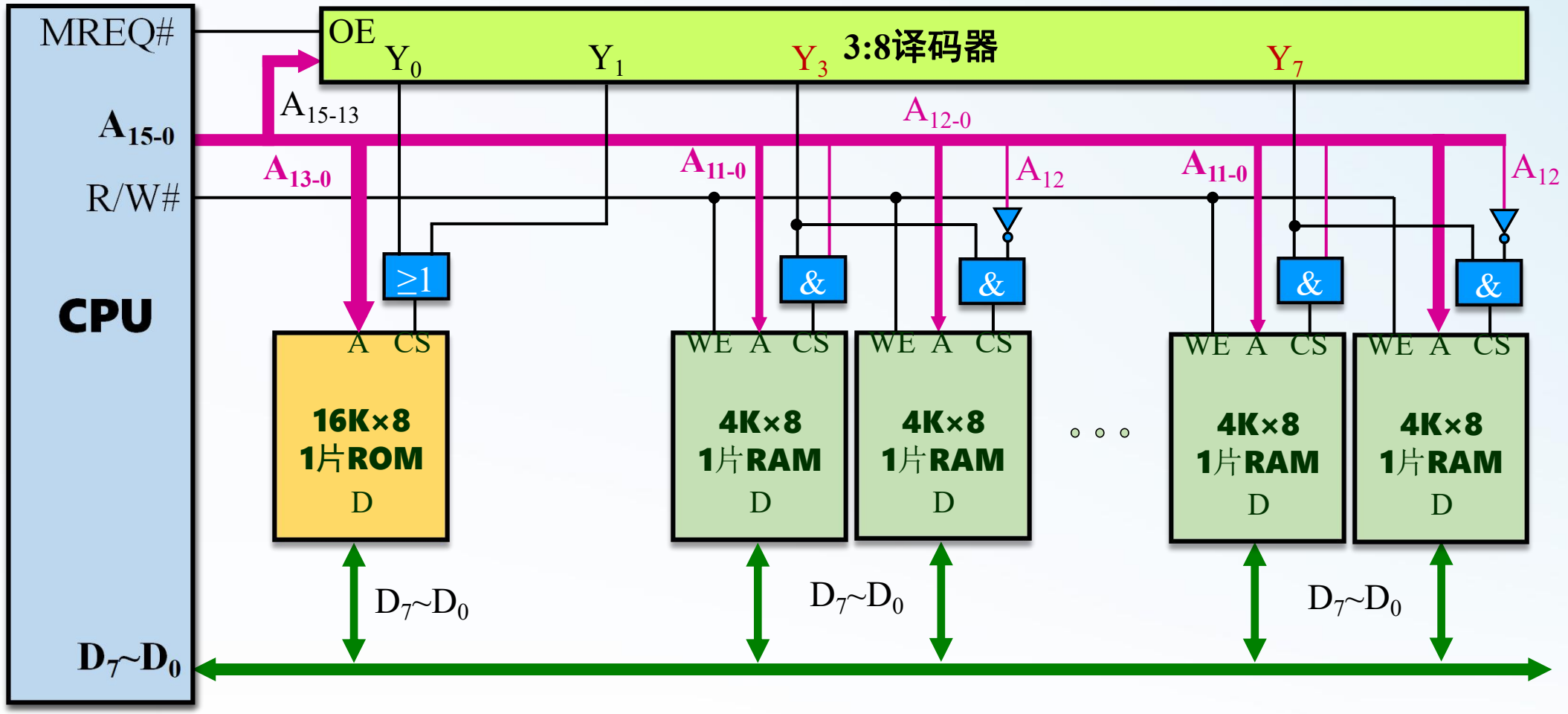
（1）方案一：ROM芯片：8K×8位；SRAM芯片：4K×8位。

（2）方案二：ROM芯片：16K×8位；SRAM芯片：4K×8位。

方案一： ROM芯片：8K×8位；SRAM芯片：4K×8位。



方案一： ROM芯片：16K×8位；SRAM芯片：4K×8位。



带宽计算

假设模块字长等于数据总线等宽（w位）。若模块存取一个字的存储周期是T，总线传送周期为τ，并使用m个模块来交叉存取，且$T=m\tau$，按地址顺序连续读取n个字，则顺序方式和交叉方式的带宽分别为：

按地址顺序连续读n个字，传递的信息总量q为：

$$q=w\times n$$

顺序方式所需时间为 $t_{顺}=nT$

交叉存储器所需时间为 $t_{交}=T+(n-1)\tau$

顺序存储器和交叉存储器的带宽分别是：

$$W_{顺}=q/t_{顺}=w\times n/nT=w/T$$

$$W_{交}=q/t_{交}=w\times n/(T+(n-1)\tau)$$

【例】 设主存储器容量为256字，字长为32位，模块数m=4，分别用顺序方式和交叉方式进行组织。主存储器的存储周期T=200ns，数据总线宽度为32位，总线传送周期τ=50ns。若按地址顺序连续读取4个字，问顺序存储器和交叉存储器的带宽各是多少？

【解】：顺序存储器和交叉存储器按地址顺序连续读出4个字的信息总量都是：

$q=32b\times4=128b$

顺序存储器和交叉存储器按地址顺序连续读出4个字所需的时间分别是：

$t_{顺}=nT=4\times200ns=800ns$

$t_{交}=T+(n-1)\tau=200ns+3\times50ns=350ns$

顺序存储器和交叉存储器的带宽分别是：

$W_{顺}=q/t_{顺}=128b\div800ns=160Mb/s$

$W_{交}=q/t_{交}=128b\div350ns\approx366Mb/s$

【练习】 已知主存储器的数据总线宽度为64位，存储周期T=400ns，总线传送周期τ=50ns，主存储器模块数m=8，分别采用顺序方式和交叉方式进行组织。若按地址顺序连续读取32个字，问顺序存储器和交叉存储器的带宽各是多少？

【解】：顺序存储器和交叉存储器按地址顺序连续读出32个字的信息总量都是：

$q=64b\times32=2048b$

顺序存储器和交叉存储器连续读出32个字所需的时间分别是：

$t_{顺}=nT=32\times400ns=12800ns$

$t_{交}=T+(n-1)\tau=400ns+31\times50ns=1950ns$

顺序存储器和交叉存储器的带宽分别是：

$W_{顺}=q/t_{顺}=2048b\div12800ns=160Mb/s$ （或20MB/s）

$W_{交}=q/t_{交}=2048b\div1950ns\approx1050Mb/s$ （或131MB/s）

【例】某计算机的 cache 由 1K个存储块构成，主存包含 64K 个存储块，每块由 256 个字组成，访问地址为**字地址**。

（1）主存地址和 cache 地址各有多少位?

（2）若采用全相联映射方式，列出主存地址的划分情况，并标出各部分的位数。

（3）若采用直接映射方式，列出主存地址的划分情况，并标出各部分的位数。

（4）若采用8路组相联映射方式，列出主存地址的划分情况，并标出各部分的位数。

【解】：

（1）存储块每块256字，即2^8字，块内偏移需要8位。

主存有64K块，即2^{16}块，主存块地址16位；

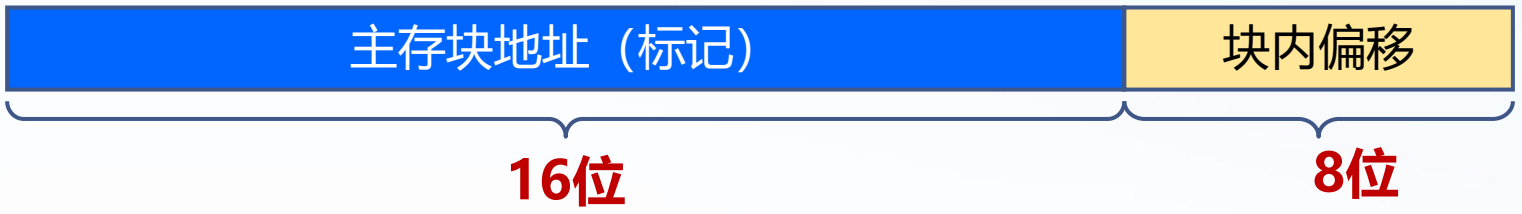
故主存地址位数=16+8=24。

Cache有1K块，即2^{10}块，Cache块地址10位。

故Cache地址位数=10+8=18。

（2）块内偏移字段位数，由（1）的分析可知，w=8位。

主存有64K块，即2^{16}块，主存块地址（标记）字段 s=16位。



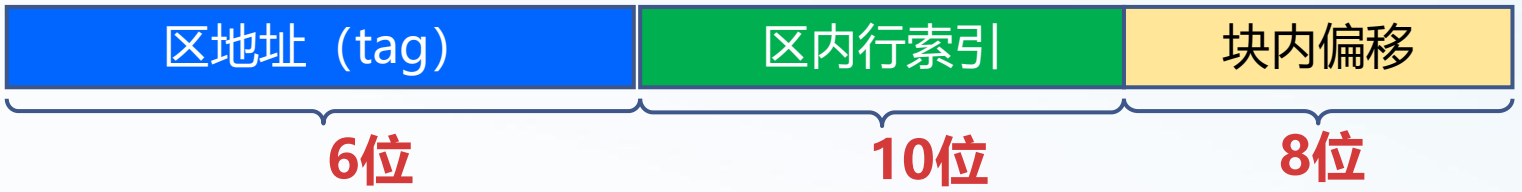
【解】：

（3）存储块每块256字，即2^8字，块内偏移需要8位。即 w=8位。

Cache有1k块，即2^{10}块（行），即区内行索引字段位数r=10位。

主存有64K块，即2^{16}块， s=16位。

区地址字段位数=s-r=16-10=6位。



【解】：

（4）存储块每块256字，即2^8字，**块内偏移**需要8位。即 w=8位。

Cache有1k块，由题干条件，采用8路组相联映射。

Cache分组数量=1K/8=128=2^7，由此可知，**组索引**字段位数d=7位。

主存有64K块，即2^{16}块， s=16位。

标记（tag）地址字段位数=s-d=16-7=9位。

【思考】若上例中，某计算机**字长32位**， cache 由 1K个存储块构成，主存包含 64K 个存储块，每块由 256 个字组成，访问地址为**字节地址**。

（1）主存地址和 cache 地址各有多少位?

（2）若采用全相联映射方式，列出主存地址的划分情况，并标出各部分的位数。

（3）若采用直接映射方式，列出主存地址的划分情况，并标出各部分的位数。

（4）若采用8路组相联映射方式，列出主存地址的划分情况，并标出各部分的位数。

【解】：（1）存储块每块256字，即2^8字，机器字长32位，即4字节=2^2字节。

所以每个存储块的大小=2^8字=2^{10}字节，因此块内偏移需要10位。

主存有64K块，即2^{16} 块，主存块地址16位；

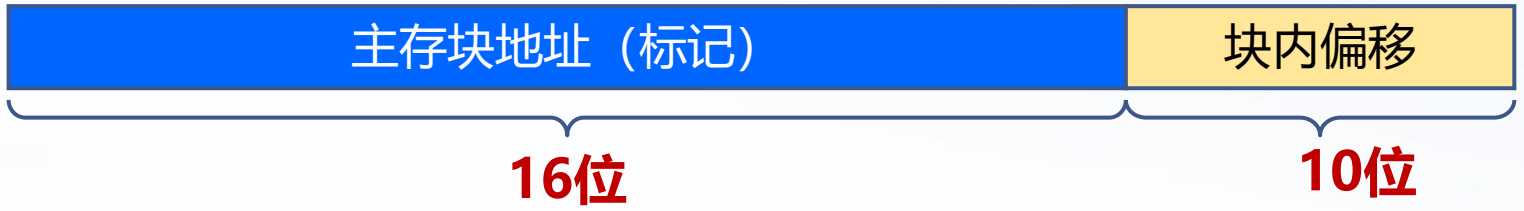
故主存地址位数=16+10=26。

Cache有1K块，即2^{10}块，Cache块地址10位。

故Cache地址位数=10+10=20。

（2）块内偏移字段位数，由（1）的分析可知，w=10位。

主存有64K块，即2^{16} 块，主存块地址（标记）字段 s=16位。

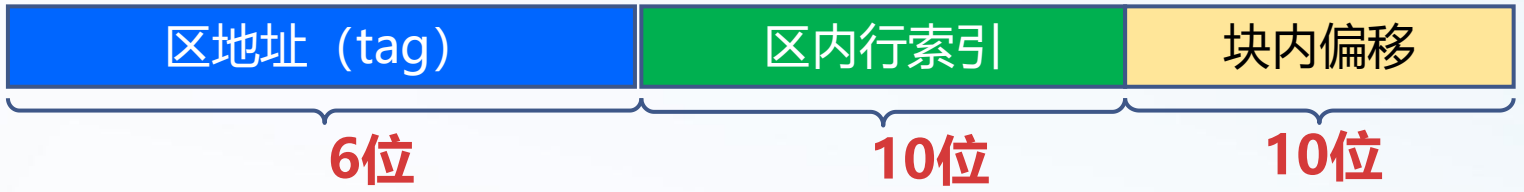


（3）由（1）可知，块内偏移需要10位。即 w=10位。

Cache有1k块，即2^{10}块（行），即区内行索引字段位数r=10位。

主存有64K块，即2^{16}块， s=16位。

区地址字段位数=s-r=16-10=6位。



【解】：

（4）由（1）可知，块内偏移需要10位。即 w=10位。

Cache有1k块，由题干条件，采用8路组相联映射。

Cache分组数量=1K/8=128=2^7，由此可知，**组索引**字段位数d=7位。

主存有64K块，即2^{16}块， s=16位。

标记（tag）地址字段位数=s-d=16-7=9位。

【例】CPU执行一段程序时，Cache完成存取的次数为1900次，主存完成存取的次数为100次，已知Cache存取周期为50ns，主存存取周期为250ns，求Cache/主存系统的效率和平均访问时间。

【解】先求出命中率及倍数：

$$h=\frac{N_c}{N_c+N_m}=\frac{1900}{1900+100}=0.95 \qquad r=\frac{t_m}{t_c}=\frac{250}{50}=5$$

Cache/主存系统的效率：

$$e=\frac{1}{h+(1-h)r}=\frac{1}{0.95+(1-0.95)\times 5}\approx 83.3\%$$

平均访问时间：

$$t_a=ht_c+(1-h)t_m=0.95\times 50ns+(1-0.95)\times 250ns=60ns$$

或

$$t_a=\frac{t_c}{e}=\frac{50\ ns}{0.833}=60\ ns$$

【例】CPU执行一段程序时，Cache完成存取的次数为1900次，主存完成存取的次数为100次，已知Cache存取周期为50ns，主存存取周期为250ns，求Cache/主存系统的效率和平均访问时间。

【思考】：结合本例数据，（1）若命中率h=95%不变，而把r的值依次改为10,100,1000时，效率e分别对应值为多少？

（2）若希望e=90%，则r的值依次改为10,100,1000时，对应的命中率各要求多少？ 从中可以得出什么结论？

（1）若命中率h=95%不变，而把r的值依次改为10,100,1000时，效率e分别对应值为69.0%，16.8%，1.96%

（2）若希望e=90%，则r的值依次改为10,100,1000时，对应的命中率各要求98.77%， 99.89%，99.99%

【例】某计算机的 cache 由 1K个存储块构成，主存包含 64K 个存储块，每块由 256 个字组成，采用全相联映射方式，访问地址为**字地址**。

（1）设cache为空，CPU从0单元开始，依次访问主存2048个字在（设访问主存一次读出一个字），重复5次，试计算cache访问的命中率。

（2）若cache的存取时间为50ns，r=8，试计算Cache/主存系统的访问效率和平均访问时间。

【解】：

（1）存储块每块256字，而CPU要访问主存单元从0开始的2048个字。对应的主存块号为：0,1,2，…，6,7共8块，每块256个字。

因cache初始时为空，访问0单元时，需装入主存0块数据（包含0,1,2，…，254,255共256个字单元），接下来访问1单元时，因已装入cache，命中；同理，访问2单元也命中，…，访问255单元也命中，即访问主存0块时，命中次数为255次。访问其它块时，除了第一次需要装入外，其它的255次访问都命中，因此，第一次访问，命中次数为255*8=2040；后面的4次重复访问，因相关块都已装入cache，都命中，其命中次数=2048*4=8192。

即访问cache的次数为：2040+8192=10232

cache访问的命中率=10232/(2048*5)=99.92%

命中率：

$$h = \frac{N_c}{N_c + N_m} = \frac{10232}{2048 \times 5} = 99.92\%$$

Cache/主存系统的效率：

$$e = \frac{1}{h + (1-h)r} = \frac{1}{0.9992 + (1-0.9992) \times 8} \approx 99.44\%$$

平均访问时间：

$$t_a = \frac{t_c}{e} = \frac{50\ ns}{0.9944} \approx 50.3\ ns$$

设置

【练】某计算机的 cache 由 256个存储块构成，主存包含 4K 个存储块，每块由 32个字组成，采用全相联映射方式，访问地址为**字地址**。

（1）列出主存地址的划分情况，并标出各部分的位数。

主存块地址（标记）字段位数： [填空1]

块内偏移字段位数： [填空2]

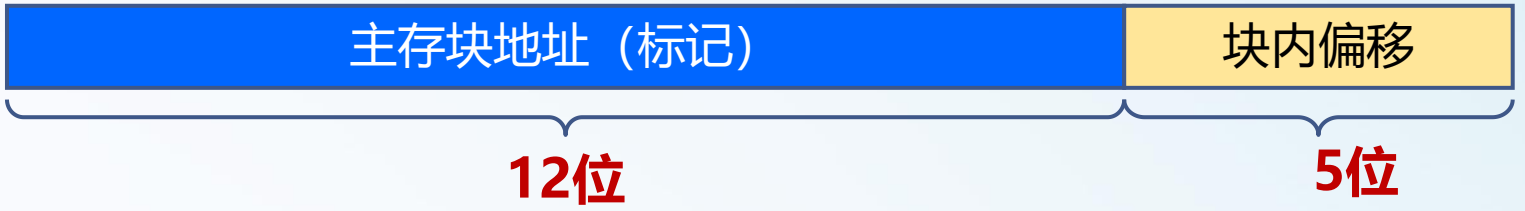
（2）设cache为空，CPU从0单元开始，依次访问主存256个字在（设访问主存一次读出一个字），重复10次，试计算cache访问的命中率。

命中率为（填写方式 XX.X%）： [填空3]

作答

【解】：（1）存储块每块32字，即2^5字，块内偏移字段位数需要5位。

主存有4K块，即2^{12}块，主存块地址12位；即主存块地址（标记）字段位数为12位。



（2）存储块每块32字，而CPU要访问主存单元从0开始的256个字。对应的主存块号为：0,1,2，…，6,7共8块，每块32个字。因cache初始时为空，访问0单元时，需装入主存0块数据，接下来访问1单元时，因已装入cache，命中；同理，…，访问31单元也命中，即访问主存0块时，命中次数为31次。访问其它块时，除了第一次需要装入外，其它的31次访问都命中，因此，第一次访问，命中次数为31*8=248；后面的9次重复访问，因相关块都已装入cache，都命中，

cache访问的命中率=(248+9*256)/(256*10)=99.6875%

【例】某计算机字长32位，采用直接映射Cache,主存容量4MB，Cache数据存储体容量为4KB，块长度为8个字。

（1）画出直接相联映射方式下主存字节地址划分情况，并说明每个字段位数。

（2）设Cache初始状态为空，若CPU顺序访问0-99号单元，并从中读出100个字，假设主存一次读一个字，并重复此顺序10次，请计算Cache命中率。

（3）如果Cache的存取时间是20ns，主存访问时间是200ns，根据（2）中计算出的命中率求存储系统的平均访问时间。

（4）计算Cache-主存系统访问效率。

【解】 （1） 直接相联映射方式下主存地址

由题意，一个Cache块为8个字，机器字长为32位（4字节），故块大小为32B(2^5B)，故w=5；

Cache存储体的行数=Cache存储体容量/块大小（块长度）

=4KB/32B=128行（2^7行），可得 r=7；

主存容量为4MB=2^{22}B，主存按Cache大小分区的数量为=4MB/4KB=2^{10}，可得，区地址的位数为10位。

或：区地址的位数=22-r-w=22-7-5=10位

（2） 由（1）得到的结果，Cache分为128行，每行（块）8个字。Cache初始状态为空。主存从0到99号单元的100个字，将依次载入Cache前13行中（最后一行只载入4个字：96--99）。

第一次访问，每个数据块的第一次读访问都没有命中，会将对应数据块载入，后续相邻的7次访问都会命中。其命中次数=**100-13=87**；

第二次循环访问开始，都全部命中，即后续的9次循环都命中，命中次数=**100×9=900**

故命中率h=(900+87）/（100×10）=98.7%

(3) 平均访问时间:

$$t_a = ht_c + (1-h)t_m = 0.987 \times 20ns + (1-0.987) \times 200ns = 22.34ns$$

(4) Cache/主存系统的效率:

$$e = \frac{1}{h + (1-h)r} = \frac{1}{0.987 + (1-0.987) \times 10} \approx 89.5\%$$

或

$$e = \frac{t_c}{t_a} = \frac{20\ ns}{22\ .34\ ns} \approx 89\ .5\%$$

【例】主存地址空间大小为256MB，按字节编址。指令数据Cache，均有8行，Cache行数据块大小为64B，数据Cache采用直接相联映射方式。现有两功能相同的程序A，B，其伪代码如下所示：

程序A

```
int a[256][256];
int sum_array1()  {
for (i = 0; i < 256; i++)
 for (j = 0; j < 256; j++)
   sum += a[i][j];      }
```

程序B

```
int a[256][256];
int sum_array1()  {
for (j = 0; j < 256; j++)
  for (i = 0; i < 256; i++)
    sum += a[i][j];     }
```

假定int型数据为32位补码，程序编译时i，j，sum均分配在寄存器中，数组a按行优先方式存放，首地址为320（十进制）。

（1）若不考虑用于Cache一致性维护和替换算法的控制位，数据cache的总容量是多少?

（2）数组元素a[0][31],a[1][1]所在主存块对应的cache行分别是多少，行号从零开始。

（3）程序A，B的数据访问命中率各是多少？那个程序的执行时间更短?

【解】

（1） Cache总容量=Cache行数×Cache行大小

不考虑一致性维护和替换算法的控制位，每个Cache行主要包括3部分：

有效位valid、区地址标记字段（tag）、数据块。

主存256MB=2^{28}B，即地址位宽28位；Cache有8行，则r=3位；每行块大小为64B=2^{6}B，即块内偏移地址位宽w=6位；由此可得，区地址（tag）的位宽为：28-3-6=19位。

因此，Cache总容量=8×（1+19+64×8）=**4256位**（bit）=**532B**（字节）

（2） 数组大小为a[256][256]，按**行优先方式**存放，首地址为320，数组元素大小占4个字节（32位）。

内存地址	320	324	328	.....	1340	1344	1348	...
行优先	a[0][0]	a[0][1]	a[0][2]	.....	a[0][255]	a[1][0]	a[1][1]	...
列优先	a[0][0]	a[1][0]	a[2][0]	.....	a[255][0]	a[0][1]	a[1][1]	...

数组a[0][31]所在的主存地址为： 320+31×4=444；

数组a[1][1]所在的主存地址为： 320+256×4+1×4=1348；

按照直接相联映射规则： **Cache行号=主存块号 mod Cache行数**

a[0][31]所在的主存块对应的Cache行号=[444/64] mod 8=6；

a[1][1]所在的主存块对应的Cache行号=[1348/64] mod 8=5。

（3） 由题意，程序编译时i，j，sum均分配在寄存器中，数组a按行优先方式存放。因此，分析命中率时，只考虑数组a的情况。程序A和B功能都是实现二维数组的累加求和，数组中的每一个元素仅被使用一次。

数据Cache的容量为8×64B=512B=128字（1字=4B=32位），可以放下数组半行的数据。

程序A中数据的访问顺序与存储顺序相同，具有较好的空间局部性，每个Cache数据块可以存放16个int数据，顺序访问时，第一次访问缺失，载入数据块，后续15次访问都会命中。程序A中的所有数据访问都符合这一规律，故命中率为15/16，即程序A的命中率h=93.75%。

（3） 程序B内循环访问时，将连续访问不同行的同一列数据，由于数组中一行数据大小是256字，是Cache容量的2倍，因此不同行的同一列数组元素对应同一个Cache行，第一次访问不命中，载入数据块，其后续访问仍然不命中，载入新的数据块到同一行。即所有的数据都无法命中，故命中率=0。

由此可知，程序A执行速度比程序B要快得多。

【例】假设某程序访问7块信息，cache分为4行，采用全相联方式组织。程序访问的块地址流依次为1，2，3，2，1，3，1，4，4，5，6，7，5，6，7，5。分析LRU算法的访问过程，并计算命中率。

地址流	1	2	3	2	1	3	1	4	4	5	6	7	5	6	7	5
第0行	1	1	1	1	**1**	1	**1**	1	1	1	1	**7**	7	7	**7**	7
第1行		2	2	**2**	2	2	2	2	2	**5**	5	5	**5**	5	5	**5**
第2行			3	3	3	**3**	3	3	3	3	**6**	6	6	**6**	6	6
第3行								4	**4**	4	4	4	4	4	4	4
命中情况	失	失	失	√	√	√	√	失	√	替	替	替	√	√	√	√

命中率=9/16=56.25%

【思考】沿用上例的数据，某程序访问7块信息，cache分为4行。程序访问的块地址流依次为：**1，2，3，2，1，3，1，4，4，5，6，7，5，6，7，5**。采用下列两种映射方式时，试分析LRU算法的访问过程，并计算对应的命中率。

（1）直接相联映射；

（2）2路组相联映射。

【解】（1）直接相联映射，cache行与主存块的对应关系为：

0行--主存0，4块；1行--1，5块；2行--2，6块；3行--3，7块

地址流	1	2	3	2	1	3	1	4	4	5	6	7	5	6	7	5
第0行								4	4	4	4	4	4	4	4	4
第1行	1	1	1	1	1	1	1	1	1	5	5	5	5	5	5	5
第2行		2	2	2	2	2	2	2	2	2	6	6	6	6	6	6
第3行			3	3	3	3	3	3	3	3	3	7	7	7	7	7
命中情况	失	失	失	√	√	√	√	失	√	替	替	替	√	√	√	√

命中率=9/16=56.25%

【解】（1）2路组相联映射，每组2行，cache分为2组，行与主存块的对应关系为： 0组（0,1行--主存0,2,4,6块），1组（2,3行--主存1,3,5,7块）

地址流		1	2	3	2	1	3	1	4	4	5	6	7	5	6	7	5
0组	第0行		2	2	**2**	2	2	2	2	2	2	**6**	6	6	**6**	6	6
0组	第1行								4	**4**	4	4	4	4	4	4	4
1组	第2行	1	1	1	1	**1**	1	**1**	1	1	1	1	**7**	7	7	**7**	7
1组	第3行			3	3	3	**3**	3	3	3	**5**	5	5	**5**	5	5	**5**
命中情况		失	失	失	√	√	√	√	失	√	替	替	替	√	√	√	√

命中率=9/16=56.25%

【练习】某计算机的 cache 由 1K个存储块构成，主存包含 64K 个存储块，每块由 256 个字组成，访问地址为字地址。若程序要访问下列地址单元的数据，请给出不同映射方式下，cache的相应标志（即载入cache哪一行/组，对应tag是多少，要求用十六进制表示）。设cache为空，访问对应地址单元时从主存载入数据到cache。

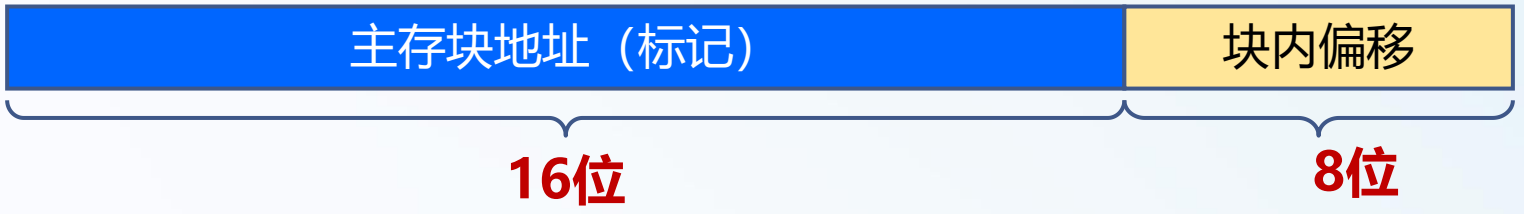
主存地址单元：000000H， 240840H， FFFFF8H

（1）全相联映射方式；

（2）直接相联映射方式；

（3）8路组相联映射方式。

【解】：（1）**全相联映射方式**：存储块每块256字，即2^8字，块内偏移需要8位。主存有64K块，即2^{16}块，主存块地址16位；



主存地址单元：000000H， 240840H， FFFFF8H

载入后cache相应标志（十六进制表示）

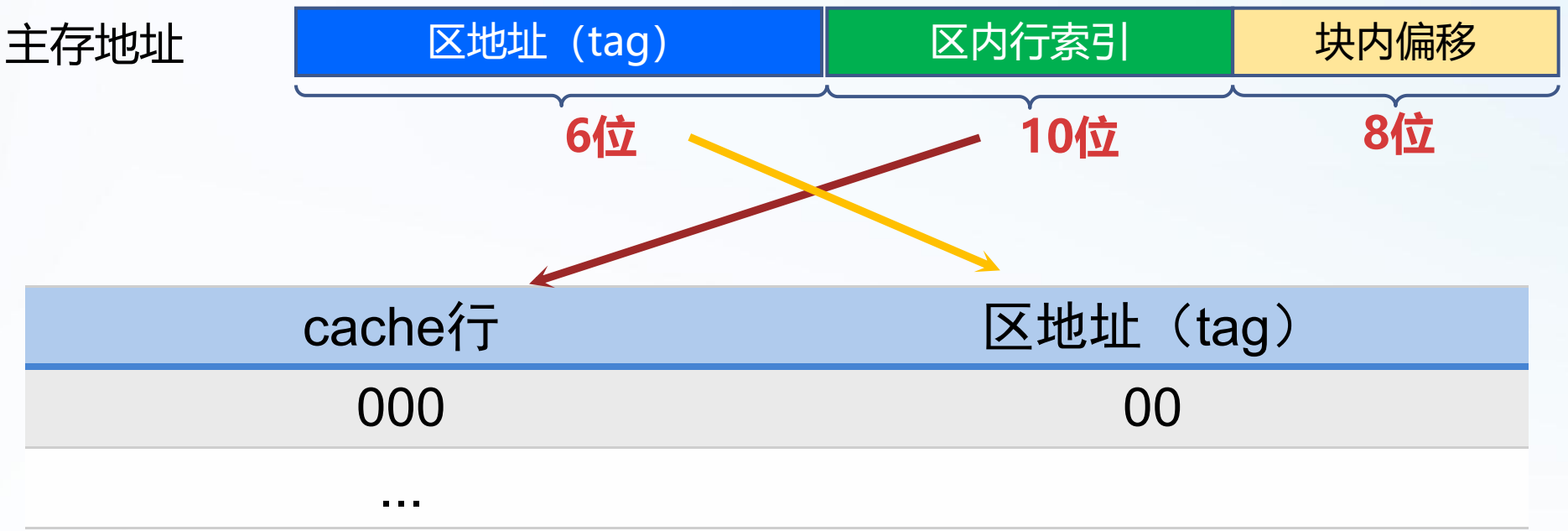
cache行	主存块地址（标记）
000	0000
001	2408
002	FFFF

【解】：（2）**直接相联映射方式**：

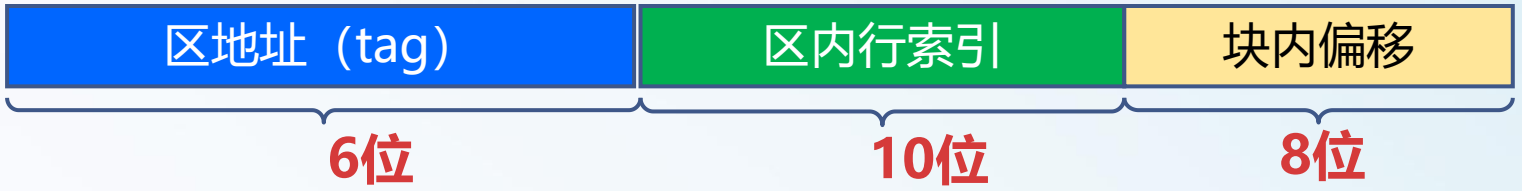
存储块每块256字，即2^8字，块内偏移需要8位。即 w=8位。

Cache有1k块，即2^{10}块（行），即区内行索引字段位数r=10位。

主存有64K块，即2^{16}块， s=16位。 区地址字段位数=s-r=16-10=6位。



主存地址



主存地址单元：000000H， 240840H， FFFFF8H

240840H→**0010,0100,0000,1000,0100,0000**

FFFFF8H→**1111,1111,1111,1111,1111,1000**

载入后cache相应标志（十六进制表示）

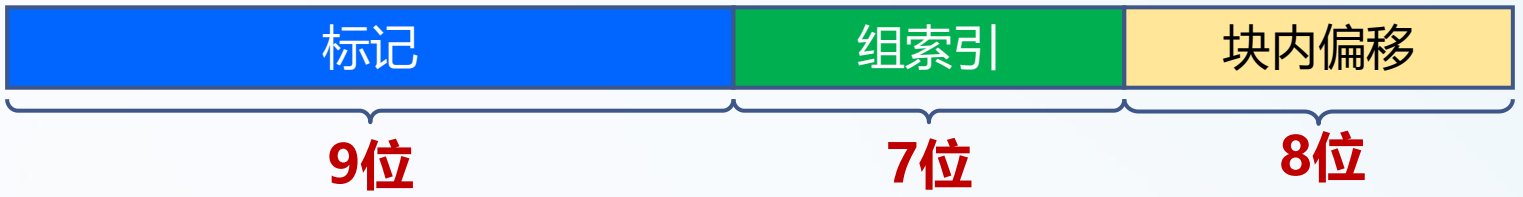
cache行	区地址（tag）
000	00
008	09
3FF	3F

【解】：（3）**组相联映射方式**：

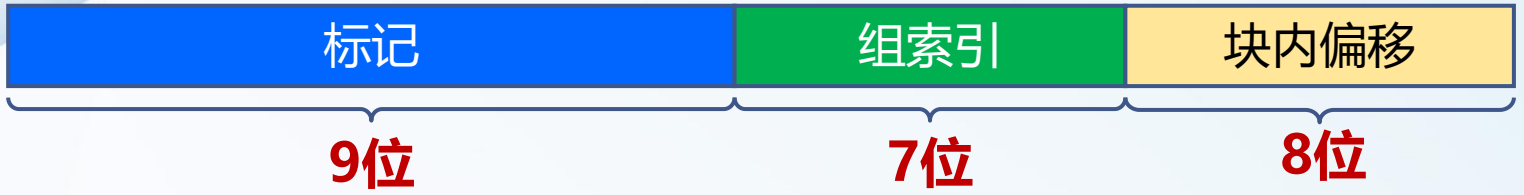
Cache有1k块，由题干条件，采用8路组相联映射。

组索引字段位数d=7位。 主存有64K块，即2^{16}块， s=16位。

标记（tag）地址字段位数=s-d=16-7=9位。



主存地址	cache组	标记（tag）
000000	00	000
240840	08	048
FFFFF8	7F	1FF



主存地址单元：000000H， 240840H， FFFFF8H

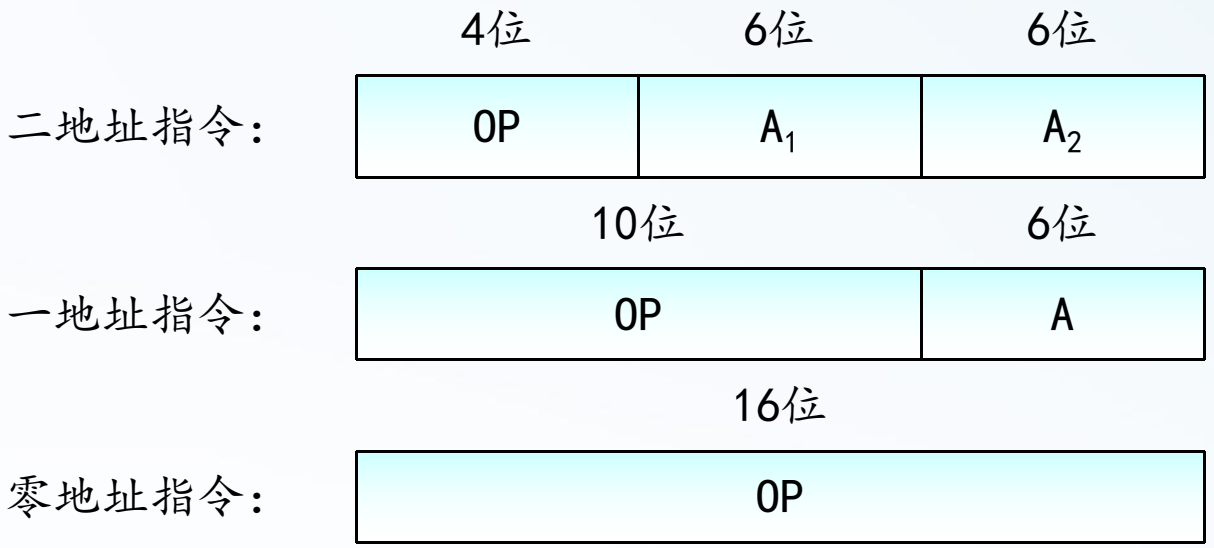
240840H→**0010,0100,0000,1000,0100,0000**

FFFFF8H→**1111,1111,1111,1111,1111,1000**

载入后cache相应标志（十六进制表示）

主存地址	cache组	标记（tag）
000000	00	000
240840	08	048
FFFFF8	7F	1FF

【例】 设某指令系统，有三类指令，如图所示，假设二地址指令有15条，一地址指令62条，则零地址指令最多有多少条？整个指令系统可以有多少条指令？

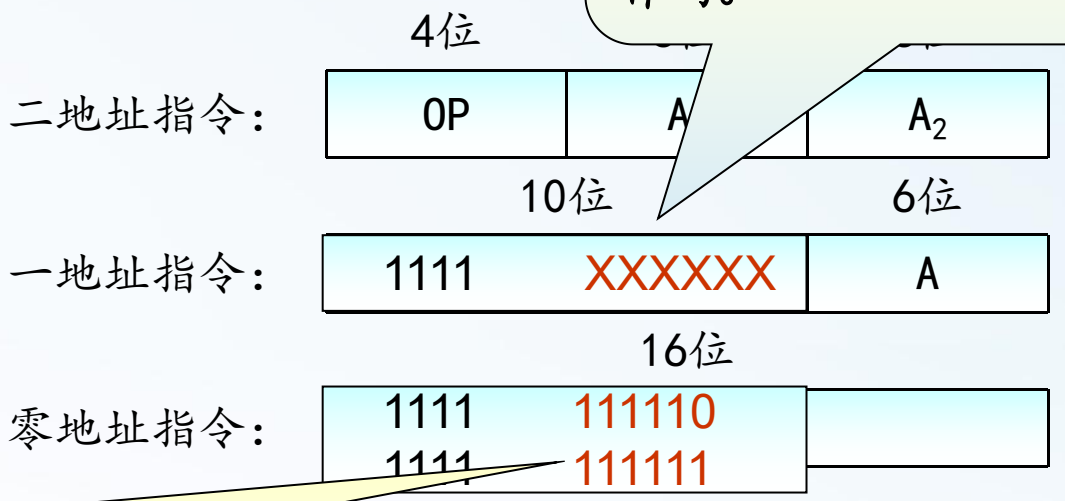


【分析】 二地址指令有15条，则其未使用的编码

设没使用的编码为1111，这个编码作为一地址指

由于A_1字段的位数是6位，因此用一个扩展标志1111就可以扩展出2^6=64种一地址指令的操作码。

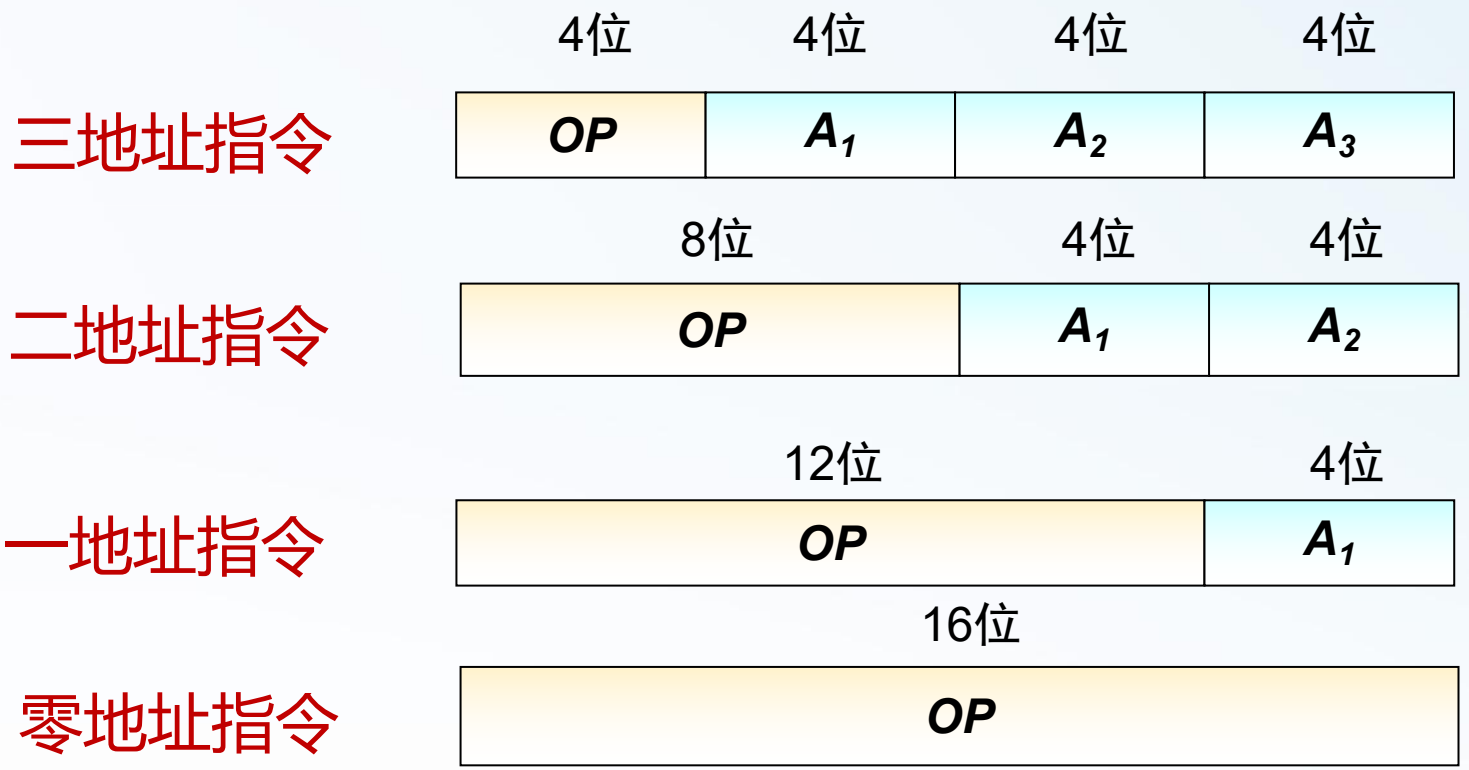
由于A字段的位数是6位，因此用两个标志位可以扩展出2×2^6=128种零地址指令的操作码。



如果机器只需要62条一地址指令，则余下的两个编码（1111111110、1111111111）都可以作为零地址指令操作码的扩展标志，扩展到一地址指令的A字段，就形成了零地址指令。

令。

【例】 设某指令系统，指令字长为16位，每个地址码为4位。有四类指令，如图所示，三地址指令有15条，二地址指令有14条，一地址指令22条，则该指令系统最多可以设计零地址指令多少条?



【解】 由题意，可得该指令系统的扩展码编码方案为：4-8-12-16。

三地址指令15条，则二地址指令可用的扩展标志（剩余状态）数为：

$2^4-15=1$

二地址指令14条，则一地址指令可用的扩展标志数为：$1\times2^4-14=2$

一地址指令22条，则零地址指令可用的扩展标志数为：$2\times2^4-22=10$

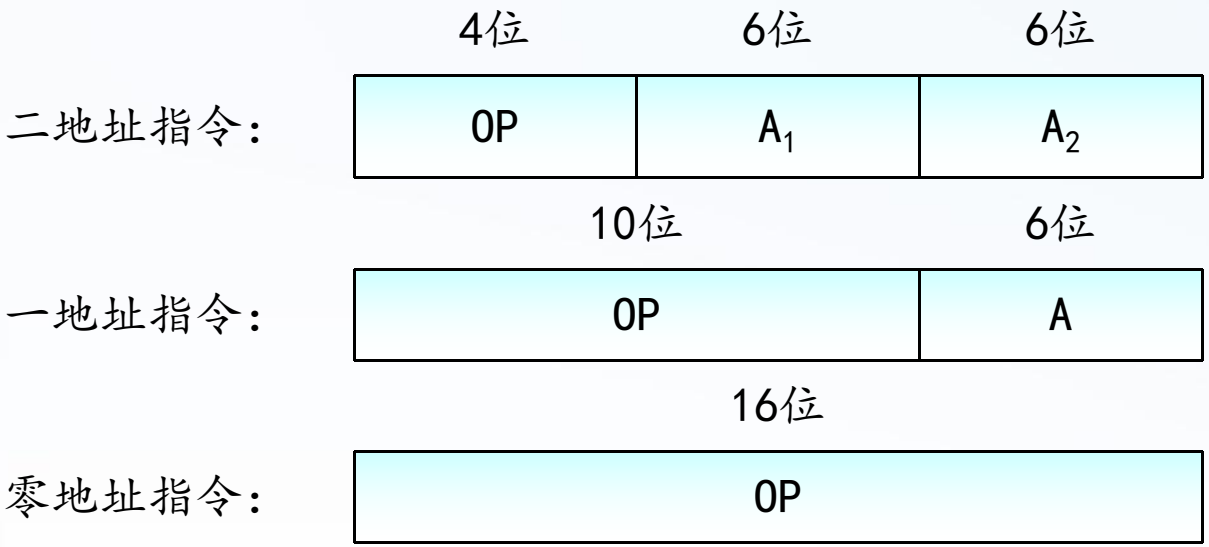
零地址最多可以编码数量为：$10\times2^4=160$

填空题 10分

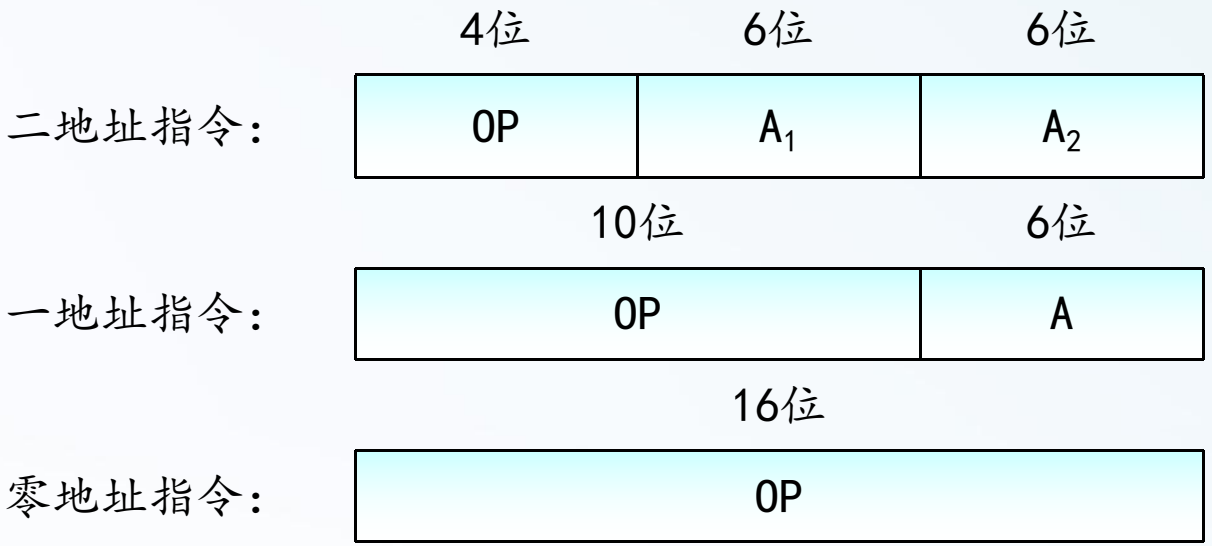
【思考】 指令格式如图所示，若指令系统要求设计180条一地址指令，则二地址指令最多可以有多少条？此时零地址指令最多可以有多少？

【答】二地址指令最多： [填空1] 条，

零地址指令最多： [填空2] 条。



【思考】 若指令系统要求设计180条一地址指令，则二地址指令最多可以有多少条？此时零地址指令最多可以有多少？



若一地址指令180条，则需要3个扩展标志，则二地址指令最多可以有$2^4-3=13$条，零地址指令最多可以有 $(3*2^6-180)*2^6=768$ 条

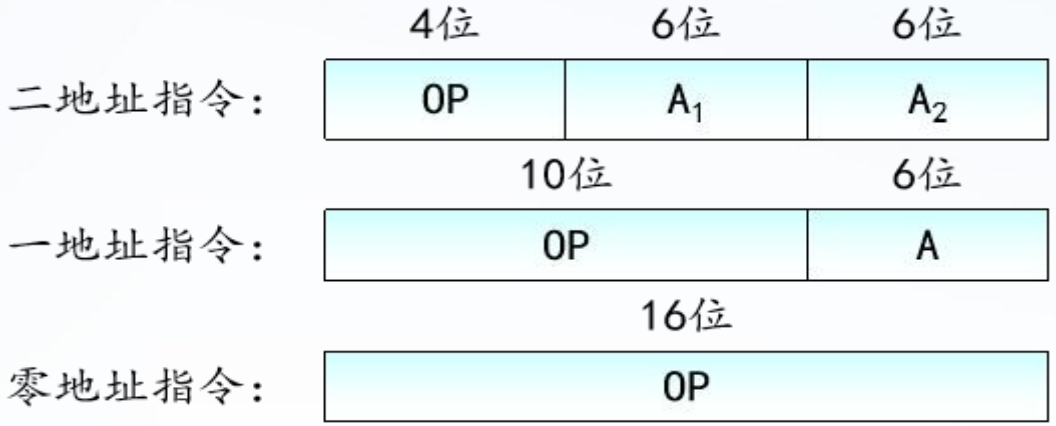
填空题 15分

【思考与练习】设某指令系统，指令字长为16位，有三类指令，分别为二地址指令，一地址指令和零地址指令，指令格式如图所示。地址码为6位，操作码采用可变长扩展码，编码方案为4-10-16。已知该指令系统中，一地址指令125条，试完成如下内容：

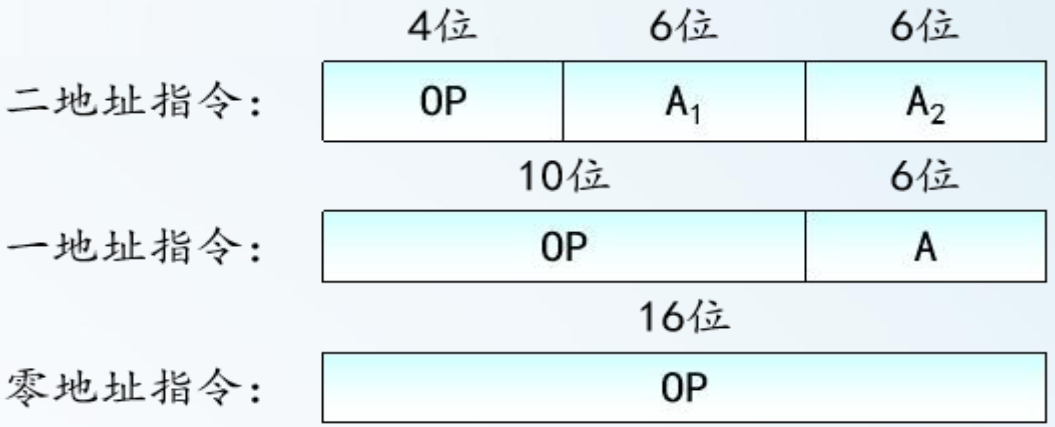
（1）二地址指令最多有多少条？ [填空1]

（2）在此基础上，零地址指令最多有多少条？ [填空2]

（3）该指令系统最多有多少条指令？ [填空3]



【解】（1）指令格式

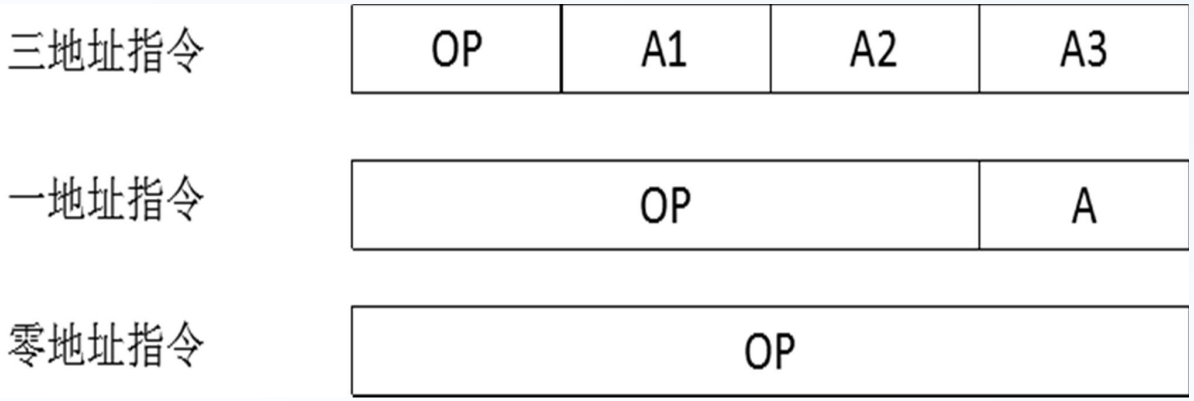


一地址指令有125条，因为其实际编码位数为6位，要满足125个编码要求，其需要2个扩展标志。故二地址最多有：$2^4-2=14$条。

（2） 由上面的分析可知，一地址指令留给零地址指令的扩展标志为：$2\times2^6-125=3$个。故零地址指令最多有：$3\times2^6=192$条。

（3）该指令系统最多有： $14+125+192=331$条指令

【思考与练习】某机器指令字长为12位，其指令形式有三种：三地址指令，单地址指令和零地址指令，指令格式如图所示。已知三地址指令有4条，单地址指令有255条，零地址指令有16条。若每个地址的码长均为3位，能否以扩展操作码为其编码？如果把单地址指令改为254，能否完成编码？（提示：先求出各指令OP长度）



【解】： 根据已知条件，可以知道，该指令系统中，操作码长度分别为：3（三地址指令）、9（单地址指令）和12（零地址指令）。即编码方案为：3-9-12。

因为三地址指令有4条，故可以有4个扩展标志（码）给单地址指令；零地址指令有16条，需要的扩展标志为 2个（即 $2\times2^3=16$）。而单地址指令有255条，剩余的标志只有1个（$4\times2^6-255=1$），不能满足要求，所以不能满足要求。

若改为254条，剩余标志正好有2个，则正好满足编码要求，可以完成编码。

【例】设某机的指令字长16位，格式、有关寄存器和主存内容如下，MOD为寻址方式，D为形式地址，请在下表中填入有效地址EA及操作数S的值。

OP	MOD	D=100

PC=1000	$R_{变}$=2000	$R_{基}$=400

地址	主存
100	200
200	500
500	800
1100	400
1102	350
2100	600

寻址方式MOD	有效地址EA	操作数S的值
立即寻址	S=D	100
直接寻址	EA=D=100	200
间接寻址	EA=（D）=200	500
相对寻址	EA=PC+D=1100	400
变址寻址	$EA=R_{变}+D=2100$	600
基址寻址	$EA=R_{基}+D=500$	800

填空题 15分

【例】存储器中相应地址（16位）及其存放的内容如表所示，已知寄存器R的值为3000H，PC的值为7000H，变址寄存器R_X的值为2500H，基址寄存器R_B的值为3500H，D为形式地址（16位）。

地址	1000H	2000H	3000H	3A00H	4000H	5000H	6000H	7000H	7100H
数据	3000H	8000H	5204H	9000H	6600H	2019H	3800H	1177H	3502H

试分析，在下列给出的寻址方式中，指令访问得到操作数S的值是多少？

（1）寄存器直接寻址，R；　　[填空1] H

（2）寄存器间接寻址，(R)；　　[填空2] H

（3）直接寻址，D=5000H；　　[填空3] H

（4）基址寻址，D=500H；　　[填空4] H

（5）变址寻址，D=3B00H。　　[填空5] H

【解】

（1） 寄存器直接寻址：操作数S就在R中，即S的值为3000H；

（2） 寄存器间接寻址：EA=(R)=3000H， ∴S=（EA）=5204H；

（3） 直接寻址： EA=D=5000H， ∴S=（EA）=2019H；

（4） 基址寻址： EA=（R_B）+D=3500H+500H=3A00H，

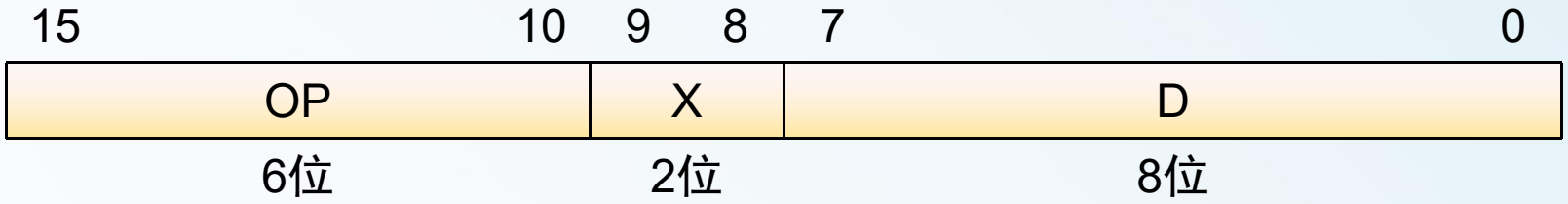
∴ S=（EA）=9000H；

（5） 变址寻址： EA=（R_X）+D=2500H+3B00H=6000H，

∴ S=（EA）=3800H；

设置

【练习】已知机器字长16位，指令格式如下所示：



格式中 D为形式地址，X为寻址方式特征值：

X=00，直接寻址；　　X=01，用变址寄存器R_I进行变址；

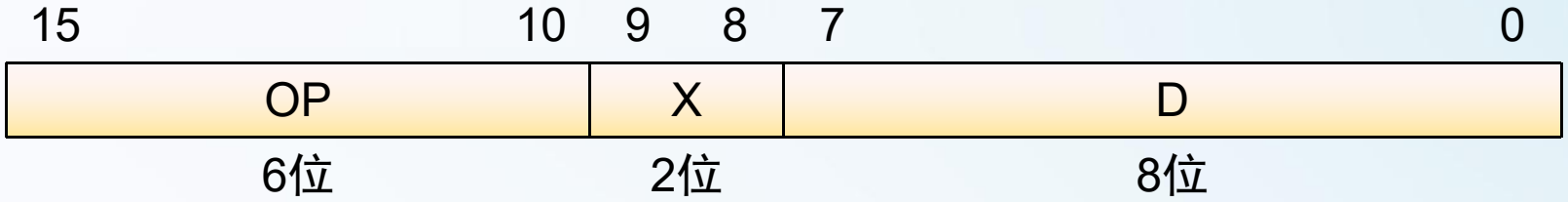
X=11，相对寻址；　　X=10，用基址寄存器R_B进行寻址。

设（PC）=2000H，（R_I）=0150H，(R_B)=1889H，请确定如下指令的有效地址：（设EA位数=机器字长）

（1）4420H　(2) 2244H　(3) 730AH　(4) 3566H　(5) 6783H

答：(1) [填空1]；(2) [填空2]；(3) [填空3]；(4) [填空4]；(5) [填空5]

作答



【解】：

1）X=00 ， D=20H ， 有效地址EA=20H

2）X=10 ， D=44H ， 有效地址EA=1889H+44H=18CDH

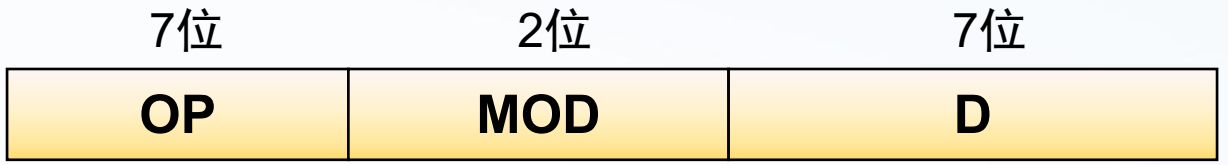
3）X=11 ， D=0AH ， 有效地址EA=2000H+0AH=200AH

4）X=01 ， D=66H ， 有效地址EA=0150H+66H=01B6H

5）X=11 ， D=83H ， 有效地址EA=2000H+FF83H=1F83H

【例】某计算机字长16位，主存64KB，指令采用单字长，单地址结构。要求能提供至少80条指令，支持四种寻址方式：直接、间接、相对、变址寻址。试设计对应指令格式，并指出哪些寻址方式能满足主存寻址要求（即寻址范围能覆盖或达到主存最大地址）。

【解】由题干可得，指令字长16位，操作码OP位数7位，寻址方式特征位（用MOD表示）需要2位，故形式地址D的位数=16-7-2=7位。指令格式如图所示：



主存地址为64KB，则寻址范围要求达到：0---2^{16}-1（即0--65535）

MOD	对应寻址方式	有效地址EA	最大寻址范围	是否满足主存寻址要求
00	相对寻址	EA=PC+D	0---$2^{16}+2^{6}-2$	√
01	变址寻址	EA=R[X]+D	0---$2^{16}+2^{6}-2$	√
10	直接寻址	EA=D	0---127	×
11	间接寻址	EA=(D)	0---2^{16}-1	√

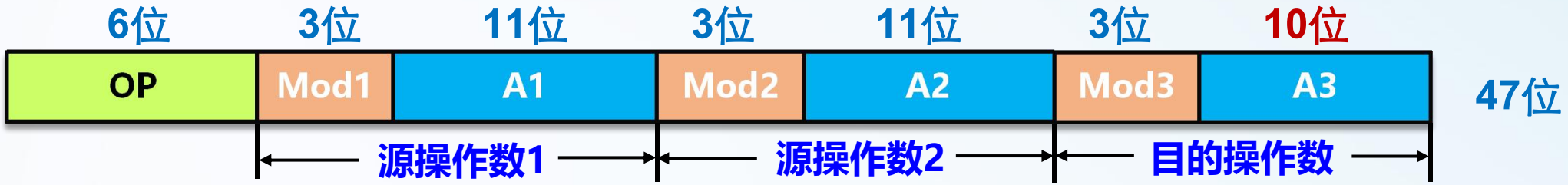
【例】某机字长32位，采用三地址指令（A1 OP A2→A3），每个操作数均支持7种寻址操作（包含直接寻址），完成60种操作，各寻址方式均可在2K主存范围内取得操作数，并可在1K范围内保存运算结果。问:

（1）应采用什么样的指令格式？指令字长最少应为多少位？试设计指令格式并标注每个字段位数。

（2）执行一条全部采用直接寻址方式的指令，最多要访问多少次主存?

（3）若寻址方式中，采用寄存器间接寻址方式，则其最大寻址范围是多少?

【解】（1）由题干可得，60种操作，操作码OP位数至少需要6位，寻址方式特征位（用MOD表示,7种寻址方式）各需要3位，能访问2K主存，对于直接寻址而言，需要11位。1K，则需要10位。



（2）47位指令字需占用2个存储字，取指需访存2次；取源操作数访存2次，写结果1次，即执行一条全部采用直接寻址方式的指令，防存次数共5次。

（3）若采用寄存器间接寻址方式，其最大寻址范围是：0---2^{32} -1。

【例】已知机器字长16位，分析以下指令格式及寻址方式特点，其中Ms、Md分别为源地址、目的地址寻址方式特征位，Rs、Rd表示源地址、目的地址寄存器。

【解】 (1) 属于单字长二地址指令；

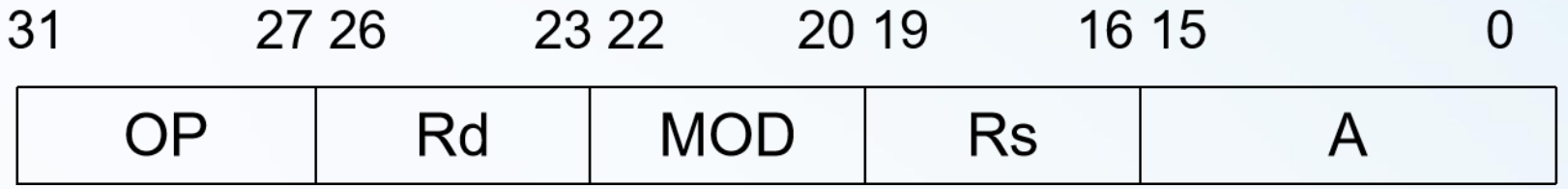
(2) 操作码有5位，最多可以指定32条指令；

(3) 源地址支持4种寻址方式，目的地址最多支持8种寻址方式；

(4) 源地址寄存器和目的地址寄存器均最多有8个。

填空题 15分

【课堂练习】已知机器字长32位，指令格式如图所示，为单字长双地址RS型指令。Rd、Rs为通用寄存器，目标操作数采用寄存器寻址方式，MOD表示源操作数寻址方式，A为形式地址。



试分析：

（1）该指令模型中最多有多少条指令？ [填空1]

（2）通用寄存器有多少个？ [填空2]

（3）能达到最大的寻址空间是多少？0--- [填空3]

【例】 设有一台计算机，其指令长度为16位，有一类RS型指令的格式：

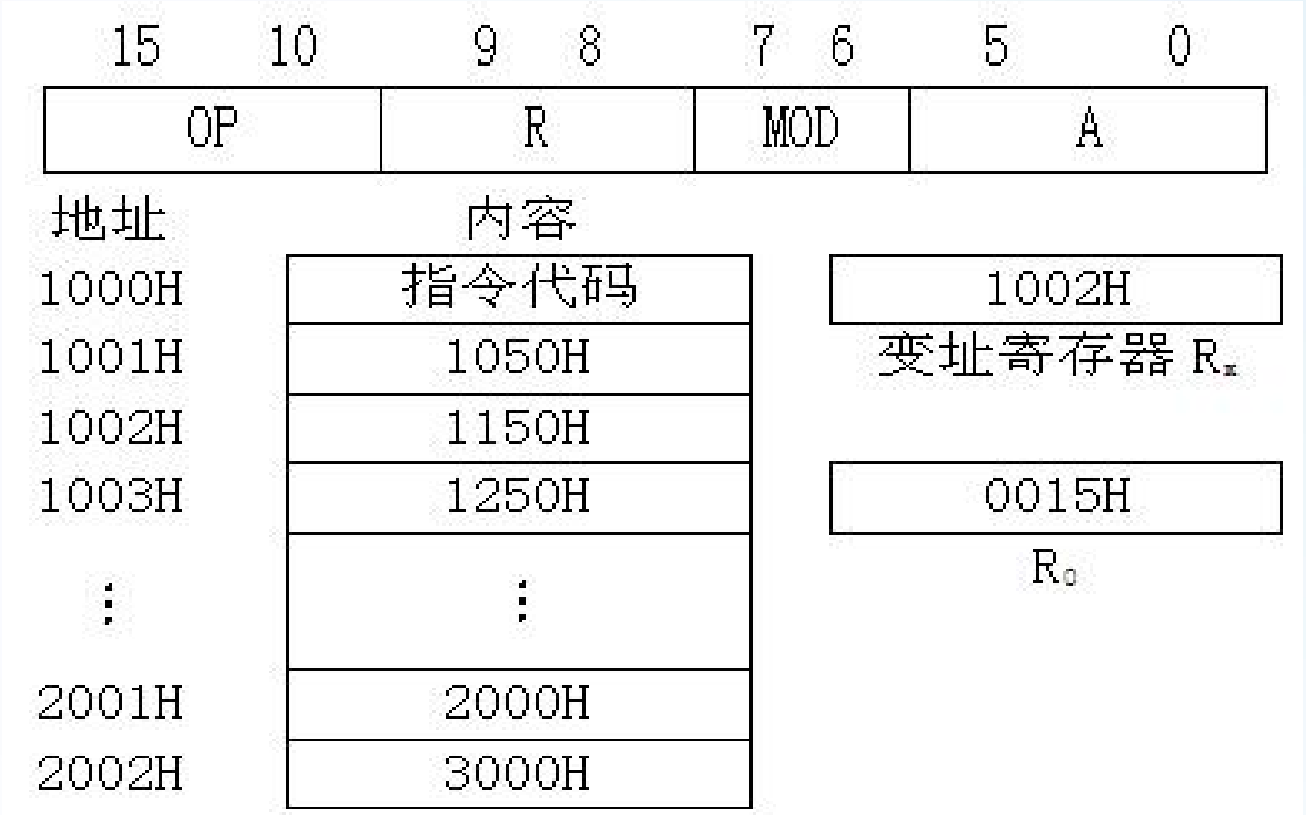
其中，OP为操作码，占6位；R为寄存器编号，占2位，可访问4个不同的通用寄存器；MOD为寻址方式，占2位，与形式地址A一起决定源操作数，规定如下：

MOD=00，为立即寻址，A为立即数；

MOD=01，为相对寻址，A为位移量；

MOD=10，为变址寻址，A为位移量。

假定要执行的指令为加法指令，存放在1000H单元中，形式地址A的编码为01H，其中H表示十六进制数。该指令执行前存储器和寄存器的存储情况如图所示，假定此加法指令的两个源操作数中一个来自于形式地址A或者主存，另一个来自于目的寄存器R_0，并且加法的结果一定存放在目的寄存器R_0中。



在以下几种情况下，该指令执行后，R0和PC的内容为多少？

(1) 若MOD=00，(R0)=_____；

(2) 若MOD=01，(R0)=_____；

(3) 若MOD=10，(R0)=_____；(PC)=_____。

【解】：(1) 若MOD=00，为立即寻址，则指令格式中的形式地址部分即为立即数，因此一个源操作数为01H，另一个源操作数为R0的内容0015H，加法指令执行的结果为(R0)= 0016H 。

(2) 若MOD=01，为相对寻址，则一个源操作数的有效地址EA=(PC)+A，在执行加法指令时，PC的值为下一条指令的地址，即(PC)=1001H，由此可以算出这个源操作数的有效地址为EA=1001H+01H=1002H，这个操作数为(E)=1150H，另一个源操作数为R_0的内容0015H，加法指令执行的结果为(R_0)= 1165H 。

(3) 若MOD=10，为变址寻址，则一个源操作数的有效地址EA=(R_x)+A，由此可以算出这个源操作数的有效地址为EA=1002H+01H=1003H，这个操作数为(E)=1250H，另一个源操作数为R0的内容0015H，加法指令执行的结果为(R_0)= 1265H ；在执行加法指令时，PC的值为下一条指令的地址，即(PC)= 1001H 。

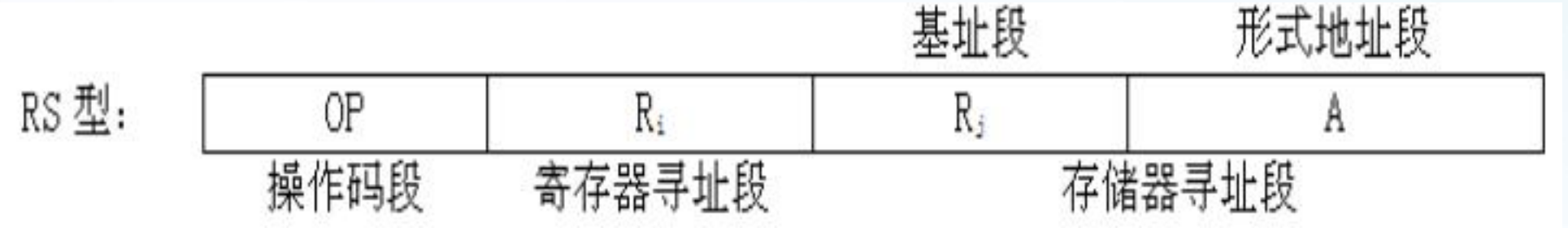
【例】 已知某计算机系统中有一类RS型指令，其指令格式如图所示：

该指令长度为16位，已知CPU中有8个16位长的通用寄存器，这些寄存器也可作为基址寄存器使用，若要构造16条RS型指令，问：

(1) 该类指令各段占用多少位?

(2) 能寻址的最大主存地址为多少?

(3) 若将RS型指令中的操作码段扩展到寄存器寻址段而构成S型指令，问此时RS型指令最多为多少条？在此基础上S型指令最多可以设计多少条?



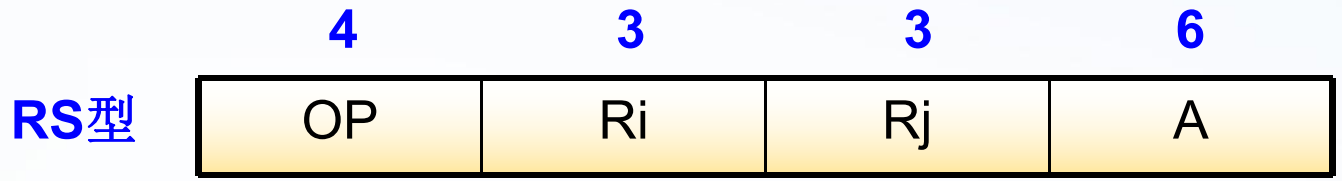
【解】：(1)操作码决定了指令的功能，若要构造16条RS型指令，则操作码段的位数为： [lb16]= 4 （位）

指令格式中的寄存器寻址段实际上是可访问的通用寄存器的编码，因为CPU中有8个16位长的通用寄存器，所以寄存器寻址段的位数为：

[lb8]= 3（位）

形式地址段的位数等于指令的字长减去操作码段、寄存器寻址段和基址段的位数，为6位。

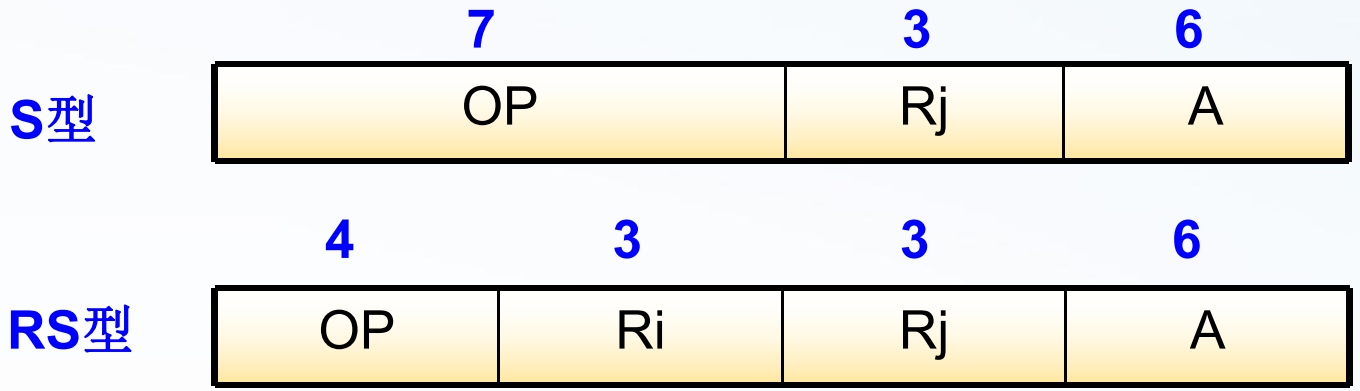
综合以上分析，该类指令各段占用的位数如下所示：



(2) 由于该类指令只有基址寻址访问主存，因此寻址的最大主存地址只由基址寻址决定。基址寻址的$E=(R_j)+A$，位移量A用补码表示，最大值为2^5-1，基址寄存器的最大值为$2^{16}-1$，因此该类指令能寻址的最大主存地址为：

$2^{16}+2^5-2$

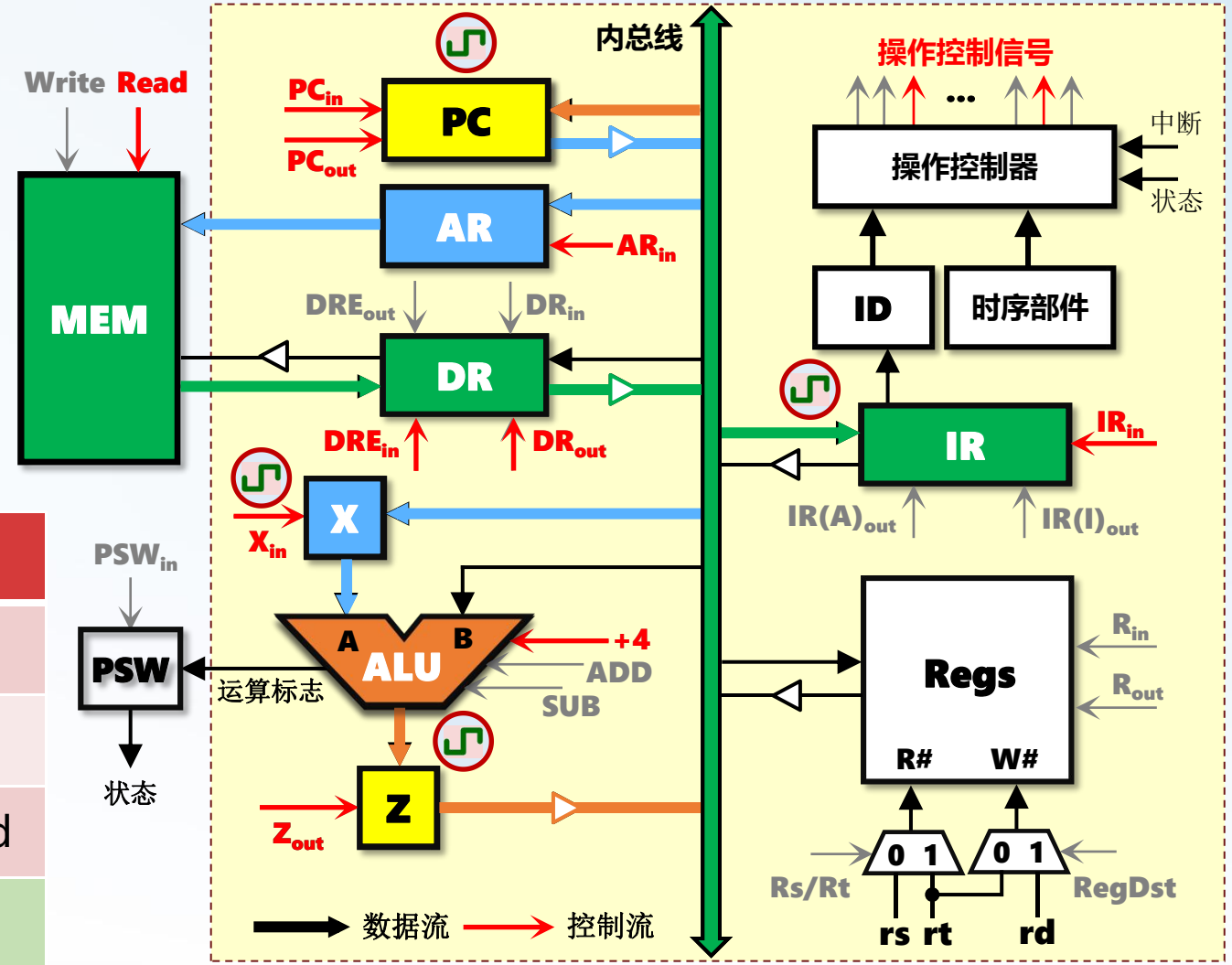
(3) RS型指令的操作码段占4位，最多有16种编码，但至少要留一种编码作为S型指令操作码的扩展标志，因此RS型指令最多为15条；在S型指令操作码只使用一种扩展标志的基础上S型指令最多可以设计**8**条。



取指令数据通路

- M[PC] → IR
- PC+指令长度→ PC
- 指令译码

节拍	数据通路 (数据流)	控制信号(控制流)
T1	PC→AR, PC→X	PC_{out}, AR_{in}, X_{in}
T2	X+4→Z	+4
T3	Z→PC, M[AR]→DR	Z_{out}, PC_{in}, DRE_{in}, Read
T4	**DR→IR**	**DR_{out}, IR_{in}**



1. lw指令的执行流程

指令功能：从主存读取一个32位的存储器字

汇编代码： lw rt, imm (rs)

指令格式：

lw指令执行共需要3个机器周期：

第一个机器周期为**取指周期M_{if}**；

第二个机器周期为**计算周期M_{cal}**，用于计算访存地址;

第三个机器周期为**执行周期M_{ex}**，用于实现存储器读取。

lw指令操作流程及控制信号

周期	节拍	操作	功能说明	控制信号
取指周期 M_{if}	T1	PC→AR; PC→X	将程序计数器PC内容送入AR,同时送入暂存器X	PC_{out}=AR_{in}=X_{in}=1
	T2	X+4→Z	将PC值加4并送入暂存器Z	+4=1
	T3	Z→PC; M[AR]→DR	将暂存器Z内容回送到PC，同时读AR内容对应主存单元的值并送入DR	Z_{out} =PC_{in}=1 Read=DRE_{in}=1
	T4	DR→IR	将DR内容送入IR,完成取指令	DR_{out}=IR_{in}=1
计算周期 M_{cal}	T1	R[rs]→X	将rs寄存器内容送入暂存器X,准备计算访存地址	R_{out}=X_{in}=1
	T2	IR(I)+X→Z	将IR中的立即数符号扩展为32位并送入ALU做加法运算	$IR(I)_{out}$=ADD=1
执行周期 M_{ex}	T1	Z→AR	将Z中暂存的访存地址送入AR	Z_{out}=AR_{in}=1
	T2	M[AR]→DR	读AR内容对应主存单元的值并送入DR	Read=DRE_{in}=1
	T3	DR→R[rt]	将DR内容送入寄存器rt	DR_{out}=R_{in}=1

注意:表中的控制信号仅给出非零值的信号，未给出的信号值为零。

3个机器周期的数据通路

从表中可以看出，lw指令的3个机器周期使用了不同的数据通路。

(1) **取指周期M_{if}**： 使用的两条数据通路。

PC→AR→MEM→DR→IR：以PC为地址访存及取指令并送入指令寄存器IR。

PC→X→ALU→Z→PC：修改PC的值，为取下一条指令做准备。

(2) **计算周期M_{cal}**：

R[rs]→X→ALU；IR(I)→ALU→Z：计算访存地址R[rs]+imm并送入暂存器Z，其中:IR(I)为指令字中的16位立即数符号扩展为32位的数值。

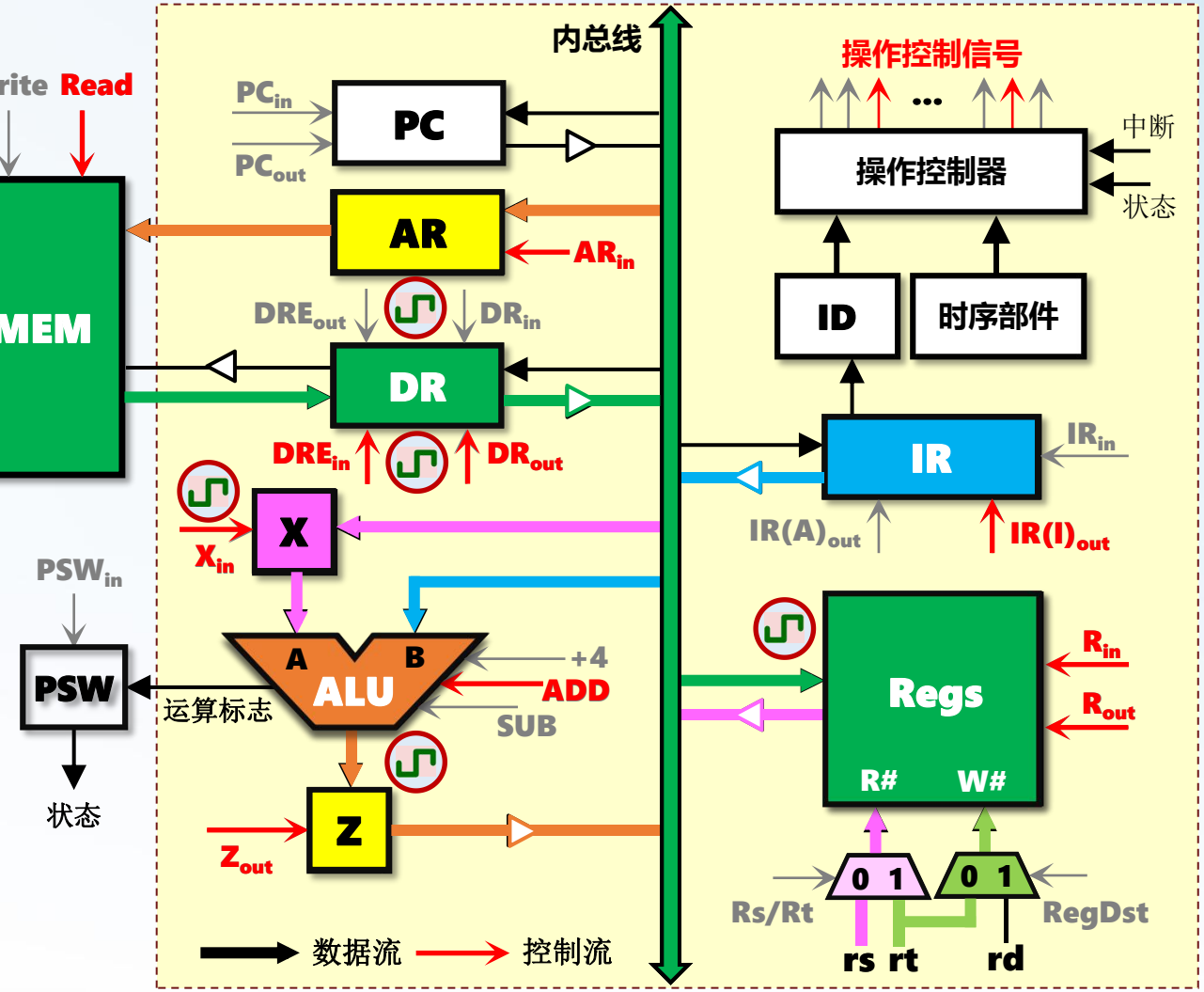
(3) **执行周期M_{ex}**：

Z→AR→MEM→DR →R[rt]：从主存中取32位存储字并送入rt。

lw 指令执行数据通路 *lw rt,imm(rs)*

M[R[rs] + imm] → R[rt]

节拍	数据通路 (数据流)	控制信号(控制流)
T5	R[rs]→X	R_{out}, X_{in}
T6	X+imm→Z	$IR(I)_{out}$,ADD
T7	Z→AR	Z_{out}, AR_{in}
T8	M[AR]→DR	DRE_{in}, Read
T9	**DR→R[rt]**	**DR_{out}, R_{in}**



2. sw指令的执行流程

指令功能：在主存中写入一个32位的存储器字

汇编代码： sw rt, imm (rs)

指令格式：

sw指令执行共需要3个机器周期：

第一个机器周期为**取指周期M_{if}**；

第二个机器周期为**计算周期M_{cal}**，用于计算访存地址;

第三个机器周期为**执行周期M_{ex}**，用于实现存储器写入动作。

sw指令操作流程及控制信号

周期	节拍	操作	功能说明	控制信号
执行周期 Mex	T1	Z→AR	将Z中暂存的访存地址送入AR	Z_{out}=AR_{in}=1
	T2	R[rt]→DR	将rt寄存器内容送入DR	R_{out}= =Rs/Rt=DR_{in}=1
	T3	DR →M[AR]	将DR内容写入AR所指向的主存单元	DRE_{out}=Write=1

注意:sw指令的前两个机器周期使用的数据通路**与lw指令相同**。

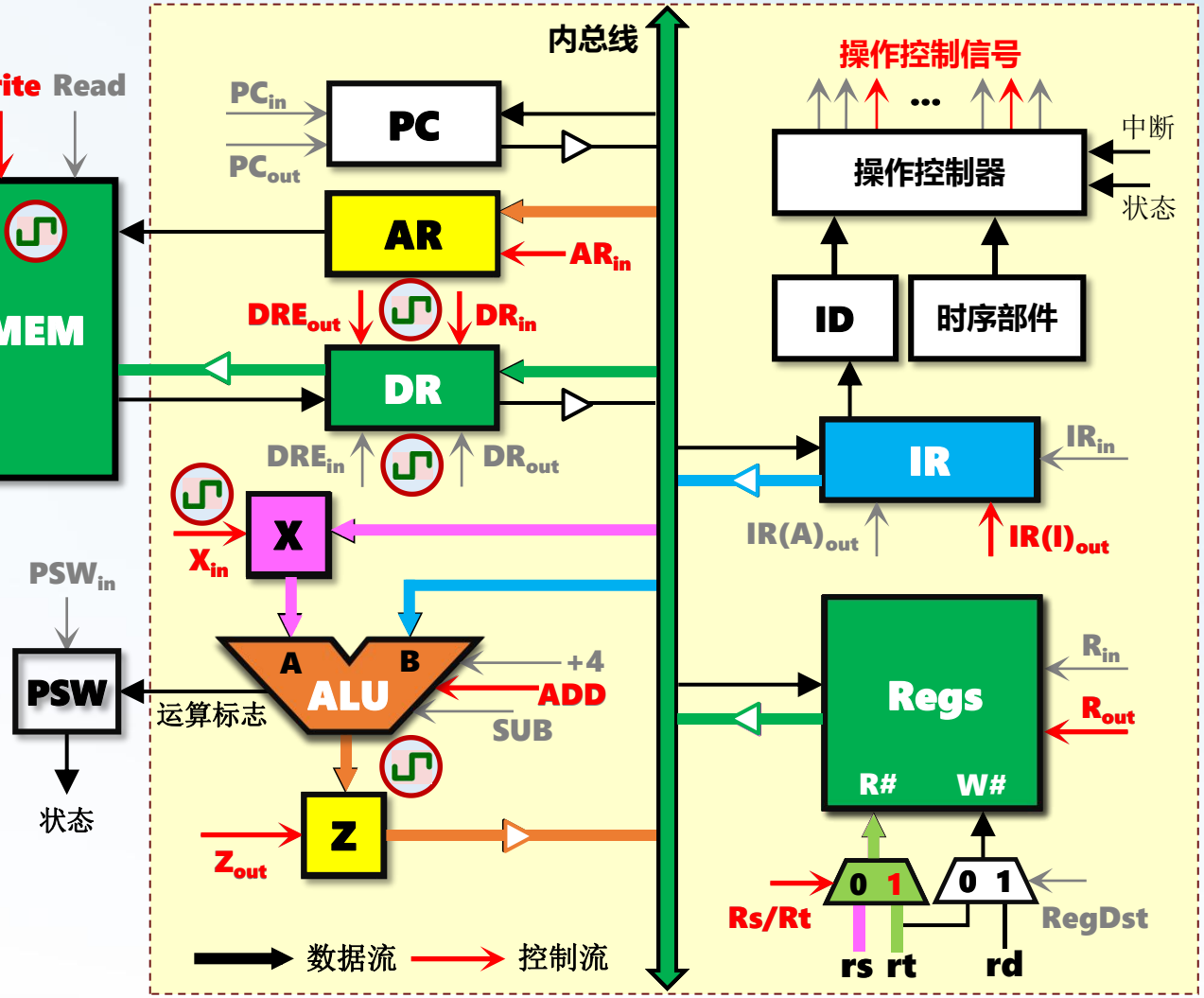
sw执行周期M_{ex}数据通路:

Z→AR； R[rt]→DR→ MEM：将rt内容写入主存单元中。

sw 指令执行数据通路 *sw rt,imm(rs)*

R[rt] → M[R[rs] + imm]

节拍	数据通路 (数据流)	控制信号(控制流)
T5	R[rs]→X	R_{out}, X_{in}
T6	X+imm→Z	$IR(I)_{out}$, ADD
T7	Z→AR	Z_{out}, AR_{in}
T8	R[rt]→DR	R_{out}, Rs/Rt, DR_{in}
T9	**DR→M[AR]**	**DRE_{out}, Write**



3. beq指令的执行流程

指令功能：比较寄存器rs和rt的值，如果相等则进行分支跳转

汇编代码： beq rs,rt,imm

指令格式：

beq指令执行共需要3个机器周期：

第一个机器周期为**取指周期M_{if}**；

第二个机器周期为**计算周期M_{cal}**，用于比较两寄存器的值并产生用于条件分支的标志位;

第三个机器周期为**执行周期M_{ex}**，负责计算分支目标地址，并根据计算周期生成的标志位决定是否进行分支跳转。

beq指令操作流程及控制信号

周期	节拍	操作	功能说明	控制信号
计算周期 M_{cal}	T1	R[rs]→X	将rs寄存器内容送入暂存器X,准备进行比较	$R_{out}=X_{in}=1$
	T2	X-R[rt]→PSW	将rt寄存器内容送入ALU做减法，可产生结果为零的标志位。本书中由ALU自动生成equal标志并送入PSW所以不用关心ALU进行何种运算。	$R_{out}=Rs/Rt=SUB=PSW_{in}=1$
执行周期 M_{ex}	T1	PC→X	将PC内容送入暂存器X	$PC_{out}=X_{in}=1$
	T2	IR(A)+X→Z	将IR中的立即数符号扩展为32位后左移两位送入ALU做加法，计算出分支目标地址	$IR(A)_{out}=ADD=1$
	T3	if (PSW.equal) Z→PC	如果equal标志位为1，将分支目标地址送入PC	$Z_{out}=1$ $PC_{in}=PSW.equal$

beq 3个机器周期的数据通路

(1) **取指周期M_{if}**： 与sw指令一样。

(2) **计算周期M_{cal}**：

R[rs]→X→ALU; R[rt] →ALU→PSW：比较寄存器生成相等标志位送入PSW;**注意**：真实MIPS处理器中是没有PSW寄存器的，在单总线结构中不能同时进行条件判断和分支地址计算，才需要使用PSW寄存器暂存比较结果。

(3) **执行周期M_{ex}**：

PC→X→ALU; IR(A)→ALU→Z：

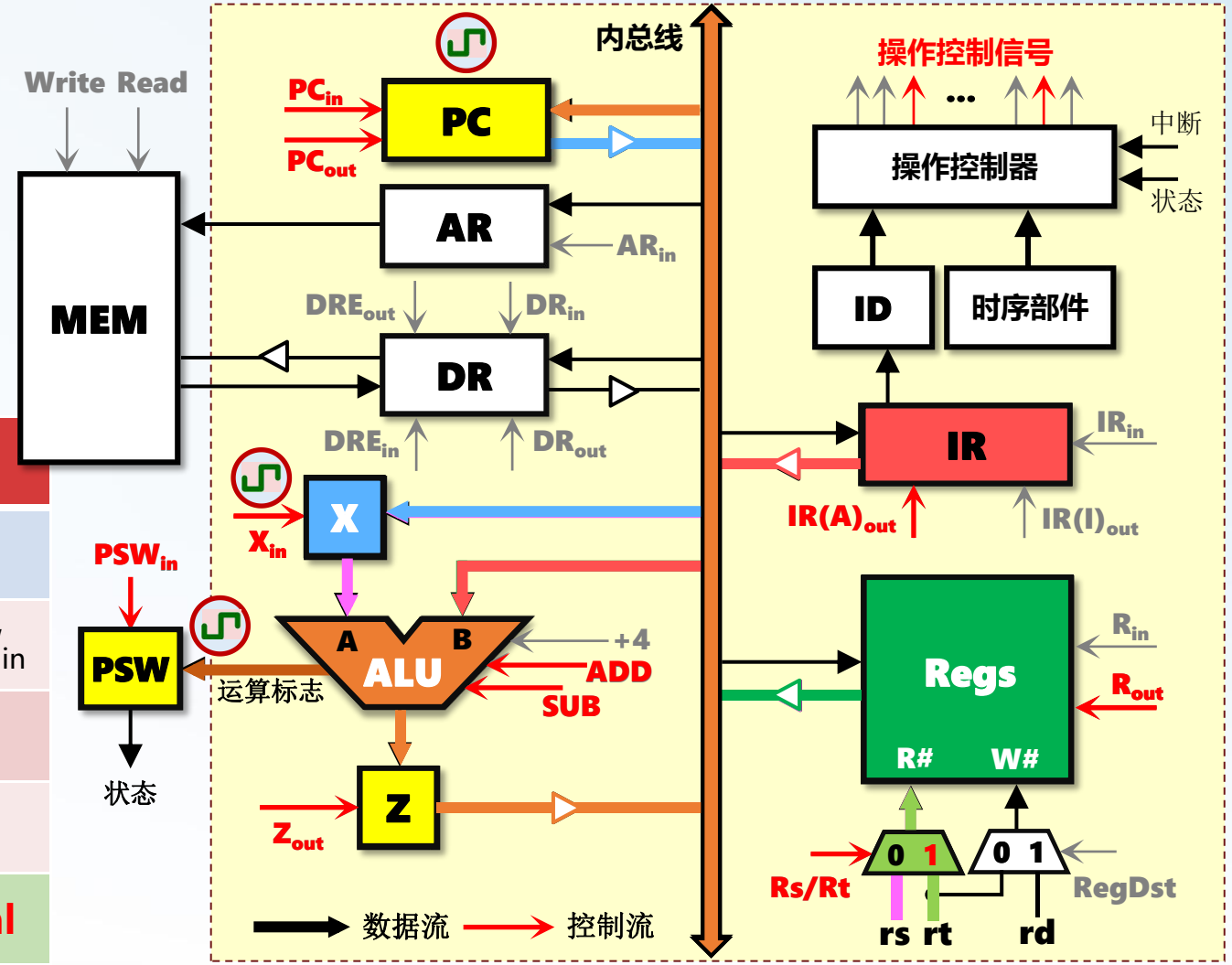
if(PSW.equal) Z→PC: 计算分支地址，根据标志位进行分支。

beq 指令数据通路 *beq rs,rt,imm*

if (R[rs] = R[rt])

PC + 4 + imm<< 2 → PC

节拍	数据通路 (数据流)	控制信号(控制流)
T5	R[rs]→X	R_{out}, X_{in}
T6	X−R[rt]→PSW	R_{out}, Rs/Rt, SUB, PSW_{in}
T7	PC→ X	PC_{out}, X_{in}
T8	IR(A)+X→Z	$IR(A)_{out}$, ADD
T9	**If (PSW.equal) Z→PC**	**Z_{out}, PC_{in}=PSW.equal**



4. addi指令的执行流程

指令功能：将rs寄存器与立即数相加的结果送入rt寄存器

汇编代码： addi rt,rs, imm

指令格式：

addi指令执行共需要2个机器周期：

第一个机器周期为**取指周期M_{if}**；

第二个机器周期为**执行周期M_{ex}**，将寄存器rs的内容送入X，再将指令字中的立即数符号扩展成32位后送入ALU进行加法运算，最后将结果送入rt寄存器。

addi指令操作流程及控制信号

周期	节拍	操作	功能说明	控制信号
执行周期 M_{ex}	T1	R[rs]→X	将rs寄存器内容送入暂存器，准备进行加法运算	$R_{out}=X_{in}=1$
	T2	IR(I)+X→Z	将IR中的立即数符号扩展成32位后送入ALU做加法，结果送到暂存器Z	$IR(I)_{out}=ADD=1$
	T3	Z→R[rt]	将暂存器Z的运算结果送入目的寄存器rt	$Z_{out}=R_{in}=1$

注意:addi指令的取指机器周期使用的数据通路**与lw指令相同**。

addi指令执行周期M_{ex}数据通路:

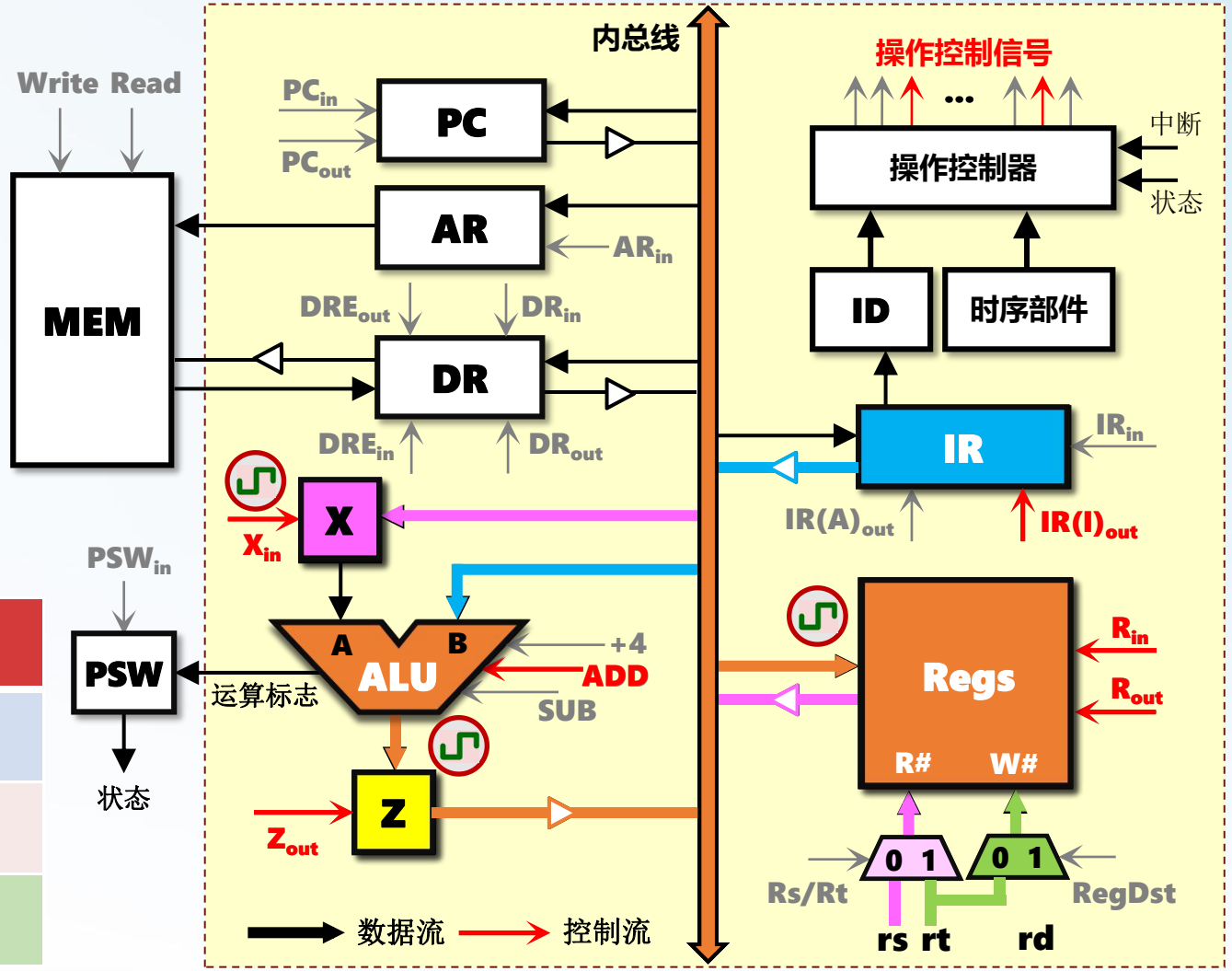
R[rs]→X；IR(I)→ ALU→Z→R[rt]

计算rs寄存器与立即数的和并送入rt寄存器。

addi指令执行数据通路 *addi rt,rs,imm*

R[rs] + imm → R[rt]

节拍	数据通路 (数据流)	控制信号(控制流)
T5	R[rs]→X	R_{out} , X_{in}
T6	X+imm→Z	IR(I)out , ADD
T7	**Z→R[rt]**	**Z_{out} , R_{in}**



5. add指令的执行流程

指令功能：将rs寄存器与rt寄存器相加的结果送入rd寄存器

汇编代码： add rd,rs,rt

指令格式：

add指令执行共需要2个机器周期：

第一个机器周期为**取指周期M_{if}**；

第二个机器周期为**执行周期M_{ex}**，即将寄存器rs的内容送入X,再将寄存器rt的内容送入ALU进行加法运算，最后将结果送入rd寄存器。

add指令操作流程及控制信号

周期	节拍	操作	功能说明	控制信号
执行周期 M_{ex}	T1	R[rs]→X	将rs寄存器内容送入暂存器，准备进行加法运算	$R_{out}=X_{in}=1$
	T2	R[rt]+X→Z	将rt寄存器内容送入ALU进行加法运算，结果送入Z	$R_{out}=Rs/Rt=ADD=1$
	T3	Z→R[rd]	将暂存器Z的运算结果送入目的寄存器rd	$Z_{out}=RegDst=R_{in}=1$

注意:add指令的取指机器周期使用的数据通路**与lw指令相同**。

add指令执行周期M_{ex}数据通路:

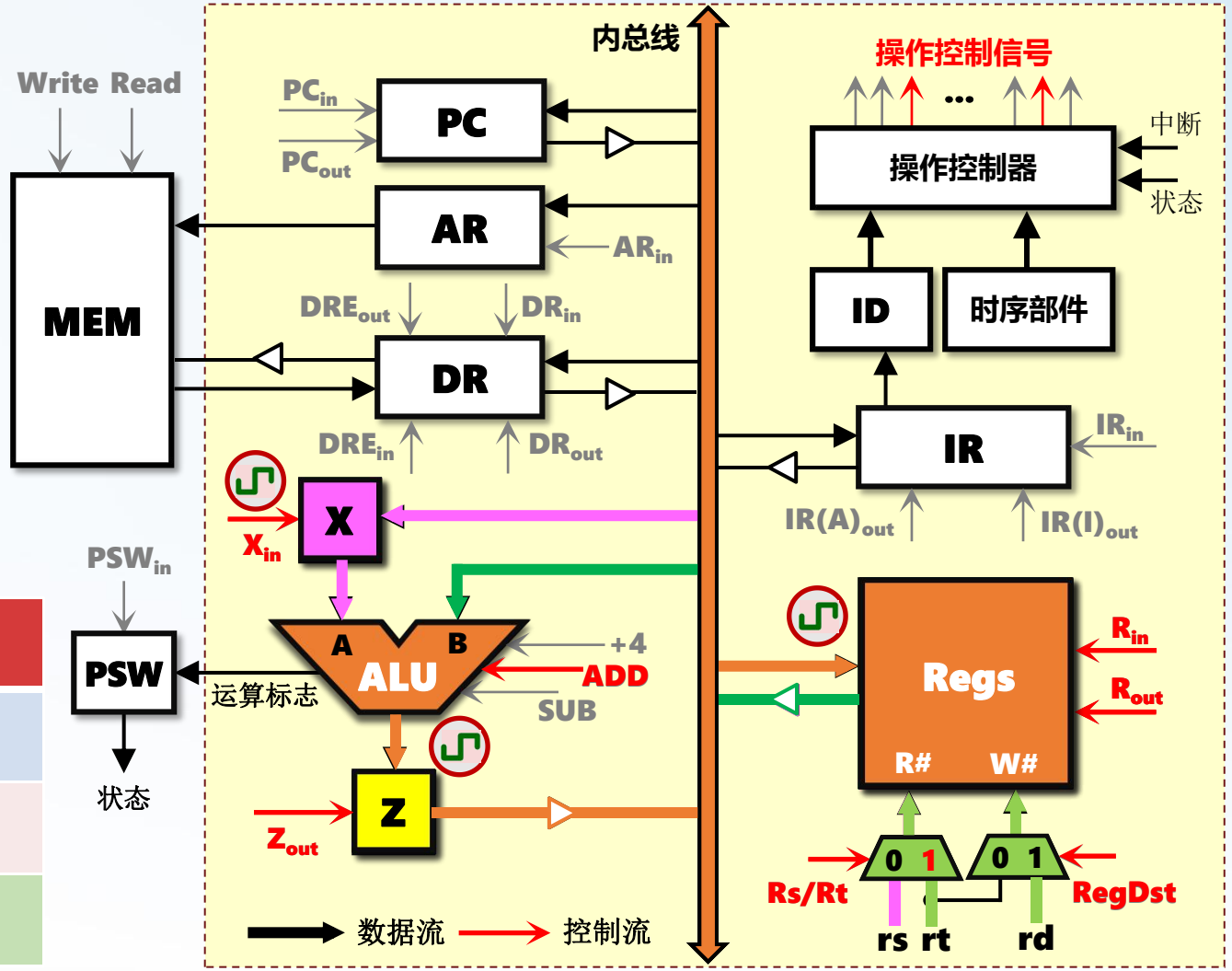
R[rs]→X→ALU; R[rt]→ALU→Z→R[rd]

计算rs与rt寄存器的和并送入rd。

add指令执行数据通路 *add rd,rs,rt*

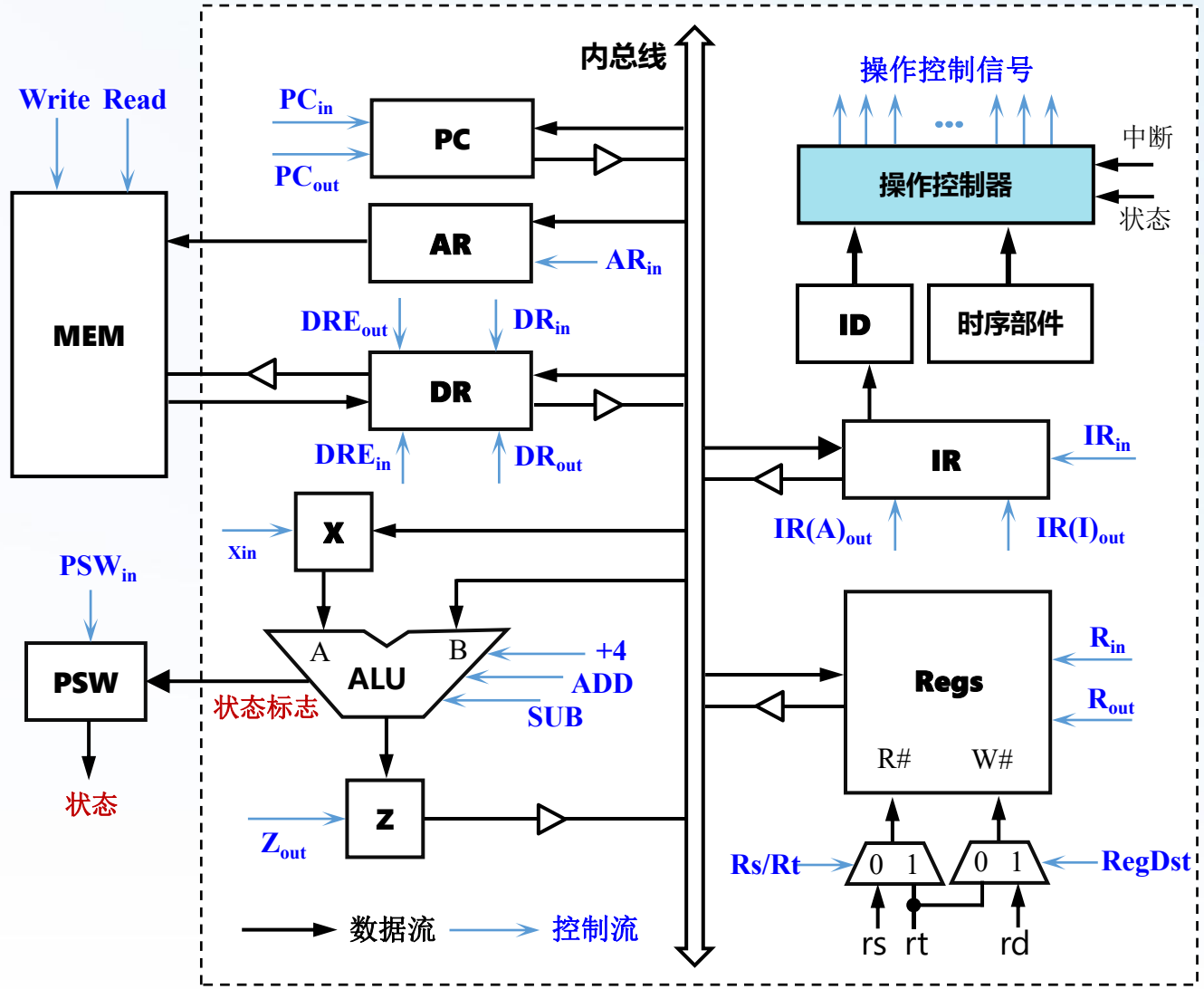
6bits	5bits	5bits	5bits	5bits	6bits
000000	rs	rt	rd	shamt	funct

R[rs] + R[rt] → R[rd]



节拍	数据通路 (数据流)	控制信号(控制流)
T5	R[rs]→X	R_{out}, X_{in}
T6	X+R[rt]→Z	Rs/Rt, R_{out}, ADD
T7	**Z→R[rd]**	**Z_{out}, R_{in}, RegDst**

【思考】： 已知sub功能为： R[rd]=R[rs]-R[rt]，试给出其数据通路及相关控制信号。



sub指令操作流程及控制信号

周期	节拍	操作	功能说明	控制信号
取指周期 M_{if}	T1	PC→AR; PC→X	将程序计数器PC内容送入AR,同时送入暂存器X	PC_{out}=AR_{in}=X_{in}=1
	T2	X+4→Z	将PC值加4并送入暂存器Z	+4=1
	T3	Z→PC; M[AR]→DR	将暂存器Z内容回送到PC，同时读AR内容对应主存单元的值并送入DR	Z_{out} =PC_{in}=1 Read=DRE_{in}=1
	T4	DR→IR	将DR内容送入IR,完成取指令	DR_{out}=IR_{in}=1
执行周期 M_{ex}	T1	R[rs]→X	将rs寄存器内容送入暂存器，准备进行减法运算	R_{out}=X_{in}=1
	T2	X-R[rt]→Z	将rt寄存器内容送入ALU进行减法运算，结果送入Z	R_{out}=Rs/Rt=SUB=1
	T3	Z→R[rd]	将暂存器Z的运算结果送入目的寄存器rd	Z_{out}=RegDst=R_{in}=1

注意:表中的控制信号仅给出非零值的信号，未给出的信号值为零。

【例】SPECINT2000 基准测试程序包含的取数据、存数据、条件分支指令、跳转指令、R型算术逻辑运算指令比例分别为25%、10%、11%、 2%、52%。求此基准测试程序在教材图6.8所示的单总线结构计算机上运行的CPI。假设程序指令数目为1000亿条。如果CPU采用 65nm CMOS工艺实现，各功能部件的时间延迟如表所示，求该计算机最大时钟频率以及基准测试程序的执行时间。

	参数	延迟	功能部件	参数	延迟
寄存器延迟	$T_{clk\_to\_q}$	30 ps	运算器ALU	T_{alu}	200 ps
存储器读	T_{mem}	250 ps	多路选择器	T_{mux}	25 ps
寄存器堆读	T_{RF_read}	150 ps	寄存器建立时间	T_{setup}	20ps

【解】：CPI是每条指令CPI的加权平均值，根据教材图6.14可知：

取数据、存数据、条件分支指令、R型运算指令的CPI分别为9、9、9、7。无条件跳转指令无须计算分支条件，无须计算周期,所以假设时钟周期数为7，因此测试程序的CPI为:

$$CPI=0.25\times9+0.1\times9+0.11\times9+0.02\times7+0.52\times7=7.92$$

单总线结构计算机的最小时钟周期为:

$$T_{min\_clk}=T_{clk\_to\_q}+\max(T_{alu},T_{mem})+T_{setup}$$

$$=30+\max(200,250)+20$$

$$=300ps$$

最大时钟频率为:

$T_{max\_freg} = 1/(300\times10^{-12}) = \mathbf{3.33GHz}$

程序执行时间为:

T_{total}=指令条数×CPI× T_{min_clk}

$=1000\times10^{8}\times7.92\times300\times10^{-12}$

$= \mathbf{237.6s}$

数据通路举例---2009考研统考

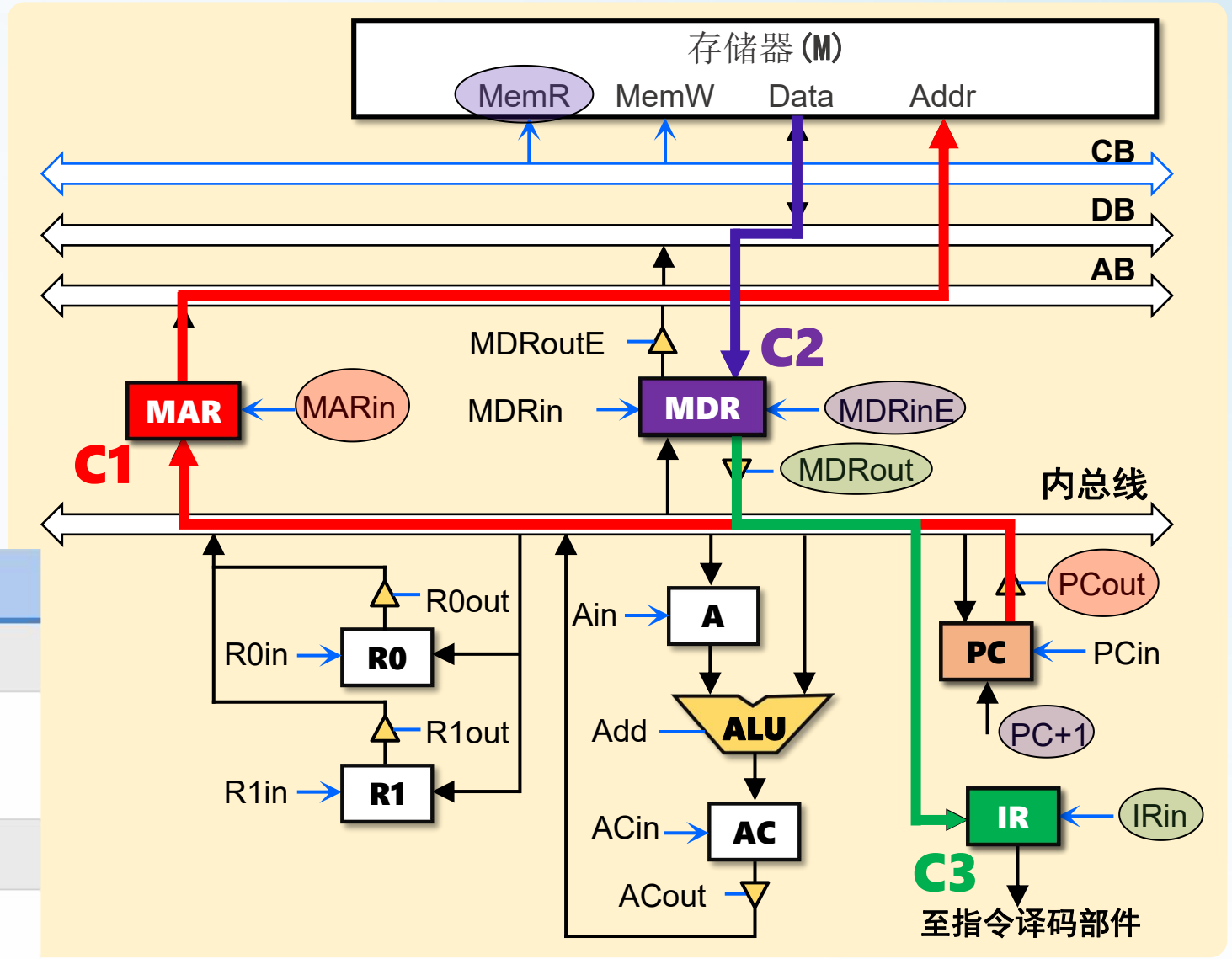
- 某机字长16位，指令16位定长；
- 指令ADD (R1)，R0的功能为(R0)+((R1)) →(R1)，即将R0中数据与R1内容所指向的主存单元的数据相加，并将结果送入R1内容所指向的主存单元中；
- 数据通路图中控制信号为1表示有效，假设MAR输出一直处于使能状态；

右表为取指令和译码阶段每个节拍(时钟周期)的功能和控制信号，请按相同方式给出执行阶段各节拍的功能和有效控制信号。

时钟	功能	有效控制信号
C1	MAR←(PC)	PCout, MARin
C2	MDR←M(MAR) PC←(PC)+1	MemR, MDRinE, PC+1
C3	IR←(MDR)	MDRout, IRin
C4	指令译码	无

数据通路举例---取指令周期

Mem(PC++) → IR

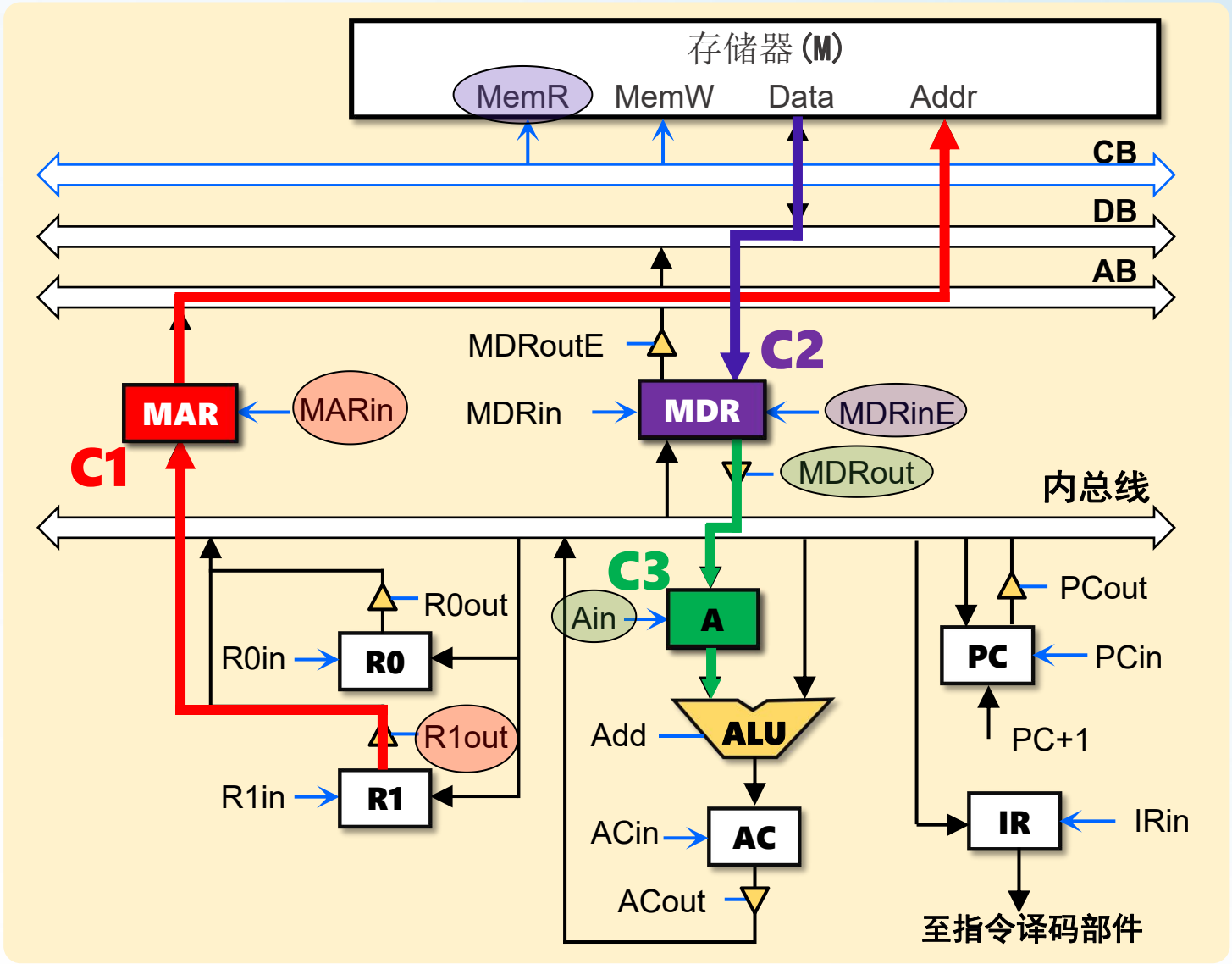


时钟	功能	有效控制信号
C1	MAR←(PC)	PCout, MARin
C2	MDR←M(MAR) PC←(PC)+1	MemR, MDRinE, PC+1
C3	IR←(MDR)	MDRout, IRin
C4	指令译码	无

数据通路举例---执行指令周期

((R1)) → A

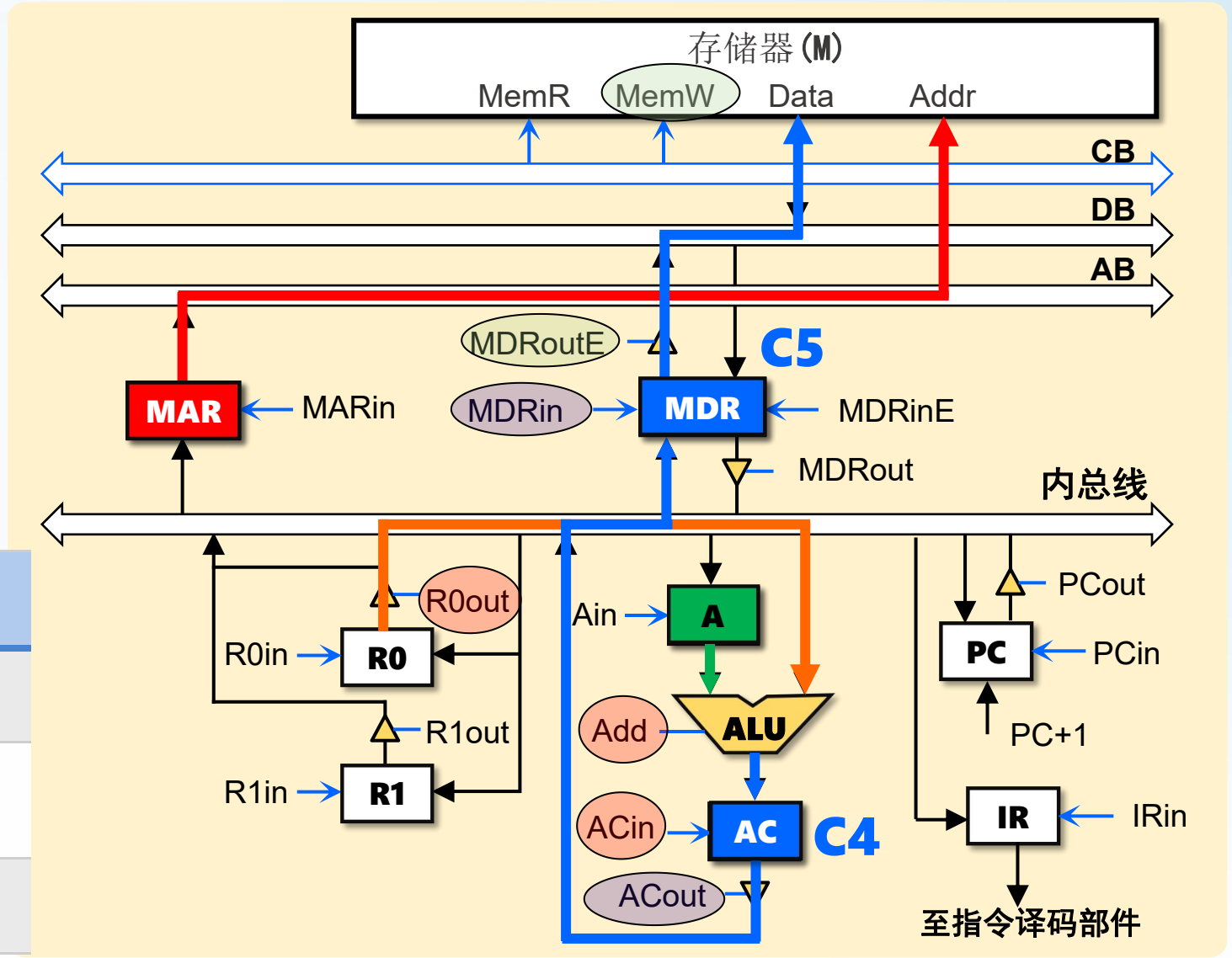
时钟	功能	有效控制信号
C1	MAR←(R1)	R1out, MARin
C2	MDR←M(MAR)	MemR, MDRinE
C3	A←(MDR)	MDRout, Ain



数据通路举例---执行指令周期

(R0)+((R1))→(R1)

时钟	功能	有效控制信号
C4	AC← ALU	R0out, ADD,ACin
C5	MDR←AC	ACout,MDRin
C6	M(MAR)←(MDR)	MDRoutE,MemW



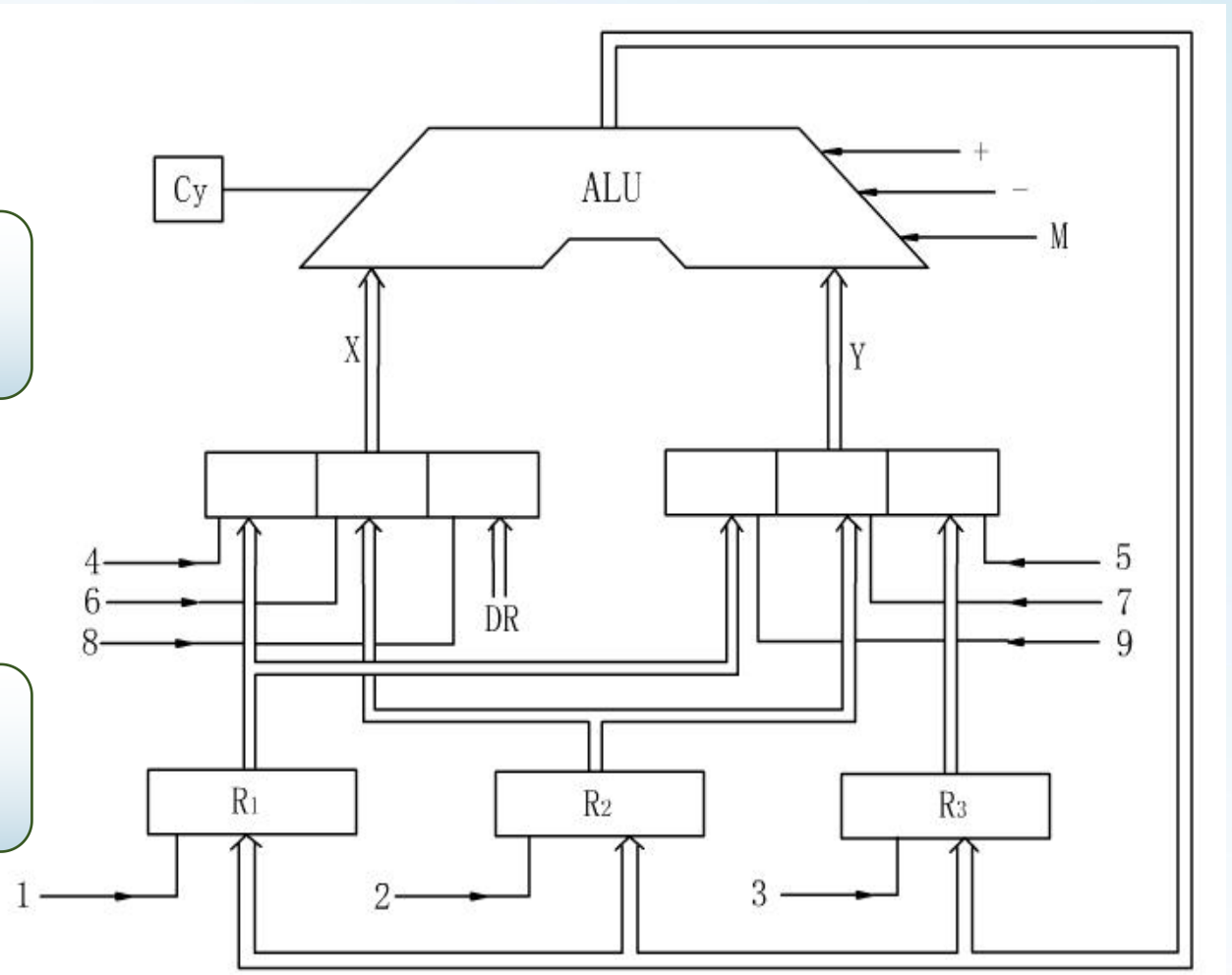
【例】观察给出的运算器模型，分析哪些微操作是相容的，哪些是互斥的？

相容的微操作

（1，2，3）、（4，5）…（8，9）等

互斥的微操作

（+，-，M）、（4，6，8）、（5，7，9）

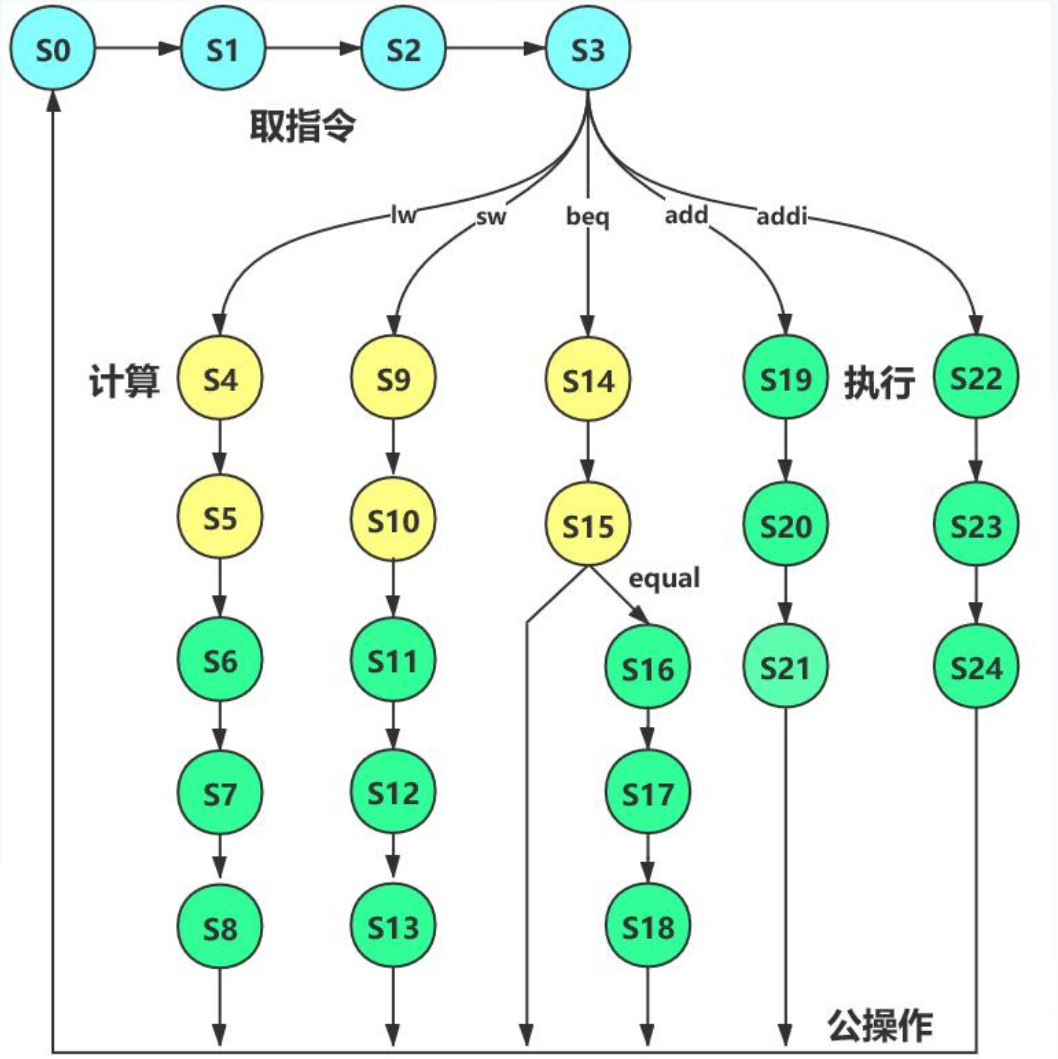


3. 微程序设计

一条指令对应一段微程序，一段微程序又包括若干条微指令，假设一个微指令周期就是一个时钟周期，那么一个指令周期需要多少个时钟节拍就应该安排多少条微指令。

具体构建微程序时可以参考教材图6.45所示的指令执行状态转换图，图中一个状态对应一个时钟周期，微操作控制信号的值仅与现态有关。控制存储器中的微指令可以和状态转换图中的状态一一对应，状态的编号值可以转换成微指令地址；而某一个状态需要给出的微操作控制信号可以映射到对应微指令操作控制字段的控制信号位中；状态之间的切换关系可以对应微指令之间的执行顺序，用于设置判别测试字段以及下址字段。

指令执行状态转换图 → 状态转换表



现态	lw	sw	beq	add	addi	equal	次态
S0	x	x	x	x	x		S1
S1	x	x	x	x	x		S2
S2	x	x	x	x	x		S3
S3	1						S4
S3		1					S9
S3			1				S14
S3				1			S19
S3					1		S22

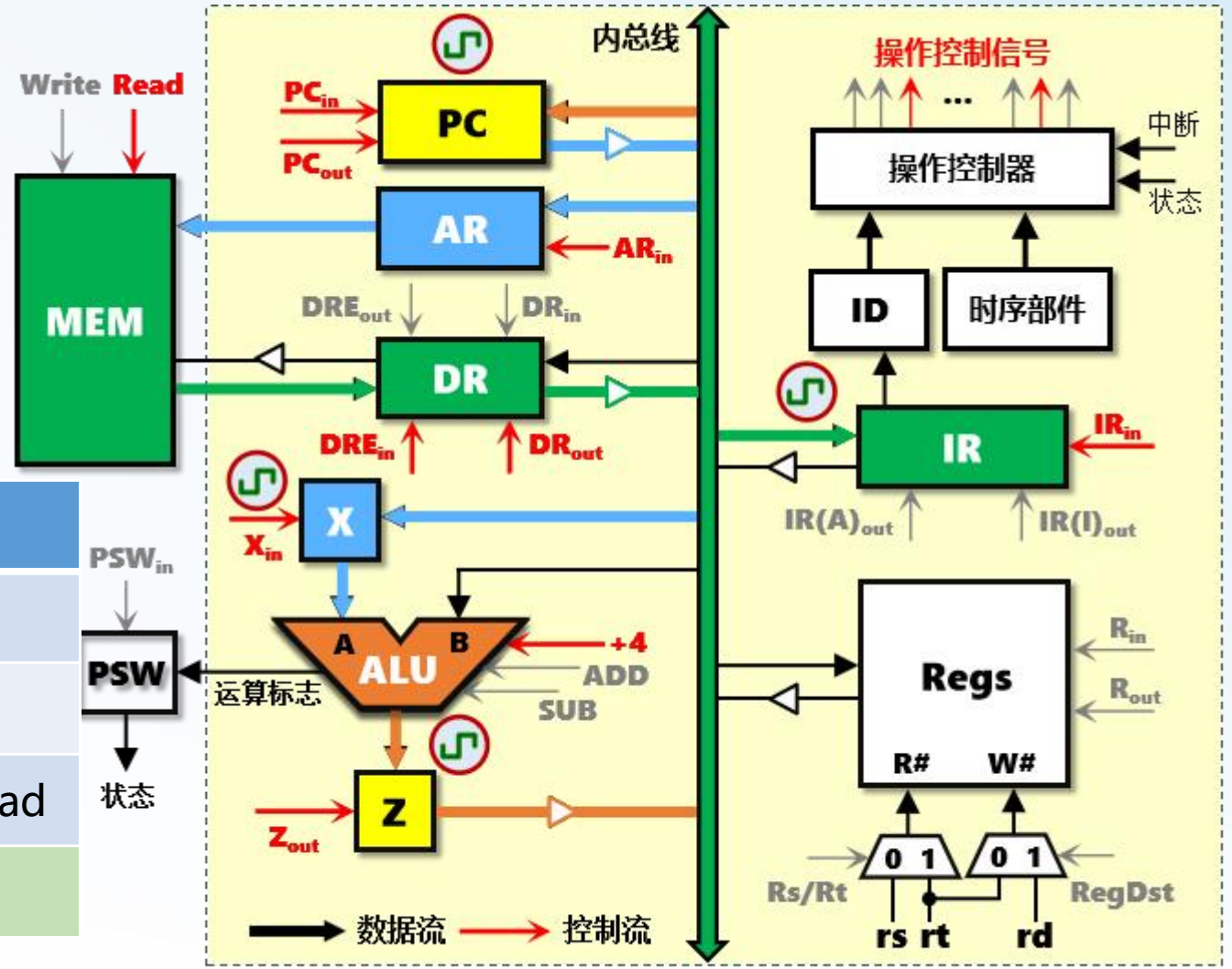
（1）取指微程序

M[PC++] → IR

- 4个时钟周期
- 四条微指令

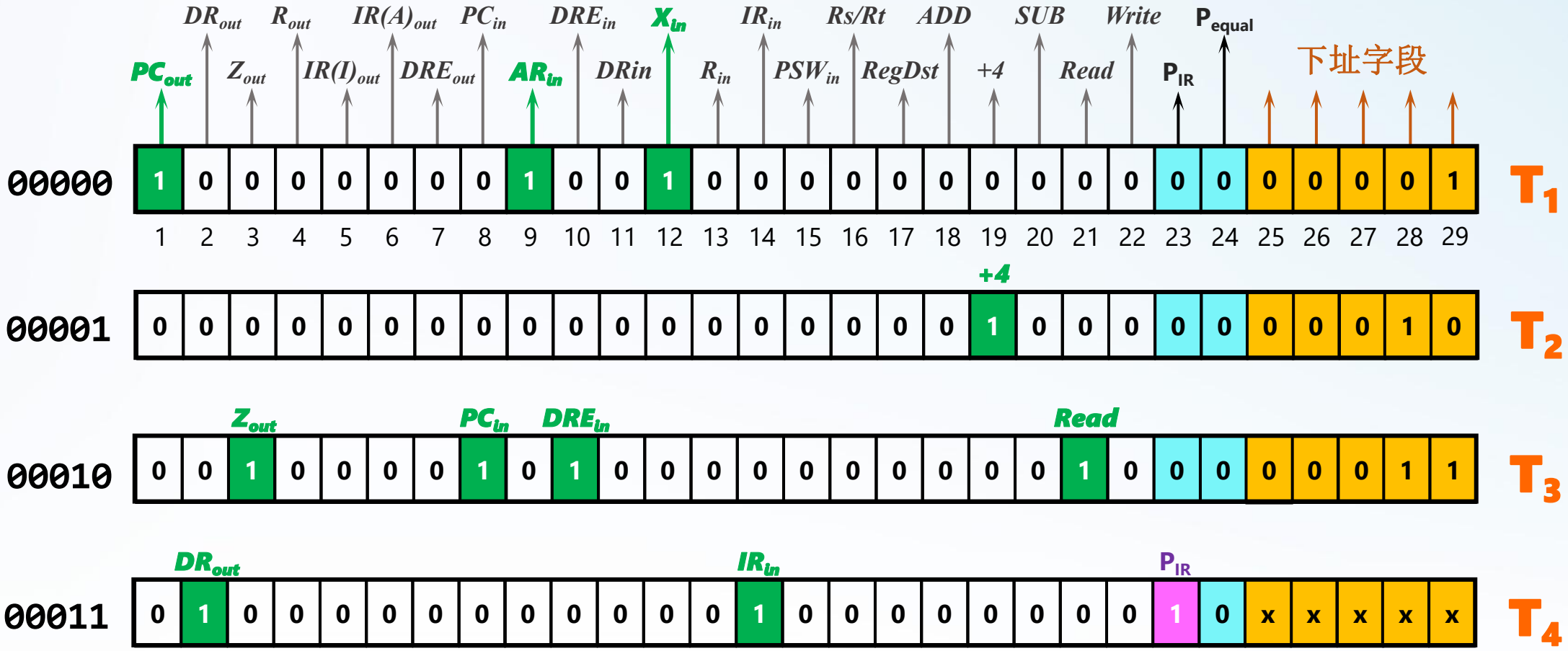
节拍	数据通路 (数据流)	控制信号(控制流)
T1	PC→AR，PC→X	PC_{out}, AR_{in}, X_{in}
T2	X+4→Z	+4
T3	Z→PC，M[AR]→DR	Z_{out}, PC_{in}, DRE_{in}, Read
T4	**DR→IR**	**DR_{out}, IR_{in}**

取指令数据通路



取指微程序设计

节拍	数据通路（数据流）	控制信号(控制流)
T1	PC→AR，PC→X	PC_{out}, AR_{in}, X_{in}
T2	X+4→Z	+4
T3	Z→PC，M[AR]→DR	Z_{out}, PC_{in}, DRE_{in}, Read
T4	DR→IR	DR_{out}, IR_{in}



（2）lw指令微程序

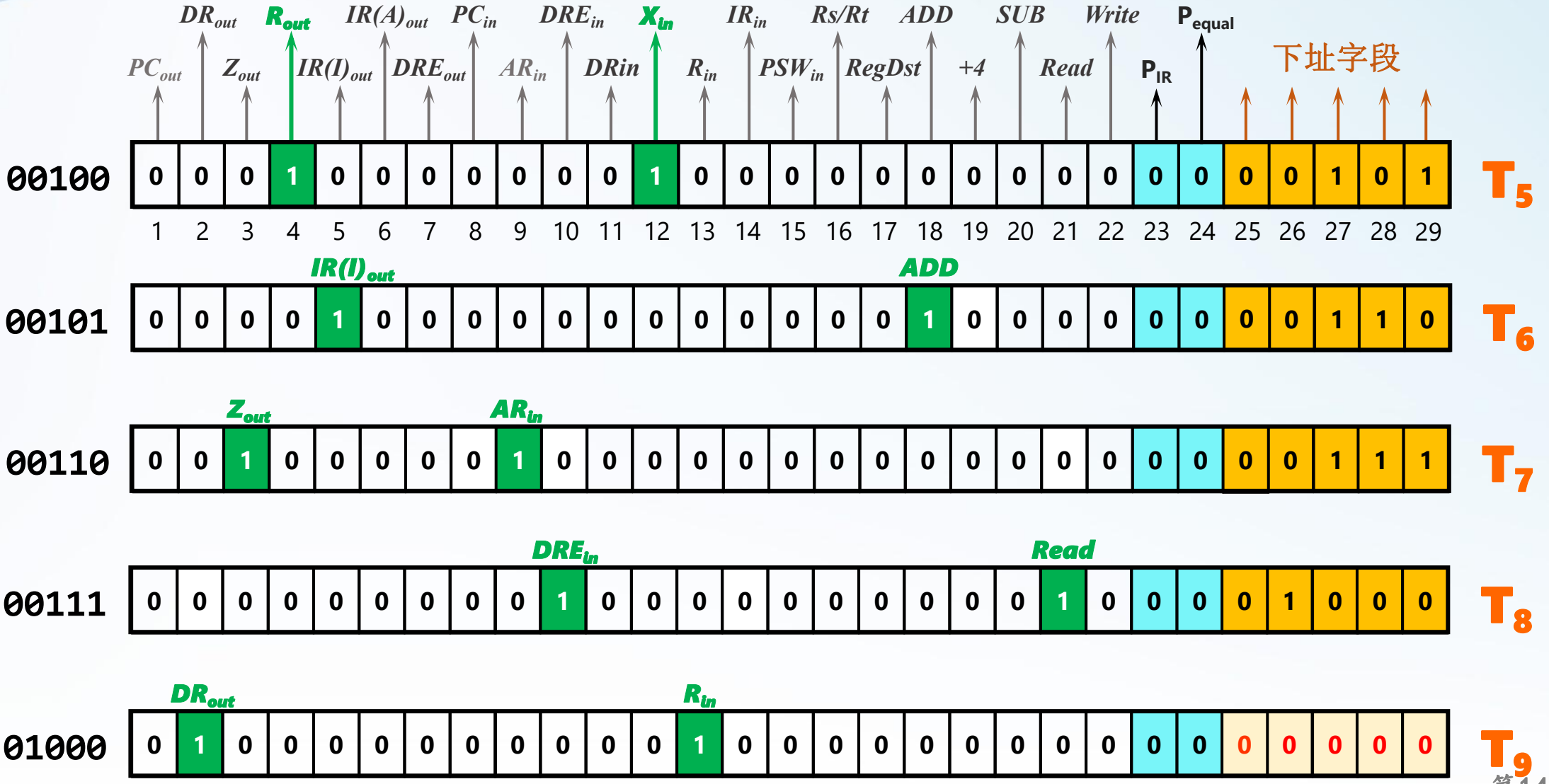
lw指令执行周期对应状态机中的S4~ S8，状态切换顺序是：

S4→S5→S6→S7→S8→S0

所以lw指令微程序包含5条微指令，假设按顺序存放在控制存储器中的4~8号单元，各指令的下址字段也可以按状态切换的顺序进行安排，分别是5、6、7、8、0。

周期	节拍	操作	功能说明	控制信号
计算周期 Mcal	T1	R[rs]→X	将rs寄存器内容送入暂存器X,准备计算访存地址	$R_{out}=X_{in}=1$
	T2	IR(I)+X→Z	将IR中的立即数符号扩展为32位并送入ALU做加法运算	$IR(I)_{out}=ADD=1$
执行周期 Mex	T1	Z→AR	将Z中暂存的访存地址送入AR	$Z_{out}=AR_{in}=1$
	T2	M[AR]→DR	读AR内容对应主存单元的值并送入DR	$Read=DRE_{in}=1$
	T3	DR→R[rt]	将DR内容送入寄存器rt	$DR_{out}=R_{in}=1$

lw指令微程序设计


DR_out R_out IR(A)_out PC_in DRE_in X_in IR_in Rs/Rt ADD SUB Write P_equal
PC_out Z_out IR(I)_out DRE_out AR_in DRin R_in PSW_in RegDst +4 Read P_IR
下址字段
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
00100 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 T5
IR(I)_out ADD
00101 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 T6
Z_out AR_in
00110 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 T7
DRE_in Read
00111 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 T8
DR_out R_in
01000 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 T9

（3）sw指令微程序

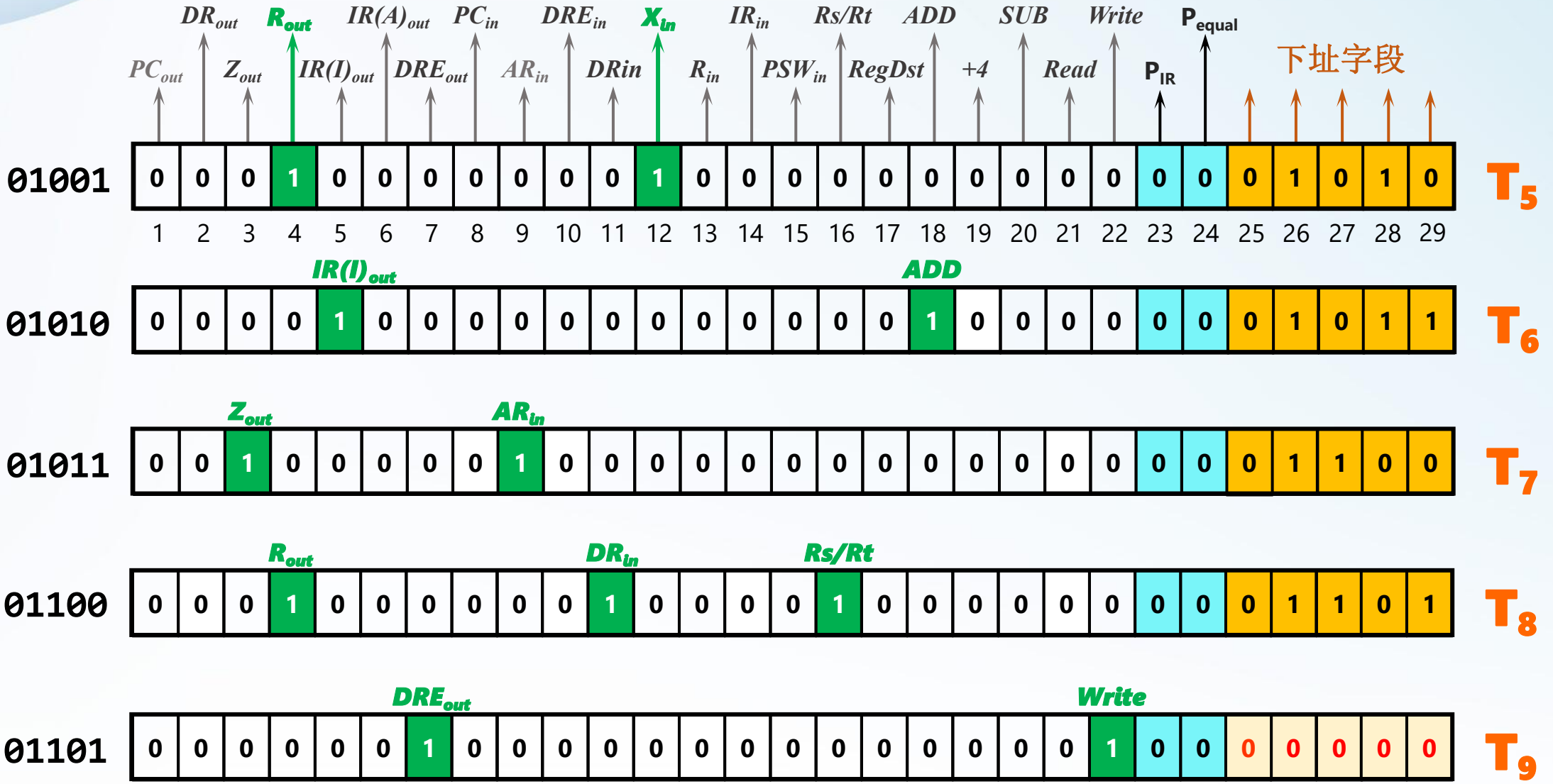
sw指令执行周期对应状态机中的S9~ S13，状态切换顺序是：

S9→S10→S11→S12→S13→S0

所以sw指令微程序包含5条微指令，假设按顺序存放在控制存储器中的9~13号单元。

周期	节拍	操作	功能说明	控制信号
计算周期 Mcal	T1	R[rs]→X	将rs寄存器内容送入暂存器X,准备计算访存地址	$R_{out}=X_{in}=1$
	T2	IR(I)+X→Z	将IR中的立即数符号扩展为32位并送入ALU做加法运算	$IR(I)_{out}=ADD=1$
执行周期 Mex	T1	Z→AR	将Z中暂存的访存地址送入AR	$Z_{out}=AR_{in}=1$
	T2	R[rt]→DR	将rt寄存器内容送入DR	$R_{out}=Rs/Rt=DR_{in}=1$
	T3	DR →M[AR]	将DR内容写入AR所指向的主存单元	$DRE_{out}=Write=1$

sw指令微程序设计


DR_out R_out IR(A)_out PC_in DRE_in X_in IR_in Rs/Rt ADD SUB Write P_equal
PC_out Z_out IR(I)_out DRE_out AR_in DRin R_in PSW_in RegDst +4 Read P_IR
下址字段
01001 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 T5
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
IR(I)_out ADD
01010 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 1 T6
Z_out AR_in
01011 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 T7
R_out DR_in Rs/Rt
01100 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 T8
DRE_out Write
01101 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 T9

（4）beq指令微程序

beq指令比较特殊，在状态机中存在两条路径：

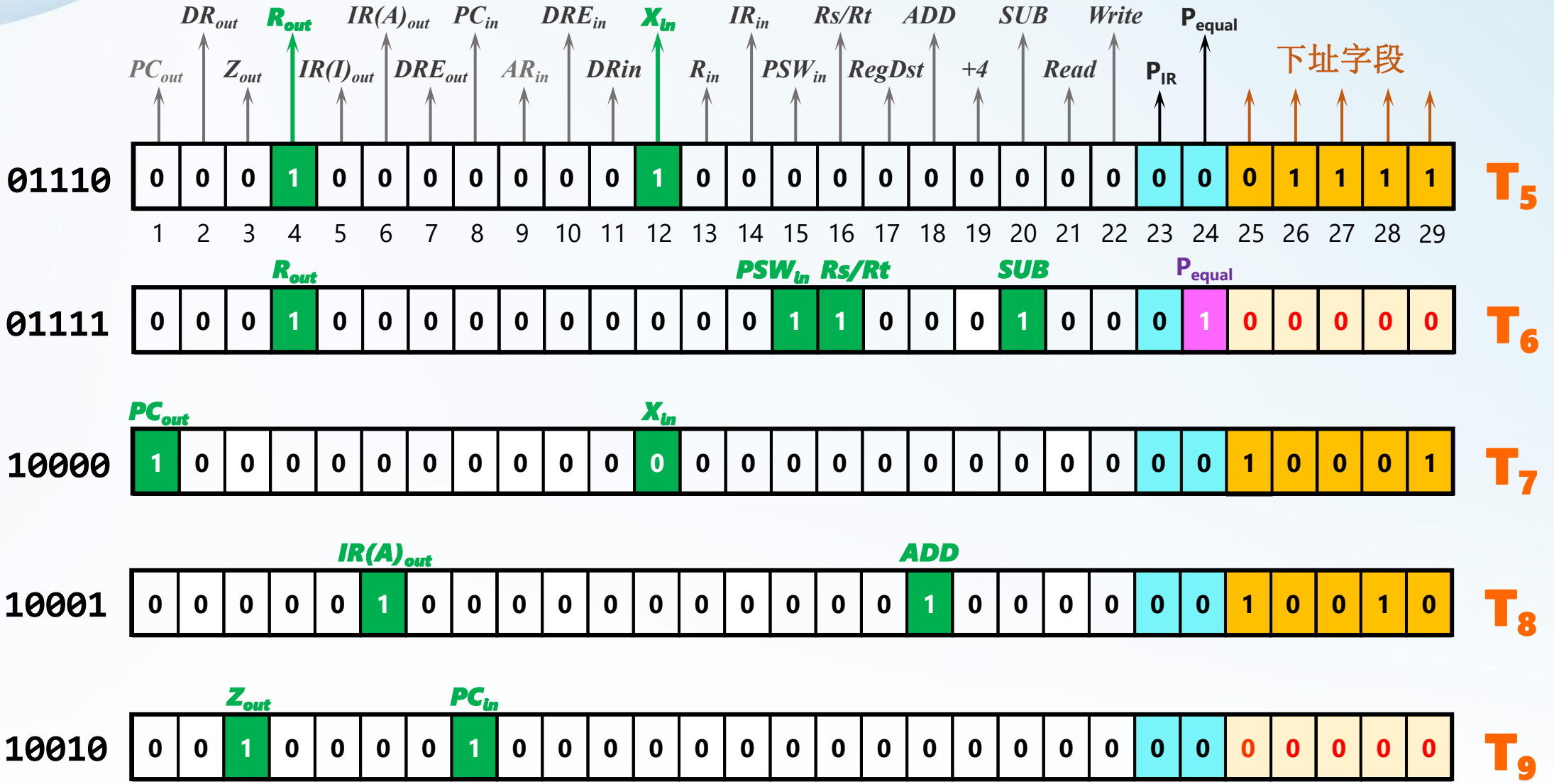
当equal=0时，状态路径是：S14→S15→S0，只需要两个时钟周期；

当equal=1时，状态路径是：S14→S15→S16→S17→S18→S0，需要5个时钟周期。

beq 指令微程序在控制存储器中的地址是14~ 18，但在15号微指令处需要进行微程序的条件分支判断。判别测试字段**P1**代表equal测试，应设置为1。执行15号微指令时，如果equal标志位为0，则下址字段有效，下址字段应该设置为0；如果equal为1，应该将PI位对应的分支目标地址16送入微地址寄存器AR，从而实现微程序的条件分支跳转。

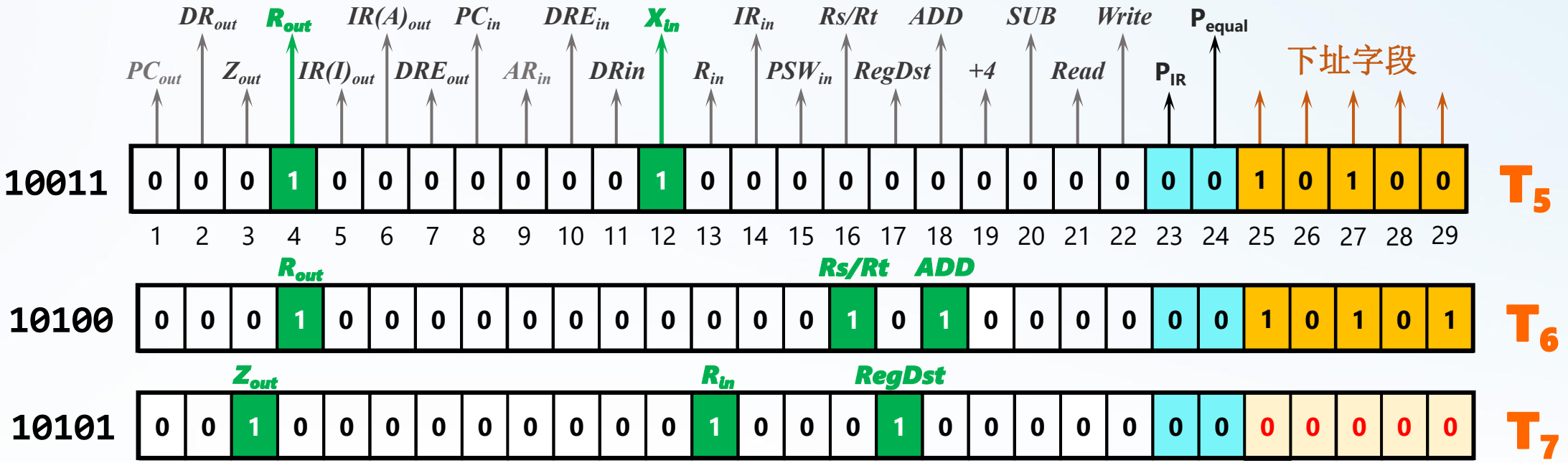
周期	节拍	操作	功能说明	控制信号
计算周期 M_{cal}	T1	R[rs]→X	将rs寄存器内容送入暂存器X,准备进行比较	$R_{out}=X_{in}=1$
	T2	X-R[rt]→PSW	将rt寄存器内容送入ALU做减法，可产生结果为零的标志位。如果equal标志位为0，则结束；为1，继续执行周期。	$R_{out}=Rs/Rt=SUB=PSW_{in}=1$
执行周期 M_{ex}	T1	PC→X	将PC内容送入暂存器X	$PC_{out}=X_{in}=1$
	T2	IR(A)+X→Z	将IR中的立即数符号扩展为32位后左移两位送入ALU做加法，计算出分支目标地址	$IR(A)_{out}=ADD=1$
	T3	Z→PC	将分支目标地址送入PC	$Z_{out}=PC_{in}=1$

beq指令微程序设计


DR_out R_out IR(A)_out PC_in DRE_in X_in IR_in Rs/Rt ADD SUB Write P_equal
PC_out Z_out IR(I)_out DRE_out AR_in DRin R_in PSW_in RegDst +4 Read P_IR
下址字段
01110 T5
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
R_out PSW_in Rs/Rt SUB P_equal
01111 T6
PC_out X_in
10000 T7
IR(A)_out ADD
10001 T8
Z_out PC_in
10010 T9

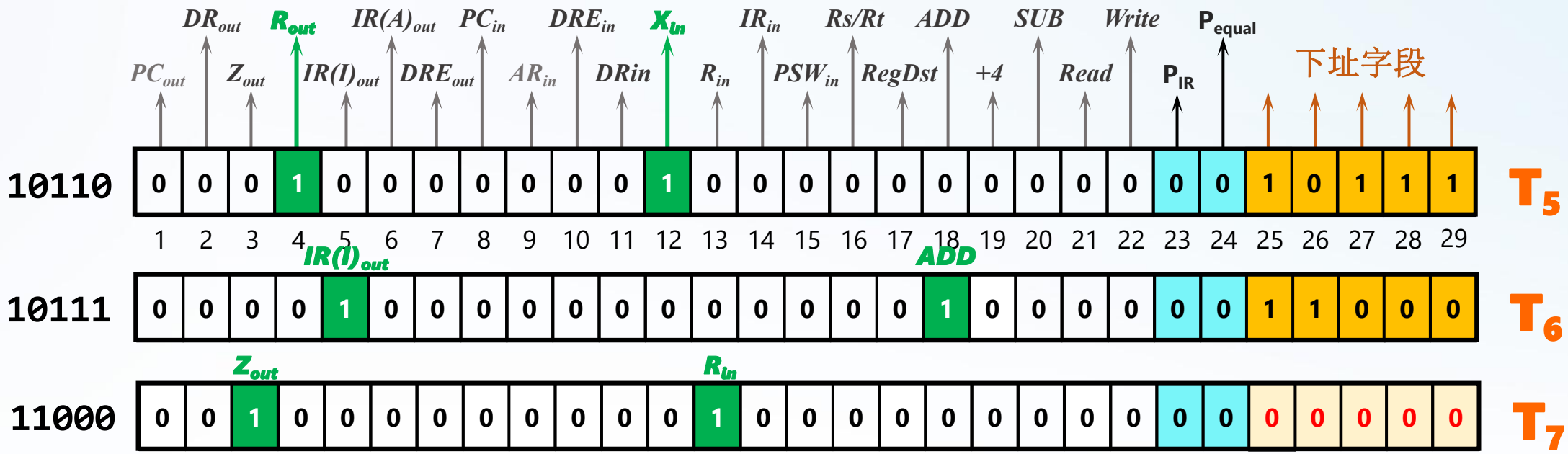
（5）add指令微程序

周期	节拍	操作	功能说明	控制信号
执行周期 M_{ex}	T1	R[rs]→X	将rs寄存器内容送入暂存器，准备进行加法运算	$R_{out}=X_{in}=1$
	T2	R[rt]+X→Z	将rt寄存器内容送入ALU进行加法运算，结果送入Z	$R_{out}=Rs/Rt=ADD=1$
	T3	Z→R[rd]	将暂存器Z的运算结果送入目的寄存器rd	$Z_{out}=RegDst=R_{in}=1$



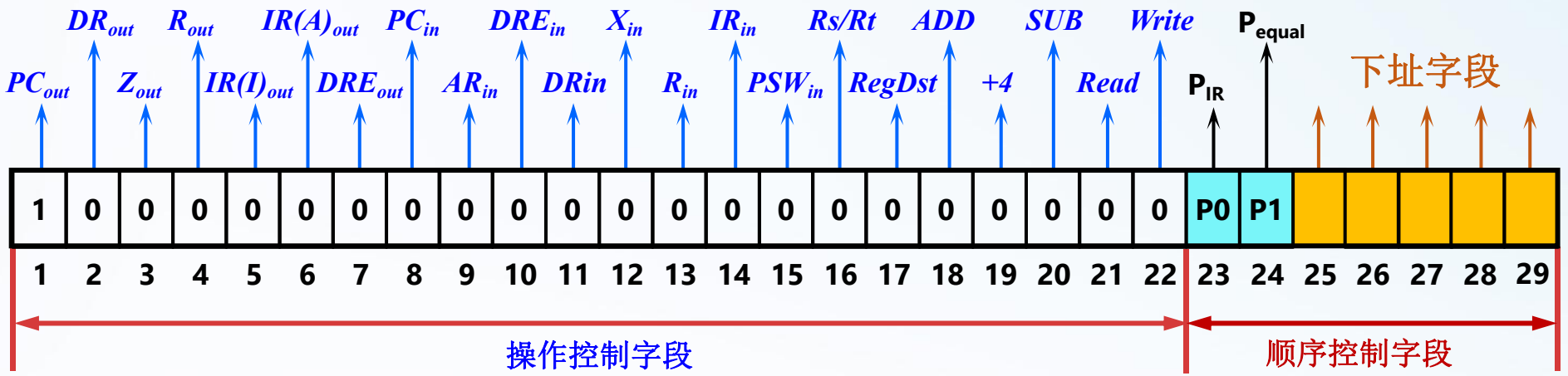
addi指令微程序

周期	节拍	操作	功能说明	控制信号
执行周期 M_{ex}	T1	R[rs]→X	将rs寄存器内容送入暂存器，准备进行加法运算	R_{out}=X_{in}=1
	T2	IR(I)+X→Z	将IR中的立即数符号扩展成32位后送入ALU做加法，结果送到暂存器Z	$IR(I)_{out}$=ADD=1
	T3	Z→R[rt]	将暂存器Z的运算结果送入目的寄存器rt	Z_{out}=R_{in}=1

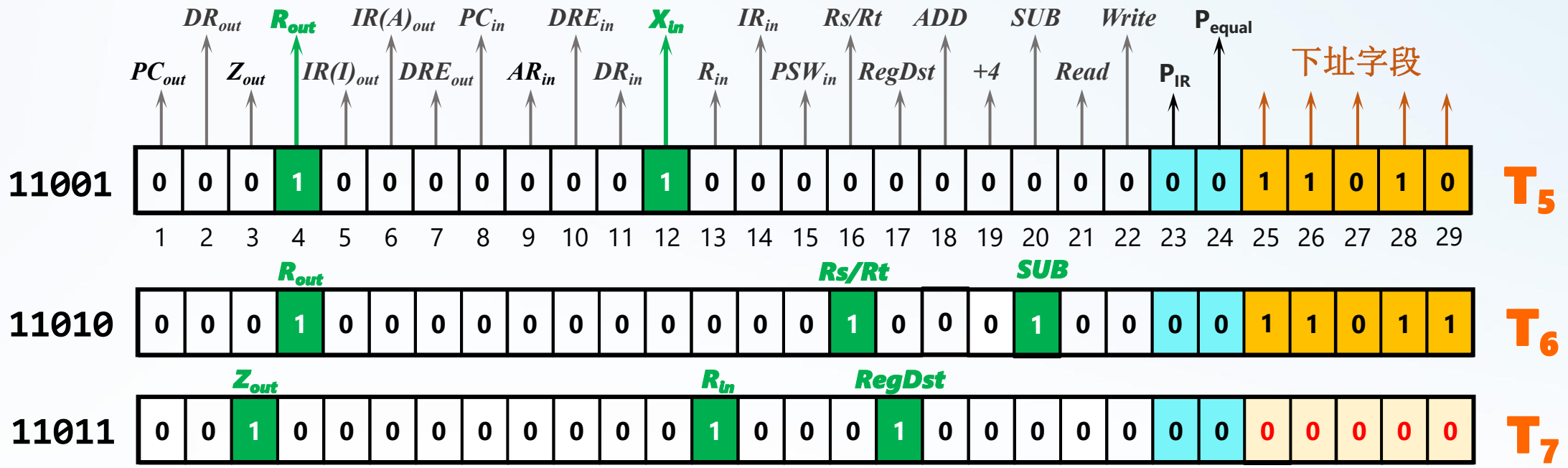


【思考】：试设计 sub微程序中的操作控制字段，其功能为:

R[rd]=R[rs]-R[rt]，设微指令分别放在25,26,27单元



周期	节拍	操作	功能说明	控制信号
执行周期 M_{ex}	T1	R[rs]→X	将rs寄存器内容送入暂存器，准备进行减法运算	$R_{out}=X_{in}=1$
	T2	X-R[rt]→Z	将rt寄存器内容送入ALU进行减法运算，结果送入Z	$R_{out}=Rs/Rt=SUB=1$
	T3	Z→R[rd]	将暂存器Z的运算结果送入目的寄存器rd	$Z_{out}=RegDst=R_{in}=1$



【例】 某微指令为24 位字长，采用混合控制法。其中23 ~ 15位用直接表示法；14 ~ 5分为A、B、C三组，均采用编码表示法，C组除表示4种控制转移的判别测试P1~P4外，其余均用于表示微命令，各字段的位数分配如图所示，回答下列问题：

(1) 该格式的微指令最多可表示多少种微命令?

(2) 一条微指令中可同时出现的微命令最多可以有多少个?

(3) 控制存储器的最大容量是多少?

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
直接表示法									A			B			C				下址字段					

【解】 (1)采用直接表示法的微命令有9个，A和B两组经译码后各可表示**7个**微命令，C字段的微命令个数为15-4=**11个**，所以该格式微指令最多可表示的微命令数目为：9+7+7+11=**34个**。

(2)采用编码表示法时每个字段最多只能使用一个微命令，该微指令有3个字段采用了编码表示法，故一条微指令中最多可同时出现的微命令个数为：

9+1+1+1=**12个**。

(3) 微指令中下址字段的位数决定了控制存储器的可寻址范围，由题干可知，下址字段位数为5位，即最多可以有 2^5=32个地址单元，该微指令格式所需的控制存储器的最大容量为：

微指令字长 x单元数量=24x32位=**768位**（或 **96字节**）

即控制存储器最多有32个24位存储单元。

【例】已知某机器共有微操作控制信号共有60个，构成5个相斥类的微命令组，各组分别包含4个，7个，8个，14个，27个微命令，已知判别测试字段长度为4位，微指令字长为32位。问：

（1）若操作控制字段采用编码表示法，给出该微指令的格式。

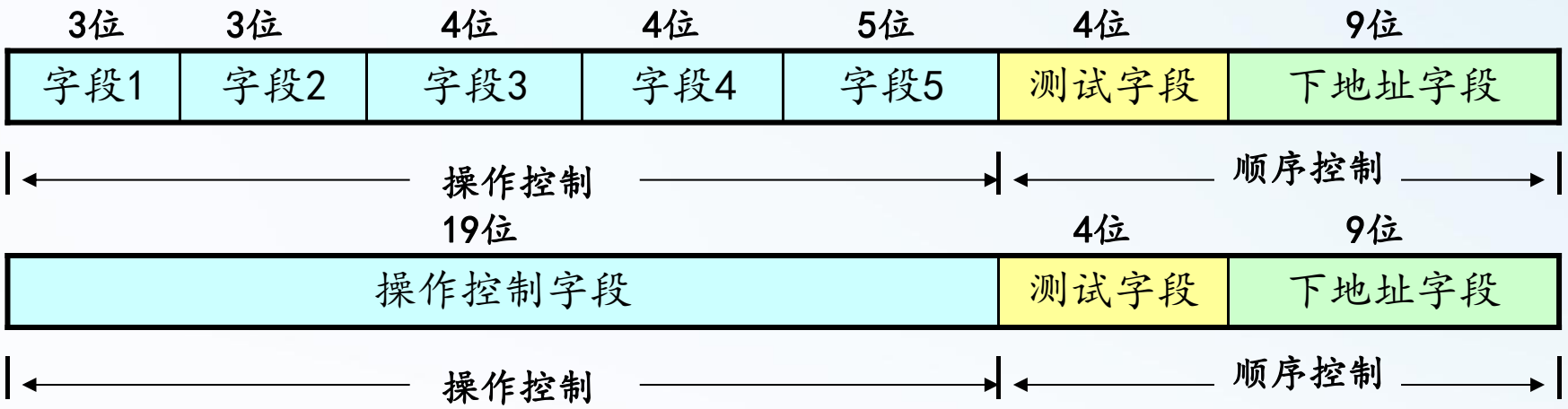
（2）若不限定微指令字长，其下址字段长度同（1）相同，每个微命令直接进行控制，则微指令字长为多少位?

（3）在（1）的条件下，控存容量为多少位?

【解】：（1） 微命令字段（操作控制）长度为：3+3+4+4+5=19位。

下地址字段长度为：32-19-4=9位。

注意：每组中必须包含一个不发出命令的操作。



（2）字长为：60+4+9=73位。

（3）控存容量为 $2^9 \times 32 = 512 \times 32 = 2^{14}$位$= 2^{11}$字节。

【例】 设某机有8条微指令I_1～I_8，每条微指令所包含的有效微命令如表所示。

每条微指令所包含的有效微命令列表

微指令	a	b	c	d	e	f	g	h	i	j
I_1	√	√	√	√	√					
I_2	√			√		√	√			
I_3		√						√		
I_4			√							
I_5			√		√		√		√	
I_6	√							√		√
I_7			√	√				√		
I_8	√	√						√		

a～j分别对应10种不同性质的微命令信号。假设一条微指令的操作控制字段仅限为8位，请安排微指令的操作控制字段格式。

【解】：为了压缩操作控制字段的长度，必须设法把互斥性微命令进行分组，并采用字段译码器译码后产生这些互斥性微命令。

从表中可以看出，不存在5个或5个以上的互斥性微命令，即对任何一条微指令而言，这些微命令中最多只有一个有效。经分析发现，在10个微命令中存在多组3个互斥的微命令，列举如下：

（b、f、i），（b、f、j），（b、g、j），（b、i、j），
（c、f、j），（d、i、j），（e、f、h），（e、f、j），
（f、h、i），（f、i、j）

为了将10个微命令信号压缩成8位来表示，需将6个不同的微命令信号分成两个小组，采用字段直接译码法来产生这些互斥性微命令，剩下的4个微命令则采用直接表示法来产生相应的微命令。

因此微指令的操作控制字段可以有四种不同的格式，如图所示 ：