计算机组成原理

第四章 存储子系统

2023/11/15

信息与软件工程学院
School of Information and Software Engineering

主 要 内 容

1. 概述
2. 半导体存储器
3. 主存储器组织
4. 磁表面存储器

4.3 主存储器组织

- 01. 主存储器的逻辑设计
- 02. 基本逻辑门及译码器
- 03. 动态存储器的刷新
- 04. 主存储器的校验方法

引言

主存储器的组织涉及到这样一些方面：

(1)从存储器基本逻辑设计角度看，半导体存储器的逻辑主要是寻址逻辑；

(2) 如果采用DRAM，还需考虑动态刷新问题；

(3) 所构成的主存如何与CPU连接、匹配；

① 驱动能力

② 存储器芯片与CPU的时序配合

③ 存储器的地址分配和片选译码

④ 行选信号和列选信号的产生

(4) 主存校验，如何保证存取信息的正确性。

一、主存储器的逻辑设计

1、设计主存时，需要注意两点：

① 需先明确所要求的总容量这一技术指标，总容量=编址单元数×位数。

② 需要确定可供选用的存储芯片，即什么类型、型号的存储芯片， 每片的容量是多少。每片容量通常低于总容量，就需要用若干块芯片组成。相应地，可能存在位数与字数的扩展问题。

一、主存储器的逻辑设计

a）位扩展

为了实现位扩展，各芯片的数据输入／输出线相拼接。而编址空间相同的芯片，地址线与片选信号分别相同，可将它们的地址线按位并联然后与地址总线相连，共用一个片选信号。

a）字数(编址空间)扩展

如果每片的字数（地址数量）不够，需用若干芯片组成总容量较大的存储器，称为字数扩展。高位地址译码产生若干不同片选信号选择芯片。低位地址线直接送往各芯片，以选择片内的某个单元。

一、主存储器的逻辑设计

译码选片和译码选单元关系:

假设拟修建一栋教学楼, 最多可以有64排座位,, 给定了一个6位数的编号(二进制): A_5 A_4 A_3 A_2 A_1 A_0

实际修建了4间教室, 每间8排(共32排)。要从64个编号中为这32排座位选32个号。假设选低32个编号。

第1间	第2间	第3间	第4间
000 000	001 000	010 000	011 000
000 001	001	010	001
000 010	010	010	010
.....	.....	.....	.....
000 111	111	111	111

(1) 每间教室内的坐位的编号都相同(由低3位确定)

(2) 高3位用来判断是哪一间教室。

一、主存储器的逻辑设计

存储芯片相当于教室，选片相当于选教室，由地址的高位部分译码产生片选信号；选哪一个单元相当于选哪一排坐位，由芯片接收地址的低位部分在芯片内译码来选择哪一个单元。这里的高位部分和低位部分的具体位数由芯片的容量、CPU提供的地址位数、以及拟构成的存储器的容量来决定。

因此，存储器设计需要为芯片分配地址，并根据所分配的地址来设计译码器以产生片选信号；而选择单元的译码由存储芯片内置的译码器完成，与存储器组织者无关。

2、举例

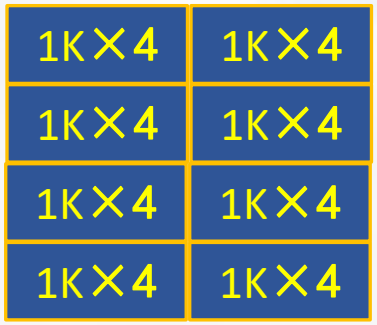
例1.用2114（1K×4）SRAM芯片组成容量为4K×8的存储器。地址总线A_{15}～A_0（低），双向数据总线D_7～D_0（低），读/写信号线R/$\overline{W}$。

给出芯片地址分配与片选逻辑,并画出M框图。

一、主存储器的逻辑设计

① 计算芯片(1K×4)数量(总容量4K×8)

a. 先扩展位数，再扩展单元数。

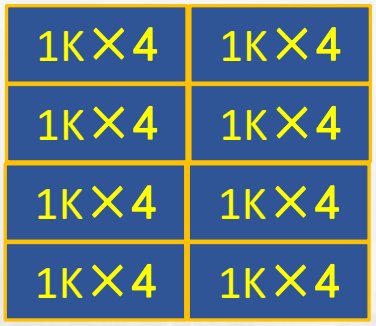


2片1K×4 ⟶ 1K×8

4组1K×8 ⟶ 4K×8

> 8片

b. 先扩展单元数，再扩展位数。



4片1K×4 ⟶ 4K×4

2组4K×4 ⟶ 4K×8

> 8片

② 地址分配与片选逻辑

存储器寻址逻辑：　　两级译码寻址系统

芯片选择（第1级）+ 芯片内寻址（第2级）

由哪几位地址形成芯片选择逻辑，以便寻找芯片

为芯片分配哪几位地址，以便寻找片内的存储单元

存储空间分配：

4KB存储器在16位地址空间（64KB）中占据任意连续区间。

（注：连续区间的起始位置的低10位为全0）

一、主存储器的逻辑设计

任意值　片选　芯片地址

A_{15}…A_{12} A_{11} A_{10} A_9……A_0

0 0 0 …… 0
<
0 0 1 …… 1

0 1 0 …… 0
<
0 1 1 …… 1

1 0 0 …… 0
<
1 0 1 …… 1

1 1 0 …… 0
<
1 1 1 …… 1

低位地址分配给芯片，高位地址形成片选逻辑。

一、主存储器的逻辑设计

低位地址分配给芯片，高位地址形成片选逻辑。

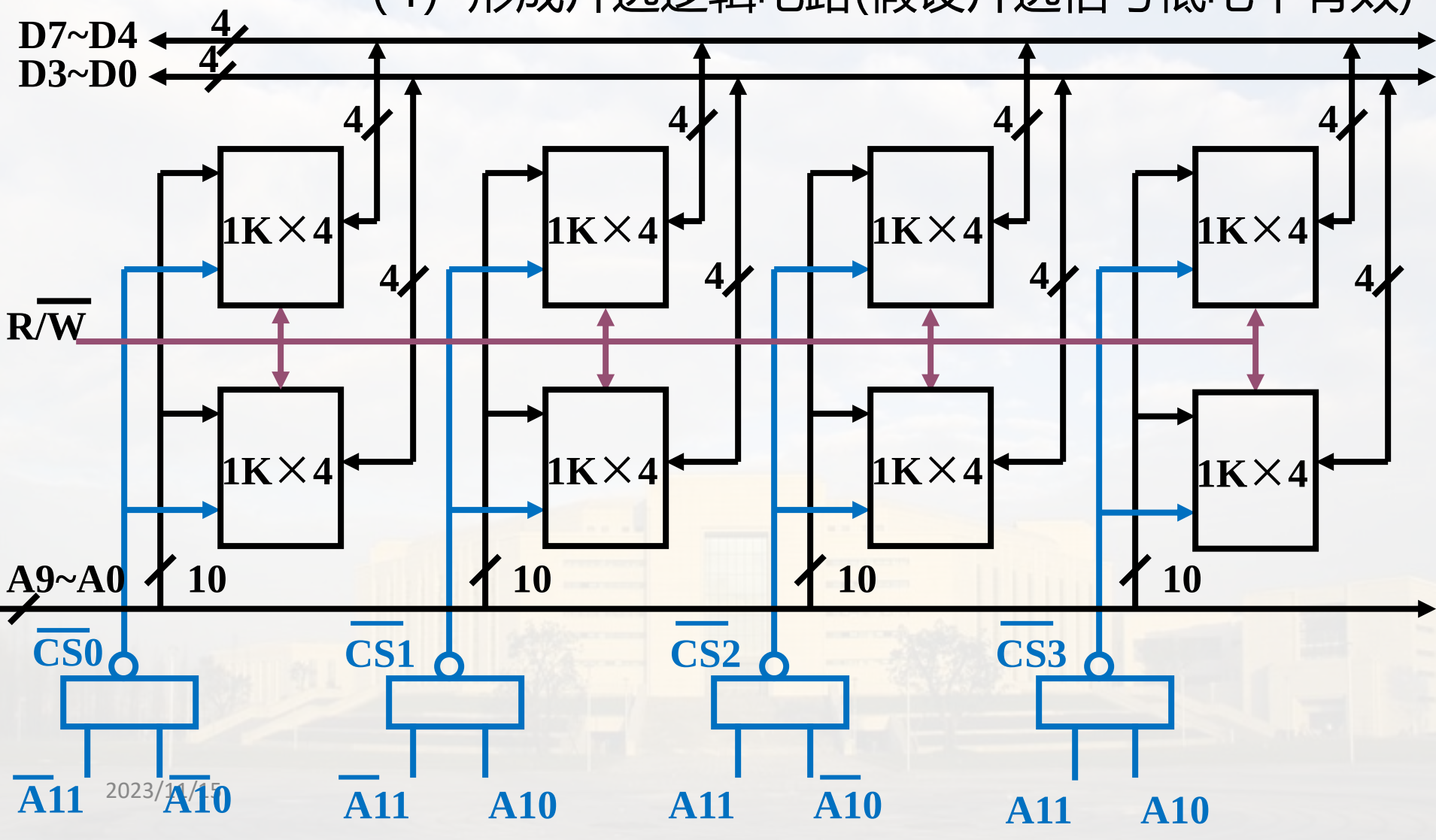
芯片	芯片地址	片选信号	片选逻辑
1K	A9～A0	$\overline{CS0}$	$\overline{\overline{A11} \cdot \overline{A10}}$
1K	A9～A0	$\overline{CS1}$	$\overline{\overline{A11} \cdot A10}$
1K	A9～A0	$\overline{CS2}$	$\overline{A11 \cdot \overline{A10}}$
1K	A9～A0	$\overline{CS3}$	$\overline{A11 \cdot A10}$

一、主存储器的逻辑设计

③连接方式

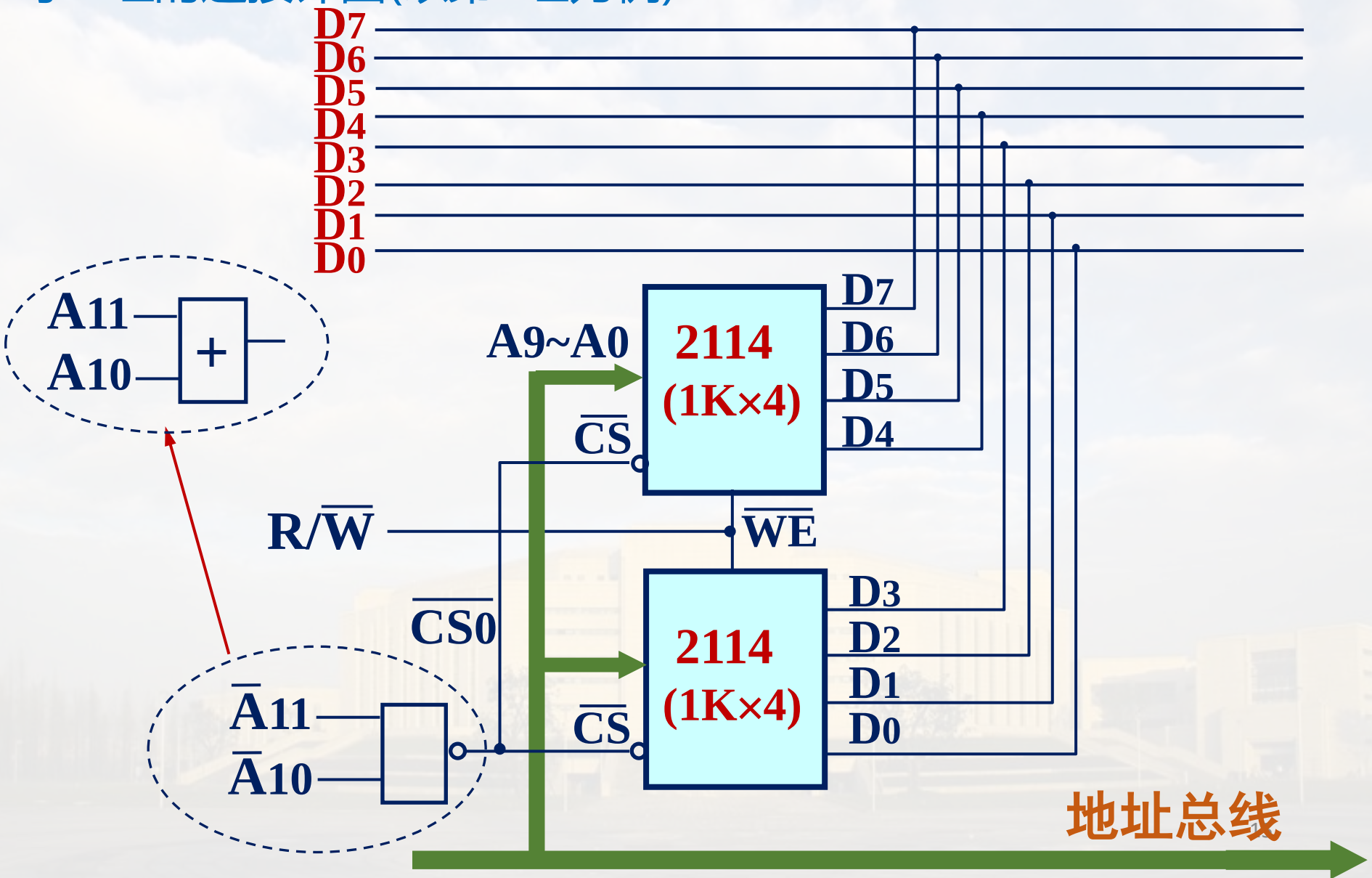
（1）扩展位数（2）扩展单元数（3）连接控制线

（4）形成片选逻辑电路(假设片选信号低电平有效)



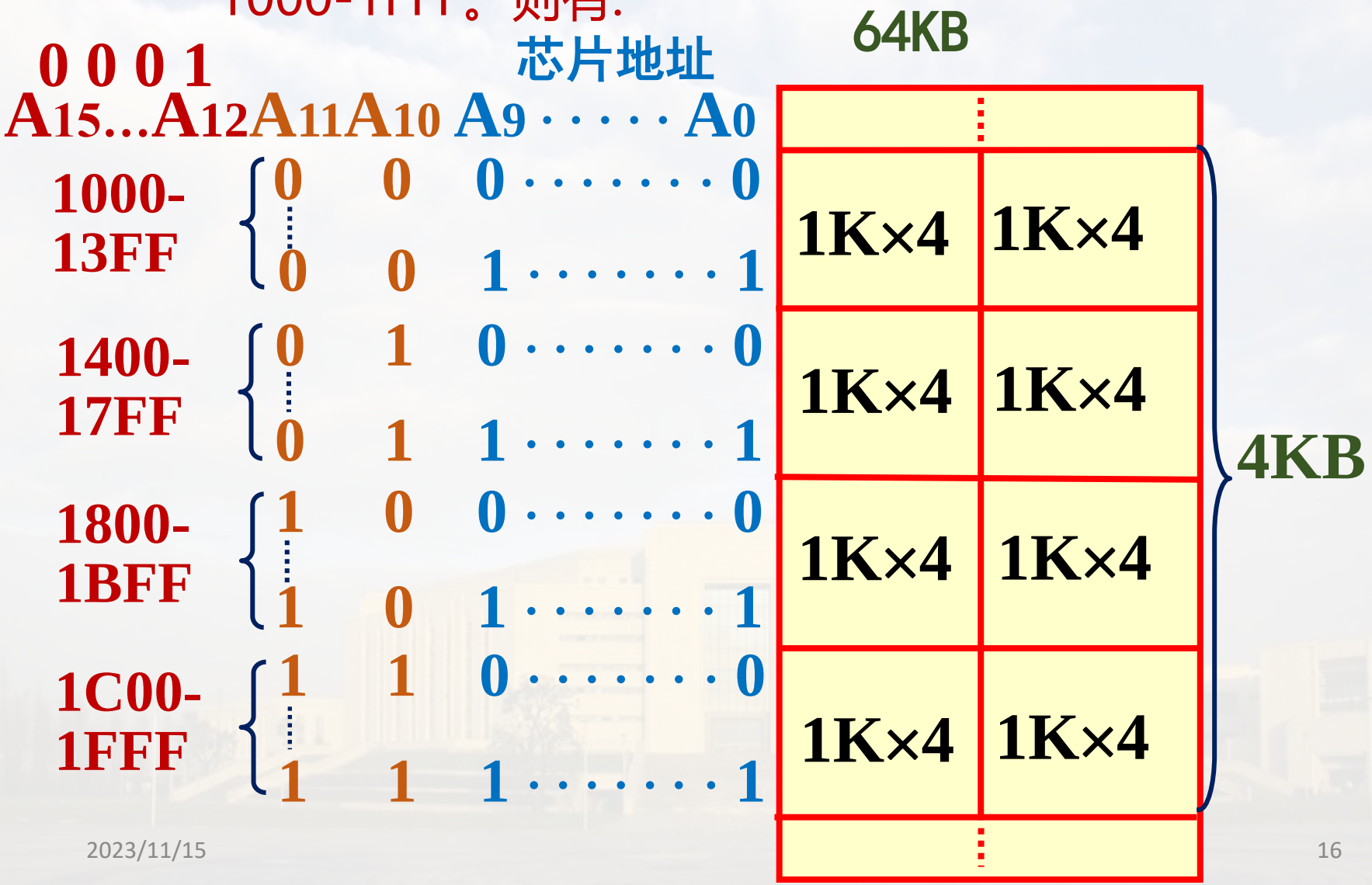
一、主存储器的逻辑设计

每一组的连接详图(以第0组为例):



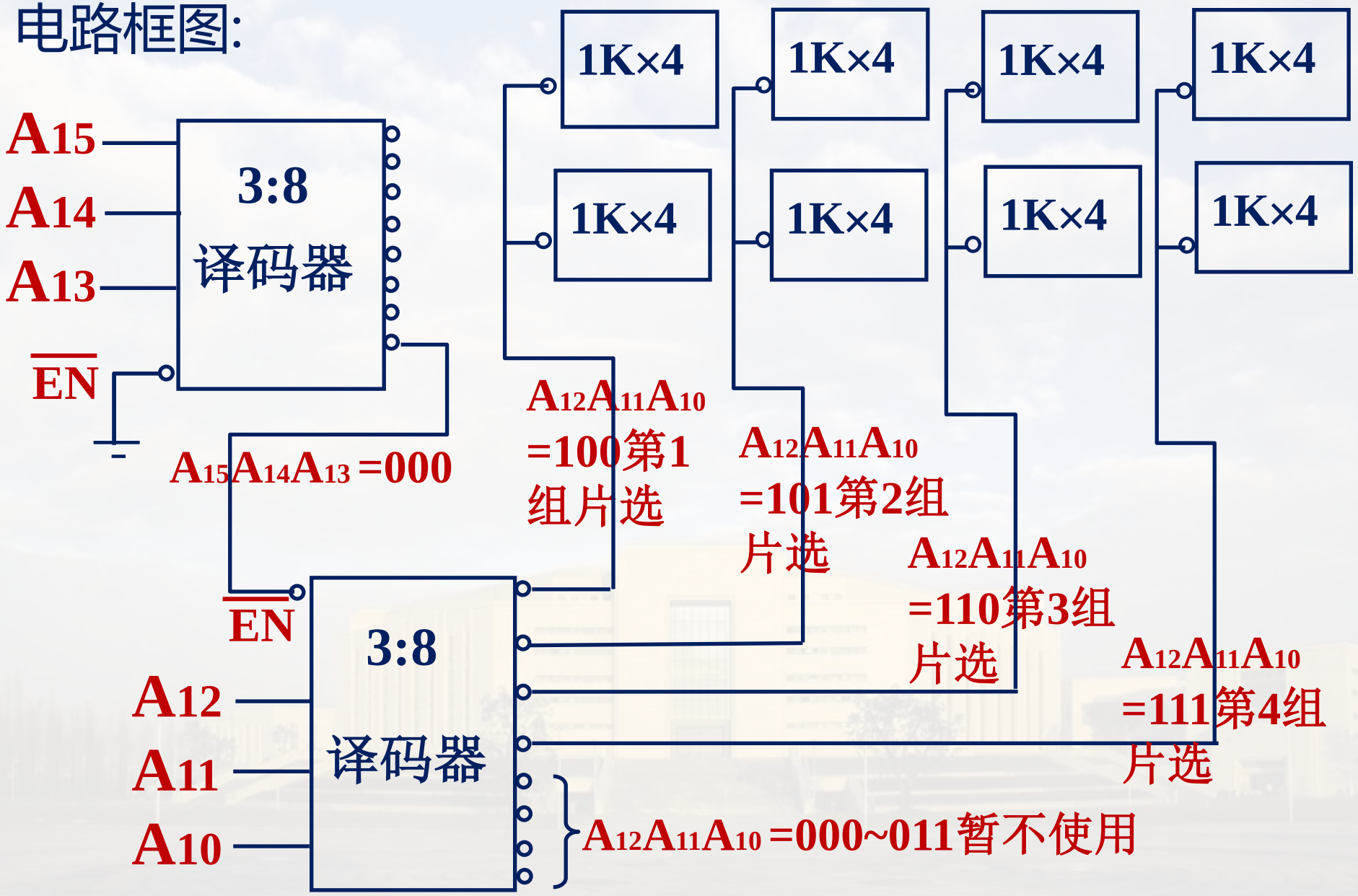
一、主存储器的逻辑设计

再假设: 上述4KB不是占据任意连续区间, 而是分配一个固定空间1000-1FFF。则有:



一、主存储器的逻辑设计

电路框图:



一、主存储器的逻辑设计

例2.某半导体存储器，按字节编址。其中，0000H～07FFH为ROM区，选用EPROM芯片（2KB/片）；0800H～13FFH为RAM区，选用RAM芯片（2KB/片和1KB/片）。地址总线A15～A0（低）。给出地址分配和片选逻辑。

① 计算容量和芯片数

ROM区：2KB　　RAM区：3KB　　共3片

② 地址分配与片选逻辑

存储空间分配：　先安排大容量芯片（放地址低端），再安排小容量芯片。

便于拟定片选逻辑。

一、主存储器的逻辑设计

$A_{15}A_{14}A_{13}A_{12}A_{11}A_{10}A_9\cdots\cdots A_0$

0000H	0	0	0	0	0	0	0	……0
07FFH	0	0	0	0	0	1	1	……1
0800H	0	0	0	0	1	0	0	……0
0FFFH	0	0	0	0	1	1	1	……1
1000H	0	0	0	1	0	0	0	……0
13FFH	0	0	0	1	0	0	1	……1

一、主存储器的逻辑设计

低位地址分配给芯片，高位地址形成片选逻辑。

芯片	芯片地址	片选信号	片选逻辑
2K	$A_{10}\sim A_0$	$\overline{CS0}$	$\overline{\overline{A_{15}}\cdot\overline{A_{14}}\cdot\overline{A_{13}}\cdot\overline{A_{12}}\cdot\overline{A_{11}}}$
2K	$A_{10}\sim A_0$	$\overline{CS1}$	$\overline{\overline{A_{15}}\cdot\overline{A_{14}}\cdot\overline{A_{13}}\cdot\overline{A_{12}}\cdot A_{11}}$
1K	$A_9\sim A_0$	$\overline{CS2}$	$\overline{\overline{A_{15}}\cdot\overline{A_{14}}\cdot\overline{A_{13}}\cdot A_{10}\cdot\overline{A_{11}}\cdot\overline{A_{10}}}$

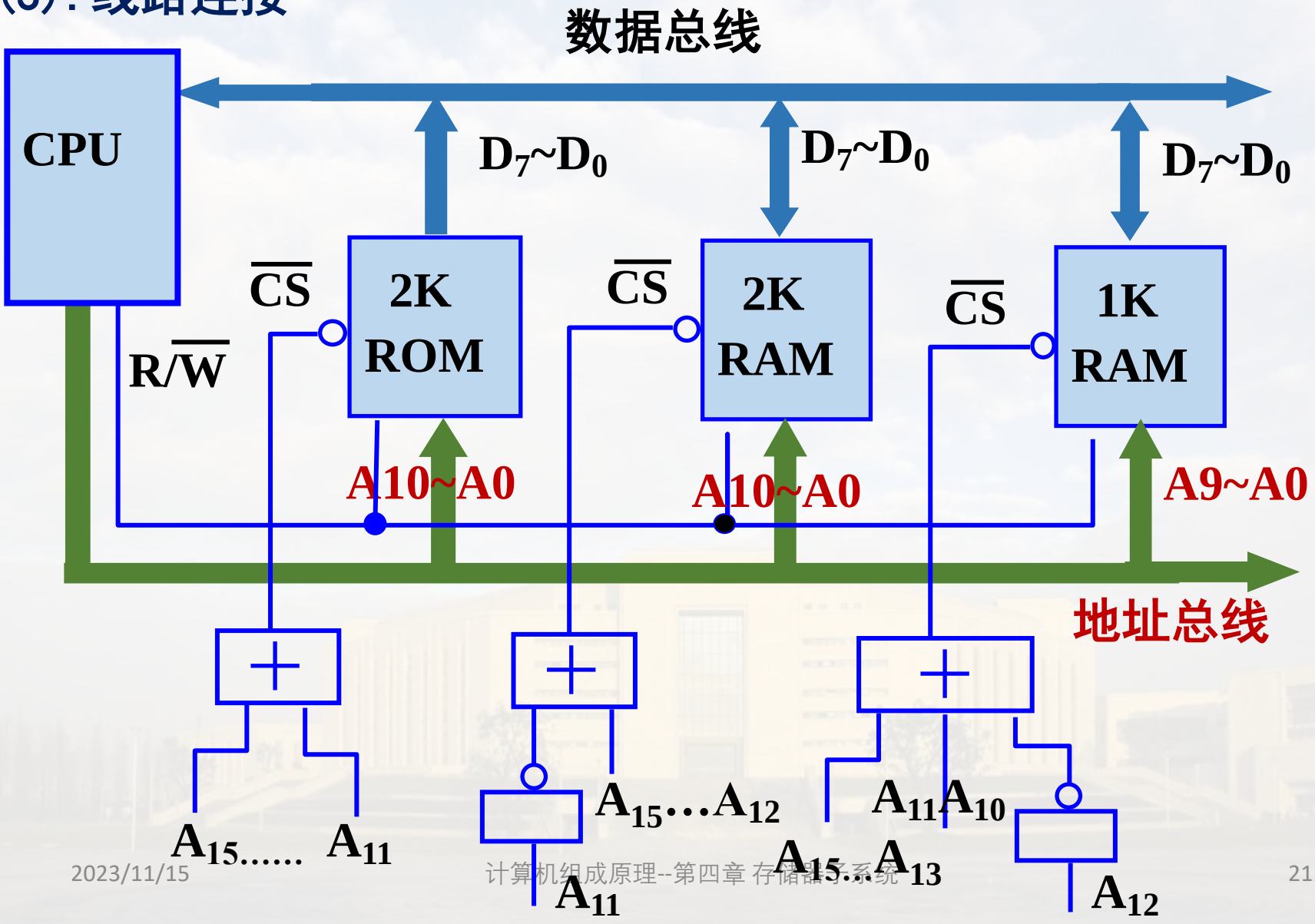
	$A_{15}A_{14}A_{13}A_{12}A_{11}A_{10}A_9\cdots\cdots A_0$
0000H	0 0 0 0 0 0 0 ……0
07FFH	0 0 0 0 0 1 1 ……1
0800H	0 0 0 0 1 0 0 ……0
0FFFH	0 0 0 0 1 1 1 ……1
1000H	0 0 0 1 0 0 0 ……0
13FFH	0 0 0 1 0 0 1 ……1

化简可得

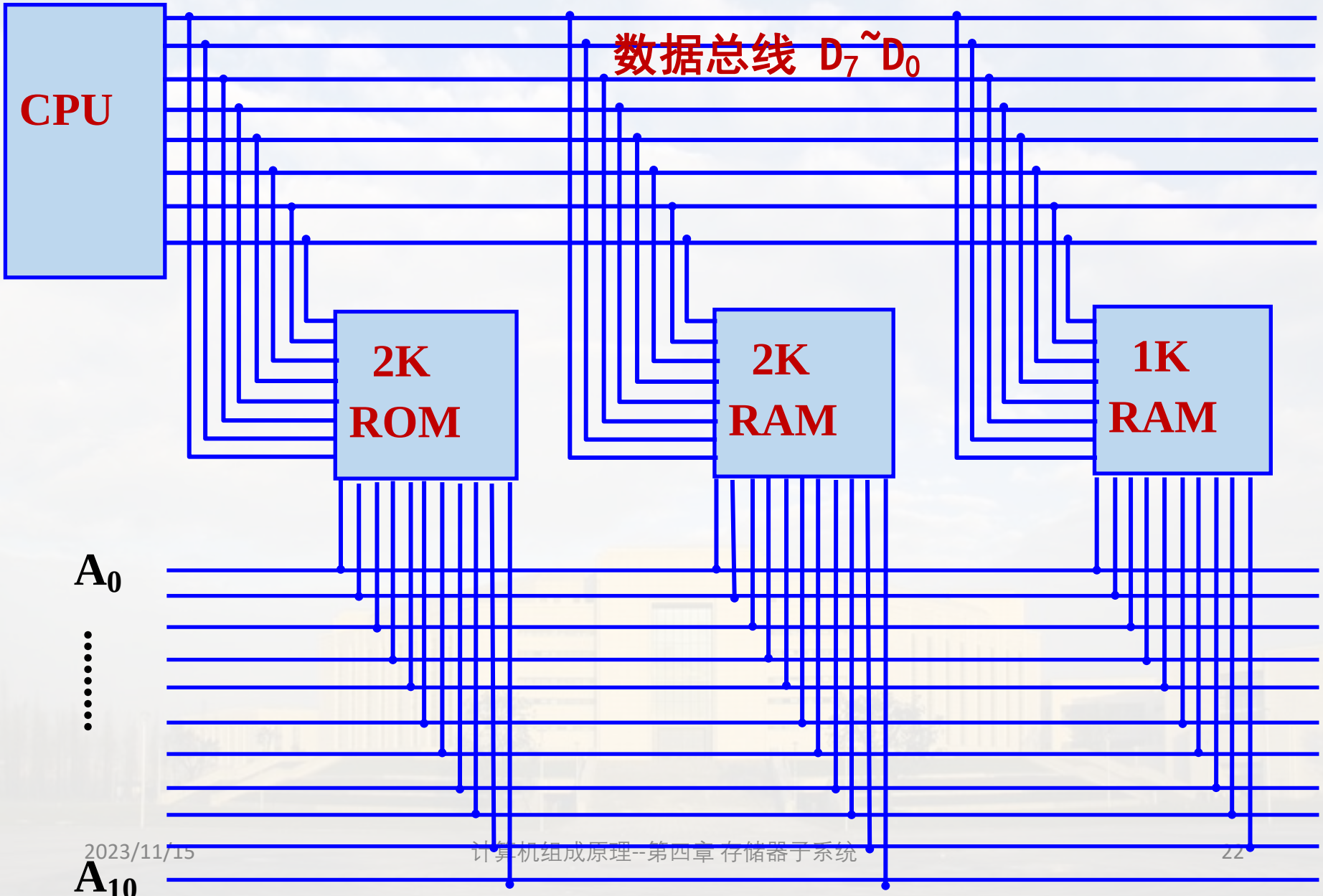
$$\overline{\overline{A_{15}}\cdot\overline{A_{14}}\cdot\overline{A_{13}}\cdot\overline{A_{12}}\cdot\overline{A_{11}}}=A_{15}+A_{14}+A_{13}+A_{12}+A_{11}$$

一、主存储器的逻辑设计

(3). 线路连接



一、主存储器的逻辑设计



3、地址译码方法：全译码、部分译码和线译码

例：用2114(1K×4)SRAM芯片组成2K×8的存储器。地址总线A15~A0, 双向数据总线D7~D0。

- 所需芯片数量: 4片
- 假设分配地址范围: 1000H ~ 17FFH

	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
第一组	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1
第二组	0	0	0	1	0	1	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

全译码　部分译码　线译码　低10位地址直接与芯片相连

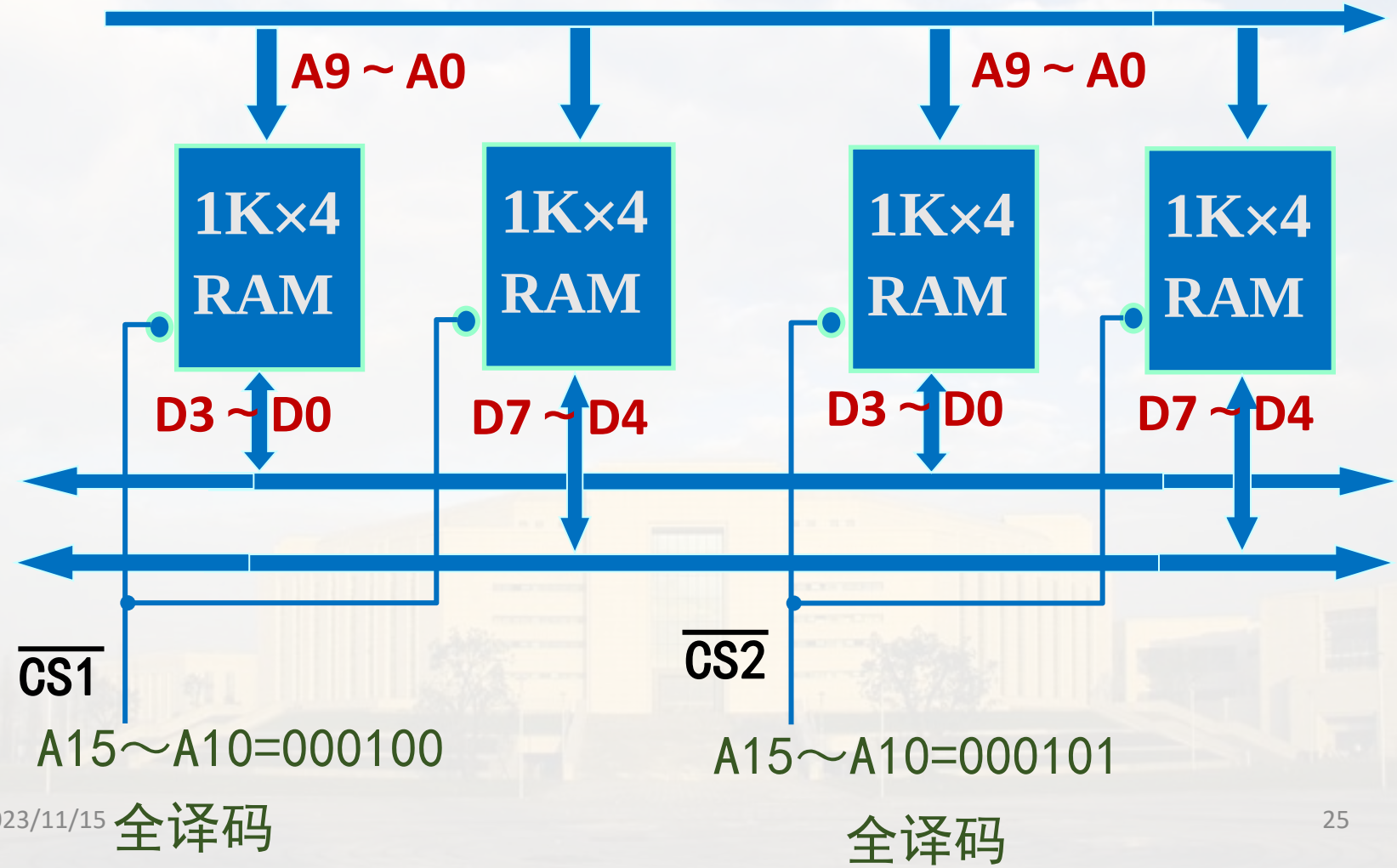
(1)全地址译码方式

就是构成存储器时要使用全部地址总线信号，即所有的高位地址信号都用来作为译码器的输入，低位地址信号接存储芯片的地址输入线，从而使存储器芯片上的每一个单元在整个内存空间中具有唯一的地址。

	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
第一组	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1
第二组	0	0	0	1	0	1	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

全译码　部分译码　线译码　低10位地址直接与芯片相连

全译码: 将除了与芯片连接的地址以外的所有高位地址用于译码产生片选信号,



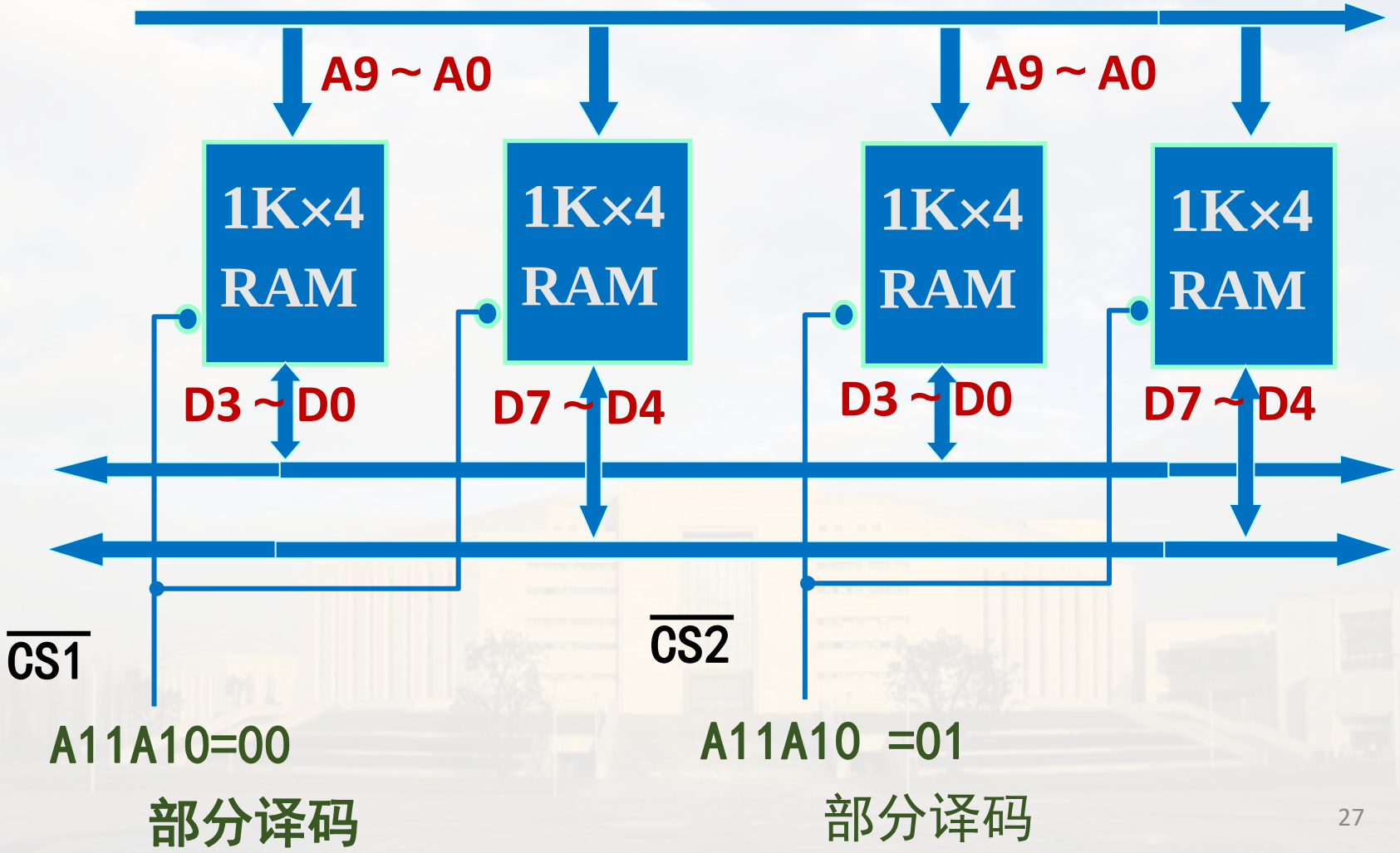
(2)部分地址译码方式(无范围: 1000H ~ 17FFH假设)

就是仅把地址总线的一部分地址信号线与存储器连接，通常是用高位地址信号的一部分（而不是全部）作为片选译码信号；低位地址信号接存储芯片的地址输入线。

	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
第一组	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1
第二组	0	0	0	1	0	1	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

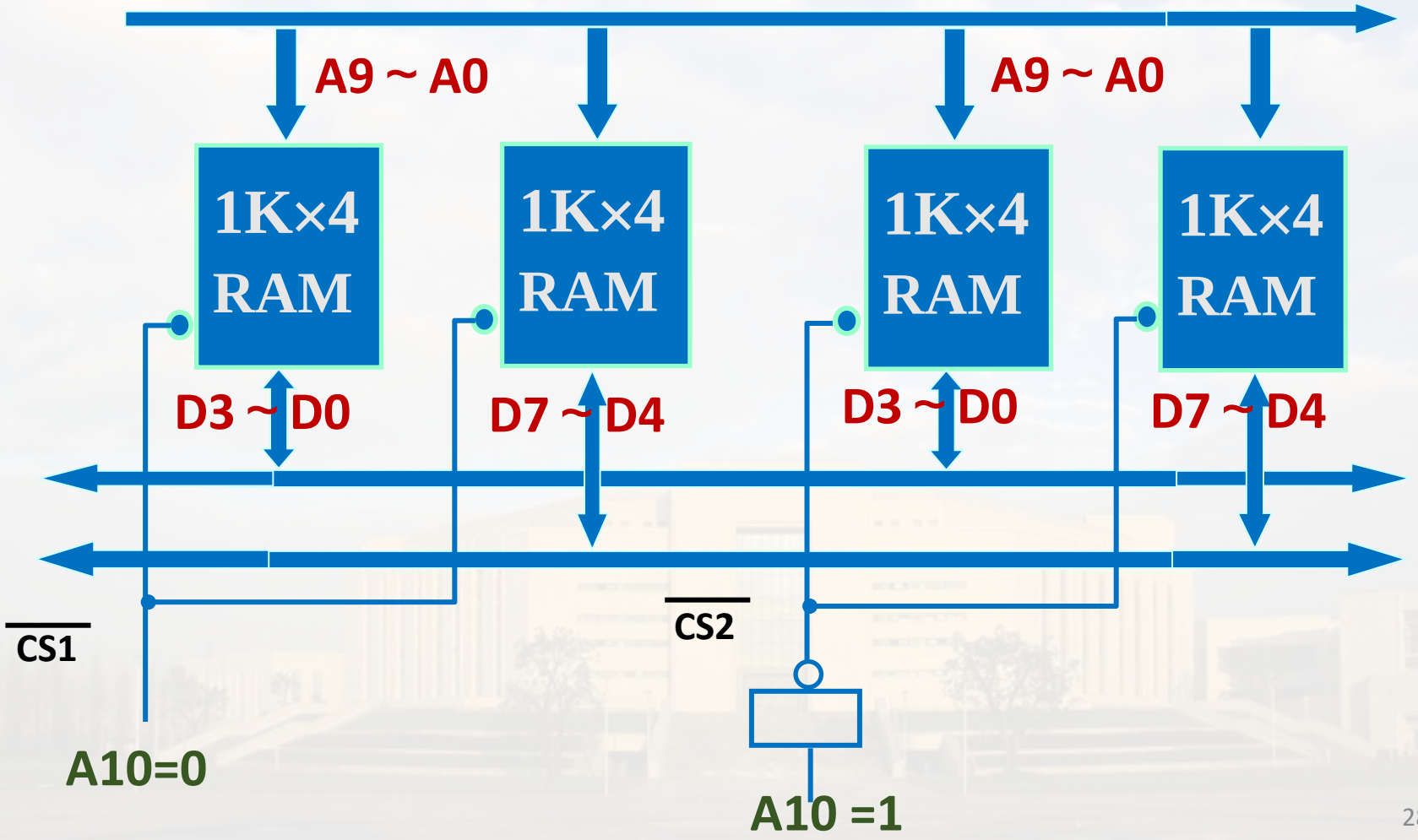
全译码　部分译码　线译码　低10位地址直接与芯片相连

部分译码: 将除了与芯片连接的地址以外的部分高位地址用于译码产生片选信号



二、基本逻辑门及译码器

线译码: 直接用一条可以区分两组地址范围的地址线的高低电平作为片选信号



(3)全地址译码、部分地址译码特点

① 部分地址译码使地址出现重叠区，而重叠区的部分必须空着不准使用，这就破坏了地址空间的连续性，也在实际上减少了总的可用存储地址空间。其优点是其译码器的构成比较简单，成本较低。

(3)全地址译码、部分地址译码特点

② 全地址译码使存储器芯片上的每一个单元在整个内存空间中具有唯一的地址。

③ 在实际应用中，采用全地址译码还是部分地址译码应根据具体情况来定。如果地址资源很富余，为使电路简单可考虑用部分地址译码；如果要充分利用地址空间，则应采用全地址译码。

三种译码方式的应用场合:

- 所设计的存储器达到(或接近达到)CPU提供的全部存储空间时，必须用全译码，且任何时候都可以使用全译码方式
- 所设计的存储器未达到CPU提供的全部存储空间时，可以用部分译码或线译码。
- 可采用全译码与部分译码相结合的方式，即部分芯片用全译码，另一些芯片采用部分译码。

例3. 用1K×8的存储芯片构成6K×8的存储器。CPU地址总线16条, 数据总线D_7~D_0。存储器的地址空间为从2000H开始的连续6K地址空间。

- 需要的芯片数: 6片
- 地址空间分配: ($8K=2^{13}$)

  2000H~23FFH, 2400H~27FFH
  2800H~2BFFH, 2C00H~2FFFH
  3000H~33FFH, 3400H~37FFH

- 写出二进制代码:

二、基本逻辑门及译码器

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1
0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	1	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
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1
0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1

二、基本逻辑门及译码器

- 用于连接芯片的地址线为 A9 ~A0
- A10以上地址线用于译码产生片选信号
- 任何一个区域的A_{15} A_{14} A_{13}= 001
- A_{12} $A_{11}A_{10}$体现不同区域的地址上的区别

0	0	0	⟶	第一区
0	0	1	⟶	第二区
0	1	0	⟶	第三区
0	1	1	⟶	第四区
1	0	0	⟶	第五区
1	0	1	⟶	第六区

- 采用的译码方式

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1
0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	1	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
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1
0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1

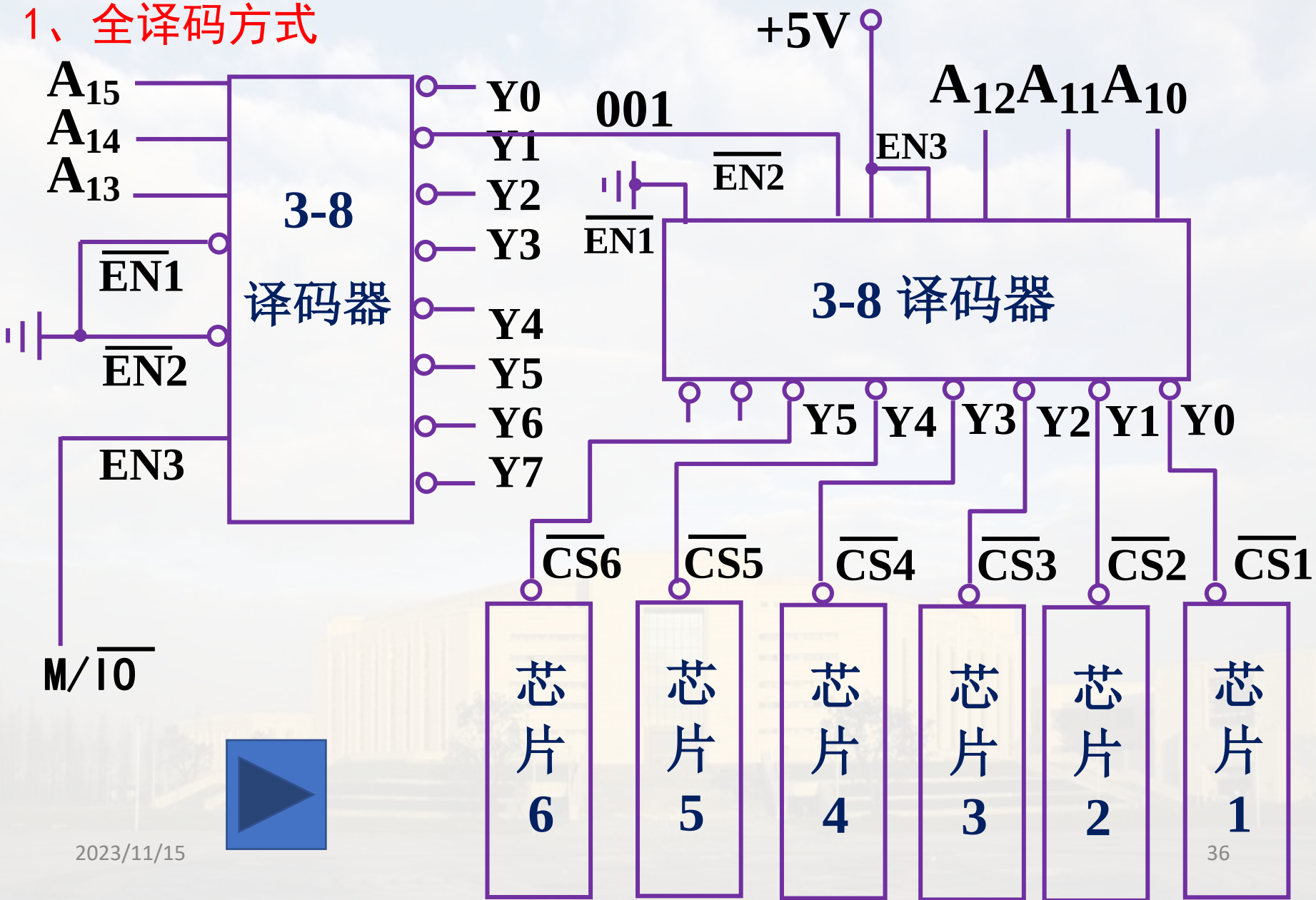
3线—8线译码器74LS138(T4138)的真值表如下：

回顾

EN_3	$\overline{EN_1}+\overline{EN_2}$	A_2	A_1	A_0	$\overline{Y_0}$	$\overline{Y_1}$	$\overline{Y_2}$	$\overline{Y_3}$	$\overline{Y_4}$	$\overline{Y_5}$	$\overline{Y_6}$	$\overline{Y_7}$
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0
0	d	d	d	d	1	1	1	1	1	1	1	1
d	1	d	d	d	1	1	1	1	1	1	1	1

二、基本逻辑门及译码器

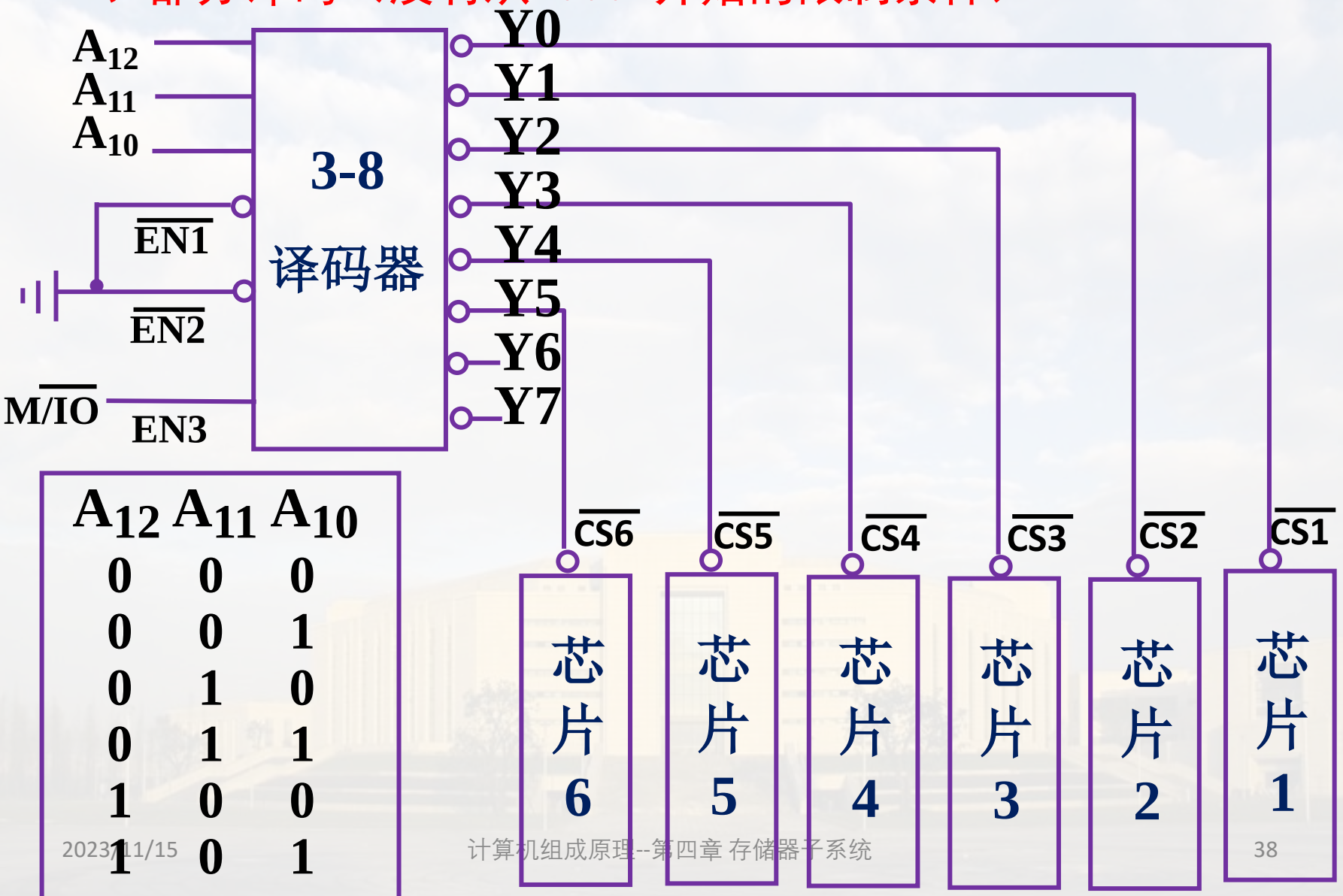
1、全译码方式



二、基本逻辑门及译码器

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1
0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	1	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
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1
0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1

2、部分译码（没有从2000H开始的限制条件）



A_{12}	A_{11}	A_{10}
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1

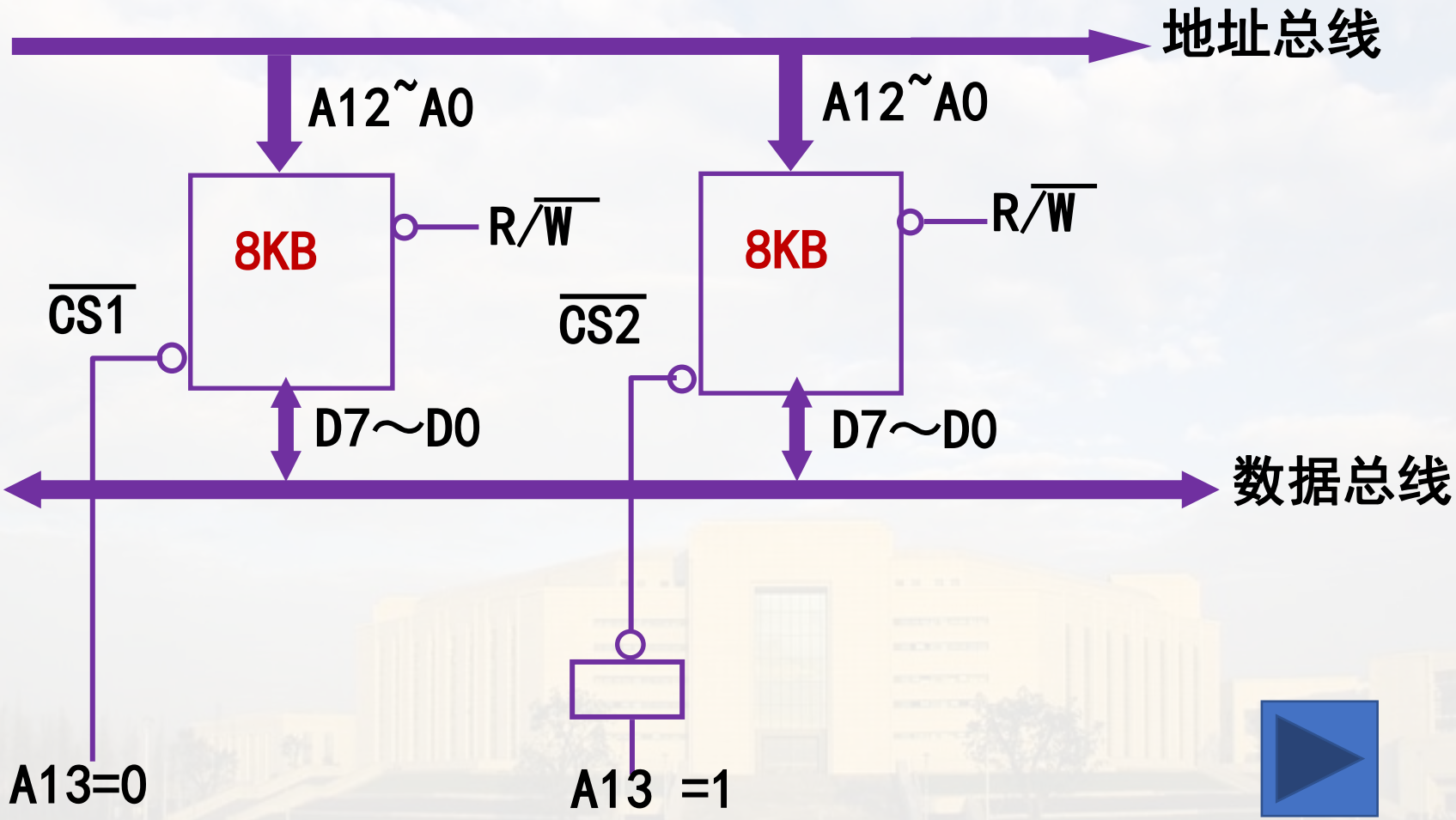
例4: 用两片8KB的SRAM芯片组成16KB的存储器。地址总线 A_{15}~A_0，数据总线D_7~D_0。

- 采用的地址分配：假设分配从0000H开始的16K地址空间。0000H~1FFFH 和 2000H~3FFFH

	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
第一片	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
第二片	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

二、基本逻辑门及译码器

- 可以采用线译码方式（无起始单元假设）：



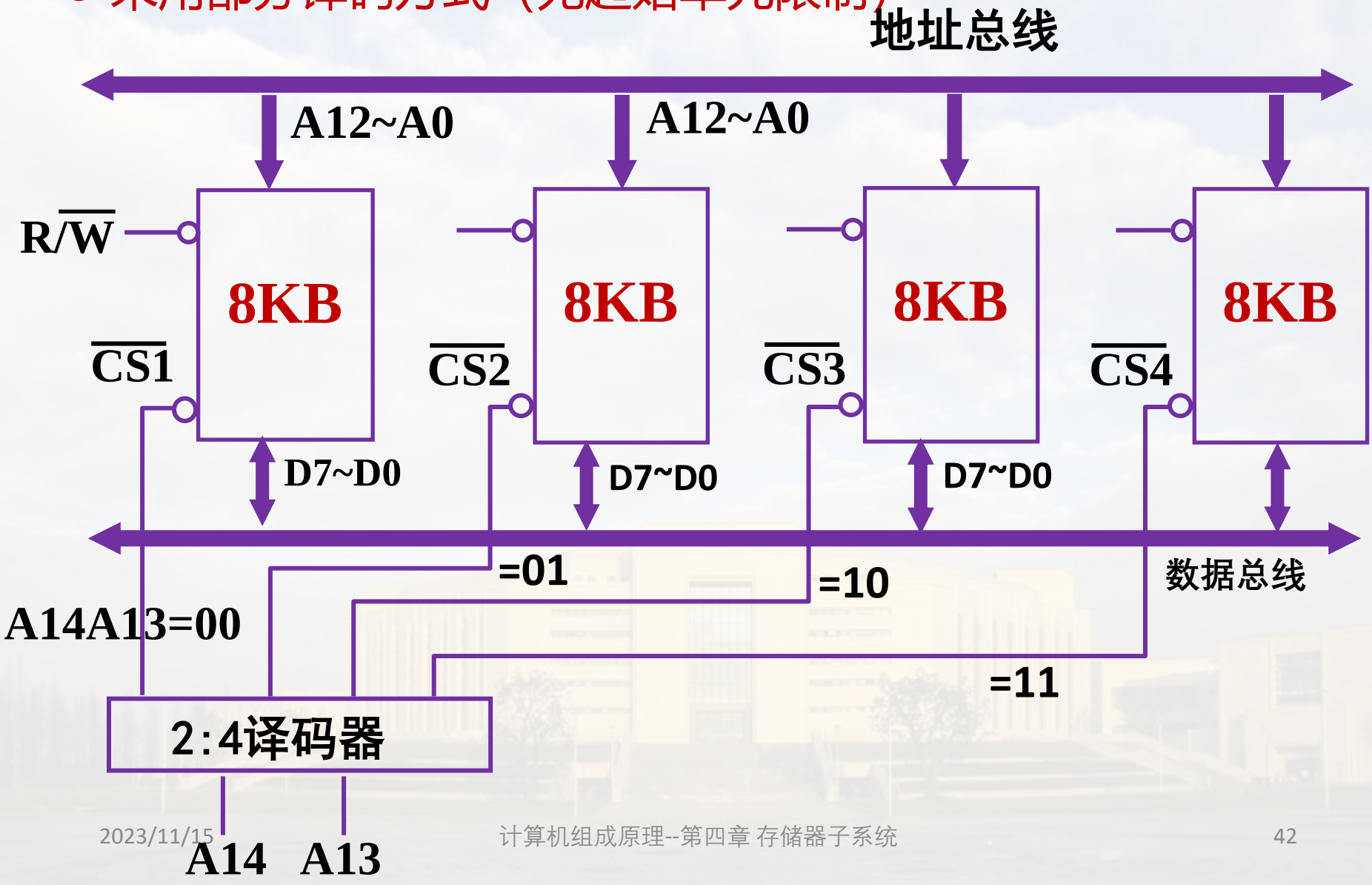
— 假设再扩展两片8KB的容量，地址分配如下：

0000H~1FFFH 和 2000H~3FFFH

4000H~5FFFH 和 6000H~7FFFH

	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
第一片	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
第二片	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
第三片	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
第四片	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

- 采用部分译码方式（无起始单元限制）



— 假设在上述四片的基础上，再扩展两片8KB的容量

作以下地址分配:

8000H~9FFFH 和 A000H~BFFFH

连同已有的四片芯片，地址分配情况是:

0000H~1FFFH 和 2000H~3FFFH

4000H~5FFFH 和 6000H~7FFFH

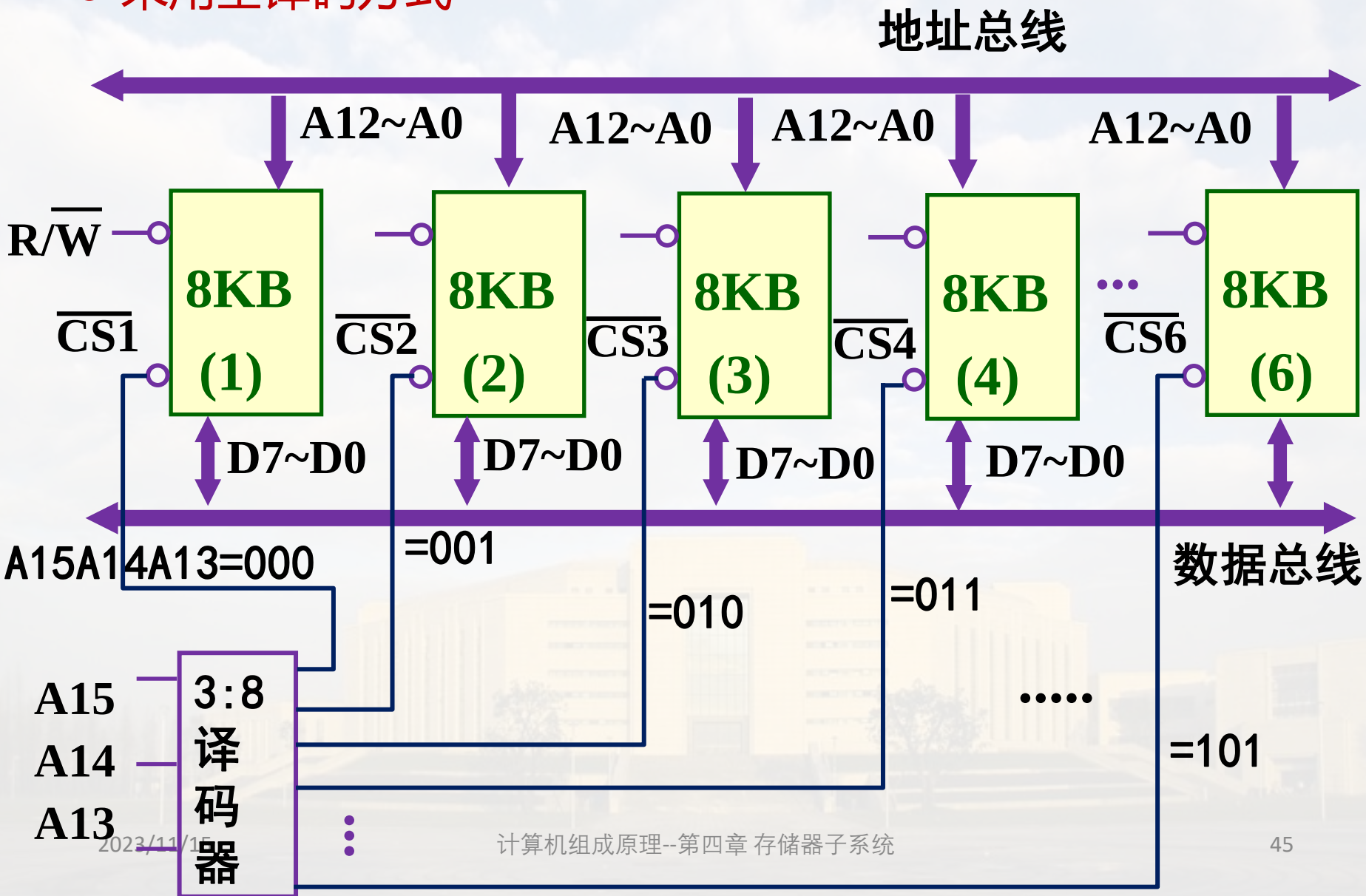
8000H~9FFFH 和 A000H~BFFFH

二、基本逻辑门及译码器

	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
第一片	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
第二片	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
第三片	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
第四片	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
第五片	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
第六片	1	0	1	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	1

二、基本逻辑门及译码器

- 采用全译码方式



小结：

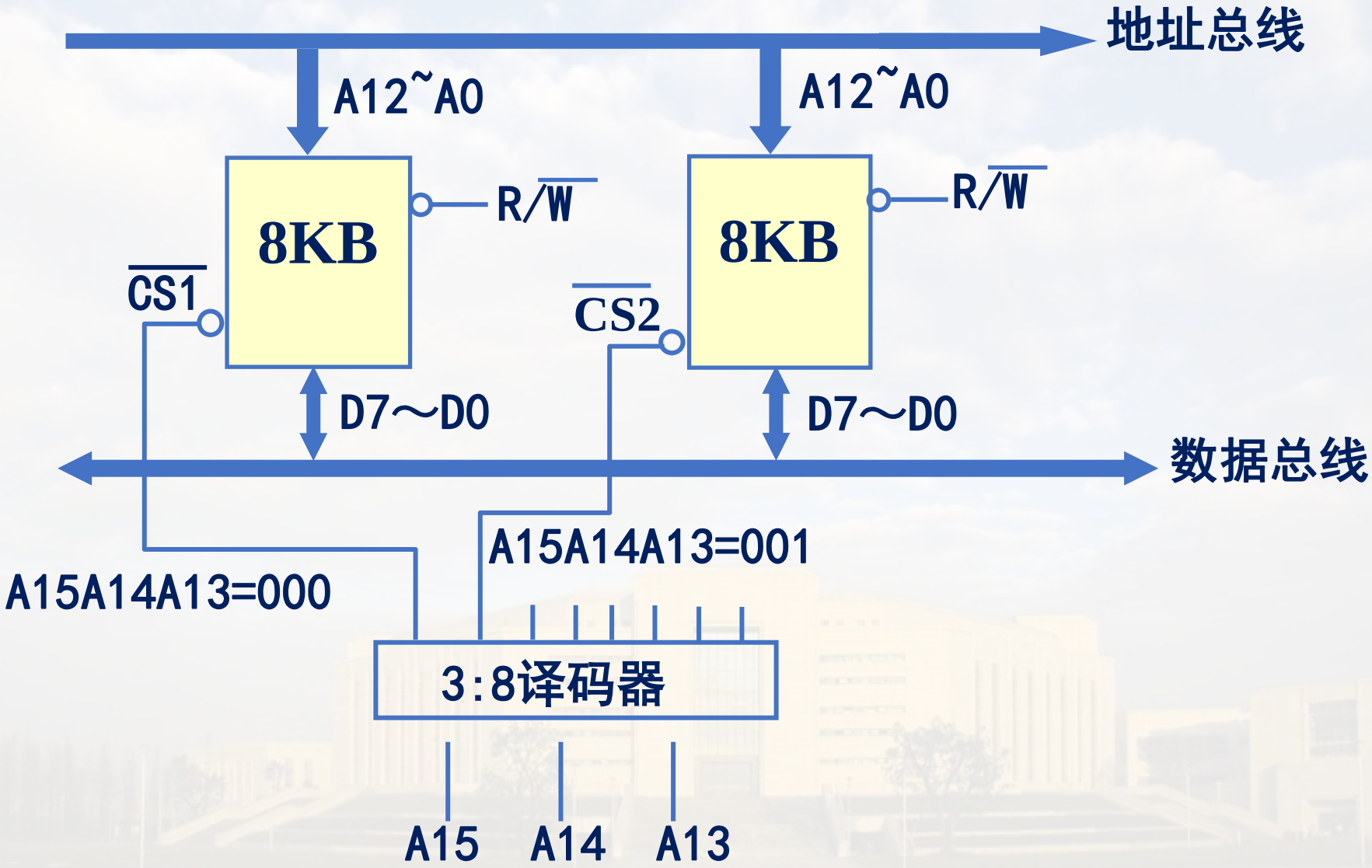
(1) 采用线译码和部分译码方式，译码线路简单，但是可扩展性差；

(2) 采用哪一种译码方式取决于所设计的存储系统的容量；

(3) 任何容量的存储器系统都可以采用全译码方式，可扩展性好。

例：仅有两片8KB的芯片时，采用全译码方式：

二、基本逻辑门及译码器


地址总线
A12~A0
A12~A0
8KB
8KB
R/W
R/W
CS1
CS2
D7～D0
D7～D0
数据总线
A15A14A13=001
A15A14A13=000
3:8译码器
A15
A14
A13

例5. 用16KB芯片1片、8KB芯片3片、4KB芯片4片、2KB芯片3片、1KB芯片1片，构成一个63KB的存储器，地址总线16条，数据总线8位。

- 存储芯片总数： 12片
- 地址空间分配（从高地址单元起，连续分配，大容量芯片在高地址单元）：

16K×8：（1片）
C000H ~ FFFFH

8K×8：（3片）
A000H~BFFFH
8000H~9FFFH
6000H~7FFFH

4K×8：（4片）
5000H ~ 5FFFH
4000H ~ 4FFFH
3000H ~ 3FFFH
2000H ~ 2FFFH

2K×8：（3片）
1800H ~ 1FFFH
1000H ~ 17FFH
0800H ~ 0FFFH

1K×8：（1片）
0400H ~ 07FFH

空闲：0000H ~ 03FFH

- 写出地址分配的二进制代码：


A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
16K
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 1
C000H~
FFFFH
8K
1 0 1 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 1
A000H~BFFFH
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1
8000H~9FFFH
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
6000H~7FFFH
4K
0 1 0 1 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
5000H~5FFFH
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1
4000H~4FFFH
0 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
3000H~3FFFH
0 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
2000H~2FFFH
2K
0 0 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
1800H~1FFFH
0 0 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
1000H~17FFH
0 0 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
0800H~0FFFH
1K
0 0 0 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
0400H~
07FFH
2008/11/9
计算机组成原理-第四章 存储器子系统
79

片选逻辑：

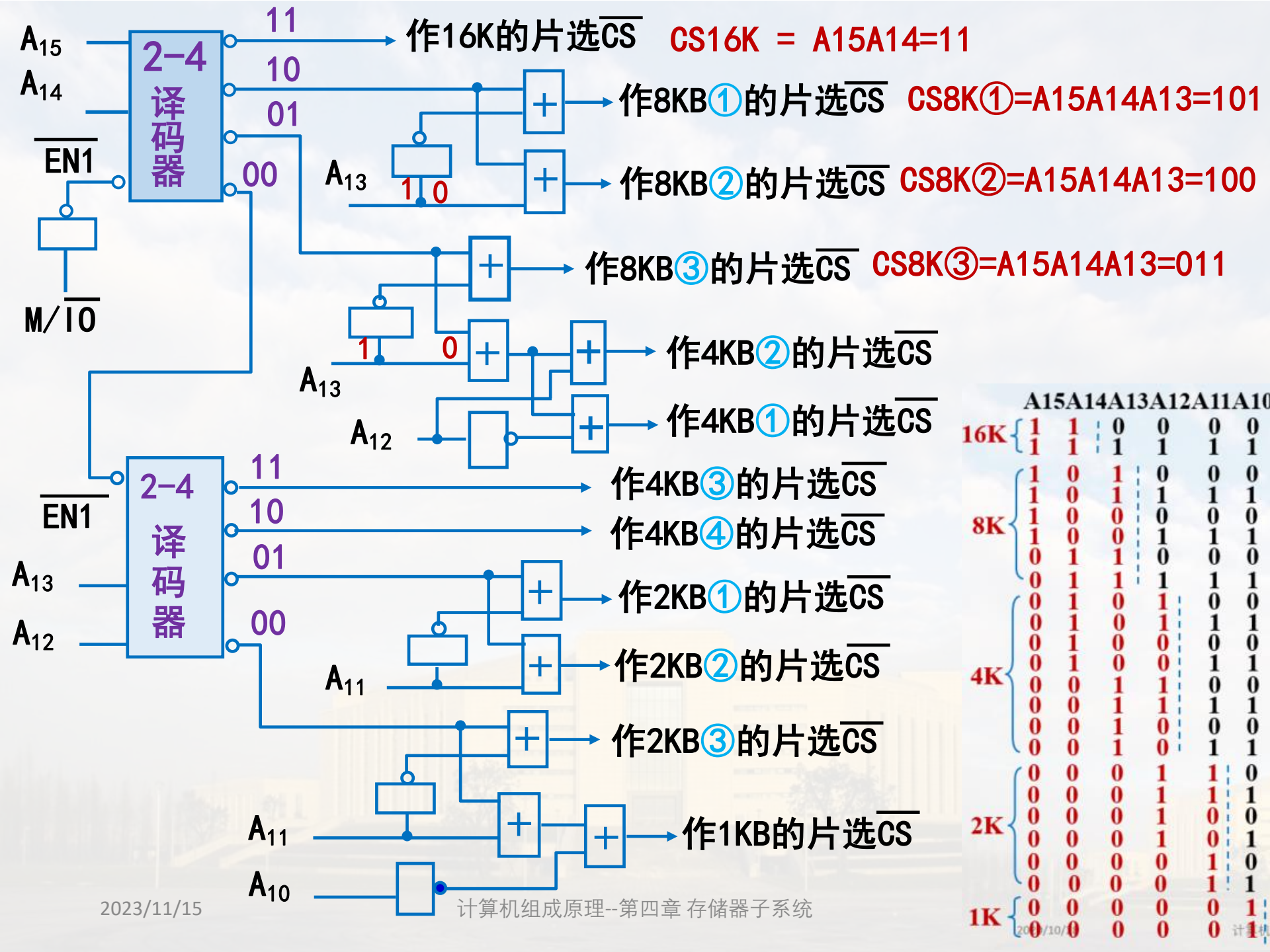
16K
- $CS16K = A_{15}A_{14}$ (=11)

8K
- $CS8K① = A_{15}\overline{A}_{14}A_{13}$ (=101)
- $CS8K② = A_{15}\overline{A}_{14}\overline{A}_{13}$ (=100)
- $CS8K③ = \overline{A}_{15}A_{14}A_{13}$ (=011)

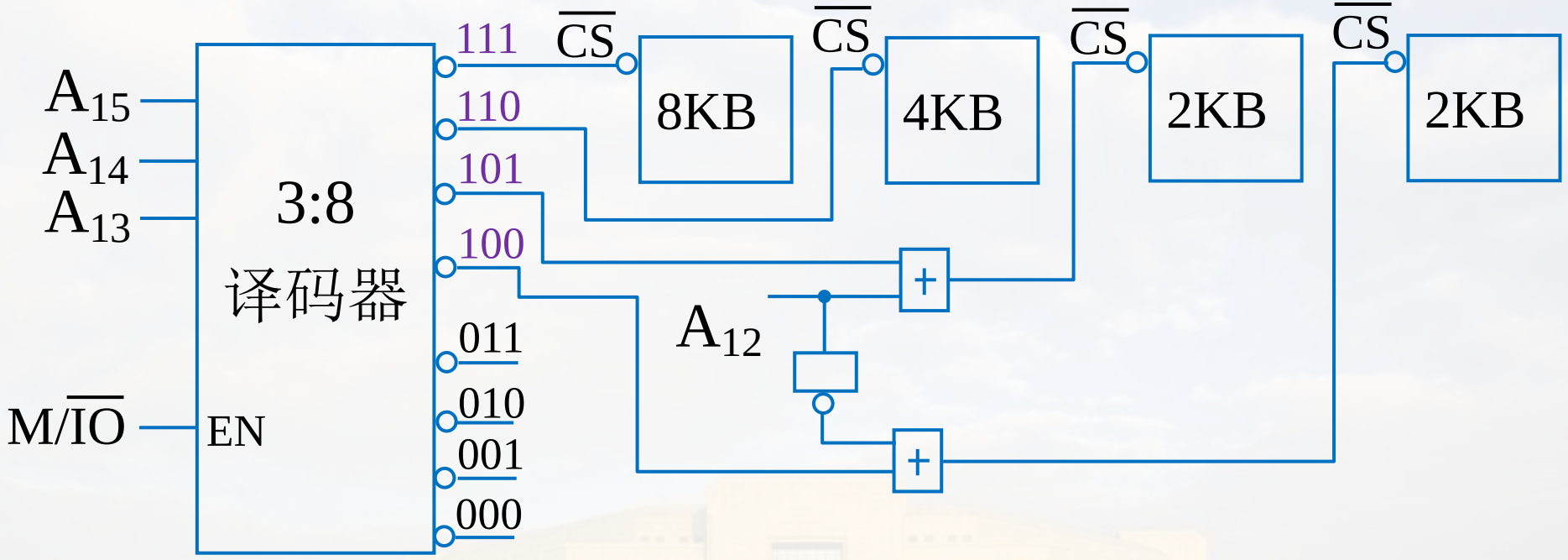
4K
- $CS4K① = \overline{A}_{15}A_{14}\overline{A}_{13}A_{12}$ (=0101)
- $CS4K② = \overline{A}_{15}A_{14}\overline{A}_{13}\overline{A}_{12}$ (=0100)
- $CS4K③ = \overline{A}_{15}\overline{A}_{14}A_{13}A_{12}$ (=0011)
- $CS4K④ = \overline{A}_{15}\overline{A}_{14}A_{13}\overline{A}_{12}$ (=0010)

2K
- $CS2K① = \overline{A}_{15}\overline{A}_{14}\overline{A}_{13}A_{12}A_{11}$ (=00011)
- $CS2K② = \overline{A}_{15}\overline{A}_{14}\overline{A}_{13}A_{12}\overline{A}_{11}$ (=00010)
- $CS2K③ = \overline{A}_{15}\overline{A}_{14}A_{13}\overline{A}_{12}A_{11}$ (=00001)

1K
- $CS1K = \overline{A}_{15}\overline{A}_{14}\overline{A}_{13}\overline{A}_{12}\overline{A}_{11}A_{10}$ (=000001)


A15
A14
2-4 译码器
EN1
M/IO
11
10
01
00
作16K的片选CS
CS16K = A15A14=11
作8KB①的片选CS
CS8K①=A15A14A13=101
作8KB②的片选CS
CS8K②=A15A14A13=100
作8KB③的片选CS
CS8K③=A15A14A13=011
A13
1
0
作4KB②的片选CS
A12
作4KB①的片选CS
EN1
2-4 译码器
A13
A12
作4KB③的片选CS
作4KB④的片选CS
作2KB①的片选CS
A11
作2KB②的片选CS
作2KB③的片选CS
A11
A10
作1KB的片选CS
A15A14A13A12A11A10
16K
8K
4K
2K
1K

例6. 假设CPU地址总线16条, 数据总线8条, 有以下存储器连接电路:



1. 写出每片芯片的地址范围;
2. 分析芯片地址重叠情况;
3. 利用已有线路且在尽可能少地改变已有线路基础上, 增加一片16KB存储器芯片。

(1) 写出每片芯片的地址范围

$A_{15}A_{14}A_{13}A_{12}A_{11}A_{10}A_9 \ldots A_2A_1A_0$

8KB	1	1	1	0	0	0	0	....	0	0	0
	1	1	1	1	1	1	1	....	1	1	1
4BK	1	1	0	0	0	0	0	....	0	0	0
	1	1	0	1	1	1	1	....	1	1	1
2KB①	1	0	1	0	0	0	0	....	0	0	0
	1	0	1	0	1	1	1	....	1	1	1
2KB②	1	0	0	1	0	0	0	....	0	0	0
	1	0	0	1	1	1	1	....	1	1	1

E000~FFFF

C000~DFFF

A000~AFFF

9000~9FFF

4K地址范围

8K地址范围

(2) 分析芯片地址重叠情况

1) 8KB芯片地址空间E000~FFFF, 没有地址重叠;

2) 4KB芯片地址空间C000~DFFF(即两个4K空间: C000~CFFF、D000~DFFF), 有4K地址重叠;

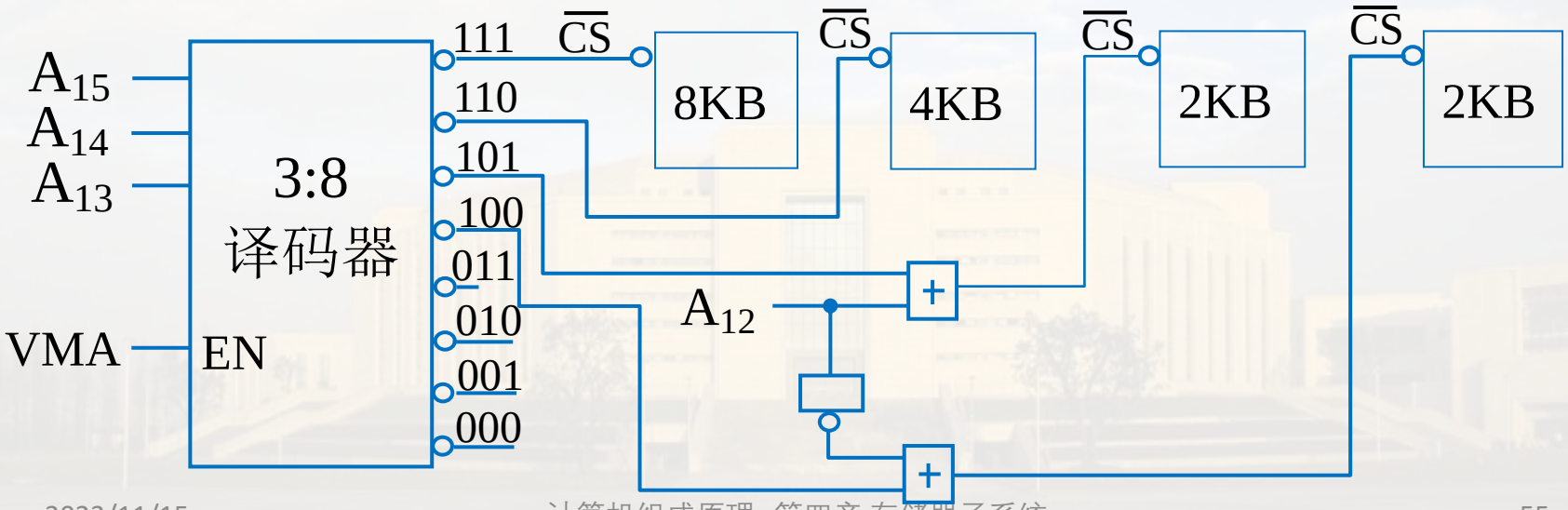
3) 2KB ①芯片地址空间A000~AFFF(即两个2K空间: A000~A7FF、A800~AFFF), 有2K地址重叠;

4) 2KB ②芯片地址空间9000~9FFF(即两个2K空间: 9000~97FF、9800~9FFF), 有2K地址重叠;

(3) 利用已有线路且在尽可能少地改变已有线路基础上，增加一片16KB存储器芯片

为尽可能少地改变已有线路基础上，利用已有3:8译码器空闲的4个输出，为新增16KB芯片分配如下地址：

	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	……	A_2	A_1	A_0
0000~	0	0	0	0	0	0	0	……	0	0	0
3FFF	0	0	1	1	1	1	1	……	1	1	1



二、基本逻辑门及译码器

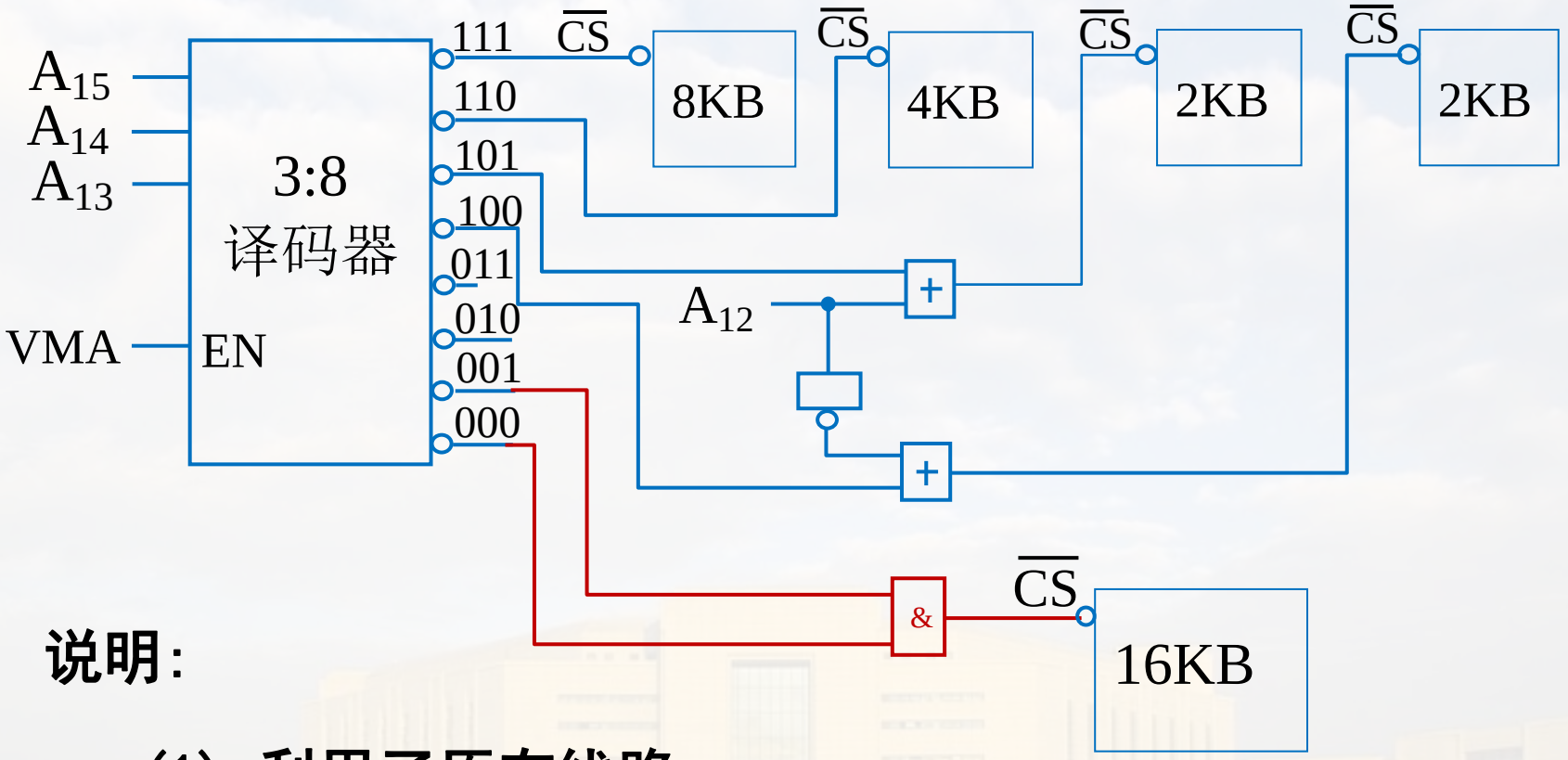
可看出, 图中3:8译码器当$A_{15}A_{14}A_{13}$为001和000是空闲输出, 每一个输出代表8K空间, 两个输出共16K范围。

可以分析这两个引脚表示的地址范围:

	$A_{15}A_{14}A_{13}A_{12}A_{11}A_{10}A_9\ldots\ldots A_2A_1A_0$	
$A_{15}A_{14}A_{13}$ =000	0 0 0 0 0 0 0 …… 0 0 0	0000H～
	0 0 0 1 1 1 1 …… 1 1 1	1FFFH
$A_{15}A_{14}A_{13}$ =001	0 0 1 0 0 0 0 …… 0 0 0	2000H～
	0 0 1 1 1 1 1 …… 1 1 1	3FFFH

总范围: 0000H~3FFFH, 即为16K芯片分配的地址范围。

因此有以下连接:



说明:

(1) 利用了原有线路

(2) 未改变原有线路

4、举例

例1：用2K×4b的芯片（若干片）构成一个8KB的存储器，其地址范围在78000H~79FFFH之间。地址总线为A_0~A_{19}，数据总线为D_0~D_7，对芯片读写采用$\overline{MEMR}$(即R操作) $\overline{MEMW}$(即W操作)控制，且片选信号要求采用74LS138译码器输出。

(1)需要2K×4b的芯片多少片构成8KB的存储?

(2)芯片地址如何分配？74LS138译码器如何设置?

(3)画出存储器逻辑电路图。

解:(1)需要的芯片：

需要2K×4b的芯片8片，2片2K×4b的芯片组成一组2KB的芯片，共4组；

(2)芯片地址的分配：2KB：A_0~A_{10}；

3线—8线译码器74LS138(T4138)的真值表如下：

回顾

G_1	$\overline{G}_{2A}+\overline{G}_{2B}$	C	B	A	$\overline{Y}_0$	$\overline{Y}_1$	$\overline{Y}_2$	$\overline{Y}_3$	$\overline{Y}_4$	$\overline{Y}_5$	$\overline{Y}_6$	$\overline{Y}_7$
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0
0	d	d	d	d	1	1	1	1	1	1	1	1
d	1	d	d	d	1	1	1	1	1	1	1	1

二、基本逻辑门及译码器

74LS138译码器设置：由于地址范围在78000H~79FFFH之间，即为8K，也就是4组存储芯片都具有唯一的地址范围，因此，须采用全译码方式:

即剩余的地址线：A_{19}~A_{11}中的全部线选做为74LS138译码器的输入端、使能端，A_{19}~A_{11}是该如何分配?

A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	.	.	.	.	.	.	.	.	A_1	A_0
0	1	1	1	1	0	0	0	0	0	.	.	.	.	.	.	.	.	0	0
0	1	1	1	1	0	0	0	0	1	.	.	.	.	.	.	.	.	1	1
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
0	1	1	1	1	0	0	1	1	0	.	.	.	.	.	.	.	.	0	0
0	1	1	1	1	0	0	1	1	1	.	.	.	.	.	.	.	.	1	1

高6位做译码器使能端输入

74LS138译码器3位输入

二、基本逻辑门及译码器

A_{19}~A_{11}是这样分配的：

输入端:ABC分别接入$A_{13}A_{12}A_{11}$

使能端: G_1: $\overline{MEMR}$、 $\overline{MEMW}$

$\overline{G_{2A}}$: A_{19} A_{14}=00

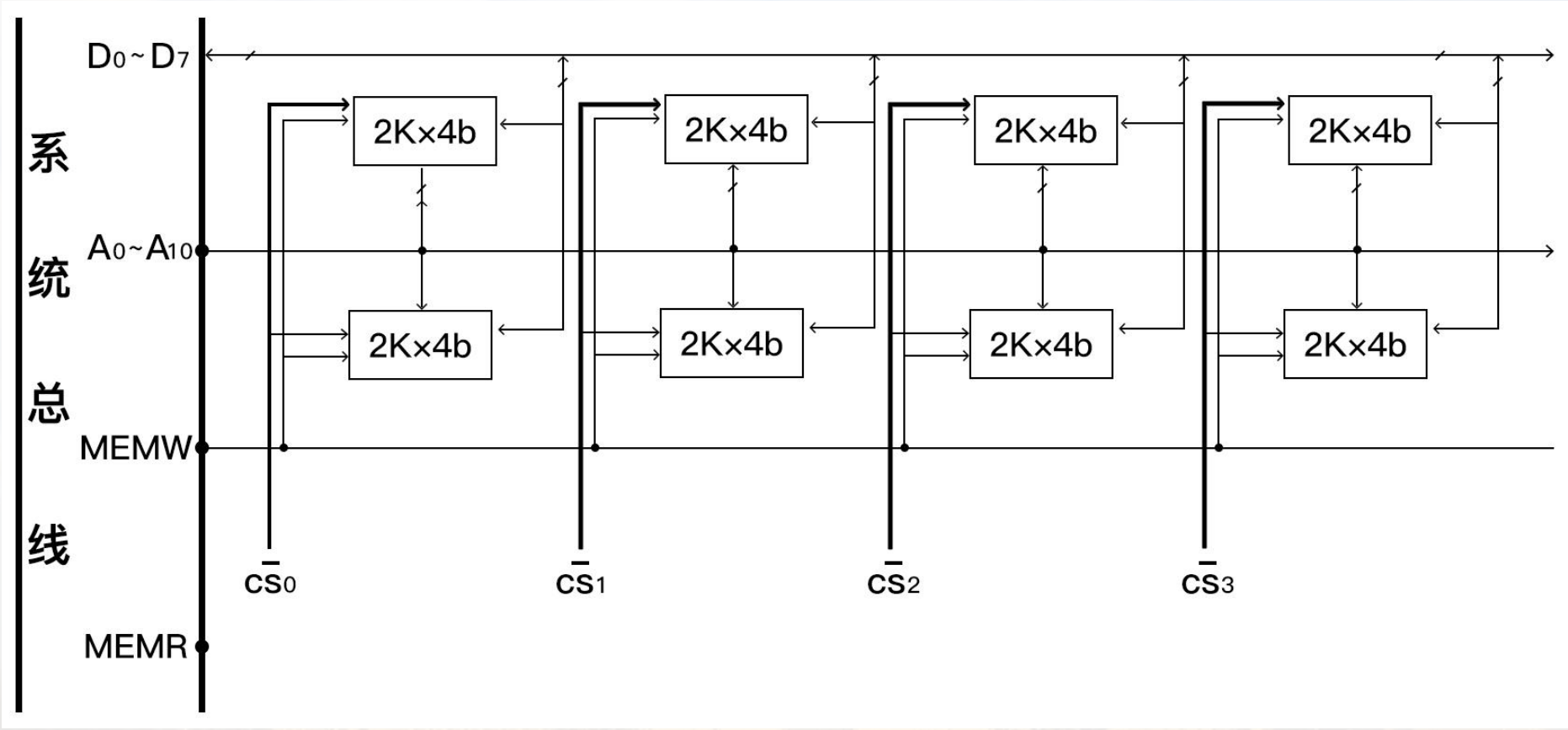
$\overline{G_{2B}}$: $\overline{A_{18}A_{17}A_{16}A_{15}} = \overline{1111}$

A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}
0	1	1	1	1	0
0	1	1	1	1	0
.	.	.	.	.	.
0	1	1	1	1	0
0	1	1	1	1	0

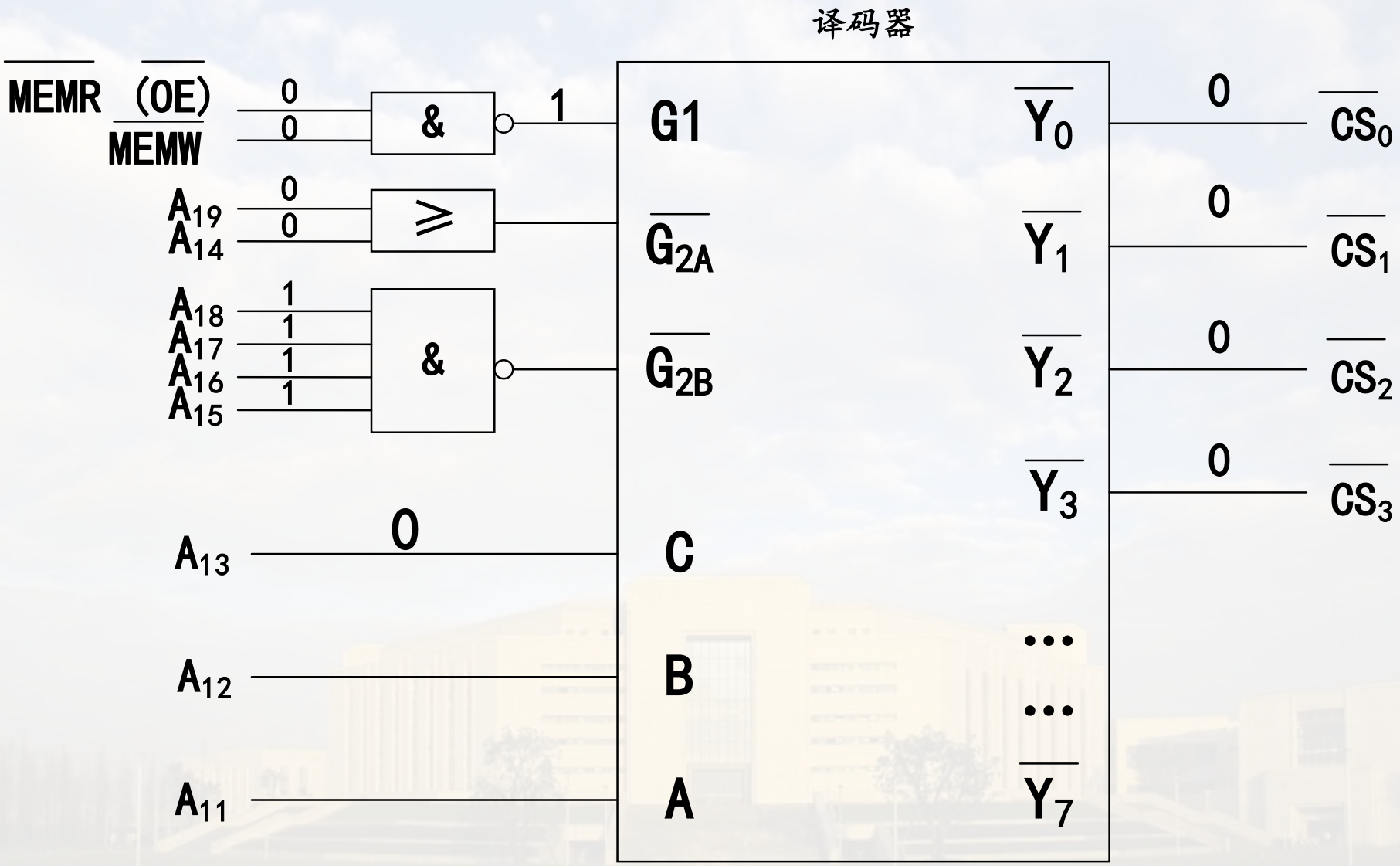
$\overline{G_{2A}}$:$A_{19}A_{14}$=00, $\overline{G_{2B}}$:$A_{18}A_{17}A_{16}A_{15}$=1111， $A_{11}A_{12}A_{13}$=000—011，

片内单元选择A_0--A_{10}:00...0—FF...F

(3)画出存储器逻辑电路图

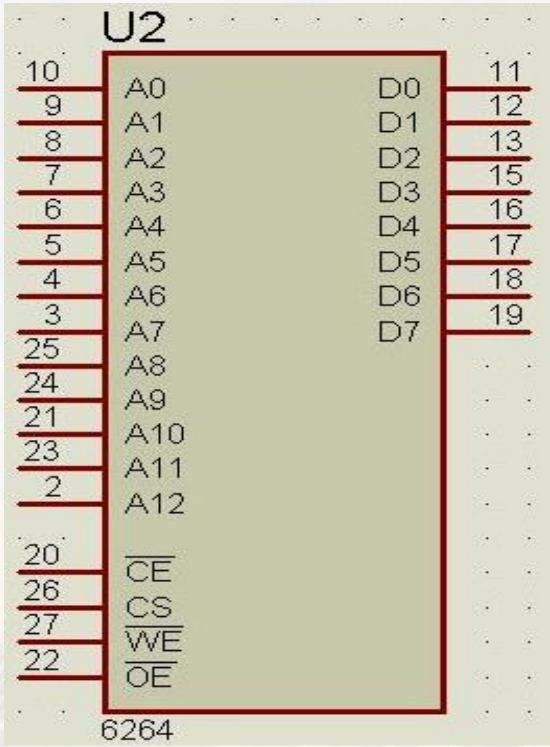


二、基本逻辑门及译码器


译码器
MEMR
(OE)
MEMW
0
0
&
1
A19
A14
0
0
≥
A18
A17
A16
A15
1
1
1
1
&
A13
0
A12
A11
G1
G2A
G2B
C
B
A
Y0
Y1
Y2
Y3
Y7
0
0
0
0
CS0
CS1
CS2
CS3

5、SRAM芯片6264（intel）

6264是8K*8b静态随机存储器芯片,采用CMOS工艺制造,单一+5V供电,额定功耗200mW,典型存取时间200ns,28线双列直插式封装。



各引脚含义如下:

1)A_0-A_{12}为地址线;

2)D_0-D_7为数据线;

3)$\overline{CE}$、CS是片选线;

4) $\overline{OE}$是读允许线; $\overline{WE}$是写允许线.

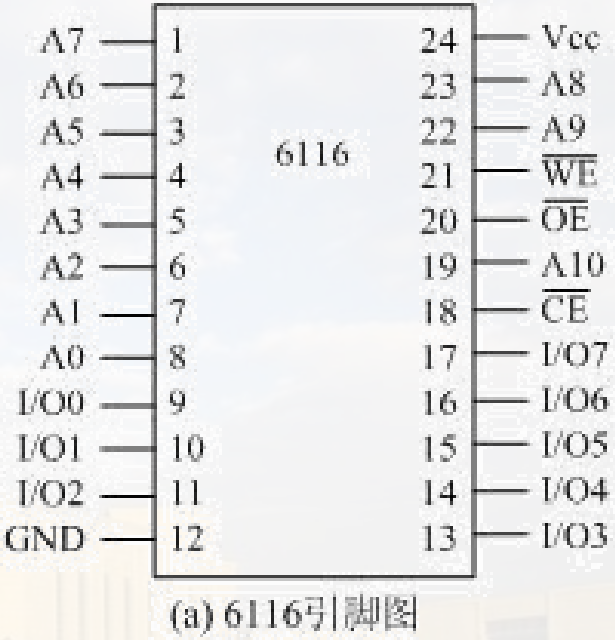
5)其它引线:Vcc为+5V电源，GND是接地端，NC表示空端。

6264功能表

使能端（片选信号）		输入端（读写控制）		输出端
$\overline{CE}$（或$\overline{CS_1}$）	CS（或CS_2）	$\overline{OE}$	$\overline{WE}$	D_0~D_7
0	1	X	0	写入
0	1	0	1	读出
0	0	X	X	三态（高阻）
1	1	X	X	
1	0	X	X	

6、SRAM芯片6116（intel）

6116是2K*8b静态随机存储器芯片,采用CMOS工艺制造,单一+5V供电,额定功耗200mW,典型存取时间200ns,24线双列直插式封装.



(a) 6116引脚图

各引脚含义如下:

1)A_0-A_{10}为地址线;

2)D_0 (I/O0) -D_7 (I/O7)为数据线;

3)$\overline{CS}$是片选线;

4) $\overline{OE}$是读允许线；$\overline{WE}$(R/$\overline{W}$)是写允许线.

5)其它引线:Vcc为+5V电源，GND是接地端.

7、例题

全地址译码方式：利用基本逻辑门电路构成或利用138译码器实现

例1：一片SRAM6264芯片(即8K*8b的SRAM芯片)与8086/8088系统(地址总线为A_0-A_{19}，I/O独立编址)的连接图：

1)要求6264芯片的地址范围为3E000H-3FFFFH（低13位可以是从全为0到全为1之间的任何一个值）；

2)要求6264芯片的地址范围为C0000H-C1FFFH。

二、基本逻辑门及译码器

例1：

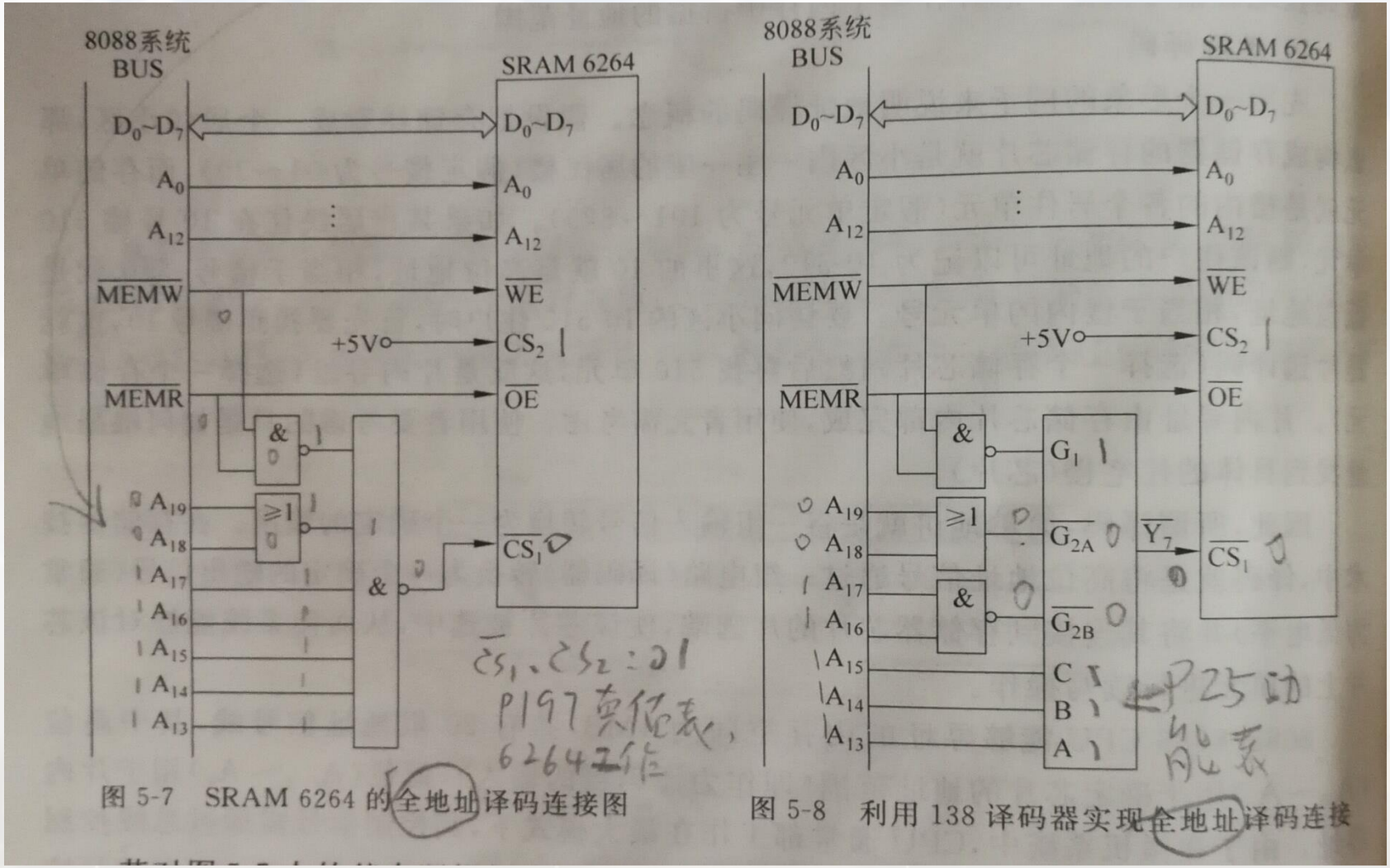


图 5-7　SRAM 6264 的全地址译码连接图

图 5-8　利用 138 译码器实现全地址译码连接

部分地址译码方式

例2：一片SRAM6264芯片与8086 /8088系统(地址总线为A_0—A_{19})的连接图：其地址范围为哪些?

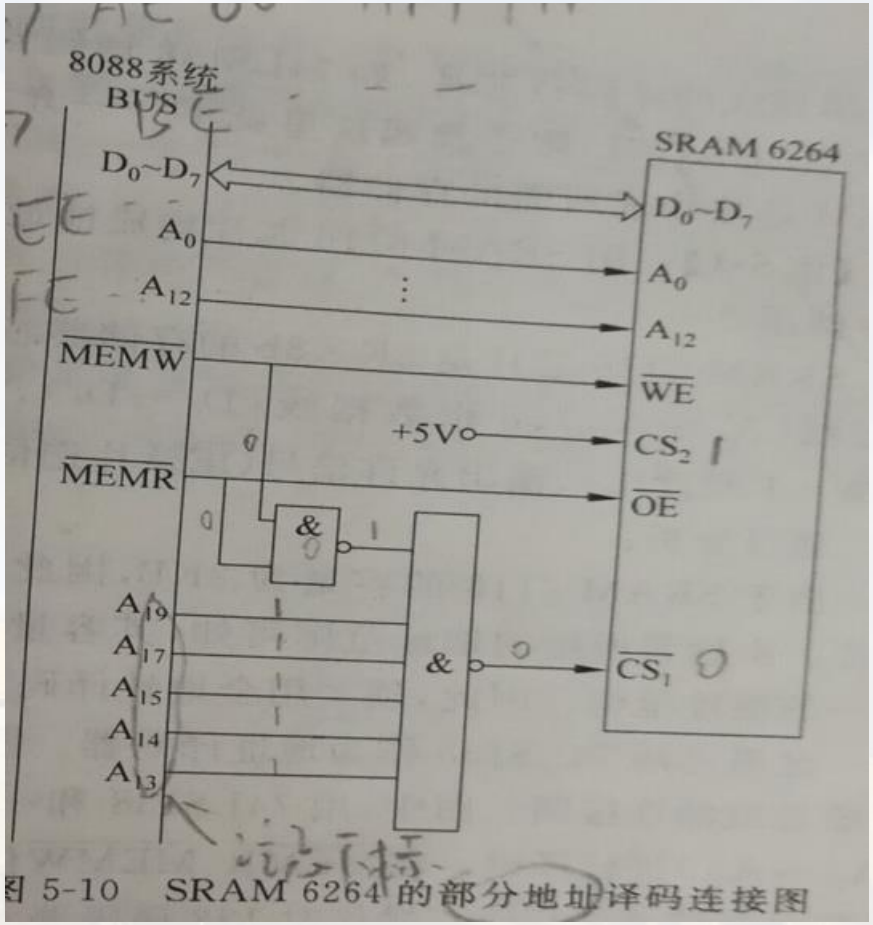


图 5-10 SRAM 6264的部分地址译码连接图

全地址译码/部分地址译码方式

例3：用SRAM6116芯片构成范围在78000H—78FFFH之间的一个4KB的存储器。

SRAM6116芯片是2K×8b的存储芯片，其外部引线如上图所示。具有11根地址线（A_0—A_{10}），8根数据线（D_0-D_7），读写控制信号线R/$\overline{W}$ （当R/$\overline{W}$ =0时写入， R/$\overline{W}$ =1且$\overline{OE}$=0时读出），输出允许信号$\overline{OE}$及片选信号$\overline{CS}$。

二、基本逻辑门及译码器

例3：

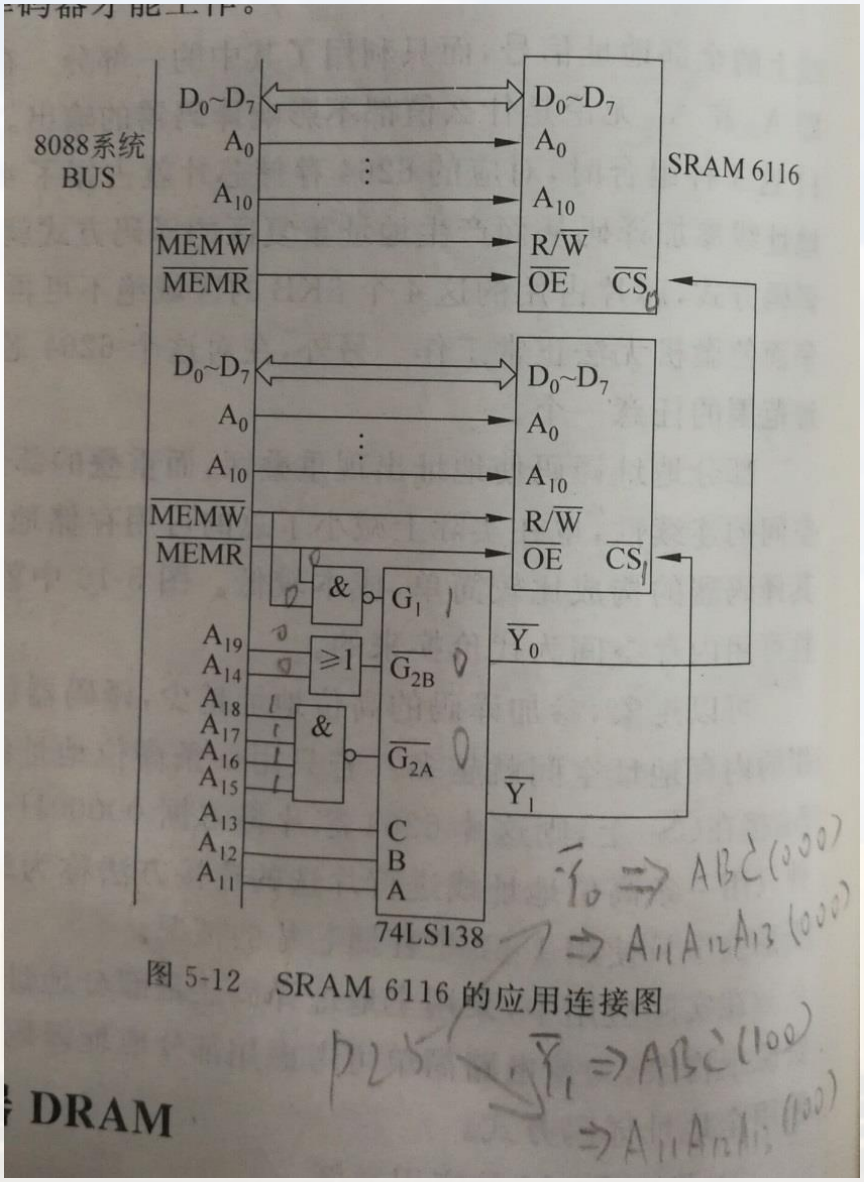


图 5-12 SRAM 6116 的应用连接图

DRAM

1、刷新定义与原因

定义：定期向电容补充电荷——刷新

原因：动态存储器依靠电容电荷存储信息。平时无电源供电，时间一长电容电荷会泄放，需定期向电容补充电荷，以保持信息不变。

刷新与重写的区别：

刷新：动态存储芯片，需补充电荷以保持原来的信息。

重写：破坏性读出后重写，以恢复原来的信息。

三、动态存储器的刷新

2、最大刷新间隔

2-64ms。在此期间，必须对所有动态单元刷新一遍。

3、刷新方法

按行读。

刷新一行所用的时间——刷新周期（存取周期）

刷新一块芯片所需的刷新周期数由芯片矩阵的行数决定。

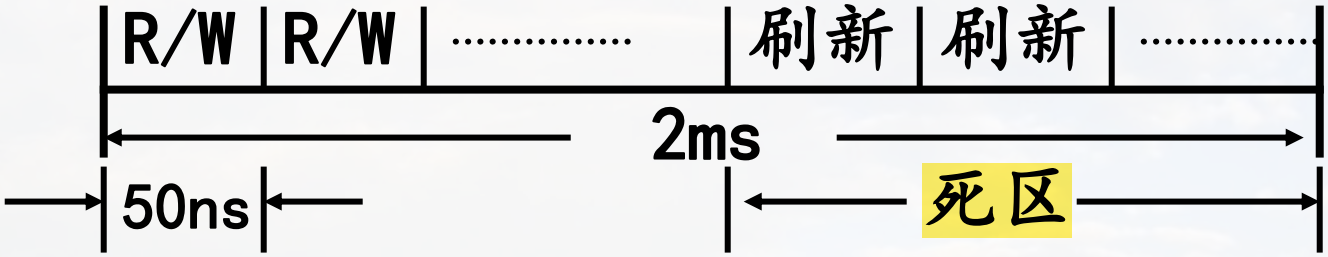
对主存的访问：

- CPU访存：由CPU提供行、列地址，随机访问。
- 动态芯片刷新：由刷新地址计数器提供行地址，定时刷新。

4、刷新周期的安排方式

1）集中刷新

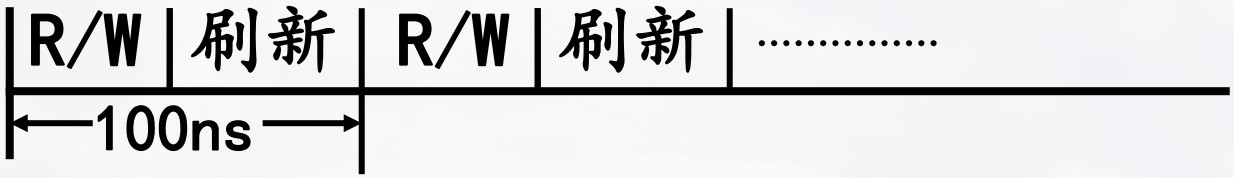
2ms内集中安排所有刷新周期。



用在实时要求不高的场合。

2）分散刷新

各刷新周期分散安排在存取周期中。

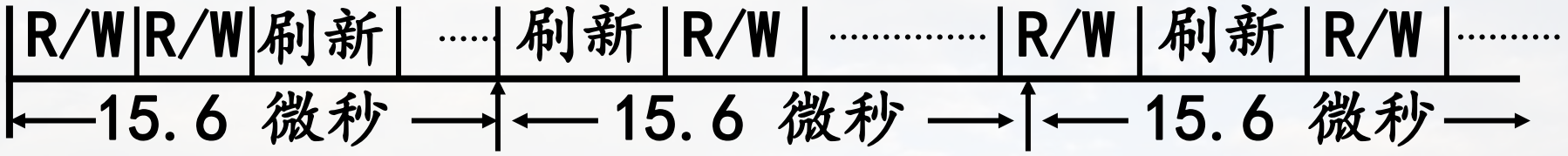


用在低速系统中。

3）异步刷新

各刷新周期分散安排在2ms内。每隔一段时间刷新一行。

例：$\frac{2ms}{128行}$ ≈15.6微秒　每隔15.6微秒提一次刷新请求，刷新一行；2毫秒内刷新完所有行。

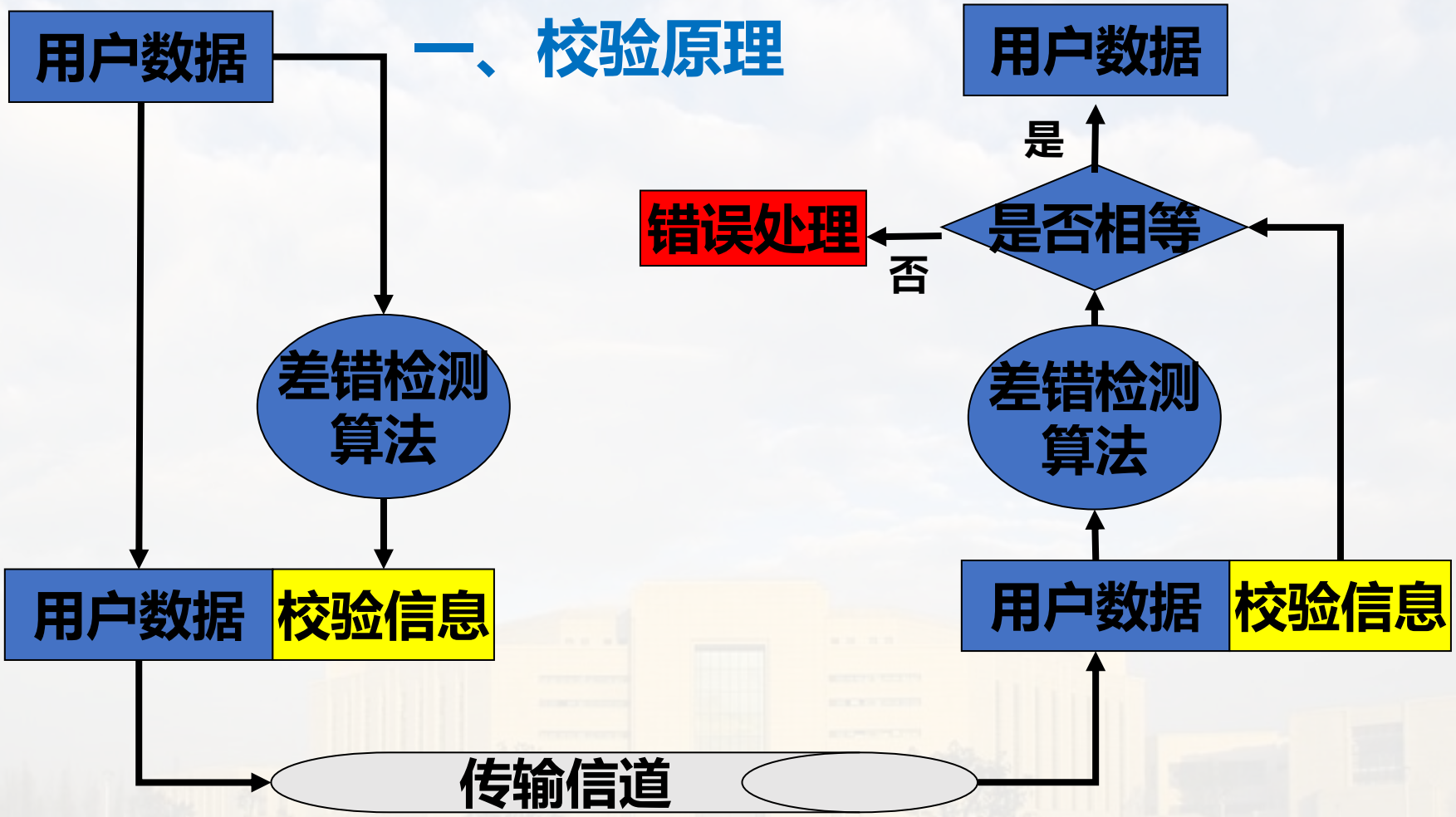


刷新请求（DMA请求）　刷新请求（DMA请求）

用在大多数计算机中。

四、主存储器的校验方法

一、校验原理



- 错误检测不是100%可靠!

2、主存中采用的奇偶校验

1）奇校验：使整个校验码(包括有效信息位和校验位)中“1”的个数为奇数；

2）偶校验：使整个校验码中(包括有效信息位和校验位)“1”的个数为偶数。

四、主存储器的校验方法

（1）奇偶校验示例

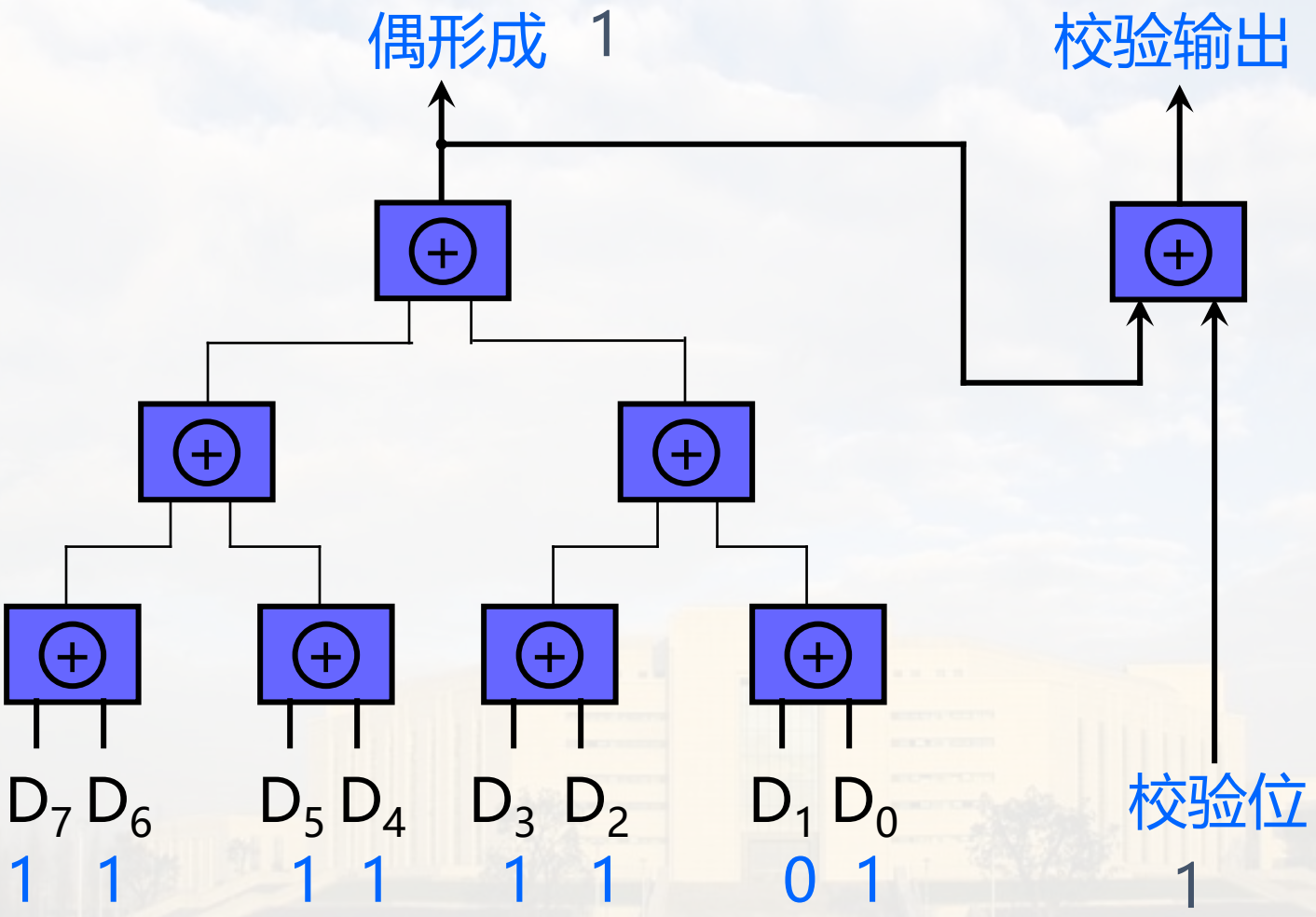
规则：编码中“1”的个数为奇数或偶数

待编码信息　　1011 0001

奇校验编码　　1011 0001　1　→ 校验位

偶校验编码　　1011 0001　0　→ 校验位

(2) 奇偶校验逻辑



奇偶校验逻辑电路

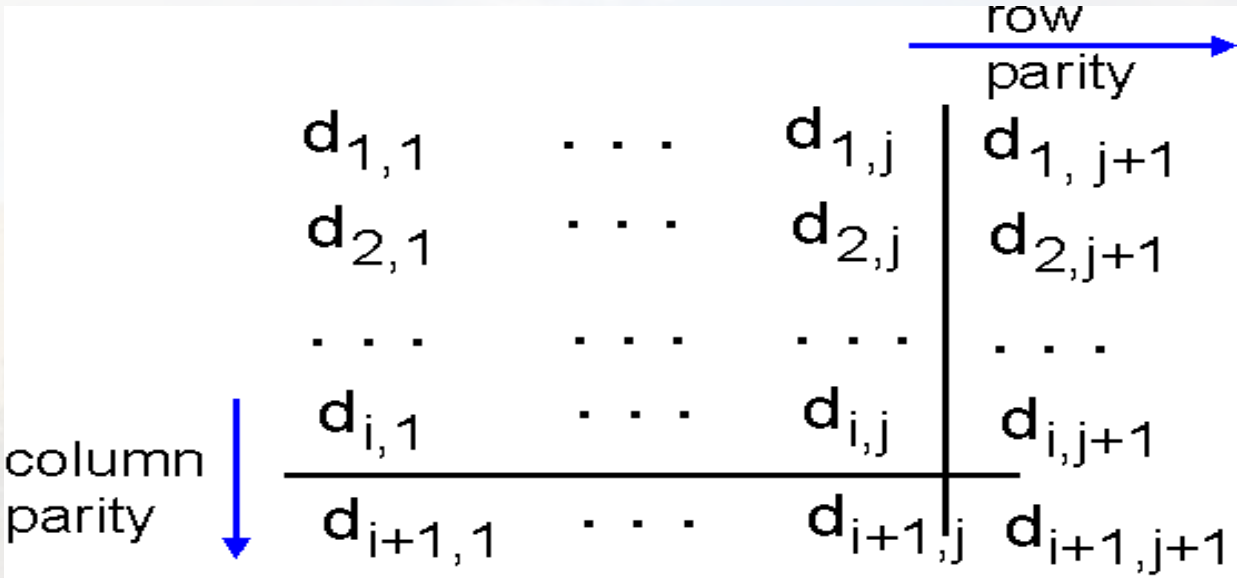
3、ECC校验

ECC（Error Checking and Correcting，错误检查和纠正）是另外一种继奇偶校验之后发展起来的校验技术，具有更高的编码效率和更强的自动纠错能力。

行业中一般将具备ECC功能的主存称为ECC主存，表明它具备ECC方式的检错和纠错。

二维奇偶校验（了解）

- **基本思想：**
  - 将要传信息D（d比特）划分为i行j 列（i 个组，每组j位）；
  - 对每行和每列分别计算奇偶值；
  - 结果的i+j+1个奇偶比特构成了帧的差错检测比特。



四、主存储器的校验方法

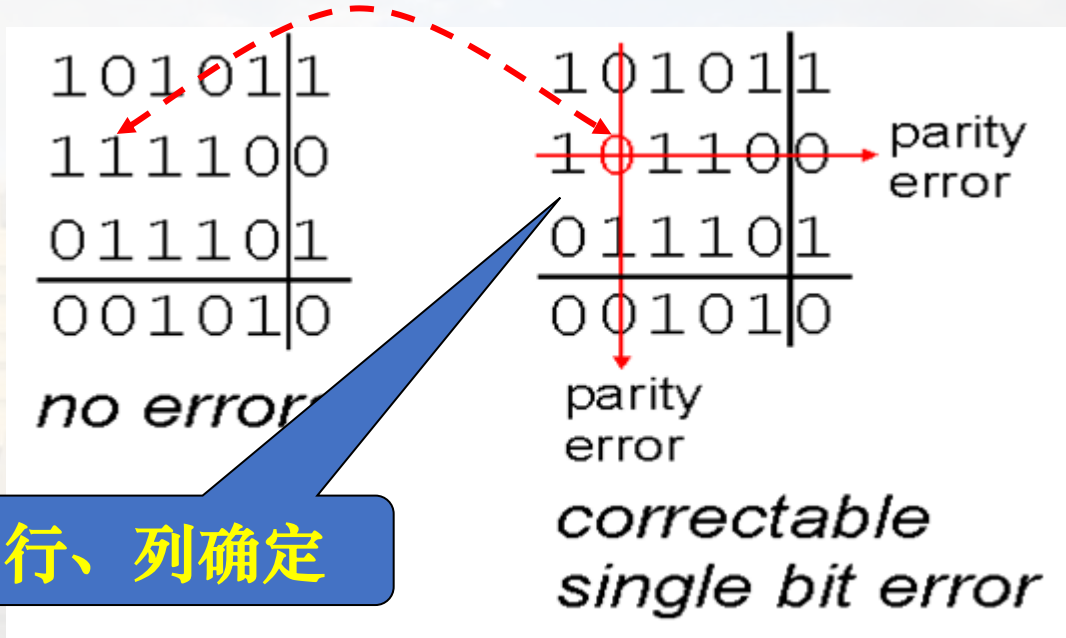
二维奇偶校验（了解）

要发送的数据比特10101 11110 01110，

划分3组，每组5个比特。进行行、列偶校验

特点：

- 可以检测并纠正单个比特差错（数据或校验位中）。
- 能够检测(但不能纠正)分组中任意两个比特的差错。



第四章练习题

重点掌握：根据要求设计存储系统。

要点如下：

- 如果给的是地址范围，则必须掌握根据地方范围计算存储空间容量的方法，比如0800H～13FFH对应是3K容量的存储空间。
- 根据要求的存储容量，选择芯片，需要注意位扩展和字数扩展。
- 分配存储空间：需要掌握确定每个芯片地址范围的方法，需要知道哪些地址线连接芯片引脚，哪些用于产生片选信号。建议先安排大容量芯片，再安排小容量芯片（如果题目有特殊要求，按题目要求处理）。

第四章练习题

重点掌握：根据要求设计存储系统。

- 产生片选信号：写出片选逻辑。
- 如果要求画出存储器的结构图，则必须注意以下事项：
  - 如果使用3-8译码器，需要知道哪些地址引脚用于38译码器的使能端（G_1、$\overline{G_{2A}}$、$\overline{G_{2B}}$）、哪些地址线用于38译码器的3根数据线线作为输入。
  - 片选信号的连接：存储芯片的使能端连接38译码器的哪个输出。如果使用独立逻辑门，则需要会画出相关的逻辑电路。
  - 注意：存储芯片的连接必须包含数据线（如果有位扩展，数据线需要画出不同的部分）、地址线（不同芯片连接的地址线的位数需要标注出来）、读写控制线。（需要注意是一根控制线R/$\overline{W}$，还是两根独立的R和W控制线），如果题目标注了是IO独立编制，还需要考虑 M/IO控制线或访存控制线。

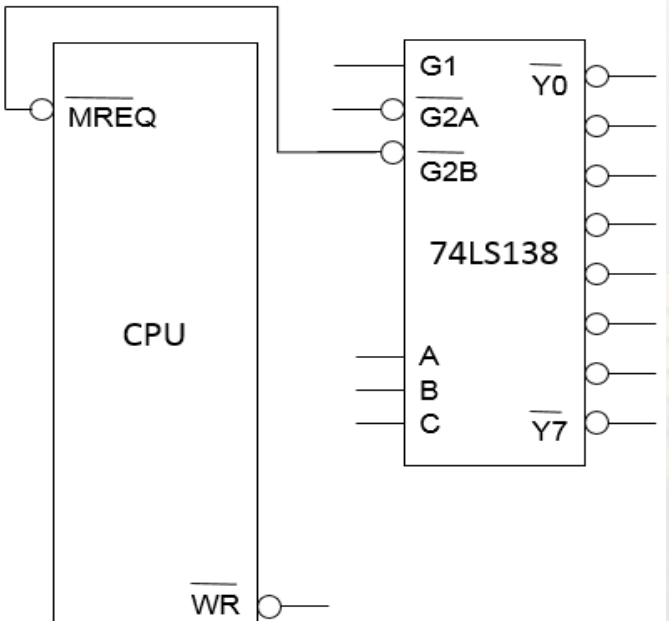
第四章练习题

例题：

设CPU有16根地址线，8根数据线，用$\overline{MREQ}$作为访存控制信号（低电平有效），用$\overline{WR}$作为读写控制信号（高电平读，低电平写）。现有下列芯片：1K×4的RAM，2K×4的RAM,4K×4的RAM,2K×8的ROM，4K×8的ROM，以及74LS138译码器。其中存储芯片引脚$\overline{WR}$(高电平读，低电平写),片选引脚$\overline{CS}$ 。仅选用上述芯片（不增加其他门电路和芯片），要求地址空间分配6000H-67FFH为系统程序区（ROM芯片），6800H-77FFH为用户程序区（RAM芯片）。

① 需要选用哪几种存储芯片？各需要多少片？芯片的地址范围是多少？

① 画出CPU、74LS138和存储芯片之间的连接图。

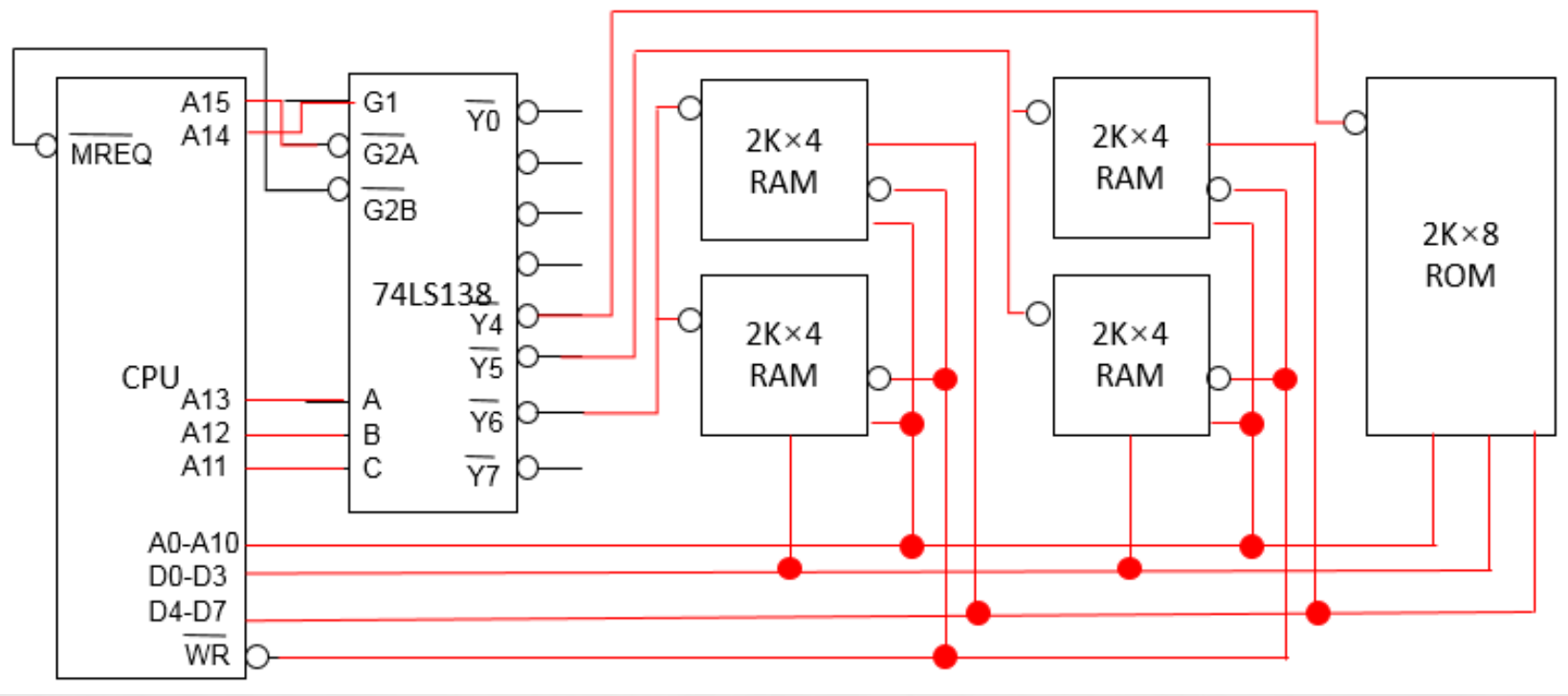


第四章练习题

参考答案：（注：该答案的推导过程省略，大家答题时务必要有推导过程）

（1）需要一片2K×8的ROM；四片2K×4的RAM。2K×8的ROM的地址范围：6000H-67FFH；第1组（两片2K×4）RAM的地址范围：6800H-6FFFH；第2组（两片2K×4）RAM的地址范围：7000H-77FFH

（2）



第四章练习题

名词术语（掌握，不用做）

存取周期、动态刷新、数据传输率

作业　P317

第3大题、第5大题、第6大题、第8大题、第9大题

题号	分数
3大题	21
5大题	43
6大题	21
8大题	18
9大题	5
总分	108

谢谢观看

计算机组成原理

2023/11/15

信息与软件工程学院
School of Information and Software Engineering