内存技术指南

第 1 页：序：不得不说的话

作为电脑中必不可少的三大件之一（其余的两个是主板与 CPU），内存是决定系统性能的关键设备之一，它就像一个临时的仓库，负责数据的中转、暂存……

不过，虽然内存对系统性能的至关重要，但长期以来，DIYer 并不重视内存，只是将它看作是一种买主板和 CPU 时顺带买的“附件”，那时最多也就注意一下内存的速度。这种现象截止于 1998 年 440BX 主板上市后，PC66/100 的内存标准开始进入普通 DIYer 的视野，因为这与选购有着直接的联系。一时间，有关内存时序参数的介绍文章大量出现（其中最为著名的恐怕就是 CL 参数）。自那以后，DIYer 才发现，原来内存也有这么多的学问。接下来，始于 2000 年底/2001 年初的 VIA 芯片组 4 路交错（4-Way Interleave）内存控制和部分芯片组有关内存容量限制的研究，则是深入了解内存的一个新开端。本刊在 2001 年第 2 期上也进行了 VIA 内存交错控制与内存与模组结构的详细介绍，并最终率先正确地解释了这一类型交错（内存交错有多种类型）的原理与容量限制的原因。从那时起，很多关于内存方面的深入性文章接踵而至，如果说那时因此而掀起了一股内存热并不夸张。大量的内存文章让更多的用户了解了内存，以及更深一层的知识，这对于 DIY 当然是一件好事情。然而，令人遗憾的是这些所谓的内存高深技术文章有不少都是错的(包括后来的 DDR 与 RDRAM 内存的介绍)，有的甚至是很低级的错误。在这近两年的时间里，国内媒体上优秀的内存技术文章可谓是寥若晨星，有些媒体还编译国外 DIY 网站的大篇内存文章，但可惜的是，外国网站也不见得都是对的（这一点，似乎国内很多作者与媒体似乎都忽视了）。就这样，虽然打开了一个新的知识领域，可“普及”的效果并不那么好，很多媒体的铁杆读者高兴地被带入内存深层世界，但也因此被引向了新的误区。

不过，从这期间（2001 年初至今）各媒体读者对这类文章的反映来看，喜欢内存技术的玩家大有人在且越来越多，这是各媒体“培养”的成果。这些用户已经不满足如何正确的使用内存，他们更渴望深入的了解这方面原来非常贫乏的知识，这些知识可能暂时不会对他们在使用内存过程中有什么帮助，但会大大满足他们的求知欲。在 2001 年初，我们揭开 VIA 芯片组 4 路交错内存控制和部分芯片组有关内存容量限制之迷时，还是主要围绕着内存使用的相关话题来展开，而且在这期间有关内存技术的话题，《电脑高手》也都是一笔带过。但在今天，在很多人希望了解内存技术而众多媒体的文章又“力不从心”时，我们觉得有必要再次站出来以正视听，也就是说，我们这次的专题不再以内存使用为中心，更多的是纯技术性介绍，并对目前现存的主要内存技术误区进行重点纠正。

在最后要强调的是，本专题以技术为主，由于篇幅的原因，不可能从太浅的方面入手，所以仍需要有一定的技术基础作保证，而对内存感兴趣的读者则绝不容错过，这也许是您最好的纠正错误认识的机会！

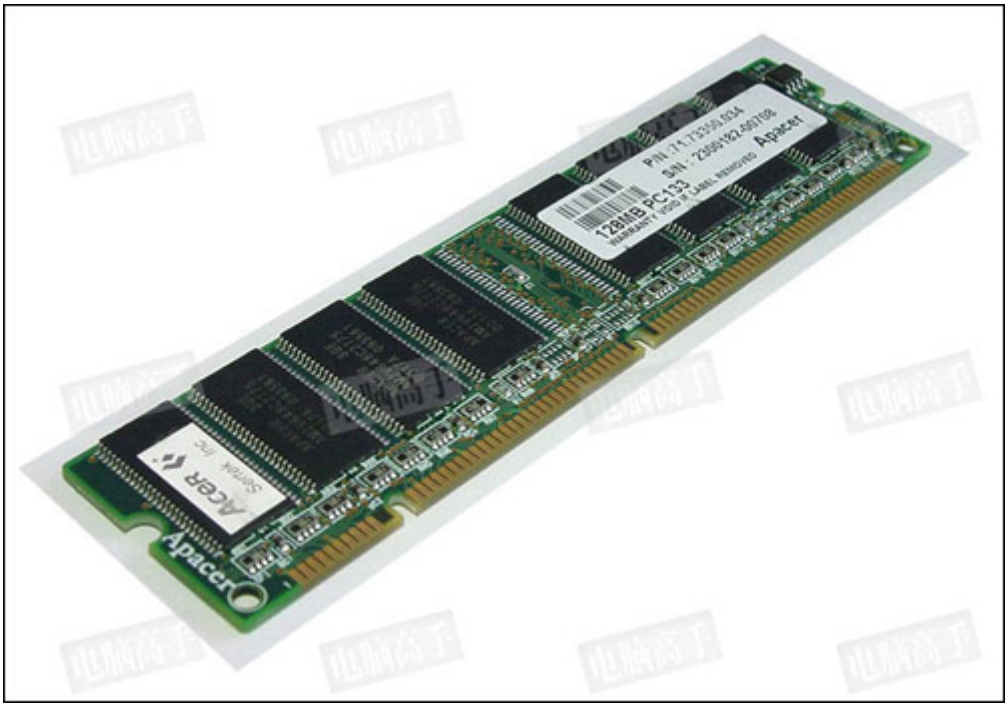
在本专题里，当讲完内存的基本操作之后，我们会给大家讲一个仓库的故事，从中相信您会更了解内存这个仓库是怎么工作的，希望您能喜欢。

第 2 页：SDRAM 与内存基础概念（一）

虽然有关内存结构与时序的基础概念，在本刊 2001 年第 2 期的专题中就已有阐述，但在这里为了保证专题的可读性，我们需要再次加强这方面的系统认识。正确并深刻理解内存的基础概念，是阅读本专题的第一条件。因为即使是 RDRAM，在很多方面也是与 SDRAM 相似的，而至于 DDR 与 DDR-Ⅱ、QBM 等形式的内存更是与 SDRAM 有着紧密的联系。

一、 SDRAM内存模组与基本结构

我们平时看到的 SDRAM 都是以模组形式出现，为什么要做成这种形式呢？这首先要接触到两个概念：物理 Bank 与芯片位宽。



PC133 时代的 168pin SDRAM DIMM

1、 物理 Bank

传统内存系统为了保证 CPU 的正常工作，必须一次传输完 CPU 在一个传输周期内所需要的数据。而 CPU 在一个传输周期能接受的数据容量就是 CPU 数据总线的位宽，单位是 bit（位）。当时控制内存与 CPU 之间数据交换的北桥芯片也因此将内存总线的数据位宽等同于 CPU 数据总线的位宽，而这个位宽就称之为物理 Bank（Physical Bank，下文简称 P-Bank）的位宽。所以，那时的内存必须要组织成 P-Bank 来与 CPU 打交道。资格稍老的玩家应该还记得 Pentium 刚上市时，需要两条 72pin 的 SIMM 才能启动，因为一条 72pin -SIMM 只能提供 32bit 的位宽，不能满足 Pentium 的 64bit 数据总线的需要。直到 168pin-SDRAM DIMM 上市后，才可以使用一条内存开机。下面将通过芯片位宽的讲述来进一步解释 P-Bank 的概念。

不过要强调一点，P-Bank 是 SDRAM 及以前传统内存家族的特有概念，在 RDRAM 中将以通道（Channel）取代，而对于像 Intel E7500 那样的并发式多通道 DDR 系统，传统的 P-Bank 概念也不适用。

提示：SDRAM、SIMM、DIMM、pin 的含义
SDRAM：Synchronous Dynamic Random Access Memory，同步动态随机存储器。同步是指其时钟频率与 CPU 前端总线的系统时钟频率相同，并且内部的命令的发送与数据的传输都以它为基准；动态是指存储阵列需要不断的刷新来保证数据不丢失；随机是指数据不是线性依次存储，而是自由指定地址进行数据的读写。 pin：模组或芯片与外部电路联接用的金属引脚，而模组的 pin 就是常说的“金手指”。 SIMM：Single In-line Memory Module，单列内存模组。内存模组就是我们常说的内存条，所谓单列是指模组电路板与主板插槽的接口只有一列引脚（虽然两侧都有金手指）。 DIMM：Double In-line Memory Module，双列内存模组。所谓双列是指模组电路板与主板插槽的接口有两列引脚，模组电路板两侧的金手指对应一列引脚。

2、 芯片位宽

上文已经讲到 SDRAM 内存系统必须要组成一个 P-Bank 的位宽，才能使 CPU 正常工作，那么这个 P-Bank 位宽怎么得到呢？这就涉及到了内存芯片的结构。

每个内存芯片也有自己的位宽，即每个传输周期能提供的数据量。理论上，完全可以做出一个位宽为 64bit 的芯片来满足 P-Bank 的需要，但这对技术的要求很高，在成本和实用性方面也都处于劣势。所以芯片的位宽一般都较小。台式机市场所用的 SDRAM 芯片位宽最高也就是 16bit，常见的则是 8bit。这样，为了组成 P-Bank 所需的位宽，就需要多颗芯片并联工作。对于 16bit 芯片，需要 4 颗（4×16bit=64bit）。对于 8bit 芯片，则就需要 8 颗了。

提示：内存芯片与颗粒
很多时候，经常听人们说到“内存颗粒”，其实这是港台地区对内存芯片的一种称呼（仅对内存，其他的芯片，港台则称为“晶片”），两者的意思是一样的。具体怎么说，就看个人喜好了，就笔者而言，而倾向于用“内存芯片”来表述。

以上就是芯片位宽、芯片数量与 P-Bank 的关系。P-Bank 其实就是一组内存芯片的集合，这个集合的容量不限，但这个集合的总位宽必须与 CPU 数据位宽相符。随着计算机应用的发展，一个系统只有一个 P-Bank 已经不能满足容量的需要。所以，芯片组开始可以支持多个 P-Bank，一次选择一个 P-Bank 工作，这就有了芯片组支持多少（物理）Bank 的说法。而在 Intel 的定义中，则称 P-Bank 为行（Row），比如 845G 芯片组支持 4 个行，也就是说它支持 4 个 P-Bank。另外，在一些文档中，也把 P-Bank 称为 Rank（列）。

回到开头的话题，DIMM 是 SDRAM 集合形式的最终体现，每个 DIMM 至少包含一个 P-Bank 的芯片集合。在目前的 DIMM 标准中，每个模组最多可以包含两个 P-Bank 的内存芯片集合，虽然理论上完全可以在一个 DIMM 上支持多个 P-Bank，比如 SDRAM DIMM 就有 4 个芯片选择信

号（Chip Select，简称片选或CS），理论上可以控制4个P-Bank的芯片集合。只是由于某种原因而没有这么去做。比如设计难度、制造成本、芯片组的配合等。至于DIMM的面数与P-Bank数量的关系，在2001年2月的专题中已经明确了，面数≠P-Bank数，只有在知道芯片位宽的情况下，才能确定P-Bank的数量，大度256MB内存就是明显一例，而这种情况在Registered模组中非常普遍。有关内存模组的设计，将在后面的相关章节中继续探讨。

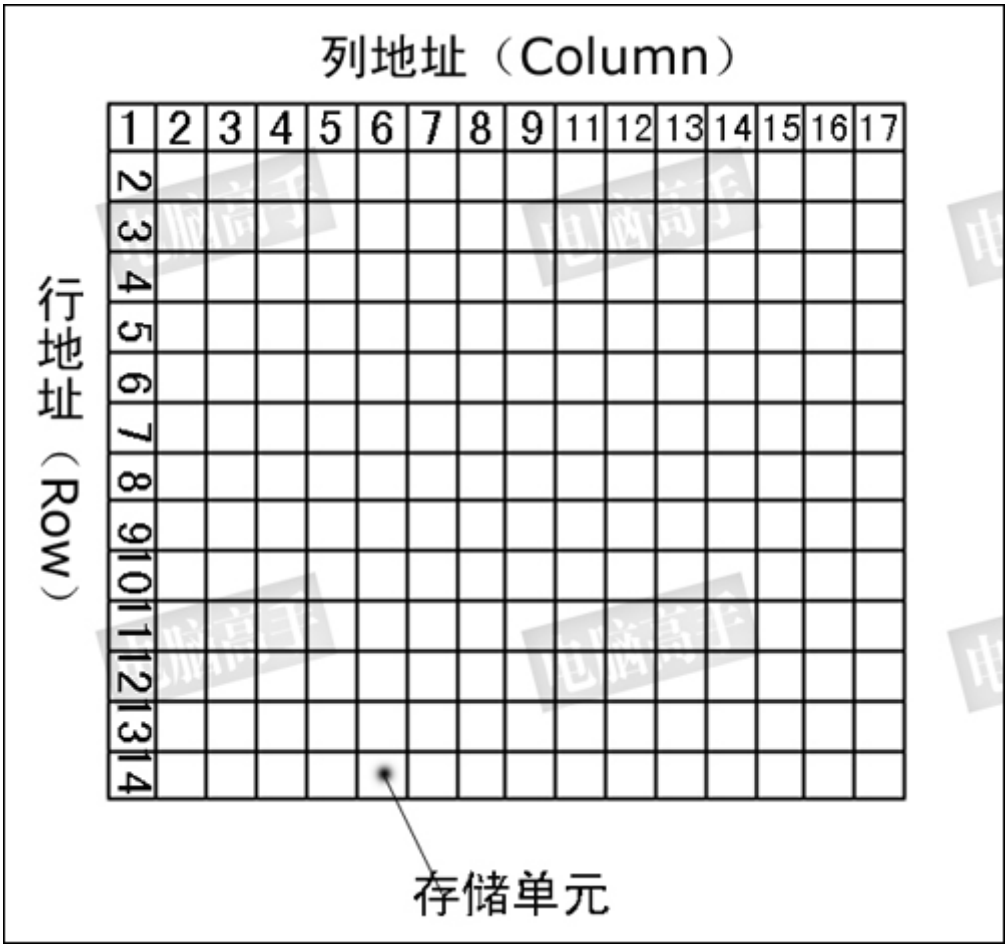
第3页：SDRAM与内存基础概念（二）

二、 SDRAM内存芯片的内部结构

1、逻辑Bank与芯片位宽

讲完SDRAM的外在形式，就该深入了解SDRAM的内部结构了。这里主要的概念就是逻辑Bank。简单地说，SDRAM的内部是一个存储阵列。因为如果是管道式存储（就如排队买票），就很难做到随机访问了。

阵列就如同表格一样，将数据“填”进去，你可以它想象成一张表格。和表格的检索原理一样，先指定一个行（Row），再指定一个列（Column），我们就可以准确地找到所需要的单元格，这就是内存芯片寻址的基本原理。对于内存，这个单元格可称为存储单元,那么这个表格（存储阵列）叫什么呢？它就是逻辑Bank（Logical Bank，下文简称L-Bank）。



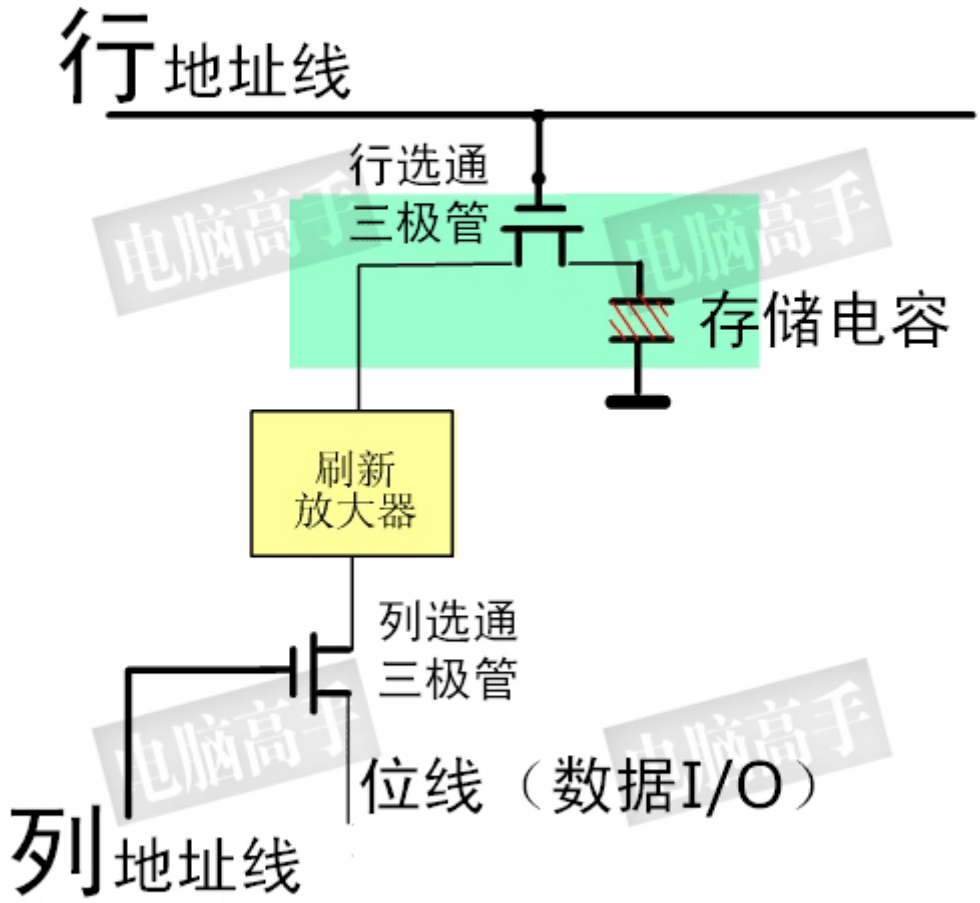
L-Bank 存储阵列示意图

由于技术、成本等原因，不可能只做一个全容量的 L-Bank，而且最重要的是，由于 SDRAM 的工作原理限制，单一的 L-Bank 将会造成非常严重的寻址冲突，大幅降低内存效率（在后文中将详细讲述）。所以人们在 SDRAM 内部分割成多个 L-Bank，较早以前是两个，目前基本都是 4 个，这也是 SDRAM 规范中的最高 L-Bank 数量。到了 RDRAM 则最多达到了 32 个，在最新 DDR-Ⅱ的标准中，L-Bank 的数量也提高到了 8 个。

这样，在进行寻址时就要先确定是哪个 L-Bank，然后再在这个选定的 L-Bank 中选择相应的行与列进行寻址。可见对内存的访问，一次只能是一个 L-Bank 工作，而每次与北桥交换的数据就是 L-Bank 存储阵列中一个“存储单元”的容量。在某些厂商的表述中，将 L-Bank 中的存储单元称为 Word（此处代表位的集合而不是字节的集合）。

从前文可知，SDRAM 内存芯片一次传输率的数据量就是芯片位宽，那么这个存储单元的容量就是芯片的位宽（也是 L-Bank 的位宽），但要注意，这种关系也仅对 SDRAM 有效，原因将在下文中说明。

提示：DRAM 的存储原理
L-Bank 中的存储单元是基本的存储单位，它的容量是若干 Bit（对于 SDRAM 而言，就是芯片的位宽），而每个 Bit 则是存放于一个单独的存储体中。这些存储体就是内存中最小的存储单元。你可以用硬盘操作中的簇与扇区的关系来理解内存中的存储形式。扇区是硬盘中的最小存储单元（相当于内存中的存储体），而每个簇则包含有多个扇区（相当于 L-Bank 中的存储单元），数据的交换都是一个簇为单位进行（一次传输一个存储单元的数据）。



DRAM 的存储原理示意图：行选与列选信号将使存储电容与外界间的传输电路导通，从而可进行放电（读取）与充电（写入）。另外，图中刷新放大器的设计并不固定，目前这一功能被并入读出放大器（Sense Amplifier，简称 S-AMP），具体操作在下文中详细讲述。

2、内存芯片的容量

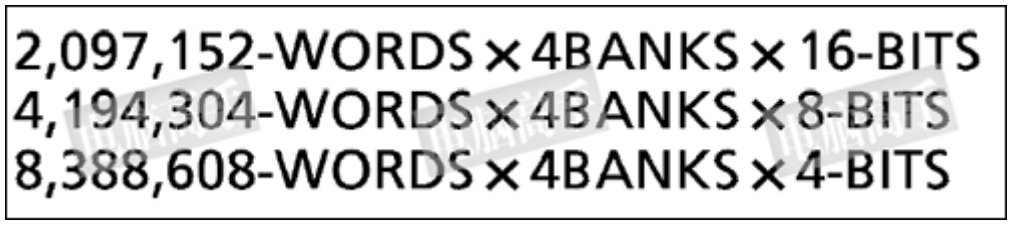
现在我们应该清楚内存芯片的基本组织结构了。那么内存的容量怎么计算呢？显然，内存芯片的容量就是所有 L-Bank 中的存储单元的容量总合。计算有多少个存储单元和计算表格中的单元数量的方法一样：

存储单元数量=行数×列数（得到一个L-Bank的存储单元数量）×L-Bank的数量

在很多内存产品介绍文档中，都会用 M×W 的方式来表示芯片的容量（或者说是芯片的规格/组织结构）。M 是该芯片中存储单元的总数，单位是兆（英文简写 M，精确值是 1048576，而不是 1000000），W 代表每个存储单元的容量，也就是 SDRAM 芯片的位宽（Width），单位是 bit。计算出来的芯片容量也是以 bit 为单位，但用户可以采用除以 8 的方法换算为字节（Byte）。比如 8M×8，这是一个 8bit 位宽芯片，有 8M 个存储单元，总容量是 64Mbit（8MB）。

提示：bit、Byte、Word 的关系
bit：位。二进制数中，一个 0 或 1 就是一个 bit。 Byte：字节。8 个 bit 为一个字节，这与 ASCII（American Standard Code for Information Interchange，美国标准信息交换代码）的规定有关，ASCII 用 8 位二进制数来表示 256 个信息代码，所以 8 个 bit 定义为一个字节。 Word：字：两个字节为一个字，这里的 Word 不是指 L-Bank 中存储单元，此外还有双字（DWords，Double Words，4 个字节）和四字（QWord，Quad Words，8 个字节）的表示法。目前一个 P-Bank 的位宽就是 QWord，这在很多 CPU 与芯片组的介绍中经常用到。

不过，M×W 是最简单的表示方法。下图则是某公司对自己内存芯片的容量表示方法，这可以说是最正规的形式之一。



业界正规的内存芯片容量表示方法

我们可以计算一下，结果可以发现这三个规格的容量都是 128Mbits，只是由于位宽的变化引起了存储单元的数量变化。从这个例子就也可以看出，在相同的总容量下，位宽可以采用多种不同的设计。

3、与芯片位宽相关的 DIMM 设计

为什么在相同的总容量下，位宽会有多种不同的设计呢？这主要是为了满足不同领域的需要。现在大家已经知道 P-Bank 的位宽是固定的，也就是说当芯片位宽确定下来后，一个 P-Bank 中芯片的个数也就自然确定了，而前文讲过 P-Bank 对芯片集合的位宽有要求，对芯片集合的容量则没有任何限制。高位宽的芯片可以让 DIMM 的设计简单一些（因为所用的芯片少），但在芯片容量相同时，这种 DIMM 的容量就肯定比不上采用低位宽芯片的模组，因为后者在一个 P-Bank 中可以容纳更多的芯片。比如上文中那个内存芯片容量标识图，容量都是 128Mbit，合 16MB。如果 DIMM 采用双 P-Bank+16bit 芯片设计，那么只能容纳 8 颗芯片，计 128MB。但如果采用 4bit 位宽芯片，则可容纳 32 颗芯片，计 512MB。DIMM 容量前后相差出 4 倍，可见芯片位宽对 DIMM 设计的重要性。因此，8bit 位宽芯片是桌面台式机上容量与成本之间平衡性较好的选择，所以在市场上也最为普及，而高于 16bit 位宽的芯片一般用在

需要更大位宽的场合，如显卡等，至于 4bit 位宽芯片很明显非常适用于大容量内存应用领域，基本不会在标准的 Unbuffered 模组设计中出现。

第 4 页：SDRAM 与内存基础概念（三）

三、 SDRAM的引脚与封装

内存芯片要想工作，必须要与内存控制器有所联系，同时对于一个电气元件，电源供应也是必不可少的，而且数据的传输要有一个时钟作为触发参考。因此，SDRAM 在封装时就要留出相应的引脚以供使用。电源与时钟的引脚就不必多说了，现在我们可以想象一下，至少应该有哪些控制引脚呢？

我们从内存寻址的步骤缕下来就基本明白了，从中我们也就能了解内存工作的大体情况。这里需要说明的是，与 DIMM 一样，SDRAM 有着自己的业界设计规范，在一个容量标准下，SDRAM 的引脚/信号标准不能只考虑一种位宽的设计，而是要顾及多种位宽，然后尽量给出一个通用的标准，小位宽的芯片也许会空出一些引脚，但高位宽的芯片可能就全部用上了。不过容量不同时，设计标准也会有所不同，一般的容量越小的芯片所需要的引脚也就越小。

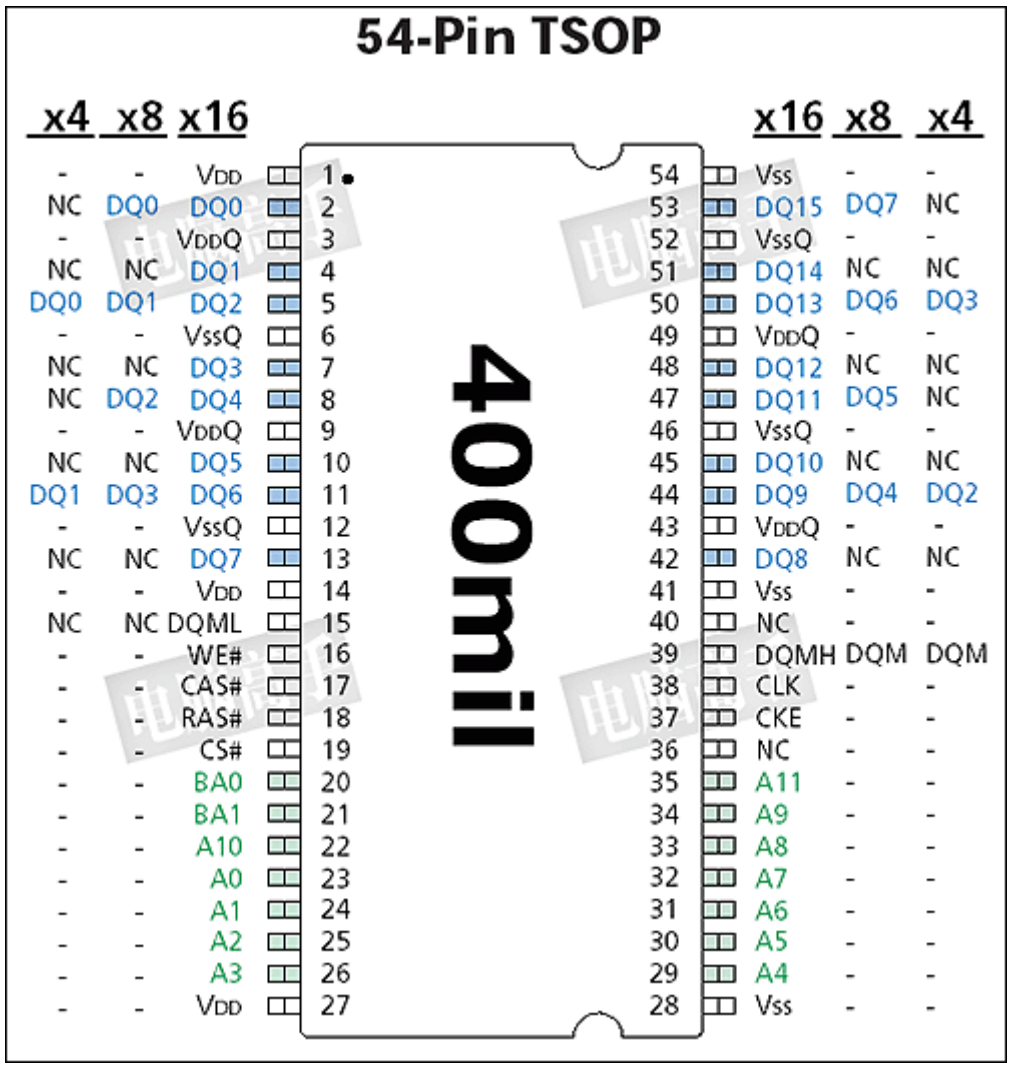
1、 首先，我们知道内存控制器要先确定一个 P-Bank 的芯片集合，然后才对这集合中的芯片进行寻址操作。因此要有一个片选的信号，它一次选择一个 P-Bank 的芯片集（根据位宽的不同，数量也不同）。被选中的芯片将同时接收或读取数据，所以要有一个片选信号。

2、 接下来是对所有被选中的芯片进行统一的L-Bank的寻址，目前SDRAM中L-Bank的数量最高为 4 个，所以需要两个L-Bank地址信号（2^2=4）。

3、 最后就是对被选中的芯片进行统一的行/列（存储单元）寻址。地址线数量要根据芯片的组织结构分别设计了。但在相同容量下，行数不变，只有列数会根据位宽的而变化，位宽越大，列数越少，因为所需的存储单元减少了。

4、 找到了存储单元后，被选中的芯片就要进行统一的数据传输，那么肯定要有与位宽相同数量的数据 I/O 通道才行，所以肯定要有相应数量的数据线引脚。

现在我们就基本知道了内存芯片的一些信号引脚，下图就是一个简单的 SDRAM 示意图，大家可以详细看看。



图注：128Mbit 芯片不同位宽的引脚图（NC 代表未使用，-表示与内侧位宽设计相同）

引脚代号	定义	引脚代号	定义
Vdd/ VddQ	工作/DQ 电压	CAS#	列地址选通脉冲
Vss/VssQ	相应电压的接地	RAS#	行地址选通脉冲
DQn	数据 I/O 线	CK	时钟信号
An	行/列地址线	CKE	时钟有效
DQM	数据掩码	BAn	L-Bank 地址线
CS#	片选	WE#	写允许

注：#表示低电平有效

根据 SDRAM 的官方规范，台式机上所用的 SDRAM 在不同容量下的各种位宽封装标准如下：

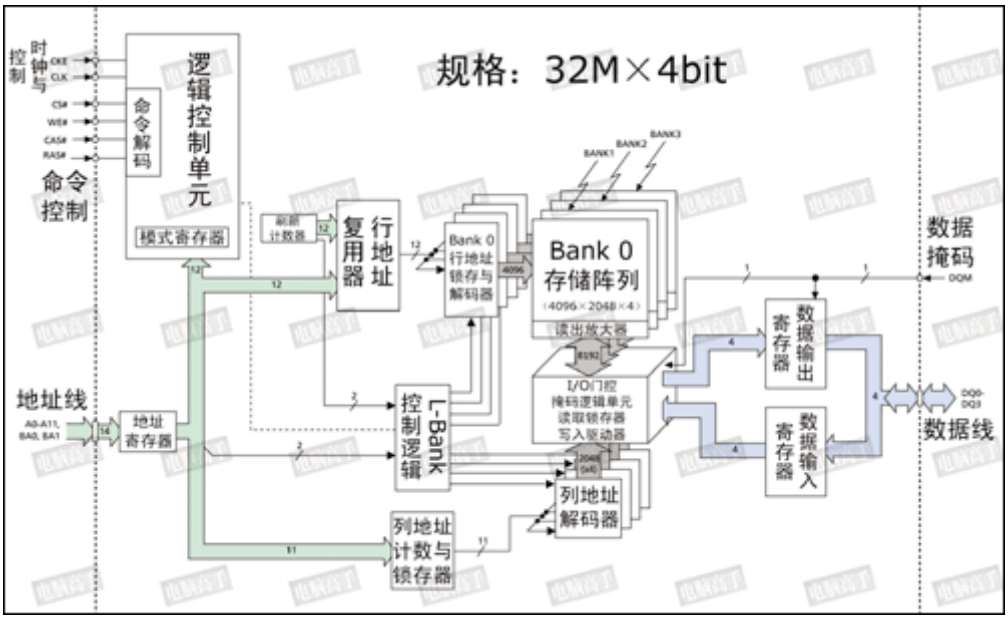
容量 \ 位宽	4bit	8bit	16bit
16Mbit	44pin TSOP-II		50pin TSOP-II
64/128Mbit	54pin TSOP-II（400mil）注		
256/512Mbit			

注：400mil（400 千分之一英寸）是指芯片封装的宽度，约合 10.16 毫米

第 5 页：SDRAM 与内存基础概念（四）

四、SDRAM的内部基本操作与工作时序

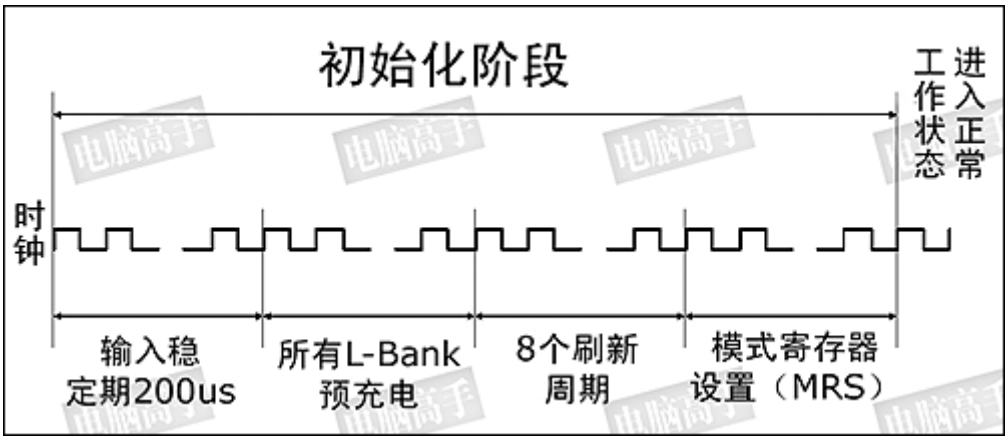
上文我们已经了解了 SDRAM 所用到的基本信号线路，下面就看看它们在 SDRAM 芯片内部是怎么“布置”的，并从这里开始深入了解内存的基本操作与过程，在这一节中我们将接触到有天书之称的时序图，但不要害怕，根据文中的指导慢慢理解，您肯定可以看懂它。首先，我们先认识一下 SDRAM 的内部结构，然后再开始具体的讲述。



128Mbit（32M×4）SDRAM 内部结构图（点击放大）

1、芯片初始化

可能很多人都想象不到，在 SDRAM 芯片内部还有一个逻辑控制单元，并且有一个模式寄存器为其提供控制参数。因此，每次开机时 SDRAM 都要先对这个控制逻辑核心进行初始化。有关预充电和刷新的含义在下文有讲述，关键的阶段就在于模式寄存器（MR，Mode Register）的设置，简称 MRS（MR Set），这一工作由北桥芯片在 BIOS 的控制下进行，寄存器的信息由地址线来提供。

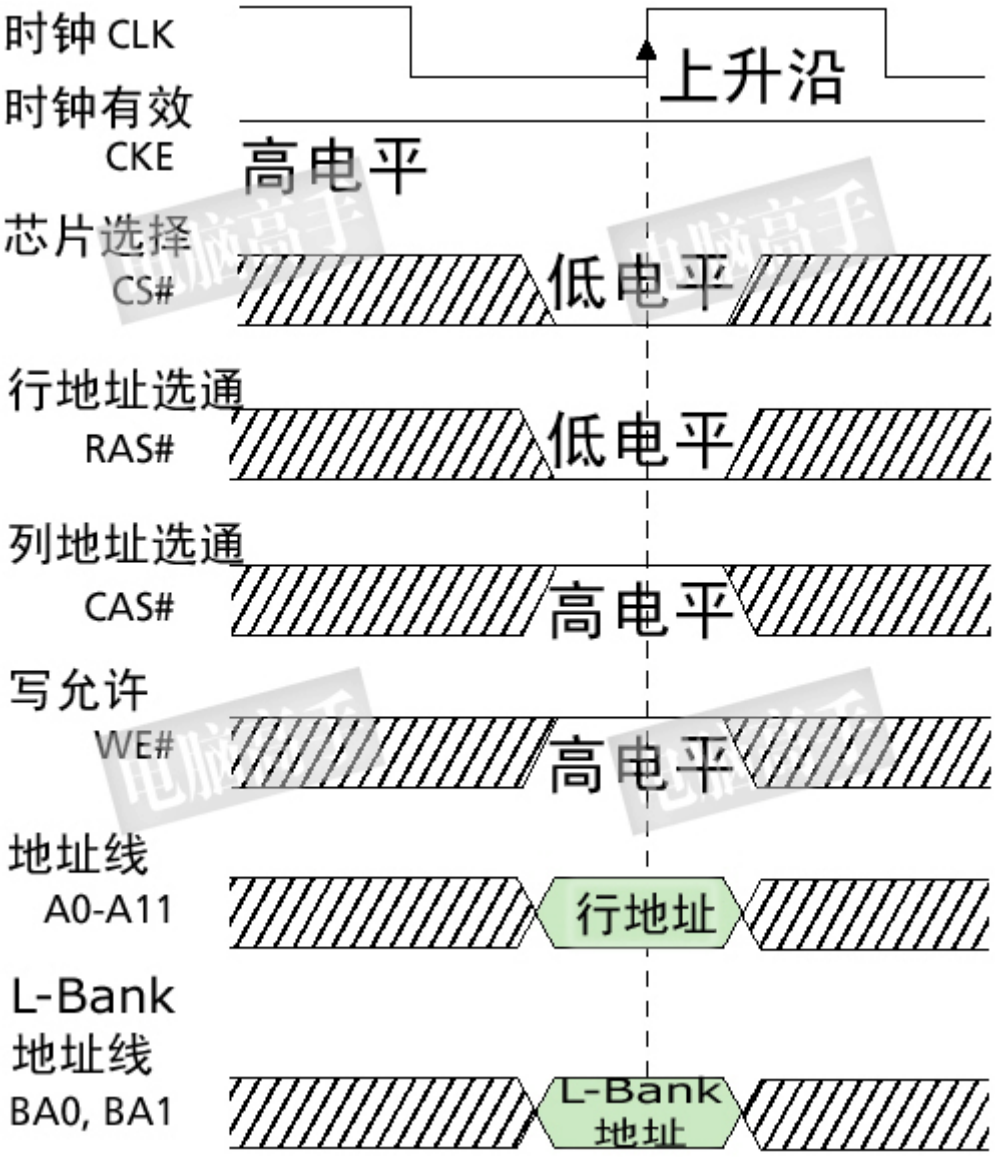


SDRAM 在开机时的初始化过程

SDRAM 模式寄存器所控制的操作参数：地址线提供不同的 0/1 信号来获得不同的参数。在设置到 MR 之后，就开始了进入正常的工作状态，图中相关参数将结合下文具体讲述

2、行有效

初始化完成后，要想对一个 L-Bank 中的阵列进行寻址，首先就要确定行（Row），使之处于活动状态（Active），然后再确定列。虽然之前要进行片选和 L-Bank 的定址，但它们与行有效可以同时进行。



行有效时序图

从图中可以看出，在CS#、L-Bank定址的同时，RAS（Row Address Strobe，行地址选通脉冲）也处于有效状态。此时An地址线则发送具体的行地址。如图中是A0-A11，共有 12 个地址线，由于是二进制表示法，所以共有 4096 个行（2^{12}=4096），A0-A11 的不同数值就确定了具体的行地址。由于行有效的同时也是相应L-Bank有效，所以行有效也可称为L-Bank有效。

3、列读写

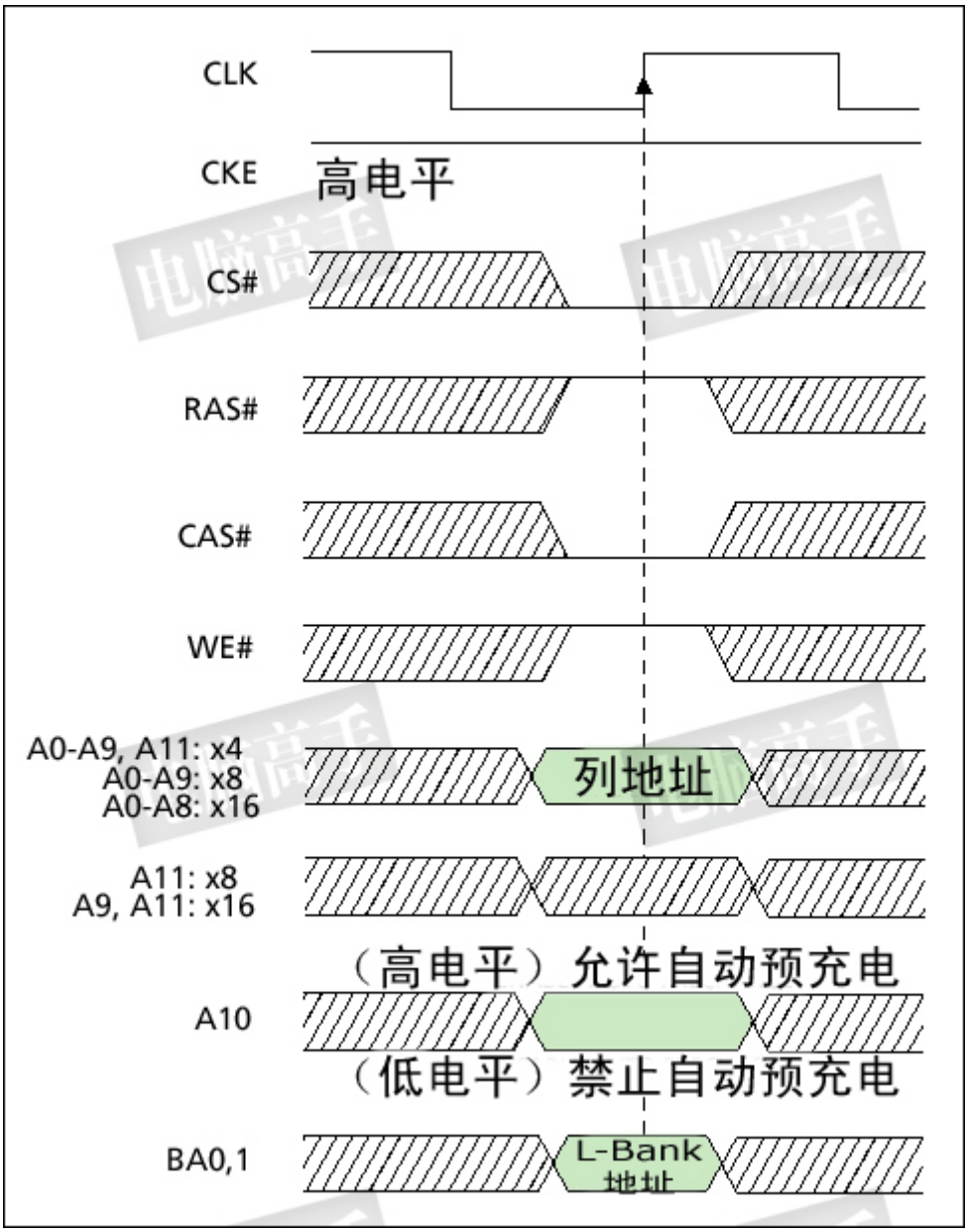
行地址确定之后，就要对列地址进行寻址了。但是，地址线仍然是行地址所用的 A0-A11（本例）。没错，在 SDRAM 中，行地址与列地址线是共用的。不过，读/写的命令是怎么发出的

呢？其实没有一个信号是发送读或写的明确命令的，而是通过芯片的可写状态的控制来达到读/写的目的。显然 WE#信号就是一个关键。WE#无效时，当然就是读取命令。

命令名（功能）	CS#	RAS#	CAS#	WE#	DQM	地址线	DQs
命令禁止（NOP）	H	X	X	X	X	X	X
无操作（NOP）	L	H	H	H	X	X	X
有效/活动（使指定L-Bank中的指定行有效）	L	L	H	H	X	Bank/行	X
读取（从指定L-Bank中的指定列开始读取数据）	L	H	L	H	L/H	Bank/列	X
写入（从指定L-Bank中的指定列开始写入数据）	L	H	L	L	L/H	Bank/列	有效
突发传输终止	L	H	H	L	X	X	活动
预充电（关闭指定或全部L-Bank中的工作行）	L	L	H	L	X	相应编码	X
自动刷新或自刷新（后者进入自刷新模式）	L	L	L	H	X	X	X
模式寄存器加载（写入）	L	L	L	L	X	操作码	X
写允许/输出允许	-	-	-	-	L	-	有效
写禁止/输出屏蔽（High-Z）	-	-	-	-	H	-	屏蔽

SDRAM 基本操作命令（上表可点击放大），通过各种控制/地址信号的组合来完成（H 代表高电平，L 代表低电平，X 表示高低电平均没有影响）。此表中，除了自刷新命令外，所有命令都是默认 CKE 有效。对于自刷新命令，下文有详解

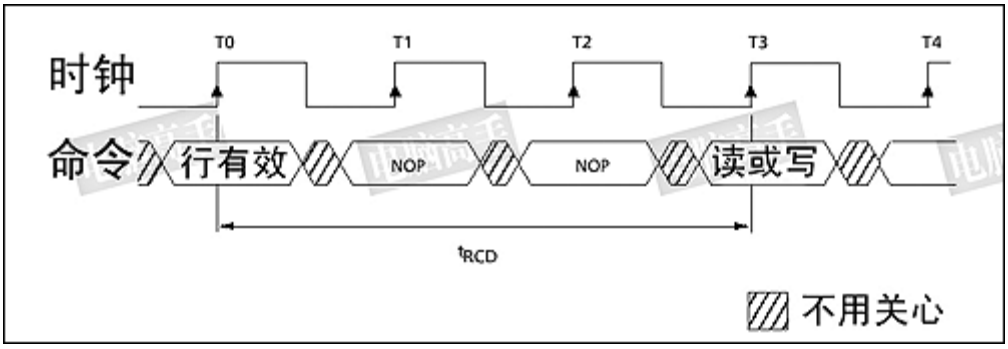
列寻址信号与读写命令是同时发出的。虽然地址线与行寻址共用，但 CAS（Column Address Strobe，列地址选通脉冲）信号则可以区分开行与列寻址的不同，配合 A0-A9，A11（本例）来确定具体的列地址。



读写操作示意图，读取命令与列地址一块发出（当 WE#为低电平是即为写命令）

然而，在发送列读写命令时必须要与行有效命令有一个间隔，这个间隔被定义为 tRCD，即 RAS to CAS Delay（RAS 至 CAS 延迟），大家也可以理解为行选通周期，这应该是根据芯片存储阵列电子元件响应时间（从一种状态到另一种状态变化的过程）所制定的延迟。tRCD 是 SDRAM 的一个重要时序参数，可以通过主板 BIOS 经过北桥芯片进行调整，但不能超过厂商的预定范围。广义的 tRCD 以时钟周期（tCK，Clock Time）数为单位，比如 tRCD=2，就代表延迟周期为两个时钟周期，具体到确切的时间，则要根据时钟频率而定，对于 PC100 SDRAM，tRCD=2，代表 20ns 的延迟，对于 PC133 则为 15ns。

提示：时钟频率、时钟周期及数据传输频率之间的关系
时钟频率以赫兹（Hz）为单位，如 PC100 SDRAM 的时钟频率为 100MHz（这里的 M=1000000），时钟周期以秒（s）为单位，两者间的换算公式为：时钟周期=1/时钟频率。如果 PC100 SDRAM，时钟频率为 1/100000000s，合 10ns（纳秒）。 至于数据传输频率，是指每秒钟传输数据的次数，它不一定等于时钟频率，但肯定以它为基准。比如 DDR 内存，数据传输频率就是时钟频率的两倍。



tRCD=3 的时序图

第 6 页：SDRAM 与内存基础概念（五）

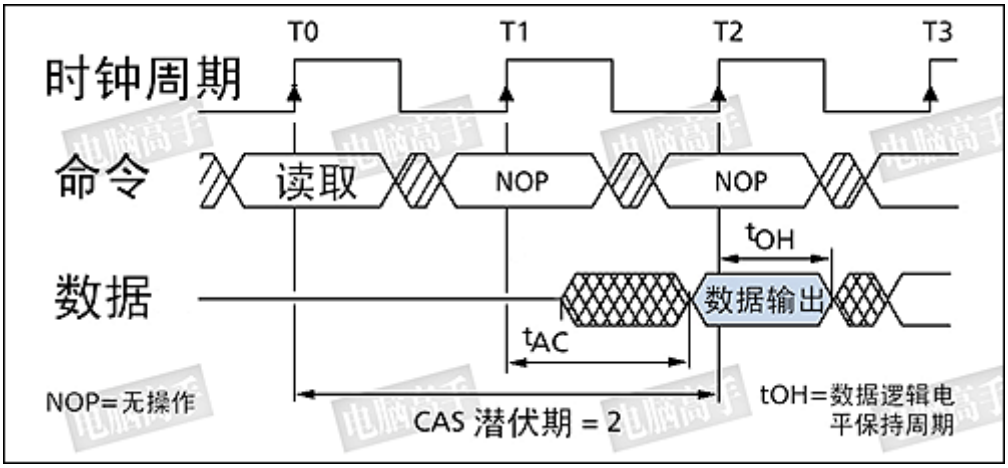
4、 数据输出（读）

在选定列地址后，就已经确定了具体的存储单元，剩下的事情就是数据通过数据 I/O 通道（DQ）输出到内存总线上了。但是在 CAS 发出之后，仍要经过一定的时间才能有数据输出，从 CAS 与读取命令发出到第一笔数据输出的这段时间，被定义为 CL（CAS Latency，CAS 潜伏期）。由于 CL 只在读取时出现，所以 CL 又被称为读取潜伏期（RL，Read Latency）。CL 的单位与 tRCD 一样，为时钟周期数，具体耗时由时钟频率决定。

不过，CAS 并不是在经过 CL 周期之后才送达存储单元。实际上 CAS 与 RAS 一样是瞬间到达的，但 CAS 的响应时间要更快一些。为什么呢？假设芯片位宽为 n 个 bit，列数为 c，那么一个行地址要选通 n×c 个存储体，而一个列地址只需选通 n 个存储体。但存储体中晶体管的反应时间仍会造成数据不可能与 CAS 在同一上升沿触发，肯定要延后至少一个时钟周期。

由于芯片体积的原因，存储单元中的电容容量很小，所以信号要经过放大来保证其有效的识别性，这个放大/驱动工作由 S-AMP 负责，一个存储体对应一个 S-AMP 通道。但它要有一个准备时间才能保证信号的发送强度（事前还要进行电压比较以进行逻辑电平的判断），因此从数据 I/O 总线上有数据输出之前的一个时钟上升沿开始，数据即已传向 S-AMP，也就是说此时数据已经被触发，经过一定的驱动时间最终传向数据 I/O 总线进行输出，这段时间我们称之为 tAC（Access Time from CLK，时钟触发后的访问时间）。tAC 的单位是 ns，对于不同的频率各有不同的明确规定，但必须要小于一个时钟周期，否则会因访问时过长而使效率

降低。比如 PC133 的时钟周期为 7.5ns，tAC 则是 5.4ns。需要强调的是，每个数据在读取时都有 tAC，包括在连续读取中，只是在进行第一个数据传输的同时就开始了第二个数据的 tAC。



CL=2 与 tAC 示意图

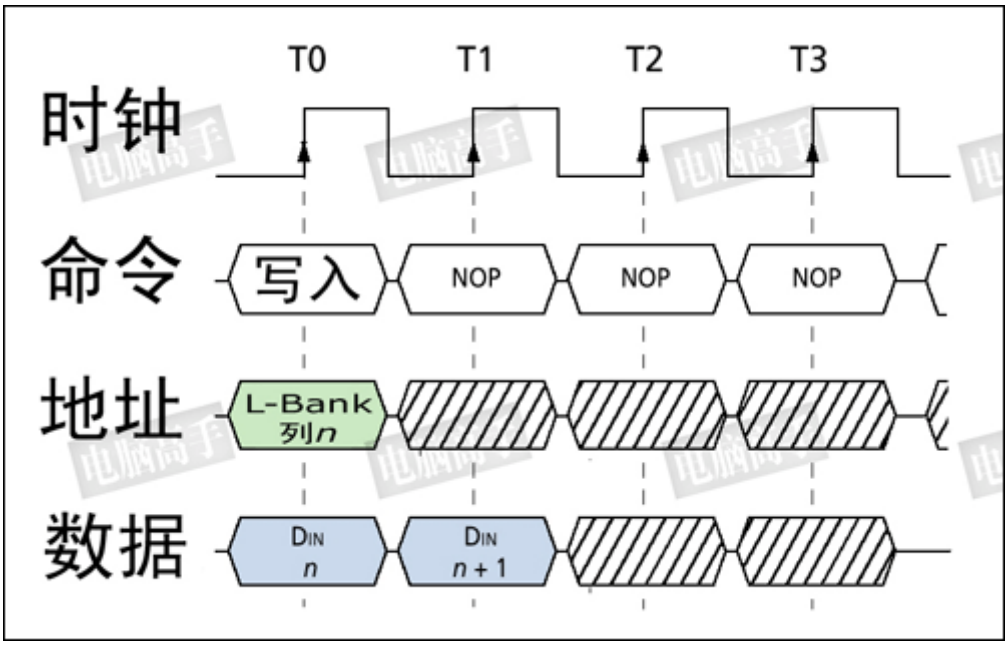
提示：潜伏期≠延迟
从表面上看 CL 造成了读取数据的滞后，但就 CAS 而言并没有延迟，因为延迟与潜伏期有着根本的不同定义（否则规范制定者也不会特意用 Latency 与 Delay 来区分）。延迟是指一个信息或一个事件被推迟的时间量，潜伏期则是指已经发生，但没有达到一定的水平，就如病毒的潜伏期一样，可能病菌早以入驻你的身体，只是产生的反应还不足以引起你的注意。我们可以说 CL 造成了输出延迟，但不能因此认为 CL=CD（CAS Delay），后者更接近于 tRCD 的含义。

CL 的数值不能超出芯片的设计规范，否则会导致内存的不稳定，甚至开不了机（超频的玩家应该有体会），而且它也不能在数据读取前临时更改。CL 周期在开机初始化过程中的 MRS 阶段进行设置，在 BIOS 中一般都允许用户对其调整，然后 BIOS 控制北桥芯片在开机时通过 A4-A6 地址线对 MR 中 CL 寄存器的信息进行更改。

不过，从存储体的结构图上可以看出，原本逻辑状态为 1 的电容在读取操作后，会因放电而变为逻辑 0。所以，以前的 DRAM 为了在关闭当前行时保证数据的可靠性，要对存储体中原有的信息进行重写，这个任务由数据所经过的刷新放大器来完成，它根据逻辑电平状态，将数据进行重写（逻辑 0 时就不重写），由于这个操作与数据的输出是同步进行互不冲突，所以不会产生新的重写延迟。后来通过技术的改良，刷新放大器被取消，其功能由 S-AMP 取代，因为在读取时它会保持数据的逻辑状态，起到了一个 Cache 的作用，再次读取时由它直接发送即可，不用再进行新的寻址输出，此时数据重写操作则可在预充电阶段完成。

5、数据输入（写）

数据写入的操作也是在 tRCD 之后进行，但此时没有了 CL（记住，CL 只出现在读取操作中），行寻址与列寻址的时序图和上文一样，只是在列寻址时，WE#为有效状态。

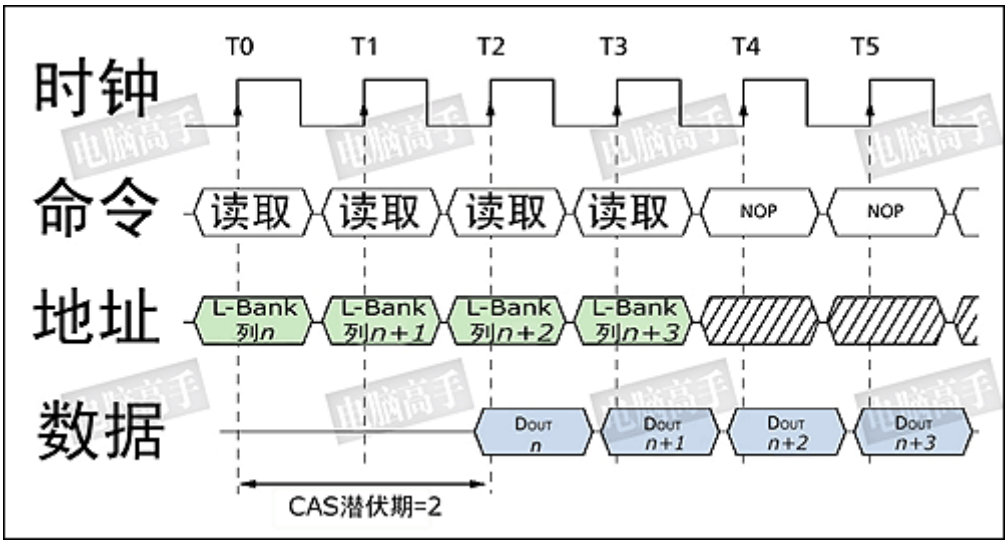


数据写入的时序图

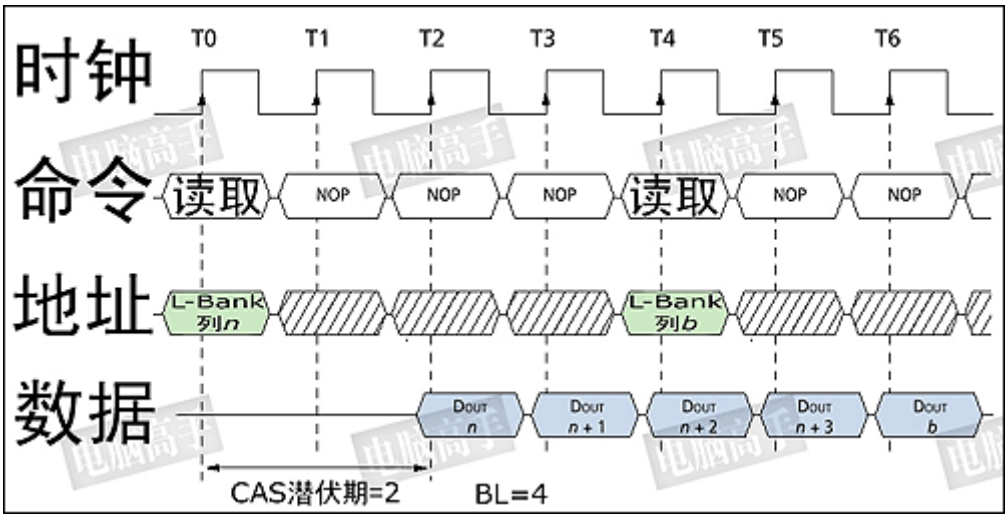
从图中可见，由于数据信号由控制端发出，输入时芯片无需做任何调校，只需直接传到数据输入寄存器中，然后再由写入驱动器进行对存储电容的充电操作，因此数据可以与 CAS 同时发送，也就是说写入延迟为 0。不过，数据并不是即时地写入存储电容，因为选通三极管（就如读取时一样）与电容的充电必须要有一段时间，所以数据的真正写入需要一定的周期。为了保证数据的可靠写入，都会留出足够的写入/校正时间（tWR，Write Recovery Time），这个操作也被称作写回（Write Back）。tWR 至少占用一个时钟周期或再多一点（时钟频率越高，tWR 占用周期越多），有关它的影响将在下文进一步讲述。

6、突发长度

突发（Burst）是指在同一行中相邻的存储单元连续进行数据传输的方式，连续传输所涉及到存储单元（列）的数量就是突发长度（Burst Lengths，简称 BL）。
在目前，由于内存控制器一次读/写 P-Bank 位宽的数据，也就是 8 个字节，但是在现实中小于 8 个字节的数据很少见，所以一般都要经过多个周期进行数据的传输。上文讲到的读/写操作，都是一次对一个存储单元进行寻址，如果要连续读/写就还要对当前存储单元的下一个单元进行寻址，也就是要不断的发送列地址与读/写命令（行地址不变，所以不用再对行寻址）。虽然由于读/写延迟相同可以让数据的传输在 I/O 端是连续的，但它占用了大量的内存控制资源，在数据进行连续传输时无法输入新的命令，效率很低（早期的 FPE/EDO 内存就是以这种方式进行连续的数据传输）。为此，人们开发了突发传输技术，只要指定起始列地址与突发长度，内存就会依次地自动对后面相应数量的存储单元进行读/写操作而不再需要控制器连续地提供列地址。这样，除了第一笔数据的传输需要若干个周期（主要是之前的延迟，一般的是 tRCD+CL）外，其后每个数据只需一个周期的即可获得。在很多北桥芯片的介绍中都有类似于 X-1-1-1 的字样，就是指这个意思，其中的 X 代表就代表第一笔数据所用的周期数。



非突发连续读取模式：不采用突发传输而是依次单独寻址，此时可等效于 BL=1。虽然可以让数据是连续的传输，但每次都要发送列地址与命令信息，控制资源占用极大



突发连续读取模式：只要指定起始列地址与突发长度，寻址与数据的读取自动进行，而只要控制好两段突发读取命令的间隔周期（与 BL 相同）即可做到连续的突发传输

至于 BL 的数值，也是不能随便设或在数据进行传输前临时决定。在上文讲到的初始化过程中的 MRS 阶段就要对 BL 进行设置。目前可用的选项是 1、2、4、8、全页（Full Page），常见的设定是 4 和 8。顺便说一下，BL 能否更改与北桥芯片的设计有很大关系，不是每个北桥都能像调整 CL 那样来调整 BL。某些芯片组的 BL 是定死而不可改的，比如 Intel 芯片组的 BL 基本都为 4，所以在相应的主板 BIOS 中也就不会有 BL 的设置选项。而由于目前的 SDRAM 系统的数据传输是以 64bit/周期进行，所以在一些 BIOS 也把 BL 用 QWord（4 字，即 64bit）来表示。如 4QWord 就是 BL=4。

提示与误区：Full Page 与 Page 的含义
Full Page（全页）突发传输是指 L-Bank 里的一行中所有存储单元从头至尾进行连续传输，至于具体的突发长度（一行中的存储单元数量）则要看内存芯片的具体设计。 不过这种针对芯片的页定义是狭义的。我们常用的则是广义上的页，文章中已经讲到，内存系统的每次传输都是以一个 P-Bank 位宽为单位的，需要多颗芯片集体工作。在每次寻址时，P-Bank 内每个芯片所得到的 L-Bank 地址与行地址都是相同的。这样在全页操作中，就等于对 P-Bank 所包含的每个芯片内同一 L-Bank 里同一行的所有存储单元读/写，而这些存储单元的总和就是相对于整体系统而言的页，本文中所指的页就是指这种广义上的页。 在一些文章中，把这种广义上的页说成是 DIMM 上所有相同行地址的行集合，显然是不对的，它没有 P-Bank 的划分也没有 L-Bank 的界定，因为不同的 P-Bank 或不同的 L-Bank 中也有相同的行地址，而一次寻址只能以 P-Bank——L-Bank——Row 的步骤进行，所以不同 P-Bank 或不同 L-Bank 内即使地址相同的行也不能同时工作，页也就无从谈起了。 — Page sizes of 2 KB, 4 KB, 8 KB and 16 KB (individually selected for every row) Intel 845 芯片组 MCH 的资料：它可以支持 2、4、8、16KB 的页容量 上图就是 845 有关页容量的支持范围，这个容量就是一页中所有存储单元的容量总合。至于页容量具体是多少则要看 DIMM 的设计。但页容量的计算公式通过上面的讲解就很容易得出：L-Bank 中一行的存储单元数×存储单元的容量×组成 P-Bank 所需要的芯片数。 以前文那张结构图的规格为例。一行共有 2048 个存储单元（列地址为 2^{11}），每个单元容量是 4bit（×4）。要组成一个 P-Bank，需要 16 颗这样的芯片，那么此这种规格芯片生产的 DIMM 所能提供的页容量就是：2048×4bit×16=16KB。

另外，在 MRS 阶段除了要设定 BL 数值之外，还要具体确定读/写操作的模式以及突发传输的模式。突发读/突发写，表示读与写操作都是突发传输的，每次读/写操作持续 BL 所设定的长度，这也是常规的设定。突发读/单一写，表示读操作是突发传输，写操作则只是一个个单独进行。突发传输模式代表着突发周期内所涉及到的存储单元的传输顺序。顺序传输是指从起始单元开始顺序读取。假如 BL=4，起始单元编号是 n，顺序就是 n、n+1、n+2、n+3。交错传输就是打乱正常的顺序进行数据传输（比如第一个进行传输的单元是 n，而第二个进行传输的单元是 n+2 而不是 n+1），至于交错的规则在 SDRAM 规范中有详细的定义表，但在这此出于必要性与篇幅的考虑就不列出了。

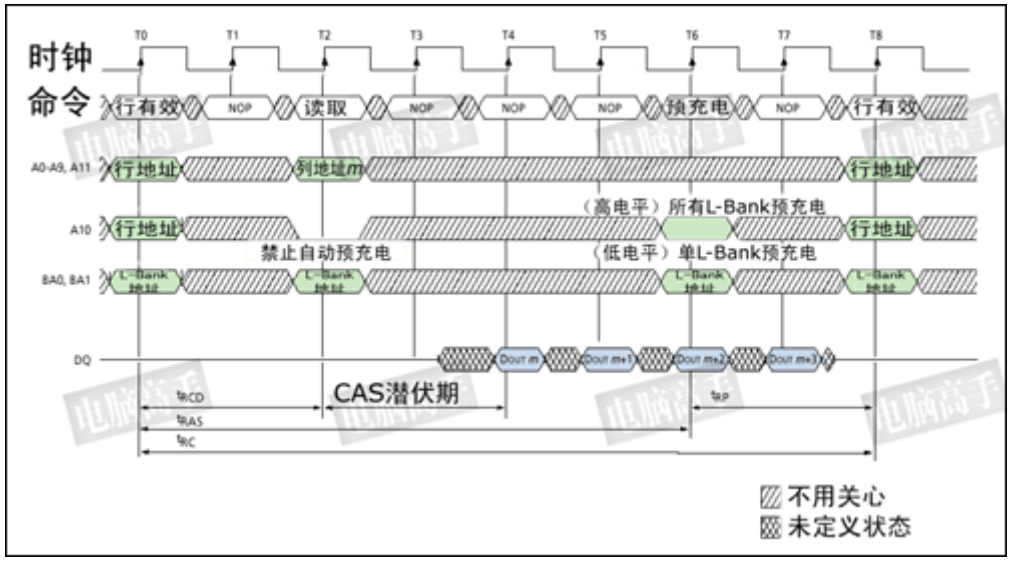
第 7 页：SDRAM 与内存基础概念（六）

7、预充电

由于 SDRAM 的寻址具体独占性，所以在进行完读写操作后，如果要对同一 L-Bank 的另一行进行寻址，就要将原来有效（工作）的行关闭，重新发送行/列地址。L-Bank 关闭现有工作行，准备打开新行的操作就是预充电（Precharge）。预充电可以通过命令控制，也可以通过辅助设定让芯片在每次读写操作之后自动进行预充电。实际上，预充电是一种对工作行中所有存储体进行数据重写，并对行地址进行复位，同时释放 S-AMP（重新加入比较电压，一般是电容电压的 1/2，以帮助判断读取数据的逻辑电平，因为 S-AMP 是通过一个参考电压与存储体位线电压的比较来判断逻辑值的），以准备新行的工作。具体而言，就是将 S-AMP 中的数据回写，即使是没有工作过的存储体也会因行选通而使存储电容受到干扰，所以也需要 S-AMP 进行读后重写。此时，电容的电量（或者说其产生的电压）将是判断逻辑状态的依据（读取时也需要），为此要设定一个临界值，一般为电容电量的 1/2，超过它的为逻辑 1，进行重写，否则为逻辑 0，不进行重写（等于放电）。为此，现在基本都将电容的另一端接入一个指定的电压（即 1/2 电容电压），而不是接地，以帮助重写时的比较与判断。

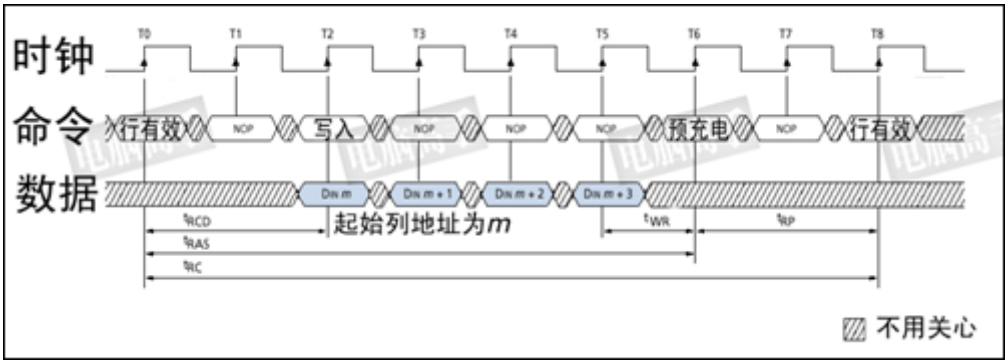
现在我们再回过头看看读写操作时的命令时序图，从中可以发现地址线 A10 控制着是否进行在读写之后当前 L-Bank 自动进行预充电，这就是上文所说的“辅助设定”。而在单独的预充电命令中，A10 则控制着是对指定的 L-Bank 还是所有的 L-Bank（当有多个 L-Bank 处于有效/活动状态时）进行预充电，前者需要提供 L-Bank 的地址，后者只需将 A10 信号置于高电平。

在发出预充电命令之后，要经过一段时间才能允许发送 RAS 行有效命令打开新的工作行，这个间隔被称为 tRP（Precharge command Period，预充电有效周期）。和 tRCD、CL 一样，tRP 的单位也是时钟周期数，具体值视时钟频率而定。



（上图可点击放大）

读取时预充电时序图：图中设定：CL=2、BL=4、tRP=2。自动预充电时的开始时间与此图一样，只是没有了单独的预充电命令，并在发出读取命令时，A10 地址线要设为高电平（允许自动预充电）。可见控制好预充电启动时间很重要，它可以在读取操作结束后立刻进入新行的寻址，保证运行效率。

误区：读写情况下都要考虑写回延迟
有些文章强调由于写回操作而使读/写操作后都有一定的延迟，但从本文的介绍中写可以看出，即使是读后立即重写的设计，由于是与数据输出同步进行，并不存在延迟。只有在写操作后进行其他的操作时，才会有这方面的影响。写操作虽然是 0 延迟进行，但每笔数据的真正写入则需要一个足够的周期来保证，这段时间就是写回周期（tWR）。所以预充电不能与写操作同时进行，必须要在 tWR 之后才能发出预充电命令，以确保数据的可靠写入，否则重写的数据可能是错的，这就造成了写回延迟。    （上图可点击放大） 数据写入时预充电操作时序图：注意其中的 tWR 参数，由于它的存在，使预充电操作延后，从而造成写回延迟

8、刷新

之所以称为 DRAM，就是因为它要不断进行刷新（Refresh）才能保留住数据，因此它是 DRAM 最重要的操作。

刷新操作与预充电中重写的操作一样，都是用 S-AMP 先读再写。但为什么有预充电操作还要进行刷新呢？因为预充电是对一个或所有 L-Bank 中的工作行操作，并且是不定期的，而刷新则是有固定的周期，依次对所有行进行操作，以保留那些久久没经历重写的存储体中的数据。但与所有 L-Bank 预充电不同的是，这里的行是指所有 L-Bank 中地址相同的行，而预充电中各 L-Bank 中的工作行地址并不是一定是相同的。

那么要隔多长时间重复一次刷新呢？目前公认的标准是，存储体中电容的数据有效保存期上限是 64ms（毫秒，1/1000 秒），也就是说每一行刷新的循环周期是 64ms。这样刷新速度就是：行数量/64ms。我们在看内存规格时，经常会看到 4096 Refresh Cycles/64ms 或 8192 Refresh Cycles/64ms 的标识，这里的 4096 与 8192 就代表这个芯片中每个 L-Bank 的行数。刷新命令一次对一行有效，发送间隔也是随总行数而变化，4096 行时为 15.625μs（微秒，1/1000 毫秒），8192 行时就为 7.8125μs。

刷新操作分为两种：自动刷新（Auto Refresh，简称 AR）与自刷新（Self Refresh，简称 SR）。不论是何种刷新方式，都不需要外部提供行地址信息，因为这是一个内部的自动操作。对于 AR， SDRAM 内部有一个行地址生成器（也称刷新计数器）用来自动的依次生成行地址。

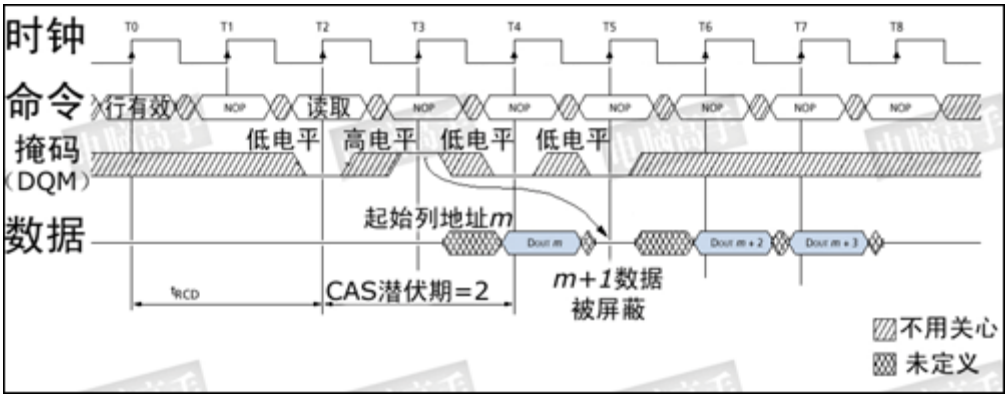
由于刷新是针对一行中的所有存储体进行，所以无需列寻址，或者说 CAS 在 RAS 之前有效。所以，AR 又称 CBR（CAS Before RAS，列提前于行定位）式刷新。由于刷新涉及到所有 L-Bank，因此在刷新过程中，所有 L-Bank 都停止工作，而每次刷新所占用的时间为 9 个时钟周期（PC133 标准），之后就可进入正常的工作状态，也就是说在这 9 个时钟期间内，所有工作指令只能等待而无法执行。64ms 之后则再次对同一行进行刷新，如此周而复始进行循环刷新。显然，刷新操作肯定会对 SDRAM 的性能造成影响，但这是没办法的事情，也是 DRAM 相对于 SRAM（静态内存，无需刷新仍能保留数据）取得成本优势的同时所付出的代价。

SR 则主要用于休眠模式低功耗状态下的数据保存，这方面最著名的应用就是 STR（Suspend to RAM，休眠挂起于内存）。在发出 AR 命令时，将 CKE 置于无效状态，就进入了 SR 模式，此时不再依靠系统时钟工作，而是根据内部的时钟进行刷新操作。在 SR 期间除了 CKE 之外的所有外部信号都是无效的（无需外部提供刷新指令），只有重新使 CKE 有效才能退出自刷新模式并进入正常操作状态。

9、数据掩码

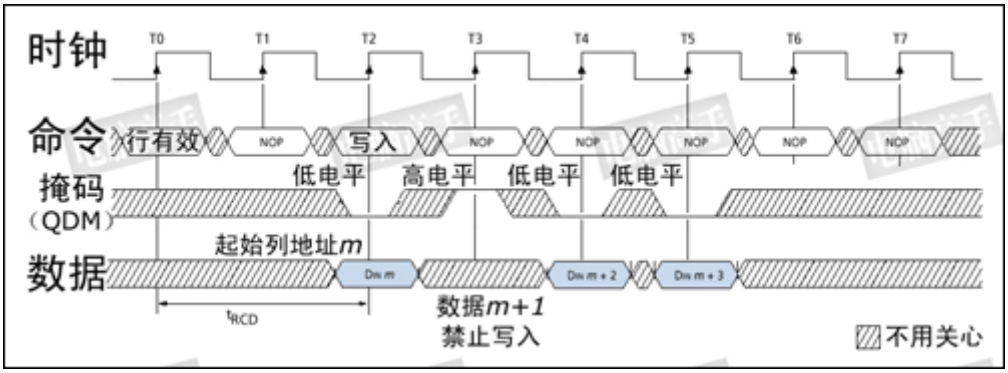
在讲述读/写操作时，我们谈到了突发长度。如果 BL=4，那么也就是说一次就传送 4×64bit 的数据。但是，如果其中的第二笔数据是不需要的，怎么办？还都传输吗？为了屏蔽不需要的数据，人们采用了数据掩码（Data I/O Mask，简称 DQM）技术。通过 DQM，内存可以控制 I/O 端口取消哪些输出或输入的数据。这里需要强调的是，在读取时，被屏蔽的数据仍然会从存储体传出，只是在“掩码逻辑单元”处被屏蔽。DQM 由北桥控制，为了精确屏蔽一个 P-Bank 位宽中的每个字节，每个 DIMM 有 8 个 DQM 信号线，每个信号针对一个字节。这样，对于 4bit 位宽芯片，两个芯片共用一个 DQM 信号线，对于 8bit 位宽芯片，一个芯片占用一个 DQM 信号，而对于 16bit 位宽芯片，则需要两个 DQM 引脚。

SDRAM 官方规定，在读取时 DQM 发出两个时钟周期后生效，而在写入时，DQM 与写入命令一样是立即成效。



（上图可点击放大）

读取时数据掩码操作，DQM 在两个周期后生效，突发周期的第二笔数据被取消



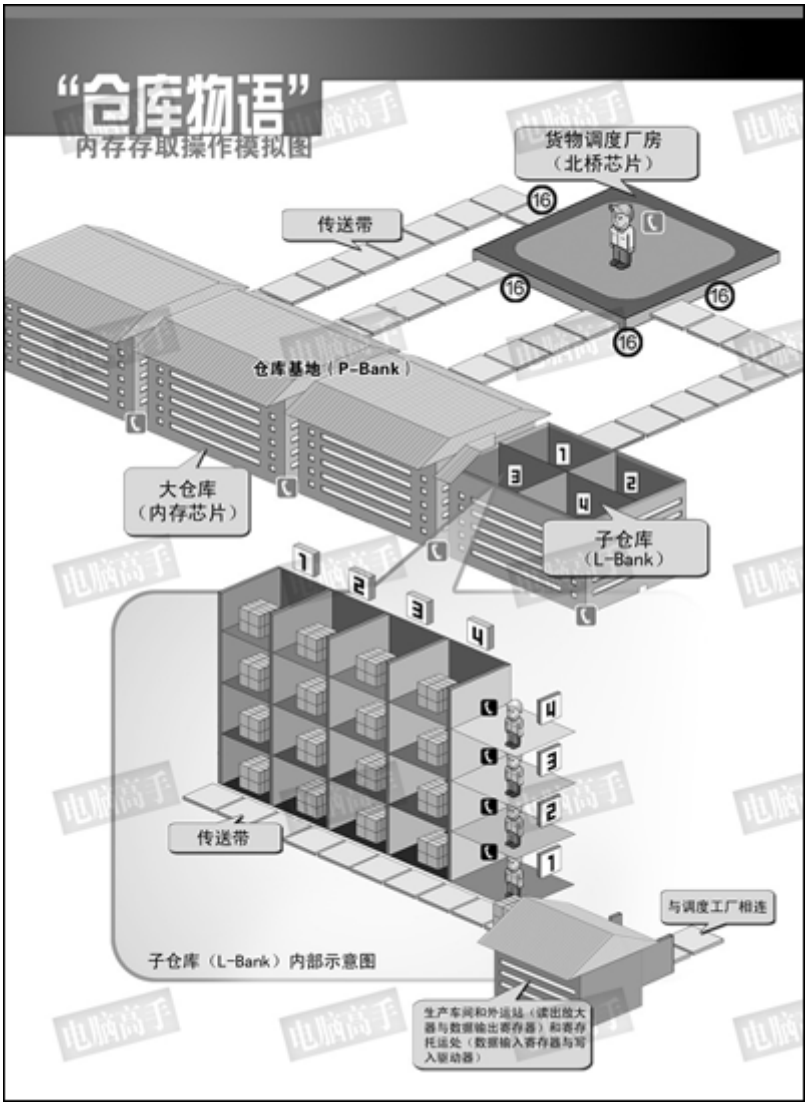
（上图可点击放大）

写入时数据掩码操作，DQM 立即生效，突发周期的第二笔数据被取消

有关内存内部的基本操作就到此结束，其实还有很多内存的操作没有描述，但都不是很重要了，限于篇幅与必要性，我们不在此介绍，有兴趣的读者可以自行查看相关资料。

仓库物语

货物基地（主板）连接着物资（数据）的供求方。基地的货物调度厂房（北桥芯片）掌管着若干个用于临时供货/生产与存储的仓库基地（P-Bank），它们通常隶属于某一仓储集团(DIMM),这种基地与调度厂房之间必须由 64 条传送带联系着(P-Bank 位宽），每条传送带一次只能运送一个标准的货物（1bit 数据），而且一次至少要传送 64 个标准货物，这是它们之间的约定，仓库基地必须满足。



（上图可点击放大）

上图就是这样的一个仓库基地（P-Bank），它由 4 个大仓库（内存芯片）组成，它们的规模都相当大，每个大仓库为基地提供 16 条传送带（芯片位宽为 16bit），总共加起来刚好就是 64 条。每个大仓库里都有四个规模和结构相同的子仓库（L-Bank），它们都被统一编了号。而子仓库中有很多层（行），每层里又有很多的储藏间（列），每个储藏间可以放置 16 个标准货物，虽然子仓库的规模很大，但每一层和每一个房间也都编好了号，而且每一层都有一个搬运工在值班。

为了与外界联系方便，仓储集团与调度室设置了专线电话，和一个国家一样，每个仓库基地有一个区号（片选），另外还有四个子仓库号码（L-Bank 地址），是所

有大仓库共享的，一个号码对应所有大仓库中编号相同的子仓库。而专线电话的数量也是四个，这样可保证与某个子仓库通话时不会妨碍给其他子仓库打电话。在子仓库的每层则设立分机给搬运工使用。子仓库的楼下就是传送带，找到货物把它扔到上面。但每个大仓库只有一个传送带，也就是说同一时间内只能有一个子仓库在工作。每个子仓库都有一个自己的生产车间（读出放大器）负责指定货物的生产，并且每个大仓库都有一个外运站（数据输出寄存器）和寄存托运处（数据输入寄存器与写入驱动器）与传送带相连，前者负责货物的输出中转，后者负责所接受货物并寄存然后帮助搬运工运送到指定储藏间。那么它是如何与调度厂房协同工作的呢？

1、需求方有货物请求了，这个请求发送到调度厂房，调度人员开根据货主的要求给指定的子仓库打电话，电话号码是：区号+子仓库号码+楼层分机（片选+L-Bank寻址+行有效/选通）。那一层的搬运工接到电话后就开始准备工作。

2、当搬运工点亮所有储藏间的门牌（tRCD）之后，调度人员会告诉搬运工，货物放在哪个储藏间里（列寻址），如果货物很多，并且是连续存放的，调度员会通知搬运工：“一会儿要搬的时候，从起始房间开始连续将后面的 n 个房间的货物都搬出来，我就不再重复了”（突发传输）。但是，他告诉搬运工要等一下，要求所有大仓库的人员统一行动，先别出货。

3、根据事先的规定，搬运工在经过指定的时间后开始将货物扔到传送带上，传送带开始运转并将货物送到生产车间，由它来复制出全新的货物，然后再送到传送带上通过外运站向调度厂房运去。人们通常把从搬运工找到具体储藏间开始，到货物真正出现在送往调度厂房的传送带上的这段时间称之为“输出潜伏期”（CL），而从值班人把货物扔到传送带到货物开始传向调度厂房的这段时间，被称为“货物输出延迟”（tAC），它体现了值班人员的反应时间和生产车间的效率，也影响着仓库基地所在集团（DIMM）的名声。

4、在这个搬运工工作的同时，由于电话对于编号相同的子仓库是并联的，所以其他子仓库相同楼层的搬运工也收到相同的命令，从相同编号的房间搬出货物，运向各自的生产车间。此时，同一批货物同时出现在各自的 16 条传送带上，并整齐地向调度厂房运去。

5、当货物传送完后，原始货物还要送回储藏间保管，这是必须的，但如果没有要求，货物可以一直保留在生产车间，如果再有需要就再生产，而不用再麻烦搬运工了（读出放大器相当于一个 Cache）。调度人员接着会进行下一批货物的调度，当他发现下一批货物在上次操作的子仓库中，但不在刚才通话的那一层，只能再重新拨电话。这时，他通知各子仓库货物翻新运回，清理生产车间，之后挂断电话（预充电命令），这一切必须要在指定时间里（tRP）完成，然后才能给新的楼层打电话。搬运员接到通知后，就将这一 层中所有房间的货物都拿到生产车间进行翻新（没有货物的就不用翻新），然后再搬回储藏间。干完这一切之后，搬运工挂了电话（关闭行）就可以休息了，他们称这种工作为“货物清理返运”（预充电）。这个工作的速度也要快，否则同样会影响集团名声。当然，这个工作可以让搬运工自动完成（自动预充电），只需调度员在当初下搬运指令时提醒一他：“货物运送完

了，就进行货物清理返运吧，我不管了”（用 A10 地址线）。

6、当有货物要运来存储时，调度员在向子仓库发送货物的同时就给指定的楼层打电话，让他们准备好房间，此时货物已经到了寄存托运处，没有任何的运送延迟（写入延迟=0），搬运工在托运间的帮助下，向指定的储藏间运送货物，这可需要一定的时间了，他们称之为货物堆放时间（tWR），必须给足搬运工们这一时间，而不能在这期间里让他们干其他的工作，否则他们会令货物丢失并罢工……

（注：本插栏是对 DRAM 操作的形象性描述，谨供辅助性理解本专题，严谨的操作说明见上文。另外，在此请各位读者注意，将内存比喻为仓库只是为了形象化描述，而不要把内存等同理解为存储，它们是有本质的不同的，在本文的比喻中，它只是一个临时性仓库，这一点请大家分清，不要因此产生新的错误概念。）

第 8 页：SDRAM 的结构、时序与性能的关系（上）

在讲完 SDRAM 的基本工作原理和主要操作之后，我们现在要重要分析一下 SDRAM 的时序与性能之间的关系，它不在局限于芯片本身，而是从整体的内存系统去分析。这也是广大 DIYer 所关心的话题。比如 CL 值对性能的影响有多大几乎是每个内存论坛都会有讨论，今天我们就详细探讨一下，其中的很多内容同样适用于 DDR 与 RDRAM。这里需要强调一点，对于内存系统整体而言，一次内存访问就是对一个页的访问，这个页的定义已经在解释 Full Page 含义时讲明了。由于在 P-Bank 中，每个芯片的寻址都是一样的，所以可以将页访问“浓缩”等效为对每芯片中指定行的访问，这样可能比较好理解。但为了与官方标准统一，在下文中会经常用页来描述相关的内容，请读者注意理解。

一、影响性能的主要时序参数

所谓的影响性能是并不是指 SDRAM 的带宽，频率与位宽固定后，带宽也就不可更改了。但这是理想的情况，在内存的工作周期内，不可能总处于数据传输的状态，因为要有命令、寻址等必要的过程。但这些操作占用的时间越短，内存工作的效率越高，性能也就越好。

非数据传输时间的主要组成部分就是各种延迟与潜伏期。通过上文的讲述，大家应该很明显看出有三个参数对内存的性能影响至关重要，它们是 tRCD、CL 和 tRP。每条正规的内存模组都会在标识上注明这三个参数值，可见它们对性能的敏感性。

以内存最主要的操作——读取为例。tRCD 决定了行寻址（有效）至列寻址（读/写命令）之间的间隔，CL 决定了列寻址到数据进行真正被读取所花费的时间，tRP 则决定了相同 L-Bank 中不同工作行转换的速度。现在可以想象一下读取时可能遇到的几种情况（分析写入操作时不用考虑 CL 即可）：

1、要寻址的行与 L-Bank 是空闲的。也就是说该 L-Bank 的所有行是关闭的，此时可直接发送行有效命令，数据读取前的总耗时为 tRCD+CL，这种情况我们称之为页命中（PH，Page Hit）。

2、要寻址的行正好是前一个操作的工作行，也就是说要寻址的行已经处于选通有效状态，此时可直接发送列寻址命令，数据读取前的总耗时仅为 CL，这就是所谓的背靠背（Back to Back）寻址，我们称之为页快速命中（PFH，Page Fast Hit）或页直接命中（PDH，Page Direct Hit）。

3、要寻址的行所在的 L-Bank 中已经有一个行处于活动状态（未关闭），这种现象就被称作寻址冲突，此时就必须要进行预充电来关闭工作行，再对新行发送行有效命令。结果，总耗时就是 tRP+tRCD+CL，这种情况我们称之为页错失（PM，Page Miss）。

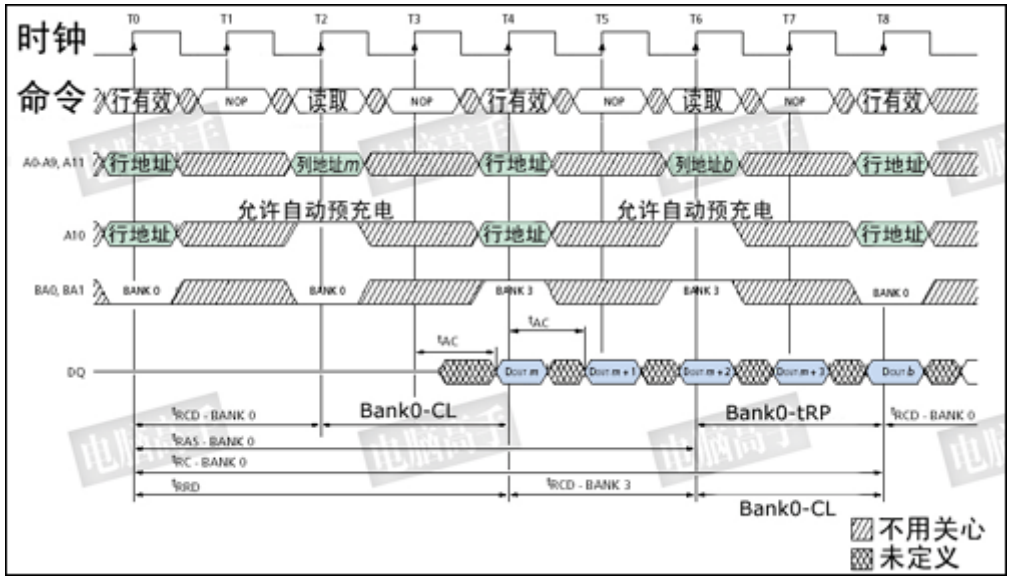
显然，PFH 是最理想的寻址情况，PM 则是最糟糕的寻址情况。上述三种情况发生的机率各自简称为 PHR——PH Rate、PFDR——PFH Rate、PMR——PM Rate。因此，系统设计人员（包括内存与北桥芯片）都尽量想提高 PHR 与 PFHR，同时减少 PMR，以达到提高内存工作效率的目的。

二、增加PHR的方法

显然，这与预充电管理策略有着直接的关系，目前有两种方法来尽量提高 PHR。自动预充电技术就是其中之一，它自动的在每次行操作之后进行预充电，从而减少了日后对同一 L-Bank 不同行寻址时发生冲突的可能性。但是，如果要在当前行工作完成后马上打开同一 L-Bank 的另一行工作时，仍然存在 tRP 的延迟。怎么办？ 此时就需要 L-Bank 交错预充电了。

VIA 的 4 路交错式内存控制就是在一个 L-Bank 工作时，对下一个要工作的 L-Bank 进行预充电。这样，预充电与数据的传输交错执行，当访问下一个 L-Bank 时，tRP 已过，就可以直接进入行有效状态了。目前 VIA 声称可以跨 P-Bank 进行 16 路内存交错，并以 LRU 算法进行预充电管理。

有关 L-Bank 交错预充电（存取）的具体执行在本刊 2001 年第 2 期已有详细介绍，这里就不再重复了。



（上图可点击放大）

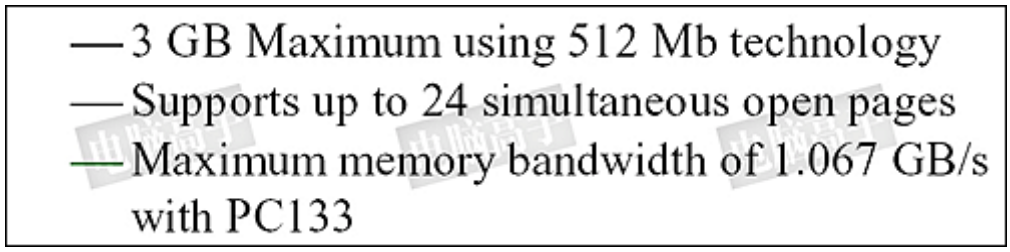
L-Bank交错自动预充电/读取时序图：L-Bank 0 与L-Bank 3 实现了无间隔交错读取，避免了tRP对性能的影响

三、增加PFHR的方法

无论是自动预充电还是交错工作的方法都无法消除 tRCD 所带来的延迟。要解决这个问题，就要尽量让一个工作行在进行预充电前尽可能多的接收多个工作命令，以达到背靠背的效果，此时就只剩下 CL 所造成的读取延迟了（写入时没有延迟）。

如何做到这一点呢？这就是北桥芯片的责任了。在上文的时序图中有一个参数 tRAS（Active to Precharge Command，行有效至预充电命令间隔周期）。它有一个范围，对于 PC133 标准，一般是预充电命令至少要在行有效命令 5 个时钟周期之后发出，最长间隔视芯片而异（基本在 120000ns 左右），否则工作行的数据将有丢失的危险。那么这也就意味着一个工作行从

有效（选通）开始，可以有 120000ns 的持续工作时间而不用进行预充电。显然，只要北桥芯片不发出预充电（包括允许自动预充电）的命令，行打开的状态就会一直保持。在此期间的对该行的任何读写操作也就不会有 tRCD 的延迟。可见，如果北桥芯片在能同时打开的行（页）越多，那么 PFHR 也就越大。需要强调的是，这里的同时打开不是指对多行同时寻址（那是不可能的），而是指多行同时处于选通状态。我们可以看到一些 SDRAM 芯片组的资料中会指出可以同时打开多少个页的指标，这可以说是决定其内存性能的一个重要因素。



Intel 845 芯片组 MCH 的资料：其中表明它可以支持 24 个页面同时处于打开状态

但是，可同时打开的页数也是有限制的。从 SDRAM 的寻址原理讲，同一 L-Bank 中不可能有两个打开的行（S-AMP 只能为一行服务），这就限制了可同时打开的页面总数。以 SDRAM 有 4 个 L-Bank，北桥最多支持 8 个 P-Bank 为例，理论上最多只能有 32 个页面能同时处于打开的状态。而如果只有一个 P-Bank，那么就只剩下 8 个页面，因为有几个 L-Bank 才能有同时打开几个行而互不干扰。Intel 845 的 MHC 虽然可以支持 24 个打开的页面，那也是指 6 个 P-Bank 的情况下（845MCH 只支持 6 个 P-Bank）。可见 845 已经将同时打开页数发挥到了极致。

不过，同时打开页数多了，也对存取策略提出了一定的要求。理论上，要尽量多地使用已打开的页来保证最短的延迟周期，只有在数据不存在（读取时）或页存满了（写入时）再考虑打开新的指定页，这也就是变向的连续读/写。而打开新页时就必须要关闭一个打开的页，如果此时打开的页面已是北桥所支持的最大值但还不到理论极限的话，就需要一个替换策略，一般都是用 LRU 算法来进行，这与 VIA 的交错控制大同小异。

提示： LRU 算法与替换策略
LRU 是 Least Recently Used（近期最少使用）的简称。一般常用于 CPU 的 Cache 中每个 Cache 行（Cache 中基本的存储单元）的替换策略。以 Cache 中的操作为例，它为每个 Cache 行设置一个计数器，Cache 每命中一次，命中行计数器清零，其它各行计数器加 1，因此它是未访问次数计数器。当需要替换时，比较各特定行的计数值，将计数值最大的行换出。内存页面的替换与预充电控制原理也基本一样，设立相应的地址寄存器以存储打开页面（L-Bank）的地址，然后给每个寄存器设立访问计数器，需要替换时，将最近最少用到的页面关闭，并打开新的页面。但是，对于 845 这样的北桥，由于它能打开的页面数已经达到理论的最大值，替换策略反而简单，因为每个 L-Bank 都有一个工作行，所以打开新行也就是就关闭了同一 L-Bank 中的工作行，而无需 LRU 的帮助，因为别无选择。

第 9 页：SDRAM 的结构、时序与性能的关系（下）

四、内存结构对PHR的影响

这是结构设计上的问题，所以单独来说。在我们介绍 L-Bank 时，曾经提到单一的 L-Bank 会造成严重的寻址冲突。现在，当我们了解了内存寻址的原理后，就不难理解这句话了。如果只有一个 L-Bank，那么除非是背靠背式的操作（PFH），否则 tRP、tRCD、CL（读取时）一个也少不了。

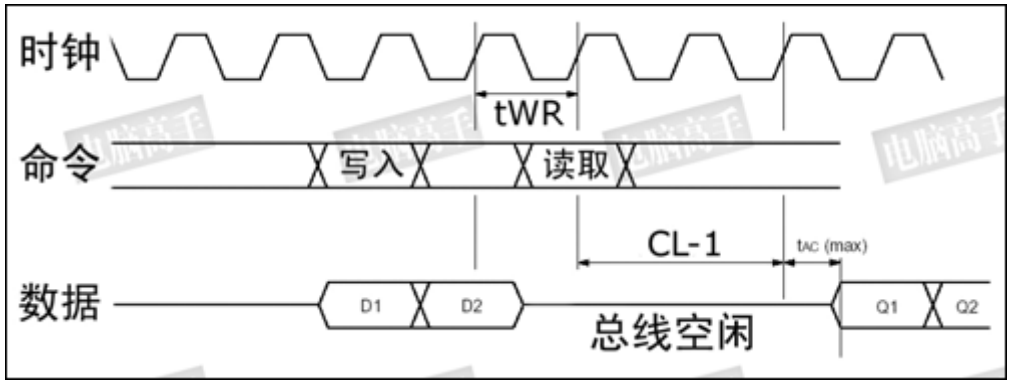
上文中，内存交错之所以能实现就是因为有多个 L-Bank，从这点就可以看出 L-Bank 数量与页命中率之间的关系了。PHR 基本上可以等于“（L-Bank 数-1）/L-Bank 数”。

SDRAM 有 4 个 L-Bank，那么页命中率就是 75%，DDR-Ⅱ SDRAM 最多将有 8 个 L-Bank，PHR 最高为 87.5%。而 RDRAM 则最多有 32 个 L-Bank，PHR 到了惊人的 96.875%，这也是当时 RDRAM 攻击 SDRAM 的一主要方面。

不过，从内存的结构图上可以看出，L-Bank 多了，相应外围辅助的元件也要增加，比如 S-AMP，L-Bank 地址线等等。在 RDRAM 的介绍中，我会讲到 L-Bank 数量增多后所带来的一些新问题。

五、读/写延迟不同对性能所造成的影响

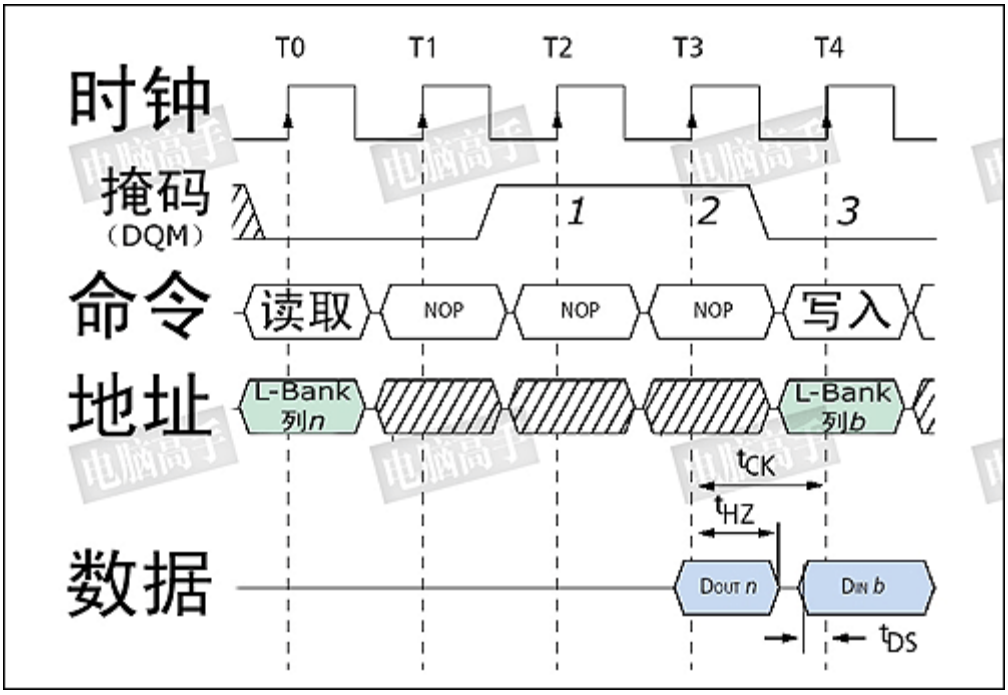
SDRAM 在读取操作时会有 CL 造成的延迟，而在写入时则是 0 延迟。这样，在读操作之后马上进行写操作的话，由于没有写延迟，数据线不会出现空闲的时候，保证了数据总线的利用率。但是，若在写操作之后马上进行读操作的话，即使是背靠背式进行，仍然会由于 tWR 与 CL 的存在而造成间隔，这期间数据总线将是空闲的，利用率受到了影响。



（上图可点击放大）

在先写后读的操作中，由于保证写入的可靠性，读取命令在 tWR 之后发出，并再经过 CL 才能输出数据，本例中 CL=3，造成了两个时钟周期的总线空闲

这里需要着重说明一下，在突发读取过程中，想立刻中断并进行新的读操作，和读后读模式（见“突发连续读取模式图”）一样，只是新的读命令根据需要提前若干个周期发出，经过 CL 后就会自动传输新的数据。但是，若想中断读后立即进行写操作，就需要数据掩码（DQM）来屏蔽写入命令发出时的数据输出，避免总线冲突。根据芯片设计的不同，有时可能会浪费一个周期进行总线 I/O 的调转，此时一个周期的总线空闲也是不可避免的。



（上图可点击放大）

突发读后写时的操作，以本图为例，在最后一个所需数据（本例为第一笔数据）输出前一个周期使 DQM 有效，屏蔽第二笔数据的输出；2、发出写入命令，此时所读取的第二笔数据被屏蔽。3、继续 DQM 以屏蔽第三笔数据的输出。其中 tHZ 表示输出数据与外部电路的连接周期，tDS 表示数据输入准备时间，如果 tHZ+tDS>tCK，那么写入操作就要延后一个周期，这要视芯片的具体设计而定

提示：为什么在中断突发读取时需要 DQM，而中断写入时不用
我们知道行列选通之后，被选中的存储体的电容所存储的信息变被自动导通输出，这是不可阻止的。为避免与输入的数据发生总线冲突，要在内部的数据屏蔽逻辑单元中通过 DQM 的控制来屏蔽掉输出的数据。在写命令发出的同时，已经有数据从 S-AMP 输出，而且下一个存储单元（列）也已被打开导通（开始 tAC 的过程），因此一般需要两个 DQM 周期来屏蔽掉已经传出的数据和将要传出的数据，以避免发生总线冲突。由于写入命令的新的列地址生效，下一个计划要读取的列就自动关闭了，所以最多只需两个 DQM。而在中断突发写操作时，之所以不需要 DQM，是因为读命令会让写允许（WE#）无效，而且数据由北桥芯片控制，在发出读取命令的同时数据的传送就会被中断，这也是为什么新的读取命令可以在上一个读取过程中发过以避免延迟的影响，但不能在上一个写过程中发出的原因。

六、BL对性能的影响

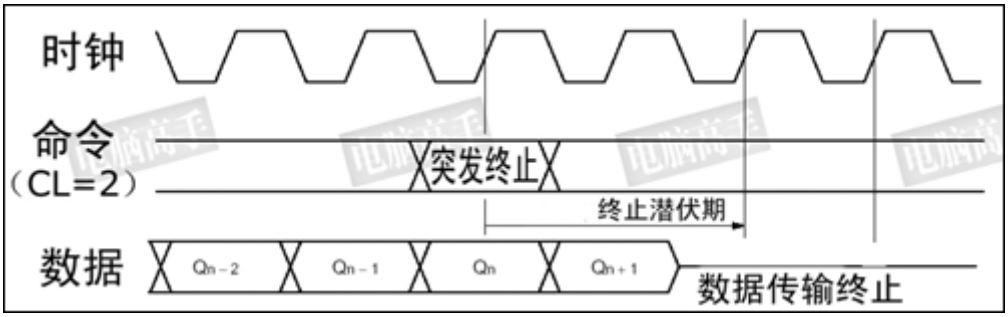
从读/写之间的中断操作我们又引出了 BL（突发长度）对性能影响的话题。首先，BL 的长短与其应用的领域有着很大关系，下表就是目前三个主要的内存应用领域所使用的 BL，这是厂商们经过多年的实践总结出来的。

BL 与相应的工作领域

BL（突发长度）	典型应用
1 或 2 （短）	网关/路由器
4 至 8 （中）	PC 内存
8 至 256 （长）	显卡

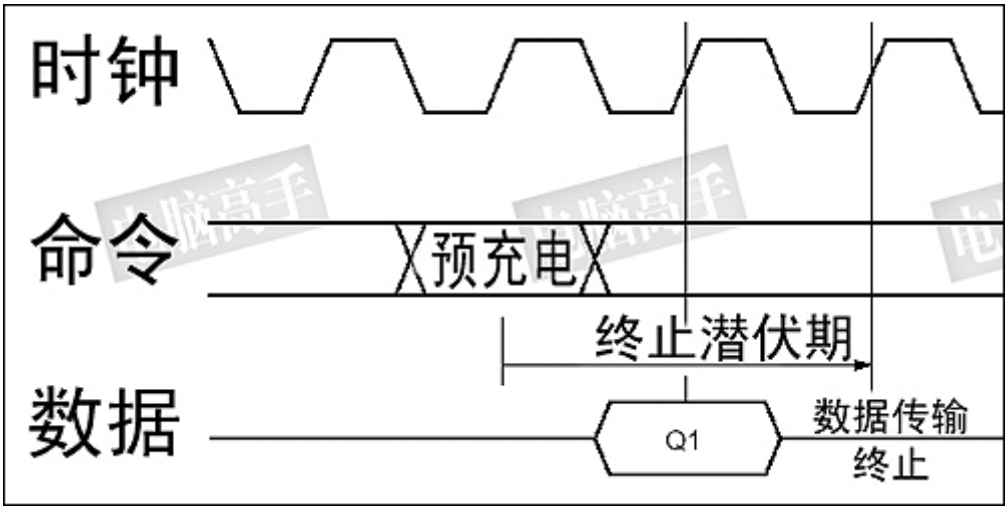
BL 越长，对于连续的大数据量传输很有好处，但是对零散的数据，BL 太长反而会造成总线周期的浪费。以 P-Bank 位宽 64bit 为例，BL=4 时，一个突发操作能传输 32 字节的数据，但如果只需要前 16 个字节，后两个周期是无效的。如果需要 40 字节，需要再多进行一次突发传输，但实际只需要一个传输周期就够了，从而浪费了三个传输周期。而对于 2KB 的数据，BL=4 的设置意味着要每隔 4 个周期发送新的列地址，并重复 63 次。而对于 BL=256，一次突发就可完成，并且不需要中途再进行控制。不少人都因此表示了 BL 设定对性能影响的担心。

但设计人员也不是傻瓜，通过上文的介绍，可以看出他们在这方面的考虑。通过写命令、DQM、读命令的配合/操作，完全可以任意地中断突发周期开始新的操作，而且 DQM 还可以帮我们在 BL 中选择有用的数据，从而最大限度降低突发传输对性能带来的影响。另外，预充电命令与专用的突发传输终止命令都可以用来中断 BL，前者在中断后进行预充电，后者在中断后不进行其他读/写操作。



（上图可点击放大）

专用的突发停止命令可用来中断突发读取，其生效潜伏期与 CL 相同。对于写入则立即有效



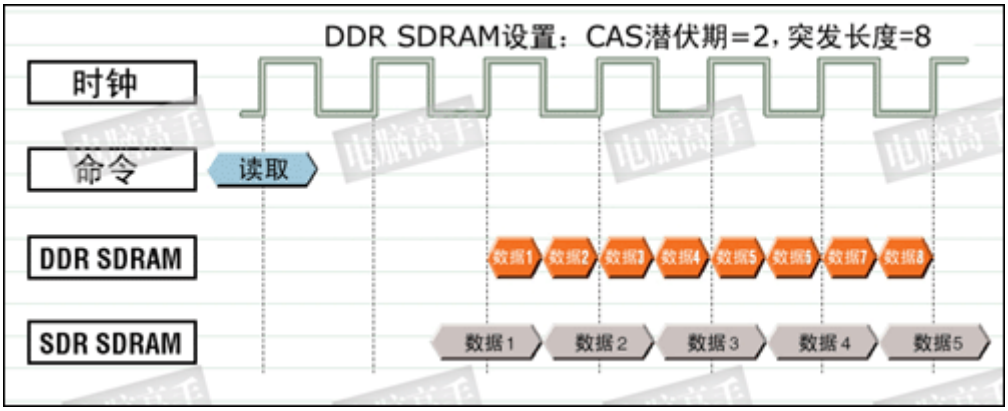
用预充电命令来中断突发读取，生效潜伏期与 CL 相同，要小于或等于 tRP。写入时预充电在最后一个有效写入周期完成，并经过 tWR 之后发出，同时立即中断突发传输

所以，突发周期的中断并不难，但用短 BL 应付大数据量存取需要不断的指令与列寻址配合，而为了取消不需要的传输周期，由于需要运用额外的控制，也将占用不少的控制资源。所以 BL 针对不同应用领域有不同设计的主要目的，就是在保证性能的同时，系统控制资源也能得到合理的运用。

误区：突发写入周期越长存储空间的浪费越严重
在一些文章中认为，突发长度越长，写入时所占用的存储空间就越大，并用硬盘中的簇（含有多个扇区）来举例，这显然是一种误导。每周期写入的数据量由 P-Bank 位宽决定，每个芯片的存储单元都不可能超过这一容量，想多存也不可能。这不像扇区远大于一个字节的容量，而一个字节就可以占用一个扇区的空间。就算只存 1 字节的数据，从 P-Bank 开始就已经浪费 56bit 的空间了，根本不关突发长度的事，而且突发传输并不是不可中断，这完全取决于北桥芯片的控制。不过，由于数据掩码的功能，造成内存物理存储“碎片”的机率则提高了，但这与浪费存储空间是两码事。

第 10 页：如日中天——DDR SDRAM（上）

DDR SDRAM 全称为 Double Data Rate SDRAM，中文名为“双倍数据流 SDRAM”。DDR SDRAM 在原有的 SDRAM 的基础上改进而来。也正因为如此，DDR 能够凭借着转产成本优势来打败昔日的对手 RDRAM，成为当今的主流。由于 SDRAM 的结构与操作在上文已有详细阐述，所以本文只着重讲讲 DDR 的原理和 DDR SDRAM 相对于传统 SDRAM（又称 SDR SDRAM）的不同。

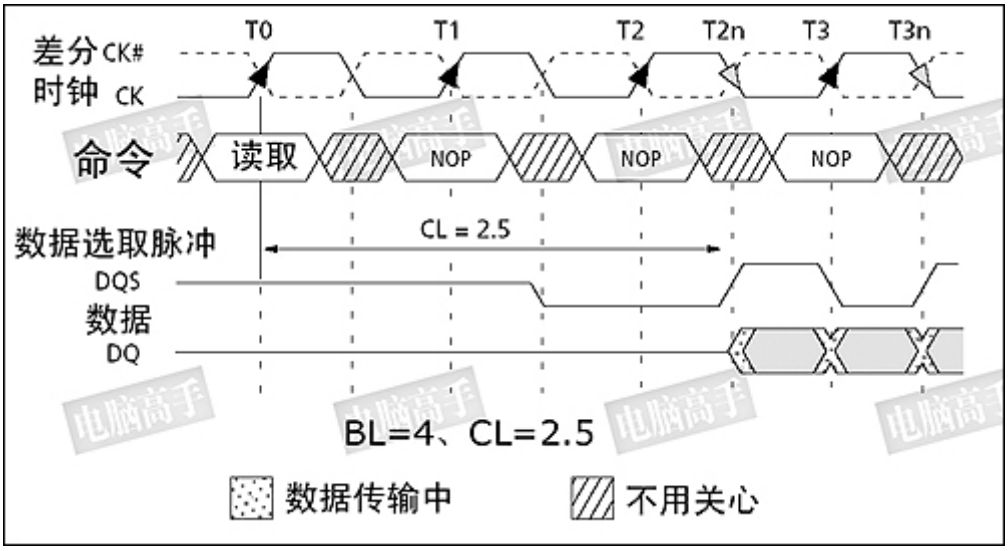


（上图可点击放大）

DDR SDRAM 可在一个时钟周期内传送两次数据

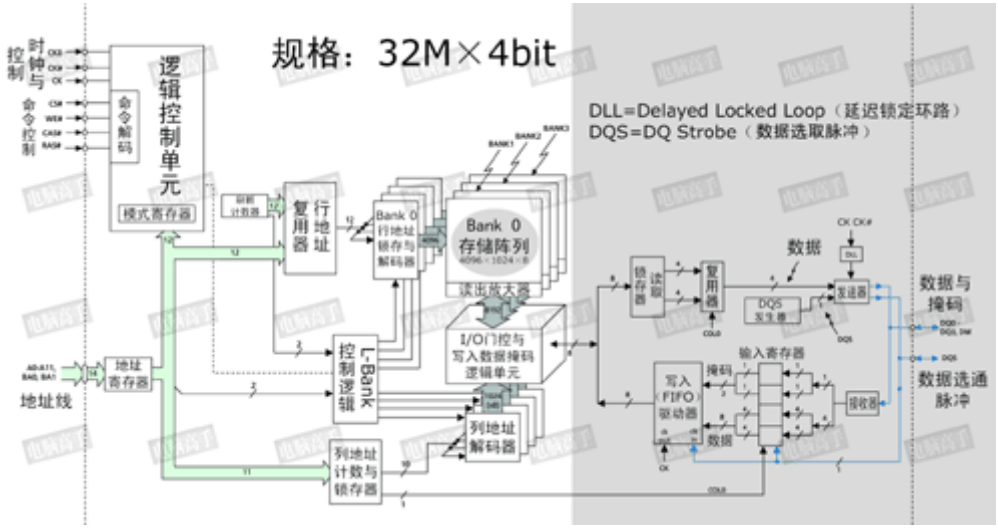
一、DDR的基本原理

有很多文章都在探讨 DDR 的原理，但似乎也不得要领，甚至还带出一些错误的观点。首先我们看看一张 DDR 正规的时序图。



DDR SDRAM 读操作时序图

从中可以发现它多了两个信号： CLK#与 DQS，CLK#与正常 CLK 时钟相位相反，形成差分时钟信号。而数据的传输在 CLK 与 CLK#的交叉点进行，可见在 CLK 的上升与下降沿（此时正好是 CLK#的上升沿）都有数据被触发，从而实现 DDR。在此，我们可以说通过差分信号达到了 DDR 的目的，甚至讲 CLK#帮助了第二个数据的触发，但这只是对表面现象的简单描述，从严格的定义上讲并不能这么说。之所以能实现 DDR，还要从其内部的改进说起。



（上图可点击放大）

DDR 内存芯片的内部结构图，注意比较上文中 SDRAM 的结构图

这也是一颗 128Mbit 的内存芯片，标称规格也与前文的 SDRAM 一样为 32×4bit。从图中可以看出来，白色区域内与 SDRAM 的结构基本相同，但请注意灰色区域，这是与 SDRAM 的不同之处。首先就是内部的 L-Bank 规格。SDRAM 中 L-Bank 存储单元的容量与芯片位宽相同，但在 DDR SDRAM 中并不是这样，存储单元的容量是芯片位宽的一倍，所以在此不能再套用讲解 SDRAM 时 “芯片位宽=存储单元容量” 的公式了。也因此，真正的行、列地址数量也与同规格 SDRAM 不一样了。

以本芯片为例，在读取时，L-Bank 在内部时钟信号的触发下一次传送 8bit 的数据给读取锁存器，再分成两路 4bit 数据传给复用器，由后者将它们合并为一路 4bit 数据流，然后由发送器在 DQS 的控制下在外部时钟上升与下降沿分两次传输 4bit 的数据给北桥。这样，如果时钟频率为 100MHz，那么在 I/O 端口处，由于是上下沿触发，那么就是传输频率就是 200MHz。

现在大家基本明白 DDR SDRAM 的工作原理了吧，这种内部存储单元容量（也可以称为芯片内部总线位宽）=2×芯片位宽（也可称为芯片 I/O 总线位宽）的设计，就是所谓的两位预取（2-bit Prefetch），有的公司则贴切的称之为 2-n Prefetch（n 代表芯片位宽）。

二、DDR SDRAM与SDRAM的不同

DDR SDRAM 与 SDRAM 的不同主要体现在以下几个方面。

DDR SDRAM 与 SDRAM 的主要不同对比表

比较项 \ 内存类型	SDRAM	DDR SDRAM
命令		
全页式突发传输	支持	不支持
时钟信号挂起	支持	不支持
读出数据屏蔽	支持	不支持
写入数据屏蔽	支持	支持
功能		
时钟	单一时钟	差分时钟
预取设计	1-bit	2-bit
数据传输率	1/时钟周期	2/时钟周期
CAS 潜伏期	2、3	1.5、2、2.5、3
写入潜伏期	0	可变
突发长度	1、2、4、8、全页	2、4、8
延迟锁定回路	可选	工作时必需
自动刷新间隔周期	固定	弹性设计（最大值与 SDRAM 的固定值相同）
数据选取脉冲	无	必需
封装与电气特性		
封装类型（≥64Mbit）	TSOP-II 54pin（400mil）	TSOP-II 66pin（400mil）
		CSP 60pin
工作电压	3.3V（LVTTL 接口）	2.5V（SSTL_2 接口）
模组	168pin DIMM	184pin DIMM

注：LVTTL=Low Voltage Transistor-Transistor Logic（低电压晶体管-晶体管逻辑电路）、SSTL=Stub Series Terminated Logic（短线串联终止逻辑电路）

（上图可点击放大）

提示：TSOP-II 与 CSP
TSOP-II：是指小外形薄型封装（Thin Small Outline Package）的第二种方式，引脚在封装的长边两侧，TSOP-I 的引脚则在短边的两侧。 CSP：是指芯片尺寸封装（Chip Scale Package），其封装尺寸与芯片核心尺寸基本相同，所以称 CSP，其内核面积与封装面积的比例约为 1:1.1，凡是符合这一标准的封装都可称之为 CSP。

DDR SDRAM 与 SDRAM 一样，在开机时也要进行 MRS，不过由于操作功能的增多，DDR SDRAM 在 MRS 之前还多了一 EMRS 阶段（Extended Mode Register Set，扩展模式寄存器设置），这个扩展模式寄存器控制着 DLL 的有效/禁止、输出驱动强度、QFC 有效/无效等。

提示与误区：QFC 的含义与作用
QFC 是指 FET Switch Controller（FET 开关控制），低电平有效。用于借助外部 FET（场效应管）开关控制模组上芯片间的相互隔离，没有读写操作时进入隔离状态，以确保芯片间不受相互干扰。QFC 是一个特选功能，厂商都是在接到芯片买家的指定要求后，才在芯片中加入这一功能，并且需要在模装配时进行相关的设计改动（如增加 VddQ 的上拉电阻），所以 DIY 市场上几乎很少见到支持这一功能的 DDR 内存。而在 2002 年 5 月，JEDEC 最新发布的 DDR 规范中，已经不在有 QFC 的定义，而且即使有 FET 开关也将由北桥控制（此时是用来隔离模组的），因此 QFC 已经成为历史。可是，在很多“深入”性介绍中，都将 QFC 认为是一个必要的过程，这显然是错的。

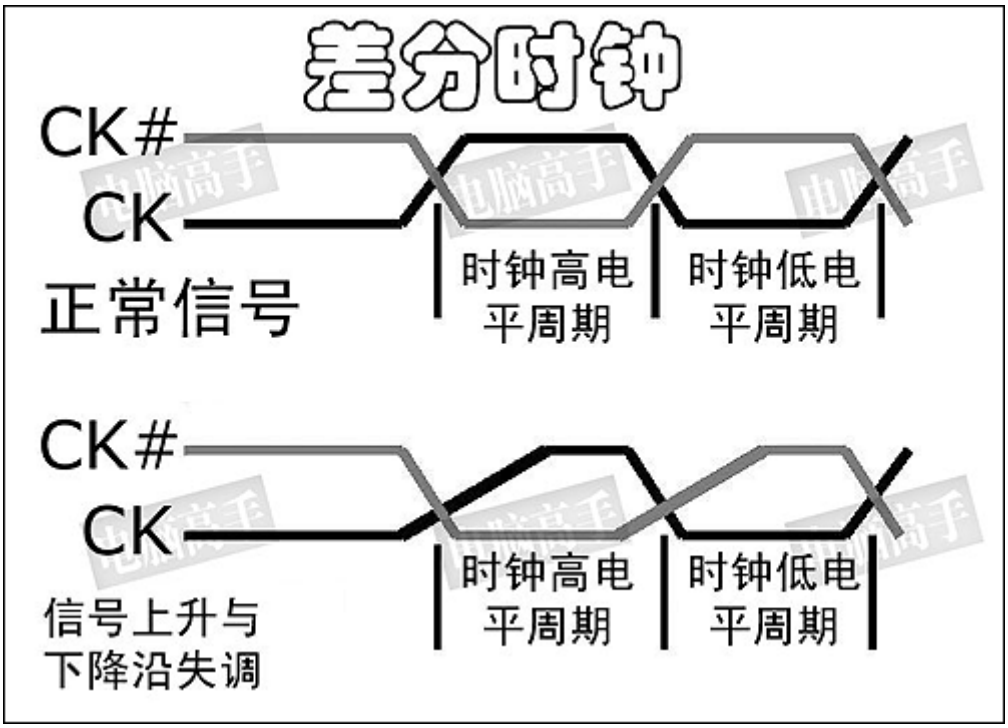
由于 EMRS 与 MRS 的操作方法与 SDRAM 的 MRS 大同小异，在此就不再列出具体的模式表了，有兴趣的话可查看相关的 DDR 内存资料。下面我们就着重说说 DDR SDRAM 的新设计与新功能。

误区：CSP 与 uBGA、WBGA、TinyBGA、FBGA 等是不同的封装技术
实际上，CSP 只是一种封装标准/类型，不涉及具体的封装技术，只要达到它的尺寸标准都可称之为 CSP 封装。近几年出现的 uBGA、WBGA、TinyBGA、FBGA 小型芯片封装技术则是 CSP 的具体表现形式（其实都是 BGA 封装技术的一种），由此可以看出 CSP 并没有固定的封装技术，它自己更不是一个封装技术，只要厂商愿意或有实力，可以开发出更多的符合 CSP 标准的封装技术。

第 11 页：如日中天——DDR SDRAM（下）

1、 差分时钟

差分时钟（参见上文“DDR SDRAM 读操作时序图”）是 DDR 的一个必要设计，但 CK#的作用，并不能理解为第二个触发时钟（你可以在讲述 DDR 原理时简单地这么比喻），而是起到触发时钟校准的作用。由于数据是在 CK 的上下沿触发，造成传输周期缩短了一半，因此必须要保证传输周期的稳定以确保数据的正确传输，这就要求 CK 的上下沿间距要有精确的控制。但因为温度、电阻性能的改变等原因，CK 上下沿间距可能发生变化，此时与其反相的 CK#就起到纠正的作用（CK 上升快下降慢，CK#则是上升慢下降快）。而由于上下沿触发的原因，也使 CL=1.5 和 2.5 成为可能，并容易实现。



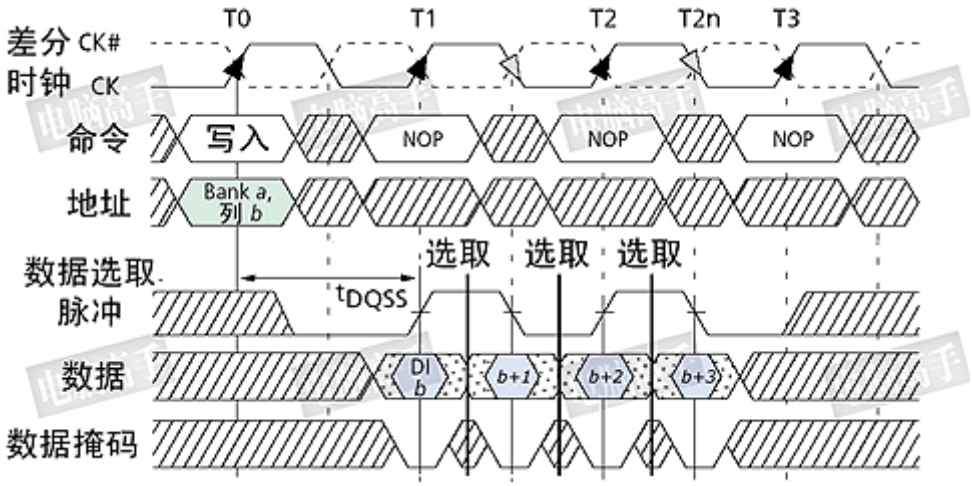
与 CK 反相的 CK#保证了触发时机的准确性

2、 数据选取脉冲（DQS）

DQS 是 DDR SDRAM 中的重要功能，它的功能主要用来在一个时钟周期内准确的区分出每个传输周期，并便于接收方准确接收数据。每一颗芯片都有一个 DQS 信号线，它是双向的，在写入时它用来传送由北桥发来的 DQS 信号，读取时，则由芯片生成 DQS 向北桥发送。完全可以说，它就是数据的同步信号。

在读取时，DQS 与数据信号同时生成（也是在 CK 与 CK#的交叉点）。而 DDR 内存中的 CL 也就是从 CAS 发出到 DQS 生成的间隔，数据真正出现在数据 I/O 总线上相对于 DQS 触发的时间间隔被称为 tAC。注意，这与 SDRAM 中的 tAC 的不同。实际上，DQS 生成时，芯片内部的预取已经完毕了，tAC 是指上文结构图中灰色部分的数据输出时间，由于预取的原因，实际的数据传出可能会提前于 DQS 发生（数据提前于 DQS 传出）。由于是并行传输，DDR 内存对 tAC 也有一定的要求，对于 DDR266，tAC 的允许范围是±0.75ns，对于 DDR333，则是±0.7ns，有关它们的时序图示见前文，其中 CL 里包含了一段 DQS 的导入期。

前文已经说了 DQS 是了保证接收放的选择数据， DQS 在读取时与数据同步传输，那么接收时也是以 DQS 的上下沿为准吗？不，如果以 DQS 的上下沿区分数据周期的危险很大。由于芯片有预取的操作，所以输出时的同步很难控制，只能限制在一定的时间范围内，数据在各 I/O 端口的出现时间可能有快有慢，会与 DQS 有一定的间隔，这也就是为什么要有一个 tAC 规定的原因。而在接收方，一切必须保证同步接收，不能有 tAC 之类的偏差。这样在写入时，芯片不再自己生成 DQS，而以发送方传来的 DQS 为基准，并相应延后一定的时间，在 DQS 的中部为数据周期的选取分割点（在读取时分割点就是上下沿），从这里分隔开两个传输周期。这样做的好处是，由于各数据信号都会有一个逻辑电平保持周期，即使发送时不同步，在 DQS 上下沿时都处于保持周期中，此时数据接收触发的准确性无疑是最高的。

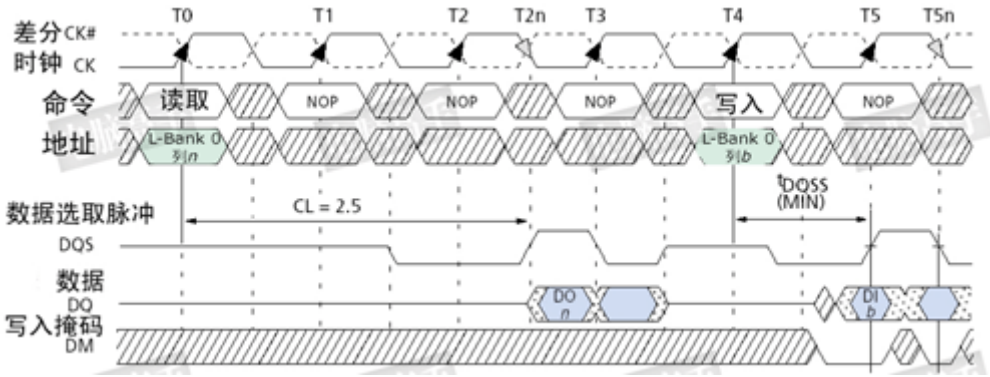


在写入时，以 DQS 的高/低电平期中部为数据周期分割点，而不是上/下沿，但数据的接收触发仍为 DQS 的上/下沿

3、 写入延迟

在上面的 DQS 写入时序图中，可以发现写入延迟已经不是 0 了，在发出写入命令后，DQS 与写入数据要等一段时间才会送达。这个周期被称为 DQS 相对于写入命令的延迟时间（tDQSS，WRITE Command to the first corresponding rising edge of DQS），对于这个时间大家应该很好理解了。

为什么要有这样的延迟设计呢？原因也在于同步，毕竟一个时钟周期两次传送，需要很高的控制精度，它必须要等接收方做好充分的准备才行。tDQSS 是 DDR 内存写入操作的一个重要参数，太短的话恐怕接受有误，太长则会造成总线空闲。tDQSS 最短不能小于 0.75 个时钟周期，最长不能超过 1.25 个时钟周期。有人可能会说，如果这样，DQS 不就与芯片内的时钟不同步了吗？对，正常情况下，tDQSS 是一个时钟周期，但写入时接受方的时钟只用来控制命令信号的同步，而数据的接受则完全依靠 DQS 进行同步，所以 DQS 与时钟不同步也无所谓。不过，tDQSS 产生了一个不利影响——读后写操作延迟的增加，如果 CL=2.5，还要在 tDQSS 基础上加入半个时钟周期，因为命令都要在 CK 的上升沿发出。



（上图可点击放大）

当 CL=2.5 时，读后写的延迟将为 tDQSS+0.5 个时钟周期（图中 BL=2）

另外，DDR 内存的数据真正写入由于要经过更多步骤的处理，所以写回时间（tWR）也明显延长，一般在 3 个时钟周期左右，而在 DDR-Ⅱ规范中更是将 tWR 列为模式寄存器的一项，可见它的重要性。

误区：DDR SDRAM 各种延迟与潜伏期的单位时间减半
一些文章认为，DDR 使数据传输率加倍，那么与之相关的延迟与潜伏期的单位时间也减半，比如 DDR-266 内存，tRCD、CL、tRP 的单位周期为 3.75ns，比 PC133 内存少了一半。这是严重的概念性错误，从我们列举的时序图中可以看出，tRCD、CL、tRP 是以时钟信号来界定的，不能用传输周期去表示，否则 CL=2.5 的参考基准是什么？对于 DDR-266，时钟频率是 133MHz，时钟周期仍是 7.5，和 PC133 的标准一样。那些文章的作者显然是将时钟周期与传输周期弄混了

4、 突发长度与写入掩码

在 DDR SDRAM 中，突发长度只有 2、4、8 三种选择，没有了随机存取的操作（突发长度为 1）和全页式突发。这是为什么呢？因为 L-Bank 一次就存取两倍于芯片位宽的数据，所以芯片至少也要进行两次传输才可以，否则内部多出来的数据怎么处理？而全页式突发事实证明在 PC 内存中是很难用得上的，所以被取消也不希奇。

但是，突发长度的定义也与 SDRAM 的不一样了（见本章节最前那幅 DDR 简示图），它不再指所连续寻址的存储单元数量，而是指连续的传输周期数，每次是一个芯片位宽的数据。对于突发写入，如果其中有不想存入的数据，仍可以运用 DM 信号进行屏蔽。DM 信号和数据信号同时发出，接收方在 DQS 的上升与下降沿来判断 DM 的状态，如果 DM 为高电平，那么之前从 DQS 中部选取的数据就被屏蔽了。有人可能会觉得，DM 是输入信号，意味着芯片不能发出 DM 信号给北桥作为屏蔽读取数据的参考。其实，该读哪个数据也是由北桥芯片决定的，所以芯片也无需参与北桥的工作，哪个数据是有用的就留给北桥自己去选吧。

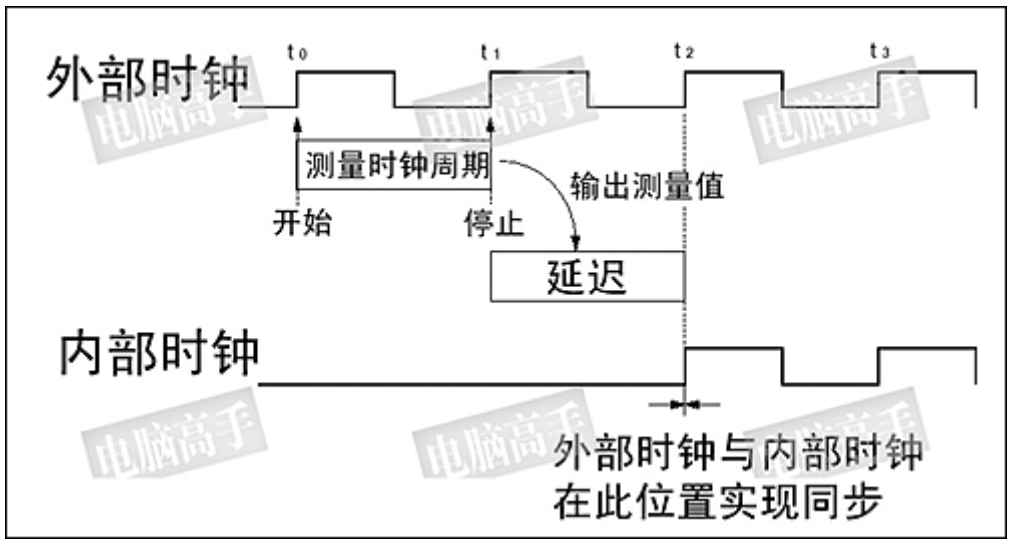
5、 延迟锁定回路（DLL）

DDR SDRAM 对时钟的精确性有着很高的要求，而 DDR SDRAM 有两个时钟，一个是外部的总线时钟，一个是内部的工作时钟，在理论上 DDR SDRAM 这两个时钟应该是同步的，但由于种种原因，如温度、电压波动而产生延迟使两者很难同步，更何况时钟频率本身也有不稳定的情况（SDRAM 也内部时钟，不过因为它的工作/传输频率较低，所以内外同步问题并不突出）。DDR SDRAM 的 tAC 就是因为内部时钟与外部时钟有偏差而引起的，它很可能造成因数据不同步而产生错误的恶果。实际上，不同步就是一种正/负延迟，如果延迟不可避免，那么若是设定一个延迟值，如一个时钟周期，那么内外时钟的上升与下降沿还是同步的。鉴于外部时钟周期也不会绝对统一，所以需要根据外部时钟动态修正内部时钟的延迟来实现与外部时钟的同步，这就是 DLL 的任务。

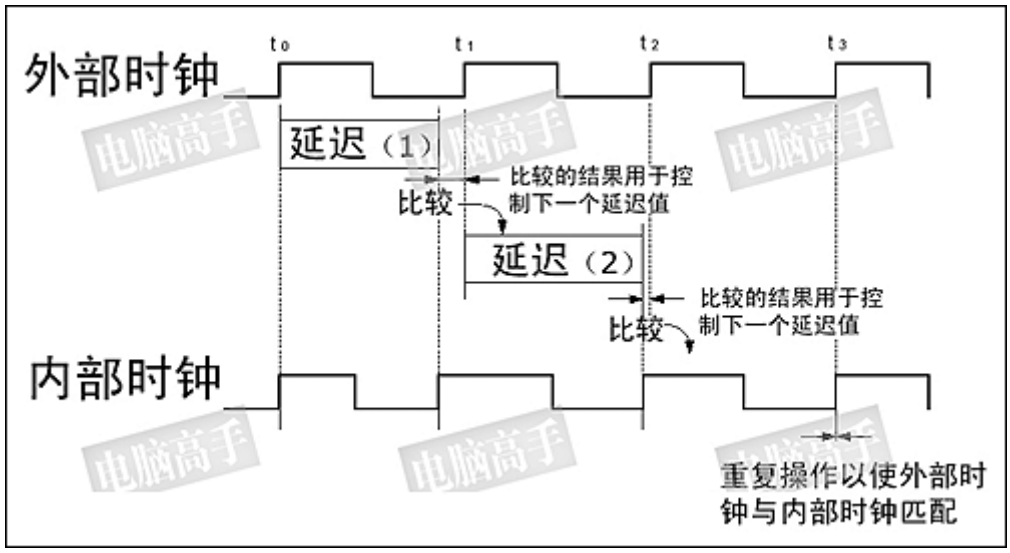
DLL 不同于主板上的 PLL，它不涉及频率与电压转换，而是生成一个延迟量给内部时钟。目前 DLL 有两种实现方法，一个是时钟频率测量法（CFM，Clock Frequency Measurement），一个是时钟比较法（CC，Clock Comparator）。CFM 是测量外部时钟的频率周期，然后以此

周期为延迟值控制内部时钟，这样内外时钟正好就相差了一个时钟周期，从而实现同步。DLL 就这样反复测量反复控制延迟值，使内部时钟与外部时钟保持同步。

CC 的方法则是比较内外部时钟的长短，如果内部时钟周期短了，就将所少的延迟加到下一个内部时钟周期里，然后再与外部时钟做比较，若是内部时钟周期长了，就将多出的延迟从下一个内部时钟中刨除，如此往复，最终使内外时钟同步。



CFM 式 DLL 工作示意图



CC 式 DLL 工作示意图

CFM 与 CC 各有优缺点，CFM 的校正速度快，仅用两个时钟周期，但容易受到噪音干扰，并且如果测量失误，则内部的延迟就永远错下去了。CC 的优点则是更稳定可靠，如果比较失败，延迟受影响的只是一个数据（而且不会太严重），不会涉及到后面的延迟修正，但它的修正时间要比 CFM 长。DLL 功能在 DDR SDRAM 中可以被禁止，但仅限于除错与评估操作，正常工作状态是自动有效的。

误区：DLL 是实现 DDR 传输的关键
“DDR 内存通过内部的 DLL 延时锁相环提供精确的时钟定位，这样就可以在每个时钟周期的上升沿和下降沿传输数据”。“DDR 使用了 DLL 来提供一个数据滤波信号 DQS 来选取数据”。 以上是目前较为流行的对 DDR 工作原理的两种解释，现在大家能看出错误所在吗？两者都把 DLL 的功能夸大了，DLL 只是一个重要的辅助校准设计，与能否实现双沿触发没有关系，它只是保证数据的输出尽量与外部时钟同步。后者则是概念性错误，从 DDR 内存结构图可看出，DQS 不是 DLL 生成的，只是由 DLL 保证其与外部时钟的同步，DQS 由单独的 DQS 发生器生成。

第 12 页：昔日贵族——Rambus DRAM（一）

谈起 DDR SDRAM 与 Rambus DRAM（简称 RDRAM）之间的恩怨，很多人现在还是津津乐道。的确，上一世纪末的内存大战虽胜负已分，但至今仍余波未平。在主流市场 DDR SDRAM 成为王者，RDRAM 则沦为“高端贵族”。

Rambus 公司于 1990 年 3 月成立，之后不久就有了 Rambus 的核心专利——RSL（Rambus Signaling Level，Rambus 发信电平技术）。Rambus 内存最早出现于 1995 年 12 月，那时它与任天堂 64（Nintendo64）游戏机一起发售，但名声不大。从 1996 年 12 月开始，Rambus 与 Intel 合作开发，准备将 Rambus 推广到 PC 领域。到 Rambus 内存真正亮相于 PC 市场时已经是 1999 年 11 月了。

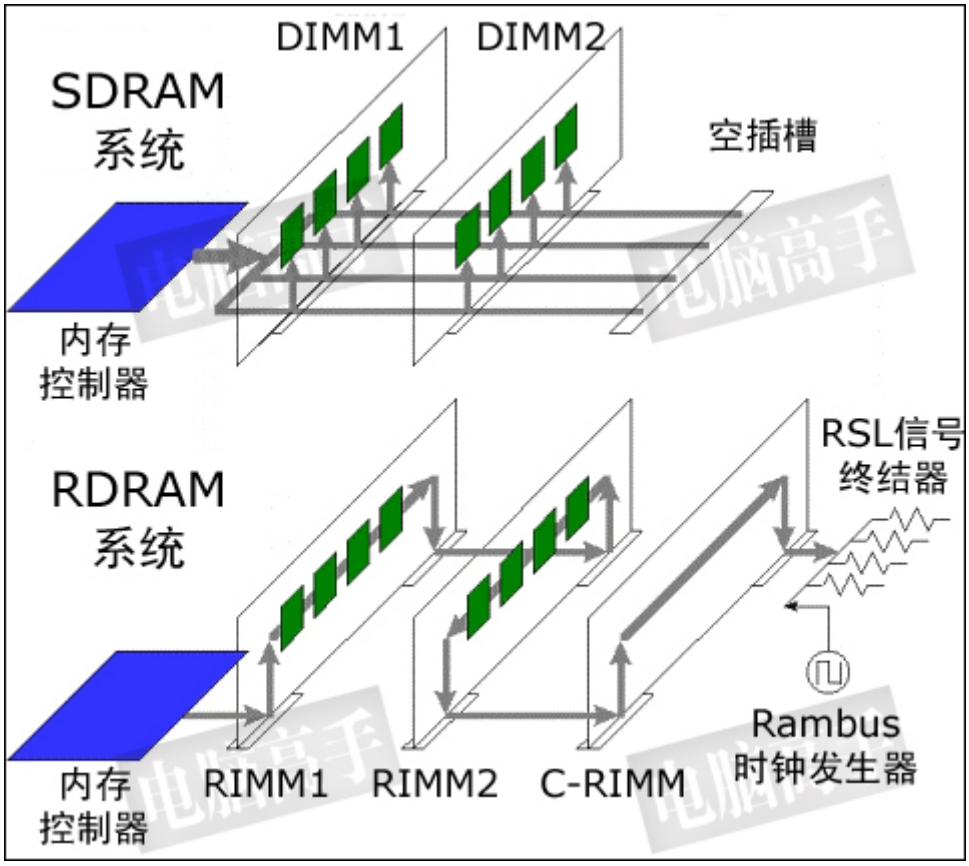
提示：RDRAM 与 Direct RDRAM
在 Intel 有关 RDRAM 的资料中，都会有一个前缀——Direct，这是为什么？原来，1995 年最早上市的 RDRAM 是一个 8bit 位宽的设计，当 Intel 看中 RDRAM 准备将其应用到 PC 领域中时，发现位宽不足，于是在 Intel 的帮助下，Rambus 设计出 Direct RDRAM。在芯片内部多了一个 8bit 数据通道，从而使位宽达到了 16bit。因此，从严格的定义上讲，目前所谓的 RDRAM 都是 Direct RDRAM，但现在连 Rambus 自己也不再这么区分，所以本文的 RDRAM 即指 Direct RDRAM。

一、RDRAM简介

RDRAM 与 DDR SDRAM 一样，也是一种采用双沿触发技术的内存， 但它在结构、控制体系方面相对于传统 SDRAM 有着不小的变化，首先我们来看看它与 SDRAM 之间的简单比较。

内存 / 指标	DDR SDRAM（非 ECC）				RDRAM（单通道）			
模组传输位宽	非 ECC	64bit			非 ECC	16/32/64bit		
	ECC	72bit			ECC	18/36/72bit		
芯片位宽	4、8、16、32bit				16bit/18bit（ECC）			
芯片封装	TSOP-II（66pin）、CSP（60pin）				CSP（54、74、62、92pin）			
刷新周期	64ms				32ms			
内存模组	184pin（22pin 接地）				184pin（72pin 接地）			
工作电压	2.5V				2.5V			
时钟频率	100MHz	133MHz	166MHz	200MHz	300MHz	355.5MHz	400MHz	533MHz
数据传输频率	200MHz	266MHz	333MHz	400MHz	600MHz	711MHz	800MHz	1066MHz
传输带宽	1.6GB/s	2.1GB/s	2.7GB/s	3.2GB/s	1.2GB/s	1.4GB/s	1.6GB/s	2.1GB/s
模组标准	PC1600	PC2100	PC2700	PC3200	RIMM600	RIMM700	RIMM800	RIMM2100
芯片标准	DDR-200	DDR-266	DDR-333	DDR-400	PC600	PC700	PC800	PC1066

（上图可点击放大）

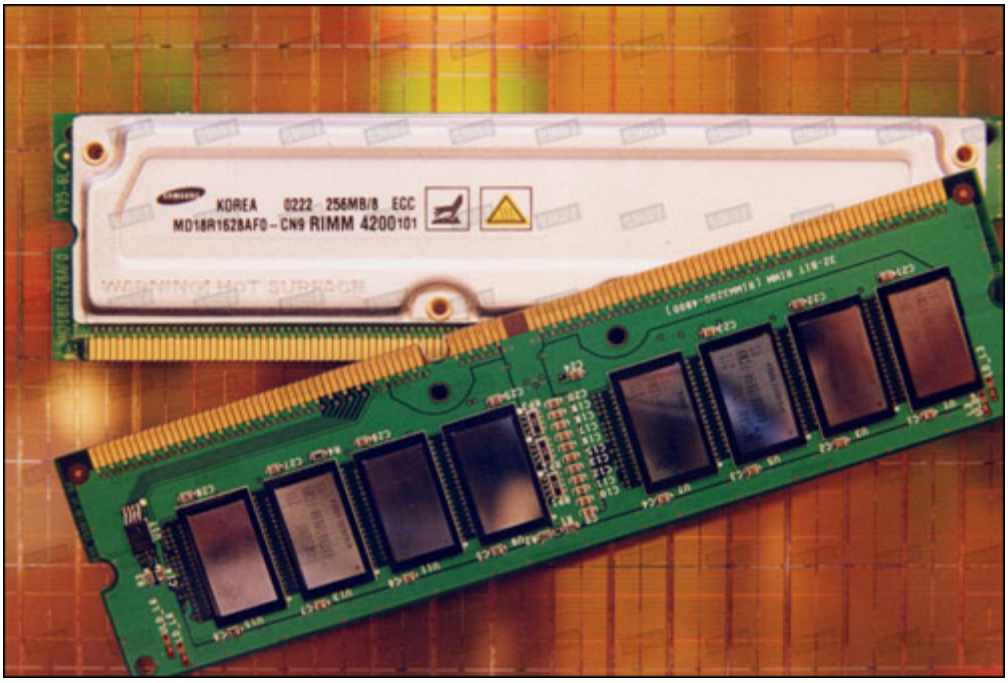


RDRAM 与传统 SDRAM 的架构比较

从架构比较图中，可以看出 RDRAM 在工作方式上与 SDRAM 有了很大不同。SDRAM 需要多颗芯片并联组成 P-Bank 与北桥沟通，而在 RDRAM 架构中每个芯片就是一个单独工作的读写单元，芯片的位宽就是与北桥接口的位宽，所以如果想用 ECC，就要用专门的 ECC 型芯片，也因此有了 16/18bit 的两种规格。而芯片的位宽就是一个 RDRAM 通道的位宽（本文以 16bit 芯片为例进行介绍）。

为了达到更高的容量，在一个通道中将多颗 RDRAM 芯片串起来，形成 RIMM（Rambus Interface Memory Module，Rambus 接口内存模组），如果主板允许，完全可以设计一个超长的内存插槽与模组，但现实中肯定不能这么做，所以在主板上 Rambus 又把模组串起来组成通道。由于是串联的形式，所以要求起始端与终结端形成一个完整的通路，而 RIMM 就是这个通路的串联器，因此 Rambus 要求所有的插槽必须插满，如果没有 RIMM 则用 C-RIMM（Continuity RIMM，RIMM 续连器）代替，以达到联通 RSL 信号并行终结器的目的。工作时，RDRAM 每次寻址一颗芯片，所需要的数据则由通道数据总线传送到北桥，而不像 SDRAM 那样由所在模组直接通过 DIMM 接口传向北桥，也因此 RIMM 的引脚定义几乎是左右对称的。

由于位宽的降低，为保证高带宽，RDRAM 使用了更高的时钟频率（这就意味着它不可能与系统时钟同步，所以只能叫 RDRAM 而不是 RSDRAM），芯片的工作频率明显高于 SDRAM/DDR，这样芯片的工作热量也急剧上升，为此 Rambus 在官方规范中规定 RIMM 必须配备散热片，从而成了现在这个样子。



32bit 位宽 PC1066 芯片标准的 RIMM，它是目前 PC 领域中性能最高的 RDRAM 产品

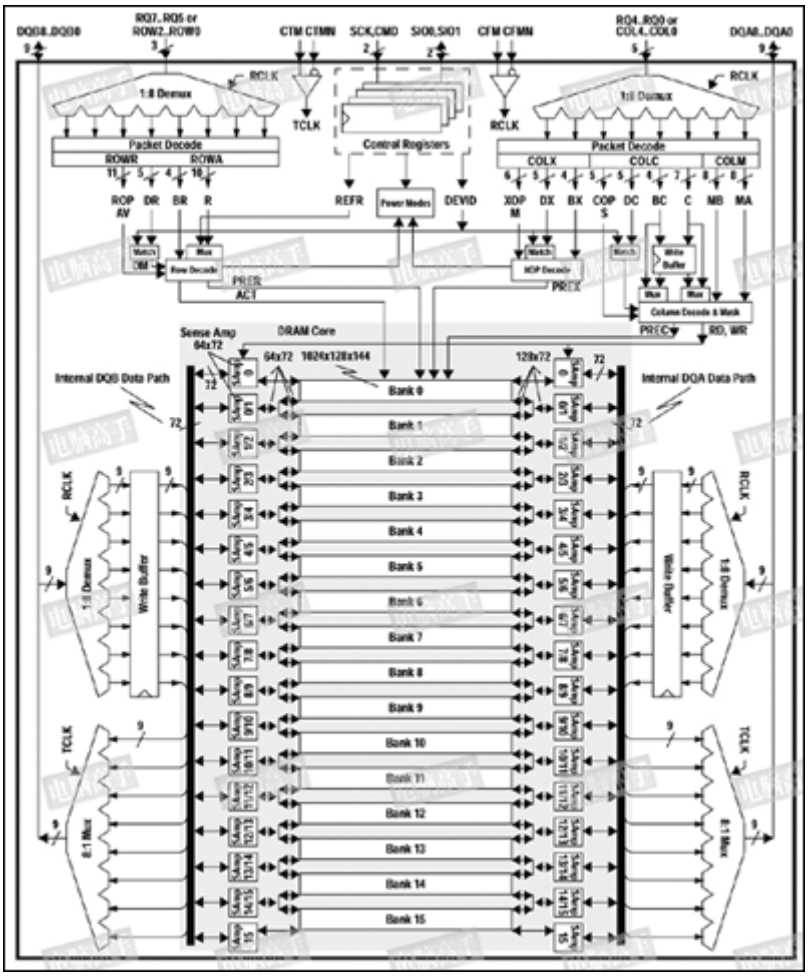
误区：RDRAM 是串行内存
串行与串联在英文中都以 Serial 表示，但串行与串联则有着根本的区别。不能看到 Serial 就说是串行或串联，而要对电路结构与传输方式进行具体分析才能得出结论。什么是串行？串行是指数据传输的方式，USB、Serial ATA、1394 就是串行传输的典型代表，它们在一个传输周期内单方向只传输 1bit 的数据，而只要超过 1bit 就不是串行而是并行了。对照 RDRAM，虽然位宽大为缩减，但仍有 16bit，所以肯定不是串行内存。 什么是串联？串联是指元件在电路中的联接方式，如并联个电阻或串联个电容等等，而谁会把它们说成并行个电阻或串行个电容呢？显然，串行与串联的概念不能混淆。 RDRAM 只能说是一种窄位宽串联结构的内存。

二、RDRAM的结构简介

1、 RDRAM 的 L-Bank 结构

RDRAM 的内部仍主要由 L-Bank 构成，但它的设计与 SDRAM 家族有很大的不同。首先，每个 L-Bank 有两个数据通道 A 和 B,各为 8bit 位宽(ECC 型号为 9bit,这种设计就是 Direct DRAM 较以前 RDRAM 的不同），每个端口都配有 S-AMP。根据 L-Bank 数量与 S-AMP 的分配方式不同，目前 RDRAM 共有三种内核结构，分别是 32s、16d 与 4i。

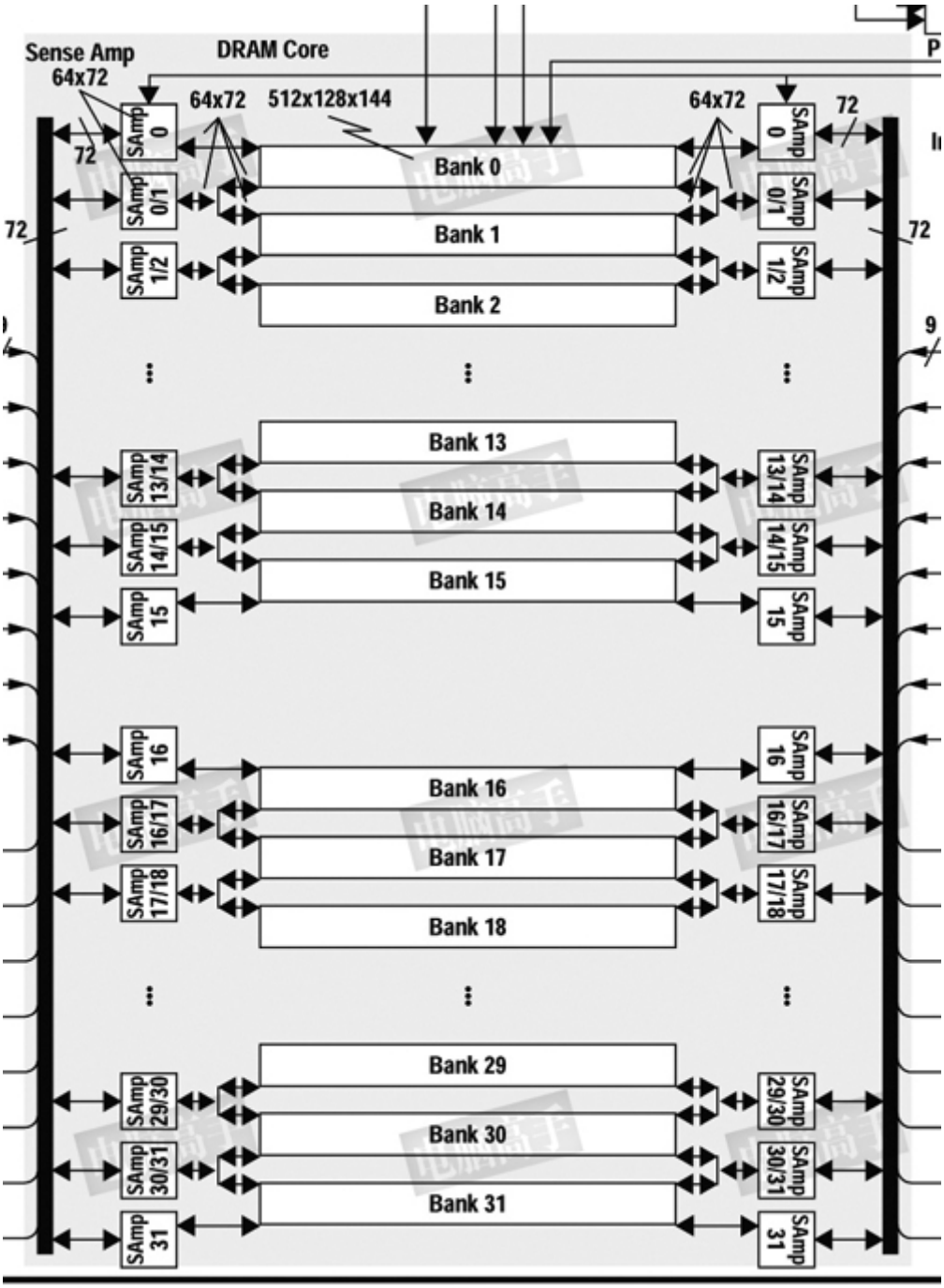
较早时，RDRAM 的设计是 16d，所谓的 d 是指 Double（双），即除了 0 与 15 号 L-Bank，其余相邻的 L-Bank 每个数据通道（A 和 B）共用一个 S-AMP。



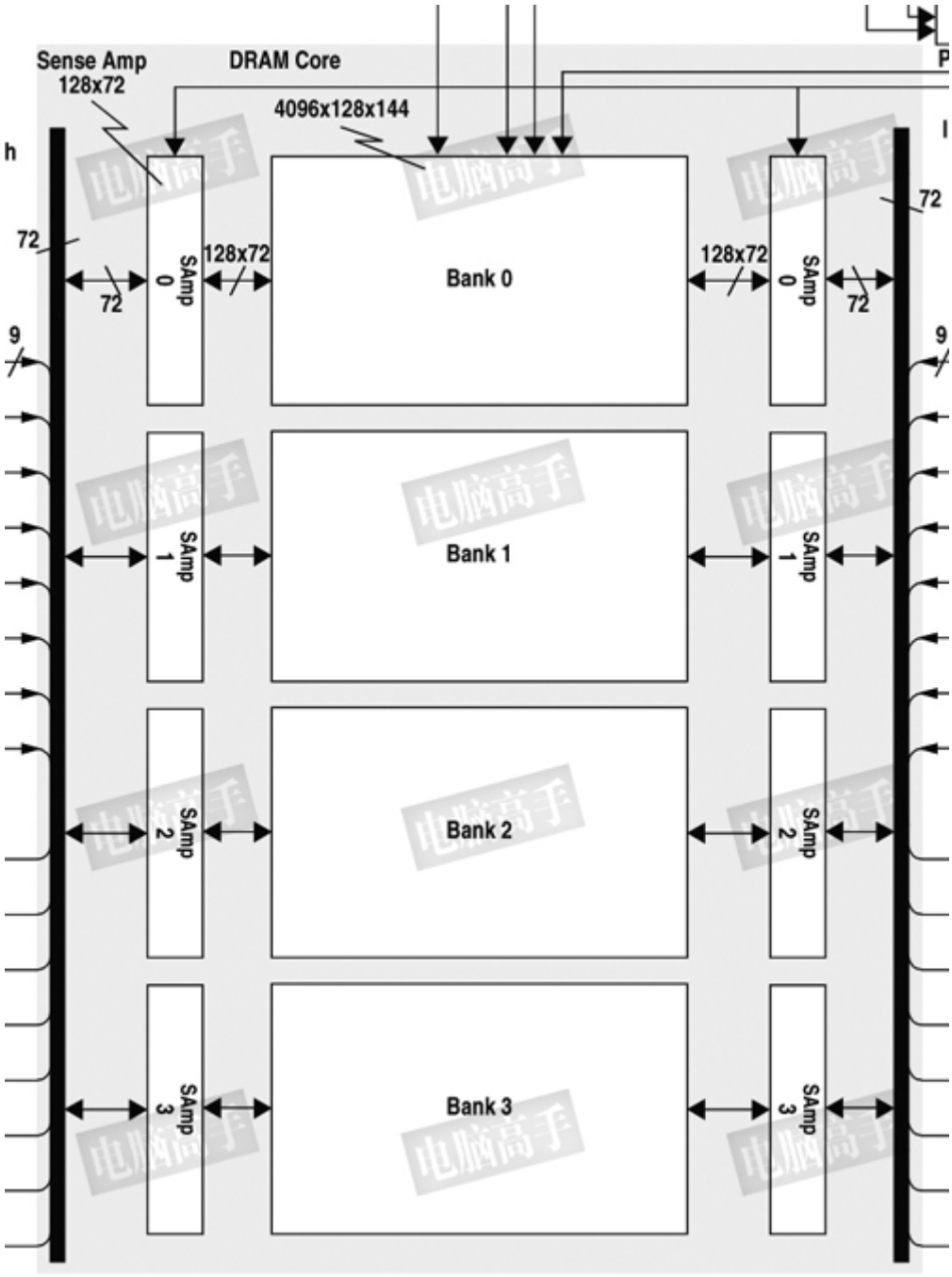
（上图可点击放大）

16d L-Bank 结构

后来分别向高端和低端领域发展了 32s 与 4i 技术。前者的 s 代表 Split，它将原来的 16d 内核分割为两个部分，各为 16d 结构，0、15、16、31 号 L-Bank 的每个数据通道各自独占一个 S-AMP。4i 则与传统的 SDRAM 相似，i 代表 Independent（独立），只有 4 个 L-Bank，各 L-Bank 的每个数据通道有单独的 S-AMP。


Sense Amp
64x72
DRAM Core
64x72
512x128x144
64x72
72
72
72
9
Bank 0
Bank 1
Bank 2
Bank 13
Bank 14
Bank 15
Bank 16
Bank 17
Bank 18
Bank 29
Bank 30
Bank 31
SAmp 0
SAmp 0/1
SAmp 1/2
SAmp 13/14
SAmp 14/15
SAmp 15
SAmp 16
SAmp 16/17
SAmp 17/18
SAmp 29/30
SAmp 30/31
SAmp 31

32s L-Bank 结构



4i L-Bank 结构

上文已经讲过，L-Bank 数越多，造成 L-Bank 寻址冲突的机率就越小，但理论上 L-Bank 越多，所用的 S-AMP 也就越多， RDRAM 内存核心加工与面积控制的难度就越大，因此 32s 与 16d 都采用了共享 S-AMP 的设计。但即使这样，RDRAM 的生产成本仍被限制在较高的水平上，在早期这成为了 RDRAM 难以普及的重要原因。而 4i 就是为解决这一问题而出现的方案，成本更低，但性能也较前两者降低了。

另外，由于共享 S-AMP 的设计，除了个别独有 S-AMP 的 L-Bank，其他的 L-Bank 每次预充电操作也都是成双成对的。为此，在逻辑控制上，RDRAM 的操作要尽量避免相邻 L-Bank 前后进行，否则也会降低 RDRAM 的实际效率。

2、RDRAM 的主要特点

目前 RDRAM 主要有两个容量规格——128Mbit 和 256Mbit。L-Bank 中存储单元的容量也并不等于 RDRAM 的接口位宽，而是它的 8 倍，因此可以说 RDRAM 是一种 8bit 预取设计，这是它最主要的特点。对于 16bit 芯片，其存储单元的容量为 128bit，这些数据分别从通道 A 和 B 传输至 L-Bank，也就是说 L-Bank 两端的 S-AMP 一次各负责 72bit 数据的传输。由于预取为 8bit，所以 RDRAM 的突发长度也固定为 8，因为如果再高，对于 PC 应用将不太适合。不过需要特别注意的是，一个字节的数据不是由数据通道中的 8 条数据线进行并排传输，而是一个字节由一条数据线进行 8 次传输，这一点也与 SDRAM 不同，它意味着北桥在进行数据读/写时，必须要等 8 个周期之后才能完成，中途不能停止。也就是说，读取时目前的北桥（如 850）一次接收 128bit（16 字节）的数据，然后再转换为两个 64bit 数据分两次向 CPU 传送。

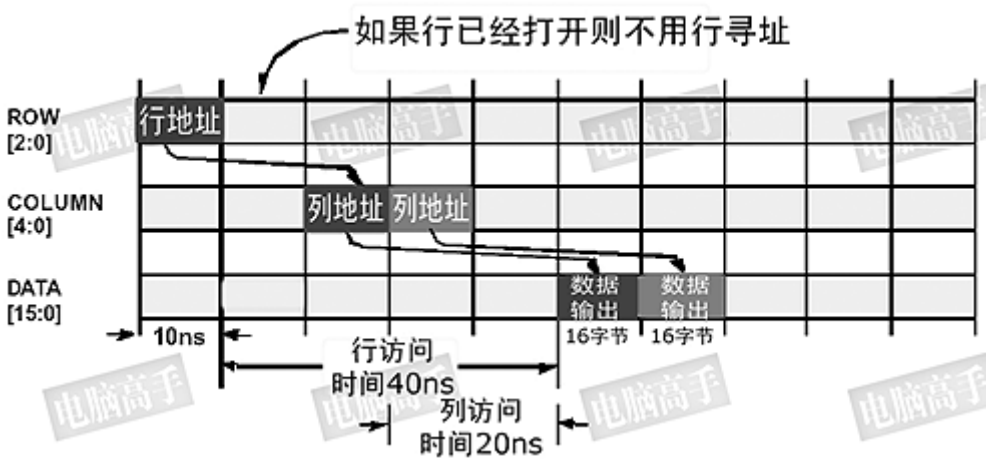
由于 RDRAM 的存储单元容量很大，所以 RDRAM 的行列地址线也大为减少，以 256Mbit 的 4i 结构的 RDRAM 为例，行地址为 12bit（4096），列地址为 7bit（128）。如果是 32s 结构的，由于 L-Bank 地址的增多，行列地址要更少（分别是 9 和 7bit）。而且 RDRAM 的行列地址线是独立的，但是 RDRAM 的行与列地址线各自只有 3 条和 5 条，显然不够用，Rambus 又是怎么搞定的呢？这就涉及到 RDRAM 具体的操作设计了。

第 13 页：昔日贵族——Rambus DRAM（二）

三、 RDRAM的具体操作与相关技术

1、 初始化与命令包

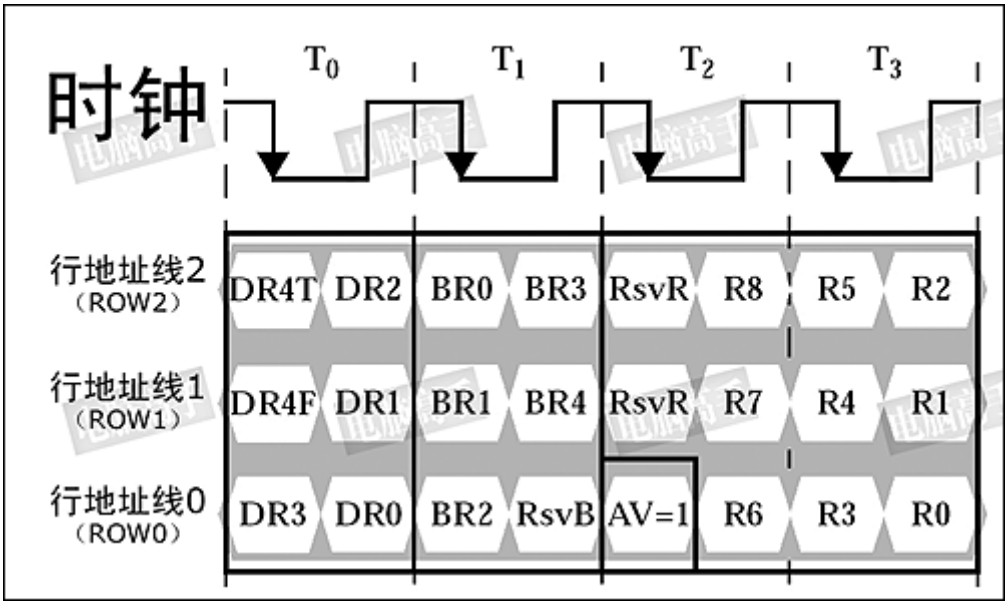
RDRAM 也有一个控制寄存器，在开机初始化过程中用来对 RDRAM 芯片进行配置，有的信息由北桥动态写入（如芯片地址、自刷新模式等），有的则是出厂时就设置好不能更改的（如刷新计数、生产商信息、支持的协议版本等）。在初始化之后，RDRAM 才能进入正常的工作状态。
RDRAM 的读写操作过程与 SDRAM 基本是一样的，也要进行片选、L-Bank 定址、行/列寻址等操作（此时的行就是指 RDRAM 内存系统中的页），但由于它的每次操作只针对一颗芯片，所以具体操作起来有很大不同，这主要体现在“命令包”的方式上。



RDRAM 读取时序图，以 PC800 为例，400MHz 时钟频率。

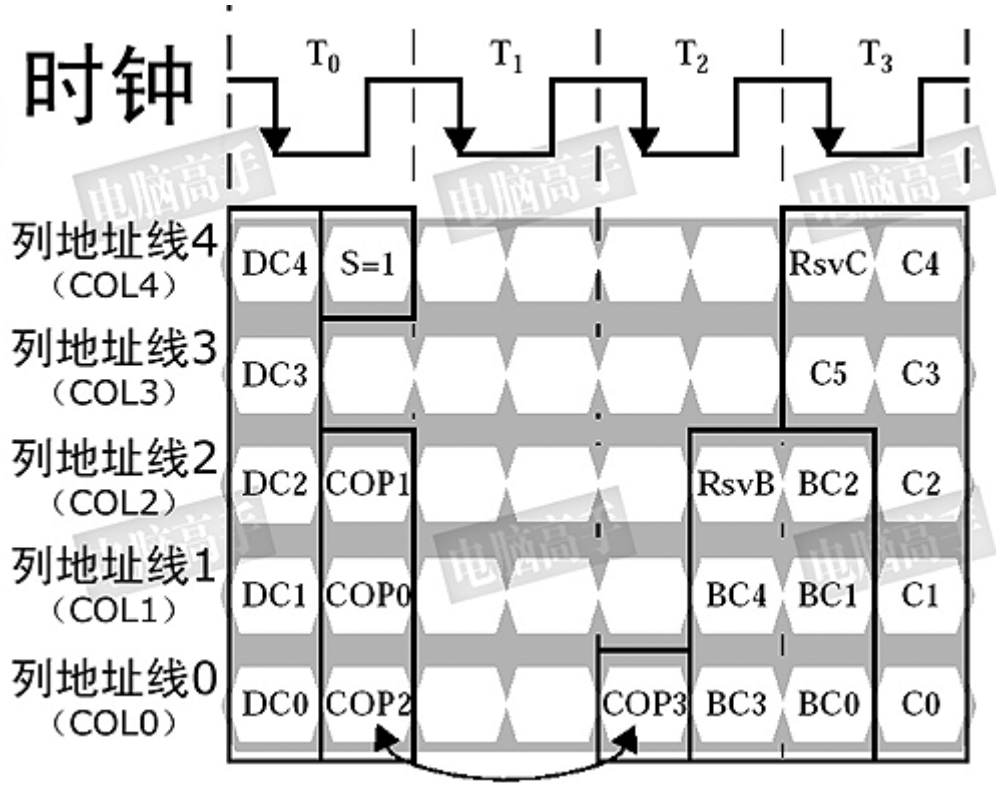
在上图中，我们可以看到行寻址命令与列寻址（读）命令并没有同时发出，而且各自占用了 10ns 的时间。我们算一算，对于 PC800，10ns 相当于 8 个传输周期。难道是传输有延迟？从行列地址的设计，就能猜到这是一个命令包形式的操作。

所谓的命令包，就是将一组命令集合在一起，统一发出。在 RDRAM 中，行命令包与列命令包都分为两种，一种是正常的读/写操作命令，一种是芯片操作命令（如数据掩码、预充电、刷新、电源管理等）。现在我们就看看行与列读/写命令包都包含哪些信息。



行读/写命令包的信息组成

DR4T、DR4F	命令包的广播类型（是对所有芯片还是单一芯片），如果是单一芯片，可用来当作最高芯片地址
DR3-DR0	芯片地址
BR4-BR0	L-Bank 地址
R0-R8	行地址
AV	区分是操作命令还是读取命令，如果是操作命令，R0-R8 将成为操作控制代码
Rsv（X）	可忽略或保留用 bit



列读/写命令包的信息组成

DC4-DC0	芯片地址
COP0-COP3	指定具体命令（是读还是写）
BC4-DC0	L-Bank 地址
C5-C0	列地址
S	允许操作控制位（1 为允许读写）
Rsv（X）	可忽略或保留用 bit

至于操作命令包就不在此多说了，因为构成的形式基本就是这样，每次用 8 个传输周期进行命令发送。而且由于 RDRAM 的命令代码很多，也比较复杂，在本专题中也不用一一列出，关键在于让大家明白 RDRAM 的寻址是怎么一回事即可，剩下的具体代码定义，如果有兴趣大家可以自行研究。

提示：RDRAM 的 32 颗芯片寻址限制
从行列命令包中，大家能发现它的芯片地址只有 5bit，最多只能寻址 32 颗芯片。这就是那个“臭名昭著”的每个 RDRAM 通道只能容纳 32 颗芯片限制的根本原因。也就是说不管通道中用了几条 RIMM，芯片的总数不能超过 32 颗，如果有 3 根 RIMM插槽，但使用了两条 16颗芯片的 RIMM，那么第三根插槽就只能插 C-RIMM。以目前的设计，RDRAM 芯片最大容量为 64MB（512Mbit-4i），这样一个通道最多只有 2GB 的容量。

2、 操作时序计算

通过上面的时序图，我们可以发现 RDRAM 计算时序的方法与 SDRAM 家族不一样，这在比较两者间时序效率时有着关键的影响。

Rambus 的时序规定与 FPE/EDO 内存时一样，在读取时延用了 tRAC、tCAC 的定义，前者是行访问周期（RAC，RAS Access Cycle/Delay），后者是列访问周期（CAC，CAS Access Cycle/Delay），你可以把它等同于 SDRAM 中的 CL，但决不能在 RDRAM 中引入 CL 这个概念。在写入时则将 tCAC 替换为 tCWD（CAS to Write Delay）。它们的单位都是时钟周期，对于 PC800，一个时钟周期就是 2.5ns，对于 PC1066 就是 1.876ns 了。显然，时钟频率越高，延迟周期就越短。

但是这些时序是从命令包发送完毕开始计算，SDRAM 则是在命令发送同时开始计算。因此，在计算 RDRAM 的操作延迟时，命令包本身占用的时间也必须要考虑进来。

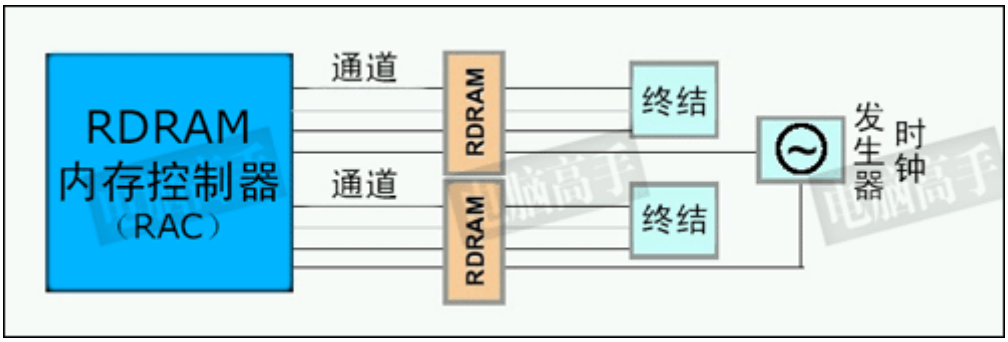
3、 写入延迟与掩码操作

RDRAM 为写入设置了专用的延迟 tCWD，这并不是被迫而是有意设计的。RDRAM 不需要 DQS 之类的信号进行同步操作，数据是可以立即接受的，但出于总线利用率的考虑，RDRAM 加入了写入延迟，它略短于 tCAC。在具体操作中，芯片上没有引脚控制写入允许/禁止，一切的命令在命令包中进行定义，所以读命令可以在写过程中发出，经过 tCAC 后有效。这样在写后读操作中，除了 tCAC 与 tCWD 之间的差距外（估计是留给写回的时间），几乎没有任何停顿，而不像 SDR/DDR SDRAM 中有较大延迟。

在写入过程中，数据都是先存在写入缓冲区中，这个操作的目的在于等待掩码的控制。RDRAM 的数据掩码只对写入有效，当收到掩码命令后，RDRAM 将指定的引脚数据从缓冲区中删除，之后再进行真正的写入。

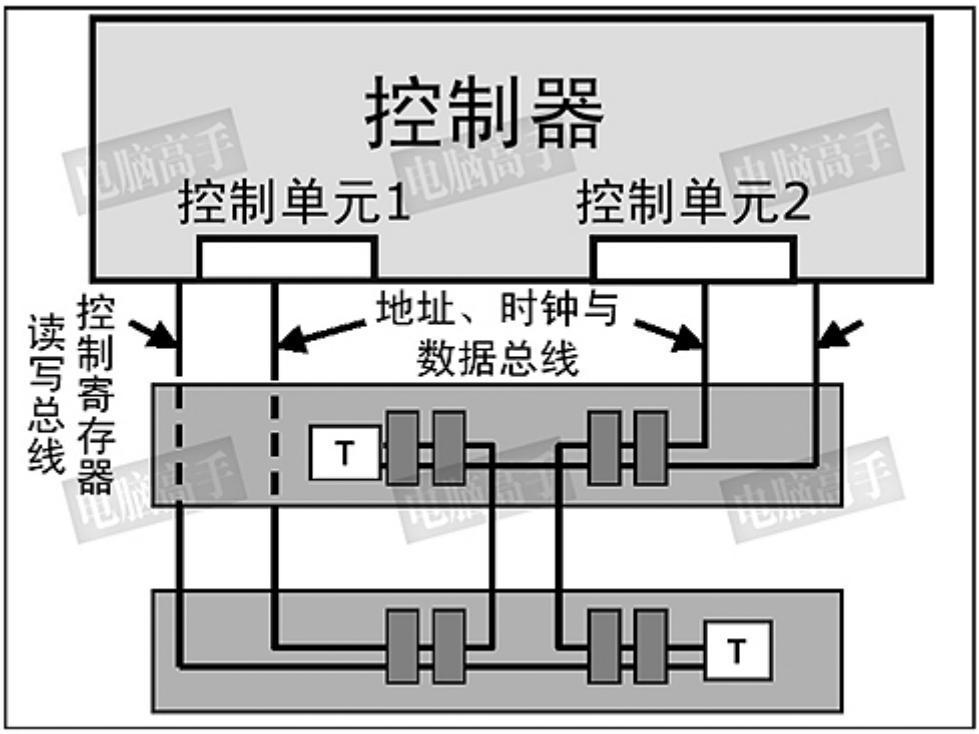
4、多通道技术与多通道模组

PC800 的速度在当时可算是 RDRAM 的一极限，但它的 1.6GB/s 带宽并不能满足高端应用的需要，而且 DDR 一方主推的产品是 P2100 的 DDR-266，为此 RDRAM 利用双通道技术来弥补带宽上的不足。简单的说，它就像一个用于内存的 RAID，两个通道的数据在 RAC 一端进行分割（写）与合并（读），两个通道的 RIMM 缺一不可并要求结构完全一致，因为寻址信号是一样的，必须适用于两个 RIMM，这也就意味着两个 RIMM 的存储轨迹也是一样。但是，数据的寻址延迟并没有变化，只是连续传输率提高了一倍达到 3.2GB/s（两个 PC800 通道），而且总的内存容量也增加了一倍。时至今日，虽然 RDRAM 使用窄位宽设计，但毕竟不是串行的方式，提升频率也越来越困难，最新的 PC1066 标准也只达到 2.1GB/s 的带宽，此时双通道设计几乎成为 RDRAM 的标配。可以说没有双通道技术的支持，RDRAM 是很难走到今天的。



RDRAM 双通道结构

以前，双通道技术是以两条 RIMM 来实现，在双通道已经是 RDRAM 标准设计的今天，这种设计的弊病很明显，比如客户的购置成本、主板的布线设计等。为此，在一些内存厂商的支持下，RDRAM 出现了多通道模组设计，其主体思路就是将每个通道的信号终结电路移植到模组上来，在一个模组上实现多通道传输。



32bit 的 RIMM 设计，每个通道的终结器做在了模组上

目前 PC 市场上 32bit RIMM 逐渐开始流行并终将取代传统的双通道设计，对于 64bit RIMM，由于是 4 通道设计，得需要 4 通道北桥芯片的支持，所以目前不可能在台式机领域里普及。

模组类型	16 bit RIMMs		32 bit RIMMs		64 bit RIMMs	
模组名称	RIMM1600	RIMM2100	RIMM3200	RIMM4200	RIMM6400	RIMM8500
RDRAM芯片数据传输频率	800 MHz	1066 MHz	800 MHz	1066 MHz	800 MHz	1066 MHz
模组数据位宽	16 or 18 bits	16 or 18 bits	32 or 36 bits	32 or 36 bits	64 or 72 bits	64 or 72 bits
模组带宽（MB/s）	1600	2133	3200	4266	6400	8532
模组上最少芯片数量	1	1	2	2	4	4
模组上最多芯片数量	16	16	16	16	16	16
地址总线数量	1	1	2	2	1	1
共享地址总线	No	No	No	No	Yes	Yes
局部数据终结	No	No	Yes	Yes	Yes	Yes
字节屏蔽支持	Yes	Yes	Yes	Yes	No	No
ECC模组位宽	18	18	36	36	72	72
模组引脚数量	168	168	232	232	326	326
数据总线阻抗	28 Ohms	28 Ohms	40 Ohms	40 Ohms	40 Ohms	40 Ohms
模组工作电压	2.5 Volts	2.5 Volts	2.5 Volts	2.5 Volts	1.8 Volts	1.8 Volts
模组终结电压	none	none	1.8 Volts	1.8 Volts	1.5 -1.8 Volts	1.5 -1.8 Volts
内同步写数据	No	No	No	No	Yes	Yes

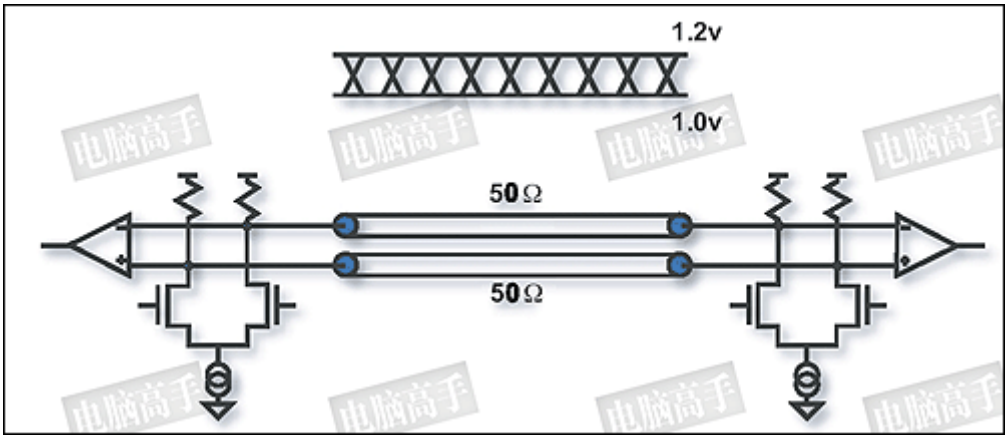
（上图可点击放大）

不同规格的 RIMM 间比较

误区：32bit RIMM 需要特殊的双通道芯片组支持
在 Intel 850E 主板上市时，很多人都发现有的主板使用了 32bit RIMM 设计，并由此断定 Intel 850E 新增了重要的功能，即支持 32bit RIMM。这是一个很常见的错误认识。32bit RIMM 相对于传统的双通道主板，只涉及物理联接形式的改变，而不涉及逻辑控制方面的变动，所以只要是支持双通道的芯片组，都可以由主板厂商通过专门的布线设计来支持 32bit RIMM，因此不要过分迷信 32bit RIMM 的技术含量，850 芯片组在理论上完全可以设计出支持 32bit RIMM 的主板，就看主板厂商愿不愿意了。

5、 黄石技术

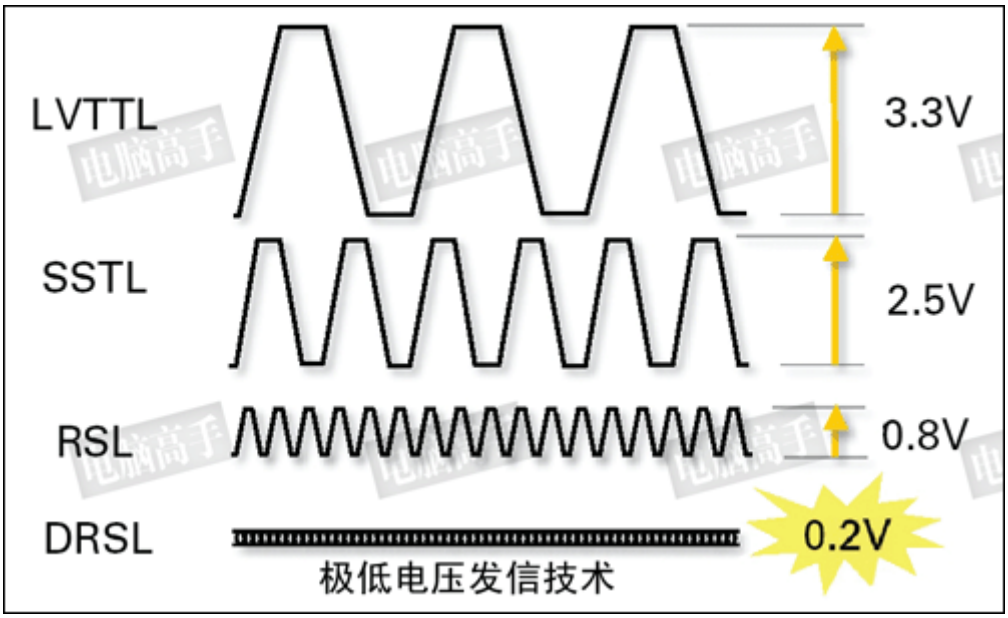
黄石（Yellowstone）是 Rambus 为了适应未来带宽的需要而开发的信号与数据传输技术，其主要的技术特点有四个：



黄石技术的物理系统结构

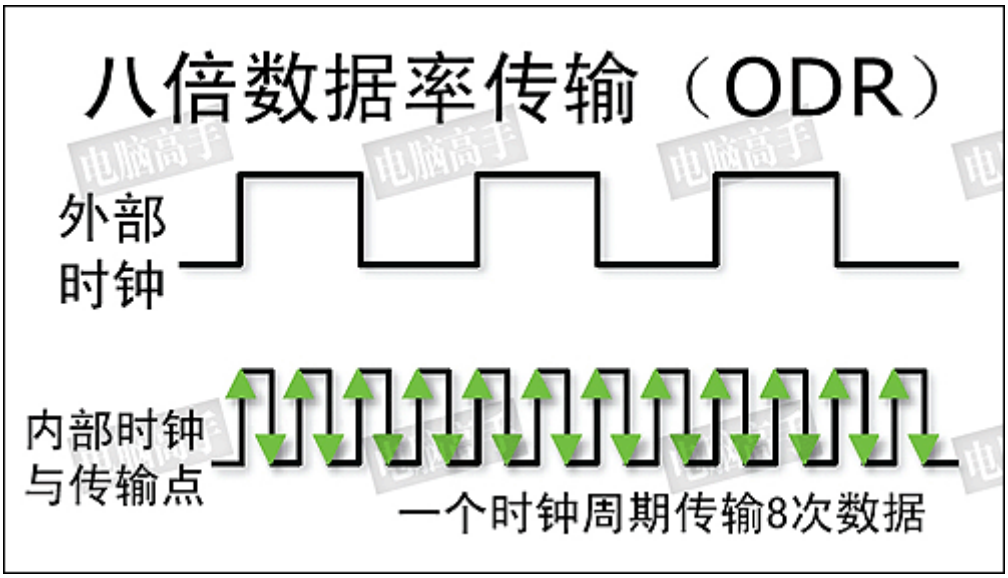
A、3.2GHz 传输频率，未来可高达 6.4GHz，按 16bit 位宽计算，带宽可达 6.4GB/s，双通道应用则为 12.8GB/s。

B、极低电压的差分 RSL 信号（DRSL），降低电源消耗并保证信号质量与制造成本。信号电压差值只有 200mV，是目前电压差最小的内存信号技术。



DRSL 发信技术与其他内存接口发信技术的比较

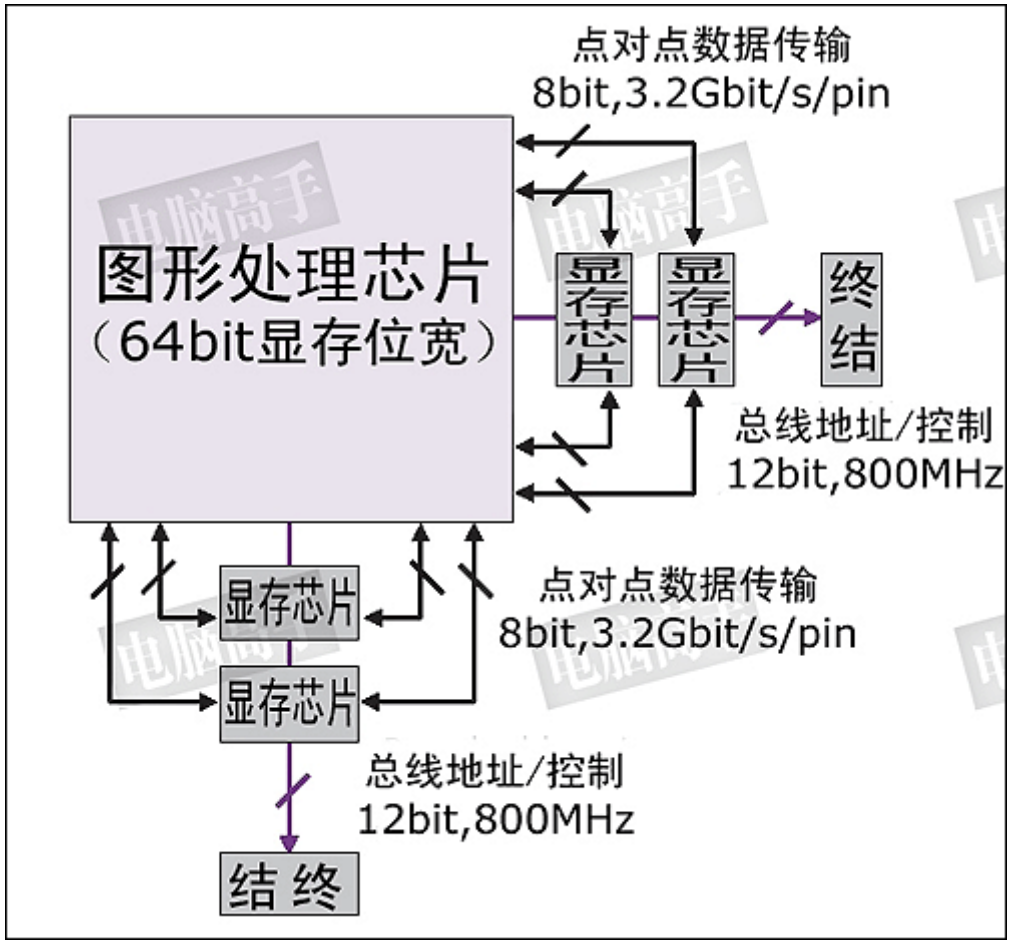
C、八倍数据流技术（ODR，Octal Data Rate）。目前采用黄石技术的 RDRAM，时钟频率仍是 400MHz，但芯片内部通过专用的锁相回路（PLL）将其转换为 1.6GHz 的内部时钟，然后在此基础上使用 DDR 技术，从而能在一个时钟周期内传输 8 次数据。数据传输频率也因此达到了 3.2GHz。



ODR 操作示意图

D、固定相位技术（FlexPhase），使内存生产者不再费力的去调校 PCB 的设计以减少延迟/潜伏期对数据/时钟间同步的影响。固定相位技术使信号本身就具备了数据/时钟同步与自校准能力，从而使外围有关时序跟踪的设计与布线变得非常简单，并有助于提高同步性，提高总线利用率。

黄石技术于 2001 年 10 月 2 日正式发布，但是虽然它有这样那样的优点，但从最近的资料中显示，Rambus 主要将其定位于消费电子、网络、通信和图形设备市场。对于目前的桌面 PC 市场，黄石在近期应用的可能性并不大。



Rambus 展示的用于显卡的点对点黄石 RDRAM 显存方案

第 15 页：昔日贵族——Rambus DRAM（四）

四、 目前的RDRAM与DDR SDRAM的比较

1、 延迟与总线利用率的比较

仍以 PC800 为例，由于 tRAC 已经包括了从行选通至数据输出的所有延迟，与是否双通道无关，所以我们只需将它再加上命令包的占用时间即可算出 RDRAM 一次访问所需要的时间。PC800 的 tRAC 基本都是 40ns（16 个时钟周期），加上命令包占用的 4 个时钟周期 10ns，总共耗时为 50ns。而这是在行关闭的情况下，没有计算预充电的时间 tRP，它一般为 12 个时钟周期（加上命令包时间），即 30ns，共计 80ns。显然，时钟频率越高，延迟就会越短。下面就来比较一下读取操作时 RDRAM 与 DDR SDRAM 的延迟。

读取操作时 RDRAM 与 DDR SDRAM 的延迟比较表

内存类型 / 时序比较	RDRAM		DDR SDRAM		
	PC800	PC1066	266B（CL=2.5）	333（CL=2.5）	400（CL=3）
时钟周期	2.5ns	1.876ns	7.5ns	6ns	5ns
需要预充电时的延迟	80ns	54.512ns	63.75ns	51ns	45ns
页面已关闭时的延迟	50ns	39.504ns	41.25ns	36ns	30ns
页面已打开时的延迟	30ns	22.512ns	18.75ns	15ns	15ns

注：DDR SDRAM 的 tRP 与 tRCD 都按 3 计算，RDRAM 方面，tCAC=8、tRP=8，PC800 的 tRAC 按 40ns 计算，PC1066 按 32ns 计算

（上图可点击放大）

从对比表中可以看出，RDRAM 相对于 DDR SDRAM 在首次寻址时的确存在较大的延迟，即使是最新的 PC1066，在与 DDR-333 的比较中也不占优势。不过，借助于双通道的设计，RDRAM 在高数据量传输应用中的优势还是比较明显的。另外，在总线的利用率方面 RDRAM 的设计也居领先地位，这为保证它的总体效率提供了坚实的保障。

各内存的总线效率比较

内存类型	PC100	PC133	DDR266	RDRAM PC800
时钟频率（MHz）	100	133	133	400
数据传输率（MHz）	100	133	266	800
系统数据总线位宽	64-bit	64-bit	64-bit	16-bit
峰值带宽（MB/s）	800	1067	2133	1600
总线利用率	62%	59%	42%	74%
最大实际带宽（MB/s）	494	631	897	1190

注：SDRAM 与 DDR SDRAM 的 L-Bank 数按 4 个计算，RDRAM 按 16 个计算

这个对比表是东芝公司经过反复实验而得出的结论，它是通过一些典型的操作（如写-读-读），结合不同页命中情况下的时序，以及刷新对内存操作的影响等分析而得出的。由于 DDR SDRAM 在 L-Bank 数量上占劣势，所以出现 L-Bank 寻址冲突的可能性要大为提高，而且在写后读操作中，RDRAM 的延迟也明显小于 SDRAM 家族，因此虽然 PC800 的峰值带宽不如 DDR-266，但综合效率要更好。这可以解释为什么在一些测试中，RDRAM 明显比 DDR 领先的原因。不过，在以零散数据为主的操作中，RDRAM 的固定传输周期以及高延迟就成为了性能的障碍。

从前面的分析可以看出，SDRAM/DDR 在数据控制上的灵活性要比 RDRAM 高，首次访问的延迟也更短，因此在某些操作中，即使带宽比 RDRAM 系统小，性能仍不见得落后。比如 845D/E 在某些应用测试中，完全可以与双通道 PC1066 一较高低。而 Intel 决定在高端服务器领域使用 DDR 芯片组，也基本是出于这个考虑，因为在服务器的操作中，零散型存取操作所占比例很大。相反，若大规模连续存取操作占比例很大（如视频与音频工作站），那么可能就要考虑 RDRAM 了。

2、 未来竞争展望

目前随着多通道技术在 DDR 上的普及，RDRAM 在带宽上的优势也变得不明显了。所以，RDRAM 如果不及时提高单通道的性能，很快会被强大的 DDR 家族赶出台式机领域。但 RDRAM 的时钟频率已经很高了，再向上提高已经很难，不少 RDRAM 厂商都表示，800MHz 时钟频率可能将是 RDRAM 的一个巨大门槛，即使能超过，成本可能也是惊人的，要知道目前 533/400MHz 的 RIMM 就已使用了 8 层 PCB，800MHz 时 PCB 成本将很难控制。这也是为什么 RDRAM 急于推出 32bit 与 64bit RIMM 的原因，毕竟内存这种高带宽应用设备，还是需要一定位宽的保证。而且高位宽的同步性也不像想象中的那么难以控制，DQS 的设计就很大程度地解决了这一问题，所以，DDR 可以借助较少的转产成本，较低的 PCB 成本（即使是 DDR-Ⅱ也是 6 层设计），成为 PC 内存的首选产品。

现在再去争论 RDRAM 与 DDR 谁胜谁败已经没有意义，RDRAM 已经很难再在主流市场重振雄风。这主要不是它的技术限制，而是早期的市场动作与成本的压力造成的。虽然现在 4i 芯片开始起步，但支持这种结构的芯片组还很难找到（至少 850E 不支持）。在 820 时代，RDRAM 由于成本而没有打开市场，现在可以通过降低成本来提高竞争力，但 DDR 一方也有了多通道技术。Rambus 也因此明智地将黄石定位于专用/定制市场。这样，在今后很长一段时间里我们只有看 DDR 的独角戏了。

第 16 页：明日之星——DDR-Ⅱ与 DDR-Ⅲ（一）

作为 DDR 的接班人，DDR-Ⅱ在规范制定之初就引起了广泛的关注，进入 2002 年，三星、Elpida、Hynix、Micron 等都相继发布了 DDR-Ⅱ芯片（最早由三星在 5 月 28 日发布），让人觉得 DDR-Ⅱ突然和我们近了。可是，DDR-Ⅱ规范却一直没有正式公开，在 JEDEC 上仍只有一篇 ATi 技术人员写的，在目前看来有些内容都已过时的简要介绍。

原来，DDR-Ⅱ标准到 2002 年 10 月完成度也没有达到 100%（厂商透露大约为 95%），而上述厂商所推出的芯片也在不断的修改中，预计正式的规范将在明年第一季度推出。不过，DDR-Ⅱ的主体设计已经完成，不会有大的改动，所以通过这些“试验性”芯片，我们仍可掌握 DDR-Ⅱ的主要信息。

DDR-Ⅱ相对于 DDR 的主要改进如下：

DDR-Ⅱ与目前的 DDR 对比表

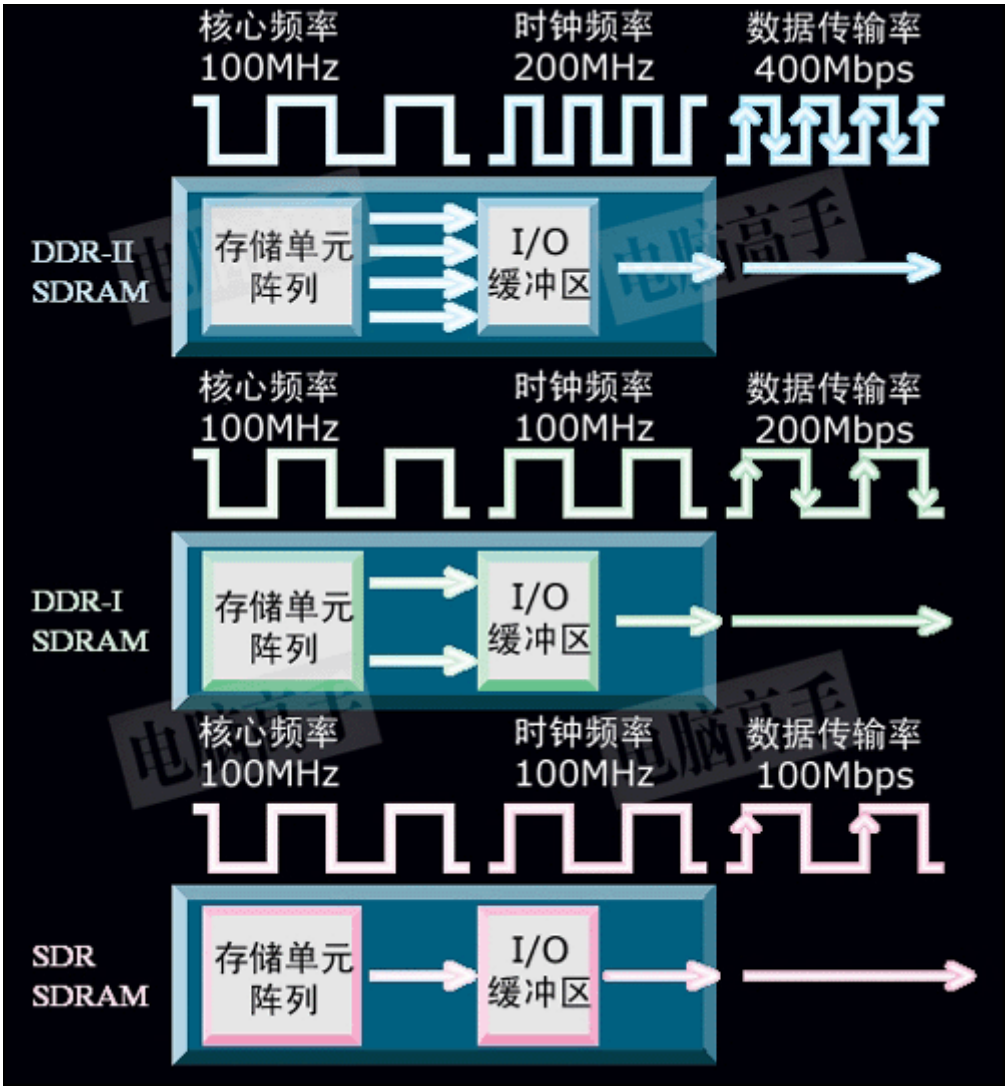
项目		DDR-II SDRAM	DDR SDRAM
基本设计	时钟频率（MHz）	200/266/333	100/133/166/200
	数据传输率（Mbps）	400/533/667	200/266/333/400
	预取设计	4bit	2bit
	突发长度	4/8	2/4/8
	L-Bank 数量	最多 8 个	最多 4 个
	CL 值	3、4、5	1.5、2、2.5、3
	数据选取脉冲	差分数据选取脉冲	单数据选取脉冲
电气性能与封装	工作电压	1.8V	2.5V
	接口标准	SSTL_18	SSTL_2
	消耗功率	304mW（最大）533Mbps	418mW（最大）266Mbps
	封装	CSP（FBGA）无铅封装，60/64/68/84/92pin	TSOP-II（66Pin）、CSP（60pin）
	模组标准	240pin DIMM	184pin DIMM
	系统最高 P-Bank 数量	4	8
功能	命令集	同 DDR	-
	基本时序定义	同 DDR	-
	新功能	ODT、OCD 调校、Posted CAS、AL	无

由于 DDR-Ⅱ相对 DDR-I 的设计变动并不大，因此很多操作就不在此详细介绍了，本文重点阐述 DDR-Ⅱ的一些重要变化。

一、 DDR-Ⅱ内存结构

DDR-Ⅱ内存的预取设计是 4bit，通过 DDR 的讲述，大家现在应该知道是什么意思了吧。

上文已经说过，SDRAM 有两个时钟，一个是内部时钟，一个是外部时钟。在 SDRAM 与 DDR 时代，这两个时钟频率是相同的，但在 DDR-Ⅱ内存中，内部时钟变成了外部时钟的一半。以 DDR-Ⅱ 400 为例，数据传输频率为 400MHz（对于每个数据引脚，则是 400Mbps/pin），外部时钟频率为 200MHz，内部时钟频率为 100MHz。因为内部一次传输的数据就可供外部接口传输 4 次，虽然以 DDR 方式传输，但数据传输频率的基准——外部时钟频率仍要是内部时钟的两倍才行。就如 RDRAM PC800 一样，其内部时钟频率也为 100MHz，是传输频率的 1/8。



DDR-Ⅱ、DDR 与 SDRAM 的操作时钟比较

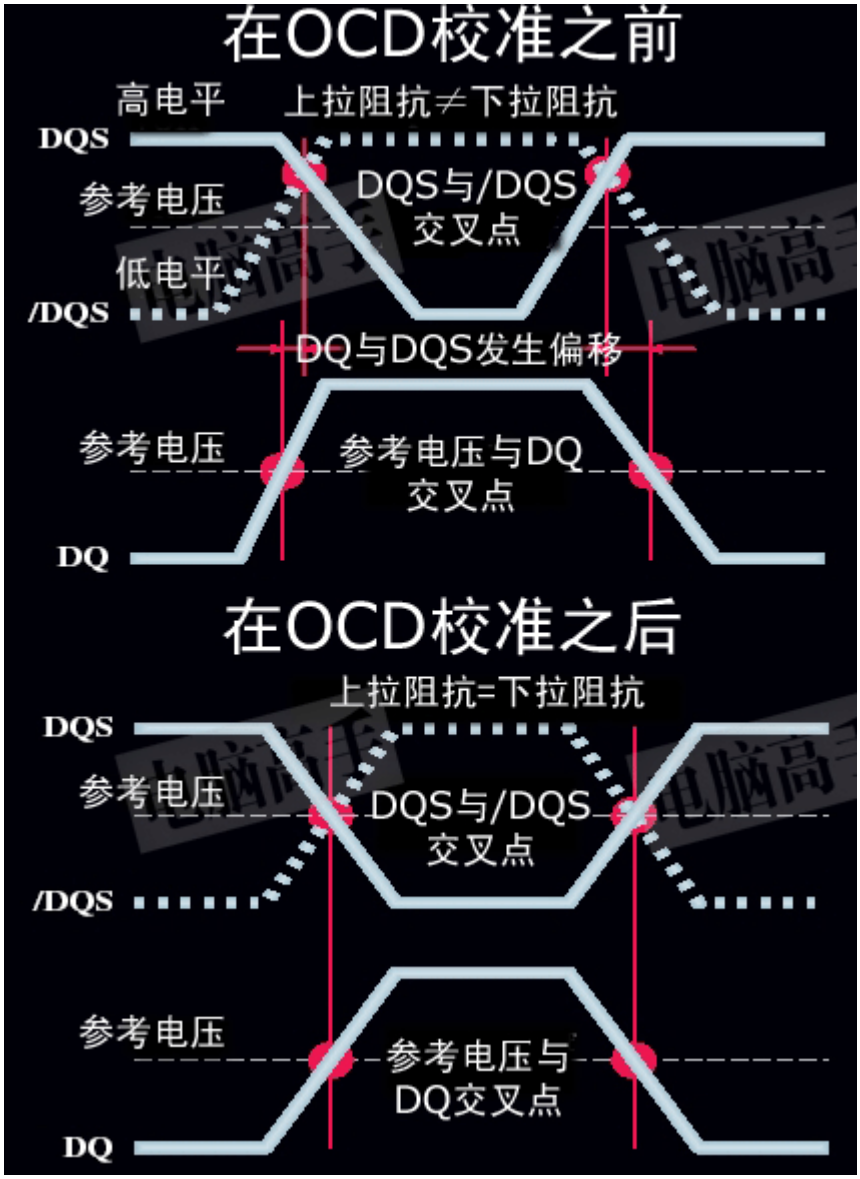
所以，当预取容量超过接口一次 DDR 的传输量时，内部时钟必须降低（除非数据传输不是 DDR 方式，而是一个时钟周期 4 次）。如果内部时钟也达到 200MHz，那外部时钟也要达到 400MHz，这会使成本有大幅度提高。因此，DDR-Ⅱ虽然实现了 4-bit 预取，但在实际效能上，与 DDR 是一样的。在上面那幅比较图中，可以看出厂商们的一种误导，它虽然表示出在相同的核心频率下，DDR-Ⅱ达到了两倍于 DDR 的的带宽，但前提是 DDR-Ⅱ的外部时钟频率也是 DDR 和 SDRAM 的两倍。在 DDR 的时钟频率已经达到 166/200MHz 的今天，再用 100MHz 去比较，显然意义不大。这点也请大家们注意识别，上图更多的是说明 DDR-Ⅱ内外时钟的差异。毕竟内部时钟由外部决定，所以外部时钟才是比较的根本基准。

总之，现在大家要明确认识，在外部时钟频率相同的情况下，DDR-Ⅱ与 DDR 的带宽一样。

二、 DDR-Ⅱ的新操作与新时序设计

1、片外驱动调校（OCD，Off-Chip Driver）

DDR-Ⅱ内存在开机时也会有初始化过程，同时在 EMRS 中加入了新设置选项，由于大同小异，在此就不多说了。在 EMRS 阶段，DDR-Ⅱ加入了可选的 OCD 功能。OCD 的主要用意在于调整 I/O 接口端的电压，来补偿上拉与下拉电阻值。目的是让 DQS 与 DQ 数据信号之间的偏差降低到最小。调校期间，分别测试 DQS 高电平/DQ 高电平，与 DQS 低电平/DQ 高电平时的同步情况，如果不满足要求，则通过设定突发长度的地址线来传送上拉/下拉电阻等级（加一档或减一档），直到测试合格才退出 OCD 操作。



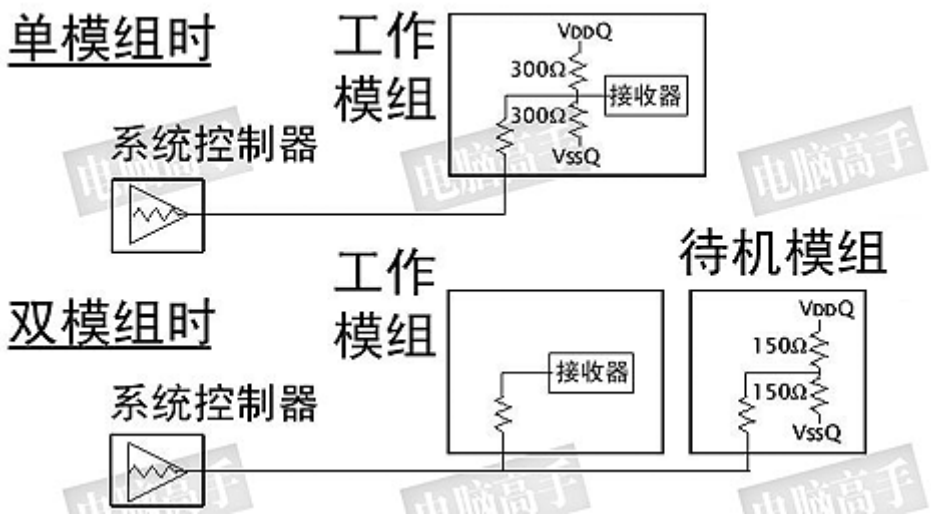
OCD 的作用在于调整 DQS 与 DQ 之间的同步，以确保信号的完整与可靠性

不过，据一些厂商的技术人员介绍，一般情况下有 DQS#（差分 DQS 时）就基本可以保证同步的准确性，而且 OCD 的调整对其他操作也有一定影响，因此在普通台式机上不需要用 OCD 功能，它一般只会出现在高端产品中，如对数据完整性非常敏感的服务器等。

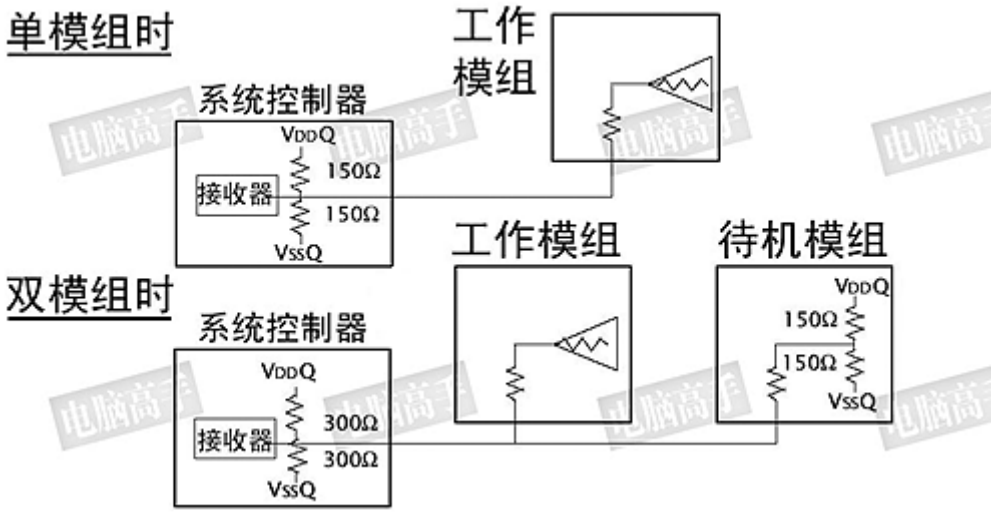
2、片内终结（ODT，On-Die Termination）

所谓的终结，就是让信号被电路的终端被吸收掉，而不会在电路上形成反射，造成对后面信号的影响。在 DDR 时代，控制与数据信号的终结在主板上完成，每块 DDR 主板在 DIMM 槽的旁边都会有一个终结电压岛的设计，它主要由一排终结电阻构成。长期以来，这个电压岛一直是 DDR 主板设计上的一个难点。而 ODT 的出现，则将这个难点消灭了。

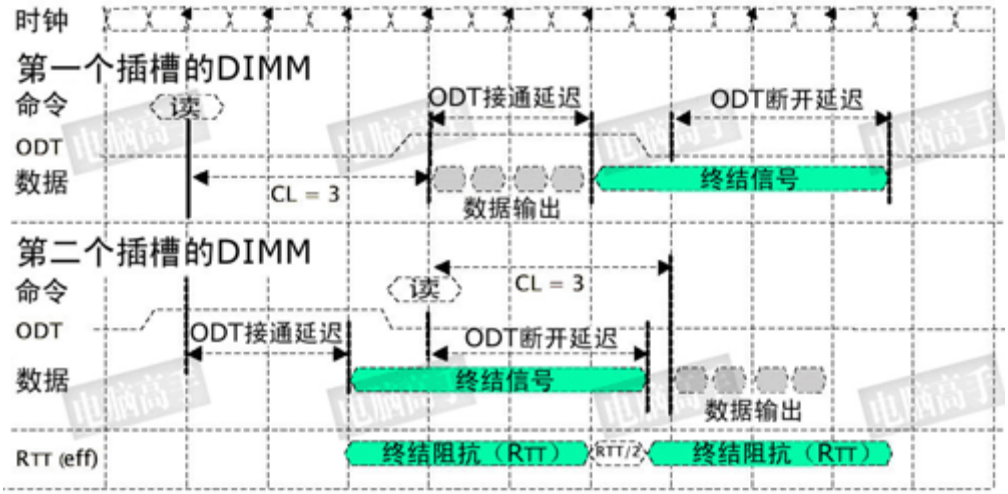
顾名思义，ODT 就是将终结电阻移植到了芯片内部，主板上不在有终结电路。ODT 的功能与禁止由北桥芯片控制，ODT 所终结的信号包括 DQS、RDQS（为 8bit 位宽芯片增设的专用 DQS 读取信号，主要用来简化一个模组中同时使用 4 与 8bit 位宽芯片时的控制设计）、DQ、DM 等。需要不需要该芯片进行终结由北桥控制。
那么具体的终结操作如果实现呢？首先要确定系统中有几条模组，并因此来决定终结的等效电阻值，有 150 和 75Ω 两档，这一切由北桥在开机进行 EMRS 时进行设置。



在向内存写入时，如果只有一条 DIMM，那么这条 DIMM 就自己进行终结，终结电阻等效为 150Ω。如果为两条 DIMM，一条工作时，另一条负责终结，但等效电阻为 75Ω



在从内存读出时，终结操作也将在北桥内进行，如果有两条 DIMM，不工作的那一条将会终结信号在另一方向的余波，等效电阻也因 DIMM 的数量而有两种设置



（上图可点击放大）

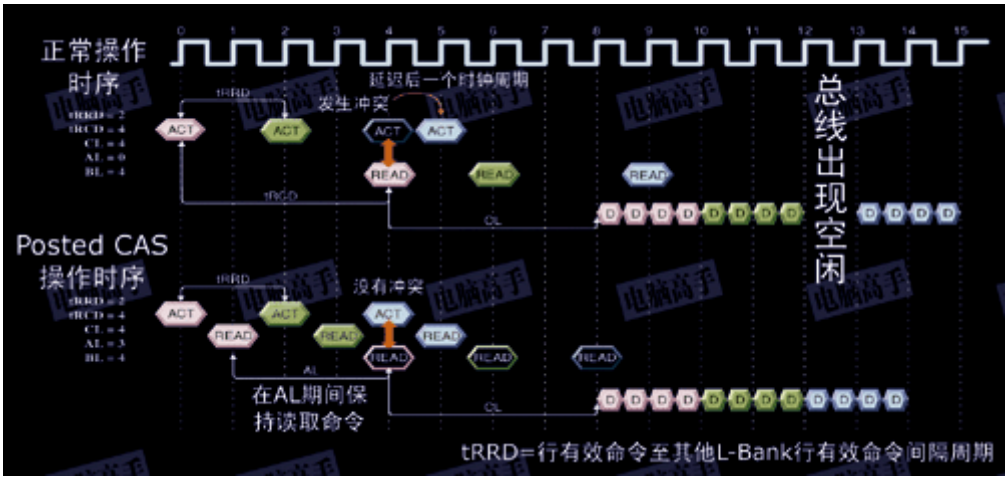
两个 DIMM 在交错工作中的 ODT 情况，第一个模组工作时，第二个模组进行终结操作，等第二个模组工作时，第一个模组进行终结操作

现在我们应该基本了解了 ODT 的功能，它在很大程度上减少了内存芯片在读取时的 I/O 功率消耗，并简化了主板的设计，降低了主板成本。而且 ODT 也要比主板终结更及时有效，从而也成为了提高信号质量的重要功能，这有助于降低日后 DDR-Ⅱ进一步提速的难度。但是，由于为了确保信号的有效终结，终结操作期将会比数据传输期稍长，从而多占用一个时钟周期的时间而造成总线空闲。不过，有些厂商的技术人员称，通过精确设置 tDQSS，可以避免出现总线空闲。

第 17 页：明日之星——DDR-Ⅱ与 DDR-Ⅲ（二）

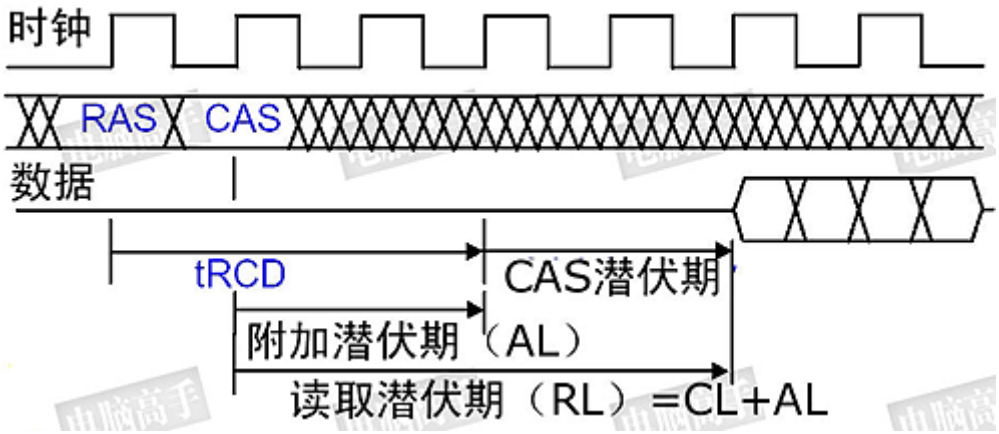
3、前置 CAS、附加潜伏期与写入潜伏期

前置 CAS（Posted CAS）是为了解决 DDR 内存中指令冲突而设计的功能。它允许 CAS 信号紧随 RAS 发送，相对于以往的 DDR 等于将 CAS 前置了。这样，地址线可以立刻空出来，便于后面的行有效命令发出，避免造成命令冲突而被迫延后的情况发生，但读/写操作并没有因此而提前，仍有要保证有足够的延迟/潜伏期，为此，DDR-Ⅱ引入了附加潜伏期的概念（AL，Additive Latency），与 CL 一样，单位为时钟周期数。AL+CL 被定义为读取潜伏期（RL，Read Latency），相应的，DDR-Ⅱ还对写入潜伏期（WL，Write Latency）制定了标准，WL 是指从写入命令发出到第一笔数据输入的潜伏期，不要将它和 tDQSS 弄混了，后者是指 DQS 而不是数据。按规定，WL=RL-1，即 AL+CL-1。



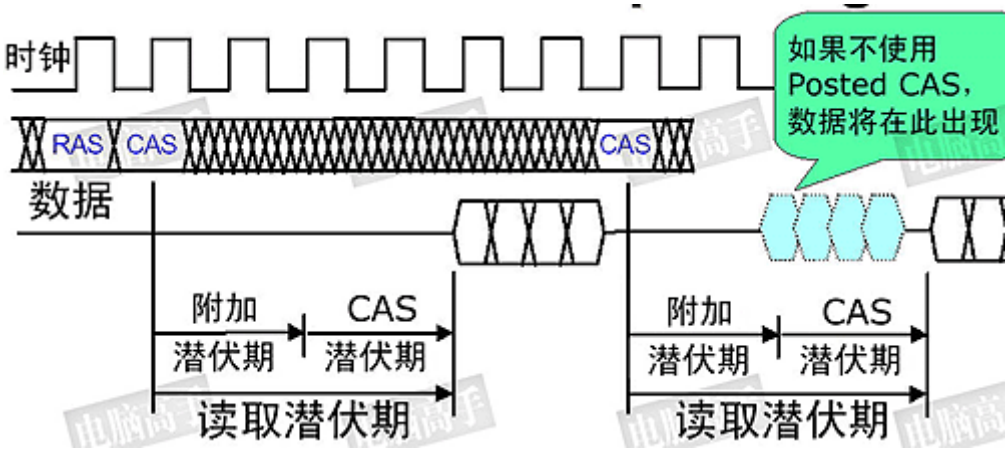
（上图可点击放大）

在没有前置 CAS 功能时，对其他 L-Bank 的寻址操作可能会因当前行的 CAS 命令占用地址线而延后，并使数据 I/O 总线出现空闲，当使用前置 CAS 后，消除了命令冲突并使数据 I/O 总线的利率提高



设置 Posted-CAS 后，必须附加潜伏期以保证应有延迟，此时读取潜伏期（RL）就等于 AL+CL，从中可以看出 AL 的值为 CL+tRCD-1

DDR-Ⅱ中 CL 最低值为 3，最高为 5，并且不再有 x.5 的设计，而 AL 值则为 0-4。当 AL 设为 0 时，前置 CAS 无效，即为传统 DDR 模式的操作。不过前置 CAS 在解决命令冲突的时间也带来了新的问题——在背靠背式读取时，仍将经过 AL+CL 的潜伏期才能读取数据，比传统的只有 CL 相比，读取的延迟反而增加了。因此，AL=0 是默认设置，只有在那些读写命令非常频繁的操作场合，才建议启动前置 CAS 功能（如服务器等），对于台式机用户，前置 CAS 的优点不足以抵消其带来的不利影响。



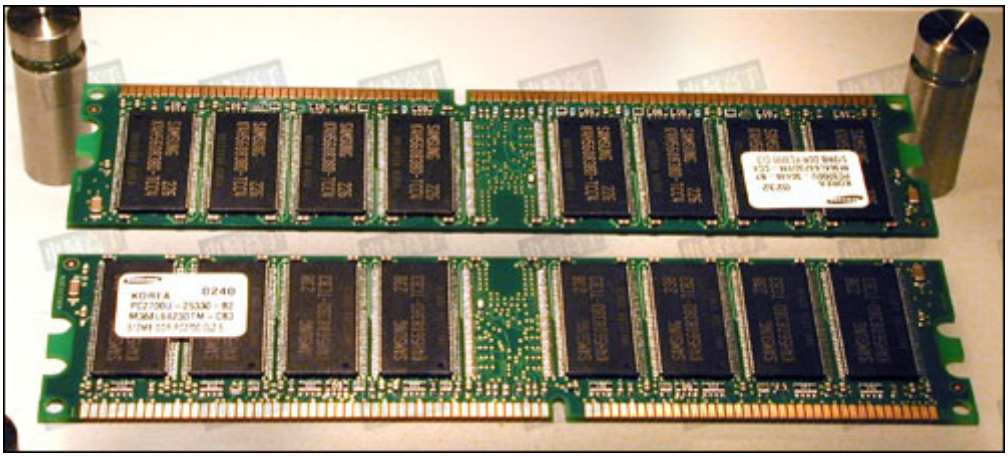
由于有了 AL，在同一行中进行再读取时，在 CL 的基础上仍将增加 AL 造成的延迟，从而影响了性能

三、 DDR-Ⅱ未来发展与DDR-Ⅲ

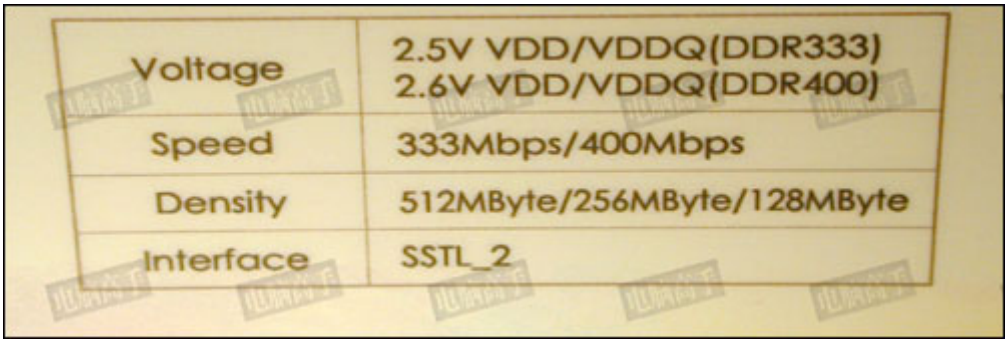
1、 DDR-Ⅱ 的发展计划

虽然目前多家厂商都推出了 DDR-Ⅱ内存芯片，但从 DDR 官方组织 JEDEC 方面得到的信息表明，距离 DDR-Ⅱ内存大规模上市还很遥远，2004 年才会是 DDR-Ⅱ普通的阶段。而由于三星、南亚与 Micron 公司的大力推广，这期间 JEDEC 很可能会接受 DDR-400 标准，目前的争执主要在于能否在 DDR-I 的体系下保证 DDR-400 的可靠性。对此（成为 JEDEC 正式标准），三星与南亚公司都表示了很强的信心。

笔者认为，DDR-400 应该会获得认可，毕竟市场上是有需要的，而让市场去等一年的时间迎接 DDR-Ⅱ 400 似乎并不现实。不过，多通道技术在 DDR 领域里的普及，可能也会改变 JEDEC 对认证 DDR-400 的想法，但关键要看多通道的性价比能不能填补这一空档，否则 DDR-400 就是一个最佳的选择（在完整/进阶版完稿之后又传来了 Intel 准备支持 DDR-400 的消息，可见 DDR-400 的前途）。



三星公司展示的 DDR-333（下）与 DDR-400（上）内存模组



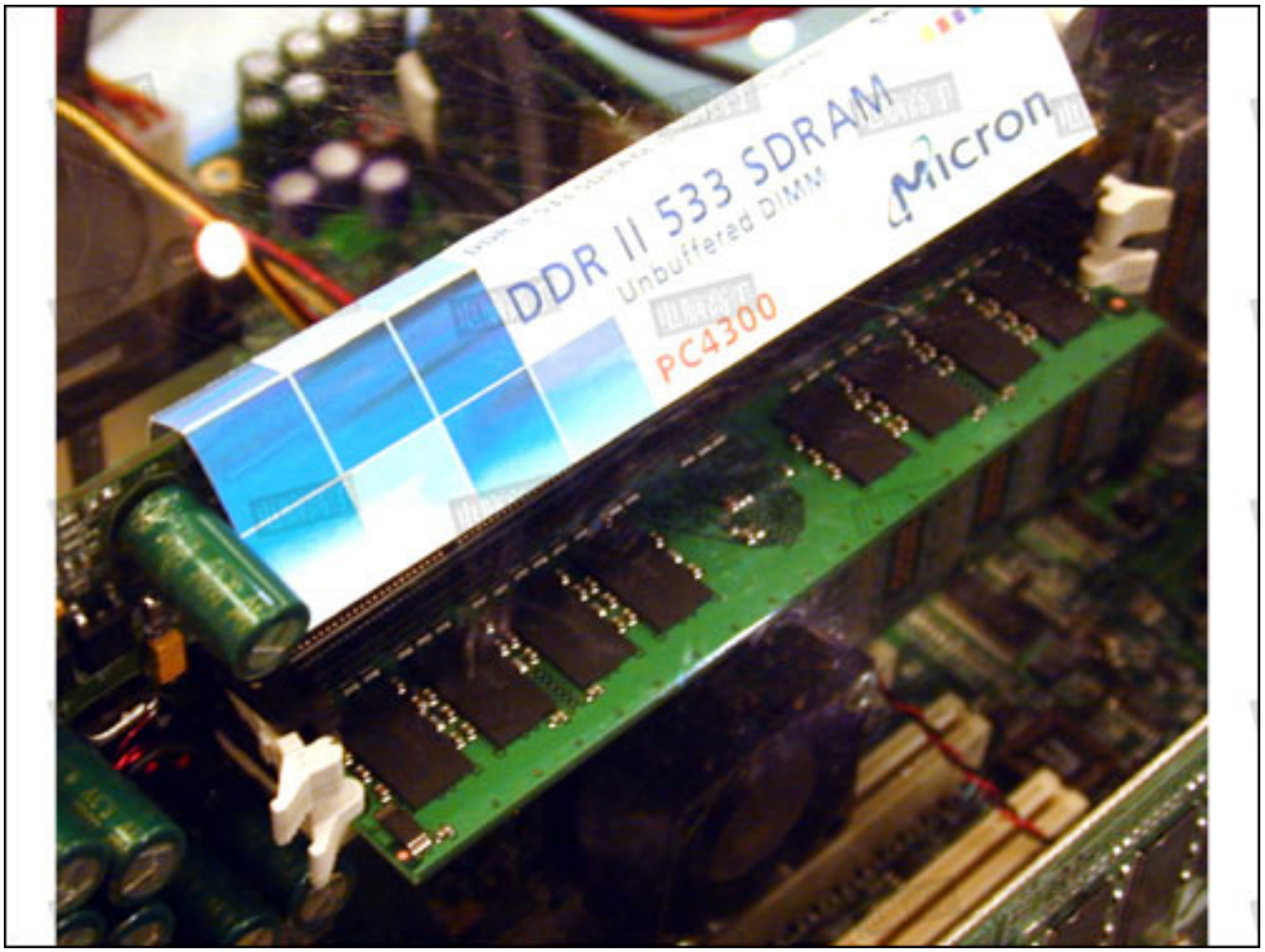
Voltage	2.5V VDD/VDDQ(DDR333) 2.6V VDD/VDDQ(DDR400)
Speed	333Mbps/400Mbps
Density	512MByte/256MByte/128MByte
Interface	SSTL_2

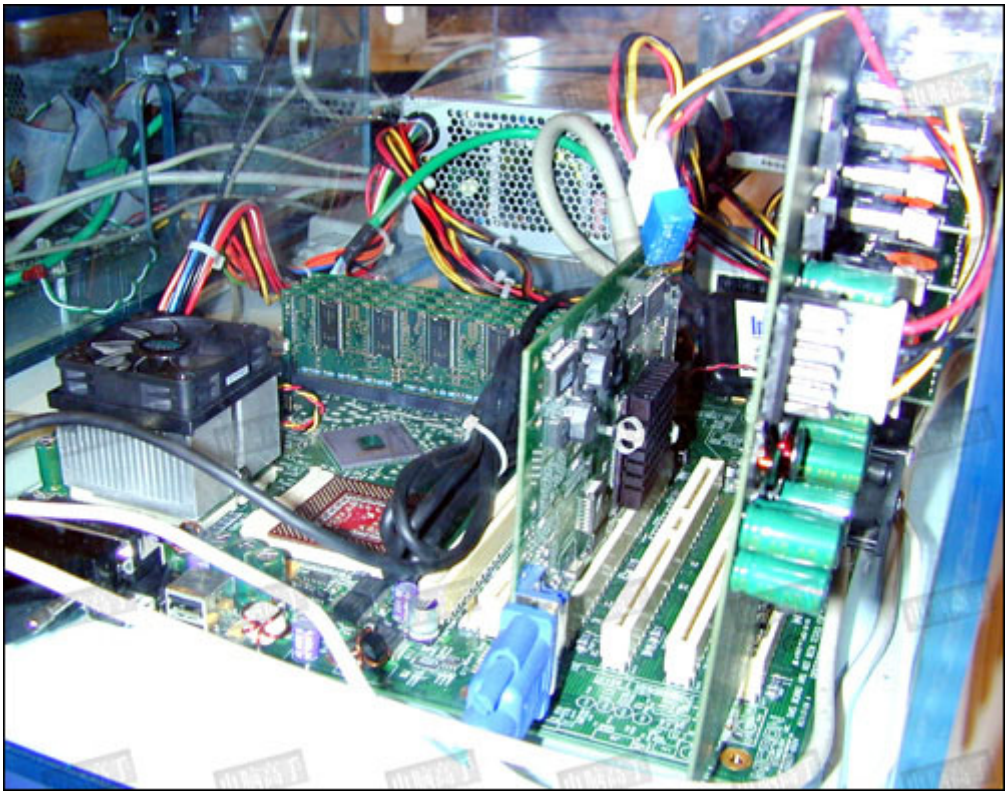
三星是 DDR-400 的主推厂商，但请注意 DDR-400 的电压变化，它可能是引起兼容性问题的根源之一

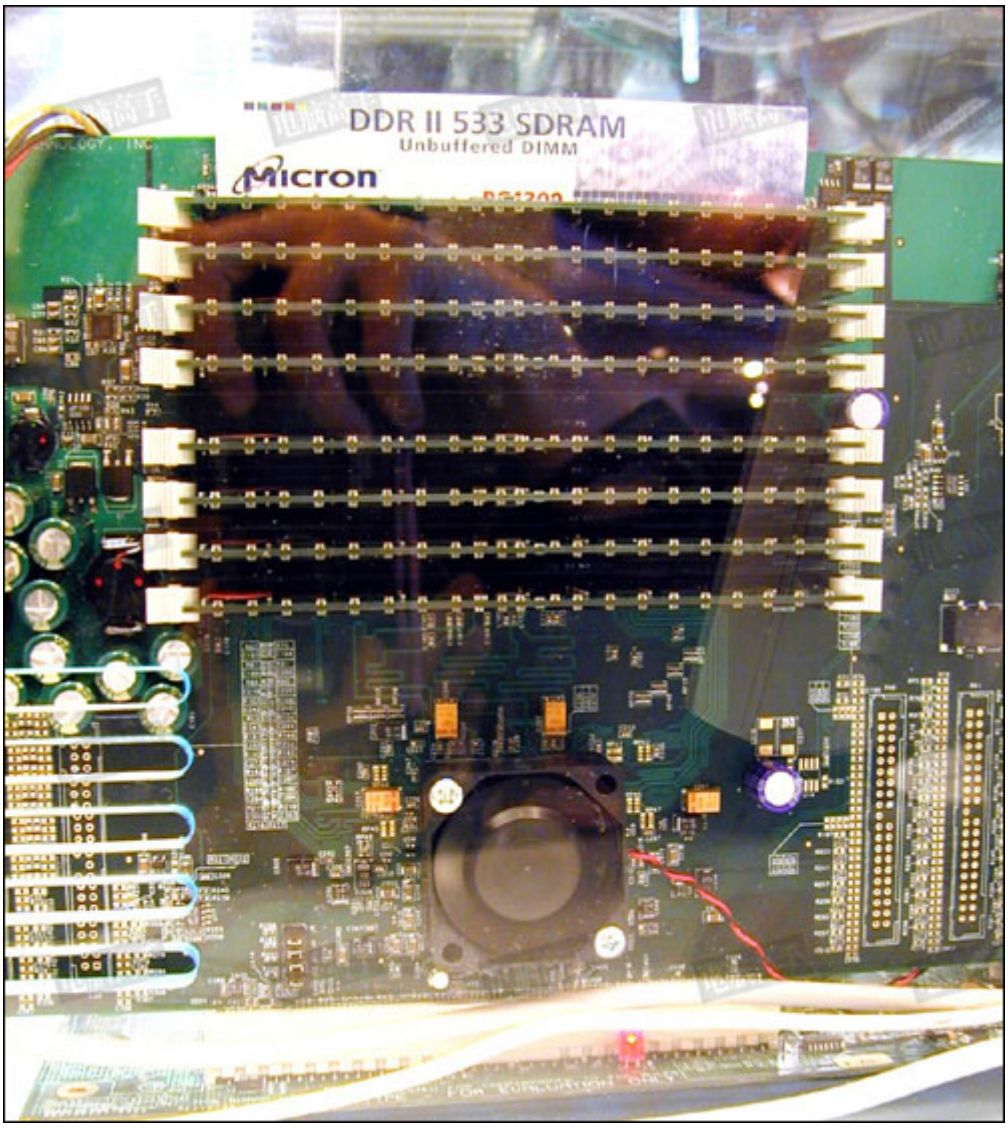
也由于多通道的出现，市场上对 DDR-Ⅱ的渴望也并不大，毕竟它与同频的 DDR-I 内存的带宽一样。而从上文可以看出，DDR-Ⅱ相对于 DDR-I 的不同设计很多都集中在了如何在更高的工作（时钟）频率下保证数据的可靠。只有当 DDR-Ⅱ依靠自身的特有功能与设计来获得更高的时钟频率时，再配合多通道，才会真正拉开与 DDR-I 的距离，那时也就是 DDR-Ⅱ普及的开始。但笔者预测 DDR-Ⅱ 400 将像 DDR-200 一样，注定是一个一出生就过时的标准，DDR-Ⅱ至少要从 533 开始流行。不过在目前情况下，我们还不必太在意 DDR-Ⅱ的进展情况，说句实话，它离我们还很远。今天的介绍只是让大家对其有一个大概的了解。

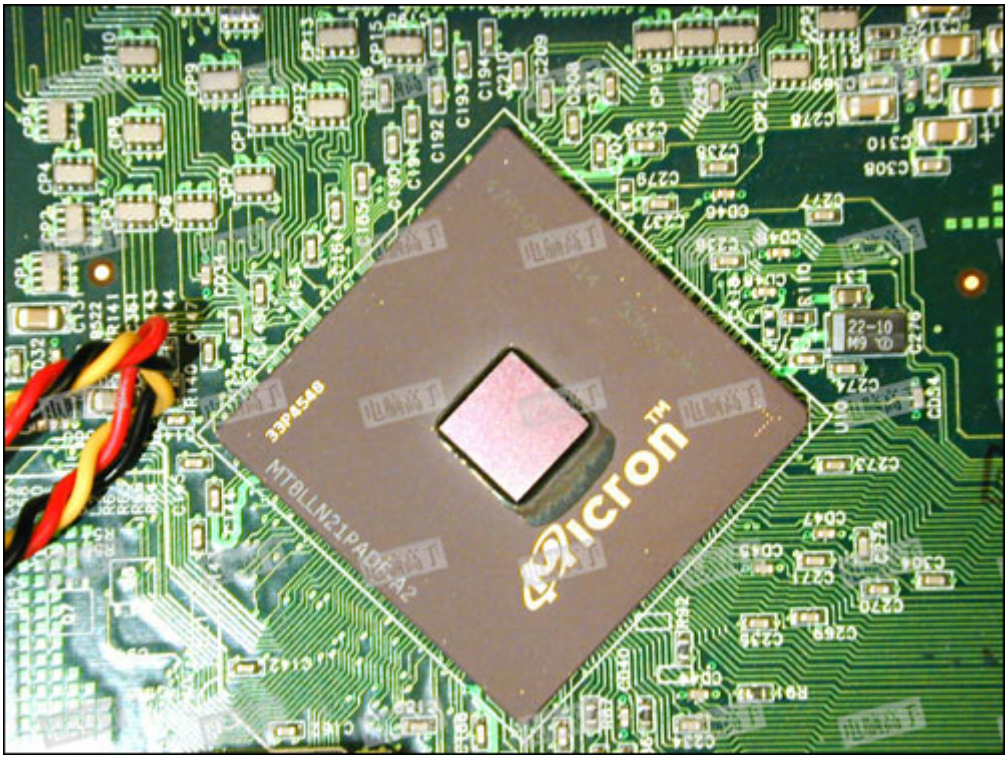
第 18 页：明日之星——DDR-Ⅱ与 DDR-Ⅲ（三）

DDR-Ⅱ内存图赏

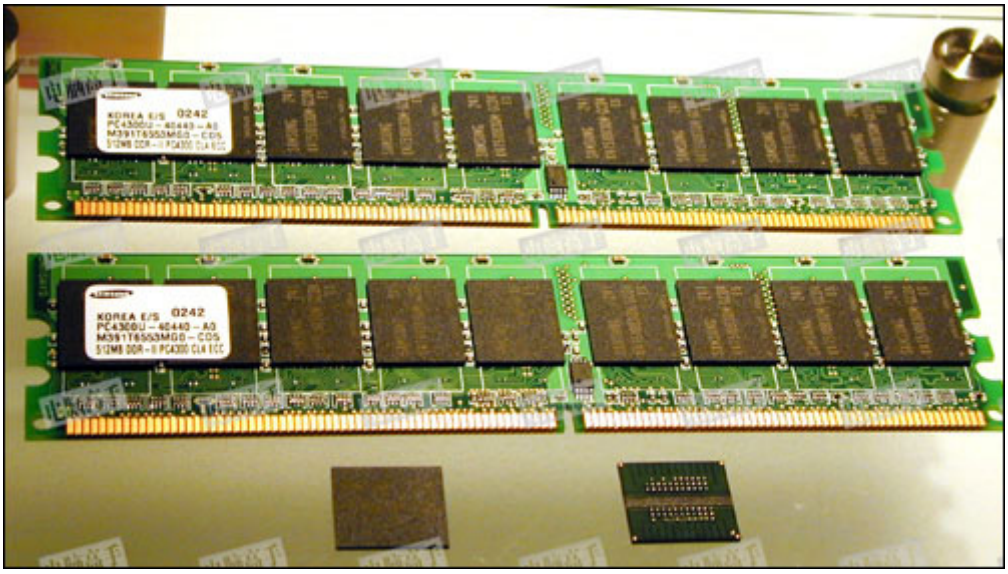

DDR II 533 SDRAM
Unbuffered DIMM
PC4300
Micron



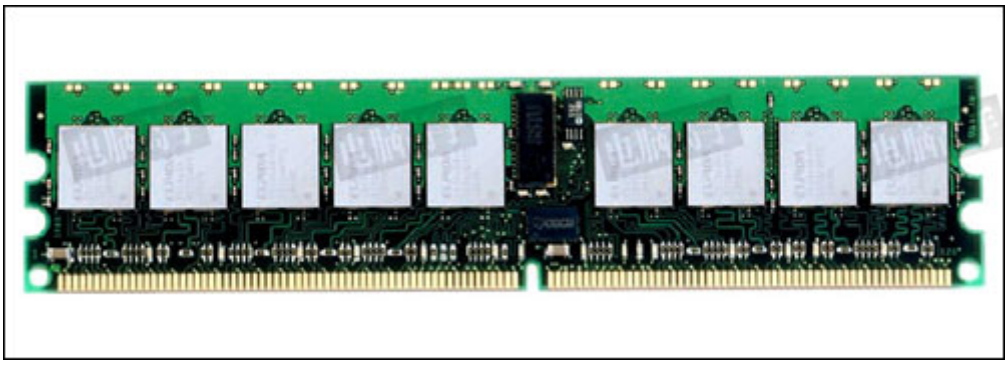

DDR II 533 SDRAM
Unbuffered DIMM
Micron



Micron 公司的 DDR-Ⅱ 533 内存与 DDR-Ⅱ分析/检测卡，它用来分析 DDR-Ⅱ内存的工作情况，并依此对内存的内部设计进行改进，值得注意的是系统平台用的是令人怀念的 Micron 自己的芯片组



三星公司展示的 DDR-Ⅱ 533 内存模组，模组标准为 PC4300，相应的，如果是 DDR-Ⅱ 400 将是 PC3200

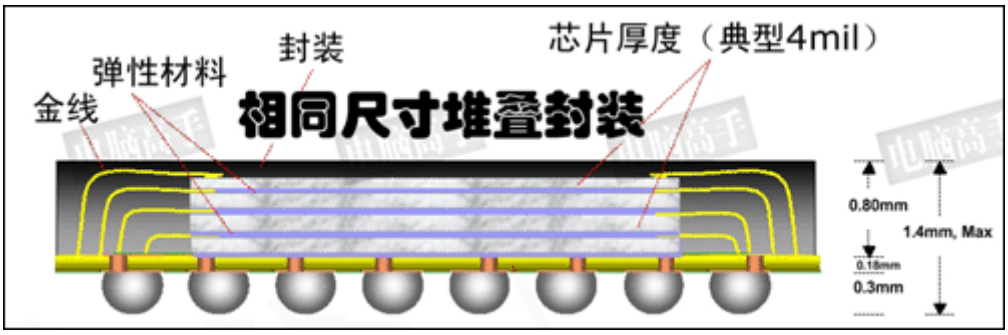


Elpida 公司的 DDR-II 内存模组，银光闪闪的 CSP 封装显得与众不同

2、DDR-II 时代的封装技术

可以肯定的是 TSOP-II 将在 DDR-II 时代彻底退出内存封装市场。并且将会出现改良型的 CSP——WLP（Wafer Level Packaging，晶圆级封装），它是比 CSP 更为贴近芯片尺寸的封装方法，由于在晶圆上就做好了封装布线，因此在可靠性方面达到了更高的水平。不过，外在的模样仍与现在的 CSP 封装差不多，WLP 更多的改进是在其内部。

另外值得一提的是为了应付更高容量的需求而采用的 SiP 封装技术，它是 System-in-a-Package 的缩写，有时又称之为 Stacked Pakage，可以看作是一种集成封装技术。它将多枚内存芯片核心堆叠在一起，然后统一封装成一颗芯片，在有限的面积内通过充分利用空间达到容量倍增的目的。SiP 并不是内存中专用的封装技术，原来是用于多种不同功能的芯片统一封装（如一颗嵌入式 CPU+DRAM 芯片）。



（上图可点击放大）

目前的 SiP 技术可以在 CSP 的基础上最多堆叠 4 枚内存芯片

3、DDR-III简介

DDR-III的设计始于 2001 年 5 月，目前只有一个大概的规格。按照 JEDEC 的计划，DDR-III将在 2007 年正式出台，数据传输率至少从 667MHz 开始，预取数据容量大于 4bit（很可能采用 RDRAM 那样的 8bit 设计），而且工作电压比 1.8V 更低，寄生干扰也将进一步减少。显然，它离我们更是遥远，还不到谈论它还的时候，要知道半导体技术日新月异， DDR-III完全有可能因此而中途改变设计。在此，我们就当个小花边新闻吧。

第 19 页：没有我不行——内存模组（上）

内存模组是内存在 PC 系统中的最终体现形式，所以在本专题的最后，我们来简要谈谈内存模的类型和未来的发展情况。不过，本章节只介绍 DIMM，而不涉及 RIMM（其实两者的很多概念是相通的）。目前经常见到的模组主要有五种：

1、 Unbuffered DIMM：无缓冲型模组，这是我们平时所用到的标准 DIMM，分有 ECC 和无 ECC 两种，简称 Unb-DIMM。

2、 Regustered DIMM：寄存型模组，这是高端服务器所使用的 DIMM，分有 ECC 和无 ECC 两种，但市场上几乎都是 ECC 的，简称 Reg-DIMM。

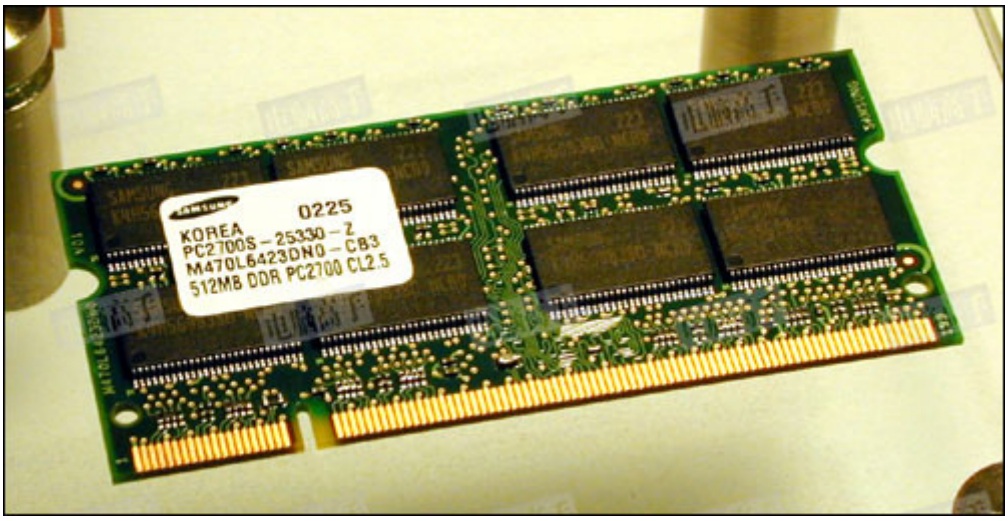
3、 SO-DIMM：Small Outline DIMM，小外型 DIMM，笔记本电脑中所使用的 DIMM，分 ECC 和无 ECC 两种，DDR-Ⅱ时代仅有无 ECC 的型号。

4、 Micro-DIMM：微型 DIMM，供小型笔记本电脑或手持式设备使用的 DIMM。

5、 Mini-DIMM：DDR-Ⅱ时代新出现的模组类型，它是 Regustered DIMM 的缩小版本，用于刀片式服务器等对体积要求苛刻的高端领域。

各类型内存 DIMM 对比表

内存类型	SDRAM	DDR SDRAM	DDR-II SDRAM
Unbuffered DIMM	168pin（可 ECC）	184pin（可 ECC）	240pin（可 ECC）
Registered DIMM	168pin（可 ECC）	184pin（可 ECC）	240pin（可 ECC）
SO-DIMM	144pin（可 ECC）	200pin（可 ECC）	200pin（无 ECC）
Micro-DIMM	144pin（无 ECC）	172pin（无 ECC）	200pin（无 ECC）
Mini-DIMM	无	无	244pin（仅 ECC）
DIMM 的 PCB 层数	4/6	6	6

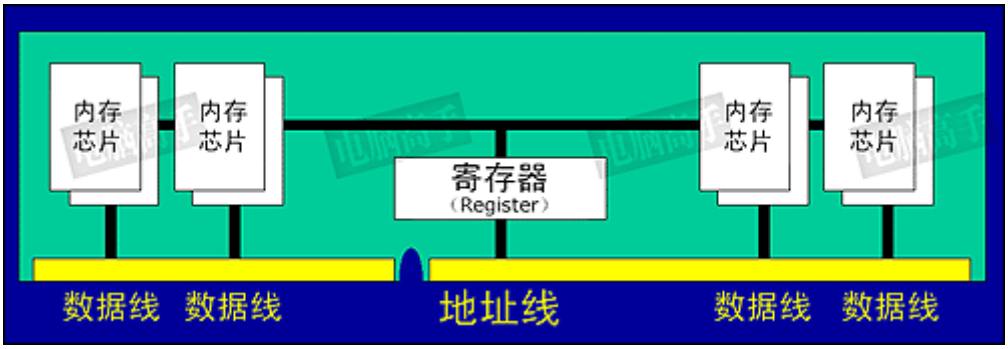


三星公司 DDR-333 标准的 SO-DIMM，容量高达 512MB

本文将重点讲一下 Unb 与 Reg-DIMM，和未来模组技术的发展

一、Unb 与 Reg-DIMM 的区别

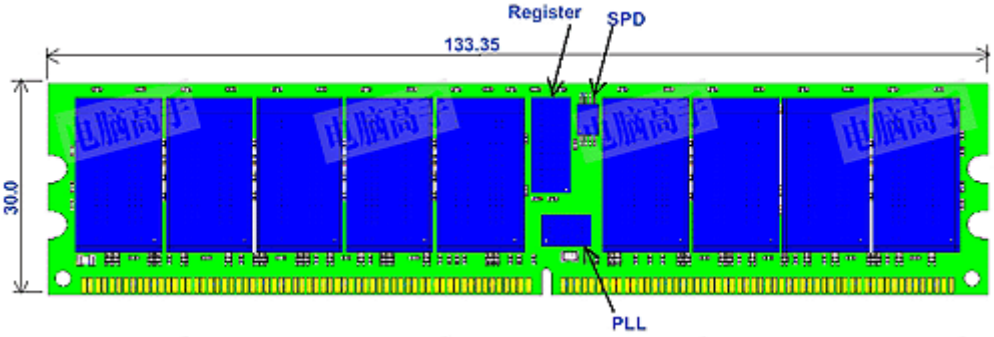
Unb 与 Reg-DIMM 的最大区别在于模组上有无寄存器。在高容量模组上，内存芯片数量很多，而且在需要大容量内存的工作场合，内存模组的安插数量也是很多的，这使命令与寻址信号的稳定性受到了严峻考验。很多芯片组的资料中都说明只有使用 Reg-DIMM 才能达到标称的最高内存容量，从这点就能猜到寄存器的作用——稳定命令/地址信号，隔离外部干扰。



Reg-DIMM 工作示意图，命令与地址信号通过寄存器中继传输至内存芯片

在工作时，命令地址信号会先送入寄存器进行“净化”并进入锁存状态，然后再发送至内存芯片，芯片中的数据则不经过寄存器而直接传向北桥。由于要经过中继传输，所以内存操作的时序也会因此而增加一个时钟周期，这是它所带来的一个弊端，但在高端应用中，内存系统的稳定可靠的重要性远在性能之上，所以 Reg-DIMM 一般只用于高端市场，并且需要芯片组的支持才行（主要是 Reg 所引起的时序变化）。而在高端设备中，ECC 基本都是必须的，因此市场上的 Reg-DIMM 也都无一例外的是 ECC 型模组，虽然也有无 ECC 的 Reg-DIMM 设计标准。

另外，为了保证内存工作时钟的稳定，Reg-DIMM 上还要有一颗 PLL 对时钟信号对主板发来的时钟信号进行跟踪/锁定。在 SDRAM 时代，这并不是必须的设计，但到了 DDR 时代，由于对时钟的敏感性，PLL 成为了必备元件。



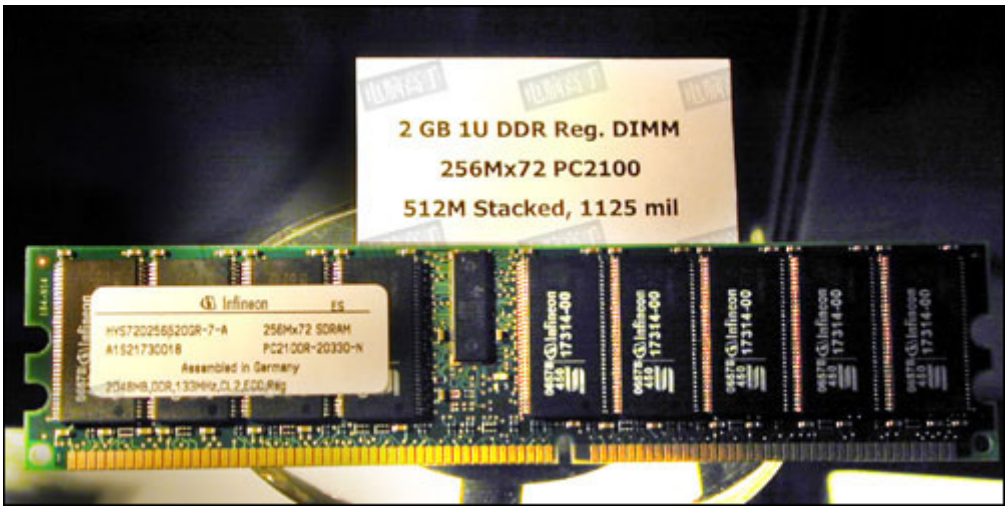
DDR 内存模组的结构图，寄存器与 PLL 是它相对于 Unb-DIMM 的最大不同

现在再回头看看 Unb-DIMM，就很明白了。它关键就少了寄存器，但为什么不称之为 Unregistered-DIMM 呢？其实，Buffered 与 Registered 是 Reg-DIMM 的两种工作模式，前者在 Reg-DIMM 上并不常用，它是以时钟异步方式工作的，输出信号的再驱动不与时钟同步，Registered 模式下输入信号的再驱动则与时钟同步。显然，Buffered 模式下的性能要更低一些。不过，从原理上讲 Registered 模式也是一种缓冲操作，只是与时钟同步而已。在 SDRAM 的 Reg-DIMM 上，Buffered 与 Registered 模式通过 REGE 信号控制，但到了 DDR SDRAM-DIMM 时代，可能由于性能的原因 Buffered 模式被取消了。

在 Unb-DIMM 上，没有寄存器也就没了这个 Buffer，但它仍可具备 ECC 功能。这里需要强调的是，ECC 与 Registered 是两码事，前者是在逻辑上保证数据的安全，后者是在物理上保证内存系统的稳定工作。

提示：什么是 ECC?
ECC 的全称是错误检测与校正（Error Checking and Correction），他通过纠错码技术（也叫 ECC，Error Correction Code）来检查并纠正数据中的错误。所使用的 ECC 基本都是汉明码，它可以纠正一位数据的错误（详见本刊 2000 年第 12 期《让我们谈谈 RAID（二）》一文）。ECC 是额外生成的数据，在目前系统中，对于 64bit 的 DIMM，用 8bit 的汉明码来进行负责纠错，这样 ECC 型 DIMM 的位宽就从 64bit 变成了 72bit。 ECC 的生成与纠错由北桥负责，在向内存写入数据的同时，北桥生成 ECC 通过 DIMM 上专用的引脚传给专用 ECC 芯片进行存储，而不是将 ECC 打散插入原有数据再统一存储。读取时，由 ECC 芯片随数据发送相应的 ECC 代码通过专用的引脚传给北桥并进行校验。

ECC 型 Unb-DIMM 示意图，其中的专用 ECC 芯片就用来存储 ECC 数据，在无 ECC 型 DIMM 上这颗芯片被取消



德国 Infineon 公司推出的容量高达 2GB 的 PC2100 Reg-DIMM

第 20 页：没有我不行——内存模组（下）

二、DIMM 引脚的基本设计

讲完 Unb-DIMM 与 Reg-DIMM 的不同之后，现在我们来看看 DIMM 引脚上的不同。其实，从内存芯片的引脚上就能推断出一些DIMM的引脚，因为芯片最终要通过DIMM来与主板打交道的。

首先，DIMM 肯定要有 64 个引脚用来数据的传输，而且要有 Ax 地址线、L-Bank 地址线、片选、数据掩码、电源、RAS、CAS……等信号，另外，ECC 型与 Reg 型 DIMM 要有额外的标定引脚，下面我就以 SDRAM 和 DDR SDRAM 为例，分 Unb-DIMM 和 Reg-DIMM 来介绍一下 DIMM 都包含有哪些的引脚。

SDRAM-DIMM 与 DDR SDRAM-DIMM 引脚数量及定义（"-" 表示没有此引脚）

类别	引脚名称	定义	SDRAM DIMM				DDR SDRAM-DIMM			
			Unb	ECC	Reg	ECC	Unb	ECC	Reg	ECC
电源	Vref	参考电压	-	-	-	-	1	1	1	1
	VddID	Vdd 标识	-	-	-	-	1	1	1	1
	Vdd	电源	17	17	17	17	9	9	9	9
	VddQ	芯片 I/O 电源	-	-	-	-	16	16	16	16
	Vss	接地	18	18	18	18	22	22	22	22
地址/命令	CS	片选（注 1）	4	4	4	4	2	2	2	2
	Ax	地址线	14	14	14	14	14	14	14	14
	BA（注 2）	L-Bank 地址线	2	2	2	2	3	3	3	3
	RAS	行选通脉冲	1	1	1	1	1	1	1	1
	CAS	列选通脉冲	1	1	1	1	1	1	1	1
	WE	写允许	1	1	1	1	1	1	1	1
时钟	CK	时钟	4	4	4	4	3	3	1	1
	CK#	反相时钟	-	-	-	-	3	3	1	1
	CKE	时钟有效	2	2	2	2	2	2	2	2
SPD	SA	SPD 地址线	3	3	3	3	3	3	3	3
	WP	SPD 写保护	1	1	1	1	-	-	-	-
	SDA	SPD 数据 I/O	1	1	1	1	1	1	1	1
	SCL	SPD 时钟	1	1	1	1	1	1	1	1
	VddSPD	SPD 电源	-	-	-	-	1	1	1	1
数据与掩码	DQ	数据 I/O	64	64	64	64	64	64	64	64
	DQS	数据选取脉冲	-	-	-	-	8	9	8	9
	DQM	数据 I/O 掩码	8	8	8	8	-	-	-	-
	DM	数据写入掩码	-	-	-	-	8	9	8	9
	CB	ECC 校验数据	-	8	-	8	-	8	-	8
寄存型专用	Reg	寄存型模组	-	-	1	1	-	-	-	-
	Reset	寄存器复位	-	-	-	-	-	-	1	1
保留	NC（注 3）	未使用的引脚	26	18	25	17	19	9	22	12
引脚数量（Pin）			168				184			

注：1、 DDR SDRAM-DIMM 中的 CS 信号数量最高也可达 4 个，但目前基本都是 2 个 CS 信号设计，如果为 4 个， NC 的数量就要减少两个

2、 DDR SDRAM-DIMM 中定义了 3 个 L-Bank 地址线，但目前只采用两个的设计，用到 3 个 L-Bank 地址线的将是 DDR-II 内存

3、 使用引脚是指在设计中预留的引脚，供未来的一些设计使用

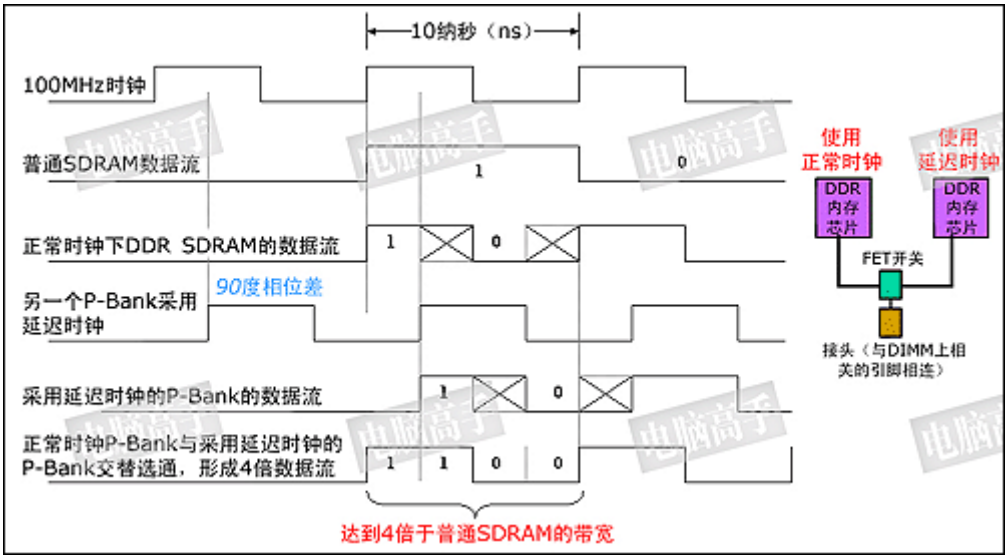
（上图可点击放大）

从上面的引脚信号列表中，大家应该能了解到 DIMM 的大体情况了。其中很多信号定义是不是非常熟悉？从中可以看到，在 DDR SDRAM 时代已经为 8 个 L-Bank 做好了准备，但业界显然没有利用到它，不光是内存厂商，DDR 芯片组中似乎没有支持 8 个 L-Bank 的设计。还有就是 CS 信号，从 SDRAM 到 DDR，都有 4 个 CS 的设计，但目前的 DIMM 还都是双 P-Bank 的设计，不同的是，SDRAM-DIMM 上，4 个 CS 是必须的，两个 CS 对应一个 P-Bank 芯片集，但到了 DDR 时代，可能是技术与工艺的进步，一个 CS 就控制了一个 P-Bank。总之，当我们了解了芯片的引脚设计后，对 DIMM 的引脚组成也就不再陌生。有兴趣的读者，可以自行深入研究。

提示：什么是 SPD？它是怎么工作的
SPD（Serial Presence Detect），笔者翻译为“配置（存在位）串行探测”，而不是“连续存在探测”，如果单从字意上理解，后者的翻译并没有问题，但从其真正用意与工作方式来看，前者更准确一些。为什么呢？下面具体说说。 SPD 是一组关于内存模组的配置信息，如 P-Bank 数量、电压、行地址/列地址数量、位宽、各种主要操作时序（如 CL、tRCD、tRP 等）……它们存放在一个容量为 256 字节的 EEPROM（Electrically Erasable Programmable Read Only Memory，电擦除可编程只读存储器）中。实际上，包括 RIMM 在内，SPD 的有效信息只用了 128 个字节。一般的，一个字节至少对应一种参数，有的参数需要多个字节来表述（如产品续列号，生产商在 JEDEC 组织中的代码）。其中，一个字节中的每个 bit 都可能用来表示这一参数的具体数值。由于 SPD 的信息很多，在此就不一一列出了，有兴趣的读者可以参阅相关文档。 SPD 的信息由模组生产商写入，主要用途就是协助北桥芯片精确调整内存的物理/时序参数，以达到最佳的使用效果。如果在 BIOS 中将内存设置选项定为“By SPD”，那么在开机时，北桥会根据 SPD 中的参数信息来自动配置相应的内存时序与控制寄存器，避免人为出现调校错误而引起故障。当然，对于 DIYer 来说，也可以自由调整时序与控制参数（物理参数仍要借助 SPD 或北桥自己检测来确定）。 那么 SPD 的信息是怎么传送的呢？从 DIMM 的引脚列表中，我们知道了有关 SPD 的引脚信号，与检测操作有关的分别是三个地址线、一个数据线和一个时钟。SPD 所使用的 EEPROM 的时钟频率大都为 100KHz，SPD 的数据通过一条数据线进行串行式交换，这就是所谓“串行探测”的含义，理论上数据的交换速度为 100Kbps，即 12500 字节/秒，读取 128 个字节约需 0.01 秒。在操作中，地址线提供 SPD 芯片地址（因为每个 DIMM 上都有一个 SPD，目前常见的 SPD 地址线为 3 个，也就意味着可访问最多 8 个 DIMM 上的 SPD 芯片），然后通过数据线用 8 个 SPD 时钟周期串行传输字节地址，之后指定字节的数据再从数据线串行传出，并由系统识别，然后用来配置相关寄存器。 显然，当我们了解了 SPD 的具体信息后，完全可以通过 SPD 的信息格式与读写操作的方式入手，来编一个 SPD 识别软件，有关编程基础的朋友不妨试试。

三、QBM 型 DIMM

之所以在前文没有介绍四倍带宽内存（QBM，Quad Band Memory），就是因为不是针对芯片的技术，而针对 DIMM 的技术。它诞生于 DDR 时代，是 Kentron 公司为了解决 DDR 带宽提供困难而提出的设计方案。主要的思路就是让 DIMM 上的两个 P-Bank 交错工作，而交错的时钟周期为原始时钟的 1/4，即相位相差 90 度。



（上图可点击放大）

QBM 的工作时序图，第二个 P-Bank 的工作时钟与第一个 P-Bank 相差 90 度（1/4 周期），这样在第一个 P-Bank 时钟的高/低电平中部就是第二个 P-Bank 的触发点，两者都是 DDR 传输，从而在一个时钟周期内完成 4 次数据触发，实现四倍带宽

为了控制两个 P-Bank 中同一位置的芯片交错工作，模组上要为每组芯片（在 QBM 模组上，一个 P-Bank 位于一侧，两个 P-Bank 中位置相对的芯片为一组）设置一个开关，以控制不同 P-Bank 间的通断。并且还要为延迟 1/4 周期的 P-Bank 提供一个 PLL 以保证相位差的准确性。

QBM 的设计是非常巧妙的，经过对现有的 DDR 模组的改装，配合新的芯片组即可将带宽提高一倍，有点类似于 32bit RIMM，在一个模组上实现了双通道的功能，只是 QBM 不是双通道并发，而是双通道交错，通过更高的传输频率实现高带宽。但是新增加的开关与 PLL 元件将增加一定的成本，不过与其所能提供的带宽相比，还是比较划算的。

	QBM533 (DDR I-266)	DDR I-333	DDR I-400	DDR II-400/533
	512MB	512MB	512MB	512MB
带宽	✓ PC4200	PC2700	PC3200	PC3200/4200
PCB成本	$2	$2	$2	目前不可用
内存芯片成本	✓ $104 ($6.50x16)	$128 ($8x16)	$160 ($10x16)	目前不可用
STD元件	$3	$3	$3	目前不可用
开关IC	$12 ($1.50 x 8)	N/A	N/A	N/A
PLL	$2	N/A	N/A	N/A
装配/检测	$5	$5	$5	N/A
模组成本	✓ $128	$138	$170 可用性有限	目前不可用 最早要到2004年

（上图可点击放大）

Kentron 公司给出的 QBM 与其他内存方案的成本比较表，从中可以看出 QBM 有较高的性价比

但是，开关元件的同步性对于 QBM 是个考验，时钟频率越高，对开关的控制精度就越高。目前，有不少大牌的模组厂商（如 Infineon）都在论证 QBM 的可行性与可靠性，据部分厂商透露，在使用 DDR-333 或之前标准时，QBM 的表现良好，但到了 DDR-400，QBM 的可靠性就会降低，如果克服这一个问题，那么延迟又会大幅提高。所以，QBM 目前的可行标准是 QBM533（DDR-266）和 QBM667（DDR-333）。VIA 在 P4X800 中将要支持的标准也是 QBM533，虽然不能使用 DDR-400，但它的 5.4GB/s 带宽（QBM667）在目前仍是无敌的。

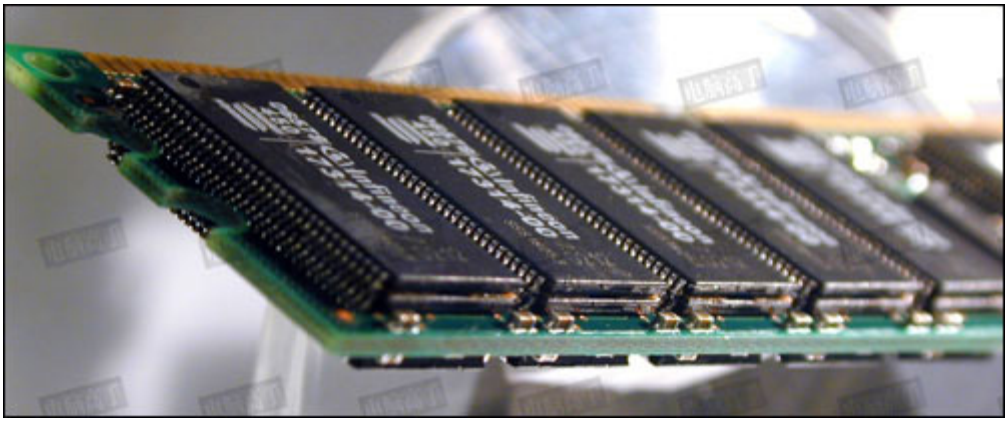
不过，由于 QBM 是针对模组的技术，所以理论上 QBM 可适用于任何 DIMM，包括 SDRAM 和 DDR-II 的 DIMM，Kentron 也有此计划研制 QBM 型 DDR-II DIMM，以保持 QBM 的生命力。另外，Kentron 已将 QBM 标准上报 JEDEC 审批，目前还不知能否通过。很多模组厂商也都在观望，毕竟 QBM 转产是很容易的，就看市场情况了。所以，QBM 虽然设计巧妙，但得到的支持并不强劲，以 Kentron 及 QBM 联盟的生产能力，显然不足以完成普及任务，一切就看 P4X800 的市场效果了。

三、模组的堆叠装配

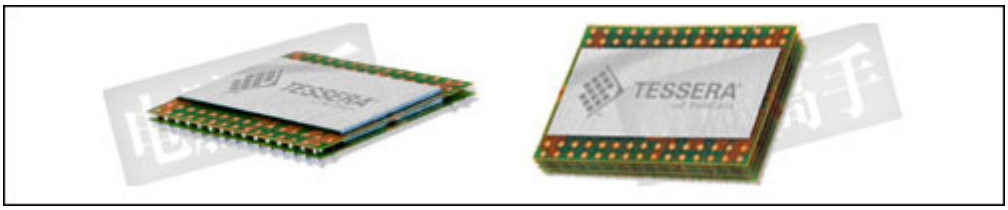
当内存芯片容量无法迅速提高的时候，高容量模组如何设计就体现了厂商间的真正实力，由于高容量模组针对的是高端应用市场，所以谁能在容量上有所突破就意味着滚滚商机。就模组而言，芯片基本是固定的，所以芯片堆叠装配（Stack Assembly）技术就是增加容量的首选。

这方面除了 Elpida、Kentron、Kingston 等公司较早以前提出的 TCP、FEMMA、EPOC 等堆叠形式外（已有多篇文章介绍过，在此不再重复），著名的封装技术开发商 Tessera 公司（它在 1990 年因研制出 CSP 封装而闻名于世）近期宣布了他们的 4 枚芯片堆叠装配的模组技术

（TCP 与 EPOC 都是两芯片堆叠）——μZ Package，当然，芯片本身的封装也要有相应的调整。而 Infineon 公司也推出了普通 TSOP-II 技术的双芯片堆叠装配技术。显然，模组厂商都想利用有限的空间（毕竟在主板上插槽之间的距离是有限的）尽量提高装配容量，若再配合 SiP 封装形式的内存芯片，DIMM 的扩容就如虎添翼了。



Infineon 的采用 TSOP-II 堆叠封装的模组，容量高达 2GB



Tessera 公司为高容量模组开发的 4 枚芯片堆叠装配技术 μZ Package