

SDRAM 设计教程之 SDRAM 结构深入剖析

小梅哥出品

未经作者许可，严禁用于商业行为，例如开发板配套资料。

SDRAM 基本概念

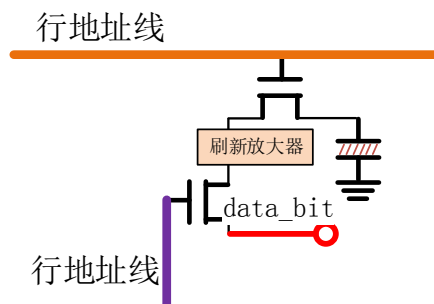
SDRAM 的全称即同步动态随机存储器（Synchronous Dynamic Random Access Memory），这里的同步是指其时钟频率与对应控制器（CPU/FPGA）的系统时钟频率相同，并且内部命令的发送与数据传输都是以该时钟为基准；动态是指存储阵列需要不断的刷新来保证数据不丢失；随机指数据的读取和写入可以随机指定地址，而不是必须按照严格的线性次序变化。

在 FPGA 和 DSP 系统中，包括现在的 MCU 系统，我们还经常见到一种名为 SRAM 的器件，该器件名字和相比 SDRAM 少了一个 D，也就表明该存储器不存在 SDRAM 的动态刷新特征。同时，这里 SRAM 的 S 也不是代表同步的意思。相反，SRAM 属于异步器件，其在工作时是不需要外部提供时钟的，SRAM 的全称叫做 Asynchronous Static RAM，即异步静态随机存储器。

在存储数据的物理结构方面，SDRAM 使用电容的电荷存储特性存储数据，而 SRAM 使用 CMOS 晶体管存储数据，因此，从这个结构方面也就决定了 SDRAM 的运行功耗要远远低于 SRAM。同时，由于使用晶体管存储数据，要能够正确的存储一位数据，需要最少 6 个晶体管。因此，从芯片面积上来说，单片 SRAM 芯片的容量不可能做到很高。目前常见的有 512K 字节和 1024K 字节容量的，而 SDRAM 则可以最多做到 512Mbit（64M 字节）。所以，在需要大量存储数据的场合，SDRAM 就成为了首选。

SDRAM 存取原理

SDRAM 存储数据是利用了电容能够保持电荷以及其充放电特性。一位数据的存取电路如下图所示，该存储单元主要由行列选通三极管，存储电容，刷新放大器组成。对于这一位的数据，要想将其读取出来，首先需要打开行地址，然后打开列地址，则电容的电平状态就能呈现在数据线（data_bit）上，即实现了读取。或者，数据线上的电平值被送到电容上，从而实现写入数据。

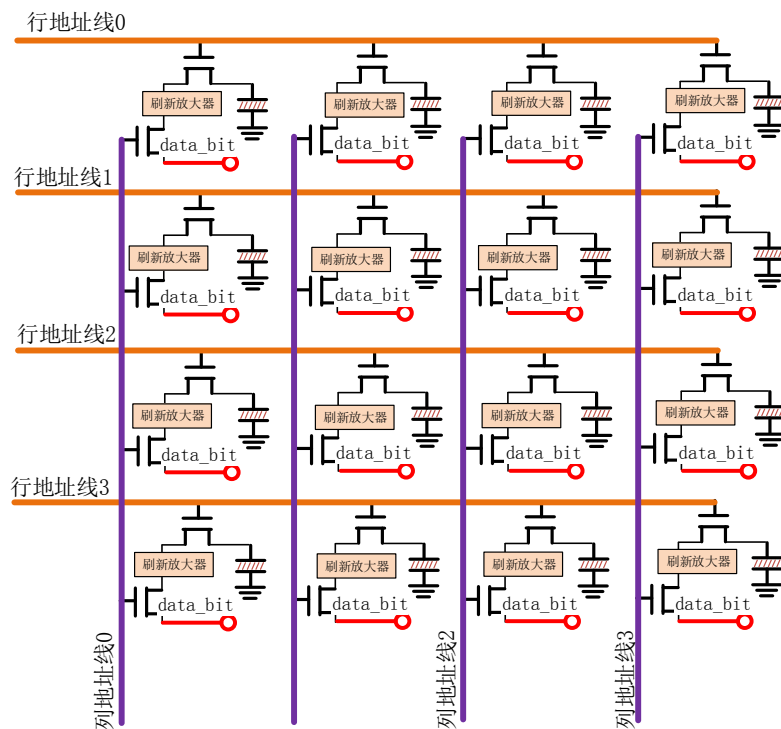


但是，打开行地址（激活）是需要一定时间的，即从打开行地址到可以打开列地址进行读写，这之间有一定的时间间隔，这个间隔叫做 **tRCD(ACTIVE-to-READ or WRITE delay)**，对于不同型号规格的器件，这个值是不一样的。例如，某个型号的 SDRAM 器件，对于最大运行频率为 133M 的器件，该值为 15ns。当 SDRAM 工作在 133M 时，其一个时钟周期为 7.52ns，因此，在实际操作时，必须等待 2 个时钟周期之后才能选通列地址。而另一款 SDRAM 器件，其最大运行频率也为 133M，但是 tRCD 值为 20，因此，必须等待 3 个时钟周期（ $7.52 \times 3 > 20$ ）之后才能打开列地址。

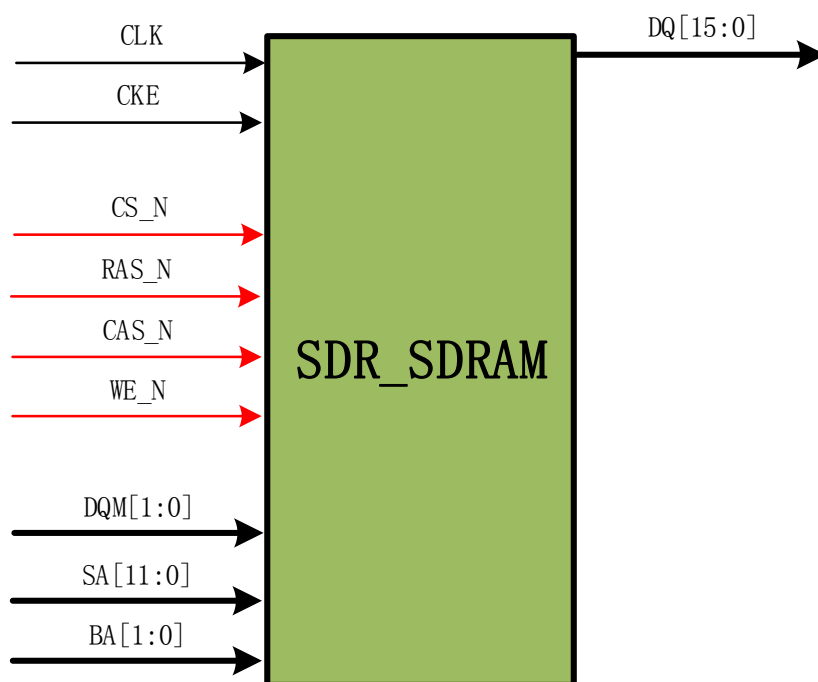
当列地址被打开后，数据并不是立即出现在最终的数据总线引脚上，而是有若干个时钟的延迟，这个延迟叫做列选通潜伏期（CL, CL = CAS READ latency），注意，列选通潜伏期的值针对不同速度等级的器件，其值是不一样的。

Speed Grade	Clock Frequency	Access Time	
		CL = 2	CL = 3
-6A	167 MHz	—	5.4ns
-7E	143 MHz	—	5.4ns
-75	133 MHz	—	5.4ns
-7E	133 MHz	5.4ns	—
-75	100 MHz	6ns	—

下图是 SDRAM 内部存储矩阵的一个简化模型，对于每一行列，只展示了 1bit 的数据。主要是为了展示行列地址与存储电容的对应关系。实际读写时，打开一个确定的行和列，就能唯一确定一个存储单元。



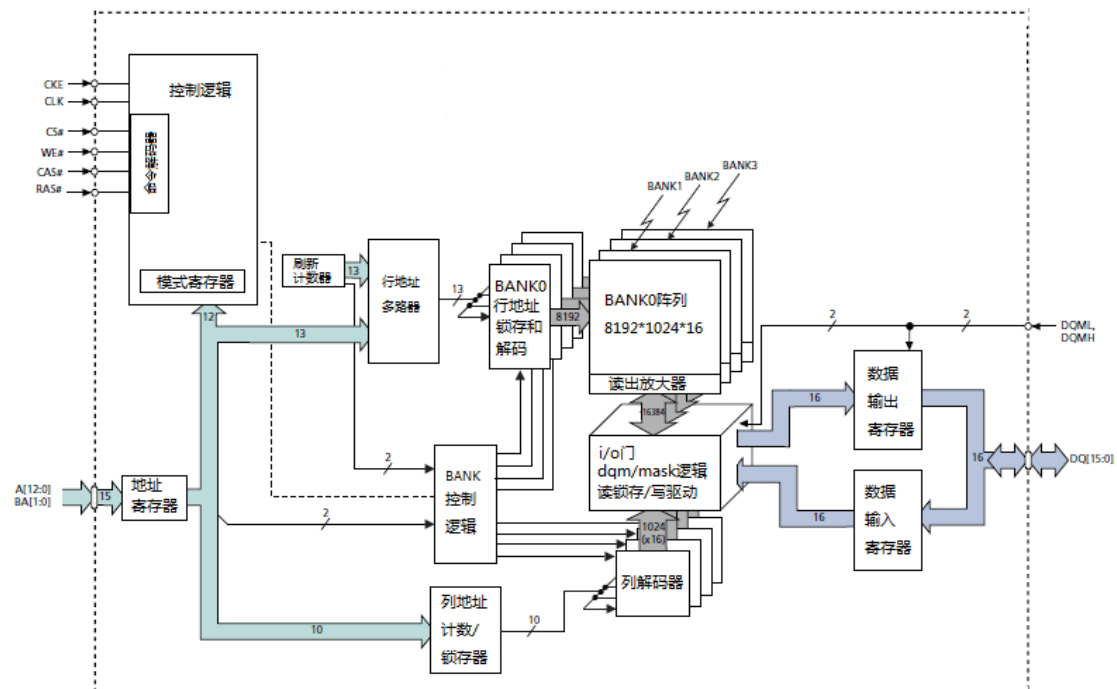
SDRAM 器件引脚说明



管脚	说明	功能描述
----	----	------

CLK	系统时钟	时钟上沿对所有输入进行采样
CKE	时钟使能	屏蔽系统时钟，以冻结当前操作，当该引脚为高电平时，所有引脚电平才能被正确送入 SDRAM 芯片。
CS_N	片选	用于屏蔽和使能所有输入端口，但不包括 CLK, CKE 和 DQM，低电平有效
RAS_N	行地址选通	该信号为低时，在时钟的上沿锁存行地址，使能行访问和预充电
CAS_N	列地址选通	该信号为低时，在时钟的上沿锁存列地址，使能列访问
WE_N	写使能	该信号为低时，使能写操作和预充电
BA[1:0]	Bank 地址	
SA[12:0]	地址总线	关于地址线上的数据，在不通的命令中有不同的意义。详见下文对 A[12:0]的功能描述。
DQM[1:0]	数据掩码	L(U)DQM，低（高）字节掩码，当其为高时，下一个时钟的上沿，数据总线的低（高）字节为高阻态
DQ[15:0]	数据总线	数据输入输出复用

SDRAM 功能框图



SDRAM 特性

- 通常情况下，SDRAM 是拥有四个 BANK 的动态刷新存储器，存储器工作在 3.3V 的电压下，拥有一个同步接口，SDRAM 的所有信号都在时钟信号的上升沿被寄存。
- 对于 256M 的 SDRAM，每个 BANK 存储 64Mbit（67108864bit）的数据。当 SDRAM 的数据位宽为 4bit 时，这些数据组成 8192 行*2048 列，每个存储单元存储 4bit 数据。当 SDRAM 的数据位宽为 8bit 时，这些数据组成 8192 行*1024 列，每个存储单元存储 8bit 数据。当 SDRAM 的数据位宽为 16bit 时，这些数据组成 8192 行*512 列，每个存储单元存储 16bit 数据。
- 对于常用的一款 SDRAM 芯片 HY57V281620，该芯片总共有 128Mbit 的存储空间，它具有 4 个 BANK，每个 BANK 存储 64Mbit（33554432bit）的数据。SDRAM 的数据位宽为 16bit，这些数据组成 4096 行*512 列，每个存储单元存储 16bit 数据。
- 对于 SDRAM 的读写是以突发的方式进行的，对 sdram 的获取（读或者写）是从一个指定的地址开始，并按照编程好的数量(长度)的地址，以编程好的数据顺序读写数据。对 SDRAM 的获取(读写)是以一个激活命令开始，然后跟随一个读或者写命令。伴随着激活命令，A0-A12 上同时被寄存的数据是期望获取的位置的行地址，BA0 和 BA1 上被寄存的数据是期望获取的位置的 BANK 地址。伴随着读写命令，A0—A8 上同时被寄存的是期望获取的位置的列地址首地址（突发获取）。

SDRAM 操作命令介绍

对 SDRAM 的操作，是通过对应命令来实现的，这些命令由 CS_N,RAS_N,CAS_N,WE_N 这几个控制信号组合成不同的状态以表示。同时，ADDR 总线和 DQM 总线作为辅助信号，提供与命令相对应的参数（设置模式寄存器）或地址（读写命令）。下表为 SDRARM 工作中使用到的所有命令与对应的控制信号线的状态

Nam e (Function)	CS#	RA S#	CA S#	WE#	DQM	LA DDR	DQs
COMMAND INHIBIT (NOP)	H	X	X	X	X	X	X
NO OPERATION (NOP)	L	H	H	H	X	X	X
ACTIVE (选择 BANK 和行)	L	L	H	H	X	Bank/row	X
READ (选择 BANK 和列，并启动读突发)	L	H	L	H	L/H8	Bank/col	X
WRITE (选择 BANK 和列，并启动写突发)	L	H	L	L	L/H8	Bank/col	Valid
BURST TERMINATE （突发终止）	L	H	H	L	X	X	Active
PRECHARGE (关闭一个或多个 BANK 中的行)	L	L	H	L	X	Code	X
AUTO REFRESH or SELF REFRESH (自刷新/自动刷新模式)	L	L	L	H	X	X	X
LOAD MODE REGISTER（加载模式寄存器）	L	L	L	L	X	Op-code	X
Write enable/output enable（使能数据写入/输出）	—	—	—	—	L	—	Active
Write inhibit/output High-Z（禁止数据写入/输出）	—	—	—	—	H	—	High-Z

SDRAM 命令详解

命令禁止（The COMMAND INHIBIT）

控制信号组成：	{CS_N, RAS_N, CAS_N, WE} = 4'b1xxx;
命令说明：	禁止新的命令执行。执行此命令时，不用顾及 CLK 是否时能（CKE），已经执行的命令不受影响。

空命令（NO OPERATION (NOP)）

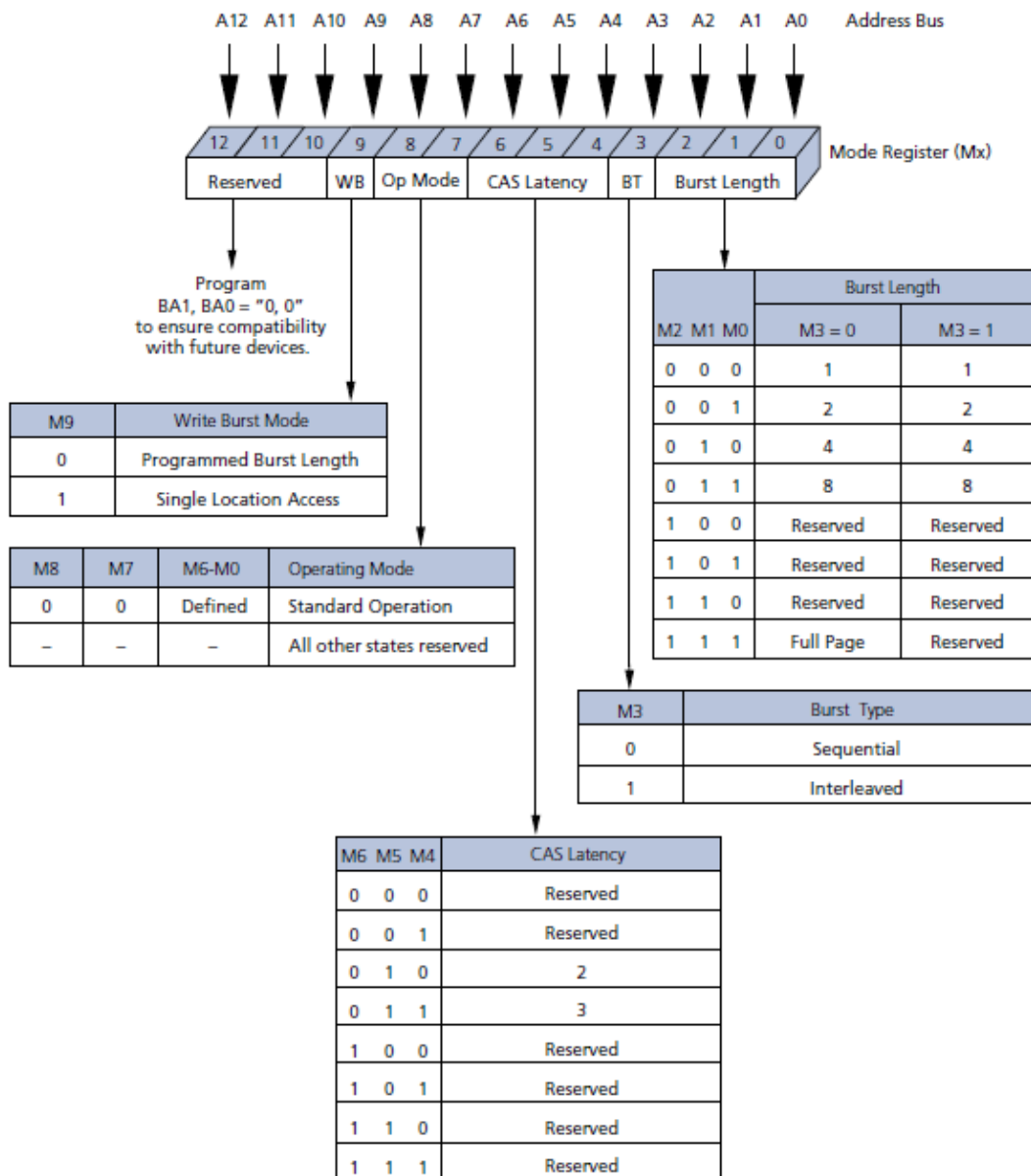
控制信号组成：	{CS_N, RAS_N, CAS_N, WE} = 4'b0111;
命令说明：	该命令主要给被选中的 SDRAM（CS_N 为低电平）传递一个无需要操作的信息， 主要是为了防止在 SDRAM 处于空闲或者等待状态时， 其他命令被写入 SDRAM。此命令对正在执行的操作没有影响。

加载模式寄存器（LOAD MODE REGISTER）

控制信号组成：	{CS_N, RAS_N, CAS_N, WE} = 4'b0000;
命令说明：	模式寄存器的值通过地址线 A0—A11 写入 SDRAM，加载模式寄存器 (LOAD MODE REGISTER)命令只有在所有的 BANK 都处于空闲状态（未激活）时才可进行，否则会出错。在执行了加载模式寄存器命令后，必须等待对应的响应周期（tMRD）后才能执行新的命令。

注意：在设置模式寄存器命令时，A0—A11 定义了模式寄存器的值，不同的值对应不同的模式设置（后面会讲到），A12 在此命令中没有用到，需要为低电平。模式寄存器中的各个位对应功能如下所示：

模式位	模式位对应功能
M0—M2	突发长度寄存器
M3	突发类型寄存器
M4—M6	列选通潜伏期设置寄存器
M7—M8	运行模式设置寄存器
M9	写突发模式设置寄存器



SDRAM 模式寄存器详解

Operating Mode:

运行模式设定位 (M7,M8)，SDRAM 器件一般存在多种模式，例如标准模式，测试模式等等，但是对普通用户，只开放了标准模式。因此，我们在使用的时候，需要设置 M7 和 M8 为 00，这样才能使 SDRAM 器件进入我们所希望的标准模式。

Write Burst Mode:

模式寄存器的第 9 位 (M9) 控制 SDRAM 的写突发模式，如果 M9 为 0，那么对于 SDRAM 的读和写都将采用突发方式，具体突发长度由突发长度寄存器 M0—M2 设定。如果 M9 为 1，那么对于 SDRAM 的读，依然遵从突发长度寄存器设定的值进行，而对于 SDRAM 的写，则不再具备突发属性，每个写入命令只能写入一个数据。

Burst Type:

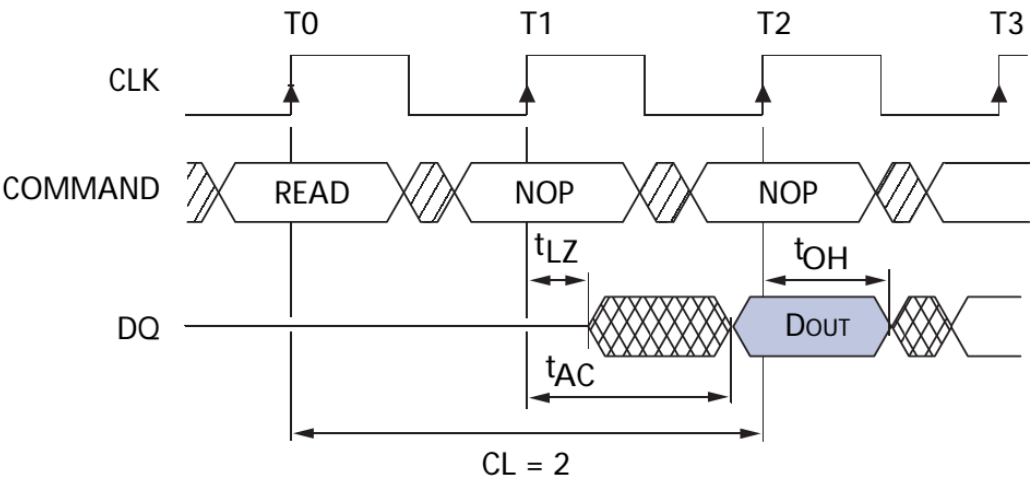
突发类型，突发类型分为两类，顺序和隔行，这个模式主要用来适应一些嵌入式系统的不同

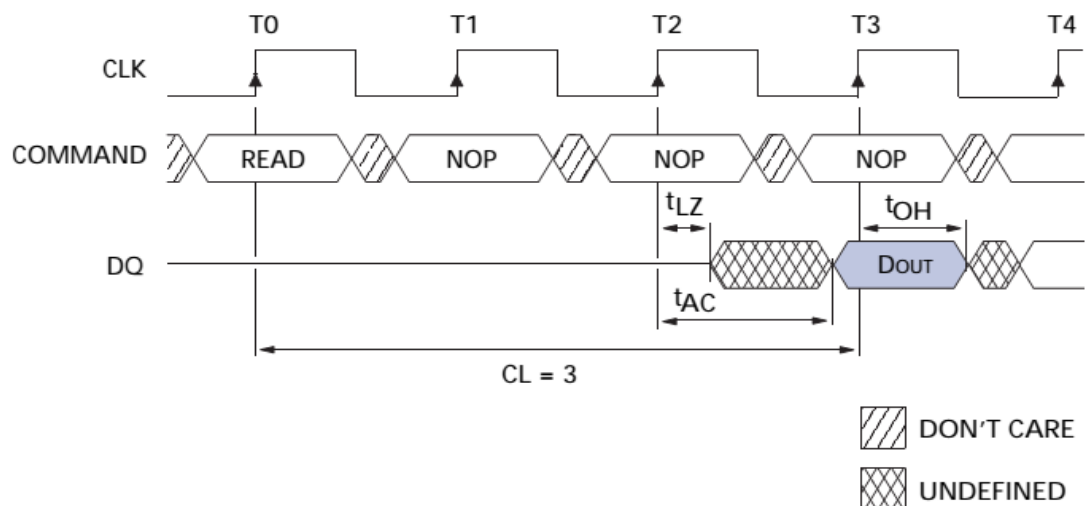
大端小端模式。

Burst Length	Starting Column Address		Order of Accesses Within a Burst	
			Type = Sequential	Type = Interleaved
2	A0			
	0		0-1	0-1
	1		1-0	1-0
4	A1	A0		
	0	0	0-1-2-3	0-1-2-3
	0	1	1-2-3-0	1-0-3-2
	1	0	2-3-0-1	2-3-0-1
	1	1	3-0-1-2	3-2-1-0
8	A2	A1	A0	
	0	0	0	0-1-2-3-4-5-6-7
	0	0	1	1-2-3-4-5-6-7-0
	0	1	0	2-3-4-5-6-7-0-1
	0	1	1	3-4-5-6-7-0-1-2
	1	0	0	4-5-6-7-0-1-2-3
	1	0	1	5-6-7-0-1-2-3-4
	1	1	0	6-7-0-1-2-3-4-5
	1	1	1	7-0-1-2-3-4-5-6
Full page (y)	n = A0-A11/9/8 (location 0-y)		Cn, Cn + 1, Cn + 2 Cn + 3, Cn + 4... ...Cn - 1, Cn...	Not supported

CAS Latency (CL) :

CL 是指从读命令被寄存到数据总线上有第一个有效数据间的时钟间隔(潜伏期)，这个潜伏期可以被设置为 2 个时钟或者 3 个时钟。





CAS Latency (CL) :

CL 是指从读命令被寄存在数据总线上有第一个有效数据间的时钟间隔(潜伏期)，这个潜伏期可以被设置为 2 个时钟或者 3 个时钟。

Speed	Allowable Operating Frequency (MHz)	
	CL = 2	CL = 3
-6A ¹	≤133	≤167
-7E ²	≤133	≤143
-75	≤100	≤133

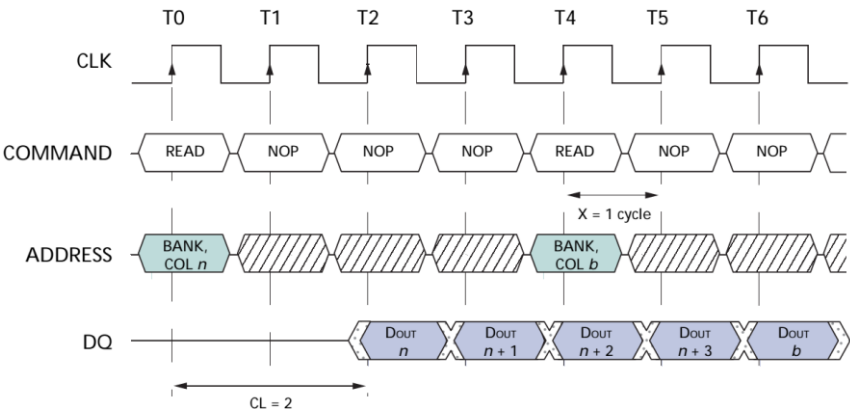
Notes: 1. -6A speed grade supports latency of 3-3-3 cycles (CAS latency- t_{RCD} - t_{RP}) at 167 MHz (MAX).
2. -7E speed grade supports latency of 2-2-2 cycles at 133 MHz (MAX).

激活/打开行地址和 BANK 地址 (ACTIVE)

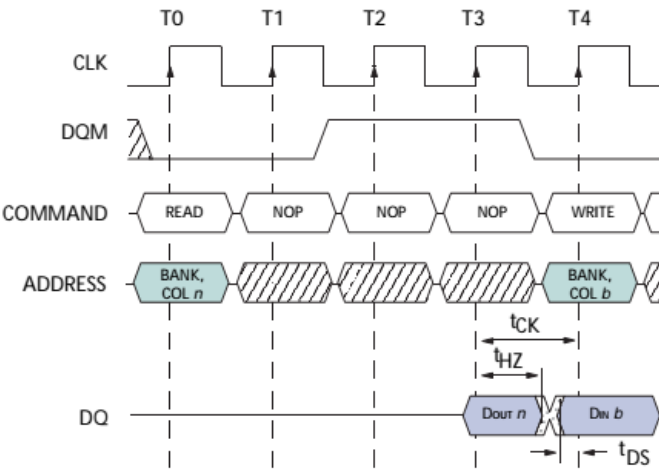
控制信号组成:	{CS_N, RAS_N, CAS_N, WE} = 4'b0011;
命令说明:	激活命令是用来为后续的操作打开(或者叫做激活)一个特定的 BANK。BA0 和 BA1 用来选择需要操作的 BANK，地址线 A0—A12 选择指定的行 (ROW)。该行会一直保持激活状态并可以进行读写，只有执行一个预充电命令 (PRECHARGE) 后，该行才会被关闭。同一个 BANK 中，每次只能激活一行。在某行已经激活的状态下，当需要对该 BANK 中的另一行进行操作时，必须首先执行一个预充电命令来关闭当前行，然后才能激活另一行。 在激活命令时，BA0 和 BA1 决定了激活哪一个 BANK，A0—A12 则决定了激活该 BANK 中的哪一行 (ROW)。

读/打开列地址和 BANK 地址（READ）

控制信号组成：	{CS_N, RAS_N, CAS_N, WE} = 4'b0101
命令说明：	读命令用来启动对一个已经激活行的突发读取操作。BA0 和 BA1 指定需要读取的 BANK，地址线上某些位指定需要读取的数据起始列地址（A0–A9, A11 (x4), A0–A9 (x8), or A0–A8 (x16)），A10 控制是否在突发读取完成之后立即执行预充电，即关闭当前行操作。若为高电平则在读取完当前行以后立即自动执行预充电操作（关闭当前行），若为低电平则不关闭该行，使其继续处于激活状态，以便于紧接着对该行执行新的读写操作。DQM 控制该位对应的数据是否被正常的读出，（例如，16 位的 SDRAM 芯片，DQM0 对应控制 DQ[7:0]，DQM1 对应控制 DQ[15:8]），若 DQM 为高电平，则迟于该 DQM 电平两个时钟周期的时候，数据总线上对应字节的状态会变为三态，即该字节数据被屏蔽。从 DQM 有效到数据总线状态对应发生变化，有两个时钟周期的延迟。



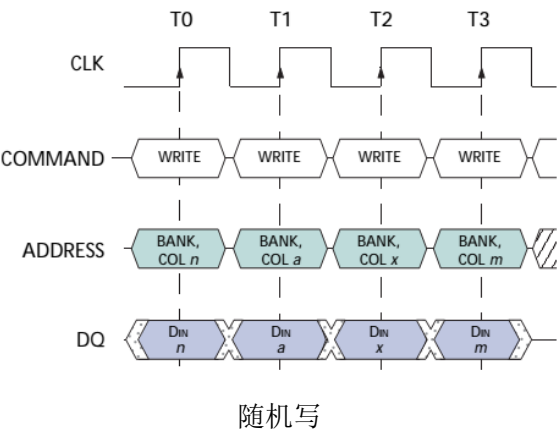
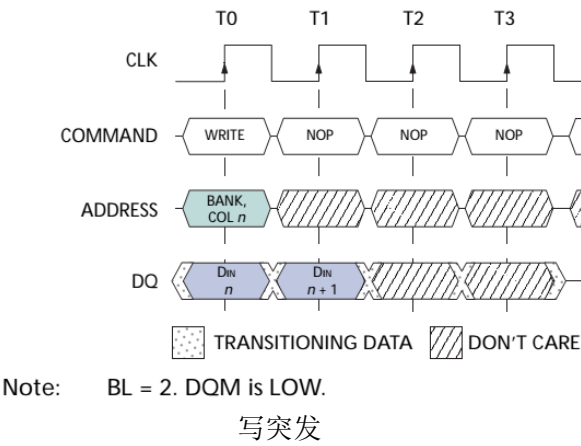
突发读

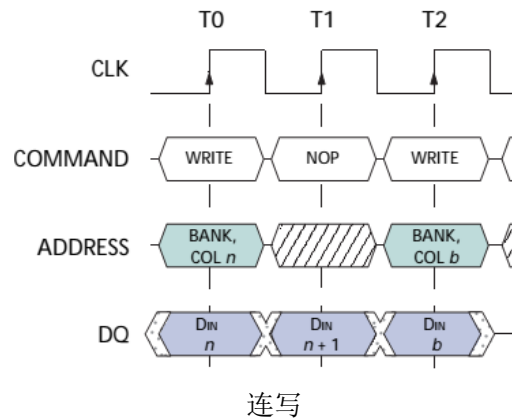


带屏蔽读

写/打开列地址和 BANK 地址（WRITE）

控制信号组成:	{CS_N, RAS_N, CAS_N, WE} = 4'b0100;
命令说明:	写命令用来启动对一个已经激活行的突发写入操作,BA0 和 BA1 指定需要写入的 BANK, 地址线上某些位指定需要写入的数据起始列地址 (A0–A9, A11 (x4), A0–A9 (x8), or A0–A8 (x16)), A10 控制是否在突发写完成之后立即执行预充电, 即关闭当前行操作。若为高电平则在写完当前行以后立即自动执行预充电操作 (关闭当前行), 若为低电平则不关闭该行, 使其继续处于激活状态, 以便于紧接着对该行执行新的读写操作。DQM 控制该位对应的字节是否被写入新的数据 (例如, 16 位的 SDRAM 芯片, DQM0 对应控制 DQ[7:0], DQM1 对应控制 DQ[15:8]), 若 DQM 为高电平, 则数据总线上对应字节的数据不会被写入到 SDRAM 中。





预充电/关闭行（PRECHARGE）

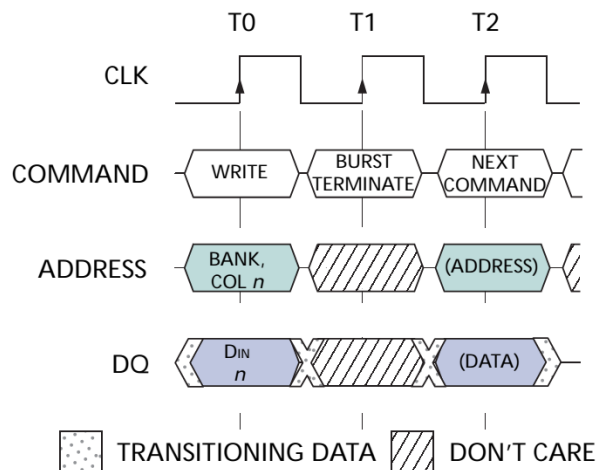
控制信号组成：	{CS_N, RAS_N, CAS_N, WE} = 4'b0010;
命令说明：	预充电命令的作用就是关闭在指定 BANK 或者全部 BANK（具体是单一或者全部 BANK 由 A10 状态选择）中已经打开的行，在 PRECHARGE 命令执行并经过 tRP 时间后，对应的 BANK 将可以重新被操作，即从执行 PRECHARGE 命令到下一次能够对该 BANK 执行新的命令，需要等待 tRP 的时间。A10 为高电平，则对所有 BANK 行进行预充电，若 A10 为低电平，则只对 BA0 和 BA1 指定的 BANK 中的行进行预充电。当某个 BANK 行被执行了预充电命令后，该 BANK 将处于空闲状态，在下次读写操作之前必须重新激活。

自动预充电（Auto Precharge）

控制信号组成：	
命令说明：	自动预充电指在不额外增加执行指令的前提下，达到使用预充电指令一样的效果。该功能是在对 SDRAM 发出读写命令时，使用 A10 指明是否在突发读写完成后立即自动执行预充电操作来实现的。自动预充电命令在突发长度为整页突发时无效的。自动预充电命令每次发出只有效一次，即如果在下次执行读写时，希望继续使用自动预充电方式，则需要使用 A10 重新指明。

突发终止（BURST TERMINATE）

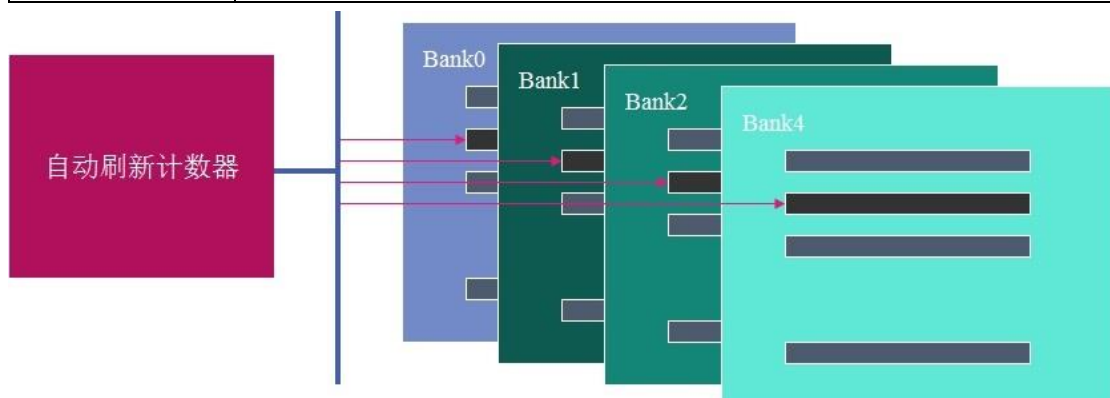
控制信号组成：	{CS_N, RAS_N, CAS_N, WE} = 4'b0110;
命令说明：	突发中断命令用来截断固定长度或者整页长度的突发。当突发中断命令被执行时，离该命令最近的一次被 SDRAM 寄存的读或者写命令被截断，突发中断命令并不会对对应行进行预充电，即执行了突发中断操作后，该行任然处于被激活状态，除非预充电命令被执行，该 BANK 才会被关闭。下图为中断一个写突发过程的时序图：



Note: DQM is LOW.

自动刷新 (AUTO REFRESH)

控制信号组成:	$\{CS_N, RAS_N, CAS_N, WE\} = 4'b0001$
命令说明:	自动刷新一般用于对 SDRAM 进行常规操作的过程中。该命令是非持续性的，即每次需要时，都需要发送此命令。在执行自动刷新命令之前，所有的 BANK 必须被预充电（关闭）。在自动刷新和预充电命令之间，必须间隔最起码的 t_{RP} 时间。自动刷新时，行地址由 SDRAM 芯片内部的刷新控制器自动更新，因此，在执行自动刷新命令时，用户不需要关心地址总线上的值。不管器件的数据位宽是多少，256Mb 的 SDRAM，商业和工业级的器件，在每 64ms 内需要执行 8192 次自动刷新操作，即每 7.813us 执行一次。或者汽车级的器件，每 16ms 内执行 8192 次自动刷新操作，即每 1.953us 执行一次。 另外自动刷新操作也可以采用突发的方式执行，即每 64ms（16ms 汽车级）都连续不间断的执行 8192 次自动刷新命令，然后其他时间再执行读写或者其他命令。



自刷新（SELF REFRESH）

自刷新命令能够用来保持 SDRAM 中的数据，即使系统中其他部分已经掉电。在自刷新模式下，SDRAM 不需要外部的时钟信号就能够保持数据。

发起自刷新命令和发起自动刷新非常类似，只是自刷新命令的 CKE 是被置为了低电平（自动刷新命令的 CKE 为高电平）。当自刷新命令被寄存在 SDRAM 中去后，SDRAM 除 CKE 以外的所有管脚状态都将被忽略，CKE 信号继续有作用，且只能为低电平。

当自刷新模式开启后，SDRAM 内部自己产生时钟，用来作为自动刷新的周期。SDRAM 进入自刷新模式后，最少需要保持在自刷新模式下 tRAS 的时间，当然，对于运行在自刷新模式下的最大时间并没有任何限制。

退出自刷新模式需要执行一串命令，首先，在 CKE 信号恢复到高电平之前，时钟信号 CLK 必须稳定(稳定的时钟可由对应的时钟引脚经过约束后的时钟提供)。当 CKE 信号回复到高电平后，必须保持 NOP 命令最低 2 个时钟周期，因为必须保证内部可能正在执行的刷新操作完成。

在退出自刷新模式后，就需要每 7.81us 或者更短的时间内执行一次自动刷新操作。（自刷新和自动刷新都是利用了内部的行刷新计数器。）
汽车级的器件不支持自刷新功能。

SDRAM 操作时序详解

SDRAM 上电初始化时序

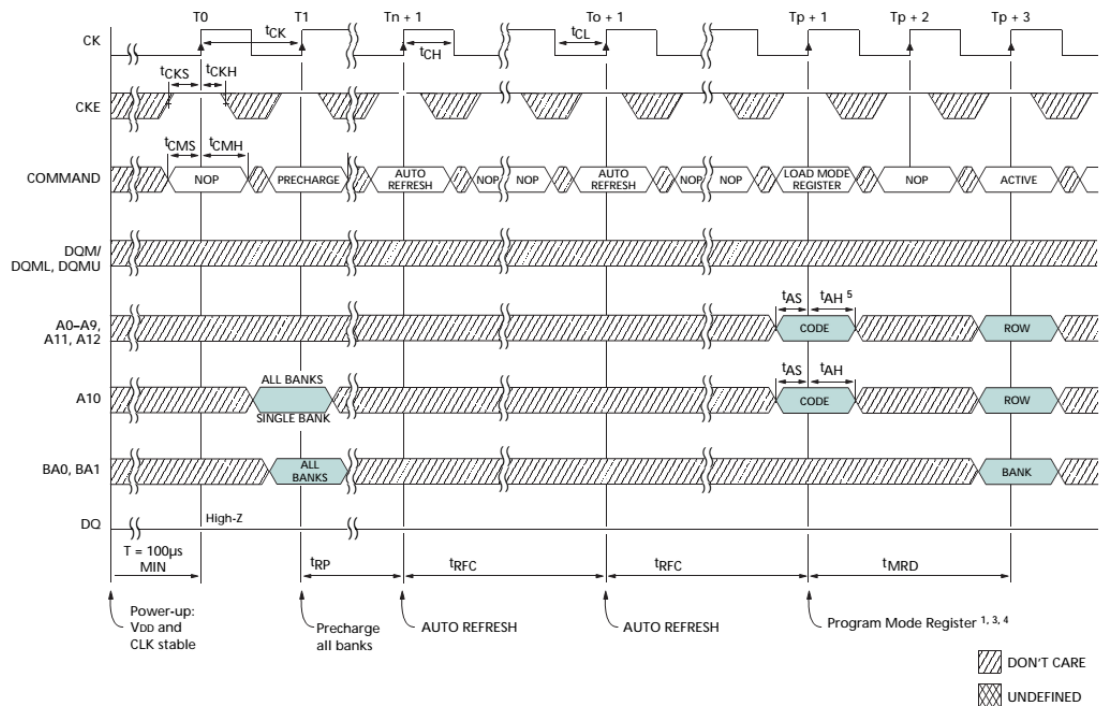
在对 SDRAM 进行正常的操作之前，SDRAM 必须被初始化。SDRAM 的上电和初始化需要按照预先定义好的方式进行，非这些指定的操作之外的命令可能导致不可预知的操作。

当 SDRAM 的 VDD 和 VDDQ 上电，并且时钟稳定后，SDRAM 首先需要延时等待 100us，在这个等待期间，对于 SDRAM 只能赋给禁止命令（INHIBIT）或者空操作（NOP）命令。

100us 的延时之后，需要对 SDRAM 首先执行一次预充电命令，所有的 BANK 都必须被预充电，以使器件所有的 BANK 都处于空闲状态。

进入空闲状态后，至少需要执行两个周期的自动刷新命令，自动刷新命令完成之后，就可以对 SDRAM 进行加载模式寄存器了，因为模式寄存器上电后处于未知的状态，因此在执行任何的操作命令之前必须加载模式寄存器，在需要的时候，在模式加载完成后可以执行两次自动刷新命令。

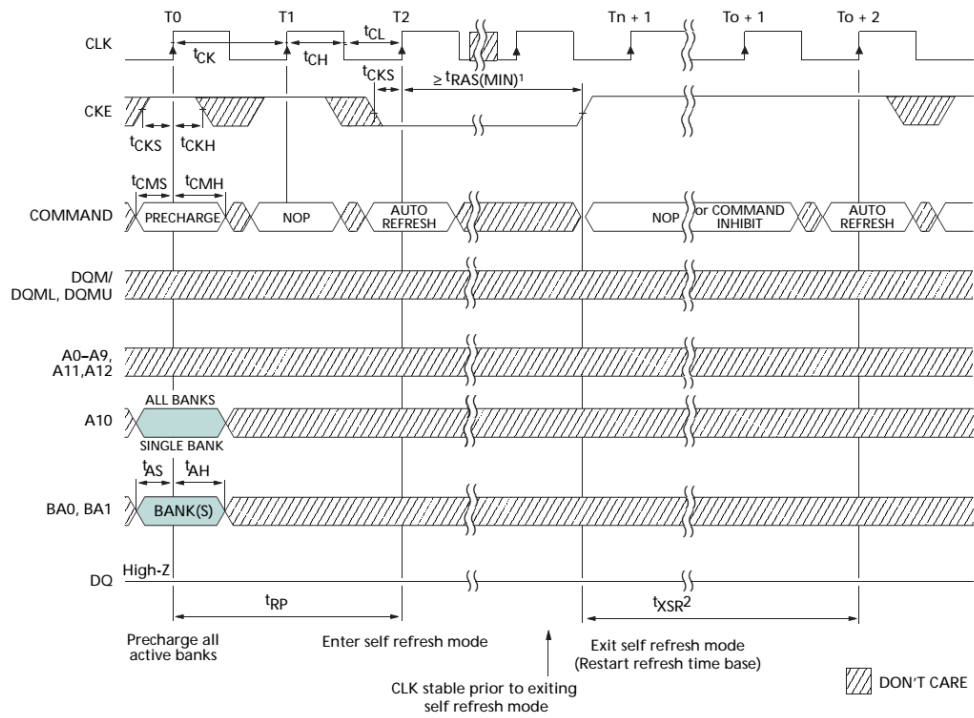
Figure 36: Initialize and Load Mode Register



1. 加载电源 (VDD 和 VDDQ)
2. CKE 设置为低 (LVTTTL 逻辑低电平)
3. 加载稳定的时钟信号
4. 等待至少 100us, 此时的命令保持为 INHIBIT 或 NOP
5. 在步骤 4 的 100us 中的某个时刻, 将 CKE 设置为高, 命令仍然保持为 INHIBIT 或 NOP
6. 步骤 4 的 100us 结束后, 即可发出一个全部 Bank 的预充电命令 (PRECHARGE ALL)
7. 等待至少 t_{RP} (行预充电最小周期), 此时命令保持为 NOP 或 DESELECT
8. 发出一个自动刷新命令 (AUTO REFRESH)
9. 等待至少 t_{RFC} (自动预充电周期), 此时的命令仅允许是 NOP 或 INHIBIT
10. 再发出一个自动刷新命令 (AUTO REFRESH)
11. 再等待至少 t_{RFC} (自动预充电周期), 此时的命令仅允许是 NOP 或 INH
12. 使用 LMR 命令设置模式寄存器
13. 等待至少 t_{MRD} (LMR 命令至激活或刷新命令的最小间隔), 此时的命令仅允许是 NOP 或 DESELECT

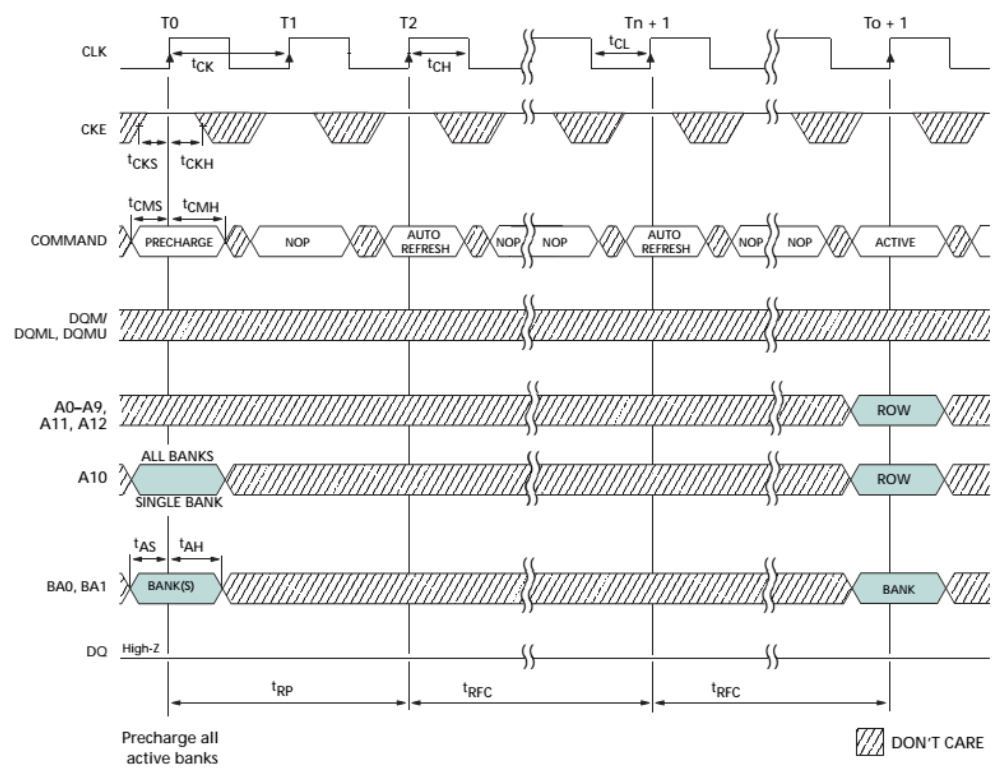
SDRAM 自刷新时序

Self Refresh Mode



SDRAM 自动刷新时序

Auto Refresh Mode



Notes: 1. t_{RFC} must not be interrupted by any executable command; COMMAND INHIBIT or NOP must be applied on each positive edge during t_{RFC} .

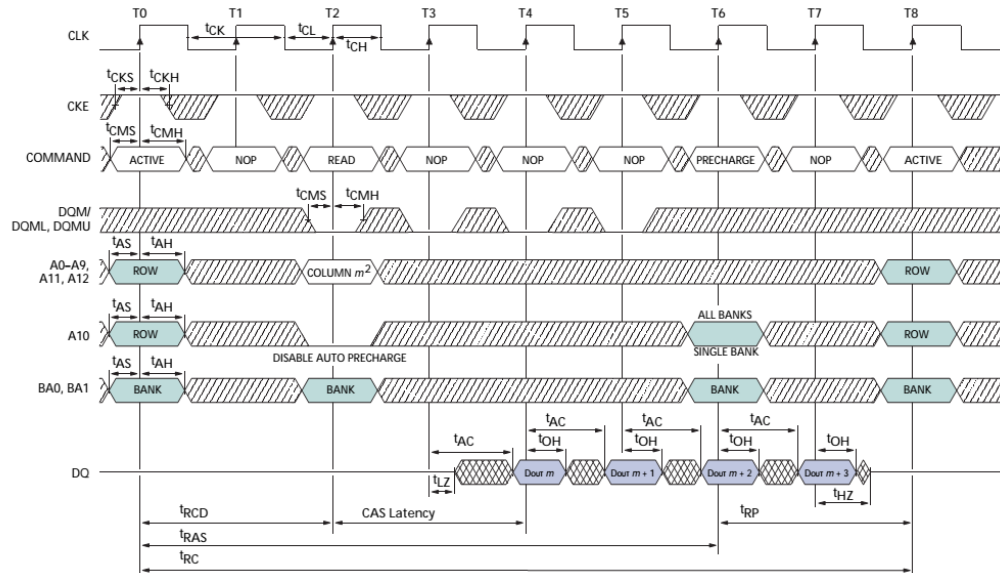
SDRAM 自刷新模式 VS 自动刷新模式

	自动刷新模式	自刷新模式
作用	保证数据不丢失	保证数据不丢失
是否需要控制器控制刷新	需要	不需要
使用场合	对 SDRAM 正常操作过程	掉电/低功耗模式
是否需要外部时钟信号	需要	不需要
刷新行地址自动控制	是	是

- 1. SDRAM 在进入和退出自刷新模式时的过程需要一系列的指令配合实现。
- 2. SDRAM 进入自刷新模式有助于减小系统待机功耗

不带自动预充电读操作时序

Figure 41: Read – Without Auto Precharge



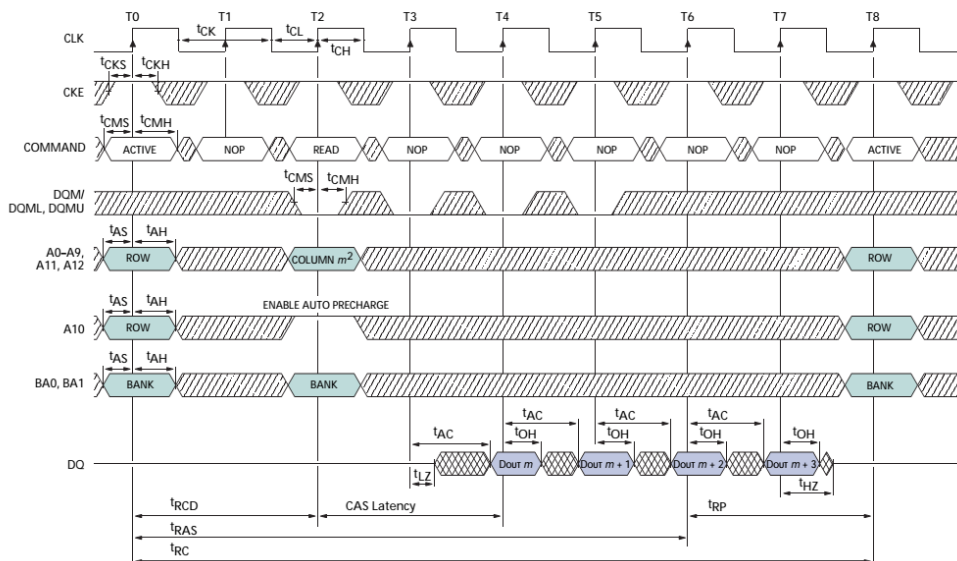
1、BL = 4

2、CL = 2

1. For this example, BL = 4, CL = 2, and the READ burst is followed by a "manual" PRECHARGE.
2. x16: A9, A11, and A12 = "Don't Care"
x8: A11 and A12 = "Don't Care"
x4: A12 = "Don't Care"

带自动预充电读操作时序

Figure 42: Read – With Auto Precharge

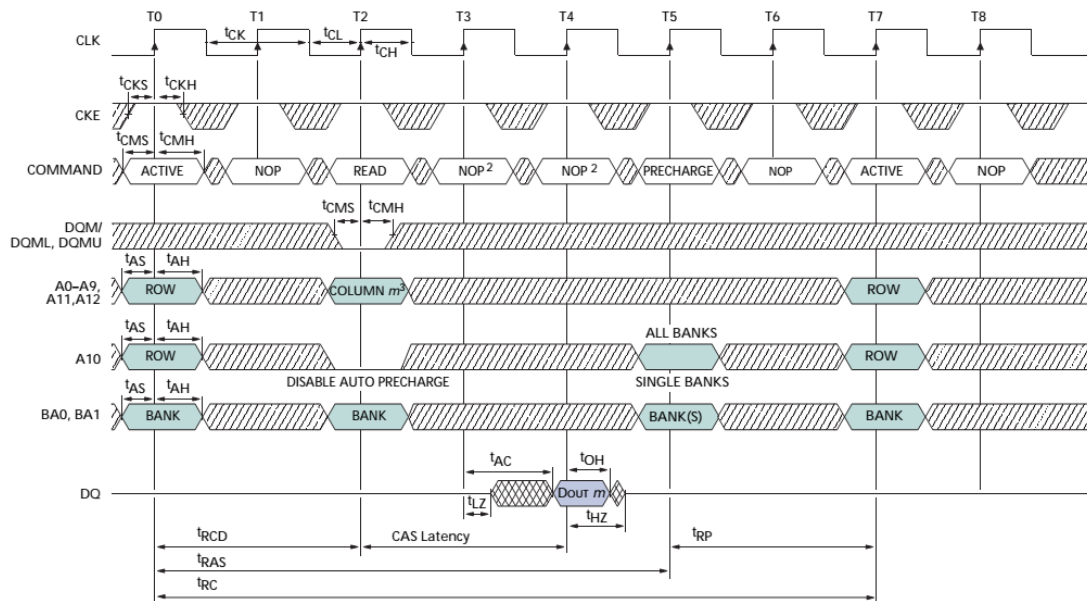


1、BL = 4

2、CL = 2

不带自动预充电单一位置读

Figure 43: Single Read - Without Auto Precharge

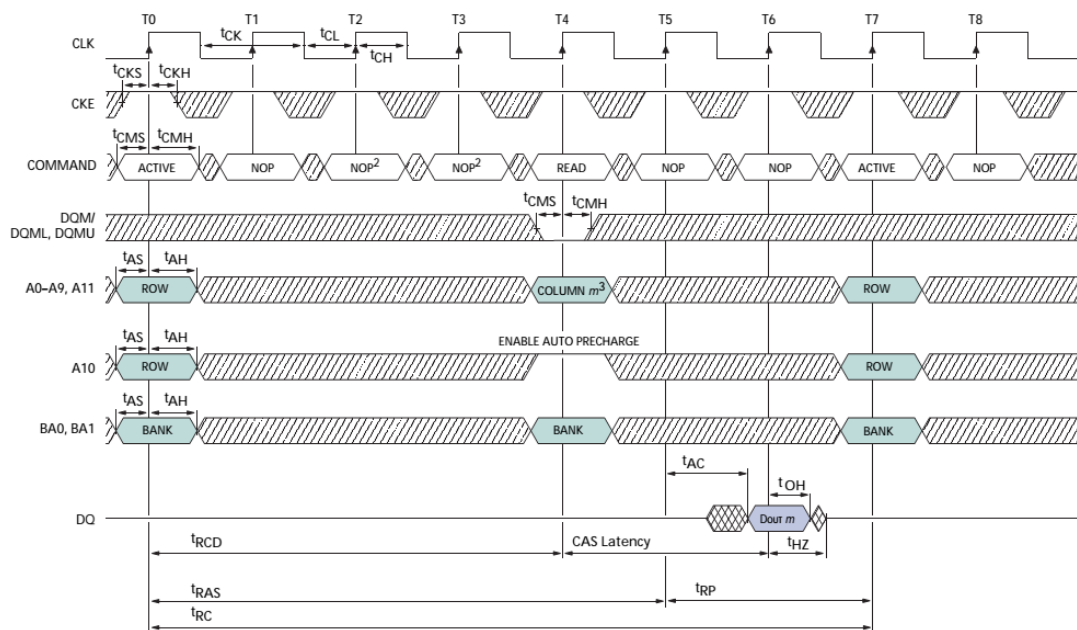


1、BL = 1

2、CL = 2

带自动预充电单一位置读

Figure 44: Single Read - With Auto Precharge

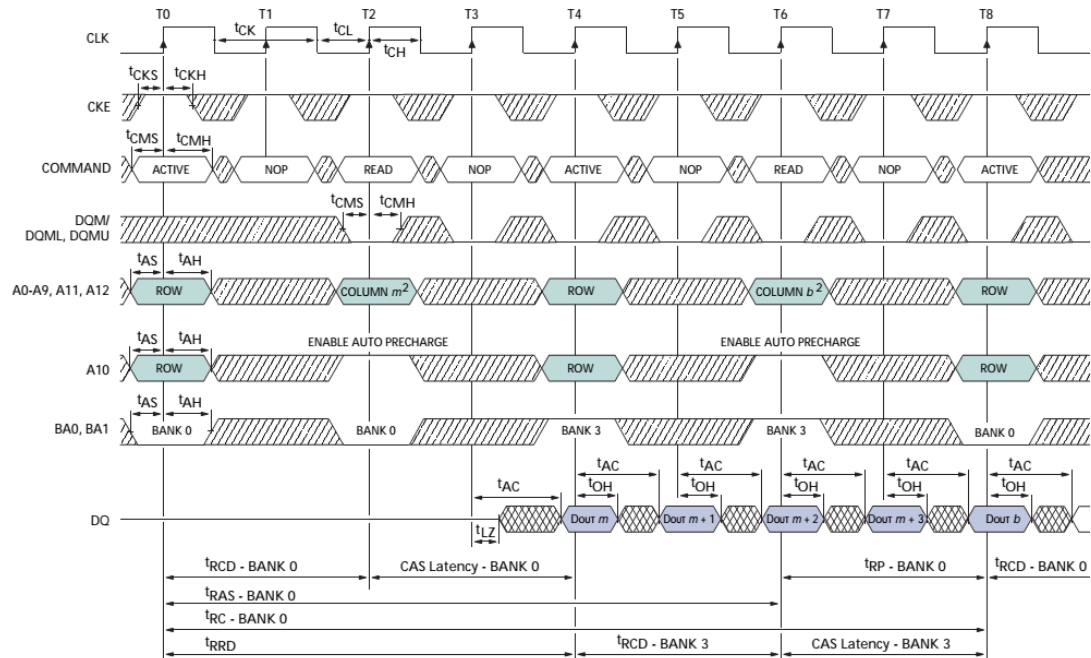


1、BL = 1

2、CL = 2

跨 BANK 随机读操作

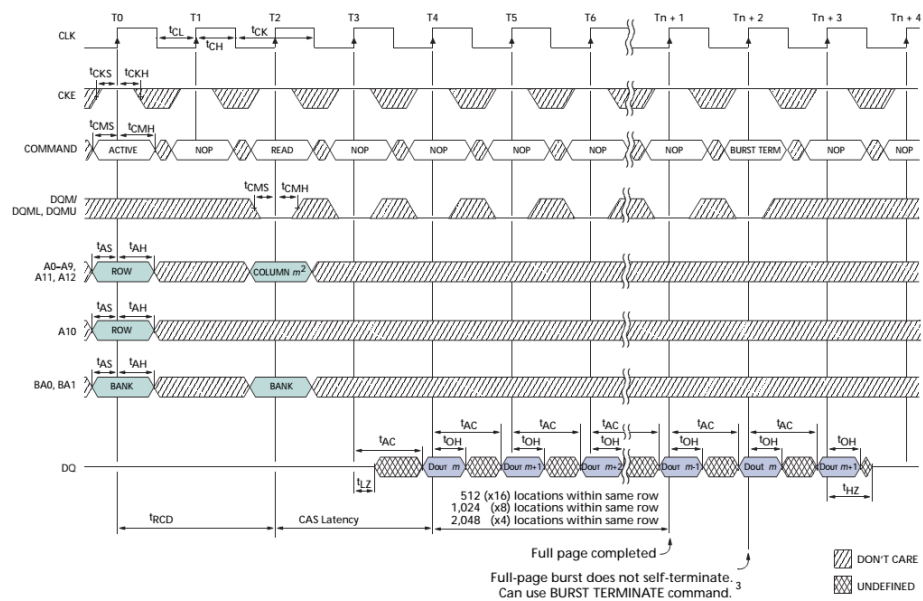
Figure 45: Alternating Bank Read Accesses



- 1、BL = 4
- 2、CL = 2

页突发读

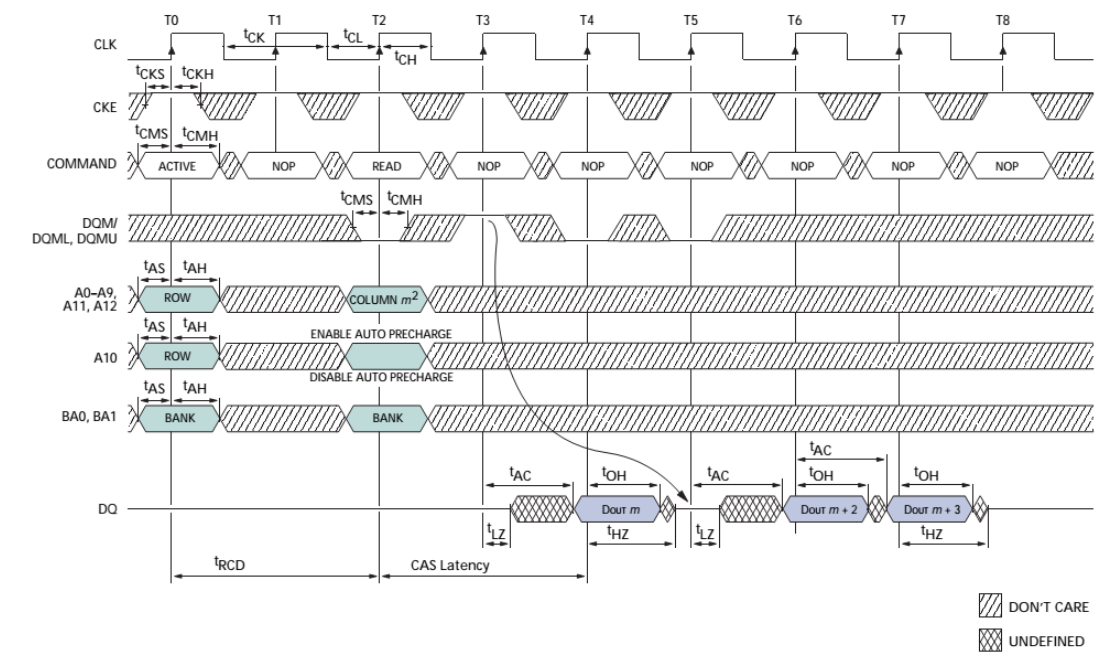
Figure 46: Read - Full-Page Burst



- 1、BL = page
- 2、CL = 2

带屏蔽的读操作

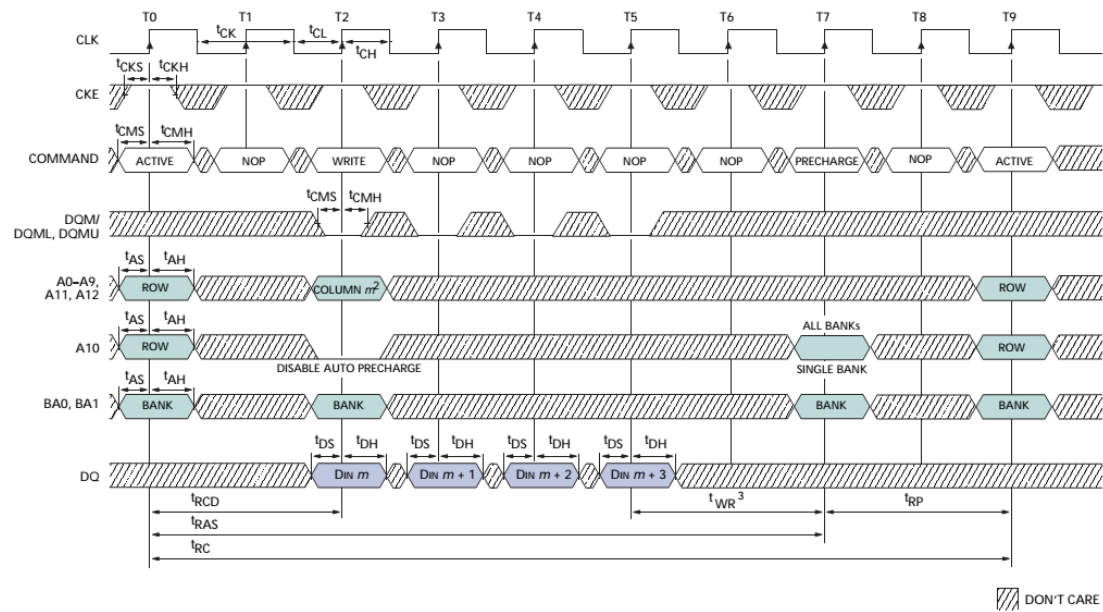
Figure 47: Read – DQM Operation



- 1、BL = 4
- 2、CL = 2

不带自动预充电写操作时序

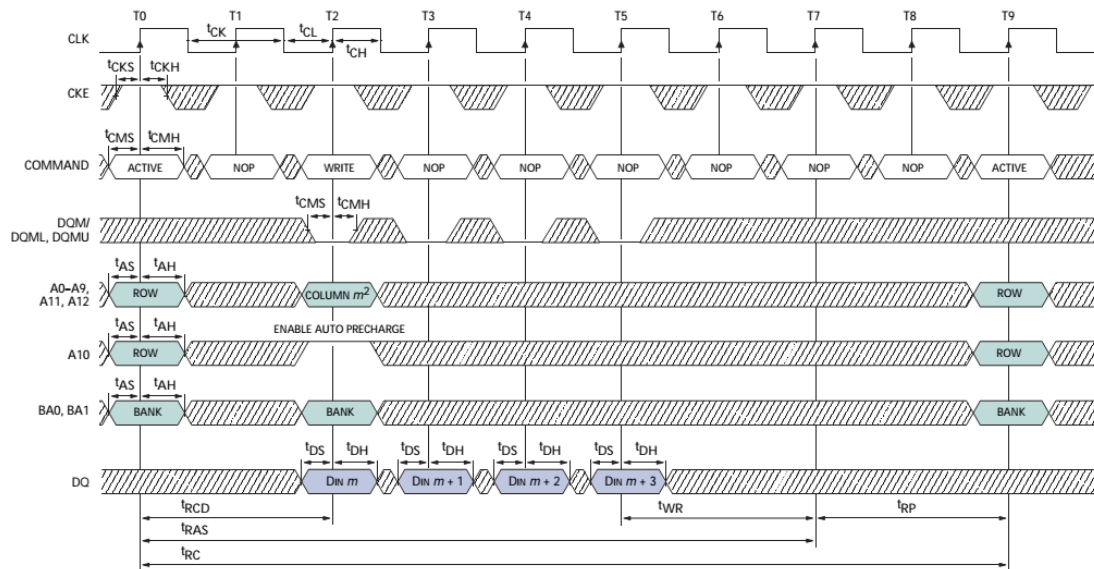
Figure 48: Write – Without Auto Precharge



- 1. BL = 4

带自动预充电写操作时序

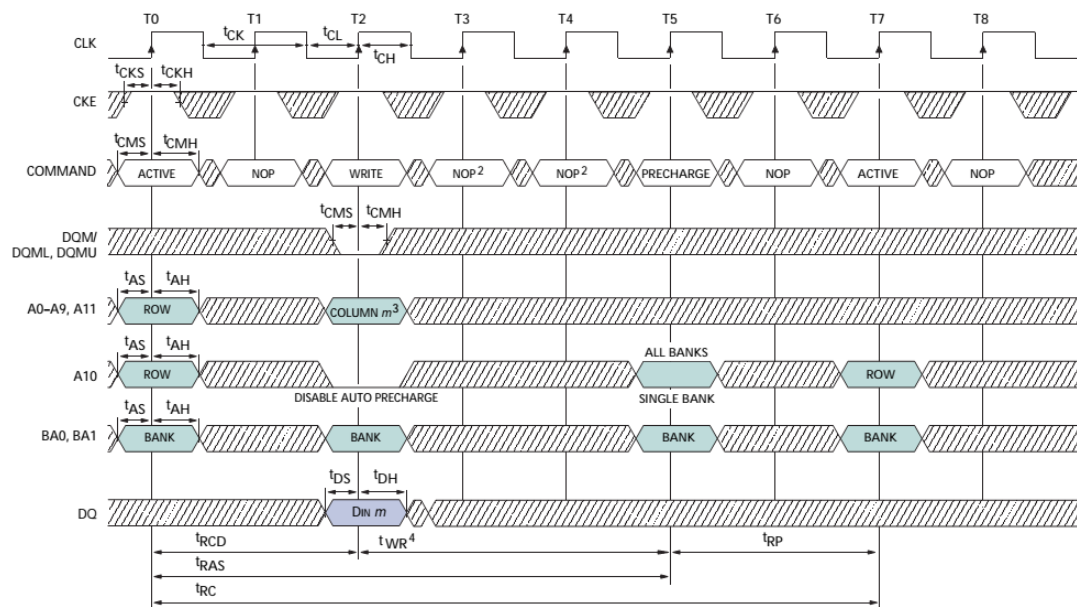
Figure 49: Write – With Auto Precharge



1. BL = 4

不带预充电的单一位置写

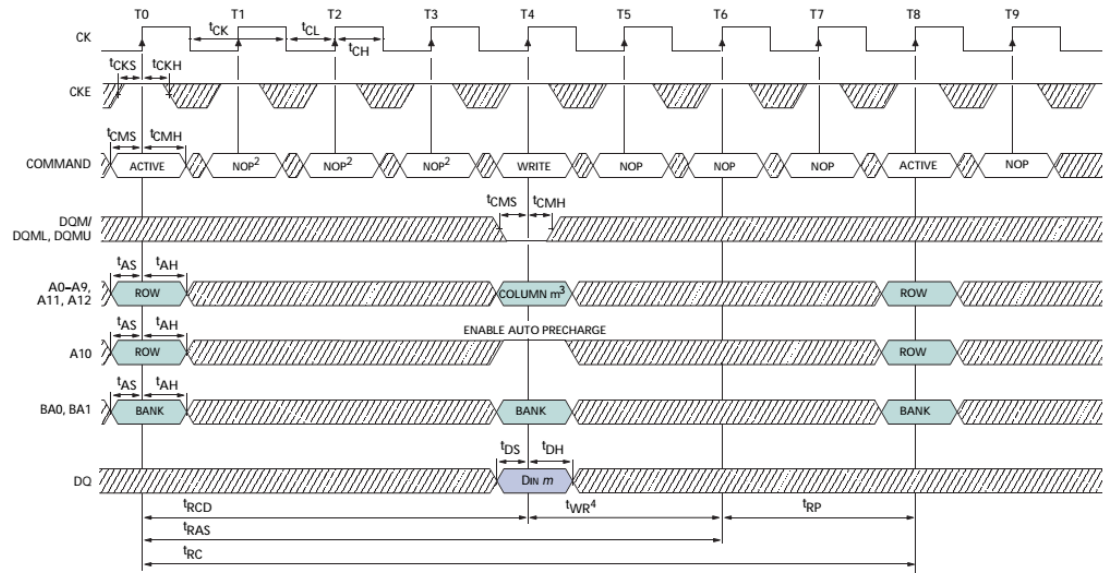
Figure 50: Single Write – Without Auto Precharge



1. BL = 1

带预充电的单一位置写

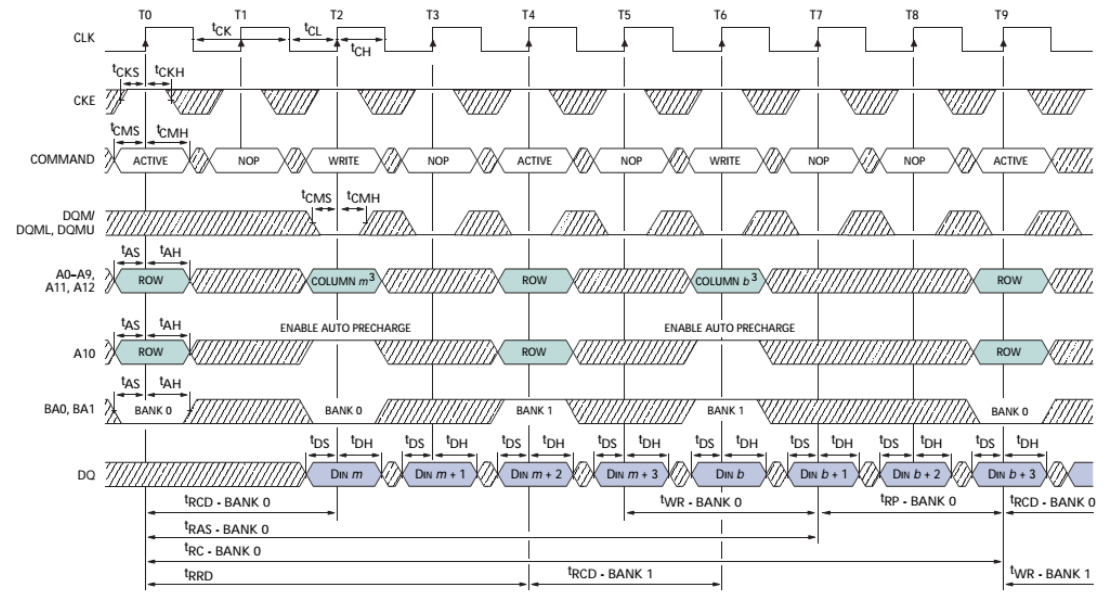
Figure 51: Single Write – With Auto Precharge



1. BL = 1

跨 BANK 写操作

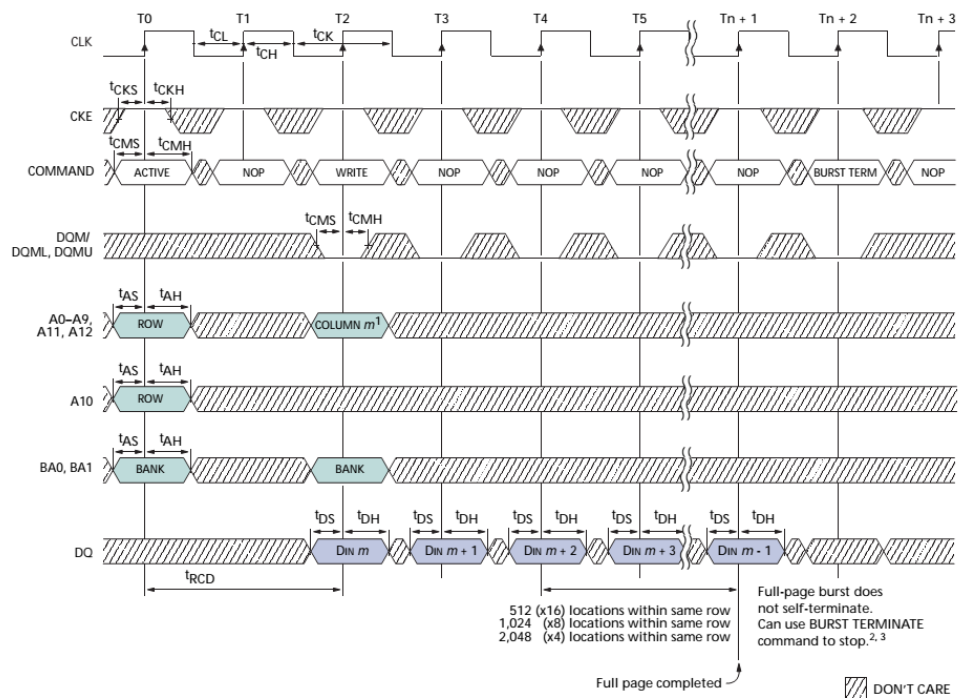
Figure 52: Alternating Bank Write Accesses



1. BL = 4

页突发写操作

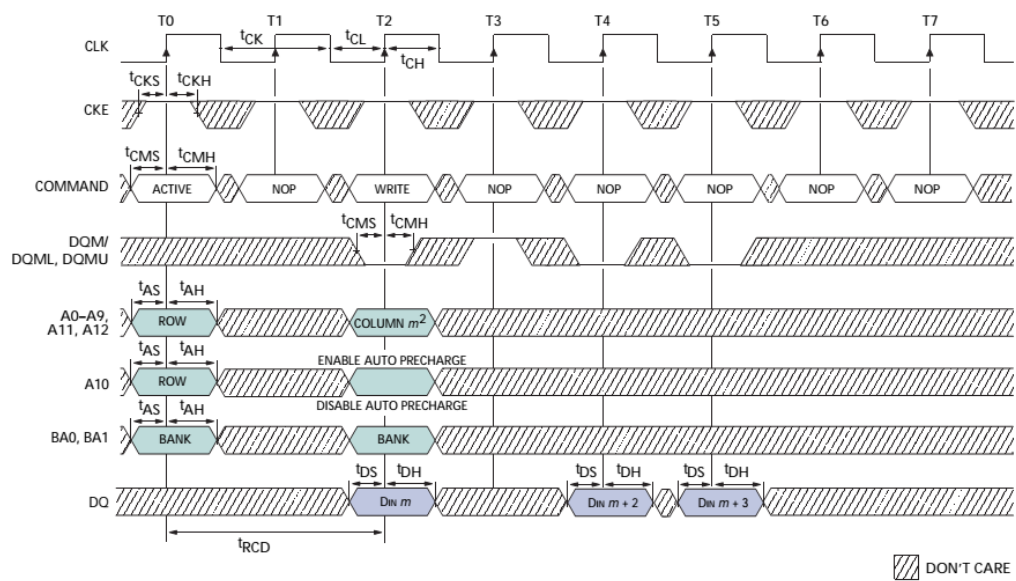
Figure 53: Write – Full-Page Burst



1. BL = page

带屏蔽的写操作

Figure 54: Write – DQM Operation

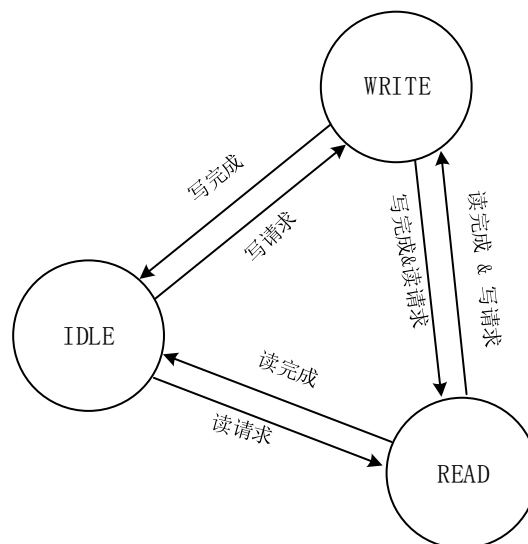


1. BL = 4

SDRAM 常见应用领域与各模式关系

视频图像缓存

由于视频图像数据量非常的大，因此在视频图像的处理系统中，经常使用 SDRAM 作为视频图像数据的缓存。而视频图像数据流一般都是顺序产生的，同时在输出时，也只需要顺序输出即可，因此，我们可以将图像数据的缓存看成一个大容量的 fifo，该 fifo 一侧实现数据的顺序输入，一侧实现数据的顺序输出。由于是顺序输入输出，因此对 SDRAM 的读写都是按照地址顺序执行的，因此我们就可以考虑针对此特性设计一个高效的图像缓存控制器，该控制器控制图像数据依次写入 SDRAM 的每一行，这样就可以每次对一行数据进行写入只需要激活一次激活命令，然后持续写满该行后再关闭该行，从而避免频繁的打开和关闭行造成的效率损失。当一行写完后，再开始一行读的操作。同样，读也只需每次先激活一行，然后按照地址顺序持续的读出该行。所以，针对大流量的视频图像数据，可以设计一个非常简单的数据流控制状态机，充分利用 SDRAM 对同一行连续进行读操作或进行写操作时只需要激活一次的特点，提高读写效率，该状态机可以简化如下所示。



整个状态机包含三个大的状态，分别为空闲状态（IDLE）、写 SDRAM 状态（WRITE）、读 SDRAM 状态（READ）。

IDLE 状态为空闲状态，即由于数据的写入和读取所需总带宽小于 SDRAM 的有效总带宽，因此必然有一段时间 SDRAM 处于无读写操作状态，此时，状态机就停留在 IDLE 状态，但是 IDLE 状态我们也并不是什么都不做，我们可以把最头疼的事儿——刷新，放在该状态来完成，即在 IDLE 状态的时候，我们不断的持续刷新 SDRAM 中的数据，这样就可以避免在对 SDRAM 进行读写操作的过程中出现刷新请求，而导致当前读写过程中断，影响 SDRAM 工作效率。即在 SDRAM 处于空闲的时候，把需要周期性进行的刷新操作提前执行一些。

WRITE 状态为写 SDRAM 状态，在此过程中，完成对一行或一行中的某一段数据的持续写入，也可以完成对多个 Bank 中相同行的写操作。此状态专注于对 SDRAM 写操作。当写入完成后，判断有无读请求，如果有读请求，跳往 READ 状态进行读操作，否则跳转回 IDLE 状态。

READ 状态为读 SDRAM 状态，在此过程中，完成对一行或一行中的某一段数据的持续读

取，也可以完成对多个 Bank 中相同行的读操作。此状态专注于对 SDRAM 读操作。当读取完成后，判断有无写请求，如果有写请求，跳往 WRITE 状态进行写操作，否则跳转回 IDLE 状态。

此种设计没有对 SDRAM 采用定时刷新方式，而是采用了空闲刷新的方式，相对于定时刷新方式可能会造成正在进行的读写操作中中断，空闲刷新能够保证读写过程的连续性，因此从一定程度上能够提高 SDRAM 的读写效率。而空闲刷新，只要保证每 64ms 内能够完成对所有 BANK 中的所有行（如 HY57V281620 需要在 64ms 内完成 4096 次刷新操作）实现一次刷新，就不会出现数据丢失。

针对这种应用，可以使用页读写的方式进一步提高 SDRAM 的读写效率，并简化读写控制逻辑。

为了保证在开始写入数据后能够有足够的被写入 SDRAM，因此可以采用缓存 fifo 实现对待写入数据的缓存，只有当 fifo 中的数据大于等于一次写入所需的数据长度后，才开始执行一次写入，从而避免待写入数据不足而造成写入提前中断或写入错误数据的情况。而使用一个深度是两倍于单次写入长度的 fifo，还能够避免在读取 SDRAM 的过程中，由于写入侧 fifo 没有被及时读取导致待写入数据可能存在的溢出情况。

同样，对于读取也是一样，因为一次性需要读出固定的长度，为了保证数据使用侧能够不断流，可以使用一个 fifo 将 SDRAM 中读取的数据实时保存起来，这样数据使用侧只需要实时从 fifo 中读取数据即可，只要保证 fifo 中一直有数据，数据使用侧就不会出现断流。而使用一个两倍于读出长度的 fifo，能够避免在写入 fifo 的时候，待写入数据长度大于 fifo 能够接收的最大数据长度而出现的数据丢失。

总结：此种应用中，根据写入和读取数据地址连续自增的特点，采用了整行读写的方式，减少了对行的激活和关闭需要的命令开销，从而提高了 SDRAM 工作效率。同时，使用 SDRAM 的页突发模式，简化了读写控制逻辑。

嵌入式系统随机内存