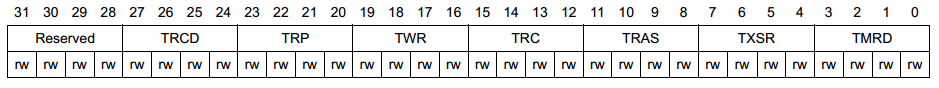
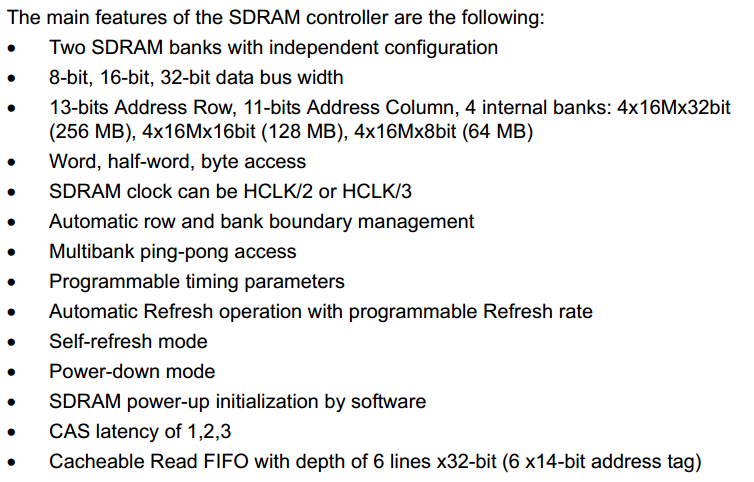
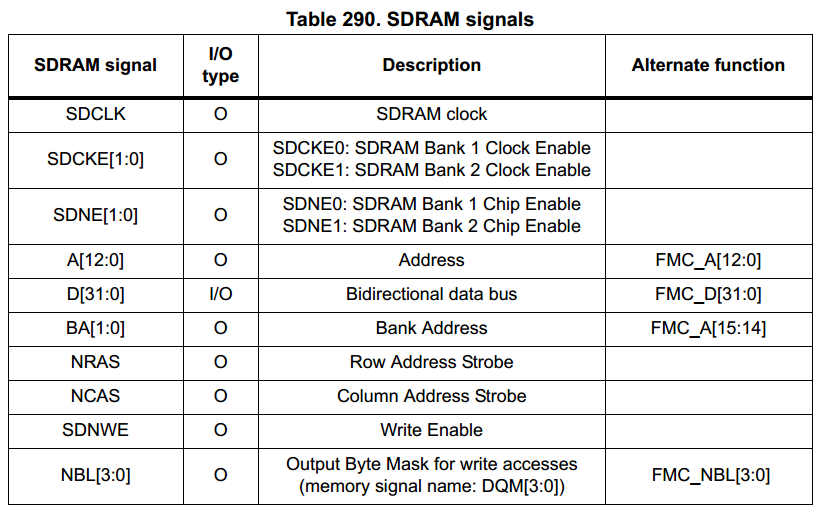
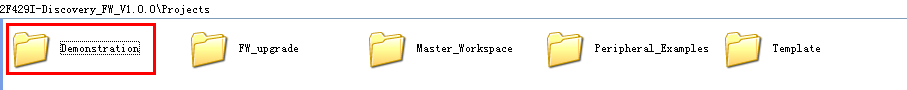
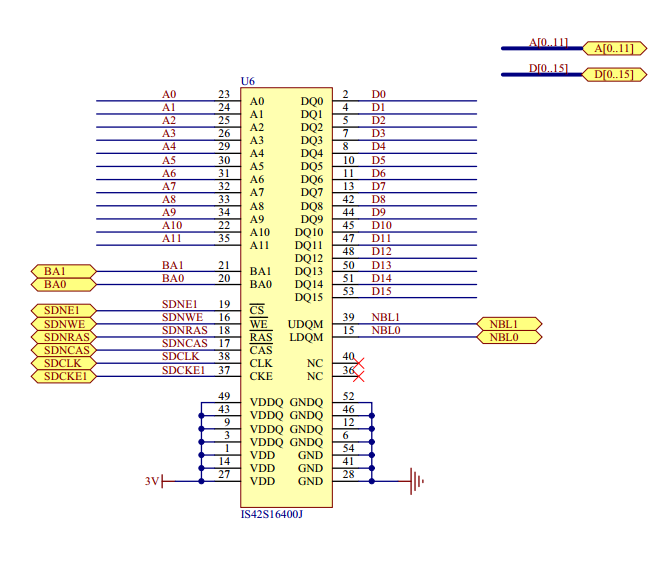
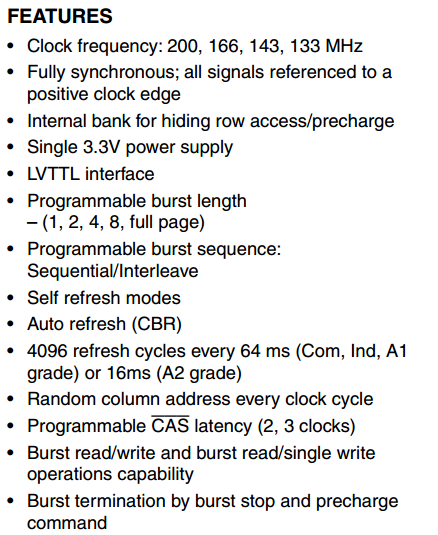
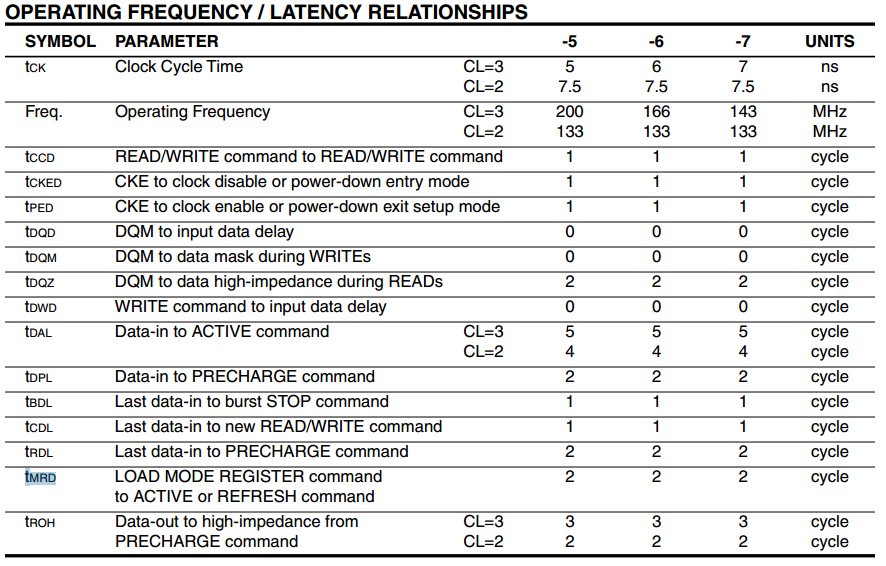
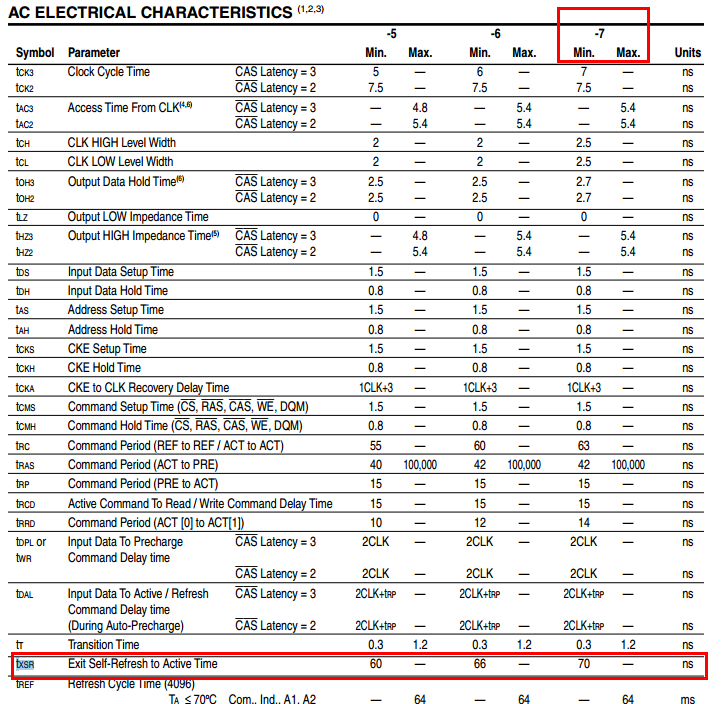
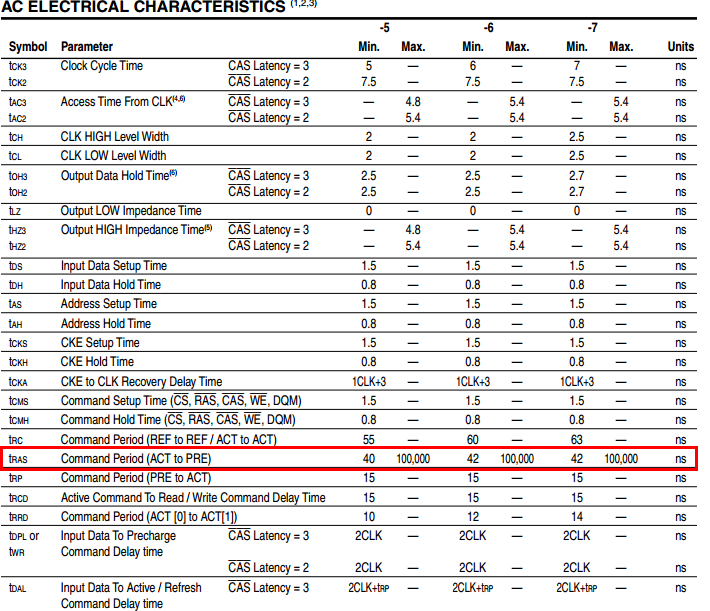
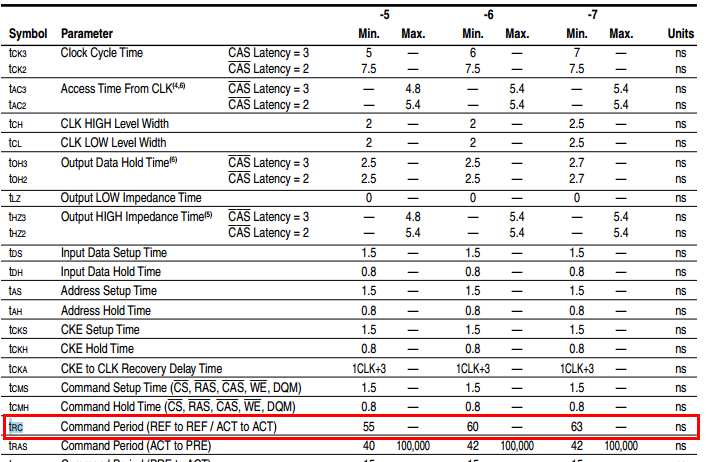
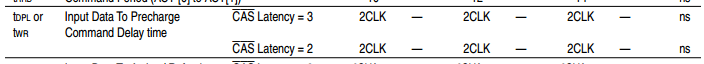
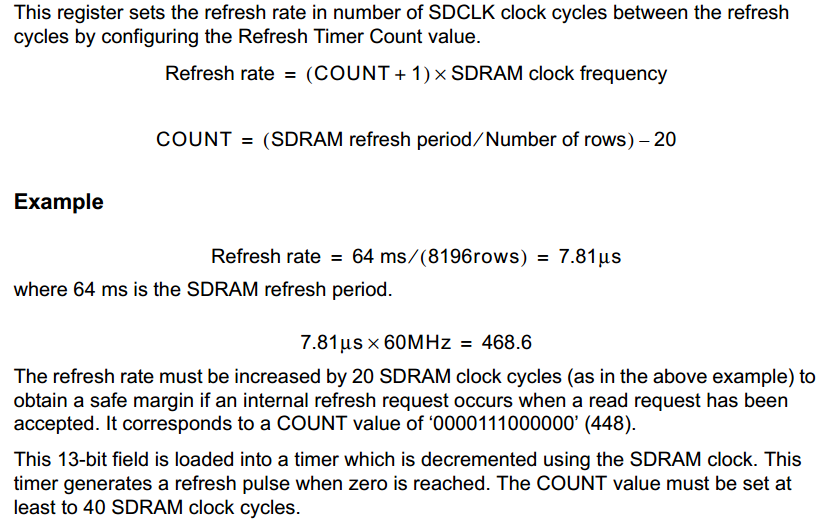
<http://blog.csdn.net/stm32f4/article/details/17963831>

# [【STM32F429-DISCOVERY学习笔记】STM32F429驱动SDRAM(IS42S16400J)详解](http://blog.csdn.net/stm32f4/article/details/17963831)

2014-01-07 17:31 1499人阅读 [评论](http://blog.csdn.net/stm32f4/article/details/17963831#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/stm32f4/article/details/17963831#report)

 http://bbs.armfly.com/read.php?tid=1942

 驱动SDRAM的时序比较的麻烦一些，不像驱动SRAM，非常简单，网上搜索一下，估计有非常多的FPGA驱

动SDRAM的资料，而且是各种的给你讲时序问题，现在F429/439集成了控制器以后就方便很多了，用户只需配  
置相应的寄存器即可，这里向大家推荐一篇文章，强烈的推荐，不懂SDRAM为何物的，一定要看看。  
《高手进阶，终极内存技术指南——完整/进阶版》 <http://bbs.armfly.com/read.php?tid=1930>   
**1.  学习SDRAM驱动前的准备工作**  
    学习SDRAM前搞清楚两个问题，一个是SDRAM的基本原理，还有一个就是那几个关键的参数，参数是F429  
配置SDRAM的关键，这几个参数大概知道是什么意思就行了，配置的时候，根据SDRAM的手册配置一下就OK  
了。在STM32F429/439的数据手册里关键参数说明，F429/439是把这几个关键的参数做到了一个寄存器里面  
了，这些参数，手册上面有一些英文说明，但比较的笼统。  
  
我推荐的那篇文章，建议大家一定要看，别的可以不看，这个必须得看，讲的实在太好了，我这里把一些关键的  
参数摘录出来：  
**tRCD:**  
       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PC100SDRAM，tRCD=2，代表20ns的延迟，对于PC133则为15ns  
**CL（CAS Latency）：**  
        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。  
       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）。  
**tWR：**  
        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，WriteRecovery Time），这个操作也被称作写回（Write Back）。  
**tRP：**  
        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。  
我这里就先贴上这几个参数，其它的参数，大家可以查阅相关的资料。  
**2.  F429/439手册中对SDRAM的介绍**  
       如果你看了我在前面推荐的那个文章，现在看这个手册还是比较容易的。我这里就象征性的贴一下F429  
439自带SDRAM控制器的特性和引脚，其它的大家自己看手册就行。  
  
  
  
**3.  F429/439驱动SDRAM详细过程**  
说明：以官方的Demonstration做例子  
  
  
**第一步：F429的时钟配置**  
        F429/439支持最高主频180Mhz，相比于407只提高了12M, 感觉劲还是不足，传说STM32F5系列将  
做到250MHz，就相当的给力了。  
F429配置过程和以前芯片的配置过程是一样的，我这里主要是想通过这个配置得到SDRAM的时钟频率， 先  
说明一下这个开发板外接的是8MHz的晶振，通过锁相环得到180的主频。  
  
#define PLL\_SOURCE\_HSE    // HSE (8MHz) used to clock the PLL, and the PLL is used as system clock source  
/\*  PLL\_VCO = (HSE\_VALUE or HSI\_VALUE / PLL\_M) \* PLL\_N  \*/  
#if defined  (PLL\_SOURCE\_HSI)  
#define PLL\_M      16  
#else  
#define PLL\_M      8  
#endif  
#define PLL\_N      360  
/\* SYSCLK = PLL\_VCO / PLL\_P \*/    
#define PLL\_P      2  
/\* USB OTG FS, SDIO and RNG Clock =  PLL\_VCO / PLLQ \*/  
#define PLL\_Q      7  
由上面的公式得到 :SYSCLK = PLL\_VCO / PLL\_P =( (HSE\_VALUE or HSI\_VALUE / PLL\_M) \* PLL\_N)/PLL\_P  = 180  
/\* HCLK = SYSCLK / 1 = 180 \*/RCC->CFGR |= RCC\_CFGR\_HPRE\_DIV1;     
/\* PCLK2 = HCLK / 2 = 90 \*/RCC->CFGR |= RCC\_CFGR\_PPRE2\_DIV2;    
/\* PCLK1 = HCLK / 4 = 45 \*/RCC->CFGR |= RCC\_CFGR\_PPRE1\_DIV4;  
第二步：引脚配置  
        驱动前要先搞清楚一个问题，驱动SDRAM的行选和列选的地址线是分时复用的，和SRAM不同  
需要完整的地址线才可以访问各个地址空间，官方提供的原理图  
  
  
关于这个芯片的特点：  
  
  
标准的SDRAM一般都是4个BANK，这个芯片也不例外，芯片的总容量  
1Mbitx 16-bit x 4-bank = 67,108,864 bits = 64-Mbit ，每个BANK的组成  
4096rows x 256 columns x 16 bits, 这个比较重要，配置的时候要用到，也就是  
12行8列。  
第三步：FMC配置  
这里FMC驱动SDRAM只支持两种频率，分别是  
SDCLK period = 2 x HCLK periods  
SDCLK period = 3 x HCLK periods  
根据我们上面的配置，HCLK是180MHz，这里SDCLK就是2分频或者3分频，官方提供的  
这个例子，在注释上面有误，他们是按照主频168MHz，2分频是SDCLK = 84MHz注释的，  
不过不影响使用，只是注释上面是这样的。按照90MHz的话，每个时钟周期就是11.1ns  
7个SDRAM关键参数的配置，下面一个一个的说。  
1.  TMRD  
/\* TMRD: 2 Clock cycles \*/  
FMC\_SDRAMTimingInitStructure.FMC\_LoadToActiveDelay = 2;   
这里为什么成2，查手册, 手册上提供的是三种速度等级时提供的参数，我们这里是用  
的90MHz，也取2个肯定是没问题的。  
  
  
**2. TXSR**  
/\* TXSR: min=70ns (6x11.10ns) \*/  
FMC\_SDRAMTimingInitStructure.FMC\_ExitSelfRefreshDelay = 7;  
开发板上面用的SDRAM速度等级的7，最高工作频率时143MHz  
  
  
3. TRAS  
/\* TRAS: min=42ns (4x11.10ns) max=120k (ns) \*/  
FMC\_SDRAMTimingInitStructure.FMC\_SelfRefreshTime = 4;  
  
  
4. TRC  
/\* TRC:  min=63 (6x11.10ns) \*/          
FMC\_SDRAMTimingInitStructure.FMC\_RowCycleDelay = 7;      
  
  
**5. TWR**  
/\* TWR:  2 Clock cycles \*/  
FMC\_SDRAMTimingInitStructure.FMC\_WriteRecoveryTime = 2;    
  
6. TRP  
/\* TRP:  15ns => 2x11.10ns \*/  
FMC\_SDRAMTimingInitStructure.FMC\_RPDelay = 2;   
http://bbs.armfly.com/attachment/Fid_25/25_58_da3136e17c6d9b5.png?5  
  
7. TRCD           
/\* TRCD: 15ns => 2x11.10ns \*/  
FMC\_SDRAMTimingInitStructure.FMC\_RCDDelay = 2;  
http://bbs.armfly.com/attachment/Fid_25/25_58_916622fdd83fb64.png?7  
  
时序设置好以后就是SDRAM控制器的配置。  
  
/\* 支持两个SDRAM的BANK，这里使用的是bank2  \*/  
  FMC\_SDRAMInitStructure.FMC\_Bank = FMC\_Bank2\_SDRAM;  
  /\* 根据这个SDRAM的特性是12行，8列，在这里配置一下 \*/  
  FMC\_SDRAMInitStructure.FMC\_ColumnBitsNumber = FMC\_ColumnBits\_Number\_8b;  
  FMC\_SDRAMInitStructure.FMC\_RowBitsNumber = FMC\_RowBits\_Number\_12b;  
  /\* 数据位宽是16 \*/  
  FMC\_SDRAMInitStructure.FMC\_SDMemoryDataWidth = SDRAM\_MEMORY\_WIDTH;  
  /\* 此芯片支持4个bank \*/  
  FMC\_SDRAMInitStructure.FMC\_InternalBankNumber = FMC\_InternalBank\_Number\_4;  
  /\* 设置CAS的延迟是3 \*/  
  FMC\_SDRAMInitStructure.FMC\_CASLatency = SDRAM\_CAS\_LATENCY;   
  FMC\_SDRAMInitStructure.FMC\_WriteProtection = FMC\_Write\_Protection\_Disable;  
  /\* 配置SDCLK的时钟频率 \*/  
  FMC\_SDRAMInitStructure.FMC\_SDClockPeriod = SDCLOCK\_PERIOD;    
  /\* 禁止读取时的突发模式 \*/  
  FMC\_SDRAMInitStructure.FMC\_ReadBurst = SDRAM\_READBURST;  
  /\* 设置ReadPipe时的延迟  \*/  
  FMC\_SDRAMInitStructure.FMC\_ReadPipeDelay = FMC\_ReadPipe\_Delay\_1;  
  FMC\_SDRAMInitStructure.FMC\_SDRAMTimingStruct = &FMC\_SDRAMTimingInitStructure;  
**第四步：SDRAM初始化**  
1.  使能时钟  
  FMC\_SDRAMCommandStructure.FMC\_CommandMode = FMC\_Command\_Mode\_CLK\_Enabled;    
  FMC\_SDRAMCommandStructure.FMC\_CommandTarget = FMC\_Command\_Target\_bank2;  
  FMC\_SDRAMCommandStructure.FMC\_AutoRefreshNumber = 1;  FMC\_SDRAMCommandStructure.FMC\_ModeRegisterDefinition = 0;  
  /\* Wait until the SDRAM controller is ready \*/     
  while(FMC\_GetFlagStatus(FMC\_Bank2\_SDRAM, FMC\_FLAG\_Busy) != RESET)  
  {  }  
  /\* Send the command \*/    
  FMC\_SDRAMCmdConfig(&FMC\_SDRAMCommandStructure);   
**2.  插入100ms的延迟**  
  \_\_Delay(10);  
**3.  预充电配置**    
  /\* Configure a PALL (precharge all) command \*/     
  FMC\_SDRAMCommandStructure.FMC\_CommandMode = FMC\_Command\_Mode\_PALL;  
  FMC\_SDRAMCommandStructure.FMC\_CommandTarget = FMC\_Command\_Target\_bank2;    
  FMC\_SDRAMCommandStructure.FMC\_AutoRefreshNumber = 1;  
  FMC\_SDRAMCommandStructure.FMC\_ModeRegisterDefinition = 0;    
  /\* Wait until the SDRAM controller is ready \*/   
  while(FMC\_GetFlagStatus(FMC\_Bank2\_SDRAM, FMC\_FLAG\_Busy) != RESET)    
  {  
  }    
  /\* Send the command \*/  
  FMC\_SDRAMCmdConfig(&FMC\_SDRAMCommandStructure);  
**4. 自动刷新配置**  
  /\* Configure a Auto-Refresh command \*/     
  FMC\_SDRAMCommandStructure.FMC\_CommandMode = FMC\_Command\_Mode\_AutoRefresh;  
  FMC\_SDRAMCommandStructure.FMC\_CommandTarget = FMC\_Command\_Target\_bank2;    
  FMC\_SDRAMCommandStructure.FMC\_AutoRefreshNumber = 4;  
  FMC\_SDRAMCommandStructure.FMC\_ModeRegisterDefinition = 0;    
  /\* Wait until the SDRAM controller is ready \*/   
  while(FMC\_GetFlagStatus(FMC\_Bank2\_SDRAM, FMC\_FLAG\_Busy) != RESET)    
  {  
  }    
  /\* Send the  first command \*/  
  FMC\_SDRAMCmdConfig(&FMC\_SDRAMCommandStructure);    
  /\* Wait until the SDRAM controller is ready \*/     
  while(FMC\_GetFlagStatus(FMC\_Bank2\_SDRAM, FMC\_FLAG\_Busy) != RESET)  
  {  }  
  /\* Send the second command \*/    
  FMC\_SDRAMCmdConfig(&FMC\_SDRAMCommandStructure);  
**5. 配置外部模式寄存器**    
  /\* Program the external memory mode register \*/  
  tmpr = (uint32\_t)SDRAM\_MODEREG\_BURST\_LENGTH\_2          |                     
                   SDRAM\_MODEREG\_BURST\_TYPE\_SEQUENTIAL   |  
                   SDRAM\_MODEREG\_CAS\_LATENCY\_3           |                     
                   SDRAM\_MODEREG\_OPERATING\_MODE\_STANDARD |  
                   SDRAM\_MODEREG\_WRITEBURST\_MODE\_SINGLE;    
  /\* Configure a load Mode register command\*/     
  FMC\_SDRAMCommandStructure.FMC\_CommandMode = FMC\_Command\_Mode\_LoadMode;  
  FMC\_SDRAMCommandStructure.FMC\_CommandTarget = FMC\_Command\_Target\_bank2;    
  FMC\_SDRAMCommandStructure.FMC\_AutoRefreshNumber = 1;  
  FMC\_SDRAMCommandStructure.FMC\_ModeRegisterDefinition = tmpr;    
  /\* Wait until the SDRAM controller is ready \*/   
  while(FMC\_GetFlagStatus(FMC\_Bank2\_SDRAM, FMC\_FLAG\_Busy) != RESET)    
  {  
  }    
  /\* Send the command \*/  
  FMC\_SDRAMCmdConfig(&FMC\_SDRAMCommandStructure);  
**6. 刷新频率设置**  
  /\* Set the refresh rate counter \*/    
  /\* (7.81 us x Freq) - 20 \*/  
  /\* Set the device refresh counter \*/    
  FMC\_SetRefreshCount(683);  
  /\* Wait until the SDRAM controller is ready \*/     
  while(FMC\_GetFlagStatus(FMC\_Bank2\_SDRAM, FMC\_FLAG\_Busy) != RESET)  
  {  }  
关于刷新频率的的数值是这么得到的，这里得详细的说一下，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 RefreshCycles/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。          
       **SDRAM的手册上说是4096 refresh cycles every 64 ms，这里就应该是15.525us，而注释上面是按8192行计算**  
**的**，郁闷，有时间得研究研究。  


图片:21.png

到这里基本就设置完了，剩下就可以像使用SRAM一样，使用SDRAM了。

开发板上面用的SDRAM数据手册  
http://bbs.armfly.com/images/wind/file/zip.gif[42-45S16400J.pdf](http://bbs.armfly.com/job.php?action=download&aid=2382) (1136 K) 下载次数:35