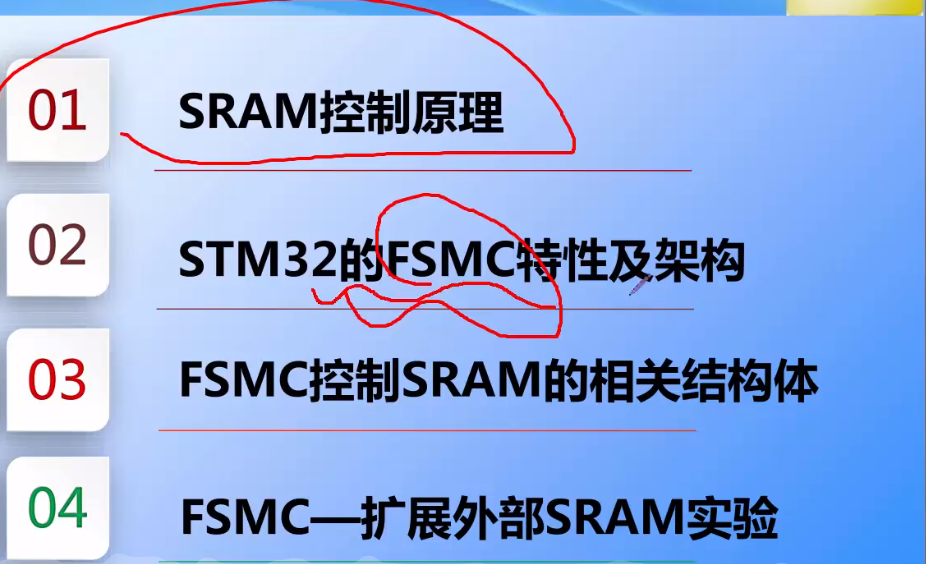
扩展外部SRAM芯片

使用STM32的一个叫做fsmc的片上外设来扩展，液晶屏也用这个

stm32ZET6这种多引脚的才能扩展SRAM。SRAM就有44个引脚

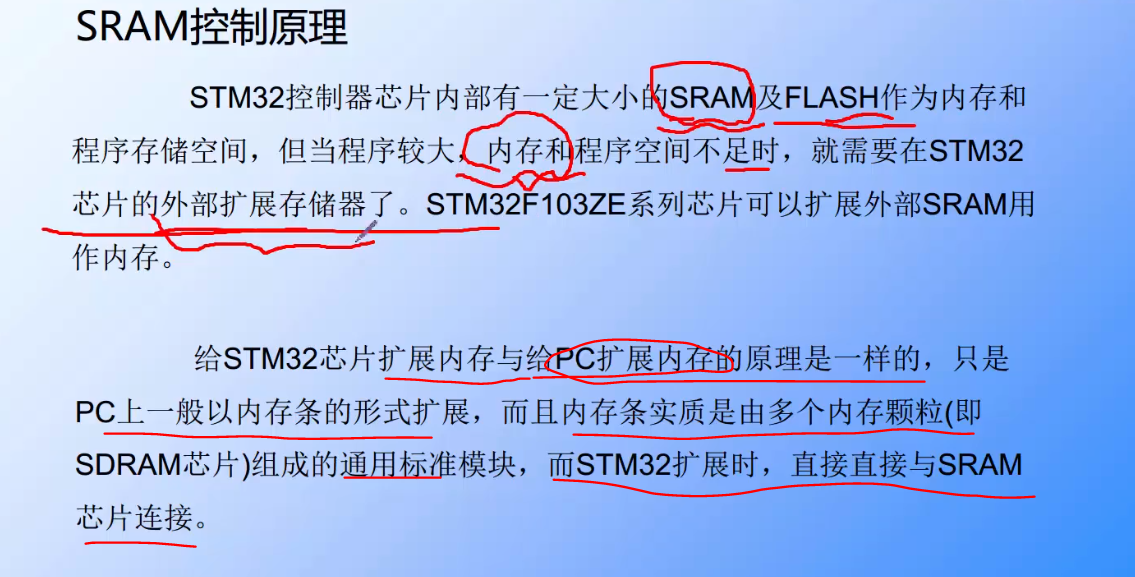
FSMC有很多内容，本章只讲和扩展SRAM有关的

1. 章节分类



1. SRAM控制原理
2. 为啥么要扩展SRAM?

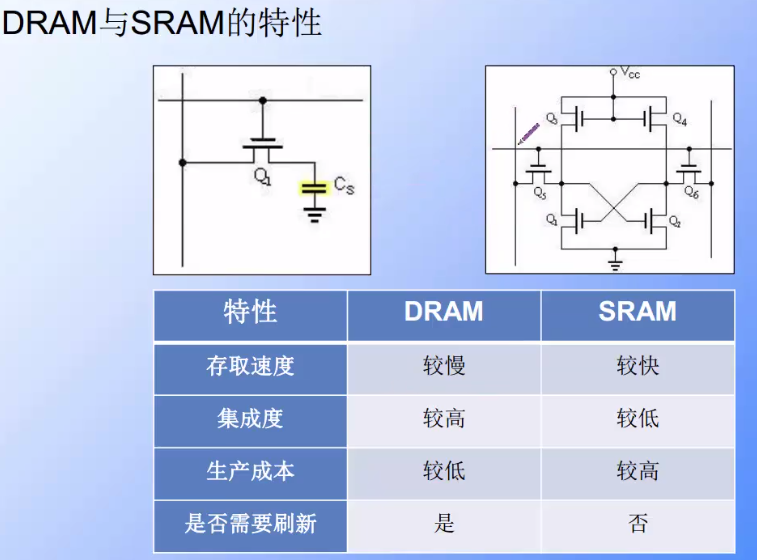
软件分四区，栈区，堆区，全局区的变量**都在SRAM里存**。有时候需要的变量特别大特别多(如MP3播放,跑系统)，32本身SRAM不足，因此要扩展外部SRAM。



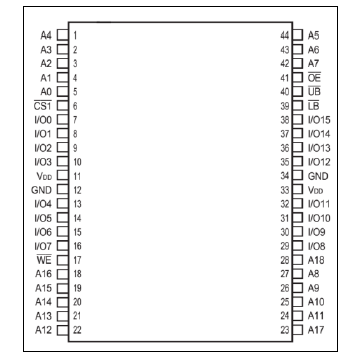
1. F1的局限性

F1只能扩展SRAM，不能扩展DRAM。因为F1内部没刷新电路，F429以上的有，才能扩展扩展DRAM。

同样大小的SRAM存储小于DRAM 价格高DRAM



1. SRAM芯片IS62WV51216



* 1. A是地址线 IO数据线。通过地址线来访问数据。
  2. 这些引脚可分三部分，A地址线，IO数据线，其他芯片控制线



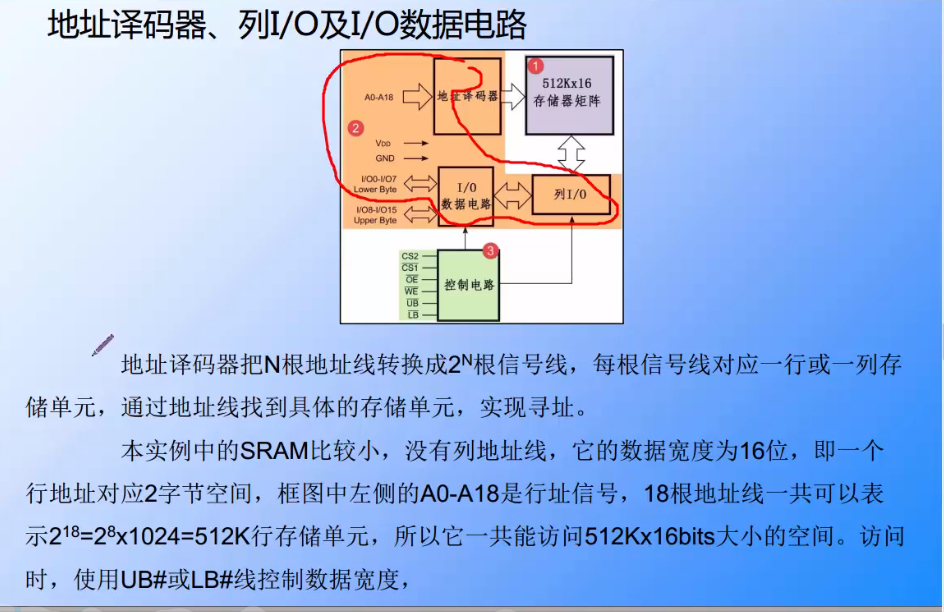
* 1. 一根地址线有16bit=2byte UB#和LB#控制写低字节还是高字节。

UB#时I/O8-I/O15高位数据线有效，则高字节有效

LB#时I/O0-I/O7低位数据线有效，则低字节有效

UB#时LB#时 I/O0- I/O15同时有效，更改16bit

1. 关于IS62WV51216的**内存大小描述**



8M bit static RAMs organized as 512K words by 16 bits.

( 8Mbit静态数据总线 按照512k的地址总线，每个地址总线上16bit数据大小分布)

* 1. A0~A18总共19根地址线，每根地址线能表示0/1的状态。
  2. 则一共能表示的信号数量为:2^19=524288 524288/1024=512k
  3. 512k的寻址地址上，每一个地址有16bit的数据内存大小
  4. 所以这个芯片容量为512\*(16/2)=1Mbyte = 8Mbit
  5. 8Mbit = 8\*1024\*1024bit=8388608bit 8388608/16/1024=512k

**注意:5K 5KB 5Kbit。只要没注明B是bit则一律按byte处置！！**

1. SRAM信号时序
2. 因为SRAM的通讯是**异步**的没时钟线，所以**信号之间的时间配合是非常重要的，一定要注释时间Taa tOHA等**
3. FSMC
4. 基本概念
   1. FSMC 是Flexible StaticMemory Controller 的缩写，译为灵活的静态存储控制器。它可以用于驱动包括SRAM、NOR FLASH 以及NAND FLSAH 类型的存储器。

**FSMC就是一个用来管理扩展存储器的一个外设。**

静态说明只能控制sram，而不能控制DRAM这种需要定时刷新的存储器

* 1. FSMC可以管理各种扩展存储器，此处只学管理SRAM
  2. 控制FSMC\_CTR寄存器的位14EXTMOD 可以让FSMC\_BWTR使能，从而让**写和读有不同的时序**

1. FSMC**数据读写原理**
   * 1. 按要求接线
     2. 通过BCR BTR BWTR寄存器 结合 扩展的存储设备手册的要把工作模式时序等 配置好
     3. STM32内部的 Exteral RAM地址0x6000 0000~0x9fff ffff**会自动通过FSMC映射到外部存储设备的地址上。然后对STM32内部的 Exteral RAM地址赋值，则直接对外部存储设备进行赋值。**
     4. 之类SPIiic的发地址检查缓存器是否为空 之类的操作一律不用。简单快捷
2. FSMC**数据读写地址**
   * 1. bank1：地址0x6800 0000~0x6BFF FFFF，地址线长度为3FF FFFF=67108863 。

67108863/1024/1024=64Ｍ，则bank1一共有64M大小的内存

**32的一根地址线指向1byte**

* + 1. SRAM的地址是0~2^19，即从0x0000 0000~0x0008 0000

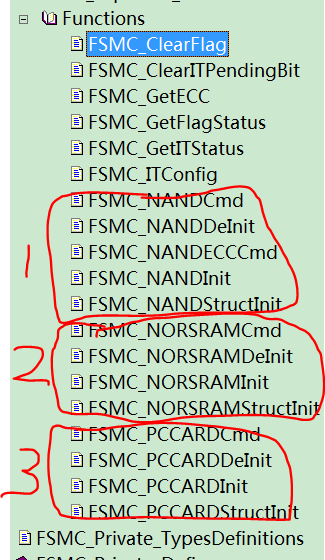
我们要想办法**把FSMC的ban1地址映射到SRAM上.**

即0x6800 0000~0x6BFF FFFF**映射到**0x0000 0000~0x0008 0000

* + 1. 只要把地址映射上了。向FSMC地址写数据时，32会自动根据时序向SRAM写数据

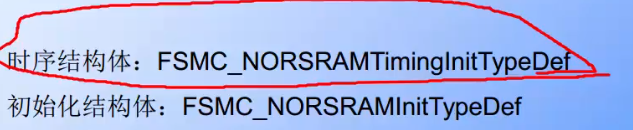
1. 结构体函数
   * 1. 主要分为三大类函数

因为FSMC只能驱动这四种外设。每种外设一种函数，两种用同一个函数

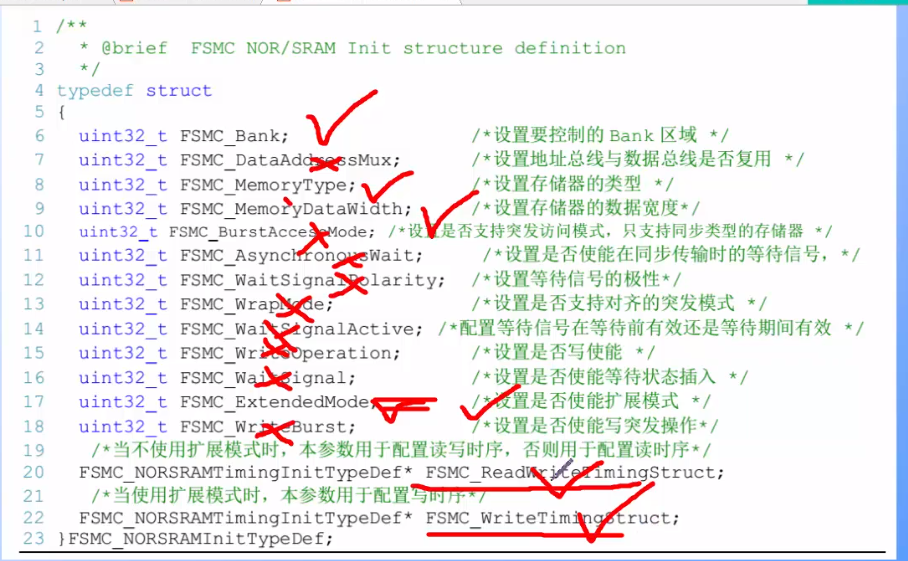


* + 1. 配置SRAM要用到以下两个结构体

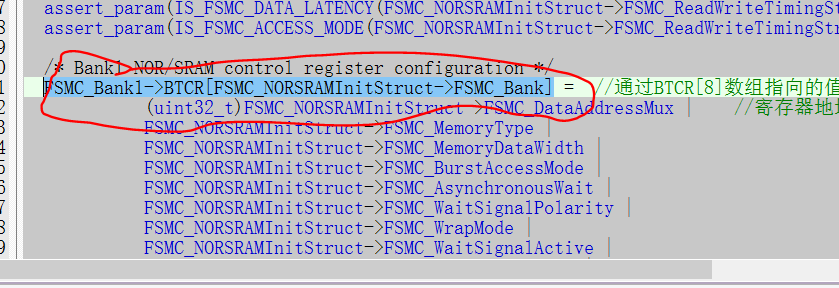
一个配置时序BTR和BWTR寄存器，一个配置初始化BCR寄存器。具体配置模式参照**《官方手册333页的模式位域》**



* + 1. 两个初始化函数区别
  1. FSMC\_NORSRAMStructInit () 默认值进行赋值，都赋值0xF
  2. FSMC\_NORSRAMInit () 根据自己的配置进行赋值
  3. 配置这些结构体成员

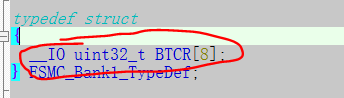


1. 编程中的一些问题
2. 关于这种写法的地址索引



FSMC\_Bank1->BTCR[FSMC\_NORSRAMInitStruct->FSMC\_Bank]代表的就是寄存器的地址，可如何把FSMC\_Bank1->BTCR[FSMC\_NORSRAMInitStruct->FSMC\_Bank]与地址联系起来呢？

* 1. FSMC\_Bank1的定义是 ((FSMC\_Bank1\_TypeDef \*) FSMC\_Bank1\_R\_BASE)，即FSMC\_Bank1\_TypeDef结构体类型的一个指针。值为0xA000 0000
  2. FSMC\_Bank1\_TypeDef结构体的成员变量为 ：即一个32位int型的数组。32的一个地址为1byte **则数组元素每增加一个就偏移4个地址。**



* 1. 假设FSMC\_NORSRAMInitStruct->FSMC\_Bank 赋值为= 0x00000002

则BTCR[FSMC\_NORSRAMInitStruct->FSMC\_Bank] == BTCR[0x00000002]

* 1. FSMC\_Bank1->BTCR[FSMC\_NORSRAMInitStruct->FSMC\_Bank] == 0xA000 0000->BTCR[0x00000002] 别忘了这个**整体是一个指针**
  2. BTCR[0]的地址就是 0xA000 0000
  3. **STM32的一个地址=1字节=8 bit，偏移一个地址等于偏移8bit**
  4. 0xA000 0000->BTCR[0x00000002]的地址就是: BTCR[0]的地址为0xA000 0000，BTCR[2]地址为0xA000 0000+4\*2=0xA000 0008
  5. 0xA000 0008即bank2的地址

**0xA000 0000 0xA000 0008 0xA000 0010 0xA000 0018**

**0xA000 0004 0xA000 000C 0xA000 0014 0xA000 001C**

**0xA000 0104 0xA000 010C 0xA000 0114 0xA000 011C**