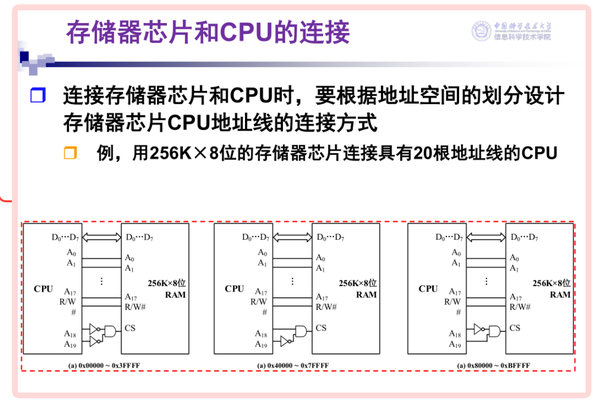
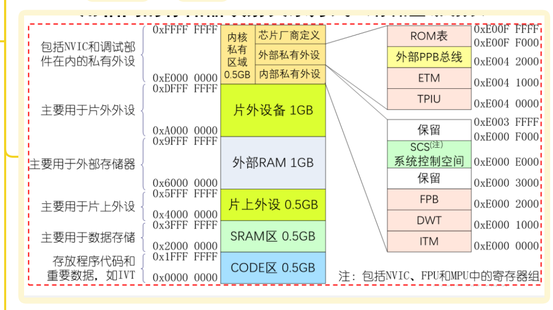
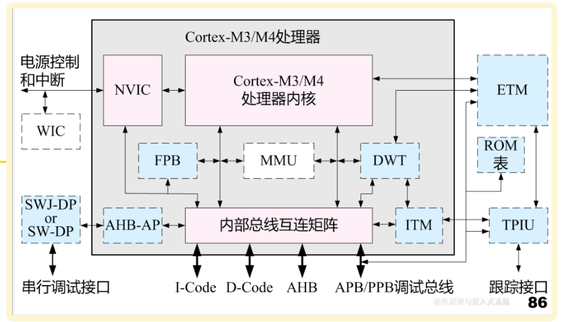
掌握

- 地址空间与存储器连接，存储器的位扩展、字扩展
  - 位扩展：在存储器芯片字数不变的前提下，进行数据的位数扩展
  - 字扩展：在存储器芯片的位数满足的前提下，进行字数扩展
  - 复合扩展
    - 在位长和字数均不足时，采用复合扩展方式
    - 先位扩展再字扩展

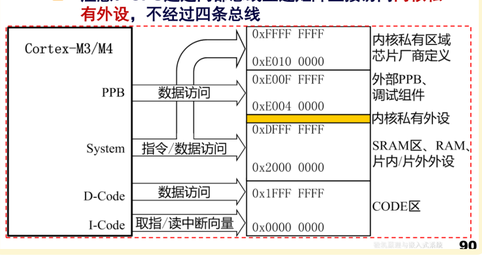


- 总线周期的四个阶段
  - 总线操作与总线周期
    - 通过总线进行信息交换的过程，称为总线操作
    - 总线设备完成一次完整信息交换的时间，称为总线周期（或总线传输周期）
    - 总线时序是指，总线操作过程中，总线上各信号之间在时间顺序上的配合关系
  - 总线周期的4个阶段
    - 请求及仲裁（Request and Arbitration）阶段：主模块请求，仲裁机构决定把下一个总线传输周期分给哪一个请求源
    - 寻址（Addressing）阶段：取得总线使用权的主模块，通过总线发出本次要访问的从模块（存储器地址或I/O端口）地址及有关命令，通知参与传输的从模块开始启动
    - 数据传输（Data Transferring）阶段：主模块和从模块进行数据传输，数据由源模块发出，经数据总线到达目的模块
    - 结束阶段：主模块、从模块的有关信息均从总线上撤销，让出总线，以便下一个总线传输周期其他模块能够使用总线
- Cortex-M3/M4处理器的存储器映射及总线系统
  - 存储器映射
    - 所有基于ARMv7M的Cortex-M系列处理器，都采用相同的存储器映射关系方式
    - 分区
      - 内核私有区域：0.5GB
      - 片外设备：1GB
      - 外部RAM区：1GB
      - 片上外设区：0.5GB
      - SRAM区：0.5GB
      - CODE区：0.5GB
    - 有助于提高设备之间的软件可移植性和代码可重用性



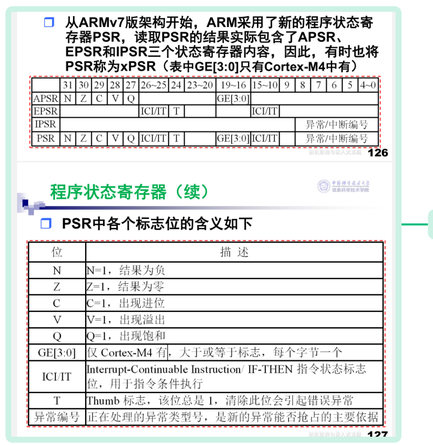


  - 总线系统
    - 核心：基于AHB总线协议的内部总线互连矩阵
    - 构成
      - I-Code总线
      - D-Code总线：I-Code总线与D-Code在物理上彼此独立，但两者之间有一个仲裁器，当I-Code和D-Code同时访问同一区域时，D-CODE优先
      - System，基于AHB-Lite规范的32位系统总线，也被称为AHB总线
      - APB/PBB，32位APB总线
      - CPU通过内部总线互连矩阵直接访问内核私有外设，不经过四条总线

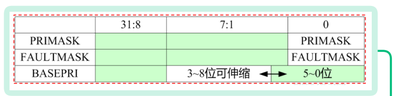


  - 其他总线
    - 调试访问端口（DAP）
      - 基于"增强型APB"总线规范的32位总线
      - 主要用于连接处理器内部的调试访问接口AP与外部调试端口DP，如SWJ-DP和SW-DP

- Cortex-M3/M4处理器16个常规寄存器及程序状态寄存器PSR
  - AAPCS(ARM Architecture Procedure Call Standard)规范
    - R0~R3：用于子程序之间的参数传递
    - R4~R11：用于保存子程序的局部变量
    - R12：作为子程序调用中间寄存器
  - 通用寄存器：R0~R12
    - R0~R7，8个低位寄存器，因受指令编码空间限制，许多16位Thumb指令只能访问低位寄存器
    - R8~R12，5个高位寄存器，可用于32位指令和少数几个16位指令（如MOV指令）
    - 系统复位之后，R0~R12的初始值均未定义
  - 栈指针：R13
    - Cortex-M3/M4处理器采用双堆栈设计
      - 有两个物理上的栈指针，也就是有两个R13寄存器
      - 一个是主栈指针MSP，另一个是进程栈指针PSP
      - 对于一般程序而言，两个栈指针只有一个可见
    - 系统复位
      - 系统复位之后，PSP的初值未定义；而MSP的初值存放在整个存储空间最开始（中断向量表）处第一个字中，系统初始化时，需要将其取出并对MSP进行赋值
    - 栈指针的选择是通过特殊寄存器 CONTROL设定的
    - MSP为默认栈指针，在系统复位后或处理器处于处理模式时，处理器使用MSP；PSP只能用于线程模式
  - 链接寄存器：R14（又称LR）
    - 用于保存函数或子程序调用时的返回地址，在函数或子程序结束时，LR中的数值用于调用返回
    - 在异常处理时，LR中将自动保存返回地址EXC_RETURN，异常处理结束用于异常/中断返回
    - 如果是嵌套调用，调用时需将LR中的数值压栈保存
  - 程序计数器：R15（又称PC）
    - 读PC时返回的是当前指令地址加4，这是因为流水线的特性以及与ARM7T系列处理器兼容的需要
    - 对PC的写操作可以使用MOVE指令以及数据处理指令来实现，以实现程序的跳转
    - 需要注意的是，无论使用跳转指令还是直接写PC寄存器，写入值必须是奇数，确保其最低位是"1"，以表示其处于Thumb状态，否则将被认为试图转入ARM模式，从而导致出现错误异常
  - 特殊寄存器
    - 这些寄存器没有映射到内存空间，只能利用MSR/ MRS指令，通过名字对其访问
      - Move Register from Special_register
        - 读
        - MRS <reg>, <special_reg>
      - Move Special_register from Register
        - 写
        - MSR <special_reg>, <reg>
    - xPSR



    - 特殊寄存器
      - 中断屏蔽寄存器



        - PRIMASK
          - 当最低位被置位（写入1）后，将屏蔽除复位、NMI和硬件错误以外所有的（优先级数值大于0的）系统异常和外部中断，类似于x86系统中的关中断（IF=0），以便处理紧急事务
          - 当紧急事务处理完毕，要及时将PRIMASK的最低位进行复位，类似于x86系统的"开中断"
        - FAULTMASK
          - 当最低位被置位后，硬件错误异常也被屏蔽。相当于把异常/中断的优先级门槛提高到"-1"
          - 常用于执行负责错误处理（包括硬件错误）的中断服务程序
          - 与PRIMASK不同的是，FAULTMASK无需主动清理
        - BASEPRI
          - BASEPRI寄存器的最低8位采用了"可伸缩"设计，具体宽度取决于芯片制造商实际设计的中断优先级数量
          - 写数不为零时，屏蔽优先级数值大于等于值的所有中断
          - 屏蔽大于等于0的中断时，需要使用PRIMASK
            - 写数为0时，不屏蔽
        - 中断屏蔽寄存器属于特殊寄存器，只有特权访问等级才可以进行读写访问
      - CONTROL寄存器
        - 用于选择线程模式的特权访问等级以及栈指针
        - nPRIV：设置线程模式的特权访问等级
          - 该位为0/1，处理器进入特权线程模式/非特权线程模式
        - SPSEL：选择线程模式中的堆栈指针
          - 当该位为0时，线程模式使用主栈指针MSP
          - 当该位为1时，线程模式使用进程栈指针PSP
          - 在处理模式下，该位始终为0，并且忽略对其的写操作
        - FPCA：配置FPU的Cortex-M4才有此位
          - 当发生异常时，若该位为1，浮点单元中的寄存器内容被压栈保存
          - 执行浮点指令时FPCA位自动置位，在异常处理程序入口处该位被硬件自动清除