第6章

# 实现Cortex-M3的全景概貌

* 流水线
* 详细的框图
* Cortex-M3的总线接口
* Cortex-M3的其它接口
* 外部私有总线
* 典型的连接方式
* 复位信号源

[译注]：本章相对篇幅较小，但读懂本章需要一些处理器体系结构基础知识，还需要一些ARM处理器特有的基础知知，且只有设计处理器的专业人员才必须精通。如果这方面比较薄弱，只需知道有哪些组件，能一句话讲出它们的功用，支持的调试方式，以及知道有哪几条总线即可。有些重要组件后面还会细讲。如NVIC，融入了后面各章中。在本章中，可能与调试有关的内容最难一下子理解。但不用担心，在第15章和第16章中，对调试相关的内容有展开论述。说实话，本章也是翻译得最累心的章节之一。

## 6.1 流水线

Cortex-M3处理器使用一个3级流水线。流水线的3个级分别是：取指，解码和执行，如图6.1所示：

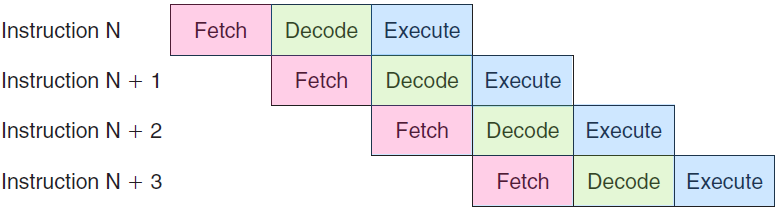


图6.1 Cortex-M3的三级流水线

有些人会提出质疑，认为其实是4级，理由是总线接口在访问内存时的行为。但是这一级是在处理器的外部，故而处理器自身还是只有3级流水线。

当运行的指令大多数都是16位时，你会发现处理器会每隔一个周期做一次取指。这是因为CM3有时可以一次取出两条指令来（一次能取32位），因此在第一条16位指令取来时，也顺带着把第二条16位指令取来了。此时总线接口就可以先歇一个周期再取指。或者如果缓冲区是满的，总线接口干脆就空闲下来了。有些指令的执行需要多个周期，在这期间流水线就会暂停。

当执行到跳转指令时，需要清洗流水线，处理器会不得不从跳转目的地重新取指。为了改善这种情况，CM3支持一定数量的ARMv7M新指令，可以避免很多微型跳转，如第4章讲到的IF-THEN语句块。

由于流水线的存在，以及出于对Thumb代码兼容的考虑，读取PC时，会返回当前指令地址+4的值。这个偏移量总是4，不管是执行16位指令还是32位指令，这就保证了在Thumb和Thumb2之间的一致性。

在处理器内核的预取单元中也有一个指令缓冲区，它允许后续的指令在执行前先在里面排队，也能在执行未对齐的32位指令时，避免流水线“断流”。不过该缓冲区并不会在流水线中添加额外的级数，因此不会使跳转导致的性能下降（penalty）更加恶化。

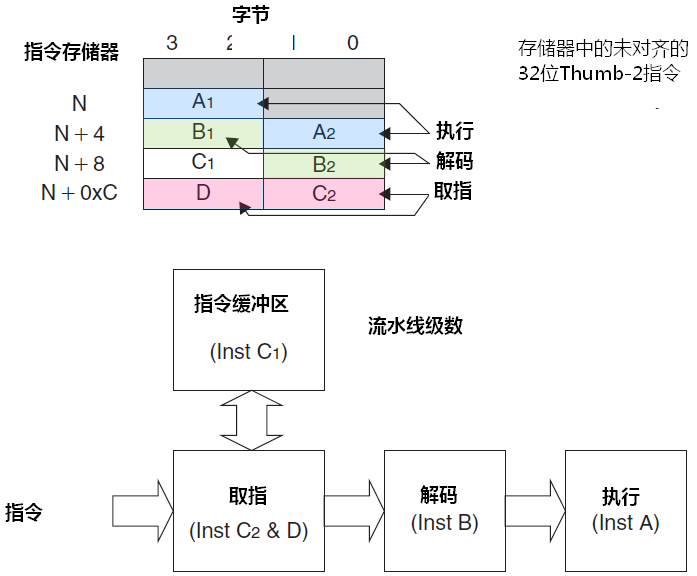


图6.2 取指单元使用缓冲区对32位指令处理的性能提升

## 6.2 详细的框图

CM3处理器其实是个大礼包，里面除了处理核心外，还有了好多其它组件，以用于系统管理和调试支持。

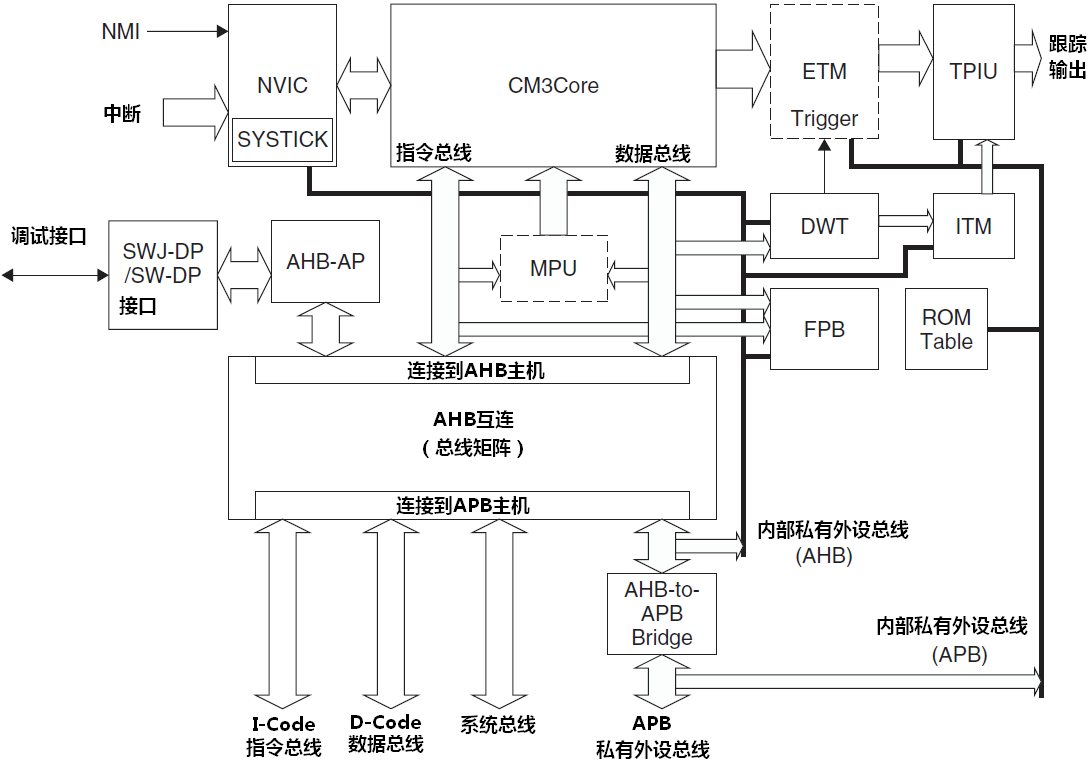


图6.3 Cortex-M3处理器系统方框图

请注意：虚线框住的MPU和ETM是可选组件，不一定会包含在每一个CM3的MCU中。好多新东东，图中一时看不清了，表6.1列出了新组件的清单。

表6.1 方框图中的缩写及其定义

|  |  |
| --- | --- |
| 缩写 | 含义 |
| NVIC | 嵌套向量中断控制器 |
| SYSTICK Timer | 一个简易的周期定时器，用于提供时基，亦被操作系统所使用 |
| MPU | 存储器保护单元（可选） |
| CM3BusMatrix | 内部的AHB互连 |
| AHB to APB | 把AHB转换为APB的总线桥 |
| SW-DP/SWJ-DP | 串行线调试端口/串行线JTAG 调试端口。通过串行线调试协议或者是传统的JTAG协议（专用于SWJ-DP），都可以用于实现与调试接口的连接 |
| AHB-AP | AHB访问端口，它把串行线/SWJ接口的命令转换成AHB数据传送 |
| ETM | 嵌入式跟踪宏单元（可选组件），调试用。 用于处理指令跟踪 |
| DWT | 数据观察点及跟踪单元，调试用。这是一个处理数据观察点功能的模块 |
| ITM | 仪器化跟踪宏单元 |
| TPIU | 跟踪单元的接口单元。所有跟踪单元发出的调试信息都要先送给它，它再转发给外部跟踪捕获硬件的。 |
| FPB | Flash地址重载及断点单元 |
| ROM表 | 一个小的查找表，其中存储了配置信息 |

可见，Cortex-M3处理器是以一个“处理器子系统”呈现的，其CPU内核本身与NVIC和一系列调试块都亲密耦合：

* + CM3Core：Cortex-M3处理器的中央处理核心
  + 嵌套向量中断控制器NVIC：NVIC是一个在CM3中内建的中断控制器。中断的具体路数由芯片厂商定义。NVIC是与CPU紧耦合的，它还包含了若干个系统控制寄存器。因为NVIC支持中断嵌套，使得在 CM3上处理嵌套中断时清爽而强大。NVIC还采用了向量中断的机制。在中断发生时，它会自动取出对应的服务例程入口地址，并且直接调用，无需软件判定中断源，为缩短中断延时做出了非常重要的贡献。
  + SysTick定时器：系统滴答定时器是一个非常基本的倒计时定时器，用于在每隔一定的时间产生一个中断，即使是系统在睡眠模式下也能工作。它使得OS在各CM3器件之间的移植中不必修改系统定时器的代码，移植工作一下子容易多了。SysTick定时器也是实现在NVIC内部的。
  + 存储器保护单元：MPU是一个选配的单元，有些CM3芯片可能没有配备此组件。如果有，则它可以把存储器分成一些regions，并分别予以保护。例如，它可以让某些regions在用户级下变成只读，从而阻止了一些用户程序破坏关键数据。
  + BusMatrix：BusMatrix是CM3内部总线系统的核心。它是一个AHB互连的网络，通过它可以让数据在不同的总线之间并行传送——只要两个总线主机不试图访问同一块内存区域。BusMatrix还提供了附加的数据传送管理设施，包括一个写缓冲以及一个按位操作的逻辑（ 位带(bit-band) ）。
  + AHB to APB Bridge：它是一个总线桥，用于把若干个APB设备连接到CM3处理器的私有外设总线上（内部的和外部的）。这些APB设备常见于调试组件。CM3还允许芯片厂商把附加的APB设备挂在这条APB总线上，并通过APB接入其外部私有外设总线。

框图中其它的组件都用于调试，通常不会在应用程序中使用它们，如下所示。

* + SW-DP/SWJ-DP：串行线调试端口（SW-DP）/串口线JTAG调试端口（SWJ-DP）都与AHB访问端口（AHB-AP）协同工作，以使外部调试器可以发起AHB上的数据传送，从而执行调试活动。在处理器核心的内部没有JTAG扫描链，大多数调试功能都是通过在NVIC控制下的AHB访问来实现的。SWJ-DP支持both串行线协议和JTAG协议，而SW-DP只支持串行线协议。
  + AHB-AP： AHB访问端口通过少量的寄存器，提供了对CM3所有存储器的访问机能。该功能块由SW-DP/SWJ-DP通过一个通用调试接口（DAP[译注]）来控制。当外部调试器需要执行动作的时候，就要通过SW-DP/SWJ-DP来访问AHB-AP，再由AHB-AP产生所需的AHB数据传送。

译注：DAP是SW-DP/SWJ-DP与AHB-AP之间的总线接口（详见第15章，图15.1）

* + 嵌入式跟踪宏单元ETM：ETM用于实现实时指令跟踪，但它是一个选配件，所以不是所有的CM3产品都具有实时指令跟踪能力。ETM的控制寄存器是映射到主地址空间上的，因此调试器可以通过DAP来控制它。
  + 数据观察点及跟踪单元DWT：通过DWT，可以设置数据观察点。当一个数据地址或数据的值匹配了观察点时，就说产生了一次匹配命中事件。匹配命中事件可以用于产生一个观察点事件，后者能激活调试器以产生数据跟踪信息，或者让ETM联动（以跟踪在哪条指令上发生了匹配命中事件——译者注）。
  + 仪器化跟踪宏单元ITM：ITM有多种用法。软件可以控制该模块直接把消息送给TPIU（类似printf风格的调试）；还可以让DWT匹配命中事件通过ITM产生数据跟踪包，并把它输出到一个跟踪数据流中。
  + 跟踪端口的接口单元TPIU：TIPU用于和外部的跟踪硬件（如跟踪端口分析仪）交互。在CM3的内部，跟踪信息都被格式化成“高级跟踪总线（ATB）包”，TPIU重新格式化这些数据，从而让外部设备能够捕捉到它们。
  + FPB：FPB提供flash地址重载和断点功能。Flash地址重载是指：当CPU访问某条指令时，若该地址在FPB中“挂了号”，则将把该地址重映射到另一个地址，后者亦在编程FPB时指出。结果，实际上是从映射过的地址处取指（通常，映射前的地址是flash中的地址，映射后的地址是SRAM中的地址，所以才是”Flash”地址重载——译者注）。此外，匹配的地址还能用来触发断点事件。Flash地址重载功能对于测试工作太有用了。例如，通过使用FPB来改变程序流程，就可以给那些不能在普通情形下使用的设备添加诊断程序代码（such as adding diagnosis program code to a device that cannot be used in normal situations unless the FPB is used to change the program control.）。
  + ROM表：它只是一个简单的查找表。其实更像一个“注册表”：提供了存储器的“注册”信息，这些信息指出，在这块CM3芯片中包括了哪些系统设备和调试组件，以及它们的位置。当调试系统定位各调试组件时，它需要找出相关寄存器在存储器中的地址，这些信息由此表给出。在绝大多数情况下，因为CM3有固定的存储器映射，所以各组件都对号入座——拥有一致的起始地址。但是因为有些组件是可选的，还有些组件是可以由制造商另行添加的，各芯片制造商可能需要定制他们芯片的调试功能。以后CM3芯片会有越来越多的品牌和型号。而林子大了什么鸟都有，如果确有厂商“玩另类”，它就必须在ROM表中给出这些“另类”的信息，这样调试软件才能判定正确的存储器映射，进而可以检测可用的调试组件是何种类型。

## 6.3 Cortex-M3的总线接口

这部分内容是给SoC设计师看的。如果你不是他们，是不能直接访问这里讲到的到总线接口的。

通常情况下，芯片厂商都会钩住（hook up）所有送往存储器和外设的总线信号。并且在少数情况下，你会发现芯片厂商把总线连接到了总线桥上，并且允许外部总线系统连接到芯片上。CM3处理器的总线接口是基于AHB-Lite和APB协议的，它们的规格在AMBA规格书（第4版）中给出。

### 6.3.1 I-Code总线

I-Code总线是一条基于AHB-Lite总线协议的32位总线，负责在0x0000\_0000 – 0x1FFF\_FFFF之间的取指操作。取指以字的长度执行，即使是对于16位指令也如此。因此CPU内核可以一次取出两条16位Thumb指令。

### 6.3.2 D-Code总线

D-Code总线也是一条基于AHB-Lite总线协议的32位总线，负责在0x0000\_0000 – 0x1FFF\_FFFF之间的数据访问操作。尽管CM3支持非对齐访问，但你绝不会在该总线上看到任何非对齐的地址，这是因为处理器的总线接口会把非对齐的数据传送都转换成对齐的数据传送。因此，连接到D-Code总线上的任何设备都只需支持AHB-Lite的对齐访问，不需要支持非对齐访问。

### 6.3.3 系统总线

系统总线也是一条基于AHB-Lite总线协议的32位总线，负责在0x2000\_0000 – 0xDFFF\_FFFF和0xE010\_0000 – 0xFFFF\_FFFF之间的所有数据传送，取指和数据访问都算上。和D-Code总线一样，所有的数据传送都是对齐的。

### 6.3.4 外部私有外设总线

这是一条基于APB总线协议的32位总线。此总线来负责0xE004\_0000 – 0xE00F\_FFFF之间的私有外设访问。但是，由于此APB存储空间的一部分已经被TPIU、ETM以及ROM表用掉了，就只留下了0xE004\_2000-E00F\_F000这个区间用于配接附加的（私有）外设。

### 6.3.5 调试访问端口总线

调试访问端口总线接口是一条基于“增强型APB规格”的32位总线，它专用于挂接调试接口，例如SWJ-DP和SW-DP。

不要挪用此总线。第15章（调试架构）给出该总线的更多信息，在ARM的文档《CoreSight Technology System Design Guide (Ref 3)》中也有更详尽的论述。

## 6.4 Cortex-M3的其它接口

除了总线接口之外，CM3还有若干个用于其它目的的接口，这些接口的信号都不大可能会引出到引脚上，而只用于连接SoC不同的部分，或者干脆就没有使用。关于这些信号的详述，请参阅《Cortex-M3 Technical Reference Manual(TRM)(Ref1)》。表6.2中给出了它们中一些信号的简短小结。

表6.2 杂项接口信号

|  |  |
| --- | --- |
| 信号组 | 功能 |
| 多处理机通信（TXEV, RXEV） | 多处理机之间的简单任务同步信号 |
| 休眠信号  (SLEEPING, SLEEPDEEP) | 电源管理所用的休眠状态 |
| 中断状态信号  (ETMINTNUM, ETMINTSTATE, CURRPRI) | 中断操作的状态，用于ETM操作和调试 |
| 复位请求(SYSRESETREQ) | 来自NVIC的复位请求输出 |
| 锁定(Lockup)［译注］和停机(Halted)状态  (LOCKUP, HALTED) | 指示处理器进入了锁定状态（由在硬fault和NMI服务例程的执行错误导致），或者指示处理器被喊停（因为调试动作导致） |
| 端输入(ENDIAN) | 在内核复位时设置端模式 |
| ETM接口 | 连接到嵌入式跟踪宏单元（用于指令跟踪） |
| ITM的ATB接口 | 高级跟踪总线（ATB）是ARM CoreSight调试架构下的一个总线协议，用于跟踪数据的传送。在这里，该接口负责把来自ITM的跟踪数据输出到TPIU |

［译注］第12章讨论有关Lockup的更多内容

## 6.5 外部私有外设总线

CM3处理器有一个外部私有外设总线(PPB)接口。外部PPB接口是基于高级外设总线(APB)协议构造的。用于非共享的系统设备，例如调试组件。为了支持CoreSight设备，该接口又包含了称为“PADDR31”的信号，给出传送的发源地。若该信号为0，则表示是运行在CM3内部的软件产生了传送操作；若为1，则表示是调试硬件产生了传送操作。有了这个信号，外设就可以有选择地响应，比如：只响应调试硬件。也可以通融点：当软件发起数据传送时，限制一些功能。

该总线是专用的，不服务于普通的外设，这个规矩只能靠芯片设计者自觉遵守。如果设计者把通用的外设连接到该总线上，用户在使用芯片时就往往会遇到各种莫名其妙的问题——由特权访问管理造成。例如，在用户级下访问这些设备，或者在使用MPU时把这些设备从其它的存储regions中分开，都会遇到问题，势必影响芯片的销量。

外部PPB不支持非对齐访问。因为该总线的宽度是32位并且是基于APB的，当你在为该存储区域设计外设时，必须确保所有的寄存器地址都是按字对齐的。另外，在编写这些设备的驱动程序时，最好让所有的访问都使用字的长度。最后，PPB访问永远是小端的。

## 6.6 连接方式样板

由上可见，CM3中有若干个总线接口，初学者很容易混淆，也不太容易弄清楚它们是怎样与其它设备和存储器连接的。这里给出一个样板的连接实例，如图6.4所示。

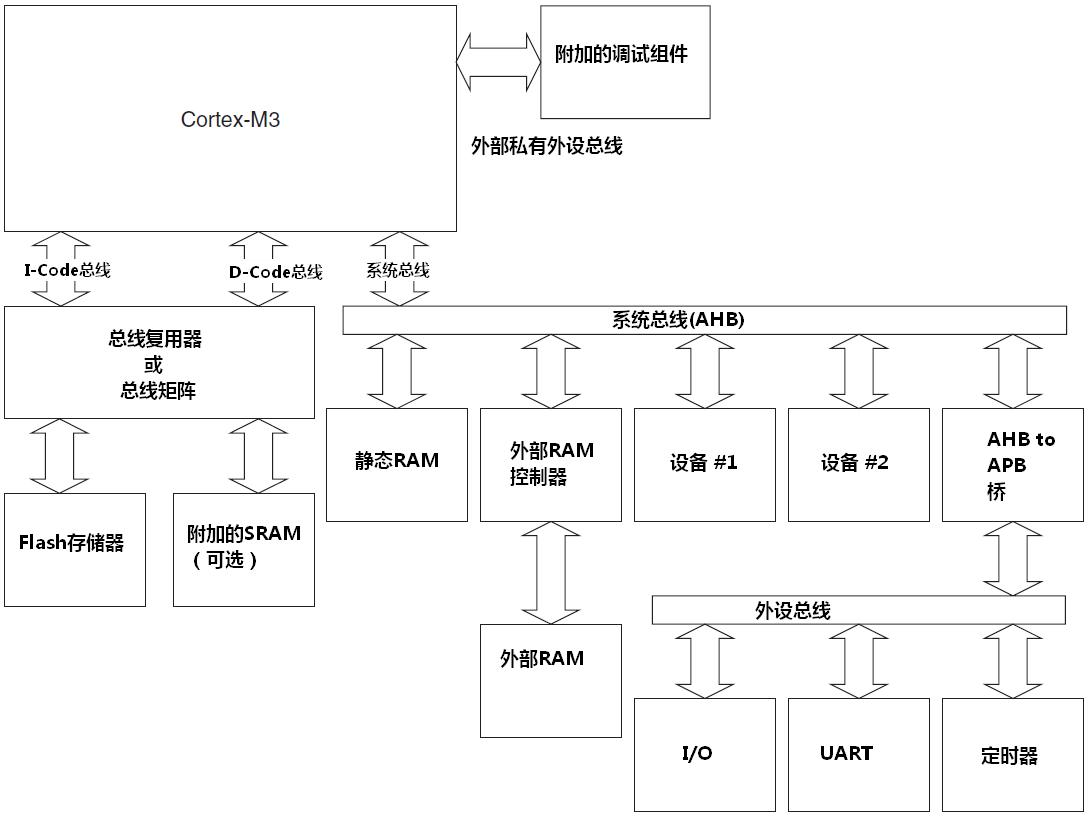


图6.4 Cortex-M3总线连接样板范例

因为代码存储区既可以由指令指令总线(I-Code)访问（当从此区取指时），也可以被数据总线（D-Code）访问（当在此区访问数据时），需要在中间插入一个总线开关，称为“总线矩阵［译注］”；或者使用一个AHB总线复用器。如果使用了总线矩阵，则闪存和附加的 SRAM（如果有的话）可以被both I-Code和D-Code访问。总线矩阵可以在ARM的AMBA开发包ADK（ADK，AMBA组件和示例系统的集合，使用VHDL/Verilog编写）中提供。

［译注］：这里所讲的总线矩阵不是CM3内部的总线矩阵，它们是两码事。CM3内部的总线矩阵是专门设计的，不能作为一个通用的AHB开关来使用。

当数据访问和取指同时尝试访问同一块区域时，可以赋予数据访问更高的优先级以提高性能。

在使用AHB总线矩阵把取指和数据访问分开后，如果指令总线和数据总线在同一时刻访问不同的存储器设备（例如，从flash中取指的同时从附加的SRAM中访问数据），则两者可以并行不悖。但若是只使用了总线复用器，则数据传送就不能同时发生了，然而这时电路尺寸能做得更小。不过，通常的CM3单片机都使用系统总线来连接SRAM。而且主SRAM确实应该使用系统总线来连接。只有这样才能落到SRAM存储器的地址区，从而得以利用CM3的位带操作能力。

有些脚数比较多的单片机会带外部总线接口（EMI）。这种情况下，需要一个外部存储器控制器，因为AHB不接受直接把片外存储器挂在它上面，通常外部存储器控制器也连接到系统总线上。其它的AHB设备则可以简单地连接到系统总线上，而不需要额外的总线矩阵。

图6.4给出的只是一个很简单的典型示范，芯片设计师也可以选择其它的总线连接方案。对于软件/固件的开发，不需了解这么多细节，只需要知道详细的存储器映射就够了。

上图显示出的功能框，像总线矩阵、AHB-to-APB总线桥、存储器控制器、I/O接口、定时器以及UART等，都可以从ARM和其它IP供应商处取得。不同的CM3单片机其片上外设也不同。因此在使用时，你还需要参考器件厂家提供的参考手册。

## 6.7 复位信号

基于CM3的单片机对复位电路有特定的要求，具体内容在《Cortex-M3 Technical Reference Manual(Ref1)》中给出，它列出了若干个可以使用的复位信号。不过，实现成单片机后，往往只用到了1至2个。至余其它的，芯片厂商会在芯片中布设复位信号发生器，由它在内部产生剩余的复位信号。如欲获取细节，还需要参考制造商提供的数据手册，以理解如何正确复位其芯片。在CM3处理器的水平上，复位信号由表6.3列出。

表6.3 Cortex-M3中的各种复位信号

|  |  |
| --- | --- |
| 复位信号 | 描述 |
| 上电复位（nPORESET） | 在器件上电时需要把复位置为有效（assert），把处理器核心和调试系统一起复位 |
| 系统复位（nSYSRESET） | 只影响处理器核心、NVIC（与调试相关的除外）以及MPU，不复位调试系统 |
| 测试复位（nTRST） | 只复位调试系统 |

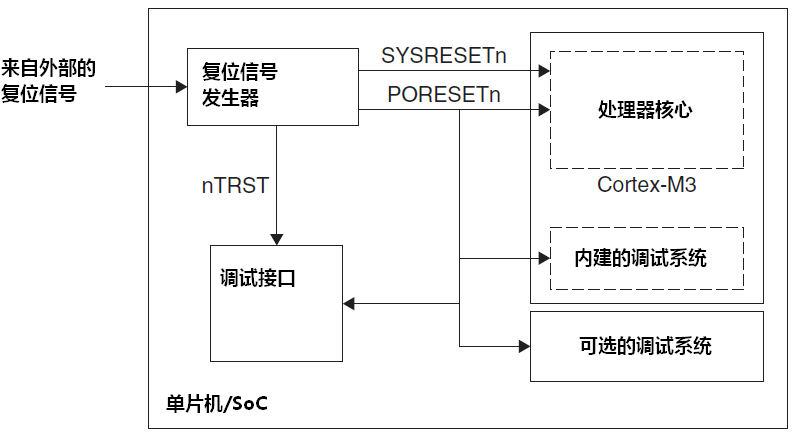


图6.5 典型的Cortex-M3芯片内部复位信号和其作用范围示意图