第 26 章 附录

目录

本章包括下列主题：

附录 A：I²C™ 概述 26-2
附录 B：CAN 概述 26-12
附录 C：编解码器协议概述 26-25

I²C 是 Philips 公司的商标。

附录 A： I^2C™ 概述

本附录提供 I^2C™ 总线的概述，其中第 A.2 节 “对 I^2C 器件寻址”讨论了在 I^2C 模式下 SSP 模块的工作原理。

I^2C 总线是双线制串行接口总线。其原始规范 （或标准模式）是为最高 100 kbps 的数据传输速率而制定的。I^2C 还支持增强型规范 （或快速模式， 400 kbps）。标准模式器件和快速模式器件连接在同一总线上时，如果总线以快速器件的工作速度运行，则两类器件均可正常工作。

I^2C 接口使用了综合的协议以确保数据发送与接收的可靠性。当发送数据时，其中一个器件作为主机，它启动总线上的传输并产生时钟信号以允许该传输，而其他器件则充当从机。除了全局呼叫支持外，从机协议的各部分都由 SSP 模块的硬件实现，而主机协议的各部分则需要在 PIC16CXX 的软件中处理。MSSP 模块可以实现所有的 I^2C 主机协议、全局呼叫地址和最大速率可达 1 Mbps 的数据传输。Microchip 的某些串行 EEPROM 支持 1 Mbps 的数据传输。表 A-1 定义了 I^2C 总线的部分术语。

在 I^2C 接口协议中，每个器件都有一个地址。当主机要启动一次数据传输时，它首先发送与之“通话”的器件地址。所有器件都 “侦听”该地址，查看是否与自己的地址匹配。在该地址中有一位用于指定主机是希望读还是希望写它寻址的从机。在数据传输过程中，主机和从机总是工作在相反的模式 （发送器 / 接收器）下。也就是说主机和从机只能工作在以下两种关系之一：

- 主机——发送器，从机——接收器
- 从机——发送器，主机——接收器

在这两种情况下，都由主机产生时钟信号。

为了实施总线的 “线与”功能，时钟线 （SCL）和数据线 （SDA）的输出都必须采用漏极开路或集电极开路的电路。需要外接上拉电阻，以保证总线在没有器件将其拉低时为高电平。 I^2C 总线上连接的器件数量仅受到最大 400 pF 的总线负载规范和寻址能力的限制。

A.1 启动和终止数据传输

在总线无数据传输时（空闲时），时钟线（SCL）和数据线（SDA）都通过外部上拉电阻拉为高电平。由启动和停止条件决定数据传输的启动和停止。启动条件定义为当 SCL 为高电平时，SDA 发生由高到低的转变； 停止条件定义为当 SCL 为高电平时，SDA 发生由低到高的转变。图 A-1 显示了启动和停止条件。主机产生这些条件以启动和终止数据传输。根据启动和停止条件的定义，当数据正在传输时，只有在 SCL 线为低电平时 SDA 线的状态才允许改变。

图 A-1： 启动和停止条件

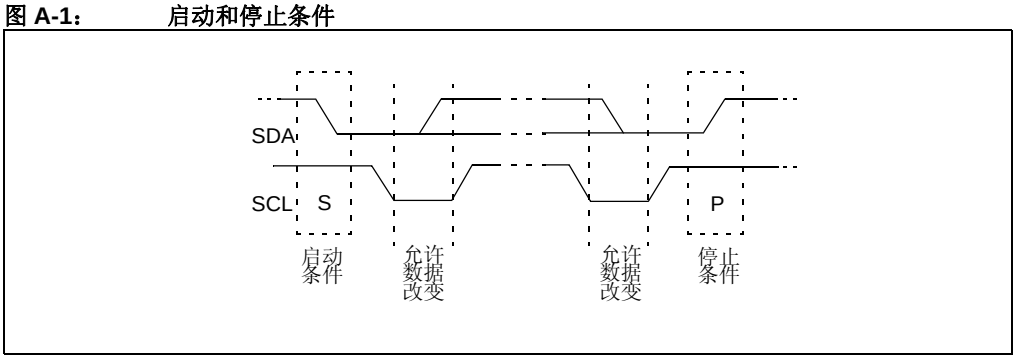


表 A-1： I^2C 总线术语表

术语	说明
发送器	向总线发送数据的器件。
接收器	从总线接收数据的器件。
主机	启动传输、产生时钟并终止传输的器件。
从机	被主机寻址的器件。
多主机	系统中有一个以上的主器件。这些主器件可尝试同时对总线进行控制而不会破坏报文。
仲裁	保证只有一个主器件控制总线的过程，可确保传输的数据不会被破坏。
同步	两个或多个器件的时钟信号同步的过程。

A.2 对 I²C 器件寻址

有两种地址格式：最简单的一种是带有 7 位地址格式 R/$\overline{W}$ 位（图 A-2）。另一种是较复杂的带有 10 位地址格式加上一个 R/$\overline{W}$ 位（图 A-3）。对于 10 位地址格式，必须发送两个字节。第一个字节的前五位用来指定当前采用的是 10 位地址格式，发送的第一个字节中包括 5 位用于指定 10 位地址的 5 位以及地址的两个 MSb 和 1 位 R/$\overline{W}$ 位。第二个字节是其余的 8 位地址。

图 A-2： 7 位地址格式

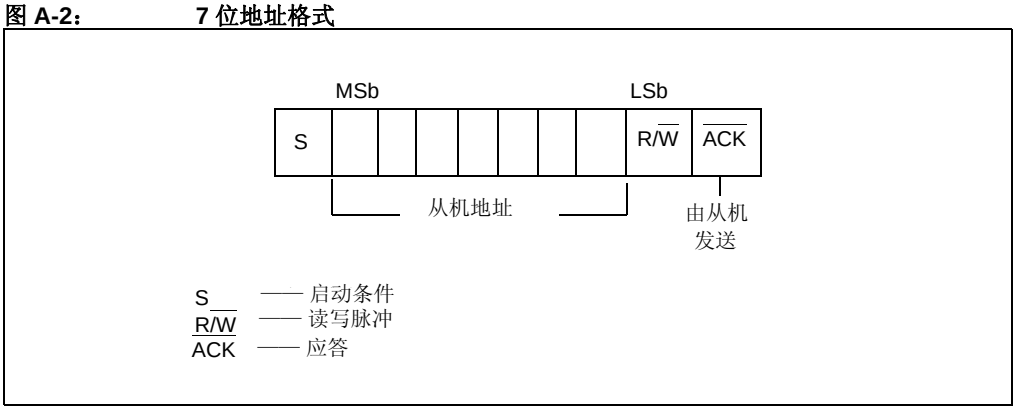
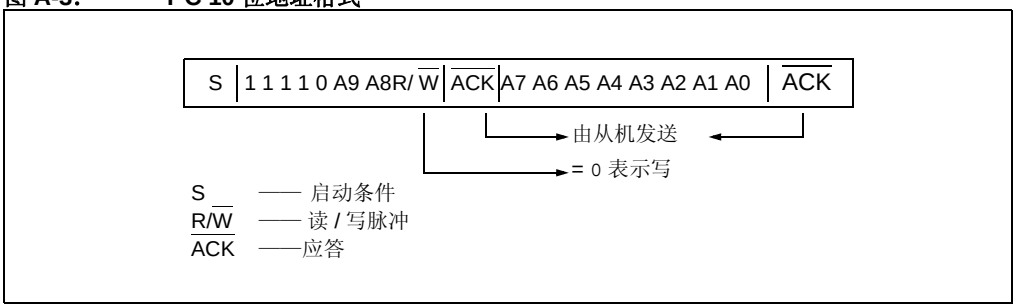


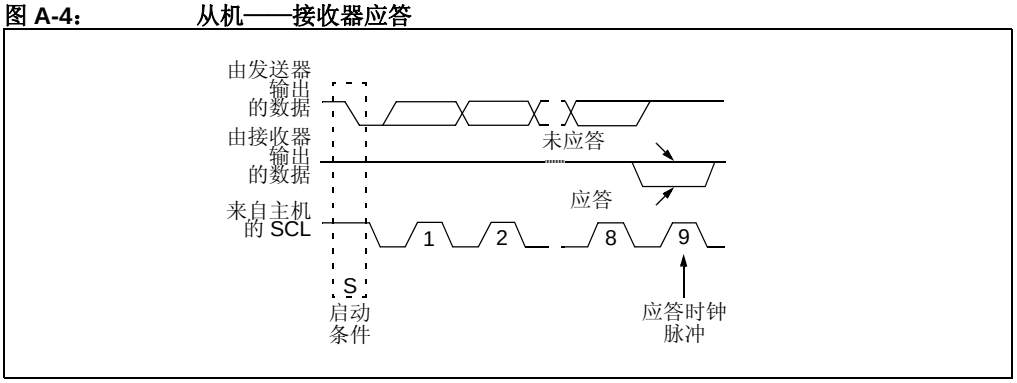
图 A-3： I²C 10 位地址格式



A.3 发送应答

所有数据都必须逐字节发送，每次数据传输所发送的字节数没有限制。接收到每个字节后，从机（接收器）会产生一个应答位（$\overline{ACK}$）（图 A-4）。如果从机（接收器）没有对从机地址或已接收的数据作出应答，主机必须中止发送。从机必须保持 SDA 为高电平，以便主机可以产生停止条件（图 A-1）。

图 A-4： 从机——接收器应答



如果主机接收数据（主机作为接收器），则除了最后一个字节外，主机在收到每一个数据字节后都会产生一个应答信号。如果主机没有发出应答信号（未作出确认），则通知从机（发送器）结束数据传输。然后从机释放 SDA 线，以便使主机能够产生停止条件。主机也可以在确认脉冲输出期间产生停止条件，以有效终止数据传输。

如果从机需要延迟下一个字节的传输，可以通过保持 SCL 线为低电平，迫使主机进入等待状态。当从机释放 SCL 线时，将继续数据传输。这就使得从机在允许时钟启动前先移入接收到的数据或取出需要发送的数据。这种等待状态技术也可以用在对位的处理上，如图 A-5 所示。

图 A-5： 数据传输等待状态

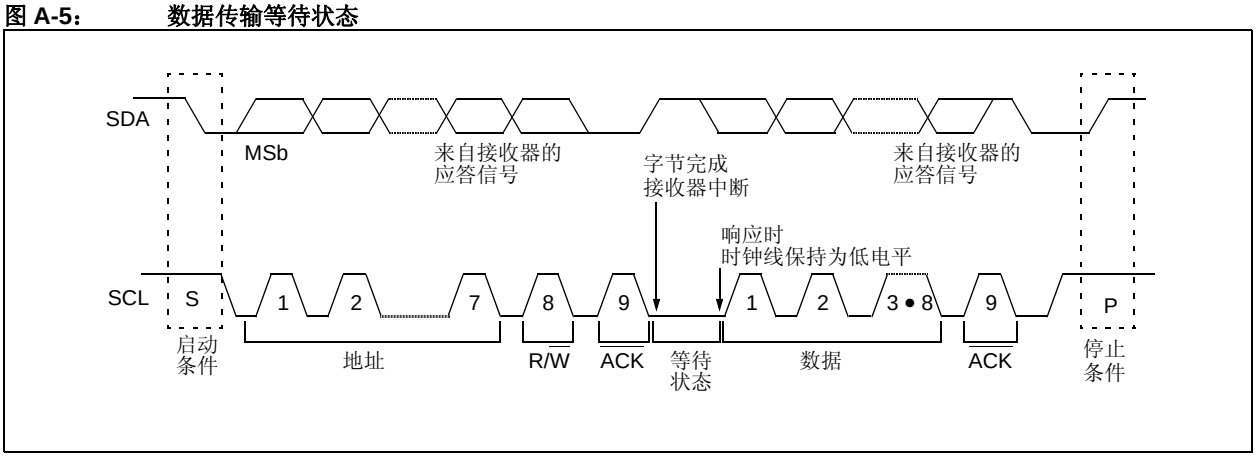


图 A-6 和图 A-7 描述了主机作为发送器和主机作为接收器时的数据传输序列。

图 A-6： 主机作为发送器的序列

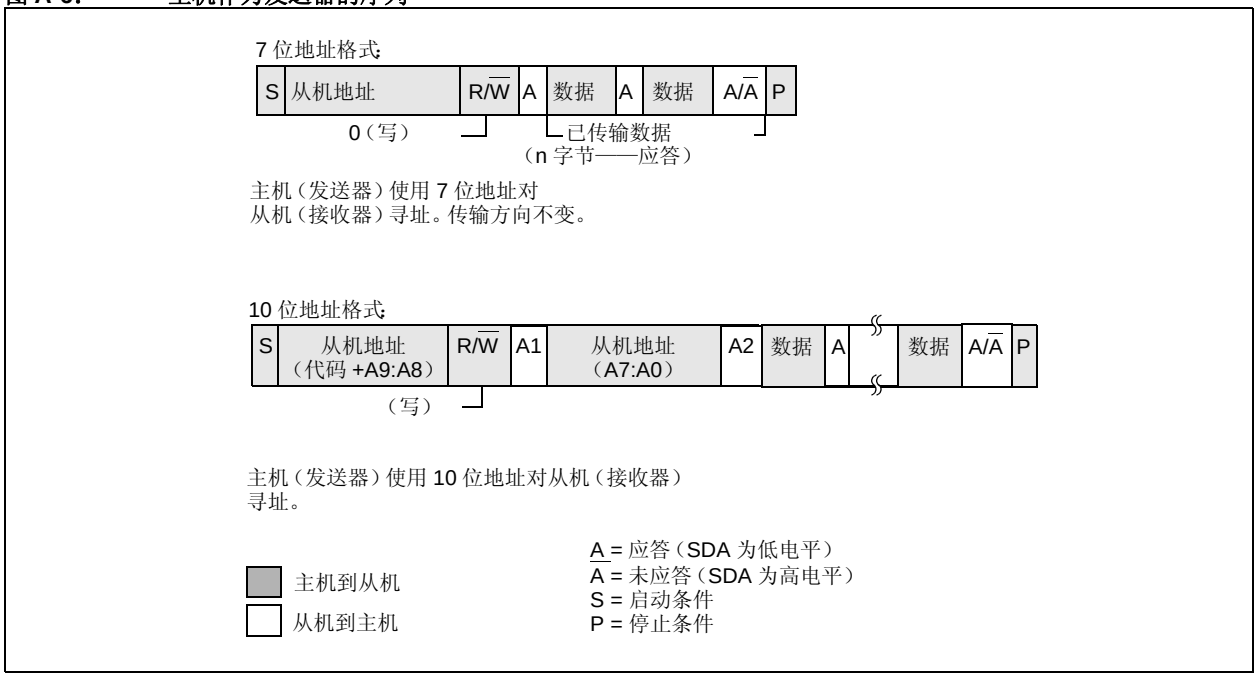
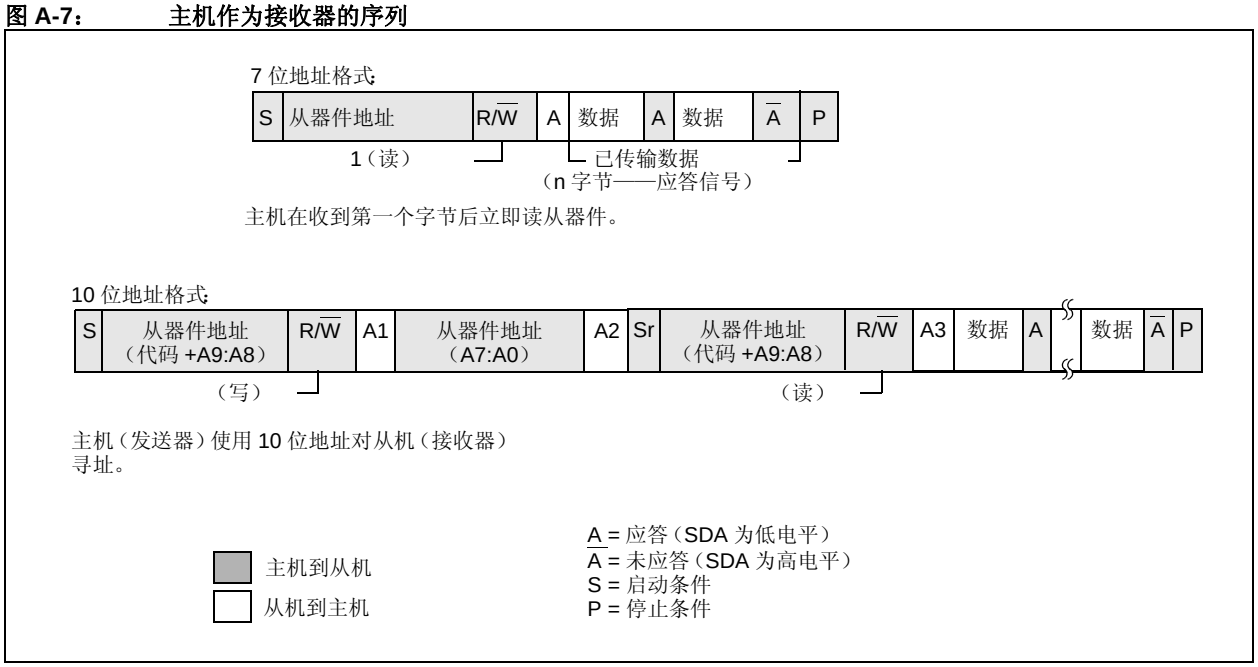
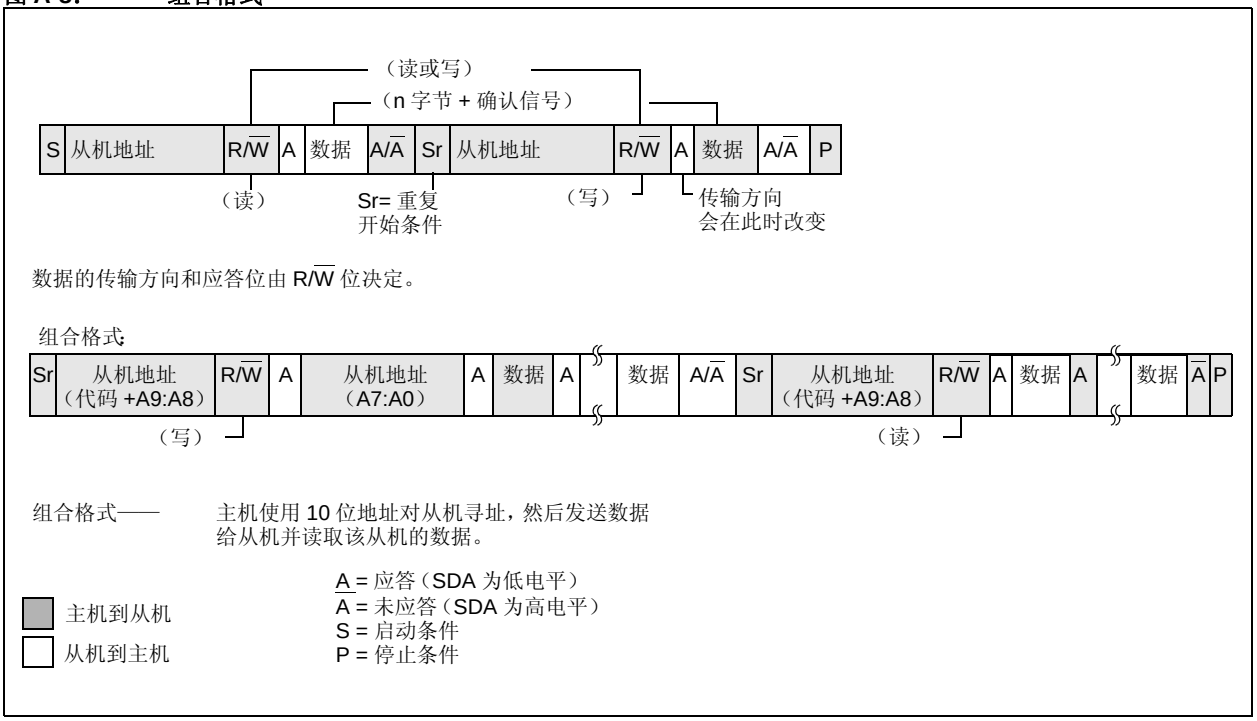


图 A-7： 主机作为接收器的序列



如果主机不想放弃总线控制权 （通过产生停止条件），必须产生一个重复启动条件 （Sr）。这个条件和启动条件相同（SCL 保持为高时，SDA 由高变低），只是它发生在数据传输应答脉冲产生之后 （而不是在总线处于空闲状态时）。这样主机就可以把 “命令” 发送给从机，然后接收所需要的信息或对另外一个从器件进行寻址。图 A-8 所示为此过程。

图 A-8： 组合格式



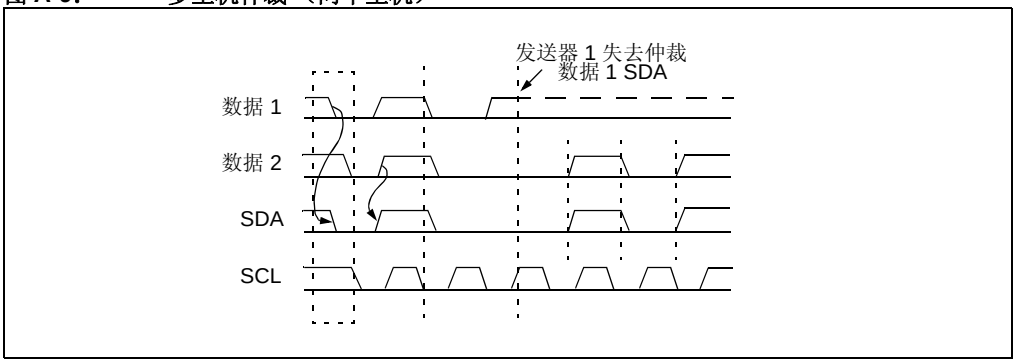
A.4 多主机

I^2C 协议允许系统有一个以上的主机，这称为多主机系统。当两个或两个以上的主机试图同时传输数据时，就会产生总线仲裁和同步。

A.4.1 仲裁

当 SCL 线为高电平时，在 SDA 线上进行仲裁。当一个主机在另一个主机发送低电平时发送了高电平，则第一个主机失去仲裁（图 A-9）并关闭其数据输出级。失去仲裁的主机在数据字节结束（即失去仲裁）之前可以产生时钟脉冲。如果多个主器件对同一器件寻址，则会继续进行数据仲裁。

图 A-9： 多主机仲裁（两个主机）



兼有从机功能且失去仲裁的主机，必须立即切换到从机（接收器）模式。这是因为赢得仲裁的主机（发送器）可能会对它进行寻址。

不允许在以下情况进行仲裁：

- 重复启动条件
- 停止条件和数据位之间
- 重复启动条件和停止条件之间

编程时应该注意以确保不会发生这些情况。

A.4.2 时钟同步

时钟同步发生在器件开始仲裁之后。它是由连接到 SCL 线上的 “线与”功能完成的。当 SCL 线的电平由高变低时，相关器件就开始对其时钟低电平时间进行计算。一旦器件时钟变为低电平，那么在其时钟线（SCL）变为高电平之前，SCL 线将一直保持为低电平。如果另一个器件的时钟仍处于低电平状态，那么器件时钟由低电平变为高电平不会改变 SCL 线的状态。 SCL 线保持低电平的时间由低电平时间最长的器件决定。低电平持续时间较短的器件将进入高电平等待状态，直到 SCL 线变为高电平。此时，所有器件开始计算时钟高电平的时间。第一个变为低电平的器件又会把 SCL 拉低。 SCL 线保持高电平的时间由高电平时间最短的器件决定，如图 A-10 所示。

图 A-10： 时钟同步

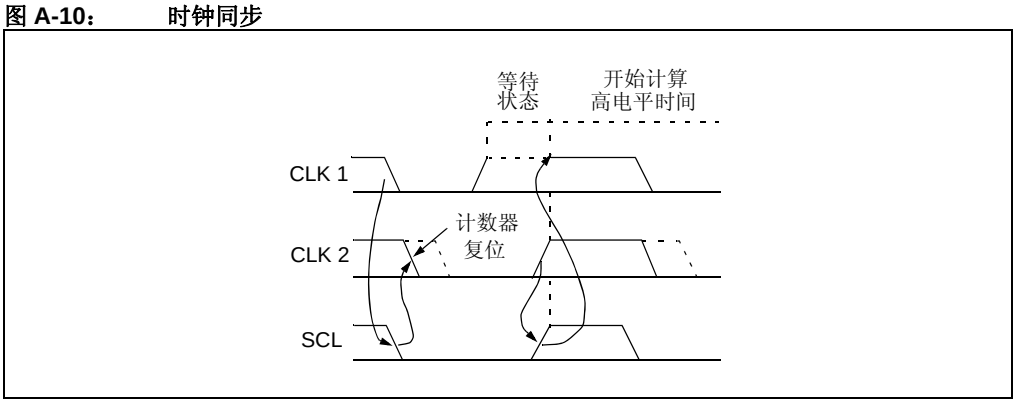


表 A-2 和表 A-3 显示了 I^2C 总线规范。“参数编号”一列是为了方便用户在器件数据手册中查找相应的参数。图 A-11 和图 A-12 显示了相应波形的时序。

图 A-11： I^2C 总线启动 / 停止位时序规范

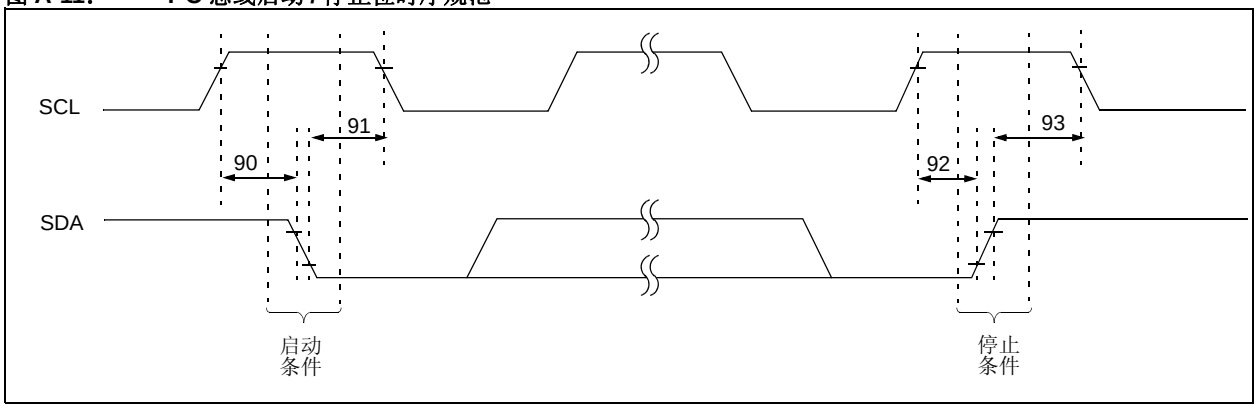


表 A-2： I^2C 总线启动 / 停止位时序规范

参数编号	符号	特性		最小值	典型值	最大值	单位	条件
90	TSU:STA	启动条件	100 kHz 模式	4700	—	—	ns	仅与重复启动条件相关
		建立时间	400 kHz 模式	600	—	—		
91	THD:STA	启动条件	100 kHz 模式	4000	—	—	ns	在这段时间后，将产生第一个时钟脉冲
		保持时间	400 kHz 模式	600	—	—		
92	TSU:STO	停止条件	100 kHz 模式	4700	—	—	ns	
		建立时间	400 kHz 模式	600	—	—		
93	THD:STO	停止条件	100 kHz 模式	4000	—	—	ns	
		保持时间	400 kHz 模式	600	—	—		

图 A-12： I^2C 总线数据时序规范

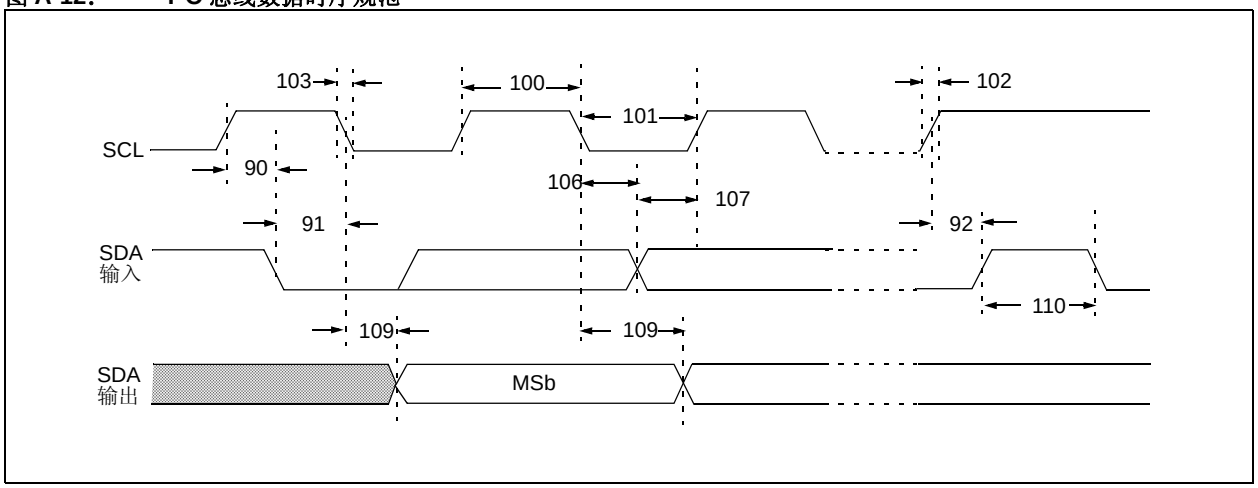


表 A-3：　　I²C 总线数据时序规范

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	4.0	—	μs	
			400 kHz 模式	0.6	—	μs	
101	TLOW	时钟低电平时间	100 kHz 模式	4.7	—	μs	
			400 kHz 模式	1.3	—	μs	
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20+0.1Cb	300	ns	指定 Cb 在 10 到 400 pF 之间
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20+0.1Cb	300	ns	指定 Cb 在 10 到 400 pF 之间
90	TSU:STA	启动条件建立时间	100 kHz 模式	4.7	—	μs	仅与重复启动条件有关
			400 kHz 模式	0.6	—	μs	
91	THD:STA	启动条件保持时间	100 kHz 模式	4.0	—	μs	在这段时间后，将产生第一个时钟脉冲
			400 kHz 模式	0.6	—	μs	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	μs	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	**注 2**
			400 kHz 模式	100	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	4.7	—	μs	
			400 kHz 模式	0.6	—	μs	
109	TAA	时钟输出有效时间	100 kHz 模式	—	3500	ns	**注 1**
			400 kHz 模式	—	1000	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	μs	在新的传输开始前总线必须处于空闲状态的时间
			400 kHz 模式	1.3	—	μs	
D102	Cb	总线的容性负载		—	400	pF	

注　1： 作为发送器，为避免意外产生启动或停止条件，器件必须提供此内部最小延迟时间，以补偿 SCL 下降沿的未定义区域（最小值 300 ns）。

2： 快速模式的 I²C 总线器件可以在标准模式 I²C 总线系统中使用，但必须满足 TSU:DAT ≥ 250 ns 的要求。如果快速模式器件没有延长 SCL 信号的低电压周期，则必然满足此条件。如果该器件延长了 SCL 信号的低电压周期，则它必须将下一个数据位输出到 SDA 线，
根据标准模式 I²C 总线规范，释放 SCL 先之前的时间为：TRmax + TSU: DAT = 1000 + 250 = 1250 ns。

附录 B： CAN 概述

本附录提供了控制器局域网 （CAN）总线的概述。在本参考手册的 CAN 一章中讨论了用于 dsPIC30F 硬件模块的 CAN 协议的实现过程。

B.1 CAN 总线背景知识

控制器局域网（CAN）是串行通信协议，它能有效支持高性能的分布式实时控制。Robert Bosch GmbH 在 1991 年的 CAN 规范 V2.0B 中完整定义了 CAN 协议。

CAN 应用涵盖从高速网络到低成本多路复用线路的各种领域。汽车电子设备 （即，引擎控制单元、传感器和防滑系统等）均是使用 CAN 以最大 1 Mb/s 的比特率连接的。CAN 网络可以用来取代汽车中的线路连接以有效节约成本。CAN 总线在噪声环境中的可靠性及其故障状态检测和从故障状态恢复的能力使其适用于 DeviceNet、 SDS 和其他现场总线协议等工控应用。

CAN 是具有一条逻辑总线线路的异步串行总线系统。它具有开放式线性总线型网络结构，总线节点平均分布。CAN 总线由两个或多个节点组成。总线上的节点数量可以动态改变而不会干扰其他节点的通信。这使得总线节点可方便地连接和断开 （如为了增加系统功能、错误恢复或总线监测）。

总线逻辑采用“线与”机制，“隐性”位（通常为逻辑 1 但非必须如此）被“显性”位覆盖（通常逻辑电平为 0）。只要没有总线节点在发送显性位，总线线路就处于隐性状态，但是只要任一总线节点发送了显性位，就会产生显性总线状态。因此，对于 CAN 总线线路，必须选择能传送两种可能位状态（显性和隐性）的传输介质。其中最常见也最便宜的方法是使用双绞线。此时总线线路被称为“CANH”和“CANL”，并且可以直接或通过连接器与节点相连。 CAN 协议中没有定义有关连接器使用的标准。双绞线的端接是通过在总线线路两端连接端接电阻完成的。当总线长度不超过 40 米时，最大总线速度可达 1 Mb。如果总线长度超过 40 米，总线速度必须降低（将总线速度降到 40 Kb 可使总线长度达到 1000 米）。对于 1000 米以上的总线，应该使用特殊的驱动器。无需额外设备，即至少可以连接 20 个节点。由于发送的差分性，即两条总线线路受 EMI 的影响相同，使差分信号不受影响，所以 CAN 本质上并不容易受到辐射电磁能的影响。也可以屏蔽总线线路以降低来自总线的电磁辐射，尤其是在高波特率下。

二进制数据是按 NRZ 码（非归零码，低电平 = 显性状态，高电平 = 隐性状态）编码的。使用位填充以确保所有总线节点时钟同步。这意味着在报文发送时，最多可以有 5 个连续位具有相同的极性。只要发送了相同极性的 5 个连续位，在继续发送后续位前，发送器将在比特流中插入一个极性相反的位。接收器也会检查同极性位的个数，并把填充位从比特流中删除 （解填充）。

在 CAN 协议中，不对总线节点寻址。地址信息包含在发送的报文中。这是通过标识符 （每个报文的组成部分）实现的，标识符标识了报文内容 （如引擎速度和油温等）。另外，标识符还会指出报文的优先级。标识符的二进制值越小，报文的优先级就越高。

对于总线仲裁，采用的是带有 NDA 的 CSMA/CD（带有非破坏性仲裁的载波侦听多路访问 / 冲突检测）。如果总线节点 A 要通过网络发送报文，它首先会检查总线是否处于空闲状态 （“载波侦听”）（即没有正在进行发送的节点）。如果是这样 （并且没有其他节点希望在此刻开始发送），节点 A 就成为总线主控节点并发送其报文。在节点 A 发送第一个发送位 （帧起始位）时，所有其他节点都会切换到接收模式。在正确接收了报文后 （每个节点均对报文进行了接收应答），每个总线节点都会检查报文标识符，如果需要的话还将存储报文。否则，就将报文丢弃。

如果两个或多个总线节点同时开始其发送 （“多路访问”），此时可通过按位仲裁 （“冲突检测 / 非破坏性仲裁”加上 “线与”机制和 “显性”位覆盖 “隐性”位）避免报文冲突。每个节点都发送报文的标识符位 （先发送 MSb）并监视总线电平。如果节点发送的是隐性标识符位但是读回的是显性标识符位，则它将失去总线仲裁并切换到接收模式。当与之发生冲突的节点的报文标识符有较低的二进制值 （显性状态 = 逻辑 0）时就会产生这种情况，即冲突节点发送的报文具有较高的优先级。这样，具有最高优先级报文的总线节点不需要花费时间重发这一报文就可以赢得仲裁。一旦总线返回到空闲状态，其他节点就会自动尝试重新发送。不允许不同的节点采用同一标识符来发送报文，因为这样的话仲裁可能会失败导致发生冲突和错误。

早期的 CAN 规范 （1.0、 1.2 和 2.0A 版）定义报文标识符为 11 位长，从而给出了 2048 个种报文标识符。后来此规范进行了更新 （到 2.0B 版）后消除了这一限制。 CAN 规范 2.0B 版允许的可用报文标识符的长度为 11 和 / 或 29 位 （29 位长的标识符可提供超过 536,000,000 种不同的报文标识符）。CAN 2.0B版也被称为“扩展CAN”； 而1.0、1.2和2.0A版则被称为“标准CAN”。

B.2 不同的 CAN 实现方法

B.2.1 标准 CAN 和扩展 CAN

根据 CAN 规范 V2.0A，只包含 11 位标识符的数据帧和远程帧被称为标准帧。这些帧可以用来标识 2048 种不同的报文（标识符为 0 － 2047）。但是，优先级最低的 16 条报文（2032 － 2047）是保留不用的。根据 CAN 规范 V2.0B，扩展帧具有 29 位的标识符。如前所述， 29 位标识符是由 11 位标识符 （“基本 ID”）和 18 位扩展标识符 （“扩展 ID”）组成的。

CAN V2.0A 规定的 CAN 模块只能根据标准 CAN 协议发送和接收标准帧。发送和接收使用 29 位标识符的报文会产生错误。如果是 CAN V2.0B 规定的器件，另外还有一种区别。称为 “Part B Passive”的模块只能发送和接收标准帧，但是在遇到扩展帧时不会产生错误帧。称为 “Part B Active”的器件能够发送和接收标准帧和扩展帧。

B.3 基本 CAN 和完全 CAN

还有一个 CAN 特性与 CAN 模块和主机 CPU 之间的接口有关，该特性将 CAN 芯片分为 “基本 CAN”和 “完全 CAN”器件。这与所使用的协议 （标准或扩展 CAN）没有关系，可以允许在同一网络中同时使用基本 CAN 和完全 CAN 器件。

在基本 CAN 器件中，硬件只实现了协议的基本功能 （如比特流的产生和检查）。判断接收到的报文是否需要被存储（接收过滤）以及整个报文管理都必须由软件完成（即由主机 CPU 完成）。CAN 芯片通常只提供一个发送缓冲器和一到两个接收缓冲器。这样使用基本 CAN 模块的主机 CPU 负担会非常重，因此这些器件只能使用在低波特率和只允许几条不同报文同时传输的低总线负荷的场合使用。基本 CAN 的优势在于芯片体积小，可以降低器件成本。

完全 CAN 器件用硬件实现整个总线协议，包括接收过滤和报文管理。它们包含几个所谓的报文对象，可以处理标识符、数据、方向 （接收或发送）以及标准 CAN/ 扩展 CAN 信息。在器件的初始化过程中，主机 CPU 决定哪些报文要发送、哪些报文要接收。如果某个接收到的报文的标识符与已编程的 （接收）报文对象之一相匹配，中断就会通知主机 CPU。这样，就减轻了 CPU 的负担。使用完全 CAN 器件可以处理高波特率和允许很多报文同时传输的高总线负荷。但这些芯片比基本 CAN 器件昂贵。

许多完全 CAN 芯片还提供“基本 CAN 功能”。可以编程某个报文对象，使之将每个报文存储起来而不与其他报文对象相匹配。这对于许多应用都很有帮助。

B.4 ISO 模型

如图 B-1 所示，ISO/OSI 参考模型用于定义通信系统协议的层。在最高层，应用程序需要互相通信。在最低层，使用某些物理介质提供电信令。

协议的高层由软件控制。通常硬件只能实现到应用层。在 CAN 总线规范中，并没有对报文类型或被传输报文的内容及意义进行定义。这些内容在有些系统中定义，比如 Volcano （Volvo 汽车 CAN 规范）、 J1939 （美国重型卡车多路布线规范）以及 Allen-Bradly DeviceNet 和 Honeywell SDS （工业协议的两个范例）。

CAN 总线模块定义包含整个协议中的两个级别如下。

- 数据链路层
  - 逻辑链路控制 （LLC）子层
  - 介质访问控制 （MAC）子层
- 物理层
  - 物理信令 （PLS）子层

LLC 子层与报文过滤、过载通知和错误恢复管理有关。 LLC 子层的功能范围包括：

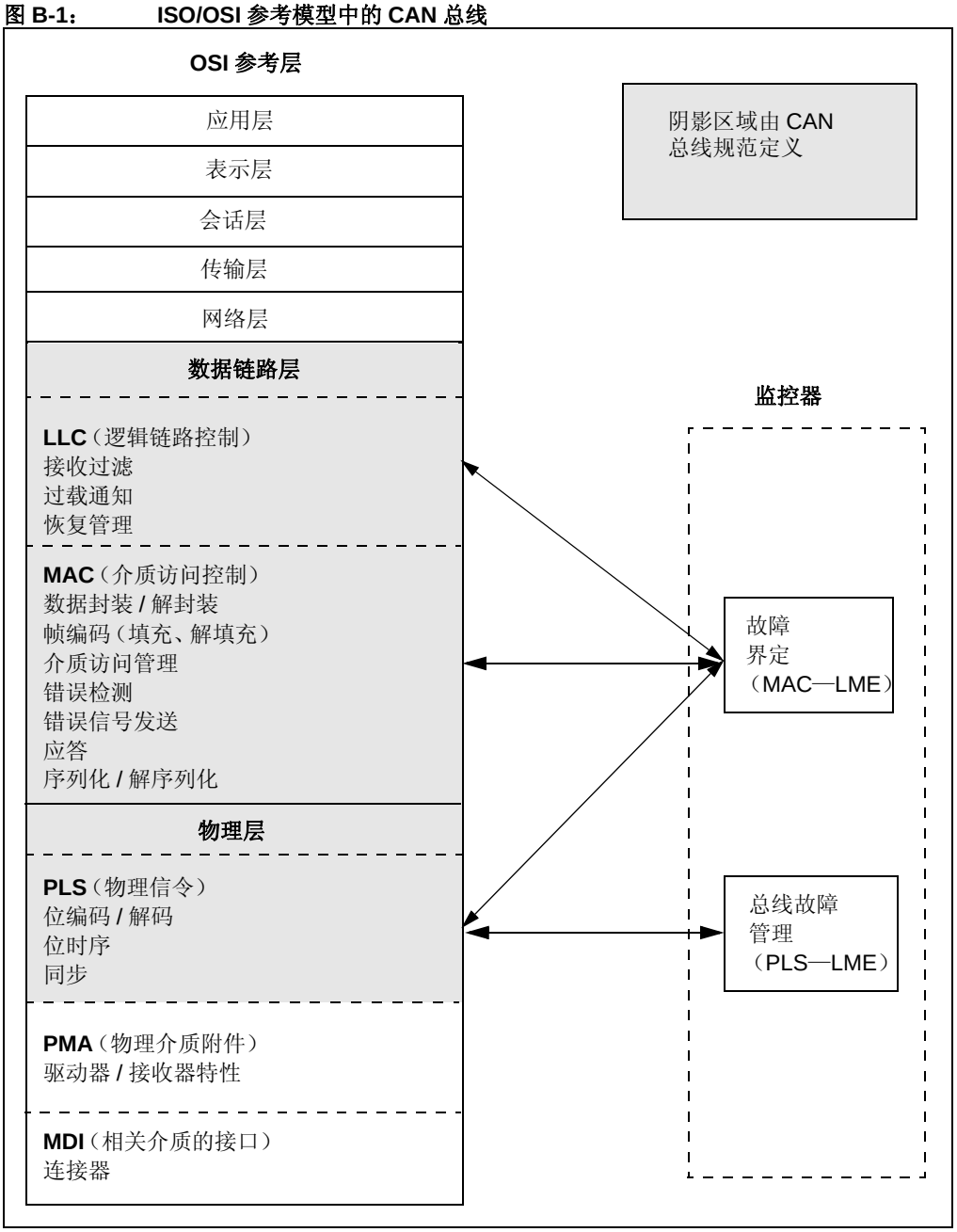
- 为数据传输和远程数据请求提供服务
- 决定 LLC 子层接收的哪些信息将被实际接收
- 提供错误恢复管理和过载通知的方式

MAC 子层是 CAN 协议的核心。 MAC 子层定义了传输协议 （即控制帧、执行仲裁、错误检测、错误信号产生以及故障界定）。该层提供从 LLC 子层的接收的报文，并接收将要发送到 LLC 子层的报文。判断总线是处于空闲状态以开始新的发送还是刚刚开始接收都是在 MAC 子层内完成的。MAC子层由一个叫做故障界定的管理实体监管，这是一种能够把短时干扰和永久失效区分开来的自检机制。而且，位时序的某些通用功能也被认为是 MAC 子层的一部分。

物理层定义了所有电气属性下不同节点之间的实际的位传输。 PLS 子层定义信号的实际发送方式，并由此对位时序、位编码和同步进行描述。

协议的较低层在驱动器 / 接收器芯片和实际接口（如双绞线或光纤等）中实现。在同一个网络中，所有节点必须使用同一个物理层。物理层的驱动器 / 接收器特性没有定义，这样可使发送介质和信令的实现根据应用进行优化。 ISO11898 道路车辆多路布线规范 （英文）中对最常见的例子给出了定义。

图 B-1：　　ISO/OSI 参考模型中的 CAN 总线



B.5 CAN 总线功能

CAN 具有以下属性：

- 报文优先级控制
- 确保响应时间
- 配置灵活
- 带时间同步的组播接收
- 全系统数据一致性
- 多主机
- 错误检测和信令
- 损坏报文的自动重发
- 区分节点的暂时错误和永久无效并自动关闭出错节点

1. 报文：总线上的信息是以固定格式的报文形式发送的，报文的长度不同但有限制。当总线空闲时，任何连接到总线的部件都可以开始发送新的报文。
2. 信息路由：在 CAN 系统中， CAN 节点不会使用系统配置 （如站点地址）的任何信息。
3. 系统灵活性：可以向 CAN 网络添加节点而无需在节点和应用层的软硬件上进行任何修改。
4. 报文路由：报文的内容是以标识符命名的。标识符没有指出报文的目的地，而是说明了数据的意义，因此网络中的所有节点都能够通过报文过滤判断它们是否需要对数据进行处理。
5. 组播：作为报文过滤的一个结果，任意数量的节点都能接收并同时处理一个报文。
6. 数据一致性：在 CAN 网络中确保报文被所有节点同时接收或不被任何节点接收。这样，就能通过组播和错误处理保证系统的数据一致性。
7. 比特率：在不同系统中 CAN 的速度可能会不同。但是，在一个给定的系统中，比特率是统一且固定的。
8. 优先级控制：标识符会在总线访问期间确定一个静态报文的优先级。
9. 远程数据请求：通过发送远程帧，请求数据的节点可要求另一个节点发送相应的数据帧。数据帧和相应的远程帧由相同的标识符命名。
10. 多主机：当总线空闲时，任何部件都可开始发送报文。具有较高优先级的待发送报文的部件会获得总线的访问权。
11. 仲裁：只要总线空闲，任何部件就可开始发送报文。如果两个或多个部件同时开始发送报文就需要使用标识符进行按位仲裁以解决总线访问冲突。仲裁机制可以确保不会丢失信息或损失时间。如果使用相同标识符的数据帧和远程帧同时开始发送，则数据帧优先于远程帧。在仲裁时，每个发送器都会将待发送位的优先级与在总线上监视到的优先级对比。如果优先级相等，则该发送器可继续发送。如果发送了“隐性”级别但是监视到“显性”级别，则该器件失去仲裁，必须撤回请求并不再发送位。
12. 安全性：为了实现数据传输的最高安全性，在每个 CAN 节点都实施了强大的错误检测、信令和自检机制。
13. 错误检测：为了检测错误，采取了以下措施：
    - 监视 （发送器将待发送位的优先级与在总线上监视到的优先级对比）
    - 循环冗余校验
    - 位填充
    - 报文帧校验

错误检测机制具有以下属性：

- 检测所有全局错误
- 检测发送器的所有本地错误
- 最大可检测到一个报文中的 5 个随机分布错误
- 检测报文中长度小于 15 的突发错误
- 检测报文中任何数量为奇数的错误。

14. 错误信令和恢复时间：任何检测到错误的节点都会标记损坏的报文。此类报文发送被中止并将自动重发。如果没有发生更多错误，从检测到错误到开始下一个报文最长需要 31 个位时间。
15. 故障界定：CAN 节点能够区别短期干扰和无效故障。出错的节点会被关闭。
16. 连接：CAN 串行通信链路是可连接一定数量器件的总线。理论上，此数量没有限制。实际上，器件总数受到总线时延和 / 或总线上的电气负载的限制。
17. 单通道：总线由载有位的单个通道组成。信息可通过经过该通道传输的数据再同步的过程获得。本规范中并不规定该通道的实现方式（即单线（接地）、两条差分线路和光纤等）。
18. 总线值：总线可以有两个互补的逻辑值，即 “显性”或 “隐性” 。同时发送 “显性”和 “隐性”位，总线值将是 “显性”的。例如，当执行总线的 “线与”操作时，逻辑 “0” 代表 “显性”，逻辑 “1”代表 “隐性”。在规范中未说明代表逻辑值的物理状态（例如电压和光）。
19. 应答：所有的接收器都会检查待接收报文的一致性，并会对一致的报文作出应答并标志不一致的报文。
20. 休眠模式及唤醒：要降低系统的功耗，可将 CAN 器件设置进入休眠模式，此模式下没有任何内部活动并且其总线驱动器也未与总线相连。任何总线活动或系统内部条件唤醒器件，休眠模式都会结束。在唤醒时，内部活动重新开始，但是 MAC 子层将等待系统振荡器稳定下来，并在总线驱动器再次置为 “在线”前，继续等待直到其状态与总线活动同步（通过检测 11 个连续的 “隐性”位）。

B.6 帧类型

B.6.1 标准数据帧

当节点希望发送数据时它会产生一个数据帧。图 B-2 所示为标准 CAN 数据帧。与其他所有帧相同，数据帧也是以帧起始 （SOF，显性状态）位开始与所有节点进行硬同步。

在 SOF 之后是仲裁字段，由 12 位组成，包括 11 位的标识符 （反映报文的内容和优先级）和 RTR 位 （远程传输请求位）。 RTR 位用于区分数据帧 （RTR，显性）和远程帧。

下一个字段是控制字段，由 6 个位组成。字段的第一位称为 IDE 位 （标识符扩展位），该位为显性状态，指定该帧为标准帧。接下来是保留位 RB0，这一位也被定义为显性位。控制字段的其余 4 位为数据长度码 （DLC），它规定了报文中包含的数据字节数。

控制字段之后为数据字段，包含正在发送的数据字节，数据字段长度由上述数据长度码 DLC 定义（1 到 8 字节）。

数据字段之后是循环冗余字段（CRC），用来检测可能的报文传输错误。CRC 字段由一个 15 位的 CRC 序列组成，以隐性 CRC 定界符位结束。

最后是应答字段。在应答间隙位 （ACK Slot bit）产生期间，发送节点发出一个隐性位。任何收到无错误帧的节点会发回一个显性位 （无论该节点是否配置为接收该特定报文），确认帧已正确接收。从这一点上可以看出， CAN 属于 “位回应”类 （In-bit-response）协议。应答间隙以隐性确认定界符结束，该字符不能被显性位改写。

B.7 扩展数据帧

如图 B-3 所示，在扩展 CAN 数据帧中，紧随帧起始 （SOF）位的是 38 位仲裁字段。仲裁字段的前 11 位为 29 位标识符的 11 个最高位 （基本 ID）。紧随这 11 位的是代理远程请求 （SRR）位，它以隐性状态发送。 SRR 位后是 IDE 位，该位是隐性的，表示这是扩展的 CAN 帧。值得注意的是，如果在扩展帧标识符的前 11 位发送完后，总线仲裁无果，而此时参与仲裁的某个节点发出标准 CAN 帧 （11 位标识符），那么由于该节点发出了显性的 IDE 位而使标准 CAN 帧赢得仲裁。另外，扩展 CAN 帧的 SRR 位应为隐性，以允许正在发送标准 CAN 远程帧的节点发出显性 RTR 位。 SRR 位和 IDE 位之后是标识符的其余 18 位 （扩展 ID）以及一个远程发送请求位。

为使标准帧和扩展帧都能在它们共享的网络上发送，应将 29 位的扩展报文标识符拆分成 11 位（最高位）和 18 位 （最低位）两部分。拆分后可确保标识符扩展位 （IDE）在标准帧和扩展帧中各位的位置保持不变。

下一个字段是控制字段，由 6 个位组成。控制字段前两位为保留位，为显性状态。控制字段的其余 4 位为数据长度码 （DLC），它规定了数据字节数。

扩展数据帧的其他部分 （数据字段、 CRC 字段、应答字段、帧结束和间断）在结构上与标准数据帧相同。

B.8 远程帧

通常，数据发送是由数据源节点自主完成的（例如，传感器发送数据帧）。但也可能发生宿节点向源节点请求发送数据的情况。要做到这一点，宿节点需要发送一个标识符与所需数据帧的标识符相匹配的“远程帧”。随后相应的数据源节点会发送一个数据帧作为对该远程请求的响应。

如图 B-4 所示，远程帧与数据帧有两点不同。第一，远程帧的 RTR 位为隐性状态；第二，远程帧没有数据字段。带有相同标识符的数据帧和远程帧同时发送的情况是很少出现的，这种情况下数据帧将赢得仲裁，这是因为其标识符之后的 RTR 位为显性。这样可使发送远程帧的节点立即收到所需数据。

B.9 错误帧

错误帧是由检测到总线错误的任何节点产生的。如图 B-5 所示，错误帧包含 2 个字段，即错误标志字段和紧随其后的错误定界符字段。错误定界符由 8 个隐性位组成，可以让总线节点在错误发生后立即重新启动总线通信。错误标志字段有两种形式。其具体形式取决于检测到错误的节点的错误状态。

当错误主动节点检测到一个总线错误时，这个节点将通过产生一个主动错误标志，中断当前的报文发送 。主动错误标志由 6 个连续的显性位构成。这种位序列有效地打破了位填充规则。所有其他站点在识别到由此产生的位填充错误后，反过来也会产生错误帧，称为错误回送标志。错误标志字段因此包含 6 到 12 个连续显性位 （由 1 个或多个节点产生）。错误帧以错误定界符字段结束。在错误帧发送完毕后，总线活动恢复正常状态，被中断的节点会尝试重新发送被中止的报文。

当错误被动节点检测到一个总线错误时，该节点将发送一个错误被动标志，后面仍然紧随错误定界符字段。错误被动标志由 6 个连续隐性位组成，因此错误被动节点的错误帧由 14 个隐性位组成。由此可知，除非总线错误被正在发送报文的总线主节点检测到，否则错误被动节点发送的错误帧将不会影响网络中任何其他节点。如果总线主节点产生了一个错误被动标志，那么由于位填充规则被打破，将导致其他节点产生错误帧。错误帧发送完毕后，错误被动节点必须等待总线上出现 6 个连续隐性位后，才能尝试重新参与总线通信。

B.10 帧间间隔

帧间间隔将前一个帧（无论何种类型）与其后的数据帧或远程帧分隔开来。帧间间隔至少由 3 个隐性位构成，也称为间断。间断使节点在开始发送 / 接收下一个报文帧之前有时间进行内部处理。在间断之后， CAN 总线将保持隐性状态 （总线空闲），直至开始发送下一个帧。

图 B-2： 标准数据帧

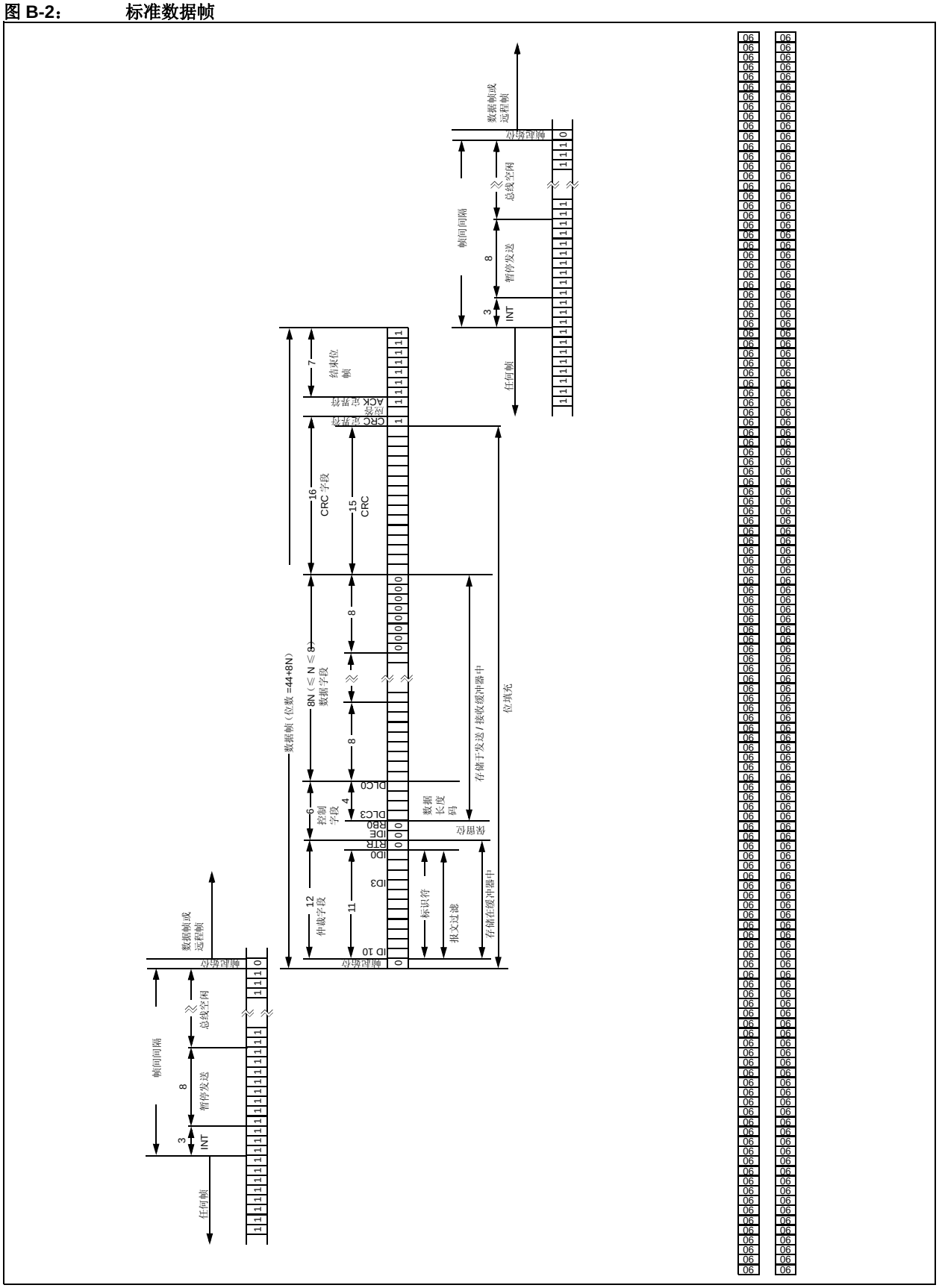


图 B-3：　　扩展数据格式

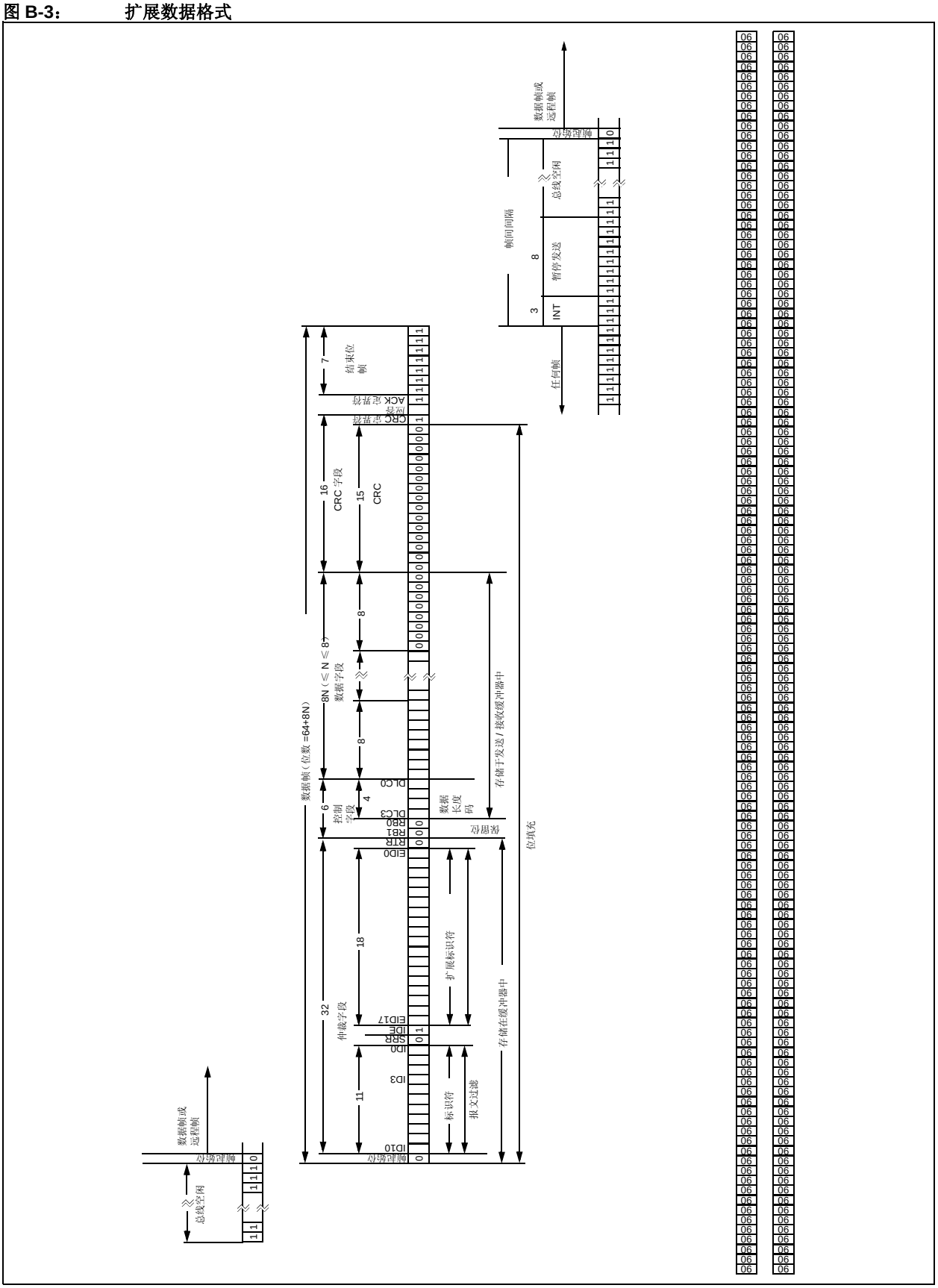

数据帧或远程帧
帧起始位
总线空闲
任何帧
INT
暂停发送
帧间间隔
数据帧（位数 =64+8N）
仲裁字段
控制字段
数据字段
CRC 字段
CRC
CRC 定界符
ACK 间隙
ACK 定界符
结束位帧
扩展标识符
标识符
报文过滤
存储在缓冲器中
存储于发送 / 接收缓冲器中
保留位
数据长度码
位填充

图 B-4：　　远程数据帧

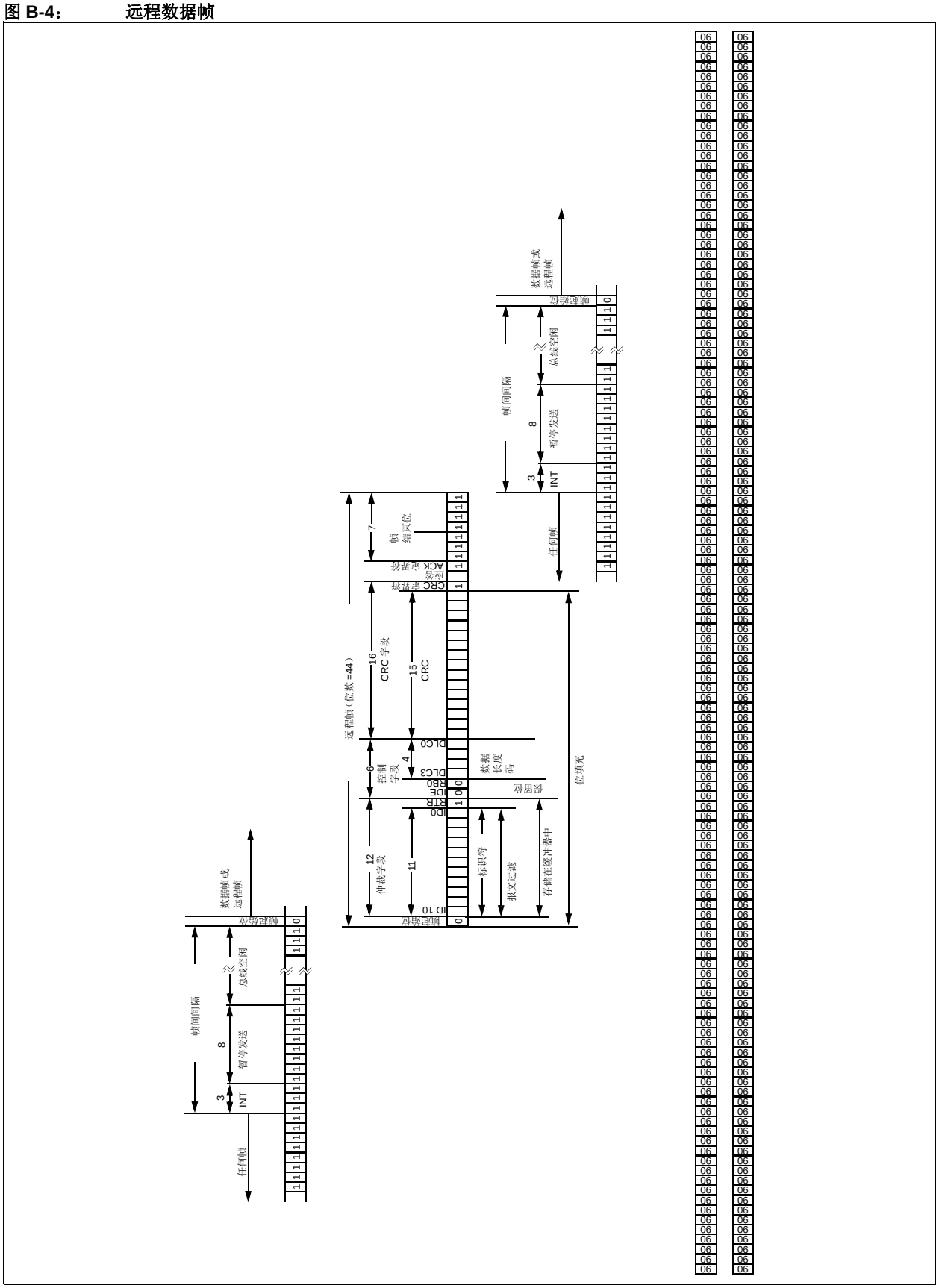
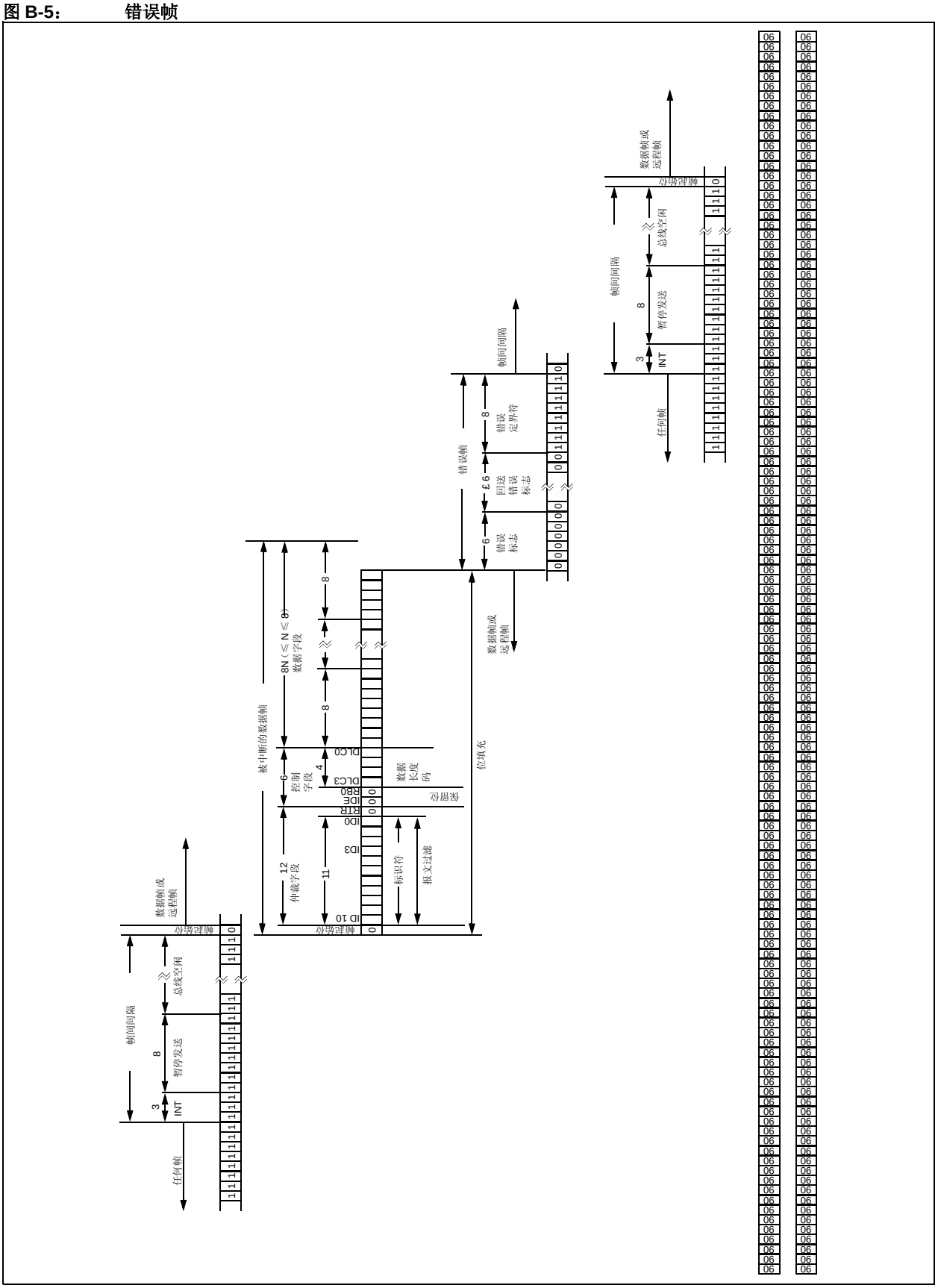


图 B-5：　　错误帧


帧间间隔
3
INT
8
暂停发送
总线空闲
数据帧或
远程帧
任何帧
帧起始
帧间间隔
3
INT
8
暂停发送
总线空闲
数据帧或
远程帧
任何帧
帧起始
帧间间隔
错误
定界符
8
回送
错误
标志
6
错误
标志
6
错误帧
数据帧或
远程帧
位填充
被中断的数据帧
8
8
DLC0
DLC3
4
控制
字段
6
保留位
数据
长度
码
RB0
IDE
RTR
ID0
ID3
ID 10
11
12
仲裁字段
标识符
报文过滤
帧起始
0

B.11 参考文档

标题	文档
Road Vehicles; Interchange of Digital Information, Controller Area Network	ISO11898
Bosch CAN Specification Version 2.0	

附录 C： 编解码器协议概述

本附录概括了 Inter-IC Sound （I²S）和 AC-Link 兼容模式接口的音频编码器 / 解码器 （编解码器，Codec）协议。很多倾向于在音频应用中使用的编解码器支持 8 kHz 到 48 kHz 的采样率，通常使用前面提到的接口协议中的一种。数据转换器接口 （DCI）模块自动处理与这些编解码器有关的接口时序。在 DCI 发送和 / 或接收被请求的数据量前，无需 CPU 开销。在两次 CPU 中断之间最多可以传输 4 个数据字。

C.1 I²S 协议说明

Inter-IC Sound （I²S）是一种简单的三线总线接口，用于在以下设备之间传输数字音频数据：

- DSP 处理器
- A/D 和 D/A 转换器
- 数字输入 / 输出接口

本附录信息旨在对 Philips, Inc 发布的 I²S Protocol Specification® 作一个补充。

I²S 总线是时分复用的并可传输两个通道的数据。这两个数据通道通常是数字音频流的左右声道。

I²S 总线具有以下连接引脚：

- SCK：I²S 串行时钟线
- SDx：I²S 串行数据线 （输入或输出）
- WS：I²S 字选择线

图 C-2 所示为数据传输的时序图。串行数据在 I²S 总线上是以首先发送 MSb 的 2 的补码格式发送的。因为此协议允许发送和接收不同数据字长度，所以必须先发送 MSb。如果接收器收到了比它能接受的数据字更多的位，则将 LSb 忽略。如果接收器收到了比原始字长更少的位，则它必须在内部将余下的 LSb 置为 0。

WS 线指出正在发送的数据通道。使用以下标准：

- WS = 0：通道 1 或左声道
- WS = 1：通道 2 或右声道

WS 线由接收器在 SCK 的上升沿采样，下一个数据字的 MSb 在 WS 改变后的下一个 SCK 周期发送。 WS 改变后一个周期的延时可以让接收器有时间存储以前发送给它的字，并准备好接收下一个字。发送器发送的串行数据在 SCK 的下降沿放到总线上，并在 SCK 的上升沿由接收器锁存。

任何器件都可以在 I²S 系统中充当系统主机。系统主机产生 SCK 和 WS 信号。通常，由发送器充当系统主机，但是接收器或其他器件也可以作为主机。图 C-1 显示了可能的 I²S 总线配置。虽然在图 C-1 中没有特别说明，但两个相连的器件可以有数据发送和数据接收两种连接方式。

图 C-1：　　I²S 总线连接

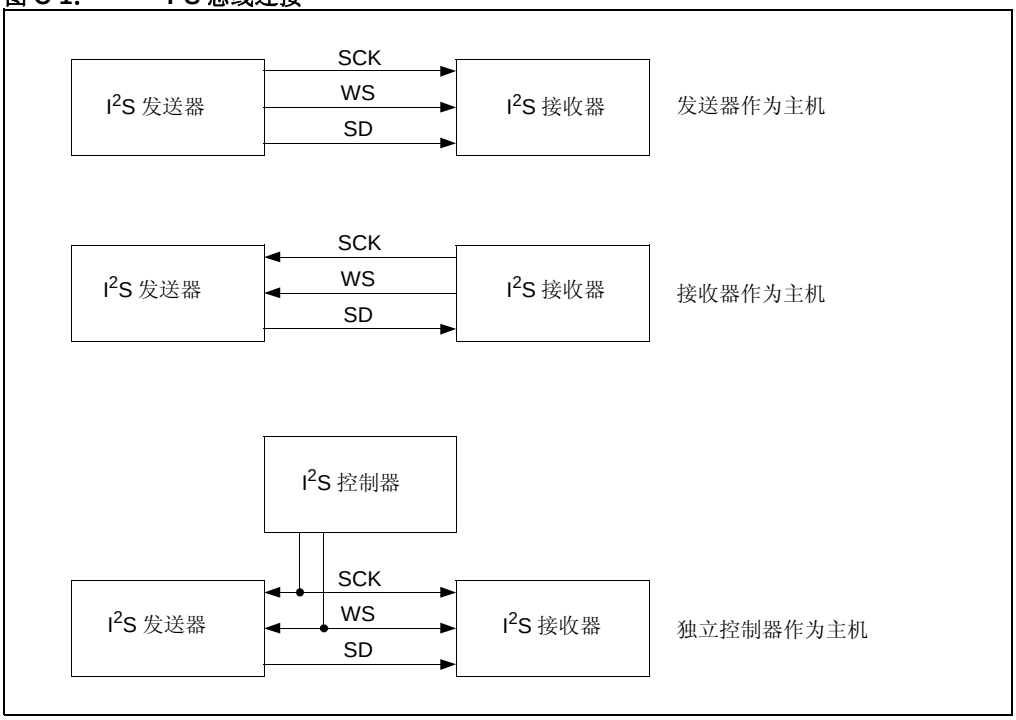
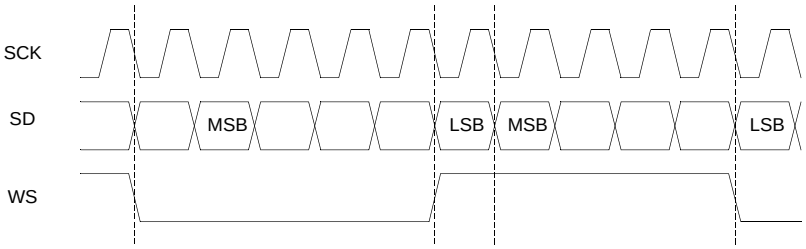


图 C-2：　　I²S 接口时序图



注：为便于说明，本图显示了字长为 5 位的数据的传输情况。I²S 协议不指定字长（字长取决于系统）。

C.2　AC’ 97 协议

音频编解码器’ 97 （Audio Codec '97， AC '97）规范定义了用于 PC 平台的音频编解码器的标准架构和数字接口协议。符合 AC '97 的编解码器的数字接口协议称为 AC-Link，它是本文讨论的重点。本文不说明 AC '97 控制器设备的具体要求和功能。

本附录信息旨在对 Intel 公司发布的 AC '97 组件规范文档作一个补充。

C.3　AC-Link 信号说明

所有 AC-Link 信号都是由 AC '97 主时钟源产生的。建议使用连接到 AC '97 编解码器的时钟源为 24.576 MHz 晶振，以使时钟抖动减至最小。24.576 MHz 时钟也可以由 AC'97 控制器或外部源提供。

所有的 AC-Link 信号名称都涉及到 AC '97 控制器，而非 AC '97 编解码器。控制器是产生 SYNC 信号以启动数据传输的器件。下面各节对每个信号进行了说明。

C.3.1 位时钟 （BIT_CLK）

系统中的主 AC '97 编解码器中提供一个 12.288 MHz BIT_CLK 信号。BIT_CLK 信号是 AC '97 控制器和系统中最多三个从动 AC '97 编解码器设备的输入信号。所有 AC-Link 数据在 BIT_CLK 的上升沿传送，并且由接收设备在 BIT_CLK 的下降沿采样。

C.3.2 串行数据输出 （SDO）

SDO 是发送到 AC '97 编解码器的时分复用数据流。

C.3.3 串行数据输入 （SDI）

SDI 是 AC '97 编解码器发出的时分复用数据流。

C.3.4 SYNC

SYNC 是由 AC '97 控制器发送到 AC '97 编解码器的 48 kHz 固定速率的采样同步信号。SYNC 信号是通过将 BIT_CLK 信号进行 256 分频得到的。SYNC 信号的电平保持 16 个 BIT_CLK 周期然后变为低电平并保持 240 个 BIT_CLK 周期。SYNC 信号只在 BIT_CLK 的上升沿改变，并且其周期确定了一个音频数据帧的边界。

C.3.5 复位

$\overline{\text{RESET}}$ 信号是系统中每个 AC '97 编解码器的输入信号，它会复位编解码器硬件。

C.4 AC-Link 协议

C.4.1 AC-Link 串行接口协议

AC-Link 串行数据流采样采用 256 位数据帧的时分复用 （TDM）机制。每个数据帧都分成 13 个时隙，编号为 Slot #0 到 Slot #12。Slot #0 是一个特殊时隙，包含 16 个位。其余 12 个时隙都是 20 位宽的。

图 C-4 显示了一个 AC-Link 帧示例。该帧在 SYNC 信号的上升沿（与 BIT_CLK 的上升沿重合）开始传输。AC '97 编解码器在紧随在该上升沿之后的 BIT_CLK 下降沿采样 SYNC 信号。此下降沿标记编解码器和控制器发现新帧起始位的时间。在 BIT_CLK 的下一个上升沿，编解码器产生 SDATA_IN 的 MSb 并产生 SDATA_OUT 的第一个边沿。此过程可以确保接收和发出的数据流的数据电平跳变点和随后的采样点在时间上是对齐的。

Slot #0、Slot #1 和 Slot #2 在 AC-Link 协议中对状态和控制有特殊用途。其余时隙都分配给某些特定类型的数字音频数据。Slot #3 到 Slot #12 的数据分配取决于所选择的 AC '97 编解码器，因此这里只能简要说明时隙用法。如需更多时隙用法的信息，请参见 AC '97 组件规范。

C.4.2 Slot #0，标记帧

Slot #0 通常称为 “标记帧”（Tag Frame）。标记帧对 AC-Link 协议的每个数据时隙都有一个对应的位位置。这些位用于指定帧中的哪些时隙可供控制器使用。Slot #0 特定位位置中的“1”表示当前音频帧中对应的时隙已经分配给一个数据流并且此时隙包含有效数据。如果某个间隙标记为无效，则应由数据源（对于输入流为 AC '97 编解码器、对于输出流为 AC '97 控制器）在该时隙的有效时间内用 0 填充所有位置。

在标记帧中也有特殊位。对于 SDATA_OUT，标记帧的 MSb 是 “帧有效”（Frame Valid）状态位。帧有效位是编解码器的全局指示符，表明在帧里至少有一个时隙包含有效数据。如果整个帧都标记为无效，编解码器会忽略标志帧后续的所有时隙。此功能用于实现非 48 kHz 的采样率。

SDATA_OUT 标记帧的两个 LSb 指出编解码器的地址。系统中最多可连接四个 AC '97 编解码器。如果系统只使用了一个编解码器，这些位保持为 0。

SDATA_IN 的 MSb 是“编解码器就绪”（Codec Ready）状态位。如果此位为 0，则表示编解码器已关闭和 / 或尚未就绪。如果 “编解码器就绪”位置位，则应由控制器查询编解码器中的状态寄存器，以确定哪个编解码器可使用。

C.4.3 Slot #1（命令地址）和 Slot #2（命令数据）

Slot #1 和 Slot #2 在 AC-Link 协议中也有特殊用途。当读取或写入 AC '97 编解码器控制寄存器时，这些时隙用于放置地址和数据值。这些时隙必须在 Slot #0 中标记为有效才能读写控制寄存器。AC '97 组件规范允许在编解码器中存在 64 个 16 位控制寄存器。在 AC-Link 协议中提供了 7 个地址位，但是只使用偶数地址位。奇数地址位的值是保留的。

SDATA_OUT 线的 Slot #1 和 Slot #2 分别称为命令地址和命令数据。SDATA_OUT 线上的命令地址间隙用于指定编解码器寄存器地址，并指定寄存器访问是读取还是写入。SDATA_OUT 上的命令数据时隙包含将要写入编解码器的某个控制寄存器的 16 位值。如果对编解码器寄存器执行了读操作，命令数据位将设置为零。

SDATA_IN 线的 Slot #1 和 Slot #2 分别称为状态地址和状态数据时隙。状态地址时隙回送先前发送到编解码器的寄存器地址。如果该值为 0，表示先前向编解码器发送的是无效地址。

状态地址时隙还有 10 个时隙请求位。时隙请求位可以由编解码器控制用于具有可变采样率的应用。

状态数据时隙返回从编解码器控制 / 状态寄存器读取的 16 位数据。

C.4.4 Slot #3（PCM 左声道）

SDATA_OUT 信号中的 Slot #3 用于合成的数据音频左声道回放流。对于声卡应用，该回放流通常是复合的 .WAV 音频和 MIDI 合成器输出相结合而成的。

SDATA_IN 信号中的 Slot #3 是从 AC '97 编解码器输入混音器获取的左声道录音数据。

C.4.5 Slot #4（PCM 右声道）

SDATA_OUT 信号中的 Slot #4 用于组合的数字音频右声道回放流。对于声卡应用，该回放流通常是复合的 .WAV 音频和 MIDI 合成器输出。

SDATA_IN 信号中的 Slot #4 是从 AC '97 编解码器输入混音器获取的右声道录音数据。

C.4.6 Slot #5 （调制解调器线 1）

SDATA_OUT 信号中的 Slot #5 用于调制解调器 DAC 数据。兼容调制解调器的 AC '97 编解码器的缺省精度为 16 位。与所有时隙一样，此时隙中未使用的位都置为 0。

SDATA_IN 信号中的 Slot #5 用于调制解调器 ADC 数据。

C.4.7 Slot #6

SDATA_OUT 信号中的 Slot #6 用于 4 声道或 6 声道声音配置中的 PCM 中间声道 DAC 数据。

SDATA_IN 信号中的 Slot #6 专门用于专用麦克风录音数据。此时隙中的数据允许在应用中使用话筒回声消除算法用于话筒应用中。

C.4.8 Slot #7

SDATA_OUT 信号中的 Slot #7 用于 4 声道或 6 声道声音配置中的 PCM 左声道 DAC 数据。

SDATA_IN 信号中的 Slot #7 在 AC '97 组件规范中保留供未来使用。

C.4.9 Slot #8

SDATA_OUT 信号中的 Slot #8 用于 4 声道或 6 声道声音配置中的 PCM 右声道 DAC 数据。

SDATA_IN 信号中的 Slot #8 在 AC '97 组件规范中保留供未来使用。

C.4.10 Slot #9

SDATA_OUT 信号中的 Slot #9 用于 6 声道声音配置中的 PCM LFE DAC 数据。

SDATA_IN 信号中的 Slot #9 在 AC '97 组件规范中保留供未来使用。

C.4.11 Slot #10 （调制解调器线 2）

Slot #10 用于调制解调器兼容设备中的调制解调器线 2 的 ADC 和 DAC 数据。

C.4.12 Slot #11 （调制解调器听筒）

Slot #11 用于调制解调器兼容设备中的调制解调器听筒的 ADC 和 DAC 数据。

C.4.13 Slot #12 （GPIO 控制 / 状态）

Slot #12 中的位用于读取和写入 AC '97 编解码器中的 GPIO 引脚。GPIO 引脚用于调制解调器兼容设备上的调制解调器控制功能。

图 C-3： AC-Link 信号连接

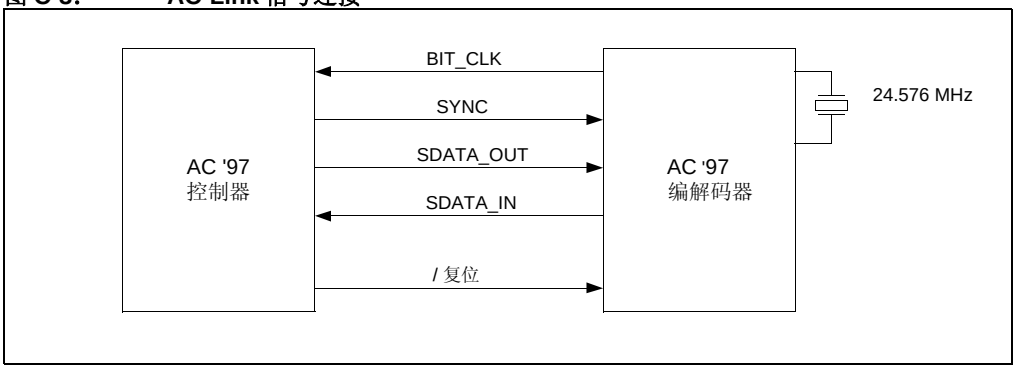


图 C-4： AC-Link 数据帧

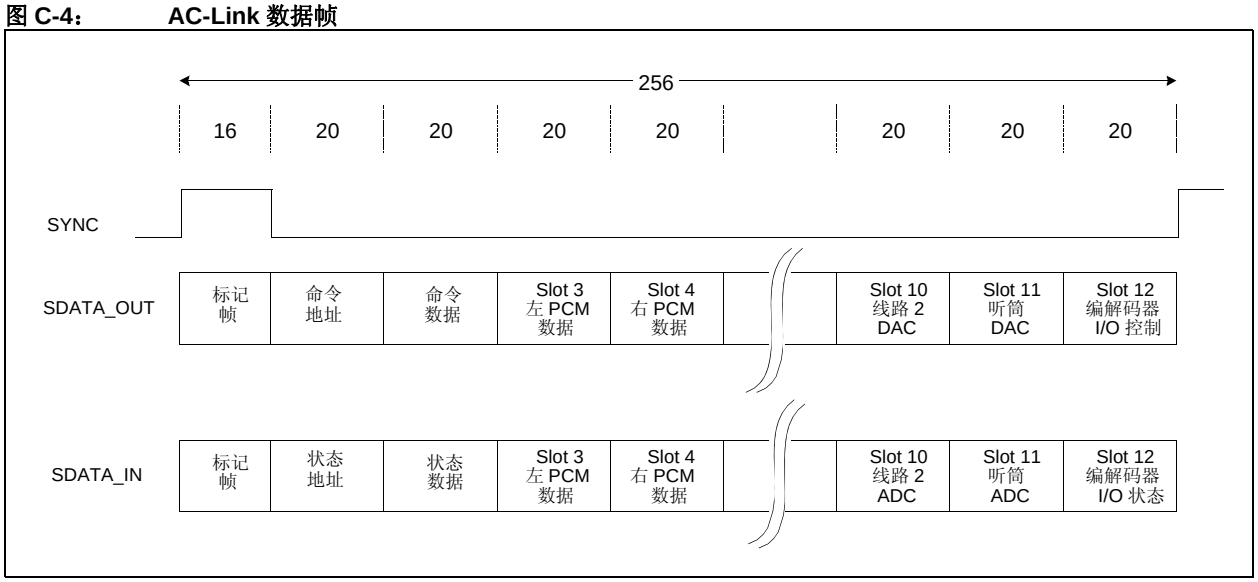


图 C-5： SLOT#0、 SLOT#1 和 SLOT#2 在 SDATA_IN 中的位置

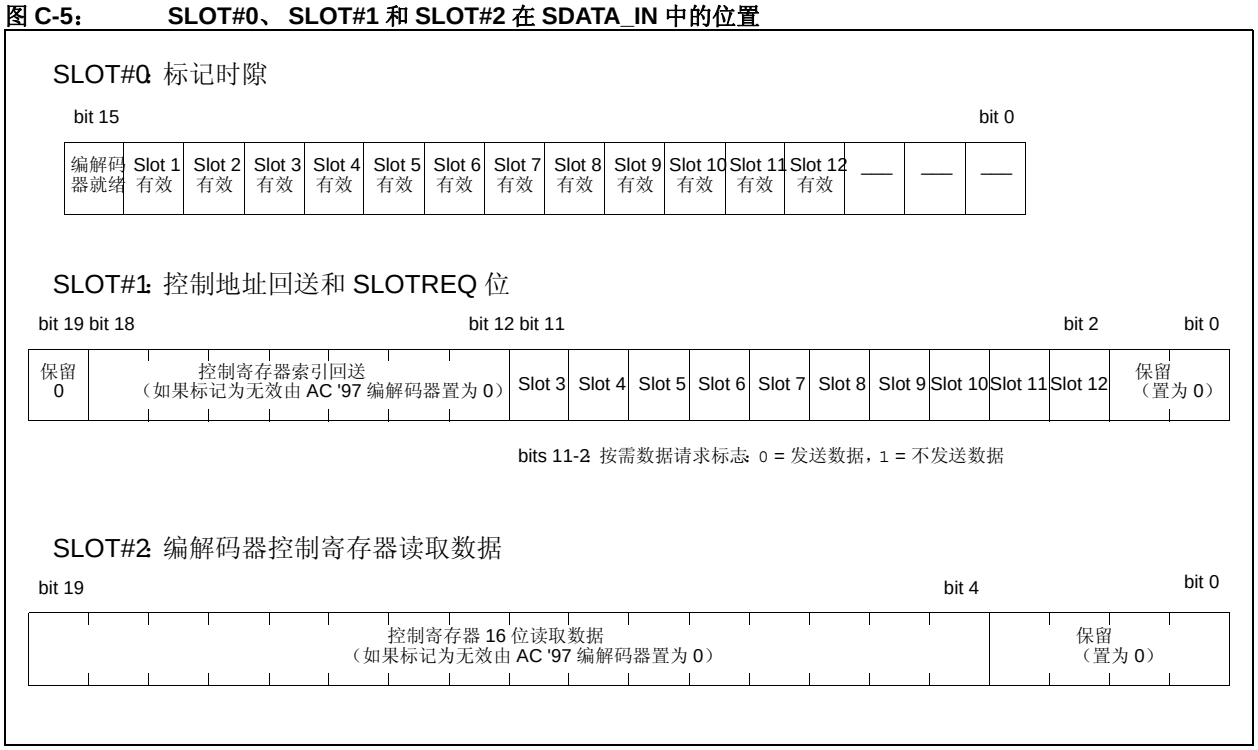
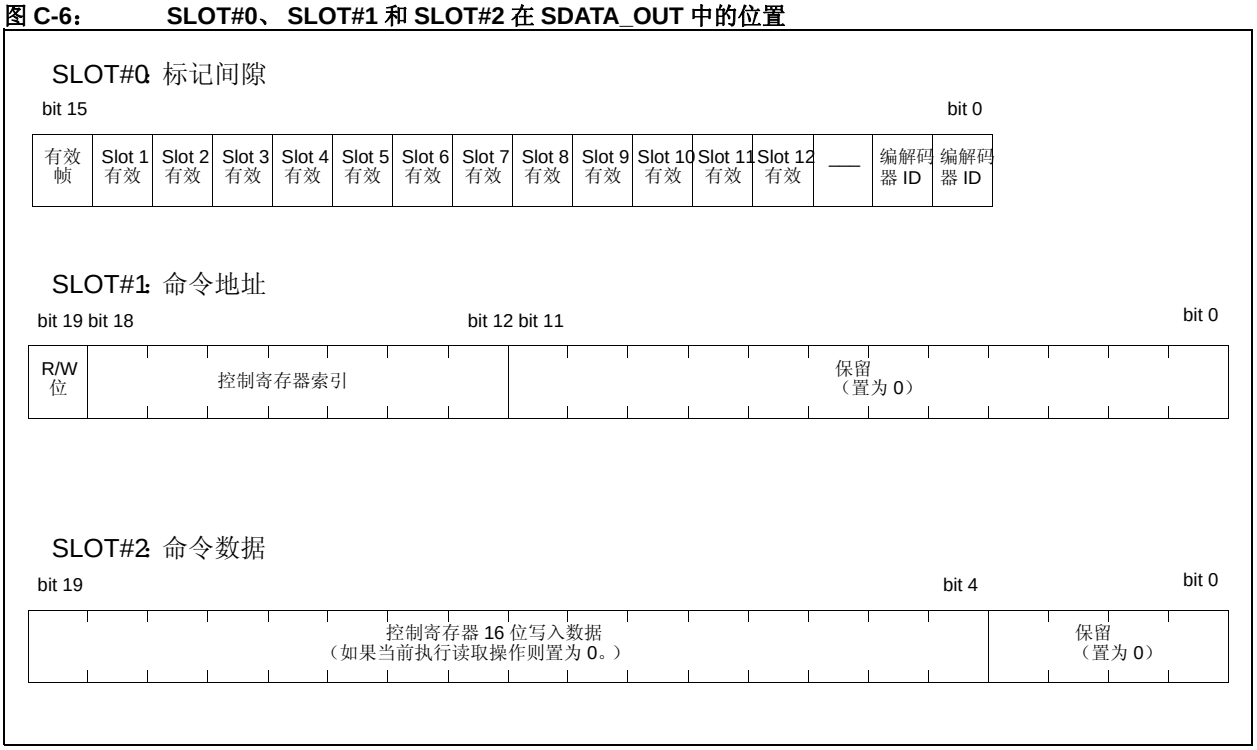


图 C-6： SLOT#0、 SLOT#1 和 SLOT#2 在 SDATA_OUT 中的位置



注：