“黑色经典”系列之《USB 应用开发技术大全》

第 3 章 USB 事务处理

本章目标

在 USB 协议中，USB 的数据传输由信息包组成，这些信息包组合起来可以构成完整的事务处理。USB 事务处理是 USB 主机和 USB 功能设备之间数据传输的基本单位。USB 的信息包和事务处理具有特定的格式。

本章主要讲解以下内容：

- USB 事务概述
- USB 字段格式
- USB 信息包
  - 令牌包
  - 数据包
  - 握手包
- USB 事务处理
- USB 总线列举

3.1 USB 事务概述

USB 事务处理是 USB 主机和 USB 设备数据通信的基础。一个完整的 USB 事务处理包含 3 个阶段，如图 3.1 所示，各个阶段的功能如下。

- 令牌阶段：其中定义了本次传输的类型，用于表征事务处理的开始。令牌阶段由同步字段、令牌包和 EOP 构成，这是所有的 USB 事务处理必须包含的阶段。
- 数据阶段：其中包含了本次传输的数据。其数据大小取决端点和传输类型，最大的数据量为 1024 字节。数据阶段由同步字段、数据包和 EOP 构成。
- 握手阶段：数据的接收方向发送方报告此次数据传输是否成功，握手阶段由同步字段、握手包和 EOP 构成。

不同的传输类型可能包含的事务处理阶段不同。

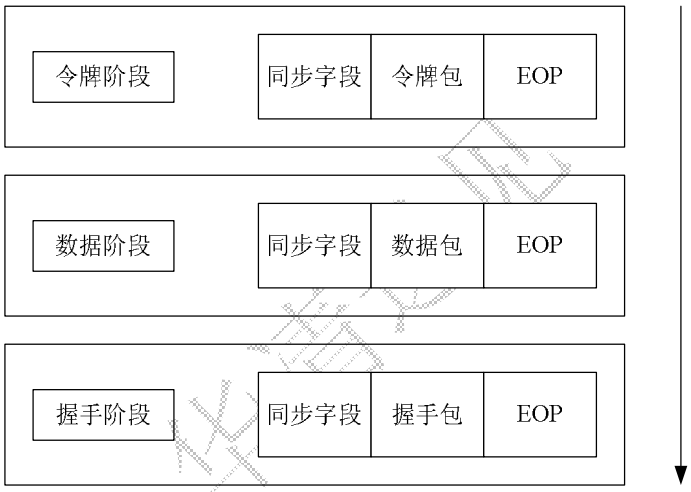

图 3.1 典型的事务处理过程

3.2 USB 字段格式

在 USB 协议中，USB 数据传输由一系列的字段构成，这些不同功能的字段，按照特定的格式组合便可以构成不同的信息包。USB 总线通信以信息包为基本传输单元进行 USB 事务处理。这些字段主要包括如下几种：

- 同步字段（SYNC）：用于数据通信的同步。
- 包标识字段（PID）：指明信息包类型，可以用于差错控制。
- 地址字段（ADDR）：指明 USB 总线上的一个 USB 设备。
- 端点字段（ENDP）：指明 USB 的端点。
- 帧号字段：指明当前帧的帧号。
- 数据字段：包含传输的数据。
- CRC 字段：循环冗余校验。

这里首先介绍组成信息包的这些主要 USB 字段的定义格式。

3.2.1 同步字段

由于 USB 主机和 USB 功能设备之间通过 USB 总线进行通信，USB 总线是两线的串行信号线，因此，通信的双方没有共同的时钟，这样很容易造成数据错位，导致数据不同步。为此，在 USB 协议中，使用同步字段进行所有信息包的同步。

1．低速/全速同步字段

对于低速/全速数据传输，同步序列的格式如图 3.2 所示。每个信息包以同步字段作为开始，同步字段长度为 8 位，数据为 10 000 000B。在数据总线上首先发送低位，然后发送高位。USB 总线数据采用 NRZI 编码，这样可以为发送方和接收方提供一个同步时钟，实现数据同步。

2．高速同步字段

高速数据传输的同步字段和低速/全速同步字段类似，但高速同步字段的长度为 32 位，数据为 80000000H，按照二进制位发送，即连续发送 31 个 0，最后发送一个 1。

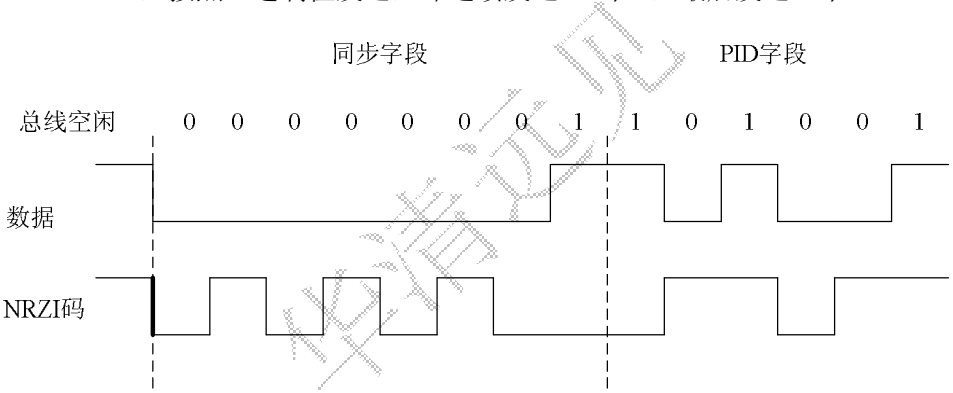

图 3.2 低速/全速的同步序列

3.2.2 包标识字段

包标识字段（PID）的定义格式如图 3.3 所示。包标识字段长度为 8 位，由低 4 位的类型字段和高 4 位的校验字段组成。这里，只有类型字段真正包含了该信息包的类型和格式信息，校验字段是类型字段的补码，接收方可以根据类型字段和校验字段共同确定数据传输的正确性。

D7	D6	D5	D4	D3	D2	D1	D0
/PID3	/PID2	/PID1	/PID0	PID3	PID2	PID1	PID0

图 3.3 包标识字段定义格式

包标识字段（PID）一般在同步字段之后，一般来说，PID 字段可以定义 4 种信息包类型：

- 令牌包：此时 PID[1:0]=01B。
- 握手包：此时 PID[1:0]=10B。
- 数据包：此时 PID[1:0]=11B。
- 特殊包：此时 PID[1:0]=00B。

1．令牌包

对于令牌包，可以在 PID 字段中指明数据传输方向和帧的开始等，主要的类型定义如下：

- 如果 PID[3:0]=0001B，则表示 OUT PID，此时数据从 USB 主机发送到 USB 设备。
- 如果 PID[3:0]=1001B，则表示 IN PID，此时数据从 USB 设备发送到 USB 主机。
- 如果 PID[3:0]=0101B，则表示 SOF PID，此时作为一个帧或者小帧的开始信息。
- 如果 PID[3:0]=1101B，则表示 SETUP PID，此时作为主机向 USB 设备发送的配置信息。

2．握手包

对于握手包，可以在 PID 字段中指明数据传输的结果，主要的类型定义如下：

- 如果 PID[3:0]=0010B，则表示 ACK PID，此时说明接收方已经正确接收到数据。
- 如果 PID[3:0]=1010B，则表示 NAK PID，此时说明接收方未能正确接收到数据。
- 如果 PID[3:0]=1110B，则表示 STALL PID，此时说明使用的端点被挂起。
- 如果 PID[3:0]=0110B，则表示 NYET PID，此时说明接收方没有任何响应。

3．数据包

对于数据包，可以在 PID 字段中指明数据包的奇偶性等，主要的类型定义如下：

- 如果 PID[3:0]=0011B，则表示 DATA0 PID，此时数据包为偶数据包。
- 如果 PID[3:0]=1011B，则表示 DATA1 PID，此时数据包为奇数据包。
- 如果 PID[3:0]=0111B，则表示 DATA2 PID，此时为一个高速同步事务的专用数据包。
- 如果 PID[3:0]=1111B，则表示 MDATA PID，此时为一个 SPLIT 事务的专用数据包。

4．特殊包

对于特殊包，可以定义一些特殊事务处理，主要的类型定义如下：

- 如果 PID[3:0]=1100B，则表示 PRE PID，如果是令牌信息，则为 USB 主机发送的先导包，用于使能 USB 低速数据通信。
- 如果 PID[3:0]=1011B，则表示 ERR PID，如果是握手信息，则用于 SPLIT 事务中的错误握手信号。
- 如果 PID[3:0]=1000B，则表示 SPLIT PID，此时为一个高速 SPLIT 事务的令牌信息。
- 如果 PID[3:0]=0100B，则表示 PING PID，此时用于数据流量检测控制。

3.2.3 地址字段

地址字段的定义格式如图 3.4 所示。地址字段的长度为 7 位，共有 128 个地址值。地址 0 作为缺省地址，不能分配给 USB 设备，因此，只有 127 个可分配的地址值。

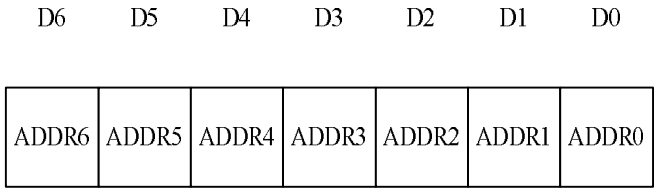

图 3.4　地址字段定义格式

在 USB 上电的时候，USB 主机首先通过缺省地址 0 和 USB 设备进行通信，当 USB 上电配置完成后，USB 主机便重新为该 USB 设备分配一个 USB 地址。

3.2.4　端点字段

端点字段的定义格式如图 3.5 所示。端点字段的长度为 4 位，总共可以表示 16 个端点。但是对于低速 USB 设备来说，USB 协议中只规定了 3 个端点；而对于高速/全速 USB 设备，则可以包含全部的 16 个端点。

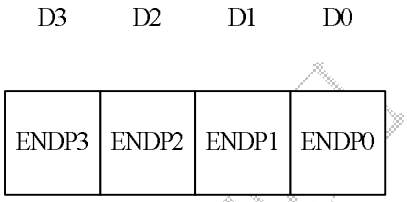

图 3.5　端点字段定义格式

在这些端点中，端点 0 是所有 USB 设备必须的。端点 0 主要用于在 USB 设备上电的初期和 USB 主机进行通信，从而完成 USB 设备的配置。

3.2.5　帧号字段

帧号字段的定义格式如图 3.6 所示。帧号字段的长度为 11 位，最大值为 07FFH，帧号字段里面的数值表征了当前帧或小帧的帧号。一般来说，在每个帧或小帧发送时候，帧号字段里的内容字段加 1。

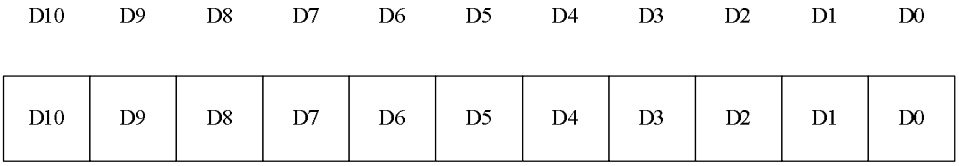

图 3.6　帧号字段定义格式

3.2.6　数据字段

数据字段的定义格式如图 3.7 所示。数据字段的长度最大为 1024 个字节。在数据传输的时候，首先传输低字节，然后传输高字节。对于每一个字节，先传输字节的低位，再传输字节的高位。实际的数据字段长度由传输类型和程序的需要决定。

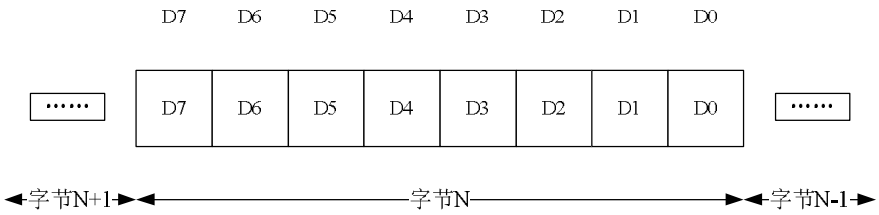

图 3.7　数据字段定义格式

3.2.7　CRC 字段

CRC 字段，即循环冗余校验字段。在 USB 协议中规定，循环冗余校验 CRC 一般在发送方的位填充操作之前进行，这样可以检验信息包的错误，保证传输的可靠性。CRC 字段主要在如下两种情况下使用：

- 令牌包：在令牌包中，一般采用 5 位循环冗余校验 CRC。
- 数据包：在数据包中，一般采用 16 位循环冗余校验 CRC。

3.3　USB 信息包

USB 数据传输中包括令牌包、数据包和握手包等 3 种信息包。

令牌包定义了本次传输的类型，用于表征事务处理的开始，令牌包由同步字段、令牌包数据和 EOP 构成。这是所有的 USB 事务处理必须包含的阶段。

数据包包含了本次传输的数据，其数据大小根据端点和传输类型而定，最大的数据量为 1024 字节。数据包由同步字段、数据包内容和 EOP 构成。

握手包用于数据的接收方向发送方报告此次数据传输是否成功，握手包由同步字段、握手包数据和 EOP 构成。

下面将分别介绍这些信息包的格式及分类。

3.4　令牌包

在 USB 协议中，使用了 7 种令牌包，按照其定义的格式，分为如下几种：

- IN、OUT、SETUP 和 PING 令牌包，这些令牌包格式大致相同。
- SOF 令牌包。
- SPLIT 令牌包。
- PRE 令牌包。

一般来说，这些令牌包都是由 USB 主机发送的，下面分别介绍各个令牌包的定义格式。

3.4.1　IN 令牌包

IN 令牌包的定义格式如图 3.8 所示。图中包含 8 位的包标识字段 PID、7 位的地址字段

ADDR、4 位的端点字段 ENDP 和 5 位的循环校验字段 CRC。

IN令牌包字段	PID	ADDR	ENDP	CRC
位数	8	7	4	5

图 3.8　IN 令牌包的定义格式

在 IN 令牌包中，各个字段的用途如下：

- PID 字段：定义了数据传输方向为 USB 设备到 USB 主机。
- ADDR 字段：指明了 USB 设备地址。
- ENDP 字段：指明了发送数据的端点号。
- CRC 字段：用于对 ADDR 字段和 ENDP 字段进行循环冗余校验。

3.4.2　OUT 令牌包

OUT 令牌包的定义格式如图 3.9 所示。图中包含 8 位的包标识字段 PID、7 位的地址字段 ADDR、4 位的端点字段 ENDP 和 5 位的循环校验字段 CRC。

OUT令牌包字段	PID	ADDR	ENDP	CRC
位数	8	7	4	5

图 3.9　OUT 令牌包的定义格式

在 OUT 令牌包中，各个字段的用途如下：

- PID 字段：定义了数据传输方向为 USB 主机到 USB 设备。
- ADDR 字段：指明了 USB 设备地址。
- ENDP 字段：指明了接收数据的端点号。
- CRC 字段：用于对 ADDR 字段和 ENDP 字段进行循环冗余校验。

3.4.3　SETUP 令牌包

SETUP 令牌包的定义格式如图 3.10 所示。图中包含 8 位的包标识字段 PID、7 位的地址字段 ADDR、4 位的端点字段 ENDP 和 5 位的循环校验字段 CRC。

SETUP令牌包字段	PID	ADDR	ENDP	CRC
位数	8	7	4	5

图 3.10　SETUP 令牌包的定义格式

在 SETUP 令牌包中，各个字段的用途如下：

- PID 字段：定义了数据传输方向为 USB 主机到 USB 设备。
- ADDR 字段：指明了 USB 设备地址。
- ENDP 字段：指明了接收数据的端点号。
- CRC 字段：用于对 ADDR 字段和 ENDP 字段进行循环冗余校验。

3.4.4 PING 令牌包

PING 令牌包的定义格式如图 3.11 所示。图中包含 8 位的包标识字段 PID、7 位的地址字段 ADDR、4 位的端点字段 ENDP 和 5 位的循环校验字段 CRC。

PING令牌包字段	PID	ADDR	ENDP	CRC
位数	8	7	4	5

图 3.11 PING 令牌包的定义格式

在 PING 令牌包中，各个字段的用途如下：

- PID 字段：定义了 USB 设备到 USB 主机的握手信号传输。
- ADDR 字段：指明了 USB 设备地址。
- ENDP 字段：指明了发送握手包的端点号。
- CRC 字段：用于对 ADDR 字段和 ENDP 字段进行循环冗余校验。

3.4.5 SOF 令牌包

SOF 令牌包的定义格式如图 3.12 所示。图中包含 8 位的包标识字段 PID、11 位的帧号字段和 5 位的循环校验字段 CRC。

SOF令牌包字段	PID	帧号字段	CRC
位数	8	11	5

图 3.12 SOF 令牌包的定义格式

在 SOF 令牌包中，各个字段的用途如下：

- PID 字段：定义了数据传输方向为 USB 主机到 USB 设备。
- ADDR 字段：指明了 USB 设备地址。
- ENDP 字段：指明了端点号。
- CRC 字段：用于对 ADDR 字段和 ENDP 字段进行循环冗余校验。

3.4.6 SPLIT 令牌包

在 USB 协议中，可以在 USB 主机和 USB 集线器之间进行高速数据传输的同时进行低速

或全速数据传输。SPLIT 令牌包便是用于这个目的。SPLIT 令牌包，包括开始 SPLIT（SSPLIT 令牌包）和结束 SPLIT（CSPLIT 令牌包），下面分别进行介绍。

1．SSPLIT 令牌包

启动令牌包 SSPLIT 的定义格式如图 3.13 所示。图中包含 8 位的包标识字段 PID、7 位的地址字段 ADDR、1 位的 SC 字段、7 位的 PORT 字段、1 位的 S 字段、1 位的 E 字段、两位的 ET 字段和 5 位的循环校验字段 CRC。

SSPLIT令牌包字段	PID	ADDR	SC	PORT	S	E	ET	CRC
位数	8	7	1	7	1	1	2	5

图 3.13　SSPLIT 令牌包的定义格式

在 SSPLIT 令牌包中，各个字段的用途如下：

- PID 字段：定义了数据传输方向为 USB 主机到 USB 集线器。
- ADDR 字段：指明了 USB 集线器的设备地址。
- SC 字段：该字段固定为 0，用于表示开始 SPLIT 令牌包。
- PORT 字段：用于指明 USB 集线器的端口号。该字段最多可以指定 128 个 USB 集线器端口。
- S 字段：一般来说，在中断传输和控制传输下，如果 S=0，则表示全速数据传输；如果 S=1，则表示低速数据传输。在块传输和同步 IN 传输下，S 必须置为 0。而对于同步 OUT 传输，则按照表 3.1 所示进行选择。

表 3.1　同步 OUT 数据传输的 S 和 E 取值

S	E	说　明
0	0	全速数据负载的中间数据是高速数据
0	1	全速数据负载的尾部数据是高速数据
1	0	全速数据负载的首部数据是高速数据
1	1	全速数据负载的全部数据是高速数据

- E 字段：对于块传输、中断传输和控制传输来说，无需传输方向，其值必须为 0。对于同步传输来说，如果是 IN 传输则 E=0，如果是 OUT 传输，则按照表 3.1 所示进行选择。
- ET 字段：用于指明在高速数据传输中，使用的低速/全速数据传输的类型。当置 ET=00 时，表示采用控制传输；当置 ET=01 时，表示采用同步传输；当置 ET=10 时，表示采用块传输；当置 ET=11 时，表示采用中断传输。

2．CSPLIT 令牌包

结束令牌包 CSPLIT 的定义格式如图 3.14 所示。图中包含 8 位的包标识字段 PID、7 位的地址字段 ADDR、1 位的 SC 字段、7 位的 PORT 字段、1 位的 S 字段、1 位的 U 字段、2 位的 ET 字段和 5 位的循环校验字段 CRC。

CSPLIT令牌包字段	PID	ADDR	SC	PORT	S	U	ET	CRC
位数	8	7	1	7	1	1	2	5

图 3.14　CSPLIT 令牌包的定义格式

在 CSPLIT 令牌包中，各个字段的用途如下：

- PID 字段：定义了数据传输方向为 USB 主机到 USB 集线器。
- ADDR 字段：指明了 USB 集线器的设备地址。
- SC 字段：该字段固定为 1，用于表示结束 SPLIT 令牌包。
- PORT 字段：用于指明 USB 集线器的端口号。该字段最多可以指定 128 个 USB 集线器端口。
- S 字段：含义和 SSPLIT 令牌包中类似。
- U 字段：其值必须为 0，保留。
- ET 字段：用于指明在高速数据传输中，使用的低速/全速数据传输的类型。当置 ET=00 时，表示采用控制传输；当置 ET=01 时，表示采用同步传输；当置 ET=10 时，表示采用块传输；当置 ET=11 时，表示采用中断传输。

3.4.7　PRE 令牌包

PRE 令牌包的定义格式如图 3.15 所示。图比较简单，只包含一个 8 位的包标识 PID 字段。PRE 令牌包是低速数据传输的先导包，在开始低速数据传输前，必须首先发送先导包 PRE，这样 USB 集线器才会激活相应的低速数据传输端口。

PRE令牌包字段	PID
位数	8

图 3.15　PRE 令牌包的定义格式

3.5　数据包

数据包的定义格式如图 3.16 所示。图中包括 8 位的包标识字段 PID、数据字段和 16 位的循环冗余校验字段 CRC。

数据包字段	PID	数据字段	CRC
位数	8	0~1024*8	16

图 3.16　数据包的定义格式

在数据包中，各个字段的具体含义如下：

- PID 字段：用于指明不同的数据包类型。支持的 4 种数据包为 DATA0、DATA1、DATA2 和 MDATA。在后面章节介绍的数据触发机制中，使用 DATA0 和 DATA1，而前面的 SPLIT 令牌事务处理则使用 DATA0、DATA1 和 MDATA。高速 USB 同步数据传输一般需要使用全部的数据包。
- 数据字段：其中包含了传输的数据。其数据的大小根据数据传输类型和用户需要而定。根据 USB 协议中的规定，低速 USB 数据传输，数据字段最大长度为 8 个字节；全速 USB 数据传输，数据字段最大长度为 1023 字节；高速 USB 数据传输，数据字段最大长度为 1024 字节。
- CRC 字段：这里使用 16 位的循环冗余校验来对数据字段进行保护。

3.6　握手包

握手包的定义格式如图 3.17 所示。握手包仅由 8 位的包标识字段 PID 构成。握手包主要用于在数据传输的末尾报告本次数据传输的状态。握手包之后便是整个事务处理的结束信号 EOP。

握手包字段	PID
位数	8

图 3.17　握手包的定义格式

根据事务处理的状态，握手包进行不同的响应，下面具体说明主要包括的几种响应。

1．ACK 握手包

当 USB 数据传输的接收方正确接收到数据包的时候，接收方将返回 ACK 握手包。ACK 握手包表征了一次正确的数据传输，之后，才可以进行下一次事务处理。

2．NAK 握手包

NAK 握手包一般由 USB 功能设备发出。对于 IN 数据传输，表示 USB 设备没有计划向 USB 主机发送数据；对于 OUT 数据传输，表示 USB 设备无法接收 USB 主机发送的数据。

3．STALL 握手包

STALL 握手包一般由 USB 功能设备发送，表示该 USB 功能设备不支持这个请求，或者

无法发送和接收数据，STALL 握手包分为两种情况：

- 协议 STALL 握手包：在控制传输中使用，表明该 USB 功能设备不支持这个请求协议。
- 功能 STALL 握手包：表明该 USB 功能设备的相应端点已经停止，无法完成发送数据或者接收数据的操作。

4．NYET 握手包

在 SPLIT 令牌包事务处理中，如果 USB 集线器无法正常处理 SPLIT 请求，则 USB 集线器向 USB 主机返回 NYET 握手包。NYET 握手包一般只发生在高速数据传输过程中。

5．ERR 握手包

ERR 握手包用于表示总线数据传输发生错误，其一般发生在高速数据传输过程中。

3.7 USB 事务处理

在 USB 协议中，USB 总线数据传输和通信的基础是事务处理。一个完整的事务处理包括令牌阶段、数据阶段和握手阶段 3 部分，其中令牌阶段是必须的，USB 协议中规定了 7 种令牌包，因此，可以根据令牌包的类型将 USB 事务处理分为 7 种：

- IN 事务处理；
- OUT 事务处理；
- SETUP 事务处理；
- PING 事务处理；
- SOF 事务处理；
- SPLIT 事务处理；
- PRE 事务处理。

下面详细讲解各个事务处理的流程以及事务处理的注意事项。

3.7.1 IN 事务处理

一个完整的 IN 事务处理流程如图 3.18 所示。IN 事务用于实现 USB 设备到 USB 主机方向的数据传输。

图 3.18 完整的 IN 事务处理流程

整个 IN 事务处理操作步骤如下：

（1）USB 主机向 USB 设备发送 IN 令牌包，表示主机可以接收数据。

（2）USB 设备正确接收到 IN 令牌包，然后向 USB 主机发送数据包。

（3）USB 主机正确接收到数据包后，向 USB 设备返回 ACK 握手包，确认传输成功。

这里数据在 USB 主机和 USB 设备之间进行传输，在实际的数据传输中，难免出现各种错误。下面分析不同情况发生时 USB 设备和 USB 主机的响应。

1．USB 设备

在正常的数据传输情况下，USB 设备响应 USB 主机的 IN 令牌包，并向 USB 主机返回数据。当发生如下异常错误时，将采用不同的处理方式。

- 当 USB 主机向 USB 设备发送的 IN 令牌包在传输过程中被损坏时，USB 设备接收不到正确的 IN 令牌包，此时 USB 设备将忽略该令牌包，不对该 IN 令牌包进行应答。
- 如果 USB 设备接收到正确的 IN 令牌包，但是 USB 设备的 IN 端点被停止，无法向 USB 主机发送数据，此时，USB 设备将向 USB 主机发送 STALL 握手包。
- 如果 USB 设备接收到正确的 IN 令牌包，但是 USB 设备由于某种原因而无法向 USB 主机提供数据，此时，USB 设备将向 USB 主机发送 NAK 握手包。

2．USB 主机

在正常的数据传输情况下，USB 主机接收 USB 设备发送的数据包，并向 USB 设备返回 ACK 握手包。当发生如下异常错误时，将采用不同的处理方式。

- 当 USB 设备向 USB 主机发送的数据包在传输过程中被损坏时，USB 主机接收不到正确的数据包，此时 USB 主机将忽略该出错的数据包，不做任何响应。
- 当 USB 设备发送的数据包正确到达 USB 主机，但此时由于某种原因而无法接收数据，则此时，USB 主机将忽略该出错的数据包，不做任何响应。

3.7.2　OUT 事务处理

一个完整的 OUT 事务处理流程如图 3.19 所示。OUT 事务用于实现 USB 主机到 USB 设备方向的数据传输。

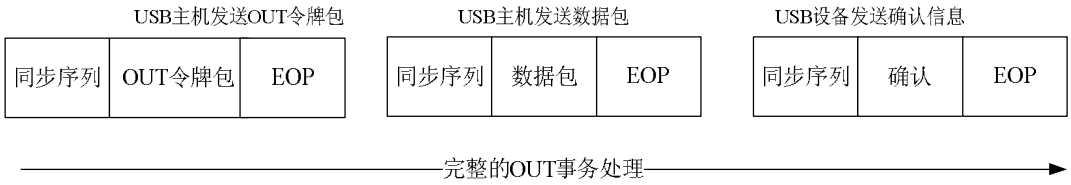

图 3.19　完整的 OUT 事务处理流程

整个 OUT 事务处理操作步骤如下：

（1）USB 主机向 USB 设备发送 OUT 令牌包，表示主机将要发送数据。

（2）USB 设备正确接收到 IN 令牌包，然后 USB 主机开始发送数据包。

（3）USB 设备正确接收到数据包后，向 USB 主机返回 ACK 握手包，确认传输成功。

这里数据在 USB 主机和 USB 设备之间进行传输。在正常的数据传输情况下，USB 设备响应 USB 主机的 OUT 令牌包，并接收 USB 主机发送数据。在实际的数据传输中，难免出现

各种错误。下面分析不同情况发生时 USB 设备的响应。

- 当 USB 主机向 USB 设备发送的 OUT 令牌包在传输过程中被损坏时，USB 设备接收不到正确的 OUT 令牌包，此时 USB 设备将忽略该令牌包，不对该 OUT 令牌包进行应答。
- 当 USB 设备接收到正确的 OUT 令牌包，但是数据包在传输过程中被损坏，则 USB 设备接收不到正确的数据包，此时，USB 设备将忽略该数据包，不对该数据包进行任何应答。
- 如果 USB 设备接收到正确的 OUT 令牌包，但是 USB 设备的 OUT 端点被停止，无法接收 USB 主机发送的数据，此时，USB 设备将向 USB 主机发送 STALL 握手包。
- 如果 USB 设备接收到正确的 OUT 令牌包，但是 USB 设备由于某种原因而无法接收 USB 主机发送的数据，此时，USB 设备将向 USB 主机发送 NAK 握手包。
- 如果 USB 设备的数据触发位和接收到数据包的触发位不一致，则 USB 设备将丢弃该数据包，然后向 USB 主机返回 ACK 握手包。

3.7.3 SETUP 事务处理

SETUP 事务处理是一种特殊的 USB 事务处理，其只在 USB 控制传输阶段使用。SETUP 事务的数据传输方向为，USB 主机到 USB 设备。

在 USB 协议中，使用 SETUP 的控制事务传输对 USB 设备进行配置。这个 SETUP 事务处理常称为建立阶段。经过配置后，USB 主机便可以向 USB 设备发送命令或者请求状态，也可以由 USB 设备向 USB 主机发送命令或者请求状态。

一个完整的 SETUP 事务处理流程如图 3.20 所示。SETUP 事务用于实现 USB 主机到 USB 设备方向的数据传输。

图 3.20 完整的 SETUP 事务处理流程

整个 OUT 事务处理操作步骤如下：

（1）USB 主机向 USB 设备发送 SETUP 令牌包，表示主机将要发送 DATA0 数据。

（2）USB 设备正确接收到 SETUP 令牌包，然后 USB 主机开始发送 DATA0 数据包。

（3）USB 设备正确接收到 DATA0 数据包后，向 USB 主机返回 ACK 握手包，确认传输成功。

这里数据在 USB 主机和 USB 设备之间进行传输。在正常的数据传输情况下，USB 设备响应 USB 主机的 SETUP 令牌包，并接收 USB 主机发送的 DATA0 数据。在实际的数据传输中，难免出现各种错误，如果接收到的 SETUP 令牌包有错误，则 USB 设备将忽略该信息包，不做出任何响应。

3.7.4 PING 事务处理

PIING 事务处理主要应用于高速数据传输中。一个完整的 PING 事务处理如图 3.21 所示。

PING 事务处理只包含令牌包和握手包阶段，而不包含数据阶段。PIING 事务处理的操作步骤如下：

（1）USB 主机向 USB 设备发送 PING 令牌包，表示一个 PING 事务的开始。

（2）USB 正确接收到该令牌包，然后 USB 设备向 USB 主机返回各种握手包进行响应。

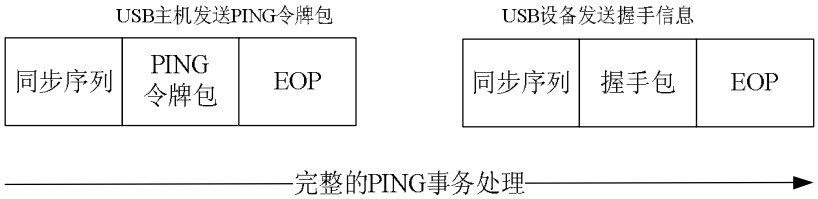

图 3.21　完整的 PING 事务处理流程

这里 USB 根据自身的情况可以产生 3 种握手包进行响应：

- ACK 握手包，表示该 USB 设备具有足够的空间来接收 USB 主机发送的数据。此时，USB 主机便停止 PING 事务处理，转而向该 USB 设备发送数据。
- NAK 握手包，表示该 USB 设备没有足够的空间来接收 USB 主机发送的数据。此时，USB 主机将重复发送 PING 令牌包进行查询。
- STALL 握手包，表示该 USB 设备的 OUT 端点已经停止工作。此时，USB 主机将停止 PING 事务，不再向该 USB 设备进行数据传输。
- 如果 USB 设备接收到的 PING 令牌包有错误，则 USB 设备忽略这里 PING 令牌包，不对其进行任何响应。

对于低速和全速 USB 事务处理，USB 主机采用循环重复尝试的方式，不断对 USB 执行相应的事务处理，直至 USB 设备可以处理该事务。如果数据包比较大，则这种方式很浪费 USB 总线资源。PING 事务处理便可以解决这个问题，防止高速 USB 数据传输中重复发送 OUT 事务处理数据包，从而可以提高 USB 总线的效率。

3.7.5　SOF 事务处理

SOF 事务处理比较简单，USB 主机发送一个 SOF 令牌包即可。SOF 事务处理表示了一个 USB 帧或者 USB 小帧的开始。整个事务处理过程没有数据阶段，也不需要 USB 设备进行握手响应。

对于不同的 USB 传输速度，SOF 事务处理的时间要求不一样。

- 对于低速和全速 USB 传输，每隔 1ms 产生一个 SOF 令牌包。
- 对于高速 USB 传输，每隔 125μs 产生一个小帧，每隔 7 个小帧，产生一个 SOF 令牌包。

3.7.6　SPLIT 事务处理

SPLIT 事务处理的目的是在 USB 高速数据传输的过程中，可以插入低速和全速 USB 数据传输，这样可以提高 USB 总线利用率。SPLIT 令牌包分为两种，因此，SPLIT 事务处理可以分为两类。

1．SSPLIT 事务处理

SSPLIT 事务处理是开始 SPLIT。一个完整的 SSPLIT 事务处理流程如图 3.22 所示。整个事务处理过程包括 3 个阶段，其操作步骤如下：

（1）USB 主机发送令牌包，这里包括 SSPLIT 令牌包和低速/全速令牌包两个令牌包。

（2）USB 设备正确接收到该令牌包，此时 USB 主机发送 DATAx 数据包。

（3）USB 设备正确接收到 DATAx 数据包，在握手阶段返回握手信息。

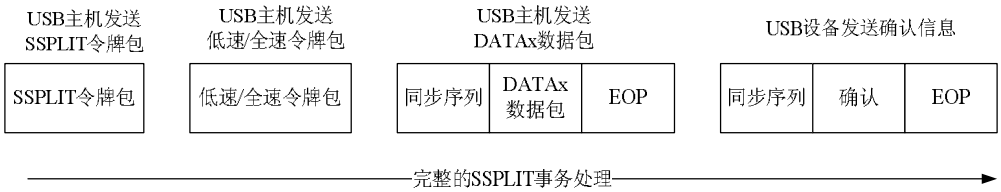

图 3.22　完整的 SSPLIT 事务处理流程

2．CSPLIT 事务处理

CSPLIT 事务处理是结束 SPLIT。一个完整的 CSPLIT 事务处理流程如图 3.23 所示。整个事务处理过程包括两个阶段，其中 DATAx 数据包和握手包可选，其操作步骤如下：

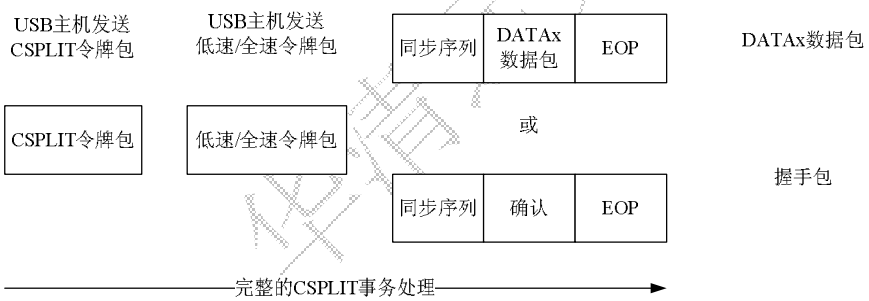

图 3.23　完整的 CSPLIT 事务处理流程

（1）USB 主机发送令牌包，这里包括 CSPLIT 令牌包和低速/全速令牌包两个令牌包。

（2）USB 设备正确接收到该令牌包，此时 USB 主机发送 DATAx 数据包，或者 USB 设备返回握手信息。

3.7.7　PRE 事务处理

一个完整的 PRE 事务处理流程如图 3.24 所示。PRE 事务处理比较简单，USB 主机直接发送 PRE 令牌包即可，也就是说，PRE 事务处理不需要数据包和握手包。

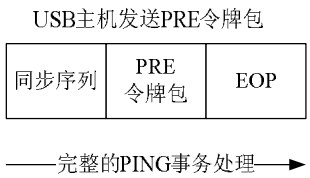

图 3.24　完整的 PRE 事务处理流程

SPLIT 事务可以在高速 USB 总线上同时运行低速和全速 USB 事务。但在低速和全速 USB 总线情况下，USB 协议中禁止低速端口，从而阻止全速 USB 事务在低速 USB 设备上的运行。为此，USB 在开始低速数据传输前，首先需要发送 PRE 事务处理，即先导包，PRE 事务处理将开启一个 USB 低速数据传输。

这里需要注意的是，PRE 事务处理只在 USB 主机和 USB 集线器之间进行，即只有 USB 集线器才可以响应 PRE 事务处理。在 USB 协议中，PRE 事务处理只在低速数据传输中使用，不适用于 USB 高速数据传输。

3.8 USB 设备

在实际的 USB 开发中，对 USB 功能设备的开发占大多数，因此这里主要介绍 USB 功能设备开发中相关的 USB 协议。这些协议是 USB 功能设备工作的基础，下面详细介绍 USB 设备的状态和总线列举操作。

3.8.1 USB 设备的状态

一个典型的 USB 功能设备和 USB 主机的连接，需要经过 5 个步骤，涉及 6 个状态，如图 3.25 所示，下面分别介绍这些 USB 设备状态。

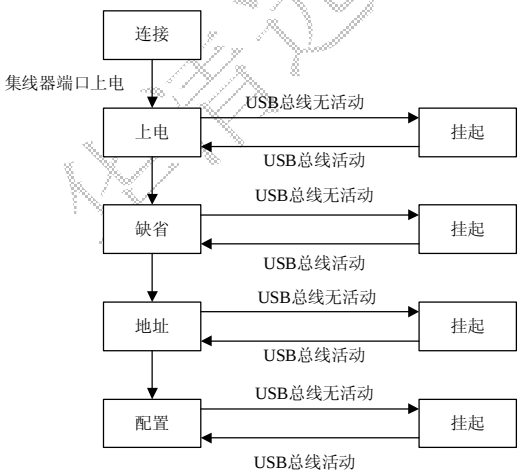

图 3.25　USB 设备状态间的转换

1．连接状态

当 USB 功能设备通过 USB 电缆连接到 USB 主机或集线器的下行端口时，即进入连接状态，此时 USB 总线开始向 USB 设备提供 V_{BUS}，至电源稳定工作。

2．上电状态

当连接到 USB 主机的 USB 功能设备得到稳定的 USB 总线电源 V_{BUS} 后，便处于上电状

态，此时其还没有被复位，不能对任何 USB 事务进行处理。

USB 功能设备可以从 USB 总线上获得电源，也可以支持自供电。USB 设备在其配置描述符的 wMaxPower 字段中指定和报告了其所需要的 V_{BUS} 电流大小。

3．缺省状态

当 USB 功能设备上电后，USB 功能设备会响应 USB 主机发出的复位信号，进行复位操作。复位结束后，USB 功能设备便进入缺省状态。

在缺省状态下，USB 设备可以从 USB 总线上获得小于 100mA 的电流，并使用缺省的设备地址对一些 USB 事务进行处理。

例如，当高速 USB 设备连接至全速 USB 主机的下行端口时，其以全速传输速率进行复位，并响应 USB 主机发出的标准 USB 设备请求，包括 SetAddress、SetConfiguration、GetDescriptor（Device）和 GetDescriptor（Configuration）。

4．地址状态

在 USB 功能设备复位结束后，USB 主机重新为该 USB 设备分配一个设备地址，这个地址是惟一的，此时便处于地址状态。

在地址状态之后，USB 设备将不再使用缺省的设备地址，而使用新分配的地址来进行以后的总线活动。

5．配置状态

在 USB 设备复位和分配地址后，主机发出 SetConfiguration 请求，USB 在正确响应配置操作后，便进入配置状态。

在配置状态下，所有寄存器返回至缺省状态，主机软件可以和 USB 设备的功能单元进行数据传输，为主机提供额外的功能。

6．挂起状态

在 USB 协议中规定，如果 USB 设备在 3ms 内没有检测到总线活动，其将自动进入挂起状态，这样可以节省 USB 系统的功率消耗。

在挂起状态下，USB 将保持原有的设备地址和配置值。在 USB 连接的任何过程中都可以进入挂起状态，USB 总线的任何活动都可使其退出挂起状态。

3.8.2 USB 总线列举

在 USB 协议中，USB 总线使用列举操作来管理 USB 设备的连接和断开。当 USB 设备连接到 USB 主机时，主机自动进行列举操作。下面分别介绍 USB 设备的连接和断开过程。

1．USB 设备的连接

在 USB 功能设备进行连接时，主机通过缺省的控制管道向其发出标准 USB 设备请求。整个连接过程分为如下步骤：

（1）当 USB 设备连接到主机或者集线器的下行端口后，总线立即为其提供电源。

（2）主机检测 D_-/D_+线上的电压，确认其下行端口有 USB 设备连接。

（3）USB 集线器通过中断 IN 管道向 USB 主机报告下行端口有 USB 设备连接。主机接到通知后，通过集线器设备类请求 GetPortStatus 获得更多的信息。

（4）USB 主机等待 100ms，以确保 USB 设备连接的稳定性。

（5）主机发送集线器设备类请求 SetPortStatus，复位连接的 USB 设备。

（6）复位结束后，USB 设备进入缺省状态，从 USB 总线获取小于 100mA 的电流，用于使用缺省地址对管道 0 的控制事务响应。

（7）主机向 USB 功能设备发送 GetDescriptor（Device）请求，获取缺省控制管道的最大数据包长度。

（8）主机发出 SetAddress 请求，为连接的 USB 设备分配一个惟一的设备地址。

（9）主机使用新的地址向 USB 设备发送 GetDescriptor（Device）请求，并读取其设备描述符的全部字段，包括产品 ID、供应商 ID 等。

（10）主机循环发送 GetDescriptor（Configuration）请求，获取完整的配置信息，包括配置描述符、接口描述符、端点描述符以及各种设备类定义描述符和供应商自定义描述符。

（11）主机根据 USB 设备的配置信息，如产品 ID、供应商 ID 等，为其选择并加载一个合适的主机驱动程序。

（12）在加载驱动程序后，便可以进行各种配置操作以及数据传输等。

2．USB 设备的断开

USB 设备从总线中断开的时候，USB 主机和 USB 集线器自动处理断开操作，整个过程包括如下步骤：

（1）当 USB 设备从集线器下行端口断开时，集线器禁止该断口，并通过中断 IN 管道向 USB 主机报告其端口状态的变化。

（2）主机发送 GetPortStatus 请求，了解详细信息，并处理该断开操作。

（3）USB 驱动程序释放其所占用的任何系统资源，如内存空间等。

如果断开的是一个 USB 集线器，则 USB 主机将对该集线器和集线器中连接的所有 USB 设备进行断开操作。

3.9 小结

本章主要讲解了 USB 总线数据传输协议中 USB 字段的格式，USB 信息包的格式，以及令牌包、数据包和握手包的结构。另外本章讲述了典型的 USB 事务处理和 USB 设备的状态和总线列举。读者应该着重掌握 USB 数据传输的格式以及典型的事务处理，这是 USB 数据通信的基础。