**介绍：CAN总线**

CAN总线是广播类型的总线。这意味着所有节点都可以侦听到所有传输的报文。无法将报文单独发送给指定节点；所有节点都将始终捕获所有报文。但是，CAN硬件能够提供本地过滤功能，让每个节点对报文有选择性地做出响应。

总线使用不归零位填充。模块以线与逻辑连接到总线：如果只有一个节点向总线传输逻辑0，那么不管有多少个节点向总线传输逻辑1，整个总线都处于逻辑0状态。

CAN标准定义四种不同的报文类型。报文使用逐位仲裁智能方案来控制对总线的访问，每条报文都带有优先级标记。

CAN标准还为错误处理和消除定义了详细的方案，这在第9节“CAN错误处理”（第23页）中有更详细的说明。

在本教程第8页讨论位时序和时钟同步。[位时序计算器](http://www.kvaser.cn/support/calculators/bit-timing-calculator/)见此页面，您可以用它来计算CAN总线参数和寄存器设置。

CAN可以使用不同的物理层来实现（第5页），这里阐述其中一些。而且，可以使用许多不同种类的连接器。我们还为对报文细节感兴趣的用户提供了许多示波器图片（第6页）。

**CAN报文（第1页/共3页）**

CAN总线是广播类型的总线。这意味着所有节点都可以侦听到所有传输的报文。无法将报文单独发送给指定节点；所有节点都将始终捕获所有报文。但是CAN硬件能够提供本地过滤功能，让每个节点对报文有选择性地做出响应。

**CAN报文**

CAN使用短报文 – 最大实用负载是94位。报文中没有任何明确的地址；相反，可以认为报文是通过内容寻址，也就是说，报文的内容隐式地确定其地址。

**报文类型**

CAN总线上有四种不同的报文类型（或“帧”）：

1. 数据帧
2. 远程帧
3. 错误帧
4. 过载帧

**1. 数据帧**

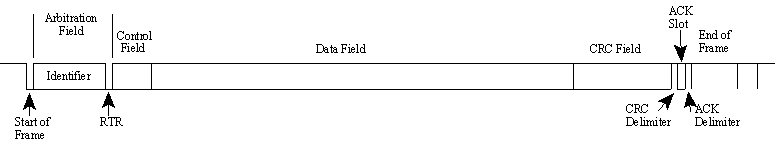
概要：“大家好，这是一些标签为X的数据，希望满足您的需要！”

数据帧是最常见的报文类型。由下列主要部分组成（出于简洁性目的，忽略了一些细节）：

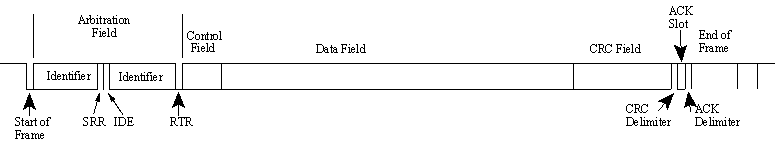
* 仲裁字段。当两个或多个节点竞争总线时，确定报文的优先级。仲裁字段包含：
  + 对于CAN 2.0A，一个11位的标识符和一个支配数据帧的RTR位。
  + 对于CAN 2.0B，一个29位的标识符（其中还包含两个隐性位：SRR和IDE）和RTR位。
* 数据字段。包含0到8字节数据。
* CRC字段。包含一个基于报文大部分数据计算得到的15位校验和。校验和用于错误检测。
* 应答时隙。任何能够正确接收报文的CAN控制器都会在每条报文的末尾发送一个应答位。传送节点检查应答位是否存在，如果没有检测到应答位，会重新发送报文。

注1：请注意，总线上存在应答位不表示任何目标地址已经收到报文。唯一表示的是，总线上的一个或多个节点已经正确收到报文。

注2：仲裁字段中的标识符不一定标识报文的内容，尽管其名字会让人如此认为。

[](http://www.kvaser.cn/wp-content/uploads/2014/01/1-can-msg-1-3.gif)

*CAN 2.0A（“标准CAN”）数据帧*

*[](http://www.kvaser.cn/wp-content/uploads/2014/01/2-canmsg-1-3.gif)*

*CAN 2.0B（“扩展CAN”）数据帧*

**2. 远程帧**

概要：“大家好，有没有人能够生成标签为X的数据？”

远程帧与数据帧十分相似，但是有两个重要的区别：

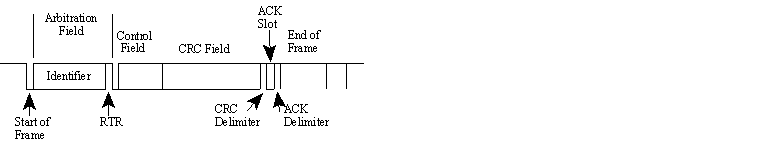
* 它被显式标记为远程帧（仲裁字段中的RTR位为隐性），并且
* 它没有数据字段。

远程帧的预期目的是征求传输相应的数据帧。例如，如果节点A传输一个仲裁字段设置为234的远程帧，那么节点B（如果已经正确初始化）可能通过一个仲裁字段也设置为234的数据帧进行响应。

远程帧可以用来实现总线通信管理的请求-响应类型。但是，远程帧在实践中很少使用。还有一点值得注意的是，CAN标准没有规定这里列出的特性。大部分CAN控制器都可以进行编程自动响应远程帧或通知本地CPU。

关于远程帧的一个注意事项：数据长度代码必须设置成预期响应报文的长度。否则仲裁将不起作用。

有时据称响应远程帧的节点会在识别标识符后立即开始传输，从而“填充”空的远程帧。但是事实并非如此。  
远程帧（2.0A类型）：

*[](http://www.kvaser.cn/wp-content/uploads/2014/01/1-can-msg-2-3.gif)*

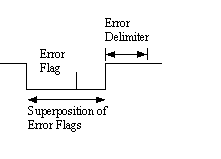
**3. 错误帧**

概要：“大家好（大声），让我们重新试一下”

简单地说，错误帧是一种违背CAN报文帧规则的特殊报文。一个节点咋在检测到故障时传送错误帧，这将导致所有其它节点也检测到故障，它们也将发送错误帧。然后传送节点将自动尝试重传报文。有一种精准的错误计数器方案，可以确保节点无法通过重复传输错误帧来破坏总线通信。

错误帧包含一个错误标志。这个错误标志是6位长的相同值（因此违背位填充规则）和一个错误定界符（8个隐性位）。错误定界符提供一些空间，以便总线上其它节点在检测到第一个错误标志时可以发送它们的错误标志。

错误帧图示如下：

*[](http://www.kvaser.cn/wp-content/uploads/2014/01/2-can-msg-2-3.gif)*

**4. 过载帧**

概要：“我是非常繁忙的小型82526，您可以稍等片刻吗？”

这里，我们仅仅出于知识完整性目的而提及过载帧。在格式方面，过载帧与错误帧非常相似，并且由会变得重荷的节点传送。过载帧并不常用，因为当今的CAN控制器会非常智能化地避免使用过载帧。事实上，会生成过载帧的唯一一种控制器是现在已经过时的82526。

**标准CAN和扩展CAN**

最初，CAN标准把仲裁字段中标识符的长度定义为11位。后来，客户的需求推动了该标准得延伸。新格式通常称为扩展CAN，标识符不允许少于29位。为了区分这两种帧类型，在控制字段中使用了一个保留位。

标准的正式名称是

* 2.0A，仅支持11位标识符
* 2.0B，支持完整的29位标识符（也可以混合使用11位标识符）的扩展版本。2.0B节点可以是
  + “2.0B主动型”，也就是说，它可以发送和接收扩展帧，或者
  + “2.0B被动型”，也就是说，它将静默丢弃接收到的扩展帧（但是请参考下述内容）
* 1.x表示初始规范及其修订版。

如今的新型CAN控制器通常是2.0B类型。1.x或2.0A类型的控制器如果接收到29个仲裁位的报文会很麻烦。2.0B被动型控制器容许这些报文，如果它们正确的话会进行应答，然后抛弃这些报文。2.0B主动型控制器既可传送也可接收这些报文。

实现2.0B和2.0A（和1.x）的控制器相互兼容，并且可以在同一总线上使用它们，则只要实现2.0B的控制器不发送扩展帧即可！

有时候人们说标准CAN“优于”扩展CAN，因为扩展CAN报文会产生更多的间接费用。这种说法未必正确。如果您使用仲裁字段来传送数据，那么扩展CAN实际上的发生的间接费用可能比标准CAN少。

**基本CAN和完整CAN**

术语“基本CAN”和“完整CAN”源自CAN的初创年代。历史上曾经有Intel 82526 CAN控制器，它为程序员提供DPRAM型接口。然后出现了Philips 82C200 CAN控制器，它使用面向FIFO（队列）的编程模型和有限制的过滤功能。为了区分这两种编程模型，人们出于某些原因把Intel的模型称为“完整CAN”，把Philips的模型称为“基本CAN”。如今，大多数CAN控制器同时支持这两种编程模型，所以没有理由继续使用术语“基本CAN”和“完整CAN”。事实上，这些术语可能会令人迷惑，应当尽量避免。

当然，“完整CAN”控制器可以和“基本CAN”进行通信，反之也可。不存在任何兼容性问题。

**总线仲裁和报文优先级**

报文仲裁（两个或多个CAN控制器协商确定谁使用总线的过程）对于数据传送中真正可用的带宽非常重要。

任何CAN控制器都可能在检测到空闲总线时开始传送数据。这可能会导致两个或多个控制器（几乎）同时开始传送报文。通过下述方法解决这种冲突。传送节点在发送报文时监视总线。如果节点在自身发送隐性位时检测到显性位，它将立即退出仲裁过程并转变成接收方。仲裁针对整个仲裁字段进行，当该字段已经被发送时，总线上恰好只剩一个传送节点。这个节点就像什么都没发生那样继续传送。其它潜在的传送节点将在总线下次空闲时尝试重传它们的报文。仲裁过程不会消耗时间。

这种逐位仲裁成功的重要条件是两个节点不会传送相同的仲裁字段。这个规则有一个例外：如果报文不包含任何数据，那么任何节点都可传送该报文。

因为总线采用线与逻辑，并且显性位是逻辑0，所以包含数值最低仲裁字段的报文将赢得仲裁。

*问：如果一个节点是总线上的唯一节点并且试图进行传送，会发生什么情况？*  
答：当然，节点将赢得仲裁并顺利地进行报文传送。但是，当进行应答的时候，没有任何节点将在应答时隙（ACK时隙）发送显性位，所以传送节点将检测到ACK错误，发送错误标志，将传送错误计数器加8并开始重传。这将发生16次。然后传送节点将进入错误被动状态。通过错误限制算法的特殊规则，如果节点为被动错误并且错误是ACK错误，则传送错误计数器不再增加。所以节点将一直继续传送，至少要到有人应答报文。

**报文寻址和标识**

再次提醒，CAN报文中无显式地址。每个CAN控制器都会检查总线上的所有通信，并使用硬件过滤器和软件的组合来确定是否对该报文“感兴趣”。

事实上，CAN中没有报文地址的概念。报文的内容由存在于报文中某处的标识符进行标识。CAN报文被称为“内容寻址”的报文。

常规的报文地址类似于“这是发给节点X的报文”。而内容寻址的报文类似于“这是包含标签为X的数据的报文”。这两个概念之间的区别很小但是非常重要。

根据标准，仲裁字段的内容用来确定总线上报文的优先级。所有CAN控制器还将在硬件过滤过程中使用整个（一些控制器将只使用一部分）仲裁字段作为主键。

标准没有规定仲裁字段必须用作报文标识符。但是不管怎样，这是一种常见情况。

**标识符值注意事项**

我们说过，标识符可以是11位（CAN 2.0A）或29位（CAN 2.0B）。这不完全正确。出于与某种老式CAN控制器（猜猜是哪种）兼容的原因，标识符不得把7个最高有效位都设置为1。所以，对于11位标识符，只剩下0到2031，29位标识符可以使用532676608个不同的值。

注意，所有其它CAN控制器都接受“非法”标识符，所以现代CAN系统标识符2032到2047可以自由使用。

**CAN物理层**

**CAN总线**

CAN总线使用不归零（NRZ）的位填充。有两种不同的信令状态：显性（逻辑0）和隐性（逻辑1）。这些信令状态对应于所在物理层（存在几种不同的物理层）的某种电平。模块以线与逻辑连接到总线：哪怕只有一个节点发送逻辑0使得总线处于显性状态，那么不管有多少隐形状态的节点传送，则整个总线都处于显性状态。

**不同的物理层**

物理层定义总线上的电平和信令方案、缆线阻抗和类似的方面。

有几种不同的物理层：

* 最常见的类型由CAN标准ISO11898-2部分定义的，它是双线平衡信令方案。有时也称为“高速CAN”。
* 同一个ISO标准的另一部分（ISO 11898-3）为低总线速度定义了另一种双线平衡信令方案。它具有容错能力，所以即使一条总线线缆断开或对地短路或连接到备用电池，信号都可以继续发出。有时也称为“低速CAN”。
* SAE J2411定义单线缆（当然加上接地）物理层。主要用在汽车中 – 例如GM-LAN。
* 还有几种专有的物理层。
* 在没有CAN驱动程序的早期，RS485修订版。
* 有关报文的详细信息，请转到第6页查看多个示波器图片。

规则规定不同的物理层不能交互操作。某些组合可能在良好的条件下工作，或看上去可以工作。例如，在同一条总线上同时使用“高速”和“低速”收发器，有时可以工作。

大量CAN收发器芯片产自Philips；其它厂商包括Bosch、Infineon、Siliconix和Unitrode。

一种常见的收发器类型是82C250，它实现ISO 11898定义的物理层。82C251是一种改进的版本。

“低速CAN”的一种常见收发器是产自Philips的TJA1054。

**最大总线速度**

按照标准，CAN总线的最大速度是1 Mbps。然而，一些CAN控制器能够处理比1Mbps更快的速度，可以在特殊应用场合中使用。

低速CAN（ISO 11898-3，参见上文）最快可以达到125 kbps。

单线CAN在标准模式下可高达约50kbps，并且如果使用特殊的高速模式，诸如用于ECU编程，可以高达约100kbps。

**最小总线速度**

注意，一些总线收发器不允许低于特定的比特率。例如，使用82C250或82C251时，低于10kbps不会有问题。但是，如果使用TJA1050，则不能低于大约50kbps。请查阅数据表。

**最大线缆长度**

如果速度为1Mbps，可以使用的最大线缆长度大约为40米（130英尺）。这是因为仲裁方案要求信号的峰值可以到达最远的节点并且在位采样之前再次返回。换言之，线缆长度受光速限制。曾有人提出提高光速，但是因为这会产生时空交错而被驳回。

其它的最大线缆长度是（这些是近似值） –

* 100米（330英尺），比特率为500kbps
* 200米（650英尺），比特率为250kbps
* 500米（1600英尺），比特率为125kbps
* 6千米（20000英尺），比特率为10kbps

如果使用光耦合器来提供电隔离，那么最大总线长度会相应地减小。提示：使用快速光耦合器并通过设备查看延迟，而不是采用指定的最大比特率。

**总线端接**

ISO 11898 CAN总线必须进行端接。 通过在总线各端点使用120欧姆的电阻达到这个要求。端接可以达到两个目的：

1. 消除总线终端处的信号反射。
2. 确保总线获得正确的直流电电平。

不管速度快慢，ISO 11898 CAN总线始终都必须进行端接。我将重复这一点：不管速度快慢，ISO 11898 CAN总线始终都必须进行端接。对于实验性工作，一个端接器可能就已足够。如果没有连接任何端接器，但是您的CAN总线仍然能够正常工作，那么您只是比较幸运而已。

注意，其它物理层（例如“低速CAN”、单线缆CAN和其它物理层）不一定需要进行端接。但是您的常用高速ISO 11898 CAN总线总是需要至少一个端接器。

请参考[这篇文章](http://www.kvaser.cn/using-termination-ensure-recessive-bit-transmission/)获取关于CAN总线端接的更多信息。

**线缆**

ISO11898规定线缆阻抗标称值为120欧姆，但允许[108-132]欧姆区间的阻抗。

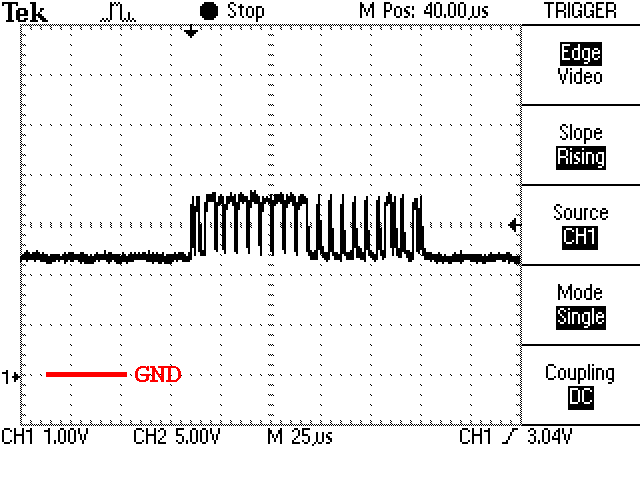
现今市场上满足这种要求的线缆并不多。允许的阻抗区间将来很有可能会扩大。

ISO 11898针对屏蔽或非屏蔽的双绞线而定义。单线缆标准SAE J2411的相关工作正在进行之中。

**CAN连接器**

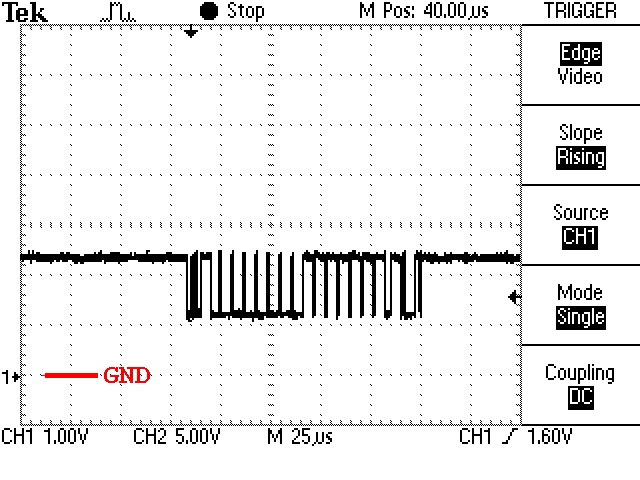
对于CAN总线连接器，根本没有任何标准！通常，每种高层协议都会定义一种或一些首选的连接器类型。常见的类型包括

* 9针DSUB（由iCiA建议）。
* DeviceNet和SDS使用的5针Mini-C和/或Micro-C连接器。
* 由CANHUG提出的用于移动液压装置的6针Deutch连接器。
* 参见第7页中一些不同的连接器布局。
* **CAN示波器图片**
* 这是一幅来自最普通的ISO11898CAN总线的图片，以1Mbps速率运行。收发器是82C251。也就是说，物理层是由ISO 11898指定的。

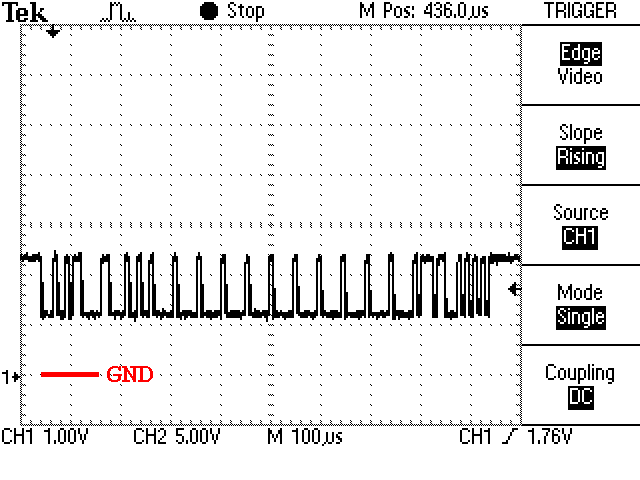
[](http://www.kvaser.cn/wp-content/uploads/2014/01/1-canhigh1.gif)

测量在CAN\_H（CAN高位）和GND（接地）之间进行。注意，静态和隐性总线电压在2.5 V左右。传送显性位时，电压升高到3.5V左右。

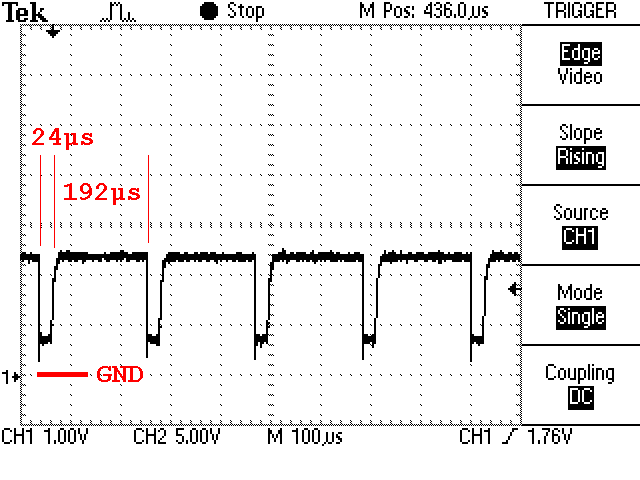
以下是同一个总线，但是测量在CAN\_L（CAN低位）和GND（接地）之间进行：

[](http://www.kvaser.cn/wp-content/uploads/2014/01/2-canlow1.gif)

这是另一个以125 kbps比特率发送的报文。报文的（11位）标识符是300（十六进制值为12c）。如果仔细察看，您应该能够识别报文中前面的位。

[](http://www.kvaser.cn/wp-content/uploads/2014/01/3-canframe1.gif)

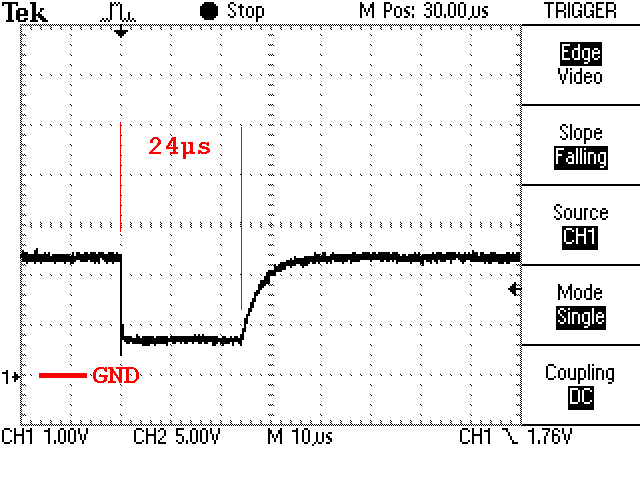
这是一幅更复杂的图片。它显示与上面例子相同的报文。仍然是11位标识符300，比特率仍然是125kbps，但是CAN总线上没有端接。CAN线缆是短扁平带状线缆。

[](http://www.kvaser.cn/wp-content/uploads/2014/01/4-canerrorpassive1.gif)  
那么，发生了什么情况？这里，比特率是125kbps，所以一个位的时间是8微秒。

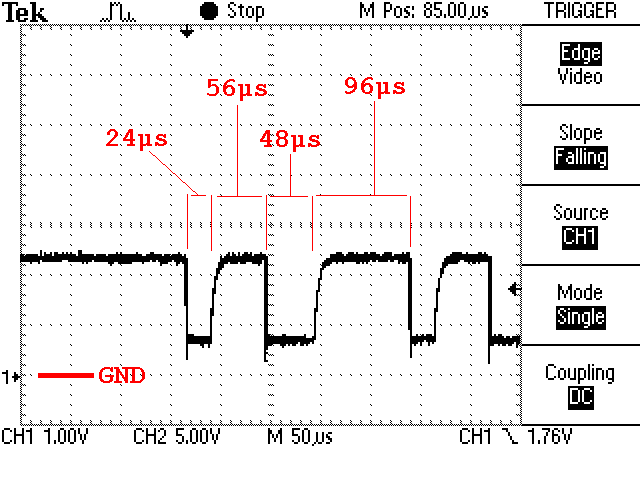
* 1. 首先传送节点发送一个起始位。这是一个逻辑’0′，也就是一个显性电平。
  2. 然后传送标识符。十进制300的十六进制值是12c，或者二进制表示是001 0010 1100。前两个0会顺利传送。这解释了图片中看到的24微秒的显性电平。
  3. 然后应该传送一个’1′。但是因为总线没有端接，所以斜率上升不是预期的结果。传送节点现在将认为它在总线上看到的是’0′。
  4. 因为这种情况发生在仲裁阶段，所以传送节点将停止传送，其认为有其它节点正在传送。总线现在将变为隐性状态，因为实际上没有节点在传送。
  5. 在6个隐性位之后，传送节点和接收节点都将检测到填充错误，并开始进行错误处理。这时，已经经过了80微秒（一个起始位、两个2个‘0’、一个误解位和六个隐性位，总共10个位，等于80微秒）。
  6. 检测到数据错误的所有节点现在将开始传送一个错误帧。这种情况下，因为在捕获到上方图片之前产生了许多错误，错误帧为被动型，所以传送节点是错误被动型。被动型错误帧和主动型错误帧相似，但是使用隐性电平进行传送，所以在总线上不可见。
  7. 被动型错误帧持续6位的时间。
  8. 然后，所有节点等待8个隐性位的时长（称为错误定界符）。
  9. 然后，所有节点等待3个隐性位的时长（称为间歇）。
  10. 对以上时间求和，结果是1+6+6+8+3 = 24个隐性位 = 192微秒（如图）

经验提示：始终端接CAN总线！反射不一定有害，但是损坏的边缘形状将破坏通信。

这里是同一个CAN总线在另一个时间刻度中的情况：

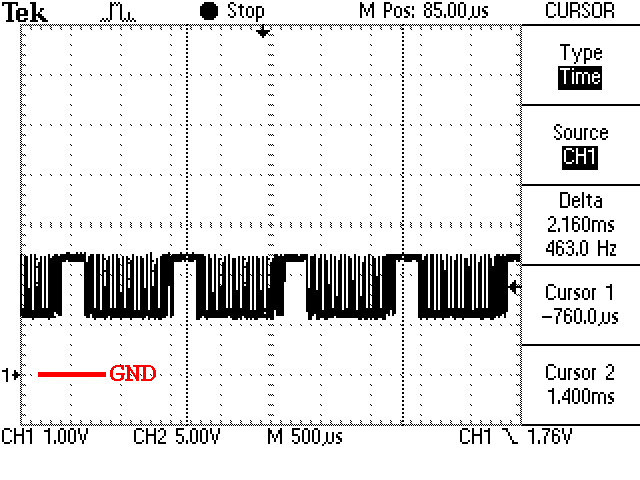
[](http://www.kvaser.cn/wp-content/uploads/2014/01/5-canepcloseup1.gif)  
CAN总线大约2分米（8英寸）长。信号的下冲和振铃均可见，但是在这种情况中无关紧要。这次，平缓的上升斜率是问题所在。

这里是相同的设置，但是这次传送节点和接收节点都是错误主动型：

[](http://www.kvaser.cn/wp-content/uploads/2014/01/6-canerroractive1.gif)  
发生了什么情况？

* 1. 如上图所示，传送了三个‘0’（花费24微秒），接下去的位被误解，所以传送器认为它已经失去仲裁。
  2. 传送节点等待6位，然后检测到一个填充错误。误解的位和这6个位花费56微秒。
  3. 传送节点和接收节点现在开始传送错误帧。它是6个显性位（48微秒）。
  4. 传送错误帧的节点现在等待8个隐性位。但是，因为上升斜率不对，第一个位被误解。节点将认为这是另一个节点在传送错误帧，所以将忽略它。
  5. 当总线回到隐性电平时，所有节点等待8位。
  6. 然后是3个隐性位的间歇。
  7. 3+9 = 12个位 = 96微秒（如图中所示）。
  8. 然后，传送节点重新尝试并得到相同的结果。一段时间以后，传送节点进入错误被动型状态，并将如前所述那样运行。

这里是另一幅图片。在此设置中，CAN总线上只有一个节点（正确端接）。该节点试图传送一条报文，但是没有其它节点在侦听。

[](http://www.kvaser.cn/wp-content/uploads/2014/01/7-justonenodetransmitting1.gif)  
那么，会发生什么情况？

* 1. 首先，传送节点发送整个报文。
  2. 传送节点期望在ACK时隙中填充一个显性电平。但是，因为没有其它节点在侦听，没有任何ACK到达，所以传送节点检测到一个应答错误。
  3. 然后传送器传送一个被动型错误标志（上图中，尝试发送了几秒钟，所以不再是错误主动型，而是被动型）。
  4. 被动型错误标志后面跟随一个错误定界符和间歇。
  5. 因为这个节点尝试发送一个报文但是操作失败，它必须再等待8位才能开始新的传送。这种情况在CAN规范中称为“挂起传送”。
  6. 传送节点还必须将其传送错误计数加8。但是，这是CAN规范中的特殊情况，只有当传送节点是错误主动型时才会发生。当传送节点进入错误被动型时，它不会增加其传送节点错误计数（这种情况中），而是会不断重试传送。

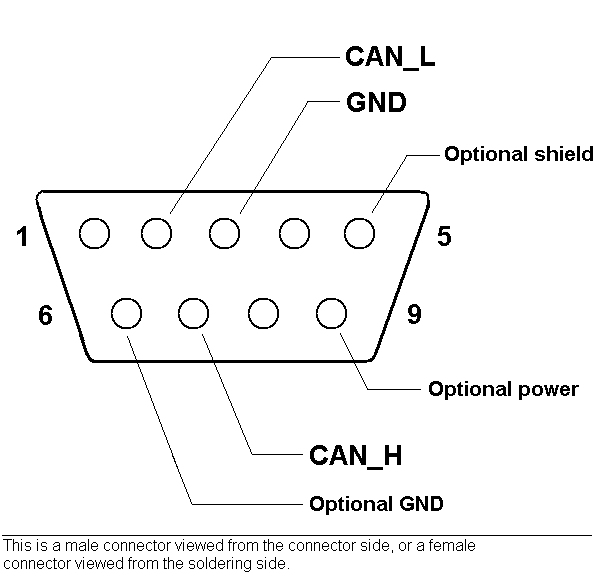
所以，上图表示，一条报文被传送，然后短暂停顿（时间为错误标志、错误定界符、间歇和挂起传送的总和）。然后报文被不断重传……

* **CAN连接器**
* **9针DSUB**

[CiA](http://www.kvaser.cn/can-automation-cia/)推荐使用这种连接器布局，是事实上的工业标准。

|  |  |  |
| --- | --- | --- |
| 1 | – | 保留位 |
| 2 | CAN\_L（CAN低位） | CAN\_L总线线路（显性低位） |
| 3 | CAN\_GND（CAN接地） | CAN接地 |
| 4 | – | 保留位 |
| 5 | CAN\_SHLD（CAN屏蔽） | 可选 CAN屏蔽 |
| 6 | GND（接地） | 可选 CAN接地 |
| 7 | CAN\_H（CAN高位） | CAN\_H总线线路（显性高位） |
| 8 | – | 保留位（错误行） |
| 9 | CAN\_V+（CAN电源） | 可选 电源 |

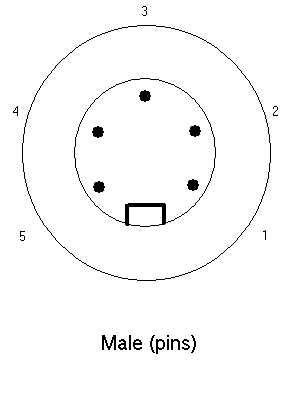
*KVASER用户：请注意，文档“LAPcan硬件指南”中阐述了如何在KVASER DRVcan驱动器线缆上使用这些引脚，可以*[*在此下载*](https://www.kvaser.cn/downloads-kvaser/)*该文档。*  
如果如果供电，电压范围应该为+7至+13 V，额定电流为100 mA。模块提供一个公接头，内部必须连接引脚3和6。

[](http://www.kvaser.cn/wp-content/uploads/2014/01/can-connectors-1.gif)

引脚编号适用于接头侧视图的公接头或焊接侧视图的母接头。为了方便记住引脚编号，可留意CAN\_LOW的引脚号小，而CAN\_HIGH的引脚号大。

**5针迷你C型接头**

[DeviceNet](http://www.kvaser.cn/about-can/higher-layer-protocols/devicenet/)和[SDS](http://www.honeywell.com/sensing/prodinfo/sds)都使用这种接头，并且这两种协议碰巧兼容。

[](http://www.kvaser.cn/wp-content/uploads/2014/01/can-connectors-2.gif)

|  |  |  |
| --- | --- | --- |
| 引脚 | 功能 | DeviceNet颜色 |
| 1 | 信号输出 | 无颜色 |
| 2 | V+ | 红色 |
| 3 | V- | 黑色 |
| 4 | CAN\_H | 白色 |
| 5 | CAN\_L | 蓝色 |

模块含公接头。输入电压为24V+/-1%。

注意：在DeviceNet规范版本1.x中，图9.13中母接头的编号顺序错误。规范2.0及后续版本的编号顺序正确。

**6针德驰DT04-6P**

CANHUG建议用在移动液压应用场合中。

模块端为公接头，总线端为母接头。对于输入电压，目前没有建议的值。

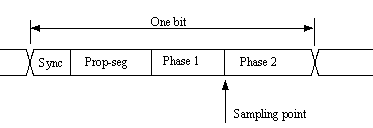
|  |  |  |
| --- | --- | --- |
| 引脚 | 功能 | 建议的线缆颜色 |
| 1 | 电源负极 | 黑色 |
| 2 | CAN\_H | 白色 |
| 3 | 可选 – 信号接地 | 黄色 |
| 4 | 可选 – 启动 | 灰色 |
| 5 | 电源正极 | 红色 |
| 6 | CAN\_L | 蓝色 |

* **CAN位时序**
* **位布局**

出于时序目的，CAN总线上的每个位都划分成至少4个时间份额。时间份额逻辑上划分成四个组或段-

* 1. 同步段
  2. 传播段
  3. 相位段1
  4. 相位段2

以下是CAN数据位的图片：

[](http://www.kvaser.cn/wp-content/uploads/2014/01/bit-timing-1.gif)

**辅助工具**

**位时序计算器**

指定输入频率和总线速度计算所有可能的CAN总线参数集

[点击计算](https://www.kvaser.cn/support/calculators/bit-timing-calculator/)

同步段始终是一个时间份额长，用于时钟同步。总线上的数据改变时，预计会出现一个位边沿。

传播段用来补偿总线线路中的延迟。

如有必要，相位段可以缩短（相位段1）或延长（相位段2），以保持时钟同步。

在相位段1和相位段2之间的边沿上对总线电平进行采样。

大部分CAN控制器还提供选项可以在一个位期间进行三次采样。这种情况中，在采样点之前的两个时间份额的边沿上进行采样，择多译码（至少82527是这样）。

**时钟同步**

为了调整片上总线时钟，CAN控制器可以将位的长度缩短或延长整数倍个时间份额。这些位时间调整的最大值称为同步跳变宽度（SJW）。

硬同步在起始位从隐性到显性转换时产生。位时间从该边沿重新开始。

重新同步是在报文的同步段中不发生位边沿时产生。一个相位段被缩短或延长，其缩短值或延长值取决于信号中的相位误差。可以使用的最大值由同步跳变宽度参数决定。

**位时序寄存器计算**

大部分CAN控制器允许程序员使用下列参数设置位时序：

* 1. 时钟预分频器值
  2. 采样点前的时间份额数
  3. 采样点后的时间份额数
  4. 同步跳变宽度（SJW）中的时间份额数

通常为此提供两个寄存器：btr0和btr1。但是，不同控制器之间略有差异，因此请仔细阅读数据表。

恩智浦（恩飞公司）82c200和SJA1000的寄存器布局如下：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| btr0 | SJW1 | SJW0 | BRP5 | BRP4 | BRP3 | BRP2 | BRP1 | BRP0 |
| btr1 | SAM | TSEG22 | TSEG21 | TSEG20 | TSEG13 | TSEG12 | TSEG11 | TSEG10 |

* 1. BRP0..BRP5设置时钟预分频器值
  2. SJW0..SJW1设置SJW的长度
  3. TSEG10..TSEG13设置采样点前的时间份额数（不包含起始位）
  4. TSEG20..TSEG22设置采样点后的时间份额数
  5. 如果要获取三个样本，则SAM设置为1，如果一个样本就已足够，则设置为0。

*注意：这些参数的实际值比写入寄存器的值大1。*

示例：如果提供给SJA1000的振荡器信号是16MHz，我们想要250kbps的比特率，采样点接近整个位的62%，以及SJW为2个时间份额，那么我们可以设置 –

BRP = 4，指定时间份额长度为2 \* 4 / 16000000 s = 500纳秒，以及  
TSEG1 = 5，指定采样点之前有5个时间份额，以及  
TSEG2 = 3，指定采样点之后有3个时间份额。

这样，每个位都将包含5 + 3 = 8个时间份额，最终的比特率为1 / (8 \* 500 ns) = 250 kbps。寄存器值应该为

|  |  |
| --- | --- |
| btr0 = | (SJW – 1) \* 64 + (BRP -1) = (2-1)\*64 + (4-1) = 67 = 0×43 |
| btr1 = | SAM \* 128 + (TSEG2 – 1)\* 16 + (TSEG1 – 1) = 0\*128 + (3-1)\*16 + (4-1) = *(“4″ 因为不包含起始位)* 35 = 0×23 |

采样点为位的5/8=62.5%。

**CAN错误处理**

* **CAN怎样处理错误**

错误处理内建在CAN协议中，对CAN系统的运行十分重要。错误处理的目标是检测CAN总线上出现的报文中的错误，从而传送器可以重传出错的报文。总线上的每个CAN控制器都会尝试检测报文中的错误。如果发现错误，发现节点将传送一个错误标志，从而中断总线通信。其它节点将检测错误标志导致的错误（如果它们尚未检测到初始错误）并采取合适的措施（例如丢弃当前报文）。

每个节点维护两个错误计数器：传送错误计数器和接收错误计数器。有几个规则规定这些计数器怎样增加和/或减少计数。总的来讲，检测到故障的传送器增加其传送错误计数器比侦听节点增加其接收错误计数器要快。这是因为，很有可能是传送器发生了故障！当任何错误计数器增加到超过某个值时，节点将首先进入“错误被动”，也就是说，它在检测到错误时不会积极地阻止总线通信，然后“离开总线”，这意味着节点根本不参与总线通信。

使用错误计数器，CAN节点不但可以检测故障，而且可以执行错误限制。

**错误检测机制**

CAN协议定义了五种以上不同的方法来检测错误。其中两种工作在位层次，另外三种工作在报文层次。

* 1. 位监视
  2. 位填充
  3. 帧检查
  4. 应答检查
  5. 循环冗余检查

**1. 位监视**

CAN总线上的每个传送器都会监视（也就是回读）传送的信号电平。如果真正读到的位电平与传送的位电平不同，会发出信号指示位错误（仲裁过程中不会发出位错误）。

**2. 位填充**

节点连续发送五个具有相同电平的位后，将在发送出去的位流中加上第六个相反电平的位。接收方将删除这个额外的位。这样做是为了避免总线上出现过度的直流电成分，但它同时也给予了接收方检测错误的额外机会：如果总线上出现五个以上相同电平的连续位，会发出信号指示填充错误。

**3. 帧检查**

CAN报文的一些部分具有固定的格式，也就是说，标准明确定义了何种电平和何时出现这种电平（CRC定界符、ACK定界符、帧结束以及间歇，但是间歇还有一些另外的特殊错误检查规则）。如果一个CAN控制器在这些固定字段中的一个中检测到无效值，将发出组成错误。

**4. 应答检查**

总线上正确接收报文的所有节点（不管这些节点是否对报文内容“感兴趣”）预期将在报文中所谓的应答时隙中发送一个显性电平。这是，发送方将发送一个隐性电平。如果发送方无法在应答时隙中检测到显性电平，会发出应答错误的信号。

**5. 循环冗余检查**

每个报文都包含一个15位的循环冗余校验和（CRC）。节点如果在报文中检测到与自己计算所得不同的CRC，将发出CRC错误的信号。

**错误限制机制**

总线上的每个CAN控制器都将在每个报文中尝试检测以上列出的错误。如果发现错误，发现节点将发送一个错误标志，从而停止总线通信。其它节点将检测到由错误标志引发的错误（如果它们尚未检测到原始错误）并采取恰当的措施（也就是丢弃当前报文）。

每个节点维护两个错误计数器：传送错误计数器和接收错误计数器。有几个规则规定这些计数器怎样增加和/或减少计数。总的来讲，检测到故障的传送器增加其传送错误计数器比侦听节点增加其接收错误计数器要快。这是因为很有可能是传送器发生了故障！

节点开始时处于主动错误模式。当两个错误计数器中的任何一个的计数超过127时，当两个错误计数器中的任何一个计数超过127时，节点将进入称为被动错误的状态。当传送错误计数器达到255以上时，节点将进入总线离线状态。

* 1. 主动错误节点将在检测到错误时发送主动错误标志。
  2. 被动错误节点将在检测到错误时发送被动错误标志。
  3. 总线离线节点不会在总线上传送任何内容。

增加和减少错误计数器的规则略显复杂，但是原理比较简单：传送错误的步进为8个错误点数，而接收错误的步进为1个错误点数。正确传送和/或接收的报文会减小计数器值。  
示例（稍加简化）：假设总线上的节点A运气不好。不管什么时候A发送报文都发生失败（由于种种原因）。每次失败时，它的传送错误计数器增加8并发送主动错误标志。然后它将尝试重传报文，但是一直失败。

当传送错误计数器超过127时（也就是在16次尝试之后），节点A进入被动错误模式。区别在于，它现在将在总线上传送被动错误标志。被动错误标志包含6个隐性位，不会影响其它的总线通信，所以其它节点不会侦听到有关总线错误的反馈。但是，A继续增加其传送错误计数器。当计数值超过255时，节点A最终放弃传送并进入总线离线模式。

其它节点怎样响应节点A？对于A传送的每个主动错误标志，其它节点将为其接收错误计数器增加1。当A进入总线离线状态时，其它节点的接收错误计数器的计数大大低于错误被动的限制值（127）。每次正确收到一条报文，这个计数值将减1。但是，节点A将停留在总线离线状态。

大部分CAN控制器将为两种状态提供状态位（和相应的中断）：

* 1. “错误警告” – 一个或两个错误计数器的计数超过96
  2. “总线离线”，如上所述。

一些控制器（但不是所有）还为被动错误状态提供一个位。一些控制器还可以直接访问错误计数器。

CAN控制器在发生错误时自动重传报文的特性有时候会很烦人。市场上至少有一种控制器（飞利浦公司生产的SJA1000）允许对错误处理完全人工控制。

**总线故障模式**

ISO 11898标准列出了CAN总线线缆的几种故障模式：

* 1. CAN\_H中断
  2. CAN\_L中断
  3. CAN\_H短路到电池电压
  4. CAN\_L短路到接地
  5. CAN\_H短路到接地
  6. CAN\_L短路到电池电压
  7. CAN\_L短路到CAN\_H线路
  8. CAN\_H和CAN\_L在同一个位置中断
  9. 丢失到终端网络的连接

对于故障1-6和9，建议减小S/N比率保持总线继续工作。对于故障8，建议保持产生的子系统继续工作。对于故障7，可以选择减小S/N比率继续工作。

实践中，使用82C250类型收发器的CAN系统无法在发生故障1-7时继续工作，在发生故障8-9时可能无法继续工作。

但是，存在诸如TJA1053等容错能力强的驱动程序可以处理所有故障。通常，获得这种容错能力的代价是最大速度受到限制。对于TJA1053，最大速度是125kbps。

* **高层协议**
* CAN标准定义的是硬件（物理层 – 有几种）和通信的基本层面（数据链路层）。CAN协议本身只规定如何使用共享的通信介质将小型数据包从A点传输到B点。为了管理系统中的通信，需要一种高层协议（HLP）。

高层协议包括常见的标准，例如J1939、CANopen和CCP/XCP等。