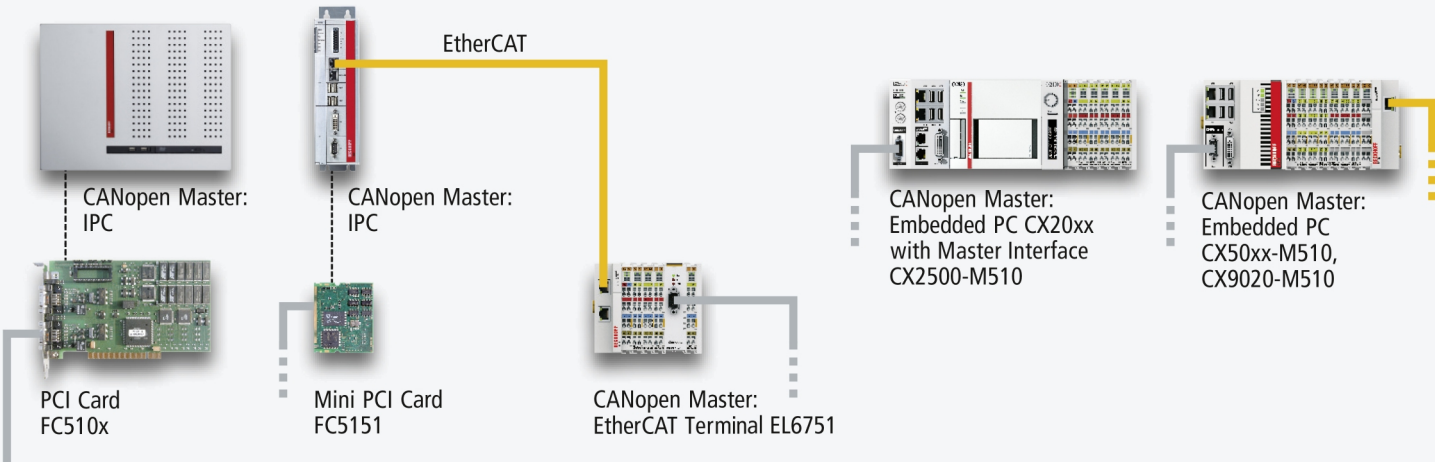


文件资料 | ZH

## CAN 接口

用于 Beckhoff CANopen 主站的 CAN 接口





# 目录

<b>1 前言</b> .....	<b>5</b>
1.1 文档说明 .....	5
1.2 安全说明 .....	6
1.3 文档发行状态 .....	7
1.4 关于文档的建议或提议 .....	7
<b>2 简介</b> .....	<b>8</b>
<b>3 TwinCAT 集成</b> .....	<b>9</b>
<b>4 缓冲区大小与循环时间的关系</b> .....	<b>10</b>
<b>5 功能</b> .....	<b>11</b>
5.1 CAN 队列 .....	11
5.1.1 快速 CAN 队列 (无缓存) .....	11
5.1.2 优化后的 CAN 队列 (缓存) .....	11
5.2 事务编号 .....	12
5.3 时间戳 .....	12
<b>6 CAN 接口的结构</b> .....	<b>13</b>
<b>7 使用筛选器</b> .....	<b>16</b>
<b>8 通过 FC532x 和 CX-M530 访问 CAN FD</b> .....	<b>17</b>
8.1 CAN FD 接口 .....	17
8.2 CAN FD 设备和波特率设置 .....	17
8.3 CAN FD 报文数据结构 .....	19
8.3.1 数据长度 .....	19
8.3.2 CAN FD bit .....	20
8.3.3 InfoData .....	21
<b>9 附录</b> .....	<b>22</b>
9.1 技术支持和服务 .....	22



# 1 前言

## 1.1 文档说明

### 目标受众

本说明仅适用于熟悉国家标准且经过培训的控制和自动化工程专家。  
在安装和调试组件时，必须遵循文档和以下说明及解释。  
操作人员应具备相关资质，并始终使用最新的生效文档。

相关负责人员必须确保所述产品的应用或使用符合所有安全要求，包括所有相关法律、法规、准则和标准。

### 免责声明

本文档经过精心准备。然而，所述产品正在不断开发中。

我们保留随时修改和更改本文档的权利，恕不另行通知。

不得依据本文档中的数据、图表和说明对已供货产品的修改提出赔偿。

### 商标

Beckhoff®、TwinCAT®、TwinCAT/BSD®、TC/BSD®、EtherCAT®、EtherCAT G®、EtherCAT G10®、EtherCAT P®、Safety over EtherCAT®、TwinSAFE®、XFC®、XTS® 和 XPlanar® 是倍福自动化有限公司的注册商标并得到授权。本出版物中使用的其他名称可能是商标，第三方出于自身目的使用它们可能侵犯商标所有者的权利。



EtherCAT® 是注册商标和专利技术，由德国倍福自动化有限公司授权使用。

### 版权所有

© 德国倍福自动化有限公司。  
未经明确授权，禁止复制、分发和使用本文件以及将其内容传达给他人。  
违者将被追究赔偿责任。在专利授权、工具型号或设计方面保留所有权利。

### 第三方品牌

本文档可能使用了第三方商标。有关商标信息，可以访问：<https://www.beckhoff.com/trademarks>

## 1.2 安全说明

### 安全规范

请注意以下安全说明和解释！  
可在以下页面或安装、接线、调试等区域找到产品相关的安全说明。

### 责任免除

所有组件在供货时都配有适合应用的特定硬件和软件配置。禁止未按文档所述修改硬件或软件配置，德国倍福自动化有限公司不对此承担责任。

### 人员资格

本说明仅供熟悉适用国家标准的控制、自动化和驱动工程专家使用。

### 警示性词语

文档中使用的警示信号词分类如下。为避免人身伤害和财产损失，请阅读并遵守安全和警告注意事项。

#### 人身伤害警告

##### ⚠ 危险

存在死亡或重伤的高度风险。

##### ⚠ 警告

存在死亡或重伤的中度风险。

##### ⚠ 谨慎

存在可能导致中度或轻度伤害的低度风险。

#### 财产或环境损害警告

##### 注意

可能会损坏环境、设备或数据。

#### 操作产品的信息



这些信息包括：  
有关产品的操作、帮助或进一步信息的建议。

## 1.3 文档发行状态

版本	注释
1.7	• 信息数据章节已更新
1.6	• CAN 接口结构章节已更新
1.5	• 使用 FC532x 和 CX-M530 访问 CAN FD 章节已更新
1.4	• 新增使用 FC532x 和 CX-M530 访问 CAN FD 章节
1.3	• 简介已更新 • 符合 IEC 82079-1 标准的安全说明
1.2	• 迁移和结构调整
1.1	• 修订版
1.0	• 首次发布

## 1.4 关于文档的建议或提议

如果您对我们的文档有任何建议或意见，请发送电子邮件至 [documentation@beckhoff.com](mailto:documentation@beckhoff.com)，并注明文档标题和版本号。

## 2 简介

倍福几乎所有的 CANopen 主站都提供所谓的 CAN 接口。CAN 接口是 CAN 协议第 2 层的一种实现。它能够接收和发送任何所需的 CAN 报文。这里，高层协议并不重要，即可以使用所有基于 CAN 的协议；然而，协议部分则必须在 PLC 中实现。

CAN 接口包含一个循环处理的缓冲区。缓冲区可以存储 11 至 32 个数据报文。

发送缓冲区 (Tx) 用于存储待发送的数据，而接收缓冲区 (Rx) 则用于存储已接收的数据。根据 CAN 总线主控制器的不同，可以接收或发送 11 bit 或 29 bit 的数据报文。缓冲区的处理与任务的循环时间同步进行。因此，如果缓冲区容量为 10，则每个任务周期内最多可以发送或接收 10 个 CAN 报文。

11 bit 标识符，也称作“基础帧格式” (CAN 2.0A)

29 bit 标识符，也称为“扩展帧格式” (CAN 2.0B)

### CAN 接口 - 支持的功能

	CAN2.0A 11 bit ID	CAN2.0B 29 bit ID	CAN FD	快速 CAN 队列 <sup>1</sup>	优化后的 CAN 队列 <sup>1</sup>	事务编号 [▶ 12]	时间戳 <sup>2</sup>
EL6751 传统映射	☒	☒	-	-	☒	☒ <sup>3</sup>	-
EL6751 MDP 映射	☒	☒	-	☒	☒	☒	-
CCAT	(☒) <sup>4</sup>	☒	-	-	-	☒ 自 FW 1.17 起	☒ 自 FW 1.17 起
CX1500-M510	☒	☒	-	-	-	-	-
FC510x、 FC5151	☒	☒ 自 FW 2.14 起	-	-	-	-	-
FC532x、 CX-M530 <sup>5</sup>	☒	☒	☒	-	-	☒	☒

<sup>1</sup>) 不在 29 bit 模式中，不带事务编号 [▶ 12]

<sup>2</sup>) 仅在 29 bit 模式中，带事务编号 [▶ 12]

<sup>3</sup>) 仅在 29 bit 模式中

<sup>4</sup>) 由 29 bit 选项涵盖

<sup>5</sup>) 准备中

### ● CCAT

什么是 CCAT？这是由倍福的哪些产品所支持的？

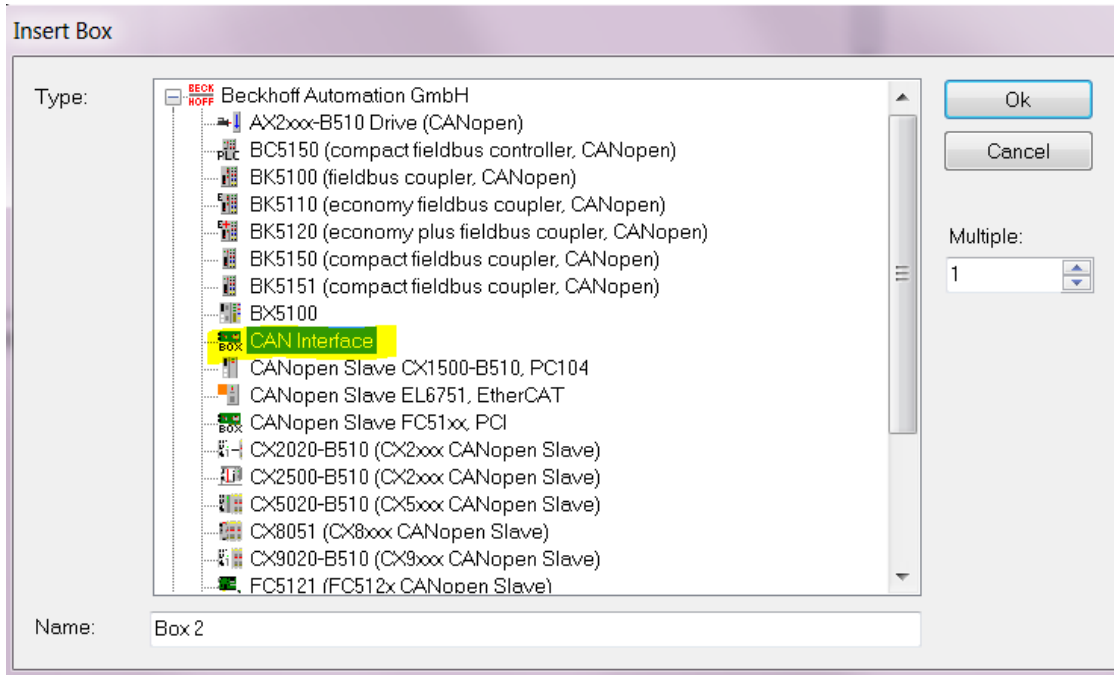
CCAT 接口是倍福公司当前的 CAN 通信解决方案，适用于倍福的 PCI-Express 板卡以及倍福嵌入式 PC 的板载接口。例如，以下产品仅适用于 CANopen 主站：

C20xx-M510、CX51xx-M510、CX8x50、CX9x20-M510、FC512x



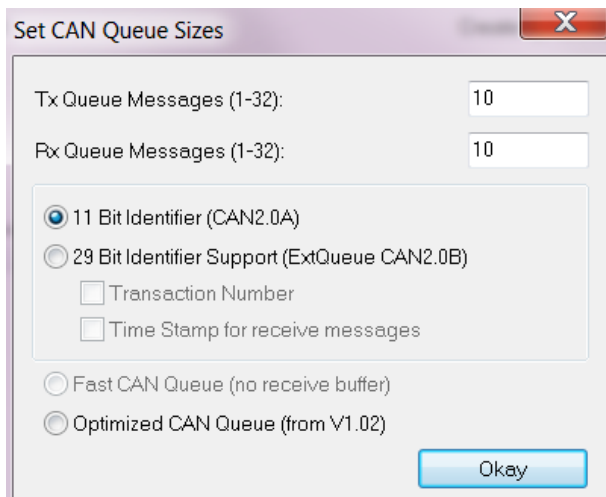
### 3 TwinCAT 集成

如果您在 TwinCAT 软件中配置了一个 CANopen 主站，那么在选择接口时，您应挑选 CAN 接口，而非 CANopen 从站。



附图 1: 选择 CAN 接口

接下来，系统将提示您选择想要使用的相应接口，或确定缓冲区的大小。如果您之后需要更改接口，请注意，通常接口会根据所选模式重新设置，这意味着原有的链接可能会被删除。另外，请确保仅选择您的硬件实际支持的模式（请参见 *CAN 接口支持功能表*）。



附图 2: 选择帧格式

## 4 缓冲区大小与循环时间的关系

您需要预估缓冲区的大小，或确定任务周期的快慢。下表将帮助您提前确定这一点。

CCAT CAN 主站的内存可存储 512 条消息，而 EL6751 的内存则可存储 150 条接收消息。若无法及时从内存中读取数据，则数据将会丢失。由于未给出任何提示，因此应预估最坏的情况，或者尽可能确保变量 NoOfRxMessages 的值小于缓冲区的最大值。如果该变量值始终或几乎在每个周期都达到最大值，那么这就说明接收到的数据量超出了每个周期所能记录的数据量。解决方案：缩短任务周期或增加 CAN 队列的缓冲区大小。

### 示例

在 500 Kbit/s 的传输速率下，一个包含 11 bit 标识符和 8 字节用户数据的 CAN 报文大约需要 260  $\mu$ s 的时间来传输。若在最坏情况下假设总线负载达到 100%，则在 1 ms 内最多能传输 3 个报文。这意味着，在这种情况下，一个最多能存储 4 个报文的缓冲区就足够了。如果将任务周期设置为 5 ms 而非 1 ms，那么缓冲区的大小至少应为 20 (5000  $\mu$ s / 260  $\mu$ s)。需要注意的是，本研究仅考虑了一个方向上的数据传输，且假定 CAN 报文中的数据部分始终为 8 字节。由于通常不会假设总线负载达到 100%，因此，我们还可以通过观察变量 NoOfRxMessages 的值，来判断它在大多数情况下是否低于已创建缓冲区的最大容量。如果 NoOfRxMessages 的值经常达到最大，那么就on应该考虑缩短任务周期或增加缓冲区的大小。

### 最坏情况

不过，CAN 接口的设计通常遵循这样一个原则，即数据的检索速度通常总是快于数据进入缓冲区的速度。

### 示例

1 MBaud 数据长度 0 意味着每个 CAN 消息 50  $\mu$ s。如果任务周期为 1 ms (即 1000  $\mu$ s)，而每个 CAN 报文的传输时间为 50  $\mu$ s，那么在 1 ms 内最多可以传输

$$1000 \mu / 50 \mu = 20$$

个报文。这意味着，即使在最坏的情况下，一个容量为 20 的缓冲区也足够用来接收所有的 CAN 报文。

表格：11 bit 标识符 CAN 报文传输时间 [ms]<sup>1)</sup>

比特率 [kbit/s]	数据长度 (字节)								
	0	1	2	3	4	5	6	7	8
50	1.09	1.28	1.47	1.66	1.86	2.05	2.24	2.34	2.62
125	0.44	0.51	0.59	0.67	0.74	0.82	0.90	0.97	1.05
250	0.22	0.26	0.29	0.33	0.37	0.41	0.45	0.49	0.52
500	0.11	0.13	0.15	0.17	0.19	0.21	0.22	0.24	0.26
1000	0.05	0.06	0.07	0.08	0.09	0.10	0.11	0.12	0.13

<sup>1)</sup>CIA 的数据

## 5 功能

### 5.1 CAN 队列

#### 5.1.1 快速 CAN 队列（无缓存）

EL6751 不再对接收到的消息进行缓存处理。因此，RxQueue 应该足够大，以便理论上在一个 EtherCAT 周期内能够接收到的所有消息都能容纳在内。

接收到的信息不再需要确认。EL6751 在接收到新消息时增加 RxCounter 的值。

在传输方向上也是如此，只有依赖于 TxCounter 和 NoOfTxMessages 变化的数据会被复制，因此队列中并行消息的数量实际上对运行时没有进一步的影响（只有 NoOfTxMessages 有影响）。

EL6751 采用 3 缓存区模式运行（这意味着它始终保留一个空闲缓冲区用于存储接收到的 CAN 消息），因此，在某些情况下，那些**未被使用的**缓冲区中可能存储了错误或过时的数据。

对象字典可在 *CoE-Online* 选项卡上读取。如果索引 0x1C32:08 已经/正在设置为 1，则将测量 EL6751 的本地循环时间并将其存储在索引 0x1C32:05 中（最大值）。因此，您可以查看 EL6751 是否会在 EtherCAT 周期内完成。

快速 CAN 队列中可能不包含其他 CANopen 或 CAN 层2节点。

#### 5.1.2 优化后的 CAN 队列（缓存）

EL6751 接收到的消息被缓存。EL6751 以单缓模式运行。

以下适用于快速 CAN 队列和优化 CAN 队列这两个功能：

优势

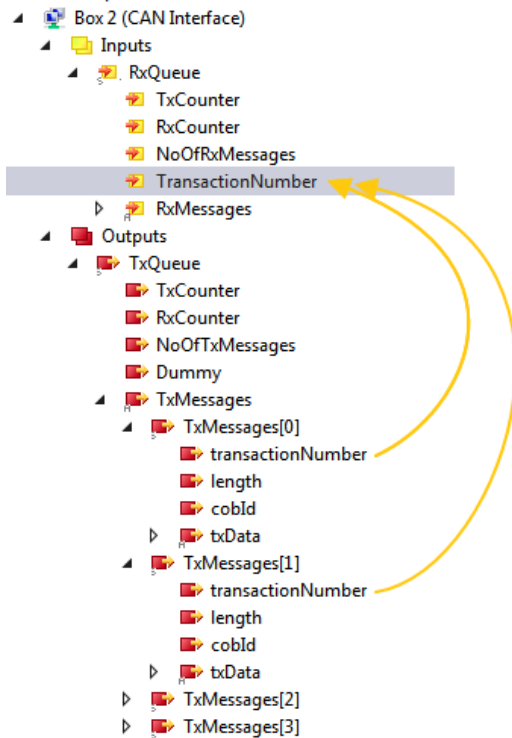
- 处理速度更快
- 快速 CAN 队列不包含任何附加组件，因此注定能最快地处理/响应来自总线的数据。

缺点

- 两种模式都只支持 11 bit 标识符。
- 不得使用筛选器。

## 5.2 事务编号

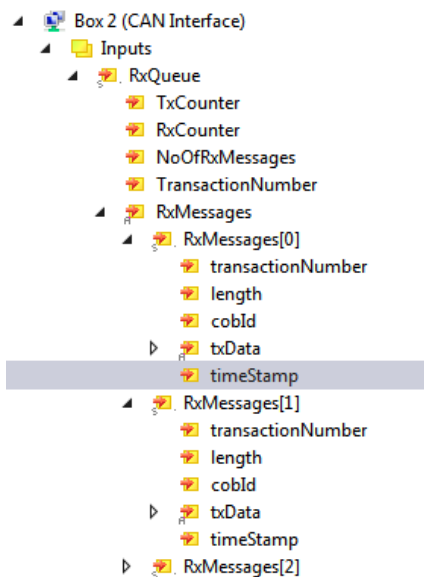
根据事务编号，可以确定在上一 CAN 周期中发生了传输的 CAN 队列中的 CAN 消息。通过单个 TxMessages[n]，可以输入任何事务编号（例如序列号）。每个 CAN 周期结束时，会将最后发送的 TX 消息的事务编号记录在 Inputs.RxQueue.TransactionNumber 中。



附图 3: 事务编号

## 5.3 时间戳

基于 CCAT 的 CAN 控制器（例如 FC512x 和 M510）会利用接收时间戳（64 bit 纳秒分辨率整型值）来记录 CAN 帧到达的具体时间。



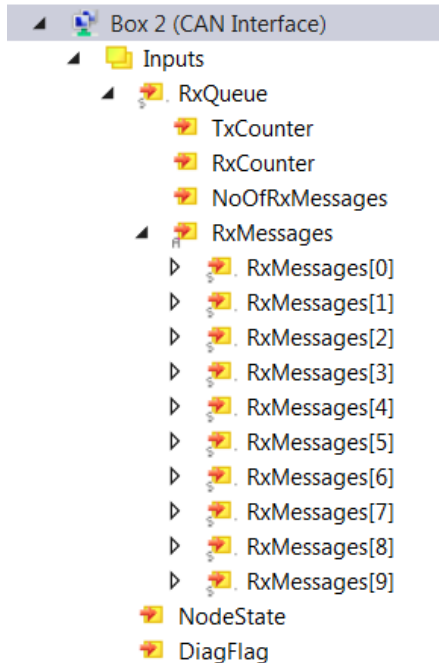
附图 4: 时间戳

## 6 CAN 接口的结构

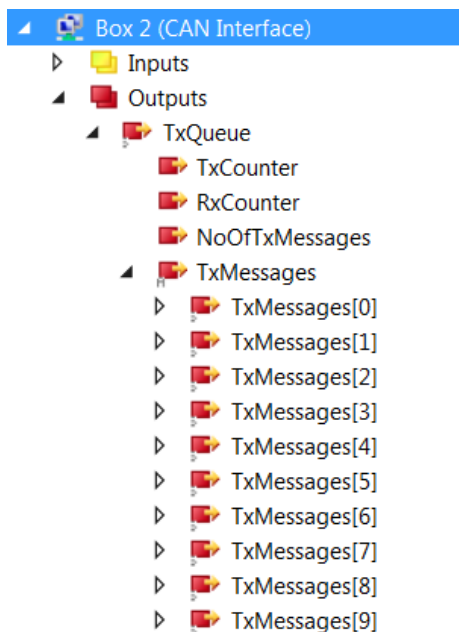
根据所选选项的不同，CAN 接口的显示也不同。11 bit 标识符接口的结构与 29 bit 标识符接口不同。此外，接口还可能包含事务编号和时间戳。在采用结构体时，需要考虑目标系统及其所兼容的数据对齐格式。在 TwinCAT 3 下，可以使用对应的属性 {attribute 'pack\_mode':='0'}。

该接口包括与其他接口的通信功能，并能处理最多 32 个 CAN 消息。只有在对输出进行写入操作后，才能读取输入内容。

接口的访问方式如下：



附图 5: CAN 接口 – 输入



附图 6: CAN 接口 – 输出

如果要传输数据，Outputs.TxCounter 将设置为 +1。NoOfTxMessages 还表示将从缓存区中发送多少条消息。RxCounter 表示缓存区中是否有新数据。NoOfRxMessages 表示缓存区中有多少新数据。

数据获取完成后，请设置 `Outputs.RxCounter:=Inputs.RxCounter`。之后，CAN 接口便知道可以重新向缓存区中写入数据了。所有数据都必须及时读出，因为 CAN 接口会在需要时重新填充所有消息结构。

### 传输代码示例

```

if (Outputs.TxCounter = Inputs.TxCounter)           // check if the interface is ready
and NumberOfMessagesToSend >0 then                 // and messages are to send
  for i:=0 to NumberOfMessagesToSend-1 do          // LOOP for copying the CAN message to the
    Outputs.TxMessage[i] := MessageToSend[i];     // interface
  End for                                           // copy
  Outputs.NoOfTxMessages := NumberOfMessagesToSend; // number of CAN messages you are going to send
  Outputs.TxCounter := Inputs.TxCounter + 1;       // inc. shows the CAN interface that new data
                                                    // is available and to send this data
end_if

```

### 读取代码示例

```

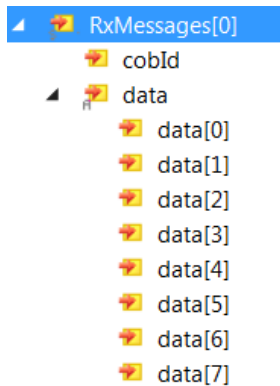
if Outputs.RxCounter <> Inputs.RxCounter then       // check if new data is in the buffer
  for i := 0 to (Inputs.NoOfRxMessages-1) do      // start the LOOP and check how much data
    MessageReceived[i] := Inputs.RxMessage [i];  // is in the buffer
  End for                                           // copy the CAN message
  Outputs.RxCounter := Inputs.RxCounter;          // set equal: the CAN interface then knows,
                                                    // that you have copied the CAN data
end_if

```

### 使用 11 bit 标识符时的报文结构

当采用 11 bit 标识符时，报文结构会包含 2 字节的 COB ID 以及 8 字节的数据。COB ID 文件结构如下：

- Bit 0–3: 数据长度 (0 … 8)
- Bit 4: RTR
- Bit 5–15: 11 bit 标识符



附图 7: 使用 11 bit 标识符时的报文结构

由于 11 bit 标识符将 COB ID、数据长度和 RTR 位都编码在一个字内，因此以下示例将帮助我们该字中解析出相关数据。在此选择一个结构来存储解码数据。

```
IF RXCounter_Out <> RXCounter_In THEN
  FOR I := 0 TO (NoOfTxMessages-1) DO
    stCANInterfaceMessageValue[i].Lengh:=WORD_TO_BYTE(stCANInterfaceMessage[i].CobID) AND 16#0F;
    stCANInterfaceMessageValue[i].RTR:=stCANInterfaceMessage[i].CobID.4;
    stCANInterfaceMessageValue[i].CobID :=ROR(stCANInterfaceMessage[i].CobID,5) AND 16#07FF;
    stCANInterfaceMessageValue[i].Data := stCANInterfaceMessage[i].Data;
    CASE stCANInterfaceMessageValue[i].CobID OF
      16#318: COB318:=COB318+1;
      16#718: COB718:=COB718+1;
      16#1CD: COB1CD:=COB1CD+1;
      memcpy(ADR(TempValue),ADR(stCANInterfaceMessage[i].Data[6]),2);
      16#1ED: COB1ED:=COB1ED+1;
    ELSE
      COBALLOther:=COBALLOther+1;
    END_CASE
  End_for
  RXCounter_Out:=RXCounter_In;
END_IF
```

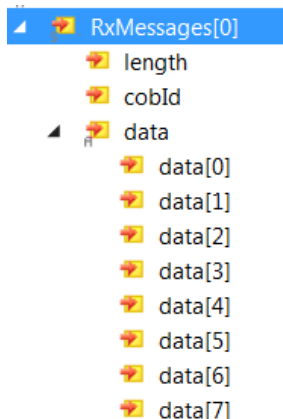
### 使用 29 bit 标识符时的报文结构

使用 29 bit 标识符时，报文结构由 长度 [2 字节]，COB ID [4 字节]和 8 字节数据组成。

长度：数据长度 (0 … 8)

COB ID 文件结构如下：

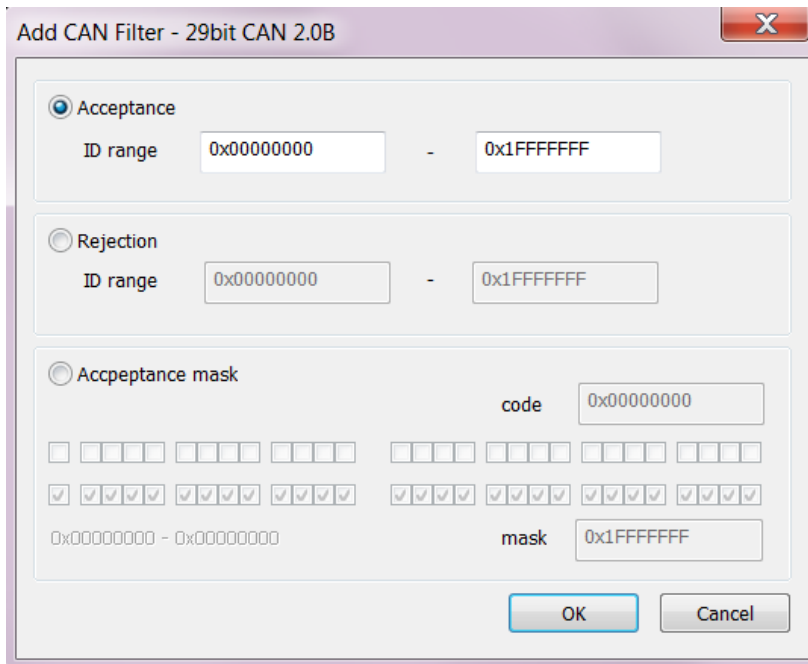
- Bit 0–28: 29 bit 标识符
- Bit 30: RTR
- Bit 31:
  - 0: 11 bit 标识符,
  - 1: 29 bit 标识符



附图 8: 使用 29 bit 标识符时的报文结构

## 7 使用筛选器

如果不希望接收 CAN 接口中的所有报文，可以选择设置筛选器。这样做可以减少 CAN 接口中的报文数量，从而确保只有实际需要的报文才会被接收。



附图 9: CAN 筛选器

接受：

在此输入要转发到 CAN 接口的标识符。

拒绝：

在此输入不转发到 CAN 接口的标识符。

接受掩码：

在此处按位指定要转发至 CAN 接口的标识符。

### 基于 29 bit 标识符的示例

在示例中，标识符为 0x0400 ... 0x0700 的所有报文都会被发送至 CAN 接口。信息旁边会显示一个“+”。

“+”表示筛选器允许数据通过 CAN 接口（接受）

“-”表示筛选器不允许数据通过 CAN 接口（拒绝）

CAN Rx	Acceptance	Rejection	Info	Comment
Filter 1		0x00000000 - 0x000003FF	-	
Filter 2	0x00000400 - 0x00000700		+	manually added (code/mask)
Filter 3		0x00000701 - 0x1FFFFFFF	-	

附图 10: 基于 29 bit 标识符的示例



## 8 通过 FC532x 和 CX-M530 访问 CAN FD

### 发送和接收 FD 信息

本章介绍 CAN FD 接口的 CAN FD 功能。

### 8.1 CAN FD 接口

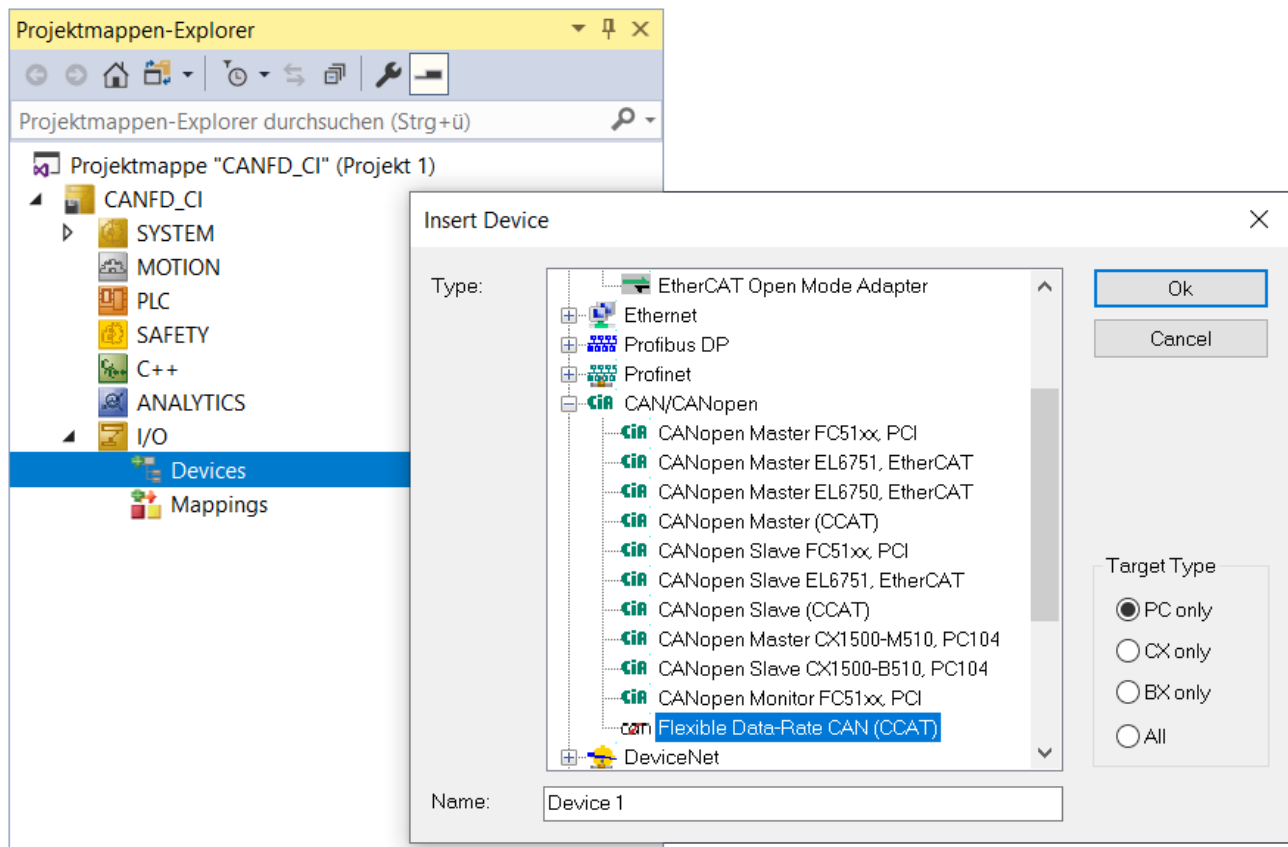
FC532x、CX2500-M530 的 CAN 接口以及 CX-M530 CAN 的可选接口均支持对 CAN FD 功能的访问。

CAN 接口的操作以及 *事务编号* 和 *时间戳* 功能与已知接口一致（参见 [CAN 接口的结构 \[▶ 13\]](#) 章节）。

RX 和 TX 队列采用了新的消息数据类型，并且支持新的波特率设置。

### 8.2 CAN FD 设备和波特率设置

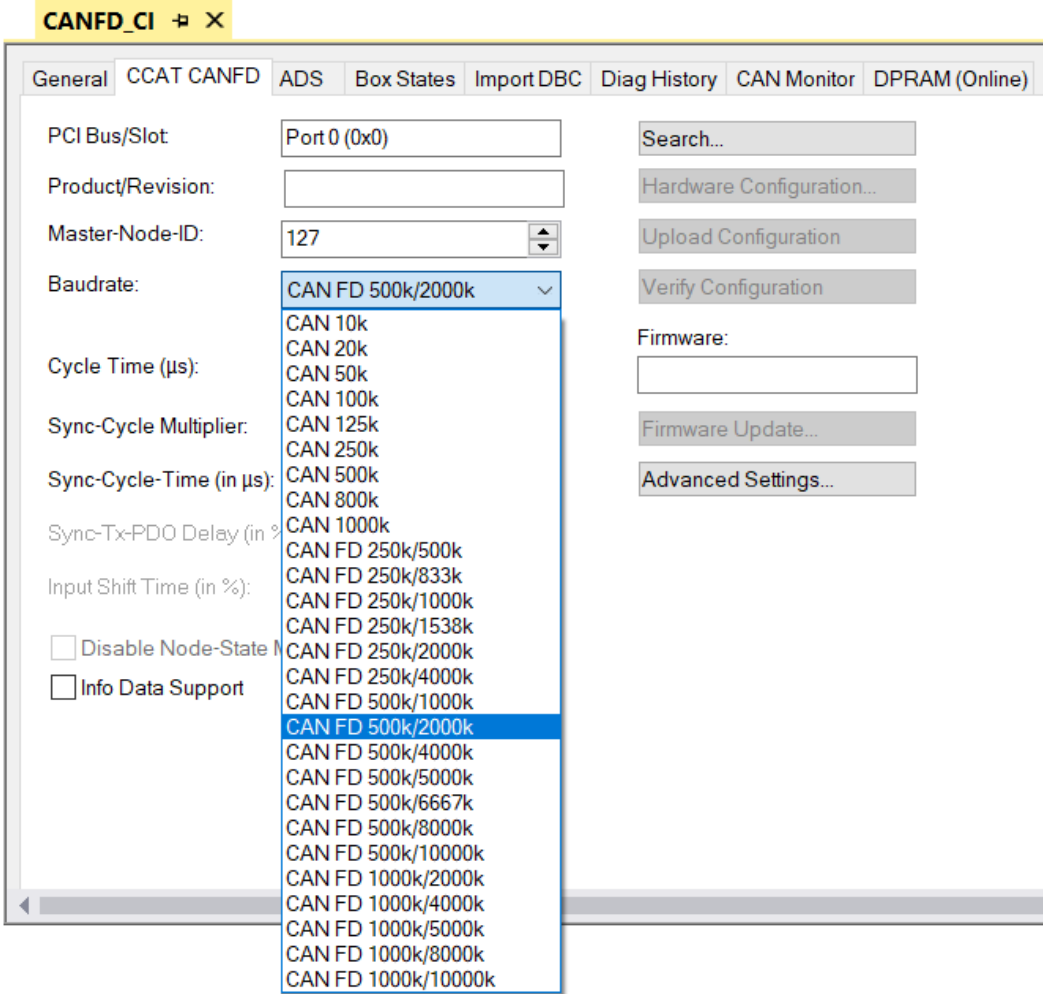
CAN FD 功能可通过设备 *Flexible Data-Rate CAN (CCAT)* 访问。



附图 11: 灵活数据速率 CAN (CCAT)

CAN FD 的仲裁阶段和数据传输阶段可以采用不同的波特率。与传统 CAN 一样，您也可以为 CAN FD 的两段设置相同的波特率。

可能出现以下波特率：



附图 12: 可能的波特率

第一个数字代表仲裁段的波特率，而第二个数字则代表数据段的波特率。

**● 关于波特率 10 Mbit/s 的说明**

**i** 目前，只有在非常理想的条件下才可能实现 10 Mbit/s 的波特率，并且这一速率目前并不适用于实际应用场景中的配置。CAN FD 目前常用的设置是 500k/2000k。

## 8.3 CAN FD 报文数据结构

为了支持 CAN 接口上的 CAN FD 功能，我们引入了以下新的数据结构。

```

TYPE CANFDTSRXQUEUE :
  STRUCT
    dataLength : BYTE;
    EDL : BIT;
    BSR : BIT;
    ESI : BIT;
    cobId : UDINT;
    rxData : CANFDMESSAGE;
    timeStamp : ULINT;
  END_STRUCT
END_TYPE

TYPE CANFDTXQUEUE :
  STRUCT
    transactionNumber : UINT;
    dataLength : BYTE;
    EDL : BIT;
    BSR : BIT;
    ESI : BIT;
    cobId : UDINT;
    txData : CANFDMESSAGE;
  END_STRUCT
END_TYPE

TYPE CANFDMESSAGE :
  ARRAY [0..63] OF USINT;
END_TYPE

```

这些结构可用于 System Manager 和 PLC 中的 IO。

### 8.3.1 数据长度

Can FD 帧的数据长度可以达到 64 字节。由于这些数值是在传统 CAN 的仲裁报头中传输的，因此可能会出现以下数值：

0 ... 8、12、16、20、24、36、48 和 64

如果传输过程中数据长度与上述数值不符，设备会自动将其调整至下一个更高的数值。

在 CAN FD 帧中，DLC 字段的有效取值范围是 0 到 15。DLC 字段的值决定了数据字段中的字节数量，其解释方式如下：

DLC value	Data Field size (Bytes)	DLC value	Data Field size (Bytes)
0 .. 8	0 .. 8	12	24
9	12	13	36
10	16	14	48
11	20	15	64

附图 13: DLC 数值对比 数据字段大小

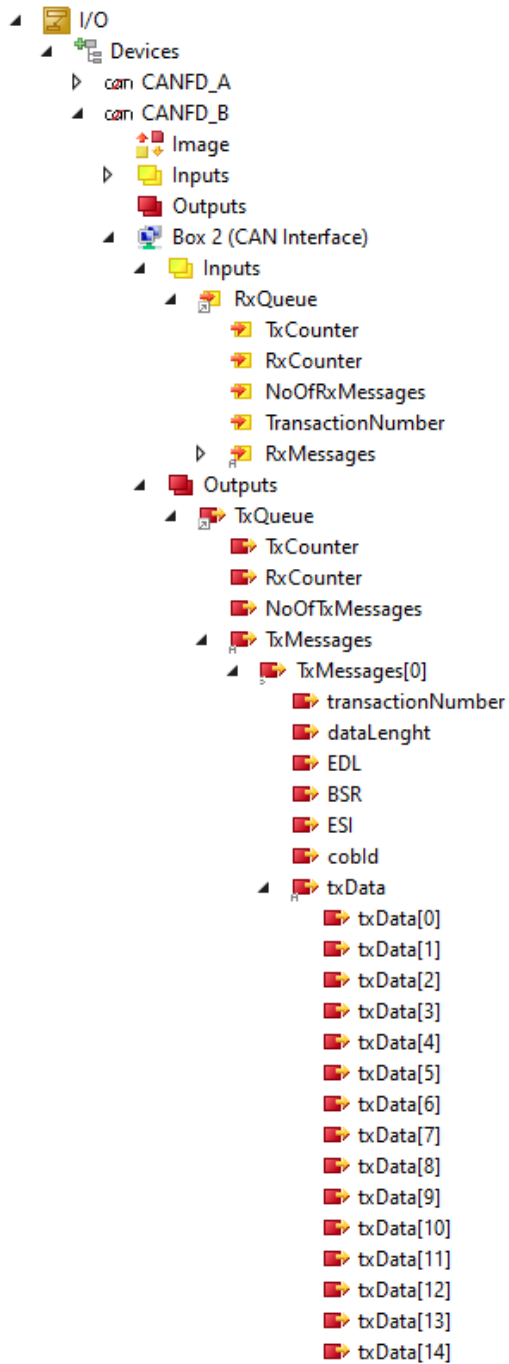
CAN 接口会自动完成这种转换，也就是说，在 CAN 接口上，您只需指定实际的数据字节数（允许的范围是 0 到 64 字节）。

CAN 接口会在后台自动使用下一个更大的 DLC 值来完成向 DLC 值的转换。

示例：如果您在 CAN 接口中输入数据长度为 32，则会使用 DLC 值 13 发送 36 字节。

### 8.3.2 CAN FD bit

与传统 CAN 接口相比，数据结构队列中新增的 bit 具有以下含义。



附图 14: 数据结构队列

#### EDL

**EDL** (扩展数据长度) 位用于确定发送的是 FD 帧还是传统帧，或者接收到的帧是哪种类型。

#### BSR

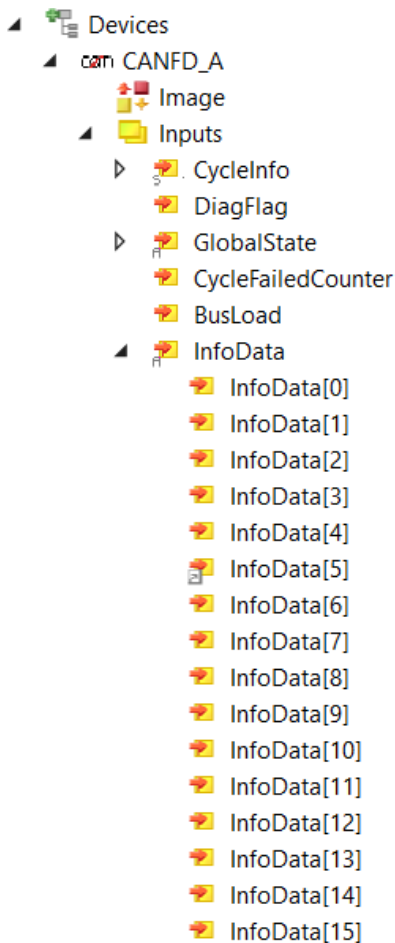
**BSR** (比特率切换) bit 指定在数据阶段是否应切换到更高的波特率，或者帧是如何被接收的。

#### ESI

**ESI bit** (错误状态指示) 表明帧在接收过程中是否出错 (Rx) 或在发送过程中是否存在错误 (Tx)。

### 8.3.3 InfoData

FC532x / CX-M530 的信息数据中可以读取到以下信息：



附图 15: InfoData

#### InfoData[0] – CAN 状态

Bit0 – 总线关闭 (0 无错误 – 1 总线关闭错误)  
 Bit1 – 被动错误  
 Bit2 – 节点活跃  
 Bit3 – 警告限制  
 Bit4 – 过载  
 Bit6 – 总线空闲；

#### InfoData[1] – 仲裁阶段波特率

Byte0 = 跳转宽度  
 Byte1 = 时间 A  
 Byte2 = 时间 B  
 Byte3 = 预分频器；

#### InfoData[2] – 数据阶段波特率

Byte0 = 跳转宽度  
 Byte1 = 时间 A  
 Byte2 = 时间 B  
 Byte3 = 预分频器；

#### InfoData[3] – 收发器延迟时间 [1 TQ] 时间量子，仅 CAN FD；

#### InfoData[4] – 发送 FIFO (高优先级) 已用字数；

#### InfoData[5] – 发送 FIFO (低优先级) 已用字数；

#### InfoData[6] – 接收 FIFO 计数器；

#### InfoData[7] – AckError 计数器；

#### InfoData[8] – BitError 计数器；

#### InfoData[9] – CrcError 计数器；

#### InfoData[10] – FormError 计数器；

**InfoData[11] – CAN 内核**  
 = 1 (CAN 内核正在运行) ]  
 = 0 (CAN 内核未运行) ]；

#### InfoData[12] – 填充位错误计数器；

**InfoData[13] – 接收错误计数器** (仅在支持 CAN FD 时可用) ；

**InfoData[14] – 发送错误计数器** (仅在支持 CAN FD 时可用) ；

**InfoData[15] – 未使用；**

## 9 附录

### 9.1 技术支持和服务

倍福公司及其合作伙伴在世界各地提供全面的技术支持和服务，对与倍福产品和系统解决方案相关的所有问题提供快速有效的帮助。

#### 倍福分公司和代表处

有关倍福产品本地支持和服务方面的信息，请联系倍福分公司或代表处！

世界各地倍福分公司和代表处的地址可参见以下网页：<http://www.beckhoff.com>

该网页还提供更多倍福产品组件的文档。

#### 支持

倍福支持部门提供全面的技术援助，不仅帮助使用各种倍福产品，还提供其他广泛的服务：

- 技术支持
- 复杂自动化系统的设计、编程和调试
- 以及倍福系统组件的各种培训课程

热线电话： +49 5246 963 157  
电子邮箱： support@beckhoff.com  
网址： [www.beckhoff.com/support](http://www.beckhoff.com/support)

#### 服务

倍福服务中心提供所有售后服务：

- 现场服务
- 维修服务
- 备件服务
- 热线服务

热线电话： +49 5246 963 460  
电子邮箱： service@beckhoff.com  
网址： [www.beckhoff.com/service](http://www.beckhoff.com/service)

#### 德国总部

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20  
33415 Verl  
Germany

电话： +49 5246 963 0  
电子邮箱： info@beckhoff.com  
网址： [www.beckhoff.com](http://www.beckhoff.com)

## 图表目录

附图 1	选择 CAN 接口 .....	9
附图 2	选择帧格式 .....	9
附图 3	事务编号 .....	12
附图 4	时间戳 .....	12
附图 5	CAN 接口 - 输入 .....	13
附图 6	CAN 接口 - 输出 .....	13
附图 7	使用 11 bit 标识符时的报文结构 .....	14
附图 8	使用 29 bit 标识符时的报文结构 .....	15
附图 9	CAN 筛选器 .....	16
附图 10	基于 29 bit 标识符的示例 .....	16
附图 11	灵活数据速率 CAN (CCAT) .....	17
附图 12	可能的波特率 .....	18
附图 13	DLC 数值对比 数据字段大小 .....	19
附图 14	数据结构队列 .....	20
附图 15	InfoData .....	21

## **Trademark statements**

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.



Beckhoff Automation GmbH & Co. KG  
Hülshorstweg 20  
33415 Verl  
Germany  
电话号码: +49 5246 9630  
info@beckhoff.com  
www.beckhoff.com