第56章 具有灵活数据速率的控制器局域网（CAN FD）

本章包括下列主题：

56.1 简介 56-2
56.2 CAN FD报文帧 56-5
56.3 控制寄存器 56-9
56.4 工作模式 56-74
56.5 配置 56-80
56.6 报文发送 56-89
56.7 发送事件FIFO —— TEF 56-98
56.8 报文过滤 56-105
56.9 报文接收 56-110
56.10 FIFO行为 56-117
56.11 时间戳 56-129
56.12 中断 56-130
56.13 错误处理 56-137
56.14 相关应用笔记 56-139
56.15 版本历史 56-140

> **注：** 本系列参考手册章节旨在用作对器件数据手册的补充。本手册章节可能不适用于某些PIC32器件，具体取决于器件型号。
>
> 请参见最新器件数据手册中 **“具有灵活数据速率的控制器局域网 （CAN FD）”** 章节开头部分的注，确定本文档是否支持您所使用的器件。
>
> 器件数据手册和系列参考手册章节可从Microchip网站下载：http://www.microchip.com。

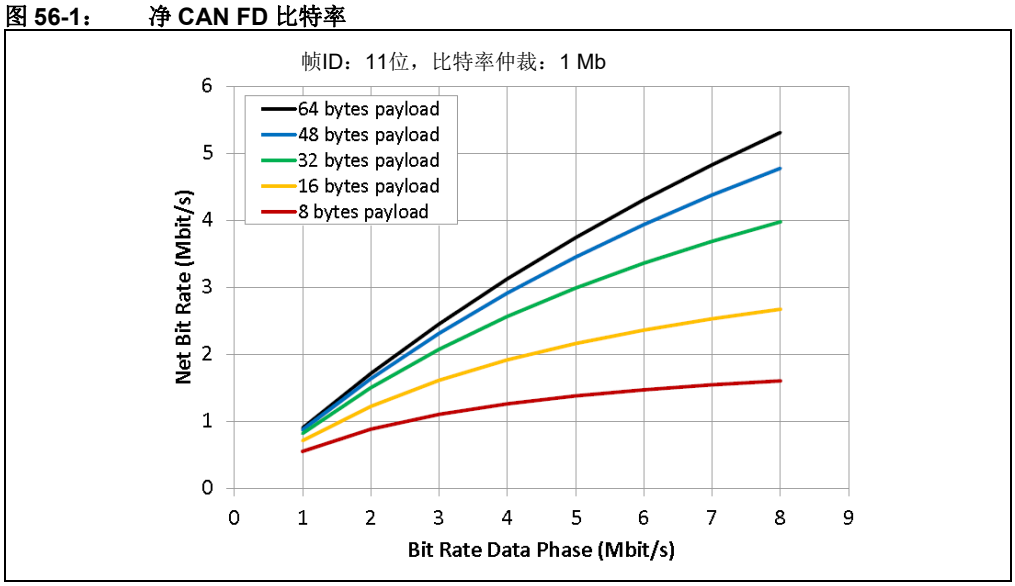
56.1 简介

CAN灵活数据速率 （Flexible Data-Rate， FD）可满足对CAN总线不断增长的带宽需求。 CAN 2.0B具有以下两个增强功能：

- 数据字段增加至最大64个数据字节 （目前为8个字节）
- 可以在仲裁字段之后切换到更快的比特率

图56-1显示了较高数据比特率（Data Bit Rate，DBR）和每帧增加的数据字节可能带来的净比特率增加 （© Robert Bosch GmbH）。

图 56-1： 净 CAN FD 比特率



CAN FD协议定义为允许CAN 2.0报文和CAN FD报文在同一总线上共存。这并不表示可以在同一总线上混合使用非CAN FD控制器与CAN FD控制器。非CAN FD控制器在接收CAN FD报文时将生成错误帧。

 © 2019 Microchip Technology Inc.

56.1.1 特性

CAN FD模块具有以下特性：

通用

- 最高1 Mbps的标称 （仲裁）比特率
- 最高8 Mbps的数据比特率
- CAN FD控制器模式：
  - CAN 2.0B和CAN FD混合模式
  - CAN 2.0B模式
- 符合ISO11898-1:2015标准

报文FIFO

- 31个可配置为发送或接收的FIFO
- 1个发送队列 （Transmit Queue， TXQ）
- 带32位时间戳的发送事件FIFO （Transmit Event FIFO， TEF）

报文发送

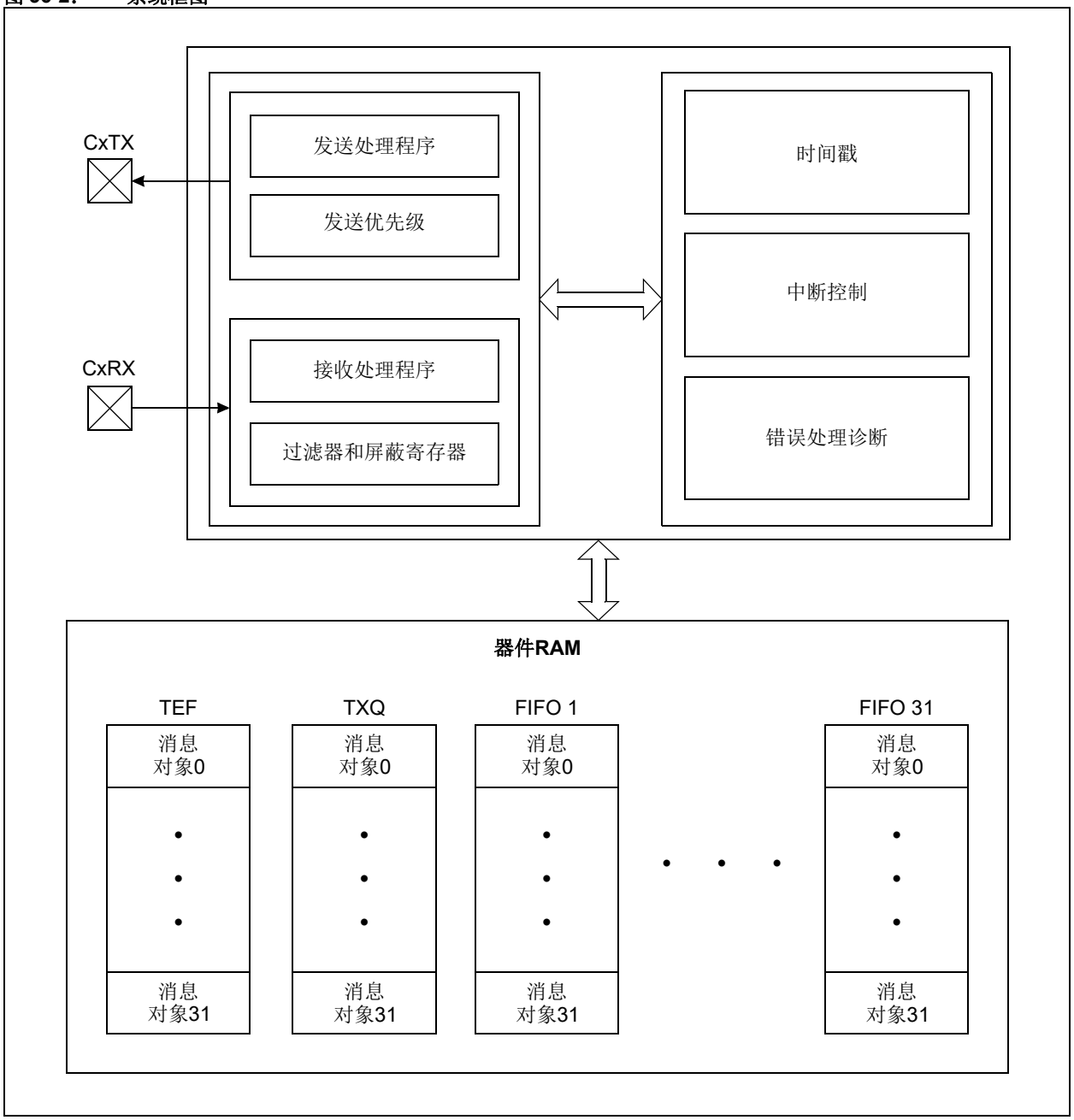
- 报文发送优先级取决于以下任一或两个因素：
  - 基于优先级位域
  - 使用TXQ先发送ID最低的报文
- 可编程自动重发尝试：不限次数、 3次尝试或禁止尝试

报文接收

- 32个灵活的过滤器和屏蔽器对象
- 可以使用以下任一项将每个对象配置为过滤器：
  - 标准ID和前18个数据位
  - 扩展ID
- 32位时间戳
- CAN FD比特流处理器 （Bit Stream Processor， BSP）实现了ISO11898-1:2015标准中说明的CAN FD协议介质访问控制。它可以对比特流进行序列化和反序列化处理、对CAN FD帧进行编码和解码、管理介质访问、应答帧以及检测错误和发送错误信号。
- 发送处理程序会优先处理发送FIFO请求发送的报文。该处理程序通过RAM接口从RAM中获取发送数据并将其提供给BSP进行发送。
- BSP向接收处理程序提供接收到的报文。接收处理程序使用接收过滤器来过滤将存储在接收FIFO中的报文。该处理程序使用RAM接口将接收到的数据存储到RAM中。
- 每个FIFO都可以配置为发送或接收FIFO。 FIFO控制会跟踪FIFO头部和尾部，并计算用户地址。在发送FIFO中，用户地址指向RAM中用于存储下一个发送报文的数据的地址。在接收FIFO中，用户地址指向RAM中用于读取下一个接收报文的数据的地址。通过递增FIFO的头部/尾部，用户可以通知FIFO已向RAM写入或从中读取了报文。
- TXQ是一个特殊的发送FIFO，它会根据队列中存储的报文的ID发送报文。
- TEF用于存储已发送报文的报文ID。
- 自由运行的时基计数器 （Time Base Counter， TBC）用于为接收的报文添加时间戳。 TEF中的报文也可以添加时间戳。
- CAN FD控制器模块会在接收到新的报文时或在成功发送报文时产生中断。

图56-2显示了系统框图。

图 56-2： 系统框图



 © 2019 Microchip Technology Inc.

56.2 CAN FD报文帧

ISO11898-1:2015中详细说明了不同的CAN报文帧。图56-3至图56-8说明并总结了报文和字段的结构。

有四种不同的CAN数据或远程帧 （见图56-4）：

- **CAN基本帧：**使用标准ID的经典CAN 2.0帧
- **CAN FD基本帧：**使用标准ID的CAN FD帧
- **CAN扩展帧：**使用扩展ID的经典CAN 2.0帧
- **CAN FD扩展帧：**使用扩展ID的CAN FD帧

CAN FD帧中没有远程帧；因此，将RTR位替换为RRS位 （见图56-4）。可以使用CAN FD基本帧中的RRS位将SID扩展至12位。使能后， RRS位用作SID<11:0>的LSB，称为SID11。

图56-5指定了不同CAN报文的控制字段。在将CAN FD添加到ISO11898-1:2015之前， FDF位为保留位。现在， FDF位可以选择经典格式或CAN FD格式。

BRS位用于选择是否应在CAN FD帧的数据阶段切换比特率。图56-8展示了错误帧和过载帧。这些特殊的帧不会发生更改。

> **注：** 如果在CAN FD帧的数据阶段期间检测到错误，则比特率将切换回标称比特率（Nominal Bit Rate， NBR）。错误帧始终以仲裁比特率发送。

56.2.1 ISO CRC与非ISO CRC

为支持非ISO CRC ECU的系统验证，CAN FD控制器模块支持ISO CRC（根据ISO11898-1:2015）和非ISO CRC （见图56-6和图56-7）。 CRC字段可以使用ISOCRCEN位 （CFDxCON<5>）进行选择。ISO CRC字段包含填充计数。原始CAN FD规范中不包含该计数；添加该计数的目的是修正原始规范错误检测中的一个小问题。

CAN FD帧使用两种不同长度的CRC：17位 （不超过16个数据字节时使用）和21位 （至少20个数据字节时使用）。从技术上说， CAN FD中共有六个不同的CAN数据/远程帧。

图 56-3： 通用数据帧

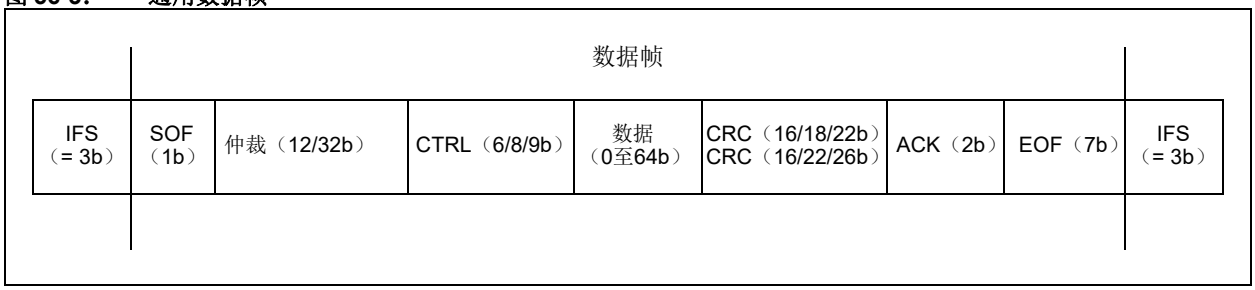


图 56-4： 仲裁字段

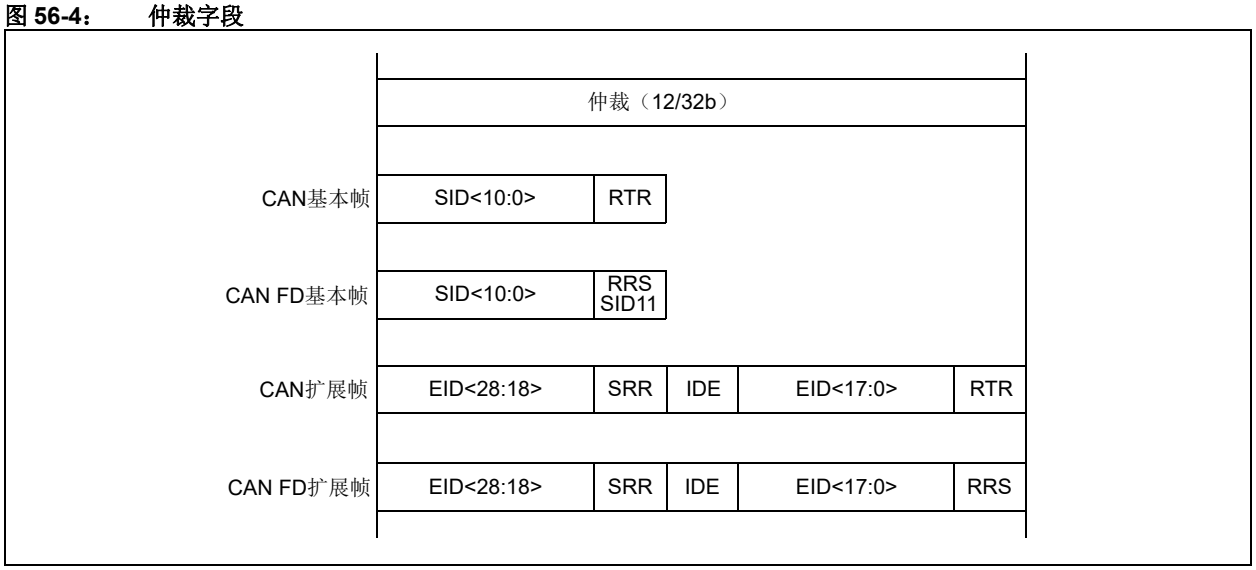


图 56-5： 控制字段

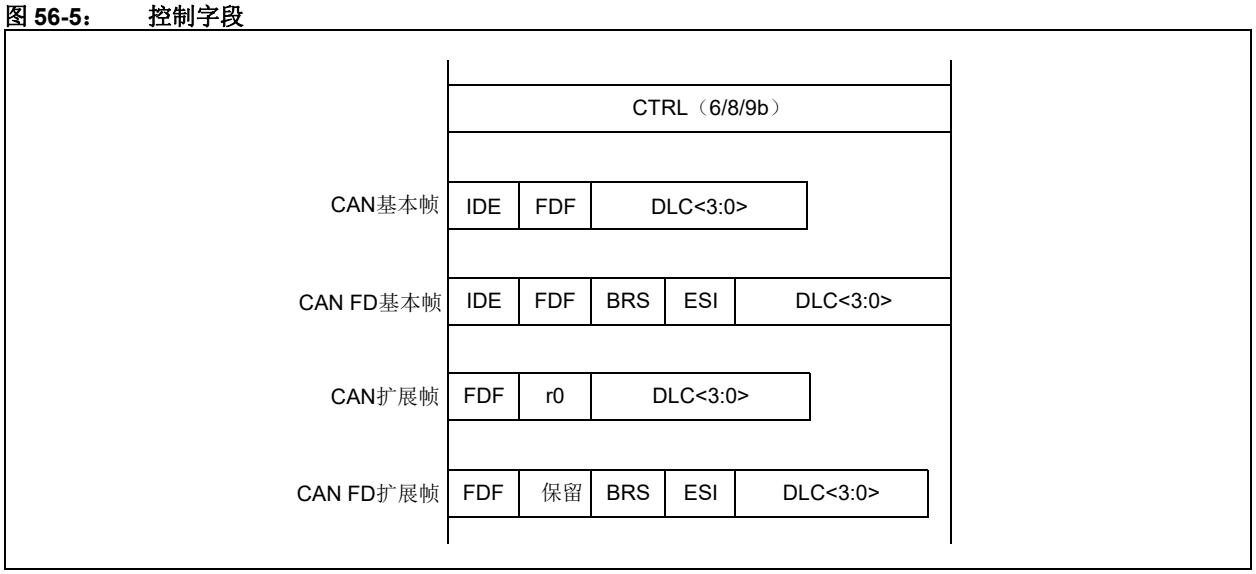
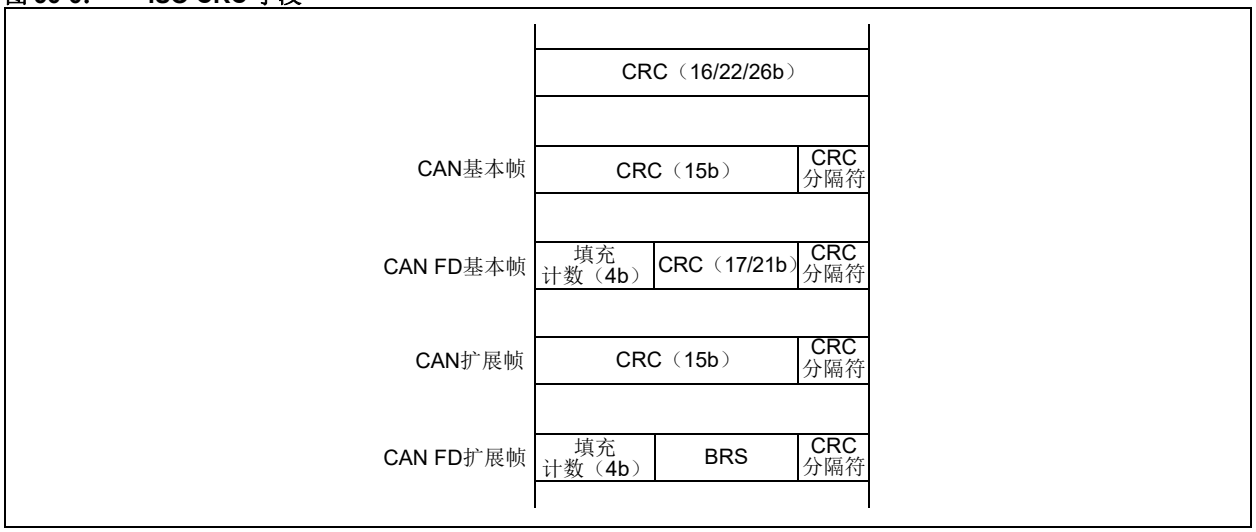


图 56-6： ISO CRC 字段



 © 2019 Microchip Technology Inc.

图 56-7： 非 ISO CRC 字段

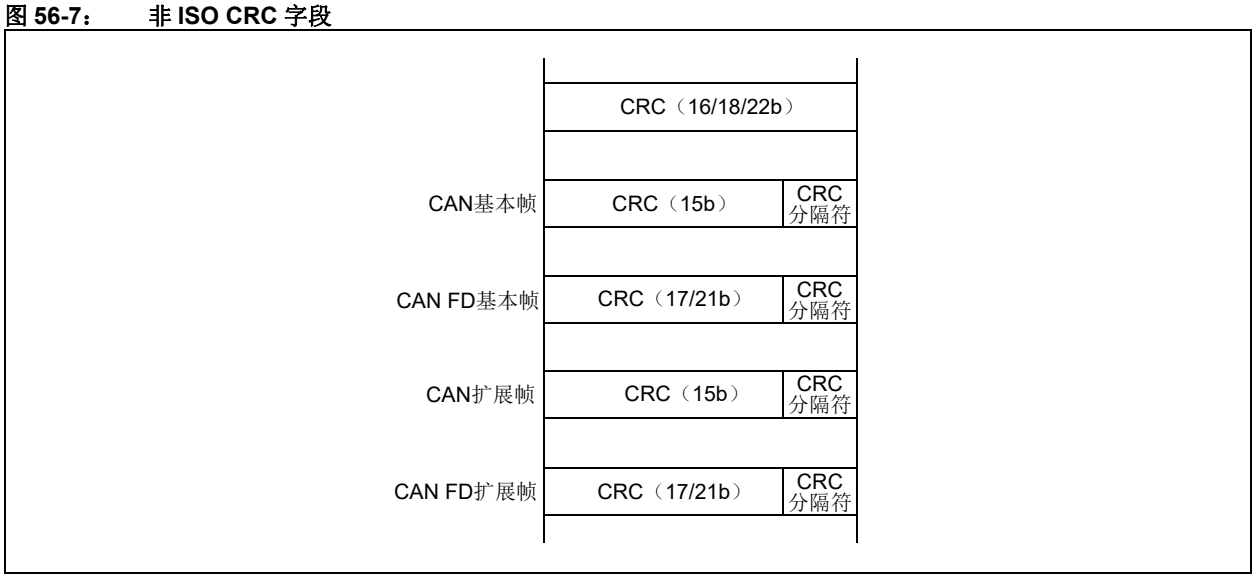
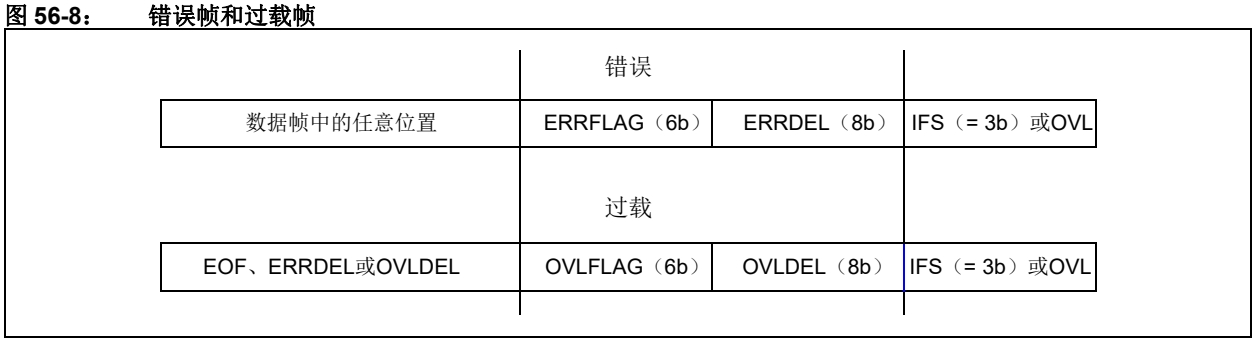


图 56-8： 错误帧和过载帧



56.2.1.1 DLC编码

数据长度码（Data Length Code，DLC）指定了报文帧中包含的数据字节数。表56-1提供了编码详细信息。

表 56-1： DLC 编码

帧	DLC	数据字节数
CAN 2.0和CAN FD	0	0
	1	1
	2	2
	3	3
	4	4
	5	5
	6	6
	7	7
	8	8
CAN 2.0	9-15	8

表 **56-1：** **DLC** 编码 （续）

帧	DLC	数据字节数
CAN FD	9	12
	10	16
	11	20
	12	24
	13	32
	14	48
	15	64

 © 2019 Microchip Technology Inc.

56.3 控制寄存器

PIC32 CAN FD模块具有以下特殊功能寄存器 （Special Function Register， SFR）：

- **CFDxCON：CAN控制寄存器 （x = 1-4）**

该寄存器控制相应CAN-FD模块的基本操作，包括多次连续发送之间的延时、工作模式、通知相应模块的所有发送缓冲区中止发送、定义重发尝试次数、空闲模式下的工作、配置唤醒滤波器、选择时钟源以及使能或禁止该模块。

- **CFDxNBTCFG：标称位时间配置寄存器 （x = 1-4）**

  该寄存器配置相应CAN-FD模块的标称位时序 （仲裁阶段）。
- **CFDxDBTCFG：数据位时间配置寄存器 （x = 1-4）**

  该寄存器配置相应CAN-FD模块的数据位时间 （数据阶段）。
- **CFDxTDC：发送器延时补偿寄存器 （x = 1-4）**

  该寄存器使能发送器延时补偿。
- **CFDxTBC：CAN时基计数器寄存器 （x = 1-4）**

  该寄存器保存时间戳定时器的32位计数。
- **CFDxTSCON：CAN时间戳控制寄存器 （x = 1-4）**

  该寄存器控制时间戳定时器的操作。
- **CFDxVEC：中断代码寄存器 （x = 1-4）**

  该寄存器标识CAN模块的待处理中断源。
- **CFDxINT：中断寄存器 （x = 1-4）**

  该寄存器可允许/禁止CAN-FD模块内部的中断。
- **CFDxRXIF：接收中断状态寄存器 （x = 1-4）**

  该寄存器标识待处理的接收FIFO中断。
- **CFDxRXOVIF：接收溢出中断状态寄存器 （x = 1-4）**

  该寄存器标识待处理的接收FIFO溢出中断。
- **CFDxTXIF：发送中断状态寄存器 （x = 1-4）**

  该寄存器标识待处理的发送FIFO中断。
- **CFDxTXATIF：发送尝试中断状态寄存器 （x = 1-4）**

  该寄存器标识待处理的发送FIFO尝试中断。
- **CFDxTXREQ：发送请求寄存器 （x = 1-4）**

  该寄存器允许每个发送FIFO排队等待发送。
- **CFDxFIFOBA：报文存储器基址寄存器 （x = 1-4）**

  该寄存器定义发送事件FIFO的基址。
- **CFDxTXQCON：发送队列控制寄存器 （x = 1-4）**

  该寄存器配置和控制发送队列。
- **CFDxTXQSTA：发送队列状态寄存器 （x = 1-4）**

  该寄存器反映发送队列状态。
- **CFDxFIFOCONn：FIFO控制寄存器 （x = 1-4；n = 1-31）**

  该寄存器配置和控制FIFO。
- **CFDxFIFOSTAn：FIFO状态寄存器 （x = 1-4；n = 1-31）**

  该寄存器反映FIFO状态。
- **CFDxTEFCON：发送事件FIFO控制寄存器 （x = 1-4）**

  该寄存器配置和控制发送事件FIFO。
- **CFDxTEFSTA：发送事件FIFO状态寄存器 （x = 1-4）**

  该寄存器反映发送事件FIFO状态。
- **CFDxFIFOUAn：定义寄存器 （x = 1-4；n = 1-31）**

  该寄存器允许将每个FIFO定义为发送或接收缓冲区。

- **CFDxTEFUA：发送事件FIFO用户地址寄存器 （x = 1-4）**

  该寄存器提供用于读取下一个事件的地址。

- **CFDxTXQUA：发送队列用户地址寄存器 （x = 1-4）**

  该寄存器提供发送队列中下一个报文的地址。

- **CFDxTREC：发送/接收错误计数寄存器 （x = 1-4）**

  该寄存器提供发送和接收错误计数。

- **CFDxBDIAG0：总线诊断寄存器0 （x = 1-4）**

  该寄存器分别跟踪标称比特率和数据比特率阶段的总线发送与接收错误。

- **CFDxBDIAG1：总线诊断寄存器1 （x = 1-4）**

该寄存器记录总线上的无错误报文数。每次读取该寄存器时，其计数都会清零。该寄存器还会标识自上次读取寄存器以来发生的错误类型。

- **CFDxFLTCON0：过滤器控制寄存器 （x = 1-4）**

  该寄存器使能或禁止报文过滤器0-3。过滤器匹配时，该寄存器还指向用于存储与报文过滤器0-3匹配的报文的FIFO。

- **CFDxFLTCON1：过滤器控制寄存器 （x = 1-4）**

  该寄存器使能或禁止报文过滤器4-7。过滤器匹配时，该寄存器还指向用于存储与报文过滤器4-7匹配的报文的FIFO。

- **CFDxFLTCON2：过滤器控制寄存器 （x = 1-4）**

  该寄存器使能或禁止报文过滤器8-11。过滤器匹配时，该寄存器还指向用于存储与报文过滤器8-11匹配的报文的FIFO。

- **CFDxFLTCON3：过滤器控制寄存器 （x = 1-4）**

  该寄存器使能或禁止报文过滤器12-15。过滤器匹配时，该寄存器还指向用于存储与报文过滤器12-15匹配的报文的FIFO。

- **CFDxFLTCON4：过滤器控制寄存器 （x = 1-4）**

  该寄存器使能或禁止报文过滤器16-19。过滤器匹配时，该寄存器还指向用于存储与报文过滤器16-19匹配的报文的FIFO。

- **CFDxFLTCON5：过滤器控制寄存器 （x = 1-4）**

  该寄存器使能或禁止报文过滤器20-23。过滤器匹配时，该寄存器还指向用于存储与报文过滤器20-23匹配的报文的FIFO。

- **CFDxFLTCON6：过滤器控制寄存器 （x = 1-4）**

该寄存器使能或禁止报文过滤器24-27。过滤器匹配时，该寄存器还指向用于存储与报文过滤器24-27匹配的报文的FIFO。

- **CFDxFLTCON7：过滤器控制寄存器 （x = 1-4）**

  该寄存器使能或禁止报文过滤器28-31。过滤器匹配时，该寄存器还指向用于存储与报文过滤器28-31匹配的报文的FIFO。

- **CFDxFLTOBJn：过滤器对象寄存器 （x = 1-4；n = 0-31）**

  该寄存器设置报文过滤器的过滤器对象。

- **CFDxMASKn：屏蔽寄存器 （x = 1-4；n = 0-31）**

  该寄存器设置报文过滤器的屏蔽寄存器对象。

 © 2019 Microchip Technology Inc.

表56-2： CAN FD外设寄存器汇总

寄存器名称	位范围	Bit 31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
CFD1CON	31:16	TXBWS<3:0>				ABAT	REQOP<2:0>			OPMOD<2:0>			TXQEN	STEF	SERR2LOM	ESIGM	RTXAT
	15:0	ON	—	SIDL	BRSDIS	BUSY	WFT<1:0>		WAKFIL	CLKSEL0	PXEDIS	ISOCRCEN	DNCNT<4:0>				
CFD1NBTCFG	31:16	BRP<7:0>								TSEG1<7:0>							
	15:0	—	TSEG2<6:0>							—	SJW<6:0>						
CFD1DBTCFG	31:16	BRP<7:0>								—	—	—	TSEG2<4:0>				
	15:0	—	—	—	—	TSEG2<3:0>				—	—	—	—	SJW<3:0>			
CFD1TDC	31:16	—	—	—	—	—	—	EDGFLTEN	SID11EN	—	—	—	—	—	—	TDCMOD<1:0>	
	15:0	—	TDCO<6:0>							—	—	TDCV<5:0>					
CFD1TBC	31:16	TBC<31:16>															
	15:0	TBC<15:0>															
CFD1TSCON	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	TSRES	TSEOF	TBCEN
	15:0	—	—	—	—	—	—	TBCPRE<9:0>									
CFD1VEC	31:16	—	RXCODE<6:0>							—	TXCODE<6:0>						
	15:0	—	—	—	FILHIT<4:0>					—	ICODE<6:0>						
CFD1INT	31:16	IVMIE	WAKIE	CERRIE	SERRIE	RXOVIE	TXATIE	—	—	—	—	—	TEFIE	MODIE	TBCIE	RXIE	TXIE
	15:0	IVMIF	WAKIF	CERRIF	SERRIF	RXOVIF	TXATIF	—	—	—	—	—	TEFIF	MODIF	TBCIF	RXIF	TXIF
CFD1RXIF	31:16	RFIF<31:16>															
	15:0	RFIF<15:1>															—
CFD1TXIF	31:16	TFIF<31:16>															
	15:0	TFIF<15:0>															
CFD1RXOVIF	31:16	RFOVIF<31:16>															
	15:0	RFOVIF<15:1>															—
CFD1TXATIF	31:16	TFATIF<31:16>															
	15:0	TFATIF<15:0>															
CFD1TXREQ	31:16	TXREQ<31:16>															
	15:0	TXREQ<15:0>															
CFD1TREC	31:16	—	—	—	—	—	—	—	—	—	—	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN
	15:0	TERRCNT<7:0>								RERRCNT<7:0>							
CFD1BDIAG0	31:16	DTERRCNT<7:0>								DRERRCNT<7:0>							
	15:0	NTERRCNT<7:0>								NRERRCNT<7:0>							
CFD1BDIAG1	31:16	DLCMM	ESI	DCRCERR	DSTUFERR	DFORMERR	—	DBIT1ERR	DBIT0ERR	—	—	NCRCERR	NSTUFERR	NFORMERR	NACKERR	NBIT1ERR	NBIT0ERR
	15:0	EFMSGCNT<15:0>															
CFD1TEFCON	31:16	—	—	—	FSIZE<4:0>					—	—	—	—	—	—	—	—
	15:0	—	—	—	—	—	FRESET	—	UINC	—	—	TEFTSEN	—	TEFOVIE	TEFFIE	TEFHIE	TEFNEIE
CFD1TEFSTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	—	—	—	—	—	—	—	—	—	TEFOVIF	TEFFIF	TEFHIF	TEFNEIF
CFD1TEFUA	31:16	TEFUA<31:16>															
	15:0	TEFUA<15:0>															
CFD1FIFOBA	31:16	FIFOBA<31:16>															
	15:0	FIFOBA<15:0>															
CFD1TXQCON	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	—	—	TXATIE	—	TXQEIE	—	TXQNIE

图注： x = 复位时的未知值；— = 未实现 （读为0）。复位值以十六进制显示。

注 1： 32位寄存器的低位字节位于低地址。

© 2019 Microchip Technology Inc.

表56-2： CAN FD外设寄存器汇总（续）

寄存器名称	位范围	Bit 31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
CFD1TXQSTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	TXQCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	—	TXQEIF	—	TXQNIF
CFD1TXQUA	31:16	TXQUA<31:16>															
	15:0	TXQUA<15:0>															
CFD1FIFOCONn（n = 1）	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	RTREN	RXTSEN	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE
CFD1FIFOSTAn（n = 1）	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	FIFOCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF
CFD1FIFOUAn（n = 1）	31:16	FIFOUA<31:16>															
	15:0	FIFOUA<15:0>															
CFD1FIFOCONn（n = 2-31）	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	RTREN	RXTSEN	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE
CFD1FIFOSTAn（n = 2-31）	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	FIFOCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF
CFD1FIFOUAn（n = 2-31）	31:16	FIFOUA<31:16>															
	15:0	FIFOUA<15:0>															
CFD1FLTCON0	31:16	FLTEN3	—	—	F3BP<4:0>					FLTEN2	—	—	F2BP<4:0>				
	15:0	FLTEN1	—	—	F1BP<4:0>					FLTEN0	—	—	F0BP<4:0>				
CFD1FLTCON1	31:16	FLTEN7	—	—	F7BP<4:0>					FLTEN6	—	—	F6BP<4:0>				
	15:0	FLTEN5	—	—	F5BP<4:0>					FLTEN4	—	—	F4BP<4:0>				
CFD1FLTCON2	31:16	FLTEN11	—	—	F11BP<4:0>					FLTEN10	—	—	F10BP<4:0>				
	15:0	FLTEN9	—	—	F9BP<4:0>					FLTEN8	—	—	F8BP<4:0>				
CFD1FLTCON3	31:16	FLTEN15	—	—	F15BP<4:0>					FLTEN14	—	—	F14BP<4:0>				
	15:0	FLTEN13	—	—	F13BP<4:0>					FLTEN12	—	—	F12BP<4:0>				
CFD1FLTCON4	31:16	FLTEN19	—	—	F19BP<4:0>					FLTEN18	—	—	F18BP<4:0>				
	15:0	FLTEN17	—	—	F17BP<4:0>					FLTEN16	—	—	F16BP<4:0>				
CFD1FLTCON5	31:16	FLTEN23	—	—	F23BP<4:0>					FLTEN22	—	—	F22BP<4:0>				
	15:0	FLTEN21	—	—	F21BP<4:0>					FLTEN20	—	—	F20BP<4:0>				
CFD1FLTCON6	31:16	FLTEN27	—	—	F28BP<4:0>					FLTEN26	—	—	F26BP<4:0>				
	15:0	FLTEN25	—	—	F25BP<4:0>					FLTEN24	—	—	F30BP<4:0>				
CFD1FLTCON7	31:16	FLTEN31	—	—	F31BP<4:0>					FLTEN30	—	—	F26BP<4:0>				
	15:0	FLTEN29	—	—	F29BP<4:0>					FLTEN28	—	—	F28BP<4:0>				
CFD1FLTOBJn（n = 0）	31:16	—	EXIDE	SID11	EID<17:5>												
	15:0	EID<4:0>					SID<10:0>										
CFD1MASKn（n = 0）	31:16	—	MIDE	MSID11	MEID<17:5>												
	15:0	MEID<4:0>					MSID<10:0>										
CFD1FLTOBJn（n = 1-31）	31:16	—	EXIDE	SID11	EID<17:5>												
	15:0	EID<4:0>					SID<10:0>										
CFD1MASKn（n = 1-31）	31:16	—	MIDE	MSID11	MEID<17:5>												
	15:0	MEID<4:0>					MSID<10:0>										
CFD2CON	31:16	TXBWS<3:0>				ABAT	REQOP<2:0>			OPMOD<2:0>			TXQEN	STEF	SERR2LOM	ESIGM	RTXAT
	15:0	ON	—	SIDL	BRSDIS	BUSY	WFT<1:0>		WAKFIL	CLKSEL0	PXEDIS	ISOCRCEN	DNCNT<4:0>				
CFD2NBTCFG	31:16	BRP<7:0>								TSEG1<7:0>							
	15:0	—	TSEG2<6:0>							—	SJW<6:0>						

图注： x = 复位时的未知值；— = 未实现 （读为0）。复位值以十六进制显示。

注 1： 32位寄存器的低位字节位于低地址。

初稿

© 2019 Microchip Technology Inc.

表56-2： CAN FD外设寄存器汇总（续）

寄存器名称	位范围	Bit 31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
CFD2DBTCFG	31:16	BRP<7:0>								—	—	—	TSEG2<4:0>				
	15:0	—	—	—	—	TSEG2<3:0>				—	—	—	—	SJW<3:0>			
CFD2TDC	31:16	—	—	—	—	—	—	EDGFLTEN	SID11EN	—	—	—	—	—	—	TDCMOD<1:0>	
	15:0	—	TDCO<6:0>							—	—	TDCV<5:0>					
CFD2TBC	31:16	TBC<31:16>															
	15:0	TBC<15:0>															
CFD2TSCON	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	TSRES	TSEOF	TBCEN
	15:0	—	—	—	—	—	—	TBCPRE<9:0>									
CFD2VEC	31:16	—	RXCODE<6:0>							—	TXCODE<6:0>						
	15:0	—	—	—	FILHIT<4:0>					—	ICODE<6:0>						
CFD2INT	31:16	IVMIE	WAKIE	CERRIE	SERRIE	RXOVIE	TXATIE	—	—	—	—	—	TEFIE	MODIE	TBCIE	RXIE	TXIE
	15:0	IVMIF	WAKIF	CERRIF	SERRIF	RXOVIF	TXATIF	—	—	—	—	—	TEFIF	MODIF	TBCIF	RXIF	TXIF
CFD2RXIF	31:16	RFIF<31:16>															
	15:0	RFIF<15:1>															—
CFD2TXIF	31:16	TFIF<31:16>															
	15:0	TFIF<15:0>															
CFD2RXOVIF	31:16	RFOVIF<31:16>															
	15:0	RFOVIF<15:1>															—
CFD2TXATIF	31:16	TFATIF<31:16>															
	15:0	TFATIF<15:0>															
CFD2TXREQ	31:16	TXREQ<31:16>															
	15:0	TXREQ<15:0>															
CFD2TREC	31:16	—	—	—	—	—	—	—	—	—	—	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN
	15:0	TERRCNT<7:0>								RERRCNT<7:0>							
CFD2BDIAG0	31:16	DTERRCNT<7:0>								DRERRCNT<7:0>							
	15:0	NTERRCNT<7:0>								NRERRCNT<7:0>							
CFD2BDIAG1	31:16	DLCMM	ESI	DCRCERR	DSTUFERR	DFORMERR	—	DBIT1ERR	DBIT0ERR	—	—	NCRCERR	NSTUFERR	NFORMERR	NACKERR	NBIT1ERR	NBIT0ERR
	15:0	EFMSGCNT<15:0>															
CFD2TEFCON	31:16	—	—	—	FSIZE<4:0>					—	—	—	—	—	—	—	—
	15:0	—	—	—	—	—	FRESET	—	UINC	—	—	TEFTSEN	—	TEFOVIE	TEFFIE	TEFHIE	TEFNEIE
CFD2TEFSTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	—	—	—	—	—	—	—	—	—	TEFOVIF	TEFFIF	TEFHIF	TEFNEIF
CFD2TEFUA	31:16	TEFUA<31:16>															
	15:0	TEFUA<15:0>															
CFD2FIFOBA	31:16	FIFOBA<31:16>															
	15:0	FIFOBA<15:0>															
CFD2TXQCON	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	—	—	TXATIE	—	TXQEIE	—	TXQNIE
CFD2TXQSTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	TXQCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	—	TXQEIF	—	TXQNIF
CFD2TXQUA	31:16	TXQUA<31:16>															
	15:0	TXQUA<15:0>															
CFD2FIFOCONn（n = 1）	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	RTREN	RXTSEN	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE

图注： x = 复位时的未知值；— = 未实现 （读为0）。复位值以十六进制显示。

注 1： 32位寄存器的低位字节位于低地址。

© 2019 Microchip Technology Inc.

初稿

表56-2： CAN FD外设寄存器汇总（续）

寄存器名称	位范围	Bit 31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
CFD2FIFOSTAn（n = 2-31）	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	FIFOCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF
CFD2FIFOUAn（n = 1）	31:16	FIFOUA<31:16>															
	15:0	FIFOUA<15:0>															
CFD2FIFOCONn（n = 2-31）	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	RTREN	RXTSEN	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE
CFD2FIFOSTAn（n = 1）	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	FIFOCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF
CFD2FIFOUAn（n = 2-31）	31:16	FIFOUA<31:16>															
	15:0	FIFOUA<15:0>															
CFD2FLTCON0	31:16	FLTEN3	—	—	F3BP<4:0>					FLTEN2	—	—	F2BP<4:0>				
	15:0	FLTEN1	—	—	F1BP<4:0>					FLTEN0	—	—	F0BP<4:0>				
CFD2FLTCON1	31:16	FLTEN7	—	—	F7BP<4:0>					FLTEN6	—	—	F6BP<4:0>				
	15:0	FLTEN5	—	—	F5BP<4:0>					FLTEN4	—	—	F4BP<4:0>				
CFD2FLTCON2	31:16	FLTEN11	—	—	F11BP<4:0>					FLTEN10	—	—	F10BP<4:0>				
	15:0	FLTEN9	—	—	F9BP<4:0>					FLTEN8	—	—	F8BP<4:0>				
CFD2FLTCON3	31:16	FLTEN15	—	—	F15BP<4:0>					FLTEN14	—	—	F14BP<4:0>				
	15:0	FLTEN13	—	—	F13BP<4:0>					FLTEN12	—	—	F12BP<4:0>				
CFD2FLTCON4	31:16	FLTEN19	—	—	F19BP<4:0>					FLTEN18	—	—	F18BP<4:0>				
	15:0	FLTEN17	—	—	F17BP<4:0>					FLTEN16	—	—	F16BP<4:0>				
CFD2FLTCON5	31:16	FLTEN23	—	—	F23BP<4:0>					FLTEN22	—	—	F22BP<4:0>				
	15:0	FLTEN21	—	—	F21BP<4:0>					FLTEN20	—	—	F20BP<4:0>				
CFD2FLTCON6	31:16	FLTEN27	—	—	F28BP<4:0>					FLTEN26	—	—	F26BP<4:0>				
	15:0	FLTEN25	—	—	F25BP<4:0>					FLTEN24	—	—	F30BP<4:0>				
CFD2FLTCON7	31:16	FLTEN31	—	—	F31BP<4:0>					FLTEN30	—	—	F26BP<4:0>				
	15:0	FLTEN29	—	—	F29BP<4:0>					FLTEN28	—	—	F28BP<4:0>				
CFD2FLTOBJn（n = 0）	31:16	—	EXIDE	SID11	EID<17:5>												
	15:0	EID<4:0>			SID<10:0>												
CFD2MASKn（n = 0）	31:16	—	MIDE	MSID11	MEID<17:5>												
	15:0	MEID<4:0>			MSID<10:0>												
CFD2FLTOBJn（n = 1-31）	31:16	—	EXIDE	SID11	EID<17:5>												
	15:0	EID<4:0>			SID<10:0>												
CFD2MASKn（n = 1-31）	31:16	—	MIDE	MSID11	MEID<17:5>												
	15:0	MEID<4:0>			MSID<10:0>												
CFD3CON	31:16	TXBWS<3:0>				ABAT	REQOP<2:0>			OPMOD<2:0>			TXQEN	STEF	SERR2LOM	ESIGM	RTXAT
	15:0	ON	—	SIDL	BRSDIS	BUSY	WFT<1:0>		WAKFIL	CLKSEL0	PXEDIS	ISOCRCEN	DNCNT<4:0>				
CFD3NBTCFG	31:16	BRP<7:0>								TSEG1<7:0>							
	15:0	—	TSEG2<6:0>							—	SJW<6:0>						
CFD3DBTCFG	31:16	BRP<7:0>								—	—	—	TSEG2<4:0>				
	15:0	—	—	—	—	TSEG2<3:0>				—	—	—	—	SJW<3:0>			
CFD3TDC	31:16	—	—	—	—	—	—	EDGFLTEN	SID11EN	—	—	—	—	—	—	TDCMOD<1:0>	
	15:0	—	TDCO<6:0>							—	—	TDCV<5:0>					
CFD3TBC	31:16	TBC<31:16>															
	15:0	TBC<15:0>															

图注： x = 复位时的未知值；— = 未实现 （读为0）。复位值以十六进制显示。

注 1： 32位寄存器的低位字节位于低地址。

初稿

© 2019 Microchip Technology Inc.

表56-2： CAN FD外设寄存器汇总（续）

寄存器名称	位范围	Bit 31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
CFD3TSCON	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	TSRES	TSEOF	TBCEN
	15:0	—	—	—	—	—	—	TBCPRE<9:0>									
CFD3VEC	31:16	—	RXCODE<6:0>							—	TXCODE<6:0>						
	15:0	—	—	—	FILHIT<4:0>					—	ICODE<6:0>						
CFD3INT	31:16	IVMIE	WAKIE	CERRIE	SERRIE	RXOVIE	TXATIE	—	—	—	—	—	TEFIE	MODIE	TBCIE	RXIE	TXIE
	15:0	IVMIF	WAKIF	CERRIF	SERRIF	RXOVIF	TXATIF	—	—	—	—	—	TEFIF	MODIF	TBCIF	RXIF	TXIF
CFD3RXIF	31:16	RFIF<31:16>															
	15:0	RFIF<15:1>															—
CFD3TXIF	31:16	TFIF<31:16>															
	15:0	TFIF<15:0>															
CFD3RXOVIF	31:16	RFOVIF<31:16>															
	15:0	RFOVIF<15:1>															—
CFD3TXATIF	31:16	TFATIF<31:16>															
	15:0	TFATIF<15:0>															
CFD3TXREQ	31:16	TXREQ<31:16>															
	15:0	TXREQ<15:0>															
CFD3TREC	31:16	—	—	—	—	—	—	—	—	—	—	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN
	15:0	TERRCNT<7:0>								RERRCNT<7:0>							
CFD3BDIAG0	31:16	DTERRCNT<7:0>								DRERRCNT<7:0>							
	15:0	NTERRCNT<7:0>								NRERRCNT<7:0>							
CFD3BDIAG1	31:16	DLCMM	ESI	DCRCERR	DSTUFERR	DFORMERR	—	DBIT1ERR	DBIT0ERR	—	—	NCRCERR	NSTUFERR	NFORMERR	NACKERR	NBIT1ERR	NBIT0ERR
	15:0	EFMSGCNT<15:0>															
CFD3TEFCON	31:16	—	—	—	FSIZE<4:0>					—	—	—	—	—	—	—	—
	15:0	—	—	—	—	—	FRESET	—	UINC	—	—	TEFTSEN	—	TEFOVIE	TEFFIE	TEFHIE	TEFNEIE
CFD3TEFSTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	—	—	—	—	—	—	—	—	—	TEFOVIF	TEFFIF	TEFHIF	TEFNEIF
CFD3TEFUA	31:16	TEFUA<31:16>															
	15:0	TEFUA<15:0>															
CFD3FIFOBA	31:16	FIFOBA<31:16>															
	15:0	FIFOBA<15:0>															
CFD3TXQCON	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	—	—	TXATIE	—	TXQEIE	—	TXQNIE
CFD3TXQSTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	TXQCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	—	TXQEIF	—	TXQNIF
CFD3TXQUA	31:16	TXQUA<31:16>															
	15:0	TXQUA<15:0>															
CFD3FIFOCONn（n = 1）	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	RTREN	RXTSEN	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE
CFD3FIFOSTAn（n = 1）	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	FIFOCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF
CFD3FIFOUAn（n = 1）	31:16	FIFOUA<31:16>															
	15:0	FIFOUA<15:0>															
CFD3FIFOCONn（n = 2-31）	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	RTREN	RXTSEN	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE

图注： x = 复位时的未知值；— = 未实现 （读为0）。复位值以十六进制显示。

注 1： 32位寄存器的低位字节位于低地址。

© 2019 Microchip Technology Inc.

初稿

表56-2： CAN FD外设寄存器汇总（续）

寄存器名称	位范围	Bit 31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
CFD3FIFOSTAn（n = 2-31）	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	FIFOCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF
CFD3FIFOUAn（n = 2-31）	31:16	FIFOUA<31:16>															
	15:0	FIFOUA<15:0>															
CFD3FLTCON0	31:16	FLTEN3	—	—	F3BP<4:0>					FLTEN2	—	—	F2BP<4:0>				
	15:0	FLTEN1	—	—	F1BP<4:0>					FLTEN0	—	—	F0BP<4:0>				
CFD3FLTCON1	31:16	FLTEN7	—	—	F7BP<4:0>					FLTEN6	—	—	F6BP<4:0>				
	15:0	FLTEN5	—	—	F5BP<4:0>					FLTEN4	—	—	F4BP<4:0>				
CFD3FLTCON2	31:16	FLTEN11	—	—	F11BP<4:0>					FLTEN10	—	—	F10BP<4:0>				
	15:0	FLTEN9	—	—	F9BP<4:0>					FLTEN8	—	—	F8BP<4:0>				
CFD3FLTCON3	31:16	FLTEN15	—	—	F15BP<4:0>					FLTEN14	—	—	F14BP<4:0>				
	15:0	FLTEN13	—	—	F13BP<4:0>					FLTEN12	—	—	F12BP<4:0>				
CFD3FLTCON4	31:16	FLTEN19	—	—	F19BP<4:0>					FLTEN18	—	—	F18BP<4:0>				
	15:0	FLTEN17	—	—	F17BP<4:0>					FLTEN16	—	—	F16BP<4:0>				
CFD3FLTCON5	31:16	FLTEN23	—	—	F23BP<4:0>					FLTEN22	—	—	F22BP<4:0>				
	15:0	FLTEN21	—	—	F21BP<4:0>					FLTEN20	—	—	F20BP<4:0>				
CFD3FLTCON6	31:16	FLTEN27	—	—	F28BP<4:0>					FLTEN26	—	—	F26BP<4:0>				
	15:0	FLTEN25	—	—	F25BP<4:0>					FLTEN24	—	—	F30BP<4:0>				
CFD3FLTCON7	31:16	FLTEN31	—	—	F31BP<4:0>					FLTEN30	—	—	F26BP<4:0>				
	15:0	FLTEN29	—	—	F29BP<4:0>					FLTEN28	—	—	F28BP<4:0>				
CFD3FLTOBJn（n = 0）	31:16	—	EXIDE	SID11	EID<17:5>												
	15:0	EID<4:0>					SID<10:0>										
CFD3MASKn（n = 0）	31:16	—	MIDE	MSID11	MEID<17:5>												
	15:0	MEID<4:0>					MSID<10:0>										
CFD3FLTOBJn（n = 1-31）	31:16	—	EXIDE	SID11	EID<17:5>												
	15:0	EID<4:0>					SID<10:0>										
CFD3MASKn（n = 1-31）	31:16	—	MIDE	MSID11	MEID<17:5>												
	15:0	MEID<4:0>					MSID<10:0>										
CFD4CON	31:16	TXBWS<3:0>				ABAT	REQOP<2:0>			OPMOD<2:0>			TXQEN	STEF	SERR2LOM	ESIGM	RTXAT
	15:0	ON	—	SIDL	BRSDIS	BUSY	WFT<1:0>		WAKFIL	CLKSEL0	PXEDIS	ISOCRCEN	DNCNT<4:0>				
CFD4NBTCFG	31:16	BRP<7:0>								TSEG1<7:0>							
	15:0	—	TSEG2<6:0>							—	SJW<6:0>						
CFD4DBTCFG	31:16	BRP<7:0>								—	—	—	TSEG2<4:0>				
	15:0	—	—	—	—	TSEG2<3:0>				—	—	—	—	SJW<3:0>			
CFD4TDC	31:16	—	—	—	—	—	—	EDGFLTEN	SID11EN	—	—	—	—	—	—	TDCMOD<1:0>	
	15:0	—	TDCO<6:0>							—	—	TDCV<5:0>					
CFD4TBC	31:16	TBC<31:16>															
	15:0	TBC<15:0>															
CFD4TSCON	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	TSRES	TSEOF	TBCEN
	15:0	—	—	—	—	—	—	TBCPRE<9:0>									
CFD4VEC	31:16	—	RXCODE<6:0>							—	TXCODE<6:0>						
	15:0	—	—	—	FILHIT<4:0>					—	ICODE<6:0>						
CFD4INT	31:16	IVMIE	WAKIE	CERRIE	SERRIE	RXOVIE	TXATIE	—	—	—	—	—	TEFIE	MODIE	TBCIE	RXIE	TXIE
	15:0	IVMIF	WAKIF	CERRIF	SERRIF	RXOVIF	TXATIF	—	—	—	—	—	TEFIF	MODIF	TBCIF	RXIF	TXIF

图注： x = 复位时的未知值；— = 未实现（读为0）。复位值以十六进制显示。

注 1： 32位寄存器的低位字节位于低地址。

初稿

© 2019 Microchip Technology Inc.

表56-2： CAN FD外设寄存器汇总（续）

寄存器名称	位范围	Bit 31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
CFD4RXIF	31:16	RFIF<31:16>															
	15:0	RFIF<15:1>															—
CFD4TXIF	31:16	TFIF<31:16>															
	15:0	TFIF<15:0>															
CFD4RXOVIF	31:16	RFOVIF<31:16>															
	15:0	RFOVIF<15:1>															—
CFD4TXATIF	31:16	TFATIF<31:16>															
	15:0	TFATIF<15:0>															
CFD4TXREQ	31:16	TXREQ<31:16>															
	15:0	TXREQ<15:0>															
CFD4TREC	31:16	—	—	—	—	—	—	—	—	—	—	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN
	15:0	TERRCNT<7:0>								RERRCNT<7:0>							
CFD4BDIAG0	31:16	DTERRCNT<7:0>								DRERRCNT<7:0>							
	15:0	NTERRCNT<7:0>								NRERRCNT<7:0>							
CFD4BDIAG1	31:16	DLCMM	ESI	DCRCERR	DSTUFERR	DFORMERR	—	DBIT1ERR	DBIT0ERR	—	—	NCRCERR	NSTUFERR	NFORMERR	NACKERR	NBIT1ERR	NBIT0ERR
	15:0	EFMSGCNT<15:0>															
CFD4TEFCON	31:16	—	—	—	FSIZE<4:0>					—	—	—	—	—	—	—	—
	15:0	—	—	—	—	—	FRESET	—	UINC	—	—	TEFTSEN	—	TEFOVIE	TEFFIE	TEFHIE	TEFNEIE
CFD4TEFSTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	—	—	—	—	—	—	—	—	—	TEFOVIF	TEFFIF	TEFHIF	TEFNEIF
CFD4TEFUA	31:16	TEFUA<31:16>															
	15:0	TEFUA<15:0>															
CFD4FIFOBA	31:16	FIFOBA<31:16>															
	15:0	FIFOBA<15:0>															
CFD4TXQCON	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	—	—	TXATIE	—	TXQEIE	—	TXQNIE
CFD4TXQSTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	TXQCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	—	TXQEIF	—	TXQNIF
CFD4TXQUA	31:16	TXQUA<31:16>															
	15:0	TXQUA<15:0>															
CFD4FIFOCONn（n = 1）	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	RTREN	RXTSEN	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE
CFD4FIFOSTAn（n = 1）	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	FIFOCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF
CFD4FIFOUAn（n = 1）	31:16	FIFOUA<31:16>															
	15:0	FIFOUA<15:0>															
CFD4FIFOCONn（n = 2-31）	31:16	PLSIZE<2:0>			FSIZE<4:0>					—	TXAT<1:0>		TXPRI<4:0>				
	15:0	—	—	—	—	—	FRESET	TXREQ	UINC	TXEN	RTREN	RXTSEN	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE
CFD4FIFOSTAn（n = 2-31）	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	15:0	—	—	—	FIFOCI<4:0>					TXABT	TXLARB	TXERR	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF
CFD4FIFOUAn（n = 2-31）	31:16	FIFOUA<31:16>															
	15:0	FIFOUA<15:0>															
CFD4FLTCON0	31:16	FLTEN3	—	—	F3BP<4:0>					FLTEN2	—	—	F2BP<4:0>				
	15:0	FLTEN1	—	—	F1BP<4:0>					FLTEN0	—	—	F0BP<4:0>				

图注： x = 复位时的未知值；— = 未实现 （读为0）。复位值以十六进制显示。

注 1： 32位寄存器的低位字节位于低地址。

© 2019 Microchip Technology Inc.

表56-2： CAN FD外设寄存器汇总（续）

寄存器名称	位范围	Bit 31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
CFD4FLTCON1	31:16	FLTEN7	—	—	F7BP<4:0>					FLTEN6	—	—	F6BP<4:0>				
	15:0	FLTEN5	—	—	F5BP<4:0>					FLTEN4	—	—	F4BP<4:0>				
CFD4FLTCON2	31:16	FLTEN11	—	—	F11BP<4:0>					FLTEN10	—	—	F10BP<4:0>				
	15:0	FLTEN9	—	—	F9BP<4:0>					FLTEN8	—	—	F8BP<4:0>				
CFD4FLTCON3	31:16	FLTEN15	—	—	F15BP<4:0>					FLTEN14	—	—	F14BP<4:0>				
	15:0	FLTEN13	—	—	F13BP<4:0>					FLTEN12	—	—	F12BP<4:0>				
CFD4FLTCON4	31:16	FLTEN19	—	—	F19BP<4:0>					FLTEN18	—	—	F18BP<4:0>				
	15:0	FLTEN17	—	—	F17BP<4:0>					FLTEN16	—	—	F16BP<4:0>				
CFD4FLTCON5	31:16	FLTEN23	—	—	F23BP<4:0>					FLTEN22	—	—	F22BP<4:0>				
	15:0	FLTEN21	—	—	F21BP<4:0>					FLTEN20	—	—	F20BP<4:0>				
CFD4FLTCON6	31:16	FLTEN27	—	—	F28BP<4:0>					FLTEN26	—	—	F26BP<4:0>				
	15:0	FLTEN25	—	—	F25BP<4:0>					FLTEN24	—	—	F30BP<4:0>				
CFD4FLTCON7	31:16	FLTEN31	—	—	F31BP<4:0>					FLTEN30	—	—	F26BP<4:0>				
	15:0	FLTEN29	—	—	F29BP<4:0>					FLTEN28	—	—	F28BP<4:0>				
CFD4FLTOBJn（n = 0）	31:16	—	EXIDE	SID11	EID<17:5>												
	15:0	EID<4:0>					SID<10:0>										
CFD4MASKn（n = 0）	31:16	—	MIDE	MSID11	MEID<17:5>												
	15:0	MEID<4:0>					MSID<10:0>										
CFD4FLTOBJn（n = 1-31）	31:16	—	EXIDE	SID11	EID<17:5>												
	15:0	EID<4:0>					SID<10:0>										
CFD4MASKn（n = 1-31）	31:16	—	MIDE	MSID11	MEID<17:5>												
	15:0	MEID<4:0>					MSID<10:0>										

图注： x = 复位时的未知值；— = 未实现 （读为0）。复位值以十六进制显示。

注 1： 32位寄存器的低位字节位于低地址。

初稿

© 2019 Microchip Technology Inc.

寄存器56-1：CFDxCON：CAN控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	S和HC-0	R/W-1	R/W-0	R/W-0
	TXBWS<3:0>				ABAT	REQOP<2:0>		
23:16	R-1	R-0	R-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0
	OPMOD<2:0>			TXQEN(1)	STEF(1)	SERR2LOM(1)	ESIGM(1)	RTXAT(1)
15:8	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-1	R/W-1	R/W-1
	ON	—	SIDL(2)	BRSDIS	BUSY	WFT<1:0>		WAKFIL(1)
7:0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	CLKSEL0(1)	PXEDIS(1)	ISOCRCEN(1)	DNCNT<4:0>				

图注：	S = 可置1位	HC = 硬件清零位	
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-28 **TXBWS<3:0>**：发送带宽共享位
两次连续发送之间的延时 （仲裁位时间）
1111-1100 = 4096
1011 = 2048
1010 = 1024
1001 = 512
1000 = 256
0111 = 128
0110 = 64
0101 = 32
0100 = 16
0011 = 8
0010 = 4
0001 = 2
0000 = 无延时

bit 27 **ABAT**：中止所有待处理的发送位
1 = 通知所有发送缓冲区中止发送
0 = 模块将在所有发送中止时清零该位

bit 26-24 **REQOP<2:0>**：请求工作模式位
111 = 置为受限工作模式
110 = 置为正常CAN 2.0模式；接收CAN FD帧时生成错误帧
101 = 置为外部环回模式
100 = 置为配置模式
011 = 置为仅监听模式
010 = 置为内部环回模式
001 = 置为禁止模式
000 = 置为正常CAN FD模式；支持混合使用全CAN FD帧和经典CAN 2.0帧

注 1： 只能在配置模式 （OPMOD<2:0>位 = 100）下修改此位。

寄存器56-1：CFDxCON：CAN控制寄存器（ x = 1-4）（续）

bit 23-21 **OPMOD<2:0>**：工作模式状态位

111 = 模块处于受限工作模式
110 = 模块处于正常CAN 2.0模式；接收CAN FD帧时生成错误帧
101 = 模块处于外部环回模式
100 = 模块处于配置模式
011 = 模块处于仅监听模式
010 = 模块处于内部环回模式
001 = 模块处于禁止模式
000 = 模块处于正常CAN FD模式；支持混合使用全CAN FD帧和经典CAN 2.0帧

注： 在受限工作模式下，节点能够接收数据帧和远程帧，也能够应答有效帧，但不发送数据帧、远程帧、有效错误帧或过载帧。

bit 20 **TXQEN**：使能发送队列位(1)

1 = 使能发送队列并在RAM中保留空间
0 = 不在RAM中为发送队列保留空间

注： 只能在配置模式下更改此位，因为它会更改RAM中的地址。

bit 19 **STEF**：发送事件FIFO存储位(1)

1 = 将发送的报文保存在TEF中
0 = 不将发送的报文保存到TEF中

注： 只能在配置模式下更改此位，因为它会更改RAM中的地址。

bit 18 **SERR2LOM**：发生系统错误时切换到仅监听模式位(1)

1 = 切换到仅监听模式
0 = 切换到受限工作模式

bit 17 **ESIGM**：在网关模式下发送ESI位(1)

1 = 当报文的ESI为高电平或CAN控制器处于被动错误状态时， ESI以隐性状态发送
0 = ESI反映CAN控制器的错误状态

bit 16 **RTXAT**：限制重发尝试位(1)

1 = 重发尝试受限，使用TXAT<1:0>位 （CFDxFIFOCONn<22:21>）
0 = 将忽略TXAT<1:0>位，重发尝试次数不受限

bit 15 **ON**：使能位

1 = 使能CAN模块
0 = 禁止CAN模块

bit 14 **未实现**：读为0

bit 13 **SIDL**：空闲模式停止控制位(2)

1 = 模块在空闲模式下停止工作
0 = 模块在空闲模式下不停止工作

bit 12 **BRSDIS**：比特率切换禁止位

1 = 无论发送报文对象中的BRS状态如何，都禁止比特率切换
0 = 根据发送报文对象中的BRS进行比特率切换

bit 11 **BUSY**：CAN模块忙状态位

1 = CAN模块处于工作状态
0 = CAN模块处于非工作状态

bit 10-9 **WFT<1:0>**：可选唤醒滤波器时间位

11 = $T11_{FILTER}$ （典型值为600 ns）
10 = $T10_{FILTER}$ （典型值为375 ns）
01 = $T01_{FILTER}$ （典型值为187 ns）
00 = $T00_{FILTER}$ （典型值为100 ns）

注 1： 只能在配置模式 （OPMOD<2:0>位 = 100）下修改此位。

 © 2019 Microchip Technology Inc.

寄存器56-1：CFDxCON：CAN控制寄存器（ x = 1-4）（续）

bit 8 **WAKFIL**：使能CAN总线线路唤醒滤波器位(1)

1 = 使用CAN总线线路滤波器来唤醒
0 = 不使用CAN总线线路滤波器来唤醒

bit 7 **CLKSEL0**：模块时钟源选择位(1)

1 = 使能CAN FD模块时REFCLK4处于工作状态
0 = 使能CAN FD模块时SYSCLK处于工作状态

bit 6 **PXEDIS**：协议异常事件检测禁止位(1)

隐性FDF位后的隐性 “保留位”称为 “协议异常”。

1 = 将协议异常视为格式错误。
0 = 如果检测到协议异常， CAN FD模块将进入总线集成状态。

bit 5 **ISOCRCEN**：使能CAN FD帧中的ISO CRC位(1)

1 = CRC字段中包含填充位计数，使用非零CRC初始化向量
0 = CRC字段中不包含填充位计数，使用全零CRC初始化向量

bit 4-0 **DNCNT<4:0>**：DeviceNet™过滤器位编号位

11111-10011 = 无效选择 （与EID<0:17>进行比较）
10010 = 最多可将数据字节0<7:0>、 1<7:0>和2<6>与EID17进行比较
10001 = 最多可将数据字节0<7:0>、 1<7:0>和2<7>与EID<0:16>进行比较
10000 = 最多可将数据字节0<7:0>和数据字节1<7:0>与EID<0:15>进行比较
01111 = 最多可将数据字节0<7:0>和数据字节1<7:1>与EID<0:14>进行比较
01110 = 最多可将数据字节0<7:0>和数据字节1<7:2>与EID<0:13>进行比较
01101 = 最多可将数据字节0<7:0>和数据字节1<7:3>与EID<0:12>进行比较
01100 = 最多可将数据字节0<7:0>和数据字节1<7:4>与EID<0:11>进行比较
01011 = 最多可将数据字节0<7:0>和数据字节1<7:5>与EID<0:10>进行比较
01010 = 最多可将数据字节0<7:0>和数据字节1<7:6>与EID<0:9>进行比较
01001 = 最多可将数据字节0<7:0>和数据字节1<7>与EID<0:8>进行比较
01000 = 最多可将数据字节0<7:0>与EID<0:7>进行比较
00111 = 最多可将数据字节0<7:1>与EID<0:6>进行比较
00110 = 最多可将数据字节0<7:2>与EID<0:5>进行比较
00101 = 最多可将数据字节0<7:3>与EID<0:4>进行比较
00100 = 最多可将数据字节0<7:4>与EID<0:3>进行比较
00011 = 最多可将数据字节0<7:5>与EID<0:2>进行比较
00010 = 最多可将数据字节0<7:6>与EID<0:1>进行比较
00001 = 最多可将数据字节0的bit 7与EID0进行比较
00000 = 不比较数据字节

注 1： 只能在配置模式 （OPMOD<2:0>位 = 100）下修改此位。

寄存器56-2： CFDxNBTCFG：标称位时间配置寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	BRP<7:0>							
23:16	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-0
	TSEG1<7:0>							
15:8	U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
	—	—	—	TSEG2<4:0>				
7:0	U-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1
	—	SJW<6:0>						

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-24 **BRP<7:0>**：波特率预分频比位
11111111 = TQ = 256/FSYS
•
•
•
00000000 = TQ = 1/FSYS

bit 23-16 **TSEG1<7:0>**：时间段1位 （传播段 + 相位段1）
1111 1111 = 长度为256个TQ
•
•
•
00000000 = 长度为1个TQ

bit 15-13 **未实现**：读为0

bit 12-8 **TSEG2<4:0>**：时间段2位 （相位段2）
111111 = 长度为128个TQ
•
•
•
000000 = 长度为1个TQ

bit 7 **未实现**：读为0

bit 6-0 **SJW<6:0>**：同步跳转宽度位
1111111 = 长度为128个TQ
•
•
•
0000000 = 长度为1个TQ

注 1：只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改此寄存器。
2：以下内容适用于此寄存器：
- TQ = ((BRP + 1))/FCAN)
- 标称位周期 = (TQ * ((SYNC + (TSEG1 + 1) + (TSEG2 +1)))
- 标称比特率计算方式为(1/位周期)
  或
- CAN比特率计算方式为(1/(((BRP + 1))/FCAN) * (SYNC + (TSEG1 + 1) + (TSEG2 + 1))))

3：最大允许CAN FD误差≤ 1%；误差百分比 = (((计算得到的CAN比特率 - 所需的CAN比特率)/所需的CAN比特率) * 100)。

 © 2019 Microchip Technology Inc.

寄存器56-3： CFDxDBTCFG：数据位时间配置寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	BRP<7:0>							
23:16	U-0	U-0	U-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-0
	—	—	—	TSEG1<4:0>				
15:8	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-1	R/W-1
	—	—	—	—	TSEG2<3:0>			
7:0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-1	R/W-1
	—	—	—	—	SJW<3:0>			

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-24 **BRP<7:0>**：波特率预分频比位
11111111 = TQ = 256/FSYS
.
.
.
0000 0000 = TQ = 1/FSYS

bit 23-21 未实现：读为0

bit 20-16 **TSEG1<4:0>**：时间段1位 （传播段 + 相位段1）
11111 = 长度为32个TQ
.
.
.
00000 = 长度为1个TQ

bit 15-12 未实现：读为0

bit 11-8 **TSEG2<3:0>**：时间段2位 （相位段2）
1111 = 长度为16个TQ
.
.
.
0000 = 长度为1个TQ

bit 7-4 未实现：读为0

注 1： 只能在配置模式 （OPMOD<2:0>位 （**CFDx**CON<23:21>） = 100）下修改此寄存器。
2： 以下内容适用于此寄存器：
- TQ = ((BRP + 1))/FCAN)
- 标称位周期 = (TQ * ((SYNC + (TSEG1 + 1) + (TSEG2 +1)))
- 标称比特率计算方式为(1/位周期)
  或
- CAN比特率计算方式为(1/(((BRP + 1))/FCAN) * (SYNC + (TSEG1 + 1) + (TSEG2 + 1))))

3： 最大允许CAN FD误差≤ 1%；误差百分比 = (((计算得到的CAN比特率 - 所需的CAN比特率)/所需的CAN比特率) * 100)。

寄存器56-3： CFDxDBTCFG：数据位时间配置寄存器（ x = 1-4）（续）

bit 3-0 **SJW<3:0>**：同步跳转宽度位

1111 = 长度为16个TQ

·

·

·

0000 = 长度为1个TQ

注 1：只能在配置模式 （OPMOD<2:0>位 （**CFDx**CON<23:21>） = 100）下修改此寄存器。

2：以下内容适用于此寄存器：

- TQ = ((BRP + 1))/FCAN)
- 标称位周期 = (TQ * ((SYNC + (TSEG1 + 1) + (TSEG2 +1)))
- 标称比特率计算方式为(1/位周期)

  或
- CAN比特率计算方式为(1/(((BRP + 1))/FCAN) * (SYNC + (TSEG1 + 1) + (TSEG2 + 1))))

3：最大允许CAN FD误差≤ 1%；误差百分比 = (((计算得到的CAN比特率 - 所需的CAN比特率)/所需的CAN比特率) * 100)。

 © 2019 Microchip Technology Inc.

寄存器56-4： CFDxTDC：发送器延时补偿寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
	—	—	—	—	—	—	EDGFLTEN	SID11EN
23:16	U-0	U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0
	—	—	—	—	—	—	TDCMOD<1:0>	
15:8	U-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
	—	TDCO<6:0>						
7:0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	—	TDCV<5:0>					

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-24 **未实现：**读为0

bit 25 **EDGFLTEN：**在总线集成状态下使能边沿滤波位
1 = 根据ISO11898-1:2015标准使能边沿滤波
0 = 禁止边沿滤波

bit 24 **SID11EN：**使能CAN FD基本格式报文中的12位SID位
1 = RRS用作CAN FD基本格式报文中的SID11：SID<11:0> = {SID<10:0>, SID11}
0 = 不使用RRS；SID<10:0>符合ISO11898-1:2015标准

bit 23-18 **未实现：**读为0

bit 17-16 **TDCMOD<1:0>：**发送器延时补偿模式位
二次采样点 （Secondary Sample Point， SSP）。
10 = 自动；测量延时并与CFDxDBTCFG.TSEG1相加；再加上TDCO
11 = 自动；测量延时并与CFDxDBTCFG.TSEG1相加；再加上TDCO
01 = 手动；不测量延时，使用寄存器的TDCV + TDCO
00 = 禁止

bit 15 **未实现：**读为0

bit 14-8 **TDCO<6:0>：**发送器延时补偿偏移位
二次采样点 （SSP）。二进制补码；偏移可以为正值、零或负值。
1111111 = -64个SYSCLK
•
•
•
0111111 = 63个SYSCLK
•
•
•
0000000 = 0个SYSCLK

bit 7-6 **未实现：**读为0

bit 5-0 **TDCV<5:0>：**发送器延时补偿值位；二次采样点 （SSP）
111111 = 63个SYSCLK
•
•
•
000000 = 0个SYSCLK

注： 只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改此寄存器。

寄存器56-5： CFDxTBC：CAN时基计数器寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	TBC<31:24>							
23:16	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	TBC<23:16>							
15:8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	TBC<15:8>							
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	TBC<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-0 **TBC<31:0>**：CAN时基计数器位

这是自由运行的32位时间戳定时器。当TTBCEN位 （CFDxTSCON<8>）置1时，每经过TBCPRE<9:0>（CFDxTSCON<9:0>）个SYSCLK时钟周期时，该定时器会递增一次。

注 1： 当TBCEN位 （CFDxTSCON<16>） = 0时， TBC将停止并复位以节省功耗。

2： 对CFDxTBC寄存器的任何写操作都会使TBC的预分频比计数复位 （TBCPRE<9:0> （CFDxTSCON<9:0>）不受影响）。

注： 定时器/计数器每经过一个SYSCLK时钟周期递增一次，然后归零。
有两个主要的时间戳寄存器：
- CFDxTBC：32位时基计数器
- CFDxTSCON：时间戳控制寄存器

可选的时间戳事件：
- 经典CAN帧：帧起始 （SOF）与帧结束 （EOF）
- CAN FD帧：在SOF/FDF与EOF之后

 © 2019 Microchip Technology Inc.

寄存器56-6： CFDxTSCON：CAN时间戳控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
23:16	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
	—	—	—	—	—	TSRES	TSEOF	TBCEN
15:8	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
	—	—	—	—	—	—	TBCPRE<9:8>	
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	TBCPRE<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-19 **未实现：**读为0

bit 18 **TSRES：**时间戳分辨率位
仅限FD帧。
1 = 在FDF位的后一位的采样点处
0 = 在SOF的采样点处

bit 17 **TSEOF：**EOF时间戳位
1 = 在帧生效后添加时间戳 （11898-1 10.7）：
- 在EOF的倒数第二位之前未产生接收错误
- 在EOF结束之前未产生发送错误

0 = 在帧 “开始”时添加时间戳：
- 经典帧：在SOF的采样点处
- FD帧：请参见TSRES位

bit 16 **TBCEN：**时基计数器使能位
1 = 使能TBC
0 = 停止并复位TBC

bit 15-10 **未实现：**读为0

bit 9-0 **TBCPRE<9:0>：**CAN时基计数器预分频比位
111111111 = 每经过1024个SYSCLK时钟周期时， TBC递增一次
.
.
.
000000000 = 每经过1个SYSCLK时钟周期时， TBC递增一次

寄存器56-7： CFDxVEC：中断代码寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	R-1	R-0	R-0	R-0	R-0	R-0	R-0
	—	RXCODE<6:0>(1)						
23:16	U-0	R-1	R-0	R-0	R-0	R-0	R-0	R-0
	—	TXCODE<6:0>(1)						
15:8	U-0	U-0	U-0	R-0	R-0	R-0	R-0	R-0
	—	—	—	FILHIT<4:0>(1)				
7:0	U-0	R-1	R-0	R-0	R-0	R-0	R-0	R-0
	—	ICODE<6:0>(1)						

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **未实现：**读为0

bit 30-24 **RXCODE<6:0>：**接收中断标志代码位(1)

1111111 = 保留
.
.
.
1000001 = 保留
1000000 = 无中断
0111111 = 保留
.
.
.
0100000 = 保留
0011111 = FIFO 31中断 （RFIF<31>置1）
.
.
.
0000010 = FIFO 2中断 （RFIF<2>置1）
0000001 = FIFO 1中断 （RFIF<1>置1）
0000000 = 保留。 FIFO 0无法接收。

bit 23 **未实现：**读为0

注 1： 如果多个中断处于待处理状态，则将指示编号最高的中断。

 © 2019 Microchip Technology Inc.

寄存器56-7： CFDxVEC：中断代码寄存器（ x = 1-4）（续）

bit 22-16 **TXCODE<6:0>**：发送中断标志代码位[(1)]

1111111 = 保留
.
.
.
1000001 = 保留
1000000 = 无中断
0111111 = 保留
.
.
.
0100000 = 保留
0011111 = FIFO 31中断 （TFIF<31>位置1）
.
.
.
0000001 = FIFO 1中断 （TFIF<1>位置1）
0000000 = FIFO 0中断 （TFIF<0>位置1）

bit 15-13 **未实现**：读为0

bit 12-8 **FILHIT<4:0>**：命中过滤器编号位[(1)]

11111 = 过滤器31
11110 = 过滤器30
.
.
.
00001 = 过滤器1
00000 = 过滤器0

bit 7 **未实现**：读为0

bit 6-0 **ICODE<6:0>**：中断标志代码位[(1)]

1111111 = 保留
.
.
.
1001011 = 保留
1001010 = 发送尝试中断 （CFDxTXATIF寄存器中的任意一位置1）
1001001 = 发送事件FIFO中断 （CFDxTEFIF寄存器中的任意一位置1）
1001000 = 出现无效报文 （IVMIF/IE）
1000111 = CAN模块模式发生变化 （MODIF/IE）
1000110 = CAN定时器溢出 （CTMRIF/IE）
1000101 = 发送/接收MAB上溢/下溢（接收：在将前一个报文存储到存储器之前接收的报文；发送：无法足够快速地为发送MAB提供数据以发送一致的数据）（SERRIF/IE）
1000100 = 地址错误中断 （向系统送入了非法FIFO地址）（SERRIF/IE）
1000011 = 接收FIFO溢出中断 （CFDxRXOVIF中的任意一位置1）
1000010 = 唤醒中断 （WAKIF/WAKIE）
1000001 = 错误中断 （CERRIF/IE）
1000000 = 无中断
0111111 = 保留
0100000 = 保留
0011111 = FIFO 31中断 （TFIF<31>或RFIF<31>置1）
.
.
.
0000001 = FIFO 1中断 （TFIF<1>或RFIF<1>置1）
0000000 = FIFO 0中断 （TFIF<0>置1）

注 1： 如果多个中断处于待处理状态，则将指示编号最高的中断。

寄存器56-8： CFDxINT：中断寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
	IVMIE	WAKIE	CERRIE	SERRIE	RXOVIE	TXATIE	—	—
23:16	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	—	—	TEFIE	MODIE	TBCIE	RXIE	TXIE
15:8	HS/C-0	HS/C-0	HS/C-0	HS/C-0	R-0	R-0	U-0	U-0
	IVMIF(1)	WAKIF(1)	CERRIF(1)	SERRIF(1)	RXOVIF	TXATIF	—	—
7:0	U-0	U-0	U-0	R-0	HS/C-0	HS/C-0	R-0	R-0
	—	—	—	TEFIF	MODIF(1)	TBCIF(1)	RXIF	TXIF

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **IVMIE**：无效报文中断允许位
1 = 允许中断
0 = 禁止中断

bit 30 **WAKIE**：总线唤醒活动中断允许位
1 = 允许中断
0 = 禁止中断

bit 29 **CERRIE**：CAN总线错误中断允许位
1 = 允许中断
0 = 禁止中断

bit 28 **SERRIE**：系统错误中断允许位
1 = 允许中断
0 = 禁止中断

bit 27 **RXOVIE**：接收缓冲区溢出中断允许位
1 = 允许中断
0 = 禁止中断

bit 26 **TXATIE**：发送尝试中断允许位
1 = 允许中断
0 = 禁止中断

bit 25-21 **未实现**：读为0

bit 20 **TEFIE**：发送事件FIFO中断允许位
1 = 允许中断
0 = 禁止中断

bit 19 **MODIE**：模式更改中断允许位
1 = 允许中断
0 = 禁止中断

bit 18 **TBCIE**：CAN定时器中断允许位
1 = 允许中断
0 = 禁止中断

bit 17 **RXIE**：接收对象中断允许位
1 = 允许中断
0 = 禁止中断

注 1： 标志由硬件置1，由用户应用程序清零。

 © 2019 Microchip Technology Inc.

寄存器56-8： CFDxINT：中断寄存器（ x = 1-4）（续）

bit 16 **TXIE**：发送对象中断允许位
1 = 允许中断
0 = 禁止中断

bit 15 **IVMIF**：无效报文中断标志位(1)
1 = 无效报文中断处于待处理状态
0 = 没有无效报文中断处于待处理状态

bit 14 **WAKIF**：总线唤醒活动中断标志位(1)
1 = 唤醒中断处于待处理状态
0 = 没有唤醒中断处于待处理状态

bit 13 **CERRIF**：CAN总线错误中断标志位(1)
1 = CAN总线错误中断处于待处理状态
0 = 没有CAN总线错误中断处于待处理状态

bit 12 **SERRIF**：系统错误中断标志位(1)
1 = 发生了系统错误 （双端口RAM冲突）
0 = 未发生系统错误

bit 11 **RXOVIF**：接收对象溢出中断标志位
1 = 发生了接收对象溢出
0 = 未发生接收对象溢出

bit 10 **TXATIF**：发送尝试中断标志位
1 = 指定32个FIFO中有一个或多个FIFO发生等待处理的发送中断 （具体FIFO通过CFDxTXATIF寄存器的32个位中处于置1状态的位表示）
0 = 没有发送FIFO中断处于待处理状态

bit 9-5 **未实现**：读为0

bit 4 **TEFIF**：发送事件FIFO中断标志位
1 = 发生了接收缓冲区溢出
0 = 未发生接收缓冲区溢出

bit 3 **MODIF**：CAN模式更改中断标志位(1)
1 = CAN模块模式发生更改 （OPMOD已更改以反映REQOP<2:0>位）
0 = CAN模块模式未更改

bit 2 **TBCIF**：CAN定时器溢出中断标志位(1)
1 = TBC已溢出
0 = TBC未溢出

bit 1 **RXIF**：接收对象中断标志位
1 = 接收对象中断处于待处理状态
0 = 没有接收对象中断处于待处理状态

bit 0 **TXIF**：发送对象中断标志位
1 = 发送对象中断处于待处理状态
0 = 没有发送对象中断处于待处理状态

注 1： 标志由硬件置1，由用户应用程序清零。

寄存器56-9： CFDxRXIF：接收中断状态寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RFIF<31:24>(1)							
23:16	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RFIF<23:16>(1)							
15:8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RFIF<15:8>(1)							
7:0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	U-0
	RFIF<7:1>(1)							—

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-1 **RFIF<31:1>**：接收FIFO中断待处理位(1)
1 = 有一个或多个已允许的接收FIFO中断等待处理
0 = 没有已允许的接收FIFO中断等待处理

bit 0 **未实现**：读为0

注 1： RFIF = 已使能的RXFIFO标志的逻辑或运算结果 （这些标志需在相应FIFO寄存器中清零）。

 © 2019 Microchip Technology Inc.

寄存器56-10： CFDxRXOVIF：接收溢出中断状态寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RFOVIF<31:24>(1)							
23:16	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RFOVIF<23:16>(1)							
15:8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RFOVIF<15:8>(1)							
7:0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	U-0
	RFOVIF<7:1>(1)							—

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-1 **RFOVIF<31:1>**：接收FIFO溢出中断待处理位(1)
1 = 中断处于待处理状态
0 = 中断未处于待处理状态

bit 0 未实现：读为0

注 1： RVOVIF （标志需在相应FIFO寄存器中清零）。

寄存器56-11： CFDxTXIF：发送中断状态寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFIF<31:24>(1)							
23:16	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFIF<23:16>(1)							
15:8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFIF<15:8>(1)							
7:0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFIF<7:0>(1)							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-0 **TFIF<31:0>**：发送FIFO/发送队列(2)中断待处理位(1)
1 = 有一个或多个已允许的发送FIFO/发送队列中断处于待处理状态
0 = 没有已允许的发送FIFO/发送队列中断处于待处理状态

注 1： TFIF = 已使能的TXFIFO标志的逻辑或运算结果 （这些标志需在相应FIFO寄存器中清零）。
2： TFIF<0>用于发送队列。

 © 2019 Microchip Technology Inc.

寄存器56-12： CFDxTXATIF：发送尝试中断状态寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFATIF<31:24>(1)							
23:16	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFATIF<23:16>(1)							
15:8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFATIF<15:8>(1)							
7:0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TFATIF<7:0>(1)							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-0 **TFATIF<31:0>**：发送FIFO/发送队列(2)尝试中断待处理位(1)

1 = 指定32个FIFO中有一个或多个FIFO发生等待处理的发送中断（具体FIFO通过CFDxTXATIF寄存器的32个位中处于置1状态的位表示）

0 = 没有发送FIFO中断处于待处理状态

注 1： TFATIF = 已使能的TXFIFO标志的逻辑或运算结果 （这些标志需在相应FIFO寄存器中清零）。

2： TFATIF<0>用于发送队列。

寄存器56-13： CFDxTXREQ：发送请求寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0
	TXREQ<31:24>(1)							
23:16	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0
	TXREQ<23:16>(1)							
15:8	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0
	TXREQ<15:8>(1)							
7:0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0	S, HC-0
	TXREQ<7:0>(1)							

图注：	S = 可置1位		
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-1 **TXREQ<31:1>**：报文发送请求位(1)

TXEN = 1：（对象配置为发送对象）

该位设为1将请求发送报文。

在成功发送对象中排队的报文之后，该位将自动清零。该位不能用于中止发送。

TXEN = 0：（对象配置为接收对象）

该位无影响。

bit 0 **TXREQ<0>**：发送队列报文发送请求位(1)

该位设为1将请求发送报文。

在成功发送对象中排队的报文之后，该位将自动清零。该位不能用于中止发送。

注 1： 在仅监听模式 （REQOP<2:0>位 （CFDxCON<26:24>） = 0’b011）下，将忽略TXREQ<31:0>位。

 © 2019 Microchip Technology Inc.

寄存器56-14： CFDxFIFOBA：报文存储器基址寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FIFOBA<31:24>							
23:16	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FIFOBA<23:16>							
15:8	R/W-0	R/W-0	R/W-0	R/W-0	SR/W-0	R/W-0	R/W-0	R/W-0
	FIFOBA<15:8>							
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
	FIFOBA<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-0 **FIFOBA<31:0>：**报文存储器基址位
定义发送事件FIFO的基址，并后跟报文对象。

注 1： 将强制位<1:0>设为0以按字对齐。
2： 只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改此寄存器。

寄存器56-15： CFDxTXQCON：发送队列控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	PLSIZE<2:0>(1)			FSIZE<4:0>(1)				
23:16	U-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	TXAT<1:0>		TXPRI<4:0>				
15:8	U-0	U-0	U-0	U-0	U-0	S, HC-1	R/W, HC-0	S, HC-0
	—	—	—	—	—	FRESET(2)	TXREQ	UINC
7:0	R-1	U-0	U-0	R/W-0	U-0	R/W-0	U-0	R/W-0
	TXEN	—	—	TXATIE	—	TXQEIE	—	TXQNIE

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-29 **PLSIZE<2:0>**：有效负载大小位(1)
111 = 64个数据字节
110 = 48个数据字节
101 = 32个数据字节
100 = 24个数据字节
011 = 20个数据字节
010 = 16个数据字节
001 = 12个数据字节
000 = 8个数据字节

bit 28-24 **FSIZE<4:0>**：FIFO大小位(1)
11111 = FIFO为32个报文深
.
.
.
00010 = FIFO为3个报文深
00001 = FIFO为2个报文深
00000 = FIFO为1个报文深

bit 23 **未实现**：读为0

bit 22-21 **TXAT<1:0>**：重发尝试位
当RTXAT位 （CFDxCON<8>）置1时，使能该功能。
11 = 重发尝试次数不受限制
10 = 重发尝试次数不受限制
01 = 3次重发尝试
00 = 禁止重发尝试

注：用户应用程序必须能够在正常模式下更改这些位。这样就可在离线后恢复总线，以检查是否重新发送。

注 1： 只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改此位。
2： 该位在配置模式下置1，在正常模式下自动清零。

 © 2019 Microchip Technology Inc.

寄存器56-15： CFDxTXQCON：发送队列控制寄存器（ x = 1-4）（续）

bit 20-16 **TXPRI<4:0>**：报文发送优先级位 （扩展）
11111 = 最高报文优先级
·
·
·
00000 = 最低报文优先级

bit 15-11 未实现：读为0

bit 10 **FRESET**：FIFO复位位(2)
1 = 当该位置1时， FIFO复位；当FIFO复位时，该位由硬件清零。用户应用程序在采取任何操作前都应轮询该位是否清零。
0 = 无影响

bit 9 **TXREQ**：报文发送请求位
1 = 请求发送报文；在发送队列中排队的所有报文都成功发送之后，该位会自动清零
0 = 在该位置1的情况下将其清零会请求中止报文。

bit 8 **UINC**：递增头部/尾部位
当该位置1时， FIFO头部将递增一个报文。

bit 7 **TXEN**：发送使能
1 = 发送报文队列。该位将始终读为1。

bit 6-5 未实现：读为0

bit 4 **TXATIE**：超过发送尝试次数中断允许位
1 = 允许中断
0 = 禁止中断

bit 3 未实现：读为0

bit 2 TXQEIE：发送队列空中断允许位
1 = 允许在发送队列为空时产生中断
0 = 禁止在发送队列为空时产生中断

bit 1 未实现：读为0

bit 0 TXQNIE：发送队列未满中断允许位
1 = 允许在发送队列未满时产生中断
0 = 禁止在发送队列未满时产生中断

注 1： 只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改此位。
2： 该位在配置模式下置1，在正常模式下自动清零。

寄存器56-16：CFDxTXQSTA：发送队列状态寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
23:16	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
15:8	U-0	U-0	U-0	R-0	R-0	R-0	R-0	R-0
	—	—	—	TXQCI<4:0>(1)				
7:0	R-0	R-0	R-0	HS/C-0	U-0	R-0	U-0	R-0
	TXABT(2,3)	TXLARB(2,3)	TXERR(2,3)	TXATIF	—	TXQEIF	—	TXQNIF

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-13 **未实现：**读为0

bit 12-8 **TXQCI<4:0>：**发送队列报文索引位(1)
读取该寄存器将返回一个索引，该索引指向FIFO下一次尝试发送的报文。

bit 7 **TXABT：**报文中止状态位(2,3)
1 = 报文中止
0 = 成功完成报文发送

bit 6 **TXLARB：**报文仲裁失败状态位(2,3)
1 = 报文在发送过程中仲裁失败
0 = 报文在发送过程中没有仲裁失败

bit 5 **TXERR：**在发送过程中检测到错误位(2,3)
1 = 发送报文时发生了总线错误
0 = 发送报文时未发生总线错误

bit 4 **TXATIF：**超过发送尝试次数中断待处理位
1 = 中断处于待处理状态
0 = 中断不处于待处理状态

bit 3 **未实现：**读为0

bit 2 TXQEIF：发送队列空中断标志位
1 = 发送队列为空
0 = 发送队列非空，至少有一个报文在排队等待发送

bit 1 **未实现：**读为0

bit 0 TXQNIF：发送队列未满中断标志位
1 = 发送队列未满
0 = 发送队列已满

注 1： TXQCI<4:0>为发送队列中的报文分配零值索引值。如果发送队列为4个报文深 （FSIZE<4:0>位（CFDxFIFOCONn<28:24>） = 5’h03），则TXQCI<4:0>将根据发送队列的状态从0到3中取值。
2： 每次读取该寄存器或发送队列复位时，该位都会复位。
3： 报文完成 （或中止）或发送队列复位时，该位会更新。

 © 2019 Microchip Technology Inc.

寄存器56-17：CFDxFIFOCONn：FIFO控制寄存器（ x = 1-4；n = 1-31）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	PLSIZE<2:0>(1)			FSIZE<4:0>(1)				
23:16	U-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	TXAT<1:0>		TXPRI<4:0>				
15:8	U-0	U-0	U-0	U-0	U-0	S, HC-1	R/W, HC-0	S, HC-0
	—	—	—	—	—	FRESET	TXREQ	UINC
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	TXEN(1)	RTREN	RXTSEN(1)	TXATIE	RXOVIE	TFERFFIE	TFHRFHIE	TFNRFNIE

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-29 **PLSIZE<2:0>**：有效负载大小位(1)
111 = 64个数据字节
110 = 48个数据字节
101 = 32个数据字节
100 = 24个数据字节
011 = 20个数据字节
010 = 16个数据字节
001 = 12个数据字节
000 = 8个数据字节

bit 28-24 **FSIZE<4:0>**：FIFO大小位(1)
11111 = FIFO为32个报文深
.
.
.
00010 = FIFO为3个报文深
00001 = FIFO为2个报文深
00000 = FIFO为1个报文深

bit 23 未实现：读为0

bit 22-21 **TXAT<1:0>**：重发尝试位
当RTXAT位 （CFDxCON<8>）置1时，使能该功能。
11 = 重发尝试次数不受限制
10 = 重发尝试次数不受限制
01 = 3次重发尝试
00 = 禁止重发尝试
注：应用程序必须能够在正常模式下更改此位。这样就可在离线后恢复总线，以检查是否重新发送

注 1： 只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改此位。
2： FRESET在配置模式下置1，在正常模式下自动清零。
3： 报文完成 （或中止）或FIFO复位时，该位会更新。

寄存器56-17：CFDxFIFOCONn：FIFO控制寄存器（ x = 1-4；n = 1-31）（续）

bit 20-16 **TXPRI<4:0>**：报文发送优先级位 （扩展）

11111 = 最高报文优先级
.
.
.
00000 = 最低报文优先级

bit 15-11 **未实现**：读为0

bit 10 **FRESET**：FIFO复位位(2)

1 = 当该位置1时， FIFO复位；当FIFO复位时，该位由硬件清零。用户在采取任何操作前都应轮询该位是否清零
0 = 无影响

bit 9 **TXREQ**：报文发送请求位(3)

TXEN = 1：（FIFO配置为发送FIFO）

1 = 请求发送报文；在FIFO中排队的所有报文都成功发送之后，该位将自动清零
0 = 在该位置1的情况下将其清零会请求中止报文。

TXEN = 0：（FIFO配置为接收FIFO）

该位无影响。

bit 8 **UINC**：递增头部/尾部位

TXEN = 1：（FIFO配置为发送FIFO）

当该位置1时， FIFO头部将递增一个报文

TXEN = 0：（FIFO配置为接收FIFO）

当该位置1时， FIFO尾部将递增一个报文

bit 7 **TXEN**：发送/接收缓冲区选择位(1)

1 = 发送报文对象
0 = 接收报文对象

bit 6 **RTREN**：自动远程发送 （RTR）使能位

1 = 接收到远程发送时， TXREQ将置1。
0 = 接收到远程发送时， TXREQ不受影响。

bit 5 **RXTSEN**：接收的报文时间戳使能位(1)

1 = 捕捉RAM中接收到的报文对象的时间戳。
0 = 不捕捉时间戳。

注： 只能在配置模式下更改此位，因为该位用于进行地址计算。

bit 4 **TXATIE**：超过发送尝试次数中断允许位

1 = 允许中断
0 = 禁止中断

bit 3 **RXOVIE**：溢出中断允许位

1 = 允许在发生溢出时产生中断
0 = 禁止在发生溢出时产生中断

注 1： 只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改此位。
2： FRESET在配置模式下置1，在正常模式下自动清零。
3： 报文完成 （或中止）或FIFO复位时，该位会更新。

 © 2019 Microchip Technology Inc.

寄存器56-17：CFDxFIFOCONn：FIFO控制寄存器（ x = 1-4；n = 1-31）（续）

bit 2 **TFERFFIE**：发送/接收FIFO空/满中断允许位

TXEN = 1：（FIFO配置为发送FIFO）

允许发送FIFO空中断

1 = 允许在FIFO为空时产生中断

0 = 禁止在FIFO为空时产生中断

TXEN = 0：（FIFO配置为接收FIFO）

允许接收FIFO满中断

1 = 允许在FIFO满时产生中断

0 = 禁止在FIFO满时产生中断

bit 1 **TFHRFHIE**：发送/接收FIFO半空/半满中断允许位

TXEN = 1：（FIFO配置为发送FIFO）

允许发送FIFO半空中断

1 = 允许在FIFO半空时产生中断

0 = 禁止在FIFO半空时产生中断

TXEN = 0：（FIFO配置为接收FIFO）

允许接收FIFO半满中断

1 = 允许在FIFO半满时产生中断

0 = 禁止在FIFO半满时产生中断

bit 0 **TFNRFNIE**：发送/接收FIFO未满/非空中断允许位

TXEN = 1：（FIFO配置为发送FIFO）

允许发送FIFO未满中断

1 = 允许在FIFO未满时产生中断

0 = 禁止在FIFO未满时产生中断

TXEN = 0：（FIFO配置为接收FIFO）

允许接收FIFO非空中断

1 = 允许在FIFO非空时产生中断

0 = 禁止在FIFO非空时产生中断

注 1：只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改此位。

2：FRESET在配置模式下置1，在正常模式下自动清零。

3：报文完成 （或中止）或FIFO复位时，该位会更新。

寄存器56-18：CFDxFIFOSTAn：FIFO状态寄存器（ x = 1-4；n = 1-31）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
23:16	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
15:8	U-0	U-0	U-0	R-0	R-0	R-0	R-0	R-0
	—	—	—	FIFOCI<4:0>(1)				
7:0	R-0	R-0	R-0	HS/C-0	HS/C-0	R-0	R-0	R-0
	TXABT(2,3)	TXLARB(2,3)	TXERR(2,3)	TXATIF	RXOVIF	TFERFFIF	TFHRFHIF	TFNRFNIF

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-13 **未实现：**读为0

bit 12-8 **FIFOCI<4:0>：**FIFO报文索引位(1)

TXEN = 1：（FIFO配置为发送缓冲区）
读取该寄存器将返回一个索引，该索引指向FIFO下一次尝试发送的报文。

TXEN = 0：（FIFO配置为接收缓冲区）
读取该寄存器将返回一个索引，该索引指向FIFO用于保存的下一个报文。

bit 7 **TXABT：**报文中止状态位(2,3)

1 = 报文中止
0 = 成功完成报文发送

bit 6 **TXLARB：**报文仲裁失败状态位(2,3)

1 = 报文在发送过程中仲裁失败
0 = 报文在发送过程中没有仲裁失败

bit 5 **TXERR：**在发送过程中检测到错误位(2,3)

1 = 发送报文时发生了总线错误
0 = 发送报文时未发生总线错误

bit 4 **TXATIF：**超过发送尝试次数中断待处理位

TXEN = 1：（FIFO配置为发送缓冲区）
1 = 中断处于待处理状态
0 = 中断未处于待处理状态

TXEN = 0：（FIFO配置为接收缓冲区）
未用 （读为0）

bit 3 **RXOVIF：**接收FIFO溢出中断标志位

TXEN = 1：（FIFO配置为发送缓冲区）
未用 （读为0）

TXEN = 0：（FIFO配置为接收缓冲区）
1 = 发生了溢出事件
0 = 未发生溢出事件

注 1： FIFOCI<4:0>为FIFO中的报文分配零值索引值。如果FIFO为4个报文深 （FSIZE = 5’h03），则FIFOCI将根据FIFO的状态从0到3中取值。

2： 每次读取该寄存器或发送队列复位时，该位都会复位。

3： 报文完成 （或中止）或FIFO复位时，该位会更新。

 © 2019 Microchip Technology Inc.

寄存器56-18：CFDxFIFOSTAn：FIFO状态寄存器（ x = 1-4；n = 1-31）（续）

bit 2 **TFERFFIF**：发送/接收FIFO空/满中断标志位
TXEN = 1：（FIFO配置为发送FIFO）
发送FIFO空中断标志
1 = FIFO为空
0 = FIFO非空，至少有一个报文在排队等待发送
TXEN = 0：（FIFO配置为接收FIFO）
接收FIFO满中断标志
1 = FIFO已满
0 = FIFO未满

bit 1 **TFHRFHIF**：发送/接收FIFO半空/半满中断标志位
TXEN = 1：（FIFO配置为发送FIFO）
发送FIFO半空中断标志
1 = FIFO低于或处于半满状态
0 = FIFO超过半满状态
TXEN = 0：（FIFO配置为接收FIFO）
接收FIFO半满中断标志
1 = FIFO超过或处于半满状态
0 = FIFO低于半满状态

bit 0 **TFNRFNIF**：发送/接收FIFO未满/非空中断标志位
TXEN = 1：（FIFO配置为发送FIFO）
发送FIFO未满中断标志
1 = FIFO未满
0 = FIFO已满
TXEN = 0：（FIFO配置为接收FIFO）
接收FIFO非空中断标志
1 = FIFO非空 （至少有一个报文）
0 = FIFO为空

注 1： FIFOCI<4:0>为FIFO中的报文分配零值索引值。如果FIFO为4个报文深 （FSIZE = 5’h03），则FIFOCI将根据FIFO的状态从0到3中取值。
2： 每次读取该寄存器或发送队列复位时，该位都会复位。
3： 报文完成 （或中止）或FIFO复位时，该位会更新。

寄存器56-19：CFDxTEFCON：发送事件FIFO控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	—	—	FSIZE<4:0>(1)				
23:16	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
15:8	U-0	U-0	U-0	U-0	U-0	S, HC-1	U-0	S, HC-0
	—	—	—	—	—	FRESET	—	UINC
7:0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	—	TEFTSEN(1)	—	TEFOVIE	TEFFIE	TEFHIE	TEFNEIE

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-29 未实现：读为0

bit 28-24 **FSIZE<4:0>**：FIFO大小位(1)

11111 = FIFO为32个报文深
.
.
.
00010 = FIFO为3个报文深
00001 = FIFO为2个报文深
00000 = FIFO为1个报文深

bit 23-11 未实现：读为0

bit 10 **FRESET**：FIFO复位位

1 = 当该位置1时， FIFO复位；当FIFO复位时，该位由硬件清零。用户在采取任何操作前都应轮询该位是否清零
0 = 无影响

bit 9 未实现：读为0

bit 8 **UINC**：递增尾部位

当该位置1时， FIFO尾部递增一个报文

bit 7-6 未实现：读为0

bit 5 **TEFTSEN**：发送事件FIFO时间戳使能位(1)

1 = 为TEF中的元素添加时间戳
0 = 不为TEF中的元素添加时间戳

bit 4 未实现：读为0

bit 3 **TEFOVIE**：发送事件FIFO溢出中断允许位

1 = 允许在发生溢出时产生中断
0 = 禁止在发生溢出时产生中断

bit 2 **TEFFIE**：发送事件FIFO满中断允许位

1 = 允许在FIFO满时产生中断
0 = 禁止在FIFO满时产生中断

bit 1 **TEFHIE**：发送事件FIFO半满中断允许位

1 = 允许在FIFO半满时产生中断
0 = 禁止在FIFO半满时产生中断

bit 0 **TEFNEIE**：发送事件FIFO非空中断允许位

1 = 允许在FIFO非空时产生中断
0 = 禁止在FIFO非空时产生中断

注 1： 只能在配置模式 （OPMOD<2:0>位 （CFDxCON<23:21>） = 100）下修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-20：CFDxTEFSTA：发送事件FIFO状态寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
23:16	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
15:8	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
7:0	U-0	U-0	U-0	U-0	HS和C-0	R-0	R-0	R-0
	—	—	—	—	TEFOVIF	TEFFIF(1)	TEFHIF(1)	TEFNEIF(1)

图注：	C = 可清零位	HS = 由硬件置1	
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-4 **未实现：**读为0

bit 3 **TEFOVIF：**发送事件FIFO溢出中断标志位
1 = 发生了溢出事件
0 = 未发生溢出事件

bit 2 **TEFFIF：**发送事件FIFO满中断标志位(1)
1 = FIFO已满
0 = FIFO未满

bit 1 **TEFHIF：**发送事件FIFO半满中断标志位(1)
1 = FIFO超过或处于半满状态
0 = FIFO低于半满状态

bit 0 **TEFNEIF：**发送事件FIFO非空中断标志位(1)
1 = FIFO非空 （至少有一个报文）
0 = FIFO为空

注 1： 该位是只读位，用于反映FIFO的状态。

寄存器56-21：CFDxFIFOUAn：定义寄存器（ x = 1-4；n = 1-31）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	FIFOUA<31:24>							
23:16	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	FIFOUA<23:16>							
15:8	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	FIFOUA<15:8>							
7:0	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	FIFOUA<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-0 **FIFOUA<31:0>**：FIFO用户地址位

TXEN = 1：（FIFO配置为发送缓冲区）

读取该寄存器将返回用于写入下一个报文的地址 （FIFO头部）。

TXEN = 0：（FIFO配置为接收缓冲区）

读取该寄存器将返回用于读取下一个报文的地址 （FIFO尾部）。

注 1： 在配置模式下，不能保证可以正确读取该寄存器，应当仅在模块不处于配置模式时访问该寄存器。

2： 该寄存器提供报文存储器中用于存储FIFO中的下一个元素的字节地址。应用程序直接使用此地址来访问RAM。

- 对于接收FIFO，该地址指向应用程序应从中读取数据的下一个元素
- 对于发送FIFO，该地址指向应用程序应向其写入数据的下一个元素

在访问该寄存器后，用户应用程序必须将CFDxFIFOCONn寄存器的UINC位置1，以更新FIFO指针。

 © 2019 Microchip Technology Inc.

寄存器56-22：CFDxTEFUA：发送事件FIFO用户地址寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	TEFUA<31:24>							
23:16	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	TEFUA<23:16>							
15:8	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	TEFUA<15:8>							
7:0	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	TEFUA<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-0 **TEFUA<31:0>：**发送事件FIFO用户地址位
读取该寄存器将返回用于读取下一个事件的地址 （FIFO尾部）。

注 1： 在配置模式下，不能保证可以正确读取该寄存器，应当仅在模块不处于配置模式时访问该寄存器。
2： 可以通过此寄存器访问发送事件FIFO中的元素。该寄存器提供报文存储器中用于存储缓冲区中的下一个元素的字节地址。应用程序直接使用此地址来访问RAM。该地址指向应用程序应从中读取数据的下一个元素。在访问该寄存器后，用户应用程序必须将CFDxTEFCON寄存器的UINC位置1，以更新FIFO指针。

寄存器56-23：CFDxTXQUA：发送队列用户地址寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	TXQUA<31:24>							
23:16	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	TXQUA<23:16>							
15:8	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	TXQUA<15:8>							
7:0	R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
	TXQUA<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-0 **TXQUA<31:0>：**发送队列用户地址位
读取该寄存器将返回用于写入下一个报文的地址 （发送队列头部）。

注 1： 在配置模式下，不能保证可以正确读取该寄存器，应当仅在模块不处于配置模式时访问该寄存器。
2： 可以通过此寄存器访问发送队列中的元素。该寄存器提供报文存储器中用于存储发送队列中的下一个元素的字节地址。应用程序直接使用此地址来访问RAM。该地址指向应用程序应向其写入数据的下一个元素。在访问该寄存器后，用户应用程序必须将CFDxTXQCON寄存器的UINC位置1，以更新TXQ指针。

 © 2019 Microchip Technology Inc.

寄存器56-24：CFDxTREC：发送/接收错误计数寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
23:16	U-0	U-0	R-1	R-0	R-0	R-0	R-0	R-0
	—	—	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN
15:8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	TERRCNT<7:0>							
7:0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	RERRCNT<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-22 未实现：读为0

bit 21 **TXBO**：发送器处于离线错误状态位 （TERRCNT > 255）
在配置模式下，由于CAN FD模块未处于总线上而将TXBO置1。
1 = 表示发送错误的数量大于255。因此， CAN FD模块将自动进入离线模式。
0 = 表示发送错误的数量小于255

bit 20 **TXBP**：发送器处于被动错误状态位 （TERRCNT > 127）
1 = 表示发送错误的数量大于127。因此， CAN FD模块将自动进入总线被动模式。
0 = 表示发送错误的数量小于或等于127

bit 19 **RXBP**：接收器处于被动错误状态位 （RERRCNT > 127）
1 = 表示接收错误的数量大于127。因此， CAN FD模块将自动进入总线被动模式。
0 = 表示发送错误的数量小于或等于127

bit 18 **TXWARN**：发送器处于错误警告状态位 （128 > TERRCNT > 95）
1 = 表示发送错误的数量小于128，但大于95。
0 = 表示发送错误的数量大于或等于95

bit 17 **RXWARN**：接收器处于错误警告状态位 （128 > RERRCNT > 95）
1 = 表示接收错误的数量小于128，但大于95。
0 = 表示接收错误的数量大于或等于95

bit 16 **EWARN**：发送器或接收器处于错误警告状态位
1 = TXWARN或RXWARN置1
0 = 无发送或接收警告

注： 仲裁和数据阶段分别有单独的接收与发送错误计数器，这些计数器具有以下功能：
- 包含成功报文计数器
- 跟踪接收到的错误状态指示 （Error State Indicator， ESI）

寄存器56-24：CFDxTREC：发送/接收错误计数寄存器（ x = 1-4）（续）

bit 15-8 **TERRCNT<7:0>**：发送错误计数器位

11111111 = 至少255个发送错误。如果TERRCNT<7:0>位大于255，则TXBO位等于1。
11111110 = 254个发送错误
•
•
•
00000001 = 1个发送错误
00000000 = 无发送错误

注：具体情况如下：

- 如果TERRCNT大于95且小于128，则TXWARN位等于1
- 如果TERRCNT大于95，则TXBP位等于1，并且发送进入总线被动模式
- 如果TERRCNT大于255，则TXBO和CERRIF位置1，并且CAN FD模块进入离线模式。一旦进入离线模式， TERRCNT位就会装入值128，以启动离线恢复序列，即等待检测128个总线空闲事件。
- 读取TXBO （CFDxTREC<21>）位的状态，以确定发送器是否处于离线状态。

bit 7-0 **RERRCNT<7:0>**：接收错误计数器位

11111111 = 至少255个接收错误
11111110 = 254个接收错误
•
•
•
00000001 = 1个接收错误
00000000 = 无接收错误

注：具体情况如下：

- 如果RERRCNT大于127，则RXBP位等于1
- 如果RERRCNT大于95且小于128，则RXWARN位等于1

注：仲裁和数据阶段分别有单独的接收与发送错误计数器，这些计数器具有以下功能：
• 包含成功报文计数器
• 跟踪接收到的错误状态指示 （Error State Indicator， ESI）

 © 2019 Microchip Technology Inc.

寄存器56-25：CFDxBDIAG0：总线诊断寄存器0（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	DTERRCNT<7:0>							
23:16	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	DRERRCNT<7:0>							
15:8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	NTERRCNT<7:0>							
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	NRERRCNT<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31-24 **DTERRCNT<7:0>**：数据比特率发送错误计数器位

bit 23-16 **DRERRCNT<7:0>**：数据比特率接收错误计数器位

bit 15-8 **NTERRCNT<7:0>**：标称比特率发送错误计数器位

bit 7-0 **NRERRCNT<7:0>**：标称比特率接收错误计数器位

注： 该寄存器分别跟踪标称比特率阶段和数据比特率阶段的总线错误。这些计数器的工作方式与CFDxTREC寄存器有所不同：

- 任何总线错误都会使这些计数器递增一次
- 这些计数器不会递减
- 读取寄存器会使这些计数器清零

寄存器56-26：CFDxBDIAG1：总线诊断寄存器1（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
	DLCMM	ESI	DCRCERR	DSTUFERR	DFORMERR	—	DBIT1ERR	DBIT0ERR
23:16	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	TXBOERR	—	NCRCERR	NSTUFERR	NFORMERR	NACKERR	NBIT1ERR	NBIT0ERR
15:8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	EFMSGCNT<15:8>							
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	EFMSGCNT<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **DLCMM：**数据长度码 （DLC）不匹配状态位
1 = 有效负载大小错误。在发送或接收期间，数据长度码超过PLSIZE<2:0>位（CFDxFIFOCONn<31:29>）中定义的FIFO元素的有效负载大小。
0 = 未发生有效负载大小错误

bit 30 **ESI：**错误状态指示 （ESI）标志状态位
1 = 接收的CAN FD报文的ESI标志已置1
0 = 接收的CAN FD报文的ESI标志未置1

bit 29 **DCRCERR：**数据CRC错误状态位
1 = 接收的报文的CRC校验和不正确。输入报文的CRC与通过接收的数据计算得到的CRC不匹配
0 = 接收的报文未发生数据CRC错误

bit 28 **DSTUFERR：**数据填充错误状态位
1 = 在所接收报文的一部分中，序列中包含了5个以上相等位，而报文中不允许出现这种序列
0 = 接收的报文没有数据错误

bit 27 **DFORMERR：**数据格式错误状态位
1 = 接收的帧的数据固定格式部分有格式错误
0 = 未发生格式错误

bit 26 **未实现：**读为0

bit 25 **DBIT1ERR：**数据位逻辑1错误状态位
1 = 在发送报文数据 （仲裁字段除外）期间，器件要发送隐性电平 （逻辑值为1的位），但监视的总线值为显性。
0 = 未发生数据逻辑1位发送错误

注： 该寄存器可以显示自上次读取后发生的错误类型。发生错误时，相应位会置1，但每次读取该寄存器或R-M-W位操作指令时，所有位都会清零。该寄存器会分别跟踪数据比特率阶段和标称比特率阶段的错误。可以搭配错误计数器和错误标志使用无错误报文计数器位 （EFMSGCNT<15:0>）来确定总线的质量。

 © 2019 Microchip Technology Inc.

寄存器56-26：CFDxBDIAG1：总线诊断寄存器1（ x = 1-4）（续）

bit 24 **DBIT0ERR：**数据位逻辑0错误状态位

1 = 在发送报文数据 （或应答位、主动错误标志或过载标志）期间，器件要发送显性电平 （逻辑值为0的数据或标识符位），但监视的总线值为隐性。在离线恢复期间，每监视到一个包含11个隐性位的序列时，该状态便会置1。这使得CPU能够监视离线恢复序列的进度 （表示总线未滞留在显性状态或未受到持续干扰）。

0 = 未发生数据逻辑0位发送错误

bit 23 **TXBOERR：**发送离线错误状态位

1 = 发生了发送错误，器件进入离线模式并自动恢复

0 = 未发生发送离线错误

bit 22 **未实现：**读为0

bit 21 **NCRCERR：**标称CRC错误状态位

1 = 接收的标称报文的CRC校验和不正确。输入标称报文的CRC与通过接收的数据计算得到的CRC不匹配。

0 = 接收的标称报文未发生CRC错误

bit 20 **NSTUFERR：**标称填充错误状态位

1 = 在所接收标称报文的一部分中，序列中包含了5个以上相等位，而报文中不允许出现这种序列

0 = 未发生标称位填充错误

bit 19 **NFORMERR：**标称格式错误状态位

1 = 接收的标称帧的固定格式部分有格式错误

0 = 未发生标称格式错误

bit 18 **NACKERR：**未应答错误状态位

1 = 发送的报文未获应答

0 = 发送的报文已获应答

bit 17 **NBIT1ERR：**标称位逻辑1错误状态位

1 = 在发送标称报文 （仲裁字段除外）期间，器件要发送隐性电平 （逻辑值为1的位），但监视的总线值为显性

0 = 未发生标称位逻辑1错误

bit 16 **NBIT0ERR：**标称位逻辑0错误状态位

1 = 在发送标称报文 （或应答位、主动错误标志或过载标志）期间，器件要发送显性电平 （逻辑值为0的数据或标识符位），但监视的总线值为隐性。在离线恢复期间，每监视到一个包含11个隐性位的序列时，该状态便会置1。这使得CPU能够监视离线恢复序列的进度 （表示总线未滞留在显性状态或未受到持续干扰）。

0 = 未发生标称位逻辑0错误

bit 15-0 **EFMSGCNT<15:0>：**无错误报文计数器状态位

1111111111111111 = 自上次读取该寄存器后的无错误报文数为65,536个

•

•

•

0000000000000000 = 自上次读取该寄存器后的无错误报文数为0个

注： 总线上的任何无错误报文都会使 EFMSGCNT<15:0> 位递增一次。每次读取该寄存器都会使这些位清零。

注： 该寄存器可以显示自上次读取后发生的错误类型。发生错误时，相应位会置1，但每次读取该寄存器或R-M-W位操作指令时，所有位都会清零。该寄存器会分别跟踪数据比特率阶段和标称比特率阶段的错误。可以搭配错误计数器和错误标志使用无错误报文计数器位 （EFMSGCNT<15:0>）来确定总线的质量。

寄存器56-27：CFDxFLTCON0：过滤器控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN3	—	—	F3BP<4:0>(1)				
23:16	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN2	—	—	F2BP<4:0>(1)				
15:8	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN1	—	—	F1BP<4:0>(1)				
7:0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN0	—	—	F0BP<4:0>(1)				

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **FLTEN3**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 30-29 **未实现**：读为0

bit 28-24 **F3BP<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 23 **FLTEN2**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 22-21 **未实现**：读为0

bit 20-16 **F2P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 15 **FLTEN1**使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 14-13 **未实现**：读为0

注 1：仅当相应过滤器处于禁止状态（FLTENx位（CFDxFLTCONn）= 0）时才能修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-27：CFDxFLTCON0：过滤器控制寄存器（ x = 1-4） （续）

bit 12-8 **F1BP<4:0>**：过滤器n命中时指向对象的指针位(1)

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 7 **FLTEN0**：使能过滤器n接收报文位

1 = 使能过滤器
0 = 禁止过滤器

bit 6-5 未实现：读为0

bit 4-0 **F0BP<4:0>**：过滤器n命中时指向对象的指针位(1)

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

注 1： 仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

寄存器56-28：CFDxFLTCON1：过滤器控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN7	—	—	F7BP<4:0>(1)				
23:16	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN6	—	—	F6BP<4:0>(1)				
15:8	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN5	—	—	F5BP<4:0>(1)				
7:0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN4	—	—	F4BP<4:0>(1)				

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **FLTEN7**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 30-29 **未实现**：读为0

bit 28-24 **F7BP<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 23 **FLTEN6**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 22-21 **未实现**：读为0

bit 20-16 **F6P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 15 **FLTEN5**使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 14-13 **未实现**：读为0

注 1：仅当相应过滤器处于禁止状态（FLTENx位（CFDxFLTCONn）= 0）时才能修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-28：CFDxFLTCON1：过滤器控制寄存器（ x = 1-4） （续）

bit 12-8 **F5BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 7 **FLTEN4**：使能过滤器n接收报文位

1 = 使能过滤器
0 = 禁止过滤器

bit 6-5 **未实现**：读为0

bit 4-0 **F4BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

注 1： 仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

寄存器56-29：CFDxFLTCON2：过滤器控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN11	—	—	F11BP<4:0>(1)				
23:16	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN10	—	—	F10BP<4:0>(1)				
15:8	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN9	—	—	F9BP<4:0>(1)				
7:0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN8	—	—	F8BP<4:0>(1)				

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **FLTEN11**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 30-29 **未实现**：读为0

bit 28-24 **F11BP<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 23 **FLTEN10**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 22-21 **未实现**：读为0

bit 20-16 **F10P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 15 **FLTEN9**使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 14-13 **未实现**：读为0

注 1：仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-29：CFDxFLTCON2：过滤器控制寄存器（ x = 1-4） （续）

bit 12-8 **F9BP<4:0>**：过滤器n命中时指向对象的指针位(1)

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 7 **FLTEN8**：使能过滤器n接收报文位

1 = 使能过滤器
0 = 禁止过滤器

bit 6-5 未实现：读为0

bit 4-0 **F8BP<4:0>**：过滤器n命中时指向对象的指针位(1)

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

注 1： 仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

寄存器56-30：CFDxFLTCON3：过滤器控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN15	—	—	F15BP<4:0>(1)				
23:16	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN4	—	—	F14BP<4:0>(1)				
15:8	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN13	—	—	F13BP<4:0>(1)				
7:0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN12	—	—	F12BP<4:0>(1)				

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **FLTEN15**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 30-29 **未实现**：读为0

bit 28-24 **F15BP<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 23 **FLTEN14**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 22-21 **未实现**：读为0

bit 20-16 **F14P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 15 **FLTEN13**使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 14-13 **未实现**：读为0

注 1：仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-30：CFDxFLTCON3：过滤器控制寄存器（ x = 1-4） （续）

bit 12-8 **F13BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 7 **FLTEN12**：使能过滤器n接收报文位

1 = 使能过滤器
0 = 禁止过滤器

bit 6-5 未实现：读为0

bit 4-0 **F12BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

注 1： 仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

寄存器56-31：CFDxFLTCON4：过滤器控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN19	—	—	F19BP<4:0>(1)				
23:16	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN18	—	—	F18BP<4:0>(1)				
15:8	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN17	—	—	F17BP<4:0>(1)				
7:0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN16	—	—	F16BP<4:0>(1)				

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **FLTEN19**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 30-29 **未实现**：读为0

bit 28-24 **F19BP<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 23 **FLTEN18**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 22-21 **未实现**：读为0

bit 20-16 **F18P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 15 **FLTEN17**使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 14-13 **未实现**：读为0

注 1：仅当相应过滤器处于禁止状态（FLTENx位（CFDxFLTCONn）= 0）时才能修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-31：CFDxFLTCON4：过滤器控制寄存器（ x = 1-4） （续）

bit 12-8 **F17BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 7 **FLTEN16**：使能过滤器n接收报文位

1 = 使能过滤器
0 = 禁止过滤器

bit 6-5 未实现：读为0

bit 4-0 **F16BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

注 1： 仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

寄存器56-32：CFDxFLTCON5：过滤器控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN23	—	—	F23BP<4:0>(1)				
23:16	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN22	—	—	F22BP<4:0>(1)				
15:8	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN21	—	—	F21BP<4:0>(1)				
7:0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN20	—	—	F20BP<4:0>(1)				

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **FLTEN23**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 30-29 **未实现**：读为0

bit 28-24 **F23BP<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 23 **FLTEN22**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 22-21 **未实现**：读为0

bit 20-16 **F22P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 15 **FLTEN21**使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 14-13 **未实现**：读为0

注 1：仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-32：CFDxFLTCON5：过滤器控制寄存器（ x = 1-4） （续）

bit 12-8 **F21BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 7 **FLTEN20**：使能过滤器n接收报文位

1 = 使能过滤器
0 = 禁止过滤器

bit 6-5 未实现：读为0

bit 4-0 **F20BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

注 1： 仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

寄存器56-33：CFDxFLTCON6：过滤器控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN27	—	—	F27BP<4:0>(1)				
23:16	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN26	—	—	F26BP<4:0>(1)				
15:8	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN25	—	—	F25BP<4:0>(1)				
7:0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN24	—	—	F24BP<4:0>(1)				

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **FLTEN27**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 30-29 **未实现**：读为0

bit 28-24 **F27P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 23 **FLTEN26**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 22-21 **未实现**：读为0

bit 20-16 **F25P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 15 **FLTEN25**使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 14-13 **未实现**：读为0

注 1：仅当相应过滤器处于禁止状态（FLTENx位（CFDxFLTCONn）= 0）时才能修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-33：CFDxFLTCON6：过滤器控制寄存器（ x = 1-4） （续）

bit 12-8 **F25BP<4:0>**：过滤器n命中时指向对象的指针位(1)

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 7 **FLTEN24**：使能过滤器n接收报文位

1 = 使能过滤器
0 = 禁止过滤器

bit 6-5 未实现：读为0

bit 4-0 **F24BP<4:0>**：过滤器n命中时指向对象的指针位(1)

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

注 1： 仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

寄存器56-34：CFDxFLTCON7：过滤器控制寄存器（ x = 1-4）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN31	—	—	F31BP<4:0>(1)				
23:16	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN30	—	—	F30BP<4:0>(1)				
15:8	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN29	—	—	F29BP<4:0>(1)				
7:0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	FLTEN28	—	—	F28BP<4:0>(1)				

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **FLTEN31**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 30-29 **未实现**：读为0

bit 28-24 **F31BP<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 23 **FLTEN30**：使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 22-21 **未实现**：读为0

bit 20-16 **F30P<4:0>**：过滤器n命中时指向对象的指针位(1)
11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 15 **FLTEN29**使能过滤器n接收报文位
1 = 使能过滤器
0 = 禁止过滤器

bit 14-13 **未实现**：读为0

注 1：仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

 © 2019 Microchip Technology Inc.

寄存器56-34：CFDxFLTCON7：过滤器控制寄存器（ x = 1-4） （续）

bit 12-8 **F29BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

bit 7 **FLTEN28**：使能过滤器n接收报文位

1 = 使能过滤器
0 = 禁止过滤器

bit 6-5 未实现：读为0

bit 4-0 **F28BP<4:0>**：过滤器n命中时指向对象的指针位[(1)]

11111 = 与过滤器匹配的报文存储在对象31中
11110 = 与过滤器匹配的报文存储在对象30中
.
.
.
00010 = 与过滤器匹配的报文存储在对象2中
00001 = 与过滤器匹配的报文存储在对象1中
00000 = 保留。对象0为发送队列，不能接收报文。

注 1： 仅当相应过滤器处于禁止状态 （FLTENx位 （CFDxFLTCONn） = 0）时才能修改这些位。

寄存器56-35：CFDxFLTOBJn：过滤器对象寄存器（ x = 1-4；n = 0-31）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	EXIDE	SID11	EID<17:13>				
23:16	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	EID<12:5>							
15:8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	EID<4:0>					SID<10:8>		
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	SID<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **未实现**：读为0

bit 30 **EXIDE**：扩展标识符使能位

如果MIDE = 1：

1 = 仅匹配带有扩展标识符地址的报文
0 = 仅匹配带有标准标识符地址的报文

bit 29 **SID11**：标准标识符过滤器位

1 = 已将标准标识符扩展至12位 （即SID<11:0>）
0 = 不使用扩展标准标识符位

注： CAN FD基本帧中的标准标识符过滤器位RRS可用于将SID扩展至12位。使能后， RRS用作SID<11:0>的MSB （称为SID11）。

bit 28-11 **EID<17:0>**：扩展标识符过滤器位

在DeviceNet™模式下，这些过滤器位可与DeviceNet过滤器位编号位 （DNCNT<4:0>（CFDxCON<4:0>））搭配使用。

bit 10-0 **SID<10:0>**：标准标识符过滤器位

这些是标准ID报文过滤器位。

注： 仅当禁止该过滤器 （FLTENx位 （CFDxFLTCONn） = 0）时，才能更改该寄存器。

 © 2019 Microchip Technology Inc.

寄存器56-36： CFDxMASKn：屏蔽寄存器（ x = 1-4；n = 0-31）

位范围	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	MIDE	MSID11	MEID<17:13>				
23:16	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	MEID<12:5>							
15:8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	MEID<4:0>					MSID<10:8>		
7:0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	MSID<7:0>							

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 31 **未实现：**读为0

bit 30 **MIDE：**标识符接收模式位
1 = 仅匹配与过滤器中EXIDE位对应的报文类型 （标准/扩展地址）
0 = 如果过滤器匹配（即，如果（过滤器SID）=（报文SID），或（过滤器SID/EID）=（报文SID/EID）），则与标准或扩展地址报文匹配

bit 29 **MSID11：**标准标识符屏蔽寄存器位
1 = 允许将标准标识符屏蔽寄存器位扩展至12位 （MSID<10:0>和MEID<0>）
0 = 不允许扩展标准标识符屏蔽寄存器 （MSID<10:0>）

bit 28-11 **MEID<17:0>：**扩展标识符屏蔽寄存器位
在DeviceNet™模式下，这些位是用于扩展CAN ID的屏蔽寄存器位，即用作前2个数据字节。

bit 10-0 **MSID<10:0>：**标准标识符屏蔽寄存器位
这些是标准CAN报文标识符屏蔽寄存器位。

注： 仅当禁止该过滤器 （FLTENx位 （CFDxFLTCONn） = 0）时，才能更改该寄存器。

56.4 工作模式

具有灵活数据速率的控制器局域网 （CAN FD）具有八种工作模式：

- 配置模式
- 正常CAN FD模式：支持混合使用CAN FD和CAN 2.0报文
- 正常CAN 2.0模式：在接收CAN FD报文时将生成错误帧。即使在发送报文对象中将FDF位置1，也会强制将该位设为零，并且仅发送CAN 2.0帧。
- 禁止模式
- 仅监听模式
- 受限工作模式
- 内部环回模式
- 外部环回模式

工作模式主要可分为四组：配置、正常、休眠和调试 （见图56-9）。

56.4.1 模式更改

图56-9展示了可能的模式切换。通过写入REQOP<2:0> （CFDxCON<26:24>）位来请求新的工作模式。工作模式不会立即更改。只有在总线空闲时，才能更改模式。

当前的工作模式由OPMOD<2:0> （CFDxCON<23:21>）位指示。应用程序可以允许OPMODx位变化中断或轮询OPMODx位。

56.4.1.1 在正常模式之间切换

不允许在正常模式之间直接切换。必须先选择配置模式，然后才能选择新的正常模式。

56.4.1.2 在调试模式之间切换

不允许在调试模式之间直接切换。必须先选择配置模式，然后才能选择新的调试模式。

56.4.1.3 退出正常模式

只有在当前报文发送后，器件才能切换到配置模式或休眠模式。

56.4.1.4 进入和退出禁止模式

具有灵活数据速率的控制器局域网 （CAN FD）在休眠模式请求后进入禁止模式，在其他模式请求后退出禁止模式。

如果WAKIE置1，则CxRX上的显性边沿将产生中断。CPU必须通过请求正常模式来使能CAN模块。

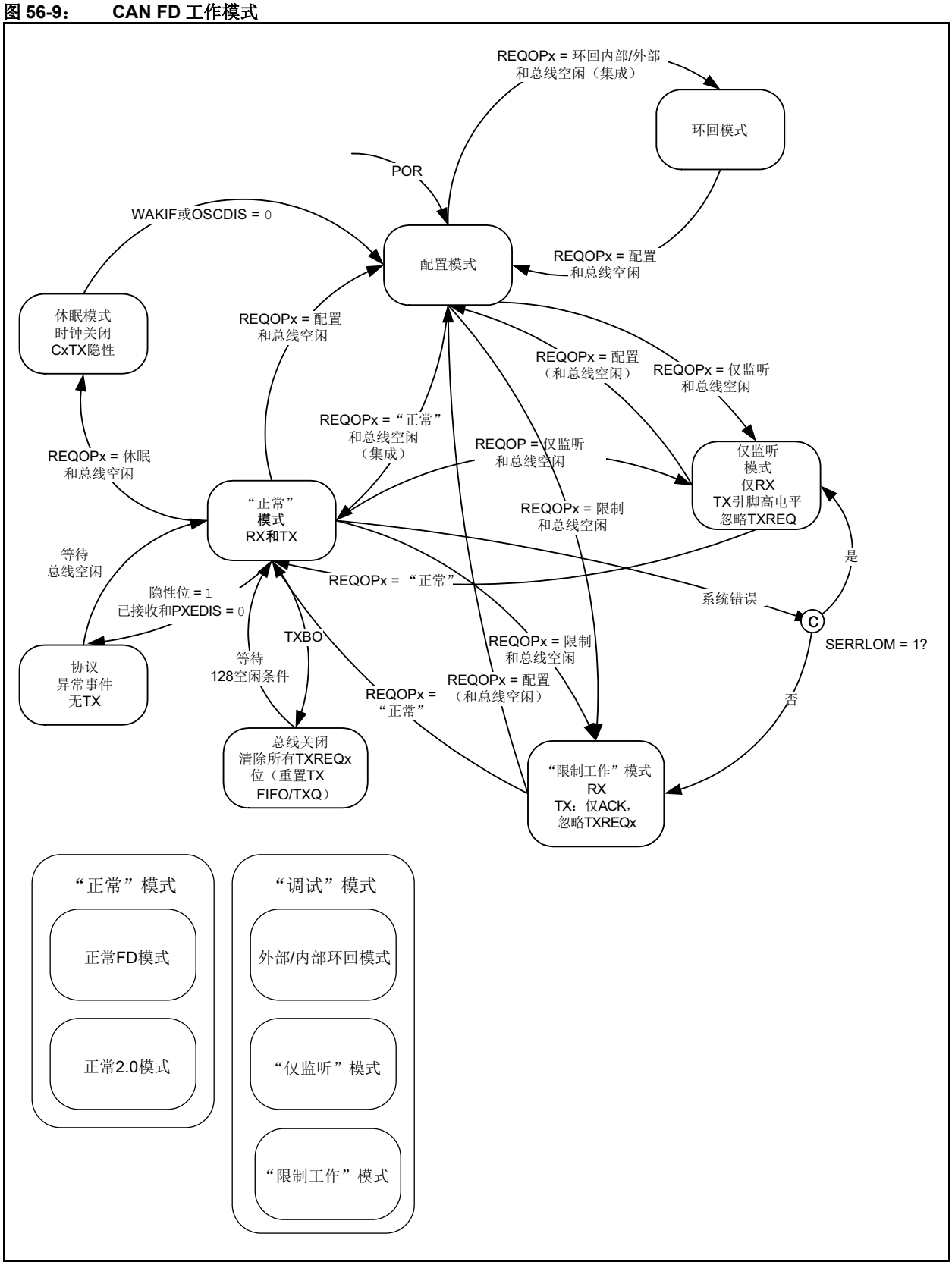
56.4.1.5 总线集成模式

在下列条件下，具有灵活数据速率的控制器局域网 （CAN FD）按照ISO11898-1:2015规范 （11个连续隐性位）集成至总线：

- 从配置模式切换到某一正常模式或调试模式
- 从禁止模式切换到某一正常模式

 © 2019 Microchip Technology Inc.

图 56-9： CAN FD 工作模式



56.4.2 配置模式

复位后，具有灵活数据速率的控制器局域网 （CAN FD）进入配置模式。错误计数器会清零，并且所有寄存器均包含复位值。

具有灵活数据速率的控制器局域网 （CAN FD）激活前必须先进行初始化。仅当模块处于配置模式（OPMOD<2:0> = 100）时才能执行该操作。通过设置REQOP<2:0> = 100来请求配置模式。

具有灵活数据速率的控制器局域网 （CAN FD）可防止用户因编程错误而意外违反CAN协议。只能在配置模式下对下列寄存器和位域进行编程：

- CFDxCON：WAKFIL、CLKSEL、PXEDIS、ISOCRCEN、TXQEN、STEF、SERRLOM、ESIGM和RTXAT
- CFDxNBTCFG、 CFDxDBTCFG和CFDxTDC
- CFDxTXQCON：PLSIZE<2:0>和FSIZE<4:0>
- CFDxFIFOCONn：TXEN、 RXTSEN、 PLSIZE<2:0>和FSIZE<4:0>
- CFDxTEFCON：TEFTSEN和FSIZE<4:0>
- CFDxFIFOBA

在发送或接收期间，不允许具有灵活数据速率的控制器局域网 （CAN FD）进入配置模式，从而避免模块造成CAN总线错误。在退出配置模式时，以下寄存器会复位：

- CFDxTREC
- CFDxBDIAG0
- CFDxBDIAG1

在配置模式下，CFDxFIFOCON、CFDxTXQCON和CFDxTEFCON寄存器中的FRESET置1，并且所有FIFO和TXQ都会复位。

56.4.3 正常模式

56.4.3.1 正常CAN FD模式

配置器件后，可以通过设置REQOP<2:0> = 000来请求正常工作模式。

在该模式下，器件将处于CAN总线上。器件可以在CAN FD模式下发送和接收报文，使能比特率切换，且最多可以发送和接收64个数据字节。

56.4.3.2 正常CAN 2.0模式

可以通过设置REQOP<2:0> = 110来请求正常CAN 2.0工作模式。

在该模式下，器件将处于CAN总线上。该模式为经典CAN 2.0模式。模块将不会接收CAN FD帧。如果在总线上检测到CAN FD帧，则模块可能会发送错误帧。将忽略发送对象中的FDF、 BRS和ESI位并作为0发送。

56.4.4 禁止模式

禁止模式与配置模式类似，只是错误计数器不会复位。通过设置REQOP<2:0> = 001来请求禁止模式。

正在进行发送或接收期间，不允许CAN模块进入禁止模式，以避免造成CAN总线错误。模块将在当前的报文传输完成后进入禁止模式。

OPMODx位用于指示模块是否已成功进入禁止模式。应用程序软件应将该位域用作禁止模式请求的握手指示。

当模块处于禁止模式时， CxTX引脚将保持在隐性状态，以避免意外的CAN总线错误。

 © 2019 Microchip Technology Inc.

56.4.5 调试模式

56.4.5.1 仅监听模式

仅监听模式是正常CAN FD工作模式的一种变形。如果激活仅监听模式，则CAN总线上的模块会进入被动状态。该模块会接收报文，但不会发送任何位。将忽略TXREQx位。不发送错误标志或应答信号。错误计数器在该状态下不可用。仅监听模式可用于检测CAN总线上的波特率。至少还需要有其他两个可相互通信的节点。可根据经验来检测波特率，方法是使用多个不同值进行测试，直至成功接收到报文为止。该模式也有助于监视CAN总线，而不会对其造成影响。

56.4.5.2 受限工作模式

在受限工作模式下，节点能够接收数据帧和远程帧，也能够应答有效帧，但不发送数据帧、远程帧、错误帧或过载帧。出现错误或过载条件时，节点不发送显性位；而是等待总线进入空闲状态，以自行重新同步来与CAN通信。错误计数器不会递增。

56.4.5.3 环回模式

环回模式是正常CAN FD工作模式的一种变形。该模式允许在内部将报文从发送FIFO发送到接收FIFO。模块无需来自总线的外部应答。由于CxRX引脚已断开连接，因此无法接收来自总线的报文。

56.4.5.4 内部环回模式

在内部将发送信号连接到接收端，并将CxTX引脚驱动为高电平。

56.4.5.5 外部环回模式

在外部将发送信号连接到接收端，并可在CxTX引脚上监视发送报文。

56.4.6 低功耗模式

56.4.6.1 休眠模式

在CAN模块中，需要满足特殊条件才能进入休眠模式。必须先设置REQOPx = 001以将模块切换到禁止模式。当OPMODx = 001时，即表示禁止模式已激活， CAN FD协议模块会在休眠模式请求后进入休眠模式。

在休眠模式下，寄存器内容保持不变，因此OPMODx位也保持不变。在休眠结束时，模块将继续处于休眠模式之前OPMODx位所指定的模式下 （应为禁止模式，即OPMODx = 001）。

如果用户没有切换至禁止模式就执行SLEEP指令，则模块会假定时钟可用于从RAM中进行读/写。

由于系统时钟输入在休眠模式下不可用，因此CAN模块无法运行，因为它需要系统时钟才能进行发送或接收。此外， FIFO位于系统RAM中，而系统RAM在休眠模式下没有时钟信号。

建议步骤：

1. 向REQOP<2:0>位写入001；模块将进入禁止模式。
2. 轮询OPMOD<2:0>位以验证其是否为001，该值表示模块已成功进入禁止模式。
3. 执行SLEEP指令。

56.4.6.2 空闲模式

系统可设置为以低功耗模式（称为空闲模式）运行。当器件处于空闲模式时， CPU处于禁止状态，只有选定的外设处于工作状态。

根据CAN SIDL位的配置，模块可以进入或退出空闲模式：

- 如果SIDL = 0，则模块继续在空闲模式下工作。如果模块在空闲模式下产生中断，则中断可能生成唤醒事件。
- 如果SIDL = 1，则模块会在器件处于空闲模式时停止工作。在空闲模式下停止工作的模块会执行与其在禁止模式下相同的步骤，并且遵循相同的要求。

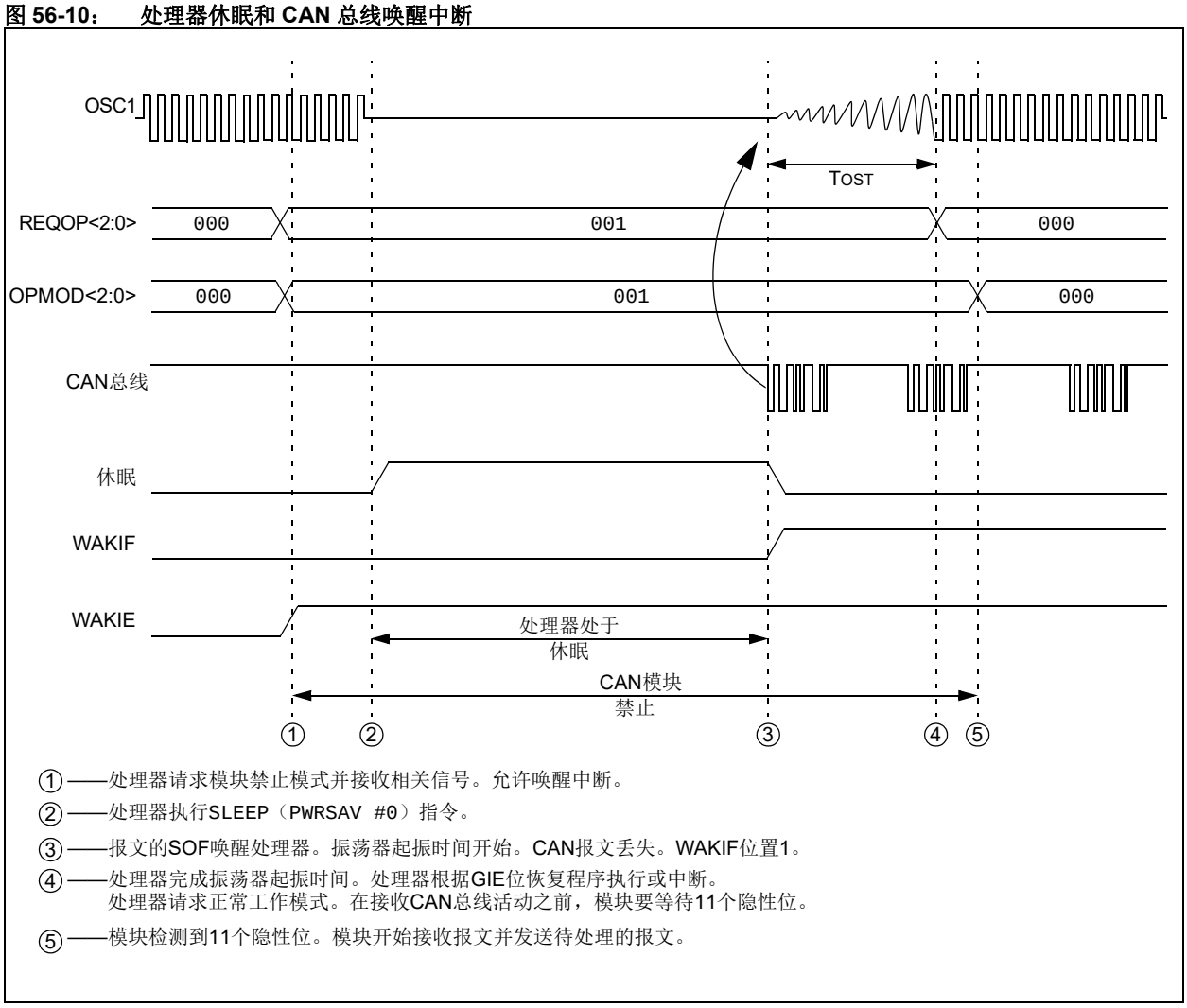
SIDL = 1时，用户必须确保在CPU切换到空闲模式时模块未处于工作状态。为避免CAN总线系统遭受因违反该规则而造成的致命后果，模块在空闲模式下停止工作后，将驱动发送引脚进入隐性状态。

如果CAN SIDL位置1，则建议在器件进入空闲模式之前将模块切换到禁止模式。

56.4.6.3 从休眠模式唤醒

图56-10展示了CAN模块将如何执行SLEEP指令以及模块如何唤醒总线活动。从休眠模式唤醒时， WAKIF标志置1。

图 56-10： 处理器休眠和 CAN 总线唤醒中断



① ——处理器请求模块禁止模式并接收相关信号。允许唤醒中断。

② ——处理器执行SLEEP（PWRSAV #0）指令。

③ ——报文的SOF唤醒处理器。振荡器起振时间开始。CAN报文丢失。WAKIF位置1。

④ ——处理器完成振荡器起振时间。处理器根据GIE位恢复程序执行或中断。
处理器请求正常工作模式。在接收CAN总线活动之前，模块要等待11个隐性位。

⑤ ——模块检测到11个隐性位。模块开始接收报文并发送待处理的报文。

 © 2019 Microchip Technology Inc.

处于休眠模式时，模块将对CAN接收线的活动进行监视。使能WAKIE时，器件将在CxRX的下降沿产生唤醒中断。

器件将在新的模式请求或CxRX的下降沿后退出休眠模式。

以下任一条件为真时，模块将进入休眠模式：

- 系统在休眠模式后进入禁止模式
- 系统处于空闲模式且SIDL = 1

注 1： 当模块处于休眠模式时，如果WAKIE位 （CFDxINT<30>）置1并且检测到总线活动，则模块会产生中断。由于振荡器起振和CPU启动时存在延时，因此导致唤醒的报文活动会丢失。

2: 该模块可编程为在禁止、休眠或空闲模式下对 CAN 接收输入线应用低通滤波器功能。该功能可用于保护模块，避免由于CAN总线线路上的短暂毛刺而导致唤醒。当模块处于休眠模式时，由WAKFIL位 （CFDxCON<8>）使能或禁止该滤波器。

56.5 配置

56.5.1 CAN配置

CFDxCON寄存器包含多个只能在配置模式下配置的位。

56.5.2 ISO CRC使能

模块支持ISO CRC （符合ISO11898-1:2015）和非ISO CRCC （见**第56.2.1节 “ISO CRC与非ISO CRC”**）。将ISOCRCEN位置1可使能ISO CRC。

56.5.3 协议异常禁止

CAN FD帧中FDF位与 “保留位”之间的下降沿对计算收发器延时和硬同步至关重要。因此，如果检测到FDF位后的 “保留位”为隐性，则具有灵活数据速率的控制器局域网 （CAN FD）会将其视为格式错误。这称为 “协议异常事件检测禁止”，可通过将PXEDIS位置1进行配置。

将PXEDIS位清零可以使能协议异常事件检测禁止。发生协议异常事件后，错误计数器保持不变，硬同步进入使能状态，同时模块发送隐性位并进入总线集成状态。

56.5.4 唤醒滤波器——WFT<1:0>

WAKFIL位用于使能/禁止CxRX引脚上的低通滤波器。该滤波器仅在休眠模式下工作。WFTx位允许配置不同的滤波时间。

56.5.5 限制发送尝试

ISO11898-1:2015要求自动重发仲裁失败且未得到应答或因错误损坏的帧。此外，还可以对重发尝试的次数进行限制。

当RTXAT位置1时，可以使用FIFO控制寄存器中的TXAT<1:0>位限制重发尝试。如果RTXAT位清零，则将忽略FIFO控制寄存器中的TXATx位，并且不会限制重发尝试。

56.5.6 网关模式下的错误状态指示（ ESI）

通常，发送报文中的ESI位反映了具有灵活数据速率的控制器局域网（CAN FD）的错误状态。当模块处于被动错误状态时， ESI将作为隐性发送。如果在网关应用中使用该模块，则即使网关模块处于主动错误状态，也会出现报文中的ESI位应作为隐性发送的情况。可通过将ESIGM位置1来进行配置。

56.5.7 发生系统错误时的模式选择

SERRLOM位用于选择在发生系统错误时模块将切换到哪种模式。模块可以切换到受限工作模式或仅监听模式。

56.5.8 为TXQ和TEF保留报文存储器

将TXQEN位置1会为TXQ保留RAM。如果TXQEN位清零，则不能使用TXQ。

将STEF位置1会为TEF保留RAM，所有发送的报文都将存储在TEF中。

56.5.9 CAN FD位时间配置

为了实现更高的带宽， CAN FD帧中的各个位会以两种不同的比特率进行发送：

- 标称比特率 （NBR）：在仲裁期间使用该比特率，直到BRS位的采样点和CRC分隔符的采样点达到EOF
- 数据比特率 （DBR）：在数据和CRC字段中使用该比特率

 © 2019 Microchip Technology Inc.

NBR受CAN网络传播延时的限制 （见第56.5.9.2节 “传播延时”）。在数据阶段，只保留一个发送器；因此，可以提高比特率。发送节点始终会比较预期的发送位与CAN总线上的实际位。数据阶段的传播延时可能比位时间长。在这种情况下，数据位在二次采样点 （SSP）处采样 （见第56.5.9.3节 “发送器延时补偿 （Transmitter Delay Compensation， TDC）”）。

NBR是仲裁阶段的每秒位数。它是标称位时间（Nominal Bit Time，NBT）的倒数（见公式56-1）。

公式56-1： 标称比特率/标称位时间

$$NBR = \frac{1}{NBT}$$

DBR是数据阶段的每秒位数。它是数据位时间 （Data Bit Time， DBT）的倒数 （见公式56-2）。

公式56-2： 数据比特率/数据位时间

$$DBR = \frac{1}{DBT}$$

波特率预分频比（Baud Rate Prescaler，BRP）用于对SYSCLK进行分频。分频后的SYSCLK用于生成位时间。

有两种预分频比：标称比特率预分频比 （Nominal Bit Rate Prescaler， NBRP）和数据比特率预分频比 （Data Bit Rate Prescaler， DBRP）。按照公式56-3和公式56-4所示选择相应时间份额：标称时间份额 （Nominal Time Quanta， NTQ）和数据时间份额 （Data Time Quanta， DTQ）。

公式56-3： 标称时间份额

$$NTQ = NBRP \times T_{SYSCLK} = \frac{NBRP}{F_{SYSCLK}}$$

公式56-4： 数据时间份额

$$DTQ = DBRP \times T_{SYSCLK} = \frac{DBRP}{F_{SYSCLK}}$$

按照ISO11898-1:2015中的规定， CAN位时间具有四个时间段 （见图56-11）。

同步段 （SYNC） ——同步CAN总线上连接的不同节点。预计该时间段内存在位边沿。同步段始终为1个TQ。

传播段 （PRSEG） ——用于补偿总线上的传播延时。 PRSEG必须比最大传播延时长。

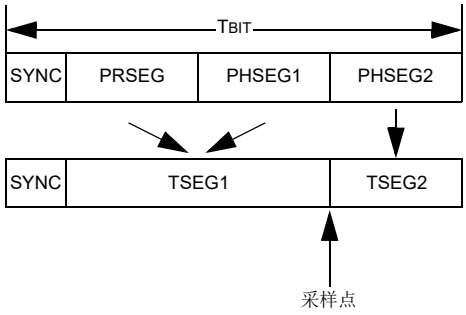
相位段1 （PHSEG1） ——用于补偿因边沿处相移而可能产生的误差。为补偿相移而进行重同步期间，该时间段可自动延长。

相位段2 （PHSEG2） ——用于补偿因边沿处相移而可能产生的误差。为补偿相移而进行重同步期间，该时间段可自动缩短。

在位时间寄存器中，PRSEG与PHSEG1组合构成了TSEG1。PHSEG2称为TSEG2。每段都有多个时间份额 （TQ）。采样点位于TSEG1和TSEG2之间。

表56-3 和表56-4显示了位时间配置参数的范围。

图 56-11： 位时间的分段



位时间中的TQ总数可编程，并且可使用公式56-5和公式56-6进行计算。

公式56-5： NBT中的NTQ数

$$\frac{NBT}{NTQ} = NSYNC + NTSEG1 + NTSEG2$$

公式56-6： DBT中的DTQ数

$$\frac{DBT}{DTQ} = DSYNC + DTSEG1 + DTSEG2$$

表56-3： 标称比特率配置范围

段	最小值	最大值
NSYNC	1	1
NTSEG1	2	256
NTSEG2	1	128
NSJW	1	128
每位NTQ	4	385

表56-4： 数据比特率配置范围

段	最小值	最大值
DSYNC	1	1
DTSEG1	1	32
DTSEG2	1	16
DSJW	1	16
每位DTQ	3	49

56.5.9.1 采样点

采样点是位时间中用于读取和解析位逻辑电平的时间点。可使用公式56-1和公式56-2计算采样点百分比。

公式56-1： 标称采样点 （%）

$$NSP = \frac{1 + NTSEG1}{\frac{NBT}{NTQ}} \times 100$$

 © 2019 Microchip Technology Inc.

公式56-2： 数据采样点 （%）

$$DSP = \frac{1 + DTSEG1}{\frac{DBT}{DTQ}} \times 100$$

56.5.9.2 传播延时

图56-12显示了总线上两个CAN节点之间的传播延时（假设节点A正在发送CAN报文）。发送的位将从发送CAN节点A开始传播，通过发送CAN收发器经由CAN总线发送至接收CAN收发器，然后接收CAN收发器再将其发送至接收CAN节点B。

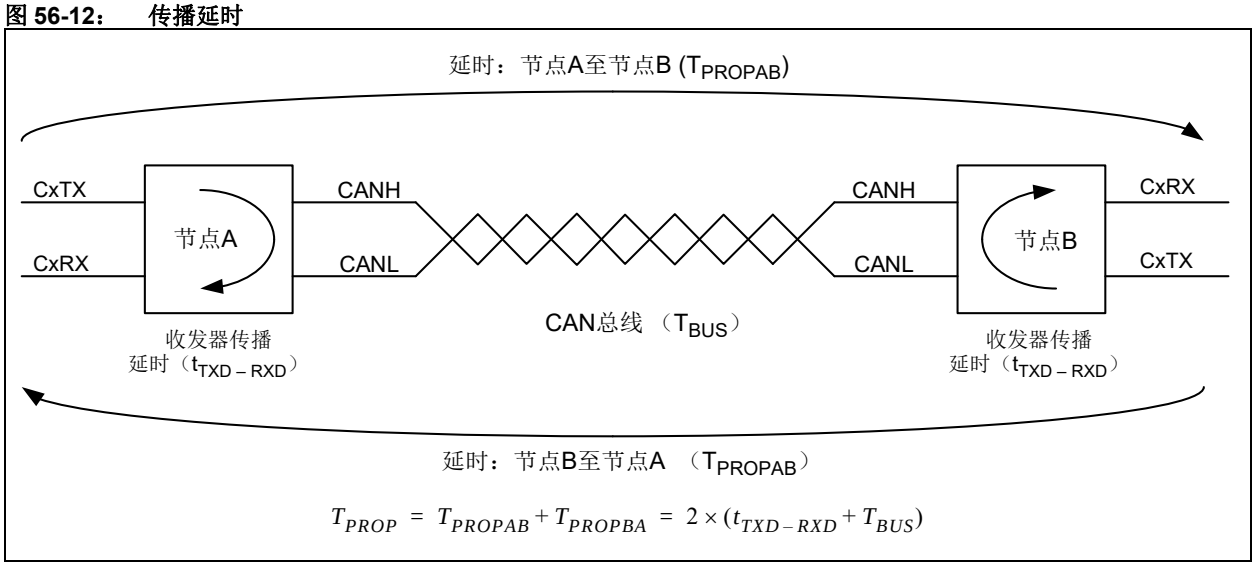
在CAN报文的仲裁阶段，发送器会对CAN总线进行采样并检查发送的位与接收的位是否匹配。发送节点必须将采样点放在最大传播延时之后。

公式56-1说明了最大传播延时；其中$t_{TXD-RXD}$是收发器的传播延时，最大值为255 ns （符合ISO11898-1:2015标准）；T_{BUS}是CAN总线上的延时，约为5 ns/m。因数2来自最糟糕的情况，即节点B刚好在来自节点A的位到达时开始发送。

公式56-1： 最大传播延时

$$T_{PROP} = 2 \times (t_{TXD-RXD} + T_{BUS})$$

图 56-12： 传播延时



56.5.9.3 发送器延时补偿（ TRANSMITTER DELAY COMPENSATION， TDC）

在CAN FD发送的数据阶段中，只有一个节点正在进行发送；而其他节点则正在进行接收。因此，传播延时不限制最大数据速率。

通过CxTX引脚发送时，具有灵活数据速率的控制器局域网 （CAN FD）通过CxRX引脚从其本地的CAN收发器接收发送的数据。接收到的数据会因CAN收发器的环路延时而延迟。如果该延时大于1 + DTSEG1，则会检测到位错误。

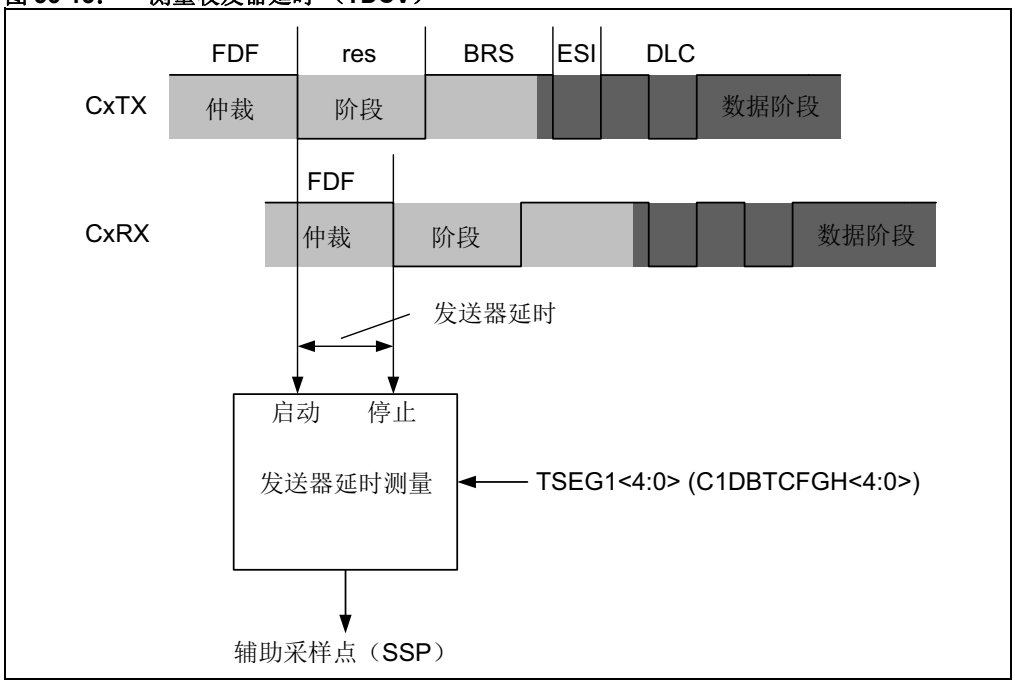
为使数据阶段位时间短于收发器环路延时，采用了发送器延时补偿 （TDC）。这样，只需计算二次采样点 （SSP）并利用此采样点在CAN FD报文的数据阶段进行采样，而无需在DTSEG1之后采样。

图56-13展示了如何测量收发器环路延时，公式56-1演示了如何计算SSP。

公式56-1： 二次采样点

$$SSP = TDCV<5:0> + TDCO<6:0>$$

图 56-13： 测量收发器延时 （TDCV）



56.5.9.4 同步

为补偿CAN总线上节点振荡器频率之间的相移，每个CAN控制器都必须能够与传入信号的相关边沿同步。

CAN控制器希望接收信号中的边沿出现在SYNC段内。只有隐性至显性边沿会用于同步。

有两种用于同步的机制：

- **硬同步**——使已经出现的边沿强制位于SYNC段内。位时间计数器使用SYNC重新启动。
- **重新同步**——如果边沿落在SYNC段外，则调整PHSEG1或PHSEG2。

关于CAN同步的更详细说明，请参见ISO11898-1:2015。

56.5.9.5 同步跳转宽度

同步跳转宽度（Synchronization Jump Width， SJW）是重新同步期间PHSEG1和PHSEG2可以调整的最大量。 SJW可编程 （见表56-3和表56-4）。

56.5.9.6 振荡器容差

振荡器容差*df*约等于振荡器的标称频率*fnom*，在公式56-1中定义。

公式56-2至公式56-6描述了振荡器最大容差的条件。

 © 2019 Microchip Technology Inc.

公式56-1： 振荡器容差

$$(1-df)\times fnom \le F_{SYSCLK} \le (1+df)\times fnom$$

公式56-2： 条件1

$$df \le \frac{NSJW}{2\times 10\times \frac{NBT}{NTQ}}$$

公式56-3： 条件2

$$df \le \frac{min(NPHSEG1,NPHSEG2)}{2\times\left(13\times\frac{NBT}{NTQ}-NPHSEG2\right)}$$

公式56-4： 条件3

$$df \le \frac{DSJW}{2\times 10\times \frac{DBT}{DTQ}}$$

公式56-5： 条件4

$$df \le \frac{min(NPHSEG1,NPHSEG2)}{2\times\left(\left(6\times\frac{DBT}{DTQ}-DPHSEG2\right)\times\frac{DBRP}{NBRP}+7\times\frac{NBT}{NTQ}\right)}$$

公式56-6： 条件5

$$df \le \frac{DSJW-max\left(0,\left(\frac{NBRP}{DBRP}-1\right)\right)}{2\times\left(\left(2\times\frac{NBT}{NTQ}\times HNSEGP2\right)\times\frac{NBRP}{DBRP}+DPHSEG2+4\times\frac{DBT}{DTQ}\right)}$$

56.5.9.7 位时间配置示例

下面的表格展示了CAN FD位时间寄存器的配置，假定汽车中存在具有以下参数的CAN FD网络：

- 500 kbps NBR——采样点在80％处
- 2 Mbps DBR——采样点在80％处
- 40米——最小总线长度

表56-5和表56-6展示了如何计算位时间参数。由于参数取决于多个约束和公式，并且使用迭代过程进行计算，因此建议在电子表格中输入公式。

表56-7可将计算值转换为寄存器值。建议让具有灵活数据速率的控制器局域网 （CAN FD）测量发送器延时补偿值 （Transmitter Delay Compensation Valu， TDCV）。此操作通过设置TDCMOD<1:0>（C1TDCH<1:0>）= 10（自动模式）实现。为了将SSP设为80％，TDCO<6:0>应设为1 + DTSEG1。

表 56-5： 标称位速率逐步配置

参数	约束	值	单位	公式和备注
NBT	NBT ≥ 1 µs	2	µs	公式56-1中所述。
FOSC	FOSC ≤ 80 MHz	80	MHz	FSYSCLK = FOSC/2 = 40 MHz。
NBRP	1至256	1	—	选择尽可能小的BRP值以最大化分辨率。

参数	约束	值	单位	公式和备注
NTQ	NBT和FSYSCLK	12.5	ns	公式56-3中所述。
NBT/NTQ	4至385	160	—	公式56-5中所述。
NSYNC	固定	1	NTQ	在ISO11898-1:2015中定义。
NPRSEG	NPRSEG > T_{PROP}	95	NTQ	公式56-1：T_{PROP} = 910 ns， 最小NPRSEG = T_{PROP}/NTQ = 72.8 NTQ。 选择95将支持最长60m的总线长度。
NTSEG1	2至256 NTQ	127	NTQ	公式56-1。选择NTSEG1可达到80%NSP。
NTSEG2	1至128 NTQ	32	NTQ	剩下32 NTQ以达到NBT/NTQ = 160。
NSJW	1至128 NTQ； SJW ≤ min （NPHSEG1, NPHSEG2）	32	NTQ	最大化NSJW可降低对振荡器容差的要求。

表 56-6： 数据位速率逐步配置

参数	约束	值	单位	公式和备注
DBT	DBT ≥ 125 ns	500	ns	公式56-2中所述。
DBRP	1至256	1	—	选择与NBT相同的预分频比可确保位速率转换期间TQ分辨率保持不变。
DTQ	DBT和FSYSCLK	12.5	ns	公式56-4中所述。
DBT/DTQ	3至49	40	—	公式56-6中所述。
DSYNC	固定	1	DTQ	在ISO11898-1:2015中定义。
DTSEG1	1至32 DTQ	31	DTQ	公式56-1。选择DTSEG1可达到80% DSP。
DTSEG2	1至16 DTQ	8	DTQ	剩下8 DTQ以达到DBT/DTQ = 40。
DSJW	1至16 DTQ； SJW ≤ min （DPHSEG1, DPHSEG2）	8	DTQ	最大化DSJW可降低对 振荡器容差的要求。
振荡器容差 条件1-5	条件1-5的最小值	0.78	%	公式56-1至公式56-6。

表 56-7： 位时间寄存器初始化 （500k/2M）

C1NBTCFGH/L	值	C1DBTCFGH/L	值	C1TDCH/L	值
BRP<7:0>	0	BRP<7:0>	0	TDCMOD<1:0>	2
TSEG1<7:0>	126	TSEG1<4:0>	30	TDCO<6:0>	31
TSEG2<6:0>	31	TSEG2<3:0>	7	TDCV<5:0>	0
SJW<6:0>	31	SJW<3:0>	7	—	—

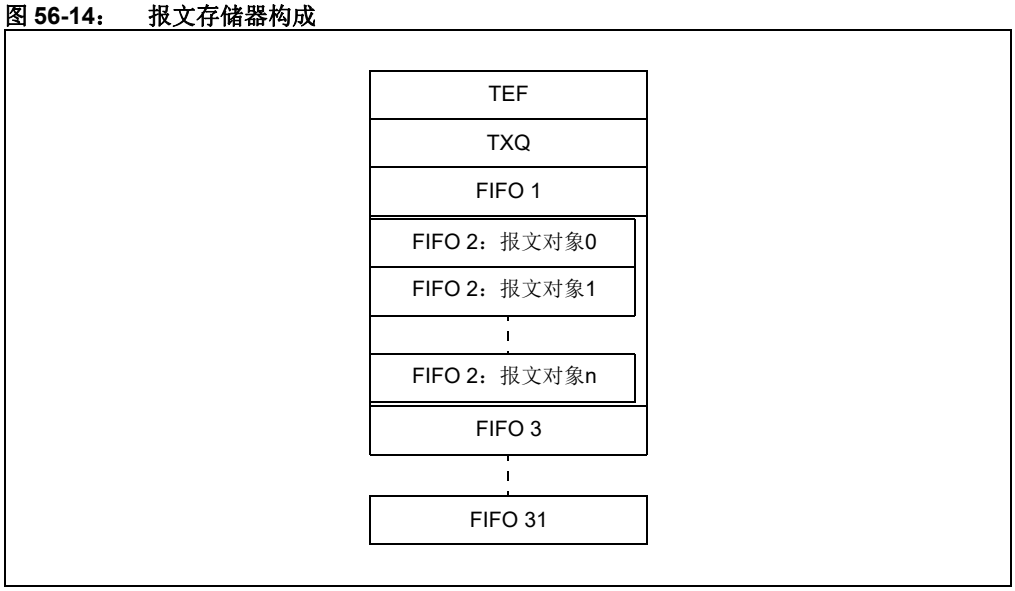
56.5.10 报文存储器配置

TEF、 TXQ和发送/接收FIFO的报文对象位于RAM中 （见图56-14）。应用程序必须在报文对象0和报文对象31之间配置FIFO中的报文对象数量。此外，应用程序还必须在每个FIFO中配置报文对象的有效负载大小。此配置可确定报文对象在RAM中的位置。 RAM分配只能在 “配置”模式下配置。

为了优化RAM的使用，应用程序应使用TEF开始配置RAM，然后是TXQ，再使用FIFO 1、 FIFO 2和FIFO 3等。如果用户应用程序需要TEF、 TXQ和16个额外的FIFO，则应配置TEF和TXQ，然后配置FIFO 1至FIFO 16。无需配置未使用的FIFO 17至31。

 © 2019 Microchip Technology Inc.

图 56-14： 报文存储器构成



56.5.10.1 发送事件FIFO配置

要在RAM中为TEF保留空间，必须将STEF位 （CFDxCON<19>）置1。 TEF中的报文对象数量通过FSIZE<4:0>位（CFDxTEFCON<28:24>）进行配置。通过将TEFTSEN位（CFDxTEFCON<5>）置1，可以为发送报文附加时间戳。

56.5.10.2 发送队列配置

要在RAM中为TXQ保留空间，必须将TXQEN位 （CFDxCON<20>）置1。 TXQ中的报文对象数量通过FSIZE<4:0>位（CFDxTXQCON<28:24>）进行配置。TXQ中的所有对象采用相同的有效负载大小 （数据字节数），该数值通过PLSIZE<2:0>位 （CFDxTXQCON<31:29>）进行配置。

56.5.10.3 发送FIFO配置

通过将CFDxFIFOCONn寄存器中的TXEN置1，可将FIFO 1至FIFO 31配置为发送FIFO。每个发送FIFO中的报文对象数量通过FSIZE<4:0>位 （CFDxFIFOCONn<28:24>）进行配置。一个发送FIFO 中的所有对象采用相同的有效负载大小 （数据字节数），该数值由PLSIZE<2:0> 位（CFDxFIFOCONn<31:29>）确定。

56.5.10.4 接收FIFO配置

通过将CFDxFIFOCONn寄存器中的TXEN清零，可将FIFO 1至FIFO 31配置为接收FIFO。每个接收FIFO中的报文对象数量通过FSIZE<4:0>位 （CFDxFIFOCONn<28:24>）进行配置。一个接收FIFO 中的所有对象采用相同的有效负载大小 （数据字节数），该数值由PLSIZE<2:0> 位（CFDxFIFOCONn<31:29>）确定。通过将RXTSEN位 （CFDxFIFOCONn<5>）置1，可以为接收报文附加时间戳。

56.5.10.5 所需报文存储器的计算

所需RAM的大小取决于每个FIFO的配置。公式56-1至公式56-3以字节为单位指定TEF、 TXQ和FIFO的大小。如果TEF或TXQ的大小为零，则不使用。

由于集成RAM的大小有限，因此用户必须检查存储器配置是否适合RAM。公式56-4可用于计算RAM使用情况 （以字节为单位）。

TEF对象的大小取决于时间戳是否使能。如果TEFTSEN置1，则tefts = 4，否则tefts = 0。

*PayLoad(i)*使用数据字节定义。

RX FIFO的报文对象的大小取决于时间戳是否使能。如果*FIFO(i)*的RXTSEN = 1且TXEN = 0，则*rxts(i)* = 4，否则*rxts(i)* = 0。

*N*被定义为除TEF和TXQ之外使用的FIFO的数量。

公式56-1： TEF的大小

$$S_{TEF} = N_{Elements}(TEF) \times (tefts + 8)$$

公式56-2： TXQ的大小

$$S_{TXQ} = N_{Elements}(TXQ) \times (8 + PayLoad(TXQ))$$

公式56-3： FIFO的大小

$$S_{FIFO}(i) = N_{Elements}(i) \times (rxts(i) + 8 + PayLoad(i))$$

公式56-4： 总RAM使用情况

$$S_{RAM} = \left(S_{TEF} + S_{TXQ} + \sum_{i=1}^{N} S_{FIFO}(i) \right)$$

例如：

- 如果TEF为4个报文深 （$N_{Elements}$ (TEF) = 4）且TEFTSEN清零，
  则TEF的大小为S_{TEF} = 4 x (0 + 8) = 32字节
- 如果$N_{Elements}$ (TXQ) = 1、 PayLoad (TXQ) = 12，
  则TXQ的大小为 S_{TXQ} = 1 x (8 + 12) = 20字节
- 如果$N_{Elements}$ (FIFO) = 3、 PayLoad (FIFO) = 8，
  则FIFO的大小为S_{FIFO} = 3 x (8 + 8) = 48字节

因此， SRAM = S_{TEF} + S_{TXQ} + S_{FIFO} = 32 + 20 + 48 = 100字节。

 © 2019 Microchip Technology Inc.

56.6 报文发送

应用程序必须配置FIFO或TXQ后才能用于发送（见**第56.5.10.3节 “发送FIFO配置”**和**第56.5.10.2节 “发送队列配置”**）。

56.6.1 发送报文对象

表56-8指定了TXQ和发送FIFO使用的发送报文对象。发送对象包含报文ID、控制位和有效负载。

- **SID**：标准标识符或基本标识符。
- **EID**：扩展标识符。
- **DLC**：数据长度代码；指定要发送的数据字节数（见**第56.2.1.1节 “DLC编码”**）。
- **IDE**：标识符扩展；把此位清零将发送基帧，把此位置1将发送扩展帧。
- **RTR**：远程发送请求；此位仅在CAN 2.0帧中指定。把此位置1将请求发送接收节点。
- **FDF**：FD格式；如果此位置1，将发送CAN FD帧；否则，将发送CAN 2.0帧。如果选择了正常CAN 2.0模式，则会忽略此位，仅发送CAN 2.0帧。
- **BRS**：位速率转换；如果此位置1，则会通过DBR发送CAN FD帧的数据相位。如果此位清零，则会通过NBR发送整个帧。
- **ESI**：错误状态指示位；通常， ESI位反映发送节点的错误状态。 CAN FD帧中的隐性ESI位表示发送节点处于被动错误状态，显性位表示发送节点处于主动错误状态。如果ESIGM（CFDxCON<17>）= 0，对象中的此位会被忽略。如果ESIGM = 1，当具有灵活数据速率的控制器局域网 （CAN FD）处于被动错误状态或报文对象中的ESI位置1时，发送报文中的ESI位将隐性发送。网关应用程序将使用它来发送表明发送节点的ESI位已置1的信号。
- **SEQ**：序列号；SEQ不在CAN总线上发送。它用于跟踪发送报文。 SEQ存储在TEF报文对象中。
- **发送缓冲区数据**：包含报文的有效负载。仅发送由DLC指定的数据字节数。首先发送字节0，然后发送1、 2，依此类推。

56.6.2 将报文加载到发送FIFO

将报文加载到FIFO前，应用程序必须确保FIFO未满。如果TFNRFNIF位（CFDxFIFOSTAn<0>）置1，则FIFO中有空间。将报文加载到满FIFO可能会破坏正在发送的报文。

FIFO用户地址指向应用程序应存储报文的下一个发送报文对象的RAM中的地址。 RAM中的实际地址通过公式56-1计算。首先加载发送报文对象的“T0”，然后加载T1、T2，依此类推。最大数据字节数受配置的有效负载限制。只需加载由DLC指定的数据字节数。

公式56-1： 下一个报文对象的地址

$$A = CFDxFIFOBA + CFDxFIFOUAn$$

将报文对象加载到RAM后，需通过将UINC位 （CFDxFIFOCONn<8>）置1来使FIFO递增，此操作会使具有灵活数据速率的控制器局域网 （CAN FD）递增FIFO的头部，然后更新CFDxFIFOUAn。

现在报文已准备好进行发送，下一条报文可以在新地址加载。

56.6.3 将报文加载到发送队列

将发送报文对象加载到TXQ的工作原理与将报文对象加载到发送FIFO类似。应用程序必须检查CFDxTXQSTA 寄存器，查看 TXQ 中是否有空间。应使用 CFDxTXQUA 寄存器（而不是CFDxFIFOUAn寄存器）来计算用于加载报文的地址，并通过将UINC位（CFDxTXQCON<8>）置1来使TXQ的头部递增。

表 56-8： 发送报文对象（TXQ 和 TX FIFO）

字	位	Bit 15/7	Bit 14/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0
T0	15:8	EID<4:0>					SID<10:8>		
	7:0	SID<7:0>							
T1	15:8	—	—	SID11	EID<17:13>				
	7:0	EID<12:5>							
T2	15:8	SEQ<6:0>							ESI
	7:0	FDF	BRS	RTR	IDE	DLC<3:0>			
T3	15:8	—	—	—	—	—	—	—	—
	7:0	—	—	—	—	—	—	—	—
T4(1)	15:8	发送数据字节1							
	7:0	发送数据字节0							
T5(1)	15:8	发送数据字节3							
	7:0	发送数据字节2							
T6	15:8	发送数据字节5							
	7:0	发送数据字节4							
T7	15:8	发送数据字节7							
	7:0	发送数据字节6							
Ti-1	15:8	发送数据字节n-3							
	7:0	发送数据字节n-2							
Ti	15:8	发送数据字节n							
	7:0	发送数据字节n-1							

bit 15:11 （T0） **EID<4:0>**：扩展标识符位

bit 10-0 （T0） **SID<10:0>**：标准标识符位

bit 15-14 （T1） **未实现**：读为x

bit 13 （T1） **SID11**：在FD模式下，标准ID可使用r1扩展到12位

bit 12-0 （T1） **EID<17:5>**：扩展标识符位

bit 15-9 （T2） **SEQ<6:0>**：用于在发送事件FIFO中跟踪发送报文的序列

bit 8 （T2） **ESI**：错误状态指示位

在CAN到CAN网关模式（ESIGM（CFDxCON<17>）= 1）下，发送的ESI标志是ESI（T1）和CAN控制器的被动错误状态两者的 “逻辑或”。

在正常模式下， ESI指示错误状态：
1 = 发送节点处于被动错误状态
0 = 发送节点处于主动错误状态

bit 7 （T2） **FDF**：FD帧；区分CAN和CAN FD格式

bit 6 （T2） **BRS**：位速率转换；选择是否转换数据位速率

bit 5 （T2） **RTR**：远程发送请求；不用于CAN FD

bit 4 （T2） **IDE**：标识符扩展标志；区分基本格式和扩展格式

bit 3-0 （T2） **DLC<3:0>**：数据长度代码

bit 15:0 （T3） **未实现**：读为x

注 1： 数据字节0-n：有效负载大小在PLSIZE<2:0>位（CFDxFIFOCONn<31:29>）中单独配置。

 © 2019 Microchip Technology Inc.

56.6.4 请求发送发送FIFO中的报文

将报文加载到发送FIFO后，该报文就可以用于发送。应用程序通过将TXREQ位（CFDxFIFOCON<9>）置1或将C1TXREQH/L寄存器中的对应位置1来启动FIFO中所有报文的发送。发送完所有报文后， TXREQ将清零。应用程序可以同时请求发送多个FIFO和TXQ。具有最高优先级的FIFO或TXQ将首先开始发送。 FIFO中的报文将以先进先出的方式发送。

当FIFO正在发送报文时也可以将报文加载到FIFO中。由于FIFO清空后TXREQ会被FIFO自动清零，因此必须在附加报文后将CFDxFIFOCONn寄存器的UINC和TXREQ同时置1。这可确保发送FIFO中的所有报文，包括附加的报文。

56.6.5 请求发送发送队列中的报文

将报文加载到TXQ后，该报文就可以用于发送。应用程序通过将TXREQ （CFDxTXQCON<9>）置1来启动队列中所有报文的发送。发送完所有报文后， TXREQ将清零。应用程序可以同时请求发送TXQ和多个FIFO。CFDxTXQCON寄存器中具有最高优先级的TXQ或FIFO将首先开始发送。TXQ中的报文将根据其ID进行发送。具有最高优先级ID和最低ID值的报文将先发送。

当TXQ发送报文时也可以将报文加载到TXQ中。由于TXQ清空后TXREQ会被TXQ自动清零，因此必须在附加报文后将CFDxTXQCON寄存器的UINC和TXREQ同时置1。这可确保发送TXQ中的所有报文，包括附加的报文。

56.6.6 C1TXREQ寄存器

CFDxTXREQ寄存器包含TXQ和所有TX FIFO的TXREQ<31:0>位。它们具有以下用途：

- 用户应用程序使用一条SPI指令即可请求发送TXQ和/或一个或多个TX FIFO，方法是将CFDxTXREQ寄存器中的对应位置1。将位清零不会中止任何发送过程。
- 读取CFDxTXREQ寄存器可以提供与哪些发送FIFO具有待发送报文相关的信息。

CFDxTXREQ<0>映射到TXQ，CFDxTXREQ<1>映射到TX FIFO 1，CFDxTXREQ<2>映射到TX FIFO 2，依此类推。 CFDxTXREQ<31>映射到TX FIFO 31。

56.6.7 发送优先级

需要通过TXPRIx位 （CFDxFIFOCONn<20:16>和CFDxTXQCON<20:16>）配置FIFO和TXQ的发送优先级。

发送报文前，会比较排队等待发送的TXQ和TX FIFO的优先级。首先会发送具有最高优先级的FIFO/TXQ。例如，如果发送FIFO 1的优先级高于FIFO 3，则会先发送FIFO 1中的所有报文。如果多个FIFO具有相同的优先级，则会发送具有最高索引的FIFO。例如，如果FIFO 1和FIFO 3具有相同的优先级设置，则会先发送FIFO 3中的所有报文。如果TXQ和一个或多个FIFO具有相同的优先级，则会先发送TXQ中的所有报文。

每次成功发送单条报文后，将重新计算发送优先级。

56.6.7.1 FIFO中报文的发送优先级

在此方法中， FIFO中的报文以先进先出的方式发送。

56.6.7.2 TXQ中报文的发送优先级

TXQ中的报文根据报文ID发送。首先发送具有最低报文ID （最高优先级）的报文。

56.6.7.3 基于ID的发送优先级

根据ID发送CAN报文的目的是避免 “内部优先级反转” 。如果低优先级报文因总线通信量 （仲裁）而等待发送，则更高优先级的报文可能无法发送。TXQ通过根据优先级（ID）重新确定队列中报文的优先级来解决此问题。

56.6.8 发送带宽共享

带宽共享功能的工作原理如下：

- 成功发送报文后，在尝试发送下一条报文前模块将保持空闲状态，持续*n*个仲裁位时间；它会暂停下一次发送。
- 器件收到报文后，只要总线空闲，模块就可以发送下一条报文。

这使得总线上的其他节点可以发送它们的报文，即使它们的优先级更低。

两次发送之间的仲裁位时间数值可以通过TXBWS<3:0>位 （CFDxCON<31:28>）进行配置。

56.6.9 重发尝试

重发尝试的次数可配置为以下几个选项：

- 禁止重发尝试
- 3次重发尝试
- 无限次重发

可以通过将RTXAT （CFDxCON<16>）置1限制重发尝试。可以通过TXAT<1:0>（CFDxFIFOCONn<22:21>和CFDxTXQCON<22:21>）为每个发送FIFO和TXQ单独配置重发尝试次数。

如果RTXAT = 0，所有发送FIFO将采用无限次重发尝试，并且将忽略TXQ和TXATx。

56.6.9.1 禁止重发尝试

尝试发送报文后，TXREQ将清零。如果因仲裁丢失或错误而没有成功发送报文，则CFDxFIFOSTAn或CFDxTXQSTA寄存器中的TXATIF将置1。

56.6.9.2 3次重发尝试

如果在发送期间检测到错误，具有灵活数据速率的控制器局域网 （CAN FD）将减少剩余尝试次数，并在下次总线空闲时尝试重发该报文。如果仲裁丢失，剩余尝试次数不会改变。如果所有重发尝试次数都已用完， TXREQ将清零， CFDxFIFOSTAn或CFDxTXQSTA中的TXATIF将置1。

在重发报文前，将重新计算发送优先级。如果选择不同的TX FIFO或TXQ进行发送，或者在最后一次发送尝试后收到了报文，则重发尝试将重新初始化。

56.6.9.3 无限次重发

只有在成功发送TX FIFO或TXQ中的所有报文后， TXREQ才会清零。

 © 2019 Microchip Technology Inc.

56.6.10 中止发送

待处理发送只能在开始发送报文前、在帧起始 （SOF）前中止。

特定FIFO的发送可以通过将发送队列控制寄存器中的TXREQ清零中止；将CFDxTXREQ寄存器中的位清零无法中止。向CFDxTXREQ寄存器中的某个位写入“0”将被忽略。成功中止后，FIFO状态寄存器中的TXABT位将置1。 TXREQ将保持置1的状态，直到报文中止或成功发送。

把ABAT（CFDxCON<27>）置1将中止所有FIFO的所有待处理报文。在所有TXREQx位清零后，必须将ABAT清零才能发送新报文。

将发送FIFO的TXREQ清零会尝试中止FIFO中的所有发送。如果报文成功发送，FIFO索引将正常更新。如果报文成功中止，则FIFO索引不会更改。

然后用户可以通过FIFO报文索引位FIFOCI<4:0>（CFDxFIFOSTAn<12:8>）来识别发送的报文。要重置发送FIFO索引并擦除所有待处理报文，用户可以将FRESET置1。然后FIFO便可加载要发送的新报文。

56.6.11 远程发送请求——RTR

CAN总线系统提供一种让主节点从另一个节点请求数据的方法。主器件在RTR位置1的情况下发送报文。此报文没有数据，仅包含触发过滤器匹配的地址。

只有CAN 2.0帧会指定远程帧；CAN FD帧不支持远程帧。

配置为响应远程发送请求的过滤器将指向配置为用于发送的FIFO，并且必须将RTREN置1。

无需MCU干预即可处理自动远程数据请求。如果FIFO已正确配置，当过滤器匹配并指向FIFO时， FIFO将排队等待发送。

FIFO必须如下所示进行配置：

- 将TXEN置 “1”。
- 过滤器必须由匹配的报文标识符使能和加载
- 该过滤器的缓冲区指针必须指向TX FIFO。（通常，过滤器指向RX FIFO。）
- 要使能RTR，必须将RTREN位置 “1”。
- FIFO必须预先加载至少一条要发送的报文。

当接收到RTR报文并且报文与指向正确配置的发送FIFO的过滤器匹配时，TXREQ位置1，同时根据优先级对用于发送的对象进行排队。

只有TXEN和RTREN置1且FIFO不为空时才会发送FIFO。如果在FIFO为空时发生远程发送请求，此事件将被当做溢出处理， RXOVIF位将置1。

56.6.12 发送过程中DLC与有效负载大小不匹配

PLSIZEx位在发送FIFO中留有一定数量的字节。具有灵活数据速率的控制器局域网 （CAN FD）按照下面的方式处理DLC和有效负载大小之间的不匹配：

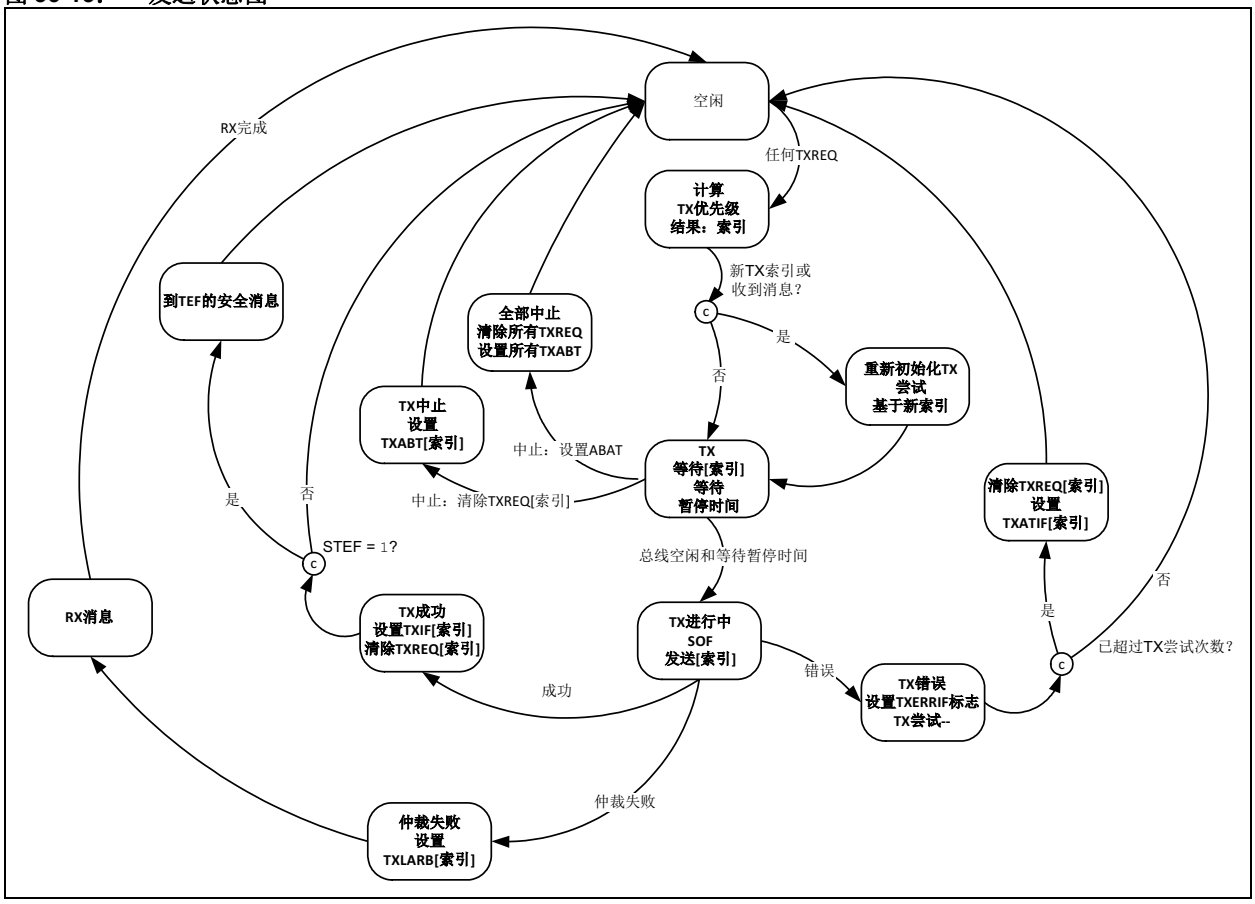
- 如果DLC小于保留的有效负载，将发送由DLC指定的数据字节数。
- 如果DLC大于保留的有效负载，模块将不发送报文。相反，它将把IVMIF （CFDxINT<15>）和DLCMM （CFDxBDIAG1<31>）标志置1并把TXREQ标志清零。应用程序可以使用TEF识别未发送的报文。

56.6.13 发送状态图

图56-15描述了报文如何排队等待发送。它展示了最重要的发送标志是如何置1和清零的：

1. 通过将TXREQ标志置1让报文排队等待发送。
2. 将确定发送优先级。将选择具有最高优先级TXPRIx标志的FIFO或TXQ。将计算FIFO或TXQ中的TX报文的索引。
3. TX报文正在等待发送。
4. 发送只能在总线空闲时开始。
5. 待处理发送只能在发送SOF前中止。
6. 在发送报文期间，具有灵活数据速率的控制器局域网 （CAN FD）会检查以下内容：
   a) 仲裁字段中仲裁丢失。
   b) 发送错误。
7. TX FIFO或TXQ的报文成功发送时，只有在FIFO的所有报文发送完后才会将TXREQ清零。发送任何报文后， FIFO或TXQ的状态标志会更新。如果STEF位 （CFDxCON<19>）置1，报文将存储到TEF中，如果时间戳使能，将附加时间戳。
8. 如果仲裁丢失， TX FIFO或TXQ的TXLARB将置1，器件将切换以接收报文 （见**第56.9节 “报文接收”**）。
9. 如果在发送报文期间检测到错误，将发送错误帧并将相应的错误标志置 1。报文将根据**第56.6.9节 “重发尝试”**重发。

图 56-15： 发送状态图



 © 2019 Microchip Technology Inc.

56.6.14 复位发送FIFO

可以使用以下任一选项复位发送FIFO：

- 将FRESET位 （CFDxFIFOCONn<10>）置1
- 将模块置于 “配置”模式 （OPMOD<2:0> = 100）

复位FIFO会将头和尾指针以及CFDxFIFOSTAn寄存器复位。CFDxFIFOCONn寄存器中的设置保持不变。

使用FRESET复位TX FIFO前，请确保没有待处理的发送。

56.6.15 复位发送队列

可以使用以下任一选项复位发送队列：

- 将FRESET位 （CFDxTXQCON<10>）置1
- 将模块置于 “配置”模式 （OPMOD<2:0> = 100）

复位TXQ会将头和尾指针以及CFDxTXQSTA寄存器复位。 CFDxTXQCON寄存器中的设置保持不变。

使用FRESET复位TXQ前，请确保没有待处理的发送。

56.6.16 报文发送代码示例

例56-1： 报文发送代码

```
#include <xc.h>
/* This code example demonstrates a method to configure the CAN FD module to
transmit Standard and Extended ID CAN FD messages. This uses CAN1, TXQ and
FIFO1. TXQ size is 1 and FIFO1 size is 2. */

/* Include fuse configuration code here. */

#define   MAX_WORDS 100
unsigned  int __attribute__((aligned(4)))CanTxBuffer[MAX_WORDS];

/*Data structure to implement a CANFD  message buffer. */
/* CANFD Message Time Stamp */
typedef   unsigned long CANFD_MSG_TIMESTAMP;

/* CAN TX Message Object Control*/
typedef   struct   _CANFD_TX_MSGOBJ_CTRL {
    unsigned DLC:4;
    unsigned IDE:1;
    unsigned RTR:1;
    unsigned BRS:1;
    unsigned FDF:1;
    unsigned ESI:1;
    unsigned SEQ:7;
    unsigned unimplemented1:16;
} CANFD_TX_MSGOBJ_CTRL;

/* CANFD TX Message ID*/
typedef   struct   _CANFD_MSGOBJ_ID {
    unsigned SID:11;
    unsigned long EID:18;
    unsigned SID11:1;
    unsigned unimplemented1:2;
} CANFD_MSGOBJ_ID;
```

未经测试的代码——仅供参考

例56-1：　报文发送代码示例 （续）

```
/* CAN TX Message Object*/
typedef   union    _CANFD_TX_MSGOBJ {
    struct  {
        CANFD_MSGOBJ_ID id;
        CANFD_TX_MSGOBJ_CTRL ctrl;
        CANFD_MSG_TIMESTAMP timeStamp;
    } bF;
    unsigned int  word[4];
    unsigned char byte[8];
} CANFD_TX_MSGOBJ;

int       main(void)
{
unsigned  char index;

/* The PIC32 device features I/O remap. This I/O remap configuration for the
CAN FD module can be performed here. */
SetIORemapForECANModule();

/* Enable the CANFD module */
CFD1CONbits.ON = 1;

/* Place CAN module in configuration mode */
CFD1CONbits.REQOP = 4;
while(C1CONHbits.OPMOD != 4);

/* Initialize the CFDxFIFOBA with the start address of the CAN FIFO message
buffer area. */
CFD1FIFOBA = (unsigned int) &CanTxBuffer;

/* Set up the CANFD module for 1Mbps of Nominal bit rate speed and 2Mbps of
Data bit rate. */
CFD1NBTCFG = 0x003E0F0F;
CFD1DBTCFG = 0x001E0707;
CFD1TDC = 0x00021F00;                //TDCMOD is Auto

/* Configure CANFD module to enable Transmit Queue and BRS*/
CFD1CONbits.BRSDIS = 0x0;
CFD1CONbits.STEF = 0x0;              //Don't save transmitted messages in
TEF
CFD1CONbits.TXQEN = 0x1;

/* Configure TXQ to transmit 1 message*/
CFD1TXQCONbits.FSIZE  = 0x0;         // single message
CFD1TXQCONbits.PLSIZE = 0x7;         // 64 bytes of data

/* Configure FIFO1 to transmit 2 messages*/
CFD1FIFOCON1bits.FSIZE = 0x1;        //2 messages
CFD1FIFOCON1bits.PLSIZE = 0x2;       //16 bytes of data
CFD1FIFOCON1bits.TXEN = 0x1;         // Set TXEN bit ,transmit fifo
/* Place the CAN module in Normal mode. */
CFD1CONbits.REQOP = 0;
while(CFD1CONbits.OPMOD != 0);
```

未经测试的代码——仅供参考

© 2019 Microchip Technology Inc.

例56-1： 报文发送代码示例 （续）

```
/* Get the address of the message buffer to write to. Load the buffer and then
set the UINC bit. Set the TXREQ bit next to send the message. */

CANFD_TX_MSGOBJ *txObj;

/* Transmit message from TXQ - CANFD base frame with BRS*/

/* SID = 0x100, 64 bytes of data */
txObj  = (CANFD_TX_MSGOBJ *)CFD1TXQUA;
txObj->bF.id.SID = 0x100;
txObj->bF.id.EID = 0x0000;
txObj->bF.ctrl.BRS = 1 ;              //Switch bit rate
txObj->bF.ctrl.DLC = 0xF;             //64 bytes
txObj->bF.ctrl.FDF = 1;               // CANFD frame
txObj->bF.ctrl.IDE = 0;               //Standard frame

for (index=0;index<0x40;index++ )
{
    txObj->byte[index+8] = 0x5A ;     // 64 bytes of 0x5A
}
CFD1TXQCONbits.UINC = 1;              // Set UINC bit
CFD1TXQCONbits.TXREQ = 1;             // Set TXREQ bit

/* Transmit message 0 from FIFO 1 - CANFD base frame with BRS*/

/* SID = 0x300 , 16 bytes of data */
txObj = (CANFD_TX_MSGOBJ *)C1FIFOUA1L;
txObj->bF.id.SID = 0x300;
txObj->bF.id.EID = 0x0000;
txObj->bF.ctrl.BRS = 1 ;              //Switch bit rate
txObj->bF.ctrl.DLC = 0xA;             //16 bytes
txObj->bF.ctrl.FDF = 1;               // CANFD frame
txObj->bF.ctrl.IDE = 0;               //Standard frame

for (index=0;index<0x10;index++ )
{
    txObj->byte[index+8] = 0xA5 ;     // 16 bytes of 0xA5
}
CFD1FIFOCON1bits.UINC  = 1;           // Set UINC bit
CFD1FIFOCON1bits.TXREQ = 1;           // Set TXREQ bit

/* Transmit message 1 from FIFO 1 - CANFD base frame with BRS*/

/* SID = 0x500, EID = 0xC000, 12 bytes of data */
txObj = (CANFD_TX_MSGOBJ *)C1FIFOUA1L;
txObj->bF.id.SID = 0x500;
txObj->bF.id.EID = 0xC000;
txObj->bF.ctrl.BRS = 1 ;              //Switch bit rate
txObj->bF.ctrl.DLC = 0x9;             //12 bytes
txObj->bF.ctrl.FDF = 1;               // CANFD frame
txObj->bF.ctrl.IDE = 0;               //Standard frame

for (index=0;index<0xC;index++ )
{
    txObj->byte[index+8] = 0x55 ;     // 12 bytes of 0x55
}
CFD1FIFOCON1bits.UINC  = 1;           // Set UINC bit
CFD1FIFOCON1bits.TXREQ = 1;           // Set TXREQ bit

while(1);
}
```

未经测试的代码，仅供参考

56.7 发送事件FIFO —— TEF

TEF使应用程序能够跟踪报文发送的顺序和时间。TEF的工作原理与接收FIFO类似。它不存储接收的报文，而是存储发送的报文。STEF位（CFDxCON<19>）置1时才会保存报文。发送报文的序列号（SEQ）会复制到TEF对象中。有效负载数据不会被存储。如果TEFTSEN置1，发送报文会附加时间戳。

表56-9指定了TEF对象。 TEF对象的前两个字是发送报文对象的副本。发送报文时，可选择性地让TEF对象包含时间戳。

56.7.1 读取TEF对象

读取TEF对象前，应用程序必须通过读取CFDxTEFSTA寄存器来检查TEF是否为空。如果TEFNEIF置1，则TEF不为空。

TEF用户地址指向要读取的下一个TEF对象的RAM中的地址。RAM中的实际地址通过公式56-1计算。首先读取TEF对象的TE0，然后读取TE1和TE2。

公式56-1： TEF对象的起始地址

$$A = BaseAddress = C1FIFOBA$$

从RAM读取TEF对象后，需通过将UINC （CFDxTEFCON<8>）置1来使TEF递增。这会使具有灵活数据速率的控制器局域网 （CAN FD）递增尾部并更新CFDxTEFUA。

现在可以从TEF读取下一条报文。

56.7.1.1 复位TEF

可以使用以下任一选项复位TEF：

- 将FRESET位 （CFDxTEFCON<10>）置1
- 将模块置于 “配置”模式 （OPMOD<2:0> = 100）

复位FIFO会将头和尾指针以及CFDxTEFSTA寄存器复位。 CFDxTEFCON寄存器中的设置保持不变。

 © 2019 Microchip Technology Inc.

表 56-9： 发送事件 FIFO 对象

字	位	Bit 15/7	Bit 14/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0
TE0	15:8	EID<4:0>					SID<10:8>		
	7:0	SID<7:0>							
TE1	15:8	—	—	SID11	EID<17:6>				
	7:0	EID<12:5>							
TE2	15:8	SEQ<6:0>							ESI
	7:0	FDF	BRS	RTR	IDE	DLC<3:0>			
TE3	15:8	—	—	—	—	—	—	—	—
	7:0	—	—	—	—	—	—	—	—
TE4(1)	15:8	TXMSGTS<15:8>							
	7:0	TXMSGTS<7:0>							
TE5(1)	15:8	TXMSGTS<31:24>							
	7:0	TXMSGTS<23:16>							

bit 15:11 （TE0） **EID<4:0>**：扩展标识符

bit 10-0 （TE0） **SID<10:0>**：标准标识符

bit 15-14 （TE1） **未实现**：读为x

bit 13 （TE1） **SID11**：在FD模式下，标准ID可使用r1扩展到12位

bit 12-0 （TE1） **EID<13:0>**：扩展标识符

bit 15-9 （TE2） **SEQ<6:0>**：用于在发送事件FIFO中跟踪发送报文的序列

bit 8 （TE2） **ESI**：错误状态指示位

在CAN到CAN网关模式下 （CFDxCON.ESIGM = 1），发送的ESI标志是T1.ESI和CAN控制器的被动错误状态两者的 “逻辑或”

在正常模式下， ESI指示错误状态

1 = 发送节点处于被动错误状态

0 = 发送节点处于主动错误状态

bit 7 （TE2） **FDF**：FD帧；区分CAN和CAN FD格式

bit 6 （TE2） **BRS**：位速率转换；选择是否转换数据位速率

bit 5 （TE2） **RTR**：远程发送请求；不用于CAN FD

bit 4 （TE2） **IDE**：标识符扩展标志；区分基本格式和扩展格式

bit 3-0 （TE2） **DLC<3:0>**：数据长度代码

bit TE3.15-0 **未实现**：读为x

bit TE4.15:0 **TXMSGTS<15:0>**：发送报文时间戳

bit TE5.31-24 **TXMSGTS<15:0>**：发送报文时间戳

注 1： TE4和TE5 （TXMSGTS）仅在CFDxTEFCON.TEFTSEN置1的对象中退出。

56.7.2 发送事件FIFO代码示例

通过TEF保存发送报文的代码示例如例56-2所示。

例56-2： 使用发送事件FIFO代码示例

```
#include <xc.h>

/* This code example demonstrates a method to configure the CAN FD
module to save the transmitted messages in the TEF. This example uses
CAN1, FIFO1 and TEF */


/* Include fuse configuration code here. */
```

未经测试的代码示例 仅供参考

 © 2019 Microchip Technology Inc.

使用发送事件FIFO代码示例——续

```
#defineMAX_WORDS 100
unsigned    int    __attribute__((aligned(4)))CanTxBuffer[MAX_WORDS];
//message buffer to be written
unsigned int * currentMessageBuffer; // Points to message buffer to
be read


/*data structure to implement a CANFD  message buffer. */
/* CANFD Message Time Stamp */
typedefunsignedlong CANFD_MSG_TIMESTAMP;
/* CAN TX Message Object Control*/
typedefstruct_CANFD_TX_MSGOBJ_CTRL {
    unsigned DLC:4;
    unsigned IDE:1;
    unsigned RTR:1;
    unsigned BRS:1;
    unsigned FDF:1;
    unsigned ESI:1;
    unsigned SEQ:7;
    unsigned unimplemented1:16;
} CANFD_TX_MSGOBJ_CTRL;
/* CANFD TX Message ID*/
typedefstruct_CANFD_MSGOBJ_ID {
    unsigned SID:11;
    unsigned long EID:18;
    unsigned SID11:1;
    unsigned unimplemented1:2;
} CANFD_MSGOBJ_ID;
/* CAN TX Message Object*/
typedefunion_CANFD_TX_MSGOBJ {
    struct  {
        CANFD_MSGOBJ_ID id;
        CANFD_TX_MSGOBJ_CTRL ctrl;
        CANFD_MSG_TIMESTAMP timeStamp;
    } bF;
    unsigned int word[4];
    unsigned char byte[8];
} CANFD_TX_MSGOBJ;
```

未经测试的代码——仅供参考

使用发送事件FIFO代码示例——续

```
/* CANFD TEF Message Object */
typedef union _CAN_TEF_MSGOBJ {
    struct  {
        CANFD_MSGOBJ_ID id;
        CANFD_TX_MSGOBJ_CTRL ctrl;
        CANFD_MSG_TIMESTAMP timeStamp;
    } bF;
    unsigned int word[4];
} CANFD_TEF_MSGOBJ;

intmain(void)
{
unsigned char index,fifoSize;

/* The PIC32 device features I/O remap. This I/O remap configuration
for the CAN FD
module can be performed here. */
SetIORemapForECANModule();

/* Enable the CANFD module */
CFD1CONbits.CON = 1;

/* Place CAN module in configuration mode */
CFD1CONbits.REQOP = 4;
while(CFD1CONbits.OPMOD != 4);
/* Initialize the C1FIFOBA with the start address of the CAN FIFO
message buffer area. */
CFD1FIFOBA = (unsigned int) &CanTxBuffer;

/* Set up the CANFD module for 1Mbps of Nominal bit rate speed and
2Mbps of Data bit rate. */
```

未经测试的代码——仅供参考

 © 2019 Microchip Technology Inc.

使用发送事件FIFO代码示例——续

```
/* Set up the CANFD module for 1Mbps of Nominal bit rate speed and
2Mbps of Data bit rate. */
CFD1NBTCFG = 0x003E0F0F;
CFD1DBTCFG = 0x001E0707;
CFD1TDC = 0x00021F00;//TDCMOD is Auto

/* Configure CANFD module to save transmitted messages in TEF and BRS*/
CFD1CONbits.BRSDIS = 0x0;
CFD1CONbits.STEF = 0x1;
CFD1CONbits.TXQEN = 0x0;  // Disable TXQ

/* Configure TEF to save 5 messages*/
CFD1TEFCONbits.FSIZE = 0x4; // save 5 messages
CFD1TEFCONbits.TEFTSEN = 1;

/* Configure FIFO1 to transmit 5 messages*/
CFD1FIFOCON1bits.FSIZE = 0x4;//5 messages
CFD1FIFOCON1bits.PLSIZE = 0x7; //64 bytes of data
CFD1FIFOCON1bits.TXEN = 0x1;// Set TXEN bit ,transmit fifo

/* Place the CAN module in Normal mode. */
CFD1CONbits.REQOP = 0;
while(CFD1CONbits.OPMOD != 0);

/* Get the address of the message buffer to write to. Load the buffer
and */
/* then set the UINC bit. Set the TXREQ bit to send the message. */

CANFD_TX_MSGOBJ *txObj;

/* Transmit 5 messages from FIFO 1 - CANFD base frame with BRS*/
/* SID = 0x300 , 64 bytes of data */
```

未经测试的代码——仅供参考

使用发送事件FIFO代码示例——续

```
for (fifoSize= 0; fifoSize < 5; fifoSize++)
{
txObj = (CANFD_TX_MSGOBJ *)CFD1FIFOUA1;
txObj->bF.id.SID = 0x300;
txObj->bF.id.EID = 0x0000;
txObj->bF.ctrl.BRS = 1 ;//Switch bit rate
txObj->bF.ctrl.DLC = 0xF;//64 bytes
txObj->bF.ctrl.FDF = 1;// CANFD frame
txObj->bF.ctrl.IDE = 0;//Standard frame
txObj->bF.ctrl.SEQ = fifoSize ; // Sequence does not get transmitted,
but stored in TEF

for (index=0;index<0x40;index++ )
{
   txObj->byte[index+8] = 0xA5 ; // 64 bytes of 0xA5
}
CFD1FIFOCON1bits.UINC = 1; // Set UINC bit
}

CFD1FIFOCON1bits.TXREQ = 1;// Set TXREQ bit
while (CFD1FIFOCON1bits.TXREQ == 1);
/* Keep reading the TEF objects until the last transmitted message*/
for (fifoSize= 0; fifoSize < 5; fifoSize++)
{
while(CFD1TEFSTAbits.TEFNEIF ==0);
CANFD_TEF_MSGOBJ *tefObj;
tefObj = (CANFD_TEF_MSGOBJ *)CFD1TEFUA;
//ProcessTEFMessages (currentMessageBuffer) ;
CFD1TEFCONbits.UINC = 1 ;  // Set UINC bit
}
while(1);
}
```

未经测试的代码——仅供参考

 © 2019 Microchip Technology Inc.

56.8 报文过滤

CAN网络上的所有报文将被所有节点接收。为了能够只处理感兴趣的报文，采用了硬件过滤机制。具有灵活数据速率的控制器局域网 （CAN FD）可配置为只接收感兴趣的报文。模块包含32个接收过滤器。每个接收过滤器都包含一个过滤对象和屏蔽对象。用户应用程序会配置特定过滤器以接收具有给定标识符的报文，方法是将过滤对象和屏蔽对象设置为与要接收的报文的标识符相匹配。

56.8.1 过滤器配置

过滤器由CFDxFLTCONn寄存器控制。更改过滤或屏蔽对象前，必须通过将FLTEN位清零来禁止过滤器。模块不需要处于 “配置”模式。更新过滤对象后，必须初始化缓冲区指针FnBP，过滤器可通过将FLTEN位置1使能。 FnBP指向需要存储匹配接收报文的FIFO。

56.8.2 过滤接收报文

具有灵活数据速率的控制器局域网 （CAN FD）在接收仲裁字段和报文的前三个数据字节后开始接收过滤。图56-16描述了报文过滤的流程。

模块在所有过滤器之间循环，从具有最高优先级的过滤器0开始。接收报文组合缓冲区（RXMAB）中的报文与过滤器和屏蔽器进行比较。如果报文与过滤器匹配，并且接收没有任何错误，则报文将被存储到FnBP指向的RX FIFO。接收过滤停止，相关RFIF置1。

如果接收到RTR， FnBP指向的TX FIFO的TXREQ位将置1。

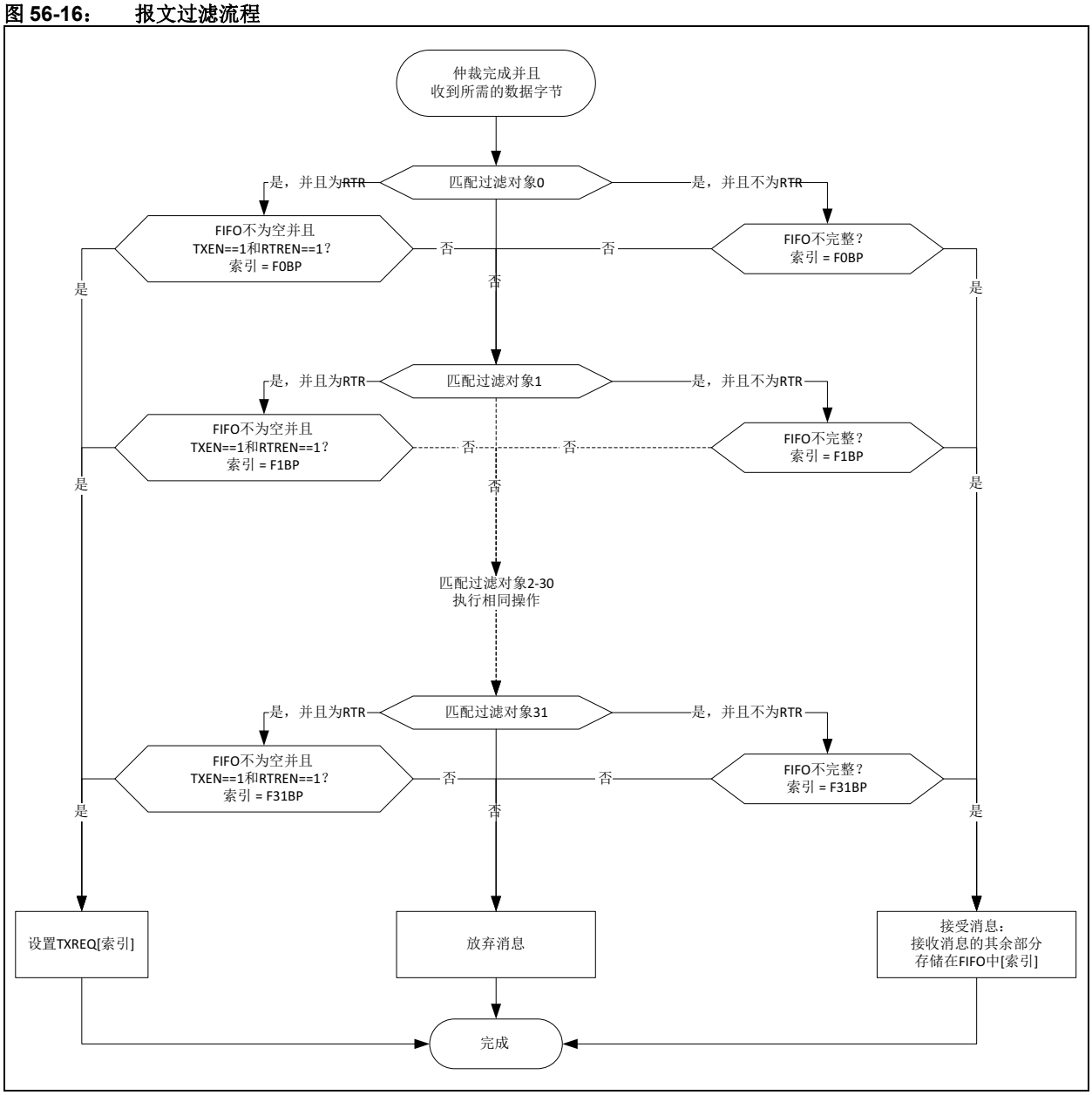
过滤将持续到下一个过滤器，只有在发生以下任一情况时RXOVIF才会置1：

- 过滤器匹配，但RX FIFO已满。
- 当多个过滤器匹配同一报文且所有匹配的RX FIFO都已满时，只会将具有最高优先级的过滤器指向的FIFO的RXOVIF置1。
- 如果在RTR期间TX FIFO为空 （TXEN = 1， RTREN = 1）， RXOVIF将置1。

如果没有任何过滤器匹配，将丢弃接收报文。

注：	如果模块接收到与过滤器匹配的报文，但对应FIFO为TX FIFO（TXEN = 1，RTREN = 0），则模块将丢弃接收报文。

图 56-16： 报文过滤流程



56.8.2.1 过滤标准或扩展帧

图56-17展示了将单个过滤对象与RXMAB中的接收报文进行匹配的流程。

过滤对象可配置为接收标准帧和/或扩展帧。如果MIDE清零，则将接收标准帧和扩展帧。

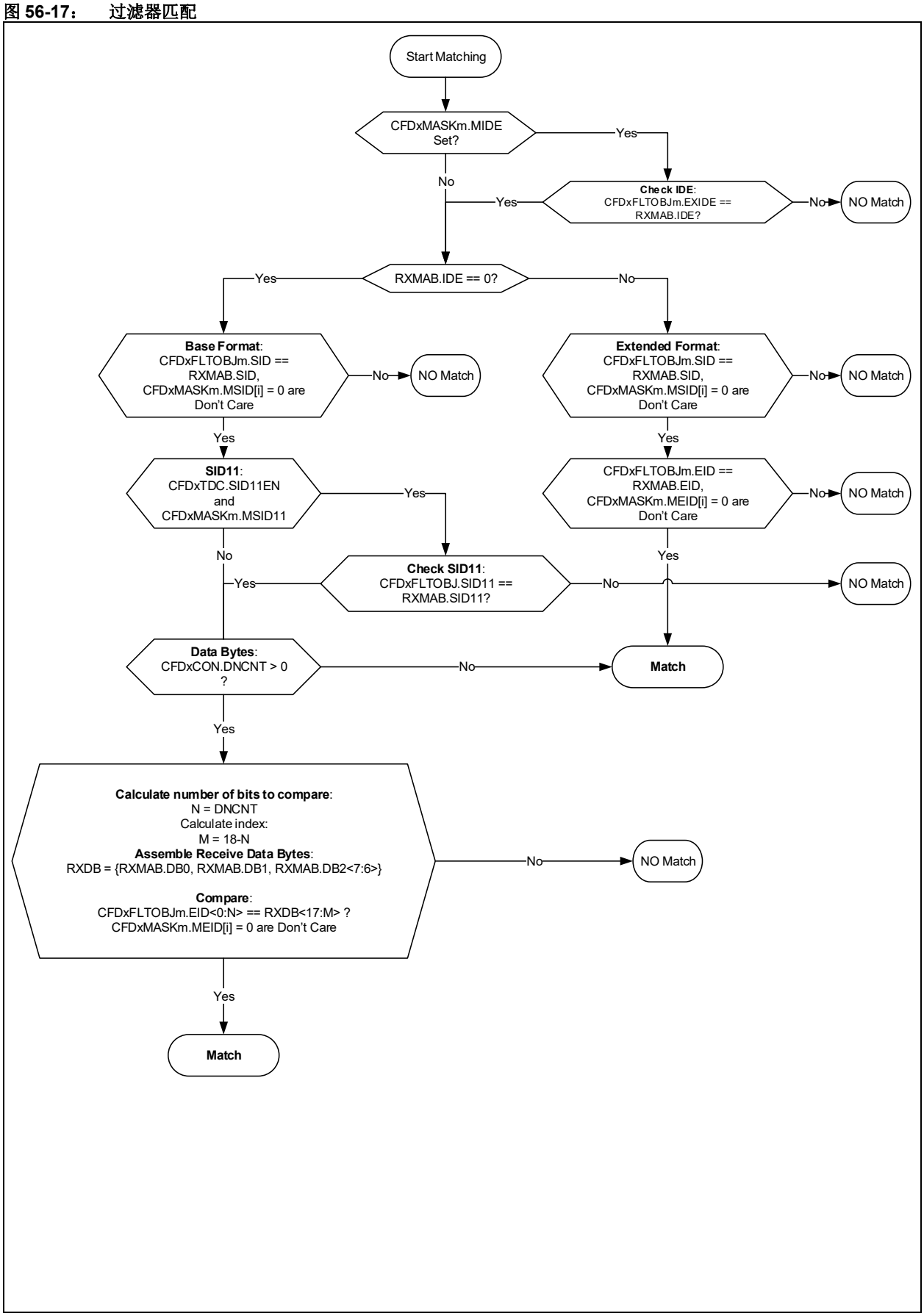
如果过滤器应该只接收标准帧，则必须将MIDE置1、将EXIDE清零。如果过滤器应该只接收扩展帧，则必须将MIDE和EXIDE都置1。

56.8.2.2 屏蔽位

屏蔽对象用于忽略接收标识符的选定位。 RXMAB的屏蔽位 （值为 “0”的屏蔽位）将不会与过滤对象中的位进行比较。例如，要接收所有具有标识符0、1、2和3的报文，需要通过将屏蔽对象的对应位清零来屏蔽标识符较低的两位。

 © 2019 Microchip Technology Inc.

图 56-17： 过滤器匹配


Start Matching
CFDxMASKm.MIDE Set?
Yes
No
Check IDE: CFDxFLTOBJm.EXIDE == RXMAB.IDE?
Yes
No
NO Match
RXMAB.IDE == 0?
Yes
No
Base Format: CFDxFLTOBJm.SID == RXMAB.SID, CFDxMASKm.MSID[i] = 0 are Don't Care
No
NO Match
Extended Format: CFDxFLTOBJm.SID == RXMAB.SID, CFDxMASKm.MSID[i] = 0 are Don't Care
No
NO Match
Yes
Yes
SID11: CFDxTDC.SID11EN and CFDxMASKm.MSID11
Yes
CFDxFLTOBJm.EID == RXMAB.EID, CFDxMASKm.MEID[i] = 0 are Don't Care
No
NO Match
No
Yes
Check SID11: CFDxFLTOBJ.SID11 == RXMAB.SID11?
No
NO Match
Yes
Data Bytes: CFDxCON.DNCNT > 0 ?
No
Match
Yes
Calculate number of bits to compare:
N = DNCNT
Calculate index:
M = 18-N
Assemble Receive Data Bytes:
RXDB = {RXMAB.DB0, RXMAB.DB1, RXMAB.DB2<7:6>}
Compare:
CFDxFLTOBJm.EID<0:N> == RXDB<17:M> ?
CFDxMASKm.MEID[i] = 0 are Don't Care
No
NO Match
Yes
Match

56.8.2.3 数据字节过滤

当过滤器配置为接收标准帧时，可以选择过滤和屏蔽对象的EID部分来进行数据字节过滤。CFDxCON中的DNCNT用于选择数据字节中用于比较的位数。表56-10说明了比较多少数据位，以及比较哪些过滤器位和数据位。

如果DNCNT是：

- “0”，则禁止数据字节过滤。
- 非零， DNCNT中指定的所有数据位都将开始过滤。选中过滤器将需要匹配SID位，并且使*n*个数据位与过滤器EID<0:17>位相匹配。数据字节0<7>总是与EID<0>比较，数据字节0<6>总是与EID<1>比较，数据字节2<6>总是与EID<17>比较。
- 大于18，表示用户选择的位数大于EID位总数，过滤器比较将在数据的第18位终止。
- 大于16，且接收报文的DLC = 2，表示有效负载为两个数据字节，过滤器比较将在数据的第16位终止。
- 大于8，且接收报文的DLC = 1，表示有效负载为一个数据字节，过滤器比较将在数据的第8位终止。
- 大于0，且接收报文的DLC = 0，表示没有数据有效负载，过滤器比较将在标识符处终止。

56.8.2.4 12位标准ID

将CFDxTDC.SID11EN置1可将RRS用作SID（LSB）的位12。12位SID模式仅适用于CAN FD基帧。过滤器通过SID11和MSID11扩展。数据字节也可以在此模式下过滤。

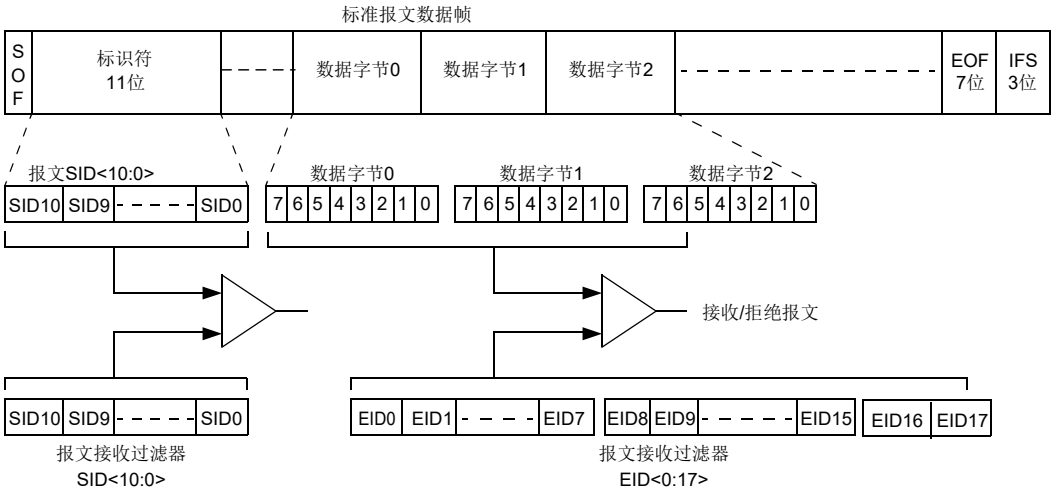
表56-10： 数据字节过滤器配置

DNCNT<4:0>	要进行比较的接收报文数据位 （字节<位>）	用于接收过滤器的EID位
00000	不比较	不比较
00001	数据字节0<7>	EID<0>
00010	数据字节0<7:6>	EID<0:1>
00011	数据字节0<7:5>	EID<0:2>
00100	数据字节0<7:4>	EID<0:3>
00101	数据字节0<7:3>	EID<0:4>
00110	数据字节0<7:2>	EID<0:5>
00111	数据字节0<7:1>	EID<0:6>
01000	数据字节0<7:0>	EID<0:7>
01001	数据字节0<7:0>和数据字节1<7>	EID<0:8>
01010	数据字节0<7:0>和数据字节1<7:6>	EID<0:9>
01011	数据字节0<7:0>和数据字节1<7:5>	EID<0:10>
01100	数据字节0<7:0>和数据字节1<7:4>	EID<0:11>
01101	数据字节0<7:0>和数据字节1<7:3>	EID<0:12>
01110	数据字节0<7:0>和数据字节1<7:2>	EID<0:13>
01111	数据字节0<7:0>和数据字节1<7:1>	EID<0:14>
10000	数据字节0<7:0>和数据字节1<7:0>	EID<0:15>
10001	字节0<7:0>、字节1<7:0>和字节2<7>	EID<0:16>
10010至11111	字节0<7:0>、字节1<7:0>和字节2<7:6>	EID<0:17>

 © 2019 Microchip Technology Inc.

图56-18展示了接收报文数据有效负载的前18个数据位与报文接收过滤器（CFDxFLTOBJn.EID）的对应EID位是如何进行比较的。接收报文的IDE位必须为“0”。

图 56-18： 使用 DeviceNet™ 过滤的 CAN 操作



注： EID位所显示的DeviceNet™过滤配置为DNCNT<4:0> = 10010。

56.9 报文接收

应用程序必须配置RX FIFO后才能用于接收，请参见第56.5.10.4节“接收FIFO配置”。此外，应用程序还必须配置并使能至少一个过滤器，请参见第56.8.1节 “过滤器配置”。

具有灵活数据速率的控制器局域网 （CAN FD）会持续不断地监视CAN总线。与过滤器匹配的报文存储在过滤器指向的RX FIFO中，请参见第56.8.2节 “过滤接收报文”。报文数据存储在接收报文对象中。

56.9.1 接收报文对象

表56-11指定RX FIFO使用的接收报文对象。接收对象包含报文ID、控制位、有效负载和时间戳：

- **SID**：标准标识符 （ID）或基本ID。
- **EID**：扩展ID。
- **DLC**：数据长度代码；指定帧中的数据字节数 （见第56.2.1.1节 “DLC编码”）。
- **IDE**：ID扩展；IDE = 0表示接收到基本标识符帧， IDE = 1表示接收到扩展标识符帧。
- **RTR**：远程发送请求；此位仅在CAN 2.0帧中指定。如果此位置1，会请求模块对帧发送进行响应。
- **FDF**：FD格式；如果此位置1，会接收CAN FD帧；否则，接收CAN 2.0帧。
- **BRS**：位速率转换；如果此位置1，则会通过DBR接收CAN FD帧的数据相位。如果此位清零，则会通过NBR接收整个帧。
- **ESI**：错误状态指示位；ESI位反映发送节点的错误状态。 CAN FD帧中的隐性ESI位表示发送节点处于被动错误状态，显性位表示发送节点处于主动错误状态。
- **FILHIT**：表示与接收报文匹配的过滤器的编号。
- **RXMSGTS**：接收报文的时间戳。可以通过CFDxFIFOCONn.RXTSEN分别为每个RX FIFO使能时间戳。如果时间戳禁止，则接收报文对象将不包含RXMSGTS。
- **接收缓冲区数据**：包含报文的有效负载。最大有效负载在CFDxFIFOCONn.PLSIZE中配置。

56.9.1.1 读取接收报文对象

读取接收报文对象前，应用程序必须通过读取CFDxFIFOSTAn寄存器来确保RX FIFO不为空。如果TFNRFNIF置1，则RX FIFO不为空。

RX FIFO用户地址指向要读取的下一个接收报文对象的RAM中的地址。 RAM中的实际地址通过公式56-1计算。首先读取接收报文对象的R0，然后读取R1、 R2，依此类推。

公式56-1： 下一个报文对象的地址

$$A = CFDxFIFOBA + CFDxFIFOUAn$$

从RAM读取接收报文对象后，需通过将CFDxFIFOCON.UINC置1来使RX FIFO递增。此操作会使具有灵活数据速率的控制器局域网 （CAN FD）递增FIFO的尾部，然后更新CFDxFIFOUAn。

现在，应用程序可以从RX FIFO读取下一条报文。

 © 2019 Microchip Technology Inc.

表 56-11： 接收报文对象

字		Bit 15/7	Bit 14/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0
R0	15:8	EID<4:0>					SID<10:8>		
	7:0	SID<7:0>							
R1	15:8	—	—	SID11	EID<17:6>				
	7:0	EID<12:5>							
R2	15:8	FILHIT<4:0>					—	—	ESI
	7:0	FDF	BRS	RTR	IDE	DLC<3:0>			
R3	15:8	—	—	—	—	—	—	—	—
	7:0	—	—	—	—	—	—	—	—
R4 (2)	15:8	RXMSGTS<15:8>							
	7:0	RXMSGTS<7:0>							
R5 (2)	15:8	RXMSGTS<31:24>							
	7:0	RXMSGTS<23:16>							
R6 (1)	15:8	接收数据字节1							
	7:0	接收数据字节0							
R7 (1)	15:8	接收数据字节3							
	7:0	接收数据字节2							
R8	15:8	接收数据字节5							
	7:0	接收数据字节4							
R9	15:8	接收数据字节7							
	7:0	接收数据字节6							
Ri-1	15:8	接收数据字节n-2							
	7:0	接收数据字节n-3							
Ri	15:8	接收数据字节n							
	7:0	接收数据字节n-1							

bit R0. 15:11 **EID<4:0>**：扩展标识符

bit R0.10-0 **SID<10:0>**：标准标识符

bit R1.15-14 **未实现**：读为x

bit R1.13 **SID11**：在FD模式下，标准ID可使用r1扩展到12位

bit R1.12-0 **EID<13:0>**：扩展标识符

bit R2.15-9 **SEQ<6:0>**：用于在发送事件FIFO中跟踪发送报文的序列

bit R2.8 **ESI**：错误状态指示位

在CAN到CAN网关模式（CFDxCON.ESIGM = 1）下，发送的ESI标志是T1.ESI和CAN控制器的被动错误状态两者的 “逻辑或”；

在正常模式下， ESI指示错误状态

1 = 发送节点处于被动错误状态

0 = 发送节点处于主动错误状态

bit R2.7 **FDF**：FD帧；区分CAN和CAN FD格式

bit R2.6 **BRS**：位速率转换；选择是否转换数据位速率

bit R2.5 **RTR**：远程发送请求；不用于CAN FD

bit R2.4 **IDE**：标识符扩展标志；区分基本格式和扩展格式

bit R2.3-0 **DLC<3:0>**：数据长度代码

bit R3 15:11 **EID<4:0>**：扩展标识符

bit R3.15:0 **未实现**：读为x

bit R4.15:0 **RXMSGTS<31:0>**：接收报文时间戳

bit R5.15:0 **RXMSGTS<31:0>**：接收报文时间戳

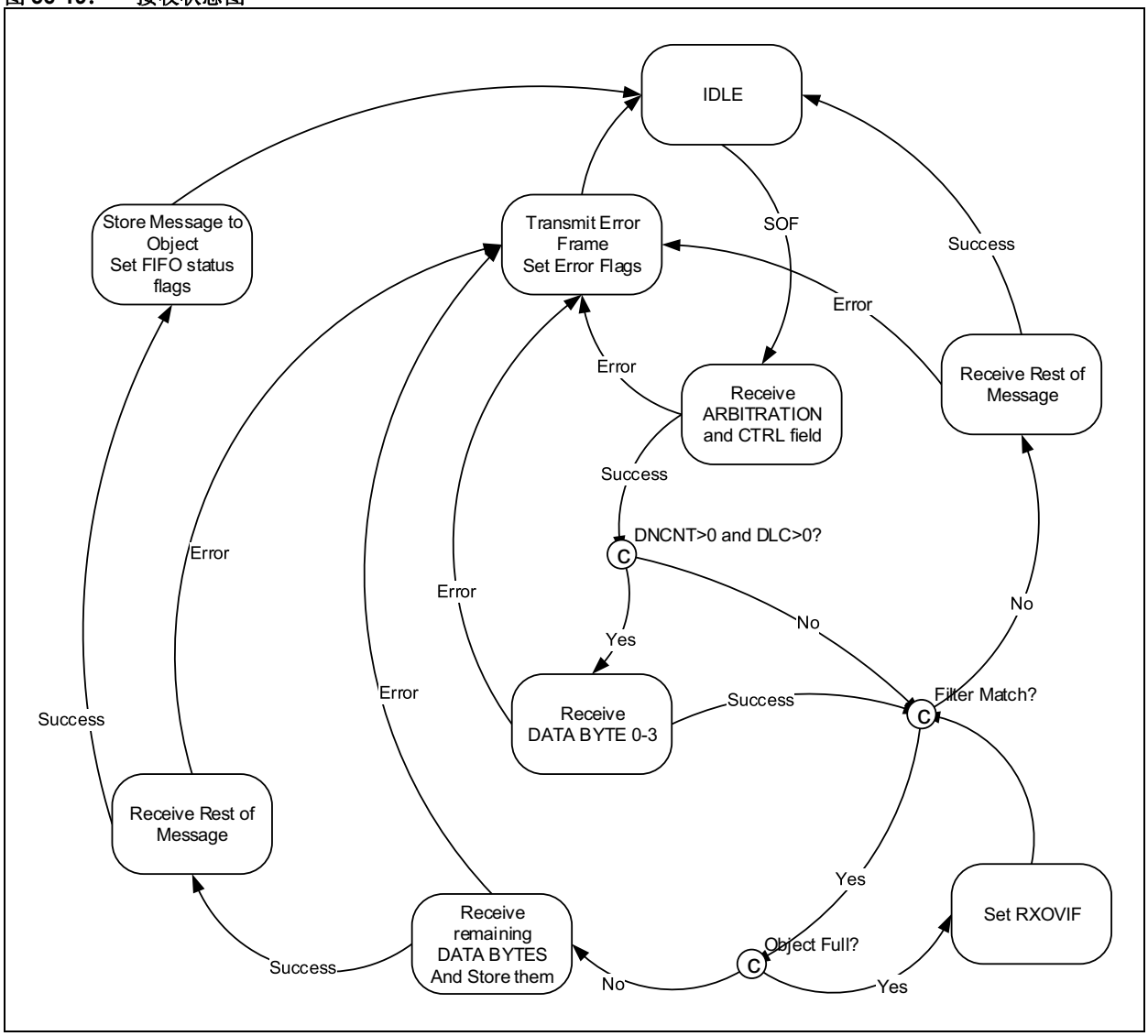
注 **1**：RXMOBJ：数据字节0-n：有效负载大小在FIFO控制寄存器 （CFDxFIFOCONn..PLSIZE<2:0>）中单独配置。
2：R2 （RXMSGTS）仅在CFDxFIFOCONn.RXTSEN置1的对象中退出。

56.9.2 接收状态图

图56-19展示了报文如何接收。它说明了最重要的接收标志是如何置1和清零的：

- 检测到SOF前，具有灵活数据速率的控制器局域网 （CAN FD）一直保持空闲状态。
- 检测到SOF后，模块将接收仲裁和控制字段。
- 根据DNCNT和接收的DLC，接收过滤将开始。更多详细信息请参见图56-16。
- 如果没有任何过滤器匹配，仍将接收报文，但不会存储。
- 如果有过滤器匹配，器件会检查过滤器指向的接收对象是否已满。
- 如果接收对象已满， RXOVIF将置1。
- 如果接收对象未满，则会接收剩余的数据字节并将其存储至接收对象。
- 如果接收到完整的报文，将存储该报文，附加时间戳，并将接收标志置1：FIFO状态标志将更新， FIFO头将递增。
- 如果检测到错误，在接收报文期间，将发送错误帧并将相应的错误标志置1。

图 56-19： 接收状态图



 © 2019 Microchip Technology Inc.

56.9.3 复位RX FIFO

可以使用以下任一选项复位接收FIFO：

- 将CFDxFIFOCONn.FRESET置1
- 将模块置于 “配置”模式 （OPMOD = 100）

复位FIFO会将头和尾指针以及CFDxFIFOSTAn寄存器复位。CFDxFIFOCONn寄存器中的设置保持不变。

使用FRESET复位RX FIFO前，请确保没有使能的过滤器指向FIFO。

56.9.4 接收过程中DLC与有效负载大小不匹配

PLSIZE在接收报文对象中保留一定数量的字节。模块按照下面的方式处理DLC和有效负载大小之间的不匹配：

- 如果由DLC指定的字节数小于由PLSIZE指定的字节数，则接收报文字节将存储在报文对象中，不会进行任何填充。
- 如果由DLC指定的字节数大于由PLSIZE指定的字节数，则会存储适合接收报文对象的数据字节，丢弃不适合的其他数据字节。模块可确保RAM中的下一个报文对象不会被覆盖。模块将把报文存储在接收对象中，并且RX FIFO状态标志将更新。此外， CFDxINT.IVMIF和CFDxBDIAG1.DLCMM标志将置1。

56.9.5 报文接收代码示例

通过过滤器0接收CANFD扩展帧并将报文保存在FIFO 1中的代码示例如例56-3所示。

例56-3： 报文接收

```
#include <xc.h>

/* This code example demonstrates a method to configure the CAN FD
module to receive the extended ID CAN FD messages. This uses CAN1,
FIFO1 and filter 0. FIFO1 is configured to receive 2 messages. */


/* Include fuse configuration code here. */

#defineMAX_WORDS 100

unsigned int __attribute__((aligned(4)))CanRxBuffer[MAX_WORDS];


/*data structure to implement a CANFD  message buffer. */

/* CANFD Message Time Stamp */

typedefunsignedlong CANFD_MSG_TIMESTAMP;
```

未经测试的代码 — 信息 — 勿复制

报文接收代码示例——续

```
/* CANFD RX Message Object Control*/
typedef struct _CANFD_RX_MSGOBJ_CTRL {
    unsigned DLC:4;
    unsigned IDE:1;
    unsigned RTR:1;
    unsigned BRS:1;
    unsigned FDF:1;
    unsigned ESI:1;
    unsigned unimplemented1:2;
    unsigned FilterHit:5;
    unsigned unimplemented2:16;
} CANFD_RX_MSGOBJ_CTRL;
/* CANFD RX Message ID*/
typedefstruct_CANFD_MSGOBJ_ID {
    unsigned SID:11;
    unsigned long EID:18;
    unsigned SID11:1;
    unsigned unimplemented1:2;
} CANFD_MSGOBJ_ID;

/* CANFD RX Message Object */
typedef union _CANFD_RX_MSGOBJ {
    struct  {
        CANFD_MSGOBJ_ID id;
        CANFD_RX_MSGOBJ_CTRL ctrl;
        CANFD_MSG_TIMESTAMP timeStamp;
    } bF;
    unsigned int word[4];
    unsigned char byte[8];
} CANFD_RX_MSGOBJ;


intmain(void)
{
```

未经测试的代码——仅供参考

 © 2019 Microchip Technology Inc.

报文接收代码示例——续

```
/* The PIC32 device features I/O remap. This I/O remap configuration
for the CAN FD module can be performed here. */
SetIORemapForECANModule();

/* Enable the CANFD module */
CFD1CONbits.CON = 1;

/* Place CAN module in configuration mode */
CFD1CONbits.REQOP = 4;
while(CFD1CONbits.OPMOD != 4);

/* Initialize the C1FIFOBA with the start address of the CAN FIFO
message buffer area. */
CFD1FIFOBA = (unsigned int) &CanRxBuffer;

/* Set up the CANFD module for 1 Mbps of Nominal bit rate speed and
2 Mbps of Data bit rate. */
CFD1NBCFG = 0x003E0F0F;
CFD1DBTCFG = 0x001E0707;
CFD1TDC = 0x00021F00;//TDCMOD is Auto

/* Configure CANFD module to enable BRS */
CFD1CONbits.BRSDIS = 0x0;
CFD1CONbits.STEF = 0x0;//Don't save transmitted messages in TEF
CFD1CONbits.TXQEN = 0x0; // No TXQ

/* Configure FIFO1 to Receive 2 messages*/
CFD1FIFOCON1bits.FSIZE = 0x1;//2 messages
CFD1FIFOCON1bits.PLSIZE = 0x7; //64 bytes of data
CFD1FIFOCON1bits.TXEN = 0x0; //Receive fifo
```

未经测试的代码——仅供参考

报文接收代码示例——续

```
/* Configure filter 0 and MASK 0 to accept extended id messages with id = 2 and 3 */
CFD1FLTCON0bits.F0BP = 1; // message stored in FIFO1
CFD1FLTOBJ0 = 0x40001000; // EID = 0x00002, Match messages with //extended identifier address.
CFD1MASK0 = 0xFFFFF7FF; // MEID = 0x1FFFE - Last it is 0. Match Message //Types
CFD1FLTCON0bits.FLTEN0 = 1; // Enable the filter 0

/* Place the CAN module in Normal mode. */
CFD1CONbits.REQOP = 0;
while(CFD1CONbits.OPMOD != 0);

/* Get the address of the message buffer to read the received messages.*/
/* set UINC bit to update the FIFO tail */
CANFD_RX_MSGOBJ *rxObj;
rxObj = (CANFD_RX_MSGOBJ *)CFD1FIFOUA1;
while(CFD1FIFOSTA1bits.TFNRFNIF ==0);
//Process the received messages
CFD1FIFOCON1bits.UINC = 1; // Update the FIFO message pointer.
while(1);
}
```

未经测试的代码——仅供参考

 © 2019 Microchip Technology Inc.

56.10 FIFO行为

本节介绍TEF和TXQ使能时的FIFO行为。 FIFO1配置为TX FIFO， FIFO2配置为RX FIFO。其余的FIFO未配置。

> **注 1：** 起始地址根据FIFO和PLSIZE中的对象数进行计算。
>
> **2：** 表56-12给出的FIFO起始地址在TEF从0x1400开始时计算。

表56-12： 示例FIFO配置

FIFO	FIFO中的对象	每个对象的有效负载	时间戳	对象中的字节数	FIFO中的字节数	起始地址
TEF	12	不适用	有	12	144	0x1400
TXQ	8	32	不适用	40	320	0x1490
FIFO1	5	64	不适用	72	360	0x15D0
FIFO2	16	64	有	76	1216	0x1738
FIFO3	不适用	—	—	—	—	0x1BF8

56.10.1 FIFO状态标志

FIFO 1至FIFO 31可配置为发送或接收FIFO。CFDxFIFOSTAn中的相同状态标志用于发送和接收FIFO。根据所选配置，状态标志的行为会有所不同。

56.10.1.1 TX FIFO状态标志

有三种发送状态标志：

- TFEIF （**TFE**RFF**IF**）：发送FIFO空IF；当FIFO为空时置1。
- TFHIF （**TFH**RFH**IF**）：发送FIFO半空IF；当FIFO不到半满时置1。
- TFNIF （**TFN**RFN**IF**）：发送FIFO不满IF；当FIFO未满时置1。

当有空间将新报文对象加载到FIFO中时，发送FIFO的状态标志将置1。加载第一个报文对象前（FIFO复位后），所有状态标志将置1。 FIFO完全加载后，所有标志将清零。

56.10.1.2 RX FIFO状态标志

有三种接收状态标志：

- RFFIF （TFE**RFFIF**）：接收FIFO满IF；当FIFO已满时置1。
- RFHIF （TFH**RFHIF**）：接收FIFO半满IF；当FIFO至少为半满时置1。
- RFNIF （TFN**RFNIF**）：接收FIFO非空IF；当FIFO中至少有一条报文时置1。

FIFO中有接收报文时，接收FIFO的状态标志将置1。接收到第一条报文前（FIFO复位后），所有状态标志将清零。 FIFO已满时，所有标志将置1。

56.10.1.3 TXQ状态标志

有两种TXQ状态标志：

- TXQEIF：TXQ空IF——当TXQ为空时置1。
- TXQNIF：TXQ未满IF——当TXQ未满时置1。

当有空间将新报文对象加载到TXQ中时， TXQ的状态标志将置1。加载第一个报文对象前（TXQ复位后），所有状态标志将置1。 TXQ完全加载后，所有标志将清零。

56.10.1.4 TEF状态标志

有四种TEF状态标志：

- TEFFIF：TEF满IF——当TEF已满时置1
- TEFHIF：TEF半满IF——当TEF至少为半满时置1
- TEFNEIF：TEF非空IF——当TEF中至少有一条报文时置1
- TEFOVIF：TEF溢出IF——发生溢出时置1

FIFO中有发送报文时， TEF的状态标志将置1。存储第一条报文前 （TEF复位后），所有状态标志将清零。当TEF已满时，所有标志将置1。

56.10.2 发送FIFO行为

FIFO 1配置为TX FIFO。CFDxFIFOCONn用于控制FIFO。CFDxFIFOSTAn包含状态标志和FIFO索引 （FIFOCI）。 CFDxFIFOUAn包含要加载的下一个发送报文对象的用户地址。

实际RAM地址通过公式56-1计算。

图56-20至图56-25展示了状态标志、用户地址和FIFO索引是如何更新的。

图56-20显示了复位后FIFO 1的状态。报文对象MO0至MO4为空。所有状态标志置1。用户地址和FIFO索引指向MO0。

图 56-20： FIFO 1—— 初始状态

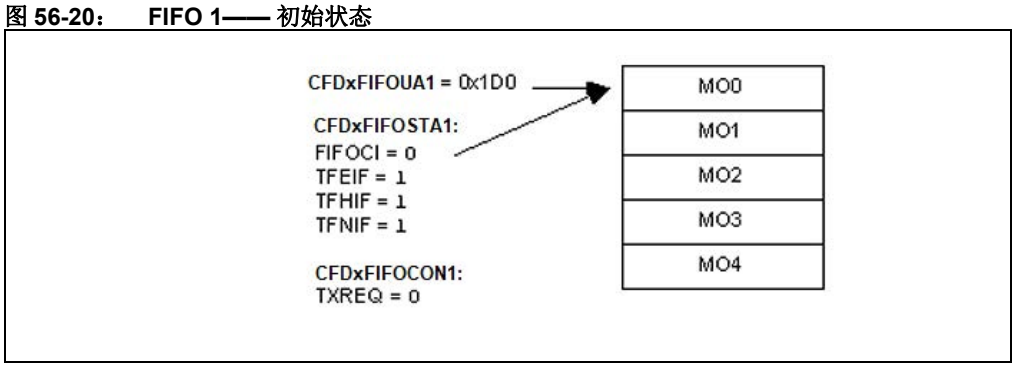


图56-21展示了加载第一条报文 （MSG0）后FIFO 1的状态。 MO0现在包含MSG0。用户应用程序将CFDxFIFOCONn.UINC置1，使FIFO头前进。用户地址现在指向MO1。由于FIFO不再为空，因此TFEIF将清零。用户应用程序现在将TXREQ置1以请求发送MSG0。

图 56-21： FIFO 1—— 加载第一条报文后

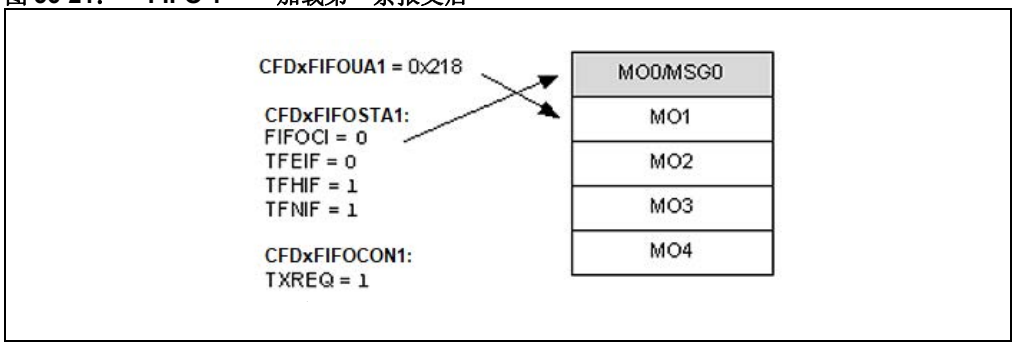


图56-22展示了发送MSG0后FIFO 1的状态。FIFO再次为空。TFEIF置1，TXREQ清零。FIFOCI现在通过用户地址0x218指向MO1。

 © 2019 Microchip Technology Inc.

图 56-22： FIFO 1—— 发送第一条报文后

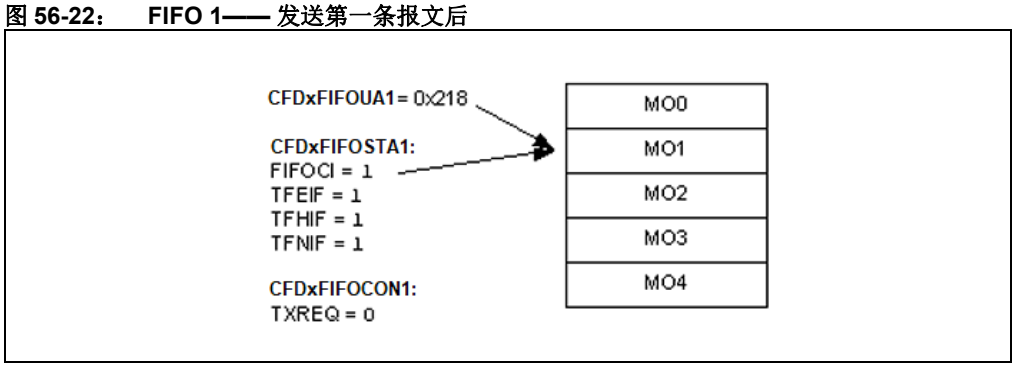


图56-23展示了再加载三条报文后FIFO 1的状态：MSG1-MSG3。用户地址现在指向MO4。由于FIFO现在不到半空，所以TFHIF将清零。

图 56-23： FIFO 1—— 再加载三条报文后

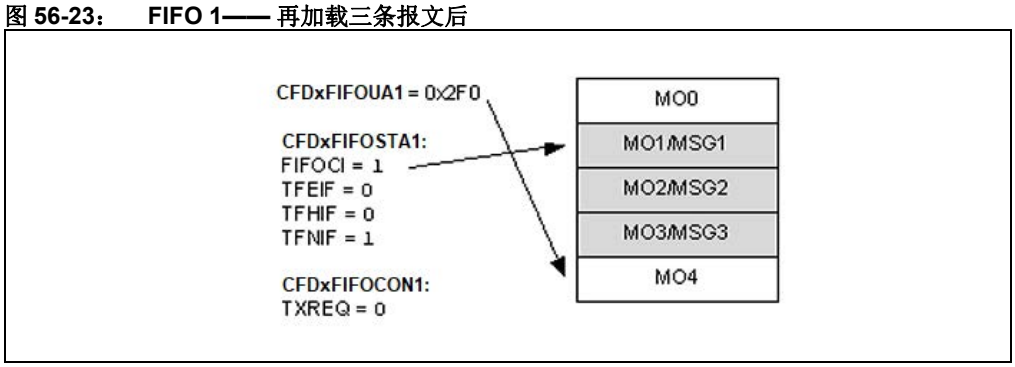


图56-24展示了再加载两条报文后FIFO 1的状态：MSG4和MSG5。C1FIFOUA1L现在指向MO1。由于FIFO已满，所以所有状态标志现在清零。用户地址和FIFO索引现在指向MO1。用户应用程序现在将TXREQ置1以请求发送MSG1-MSG5。

图 56-24： FIFO 1——FIFO 完全加载后

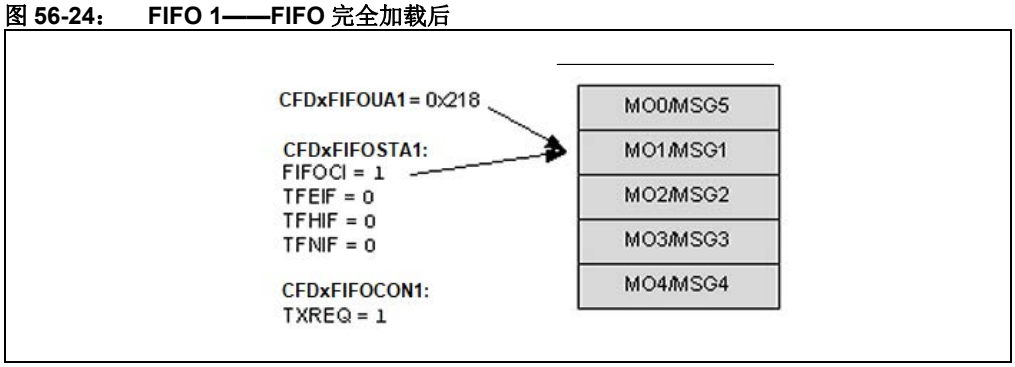
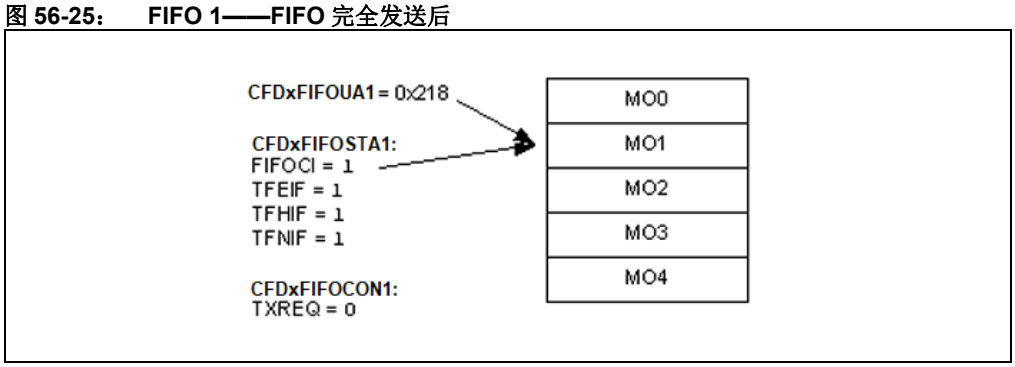


图56-25展示了发送MSG1-MSG5后FIFO 1的状态。FIFO再次为空。所有状态标志置1，TXREQ清零。用户地址和FIFO索引再次指向MO1。

图 56-25： FIFO 1——FIFO 完全发送后



56.10.3 接收FIFO行为

FIFO 2配置为RX FIFO。CFDxFIFOCON2用于控制FIFO。CFDxFIFOSTA2包含状态标志和FIFO索引 （FIFOCI）。 CFDxFIFOUA2包含要读取的下一个报文对象的用户地址。

实际RAM地址通过公式56-1计算。

图56-26至图56-33展示了状态标志、用户地址和FIFO索引是如何更新的。

图56-26显示了复位后FIFO 2的状态。报文对象MO0至MO15为空。所有状态标志清零。用户地址和FIFO索引指向MO0。

图 56-26： FIFO 2—— 初始状态

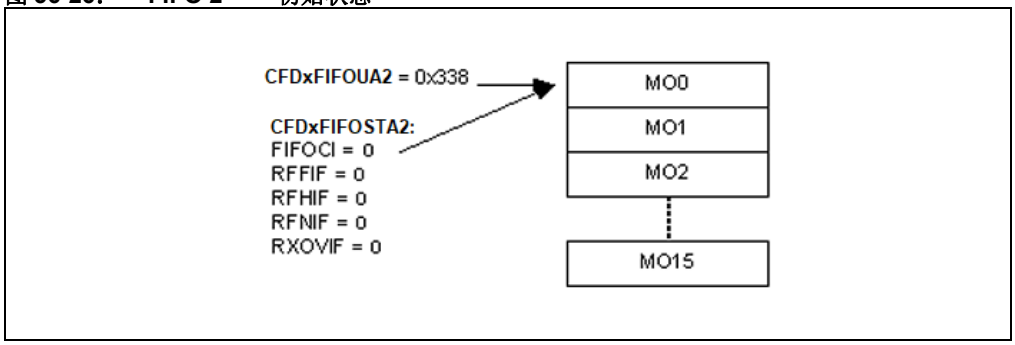


图56-27展示了接收第一条报文（MSG0）后FIFO 2的状态。 MO0现在包含MSG0。 FIFO索引现在指向MO1。由于FIFO不再为空，所以RFNIF置1。

图 56-27： FIFO 2—— 接收第一条报文后

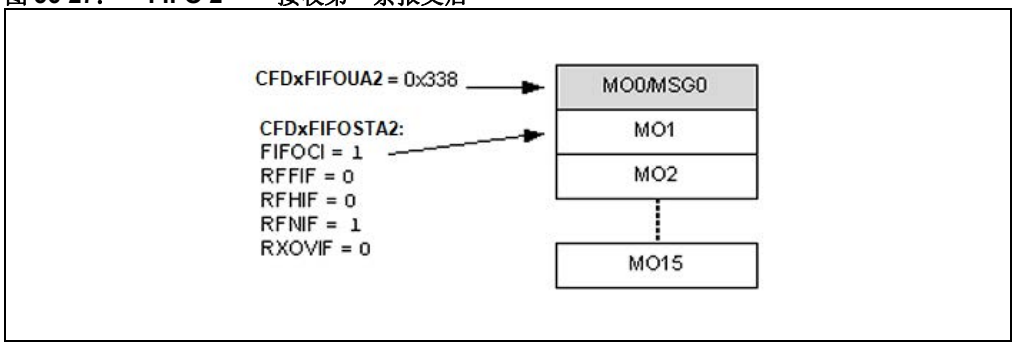


图56-28展示了读取MSG0后FIFO 2的状态。用户应用程序从RAM读取报文并将CFDxFIFOCONn.UINC置1。用户地址递增并指向MO1。 FIFO索引保持不变。 FIFO再次为空。所有标志清零。

 © 2019 Microchip Technology Inc.

图 56-28： FIFO 2—— 读取第一条报文后

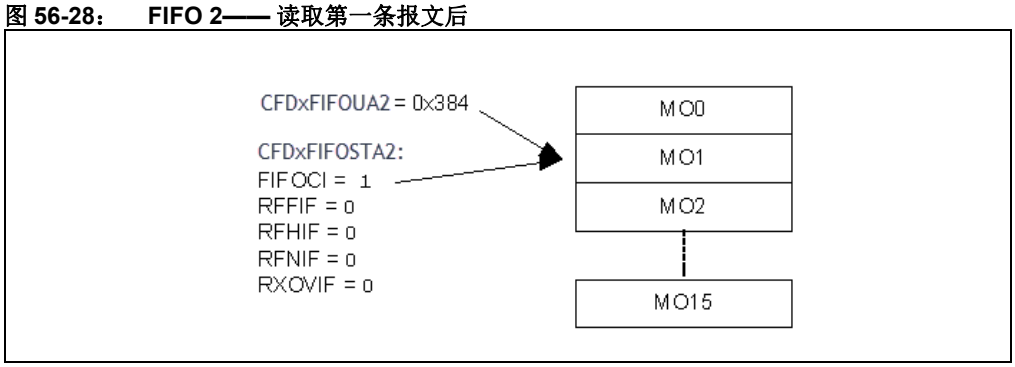


图56-29展示了再接收八条报文后FIFO 2的状态：MSG1-MSG8。用户地址仍指向MO1。由于FIFO现在已半满，所以RFNIF和RFHIF置1。 FIFO索引指向MO9。

图 56-29： FIFO 2—— 半满

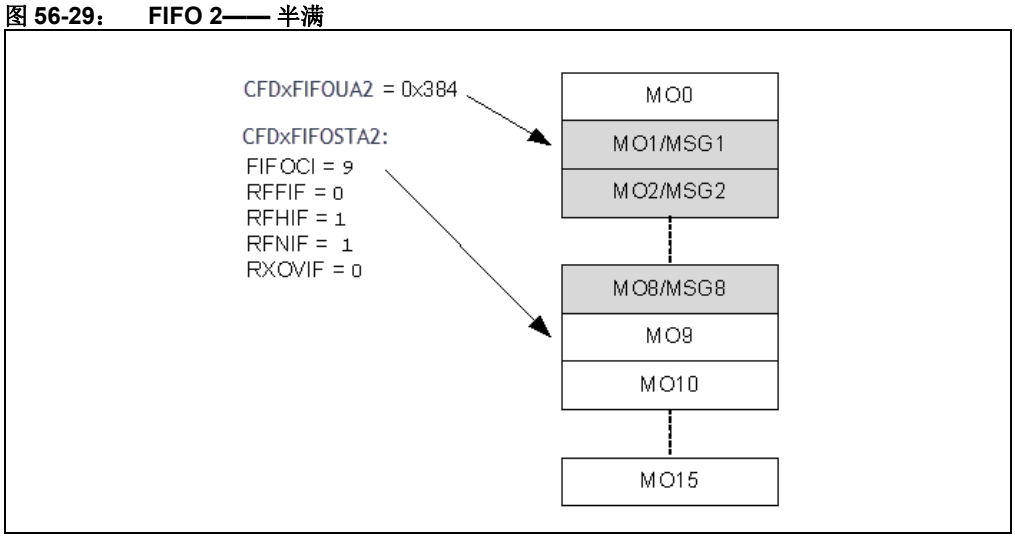


图56-30展示了再接收10条报文后FIFO 2的状态：MSG5-MSG15。用户地址仍指向MO1。 FIFO索引指向MO0。 RFNIF和RFHIF置1。

图 56-30： FIFO 2——FIFO 即将满

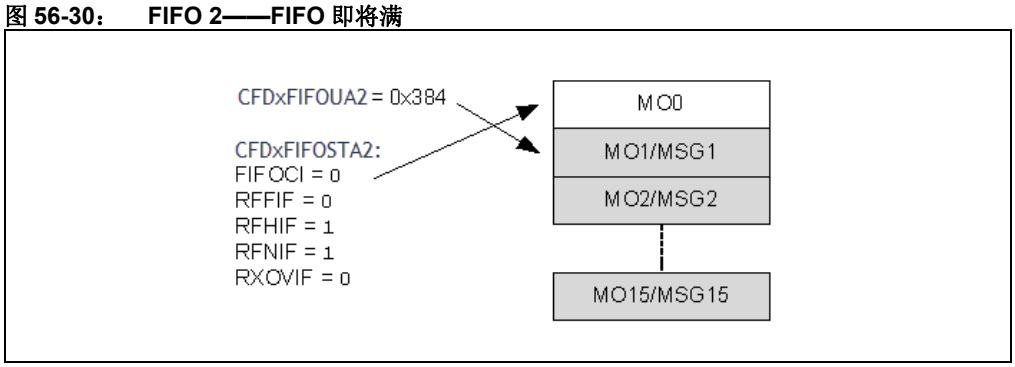


图56-31展示了再接收一条报文后FIFO 2的状态：MSG16。由于FIFO已满，所以所有状态标志置1。用户地址和FIFO索引指向MO1。

图 56-31： FIFO 2——FIFO 已满

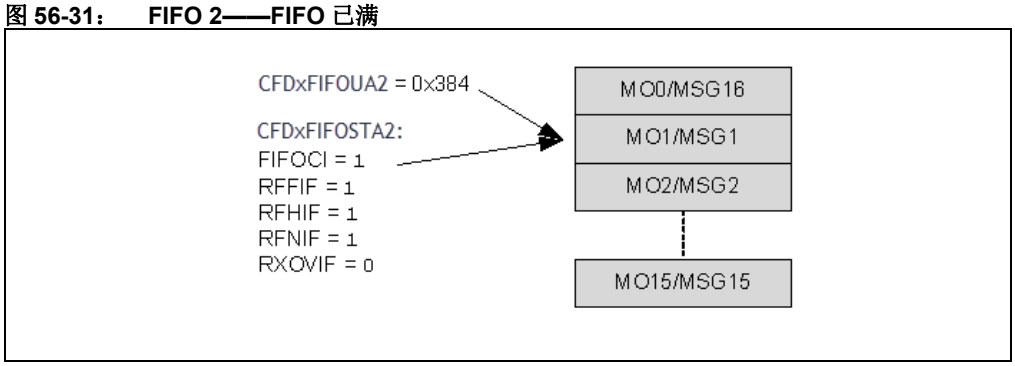


图56-32展示了再接收一条报文后FIFO 2的状态。由于FIFO 2已满，所以发生溢出。报文被丢弃，RXOVIF置1。用户地址和FIFO索引保持不变。

图 56-32： FIFO 2——FIFO 溢出

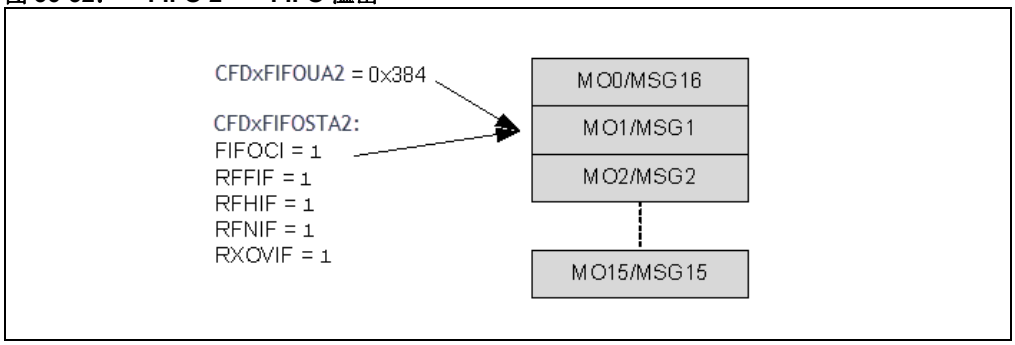
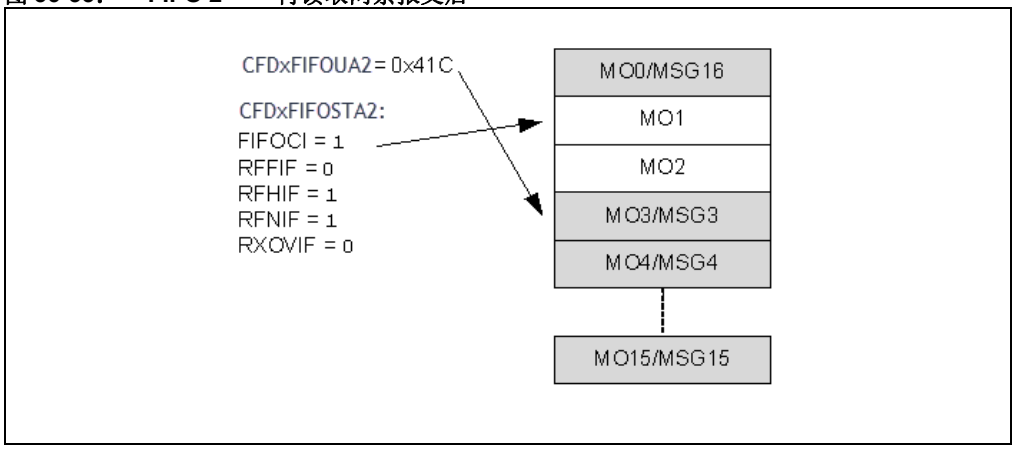


图56-33展示了应用程序将RXOVIF清零并再读取两条报文后FIFO 2的状态。由于FIFO不再为满，所以RFFIF清零。用户地址指向MO3。 FIFO索引保持不变。

图 56-33： FIFO 2—— 再读取两条报文后



 © 2019 Microchip Technology Inc.

56.10.4 发送队列行为

CFDxTXQCON 用于控制 TXQ。 CFDxTXQSTA 包含状态标志和 TXQ 索引 （TXQCI） 。CFDxTXQUA包含要加载的下一个发送报文对象的用户地址。

具有灵活数据速率的控制器局域网（CAN FD）使用TXQCI计算要发送的下一条报文。 TXQCI不会线性递增，每次发送报文或TXREQ置1后，都会重新计算。

实际RAM地址通过公式56-1计算。

图56-34至图56-39展示了状态标志和用户地址是如何更新的。用户应用程序无需使用TXQCI，因此图中未显示。

图56-34显示了复位后TXQ的状态。报文对象MO0至MO7为空。所有状态标志置1。用户地址指向MO0。

图 56-34： TXQ—— 初始状态

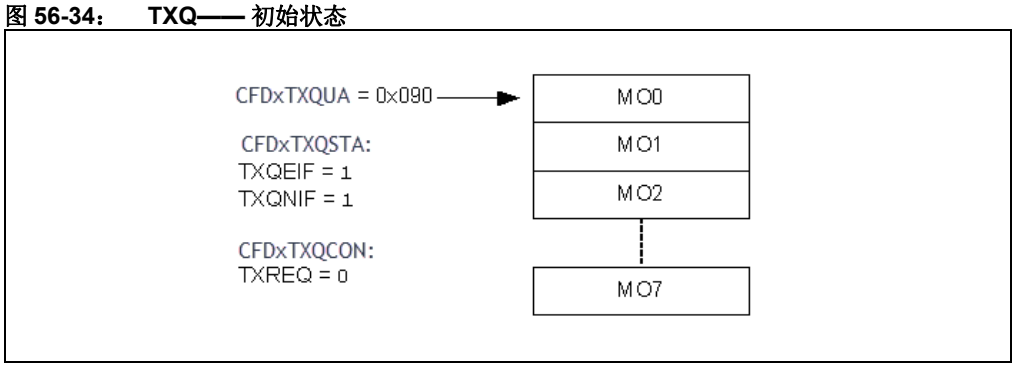


图56-35展示了加载第一条报文 （MSG0）后TXQ的状态。 MO0现在包含MSG0。用户应用程序将CFDxTXQCON.UINC置1，使FIFO头前进。用户地址现在指向MO1。由于队列不再为空，因此TXQEIF清零。用户应用程序现在将TXREQ置1以请求发送MSG0。

图 56-35： TXQ—— 加载第一条报文后

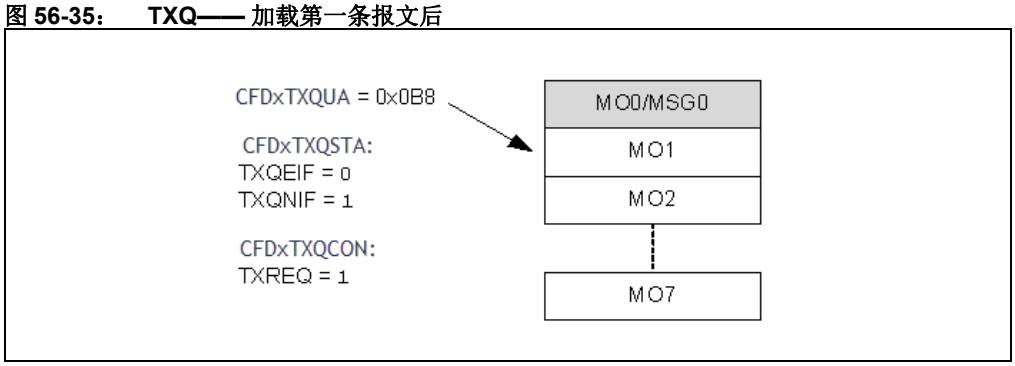


图56-36展示了发送MSG0后TXQ的状态。TXQ再次为空。TFEIF置1，TXREQ清零。由于UINC未置1，所以用户地址仍然指向MO1。

图 56-36： TXQ—— 发送第一条报文后

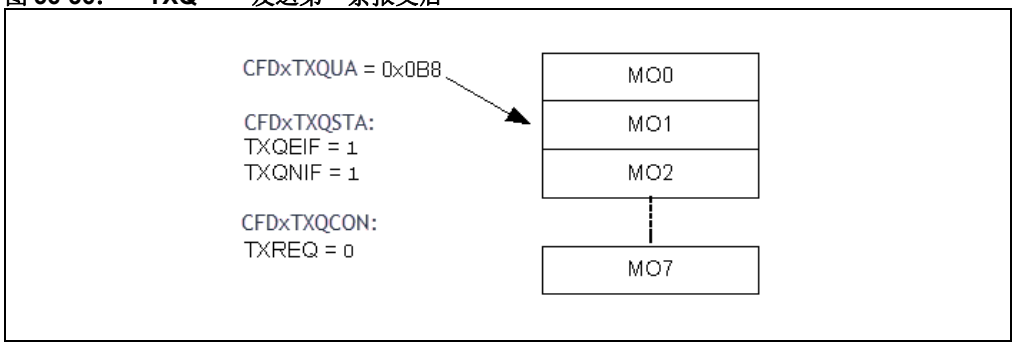


图56-37展示了加载MSG1且UINC置1后TXQ的状态。用户地址现在指向下一个空闲报文对象：MO0。

图 56-37： TXQ—— 加载下一条报文后

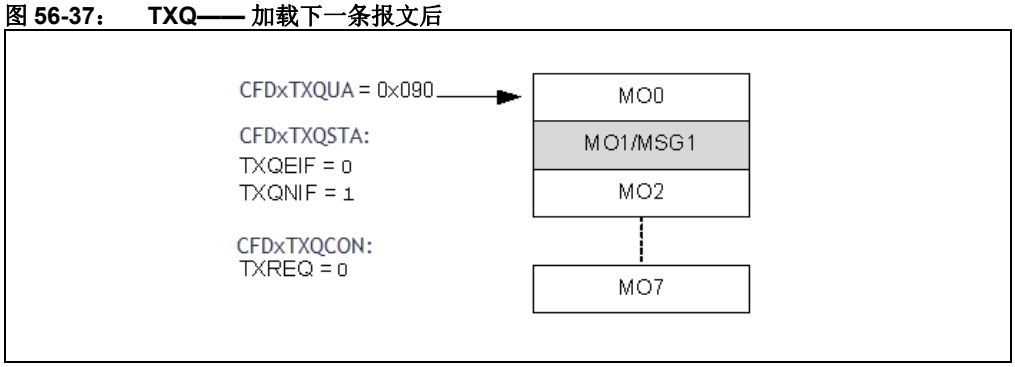
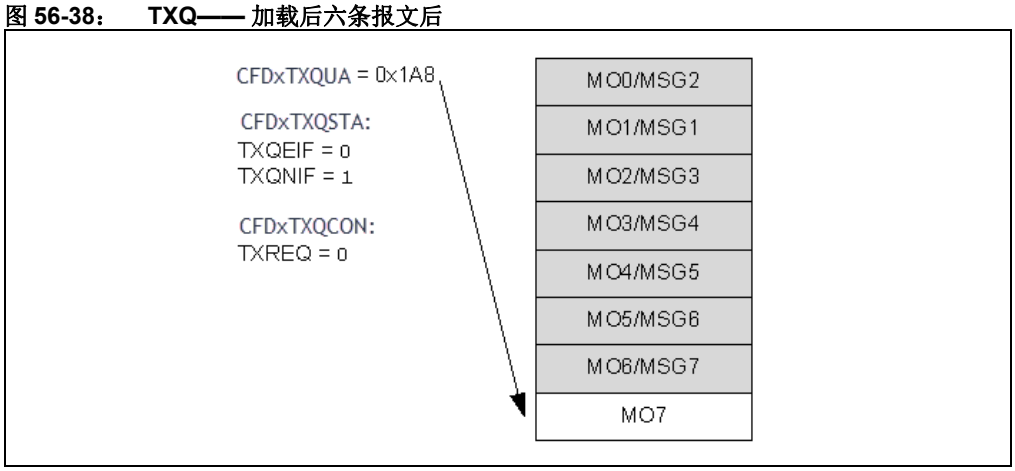


图56-38展示了再加载六条报文后TXQ的状态：MSG2-MSG7。用户地址现在指向最后一个空闲报文对象：MO7。

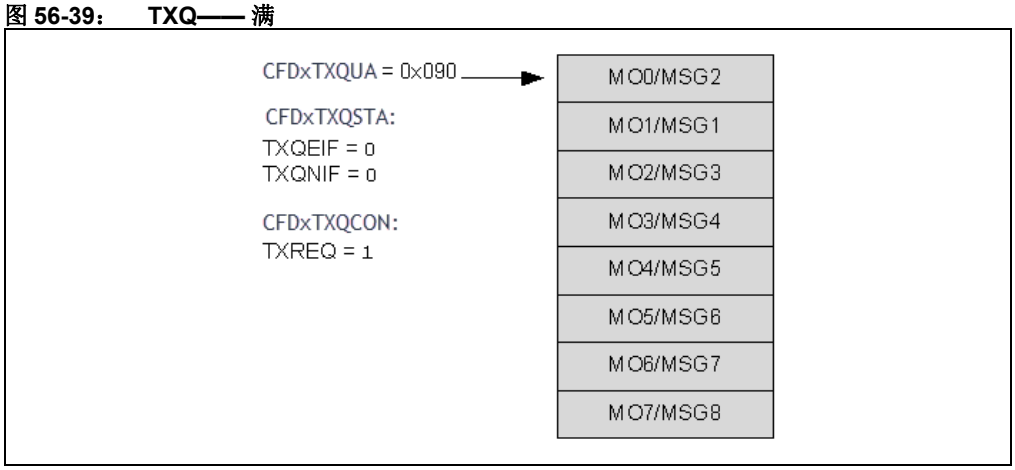
图 56-38： TXQ—— 加载后六条报文后



 © 2019 Microchip Technology Inc.

图56-39展示了加载MSG8且UINC置1后TXQ的状态。TXQ现在已满，所有标志清零。用户地址现在指向MO0。用户应用程序现在将TXREQ置1。报文将根据其ID的优先级进行发送。

图 56-39： TXQ—— 满



56.10.5 发送事件FIFO行为

CFDxTEFCON用于控制TEF。 CFDxTEFSTA包含状态标志。 CFDxTEFUA包含要读取的下一个报文对象的用户地址。

实际RAM地址通过公式56-1计算。

图56-40至图56-47展示了状态标志和用户地址是如何更新的。 TEF会存储发送报文。因此，标志的行为类似于RX FIFO。

图56-40显示了复位后TEF的状态。报文对象MO0至MO11为空。所有状态标志清零。用户地址指向MO0。

图 56-40： TEF—— 初始状态

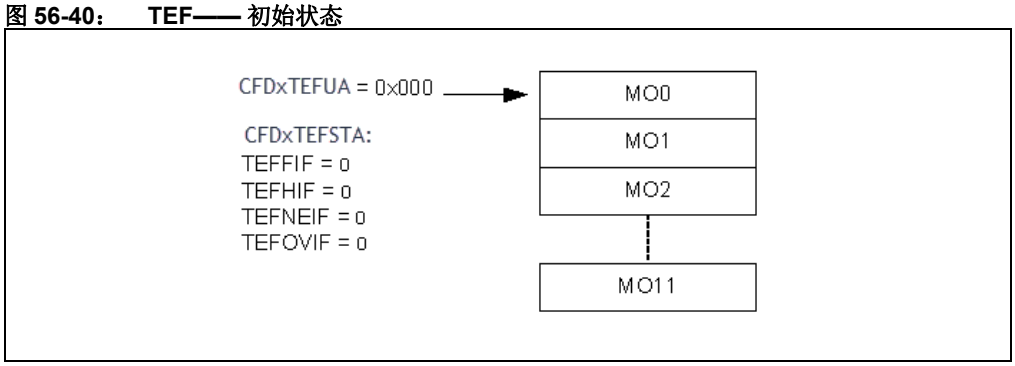


图56-41显示了存储第一条发送报文后TEF的状态。 MO0包含ID0，即MSG0的ID。由于TEF不为空，所以TEFNEIF置1。用户地址指向MO0。

图 56-41： TEF—— 存储第一条发送报文后

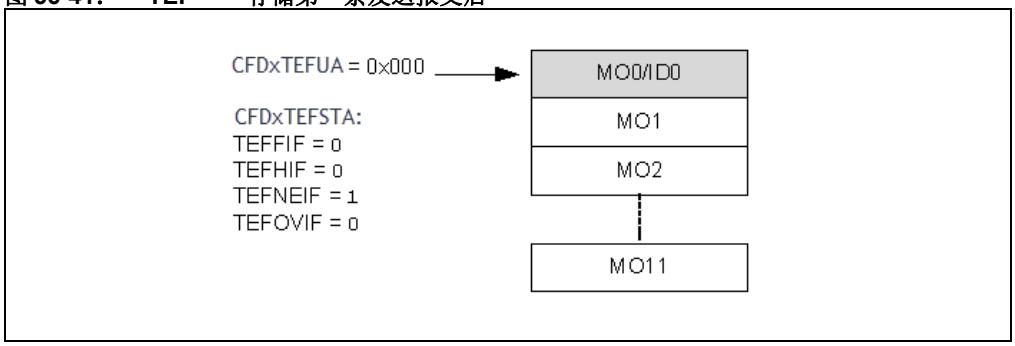


图56-42展示了读取ID0后TEF的状态。用户应用程序从RAM读取ID并将CFDxTEFCON.UINC置1。用户地址递增并指向MO1。 TEF再次为空。所有标志清零。

图 56-42： TEF—— 读取第一个 ID 后

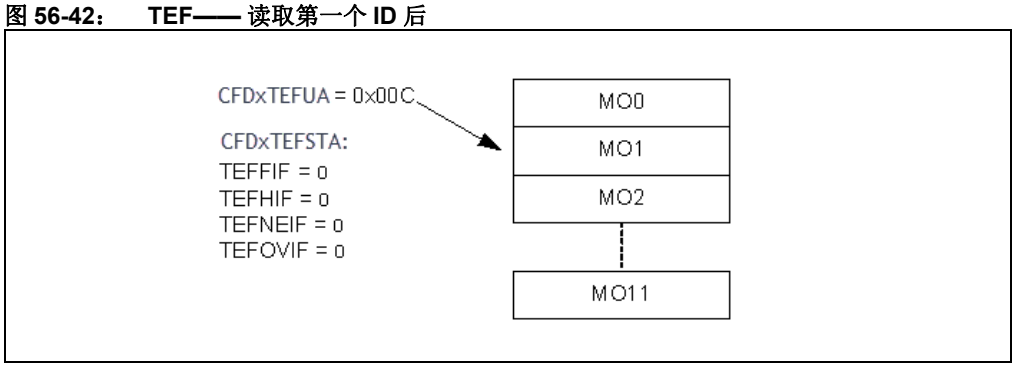
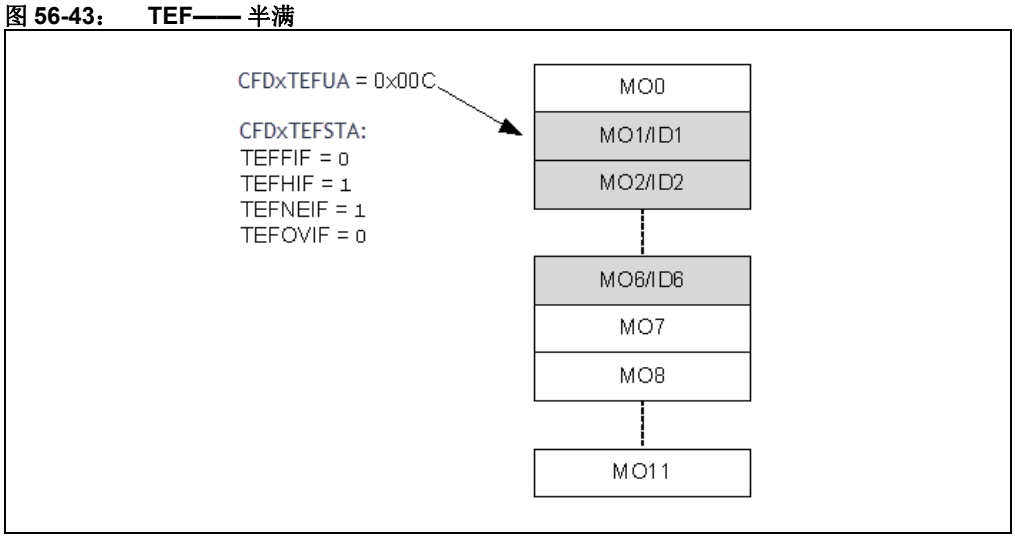


图56-43展示了再发送六条报文后TEF的状态：MSG1-MSG6。用户地址指向MO1。由于TEF已经半满，所以TEFNEIF和TEFHIF置1。

图 56-43： TEF—— 半满



 © 2019 Microchip Technology Inc.

图56-44展示了再发送五条报文后TEF的状态：MSG7-MSG11。用户地址仍指向MO1。TEFNEIF和TEFHIF置1。

图 56-44： TEF—— 即将满

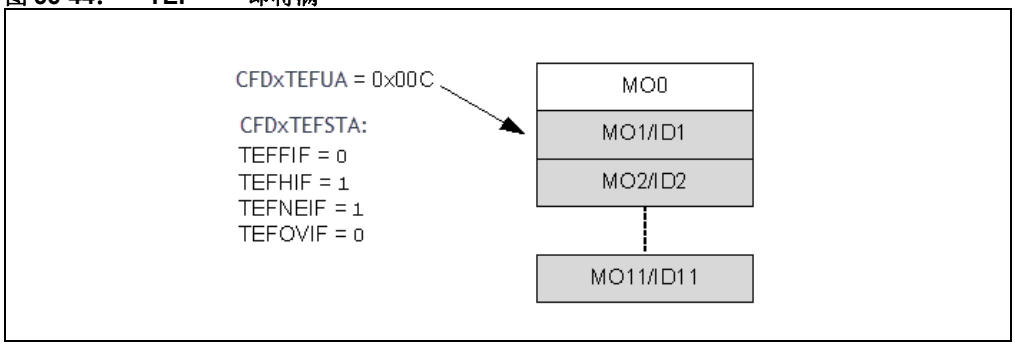


图56-45展示了再发送一条报文后TEF的状态：MSG12。由于TEF已满，所以所有状态标志置1。用户地址指向MO1。

图 56-45： TEF—— 满

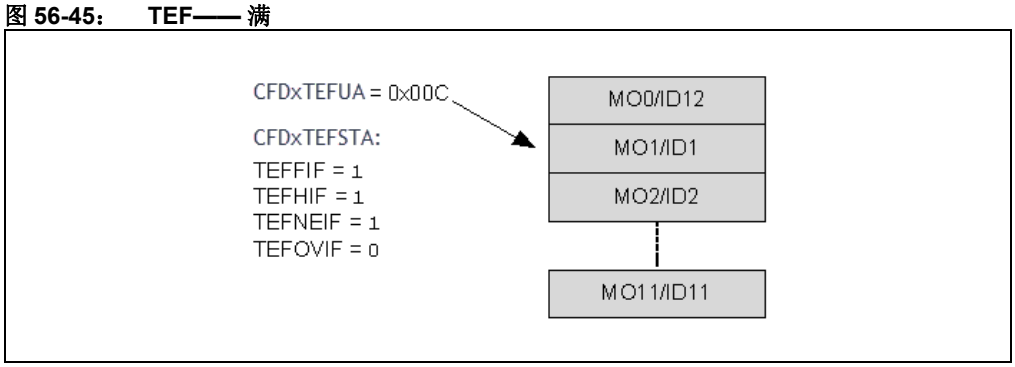


图56-46展示了再发送一条报文后TEF的状态。由于TEF已满，所以发生溢出。ID被丢弃，TEFOVIF置1。用户地址保持不变。

图 56-46： TEF—— 溢出

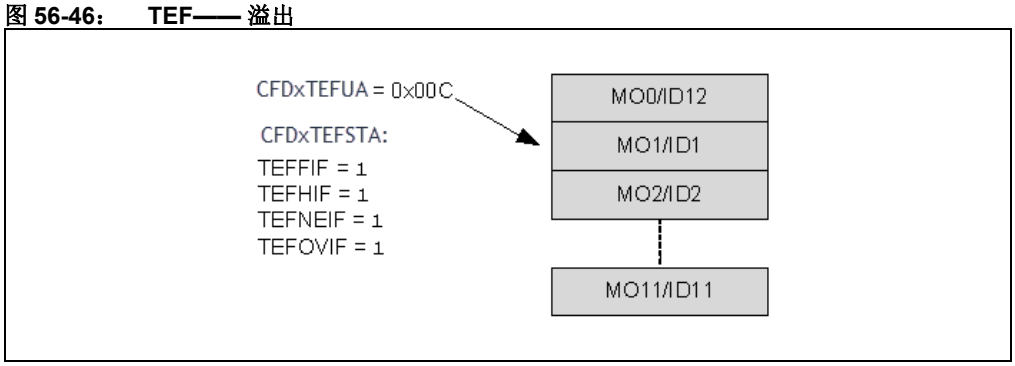
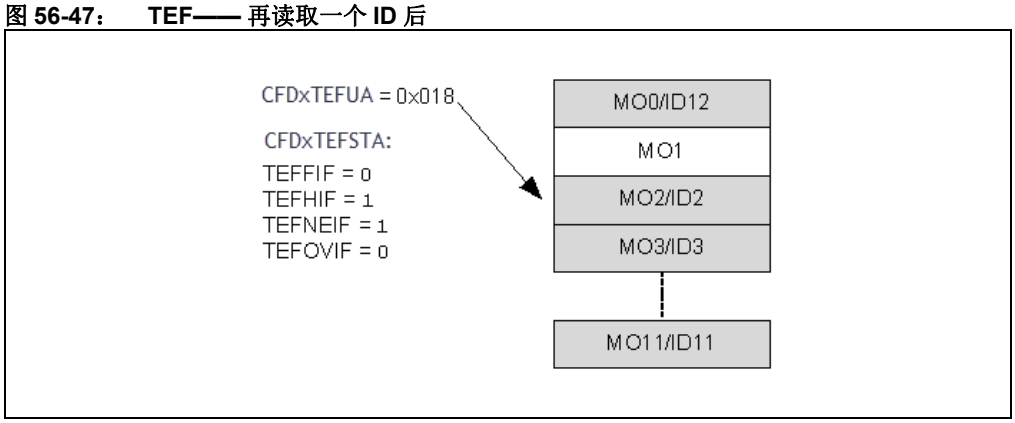


图56-47展示了应用程序将TEFOVIF清零并再读取一条报文后TEF的状态。由于TEF不再为满，所以TEFFIF清零。用户地址指向MO2。

图 56-47： TEF—— 再读取一个 ID 后



 © 2019 Microchip Technology Inc.

56.11 时间戳

具有灵活数据速率的控制器局域网 （CAN FD）包含时基计数器 （TBC）。 TBC是一个32位自由运行计数器，它按SYSCLK的倍数递增，并在以下情况下归零：

- 使用CFDxTSCON.TBCPRE为TBC配置预分频比。
- 将CFDxTSCON.TBCEN置1可使能TBC。
- 将TBCEN清零可禁止、停止和复位TBC。
- 在写入CFDxTBC前必须通过将CFDxTSCON.TBCEN清零来禁止TBC。
- 必须将CFDxTEFCON.TEFTSEN置1以便为TEF中的报文附加时间戳。
- 必须将CFDxFIFOCONn.RXTSEN置1以便为单个RX FIFO中的报文附加时间戳。
- 应用程序可以随时读取CFDxTBC。与所有多字节计数器类似，应用程序必须考虑到计数器会递增，并且可能会在读取计数器的不同字节时计满返回。

所有时间戳都是32位，这使得时间戳能够以高分辨率用于系统时间同步。

如果TBCIE置1， TBC的计满返回将产生中断。

可以在帧的头部或尾部为报文附加时间戳，具体取决于CFDxTSCON.TSEOF。当TSEOF = 0时，CFDxTSCON.TSRES指定是否在SOF或 “保留位” 为FD帧附加时间戳。表56-13指定了附加时间戳的参考点。在参考点处，会捕捉TBC CFDxTBC的值并将其存储到报文对象中：

- 接收报文对象：TBC值存储在RXMSGTS中，请参见表56-11。
- TEF对象：TBC值存储在TXMSGTS中，请参见表56-9。

表56-13： 参考点

帧	CAN 2.0	CAN FD
TX的起点	SOF的采样点	SOF的采样点或FDF后的位
RX的起点	SOF的采样点	SOF的采样点或FDF后的位
有效TX	EOF结束前都没有错误	EOF结束前都没有错误
有效RX	EOF的倒数第二位前都没有错误	EOF的倒数第二位前都没有错误

56.12 中断

中断可以分成多层。较低层中断会通过将其复用为单个中断传播到较高层。图56-48展示了中断的各层：

- FIFO单独中断
- FIFO组合中断
- 主中断

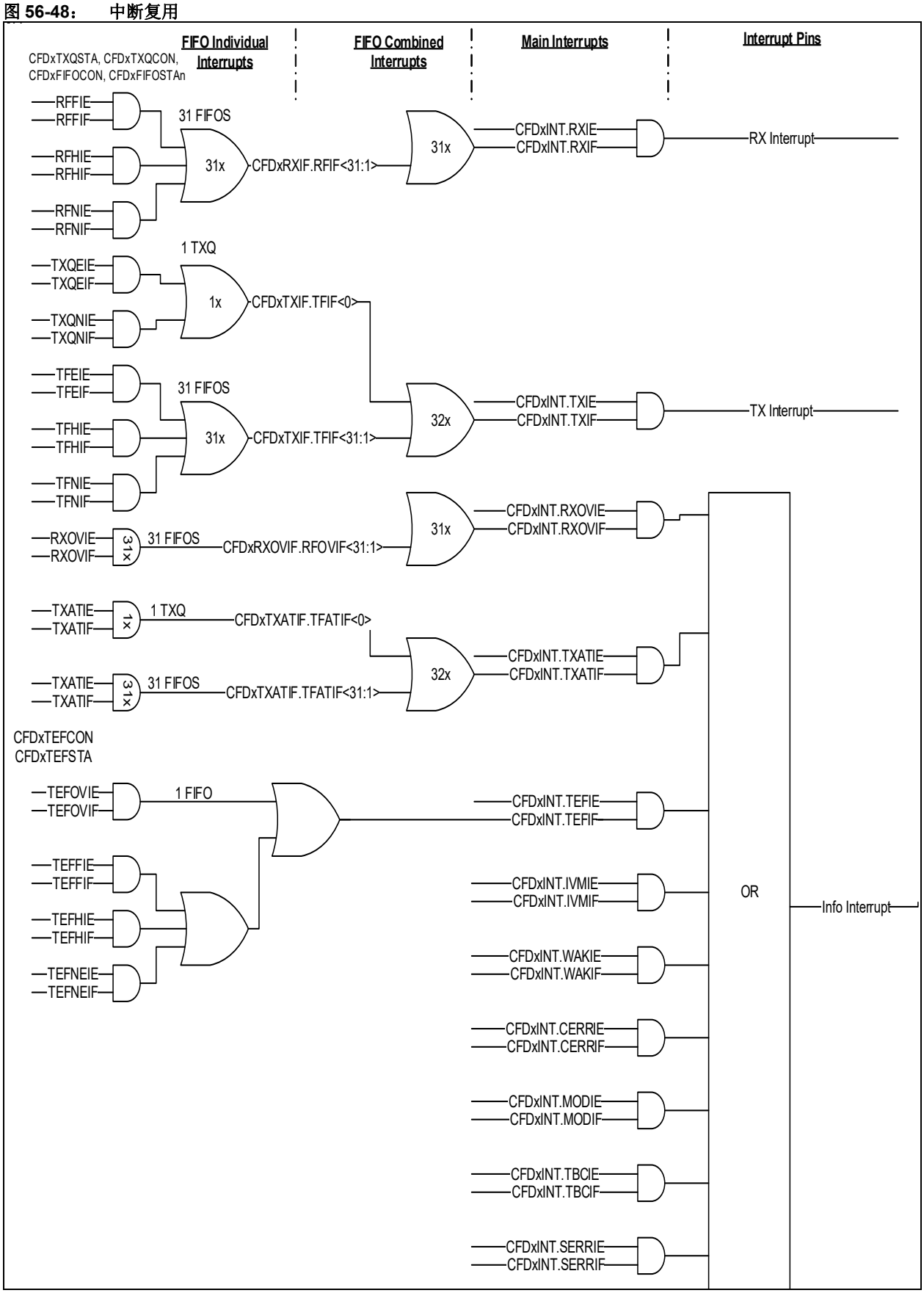
这些中断随后会汇集到三个独立的模块中断中：

- 接收中断
- 发送中断
- 信息中断

所有模块中断都是持久性的，这意味着必须先清除模块内引起中断的条件，然后才能删除中断请求。

 © 2019 Microchip Technology Inc.

图 56-48： 中断复用


FIFO Individual Interrupts
FIFO Combined Interrupts
Main Interrupts
Interrupt Pins
CFDxTXQSTA, CFDxTXQCON, CFDxFIFOCON, CFDxFIFOSTAn
RFFIE
RFFIF
RFHIE
RFHIF
RFNIE
RFNIF
31 FIFOS
31x
CFDxRXIF.RFIF<31:1>
31x
CFDxINT.RXIE
CFDxINT.RXIF
RX Interrupt
1 TXQ
TXQEIE
TXQEIF
TXQNIE
TXQNIF
1x
CFDxTXIF.TFIF<0>
TFEIE
TFEIF
TFHIE
TFHIF
TFNIE
TFNIF
31 FIFOS
31x
CFDxTXIF.TFIF<31:1>
32x
CFDxINT.TXIE
CFDxINT.TXIF
TX Interrupt
RXOVIE
RXOVIF
31x
31 FIFOS
CFDxRXOVIF.RFOVIF<31:1>
31x
CFDxINT.RXOVIE
CFDxINT.RXOVIF
TXATIE
TXATIF
1x
1 TXQ
CFDxTXATIF.TFATIF<0>
TXATIE
TXATIF
31x
31 FIFOS
CFDxTXATIF.TFATIF<31:1>
32x
CFDxINT.TXATIE
CFDxINT.TXATIF
CFDxTEFCON
CFDxTEFSTA
TEFOVIE
TEFOVIF
1 FIFO
CFDxINT.TEFIE
CFDxINT.TEFIF
TEFFIE
TEFFIF
TEFHIE
TEFHIF
TEFNEIE
TEFNEIF
CFDxINT.IVMIE
CFDxINT.IVMIF
CFDxINT.WAKIE
CFDxINT.WAKIF
CFDxINT.CERRIE
CFDxINT.CERRIF
CFDxINT.MODIE
CFDxINT.MODIF
CFDxINT.TBCIE
CFDxINT.TBCIF
CFDxINT.SERRIE
CFDxINT.SERRIF
OR
Info Interrupt

56.12.1 FIFO单独中断

CFDxFIFOCONn包含FIFO的中断允许， CFDxFIFOSTAn包含FIFO的中断标志。每个FIFO都有一个独立的寄存器。

56.12.2 发送队列中断

CFDxTXQCON包含TXQ的中断允许， CFDxTXQSTA包含TXQ的中断标志。

当TXQ的状态发生变化时，会发生TXQ中断。有两种中断源：

- TXQ未满中断——TXQNIF
- TXQ空中断——TXQEIF

可以单独允许这两种中断。中断无法由应用程序清除；它们将在FIFO的条件终止时清除。

两种中断源一起进行或运算，并在CFDxTXIF.TFIF<0>标志中反映。

56.12.2.1 接收FIFO中断——RFIF

当接收FIFO的状态发生变化时，会发生接收FIFO中断。有三种中断源：

- 接收FIFO满中断——RFFIF
- 接收FIFO半满中断——RFHIF
- 接收FIFO非空中断——RFNIF

可以单独允许这三种中断。中断无法由应用程序清除，它们将在FIFO的条件终止时清除。

三种中断源一起进行或运算，并在CFDxRXIF<31:1>标志中反映。

56.12.2.2 发送FIFO中断——TFIF

当发送FIFO的状态发生变化时，会发生发送FIFO中断。有三种中断源：

- 发送FIFO未满中断——TFNIF
- 发送FIFO半空中断——TFHIF
- 发送FIFO空中断——TFEIF

可以单独允许这三种中断。中断无法由应用程序清除，它们将在FIFO的条件终止时清除。

三种中断源一起进行或运算，并在CFDxTXIF<31:1>标志中反映。

56.12.2.3 接收FIFO溢出中断——RXOVIF

成功接收报文但FIFO已满时，单个FIFO的RXOVIF将置1。该标志必须由应用程序清零。

56.12.2.4 发送FIFO尝试中断——TXATIF

当由于错误导致报文重发失败且所有重发尝试次数都已用完时， TXATIF将置1。该标志必须由应用程序清零。

56.12.2.5 发送事件FIFO中断——TEFIF

当TEF的状态发生变化时，会发生TEF中断。有四种中断源：

- TEF满中断——TEFFIF
- TEF半满中断——TEFHIF
- TEF非空中断——TEFNEIF
- TEF溢出中断——TEFOVIF

 © 2019 Microchip Technology Inc.

TEF中断的工作原理类似于接收FIFO中断。可以单独允许所有四种中断。

TEFFIF、 TEFHIF和TEFNEIF无法由应用程序清零，它们将在FIFO的状态终止时清零。

TEFOVIF必须由应用程序清零。

四种中断源一起进行或运算，并在CFDxINT.TEFIF标志中反映。

56.12.3 FIFO组合中断

以下中断是单独的FIFO中断：

- FIFO/TXQ：RFIF、 TFIF、 RFOVIF和TFATIF

它们组合成单个中断状态寄存器：

- CFDxRXIF、 CFDxTXIF、 CFDxRXOVIF和CFDxTXATIF。

状态寄存器中的位按照下面的方式映射到FIFO：Bit 0映射到TXQ， Bit 1映射到FIFO 1， Bit 2映射到FIFO 2，依此类推，Bit 31映射到FIFO 31。由于Bit 0与TXQ对应，CFDxRXIF和CFDxRXOVIF的Bit 0会被保留。因此，通过读取一个寄存器，应用程序可以检查特定中断的所有FIFO的状态（例如任何待处理的RFIF）。

FIFO中断在CFDxFIFOCONn中允许。

TXQ中断在CFDxTXQCON中允许。

第56.12.1节 “FIFO单独中断”对清除FIFO中断进行了说明。

56.12.4 主中断

CFDxINT寄存器包含所有主中断。以下中断是所有组合FIFO中断的逻辑 “或”：RXIF、 TXIF、 RXOVIF和TXATIF。这些标志是只读的，必须在先前的层级中清零。

TEFIF在TEF中生成。该标志是只读的，必须在先前的层级中清零。

可以单独允许CFDxINT中的所有中断。

56.12.4.1 无效报文中断——IVMIF

如果在发送或接收最后一条报文期间检测到CAN 总线错误或DLC 不匹配， IVMIF 将置1。CFDxBDIAGn寄存器为每个错误设置一个标志。该标志必须由应用程序清零。

以下CAN总线错误将在发送了错误帧时触发中断：CRC错误、填充位错误、格式错误、位错误或ACK错误。

如果接收报文的ESI置1，标志将不会置1。

56.12.4.2 唤醒中断——WAKIF

在模块处于 “休眠”模式时检测到总线活动。该标志必须由应用程序清零。

56.12.4.3 CAN总线错误中断——CERRIF

CFDxTREC寄存器将在发送和接收期间根据ISO11898-1:2015对错误计数。 CERRIF标志将根据错误计数器的数值置1。该标志必须由应用程序清零。

每当TEC/REC计数器的阈值被以下条件超过时， CERRIF将置1：

- TEC或REC超过错误警告状态阈值
- 发送器或接收器转换为被动错误状态
- 发送器转换为离线状态
- 发送器或接收器从被动错误状态转换为主动错误状态
- 在离线恢复序列后，模块从离线转换为主动错误状态

当用户将CERRIF清零时，它将保持清零，直到发生新的计数器超限事件。

56.12.4.4 CAN模式更改中断——MODIF

当OPMOD位更改时， MODIF标志将置1。该标志必须由应用程序清零。

56.12.4.5 CAN定时器中断——TBCIF

当时基计数器计满返回时， TBCIF将置1。该标志必须由应用程序清零。

56.12.4.6 系统错误中断——SERRIF

- 总线带宽错误：

  接收和发送期间可能会发生带宽错误。

  当模块不能在下一条报文到达前将接收到的CAN报文写入RAM中时，接收报文组合缓冲区（RX MAB）将发生上溢。

  当模块向TX MAB馈送的速度不足以向比特流处理器提供一致的数据时，发送报文组合缓冲区 （TX MAB）将发生下溢。

  SERRIF标志将置1， CFDxVEC.ICODE位将设为`100 0101`。

- 处理RX MAB溢出错误：

  某些应用程序不能接受RX MAB溢出。为了防止溢出，应尽可能早地开始帧过滤和数据保存，最迟不超过接收报文的CRC字段开始的时间。更新FIFO状态必须等到EOF字段的第7位才能开始，这是因为接收帧仅在此时有效。必须保存完整的报文，并且FIFO必须更新，直到下一条报文的仲裁字段结束。

  如果RX MAB溢出，将丢弃引起溢出的新报文。模块继续存储完全接收并过滤的报文。之后，模块将能够在总线上接收新报文。将使用SERRIF通知应用程序。

  将通过向CFDxINT.SERRIF写入零来清零SERRIF。这也会将SERRIF条件从ICODE中清除。

- 处理TX MAB下溢错误：

  ISO11898-1:2015要求MAC数据具有一致性：发送报文必须包含一致的数据。如果由于ECC错误或TX MAB下溢而发生数据错误，发送将不会开始。如果发送正在进行，它将停止，模块将转换为 “受限工作”或 “仅监听”模式，可通过SERRLOM位 （CFDxCON<18>）进行选择。

 © 2019 Microchip Technology Inc.

模块通过停止发送并转换为“受限工作”或“仅监听”模式来处理这些错误。CxTX引脚将被强制为高电平。此外，所有TXREQ都将被忽略。将使用SERRIF通知应用程序。模块将继续接收报文。

56.12.5 中断处理

具有灵活数据速率的控制器局域网 （CAN FD）允许应用程序通过以下方式高效地处理中断：

- 通过CFDxVEC寄存器实现查找表。
- 使用状态寄存器并决定首先服务哪个中断。

应用程序还可以结合使用这两种方法来处理中断。

56.12.5.1 中断查找表

CFDxVEC寄存器中的ICODE和FILTHIT位允许应用程序使用查找表来实现ISR。

以下位字段允许应用程序充分利用三个中断引脚：

- TXCODE：反映哪个对象有待处理的发送中断。
- RXCODE：反映哪个对象有待处理的接收中断。

可以为发送和接收中断实现单独的查找表。

如果多个对象具有待处理的中断，则具有最高编号的中断或FIFO将显示在RXCODE、 TXCODE和ICODE中。清除具有最高优先级的中断后，下一个最高优先级中断将显示在CFDxVEC中。RXCODE、 TXCODE和ICODE将中断标志用作输入，采用组合逻辑实现。

56.12.5.2 中断状态寄存器

具有灵活数据速率的控制器局域网 （CAN FD）包含31个FIFO和1个TXQ。由于中断优先级由模块决定，所以使用ICODE会很复杂。因此，采取了以下措施来确保中断处理的高效性：

- CFDxINT包含所有主中断源。应用程序可以识别待处理中断的类别，并确定处理中断的顺序 （例如RXIF）。
- 所有FIFO的中断的所有类别组合成单个寄存器：CFDxRXIF、 CFDxTXIF、 CFDxRXOVIF和CFDxTXATIF。应用程序可以通过只读取一个寄存器来识别待处理的RFIF。 TFIF、RXOVIF和TXATIF也是如此。
- 在寄存器映射中，中断状态寄存器排列在一个块中：CFDxVEC，然后是CFDxINT、CFDxRXIF、 CFDxTXIF、 CFDxRXOVIF和CFDxTXATIF。这种排列允许通过单次读访问读取所有状态寄存器。

56.12.6 中断标志

表56-14汇总了所有中断标志，并列出了中断是如何清除的。

表 56-14： 中断标志

标志	寄存器	类别	由模块清零(1)	由应用程序清零	只读(2)	说明
RFFIF RFHIF RFNIF	CFDxFIFOSTAn	FIFO	X	—	—	RX FIFO
TFNIF TFHIF TFEIF	CFDxFIFOSTAn	FIFO	X	—	—	TX FIFO
TXQNIF TXQEIF	CFDxTXQSTA	TXQ	X	—	—	发送队列
RXOVIF	CFDxFIFOSTAn	FIFO	—	X	—	RX溢出
TXATIF	CFDxFIFOSTAn CFDxTXQSTA	FIFO TXQ	—	X	—	TX尝试
TEFFIF TEFHIF TEFNEIF	CFDxTEFSTA	FIFO	X	—	—	TEF
TEFOVIF	CFDxTEFSTA	FIFO	—	X	—	TEF溢出
RFIF	CFDxRXIF	组合	—	—	X	所有RX FIFO
TFIF	CFDxTXIF	组合	—	—	X	所有TX FIFO
RFOVIF	CFDxRXOVIF	组合	—	—	X	所有RX FIFO溢出
TFATIF	CFDxTXATIF	组合	—	—	X	所有TX FIFO尝试
RXIF	CFDxINT	主	—	—	X	RX
TXIF	CFDxINT	主	—	—	X	TX
RXOVIF	CFDxINT	主	—	—	X	RX溢出
TXATIF	CFDxINT	主	—	—	X	TX尝试
TEFIF	CFDxINT	主	—	—	X	TEF
IVMIF	CFDxINT	主	—	X	—	无效报文
WAKIF	CFDxINT	主	—	X	—	唤醒
CERRIF	CFDxINT	主	—	X	—	CAN总线错误
MODIF	CFDxINT	主	—	X	—	模式更改
TBCIF	CFDxINT	主	—	X	—	时基计数器
SERRIF	CFDxINT	主	—	X	—	系统错误

注 1：当FIFO的条件终止时，标志将清零，由CFDxFIFOCONn.UINC启动。
2：标志需要在先前的层级中清零。

 © 2019 Microchip Technology Inc.

56.13 错误处理

每个CAN控制器都会检查总线上的报文是否存在以下错误：位错误、填充错误、CRC错误、格式错误和ACK错误。只要控制器检测到错误，就会发送错误帧，从而删除总线上的报文。错误帧始终使用标称位速率发出。

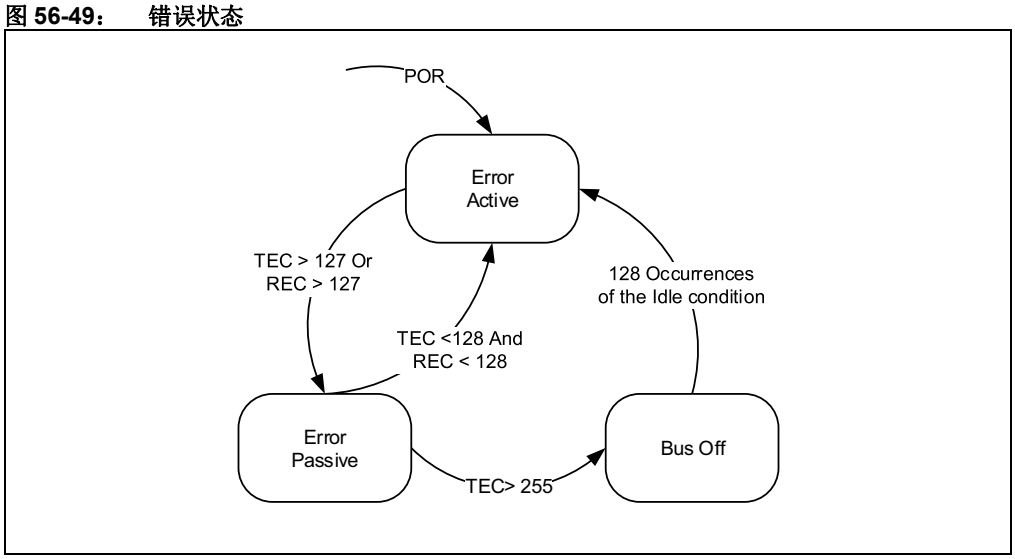
错误检测和故障界定参见ISO11898-1:2015。CFDxTREC包含错误计数器、TEC和REC。它还包含错误警告和错误状态位。 TEC和REC根据ISO11898-1:2015规范递增和递减。

图56-49展示了具有灵活数据速率的控制器局域网 （CAN FD）的不同错误状态。模块开始时为主动错误状态。如果TEC或REC超过127，模块将转换为被动错误状态。如果TEC超过255，模块将转换为离线状态。

处于主动错误状态时，模块发送主动错误帧。处于被动错误状态时，它将发送被动错误帧。当模块处于离线状态时， CxTX始终被驱动为高电平，并且不会发送任何显性位。

为避免模块转换为被动错误状态，模块会在TEC或REC达到96时通过CERRIF中断标志提醒应用程序，请参见**第56.12.4.3节 “CAN总线错误中断——CERRIF”**。这使得应用程序能够在进入被动错误状态前采取行动。

图 56-49： 错误状态



总线诊断寄存器提供更多有关CAN总线健康状况的信息：

- CFDxBDIAGn包含用于接收/发送和标称/数据位速率的独立错误计数器。计数器的工作原理与CFDxTREC寄存器中的计数器不同。每个错误都会增加 “1”。它们永远不会递减，但可以通过将 “0”写入寄存器清零。
- CFDxBDIAGn会记录上次清零寄存器后发生的错误的类型。 CFDxBDIAGn寄存器还包含无错误报文计数器。标志和计数器通过将 “0”写入寄存器清零。

可结合使用无错误报文计数器与错误计数器和错误标志来确定总线的质量。

56.13.1 从离线状态恢复

如果TEC超过255， CFDxTREC.TXBO和CFDxINT.CERRIF将置1。模块将离开离线状态并启动离线恢复序列。

离线恢复序列自动启动。只有在检测到128空闲条件后，模块才会转出离线状态，请参见“ISO11898-1:2015: Bus Off Management”。进入离线状态时，模块将把所有发送FIFO的FRESET置1，以确保模块不会无限次地尝试重发。应用程序将由CERRIF通知，并且可以选择对要发送的新报文进行排队。

模块通过 CERRIF 和将 CFDxBDIAGn.TXBOERR 置 1 发出从离线状态退出的信号。此外，CFDxTREC还将被复位。

 © 2019 Microchip Technology Inc.

56.14 相关应用笔记

本节列出了与手册本章内容相关的应用笔记。这些应用笔记可能并不是专为PIC32器件系列而编写的，但其概念是相近的，适当修改后且在一定的限制条件下可以使用。与具有灵活数据速率的控制器局域网 （CAN FD）相关的最新应用笔记包括：

标题	应用笔记编号
目前没有相关的应用笔记。	N/A

注： 如需获取更多PIC32系列器件的应用笔记和代码示例，请访问Microchip网站（www.microchip.com）。

56.15 版本历史

版本A （2018年11月）

本文档的初始版本。

 © 2019 Microchip Technology Inc.

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下， Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿与那些注重代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是 “牢不可破”的。

代码保护功能处于持续发展中。 Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了 《数字器件千年版权法案 （Digital Millennium Copyright Act）》。如果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

提供本文档的中文版本仅为了便于理解。请勿忽视文档中包含的英文部分，因为其中提供了有关 Microchip 产品性能和使用情况的有用信息。Microchip Technology Inc. 及其分公司和相关公司、各级主管与员工及事务代理机构对译文中可能存在的任何差错不承担任何责任。建议参考 Microchip Technology Inc. 的英文原版文档。

本出版物中所述的器件应用信息及其他类似内容仅为您提供便利，它们可能由更新之信息所替代。确保应用符合技术规范，是您自身应负的责任。**Microchip 对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对其使用情况、质量、性能、适销性或特定用途的适用性的声明或担保。** Microchip 对因这些信息及使用这些信息而引起的后果不承担任何责任。如果将 Microchip 器件用于生命维持和 / 或生命安全应用，一切风险由买方自负。买方同意在由此引发任何一切伤害、索赔、诉讼或费用时，会维护和保障 Microchip 免于承担法律责任，并加以赔偿。除非另外声明，在 Microchip 知识产权保护下，不得暗中或以其他方式转让任何许可证。

有关 Microchip 质量管理体系的更多信息，请访问 www.microchip.com/quality。

商标

Microchip 的名称和徽标组合、 Microchip 徽标、 Adaptec、AnyRate、AVR、AVR 徽标、AVR Freaks、BesTime、BitCloud、chipKIT、 chipKIT 徽标、 CryptoMemory、 CryptoRF、 dsPIC、FlashFlex、flexPWR、 HELDO、 IGLOO、 JukeBlox、 KeeLoq、Kleer、 LANCheck、 LinkMD、 maXStylus、 maXTouch、MediaLB、 megaAVR、 Microsemi、 Microsemi 徽标、 MOST、MOST 徽标、 MPLAB、 OptoLyzer、 PackeTime、 PIC、picoPower、 PICSTART、 PIC32 徽标、 PolarFire、 Prochip Designer、 QTouch、 SAM-BA、 SenGenuity、 SpyNIC、 SST、SST 徽标、 SuperFlash、 Symmetricom、 SyncServer、Tachyon、 TempTrackr、 TimeSource、 tinyAVR、 UNI/O、Vectron 及 XMEGA 均为 Microchip Technology Inc. 在美国和其他国家或地区的注册商标。

APT、ClockWorks、The Embedded Control Solutions Company、EtherSynch、FlashTec、Hyper Speed Control、HyperLight Load、IntelliMOS、 Libero、 motorBench、 mTouch、 Powermite 3、PrecisionEdge、ProASIC、ProASIC Plus、ProASIC Plus 徽标、Quiet-Wire、SmartFusion、SyncWorld、Temux、TimeCesium、TimeHub、TimePictra、TimeProvider、Vite、WinPath 和 ZL 均为 Microchip Technology Inc. 在美国的注册商标。

Adjacent Key Suppression、 AKS、 Analog-for-the-Digital Age、Any Capacitor、 AnyIn、 AnyOut、 BlueSky、 BodyCom、CodeGuard、 CryptoAuthentication、 CryptoAutomotive、CryptoCompanion、 CryptoController、 dsPICDEM、dsPICDEM.net、 Dynamic Average Matching、 DAM、 ECAN、EtherGREEN、 In-Circuit Serial Programming、 ICSP、 INICnet、Inter-Chip Connectivity、 JitterBlocker、 KleerNet、 KleerNet 徽标、 memBrain、 Mindi、 MiWi、 MPASM、 MPF、 MPLAB Certified 徽标、 MPLIB、 MPLINK、 MultiTRAK、 NetDetach、Omniscient Code Generation、PICDEM、PICDEM.net、PICkit、PICtail、 PowerSmart、 PureSilicon、 QMatrix、 REAL ICE、Ripple Blocker、 SAM-ICE、 Serial Quad I/O、 SMART-I.S.、SQI、 SuperSwitcher、 SuperSwitcher II、 Total Endurance、TSHARC、 USBCheck、 VariSense、 ViewSpan、 WiperLock、Wireless DNA 和 ZENA 均为 Microchip Technology Inc. 在美国和其他国家或地区的商标。

SQTP 为 Microchip Technology Inc. 在美国的服务标记。

Adaptec 徽标、Frequency on Demand、Silicon Storage Technology 和 Symmcom 为 Microchip Technology Inc. 在除美国外的国家或地区的注册商标。

GestIC 为 Microchip Technology Inc. 的子公司 Microchip Technology Germany II GmbH & Co. & KG 在除美国外的国家或地区的注册商标。

在此提及的所有其他商标均为各持有公司所有。

© 2019, Microchip Technology Inc. 版权所有。

ISBN：978-1-5224-4894-5

全球销售及服务网点

美洲

公司总部Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://www.microchip.com/support
网址：www.microchip.com

亚特兰大Atlanta
Duluth, GA
Tel: 1-678-957-9614
Fax: 1-678-957-1455

奥斯汀Austin, TX
Tel: 1-512-257-3370

波士顿Boston
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

达拉斯Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律Detroit
Novi, MI
Tel: 1-248-848-4000

休斯敦Houston, TX
Tel: 1-281-894-5983

印第安纳波利斯Indianapolis
Noblesville, IN
Tel: 1-317-773-8323
Fax: 1-317-773-5453
Tel: 1-317-536-2380

洛杉矶Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608
Tel: 1-951-273-7800

罗利Raleigh, NC
Tel: 1-919-844-7510

纽约New York, NY
Tel: 1-631-435-6000

圣何塞San Jose, CA
Tel: 1-408-735-9110
Tel: 1-408-436-4270

加拿大多伦多Toronto
Tel: 1-905-695-1980
Fax: 1-905-695-2078

亚太地区

中国-北京
Tel: 86-10-8569-7000

中国-成都
Tel: 86-28-8665-5511

中国-重庆
Tel: 86-23-8980-9588

中国-东莞
Tel: 86-769-8702-9880

中国-广州
Tel: 86-20-8755-8029

中国-杭州
Tel: 86-571-8792-8115

中国-南京
Tel: 86-25-8473-2460

中国-青岛
Tel: 86-532-8502-7355

中国-上海
Tel: 86-21-3326-8000

中国-沈阳
Tel: 86-24-2334-2829

中国-深圳
Tel: 86-755-8864-2200

中国-苏州
Tel: 86-186-6233-1526

中国-武汉
Tel: 86-27-5980-5300

中国-西安
Tel: 86-29-8833-7252

中国-厦门
Tel: 86-592-238-8138

中国-香港特别行政区
Tel: 852-2943-5100

中国-珠海
Tel: 86-756-321-0040

台湾地区-高雄
Tel: 886-7-213-7830

台湾地区-台北
Tel: 886-2-2508-8600

台湾地区-新竹
Tel: 886-3-577-8366

亚太地区

澳大利亚Australia - Sydney
Tel: 61-2-9868-6733

印度India - Bangalore
Tel: 91-80-3090-4444

印度India - New Delhi
Tel: 91-11-4160-8631

印度India - Pune
Tel: 91-20-4121-0141

日本Japan - Osaka
Tel: 81-6-6152-7160

日本Japan - Tokyo
Tel: 81-3-6880-3770

韩国Korea - Daegu
Tel: 82-53-744-4301

韩国Korea - Seoul
Tel: 82-2-554-7200

马来西亚
Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

马来西亚Malaysia - Penang
Tel: 60-4-227-8870

菲律宾Philippines - Manila
Tel: 63-2-634-9065

新加坡Singapore
Tel: 65-6334-8870

泰国Thailand - Bangkok
Tel: 66-2-694-1351

越南Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

欧洲

奥地利Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

丹麦
Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

芬兰Finland - Espoo
Tel: 358-9-4520-820

法国France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国Germany - Garching
Tel: 49-8931-9700

德国Germany - Haan
Tel: 49-2129-3766400

德国Germany - Heilbronn
Tel: 49-7131-72400

德国Germany - Karlsruhe
Tel: 49-721-625370

德国Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

德国Germany - Rosenheim
Tel: 49-8031-354-560

以色列Israel - Ra'anana
Tel: 972-9-744-7705

意大利Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

意大利Italy - Padova
Tel: 39-049-7625286

荷兰Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

挪威Norway - Trondheim
Tel: 47-7288-4388

波兰Poland - Warsaw
Tel: 48-22-3325737

罗马尼亚
Romania - Bucharest
Tel: 40-21-407-87-50

西班牙Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

瑞典Sweden - Gothenberg
Tel: 46-31-704-60-40

瑞典Sweden - Stockholm
Tel: 46-8-5090-4654

英国UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

© 2019 Microchip Technology Inc.