LPC1100 系列微控制器

第十三章 C_CAN

用户手册 Rev.1.00

广州周立功单片机发展有限公司

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4

网址：http://www.zlgmcu.com

销售与服务网络（一）

广州周立功单片机发展有限公司

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4
邮编：510630
电话：(020)38730916　38730917　38730972　38730976　38730977
传真：(020)38730925
网址：www.zlgmcu.com

广州专卖店

地址：广州市天河区新赛格电子城 203-204 室
电话：(020)87578634　87569917
传真：(020)87578842

北京周立功

地址：北京市海淀区知春路 113 号银网中心 A 座 1207-1208 室（中发电子市场斜对面）
电话：(010)62536178　62536179　82628073
传真：(010)82614433

杭州周立功

地址：杭州市天目山路 217 号江南电子大厦 502 室
电话：(0571)89719480　89719481　89719482
89719483　89719484　89719485
传真：(0571)89719494

深圳周立功

地址：深圳市深南中路 2070 号电子科技大厦 C 座 4 楼 D 室
电话：(0755)83781788（5 线）
传真：(0755)83793285

上海周立功

地址：上海市北京东路 668 号科技京城东座 7E 室
电话：(021)53083452　53083453　53083496
传真：(021)53083491

南京周立功

地址：南京市珠江路 280 号珠江大厦 1501 室
电话：(025) 68123901　68123902
传真：(025) 68123900

重庆周立功

地址：重庆市石桥铺科园一路二号大西洋国际大厦（赛格电子市场）1611 室
电话：(023)68796438　68796439
传真：(023)68796439

成都周立功

地址：成都市一环路南二段 1 号数码科技大厦 403 室
电话：(028)85439836　85437446
传真：(028)85437896

武汉周立功

地址：武汉市洪山区广埠屯珞瑜路 158 号 12128 室（华中电脑数码市场）
电话：(027)87168497　87168297　87168397
传真：(027)87163755

西安办事处

地址：西安市长安北路 54 号太平洋大厦 1201 室
电话：(029)87881296　83063000　87881295
传真：(029)87880865

销售与服务网络（二）

广州致远电子有限公司

地址：广州市天河区车陂路黄洲工业区 3 栋 2 楼
邮编：510660
传真：(020)38601859
网址：www.embedtools.com （嵌入式系统事业部）
www.embedcontrol.com （工控网络事业部）
www.ecardsys.com （楼宇自动化事业部）

技术支持：

CAN-bus：

电话：(020)22644381 22644382 22644253
邮箱：can.support@embedcontrol.com

iCAN 及数据采集：

电话：(020)28872344 22644373
邮箱：ican@embedcontrol.com

MiniARM：

电话：(020)28872684 28267813
邮箱：miniarm.support@embedtools.com

以太网：

电话：(020)22644380 22644385
邮箱：ethernet.support@embedcontrol.com

无线通讯：

电话：(020) 22644386
邮箱：wireless@embedcontrol.com

串行通讯：

电话：(020)28267800 22644385
邮箱：serial@embedcontrol.com

编程器：

电话：(020)22644371
邮箱：programmer@embedtools.com

分析仪器：

电话：(020)22644375
邮箱：tools@embedtools.com

ARM 嵌入式系统：

电话：(020) 22644383 22644384
邮箱：NXPARM@zlgmcu.com

楼宇自动化：

电话：(020)22644376 22644389 28267806
邮箱：mjs.support@ecardsys.com
mifare.support@zlgmcu.com

销售：

电话：(020)22644249 22644399 22644372 22644261 28872524
28872342 28872349 28872569 28872573 38601786

维修：

电话：(020)22644245

目　录

第13章 C_CAN

13.1 本章导读

只有 LPC11Cxx（LPC11C00 系列）器件才具有 C_CAN 模块。

LPC11C22 和 LPC11C24 集成了片上高速 CAN 收发器，其 CAN_TXD 和 CAN_RXD 引脚内部已连接到片上 CAN 收发器，并且收发器的引脚已引出（见
表 13.2）。

13.2 基本配置

C_CAN 使用下列寄存器进行配置：

1）电源：在 SYSAHBCLKCTRL 寄存中设置位 17（“系统 AHB 时钟控制寄存器位描述”表）。

2）复位：在访问 C_CAN 模块之前，确保将 PRESETCTRL 寄存器（“外设复位控制寄存器位描述”表）的 CAN_RST_N 位（位 3）设为 1。这会令 C_CAN 模块的复位信号无效。

13.3 特性

- 遵从 2.0 版本协议的 A 和 B 部分；
- 支持的位速率可高达 1Mbit/s；
- 支持 32 个报文对象；
- 每个报文对象具有自身拥有的标识符屏蔽；
- 提供可编程的 FIFO 模式（链接报文对象）；
- 提供可屏蔽的中断；
- 对于定时触发的 CAN 应用程序，提供禁能的自动重发（DAR）模式；
- 提供可编程的环回模式（loop-back mode）来进行自我测试操作。

13.4 概述

控制器局域网络（CAN）的定义是高性能的通信协议，用于执行串行数据的通信。依据 2.0B 版本的 CAN 规范，C_CAN 控制器被设计的目的是用来完全执行 CAN 协议。C_CAN 控制器可以以低成本的多路复用接线网络建立强大的本地网络，这通过分配实时控制来实现，同时并具有极高的安全性。

CAN 控制器由 CAN 内核、报文 RAM、报文处理程序、控制寄存器和 APB 接口组成。

对于 CAN 网络中的通信，要单独配置各个报文对象。要对所接收到的报文进行接收过滤的报文对象和标识符屏蔽是存放在报文 RAM 中。

所有关于报文处理的函数均由报文处理程序执行。这些函数是接收过滤、CAN 内核和报文 RAM 之间的报文传输、处理发送请求和产生模块中断的函数。

CAN 控制器的寄存器集可由外部 CPU 通过 APB 总线来进行直接访问。这些寄存器用来控制/配置 CAN 内核和报文处理程序，以及访问报文 RAM。

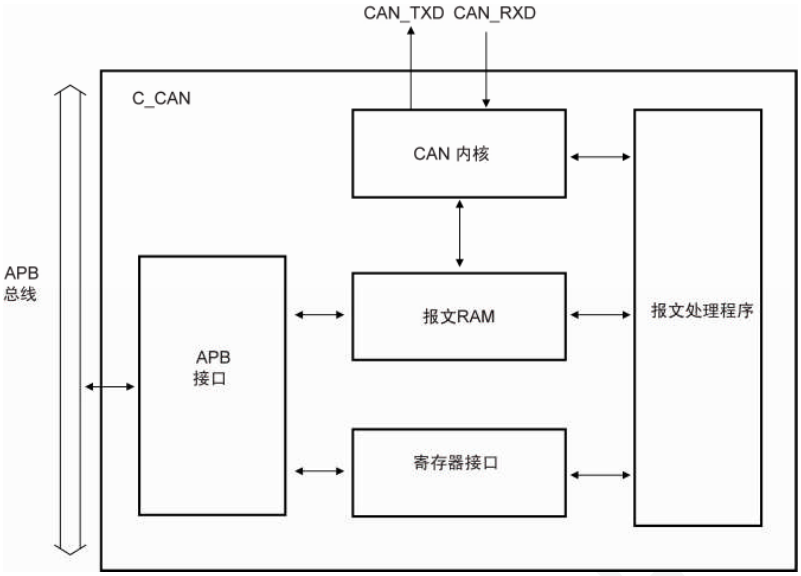

图 13.1 C_CAN 方框图

13.5 引脚描述

表 13.1 CAN 引脚描述（LPC11C12/C14）

引脚	类型	描述
CAN_TXD	O	C_CAN 发送输出
CAN_RXD	I	C_CAN 接收输入

表 13.2 CAN 引脚描述（LPC11C22/C24）

引脚	类型	描述
CANL	I/O	CAN 总线低电平
CANH	I/O	CAN 总线高电平
STB	I	CAN 收发器静默模式输入控制（0：正常模式控制输入；1：静默模式控制输入）
VDD_CAN	-	CAN 收发器 I/O 电压
V_{CC}	-	CAN 收发器电源
GND	-	CAN 收发器地

13.6 时钟和电源控制

C_CAN（C_CAN 系统时钟）和可编程的 C_CAN 时钟分频器（见表 13.33）的外设时钟由系统时钟提供（见“系统 AHB 时钟控制寄存器位描述”表）。该时钟可由 SYSAHBCLKCTRL 寄存器的位 17 来禁能，以节省功率。

13.7 复位和中断配置

C_CAN 控制器的复位可由系统控制模块 PRESETCTRL 寄存器（地址 0x4004 8004）的位 3 来实现。

C_CAN 中断接入 NVIC 是通过使用异常编号 13 来实现。

13.8 寄存器描述

C_CAN 寄存器是 32 位宽的寄存器。

二组接口寄存器（(IF1 和 IF2）控制 CPU 访问报文 RAM。它们将往返 RAM 中的数据进行缓冲，避免在 CPU 访问和报文接收/发送之间造成冲突。

表 13.3　寄存器汇总：CCAN（基址 0x4005 0000）

名称	访问	地址偏移量	描述	复位值
CANCNTL		0x000	CAN 控制	0x0001
CANSTAT		0x004	状态寄存器	0x0000
CANEC	RO	0x008	错误计数器	0x0000
CANBT		0x00C	位定时寄存器	0x2301
CANINT	RO	0x010	中断寄存器	0x0000
CANTEST		0x014	测试寄存器	-
CANBRPE		0x018	波特率预分频器扩展寄存器	0x0000
-	-	0x01C	保留	-
CANIF1_CMDREQ		0x020	报文接口 1 命令请求	0x0001
CANIF1_CMDMSK		0x024	报文接口 1 命令屏蔽	0x0000
CANIF1_MSK1		0x028	报文接口 1 屏蔽 1	0xFFFF
CANIF1_MSK2		0x02C	报文接口 1 屏蔽 2	0xFFFF
CANIF1_ARB1		0x030	报文接口 1 仲裁 1	0x0000
CANIF1_ARB2		0x034	报文接口 1 仲裁 2	0x0000
CANIF1_MCTRL		0x038	报文接口 1 报文控制	0x0000
CANIF1_DA1		0x03C	报文接口 1 数据 A1	0x0000
CANIF1_DA2		0x040	报文接口 1 数据 A2	0x0000
CANIF1_DB1		0x044	报文接口 1 数据 B1	0x0000
CANIF1_DB2		0x048	报文接口 1 数据 B2	0x0000
-		0x04C～0x07C	保留	
CANIF2_CMDREQ		0x080	报文接口 2 命令请求	0x0001
CANIF2_CMDMSK		0x084	报文接口 2 命令屏蔽	0x0000
CANIF2_MSK1		0x088	报文接口 2 屏蔽 1	0xFFFF
CANIF2_MSK2		0x08C	报文接口 2 屏蔽 2	0xFFFF
CANIF2_ARB1		0x090	报文接口 2 仲裁 1	0x0000
CANIF2_ARB2		0x094	报文接口 2 仲裁 2	0x0000
CANIF2_MCTRL		0x098	报文接口 2 报文控制	0x0000
CANIF2_DA1		0x09C	报文接口 2 数据 A1	0x0000
CANIF2_DA2		0x0A0	报文接口 2 数据 A2	0x0000
CANIF2_DB1		0x0A4	报文接口 2 数据 B1	0x0000
CANIF2_DB2		0x0A8	报文接口 2 数据 B2	0x0000
-	-	0x0AC～0x0FC		
CANTXREQ1	RO	0x100	发送请求 1	0x0000
CANTXREQ2	RO	0x104	发送请求 2	0x0000
-	-	0x108～0x11C	保留	-

续上表

名称	访问	地址偏移量	描述	复位值
CANND1	RO	0x120	新数据 1	0x0000
CANND2	RO	0x124	新数据 2	0x0000
-	-	0x128～0x13C	保留	-
CANIR1	RO	0x140	中断挂起 1	0x0000
CANIR2	RO	0x144	中断挂起 2	0x0000
-	-	0x148～0x15C	保留	-
CANMSGV1	RO	0x160	报文有效 1	0x0000
CANMSGV2	RO	0x164	报文有效 2	0x0000
-	-	0x168～0x17C	保留	
CANCLKDIV	R/W	0x180	CAN 时钟分频器寄存器	0x0001

13.8.1 CAN 协议寄存器

1. CAN 控制寄存器

CANCTRL 寄存器的复位值 0x0001 可令软件进行初始化（INIT=1）。C_CAN 不会影响 CAN 总线，直至 CPU 将 INIT 位复位为 0。

表 13.4 CAN 控制寄存器（CANCNTL，地址 0x4005 0000）位描述

位	符号	值	描述	复位值	访问
0	INIT		初始化	1	R/W
		0	正常操作		
		1	启动初始化。复位时，软件需要初始化 CAN 控制器		
1	IE		模块中断使能	0	R/W
		0	禁能 CAN 中断。中断线总是为高电平		
		1	使能 CAN 中断。中断线被设为低电平，并保持为低电平，直到所有挂起的中断被清除		
2	SIE		状态更改中断使能	0	R/W
		0	禁能状态更改中断。不会产生状态更改中断		
		1	使能状态更改中断。当成功完成报文传输或检测到 CAN 总线错误时，产生状态更改中断		
3	EIE		错误中断使能	0	R/W
		0	禁能错误中断。不会产生错误状态中断		
		1	使能错误中断。CANSTAT 寄存器的 BOFF 或 EWARN 位发生改变时会产生中断		
4	-	-	保留	0	-
5	DAR		禁能自动重发	0	R/W
		0	使能被干扰报文的自动重发		
		1	禁能自动重发		

续上表

位	符号	值	描述	复位值	访问
6	CCE		配置更改使能	0	R/W
		0	CPU 不对位定时寄存器进行写访问		
		1	CPU 会在 INIT 位为 1 时对 CANBT 寄存器进行写访问		
7	TEST		测试模式使能	0	R/W
		0	正常操作		
		1	测试模式		
31:8	-		保留	-	-

注：总线关闭恢复序列（见 2.0 版本的 CAN 规范）不可以通过设置或复位 INIT 位来变短。如果设备进入总线关闭状态，它将会设置 INIT，停止所有的总线活动。一旦 CPU 清除了 INIT，设备将会先等待 129 个总线空闲发生（129×11 个连续高电平/隐性位），然后再恢复正常操作。在结束总线关闭恢复序列时，错误管理计数器将会复位。

在复位 INT 后的等待时间内，每次都监控到 11 个高电平/隐性位的序列，而 Bit0Error 代码被写入到状态寄存器 CANSTAT 中，使得 CPU 可以监控总线关闭恢复序列的过程，从而决定是否令 CAN 总线一直处于低电平/显性或连续被干扰。

2. CAN 状态寄存器

BOFF、EWARN、RXOK、TXOK、或 LEC 位可以产生状态中断。BOFF 和 EWARN 产生错误中断，RXOK、TXOK 和 LEC 产生状态更改中断，如果 EIE 和 SIE 分别在 CANCTRL 寄存器里被使能。

EPASS 位发生改变，写 RXOK、TXOK 或 LEC 将永不会产生状态中断。

读 CANSTAT 寄存器将会在 CANIR 寄存器中清除状态中断值（0x8000）。

表 13.5 CAN 状态寄存器（CANSTAT，地址 0x4005 0004）位描述

位	符号	值	描述	复位值	访问
2:0	LEC		最近错误代码 出现在 CAN 总线上的最近错误类型。LEC 域保存着表示最近出现在 CAN 总线上的错误类型的代码。当无错误地完成报文传输时，该域将会被清除为‘0’。未用到的代码为‘111’，CPU 可以写入这个代码来检查更新	000	R/W
		000	**无错误**		
		001	**Stuff error**：在接收到报文里，序列中存在着超过 5 个相同的位，在这里是不允许发生的		
		010	**Form error**：接收帧的固定格式部分的格式错误		
		011	**AckError**：该 CAN 内核传送的报文不被应答		
		100	**Bit1Error**：在报文传送期间（仲裁域除外），设备想要发送高电平/隐性电平（位逻辑值为‘1’），但是监控到的总线为低电平/显性电平		

续上表

位	符号	值	描述	复位值	访问
2:0	LEC	101	**Bit0Error**：在报文的传送期间（或应答位、或有效错误标志、或过载标志），设备想要发送低电平/显性电平（数据或标识符位逻辑位值为‘0’），但是监控到的总线值是高电平/隐性电平。在总线关闭恢复期间，在每次监控到 11 个高电平/隐性位的序列时，要将该状态进行设置。这可令 CPU 监控总线关闭恢复序列的进程（表示总线不处于低电平/显性电平或连续被干扰）	000	R/W
		110	**CRCError**：所接到的报文中的 CRC 校验和不正确		
		111	**Unused**：不检测到 CAN 总线事件（由 CPU 写入）		
3	TXOK		成功发送报文 该位由 CPU 复位。CAN 控制器不能将它复位	0	R/W
		0	由于该位由 CPU 复位，因此报文没有成功发送		
		1	由于该位由 CPU 最后复位，因此报文成功发送（无错误和至少被另一个节点进行应答）		
4	RXOK		成功接收报文 该位由 CPU 复位。CAN 控制器不能将它复位	0	R/W
		0	由于该位由 CPU 最后复位，因此报文没有成功发送		
		1	由于该位由 CPU 最后设置为 0，因此报文接收成功，与接收过滤的结果无关		
5	EPASS		错误消极	0	RO
		0	CAN 控制器处于错误有效状态		
		1	CAN 控制器处于 CAN2.0 规范中所定义的错误消极状态		
6	EWARN		警告状态	0	RO
		0	二个错误计数器的错误警告数低于 96 这个限数		
		1	EML 中至少有一个错误计数器的数值达到了 96 个错误警告上限		
7	BOFF		总线关闭状态	0	RO
		0	CAN 模块不处于总线关闭状态		
		1	CAN 控制器处于总线关闭状态		
31:8	-	-	保留		

3. CAN 错误计数器

表 13.6 CAN 错误计数器（CANEC，地址 0x4005 0008）位描述

位	符号	值	描述	复位值	访问
7:0	TEC[7:0]		发送错误计数器 发送错误计数器的当前值（最大值为 127）	0	RO
14:8	REC[14:8]		接收错误计数器 接收错误计数器的当前值（最大值为 255）	-	RO
15	RP		接收错误消极	-	RO
		0	接收计数器没有达到错误消极状态		
		1	接收计数器达到了在 CAN2.0 规范中所定义的错误消极状态		
31:16	-	-	保留	-	-

4. CAN 位定时寄存器

表 13.7 CAN 位定时寄存器（CANBT，地址 0x4005 000C）位描述

位	符号	值	描述	复位值	访问
5:0	BRP	0x00～0x3F[1]	波特率预分频器 振荡器频率被分频的值是用于产生位时间量子。位时间值是该量子的整数倍，波特率预分频的有效值为 0～63	000001	R/W
7:6	SJW	0x00～0x03[1]	（重新）同步跳转宽度 有效的编程值为 0～3	00	R/W
11:8	TSEG1	0x01～0x0F[1]	采样点之后的时间段 有效值为 1～15	0011	R/W
14:12	TSEG2	0x00～0x07[1]	采样点之前的时间段 有效值为 0～7	010	R/W
31:15	-	-	保留	-	-

[1] 硬件把往这些位编写入的值理解为位值+1。

例如，将模块时钟 CAN_CLK LPC11Cx 系列 Cortex-M0 微控制器系统时钟设为 8MHz，0x2301 的复位值将 C_CAN 配置，以进行位速率为 500KBit/s 的操作。

如果 CANCTRL 中的配置更改使能，且软件初始化了控制器（CAN 控制寄存器中的 CCE 和 INIT 位被置位），则寄存器都为只写寄存器。

有关位定时的详细描述，请参考“位定时”小节和 1.2 版本的 Bosch C_CAN 用户手册。

波特率预分频器

位时间量子 t_q 由 BRP 值决定：

$t_q = BRP / f_{sys}$

（f_{sys} 是 C_CAN 模块的 LPC11Cx 系列 Cortex-M0 微控制器系统时钟）

时间段 1 和 2

时间段 TSEG1 和 TSEG2 决定每位时间的时间量子数量和采样点的位置：

$t_{TSEG1/2} = t_q \times (TSEG1/2 + 1)$

同步跳转宽度

为了向不同总线控制器的时钟振荡器之间的相位漂移进行补偿，任何总线控制器必须在当前传送的相关信号边沿上重新同步。同步跳转宽度 t_{SJW} 定义时钟周期的最大数目，执行一次重新同步操作时，某位的周期可能变短或变长：

$$t_{SJW} = t_q \times (SJW + 1)$$

5. CAN 中断寄存器

表 13.8 CAN 中断寄存器（CANINT，地址 0x4005 0010）位描述

位	符号	值	描述	复位值	访问
15:0	INTID[15:0]	0x0000	无中断挂起		R
		0x0001～0x0020	引发中断的报文对象编号		
		0x0021～0x7FFF	未使用		
		0x8000	状态中断		
		0x8001～0xFFFF	未使用		
31:16	-	-	保留	-	-

如果有几个中断被挂起，CAN 中断寄存器则会指向具有最高优先级的挂起中断，而忽略它们的时间排序。中断仍保持挂起状态，直至 CPU 将其清除。如果 INTID 不同于 0x0000，且 IE 被置位，则到 CPU 的中断线有效。中断线保持有效直至 INTID 返回到 0x0000 值（原因是中断被复位）或直至 IE 被复位。

状态中断具有最高的优先级。在这些报文中断里，报文对象的中断优先级是随着报文编号的增加而递减。

通地清除报文对象的 INTPND 位，即可以清除报文中断。而状态中断的清除则通过读取状态寄存器来完成。

6. CAN 测试寄存器

通过设置 CAN 控制寄存器的测试位，即可使能对测试寄存器的写访问。

用户可以组合出不同的测试功能，但是，当选择了 TX[1:0]=“00”时，报文传输被干扰。

表 13.9 CAN 测试寄存器（CANTEST，地址 0x4005 0014）位描述

位	符号	值	描述	复位值	访问
1:0	-	-			-
2	BASIC		基本模式	0	R/W
		0	基本模式禁能		
		1	IF1 寄存器用作 TX 缓冲区，IF2 寄存器用作 RX 缓冲区		
3	SILENT		安静模式	0	R/W
		0	正常操作		
		1	模块处于静默模式		
4	LBACK		环回模式	0	R/W
		0	禁能环回模式		
		1	使能环回模式		

续上表

位	符号	值	描述	复位值	访问
6:5	TX[1:0]		TD 引脚的控制	00	R/W
		00	TD 引脚上的电平由 CAN 控制器控制。这是复位时的值		
		01	可在 TD 引脚上监控采样点		
		10	TD 引脚被驱动为低电平/显性		
		11	TD 引脚被驱动为高电平/隐性		
7	RX		CAN 总线为隐性电平（CAN_RXD=‘1’）	0	R
		0	TD 引脚被驱动为低电平/显性		
		1	CAN 总线为显性（CAN_RXD=‘0’）		
31:8	-		R/W		-

7. CAN 波特率预分频器扩展寄存器

表 13.10 CAN 波特率预分频器扩展寄存器（CANBRP，地址 0x4005 8018）位描述

位	符号	值	描述	复位值	访问
3:0	BRPE	0x00～0x0F	波特率预分频器扩展 通过编程 BRPE，波特率预分频器的值可以扩展至 1023。硬件将这个值理解为 BRPE（高位）和 BRP（低位）的值加 1	0x0000	R/W
31:4	-	-	保留	-	-

13.8.2 报文接口寄存器

二组接口寄存器用于控制 CPU 对报文 RAM 的访问。接口寄存器通过将传送的数据进行缓冲，以避免 CPU 对报文 RAM 和 CAN 报文收发之间的访问造成冲突。在单次传输中，完整的报文对象（见“报文对象”小节）或部分报文对象可以在报文 RAM 和 IFx 报文缓冲区寄存器之间传送。

二组接口寄存器的功能相同（测试模式基本配置除外）。一组寄存器用于将数据传送到报文 RAM 中，另一组寄存器报文 RAM 中的数据进行传送，从而允许这二个进程可由对方中断。

每组接口寄存器由报文缓冲区寄存器组成，报文缓冲区寄存器由各自的命令寄存器控制。命令屏蔽寄存器指定数据传送的方向，和要传送哪一部分报文对象。命令请求寄存器在报文 RAM 选择一个报文对象，以作为传输的目标对象或源对象，同时它也可以启动在命令屏蔽寄存器中指定的操作。

表 13.11 报文接口寄存器

IF1 寄存器名称	IF1 寄存器组	IF2 寄存器名称	IF2 寄存器组
CANIF1_CMDREQ	IF1 命令请求	CANIF2_CMDREQ	IF2 命令请求
CANIF1_CMDMASK	IF1 命令屏蔽	CANIF2_CMDMASK	IF2 命令屏蔽
CANIF1_MASK1	IF1 屏蔽 1	CANIF2_MSK1	IF2 屏蔽 1
CANIF1_MASK2	IF1 屏蔽 2	CANIF2_MSK2	IF2 屏蔽 2
CANIF1_ARB1	IF1 仲裁 1	CANIF2_ARB1	IF2 仲裁 1
CANIF1_ARB2	IF1 仲裁 2	CANIF2_ARB2	IF2 仲裁 2
CANIF1_MCTRL	IF1 报文控制	CANIF2_MCTRL	IF2 报文控制
CANIF1_DA1	IF1 数据 A1	CANIF2_DA1	IF2 数据 A1
CANIF1_DA2	IF1 数据 A2	CANIF2_DA2	IF2 数据 A2
CANIF1_DB1	IF1 数据 B1	CANIF2_DB1	IF1 数据 B1
CANIF1_DB2	IF1 数据 B2	CANIF2_DB2	IF1 数据 B2

在报文 RAM 中有 32 个报文对象。为了避免 CPU 对报文 RAM 和 CAN 报文接收和发送之间的访问造成冲突，CPU 不可以直接访问报文对象。通过 IFx 接口寄存器可以访问报文对象。

有关报文处理的详细描述，请参考“CAN 报文处理程序”小节。

1. 报文对象

报文对象包含着来自于报文接口寄存器中各个位的信息。表 13.12 所示为报文对象的结构图示。报文对象和相关的接口寄存器的位里，显示了位被设置或清零的情况。有关位功能的描述，请参考相关的接口寄存器。

表 13.12 报文 RAM 中报文对象的结构

UMASK	MSK[28:0]	MXTD	MDIR	EOB	NEWDAT	MSGLST	RXIE	TXIE	INTPND
IF1/2_MCTRL	IF1/2_MSK1/2			IF1/2_MCTRL					
RMTEN	TXRQST	MSGVAL	ID[28:0]	XTD	DIR	DLC3	DLC2	DLC1	DLC0
IF1/2_MCTRL		IF1/2_ARB1/2				IF1/2_MCTRL			
DATA0	DATA1	DATA2	DATA3	DATA4	DATA5	DATA6	DATA7		
IF1/2_DA1		IF1/2_DA2		IF1/2_DB1		IF1/2_DB2			

2. CAN 报文接口命令请求寄存器

一旦 CPU 将报文编号写入到命令请求寄存器，报文传输启动。随着写操作的进行，BUSY 位被自动设置为‘1’，信号 CAN_WAIT_B 被拉低来向 CPU 通告正在处理传输。在等待 3～6 个 CAN_CLK 周期后，接口寄存器和报文 RAM 之间的传输完成。BUSY 位被设回为 0，且信号 CAN_WAIT_B 也被设回原状态。

表 13.13 CAN 报文接口命令请求寄存器（CANIF1_CMDREQ 和 CANIF2_CMDREQ，地址 0x4005 0020 和 0x4005 0080）位描述

位	符号	值	描述	复位值	访问
5:0	Message Number		报文编号	0x00	R/W
		0x01～0x20	有效的报文编号 报文 RAM 中的报文对象，被选用于执行数据传输		
		0x00	无效的报文编号。该值被看为 0x20[1]		
		0x21～0x3F	无效的报文编号。该值被看为 0x01～0x1F [1]		
14:6	-		保留	-	-
15	BUSY		BUSY 标志	0	RO
		0	当对该命令请求寄存器的读/写操作完成时，硬件将其置为 0		
		1	当写该命令请求寄存器时，硬件将其置为 1		
31:16	-	-	保留	-	-

[1] 当向命令请求寄存器写入无效的报文编号时，报文编号将会变为一个有效的值，并传送此报文对象。

3. CAN 报文接口命令屏蔽寄存器

IFx 命令屏蔽寄存器的控制位指定传输的方向，并选择出将哪一个 IFx 报文缓冲区寄存器用作进行数据传输的源寄存器或是目标寄存器。寄存器位的功能由传输的方向（读或写）决定，方向由该命令屏蔽寄存器的 WR/RD 位（位 7）进行选择。

WR/RD 的选择情况如下:

- 如果为 1，则为写传输方向（写报文 RAM）;
- 如果为 0，则为读传输方向（读报文 RAM）。

表 13.14 CAN 报文接口命令屏蔽寄存器（CANIF1_CMDMASK 和 CANIF2_CMDMASK，地址 0x4005 0024 和 0x4005 0084)位描述-写方向

位	符号	值	描述	复位值	访问
0	DATA_B		访问数据字节 4～7	0	R/W
		0	数据字节 4～7 不变		
		1	将数据字节 4～7 传送到报文对象		
1	DATA_A		访问数据字节 0～3	0	R/W
		0	数据字节 0～3 不变		
		1	将数据字节 0～3 传送到报文对象		
2	TXRQST		访问传送请求位	0	R/W
		0	无传送请求。IF1/2_MCTRL 中的 TXRQSRT 位不变 注：如果是通过编程该位来请求进行传送，则要忽略 CANIFn_MCTRL 寄存器中的 TXRQST 位		
		1	请求传送。将 IF1/2_MCTRL 的 TXRQST 位置位		
3	CLRINTPND	-	该位在写方向的操作里被忽略	0	R/W
4	CTRL		访问控制位	0	R/W
		0	控制位不改变		
		1	将控制位传送到报文对象		

续上表

位	符号	值	描述	复位值	访问
5	ARB		访问仲裁位	0	R/W
		0	仲裁位不改变		
		1	将标识符、DIR、XTD、和 MSGVAL 位传送到报文对象里		
6	MASK		访问屏蔽位	0	R/W
		0	屏蔽位不改变		
		1	将 MASK+MDIR+MXTD 标识符传送到报文对象		
7	WR/RD	1	写传输 将所选报文缓冲区寄存器的数据传送到命令请求寄存器 CANIFn_CMDREQ 所寻址的报文对象	0	R/W
31:8	-	-	保留	0	-

表 13.15 CAN 报文接口命令屏蔽寄存器（CANIF1_CMDMASK 和 CANIF2_CMDMASK，地址 0x4005 0024 和 0x4005 0084)位描述-读方向

位	符号	值	描述	复位值	访问
0	DATA_B		访问数据字节 4～7	0	R/W
		0	数据字节 4～7 不变		
		1	将数据字节 4～7 传送到 IFx 报文缓冲区寄存器		
1	DATA_A		访问数据字节 0～3	0	R/W
		0	数据字节 0～3 不变		
		1	将数据字节 0～3 传送到 IFx 报文缓冲区		
2	NEWDAT		访问新数据位	0	R/W
		0	NEWDAT 位保持不变 注：读访问报文对象时可以将 IF1/2_MCTRL 寄存器的 INTPND 和 NEWDAT 控制位复位操作整合起来。这些被传送到 IFx 报文控制寄存器的位值总是反映着这些位被复位之前的状态		
		1	清除报文对象里的 NEWDAT 位		
3	CLRINTPND		清除中断挂起位	0	R/W
		0	INTPND 位保持不变		
		1	清除报文对象的 INTPND 位		
4	CTRL		访问控制位	0	R/W
		0	控制位不改变		
		1	将控制位传送到 IFx 报文缓冲区		
5	ARB		访问仲裁位	0	R/W
		0	仲裁位不改变		
		1	将标识符、DIR、XTD、和 MSGVAL 位传送到 IFx 报文缓冲区寄存器		

续上表

位	符号	值	描述	复位值	访问
6	MASK		访问屏蔽位	0	R/W
		0	屏蔽位不改变		
		1	将 MASK+MDIR+MXTD 标识符传送到 IFx 报文缓冲区寄存器		
7	WR/RD	0	读传输 将命令请求寄存器所寻址的报文对象的数据传送到所选的报文缓冲区寄存器 CANIFn_CMDREQ	0	R/W
31:8	-	-	保留	0	-

4. IF1 和 IF2 报文缓冲区寄存器

报文缓冲区寄存器的位反映报文 RAM 中的报文对象。

（1）CAN 报文接口命令屏蔽 1 寄存器

表 13.16 CAN 报文接口命令屏蔽 1 寄存器（CANIF1_MASK1 和 CANIF2_MASK1，地址 0x4005 0028 和 0x4005 0088)位描述

位	符号	值	描述	复位值	访问
15:0	MSK[15:0]		标识符屏蔽	0xFF	R/W
		0	报文标识符的相应位不能禁止接收过滤里的匹配操作		
		1	相应的标识符位用于接收过滤操作		
31:16	-	-	保留	0	-

（2）CAN 报文接口命令屏蔽 2 寄存器

表 13.17 CAN 报文接口命令屏蔽 2 寄存器（CANIF1_MASK2 和 CANIF2_MASK2，地址 0x4005 002C 和 0x4005 008C)位描述

位	符号	值	描述	复位值	访问
12:0	MSK[28:16]		标识符屏蔽	0xFF	R/W
		0	报文标识符的相应位不能禁止接收过滤里的匹配操作		
		1	相应的标识符位用于接收过滤操作		
-	-	-	保留	-	-
14	MDIR		屏蔽报文方向	1	R/W
		0	报文方向位（DIR）不会影响接收过滤操作		
		1	报文方向位（DIR）用于接收过滤操作		
15	MXTD		屏蔽扩展标识符	1	R/W
		0	扩展的标识符位（IDE）不会影响接收过滤操作		
		1	扩展的标识符位（IDE）用于接收过滤操作		
31:16	-	-	保留	0	-

（3）CAN 报文接口命令仲裁 1 寄存器

表 13.18 CAN 报文接口命令仲裁 1 寄存器（CANIF1_ARB1 和 CANIF2_ARB1，地址 0x4005 0030 和 0x4005 0090）位描述

位	符号	值	描述	复位值	访问
15:0	ID[15:0]		报文标识符 29 位标识符（“扩展的帧”） 11 位标识符（“标准的帧”）	0x00	R/W
31:16	-	-	保留	0	-

（4）CAN 报文接口命令仲裁 2 寄存器

表 13.19 CAN 报文接口命令仲裁 2 寄存器（CANIF1_ARB2 和 CANIF2_ARB2，地址 0x4005 0034 和 0x4005 0094）位描述

位	符号	值	描述	复位值	访问
12:0	ID[28:16]		报文标识符 29 位标识符（“扩展的帧”） 11 位标识符（“标准的帧”）	0x00	R/W
13	DIR		报文方向	0x00	R/W
		0	方向=接收 在 TXRQST 里，把带该报文对象标识符的远程帧发送出去。在接收到带匹配标识符的数据帧时，将此寄存器存放在该报文对象里		
		1	方向=发送 在 TXRQST 里，相应的报文对象作为数据帧发送出去。在接收到带匹配标识符的远程帧时，该报文对象的 TXRQST 位被置位（如果 RMTEN=1）		
14	XTD		扩展标识符	0x00	R/W
		0	11 位标准标识符用于该报文对象		
		1	29 位扩展标识符用于该报文对象		
15	MSGVAL		报文有效 注：CPU 在复位 CAN 控制寄存器的 INIT 位之前的初始化进程中，必须复位所有未被使用的报文对象的 MSGVAL 位。在修改标识符 ID28:0、控制位 XTD、DIR 或数据长度代码 DLC3:0 之前，或不再要求使用报文对象时，必须也要将该位复位	0	R/W
		0	报文处理程序忽略报文对象		
		1	报文对象被配置，且由报文处理程序进行识别		
31:16	-	-	保留	0	-

（5）CAN 报文接口报文控制寄存器

表 13.20 CAN 报文接口报文控制寄存器（CANIF1_MCTRL 和 CANIF2_MCTRL0094，地址 0x4005 0038 和 0x4005 0098）位描述

位	符号	值	描述	复位值	访问
3:0	DLC[3:0]		数据长度代码 注：报文对象的数据长度代码的定义必须要与所有相应的对象相同，在其它的节上具有相同的标识符。当报文处理程序存放数据帧时，它将会把 DLC 写入接收到的报文所给定的值	0000	R/W
		0000～0100	数据帧具有 0～8 个数据字节		
		0101～1111	数据帧具有 8 个数据字节		
6:4	-		保留	-	-
7	EOB		缓冲区结束	0	R/W
		0	报文对象属于 FIFO 缓冲区，且不会是 FIFO 缓冲区的最后一个报文对象		
		1	FIFO 缓冲区的单个报文对象或最后一个报文对象		
8	TXRQST		发送请求	0	R/W
		0	该报文对象不是在等待着被传送		
		1	请求发送该报文对象，但还没有完成发送		
9	RMTEN		远程使能	0	R/W
		0	在接收到远程帧时，不会更改 TXRQST		
		1	在接收到远程帧时，置位 TXRQST		
10	RXIE		接收中断使能	0	R/W
		0	在成功接收帧后，INTPND 不改变		
		1	在成功接收帧后，INTPND 将置位		
11	TXIE		发送中断使能	0	R/W
		0	在成功接收帧后，INTPND 位不改变		
		1	在成功接收帧后，INTPND 位将置位		
12	UMASK		使用接收屏蔽 注：如果用户将 UMASK 设置为 1，那么就要在将 MAGVAL 设为 1 之前的初始化进程中，要对报文对象的屏蔽位进行编写	0	R/W
		0	忽略屏蔽		
		1	使用屏蔽（MSK[28:0]、MXTD 和 MDIR）进行接收过滤操作		
13	INTPND		中断挂起	0	R/W
		0	该报文对象不是中断源		
		1	该报文对象是中断源。中断寄存器的中断标识符将会指向该报文对象，如果不存在更高优先级的中断源		

续上表

位	符号	值	描述	复位值	访问
14	MSGLST		报文丢失（只对接收方向的报文对象有效）	0	R/W
		0	无报文丢失，因为 CPU 将该位最后复位		
		1	在 NEWDAT 仍保持置位时报文处理程序将一个新的报文存放在该对象中，现 CPU 将报文丢失		
15	NEWDAT		新数据	0	R/W
		0	报文处理程序没有把新数据被写入到该报文对象的数据部分，因为该标志由 CPU 最后清零		
		1	报文处理程序或 CPU 已将新数据写入到该报文对象的数据部分		
31:16	-	-	保留	0	-

（6）CAN 报文接口数据 A1 寄存器

在 CAN 数据帧中，DATA0 是首先被发送或接收的字节，而 DATA7（在 CAN_IF1B2 和 CAN_IF2B2）则是最后被发送或接收的字节。在 CAN 的串行位流中，首先发送的是每个字节的最高位。

注：在接收进程中，字节 DATA0 是最先被移送到 CAN 内核的移位寄存器的数据字节，字节 DATA7 则是最后发送。当报文处理程序存放数据帧时，它将会把所有八个数据字节写入到报文对象中。如果数据长度代码少于 8，则剩余在报文对象的字节将会被非指定的值覆盖。

表 13.21 CAN 报文接口数据 A1 寄存器（CANIF1_DA1 和 CANIF2_DA1，地址 0x4005 003C 和 0x4005 009C）位描述

位	符号	描述	复位值	访问
7:0	DATA0	数据字节 0	0x00	R/W
15:8	DATA1	数据字节 1	0x00	R/W
31:16	-	保留	-	-

（7）CAN 报文接口数据 A2 寄存器

表 13.22 CAN 报文接口数据 A2 寄存器（CANIF1_DA2 和 CANIF2_DA2，地址 0x4005 0040 和 0x4005 00A0）位描述

位	符号	描述	复位值	访问
7:0	DATA2	数据字节 2	0x00	R/W
15:8	DATA3	数据字节 3	0x00	R/W
31:16	-	保留	-	-

（8）CAN 报文接口数据 B1 寄存器

表 13.23 CAN 报文接口数据 B1 寄存器（CANIF1_DB1 和 CANIF2_DB1，地址 0x4005 0044 和 0x4005 00A4）位描述

位	符号	描述	复位值	访问
7:0	DATA4	数据字节 4	0x00	R/W
15:8	DATA5	数据字节 5	0x00	R/W
31:16	-	保留	-	-

（9）CAN 报文接口数据 B2 寄存器

表 13.24 CAN 报文接口数据 B2 寄存器（CANIF1_DB2 和 CANIF2_DB2，地址 0x4005 0048 和 0x4005 00A8）位描述

位	符号	描述	复位值	访问
7:0	DATA6	数据字节 6	0x00	R/W
15:8	DATA7	数据字节 7	0x00	R/W
31:16	-	保留	-	-

13.8.3 报文处理程序寄存器

所有报文处理程序寄存器都是只读寄存器。它们的内容（每一个报文对象的 TXRQST、NEWDAT、INTPND 和 MSGVAL 位和中断标识符）是由报文处理程序 FSM 提供的状态信息。

1. CAN 发送请求 1 寄存器

该寄存器包含报文对象（1～16）的 TXRQST 位。通过读出 TXRQST 位，CPU 可以查看出哪一个报文对象的发送请求被挂起。在接收远程帧后，或成功发送后，特定报文对象的 TXRQST 位可由 CPU 通过 IFx 报文接口寄存器来进行设置/复位，或可由报文处理程序进行设置/复位。

表 13.25 CAN 发送请求 1 寄存器（(CANTXREQ1，地址 0x4005 0100）位描述

位	符号	值	描述	复位值	访问
15:0	TXRQST[16:1]		报文对象 16～1 的发送请求	0x00	R
		0	该报文对象不是在等待着被发送		
		1	请求发送该报文对象，但发送仍未完成		
31:16	-	-	保留	-	-

2. CAN 发送请求 2 寄存器

该寄存器包含报文对象（32～17）的 TXRQST 位。通过读出 TXRQST 位，CPU 可以查看出哪一个报文对象的发送请求被挂起。在接收远程帧后，或成功发送后，特定报文对象的 TXRQST 位可由 CPU 通过 IFx 报文接口寄存器来进行设置/复位，或可由报文处理程序进行设置/复位。

表 13.26 CAN 发送请求 2 寄存器（(CANTXREQ2，地址 0x4005 0104）位描述

位	符号	值	描述	复位值	访问
15:0	TXRQST[32:17]		报文对象 32～17 的发送请求	0x00	R
		0	该报文对象不是在等待着被发送		
		1	请求发送该报文对象，但发送仍未完成		
31:16	-	-	保留	-	-

3. CAN 新数据 1 寄存器

该寄存器包含报文对象（16～1）的 NEWDAT 位。通过读出 NEWDAT 位，CPU 可以查看出哪一个报文对象的数据部分被更新。在接收完数据帧，或成功发送后，特定报文对象的 NEWDAT 位可由 CPU 通过 IFx 报文接口寄存器来进行设置/复位，或可由报文处理程序进行设置/复位。

表 13.27 CAN 新数据 1 寄存器（CANND1，地址 0x4005 0120）位描述

位	符号	值	描述	复位值	访问
15:0	NEWDAT[16:1]		报文对象 16～1 的新数据位	0x00	R
		0	报文处理程序没有把新数据写入该报文对象的数据部分，因为 CPU 之前已把该标志清零		
		1	报文处理程序或 CPU 已把新数据写入该报文对象的数据部分		
31:16	-	-	保留	-	-

4. CAN 新数据 2 寄存器

该寄存器包含报文对象（32～17）的 NEWDAT 位。通过读出 NEWDAT 位，CPU 可以查看出哪一个报文对象的数据部分被更新。在接收完数据帧，或成功发送完后，特定报文对象的 NEWDAT 位可由 CPU 通过 IFx 报文接口寄存器来进行设置/复位，或可由报文处理程序进行设置/复位。

表 13.28 CAN 新数据 2 寄存器（CANND2，地址 0x4005 0124）位描述

位	符号	值	描述	复位值	访问
15:0	NEWDAT[32:17]		报文对象 32～17 的新数据位	0x00	R
		0	报文处理程序没有把新数据写入该报文对象的数据部分，因为 CPU 之前已把该标志清零		
		1	报文处理程序或 CPU 已把新数据写入该报文对象的数据部分		
31:16	-	-	保留	-	-

5. CAN 中断挂起 1 寄存器

该寄存器包含报文对象（16～1）的 INTPND 位。通过读出 INTPND 位，CPU 可以查看出哪一个报文对象的中断被挂起。在接收完帧，或成功发送完帧后，特定报文对象的 INTPND 位可由 CPU 通过 IFx 报文接口寄存器来进行设置/复位，或可由报文处理程序进行设置/复位。这将还会影响中断寄存器中 INTPND 位的值。

表 13.29 CAN 中断挂起 1 寄存器（(CANIR1，地址 0x4005 0140）位描述

位	符号	值	描述	复位值	访问
15:0	INTPND[16:1]		报文对象 16～1 的中断挂起位	0x00	R
		0	报文处理程序忽略该报文对象		
		1	该报文对象是中断源		
31:16	-	-	保留	-	-

6. CAN 中断挂起 2 寄存器

该寄存器包含报文对象（32～17）的 INTPND 位。通过读出 INTPND 位，CPU 可以查看出哪一个报文对象的中断被挂起。在接收完帧，或成功发送完帧后，特定报文对象的 INTPND 位可由 CPU 通过 IFx 报文接口寄存器来进行设置/复位，或可由报文处理程序进行设置/复位。这将还会影响中断寄存器中 INTPND 位的值。

表 13.30 CAN 中断挂起 2 寄存器（(CANIR2，地址 0x4005 0144）位描述

位	符号	值	描述	复位值	访问
15:0	INTPND[32:17]		报文对象 32～17 的中断挂起位	0x00	R
		0	报文处理程序忽略该报文对象		
		1	该报文对象是中断源		
31:16	-	-	保留	-	-

7. CAN 报文有效 1 寄存器

该寄存器包含报文对象（16～1）的 MSGVAL 位。通过读出 MSGVAL 位，CPU 可以查看出哪一个报文对象有效。特定报文对象的 MSGVAL 位可由 CPU 通过 IFx 报文接口寄存器来进行设置/复位。

表 13.31 CAN 报文有效 1 寄存器（CANMSGV1，地址 0x4005 0160）位描述

位	符号	值	描述	复位值	访问
15:0	MSGVAL[16:1]		报文对象 16～1 的报文有效位	0x00	R
		0	报文处理程序忽略该报文对象		
		1	该报文对象被配置，且应被报文处理程序识别		
31:16	-	-	保留	-	-

8. CAN 报文有效 2 寄存器

该寄存器包含报文对象（32～17）的 MSGVAL 位。通过读出 MSGVAL 位，CPU 可以查看出哪一个报文对象有效。特定报文对象的 MSGVAL 位可由 CPU 通过 IFx 报文接口寄存器来进行设置/复位。

表 13.32 CAN 报文有效 2 寄存器（CANMSGV2，地址 0x4005 0164）位描述

位	符号	值	描述	复位值	访问
15:0	MSGVAL[32:17]		报文对象 32～17 的报文有效位	0x00	R
		0	报文处理程序忽略该报文对象		
		1	该报文对象被配置，且应被报文处理程序识别		
31:16	-	-	保留	-	-

13.8.4 CAN 定时寄存器

1. CAN 时钟分频器寄存器

该寄存器决定 CAN 时钟信号。以该寄存器的值对外设时钟 PCLK 进行分频，可得到 CAN_CLK。

表 13.33 CAN 时钟分频器寄存器（CANCLKDIV，地址 0x4005 0180）位描述

位	符号	值	描述	复位值	访问
2:0	CLKDIVVAL		时钟分频器值	001	R/W
		000	CAN_CLK=对 PCLK 进行 1 分频		
		001	CAN_CLK=对 PCLK 进行 2 分频		
		010	CAN_CLK=对 PCLK 进行 4 分频		
		011	CAN_CLK=对 PCLK 进行 8 分频		
		100	CAN_CLK=对 PCLK 进行 16 分频		
		101	CAN_CLK=对 PCLK 进行 32 分频		
		110	CAN_CLK=无效（默认为进行 2 分频）		
		111	CAN_CLK=无效（默认为进行 2 分频）		
31:3	-	-	保留	-	-

13.9 功能描述

13.9.1 复位后 C_CAN 控制器状态

在硬件复位后，寄存器应保存表 13.3 所描述的值。另外，总线关闭状态被复位，输出 CAN_TXD 被设为隐性（高电平）。CAN 控制寄存器的 0x0001 值（INIT= ‘1’）使能软件初始化。CAN 控制器为不会与 CAN 总线进行通信，直至 CPU 将 INIT 复位为 ‘0’。

存放在报文 RAM 中的数据不受硬件复位的影响。在上电后，报文 RAM 中的内容未定义。

13.9.2 C_CAN 操作模式

1. 软件初始化

软件初始化通过设置 CAN 控制寄存器的 INIT 位来启动，也可通过软件或硬件复位，或进入总线关闭状态来启动。

在软件初始化（INIT 位被置位）期间，要符合下列的条件：

- 所有往返 CAN 总线的报文传输停止；
- CAN 输出 CAN_TXD 的状态为隐性（高电平）；
- EML 计数器不改变；
- 配置寄存器不改变；
- 如果 CAN 控制寄存器的 CCE 位也被置位，访问位定时寄存器且使能 BRP 扩展寄存器。

为了初始化 CAN 控制器，软件必须设置位定时寄存器和每一个报文对象。如果不需要报文对象，可以将 MSGVAL 位设置为无效。否则，必须初始化整个报文对象。

复位 INIT 位来完成软件的初始化。在位流处理器 BSP 参与总线活动和启动报文传输之前，可通过对 11 个执行隐性位出现的序列（1/2 总线空闲）执行写操作，即可将其同步到 CAN 总线上的数据传输。

注：报文对象的初始化与 INIT 的状态无关，也可实时完成，但是在 BSP 启动报文传输之前，在执行软件初始化的过程中，应要将报文对象配置到特定的标识符中或设置为无效。为了在正常操作的过程中更改报文对象的配置，必须要将 MSGVAL 位设置为无效来启动 CPU。当配置完成时，再次将 MSAGVAL 设为有效。

2. CAN 报文传输

一旦 CAN 控制器启动且 INIT 被复位为 0，CAN 内核会把自身同步到 CAN 总线，并启动报文传输。

如果接收到报文传递报文处理程序的接收过滤，则它们会被存放在各自相关的报文对象里。整个报文，包括所有仲裁位、DLC 和 8 个数据字节，都存放在报文对象中。如果使用了标识符屏蔽，则被标识为“无关”的仲裁位有可能在报文对象中被覆盖。

CPU 通过接口寄存器可随时读或写每一个报文。在并发存取的情况下，报文处理程序可以保证数据的连贯性。

要被发送的报文由 CPU 进行更新。如果报文有一个永久的报文对象存在（仲裁和控制位在配置过程中设置），则只更新数据字节，然后 TXRQUT 位和 NEWDAT 位被置位来启动传送。如果向同一个报文对象分配几个发送报文（当报文对象的编号不够用时），则在请求发送该报文之前就要把整个报文对象配置好。

任何编号的报文对象的传送可以在同时被请求，其后依照它们的内部优先级来进行发送。报文可以随时被更新或被设为无效，即使它们请求的传送仍在挂起。在启动挂起的传送之前，如果更新了报文，则会丢弃旧的数据。

根据报文对象的配置，在的接收到具有匹配标识符的远程帧时，可以自主请求传文传送。

3. 禁能的自动重发（DAR）

依照 CAN 规范（ISO11898，6.3.3 恢复管理），CAN 控制器为在传送过程中丢失仲裁的帧或被错误干扰的帧提供了几种自动重发的方法。在传送成功完成之前，用户不可以使用帧发送服务。在默认情况下，当出现丢失仲裁或错误时，自动重发使能。可以将其禁能，以令 CAN 控制器可以在定时触发的 CAN（TTCAN，见 ISO11898-1）环境内工作。

将 CAN 控制寄存器的 DAR 位编写为 1，即可将禁能的自动重发模式使能。在该操作模式下，编程者必须要考虑报文缓冲区里控制寄存器的 TXRQST 和 NEWDAT 位的不同行为：

- 当传送启动时，相关报文缓冲区的 TXRQST 位复位，而 NEWDAT 位保持置位；
- 当传送成功完成时，位 NEWDAT 复位；
- 当传送失败时（丢失仲裁或错误），NEWDAT 位保持置位。为重启传送，CPU 必须要将 TXRQST 重设回 1。

4. 测试模式

在 CAN 控制寄存器中将 TEST 位设置为 1，即可进入测试模式。在测试模式下，可对测试寄存器的 TX1、TX0、LBACK、SILENT 和 BASIC 位进行写操作。位 RX 监控着引脚 RD0、1 的状态，因此它是只读位。所有测试寄存器的功能在位 TEST 被复位为 0 时禁能。

（1）静默模式

将测试寄存器的 SILENT 位设为 1，即可令 CAN 内核处于静默模式。

在静默模式下，CAN 控制器可以接收有效的数据帧和有效的远程帧。但是它在 CAN 总线上只发送隐性位，并不能启动传送。如果要求 CAN 内核发送显性位（ACK 位，过载标志、有效错误标志），则内部重发该位，因此 CAN 内核监控该显性位，虽然 CAN 总线可能仍处于隐性状态。通过传送显性位（应答位、错误帧），静默模式可在不影响总线的情况下分析 CAN 总线的流量。

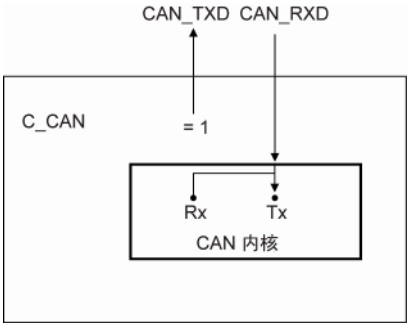

图 13.2　静默模式下的 CAN 内核

（2）环回模式

将测试寄存器的 LBACK 位编写为 1，即可令 CAN 内核处于环回模式。在环回模式下，CAN 内核将自身拥有被发送的报文看作是接收到的报文，并将它们（如果它们传递接收过滤）存放到接收缓冲区中。

该模式提供自测试功能。为独立于外部仿真，CAN 内核在环回模式下会忽略应答错误（数据/远程帧应答槽上采样到的隐性位）。在该模式中，CAN 内核执行从 CAN_TXD 输出到 CAN_RXD 输入的内部反馈。CAN_RXD 输入引脚的真实值被 CAN 内核忽略。可以在 CAN_TXD 引脚上监控被发送的报文。

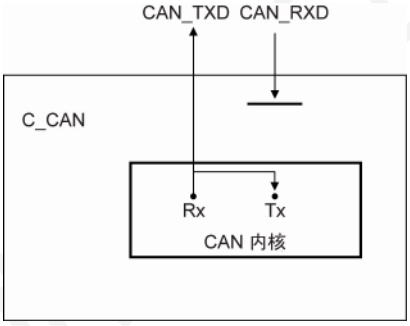

图 13.3　环回模式下的 CAN 内核

（3）环回模式和静默模式的整合

将 LBACK 和 SLIENT 位同时编写为 1，可以整合环回模式和静默模式。该模式用于“Hot Selftest”，表示在不影响正在运行的 CAN 系统（接入 CAN_TXD 和 CAN_RXD 引脚）前提下，可以测试 C_CAN。在该模式下，CAN_RXD 不接入 CAN 内核，且 CAN_TXD 引脚保持为隐性。

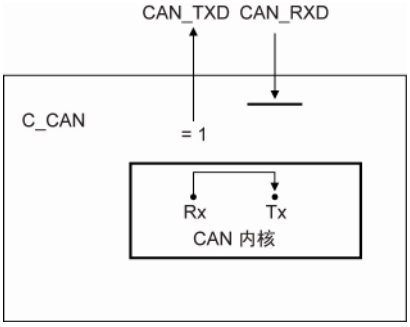

图 13.4　环回模式和静默模式整合下的 CAN 内核

（4）基本模式

将测试寄存器的 BASIC 位编写为 1，即可令 CAN 内核进入基本模式。在该模式下，CAN 控制器运行时不需要使用报文 RAM。

IF1 寄存器用作发送缓冲区。将 IF1 命令请求寄存器的 BUSY 位写‘1’，可请求发送 IF1 寄存器的内容。当 BUSY 置位时，IF1 寄存器被锁定。BUSY 位表示传送正在挂起。

一旦 CAN 总线空闲，IF1 寄存器被载入到 CAN 内核的移位寄存器中，并启动传送。当传送结束时，BUSY 位复位，且锁定的 IF1 寄存器被释放。

当 IF1 寄存器被锁定时，将 IF1 命令请求寄存器的 BUSY 位复位，可中止挂起的传送。如果 CPU 已经复位 BUSY 位，则在丢失仲裁或禁能错误的情况下，可以进行重发送操作。

IF2 寄存器用作接收缓冲区。在接收到报文之后，移位寄存器的内容被存放到 IF2 寄存器中，而不用进行任何接收过滤。

另外，可以在报文传输的过程中监控移位寄存器的实际内容。每次向 IF2 命令请求寄存器的 BUSY 位写入‘1’，都会启动读报文对象操作，移位寄存器的内容存放在 IF2 寄存器中。

在基本模式下，IFx 命令屏蔽寄存器控制位上与控制和状态位相关的所有报文对象的评估被关闭。命令请求寄存器的报文编号不能被评估。IF2 报文控制寄存器的 NWEDAT 和 MSGLST 位保留着它们的功能，DLC3～0 将会显示所接收到的 DLC，其它的控制位被读为‘0’。

在基本模式下，就绪输出 CAN_WAIT_B 被禁能（总为‘1’）。

（5）软件控制 CAN_TXD 引脚

在 CAN 发送 CAN_TXD 引脚上，共有 4 种输出功能：

- 串行数据输出（默认）；
- 驱动 CAN 采样点信号来监控 CAN 控制器的时序；
- 驱动隐性常量值；
- 驱动显性常量值。

最后的二个功能，结合可读的 CAN 接收 CAN_RXD 引脚，可用于检查 CAN 总线的物理层。

将测试寄存器位 TX1 和 TX0 按“CAN 测试寄存器”小节那样来编写，即可选择 TD 引脚的输出模式。

注：CAN_TXD 引脚的三个测试功能涉及所有的 CAN 协议函数。当 CAN 报文传输或选择了从环回模式、静默模式或基本模式中的任何一种模式时，必须令 CAN_TXD 引脚处于其默认功能。

13.9.3 CAN 报文处理程序

报文处理程序控制 CAN 内核 Rx/Tx 移位寄存器、报文 RAM 和 IFx 寄存器之间的数据传输，见图 13.5。

报文处理程序控制下列函数：

- IFx 寄存器和报文 RAM 之间的数据传输；
- 将移位寄存器的数据传输到报文 RAM 中；
- 将报文 RAM 中的数据传输到移位寄存器中；
- 将移位寄存器的数据传输到接收过滤单元中；
- 扫描报文 RAM，以查找符合匹配的报文对象；
- 处理 TXRQST 标志；
- 处理中断。

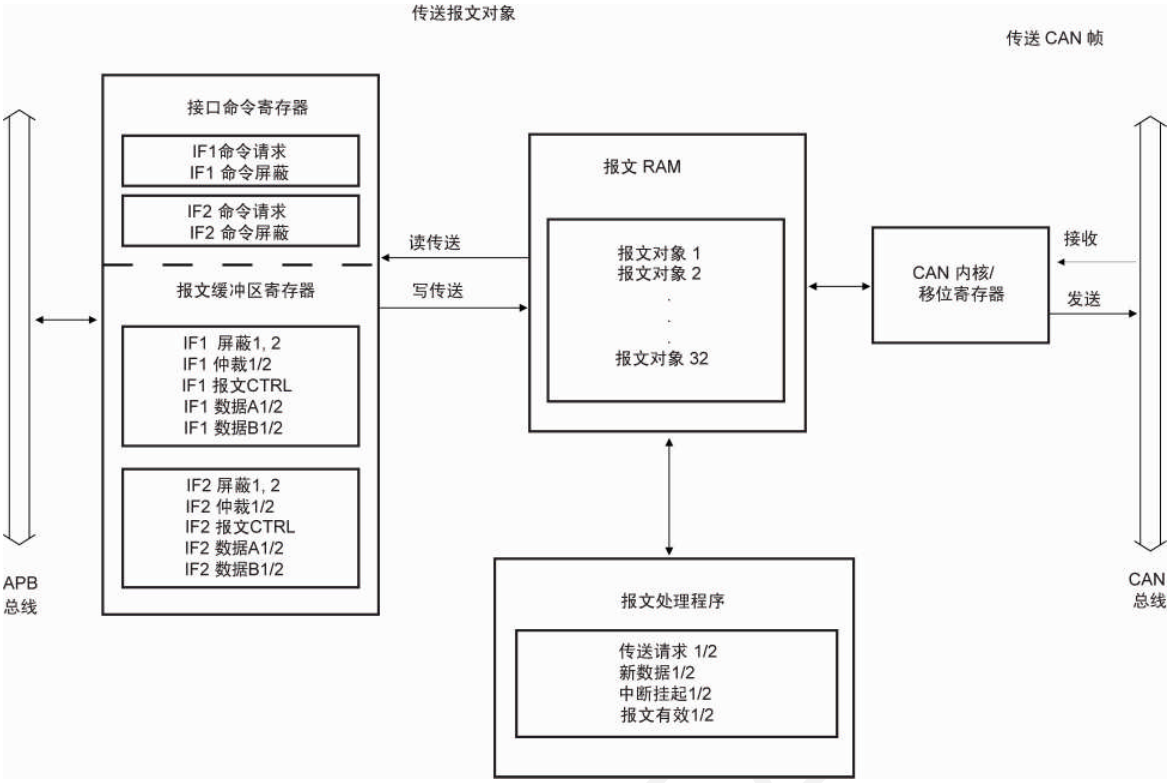

图 13.5　报文对象传输的方框图

1.　报文对象的管理

复位芯片时，不会影响到报文 RAM 中报文对象的配置（MSGVAL、NEWDAT、INTPND 和 TXRQST 位除外）。所有的报文对象必须由 CPU 启动，或必须将它们设为无效（MSGVAL=‘0’）。位时序必须要在 CPU 清除 CAN 控制寄存器的 INIT 位之前配置好。

将二个接口寄存器中的一个寄存器的屏蔽、仲裁、控制和数据域设为所要求的值，即可完成报文对象的配置。写相应的 IFx 命令请求寄存器，可将 IFx 报文缓冲区寄存器载入到在报文 RAM 中寻址的报文对象中。

当 CAN 控制寄存器的 INIT 位清零时，CAN 内核的 CAN 协议控制器状态机和报文处理程序状态机控制着 CAN 控制器的内部数据流。传递接收过滤的接收报文被存放到报文 RAM 中，带有挂起发送请求的报文被载入到 CAN 内核的移位寄存器中，并通过 CAN 总线来发送。

CPU 通过 IFx 接口寄存器来读取接收到的报文并更新将被发送的报文。根据配置，CPU 会被某些 CAN 报文和 CAN 错误事件中断。

2.　IFx 寄存器和报文 RAM 之间的数据传输

当 CPU 启动 IFx 寄存器和报文 RAM 之间的数据传输时，报文处理程序将相关命令寄存器的 BUSY 位设为‘1’。在完成传输后，BUSY 位被为‘0’。

命令屏蔽寄存器指定是传输一个完整的报文对象还是只传输报文对象的一部分。由于报文 RAM 的结构的原因，不可能写一个报文对象的单个位/字节。软件必须将完整的报文对象写入到报文 RAM 中。因此，将 IFx 寄存器的数据传送到报文 RAM 要求一个读-改-写周期：

1）使用命令屏蔽寄存器来读取报文对象的器件（不会从报文 RAM 中被更改）。

- 在部分读取报文对象后，不是在命令屏蔽寄存器中选择出的报文缓冲区寄存器将会保持不变。

2）将报文缓冲区寄存器的完整内容写入到报文对象中。

- 在部分写报文对象后，不是在命令屏蔽寄存器中选择出的报文缓冲区寄存器将会被设为所选的报文对象的实际内容。

3. CAN 内核的移位寄存器和报文缓冲区之间的报文传送

如果 CAN 内核单元的移位寄存器已准备好进行加载，并且，如果 IFx 寄存器和报文 RAM 之间没有数据要传输，则要对报文有效寄存器的 MSGVAL 位和传送请求寄存器的 TXRQST 位进行评估。挂起传送请求的、具有最高优先级的有效报文对象会被报文处理程序载入到移位寄存器中，并启动传送。报文对象的 NEWDAT 位复位。

在成功完成传送后，且没有把新数据写入到报文对象（NEWDAT=‘0’），由于启动了传送，所以 TXRQST 位将会复位。如果 TXIE 置位，INTPND 将会在成功完成传输后置位。如果在传送的过程中，CAN 控制器丢失了仲裁或发生了错误，一旦 CAN 总线再次空闲，将会重发报文。如果此时请求了要发送一个优先级更高的报文，则会按报文优先级的排序来发送报文。

4. 接收到的报文的接收过滤

当正在进入报文的仲裁和控制域（标识符+IDE+RTR+DLC）已完全移入到 CAN 内核的 Rx/Tx 移位寄存器中时，报文处理程序状态机会启动报文 RAM 的扫描操作，以查找匹配有效的报文对象。

为了扫描报文 RAM 以查找到匹配的报文对象，要将 CAN 内核移位寄存器的仲裁位载入到接收过滤单元中。然后报文对象 1 的仲裁和屏蔽域（包括 MSGVAL、UMASK、NEWDAT、和 EOB）被载入到接收过滤单元中，并将它们与移位寄存器的仲裁域进行比较。对每一个随后的报文对象都进行重复的操作，直至查找到匹配的报文对象，或直至到达报文 RAM 的尽头。

如果存在匹配的报文对象，则扫描停止，报文处理程序状态机会根据所接收到的帧（数据帧或远程帧）类型作出处理。

（1）接收数据帧

报文处理程序状态机将 CAN 内核移位寄存器上的报文存放到报文 RAM 中相关的报文对象。数据字节、所有仲裁位和数据长度代码被存放在相关的报文对象里。执行这样子的操作是为了令数据字节与标识符相连，即使使用了仲裁屏蔽寄存器。

置位 NEWDAT 位来表示已接收到新数据（CPU 没有发现）。当 CPU/软件读取报文对象时，应要复位 NEWDAT。如果在接收的时候，NEWDAT 位已被置位，则置位 MSGLST 位来表示之前的数据（假定 CPU 没有发现）丢失。如果 RxIE 位被置位，INTPND 位也置位，会令中断寄存器指向该报文对象。

在刚接收完请求的数据帧时，复位该报文对象的 TXRQST 位来阻止传送远程帧。

（2）接收远程帧

当接收远程帧时，必须要考虑匹配报文对象的三种不同配置：

1）DIR=‘1’（方向=发送），RMTEN=‘1’，UMASK=‘1’或‘0’

在接收到匹配远程帧时，该报文对象的 TXRQST 位被置位。其余的保持不变。

2）DIR=‘1’（方向=发送），RMTEN=‘0’，UMASK=‘0’

在接收到匹配远程帧时，该报文对象的 TXRQST 位保持不变，远程帧被忽略。

3）DIR=‘1’（方向=发送），RMTEN=‘0’，UMASK=‘1’

在接收到匹配远程帧时，该报文对象的 TXRQST 位复位。移位寄存器的仲裁和控制域（标识符+IDE+RTR+DLC）被存放到报文 RAM 中的报文对象中，并设置该报文对象的 NEWDAT 位。报文对象的数据域保持不变，处理远程帧的方式与处理接收到的数据帧相似。

5. 接收/发送优先级

报文对象的接收/发送优先级与报文编号相关。报文对象 1 具有最高的优先级，而报文对象 32 具有最低的优先级。如果超过一个发送请求被挂起，要按相应的报文对象的优先级来进行处理。

6. 配置发送对象

表 13.34 所示为软件应要如何初始化一个发送对象（也可参考表 13.12）：

表 13.34 初始化发送对象

MSGVAL	仲裁位	数据位	屏蔽位	EOB	DIR	NEWDAT
1	取决于应用	取决于应用	取决于应用	1	1	0
MSGLST	RXIE	TXIE	INTPND	RMTEN	TXRQST	
0	0	取决于应用	0	取决于应用	0	

仲裁寄存器（ID28:0 和 XTD 位）由应用程序给定。它们定义传送出去的报文的标识符和类型。如果使用 11 位标识符（“标准帧”），可将其编写为 ID28。在这种情况下，可以忽略 ID18、ID17～ID0。

如果 TXIE 位置位，则在成功传送完报文对象之后，置位 INTPND 位。

如果 RMTEN 位置位，匹配接收的远程帧将会令 TXRQST 位置位，且数据帧会自动应答远程帧。

数据寄存器（DLC3:0，数据 0:7）由应用程序给定。在数据有效之前，有可能不会将 TXRQST 和 RMTEM 置位。

可以使用屏蔽寄存器（Msk28～0、UMASK、MXTD 和 MDIR 位）（MUASK= ‘1’），以允许用带有相似标识符的远程帧组来设置 TXRQST 位。详情请参考“接收远程帧”小节。不应屏蔽 DIR 位。

7. 更新发送对象

CPU 随时可通过 IFx 接口寄存器来更新发送对象的数据字节。在更新之前，不必将 MSGVAL 或 TXRQST 复位。

即使只更新一部分数据字节，在将该寄存器的内容传送到报文对象之前，要令所有相关 IFx 数据 A 寄存器或 IFx 数据 B 寄存器的四个字节有效。在 CPU 写入新数据字节之前，CPU 要将所有四个字节写入到 IFx 数据寄存器中或将报文对象传送到 IFx 数据寄存器中。

当仅仅是更新数据字节（8 个）时，最先将 0x0087 写入命令屏蔽寄存器。然后将报文对象的编号写入命令请求寄存器，同时更新数据字节，并设置 TXRQST。

在更新数据时，为防止有可能在已正进行的传送结束时复位 TXRQST 位，必须要将 NEWDAT 和 TXRQST 同时置位。详情请参考“CAN 内核的移位寄存器和报文缓冲区之间的报文传送”小节。

当同时设置 NEWDAT 和 TXRQST 时，一旦新传送启动，NEWDAT 将复位。

8. 配置接收对象

表 13.35 所示为软件应要如何初始化一个发送对象（也可参考表 13.12）：

表 13.35 初始化接收对象

MSGVAL	仲裁位	数据位	屏蔽位	EOB	DIR	NEWDAT
1	取决于应用	取决于应用	取决于应用	1	0	0
MSGLST	RXIE	TXIE	INTPND	RMTEN	TXRQST	
0	取决于应用	0	0	0	0	

仲裁寄存器（ID28～0 和 XTD 位）由应用程序给定。它们定义接收到的接收报文的标识符和类型。如果使用 11 位标识符（“标准帧”），可将其编写为 ID28～ID18。可以忽略 ID17～ID0。当接收到具有 11 位标识符的数据帧时，ID17～ID0 将会被设置为‘0’。

如果 RXIE 位置位，则在接受了接收到的数据帧，并将其存到报文对象之后，置位 INTPND 位。

数据长度代码（DLC[3:0]）由应用程序给定。当报文处理程序将数据帧存放到报文对象时，它将会存放接收到的数据长度代码和 8 个数据字节。如果数据长度代码少于 8，则报文对象的其余字节将会被非指定的值覆盖。

可以使用屏蔽寄存器（Msk[28:0]、UMASK、MXTD 和 MDIR 位）（UMASK=‘1’），以允许带有相似标识符的数据帧组被接受。详情请参考“接收数据帧”小节。在典型应用程序中，不应屏蔽 DIR 位。

9. 处理接收到的报文

CPU 随时可通过 IFx 接口寄存器读取接收到的报文。报文处理程序状态机可保证数据的连贯性。

为将报文 RAM 中整个接收到的报文传送到报文缓冲区中，软件首先必须要将 0x007F 写入命令屏蔽寄存器。然后将报文对象的编号写入命令请求寄存器。另外，在报文 RAM（不是在报文缓冲区）中清除 NEWDAT 和 INTPND 位。

如果报文对象使用屏蔽来进行接收过滤操作，仲裁位会显示已接收到哪一个匹配的报文。

NEWDAT 的真实值显示自从上一次读取该报文对象之后，是否接收到一个新的报文。MSGLST 的真实值显示自从上一次读取该报文对象之后，是否接收到个数大于 1 的报文。MSGLST 不能自动复位。

使用远程帧时，CPU 可能要求另一个 CAN 节点为接收报文提供新数据。令接收报文的 TXRQST 位置位将会令带接收报文对象标识符的远程帧被传送。该远程帧触发其它 CAN 节点来启动匹配数据帧的传送。如果在可以发送远程帧之前接收到匹配数据帧，TXRQST 位会自动复位。

10. 配置 FIFO 缓冲区

除 EOB 位之外，属于 FIFO 缓冲区的接收对象的配置与（单个）接收对象的配置相同，见“配置接收对象”小节。

为将二个或多个报文对象放入 FIFO 缓冲区，必须要这些报文对象的标识符和屏蔽（如有使用）编写为匹配值。由于报文对象的隐性优先级，具有最低编号的报文对象将会是 FIFO 缓冲区的第一个报文对象。FIFO 缓冲区的所有报文对象的 EOB 位（最后报文对象的除外）必须被编写为 0。FIFO 缓冲区的最后报文对象的 EOB 位被设为 1，将其配置为模块的末端。

（1）接收 FIFO 缓冲区报文

具有匹配 FIFO 缓冲区标识符的接收报文被存放在该 FIFO 缓冲区的报文对象中，从最低报文编号的报文对象开始执行。

当报文存放在 FIFO 缓冲区的报文对象时，该报文对象的 NEWDAT 位被置位。在 EOB 为 0 时置位 NEWDAT，报文对象被锁定，由报文处理程序进行进一步的写访问操作，直至 CPU 将 NEWDAT 位写回为 0。

报文存放在 FIFO 缓冲区中，直至到达该 FIFO 缓冲区的最后一个报文对象。如果将 NEWDAT 写为 0 不会释放进程中的报文对象，则该 FIFO 缓冲区的所有以后报文都会被写入到 FIFO 缓冲区的最后一个报文对象，因此会覆盖之前的报文。

（2）读取 FIFO 缓冲区

当 CPU 通过将报文对象编号写入到 IFx 命令请求寄存器来将报文对象的内容传送到 IFx 报文缓冲区寄存器时，相关命令屏蔽寄存器的 NEWDAT 位 INTPND 位应被复位为 0（TXRQST/NEWDAT=‘1’和 ClrINTPND=‘1’）。报文控制寄存器中这些位值总是反映着这些位复位之前的状态。

为确保 FIFO 缓冲区功能正确，CPU 应在最低报文编号的 FIFO 对象处开始读出报文对象。

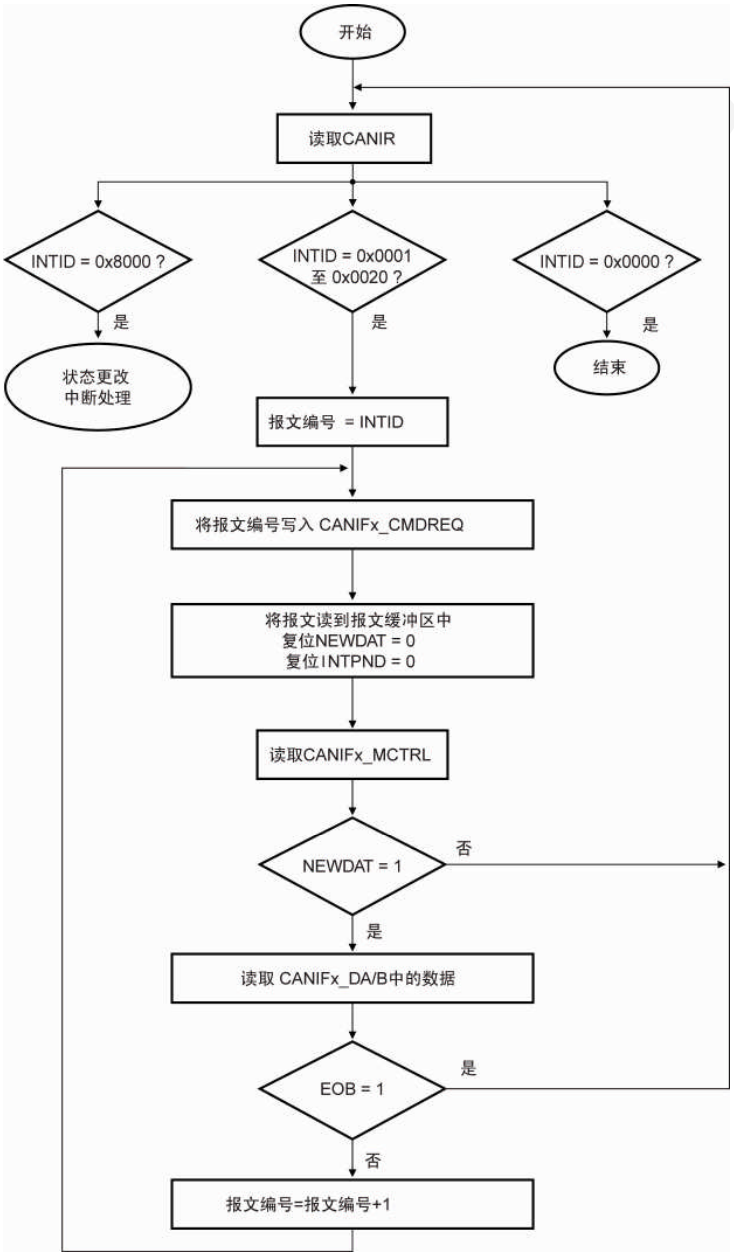

图 13.6　将 FIFO 缓冲区的报文读取到报文缓冲区

13.9.4 中断处理

如果有几个中断被挂起，CAN 中断寄存器将会指向具有最高优先级的挂起中断，忽略它们的时间先后排列的顺序。中断会保持挂起，直至 CPU 将其清除。

状态中断具有最高的优先级。在报文中断里，报文对象的中断优先级随着报文编号的递增而下降。

将报文对象的 INTPND 位清除，即可清除报文中断。状态中断的清除则通过读取状态寄存器来完成。

中断寄存器中的中断标识符 INTID 表示引发中断的原因。当没有中断挂起时，寄存器将会保存 0 值。如果中断寄存器的值不为 0，那么存在中断被挂起，如果 IE 置位，则到 CPU 的中断线 IRQ_B 有效。中断线保持有效，直至 IE 复位。

0x8000 值表示中断正在被挂起，因为 CAN 内核已更新（没有必要更改）了状态寄存器（错误中断或状态中断）。该中断具有最高的优先级。CPU 可以更新（复位）状态位 RXOK、TXOK 和 LEC，但是 CPU 对状态寄存器执行写访问操作将永不会产生或复位中断。

所有其它值表示中断源是其中的一个报文对象，其中 INTID 指向具有最高级中断优先级的挂起报文中断。

当中断寄存器不为 0 时（CAN 控制寄存器的 IE 位），CPU 控制着状态寄存器的改变是否会引发中断（CAN 控制寄存器的 EIE 和 SIE 位）和中断线是否有效。即使 IE 复位时，中断寄存器会被更新。

CPU 有二种方式来查找报文中断源：

- 软件可以查看中断寄存器的 INTID 位；
- 软件可以轮询中断挂起寄存器，见“CAN 中断挂起 1 寄存器”小节。

读取报文（这报文是中断源）的中断服务程序可以同时读取报文并复位报文对象的 INTPND（命令屏蔽寄存器的 ClrINTPND）。当 INTPND 被清除时，中断寄存器将会指向下一个具有挂起中断的报文对象。

13.9.5 位定时

即使 CAN 位定时配置里的映射错误不会导致立即的错误，但却会大大削弱了 CAN 网络的性能。在许多情况下，CAN 位同步会将 CAN 位定时的错误配置改良到这样一个程度：只偶尔产生一个错误帧。但是，在仲裁的情况下，当二个或多个 CAN 节点同时想要发送帧时，错位的采样点将会令其中一个发送器变为错误消极。

对偶尔错误的分析要求对 CAN 节点内的 CAN 位同步和 CAN 总线上的 CAN 节点的互动知识有详细的了解。

1. 位时间和位速率

CAN 支持的位速率范围为少于 1KBit/s～1000KBit/s。每一个 CAN 网络上的成员都具有自身拥有的时钟发生器，通常是石英振荡器。可以为每一个 CAN 节点单独配置位时间的定时参数（即位速率的互等数，以产生共同的位速率，即使 CAN 节点的振荡器周期（f_{osc}）可能不同。

这些振荡器的频率并不是绝对稳定的，因为温度的改变或电压和元件的恶化都会令频率发生微小的变化。只要变化量位于指定的振荡器容忍范围（df）内，CAN 节点可以通过重新同步到位流来向不同的位速率进行补偿。

依据 CAN 规范，位时间被分成四个段（图 13.7 位时序）。同步段、传播时间段、相位缓冲区段 1 和相位缓冲区段 2。每一个段包含有一个指定的、可编程的时间量子数（见表 13.36）。位时间

的基本时间单元的时间量子的长度（t_q）由 CAN 控制器的系统时钟 f 和波特经预分频器确定（BRP）：$t_q = BRP / f_{sys}$。C_CAN 的系统时钟 f_{sys} 是 LPC11Cx 系列 Cortex-M0 微控制器系统时钟的频率（见“syscon”章节）。

同步段 Sync_Seg 是位时间的一部分，CAN 总线的边沿电平会在这里发生；在 Sync_Seg 外部发生边沿和 Sync_Seg 上发生边沿之间的距离叫做该边沿的相位错误。传播时间段 Prop_Seg 用于对 CAN 网络内的物理延迟时间进行补偿。相位缓冲区段 Phase_Seg1 和 Phase_Seg2 包围着采样点。（重新）同步跳转宽度（SJW）定义着重新同步操作在相位缓冲区段所定义的内部限制内可将采样点移动的距离，以实现对边沿相位错误作出补偿。

表 13.36 描述了 CAN 协议所要求的最小可编程范围。位时间参数通过 CANBT 寄存器（表 13.7）来进行编程。关于位定时和范例的详细描述，请参考 1.2 版本的 C_CAN 用户指南。

表 13.36　C_CAN 位时间的参数

参数	范围	功能
BRP	（1...32）×t_q	定义时间量子 t_q 的长度
SYNC_SEG	1t_q	同步段。长度固定。将总线输入同步到系统时钟
PROP_SEG	（1...8）×t_q	传播时间段。对物理延迟时间进行补偿。该参数由 C_CAN 网络里的系统延迟时间决定
TSEG1	（1...8）×t_q	相位缓冲区段 1。可通过同步操作临时加长时间长度
TSEG2	（1...8）×t_q 依照 CANBT 寄存器，应为 1…16	相位缓冲区段 2。可通过同步操作临时缩短时间长度
SJW	（1...4）×t_q	（重新）同步跳转宽度。其长度不可以长于相位缓冲区段

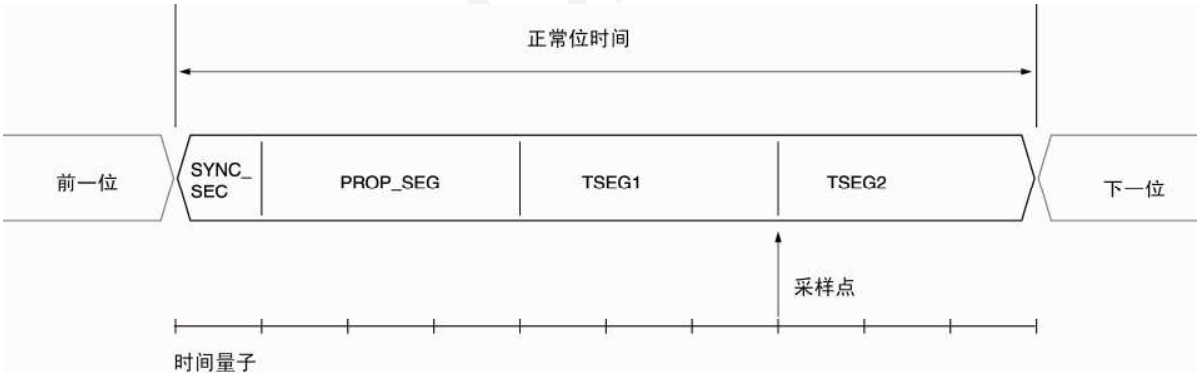

图 13.7　位时序