17 控制器局域网（CAN）模块

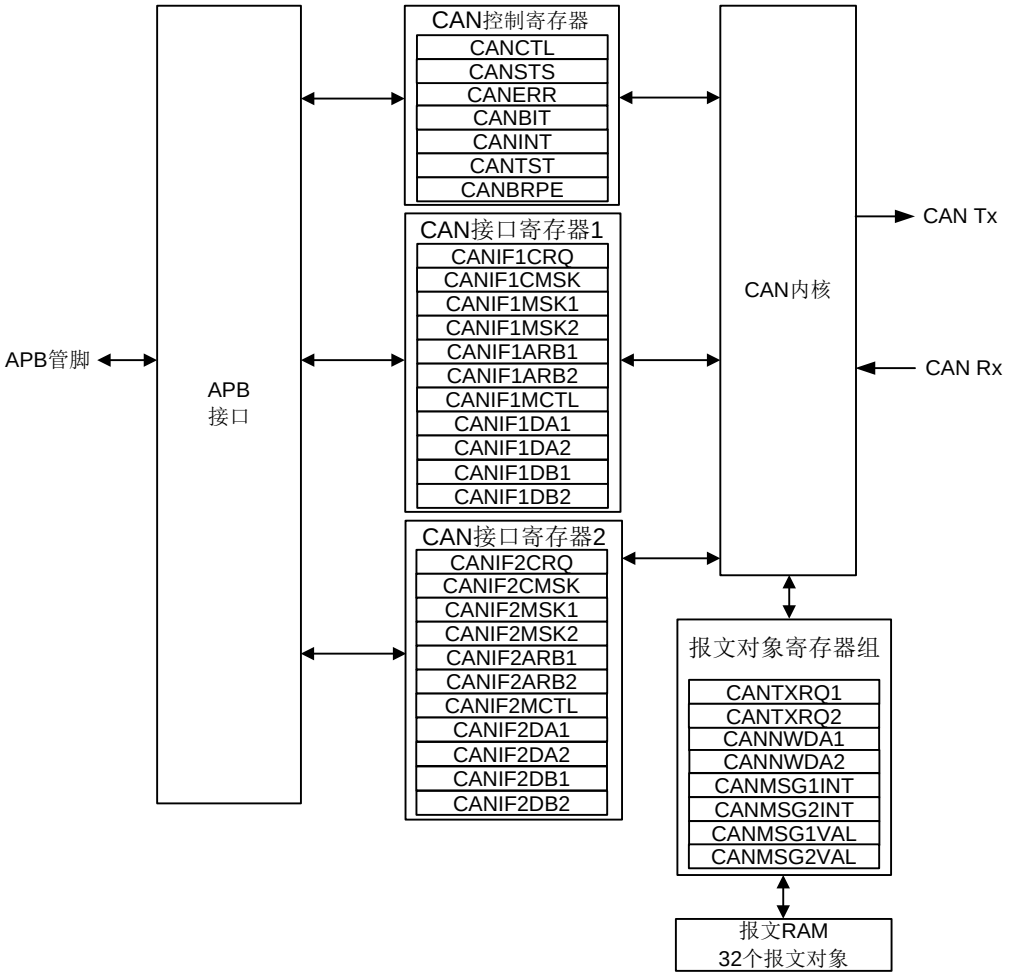
控制器局域网（**Controller Area Network**，简写为 **CAN**）是一种用于连接电子控制设备（Eletronic Control Unit，简写为 ECU）的多主共享型串行总线标准。CAN 总线针对抗电磁干扰进行了专门设计，适用于具有较强电磁干扰的环境，不但可以使用与 RS-485 类似的差分平衡传输线，也可以使用更加可靠的双绞线。CAN 总线最初是针对汽车应用而研发的，不过时至今日已经广泛应用于各种嵌入式控制领域（例如工业方面和医疗方面）。CAN 总线在总线长度小于 40 米时最高可达 1Mbps 位速率。位速率越低则有效通讯距离越远（例如 125kbps 时通讯距离可达 500 米）。

Stellaris® LM3S9B96 微控制器集成了两个 CAN 模块，具有以下特性：

- 支持 CAN 总线协议 2.0 A/B；
- 位速率最高可达 1Mbps；
- 32 个报文对象，每个报文对象都具有独立的标识符掩码；
- 可屏蔽中断；
- 支持**禁用自动重新发送（Disable Automatic Retransmission，简写为 DAR**）模式，因此可用于时间触发 CAN（Time Triggered CAN，简写为 TTCAN）应用；
- 可编程的环回模式，用于实现自检；
- 可编程的 FIFO 模式，能存储多个报文对象；
- 提供 CANnTX、CANnRX 管脚，可无缝连接片外 CAN 收发器。

17.1 框图

图 17-1. CAN 控制器模块框图



17.2 信号描述

第 864 页的表 17-1 和表 17-2 列出了与 CAN 控制器相关的所有外部信号并逐一描述其功能。CAN 控制器信号通常是 GPIO 信号的备选功能，因此这些管脚在复位时默认设置为 GPIO 信号。表中“复用管脚/赋值”一列是各 CAN 信号所对应的 GPIO 管脚。当需要使用 CAN 功能时，应将相关 **GPIO 备选功能选择寄存器（GPIOAFSEL）**（见第 429 页）中的 AFSEL 位置位，表示启用 GPIO 的备选功能；同时还应将括号内的数字写入 **GPIO 端口控制寄存器（GPIOCTRL）**（见第 447 页）的 PMCn 位域，表示将 CAN 信号赋给指定的 GPIO 管脚。关于配置 GPIO 的详细信息，请参阅第 405 页的“通用输入/输出（GPIO）”一章。

表 17-1. CAN 信号（LQFP100 封装）

管脚名称	管脚序号	复用管脚/赋值	管脚类型	缓冲类型 [a]	功能描述
CAN0Rx	10 30 34 92	PD0(2) PA4(5) PA6(6) PB4(5)	I	TTL	CAN 模块 0 接收信号
CAN0Tx	11 31 35 91	PD1(2) PA5(5) PA7(6) PB5(5)	O	TTL	CAN 模块 0 发送信号
CAN1Rx	47	PF0(1)	I	TTL	CAN 模块 1 接收信号
CAN1Tx	61	PF1(1)	O	TTL	CAN 模块 1 发送信号

a：“TTL”表示该管脚兼容 TTL 电平标准。

表 17-2. CAN 信号（BGA108 封装）

管脚名称	管脚序号	复用管脚/赋值	管脚类型	缓冲类型 [a]	功能描述
CAN0Rx	G1 L5 L6 A6	PD0(2) PA4(5) PA6(6) PB4(5)	I	TTL	CAN 模块 0 接收信号
CAN0Tx	G2 M5 M6 B7	PD1(2) PA5(5) PA7(6) PB5(5)	O	TTL	CAN 模块 0 发送信号
CAN1Rx	M9	PF0(1)	I	TTL	CAN 模块 1 接收信号
CAN1Tx	H12	PF1(1)	O	TTL	CAN 模块 1 发送信号

a：“TTL”表示该管脚兼容 TTL 电平标准。

17.3 功能描述

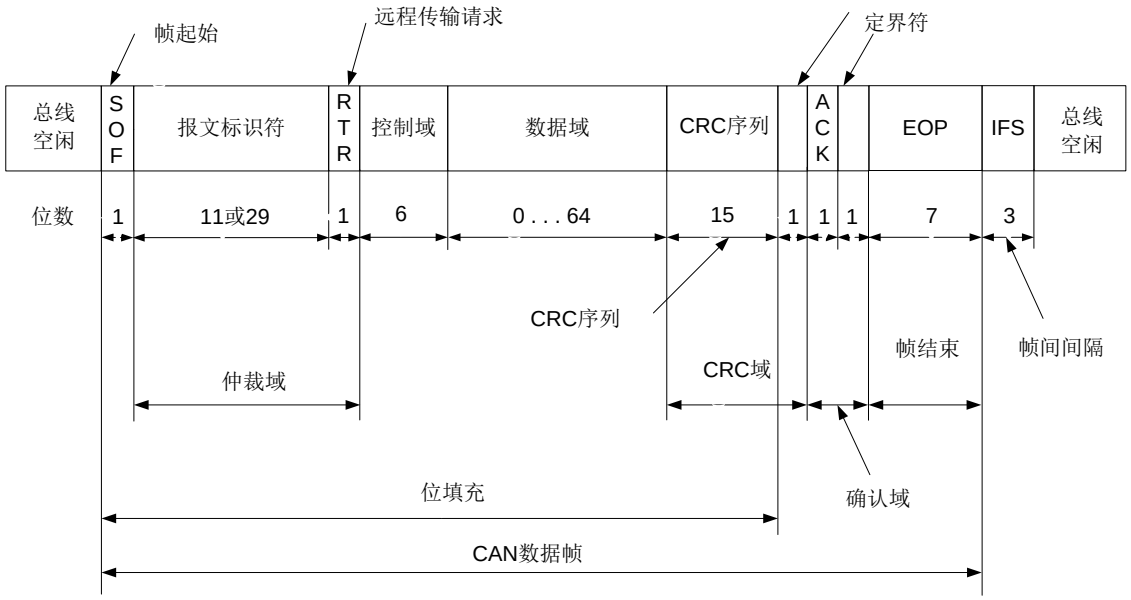
Stellaris® CAN 控制器符合 CAN 总线 2.0 版协议（A/B），支持传输的报文类型包括数据帧、远程帧、错误帧以及过载帧，标识域的长度可以是 11 位（标准协议）或 29 位（扩展协议），位速率最高可达 1Mbps。

CAN 模块由 3 个主要部分组成：

- CAN 协议控制器及报文处理器
- 报文存储器
- CAN 寄存器接口

数据帧中包含要发送的数据，而远程帧不包含任何数据、仅用于向其它节点请求指定的报文对象。CAN 数据帧/远程帧的结构如图 17-2 所示。

图 17-2. CAN 数据帧/远程帧结构



协议控制器从 CAN 总线上接收和发送串行数据，并将数据传递给报文处理器。报文处理器基于当前的过滤设置以及报文对象存储器中的标识符，将适当的报文内容载入与之对应的报文对象。报文处理器还负责根据 CAN 总线的事件产生中断。

报文对象存储器是一组 32 个完全相同的存储模块，可为每个报文对象保存其当前的配置、状态以及实际数据。这些报文对象可通过两组 CAN 报文对象寄存器接口中的任何一组进行访问。

报文存储器在 Stellaris 存储器映射中是无法直接访问的，因此 Stellaris CAN 控制器提供了间接访问的接口：用户可通过两个 CAN 接口寄存器组与报文对象进行通讯。任何报文对象都必须通过这两个接口寄存器组进行访问。两组报文对象接口也可以并行访问 CAN 控制器报文对象，因此当多个报文对象同时包含亟待处理的新信息时完全可以并行处理。一般我们用一组接口专司发送，另一组接口专司接收。

17.3.1 初始化

要正常使用 CAN 控制器，必须先通过 **RCGC0** 寄存器（见第 273 页）使能外设时钟。此外，还必须通过 **RCGC2** 寄存器（见第 293 页）使能相关 GPIO 模块的时钟。至于需要开启哪些 GPIO 模块的时钟，可参见第 1252 页的表 24-4。此后，应将相关 GPIO 管脚的 AFSEL 位（见第 429 页）置位，并配置 **GPIOPCTL** 寄存器的 PMCn 位域，将 CAN 信号赋给相应的管脚。详见第 447 页以及第 1261 页的表 24-5。

将 **CAN 控制寄存器（CANCTL）**的 INIT 位置位（该标志位可通过软件置位，也会在硬件复位后自动置位）即可开始软件初始化；或当 CAN 控制器进入离线状态（发送器的错误计数超过 255）后也会开始软件初始化。当 INIT 位置位时，所有与 CAN 总线正在进行的收发任务都将中止，并且 CANnTx 脚将保持拉高。进入初始化状态并不会改变 CAN 控制器、报文对象或错误计数器的配置。不过，某些配置寄存器只能在初始化状态下才能访问。

在初始化 CAN 控制器时，应设置 **CAN 位定时寄存器（CANBIT）**并依次配置每个报文对

象。如果不需使用某个报文对象，只需在 **CAN IFn 仲裁寄存器 2（CANIFnARB2）**中将其标记为无效（将相应 MSGVAL 位清零）即可。如果需要使用某个报文对象，则应当详细初始化整个报文对象，因为一旦报文对象的某个域包含无效的数据将会导致无法预料的后果。必须将 **CANCTL** 寄存器的 INIT 位和 CCE 位都置位，这样才能访问 **CANBIT** 寄存器以及 **CAN 波特率预分频系数扩展寄存器（CANBRPE）**以配置位定时参数。要退出初始化状态，必须将 INIT 位清零。此后，内部比特流处理器（Bit Stream Processor，简写为 BSP）首先等待一个连续 11 个隐性位序列的产生（表明总线处于空闲状态），依此序列实现与 CAN 总线数据传输的同步，之后才能参与总线活动、开始报文传输。报文对象随时都可以初始化，与 CAN 控制器的初始化状态无关。不过，在开始传输报文之前必须保障所有报文对象已经配置有合适的标识符或已经被设置为无效。在正常工作时如果需要修改某个报文对象的配置，应先将 **CANIFnARB2** 寄存器的相应 MSGVAL 位清零，暂时将该报文对象设为无效；待修改配置后再重新将 MSGVAL 位置位，将该报文对象设为有效。

17.3.2 基本操作

CAN 模块提供两组 CAN 接口寄存器（**CANIF1x** 和 **CANIF2x**）用于访问报文 RAM 中的报文对象。CAN 控制器自动将与报文 RAM 的数据交互映射为与寄存器的数据交互。两组寄存器相互独立而又完全相同，可用于实现连续会话。一般来说，用一组接口专司发送，另一组接口专司接收。

一旦 CAN 模块完成初始化并且 **CANCTL** 寄存器的 INIT 位清零，CAN 模块将自动与 CAN 总线同步，同步后将开始传输报文。收到的每个报文都需要经过报文处理器的验收过滤，如果能够通过才会保存到由 **CAN IFn 指令请求寄存器（CANIFnCRQ）**MNUM 位域所指定的报文对象中。整个报文（包括所有仲裁位、数据长度码以及 8 个数据字节）都将保存在报文对象中。假如使用到标识符掩码（**CAN IFn 掩码寄存器 1** 和 **CAN IFn 掩码寄存器 2（CANIFnMSKn）**），那么报文对象中掩码设置为“无关”的仲裁位则可能会被覆盖。

CPU 随时可以通过 CAN 接口寄存器读写任一报文。当出现同时访问的情况时，由报文处理器负责保障数据的一致性。

报文对象的发送是在由管理 CAN 硬件的软件所管理的。报文对象既可以仅用于单次数据传输、也可以作为永久报文对象实现周期性的响应。永久报文对象的所有仲裁部分及控制部分是预先设好的固定值，只需不断刷新各数据字节。要启动发送报文，应将 **CAN 发送请求寄存器 n（CANTXRQn）**的相应 TXRQST 位置位、并且将 **CAN 新数据寄存器 n（CANNWDAn）**的相应 NEWDAT 位置位。假如多个要发送的报文都使用同一报文对象（当报文对象不够用时），必须在请求发送报文之前完整配置报文对象。

CAN 控制器允许同时请求发送任意数量的报文对象；当多个报文对象同时等待发送时，发送顺序是按照内部优先级定义的，即按照报文对象的序号（MNUM）排序，其中 1 是最高优先级、32 是最低优先级。报文可以随时刷新或设置为无效，即使其发送请求尚被挂起也是如此。当报文的发送请求已挂起并且尚未开始时，刷新报文内容时将丢弃旧内容。按照报文对象的配置，当收到标识符相同的远程帧时能够自动请求发送报文。

当收到匹配的远程帧时可以自动开始请求发送。要启用这种模式，应将 **CAN IFn 报文控制寄存器（CANIFnMCTL）**的 RMTEN 位置位。当收到匹配的远程帧时，会使得相应的 TXRQST

位置位，并且报文对象将自动发送其数据部分或产生中断表示有远程帧的请求。远程帧可以是严格限定的某个报文标识符，也可以是报文对象所定义的一个标识符范围。通过 **CAN 掩码寄存器（CANIFnMSKn）**配置哪一组帧被识别为远程帧请求。**CANIFnMSKn** 寄存器的 MSK 位域通过 **CANIFnMCTL** 寄存器的 UMASK 位使能。如果准备以 29 位扩展标识符触发远程帧请求，则应将 **CANIFnMSK2** 寄存器的 MXTD 位置位。

17.3.3 报文对象的发送

假如 CAN 模块的内部发送移位寄存器已经准备好装载，并且 CAN 接口寄存器与报文 RAM 之前无数据传输，那么已挂起发送请求并且优先级最高的有效报文对象将会被报文处理器装载入发送移位寄存器中，并开始发送流程。该报文对象在 **CANNWDA** 寄存器中对应的 NEWDAT 位将清零。当发送成功后，如果自开始发送以后该数据对象没有写入新的数据，则 **CANTXRQn** 寄存器的对应 TXRQST 位将清零。如果 CAN 控制器配置为每成功发送一个报文对象后即产生一次中断，并且 **CAN IFn 报文控制寄存器（CANIFnMCTL）**的 TXIE 位置位，则每次成功发送一个报文对象后 **CANIFnMCTL** 寄存器的 INTPND 位都将置位。如果 CAN 模块丢失仲裁（竞争总线失败）或在发送期间产生错误，该报文将在 CAN 总线一有空闲时立即重新发送。如果与此同时有更高优先级的报文对象请求发送，那么将按照优先级的顺序依次发送报文。

17.3.4 待发送报文对象的配置

可按照下面的步骤配置要发送的报文对象：

1. 在 **CAN IFn 指令掩码寄存器（CANIFnCMSK）**中：
   - 将 WRNRD 位置位，表明是对 **CANIFnCMSK** 寄存器的写操作；通过 MASK 位指定是否将报文对象的 IDMASK、DIR 和 MXTD 发送给 **CAN IFn** 寄存器组；
   - 通过 ARB 位指定是否将报文对象的 ID、DIR、XTD 和 MSGVAL 发送给接口寄存器；
   - 通过 CONTROL 位指定是否将控制位发送给接口寄存器；
   - 通过 CLRINTPND 位指定是否将 **CANIFnMCTL** 寄存器的 INTPND 位清零；
   - 通过 NEWDAT 位指定是否将 **CANNWDAn** 寄存器的 NEWDAT 位清零；
   - 通过 DATAA 和 DATAB 位指定要发送哪些数据位
2. 通过 **CANIFnMSK1** 寄存器的 MSK[15:0]位域来指定 29 位或 11 位报文标识符中哪些位用于验收过滤。请注意此寄存器的 MSK[15:0]位域只对应于 29 位标识符的[15:0]位域，而并不适用于 11 位标识符。当 MSK 位域的值为 0x00 时，表示任何报文均可通过验收过滤。同时应注意到，要想将这些位用于验收过滤，还应将 **CANIFnMCTL** 寄存器的 UMASK 位置位，使能验收过滤功能；
3. 通过 **CANIFnMSK2** 寄存器的 MSK[12:0]位域来指定 29 位或 11 位报文标识符中哪些位用于验收过滤。请注意此寄存器的 MSK[12:0]位域对应于 29 位标识符的[28:16]，MSK[12:2]位域对应于 11 位标识符的[10:0]。此外通过 MXTD 和 MDIR 位来指定 XTD

位和 DIR 位是否参与验收滤波。同时应注意到，要想将这些位用于验收过滤，还应将 **CANIFnMCTL** 寄存器的 UMASK 位置位，使能验收过滤功能；

4. 对 29 位标识符，配置 **CANIFnARB1** 寄存器的 ID[15:0]位域，对应于报文标识符的[15:0]，配置 **CANIFnARB2** 寄存器的 ID[12:0]位域，对应于报文标识符的[28:16]。此外应将 XTD 位置位，表示采用扩展标识符；将 DIR 位置位，表示发送；将 MSGVAL 位置位，表示该报文对象有效；

5. 对 11 位标识符，无需考虑 **CANIFnARB1** 寄存器，只需配置 **CANIFnARB2** 寄存器的 ID[12:2]位域，对应于报文标识符的[10:0]。此外应将 XTD 位清零，表示采用标准标识符；将 DIR 位置位，表示发送；将 MSGVAL 位置位，表示该报文对象有效；

6. 在 **CANIFnMCTL** 寄存器中

   - 可选设置：将 UMASK 位置位，使能掩码（**CANIFnMSK1** 寄存器和 **CANIFnMSK2** 寄存器中的 MSK、MXTD、MDIR）以便验收过滤；
   - 可选设置：将 TXIE 位置位，这样每当成功发送一帧后，INTPND 位自动置位；
   - 可选设置：将 RMTEN 位置位，这样每当收到匹配的远程帧后 TXRQST 位自动置位，从而允许自动传输；
   - 将 EOB 位置位，表明只发送单个报文对象；
   - 配置 DLC[3:0]位域，指定数据帧的长度。此时注意不要将 NEWDAT、MSGLST、INTPND 和 TXRQST 位置位

7. 将要发送的数据依次填入 **CAN IFn 数据寄存器**（**CANIFnDA1**、**CANIFnDA2**、**CANIFnDB1**、**CANIFnDB2**）中。CAN 数据帧的第 0 字节保存在 **CANIFnDA1** 寄存器的 DATA[7:0]位域中；

8. 在 **CAN IFn 指令请求寄存器**（**CANIFnCRQ**）的 MNUM 位域填写要发送的报文对象序号；

9. 当所有配置都成功完成后，将 **CANIFnMCTL** 寄存器的 TXRQST 位置位。此标志位置位后，报文对象就会在总线空闲时按照优先级进行发送了。请注意，如果 **CANIFnMCTL** 寄存器的 RMTEN 位置位，则在接收到匹配的远程帧后也会自动开始发送报文。

17.3.5 待发送报文对象的刷新

CPU 随时可以通过 CAN 接口寄存器刷新待发送报文对象的数据字节，而且在刷新之前并不一定需要将 **CANIFnARB2** 寄存器的 MSGVAL 位或 **CANIFnMCTL** 寄存器的 TXRQST 位清零。

在 **CANIFnDAn / CANIFnDBn** 寄存器的内容传递给报文对象之前，即使只需刷新其中的几个数据字节，也必须保证 4 个字节全都有效。软件处理时既可以将所有 4 个字节直接写入 **CANIFnDAn / CANIFnDBn** 寄存器中，也可以先将报文对象载入 **CANIFnDAn / CANIFnDBn** 寄存器、然后再写入数据字节。

要想只更新报文对象中的数据字节，应将 **CANIFnMSKn** 寄存器的 WRNRD、DATAA 和 DATAB

位置位，随后将刷新数据写入 **CANIFnDA1**、**CANIFnDA2**、**CANIFnDB1**、**CANIFnDB2** 寄存器中，并将报文对象的序号写入 **CAN IFn 指令请求寄存器（CANIFnCRQ）**的 MNUM 位域中。要想尽快开始发送新的数据，应将 **CANIFnMSKn** 寄存器的 TXRQST 位置位。

如果在刷新数据时此报文对象的前一组数据正在发送，那么当其发送完成后会自动将 **CANIFnMCTL** 寄存器的 TXRQST 位清零，导致填充新数据的报文对象停止发送。为了避免出现这种情况，应同时将 **CANIFnMCTL** 寄存器的 NEWDAT 位和 TXRQST 位置位。只要这两个位同时置位，则当开始新的发送过程后，NEWDAT 位将立即清零。

17.3.6 已接收报文对象的接受

当已接收报文的仲裁域和控制域(**CANIFnARB2** 寄存器的 ID 位域和 XTD 位、**CANIFnMCTL** 寄存器的 RMTEN 位和 DLC[3:0]位域）已经全部移入 CAN 控制器后，报文处理器将开始扫描报文 RAM 搜索与之匹配的有效报文对象。在扫描报文 RAM 搜索匹配报文对象时，控制器通过 **CANIFnMSKn** 寄存器的掩码位域进行验收滤波(必须先通过 **CANIFnMCTL** 寄存器的 UMASK 位使能此功能)。控制器将从报文对象 1 开始逐个将有效报文对象与收到的报文进行比对，以期在报文 RAM 中找到匹配的报文对象。当找到匹配的报文对象后，扫描过程就此结束，随后报文处理器将根据收到的是数据帧还是请求帧予以分别处理。

17.3.7 接收数据帧

报文处理器将来自 CAN 控制器接收移位寄存器报文保存入报文 RAM 中的匹配报文对象中，保存的内容包括数据字节、所有仲裁位、DLC 位域。即使采用了仲裁掩码，数据字节也是与标识符相关联的。**CANIFnMCTL** 寄存器的 NEWDAT 位置位时表示已经收到了新数据。CPU 在读取此报文对象之后必须将此标志位清零，告诉控制器已经接收处理此报文、该条缓冲可用于接收新的报文了。假如当 NEWDAT 位已经置位时 CAN 控制器又接收了一条新的报文，那么 **CANIFnMCTL** 寄存器的 MSGLST 位将自动置位，表示前一条数据丢失。假如系统需要在成功接收完一帧后产生中断，则应将 **CANIFnMCTL** 寄存器的 RXIE 位置位。此后当某个报文对象成功接收一帧后，该寄存器的 INTPND 位将自动置位，使得 **CANINT** 寄存器指向该报文对象。注意该报文对象的 TXRQST 位应当清零，防止发送远程帧。

17.3.8 接收远程帧

远程帧中不包含数据，而是通知其它节点指示应当发送哪一报文对象。当收到远程帧时，应考虑匹配报文对象的 3 种不同配置：

表 17-3. 报文对象的配置

CANIFnMCTL 中的配置	描述
■ **CANIFnARB2** 寄存器中的 DIR = 1（方向为发送） ■ RMTEN = 1（当收到匹配的远程帧时，**CANIFnMCTL** 寄存器的 TXRQST 位自动置 1 请求发送） ■ UMASK = 1 或 0	当收到匹配的远程帧时，该报文对象的 TXRQST 位将置位，报文对象的其余部分无变化。控制器将尽快开始自动发送该报文对象中的数据帧。
■ **CANIFnARB2** 寄存器中的 DIR = 1（方向为发送） ■ RMTEN = 0（当收到匹配的远程帧时，**CANIFnMCTL** 寄存器的 TXRQST 位无变化） ■ UMASK = 0（忽略 **CANIFnMSKn** 寄存器中的掩码设置）	当收到匹配的远程帧时，该报文对象的 TXRQST 位保持不变，远程帧将被忽略。在这种模式下相当于禁用了远程帧功能，控制器不会发送匹配的数据帧，也不会留下任何标志表明曾经接收过远程帧。
■ **CANIFnARB2** 寄存器中的 DIR = 1（方向为发送） ■ RMTEN = 0（当收到匹配的远程帧时，**CANIFnMCTL** 寄存器的 TXRQST 位无变化） ■ UMASK = 1（采用 **CANIFnMSKn** 寄存器中的 MSK、MSKD、MDIR 掩码用于验收过滤）	当收到匹配的远程帧时，该报文对象的 TXRQST 位将清零。来自移位寄存器的仲裁域与控制域（ID + XTD + RMTEN + DLC）将保存到报文 RAM 中的相应报文对象中，并且将该报文对象的 NEWDAT 位置位。报文对象的数据域保持不变；该远程帧按照收到数据帧进行处理。当 Stellaris 控制器并未有任何待发送的数据却收到来自其它 CAN 节点的远程数据请求时，这种模式将显得十分重要。此时软件应根据请求的对象自行填充数据并应答此远程帧。

17.3.9 接收/发送优先级

报文对象的接收/发送优先级是由报文对象的序号决定的，报文对象 1 的优先级总是最高，报文对象 32 的优先级总是最低。假如当前挂起的发送请求不止一个，那么将优先发送序号最小的那个报文对象。请注意，这里所说的优先级与 CAN 总线上通过报文标识符强制实现的总线优先级没有半点关系。举例来说，假如报文对象 1 和报文对象 2 同时有待发送的有效报文，那么报文对象 1 将总是优先发送，不管其报文标识符的总线优先级如何。

17.3.10 接收报文对象的配置

可按照下面的步骤配置接收报文对象：

1. 按照第 867 页的“待发送报文对象的配置”一节，配置 **CAN IFn 指令掩码寄存器（CANIFnCMASK）**。其中 WRNRD 位必须置位，表明向报文 RAM 执行写操作；
2. 按照第 867 页的“待发送报文对象的配置”一节，通过配置 **CANIFnMSK1** 和 **CANIFnMSK2** 寄存器选择将哪些位用于验收过滤。请注意还应将 **CANIFnMCTL** 寄存器的 UMASK 位置位，才能对这些位真正实现验收过滤；
3. 配置 **CANIFnMSK2** 寄存器的 MSK[12:0]位域，定义在 29 位或 11 位标识符中哪些仲裁位用于验收过滤。请注意对于 29 位报文标识符来说，MSK[12:0]位域对应于标识符的[28:16]；对于 11 位报文标识符来说，MSK[12:2]位域对应于标识符的[10:0]。通过

MTXD和MDIR位可选择是否将XTD位和DIR位也用于验收过滤。若此寄存器写入0x00，则所有报文均可通过验收过滤。此外还要注意，为了使这些用于验收过滤的掩码位生效，必须将 **CANIFnMCTL** 寄存器的 UMASK 位置位；

4. 按照第 867 页的“待发送报文对象的配置”一节，配置 **CANIFnARB1** 和 **CANIFnARB2** 寄存器。配置 XTD 位和 ID 位域用于验收滤波；将 MSGVAL 位置位表示报文有效；将 DIR 位清零表示用于接收；

5. 在 **CANIFnMCTL** 寄存器中：

   - 可选设置：将 UMASK 位置位，使能验收过滤的掩码部分（**CANIFnMSK1** 和 **CANIFnMSK2** 寄存器中的 MSK、MXTD、MDIR 位域）；
   - 可选设置：将 RXIE 位置位，这样每当成功接收一帧后 INTPND 位将自动置位；
   - 将 RMTEN 位清零，使得 TXRQST 位保持不变；
   - 对于单个报文对象，应将 EOB 位置位；
   - 配置 DLC[3:0]位域，指定数据帧的数据域长度

   配置时应注意不要将 NEWDAT、MSGLST、INTPND、TXRQST 位置位。

6. 在 **CAN IFn 指令请求寄存器（CANIFnCRQ）**的 MNUM 位域中写入要接收报文对象的序号。当 CAN 总线上有匹配的帧时，将立即开始接收报文对象。

当报文处理器向数据对象保存一个数据帧时，会向其写入接收到的数据长度码（Data Length Code，简写为 DLC）以及 8 个数据字节（写入 **CANIFnDA1**、**CANIFnDA2**、**CANIFnDB1**、**CANIFnDB2** 寄存器中）。CAN 数据帧的数据字节 0 将保存在 **CANIFnDA1** 寄存器的 DATA[7:0]位域中。假如数据长度码小于 8，那么报文对象的剩余数据字节将以随机值予以覆盖。

通过 **CAN 掩码寄存器（CANIFnMSKn）**可以允许某个报文对象只接收某些数据帧。而 **CANIFnMCTL** 寄存器的 UMASK 位控制是否采用 **CANIFnMSKn** 寄存器的 MSK 位域过滤报文。如果报文对象用于接收扩展帧，则 **CANIFnMSK2** 寄存器中的 MXTD 位应当置位。

17.3.11 已接收报文对象的处理

报文处理器状态机已足可保障数据的一致性，因此 CPU 随时都能通过 CAN 接口寄存器组来读取已接收的报文。

一般来说，CPU 应先向 **CANIFnCMSK** 寄存器写入 0x007F，随后向 **CANIFnCRQ** 寄存器写入报文对象的序号。这个操作组合将使已接收的整条报文从报文 RAM 移入报文缓冲寄存器中（**CANIFnMSKn**、**CANIFnARBn**、**CANIFnMCTL** 寄存器）。此外，报文 RAM 中的 NEWDAT 位和 INTPND 位也会清零，确认该报文对象已经被读出，并清除该报文对象所挂起的中断。

假如报文对象设置了验收过滤的掩码，那么可以通过 **CANIFnARBn** 寄存器查询已接收的报文在掩码操作之前的完整 ID。

CANIFnMCTL 寄存器的 NEWDAT 位能够指示出自从上一次读取此报文对象之后是否又收到

了新的报文。**CANIFnMCTL** 寄存器的 MSGLST 位能够指示出自从上一次读取此报文对象之后是否收到了不止一条报文。MSGLST 位不会自动清零，因此软件必须在每次读取其状态后手动将其清零。

通过运用远程帧，CPU 可以向 CAN 总线的其它节点请求新的数据。如果某个报文对象的方向已经设置为接收，则将该报文对象的 TXRQST 位置位时会以其报文标识符发送一个远程帧。此远程帧将触发其它 CAN 节点在总线上发送标识符匹配的数据帧。假如在远程帧发送完成之前收到了匹配的数据帧，那么 TXRQST 位将自动复位。这样在 CAN 总线上的其它节点已经发送数据帧（稍微早于预期的时间）时，可以避免发生数据丢失。

17.3.11.1 FIFO 缓冲区的配置

除 **CANIFnMCTL** 寄存器的 EOB 位以外，属于某个 FIFO 缓冲区的接收报文对象的配置方式与单个接收报文对象的配置方法是一样的（参见第 870 页的“接收报文对象的配置”一节）。要将 2 个或 2 个以上的报文对象连锁到同一个 FIFO 缓冲区中，这些报文对象的标识符及掩码（如果有的话）都应当编程为相同的值。由于报文对象已经有固定的优先级，因此序号最小的报文对象必然是 FIFO 缓冲区中的首个报文对象。FIFO 中除了最后一个报文对象外，其它报文对象的 EOB 位必须清 0。最后一个报文对象的 EOB 位置位即可表明它是缓冲区中的最后一个单元。

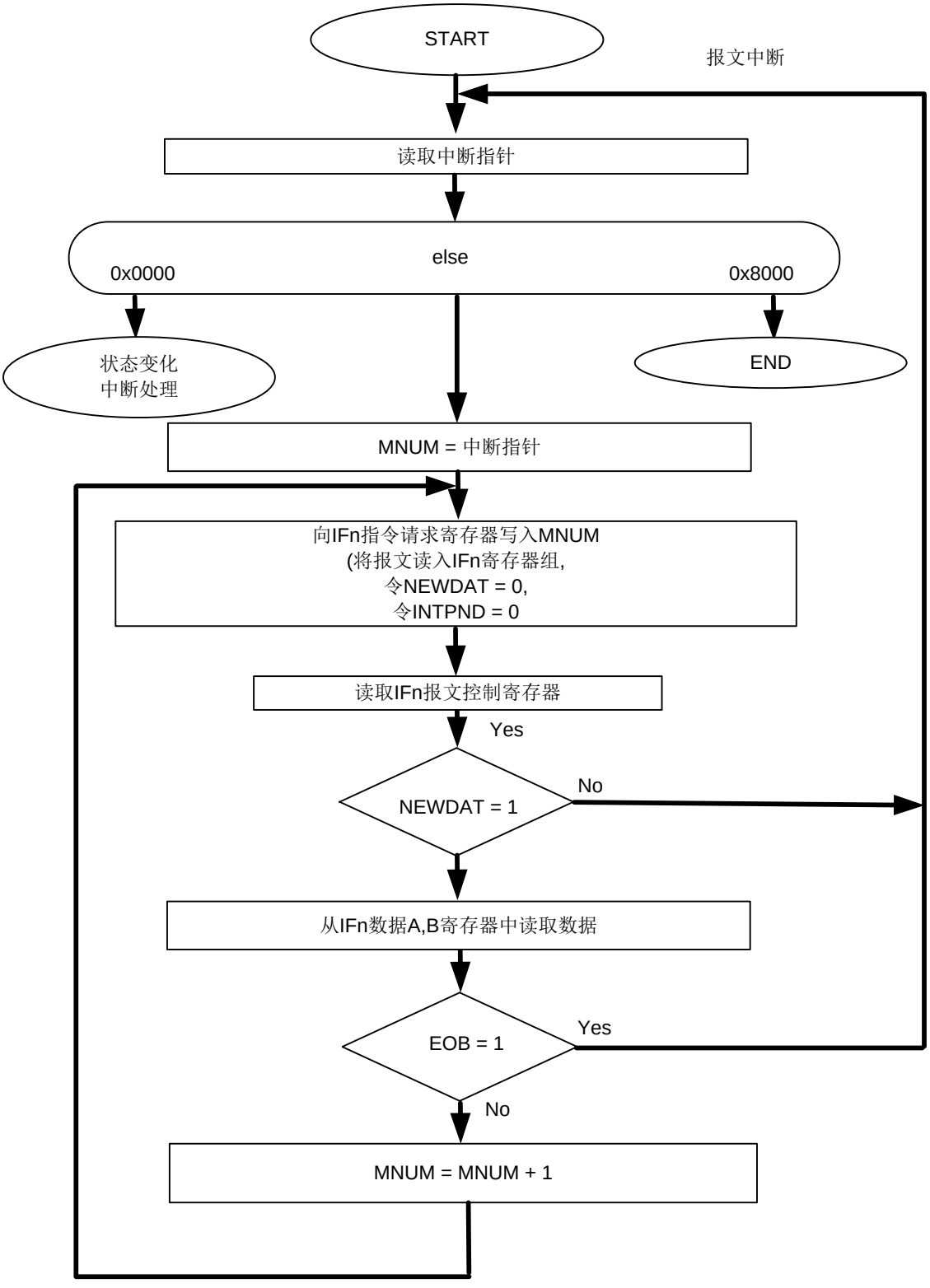
17.3.11.2 采用 FIFO 缓冲区的报文接收过程

当接收到若干个标识符与 FIFO 缓冲区匹配的报文时，这些报文将从序号最低的报文对象开始依次写入 FIFO 缓冲区内。只要 FIFO 中的某个报文对象保存了一条报文，该报文对象中 **CANIFnMCTL** 寄存器的 NEWDAT 位便会自动置位。若在 EOB 为 0 时 NEWDAT 位置位，该报文对象将自动锁定，此后报文处理器将无法再对其写入，直到 CPU 将 NEWDAT 位清零后才会解除锁定状态。匹配的报文将依次保存入 FIFO 缓冲区中，一直到此 FIFO 缓冲区的最后一个报文对象。当 FIFO 缓冲区填满后，除非前面的所有报文对象已经将 NEWDAT 位清零解除锁定，否则只要再收到匹配的报文都将写入最后一个报文对象中，因此最后一个报文对象的内容可能会被反复覆盖。

17.3.11.3 FIFO 缓冲区的读取

当 CPU 向 **CANIFnCRQ** 寄存器写入序号读取某个报文对象的内容时，如果该报文对象位于 FIFO 缓冲区中，则应将 **CANIFnCMSK** 寄存器的 TXRQST 位和 CLRINTPND 位置位，这样 **CANIFnMCTL** 寄存器的 NEWDAT 位和 INTPEND 位才会在读操作后自动清零。在手动将其清零之前，**CANIFnMCTL** 寄存器中的这两个位始终反映该报文对象的真实状态。要确保 FIFO 缓冲区能够正确工作，CPU 应当按照序号由小到大的顺序依次读取各报文对象的内容。当从 FIFO 缓冲中读取数据报文时，用户必须明白：此时如果再接收到新的报文，将可能写入任何一个 **CANIFnMCTL** 寄存器 NEWDAT 位已被清零的报文对象中。因此这之后的 FIFO 中接收报文的顺序是无法得到保障的。第 873 页的图 17-3 描绘出 CPU 应如何正确处理 FIFO 缓冲区中的一组报文对象。

图 17-3. FIFO 缓冲中的报文对象



17.3.12 中断的处理

如果有多个中断同时挂起，**CAN 中断寄存器（CANINT）**将始终指向优先级最高的已挂起中断，而并非按照时间顺序指向最新或最旧中断。状态中断始终具有最高优先级；而报文中

断的优先级由报文对象序号确定，序号最小的报文对象具有最高中断优先级。报文中断既可以通过将相应报文对象 **CANIFnMCTL** 寄存器的 INTPND 位清零予以清除，也可以通过读取 **CAN 状态寄存器（CANSTS）**予以清除。状态中断只能通过读取 **CANSTS** 寄存器予以清除。

CANINT 寄存器中的 INTID 位域可指示出中断的产生原因。当没有任何中断挂起时，该寄存器的回读值为 0x0000。只要 INTID 位域的值不是 0x0000，即表明有中断挂起。若 **CANCTL** 寄存器的 IE 位置位，则允许 CAN 控制器向中断控制器产生中断。一旦产生中断将保持激活状态，直到 INTID 位域为 0x0000（所有中断源都已清除：中断的条件已经解除）或 IE 位被清零（也就是禁用 CAN 控制器的中断）时才会清除。

CANINT 寄存器的 INTID 位域指向挂起的最高优先级报文中断。**CANCTL** 寄存器的 SIE 位用于控制是否允许 **CANSTS** 寄存器中 RXOK、TXOK、LEC 位的改变产生中断。**CANCTL** 寄存器的 EIE 位用于控制是否允许 **CANSTS** 寄存器中 BOFF、EWARN 位的改变产生中断。**CANCTL** 寄存器的 IE 位用于控制是否将 CAN 控制器产生的原始中断发送给中断控制器。即使 **CANCTL** 寄存器的 IE 位清零，**CANINT** 寄存器也会自动更新，只不过中断不会指示给 CPU。

当 **CANINT** 寄存器的值为 0x8000 时，表示 CAN 模块刷新了 **CANSTS** 寄存器（但并不一定改变了其内容）并挂起了中断，因此产生的中断是错误中断或状态中断。对 **CANSTS** 寄存器执行写操作即可将其中的 RXOK、TXOK、LEC 位清零，但如果想要清除状态中断的中断源，唯一的办法是读 **CANSTS** 寄存器。

在进行中断处理时可通过以下两种方法确定中断源：一种方法是读取 **CANINT** 寄存器的 INTID 位域，确定目前挂起的最高优先级中断；另一种方法是读取 **CAN 报文中断挂起寄存器（CANMSGnINT）**，查看所有挂起中断的报文对象。

当某个报文对象是中断源时，若 **CANIFnCMSK** 寄存器的 CLRINTPND 位置位，那么在中断服务子程序中读出该报文的同时会自动清除该报文对象的 INTPND 位。一旦 INTPND 位清零，**CANINT** 寄存器的内容将变成下一个挂起中断的报文对象序号。

17.3.13 测试模式

CAN 模块提供测试模式，方便进行各种诊断。将 **CANCTL** 寄存器的 TEST 位置位即可进入测试模式。进入测试模式后，可通过 **CAN 测试寄存器（CANTST）**的 TX[1:0]、LBACK、SILENT 以及 BASIC 等位域将 CAN 控制器配置成工作于各种不同的诊断模式下。**CANTST** 寄存器的 RX 位能够监控 CANnRx 管脚状态。若 TEST 位清 0，则 **CANTST** 寄存器的所有设置全部被禁用。

17.3.13.1 安静模式

安静模式下 CAN 控制器不会发送显性位（确认位、错误帧等等），因此在分析 CAN 总线数据流的同时不会对 CAN 总线通讯造成任何影响。将 **CANTST** 寄存器的 SILENT 位置位即可进入安静模式。在安静模式下，CAN 控制器能够接收有效的数据帧和远程帧，但本身只能在 CAN 总线上发送隐性位，并且无法发送任何报文。即使 CAN 控制器必须发送显性位（确认位、过载位或有效错误标志位），也仅在控制器内部连接。也就是说只有 CAN 控制器本身能够监控到发送的显性位，在 CAN 总线上仍然发送的是隐形位。

17.3.13.2 环回模式

环回模式用于实现自检功能。在环回模式下，CAN 控制器在内部将 CANnTx 与 CANnRx 信号连接到一起，将自己发送的报文视为接收到的报文并存储（假如能够顺利通过验收过滤的话）到报文缓冲中。将 **CANTST** 寄存器的 LBACK 位置位即可进入环回模式。为了彻底隔绝外部事件的影响，在环回模式下 CAN 控制器将会忽略确认错误（在数据帧/远程帧的确认位时间内采样到隐性位），并且也会忽略 CANnRx 管脚的实际电平。不过，CANnTx 管脚上仍然能够监控到发送的报文。

17.3.13.3 环回+安静模式

如果混合使用环回模式与安静模式，CAN 控制器既能实现在线自检、也不会影响与 CANnRx 及 CANnTx 管脚连接的 CAN 总线系统的运行。在这种模式下，CANnRx 信号并不连入 CAN 控制器，并且 CANnTx 信号始终保持隐性位状态。将 **CANTST** 寄存器的 LBACK 和 SILENT 位同时置位即可进入这种模式。

17.3.13.4 基本模式

所谓基本模式就是 CAN 控制器不使用报文 RAM 的工作模式。在基本模式下，CANIF1 寄存器组用作发送缓冲。将 **CANIF1CRQ** 寄存器的 BUSY 位置位即会请求发送 IF1 寄存器组的内容。当 BUSY 位置位时 CANIF1 寄存器组即被锁定不可更改，此时 BUSY 位还能指示出有挂起的发送。只要 CAN 总线有空闲，CANIF1 寄存器组就会把内容载入 CAN 控制器的移位寄存器并开始发送。发送结束后 BUSY 位将自动清零，并且 CANIF1 寄存器组也将被解除锁定状态。若当前有发送挂起且 CANIF1 寄存器组被锁定，随时都能通过将 **CANIF1CRQ** 寄存器的 BUSY 位清零而中止发送。如果报文因为丢失仲裁（竞争总线失败）或者发送出错而等待重发时，只要 CPU 将 BUSY 位清零，就能禁止自动重发此报文。

CANIF2 寄存器组用作接收缓冲。当接收一条报文后，移位寄存器中的内容将不经过任何验收过滤、直接保存到 CANIF2 寄存器组中。此外，在报文传输期间可以监视移位寄存器的实际内容。每当 **CANIF2CRQ** 寄存器的 BUSY 位置位（开始读取报文对象）时，移位寄存器的内容都将保存到 CANIF2 寄存器组中。

在基本模式下，所有与报文对象相关的控制位、状态位，以及 **CANIFnCMSK** 寄存器的控制位都无意义。**CANIFnCRQ** 寄存器的报文序号也无意义。在 **CANIF2MCTL** 寄存器中，NEWDAT 和 MSGLST 位保持其原有功能，DLC[3:0]位域能显示出收到的 DLC，其余控制位均清零。

将 **CANTST** 寄存器的 BASIC 位置位，即可使能基本模式。

17.3.13.5 发送控制

软件可以设置 CANnTx 信号的控制方式，有如下 4 种不同的方式可选：

- 由 CAN 控制器控制 CANnTx 管脚信号；
- CANnTx 信号按照采样点驱动输出，用以监控位定时；
- CANnTx 管脚始终驱动输出低电平；
- CANnTx 管脚始终驱动输出高电平；

最后两项功能结合可读的 CAN 接收管脚 CANnRx，可用于校验 CAN 总线物理层是否工作正常。

发送控制功能是通过 **CANTST** 寄存器的 TX[1:0]位域进行选择的。CANnTx 信号的后三种测试功能与所有 CAN 协议功能均不相符，因此，不论是在正常的 CAN 报文传输模式下，还是在环回模式、安静模式或基本模式下，都必须将 TX[1:0]位域清 0。

17.3.14 位定时配置错误的注意事项

如果在配置 CAN 的位定时参数时出现微小的错误，即使其并未立即体现出总线故障，也可能会严重降低 CAN 总线的性能。在很多情况下，CAN 总线本身的位同步功能将会掩盖 CAN 位定时参数配置的不当，将其影响降低到只会偶尔产生错误帧的程度。但在仲裁（竞争总线）时，当两个或两个以上的 CAN 节点同时试图发送帧时，偏移的采样点可能导致其中一个收发器进入被动错误状态。这种偶发错误非常难于分析，需要对 CAN 节点内的位同步原理以及 CAN 节点与 CAN 总线的相互作用有详尽的了解才能予以断定。

17.3.15 位时间与位速率

CAN 总线支持的位速率从 1kbps 到 1000kbps 不等。CAN 网络中的每个成员都有自己的时钟发生电路。即使各 CAN 节点所采用的振荡器频率并不相同，也能够通过单独设置各个 CAN 节点的位定时参数，最终实现位速率的统一。

由于温度/电压波动以及器件本身老化导致频率总会存在微小偏差，因此这些振荡器不可能始终保持稳定。不过，只要偏差保持在指定的振荡器容差范围内，CAN 节点都可通过定期与总线上的比特流重新同步的方式为各种位速率进行纠偏。

按照 CAN 总线的规范，每个位时间被划分为 4 个时间段（见第 877 页的图 17-4），分别是：同步段、传播时间段、相位缓冲段 1、相位缓冲段 2。每个时间段由若干个基本时间份额组成，并且每个时间段的时间份额数均可编程设置（见第 877 页的表 17-4）。时间份额是位定时配置的基本单位，每个时间份额的长度（t_q）由 CAN 控制器的输入时钟（fsys）以及波特率预分频系数（Baud Rate Prescaler，简写为 BRP）共同决定：

$$t_q = BRP / fsys$$

其中 fsys 就是由 **RCC** 或 **RCC2** 寄存器（见第 229 页或第 237 页）设置的输入系统时钟频率。

同步段（Sync）是位时间的组成部分，在同步段内期望产生 CAN 总线电平的跳变沿；若跳变沿位于 Sync 之外，则将跳变沿与 Sync 的间隔称为该跳变沿的相位误差。

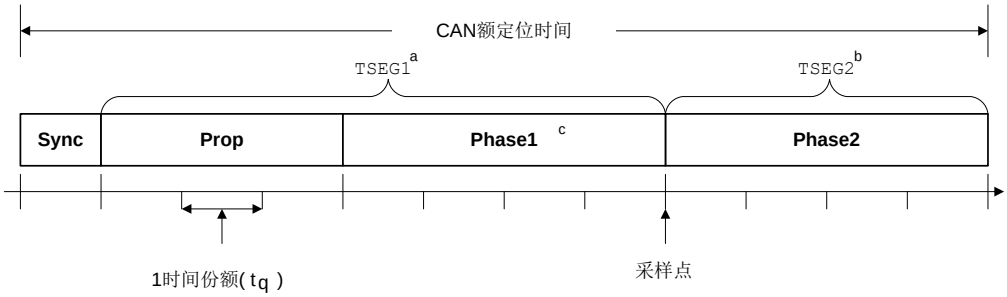
传播段（Prop）用于补偿 CAN 网络中的物理延迟时间。

相位缓冲段 1 和相位缓冲段 2 共同决定采样点在位时间中的位置。

同步跳转宽度（Synchronization Jump Width，简写为 SJW）定义了再同步的跳转距离，当出现跳变沿相位误差时，可按该距离在相位缓冲段的有效范围内移动采样点进行补偿。

当给定位速率后，可能有多种不同的位时间配置均可满足速率的要求；但是为了保障 CAN 网络正常工作，还应当慎重考虑并计算物理延迟时间以及振荡器的容差范围。

图 17-4. CAN 位时间示意图



a：TSEG1 = Prop + Phase1
b：TSEG2 = Phase2
c：Phase1 = Phase2 或 Phase1 + 1 = Phase2

表 17-4. CAN 协议参数及有效范围 [a]

参数名称	有效范围	注释
BRP	1~64	定义时间份额 t_q 的长度。结合 **CANBRPE** 寄存器可将此范围扩展到 1024。
Sync	$1t_q$	固定长度，将总线输入与系统时钟进行同步
Prop	$1\sim8t_q$	补偿 CAN 总线传输时的物理延时
Phase1	$1\sim8t_q$	根据同步的情况可能会适当延长
Phase2	$1\sim8t_q$	根据同步的情况可能会适当缩短
SJW	$1\sim4t_q$	不得超过任一相位缓冲段的长度

a：本表中的有效范围是 CAN 协议所要求的最小可编程范围。

位定时的配置是通过 **CANBIT** 寄存器中两个字节设置的。在 **CANBIT** 寄存器中，TSEG2、TSEG1、SJW 和 BRP 这 4 个位域的值必须填写为其实际取值减去 1；因此虽然上表中定义的有效范围是 1~n，实际对寄存器编程值的有效范围是 0~n–1。举例来说，同步跳转宽度 SJW（有效范围 1~4）在 SJW 位域中只需两个位即可表示。表 17-5 列出了 **CANBIT** 寄存器位域值与实际参数的关系。

表 17-5. CANBIT 寄存器位域有效范围

CANBIT 寄存器位域	取值含义
TSEG2	Phase2 – 1
TSEG1	Prop + Phase1 – 1
SJW	SJW – 1
BRP	BRP

因此，位时间的长度为（以编程值计算）：

$[TSEG1 + TSEG2 + 3] \times t_q$

或以实际值表述为：

$[Sync + Prop + Phase1 + Phase2] \times t_q$

CANBIT 寄存器中的数值将作为 CAN 协议控制器的配置输入。波特率预分频系数（由 BRP 位域配置）定义了每个时间份额（位时间的基本时间单元）的长度；位时序逻辑（由 TSEG1、TSEG2、SJW 配置）定义了每个位时间中包含多少个时间份额。

位时间的处理、采样点的位置计算以及偶发的同步动作均由 CAN 控制器控制，并且在每个时间份额都要进行判定。

CAN 控制器负责将报文转译为帧以及将帧转译为报文。此外，控制器还要负责以下任务：产生/丢弃帧中的固定格式位；插入/解出填充位；计算/校验 CRC 校验码；执行错误管理；确定采用何种类型的同步。一个位的值是在采样点接收或发送的。信息处理时间（Information Processing Time，简写为 IPT）是采样点后控制器做好向 CAN 总线发送下一个位所需的时间。IPT 期间控制器需要进行的工作包括：获取下一个位、进行 CRC 校验、判断是否需要位填充、根据需要产生错误标志。

IPT 与具体的应用有关，一般不应超过 $2t_q$；CAN 的 IPT 是 $0t_q$。IPT 长度应作为是 Phase2 可编程长度的下限值。当发生再同步时，缩短后的 Phase2 可能会小于 IPT，但这并不影响总线时序。

17.3.16 位定时参数的计算

通常在计算位定时参数时，首先应当确定需要达到的位速率或位时间。位时间（即位速率的倒数）**必须**是系统时钟周期的整数倍。

每个位时间由 4 到 25 个时间份额所组成。通过将各个时间段进行组合，就能得到所需的位时间，即重复以下步骤。

首先要确定 Prop 段。Prop 段的长度将决定 CAN 总线的最大延迟时间，反之也可以根据 CAN 总线的实际最大延迟时间来确定其长度。为保证 CAN 总线系统具有可扩展性，应明确定义最长总线距离下节点间的最大延时。Prop 的定义结果应转换为时间份额数（进位得到最接近的整数值，即 t_q 的整数倍）。

Sync 段的长度固定是 1 t_q。因此现在剩下的份额将由两个相位缓冲段瓜分。假如剩余的 t_q 数是偶数值，可以平均分配给两个相位缓冲段，即 Phase2 = Phase1；假如是奇数值，则 Phase2 = Phase1 + 1。

必须考虑到 Phase2 是有额定的最小长度的：Phase2 长度不得小于 CAN 控制器的信息处理时间 IPT，该参数取决于实际情况而有所不同，通常在 0~2 t_q 范围内。

同步跳转宽度可选择以下 3 个值中的最小值：4、Phase1、Phase2。

确定了以上配置参数后，即可根据下面的公式计算振荡器容差范围：

$$(1 - df) \times fnom \leq fosc \leq (1 + df) \times fnom$$

其中：

- df = 振荡器频率的最大容差
- fosc = 振荡器实际频率
- fnom = 振荡器额定频率

最大频率容限还必须满足以下要求：

$$df \le \frac{(Phase\_seg1, Phase\_seg2)\ \min}{2 \times (13 \times tbit - Phase\_Seg2)}$$

$$df\ \max = 2 \times df \times fnom$$

其中：

- Phase1 和 Phase2 取自第 877 页的表 17-4
- tbit = 位时间
- dfmax = 两个振荡器之间的最大差异

假如可选的配置参数不止一组，那么应当优先选用振荡器容差范围最大的一组参数。

如果多个系统时钟频率不同的 CAN 节点需要达到相同的位速率，需要对其各自进行不同的配置。在计算传输时间时，应在整个 CAN 网络内选取距离最远的节点计算最大延迟时间。

整个 CAN 系统的振荡器容差范围将由系统中容差范围最小的节点所决定（木桶原理）。

以上计算结果表明：要想得到与协议兼容的位定时参数配置，有时必须在诸多因素中进行取舍；要么必须减小总线长度或位速率，要么必须提高振荡器频率的稳定性。

17.3.16.1 高波特率下的位定时参数计算

在此示例中，假设 CAN 模块输入时钟为 25MHz、CAN 总线位速率为 1Mbps。

```
位时间 = 1μs = n * tq = 5 * tq
tq = 200ns
tq = (波特率预分频系数)/CAN 模块输入时钟
波特率预分频系数 = tq * CAN 模块输入时钟
波特率预分频系数 = 200E-9 * 25E6 = 5

tSync = 1 * tq = 200ns    //固定为 1 个时间份额

总线驱动延时 50ns
接收电路延时 30ns
总线传输线(40m)延时 220ns
tProp 400ns = 2 * tq      //300ns 是 1.5 个 tq，因此进位取整为 2tq

位时间 = tSync + tTseg1 + tTSeg2 = 5 * tq
位时间 = tSync + tProp + tPhase1 + tPhase2
tPhase1 + tPhase2 = 位时间 - tSync - tProp
tPhase1 + tPhase2 = (5 * tq) - (1 * tq) - (2 * tq)
tPhase1 + tPhase2 = 2 * tq
tPhase1 = 1 * tq
tPhase2 = 1 * tq          //平均分配，因此 tPhase2 = tPhase1
```

```
tTseg1 = tProp + tPhase1
tTseg1 = (2 * tq) + (1 * tq)
tTseg1 = 3 * tq

tTseg2 = tPhase2
tTseg2 = (IPT + 1) * tq
tTseg2 = 1 * tq          //假定 IPT = 0

tSJW = 1 * tq            //取值为 min(4, tPhase1, tPhase2)
```

在上面的例子中，**CANBIT** 寄存器各位域的值为：

TSEG2	=TSeg2 – 1 =1 – 1 =0
TSEG1	=TSeg1 – 1 =3 – 1 =2
SJW	=SJW – 1 =1 – 1 =0
BRP	=波特率预分频系数 – 1 =5 – 1 =4

因此最终写入 **CANBIT** 寄存器的值是 0x0204。

17.3.16.2 低波特率下的位定时参数计算

在此计算中，假定 CAN 模块输入时钟为 50MHz，CAN 总线位速率为 100kbps。

```
位时间 = 10μs = n * tq = 10 * tq
tq = 1μs
tq = (波特率预分频系数)/CAN 模块输入时钟
波特率预分频系数 = tq * CAN 模块输入时钟
波特率预分频系数 = 1E-6 * 50E6 = 50

tSync = 1 * tq = 1μs     //固定为 1 个时间份额

总线驱动延时 200ns
接收电路延时 80ns
总线传输线(40m)延时 220ns
tProp 1μs = 1 * tq       //520ns 是 0.52 个 tq，因此进位取整为 1tq

位时间 = tSync + tTseg1 + tTSeg2 = 10 * tq
位时间 = tSync + tProp + tPhase1 + tPhase2
tPhase1 + tPhase2 = 位时间 – tSync – tProp
tPhase1 + tPhase2 = (10 * tq) – (1 * tq) – (1 * tq)
```

```
tPhase1 + tPhase2 = 8 * tq
tPhase1 = 4 * tq
tPhase2 = 4 * tq          //平均分配，因此 tPhase2 = tPhase1

tTseg1 = tProp + tPhase1
tTseg1 = (1 * tq) + (4 * tq)
tTseg1 = 5 * tq
tTseg2 = tPhase2
tTseg2 = (IPT + 4) * tq
tTseg2 = 4 * tq           //假定 IPT = 0

tSJW = 4 * tq             //取值为 min(4, tPhase1, tPhase2)
```

在上面的例子中，**CANBIT** 寄存器各位域的值为：

TSEG2	=TSeg2 – 1 =4 – 1 =3
TSEG1	=TSeg1 – 1 =5 – 1 =4
SJW	=SJW – 1 =4 – 1 =3
BRP	=波特率预分频系数 – 1 =50 – 1 =49

因此最终写入 **CANBIT** 寄存器的值是 0x34F1。

17.4 寄存器映射

第 881 页的表 17-6 列出了 CAN 寄存器。表中偏移量一列是指相对于 CAN 基地址的十六进制地址增量，各个 CAN 模块的基地址分别为：

- CAN0：0x4004 0000
- CAN1：0x4004 1000

在操作 CAN 寄存器之前，注意应先使能 CAN 模块时钟，参见第 273 页。

表 17-6. CAN 寄存器映射

偏移量	寄存器名称	类型	复位值	寄存器描述	参见页码
0x000	CANCTL	R/W	0x0000 0001	CAN 控制寄存器	884
0x004	CANSTS	R/W	0x0000 0000	CAN 状态寄存器	886
0x008	CANERR	RO	0x0000 0000	CAN 错误计数寄存器	889
0x00C	CANBIT	R/W	0x0000 2301	CAN 位定时寄存器	890
0x010	CANINT	RO	0x0000 0000	CAN 中断寄存器	892

偏移量	寄存器名称	类型	复位值	寄存器描述	参见页码
0x014	CANTST	R/W	0x0000 0000	CAN 测试寄存器	893
0x018	CANBRPE	R/W	0x0000 0000	CAN 波特率预分频系数扩展寄存器	895
0x020	CANIF1CRQ	R/W	0x0000 0001	CAN IF1 指令请求寄存器	896
0x024	CANIF1CMSK	R/W	0x0000 0000	CAN IF1 指令掩码寄存器	898
0x028	CANIF1MSK1	R/W	0x0000 FFFF	CAN IF1 掩码寄存器 1	901
0x02C	CANIF1MSK2	R/W	0x0000 FFFF	CAN IF1 掩码寄存器 2	902
0x030	CANIF1ARB1	R/W	0x0000 0000	CAN IF1 仲裁寄存器 1	904
0x034	CANIF1ARB2	R/W	0x0000 0000	CAN IF1 仲裁寄存器 2	905
0x038	CANIF1MCTL	R/W	0x0000 0000	CAN IF1 报文控制寄存器	907
0x03C	CANIF1DA1	R/W	0x0000 0000	CAN IF1 数据寄存器 A1	910
0x040	CANIF1DA2	R/W	0x0000 0000	CAN IF1 数据寄存器 A2	910
0x044	CANIF1DB1	R/W	0x0000 0000	CAN IF1 数据寄存器 B1	910
0x048	CANIF1DB2	R/W	0x0000 0000	CAN IF1 数据寄存器 B2	910
0x080	CANIF2CRQ	R/W	0x0000 0001	CAN IF2 指令请求寄存器	896
0x084	CANIF2CMSK	R/W	0x0000 0000	CAN IF2 指令掩码寄存器	898
0x088	CANIF2MSK1	R/W	0x0000 FFFF	CAN IF2 掩码寄存器 1	901
0x08C	CANIF2MSK2	R/W	0x0000 FFFF	CAN IF2 掩码寄存器 2	902
0x090	CANIF2ARB1	R/W	0x0000 0000	CAN IF2 仲裁寄存器 1	904
0x094	CANIF2ARB2	R/W	0x0000 0000	CAN IF2 仲裁寄存器 2	905
0x098	CANIF2MCTL	R/W	0x0000 0000	CAN IF2 报文控制寄存器	907
0x09C	CANIF2DA1	R/W	0x0000 0000	CAN IF2 数据寄存器 A1	910
0x0A0	CANIF2DA2	R/W	0x0000 0000	CAN IF2 数据寄存器 A2	910
0x0A4	CANIF2DB1	R/W	0x0000 0000	CAN IF2 数据寄存器 B1	910
0x0A8	CANIF2DB2	R/W	0x0000 0000	CAN IF2 数据寄存器 B2	910
0x100	CANTXRQ1	RO	0x0000 0000	CAN 发送请求寄存器 1	911
0x104	CANTXRQ2	RO	0x0000 0000	CAN 发送请求寄存器 2	911
0x120	CANNWDA1	RO	0x0000 0000	CAN 新数据寄存器 1	912
0x124	CANNWDA2	RO	0x0000 0000	CAN 新数据寄存器 2	912
0x140	CANMSG1INT	RO	0x0000 0000	CAN 报文 1 中断挂起寄存器	913
0x144	CANMSG2INT	RO	0x0000 0000	CAN 报文 2 中断挂起寄存器	913
0x160	CANMSG1VAL	RO	0x0000 0000	CAN 报文 1 有效寄存器	914
0x164	CANMSG2VAL	RO	0x0000 0000	CAN 报文 2 有效寄存器	914

17.5 寄存器描述

本章的剩余部分按照地址偏移量由小到大的顺序依次详细介绍各寄存器。CAN 模块提供两组接口寄存器用于访问报文 RAM 中的报文对象：**CANIF1x** 和 **CANIF2x**。这两组寄存器的功能完全相同，可用于实现连续会话。

寄存器 1：CAN 控制寄存器（CANCTL），偏移量 0x000

此控制寄存器用于启动 CAN 模块、使能测试模式以及使能中断。

即使置位或清零 INIT 位，也无法缩短离线恢复序列（见 CAN 2.0 规范）。器件一旦进入离线状态便会将 INIT 位置位，并终止所有总线活动。当 CPU 将 INIT 位清零后，器件需等待总线出现 129 个空闲态后（129 * 连续 11 个隐形位）才能恢复正常的总线操作。离线恢复序列结束时，错误管理计数器都会复位。

在 INIT 清零后的等待期间，每当监视到 11 个连续的高位后，都会向 **CANSTS** 寄存器写入一个第 0 位错误码（LEC 位域 = 0x5），这样 CPU 随时可以检查总线是否被拉低或受到持续干扰，同时还能监视离线恢复序列的进度。

CAN 控制寄存器（CANCTL）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x000
类型 R/W，复位值 0x0000 0001

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
域	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
域	保留								TEST	CCE	DAR	保留	EIE	SIE	IE	INIT
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位/位域	名称	类型	复位值	描述
31:8	保留	RO	0x0000 00	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
7	TEST	R/W	0	测试模式使能 取值 描述 0 CAN 控制器工作在正常模式 1 CAN 控制器工作在测试模式
6	CCE	R/W	0	配置修改使能 取值 描述 0 禁止对 **CANBIT** 寄存器写操作 1 若 INIT = 1，则允许对 **CANBIT** 寄存器写操作
5	DAR	R/W	0	禁用自动重发 取值 描述 0 允许自动重发发送失败的报文 1 禁用自动重发，报文均只尝试发送一次

位/位域	名称	类型	复位值	描述
4	保留	RO	0	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
3	EIE	R/W	0	错误中断使能 取值 描述 0 不产生错误中断 1 每当 **CANSTS** 寄存器的 BOFF 或 EWARN 标志位发生变化时，将产生一个中断
2	SIE	R/W	0	状态中断使能 取值 描述 0 不产生状态中断 1 每当正确收发一条报文或检测到 CAN 总线错误时，将产生一个中断。**CANSTS** 寄存器的 TXOK、RXOK 或 LEC 标志位发生变化，将产生一个中断。
1	IE	R/W	0	CAN 中断使能 取值 描述 0 禁用中断 1 使能中断
0	INIT	R/W	1	初始化 取值 描述 0 正常工作 1 开始初始化

寄存器 2：CAN 状态寄存器（CANSTS），偏移量 0x004

重要提示：读此寄存器时务必小心。读操作将改变寄存器内容。

状态寄存器中包含中断服务所需的信息，例如离线状况、错误计数门限以及错误类型等等。

LEC 位域保存代码能够表明 CAN 总线上一次错误的类型。当成功完成一个报文的传输（接收或发送）后此位域将自动清除。CPU 可以向此位域写入未用的错误码 0x7，将来此位域发生变化时可以很容易发现。

若 **CAN 控制寄存器（CANCTL）**中的 BOFF 或 EWARN 位置位，即可产生错误中断；若 TXOK、RXOK 或 LEC 位置位，即可产生状态中断。EPASS 位发生变化或对 RXOK、TXOK、LEC 位的写操作都不会产生中断。

假如有中断挂起，读取 **CAN 状态寄存器（CANSTS）**还会清除 **CAN 中断寄存器（CANINT）**。

CAN 状态寄存器（CANSTS）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x004
类型 R/W，复位值 0x0000 0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								BOFF	EWARN	EPASS	RXOK	TXOK		LEC	
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/位域	名称	类型	复位值	描述
31:8	保留	RO	0x0000 00	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
7	BOFF	RO	0	离线状态 取值 描述 0 CAN 控制器未处于离线状态 1 CAN 控制器处于离线状态
6	EWARN	RO	0	警告状态 取值 描述 0 两组错误计数器都低于错误警告门限（96 次） 1 至少有一组错误计数器已达到错误警告门限（96 次）

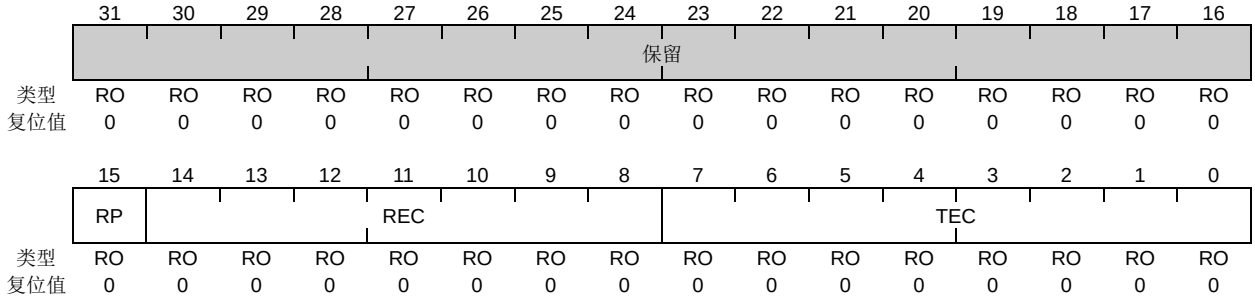
位/位域	名称	类型	复位值	描述
5	EPASS	RO	0	被动错误 取值 描述 0 CAN 模块处于主动错误状态，也就是说，接收错误计数或发送错误计数都小于等于 127 1 CAN 模块处于被动错误状态，也就是说，接收错误计数或发送错误计数至少有一个大于 127
4	RXOK	R/W	0	接收报文成功 取值 描述 0 从本标志位上一次清零以来，并未成功接收到新的报文 1 从本标志位上一次清零以来，至少成功接收到一条新的报文。此标志位与验收滤波无关 本标志位必须手动写 0 予以清除。
3	TXOK	R/W	0	发送报文成功 取值 描述 0 从本标志位上一次清零以来，并未成功发送新的报文 1 从本标志位上一次清零以来，至少成功发送了一条新的报文，没有错误并且至少有一个节点确认 本标志位必须手动写 0 予以清除。

位/位域	名称	类型	复位值	描述
2:0	LEC	R/W	0	上一错误码 此位域包含 CAN 总线上次发生的错误代码 取值 描述 0x0 无错误产生 0x1 填充错误 接收的报文中某个部分包含连续 5 个以上的相同位，而这在 CAN 协议中是不允许的 0x2 格式错误 接收的报文中某个本该具有固定格式的部分内容错误 0x3 确认错误 发送的报文未收到任何节点的确认 0x4 第 1 位错误 当发送报文时，CAN 控制器监控数据线是否存在总线冲突。在发送仲裁域期间，总线冲突是仲裁协议的一部分；在发送其它帧域器件，总线冲突即视为错误。 第 1 位错误表示器件企图发送高电平（逻辑 1），但是监视到的总线电平为低电平（逻辑 0） 0x5 第 0 位错误 第 0 位错误表示器件企图发送低电平（逻辑 0），但是监视到的总线电平为高电平（逻辑 1）。 在离线恢复期间，此状态将在每次监控到 11 个连续高电平时产生。通过检查此状态，软件就能监控离线恢复序列的运行情况，同时不对总线产生任何影响。 0x6 CRC 错误 接收到报文中的 CRC 校验和有误，表明收到的 CRC 校验和值与接收数据的 CRC 校验和值不符。 0x7 无事件 当 LEC 位域是这个值时，表明自上一次对 LEC 位域写 0x7 以来，未产生任何 CAN 总线事件。

寄存器 3：CAN 错误计数寄存器（CANERR），偏移量 0x008

本寄存器包含错误计数值，可用于分析错误的成因。

CAN 错误计数寄存器（CANERR）
CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x008
类型 RO，复位值 0x0000 0000

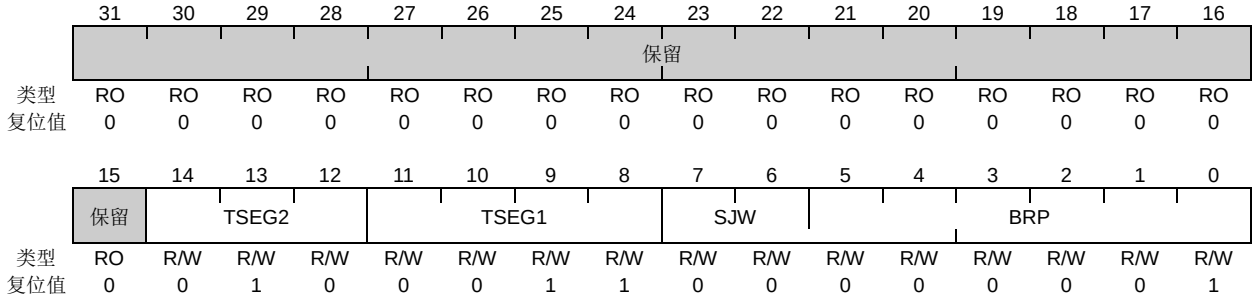


位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15	RP	RO	0	接收被动错误 取值 描述 0 接收错误计数低于被动错误门限（127 或更低） 1 接收错误计数已达到被动错误门限（128 或更高）
14:8	REC	RO	0	接收错误计数 此位域包含接收错误计数器的值（0～127）
7:0	TEC	RO	0	发送错误计数 此位域包含发送错误计数器的值（0～255）

寄存器 4：CAN 位定时寄存器（CANBIT），偏移量 0x00C

本寄存器用于定义位时间以及时间份额。写入的数值应按照系统时钟频率进行计算。只有将 **CANCTL** 寄存器的 CCE 位和 INIT 位置位时，此寄存器才允许写入。关于位定时配置的详细信息，请参阅第 876 页的“位时间及位速率”一节。

CAN 位定时寄存器（CANBIT）
CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x00C
类型 R/W，复位值 0x0000 2301



位/位域	名称	类型	复位值	描述
31:15	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
14:12	TSEG2	R/W	0x2	采样点后的时间段 有效范围 0x00～0x07。硬件实际采用的时间份额数是此位域值再加 1。 例如，复位值 0x2 代表定义 Phase2 的长度为 3（2+1）个位时间份额（参见第 877 页的图 17-4）。位时间份额由 BRP 位域确定。
11:8	TSEG1	R/W	0x3	采样点前的时间段 有效范围 0x00～0x0F。硬件实际采用的时间份额数是此位域值再加 1。 例如，复位值 0x3 代表定义给 Phase1 的长度为（3+1）个位时间份额（参见第 877 页的图 17-4）。位时间份额由 BRP 位域确定。
7:6	SJW	R/W	0x0	（再）同步跳转宽度 有效范围 0x00～0x03。硬件实际采用的时间份额数是此位域值再加 1。 在帧起始（SOF）期间，假如 CAN 控制器检测到存在相位误差，可以按 SJW 位域调整 TSEG2 或 TSEG1 的长度来补偿。复位值 0 表示允许调整 1 个时间份额。

位/位域	名称	类型	复位值	描述
5:0	BRP	R/W	0x1	波特率预分频系数 振荡器频率将按照此系数分频，产生位时间份额。位时间是此时间份额的整数倍。 有效范围 0x00～0x3F。硬件实际采用的数值是此位域值再加 1。 BRP 位域定义每个时间份额由多少个系统时钟周期组成，因此复位值表示 2（1+1）个系统时钟周期组成 1 个时间份额。 **CANBRPE** 寄存器可对位时间进行进一步细分。

寄存器 5：CAN 中断寄存器（CANINT），偏移量 0x010

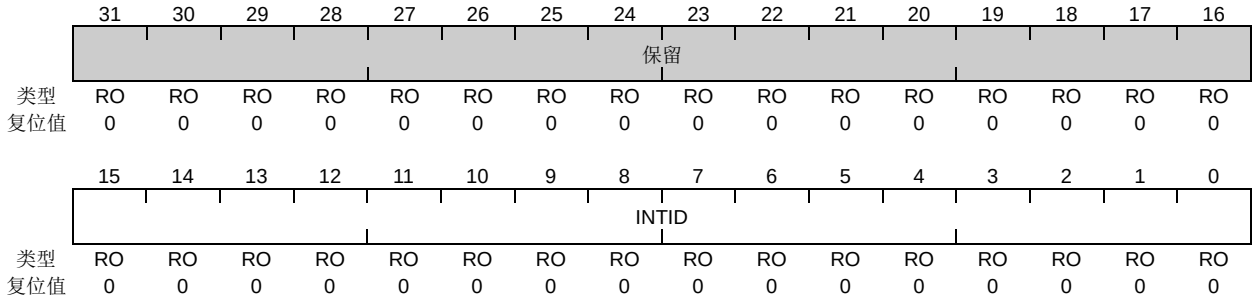
本寄存器包含的内容是中断源。

假如多个中断同时挂起，则 **CAN 中断寄存器（CANINT）**始终指向挂起的最高优先级中断，而并非按照中断产生的顺序。中断在被 CPU 清除之前将保持挂起。假如 INTID 位域的值不是 0x0000(默认值)，并且 **CANCTL** 寄存器的 IE 位置位，则该中断就是激活的。在 INTID 位域被清除（通过读取 **CANSTS** 寄存器）之前，或在 **CANCTL** 寄存器的 IE 位清零之前，中断线将始终保持激活。

注：假如 **CAN 中断寄存器（CANINT）**中包含挂起的中断，那么读取 **CAN 状态寄存器（CANSTS）**会将其清除。

CAN 中断寄存器（CANINT）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x010
类型 RO，复位值 0x0000 0000



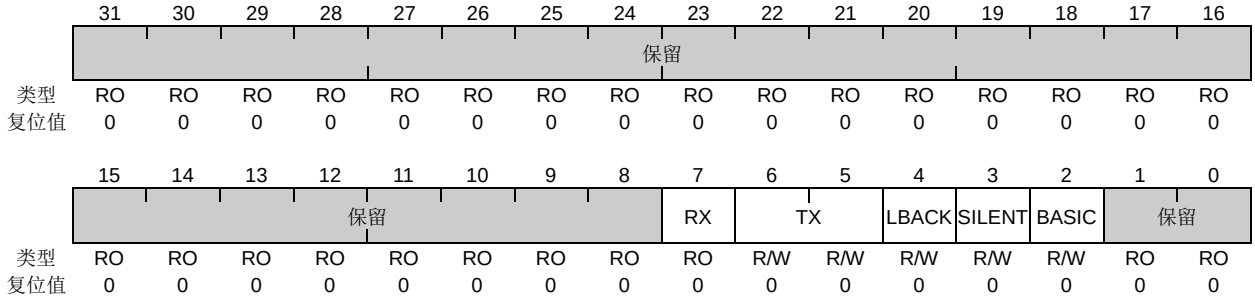
位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15:0	INTID	RO	0x0000	中断标识 此位域中的数值代表中断源。

取值	描述
0x0000	无中断挂起
0x0001~0x0020	此编号对应的报文对象产生中断
0x0021~0x7FFF	保留
0x8000	状态中断
0x8001~0xFFFF	保留

寄存器 6：CAN 测试寄存器（CANTST），偏移量 0x014

本寄存器用于自检以及访问外部管脚。只有当 **CANCTL** 寄存器的 TEST 位置位后，才允许对本寄存器进行写操作。用户可以按需组合不同的测试模式，不过应当注意到：假如 TX 位域的值不是 0x0，将可能影响 CAN 发送功能。

CAN 测试寄存器（CANTST）
CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x014
类型 R/W，复位值 0x0000 0000



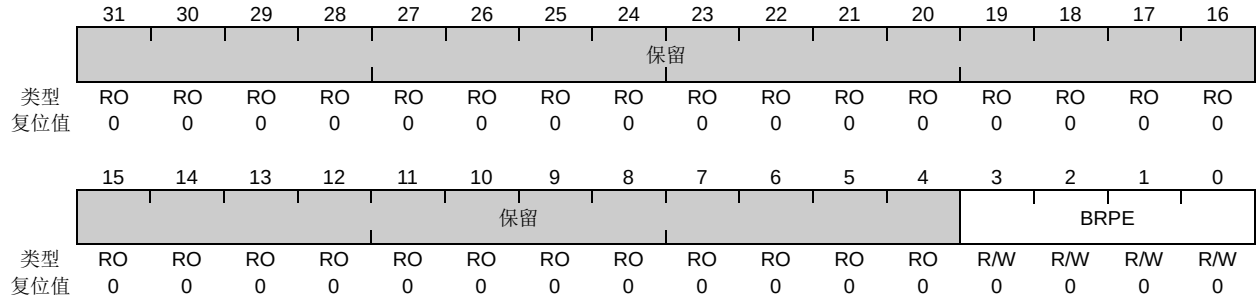
位/位域	名称	类型	复位值	描述
31:8	保留	RO	0x0000 00	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
7	RX	RO	0	接收管脚状态 取值 描述 0 CANnRx 脚为低电平 1 CANnRx 脚为高电平
6:5	TX	R/W	0x0	发送控制 控制 CANnTx 脚。 取值 描述 0x0 CAN 模块控制 CANnTx 脚由 CAN 模块控制。默认设置。 0x1 采样点 采样点在 CANnTx 脚驱动输出。这种模式适用于监控位时间。 0x2 拉低 CANnTx 脚始终驱动输出低电平。这种模式主要用于验证 CAN 总线物理层的好坏。 0x3 拉高 CANnTx 脚始终驱动输出高电平。这种模式主要用于验证 CAN 总线物理层的好坏。

位/位域	名称	类型	复位值	描述
4	LBACK	R/W	0	环回模式 取值 描述 0 禁用环回模式 1 使能环回模式。在环回模式下，发送器所发送的数据将直接环回到接收器。而接收管脚所接收的输入数据将被忽略。
3	SILENT	R/W	0	安静模式 取值 描述 0 禁用安静模式 1 使能安静模式。在安静模式下，CAN 控制器不发送任何数据，只监控总线。此模式通常也称为总线监控模式。
2	BASIC	R/W	0	基本模式 取值 描述 0 禁用基本模式 1 使能基本模式。在基本模式下，软件应当采用 **CANIF1** 寄存器组作为发送缓冲，采用 **CANIF2** 寄存器组作为接收缓冲。
1:0	保留	RO	0x0	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。

寄存器 7：CAN 波特率预分频系数扩展寄存器（CANBRPE），偏移量 0x018

本寄存器结合 **CANBIT** 寄存器中的 BRP 位域共同实现位时间的分频设置。当 **CANCTL** 寄存器的 CCE 位置位时，本寄存器即被使能。

CAN 波特率预分频系数扩展寄存器（CANBRPE）
CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x018
类型 R/W，复位值 0x0000 0000



位/位域	名称	类型	复位值	描述
31:4	保留	RO	0x0000 000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
3:0	BRPE	R/W	0x0	波特率预分频系数扩展 0x0~0xF：这 4 个位结合 **CANBIT** 寄存器的 BRP 位域，可将其取值范围扩展到最大 1023。硬件实际采用的波特率预分频系数是 BRPE（高 4 位）和 BRP（低 6 位）的值再加 1。

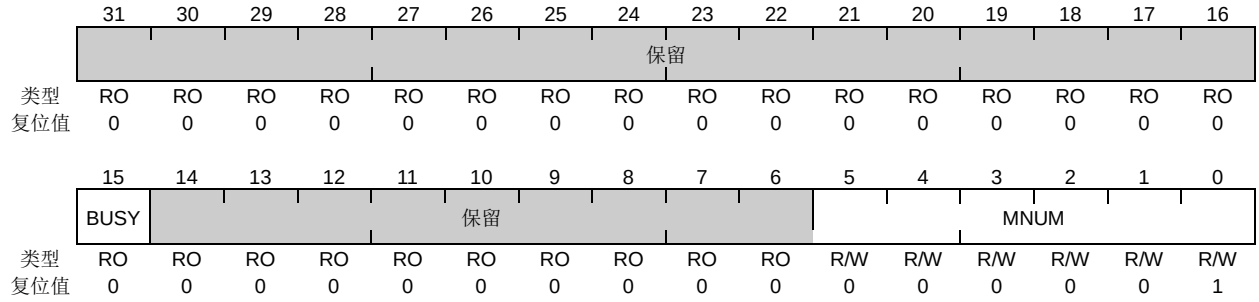
寄存器 8：CAN IF1 指令请求寄存器（CANIF1CRQ），偏移量 0x020

寄存器 9：CAN IF2 指令请求寄存器（CANIF2CRQ），偏移量 0x080

当对本寄存器的MNUM位域写入某个报文对象的序号、并且**CANIF1MCTL**寄存器的TXRQST位置位时，即会启动报文传输。写 MNUM 位域的同时 BUSY 位也会自动置位，表明在 CAN 接口寄存器与内部报文 RAM 之间正在进行数据交换。等待 3 到 6 个 CAN_CLK 周期后，接口寄存器与报文 RAM 之间的传输结束，随后将 BUSY 位清零。

CAN IF1 指令请求寄存器（CANIF1CRQ）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x020
类型 R/W，复位值 0x0000 0001



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15	BUSY	RO	0	忙标志 取值 描述 0 当读/写活动结束后，此标志位自动清零 1 当向本寄存器中写入报文对象序号时，此标志位即会自动置位
14:6	保留	RO	0x00	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。

位/位域	名称	类型	复位值	描述
5:0	MNUM	R/W	0x01	报文序号 从报文 RAM 的 32 个报文对象中选择一个进行数据传输。报文对象的序号为 1 到 32 之间的数字。 取值 描述 0x00 保留 0x01~0x20 报文序号 代表 1 到 32 号报文对象 0x21~0x3F 保留 无效的报文序号，此时本位域的值将自动平移为 0x01~0x1F 范围内

寄存器 10：CAN IF1 指令掩码寄存器（CANIF1CMSK），偏移量 0x024

寄存器 11：CAN IF2 指令掩码寄存器（CANIF2CMSK），偏移量 0x084

读取指令掩码寄存器，可获取各种功能的当前状态。写入指令掩码寄存器，可以选择传输方向、选择缓冲寄存器组作为数据传输的源或目的。

请注意，若 WRNRD 位清 0 并且 CLRINTPND 及/或 NEWDAT 位置 1，则在读取报文对象缓冲时，这些报文对象缓冲中的中断等待标志及/或新数据标志将被清除。

CAN IF1 指令掩码寄存器（CANIF1CMSK）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x024
类型 R/W，复位值 0x0000 0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	保留								WRNRD	MASK	ARB	CONTROL	CLRINTPND	NEWDAT / TXRQST	DATAA	DATAB
类型	RO	RO	RO	RO	RO	RO	RO	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/位域	名称	类型	复位值	描述
31:8	保留	RO	0x0000 00	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
7	WRNRD	R/W	0	写，非读 取值 描述 0 将 **CANIFnCRQ** 寄存器 MNUM 位域所指定的 CAN 报文对象中的数据移动到 CANIFn 寄存器组中。 1 将 CANIFn 寄存器组中的数据移动到 **CANIFnCRQ** 寄存器 MNUM 位所指定的 CAN 报文对象中。 **注：**当 CLRINTPND 及/或 NEWDAT 位置 1 时，可通过从报文缓冲的读操作（WRNRD = 0）来清除中断挂起状况以及新数据状况。
6	MASK	R/W	0	访问掩码位 取值 描述 0 掩码位无变化 1 将报文对象的 IDMASK + DIR + MXTD 传递给接口寄存器

位/位域	名称	类型	复位值	描述
5	ARB	R/W	0	访问仲裁位 取值 描述 0 仲裁位无变化 1 将报文对象的 ID + DIR + XTD + MSGVAL 传递给接口寄存器
4	CONTROL	R/W	0	访问控制位 取值 描述 0 控制位无变化 1 将 **CANIFnMCTL** 寄存器的控制位传递给接口寄存器
3	CLRINTPND	R/W	0	清除中断挂起位 此标志位的功能取决于 WRNRD 位的设置。 取值 描述 0 若 WRNRD = 0，则中断挂起状态从报文缓冲传递给 **CANIFnMCTL** 寄存器； 若 WRNRD = 1，报文对象中的 INTPND 位保持不变。 1 若 WRNRD = 0，则清除报文缓冲中的中断挂起状态。请注意传递给 **CANIFnMCTL** 寄存器的这个标志位总是反映其被清除之前的真实状态； 若 WRNRD = 1，报文对象中的 INTPND 位将被清零。
2	NEWDAT / TXRQST	R/W	0	NEWDAT / TXRQST 位 此标志位的功能取决于 WRNRD 位的设置。 取值 描述 0 若 WRNRD = 0，则新数据状态位将从从报文缓冲传递给 **CANIFnMCTL** 寄存器； 若 WRNRD = 1，将不会请求发送。 1 若 WRNRD = 0，则清除报文缓冲中的新数据状态。请注意传递给 **CANIFnMCTL** 寄存器的这个标志位总是反映其被清除之前的真实状态； 若 WRNRD = 1，将请求发送。请注意当本标志位置位时，**CANIFnMCTL** 寄存器中的 TXRQST 位将被忽略。

位/位域	名称	类型	复位值	描述
1	DATAA	R/W	0	访问数据字节 0 到 3 此标志位的功能取决于 WRNRD 位的设置。 取值 描述 0 数据字节 0~3 不会改变 1 若 WRNRD = 0，则将 **CANIFnDA1** 寄存器以及 **CANIFnDA2** 寄存器的数据字节 0~3 传递给报文对象； 若 WRNRD = 1，则将报文对象中的数据字节 0~3 传递给 **CANIFnDA1** 寄存器和 **CANIFnDA2** 寄存器。
0	DATAB	R/W	0	访问数据字节 4 到 7 此标志位的功能取决于 WRNRD 位的设置。 取值 描述 0 数据字节 4~7 不会改变 1 若 WRNRD = 0，则将 **CANIFnDB1** 寄存器以及 **CANIFnDB2** 寄存器的数据字节 4~7 传递给报文对象； 若 WRNRD = 1，则将报文对象中的数据字节 4~7 传递给 **CANIFnDB1** 寄存器和 **CANIFnDB2** 寄存器。

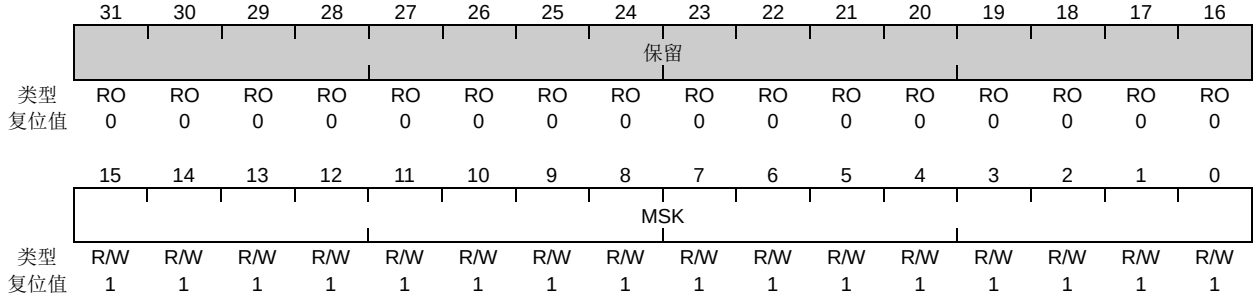
寄存器 12：CAN IF1 掩码寄存器 1（CANIF1MSK1），偏移量 0x028

寄存器 13：CAN IF2 掩码寄存器 1（CANIF2MSK1），偏移量 0x088

本寄存器中包含的掩码信息将连同数据信息（**CANIFnDAn**）、仲裁信息（**CANIFnARBn**）、控制信息（**CANIFnMCTL**）一起写入报文 RAM 中的报文对象。本寄存器中的掩码可结合 **CANIFnARBn** 寄存器 ID 位域实现验收过滤。其余掩码信息在 **CANIFnMSK2** 寄存器中。

CAN IF1 掩码寄存器 1（CANIF1MSK1）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x028
类型 R/W，复位值 0x0000 FFFF



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15:0	MSK	R/W	0xFFFF	标识符掩码 对于 29 位标识符，本位域对应于 ID 的[15:0]位域，**CANIFnMSK2** 寄存器的 MSK 位域对应于 ID 的[28:16]位域。对于 11 位标识符，本位域被忽略。 取值 描述 0 报文对象中的对应标识符域（ID）不用于验收过滤。 1 报文对象中的对应标识符域（ID）用于验收过滤。

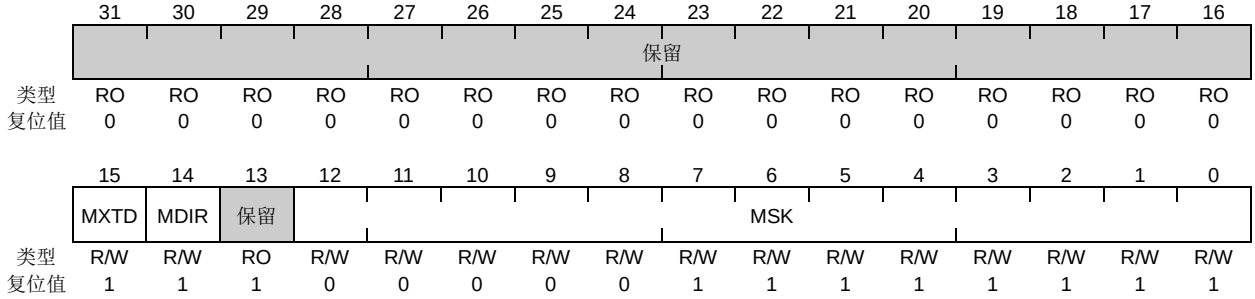
寄存器 14：CAN IF1 掩码寄存器 2（CANIF1MSK2），偏移量 0x02C

寄存器 15：CAN IF2 掩码寄存器 2（CANIF2MSK2），偏移量 0x08C

本寄存器中包含的扩展掩码信息需结合 **CANIFnMSK1** 寄存器共同使用。

CAN IF1 掩码寄存器 2（CANIF1MSK2）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x02C
类型 R/W，复位值 0x0000 E0FF



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15	MXTD	R/W	1	掩码扩展标识符 取值 描述 0 扩展标识符（**CANIFnARB2** 寄存器的 XTD 位）不影响验收过滤 1 扩展标识符位 XTD 也用于验收过滤
14	MDIR	R/W	1	掩码报文方向 取值 描述 0 报文方向位（**CANIFnARB2** 寄存器的 DIR 位）不影响验收过滤 1 报文方向位 DIR 也用于验收过滤
13	保留	RO	1	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。

位/位域	名称	类型	复位值	描述
12:0	MSK	R/W	0xFF	标识符掩码 对于 29 位标识符，此位域用于 ID 的[28:16]位域，**CANIFnMSK1** 寄存器的 MSK 位域用于 ID 的[15:0]位域。对于 11 位标识符，MSK[12:2]位域用于 ID 的[10:0]位域。 取值 描述 0 报文对象中的对应标识符域（ID）不用于验收过滤。 1 报文对象中的对应标识符域（ID）用于验收过滤。

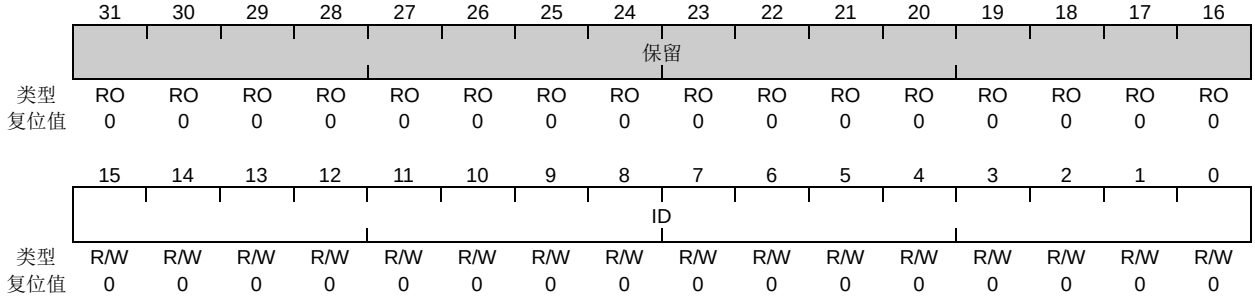
寄存器 16：CAN IF1 仲裁寄存器 1（CANIF1ARB1），偏移量 0x030

寄存器 17：CAN IF2 仲裁寄存器 1（CANIF2ARB1），偏移量 0x090

本寄存器包含用于验收过滤的报文标识符。

CAN IF1 仲裁寄存器 1（CANIF1ARB1）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x030
类型 R/W，复位值 0x0000 0000



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15:0	ID	R/W	0x0000	报文标识符 此位域结合 **CANIFnARB2** 寄存器的 ID 位域，可用于创建报文标识符。 对于 29 位报文标识符，**CANIFnARB1** 寄存器的[15:0]位域用于 ID 的[15:0]，**CANIFnARB2** 寄存器的[12:0]位域用于 ID 的[28:16]。 对于 11 位报文标识符，本位域无意义。

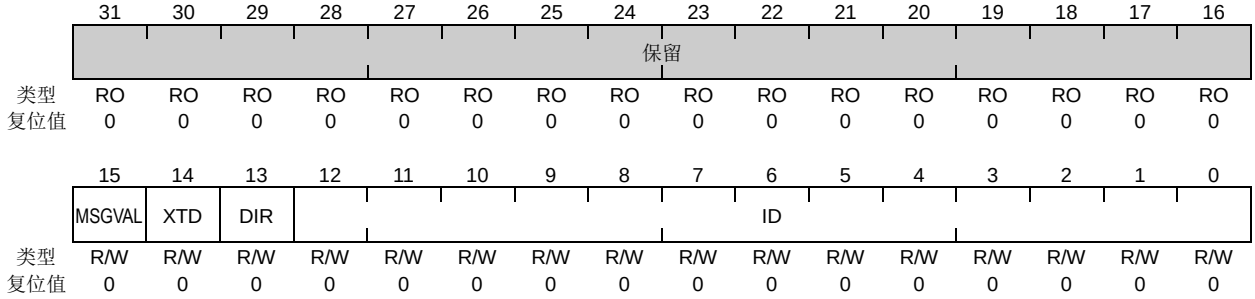
寄存器 18：CAN IF1 仲裁寄存器 2（CANIF1ARB2），偏移量 0x034

寄存器 19：CAN IF2 仲裁寄存器 2（CANIF2ARB2），偏移量 0x094

本寄存器包含用于验收过滤的相关信息。

CAN IF1 仲裁寄存器 2（CANIF1ARB2）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x034
类型 R/W，复位值 0x0000 0000



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15	MSGVAL	R/W	0	有效报文 取值 描述 0 该报文对象被报文处理器忽略 1 该报文对象已经配置完成，并准备由 CAN 控制器的报文处理器处理管理。 初始化期间，所有不用到的报文对象必须将此位清零；将 **CANCTL** 寄存器的 INIT 位清零之前也需要将此位清零。另外，当不再用到某个报文对象时，或修改以下任何一个位时，都必须清除 MSGVAL 位：**CANIFnARBn** 寄存器的 ID 位域、**CANIFnARB2** 寄存器的 XTD 位和 DIR 位、**CANIFnMCTL** 寄存器的 DLC 位域。
14	XTD	R/W	0	扩展标识符 取值 描述 0 该报文对象采用 11 位标准标识符 1 该报文对象采用 29 位扩展标识符

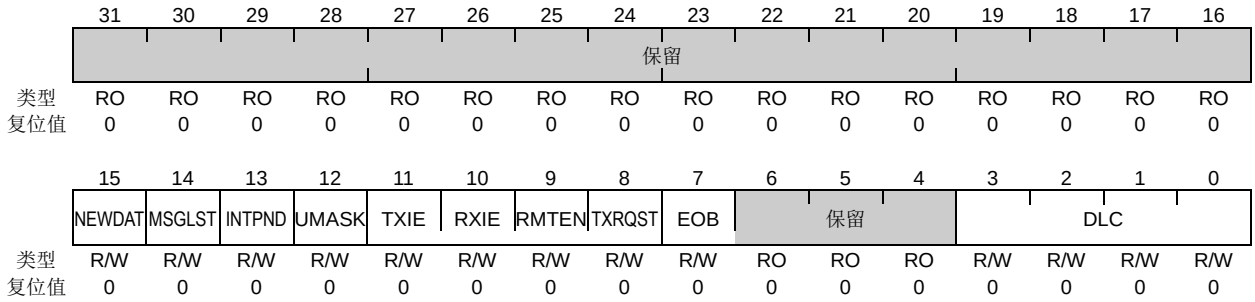
位/位域	名称	类型	复位值	描述
13	DIR	R/W	0	报文方向 取值 描述 0 接收。当 **CANIFnMCTL** 寄存器的 TXRQST 位置位时，表明接收到标识符匹配的远程帧。当接收到标识符匹配的数据帧时，该报文将存储在此报文对象中。 1 发送。当 **CANIFnMCTL** 寄存器的 TXRQST 位置位时，相应的报文对象将作为数据帧发送。当接收到标识符匹配的远程帧时，该报文对象的 TXRQST 位将置位。
12:0	ID	R/W	0x000	报文标识符 此位域结合 **CANIFnARB2** 寄存器的 ID 位域，可用于创建报文标识符。 对于 29 位报文标识符，**CANIFnARB1** 寄存器的 ID[15:0]位域用于 ID 的[15:0]，本位域用于 ID 的[28:16]。 对于 11 位报文标识符，本位域的[12:2]用于 ID 的[10:0]。**CANIFnARB1** 寄存器的 ID 位域被忽略。

寄存器 20：CAN IF1 报文控制寄存器（CANIF1MCTL），偏移量 0x038

寄存器 21：CAN IF2 报文控制寄存器（CANIF2MCTL），偏移量 0x098

本寄存器包含将要发送给报文 RAM 的报文对象的控制信息。

CAN IF1 报文控制寄存器（CANIF1MCTL）
CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x038
类型 R/W，复位值 0x0000 0000



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15	NEWDAT	R/W	0	新数据 取值 描述 0 此标志位上一次由 CPU 清零之后，报文控制器并未向此报文对象的数据部分写入新的数据。 1 报文处理器或 CPU 已经向此报文对象的数据部分写入了新的数据。
14	MSGLST	R/W	0	报文丢失 取值 描述 0 此标志位上一次由 CPU 清零之后，并未丢失新的报文。 1 NEWDAT 位置位并且报文处理器在此对象中又写入了新的报文，导致报文丢失。 仅当报文对象 **CANIFnARB2** 寄存器的 DIR 位清零（接收方向）时，此标志位才有效。
13	INTPND	R/W	0	中断挂起 取值 描述 0 此报文对象不是中断源 1 此报文对象是中断源。假如没有其它更高优先级的中断源，那么 **CANINT** 寄存器中的中断标识符将指向此报文对象。

位/位域	名称	类型	复位值	描述
12	UMASK	R/W	0	开启验收过滤 取值 描述 0 掩码位被忽略 1 使用掩码位（**CANIFnMSKn** 寄存器的 MSK、MXTD、MDIR 位）进行验收过滤
11	TXIE	R/W	0	发送中断使能 取值 描述 0 成功发送一帧后，**CANIFnMCTL** 寄存器的 INTPND 位不会变化 1 成功发送一帧后，**CANIFnMCTL** 寄存器的 INTPND 位将自动置位
10	RXIE	R/W	0	接收中断使能 取值 描述 0 成功接收一帧后，**CANIFnMCTL** 寄存器的 INTPND 位不会变化 1 成功接收一帧后，**CANIFnMCTL** 寄存器的 INTPND 位将自动置位
9	RMTEN	R/W	0	远程帧使能 取值 描述 0 收到远程帧后，**CANIFnMCTL** 寄存器的 TXRQST 位不会变化 1 收到远程帧后，**CANIFnMCTL** 寄存器的 TXRQST 位将自动置位
8	TXRQST	R/W	0	发送请求 取值 描述 0 此报文对象并未等待发送 1 已经请求发送此报文对象，并且尚未发送完成

注：若 **CANIFnCMSK** 寄存器的 WRNRD 位和 TXRQST 位置位，则此标志位被忽略。

位/位域	名称	类型	复位值	描述
7	EOB	R/W	0	缓冲结束 取值 描述 0 此报文对象属于 FIFO 缓冲区，但不是 FIFO 的最后一条报文对象 1 此报文对象是 FIFO 缓冲区的最后一个报文对象，或只是单个缓冲。 此标志位用于将两个或两个以上报文对象（最多 32 个）组成 FIFO 缓冲区。对于单个报文对象（因此并不属于 FIFO 缓冲），此标志位必须置位。
6:4	保留	RO	0x0	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
3:0	DLC	R/W	0	数据长度码 取值 描述 0x0~0x8 指定数据帧中的字节数 0x9~0xF 默认数据长度为 8 字节 报文对象的 **CANIFnMCTL** 寄存器的 DLC 位域，必须与其它节点上具有相同标识符的报文对象定义一致。当报文处理器保存数据帧时，会将收到报文的 DLC 写入本机的报文对象中。

寄存器 22：CAN IF1 数据寄存器 A1（CANIF1DA1），偏移量 0x03C

寄存器 23：CAN IF1 数据寄存器 A2（CANIF1DA2），偏移量 0x040

寄存器 24：CAN IF1 数据寄存器 B1（CANIF1DB1），偏移量 0x044

寄存器 25：CAN IF1 数据寄存器 B2（CANIF1DB2），偏移量 0x048

寄存器 26：CAN IF2 数据寄存器 A1（CANIF2DA1），偏移量 0x09C

寄存器 27：CAN IF2 数据寄存器 A2（CANIF2DA2），偏移量 0x0A0

寄存器 28：CAN IF2 数据寄存器 B1（CANIF2DB1），偏移量 0x0A4

寄存器 29：CAN IF2 数据寄存器 B2（CANIF2DB2），偏移量 0x0A8

这些寄存器包含将要发送或刚刚收到的数据。在 CAN 数据帧中，数据字节 0 是发送或接收的第一个字节，数据字节 7 是发送或接收的最后一个字节。在 CAN 的串行比特流中，每个数据字节都是高位在前的。

CAN IF1 数据寄存器 A1（CANIF1DA1）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x03C
类型 R/W，复位值 0x0000 0000

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA															
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15:0	DATA	R/W	0x0000	数据 **CANIFnDA1** 寄存器包含数据字节 1 和 0；**CANIFnDA2** 寄存器包含数据字节 3 和 2；**CANIFnDB1** 寄存器包含数据字节 5 和 4；**CANIFnDB2** 寄存器包含数据字节 7 和 6。

寄存器 30：CAN 发送请求寄存器 1（CANTXRQ1），偏移量 0x100

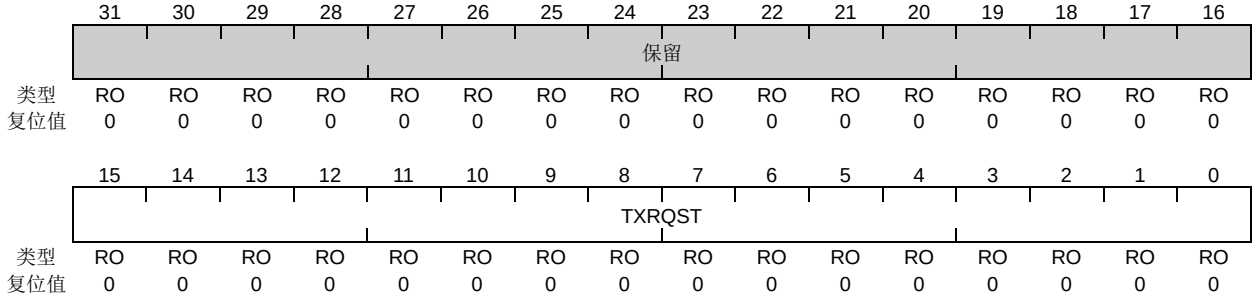
寄存器 31：CAN 发送请求寄存器 2（CANTXRQ2），偏移量 0x104

CANTXRQ1 和 **CANTXRQ2** 寄存器共同保存 32 个报文对象的 TXRQST 位状态。通过读取这些标志位，CPU 可以检查哪些报文对象有挂起的发送请求。某个报文对象的 TXRQST 位可以通过三种途径变更：(1) CPU 通过 **CANIFnMCTL** 寄存器修改；(2) 报文处理器状态机在收到远程帧后自动修改；(3) 报文处理器状态机在成功发送报文后自动修改。

CANTXRQ1 寄存器包含报文 RAM 中前 16 个报文对象的 TXRQST 位；**CANTXRQ2** 寄存器包含报文 RAM 中后 16 个报文对象的 TXRQST 位。

CAN 发送请求寄存器 1（CANTXRQ1）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x100
类型 RO，复位值 0x0000 0000



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15:0	TXRQST	RO	0x0000	发送请求位 取值 描述 0 相应的报文对象并未等待发送 1 相应的报文对象已请求发送，并且尚未发送完成

寄存器 32：CAN 新数据寄存器 1（CANNWDA1），偏移量 0x120

寄存器 33：CAN 新数据寄存器 2（CANNWDA2），偏移量 0x124

CANNWDA1 和 **CANNWDA2** 寄存器共同保存 32 个报文对象的 NEWDAT 位状态。通过读取这些标志位，CPU 可以检查哪些报文对象的数据部分有更新。某个报文对象的 NEWDAT 位可以通过三种途径变更：(1) CPU 通过 **CANIFnMCTL** 寄存器修改；(2) 报文处理器状态机在收到数据帧后自动修改；(3) 报文处理器状态机在成功发送报文后自动修改。

CANNWDA1 寄存器包含报文 RAM 中前 16 个报文对象的 NEWDAT 位；**CANNWDA2** 寄存器包含报文 RAM 中后 16 个报文对象的 NEWDAT 位。

CAN 新数据寄存器 1（CANNWDA1）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x120
类型 RO，复位值 0x0000 0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	保留															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NEWDAT															
类型	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
复位值	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15:0	NEWDAT	RO	0x0000	新数据位 取值 描述 0 自从此标志位上次被 CPU 清零以来，相应报文对象的数据部分并未写入新的数据 1 报文处理器或 CPU 向相应报文对象的数据部分写入了新的数据

寄存器 34：CAN 报文 1 中断挂起寄存器（CANMSG1INT），偏移量 0x140

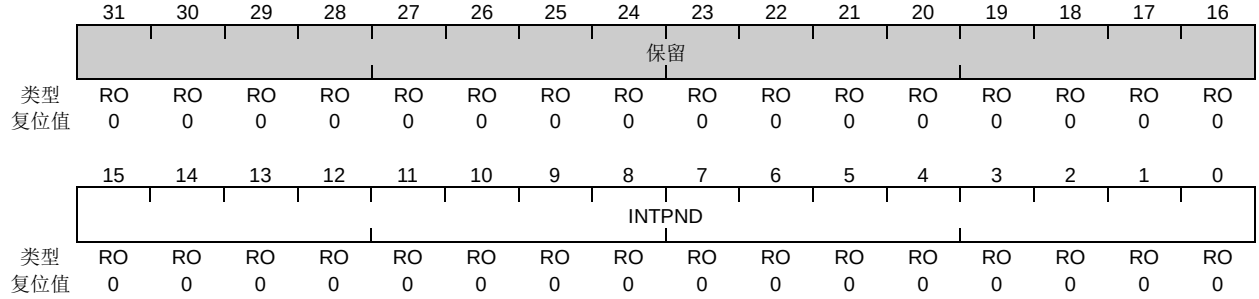
寄存器 35：CAN 报文 2 中断挂起寄存器（CANMSG2INT），偏移量 0x144

CANMSG1INT 和 **CANMSG2INT** 寄存器共同保存 32 个报文对象的 INTPND 位状态。通过读取这些标志位，CPU 可以检查哪些报文对象有挂起的中断请求。某个报文对象的 INTPND 位可以通过两种途径更改：(1) CPU 通过 **CANIFnMCTL** 寄存器修改；(2) 报文处理器状态机在成功接收或发送一帧后自动修改。

CANMSG1INT 寄存器包含报文 RAM 中前 16 个报文对象的 INTPND 位；**CANMSG2INT** 寄存器包含报文 RAM 中后 16 个报文对象的 INTPND 位。

CAN 报文 1 中断挂起寄存器（CANMSG1INT）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x140
类型 RO，复位值 0x0000 0000



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15:0	INTPND	RO	0x0000	中断挂起位 取值 描述 0 相应报文对象并非中断源 1 相应报文对象是当前挂起的中断源之一

寄存器 36：CAN 报文 1 有效寄存器（CANMSG1VAL），偏移量 0x160

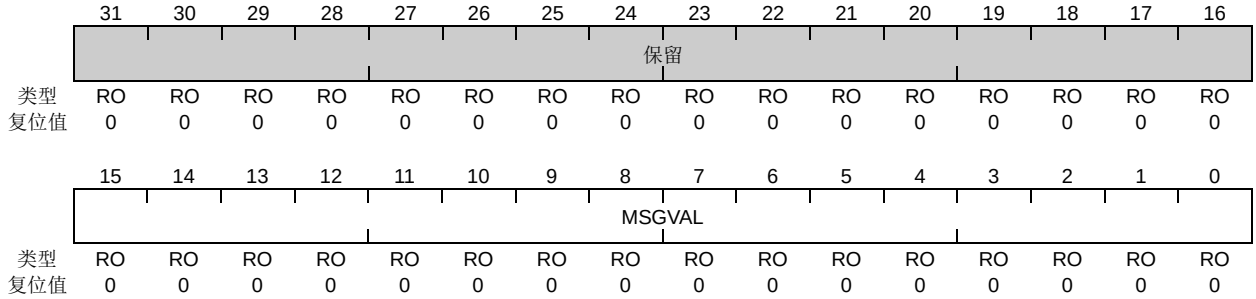
寄存器 37：CAN 报文 2 有效寄存器（CANMSG2VAL），偏移量 0x164

CANMSG1VAL 和 **CANMSG2VAL** 寄存器共同保存 32 个报文对象的 MSGVAL 位状态。通过读取这些标志位，CPU 可以检查哪些报文对象有效。某个报文对象的 MSGVAL 位可以由 CPU 修改 **CANIFnARB2** 寄存器予以变更。

CANMSG1VAL 寄存器包含报文 RAM 中前 16 个报文对象的 MSGVAL 位；**CANMSG2VAL** 寄存器包含报文 RAM 中后 16 个报文对象的 MSGVAL 位。

CAN 报文 1 有效寄存器（CANMSG1VAL）

CAN0 基地址 0x4004 0000
CAN1 基地址 0x4004 1000
偏移量 0x160
类型 RO，复位值 0x0000 0000



位/位域	名称	类型	复位值	描述
31:16	保留	RO	0x0000	软件不得依赖保留位的值。为保障软件与将来产品的兼容性，对寄存器进行“读-修改-写”操作时，不得变更保留位的值。
15:0	MSGVAL	RO	0x0000	报文有效位 取值 描述 0 相应报文对象并未配置，被报文处理器忽略 1 相应报文对象已完成配置，应当由报文处理器处理

北京锐鑫同创公司相关信息

技术支持

如果您对文档有所疑问，您可以在办公时间（星期一至星期五上午 8:30~11:50；下午 1:30~5:30）拨打技术支持电话或 E-mail 联系。

北京锐鑫同创是 TI 第三方合作伙伴，专注于 TI Stellaris M3 产品的市场推广、方案设计和技术服务，同时提供开发板、仿真器、编程器等开发工具，公司以“把握市场脉搏，专注技术创新，提供诚信服务，实现共赢发展！”为核心价值理念，为客户提供实时、高效的技术和服务。

电话：010-82418301
传真：010-82418302
Email：support@realsense.com.cn
网站：www.realsense.com.cn
技术论坛: www.hellom3.cn