注意 : この日本語版文書は参考資料としてご利用ください。最新情報は必ずオリジナルの英語版をご参照願います。

セクション 21. 拡張コントローラ エリア ネットワーク (ECAN™)

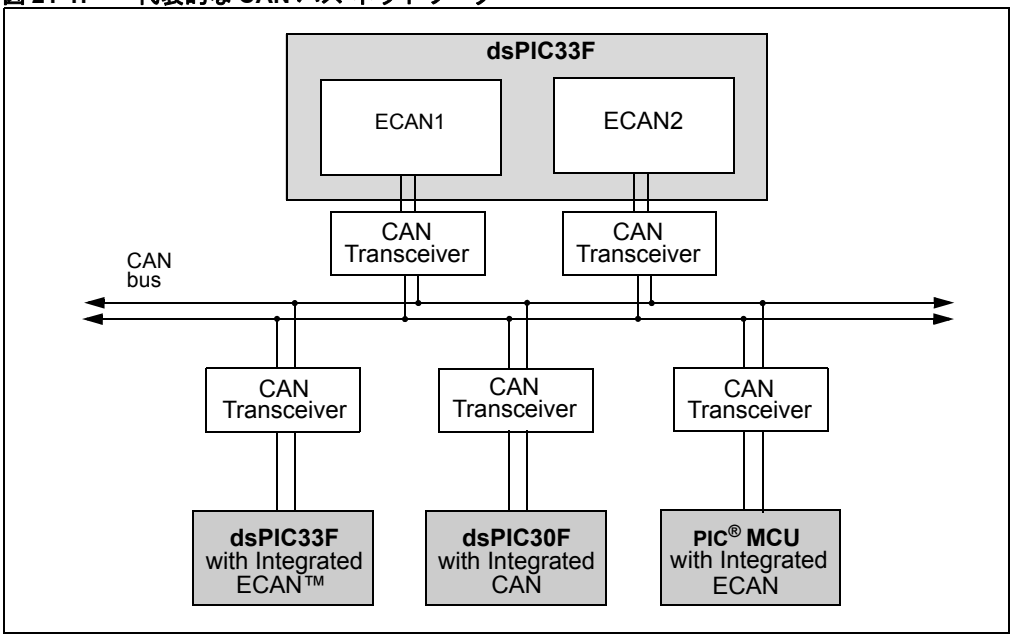
ハイライト

本セクションには以下の主要項目を記載しています。

21.1 はじめに

dsPIC33F 拡張コントローラ エリア ネットワーク (ECAN™) モジュールは、主に工業用と車載アプリケーションで使用される CAN 2.0B プロトコルを実装しています。この非同期シリアル データ通信プロトコルは、電気的ノイズが多い環境において信頼性の高い通信を実現します。dsPIC33F デバイスファミリは最大 2 つの ECAN モジュールを内蔵します。図 21-1 に代表的な CAN バス トポロジを示します。

図 21-1: 代表的な CAN バス ネットワーク



ECAN モジュールの主な機能は以下の通りです。

規格への対応 :

- CAN 2.0B に完全準拠
- 最大 1 Mbps までのビットレートをプログラム可能

メッセージ受信 :

- 32 個のメッセージ バッファ – 全てを受信用に利用可能
- メッセージ フィルタ処理用の 16 のアクセプタンス フィルタ
- メッセージ フィルタ処理用の 3 つのアクセプタンス フィルタ マスク レジスタ
- リモート送信要求に対する自動応答
- 最大 32 段のメッセージ FIFO (First In First Out) バッファ
- DeviceNet™ アドレッシングをサポート
- メッセージ受信用の DMA インターフェイス

メッセージ送信 :

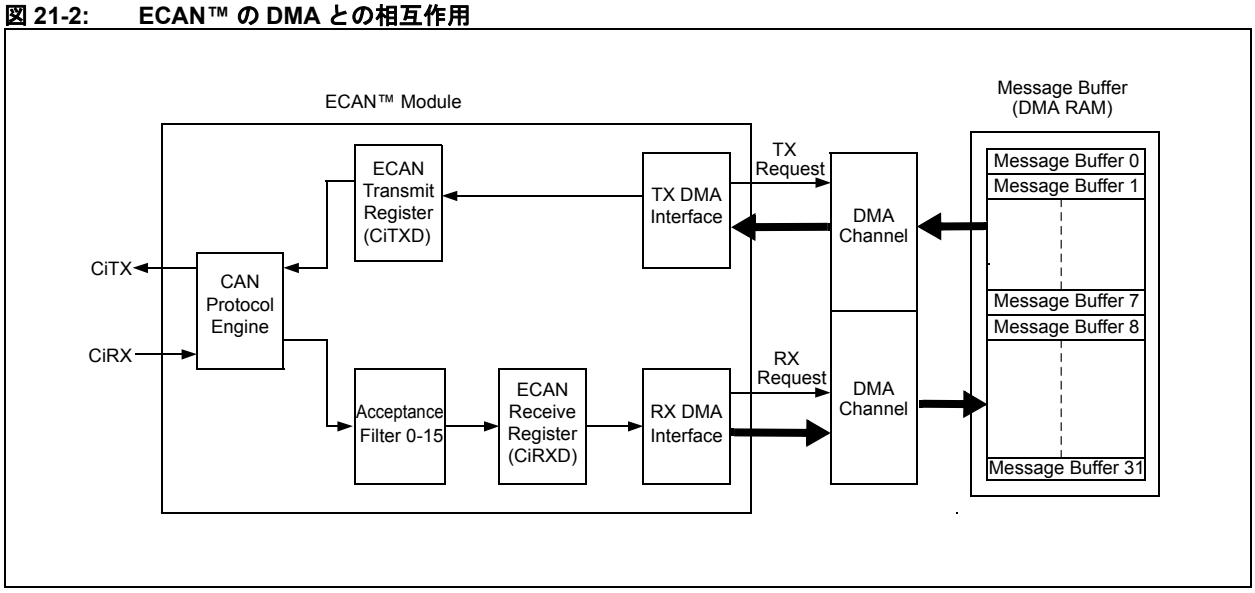
- メッセージ送信用に 8 つのメッセージ バッファを設定可能
- 送信に使用するメッセージ バッファの優先度はユーザ定義可能
- メッセージ送信用の DMA インターフェイス

その他 :

- セルフテスト、システム診断、バス監視用のループバック モード、リッスンオールメッセージ モード、リッスンオンリー モード
- 低消費電力動作モード

図 21-2 に、ECAN モジュールの一般的な構成と、DMA コントローラおよび DMA RAM との相互作用を示します。

図 21-2: ECAN™ の DMA との相互作用



21.1.1 ECAN モジュール

ECAN モジュールは、プロトコル エンジン、メッセージ アクセプタンス フィルタ、独立した送信 / 受信 DMA インターフェイスで構成されます。プロトコル エンジンは、CAN バスを経由してメッセージを送受信します (CAN バス 2.0B プロトコルに準拠)。モジュールはユーザ設定可能なアクセプタンス フィルタを使用して、受信したメッセージを検査し、メッセージを DMA メッセージ バッファに格納するか破棄するかを判定します。

受信メッセージの場合、受信 DMA インターフェイスは受信データ割り込みを生成し、DMA サイクルを開始します。受信 DMA チャンネルは、CiRXD レジスタからデータを読み出し、メッセージ バッファに書き込みます。

送信メッセージの場合、送信 DMA インターフェイスは送信データ割り込みを生成し、DMA サイクルを開始します。送信 DMA チャンネルは、メッセージ バッファから読み出し、メッセージ送信用に CiTXD レジスタに書き込みます。

21.1.2 メッセージ バッファ

ECAN モジュールは、CAN バスで送受信するデータを格納するために最大 32 のメッセージ バッファをサポートしています。これらのバッファは DMA RAM 内に存在します。メッセージ バッファ 0-7 は、送信動作と受信動作のいずれにも設定可能です。メッセージ バッファ 8-31 は、受信専用バッファでメッセージ送信には使用できません。

21.1.3 DMA コントローラ

DMA コントローラはメッセージ バッファと ECAN 間のインターフェイスとして動作し、CPU に負荷をかけずにデータを双方向に転送可能です。DMA コントローラは、DMA RAM と dsPIC33F 周辺モジュールの間でデータを転送するために最大 8 つのチャンネルをサポートしています。CAN メッセージの送信と受信をサポートするには、独立した 2 つの DMA チャンネルが必要です。

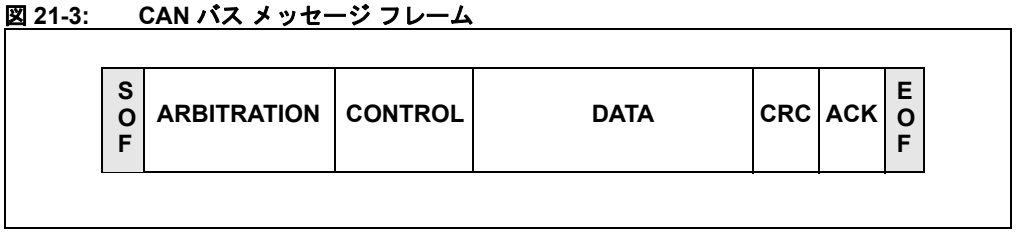
各 DMA チャンネルは、DMA 要求レジスタ (DMAxREQ) を備えています。ユーザ アプリケーションではこのレジスタを使用して割り込みイベントを割り当て、DMA ベースのメッセージ転送をトリガします。

21.2 CAN メッセージ フォーマット

CAN バス プロトコルは非同期通信を使用します。情報はトランスミッタからレシーバにデータフレーム単位で渡されます。データフレームは、図 21-3 に示すようにデータフレームの内容を定義するバイト フィールドで構成されています。

各フレームは SOF (Start-of-Frame) ビットから始まり、EOF (End-Of-Frame) ビット フィールドで終わります。SOF (Start-of-Frame) に続くアービトレーション フィールドと制御フィールドは、メッセージ タイプ、フォーマット、長さ、優先度を示します。この情報により、CAN バス上の各ノードはメッセージに対して適切に応答できます。データフィールドは、メッセージを格納する 0 ～ 8 バイトの可変長フィールドです。巡回冗長検査 (CRC) フィールドと肯定応答 (ACK) フィールドにより、エラーを検出します。

図 21-3: CAN バス メッセージ フレーム



CAN バス プロトコルは以下の 5 つのフレーム タイプをサポートしています。

- **データフレーム** – データをトランスミッタからレシーバに送信する
- **リモートフレーム** – バス上のノードが、別のノードから同じ識別子を持つデータフレームを送信するよう要求するために送信する
- **エラーフレーム** – ノードがエラーを検出した時に送信する
- **オーバーロード フレーム** – データフレームまたはリモートフレームを連続して送信する時に次のフレームを遅延させる
- **インターフレーム スペース** – 連続するフレーム同士を分離する

CAN 2.0B 仕様では、さらに以下の 2 つのデータ フォーマットを定義しています。

- **標準データフレーム** – 11 ビットの識別子を使用する標準メッセージ用
- **拡張データフレーム** –29 ビットの識別子を使用する拡張メッセージ用

CAN バス仕様には、以下の 3 つのバージョンがあります。

- **2.0A** – 29 ビットの識別子をエラーとみなす
- **2.0B パッシブ** – 29 ビットの識別子メッセージを無視する
- **2.0B アクティブ** – 11 ビットと 29 ビットの両方の識別子を処理する

dsPIC33F ECAN モジュールは CAN 2.0B アクティブ仕様に準拠するだけでなく、より強力なメッセージ フィルタ処理機能を備えています。

Note:	CAN プロトコルの詳細は、Bosch CAN バス仕様書を参照してください。

21.2.1 標準データフレーム

標準データフレーム メッセージは、図 21-4 に示すように SOF (Start-of-Frame) ビットから始まり、12 ビットのアービトレーション フィールドが続きます。アービトレーション フィールドには、11 ビットの識別子とリモート送信要求 (RTR) ビットが含まれます。識別子はメッセージに含まれる情報のタイプを定義します。また、各受信ノードは識別子によってメッセージが自ノードに関連するかどうかを判定します。RTR ビットはデータフレームとリモートフレームを識別します。標準データフレームの場合、RTR ビットはクリアされています。

アービトレーション フィールドに続くのは 6 ビットの制御フィールドで、メッセージの内容に関する追加情報を提供します。制御フィールドの最初のビットは識別子拡張 (IDE) ビットで、メッセージが標準データフレームか拡張データフレームかを識別します。標準データフレームは、IDE ビット送信時のドミナント状態 (論理レベル「0」) で示されます。制御フィールドの 2 番目のビットは予約済み (RB0) ビットで、ドミナント状態 (論理レベル「0」) です。制御フィールドの残り 4 ビットはデータ長コード (DLC) で、メッセージデータのバイト数を示します。

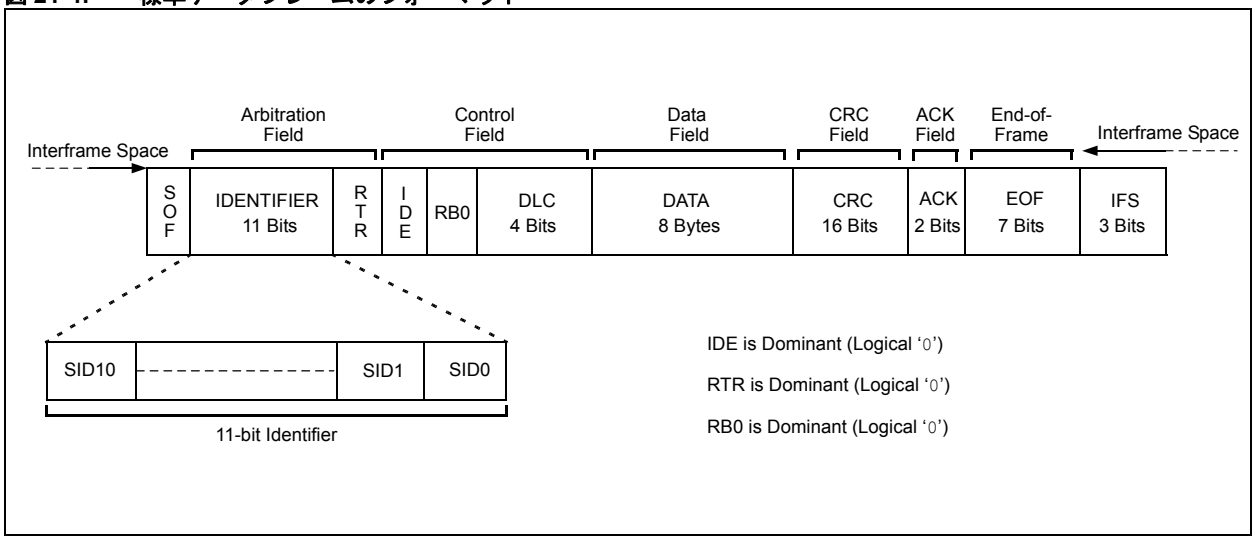
制御フィールドの後にはデータフィールドが続きます。このフィールドはメッセージ データ (データフレームの実際のペイロード) を格納しています。このフィールドは 0 ～ 8 バイトの可変長です。バイト数はユーザが選択可能です。

データフィールドの後に巡回冗長検査フィールドが続きます。これは 15 ビットの CRC シーケンスと 1 ビットのデリミタです。

肯定応答 (ACK) フィールドはリセッシブビット (論理レベル「1」) として送信され、データを正しく受信したレシーバによりドミナントビットとして上書きされます。メッセージはアクセプタンス フィルタの比較結果にかかわらず、レシーバによって肯定応答されます。

最後のフィールドは EOF (End-of-Frame) フィールドで、これはメッセージの終わりを示す 7 ビットのリセッシブビットです。

図 21-4: 標準データフレームのフォーマット



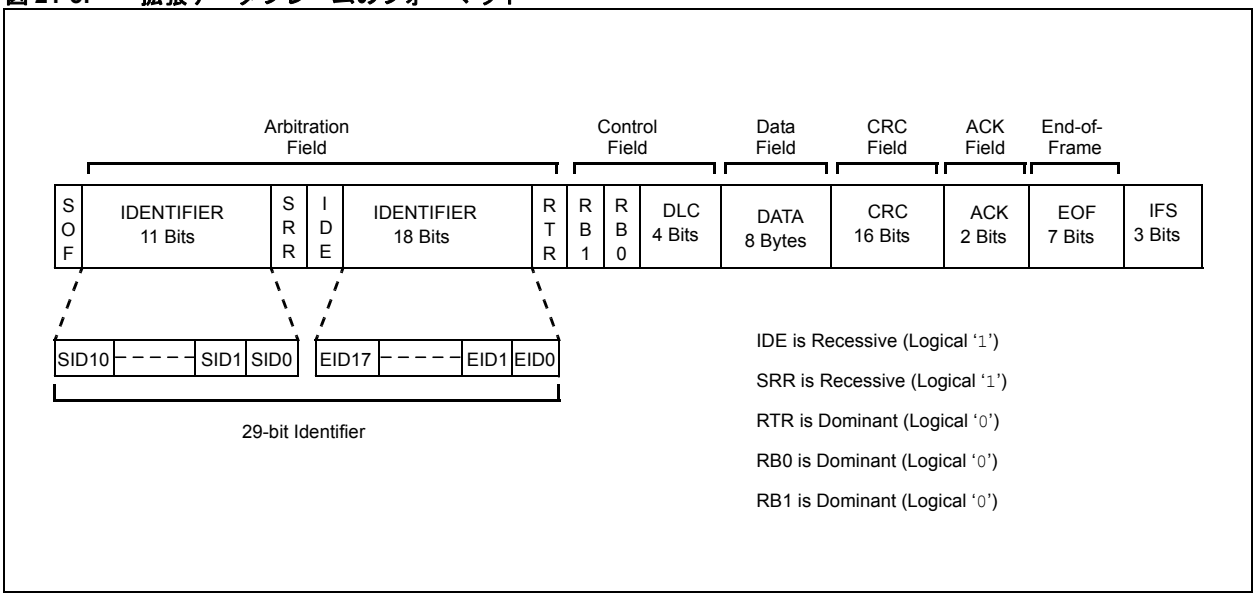
21.2.2 拡張データフレーム

拡張データフレームは、図 21-5 に示すように SOF ビットから始まり、31 ビットのアービトレーション フィールドが続きます。拡張データフレームのアービトレーション フィールドには、29 ビットの識別子が含まれ、代替リモート要求 (SRR) ビットと IDE ビットで区切られた 2 つのフィールドに格納されます。拡張データフレームの場合、SRR = 1 です。IDE ビットはデータフレームのタイプを示します。拡張データフレームの場合、IDE = 1 です。

拡張データフレームの制御フィールドは 7 ビットで構成されています。最初のビットは RTR です。拡張データフレームの場合、RTR = 0 です。続く 2 つのビット RB1 と RB0 は、ドミナント状態の予約済みビット (論理レベル「0」) です。制御フィールドの残り 4 ビットはデータ長コードで、メッセージデータのバイト数を示します。

拡張データフレーム内の残りのフィールドは標準データフレームと同様です。

図 21-5: 拡張データフレームのフォーマット



21.2.3 リモートフレーム

別のノードからのデータ受信を要求するノードは、リモートフレームを送信する事によって、送信元ノードからの所定のデータの転送を開始できます。リモートフレームは、標準フォーマット (図 21-6 参照) または拡張フォーマット (図 21-7 参照) のいずれかの形式です。

リモートフレームはデータフレームに似ていますが、以下の点が異なります。

- RTR ビットはリセッシブ (RTR = 1)
- データフィールドはなし (DLC = 0)

図 21-6: 標準リモートフレームのフォーマット

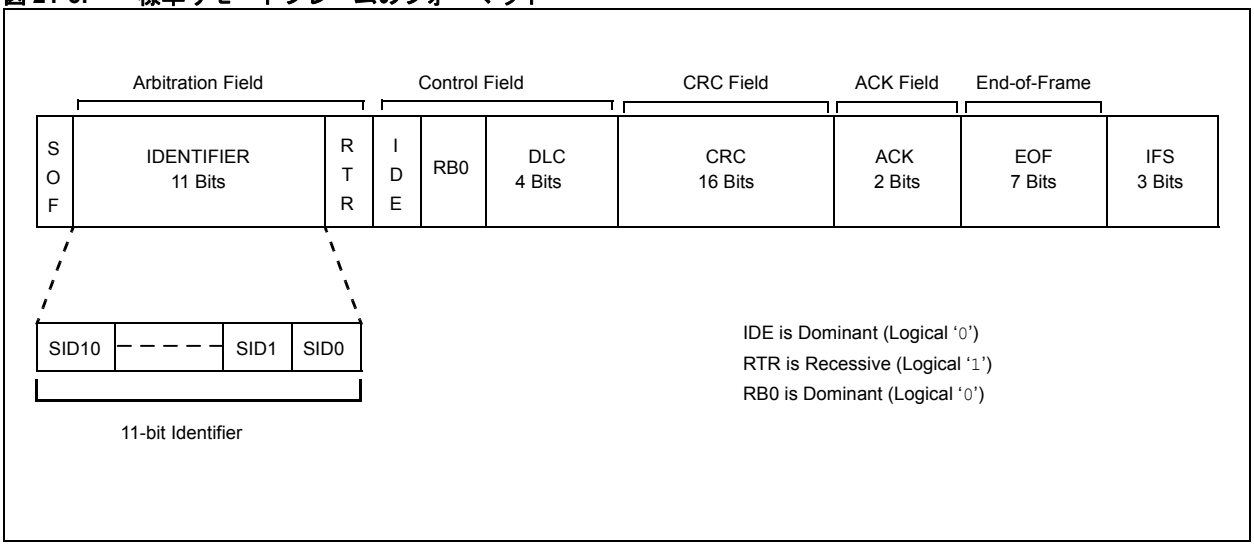
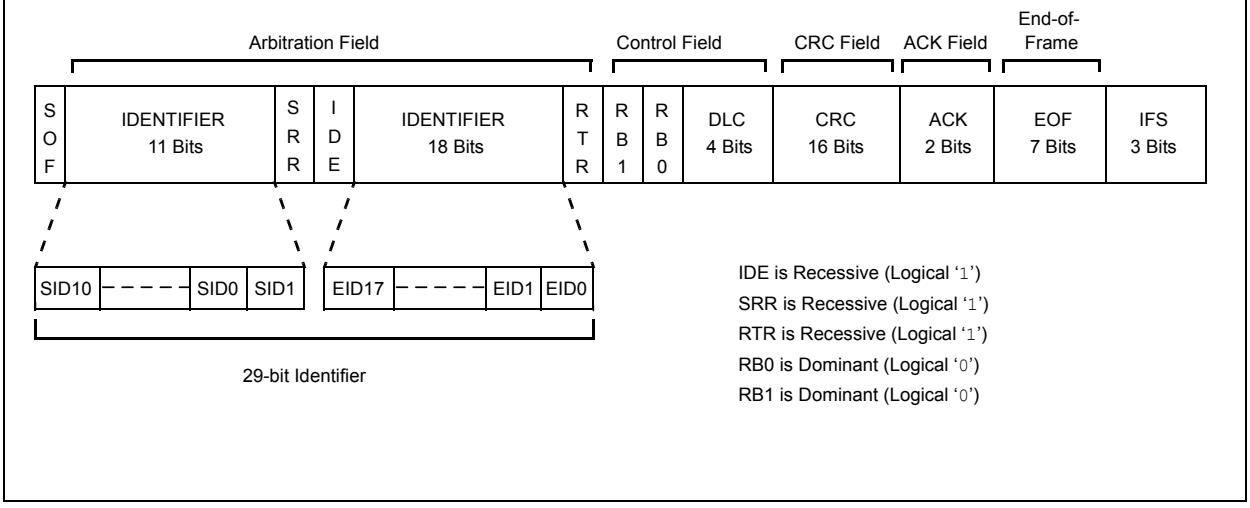


図 21-7: 拡張リモートフレームのフォーマット



21.2.4 エラーフレーム

エラーフレームはバスエラーを検出するノードが生成します。エラーフレームは、エラーフラグ フィールドと、続くエラーデリミタ フィールドで構成されます。エラーデリミタは 8 ビットのリセッシブビットで構成されており、エラー発生後バスノードはクリーンな状態で通信を再開する事が可能です。エラーフラグ フィールドには、エラーを検出するノードのエラーステータスに応じて、以下の 2 つのタイプがあります。

- **エラーアクティブ フラグ** – 連続する 6 ビットのドミナントビットが含まれる。これにより、ネットワーク上にある他の全てのノードに強制的にエラーエコー フラグを発生させる事から、バス上に連続する 6 ～ 12 のドミナントビットが出現する。
- **エラーパッシブ フラグ** – 連続する 6 ビットのリセッシブビットが含まれる。その結果、送信中のノードがバスエラーを検出しない限り、エラーパッシブ フラグの送信はネットワーク上の他のノードの通信に影響を与えない。

21.2.5 オーバーロード フレーム

ノードがオーバーロード フレームを生成するのは、インターフレーム スペースにおいてドミナントビットが検出されるか、ノードが次のメッセージを受信する準備が整わない場合です (例えば、ノードが前に受信したメッセージをまだ読み出し中の場合)。オーバーロード フレームはアクティブエラー フラグを持つエラーフレームと同様のフォーマットですが、生成されるのはインターフレーム スペースの間に限られます。6 ビットのドミナントビットを持つオーバーロード フラグ フィールドの後に、8 ビットのリセッシブビットを持つオーバーロード デリミタ フィールドが続く構成です。ノードは、次のメッセージの開始を遅延させるために、連続して最大 2 つのオーバーロード フレームを生成可能です。

21.2.6 インターフレーム スペース

インターフレーム スペースは、CAN バス上に送信されている連続するフレームを分離します。少なくとも 3 ビットのリセッシブビットで構成され、インターミッションと呼ばれます。インターフレーム スペースにより、ノードは次のフレームの開始前に、先に受信したメッセージを内部的に処理する時間が確保できます。送信ノードがエラーパッシブ状態の場合、ノードが他のメッセージを送信する前にインターフレーム スペースに8ビットのリセッシブビットを追加で挿入します。この期間を送信待機フィールドと呼び、他の送信ノードにバスを制御する時間を与えます。

21.3 ECAN レジスタ

ECAN モジュールには、メッセージ アクセプタンス フィルタとメッセージ バッファの設定に使用する特殊機能レジスタ (SFR) が多数あります。データ RAM 空間を効果的に利用するには、複数の SFR セットを同じ一連のメモリ アドレスにマッピングします。ECAN 制御レジスタ 1 (CiCTRL1<0>) の SFR マップウィンドウ選択 (WIN) ビットを使用して、SFR セットのいずれかに選択的にアクセスします。

CiCTRL1<WIN> = 1 の場合、ユーザ アプリケーションはメッセージ アクセプタンス フィルタ、マスク、フィルタ バッファ ポインタ レジスタにアクセスします。

CiCTRL1<WIN> = 0 の場合、ユーザ アプリケーションはバッファ制御レジスタ、ステータス レジスタ、データ送受信レジスタにアクセスします。

21.3.1 ECAN baud レート制御レジスタ

- **CiCFG1: ECAN™ baud レート コンフィグレーション レジスタ 1**

  このレジスタには、baud レート プリスケーラを使用して、時間単位 (TQ) を設定する制御ビットが含まれており、同期ジャンプ幅を時間単位で指定します (レジスタ 21-1 参照)。

Note:	レジスタ識別名で「i」と示される部分は、ECAN 1 または ECAN 2 を表します。

- **CiCFG2: ECAN™ baud レート コンフィグレーション レジスタ 2**

  このレジスタを使用して各 CAN ビット セグメントの時間単位数をプログラムします。この中には伝搬セグメントと位相セグメント 1 と 2 も含まれます (レジスタ 21-2 参照)。

21.3.2 ECAN メッセージ フィルタ レジスタ

- **CiFEN1: ECAN™ アクセプタンス フィルタ イネーブル レジスタ**

  このレジスタは、メッセージ フィルタ処理用のアクセプタンス フィルタ 0 ～ 15 を有効 / 無効にします (レジスタ 21-3 参照)。

- **CiRXFnSID: ECAN™ アクセプタンス フィルタ標準識別子レジスタ n (n = 0-15)**

  これら 16 個のレジスタは、アクセプタンス フィルタ 0 ～ 15 の標準メッセージ識別子を指定します。識別子ビットは、メッセージを受け入れるか拒否するかを判定するために受信メッセージ識別子を選択してマスクします (レジスタ 21-4 参照)。これらのレジスタには、WIN ビットがセットされている場合 (CiCTRL1<0> = 1 = フィルタ ウィンドウ使用) にのみユーザ アプリケーションからアクセス可能です。

- **CiRXFnEID: ECAN™ アクセプタンス フィルタ拡張識別子レジスタ n (n = 0-15)**

  これら 16 個のレジスタは、アクセプタンス フィルタ 0 ～ 15 の拡張メッセージ識別子を指定します。識別子ビットは、メッセージを受け入れるか拒否するかを判定するために受信メッセージ識別子を選択してマスクします (レジスタ 21-5 参照)。これらのレジスタには、WIN ビットがセットされている場合 (CiCTRL1<0> = 1) にのみユーザ アプリケーションからアクセス可能です。

- **CiRXMnSID: ECAN™ アクセプタンス フィルタ マスク標準識別子レジスタ n (n = 0-2)**

  これら 3 種のレジスタは、アクセプタンス マスク 0、1、2 の標準識別子マスクビットを指定します。アクセプタンス フィルタは、識別子ビットを選択して比較するために、マスク レジスタのいずれかをオプションで選択できます (レジスタ 21-6 参照)。これらのレジスタには、WIN ビットがセットされている場合 (CiCTRL1<0> = 1) にのみユーザ アプリケーションからアクセス可能です。

- **CiRXMnEID: ECAN™ アクセプタンス フィルタ マスク拡張識別子 レジスタ n (n = 0-2)**

  マスク 0、1、2 のアクセプタンス マスクビットを指定する 3 対のレジスタが用意されています。アクセプタンス フィルタは、識別子ビットを選択して比較するために、マスク レジスタのいずれかをオプションで選択できます (レジスタ 21-7 参照)。これらのレジスタには、WIN ビットがセットされている場合 (CiCTRL1<0> = 1) にのみユーザ アプリケーションからアクセス可能です。

- **CiFMSKSEL1: ECAN™ フィルタ 7-0 マスク選択レジスタ 1**

  このレジスタは CiFMSKSEL2 と併せて使用し、アクセプタンス フィルタ 0 ～ 7 に対応するアクセプタンス マスクを選択します (レジスタ 21-8 参照)。

- **CiFMSKSEL2: ECAN™ フィルタ 15-8 マスク選択レジスタ 2**

  このレジスタは CiFMSKSEL1 と併せて使用し、アクセプタンス フィルタ 8 ～ 15 に対応するアクセプタンス マスクを選択します (レジスタ 21-9 参照)。

- **CiBUFPNT1: ECAN™ フィルタ 0-3 バッファポインタ レジスタ 1**

  このレジスタは、アクセプタンス フィルタ 0 ～ 3 が受け入れるメッセージを格納するメッセージ バッファを指定します (レジスタ 21-10 参照)。このレジスタには、WIN ビットがセットされている場合 (CiCTRL1<0> = 1) にのみユーザ アプリケーションからアクセス可能です。

- **CiBUFPNT2: ECAN™ フィルタ 4-7 バッファポインタ レジスタ 2**

  このレジスタは、アクセプタンス フィルタ 4 ～ 7 が受け入れるメッセージを格納するメッセージ バッファを指定します (レジスタ 21-11 参照)。このレジスタには、WIN ビットがセットされている場合 (CiCTRL1<0> = 1) にのみユーザ アプリケーションからアクセス可能です。

- **CiBUFPNT3: ECAN™ フィルタ 8-11 バッファポインタ レジスタ 3**

  このレジスタは、アクセプタンス フィルタ 8 ～ 11 が受け入れるメッセージを格納するメッセージ バッファを指定します (レジスタ 21-12 参照)。このレジスタには、WIN ビットがセットされている場合 (CiCTRL1<0> = 1) にのみユーザ アプリケーションからアクセス可能です。

- **CiBUFPNT4: ECAN™ フィルタ 12-15 バッファポインタ レジスタ 4**

  このレジスタは、アクセプタンス フィルタ 12 ～ 15 が受け入れるメッセージを格納するメッセージ バッファを指定します (レジスタ 21-13 参照)。このレジスタには、WIN ビットがセットされている場合 (CiCTRL1<0> = 1) にのみユーザ アプリケーションからアクセス可能です。

21.3.3 ECAN メッセージ バッファ ステータス レジスタ

- **CiRXFUL1: ECAN™ 受信バッファフル レジスタ 1**

  このレジスタは、CiRXFUL2 と組み合わせてメッセージ バッファ 0 ～ 31 のバッファフル ステータスを示します。受信メッセージがメッセージ バッファに格納されると、対応するバッファフル フラグがセットされます (レジスタ 21-14 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0 = バッファ ウィンドウ使用) にのみユーザ アプリケーションからアクセス可能です。

- **CiRXFUL2: ECAN™ 受信バッファフル レジスタ 2**

  このレジスタは、CiRXFUL1 と組み合わせてメッセージ バッファ 0 ～ 31 のバッファフル ステータスを示します。受信メッセージがメッセージ バッファに格納されると、対応するバッファフル フラグがセットされます (レジスタ 21-15 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

- **CiRXOVF1: ECAN™ 受信バッファ オーバーフロー レジスタ 1**

  このレジスタは、CiRXOVF2 と組み合わせてメッセージ バッファ 0 ～ 31 のオーバーフロー ステータスを示します。受信メッセージがメッセージ バッファに格納され、対応するバッファフル フラグがセットされている場合、メッセージは失われ、対応するバッファオーバーフロー フラグがセットされます (レジスタ 21-16 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

- **CiRXOVF2: ECAN™ 受信バッファ オーバーフロー レジスタ 2**

  このレジスタは、CiRXOVF1 と組み合わせてメッセージ バッファ 0 ～ 31 のオーバーフロー ステータスを示します。受信メッセージがメッセージ バッファに格納され、対応するバッファフル フラグがセットされている場合、メッセージは失われ、対応するバッファオーバーフロー フラグがセットされます (レジスタ 21-17 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

21.3.4 ECAN FIFO 制御 / ステータス レジスタ

- **CiFCTRL: ECAN™ FIFO 制御レジスタ**

  このレジスタは、受信バッファFIFO の動作を制御します。FIFO 開始アドレスと DMA RAM 内で ECAN 用に予約するメッセージ バッファ数を指定します (レジスタ 21-18 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

- **CiFIFO: ECAN™ FIFO ステータス レジスタ**

  このレジスタには、書き込みポインタと読み出しポインタを格納します。書き込みポインタは、最も新しい受信データを格納するバッファを示します。読み出しポインタは、次に読み出すバッファをユーザ アプリケーションに通知します (レジスタ 21-19 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

21.3.5 ECAN 割り込み制御 / ステータス レジスタ

- **CiINTF: ECAN™ 割り込みフラグ レジスタ**

  このレジスタは、ECAN モジュールの各種割り込み要因のステータスを示します (レジスタ 21-20 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

- **CiINTE: ECAN™ 割り込みイネーブル レジスタ**

  このレジスタを使用して、主要な 7 つの割り込み要因を選択的に有効 / 無効にします。7 つの割り込みとは、送信バッファ割り込み、受信バッファ割り込み、受信バッファ オーバーフロー割り込み、FIFO ほぼフル割り込み、エラー割り込み、ウェイクアップ割り込み、無効なメッセージ受信割り込みです (レジスタ 21-21 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

- **CiVEC: ECAN™ 割り込みコード レジスタ**

  このレジスタには、割り込みを効率的に処理するためにジャンプ テーブルと併せて使用できる割り込みコードビットがあります (レジスタ 21-22 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

21.3.6 ECAN 制御 / エラーカウンタ レジスタ

- **CiCTRL1: ECAN™ 制御レジスタ 1**

  このレジスタは ECAN モジュールの動作モードを設定します (レジスタ 21-23 参照)。

- **CiCTRL2: ECAN™ 制御レジスタ 2**

  このレジスタは DeviceNet フィルタ処理制御ビットを格納します (レジスタ 21-24 参照)。

- **CiTRmnCON: ECAN™ TX/RX バッファ m 制御レジスタ (m = 0,2,4,6; n = 1,3,5,7)**

  このレジスタはメッセージ バッファの設定と制御を行います (レジスタ 21-25 参照)。このレジスタには、WIN ビットがクリアされている場合 (CiCTRL1<0> = 0) にのみユーザ アプリケーションからアクセス可能です。

- **CiEC: ECAN™ 送信 / 受信エラーカウント レジスタ**

  このレジスタは送受信エラーをカウントします。ユーザ アプリケーションは、このレジスタを読み出して現在の送受信エラー数を判定します (レジスタ 21-26 参照)。

- **CiRXD: ECAN 受信データ レジスタ**

  このレジスタは受信した全てのワードを一時的に保持します。これは、DMA コントローラが DMA バッファへ渡すデータを読み出すレジスタです。

- **CiTXD: ECAN 送信データ レジスタ**

  このレジスタは全ての送信データを一時的に保持します。これは、DMA コントローラが DMA バッファからのデータを書き込むレジスタです。

レジスタ 21-1: CiCFG1: ECAN™ baud レート コンフィグレーション レジスタ 1

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 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
SJW<1:0>		BRP<5:0>					
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-8 **未実装 :** 「0」として読み出し

bit 7-6 **SJW<1:0>:** 同期ジャンプ幅ビット

11 = 長さは 4 x TQ
10 = 長さは 3 x TQ
01 = 長さは 2 x TQ
00 = 長さは 1 x TQ

bit 5-0 **BRP<5:0>:** baud レート プリスケーラ ビット

11 1111 = TQ = 2 x 64 x 1/FCAN
•
•
•
00 0010 = TQ = 2 x 3 x 1/FCAN
00 0001 = TQ = 2 x 2 x 1/FCAN
00 0000 = TQ = 2 x 1 x 1/FCAN

レジスタ 21-2: CiCFG2: ECAN™ baud レート コンフィグレーション レジスタ 2

U-0	R/W-x	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	WAKFIL	—	—	—	SEG2PH<2:0>		
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SEG2PHTS	SAM	SEG1PH<2:0>			PRSEG<2:0>		
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15 **未実装 :** 「0」として読み出し

bit 14 **WAKFIL**: ウェイクアップ用 CAN バス ラインフィルタ選択ビット
1 = CAN バス ラインフィルタをウェイクアップに使用する
0 = CAN バス ラインフィルタをウェイクアップに使用しない

bit 13-11 **未実装 :** 「0」として読み出し

bit 10-8 **SEG2PH<2:0>:** 位相セグメント 2 ビット
111 = 長さは 8 x TQ
•
•
•
000 = 長さは 1 x TQ

bit 7 **SEG2PHTS:** 位相セグメント 2 時間選択ビット
1 = 自由にプログラム可能
0 = SEG1PH ビットの最大値と情報処理時間 (IPT) のいずれか大きい値

bit 6 **SAM:** サンプル CAN バス ライン ビット
1 = バスラインをサンプル点で 3 回サンプリングする
0 = バスラインをサンプル点で 1 回サンプリングする

bit 5-3 **SEG1PH<2:0>:** 位相セグメント 1 ビット
111 = 長さは 8 x TQ
•
•
•
000 = 長さは 1 x TQ

bit 2-0 **PRSEG<2:0>:** 伝搬時間セグメントビット
111 = 長さは 8 x TQ
•
•
•
000 = 長さは 1 x TQ

レジスタ 21-3: CiFEN1: ECAN™ アクセプタンス フィルタ イネーブル レジスタ

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
FLTEN15	FLTEN14	FLTEN13	FLTEN12	FLTEN11	FLTEN10	FLTEN9	FLTEN8
bit 15							bit 8

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
FLTEN7	FLTEN6	FLTEN5	FLTEN4	FLTEN3	FLTEN2	FLTEN1	FLTEN0
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **FLTENn:** フィルタ n イネーブルビット
1 = フィルタ n を有効にしてメッセージを受け入れる
0 = フィルタ n を無効にする

レジスタ 21-4: CiRXFnSID: ECAN™ アクセプタンス フィルタ標準識別子レジスタ n (n = 0-15)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 15							bit 8

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-5 **SID<10:0>:** 標準識別子ビット
1 = フィルタに一致するにはメッセージ アドレス ビット SIDx は「1」である必要がある
0 = フィルタに一致するにはメッセージ アドレス ビット SIDx は「0」である必要がある

bit 4 **未実装 :** 「0」として読み出し

bit 3 **EXIDE:** 拡張識別子イネーブルビット
MIDE = 1 の場合 :
1 = 拡張識別子アドレスを持つメッセージにのみ一致する
0 = 標準識別子アドレスを持つメッセージにのみ一致する

MIDE = 0 の場合 :
EXIDE ビットを無視する

bit 2 **未実装 :** 「0」として読み出し

bit 1-0 **EID<17:16>:** 拡張識別子ビット
1 = フィルタに一致するにはメッセージ アドレス ビット EIDx は「1」である必要がある
0 = フィルタに一致するにはメッセージ アドレス ビット EIDx は「0」である必要がある

Note: フィルタにマスクを適用しない場合、フィルタは標準フレームのみを受け入れます。EXIDE ビットが「1」にセットされていたとしても、フィルタは拡張フレームを受け入れません。

レジスタ 21-5: CiRXFnEID: ECAN™ アクセプタンス フィルタ拡張識別子レジスタ n (n = 0-15)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **EID<15:0>:** 拡張識別子ビット
1 = フィルタに一致するにはメッセージ アドレス ビット EIDx は「1」である必要がある
0 = フィルタに一致するにはメッセージ アドレス ビット EIDx は「0」である必要がある

レジスタ 21-6: CiRXMnSID: ECAN™ アクセプタンス フィルタ マスク標準識別子レジスタ n (n = 0-2)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 15							bit 8

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	MIDE	—	EID17	EID16
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-5 **SID<10:0>:** 標準識別子ビット
1 = フィルタ比較にビット SIDx を含める
0 = フィルタ比較でビット SIDx は「ドントケア」である

bit 4 **未実装 :** 「0」として読み出し

bit 3 **MIDE:** 識別子受信モードビット
1 = フィルタのEXIDEビットに対応するメッセージタイプ(標準または拡張アドレス)のみに一致する
0 = フィルタが一致する場合、標準または拡張アドレス メッセージのいずれかに一致する
(すなわち、(フィルタ SID) = (メッセージ SID) の場合または (フィルタ SID/EID) = (メッセージ SID/EID) の場合)

bit 2 **未実装 :** 「0」として読み出し

bit 1-0 **EID<17:16>:** 拡張識別子ビット
1 = フィルタ比較にビット EIDx を含める
0 = フィルタ比較でビット EIDx は「ドントケア」である

レジスタ 21-7: CiRXMnEID: ECAN™ アクセプタンス フィルタ マスク拡張識別子 レジスタ n (n = 0-2)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **EID<15:0>:** 拡張識別子ビット
1 = フィルタ比較にビット EIDx を含める
0 = フィルタ比較ではビット EIDx は「ドントケア」である

レジスタ 21-8: CiFMSKSEL1: ECAN™ フィルタ 7-0 マスク選択レジスタ 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F7MSK<1:0>		F6MSK<1:0>		F5MSK<1:0>		F4MSK<1:0>	
bit 15							bit 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
F3MSK<1:0>		F2MSK<1:0>		F1MSK<1:0>		F0MSK<1:0>	
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-14 **F7MSK<1:0>:** フィルタ 7 用マスクソース ビット
11 = 予約済み
10 = アクセプタンス マスク 2 レジスタにマスクを含む
01 = アクセプタンス マスク 1 レジスタにマスクを含む
00 = アクセプタンス マスク 0 レジスタにマスクを含む

bit 13-12 **F6MSK<1:0>:** フィルタ 6 用マスクソース ビット (bit 15-14 と同じ値)

bit 11-10 **F5MSK<1:0>:** フィルタ 5 用マスクソース ビット (bit 15-14 と同じ値)

bit 9-8 **F4MSK<1:0>:** フィルタ 4 用マスクソース ビット (bit 15-14 と同じ値)

bit 7-6 **F3MSK<1:0>:** フィルタ 3 用マスクソース ビット (bit 15-14 と同じ値)

bit 5-4 **F2MSK<1:0>:** フィルタ 2 用マスクソース ビット (bit 15-14 と同じ値)

bit 3-2 **F1MSK<1:0>:** フィルタ 1 用マスクソース ビット (bit 15-14 と同じ値)

bit 1-0 **F0MSK<1:0>:** フィルタ 0 用マスクソース ビット (bit 15-14 と同じ値)

レジスタ 21-9: CiFMSKSEL2: ECAN™ フィルタ 15-8 マスク選択レジスタ 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F15MSK<1:0>		F14MSK<1:0>		F13MSK<1:0>		F12MSK<1:0>	
bit 15							bit 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
F11MSK<1:0>		F10MSK<1:0>		F9MSK<1:0>		F8MSK<1:0>	
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-14 **F15MSK<1:0>:** フィルタ 15 用マスクソース ビット
11 = 予約済み
10 = アクセプタンス マスク 2 レジスタにマスクを含む
01 = アクセプタンス マスク 1 レジスタにマスクを含む
00 = アクセプタンス マスク 0 レジスタにマスクを含む

bit 13-12 **F14MSK<1:0>:** フィルタ 14 用マスクソース ビット (bit 15-14 と同じ値)

bit 11-10 **F13MSK<1:0>:** フィルタ 13 用マスクソース ビット (bit 15-14 と同じ値)

bit 9-8 **F12MSK<1:0>:** フィルタ 12 用マスクソース ビット (bit 15-14 と同じ値)

bit 7-6 **F11MSK<1:0>:** フィルタ 11 用マスクソース ビット (bit 15-14 と同じ値)

bit 5-4 **F10MSK<1:0>:** フィルタ 10 用マスクソース ビット (bit 15-14 と同じ値)

bit 3-2 **F9MSK<1:0>:** フィルタ 9 用マスクソース ビット (bit 15-14 と同じ値)

bit 1-0 **F8MSK<1:0>:** フィルタ 8 用マスクソース ビット (bit 15-14 と同じ値)

レジスタ 21-10: CiBUFPNT1: ECAN™ フィルタ 0-3 バッファポインタ レジスタ 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F3BP<3:0>				F2BP<3:0>			
bit 15							bit 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
F1BP<3:0>				F0BP<3:0>			
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-12 **F3BP<3:0>:** フィルタ 3 用 RX バッファ マスクビット
1111 = RX FIFO バッファで受信する事でフィルタが一致する
1110 = RX バッファ 14 で受信する事でフィルタが一致する
•
•
•
0001 = RX バッファ 1 で受信する事でフィルタが一致する
0000 = RX バッファ 0 で受信する事でフィルタが一致する

bit 11-8 **F2BP<3:0>:** フィルタ 2 用 RX バッファ マスクビット (bit 15-12 と同じ値)

bit 7-4 **F1BP<3:0>:** フィルタ 1 用 RX バッファ マスクビット (bit 15-12 と同じ値)

bit 3-0 **F0BP<3:0>:** フィルタ 0 用 RX バッファ マスクビット (bit 15-12 と同じ値)

レジスタ 21-11: CiBUFPNT2: ECAN™ フィルタ 4-7 バッファポインタ レジスタ 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F7BP<3:0>				F6BP<3:0>			
bit 15							bit 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
F5BP<3:0>				F4BP<3:0>			
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-12 **F7BP<3:0>:** フィルタ 7 用 RX バッファ マスクビット
1111 = RX FIFO バッファで受信する事でフィルタが一致する
1110 = RX バッファ 14 で受信する事でフィルタが一致する
•
•
•
0001 = RX バッファ 1 で受信する事でフィルタが一致する
0000 = RX バッファ 0 で受信する事でフィルタが一致する

bit 11-8 **F6BP<3:0>:** フィルタ 6 用 RX バッファ マスクビット (bit 15-12 と同じ値)

bit 7-4 **F5BP<3:0>:** フィルタ 5 用 RX バッファ マスクビット (bit 15-12 と同じ値)

bit 3-0 **F4BP<3:0>:** フィルタ 4 用 RX バッファ マスクビット (bit 15-12 と同じ値)

レジスタ 21-12: CiBUFPNT3: ECAN™ フィルタ 8-11 バッファポインタ レジスタ 3

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F11BP<3:0>				F10BP<3:0>			
bit 15							bit 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
F9BP<3:0>				F8BP<3:0>			
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-12 **F11BP<3:0>:** フィルタ 11 用 RX バッファ マスクビット
1111 = RX FIFO バッファで受信する事でフィルタが一致する
1110 = RX バッファ 14 で受信する事でフィルタが一致する
•
•
•
0001 = RX バッファ 1 で受信する事でフィルタが一致する
0000 = RX バッファ 0 で受信する事でフィルタが一致する

bit 11-8 **F10BP<3:0>:** フィルタ 10 用 RX バッファ マスクビット (bit 15-12 と同じ値)

bit 7-4 **F9BP<3:0>:** フィルタ 9 用 RX バッファ マスクビット (bit 15-12 と同じ値)

bit 3-0 **F8BP<3:0>:** フィルタ 8 用 RX バッファ マスクビット (bit 15-12 と同じ値)

レジスタ 21-13: CiBUFPNT4: ECAN™ フィルタ 12-15 バッファポインタ レジスタ 4

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
F15BP<3:0>				F14BP<3:0>			
bit 15							bit 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
F13BP<3:0>				F12BP<3:0>			
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-12 **F15BP<3:0>:** フィルタ 15 用 RX バッファ マスクビット
1111 = RX FIFO バッファで受信する事でフィルタが一致する
1110 = RX バッファ 14 で受信する事でフィルタが一致する
•
•
•
0001 = RX バッファ 1 で受信する事でフィルタが一致する
0000 = RX バッファ 0 で受信する事でフィルタが一致する

bit 11-8 **F14BP<3:0>:** フィルタ 14 用 RX バッファ マスクビット (bit 15-12 と同じ値)

bit 7-4 **F13BP<3:0>:** フィルタ 13 用 RX バッファ マスクビット (bit 15-12 と同じ値)

bit 3-0 **F12BP<3:0>:** フィルタ 12 用 RX バッファ マスクビット (bit 15-12 と同じ値)

レジスタ 21-14: CiRXFUL1: ECAN™ 受信バッファフル レジスタ 1

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXFUL15	RXFUL14	RXFUL13	RXFUL12	RXFUL11	RXFUL10	RXFUL9	RXFUL8
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXFUL7	RXFUL6	RXFUL5	RXFUL4	RXFUL3	RXFUL2	RXFUL1	RXFUL0
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **RXFUL<15:0>:** 受信バッファ n フル ビット
1 = バッファはフルである (モジュールでセットする)
0 = バッファはエンプティである

レジスタ 21-15: CiRXFUL2: ECAN™ 受信バッファフル レジスタ 2

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXFUL31	RXFUL30	RXFUL29	RXFUL28	RXFUL27	RXFUL26	RXFUL25	RXFUL24
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXFUL23	RXFUL22	RXFUL21	RXFUL20	RXFUL19	RXFUL18	RXFUL17	RXFUL16
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **RXFUL<31:16>:** 受信バッファ n フル ビット
1 = バッファはフルである (モジュールでセットする)
0 = バッファはエンプティである

レジスタ 21-16: CiRXOVF1: ECAN™ 受信バッファ オーバーフロー レジスタ 1

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXOVF15	RXOVF14	RXOVF13	RXOVF12	RXOVF11	RXOVF10	RXOVF9	RXOVF8
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXOVF7	RXOVF6	RXOVF5	RXOVF4	RXOVF3	RXOVF2	RXOVF1	RXOVF0
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **RXOVF<15:0>:** 受信バッファ n オーバーフロー ビット
1 = モジュールはフル バッファへの書き込みを試みた (モジュールでセットする)
0 = オーバーフロー条件はない

レジスタ 21-17: CiRXOVF2: ECAN™ 受信バッファ オーバーフロー レジスタ 2

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXOVF31	RXOVF30	RXOVF29	RXOVF28	RXOVF27	RXOVF26	RXOVF25	RXOVF24
bit 15							bit 8

R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
RXOVF23	RXOVF22	RXOVF21	RXOVF20	RXOVF19	RXOVF18	RXOVF17	RXOVF16
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **RXOVF<31:16>:** 受信バッファ n オーバーフロー ビット
1 = モジュールはフル バッファへの書き込みを試みた (モジュールでセットする)
0 = オーバーフロー条件はない

レジスタ 21-18: CiFCTRL: ECAN™ FIFO 制御レジスタ

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
DMABS<2:0>			—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	FSA<4:0>				
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-13 **DMABS<2:0>:** DMA バッファサイズ ビット

111 = 予約済み
110 = DMA RAM に 32 バッファ
101 = DMA RAM に 24 バッファ
100 = DMA RAM に 16 バッファ
011 = DMA RAM に 12 バッファ
010 = DMA RAM に 8 バッファ
001 = DMA RAM に 6 バッファ
000 = DMA RAM に 4 バッファ

bit 12-5 **未実装 :** 「0」として読み出し

bit 4-0 **FSA<4:0>:** FIFO 開始領域ビット

11111 = 読み出しバッファ RB31
11110 = 読み出しバッファ RB30
•
•
•
00010 = TX/RX バッファ TRB2
00001 = TX/RX バッファ TRB1
00000 = TX/RX バッファ TRB0

レジスタ 21-19: CiFIFO: ECAN™ FIFO ステータス レジスタ

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
—	—	FBP<5:0>					
bit 15							bit 8

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
—	—	FNRB<5:0>					
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-14 **未実装 :** 「0」として読み出し

bit 13-8 **FBP<5:0>:** FIFO バッファポインタ ビット

011111 = RB31 バッファ
011110 = RB30 バッファ
•
•
•
000001 = TRB1 バッファ
000000 = TRB0 バッファ

bit 7-6 **未実装 :** 「0」として読み出し

bit 5-0 **FNRB<5:0>:** FIFO 次読み出しバッファポインタ ビット

011111 = RB31 バッファ
011110 = RB30 バッファ
•
•
•
000001 = TRB1 バッファ
000000 = TRB0 バッファ

レジスタ 21-20: CiINTF: ECAN™ 割り込みフラグ レジスタ

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
—	—	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN
bit 15							bit 8

R/C-0	R/C-0	R/C-0	U-0	R/C-0	R/C-0	R/C-0	R/C-0
IVRIF	WAKIF	ERRIF	—	FIFOIF	RBOVIF	RBIF	TBIF
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-14 **未実装 :** 「0」として読み出し

bit 13 **TXBO:** トランスミッタ エラー状態バス OFF ビット
1 = トランスミッタはバス OFF 状態である
0 = トランスミッタはバス OFF 状態でない

bit 12 **TXBP:** トランスミッタ エラー状態バスパッシブ ビット
1 = トランスミッタはバスパッシブ状態である
0 = トランスミッタはバスパッシブ状態でない

bit 11 **RXBP** レシーバ エラー状態バスパッシブ ビット
1 = レシーバはバスパッシブ状態である
0 = レシーバはバスパッシブ状態でない

bit 10 **TXWAR:** トランスミッタ エラー状態警告ビット
1 = トランスミッタはエラー警告状態である
0 = トランスミッタはエラー警告状態でない

bit 9 **RXWAR:** レシーバ エラー状態警告ビット
1 = レシーバはエラー警告状態である
0 = レシーバはエラー警告状態でない

bit 8 **EWARN:** トランスミッタまたはレシーバ エラー状態警告ビット
1 = トランスミッタまたはレシーバはエラー警告状態である
0 = トランスミッタまたはレシーバはエラー警告状態でない

bit 7 **IVRIF:** 無効メッセージ割り込みフラグビット
1 = 割り込み要求が発生した
0 = 割り込み要求は発生していない

bit 6 **WAKIF:** バス ウェイクアップ アクティビティ割り込みフラグビット
1 = 割り込み要求が発生した
0 = 割り込み要求は発生していない

bit 5 **ERRIF:** エラー割り込みフラグビット (CiINTF<13:8> レジスタ内の複数の要因)
1 = 割り込み要求が発生した
0 = 割り込み要求は発生していない

bit 4 **未実装 :** 「0」として読み出し

bit 3 **FIFOIF:** FIFO ほぼフル割り込みフラグビット
1 = 割り込み要求が発生した
0 = 割り込み要求は発生していない

bit 2 **RBOVIF:** RX バッファ オーバーフロー割り込みフラグビット
1 = 割り込み要求が発生した
0 = 割り込み要求は発生していない

bit 1 **RBIF:** RX バッファ割り込みフラグビット
1 = 割り込み要求が発生した
0 = 割り込み要求は発生していない

bit 0 **TBIF:** TX バッファ割り込みフラグビット
1 = 割り込み要求が発生した
0 = 割り込み要求は発生していない

レジスタ 21-21: CiINTE: ECAN™ 割り込みイネーブル レジスタ

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
IVRIE	WAKIE	ERRIE	—	FIFOIE	RBOVIE	RBIE	TBIE
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-8 **未実装 :** 「0」として読み出し

bit 7 **IVRIE:** 無効メッセージ割り込みイネーブルビット
1 = 割り込み要求を有効にする
0 = 割り込み要求を無効にする

bit 6 **WAKIE:** バス ウェイクアップ アクティビティ割り込みフラグビット
1 = 割り込み要求を有効にする
0 = 割り込み要求を無効にする

bit 5 **ERRIE:** エラー割り込みイネーブルビット
1 = 割り込み要求を有効にする
0 = 割り込み要求を無効にする

bit 4 **未実装 :** 「0」として読み出し

bit 3 **FIFOIE:** FIFO ほぼフル割り込みイネーブルビット
1 = 割り込み要求を有効にする
0 = 割り込み要求を無効にする

bit 2 **RBOVIE:** RX バッファ オーバーフロー割り込みイネーブルビット
1 = 割り込み要求を有効にする
0 = 割り込み要求を無効にする

bit 1 **RBIE:** RX バッファ割り込みイネーブルビット
1 = 割り込み要求を有効にする
0 = 割り込み要求を無効にする

bit 0 **TBIE:** TX バッファ割り込みイネーブルビット
1 = 割り込み要求を有効にする
0 = 割り込み要求を無効にする

レジスタ 21-22: CiVEC: ECAN™ 割り込みコード レジスタ

U-0	U-0	U-0	R-0	R-0	R-0	R-0	R-0
—	—	—	FILHIT<4:0>				
bit 15							bit 8

U-0	R-1	R-0	R-0	R-0	R-0	R-0	R-0
—	ICODE<6:0>						
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-13 **未実装 :** 「0」として読み出し

bit 12-8 **FILHIT<4:0>:** フィルター致番号ビット

10000-11111 = 予約済み
01111 = フィルタ 15
•
•
•
00001 = フィルタ 1
00000 = フィルタ 0

bit 7 **未実装 :** 「0」として読み出し

bit 6-0 **ICODE<6:0>:** 割り込みフラグコード ビット

1000101-1111111 = 予約済み
1000100 = FIFO ほぼフル割り込み
1000011 = レシーバ オーバーフロー割り込み
1000010 = ウェイクアップ割り込み
1000001 = エラー割り込み
1000000 = 割り込みなし
•
•
•
0100000-0111111 = 予約済み
0011111 = RB31 バッファ割り込み
0011110 = RB30 バッファ割り込み
•
•
•
0001001 = RB9 バッファ割り込み
0001000 = RB8 バッファ割り込み
0000111 = TRB7 バッファ割り込み
0000110 = TRB6 バッファ割り込み
0000101 = TRB5 バッファ割り込み
0000100 = TRB4 バッファ割り込み
0000011 = TRB3 バッファ割り込み
0000010 = TRB2 バッファ割り込み
0000001 = TRB1 バッファ割り込み
0000000 = TRB0 バッファ割り込み

レジスタ 21-23: CiCTRL1: ECAN™ 制御レジスタ 1

U-0	U-0	R/W-0	R/W-0	r-0	R/W-1	R/W-0	R/W-0
—	—	CSIDL	ABAT	—	REQOP<2:0>		
bit 15							bit 8

R-1	R-0	R-0	U-0	R/W-0	U-0	U-0	R/W-0
OPMODE<2:0>			—	CANCAP	—	—	WIN
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする	r = ビットは予約済み「0」の書き込みのみ可
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　x = ビットは未知

bit 15-14 **未実装 :** 「0」として読み出し

bit 13 **CSIDL:** アイドルモード時停止ビット
1 = デバイスがアイドルモードに移行するとモジュールの動作を停止する
0 = アイドルモード中もモジュールの動作を継続する

bit 12 **ABAT:** 保留中送信の一括中止 ビット
1 = 全送信バッファに送信中止を通知する
0 = モジュールは全送信が中止されるとこのビットをクリアする

bit 11 **予約済み :** 使用不可

bit 10-8 **REQOP<2:0>:** 要求動作モードビット
000 = 通常動作モードを設定する
001 = ディセーブル モードを設定する
010 = ループバック モードを設定する
011 = リッスンオンリー モードを設定する
100 = コンフィグレーション モードを設定する
101 = 予約済み
110 = 予約済み
111 = リッスンオールメッセージ モードを設定する

bit 7-5 **OPMODE<2:0>:** 動作モードビット
000 = モジュールは通常動作モードである
001 = モジュールはディセーブル モードである
010 = モジュールはループバック モードである
011 = モジュールはリッスンオンリー モードである
100 = モジュールはコンフィグレーション モードである
101 = 予約済み
110 = 予約済み
111 = モジュールはリッスンオールメッセージ モードである

bit 4 **未実装 :** 「0」として読み出し

bit 3 **CANCAP:** CAN メッセージ受信タイマ キャプチャ イベント イネーブルビット
1 = CAN メッセージ受信に基づき入力キャプチャを有効にする
0 = CAN キャプチャを無効にする

bit 2-1 **未実装 :** 「0」として読み出し

bit 0 **WIN:** SFR マップ ウィンドウ選択 ビット
1 = フィルタ ウィンドウを使用する
0 = バッファ ウィンドウを使用する

レジスタ 21-24: CiCTRL2: ECAN™ 制御レジスタ 2

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R-0	R-0	R-0	R-0	R-0
—	—	—	DNCNT<4:0>				
bit 7							bit 0

凡例 :		
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-5 **未実装 :** 「0」として読み出し

bit 4-0 **DNCNT<4:0>:** DeviceNet™ フィルタ ビット番号ビット

00000 = データバイトを比較しない
00001 = EID<17> とデータバイト 0 の bit 7 を比較する
00010 = EID<17:16> とデータバイト 0 のビット <7:6> を比較する
00011 = EID<17:15> とデータバイト 0 のビット <7:5> を比較する
00100 = EID<17:14> とデータバイト 0 のビット <7:4> を比較する
00101 = EID<17:13> とデータバイト 0 のビット <7:3> を比較する
00110 = EID<17:12> とデータバイト 0 のビット <7:2> を比較する
00111 = EID<17:11> とデータバイト 0 のビット <7:1> を比較する
01000 = EID<17:10> とデータバイト 0 のビット <7:0> を比較する
01001 = EID<17:9> と、データバイト 0 のビット <7:0> およびデータバイト 1 のビット <7> を比較する
01010 = EID<17:8> と、データバイト 0 のビット <7:0> およびデータバイト 1 のビット <7:6> を比較する
01011 = EID<17:7> と、データバイト 0 のビット <7:0> およびデータバイト 1 のビット <7:5> を比較する
01100 = EID<17:6> と、データバイト 0 のビット <7:0> およびデータバイト 1 のビット <7:4> を比較する
01101 = EID<17:5> と、データバイト 0 のビット <7:0> およびデータバイト 1 のビット <7:3> を比較する
01110 = EID<17:4> と、データバイト 0 のビット <7:0> およびデータバイト 1 のビット <7:2> を比較する
01111 = EID<17:3> と、データバイト 0 のビット <7:0> およびデータバイト 1 のビット <7:1> を比較する
10000 = EID<17:2> と、データバイト 0 のビット <7:0> およびデータバイト 1 のビット <7:0> を比較する
10001 = EID<17:1> と、バイト 0 のビット <7:0>、バイト 1 のビット <7:0>、バイト 2 のビット <7> を比較する
10010 = EID<17:0> と、バイト 0 のビット <7:0>、バイト 1 のビット <7:0>、バイト 2 のビット <7:6> を比較する
10011-11111 = 選択は無効である

レジスタ 21-25: CiTRmnCON: ECAN™ TX/RX バッファ m 制御レジスタ (m = 0,2,4,6; n = 1,3,5,7)

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXENn	TXABTn	TXLARBn	TXERRn	TXREQn	RTRENn	TXnPRI<1:0>	
bit 15							bit 8

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXENm	TXABTm(1)	TXLARBm(1)	TXERRm(1)	TXREQm	RTRENm	TXmPRI<1:0>	
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-8　bit 7-0 の定義を参照、制御バッファ n

bit 7　**TXENm:** TX/RX バッファ選択ビット
1 = バッファ TRBn は送信バッファである
0 = バッファ TRBn は受信バッファである

bit 6　**TXABTm:** メッセージ中止ビット (1)
1 = メッセージは中止された
0 = メッセージは正常に送信を完了した

bit 5　**TXLARBm:** メッセージのアービトレーション ロストビット (1)
1 = 送信中にメッセージのアービトレーション ロストが発生した
0 = 送信中にメッセージのアービトレーション ロストは発生していない

bit 4　**TXERRm:** 送信中のエラー検出ビット (1)
1 = メッセージ送信中にバスエラーが発生した
0 = メッセージ送信中にバスエラーは発生していない

bit 3　**TXREQm:** メッセージ送信要求ビット
1 = メッセージの送信を要求する。ビットはメッセージが正常に送信されると自動的にクリアされる
0 = セットされているときにビットを「0」にクリアすると、メッセージの中止が要求される

bit 2　**RTRENm:** 自動リモート送信イネーブルビット
1 = リモート送信を受信すると、TXREQ がセットされる
0 = リモート送信を受信しても、TXREQ は影響を受けない

bit 1-0　**TXmPRI<1:0>:** メッセージ送信優先度ビット
11 = 最も高いメッセージ優先度
10 = 2 番目に高いメッセージ優先度
01 = 3 番目に高いメッセージ優先度
00 = 最も低いメッセージ優先度

Note 1: このビットは TXREQ がセットされるとクリアされます。

レジスタ 21-26: CiEC: ECAN™ 送信 / 受信エラーカウント レジスタ

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
TERRCNT<7:0>							
bit 15							bit 8

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
RERRCNT<7:0>							
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-8 **TERRCNT<7:0>:** 送信エラーカウント ビット

bit 7-0 **RERRCNT<7:0>:** 受信エラーカウント ビット

レジスタ 21-27: CiRXD: ECAN™ 受信データ レジスタ

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
受信データワード							
bit 15							bit 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
受信データワード							
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **DATA<15:0>:** 受信データ ビット

レジスタ 21-28: CiTXD: ECAN™ 送信データ レジスタ

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
送信データワード							
bit 15							bit 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
送信データワード							
bit 7							bit 0

凡例 :	C = 書き込み可能ビット、ただしビットをクリアする「0」の書き込みのみ可	
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-0 **DATA<15:0>:** 送信データ ビット

21.4 ECAN メッセージ バッファ

ECAN メッセージ バッファは DMA RAM 内にあります。これらは ECAN 特殊機能レジスタではありません。ユーザ アプリケーションは、ECAN メッセージ バッファに設定されている DMA RAM 領域へ直接書き込む必要があります。バッファ領域の位置とサイズは、ユーザ アプリケーションで定義します。

本セクションでは、メッセージ バッファ ワードを送受信用に構成する方法を説明します。(メッセージ バッファ レイアウトの詳細は **21.2「CAN メッセージ フォーマット」**を、DMA RAM 内の ECAN メッセージ バッファの設定方法に関する詳細は **21.8「DMA コントローラの設定」**を参照してください。)

バッファ 21-1: ECAN メッセージ バッファ ワード 0

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	SID10	SID9	SID8	SID7	SID6
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID5	SID4	SID3	SID2	SID1	SID0	SRR	IDE
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-13 **未実装 :** 「0」として読み出し

bit 12-2 **SID<10:0>:** 標準識別子ビット

bit 1 **SRR:** 代替リモート要求ビット
TXIDE = 0 の場合 :
1 = メッセージはリモート送信を要求する
0 = 通常メッセージである
TXIDE = 1 の場合 :
SRR ビットは「1」にセットする必要がある

bit 0 **IDE:** 拡張識別子ビット
1 = メッセージは拡張識別子を送信する
0 = メッセージは標準識別子を送信する

バッファ 21-2: ECAN メッセージ バッファ ワード 1

U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	—	EID17	EID16	EID15	EID14
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID13	EID12	EID11	EID10	EID9	EID8	EID7	EID6
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-12 **未実装 :** 「0」として読み出し

bit 11-0 **EID<17:6>:** 拡張識別子ビット

バッファ 21-3: ECAN™ メッセージ バッファ ワード 2

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID5	EID4	EID3	EID2	EID1	EID0	RTR	RB1
bit 15							bit 8

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	RB0	DLC3	DLC2	DLC1	DLC0
bit 7							bit 0

凡例 :		
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-10 **EID<5:0>:** 拡張識別子ビット

bit 9 **RTR:** リモート送信要求ビット
TXIDE = 1 の場合 :
1 = メッセージはリモート送信を要求する
0 = 通常メッセージである
TXIDE = 0 の場合 :
RTR ビットは無視される

bit 8 **RB1:** 予約済みビット 1
ユーザは CAN プロトコルごとにこのビットを「0」にセットする必要がある

bit 7-5 **未実装 :** 「0」として読み出し

bit 4 **RB0:** 予約済みビット 0
ユーザは CAN プロトコルごとにこのビットを「0」にセットする必要がある

bit 3-0 **DLC<3:0>:** データ長コードビット

バッファ 21-4: ECAN™ メッセージ バッファ ワード 3

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte 1							
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte 0							
bit 7							bit 0

凡例 :		
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア　　x = ビットは未知

bit 15-8 ECAN メッセージ Byte 1

bit 7-0 ECAN メッセージ Byte 0

バッファ 21-5: ECAN™ メッセージ バッファ ワード 4

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte 3							
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte 2							
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-8 ECAN メッセージ Byte 3
bit 7-0 ECAN メッセージ Byte 2

バッファ 21-6: ECAN™ メッセージ バッファ ワード 5

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte 5							
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte 4							
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-8 ECAN メッセージ Byte 5
bit 7-0 ECAN メッセージ Byte 4

バッファ 21-7: ECAN™ メッセージ バッファ ワード 6

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte 7							
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
Byte 6							
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-8 ECAN メッセージ Byte 7
bit 7-0 ECAN メッセージ Byte 6

バッファ 21-8: ECAN™ メッセージ バッファ ワード 7

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 7							bit 0

凡例 :			
R = 読み出し可能ビット	W = 書き込み可能ビット	U = 未実装ビット、「0」として読み出し	
-n = POR 時の値	「1」= ビットをセット	「0」= ビットをクリア	x = ビットは未知

bit 15-13 **未実装 :** 「0」として読み出し

bit 12-8 **FILHIT<4:0>:** フィルター致コードビット
このバッファの書き込みを引き起こしたフィルタ番号をエンコードする (受信バッファでのみモジュールによって書き込まれ、送信バッファでは使用しない)

bit 7-0 **未実装 :** 「0」として読み出し

21.5 ECAN 動作モード

ECAN モジュールは複数のモードのいずれかで動作し、ユーザ アプリケーションからモードを選択する事ができます。以下のモードがあります。

- コンフィグレーション モード
- 通常動作モード
- リッスンオンリー モード
- リッスンオールメッセージ モード
- ループバック モード
- ディセーブル モード

動作モードは、ECAN 制御レジスタ 1 (CiCTRL1<10:8>) の要求動作モード (REQOP<2:0>) ビットにユーザ アプリケーションが書き込む事によって要求されます。ECAN は動作モード (OPMODE<2:0>) ビット (CiCTRL1<7:5>) によって要求されたモードへの移行を肯定応答します。モードの遷移は、CAN ネットワークと同期して実行されます。つまり、ECAN コントローラはバスアイドル シーケンス (11 ビットのリセッシブビット) を検出するまで待機し、検出後にモードを切り換えます。

21.5.1 コンフィグレーション モード

ハードウェアのリセット後、ECAN モジュールはコンフィグレーション モードになります (OPMODE<2:0> = 100)。エラーカウンタはクリアされ、全レジスタにリセット値が格納されます。ECAN ビット時間制御レジスタ (CiCFG1 と CiCFG2) を変更するには、ECAN モジュールをコンフィグレーション モードにする必要があります。

21.5.2 通常動作モード

通常動作モードでは、ECAN モジュールは CAN メッセージの送受信が可能です。通常動作モードは、REQOP<2:0> ビット (CiCTRL1<10:8>) を「000」にプログラミングして初期化した後に要求します。OPMODE<2:0> = 000 とすると、モジュールは通常動作を継続します。

21.5.3 リッスンオンリー モード

リッスンオンリー モードは、実際に送信プロセスには関与せずにバスを監視する目的で主に使用します。リッスンオンリー モードのノードは、肯定応答またはエラー フレームを生成しません。フレームの生成は他のいずれかのノードが行う必要があります。リッスンオンリー モードを使用すると、CAN バス上の baud レートを検出できます。

21.5.4 リッスンオールメッセージ モード

リッスンオールメッセージ モードは、システムのデバッグに使用します。基本的に、エラー発生時でも識別子にかかわらず全メッセージを受信します。リッスンオールメッセージ モードが有効な場合、メッセージ受信でエラーが発生した場合でもメッセージ バッファに転送される事以外は、通常動作モードと同様に送受信が動作します。

21.5.5 ループバック モード

ループバック モードはセルフテストで使用し、ECAN モジュールが自身のメッセージを受信できます。このモードでは、ECAN 送信パスは内部的に受信パスに接続されます。ダミーの肯定応答が送信されるため、別のノードが肯定応答ビットを送信する必要はありません。

21.5.6 ディセーブル モード

ディセーブル モードは、デバイスをスリープモードまたはアイドルモードに移行させる前に安全なシャットダウンを保証するために使用します。つまり、ECAN コントローラはバスアイドル シーケンス (11 ビットのリセッシブビット) を検出するまで待機し、検出後にモードを切り換えます。モジュールはディセーブル モードに移行すると、CPU または他のモジュールに影響を与えずに固有のクロックを停止します。バス アクティビティが発生するか、CPU が OPMODE<2:0> を「000」にセットすると、モジュールはウェイクアップします。

モジュールがディセーブル モード状態の間、CiTX ピンはリセッシブ状態を維持します。

21.6 ECAN メッセージの送信

メッセージを生成するノードがそのメッセージのトランスミッタです。このノードは、バスがアイドル状態になるか、そのユニットにアービトレーション ロストが発生するまではトランスミッタのままです。図 21-8 に標準的な ECAN 送信プロセスを示します。

メッセージ バッファ 0 ～ 7 (DMA RAM に配置) は、対応する ECAN TX/RX バッファ m 制御レジスタ (CiTRmnCON<7>) 内の TX/RX バッファ選択 (TXENn) ビットを使用して、CAN メッセージを送受信するように設定されています。TXENn ビットがセットされている場合、メッセージ バッファは送信用に設定されます。メッセージ バッファの標準および拡張フレームのレイアウトと、CAN プロトコルによる標準データ、拡張データ、標準リモート、拡張リモート フレームの IDE、SRR、RTR、RB0、RB1 の各ビットの状態については、**21.2「CAN メッセージ フォーマット」**を参照してください。

21.6.1 メッセージ送信フロー

CAN バス上でメッセージを送信するには、ユーザ アプリケーションは以下のタスクを実行する必要があります。

- メッセージ バッファを送信用に設定し、バッファに優先度を割り当てる
- CAN メッセージを DMA RAM 内のメッセージ バッファに書き込む
- メッセージ送信を開始するバッファの送信要求ビットをセットする

メッセージ送信を開始するには、ECAN 送信 / 受信制御レジスタ (CiTRmnCON<3>) のメッセージ送信要求 (TXREQm) ビットをセットします。メッセージ送信後、TXREQm ビットは自動的にクリアされます。SOF の送信前に、送信準備が整っている全バッファを検査して最も高い優先度を持つバッファを判定します。最も高い優先度を持つ送信バッファが最初に送信されます。

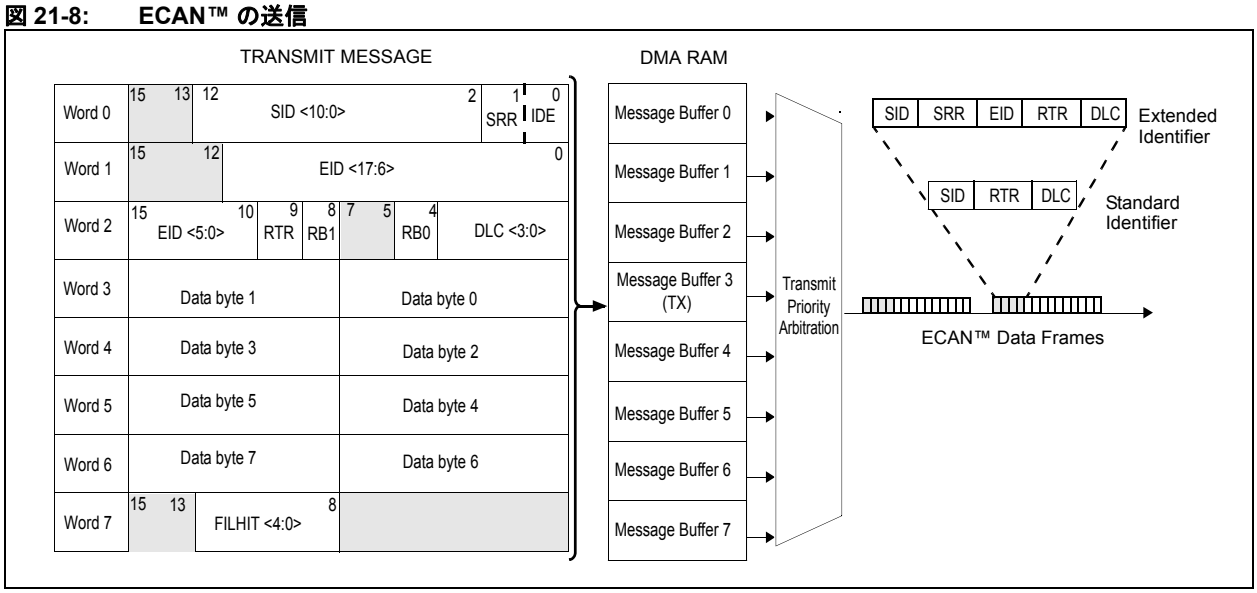
CiTRmnCON<TXnPRI> ビットを使用して、ユーザ アプリケーション定義の 4 つの優先度レベルのいずれかを各送信メッセージ バッファに割り当てる事が可能です。

TXnPRI<1:0> メッセージ送信優先度は以下のいずれかを選択できます。

- 11 = 最も高い優先度を持つ送信メッセージ
- 10 = 2 番目に高い優先度を持つ送信メッセージ
- 01 = 3 番目に高い優先度を持つ送信メッセージ
- 00 = 最も低い優先度を持つ送信メッセージ

ユーザ アプリケーション定義の優先度が同一レベルのメッセージ バッファには、自然順序優先度を適用します。自然順序優先度が最も高いのはメッセージ バッファ 7 です。ユーザ アプリケーション定義の優先度は、自然順序優先度よりも優先されます。

図 21-8: ECAN™ の送信



例 21-1 に、メッセージ バッファ 0 を使用して標準フレームを送信するコードを示します。

例 21-1:　　標準データ フレーム送信用のサンプルコード

```
/* Assign 32x8word Message Buffers for ECAN1 in DMA RAM */

unsigned int ecan1MsgBuf[32][8] __attribute__(space(dma));
DMA1STA = __builtin_dmaoffset(ecan1MsgBuf);

/* Configure Message Buffer 2 for Transmission and assign priority*/

C1TR01CONbits.TXEN2 = 0x1;
C1TR01CONbits.TX2PRI = 0x3;

/* WRITE TO MESSAGE BUFFER 0 */
/* CiTRBnSID = 0bxxx1 0010 0011 110
IDE = 0b0
SRR = 0b0
SID<10:0>= 0b100 1000 1111 */

ecan1MsgBuf[0][0] = 0x123C;

/* CiTRBnEID = 0bxxxx 0000 0000 0000
EID<17:6> = 0b0000 0000 0000 */

ecan1MsgBuf[0][1] = 0x0000;

/* CiTRBnDLC = 0b0000 0000 xxx0 1111
EID<17:6> = 0b000000
RTR = 0b0
RB1 = 0b0
RB0 = 0b0
DLC = 0b1000 */

ecan1MsgBuf[0][2] = 0x8;

/* WRITE MESSAGE DATA BYTES */

ecan1MsgBuf[0][3] = 0xabcd;
ecan1MsgBuf[0][4] = 0xabcd;
ecan1MsgBuf[0][5] = 0xabcd;
ecan1MsgBuf[0][6] = 0xabcd;

/* REQUEST MESSAGE BUFFER 0 TRANSMISSION */

C1TR01CONbits.TXREQ0 = 0x1;
```

例 21-2 に、メッセージ バッファ 2 を使用して拡張フレームを送信するコードを示します。

例 21-2:　拡張データフレーム送信用のサンプルコード

```
/* Assign 32x8word Message Buffers for ECAN1 in DMA RAM */
unsigned int ecan1MsgBuf[32][8] __attribute__(space(dma));
DMA1STA = __builtin_dmaoffset(ecan1MsgBuf);

/* Configure Message Buffer 2for Transmission and assign priority*/

C1TR23CONbits.TXEN2 = 0x1;
C1TR23CONbits.TX2PRI = 0x2;

/* WRITE TO MESSAGE BUFFER 2 */
/* CiTRBnSID = 0bxxx1 0010 0011 1101
IDE = 0b1
SRR = 0b0
SID<10:0> : 0b100 1000 1111 */

ecan1MsgBuf[2][0] = 0x123D;

/* CiTRBnEID = 0bxxxx 1111 0000 0000
EID<17:6> = 0b1111 0000 0000 */

ecan1MsgBuf[2][1] = 0x0F00;

/* CiTRBnDLC = 0b0000 1100 xxx0 1111
EID<17:6> = 0b000011
RTR = 0b0
RB1 = 0b0
RB0 = 0b0
DLC = 0b1000 */

ecan1MsgBuf[2][2] = 0x0C08;

/* WRITE MESSAGE DATA BYTES */

ecan1MsgBuf[2][3] = 0xabcd;
ecan1MsgBuf[2][4] = 0xabcd;
ecan1MsgBuf[2][5] = 0xabcd;
ecan1MsgBuf[2][6] = 0xabcd;

/* REQUEST MESSAGE BUFFER 2 TRANSMISSION */

C1TR23CONbits.TXREQ2 = 0x1;
```

21.6.2　メッセージ送信の中止

ECAN 制御レジスタ 1 (CiCTRL1<12>) の保留中送信の一括中止 (ABAT) ビットをセットすると、保留中の全メッセージを中止するよう要求します。特定のメッセージを中止するには、そのメッセージ バッファに関連するメッセージ送信要求 (TXREQm) ビット (CiTRmnCON<3>) をクリアする必要があります。いずれの場合も、ECAN モジュールがバス上でメッセージ送信を開始していない場合に限りメッセージ送信は中止されます。

21.6.3 リモート送信要求 / 応答

21.6.3.1 リモート送信要求

特定の識別子値を持つデータフレームの受信を期待するノードは、リモートフレームを送信する事によって、別のノードからの所定のデータの転送を開始できます。リモートフレームは、標準フォーマットまたは拡張フォーマットのいずれかの形式です。

リモートフレームはデータフレームに類似していますが、以下の点が異なります。

- RTR ビットはリセッシブ (RTR = 1)
- データフィールドなし (DLC = 0)

リモートフレームを送信するには、ユーザ アプリケーションは以下のタスクを実行する必要があります。

- メッセージ バッファを送信用に設定し、バッファに優先度を割り当てる。
- リモートフレームを適切なメッセージ バッファに書き込む。送信識別子は、受信するデータフレームの識別子と同じである必要がある。
- リモートフレームの送信を開始するバッファの送信要求ビットをセットする。

21.6.3.2 リモート送信応答

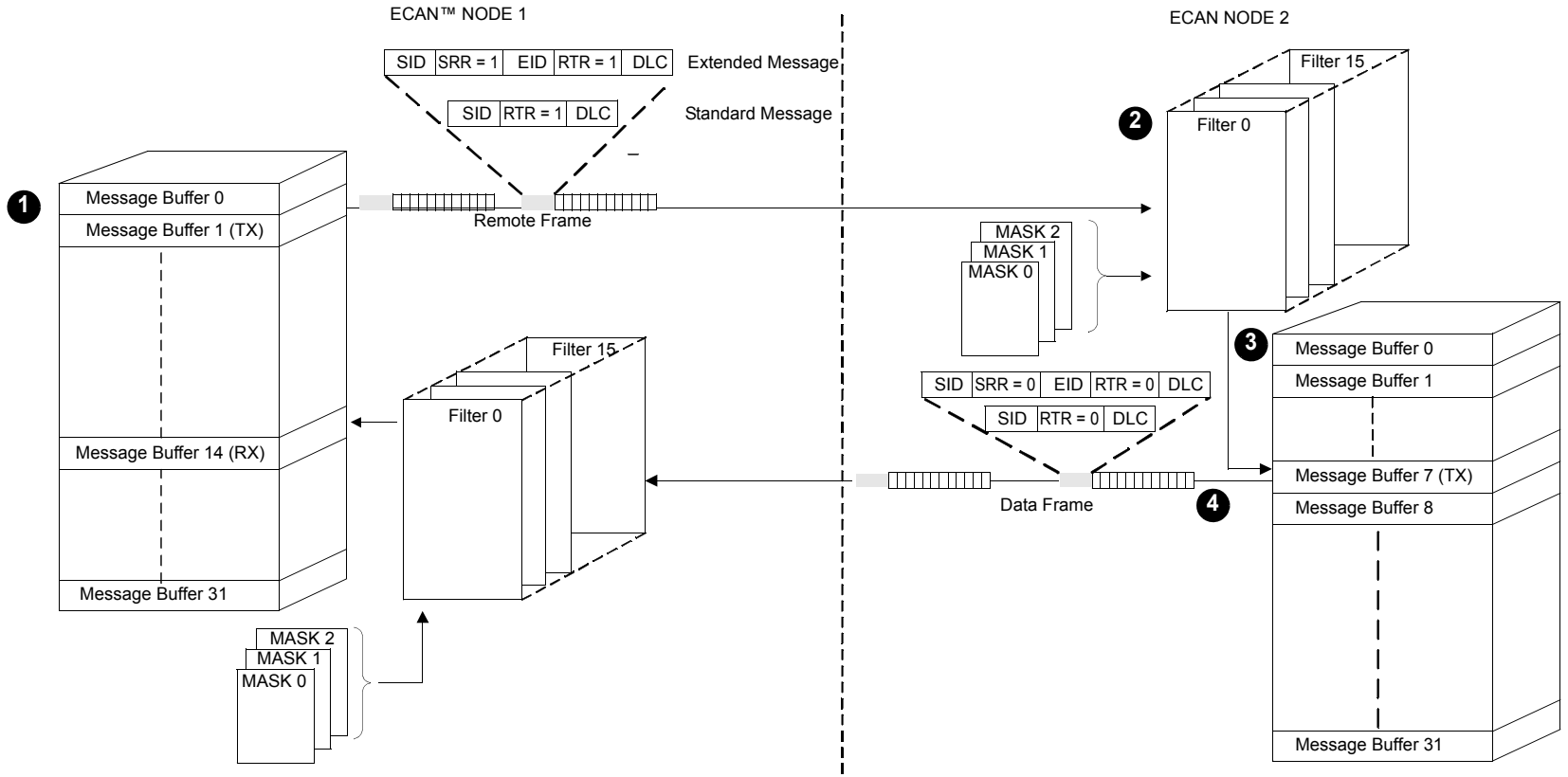
リモートフレーム要求に応答する送信元として機能するノードは、リモート要求フレームの識別子に一致するようアクセプタンス フィルタを設定する必要があります。メッセージ バッファ 0 ～ 7 はリモート要求に応答可能です。従って、アクセプタンス フィルタ バッファポインタ (FnBP) は 8 つのメッセージ バッファのいずれかを指す必要があります。ECAN 送信 / 受信制御レジスタ (CiTRmnCON) の TX/RX バッファ選択 (TXENn) ビットと自動リモート送信イネーブル (RTRENn) ビットは、リモート要求フレームに応答するようにセットする必要があります。

これは、アクセプタンス フィルタ バッファポインタ (FnBP) が送信用に設定 (TXENn = 1) されているメッセージ バッファを指す場合にのみ該当します。

図 21-9 にリモート フレームの処理プロセスを示します。

1. ECAN ノード 1 はリモート送信要求を送信する (メッセージ バッファ 1 を使用する)。
2. ECAN ノード 2 は要求を受信し、データフレームを送信する事によって応答する (メッセージ バッファ 7 を使用する)。
3. ECAN ノード 1 がデータフレームを受信する。
4. データフレームは ECAN ノード 1 のメッセージ バッファ 14 に格納される。

図 21-9: リモート送信要求 / 応答



1. リモートフレームを送信するノードは、リモートフレームを送信するための送信バッファと、データフレームを受信するための受信バッファ 1 つを備えている必要がある。
2. リモートフレームを受信するノードは、受信リモートフレームに応答するデータフレームを送信するための送信バッファを備えている必要がある。
3. CiBUFPNTm<FnBP> は、リモート送信に備えていずれかの送信バッファを指している必要がある。
4. リモート送信を受信した時に CiTRmnCON<TXREQ> ビットが自動的にセットされるように、CiTRmnCON<RTREN> をセットしておく必要がある。

例 21-3 に、メッセージ バッファ 2 を使用して拡張リモートフレームを送信するために必要なコードを示します。

例 21-3:　拡張リモートフレーム送信用のサンプルコード

```
/* Assign 32x8word Message Buffers for ECAN1 in DMA RAM */

unsigned int ecan1MsgBuf[32][8] __attribute__(space(dma));
DMA1STA = __builtin_dmaoffset(ecan1MsgBuf);

/* Configure Message Buffer 0 for Transmission and assign priority*/

C1TR23CONbits.TXEN0 = 0x1;
C1TR23CONbits.TX0PRI = 0x2;

/* WRITE TO MESSAGE BUFFER 0 */
/* CiTRBnSID = 0bxxx1 0010 0011 1111
IDE = 0b1
SRR = 0b1
SID<10:0> : 0b100 1000 1111 */

ecan1MsgBuf[2][0] = 0x123F;

/* CiTRBnEID = 0bxxxx 1111 0000 0000
EID<17:6> = 0b1111 0000 0000 */

ecan1MsgBuf[2][1] = 0x0F00;

/* CiTRBnDLC = 0b0000 1110 xxx0 1111
EID<17:6> = 0b000011
RTR = 0b1
RB1 = 0b0
RB0 = 0b0
DLC = 0b0 */

ecan1MsgBuf[2][2] = 0x0E00;

/* THERE ARE NO DATA BYTES FOR A REMOTE MESSAGE */
/* REQUEST MESSAGE BUFFER 2 TRANSMISSION */

C1TR23CONbits.TXREQ2 = 0x1;
```

21.7 ECAN メッセージの受信

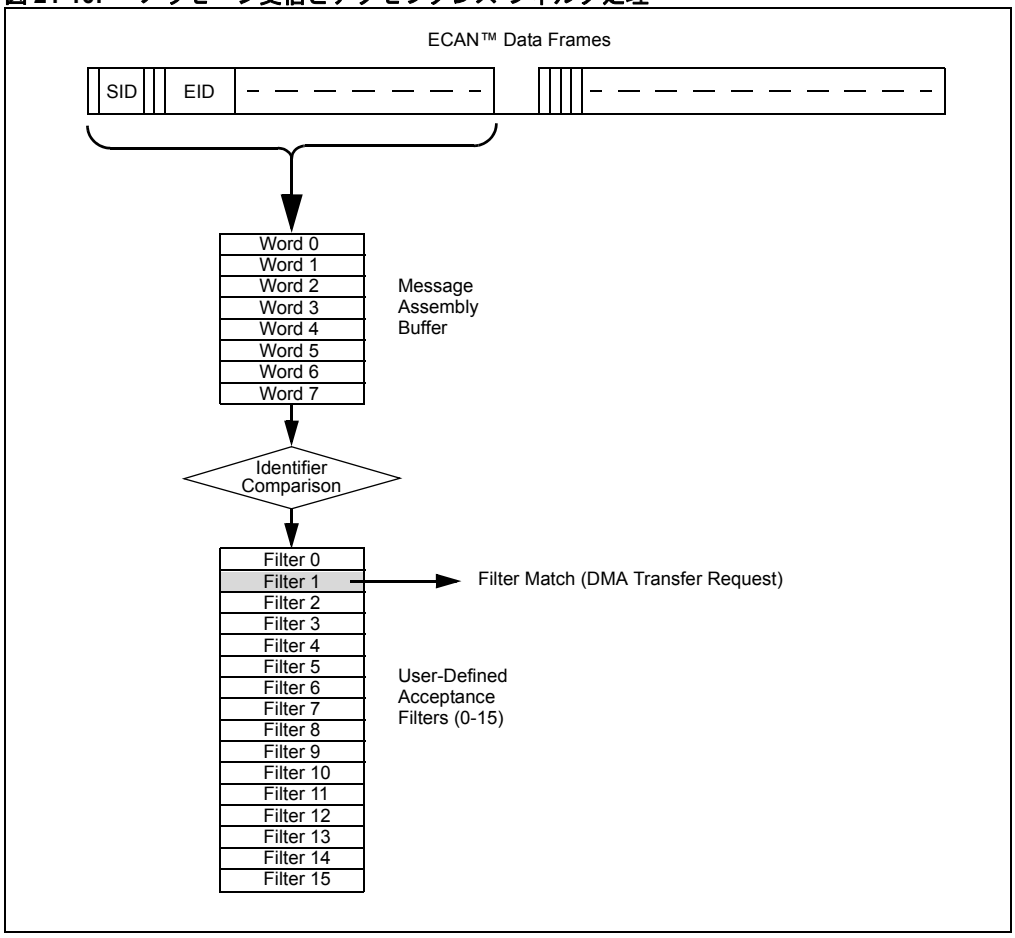
ECAN モジュールは CAN バスノード上で標準フレームと拡張フレームのいずれも受信できます。さらに、モジュールは受信したメッセージを DMA RAM 内のユーザ定義バッファに自動的に転送する追加機能を備えている事から、ユーザ アプリケーションがハードウェア レジスタからユーザ定義バッファにメッセージを明示的にコピーする必要がありません。DMA バッファ内に各メッセージを格納するフォーマットは、送信バッファのフォーマットと同じで、各メッセージ (関連するステータス レジスタを含む) が DMA RAM 内の 8 ワードを占有します。

ECAN 受信プロセスを構成する 2 つの主要な段階は、後続のセクションを参照してください。図 21-10 と図 21-13 に受信プロセスを簡略化した例を示します。

21.7.1 メッセージ受信とアクセプタンス フィルタ処理

図 21-10 に示すように、バス上の全ての受信メッセージはメッセージ アセンブリ バッファで受信し、その識別子フィールドを 16 のユーザ定義アクセプタンス フィルタのセットと比較します。受信した各標準データフレームには 11 ビットの標準識別子 (SID) があり、各拡張データフレームには 11 ビットの SID と 18 ビットの拡張識別子 (EID) があります。受信している識別子の全ビットがいずれかのアクセプタンス フィルタの対応するビットに完全に一致する場合、そのメッセージを DMA RAM 内の適切なバッファで受信できるように、ECAN モジュールは DMA コントローラに対して DMA 転送要求を生成します。

図 21-10: メッセージ受信とアクセプタンス フィルタ処理



21.7.1.1 アクセプタンス フィルタ

図 21-11 に、標準フレームのフィルタ / マスクビットと比較した受信メッセージ識別子を示します。図 21-12 に、拡張フレームのフィルタ / マスクビットと比較した受信メッセージ識別子を示します。

図 21-11: 標準メッセージ用のアクセプタンスフィルタ

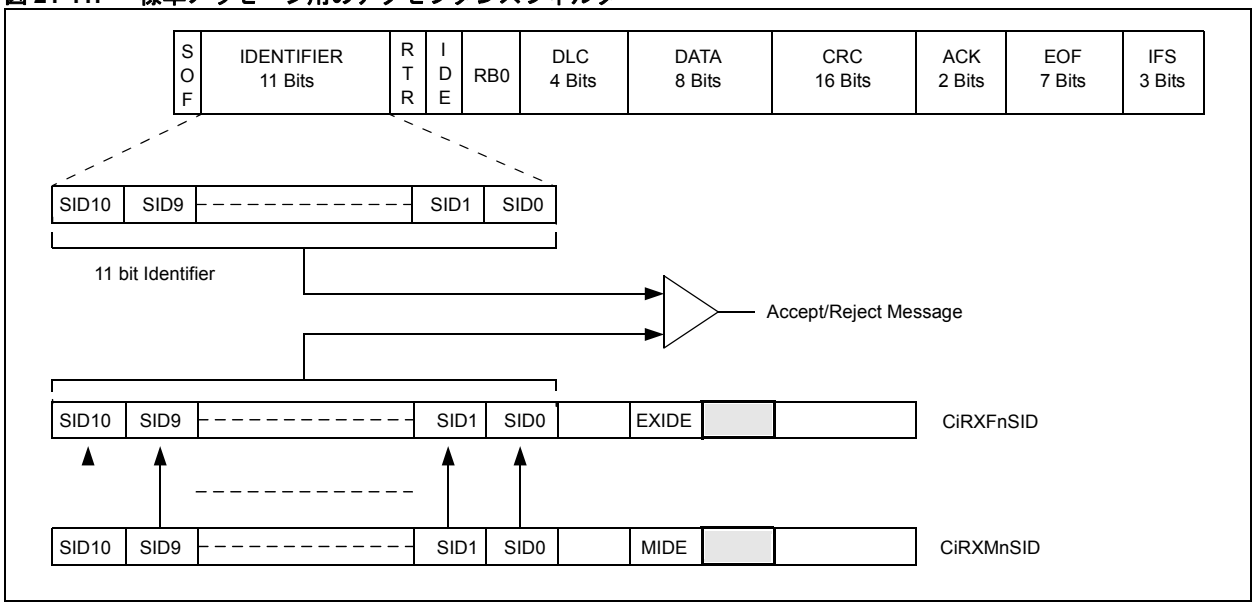
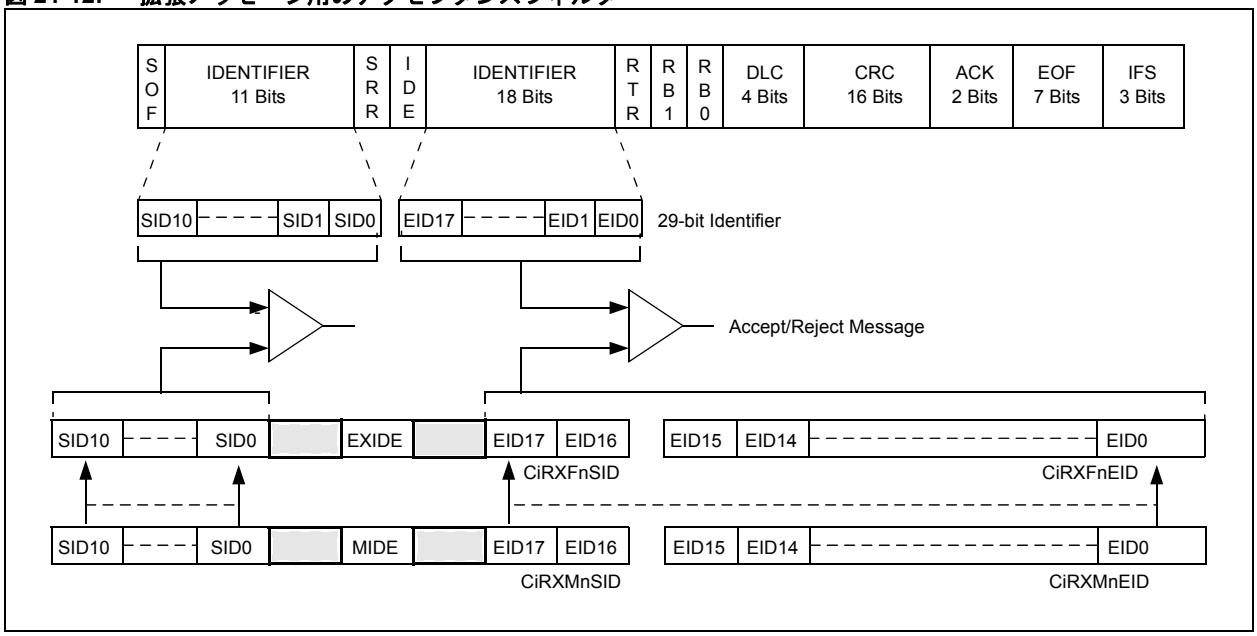


図 21-12: 拡張メッセージ用のアクセプタンスフィルタ



ECAN アクセプタンス フィルタ イネーブル (CiFEN1<15:0>) レジスタのフィルタ イネーブル (FLTENn) ビットを使用して、アクセプタンス フィルタ 0 ～ 15 を個別に有効化または無効化できます。「n」の値はレジスタビットを表し、アクセプタンス フィルタのインデックスに対応します。

アクセプタンス フィルタ 0 ～ 15 は、受信バッファに内容を渡すために受信メッセージが格納している必要がある識別子を指定します。これらのフィルタはそれぞれ、標準識別子用と拡張識別子用の 2 つのレジスタで構成されています。これらのレジスタは以下のように識別されます。

- **CiRXFnSID:** ECAN アクセプタンス フィルタ n 標準識別子レジスタ (n = 0 ～ 15)
- **CiRXFnEID:** ECAN アクセプタンス フィルタ n 拡張識別子レジスタ (n = 0 ～ 15)

21.7.1.2 アクセプタンス フィルタ マスク

図 21-11 と図 21-12 に示すように、アクセプタンス フィルタ マスクは、アクセプタンス フィルタで検査する受信メッセージ識別子のビットを決定します。

アクセプタンス フィルタは、CiFMSKSEL1 と CiFMSKSEL2 レジスタのマスクソース選択 (FnMSK<1:0>) マスク選択ビットを使用して、アクセプタンス フィルタ マスクのいずれかを必要に応じて選択します

- **CiFMSKSEL1<FnMSK>:** フィルタ 0 ～ 7 のマスク選択
- **CiFMSKSEL2<FnMSK>:** フィルタ 8 ～ 15 のマスク選択

表 21-1 に FnMSK<1:0> ビットの選択値を示します。

表 21-1: FnMSK<1:0> 選択と値

値	選択
00	アクセプタンス フィルタ マスク 0 を選択
01	アクセプタンス フィルタ マスク 1 を選択
10	アクセプタンス フィルタ マスク 2 を選択
11	予約済み

表 21-2 は真理値表で、メッセージの受け入れまたは拒否を判定するために識別子の各ビットとマスクおよびフィルタをどのように比較するかを示します。マスクビットは実際にどのビットにフィルタを適用するかを決定します。マスクビットがゼロにセットされている場合、そのビットはフィルタビットに関係なく自動的に受け入れられます。

表 21-2: アクセプタンス フィルタ / マスクの真理値表

マスク (SIDn/EIDn)	フィルタ (SIDn/EIDn)	メッセージ (SIDn/EIDn)	bit n を受け入れまたは拒否
0	x	x	受け入れ
1	0	0	受け入れ
1	0	1	拒否
1	1	0	拒否
1	1	1	受け入れ

21.7.1.3 メッセージ タイプの選択

ECAN アクセプタンス フィルタ n 標準識別子 (CiRXFnSID<3>) レジスタの拡張識別子イネーブル (EXIDE) ビットは、標準識別子または拡張識別子のメッセージの受信を有効にします。ECAN アクセプタンス フィルタ マスク n 標準識別子 (CiRXMnSID<3>) レジスタの識別子受信モード (MIDE) ビットは、CiRXFnSID<EXIDE> ビットを有効にします。CiRXMnSID<MIDE> ビットがセットされている場合、CiRXFnSID<EXIDE> ビットによって選択されるメッセージ タイプのみが受け入れられます。CiRXMnSID<MIDE> ビットがクリアされている場合、CiRXFnSID<EXIDE> ビットは無視され、フィルタに一致する全てのメッセージが受け入れられます。

表 21-3 にビットの設定と選択結果を示します。

表 21-3: EXIDE と MIDE の選択

EXIDE	MIDE	選択
0	1	アクセプタンス フィルタは標準識別子を検査する
1	1	アクセプタンス フィルタは拡張識別子を検査する
x	0	アクセプタンス フィルタは標準 / 拡張識別子を検査する

21.7.1.4 アクセプタンス フィルタのコンフィグレーション

例 21-4 に、SID<2:0> ビットをマスクするアクセプタンス フィルタ マスク レジスタを使用して、標準識別子メッセージを受信するようアクセプタンス フィルタ 0 を設定するコードを示します。

例 21-4: 標準データフレームをフィルタ処理するためのサンプルコード

```
/* Enable Window to Access Acceptance Filter Registers */
C1CTRL1bits.WIN=0x1;

/* Select Acceptance Filter Mask 0 for Acceptance Filter 0 */
C1FMSKSEL1bits.F0MSK=0x0;

/* Configure Acceptance Filter Mask 0 register to mask SID<2:0>
Mask Bits (11-bits) : 0b000 0000 0111 */
C1RXM0SIDbits.SID = 0x0007;

/* Configure Acceptance Filter 0 to match standard identifier
Filter Bits (11-bits): 0b011 1010 xxx */
C1RXF0SIDbits.SID = 0x01D0;

/* Acceptance Filter 0 to check for Standard Identifier */
C1RXM0SIDbits.MIDE = 0x1;
C1RXF0SIDbits.EXIDE= 0x0;

/* Acceptance Filter 0 to use Message Buffer 10 to store message */
C1BUFPNT1bits.F0BP = 0xA;

/* Filter 0 enabled for Identifier match with incoming message */
C1FEN1bits.FLTEN0=0x1;

/* Clear Window Bit to Access ECAN Control Registers */
C1CTRL1bits.WIN=0x0;
```

例 21-5 に、EID<5:0> ビットをマスクするアクセプタンス フィルタ マスク レジスタを使用して、拡張識別子メッセージを受信するようアクセプタンス フィルタ 2 を設定するコードを示します。

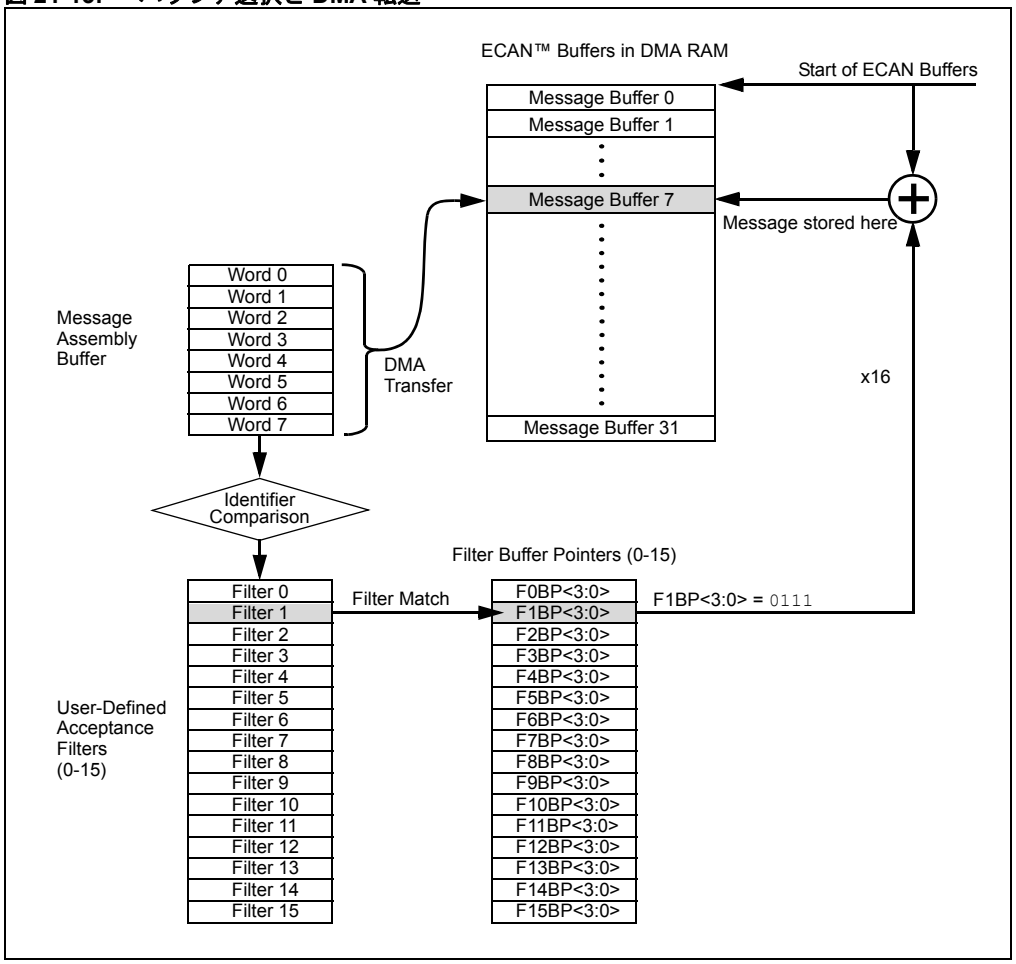
例 21-5: 拡張データ フレームをフィルタ処理するためのサンプルコード

```
/* Select Acceptance Filter Mask 1 for Acceptance Filter 2 */

C1FMSKSEL1bits.F2MSK=0x1;

/* Configure Acceptance Filter Mask 1 register to mask EID<5:0>
Mask Bits (29-bits) : 0b0 0000 0000 0000 0000 0000 0011 1111
SID<10:0> : 0b00000000000 ..SID<10:0> or EID<28:18>
EID<17:16> : 0b00 ..EID<17:16>
EID<15:0> : 0b0000000000111111 ..EID<15:0> */

C1RXM1SIDbits.SID = 0x0;
C1RXM1SIDbits.EID = 0x0;
C1RXM1EIDbits.EID = 0x3F;

/* Configure Acceptance Filter 2 to match extended identifier
Filter Bits (29-bits) : 0b1 1110 0000 0011 1111 0101 10xx xxxx
SID<10:0> : 0b11110000000 ..SID<10:0> or EID<28:18>
EID<17:16> : 0b11 ..EID<17:16>
EID<15:0> : 0b1111010110xxxxxx ..EID<15:0> */

C1RXF2SIDbits.SID = 0x780;
C1RXF2SIDbits.EID = 0x3;
C1RXM2EIDbits.EID = 0xF680;

/* Acceptance Filter 2 to check for Extended Identifier */

C1RXM1SIDbits.MIDE = 0x1;
C1RXF2SIDbits.EXIDE= 0x1;

/* Acceptance Filter 2 to use Message Buffer 5 to store message */

C1BUFPNT1bits.F2BP = 0x6;

/* Enable Acceptance Filter 2 */

C1FEN1bits.FLTEN1=0x1;
```

21.7.2 バッファ選択と DMA 転送

図 21-13 に示すように、フィルタが一致した場合、ECAN モジュールによって DMA コントローラに対する DMA 転送要求が生成され、ユーザ定義の DMA RAM 領域内の適切なメッセージ バッファに受信メッセージが自動的にコピーされます。ECAN モジュールは、最大で 32 のメッセージ バッファをサポートします。ユーザ アプリケーションは、ECAN FIFO 制御 (CiFCTRL<15:13>) レジスタの DMA バッファサイズ (DMABS<2:0>) ビットを使用して、4、6、8、12、16、24、32 のいずれかの数のメッセージ バッファを選択できます。受信バッファ インデックスの選択 (すなわち DMA コントローラによってメッセージが書き込まれている DMA RAM アドレス) は、受信している識別子に一致するフィルタによって決まり、ユーザ設定可能です。DMA コントローラはデータを DMA RAM 領域内の適切なアドレスに移動し、ユーザが指定した数のワードが転送された後に DMA 割り込みを生成します。ECAN データ転送用の DMA チャンネル コンフィグレーションの詳細は、**21.8「DMA コントローラの設定」**を参照してください。

図 21-13: バッファ選択と DMA 転送



21.7.2.1 バッファの選択

アクセプタンス フィルタ 0 ～ 15 に対して、受信メッセージをどのメッセージ バッファに格納するかを選択するために、以下の 4 つのアクセプタンス フィルタ バッファポインタ レジスタがあります。

- **CiBUFPNT1<FnBP>:** アクセプタンス フィルタ 0 ～ 3 用バッファポインタ
- **CiBUFPNT2<FnBP>:** アクセプタンス フィルタ 4 ～ 7 用バッファポインタ
- **CiBUFPNT3<FnBP>:** アクセプタンス フィルタ 8 ～ 11 用バッファポインタ
- **CiBUFPNT4<FnBP>:** アクセプタンス フィルタ 12 ～ 15 用バッファポインタ

アクセプタンス フィルタのいずれかで受信メッセージ識別子が一致すると、内部ロジックはバッファポインタ (FnBP<3:0>) を調べ、対応するメッセージ バッファのアドレスとしてそのポインタを使用します。アドレスは周辺モジュールによって DMA チャンネルに受け渡されます。従って、DMA チャンネルはペリフェラル間接アドレッシング モードに設定されている必要があります。

FnBP<3:0> の値は以下のように解釈します。

0000 メッセージ バッファ 0 でメッセージを受信する
0001 メッセージ バッファ 1 でメッセージを受信する
•
•
•
1110 メッセージ バッファ 14 でメッセージを受信する
1111 受信 FIFO バッファでメッセージを受信する

Note: マルチメッセージのバッファリングは、複数のアクセプタンス フィルタを同じ値に設定する事によってユーザ アプリケーションで実装可能です。この場合、受信メッセージが複数のフィルタに一致すると、ECAN モジュールはエンプティ バッファを指す最も小さい番号の一致フィルタにメッセージを割り当てます。

21.7.2.2 メッセージ バッファ 0 ～ 7 へのメッセージ受信

メッセージ バッファ 0 ～ 7 は、ECAN TX/RX バッファ m 制御レジスタ (CiTRmnCON<7>) 内の TX/RX バッファ選択 (TXENm) ビットを使用して、CAN メッセージを送信または受信するように設定できます。受信バッファとして設定されている場合 (TXENm = 0)、アクセプタンス フィルタ バッファポインタ (FnBP) は、このメッセージ バッファのいずれかを選択して受信メッセージを格納します。

メッセージ バッファが CiTRmnCON<RTRENn> ビットをセットして送信用にセットアップされており、そのメッセージ バッファを指すアクセプタンス フィルタがメッセージを検出した場合、メッセージ バッファはメッセージを格納せずに RTR を処理します。これは、アクセプタンス フィルタ バッファポインタ (FnBP) が送信用に設定 (TXENn = 1) されているメッセージ バッファを指す場合にのみ該当します。

Note: バッファが受信用に設定されている場合 (TXEN = 0)、ユーザ アプリケーションで CiTRmnCON レジスタの送信要求 (TXREQ) ビットをセットしないでください。これにより、モジュールが予期せぬ挙動をする可能性があります。

21.7.2.3 メッセージ バッファ 8 ～ 14 へのメッセージ受信

メッセージ バッファ 8 ～ 14 は、受信バッファです。アクセプタンス フィルタ バッファポインタ (FnBP) は、使用するメッセージ バッファを決定します。

21.7.2.4 メッセージ バッファ 15 ～ 31 へのメッセージ受信

メッセージ バッファ 15 ～ 31 は受信バッファで、FIFO バッファとしてのみ使用可能です。これは、アクセプタンス フィルタ バッファポインタ (FnBP<3:0>) ビットが直接アドレッシングできるのは 16 のエンティティに限られているからです。FnBP<3:0> = 1111 の場合、そのフィルタの一致結果は FIFO 内で次に使用可能なバッファ位置に書き込まれます。

21.7.2.5 受信バッファ ステータスビット

受信バッファはメッセージ バッファごとに、メッセージ バッファフル フラグ (RXFULn) とメッセージ バッファ オーバーフロー フラグ (RXOVFn) という 2 つのステータスビットを備えています。これらのステータスビットは、バッファフル ステータスとバッファ オーバーフロー ステータスのレジスタに分けられます。

- **CiRXFUL1<RXFULn>:** メッセージ バッファ 0 ～ 15 用の受信バッファフル ステータス
- **CiRXFUL2<RXFULn>:** メッセージ バッファ 16 ～ 31 用の受信バッファフル ステータス
- **CiRXOVF1<RXOVFn>:** メッセージ バッファ 0 ～ 15 用の受信バッファ オーバーフロー ステータス
- **CiRXOVF2<RXOVFn>:** メッセージ バッファ 16 ～ 31 用の受信バッファ オーバーフロー ステータス

受信メッセージがメッセージ バッファに格納されると、受信バッファフル レジスタの対応するバッファフル フラグ (RXFULn) がセットされ、受信バッファ割り込み (CiINTF<RBIF>) が生成されます。受信メッセージがフィルター致を引き起こし、一致フィルタに割り当てられているメッセージ バッファがフル (すなわち、そのバッファに関連する RXFULn ビットが既に「1」) の場合、対応する RXOVFn ビット (「n」は、そのバッファに関連するメッセージ バッファの番号) がセットされ、受信バッファ オーバーフロー割り込みが生成されます (CiINTF<RBOVIF>)。このメッセージは失われます。

Note:	複数のフィルタが受信メッセージの識別子に一致し、一致する全てのフィルタに割り当てられているメッセージ バッファが全てフルの場合、最も小さい番号の一致フィルタに対応する RXOVFn ビットがセットされます。

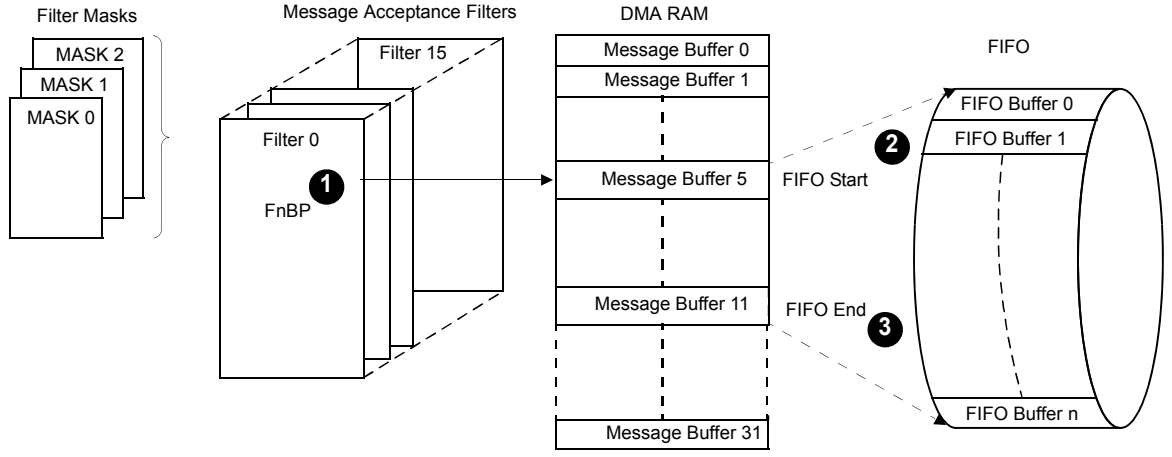
21.7.3 FIFO バッファ動作

ECAN モジュールは、最大で 32 のメッセージ バッファをサポートします。ユーザ アプリケーションは、ECAN FIFO 制御 (CiFCTRL) レジスタの DMA バッファサイズ (DMABS<2:0>) ビットを使用して、4、6、8、12、16、24、32 のいずれかの数のメッセージ バッファを指定できます。FIFO 開始領域 (FSA<4:0>) ビット (CiFCTRL<4:0>) を使用して、バッファ領域内の FIFO の開始位置を指定します。FIFO の末尾は、DMABS<2:0> ビットで定義するメッセージ バッファ数で決まります。

ユーザ アプリケーションで、送信バッファを含む FIFO 領域を割り当てないでください。割り当てた場合、モジュールは送信バッファを指そうとしますが、そのバッファでメッセージを受信すると、オーバーフロー条件によりメッセージ内容が失われます。

図 21-14 に、メッセージ アクセプタンス フィルタの 1 つが FIFO に受信メッセージを格納するように設定されている例 (FnBP = 1111) を示します。12 のメッセージ バッファを割り当て、FIFO の開始位置をメッセージ バッファ 5 (CiFCTRL<FSA> = 00101) に、FIFO の末尾をメッセージ バッファ 11 (CiFCTRL<DMABS> = 011) に設定します。

図 21-14: FIFO でのメッセージ受信



1. アクセプタンス フィルタ バッファポインタ「FnBP」は、FIFOに受信メッセージを格納するために「111」である必要がある。
2. FIFO の開始アドレスは CiFCTRL<FSA> で指定する。上図の例では、FSA <4:0> = 「00101」に設定する。
3. FIFO の終了アドレスは CiFCTRL<DMABS> で指定する。上図の例では、DMABS <2:0> = 「011」に設定する。

21.7.3.1 FIFO 領域へのメッセージ受信

アクセプタンス フィルタは FnBP<3:0> = 1111 の場合、受信メッセージを FIFO 領域に格納します。フィルタはシンプルなバッファポインタを使用して、上記で定義した FIFO の開始位置から始め、FIFO 領域内のバッファを順番にインクリメントして行きます。最後のバッファに到達すると、カウンタが戻り FIFO 領域の先頭を指します。

書き込みポインタの値には、アプリケーション ソフトウェアから CiFIFO<FBP> ビットでアクセスでき、読み込み (のみ) 可能です。メッセージがバッファに格納されると、バッファに関連する RXFUL ビットがセットされ、FIFO バッファ カウンタがインクリメントします。

FBP<5:0> 値が 1 つのバッファを指していて、フィルタが一致した時点でメッセージ内容を書き込む前にそのバッファに関連する RXFUL ビットが既に「1」である場合、そのバッファに関連する RXOVL ビットがセットされメッセージは失われます。メッセージが失われた後、FBP<5:0> 値は通常通りインクリメントします。

FBP<5:0> 値が、フィルタが一致した時点でメッセージ内容を書き込む前に送信バッファとして選択された送信 / 受信バッファを指している場合、そのバッファに関連する RXOVL ビットがセットされメッセージは失われます。メッセージが失われた後、FBP<5:0> 値は通常通りインクリメントします。

アプリケーション ソフトウェアは、バッファの内容を読み出す事によって FIFO を処理します。バッファ位置が読み込まれると、アプリケーション ソフトウェアはそのバッファに対応する RXFUL ビットをクリアします。RXFUL ビットがクリアされると、そのバッファに対応する数に 1 を加えた値がモジュールによって CiFIFO <FNRB> ビットに書き込まれます。アプリケーション ソフトウェアはこの値を読み込み (のみ) 可能で、4 ビット分左シフトして次のバッファを読み込むためのアドレス オフセットとして使用します。アプリケーション ソフトウェアはバッファを順次読み出します。

FIFO がフルに近づくと、モジュールは割り込み条件を生成します。この条件は、式 21-1 に示すように数学的に計算されます。

式 21-1: FIFO 割り込みの計算

$$FNRB - FBP = 1$$

または

$$(FNRB = START)\ AND\ (FBP = END)$$

直前に書き込まれたバッファの RXFUL ビットがセットされ、FBP ビットが更新された後、割り込みが生成されます。計算では更新された FBP 値を使用します。

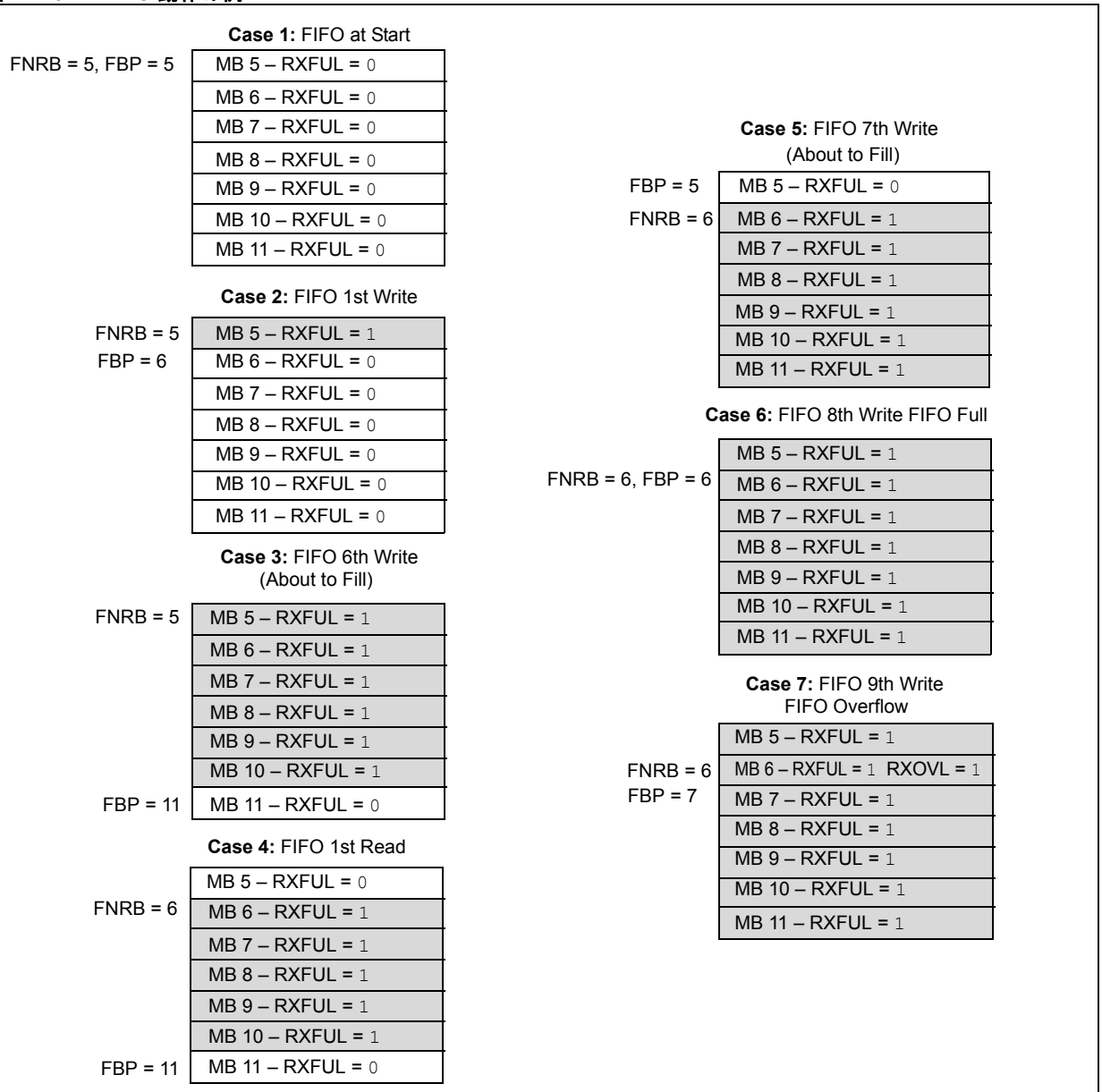
21.7.4 FIFO の例

図 21-15 に FIFO 動作の 7 つの例を示します。ここでは、FIFO の開始位置をメッセージ バッファ 5 (CiFCTRL<FSA> = 101) に、FIFO の末尾をメッセージ バッファ 11 (CiFCTRL<DMABS> = 011) に設定していると仮定します。

- **Case 1** は、メッセージを受信する前の初期状態の FIFO を表します。FIFO バッファポインタはメッセージ バッファ 5 (FRB = 5) を指し、FIFO 次読み出しバッファポインタはメッセージ バッファ 6 (FNRB = 5) を指します。
- **Case 2** は、1 つのメッセージを受信してメッセージ バッファ 5 に転送した後の FIFO を表します。FIFO バッファポインタはインクリメントし (FBP = 6)、メッセージ バッファ 5 の RXFUL ステータスビットがセットされます (RXFUL = 1)。
- **Case 3** は、6 番目のメッセージを受信した後の FIFO を表します。FIFO バッファポインタは FIFO 領域の最後の位置を指し (FBP = 5 + 6 = 11)、FIFO 次読み出しポインタは FIFO の開始位置を指しています (FNRB = 5)。この時 FIFO はほぼフルであり、FIFO 割り込みを生成します。
- **Case 4** は、アプリケーション ソフトウェアが最初の受信メッセージを読み出した後の FIFO を表します。アプリケーション ソフトウェアがメッセージ バッファ 5 の RXFUL ステータスビットをクリアすると、モジュールは MB5 に 1 を加えた値 (FNRB = 5 + 1 = 6) を FIFO 次読み出しバッファポインタに書き込みます。
- **Case 5** は、7 番目のメッセージを受信し、メッセージ バッファ 11 に書き込んだ後の FIFO を表します。メッセージバッファ 11 の RXFUL ステータ スビットがセットされます (RXFUL = 1)。インクリメントするのではなく、FIFO バッファポインタに FIFO 開始アドレスが再読み込みされます (FBP = FSA = 5)。ここで FBP が FNRB より 1 小さい数値である事に注意してください。これは、メッセージ バッファ 11 の RXFUL ステータスビットがセットされた時点で FIFO 割り込みが生成される条件です。

- **Case 6** は、8 番目のメッセージを受信し、メッセージ バッファ 5 に書き込んだ後の FIFO を表します。FIFO はフルの状態です。この条件を通知する割り込みはありません。
- **Case 7** は、9 番目のメッセージを受信した後の FIFO を表します。ここでは FIFO はオーバーフローしています。モジュールは書き込み用バッファの RXOVL ビットをセットします。このメッセージは失われます。モジュールは受信オーバーフロー割り込みを生成します。

図 21-15: FIFO 動作の例



凡例 : 網掛けしたメッセージ バッファは、ユーザ アプリケーションから読み出す準備が整った受信メッセージがある事を示します。
MB n は、メッセージ バッファ 5 ～ 11 を表します。
Write は、メッセージが FIFO メッセージ バッファに格納され、そのバッファに関連する RXFUL フラグがセットされた事を意味します。
Read は、アプリケーション ソフトウェアがそのバッファの内容を読み出す事によって FIFO メッセージ バッファを処理する事を意味します。バッファ位置が読み込まれると、アプリケーション ソフトウェアはそのバッファに対応する RXFUL ビットをクリアします。

21.7.5 DeviceNet™ のフィルタ処理

DeviceNet のフィルタ処理機能は、SID に加えてデータフィールドの最大 18 ビットまでをメッセージ アクセプタンス フィルタの EID と比較できる CAN 2.0A プロトコルを基にしています。

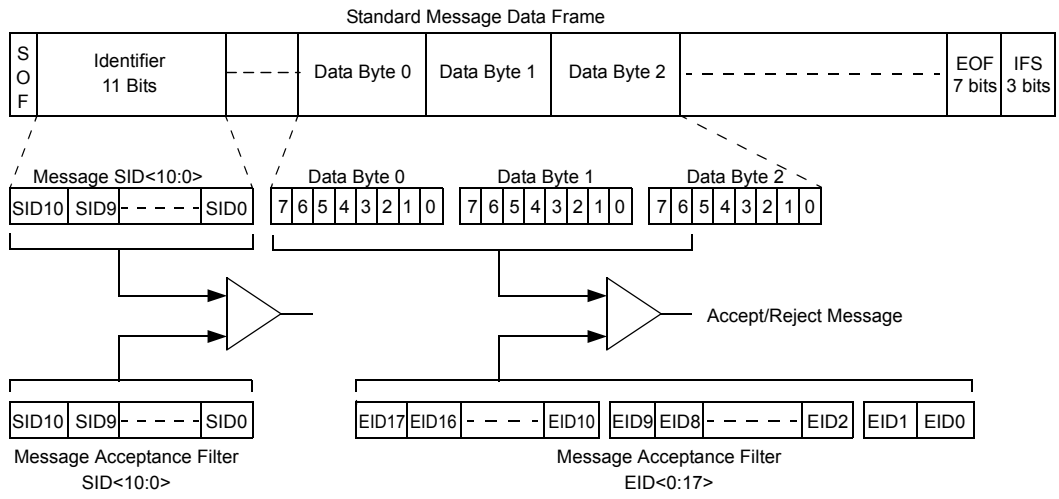
DeviceNet 機能は、ECAN 制御レジスタ 2 (CiCTRL2<4:0>) の DeviceNet フィルタ ビット番号 (DNCNT<4:0>) 制御ビットで有効化または無効化します。DNCNT フィールドで指定する値によって、メッセージ アクセプタンス フィルタの EID ビットとの比較に使用するデータ ビット数が決まります。CiCTRL2<DNCNT> ビットがクリアされている場合、DeviceNet 機能は無効です。

メッセージが受け入れられるには、11 ビットの SID はメッセージ アクセプタンス フィルタの SID<10:0> ビットに一致する必要があり、メッセージの最初の「n」データ ビットはメッセージ アクセプタンス フィルタの EID<17:0> ビットに一致する必要があります。例えば、図 21-16 で示すように、受信メッセージの最初の 18 データ ビットはメッセージ アクセプタンス フィルタの対応する識別子ビット (EID<17:0>) と比較されます。

Note: DeviceNet フィルタ処理機能は、以下の条件の全てがあてはまる場合に限り機能します。

- メッセージ IDE ビット = 0、メッセージが標準 ID メッセージである事を意味する
- CiRXFn レジスタの EXIDE ビット = 0
- CiRXMn レジスタの MIDE ビット = 1
- DNCNT ビットの値がゼロ以外の値である

図 21-16: DeviceNet™ フィルタ処理を使用する ECAN™ 動作



Note: EID ビットを表す DeviceNet™ フィルタ処理のコンフィグレーションは、DNCNT<4:0> = 10010 です。

21.7.5.1 フィルタの比較

表 21-4 に、CiCTRL2 (DNCNT<4:0>) 制御ビットで設定されるフィルタの比較を示します。例えば、DNCNT<4:0> = 00011 の場合、11 ビットの標準識別子が SID アクセプタンス フィルタ (SID<10:0>) に一致し、データバイト 0 の bit 7、6、5 が拡張識別子フィルタ (EID<0:2>) に一致するメッセージのみが受け入れられます。

表 21-4: DeviceNet™ フィルタ ビットのコンフィグレーション

DeviceNet™ フィルタのコンフィグレーション (DNCNT<4:0>)	比較される受信メッセージ データビット (バイト < ビット >)	アクセプタンス フィルタで使用される EID ビット
00000	比較なし -{}-	比較なし
00001	データバイト 0<7>	EID<17>
00010	データバイト 0<7:6>	EID<17:16>
00011	データバイト 0<7:5>	EID<17:15>
00100	データバイト 0<7:4>	EID<17:14>
00101	データバイト 0<7:3>	EID<17:13>
00110	データバイト 0<7:2>	EID<17:12>
00111	データバイト 0<7:1>	EID<17:11>
01000	データバイト 0<7:0>	EID<17:10>
01001	データバイト 0<7:0> とデータバイト 1<7>	EID<17:9>
01010	データバイト 0<7:0> とデータバイト 1<7:6>	EID<17:8>
01011	データバイト 0<7:0> とデータバイト 1<7:5>	EID<17:7>
01100	データバイト 0<7:0> とデータバイト 1<7:4>	EID<17:6>
01101	データバイト 0<7:0> とデータバイト 1<7:3>	EID<17:5>
01110	データバイト 0<7:0> とデータバイト 1<7:2>	EID<17:4>
01111	データバイト 0<7:0> とデータバイト 1<7:1>	EID<17:3>
10000	データバイト 0<7:0> とデータバイト 1<7:0>	EID<17:2>
10001	バイト 0<7:0> とバイト 1<7:0> とバイト 2<7>	EID<17:1>
10010	バイト 0<7:0> とバイト 1<7:0> とバイト 2<7:6>	EID<17:0>
10011 ～ 11111	無効な選択	無効な選択

21.7.5.2 特殊なケース

メッセージに含まれるデータ ビットが、DeviceNet フィルタ コンフィグレーションが要求するビット数より少ないという特殊なケースがあります。

- **Case 1 –** DNCNT<4:0> が 18 を超え、ユーザ アプリケーションが EID ビットの合計数を上回る多数のビットを選択している事を示す場合、フィルタ比較はデータの 18 番目のビット (データバイト 2 の bit 6) で終了します。SID と 18 データ ビット全てが一致する場合、メッセージは受け入れられます。
- **Case 2 –** DNCNT<4:0> が 16 を超え、受信メッセージのデータ長コード (DLC) が 2 (2 つのデータバイトのペイロードを示す) の場合、フィルタ比較はデータの 16 番目のビット (データバイト 1 の bit 0) で終了します。SID と 16 データ ビット全てが一致する場合、メッセージは受け入れられます。
- **Case 3 –** DNCNT<4:0> が 8 を超え、受信メッセージの DLC = 1 (1 つのデータバイトのペイロードを示す) の場合、フィルタ比較はデータの 8 番目のビット (データバイト 0 の bit 0) で終了します。SID と 8 データ ビット全てが一致する場合、メッセージは受け入れられます。
- **Case 4 –** DNCNT<4:0> が 0 を超え、受信メッセージの DLC = 0 の場合 (データ ペイロードがない事を示す)、フィルタ比較は標準識別子で終了します。SID が一致する場合、メッセージは受け入れられます。

21.8 DMA コントローラの設定

ECAN モジュールでは、メッセージ バッファが CAN メッセージの送受信の両方をサポートできるように DMA RAM の一部を使用しています。ECAN モジュールで使用するメッセージ バッファ数は、ECAN FIFO 制御 (CiFCTRL<15:13>) レジスタ内の DMA バッファサイズ (DMABS<2:0>) ビットで指定されます。DMA コントローラ内の DMAxSTA レジスタが、CAN バッファ領域の開始位置を定義します。DMA コントローラは、CPU に負荷をかけずに ECAN と DMA メッセージ バッファ間でデータを移動させます。

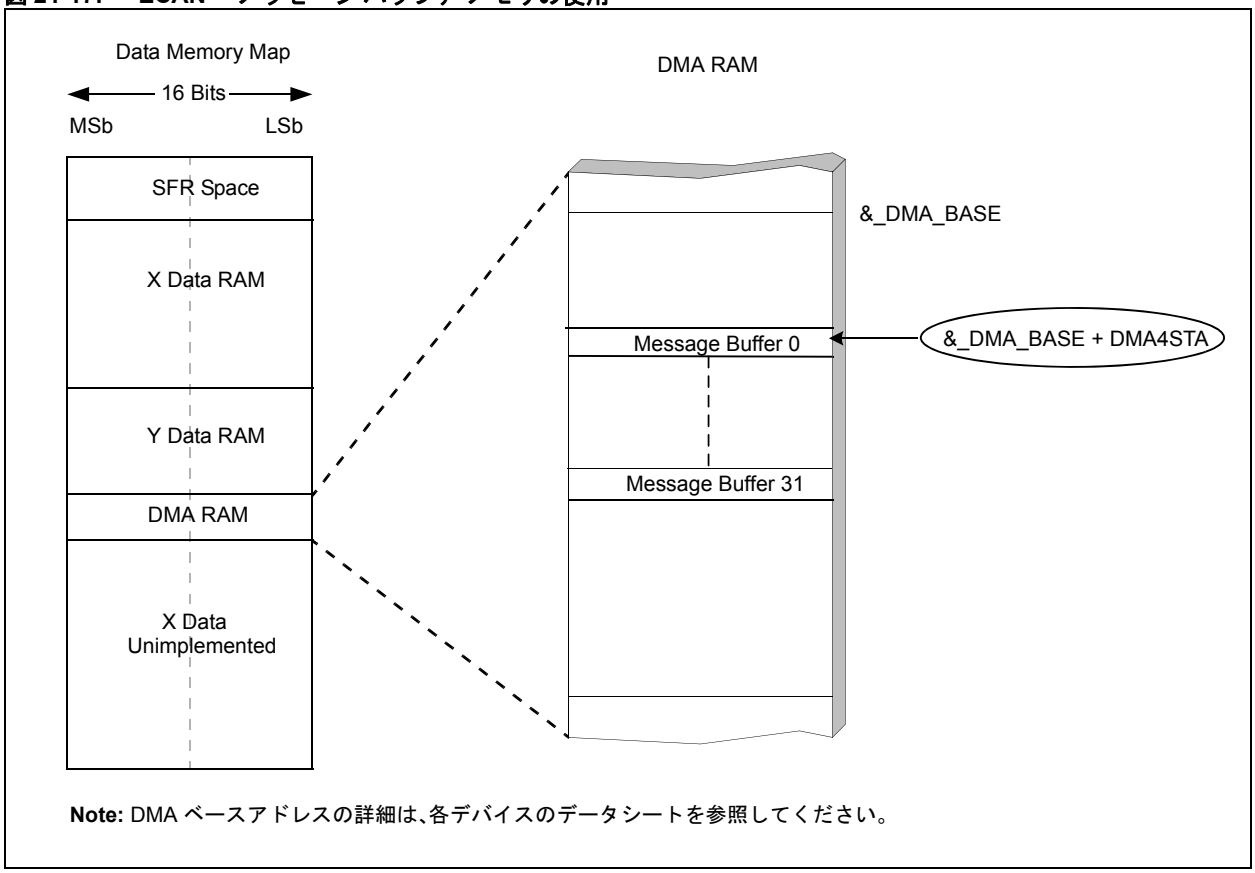
DMA コントローラは、DMA RAM と DMA 対応の dsPIC33F 周辺機能間でのデータ転送用に 8 つのチャンネルを備えています。CAN メッセージの送信と受信を両方ともサポートするには、DMA チャンネルが 2 つ必要です。各チャンネルには、表 21-5 に示すように、発生したメッセージ転送に基づいてイベントを割り当てる要求レジスタ (DMAxREQ) があります。

表 21-5: DMA のチャンネル要求

イベント	DMAxREQ レジスタを初期化する IRQ 番号
ECAN1 受信	34
ECAN1 送信	70
ECAN2 受信	55
ECAN2 送信	71

図 21-17 に、DMA チャンネル 4 を使用して ECAN バッファにアクセスする例を示します。

図 21-17: ECAN™ メッセージ バッファ メモリの使用



Note: DMA ベースアドレスの詳細は、各デバイスのデータシートを参照してください。

21.8.1 データ送信用の DMA 動作

ユーザ アプリケーションは、ECAN TX/RX バッファ m 制御 (CiTRmnCON<3>) レジスタ内のメッセージ送信要求 (TXREQ) ビットをセットする事によって、送信用メッセージを選択します。ECAN コントローラは DMA を使用し、メッセージ バッファからメッセージを読み出して送信します。ECAN モジュールは DMA サイクルを開始する送信データ割り込みを生成します。この割り込みに応答して、ECAN メッセージ送信に設定されている DMA チャンネルは、DMA RAM 内のメッセージ バッファからメッセージを読み出して ECAN 送信データ (CiTXD) レジスタに格納します。ECAN コントローラによって送信されるメッセージごとに 8 つのワードが転送されます。**21.2「CAN メッセージ フォーマット」**で、送信メッセージ バッファの詳細なレイアウトを説明します。例 21-6 に、DMA チャンネルを ECAN 1 送信用に設定するサンプルコードを示します。

例 21-6: ECAN 1 送信用の DMA チャンネル 0 の設定

```
/* Data Transfer Size: Word Transfer Mode */
DMA0CONbits.SIZE = 0x0;

/* Data Transfer Direction: DMA RAM to Peripheral */
DMA0CONbits.DIR = 0x1;

/* DMA Addressing Mode: Peripheral Indirect Addressing mode */
DMA0CONbits.AMODE = 0x2;

/* Operating Mode: Continuous, Ping-Pong modes disabled */
DMA0CONbits.MODE = 0x0;

/* Assign ECAN1 Transmit event for DMA Channel 0 */
DMA0REQ = 70;

/* Set Number of DMA Transfer per ECAN message to 8 words */
DMA0CNT = 7;

/* Peripheral Address: ECAN1 Transmit Register */
DMA0PAD = &C1TXD;

/* Start Address Offset for ECAN1 Message Buffer 0x0000 */
DMA0STA = 0x0000;

/* Channel Enable: Enable DMA Channel 0 */
DMA0CONbits.CHEN = 0x1;

/* Channel Interrupt Enable: Enable DMA Channel 0 Interrupt */
IEC0bits DMA0IE = 1;
```

Note: DMA コントローラの設定方法の詳細は、**セクション 22「ダイレクト メモリ アクセス (DMA)」**(DS70182) を参照してください。最新の文書はマイクロチップ社のウェブサイト (www.microchip.com) でご確認ください。

21.8.2 データ受信用の DMA 動作

ECAN コントローラがメッセージの受信 (8 ワード) を完了すると、完了したメッセージは DMA によって DMA RAM 内のメッセージ バッファに転送されます。ECAN モジュールは DMA サイクルを開始する受信データ割り込みを生成します。この割り込みに対して、ECAN メッセージ受信用に設定されている DMA チャンネルは、ECAN 受信データ (CiRXD) レジスタからメッセージを読み出して DMA RAM バッファに格納します。ECAN コントローラによって受信されるメッセージごとに 8 つのワードが転送されます。**21.2「CAN メッセージ フォーマット」**で、受信メッセージの詳細なレイアウトを説明しました。例 21-7 に、DMA チャンネルを ECAN 1 受信用に設定するサンプルコードを示します。

例 21-7: ECAN1 受信用の DMA チャンネル 1 のコンフィグレーション

```
/* Data Transfer Size: Word Transfer Mode */
DMA1CONbits.SIZE = 0x0;

/* Data Transfer Direction: Peripheral to DMA RAM */
DMA1CONbits.DIR = 0x0;

/* DMA Addressing Mode: Peripheral Indirect Addressing mode */
DMA1CONbits.AMODE = 0x2;

/* Operating Mode: Continuous, Ping-Pong modes disabled */
DMA1CONbits.MODE = 0x0;

/* Assign ECAN1 Receive event for DMA Channel 0 */
DMA1REQ = 34;

/* Set Number of DMA Transfer per ECAN message to 8 words */
DMA1CNT = 7;

/* Peripheral Address: ECAN1 Receive Register */
DMA1PAD = &C1RXD;

/* Start Address Offset for ECAN1 Message Buffer 0x0000 */
DMA1STA = 0x0000;

/* Channel Enable: Enable DMA Channel 1 */
DMA1CONbits.CHEN = 0x1;

/* Channel Interrupt Enable: Enable DMA Channel 1 Interrupt */
IEC0bits.DMA1IE = 1;
```

Note: DMA コントローラの設定方法の詳細は、**セクション 22「ダイレクト メモリ アクセス (DMA)」**(DS70182) を参照してください。最新の文書はマイクロチップ社のウェブサイト (www.microchip.com) でご確認ください。

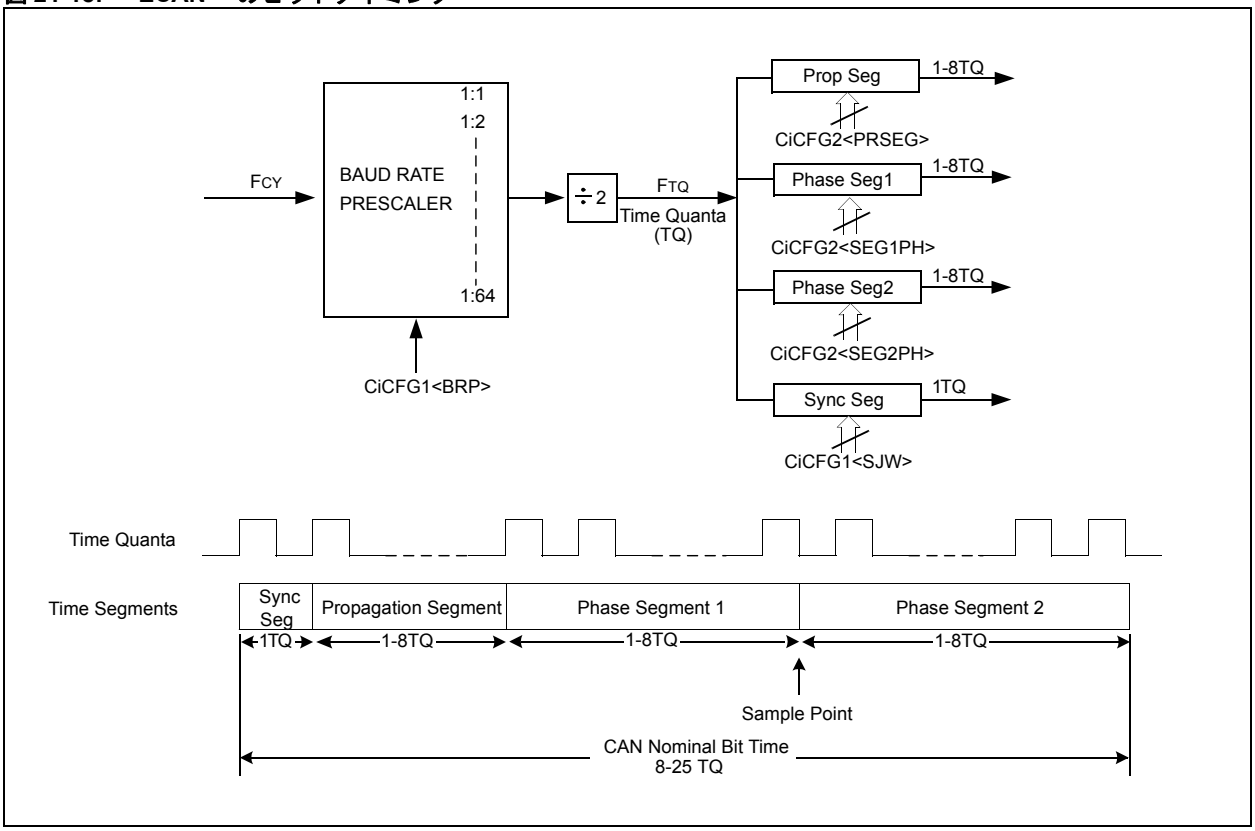
21.9 ビットタイミング

公称ビットレートは、CAN バス上で送信される 1 秒あたりのビット数です。定格ビット時間 = 1 ÷ 公称ビットレートです。

ビット時間には、オシレータのドリフトまたは伝搬遅延による位相差を補正するための 4 つの時間セグメントがあります。これらの時間セグメントは重なる事はなく、時間単位 (TQ) として表されます。1 TQ は、オシレータ クロックから得られる固定の時間単位です。定格ビット時間の時間単位の総数は、8 ～ 25 TQ の間でプログラムする必要があります。

図 21-18 に、システムクロックから時間単位クロック (FTQ) を取得する方法と、各時間セグメントをプログラムする方法を併せて示します。

図 21-18: ECAN™ のビットタイミング



21.9.1 ビットセグメント

各ビット送信時間は 4 つの時間セグメントから構成されます。

- **同期セグメント** – この時間セグメントは CAN バスに接続されている各ノードを同期させます。ビットエッジはこのセグメント内に収まる事が予測されます。CAN プロトコルに基づき、同期セグメントは 1 TQ と仮定されます。
- **伝搬セグメント** – この時間セグメントはバスラインまたはそのバスに接続された各種トランシーバによって発生する時間遅延を補正します。
- **位相セグメント 1** – この時間セグメントはエッジでの位相シフトによるエラーを補正します。この時間セグメントは、位相シフトを補正するために再同期中に延長されます。
- **位相セグメント 2** – この時間セグメントはエッジでの位相シフトによるエラーを補正します。この時間セグメントは、位相シフトを補正するために再同期中に短縮されます。位相セグメント 2 の時間はプログラマブルにするか、位相セグメント 1 の時間によって指定する事ができます。

21.9.2 サンプル ポイント

サンプル ポイントとは、サンプリングを行い、バスの状態を読み出して解釈する CAN ビット時間間隔内にあるポイントです。サンプル点は位相セグメント 1 と位相セグメント 2 の間に位置します。ECAN baud レート コンフィグレーション レジスタ (CiCFG2<6>) のサンプル CAN バス ライン (SAM) ビットの設定に応じて、サンプル点で 1 回または 3 回の CAN バスのサンプリングが可能です。

- CiCFG2<SAM> = 1 の場合、CAN バスはサンプル点で 3 回サンプリングされます。3 回のサンプリングでの最頻値によってビット値が決定します。
- CiCFG2<SAM> = 0 の場合、CAN バスはサンプル点で 1 回だけサンプリングされます。

21.9.3 同期

同期には 2 つのタイプがあります – ハード同期と再同期です。ハード同期はフレームの開始時点で発生します。再同期はフレーム内で発生します。

- **ハード同期**は、開始ビットのリセッシブからドミナントへの遷移で実行されます。ビット時間はそのエッジから再開します。
- **再同期**は、メッセージの同期セグメント内でビットエッジが発生しない場合に実行されます。信号内の位相エラーの量によって、位相セグメントの 1 つが短縮または延長されます。最大変化量は同期ジャンプ幅パラメータ (CiCFG1<SJW>) で決めます。

位相セグメント 1 および 2 の長さは、送受信ノードのオシレータ許容誤差に応じて変更可能です。再同期は、送受信用ノードのオシレータが原因で発生する位相シフトを補正します。

- **ビット延長** – ECAN 内の送信ノードのオシレータが受信ノードより遅い場合、ビット時間の位相セグメント 1 を延長して、次の立ち下がりエッジ (サンプル点) を遅らせる事ができます。
- **ビット短縮** – ECAN 内の送信ノードのオシレータが受信ノードより速い場合、ビット時間の位相セグメント 2 を短縮して、次の立ち下がりエッジ (次のビットのサンプル点) を早める事ができます。
- **同期ジャンプ幅(SJW)** – ECAN baud レート コンフィグレーション レジスタ (CiCFG1<7:6>) の SJW<1:0> ビットは、位相セグメント 1 および 2 の時間間隔に適用できる延長または短縮の量を制限する事によって、同期ジャンプ幅を決定します。このセグメントは、位相セグメント 2 の時間より長くする事はできません。同期ジャンプ幅は 1 ～ 4 TQ です。

21.9.4 ECAN ビット時間の計算

ECAN モジュールのビット タイミングを設定するためにユーザ アプリケーションで実行する必要がある手順を、以下で例を挙げて説明します。

1. 時間単位の周波数 (FTQ) を計算します。
   a) CAN バスの baud レート (FBAUD) を選択します。
   b) システム要件に応じてビット時間の時間単位数を選択します。

   時間単位の周波数 (FTQ) は、式 21-2 によって求めます。

式 21-2: 時間単位の周波数の計算

$$F_{TQ} = N \cdot F_{BAUD}$$

Note 1: 定格ビット時間の時間単位の総数は、8 ～ 25 TQ の間でプログラムする必要があります。従って、時間単位の周波数 (FTQ) は、baud レート (FBAUD) の 8 ～ 25 倍の間です。

2: 正確なビット時間を得るために、FTQ を FBAUD の整数倍にします。整数倍でない場合、オシレータ入力周波数または baud レートを変更する必要があります。

2. baud レート プリスケーラ (CiCFG1<BRP>) を計算します。baud レート プリスケーラは、式 21-3 によって求めます。

式 21-3: baud レート プリスケーラの計算

$$CiCFG1(BRP) = \left(\frac{F_{CY}}{(2 \cdot F_{TQ})}\right) - 1$$

3. 個々のビット時間セグメントを選択します。個々のビット時間セグメントは、CiCFG2 レジスタを使用して選択します。
   ビット時間 = 同期セグメント + 伝搬セグメント + 位相セグメント 1 + 位相セグメント 2.

Note 1: (伝搬セグメント + 位相セグメント 1) は位相セグメント 2 の長さ以上である必要があります。

2: 位相セグメント 2 は同期ジャンプ幅より長い必要があります。

例 21-8: CAN ビット タイミングの計算例

- ステップ 1: 時間単位の周波数 (FCY = 40 MHz) を計算します。
  FBAUD = 1 Mbps、時間単位数 N = 20 の場合、FTQ = 20 MHz です。
- ステップ 2: baud レート プリスケーラを計算します。
  CiCFG1<BRP> = (40000000/(2 * FTQ)) - 1 = 1 - 1 = 0.
- ステップ 3: 個々のビット時間セグメントを選択します。
  - 同期セグメント = 1 TQ (定数) とします。
  - システム特性に基づき、伝搬遅延 = 5 TQ と仮定します。
  - サンプル点は定格ビット時間の 70% の時点と仮定します。位相セグメント 2 = 定格ビット時間の 30% = 6 TQ とします。
  - この結果、位相セグメント 1 = 20 TQ - (1 TQ + 5 TQ + 6 TQ) = 8 TQ です。

例 21-9 に、ECAN ビットのタイミング パラメータを設定するコードを示します。

例 21-9: ECAN™ ビットタイミング パラメータを設定するサンプルコード

```
/* Set the Operating Frequency of the device to be 40MHz */

#define FCY 40000000

/* Set the ECAN module for Configuration Mode before writing into the Baud
Rate Control Registers*/

C1CTRL1bits.REQOP = 4;

/* Wait for the ECAN module to enter into Configuration Mode */

while(C1CTRL1bits.OPMODE! = 4);

/* Phase Segment 1 time is 8 TQ */
C1CFG2bits.SEG1PH = 0x7;

/* Phase Segment 2 time is set to be programmable */

C1CFG2bits.SEG2PHTS = 0x1;

/* Phase Segment 2 time is 6 TQ */

C1CFG2bits.SEG2PH = 0x5;

/* Propagation Segment time is 5 TQ */

C1CFG2bits.PRSEG = 0x4;

/* Bus line is sampled three times at the sample point */

C1CFG2bits.SAM = 0x1;

/* Synchronization Jump Width set to 4 TQ */

C1CFG1bits.SJW = 0x3;

/* Baud Rate Prescaler bits set to 1:1, i.e., TQ = (2*1*1)/ FCAN */

C1CFG1bits.BRP = 0x0 ;

/* Put the ECAN Module into Normal Mode Operating Mode*/

C1CTRL1bits.REQOP = 0;

/* Wait for the ECAN module to enter into Normal Operating Mode */

while(C1CTRL1bits.OPMODE! = 0);
```

21.10 ECAN のエラー管理

21.10.1 CAN バスのエラー

CAN プロトコルでは、エラーを検出する 5 種類の方法を定義しています。

- ビットエラー
- 肯定応答エラー
- フォームエラー
- スタッフィング エラー
- CRC エラー

ビットエラーと肯定応答エラーはビットレベルで発生します。他の 3 つのエラーはメッセージレベルで発生します。

21.10.1.1 ビットエラー

バス上でビットを送信しているノードは、バスも監視しています。監視されたビット値が送信ビット値と異なると、ビットエラーが検出されます。例外は、アービトレーション フィールドのスタッフ ビット ストリーム中か、ACK スロット中にリセッシブビットが送信される場合です。この場合、ドミナントビットを監視している際にはビットエラーは発生しません。パッシブ エラー フレームを送信しドミナントビットを検出するトランスミッタでは、これをビットエラーと解釈しません。

21.10.1.2 肯定応答エラー

メッセージの肯定応答フィールドでは、肯定応答スロット (リセッシブビットとして送信されている) にドミナントビットが含まれているかどうかをトランスミッタが検査します。含まれてない場合、他にフレームを正しく受信したノードがない事を意味します。肯定応答エラーが発生した場合、結果としてそのメッセージの処理を繰り返す必要があります。この場合、エラーフレームは生成されません。

21.10.1.3 フォームエラー

固定された形式のビットフィールド (EOF、インターフレーム スペース、肯定応答デリミタまたは CRC デリミタ) が不正なビットを含む場合、フォームエラーが検出されます。レシーバでは、EOF (End-of-Frame) の最終ビットの間、ドミナントビットはフォームエラーとして扱われません。

21.10.1.4 スタッフィング エラー

スタッフィング エラーは、ビット スタッフィングのメソッドでコーディングされた、メッセージ フィールド内で 6 回連続した同じビットレベルのビット時間で検出されます。

21.10.1.5 CRC エラー

メッセージを送信するノードは、送信メッセージに対応する CRC を計算して送信します。バス上の全レシーバは、トランスミッタと同じ CRC 計算を実行します。計算結果が受信メッセージから取得した CRC 値と等しくない場合、CRC エラーが検出されます。

21.10.2 障害隔離

バス上の全ての CAN コントローラは、各メッセージ内で上述のエラーを検出しようと試みます。エラーを検出すると、検出ノードはエラーフレームを送信し、バストラフィックを破棄します。他のノードはエラーフレームが引き起こすエラーを検出し (他のノードが元のエラーを検出していない場合)、適切な措置を取ります (現在のメッセージは破棄します)。

ECAN モジュールは、以下の 2 つのエラーカウンタを保持します。

- 送信エラーカウンタ (CiEC<TERRCNT>)
- 受信エラーカウンタ (CiEC<RERRCNT>)

これらのカウンタのインクリメント / デクリメントに関してはいくつかの規則があります。基本的に、障害を検出したトランスミッタは、リッスンしているノードが受信エラーカウンタをインクリメントするより早く、送信エラーカウンタをインクリメントします。これは、障害が発生しているのはトランスミッタである可能性が高いためです。

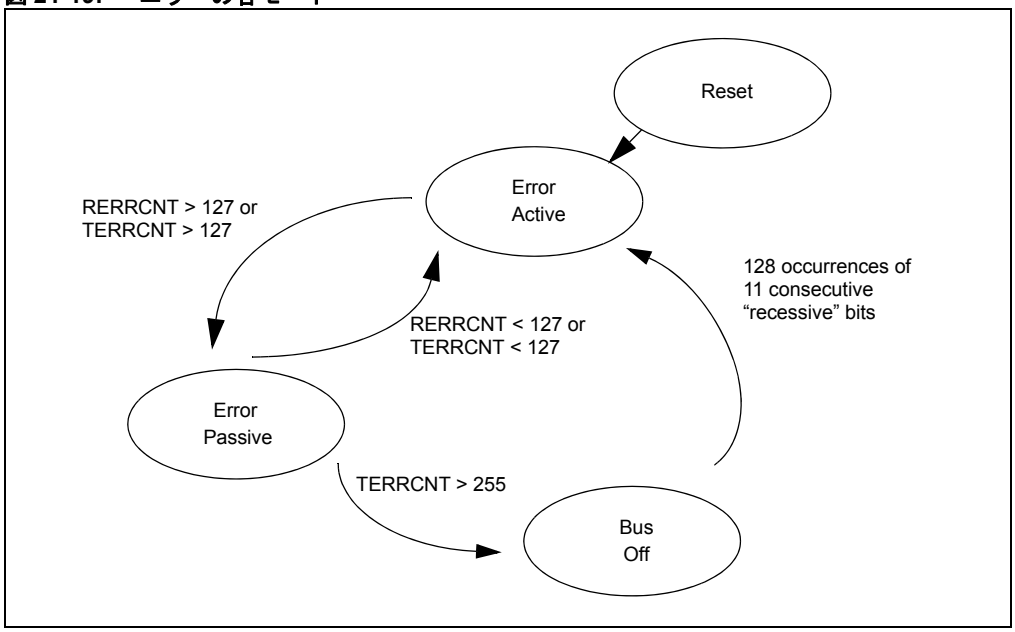
Note: エラーカウンタは CAN 2.0B 仕様に従って変更できます。

ノードはエラーアクティブ モードで開始されます。2 つあるエラーカウンタのいずれかの値が 127 以上になると、ノードはエラーパッシブと呼ばれる状態に移行します。送信エラーカウンタの値が 255 を超えると、ノードはバス OFF 状態に移行します。

- エラーアクティブ ノードがエラーを検出すると、アクティブエラー フレームを送信します。
- エラーパッシブ ノードがエラーを検出すると、パッシブエラー フレームを送信します。
- バス OFF 状態のノードは、バス上に何も送信しません。

さらに、図 21-19 に示すように、ECAN モジュールはノードがエラーパッシブ状態に移行する前に (送信エラーカウンタの値が 96 以上になった場合) ユーザ アプリケーションに警告するエラー警告機能を使用します。

図 21-19: エラーの各モード



21.10.2.1 エラーパッシブ状態でのトランスミッタ

トランスミッタ エラーパッシブ (CiINTF<TXBP>) ビットは、送信エラーカウンタの値が 128 以上になるとセットされ、エラーパッシブ状態に移行した時点でエラー割り込み (CiINTF<ERRIF>) が生成されます。送信エラーパッシブ フラグは、送信エラーカウンタの値が 128 未満になると、ハードウェアによって自動的にクリアされます。

21.10.2.2 エラーパッシブ状態でのレシーバ

レシーバ エラーパッシブ (CiINTF<RXBP>) ビットは、受信エラーカウンタの値が 128 以上になるとセットされ、エラーパッシブ状態に移行した時点でエラー割り込み (CiINTF<ERRIF>) が生成されます。受信エラーパッシブ フラグは、受信エラーカウンタの値が 128 未満になると、ハードウェアによって自動的にクリアされます。

21.10.2.3 バス OFF 状態でのトランスミッタ

トランスミッタ バス OFF (CiINTF<TXBO>) ビットは、送信エラーカウンタの値が 256 以上になるとセットされ、エラー割り込み (CiINTF<ERRIF>) が生成されます。

21.10.2.4 エラー警告状態でのトランスミッタ

トランスミッタ エラー警告 (CiINTF<TXWAR>) ビットは、送信エラーカウンタの値が 96 ～ 127 の範囲 (両端値を含む) になるとセットされ、エラー警告状態に移行した時点でエラー割り込み (CiINTF<ERRIF>) が生成されます。送信エラー警告フラグは、送信エラーカウンタの値が 96 未満または 127 超になる (すなわち ECAN モジュールがバスパッシブ状態に移行した) と、ハードウェアによって自動的にクリアされます。

21.10.2.5 エラー警告状態でのレシーバ

レシーバ エラー警告 (CiINTF<RXWAR>) ビットは、受信エラーカウンタの値が 96 ～ 127 の範囲 (両端値を含む) になるとセットされ、エラー警告状態に移行した時点でエラー割り込み (CiINTF<ERRIF>) が生成されます。受信エラー警告フラグは、受信エラーカウンタの値が 96 未満または 127 超になる (すなわち ECAN モジュールがバスパッシブ状態に移行した) と、ハードウェアによって自動的にクリアされます。

さらに、エラー状態警告フラグ (CiINTF<EWARN>) ビットがあり、エラーカウンタの少なくとも 1 つの値がエラー警告限界の 96 以上になるとセットされます。EWARN は、2 つのエラーカウンタの値がエラー警告限界未満になるとリセットされます。

21.11 ECAN 割り込み

ECAN モジュールは 3 種類の割り込みを生成します。それぞれが固有の割り込みベクタ、割り込みイネーブル制御ビット、割り込みステータスフラグ、割り込み優先度制御ビットを備えています。以下の割り込みがあります。

- **CiTX** – ECAN 送信データ要求割り込み
- **CiRX** – ECAN 受信データレディ割り込み
- **Ci** – ECAN イベント割り込み

21.11.1 ECAN 送信データ要求割り込み

送信データ要求割り込みは、ECAN 送信データ (CiTXD) レジスタを経由する ECAN メッセージのシングルワード送信です。ユーザ アプリケーションは、適切な DMA RAM バッファから ECAN モジュール (CiTXD レジスタ) に自動的にメッセージを転送するのに、ECAN 送信データ要求割り込みを DMA チャンネルに割り当てる必要があります。

21.11.2 ECAN 受信データレディ割り込み

受信データ要求割り込みは、ECAN 受信データ (CiRXD) レジスタを経由する ECAN メッセージのシングルワード受信です。ユーザ アプリケーションは、ECAN モジュール (CiRXD レジスタ) から適切な DMA RAM バッファに自動的にメッセージを転送するのに、ECAN 受信データレディ割り込みを DMA チャンネルに割り当てる必要があります。

21.11.3 ECAN イベント割り込み

ECAN イベント割り込みには 7 つの主要な要因があり、各々を個別に有効にできます。割り込みフラグ (CiINTF) レジスタには割り込みフラグがあり、割り込みイネーブル (CiINTE) レジスタにはイネーブルビットがあります。ECAN 割り込みコード (CiVEC<6:0>) レジスタ内の割り込みフラグコード (ICODE<6:0>) ビットは、割り込みを効率的に処理するためのジャンプテーブルと組み合わせて使用できます。エラー割り込みを除いて、全ての割り込みに 1 つの要因があります。5 つのエラー割り込み要因 (TX エラー警告、RX エラー警告、TX エラーパッシブ、RX エラーパッシブ、TX バス OFF) のいずれも、エラー割り込みフラグをセットできます。エラー割り込み要因は、CiINTF レジスタを読み出す事によって判定されます。ECAN モジュールは、CiINTF レジスタ内の少なくとも 1 つの条件がアクティブである場合に限り ECAN イベント割り込み (Ci 割り込み) を生成し、CiINTE レジスタ内の対応する割り込みが有効になります。CPU は、CiIF ビットと CiIE ビットがセットされると、CAN イベント割り込みサービスルーチンにジャンプします。

Note:	ICODE<6:0> ビットには、アクティブな最優先 CAN 割り込み条件が反映されます。割り込みは有効になっている (CiINTE レジスタ内の IE ビットがセットされている) 必要があり、割り込み条件はアクティブである (CiINTF レジスタ内の IF ビットがセットされている) 必要があります。

図 21-20 に、各種割り込み要因からの ECAN イベント割り込み生成を示します。

21.11.3.1 送信バッファ割り込み

メッセージ送信に設定されているメッセージ バッファ 0 ～ 7 は、CAN メッセージの送信後に送信バッファ割り込み (CiINTF<TBIF>) ビットを生成します。ICODE ビットは、送信バッファ割り込みを生成したメッセージ バッファを示します。送信バッファ割り込みは、TBIF ビットをクリアする事によって割り込みサービスルーチンでクリアする必要があります。

21.11.3.2 受信バッファ割り込み

メッセージを正常に受信して受信バッファの 1 つに読み込むと (メッセージ バッファ0 ～ 31)、モジュールが CiRXFULm<RXFULn> ビットをセットした後、受信バッファ割り込み (CiINTF<RBIF>) が生成されます。ICODE ビットは割り込みを生成したバッファを示します。受信バッファ割り込みは、RBIF ビットをクリアする事によって割り込みサービスルーチンでクリアする必要があります。

21.11.3.3 受信バッファ オーバーフロー割り込み

メッセージを正常に受信したにもかかわらず指定したバッファがフルの場合、モジュールがCiRXOVFm<RXOVFn> ビットをセットした後、受信オーバーフロー割り込み (CiINTF<RBOVIF>) が生成されます。ECAN 割り込みコード (CiVEC<6:0>) レジスタ内の割り込みフラグ コード (ICODE<6:0>) ビットは、割り込みを生成したバッファを示します。受信バッファ オーバーフロー割り込みは、RBOVIF ビットをクリアする事によって割り込みサービスルーチンでクリアする必要があります。

21.11.3.4 FIFO ほぼフル割り込み

FIFO に利用できるバッファが 1 つしか残っていない場合、モジュールが最後から 2 番目のバッファの CiRXFULm<RXFULn> ビットをセットした後、FIFO 割り込み (CiINTF<FIFOIF>) が生成されます。CiVEC<ICODE> ビットは FIFO オーバーフロー状態を示します。FIFO ほぼフル割り込みは、CiINTF<FIFOIF> ビットをクリアする事によって割り込みサービスルーチンでクリアする必要があります。

21.11.3.5 エラー割り込み

エラー割り込み (CiINTF<ERRIF>) は以下の 5 つの要因によって生成されます。

- TX エラー警告
- RX エラー警告
- TX エラーパッシブ
- RX エラーパッシブ
- TX バス OFF

CiVEC<ICODE> ビットはエラー状態を示します。エラー割り込みは、CiINTF<ERRIF> ビットをクリアする事によって割り込みサービスルーチンでクリアする必要があります。

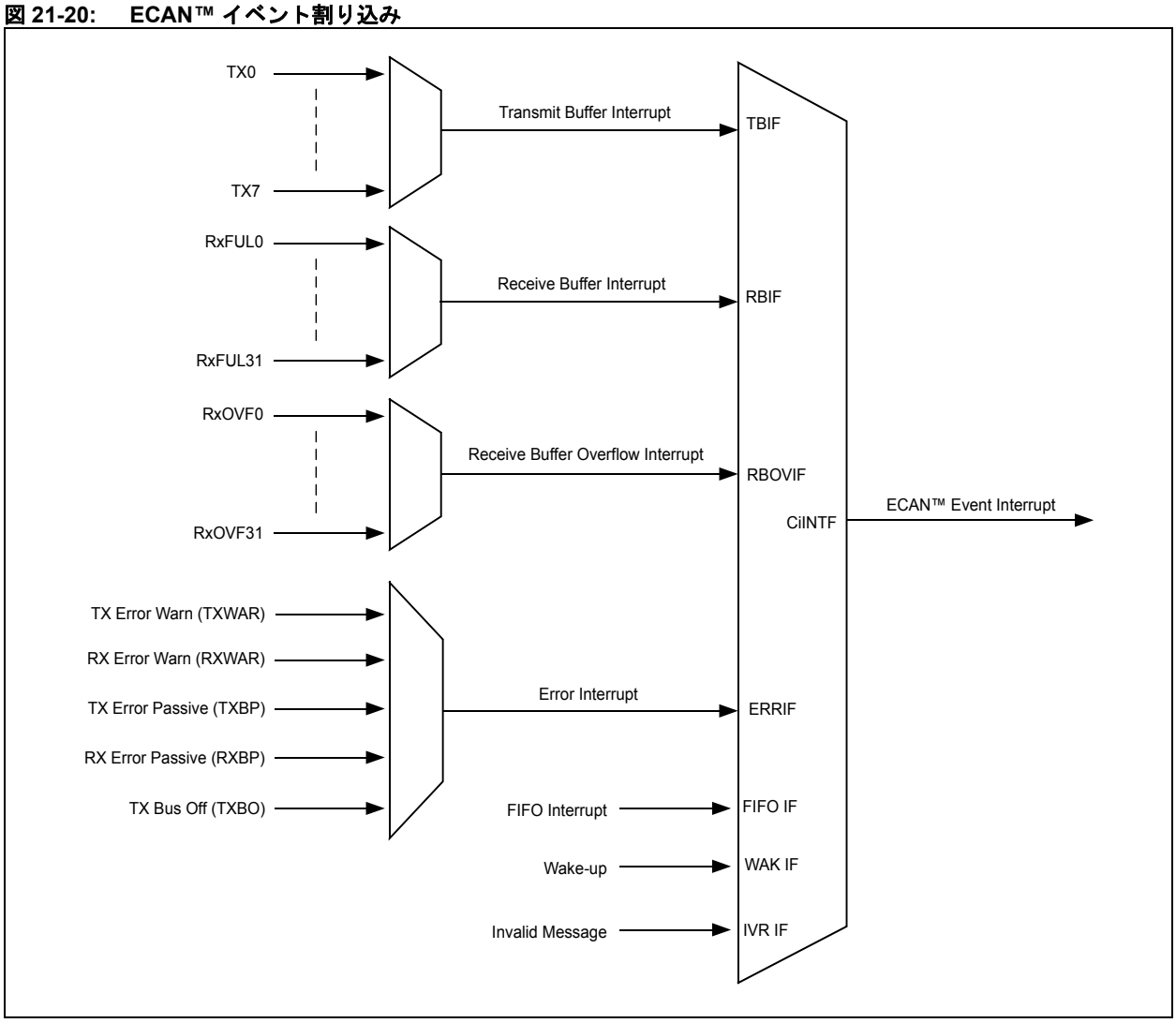
21.11.3.6 ウェイクアップ割り込み

スリープモードでは、デバイスは ECAN 受信ピン (CiRX) のバス アクティビティを監視します。バス アクティビティが検出されると、ウェイクアップ (CiINTF<WAKIF>) 割り込みが生成されます。CiVEC<ICODE> ビットはウェイクアップ状態を示します。ウェイクアップ割り込みは、CiINTF<WAKIF> ビットをクリアする事によって割り込みサービスルーチンでクリアする必要があります。

21.11.3.7 無効メッセージ受信割り込み

メッセージ受信中の他のタイプのエラーの場合、無効メッセージ受信割り込みが生成されます。

図 21-20: ECAN™ イベント割り込み



21.12 ECAN 低消費電力モード

ECAN モジュールは、CPU の PWRSAV 命令に応答可能です。

21.12.1 スリープモード

CPU の PWRSAV,1 命令は、オシレータを停止し、全てのシステムクロックをシャットダウンします。ユーザ アプリケーションでは、CPU がスリープモードに移行する際にモジュールを確実に非アクティブにする必要があります。この原則に違反して生じる致命的な結果から CAN バス システムを保護するために、モジュールはスリープ時に CiTX ピンをリセッシブ状態にします。CPU の PWRSAV 命令を実行する前に、モジュールを無効モードに移行させる事を推奨します。

21.12.2 アイドルモード

CPU の PWRSAV,0 命令は、オプションでクロックをシャットダウンするようモジュールに通知します。ECAN 制御レジスタ 1 (CiCTRL1<13>) のアイドルモード時停止 (CSIDL) ビットが「1」の場合、モジュールの電源を遮断します。ユーザ アプリケーションでは、CPU がアイドルモードに移行する前にモジュールを確実に非アクティブにする必要があります。この原則に違反して生じる致命的な結果から CAN バス システムを保護するために、モジュールはスリープ時に CiTX ピンをリセッシブ状態にします。CPU の PWRSAV 命令を実行する前に、モジュールを無効モードに移行させる事を推奨します。

21.12.3 ウェイクアップ機能

本モジュールはデバイスがスリープ状態の間、RX ラインのアクティビティを監視します。WAKIE ビットがセットされている場合、バス アクティビティが検出されるとモジュールは割り込みを生成します。オシレータと CPU の起動に遅延が生じるため、ウェイクアップを引き起こしたメッセージ アクティビティは失われます。

CPU がスリープからウェイクアップした後、CPU は CAN イベント割り込みサービスルーチンを実行します (割り込みが有効な場合)。ただし、CAN モジュール自体は依然として無効な状態です。CAN バス ウェイクアップ機能は、デバイスがスリープモードにある場合にのみ動作します。

本モジュールには、CiRX 入力ライン上にローパス フィルタ機能があり、モジュールが CPU スリープモードにある際はこの機能が有効になっている必要があります。このフィルタは、CAN バス上のグリッチによるウェイクアップからモジュールを保護します。フィルタを有効にするには、CiCFG2<WAKFIL> ビットをセットします。

21.13 入力キャプチャを使用する ECAN タイムスタンプ

ECAN モジュールは、有効なフレームが受け入れられた時点でタイマキャプチャ入力に送信可能な信号を生成します。これはタイムスタンプとネットワークの同期に役立ちます。CAN 仕様では、EOF フィールドが正常に送信される前にエラーが発生していない場合にフレームが有効であると定義しているため、EOF の直後にタイマ信号が生成されます。1 ビット時間のパルスが生成されます。

タイムスタンプは、CAN メッセージ受信タイマ キャプチャ イベント イネーブル (CANCAP) 制御ビット(CiCTRL1<3>)で有効にします。タイムスタンプには、IC2キャプチャ入力を使用します。

Note:	CAN キャプチャが有効な場合、IC2 ピンは汎用の入力キャプチャピンとして使用できません。このモードでは、IC2 チャンネルは IC2 ピンの代わりに C1RX または C2RX ピンから入力信号を得ます。

21.14 レジスタマップ

ECAN に関連するレジスタマップは、以下の通りです。

- ECAN1 レジスタマップ (C1CTRL1.WIN = 0 または 1 の場合) (表 21-6 参照)
- ECAN1 レジスタマップ (C1CTRL1.WIN = 0 の場合) (表 21-7 参照)
- ECAN1 レジスタマップ (C1CTRL1.WIN = 1 の場合) (表 21-8 参照)
- ECAN2 レジスタマップ (C2CTRL1.WIN = 0 または 1 の場合) (表 21-9 参照)
- ECAN2 レジスタマップ (C2CTRL1.WIN = 0 の場合) (表 21-10 参照)
- ECAN2 レジスタマップ (C2CTRL1.WIN = 1 の場合) (表 21-11 参照)

表 21-6: ECAN1 レジスタマップ (C1CTRL1.WIN = 0 または 1 の場合)

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
C1CTRL1	—	—	CSIDL	ABAT	—	REQOP<2:0>			OPMODE<2:0>			—	CANCAP	—	—	WIN	0480
C1CTRL2	—	—	—	—	—	—	—	—	—	—	—	DNCNT<4:0>					0000
C1VEC	—	—	—	FILHIT<4:0>					—	ICODE<6:0>							0040
C1FCTRL	DMABS<2:0>			—	—	—	—	—	—	—	—	FSA<4:0>					0000
C1FIFO	—	—	FBP<5:0>						—	—	FNRB<5:0>						0000
C1INTF	—	—	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN	IVRIF	WAKIF	ERRIF	—	FIFOIF	RBOVIF	RBIF	TBIF	0000
C1INTE	—	—	—	—	—	—	—	—	IVRIE	WAKIE	ERRIE	—	FIFOIE	RBOVIE	RBIE	TBIE	0000
C1EC	TERRCNT<7:0>								RERRCNT<7:0>								0000
C1CFG1	—	—	—	—	—	—	—	—	SJW<1:0>		BRP<5:0>						0000
C1CFG2	—	WAKFIL	—	—	—	SEG2PH<2:0>			SEG2PHTS	SAM	SEG1PH<2:0>			PRSEG<2:0>			0000
C1FEN1	FLTEN15	FLTEN14	FLTEN13	FLTEN12	FLTEN11	FLTEN10	FLTEN9	FLTEN8	FLTEN7	FLTEN6	FLTEN5	FLTEN4	FLTEN3	FLTEN2	FLTEN1	FLTEN0	0000
C1FMSKSEL1	F7MSK<1:0>		F6MSK<1:0>		F5MSK<1:0>		F4MSK<1:0>		F3MSK<1:0>		F2MSK<1:0>		F1MSK<1:0>		F0MSK<1:0>		0000
C1FMSKSEL2	F15MSK<1:0>		F14MSK<1:0>		F13MSK<1:0>		F12MSK<1:0>		F11MSK<1:0>		F10MSK<1:0>		F9MSK<1:0>		F8MSK<1:0>		0000

凡例 : — = 未実装、「0」として読み出し、リセット時の値を 16 進数で表示

表 21-7: ECAN1 レジスタマップ (C1CTRL1.WIN = 0 の場合)

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
	WIN = x の場合、定義を参照																
C1RXFUL1	RXFUL15	RXFUL14	RXFUL13	RXFUL12	RXFUL11	RXFUL10	RXFUL9	RXFUL8	RXFUL7	RXFUL6	RXFUL5	RXFUL4	RXFUL3	RXFUL2	RXFUL1	RXFUL0	0000
C1RXFUL2	RXFUL31	RXFUL30	RXFUL29	RXFUL28	RXFUL27	RXFUL26	RXFUL25	RXFUL24	RXFUL23	RXFUL22	RXFUL21	RXFUL20	RXFUL19	RXFUL18	RXFUL17	RXFUL16	0000
C1RXOVF1	RXOVF15	RXOVF14	RXOVF13	RXOVF12	RXOVF11	RXOVF10	RXOVF9	RXOVF8	RXOVF7	RXOVF6	RXOVF5	RXOVF4	RXOVF3	RXOVF2	RXOVF1	RXOVF0	0000
C1RXOVF2	RXOVF31	RXOVF30	RXOVF29	RXOVF28	RXOVF27	RXOVF26	RXOVF25	RXOVF24	RXOVF23	RXOVF22	RXOVF21	RXOVF20	RXOVF19	RXOVF18	RXOVF17	RXOVF16	0000
C1TR01CON	TXEN1	TXABT1	TXLARB1	TXERR1	TXREQ1	RTREN1	TX1PRI<1:0>		TXEN0	TXABAT0	TXLARB0	TXERR0	TXREQ0	RTREN0	TX0PRI<1:0>		0000
C1TR23CON	TXEN3	TXABT3	TXLARB3	TXERR3	TXREQ3	RTREN3	TX3PRI<1:0>		TXEN2	TXABAT2	TXLARB2	TXERR2	TXREQ2	RTREN2	TX2PRI<1:0>		0000
C1TR45CON	TXEN5	TXABT5	TXLARB5	TXERR5	TXREQ5	RTREN5	TX5PRI<1:0>		TXEN4	TXABAT4	TXLARB4	TXERR4	TXREQ4	RTREN4	TX4PRI<1:0>		0000
C1TR67CON	TXEN7	TXABT7	TXLARB7	TXERR7	TXREQ7	RTREN7	TX7PRI<1:0>		TXEN6	TXABAT6	TXLARB6	TXERR6	TXREQ6	RTREN6	TX6PRI<1:0>		xxxx
C1RXD	受信データワード																xxxx
C1TXD	送信データワード																xxxx

凡例 : x = リセット時の未知の値、— = 未実装、「0」として読み出し、リセット時の値を 16 進数で表示

表 21-8: ECAN1 レジスタマップ (C1CTRL1.WIN = 1 の場合)

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
	WIN = x の場合、定義を参照																
C1BUFPNT1	F3BP<3:0>				F2BP<3:0>				F1BP<3:0>				F0BP<3:0>				0000
C1BUFPNT2	F7BP<3:0>				F6BP<3:0>				F5BP<3:0>				F4BP<3:0>				0000
C1BUFPNT3	F11BP<3:0>				F10BP<3:0>				F9BP<3:0>				F8BP<3:0>				0000
C1BUFPNT4	F15BP<3:0>				F14BP<3:0>				F13BP<3:0>				F12BP<3:0>				0000
C1RXM0SID	SID<10:3>								SID<2:0>			—	MIDE	—	EID<17:16>		xxxx
C1RXM0EID	EID<15:8>								EID<7:0>								xxxx
C1RXM1SID	SID<10:3>								SID<2:0>			—	MIDE	—	EID<17:16>		xxxx
C1RXM1EID	EID<15:8>								EID<7:0>								xxxx
C1RXM2SID	SID<10:3>								SID<2:0>			—	MIDE	—	EID<17:16>		xxxx
C1RXM2EID	EID<15:8>								EID<7:0>								xxxx
C1RXF0SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF0EID	EID<15:8>								EID<7:0>								xxxx
C1RXF1SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF1EID	EID<15:8>								EID<7:0>								xxxx
C1RXF2SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF2EID	EID<15:8>								EID<7:0>								xxxx
C1RXF3SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF3EID	EID<15:8>								EID<7:0>								xxxx
C1RXF4SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF4EID	EID<15:8>								EID<7:0>								xxxx
C1RXF5SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF5EID	EID<15:8>								EID<7:0>								xxxx
C1RXF6SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF6EID	EID<15:8>								EID<7:0>								xxxx
C1RXF7SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF7EID	EID<15:8>								EID<7:0>								xxxx
C1RXF8SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF8EID	EID<15:8>								EID<7:0>								xxxx
C1RXF9SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF9EID	EID<15:8>								EID<7:0>								xxxx
C1RXF10SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF10EID	EID<15:8>								EID<7:0>								xxxx
C1RXF11SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx

凡例 : x = リセット時の未知の値、— = 未実装、「0」として読み出し、リセット時の値を 16 進数で表示

表 21-8: ECAN1 レジスタマップ (C1CTRL1.WIN = 1 の場合) (続き)

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
C1RXF11EID	EID<15:8>								EID<7:0>								xxxx
C1RXF12SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF12EID	EID<15:8>								EID<7:0>								xxxx
C1RXF13SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF13EID	EID<15:8>								EID<7:0>								xxxx
C1RXF14SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF14EID	EID<15:8>								EID<7:0>								xxxx
C1RXF15SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C1RXF15EID	EID<15:8>								EID<7:0>								xxxx

凡例 : x = リセット時の未知の値、— = 未実装、「0」として読み出し、リセット時の値を 16 進数で表示

表 21-9: ECAN2 レジスタマップ (C2CTRL1.WIN = 0 または 1 の場合)

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
C2CTRL1	—	—	CSIDL	ABAT	—	REQOP<2:0>			OPMODE<2:0>			—	CANCAP	—	—	WIN	0480
C2CTRL2	—	—	—	—	—	—	—	—	—	—	—	DNCNT<4:0>					0000
C2VEC	—	—	—	FILHIT<4:0>					—	ICODE<6:0>							0040
C2FCTRL	DMABS<2:0>			—	—	—	—	—	—	—	—	FSA<4:0>					0000
C2FIFO	—	—	FBP<5:0>						—	—	FNRB<5:0>						0000
C2INTF	—	—	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN	IVRIF	WAKIF	ERRIF	—	FIFOIF	RBOVIF	RBIF	TBIF	0000
C2INTE	—	—	—	—	—	—	—	—	IVRIE	WAKIE	ERRIE	—	FIFOIE	RBOVIE	RBIE	TBIE	0000
C2EC	TERRCNT<7:0>								RERRCNT<7:0>								0000
C2CFG1	—	—	—	—	—	—	—	—	SJW<1:0>		BRP<5:0>						0000
C2CFG2	—	WAKFIL	—	—	—	SEG2PH<2:0>			SEG2PHTS	SAM	SEG1PH<2:0>			PRSEG<2:0>			0000
C2FEN1	FLTEN15	FLTEN14	FLTEN13	FLTEN12	FLTEN11	FLTEN10	FLTEN9	FLTEN8	FLTEN7	FLTEN6	FLTEN5	FLTEN4	FLTEN3	FLTEN2	FLTEN1	FLTEN0	0000
C2FMSKSEL1	F7MSK<1:0>		F6MSK<1:0>		F5MSK<1:0>		F4MSK<1:0>		F3MSK<1:0>		F2MSK<1:0>		F1MSK<1:0>		F0MSK<1:0>		0000
C2FMSKSEL2	F15MSK<1:0>		F14MSK<1:0>		F13MSK<1:0>		F12MSK<1:0>		F11MSK<1:0>		F10MSK<1:0>		F9MSK<1:0>		F8MSK<1:0>		0000

凡例 : — = 未実装、「0」として読み出し、リセット時の値を 16 進数で表示

表 21-10: ECAN2 レジスタマップ (C2CTRL1.WIN = 0 の場合)

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
	WIN = x の場合、定義を参照																
C2RXFUL1	RXFUL15	RXFUL14	RXFUL13	RXFUL12	RXFUL11	RXFUL10	RXFUL9	RXFUL8	RXFUL7	RXFUL6	RXFUL5	RXFUL4	RXFUL3	RXFUL2	RXFUL1	RXFUL0	0000
C2RXFUL2	RXFUL31	RXFUL30	RXFUL29	RXFUL28	RXFUL27	RXFUL26	RXFUL25	RXFUL24	RXFUL23	RXFUL22	RXFUL21	RXFUL20	RXFUL19	RXFUL18	RXFUL17	RXFUL16	0000
C2RXOVF1	RXOVF15	RXOVF14	RXOVF13	RXOVF12	RXOVF11	RXOVF10	RXOVF09	RXOVF08	RXOVF7	RXOVF6	RXOVF5	RXOVF4	RXOVF3	RXOVF2	RXOVF1	RXOVF0	0000
C2RXOVF2	RXOVF31	RXOVF30	RXOVF29	RXOVF28	RXOVF27	RXOVF26	RXOVF25	RXOVF24	RXOVF23	RXOVF22	RXOVF21	RXOVF20	RXOVF19	RXOVF18	RXOVF17	RXOVF16	0000
C2TR01CON	TXEN1	TXABAT1	TXLARB1	TXERR1	TXREQ1	RTREN1	TX1PRI<1:0>		TXEN0	TXABAT0	TXLARB0	TXERR0	TXREQ0	RTREN0	TX0PRI<1:0>		0000
C2TR23CON	TXEN3	TXABAT3	TXLARB3	TXERR3	TXREQ3	RTREN3	TX3PRI<1:0>		TXEN2	TXABAT2	TXLARB2	TXERR2	TXREQ2	RTREN2	TX2PRI<1:0>		0000
C2TR45CON	TXEN5	TXABAT5	TXLARB5	TXERR5	TXREQ5	RTREN5	TX5PRI<1:0>		TXEN4	TXABAT4	TXLARB4	TXERR4	TXREQ4	RTREN4	TX4PRI<1:0>		0000
C2TR67CON	TXEN7	TXABAT7	TXLARB7	TXERR7	TXREQ7	RTREN7	TX7PRI<1:0>		TXEN6	TXABAT6	TXLARB6	TXERR6	TXREQ6	RTREN6	TX6PRI<1:0>		xxxx
C2RXD	受信データワード																xxxx
C2TXD	送信データワード																xxxx

凡例 : x = リセット時の未知の値、— = 未実装、「0」として読み出し、リセット時の値を 16 進数で表示

表 21-11: ECAN2 レジスタマップ (C2CTRL1.WIN = 1 の場合)

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
	WIN = x の場合、定義を参照																
C2BUFPNT1	F3BP<3:0>				F2BP<3:0>				F1BP<3:0>				F0BP<3:0>				0000
C2BUFPNT2	F7BP<3:0>				F6BP<3:0>				F5BP<3:0>				F4BP<3:0>				0000
C2BUFPNT3	F11BP<3:0>				F10BP<3:0>				F9BP<3:0>				F8BP<3:0>				0000
C2BUFPNT4	F15BP<3:0>				F14BP<3:0>				F13BP<3:0>				F12BP<3:0>				0000
C2RXM0SID	SID<10:3>								SID<2:0>			—	MIDE	—	EID<17:16>		xxxx
C2RXM0EID	EID<15:8>								EID<7:0>								xxxx
C2RXM1SID	SID<10:3>								SID<2:0>			—	MIDE	—	EID<17:16>		xxxx
C2RXM1EID	EID<15:8>								EID<7:0>								xxxx
C2RXM2SID	SID<10:3>								SID<2:0>			—	MIDE	—	EID<17:16>		xxxx
C2RXM2EID	EID<15:8>								EID<7:0>								xxxx
C2RXF0SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF0EID	EID<15:8>								EID<7:0>								xxxx
C2RXF1SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF1EID	EID<15:8>								EID<7:0>								xxxx
C2RXF2SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF2EID	EID<15:8>								EID<7:0>								xxxx
C2RXF3SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF3EID	EID<15:8>								EID<7:0>								xxxx
C2RXF4SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF4EID	EID<15:8>								EID<7:0>								xxxx
C2RXF5SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF5EID	EID<15:8>								EID<7:0>								xxxx
C2RXF6SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF6EID	EID<15:8>								EID<7:0>								xxxx
C2RXF7SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF7EID	EID<15:8>								EID<7:0>								xxxx
C2RXF8SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF8EID	EID<15:8>								EID<7:0>								xxxx
C2RXF9SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF9EID	EID<15:8>								EID<7:0>								xxxx
C2RXF10SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF10EID	EID<15:8>								EID<7:0>								xxxx
C2RXF11SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx

凡例 : x = リセット時の未知の値、— = 未実装、「0」として読み出し、リセット時の値を 16 進数で表示

表 21-11: ECAN2 レジスタマップ (C2CTRL1.WIN = 1 の場合) (続き)

レジスタ名	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	全リセット
C2RXF11EID	EID<15:8>								EID<7:0>								xxxx
C2RXF12SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF12EID	EID<15:8>								EID<7:0>								xxxx
C2RXF13SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF13EID	EID<15:8>								EID<7:0>								xxxx
C2RXF14SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF14EID	EID<15:8>								EID<7:0>								xxxx
C2RXF15SID	SID<10:3>								SID<2:0>			—	EXIDE	—	EID<17:16>		xxxx
C2RXF15EID	EID<15:8>								EID<7:0>								xxxx

凡例 : x = リセット時の未知の値、— = 未実装、「0」として読み出し、リセット時の値を 16 進数で表示

21.15 関連アプリケーション ノート

本セクションに関連するアプリケーション ノートの一覧を次に示します。一部のアプリケーション ノートは dsPIC33F ファミリ向けではありません。ただし概念は共通しており、変更が必要であったり制限事項が存在するものの利用が可能です。拡張コントローラ エリア ネットワーク (ECAN™) モジュールに関連する最新のアプリケーション ノートは以下の通りです。

タイトル	アプリケーション ノート番号

現在、アプリケーション ノートはありません。

Note: dsPIC33F デバイス ファミリ向けのその他のアプリケーション ノートとサンプル コードは、マイクロチップ社のウェブサイト (www.microchip.com) をご覧ください。

21.16 改訂履歴

リビジョン A (2007 年 1 月)

初版発行

リビジョン B (2009 年 11 月)

このリビジョンでの変更内容は以下の通りです。

- **全体的な変更 :**
  - 図 21-18、**21.9.4「ECAN ビット時間の計算」**、例 21-8、例 21-9、表 21-6、表 21-9 の CANCKS ビットへの言及を全て削除した。
- **例 :**
  - 最大 CAN メッセージ サイズ 8 (0x0008) を反映するために、例 21-1: 標準データフレーム送信用のサンプルコードを更新した。例では 15 (0x000F) の値の記述が不正確であった。
  - メッセージ バッファ 0 に代わってメッセージ バッファ 2 を反映するために、例 21-2: 拡張データフレーム送信用のサンプルコードのコードを更新した。
- **図 :**
  - 図 21-5 の SRR ビットの論理値を「0」から「1」に更新した。
- **Notes:**
  - **21.7.2.2「メッセージ バッファ 0 ～ 7 へのメッセージ受信」**に網掛け注釈ボックスを追加し、バッファが受信用に設定されている場合に TXREQ ビットをセットする事の影響を説明した。
  - **21.7.5「DeviceNet™ のフィルタ処理」**に網掛け注釈ボックスを追加し、動作の制約事項を説明した。
  - **21.11.3「ECAN イベント割り込み」**に網掛け注釈ボックスを追加し、最優先 CAN 割り込み条件について補足説明した。
- **レジスタ :**
  - **CiFCTRL: ECAN™ FIFO 制御レジスタ**から bit 11 (CANCKS) を削除した (レジスタ 21-18 参照)。
  - CiFEN1: ECAN アクセプタンス フィルタ イネーブル レジスタの FLTEN6 ～ FLTEN15 の既定値を R/W-0 から R/W-1 に変更した (レジスタ 21-3 参照)。
  - CiRXFnSID: ECANアクセプタンス フィルタ標準識別子レジスタ に網掛け注釈ボックスを追加した (レジスタ 21-4 参照)。
  - F7MSK<1:0> = 11 を「マスクなし」から「予約済み」に変更した (レジスタ 21-8 参照)。
  - F15MSK<1:0> = 11 を「マスクなし」から「予約済み」に変更した (レジスタ 21-9 参照)。
- **セクション :**
  - **21.2.2「拡張データフレーム」**の SRR ビットの論理値を「0」から「1」に更新した。
  - **21.3.6「ECAN 制御 / エラーカウンタ レジスタ」**の WIN ビットへの言及を削除した。
  - **21.10.2「障害隔離」**の「CiERR」を「CiEC」に変更した。
  - **21.10.2.4「エラー警告状態でのトランスミッタ」**の送信エラーカウンタの範囲を更新した。
  - **21.10.2.5「エラー警告状態でのレシーバ」**の受信エラーカウンタの範囲を更新した。
  - **21.12.1「スリープモード」**の TXCAN を CiTX に変更した。
  - **21.12.3「ウェイクアップ機能」**を全面的に改訂した。
- 表現と体裁の変更等、本書全体の細部を修正した。

ISBN: 978-1-60932-870-2