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 17. UART

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。

Note: ファミリ リファレンス マニュアルの本セクションは、デバイス データシートの補足を目的としています。本セクションの内容は dsPIC33F/PIC24H ファミリの一部のデバイスには対応していません。

本書の内容がお客様のご使用になるデバイスに対応しているかどうかは、最新デバイス データシート内の「**UART**」の冒頭に記載している注意書きでご確認ください。

デバイス データシートとファミリ リファレンス マニュアルの各セクションは、マイクロチップ社のウェブサイト (http://www.microchip.com) からダウンロードできます。

17.1 はじめに

汎用非同期送受信 (UART) モジュールは、dsPIC33F/PIC24H ファミリのデバイスで利用可能なシリアル I/O の 1 つです。UART は RS-232、RS-485、LIN、IrDA® 等のプロトコルを使用して周辺デバイスやパーソナル コンピュータと通信する全二重方式の非同期通信チャンネルです。このモジュールは UxCTS および UxRTS ピンを使用するハードウェア フロー制御オプションもサポートし、さらに IrDA エンコーダ / デコーダも備えます。

UART モジュールの主な特長を以下に挙げます。

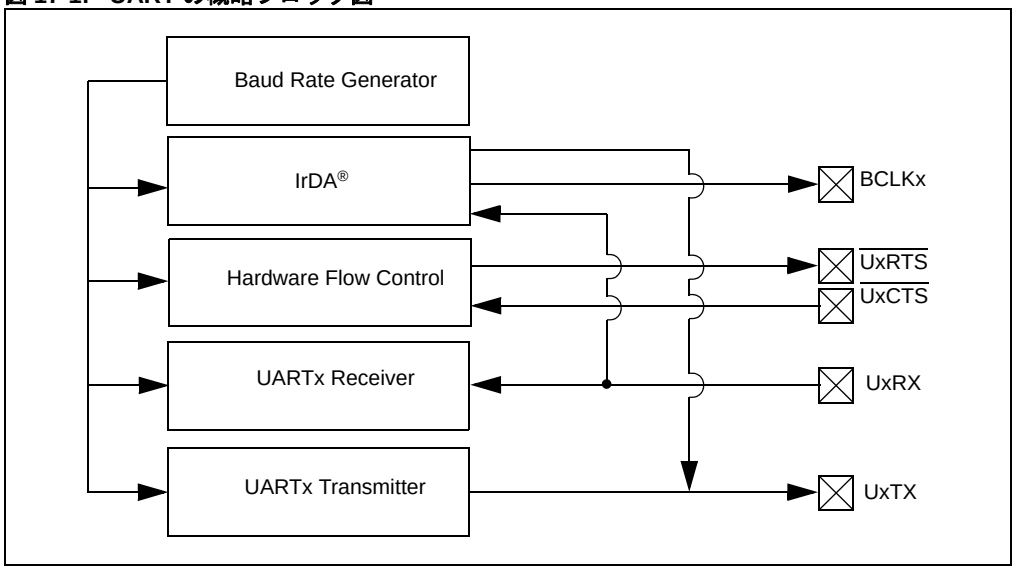
- UxTX および UxRX ピンを介する全二重 8 ビットまたは 9 ビット データ転送
- 偶数パリティ、奇数パリティ、パリティなしを選択可能 (8 ビットデータの場合)
- 1 個また 2 個のストップビット
- ハードウェアによる baud レート自動検出機能
- UxCTS および UxRTS ピンを使用するハードウェア フロー制御オプション
- 16 ビット プリスケーラ付き baud レート ジェネレータ (BRG) を内蔵
- 40 MIPS で 10 Mbps ～ 38 bps の baud レート
- 4 段の先入れ先出し (FIFO) 送信データバッファ
- 4 段の先入れ先出し (FIFO) 受信データバッファ
- パリティエラー、フレーミング エラー、バッファ オーバーラン エラーの検出
- アドレス検出を使用する 9 ビットモードをサポート (第 9 ビット = 1)
- 送信および受信割り込み
- 診断用ループバック モードをサポート
- IrDA エンコーダ / デコーダ ロジック
- LIN バスをサポート
- 外部 IrDA エンコーダ / デコーダをサポートするための 16x baud クロック出力

Note: dsPIC33F/PIC24H ファミリは 1 つまたは複数の UART モジュールを備えます。ピン、制御 / ステータスビット、レジスタの名前で使用する添え字「x」は UART モジュールの番号を表します。詳細は各デバイスのデータシートを参照してください。

UART の概略ブロック図を図 17-1 に示します。UART モジュールは下記の主要ハードウェアエレメントで構成されます。

- baud レート ジェネレータ
- 非同期トランスミッタ
- 非同期レシーバ

図 17-1: UART の概略ブロック図



17.2 制御レジスタ

以下では、UART モジュールの動作を制御する各レジスタの機能について説明します。

- **UxMODE: UARTx モードレジスタ**
  - UART モジュールの有効化 / 無効化
  - IrDA エンコーダ / デコーダの有効化 / 無効化
  - WAKE、ABAUD、ループバック機能の有効化 / 無効化
  - $\overline{\text{UxRTS}}$ および $\overline{\text{UxCTS}}$ ピンの有効化 / 無効化
  - UxRTS ピンの動作モードの設定
  - UxRx ピン極性の設定
  - baud レート タイプの選択
  - データビット数、パリティ、ストップビット数の選択
- **UxSTA: UARTx ステータス / 制御レジスタ**
  - 送信割り込みモードの選択
  - 受信割り込みモードの選択
  - UART 転送の有効化 / 無効化
  - アドレス検出モードの制御
  - 送信および受信バッファの状態、パリティエラー、フレーミングエラー、オーバーフロー エラー等、各種ステータスの表示
- **UxRXREG: UARTx 受信レジスタ**
  - 受信データを格納
- **UxTXREG: UARTx 送信レジスタ (書き込み専用)**
  - 送信データを格納
- **UxBRG: UARTx baud レートレジスタ**
  - 送信または受信データの baud レート値を格納

レジスタ 17-1: UxMODE: UARTx モードレジスタ

R/W-0	U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
UARTEN	—	USIDL	IREN(1)	RTSMD	—	UEN<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
WAKE	LPBACK	ABAUD	URXINV	BRGH	PDSEL<1:0>		STSEL
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 **UARTEN:** UARTx イネーブルビット
1 = UARTx を有効にする (UARTx ピンは UEN<1:0> および UTXEN 制御ビットの定義に従って UARTx によって制御される)
0 = UARTx を無効にする (UARTx ピンは対応する PORT、LAT、TRIS ビットによって制御される)

bit 14 **予約**

bit 13 **USIDL:** 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した時に動作を停止する
0 = アイドルモード中も動作を継続する

bit 12 **IREN:** IrDA エンコーダ / デコーダ イネーブルビット (1)
1 = IrDA エンコーダ / デコーダを有効にする
0 = IrDA エンコーダ / デコーダを無効にする

bit 11 **RTSMD:** $\overline{\text{UxRTS}}$ ピンモード選択ビット
1 = $\overline{\text{UxRTS}}$ を片方向モードにする
0 = $\overline{\text{UxRTS}}$ をフロー制御モードにする

bit 10 **予約**

bit 9-8 **UEN<1:0>:** UARTx イネーブルビット
11 = UxTX、UxRX、BCLKx ピンを有効化して使用する ($\overline{\text{UxCTS}}$ ピンをポートラッチで制御する)
10 = UxTX、UxRX、$\overline{\text{UxCTS}}$、UxRTS ピンを有効化して使用する
01 = UxTX、UxRX、UxRTS ピンを有効化して使用する ($\overline{\text{UxCTS}}$ ピンをポートラッチで制御する)
00 = UxTX および UxRX ピンを有効化して使用する ($\overline{\text{UxCTS}}$、$\overline{\text{UxRTS}}$、BCLKx ピンをポートラッチで制御する)

bit 7 **WAKE:** スリープモード中スタートビット検出時ウェイクアップ イネーブルビット
1 = ウェイクアップを有効にする
0 = ウェイクアップを無効にする

bit 6 **LPBACK:** UARTx ループバック モード選択ビット
1 = ループバック モードを有効にする
0 = ループバック モードを無効にする

bit 5 **ABAUD:** baud レート自動検出イネーブルビット
1 = 次のキャラクタで baud レート自動検出を有効にする (同期フィールド (0x55) を受信する必要があります。このビットは完了時にハードウェアでクリアされます)
0 = baud レート自動検出を無効にする、または自動検出は完了した

bit 4 **URXINV:** 受信極性反転ビット
1 = UxRX のアイドル状態は「0」
0 = UxRX のアイドル状態は「1」

Note 1: この機能は低速モード (BRGH = 0) でのみ利用できます。詳細は各デバイスのデータシートを参照してください。

レジスタ 17-1: UxMODE: UARTx モードレジスタ (続き)

bit 3	**BRGH:** 高速 baud レート選択ビット 1 = 高速モード 0 = 低速モード
bit 2-1	**PDSEL<1:0>:** パリティ / データ選択ビット 11 = 9 ビットデータ、パリティなし 10 = 8 ビットデータ、奇数パリティ 01 = 8 ビットデータ、偶数パリティ 00 = 8 ビットデータ、パリティなし
bit 0	**STSEL:** ストップビット選択ビット 1 = 2 個のストップビット 0 = 1 個のストップビット

Note 1: この機能は低速モード (BRGH = 0) でのみ利用できます。詳細は各デバイスのデータシートを参照してください。

レジスタ 17-2: UxSTA: UARTx ステータス / 制御レジスタ

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R-0	R-1
UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/C-0	R-0
URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7							bit 0

凡例 :	C = クリア可能ビット		
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 **UTXISEL<1:0>** 送信割り込みモード選択ビット

11 =予約
10 =キャラクタを送信シフトレジスタへ転送して送信バッファがエンプティになった時に割り込みを生成する
01 =最終キャラクタを送信シフトレジスタからシフトアウトして全ての送信動作を完了した時に割り込みを生成する
00 =キャラクタを送信シフトレジスタへ転送するたびに(送信バッファ内の少なくとも1箇所がエンプティになるたびに) 割り込みを生成する

bit 14 **UTXINV:** 送信極性反転ビット

IREN = 0 の場合 :
1 = UxTX のアイドル状態は「0」
0 = UxTX のアイドル状態は「1」
IREN = 1 の場合 :
1 = IrDA エンコードされた UxTX のアイドル状態は「1」
0 = IrDA エンコードされた UxTX のアイドル状態は「0」

bit 12 **未実装 :**「0」として読み出し

bit 11 **UTXBRK:** 送信ブレークビット

1 = トランスミッタの状態に関係なく UxTX ピンを LOW に駆動する (同期ブレーク送信 ースタートビットに続いて 12 個の「0」と 1 個のストップビットを送信)
0 = 同期ブレーク送信は無効または完了した

bit 10 **UTXEN:** 送信イネーブルビット

1 = UARTx トランスミッタを有効にする (UARTEN = 1 の時に UARTx が UxTX ピンを制御する)
0 = UARTx トランスミッタを無効にする (保留中の全ての送信を中止してバッファをリセットする ; PORT が UxTX ピンを制御する)

bit 9 **UTXBF:** 送信バッファフル ステータスビット (読み出し専用)

1 = 送信バッファはフル
0 = 送信バッファはフルではない (少なくとも 1 つのワードデータを書き込み可能)

bit 8 TRMT**:** 送信シフトレジスタ エンプティビット (読み出し専用)

1 = 送信シフトレジスタと送信バッファはエンプティ (直前の送信は完了した)
0 = 送信シフトレジスタはエンプティではない(送信中または送信バッファ内のデータが送信待ち中)

bit 7-6 **URXISEL<1:0>:** 受信割り込みモード選択ビット

11 = 受信バッファがフルの時 (4 個のデータキャラクタを格納した時) に割り込みフラグビットをセットする
10 = 受信バッファが 3 個のデータキャラクタを格納した時に割り込みフラグビットをセットする
0x = 1 キャラクタを受信した時に割り込みフラグビットをセットする

bit 5 **ADDEN:** アドレス キャラクタ検出ビット (受信データの bit 8 = 1 を検出)

1 = アドレス検出モードを有効にする (このビットは 9 ビットモードを選択した場合にのみ効果を持ちます)
0 = アドレス検出モードを無効にする

レジスタ 17-2: UxSTA: UARTx ステータス / 制御レジスタ (続き)

bit 4 **RIDLE:** レシーバアイドル ビット (読み出し専用)
1 = レシーバはアイドル中
0 = レシーバはデータを受信中

bit 3 **PERR:** パリティエラー ステータスビット (読み出し専用)
1 = 現在のキャラクタでパリティエラーを検出した
0 = パリティエラーは検出されていない

bit 2 **FERR:** フレーミング エラー ステータスビット (読み出し専用)
1 = 現在のキャラクタでフレーミング エラーを検出した
0 = フレーミング エラーは検出されていない

bit 1 **OERR:** 受信バッファ オーバーラン エラー ステータスビット (クリアと読み出しのみ可能)
1 = 受信バッファはオーバーフローした
0 = 受信バッファはオーバーフローしていない(セットされたOERRビットをクリアすると、受信バッファと RSR はエンプティ状態にリセットされます)

bit 0 **URXDA:** 受信バッファ ステータスビット (読み出し専用)
1 = 受信バッファにデータが存在する (少なくとも 1 個のデータを読み出し可能)
0 = 受信バッファはエンプティ

レジスタ 17-3: UxRXREG: UARTx 受信レジスタ

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R-0
—	—	—	—	—	—	—	URX8
bit 15							bit 8

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
URX<7:0>							
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 15-9 **未実装 :**「0」として読み出し
bit 8 URX8**:** 受信キャラクタの bit 8 (9 ビットモード時)
bit 7-0 **URX<7:0>:** 受信キャラクタの bit 7 ～ 0

レジスタ 17-4: UxTXREG: UARTx 送信レジスタ (書き込み専用)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	W-x
—	—	—	—	—	—	—	UTX8
bit 15							bit 8

W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
UTX<7:0>							
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 15-9 **未実装 :**「0」として読み出し
bit 8 UTX8**:** 送信キャラクタの bit 8 (9 ビットモード時)
bit 7-0 **UTX<7:0>:** 送信キャラクタの bit 7 ～ 0

レジスタ 17-5: UxBRG: UARTx baud レートレジスタ

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	W-0
BRG<15:8>							
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
BRG<7:0>							
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 15-0 BRG<15:0>**:** baud レート分周ビット

17.3 UART BAUD レート ジェネレータ

UART モジュールは専用の 16 ビット baud レート ジェネレータ (BRG) を備えます。UxBRG レジスタはフリーランニング 16 ビットタイマの周期を制御します。式 17-1 に、BRGH = 0 の時の baud レートの計算式を示します。

式 17-1: UART baud レート (BRGH = 0)

$$baudレート = \frac{F_{CY}}{16 \times (UxBRG + 1)} \quad \text{.....(1)}$$

$$UxBRG = \frac{F_{CY}}{16 \times baudレート} - 1 \quad \text{.....(2)}$$

Note: F_{CY} は命令サイクルクロック周波数 (FOSC/2) です。

例 17-1 に、下記条件における baud レート誤差の計算式を示します。

- FCY = 4 MHz
- 目標 baud レート = 9600

例 17-1: baud レート誤差の計算 (BRGH = 0)

$$目標baudレート = \frac{F_{CY}}{16 \times (UxBRG + 1)} \quad \text{.....(1)}$$

UxBRG 値を求める

$$UxBRG = \frac{F_{CY} / 目標baudレート}{16} - 1$$

$$= \left(\frac{4000000/9600}{16} - 1\right)$$

$$= 25$$

$$計算baudレート = \frac{4000000}{16 \times (25 + 1)}$$

$$= 9615$$

$$誤差 = \frac{計算baudレート - 目標baudレート}{目標baudレート} \quad \text{.....(2)}$$

$$= \frac{9615 - 9600}{9600}$$

$$= 0.16\%$$

可能な最大 baud レート (BRGH = 0) は FCY/16 (UxBRG = 0 の時)、可能な最小 baud レートは FCY/(16 * 65536) です。

式 17-2 に、BRGH = 1 の時の baud レートの計算式を示します。

式 17-2: UART baud レート (BRGH = 1)

$$baudレート = \frac{F_{CY}}{4 \times (UxBRG + 1)} \quad .....(1)$$

$$UxBRG = \frac{F_{CY}}{4 \times baudレート} - 1 \quad .....(2)$$

Note: F_{CY} は命令サイクルクロック周波数です。

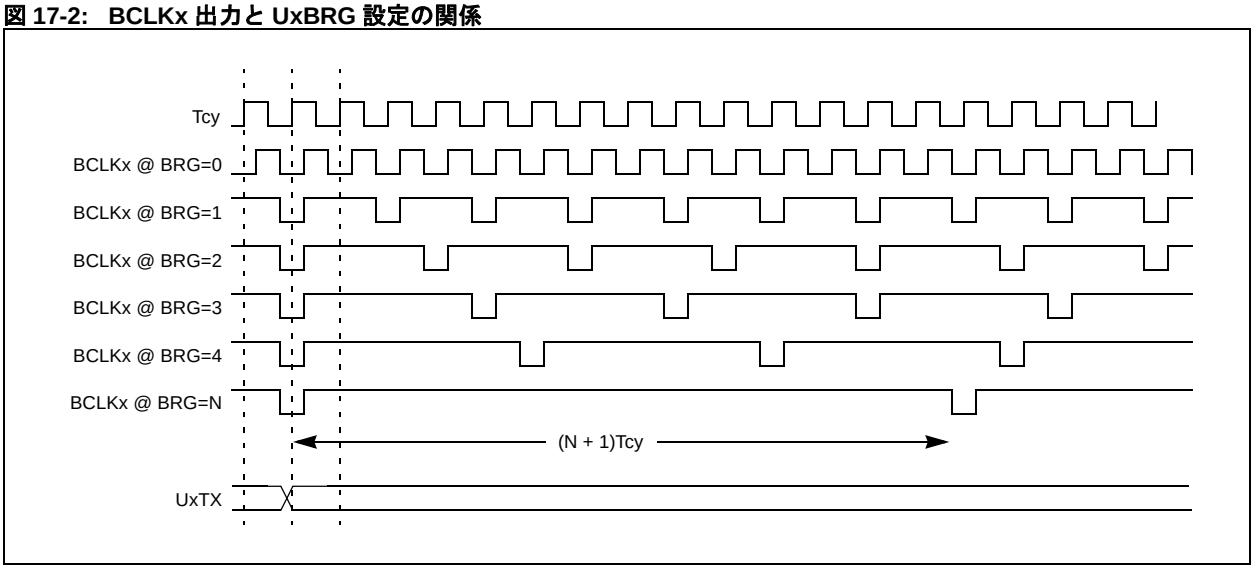
可能な最大 baud レート (BRGH = 1) は FCY/4 (UxBRG = 0 の時)、可能な最小 baud レートは FCY/(4 * 65536) です。

UxBRG レジスタに新たな値を書き込むと、BRG タイマがリセット (クリア) されます。これにより、BRG はタイマのオーバーフローを待たずに新たな baud レートの生成を開始します。

17.3.1 BCLKx 出力

UART と BCLKx 出力が有効 (UEN<1:0> = 11) な場合、BCLKx ピンは 16x baud クロックを出力します。この機能は、外部の IrDA エンコーダ / デコーダをサポートするために使用します (図 17-2 参照)。スリープモード中、BCLKx 出力は HIGH 状態を維持します。UART が (UEN<1:0> = 11) モードを維持する限り、PORTx および TRISx ラッチビットの状態に関係なく、BCLKx は常に出力として機能します。

図 17-2: BCLKx 出力と UxBRG 設定の関係



17.4 UART のコンフィグレーション

UART は標準の非ゼロ復帰 (NRZ) フォーマット (1 x スタートビット、8 または 9 x データビット、1 または 2 x ストップビット) を使用します。パリティはハードウェアでサポートされ、ユーザ アプリケーションで偶数パリティ、奇数パリティ、パリティなしのいずれかに設定できます。POR 時の既定値設定は、最も一般的なデータ フォーマット「8 ビット、パリティなし、1 個のストップビット」(8, N, 1) です。データビット数、スタートビット数、パリティの指定には PDSEL<1:0> ビット (UxMODE<2:1>) と STSEL ビット (UxMODE<0>) を使用します。内蔵の専用 16 ビット BRG を使用すると、オシレータから標準的周波数の baud レートを生成できます。UART は最下位ビット (LSb) を先頭に送受信を行います。UART モジュールのトランスミッタとレシーバは機能的に独立していますが、同一のデータ フォーマットと baud レートを使用します。

17.4.1 UART の有効化

UART モジュールは、UARTEN ビット (UxMODE<15>) と UTXEN ビット (UxSTA<10>) をセットする事により有効になります。モジュールを有効にすると、UxTX および UxRX ピンはそれぞれ出力および入力として設定されます (これは対応する I/O ポートピンの TRIS および PORT レジスタビットの設定よりも優先されます)。送信実行中ではない時の UxTX ピンの論理状態は「1」です。

Note:	先に UARTEN ビットをセットしてから UTXEN ビットをセットする必要があります。この順番を守らないと UART 送信は有効になりません。

17.4.2 UART の無効化

UART モジュールを無効化するには、UARTEN ビット (UxMODE<15>) をクリアします。これは全てのリセット後の既定値状態です。UART を無効にすると、全ての UART ピンは対応する PORT および TRIS ビットの設定に従ってポートピンとして動作します。

UART モジュールを無効化するとバッファはエンプティ状態にリセットされます。バッファ内の全てのデータキャラクタは失われ、baud レートカウンタはリセットされます。

UART モジュールの無効化により、このモジュールに関連する全てのエラーフラグとステータスフラグはリセットされます。また URXDA、OERR、FERR、PERR、UTXEN、UTXBRK、UTXBF ビットはクリアされ、RIDLE および TRMT ビットはセットされます。ADDEN、URXISEL<1:0>、UTXISEL<1:0> を含むその他の制御ビットと UxMODE および UxBRG レジスタはリセットの影響を受けません。

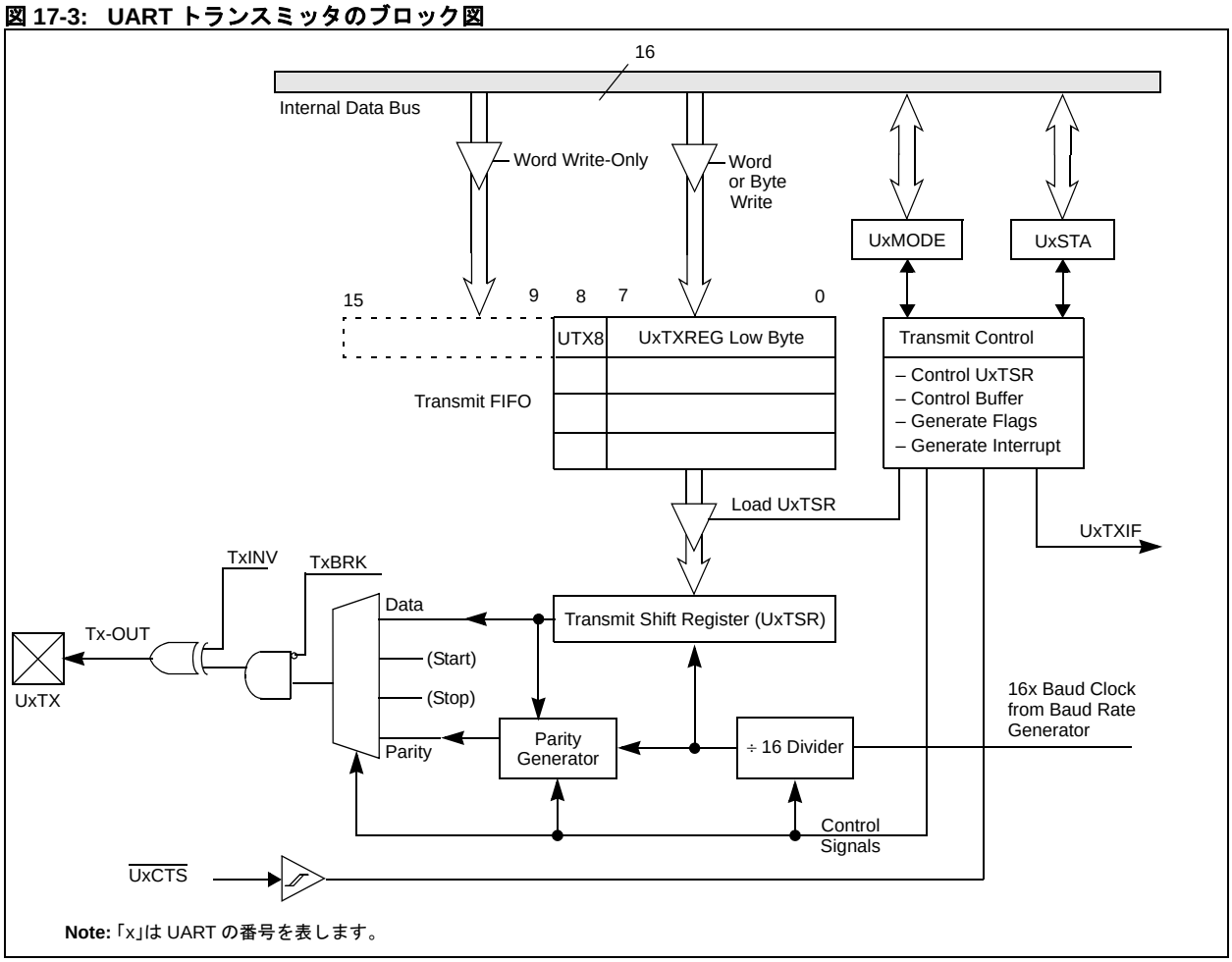
UART が動作している時に UARTEN ビットをクリアすると、保留中の送受信は全て中止され、モジュールは上記の状態にリセットされます。UART を再度有効にすると、モジュールは以前と同じコンフィグレーションで再起動します。

17.5 UART トランスミッタ

UART トランスミッタのブロック図を図 17-3 に示します。トランスミッタの中核となるのが送信シフトレジスタ (UxTSR) です。このシフトレジスタには送信 FIFO バッファ (UxTXREG) からデータが転送されます。UxTXREG レジスタにはソフトウェアでデータを書き込みます。UxTSR レジスタ内の既存データのストップビットが送信されるまで、UxTSR レジスタに新しいデータは転送されません。ストップビットが送信されると即座に UxTXREG レジスタから UxTSR レジスタに新しいデータが転送されます (送信すべきデータが存在する場合)。

Note:	UxTSR レジスタはデータメモリ内に配置されないため、ユーザ アプリケーションからアクセスする事はできません。

図 17-3: UART トランスミッタのブロック図



Note: 「x」は UART の番号を表します。

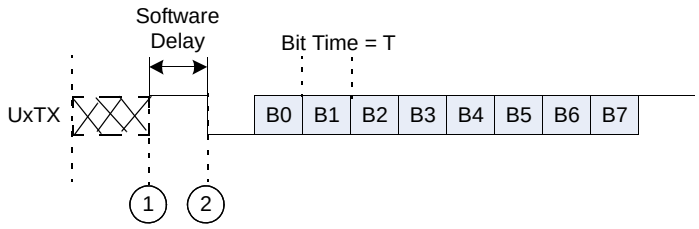
UTXEN イネーブルビット (UxSTA<10>) をセットすると送信が有効化されます。UxTXREG レジスタにデータが書き込まれ、かつ baud レート ジェネレータ (UxBRG) がシフトクロックを生成するまで送信は開始されません (図 17-3 参照)。UxTXREG にデータを転送すると即座に UxTSR に転送されるため、通常、送信開始時に UxTSR レジスタはエンプティです。送信中に UTXEN ビットをクリアすると送信は中止され、トランスミッタはリセットされます。この結果 UxTX ピンは高インピーダンス状態に戻ります。

9 ビット送信を選択するには、PDSEL<1:0> ビットを「11」に設定し、かつ第 9 ビットを UTX8 ビット (UxTXREG<8>) に書き込む必要があります。9 ビットを一度に書き込むために、UxTXREG にはワード書き込みを実行する必要があります。

Note:	9 ビットデータ送信の場合パリティはありません。

デバイスリセット時に UxTX ピンは入力として設定されるため、UxTX ピンのリセット状態は未知です。UART モジュールを有効化した時、送信ピンは HIGH に駆動され、データが送信バッファ (UxTXREG) に書き込まれるまで HIGH 状態を維持します。先頭データを UxTXREG レジスタに書き込むと、送信ピンは即座に LOW に駆動されます。スタートビットの検出を確実にするために、UARTx を有効化 (UARTEN = 1) してから送信を開始するまでに一定の遅延時間を設ける事を推奨します。この遅延時間は baud レートによって異なり、1 データビットの送信に要する時間以上を確保する必要があります。

図 17-4: UART 送信



1. UART モジュールを有効化 (UARTEN = 1)
2. データを送信バッファ (UxTXREG) に書き込んで送信を開始

17.5.1 送信バッファ (UxTXREG)

送信バッファは 9 ビット幅の 4 段バッファです。送信シフトレジスタ (UxTSR) と合わせて 5 段バッファとして使用できます。このバッファは先入れ先出し (FIFO) バッファとして構成されます。UxTXREG の内容を UxTSR レジスタに転送すると、そのバッファ位置に新しいデータを書き込む事が可能となります。また、次のバッファ位置が次回の UxTSR レジスタへの転送元となります。バッファがフルになると UTXBF (UxSTA<9>) ステータスビットがセットされます。ユーザ アプリケーションがフル状態のバッファに書き込みを試みた場合、FIFO はそのデータを受け入れません。

FIFO はデバイスリセット時に常にリセットされますが、デバイスが省電力モードに移行した時あるいは省電力モードからウェイクアップした時には影響を受けません。

17.5.2 送信割り込み

送信割り込みフラグ (UxTXIF) は、対応する割り込みフラグステータス レジスタ (IFS) に格納されています。UTXISEL<1:0> 制御ビット (UxSTA<15,13>) は、UART が送信割り込みを生成するタイミングを指定します。

- UTXISEL<1:0> = 00 の場合、送信バッファから送信シフトレジスタ (UxTSR) に 1 キャラクタを転送した時に UxTXIF がセットされます。これは送信バッファ内の少なくとも 1 箇所がエンプティである事を意味します。
- UTXISEL<1:0> = 01 の場合、最終キャラクタを UxTSR レジスタから送出した時に UxTXIF がセットされます。これは送信動作が全て完了した事を意味します。
- UTXISEL<1:0> = 10 の場合、UxTSR レジスタにキャラクタを転送して送信バッファがエンプティになった時に UxTIF がセットされます。

UxTXIF ビットはモジュールが有効化された時にセットされます。ユーザ アプリケーションは割り込みサービスルーチン (ISR) で UxTXIF ビットをクリアする必要があります。

これら 3 つの割り込みモードは動作中に切り換える事ができます。

Note:	UTXISEL<1:0> = 00 の場合、UTXEN ビットをセットした時に UxTXIF フラグビットもセットされます。これは、その時点で送信バッファがフルではない (送信データを UxTXREG レジスタに転送できる) ためです。

UxTXIF フラグビットは UxTXREG レジスタのステータスを示し、TRMT ビット (UxSTA<8>) は UxTSR のステータスを示します。TRMT ステータスビットは UxTSR がエンプティの時にセットされる読み出し専用ビットです。このビットには割り込みロジックは関連付けられていません。従ってユーザ アプリケーションはこのビットをポーリングする事によって UxTSR がエンプティであるかどうかを判定する必要があります。

17.5.3 UART 送信のセットアップ

送信のセットアップ手順は下記の通りです。

1. UxBRG レジスタを適切な baud レートに初期化する (**17.3「UART baud レート ジェネレータ」**参照)
2. PDSEL<1:0>(UxMODE<2:1>) および STSEL (UxMODE<0>) ビットでデータビット数、ストップビット数、パリティを設定する
3. 送信割り込みが必要な場合、対応する割り込みイネーブル制御レジスタ (IEC) の UxTXIE 制御ビットをセットする
   - UTXISEL<1:0> (UxSTA<15,13>) ビットで送信割り込みモードを選択する
   - 対応する割り込み優先度制御レジスタ (IPC) の UxTXIP<2:0> 制御ビットで送信割り込みの優先度を指定する
4. UARTEN ビット (UxMODE<15>) をセットして UART モジュールを有効にする
5. UTXEN ビット (UxSTA<10>) をセットして送信を有効にする (UxTXIF ビットもセットされる)

   UxTXIF ビットは、UART 送信割り込みをサービスするソフトウェア ルーチンでクリアする必要があります。UxTXIF ビットの動作は UTXISEL<1:0> 制御ビットにより制御されます。

6. UxTXREG レジスタにデータを書き込む (送信を開始)

   9 ビット送信を選択した場合、ワード書き込みを行います。8 ビット送信を選択した場合、バイト書き込みを行います。UTXBF ステータスビット (UxSTA<9>) がセットされるまでデータをバッファに書き込む事ができます。

> **Note:** 先に UARTEN ビットをセットしてから UTXEN ビットをセットする必要があります。この順番を守らないと UART 送信は有効になりません。

例 17-2 に、UART を送信用にセットアップするサンプルコードを示します。

図 17-5: 送信 (8 ビットまたは 9 ビットデータ)

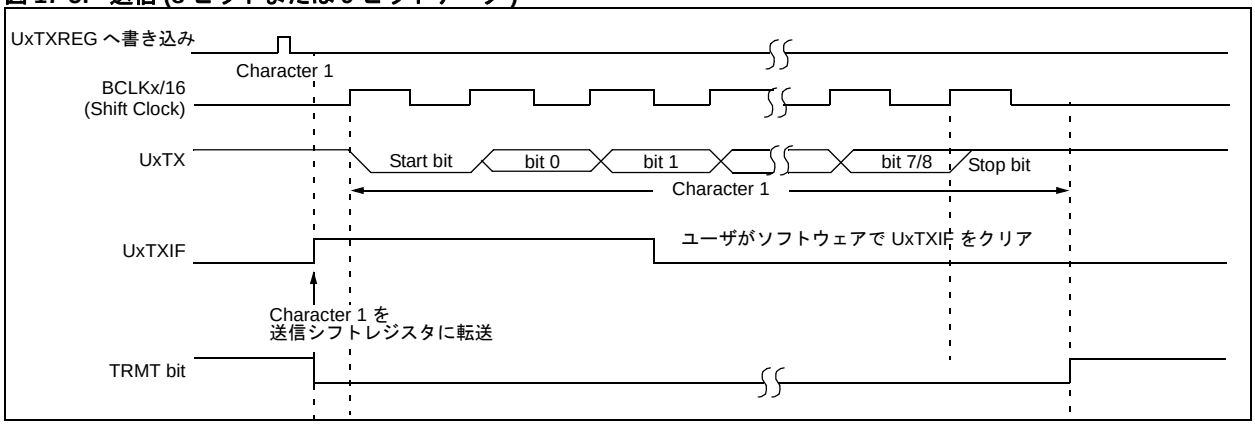
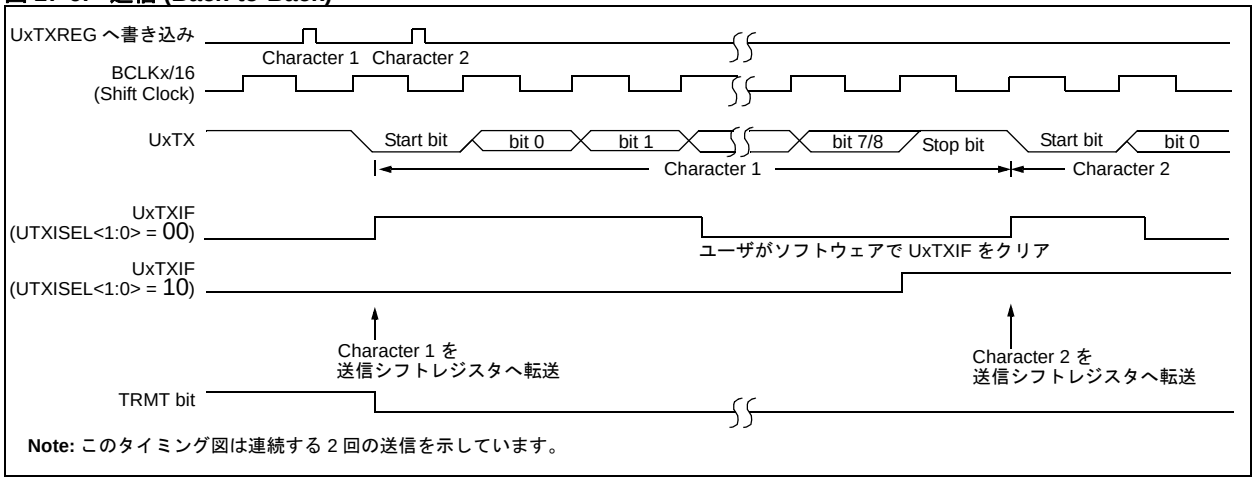


図 17-6: 送信 (Back-to-Back)



Note: このタイミング図は連続する 2 回の送信を示しています。

例 17-2: 割り込みを生成する UART 送信

```
#define FCY 40000000
#define BAUDRATE 9600
#define BRGVAL ((FCY/BAUDRATE)/16)-1
unsigned int i;

int main(void)
{

// Configure Oscillator to operate the device at 40 MHz
// Fosc = Fin * M/(N1 * N2), Fcy = Fosc/2
// Fosc = 8M * 40(2 * 2) = 80 MHz for 8M input clock
 PLLFBD = 38;// M = 40
 CLKDIVbits.PLLPOST = 0;             // N1 = 2
 CLKDIVbits.PLLPRE = 0;              // N2 = 2
 OSCTUN = 0;                         // Tune FRC oscillator, if FRC is used
 RCONbits.SWDTEN = 0;                // Disable Watch Dog Timer

 while(OSCCONbits.LOCK!= 1) {};      // Wait for PLL to lock

 U1MODEbits.STSEL = 0;               // 1 Stop bit
 U1MODEbits.PDSEL = 0;               // No Parity, 8 data bits
 U1MODEbits.ABAUD = 0;               // Auto-Baud Disabled
 U1MODEbits.BRGH = 0;                // Low Speed mode

 U1BRG = BRGVAL;                     // BAUD Rate Setting for 9600

 U1STAbits.UTXISEL0 = 0;             // Interrupt after one TX Character is transmitted
 U1STAbits.UTXISEL1 = 0;

IEC0bits.U1TXIE = 1;                 // Enable UART TX Interrupt

 U1MODEbits.UARTEN = 1;              // Enable UART
 U1STAbits.UTXEN = 1;                // Enable UART TX

 /* wait at least 104 usec (1/9600) before sending first char */
 for(i = 0; i < 4160; i++)
 {
  Nop();
 }

U1TXREG = 'a';                       // Transmit one character

 while(1)
 {
 }
}

void __attribute__((__interrupt__)) _U1TXInterrupt(void)
{
 IFS0bits.U1TXIF = 0;                // clear TX interrupt flag
 U1TXREG = 'a';                      // Transmit one character
}
```

17.5.4 ブレーク キャラクタの送信

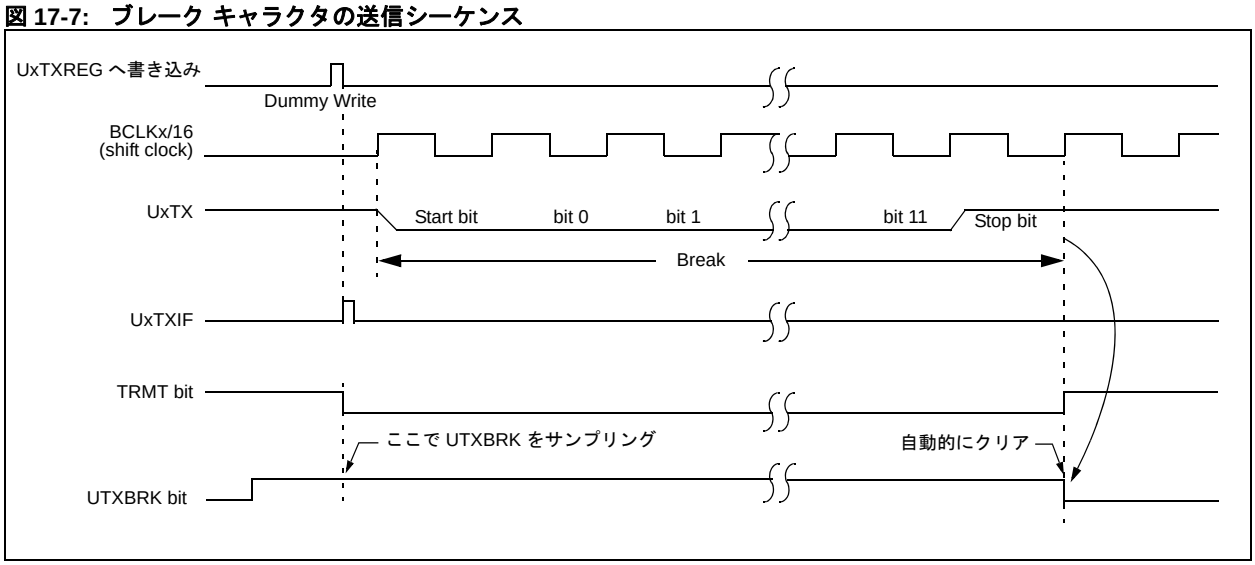
ブレーク キャラクタは 1 個のスタートビットを先頭に 12 ビットの「0」と 1 個のストップビットで構成されます。フレーム ブレーク キャラクタは、UTXBRK および UTXEN ビットをセットした状態で送信シフトレジスタにデータを書き込んだ時に送信されます。ブレーク キャラクタの送信を開始するには、UxTXREG レジスタに対する 1 回のダミー書き込みが必要です。ブレーク キャラクタ用に UxTXREG レジスタに書き込まれたデータ値は無視されます。この書き込みは、正しいシーケンスを開始する事を目的とします (全て「0」のデータを送信)。

対応するストップビットが送信されると、UTXBRK ビットはハードウェアによってリセットされます。これにより、ユーザ アプリケーションは、ブレーク キャラクタの次に送信するバイト (通常は LIN 仕様の同期キャラクタ) を送信 FIFO にプリロードできます。

Note:	ユーザ アプリケーションは、UTXBRK ビットをセットする前に、トランスミッタがアイドル (TRMT = 1) になるまで待機する必要があります。UTXBRK ビットは、他の全てのトランスミッタ動作よりも優先されます。シーケンスが完了する前にユーザ アプリケーションが TXBRK ビットをクリアした場合、予期せぬモジュール挙動が発生します。ブレーク キャラクタの送信は送信割り込みを生成しません。

通常の送信と同様に、TRMT ビットは送信シフトレジスタがエンプティかどうかを示します。図 17-7 にブレーク キャラクタ送信シーケンスのタイミングを示します。

図 17-7: ブレーク キャラクタの送信シーケンス



17.5.4.1 ブレークおよび同期送信シーケンス

下記のシーケンスは、ブレークとこれに続く baud レート自動検出用同期バイトで構成されたメッセージフレーム ヘッダを送信します。これは LIN バスマスタに特有のシーケンスです。

1. UART を必要なモードに設定する
2. UTXEN および UTXBRK をセットする
3. UxTXREG にダミー キャラクタを書き込んで送信を開始する (この値は無視される)
4. UxTXREG に 0x55 を書き込む (送信 FIFO に同期キャラクタを書き込む)

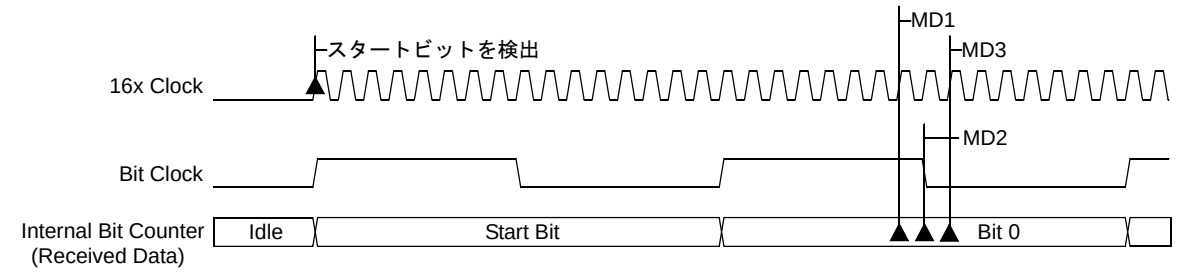
ブレーク送信後に、UTXBRK ビットがハードウェアによってリセットされ、同期キャラクタが送信されます。

17.6 データビットの検出

17.6.1 16x クロックモード (BRGH = 0)

16x クロックモードでは、受信データの各ビットは 16 クロックのパルス幅を持ちます。受信データビットの値を検出するために、クロックの 7、8、9 番目の立ち上がりエッジでビットをサンプリングします。これらの立ち上がりエッジは多数決検出エッジと呼ばれます。このモードは 4x クロックモードよりも信頼性に優れます。

図 17-8: 多数決検出を使用する 16x クロックモード

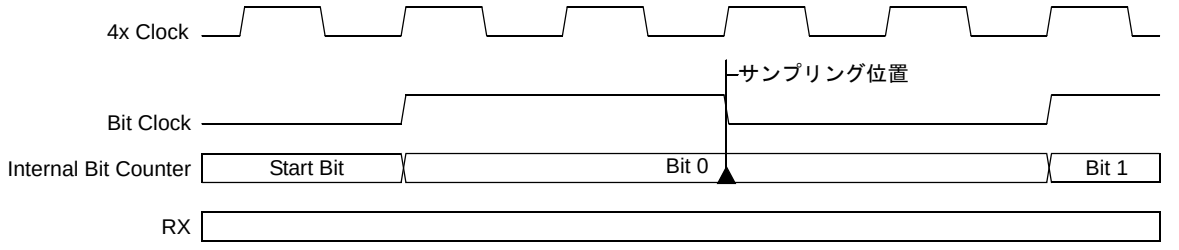


Note: 16x クロックモードでは、クロックの 7、8、9 番目の立ち上がりエッジで各ビットをサンプリングします。

17.6.2 4x クロックモード (BRGH = 1)

4x クロックモードでは、受信データの各ビットは 4 クロックのパルス幅を持ちます。4x クロックモードではエッジ数が少ないため、多数決検出法を適用しません。従って受信データを 1/2 ビット幅位置で 1 回だけサンプリングします。

図 17-9: 多数決検出を使用しない 4x クロックモード



Note: 4x クロックモードでは、1/2 ビット幅位置で 1 回だけサンプリングします。

.

17.7 UART 受信

レシーバのブロック図を図 17-10 に示します。レシーバの中核となるのが受信 (シリアル) シフト (UxRSR) レジスタです。UxRX ピンで受信したデータはデータ復元ブロックへ送られます。データ復元ブロックは baud レートの 16 倍で動作し、メインの受信シリアルシフタは baud レートで動作します。UxRX ピンでストップビットを検出すると、UxRSR 内の受信データを受信 FIFO へ転送します (FIFO がフルではない場合)。

Note: UxRSR レジスタはメモリ内に配置されないため、ユーザ アプリケーションからアクセスする事はできません。

多数決検出回路は、UxRX ピン上のデータを複数回サンプリングする事によって、UxRX ピンの状態 (HIGH または LOW レベル) を判定します。

17.7.1 受信バッファ (UxRXREG)

UART レシーバは 4 段の 9 ビット幅 FIFO 受信データバッファを備えます。UxRXREG レジスタはメモリ内に配置されるため、ユーザ アプリケーションは FIFO の出力にアクセスできます。4 ワードのデータを受信して FIFO に転送した後に 5 番目のワードを UxRSR レジスタにシフトし始めても、この時点ではバッファ オーバーランは発生しません。

17.7.2 レシーバエラーの対応

FIFO がフル (4 キャラクタを格納) の状態で 5 番目のキャラクタが完全に UxRSR レジスタに格納された時点で、オーバーラン エラーOERR ビット (UxSTA<1>) がセットされます。UxRSR 内のワードは保持されますが、OERR ビットがセットされている間の受信 FIFO への転送は禁止されます。後続データの受信を可能にするには、ソフトウェアで OERR ビットをクリアする必要があります。

オーバーラン前に受信したデータを保持する必要がある場合、ユーザ アプリケーションは 5 個のキャラクタを全て読み出してから OERR ビットをクリする必要があります。受信した 5 個のキャラクタを破棄してもよい場合、ユーザ アプリケーションはキャラクタを読み出さずに OERR ビットをクリアしてもかまいません。OERR ビットをクリアすると受信 FIFO がリセットされるため、それまでの受信データは全て失われます。

Note: 受信 FIFO 内のデータは、OERR ビットをクリアする前に読み出す必要があります。OERR ビットをクリアすると FIFO がリセットされるため、バッファ内の全てのデータが失われます。

論理 LOW レベルでストップビットを検出した場合、フレーミング エラービット FERR (UxSTA<2>) がセットされます。

バッファの先頭 (現在のワード) でデータワードにパリティエラーを検出した場合、パリティエラー ビット PERR (UxSTA<3>) がセットされます。例えば、偶数パリティを選択した場合、データ内の「1」の総数が奇数の時にパリティエラーが発生します。PERR ビットは 9 ビットモードでは効果を持ちません。FERR および PERR ビットは対応するワードと一緒にバッファリングされます。データワードを読み出す前に、これらのビットを読み出す必要があります。

OERR、FERR、PERR エラーのいずれかが生じると割り込みが発生します。割り込み発生時に対応する割り込みベクタ位置へ移動するために、ユーザ アプリケーションは対応する割り込みイネーブル制御ビット (IEC4<UxERIE>) 有効にする必要があります。

17.7.3 受信割り込み

UART 受信割り込みフラグ (UxRXIF) は、対応する割り込みフラグステータス レジスタ (IFS) に格納されています。URXISEL<1:0> (UxSTA<7:6>) 制御ビットは、UART レシーバが割り込みを生成するタイミングを決定します。

- URXISEL<1:0> = 00 または 01 の場合、1 データワードを受信シフト (UxRSR) レジスタから受信バッファに転送するたびに割り込みを生成します。受信バッファには 1 つまたは複数のキャラクタが既に格納されていてもかまいません。
- URXISEL<1:0> = 10 の場合、1 ワードを UxRSR レジスタから受信バッファに転送した結果として受信バッファ内のキャラクタ数が 3 または 4 になった時に割り込みを生成します
- URXISEL<1:0> = 11 の場合、1 ワードを UxRSR レジスタから受信バッファに転送した結果として受信バッファ内のキャラクタ数が 4(フル) になった時に割り込みを生成します。

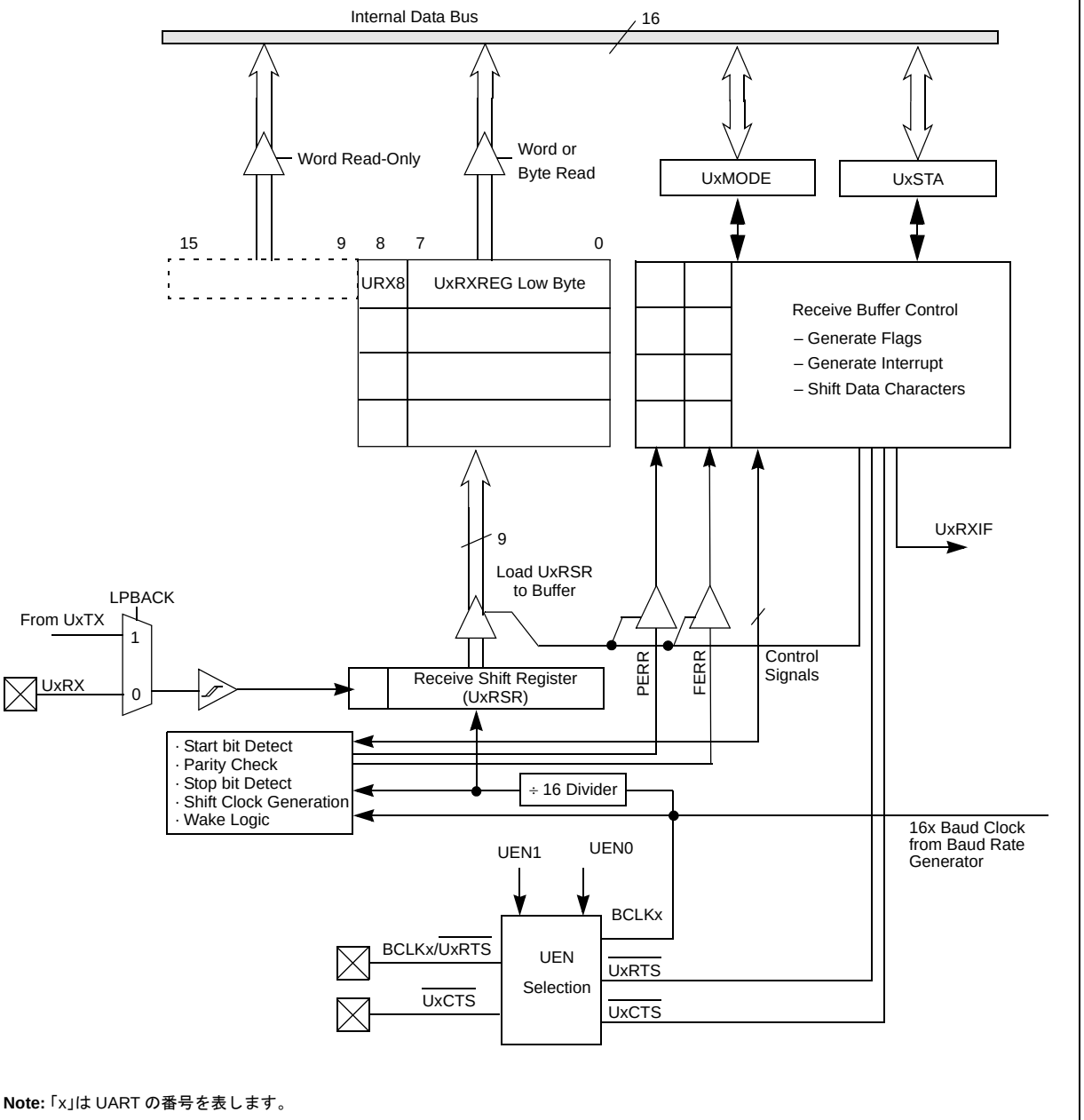
これら 3 つの割り込みモードは動作中に切り換える事ができます。

URXDA および UxRXIF フラグビットは UxRXREG レジスタのステータスを示し、RIDLE ビット (UxSTA<4>) は UxRSR レジスタのステータスを示します。RIDLE ステータスビットは、レシーバがアイドルモード(UxRSRレジスタがエンプティ)の時にセットされる読み出し専用ビットです。このビットには割り込みロジックは関連付けられていません。従ってユーザ アプリケーションはこのビットをポーリングする事によって UxRSR がアイドル中であるかどうかを判定する必要があります。

URXDA ビット (UxSTA<0>) は受信バッファがエンプティかどうかを示します。このビットは、受信バッファ内に読み出し可能なキャラクタが 1 つ以上存在していればセットされます。URXDA は読み出し専用ビットです。

図 17-10 に UART レシーバのブロック図を示します。

図 17-10: UART レシーバのブロック図



Note:「x」は UART の番号を表します。

17.7.4 UART 受信のセットアップ

受信のセットアップ手順は下記の通りです。

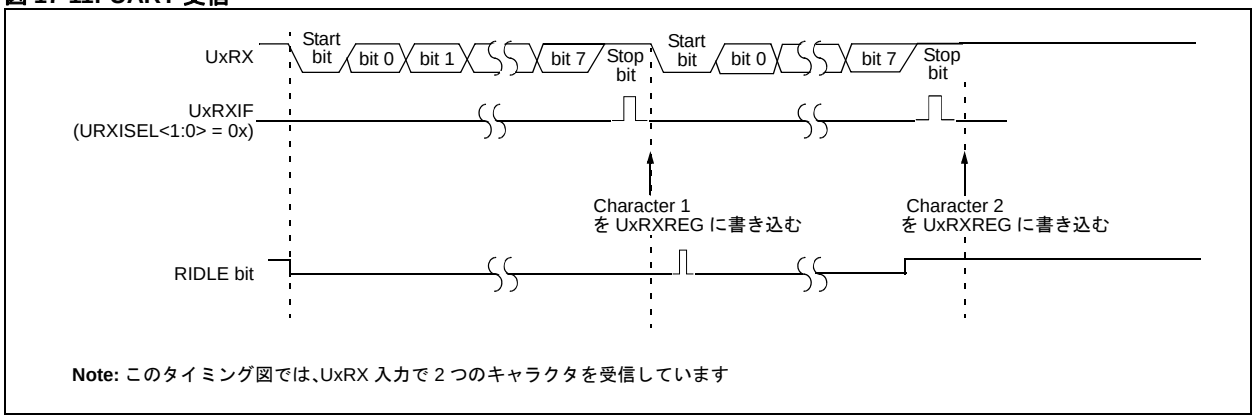
1. UxBRG レジスタを適切な baud レートに初期化する (**17.3「UART baud レート ジェネレータ」**参照)
2. PDSEL<1:0> ビット (UxMODE<2:1>) と STSEL (UxMODE<0>) ビットでデータビット数、スタートビット数、パリティを設定する
3. 割り込みが必要な場合、対応する割り込みイネーブル制御レジスタ (IEC) の UxRXIE ビットをセットする
   - URXISEL<1:0> ビット (UxSTA<7:6>) で受信割り込みモードを選択する
   - 対応する割り込み優先度制御レジスタ (IPC) の UxRXIP<2:0> 制御ビットで割り込みの優先度を指定する
4. UARTEN ビット (UxMODE<15>) をセットして UART モジュールを有効にする
5. 受信割り込みは URXISEL<1:0> 制御ビットの設定に影響されます。

   受信割り込みを有効にしない場合、ユーザ アプリケーションは URXDA ビットをポーリングできます。UxRXIF ビットは、UART 受信割り込みをサービスするソフトウェア ルーチンでクリアする必要があります。
6. 受信バッファからデータを読み出す

   9 ビット転送モードを選択した場合、ワード読み出しまたはバイト読み出しを行います。バッファ内にデータが存在する場合、URXDA ステータスビット (UxSTA<0>) がセットされます。

例 17-3 に、UART を受信用にセットアップするサンプルコードを示します。

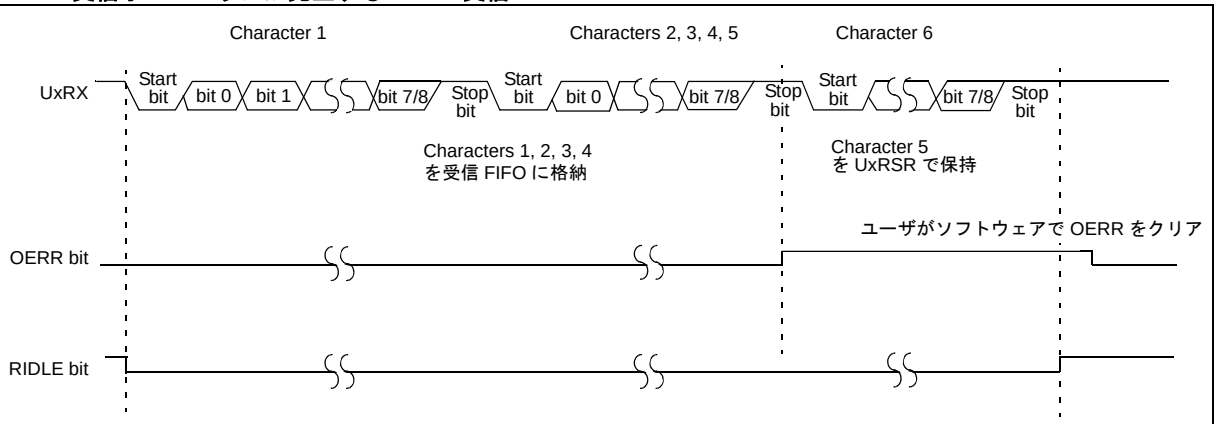
図 17-11: UART 受信



Note: このタイミング図では、UxRX 入力で 2 つのキャラクタを受信しています

Note:	アプリケーション ソフトウェアが割り込みフラグに従って UART 受信を無効化する場合、ソフトウェアは送信を無効化する前に 1 ビット分の時間を待機する必要があります。

図 17-12: 受信オーバーランが発生する UART 受信



Note: この図では、入力バッファを読み出す事なく 6 個のキャラクタを受信しようとしています。
第 5 キャラクタは受信シフトレジスタで保持されます。 第 6 キャラクタの受信開始時にオーバーランが発生します。

例 17-3: ポーリングによる UART 受信 (割り込みを無効にする場合)

```
#define FCY 40000000
#define BAUDRATE 9600
#define BRGVAL ((FCY/BAUDRATE)/16) - 1

int main(void)
{


// Configure Oscillator to operate the device at 40 MHz
// Fosc = Fin * M/(N1 * N2), Fcy = Fosc/2
// Fosc = 8M * 40(2 * 2) = 80 MHz for 8M input clock
 PLLFBD = 38;// M = 40
 CLKDIVbits.PLLPOST = 0;             // N1 = 2
 CLKDIVbits.PLLPRE = 0;              // N2 = 2
 OSCTUN = 0;                         // Tune FRC oscillator, if FRC is used

 RCONbits.SWDTEN = 0;                // Disable Watch Dog Timer

 while(OSCCONbits.LOCK!= 1) {};      // Wait for PLL to lock

 U1MODEbits.STSEL = 0;               // 1 Stop bit
 U1MODEbits.PDSEL = 0;               // No Parity, 8 data bits
 U1MODEbits.ABAUD = 0;               // Auto-Baud Disabled
 U1MODEbits.BRGH = 0;                // Low-Speed mode

 U1BRG = BRGVAL;                     // BAUD Rate Setting for 9600

 U1STAbits.URXISEL = 0;              // Interrupt after one RX character is received;

 U1MODEbits.UARTEN = 1;              // Enable UART

 while(1)
 {
   char ReceivedChar;

   /* check for receive errors */
   if(U1STAbits.FERR = 1)
   {
       continue;
   }

   /* must clear the overrun error to keep uart receiving */
   if(U1STAbits.OERR = 1)
   {
       U1STAbits.OERR = 0;
       continue;
   }

   /* get the data */
   if(U1STAbits.URXDA = 1)
   {
       ReceivedChar = U1RXREG;
   }
 }
```

17.8 UART による 9 ビット通信

UART レシーバは、マルチプロセッサ通信向けの 9 ビットデータ モードでも使用できます。9 ビットデータ モードで ADDEN ビットをセットすると、レシーバはデータの第 9 ビットが「0」の時にそのデータを無視します。この機能はマルチプロセッサ環境で使用できます。

17.8.1 マルチプロセッサ通信

一般的なマルチプロセッサ通信プロトコルは、データバイトとアドレス / 制御バイトを区別します。一般的なスキームでは、第 9 データビットを使用してそのデータバイトがアドレスなのかデータ情報なのかを判別します。第 9 ビットがセットされていれば、そのデータをアドレスまたは制御情報として処理します。第 9 ビットがクリアされていれば、そのデータワードを先に受信したアドレス / 制御バイトに関連付けられたデータとして処理します。

このプロトコルは下記のように動作します。

- マスタデバイスが第 9 ビットをセットしたデータワード (スレーブデバイスのアドレスを格納) を送信する
- 通信系統内の全てのスレーブデバイスがアドレスワードを受信してスレーブアドレス値をチェックする
- アドレス先のスレーブデバイスはマスタデバイスが送信する後続データバイトを受信して処理し、他のスレーブデバイスは新たなアドレスワード (第 9 ビットがセットされたワード) を受信するまで後続データバイトを破棄する

17.8.2 ADDEN 制御ビット

UART レシーバはアドレス検出モードを備えます。このモードでは、UART レシーバは第 9 ビットがクリアされているデータワードを無視します。このモードは第 9 ビットがクリアされたデータワードをバッファリングしないため、割り込みオーバーヘッドを低減します。ADDEN ビット (UxSTA<5>) をセットするとこの機能が有効になります。

アドレス検出モードを使用するには、UART を 9 ビットデータ モードに設定する必要があります。レシーバを 8 ビットデータ モードに設定した場合、ADDEN ビットは効果を持ちません。

17.8.3 9 ビット送信のセットアップ

9 ビット送信のセットアップ手順は、8 ビット送信モードの手順と基本的に同じですが、PDSEL<1:0> ビット (UxMODE<2:1>) を「11」に設定するという点で異なります (**17.5.3「UART 送信のセットアップ」** 参照)。

UxTXREG レジスタ (送信開始) にはワード書き込みを実行する必要があります。

17.8.4 アドレス検出モードを使用する 9 ビット受信のセットアップ

9 ビット受信のセットアップ手順は、8 ビット受信モードの手順と基本的に同じですが、PDSEL<1:0> ビット (UxMODE<2:1>) を「11」に設定するという点で異なります (**17.7.4「UART 受信のセットアップ」**参照)。

URXISEL<1:0> および (UxSTA<7:6>) ビットで受信割り込みモードを設定する必要があります。

Note:	アドレス検出モードを有効 (ADDEN = 1) にした場合、1 ワードを受信するたびに割り込みを生成するようにURXISEL<1:0>制御ビットを設定する必要があります。各データワードを受信した直後にソフトウェアで受信データのアドレス一致をチェックする必要があります。

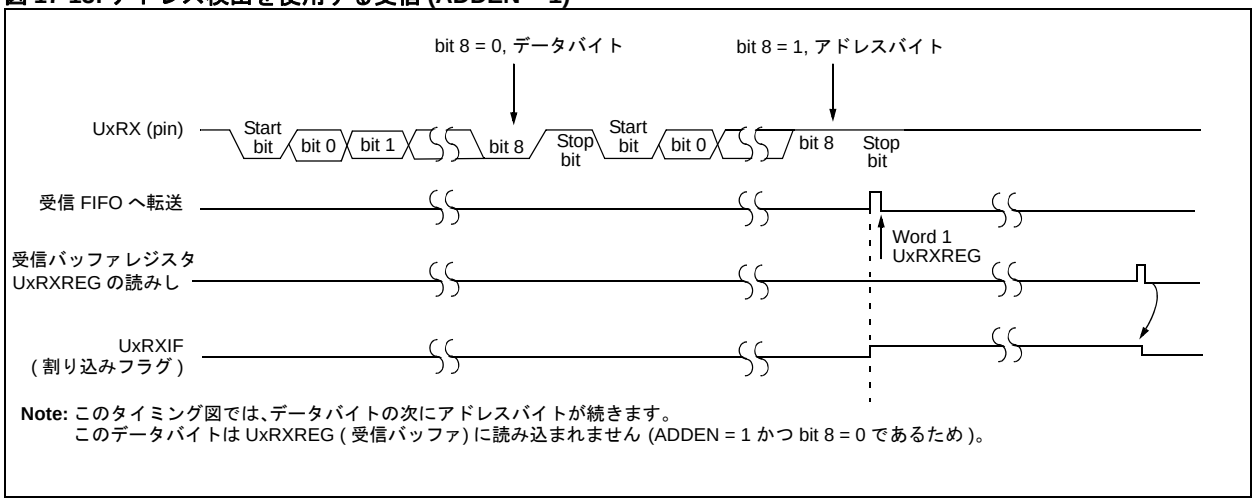
アドレス検出モードを使用する場合の手順は下記の通りです。

1. ADDEN ビット (UxSTA<5>) をセットしてアドレス検出を有効にし、1 ワードを受信するたびに割り込みを生成するように URXISEL 制御ビットが設定されている事を確認する
2. UxRXREG レジスタを読み出して各 8 ビットアドレスをチェックし、当該デバイスがアドレス先として指定されているかどうかを判別する
3. 当該デバイスがアドレス先ではない場合、受信したワードを破棄する
4. 当該デバイスがアドレス先である場合、ADDEN ビットをクリアして後続データバイトを受信バッファへ読み出し可能とし、CPU に割り込む

   長いデータパケットが予期される場合、受信割り込みモードを変更して割り込みと割り込みの間に複数データバイトをバッファリングする事がでます。
5. 最終データバイトを受信した後、アドレスバイトだけを受信するように ADDEN ビットを再度セットする

   加えて、1 ワードを受信するたびに割り込みを生成するように URXISEL 制御ビットが設定されている事を確認する必要があります。

図 17-13: アドレス検出を使用する受信 (ADDEN = 1)



Note: このタイミング図では、データバイトの次にアドレスバイトが続きます。
このデータバイトは UxRXREG (受信バッファ) に読み込まれません (ADDEN = 1 かつ bit 8 = 0 であるため)。

17.9 UART のその他の機能

17.9.1 ループバック モードの UART

LPBACK ビットをセットすると、ループバック モードが有効になり、UxTX 出力は内部で UxRX 入力に接続されます。ループバック モード向けに設定した場合、UxRX ピンは内部 UART 受信ロジックから切断されます。ただし UxTX ピンは通常通り機能します。

ループバック モードを選択するには下記の手順が必要です。

1. UART を必要な動作モードに設定する
2. 送信を有効にする (**17.5「UART トランスミッタ」**参照)
3. LPBACK (UxMODE<6>) ビットを「1」にセットしてループバック モードを有効にする

ループバック モードは表 17-1 に示すように UEN<1:0> ビットの影響を受けます。

表 17-1: ループバック モードのピン機能

UEN<1:0>	ピン機能、LPBACK = 1(1)
00	UxRX 入力を UxTX に接続する ; UxTX ピンは機能する ; UxRX ピンを無視する ; $\overline{\text{UxCTS}}$/$\overline{\text{UxRTS}}$ を使用しない
01	UxRX 入力を UxTX に接続する ; UxTX ピンは機能する ; UxRX ピンを無視する ; $\overline{\text{UxRTS}}$ ピンは機能する ; $\overline{\text{UxCTS}}$ を使用しない
10	UxRX 入力を UxTX に接続する ; UxTX ピンは機能する ; UxRX ピンを無視する ; $\overline{\text{UxRTS}}$ ピンは機能する ; $\overline{\text{UxCTS}}$ 入力を $\overline{\text{UxRTS}}$ に接続する ; $\overline{\text{UxCTS}}$ ピンを無視する
11	UxRX 入力を UxTX に接続する ; UxTX ピンは機能する ; UxRX ピンを無視する ; BCLKx ピンは機能する ; $\overline{\text{UxCTS}}$/$\overline{\text{UxRTS}}$ を使用しない

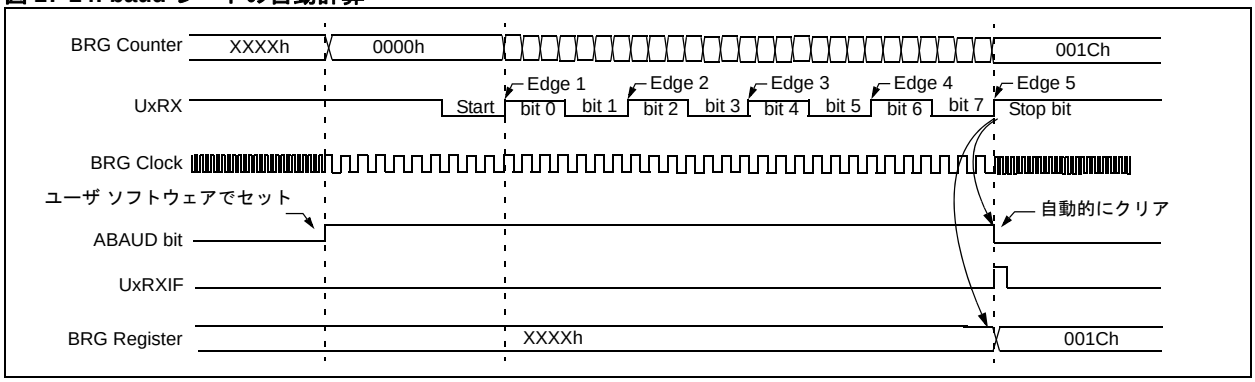
Note 1: LPBACK ビットは、UART モジュールに関連する他のビットを有効にした後に「1」にセットする必要があります。

17.9.2 baud レート自動検出のサポート

システムによる受信キャラクタ baud レートの検出を可能にするには、ABAUD ビットをセットします。baud レート自動検出を有効 (ABAUD = 1) にすると、UART はスタートビットを受信した時に baud レートの自動計測シーケンスを開始します。計算は平均値です。ABAUD ビットをセットすると、BRG カウンタ値がクリアされ、スタートビットを待機します。スタートビットは HIGH から LOW への遷移に続く LOW から HIGH への遷移として定義されます。

baud レート自動検出機能は、スタートビットに続いて 1 つの ASCII キャラクタ 「U」(55h または 0x55) を受信する事によってビットレートを算出します。LOW ビット時間と HIGH ビット時間の両方で計測を行うため、入力信号の非対称性による影響を最小限に抑えられます。UxRXピンの5番目の立ち上がりエッジで、所定のBRG周期中のBRGカウンタ積算値がUxBRGレジスタに転送されます。ABAUD ビットは自動的にクリアされます。ユーザ アプリケーションがシーケンス完了前に ABAUD ビットをクリアした場合、予期せぬモジュール挙動が発生する可能性があります。図 17-14 に baud レート自動検出シーケンスを示します。

図 17-14: baud レートの自動計算



baud レート自動検出シーケンスの実行中、UART ステートマシンはアイドル状態に保持されます。UxRXIF 割り込みは、URXISEL<1:0> の設定に関係なく、5 番目の UxRX 立ち上がりエッジでセットされます。受信 FIFO は更新されません。

17.10 DMA を使用する UART 動作

一部の dsPIC33F/24H デバイスでは、ダイレクト メモリアクセス (DMA) モジュールを使用する事により、CPU に負荷をかけずに CPU と UART 間のデータ転送を行えます。ご使用になるデバイスが DMA を備えるかどうかは、各 dsPIC33F/24H デバイスのデータシートを参照してください。DMA モジュールに関する詳細は、dsPIC33F ファミリ リファレンス マニュアルの **セクション 22.「ダイレクト メモリアクセス (DMA)」** (DS701832) を参照してください。

17.10.1 DMA を使用する UART 受信

DMA チャンネルを UART レシーバに関連付けた場合、UART は 1 キャラクタが UART から RAM へ転送可能となるたびに DMA 要求を発行する必要があります。DMA は、あらかじめ決められた数のデータを UxRXREG レジスタから RAM へ転送した後に CPU 割り込みを生成します。DMA チャンネルは片方向であるため、UART 受信動作用に 1 つの DMA チャンネルが必要です。UART 受信用 DMA チャンネルは表 17-2 のように設定する必要があります。

表 17-2: UART 受信で DMA を使用するための DMA チャンネル レジスタの初期化

周辺モジュールから DMA への関連付け	DMAxREQ レジスタ IRQSEL<6:0> ビット	周辺モジュールから読み出すための DMAxPAD レジスタ値
UART1RX – UART1 レシーバ	0001011	0x0226 (U1RXREG)
UART2RX – UART2 レシーバ	0011110	0x0236 (U2RXREG)

1 キャラクタを受信するたびに割り込みを生成するように UART を設定する必要があります。1 キャラクタを受信するたびに UART レシーバが Rx 割り込みを生成するには、ステータスおよび制御 (UxSTA) レジスタの受信割り込みモード選択ビット (URXISEL<1:0>) を「00」または「01」に設定する必要があります。UART 向けに DMA チャンネルを設定した場合、UART レシーバはデータを受信すると即座に DMA 要求を発行します。DMA 転送を開始するためにユーザ アプリケーションによる特別な手順は不要です。

UART を受信向けに設定している時、DMA チャンネル ワードサイズを 16 ビットに設定する必要があります。この設定により、DMAチャンネルは読み出し可能なデータが存在する時にUARTモジュールから 16 ビットを読み出します。データの下位バイトは、UART モジュールが受信した実際のデータバイトに対応します。データの上位バイトは、そのバイトを受信した時の UART のステータスを格納します。DMA を使用する UART 受信では、UxSTA レジスタを読み出しても FERR と PERR のステータスは返されません。これらのステータスは、DMA チャンネルが UART モジュールから DMA RAM に転送する 16 ビットワードの上位バイトに格納されます。表 17-15 に、DMA が UART モジュールから DMA RAM へ転送する 16 ビットワードの構成を示します。

図 17-15: DMA が RAM へ転送する 16 ビット UART 受信データワードのフォーマット

DMAxPAD レジスタの内容は、前回受信した UxRXREG レジスタの内容のままです。直前の UART 受信でフレーミング エラーまたはパリティエラーが発生した場合、UART エラー割り込みフラグ (UxEIF) がセットされます。UxEIE ビットがセットされると、CPU のコード実行は UART エラー割り込みサービスルーチンへ移動します。この時アプリケーションは、直前に転送されたワードの上位バイトをチェックする事により、割り込みの原因となったエラー条件を判別できます 。

17.10.2 DMA を使用する UART 送信

DMA チャンネルを UART トランスミッタに関連付けた場合、UART は各送信に成功するたびに DMA 要求を発行します。各 DMA 要求の後に、DMA が新たなデータを UxTXREG レジスタに転送し、あらかじめ決められた数の転送を実行した後に CPU 割り込みを生成します。DMA チャンネルは片方向であるため、送信用に 1 つの DMA チャンネルが必要です。各 DMA チャンネルは、表 17-3 に示すように初期化する必要があります。

表 17-3: UART 送信で DMA を使用するための DMA チャンネル レジスタの初期化

周辺モジュールから DMA への関連付け	DMAxREQ レジスタ IRQSEL<6:0> ビット	周辺モジュールから書き込むための DMAxPAD レジスタ値
UART1TX – UART1 トランスミッタ	0001100	0x0224 (U1TXREG)
UART2TX – UART2 トランスミッタ	0011111	0x0234 (U2TXREG)

加えて、1 キャラクタを送信するたびに割り込みを生成するように UART を設定する必要があります。1 キャラクタを送信するたびに UART トランスミッタが Tx 割り込みを生成するには、UxSTA レジスタの送信割り込みモード選択ビット (UTXISEL0 と UTXISEL1) を「0」に設定する必要があります。

UART とトランスミッタを有効にすると、即座に UART トランスミッタが DMA 要求を発行します。従って UART とトランスミッタを有効にする前に、DMA チャンネルとバッファを初期化しておく必要があります。別の方法として、DMA チャンネルを有効にする前に、UART と UART トランスミッタを有効にする事もできます。この場合、UART トランスミッタの DMA 要求は失効するため、ユーザ アプリケーションは DMAxREQ レジスタの FORCE ビットをセットする事によって DMA 要求を発行 (DMA 転送を開始) する必要があります。

17.10.3 UART DMA コンフィグレーションの例

例 17-4 に、2 つの DMA チャンネルを使用する UART 送受信のサンプルコードを示します。UART は 9600 bps で HyperTerminal からキャラクタを受信してバッファリングします。8 個のキャラクタを受信した後に、UART はそれらのキャラクタを HyperTerminal へ返送します (エコー)。

UART 送信向けに DMA チャンネル 0 を下記のように設定します。

- RAM から UART へのデータ転送
- ワンショットモード
- ポスト インクリメントによるレジスタ間接
- 単一バッファを使用
- バッファあたり 8 転送
- ワード転送

UART 受信向けに DMA チャンネル 1 を下記のように設定します。

- UART から RAM への連続データ転送
- ポスト インクリメントによるレジスタ間接
- 2 つのバッファを使用
- バッファあたり 8 転送
- ワード転送

例 17-4: DMA を使用する UART 送受信

UART を RX および TX 用に設定する :

```
#define FCY 40000000
#define BAUDRATE 9600
#define BRGVAL ((FCY/BAUDRATE)/16) - 1

U2MODEbits.STSEL = 0;             // 1 Stop bit
U2MODEbits.PDSEL = 0;             // No Parity, 8 data bits
U2MODEbits.ABAUD = 0;             // Auto-Baud Disabled

U2BRG = BRGVAL;// BAUD Rate Setting for 9600

U2STAbits.UTXISEL0 = 0;           // Interrupt after one TX character is transmitted
U2STAbits.UTXISEL1 = 0;
U2STAbits.URXISEL = 0;            // Interrupt after one RX character is received

U2MODEbits.UARTEN = 1;            // Enable UART
U2STAbits.UTXEN = 1;              // Enable UART TX

-U2EIF = 0;                       // Clear UART2 error interrupt Flag
-U2EIE = 1;                       // Enable UART2 error interrupt
```

DMA チャンネル 0 をワンショット / 単一バッファモードで送信するように設定する :

```
unsigned int BufferA[8] __attribute__((space(dma)));
unsigned int BufferB[8] __attribute__((space(dma)));

DMA0CON = 0x2001;                 // One-Shot, Post-Increment, RAM-to-Peripheral
DMA0CNT = 7;                      // 8 DMA requests
DMA0REQ = 0x001F;                 // Select UART2 Transmitter

DMA0PAD = (volatile unsigned int) &U2TXREG;
DMA0STA = __builtin_dmaoffset(BufferA);

IFS0bits.DMA0IF = 0;              // Clear DMA Interrupt Flag
IEC0bits.DMA0IE = 1;              // Enable DMA Interrupt
```

DMA チャンネル 1 を連続ピンポンモードで受信するように設定する :

```
DMA1CON = 0x0002;                 // Continuous, Ping-Pong, Post-Inc., Periph-RAM
DMA1CNT = 7;                      // 8 DMA requests
DMA1REQ = 0x001E;                 // Select UART2 Receiver

DMA1PAD = (volatile unsigned int) &U2RXREG;
DMA1STA = __builtin_dmaoffset(BufferA);
DMA1STB = __builtin_dmaoffset(BufferB);

IFS0bits.DMA1IF = 0;              // Clear DMA interrupt
IEC0bits.DMA1IE = 1;              // Enable DMA interrupt
DMA1CONbits.CHEN = 1;             // Enable DMA Channel

void __attribute__((__interrupt__,no_auto_psv)) _U2EInterrupt(void)
{
    // An error has occured on the last
    // reception.Check the last received
    // word.

    _U2EIF = 0;

    int lastWord;
```

例 17-4: DMA を使用する UART 送受信 (続き)

```
// Check which DMA Ping pong channel
    // was selected.

    if(DMACS1bits.PPST1 == 0)
    {
        // Get the last word received from ping pong buffer A.
        lastWord = *(unsigned int *)((unsigned int)(BufferA) + DMA1STA);
    }
    else
    {
        // Get the last word received from ping pong buffer B.
        lastWord = *(unsigned int *)((unsigned int)(BufferB) + DMA1STB);
    }

    //Check for Parity Error

    if((lastWord & 0x800) != 0)
    {
        // There was a parity error
        // Do something about it here.

    }

    // Check for framing error
    if ((lastWord & 0x400) != 0)
    {
        // There was a framing error
        // Do some thing about it here.

    }

}
```

DMA 割り込みハンドラをセットアップする :

```
void __attribute__((__interrupt__)) _DMA0Interrupt(void)
{
 IFS0bits.DMA0IF = 0;// Clear the DMA0 Interrupt Flag;
}

void __attribute__((__interrupt__)) _DMA1Interrupt(void)
{
 static unsigned int BufferCount = 0;// Keep record of which buffer contains RX Data

 if(BufferCount = 0)
 {
   DMA0STA = __builtin_dmaoffset(BufferA);// Point DMA 0 to data
                          // to be transmitted
 }
 else
 {
   DMA0STA = __builtin_dmaoffset(BufferB); // Point DMA 0 to data
                          // to be transmitted
 }

 DMA0CONbits.CHEN = 1;// Enable DMA0 Channel
 DMA0REQbits.FORCE = 1;// Manual mode:Kick-start the 1st transfer

 BufferCount ^= 1;
 IFS0bits.DMA1IF = 0;// Clear the DMA1 Interrupt Flag}
```

17.11 CPU スリープ / アイドルモード時の UART 動作

17.11.1 スリープモード時の UART 動作

デバイスがスリープモードに移行すると、UART モジュールへのクロック供給源は全て停止して論理「0」状態を維持します。UART 送信または受信動作中にデバイスがスリープモードに移行した場合、その動作は中止され、UART ピン (BCLKx、UxRTS、UxTX) は既定値状態に駆動されます。

デバイスがスリープモードに移行する前に WAKE ビット (UxMODE<7>) をセットする事により、UART 受信 (UxRX) ピンでスタートビットを検出した時にデバイスをスリープモードからウェイクアップする事ができます。UART 受信割り込み (UxRXIE) を有効にした場合、スリープモード中に UART 受信ピンで立ち下がりエッジを検出すると、UART 受信割り込み (UxRXIF) が発生します。

受信割り込みによってデバイスはスリープ状態からウェイクアップし、下記のように動作します。

- 割り込み優先度が現在の CPU 優先度以下である場合、ウェイクアップしたデバイスはスリープモードを起動した PWRSAV 命令の次の命令からコード実行を再開します。
- 割り込み優先度が現在の CPU 優先度よりも高い場合、ウェイクアップしたデバイスは CPU 例外処理を開始します。この場合キャプチャ ISR の先頭命令からコード実行を再開します。

WAKE ビットは、ウェイクアップ イベント後に UxRX ラインで LOW から HIGH への状態遷移が検出された時に、自動的にクリアされます。

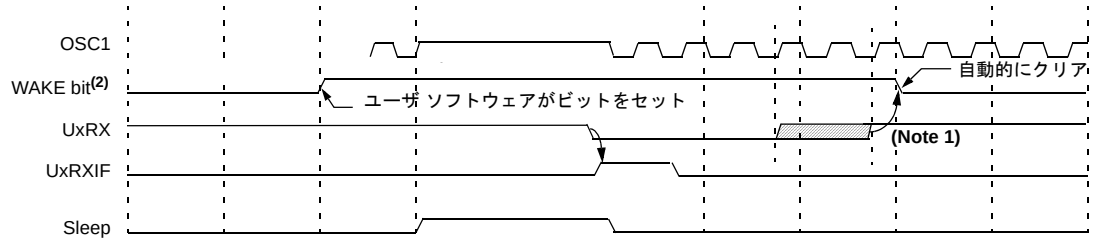
17.11.2 アイドルモード時の UART 動作

デバイスがアイドルモードに移行してもシステムクロック源は動作し続けますが、CPU はコード実行を停止します。アイドルモード時に UART モジュールが停止するか動作を続けるかは、UART モードレジスタ (UxMODE<13>) の UART アイドル時停止ビット (USIDL) によって決まります。

- USIDL = 0 (UxMODE<13>) の場合、モジュールはアイドルモードでも動作を継続し、完全に機能します。
- USIDL = 1 (UxMODE<13>) の場合、モジュールはアイドルモード時に停止し、スリープモード時と同様に機能します (**17.11.1「スリープモード時の UART 動作」**参照)。

1:	選択したオシレータが起動して UART を初期化するのに十分な時間を与えるために、同期ブレーク (またはウェイクアップ信号) キャラクタには十分な長さが必要です。ユーザ アプリケーションは WAKE ビットの値を読み出して UART が正しくウェイクアップした事を確認する必要があります。WAKE ビットがクリアされている場合、UART は次のキャラクタを正しく受信できず、モジュールをバスに再同期させる必要があるかもしれません。
2:	スリープモード時にスタートビットの検出によってデバイスをウェイクアップするには、デバイスをスリープモードへ移行する前に WAKE ビット (UxMODE<7>) をセットする必要があります。
3:	スリープおよびアイドルモードでは UART 受信ピンの立ち下がりエッジで UART 受信割り込みを生成するため、初回の UART 受信割り込みで UART 受信バッファを読み出す場合、ダミーバイトがコピーされます。

図 17-16: スリープモード時の自動ウェイクアップビット (WAKE) のタイミング



Note 1: ウェイクアップ イベントによるオシレータのウォームアップに長時間を要する場合、システムクロックがまだアクティブな時に WAKE ビットの自動クリアが発生する可能性があります。このシーケンスになるような FCY にすべきではありません。

2: UART ステートマシンは、WAKE ビットがアクティブな間アイドルモードに保持されます。

17.12 UxCTS および UxRTS 制御ピンの動作

UxCTS (送信クリア) および UxRTS (送信要求) ピンは、UART モジュールに関連付けられたハードウェア制御ピンです。これら 2 つのピンにより、UART はフロー制御モードと片方向モードで動作できます (**17.12.2「フロー制御モードにおける UxRTS ピンの機能」** と **17.12.3「片方向モードにおける UxRTS ピンの機能」** 参照)。これらのピンは UART とデータ端末装置 (DTE) 間の送受信を制御します。

17.12.1 UxCTS の機能

UART の動作中に UxCTS ピンは入力として機能し、送信を制御する事ができます。このピンは別のデバイス (一般的に PC) により制御されます。UxCTS ピンの設定には UEN<1:0> を使用します。UEN<1:0> = 10 の場合、UxCTS ピンは入力として設定されます。UxCTS = 1 の場合、トランスミッタは送信シフトレジスタにデータを転送しますが、送信を開始しません。これにより、DTE は DTE 側の要件に基づいてコントローラからのデータを制御および受信する事ができます。

UxCTS ピンは、送信データの変化時(16 baud クロックの開始時)に同期してサンプリングされます。送信は UxCTS ピンの状態が LOW と検出された時にのみ開始されます。UxCTS ピンは周期 TCY で内部サンプリングされます。従って UxCTS ピンにおける許容最小パルス幅は 1 TCY です。しかし TCY は使用するクロックに応じて変化するため、この最小パルス幅を 1 つの仕様値として定める事はできません。

ユーザ アプリケーションは、対応するポートピンを読み出す事によって UxCTS ピンのステータスを読み出す事もできます。

17.12.2 フロー制御モードにおける UxRTS ピンの機能

フロー制御モードでは、DTE の UxRTS ピンは dsPIC33F/PIC24H ファミリの UxCTS ピンに接続され、DTE の UxCTS ピンは dsPIC33F/PIC24H ファミリの UxRTS ピンに接続されます (図 17-17 参照)。UxRTS 信号は、デバイスがデータを受信可能な状態であるかどうかを示します。UEN<1:0> = 01 または 10 の場合、UxRTS ピンは常に出力として駆動されます。レシーバがデータ受信可能状態である時、UxRTS ピンは常にアクティブ (LOW) に駆動されます。RTSMD = 0 の場合 (デバイスがフロー制御モードである場合)、UxRTS ピンは受信バッファがフルではない時または OERR ビットがセットされていない時に LOW に駆動されます。RTSMD = 0 の場合、UxRTS ピンはデバイスが受信可能状態ではない時 (受信バッファがフルまたはシフト中の時) に HIGH に駆動されます。

データ端末装置 (DTE) の UxRTS ピンは dsPIC33F/PIC24H ファミリの UxCTS ピンに接続されるため、DTE がデータ受信可能状態の時に UxRTS ピンが UxCTS ピンを LOW に駆動します。データの送信は UxCTS ピンが LOW に遷移した時に開始されます (**17.12.1「UxCTS の機能」** 参照)。

17.12.3 片方向モードにおける UxRTS ピンの機能

片方向モードでは、データ通信装置 (DCE) の UxRTS ピンは dsPIC33F/PIC24H ファミリの UxRTS ピンに接続され、DCE の UxCTS ピンは dsPIC33F/PIC24H ファミリの UxCTS ピンに接続されます (図 17-18 参照)。片方向モードでは、UxRTS 信号は DTE が送信可能状態であるかどうかを示します。DCE がデータを受信可能な状態である時、DCE は UxRTS 信号に対して有効な UxCTS 信号で応答します。DTE は有効な UxCTS 信号を受信した時に送信を開始します。

図 17-19 に示すように、IEEE-485 システムではトランスミッタを有効化するために片方向モードを使用します。UxRTS 信号によって DTE が送信可能状態である事が示されると、UxRTS 信号がドライバを有効化します。

UxRTS ピンは出力として設定され、UEN<1:0> = 01 または 10 の時に駆動されます。RTSMD = 1 の場合、送信可能なデータが存在する (TRMT = 0) 時に UxRTS ピンがアクティブ (LOW) に駆動されます。RTSMD = 1 の場合、トランスミッタがエンプティ (TRMT = 1) の時に UxRTS ピンが非アクティブ (HIGH) に駆動されます。

図 17-17: DTE-DTE 向けの $\overline{\text{UxRTS}}$/$\overline{\text{UxCTS}}$ フロー制御 (RTSMD = 0、フロー制御モード)

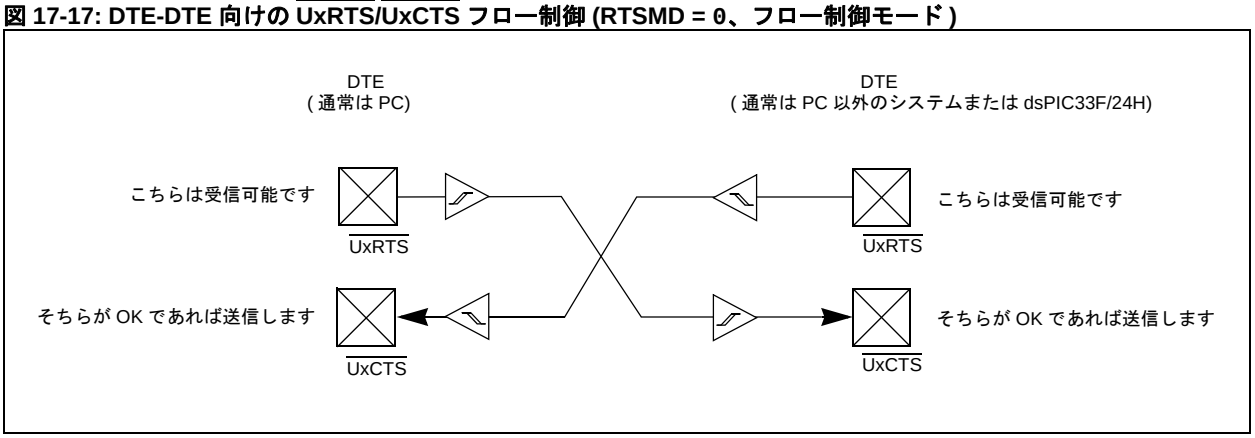


図 17-18: DTE-DCE 向けの $\overline{\text{UxRTS}}$/$\overline{\text{UxCTS}}$ ハンドシェイク (RTSMD = 1、片方向モード)

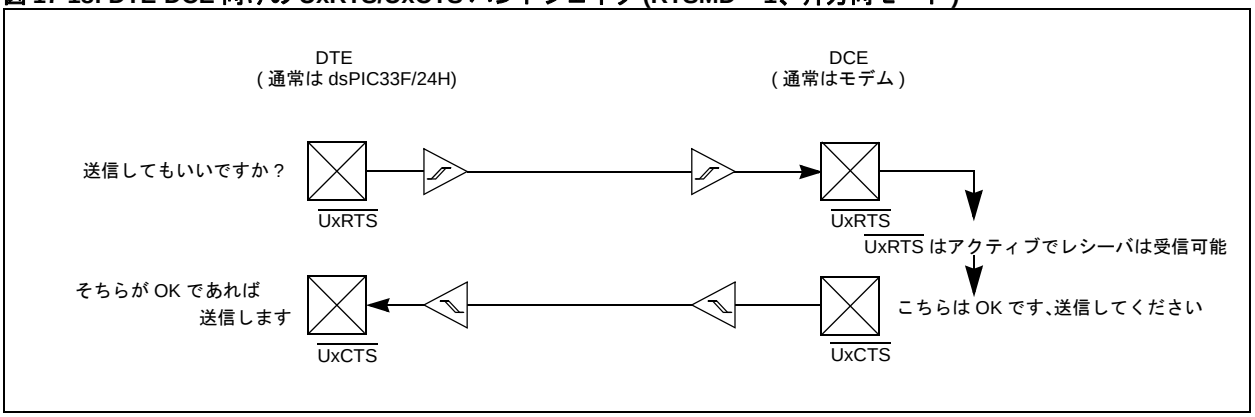
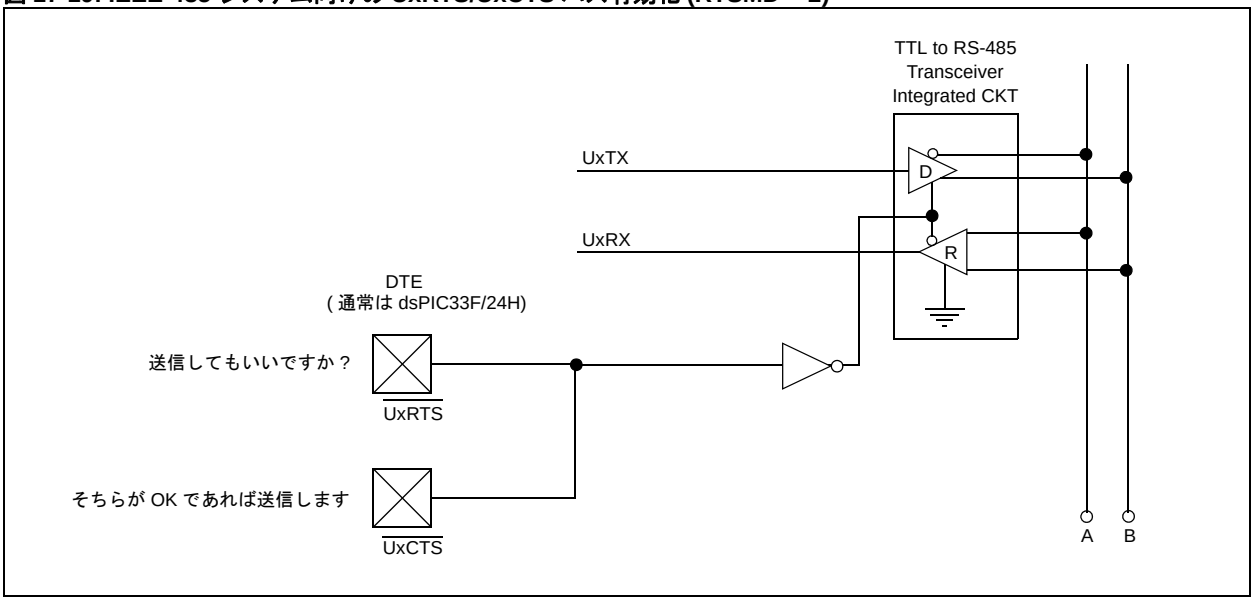


図 17-19: IEEE-485 システム向けの $\overline{\text{UxRTS}}$/$\overline{\text{UxCTS}}$ バス有効化 (RTSMD = 1)



17.13 赤外線サポート

UART モジュールは下記の赤外線 UART サポートを提供します。

- 外部 IrDA エンコーダ / デコーダデバイスをサポートする IrDA クロック出力 (レガシーのモジュール サポート)
- 完全内蔵 IrDA エンコーダ / デコーダ

Note: この機能は低速モード (BRGH = 0、baud レート 1200 以上) でのみ利用できます。

17.13.1 外部 IrDA サポート – IrDA クロック出力

外部 IrDA エンコーダ / デコーダをサポートするために、BCLKx ピンを 16x baud クロックの生成用に設定できます。UEN<1:0> = 11 の場合、UART モジュールが有効な時に BCLKx ピンは 16x baud クロックを出力します。これを使用して IrDA コーデックチップをサポートできます。

17.13.2 内蔵 IrDA エンコーダ / デコーダ

UART モジュールは IrDA エンコーダ / デコーダをモジュールの一部として実装しています。内蔵 IrDA エンコーダ / デコーダ機能の有効化には IREN ビット (UxMODE<12>) を使用します。これ有効 (IREN = 1) にした場合、受信ピン (UxRX) は赤外線レシーバからの入力として機能し、送信ピン (UxTX) は赤外線トランスミッタへの出力として機能します。

17.13.2.1 IrDA エンコーダの機能

このエンコーダは UART からシリアルデータを取得し、下記の方法でデータを置換します。

ビットデータ「1」の送信は 16x baud クロックの全クロックで「0」としてエンコードされます。ビットデータ「0」の送信は 16x baud クロックの先頭から 7 クロックで「0」、次の 3 クロックで「1」、残りの 6 クロックで「0」としてエンコードされます (図 17-20 と図 17-22 参照)。

17.13.2.2 送信の極性

送信極性の選択には UTXINV ビット (UxSTA<14>) を使用します。UTXINV = 0 の場合、UxTX ラインのアイドル状態は「0」です (図 17-20 参照)。UTXINV = 1 の場合、UxTX ラインのアイドル状態は「1」です (図 17-21 参照)。

図 17-20: IrDA® のエンコード スキーム (UTXINV = 0)

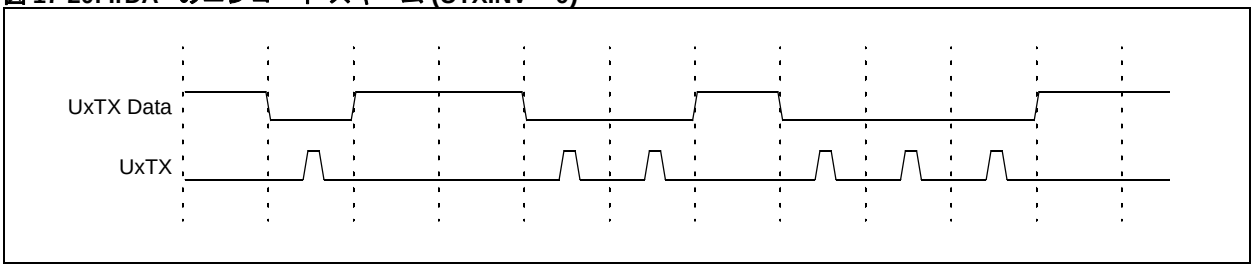


図 17-21: IrDA® エンコード スキーム (UTXINV = 1)

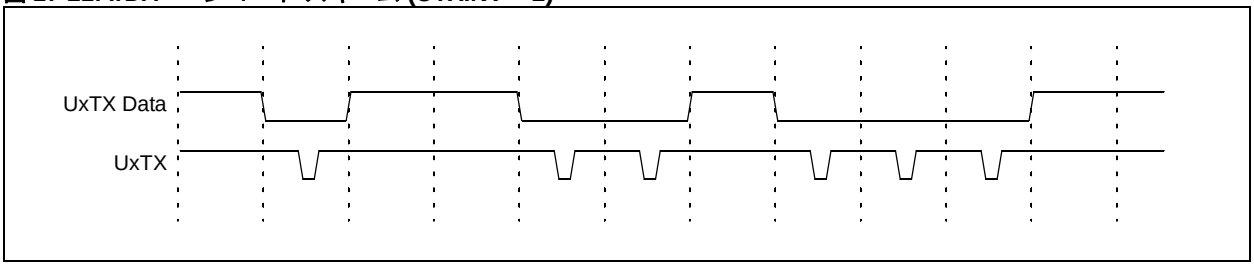
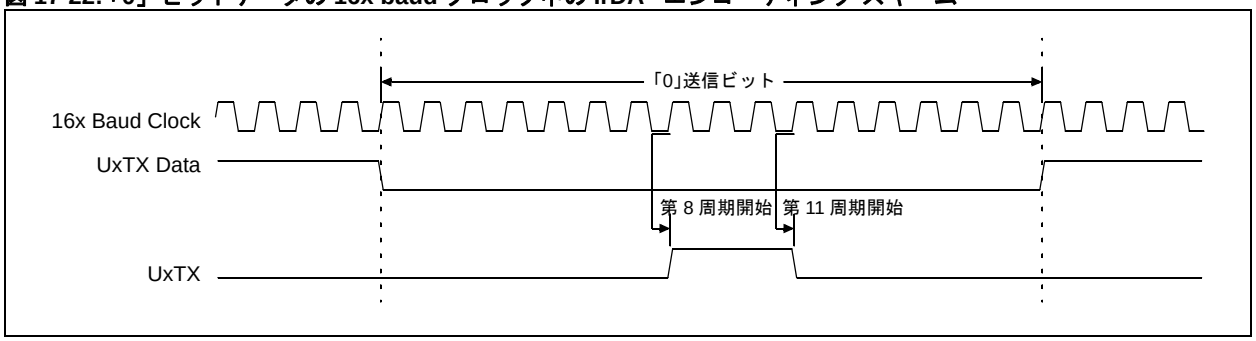


図 17-22:「0」ビットデータの 16x baud クロック中の IrDA® エンコーディング スキーム



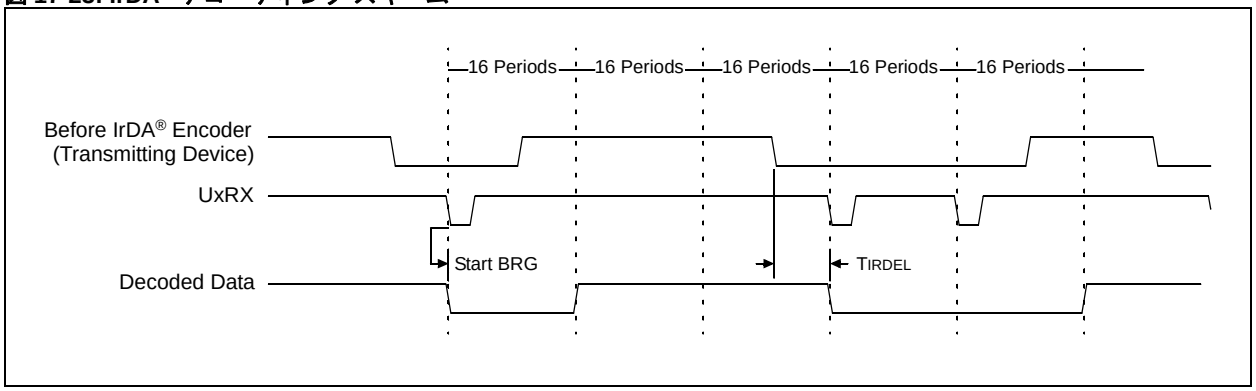
17.13.2.3 IrDA デコーダの機能

デコーダは、UxRX ピンからシリアルデータを取得し、このデータをデコード データストリームに置換します。データストリームは、UxRX 入力の立ち下がりエッジ検出に基づいてデコーディングされます。

UxRX 入力で立ち下がりエッジが検出されると、デコードデータは 16x baud クロックの 16 クロックで LOW に駆動されます。16 クロックが経過した時に次の立ち下がりエッジが検出された場合、デコードデータは続く 16 クロックで LOW を維持します。立ち下がりエッジが検出されなかった場合、デコードデータは HIGH に駆動されます。

デバイスへ送られるデータストリームは、16x baud クロックで 7 または 8 クロックシフトされた (遅れた) 信号を出力します。このような 1 クロック周期の変動は、クロックのエッジ分解能が原因で発生します。詳細は図 17-23 を参照してください。

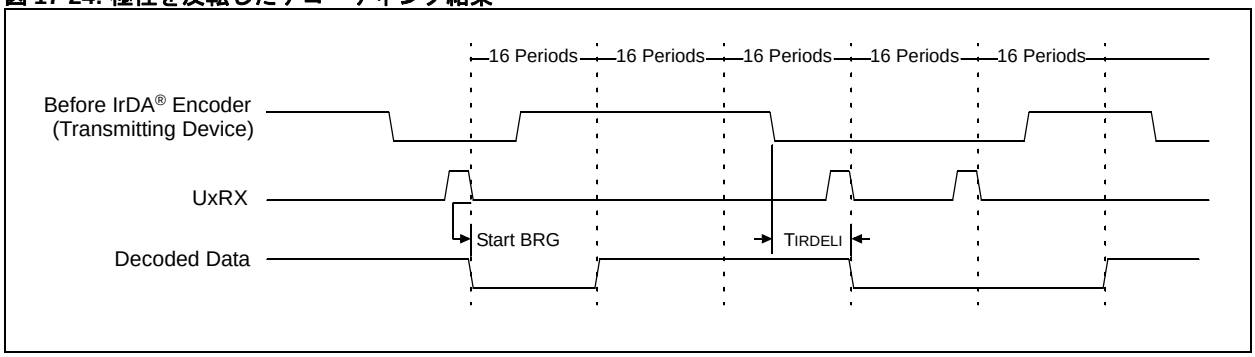
図 17-23: IrDA® デコーティング スキーム



17.13.2.4 IrDA 受信の極性

IrDA 信号入力の極性を反転する事ができます。反転しても信号のデコーディング ロジックは同じですが、デコード データストリームは 16x baud クロックで 10 または 11 クロックシフトされた (遅れた) 信号を出力します。このような 1 クロック周期の変動は、クロックのエッジ分解能が原因で発生します。詳細は図 17-24 を参照してください。

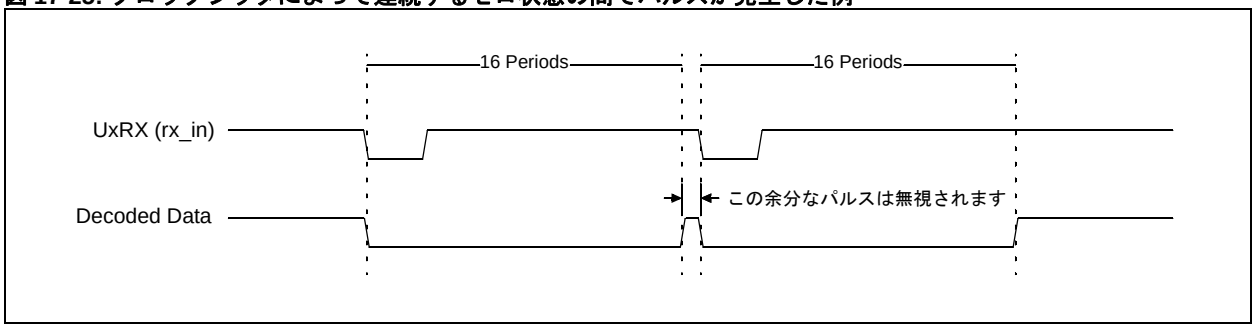
図 17-24: 極性を反転したデコーディング結果



17.13.2.5 クロックのジッタ

ジッタまたはデバイス間の周波数差が原因で、次の立ち下がりエッジを 16x baud クロックの 1 クロックで検出できない場合があります。この場合、1 クロック幅のパルスがデコード データストリームに発生します。UART はビットの中央で多数決検出を行うため、このような場合でもデータに問題を生じません。詳細は図 17-25 を参照してください。

図 17-25: クロックジッタによって連続するゼロ状態の間でパルスが発生した例



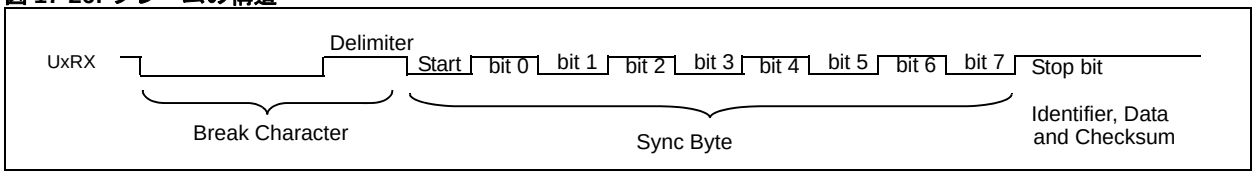
17.14 LIN のサポート

17.14.1 はじめに

LIN プロトコルは、フレームと呼ばれる小さな複数ブロックの形態でデータを送信します。各フレームはデリミタ付きのブレーク キャラクタ、同期バイト、保護された ID、送信データを格納します。詳細は図 17-26 を参照してください。

- ブレーク シーケンス : ブレーク シーケンスはフレームの先頭を示します。マスタノードが生成するブレーク シーケンスは、スタートビットを先頭に 12 ビットの「0」とブレーク デリミタによって構成されます。
- 同期バイト : 同期バイトはデータ値 0x55 を格納した 1 バイト フィールドです。baud レート自動検出機能を有効にした場合、UART モジュールは同期バイトを使用して受信データの baud レートを算出します。
- 保護された ID: 保護された ID は ID と ID パリティを格納します。

図 17-26: フレームの構造



17.14.2 LIN プロトコルを使用するデータの受信

LIN プロトコルを使用する場合、UART モジュールはフレーム形式で受信したデータを受信バッファに転送します。効率的にデータを受信するために、BRG カウンタに受信データの baud レートを書き込む必要があります。

下記の方法により、受信データのビットレートを検出できます。

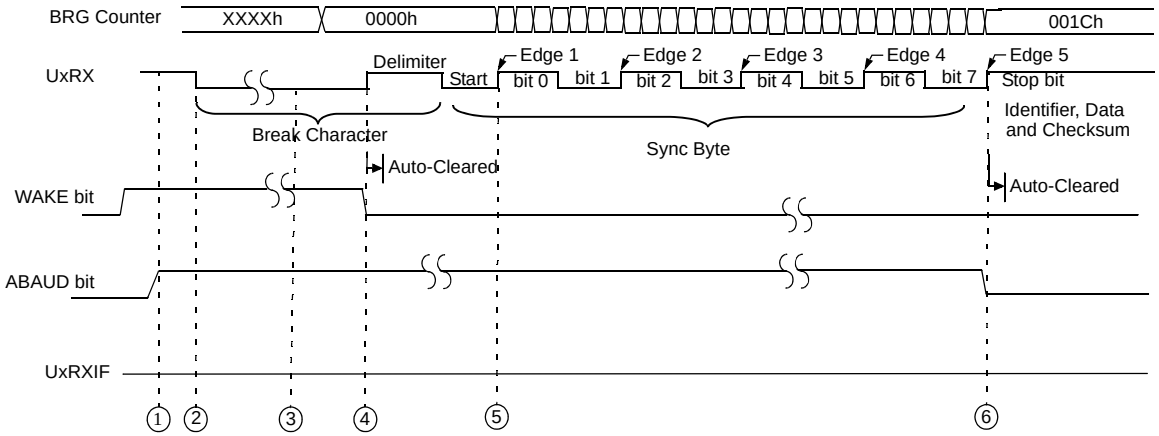
- baud レート自動検出を有効にする
- ABAUD ビットをセットする前に WAKE ビットをセットする

UART モジュールは同期バイトを使用して受信データの baud レートを算出します。ABAUD ビットをセットする前に WAKE ビットをセットした場合、ブレーク キャラクタの次のバイトで baud レート自動検出が発生します。モジュールがブレーク キャラクタのスタートビットとデータに続いて無効なストップビットを受信した場合 (これは FERR をセットする)、モジュールは有効なストップビットを受信するまで待機し、これを受信するまでは次の開始ビットを受信しません。従って有効なストップビットを受信するまで、以降の受信はできません。有効なストップビットを受信すると WAKE ビットがクリアされます。同期キャラクタの 5 番目の立ち上がりエッジの後に、受信データの baud レートが BRG カウンタに書き込まれ、ABAUD ビットが自動的にクリアされます。

WAKE ビットをセットせずに baud レート自動検出機能を有効にした場合、スタートビットではなくデリミタが同期バイトの先頭ビットとみなされます。この結果、baud レートの計算に問題が生じます。これはレシーバが最初に同期バイトを受信する事を期待するためです。しかし LIN プロトコルはブレーク文字で送信を開始し、その後に同期バイトが続きます。従って RX ラインの最初の LOW から HIGH への遷移は 1 ～ 4 ビット幅のデリミタによって発生します。このためスタートビットではなくデリミタが同期バイトの先頭ビットとして機能する事になります。詳細は図 17-27 を参照してください。

Note:	データ受信を開始する前に、ユーザ アプリケーションは UART モジュールの BRG カウンタに受信データの予測ビットレートを書き込む必要があります

図 17-27: LIN プロトコルを使用するデータ受信



1. ユーザ アプリケーションが WAKE ビットをセットしてから ABAUD ビットをセットする
2. ブレーク キャラクタを受信するとデバイスがウェイクアップするため RX 割り込みが発生する
3. FERR ビットがセットされる
4. WAKE ビットがクリアされる
5. baud レート自動検出機能が同期バイトの最初の立ち上がりエッジを検出する
6. 同期バイトの 5 番目の立ち上がりエッジを検出すると、ABAUD ビットをクリアし、BRG カウンタに受信データの baud レートを書き込む

17.15 レジスタマップ

dsPIC33F/PIC24H ファミリの UARTx モジュールに関連するレジスタの要約を表 17-4 に示します。

表 17-4: UARTx 関連のレジスタ

SFR 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
UxMODE	UARTEN	—	USIDL	IREN	RTSMD	—	UEN<1:0>		WAKE	LPBACK	ABAUD	URXINV	BRGH	PDSEL<1:0>		STSEL	0000
UxSTA	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
UxTXREG	—	—	—	—	—	—	—	UTX8	送信レジスタ								xxxx
UxRXREG	—	—	—	—	—	—	—	URX8	受信レジスタ								0000
UxBRG	baud レート ジェネレータ プリスケーラ																0000
IFS0	—	—	—	U1TXIF	U1RXIF	—	—	—	—	—	—	—	—	—	—	—	0000
IFS1	U2TXIF	U2RXIF	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
IFS4	—	—	—	—	—	—	—	—	—	—	—	—	—	U2EIF	U1EIF	—	0000
IEC0	—	—	—	U1TXIE	U1RXIE	—	—	—	—	—	—	—	—	—	—	—	0000
IEC1	U2TXIE	U2RXIE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
IEC4	—	—	—	—	—	—	—	—	—	—	—	—	—	U2EIE	U1EKE	—	0000
IPC2	—	U1RXIP<2:0>			—	—	—	—	—	—	—	—	—	—	—	—	4444
IPC3	—	—	—	—	—	—	—	—	—	—	—	—	—	U1TXIP<2:0>			4444
IPC7	—	U2TXIP<2:0>			—	U2RXIP<2:0>			—	—	—	—	—	—	—	—	4444
IPC16	—	—	—	—	—	U2EIP<2:0>			—	U1EIP<2:0>			—	—	—	—	4444

凡例 : x = リセット時に未知の値、— = 未実装、「0」として読み出し、リセット値は 16 進数で表記

Note: UARTx 関連のレジスタを参考として記載しています。他の UART モジュールに関連するレジスタについては、各デバイスのデータシートを参照してください。

17.16 設計のヒント

質問 1: **UART を使用して送信したデータが正しく受信されません。何が原因でしょうか。**

回答 : 受信エラーの最も一般的な原因として、UART baud レート ジェネレータに対して正しい baud レート値が計算されなかった事が考えられます。UxBRG レジスタに書き込まれた値が正しいかどうかを確認してください。

質問 2: **UART 受信ピンの信号に問題はなさそうですが、フレーミング エラーが発生します。何が原因でしょうか。**

回答 : 下記の制御ビットが正しく設定されている事を確認してください。

- UxBRG: UART baud レートレジスタ
- PDSEL<1:0>: パリティ / データサイズ選択ビット
- STSEL: ストップビットの選択

17.17 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 ノートの一覧を下に記載します。一部のアプリケーション ノートは dsPIC33F/PIC24H デバイス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。UART 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
関連するアプリケーション ノートはありません。	

Note: dsPIC33F/PIC24H ファミリ関連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) でご覧頂けます。

17.18 改訂履歴

リビジョン A (2007 年 2 月)

本書の初版

リビジョン B (2007 年 2 月)

本書全体の小規模な更新

リビジョン C (2008 年 7 月)

このリビジョンでの変更内容は以下の通りです。

- 本書全体の誤植と体裁等、細部の修正
- **17.14「LIN のサポート」**の LIN に関する記述を更新
- **17.2「制御レジスタ」**内の各レジスタの機能説明を追加
- UxSTA: UARTx ステータス / 制御レジスタ (レジスタ 17-2 参照) の bit 14 に関するビット値の説明を変更
- **17.3.1「BCLKx 出力」**の BCLKx 出力ステータス情報を変更
- **17.5「UART トランスミッタ」**の最終段落を削除し、図 17-4 を追加して、UART モジュールを有効にしてから UARTx 送信レジスタにデータを転送するまでのソフトウェア遅延の重要性を説明
- **17.6「データビットの検出」**を追加
- **17.7.2「レシーバエラーの対応」**の最終段落から第 2 文を削除
- 17.8「ブレーク キャラクタの受信」を削除 (この内容は **17.14「LIN のサポート」**に記載済み)
- **17.9.2「baud レート自動検出のサポート」**の第 1 段落の第 4 文と第 2 段落の第 3 文を削除
- 17.9.2.1「ブレーク検出シーケンス」を削除 (この内容は **17.14「LIN のサポート」**に記載済み)
- **17.11「CPU スリープ / アイドルモード時の UART 動作」**を更新
- **17.13「赤外線サポート」**に注釈を追加
- **17.13.2.2「送信の極性」**内の IrDA への参照を削除
- **17.14「LIN のサポート」**を追加

リビジョン D (2011 年 5 月)

このリビジョンでの変更内容は以下の通りです。

- 本書全体の誤植と体裁等、細部の修正
- UxSTA: UARTx ステータス / 制御レジスタ (レジスタ 17-2 参照) 内の bit 14 に関するビット値の説明を変更
- dsPIC33F および PIC24H ファミリ リファレンス マニュアルの関連セクションを統合
- **17.10「DMA を使用する UART 動作」**を更新
- **例 17-4:「DMA を使用する UART 送受信」**を更新
- **17.7.4「UART 受信のセットアップ」**に注釈を追加
- **17.11.2「アイドルモード時の UART 動作」**の下の注釈を更新
- **図 17-11:「UART 受信」**を変更

NOTE:

マイクロチップ社製デバイスのコード保護機能に関して次の点にご注意ください。

- マイクロチップ社製品は、該当するマイクロチップ社データシートに記載の仕様を満たしています。
- マイクロチップ社では、通常の条件ならびに仕様に従って使用した場合、マイクロチップ社製品のセキュリティ レベルは、現在市場に流通している同種製品の中でも最も高度であると考えています。
- しかし、コード保護機能を解除するための不正かつ違法な方法が存在する事もまた事実です。弊社の理解ではこうした手法は、マイクロチップ社データシートにある動作仕様書以外の方法でマイクロチップ社製品を使用する事になります。このような行為は知的所有権の侵害に該当する可能性が非常に高いと言えます。
- マイクロチップ社は、コードの保全性に懸念を抱くお客様と連携し、対応策に取り組んでいきます。
- マイクロチップ社を含む全ての半導体メーカーで、自社のコードのセキュリティを完全に保証できる企業はありません。コード保護機能とは、マイクロチップ社が製品を「解読不能」として保証するものではありません。

コード保護機能は常に進歩しています。マイクロチップ社では、常に製品のコード保護機能の改善に取り組んでいます。マイクロチップ社のコード保護機能の侵害は、デジタル ミレニアム著作権法に違反します。そのような行為によってソフトウェアまたはその他の著作物に不正なアクセスを受けた場合は、デジタル ミレニアム著作権法の定めるところにより損害賠償訴訟を起こす権利があります。

商標

ISBN: 978-1-60932-866-5

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

マイクロチップ社では、Chandler および Tempe (アリゾナ州)、Gresham (オレゴン州) の本部、設計部およびウェハー製造工場そしてカリフォルニア州とイドのデザインセンターが ISO/TS-16949:2002 認証を取得しています。マイクロチップ社の品質システム プロセスおよび手順は、PIC®MCU および dsPIC®DSC、KEELOQ® コード ホッピング デバイス、シリアル EEPROM、マイクロペリフェラル、不揮発性メモリ、アナログ製品に採用されています。さらに、開発システムの設計と製造に関するマイクロチップ社の品質システムは ISO 9001:2000 認証を取得しています。

各国の営業所とサービス

北米

本社
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel:480-792-7200
Fax:480-792-7277
技術サポート :
http://www.microchip.com/support
URL:
www.microchip.com

アトランタ
Duluth, GA
Tel:678-957-9614
Fax:678-957-1455

ボストン
Westborough, MA
Tel:774-760-0087
Fax:774-760-0088

シカゴ
Itasca, IL
Tel:630-285-0071
Fax:630-285-0075

クリーブランド
Independence, OH
Tel:216-447-0464
Fax:216-447-0643

ダラス
Addison, TX
Tel:972-818-7423
Fax:972-818-2924

デトロイト
Farmington Hills, MI
Tel:248-538-2250
Fax:248-538-2260

インディアナポリス
Noblesville, IN
Tel:317-773-8323
Fax:317-773-5453

ロサンゼルス
Mission Viejo, CA
Tel:949-462-9523
Fax:949-462-9608

サンタクララ
Santa Clara, CA
Tel:408-961-6444
Fax:408-961-6445

トロント
Mississauga, Ontario, Canada
Tel:905-673-0699
Fax:905-673-6509

アジア / 太平洋

アジア太平洋支社
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel:852-2401-1200
Fax:852-2401-3431

オーストラリア - シドニー
Tel:61-2-9868-6733
Fax:61-2-9868-6755

中国 - 北京
Tel:86-10-8569-7000
Fax:86-10-8528-2104

中国 - 成都
Tel:86-28-8665-5511
Fax:86-28-8665-7889

中国 - 重慶
Tel:86-23-8980-9588
Fax:86-23-8980-9500

中国 - 武漢
Tel:86-571-2819-3180
Fax:86-571-2819-3189

中国 - 香港 SAR
Tel:852-2401-1200
Fax:852-2401-3431

中国 - 南京
Tel:86-25-8473-2460
Fax:86-25-8473-2470

中国 - 青島
Tel:86-532-8502-7355
Fax:86-532-8502-7205

中国 - 上海
Tel:86-21-5407-5533
Fax:86-21-5407-5066

中国 - 瀋陽
Tel:86-24-2334-2829
Fax:86-24-2334-2393

中国 - 深圳
Tel:86-755-8203-2660
Fax:86-755-8203-1760

中国 - 武漢
Tel:86-27-5980-5300
Fax:86-27-5980-5118

中国 - 西安
Tel:86-29-8833-7252
Fax:86-29-8833-7256

中国 - 厦門
Tel:86-592-2388138
Fax:86-592-2388130

中国 - 珠海
Tel:86-756-3210040
Fax:86-756-3210049

アジア / 太平洋

インド - バンガロール
Tel:91-80-3090-4444
Fax:91-80-3090-4123

インド - ニューデリー
Tel:91-11-4160-8631
Fax:91-11-4160-8632

インド - プネ
Tel:91-20-2566-1512
Fax:91-20-2566-1513

日本 - 横浜
Tel:81-45-471- 6166
Fax:81-45-471-6122

韓国 - 大邱
Tel:82-53-744-4301
Fax:82-53-744-4302

韓国 - ソウル
Tel:82-2-554-7200
Fax:82-2-558-5932 または
82-2-558-5934

マレーシア - クアラルンプール
Tel:60-3-6201-9857
Fax:60-3-6201-9859

マレーシア - ペナン
Tel:60-4-227-8870
Fax:60-4-227-4068

フィリピン - マニラ
Tel:63-2-634-9065
Fax:63-2-634-9069

シンガポール
Tel:65-6334-8870
Fax:65-6334-8850

台湾 - 新竹
Tel:886-3-6578-300
Fax:886-3-6578-370

台湾 - 高雄
Tel:886-7-213-7830
Fax:886-7-330-9305

台湾 - 台北
Tel:886-2-2500-6610
Fax:886-2-2508-0102

タイ - バンコク
Tel:66-2-694-1351
Fax:66-2-694-1350

ヨーロッパ

オーストリア - ヴェルス
Tel:43-7242-2244-39
Fax:43-7242-2244-393

デンマーク - コペンハーゲン
Tel:45-4450-2828
Fax:45-4485-2829

フランス - パリ
Tel:33-1-69-53-63-20
Fax:33-1-69-30-90-79

ドイツ - ミュンヘン
Tel:49-89-627-144-0
Fax:49-89-627-144-44

イタリア - ミラノ
Tel:39-0331-742611
Fax:39-0331-466781

オランダ - ドリューネン
Tel:31-416-690399
Fax:31-416-690340

スペイン - マドリッド
Tel:34-91-708-08-90
Fax:34-91-708-08-91

イギリス - ウォーキンガム
Tel:44-118-921-5869
Fax:44-118-921-5820

05/02/11