AVR1307 : XMEGA USARTの使い方

要点

- USARTの初期設定と使用
- ｺｰﾄﾞ例
  - ﾎﾟｰﾘﾝｸﾞUSART
  - 割り込み制御USART

8ビット AVR® マイクロ コントローラ

応用記述

1. 序説

USART(Universal Synchronus Asynchronous Receiver Transmitter)はｺﾝﾋﾟｭｰﾀ、端末、その他装置間の直列通信での核となる要素です。

本応用記述はXMEGA®で非同期動作でのUSARTの初期設定と使用の方法を記述します。ﾎﾟｰﾘﾝｸﾞと割り込み制御の両方のUSART応用に対してCｺｰﾄﾞのﾄﾞﾗｲﾊﾞと例が含まれています。ﾎﾟｰﾘﾝｸﾞ版は全てのｷｬﾗｸﾀ長(5～9ﾋﾞｯﾄ)を支援し、9ﾋﾞｯﾄ ｷｬﾗｸﾀを支援するために割り込みに基く版が修正され得る方法に対する例としても扱います。割り込みに基くﾄﾞﾗｲﾊﾞは5～8ﾋﾞｯﾄ ｷｬﾗｸﾀだけを支援します。

本応用記述は送信部と受信部のｸﾛｯｸが独立で同期されていない非同期動作を網羅します。

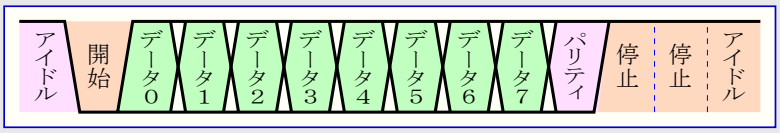
2. 動作の理屈

UART通信はｷｬﾗｸﾀに基き、ｷｬﾗｸﾀは5,6,7,8,9ﾋﾞｯﾄで定義されます。ｷｬﾗｸﾀは開始ﾋﾞｯﾄが先行し、1つまたは2つの停止ﾋﾞｯﾄが後続します。開始ﾋﾞｯﾄは常に信号線のHighｱｲﾄﾞﾙ状態と逆のLowです。

ﾊﾟﾘﾃｨ ﾋﾞｯﾄを含むことは任意選択で、そしてこれはﾃﾞｰﾀ ｷｬﾗｸﾀの後、停止ﾋﾞｯﾄの前に配置されます。ﾊﾟﾘﾃｨ ﾋﾞｯﾄは単独ﾋﾞｯﾄの誤り検出を提供しますが、複数ﾋﾞｯﾄの誤りに対する効果はより劣ります。

図2-1.は1開始ﾋﾞｯﾄ、8ﾃﾞｰﾀ ﾋﾞｯﾄ、1ﾊﾟﾘﾃｨ ﾋﾞｯﾄ、2停止ﾋﾞｯﾄを含む標準的なﾃﾞｰﾀ転送を図解します。

図2-1. 代表的なﾃﾞｰﾀ送信



2.1. XMEGA USART部署

XMEGAのUSARTは既定でUART(非同期)動作に設定されています。送信と受信は個別に許可され、転送は対応するﾌﾗｸﾞのﾎﾟｰﾘﾝｸﾞを使用して、または割り込みﾙｰﾁﾝによって実装することができます。9ﾋﾞｯﾄ ｷｬﾗｸﾀ長使用時を除いて、USARTの使用は少しの落とし穴しかありません。

9ﾋﾞｯﾄ ｷｬﾗｸﾀを使用する場合に特別な注意が祓われなければなりません。ﾃﾞｰﾀ受信に関しては第9ﾋﾞｯﾄ(RXB8)が状態ﾚｼﾞｽﾀ(USARTxn.STATUS)に配置され、USARTから受信したﾃﾞｰﾀを読む時に含められなければなりません。USARTxn.DATAﾚｼﾞｽﾀの読み込み後にRXCIFﾌﾗｸﾞが解除(0)されるので、ﾃﾞｰﾀ ﾚｼﾞｽﾀ(USARTxn.DATA)からの下位ﾊﾞｲﾄの前にRXB8が読まれなければなりません。

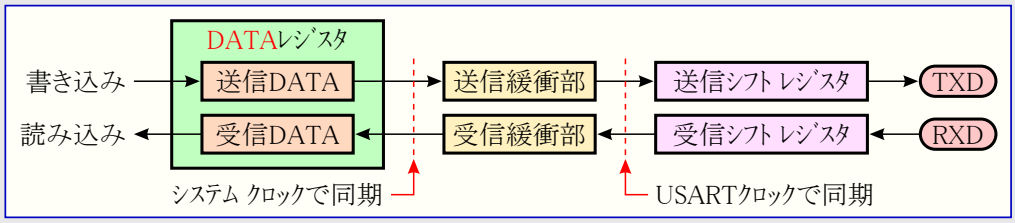
ﾃﾞｰﾀ送信に関しては第9ﾋﾞｯﾄ(TXB8)が制御ﾚｼﾞｽﾀB(USARTxn.CTRLB)に配置されています。USARTxn.DATAへの下位ﾊﾞｲﾄ書き込みが送信を起動し、故にTXB8が先に書かれなければなりません。

2.1.1. 緩衝部

USARTは送信(TX)と受信(RX)の両方向で2重緩衝されています。I/Oﾋﾟﾝに接続されたRXとTXのｼﾌﾄ ﾚｼﾞｽﾀに加えて、RXとTXの両方に対して2つの緩衝ﾚｼﾞｽﾀがあります。1つの緩衝部はDATAﾚｼﾞｽﾀとして参照され、もう一方は緩衝ﾚｼﾞｽﾀとして参照されます。通常動作の間はDATAﾚｼﾞｽﾀだけをｱｸｾｽし、ﾃﾞｰﾀはDATAﾚｼﾞｽﾀとｼﾌﾄ ﾚｼﾞｽﾀ間をﾊｰﾄﾞｳｪｱによって自動的に移動されます。RXとTXのDATAﾚｼﾞｽﾀは同じ物理ｱﾄﾞﾚｽを共用しますが、読み込みは受信DATAﾚｼﾞｽﾀをｱｸｾｽし、書き込みは送信DATAﾚｼﾞｽﾀをｱｸｾｽします。

Rev. 8049A-02/08, 8049AJ4-03/14

図2-2. USART緩衝



完全なキャラクタがシフト レジスタに受信されると、それが緩衝部に複写され、シフト レジスタは第2(または第3)のキャラクタを受信する準備が整います。DATAレジスタの読み込みなしに3キャラクタが受信された場合、第4キャラクタがシフト レジスタ内のキャラクタの損失を引き起こし、第3バイトが失われることに注意してください。この場合、緩衝部溢れフラグ(USARTxn.STATUSのBUFOVF)が設定(1)されます。従ってデータ損失を避けるために応用は充分に速くデータを読まなければならず、そしてこれは追加のソフトウェア緩衝または割り込みに基いたドライバを必要とするかもしれません。

送信緩衝部は受信緩衝部のような3段のFIFO緩衝部で、緩衝部初段がシフト レジスタに複写されて送信が開始されると直ぐに、第3バイト書き込みに対して開放されます。

2.1.2. フラグ

受信完了(RXCIF)、送信完了(TXCIF)、データ レジスタ空(DREIF)に対するフラグはUSART操作で重要です。

RXCIFフラグは受信緩衝部内に未読データがある時に設定(1)され、受信緩衝部が空の時に解除(0)されます。RXCIFフラグはデータ読み込みによって解除(0)され、手動でフラグを解除(0)する必要はありません。TXCIFは送信シフト レジスタ内の枠組み全体がシフト出力され、新しいデータが送信緩衝部内に現在存在しない時に設定(1)されます。DREIFフラグは送信緩衝部(USARTxn.DATA)が追加データに対して空き場所があることを示します。

TXCIFとDREIFは同じ様に思えるかもしれませんが、データ転送の速度向上に利用することができる小さな違いがあります。TXCIFは送信緩衝部が空でUSARTが送信シフト レジスタ内の全データの送信を完了するまで設定(1)されません。これに反してDREIFは緩衝部が更なるデータに対し空き場所がある時、既に設定(1)されます。これは全データが送信される前に新規データを緩衝部内に満たすことができ、そして送信される各キャラクタ間での小さな休止を避けることができることを意味します。

3つのフラグ全てが割り込みを生成するのに使用できます。

2.1.3. ボーレート選択

USARTxn.BAUDCTRLA/Bレジスタの BSEL11～0ビットによって制御されるボーレート クロック発生器に加えて、XMEGAはUSARTxn.BAUDCTRLBの上位ニブルに配置されたBSCALE3～0ビットを持ち、これは数学的なボーレート クロック設定を制御します。

BSCALEを0に設定した場合、ボーレート発生器は追加の尺度調整なしで動作し、ボーレートは**式2-1.**に従ってBSEL11～0によって設定されます。

式2-1. 周波数計算、周波数生成動作

$$f_{BAUD} = \frac{f_{PER}}{16 \times (BSEL+1)}$$

標準動作だけを使用することによって広いボーレート選択が達成できますが、BSCALE3～0ビットがもっと柔軟性を加えます。これらのビットの(0を除く)-7～+7間のどれかへの設定が数学的なクロック尺度調整を許可し、利用可能なボーレートを拡大します。ボーレート尺度係数と倍速動作のより多くの情報に関してはXMEGA手引書をご覧ください。

2.2. DMA転送

XMEGAのDMA機能はUSARTにデータの転送を許します。これはより大きな塊が送受信される時のCPU負荷を減らすのに用いることができます。DMAはデータの塊の転送完了時に割り込みを生成するように設定することができます。

DMAについてのより多くの情報に関してはAVR1304応用記述をご覧ください。

3. USARTドライバ

本応用記述はCで実装された基本ドライバの一括ソース コードを含みます。それはIAR Embedded Workbench®コンパイラで書かれています。

このドライバが高い可読性と周辺機能部署の使用法の一般的な例として書かれていることに注意してください。ドライバ内の多くの関数は速度と容量をもっと効率的にするためにマクロとして実装されています。CPUに集中的な応用では割り込みに基いたドライバが優位性を示します。割り込みで制御されたドライバの使用によって、CPUはデータが送受信されたかを調べる必要がありませんが、これが起きる時に自動的に通知されるでしょう。

ポーリング対割り込み駆動のドライバ選択は応用依存で、度々データ転送に用いられる規約に依存します。

(ポーリングと割り込みの)例内のコードはいくつかの値を送出して、受信値が送出値と等しいことを調べます。これはポートCのI/Oピン2と3間に戻し用の短絡線を使用して検査することができます。

3.1. ファイル

一括ソース コードは次の4つのファイルから成ります。

- usart_driver.c : ポーリング ドライバ ソース ファイル
- usart_driver.h : ポーリング ドライバ ヘッダ ファイル
- usart_example_polled.c : ポーリング ドライバを使用するコード例
- usart_example_interrupt.c : 割り込みドライバを使用するコード例

3.2. 9ビット キャラクタ版

ポーリング ドライバだけが9ビット キャラクタ支援に作られていますが、いくつかの小さな変更で割り込みドライバもまたそれを支援できます(3.3.項をご覧ください)。ポーリング ドライバはUSART_PutCharとUSART_GetCharの特別版を含みます。これらの9ビット版は全てを9ビットで扱うように改造したUSART_NineBits_PutCharとUSART_NineBits_GetCharです。

3.3. 割り込みドライバ

応用に適合する割り込みドライバを保証するのにいくつかの改造が行われるかもしれません。

3.3.1. 緩衝

割り込み制御ドライバで9ビット キャラクタの支援が必要とされるなら、コード例に関してポーリング ドライバをご覧ください。第9ビットの読み書きに加えて、下の例証のように9ビット値を保持するために緩衝部の拡大が重要です。

```
typedef struct Buffer
{
    uint16_t RX[USART_RX_BUFFER_SIZE];
    uint16_t TX[USART_TX_BUFFER_SIZE];
    uint8_t RX_Head, RX_Tail;
    uint8_t TX_Head, TX_Tail;
} Buffer_t;
```

3.3.2. 割り込み処理ルーチン

例はUSART C0を使用して書かれています。他のどれかのUSARTが使用されるべきなら、下のUSART C0に基いてそれらに対する同様の割り込み処理ルーチンを追加することを憶えて置いてください。

```
#pragma vector=USARTC0_RXC_vect
__interrupt void USARTC0_RxcIsr(void)
{
    USART_Rxc(&USART_C0);
}

#pragma vector=USARTC0_DRE_vect
__interrupt void USARTC0_DreIsr(void)
{
    USART_Dre(&USART_C0);
}
```

割り込み処理ルーチンは引数として正しいUSARTへのポインタで共通処理部(USART_RxcとUSART_Dre)を呼び出すべきです。

3.3.3. 環状緩衝

図3-1.は割り込み制御ドライバで使用する環状緩衝の形式を図解します。環状緩衝は環状を想像した配列です。緩衝部への書き込みは書く度毎に進行方向へ1段移動し、最後を通過する時に始めから再び開始します。読み込みは同じ方法で移動し、データが書き込みと同じまたはより速い速度で読まれる限り、緩衝部は待ち行列(キュー)のように働きます。時間に関してデータが読むよりも速く書かれた場合、緩衝部は満杯になるでしょう。

図3-1. N要素の容量での環状緩衝

3.4. Doxygen資料化

全てのソース ファイルはDoxygenを使用する自動資料生成用に準備されています。Doxygenは特別なキーワードを使用してソース コードを分析することによって、ソース コードから資料を作成するツールです。Doxygenについてのより多くの詳細に関してはhttp://www.doxygen.orgを訪ねてください。予めコンパイルされたDoxygen資料は本応用記述に伴うソース コードと共に供給され、ソース コード フォルダのreadme.htmlファイルから利用可能です。

本社

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
TEL 1(408) 441-0311
FAX 1(408) 487-2600

国外営業拠点

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
TEL (852) 2245-6100
FAX (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
TEL (33) 1-30-60-70-00
FAX (33) 1-30-60-71-11

Atmel Japan
104-0033 東京都中央区
新川1-24-8
東熱新川ﾋﾞﾙ 9F
ｱﾄﾒﾙ ｼﾞｬﾊﾟﾝ株式会社
TEL (81) 03-3523-3551
FAX (81) 03-3523-7581

製品窓口

ｳｪﾌﾞｻｲﾄ
www.atmel.com

技術支援
avr@atmel.com

販売窓口
www.atmel.com/contacts

文献請求
www.atmel.com/literature

本応用記述はATMELのAVR1307応用記述(doc8049.pdf Rev.8049A-02/08)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には( )内に英語表記や略称などを残す形で表記しています。

青字の部分はﾘﾝｸとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。