TB3216

万能同期非同期送受信器(USART)での開始に際して

序説

著者: Alexandru Niculae, Microchip Technology Inc.

この資料の目的はmegaAVR® 0系、tinyAVR® 0及び1系、AVR® DAﾃﾞﾊﾞｲｽでUSART周辺機能をどう構成設定するかを段階的に記述することです。これが複雑な周辺機能で様々な動作形態で動くことができるとは言え、この資料は以下の使用事例に対してそれを非同期動作で使います。

- **端末へ‘Hello World’送出**

  PCに文字列を送出する方法を実演し、それを端末で見せます。

- **書式化した文字列送出/’printf’を使う文字列雛形送出**

  USART上で文字列を送るのに’printf’関数を使うことができる能力で最初の使用事例を強化します。

- **制御命令受信**

  度々、USARTは命令行ｲﾝﾀｰﾌｪｰｽを実装するのに使われます。この方法で、ﾏｲｸﾛ ｺﾝﾄﾛｰﾗはUSART経由で制御命令を受け取ることができます。

加えて、この文書は同期動作と単線動作でのUSART構成設定方法の情報を提供します。

注: この文書で記述される各使用事例に対して2つのｺｰﾄﾞ例があります。1つはATmega4809で素の状態で開発され、1つはAVR128DA48で開発されたMPLAB®ｺｰﾄﾞ構成部(MCC)で生成されました。

GitHubでATmega4809ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

GitHubでAVR128DA48ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

目次

1. 関連ﾃﾞﾊﾞｲｽ

本章はこの文書に関連するﾃﾞﾊﾞｲｽを一覧にします。以下の図はﾋﾟﾝ数の変種とﾒﾓﾘ量を展開して各種系統ﾃﾞﾊﾞｲｽを示します。

- これらのﾃﾞﾊﾞｲｽが完全にﾋﾟﾝ互換で同じまたはより多くの機能を提供するため、垂直上方向移植はｺｰﾄﾞ変更なしで可能です。tinyAVR® 1系統ﾃﾞﾊﾞｲｽでの下方向移植はいくつかの周辺機能のより少ない利用可能な実体のため、ｺｰﾄﾞ変更を必要とするかもしれません。
- 左への水平方向移植はﾋﾟﾝ数、従って利用可能な機能を減らします。
- 異なるﾌﾗｯｼｭ ﾒﾓﾘ量を持つﾃﾞﾊﾞｲｽはまた一般的に異なるSRAMとEEPROMの量を持ちます。

図1-1. tinyAVR® 0系統概要

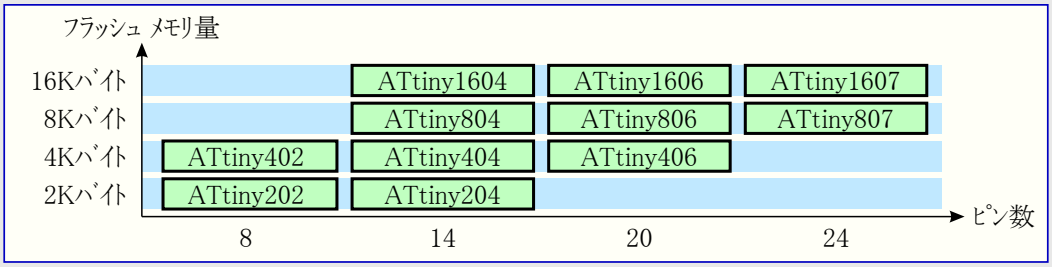


図1-2. tinyAVR® 1系統概要

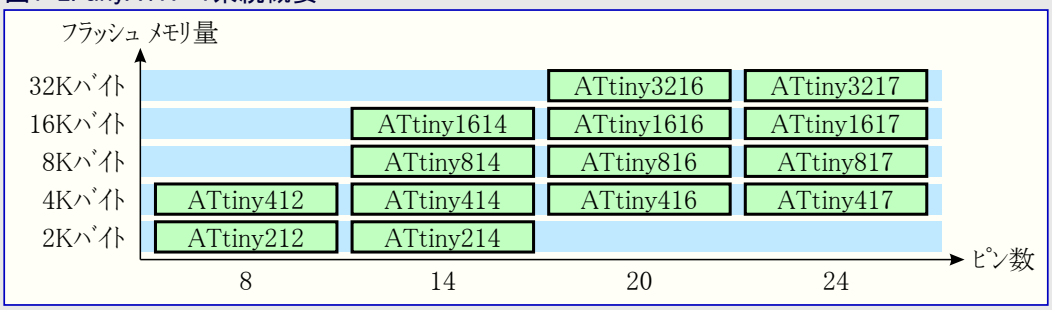


図1-3. megaAVR® 0系統概要

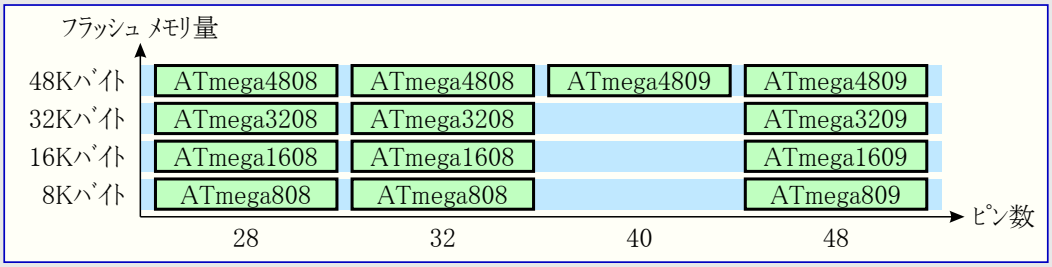
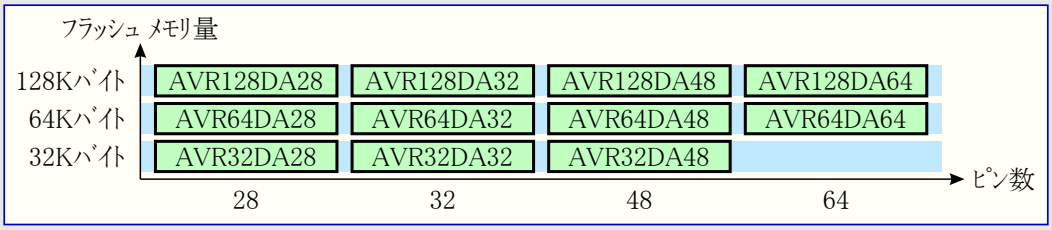


図1-4. AVR® DA系統概要

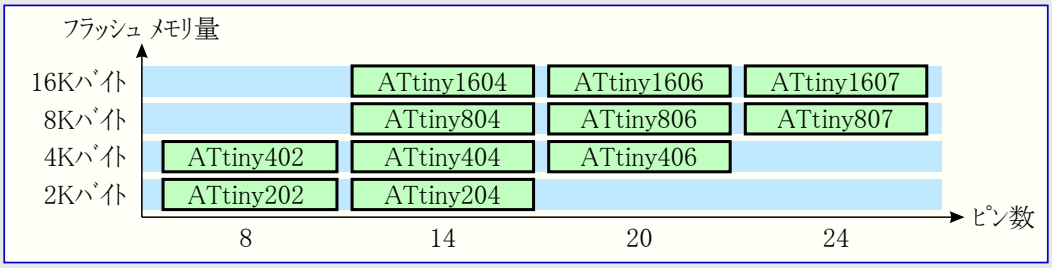


1.1. tinyAVR® 0系統

下図はﾋﾟﾝ数の変種とﾒﾓﾘ量を展開してtinyAVR® 0系統ﾃﾞﾊﾞｲｽを示します。

- これらのﾃﾞﾊﾞｲｽが完全にﾋﾟﾝと機能が互換のため、垂直方向移植はｺｰﾄﾞ変更なしで可能です。
- 左への水平方向移植はﾋﾟﾝ数、従って利用可能な機能を減らします。

図1-5. tinyAVR® 0系統概要



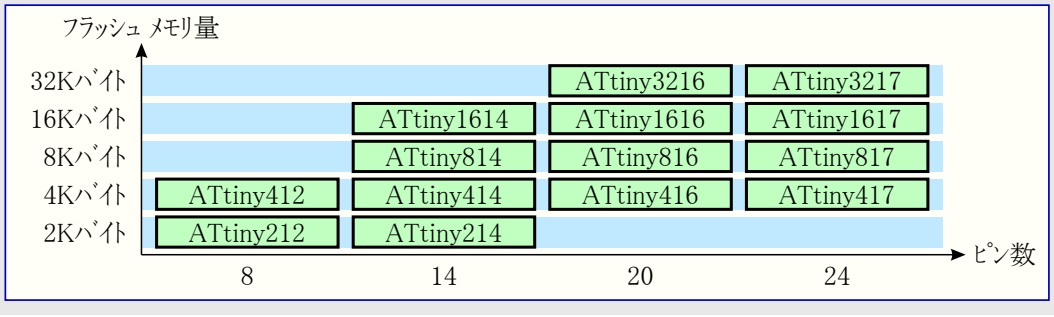
異なるﾌﾗｯｼｭ ﾒﾓﾘ量を持つﾃﾞﾊﾞｲｽは一般的に異なるSRAMとEEPROMの量を持ちます。

1.2. tinyAVR® 1系統

下図はﾋﾟﾝ配置変種とﾒﾓﾘ量を展開してtinyAVR® 1系統ﾃﾞﾊﾞｲｽを示します。

- これらのﾃﾞﾊﾞｲｽがﾋﾟﾝ互換で同じまたはより多くの機能を提供するため、垂直上方向移植はｺｰﾄﾞ変更なしに可能です。下方向移植はより少ない利用可能ないくつかの周辺機能の実体のためにｺｰﾄﾞ変更が必要かもしれません。
- 左への水平方向移植はﾋﾟﾝ数、従って利用可能な機能を減らします。

図1-6. tinyAVR® 1系統概要



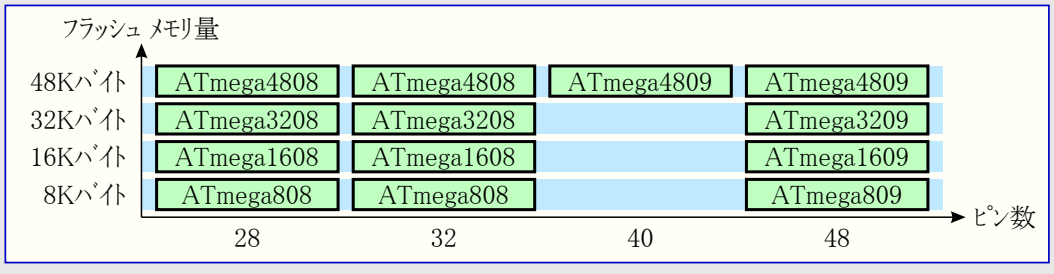
異なるﾌﾗｯｼｭ ﾒﾓﾘ量を持つﾃﾞﾊﾞｲｽは一般的に異なるSRAMとEEPROMの量を持ちます。

1.3. megaAVR® 0系統

下図はﾋﾟﾝ配置変種とﾒﾓﾘ量を展開してmegaAVR® 0系統ﾃﾞﾊﾞｲｽを示します。

- これらのﾃﾞﾊﾞｲｽが完全にﾋﾟﾝと機能が互換のため、垂直方向移植はｺｰﾄﾞ変更なしで可能です。
- 左への水平方向移植はﾋﾟﾝ数、従って利用可能な機能を減らします。

図1-7. megaAVR® 0系統概要



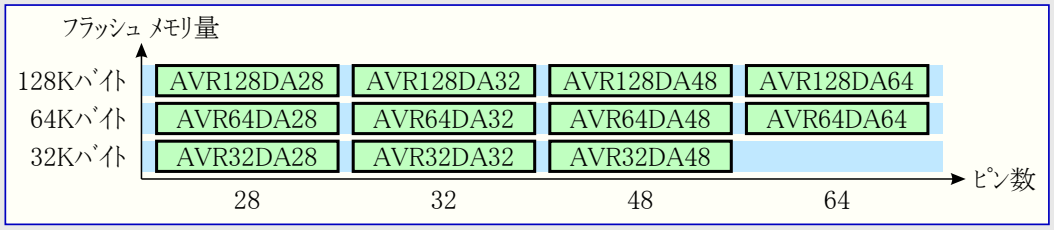
異なるﾌﾗｯｼｭ ﾒﾓﾘ量を持つﾃﾞﾊﾞｲｽは一般的に異なるSRAMとEEPROMの量を持ちます。

1.4. AVR® DA系概要

次図はﾋﾟﾝ配置変種とﾒﾓﾘ量を展開してAVR® DAﾃﾞﾊﾞｲｽを示します。

- これらのﾃﾞﾊﾞｲｽが完全にﾋﾟﾝと機能が互換のため、垂直方向移植はｺｰﾄﾞ変更なしで可能です。
- 左への水平方向移植はﾋﾟﾝ数、従って利用可能な機能を減らします。

図1-8. AVR® DA系統概要

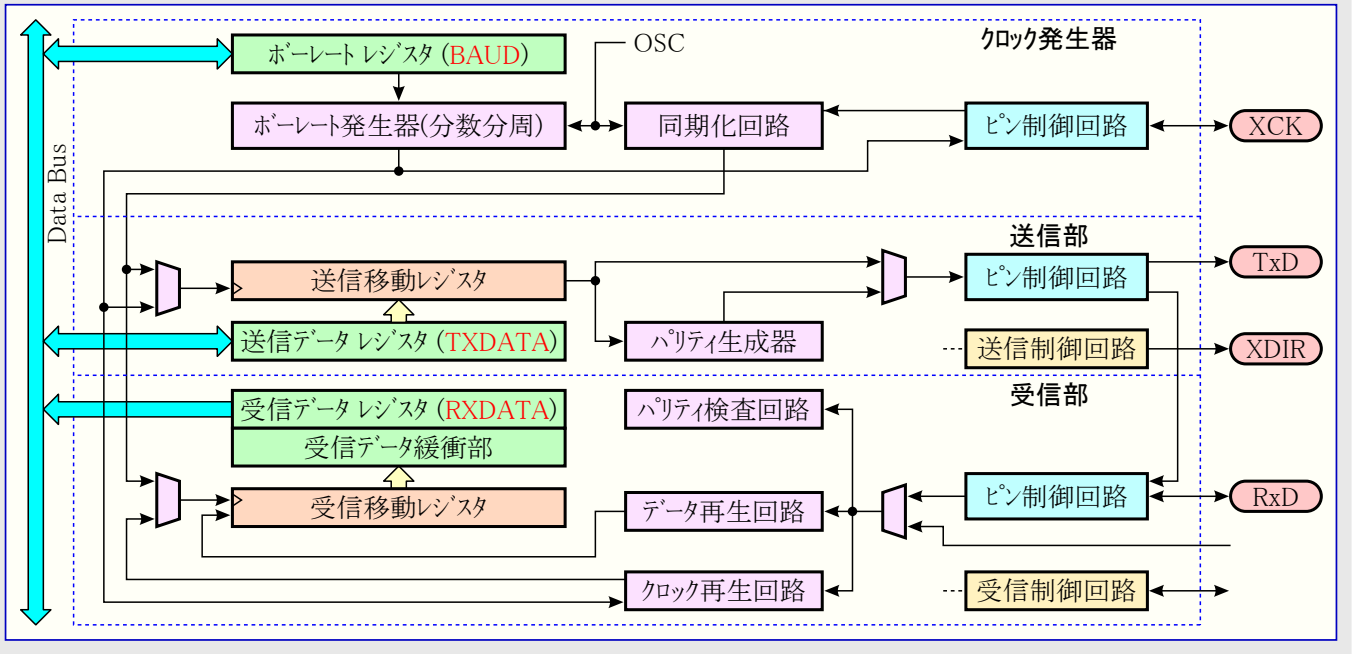


異なるフラッシュ メモリ量を持つデバイスは一般的に異なるSRAM量を持ちます。

2. 概要

USART単位部はRX(受信)、TX(送信)、XCK(クロック)、XDIR(方向)と名付けられた4つのピンを持ちます。単線動作だけは、送信と受信の両方にTXピンが使われます。この動作の欠点は半二重通信だけを提供することです。非同期動作ではRXとTXの両ピンが使われ、従って全二重通信を成し遂げます。XCKピンは同期動作でのクロック信号に使われ、XDIRピンはRS485動作に使われます。

図2-1. USART構成図



最も一般的なUSART構成設定は"9600 8N1"として参照され、ボーレート9600、8データ ビット、パリティなし、1停止ビットを意味します。従って、代表的にUSARTフレームは10ビット(1開始ビット、8データ ビット、1停止ビット)を持ち、1つのASCII文字を表すことができ、これは"8N1"構成設定が秒毎にBAUD_RATE/10個のASCII文字を送ることを意味します。

注: この資料で記述される全ての例はボーレート9600と"8N1"フレーム形式を使います。シリアル端末はこの構成設定に設定されなければなりません。

その上、USARTは複雑な周辺機能で以下のような少数の他の規約を達成するのに使うことができます。

- 主装置SPI
- 従装置LIN
- IR通信
- アドレス指定可能なUSART (または複数プロセッサ通信と呼ばれる)
- RS485

3. 'Hello World' 送出

この使用事例はPCに文字列をどう送るかを実演してそれを端末で可視化します。USARTは非同期動作に構成設定され、TXピンだけが使われます。

USART1実体の既定ピンのPC0(TX)とPC2(RX)は直接デバッガ インターフェースに接続されます。これらは組み込みデバッガ(EDBG)の仮想COMポート(CDC)インターフェースを使うことによってホストPCへデータ バイトを送るのに使うことができ、追加のUARTからUSBへの変換器の必要を回避します。

この使用事例は次のような手順に従います。

- ボーレート設定
- 送信部(TX)許可
- ピン構成設定

ボーレート構成設定方法

ボーレートは秒毎にどの位のビットが送られるかを示します。より高いボーレートがより速い通信です。一般的なボーレートは、1200、2400、4800、9600、19200、38400、57600、115200で、9600が最も一般的に使われるものです。

megaAVR 0系では最大ボーレート速度が、非同期動作で(最大USARTクロック)×1/8、同期動作で(最大USARTクロック)×1/2に制限されます。ボーレートを設定するにはUSARTn.BAUDレジスタに書いてください。

```
USART1.BAUD = (uint16_t)USART1_BAUD_RATE(9600);
```

ボーレートからレジスタの値を計算するUSART1_BAUD_RATEマクロの使用に注意してください。このマクロは下図の式に基づいて定義されなければなりません。この式はUSART構成設定に依存し、故に他の動作で同じにならないかもしれません。

図3-1. ボーレート レジスタ設定を計算するための等式

動作形態	条件	ボーレート(bps)計算式	USARTn.BAUDレジスタ値計算式
非同期動作	$f_{BAUD} \leqq \frac{f_{CLK\_PER}}{S}$	$f_{BAUD} = \frac{64 \times f_{CLK\_PER}}{S \times BAUD}$	$BAUD = \frac{64 \times f_{CLK\_PER}}{S \times f_{BAUD}}$
同期動作	$f_{BAUD} < \frac{f_{CLK\_PER}}{2}$	$f_{BAUD} = \frac{f_{CLK\_PER}}{2 \times BAUD[15\sim6]}$	$BAUD[15\sim6] = \frac{f_{CLK\_PER}}{2 \times f_{BAUD}}$

Sはビット当たりの採取数です。非同期動作では16(標準(NORMAL)動作)または8(倍速(CLK2X)動作)です。同期動作形態についてはS=2です。

これはUSART1_BAUD_RATEマクロがどう定義されるかです。USARTクロックがCPUクロックと一致するため、F_CPUを使います。

```
#define F_CPU 3333333
#define USART1_BAUD_RATE(BAUD_RATE) ((float)(F_CPU * 64 / (16 * (float)BAUD_RATE)) + 0.5)
```

送信部を許可してデータを送る方法

応用の要求に応じて、使用者はUSART単位部の受信部または送信部だけを許可するように選ぶことができます。この使用事例ではマイクロ コントローラがメッセージを送るだけなので、送信部だけ許可されることが必要です。

```
USART1.CTRLB |= USART_TXEN_bm;
```

データを送る前に、使用者はUSARTn.STATUSレジスタを調べることによって以前の送信が完了されているかを調べることが必要です。以下のコード例は送信データ(TXDATA)レジスタが空になるまで待ってその後にUSARTn.TXDATAレジスタへ文字を書きます。

```
void USART1_sendChar(char c)
{
    while (!(USART1.STATUS & USART_DREIF_bm))
    {
        ;
    }
    USART1.TXDATAL = c;
}
```

送出レジスタは9ビット長です。従って、TXDATALと呼ばれる最初の8ビットを保持する下位側部分とTXDATAHと呼ばれる残りの1ビットを保持する上位側部分の2つの部分に分けられました。TXDATAHはUSARTが9ビット データを使うように構成設定される時にだけ使われます。使われる時に、USARTn.CTRLCのCHSIZEがUSARTn.TXDATALが先に書かれるべき'9ビット−下位バイト先行'に設定される場合を除き、この第9ビットはUSARTn.TXDATALへ書く前に書かれなければなりません。

ピン構成設定方法

TXピンは出力として構成設定されなければなりません。既定で、各周辺機能はいくつかの関連したピン位置を持ちます。ピンはデバイス特定データシートの「多重化された信号」項で記述されます。各USARTは2組のピン位置を持ちます。USART1に対する既定と代替のピン位置が次に示されます。

表3-1. 多重化された信号

ピン名(注1,2)	PC0	PC1	PC2	PC3	VDD	GND	PC4	PC5	PC6	PC7
USARTn	1,TxD	1,RxD	1,XCK	1,XDIR			1,TxD (注3)	1,RxD (注3)	1,XCK (注3)	1,XDIR (注3)

注1: ピン名はポートの実体(A,B,C,～)となるxとピン番号のnを持つPxn形式です。信号の表記法はPORTx_PINnです。全てのピンを事象システムとして使うことができます。

注2: 全てのピンは外部割り込みとして使うことができ、各ポートのPx2とPx6のピンは完全な非同期検出を持ちます。

注3: 赤文字は代替ピン位置。代替位置を選ぶことについてはポート多重器(PORTMUX)文章を参照してください。

この使用事例については既定USART1ピン位置が使われ、それはPC0～PC3です。以下のコードはTXピンの方向を出力に設定しま

```
PORTC.DIR |= PIN0_bm;
```

代替ピン位置を使うには以下のようにPORTMUX.USARTROUTEAレジスタを書いてください。

```
PORTMUX.USARTROUTEA |= PORTMUX_USART10_bm;
```

注: この例では代替ピン位置ではなく、既定ピン位置が使われます。

実演コード

このコード例はUSARTを通して継続的に'Hello World!'文字列を送りのに使われます。文字列は文字単位で送られます。'USART0_sendString'関数は'Hello World!'文字列内の各文字に対して'USART1_sendCharacter'関数を呼びます。各文字送出前に、'USART1_sendCharacter'関数は直前の文字送信完了を待ちます。これはステータス(STATUS)レジスタのデータ レジスタ空割り込み要求フラグ(DREIF)が設定(1)されるまでポーリングすることによって行われます。

```
#define F_CPU 3333333
#define USART1_BAUD_RATE(BAUD_RATE) ((float)(F_CPU * 64 / (16 * (float)BAUD_RATE)) + 0.5)

#include <avr/io.h>
#include <util/delay.h>
#include <string.h>

void USART1_init(void);
void USART1_sendChar(char c);
void USART1_sendString(char *str);

void USART1_init(void)
{
    PORTC.DIR &= ~PIN1_bm;
    PORTC.DIR |= PIN0_bm;

    USART1.BAUD = (uint16_t)USART1_BAUD_RATE(9600);

    USART1.CTRLB |= USART_TXEN_bm;
}

void USART1_sendChar(char c)
{
    while (!(USART1.STATUS & USART_DREIF_bm))
    {
        ;
    }
    USART1.TXDATAL = c;
}

void USART1_sendString(char *str)
{
    for(size_t i = 0; i < strlen(str); i++)
    {
        USART1_sendChar(str[i]);
    }
}

int main(void)
```

```
{
    USART1_init();

    while (1)
    {
        USART1_sendString("Hello World!¥r¥n");
        _delay_ms(500);
    }
}
```

注: 遅延関数を正しく動かすために<avr/delay.h>ﾍｯﾀﾞがｲﾝｸﾙｰﾄﾞされる前にCPU周波数を定義してください。

注: 既定構成設定はCPUｸﾛｯｸ周波数とUSARTﾌﾚｰﾑ構造体に対して使われます。CPUと周辺機能の既定ｸﾛｯｸ周波数は3.33MHzです。USART既定ﾌﾚｰﾑ構造体は8ﾃﾞｰﾀ ﾋﾞｯﾄ、ﾊﾟﾘﾃｨなし、1停止ﾋﾞｯﾄ(8N1)で構成されます。

GitHubでATmega4809ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

本章で記述されたのと同じ機能を持つAVR128DA48用のMPLAB®ｺｰﾄﾞ構成部(MCC)生成されたｺｰﾄﾞ例は以下のここで見つけることができます。

GitHubでAVR128DA48ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

4. 書式化した文字列送出/printfを使う文字列雛形送出

これは可変領域を持つ文字列を送るような応用、例えば、応用がそれの状態や計数器値を報告する時用の一般的な事例です。書式化した文字列の使用は非常に柔軟な手段でｺｰﾄﾞ行数を減らします。これは'printf'関数の出力ﾃﾞｰﾀ列の流れを変更することによって達成することができます。

この使用事例は次のこれらの手順に従います。

- USART周辺機能を最初の使用事例と同じに構成設定します。
- 使う定義済みﾃﾞｰﾀ列を作成します。
- 使用者定義ﾃﾞｰﾀ列の流れで標準出力ﾃﾞｰﾀ列の流れを置き換えます。

通常、'printf'使用時、文字は標準出力ﾃﾞｰﾀ列と呼ばれるﾃﾞｰﾀ列の流れへ送られます。PCでは標準出力ﾃﾞｰﾀ列が画面で文字を表示する関数によって処理されます。しかし、ﾃﾞｰﾀ列の流れは別の関数がそれらのﾃﾞｰﾀを処理するように作成することができます。

以下のｺｰﾄﾞはUSART1_printchar関数によって処理される使用者定義ﾃﾞｰﾀ列の流れを作成します。この関数はUSART1_sendChar関数の覆い部ですが、FDEV_SETUP_STREAMがﾊﾟﾗﾒｰﾀとして期待するものと一致するように僅かに異なる用法を持ちます。

```
static void USART1_sendChar(char c)
{
    while (!(USART1.STATUS & USART_DREIF_bm))
    {
        ;
    }
    USART1.TXDATAL = c;
}

static int USART1_printChar(char c, FILE *stream)
{
    USART1_sendChar(c);
    return 0;
}

static FILE USART_stream = FDEV_SETUP_STREAM(USART1_printChar, NULL, _FDEV_SETUP_WRITE);
```

その後にUSART送出関数によって処理される使用者定義ﾃﾞｰﾀ列の流れで標準出力ﾃﾞｰﾀ列の流れを置き換えてください。

```
stdout = &USART_stream;
```

応用は今や直接USARTﾚｼﾞｽﾀに書く代わりに'printf'を使うことができます。

```
uint8_t count = 0;
while (1)
{
    printf("Counter value is: %d¥r¥n", count++);
    _delay_ms(500);
}
```

注: 'printf'関数は文字列雛形内に変数を挿入する場所を記すのに指定子を使います。いくつかの利用可能な指定子が右表で示されます。

表4-1. printf指定子

指定子	説明
%d	符号付き整数を挿入
%s	一続きの文字(文字列)を挿入
%c	文字を挿入
%x	16進数形式で符号なし整数を挿入

他の設定は変更せず、従って上のｺｰﾄﾞ断片では飛ばされています。GitHubで完全なｺｰﾄﾞ例をご覧ください。

GitHubでATmega4809ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

本章で記述されたのと同じ機能を持つAVR128DA48用のMPLAB MCC生成されたｺｰﾄﾞ例は以下のここで見つけることができます。

GitHubでAVR128DA48ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

5. 制御命令受信

USARTの1つの重要な使い方は命令行ｲﾝﾀｰﾌｪｰｽの実装を意味します。この方法で、ﾏｲｸﾛ ｺﾝﾄﾛｰﾗはUSART経由で制御命令を受け取ることができます。命令分離子として行終了子を使うのが便利で、故にこの使用事例についてはUSARTが完全な行を読みます。

この使用事例は次のような手順に従います。

- USART周辺機能を最初の使用事例と同じに構成設定します。
- 受信部を許可します。
- 行の最後まで到着ﾃﾞｰﾀを読んで格納します。
- 受信したﾃﾞｰﾀが有効な命令かを調べて、そうならばそれを実行します。

受信部を許可してﾃﾞｰﾀを受信する方法

USART1について、RX用の既定ﾋﾟﾝ位置はﾎﾟｰﾄCの1番(PC1)ﾋﾟﾝです。以下の行はPC1の方向を入力に設定します。

```
PORTC.DIR &= ~PIN1_bm;
```

送信部と同じで、受信部はUSARTn.CTRLBﾚｼﾞｽﾀに書くことによって許可されます。

```
USART1.CTRLB |= USART_RXEN_bm;
```

ﾃﾞｰﾀを読む前に、使用者は受信完了割り込み要求ﾌﾗｸﾞ(RXCIF)をﾎﾟｰﾘﾝｸﾞすることによって利用可能なﾃﾞｰﾀを待たなければなりません。

```
uint8_t USART1_read()
{
    while (!(USART1.STATUS & USART_RXCIF_bm))
    {
        ;
    }
    return USART1.RXDATAL;
}
```

行を読む方法

以下のｺｰﾄﾞ断片は1行のﾃﾞｰﾀを読んでそれを配列に格納します。有効な行がこの配列長よりも短いと仮定します。

より長い受信行の場合に緩衝部溢れ異常を避けるため、配列が最後に達した時に配列の指標が0にﾘｾｯﾄされます。'¥n'(改行)と'¥r'(復帰)はこれらが行終了子の部分なので無視されます。'¥n'が見つかると、命令に対して文字列終了(NULL)が追加され、'executeCommand'関数が命令文字列の値に基づいた関数を呼びます。

```
char command[MAX_COMMAND_LEN];
uint8_t index = 0;
char c;

/* この遅延はﾃﾞﾊﾞｲｽ ﾘｾｯﾄ後にTX線上の初期雑音を無効にします。 */
    _delay_ms(10);

while (1)
{
    c = USART1_readChar();
    if(c != '¥n' && c != '¥r')
    {
        command[index++] = c;
        if(index > MAX_COMMAND_LEN)
        {
            index = 0;
        }
    }
    if(c == '¥n')
    {
        command[index] = '¥0';
        index = 0;
        executeCommand(command);
    }
}
```

以下のGitHubでのｺｰﾄﾞ例ではUSARTが‘ON’または‘OFF’の命令を受信してﾏｲｸﾛ ｺﾝﾄﾛｰﾗがGPIO出力を制御し、これは、例えば、LEDを切り替えます。

GitHubでATmega4809ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

本章で記述されたのと同じ機能を持つAVR128DA48用のMPLAB MCC生成されたｺｰﾄﾞ例は以下のここで見つけることができます。

GitHubでAVR128DA48ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

6. 他の実装動作形態

前で記述された応用は基本的なUSART機能を実演します。本章は同期動作と単線動作でのUSART構成設定を記述します。

6.1. 同期動作

図6-1. 制御C(CTRLC)ﾚｼﾞｽﾀのUSART通信動作(CMODE)ﾋﾞｯﾄ領域

ﾋﾞｯﾄ	7	6	5	4	3	2	1	0
	CMODE1,0		PMODE1,0		SBMODE		CHSIZE2~0	
ｱｸｾｽ種別	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ﾘｾｯﾄ値	0	0	0	0	0	0	1	1

CTRLCﾚｼﾞｽﾀのCMODEﾋﾞｯﾄ領域は通信動作形態を制御します。

非同期動作の欠点は受信部ﾁｯﾌﾟと送信部ﾁｯﾌﾟが同じﾎﾞｰﾚｰﾄを必要とし、正解なﾀｲﾐﾝｸﾞが必要とされることです。非同期規約はｸﾛｯｸ信号に対して分離された線を使い、故にｸﾛｯｸを生成するﾁｯﾌﾟが通信速度を規定し、これは正確なﾀｲﾐﾝｸﾞの面でより一層の柔軟性を持ち、通信での2つの役割、ｸﾛｯｸを生成する主装置とｸﾛｯｸを受け取る従装置を作ります。

同期USART動作では、追加のｸﾛｯｸ ﾋﾟﾝのXCKが使われます。RXとTXのﾋﾟﾝと同じく、XCDは既定ﾋﾟﾝを持ち、PORTMUXﾚｼﾞｽﾀの変更はXCKも変更します。XCKの方向変更はﾃﾞﾊﾞｲｽが主装置(ｸﾛｯｸを生成)か、または従装置(ｸﾛｯｸを受け取り)かを決めます。

同期動作を達成するには、

・ XCK(PC2)ﾋﾟﾝの方向を出力として構成設定してください。

```
PORTC.DIR &= ~PIN2_bm;
```

・USARTn.CTRLCﾚｼﾞｽﾀのCMODEﾋﾞｯﾄ領域に0x01を書いてください。

図6-2. USART通信動作形態

値	0 0	0 1	1 0	1 1
名称	ASYNCHRONOUS	SYNCHRONOUS	IRCOM	MSPI
説明	非同期USART	同期USART	赤外線通信	主装置SPI

```
USART1.CTRLC = USART_CMODE_SYNCHRONOUS_gc;
```

GitHubでATmega4809ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

本章で記述されたのと同じ機能を持つAVR128DA48用のMPLAB MCC生成されたｺｰﾄﾞ例は以下のここで見つけることができます。

GitHubでAVR128DA48ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

6.2. 単線動作

1つの線だけを使うことはUSART通信に使われるﾋﾟﾝ数を効率的に1つに減らします。RXとTXは内部的に接続され、TXﾋﾟﾝだけが使われ、これはやって来るﾃﾞｰﾀと出て行くﾃﾞｰﾀの両方が同じ線を共用することを意味し、故に送信と受信は同時に起き得ません。これは半二重通信と呼ばれます。

図6-3. 制御A(CTRLA)ﾚｼﾞｽﾀの折り返し動作許可(LBME)ﾋﾞｯﾄ

ﾋﾞｯﾄ	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	DREIE	RXSIE	LBME	ABEIE	RS485[1,0]	
ｱｸｾｽ種別	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ﾘｾｯﾄ値	0	0	0	0	0	0	0	0

RXとTX間の内部送り戻し接続を許可するにはCTRLAﾚｼﾞｽﾀのLBMEﾋﾞｯﾄを使ってください。RXとTX間の内部接続はUSARTn.CTRLAに書くことによって作成することができます。

```
USART1.CTRLA = USART_LBME_bm;
```

これはRXとTXを内部的に接続しますが、TXﾋﾟﾝだけが使われます。TXﾋﾟﾝが送受信の両方に使われるため、ﾋﾟﾝ方向は各送信の前に出力として構成設定され、送信の最後の時に入力へ切り替え戻されることが必要です。

送信中にRXが内部的にTXへ接続されるため、送られつつあるﾃﾞｰﾀを受信し、これは衝突検出機構として使うことができます。別に起きている送信がある場合、受信したﾃﾞｰﾀは送信したﾃﾞｰﾀと一致しません。高度な単線ﾄﾞﾗｲﾊﾞはこの方法を利用することができます。

GitHubでATmega4809ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

本章で記述されたのと同じ機能を持つAVR128DA48用のMPLAB MCC生成されたｺｰﾄﾞ例は以下のここで見つけることができます。

GitHubでAVR128DA48ｺｰﾄﾞ例を見てください。
貯蔵庫を閲覧するにはｸﾘｯｸしてください。

7. 参考文献

1. ATmega4809ｳｪﾌﾞ頁：https://www.microchip.com/wwwproducts/en/ATMEGA4809
2. megaAVR® 0系手引書 (DS40002015)
3. ATmega3209/4809 – 48ﾋﾟﾝ megaAVR® 0系ﾃﾞｰﾀｼｰﾄ (DS40002016)
4. ATmega4809 Xplained Proｳｪﾌﾞ頁：https://www.microchip.com/developmenttools/ProductDetails/atmega4809-xpro
5. AVR128DA48製品頁：www.microchip.com/wwwproducts/en/AVR128DA28
6. AVR128DA48 Curiosity Nano評価ｷｯﾄ ｳｪﾌﾞ頁：www.microchip.com/Developmenttools/ProductDetails/DM164151
7. AVR128DA28/32/48/64 (DS40002183)
8. AVR® DA系での開始に際して (DS00003429)

8. 改訂履歴

文書改訂	日付	注釈
A	2018年12月	初版文書公開
B	2019年6月	3. ‘Hello World’章と4. ‘書式化した文字列送出/printfを使う文字列雛形送出’章のｺｰﾄﾞ例を更新。改訂履歴を追加。その他些細な修正。
C	2021年1月	GitHub貯蔵庫ﾘﾝｸ更新。「AVR® DA系概要」追加、「参考文献」、「改訂履歴」章更新。各使用事例に対してAVR128DA48で動くMCC版を追加。些細な編集上の修正。

Microchipウェブ サイト

Microchipはwww.microchip.com/で当社のウェブ サイト経由でのオンライン支援を提供します。このウェブ サイトはお客様がファイルや情報を容易に利用可能にするのに使われます。利用可能な情報のいくつかは以下を含みます。

- **製品支援** – データシートと障害情報、応用記述と試供プログラム、設計資源、使用者の手引きとハードウェア支援資料、最新ソフトウェア配布と保管されたソフトウェア
- **全般的な技術支援** – 良くある質問(FAQ)、技術支援要求、オンライン検討グループ、Microchip設計協力課程会員一覧
- **Microshipの事業** – 製品選択器と注文の手引き、最新Microchip報道発表、セミナーとイベントの一覧、Microchip営業所の一覧、代理店と代表する工場

製品変更通知サービス

Microchipの製品変更通知サービスはMicrochip製品を最新に保つのに役立ちます。加入者は指定した製品系統や興味のある開発ツールに関連する変更、更新、改訂、障害情報がある場合に必ず電子メール通知を受け取ります。

登録するにはwww.microchip.com/pcnへ行って登録指示に従ってください。

お客様支援

Microchip製品の使用者は以下のいくつかのチャネルを通して支援を受け取ることができます。

- 代理店または販売会社
- 最寄りの営業所
- 組み込み解決技術者(ESE:Embedded Solutions Engineer)
- 技術支援

お客様は支援に関してこれらの代理店、販売会社、またはESEに連絡を取るべきです。最寄りの営業所もお客様の手助けに利用できます。営業所と位置の一覧はこの資料の後ろに含まれます。

技術支援はwww.microchip.com/supportでのウェブ サイトを通して利用できます。

Microchipデバイス コード保護機能

Microchipデバイスでの以下のコード保護機能の詳細に注意してください。

- Microchip製品はそれら特定のMicrochipデータシートに含まれる仕様に合致します。
- Microchipは意図した方法と通常条件下で使われる時に、その製品系統が安全であると考えます。
- Microchipデバイスのコード保護機能を破ろうとする試みに使われる不正でおそらく違法な方法があります。当社はこれらの方法がMicrochipのデータシートに含まれた動作仕様外の方法でMicrochip製品を使うことが必要とされると確信しています。これらのコード保護機能を破ろうとする試みは、おそらく、Microchipの知的財産権に違反することなく達成することはできません。
- Microchipはそれのコードの完全性について心配されている何れのお客様とも共に働きたいと思います。
- Microchipや他のどの半導体製造業者もそれのコードの安全を保証することはできません。コード保護は製品が”破ることができない”ことを当社が保証すると言うことを意味しません。コード保護は常に進化しています。Microchipは当社製品のコード保護機能を継続的に改善することを約束します。Microchipのコード保護機能を破る試みはデジタル ミレニアム著作権法に違反するかもしれません。そのような行為があなたのソフトウェアや他の著作物に不正なアクセスを許す場合、その法律下の救済のために訴権を持つかもしれません。

法的通知

この刊行物含まれる情報はMicrochip製品を使って設計する唯一の目的にのために提供されます。デバイス応用などに関する情報は皆さまの便宜のためにだけ提供され、更新によって取り換えられるかもしれません。皆さまの応用が皆さまの仕様に合致するのを保証するのは皆さまの責任です。

この情報はMicrochipによって「現状そのまま」で提供されます。Microchipは非侵害、商品性、特定目的に対する適合性の何れの黙示的保証やその条件、品質、性能に関する保証を含め、明示的にも黙示的にもその情報に関連して書面または表記された書面または黙示の如何なる表明や保証もしません。

如何なる場合においても、Microchipは情報またはその使用に関連するあらゆる種類の間接的、特別的、懲罰的、偶発的または結果的な損失、損害、費用または経費に対して責任を負わないものとします。法律で認められている最大限の範囲で、情報またはその使用に関連する全ての請求に対するMicrochipの全責任は、もしあれば、情報のためにMicrochipへ直接支払った料金を超えないものとします。生命維持や安全応用でのMicrochipデバイスの使用は完全に購入者の危険性で、購入者はそのような使用に起因する全ての損害、請求、訴訟、費用からMicrochipを擁護し、補償し、免責にすることに同意します。他に言及されない限り、Microchipのどの知的財産権下でも暗黙的または違う方法で許認可は譲渡されません。

商標

品質管理ｼｽﾃﾑ

Microchipの品質管理ｼｽﾃﾑに関する情報についてはwww.microchip.com/qualityを訪ねてください。

日本語© *HERO* 2021.

本技術概説はMicrochipのTB3216技術概説(DS90003216C-2021年1月)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。

世界的な販売とサービス

米国	亜細亜/太平洋	亜細亜/太平洋	欧州
本社 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 技術支援: www.microchip.com/support ｳｪﾌﾞｱﾄﾞﾚｽ: www.microchip.com **ｱﾄﾗﾝﾀ** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **ｵｰｽﾁﾝ** TX Tel: 512-257-3370 **ﾎﾞｽﾄﾝ** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **ｼｶｺﾞ** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **ﾀﾞﾗｽ** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **ﾃﾞﾄﾛｲﾄ** Novi, MI Tel: 248-848-4000 **ﾋｭｰｽﾄﾝ** TX Tel: 281-894-5983 **ｲﾝﾃﾞｱﾅﾎﾟﾘｽ** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **ﾛｻﾝｾﾞﾙｽ** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **ﾛｰﾘｰ** NC Tel: 919-844-7510 **ﾆｭｰﾖｰｸ** NY Tel: 631-435-6000 **ｻﾝﾎｾ** CA Tel: 408-735-9110 Tel: 408-436-4270 **ｶﾅﾀﾞ - ﾄﾛﾝﾄ** Tel: 905-695-1980 Fax: 905-695-2078	**ｵｰｽﾄﾗﾘｱ - ｼﾄﾞﾆｰ** Tel: 61-2-9868-6733 **中国 - 北京** Tel: 86-10-8569-7000 **中国 - 成都** Tel: 86-28-8665-5511 **中国 - 重慶** Tel: 86-23-8980-9588 **中国 - 東莞** Tel: 86-769-8702-9880 **中国 - 広州** Tel: 86-20-8755-8029 **中国 - 杭州** Tel: 86-571-8792-8115 **中国 - 香港特別行政区** Tel: 852-2943-5100 **中国 - 南京** Tel: 86-25-8473-2460 **中国 - 青島** Tel: 86-532-8502-7355 **中国 - 上海** Tel: 86-21-3326-8000 **中国 - 瀋陽** Tel: 86-24-2334-2829 **中国 - 深圳** Tel: 86-755-8864-2200 **中国 - 蘇州** Tel: 86-186-6233-1526 **中国 - 武漢** Tel: 86-27-5980-5300 **中国 - 西安** Tel: 86-29-8833-7252 **中国 - 厦門** Tel: 86-592-2388138 **中国 - 珠海** Tel: 86-756-3210040	**ｲﾝﾄﾞ - ﾊﾝｶﾞﾛｰﾙ** Tel: 91-80-3090-4444 **ｲﾝﾄﾞ - ﾆｭｰﾃﾞﾘｰ** Tel: 91-11-4160-8631 **ｲﾝﾄﾞ - ﾌﾟﾈｰ** Tel: 91-20-4121-0141 **日本 - 大阪** Tel: 81-6-6152-7160 **日本 - 東京** Tel: 81-3-6880-3770 **韓国 - 大邱** Tel: 82-53-744-4301 **韓国 - ソウル** Tel: 82-2-554-7200 **ﾏﾚｰｼｱ - ｸｱﾗﾙﾝﾌﾟｰﾙ** Tel: 60-3-7651-7906 **ﾏﾚｰｼｱ - ﾍﾟﾅﾝ** Tel: 60-4-227-8870 **ﾌｨﾘﾋﾟﾝ - ﾏﾆﾗ** Tel: 63-2-634-9065 **ｼﾝｶﾞﾎﾟｰﾙ** Tel: 65-6334-8870 **台湾 - 新竹** Tel: 886-3-577-8366 **台湾 - 高雄** Tel: 886-7-213-7830 **台湾 - 台北** Tel: 886-2-2508-8600 **ﾀｲ - ﾊﾞﾝｺｸ** Tel: 66-2-694-1351 **ﾍﾞﾄﾅﾑ - ﾎｰﾁﾐﾝ** Tel: 84-28-5448-2100	**ｵｰｽﾄﾘｱ - ヴェルス** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **ﾃﾞﾝﾏｰｸ - ｺﾍﾟﾝﾊｰｹﾞﾝ** Tel: 45-4485-5910 Fax: 45-4485-2829 **ﾌｨﾝﾗﾝﾄﾞ - エスポー** Tel: 358-9-4520-820 **フランス - パリ** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **ﾄﾞｲﾂ - ｶﾞﾙﾋﾝｸﾞ** Tel: 49-8931-9700 **ﾄﾞｲﾂ - ハーン** Tel: 49-2129-3766400 **ﾄﾞｲﾂ - ハイルブロン** Tel: 49-7131-72400 **ﾄﾞｲﾂ - カールスルーエ** Tel: 49-721-625370 **ﾄﾞｲﾂ - ミュンヘン** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **ﾄﾞｲﾂ - ローゼンハイム** Tel: 49-8031-354-560 **イスラエル - ラーナナ** Tel: 972-9-744-7705 **イタリア - ミラノ** Tel: 39-0331-742611 Fax: 39-0331-466781 **イタリア - パドバ** Tel: 39-049-7625286 **オランダ - デルーネン** Tel: 31-416-690399 Fax: 31-416-690340 **ノルウェー - トロンハイム** Tel: 47-72884388 **ポーランド - ワルシャワ** Tel: 48-22-3325737 **ルーマニア - ブカレスト** Tel: 40-21-407-87-50 **スペイン - マドリード** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **スウェーデン - イェーテボリ** Tel: 46-31-704-60-40 **スウェーデン - ストックホルム** Tel: 46-8-5090-4654 **イギリス - ウォーキンガム** Tel: 44-118-921-5800 Fax: 44-118-921-5820