AVR1500 : Xplain練習 – XMEGA 基礎

前提条件

■ 必要な知識
- ﾏｲｸﾛ ｺﾝﾄﾛｰﾗとCﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語の基礎知識

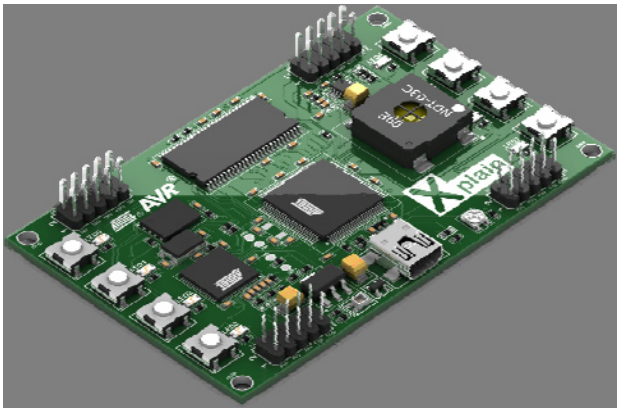
■ ｿﾌﾄｳｪｱ必要条件
- ATMEL® AVR® Studio® 4.18またはそれ以降
- WinAVR/GCC 20100110またはそれ以降

■ ﾊｰﾄﾞｳｪｱ必要条件
- Xplain評価基板
- JTAGICEmk Ⅱ

■ 予想完了時間
- 2時間

1. 序説

本応用記述は4つの課題で様々な構想を示すのにI/Oﾎﾟｰﾄを用い、ATMELのAVR XMEGA®の基礎を網羅します。この練習の目的はより効率的で簡素なｺｰﾄﾞのためにXMEGAﾍｯﾀﾞ ﾌｧｲﾙとXMEGAのいくつかの機能を利用する小さなｺｰﾄﾞの断片で始めさせることです。



2. XMEGA用のCｺｰﾄﾞ書き

開発時間の短縮の圧力と同時に電気製品の高品質保証は高位ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語を必要条件にさせます。それは保守と再利用が容易でより良い可搬性と可読性を与えます。

ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語の選択だけでは高い可読性と再利用性を確実にせず、それは良いｺｰﾄﾞ書きの流儀です。従ってATMEL XMEGAの周辺機能、ﾍｯﾀﾞ ﾌｧｲﾙとﾄﾞﾗｲﾊﾞはこの考えで設計されています。

以降の副項目はXMEGAに対する新しいﾌﾟﾛｸﾞﾗﾐﾝｸﾞ形式のいくつかの概要説明を与えます。より多くの詳細な説明は「AVR1000:XMEGAに対してCｺｰﾄﾞを書く前に」応用記述で与えられます。

2.1. ﾋﾞｯﾄ遮蔽とﾋﾞｯﾄ群遮蔽

ﾚｼﾞｽﾀ ﾋﾞｯﾄは予め定義された遮蔽値、または代わりにﾋﾞｯﾄ位置(これは推奨されません。)を用いて操作することができます。予め定義されたﾋﾞｯﾄ遮蔽値はﾋﾞｯﾄ遮蔽と呼ばれる個別ﾋﾞｯﾄまたはﾋﾞｯﾄ群に関連付けされます。予め定義されたﾋﾞｯﾄ群遮蔽は短縮で群遮蔽と呼ばれます。

ﾋﾞｯﾄ遮蔽は個別ﾋﾞｯﾄの設定と解除の両方の時に使用されます。ﾋﾞｯﾄ群遮蔽は主にﾋﾞｯﾄ群内の複数ﾋﾞｯﾄを解除する時に用いられます。

あなたがATmegaやATtinyのAVRﾏｲｸﾛ ｺﾝﾄﾛｰﾗに慣れているなら、代表的に例1.と同様に事象遅延(EVDLY)ﾋﾞｯﾄを設定(1)するでしょう。

例1. 標準的なATMELのAVRﾏｲｸﾛ ｺﾝﾄﾛｰﾗでのﾋﾞｯﾄ位置使用法

```
TCD0.CTRLD |= (1 << EVDLY);
```

8ﾋﾞｯﾄ AVR®
ﾏｲｸﾛ ｺﾝﾄﾛｰﾗ

応用記述

本書は一般の方々の便宜のため有志により作成されたもので、ATMEL社とは無関係であることを御承知ください。しおりのはじめにでの内容にご注意ください。

Rev. 8308A-06/10, 8308AJ1-03/14

これはこれらのマイクロ コントローラ用のヘッダ ファイルがビット位置を指示し、そしてシフト演算子(<<)でビット遮蔽を作成するためです。XMEGA ヘッダ ファイルとで、これはビット位置とビット遮蔽が既に定義されているため、より読み易くなります。

例2. XMEGAでのビット遮蔽使用法

```
TCD0.CTRLD |= TC0_EVDLY_bm;                    // ビット遮蔽指示子とでの使用
```

例3. XMEGAでのビット位置使用法

```
TCD0.CTRLD |= (1 << TC0_EVDLY_bp);             // ビット位置指示子とでの使用
```

例2.の形式を用いることが推奨されますが、両例ともビット4に1の値を設定、即ちレジスタ値は2進値'0001 0000'と論理和(OR)されます。

多くの形態設定はビットの群によって制御されます。例えば、タイマ/カウンタ制御」レジスタD(CTRLD)(図1.をご覧ください。)で、事象活動(EVACT2～0)と事象元選択(EVSEL3～0)のビットは群化されたビットです。群内のビットの値が形態設定を選びます。群遮蔽はビット群内のビットらと同じ名前と接尾語"_gm"を用い、一方ビット群の位置は"_gp"が接尾されます。

図1. ATMEL XMEGA A手引書で描かれるようなタイマ/カウンタ制御Dレジスタ

ビット	7	6	5	4	3	2	1	0	
+$03	EVACT2~0			EVDLY	EVSEL3~0				CTRLD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初期値	0	0	0	0	0	0	0	0	

ビット群遮蔽は新しい値を書く前にビット群の旧形態設定を最初に解除(=0)することが意図されています。ビット群位置は数値係数、例えばPLL用の倍率係数を設定する時に有用です。

例4. XMEGAでのビット位置使用法

```
TCD0.CTLD &= ~(TC0_EVACT_gm);                  // 群遮蔽での群ビット解除
```

それらがXMEGAヘッダ ファイル内にあるので、遮蔽とビットの関連を調べることにより、上記が何を行うかを見ます。

```
#define TC0_EVACT_gm    0xE0                   /* 事象活動群遮蔽 */
#define TC0_EVACT_gp    5                      /* 事象活動群位置 */

#define TC0_EVACT0_bm   (1<<5)                 /* 事象活動ビット0遮蔽 */
#define TC0_EVACT0_bp   5                      /* 事象活動ビット0位置 */
#define TC0_EVACT1_bm   (1<<6)                 /* 事象活動ビット1遮蔽 */
#define TC0_EVACT1_bp   6                      /* 事象活動ビット1位置 */
#define TC0_EVACT2_bm   (1<<7)                 /* 事象活動ビット2遮蔽 */
#define TC0_EVACT2_bp   7                      /* 事象活動ビット2位置 */
```

TC0_EVACT_gmが1110 0000の2進値を持つことを理解すると、それはビット群を解除するのに非常に有用です。

2.2. ビット群形成遮蔽

望む形成にビット群を設定する時に何のビット模様が使用される必要があるかを調査するためにデータシートを調べることが度々必要とされます。これはコードを読むまたはデバッグする時にも適用されます。可読性を増してビット群での不正なビット設定の可能性を最小とするため、多数の群形成遮蔽値が利用可能にされています。群形成の名前は接尾語"_gc"を持ちます。

図2. 群形成名合成

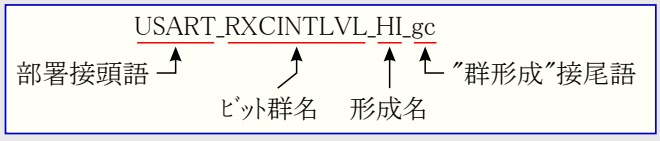


図2.からUSART部署で受信完了割り込みレベル(RXCINTLVL)ビットに対して使用される群形態設定を見ることができます。この指定群形態設定は高(HI)割り込みレベルを選択します。

ビット群を新しい形態設定に変更するには、旧形態設定が先に消去されるのを保証するため、代表的にビット群形成がビット群遮蔽値と連結して使用されます。

例5.

```
USARTC0.CTRLA = (USARTC0.CTRLA & ~USART_RXCINTLVL_gm) | USART_RXCINTLVL_MED_gc;
```

上の例は共に使用される群遮蔽と形成遮蔽を示します。例の最初の部分(USARTC0.CTRLA & ~USART_RXCINTLVL_gm)は例4.で示されるものと同様の方法でUSARTC0レジスタ内のRCXINTLVLビットを解除(=0)します。最後の部分(USART_RXCINTLVL_MED_gc)は中位割り込みレベルを得るように新しい値を設定します。

このコードはレジスタ内の他のビットに影響を及ぼさずに、USARTC0受信完了割り込みレベルを中に再形態設定するのに用いられます。多数の応用記述と共にやって来るドライバの多くでこれと同様のコードを見るでしょう。

2.3. 部署レジスタと部署アドレス

ATMEL XMEGAのI/O割り当ては与えられた周辺機能部署に関する全てのレジスタが1つの連続するメモリ部に配置されるように構成されます。これは全ての周辺機能部署をC構造体に構成することを可能にし、ここで構造体のアドレスはその部署の基準アドレスを定義します。部署に属する全レジスタがその部署構造体の要素です。

以下のコードは設定可能な多段割り込み制御器(PMIC)に関する各種レジスタがXMEGAヘッダ ファイルでどう定義されるかを示します。

例. XMEGAヘッダ ファイル(iox128a1.h)での定義

```
typedef struct PMIC_struct {
    register8_t STATUS;                    /* 状態レジスタ */
    register8_t INTPRI;                    /* 割り込み優先権 */
    register8_t CTRL;                      /* 制御レジスタ */
} PMIC_t
```

このコードはPMIC部署で利用可能なレジスタを定義します。構造体は指定メモリ アドレスでPMICを定義するのに使用されます。

例. 周辺機能部署定義

```
#define PMIC    (*(PMIC_t *) 0x00A0)
```

上の例は部署事例定義が部署事例基準アドレスと符合して、メモリに於いて間接参照ポインタを絶対アドレスへどう使用するかを示します。部署ポインタはXMEGAヘッダ ファイルで予め定義され、従ってソース ファイルにそれらの定義を追加する必要はありません。

上の例が理解できなくても、心配しないでください。

知る必要があるのはこれらの定義をどう使用するかです。上の定義はXMEGAヘッダ ファイルの一部で、以下の例で示されるように、"."(ドット)構文規則で部署内のどのレジスタでもアクセスすることができます。

使用例

```
Unsigned char temp;
temp = PMIC.STATUS;                    // tempに状態レジスタ読み込み
PMIC.CTRL |= PMIC_PMRRPE_bm;           // 制御レジスタのPMRRPEビットを設定(1)
```

上で示された部署レジスタと部署アドレスを使用する主な利点はATMELのXMEGA系統の各種デバイス間と実際の周辺機能と無関係に(例えばUASRT1とUSART4の両方で動く)、ドライバの作成ができることです。

3. 概要

ここはこの練習に於ける課題の短い概要です。

この練習は4つの課題で様々な構想を示すのにI/Oポートを用いて、XMEGAの基礎を網羅します。この練習の目的はより効率的で簡素なコードのためにXMEGAヘッダ ファイルとXMEGAのいくつかの機能を利用する小さなコードの断片で始めさせることです。

課題1. 基本LED制御

この課題は可搬可能なコードを作成するためにXMEGAヘッダ ファイルから#defineと部署名を使用する方法とI/Oポートの操作方法を示します。

課題2. 一般的なドライバ

この課題は一般的なドライバ コードを作るための周辺機能部署へのポインタ使用法と、スイッチ読み込みとLED出力の方法を示します。

課題3. 出力とプルアップ/ダウンの形態設定

この課題はレジスタ内のビット領域を効率的に変更するためにそれの群遮蔽と群形態設定値とでXMEGAヘッダ ファイルを使用する方法を示します。

課題4. 複数形態設定

この課題は同時に1つよりも多くのピンを形態設定するための複数形態設定レジスタの使用方法を示します。

ご幸運を!

4. 課題1: 基本LED制御

あなたの開発基板上で点滅するLEDを持つことほど面白いものはありません!。この課題はまさにそれで、他に何もしません。独自化が容易なｺｰﾄﾞを作るためにATMELのXMEGAﾍｯﾀﾞ ﾌｧｲﾙを利用する方法を示します。それはXMEGAのI/Oﾎﾟｰﾄ機能のいくつかも示します。

この課題の目的はあなたが以下の方法を知ることです。

- 1つの#defineだけでLEDを使用するためにﾎﾟｰﾄを変更してください。
- I/Oﾋﾟﾝの方向を形態設定して出力値を設定してください。
- 1つの単一書き込みｱｸｾｽだけで既存のﾎﾟｰﾄ出力値を変更してください。

すること

1. ATMELのAVR Studioを開始してBasicLED_Contorol.apsﾌﾟﾛｼﾞｪｸﾄ ﾌｧｲﾙを開いてください。そしてtask1.cを探してください(図3.をご覧ください)。
2. 図4.は別のLEDﾎﾟｰﾄを望む場合にLEDPORT定義を変更する容易な方法を示します。

図3. ﾌﾟﾛｼﾞｪｸﾄを開く

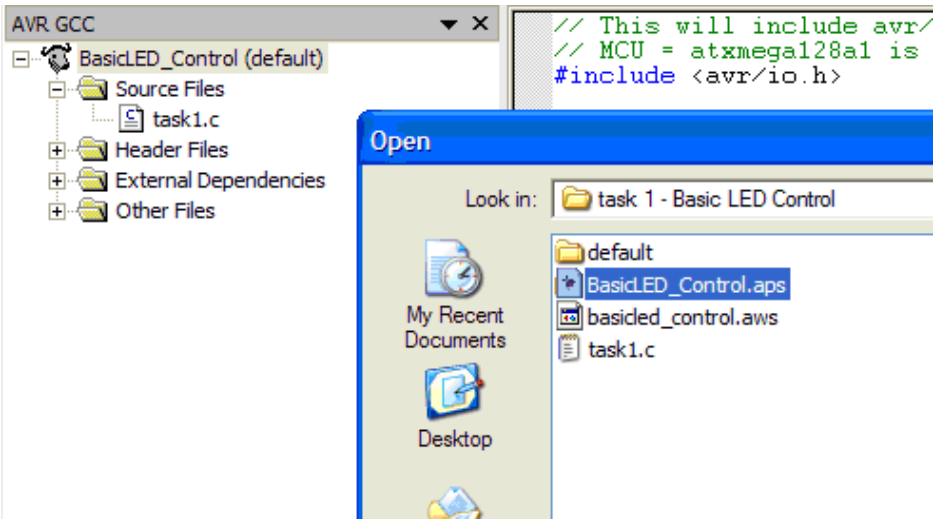
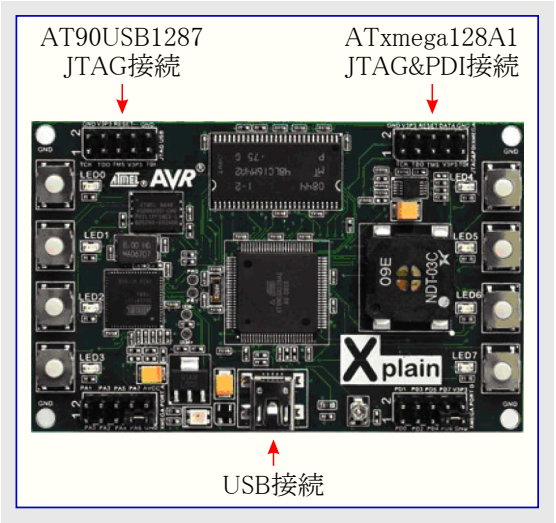


図4. LEDﾎﾟｰﾄ変更

```
// file, it is enough to define t
// #defines for all registers for
// IO port to use for LED output
#define LEDPORT PORTE

volatile unsigned int counter;
```

3. JTAGICEmk IIをXplain評価基板の右の角のXMEGA JTAGﾍｯﾀﾞに接続してください。USBｹｰﾌﾞﾙでJTAGICEmk IIをｺﾝﾋﾟｭｰﾀに接続してそれをONに切り替えてください。そしてXplain基板を対応するUSBｹｰﾌﾞﾙでｺﾝﾋﾟｭｰﾀに接続してください。



4. ﾌﾟﾛｼﾞｪｸﾄを構築(F7を押)してください(図5.をご覧ください)。
5. 開始釦を押下することによってﾃﾞﾊﾞｯｸﾞを始めてください(図6.をご覧ください)。

図5. ﾌﾟﾛｼﾞｪｸﾄ構築(F7)

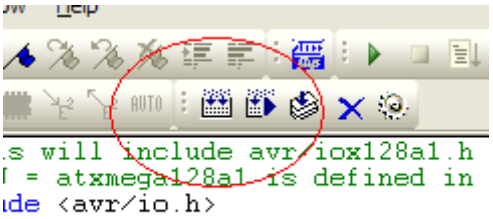
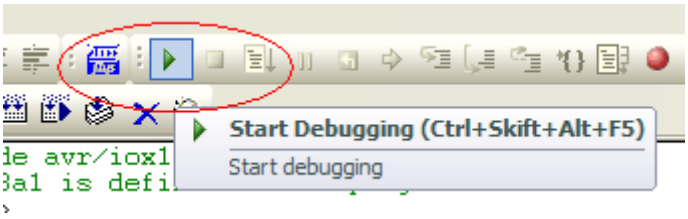


図6. ﾃﾞﾊﾞｯｸﾞ開始




```
.s will enable interrupts
```

6. LEDﾎﾟｰﾄのためにI/Oｳｨﾝﾄﾞｳ(Alt+5)を開いてください(図7.をご覧ください)。I/Oﾋﾟﾝを扱う、かなり多くのﾚｼﾞｽﾀがあります。ﾚｼﾞｽﾀはｺｰﾄﾞをより簡潔にし、開発者に多くの任意選択を提供します。

図7. LEDﾎﾟｰﾄでのﾚｼﾞｽﾀ

Name	Address	Value	Bits
DIR	0x660	0xFF	■■■■■■■■
DIRCLR	0x662	0xFF	■■■■■■■■
DIRSET	0x661	0xFF	■■■■■■■■
DIRTGL	0x663	0xFF	■■■■■■■■
IN	0x668	0xFF	■■■■■■■■
INT0MASK	0x66A	0x00	□□□□□□□□
INT1MASK	0x66B	0x00	□□□□□□□□
INTCTRL	0x669	0x00	■■■■□□□□
INTFLAGS	0x66C	0x00	■■■■■■□□
OUT	0x664	0xFF	■■■■■■■■
OUTCLR	0x666	0xFF	■■■■■■■■
OUTSET	0x665	0xFF	■■■■■■■■
OUTTGL	0x667	0xFF	■■■■■■■■
PIN0CTRL	0x670	0x00	□□□□□□□□
PIN1CTRL	0x671	0x00	□□□□□□□□
PIN2CTRL	0x672	0x00	□□□□□□□□
PIN3CTRL	0x673	0x00	□□□□□□□□
PIN4CTRL	0x674	0x00	□□□□□□□□
PIN5CTRL	0x675	0x00	□□□□□□□□
PIN6CTRL	0x676	0x00	□□□□□□□□
PIN7CTRL	0x677	0x00	□□□□□□□□

7. ｺｰﾄﾞ全体を1行実行(F11を押)して目的対象基板とI/OｳｨﾝﾄﾞｳでLEDを観測してください(図8.をご覧ください)。
8. 1行実行(F11)してI/Oｳｨﾝﾄﾞｳを見ながら、各種ﾎﾟｰﾄ ﾚｼﾞｽﾀを理解してみてください。
9. LEDを点滅するｺｰﾄﾞの最終部を見るためにｺｰﾄﾞを走行(F5を押)してください。

図8. 1行実行(F10)開始

```
// IO port to use for LED output in this e
#define LEDPORT PORTD


void main( void )
{
    LEDPORT.DIR = 0xff; // Set all pins of
    LEDPORT.OUT = 0xf0; // Set upper four
    LEDPORT.OUTSET = 0x0f; // Set lower fo
    LEDPORT.OUTCLR = 0xf0; // Clr upper fo
    LEDPORT.OUTTGL = 0xff; // Toggle all p

    // Now, just for fun, we toggle all LE
```

5. 課題2: 一般的なﾄﾞﾗｲﾊﾞ

正しいﾚｼﾞｽﾀを選ぶのに"switch/case"や"if else"宣言の必要なしにどのI/Oﾎﾟｰﾄ(またはどのA/D変換器、またはどのD/A変換器、またはどのﾀｲﾏ/ｶｳﾝﾀ)をもｱｸｾｽすることができる1つの関数をこれまでに望みましたか?。この課題はI/Oﾎﾟｰﾄ部署へのﾎﾟｲﾝﾀを取るｺｰﾄﾞを作るのにATMELのXMEGAのI/O割り当ての配置を利用し、そして正しいﾚｼﾞｽﾀをｱｸｾｽするのに一般的なｺｰﾄﾞを使用する方法を示します。この方法はI/Oﾎﾟｰﾄ、A/D変換器、D/A変換器、ﾀｲﾏ/ｶｳﾝﾀなどに対する一般的なﾄﾞﾗｲﾊﾞの作成に使用することができます。

この課題の目的はあなたが以下の方法を知ることです。

- ﾎﾟｲﾝﾀ変数を作成してそれを何れかのI/Oﾎﾟｰﾄ部署へ位置付けしてください。
- 関数のﾊﾟﾗﾒｰﾀとして部署ﾎﾟｲﾝﾀを使用してください。
- 部署ﾎﾟｲﾝﾀを通して部署ﾚｼﾞｽﾀをｱｸｾｽしてください。


すること

1. 課題2用のﾌﾟﾛｼﾞｪｸﾄ ﾌｫﾙﾀﾞに於いて、Generic_Drivers.apsﾌﾟﾛｼﾞｪｸﾄ ﾌｧｲﾙを開始し、そしてtask2.cを見てください。
2. LEDとｽｲｯﾁに他のﾎﾟｰﾄの使用を望む場合にledPortとswitchPortのﾎﾟｲﾝﾀ割り当てが容易に変更できることを観察してください。
3. ﾌﾟﾛｼﾞｪｸﾄを構築してください(F7)。
4. ﾃﾞﾊﾞｯｸを開始してください。
5. ｺｰﾄﾞ内を1行実行(F11を押)してLEDを観測してください。GetSwitches関数の内側の時に各種ｽｲｯﾁを押してみてください。
6. 実時間でｽｲｯﾁ状態をLEDに複写するｺｰﾄﾞを見るためにｺｰﾄﾞを走行(F5を押)してください。

図9. 関数の1行実行

```
{
    // Prepare pointers to the peripheral port r
    PORT_t * ledPort = &PORTE;
    PORT_t * switchPort = &PORTF;

    // Enable pullups for switches.
    PORTCFG.MPCMASK = 0xff;
    switchPort->PIN0CTRL = PORT_OPC_PULLUP_gc;

    // Now copy switch state to LEDs again and a
    while(1)
    {
        value = GetSwitches( switchPort );
        SetLEDs( ledPort, value );
    }
```

6. 課題3: 出力とﾌﾟﾙｱｯﾌﾟ/ﾀﾞｳﾝの形態設定

ATMELのXMEGAでは全ての部署の全てのﾋﾞｯﾄ領域に対する全ての形態設定任意選択が1つのﾍｯﾀﾞ ﾌｧｲﾙ(例えばiox128a1.h)で名付けられた定数として利用可能です。この課題はﾍｯﾀﾞ ﾌｧｲﾙから群遮蔽と群形成値を使用する方法と、ﾃﾞｰﾀｼｰﾄでそれらを探す方法、そしてあなたのｺｰﾄﾞでそれらを効果的且つ簡潔にそれらを使用する方法を示します。また、この課題はXMEGAのI/Oﾎﾟｰﾄの機能のいくつかの洞察を与えます。

この課題の目的はあなたが以下の方法を知ることです。

- ﾚｼﾞｽﾀ内のﾋﾞｯﾄ領域を変更するために効率的且つ簡潔なｺｰﾄﾞを書いてください。
- ﾃﾞｰﾀｼｰﾄとﾍｯﾀﾞ ﾌｧｲﾙから群遮蔽と群形成の値を探せ、そして使用してください。
- XMEGAのI/Oﾎﾟｰﾄの出力とﾌﾟﾙｱｯﾌﾟ/ﾀﾞｳﾝの各種設定について体験してください。

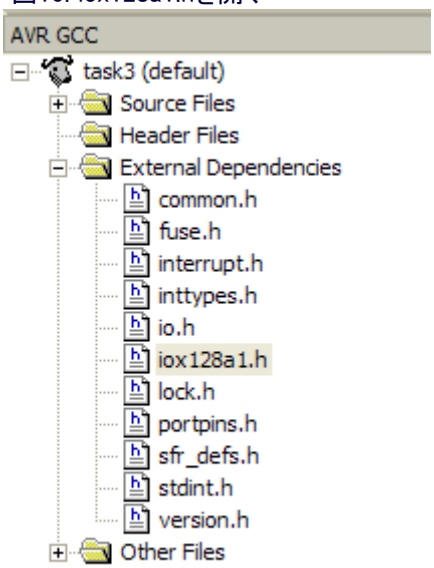

すること

1. ATMELのAVR Studioで、Basic_LED_ControlﾌｫﾙﾀﾞからOutPutAndPull.apsﾌﾟﾛｼﾞｪｸﾄ ﾌｧｲﾙを開いてください。
2. ﾌﾟﾛｼﾞｪｸﾄを構築、F7を押して、ﾃﾞﾊﾞｯｸﾞ作業を開始してください。
3. XMEGA A手引書を開いてPINnCTRLﾚｼﾞｽﾀ形態設定を探し出してください。ﾚｼﾞｽﾀ説明項で出力とﾌﾟﾙｱｯﾌﾟ/ﾀﾞｳﾝ形態設定(OPC)値を考察してください。
4. 構築ﾂﾘｰで外部依存性(External Dependencies)を展開してiox128a1.hを開いてください(図10.をご覧ください)。PORT_OPC_WIREDAND_gcのOPC形態設定が見つかりましたか?。(助言:"WIRED"で検索してください。)
5. task.cに於いて、次のこの行に位置付けて1行実行を行ってください。

   ```
   SWITCHPORT.PINOCTRL = (SWITCHPORT.PINOCTRL & ~PORT_OPC_gm) | PORT_OPC_WIREDANDPULL_gc;
   ```

   2.2.節を参照して上のﾋﾞｯﾄ形成遮蔽を理解してみてください。(ﾜｲﾔｰﾄﾞAND形態設定は次の課題で詳述されます。)
6. I/Oｳｨﾝﾄﾞｳを考察すると同時に更に1行実行を行ってください。手引書の形態設定ﾚｼﾞｽﾀ説明と比較してOPC形態設定を検証してください。

図10. iox128a1.hを開く



7. 課題4: 複数形態設定

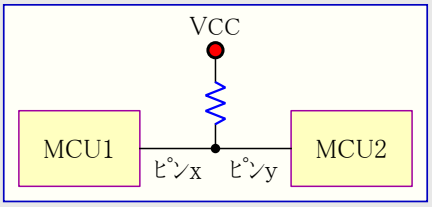
直前の課題で各入出力ﾋﾟﾝに対して1つの形態設定ﾚｼﾞｽﾀがあることを学びました。同時に多数のﾋﾟﾝの形態設定を望む場合はどうでしょうか?。複数形態設定で、ﾎﾟｰﾄ内の多数のﾋﾟﾝを同時に形態設定することができます。

各ﾋﾟﾝに対して形態設定ﾚｼﾞｽﾀを持つことは、単一ﾎﾟｰﾄを形態設定するために必要とする操作数が増加することを意味します。書き込み操作数は全てのﾋﾟﾝに対して共通の全体的な複数ﾋﾟﾝ形態設定許可(MPCMASK)ﾚｼﾞｽﾀの導入によって減らされます。MPCMASKはﾋﾟﾝ形態設定ﾚｼﾞｽﾀに対してﾋﾞｯﾄ遮蔽を設定するのに使用することができます。MPCMASKでﾋﾞｯﾄnを設定すると、PINnCTRLはﾋﾟﾝ形成遮蔽が追加されます。ﾎﾟｰﾄ ﾋﾟﾝ形態設定ﾚｼﾞｽﾀのどれかへの次の書き込みの間に、遮蔽(値)によって設定される全てのﾎﾟｰﾄﾋﾟﾝ形態設定ﾚｼﾞｽﾀは同じ値が書かれます。MPCMASKﾚｼﾞｽﾀはﾋﾟﾝ形態設定ﾚｼﾞｽﾀが完了されるための書き込み操作後に自動的に解除されます。

この課題は出力ﾌﾟﾙｱｯﾌﾟ/ﾀﾞｳﾝ形態設定(OPC)の1つ、ﾜｲﾔｰﾄﾞANDﾌﾟﾙｱｯﾌﾟの実演でもあります。下図は共に接続された2つのﾏｲｸﾛｺﾝﾄﾛｰﾗと外部ﾌﾟﾙｱｯﾌﾟを図解します。図の右の表に於いてﾃﾞｰﾀ出力値(OUT)ﾚｼﾞｽﾀの値に応じて入出力ﾋﾟﾝ駆動部がどう反応するかが分かります。

この課題では1つのﾏｲｸﾛ ｺﾝﾄﾛｰﾗだけを持ち、故にｼﾞｬﾝﾊﾟで2つのﾋﾟﾝを共に接続し、内部ﾌﾟﾙｱｯﾌﾟを使用します。

図11. 代表的なﾜｲﾔｰﾄﾞAND使用法



ﾜｲﾔｰﾄﾞAND時の入出力ﾎﾟｰﾄの動きは次のように形態設定されます。

OUTﾚｼﾞｽﾀ	入出力ﾋﾟﾝ出力
0	Low(0)引き込み
1	Hi-Z

この課題の目的はあなたが以下の方法を知ることです。

- 複数ﾋﾟﾝ形態設定許可ﾚｼﾞｽﾀを使用してください。
- ATMEL XMEGAの入出力ﾎﾟｰﾄの出力とﾌﾟﾙｱｯﾌﾟ/ﾀﾞｳﾝについてもっと理解してください。
- 指定のﾜｲﾔｰﾄﾞAND例を理解してください。

すること

1. MultiConfiguration.apsﾌﾟﾛｼﾞｪｸﾄ ﾌｧｲﾙを開いてAVR Studioで考察してください。
2. ﾎﾟｰﾄDのPD0とPD1を接続するために**ｼﾞｬﾝﾊﾟを使用**してください。
3. ATMELのAVR Studioでﾌﾟﾛｼﾞｪｸﾄを構築してﾃﾞﾊﾞｯｸﾞを開始してください。
4. 図12.で示されるようにwhile繰り返しの前に中断点を追加して走行するためにF5を押してください。中断点を配置するには中断点を欲するｺｰﾄﾞ行にｶｰｿﾙを置いてF9を押してください。

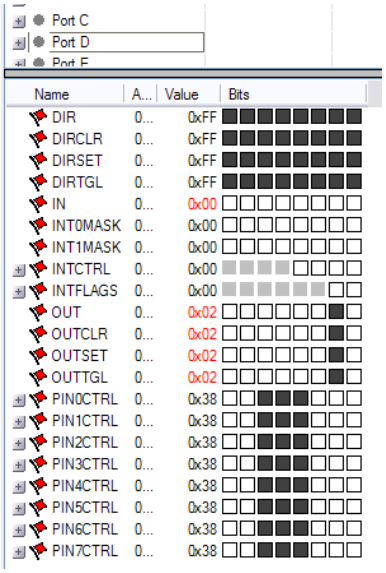
図12. while(1)繰り返し前に中断点追加

```
// same value will be written to all the port's pin confi
PORTCFG.MPCMASK = 0xFF;

// The WIREDANDPULL enables the internal pull-up.
// Also, when the port is configured as output (dir=1),
// setting a pin low will drive low,
// but setting a pin high will set the pin in tri-state
TESTPORT.PIN0CTRL = (TESTPORT.PIN0CTRL & ~PORT_OPC_gm) |

// The MPCMASK register is cleared automatically after th
// configuration registers is finished.

// Connect TESTPORT pin 0 and 1 with a jumper on the boar

while(1)
{
    // Set pins 0 and 1 high
    TESTPORT.OUTSET = PIN0_bm;
    TESTPORT.OUTSET = PIN1_bm;
    // Now, the pins are in tri-state
```

5. I/Oｳｨﾝﾄﾞｳを開いてTESTPORT(例えばﾎﾟｰﾄD)ﾚｼﾞｽﾀ視野を見てください。1行実行(F10)する時にPINnCTRLﾋﾟﾝに何が起きますか?。
6. ﾜｲﾔｰﾄﾞAND設定が何を行うか見てみましょう。ｺｰﾄﾞはﾋﾟﾝ0とﾋﾟﾝ1の両方をHighに設定し、INﾚｼﾞｽﾀでその結果がどうかを調べてください。その後、ﾋﾟﾝ0をLowに設定し、そしてﾋﾟﾝ1に関してINﾚｼﾞｽﾀでその結果を調べてください。
7. while繰り返しを1行実行してI/OｳｨﾝﾄﾞｳでTESETPORT.INﾚｼﾞｽﾀを観測してください。ﾋﾟﾝ0がLowに設定された時にﾋﾟﾝ1のIN値で何が起きますか?。
8. 2つのﾋﾟﾝを共に接続(それは短絡回路)して標準的な形態設定を持つ両ﾋﾟﾝで1つのﾋﾟﾝをHighに他方をLowに形態設定する場合に、ﾋﾟﾝ出力駆動部が違う方向に駆動し、ﾋﾟﾝ間に強い電流の流れを生じることに注意してください。例示のようにﾜｲﾔｰﾄﾞAND形態設定を用いると、これが避けられます。

図13. ﾋﾟﾝ0のLow駆動がﾋﾟﾝ1をLowに駆動

```
// configuration registers is finished.

// Connect TESTPORT pin 0 and 1 with a jumper on the

while(1)
{
    // Set pins 0 and 1 high
    TESTPORT.OUTSET = PIN0_bm;
    TESTPORT.OUTSET = PIN1_bm;
    // Now, the pins are in tri-state
    // The internal pull-up will drive the line high

    // Check state of pin 1
    if(TESTPORT.IN & PIN1_bm)
    {
        // Pin 1 is high
        // No other pins are driving low
        LEDPORT.OUTCLR=0x01;
        nop();
    }

    // Let pin 0 drive the line low
    TESTPORT.OUTCLR = PIN0_bm;

    // Recheck state of pin 1
    if((~TESTPORT.IN) & PIN1_bm)
    {
        // Pin 1 is low
        // Another pin is driving the line low!
        LEDPORT.OUTSET=0x01;
        nop();
    }
}
}
```



9. ﾎﾟｰﾄDからｼﾞｬﾝﾊﾟを取り外してください。

8. 要約

これはあなたがこの課題の最中に学んだ主な特徴/機能のいくつかです。

・ ATMELのXMEGA用のCｺｰﾄﾞの書き方
・ 基本的なﾎﾟｰﾄ形態設定
・ 標準的なﾄﾞﾗｲﾊﾞの作り方
・ 出力とﾌﾟﾙｱｯﾌﾟ/ﾀﾞｳﾝの形態設定
・ ﾋﾟﾝの複数形態設定
・ ﾜｲﾔｰﾄﾞAND

9. 資料

・ XMEAの手引書とﾃﾞｰﾀｼｰﾄ
  ○ http://www.atmel.com/xmega
・ ATMELのﾍﾙﾌﾟ ﾌｧｲﾙ付きAVR studio
  ○ http://www.atmel.com/products/AVR/
・ WINAVR GCCｺﾝﾊﾟｲﾗ
  ○ http://winavr.souceforge.net/
・ ATMEL用IAR Embedded Workbench®ｺﾝﾊﾟｲﾗ
  ○ http://www.iar.com/

10. ATMEL技術支援ｾﾝﾀｰ

ATMELは以下の利用可能な多数の支援ﾁｬﾈﾙを持ちます。

○ ｳｪﾌﾞ入り口	：	http://www.atmel.no/	全てのATMELﾏｲｸﾛ ｺﾝﾄﾛｰﾗ
○ Eﾒｰﾙ	：	avr@atmel.com	全てのATMEL AVR製品
○ Eﾒｰﾙ	：	avr32@atmel.com	全ての32ﾋﾞｯﾄAVR製品

以下のｻｰﾋﾞｽへのｱｸｾｽを得るにはｳｪﾌﾞ入り口で登録してください。

○ 豊富なFAQﾃﾞｰﾀﾍﾞｰｽへのｱｸｾｽ
○ 技術支援要請の容易な依頼
○ あなたの過去の全支援要請の履歴
○ ATMELﾏｲｸﾛ ｺﾝﾄﾛｰﾗ時事通信の受信のための登録
○ 利用可能な練習と練習材料についての情報取得

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

お断り: 本資料内の情報はATMEL製品と関連して提供されています。本資料またはATMEL製品の販売と関連して承諾される何れの知的所有権も禁反言あるいはその逆によって明示的または暗示的に承諾されるものではありません。**ATMELのｳｪﾌﾞｻｲﾄに位置する販売の条件とATMELの定義での詳しい説明を除いて、商品性、特定目的に関する適合性、または適法性の暗黙保証に制限せず、ATMELはそれらを含むその製品に関連する暗示的、明示的または法令による如何なる保証も否認し、何ら責任がないと認識します。たとえATMELがそのような損害賠償の可能性を進言されたとしても、本資料を使用できない、または使用以外で発生する(情報の損失、事業中断、または利益の損失に関する制限なしの損害賠償を含み)直接、間接、必然、偶然、特別、または付随して起こる如何なる損害賠償に対しても決してATMELに責任がないでしょう。**ATMELは本資料の内容の正確さまたは完全性に関して断言または保証を行わず、予告なしでいつでも製品内容と仕様の変更を行う権利を保留します。ATMELはここに含まれた情報を更新することに対してどんな公約も行いません。特に別の方法で提供されなければ、ATMEL製品は車載応用に対して適当ではなく、使用されるべきではありません。ATMEL製品は延命または生命維持を意図した応用での部品としての使用に対して意図、認定、または保証されません。

© Atmel Corporation 2010. **全権利予約済** ATMEL®、ﾛｺﾞとそれらの組み合わせ、AVR®とその他はATMEL Corporationの登録商標、XMEGA®とその他は商標またはその付属物です。他の用語と製品名は一般的に他の商標です。

© *HERO* 2014.

本応用記述はATMELのAVR1500応用記述(doc8308.pdf Rev.8308A-06/10)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。