
tinyAVR® 1系での開始に際して

事前要件

- ・ **ハードウェア事前要件**
 - Microchip ATtiny817 Xplained Pro基板
 - マイクロUSBケーブル (A型/マイクロB)
 - 1本の雄雌線
 - インターネット接続
- ・ **ソフトウェア事前要件**
 - Atmel Studio 7.0
 - Atmel Studio ATtiny_DFP 1.2.112またはそれ以降版
 - 支援されるブラウザの一覧は<http://start.atmel.com/static/help/>⇒Requirements and Compatibility(要件と互換性)⇒Supported Web Browsers(支援されるウェブ ブラウザ)で見つけることができます。
- ・ **予測完了時間 : 120分**

序説

この実地訓練はそれらが提供する豊富なユーザーインターフェースと他の素晴らしい開発ツールと共にAtmel StudioとAtmel STARTでAVR®応用をどう開発するかを実演します。

Atmel STARTはMicrochipマイクロ コントローラ開発の開始を助けます。これは使い易くて最適化された規則でMCUを選び、ソフトウェア構成部品、ドライバ、ミドルウェアと例プロジェクトを組み込み応用へ構成設定することを許します。一旦構成設定が完了すると、Atmel Studioまたは他の第三者開発ツールでプロジェクトを生成することができます。最終製品にプロジェクト外の機能を拡張するのに必要とされるコードを開発するだけでなく、ダウンロードしたコードをコンパイル、書き込み、デバッグするのにもIDEが使われます。

Atmel STARTで、

- ・ ソフトウェアとハードウェアの両要件に基づくMCU選択の手助けを得てください。
- ・ 例を見つけて開発してください。
- ・ ドライバ、ミドルウェア、例プロジェクトを構成設定してください。
- ・ 有効なPINMUX配置の準備で手助けを得てください。
- ・ システム クロック設定を構成設定してください。

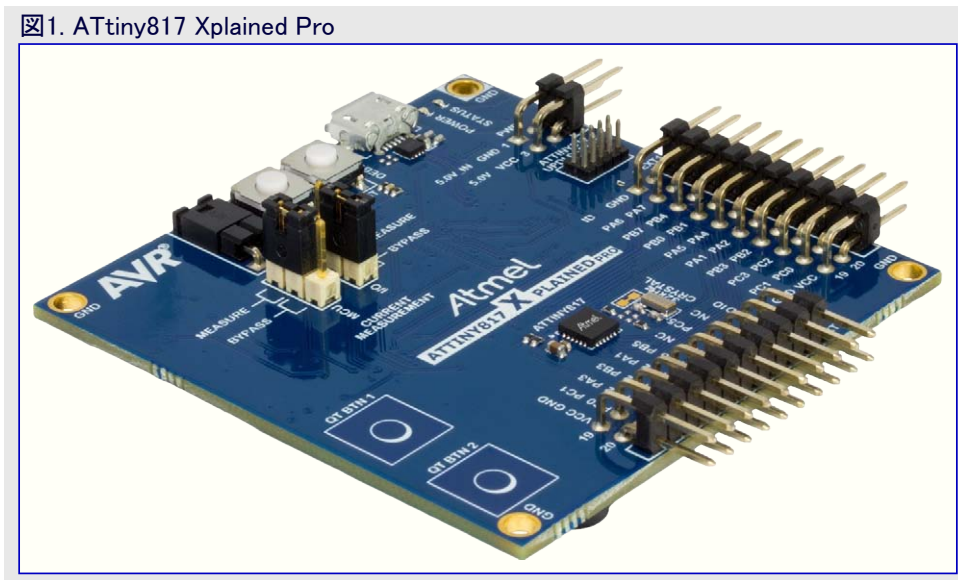
ATtiny817 Xplained Pro評価キットはATtiny817マイクロ コントローラを評価するためのハードウェア基盤です。Atmel Studioと継ぎ目なしの統合を提供する完全に統合された組み込みデバッグをキットに含みます。ATtiny817の機能への容易なアクセスがキットによって許され、お客様の設計に於いてデバイスの簡単な統合を楽にします。

この訓練単位部はAtmel STARTでの応用構成設定、Atmel STARTプロジェクトの再構成設定、Atmel Studio 7での実装継続の方法を実演します。

応用を作成するのに使われる周辺機能はGPIO(汎用入出力)、TCAとTCBの計時器、事象システム、USART、構成設定可能な注文論理回路(CCL:Configurable Custom Logic)、PIT(周期割り込み間隔)です。

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、Microchip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。

図1. ATtiny817 Xplained Pro



以下の話題が網羅されます。

- Atmel STARTでのドライバ構成設定
- PINMUXドライバ構成設定と釦押下でのLED切り替え調査
- タイマ/カウンタ(TCA)を使うことによってPWMを生成してRTC割り込みを使うことによって可変パルス幅を実装
- TCBの捕獲入力を使ってデューティサイクルと周波数を測定
- USART構成設定
- シリアル端末にデータを送るのにデータ可視器(Data Visualizer)ツールを使用
- 構成設定可能な注文論理回路(CCL:Configurable Custom Logic): 簡単な接続論理回路機能用の外部論理回路ゲートを省くことを使用者に許す、デバイスピン、事象、または周辺機能に接続することができる設定可能な論理回路周辺機能。ここではGPIOからの事象を使って特別な信号と2つのPWM信号を生成するようにCCLを構成設定してください。

注: この訓練用解決策プロジェクトはAtmel START⇒[BROWSE EXAMPLES](#)(例閲覧): ‘[Getting Started with tinyAVR® 1-series](#)(AVR® 1系での開始に際して)’で見つけることができます。

目次

事前要件	1
序説	1
1. 関連デバイス	4
1.1. tinyAVR 1系統	4
2. アイコン鍵識別子	4
3. 課題1：LED切り替え応用	5
3.1. Atmel STARTプロジェクト作成	5
3.2. Atmel STARTプロジェクト概要	8
3.3. コード開発	9
3.4. 応用のデバッグ	10
4. 課題2：PWM生成、デューティサイクルと周波数の測定	12
4.1. TCAドライバ	12
4.2. RTCドライバ	16
4.3. TCBドライバ	17
4.4. 事象システムドライバ	19
4.5. USARTドライバ	19
4.6. プロジェクト生成、コード開発	20
5. 課題3：2値周波数移動符号化の基礎	25
5.1. CCLドライバ	26
5.2. 事象システムドライバ	27
5.3. PITドライバ	28
5.4. プロジェクト生成、コード走行	29
6. 結び	30
7. Atmel STARTからのソースコード取得	30
8. 改訂履歴	30
Microchipウェブ サイト	31
お客様への変更通知サービス	31
お客様支援	31
Microchipデバイスコード保護機能	31
法的通知	31
商標	32
DNVによって認証された品質管理システム	32
世界的な販売とサービス	33

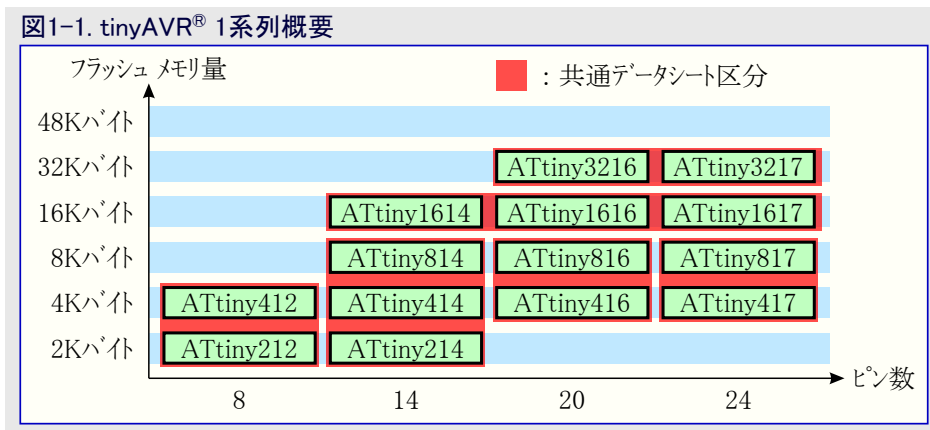
1. 関連デバイス

本章はこの資料に関連するデバイスを一覧にします。

1.1. tinyAVR 1系統

下図はピン配置変種とメモリ量を展開してtinyAVR® 1系統デバイスを示します。

- これらのデバイスがピ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ド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ド変更が必要かもしれません。
- 左への水平方向移植はピン数、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デバイスは一般的に異なるSRAMとEEPROMの量を持ちます。

2. アイコン鍵識別子

以下のアイコンは各種課題部分を識別して複雑さを減らすためにこの資料で使われるアイコンです。

情報: 特定の話題についての脈絡上の情報を伝えます。

助言: 有用な助言と技術を強調します。

行うこと: 完了されるべき目標を強調します。

結果: 課題の段階の予測される結果を強調します。

警告: 重要な情報を表示します。

実行: 必要とされる時に目的対象の中から実行されるべき行動を強調します。

3. 課題1 : LED切り替え応用

基板上的の押し釦を使ってLEDを制御する応用が開発されます。LEDは釦押下でOFF、既定状態はLED ONです。プロジェクトはPINMUXドライバ構成設定とクロック構成設定を使ってAtmel STARTで構成設定され、対応するAtmel Studio 7プロジェクトの生成が続きます。

コードはAtmel START構成設定によって生成されたPINMUXドライバ関数をつかってAtmel Studio 7で開発されます。

ATtiny817 Xplained Pro基板ではLED0がPB4ピンに接続され、押し釦(SW0)がPB5ピンに接続されます。

応用に対して、

- 使われる周辺機能 : GPIO L(PB4、PB5)
- クロック : 3.33MHz

3.1. Atmel STARTプロジェクト作成

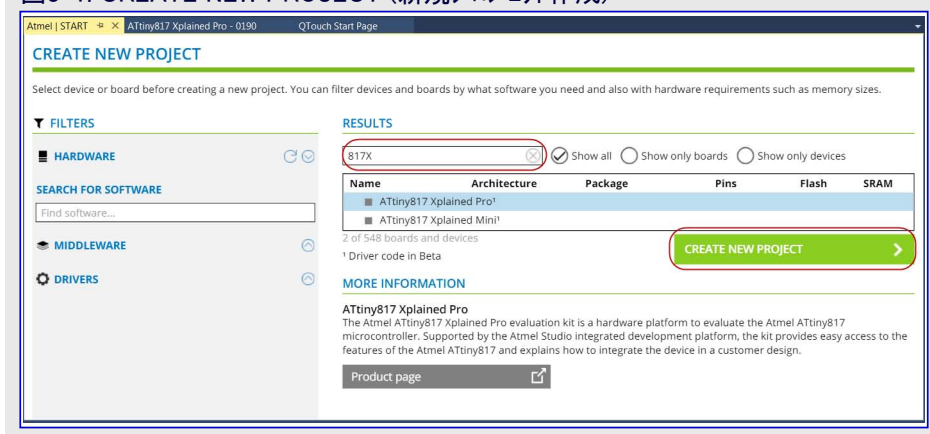
Atmel STARTでPINMUXドライバとCLOCKを構成設定してプロジェクトを作成してください。



行うこと: 新しいAtmel STARTプロジェクトを作成してください。

1. Atmel Studioを開いてください。
2. File(ファイル)⇒New(新規)⇒Atmel Start project(Atmel STARTプロジェクト)を選んでください。
3. Atmel Studio 7内にCREATE NEW PROJECT(新規プロジェクト作成)ウィンドウが現れます。下で示されるように、”Filter on device... (デバイスで選別)”文字枠で817Xを入力し、その後に一覧からATtiny817 Xplained Proを選んでATtiny817 Xplained Proが強調表示されているのを確認し、その後にCREATE NEW PROJECT(新規プロジェクト作成)をクリックしてください。

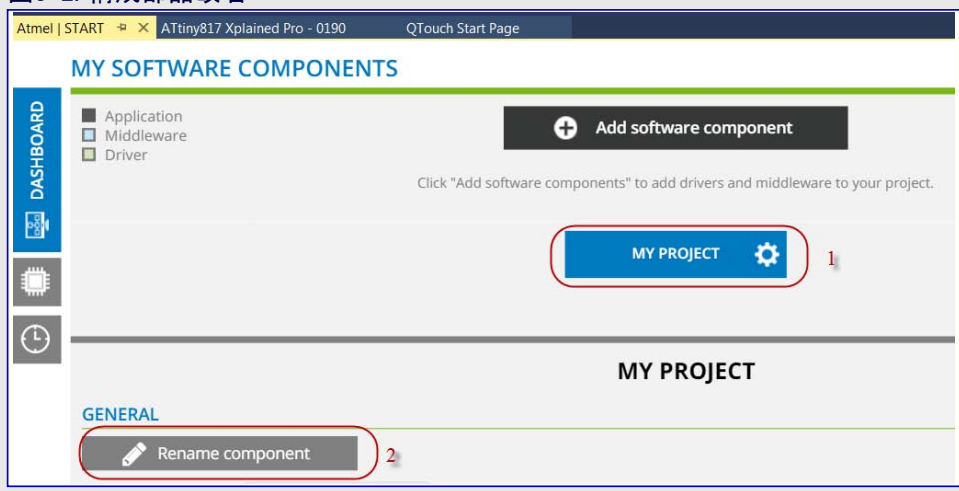
図3-1. CREATE NEW PROJECT (新規プロジェクト作成)




情報: 今やMY SOFTWARE COMPONENTS(私のソフトウェア構成部品)ウィンドウが現れます。

4. MY SOFTWARE COMPONENTS(私のソフトウェア構成部品)ウィンドウで、
 - 1. MY PROJECT(私のプロジェクト)をクリックしてください。
 - 2. Rename component(構成部品改名)を選んでください。

図3-2. 構成部品改名



i 情報: 今やRENAME COMPONENT(構成部品改名)ウィンドウが現れるでしょう。

5. RENAME COMPONENT(構成部品改名)ウィンドウで新しいプロジェクト名を”Assignment_ATtiny817”として指定してRename(改名)を選んでください。
6. 次に、PINMUX構成設定について、ウィンドウの左側の誘導タブで  をクリックしてください。

i 情報: PINMUX構成設定部は選んだデバイス外圍器の図を表示します。これは各種周辺機能によって現在どのピンが使われているかを示します。GPIOピンはここで構成設定することができます。

i 情報: ここでPB4はLED0として、PB5はSW0として構成設定されます。構成設定は(下図で1,2,3,4と赤く番号付けで記される)4つの段階で示されます。

7. PB4の構成設定

- 1. PB4をクリックしてください。
- 2. “User label:(使用者標識:)”をLED0として入力してください。
- 3. “Pin mode:(ピン動作形態:)”をDigital output(デジタル出力)として入力してください。
- 4. “Initial level:(初期水準:)”をLowとして選んでください。




8. PB5の構成設定

- 1. PB5をクリックしてください。
- 2. “User label:(使用者標識:)”をSW0として入力してください。
- 3. “Pin mode:(ピン動作形態:)”をDigital input(デジタル入力)として入力してください。
- 4. “Initial level:(初期水準:)”をPull-up(プルアップ)として選んでください。



i 情報: ATtiny817 Xplained Proに関連する技術資料はAtmel StudioでATtiny817 Xplained Pro⇒Technical Documentation(技術資料)の頁からダウンロードすることができます。ATtiny817 Xplained Proの頁は一旦ATtiny817 Xplained Pro基板がコンピュータに接続されると表示されます。

9. **CLOCK CONFIGURATOR**(クロック構成設定部) : 次に、クロック構成設定について、ウインドウの左側の誘導タブで  をクリックしてください。

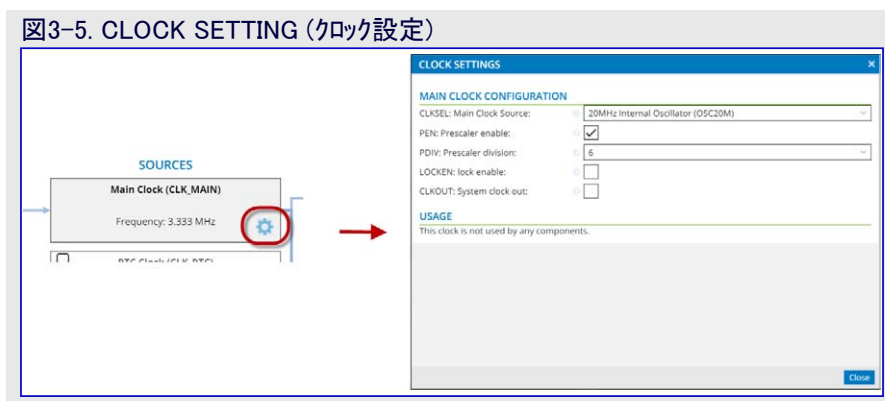
情報: 今や**CLOCK CONFIGURATOR**(クロック構成設定部)ウインドウが現れます。これは各種形式の発振器とクロック供給元から成ります。必要なクロック元を選ぶことができ、計算された出力周波数が表示されます。


- **OSCILLATORS**(発振器)部分は選んだデバイスに対して利用可能な発振器を表示します。発振器パラメータは”設定ダイアログ”(歯車アイコン)を選ぶことによって構成設定することができます。

- **SOURCES**(供給元)部分は入力信号を選んで倍率変更することによってクロック周波数を構成設定するのに使われます。


10. 下図で示されるように、**SOURCES**(供給元)から既定**Main clock**(主クロック)設定を見るために”設定ダイアログ”(歯車アイコン)をクリックしてください。

情報: **CLOCK SETTING**(クロック設定)ウインドウが表示されるでしょう。

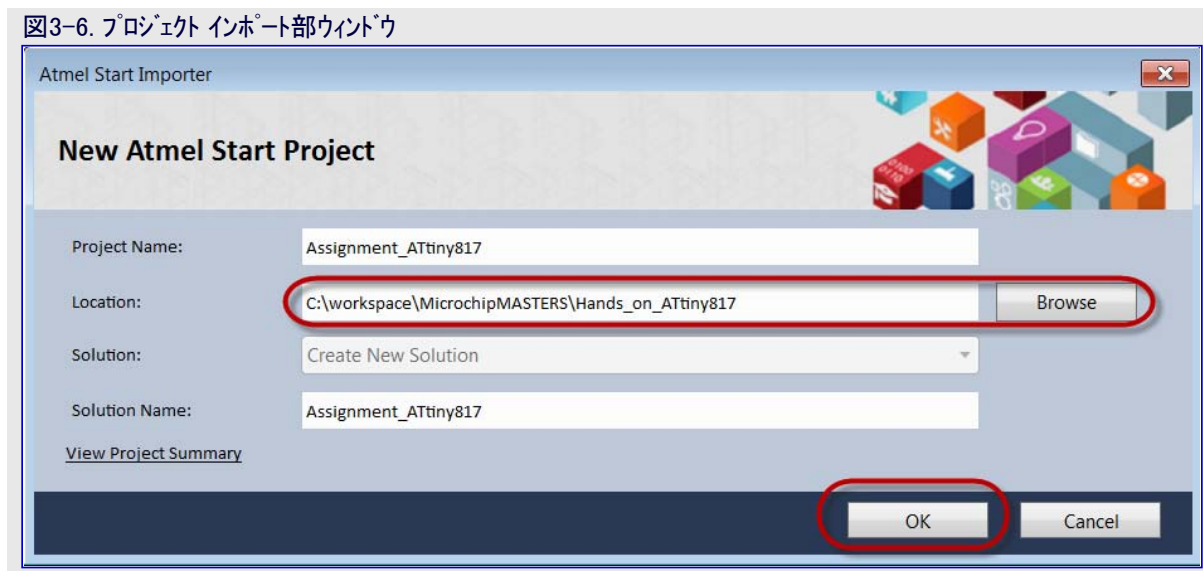


情報: この応用の既定クロック設定がそのまま維持されます。ここでは主クロック元が20MHz OSCで、前置分周器は6分周です。結果のCPUクロック周波数は3.33MHzです。各構成設定傍らの”疑問符”  のクリックは個別ビット設定のデータシート説明へ案内します。

11. **CLOCK SETTING**(クロック設定)ウインドウで**Close**(閉じる)をクリックしてください。

12. 次に、**GENERATE PROJECT**(プロジェクト生成)  **GENERATE PROJECT** 鈕をクリックしてください。

13. 下図で示されるように、プロジェクトが格納されるべき場所の望むパスを選び、その後に**OK**をクリックしてください。



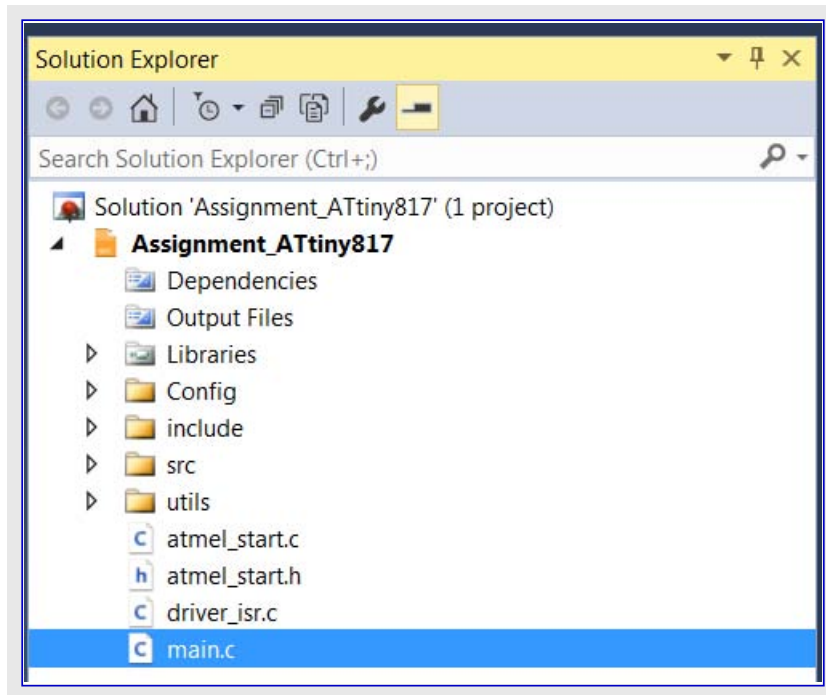
結果: Atmel STARTプロジェクトが作成されます。

3.2. Atmel STARTプロジェクト概要

Atmel STARTで構成設定されたプロジェクトは周辺機能ドライバ関数とファイルだけでなく、全てのドライバを初期化するmain()関数も生成します。


Atmel STARTによって生成されたフォルダとファイルについては以下のとおりです。

- **Config**フォルダはクロック構成設定を含みます。F_CPUはclock_config.hで定義されます。
- ドライバのヘッダとソースのファイルはsrcとincludeのフォルダで見つかります。
- **include**フォルダのatmel_start_pins.hファイルはPINMUXドライバ関数を含みます。
- **utils**フォルダは一般的にドライバと応用によって使われるいくつかの関数を定義するファイルを含みます。
- **atmel_start.c**ファイルではatmel_start_init()関数がプロジェクト内のMCU、ドライバ、ミドルウェアを初期化します。
- **driver_isr.c**ファイルはプロジェクトの構成設定で割り込みが許可された場合のISR(割り込み処理ルーチン)を含みます。




 **行うこと:** Atmel STARTプロジェクトの概要を得てください。

1. Assignment_ATtiny817プロジェクトで、Solution Explorer(解決策エクスプローラ)ウィンドウからmain.cファイルをダブルクリックしてください。
2. atmel_start_init関数、その後に右クリック⇒Goto implementation(実装へ行く)を選んでください。関数定義へ案内するためにsystem_init()関数に対してこの手順を繰り返してください。

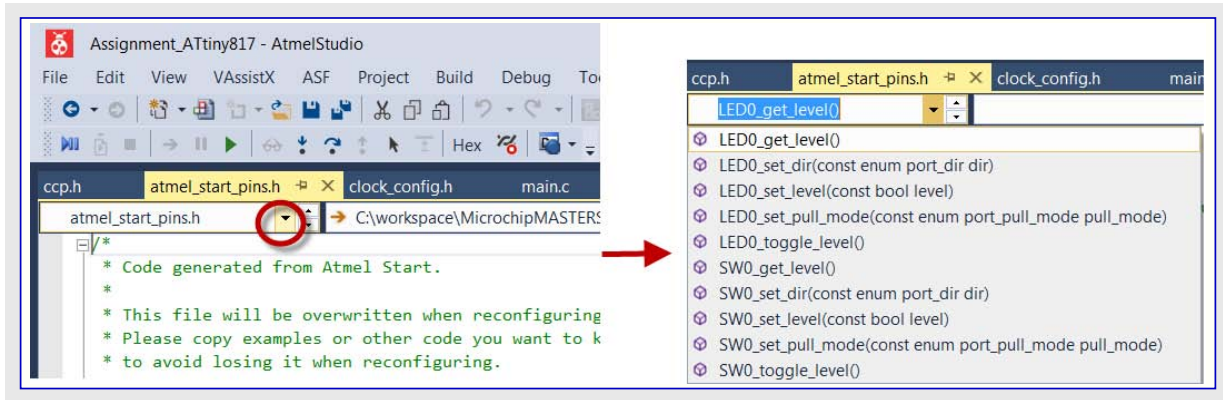
 **情報:** mcu_init()関数は電力消費を減らすために全てのピンで内部プルアップ抵抗を許可します。全てのドライバ初期化関数はsystem_init()関数から呼ばれます。また、LED0とSW0のポートピンは初期ピン状態で出力と入力に構成設定されません。

3. CLKCTRL_init()の実装へ行き、既定クロック設定が注釈にされていることを観察してください。

 **助言:** driver_init.c⇒system_init()⇒CLKCTRL_init()

 **情報:** Atmel STARTで既定以外のクロック設定が選ばれた場合、それはCLKCTRL_init()に反映されます。

4. `atmel_start_pins.h`を開き、このファイルで定義された関数の一覧を開くために下図で示されるように下向き矢印をクリックしてください。多くの有用なGPIO関数が生成されていることを観察してください。



結果: Atmel STARTプロジェクト概要は完了です。

3.3. コード開発

ATtiny817 Xplained Pto基板に於いて、LED0とSW0に関連付けられたピンの動きは次のとおりです。

表3-1. LED0とSW0に関連づけられたピンの動き

SW0/LED0	状態	ピン値
SW0	釦押下	PB5 : Low
	釦開放 (既定状態)	PB5 : High
LED0	ON	PB4 : Low
	OFF	PB4 : High

行うこと: 釦が押下された時にLEDをOFFに切り替え、釦が解放される時にLEDをONに戻すコードを書いてください。

1. Assignment_ATtiny817プロジェクトで`main.c`ファイルを開いてください。
2. SW0の状態を読んで下で言及するようにLED0の状態を構成設定するようにwhile (1)内にコードを挿入してください。
 - 釦(SW0)が押下された時にLED0はOFFです(LED0ピンのレベル=High)。
 - 釦(SW0)が開放された時にLED0はONです(LED0ピンのレベル=Low)。

```

if(!SW0_get_level())           // (SW0)釦押下、PB5はLow
{
    LED0_set_level(true);      // LED0はOFFにされます。PB4はHigh
    while(!SW0_get_level());   // (SW0)釦開放まで待ち
    LED0_set_level(false);     // (SW0)釦開放、LED0はONにされます。PB4はLow
}
    
```

情報: `SW0_get_level()`関数はSW0ピンの状態を返します。`LED0_set_level(true)`関数はLED0のレベルをHighに設定します。

3. コード完了後、解決策を構築するためにF7を押してください。構築は異常なしで成功裏に終わるべきです。

結果: コードが右で示される画像のように見えるべきです。

図3-7. 課題1のコード


```

#include <atmel_start.h>

int main(void)
{
    /* Initializes MCU, drivers and middleware */
    atmel_start_init();

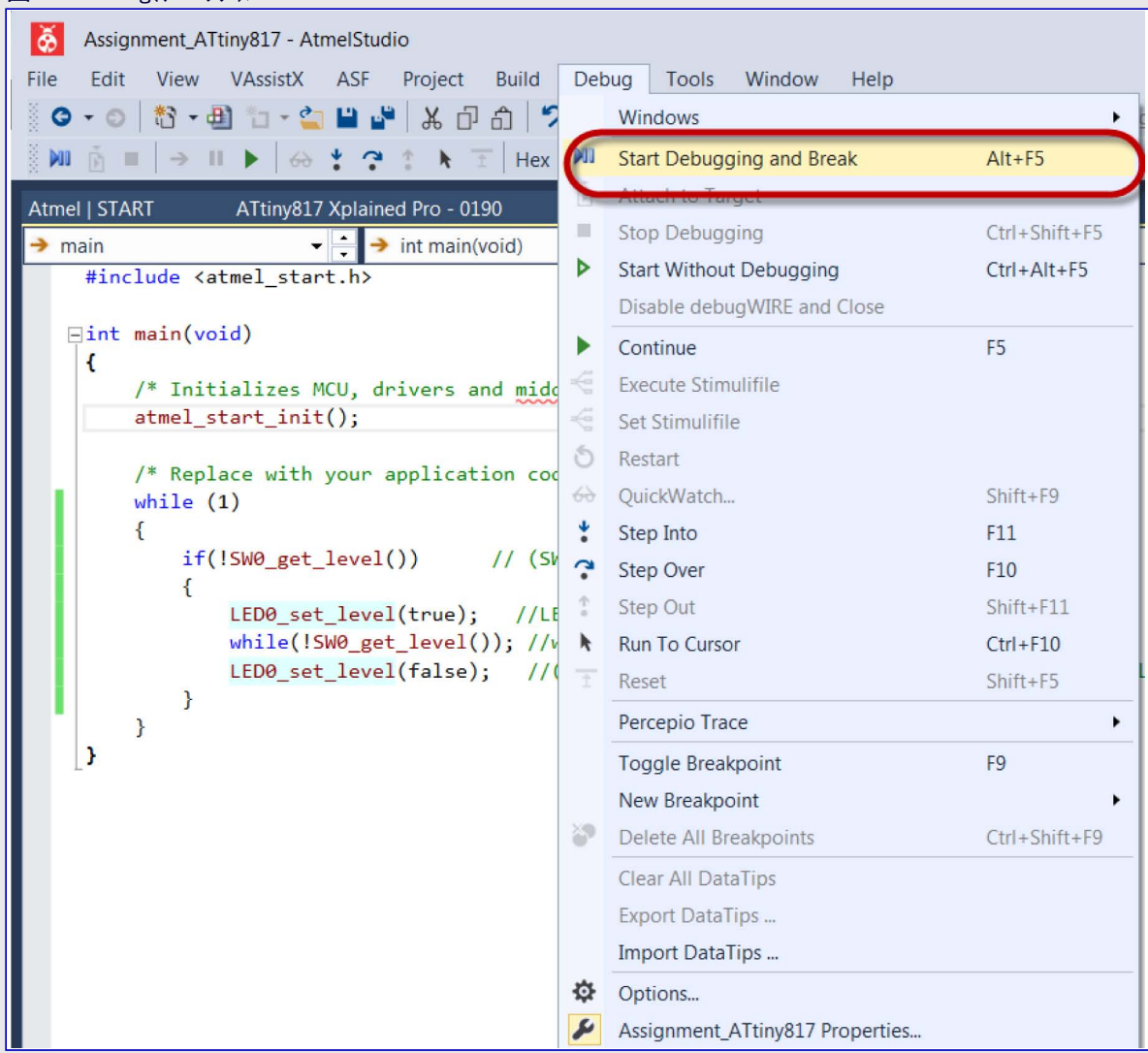
    /* Replace with your application code */
    while (1)
    {
        if(!SW0_get_level())    // (SW0)button pressed. PB5 is low
        {
            LED0_set_level(true); //LED0 is turned OFF : PB4 is high
            while(!SW0_get_level()); //wait till (SW0)button release.
            LED0_set_level(false); // (SW0)button released. LED0 is turned ON : PB4 is low
        }
    }
}
    
```


3.4. 応用のデバッグ

 行うこと: ‘課題1:LED切り替え’ 応用をデバッグしてください。

1. マイクロUSBケーブルでコンピュータにATtiny817 Xplained Pro基板を接続することによって通電してください。
2. Debug(デバッグ)⇒Start Debugging and Break(デバッグ開始と中断)(またはAlt+F5)を選んでください。Programmer/Debugger(書き込み器/デバッガ)としてATtiny817 Xplained Pro EDBGを選ぶため、指示に従って再びAlt+F5を押してください。

図3-8. Debug(デバッグ)メニュー



 **情報:** 基板上デバッガのファームウェア版がAtmel Studioインストール時のものよりも古い場合、ファームウェア更新が問われます。

Upgrade(更新)を選んでください。進捗バーが完了の時にClose(閉じる)を選んでください。次に、Debug(デバッグ)⇒Start Debugging and Break(デバッグ開始と中断)(代替:Alt+F5)を選ぶことによってデバッグを始めてください。


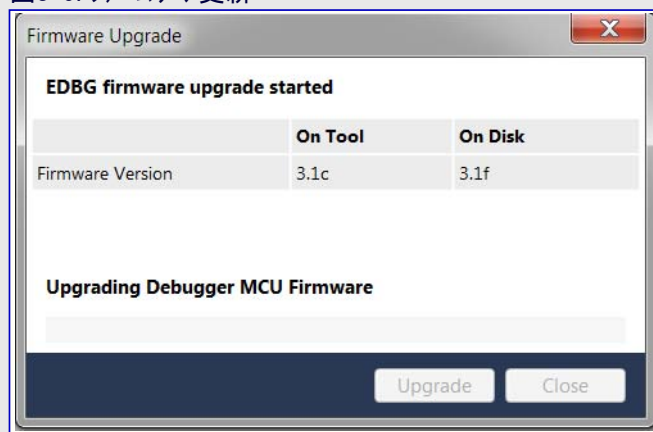
 **結果:** デバッグが開始されてmain()で中断します。今やデバッグを始めることができます。

図3-9. ファームウェア更新



3. 右図で示されるように中断点(ブレークポイント)を挿入するために余白をクリックしてください。

図3-10. 挿入された中断点

```

int main(void)
{
    /* Initializes MCU, drivers and middleware */
    atmel_start_init();

    /* Replace with your application code */
    while (1)
    {
        if(!SW0_get_level()) // (SW0)button pressed. f
        {
            LED0_set_level(true); //LED0 is turned OFF :
            while(!SW0_get_level()); //wait till (SW0)butto
            LED0_set_level(false); //((SW0)button release
        }
    }
}
    
```

4. **Debug**(デバッグ)⇒**Continue**(継続)をクリックすることによって中断点へ走らせてください。



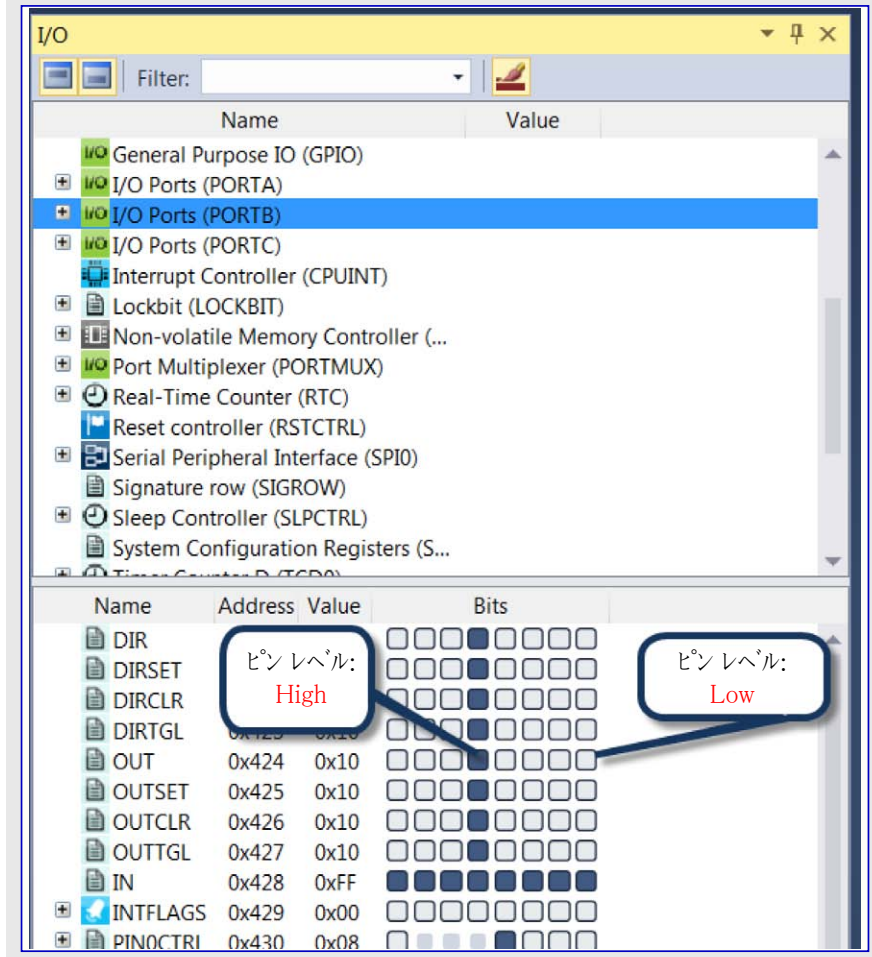
助言: デバッグを開始または継続するのにAtmel Studio 7ウインドウ上部近くのツールバーに置かれた再生 ▶ 鈕を使うことができます。F5キーボード ショートカットを使うこともできます。

5. **SW0**押し鈕を押して実行が中断点で停止するのを観察してください。
 6. 全てのポートピンの状態を調べるため、**Debug**(デバッグ)⇒**Windows**(ウインドウ)⇒**I/O**(入出力)を選ぶことによってI/O表示ウインドウを開き、下の画像で示されるようにPORTBレジスタ群をクリックしてください。



情報: I/O表示部で、全ての周辺機能の状態を観察することができます。ピンの状態は**OUT**レジスタ内のBits(ビット)列下で左から右にPB7～PB0で示されます。下の画像で示されるように、PB4(**LED0**)のレベルは塗り潰された四角がビット状態**1**に対応するためHighです。空の四角はビット状態**0**に対応します。

図3-11. I/O表示部



7. F10を押すことによって単一段階デバッグを行い、PB4の状態を観察してください。



情報: PB7が一番左でPB0が一番右です。

8. 中断点上をクリックすることによってそれを取り去り、Debug(デバッグ)⇒Continue(継続)をクリックするか、または▶を選ぶことによってコードを走らせてください。
9. ■を選ぶことによってデバッグを停止してください。
10. SW0押し釦を押し、LED0切り替えを観察してください。



情報: 応用は成功裏にATtiny817 Xplained Pro基板に書かれています。

4. 課題2 : PWM生成、デューティサイクルと周波数の測定

ATtiny817は2つの16ビット タイマ/カウンタの実体であるTCAとTCB、1つの12ビット タイマ/カウンタ、TCDを持ちます。

ここでは、応用がTCAを使ってPWMを生成するように開発されます。PWMのデューティサイクルを変えるのにRTC割り込みが使われま

す。TCA波形出力は事象システムを通してTCBへの入力として使われ、波形のデューティサイクルと周波数を測定するのにTCBの捕獲入力動作が使われます。測定したデータはUSARTを通して端末へ送られます。

TCAに対して単一傾斜PWM生成動作が使われます。ここでは、周期が定期(PER)レジスタによって制御され、同時に比較n(CMPn)レジスタの値が波形生成(WG)した出力(WOn)のデューティサイクルを制御します。波形の周期やパルス幅を設定するために計数器値はCMPnレジスタ及びPERレジスタと比較されます。

「課題1:LED切り替え(Assignment_ATtiny817)」からのAtmel STARTプロジェクトはTCA、TCB、事象システム、RTC、USART用のドライバを追加するために再構成設定されます。

この応用に対して使われる周辺機能は次のとおりです。

- TCA (PB0でのWO0波形出力)
- TCB
- 事象システム (PA5ピンからの事象入力)
- RTC
- USART

クロックは次のとおりです。

- 3.33MHz主クロック
- 1kHz RTCクロック

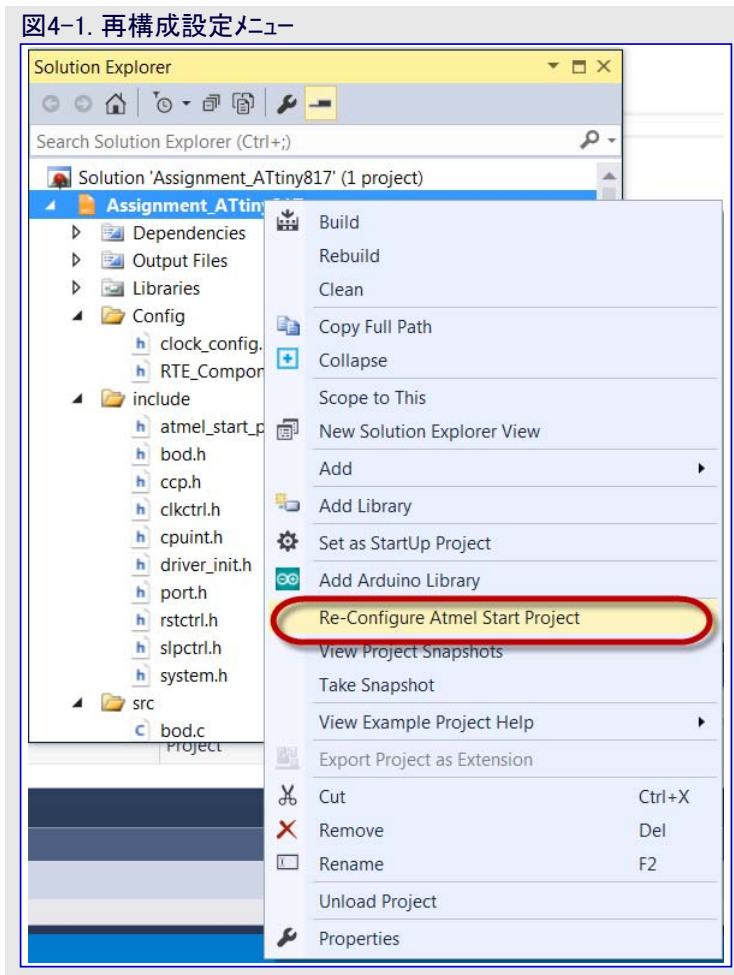
4.1. TCAドライバ



行うこと: PWM信号を生成するためにTCAドライバを追加してください。

1. “課題1:LED切り替えプロジェクトのAssignment_ATtiny817を開いてください。
2. 次に示されるようにSolution Explorer(解決策エクスプローラ)ウィンドウでプロジェクト名のAssignment_ATtiny817を、その後に右クリック⇒Re-Configure Atmel Start Project(Atmel STARTプロジェクト再構成設定)をクリックしてください。

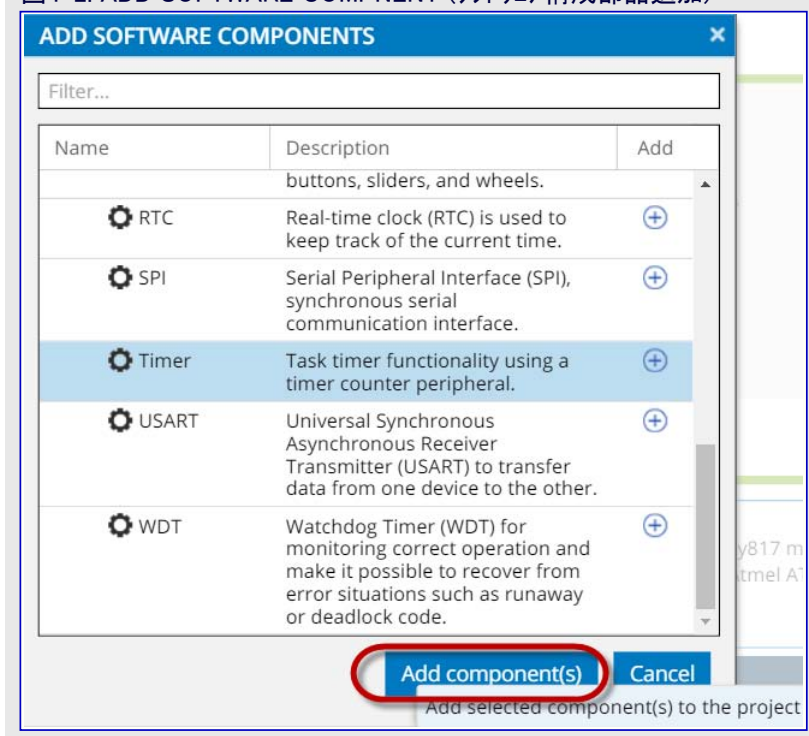
図4-1. 再構成設定メニュー



情報: “Atmel | START”ウィンドウが現れます。

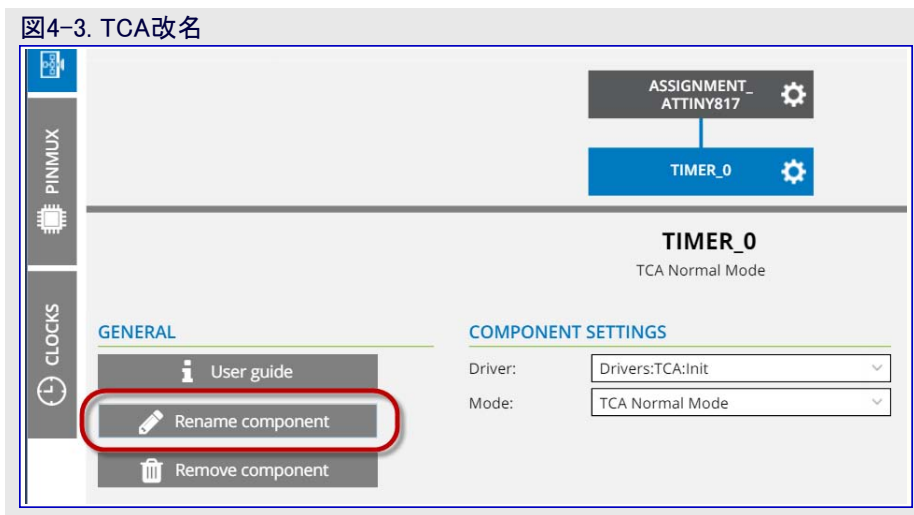
3. **Add software component** 枠をクリックし、その後に**ADD SOFTWARE COMPONENTS**(ソフトウェア構成部品追加)から**Drivers**(ドライバ)を展開してください。
4. 右図で示されるように、**Timer**(計時器)ドライバを検索し、それを選択してその後に**Add Component(s)**(構成部品追加)をクリックしてください。

図4-2. ADD SOFTWARE COMPONENT (ソフトウェア構成部品追加)



情報: プロジェクトに**Timer_0**ドライバが追加されま

5. TIMER_0をクリックし、その後に下図で示されるようにRename component(構成部品改名)をクリックしてください。



6. 新しい名前をTCA_PWMとして指定してRename(改名)をクリックしてください。

情報: 今やTCA周辺機能構成設定はPWM信号の生成が行われることが必要です。これは下で示されます。

行うこと: PB0ピンで周波数:32kHz、初期デューティサイクル:10のPWM(WO)を生成するようにTCAドライバを構成設定してください。

情報: 構成設定は図4-4.で1~8の番号付けられた赤色印で示されます。

1. “Driver:(ドライバ:)”を”TCA:Init”とし、“Mode:(動作形態:)”を”TCA Normal Mode(TCA標準動作)”として選んでください。

情報: ここでは既定で“Driver:(ドライバ:)”が”TCA:Init”として選ばれ、“Mode:(動作形態:)”は”TCA Normal Mode(TCA標準動作)”として選ばれます。この計時器は“Normal Mode(標準動作)”(1つの16ビットタイマ/カウンタ)と“Split Mode(分割動作)”の2つの動作形態を持ちます。分割動作では16ビットタイマ/カウンタが各々PWM生成用に3つの比較チャネルを持つ2つの独立した8ビット計時器として働きます。

2. PB0での波形出力WO/0を選んでください。

助言: TCAのWO/0出力ピンを確認するにはデータシートの「入出力多重化と考察」項を参照してください。TCA WO/0ピンを調べるにはどれかの構成設定傍らの”疑問符”をクリックして「入出力多重化と考察」項へ行ってください。

3. “ENABLE:Module Enable(ENABLE:単位部許可)”のチェック印を選ぶことによってこの周辺機能を許可してください。

4. タイマ/カウンタ用クロック周波数を”System Clock(システムクロック)”として選んでください。

情報: ここでは既定で(前置分周なしの)”System Clock(システムクロック)”として選ばれています。

5. “PER:Period(PER:定期)”レジスタ値を100(または0x64)として入力してください。

情報: これはタイマ/カウンタの16ビットTOP値を含みます。これはPWM周波数を決めます。

6. WO出力用PORT出力設定を得るために“CMP0EN:Compare 0 Enable(CMP0EN:比較0許可)”のチェック印を選んでください。

7. “CMP0:Compare Register 0(CMP0:比較0レジスタ)”値を10(または0xA)として入力してください。

情報: このレジスタは計時器の計数値と継続的に比較されます。通常、比較器からの出力はその後に波形を生成するのに使われます。このレジスタはPWMのデューティサイクルを決めます。初期デューティサイクルは10(0xA)として選ばれています。WO/0波形出力が必要とされるので、ここで比較チャネル0が構成設定されます。

8. “WGMODE:waveform generation mode(WGMODE:波形生成動作形態)”を”Single Slope PWM(単一傾斜PWM)”として選んでください。



情報: 波形生成動作形態は計数器の計数の流れ、TOP値、UPDATE(更新)条件、割り込み条件、生成される波形の形式を制御します。この単一傾斜PWM動作では、周期がTCA.PERによって制御され、同時にTCA.CMPnの値がWG出力のデューティサイクルを制御します。単一傾斜PWM周波数(f_{PWM_SS})は周期設定(TCA.PER)、システムの周辺機能クロック周波数(f_{CLK_PER})、TCAクロック前置分周器(CLKSEL:clock selection(CLKSEL:クロック選択))に依存します。これは次式によって計算されます。


$$f_{PWM_SS} = \frac{f_{CLK_PER}}{N \times (PER+1)} \quad \text{ここで } N=1, PER=100 \text{ で、} f_{PWM_SS}=3.3\text{MHz} \text{ です。}$$

図4-4. TCA構成設定




結果: TCA構成設定は完了されます。

4.2. RTCドライバ


 **行うこと:** 溢れ割り込みを生成するためにRTCドライバを追加してください。

1. Add software component(ソフトウェア構成部品追加)枠をクリックしてください。
2. ADD SOFTWARE COMPONENTS(ソフトウェア構成部品追加)ウィンドウからDrivers(ドライバ)を展開してください。RTCドライバを検索してそれを選択し、その後にAdd Component(s)(構成部品追加)をクリックしてください(「TCADドライバ」で段階3と4を参照してください)。


注: ここでRTCドライバを追加してください。

 **情報:** RTC_0ドライバがAssignment_ATtiny817プロジェクトに追加されます。今やRTC周辺機能構成設定が下で示されるように行われることが必要です。


 **行うこと:** 500ms毎に溢れ割り込みを生成するためにRTC_0枠をクリックしてRTCドライバを構成設定してください。

 **情報:** 構成設定は図4-5.で1~6の番号付けられた赤色印で示されます。

1. “RTCCEN:Enable(RTCCEN:許可)”のチェック枠を選ぶことによってこの周辺機能を許可してください。
2. “RTC Clock Source Selection(RTCクロック元選択)”を”32kHz Internal Ultra-Low Power oscillator (OSCULP32K)(32kHz内部超低電力発振器 (OSCULP32K))”として選んでください。
3. “PRESCALER:Prescaling Factor(PRESCALER:前置分周係数)”を500(または0x1F4)として選んでください。

 **助言:** 結果のRTCクロックは(ウィンドウの右側の)CLOCKS(クロック)誘導釘を選ぶことによって確認することができます。ドライバ構成設定に戻るにはDASHBOARD(計器盤)を選んでください。

4. “PER:Period(PER:定期)”レジスタ値を500(または0x1F4)として選んでください。

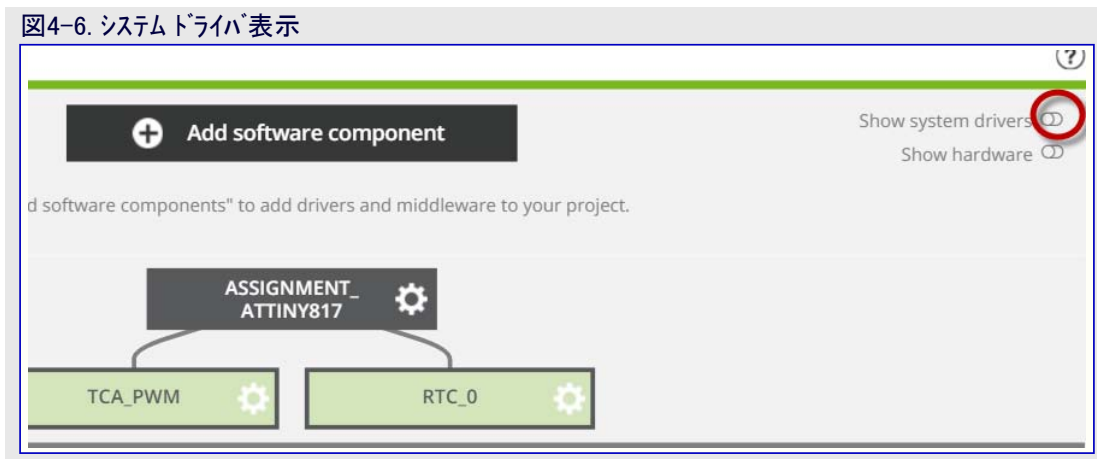
 **情報:** これは16ビット定期レジスタです。計数器レジスタがこの値に達すると、溢れ割り込みが設定され、計数器レジスタがリセットされます。

5. “Include ISR harness in driver_isr.c(driver_isr.cにISR利用を含む)”チェック枠を選ぶことによってdriver_isr.cにISRコードを生成してください。
6. “OVF:Overflow Interrupt enable(OVF:溢れ割り込み許可)”を選ぶことによって溢れ割り込みを許可してください。

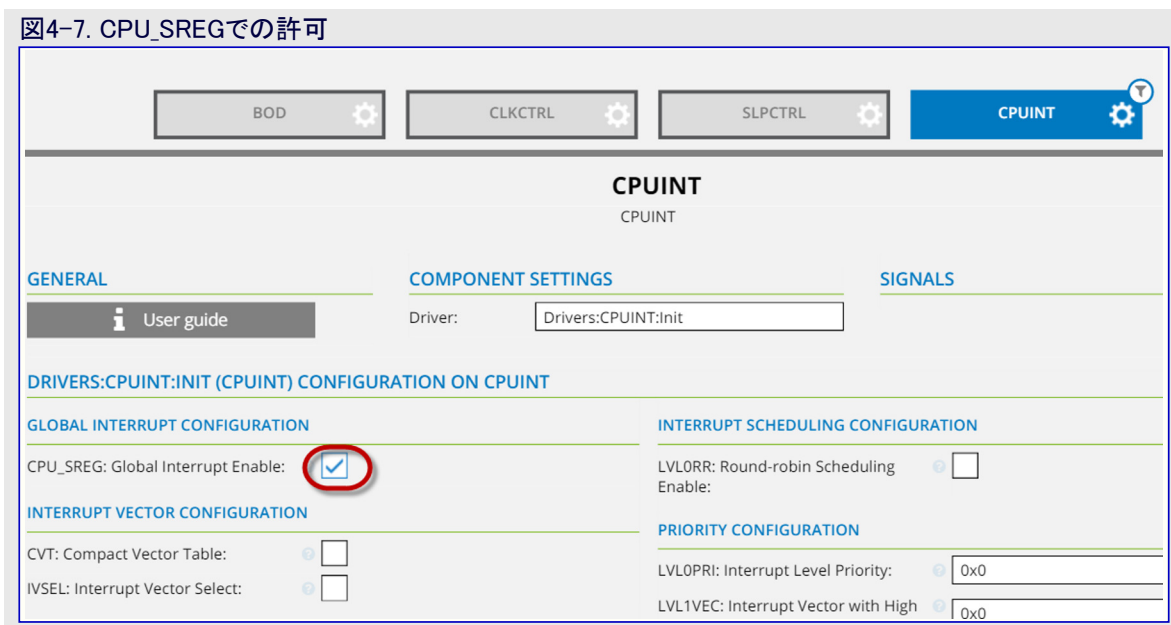


情報: 割り込みを生成するにはステータスレジスタ(SREG)の全体割り込み許可(I)ビットを許可することが必要とされます。

7. 下図で示されるように、“Show system driver(システムドライバ表示)”で小さな円を右側へクリック移動してください。



8. CPUINT(CPU割り込み)をクリックしてその後下図で示されるようにCPU_SREG:Global Interrupt Enable(CPU_REG:全体割り込み許可)チェック枠を選んでください。



結果: RTC構成設定が完了されます。

4.3. TCBドライバ

TCB周辺機能は捕獲機能を試験するためにPWM信号を捕獲するように構成設定されます。

行うこと: TCBドライバを追加してください。

1. Add software component(ソフトウェア構成部品追加)枠をクリックしてください。
2. ADD SOFTWARE COMPONENTS(ソフトウェア構成部品追加)ウィンドウからDrivers(ドライバ)を展開してください。
3. Timer(計時器)ドライバを検索し、それを選択してその後Add Component(s)(構成部品追加)をクリックしてください(「TCBドライバ」で段階3と4を参照してください)。

情報: TIMER_0ドライバがAssignment_ATtiny817プロジェクトに追加されます。

4. TIMER_0をクリックし、その後Rename component(構成部品改名)をクリックしてください。新しい名前をTCB_Duty_Frequency_Measureとして指定してRename(改名)をクリックしてください(「TCBドライバ」で段階5と6を参照してください)。

情報: 今やTCB周辺機能構成設定は捕獲入力動作を行われることが必要です。

行うこと: 与えられたPWM入力のデューティ サイクルと周波数を測定するようにTCBドライブを捕獲入力動作で構成設定してください。

情報: 構成設定は図4-8.で1~5の番号付けられた赤色印で示されます。

1. “Driver:(ドライブ:)”を”TCB:Init”として選んでください。

情報: ここでは既定で“Driver:(ドライブ:)”が”TCB:Init”として選ばれます。

2. “ENABLE:Enable(ENABLE:許可)”のチェック枠を選ぶことによってこの周辺機能を許可してください。
3. CLKSEL:Clock Select(CLKSEL:クロック選択)を”CLK_PER (No Prescaling)(CLK_PER(前置分周なし))”として選んでください。

情報: ここで、この周辺機能用クロック元は前置分周なしで選ばれます。この周辺機能はシステムの周辺機能クロックCLK_PERを使います。

4. CNTMODE:Timer mode(CNTMODE:計時器動作形態)をCapture Frequency and Pulse-Width measurement(周波数捕獲とパルス幅測定)として選んでください。
5. CAPTEI:Event Input Enable(CAPTEI:事象入力許可)チェック枠を選んでください。

情報: TCBへの事象入力としてPWM信号が与えられるために事象入力が許可されます。”EVENT CONFIGURATION(事象構成設定)”で、EDGE(端)は選ばれず(非選択)、故に計時器は事象入力信号で正端が検出される時に計数を開始します。後続する下降端で計数値が捕獲されます。計数器は事象入力信号の2つ目の上昇端が検出される時に停止します。EDGE(端)が選ばれる(選択)の場合、計時器は負端が検出される時に計数を開始します。

注: 事象システム構成設定は次の項で示されます。

図4-8. TCB構成設定

The screenshot shows the configuration interface for the TCB driver. The title is "TCB_DUTY_FREQUENCY_MEASURE" with the subtitle "TCB driver".


- GENERAL:** Includes "User guide", "Rename component", and "Remove component" buttons.
- COMPONENT SETTINGS:** The "Driver:" dropdown is set to "Drivers:TCB:Init" (marked with a red circle and '1').
- DRIVERS:TCB:INIT (TCB) CONFIGURATION ON TCB0:**
 - CONFIGURATION:**
 - ENABLE: Enable: (marked with a red circle and '2')
 - RUNSTDBY: Run Standby:
 - SYNCUPD: Synchronize Update:
 - ASYNC: Asynchronous Enable:
 - DBGUN: Debug Run:
 - CLKSEL: Clock Select: "CLK_PER (No Prescaling)" (marked with a red circle and '3')
 - CNTMODE: Timer Mode: "Input Capture Frequency and Pulse-Width" (marked with a red circle and '4')
 - INTERRUPT CONFIGURATION:**
 - CAPT: Capture or Timeout:
 - EVENT CONFIGURATION:**
 - EDGE: Event Edge:
 - CAPTEI: Event Input Enable: (marked with a red circle and '5')
 - FILTER: Input Capture Noise Cancellation Filter:
- COUNT AND COMPARE/CAPTURE:**
 - CCMP: Compare or Capture: 0
 - CNT: Count: 0

結果: TCB構成設定が完了されます。

4.4. 事象システムドライバ

事象システム(EVSY)は周辺機能から周辺機能への直接的な合図を許します。これはCPUを使うことなく、1つの周辺機能(事象生成部)での変化に事象チャンネルを通して他の周辺機能(事象使用部)での活動を起動することを許します。チャンネル経路は主クロックに対して非同期または同期のどちらかにすることができます。この動作形態は応用の必要条件に基づいて選ばれなければなりません。


ここではTCAからのPWM信号がPA5に接続されて事象入力として使われます。

 **行うこと:** 事象システムドライバを追加してください。

1. Add software component(ソフトウェア構成部品追加)枠をクリックしてください。
2. ADD SOFTWARE COMPONENTS(ソフトウェア構成部品追加)ウィンドウからDrivers(ドライバ)を展開してください。
3. Event System(事象システム)ドライバを検索し、それを選択してその後にAdd Component(s)(構成部品追加)をクリックしてください(「TC Aドライバ」で段階3と4を参照してください)。

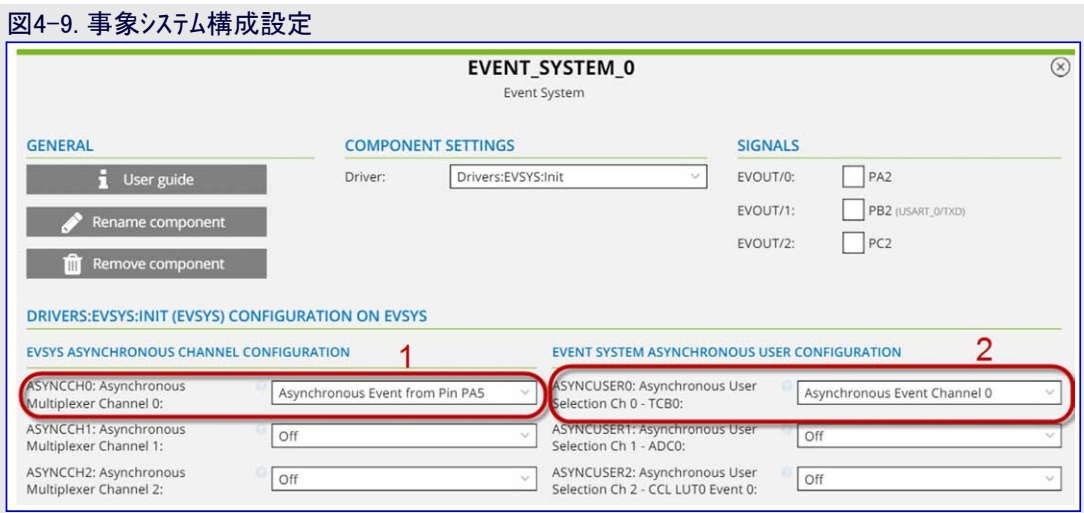
 **情報:** EVENT_SYSTEM_0ドライバがAssignment_ATtiny817プロジェクトに追加されます。今やEVENT_SYSTEM周辺機能構成設定が行われることが必要です。


 **行うこと:** PA5ピンでの事象入力と事象使用部をTCBとして事象システムドライバを構成設定してください。

 **情報:** EVENT_SYSTEM構成設定は図4-9.の1と2で番号付けられた赤色印で示されます。

1. EVENT_SYSTEM_0枠をクリックしてください。
1. “ASYNCCH0:Asynchronous Multiplexer Channel 0(ASYNCCH0:非同期多重器チャンネル0)”を”Asynchronous Event from pin PA5 (PA5ピンからの非同期事象)”として選んでください。
2. “ASYNCUSER0:Asynchronous User Selection Ch0 - TCB0(ASYNCUSER0:非同期使用部選択チャンネル0 - TCB0)”を”Asynchronous Event Channel 0(非同期事象チャンネル0)”として選んでください。

 **情報:** 事象使用部はTCB0で事象入力はASYNCCH0を通して配線され、故にEVENT SYSTEM USER CONFIGURATION (事象システム使用部構成設定)はAsynchronous Event Channel 0(非同期事象チャンネル0)として選ばれます。




 **結果:** 事象システム構成設定が完了されます。

4.5. USARTドライバ

USART周辺機能は与えられたPWM信号の周波数と計算されたデューティサイクルを送るように構成設定されます。


ATtiny817 Xplained Proは組み込みデバugg(EDBG)、仮想COMポートインターフェースを持つ複合USB装置を含みます。仮想COMポートはATtiny817のUARTに接続され、端末ソフトウェアを通して目的対象応用と通信する容易な方法を提供します。これは可変のボーレート、パリティ、停止ビット設定を提供します。ATtiny817の設定が端末ソフトウェアで与えられる設定と合わなければならないことに注意してください。


仮想COMポート接続 : PB2 - UART TXD (ATtiny817 TX線)、PB3 - UART RXD (ATtiny817 RX線)


 行うこと: USARTドライバを追加してください。

1. Add software component(ソフトウェア構成部品追加)枠をクリックしてください。
2. ADD SOFTWARE COMPONENTS(ソフトウェア構成部品追加)ウィンドウからDrivers(ドライバ)を展開してください。
3. USARTドライバを検索し、それを選択してその後にAdd Component(s)(構成部品追加)をクリックしてください(「TCAドライバ」で段階3と4を参照してください)。

注: ここでUSARTドライバを追加してください。

 **情報:** USART_0ドライバがAssignment_ATtiny817プロジェクトに追加されます。今やUSART周辺機能構成設定が行われるが必要です。

 行うこと: USARTドライバをボーレート:9600、PB2:TXD、PB3:RXDに構成設定してください。

 **情報:** 構成設定は図4-10.の1~4で番号付けられた赤色印で示されます。

1. COMPONENT SETTINGS(構成部品設定)下で、Deiver:(ドライバ:)をDriver:USART_Basic(ドライバ:USART基本)とし、Mode:(動作形態:)をAsync Polled Mode(非同期ポーリング動作)として設定してください。
2. SIGNALS(信号)下で、RXD=PB3とTXD=PB2を選んでください。
3. TXEN:Transmitter Enable(TXEN:送信許可)チェック枠を選ぶことによって送信部を許可してください。RXEN:Receiver enable(RXEN:受信許可)がチェックされている場合、それを非チェックにしてください。
4. baud rate(ボーレート)を9600に選んでください。



 **情報:** ここでは既定でボーレートが9600です。


図4-10. USART構成設定

 **結果:** USART構成設定が完了されます。

4.6. プロジェクト生成、コード開発

必要とされる全てのドライバが追加されています。プロジェクトは今や生成されて、コードは以下のように追加されるでしょう。

- RTC ISRでPWMのデューティサイクルを変化
- 与えられたPWM信号の周波数とデューティサイクルを計算
- USARTを通して計算されたデューティサイクルと周波数を送出

 行うこと: プロジェクトを生成してください。

1. (ウインドウ下部の)  枠をクリックしてください。

i 情報: “Project Summary(プロジェクト要約)”ウインドウはこの再構成設定プロジェクトに対して変更されたファイルと追加されたファイルを一覧にします。全ての新規と構成設定されたドライバのファイルがここで一覧にされます。

2. `main.c`ファイルを選んでOKをクリックしてください。

i 情報: `main.c`ファイルの選択は新しい空の`main.c`ファイルを作成します。

3. 構成設定した全てのドライバが生成されたプロジェクトの`system_init()`関数で初期化されることを確認してください。

🖋️ 助言: `src`⇒`driver_init.c`⇒`system_init()`

✅ 結果: 構成設定したドライバを持つプロジェクトが生成されます。

🔧 行うこと: RTC ISR内でTCAによって生成されたPWMのデューティサイクルを変えるコードを追加してください。

- `driver_isr.c`ファイルを開いてください。
- ISRの上でISRで変更される変数を定義し、初期デューティサイクルが10%として構成設定されるよう10に初期化してください。

```
volatile uint8_t change_duty_cycle=10;
```

- ISRでデューティサイクルを10%変えて100%到達後10%に反転するようにISR内に下のコードを追加してください。

```
change_duty_cycle+=10;
if (change_duty_cycle >100)
{
    change_duty_cycle =10;
}
TCA0.SINGLE.CMP0 = change_duty_cycle;
```

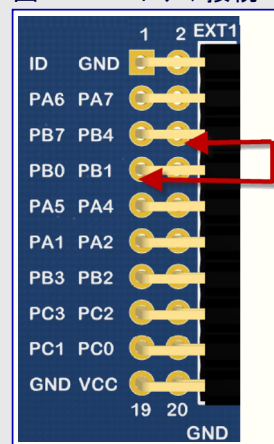
i 情報: `CMP0`レジスタがTCAのPWMのデューティサイクルを決めます。

- プロジェクトを構築するために’F7’を押してください(構築は異常なしで成功裏に終わるべきです)。
- Ctrl+Alt+F5キー押下によって更新したコードをデバイスに書いてください。
- 右図で示されるようにATtiny817 Xplained Pro基板でPB0からPB4へ線を接続してください。

注: EXT1コネクタで列’1’(上)は左側のピンの列に対応し、列’2’(下)は右側のピンの列に対応し、故にPB0は列1(上)でPB4は列2(下)です。

i 情報: TCA PWM出力はPB0で生成され、LEDはATtiny817 Xplained Pro基板のPB4ピンに接続されます。故にPB0とPB4の接続は生成したPWMでLEDの輝度を変えます。

図4-11. ハードウェア接続



✅ 結果: LEDはPWMデューティサイクルが変わる時に変えられる輝度で観察されます。



行うこと: 与えられたPWMのデューティサイクルと周波数を計算するコードを追加してください。

TCBの動作形態は'捕獲入力周波数とパルス幅測定動作'として構成設定されます。

この動作形態では、事象入力信号で正端が検出される時に計時器が計数を開始します。後続する下降端で計数値が捕獲されます。計数器は事象入力信号の2つ目の上昇端が検出される時に停止します。これは割り込み要求フラグも設定します。捕獲読み込みはこの割り込み要求フラグを解除します。捕獲レジスタが読まれるか、または割り込み要求フラグが解除されると、TCは新しい捕獲の流れの準備が整います。

1. **main.c**ファイルを開いてください。
2. 下で示されるように、**main()**の上に計時器レジスタを読んで計算したデューティサイクルと周波数を格納する変数を定義してください。

```
volatile uint16_t period_after_capture = 0;
volatile uint16_t pulse_width_after_capture = 0;

volatile uint8_t capture_duty = 0;
volatile uint16_t capture_frequency = 0;
```

3. デューティサイクルと周波数を計算するためにwhile (1)内に下のコードを追加してください。

```
if (TCB0.INTFLAGS)
{
    TCB0.INTFLAGS=1;

    period_after_capture = TCB0.CNT;
    pulse_width_after_capture = TCB0.CCMP;

    capture_duty = ((pulse_width_after_capture * 100 )/period_after_capture);
    if (capture_duty>100)
    {
        capture_duty=0;
    }
    capture_frequency = F_CPU /period_after_capture;
}
```



情報: 捕獲した値が利用可能な時にTCB0.**INTFLAGS**が設定(1)されます。このフラグはそれへの1書き込みによって解除されます。周期はTCB0.**CNT**を通して読まれ、パルス幅はTCB0.**CCMP**レジスタを通して読まれ、デューティサイクルが%で計算されます。捕獲周波数はHzです。



行うこと: 与えられたPWMを計算したデューティサイクルと周波数をUSARTを通して送るコードを追加してください。

RTC割り込みが500ms毎に生成され、デューティサイクルが500ms毎に変えられるため、コードは毎回の500ms後に計算されたデューティサイクルと周波数を送るように追加されます。

1. **main.c**ファイルを開いてください。
2. 下で示されるように、**main()**の上に500ms毎にフラグを設定する変数を定義し、送信緩衝部文字列を定義してください。

```
volatile uint8_t rtc_500ms_flg=0;
const char tx_buf[]={"¥ncaptured data:"};
```

3. **driver_isr.c**を開いてください。**driver_isr.c**ファイルでISRの上にexternとして**rtc_500ms_flg**を定義してください。

```
extern uint8_t rtc_500ms_flg;
```

4. **driver_isr.c**内のRTC ISRで**rtc_500ms_flg**を設定してください。

```
rtc_500ms_flg=1;
```

5. `main.c`ファイルで、下で示されるように、計算したデューティサイクルと周波数を文字列に変換してUSARTを通してそれを送るように、`main()`の上に`send_data()`関数を定義してください。

```
void send_data()
{
    uint8_t i=0;
    char duty_str[4]={0}, freq_str[10]={0}, tx_data[30]={0};

    itoa(capture_duty, duty_str, 10); // デューティサイクルをASCIIへ
    ltoa(capture_frequency, freq_str, 10); // 周波数をASCIIへ

    strcat(tx_data, tx_buf); // tx_data=tx_buf+デューティサイクル文字列+周波数文字列
    strcat(tx_data, duty_str);
    strcat(tx_data, "% ");
    strcat(tx_data, freq_str);
    strcat(tx_data, "Hz");

    while(tx_data[i] !=0) // スル文字調査
    {
        USART_0_write(tx_data[i]); // データ送出
        i++;
    }
}
```

6. ファイルの先頭で`string.h`ファイルをインクルードしてください。

```
#include <string.h>
```



情報: `string.h`で定義された`itoa`と`ltoa`の関数はデューティサイクルと周波数をASCIIに変換します。`strcat`関数はデューティサイクル文字列と周波数文字列を`tx_buf`に連結し、`tx_buf`文字列での結果の文字列が端末に送られます。`USART_0_putc`関数はUSARTのTXDATAレジスタにデータを書きます。TXDATAはレジスタが空で新しいデータの準備が整っていることを示すデータレジスタ空フラグ(USART.STATUS内のDREIF)が設定(1)される時にだけ書くことができます。

7. 毎回の500ms後に`send_data()`関数を呼ぶようにwhile (1)内にコードを追加してください。

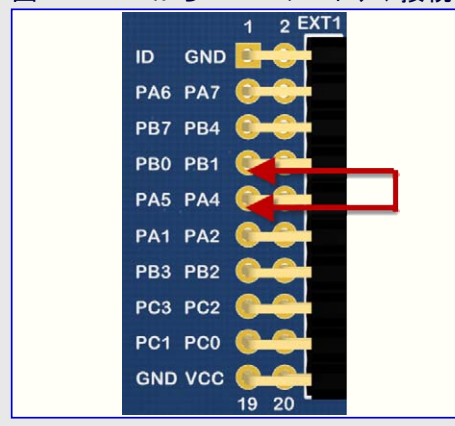
```
if (rtc_500ms_flg==1)
{
    rtc_500ms_flg=0;
    send_data();
}
```

8. プロジェクトを構築するために'F7'を押してください(構築は異常なしで成功裏に終わるべきです)。
 9. Debug(デバッグ)⇒Start without Debugging(デバッグなしで開始)(代替はCtrl+Alt+F5)を選ぶことによって更新したコードでデバイスを書いてください。
 10. 右図で示されるようにATtiny817 Xplained Pro基板でPB0からPA5へ線を接続してください。

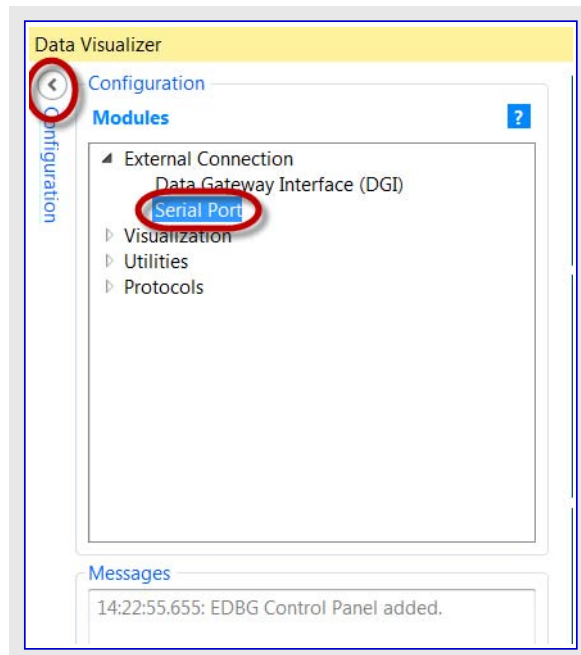


情報: TCBはPA5でPWM信号を捕獲し、TCAのPWMはPB0で生成されるように構成設定されています。

図4-12. PB0からPB5へのハードウェア接続




11. Atmel Studio 7でメニューからTools(ツール)⇒Data Visualizer(データ可視器)を選んでください。
12. Data Visualizer(データ可視器)ウィンドウでConfiguration(構成設定)タブをクリックし、その後にSerial Port(シリアルポート)をダブルクリックしてください。



13. “Serial Port Control Panel(シリアルポート制御盤)”で一覧にされたEDBG Virtual COM Port(EDBG仮想COMポート)番号を選び、Baud rate(ボーレート)を9600に設定してください。Open Terminal(端末を開く)鈕、その後にConnect(接続)をクリックしてください。



助言: 接続したATtiny817 Xplained Pro基板のEDBG仮想COMポート番号はSTART(開始)⇒Control Panel(コントロールパネル)⇒Device Manager(デバイスマネージャ)⇒Ports(ポート)でも一覧にされます。

-  **結果:** 与えられたPWM信号を捕獲したデューティサイクルと周波数が端末ウィンドウで表示されます。

```

captured data:49% 33333Hz
captured data:59% 33333Hz
captured data:69% 33333Hz
captured data:79% 33333Hz
captured data:89% 33333Hz
captured data:99% 33333Hz
captured data:9% 33333Hz
captured data:19% 33333Hz
captured data:29% 33333Hz
captured data:39% 33333Hz
captured data:49% 33333Hz
captured data:59% 33333Hz
captured data:69% 33333Hz
captured data:79% 33333Hz
captured data:89% 33333Hz
captured data:99% 33333Hz
captured data:9% 33333Hz
    
```

最終的なコードは下で示されるように見えるべきです。

```

#include <string.h>
#include <atmel_start.h>

volatile uint16_t period_after_capture = 0;
volatile uint16_t pulse_width_after_capture = 0;
volatile uint8_t capture_duty = 0;
volatile uint16_t capture_frequency = 0;

volatile uint8_t rtc_500ms_flg=0;
const char tx_buf[]={"%ncaptured data:"};

void send_data()
{
    
```

```

uint8_t i=0;
char duty_str[4]={0}, freq_str[10]={0}, tx_data[30]={0};
itoa(capture_duty, duty_str, 10);           // デューティサイクルをASCIIへ
ltoa(capture_frequency, freq_str, 10);      // 周波数をASCIIへ
strcat(tx_data, tx_buf);                    // tx_data=tx_buf+デューティサイクル文字列+周波数文字列
strcat(tx_data, duty_str);
strcat(tx_data, "% ");
strcat(tx_data, freq_str);
strcat(tx_data, "Hz");
while(tx_data[i] !=0)                       // スル文字調査
{
    USART_0_write(tx_data[i]);             // データ送付
    i++;
}

int main(void)
{
    /* MCU、ドライバ、ミドルウェアを初期化 */
    atmel_start_init();

    /* あなたの応用コードで置き換えてください。 */
    while (1) {
        if(TCBO. INTFLAGS)
        {
            TCBO. INTFLAGS=1;
            period_after_capture = TCBO. CNT;
            pulse_width_after_capture = TCBO. CCMP;
            capture_duty = ((pulse_width_after_capture * 100 )/period_after_capture);
            if (capture_duty>100)
            {
                capture_duty=0;
            }
            capture_frequency = F_CPU /period_after_capture;
        }
        if (rtc_500ms_flg==1)
        {
            rtc_500ms_flg=0;
            send_data();
        }
    }
}

```

5. 課題3：2値周波数移動符号化の基礎

構成設定可能な注文論理回路(CCL:Configurable Custom Logic)単位部はデバイス上のピン、事象、または周辺機能に接続することができる設定可能な論理回路周辺機能です。これは簡単な接続用論理回路機能に対して外部論理回路ゲートを省くことを使用者に許します。

この課題では、CCLに基づく小さな応用が開発されます。これは釦が押されているか否かに応じて2つの異なる周波数でLEDを点滅します。異なる周波数の2つのパルス列が生成されてCCLに配線されます。CCLは下図で示されるように第3の入力信号の状態に基づいてどのパルス列を出力に渡すかを選ぶように構成設定されます。

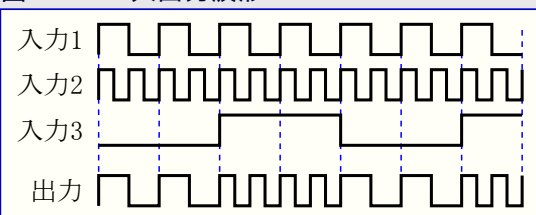
ここで入力3がLowの時に出力=入力1、入力3がHighの時に出力=入力2です。

応用は2つの周波数間を離散移動するように2値データを符号化する2値周波数移動符号化の仕組みの基礎にすることができます。

故に、この応用では次のとおりです。

- “入力1”は課題2でTCAを使って生成したPWM信号です。
- “入力2”はPITからの事象出力です。
- “入力3”は釦(SWO)の状態です。
- “出力”はPA7でのCCL出力です。

図5-1. CCL入出力波形



LEDがPA7に接続され、釦押下と釦開放で違う周波数で点滅します。

RTC周辺機能は実時間計数器(RTC:Real-Time Counter)と周期的割り込み計時器(PIT:Periodic Interrupt Timer)の2つのタイミング機能を提供します。

RTC機能と同じクロック元を使うことにより、PITは第nクロック周期毎に割り込みまたは出力事象の起動を要求することができます。nは4,8,16,~32768から選ぶことができます。ここでの事象システムはPITからの事象を出力するように構成設定されます。PITからの事象信号はRTCクロック周期の各々の数に対応する周期を持つクロック信号の形式を持ちます。この応用は8192 RTC周期に対応する事象信号を使います。PITからの事象信号は32kHz/8192=3.9Hz(概ね250ms周期)の周波数を持ちます。

Assignment_ATtiny817プロジェクトはCCLドライバを追加して事象システムとRTCドライバを編集するように再構成設定されます。

この応用について使われる周辺機能は次のとおりです。

- TCA (PB0でのWO0波形出力)
- PIT (事象出力)
- CCL (PA7出力)
- 事象システム
- PB5 (SW0)

クロック詳細は次のとおりです。

- 3.33MHz主クロック
- PIT 32kHz

5.1. CCLドライバ

構成設定可能な注文論理回路(CCL)はデバイスのピン、事象、または他の内部周辺機能に接続することができる設定可能な論理回路周辺機能です。CCLはデバイス周辺機能と外部装置間の”接続論理回路”として扱うことができます。

CCL周辺機能は1つの参照表(LUT:Lookup Table)の対を持ちます。各LUTは3つの入力、真理値表、濾波器/端検出器から成ります。各LUTは3つの入力を持つユーザー設定可能な論理式として出力を生成することができます。この出力は各種組み合わせで入力から生成することができます。


この応用ではLUT1が次のように使われます。

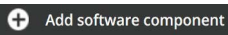
- “入力1”はTCAを使って生成されたPWMです。:(TCAのWO0)
- “入力2”はPITからの事象出力です。:事象供給元1
- “入力3”は釦(SW0)押下です。:事象供給元0
- “出力”はPA7でのCCL出力です。

必要とされる組み合わせ用の真理値表は下表で示されます。IN2が(SW0)釦で、IN2=0の時にOUT=IN1で、IN2=1の時にOUT=IN0です。故にOUT=0xACです。

表5-1. LUTの真理値表

IN2	1	1	1	1	0	0	0	0
IN1	1	1	0	0	1	1	0	0
IN0	1	0	1	0	1	0	1	0
OUT	(MSB) 1	0	1	0	1	1	0	0 (LSB)

 **行うこと:** CCLドライバを追加してください。

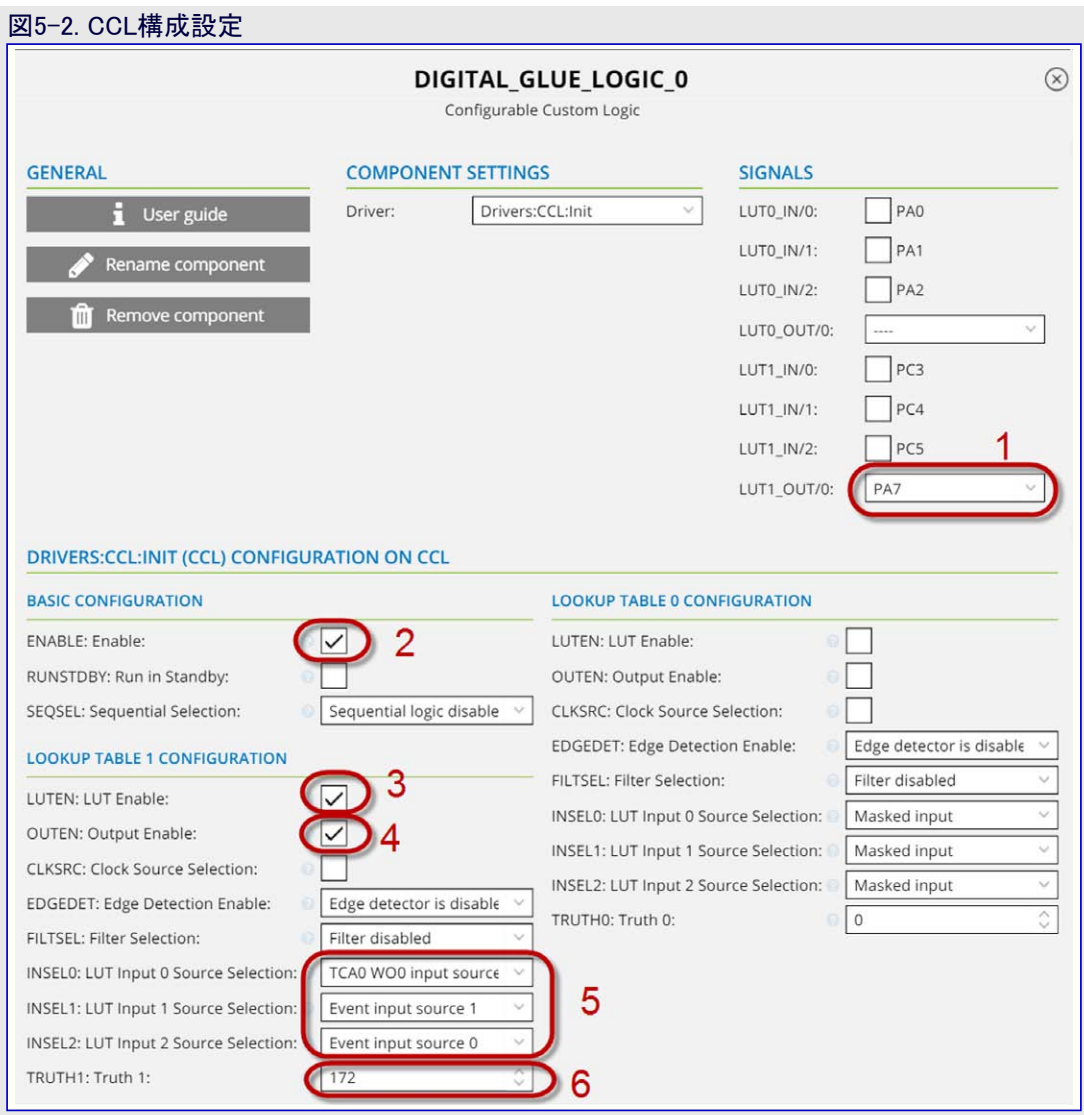
1. Assignment_ATtiny817プロジェクトを開いてください。
2. Solution Explorer(解決策エクスプローラ)ウィンドウでプロジェクト名のAssignment_ATtiny817を、その後に右クリック⇒Re-Configure Atmel Start Project(Atmel STARTプロジェクト再構成設定)をクリックしてください。
3. “Atmel | START”ウィンドウで  枠をクリックし、その後にADD SOFTWARE COMPONENTS(ソフトウェア構成部品追加)からDrivers(ドライバ)を展開してください。
4. Digital Glue Logic(CCL)(デジタル接続論理回路(CCL))ドライバを検索し、それを選択してその後にAdd Component(s)(構成部品追加)をクリックしてください(「TCAドライバ」で段階3と4を参照してください)。


 **情報:** DIGITAL_GLUE_LOGIC_0ドライバがAssignment_ATtiny817プロジェクトに追加されます。今やCCL周辺機能構成設定が下のように行われる必要があります。

 **行うこと:** IN2=事象元0、IN1=事象元1、IN0=WO0 TCA、PA7での出力で使うようにCCLドライバを構成設定してください。

 **情報:** 構成設定は図5-2.で1~6の番号付けられた赤色印で示されます。

1. LUT1_OUT/0でPA7を選んでください。
2. ENABLE:Enable(ENABLE:許可)のチェック印を選ぶことによってこの周辺機能を許可してください。
3. LOOKUP TABLE 1 CONFIGURATION(参照表1構成設定)下のLUTEN:LUT Enable(LUTEN:LUT許可)チェック枠を選んでください。
4. LOOKUP TABLE 1 CONFIGURATION(参照表1構成設定)下のOUTEN:Output Enable(OUTEN:出力許可)チェック枠を選んでください。
5. 次のように入力を選んでください。
 - INSEL0:LUT input 0 source selection:(INSEL0:LUT入力0供給元選択:)=TCA W00 input source(TCA W00入力供給元)
 - INSEL1:LUT input 1 source selection:(INSEL1:LUT入力1供給元選択:)=Event input source 1(事象入力1供給元)
 - INSEL2:LUT input 2 source selection:(INSEL2:LUT入力2供給元選択:)=Event input source 0(事象入力0供給元)
6. 真理値表値TRUTH1:Truth 1:(TRUTH1:真理値表1:)を172(または0xAC)として入力してください(表5-1.でのOUT行を調べてください)。



 **結果:** CCL構成設定が完了されます。

5.2. 事象システムドライバ


事象システム(EVSYS)は周辺機能から周辺機能への直接的な合図を許します。これはCPUを使うことなく、1つの周辺機能(事象生成部)での変化に事象チャンネルを通して他の周辺機能(事象使用部)での活動を起動することを許します。

ここでの事象システムはSW0(PB5)押し錠とPIT(周期的割り込み計時器)から生成されます。


表5-2. 事象の使用部と生成部

事象使用部	事象生成部
CCL LUT1事象0	PB5からの事象
CCL LUT1事象1	PITからの事象:8192 RTCクロック周期間隔

 **行うこと:** EVENT_SYSTEM_0枠をクリックしてEVENT_SYSTEM(事象システム)を表5-2のように構成設定してください。

 **情報:** 構成設定は図5-3.の1~4で番号付けられた赤色印で示されます。


1. “ASYNCCH1:Asynchronous Multiplexer Channel 1(ASYNCCH1:非同期多重器チャネル1)”を”Asynchronous Event from pin PB5 (PB5ピンからの非同期事象)”として選んでください。
2. “ASYNCCH3:Asynchronous Multiplexer Channel 3(ASYNCCH3:非同期多重器チャネル3)”を”Periodic Interrupt CLK_RTC div 8192(CLK_RTCの8192分周の周期的割り込み)”として選んでください。
3. “ASYNCUSER3:Asynchronous User Selection Ch3 - CCL LUT1 Event0(ASYNCUSER3:非同期使用部選択チャネル3 - CCL LUT1事象0)”を”Asynchronous Event Channel 1(非同期事象チャネル1)”として選んでください。

 **情報:** 事象使用部はCCL LUT1 Event0(CCL LUT1事象0)で、故にASYNCUSER3使用部はASYNCCH1を通ります。

4. “ASYNCUSER5:Asynchronous User Selection Ch5 - CCL LUT1 Event1(ASYNCUSER5:非同期使用部選択チャネル5 - CCL LUT1事象1)”を”Asynchronous Event Channel 3(非同期事象チャネル3)”として選んでください。


 **情報:** 事象使用部はCCL LUT1 Event1(CCL LUT1事象1)で、故にASYNCUSER5使用部はASYNCCH3を通ります。

図5-3. 事象システム構成設定


 **結果:** 事象システム構成設定が完了されます。

5.3. PITドライバ

RTC周辺機能は実時間計数器(RTC:Real-Time Counter)と周期的割り込み計時器(PIT:Periodic Interrupt Timer)の2つのタイミング起動を提供します。PIT機能はRTC機能から独立して許可することができます。


 **行うこと:** PITを許可するようにRTCドライバを編集してください。

1. RTC_0をクリックしてください。
2. 下スクロールしてPITを許可するようにRTCを構成設定してください。PITENの次の枠をチェックしてください。


 **結果:** PITが許可されます。

5.4. プロジェクト生成、コード走行


必要とされる全てのドライバが追加され、プロジェクトを今や生成することができます。

 **行うこと:** プロジェクトを生成してください。


1. (ウィンドウ下部の)  枠をクリックしてください。


 **情報:** “Project Summary(プロジェクト要約)”ウィンドウはこの再構成設定プロジェクトに対して変更されたファイルと追加されたファイルを一覧にします。全ての新規と構成設定されたドライバのファイルがここで一覧にされます。

2. Project Summary(プロジェクト要約)ウィンドウでOKをクリックしてください。

 **情報:** main.cとdriver_isr.cのファイルが選択された場合、初期のどのコードも上書きされ、2つのファイルは空のファイルとして作成されます。

3. 構成設定した全てのドライバが生成されたプロジェクト、'src'フォルダの'driver_init.c'ファイル内のsystem_init()関数で初期化されることを確認してください。

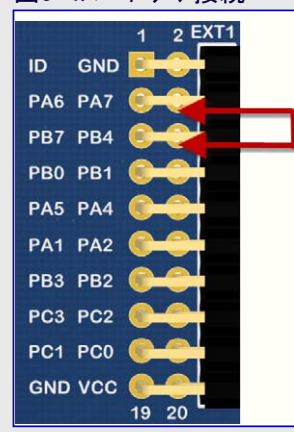
 **結果:** 構成設定したドライバを持つプロジェクトが生成されます。


 **行うこと:** コードを走らせてください。

1. プロジェクトを構築するために'F7'を押してください(構築は異常なしで成功裏に終わるべきです)。
2. Debug(デバッグ)⇒Start without Debugging(デバッグなしで開始)を選ぶことによって更新したコードをデバイスに書いてください。
3. 右図で示されるように、ATtiny817 Xplained Pro基板でPA7からPB4へ線を接続してください。


注: EXT1コネクタで列'1'(上)は左側のピンの列に対応し、列'2'(下)は右側のピンの列に対応し、故にPA7とPB4は列2(下)です。


図5-4. ハードウェア接続



 **情報:** CCL出力はPA7ピンで生成され、LEDはATtiny817 Xplained Pro基板上のPB4ピンに接続されます。PB4とPA7を接続することによって違う周波数でのLED点滅を観察することができます。

4. SW0押し釦を押してLEDを観察し、押し釦を開放してLEDを観察してください。

 **情報:** 押し釦が開放されると、LEDはデューティサイクルが500ms毎の後で変わってLEDの輝度が変わるTCAのPWMで点滅します。押し釦が押されると、LEDはPITでのRTC_CLK/8192期間間隔で点滅します。

 **情報:** この応用はRTCのOVFが禁止された場合にコアから独立して作ることができます。LED点滅を見るにはLED点滅が可視であるようなPWM信号(例えば、50%デューティサイクルの12Hz信号)を生成するようにTCAドライバを編集してください。

6. 結び

実演されるこの訓練は次のとおりです。

- Atmel STARTを通すドライバ構成設定
- コード開発で自動的に生成されたドライバ関数の使用
- ドライバの編集と追加のためのプロジェクトの再構成設定

以下も学びます。

- tinyAVR 1系の各種周辺機能
- 事象を生成するための事象システムの使い方
- 出力を生成するためのCCLの使い方

Atmel Studioとで応用の実時間デバッグを走らせてI/O表示部を見ることが容易で、これはレジスタ表示能力を提供し、実時間でのマイクロコントローラのレジスタ変更を許します。以下のような様々なデバッグ方法を使って応用をデバッグすることが可能です。

- 中断点(ブレークポイント)
- 単一実行
- I/O表示部

7. Atmel | STARTからのソースコード取得

コード例は画像使用者インターフェース(GUI)を通して応用コードの構成設定を許すウェブに基づくAtmel | STARTを通して利用可能です。コードは下の直接コード例リンクまたはAtmel | START先頭頁のBROWSE EXAMPLES(例検索)鉤經由Atmel Studio 7.0とIAR Embedded Workbench®の両方に対してダウンロードすることができます。

Atmel | STARTウェブ ページ : <http://microchip.com/start>

コード例

- tinyAVR 1系での開始に際して (Getting Started with the tinyAVR 1-series:)
 - http://start.atmel.com/#example/Atmel:getting_started_with_the_tinyavr_1_series:1.0.0::Application:Getting_Started_with_the_tinyAVR_1-series:

例プロジェクトについての詳細と情報に関してはAtmel | STARTでUser guide(使用者の手引き)をクリックしてください。User guide鉤はAtmel | STARTプロジェクト構成設定部内の一覧画面でプロジェクト名をクリックすることにより、例閲覧部で見つけることができます。

Atmel Studio

DOWNLOAD SELECTED EXAMPLE(選んだ例をダウンロード)をクリックすることにより、Atmel | STARTで例閲覧部からAtmel Studio用.atzipファイルとしてコードをダウンロードしてください。Atmel | START内からファイルをダウンロードするには、EXPORT PROJECT(プロジェクトをエクスポート)に続いてDOWNLOAD PACK(一括ダウンロード)をクリックしてください。

ダウンロードした.atzipファイルをダブルクリックしてください。プロジェクトがAtmel Studio 7.0に導入されます。

IAR Embedded Workbench

IAR Embedded Workbenchでプロジェクトをインポートする方法の情報についてはAtmel | START使用者の手引きを開き、Using Atmel Start Output in External Tools(外部ツールでAtmel START出力を使用)とIAR Embedded Workbenchを選んでください。Atmel | START使用者の手引きへのリンクは共に頁の右上隅に置かれたAtmel | START先頭頁からHelp(手助け)またはプロジェクト構成設定部内のHelp And Support(手助けと支援)をクリックすることによって見つけることができます。

8. 改訂履歴

資料改訂	日付	注釈
A	2017年8月	初版資料公開
B	2018年2月	「関連デバイス」章と例の始めにお読みくださいに「STARTからのコード取得」項を追加

Microchipウェブ サイト

Microchipは<http://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手段として使われます。お気に入りのインターネット ブラウザを用いてアクセスすることができ、ウェブ サイトは以下の情報を含みます。

- **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
- **一般的な技術支援** – 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相談役プログラム員一覧
- **Microchip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

お客様への変更通知サービス

Microchipのお客様通知サービスは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<http://www.microchip.com/>でMicrochipのウェブ サイトをアクセスしてください。”Support”下で”Customer Change Notification”をクリックし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
- 現場応用技術者(FAE:Field Application 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、または現場応用技術者(FAE)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<http://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、その製品系統が今日の市場でその種類の最も安全な系統の1つであると考えます。
- コード保護機能を破るのに使われる不正でおそらく違法な方法があります。当社の知る限りこれらの方法の全ては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です。おそらく、それを行う人は知的財産の窃盗に関与しています。
- Microchipはそれらのコードの完全性について心配されているお客様と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らのコードの安全を保証することはできません。コード保護は当社が製品を”破ることができない”として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

デバイス応用などに関してこの刊行物に含まれ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。Microchipはその条件、品質、性能、商品性、目的適合性を含め、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ありません。Microchipはこの情報とそれの使用から生じる全責任を否認します。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

Microchipの名前とロゴ、Mchip、AnyRate、AVR、AVR Freaks、BitCloud、chipKIT、chipKIT、CryptoMemory、CryptoRF、dsPIC、FlashFlex、flexPWR、Heldo、JukeBlox、KeeLoq、KeeLoq、Kleer、LANCheck、LINK MD、maXStylus、maXTouch、MediaLB、megaAVR、MOST、MOST、MPLAB、OptoLyzer、PIC、picoPower、PICSTART、PIC32、Prochip Designer、QTouch、SAM-BA、SpyNIC、SST、SST、SuperFlash、tinyAVR、UNI/O、XMEGAは米国と他の国に於けるMicrochip Technology Incorporatedの登録商標です。

ClockWorks、The Embedded Control Solutions Company、EtherSynch、Hyper Speed Control、HyperLight Load、IntelliMOS、mTouch、Precision Edge、Quiet-Wireは米国に於けるMicrochip Technology Incorporatedの登録商標です。

Adjacent Key Suppression、AKS、Analog-for-the-Digital Age、Any Capacitor、AnyIn、AnyOut、BodyCom、CodeGuard、CryptoAuthentication、CryptoCompanion、CryptoController、dsPICDEM、dsPICDEM.net、Dynamic Average Matching、DAM、ECAN、EtherGREEN、In-Circuit Serial Programming、ICSP、INICnet、Inter-Chip Connectivity、JitterBlocker、KleerNet、KleerNet、memBrain、Mindi、MiWi、motorBench、MPASM、MPF、MPLAB Certified、MPLAB、MPLINK、MultiTRAK、NetDetach、Omniscient Code Generation、PICDEM、PICDEM.net、PICkit、PICtail、PowerSmart、PureSilicon、QMatrix、REAL ICE、Ripple Blocker、SAM-ICE、Serial Quad I/O、SMART-I.S.、SQI、SuperSwitcher、SuperSwitcher II、Total Endurance、TSHARC、USBCheck、VariSense、View Sense、WiperLock、Wireless DNA、ZENAは米国と他の国に於けるMicrochip Technology Incorporatedの商標です。

SQTPは米国に於けるMicrochip Technology Incorporatedの役務標章です。

Silicon Storage Technologyは他の国に於けるMicrochip Technology Inc.の登録商標です。

GestICは他の国に於けるMicrochip Technology Inc.の子会社であるMicrochip Technology Germany II GmbH & Co. KGの登録商標です。

ここで言及した以外の全ての商標はそれら各々の会社の所有物です。

© 2018年、Microchip Technology Incorporated、米国印刷、不許複製

DNVによって認証された品質管理システム

ISO/TS 16949

Microchipはその世界的な本社、アリゾナ州のチャンドラーとテンペ、オレゴン州グラシャムの設計とウェハー製造設備とカリフォルニアとインドの設計センターに対してISO/TS-16949:2009認証を取得しました。当社の品質システムの処理と手続きはPIC[®] MCUとdsPIC[®] DSC、KEELOQ符号飛び回りデバイス、直列EEPROM、マイクロ周辺機能、不揮発性メモリ、アナログ製品用です。加えて、開発システムの設計と製造のためのMicrochipの品質システムはISO 9001:2000認証取得です。

日本語© HERO 2020.

本訓練の手引きはMicrochipのtinyAVR[®] 1系での開始に際して訓練の手引き(DS40001949B-2018年2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識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。



MICROCHIP

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: http://www.microchip.com/support ウェブアドレス: 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 ダラス Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 デトロイト Novi, MI Tel: 248-848-4000 ヒューストン TX Tel: 281-894-5983 インディアナポリス Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 ロサンゼルス 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 カナダ - トロント Tel: 905-695-1980 Fax: 905-695-2078	オーストラリア - シドニー 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 - 香港特别行政区 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 - 廈門 Tel: 86-592-2388138 中国 - 珠海 Tel: 86-756-3210040	インド - ハンガロール Tel: 91-80-3090-4444 インド - ニューデリー Tel: 91-11-4160-8631 インド - フネー 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 - クアラルンプール Tel: 60-3-7651-7906 マレーシア - ペナン Tel: 60-4-227-8870 フィリピン - マニラ Tel: 63-2-634-9065 シンガポール 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 - バンコク Tel: 66-2-694-1351 ベトナム - ホーチミン Tel: 84-28-5448-2100	オーストラリア - ウェルズ 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 フィンランド - 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 ドイツ - ガルピング Tel: 49-8931-9700 ドイツ - ハーン Tel: 49-2129-3766400 ドイツ - ハイムブロン Tel: 49-7131-67-3636 ドイツ - カールスルーエ Tel: 49-721-625370 ドイツ - ミュンヘン Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 ドイツ - 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 - マドリッド Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 スウェーデン - イェテボリ 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 - ウォーキングム Tel: 44-118-921-5800 Fax: 44-118-921-5820