お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

RENESAS

アプリケーション・ノート

78K0/Kx2

サンプル・プログラム

8ビットPWM出力編

この資料は，サンプル・プログラムの「8ビットPWM出力」の動作概要と，マイコンの基本的なPWM出力の設定を説明したものです。サンプル・プログラムでは，PWMの設定を行ったあとに，その出力制御を行います。

対象デバイス

78K0/KB2マイクロコントローラ
78K0/KC2マイクロコントローラ
78K0/KD2マイクロコントローラ
78K0/KE2マイクロコントローラ
78K0/KF2マイクロコントローラ

目　次

資料番号　U19032JJ1V0AN00（第1版）
発行年月　May 2009 NS

- **本資料に記載されている内容は2009年5月現在のもので，今後，予告なく変更することがあります。**
  量産設計の際には最新の個別データ・シート等をご参照ください。
- 文書による当社の事前の承諾なしに本資料の転載複製を禁じます。当社は，本資料の誤りに関し，一切その責を負いません。
- 当社は，本資料に記載された当社製品の使用に関連し発生した第三者の特許権，著作権その他の知的財産権の侵害等に関し，一切その責を負いません。当社は，本資料に基づき当社または第三者の特許権，著作権その他の知的財産権を何ら許諾するものではありません。
- 本資料に記載された回路，ソフトウエアおよびこれらに関連する情報は，半導体製品の動作例，応用例を説明するものです。お客様の機器の設計において，回路，ソフトウエアおよびこれらに関連する情報を使用する場合には，お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し，当社は，一切その責を負いません。
- 当社は，当社製品の品質，信頼性の向上に努めておりますが，当社製品の不具合が完全に発生しないことを保証するものではありません。また，当社製品は耐放射線設計については行っておりません。当社製品をお客様の機器にご使用の際には，当社製品の不具合の結果として，生命，身体および財産に対する損害や社会的損害を生じさせないよう，お客様の責任において冗長設計，延焼対策設計，誤動作防止設計等の安全設計を行ってください。
- 当社は，当社製品の品質水準を「標準水準」，「特別水準」およびお客様に品質保証プログラムを指定していただく「特定水準」に分類しております。また，各品質水準は，以下に示す用途に製品が使われることを意図しておりますので，当社製品の品質水準をご確認ください。

  標準水準：コンピュータ，OA機器，通信機器，計測機器，AV機器，家電，工作機械，パーソナル機器，産業用ロボット

  特別水準：輸送機器（自動車，電車，船舶等），交通用信号機器，防災・防犯装置，各種安全装置，生命維持を目的として設計されていない医療機器

  特定水準：航空機器，航空宇宙機器，海底中継機器，原子力制御システム，生命維持のための医療機器，生命維持のための装置またはシステム等

  当社製品のデータ・シート，データ・ブック等の資料で特に品質水準の表示がない場合は，標準水準製品であることを表します。意図されていない用途で当社製品の使用をお客様が希望する場合には，事前に当社販売窓口までお問い合わせください。

（注）

（1）本事項において使用されている「当社」とは，NECエレクトロニクス株式会社およびNECエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいう。

（2）本事項において使用されている「当社製品」とは，（1）において定義された当社の開発，製造製品をいう。

M8E0710J

第1章　概　　要

このサンプル・プログラムでは，PWM出力の設定を行います。ここでは，PWM出力機能を搭載している8ビット・タイマ／イベント・カウンタ50（TM50），8ビット・タイマH0（TMH0）の2つのタイマの設定例を紹介しています。

本サンプル・プログラムでは，およそ2 [ms] 周期のPWM出力のデューティを変更することにより，タイマ出力端子（TO50，TOH0）に接続したLEDの輝度を制御します。デューティの変更は2 [ms] のPWM周期ごとの割り込みをプログラムでカウントして得られる500 [ms] 周期（割り込み250回ごと）で行います。

なお，クロック関連の設定や周辺ハードウエアの初期設定ごとのサンプル・プログラムの初期設定編と同内容の部分は，別ファイルとしてサブルーチン・コールにて使用しています。

（1）PWM設定内容

- タイマ関連の設定
- デューティの設定
- 出力ポートの設定

（2）メイン処理動作の内容

- PWM出力用タイマを起動後，割り込み回数をカウントするカウンタを初期化して，NOP命令のみの割り込み待ちループに入ります。

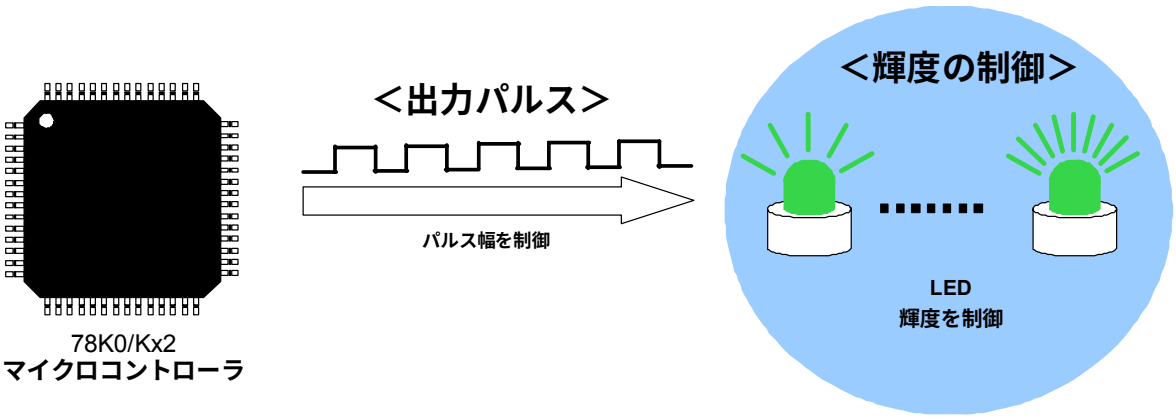
（3）割り込み処理の内容

- 2 [ms] 周期の割り込みごとに割り込み回数をカウントします。割り込み250回ごと（500 [ms] ）に増加ステップ（20 [%] ）ずつデューティを上げていき，LEDの輝度を制御します（デューティ初期値10 [%] ）。デューティが上限値（90 [%] ）まで到達したら初期値（10 [%] ）に戻します。

なお，アクティブ・レベルは，PWM出力がハイ・アクティブ，LEDがロウ・アクティブであるのでLEDの輝度は“100－デューティ比”となり，本サンプル・プログラムでは以下のようになります。

デューティ[%]	10	20	30	40	50	60	70	80	90
LEDの輝度[%]	90	80	70	60	50	40	30	20	10

※割り込みベクタは001CH（INTTMH0：TMH0版），001EH（INTTM50：TM50版）に設定します。
（ベクタ・テーブル001CH，001EH番地の割り込みハンドラ名称を“IINIT”から“IINTTMH0”，“IINTTM50”に変更します。）



(4) PWMの種類

・PWMは，8ビット・タイマのPWM出力機能を使用して実現します。使用できるのは，8ビット・タイマ／イベント・カウンタ50, 51，8ビット・タイマH0, H1の4つです。その中で本サンプル・プログラムでは，8ビット・タイマ／イベント・カウンタ50と8ビットタイマH0の設定例を紹介しています。

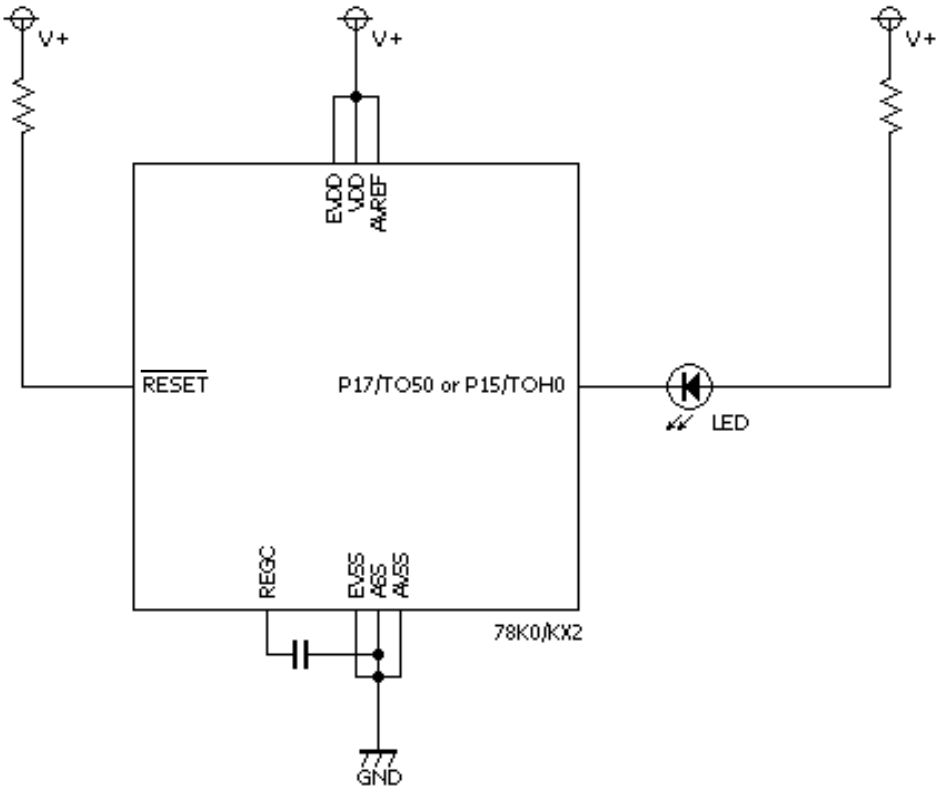
注意　デバイス使用上の注意事項については，78K0/Kx2　ユーザーズ・マニュアルを参照してください。

第2章　回路イメージ

この章では，このサンプル･プログラムで使用する場合の回路イメージおよび周辺ハードウエアを説明します。

2. 1　回路イメージ

回路イメージを次に示します。



注意1. AV$_{REF}$**端子は**V$_{DD}$**に直接接続してください。**

2. AV$_{SS}$**端子は**GND**に直接接続してください。**
3. REGC**端子はコンデンサ（**0.47～1 μF**）を介し，**V$_{SS}$**に接続してください。**
4. EV$_{DD}$**端子は**V$_{DD}$**に直接接続してください（**78K0/KE2, 78K0/KF2**のみ）。**
5. EV$_{SS}$**端子は**GND**に直接接続してください（**78K0/KE2, 78K0/KF2**のみ）。**
6. **使用電圧と動作周波数などの詳細については，ユーザーズ・マニュアルを参照してください。**
7. **ポートの出力電流値には上限値があるので，そのスペックの範囲内で駆動する**LED**を使用してください。**

2. 2　マイコン以外の使用デバイス

マイコン以外に使用するデバイスを次に示します。

（1**）**LED

PWM**出力のデューティ変更により，**LED**の輝度を制御します。**

第3章　ソフトウエアについて

この章では，ダウンロードする圧縮ファイルのファイル構成，使用するマイコンの内蔵周辺機能，サンプル・プログラムのPWM設定と動作概要，およびフロー・チャートを説明します。

3. 1　ファイル構成

ダウンロードする圧縮ファイルのファイル構成は，次のようになっています。

【C言語版】

ファイル名[注]	説 明	同封圧縮（*.zip）ファイル		
				
Kx2_Pwm.c	PWMの設定，メイン処理，割り込み処理のソース・ファイル	●	●	
Kx2_func.c	初期化処理を外部関数化したソース・ファイル	●	●	
Kx2_op.asm	オプション・バイト設定用アセンブラ・ソース・ファイル	●	●	
Kx2_Pwm.prw	統合開発環境 PM+用ワーク･スペース・ファイル		●	
Kx2_Pwm.prj	統合開発環境 PM+用プロジェクト・ファイル		●	
Kx2_Pwm.pri Kx2_Pwm.prs Kx2_Pwm.prm	システム・シミュレータ SM+ for 78K0/Kx2用プロジェクト・ファイル			●

注　各ファイル名の”x”部分は，それぞれのデバイスの名前になります。

ex) 78K0/KB2の場合 → ”KB2_Pwm.c”

備考　  ：ソース・ファイルのみ同封

 ：統合開発環境 PM+とシステム・シミュレータ SM+ for 78K0/Kx2で使用するファイルを同封

 ：システム・シミュレータ SM+ for 78K0/Kx2で使用するマイコン動作シミュレーション・ファイルを同封

【アセンブリ言語版】

ファイル名[注]	説 明	同封圧縮（*.zip）ファイル		
				
Kx2_Pwm.asm	PWMの設定，メイン処理，割り込み処理のソース・ファイル	●	●	
Kx2_subr.asm	初期化処理をサブルーチン化したソース・ファイル	●	●	
Kx2_op.asm	オプション・バイト設定用アセンブラ・ソース・ファイル	●	●	
Kx2_Pwm.prw	統合開発環境 PM+用ワーク･スペース・ファイル		●	
Kx2_Pwm.prj	統合開発環境 PM+用プロジェクト・ファイル		●	
Kx2_Pwm.pri Kx2_Pwm.prs Kx2_Pwm.prm	システム・シミュレータ SM+ for 78K0/Kx2用プロジェクト・ファイル			●

注　各ファイル名の”x”部分は，それぞれのデバイスの名前になります。

ex) 78K0/KB2の場合 → ”KB2_Pwm.asm”

備考  ：ソース・ファイルのみ同封

 ：統合開発環境 PM+とシステム・シミュレータ SM+ for 78K0/Kx2で使用するファイルを同封

 ：システム・シミュレータ SM+ for 78K0/Kx2で使用するマイコン動作シミュレーション・ファイルを同封

3. 2　使用するマイコン内蔵周辺機能

このサンプル・プログラムでは，マイコンに内蔵する次の周辺機能を使用します。

・8ビット・タイマ／イベント・カウンタ50
・8ビット・タイマH0

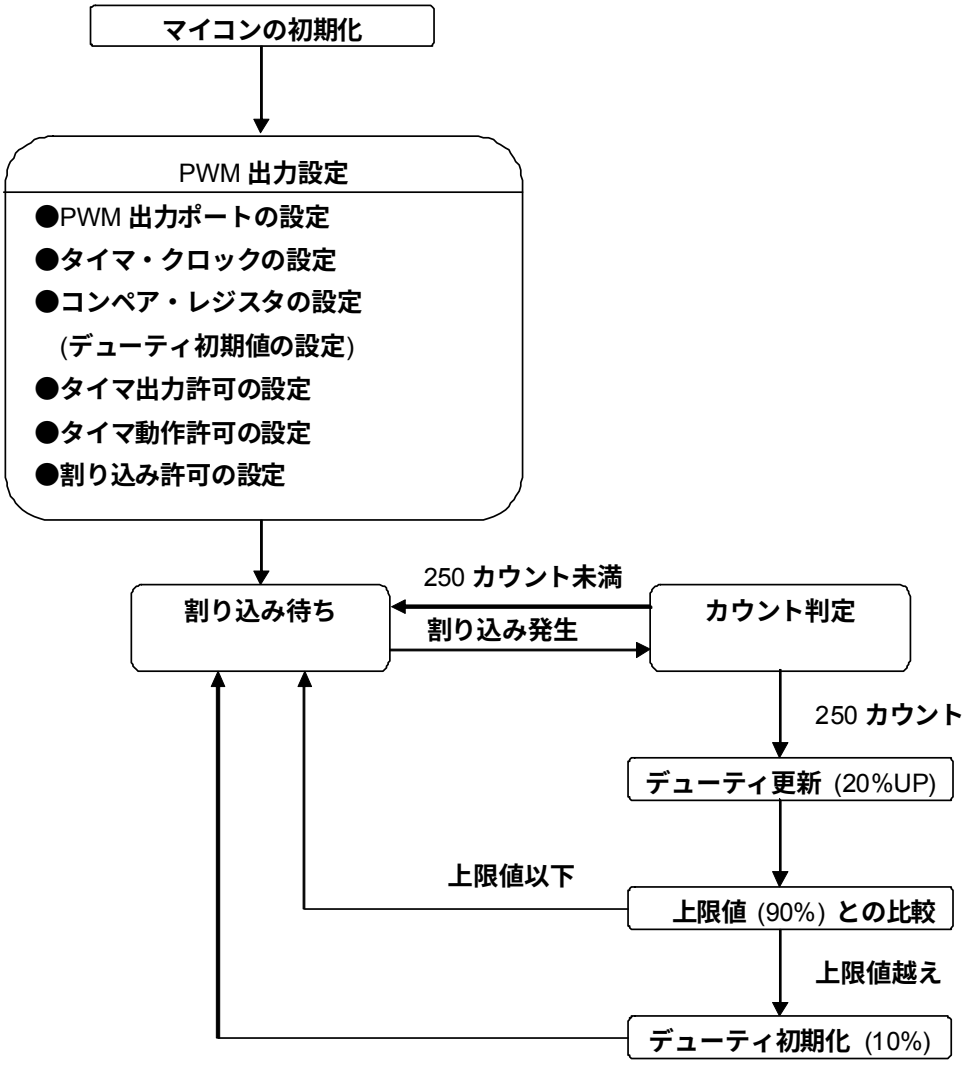
また，このサンプル・プログラムでは，上記2つのタイマについての設定例を紹介しています。
・出力ポート　　：P17/TO50 or P15/TOH0
※TM51, TMH1のサンプル・プログラムは，上記TM50, TMH0とほぼ同内容であるため，本サンプル・プログラムにおいては省略します。
なお，参考として本サンプル・プログラムにおけるTM50, TMH0との変更点をそれぞれ以下に記載します。

【TM50→TM51への変更点】
① C言語　　：　割り込み処理関数宣言をINTTM50からINTTM51に変更
　アセンブリ言語　　：　割り込みベクタ・テーブルを001EH（INTTM50）から002AH（INTTM51）に変更
② C，アセンブリ言語　：　“ PWM初期化処理 ” でのTM50の各レジスタ名称の50を51に変更
③ C，アセンブリ言語　：　“ PWM初期化処理 ” でのP17の設定をP33（TO51）に変更
④ C，アセンブリ言語　：　“ INTTM50割り込み処理 ” でのコンペア・レジスタ名の50を51に変更

【TMH0→TMH1への変更点】
① C言語　　：　割り込み処理関数宣言をINTTMH0からINTTMH1に変更
　アセンブリ言語　　：　割り込みベクタ・テーブルを001CH（INTTMH0）から001AH（INTTMH1）に変更
② C，アセンブリ言語　：　“ PWM初期化処理 ” でのTMH0の各レジスタ名称のH0をH1に変更
③ C，アセンブリ言語　：　“ PWM初期化処理 ” でのP15の設定をP16（TOH1）に変更
④ C，アセンブリ言語　：　“ INTTMH0割り込み処理 ” でのコンペア・レジスタ名のH0をH1に変更

3. 3　PWM設定と動作概要

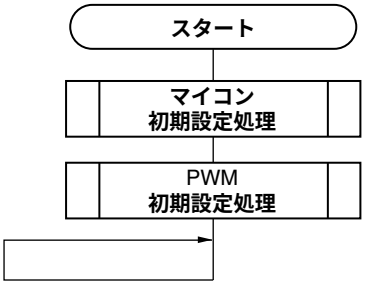
このサンプル・プログラムでは，PWMの設定を行います。
設定完了後は，設定時間（デューティ）に応じてポートへ出力を繰り返し，LEDの輝度を制御します。
動作概要については，次の状態遷移図（ステート・チャート）に示します。



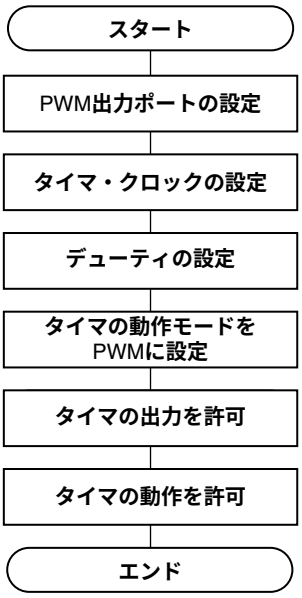
3. 4　フロー・チャート

このサンプル・プログラムのフロー・チャートを以下に示します。

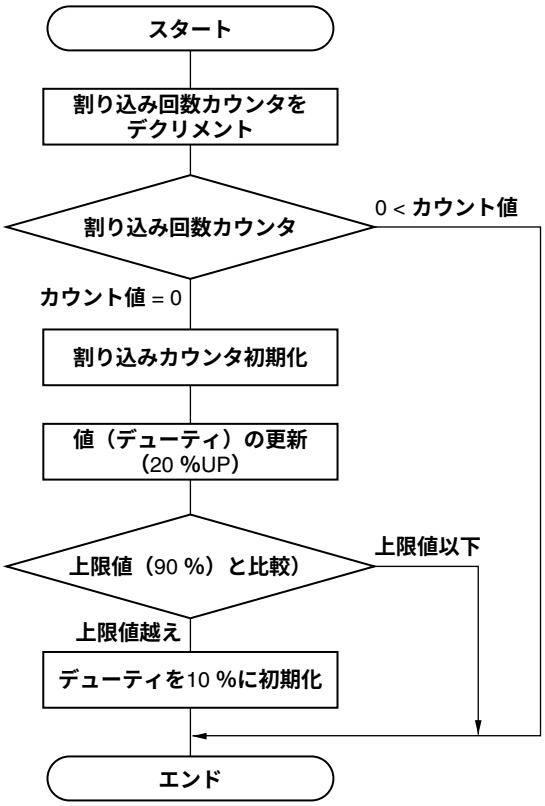
○ゼネラル・フロー (C**言語版：**Kx2_Pwm.c　**アセンブリ言語版：**Kx2_Pwm.asm)



○PWM**初期設定処理** (C**言語版：**Kx2_Pwm.c　**アセンブリ言語版：**Kx2_Pwm.asm)



○割り込み処理 (C言語版：Kx2_Pwm.c　アセンブリ言語版：Kx2_Pwm.asm)



第4章　設定方法について

この章では，PWM（8ビット・カウンタ／イベント・カウンタ50，8ビット・カウンタH0）の設定，およびメイン処理について説明します。

レジスタ設定方法の詳細については，78K0/Kx2　**ユーザーズ・マニュアル**を参照してください。

アセンブラ命令については，78K/0**シリーズ　命令編　ユーザーズ・マニュアル**を参照してください。

4. 1　前処理指令

C言語において，SFR領域に関する操作，CPU制御命令，割り込み関数などを使用するためには，#pragma指令にてソース・プログラムの冒頭に前処理指令を記述する必要があります。本サンプル・プログラムで使用する前処理指令は以下のとおりです。ここでは，TM50版のみを例として説明します。

【C言語】 (Kx2_Pwm.c)

```
/*==========================================================================
        前処理指令(#pragma指令)
==========================================================================*/
#pragma  sfr
#pragma  di
#pragma  ei
#pragma  nop
#pragma  interrupt  INTTM50 fn_intTimerPwm
```

特殊機能レジスタ（SFR）名称を記述可能にします。

DI命令を記述可能にします。

EI命令を記述可能にします。

NOP命令を記述可能にします。

割り込み関数宣言をします。

4. 2　PWMの設定

【8ビット・タイマ／イベント・カウンタ50】

PWMは，次の項目を設定します。

(1) タイマ・クロック選択

(2) デューティの設定

(3) タイマ・モードの設定

(4) タイマ出力の設定

(5) タイマの動作許可

本サンプル・プログラムでは，後述の【例】の内容で設定しています。

(1) タイマ・クロック設定

8ビット・カウンタ／イベント・カウンタ50のカウント・クロックおよびTI50端子入力の有効エッジを設定します。

図4－1　タイマ・クロック選択レジスタ50

TCL50

0	0	0	0	0	TCL 502	TCL 501	TCL 500

カウント・クロックの選択

0	0	0	TI50端子の立ち下がりエッジ
0	0	1	TI50端子の立ち上がりエッジ
0	1	0	f_{PRS}
0	1	1	$f_{PRS}/2$
1	0	0	$f_{PRS}/2^2$
1	0	1	$f_{PRS}/2^6$
1	1	0	$f_{PRS}/2^8$
1	1	1	$f_{PRS}/2^{13}$

注意1. TCL50を同一データ以外に書き換える場合は，いったんタイマ動作を停止させてから書き換えてください。

2. ビット3-7には，必ず0を設定してください。

備考　f_{PRS} ： 周辺ハードウエア・クロック周波数

(2) タイマ・モード設定

8ビット・タイマ・カウンタの動作モード，カウンタ制御，出力の状態／制御の設定をします。

図4－2　8ビット・タイマ・モード・コントロール・レジスタ50

TMC50

TCE 50	TMC 506	0	0	LVS 50	LVR 50	TCM 501	TOE 50

タイマ出力の制御

0	出力禁止
1	出力許可

アクティブ・レベルの選択（PWMモード）

0	ハイ・アクティブ
1	ロウ・アクティブ

タイマF/F出力の状態設定

0	0	変化しない
0	1	タイマ出力F/Fをクリア(0)
1	0	タイマ出力F/Fをセット(1)
1	1	設定禁止

TM50の動作モード選択

0	TM50とCR50の一致でクリア&スタート・モード
1	PWM（フリー・ランニング）モード

TM50のカウント動作制御

0	カウントを0にクリア後，カウント動作禁止（カウンタ停止）
1	カウント動作開始

注意1. LVS50とLVR50の設定は，PWMモード以外で有効になります。

2. 以下の設定は同時に行わないでください。また，設定は以下の順で行ってください。

① TMC501, TMC506を設定

② 出力を許可する場合，TOE50を設定

③ LVS50, LVR50を設定

④ TCE50を設定

3. TCE50 = 1のとき，TMC50の他のビットを設定することは禁止です。

備考1. PWMモード時は，TCE50 = 0により，PWM出力はインアクティブ・レベルになります。

2. LVS50とLVR50は読み出すと，0になっています。

3. TMC506, LVS50, LVR50, TMC501, TOE50の各ビットの値は，TCE50の値に関係なくTO50端子に反映されます。

(3) 8ビット・タイマ・コンペア・レジスタ（CR50）の設定

CR50を設定することでTO50端子から出力されるパルスのデューティを設定することができます。

CR50設定値をNとすると，デューティは以下の式で表されます。

・デューティ　＝　N / 2^8

PWMモード時は，8ビット・タイマ・カウンタ50（TM50）のカウント値がCR50に設定した値と一致するとTO50端子がインアクティブになります（割り込みは発生しません）。

図4－3　8ビット・タイマ・コンペア・レジスタ50

CR50

注意1.　CR50の値は，00H-FFHの範囲で設定します。

2.　PWMモード時は，CR50の書き換え間隔をカウント･クロック（TLC50で選択したクロック）の3カウント・クロック以上にしてください。

【例】　・1クロックを8 μsに設定（8 MHz動作時）

・PWMモードに設定

・アクティブ・レベルをハイ・レベルに設定

（サンプル・プログラムの設定と同内容）

TCL50

0	0	0	0	0	1	0	1

カウント・クロックの選択

1	0	1	$f_{PRS}/2^6$

TMC50

0→1	1	0	0	0	0	0	1

タイマ出力の制御

1	出力許可

アクティブ・レベル（PWMモード）

0	ハイ・アクティブ

タイマF/F出力の状態設定

0	0	変化しない

TM50の動作モード選択

1	PWM（フリー・ランニング）モード

TM50のカウント動作制御

0→1	カウントを0にクリア後，カウント動作禁止（カウンタ停止）→　許可

CR50

0	0	0	1	1	0	0	1

CR50値　= 25　→ デューティ初期値10 [%]　：　設定値　= 10*255/100

※ サンプル・プログラムでは，デューティ初期値を10 [%]とし，C_PWM_INIT（C言語）およびCPWMINIT（アセンブリ言語）と定義しています。

サンプル・プログラムでは以下のようになります。

【C言語】　(Kx2_Pwm.c)

```
/*=========================================================================
;
;       PWM出力デューティの定義
;
;=========================================================================
;-------------------------------------------------------------------------
;       PWM出力デューティ初期値の指定(%)
;-------------------------------------------------------------------------*/
        #define  C_PWM_INIT    10            /* デューティ10%からスタート     */
/*-------------------------------------------------------------------------
;       PWM出力増加ステップの指定(%)
;-------------------------------------------------------------------------*/
        #define  C_PWM_STEP    20            /* デューティ20%ずつ増加         */
/*-------------------------------------------------------------------------
;       PWM出力上限の指定(%)
;-------------------------------------------------------------------------*/
        #define  C_PWM_LIMIT   90            /* デューティの上限90%           */
                              •
                              •
                              •
        TCL50    =      0b00000101;          /*[タイマ・クロック選択]         */
        CR50     =      C_PWM_INIT*255/100;  /*[コンペア・レジスタ]           */
        TMC50    =      0b01000001;          /*[モード制御]                   */
        TCE50    =      1;                   /* タイマの動作許可             */
                              •
                              •
                              •
```

デューティを変更する場合，ここでの定義値を変更してください。上限は100[%]まで可能ですが，丸め誤差の影響で100[%]出力にならない場合もあります。

【コラム】デューティの設定について

PWM出力で使用するタイマごとのデューティ設定式は以下のとおりです。

・8ビット・タイマ／イベント・カウンタ50, 51

デューティ　= N / 2^8 (N：CR5n設定値)

・8ビット・タイマH0, H1

デューティ　= (M＋1) / (N＋1) (M：CMP1n設定値　N：CMP0n設定値)

【アセンブリ言語】 (Kx2_Pwm.asm)

```
;=============================================================================
;
;       PWM出力デューティの定義
;
;=============================================================================
;-----------------------------------------------------------------------------
;       PWM出力デューティ初期値の指定(%)
;-----------------------------------------------------------------------------
        CPWMINIT                EQU     10      ;デューティ10%からスタート
;-----------------------------------------------------------------------------
;       PWM出力増加ステップの指定(%)
;-----------------------------------------------------------------------------
        CPWMSTEP        EQU     20      ;デューティ20%ずつ増加
;-----------------------------------------------------------------------------
;       PWM出力上限の指定(%)
;-----------------------------------------------------------------------------
        CPWMLIMIT       EQU     90      ;デューティの上限90%
                                •
                                •
                                •
        MOV     TCL50,  #00000101B      ;[タイマ・クロック選択]
        MOV     CR50,   #CPWMINIT*255/100;[ コンペア・レジスタ]
        MOV     TMC50,  #01000001B      ;[モード制御]
        SET1    TCE50                   ;TMC50レジスタのTCE50をセット
                                •
                                •
                                •
```

デューティを変更する場合，ここでの定義値を変更してください。上限は100[%]まで可能ですが，丸め誤差の影響で100[%]出力にならない場合もあります。

【8ビット・タイマH0】

PWMは，次の項目を設定します。

(1) タイマ・クロック選択

(2) タイマ・モードの設定

(3) タイマ出力の設定

(4) 周期の設定

(5) デューティの設定

(6) タイマの動作許可

本サンプル・プログラムでは，後述の【例】の内容で設定しています。

(1) タイマ・クロック設定

8ビット・タイマH0のカウント・クロック，タイマ・モード，タイマ出力の設定をします。

図4－4　8ビット・タイマHモード・レジスタ0

TMHMD0

TMHE0	CKS02	CKS01	CKS00	TMMD01	TMMD00	TOLEV0	TOEN0

タイマ出力制御

0	出力禁止
1	出力許可

タイマ出力レベル制御（デフォルト時）

0	ロウ・レベル
1	ハイ・レベル

タイマ動作モード

0	0	インターバル・タイマ・モード
1	0	PWM出力動作モード
上記以外		設定禁止

カウント・クロックの選択

0	0	0	f_{PRS}
0	0	1	$f_{PRS}/2$
0	1	0	$f_{PRS}/2^2$
0	1	1	$f_{PRS}/2^6$
1	0	0	$f_{PRS}/2^{10}$
1	0	1	TM50の出力
上記以外			設定禁止

タイマ動作許可

0	タイマ・カウント動作停止（カウンタは0にクリア）
1	タイマ・カウント動作許可（クロックを入力することでカウント動作開始）

注意　TMHE1 = 1のとき，TMHMD1の他のビットを設定することは禁止です。
ただし，リフレッシュ（同値書き込み）することは可能です。

(2) 8ビット・タイマH・コンペア・レジスタ00, 10 (CMP00, CMP10) の設定

TMH0のPWM出力モードでは，CMP00レジスタで周期を設定し，CMP10レジスタでデューティを設定します。

8ビット・タイマH0はカウンタ値と比較し，2つの値が一致しときに割り込みを発生します。

CMP00レジスタの設定値を（N），CMP10レジスタを（M），カウント・クロックの周波数をf_{CNT}とすると，PWMパルス出力周期およびデューティは次のとおりになります。

- PWMパルス出力周期　= (N+1) / f_{CNT}
- デューティ　　= (M+1) / (N+1)

図4－5　8ビット・タイマHコンペア・レジスタ00

CMP00

注意　CMP00は，タイマ・カウント動作中に値を書き換えないでください。
ただし，タイマ・カウント動作中にリフレッシュ（同値書き込み）することは可能です。

図4－6　8ビット・タイマHコンペア・レジスタ10

CMP10

注意　タイマ・カウント動作停止（TMHE0 = 0）設定後，タイマ・カウント動作を開始する（TMHE0 = 1）場合，必ずCMP10を設定してください（CMP10への設定値が同値の場合でも，必ず再設定してください）。

【例】　・タイマ・クロックを8 [μs] に設定 (8 MHz動作時)
・PWM出力モードに設定
（サンプル・プログラムの設定と同内容）

TMHMD0

0→1	0	1	1	1	0	0	1

タイマ出力制御

1	出力許可

タイマ出力レベル制御

0	ロウ・レベル

タイマ動作モード

1	0	PWM出力モード

カウント・クロックの選択

0	1	1	$f_{PRS}/2^6$

タイマ動作許可

0→1	タイマ・カウント動作停止→動作許可

CMP00

1	1	1	1	1	0	0	1

CMP00値 = 249 → 周期2 [ms] ： 設定値 = (0.002*8000000/2^6) －1

※ サンプル・プログラムでは，周期を2 [ms] とし，C_PWM_CYCLE（C言語）およびCPWMCYCLE（アセンブリ言語）と定義しています。

CMP10

0	0	0	1	1	0	0	0

CMP10値 = 24 → デューティ初期値10 [%] ： 設定値 = (10*250/100) －1

※ サンプル・プログラムでは，デューティ初期値10[%]とし，C_PWM_INIT（C言語）およびCPWMINIT（アセンブリ言語）と定義しています。

サンプル・プログラムでは以下のようになります。

【C言語】　(Kx2_Pwm.c)

```
/*=============================================================================
;
;       PWM出力デューティの定義
;
;=============================================================================
;-----------------------------------------------------------------------------
;
;       PWMパルス出力周期用コンペア・レジスタ(CMP00)
;
;    PWMパルス出力周期（2[ms]）＝250/8000000/2^6
;    ※本プログラムではメイン・クロックを8MHz，カウント・クロックをfPRS/2^6として
;    いるので上記の計算式となる。なお，ここでは周期2msとした。
;-----------------------------------------------------------------------------*/
        #define C_PWM_CYCLE     250     /* PWMパルス出力周期2ms             */
/*-----------------------------------------------------------------------------
;       PWM出力デューティ初期値の指定(%)
;-----------------------------------------------------------------------------*/
        #define C_PWM_INIT      10      /* デューティ10%からスタート        */
/*-----------------------------------------------------------------------------
;       PWM出力増加ステップの指定(%)
;-----------------------------------------------------------------------------*/
        #define C_PWM_STEP      20      /* デューティ20%ずつ増加            */
/*-----------------------------------------------------------------------------
;       PWM出力上限の指定(%)
;-----------------------------------------------------------------------------*/
        #define C_PWM_LIMIT     90      /* デューティの上限90%              */
                                •
                                •
                                •
        TMHMD0  =       0b00111001;     /*[モード制御]                     */
        CMP00   =       C_PWM_CYCLE-1;  /*[コンペア・レジスタ]              */
        CMP10   =       C_PWM_INIT*C_PWM_CYCLE/100-1;
                                        /*[コンペア・レジスタ]              */
        TMHE0   =       1;              /*TMHMD0レジスタのTMHE0をセット
                                          →　動作開始                     */
                                •
                                •
```

デューティを変更する場合，ここでの定義値を変更してください。
上限は100[%]まで可能ですが，丸め誤差の影響で100[%]出力にならない場合もあります。

【アセンブリ言語】 (Kx2_Pwm.asm)

```
;=============================================================================
;
;       PWM出力デューティの定義
;
;=============================
;-----------------------------------------------------------------------------
;
;       PWMパルス出力周期用コンペア・レジスタ(CMP00)
;
;    PWMパルス出力周期（2[ms]）＝250/8000000/2^6
;    ※本プログラムではメイン・クロックを8MHz,カウント・クロックをfPRS/2^6として
;    いるので上記の計算式となる。なお,ここでは周期2msとした。
;-----------------------------------------------------------------------------
        CPWMCYCLE       EQU     250     ;PWMパルス出力周期2ms
;-----------------------------------------------------------------------------
;       PWM出力デューティ初期値の指定(%)
;-----------------------------------------------------------------------------
        CPWMINIT        EQU     10      ;デューティ10%からスタート
;-----------------------------------------------------------------------------
;       PWM出力増加ステップの指定(%)
;-----------------------------------------------------------------------------
        CPWMSTEP        EQU     20      ;デューティ20%ずつ増加
;-----------------------------------------------------------------------------
;       PWM出力上限の指定(%)
;-----------------------------------------------------------------------------
        CPWMLIMIT       EQU     90      ;デューティの上限90%
                                        •
                                        •
                                        •
        MOV     TMHMD0, #00111001B      ;[モード制御]
        MOV     CMP00,  #CPWMCYCLE-1    ;[コンペア・レジスタ]
        MOV     CMP10,  #CPWMINIT*CPWMCYCLE/100-1
                                        ;[コンペア・レジスタ]
        SET1    TMHE0                   ;TMHMD0レジスタのTMHE0をセット
                                        ;→　動作開始
                                        •
                                        •
                                        •
```

デューティを変更する場合，ここでの定義値を変更してください。
上限は100[%]まで可能ですが，丸め誤差の影響で100[%]出力にならない場合もあります。

4. 3　割り込みの設定

(1) 割り込み要求の設定

指定の割り込みに対して，割り込み要求フラグをクリアします。

このサンプル・プログラムでは，直接レジスタ・ビットに設定しています(【例1】，【例2】)。

図4－7　割り込み要求フラグ・レジスタ（IF0H）のフォーマット

IF0H

TMIF01	TMIF00	TMIF50	TMIFH	TMIFH	DUALI	STIF6	SRIF6
0	0		0	1	F0		

割り込み要求フラグ

0	要求クリア
1	要求セット

(2) 割り込みマスクの設定

指定の割り込み処理の許可／禁止を設定します。

このサンプル・プログラムでは，直接レジスタ・ビットに設定しています(【例1】，【例2】)。

図4－8　割り込みマスク・フラグ・レジスタ（MK0H）のフォーマット

MK0H

TMMK0	TMMK0	TMMK5	TMMK	TMMK	DUALM	STMK6	SRMK6
10	00	0	H0	H1	K0		

割り込みマスク・フラグ

0	割り込み処理許可
1	割り込み処理禁止

【例1】　・INTTM50割り込み要求（TMIF50）をクリア

・INTTM50割り込みマスク（TMMK50）をクリア（割り込み処理許可）

（サンプル・プログラムTM50編の設定と同内容）

IF0H

		0					

割り込み要求フラグ

0	要求クリア

MK0H

		0					

割り込み要求マスク・フラグ

0	割り込み処理許可

サンプル・プログラムでは1ビット・アクセスにて設定しています。

【C言語】 (Kx2_Pwm.c)

```
TMIF50  =       0;              /* INTTM50割り込み要求クリア */
TMMK50  =       0;              /* INTTM50割り込みマスク解除 */
                                •
                                •
                                •
```

【アセンブリ言語】 (Kx2_Pwm.asm)

```
CLR1    TMIF50                  ;INTTM50割り込み要求クリア
CLR1    TMMK50                  ;INTTM50割り込みマスク解除
                                •
                                •
                                •
```

【例2】　・INTTMH0割り込み要求（TMIFH0）をクリア

・INTTMH0割り込みマスク（TMMKH0）をクリア（割り込み処理許可）

（サンプル・プログラムTMH0編の設定と同内容）

IF0H

			0				

割り込み要求フラグ

0	要求クリア

MK0H

			0				

割り込み要求マスク・フラグ

0	割り込み処理許可

サンプル・プログラムでは1ビット・アクセスにて設定しています。

【C言語】 (Kx2_Pwm.c)

```
TMIFH0   =        0;              /* INTTMH0割り込み要求クリア */
TMMKH0   =        0;              /* INTTMH0割り込みマスク解除 */
                                  •
                                  •
                                  •
```

【アセンブリ言語】 (Kx2_Pwm.asm)

```
CLR1     TMIFH0                   ;INTTMH0割り込み要求クリア
CLR1     TMMKH0                   ;INTTMH0割り込みマスク解除
                                  •
                                  •
                                  •
```

4.4　ポートの設定

(1) ポートの出力の設定

このサンプル・プログラムではPWM出力の処理結果を確認するために，P1を出力ポートとして使用します。ここでは，8ビット・カウンタ／イベント・カウンタ50（TM50）版を例にとって説明します。このTM50では，P17（TO50）を出力ポートとして使用します。サンプル・プログラムTM50版では，後述の【例】のような内容で設定しています。

図4－9　ポート・モード・レジスタ1（PM1）のフォーマット

PM1

PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10

P1n（n = 0-7）端子の入出力モードの選択

0	出力モード
1	入力モード

図4－10　ポート・レジスタ1（P1）のフォーマット

P1

P17	P16	P15	P14	P13	P12	P11	P10

P1n（n = 0-7）端子の出力ラッチのレベル選択

0	ロウ・レベル出力
1	ハイ・レベル出力

【例】　・P17を出力ポートに設定

（サンプル・プログラムTM50版の設定と同内容）

PM1

0	0	0	0	0	0	0	0

P17端子の入出力モードの選択

0	出力モード

P1

0	0	0	0	0	0	0	0

PM1の設定値は「00000000」，P1の設定値は「00000000」となります。

【C言語】

```
PM1     =       0b00000000;
P1      =       0b00000000;
```

【アセンブリ言語】

```
MOV     PM1,    #00000000B
MOV     P1,     #00000000B
```

4.5　メイン処理

メイン処理では，次の動作を行います。

【C言語】 (Kx2_Pwm.c)

・PWM出力設定関数を呼び出します（タイマ処理を開始します）。

・割り込み回数カウンタを初期化（250回）します。

・メイン処理内では何もせず，割り込みを待つのでNOP命令のみのループ処理になります。

```
/************************************************************************
;
;       メインループ
;       （PWM出力をし続け，2ms毎の割り込みを待つ）
;************************************************************************/
void main(void){

        g_ucIntCnt  =       250;        /* 割り込みカウンタを初期化 */
        fn_InitTimer();                 /* PWM出力タイマの初期化    */
        EI();                           /* ベクタ割り込み許可       */

        while (1){
                NOP();
        }
}
```

PWM出力の初期化を行い，処理を開始します。

メイン処理内ではNOP命令のみで割り込みを待ちます。

【アセンブリ言語】 (Kx2_Pwm.asm)

・割り込み回数カウンタを初期化（250回）します。

・メイン処理内では何もせず，割り込みを待つのでNOP命令のみのループ処理になります。

```
;*********************************************************************
;
;       メインループ
;       (PWM出力をし続け,2ms毎の割り込みを待つ)
;*********************************************************************
        MOV     RINTCNT,#250            ;割り込みカウンタを初期化
        EI                              ;ベクタ割り込み許可

MMAIN:
        NOP                             ;何もしないで,割り込み待ち
        BR      MMAIN
```

割り込みカウンタを初期化します。

メイン処理内ではNOP命令のみで割り込みを待ちます。

※アセンブリ言語版でのPWM出力の初期化は，メイン処理に入る前に，PWM出力の設定，割り込みの設定およびポートの設定で示したアセンブリ言語版の設定例と同じ内容で初期化しています。

4. 6　割り込み処理

割り込み処理では，次の動作を行います。

【C言語】 (Kx2_Pwm.c)

・割り込み回数カウンタをデクリメントします。

・割り込み回数カウンタが0になったら（250回目）デューティを更新。

・デューティが上限値（90 [%]）を越えたら，初期値（10 [%]）に戻す。

```
/************************************************************************
;       割り込み処理
;************************************************************************/
__interrupt void fn_intTimerPwm(void)
{
     unsigned   int     duty_CMP10;
```

割り込み回数250回目に到達したら，割り込みカウンタを初期化し，デューティを更新します。

```
     g_ucIntCnt--;                          /* 割り込みが入る毎に割り込みカウンタをデクリメント */
     if(g_ucIntCnt == 0){                   /* 割り込みカウンタ＝0ならば処理実行し，
                                               Noならば処理せずに抜ける                        */
        g_ucIntCnt      =       250;
                                            /* 割り込みカウンタを初期化                        */
        duty_CMP10      =       CMP10;
                                            /* 現状の設定値を読み出し                          */
        duty_CMP10      +=      C_PWM_STEP*C_PWM_CYCLE/100;
                                            /* 値の更新(本プログラムでは20%UP)                 */
```

デューティの上限値（90%）と比較して，上限値を越えていたらデューティを初期化（10%）します。

```
        if(duty_CMP10   >       C_PWM_LIMIT*C_PWM_CYCLE/100-1){
                                            /* 上限値より値が大なら処理実行                    */
                duty_CMP10      =       C_PWM_INIT*C_PWM_CYCLE/100-1;
                                            /* デューティを10%に初期化                         */
        }
        CMP10   =       (char)duty_CMP10;
                                            /* 新しい比較値を設定                              */
     }
}
```

【アセンブリ言語】 (Kx2_Pwm.asm)

・割り込み回数カウンタをデクリメント。

・割り込み回数カウンタが0になったら（250回目）デューティを更新します。

・デューティが上限値（90 [%]）になったら初期値（10 [%]）に戻します。

```
;*********************************************************************
;       INTTMH0割り込み処理
;*********************************************************************
IINTPWM:
        PUSH    AX                      ;AXレジスタのデータをスタックへ退避
```

割り込み回数250回目に到達したら，割り込みカウンタを初期化し，デューティを更新します。

```
        DBNZ    RINTCNT,$HINTEND        ;INTTMH0カウンタをデクリメントし,
                                        ;0にならなければINTENDに分岐

        MOV     RINTCNT,#250            ;INTTMH0カウンタを初期化

        MOV     A,      CMP10           ;現状の設定値を読み出し

        ADD     A,      #CPWMSTEP*CPWMCYCLE/100
                                        ;値の更新(本プログラムでは20%上げる)
```

デューティの上限値（90%）と比較して，上限値を越えていたらデューティを初期化（10%）します。

```
        BC      $HINTINIT               ;値がオーバーフローしたら初期値に戻す
        CMP     A,      #CPWMLIMIT*CPWMCYCLE/100-1
                                        ;上限値と比較
        BZ      $HINTNEXT               ;上限値なら継続
        BC      $HINTNEXT               ;上限値未満なら継続
HINTINIT:
        MOV     A,      #CPWMINIT*CPWMCYCLE/100-1
                                        ;デューティを10%に初期化
HINTNEXT:
        MOV     CMP10,  A               ;新しい比較値を設定
HINTEND:
        POP     AX                      ;AXレジスタのデータを復帰
        RETI
```

第5章　システム・シミュレータ SM+での動作確認

この章では，  のアイコンを選択してダウンロードしたC言語用のファイルを用い，サンプル・プログラムが，システム・シミュレータ SM+ for 78K0/Kx2でどのように動作するかを説明します。

5. 1　サンプル・プログラムのビルド

サンプル・プログラムをSM+ for 78K0/Kx2（以降，「SM+」と表記します）で動作確認をするために，サンプル・プログラムをビルドしてから，SM+を起動する必要があります。ここでは，  でダウンロードしたC言語用のファイルを用いて，統合開発環境　PM+にてビルドしてから，SM+を起動するまでの動作の一例を説明します。

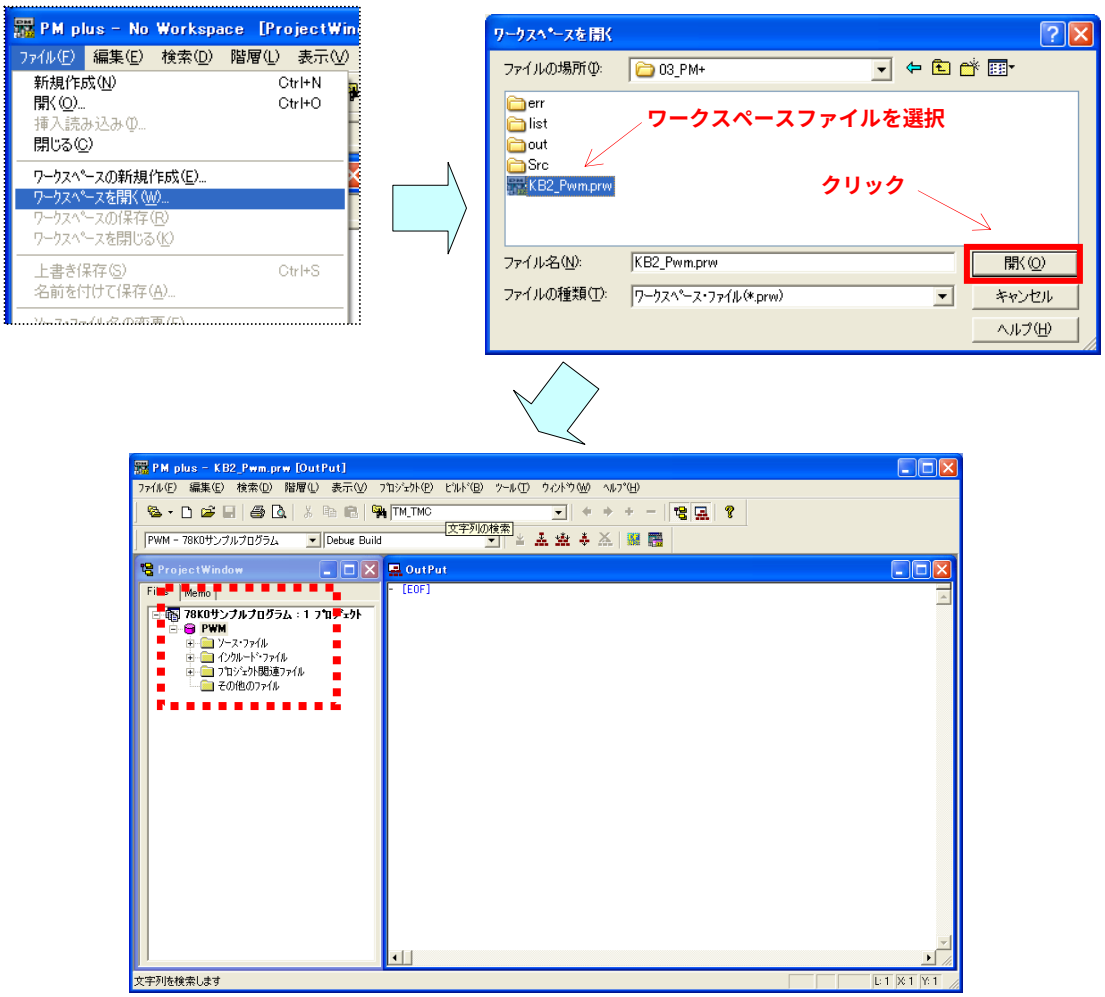
PM+操作方法の詳細については，PM+ プロジェクト・マネージャ　ユーザーズ・マニュアルを参照してください。

(1) PM+を起動してください。

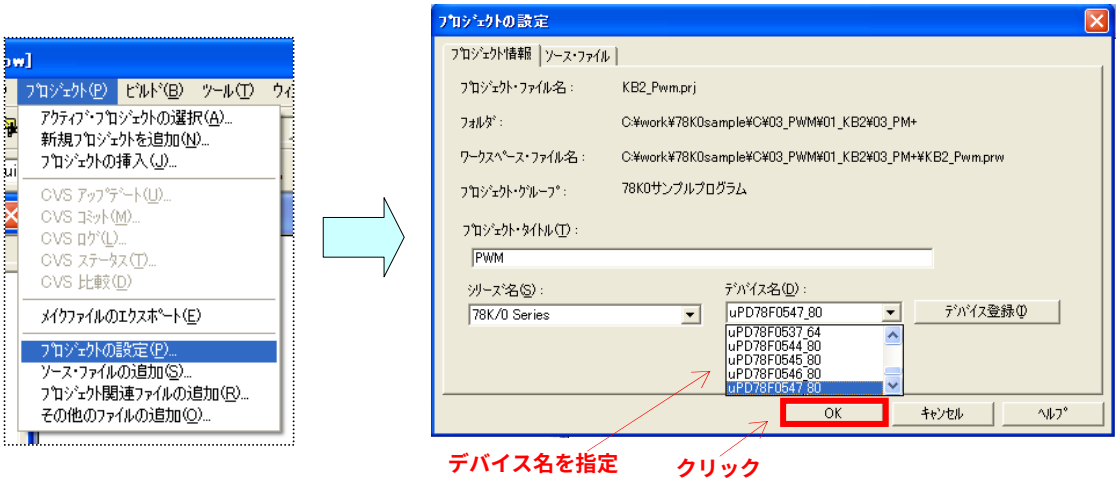
(2) [ファイル] → [ワークスペースを開く] から，「Kx2_Pwm.prw」[注]を選択し，[開く] ボタンをクリックしてください。ワークスペースが作成され，その中にソース・ファイルが自動的に読み込まれます。

注　ファイル名の”x”部分は対象デバイスにあわせて変更してください。

ex) 78K0/KB2の場合「KB2_Pwm.prw」

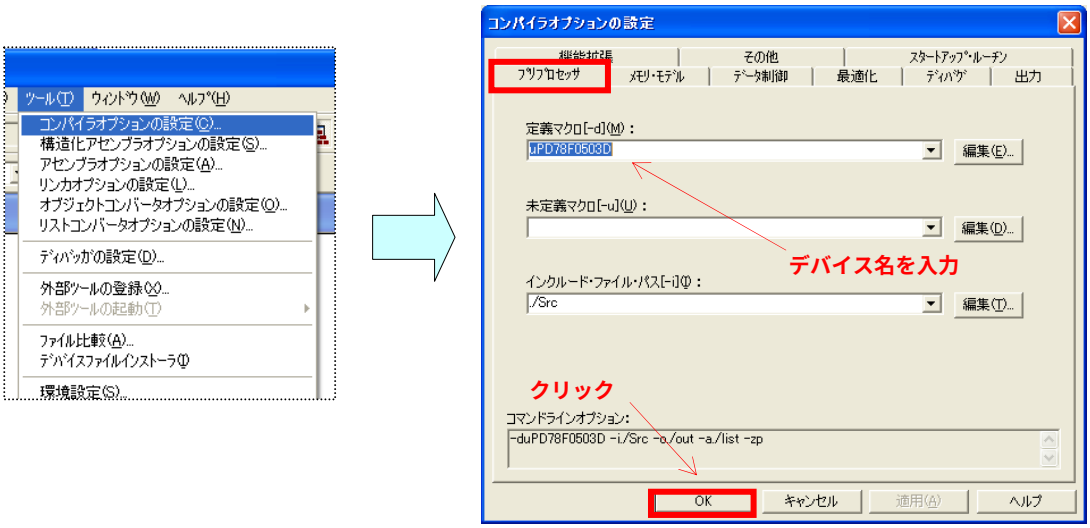


(3) [プロジェクト] → [プロジェクトの設定] を選択してください。[プロジェクトの設定] 画面が表示されたら，使用するデバイス名を選択（デフォルトでは，ROM/RAMサイズの最も大きいデバイスが選択）し，[OK] ボタンをクリックしてください。



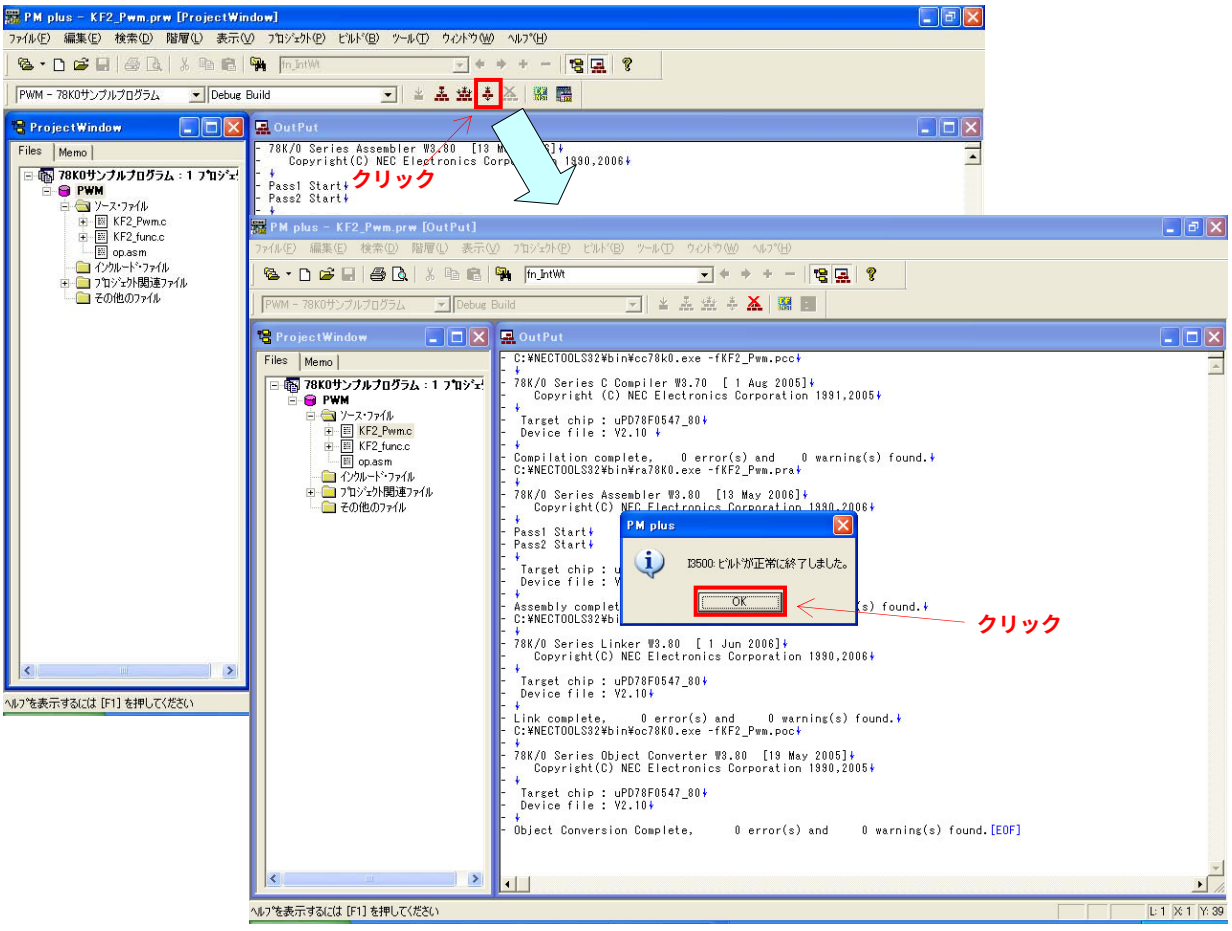
※μPD78F0500_36，μPD78F0501_36，μPD78F0502_36，μPD78F0503_36は選択しないでください。

(4) [ツール] → [コンパイラオプションの設定] を選択してください。[コンパイラオプションの設定] 画面が表示されたら，[プリプロセッサ] タグページが表示されているのを確認し，その中の定義マクロ欄に使用するデバイス名を入力し，[OK] をクリックします。



(5)  （「ビルド→ディバグ」ボタン）をクリックしてください。ソース・ファイルの「Kx2_Pwm.c」と「Kx2_func.c」と「Kx2_op.asm」が正常にビルドされると，「I3500:ビルドが正常に終了しました」というメッセージ画面が表示されます。

(6) メッセージ画面にある［OK］ボタンをクリックすると， SM+が自動的に立ち上がります。



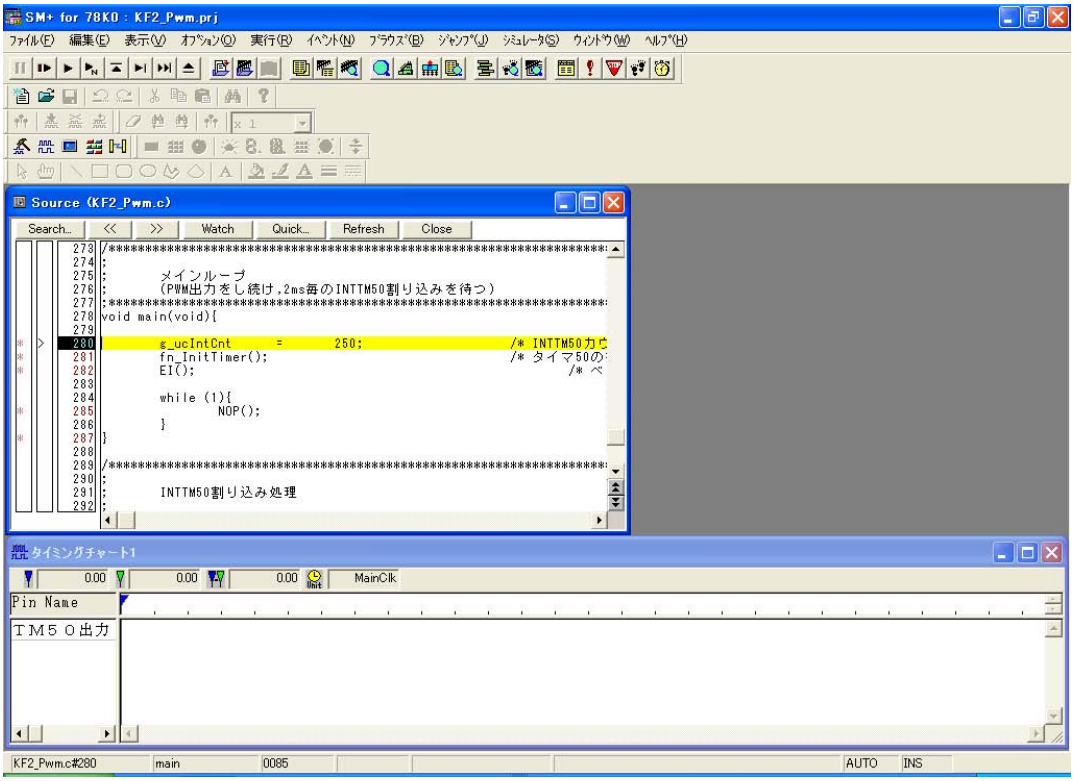
5. 2　SM+での動作

ここでは，SM+の入出力パネル・ウインドウやタイミング・チャート・ウインドウ上での動作確認の例を説明します。

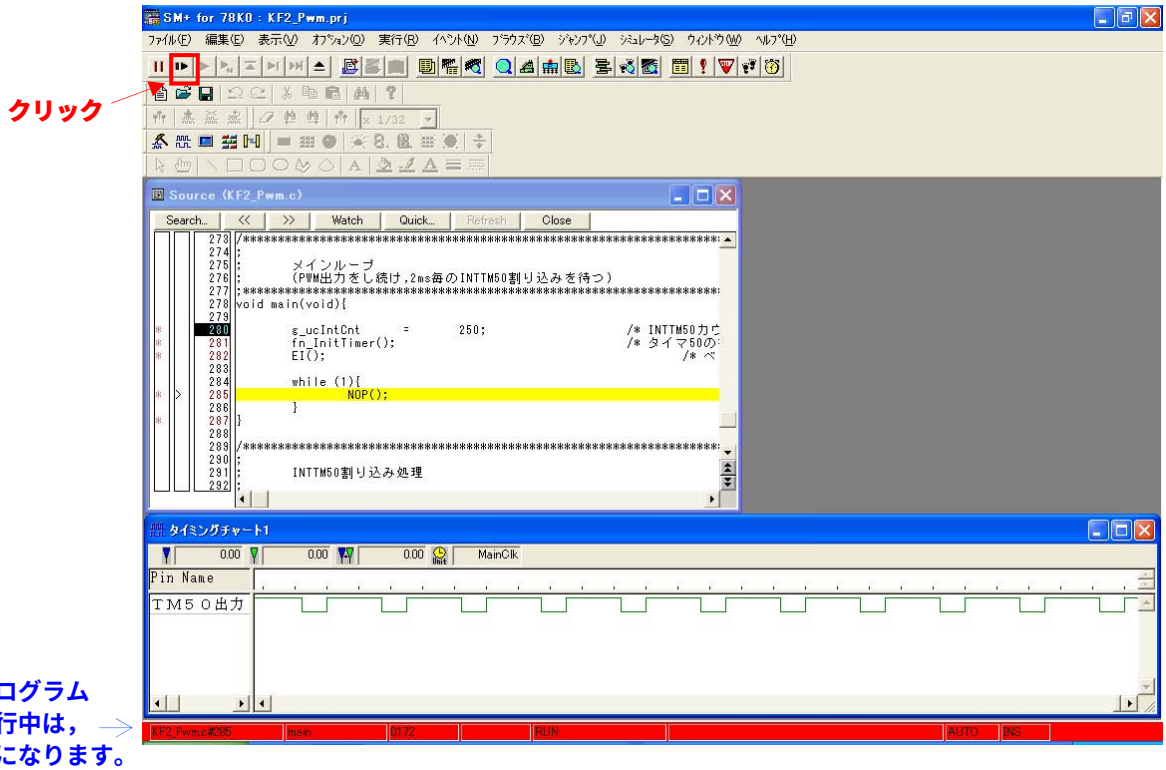
SM+操作方法の詳細については，SM+ システム・シミュレータ　操作編　ユーザーズ・マニュアルを参照してください。

なお，ここではTM50版を例にとって説明しています。

（1）PM+の「ビルド→ディバグ」からSM+を起動（5. 1を参照）すると，次のような画面になります。

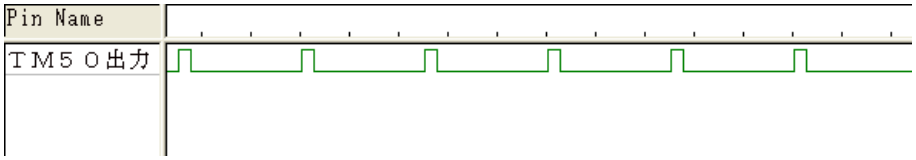
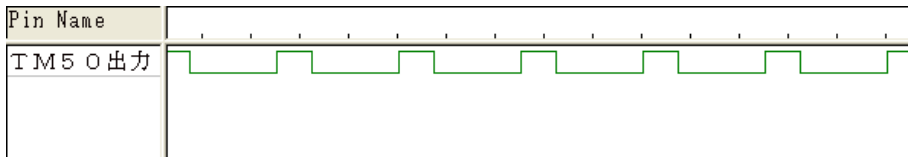
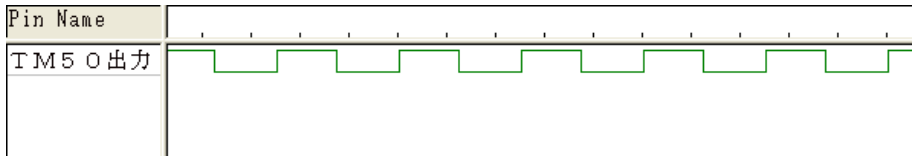
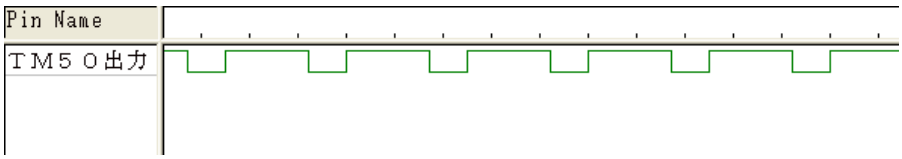
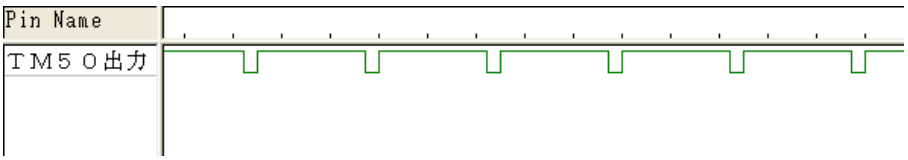


(2)  （「リスタート」ボタン）をクリックしてください。CPUリセット後，プログラムが実行され，次のような画面になります。



(3) プログラムを実行すると，タイミング・チャート上にP17の出力結果が表示されるのがわかります。

以下の表に示したタイミング・チャートのように，デューティが変化していきます。

デューティ比	タイミング・チャート・ウインドウ
10%	
30%	
50%	
70%	
90%	

5. 3　オンチップ･デバッグ時の注意

ここでは，サンプル・プログラムを用いて，オンチップ・デバッグを行う際の手順を説明します。
オンチップ・デバッグ機能については，ユーザーズ・マニュアルを参照してください。

(1) オプション・バイトの設定

本サンプル・プログラムはオプション・バイトの初期設定でオンチップ・デバッグ禁止になっています。
オプション・バイトを設定し直して，オンチップ・デバッグを許可します。
オプション・バイト設定は Kx2_op.asm で行っています。
次に，そのKx2_op.asmファイル内のオンチップ・デバッグ設定部分のみ抜粋して記載します。

```
;                                   ↓       ★印が設定値
        DB      00000000B           ;0084H  :[オンチップデバッグ]
;               ||||||++---         OCDEN1-0 :[オンチップデバッグ動作制御]
;               ||||||              ★       00:動作禁止
;               ||||||                      01:設定禁止
;               ||||||                      10:動作許可（認証失敗でフラッシュ消去せず）
;               ||||||                      11:動作許可（認証失敗でフラッシュ消去）
;               ++++++----------     0       必ず0に設定
```

動作許可（認証失敗でフラッシュ消去せず）に設定

```
;                                   ↓       ★印が設定値
        DB      00000010B           ;0084H  :[オンチップデバッグ]
;               ||||||++---         OCDEN1-0 :[オンチップデバッグ動作制御]
;               ||||||                      00:動作禁止
;               ||||||                      01:設定禁止
;               ||||||              ★       10:動作許可（認証失敗でフラッシュ消去せず）
;               ||||||                      11:動作許可（認証失敗でフラッシュ消去）
;               ++++++----------     0       必ず0に設定
```

(2) オンチップ・デバッグ使用領域の確保（アセンブリ言語版のみ）

アセンブリ言語版はオンチップ・デバッグ使用領域を確保する必要があります。

本サンプル・プログラムでは，以下のようにオンチップ・デバッグ使用領域を確保しています。

```
;=============================================================================
;         ベクタテーブル
;
;  このサンプル・プログラムでは割り込みは使用していない。割り込み
;ベクタ・テーブルは全て不要割り込み処理アドレスに定義する。
;=============================================================================
TVECTTBL        CSEG    AT      0000H

        DW      IRESET          ;0000H RESET入力，POC，LVI，WDT
;       DW      IINIT           ;0002Hはオンチップデバッグ用に空ける

TVECT_TBL1      CSEG    AT      0004H
        DW      IINIT           ;0004H INTLVI
```

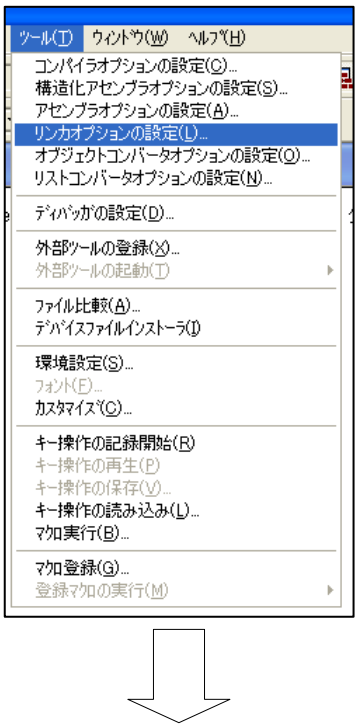
0002H番地はオンチップ・デバッグ使用領域として空けるため，0002H番地はコメント・アウトして，CSEGで再び0004H番地を定義しています。

※C言語版では不要です。

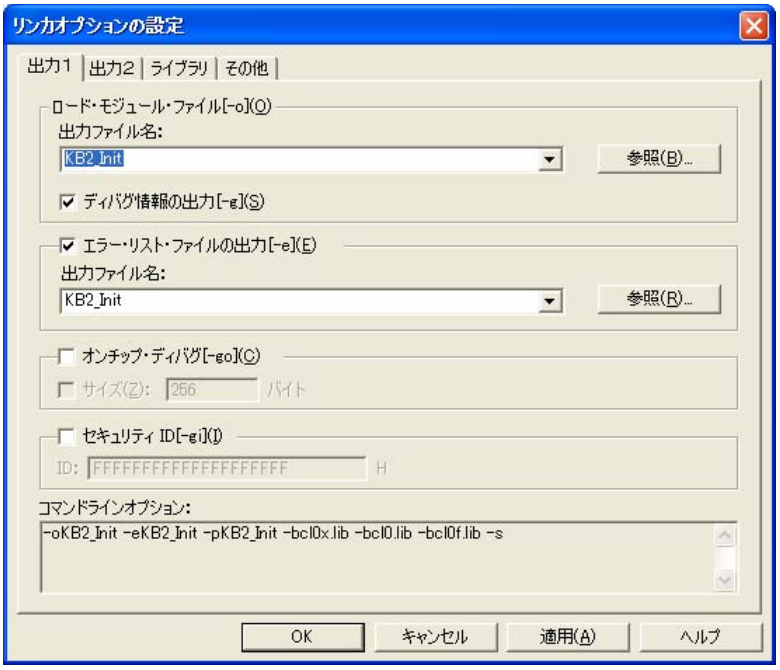
(3) リンカの設定

オンチップ・デバッグを行う場合，ビルドの際，リンカの設定を行う必要があります。

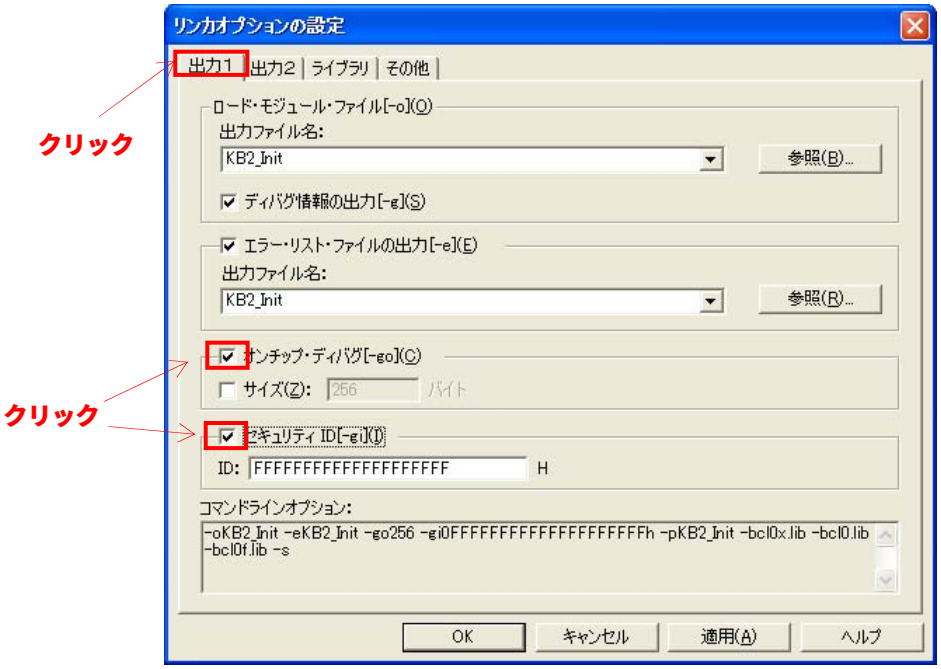
PM+の「ツール」メニューから「リンカオプションの設定」を選択してください。



「リンカオプションの設定」を選択するとリンカオプションの設定ダイアログが表示されます。



リンカオプションの設定ダイアログの「出力1」タブ上にある「オンチップ・ディバグ」と「セキュリティID」のチェックボックスをONしてください。



OKボタンを押下して設定完了です。

5. 4　開発環境のダウンロード，インストール

78K0/Kx2マイクロコントローラの開発ツールのフリーツールは，次のサイトより入手可能です。

→http://www.necel.com/micro/ja/freesoft/78k0/kx2/index.html

「SM+ for 78K0/Kx2」「RA78K0」「CC78K0」「78K0/Kx2用デバイス・ファイル」の4ファイルをダウンロードし，インストールすることで，サンプル・プログラムの動作確認が可能となります。

ダウンロード，インストールは，上記サイトの画面および説明に従って，行ってください。

備考1.　PM+は，RA78K0に同封されています。

2.　ダウンロード後，登録したEメール・アドレスに，RA78K0，CC78K0，SM+ for 78K0/Kx2のプロダクトIDが送付されます。このプロダクトIDは，各ツールのインストール時に必要となります。

第6章　関連資料

資　料　名		和文／英文
78K0/Kx2　ユーザーズ・マニュアル		PDF
78K/0シリーズ　命令編　ユーザーズ・マニュアル		PDF
RA78K0　アセンブラ・パッケージ	言語編	PDF
ユーザーズ・マニュアル	操作編	PDF
CC78K0　Cコンパイラ	言語編	PDF
ユーザーズ・マニュアル	操作編	PDF
PM+　プロジェクト・マネージャ　ユーザーズ・マニュアル		PDF
SM+　システム・シミュレータ　操作編　ユーザーズ・マニュアル		PDF

付録A　改版履歴

版　数	発行年月	改版箇所	改版内容
第1版	May 2009	－	－

【発　行】

NECエレクトロニクス株式会社

〒211-8668　神奈川県川崎市中原区下沼部1753

電話（代表）：(044)435−5111

【ホームページ】

NECエレクトロニクスの情報がインターネットでご覧になれます。

URL（アドレス）　http://www.necel.co.jp/

【資料請求先】

NECエレクトロニクスのホームページよりダウンロードいただくか，NECエレクトロニクスの販売特約店へお申し付けください。

お問い合わせ先

【営業関係，デバイスの技術関係お問い合わせ先】

半導体ホットライン

（電話：午前 9:00〜12:00，午後 1:00〜5:00）

電　話　：(044)435−9494

E-mail　：info@necel.com

【マイコン開発ツールの技術関係お問い合わせ先】

開発ツールサポートセンター

E-mail　：toolsupport-micom@ml.necel.com

C0901-A-01