

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ひ申し上げます。

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、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と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。

アプリケーション・ノート

V850ES/Jx3

サンプル・プログラム（初期設定）

LED点灯のスイッチ制御編

この資料は、サンプル・プログラムの「初期設定」の動作概要と、マイコンの基本的な初期設定を説明したものです。サンプル・プログラムでは、クロック周波数の選択や入出力ポートの選択など、マイコンの基本的な使用する周辺の初期設定を行ったあとに、1つのスイッチ入力により、2つのLED点灯を制御します。

対象デバイス

V850ES/JJ3 マイクロコントローラ
 V850ES/JG3 マイクロコントローラ

目次

- 第1章 概要 ... 3
 - 1.1 初期化 ... 3
 - 1.2 メイン処理動作の内容 ... 3
- 第2章 回路図 ... 5
 - 2.1 回路図 ... 5
 - 2.2 周辺ハードウェア ... 6
- 第3章 ソフトウェアについて ... 7
 - 3.1 ファイル構成 ... 7
 - 3.2 使用する内蔵周辺機能 ... 7
 - 3.3 初期設定と動作概要 ... 8
 - 3.4 フロー・チャート ... 9
 - 3.5 V850ES/JJ3とV850ES/JG3の違い ... 10
 - 3.6 ROM化について ... 10
 - 3.7 セキュリティIDについて ... 12
 - 3.8 MINICUBE2を用いたオンチップ・デバッグについて ... 14
 - 3.9 #pragma指令... 17
- 第4章 レジスタ設定について ... 18
 - 4.1 システム・ウェイト・コントロール・レジスタ設定 ... 19
 - 4.2 特定レジスタの設定 ... 20
 - 4.3 オンチップ・デバッグ・モード・レジスタの通常動作モード設定 ... 22
 - 4.4 内蔵発振モード・レジスタ（RCM）設定... 24
 - 4.5 ウォッチドッグ・タイマ2の設定... 25
 - 4.6 クロックの設定... 26
 - 4.7 ポート設定... 31
 - 4.8 メイン処理... 35
- 第5章 関連資料 ... 39
- 付録A プログラム・リスト ... 40

資料番号 U19837JJ1V0AN00
 発行年月 May 2009 NS

- 本資料に記載されている内容は2009年05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に、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と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。

第1章 概 要

このサンプル・プログラムでは、クロック周波数の選択、ポート入出力の設定など、V850ES/Jx3マイクロコントローラの基本的な初期設定を行います。また、初期設定完了後のメイン処理動作では、1つのスイッチ入力により、2つのLEDの点灯を制御します。

1.1 初期化

<初期設定の主な内容>

- ・システム・ウェイト・コントロール・レジスタを2クロック・ウェイトに設定
- ・オンチップ・デバッグ・モード・レジスタを通常動作モードに設定
- ・内蔵発振器を停止
- ・ウォッチドッグ・タイマ2動作停止
- ・システム・クロックはPLLを使用し8通倍して32 MHzに設定
- ・未使用ポートの設定
- ・SW入力ポート，LED制御ポートの設定

<ROM化>

- ・ROM化処理（初期値あり変数の初期化）

1.2 メイン処理動作の内容

V850ES/Jx3マイクロコントローラにて、スイッチ（SW1）入力回数に応じて、2つのLED（LED1, LED2）の点灯を制御します。

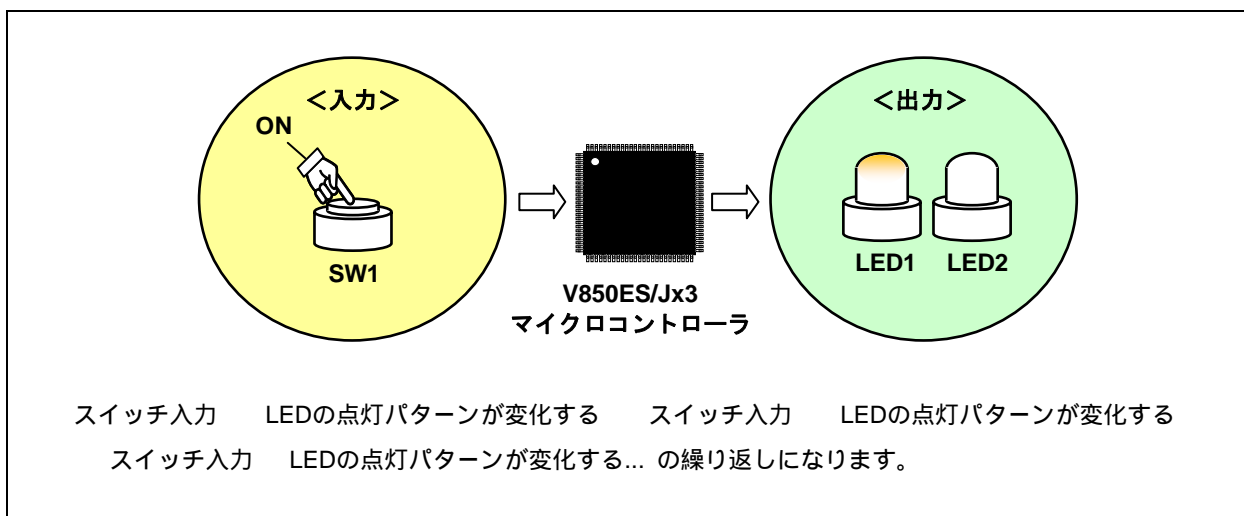


表1 - 1 LED点灯パターン

スイッチ (SW1) の入力回数 ^注	LED1	LED2
0回	OFF	OFF
1回	ON	OFF
2回	ON	ON
3回	OFF	ON

注 4回目以降の入力は0回目以降の繰り返しになります。

注意 デバイス使用上の注意事項については、製品のユーザーズ・マニュアル (V850ES/Jx3) を参照してください。



【コラム】チャタリングとは

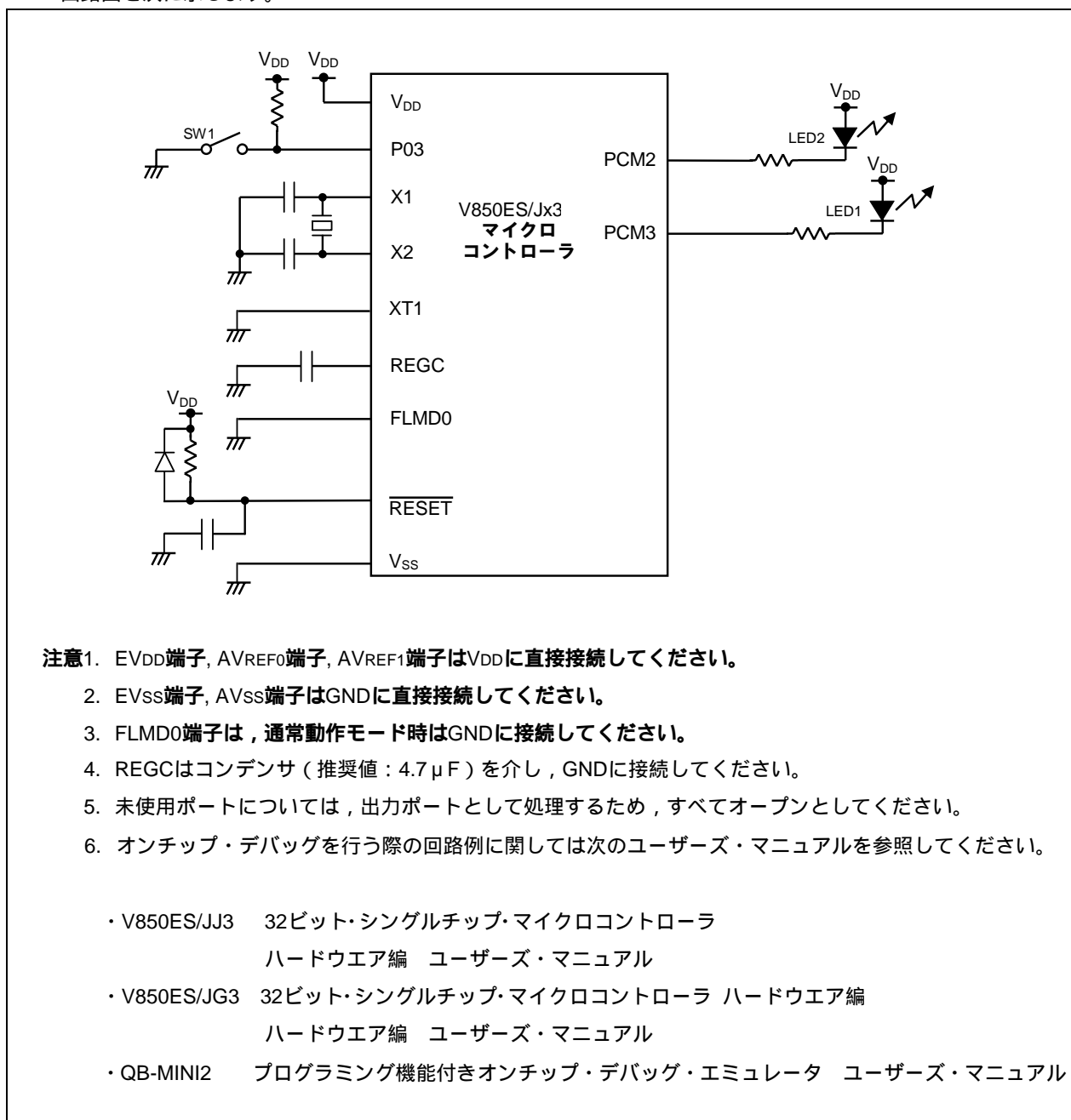
スイッチが切り替わった直後に、接点が機械的にばたつくことにより、電気信号がONとOFFを繰り返す現象のことです。

第2章 回路図

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図および周辺ハードウェアを説明します。

2.1 回路図

回路図を次に示します。



2.2 周辺ハードウェア

使用する周辺ハードウェアを次に示します。

(1) スイッチ (SW1) : LED点灯制御用の入力として、スイッチを使用します。



(2) LED (LED1, LED2) : スイッチ入力回数に対応した出力として、LEDを使用します。

第3章 ソフトウェアについて

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・プログラムの初期設定と動作概要、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。

ファイル名（ツリー構造）	説明	同封圧縮（*.zip）ファイル	
			
<ul style="list-style-type: none"> • conf <ul style="list-style-type: none"> — crtE.s — APPNOTE_LED.dir — APPNOTE_LED.prj — APPNOTE_LED.prw • src <ul style="list-style-type: none"> — main.c — minicube2.s 	スタート・アップ・ルーチン・ファイル ^{注1}	-	
	リンク・ディレクティブ・ファイル ^{注2}		
	統合開発環境 PM+用プロジェクト・ファイル	-	
	統合開発環境 PM+用ワーク・スペース・ファイル	-	
	マイコンのハードウェア初期化処理とメイン処理を記述したC言語ソース・ファイル		
	MINICUBE2用の領域予約を行う ソース・ファイル		

注1. ワークスペース新規作成時の「スタート・アップ・ファイルの指定」時に、「サンプルをコピーして使用する(C)」を選択した際にコピーされるスタート・アップ・ファイル。（デフォルト・インストール・パスであれば、C:\Program Files\NEC Electronics Tools\CA850\使用バージョン\lib850\r32\crtE.s のコピーとなります。）

2. ワークスペース新規作成時の「リンク・ディレクティブ・ファイルの指定」時に、「サンプルを作成して使用する(C)」を選択し、「メモリの使用方法：内蔵メモリのみ(I)」をチェックした際に、自動生成されるリンク・ディレクティブ・ファイルに、MINICUBE2用のセグメントを追加したもの。（デフォルト・インストール・パスであれば、C:\Program Files\NEC Electronics Tools\PM+\使用バージョン\bin\w_data\V850_i.dat が基準となります。）

備考  : ソース・ファイルのみ同封
 : 統合開発環境 PM+で使用するファイルを同封

3.2 使用する内蔵周辺機能

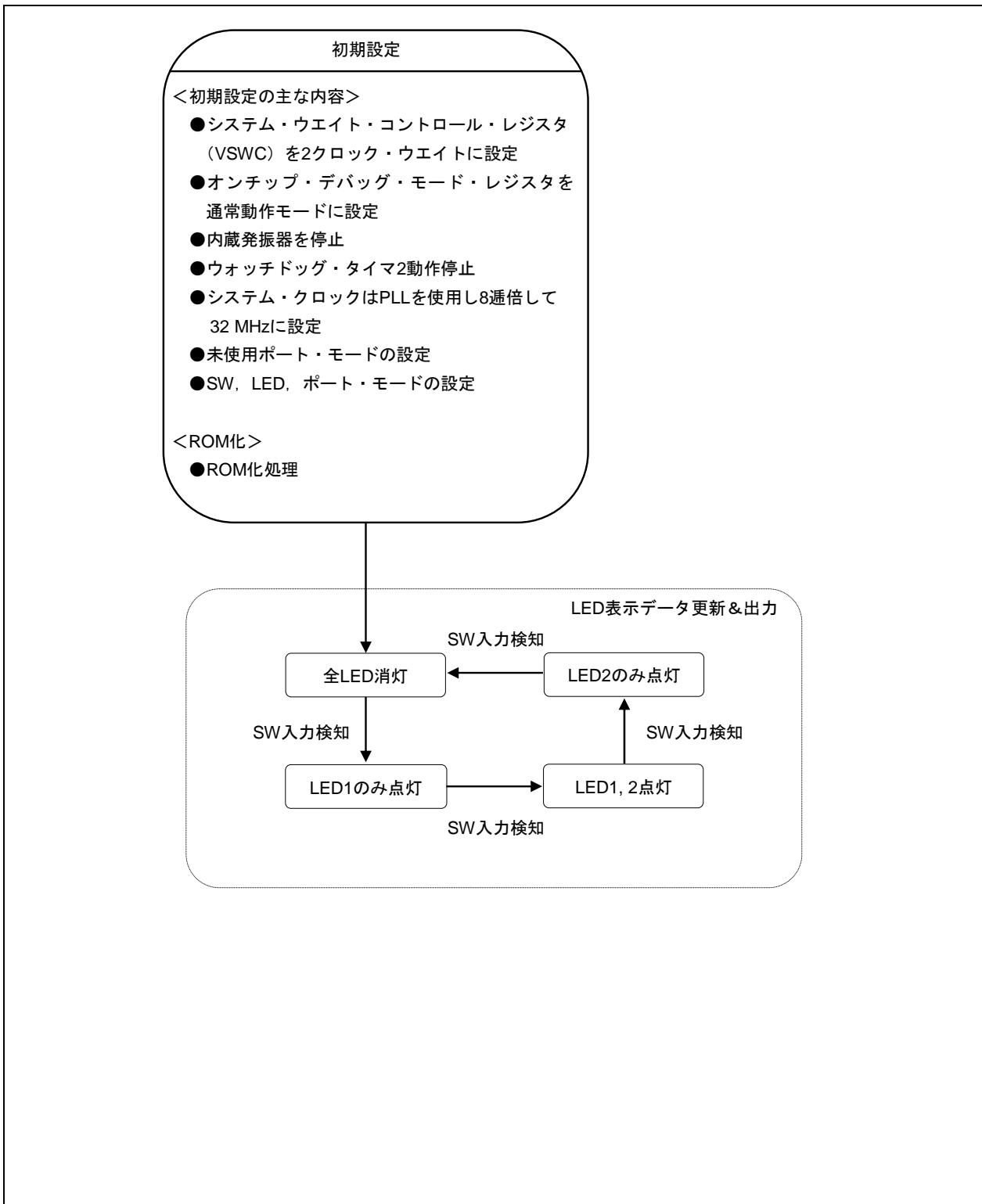
このサンプル・プログラムでは、マイコンに内蔵する次の周辺機能を使用します。

- 入力ポート（スイッチ入力用） : P03（SW1）
- 出力ポート（LED点灯用） : PCM3（LED1）, PCM2（LED2）

3.3 初期設定と動作概要

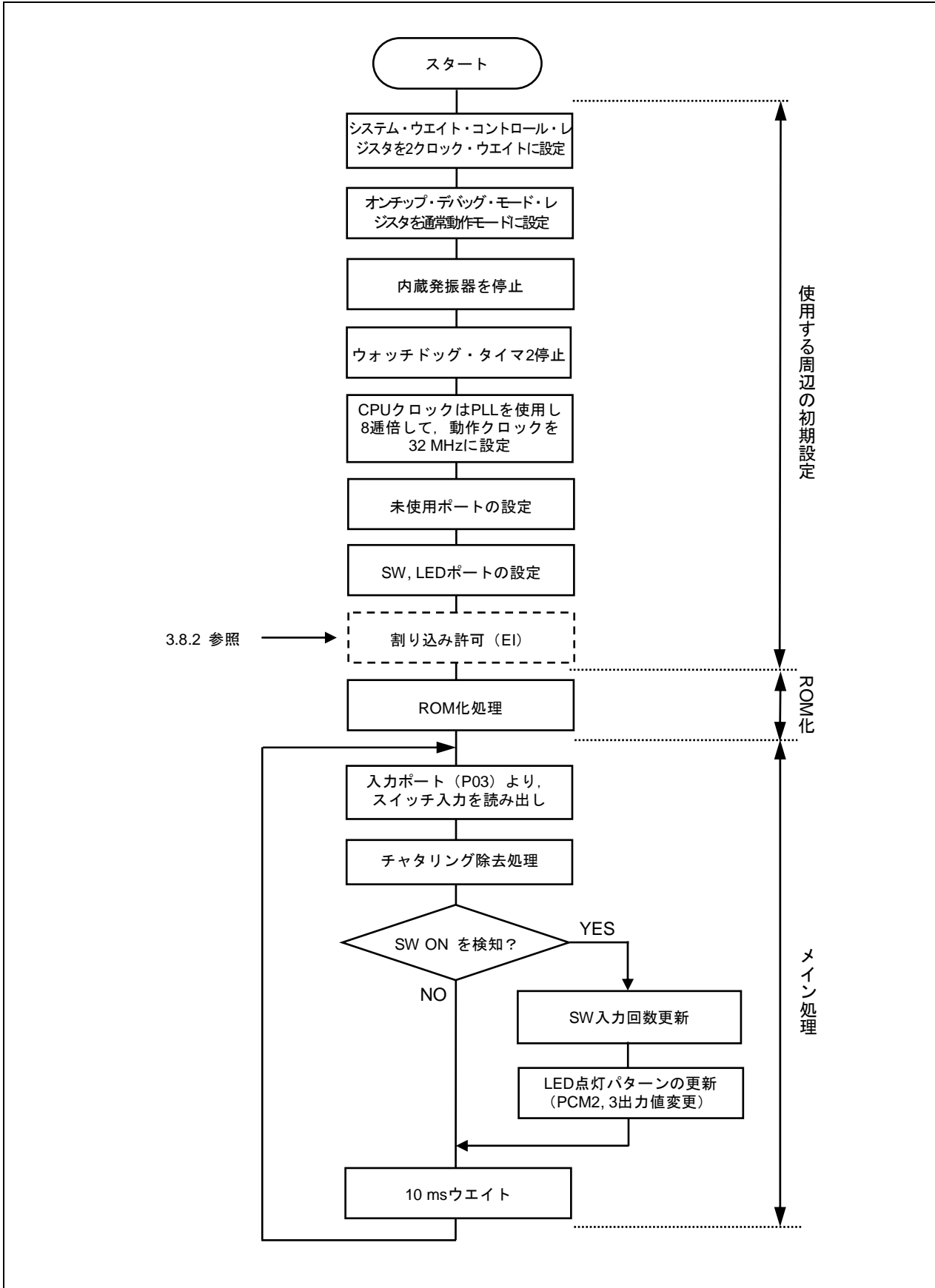
このサンプル・プログラムでは、初期設定にて、クロック周波数の選択、ウォッチドッグ・タイマ2の停止設定、入出力ポートの設定などを行います。

初期設定完了後は、スイッチ（SW1）入力回数に応じて、2つのLED（LED1, LED2）の点灯を制御します。詳細については、次の状態遷移図に示します。



3.4 フロー・チャート

このサンプル・プログラムのフロー・チャートを次に示します。



3.5 V850/JJ3とV850/JG3の違い

V850ES/JJ3は、V850ES/JG3に対して、I/O、タイマ/カウンタ、シリアル・インタフェースなどの機能を拡張したものです。

このサンプル・プログラムにおいては、I/Oの初期化におけるポートの初期化範囲が異なります。

サンプル・プログラムの詳細については、付録A プログラム・リスト参照してください。

3.6 ROM化について

このサンプル・プログラムでは、内蔵周辺初期化後に「ROM化」情報のコピー処理を行っています。

「ROM化」情報とは、初期値のある変数（初期値のある変数が配置されるセクション）の初期値情報のことで、「ROM化」情報をRAMにコピーすることにより、初期値のある変数（初期値のある変数が配置されるセクション）は、初めて初期値を持つこととなります^注。

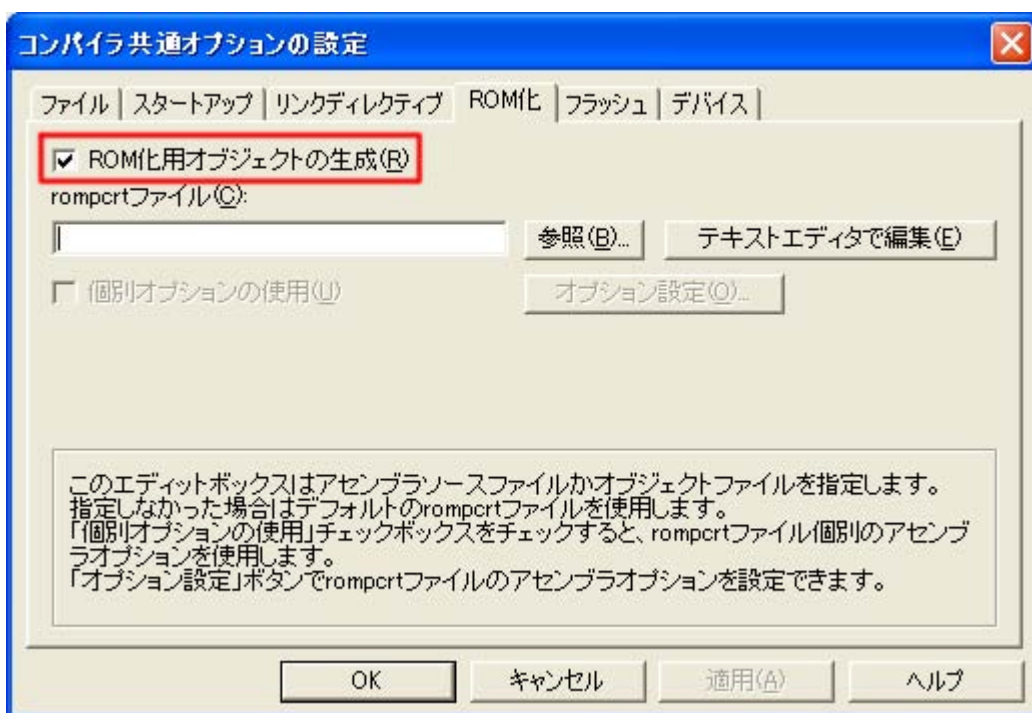
作成するプログラムで、初期値あり変数を使用している場合は、必ず「ROM化」情報の生成とコピー処理を行う必要があり、コピー処理の実施は、必ずその初期値あり変数を利用する前に行う必要があります。

注 ROM化により、パッキングされる対象となるものは、デフォルトでは「書き込み可能な属性を持つセクションに割り当てられたデータ」です。

その他のデータをパッキングすることも可能です。詳細はCA850のヘルプを参照してください。

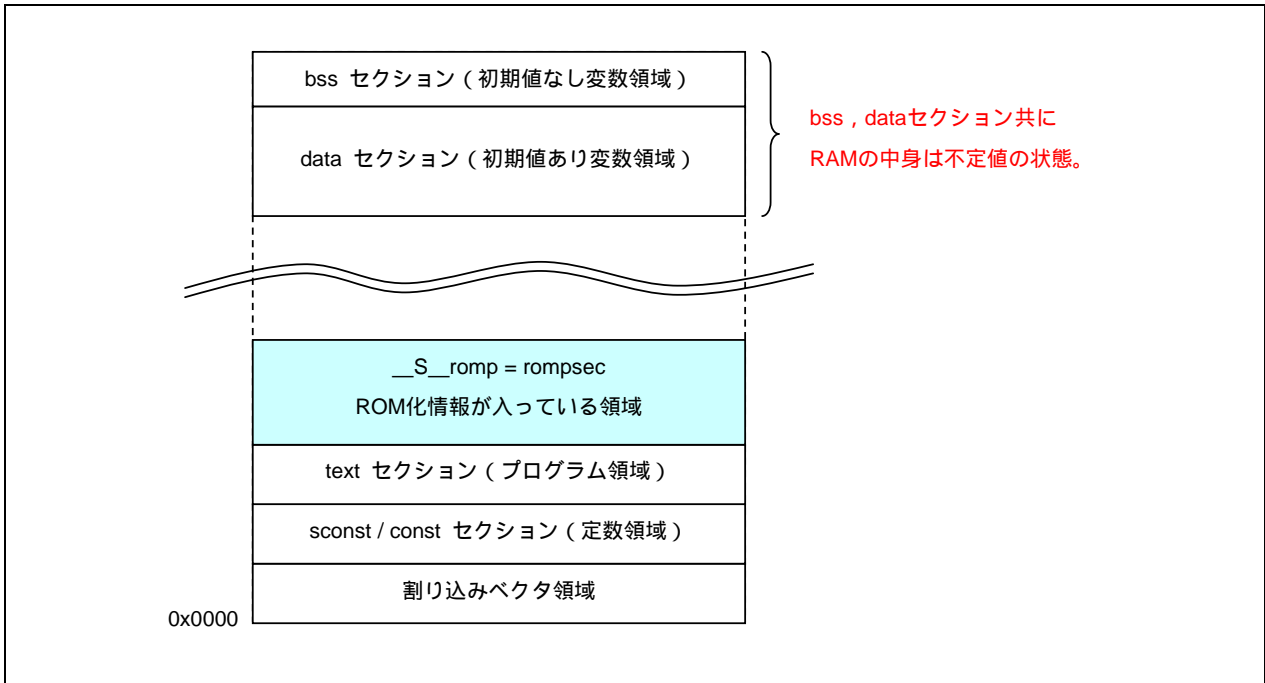
次に、ROM化を行うための手順を説明します。

まず、PM+のコンパイラ共通オプションの〔ROM化〕のタブを選択し、〔ROM化用オブジェクトの生成(R)〕をチェックします。



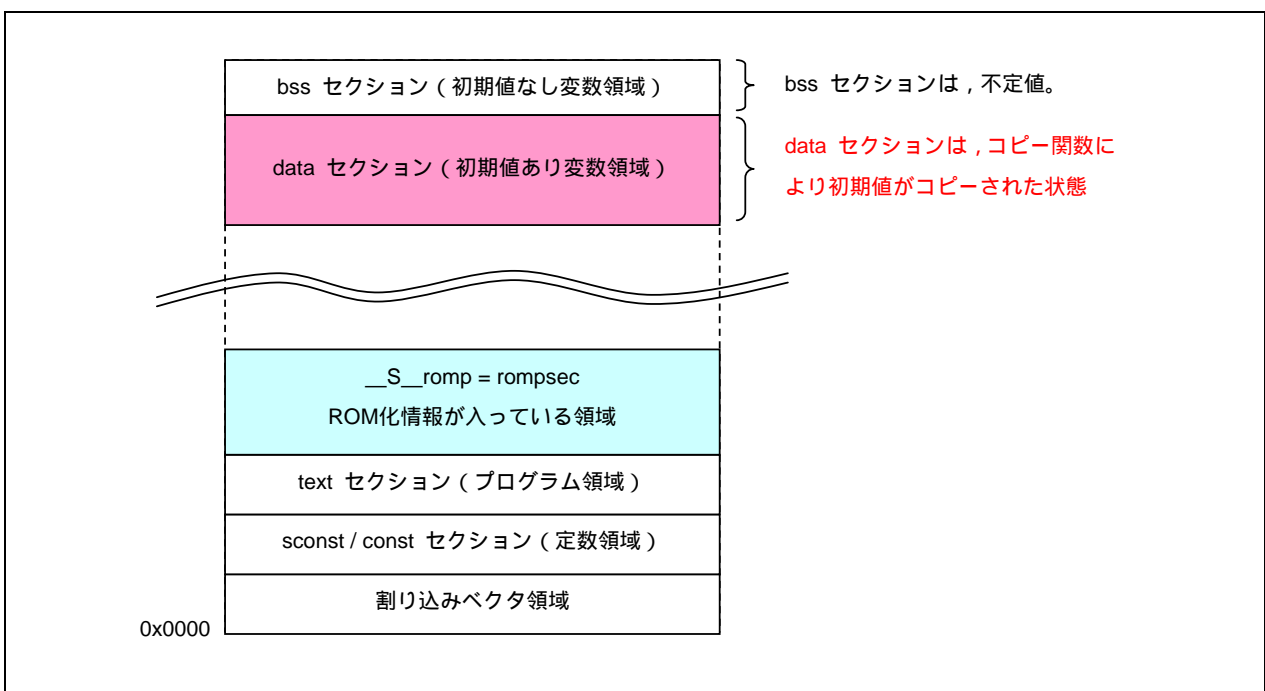
ROM化情報を格納するセクション (rompsec) は、自動でプログラム領域 (.text) セクションの直後に追加されますが、このチェックボックス操作によって、デフォルトの「rompct.o」で定義されたラベル "_S_romp" に "rompsec" と同じアドレスを指すコードが生成され、コピー関数が格納されているライブラリ "lib.a" が自動的にリンクされます。

ここまでの手順で作成された、コピー前のメモリ・イメージは次のようになります。



このままでは、初期値あり変数領域である「data セクション」の中身が不定のままなので、コピー処理を行う必要があります。

ROM化情報のコピーを行うために、_rcopy()関数を呼び出すと、コピー後のメモリ・イメージは次のようになります。



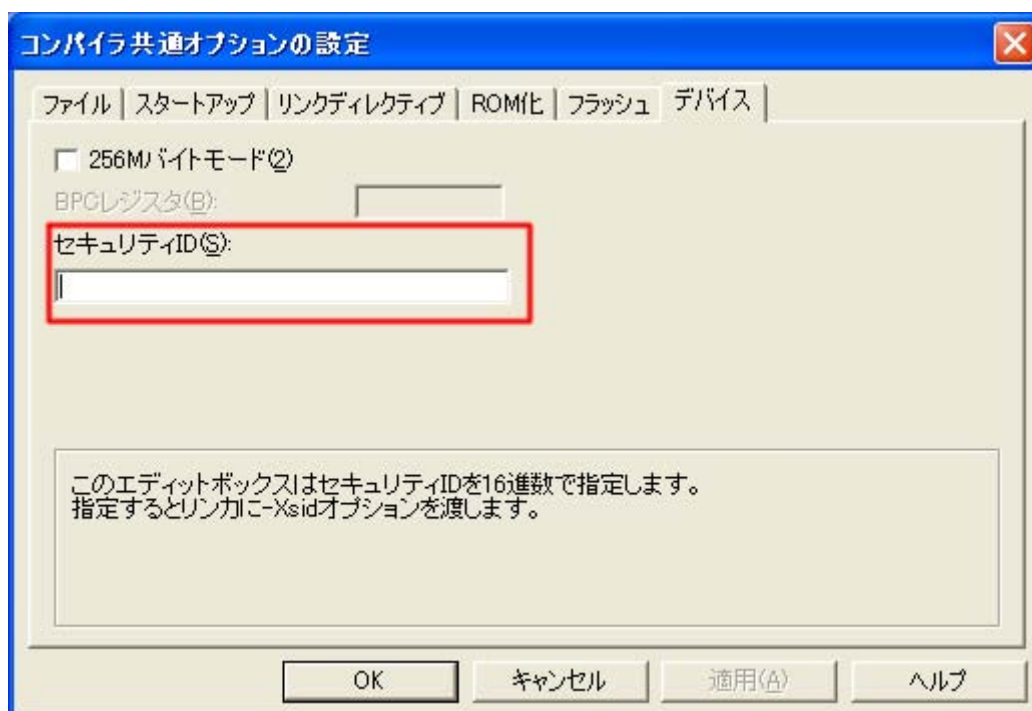
3.7 セキュリティIDについて

オンチップ・デバッグ・エミュレータによるオンチップ・デバッグ時、フラッシュ・メモリの内容を第三者に読み出されることを防ぐために、10バイトのIDコードによる認証を行います。

IDコードは、あらかじめ内蔵フラッシュ・メモリ領域の0x0000070-0x0000079の10バイト分に設定し、デバuggがID認証を行います。

このID照合が一致していれば、セキュリティが解除され、フラッシュ・メモリ読み出し許可、オンチップ・デバッグ・エミュレータ使用許可となります。

このサンプル・プログラム（環境一式版）では、セキュリティIDの設定を行っていないため、デフォルトのセキュリティID値 “0xFFFF FFFF FFFF FFFF FFFF” が適用されるようになっています。

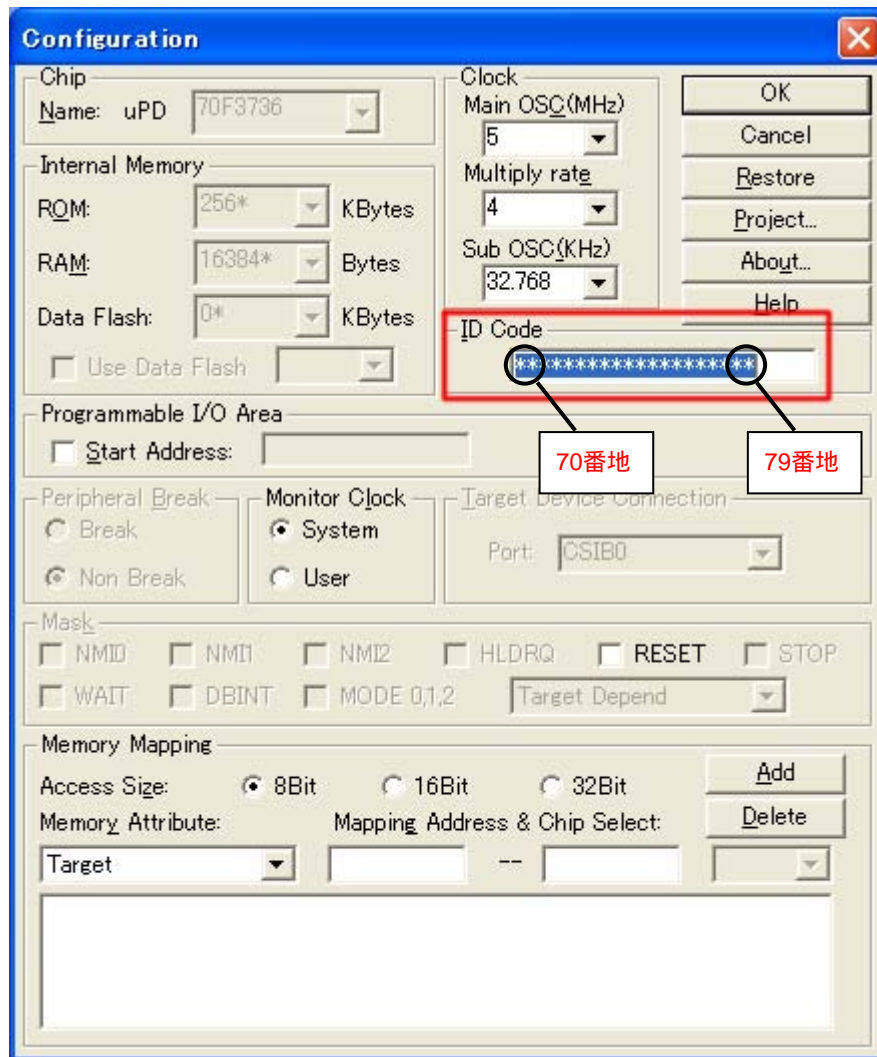


備考 コンパイラ共通オプションの設定「セキュリティID」では、フラッシュ・メモリ搭載デバイスの“セキュリティID”を設定します。

IDは、0xで始まる10バイト以内の16進数で指定します。

このオプションの指定、またはアセンブラ記述（.section "SECURITY_ID" 使用）によるセキュリティIDの指定が省略された場合、“0xFFFF FFFF FFFF FFFF FFFF” が指定されたものとして扱います。

このサンプル・プログラム（環境一式版）を使用して、プログラムをダウンロードして動作させた場合、マイコンのセキュリティID領域には、“0xFF” が設定されるため、次回デバugg接続時のID Code入力エリアに“FFFF FFFF FFFF FFFF”を設定（設定していない場合のデフォルト値）しないと、オンチップ・デバッグ・エミュレータが使用できないため、注意が必要です。



- ・ 10バイトのIDコードのうち、0x0000079のビット7は、オンチップ・デバッグ・エミュレータ使用許可フラグです（0：使用禁止，1：使用許可）。
- ・ オンチップ・デバッグ・エミュレータを起動すると、デバッガがID入力を要求します。デバッガ上で入力したIDコードと、0x0000070-0x0000079に埋め込んだIDコードが一致すればデバッガが起動します。
- ・ IDコードが一致しても、オンチップ・デバッグ・エミュレータ使用許可フラグが“0”である場合は、デバッグを行うことはできません。

3. 8 MINICUBE2を用いたオンチップ・デバッグについて

このサンプル・プログラムでは、CSIB0端子 (SIB0、SOB0、 $\overline{\text{SCKB0}}$ 、HS (PCM0)) をデバッグ通信用インタフェースとしMINICUBE2を用いてオンチップ・デバッグを行うための設定をしています。この設定は、ユーザ・プログラムやコンパイラ・オプションで設定する必要があります。

設定方法の詳細については、次のユーザーズ・マニュアルを参照してください。

- ・ V850ES/JJ3 32ビット・シングルチップ・マイクロコントローラ
ハードウェア編 ユーザーズ・マニュアル
- ・ V850ES/JG3 32ビット・シングルチップ・マイクロコントローラ ハードウェア編
ハードウェア編 ユーザーズ・マニュアル
- ・ QB-MINI2 プログラミング機能付きオンチップ・デバッグ・エミュレータ ユーザーズ・マニュアル

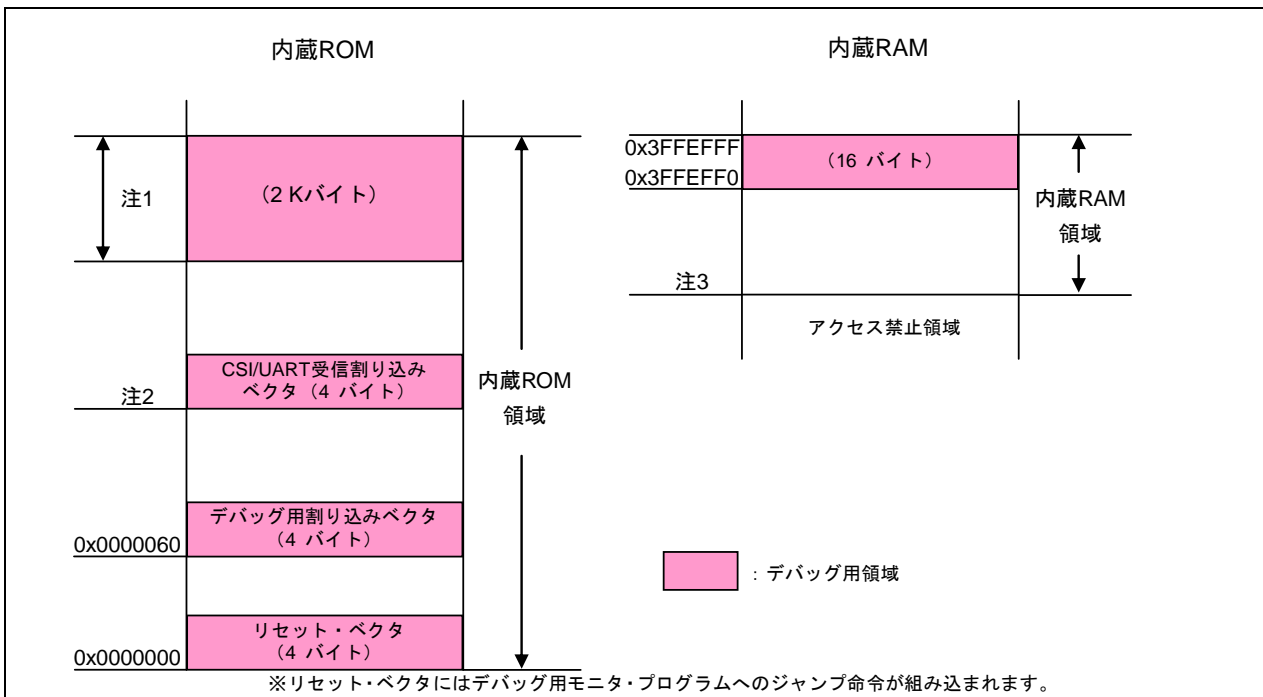
3. 8. 1 デバッグ用モニタ・プログラム領域の確保

図3 - 1に示すデバッグ用領域は、デバッグ用モニタ・プログラムが配置される空間です。モニタ・プログラムはデバッグ用の通信インタフェース (CSIB0、CSIB3、UARTA0) の初期化処理や、CPUのRUN / ブレーク処理などを行うためのものです。内蔵ROM領域については0xFFでフィルし、あらかじめ領域を確保する必要があります。

NECエレクトロニクス社製コンパイラCA850を使用している場合に領域の確保を行うには、アセンブル・ソースとリンクディレクティブ・コードを追加する必要があります。

備考 デバッグ用モニタ・プログラム領域をユーザ・プログラムで使用しない場合は、必ずしも領域を確保する必要はありません。しかし、デバッガ起動時のトラブルを回避するために、あらかじめ領域を確保しておくことを推奨いたします。

図3 - 1 デバッグ用モニタ・プログラムが配置されるメモリ空間



注1 製品によって次のようにアドレス値が異なります。

	内蔵ROMサイズ	アドレス値
μ PD70F3739、μ PD70F3743	384 Kバイト	0x005F800 - 0x005FFFF
μ PD70F3740、μ PD70F3744	512 Kバイト	0x007F800 - 0x007FFFF
μ PD70F3741、μ PD70F3745	768 Kバイト	0x00BF800 - 0x00BFFFF
μ PD70F3742、μ PD70F3746	1024 Kバイト	0x00FF800 - 0x00FFFFFF

注2 使用するシリアル通信に応じて異なります。**CSIB0**使用時は"0x0000290"、**CSIB3**使用時は"0x00002F0H"、**UARTA0**使用時は"0x0000310"から始まります。

注3 製品によって次のようにアドレス値が異なります。

	内蔵RAMサイズ	アドレス値
μ PD70F3739、μ PD70F3743	32 Kバイト	0x3FF7000
μ PD70F3740、μ PD70F3744	40 Kバイト	0x3FF5000
μ PD70F3741、μ PD70F3745	60 Kバイト	0x3FF0000
μ PD70F3742、μ PD70F3746		

備考 赤字部がサンプル・プログラムで使用した製品、もしくはシリアル通信を示しています。

・領域確保の方法

次にサンプル・プログラム（CA850を使用している場合）の領域確保を行う例を示します。次に示すように、アセンブル・ソースとリンクディレクティブ・コードを追加しています。

(a) アセンブル・ソース（次の内容をアセンブル・ソース・ファイルとして追加してください）

```

--monitorROMセクションとして2Kバイトの空間を確保
.section "MonitorROM", const
.space 0x800, 0xff

--デバッグ用割り込みベクタの確保
.section "DBG0"
.space 4, 0xff

--シリアル通信用受信割り込みベクタの確保
--セクション名は使用するシリアル通信に応じて変更してください
--サンプルプログラムではCSIB0に設定しています
.section "INTCB0R"
.space 4, 0xff

--MonitorRAMセクションとして16バイトの空間を確保
.section "MonitorRAM", bss
.lcomm monitorramsym, 16, 4

```

(b) リンク・ディレクティブ（次の内容をリンク・ディレクティブ・ファイルの内容に追加してください）

次の例は、内蔵ROMが1024Kバイト（最終アドレス：0x00FFFFFF）、内蔵RAMが60Kバイト（最終アドレス：0x3FFEFF）の場合です。

```

MROMSEG : !LOAD ?R V0x0ff800{
    MonitorROM    = $PROGBITS ?A MonitorROM;
};
MRAMSEG : !LOAD ?RW V0x03ffeff0{
    MonitorRAM    = $NOBITS ?AW MonitorRAM;
};

```

3.8.2 シリアル・インタフェースの割り込み許可命令 (EI)

MINICUBE2ではCSIB0などのシリアル通信をデバッグ通信用インタフェースとして使用するため、シリアル・インタフェースの割り込みがマスクされている場合には、強制ブレークなどのデバッグ機能が使用できません。従って、サンプル・プログラムでは次に示すように割り込み許可命令 (EI) を実行しています。

・プログラム例

```
/*-----*/
/* MINICUBE2 OCD用割り込み許可処理          */
/*-----*/
__EI();                                     /* 割り込み許可 */
```

3.9 #pragma指令

NECエレクトロニクス社製コンパイラCA850では、次のような#pragma指令が指定できます。

・アセンブラ命令の記述

```
#pragma asm
アセンブラ命令
#pragma endasm
```

C言語中に、アセンブラ命令を記述することができます。

・周辺I/Oレジスタ名の有効化指定

```
#pragma ioreg
```

周辺I/Oレジスタ名を用いて、デバイスの持つ周辺I/Oレジスタにアクセスします。周辺I/Oレジスタ名をそのまま用いてプログラミングする場合はこの#pragma指令を指定する必要があります。

プログラム例

```
#pragma ioreg                               /* 周辺I/Oレジスタ名有効化指定 */
:
:
P0 = 0b00000000;                             /* レジスタ名 P0 を使用可能 */
```

詳細については、次のユーザーズ・マニュアルを参照してください。

- ・CA850 Ver.3.20 Cコンパイラ・パッケージ C言語編

第4章 レジスタ設定について

この章では、システム・ウエイト・コントロール・レジスタ，オンチップ・デバッグ・モード・レジスタ，ウォッチドッグ・タイマ2，DMA動作の禁止，内部システム・クロック，PLLモード，端子機能の設定，およびメイン処理について説明します。

なお、リセット解除後に動作停止状態の周辺機能について、このサンプル・プログラムで使用しない周辺機能は、設定していません。

レジスタ設定方法の詳細については、次のユーザーズ・マニュアルを参照してください。

- ・ V850ES/JJ3 32ビット・シングルチップ・マイクロコントローラ
ハードウェア編 ユーザーズ・マニュアル
- ・ V850ES/JG3 32ビット・シングルチップ・マイクロコントローラ ハードウェア編
ハードウェア編 ユーザーズ・マニュアル

C言語の拡張記述の詳細については、次のユーザーズ・マニュアルを参照してください。

- ・ CA850 Cコンパイラ・パッケージ C言語編 ユーザーズ・マニュアル

4.1 システム・ウェイト・コントロール・レジスタ (VSWC) 設定

VSWCレジスタは、内蔵周辺I/Oレジスタに対するバス・アクセスのウェイトを制御するレジスタです。

内蔵周辺I/Oレジスタへのアクセスは3クロック（ノー・ウェイト時）ですが、V850ES/Jx3では動作周波数によりウェイトが必要です。使用する動作周波数に応じて、VSWCレジスタには次に示す値を設定してください。

8ビット単位でリード/ライト可能です

リセットにより0x77になります。

図4-1 VSWCレジスタのフォーマット

システム・ウェイト・コントロール・レジスタ (VSWC)
 アドレス : 0xFFFFF06E

7	6	5	4	3	2	1	0

動作周波数 (f _{CLK})	VSWC の設定値	ウェイト数
32 kHz f _{CLK} < 16.6 MHz	0x00	0 (ノー・ウェイト)
16.6 MHz f _{CLK} < 25 MHz	0x01	1
25 MHz f _{CLK} 32 MHz	0x11	2

備考 表の赤字部分がサンプル・プログラムでの設定値となります。

VSWCの設定値は「**0x11**」となります。

・プログラム例

```
VSWC = 0b00010001;            /* 内蔵周辺I/Oレジスタへのアクセス時に2ウェイトかける */
```

4.2 特定レジスタの設定

初期設定では、オンチップ・デバッグ・モード・レジスタ (OCDM)、プロセッサ・クロック・コントロール・レジスタ (PCC) の設定を行います。このレジスタは特定レジスタであり、書き込みを行う場合には特定のシーケンスで行う必要があります。

4.2.1 特定レジスタについて

特定レジスタは、プログラムの暴走などにより不正なデータが書き込まれないよう保護されているレジスタです。V850ES/Jx3には次の8個の特定レジスタがあります。

- ・パワー・セーブ・コントロール・レジスタ (PSC)
- ・クロック・コントロール・レジスタ (CKC)
- ・プロセッサ・クロック・コントロール・レジスタ (PCC)
- ・クロック・モニタ・モード・レジスタ (CLM)
- ・リセット要因フラグ・レジスタ (RESF)
- ・低電圧検出レジスタ (LVIM)
- ・内蔵RAMデータ・ステータス・レジスタ (RAMS)
- ・オンチップ・デバッグ・モード・レジスタ (OCDM)

また、プログラムの暴走により応用システムが不用意に停止しないように、特定レジスタへの書き込み動作に対するプロテクション・レジスタとしてPRCMDレジスタがあり、特定レジスタへのライト・アクセスは特定のシーケンスで行われ、不正なストア動作はシステム・ステータス・レジスタ (SYS) に報告されます。

4.2.2 特定レジスタへのデータ設定

特定レジスタへのデータ設定は次のシーケンスで行います。

DMA動作を禁止する。

任意の汎用レジスタに特定レジスタへ設定するためのデータを用意する。

PRCMDレジスタに で用意したデータを書き込む。

特定レジスタに設定データを書き込む (次の命令で行う)。

- ・ストア命令 (ST/SST命令)
 - ・ビット操作命令 (SET1/CLR1/NOT1命令)
 - (- NOP命令を挿入する (5命令)) (PSC.STPビット=1にする場合のみ)
- DMA動作が必要な場合、DMA動作を許可する。

備考 リセット解除後、DMAチャンネル・コントロール・レジスタの初期値は0x00であり、DMA動作は禁止の状態になっています。初期設定時など、DMA動作が禁止されていることが明確な場合は、 と の手順は省略できます。サンプル・プログラムでは と の手順は含みません。

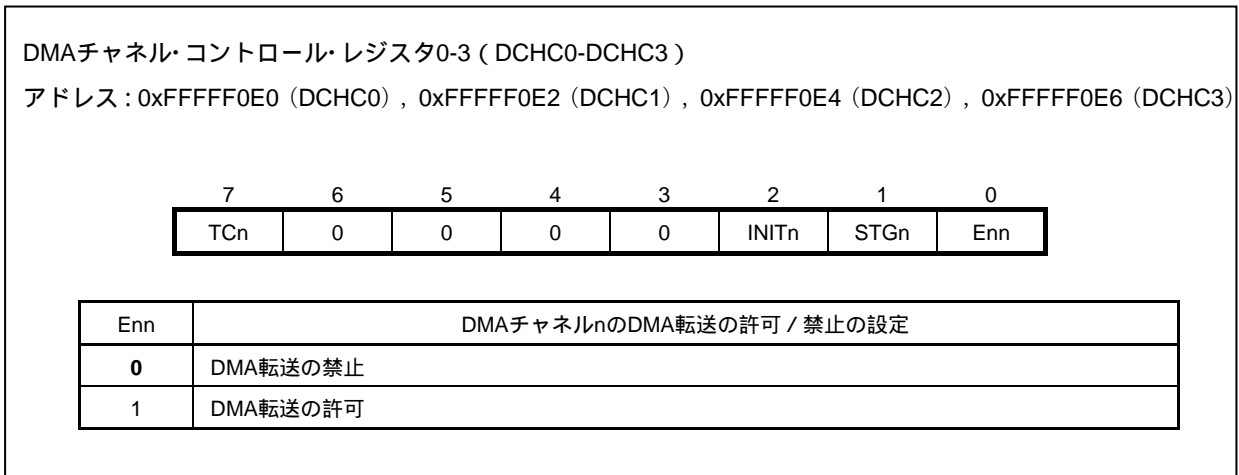
4.2.3 DMA動作を禁止する

特定レジスタヘデータ設定を書き込むために、まずDMA動作を禁止する必要があります。

DMAチャンネル・コントロール・レジスタ0-3 (DCHC0-DCHC3) でDMAチャンネル0-3のDMA転送の許可 / 禁止を設定できます。

DMAチャンネルnのDMA転送の許可 / 禁止を設定するために、DCHC.Ennビット (ビット0) を設定してください。

図4 - 2 DCHCnレジスタのフォーマット



リセット解除後、DMA転送は禁止 (DCHCn.Ennビット=0) 状態のためDMAの停止処理は省略できます。

4.3 オンチップ・デバッグ・モード・レジスタの通常動作モード設定

通常動作モードとオンチップ・デバッグ・モードを切り替えるレジスタで、オンチップ・デバッグ機能が割り付けられている兼用端子をオンチップ・デバッグ用端子として使用するか、通常のポート/周辺機能兼用端子として使用するかを指定します。また同時に、P05/INTP2/ $\overline{\text{DRST}}$ 端子の内蔵プルダウン抵抗の切断を制御します。

OCDMレジスタは特定レジスタです。特定のシーケンスの組み合わせによってだけ書き込みができます(4.2.2 特定レジスタへのデータ設定参照)。

OCDMレジスタへの書き込みは、 $\overline{\text{DRST}}$ 端子にロウ・レベルが入力されているときのみ有効です。

8/1ビット単位でリード/ライト可能です。

RESET端子入力により0x01になります。ウォッチドッグ・タイマ、クロック・モニタ、低電圧検出回路によるリセットの場合は、OCDMレジスタの値を保持します。

図4-3 OCDMレジスタのフォーマット

オンチップ・デバッグ・モード・レジスタ (OCDM)
 アドレス : 0xFFFFF9FC

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OCDM0

OCDM0	動作モード
0	【MINICUBE2 (QB-MINI2) を使用する場合、通常動作モードで使用する場合】 通常動作モード (オンチップ・デバッグ兼用端子をポート/周辺機能端子として使用) かつ、 P05/INTP2/ $\overline{\text{DRST}}$ 端子の内蔵プルダウン抵抗を切断
1	【MINICUBE (QB-V850MINI) を使用する場合】 $\overline{\text{DRST}}$ 端子がロウ・レベルの場合 : 通常動作モード (オンチップ・デバッグ兼用端子をポート/周辺機能端子として使用) $\overline{\text{DRST}}$ 端子がハイ・レベルの場合 : オンチップ・デバッグ・モード (オンチップ・デバッグ・モード用端子として使用)

備考 表の赤字部分がサンプル・プログラムでの設定値となります。

OCDM特定レジスタの設定データは「0x00」です。

・ プログラム例

```
/* OCDMを「通常動作モード」に指定 */  
  
#pragma asm  
    st.b r0, PRCMD  
    st.b r0, OCDM  
#pragma endasm
```

4.4 内蔵発振モード・レジスタ (RCM) 設定

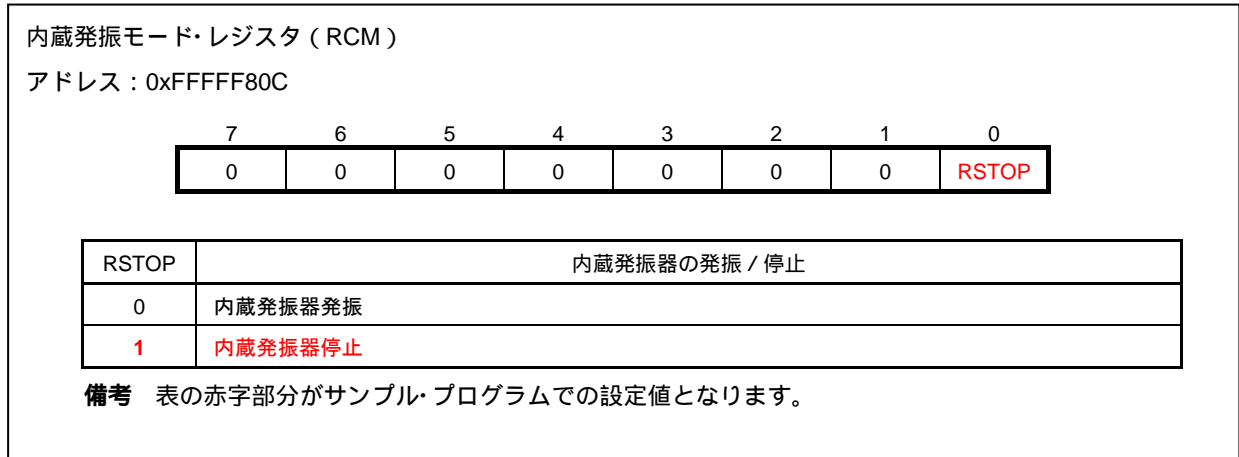
RCMレジスタは、内蔵発振器の動作モードの設定を行う8ビット・レジスタです。

このサンプル・プログラムでは、ウォッチドッグ・タイマを使用しないため、内蔵発振器を停止します。

8/1ビット単位でリード/ライト可能です。

リセットにより0x00になります。

図4-4 RCMレジスタのフォーマット



RCMの設定値は「0x01」となります。

・プログラム例

```
RSTOP = 1;      /*内蔵発振器停止設定*/
```

4.5 ウォッチドッグ・タイマ2の設定

WDTM2レジスタは、ウォッチドッグ・タイマ2のオーバフロー時間および動作クロックを設定するレジスタです。

ウォッチドッグ・タイマ2は、リセット解除後に自動的にリセット・モードでスタートします。動作を確定するために、WDTM2レジスタへ書き込みを行ってください。

このサンプル・プログラムでは、ウォッチドッグ・タイマは使用しないため、停止します。

8ビット単位でリード/ライト可能です。

リセットにより0x67になります。

図4-5 WDTM2レジスタのフォーマット

ウォッチドッグ・タイマ・モード・レジスタ2 (WDTM2)
 アドレス : 0xFFFFF6D0

7	6	5	4	3	2	1	0
0	WDM21	WDM20	WDCS24	WDCS23	WDCS22	WDCS21	WDCS20

WDM21	WDM20	ウォッチドッグ・タイマ2の動作モード選択
0	0	動作停止
0	1	ノンマスクابل割り込み要求モード (INTWDT2信号を発生)
1	-	リセット・モード (WDT2RES信号を発生)

備考 表の赤字部分がサンプル・プログラムでの設定値となります。

WDTM2の設定値は「0x00」となります。

・プログラム例

```
WDTM2 = 0b00000000; /* ウォッチドッグ・タイマ2動作停止設定 */
```

4.6 クロックの設定

このサンプル・プログラムでは、4 MHzのセラミック発振子/水晶振動子をX1, X2端子に接続し、PLLモードで8逓倍して、32 MHzを内部システム・クロックにする例を示しています。また、サブクロックは使用しません。

4.6.1 プロセッサ・クロック・コントロール・レジスタ (PCC) の設定

PCCレジスタは、サブクロックとメイン・クロックの内蔵帰還抵抗の選択、メイン・クロック発振回路の制御、内部システム・クロックの選択ができます。

PCCレジスタは特定レジスタです。特定のシーケンスの組み合わせによってだけ書き込みができます。

8/1ビット単位でリード/ライト可能です。

リセットにより0x03になります。

図4 - 6 PCCレジスタのフォーマット

プロセッサ・クロック・コントロール・レジスタ (PCC)
 アドレス : 0xFFFFF828

	7	6	5	4 ^注	3	2	1	0
	FRC	MCK	MFRC	CLS	CK3	CK2	CK1	CK0

FRC	サブクロックの内蔵帰還抵抗の選択
0	使用する (サブクロックを接続する)
1	使用しない (サブクロックを接続しない)

MFRC	メイン・クロック内蔵帰還抵抗の選択
0	使用する (セラミック発振子/水晶振動子使用時)
1	使用しない (外部クロック使用時)

CK3	CK2	CK1	CK0	クロックの選択 (f _{CLK} /f _{CPU})
0	0	0	0	f _{xx}
0	0	0	1	f _{xx} /2
0	0	1	0	f _{xx} /4
0	0	1	1	f _{xx} /8
0	1	0	0	f _{xx} /16
0	1	0	1	f _{xx} /32
0	1	1	X	設定禁止
1	X	X	X	f _{XT}

備考 表の赤字部分がサンプル・プログラムでの設定値となります。

注意 CLSビットはリードのみ可能です

PCCの設定値は「0x80」となります。

・ プログラム例

```
/* クロック分周を「なし」に設定 */  
#pragma asm  
    push r10  
    mov 0x80, r10  
    st.b r10, PRCMD  
    st.b r10, PCC  
    pop r10  
#pragma endasm
```

4.6.2 PLLコントロール・レジスタ (PLLCTL) 設定

PLLコントロール・レジスタでCPU動作クロックを選択します。

PLLを制御する8ビット・レジスタです。8/1ビット単位でリード/ライト可能です。

リセットにより0x01になります。

図4 - 7 PLLCTLレジスタのフォーマット

PLLコントロール・レジスタ (PLLCTL)
 アドレス : 0xFFFFF82C

7	6	5	4	3	2	1	0
0	0	0	0	0	0	SELPLL	PLLON

SELPLL	動作モード
0	クロック・スルー・モード
1	PLLモード

SELPLLビット = 1の設定は、PLLクロック周波数が安定した状態のときのみ可能です。安定していないとき（アンロック中）にSELPLLビットをライトすると“0”がライトされます。

PLLON	PLL動作停止レジスタ
0	PLL停止
1	PLL動作（PLLを動作開始後、周波数が安定するまで所定のロックアップ時間が必要）

4.6.3 ロック・レジスタ (LOCKR)

PLLが安定 (ロック) したことを確認するフラグです。

8/1ビット単位でリードのみ可能です。

リセットにより0x01になり、リセット解除後の発振安定時間経過後に0x00になります。

図4 - 8 LOCKRレジスタのフォーマット

ロック・レジスタ (LOCKR)
アドレス : 0xFFFFF824

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	LOCK

LOCK	動作モード
0	ロック状態
1	アンロック (ロックしていない) 状態

LOCKビットはPLLのロック状態をリアルタイムに反映するものではありません。

備考 リセット解除後、発振安定時間経過後にロック・レジスタはロック状態 (LOCKR = 0x00) になります。

4.6.4 クロック・コントロール・レジスタ (CKC) 設定

CKCレジスタは、PLLモード時の内部システム・クロックを制御するレジスタです。

CKCレジスタは特定レジスタです。特定のシーケンスの組み合わせによってだけ書き込みができます。

8/1ビット単位でリード/ライト可能です。

リセットにより0x0Aになります。

図4 - 9 CKCレジスタのフォーマット

クロック・コントロール・レジスタ (CKC)
アドレス : 0xFFFFF822

7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	CKDIV0

CKDIV0	PLLモード時の内部システム・クロック (fx)
0	$fx = 4 \times fx$ ($fx = 2.5 \sim 5.0$ MHz)
1	$fx = 8 \times fx$ ($fx = 2.5 \sim 4.0$ MHz)

fx = 5.0 ~ 10.0 MHzではPLLモードは使用できません。またCKCレジスタで4通倍 / 8通倍を切り替える場合は、クロック・スルー・モードに設定し、PLLを停止してから通倍率を変更して下さい。

備考 表の赤字部分がサンプル・プログラムでの設定値となります。

4.6.5 使用方法

PLLモードで8通倍とするため、クロック・コントロール・レジスタ (CKC) の設定値を「0x0B」とし、通倍率を切り替えます (CKCレジスタは特定レジスタです。特定のシーケンスの組み合わせによってだけ書き込みができます)。

CKCレジスタで通倍率を切り替える場合は、PLLを停止してから行います。

・PLLを停止させる場合

最初にクロック・スルー・モードとするため、SELPLLビット = 0 とします。

8クロック以上後に、PLLを停止します。PLLの停止はPLLONビット = 0 とします。

備考 初期設定時にはPLLは動作していますが、クロック・スルー・モードのため、SELPLLビット = 0、8クロック経過を省略してPLLの停止 (PLLONビット = 0) を行うことができます。

次にPLLモードとするため、PLLを動作させます。

・PLLを動作させる場合

PLL停止の状態からPLLを動作する場合は、PLLONビット = 1 として、ロック・レジスタ (LOCKR) のLOCKビット = 0 となってからSELPLLビット = 1 としてください。

・プログラム例

```

/* CPUの動作クロックPLLモード : fx = 2.5~4 MHz(fxx = 20~32 MHz)選択 */
/* 通倍率を設定するためPLLを停止 (初期設定時はクロック・スルー・モード) */

PLLON = 0;          /* PLL動作停止 */

                    /* PLLの通倍率を8通倍に設定 */

#pragma asm
  push r10
  mov 0x0B, r10
  st.b r10, PRCMD
  st.b r10, CKC
  pop r10
#pragma endasm

PLLON = 1;          /* PLL動作許可 */

while( LOCK );     /* PLL安定 (ロック) 待ち */

SELPLL = 1;        /* PLLモードに設定 */

```


4.7 ポート設定

各製品により、内蔵するポートは異なるため、設定するポートも異なります。

	V850ES/JG3	V850ES/JJ3
ポート0	P02-P06	P00-P06
ポート1	P10, P11	P10, P11
ポート3	P30-P39	P30-P39
ポート4	P40-P42	P40-P42
ポート5	P50-P55	P50-P55
ポート6	—	P60-P615
ポート7	P70-P711	P70-P715
ポート8	—	P80, P81
ポート9	P90-P915	P90-P915
ポートCD	—	PCD0-PCD3
ポートCM	PCM0-PCM3	PCM0-PCM5
ポートCS	—	PCS0-PCS7
ポートCT	PCT0, PCT1, PCT4, PCT6	PCT0-PCT7
ポートDH	PDH0-PDH5	PDH0-PDH7
ポートDL	PDL0-PDL15	PDL0-PDL15

4.7.1 ポートnレジスタ (Pn)

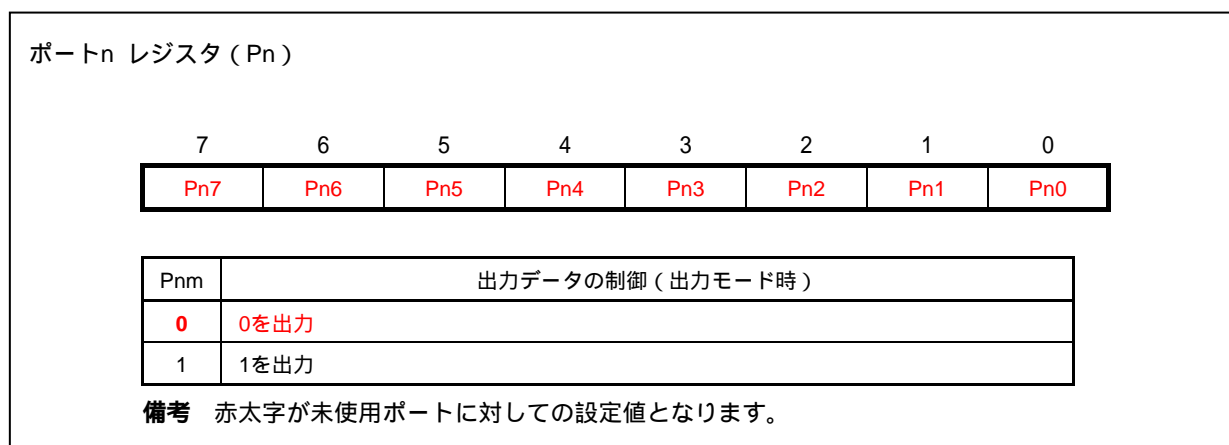
外部とのデータ入出力は、Pnレジスタへの書き込み、および読み出しによって行います。Pnレジスタは、出力データを保持する出力ラッチ、および端子の状態を読み込む回路で構成されています。

Pnレジスタの各ビットは、それぞれポートnの端子1本ずつに対応しており、1ビット単位でリード/ライト可能です。

このサンプル・プログラムでは、ポート0を後述の【例1】、ポートCMを後述の【例2】の内容で設定しています。また、未使用ポート端子は出力ポートに設定しています。

リセットにより0x00になります。

図4 - 10 Pnレジスタのフォーマット



【コラム】未使用端子の処理について

ポート端子はリセットにより入力端子に設定されます。したがって、未使用端子の処理は、個別に抵抗を介してV_{DD}またはGNDに接続することを推奨します。

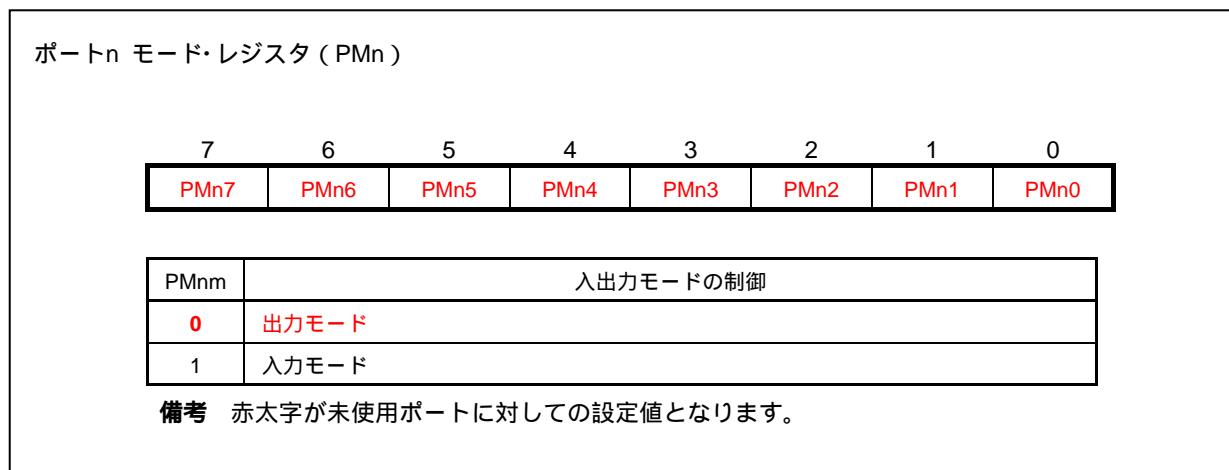
しかし、抵抗の部品点数の削減のために端子をオープンにして出力設定にすることも認めています。

4.7.2 ポートnモード・レジスタ (PMn)

ポートの入力モード/出力モードを指定します。

PMnレジスタの各ビットは、それぞれポートnの端子1本ずつに対応しており、1ビット単位で指定可能です。
リセットにより0xFFになります。

図4 - 11 PMnレジスタのフォーマット



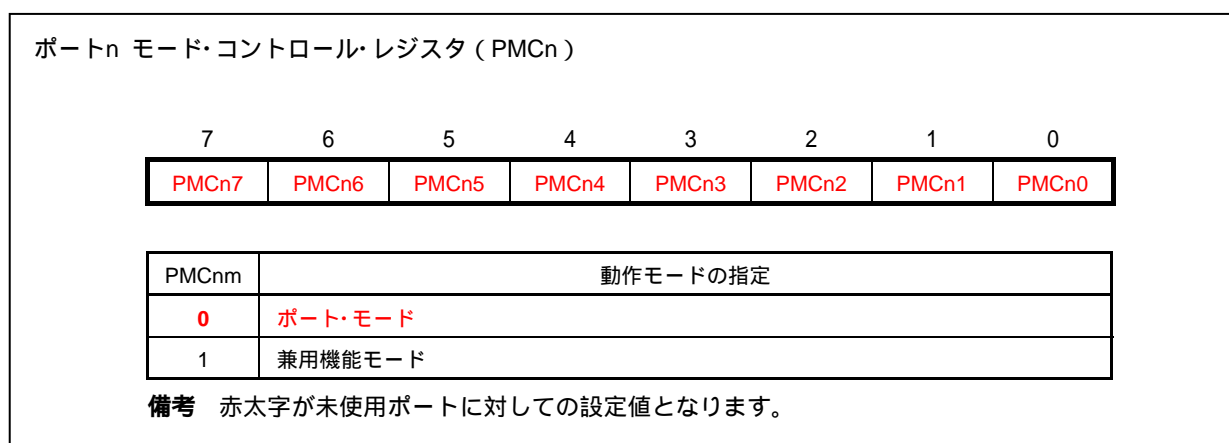
4.7.3 ポートnモード・コントロール・レジスタ

ポート・モード/兼用機能モードを指定します。

PMCnレジスタの各ビットは、それぞれのポートnの端子1本ずつに対応しており、1ビット単位で指定可能です。

リセットにより0x00になります。

図4 - 12 PMCnレジスタのフォーマット



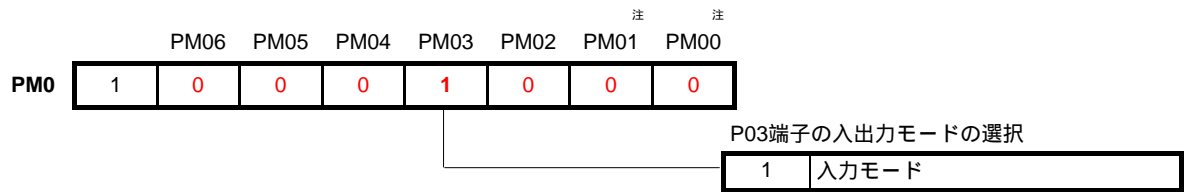
【コラム】 Pnレジスタへの書き込み/読み出しについて

Pnレジスタへの書き込みは、出力ラッチに対しての書き込みとなります。

PMnレジスタ設定によって「入力モード」に指定されている端子については、Pnレジスタへの書き込み値に関わらず、入力端子の状態に影響はありません。

また、出力ラッチに書き込まれた値は、再度出力ラッチに値を書き込まれるまで保持されます。

【例1】・P03を入力ポートに設定（V850ES/JJ3）



備考 ビット名称が記載されていないビット（0, 1, 7 ビット）の値は製品で固定されています

注意 V850ES/JG3ではポートP00, P01がないため、PM01, PM00は1に固定です

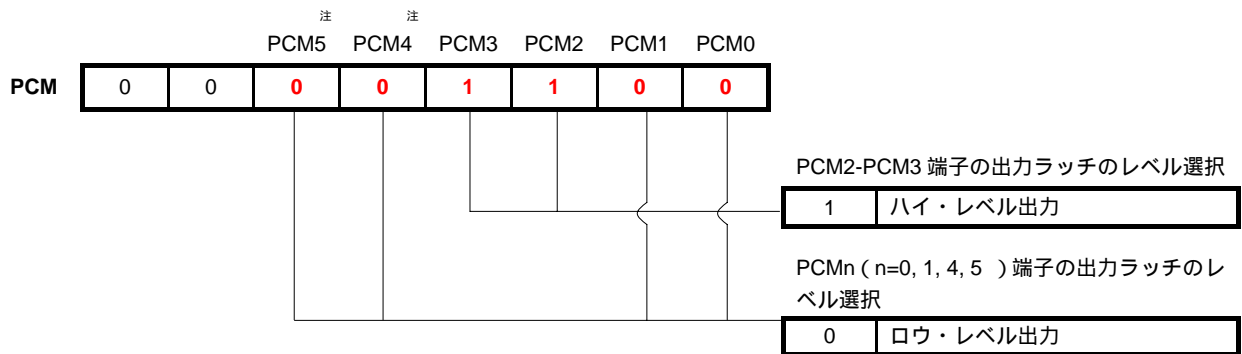
PM0の設定値は、「0x88」となります（V850ES/JG3の場合には「0x8B」となります）。

・プログラム例（V850ES/JJ3）

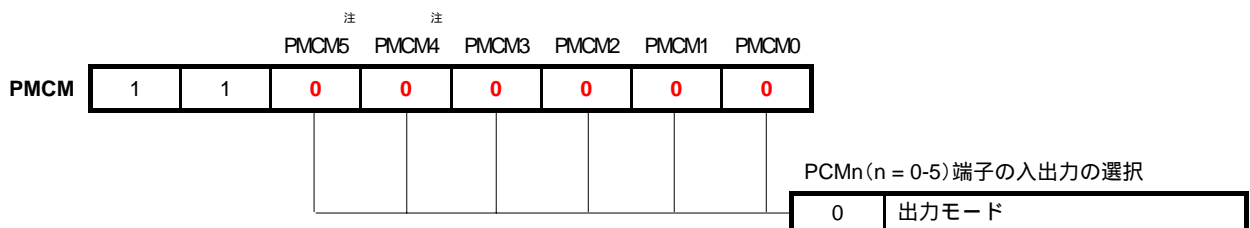
```
PM0 = 0b10001000; /* P00-P06を出力LOWに設定(P03除) */
```

【例2】・PCM2, PCM3の出力ラッチをハイ・レベル出力設定 (V850ES/JJ3)

・PCM2, PCM3を出力ポートに設定 (V850ES/JJ3)



このサンプル・プログラムでは、PCM2, PCM3をLED点灯用の出力ポートとして使用するため、初期設定でPCM2, PCM3の出力ラッチのレベルをハイ・レベル出力にしておきます (LEDは、PCM2-PCM3からの出力がロウ・レベルのときに点灯します (回路図を参照))。



このサンプル・プログラムでは、PCM2-PCM3をLED点灯用の出力ポートとして使用するため、ポートCMを出力ポートに設定します。

備考 ビット名称が記載されていないビット (0, 1, 7 ビット) の値は製品で固定されています

注意 V850ES/JG3ではポートPCM4, PCM5がないため、PCM4, PCM5は0に、PMCM4, PMCM5は1に固定です

PCMの設定値は「0x0C」, PMCMの設定値は「0xC0」となります

(V850ES/JG3では、PCMの設定値は「0x0C」, PMCMの設定値は「0xF0」となります)。

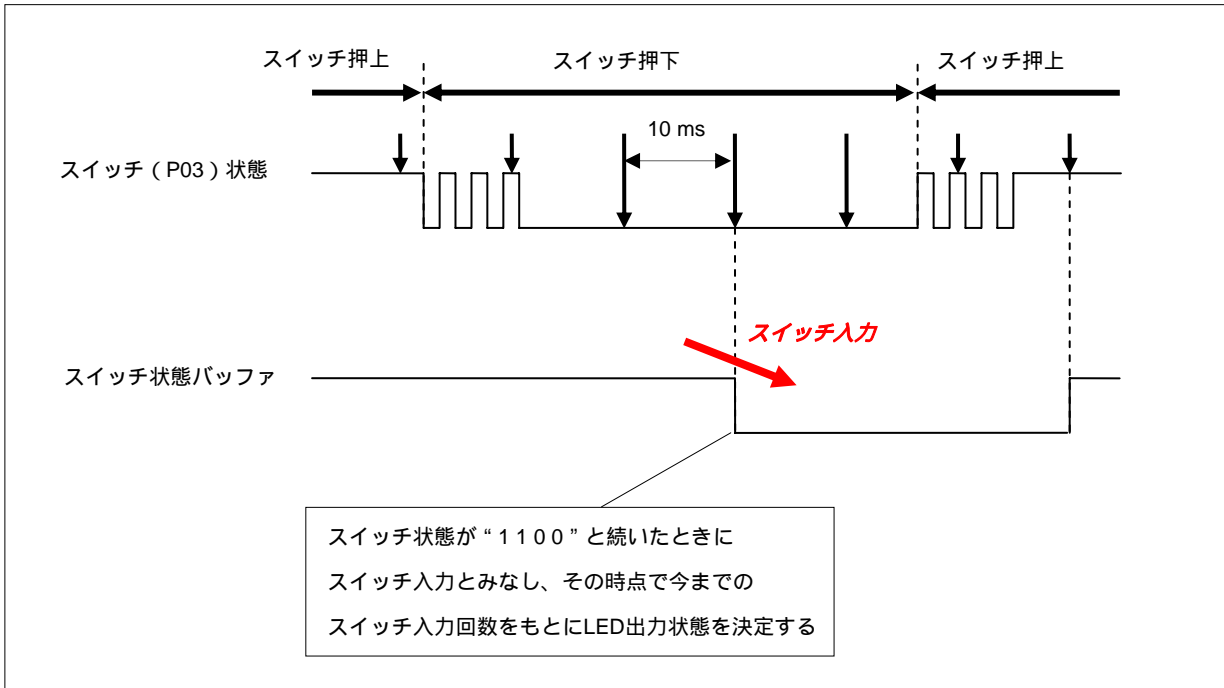
・プログラム例 (V850ES/JJ3)

```
PCM = 0b00001100;          /* PCM2, PCM3の出力ラッチHigh */
PMCM = 0b11000000;        /* PCM0-PCM5を出力ポートに設定 */
```

4.8 メイン処理

4.8.1 チャタリング対策

チャタリングを除去するために、10 msごとにスイッチ（P03）状態値を読み込み、2回連続でスイッチ状態値に同じレベルを検出したときにスイッチ入力とみなします。



10msウェイト動作は次の処理で行います。

・プログラミング例

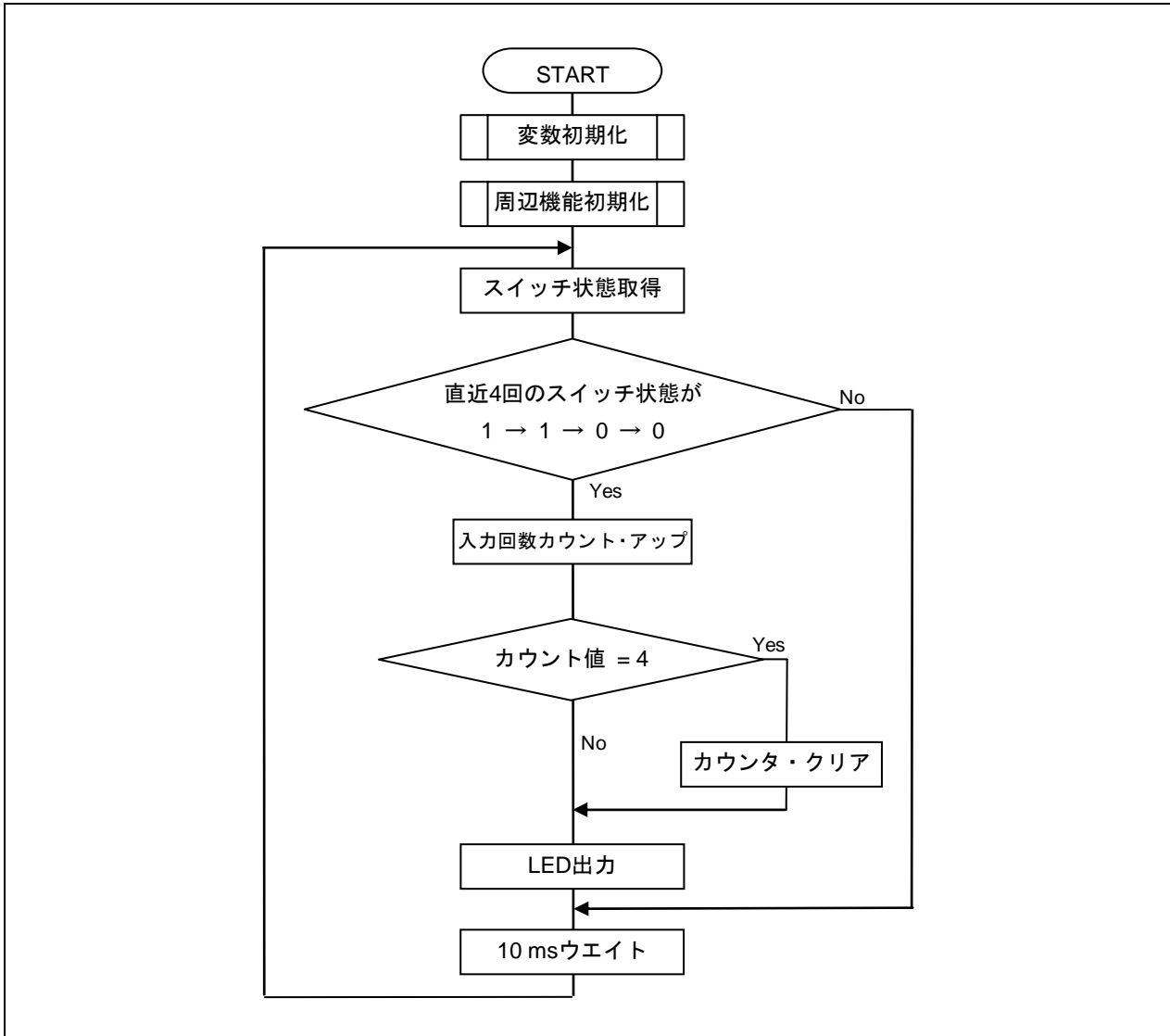
```
unsigned long loop_wait;          /* forループ用カウンタ */

/* 10msウェイト */
for ( loop_wait = 0; loop_wait <= VAL_TIMER_WAIT; loop_wait++ )
{
    __nop();
}
```

4.8.2 メイン処理

C言語のメイン処理は次の動作を行います。

このサンプル・プログラムではスイッチ入力回数と出力データの対応を、配列で設定しています。



```

/*****
メイン処理
*****/
void main( void )
{
    extern unsigned int _S_romp;          /* ROM化シンボルの外部参照      */

    /*-----*/
    /* 変数宣言や変数初期設定      */
    /*-----*/
    const unsigned char outdata[] = {     /* LED表示データパターン用配列  */
        0x0c,                             /* 全LED消灯                      */
        0x04,                             /* LED1点灯                      */
        0x00,                             /* LED1とLED2点灯                */
        0x08                             /* LED2点灯                      */
    };

    unsigned char indata = 0b00000001;    /* スイッチ状態記憶(前回値1で初期化) */
    unsigned char count;                  /* スイッチ入力回数                */
    unsigned long loop_wait;              /* forループ用カウンタ              */

    count = VAL_RST_COUNT;                /* スイッチ入力回数初期化          */

    /*-----*/
    /* 周辺機能初期化              */
    /*-----*/
    f_init_vswc();                        /* VSWCレジスタの設定              */
    f_init_ocdm();                        /* オンチップ・デバッグ・モード・レジスタの設定 */
    f_init_rcm();                          /* 内蔵発振器の禁止設定            */
    f_init_wdtm2();                       /* ウォッチドッグ・タイマ2の設定    */
    f_init_lock();                        /* CPUの動作クロックをPLLモードに設定 */
    f_init_blank_port();                  /* 未使用ポート設定                */
    f_init_use_port();                    /* SW1・LEDポート設定              */

    /*-----*/
    /* ROM化処理                  */
    /*-----*/
    _rcopy( &_S_romp, -1 );                /* ROM化処理実施                    */

    /*-----*/
    /* MINICUBE2 OCD用割り込み許可処理      */
    /*-----*/
    __EI();                                /* 割り込み許可                      */
}

```

{ }内に4個のデータを定義し、
その中に入力データを設定しています。

```

/*-----*/
/* LED点灯処理          */
/*-----*/
while ( 1 )
{
    indata <<= 1;          /* スイッチ状態前回値を更新*/
    indata |= P0.3;       /* スイッチ状態今回値を更新*/
    if ( ( indata & 0b00001111 ) == 0b00001100 )
    {
        count++;          /* スイッチ入力回数更新 */
        count &= 0b00000011
        PCM = outdata[count];          /* テーブルからLED表示データを読み出して表示 */;
    }

    /* 10msウエイト */
    for ( loop_wait = 0; loop_wait <= VAL_TIMER_WAIT; loop_wait++ )
    {
        __nop();
    }
}

return;
}

```

スイッチ入力回数と出力データの対応は次のようになります。

スイッチ入力回数	COUNT	OUTDATA	LED点灯
0回	0	0b00001100	全LED消灯
1回	1	0b00000100	LED1のみ点灯
2回	2	0b00000000	LED1/2点灯
3回	3	0b00001000	LED2のみ点灯

第5章 関連資料

資料名	資料番号
V850ES/JJ3 ユーザーズ・マニュアル ハードウェア編	U18376J
V850ES/JG3 ユーザーズ・マニュアル ハードウェア編	U18708J
V850ES アーキテクチャ編	U15943J
PM+ Ver.6.30 ユーザーズ・マニュアル	U18416J
CA850 Ver.3.20 Cコンパイラ・パッケージ 操作編	U18512J
CA850 Ver.3.20 Cコンパイラ・パッケージ C言語編	U18513J
CA850 Ver.3.20 Cコンパイラ・パッケージ リンク・ディレクティブ編	U18515J
QB-MINI2 プログラミング機能付きオンチップ・デバッグ・エミュレータ	U18371J
ID850QB Ver.3.40 統合デバッガ 操作編	U18604J

ドキュメント検索URL <http://www.necel.com/micro/ja/documentation.html>

付録A プログラム・リスト

プログラム・リスト例として、V850ES/JJ3 マイクロコントローラのソース・プログラムを次に示します。

```
• minicube2.s
#-----
#
#   NEC Electronics   V850ES/Jx3 シリーズ
#
#-----
#   V850ES/JJ3  JG3   サンプル・プログラム
#-----
#   LED点灯のスイッチ制御
#-----
# 【履歴】
#   2009.5.--   新規作成
#-----
# 【概要】
#   本サンプル・プログラムは、MINICUBE2使用時に必要なリソースの確保を行っている
#   (CSIB0を用いてMINICUBE2を使用する場合の例)
#-----

--monitorROMセクションとして2Kバイトの空間を確保
.section "MonitorROM", const
.space 0x800, 0xff

--デバッグ用割り込みベクタの確保
.section "DBG0"
.space 4, 0xff

--シリアル通信用受信割り込みベクタの確保
.section "INTCB0R"
.space 4, 0xff

--MonitorRAMセクションとして16バイトの空間を確保
.section "MonitorRAM", bss
.lcomm monitorramsym, 16, 4
```

セクション名は使用するシリアル通信に応じて変更してください

CSIB3 のとき "INTCB3R" 、
UARTA0 のとき "INTUA0R"

```

• AppNote_LED.dir
# Sample link directive file (not use RTOS/use internal memory only)
#
# Copyright (C) NEC Electronics Corporation 2002
# All rights reserved by NEC Electronics Corporation.
#
# This is a sample file.
# NEC Electronics assumes no responsibility for any losses incurred by customers or
# third parties arising from the use of this file.
#
# Generated      : PM+ V6.31 [ 9 Jul 2007]
# Sample Version : E1.00b [12 Jun 2002]
# Device         : uPD70F3746 (c:¥program files¥nec electronics tools¥dev¥DF3746.800)
# Internal RAM   : 0x3ff0000 - 0x3ffefff
#
# NOTICE:
#     Allocation of SCONST, CONST and TEXT depends on the user program.
#
#     If interrupt handler(s) are specified in the user program then
#     the interrupt handler(s) are allocated from address 0 and
#     SCONST, CONST and TEXT are allocated after the interrupt handler(s).

SCONST : !LOAD ?R {
    .sconst      = $PROGBITS      ?A .sconst;
};

CONST   : !LOAD ?R {
    .const       = $PROGBITS      ?A .const;
};

TEXT    : !LOAD ?RX {
    .pro_emi_runtime = $PROGBITS    ?AX .pro_emi_runtime;
    .text          = $PROGBITS    ?AX .text;
};

```

```

### MINICUBE2用 ###
MROMSEG : !LOAD ?R 0x0ff800 {
    MonitorROM    = $PROGBITS ?A MonitorROM;
};

```

製品の内蔵ROMサイズによりアドレスが異なります
(例は内蔵ROMが1024 Kバイトの場合)

デフォルトのリンク・ディレクティブ
ファイルに追加するコード
MINICUBE2用の予約領域を確保

```

SIDATA : !LOAD ?RW V0x3ff0000 {
    .tidata.byte = $PROGBITS ?AW .tidata.byte;
    .tibss.byte = $NOBITS ?AW .tibss.byte;
    .tidata.word = $PROGBITS ?AW .tidata.word;
    .tibss.word = $NOBITS ?AW .tibss.word;
    .tidata = $PROGBITS ?AW .tidata;
    .tibss = $NOBITS ?AW .tibss;
    .sidata = $PROGBITS ?AW .sidata;
    .sibss = $NOBITS ?AW .sibss;
};

```

```

DATA : !LOAD ?RW V0x3ff0100 {
    .data = $PROGBITS ?AW .data;
    .sdata = $PROGBITS ?AWG .sdata;
    .sbss = $NOBITS ?AWG .sbss;
    .bss = $NOBITS ?AW .bss;
};

```

```

### MINICUBE2用 ###
MRAMSEG : !LOAD ?RW V0x03ffeff0{
    MonitorRAM = $NOBITS ?AW MonitorRAM;
};

```

デフォルトのリンク・ディレクティブ
ファイルに追加するコード

MINICUBE2用の予約領域を確保

```

__tp_TEXT @ %TP_SYMBOL;
__gp_DATA @ %GP_SYMBOL &__tp_TEXT{DATA};
__ep_DATA @ %EP_SYMBOL;

```

```
●main.c
/*-----*/
/*
/*      NEC Electronics      V850ES/Jx3シリーズ
/*
/*-----*/
/*      V850ES/JJ3 サンプル・プログラム
/*-----*/
/*      LED点灯のスイッチ制御
/*-----*/
/* 【履歴】
/*      2009.05.--      新規作成
/*-----*/
/* 【概要】
/*      このサンプル・プログラムでは、クロック周波数の選択、ポート入出力の設定など、
/*      V850ES/JJ3 マイクロコントローラの基本的な初期設定を行います。
/*      また、初期設定完了後のメイン処理動作では、1つのスイッチ入力により2つのLED点灯
/*      を制御します。
/*
/*      リセット解除後に動作停止状態の周辺機能について、このサンプル・プログラムで
/*      使用しない周辺機能は、設定していません。
/*
/*
/* <初期設定の主な設定内容>
/*      ・システム・ウエイト・コントロール・レジスタを2クロック・ウエイトに設定
/*      ・オンチップ・デバッグ・モード・レジスタを通常動作モードに設定
/*      ・内蔵発振器の停止設定
/*      ・ウォッチドッグ・タイマ2動作停止
/*      ・システム・クロックはPLLを使用し8逡倍して3.2Mhzに設定
/*      ・未使用ポートの設定
/*      ・SW入力ポート、LED制御ポートの設定
/*
/*
/* <メイン処理の主な内容>
/*      ・スイッチ入力回数を検出
/*      ・LED点灯
/*
```

```

/* <スイッチ入力とLED点灯>
/*
/* +
/* | スイッチ入力回数 | LED 1 | LED 2 |
/* | (P03) | (PCM3) | (PCM2) |
/* |
/* | 0回目 | OFF | OFF |
/* | 1回目 | ON | OFF |
/* | 2回目 | ON | ON |
/* | 3回目 | OFF | ON |
/* +
/*
/* 4回目以降は0回からの繰り返し
/*
/*
/* 【ポート入出力の設定】
/*
/* ・入力ポート : P03
/* ・出力ポート : PCM2, PCM3
/* ・未使用のポート : P00-P02, P04-P06, P10-P11, P30-P39, P40-P42, P50-P55, P60-P615,
/* P70-P715, P80-P81, P90-P915, PCD0-PCD3, PCM0-PCM5, PCS0-PCS7,
/* PCT0-PCT7, PDH0-PDH7, PDL0-PDL15
/* 未使用のポートは全て出力ポート(Low出力)に設定しておく
/*
/*-----*/
/*-----*/
/* pragma指令 */
/*-----*/
#pragma ioreg /* 周辺I/Oレジスタ名有効化指定 */

/*-----*/
/* 定数定義 */
/*-----*/
#define VAL_RST_COUNT (0) /* スイッチ入力回数初期値 */
#define VAL_TIMER_WAIT (45920) /* 10msウエイト */

```

```

/*-----*/
/* プロトタイプ宣言 */
/*-----*/
static void f_init_vswc( void ); /* VSWCレジスタ設定処理 */
static void f_init_ocdm( void ); /* オンチップ・デバッグ・モード・レジスタの設
定*/
static void f_init_rcm( void ); /* 内蔵発振器の停止設定 */
static void f_init_wdtm2( void ); /* ウォッチドッグ・タイマ2の設定処理 */
static void f_init_lock( void ); /* CPU動作クロック設定処理 */
static void f_init_blank_port( void ); /* 未使用ポート設定初期化处理 */
static void f_init_use_port( void ); /* SW1・LEDポート設定初期化处理 */
void main( void ); /* メイン処理 */

/*****
/* 周辺初期設定 */
/*****
/*-----*/
/* VSWCレジスタ設定 */
/*-----*/
static void f_init_vswc( void )
{
    VSWC = 0b00010001; /* 内蔵周辺I/Oレジスタへのアクセス時に2ウェイトかける */

    return;
}

/*-----*/
/* オンチップ・デバッグ・モード・レジスタを通常動作モードに設定 */
/*-----*/
static void f_init_ocdm( void )
{
    /* OCDMを「通常動作モード」に指定 */

#pragma asm
    st.b    r0, PRCMD
    st.b    r0, OCDM
#pragma endasm

    return;
}

```

```
/*-----*/
/* 内蔵発振モード・レジスタ(RCM)設定 */
/*-----*/
static void f_init_rcm( void )
{
    RSTOP = 1;                /* 内蔵発振器の停止設定 */

    return;
}

/*-----*/
/* ウォッチドッグ・タイマ2(WDTM2)設定 */
/*-----*/
static void f_init_wdtm2( void )
{
    WDTM2 = 0b00000000;      /* ウォッチドッグ・タイマ2動作停止設定 */

    return;
}

/*-----*/
/* CPUの動作クロックをPLLモードに設定 */
/*-----*/
static void f_init_lock( void )
{
    /* CPUの動作クロックPLLモード：fx = 2.5~4 MHz(fxx = 20~32 MHz)選択 */
    /* 通倍率を設定するためPLLを停止（初期設定時はクロック・スルー・モード） */

    PLLON = 0;               /* PLL動作停止 */
                             /* PLLの通倍率を8通倍に設定 */

#pragma asm
    push r10
    mov 0x0B, r10
    st.b r10, PRCMD
    st.b r10, CKC
    pop r10
#pragma endasm

    PLLON = 1;               /* PLL動作許可 */

    while( LOCK );           /* PLL安定（ロック）待ち */

    SELPLL = 1;              /* PLLモードに設定 */
}
```



```

/* PCCレジスタ設定 */

/* クロック分周を「なし」に設定 */

#pragma asm
  push  r10
  mov   0x80, r10
  st.b  r10, PRCMD
  st.b  r10, PCC
  pop   r10
#pragma endasm

  return;
}
/*-----*/
/* 未使用ポートの設定 */
/*-----*/
static void f_init_blank_port( void )
{
  P0 = 0b00000000; /* P00-P06を出力LOWに設定(P03除) */
  PM0 = 0b10001000;

  P0 = 0b00000000; /* P02-P06を出力LOWに設定(P03除) */
  PM0 = 0b10001011;

  P1 = 0b00000000; /* P10-P11を出力LOWに設定 */
  PM1 = 0b11111100;

  P3H = 0b00000000; /* P38-P39を出力LOWに設定 */
  PM3H = 0b11111100;
  P3L = 0b00000000; /* P30-P37を出力LOWに設定 */
  PM3L = 0b00000000;

  P4 = 0b00000000; /* P40-P42を出力LOWに設定 */
  PM4 = 0b11111000;

  P5 = 0b00000000; /* P50-P55を出力LOWに設定 */
  PM5 = 0b11000000;

  P6H = 0b00000000; /* P68-P615を出力LOWに設定 */
  PM6H = 0b00000000;
  P6L = 0b00000000; /* P60-P67を出力LOWに設定 */
  PM6L = 0b00000000;

```

V850JG3では、P0はP02-P06のみ設定。

V850JG3では、P6は設定不要。

設定不要

```
P7H = 0b00000000;          /* P78-P715を出力LOWに設定      */
PM7H = 0b00000000;
P7L = 0b00000000;          /* P70-P77を出力LOWに設定      */
PM7L = 0b00000000;
```

V850JG3では、P7はP70-P711のみ設定。

```
P7H = 0b00000000;          /* P78-P711を出力LOWに設定      */
PM7H = 0b11110000;
P7L = 0b00000000;          /* P70-P77を出力LOWに設定      */
PM7L = 0b00000000;
```

```
P8 = 0b00000000; /* P80-P81を出力LOWに設定 */
PM8 = 0b11111100;
```

V850JG3では、P8は設定不要。

設定不要

```
P9H = 0b00000000; /* P98-P915を出力LOWに設定 */
PM9H = 0b00000000;
P9L = 0b00000000; /* P90-P97を出力LOWに設定 */
PM9L = 0b00000000;
```

```
PCD = 0b00000000; /* PCD0-PCD3を出力LOWに設定 */
PMCD = 0b11110000;
```

V850JG3では、PCDは設定不要。

設定不要

```
PCM = 0b00000000; /* PCM0-PCM5を出力LOWに設定 */
PMCM = 0b11000000;
```

V850ES/JG3では、PCMはPCM0-PCM3のみ設定。

```
PCM = 0b00000000; /* PCM0-PCM3を出力LOWに設定 */
PMCM = 0b11110000;
```

```
PCS = 0b00000000; /* PCS0-PCS7を出力LOWに設定 */
PMCS = 0b00000000;
```

V850JG3では、PCSは設定不要。

設定不要

```
PCT = 0b00000000; /* PCT0-PCT7を出力LOWに設定 */
PMCT = 0b00000000;
```

V850ES/JG3では、PCTはPCT0,PCT1,PCT4,PCT6のみ設定。

```
PCT = 0b00000000; /* PCT0,PCT1,PCT4,PCT6を出力LOWに設定 */
PMCT = 0b10101100;
```

```
PDH = 0b00000000; /* PDH0-PDH7を出力LOWに設定 */
PMDH = 0b00000000;
```

V850ES/JG3では、PDHはPDH0-PDH5のみ設定。

```
PDH = 0b00000000; /* PDH0-PDH5を出力LOWに設定 */
PMDH = 0b11000000;
```

```
PDLH = 0b00000000; /* PDL8-PDL15を出力LOWに設定 */
PMDLH = 0b00000000;
PDLL = 0b00000000; /* PDL0-PDL7を出力LOWに設定 */
PMDLL = 0b00000000;
```

```

    return;
}

/*-----*/
/* SW入力ポート, LED制御ポートの設定 */
/*-----*/
/static void f_init_use_port( void )
{
    /* SW出力ポート設定 */
    P0 = 0b00000000;          /* P03を入力LOWに設定          */
    PM0 = 0b10001000;

    /* LED出力ポート設定 */
    PCM = 0b00001100;        /* PCM2-PCM3を出力HIGHに設定    */
    PMCM = 0b11000000;

    return;
}

/*****
/* メインモジュール      */
/*****
void main( void )
{
    extern unsigned int _S_romp;          /* ROM化シンボルの外部参照      */

    /*-----*/
    /* 変数宣言や変数初期設定      */
    /*-----*/
    const unsigned char outdata[] = { /* LED表示データパターン用配列 */
        0x0c,          /* 全LED消灯          */
        0x04,          /* LED1点灯          */
        0x00,          /* LED1とLED2点灯    */
        0x08           /* LED2点灯          */
    };

    unsigned char indata = 0b00000001; /* スイッチ状態記憶(前回値1を初期値に) */
    unsigned char count;          /* スイッチ入力回数          */
    unsigned long loop_wait;      /* forループ用カウンタ          */

    count = VAL_RST_COUNT;        /* スイッチ入力回数初期化          */

```

```

/*-----*/
/* 周辺機能初期化          */
/*-----*/
f_init_vswc();          /* VSWCレジスタの設定          */
f_init_ocdm();         /* オンチップ・デバッグを通常動作モードに設定 */
f_init_rcm();          /* 内蔵発振器の禁止設定        */
f_init_wdtm2();        /* ウォッチドッグ・タイマ2の設定 */
f_init_lock();         /* CPUの動作クロックをPLLモードに設定 */
f_init_blank_port();   /* 未使用ポート設定            */
f_init_use_port();     /* SW入力ポート, LED制御ポートの設定 */

/*-----*/
/* ROM化処理              */
/*-----*/
_rcopy( &_S_romp, -1 ); /* ROM化処理実施              */

/*-----*/
/* MINICUBE2 OCD用割り込み許可処理 */
/*-----*/
__EI();                /* 割り込み許可                */

/*-----*/
/* LED点灯処理           */
/*-----*/
while ( 1 )
{
    indata <=<= 1;      /* スイッチ状態前回値を更新          */
    indata |= P0.3;     /* スイッチ状態今回値を更新          */
    if ( ( indata & 0b00001111 ) == 0b00001100 )
    {
        count++;       /* スイッチ入力回数更新              */
        count &= 0b00000011;
        PCM = outdata[count]; /* テーブルからLED表示データを読み出して表示 */
    }

    /* 10msウェイト */
    for ( loop_wait = 0; loop_wait <= VAL_TIMER_WAIT; loop_wait++ )
    {
        __nop();
    }
}
return;
}

```

【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