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。

お客様各位

資料中の「日立製作所」、「日立XX」等名称の株式会社ルネサス テクノロジへの変更について

2003年4月1日を以って三菱電機株式会社及び株式会社日立製作所のマイコン、ロジック、アナログ、ディスクリート半導体、及びDRAMを除くメモリ(フラッシュメモリ・SRAM等)を含む半導体事業は株式会社ルネサス テクノロジに承継されました。従いまして、本資料中には「日立製作所」、「株式会社日立製作所」、「日立半導体」、「日立XX」といった表記が残っておりますが、これらの表記は全て「株式会社ルネサス テクノロジ」に変更されておりますのでご理解の程お願い致します。尚、会社商標・ロゴ・コーポレートステートメント以外の内容については一切変更しておりませんので資料としての内容更新ではありません。

ルネサステクノロジ ホームページ (http://www.renesas.com)

2003年4月1日
株式会社ルネサス テクノロジ
カスタマサポート部

RENESAS

ご注意

安全設計に関するお願い

1. 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

本資料ご利用に際しての留意事項

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてルネサス テクノロジが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ルネサス テクノロジは責任を負いません。
3. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ (http://www.renesas.com)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したものですが万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任は負いません。
6. 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 本資料に関し詳細についてのお問い合わせ、その他お気付きの点がございましたらルネサス テクノロジ、ルネサス販売または特約店までご照会ください。

Application Note

RENESAS

SH7055 内蔵I/O 編

アプリケーションノート

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00　1999.03

ご注意

1. 本書に記載の製品及び技術のうち「外国為替及び外国貿易法」に基づき安全保障貿易管理関連貨物・技術に該当するものを輸出する場合，または国外に持ち出す場合は日本国政府の許可が必要です。
2. 本書に記載された情報の使用に際して，弊社もしくは第三者の特許権，著作権，商標権，その他の知的所有権等の権利に対する保証または実施権の許諾を行うものではありません。また本書に記載された情報を使用した事により第三者の知的所有権等の権利に関わる問題が生じた場合，弊社はその責を負いませんので予めご了承ください。
3. 製品及び製品仕様は予告無く変更する場合がありますので，最終的な設計，ご購入，ご使用に際しましては，事前に最新の製品規格または仕様書をお求めになりご確認ください。
4. 弊社は品質・信頼性の向上に努めておりますが，宇宙，航空，原子力，燃焼制御，運輸，交通，各種安全装置，ライフサポート関連の医療機器等のように，特別な品質・信頼性が要求され，その故障や誤動作が直接人命を脅かしたり，人体に危害を及ぼす恐れのある用途にご使用をお考えのお客様は，事前に弊社営業担当迄ご相談をお願い致します。
5. 設計に際しては，特に最大定格，動作電源電圧範囲，放熱特性，実装条件及びその他諸条件につきましては，弊社保証範囲内でご使用いただきますようお願い致します。
   保証値を越えてご使用された場合の故障及び事故につきましては，弊社はその責を負いません。
   また保証値内のご使用であっても半導体製品について通常予測される故障発生率，故障モードをご考慮の上，弊社製品の動作が原因でご使用機器が人身事故，火災事故，その他の拡大損害を生じないようにフェールセーフ等のシステム上の対策を講じて頂きますようお願い致します。
6. 本製品は耐放射線設計をしておりません。
7. 本書の一部または全部を弊社の文書による承認なしに転載または複製することを堅くお断り致します。
8. 本書をはじめ弊社半導体についてのお問い合わせ，ご相談は弊社営業担当迄お願い致します。

はじめに

ＳＨ７０５５はＲＩＳＣ方式のＣＰＵをコアにして、システム構成に必要な周辺機能を集積したシングルチップＲＩＳＣマイクロコンピュータです。

１チップ上にＣＰＵ、ＲＯＭ、ＲＡＭ、ＤＭＡＣ、ＡＴＵ-Ⅱ、ＳＣＩ、Ａ／Ｄ変換器、ＡＵＤ、ＵＢＣ、ＨＣＡＮ、割り込みコントローラ、Ｉ／Ｏポート等を内蔵しており、小規模システムから大規模システムまで幅広いアプリケーションに適用できます。

ＳＨ７０５５アプリケーションノート（内蔵Ｉ／Ｏ編）は、ＳＨ７０５５の周辺機能を使用したタスク例及びアプリケーション例について述べており、ユーザにてソフトウェア設計及びハードウェア設計の際、ご参考として役立てていただけるようにまとめたものです。

なお、本アプリケーションノートに掲載されているタスク例及びアプリケーション例は動作確認しておりますが、実際にご使用になる場合には、必ず動作確認の上ご使用くださいますようにお願いします。

目次

1. ＳＨ７０５５
アプリケーションノート使用手引

目　次

１．１　内蔵Ｉ／Ｏ編構成

内蔵Ｉ／Ｏ編は図１に示す構成で周辺機能の使用方法について説明しています。レジスタのラベル名は各タスク共通のヘッダファイルの名前を使用しています。

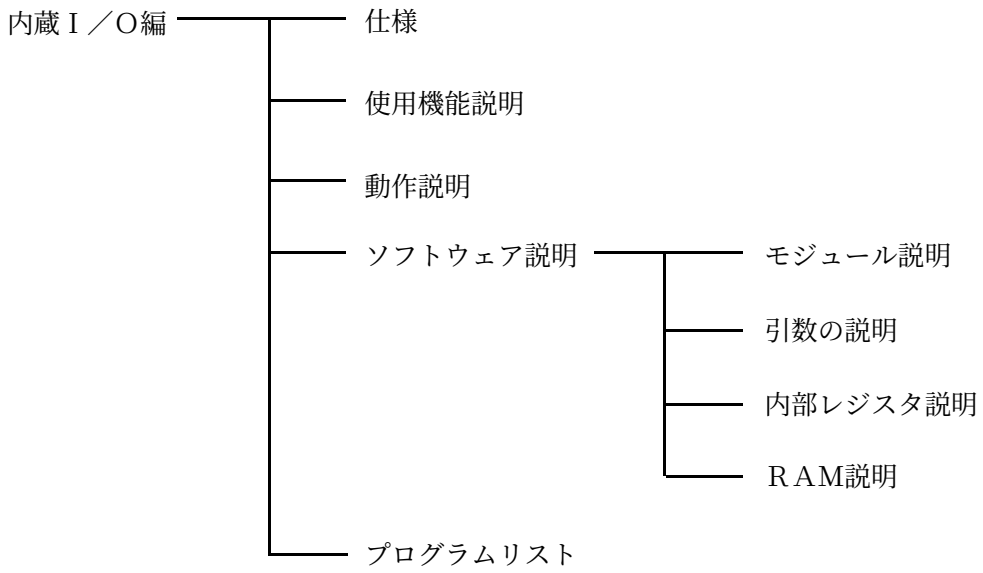


図１　内蔵Ｉ／Ｏ編構成

（１）仕様
　タスク例のシステム仕様について説明しています。

（２）使用機能説明
　タスク例で使用する周辺機能の特長および周辺機能の割り付けについて説明しています。

（３）動作説明
　タスク例の動作をタイミングチャートを使用し説明しています。

（４）ソフトウェア説明

（ａ）モジュール説明
　タスク例を動作させるソフトウェアのモジュールについて説明しています。

（ｂ）引数の説明
　モジュールを実行する際に必要な入力引数と、実行後の出力引数について説明しています。

（ｃ）内部レジスタ説明
　モジュールで設定する周辺機能の内部レジスタ（タイマコントロールレジスタ、シリアルモードレジスタ等）について説明します。

（ｄ）ＲＡＭ説明
　モジュールで使用するＲＡＭのラベル名および機能について説明します。

（５）プログラムリスト
　タスク例を実行するソフトウェアのプログラムリストを示します。尚、各プログラムで使用するヘッダファイルにはライブラリ関数用ヘッダファイル、組み込み関数用ヘッダファイル、ＳＨ７０５５内蔵Ｉ／Ｏレジスタ用ヘッダファイルがあります。ライブラリ関数用ヘッダファイルと組み込み関数用ヘッダファイルの仕様はＳＨシリーズＣコンパイラを参照してください。ＳＨ７０５５内蔵Ｉ／Ｏレジスタ用ヘッダファイルの内容は２．２７に掲載しています。

１．２　付録

付録にはＡＵＤ(ＲＡＭモニタモード)の制御例とRAMによるフラッシュメモリのエミュレーションを添付しています。

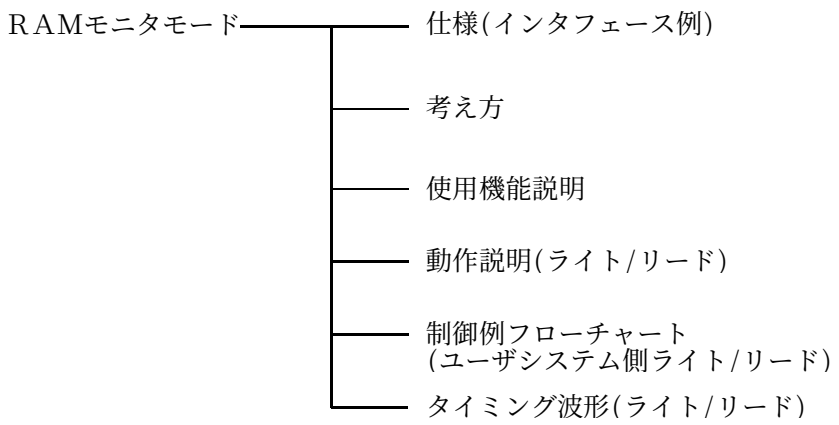


図２　ＲＡＭモニタモード構成

（１）仕様(インタフェース例)
　ＳＨ７０５５のＡＵＤインタフェースについて説明しています。

（２）考え方
　ＡＵＤインタフェースの動作をタイミングチャートを使用し説明しています。

（３）使用機能説明
　ＳＨ７０５５のＡＵＤ内蔵機能(端子)と低消費電力モードのレジスタ機能割り付けを示します。

（４）動作説明(ライト/リード)
　ＳＨ７０５５のＡＵＤによるＳＨ７０５５内蔵ＲＡＭライト/リード動作原理を示します。

（５）制御例フローチャート(ユーザシステム側ライト/リード)
　ユーザシステム側の制御フローチャートを示します。

（６）タイミング波形(ライト/リード)
　ＡＵＤの制御タイミング波形を示します。

図２に示す構成でＡＵＤ制御例について説明しています。

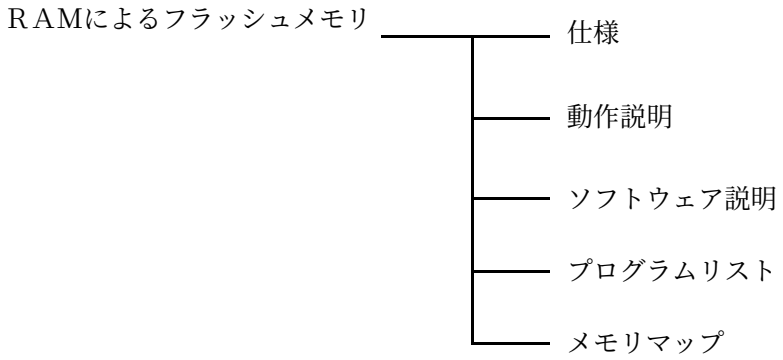


図３　ＲＡＭによるフラッシュメモリのエミュレーション構成

（１）仕様
ＳＨ７０５５のＲＡＭを使用したフラッシュメモリのエミュレーションについて説明しています。

（２）動作説明
ＲＡＭ上でのフラッシュメモリのエミュレーションから、変換データのフラッシュメモリへの書き換えまでの動作の説明しています。

（３）ソフトウェア説明
フラッシュメモリ上で動作させるアプリケーションプログラム、及び内蔵ＲＡＭ上で動作させるデータ変換プログラム内のモジュール及び使用内部レジスタの説明をしています。

（４）プログラムリスト
フラッシュメモリ上で動作させるアプリケーションプログラム、内蔵ＲＡＭ上で動作させるデータ変換プログラム、及びヘッダファイルを示します。

（５）メモリマップ
アプリケーション例のメモリマップを示します。

2. SH7055内蔵I／O編

目　次

2.1　命令フェッチブレーク	MCU	ＳＨ７０５５	使用機能	ＵＢＣ

仕様

（1）図１に示すようにユーザブレークアドレスレジスタに設定したブレークポイント（00001020番地）にある命令の手前でユーザブレーク割り込みを発生します。

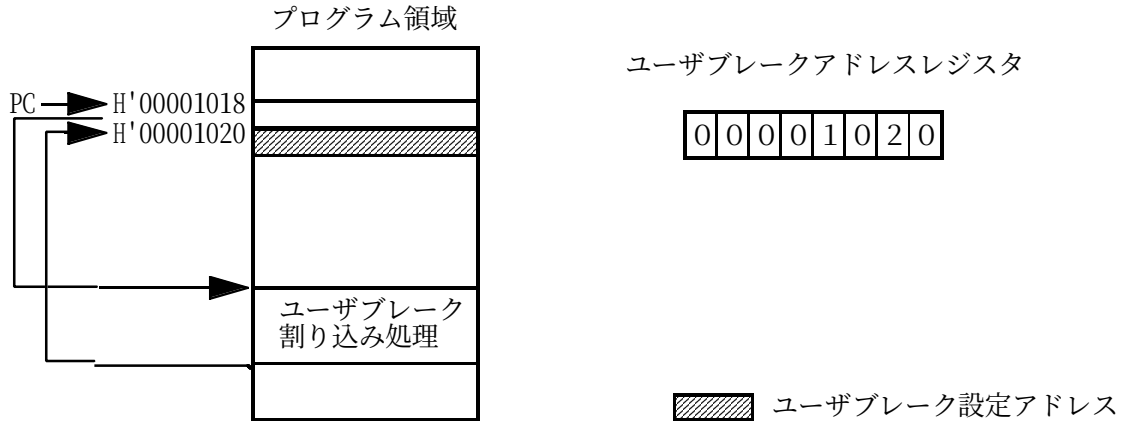


図１　ユーザブレークコントローラの動作ブロック図

使用機能説明

表１、表２に本タスク例の機能割付を示します。表１、表２に示すようにＳＨ７０５５に内蔵しているＵＢＣ及び低消費電力モードの機能を割付、ユーザブレークを行ないます。

表１　ＵＢＣ機能割付

ＵＢＣレジスタ	機　　能
UBAR	ユーザブレークアドレスを設定する。
UBAMR	ユーザブレークマスクアドレスを設定する。
UBBR	ユーザブレーク条件を設定する。

表２　低消費電力モード機能割付

低消費電力 モードレジスタ	機　　能
MSTCRW	UBCのクロック供給を設定する。

図２にユーザブレークコントローラを使用したソフトウェア例を示します。あらかじめ初期設定でプログラムブレークを設定し、ブレークポイントの手前の命令で割り込みを発生します。ブレーク処理ではブレークの有無を示すフラグをセットします。

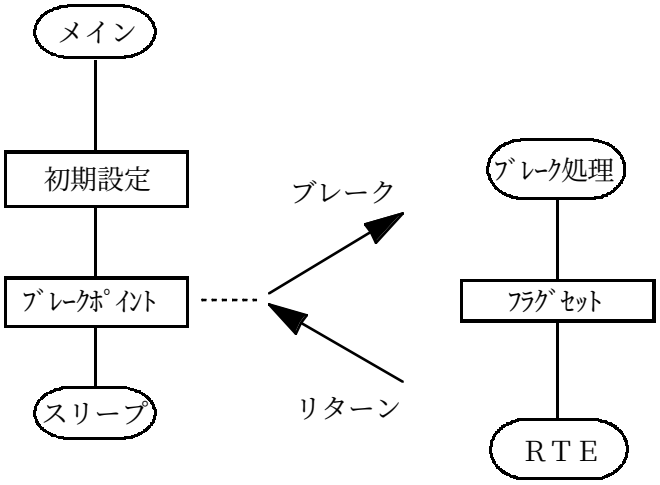


図２　ユーザブレークコントローラ機能を使用したソフトウェア例

2.1　命令フェッチブレーク	MCU	SH7055	使用機能	UBC

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機　　能
メインルーチン	main	UBCの初期設定を行なう。
ブレーク処理	ubcbk	ユーザブレークで起動し、ブレークの有無を示すフラグをセットする。

（2）引数の説明

本タスクでは引数は使用してません。

（3）使用内部レジスタ説明

レジスタ名	機　　能	設定値	使用モジュール名
UBC.UBARH UBC.UBARL	命令フェッチアドレスを設定する。	0x0000 0x1020 注	メインルーチン
UBC.UBBR	ブレーク条件をCPU、命令フェッチに設定する。	0x0054	
MSTCRW	UBCを動作に設定する。	0x3C00	

注：コンパイル後リストファイルより参照

（4）使用ＲＡＭ

ラベル名	機　　能	データ長	使用モジュール名
pcf	命令フェッチに使用する変数。	unsigned char	メインルーチン
brk	ブレークの有無を示すフラグ		

プログラムリスト

```
/************************************************************************/
/*              命令フェッチブレーク                                    */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ        */
/************************************************************************/
/*              関数プロトタイプ宣言                                    */
/************************************************************************/
void main( void );
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define pcf       (*(unsigned char  *)0xFFFFC000) /* 命令ﾌｪｯﾁに使用する変数 */
#define brk       (*(unsigned char  *)0xFFFFC001) /* ﾌﾞﾚｰｸ判定ﾌﾗｸﾞ          */
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
    MSTCRW = 0x3C00;                    /* UBC動作                      */
    UBC.UBARH = 0x0000;                 /* ﾌﾞﾚｰｸｱﾄﾞﾚｽの設定             */
    UBC.UBARL = 0x1020;                 /* ﾌﾞﾚｰｸｱﾄﾞﾚｽの設定             */
    UBC.UBBR = 0x0054;                  /* ﾊﾞｽｻｲｸﾙ:CPU、命令ﾌｪｯﾁ、ﾘｰﾄﾞ  */
    set_imask(0x0);                     /* 割り込み許可                 */
    pcf = 1;                            /* 命令ﾌｪｯﾁﾌﾞﾚｰｸ設定行          */
    sleep();
}
/************************************************************************/
/*              ブレーク処理                                            */
/************************************************************************/
#pragma interrupt( ubcbk )
void ubcbk( void )
{
    brk = 1;                            /* ﾌﾞﾚｰｸ判定ﾌﾗｸﾞｾｯﾄ             */
}
```

2.2　データアクセスブレーク	MCU	ＳＨ７０５５	使用機能	ＵＢＣ

仕様

（1）図１に示すようにユーザブレークアドレスレジスタに設定したブレークポイント（FFFFC000番地）にあるデータをアクセスし、ユーザブレーク割り込みを発生します。
（2）ブレーク条件はＣＰＵによる１バイトデータのライトです。

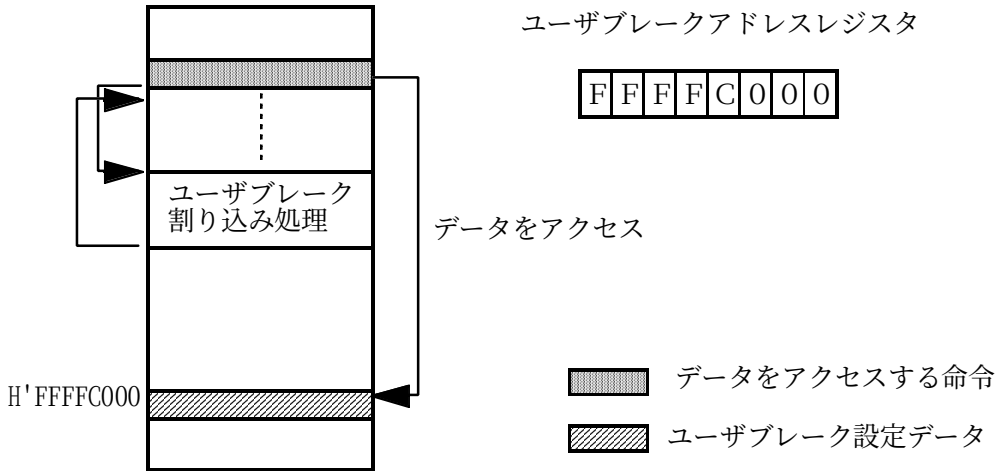


図１　ユーザブレークコントローラの動作ブロック図

使用機能説明

表１、表２に本タスク例の機能割付を示します。表１、表２に示すようにＳＨ７０５５に内蔵しているＵＢＣ及び低消費電力モードの機能を割付、ユーザブレークを行ないます。

表１　ＵＢＣ機能割付

ＵＢＣレジスタ	機　　能
UBAR	ユーザブレークアドレスを設定する。
UBAMR	ユーザブレークマスクアドレスを設定する。
UBBR	ユーザブレーク条件を設定する。

表２　低消費電力モード機能割付

低消費電力 モードレジスタ	機　　能
MSTCRW	UBCのクロック供給を設定する。

動作説明

図２にユーザブレークコントローラを使用したソフトウェア例を示します。あらかじめ初期設定でデータアクセスブレークを変数に設定し、ブレーク設定変数アクセス時にブレーク割り込みを発生します。ブレーク処理ではブレークの有無を示すフラグをセットします。

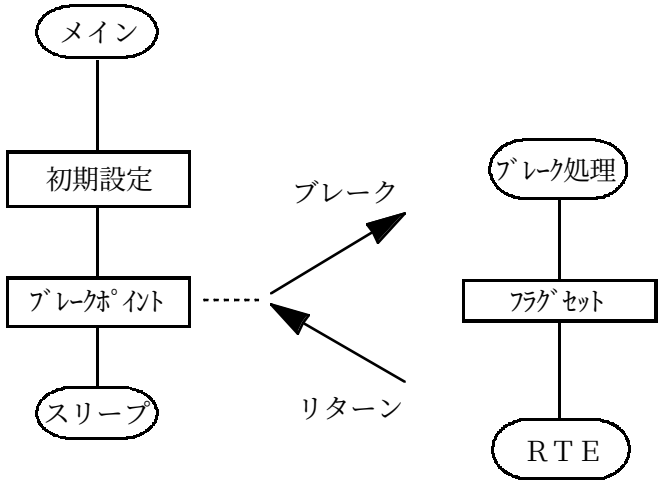


図２　ユーザブレークコントローラ機能を使用したソフトウェア例

（1）モジュール説明

モジュール名	ラベル名	機　　能
メインルーチン	main	UBCの初期設定を行なう。
ブレーク処理	ubcbk	ユーザブレークで起動し、ブレークの有無を示すフラグをセットする。

（2）引数の説明

本タスクでは引数は使用してません。

（3）使用内部レジスタ説明

レジスタ名	機　　能	設定値	使用モジュール名
UBAR	データアクセスアドレスを設定する。	&dat_ac	メインルーチン
UBC.UBBR	ユーザブレーク条件を設定する。	0x0069	
MSTCRW	UBCの動作を設定する。	0x3C00	

（4）使用ＲＡＭ

ラベル名	機　　能	データ長	使用モジュール名
dat_ac	アクセスブレークを設定する変数。	unsigned char	メインルーチン
brk	ブレークの有無を示すフラグ		

プログラムリスト

```
/************************************************************************/
/*              データアクセスブレーク                                  */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ        */
/************************************************************************/
/*              関数プロトタイプ宣言                                    */
/************************************************************************/
void main( void );
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define dat_ac     (*(unsigned char  *)0xFFFFC000)        /* ﾌﾞﾚｰｸ設定変数   */
#define brk        (*(unsigned char  *)0xFFFFC001)        /* ﾌﾞﾚｰｸ判定ﾌﾗｸﾞ   */
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
    MSTCRW = 0x3C00;                /* UBC動作の設定                    */
    UBAR = (long)&dat_ac;           /* ﾌﾞﾚｰｸｱﾄﾞﾚｽの設定                 */
    UBC.UBBR = 0x0069;              /* ﾊﾞｽｻｲｸﾙ:CPU、命令ﾌｪｯﾁ、ﾊﾞｲﾄﾃﾞｰﾀﾗｲﾄ  */
    set_imask(0x0);                 /* 割り込み許可                     */
    dat_ac = 1;                     /* ﾌﾞﾚｰｸ実行ﾌﾗｸﾞｾｯﾄ                 */
    sleep();
}
/************************************************************************/
/*              ブレーク処理                                            */
/************************************************************************/
#pragma interrupt( ubcbk )
void ubcbk( void )
{
    brk = 1;                        /* ﾌﾞﾚｰｸ判定ﾌﾗｸﾞのｾｯﾄ               */
}
```

2.3　DMACによるＳＣＩ送信	MCU	ＳＨ７０５５	使用機能	ＳＣＩ、DMAC

仕様

（1）図１に示すように、ＳＨ７０５５のSCIを調歩同期式モードで使用し、コンソールに32バイトのデータを送信します。
（2）転送プロトコルは9600bps、8ビットデータ、1ストップビット及びノンパリティとします。
（3）RAMからTDRへのデータ転送は図２に示すようにDMACの間接アドレス転送モードを使用し、CPUによるデータのライト時にユーザブレーク割り込みでDMACを起動し、以下に示すようなデータ転送を行います。
　（ａ）RAM上に格納されたデータをDMAC内のテンポラリバッファに格納し、それをアドレスとしてRAM上からデータを取り出します。
　（ｂ）取り出したデータをTDRにバイト単位で順次転送します。
（4）DMACの転送条件は表１に示す通りです。

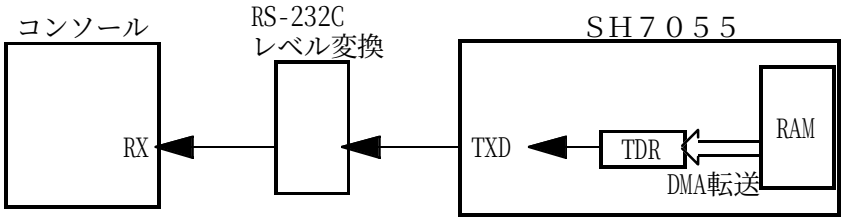


図１　ＳＨ７０５５によるＲＡＭ上データのＳＣＩ転送ブロック図

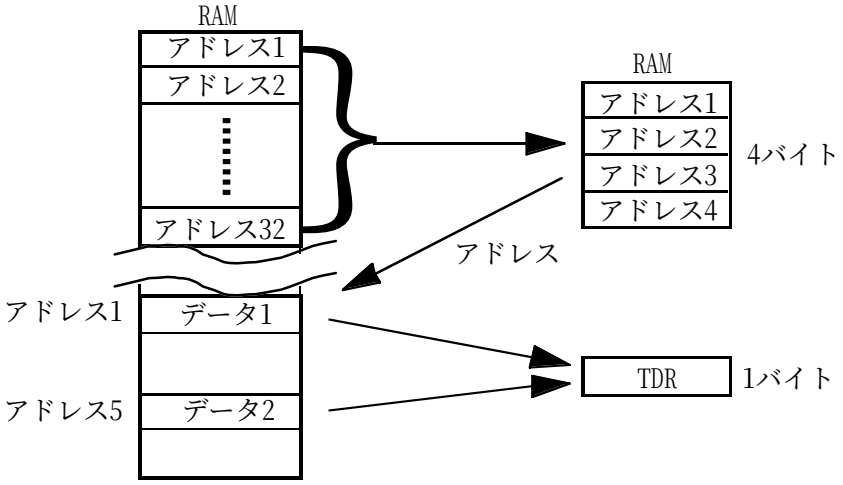


図２　ＤＭＡＣを用いたデータ転送（転送元間接アドレス）

表１　DMA転送条件

条件項目	内容
DMACチャネル	チャネル3
転送元	内蔵RAM
転送先	内蔵SCIチャネル0
転送回数	16回
転送元アドレス	増加
転送先アドレス	固定
転送要求元	内蔵SCIチャネル0
バスモード	サイクルスチール
転送単位	バイト

使用機能説明

表２、表３、表４、表５、表６に本タスク例の機能割付を示します。表２、表３、表４、表５、表６に示すようにＳＨ７０５５に内蔵しているDMAC、SCI、UBC、低消費電力モード及びPFCの機能を割付、RAM上のデータをSCIによって転送します。

表２　ＤＭＡＣ機能割付

ＤＭＡＣレジスタ	機　　能
SAR3	転送元アドレスを設定する。
DAR3	転送先アドレスを設定する。
TCR3	転送回数を設定する。
CHCR3	DMACの動作モード、転送方法等を設定する。
DMAOR	DMACの実行するチャネルの優先順位を設定する。

表３　ＳＣＩ機能割付

ＳＣＩ機能		機　　能
端子	RXD	コンソールからデータを受信する。
	TXD	コンソールへデータを送信する。
レジスタ	SMR	SCIの送信フォーマットを設定する。
	SCR	SCIの割り込みの許可／禁止を設定する。
	SSR	割り込みステータスを設定する。
	RDR	コンソールから受信したデータを設定する。
	TDR	コンソールへ送信するデータを設定する。
	BRR	転送レートを設定する。

表４　ＵＢＣ機能割付

ＳＣＩレジスタ	機　　能
UBAR	ユーザブレークアドレスを設定する。
UBBR	ユーザブレーク条件を設定する。

表５　低消費電力モード機能割付

低消費電力 モードレジスタ	機　　能
MSTCRW	UBCの動作を設定する。

表６　ＰＦＣ機能割付

ＰＦＣレジスタ	機　　能
PAIOR	端子の入出力方向を設定する。
PACRH	端子の機能を選択する。

2.3　DMACによるＳＣＩ送信	MCU	ＳＨ７０５５	使用機能	ＳＣＩ、DMAC

ソフトウェア説明

（１）モジュール説明

モジュール名	ラベル名	機　　能
メインルーチン	main	SCI及びDMACの初期設定を行なう。
転送終了	dma3	DEI3で起動し、SCI送信割り込み、DMACの転送動作を禁止する。
ユーザブレイク	ubcbk	SCI送信割り込み、DMACの転送動作を許可する。
SCI送信終了	sci_tr	DMACによって送信終了フラグをクリアする。

（２）引数の説明

ラベル名	機　　能	データ長	使用モジュール名
dat.addr0 〜 dataddr15	参照アドレスを格納する。	unsigned long	メインルーチン
data0 〜 data15	参照データ	unsigned char	メインルーチン

（３）使用内部レジスタ説明

レジスタ名	機　　能	設定値	使用モジュール名
DMAC3.SAR	転送元RAM先頭アドレスを設定する。	&dat.addr0	メインルーチン
DMAC3.DAR	TDRのアドレスを設定する。	&SCI0.TDR	メインルーチン 転送終了
DMAC3.TCR	転送回数（16回）を設定する。	0x10	
DMAC3.CHCR	DMACの動作モード、転送方法割り込みの有無を設定する。	0x10011004	メインルーチン
DMACC.DMAOR	DMACの起動を許可する。	0x0001	
PA.PAIOR	SCIの入出力を設定する。	0x4000	
PA.PACRH	端子マルチプレクスをSCI0の使用にする。	0x5000	
INTC.IPRC	DMAC3の割り込み優先レベルを13に設定する。	0x0D00	
INTC.IPRK	SCI0の割り込み優先レベルを12に設定する。	0xC000	
SCI0.SMR	SCIを調歩同期式モードに設定する。	0x00	
SCI0.SCR	送信割り込み、送信動作を許可する。	0xA0	
SCI0.BRR	転送レートを設定する。	0x40	
UBAR	転送元RAM先頭アドレスを設定する。	&data0	
UBC.UBBR	ブレーク条件をデータアクセス、ライト時、バイトサイズに設定する。	0x0069	
MSTCRW	UBCの動作クロックの供給を設定する。	0x3C00	

（４）使用ＲＡＭ

本タスクでは引数以外のＲＡＭは使用していません。

プログラムリスト

```
/************************************************************************/
/*              DMACによるSCI送信                                       */
/************************************************************************/
#include <machine.h>                            /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                               /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ        */
/************************************************************************/
/*              関数プロトタイプ宣言                                    */
/************************************************************************/
void main( void );
void dat_set( void );
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define data0 (*(volatile  unsigned char *)0xFFFFC000)
#define data1 (*(volatile  unsigned char *)0xFFFFC001)
#define data2 (*(volatile  unsigned char *)0xFFFFC002)
#define data3 (*(volatile  unsigned char *)0xFFFFC003)
#define data4 (*(volatile  unsigned char *)0xFFFFC004)
#define data5 (*(volatile  unsigned char *)0xFFFFC005)
#define data6 (*(volatile  unsigned char *)0xFFFFC006)
#define data7 (*(volatile  unsigned char *)0xFFFFC007)
#define data8 (*(volatile  unsigned char *)0xFFFFC008)
#define data9 (*(volatile  unsigned char *)0xFFFFC009)
#define data10 (*(volatile  unsigned char *)0xFFFFC00A)
#define data11 (*(volatile  unsigned char *)0xFFFFC00B)
#define data12 (*(volatile  unsigned char *)0xFFFFC00C)
#define data13 (*(volatile  unsigned char *)0xFFFFC00D)
#define data14 (*(volatile  unsigned char *)0xFFFFC00E)
#define data15 (*(volatile  unsigned char *)0xFFFFC00F)
volatile  struct addr
{
    long   addr0;                               /* 転送ｱﾄﾞﾚｽ0             */
    long   addr1;                               /* 転送ｱﾄﾞﾚｽ1             */
    long   addr2;                               /* 転送ｱﾄﾞﾚｽ2             */
    long   addr3;                               /* 転送ｱﾄﾞﾚｽ3             */
    long   addr4;                               /* 転送ｱﾄﾞﾚｽ4             */
    long   addr5;                               /* 転送ｱﾄﾞﾚｽ5             */
    long   addr6;                               /* 転送ｱﾄﾞﾚｽ6             */
    long   addr7;                               /* 転送ｱﾄﾞﾚｽ7             */
    long   addr8;                               /* 転送ｱﾄﾞﾚｽ8             */
    long   addr9;                               /* 転送ｱﾄﾞﾚｽ9             */
    long   addr10;                              /* 転送ｱﾄﾞﾚｽ10            */
    long   addr11;                              /* 転送ｱﾄﾞﾚｽ11            */
    long   addr12;                              /* 転送ｱﾄﾞﾚｽ12            */
    long   addr13;                              /* 転送ｱﾄﾞﾚｽ13            */
    long   addr14;                              /* 転送ｱﾄﾞﾚｽ14            */
    long   addr15;                              /* 転送ｱﾄﾞﾚｽ15            */
};
#define dat (*(struct addr *)0xFFFFC080)
```

プログラムリスト

```
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
    signed int lp;
    PA.PAIOR = 0x4000;              /* TXD0出力,RXD0入力                */
    PA.PACRH = 0x5000;              /* TXD0,RXD0使用                    */
    SCI0.SCR = 0x00;                /* 送信・受信動作を禁止             */
    SCI0.SMR = 0x00;                /* 調歩同期式､8ﾋﾞｯﾄﾃﾞｰﾀ、ﾊﾟﾘﾃｨなし  */
    SCI0.BRR = 0x40;                /* ﾋﾞｯﾄﾚｰﾄ9600bps                   */
    for( lp = 1; lp < 1; lp++ );    /* ｳｪｲﾄ                             */
    SCI0.SCR = 0x20;                /* 送信動作を許可する。             */
    MSTCRW = 0x3C00;                /* ﾃﾞｰﾀ設定                         */
    UBAR = (long)&data0;            /* ﾕｰｻﾞﾌﾞﾚｲｸｱﾄﾞﾚｽをRAMに設定する  */
    UBC.UBBR = 0x0069;              /* ﾃﾞｰﾀのﾗｲﾄｻｲｸﾙでﾌﾞﾚｰｸ             */
    DMAC3.SAR = (long)(&dat.addr0); /* 転送元ｱﾄﾞﾚｽ：RAM                */
    DMAC3.DAR = (long)(&SCI0.TDR);  /* 転送先ｱﾄﾞﾚｽ：SCIﾄﾗﾝｽﾐｯﾄﾃﾞｰﾀﾚｼﾞｽﾀ */
    DMAC3.CHCR = 0x10011004;        /* ｲﾝﾀﾞｲﾚｸﾄ,ｿｰｽ増加,ﾊﾞｲﾄ転送      */
    DMACC.DMAOR = 0x0001;           /* DMAC起動許可                     */
    INTC.IPRC = 0x0D00;             /* DMA3割り込み優先ﾚﾍﾞﾙを13に設定   */
    INTC.IPRK = 0xC000;             /* SCI0割り込み優先ﾚﾍﾞﾙを12に設定   */
    set_imask(0x0);                 /* 割り込み許可                     */
    dat_set();                      /* ﾃﾞｰﾀ設定                         */
    while(1);                       /* 無限ﾙｰﾌﾟ（割込み待ち）           */
}
/************************************************************************/
/*              ユーザブレイク割込みルーチン                            */
/************************************************************************/
#pragma interrupt( ubcbk )
void ubcbk( void )
{
    SCI0.SCR |= 0x80;               /* SCI受信割り込みを許可する        */
    DMAC3.DMATCR = 0x10;            /* 転送回数：16回                   */
    DMAC3.CHCR |= 0x00000001;       /* DEﾌﾗｸﾞｾｯﾄ                        */
}
/************************************************************************/
/*              DMA転送終了割込みルーチン                               */
/************************************************************************/
#pragma interrupt( dma_sci )
void dma_sci( void )
{
    DMAC3.CHCR &= 0xFFFFFFFC;       /* TEﾌﾗｸﾞｸﾘｱ                        */
    SCI0.SCR &= 0x7F;               /* SCI受信割り込みを禁止する        */
}
/************************************************************************/
/*              データエンプティ割り込みルーチン                        */
/************************************************************************/
#pragma interrupt( sci_txi )
void sci_txi( void )
{
}
```

プログラムリスト

```
/************************************************************************/
/*              データ設定ルーチン                                      */
/************************************************************************/
void dat_set( void )
{
    dat.addr0 = (long)(&data0);
    dat.addr1 = (long)(&data1);
    dat.addr2 = (long)(&data2);
    dat.addr3 = (long)(&data3);
    dat.addr4 = (long)(&data4);
    dat.addr5 = (long)(&data5);
    dat.addr6 = (long)(&data6);
    dat.addr7 = (long)(&data7);
    dat.addr8 = (long)(&data8);
    dat.addr9 = (long)(&data9);
    dat.addr10 = (long)(&data10);
    dat.addr11 = (long)(&data11);
    dat.addr12 = (long)(&data12);
    dat.addr13 = (long)(&data13);
    dat.addr14 = (long)(&data14);
    dat.addr15 = (long)(&data15);

    data0 =   'S';
    data1 =   'u';
    data2 =   'p';
    data3 =   'e';
    data4 =   'r';
    data5 =   ' ';
    data6 =   'H';
    data7 =   ' ';
    data8 =   '7';
    data9 =   '0';
    data10 =  '5';
    data11 =  '5';
    data12 =  ' ';
    data13 =  ' ';
    data14 =  ' ';
    data15 =  ' ';
}
```

仕様

（1）図1に示すように、1グループ（AN12～15の4チャネル）に入力される電圧をSH7055のA/D変換器を使用し、測定します。
（2）図2に示すようにDMACを用いてA/Dデータレジスタ内の測定結果を、ADDR12～15の8バイト毎に4回（計32バイト）RAMに格納します。
4回転送後は転送元（アドレスリロード機能を使用）、転送先を最初のアドレスに戻し、再び転送を繰り返します。転送条件を表1に示します。
（3）A/D変換器の起動はATUによるタイマ割り込みを使用し、起動周期は5msです。
（4）入力する電圧は0～5Vの範囲です。

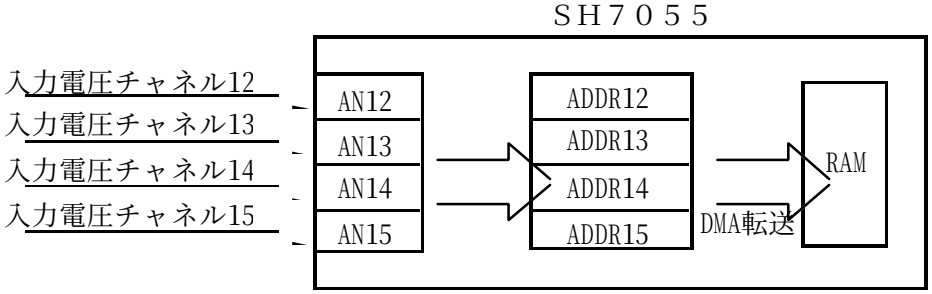


図1　SH7055による電圧の測定ブロック図

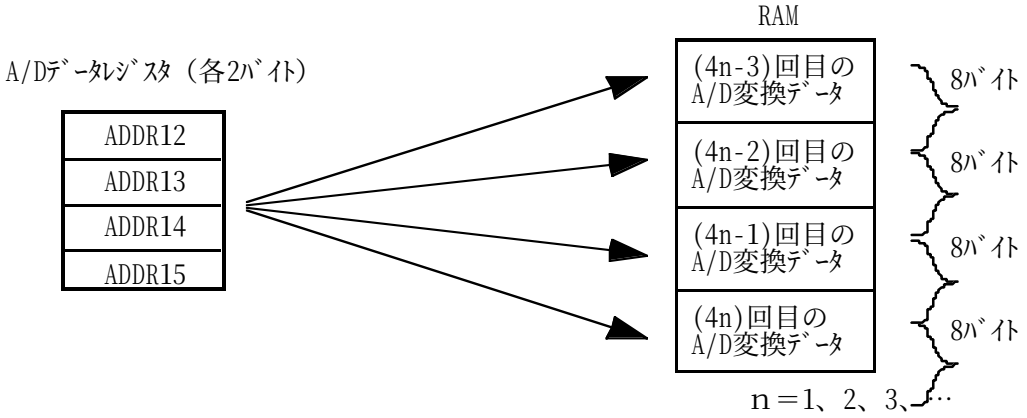


図2　DMACを用いたデータ転送

表1　DMA転送条件

条件項目	内容
DMACチャネル	チャネル2
転送元	内蔵A/D変換器
転送先	内蔵RAM
転送回数	16回（リロード回数4回）
転送元アドレス	増加
転送先アドレス	増加
転送要求元	内蔵A/D変換器
バスモード	バースト
転送単位	ワード

使用機能説明

表２、表３に本タスク例の機能割付を示します。表２、表３に示すようにＳＨ７０５５に内蔵しているDMAC、A/D変換の機能を割付、A/D変換及び変換データのRAMへの転送を行ないます。

表２　ＤＭＡＣ機能割付

DMACレジスタ	機　　能
SAR2	転送元アドレスを設定する。
DAR2	転送先アドレスを設定する。
TCR2	転送回数を設定する。
CHCR2	DMACの動作モード、転送方法等を設定する。
DMAOR	DMACの実行するチャネルの優先順位を設定する。

表３　Ａ／Ｄ変換機能割付

A／D変換器レジスタ	機　　能
ADCSR1	A/D変換のモード（単一モード／スキャンモード）、 測定端子の選択を設定する。
ADCR1	A/D変換器のクロックの選択、測定の開始及び終了を設定する。
ADDR12～15	A/D変換の結果を設定する。

動作説明

図３に動作原理を示します。図３に示すように、ＳＨ７０５５のハードウェア処理及びソフトウェア処理により4チャネルのA/D変換及び変換データのRAMへの転送を行います。

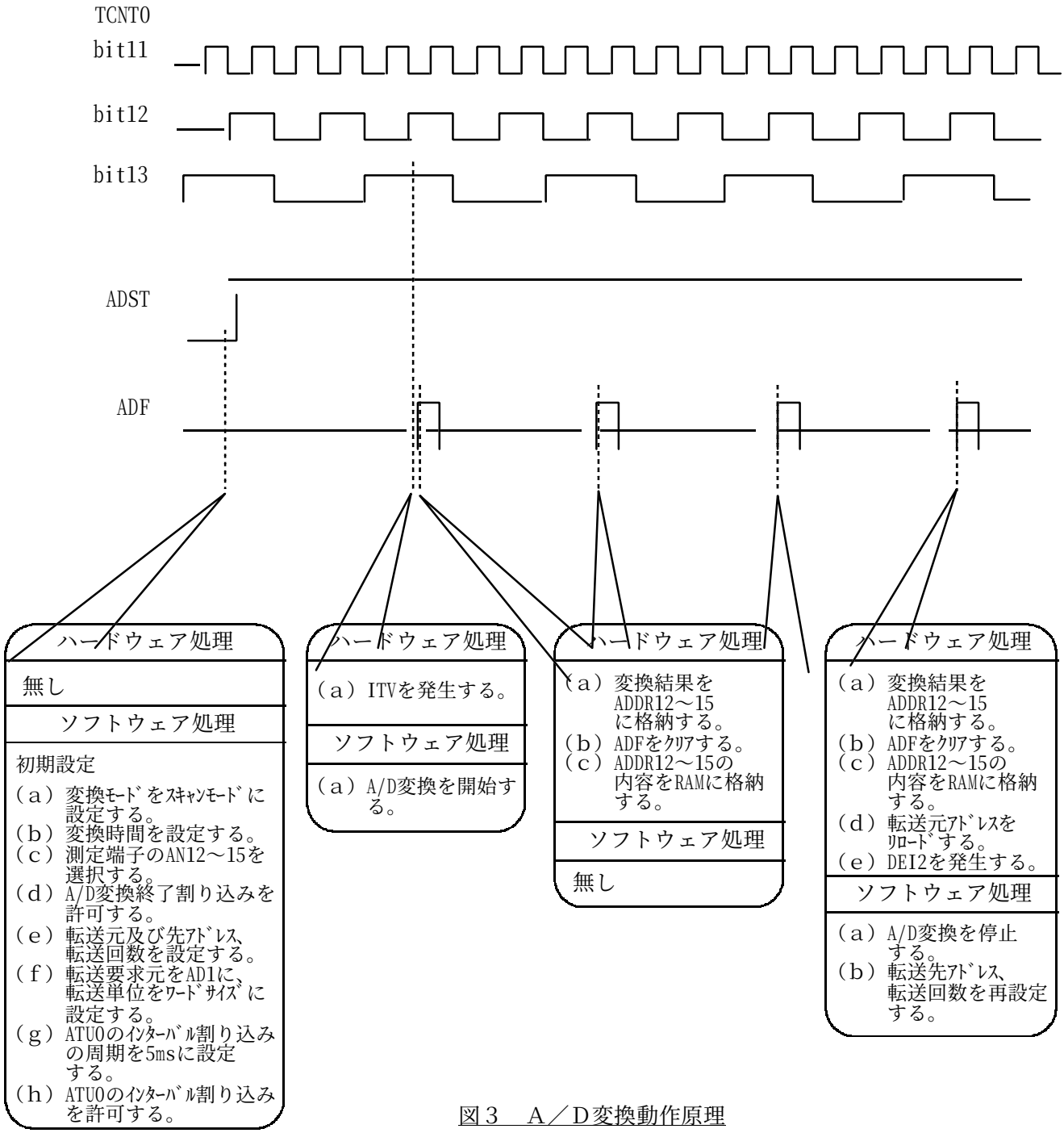


図３　Ａ／Ｄ変換動作原理

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機　　能
メインルーチン	dma_admn	A/D変換器及びDMACの初期設定を行なう。
転送終了	dma_ad	DEI2で起動し、転送先アドレス、転送回数を再設定する。

（2）引数の説明

ラベル名	機　　能	データ長	使用モジュール
ad_dat0～3	AN12～15に入力した電圧のA/D変換（スキャンモード（4n-3）回目）結果を格納する。	unsigned short	無し
ad_dat4～7	AN12～15に入力した電圧のA/D変換（スキャンモード（4n-2）回目）結果を格納する。		
ad_dat8～11	AN12～15に入力した電圧のA/D変換（スキャンモード（4n-1）回目）結果を格納する。		
ad_dat12～16	AN12～15に入力した電圧のA/D変換（スキャンモード（4n）回目）結果を格納する。		

（3）使用内部レジスタ説明

レジスタ名	機　　能	設定値	使用モジュール名
DMAC2.SAR	ADDR12のアドレスを設定する。	&AD.ADDR12	メインルーチン
DMAC2.DAR	転送先RAM先頭アドレスを設定する。	&ad_dat0	メインルーチン 転送終了
DMAC2.DMATCR	転送回数（16回）を設定する。	0x00000010	
DMAC2.CHCR	DMACの動作モード、転送方法等を設定する。	0x010C111D	メインルーチン
DMACC.DMAOR	DMACの起動を設定する。	0x0001	
INTC.IPRC	DMA2,ATU01の割り込み優先レベルを13、15に設定する。	0x0DF0	
INTC.IPRJ	A/D1の割り込み優先レベルを14に設定する。	0x00E0	
ATUC.TSTR1	TCNT0のカウントを開始する。	0x0001	
ATUC.PSCR1	TCNT0クロックをφ/6に設定する	0x05	
ATU0.ITVRR2B	TCNT0のビット13でインターバル割り込みを発生する。	0x08	
AD.ADCSR1	A/D変換器の変換モードをスキャンモード、A/D変換終了割り込み許可、測定端子AN12～15に設定する。	0x53	
AD.ADCR1	A/D変換器の変換時間を268ステートに設定する。	0x5F	

（4）使用ＲＡＭ

本タスクでは引数以外のＲＡＭは使用していません。

プログラムリスト

```
/************************************************************************/
/*            DMACによるA/D変換データ転送                               */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ        */
/************************************************************************/
/*            関数プロトタイプ宣言                                      */
/************************************************************************/
void main( void );
/************************************************************************/
/*            変数定義                                                  */
/************************************************************************/
volatile  struct add
{
    short  dat0;                                /* A/D変換ﾃﾞｰﾀ0          */
    short  dat1;                                /* A/D変換ﾃﾞｰﾀ1          */
    short  dat2;                                /* A/D変換ﾃﾞｰﾀ2          */
    short  dat3;                                /* A/D変換ﾃﾞｰﾀ3          */
    short  dat4;                                /* A/D変換ﾃﾞｰﾀ4          */
    short  dat5;                                /* A/D変換ﾃﾞｰﾀ5          */
    short  dat6;                                /* A/D変換ﾃﾞｰﾀ6          */
    short  dat7;                                /* A/D変換ﾃﾞｰﾀ7          */
    short  dat8;                                /* A/D変換ﾃﾞｰﾀ8          */
    short  dat9;                                /* A/D変換ﾃﾞｰﾀ9          */
    short  dat10;                               /* A/D変換ﾃﾞｰﾀ10         */
    short  dat11;                               /* A/D変換ﾃﾞｰﾀ11         */
    short  dat12;                               /* A/D変換ﾃﾞｰﾀ12         */
    short  dat13;                               /* A/D変換ﾃﾞｰﾀ13         */
    short  dat14;                               /* A/D変換ﾃﾞｰﾀ14         */
    short  dat15;                               /* A/D変換ﾃﾞｰﾀ15         */
};
#define ad (*(struct add *)0xFFFFC000)
/************************************************************************/
/*            メインルーチン                                                */
/************************************************************************/
void main( void )
{
    ATUC.PSCR1 = 0x05;              /* ﾌﾟﾘｽｹｰﾗ1段目　φ/6                */
    ATU0.ITVRR2B = 0x08;            /* TCNT0 13ﾋﾞｯﾄ目で割り込み発生(4.92ms) */
    ATUC.TSTR1 = 0x0001;            /* ﾁｬﾈﾙ0　ｶｳﾝﾄｽﾀｰﾄ                   */
    AD.ADCSR1 = 0x53;               /* ｽｷｬﾝﾓｰﾄﾞ,測定端子AN12～15          */
    AD.ADCR1 = 0x5F;                /* 外部ﾄﾘｶﾞによるA/D変換禁止,268ｽﾃｰﾄ   */
    DMAC2.SAR = (long)(&AD.ADDR12); /* 転送元ｱﾄﾞﾚｽ：A/Dﾃﾞｰﾀﾚｼﾞｽﾀ          */
    DMAC2.DAR = (long)(&ad.dat0);   /* 転送先ｱﾄﾞﾚｽ：RAM                   */
    DMAC2.DMATCR = 0x00000010;      /* 転送回数：16回                     */
    DMAC2.CHCR = 0x010C111D;        /* ﾘｿｰｽｱﾄﾞﾚｽﾘﾛｰﾄﾞ,ｿｰｽ/ﾃﾞｨｽﾄﾈｰｼｮﾝ増加   */
    DMACC.DMAOR = 0x0001;           /* DMAC起動許可                       */
    INTC.IPRC = 0x0DF0;             /* DMA2,ATU01割込優先ﾚﾍﾞﾙを13,15に設定 */
    INTC.IPRJ = 0x00E0;             /* A/D1割り込み優先ﾚﾍﾞﾙを14に設定      */
    set_imask(0x0);                 /* 割り込み許可                       */
    while(1);                       /* 無限ﾙｰﾌﾟ（割り込み待ち）           */
}
```

プログラムリスト

```
/************************************************************************/
/*              DMA転送終了割り込みルーチン                             */
/************************************************************************/
#pragma interrupt( dma_ad )
void  dma_ad( void )
{
    AD.ADCR1 &= 0xdf;               /* A/D変換停止                      */
    DMAC2.CHCR &= 0xFFFFFFFC;               /* TEﾌﾗｸﾞｸﾘｱ                */
    DMAC2.DAR = (long)(&ad.dat0);   /* 転送先ｱﾄﾞﾚｽ：RAM                */
    DMAC2.DMATCR = 0x00000010;      /* 転送回数：16回                   */
    DMAC2.CHCR |= 0x00000001;               /* DEﾌﾗｸﾞｾｯﾄ                */
}
/************************************************************************/
/*              インターバル割り込みルーチン                            */
/************************************************************************/
#pragma interrupt( int5ms )
void int5ms( void )
{
    ATU0.TSR0 &= 0xFF7F;
    AD.ADCR1 |= 0x20;               /* A/D変換開始                      */
}
#pragma interrupt( ad1 )
void  ad1( void )
{
}
```

2.5　パルスの周期測定（32ビット）	MCU	SH７０５５	使用機能	ＡＴＵ-Ⅱ

仕様

（１）図１に示すように、パルスの周期を測定し、結果をRAMに設定します。

（２）パルスの周期は5.0μsから429sまで100ns単位で測定可能です。

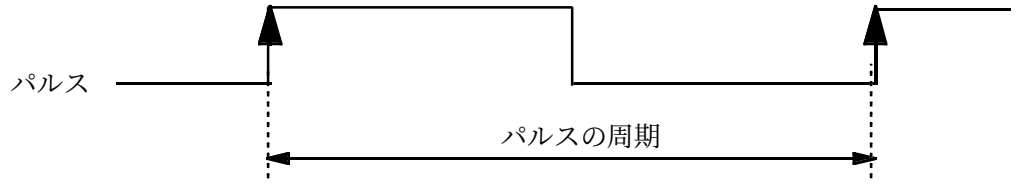


図１　パルス周期測定タイミング

使用機能説明

表１、表2に本タスク例の機能割り付けを示します。本タスク例は表１、表2に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、PFCの機能を割り付け、パルスの周期を測定します。

表１　　ＡＴＵ-Ⅱ機能割り付け

ＡＴＵ-Ⅱの内蔵機能		機能
端子	TIOA	測定するパルスを入力します。
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの設定をします。
	TIOR0	ATU-Ⅱチャネル0のエッジ検出を選択します。
	TIER0	ATU-Ⅱチャネル0の割り込み要求の許可/禁止を設定します。
	TSTR1	ATU-Ⅱチャネル0のカウンタ動作を設定します。
	ICR0A	入力パルスの立ち上がり時のカウント値を検出します。

表2　　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PAIOR	端子の入出力を設定します。
PACRL	端子の機能を選択します。

2.5　パルスの周期測定（32ビット）	MCU	ＳＨ７０５５	使用機能	ＡＴＵ-Ⅱ

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスの周期測定を行います。

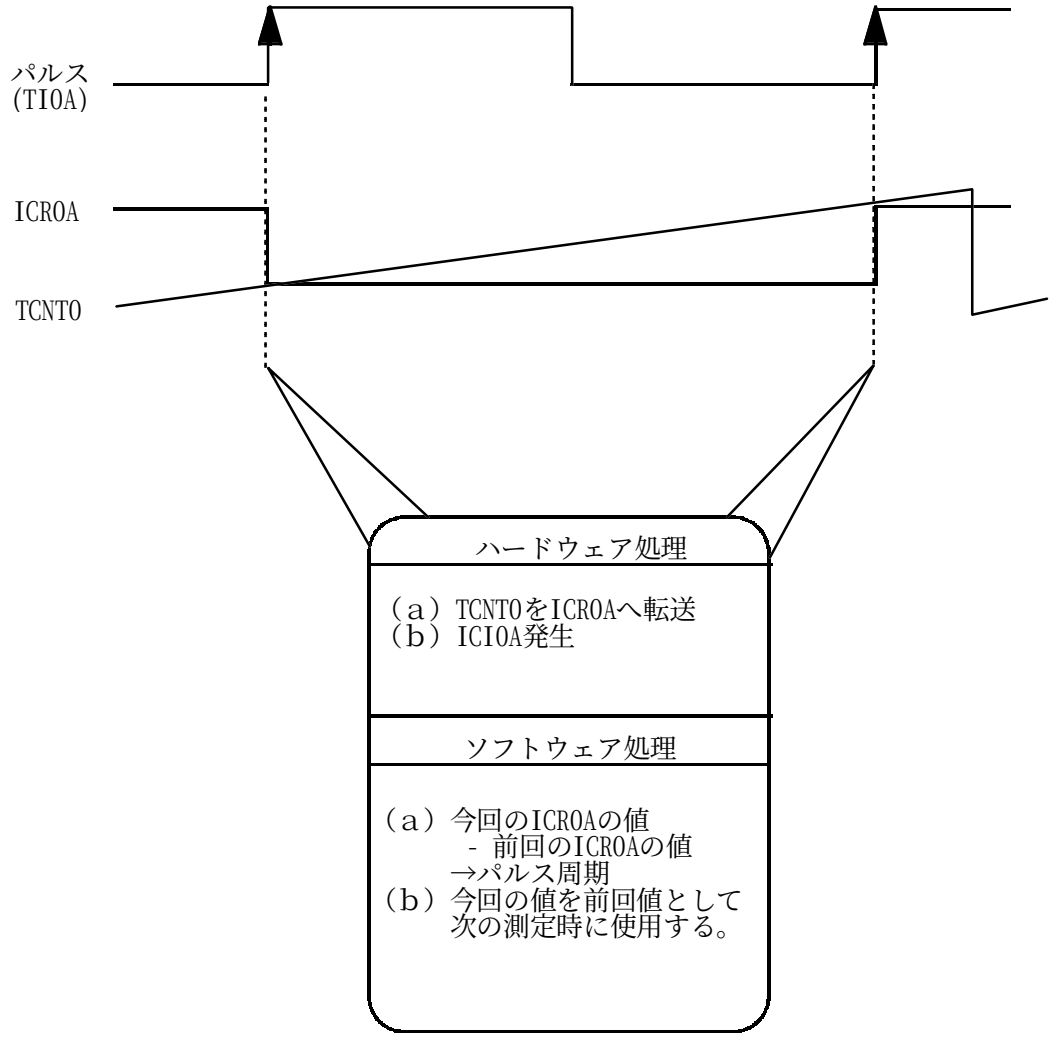


図２　パルス周期計測動作原理

2.5　パルスの周期測定（32ビット）	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱの初期設定を行ないます。
パルスの周期測定	ICI0A	ICF0Aにより起動し、ICR0Aの値からパルスの周期を測定します。

（2）使用変数の説明

ラベル名	機能	データ長	使用モジュール名
plsw	パルスの周期に相当するタイマ値を設定します。 パルスの周期は以下の式にて求まります。 パルスの周期(ns)=タイマ値×Pφ周期(20MHz動作時50ns) ×2（プリスケーラの分周比）	unsigned long	パルスの周期測定
work	インプットキャプチャ値を格納します。		

（3）使用内部レジスタの説明

レジスタ名	機能	設定値	使用モジュール名
PA.PAIOR	ポートAを入力端子に設定します。	0x0000	メインルーチン
PA.PACRL	PA0端子マルチプレクスをTI0Aに設定します。	0x0001	
ATUC.PSCR1	ATU-Ⅱのプリスケーラの1段目をPφ/2に設定します。	0x01	
ATU0.TIOR0	ATU-Ⅱチャネル0を立ち上がりエッジでICR0Aへインプットキャプチャするように設定します。	0x01	
ATUC.TSTR1	ATU-Ⅱチャネル0のカウント開始を設定します。	0x01	
ATU0.TIER0	ATU-Ⅱチャネル0のICF0Aによる割り込み要求を許可します。	0x0001	
INTC.IPRC	ATU02の割り込み優先レベルを15に設定します。	0x000F	

プログラムリスト

```
/*****************************************************************************/
/*              パルス周期測定(32bit)                                        */
/*****************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ    */
/*****************************************************************************/
/*              関数プロトタイプ宣言                                         */
/*****************************************************************************/
void main(void);
/*****************************************************************************/
/*              変数定義                                                     */
/*****************************************************************************/
#define plsw  (*(unsigned long *)0xFFFFC000)       /* ﾊﾟﾙｽ幅             */
#define work  (*(unsigned long *)0xFFFFC004)       /* ﾊﾟﾙｽ幅ﾜｰｸ用        */
/*****************************************************************************/
/*              メインルーチン                                               */
/*****************************************************************************/
void main(void)
{
        PA.PAIOR = 0x0000;              /* ﾎﾟｰﾄA入出力設定                */
        PA.PACRL = 0x0001;              /* TIOA(PA0)                     */
        ATUC.PSCR1 = 0x01;              /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/2(100ns)     */
        ATU0.TIOR0 = 0x01;              /* 立ち上がりｴｯｼﾞでｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬ  */
        ATUC.TSTR1 = 0x01;              /* ﾁｬﾈﾙ0 32ﾋﾞｯﾄｶｳﾝﾀｽﾀｰﾄ          */
        ATU0.TIER0 = 0x0001;            /* ICF0Aによる割り込み要求を許可 */
        INTC.IPRC = 0x000F;             /* ATU02割り込みﾚﾍﾞﾙ設定         */
        set_imask(0x0);                 /* 割り込みﾏｽｸﾚﾍﾞﾙ設定           */
        while(1);                       /* 無限ﾙｰﾌﾟ(割り込み待ち)         */
}
/*****************************************************************************/
/*              パルスの周期計測ルーチン                                     */
/*****************************************************************************/
#pragma interrupt(ICI0A)
void ICI0A(void)
{
        ATU0.TSR0 &= 0xFFFE;            /* ｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬ0Aﾌﾗｸﾞｸﾘｱ        */
        plsw = ATU0.ICR0A - work;       /* ﾊﾟﾙｽ周期演算                  */
        work = ATU0.ICR0A;              /* ｷｬﾌﾟﾁｬ値保存                  */

}
```

2.6　パルスのHigh幅測定(16ビット)	MCU	ＳＨ７０５５	使用機能	ＡTU−Ⅱ

仕様

（１）図１に示すように、パルスのHigh幅を測定し、結果をRAMに設定します。

（２）パルスの周期は5μsから13.1msまで200ns単位で測定可能です。

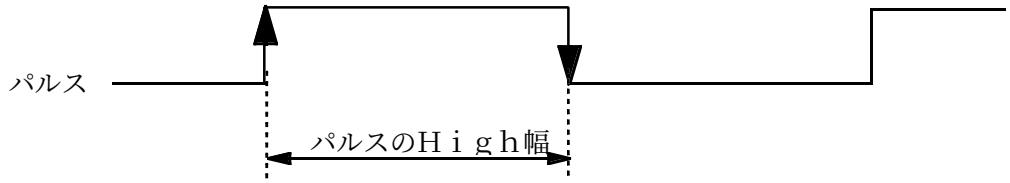


図１　パルスＨｉｇｈ幅測定タイミング

使用機能説明

表１、表2に本タスク例の機能割り付けを示します。本タスク例は表１、表2に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、PFCの機能を割り付け、パルスの周期測定します。

表１　　ＡＴＵ-Ⅱ機能割り付け

ＡＴＵ-Ⅱの内蔵機能		機能
端子	TIO11A,11B	PWMを出力します。
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの設定をします。
	TCR11	ATU-Ⅱチャネル11のプリスケーラの設定をします。
	TIOR11	ATU-Ⅱチャネル11のレジスタの機能を選択します。
	TSTR3	ATU-Ⅱチャネル11のカウンタの動作を設定します。
	TIER11	ATU-Ⅱチャネル11の割り込み要求の許可/禁止を設定します。
	GR11A	TCNT11をキャプチャした値を検出します。

表2　　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PLIOR	端子の入出力方向を選択します。
PLCRL	端子の機能を選択します。

2.6　パルスのHigh幅測定(16ビット)	MCU	ＳＨ７０５５	使用機能	ＡTU−Ⅱ

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスのHigh幅の測定を行います。

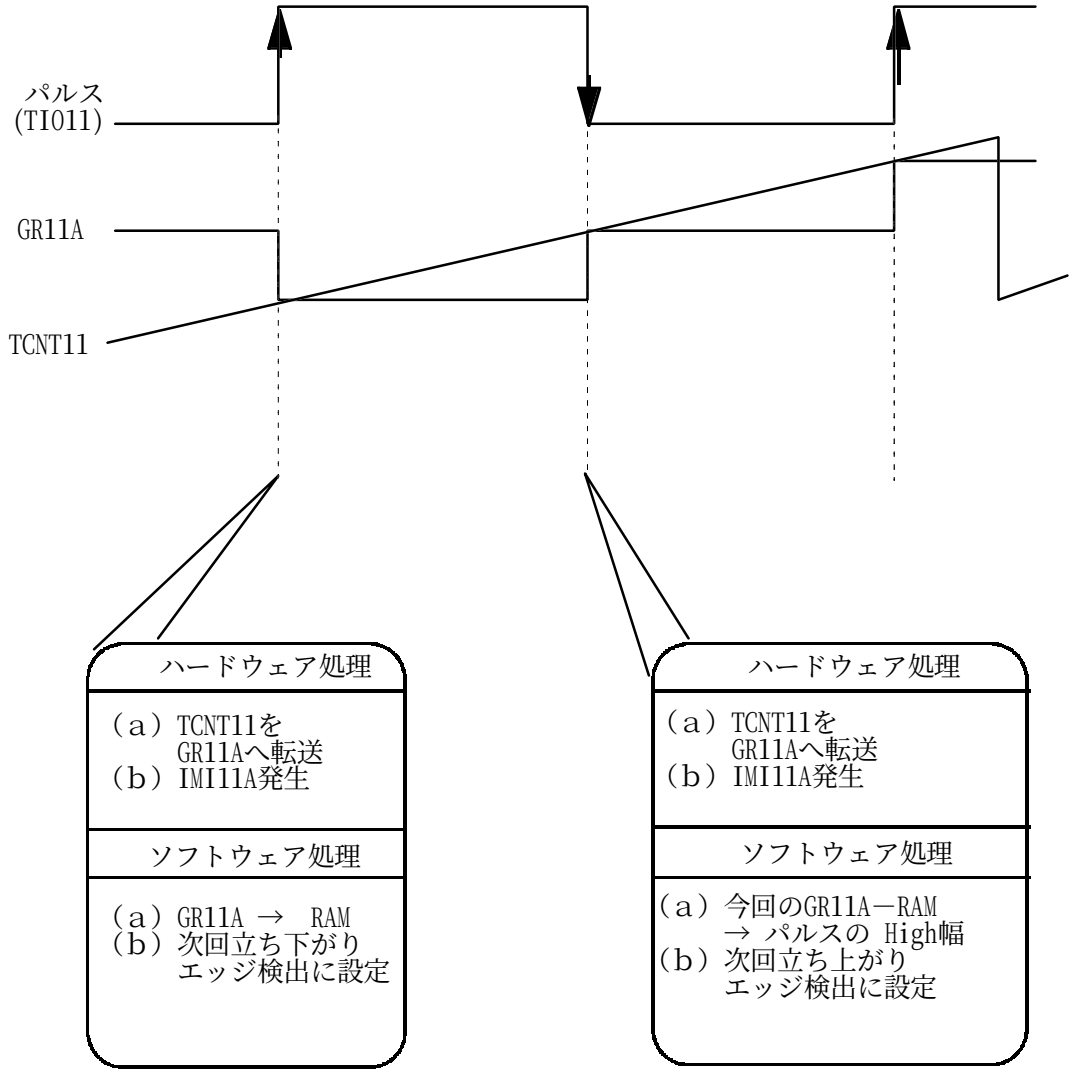


図２　パルスのＨｉｇｈ幅計測動作原理

2.6　パルスのHigh幅測定(16ビット)	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱの初期設定を行ないます。
パルスの High幅測定	IMI11A	IMF11Aにより起動し、GR11Aの値からパルスの幅を測定します。

（2）使用変数の説明

ラベル名	機能	データ長	使用モジュール名
shu_16	パルスの周期に相当するタイマ値を設定します。 パルスの周期は以下の式にて求めます。 パルスの周期(ns)=タイマ値×Pφ周期(20MHz動作時50ns) ×4（プリスケーラの分周比）	unsigned short	パルスの High幅測定
ref_cntw	測定パルスの立ち上がりエッジ検出時のTCNT11を格納します。		

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PL.PLIOR	TIO11A、11Bを入力端子に設定します。	0x0000	メインルーチン
PL.PLCRL	端子マルチプレクスをTIO11A、11Bの使用に設定します。	0x003C	
ATUC.PSCR1	ATU-Ⅱのプリスケーラ1段目をPφ/2に設定します。	0x01	
ATU11.TCR11	ATU-Ⅱチャネル11のプリスケーラをφ'/2に設定します。	0x01	
ATU11.TIOR11	ATU-Ⅱチャネル11を立ち上がりエッジでインプットキャプチャするように設定します。	0x05	
ATUC.TSTR3	ATU-Ⅱチャネル11のカウント開始を設定します。	0x01	
ATU11.TIER11	ATU-Ⅱチャネル11のCMF11Aによる割り込み要求を許可します。	0x0001	
INTC.IPRJ	ATU11の割り込み優先レベルを15に設定します。	0xF000	
ATU11.GR11A	測定パルスの立ち上がりエッジ検出時のTCNT11が設定されます。	-	パルスのHigh幅測定

プログラムリスト

```
/*****************************************************************************/
/*            High幅計測（１６ビット）                                       */
/*****************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/*****************************************************************************/
/*            関数プロトコル宣言                                             */
/*****************************************************************************/
void main( void );
/*****************************************************************************/
/*            変数定義                                                       */
/*****************************************************************************/
#define shu_16   (*(unsigned short *)0xFFFFC000)   /* ﾊﾟﾙｽ幅(下位)       */
#define ref_cntw (*(unsigned short *)0xFFFFC002)   /* ﾊﾟﾙｽ幅ﾜｰｸ用        */
/*****************************************************************************/
/*            メインルーチン                                                 */
/*****************************************************************************/
void main( void ){
       PL.PLIOR   = 0x0000;          /* ﾎﾟｰﾄL入出力設定                    */
       PL.PLCRL   = 0x0014;          /* TIO11A,11B(PL1,2)                  */
       ATUC.PSCR1 = 0x01;            /* ﾌﾟﾘｽｹｰﾗ1段目Pφ/2(100ns) ch11      */
       ATU11.TCR11 = 0x01;           /* ﾌﾟﾘｽｹｰﾗ2段目φ'/2(200ns)           */
       ATU11.TIOR11 = 0x05;          /* TIOA11端子の立上りでｷｬﾌﾟﾁｬ         */
       ATUC.TSTR3 = 0x01;            /* ﾁｬﾈﾙ11ｶｳﾝﾄｽﾀｰﾄ                     */
       ATU11.TIER11 = 0x0001;        /* IMF11Aによる割込み許可             */
       INTC.IPRJ = 0xF000;           /* ATU11の割り込み優先ﾚﾍﾞﾙを15に設定*/
       set_imask(0x0);               /* 割り込み許可                       */
       while(1);                     /* 無限ﾙｰﾌﾟ（割り込み待ち）           */
}
/*****************************************************************************/
/*            パルスのＨｉｇｈ幅測定ルーチン                                 */
/*****************************************************************************/
#pragma interrupt( IMI11A )
void IMI11A( void )
{
      ATU11.TSR11 &= 0xfffe;                /* ｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬA ﾌﾗｸﾞｸﾘｱ      */
      if((ATU11.TIOR11 & 0x05) == 0x05)     /* 立上りｴｯｼﾞか？              */
      {
           ATU11.TIOR11 = 0x06;             /* 次回立下りｴｯｼﾞでﾁｬﾌﾟﾁｬ     */
             }
       else
      {
           ATU11.TIOR11 = 0x05;             /* 次回立上りｴｯｼﾞでﾁｬﾌﾟﾁｬ     */
            shu_16  = ATU11A.GR11A - ref_cntw;    /* 周期算出              */
             }
       ref_cntw = ATU11A.GR11A;             /* ｷｬﾌﾟﾁｬ値保存               */
   }
```

2.7　イベント周期測定	MCU	SH7055	使用機能	ATU-Ⅱ

仕様

（1）図1に示すように、イベントの周期を測定し、結果をRAMに設定します。

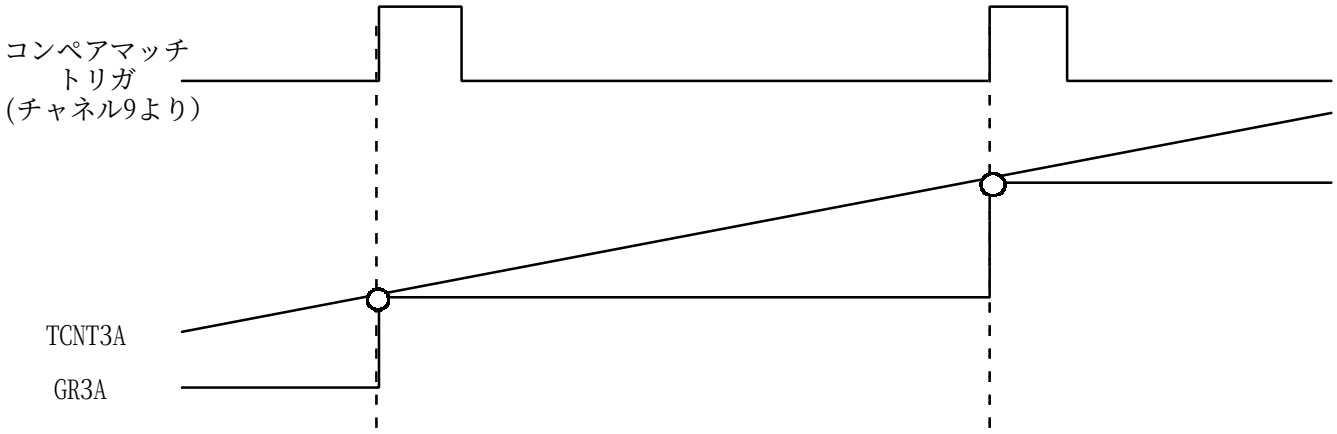


図1　パルス周期測定タイミング

使用機能説明

表1、表2に本タスク例の機能割り付けを示します。本タスク例は表1、表2に示すようにSH7055に内蔵されているATU-Ⅱ、PFCの機能を割り付け、イベント周期の測定を行ないます。

表1　ATU-Ⅱ機能割り付け

ATU-Ⅱ内蔵機能		機能
端子	TIO9A	パルスを出力します。
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの設定をします。
	TCR3	ATU-Ⅱチャネル3のプリスケーラを設定します。
	TMDR	ATU-Ⅱチャネル3の機能を選択します。
	TIOR3A	ATU-Ⅱチャネル3のレジスタの機能を設定します。
	GR3A	TCNT3をキャプチャした値が設定されます。
	TCR9A	ATU-Ⅱチャネル9のコンペアマッチ時の機能を設定します。
	GR9A	周期を設定します。
	TSTR1	ATU-Ⅱチャネル3のカウンタ動作を設定します。
	TIER9	ATU-Ⅱチャネル9の割込み要求の許可/禁止を設定します。

表2　PFC機能割り付け

PFCレジスタ	機能
PJIOR	端子の入出力方向を設定します。
PJCRL	端子の機能を設定します。

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりイベントの周期測定を行います。

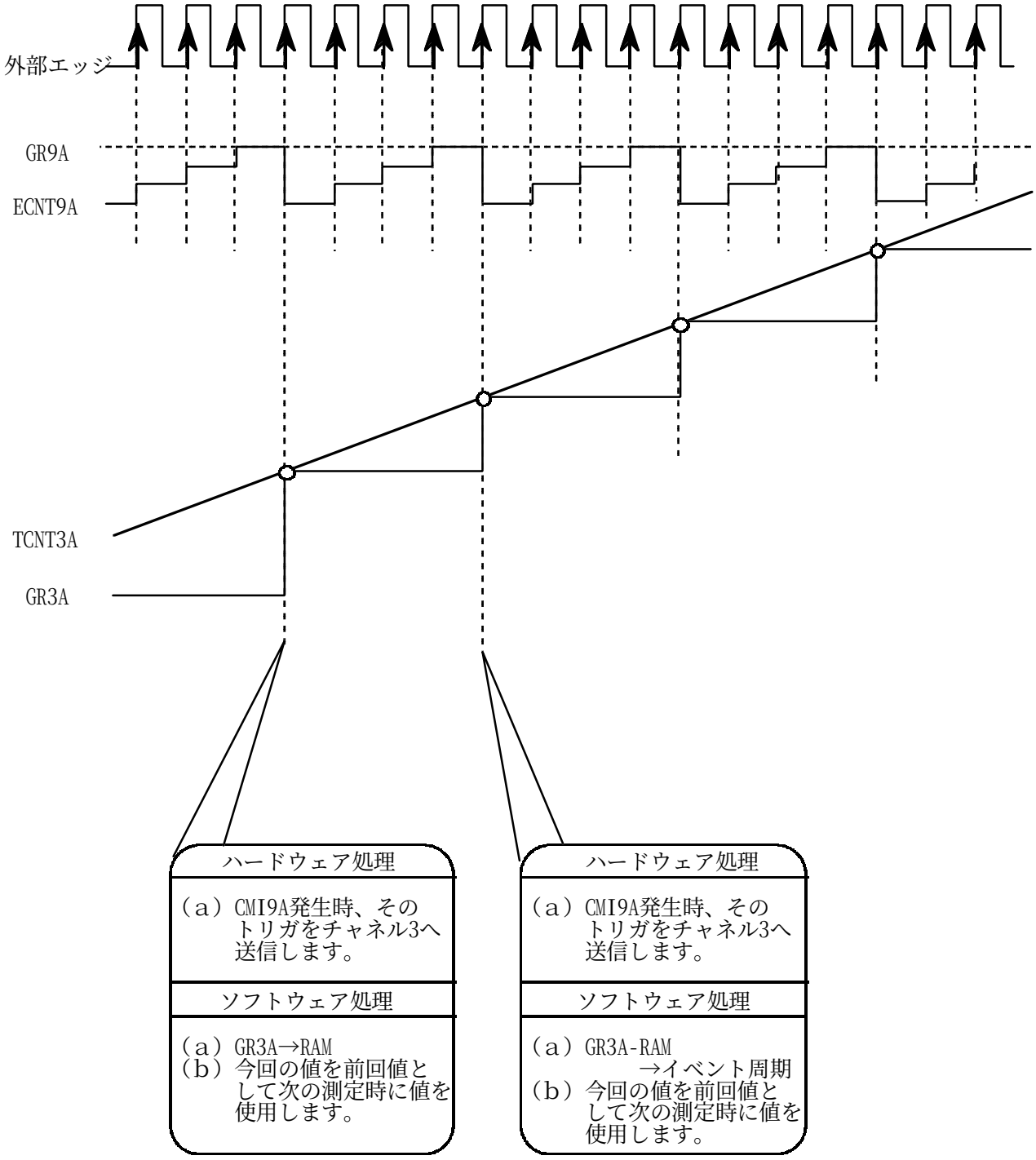


図２　パルス周期計測動作原理

2.7　イベント周期測定	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱ及びRAMの初期設定を行ないます。
イベント周期 計測ルーチン	CVI9A	CVF9Aにより起動し、イベント周期の計測を行ないます。

（2）使用変数の説明

ラベル名	機能	データ長	使用モジュール名
work	キャプチャ値を格納します。	unshigned　short	イベント周期 計測ルーチン
data	イベント周期の演算結果を格納します。		

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PJ.PJIOR	PJ10を入力端子に設定します。	0x0000	メインルーチン
PJ.PJCRH	PJ10端子マルチプレクスをTI9Aに設定します。	0x0010	
ATUC.PACR1	ATU-Ⅱのプリスケーラの1段目をPφ/5に設定します。	0x04	
ATU3.TCR3	ATU-Ⅱチャネル3のプリスケーラをφ'/16に設定します。	0x04	
ATU3.TMDR	ATU-Ⅱチャネル3はIC/OC機能を設定します。	0x00	
ATU3.TIOR3A	ATU-Ⅱチャネル3のインプットキャプチャ禁止を設定します。	0x04	
ATU9.TCR9A	ATU-Ⅱチャネル9のコンペアマッチ時に、そのトリガを チャネル3へ送信するように設定します。	0x05	
ATU9.GR9A	周期を設定します。	0x03	
ATUC.TSTR1	ATU-Ⅱチャネル3のカウント開始を設定します。	0x10	
ATU9.TIER9	ATU-Ⅱチャネル3のCMF9Aによる割り込み要求を設定します。	0x0001	
INTC.IPRI	ATU91の割り込み優先レベルを15に設定します。	0xF000	
ATU3.GR3A	TCNT3をキャプチャした値が設定されます。	――	イベント周期 計測ルーチン

プログラムリスト

```
/************************************************************************/
/*                              イベント周期測定                        */
/************************************************************************/
#include <machine.h>                    /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ              */
#include "7055.h"                       /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ            */
/************************************************************************/
/*                          関数プロトコル宣言                          */
/************************************************************************/
void main(void);
/************************************************************************/
/*                              変数定義                                */
/************************************************************************/
#define work (*(unsigned short *)0xFFFFC000)        /* ﾜｰｸ用ﾚｼﾞｽﾀ       */
#define data (*(unsigned short *)0xFFFFC002)        /* ﾃﾞｰﾀ格納用ﾚｼﾞｽﾀ  */
/************************************************************************/
/*                              メインルーチン                          */
/************************************************************************/
void main(void)
{
        PJ.PJCRH = 0x0010;              /* TI9A機能選択                 */
        PJ.PJIOR = 0x0000;              /* PJ10を入力端子に設定         */
        ATUC.PSCR1  = 0x04;             /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/5(250ns)   */
        ATU3.TCR3   = 0x04;             /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/16(4μs)    */
        ATU3.TMDR   = 0x00;             /* IC/OC機能を選択              */
        ATU3.TIOR3A = 0x04;             /* ｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬ禁止 TIO3A       */
        ATU9.TCR9A  = 0x05;             /* A立上りｴｯｼﾞでｶｳﾝﾄ ch3ﾄﾘｶﾞ許可 */
        ATU9.GR9A   = 0x03;             /* 3ｶｳﾝﾄでｺﾝﾍﾟｱﾏｯﾁ              */
        ATUC.TSTR1  = 0x10;             /* ﾁｬﾈﾙ3 ｶｳﾝﾀｽﾀｰﾄ               */
        ATU9.TIER9  = 0x0001;           /* CMF9Aによる割り込み要求を許可*/
        INTC.IPRI = 0xF000;             /* ATU91割り込みﾚﾍﾞﾙ設定        */
        set_imask(0x0);                 /* 割り込み許可                 */
        while(1);                       /* 無限ﾙｰﾌﾟ(割り込み待ち)       */
}
/************************************************************************/
/*                          イベント周期計測ルーチン                    */
/************************************************************************/
#pragma interrupt(CVI9A)
void CVI9A(void)
{
        ATU9.TSR9 &= 0xFFFE;            /* ｺﾝﾍﾟｱﾏｯﾁﾌﾗｸﾞ9Aｸﾘｱ             */
        data = ATU3.GR3A - work;        /* ｲﾍﾞﾝﾄ周期算出                */
        work = ATU3.GR3A;               /* ｷｬﾌﾟﾁｬ値格納                 */
}
```

2.8　パルス出力（チャネル11）	MCU	SH7055	使用機能	ATU-Ⅱ

仕様

（1）図１に示すように、一定の周期でトグル出力します。

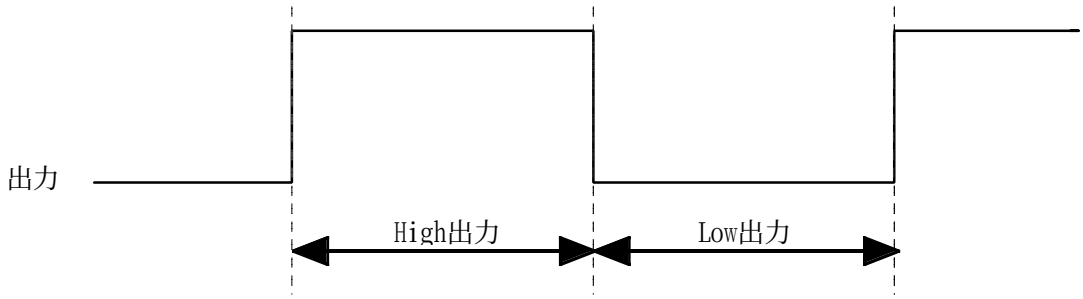


図１　トグル出力

使用機能説明

表１、表2に本タスク例の機能割り付けを示します。本タスク例は表１、表2に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、PFCの機能を割り付け、パルスを出力します。

表1　ＡＴＵ-Ⅱ機能割り付け

ＡＴＵ-Ⅱ内蔵機能		機能
端子	TIO11A,11B	パルスを出力します。
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの設定をします。
	TCR11	ATU-Ⅱチャネル11のプリスケーラを設定します。
	TIOR11	ATU-Ⅱチャネル11のレジスタの機能を設定します。
	GR11A	周期を設定します。
	TSTR3	ATU-Ⅱチャネル11のカウンタ動作を設定します。
	TIER11	ATU-Ⅱチャネル11の割込み要求の許可/禁止を設定します。

表２　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PLIOR	端子の入出力方向を設定します。
PLCRL	端子の機能を設定します。

動作原理

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスを出力します。

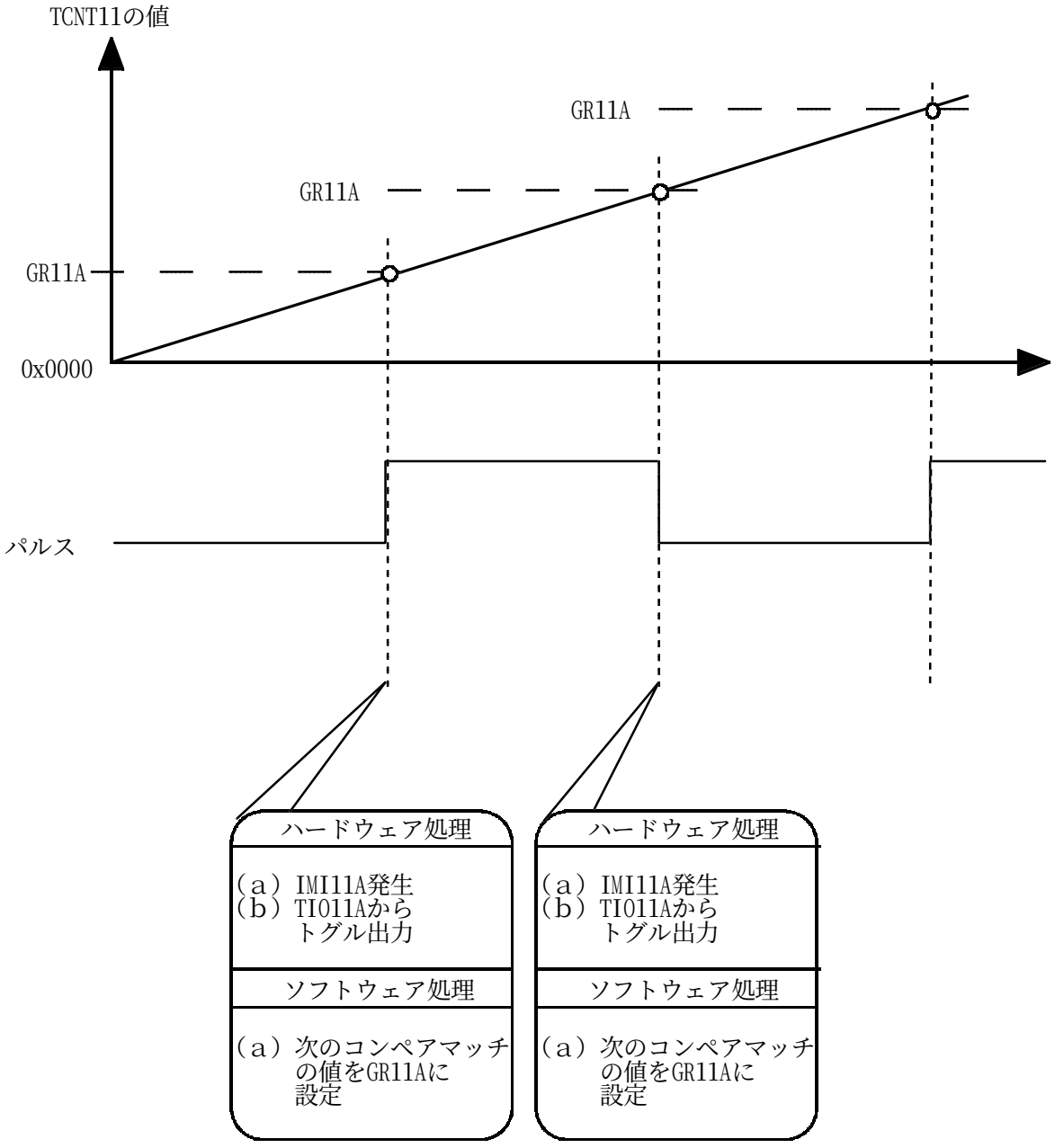


図２　パルス出力動作原理

2.8　パルス出力（チャネル11）	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱの初期設定を行ないます。
コンペアマッチ 割り込みルーチン	IMI11A	IMF11Aにより起動し、GR11Aの設定を行ないます。

（2）使用変数の説明

本タスクでは変数は使用しておりません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PL.PLIOR	PL1,2を出力端子に設定します。	0x0006	メインルーチン
PL.PLCRL	PL1,2端子マルチプレクスをTIO11A,11Bに設定します。	0x003C	
ATUC.PSCR1	ATU-Ⅱのプリスケーラの1段目をPφ/2に設定します。	0x02	
ATU11.TCR11	ATU-Ⅱチャネル11のプリスケーラをφ'/2に設定 します。	0x01	
ATU11.TIOR11	ATU-Ⅱチャネル11をコンペアマッチでトグル出力するように設定します。	0x03	
ATUC.TSTR3	ATU-Ⅱチャネル11のカウント開始を設定します。	0x01	
ATU11.TIER11	ATU-Ⅱチャネル11のIMF11Aによる割り込み要求を許可します。	0x0001	
INTC.IPRJ	ATU11の割り込み優先レベルを15に設定します。	0xF000	
ATU11.GR11A	周期を設定します。	0x0064	コンペアマッチ 割り込み

プログラムリスト

```
/************************************************************************/
/*              パルス出力                                              */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ       */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲ       */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main( void );
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
        PL.PLIOR    = 0x0006;       /* ﾎﾟｰﾄL入出力設定                  */
        PL.PLCRL    = 0x0014;       /* TIO11A,11B(PL1,2)                */
        ATUC.PSCR1 = 0x02;          /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/2(100ns)        */
        ATU11.TCR11 = 0x01;         /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/2(200ns)       */
        ATU11.TIOR11 = 0x03;        /* GR11Aのｺﾝﾍﾟｱﾏｯﾁでﾄｸﾞﾙ出力       */
        ATU11.GR11A=0x0064;         /* 20μs毎反転                      */
        ATUC.TSTR3 = 0x01;          /* ﾁｬﾈﾙ11ｶｳﾝﾄｽﾀｰﾄ                   */
        ATU11.TIER11 = 0x0001;      /* ｺﾝﾍﾟｱﾏｯﾁIMF11Aによる割込み許可  */
        INTC.IPRJ = 0xF000;         /* ATU11の割り込み優先ﾚﾍﾞﾙを15に設定*/
        set_imask(0x0);             /* 割り込み許可                     */
        while(1);                   /* 無限ﾙｰﾌﾟ（割り込み待ち）         */
}
/************************************************************************/
/*              コンペアマッチ割り込みルーチン                          */
/************************************************************************/
#pragma interrupt( IMI11A )
void IMI11A( void ){
        ATU11.TSR11 &= 0xFE;        /* ｺﾝﾍﾟｱﾏｯﾁﾌﾗｸﾞ11Aｸﾘｱ               */
        ATU11.GR11A += 0x0064;      /* 20μs毎反転                      */
}
```

2.9　パルス出力（チャネル3 High-Low切替出力）	MCU	ＳＨ７０５５	使用機能	ＡTU−Ⅱ

仕様

（1）図１に示すように、一定の周期でHigh、Lowを繰り返すパルスを出力します。

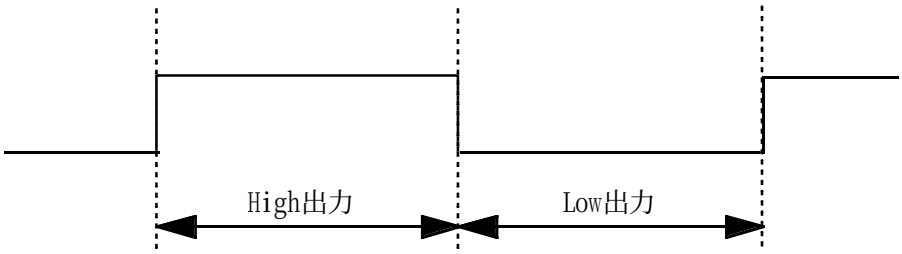


図１　パルス出力タイミング

使用機能説明

表１、表2に本タスク例の機能割り付けを示します。本タスク例は表１、表2に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、PFCの機能を割り付け、PWMを出力します。

表1　ＡＴＵ-Ⅱ機能割り付け

ＡＴＵ-Ⅱ内蔵機能		機能
端子	TIO3A～D	パルスを出力します。
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの設定をします。
	TCR3	ATU-Ⅱチャネル3のプリスケーラを設定します。
	TSTR	ATU-Ⅱチャネル3のカウンタ動作を設定します。
	TMDR	ATU-Ⅱチャネル3の機能の選択を行ないます。
	GR3A	PWMの周期を設定します。

表2　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PAIOR	端子の入出力方向を設定します。
PACRL	端子の機能を設定します。

動作原理

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスを出力します。

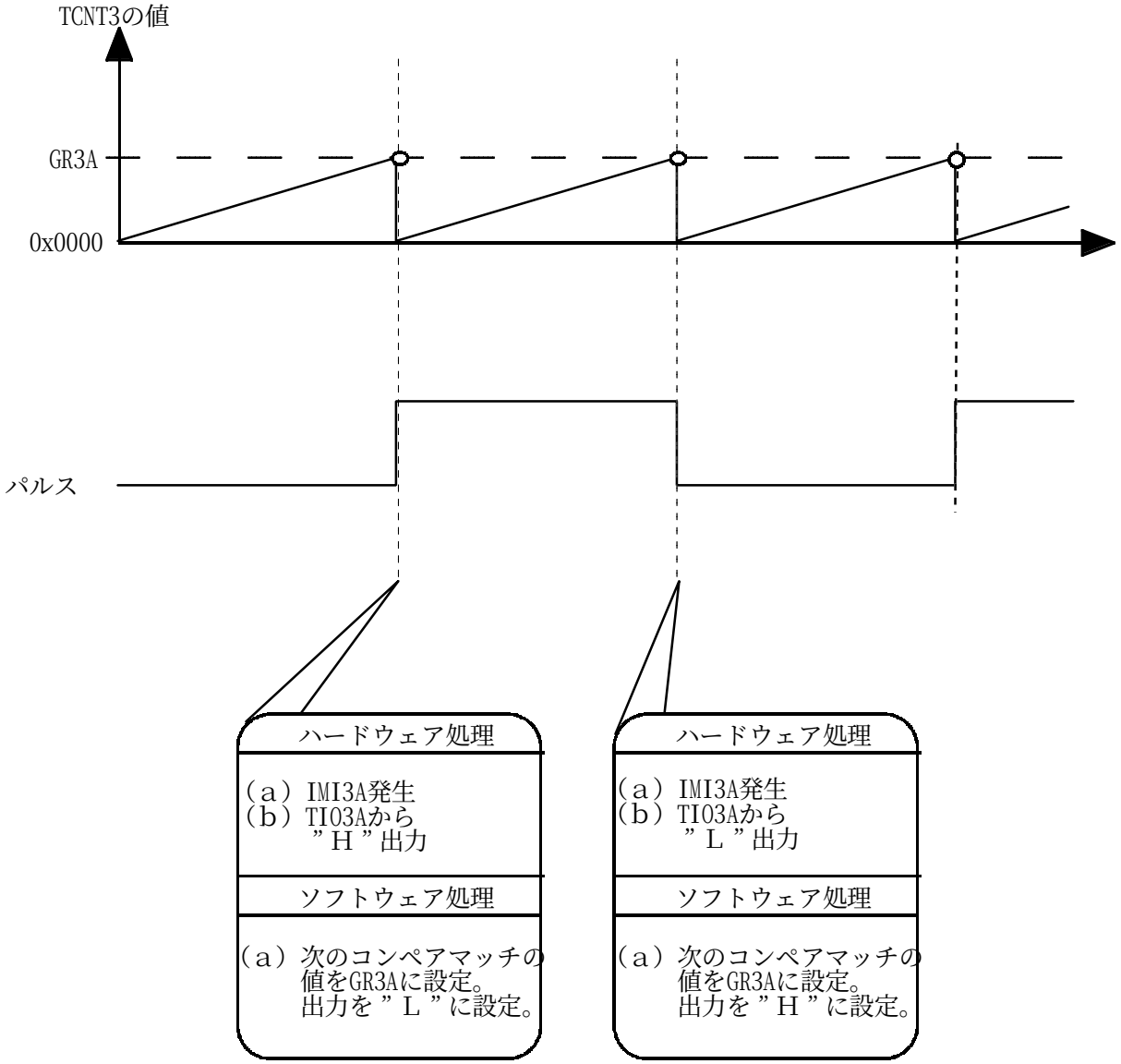


図２　PWM出力動作原理

2.9　パルス出力（チャネル3 High-Low切替出力）	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱの初期設定を行ないます。
パルス出力ルーチン	IMI3A	IMF3Aにより起動し、TIOR3Aの設定を行ないます。

（2）使用変数の説明

本タスクでは変数は使用しておりません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PA.PAIOR	PA4～7を出力端子に設定します。	0x00F0	メインルーチン
PA.PACRL	端子マルチプレクスをTIO3A～3D端子に設定します。	0x5500	
ATUC.PSCR1	ATU-Ⅱのプリスケーラ1段目をPφ/2に設定します。	0x01	
ATU3.TCR3	ATU-Ⅱチャネル3のプリスケーラをφ'/2に設定します。	0x01	
ATU3.TMDR	ATU-Ⅱチャネル3はIC/OC機能を設定します。	0x00	
ATU3.TIOR3A	ATU-Ⅱチャネル3をコンペアマッチでTCNTをクリア、1を出力するように設定します。	0x0A	メインルーチン パルス出力ルーチン
ATU3.GR3A	周期の設定をします。	0x00C8	メインルーチン
ATUC.TSTR1	ATU-Ⅱチャネル3のカウント開始を設定します。	0x10	
ATU3.TIER3	ATU-Ⅱチャネル3のコンペアマッチによる割り込み要求を許可します。	0x0001	
INTC.IPRF	ATU31の割り込み優先レベルを15に設定します。	0xF000	

プログラムリスト

```
/************************************************************************/
/*              パルス出力(High,Low切替出力)                            */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main(void);
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main(void)
{
        PA.PAIOR = 0x00F0;              /* ﾎﾟｰﾄA入出力設定               */
        PA.PACRL = 0x5500;              /* TIO3A～3D(PA4～7)             */
        ATUC.PSCR1  = 0x01;             /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/2(100ns)    */
        ATU3.TCR3   = 0x01;             /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/2(200ns)    */
        ATU3.TMDR   = 0x00;             /* IC/OC機能を選択               */
        ATU3.TIOR3A = 0x0A;             /* ｺﾝﾍﾟｱﾏｯﾁでTCNTｸﾘｱ､1出力       */
        ATU3.GR3A   = 0x00C8;           /* 周期 = 40μs                  */
        ATUC.TSTR1  = 0x10;             /* ﾁｬﾈﾙ3 ｶｳﾝﾀｽﾀｰﾄ                */
        ATU3.TIER3  = 0x0001;           /* ｺﾝﾍﾟｱﾏｯﾁによる割り込み許可   */
        INTC.IPRF   = 0xF000;           /* ATU31の割り込みﾚﾍﾞﾙを15に設定 */
        set_imask(0x0);                 /* 割り込み許可                  */
        while(1);                       /* 無限ﾙｰﾌﾟ(割込み待ち)          */
}
/************************************************************************/
/*              パルス出力ルーチン                                      */
/************************************************************************/
#pragma interrupt(IMI3A)
void IMI3A(void)
{
        ATU3.TSR3 &= 0xFFFE;            /* ｺﾝﾍﾟｱﾏｯﾁ割り込みﾌﾗｸﾞｸﾘｱ        */
        if (G_ATU3.TIOR3A == 0x0A)      /* 1出力か?                      */
        {
                G_ATU3.TIOR3A = 0x09; /* 次回ｺﾝﾍﾟｱﾏｯﾁで0出力           */
        }
        else
        {
                G_ATU3.TIOR3A = 0x0A; /* 次回ｺﾝﾍﾟｱﾏｯﾁで1出力           */
        }
}
```

2.10　パルス出力（トグル出力）	MCU	ＳＨ７０５５	使用機能	ＡＴU−Ⅱ

仕様

（1）図１に示すように、一定の周期でトグルを出力します。

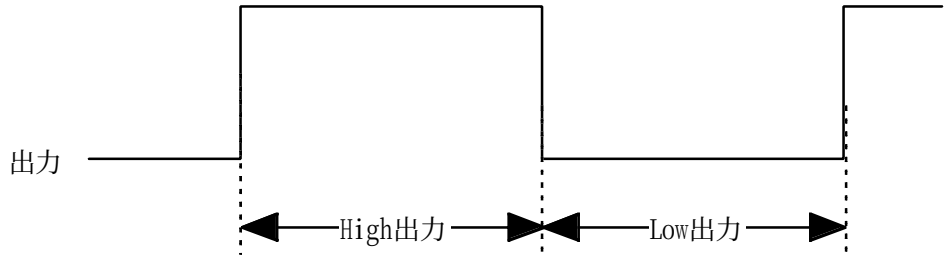


図1　トグル出力

使用機能説明

表１、表2に本タスク例の機能割り付けを示します。本タスク例は表１、表2に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、PFCの機能を割り付け、パルスの周期測定します。

表１　　ＡＴU-Ⅱ機能割り付け

ＡＴU-Ⅱの内蔵機能		機能
端子	TIO3A～3D	トグルを出力します。
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの設定をします。
	TCR3	ATU-Ⅱチャネル3のプリスケーラの設定をします。
	TIOR3A,3B	ATU-Ⅱチャネル3のレジスタの機能を選択します。
	TSTR1	ATU-Ⅱチャネル3のカウンタの動作を設定します。
	TMDR	ATU-Ⅱチャネル３の機能の選択を行います。
	GR3A～3D	周期を設定します。

表2　　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PAIOR	端子の入出力方向を選択します。
PACRL	端子の機能を選択します。

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりトグル出力の測定を行います。

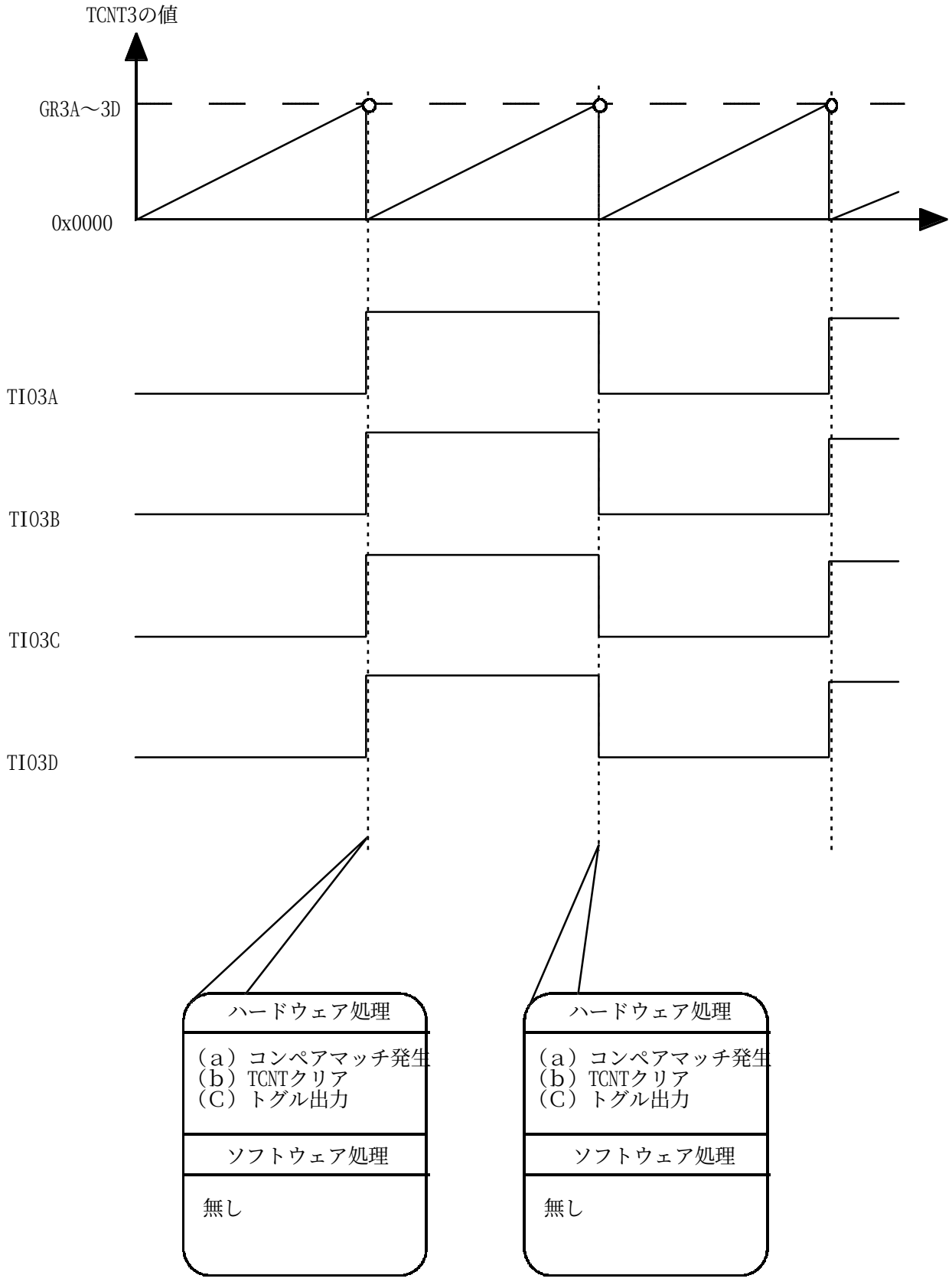


図２　トグル出力動作原理

2.10　パルス出力（トグル出力）	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱの初期設定を行います。

（2）使用変数の説明

本タスクでは変数は使用していません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PA.PAIOR	TIO3A～3Dを出力端子に設定します。	0x00F0	メインルーチン
PA.PACRL	端子マルチプレクスをTIO3A～Dの使用に設定します。	0x5500	
ATUC.PSCR1	ATU-Ⅱのプリスケーラ1段目をPφ/2に設定します。	0x01	
ATU3.TCR3	ATU-Ⅱチャネル3のプリスケーラをφ'/2に設定します。	0x01	
ATU3.TMDR	ATU-Ⅱチャネル3はIC/OC機能を設定します。	0x00	
ATU3.TIOR3A,3B	ATU-Ⅱチャネル3をコンペアマッチでTCNTクリア、トグル出力するように設定します。	0xBB	
ATUC.TSTR3	ATU-Ⅱチャネル3のカウント開始を設定します。	0x10	
ATU3.GR3A～3D	周期の設定します。	0x00C8	

プログラムリスト

```
/************************************************************************/
/*              パルス出力(トグル出力)                                  */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ    */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main(void);
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main(void)
{
        PA.PAIOR    = 0x00F0;       /* ﾎﾟｰﾄA入出力設定                  */
        PA.PACRL    = 0x5500;       /* TIO3A～3D(PA4～PA7)              */
        ATUC.PSCR1  = 0x01;         /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/2(100ns)       */
        ATU3.TCR3   = 0x01;         /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/2(200ns)      */
        ATU3.TMDR   = 0x00;         /* IC/OC機能を選択                  */
        ATU3.TIOR3A = 0xBB;         /* ｺﾝﾍﾟｱﾏｯﾁでTCNTｸﾘｱ､ﾄｸﾞﾙ出力      */
        ATU3.TIOR3B = 0xBB;         /* ｺﾝﾍﾟｱﾏｯﾁでTCNTｸﾘｱ､ﾄｸﾞﾙ出力      */
        ATU3.GR3A   = 0x00C8;       /* 周期 = 80μs                     */
        ATU3.GR3B   = 0x00C8;       /* 周期 = 80μs                     */
        ATU3.GR3C   = 0x00C8;       /* 周期 = 80μs                     */
        ATU3.GR3D   = 0x00C8;       /* 周期 = 80μs                     */
        ATUC.TSTR1  = 0x10;         /* ﾁｬﾈﾙ3 ｶｳﾝﾀｽﾀｰﾄ                   */
        while(1);                   /* 無限ﾙｰﾌﾟ(割り込み待ち)           */
}
```

2.11　ワンショットパルス出力（ｵﾌｾｯﾄ付き）	MCU	SH7055	使用機能	ATU-Ⅱ

仕様

（1）図1に示すように外部信号の立ち上がりに同期してワンショットパルスを出力します。
　　オフセット、パルス幅は内部クロックカウント値を設定します。
（2）外部信号の立ち上がりからのオフセットおよびパルス幅は以下に示す範囲で可変できます。
　　Pφ周期×1＜オフセット＜Pφ周期×65536　　＊注
　　200ns≦パルス幅＜13ms

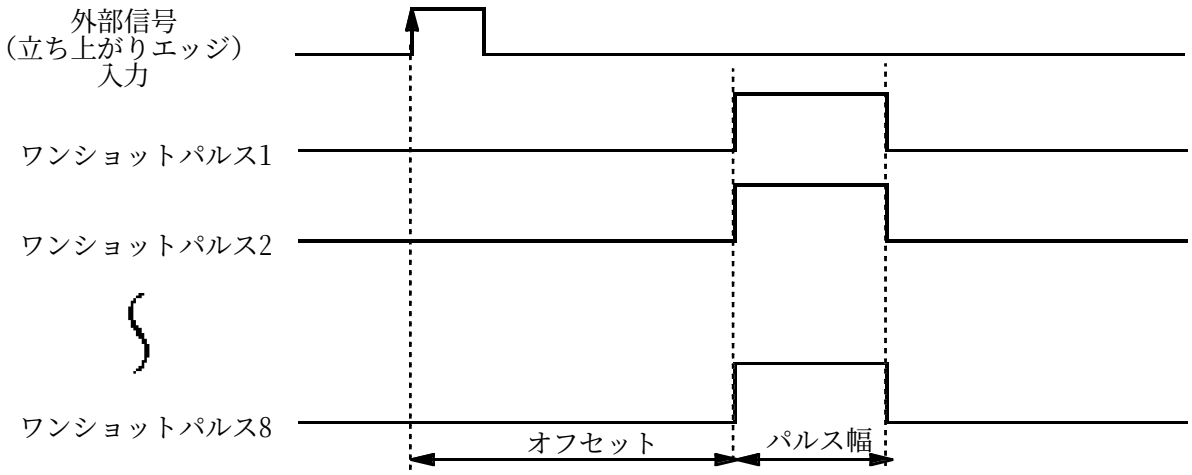


図1　ワンショットパルス出力

＊注：オフセットは外部信号の周期より小さいこと

表１、表２に本タスク例の機能割り付けを示します。本タスク例は表１、表２に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、PFCの機能を割り付け、ワンショットパルスを出力します。

表１　　ＡＴU-Ⅱ機能割り付け

ATU-Ⅱ内蔵機能		機能
端子	TIA0	外部信号を入力します。
	TOA8～H8	ワンショットパルスを出力します。
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの1段目の設定をします。
	TCR1A	ATU-Ⅱのプリスケーラの2段目の設定をします。
	TCR8	ATU-Ⅱのプリスケーラの2段目の設定をします。
	TIOR1A～1D	ATU-Ⅱチャネル1のレジスタの機能を設定をします。
	TRGMDR1	コンペアマッチをチャネル8ワンショットパルストリガとして使用するかターミネートトリガとして使用するか選択します。
	TIER0	ATU-Ⅱチャネル0の割り込み要求の許可/禁止を設定します。
	TSTR1	ATU-Ⅱチャネル0,1A,1Bのカウンタ動作を設定します。
	TCNR	DST8A～Hとダウンカウントスタートトリガの接続の許可/禁止を設定します。
	OSBR	外部信号の立ち上がり時のカウンタ値を検出します。
	GR1A～H	ワンショットパルスのオフセットの設定をします。
	DCNT8A～H	ワンショットパルスのパルス幅を設定します。

表２　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PAIOR	端子の入出力方向を設定します。
PACRL	端子の機能を設定します。
PKIOR	端子の入出力方向を設定します。
PKCRL	端子の機能を設定します。

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェア処理によりワンショットパルスを出力します。

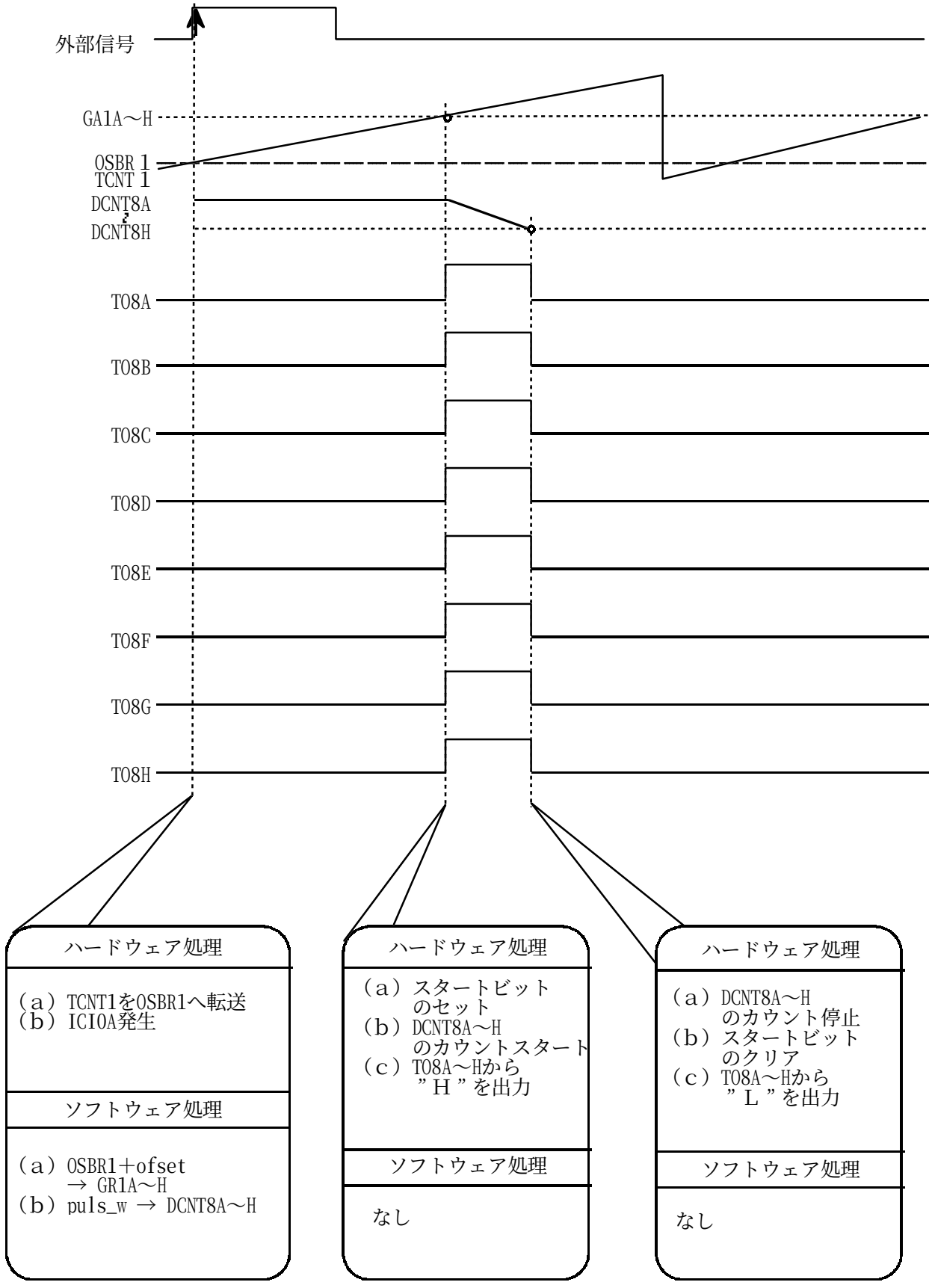


図2　ワンショットパルス出力動作原理

2.11　ワンショットパルス出力（ｵﾌｾｯﾄ付き）	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱ及びRAMの初期設定を行います。
ワンショット パルス出力	ICI0A	オフセットおよびパルス幅をGR1A～H、DCNT8A～Hに設定し、 ワンショットパルスを出力します。

（2）使用変数の説明

ラベル名	機能	データ長	使用モジュール名
ofset	ワンショットパルスのオフセットに相当するタイマ値を設定します。オフセットは以下の式にて求まります。 オフセット= タイマ値×Pφ周期×４(プリスケーラの分周比)	unsigned short	メインルーチン
puls_w	ワンショットパルスのパルス幅に相当するタイマ値を設定します。パルス幅は以下の式にて求まります。 パルス幅(ns)=タイマ値×Pφ周期(20MHz動作時50ns) ×４(プリスケーラの分周比)	unsigned short	ワンショット パルス出力

(3) 使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PA.PAIOR	PA1を入力に設定します。	0x0000	メインルーチン
PA.PACRL	端子マルチプレクスをTIOAの使用にします。	0x0001	
PK.PKIOR	ポートＫを出力に設定します。	0xFFFF	
PK.PKCRL	端子マルチプレクスをTO8A～Hの使用にします。	0x5555	
ATUC.PSCR1	ATU-Ⅱのプリスケーラ１段目をPφ/2に設定します。	0x01	
ATU1.TCR1A	ATU-Ⅱチャネル1のプリスケーラ2段目をφ'/2に設定します。	0x01	
ATU8.TCR8	ATU-Ⅱチャネル8のプリスケーラ2段目をφ'/2に設定します。	0x01	
ATU0.TIOR0	TIA0の立ち上がりエッジでICR0Aへインプットキャプチャするように設定します。	0x01	
ATU1.TIOR1A	ATU-Ⅱチャネル1のGR1A,1Bをコンペアマッチに設定します。	0x22	
ATU1.TIOR1B	ATU-Ⅱチャネル1のGR1C,1Dをコンペアマッチに設定します。	0x22	
ATU1.TIOR1C	ATU-Ⅱチャネル1のGR1E,1Fをコンペアマッチに設定します。	0x22	
ATU1.TIOR1D	ATU-Ⅱチャネル1のGR1G,1Hをコンペアマッチに設定します。	0x22	
ATU1.TRGMDR1	GR1A～Hのコンペアマッチをワンショットパルストリガの使用にします。	0x80	
ATU8.TCNR	ダウンカウントスタートトリガA～Hの接続を有効とします。	0x00FF	
ATUC.TSTR1	ATU-Ⅱチャネル0、チャネル1のカウント開始を設定します。	0x03	
ATU0.TIER0	ATU-Ⅱチャネル0のICF0Aによる割り込み要求を許可します。	0x01	
INTC.IPRC	ATU02の割り込み優先レベルを15に設定します。	0x000F	
ATU1.OSBR1	ATU-Ⅱチャネル0Aのインプットキャプチャで TCNT1を転送します。	—	ワンショット パルス出力
ATU1.GR1A～H	ワンショットパルスのオフセットを設定します。	—	
ATU8.DCNT8A～H	ワンショットパルスのパルス幅を設定します。	—	

プログラムリスト

```
/************************************************************************/
/*              オフセット付きワンショットパルス                        */
/************************************************************************/
#include "machine.h"                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main( void );
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define ofset  (*(unsigned short *)0xFFFFC000)          /*  ｵﾌｾｯﾄ      */
#define puls_w (*(unsigned short *)0xFFFFC002)          /*  ﾊﾟﾙｽ幅     */
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
        PA.PAIOR = 0x0000;              /* ﾎﾟｰﾄA入出力設定               */
        PA.PACRL = 0x0001;              /* TIOA(PA0)                    */
        PK.PKIOR = 0xFFFF;              /* ﾎﾟｰﾄKを出力に設定             */
        PK.PKCRL = 0x5555;              /* TO8H～8A(PK7～0)              */
        ATUC.PSCR1   = 0x01;            /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/2(100ns)    */
        ATU1.TCR1A   = 0x01;            /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/2(200ns)    */
        ATU8.TCR8    = 0x01;            /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/2(200ns)    */
        ATU0.TIOR0   = 0x01;  /* TIA0の立ち上がりｴｯｼﾞによるｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬ  */
        ATU1.TIOR1A  = 0x22;            /* ｺﾝﾍﾟｱﾏｯﾁ機能選択(GR1A,1B     */
        ATU1.TIOR1B  = 0x22;            /* ｺﾝﾍﾟｱﾏｯﾁ機能選択(GR1C,1D)    */
        ATU1.TIOR1C  = 0x22;            /* ｺﾝﾍﾟｱﾏｯﾁ機能選択(GR1E,1F)    */
        ATU1.TIOR1D  = 0x22;            /* ｺﾝﾍﾟｱﾏｯﾁ機能選択(GR1G,1H)    */
        ATU1.TRGMDR1 = 0x80;  /* ﾜﾝｼｮｯﾄｽﾀｰﾄﾄﾘｶﾞ(GR1A-Hのｱｳﾄﾌﾟｯﾄｺﾝﾍﾟｱ時)  */
        ATU8.TCNR    = 0x00FF;          /* ﾀﾞｳﾝｶｳﾝﾀｽﾀｰﾄﾄﾘｶﾞA-Hの接続を有効 */
        ATUC.TSTR1   = 0x03;            /* TCNT0,1Aｽﾀｰﾄ                 */
        ofset  = 0x0040;                /* ｵﾌｾｯﾄ  =  25.6μs             */
        puls_w = 0x0400;                /* ﾊﾟﾙｽ幅 = 409.6μs             */
        ATU0.TIER0  = 0x01;             /* ICF0Aによる割り込みを許可     */
        INTC.IPRC = 0x000F;             /* ATU02の割り込み優先ﾚﾍﾞﾙを15に  */
        set_imask(0x0);                 /* 割込み許可                   */
        while(1);                       /* 無限ﾙｰﾌﾟ（割込み待ち）       */
}
```

プログラムリスト

```
/************************************************************************/
/*              ワンショットパルス出力ルーチン                          */
/************************************************************************/
#pragma interrupt( ICI0A )
void ICI0A( void )
{
        ATU0.TSR0 &= 0xFE;              /* ｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬ0A割込みﾌﾗｸﾞｸﾘｱ      */
        ATU1.GR1A = ATU1.0SBR1 + ofset;      /* ｵﾌｾｯﾄ設定 25.6μs          */
        ATU1.GR1B = ATU1.0SBR1 + ofset;      /* ｵﾌｾｯﾄ設定 25.6μs          */
        ATU1.GR1C = ATU1.0SBR1 + ofset;      /* ｵﾌｾｯﾄ設定 25.6μs          */
        ATU1.GR1D = ATU1.0SBR1 + ofset;      /* ｵﾌｾｯﾄ設定 25.6μs          */
        ATU1.GR1E = ATU1.0SBR1 + ofset;      /* ｵﾌｾｯﾄ設定 25.6μs          */
        ATU1.GR1F = ATU1.0SBR1 + ofset;      /* ｵﾌｾｯﾄ設定 25.6μs          */
        ATU1.GR1G = ATU1.0SBR1 + ofset;      /* ｵﾌｾｯﾄ設定 25.6μs          */
        ATU1.GR1H = ATU1.0SBR1 + ofset;      /* ｵﾌｾｯﾄ設定 25.6μs          */
        ATU8.DCNT8A = puls_w;                /* ﾊﾟﾙｽ幅設定 409.6μs        */
        ATU8.DCNT8B = puls_w;                /* ﾊﾟﾙｽ幅設定 409.6μs        */
        ATU8.DCNT8C = puls_w;                /* ﾊﾟﾙｽ幅設定 409.6μs        */
        ATU8.DCNT8D = puls_w;                /* ﾊﾟﾙｽ幅設定 409.6μs        */
        ATU8.DCNT8E = puls_w;                /* ﾊﾟﾙｽ幅設定 409.6μs        */
        ATU8.DCNT8F = puls_w;                /* ﾊﾟﾙｽ幅設定 409.6μs        */
        ATU8.DCNT8G = puls_w;                /* ﾊﾟﾙｽ幅設定 409.6μs        */
        ATU8.DCNT8H = puls_w;                /* ﾊﾟﾙｽ幅設定 409.6μs        */
}
```

2.12　ワンショットパルス出力（ﾀｰﾐﾈｰﾄ）	MCU	SH７０５５	使用機能	ＡTU−Ⅱ

仕様

（1）図１に示すように外部信号の立ち上がりに同期してワンショットパルスを出力します。
　　オフセット、パルス幅は内部クロックカウント値を設定します。
（2）OTRへの設定によりパルス出力を強制的に終了しパルス幅を制御します。
（3）外部信号の立ち上がりからのオフセットおよびパルス幅は以下に示す範囲で可変できます。
　　Pφ周期×１＜オフセット＜Pφ周期×65536　　＊注
　　200ns≦パルス幅＜13.1ms

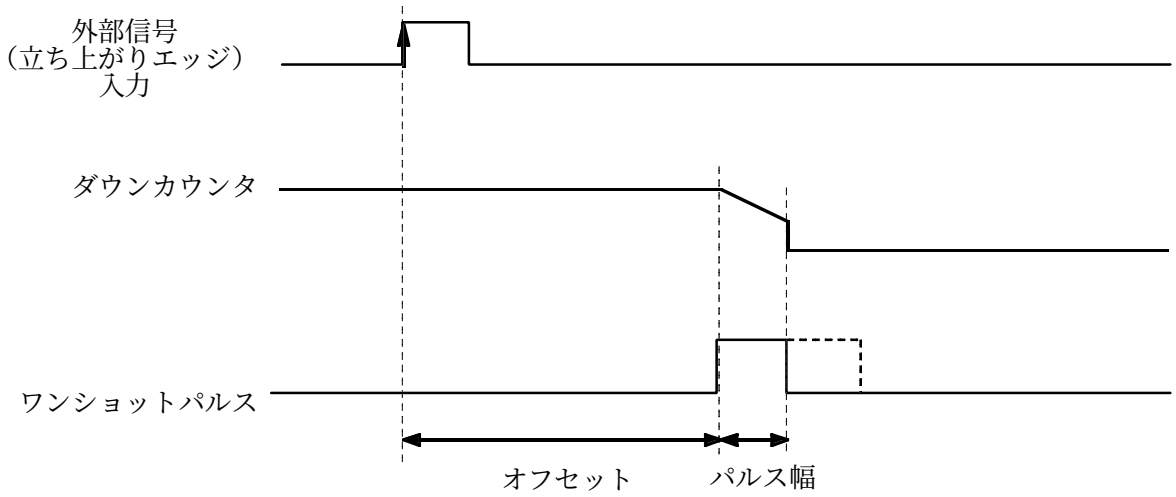


図１　ワンショットパルス出力

＊注：オフセットは外部信号の周期より小さいこと

使用機能説明

表１、表２に本タスク例の機能割り付けを示します。本タスク例は表１、表２に示すようにＳＨ７０５５に内蔵しているATU-Ⅱ、PFCの機能を割り付け、ワンショットパルスを出力します。

表１　　ＡＴＵ-Ⅱ機能割り付け

ATU-Ⅱ内蔵機能		機能
端子	TIOA	外部信号を入力します。
	TO8I～P	ワンショットパルスを出力します。
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの1段目の設定をします。
	TCR2A,2B	ATU-Ⅱのプリスケーラの2段目の設定をします。
	TCR8	ATU-Ⅱのプリスケーラの2段目の設定をします。
	TIOR0	ATU-Ⅱチャネル0のレジスタの機能を設定します。
	TIER0	ATU-Ⅱチャネル0の割り込み要求の許可/禁止を設定します。
	TIOR2A,2B	ATU-Ⅱチャネル2のレジスタの機能を設定します。
	TCNR	DST8A～Hとダウンカウントスタートトリガの接続の許可/禁止を設定します。
	OTR	ワンショットパルスの強制終了の許可/禁止を設定します。
	RLDENR	RLDRの値をDCNTへロードの許可/禁止を設定します。
	TSTR1	ATU-Ⅱチャネル0、2のカウンタ動作の設定をします。
	OSBR2	外部信号の立ち上がり時のカウンタ値を検出します。
	RLDR8	パルス幅の設定をします。
	GR2A～H	ターミネートの設定をします。
	OCR2A	ワンショットパルスのオフセットの設定をします。

表２　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PAIOR	端子の入出力方向を設定します。
PACRL	端子の機能を設定します。
PKIOR	端子の入出力方向を設定します。
PKCRH	端子の機能を設定します。

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェア処理によりワンショットパルスを遮断します。

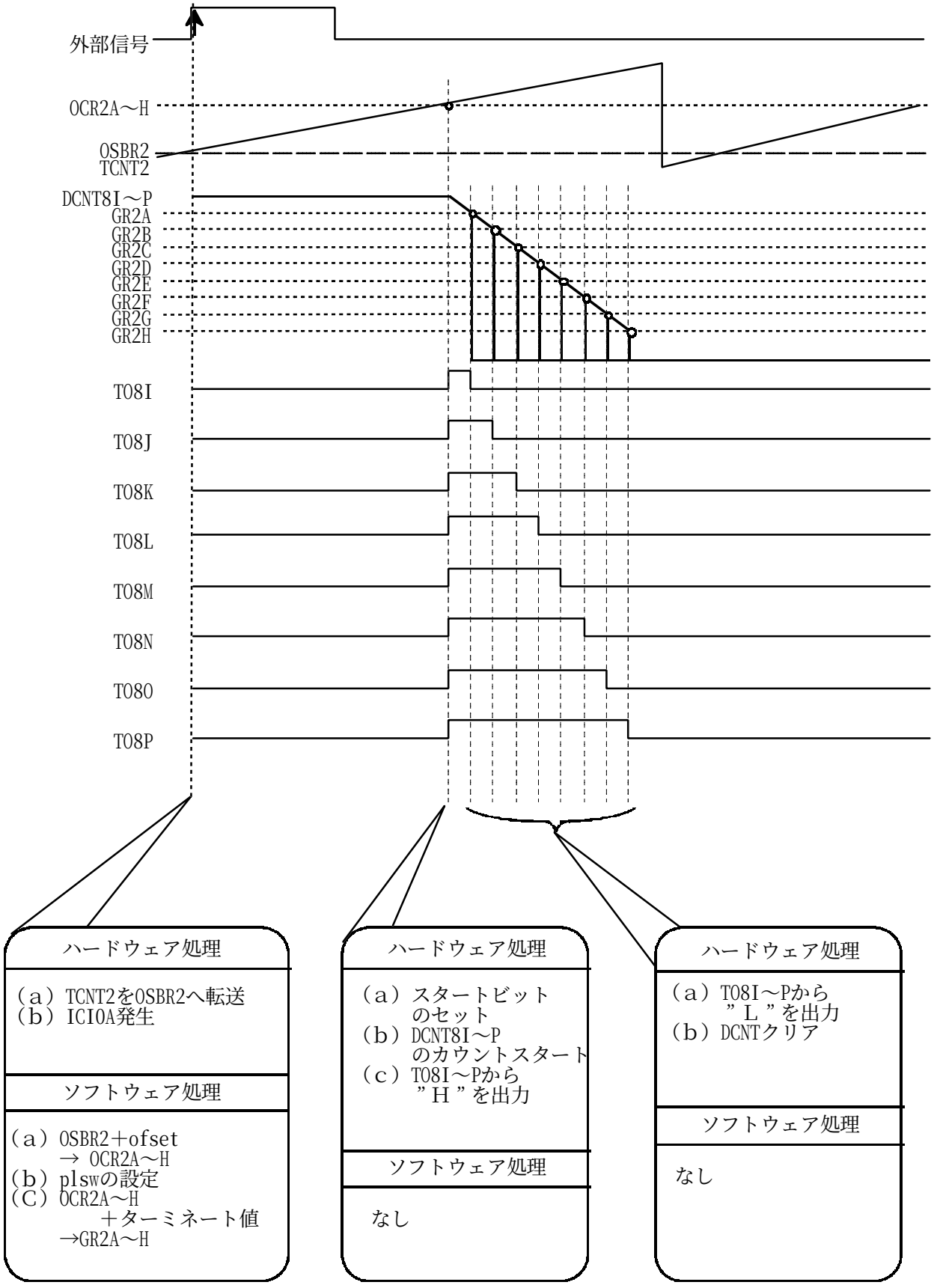


図2　ワンショットパルス遮断動作原理

2.12　ワンショットパルス出力（ﾀｰﾐﾈｰﾄ）	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱ及びRAMの初期設定を行ないます。
ワンショット パルス出力	ICI0A	オフセット及びパルス幅、ターミネートを設定し、ワンショットパルスの出力及び遮断をします。

（2）使用変数の説明

ラベル名	機能	データ長	使用モジュール名
ofset	ワンショットパルスのオフセットに相当するタイマ値を設定します。 オフセットは以下の式にて求まります。 オフセット=タイマ値×Pφ周期×4(プリスケーラの分周比)	unsigned short	メインルーチン ワンショット パルス出力
plsw	ワンショットパルスのパルス幅に相当するタイマ値を設定します。 パルス幅は以下の式にて求まります。 パルス幅(ns)=タイマ値×Pφ周期(20MHz動作時50ns) ×4(プリスケーラの分周比)	unsigned short	

（3）仕様内部レジスタの説明

レジスタ名	機能	設定値	使用モジュール名
PA.PAIOR	PA1を入力に設定します。	0x0000	メインルーチン
PA.PACRL	端子マルチプレクスをTI0Aの使用にします。	0x0001	
PK.PKIOR	ポートKを出力に設定します。	0xFFFF	
PK.PKCRH	端子マルチプレクスをT08I～Pの使用にします。	0x5555	
ATUC.PSCR1	ATU-Ⅱのプリスケーラ１段目をPφ/2に設定します。	0x01	
ATU2.TCR2A	ATU-Ⅱチャネル2のプリスケーラ2段目をφ'/2に設定します。	0x01	
ATU8.TCR8	ATU-Ⅱチャネル8のプリスケーラ2段目をφ'/2に設定します。	0x11	
ATU0.TIOR0	TI0Aの立ち上がりエッジでICR0Aへインプットキャプチャするように設定します。	0x01	
ATU2.TIOR2A	ATU-Ⅱチャネル2のGR2A,2Bをコンペアマッチに設定します。	0x11	
ATU2.TIOR2B	ATU-Ⅱチャネル2のGR2C,2Dをコンペアマッチに設定します。	0x11	
ATU2.TIOR2C	ATU-Ⅱチャネル2のGR2E,2Fをコンペアマッチに設定します。	0x11	
ATU2.TIOR2D	ATU-Ⅱチャネル2のGR2G,2Hをコンペアマッチに設定します。	0x11	
ATU8.TCNR	ダウンカウンタスタートトリガI～Pの接続を有効とします。	0xFF00	
ATU8.OTR	ダウンカウンタターミネートトリガI～Pで強制終了を許可します。	0xFF00	
ATUC.TSTR1	ATU-Ⅱチャネル0、チャネル2のカウント開始を設定します。	0x0D	
ATU8.RLDENR	RLDRの値をダウンカウンタへのロードを許可します。	0x80	
ATU0.TIER0	ATU-Ⅱチャネル0のICF0Aによる割り込み要求を許可します。	0x01	
INTC.IPRC	ATU02の割り込み優先レベルを15に設定します。	0x000F	
ATU2.OSBR2	ATU-Ⅱチャネル0のインプットキャプチャでTCNT2を設定します。	—	ワンショット パルス出力
ATU2.OCR2A～H	ワンショットパルスのオフセットを設定します。	—	
ATU8.RLDR8	ワンショットパルスのパルス幅を設定します。	—	
ATU2.GR2A～H	ターミネート値を設定します。	—	

プログラムリスト

```
/************************************************************************/
/*              オフセット付きワンショットパルス出力(ターミネート)      */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include "7055.h"                           /* 周辺レジスタ定義ﾍｯﾀﾞﾌｧｲﾙ  */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main( void );
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define ofset   (*(unsigned short *)0xFFFFC000)          /*  ｵﾌｾｯﾄ      */
#define plsw    (*(unsigned short *)0xFFFFC002)          /*  ﾊﾟﾙｽ幅     */
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
        PA.PAIOR    = 0x0000; /* ﾎﾟｰﾄA入出力設定                          */
        PA.PACRL    = 0x0001; /* TIOA(PA0)                                */
        PK.PKIOR    = 0xFFFF; /* ﾎﾟｰﾄKを出力に設定                        */
        PK.PKCRH    = 0x5555; /* TIO8P～I(PK15～8)                        */
        ATUC.PSCR1  = 0x01;   /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/2(100ns)               */
        ATU2.TCR2A  = 0x01;   /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/2(200ns)               */
        ATU2.TCR2B  = 0x01;   /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/2(200ns)               */
        ATU8.TCR8   = 0x11;   /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/2(200ns)               */
        ATU0.TIOR0  = 0x01;   /* TIOAの立ち上がりｴｯｼﾞによるｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬ  */
        ATU0.TIER0  = 0x01;   /* ICF0Aによる割り込みを許可                */
        ATU2.TIOR2A = 0x11;   /* GR2A,2B ｺﾝﾍﾟｱﾏｯﾁ機能                     */
        ATU2.TIOR2B = 0x11;   /* GR2C,2D ｺﾝﾍﾟｱﾏｯﾁ機能                     */
        ATU2.TIOR2C = 0x11;   /* GR2E,2F ｺﾝﾍﾟｱﾏｯﾁ機能                     */
        ATU2.TIOR2D = 0x11;   /* GR2G,2H ｺﾝﾍﾟｱﾏｯﾁ機能                     */
        ATU8.TCNR   = 0xFF00; /* ﾀﾞｳﾝｶｳﾝﾀｽﾀｰﾄﾄﾘｶﾞA～Hの接続を有効       */
        ATU8.OTR    = 0xFF00; /* ﾀﾞｳﾝｶｳﾝﾀﾀｰﾐﾈｰﾄﾄﾘｶﾞA～Hで強制終了を許可 */
        ATU8.RLDENR = 0x80;   /* ﾘﾛｰﾄﾞﾚｼﾞｽﾀの値をﾀﾞｳﾝｶｳﾝﾀへﾛｰﾄﾞ許可     */
        ATUC.TSTR1  = 0x0D;   /* TCNT0,2A,2Bｽﾀｰﾄ                          */
        ofset       = 0x0040; /* ｵﾌｾｯﾄ  =  12.8 μs                       */
        plsw        = 0x0400; /* ﾊﾟﾙｽ幅 = 204.8 μs                       */
        INTC.IPRC   = 0x000F; /* ATU02の割り込み優先ﾚﾍﾞﾙを15に設定        */
        set_imask(0x0);       /* 割込み許可                               */
        while(1);             /* 無限ﾙｰﾌﾟ（割込み待ち                     */
}
```

プログラムリスト

```
/************************************************************************/
/*              ワンショットパルス出力ルーチン                          */
/************************************************************************/
#pragma interrupt( ICI0A )
void ICI0A( void )
{
        ATU0.TSR0 &= 0xFE;              /* ｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬ0A割込みﾌﾗｸﾞｸﾘｱ      */
        ATU2.OCR2A = ATU2.OSBR2 + ofset     /* ｵﾌｾｯﾄ設定 12.8μs          */
        ATU2.OCR2B = ATU2.OSBR2 + ofset;    /* ｵﾌｾｯﾄ設定 12.8μs          */
        ATU2.OCR2C = ATU2.OSBR2 + ofset;    /* ｵﾌｾｯﾄ設定 12.8μs          */
        ATU2.OCR2D = ATU2.OSBR2 + ofset;    /* ｵﾌｾｯﾄ設定 12.8μs          */
        ATU2.OCR2E = ATU2.OSBR2 + ofset;    /* ｵﾌｾｯﾄ設定 12.8μs          */
        ATU2.OCR2F = ATU2.OSBR2 + ofset;    /* ｵﾌｾｯﾄ設定 12.8μs          */
        ATU2.OCR2G = ATU2.OSBR2 + ofset;    /* ｵﾌｾｯﾄ設定 12.8μs          */
        ATU2.OCR2H = ATU2.OSBR2 + ofset;    /* ｵﾌｾｯﾄ設定 12.8μs          */
        ATU8.RLDR8 = plsw;                  /* ﾊﾟﾙｽ幅設定 204.8μs        */
        ATU2.GR2A = ATU2.OCR2A + 0x0040;    /*  12.8μs後ﾀｰﾐﾈｰﾄ            */
        ATU2.GR2B = ATU2.OCR2B + 0x0080;    /*  25.6μs後ﾀｰﾐﾈｰﾄ            */
        ATU2.GR2C = ATU2.OCR2C + 0x0100;    /*  51.2μs後ﾀｰﾐﾈｰﾄ            */
        ATU2.GR2D = ATU2.OCR2D + 0x0180;    /*  76.8μs後ﾀｰﾐﾈｰﾄ            */
        ATU2.GR2E = ATU2.OCR2E + 0x0200;    /* 102.4μs後ﾀｰﾐﾈｰﾄ            */
        ATU2.GR2F = ATU2.OCR2F + 0x0280;    /* 128.0μs後ﾀｰﾐﾈｰﾄ            */
        ATU2.GR2G = ATU2.OCR2G + 0x0300;    /* 153.6μs後ﾀｰﾐﾈｰﾄ            */
        ATU2.GR2H = ATU2.OCR2H + 0x0400;    /* 204.8μs後ﾀｰﾐﾈｰﾄ            */
}
```

2.13 欠け歯検出からのワンショット出力 チャネル１,２,８,１０連動	MCU	SH７０５５	使用機能	ＡＴＵ-Ⅱ

仕様

（１）図１に示すような外部入力信号（16歯-内欠け歯2）の欠け歯及び立上りエッジを検出し、オフセット付のワンショットパルスを入力信号に対して一定のタイミング（位相差）で出力します。
（２）外部信号の入力停止を検出します。再入力時には、起動時と同じ動作をとります。
（３）以下に示す範囲で周期変動する外部入力信号に対応します。
前後のパルス周期×0.4＜今回のパルス周期＜前後のパルス周期×2.5 *注

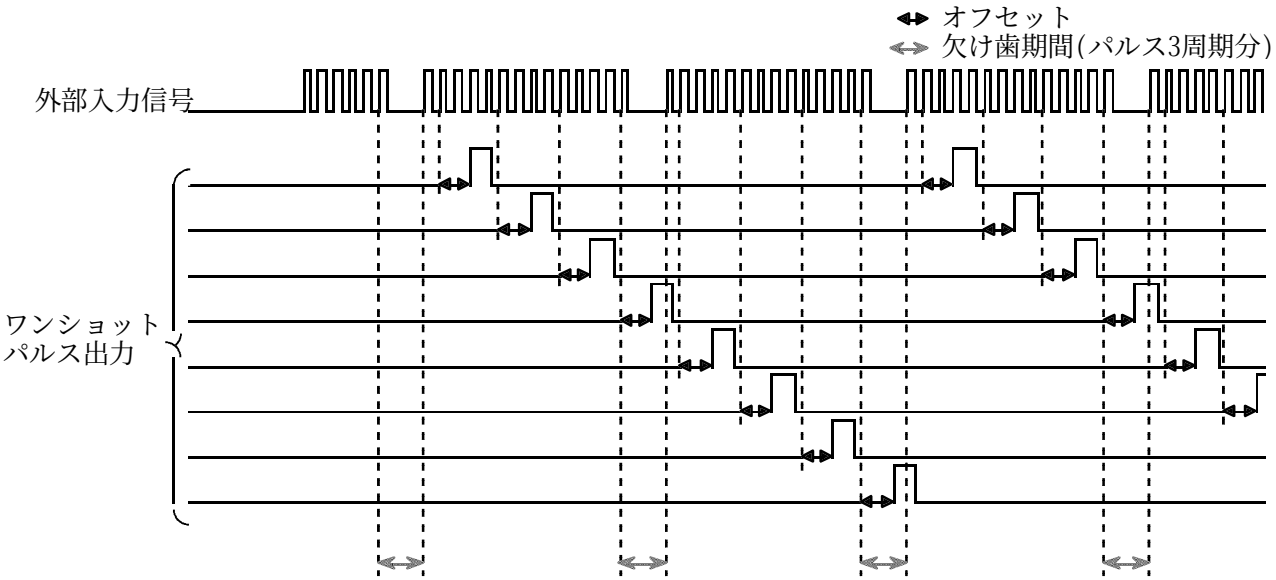


図１ 欠け歯検出からのワンショットパルス出力

*注：範囲外であった場合、欠け歯を検出しない事があります。

使用機能及び設定値

表１、表２、表３に本アプリケーション例で使用するピンファンクション、割込みコントローラとATU-Ⅱの機能割り付け及び設定値を示します。

表１ ピンファンクション

ピンファンクション		設定値	機能
端子	PA0	-	サイクル判定レベル（High：1stサイクル、Low：2ndサイクル）入力端子。
	PL0	-	ATU-Ⅱチャネル10の外部入力端子。（TI10）
	PK0～7	-	ATU-Ⅱチャネル8のワンショットパルス出力端子。（TO8A～H）
	PK8～15	-	ATU-Ⅱチャネル8のワンショットパルス出力端子。（TO8I～P）
レジスタ	PACRL	H'0000	PA0を汎用入出力端子機能に設定。
	PAIOR	H'0000	PA0を入力端子に設定。
	PADR	-	PA0の入力レベルの格納。
	PLCRL	H'0001	PL0をATU-Ⅱチャネル10の外部入力端子機能に設定。
	PLIOR	H'0000	PL0を入力端子に設定。
	PKCRL	H'5555	PK0～7をATU-Ⅱチャネル8のワンショットパルス出力端子機能に設定。
	PKCRH	H'5555	PK8～15をATU-Ⅱチャネル8のワンショットパルス出力端子機能に設定。
	PKIOR	H'FFFF	PK0～15を出力端子に設定。

2.13 欠け歯検出からのワンショット出力 チャネル１,２,８,１０連動	MCU	SH７０５５	使用機能	ＡＴＵ-Ⅱ

使用機能及び設定値

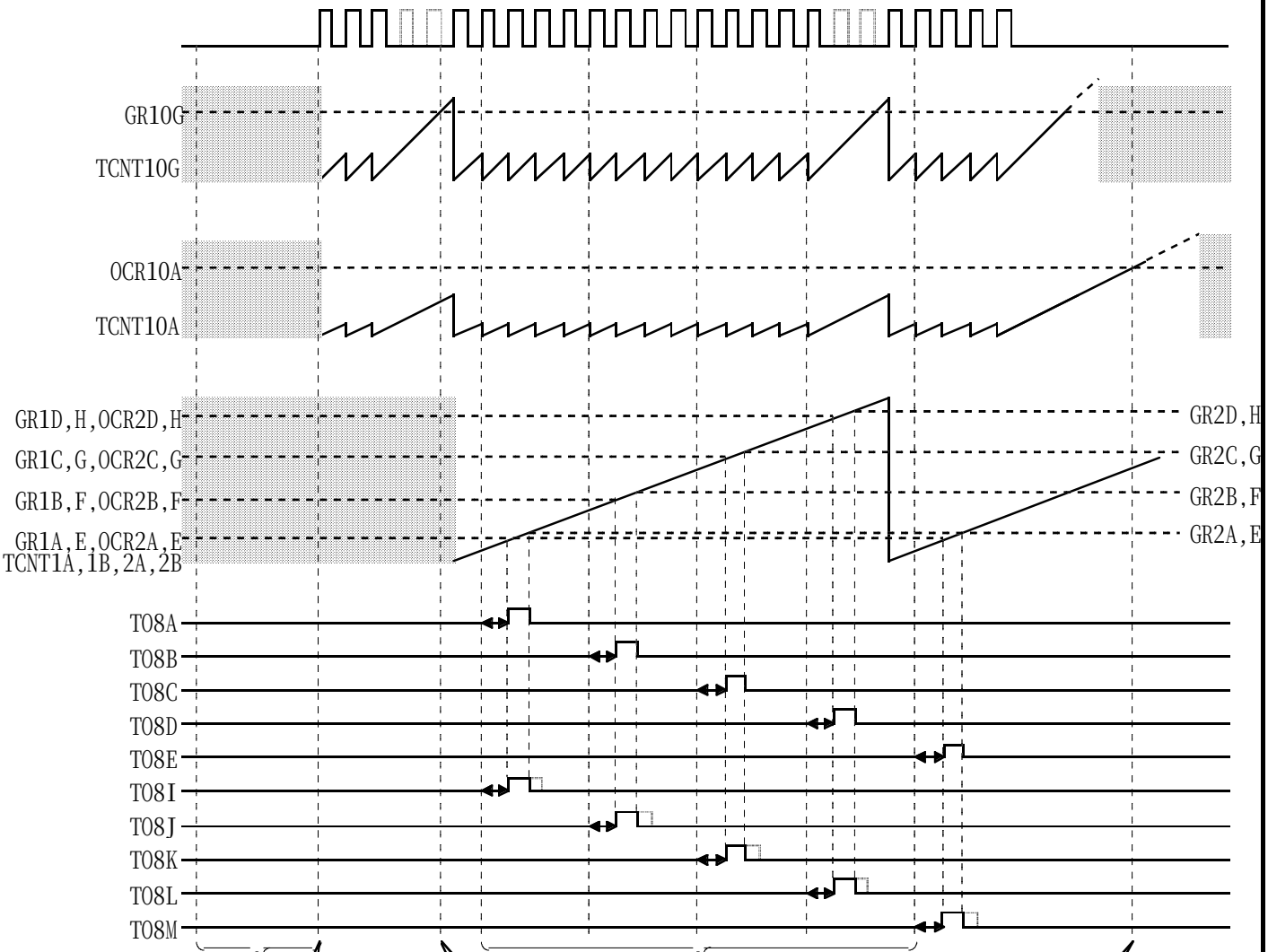
表３ ＡＴＵ-Ⅱ

ＡＴＵ-Ⅱ			設定値	機能
端子		TO8A～P	-	ワンショットパルス出力端子。
		TI10	-	外部入力端子。
レジスタ	共通	TSTR1	H'8E	TCNT1A,1B,2A,2B,10の動作を設定。
		PSCR1,4	H'04	ファーストプリスケーラをPφ/5に設定。
	チャネル1	TCR1A,B	H'09	TCNT1A,1Bを補正クロック(AGCKM)でカウントに設定。
		TIOR1A～D	H'11	GR1A～Hをコンペアマッチ機能に設定。
		GR1A～H	随時	TO8A～Hのワンショットパルスのオフセット値を設定。
		TRGMDR1	H'80	GR1A～HのコンペアマッチをTO8A～Hのワンショットパルス スタートトリガに設定。
	チャネル2	TCR2A,B	H'09	TCNT2A,2Bを補正クロック(AGCKM)でカウントに設定。
		TIOR2A～D	H'11	GR2A～Hをコンペアマッチ機能に設定。
		GR2A～H	随時	TO8I～Pのワンショットパルスのターミネート値を設定。
		OCR2A～H	随時	TO8I～Pのワンショットパルスのオフセット値を設定。
	チャネル8	TCR8	H'44	セカンドプリスケーラをφ'/16に設定。
		TCNR	H'FFFF	ATU-Ⅱチャネル８のダウンカウンタスタートレジスタ(DSTR)と チャネル１,２のコンペアマッチ信号の接続を許可。
		OTR	H'FF00	TCNT2AとGR2A～Hのコンペアマッチによるダウンカウンタターミネートトリガを有効に設定。
		DCNT8A～P	随時	ワンショットパルスのパルス幅を設定。
	チャネル10	TCR10	H'F5	TCCLR10=TCNT10FでTCNT1A,1B,2A,2Bをクリア、ノイズ キャンセル機能有効、TI10入力の立上りエッジ検出に設定。
		TIOR10	随時*注1	RLD10C更新設定、TCNT10D(シフト値)=TCNT10EでのTCNT10Eの 停止を禁止、逓倍率256に設定。
		TIER10	随時*注2	TSR10の各ステータスフラグによる割込みの許可/禁止を設定。
		OCR10A	H'0000FFFF	外部入力TI10停止検出値(＞欠け歯期間)を設定。
		OCR10B	随時*注3	外部入力TI10の4歯毎の立上りにコンペアマッチ割込みを設定。
		RLD10C	H'0013*注4	外部入力TI10周期/逓倍率/ファーストプリスケーラ周期を設定。 TIOR10の設定(RLDEN=0)によりICR10Aの値の逓倍分の1に更新。
		TCNT10C	H'0001	H'0001でRLD10Cの値が転送され、ファーストプリスケーラに従 いダウンカウント。倍周クロック(AGCK1)を生成。
		TCNT10D	随時*注5	外部入力TI10でカウントし、設定した逓倍率に従いシフトして TCNT10Eに転送。
		TCNT10F	H'1000	初期設定 =TCCLR10。補正クロック(AGCKM)を生成
		TCCLR	H'1000	サイクル中の補正クロック数(TI10×逓倍率)を設定。
		GR10G	H'0280	TCNT10Gの欠け歯検出値(逓倍率×2.5)を設定。
		NCR10	H'FF	設定値によりTI10の入力をマスク。NCR10=TCNT10Hでマスク解除。

注 1：初期設定時ICR10Aの値は不定である為RLD10Cは更新禁止、キャプチャ後は更新許可に設定。
2：初期設定時及び外部入力TI10停止検出(CMI10A)後は、ICF10Aによる割込みのみ許可。
キャプチャ(初回ICF10Aによる割込み)後は、CMF10Gによる割り込みを許可。
欠け歯検出(CMF10Gによる割込み)後は、TSR10のステータスフラグ全てによる割込みを許可するが、
ICF10Aによる割込みは1回のみの許可となります。
3：欠け歯検出後、1サイクル16歯に対して4歯毎のコンペアマッチ割込み(CMI10B)を設定。
4：初期値は予測値である為不定の場合は、RLD10Cの値を大きめに設定します。その場合、欠け歯を検出するまでに数サイクル要する事があります。小さめに設定した場合、通常の歯の間隔を誤って欠け歯と検出し、誤動作することがあります。
5：TCNT10Dは、サイクル中の欠け歯期間の補正(欠け歯期間の直前のエッジでの割込みで、TCNT10Dに欠け歯数を加える。本アプリケーション例ではこの補正は行っておりません)、
サイクル最後の欠け歯期間中の初期化(TCNT10D=H'00)が必要となります。

動作説明

（1）図2に動作原理を示します。図2に示すようにSH7055のハードウェア処理及びソフトウェアの処理により外部入力信号（16歯-内欠け歯2）の欠け歯及び立上りエッジの検出、入力信号に対して一定のタイミング（位相差）でのオフセット付のワンショットパルスの出力、外部信号入力停止の検出をします。



ハードウェア処理	ハードウェア処理	ハードウェア処理	ハードウェア処理	ハードウェア処理
(a) カウントスタート	(a) TCNT10AをICR10Aへ転送 (b) TCNT10A,Gクリア (c) TCNT10D(逓倍)をTCNT10Eへ転送 ※TCNT10D=H'00でTCNT10F=TCCLRのとき TCNT10D,E,Fクリア (d) IMI10AG発生	(a) IMI10AG発生	(a) CMI10B発生	(a) CMI10A発生
ソフトウェア処理	ソフトウェア処理	ソフトウェア処理	ソフトウェア処理	ソフトウェア処理
(a) 初期設定	(a) RLD10C更新許可 (b) CMF10Gによる割込みを許可	(a) CYLINDER設定 (b) CMF10A,Bによる割込みを許可 (c) 補正カウンタクリア TCNT10D=H'00	(a) DCNT8A～P, GR1A～H,GR2A～H OCR2A～H設定 (b) OCR10B更新 (c) CYLINDER更新	(a) カウンタ、レジスタ初期化 (b) ICF10Aによる割込みのみ許可

その他 ハードウェア処理：TCCLR=TCNT10FでTCNT1A,1B,2A,2Bクリア
ソフトウェア処理：IMI10AG内の要因判定及び要因別許可/禁止の設定。

図2 欠け歯検出からのワンショットパルス出力動作原理

2.13 欠け歯検出からのワンショット出力チャネル１,２,８,１０連動	MCU	SH７０５５	使用機能	ＡＴＵ-Ⅱ

ソフトウェア説明

（１）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱ、割込みコントローラ及びRAMの初期設定を行います。
	port_init	I/Oポートの初期設定を行います。
	ch1_init	ATU-Ⅱチャネル1の初期設定を行います。
	ch2_init	ATU-Ⅱチャネル2の初期設定を行います。
	ch8_init	ATU-Ⅱチャネル8の初期設定を行います。
	ch10_init	ATU-Ⅱチャネル10の初期設定を行います。
欠け歯及びエッジ検出	IMI10AG	外部入力TI10の欠け歯及びエッジを検出します。
エッジ検出	CMI10B INJSET IGNSET	外部入力TI10に対し一定のタイミング（位相差）で出力する オフセット付のワンショットパルスを設定します。
外部信号入力停止検出	CMI10A	外部入力TI10の信号入力停止を検出し、初期状態に戻します。

（２）使用変数の説明

ラベル名	機能	データ長	使用ラベル名
OFFSET[]	TO8A～Hから出力するワンショットパルスのオフセットに相当するタイマ値を設定します。 オフセットは以下の式にて求まります。 オフセット=タイマ値×補正クロック(AGCKM)	unsigned short	INJSET
INJPULSE[]	TO8A～Hから出力するワンショットパルスのパルス幅に相当するタイマ値を設定します。 パルス幅は以下の式にて求まります。 パルス幅(ns)=タイマ値 ×Pφ周期(20MHz動作時50ns) ×80(プリスケーラの分周比)	unsigned short	INJSET
IPW[]	TO8I～Pから出力するワンショットパルスのパルス幅に相当するタイマ値を設定します。 パルス幅は以下の式にて求まります。 パルス幅(ns)=タイマ値 ×Pφ周期(20MHz動作時50ns) ×80(プリスケーラの分周比)	unsigned short	IGNSET
DWELL[]	TO8I～Pから出力するワンショットパルスのオフセットに相当するタイマ値を設定します。 オフセットは以下の式にて求まります。 オフセット=タイマ値×補正クロック(AGCKM)	unsigned short	IGNSET
IGN[]	TO8I～Pから出力するワンショットパルスの出力遮断(ターミネート)をするタイマ値を設定します。 ターミネート値は以下の式にて求まります。 ターミネート値=タイマ値 ×補正クロック(AGCKM)	unsigned short	IGNSET
MISSTEETH	IMI10AG内のステータスを設定します。	unsigned char	main IMI10AG
SYLINDER	CMI10Bの発生要因となるエッジの位相を表します。 また、GR1A～H,GR2A～H,OCR2A～H及び変数のアドレスオフセットとして使用します。	unsigned char	IMI10AG CMI10B CMI10A

プログラムリスト

```
/*****************************************************************************/
/*                    欠け歯検出からのワンショットパルス出力                 */
/*****************************************************************************/
#include <stdio.h>                                  /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ       */
#include <machine.h>
#include "7055.h"                                   /* 周辺レジスタ定義ﾍｯﾀﾞﾌｧｲﾙ   */
/*****************************************************************************/
/*                    関数プロトタイプ宣言                                   */
/*****************************************************************************/
void main( void );
void port_init(void);
void ch1init(void);
void ch2init(void);
void ch8init(void);
void ch10init(void);
void INJSET(void);
void IGNSET(void);
/*****************************************************************************/
/*                    変数定義                                               */
/*****************************************************************************/
unsigned short OFFSET[8]
={0x0200,0x0600,0x0A00,0x0E00,0x0200,0x0600,0x0A00,0x0E00};
unsigned short INJPULSE[]
={0x0080};
unsigned short IPW[]
={0x00A0};
unsigned short DWELL[8]
={0x0200,0x0600,0x0A00,0x0E00,0x0200,0x0600,0x0A00,0x0E00};
unsigned short IGN[8]
={0x0280,0x0680,0x0A80,0x0E80,0x0280,0x0680,0x0A80,0x0E80};
unsigned char MISSTEETH;
unsigned char CYLINDER;
/*****************************************************************************/
/*                    メインルーチン                                         */
/*****************************************************************************/
void main( void )
{
        ATUC.PSCR1  = 0x04;                 /* ﾌﾟﾘｽｹｰﾗ1段目 Pφ/5(250ns)       */
        ATUC.PSCR4  = 0x04;                 /* ﾌﾟﾘｽｹｰﾗ1段目 Pφ/5(250ns)       */
        port_init();                        /* SH7055 I/Oﾎﾟｰﾄ ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      */
        ch1init();                          /* ATU-Ⅱ CH1     ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      */
        ch2init();                          /* ATU-Ⅱ CH2     ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      */
        ch8init();                          /* ATU-Ⅱ CH8     ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      */
        ch10init();                         /* ATU-Ⅱ CH10    ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      */
        MISSTEETH = 0x00;                   /* IMI10AG内ｲﾈｰﾌﾞﾙﾌﾗｸﾞｸﾘｱ          */
        ATUC.TSTR1  = 0x8E;                 /* TCNT1A,1B,2A,2B,10ｽﾀｰﾄ          */
        INTC.IPRI = 0x00FF;                 /* 割込みﾚﾍﾞﾙ設定                  */
        set_imask(0x0);                     /* 割込み許可                      */
        while(1);                           /* 無限ﾙｰﾌﾟ（割込み待ち）          */
}
```

プログラムリスト

```
/***    ﾎﾟｰﾄ ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      ***/
void port_init(void)
{
        PA.PACRL = 0x0000;                      /* PA0～PA7 を汎用入出力に設定      */
        PA.PAIOR = 0x0000;                      /* ﾎﾟｰﾄA入出力設定                  */
        PL.PLIOR = 0x0000;                      /* ﾎﾟｰﾄL入出力設定                  */
        PL.PLCRL = 0x0001;                      /* PL0をTI10に設定                  */
        PK.PKIOR = 0xFFFF;                      /* ﾎﾟｰﾄK入出力設定                  */
        PK.PKCRH = 0x5555;                      /* PK8～PK15をTO8I～TO8Pに設定      */
        PK.PKCRL = 0x5555;                      /* PK0～PK7をTO8A～8Hに設定         */
}
/***    ATU-Ⅱ CH1  ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      ***/
void ch1init(void)
{
        ATU1.TIOR1A  = 0x11;                    /* GRをｺﾝﾍﾟｱﾏｯﾁ機能に設定           */
        ATU1.TIOR1B  = 0x11;
        ATU1.TIOR1C  = 0x11;
        ATU1.TIOR1D  = 0x11;
        ATU1.TCR1A   = 0x09;                    /* AGCKM 立上りｴｯｼﾞでｶｳﾝﾄ           */
        ATU1.TCR1B   = 0x09;
        ATU1.TRGMDR1 = 0x80;                    /* GR1A～Hのｺﾝﾍﾟｱﾏｯﾁをﾜﾝｼｮｯﾄﾊﾟﾙｽ
                                                   ｽﾀｰﾄﾄﾘｶﾞに設定                   */
}
/***    ATU-Ⅱ CH2  ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      ***/
void ch2init(void)
{
        ATU2.TIOR2A = 0x11;                     /* GRをｺﾝﾍﾟｱﾏｯﾁ機能に設定           */
        ATU2.TIOR2B = 0x11;
        ATU2.TIOR2C = 0x11;
        ATU2.TIOR2D = 0x11;
        ATU2.TCR2A  = 0x09;                     /* AGCKM 立上りｴｯｼﾞでｶｳﾝﾄ           */
        ATU2.TCR2B  = 0x09;
}
/***    ATU-Ⅱ CH8  ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      ***/
void ch8init(void)
{
        ATU8.TCR8 = 0x44;                       /* 内部ｸﾛｯｸφ":φ'/16でｶｳﾝﾄ          */
        ATU8.TCNR = 0xFFFF;                     /* CH8のDSTRとCH1,2のｺﾝﾍﾟｱﾏｯﾁ
                                                   信号の接続許可                   */
        ATU8.OTR  = 0xFF00;                     /* I～Pのﾀﾞｳﾝｶｳﾝﾀｰﾐﾈｰﾄﾄﾘｶﾞ
                                                   による強制終了を許可             */
}
```

2.13 欠け歯検出からのワンショット出力 チャネル１,２,８,１０連動	MCU	SH７０５５	使用機能	ＡＴＵ-Ⅱ

プログラムリスト

```
/***    ATU-Ⅱ CH10 ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ      ***/
void ch10init(void)
{
        ATU10.TCR10 = 0xF5;                 /* TCCLR10=TCNT10FでTCNT1A,1B,2A,2Bｸﾘｱ
                                               ﾉｲｽﾞｷｬﾝｾﾙ機能有効
                                               TI10入力の立上りｴｯｼﾞを検出              */
        ATU10.TIOR10 = 0xB1;                /* RLD10C更新ﾏｽｸ,TCNT10Eの補正禁止
                                               AGCKM:TI10ｸﾛｯｸ×256でｶｳﾝﾄ              */
        ATU10.TIER10 = 0x0002;              /* 割込み許可/禁止設定                     */
        ATU10.RLD10C = 0x0013;              /* 初期値設定                              */
        ATU10.TCNT10C = 0x0001;             /* 初期値設定                              */
        ATU10.TCCLR10 = 0x1000;             /* 1周期間のAGCKM数                        */
        ATU10.TCNT10F = 0x1000;             /* 初期値設定(=TCCLR10)                    */
        ATU10.TCNT10D = 0x00;               /* 初期値設定                              */
        ATU10.GR10G = 0x0280;               /* 欠け歯検出値(逓倍率×2.5)を設定          */
        ATU10.OCR10A = 0x0000FFFF;          /* TI10入力停止検出値を設定                */
        ATU10.NCR10 = 0xFF;                 /* ﾉｲｽﾞｷｬﾝｾﾙ期間設定                       */
}
```

プログラムリスト

```
/*****************************************************************************/
/*                    割込みルーチン                                         */
/*****************************************************************************/
#pragma interrupt(IMI10AG,CMI10A,CMI10B)
void IMI10AG(void)
{
        if((ATU10.TSR10 & 0x0008) == 0x0008){               /* CMF10Gによる割込み        */
                if(MISSTEETH != 0x00){
                        ATU10.OCR10B = ATU10.TCNT10B+1;     /* OCR10B設定                */
                        if(MISSTEETH == 0x0F){
                                MISSTEETH = 0xF0;
                                ATU10.TIER10 |= 0x0005;/* CMF10A,CMF10Bによる割込みを許可 */
                        }
                        if((PA.PADR & 0x0001) == 0x0000){           /* ﾌｧｰｽﾄｻｲｸﾙ          */
                                CYLINDER = 0x00;                    /* ｼﾘﾝﾀﾞﾅﾝﾊﾞ 設定0    */
                        }else{                                      /* ｾｶﾝﾄﾞｻｲｸﾙ          */
                                CYLINDER = 0x04;                    /* ｼﾘﾝﾀﾞﾅﾝﾊﾞ 設定4    */
                        }
                        ATU10.TCNT10D = 0x00;                       /* 補正ｶｳﾝﾀｸﾘｱ        */
                        ATU10.TSR10 &= 0xFFF7;                      /* ｽﾃｰﾀｽﾌﾗｸﾞ ｸﾘｱ       */
                }
        }
        if((ATU10.TSR10 & 0x0002) == 0x0002){       /* ICF10Aによる割込み            */
                if(MISSTEETH != 0xFF){
                        if(MISSTEETH == 0xF0){
                                MISSTEETH = 0xFF;
                                ATU10.TIER10 &= 0xFFFD;     /* ICF10Aによる割込みを禁止  */
                        }
                        else{
                                MISSTEETH = 0x0F;
                                ATU10.TIOR10 = 0x31;        /* RLD10C更新許可            */
                        }
                        ATU10.TIER10 |= 0x0008;             /* CMF10Gによる割込みを許可  */
                        ATU10.TSR10 &= 0xFFF4;              /* ｽﾃｰﾀｽﾌﾗｸﾞ ｸﾘｱ             */
                }
        }
}
```

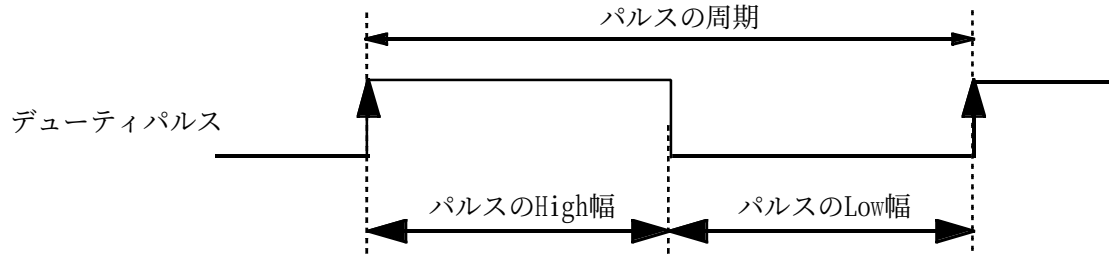
プログラムリスト

```
void CMI10B(void)                                   /* CMF10Bによる割込み          */
{
        INJSET();                                   /* ﾜﾝｼｮｯﾄﾊﾟﾙｽA～Hｾｯﾄﾙｰﾁﾝ      */
        IGNSET();                                   /* ﾜﾝｼｮｯﾄﾊﾟﾙｽI～Pｾｯﾄﾙｰﾁﾝ      */
        if((CYLINDER & 0x03) == 0x03){              /* ｼﾘﾝﾀﾞﾅﾝﾊﾞ3,7               */
                ATU10.OCR10B += 0x02;               /* OCR10B更新                  */
        }
        else{                                       /* ｼﾘﾝﾀﾞﾅﾝﾊﾞ0～2,4～6          */
                ATU10.OCR10B += 0x04;               /* OCR10B更新                  */
        }
        CYLINDER += 0x01;                           /* ｼﾘﾝﾀﾞﾅﾝﾊﾞ更新              */
        ATU10.TSR10 &= 0xFFFB;                      /* ｽﾃｰﾀｽﾌﾗｸﾞｸﾘｱ               */
}
void CMI10A(void)                                   /* CMF10Aによる割込み          */
{
        ATU10.TIER10 = 0x0002;                      /* ICF10Aによる割込みのみ許可*/
        ATU10.RLD10C = 0x0013;                      /* 初期値設定                  */
        ATU10.TCNT10C = 0x0001;                     /* 初期値設定                  */
        ATU10.TCNT10F = 0x1000;                     /* 初期値設定                  */
        MISSTEETH = 0x00;                           /* 欠け歯検出ﾌﾗｸﾞｸﾘｱ           */
    ATU10.TSR10 &= 0xFFF0;                          /* ｽﾃｰﾀｽﾌﾗｸﾞｸﾘｱ               */
}
/***    ﾜﾝｼｮｯﾄﾊﾟﾙｽA～Hｾｯﾄﾙｰﾁﾝ ***/
void INJSET(void)
{
        *(&ATU8.DCNT8A + CYLINDER) = INJPULSE[0];   /* DCNT8A～H(ﾊﾟﾙｽ幅)設定       */
        *(&ATU1.GR1A + CYLINDER) = OFFSET[CYLINDER];/* GR1A～H(ｵﾌｾｯﾄ)設定          */
}
/***    ﾜﾝｼｮｯﾄﾊﾟﾙｽI～Pｾｯﾄﾙｰﾁﾝ ***/
void IGNSET(void)
{
        *(&ATU8.DCNT8I + CYLINDER) = IPW[0];        /* DCNT8I～P(ﾊﾟﾙｽ幅)設定       */
        *(&ATU2.GR2A + CYLINDER) = IGN[CYLINDER];   /* GR2A～H(ﾀｰﾐﾈｰﾄ)設定         */
        *(&ATU2.OCR2A + CYLINDER) = DWELL[CYLINDER];/* OCR2A～H(ｵﾌｾｯﾄ)設定         */
}
```

2.14　PWM出力	MCU	ＳＨ７０５５	使用機能	ＡＴU−Ⅱ

仕様

（1）図１に示すように、デューティ及び周期を変化できるパルスを出力します。



$$デューティ＝\frac{パルスのＨｉｇｈ幅}{パルスの周期}\times 100（\%）$$

図１　PWM出力タイミング

使用機能説明

表１、表2に本タスク例の機能割り付けを示します。本タスク例は表１、表2に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、PFCの機能を割り付け、PWMを出力します。

表１　　ＡＴU-Ⅱ機能割り付け

ＡＴU-Ⅱの内蔵機能		機能
端子	TIO3A～D	PWMを出力します。
レジスタ	PSCR1	ATU-Ⅱプリスケーラの設定をします。
	TCR3	ATU-Ⅱチャネル3のプリスケーラを設定します。
	TMDR	ATU-Ⅱチャネル3の機能を選択します。
	GR3D	PWMの周期を設定します。
	GR3A～C	PWMのデューティを設定します。
	TSTR1	ATU-Ⅱチャネル3のカウンタ動作を設定します。

表2　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PAIOR	端子の入出力方向を設定します。
PACRH	端子の機能を設定します。

動作原理

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスを出力します。

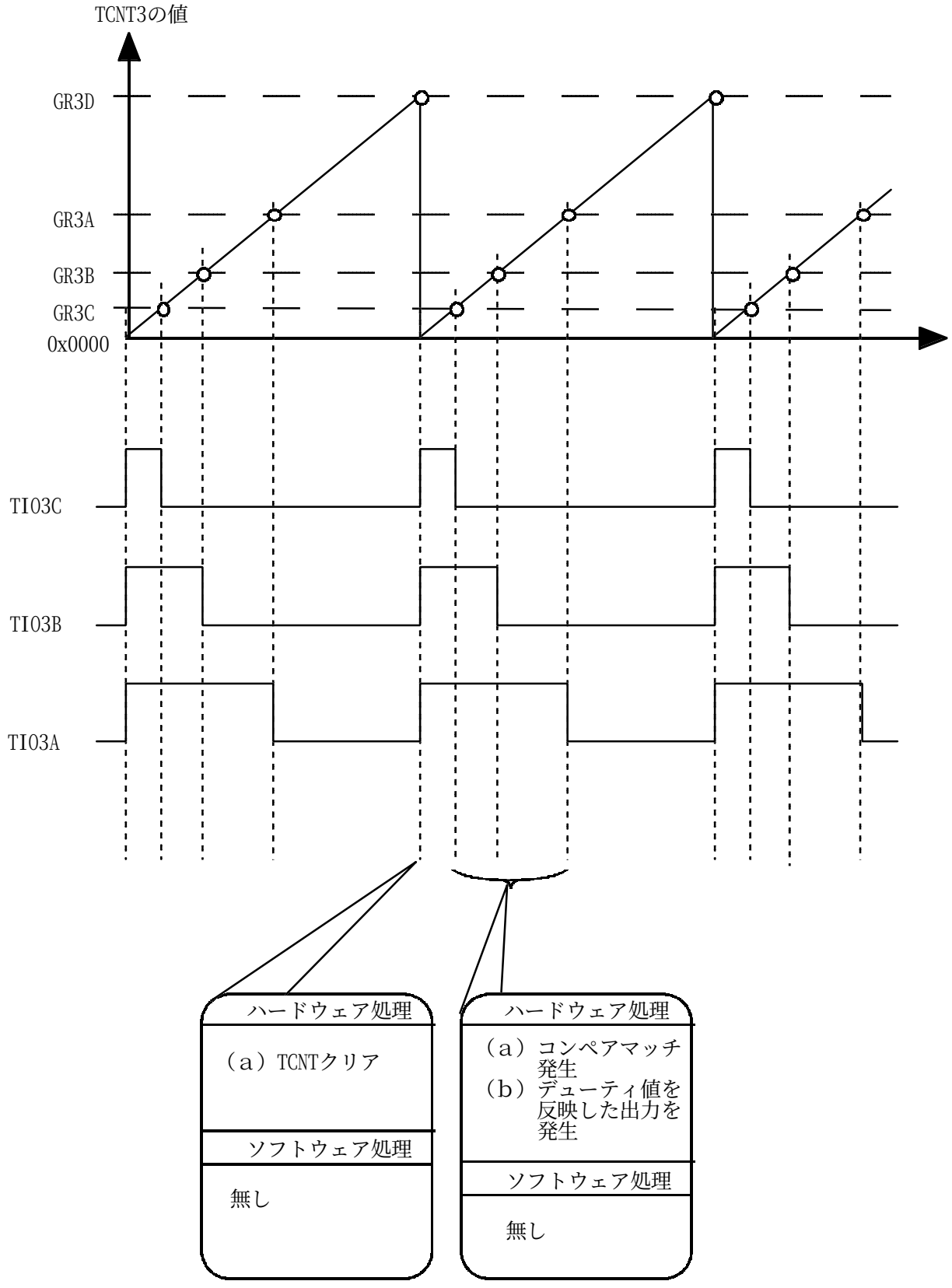


図２　PWM出力動作原理

2.14　PWM出力	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱの初期設定を行ないます。

（2）使用変数の説明

本タスクでは変数は使用しておりません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PA.PAIOR	PA4～7を出力端子に設定します。	0x00F0	メインルーチン
PA.PACRL	端子マルチプレクスをTIO3A～3D端子に設定します。	0x5500	
ATUC.PSCR1	ATU-Ⅱのプリスケーラ1段目をPφ/2に設定します。	0x01	
ATU3.TCR3	ATU-Ⅱチャネル3のプリスケーラをφ'/2に設定します。	0x01	
ATU3.TMDR	ATU-Ⅱチャネル3はPWM機能を選択します。	0x01	
ATU3.GR3D	周期を設定します。	0x6400	
ATU3.GR3A	デューティを設定します。	0x3200	
ATU3.GR3B	デューティを設定します。	0x1900	
ATU3.GR3C	デューティを設定します。	0x0C80	
ATUC.TSTR1	ATU-Ⅱチャネル3のカウント開始を設定します。	0x10	

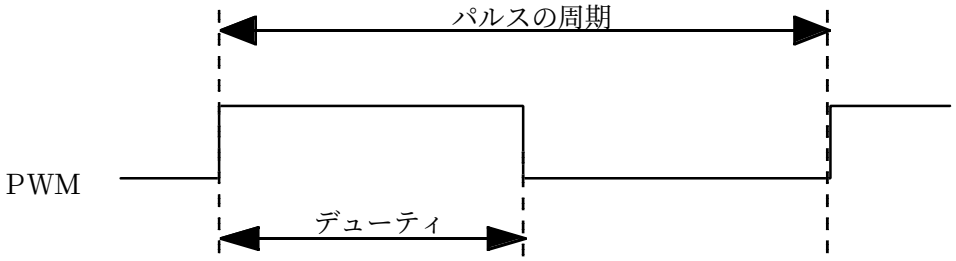
プログラムリスト

```
/*****************************************************************************/
/*              PWM出力                                                      */
/*****************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/*****************************************************************************/
/*              関数プロトコル宣言                                           */
/*****************************************************************************/
void main(void);
/*****************************************************************************/
/*              メインルーチン                                               */
/*****************************************************************************/
void main(void)
{
        PA.PAIOR   = 0x00F0;        /* ﾎﾟｰﾄA入出力設定                  */
        PA.PACRL   = 0x5500;        /* TIO3A～3D(PA4～PA7)              */
        ATUC.PSCR1 = 0x01;          /* ﾌﾟﾘｽｹｰﾗ1段目   Pφ/2(100ns)      */
        ATU3.TCR3  = 0x01;          /* ﾌﾟﾘｽｹｰﾗ2段目   φ'/2(200ns)      */
        ATU3.TMDR  = 0x01;          /* PWM機能を選択                    */
        ATU3.GR3D  = 0x6400;        /* 周期 = 10.24ms                   */
        ATU3.GR3A  = 0x3200;        /* ﾃﾞｭｰﾃｨｰ1 = 50   %                */
        ATU3.GR3B  = 0x1900;        /* ﾃﾞｭｰﾃｨｰ2 = 25   %                */
        ATU3.GR3C  = 0x0C80;        /* ﾃﾞｭｰﾃｨｰ3 = 12.5 %                */
        ATUC.TSTR1 = 0x10;          /* ﾁｬﾈﾙ3 ｶｳﾝﾀｽﾀｰﾄ                   */
        while(1);                   /* 無限ﾙｰﾌﾟ(割り込み待ち)           */
}
```

2.15　非相補PWM出力（専用）	MCU	ＳＨ７０５５	使用機能	ＡTU−Ⅱ

仕様

（1）図１に示すように、デューティおよび周期を変化できるパルスを出力します。
（2）出力するパルスのデューティは100%から0%の間で任意に設定できます。



$$\text{デューティ} = \frac{\text{パルスのHigh幅}}{\text{パルスの周期}} \times 100\%$$

図１　非相補ＰWM出力タイミング

使用機能説明

表１、表2に本タスク例の機能割り付けを示します。本タスク例は表１、表2に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、PFCの機能を割り付け、PWMを出力します。

表1　ＡＴＵ-Ⅱ機能割り付け

ＡＴＵ-Ⅱ内蔵機能		機能
端子	TO6A～D	PWMを出力します。
レジスタ	PSCR2	ATU-Ⅱのプリスケーラの設定をします。
	TCR6A,6B	ATU-Ⅱチャネル6のプリスケーラを設定します。
	PMDR6	ATU-Ⅱチャネル6のPWM出力の仕様を設定します。
	CYLR6A～D	PWMの周期を設定します。
	BFR6A～D	PWMのデューティを設定します。
	TSTR2	ATU-Ⅱチャネル6A～Dのカウンタ動作を設定します。
	TIER6	ATU-Ⅱチャネル6の割込み要求の許可/禁止を設定します。

表２　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PBIOR	端子の入出力方向を設定します。
PBCRL	端子の機能を設定します。

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスを出力します。

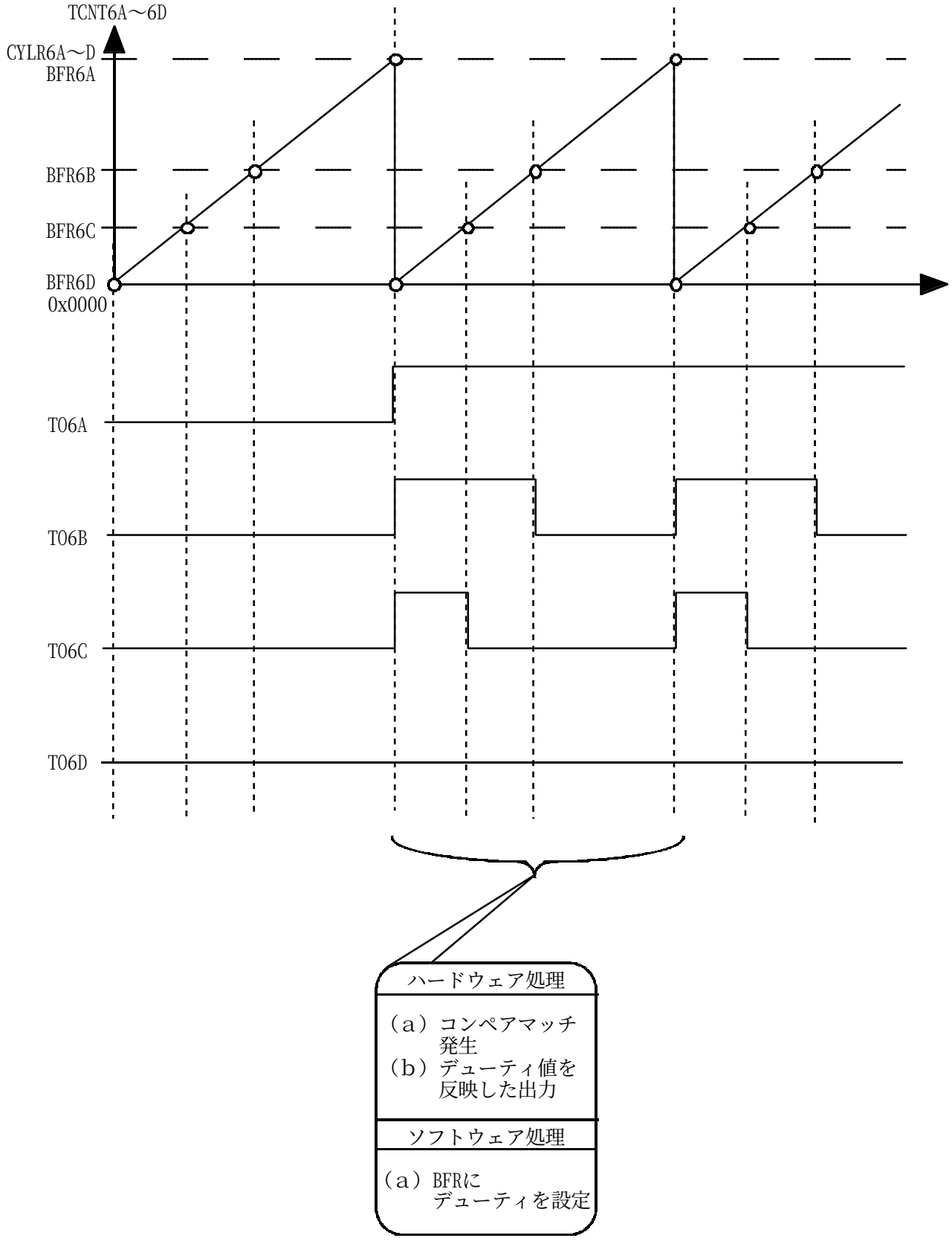


図２　非相補ＰＷＭ出力動作原理

2.15　非相補PWM出力（専用）	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱの初期設定を行ないます。

（2）使用変数の説明

本タスクでは変数は使用しておりません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PB.PBIOR	PB0～3を出力端子に設定します。	0x0055	メインルーチン
PB.PBCRL	端子マルチプレクスをTO6A～7D端子に設定します。	0x000F	
ATUC.PSCR2	ATU-Ⅱのプリスケーラ1段目をPφ/10に設定します。	0x09	
ATU6.TCR6A,6B	ATU-Ⅱチャネル6のプリスケーラをφ'/32に設定します。	0x55	
ATU6.PMDR6	ATU-Ⅱチャネル6は非相補PWMモード、オンデューティの出力を設定します。	0x00	
ATU6.CYLR6A～D	ATU-Ⅱチャネル6A～DのPWMの周期を設定します。	0x0800	
ATU6.BFR6A	ATU-Ⅱチャネル6のPWMのデューティを設定します。	0x0800	
ATU6.BFR6B	ATU-Ⅱチャネル6のPWMのデューティを設定します。	0x0400	
ATU6.BFR6C	ATU-Ⅱチャネル6のPWMのデューティを設定します。	0x0200	
ATU6.BFR6D	ATU-Ⅱチャネル6のPWMのデューティを設定します。	0x0000	
ATUC.TSTR2	ATU-Ⅱチャネル6A～Dのカウンタ開始を設定します。	0x0F	
ATU6.TIER6	ATU-Ⅱチャネル6のコンペアマッチによる割り込み要求を禁止します。	0x00	
INTC.IPRG	ATU6の割り込み優先レベルを15に設定します。	0x00F0	

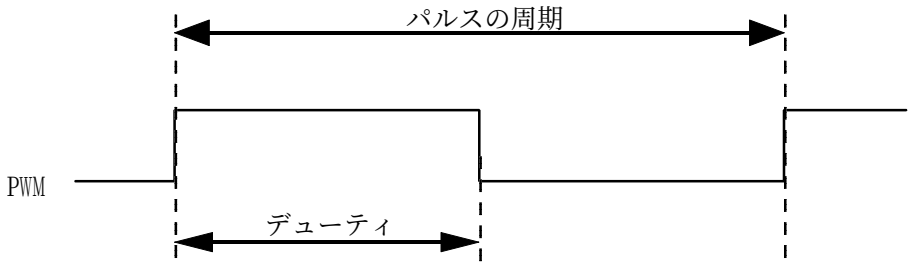
プログラムリスト

```
/************************************************************************/
/*              非相補ＰＷＭ出力（専用）                                */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ       */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main( void );
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
        PB.PBCRL = 0x0055;              /* TO6A～D機能設定                */
        PB.PBIOR = 0x000F;              /* PB0～3を出力端子に設定         */
        ATUC.PSCR2  = 0x09;             /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/10(500ns)    */
        ATU6.TCR6A  = 0x55;             /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/32(16μs)     */
        ATU6.TCR6B  = 0x55;             /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/32(16μs)     */
        ATU6.PMDR6  = 0x00;             /* ｵﾝ･ﾃﾞｭｰﾃｨ、非相補PWM          */
        ATU6.CYLR6A = 0x0800;           /* PWM周期 = 32.768[ms]           */
        ATU6.CYLR6B = 0x0800;           /* PWM周期 = 32.768[ms]           */
        ATU6.CYLR6C = 0x0800;           /* PWM周期 = 32.768[ms]           */
        ATU6.CYLR6D = 0x0800;           /* PWM周期 = 32.768[ms]           */
        ATU6.BFR6A  = 0x0800;           /* PWMﾃﾞｭｰﾃｨ 100                  */
        ATU6.BFR6B  = 0x0400;           /* PWMﾃﾞｭｰﾃｨ  50                  */
        ATU6.BFR6C  = 0x0200;           /* PWMﾃﾞｭｰﾃｨ  25                  */
        ATU6.BFR6D  = 0x0000;           /* PWMﾃﾞｭｰﾃｨ   0                  */
        ATUC.TSTR2  = 0x0F;             /* ﾁｬﾈﾙ6A～D ｶｳﾝﾄｽﾀｰﾄ             */
        ATU6.TIER6  = 0x00;             /* ｺﾝﾍﾟｱﾏｯﾁによる割込み禁止       */
        INTC.IPRG   = 0x00F0;           /* ATU6の割込み優先ﾚﾍﾞﾙを15に設定  */
        set_imask(0xF);                 /* 割込み禁止状態維持             */
        while(1);                       /* 無限ﾙｰﾌﾟ                       */
}
```

2.16　相補PWM出力（専用）	MCU	SH7055	使用機能	ATU−Ⅱ

仕様

（1）図1に示すように、デューティおよび周期を変化できるパルスを出力します。
（2）出力するパルスのデューティは100%から0%の間で任意に設定できます。



$$\text{デューティ} = \frac{\text{パルスのHigh幅}}{\text{パルスの周期}} \times 100\%$$

図1　相補PWM出力タイミング

使用機能説明

表1、表2に本タスク例の機能割り付けを示します。本タスク例は表1、表2に示すようにSH7055に内蔵されているATU-Ⅱ、PFCの機能を割り付け、PWMを出力します。

表1　ATU-Ⅱ機能割り付け

ATU-Ⅱ内蔵機能		機能
端子	TO6A～D	PWMを出力します。
レジスタ	PSCR2	ATU-Ⅱのプリスケーラの設定をします。
	TCR6A,6B	ATU-Ⅱチャネル6のプリスケーラを設定します。
	PMDR6	ATU-Ⅱチャネル6のPWM出力の仕様を設定します。
	CYLR6A～D	PWMの周期を設定します。
	BFR6A～D	PWMのデューティを設定します。
	TSTR2	ATU-Ⅱチャネル6A～Dのカウンタ動作を設定します。
	TIER6	ATU-Ⅱチャネル6の割込み要求の許可/禁止を設定します。

表2　PFC機能割り付け

PFCレジスタ	機能
PBIOR	端子の入出力方向を設定します。
PBCRL	端子の機能を設定します。

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスを出力します。

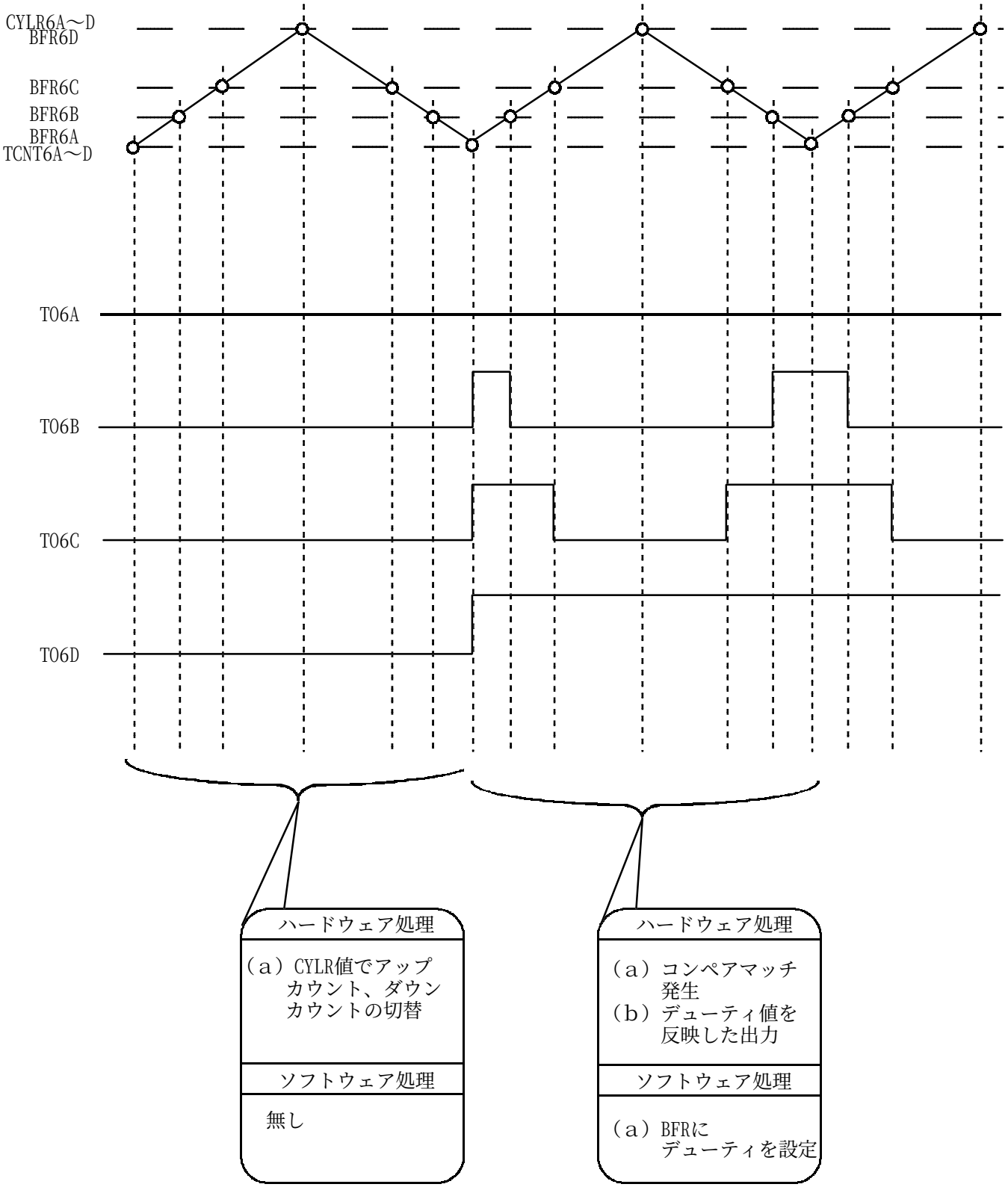


図２　相補ＰＷＭ出力動作原理

2.16　相補PWM出力（専用）	MCU	SH7055	使用機能	ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱの初期設定を行ないます。

（2）使用変数の説明

本タスクでは変数は使用しておりません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PB.PBIOR	PB0～3を出力端子に設定します。	0x0055	メインルーチン
PB.PBCRL	端子マルチプレクスをTIO6A～7D端子に設定します。	0x000F	
ATUC.PSCR2	ATU-Ⅱのプリスケーラ1段目をPφ/10に設定します。	0x09	
ATU6.TCR6A,6B	ATU-Ⅱチャネル6のプリスケーラをφ'/32に設定します。	0x55	
ATU6.PMDR6	ATU-Ⅱチャネル6は相補PWMモード、オンデューティの出力を設定します。	0x0F	
ATU6.CYLR6A～D	ATU-Ⅱチャネル6APWMの周期を設定します。	0x0800	
ATU6.BFR6A	ATU-Ⅱチャネル6のPWMのデューティを設定します。	0x0800	
ATU6.BFR6B	ATU-Ⅱチャネル6のPWMのデューティを設定します。	0x0400	
ATU6.BFR6C	ATU-Ⅱチャネル6のPWMのデューティを設定します。	0x0200	
ATU6.BFR6D	ATU-Ⅱチャネル6のPWMのデューティを設定します。	0x0000	
ATUC.TSTR2	ATU-Ⅱチャネル6A～Dのカウント開始を設定します。	0x0F	
ATU6.TIER6	ATU-Ⅱチャネル6のコンペアマッチによる割り込み要求を禁止します。	0x00	
INTC.IPRG	ATU6の割り込み優先レベルを15に設定します。	0x00F0	

プログラムリスト

```
/************************************************************************/
/*              相補ＰＷＭ出力（専用）                                  */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main( void );
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
        PB.PBCRL = 0x0055;              /* TO6A～D機能設定               */
        PB.PBIOR = 0x000F;              /* PB0～3を出力端子に設定        */
        ATUC.PSCR2  = 0x09;             /* ﾌﾟﾘｽｹｰﾗ1段目  Pφ/10(500ns)   */
        ATU6.TCR6A  = 0x55;             /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/32(16μs)    */
        ATU6.TCR6B  = 0x55;             /* ﾌﾟﾘｽｹｰﾗ2段目  φ'/32(16μs)    */
        ATU6.PMDR6  = 0x0F;             /* ｵﾝ･ﾃﾞｭｰﾃｨ、相補PWM           */
        ATU6.CYLR6A = 0x0800;           /* PWM周期 = 65.536[ms]          */
        ATU6.CYLR6B = 0x0800;           /* PWM周期 = 65.536[ms]          */
        ATU6.CYLR6C = 0x0800;           /* PWM周期 = 65.536[ms]          */
        ATU6.CYLR6D = 0x0800;           /* PWM周期 = 65.536[ms]          */
        ATU6.BFR6A  = 0x0000;           /* PWMﾃﾞｭｰﾃｨ   0                  */
        ATU6.BFR6B  = 0x0200;           /* PWMﾃﾞｭｰﾃｨ  25                  */
        ATU6.BFR6C  = 0x0400;           /* PWMﾃﾞｭｰﾃｨ  50                  */
        ATU6.BFR6D  = 0x0800;           /* PWMﾃﾞｭｰﾃｨ 100                  */
        ATUC.TSTR2  = 0x0F;             /* ﾁｬﾈﾙ6A～D ｶｳﾝﾄｽﾀｰﾄ             */
        ATU6.TIER6  = 0x00;             /* ｺﾝﾍﾟｱﾏｯﾁによる割込み禁止      */
        INTC.IPRG   = 0x00F0;           /* ATU6の割込み優先ﾚﾍﾞﾙを15に設定 */
        set_imask(0xF);                 /* 割込み禁止状態維持            */
        while(1);                       /* 無限ﾙｰﾌﾟ                      */
}
```

2.17　APC出力	MCU	ＳＨ７０５５	使用機能	ＡＰＣ,ＡＴＵ-Ⅱ

仕様

（1）図１に示すように、POPCRで選択された端子からパルスを出力します。パルスのＡ期間及びＢ期間は内部クロックカウント値を設定します。

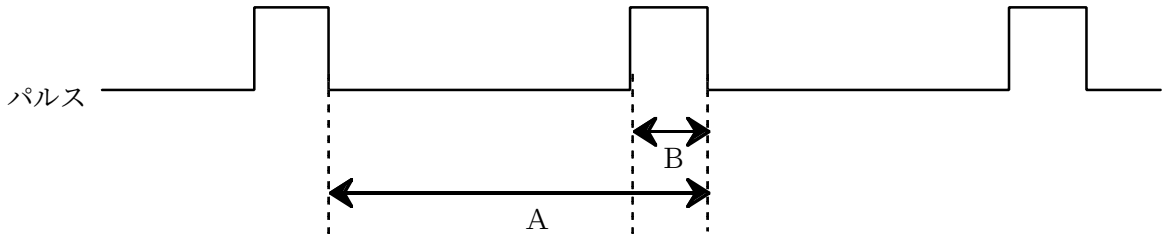


図１　ＡＰＣ出力

使用機能説明

表1、表2、表３に本タスク例の機能割り付けを示します。本タスク例は表１、表２、表３に示すようにＳＨ７０５５に内蔵しているAPC及びATU-Ⅱ、PFCの機能を割り付け、パルスを出力します。

表１　ＡＰＣ機能割り付け

ＡＰＣ機能		機能
端子	PULS0	パルスを出力します。
レジスタ	POPCR	パルスの出力端子の選択をします。

表２　ＡＴＵ-Ⅱ機能割り付け

ＡＴＵ-Ⅱ内蔵機能		機能
レジスタ	PSCR1	ATU-Ⅱのプリスケーラの設定をします。
	TCR11	ATU-Ⅱチャネル11のプリスケーラを設定します。
	TSTR3	ATU-Ⅱチャネル11のカウンタ動作を設定します。
	GR11A	APC出力パルスの立ち上がり時点を設定します。
	GR11B	APC出力パルスの立ち下がり時点を設定します。

表３　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PDIOR	端子の入出力方向を設定します。
PDCRH	端子の機能を設定します。

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスを出力します。

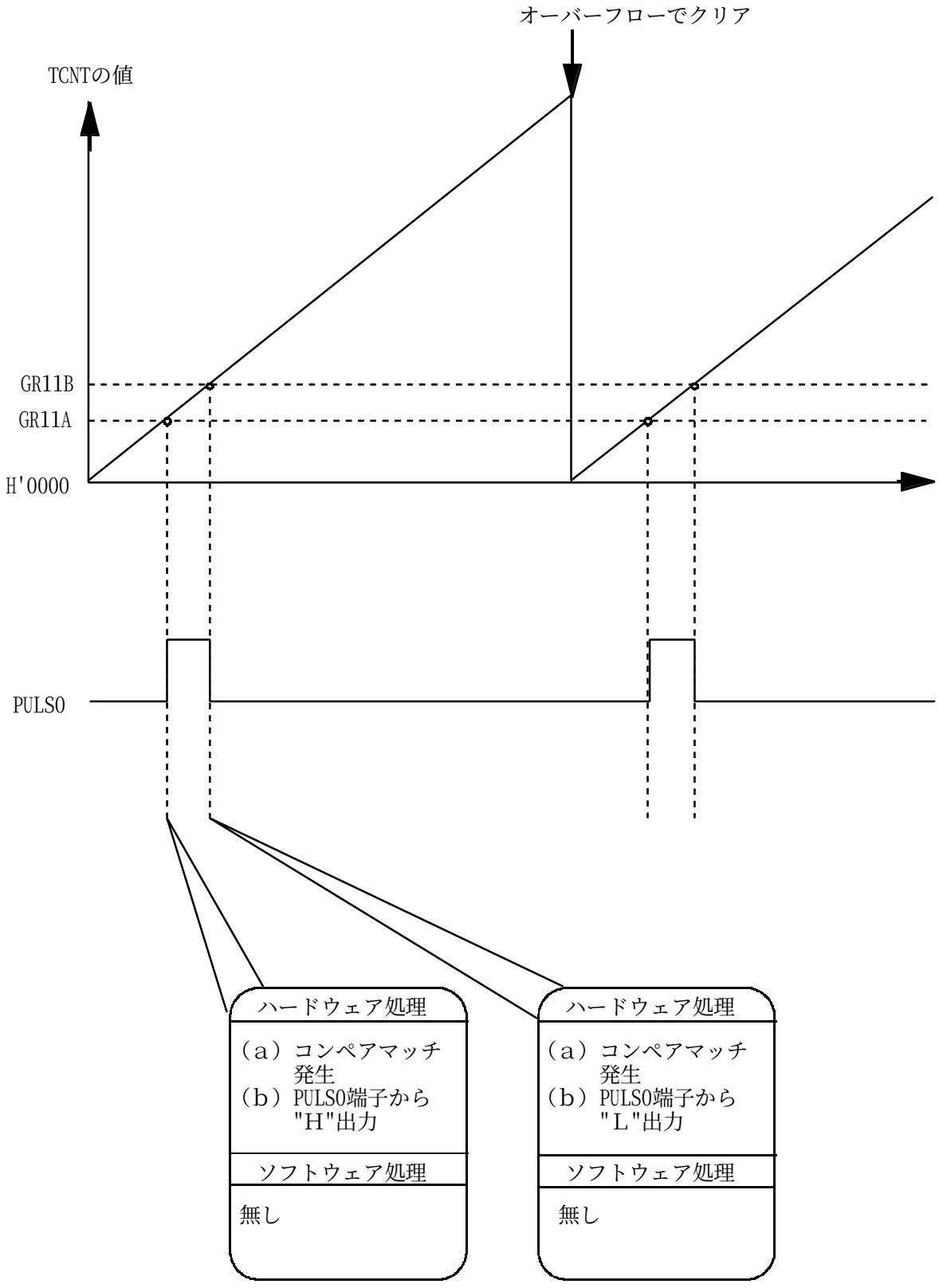


図２　ＡＰＣ出力動作原理

2.17　APC出力	MCU	SH7055	使用機能	APC,ATU-Ⅱ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱ、APCの初期設定を行ないます。

（2）使用変数の説明

本タスクでは変数は使用しておりません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PD.PDIOR	PD0を出力端子に設定します。	0x0100	メインルーチン
PD.PDCRH	端子マルチプレクスをPULS0に設定します。	0x0001	
ATUC.PSCR1	ATU-Ⅱのプリスケーラ1段目をPφ/2に設定します。	0x01	
ATU11.TCR11	ATU-Ⅱチャネル11のプリスケーラをφ'/2に設定します。	0x01	
ATU11.TIOR11	GR11A,GR11Bをアウトプットコンペアレジスタとして設定します。	0x11	
APC.POPCR	パルスの出力端子を設定します。	0x0101	
ATUC.TSTR3	ATU-Ⅱチャネル11のカウント開始を設定します。	0x01	
ATU11.GR11A	パルスの立ち上げ時点を設定します。	0x0064	
ATU11.GR11B	パルスの立ち下げ時点を設定します。	0x00C8	

プログラムリスト

```
/*****************************************************************************/
/*              APC出力                                                      */
/*****************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ        */
/*****************************************************************************/
/*              関数プロトコル宣言                                           */
/*****************************************************************************/
void main(void);
/*****************************************************************************/
/*              メインルーチン                                               */
/*****************************************************************************/
void main(void)
{
        PD.PDCRH = 0x0001;                  /* PULS0機能選択                 */
        PD.PDIOR = 0x0100;                  /* PULS0を出力端子に設定         */
        ATU11.TIOR11 = 0x11;                /* GR11A,11BをOCとして使用       */
        ATU11.GR11A = 0x0064;               /* ﾊﾟﾙｽ立ち上げ時点設定          */
        ATU11.GR11B = 0x00C8;               /* ﾊﾟﾙｽ立ち下がり時点設定        */
        ATUC.PSCR1 = 0x01;                  /* ﾌﾟﾘｽｹｰﾗ1段目Pφ/2(100ns)     */
        ATU11.TCR11 = 0x01;                 /* ﾌﾟﾘｽｹｰﾗ2段目φ'/2(200ns)     */
        APC.POPCR = 0x0101;                 /* ﾊﾟﾙｽの出力端子を設定          */
        ATUC.TSTR3 = 0x01;                  /* ﾁｬﾈﾙ11ｶｳﾝﾄｽﾀｰﾄ                */
        while(1);                           /* 無限ﾙｰﾌﾟ                      */
}
```

2.18 内蔵ウォッチドッグタイマによるパルス出力	MCU	SH7055	使用機能	WDT

仕様

（1）ウォッチドッグタイマを用いて図1のようなパルスを出力します。
（2）40MHz動作時、パルスの周期は10.24ms、デューティは50%です。

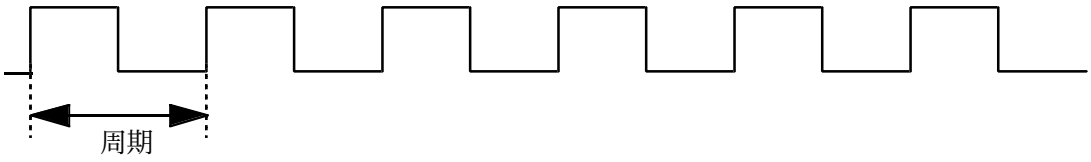


図1　ウォッチドッグタイマによるパルス出力

使用機能説明

表1、表2に本タスク例の機能割付を示します。表1、表2に示すようにSH7055に内蔵されているWDT、PFCの機能を割り付け、パルスの出力を行ないます。

表1　WDT機能割付

WDTレジスタ	機能
TCNT	オーバフロー周期を設定します。
TCSR	インターバルタイマモードに設定します。

表2　PFC機能割付

PFCレジスタ	機能
PAIOR	端子の入出力方向を設定する。
PACRL	端子の機能を選択する。

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理によりパルスを出力します。

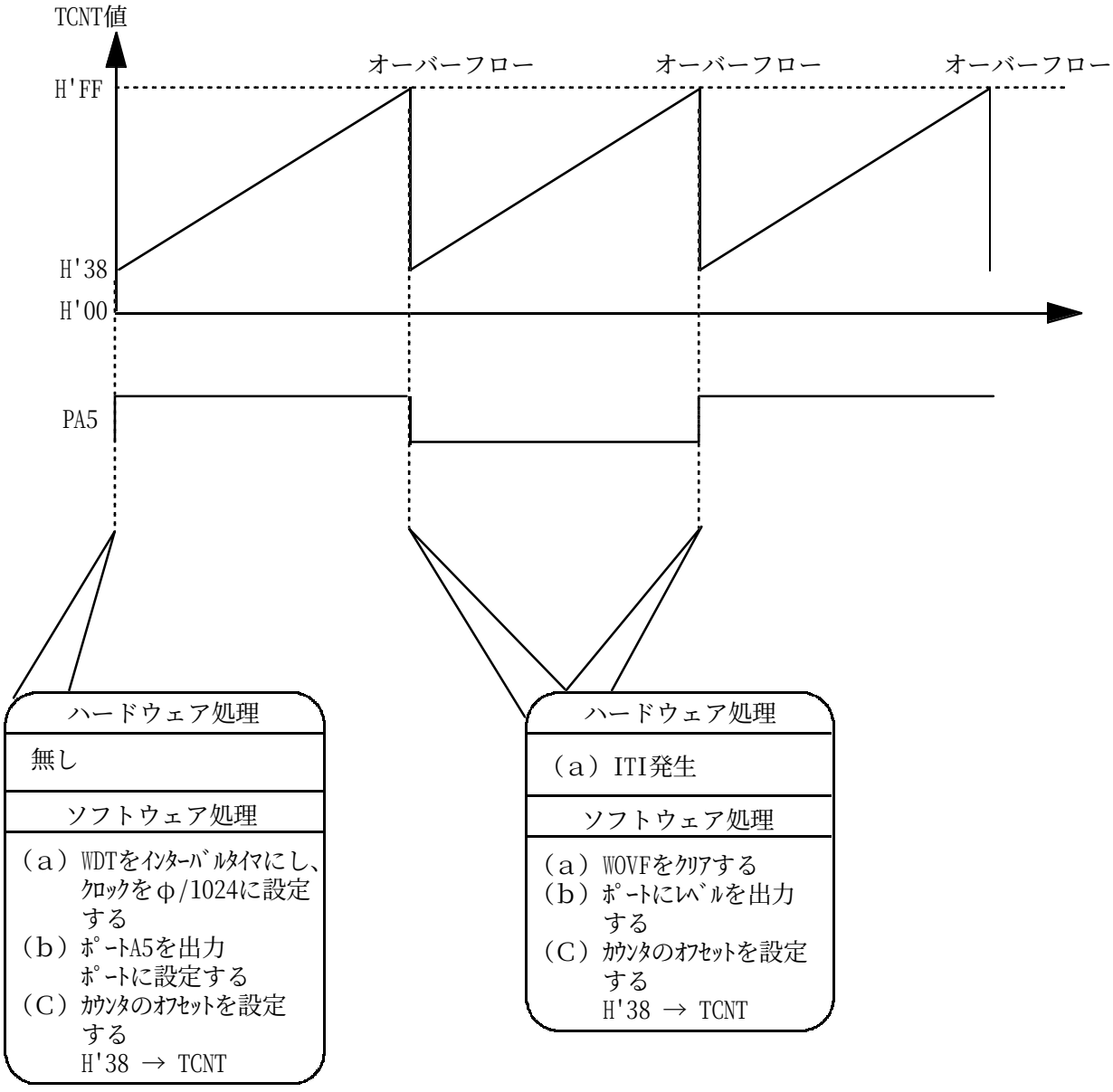


図２　パルス出力原理

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機　　能
メインルーチン	main	ウォッチドッグタイマの初期設定を行なう。
パルス出力	intwdt	ITIで起動し10ms周期でポートにHighとLowを交互に出力する。

（2）引数の説明

ラベル名	機　　能	データ長	使用モジュール名
out_dat	ポート出力データ	unsigned char	パルス出力
dmy_rd	TCSRのダミーリード用	unsigned char	

（3）使用内部レジスタ説明

レジスタ名	機　　能	設定値	使用モジュール名
WDT_TCNT	オーバーフローの周期を5.12msに設定する。	0x5A38	メインルーチン
WDT_TCSR	ウォッチドッグタイマをインターバルタイマモードに設定する。	0xA53D	
PFC.PAIOR	PA5を出力に設定する。	0x0020	
PFC.PACR	端子マルチプレクスをポートの使用にする。	0x0000	
INTC.IPRL	ITIの割り込み優先レベルを15に設定する。	0x00F0	
PFC.PADR	レベル出力を行なう。		パルス出力

（4）使用ＲＡＭ

本タスクでは引数以外のＲＡＭは使用していません。

プログラムリスト

```
/************************************************************************/
/*              内蔵ウォッチドッグタイマによるパルス出力                */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ        */
/************************************************************************/
/*              関数プロトタイプ宣言                                    */
/************************************************************************/
void main( void );
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define out_dat (*(unsigned short *)0xFFFFC000)    /* ﾎﾟｰﾄ出力ﾃﾞｰﾀ格納      */
#define dmy_rd  (*(unsigned char  *)0xFFFFC002)    /* TCSRのﾀﾞﾐｰﾘｰﾄﾞ用       */
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
    WDT_TCSRW = 0xA53D;             /* ｲﾝﾀｰﾊﾞﾙ割り込み発生, φ/1024        */
    WDT_TCNTW = 0x5A38;             /* ｵﾌｾｯﾄの設定(周期10.24ms)            */
    PA.PAIOR = 0x0020;              /* PA5を出力端子に設定する。           */
    PA.PACRL = 0x0000;              /* 端子ﾏﾙﾁﾌﾟﾚｸｽをﾎﾟｰﾄ使用にする。      */
    INTC.IPRL = 0x00F0;             /* WDT割り込み優先ﾚﾍﾞﾙを15に設定する。  */
    out_dat = 0x0020;               /* ﾎﾟｰﾄ出力ﾃﾞｰﾀの設定する。             */
    set_imask(0x0);                 /* 割り込み許可                        */
    while(1);                       /* 無限ﾙｰﾌﾟ（割り込み待ち）            */
}
/************************************************************************/
/*              WDT割り込みルーチン                                     */
/************************************************************************/
#pragma interrupt( intwdt )
void intwdt( void )
{
    dmy_rd = WDT_TCSRR;             /* TCSRのﾀﾞﾐｰﾘｰﾄﾞ                      */
    WDT_TCSRW = 0xA53D;             /* ｵｰﾊﾞｰﾌﾛｰﾌﾗｸﾞｸﾘｱ                     */
    PA.PADR = out_dat;              /* ﾃﾞｰﾀ出力                            */
    out_dat = ~out_dat;             /* 出力ﾃﾞｰﾀの反転                      */
    WDT_TCNTW = 0x5A38;             /* ｵﾌｾｯﾄの設定(周期10.24ms)            */
}
```

2.19　内蔵ｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏを使用したｼｽﾃﾑ監視	MCU	ＳＨ７０５５	使用機能	ＷＤＴ、ＡＴＵ-Ⅱ

仕様

（1）図１に示すようにウォッチドッグタイマがオーバーフローした時に内部リセット信号を発生し、ＳＨ７０５５をリセットします。
（2）オーバーフロー周期は6.6msです。

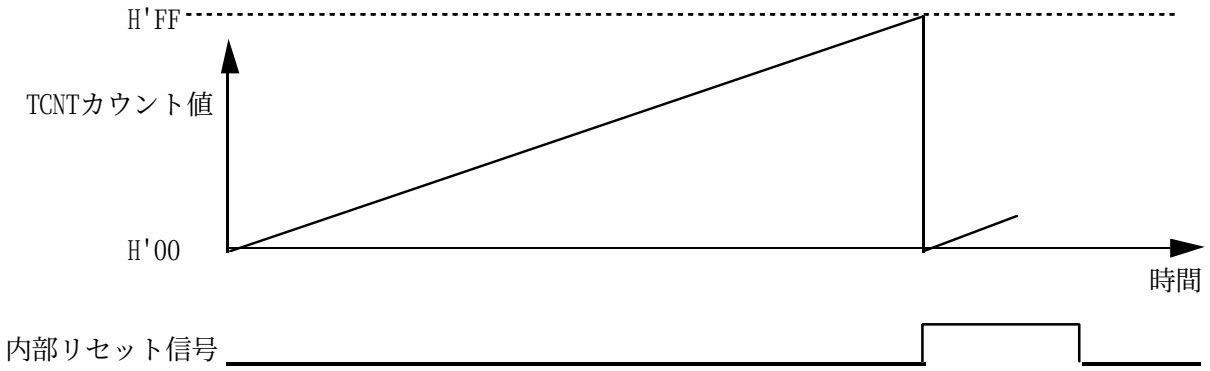


図１　ウォッチドッグタイマによる内部リセット

使用機能説明

表１、表２に本タスク例の機能割付を示します。表１、表２に示すようにＳＨ７０５５に内蔵しているＷＤＴ、ＡＴＵ-Ⅱの機能を割付、パルスの出力を行ないます。

表１　ＷＤＴ機能割付

レジスタ名	機　　能
TCNT	オーバーフローにより、$\overline{\text{WDTOVF}}$が発生する。
TCSR	ウォッチドッグタイマモードを設定する。

表２　ＡＴＵ-Ⅱ機能割付

レジスタ名	機　　能
ITVRR2A	インターバルタイマの周期を設定する。

動作説明

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェア処理によりシステムの監視を行います。

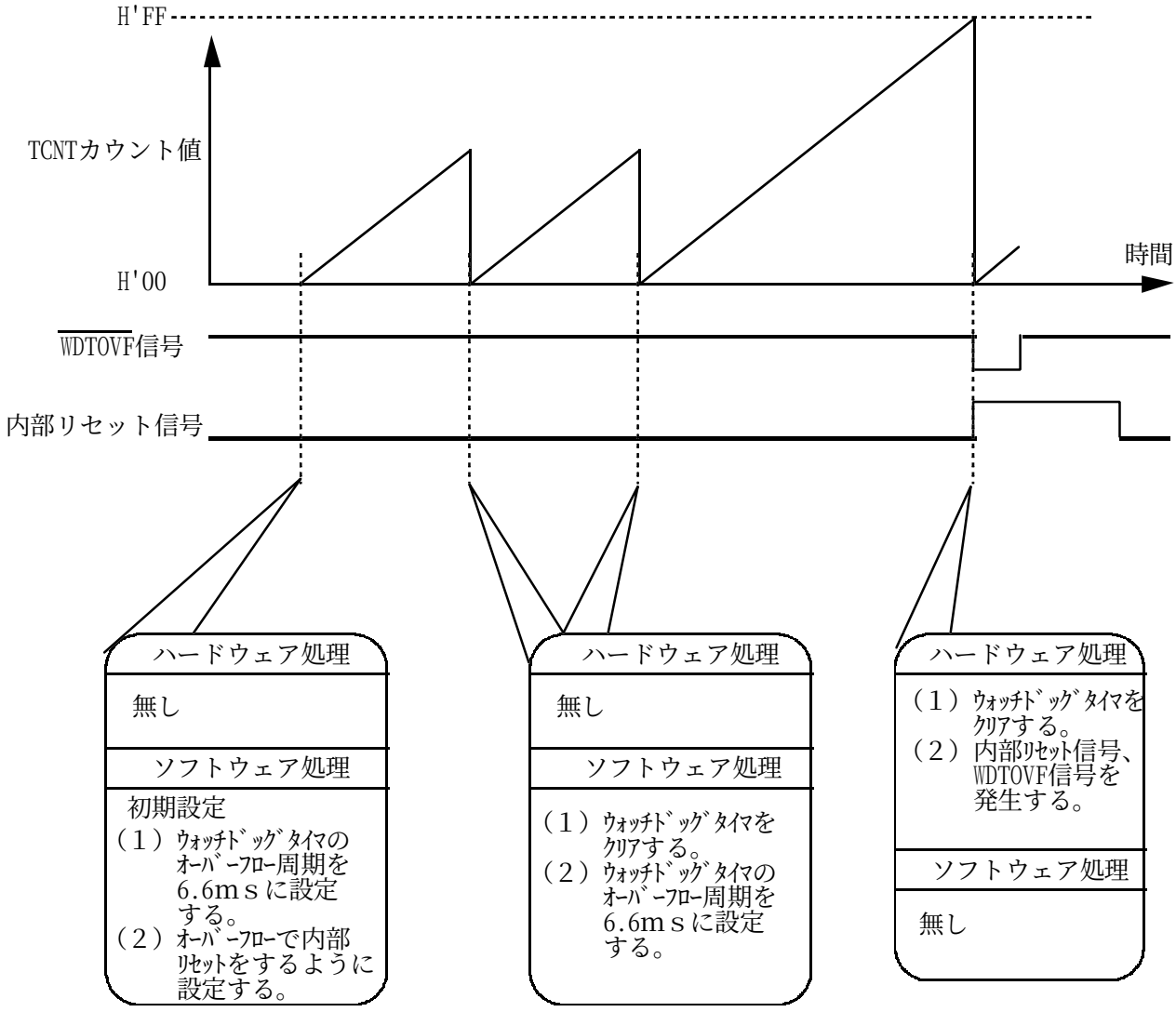


図２　ウォッチドッグタイマ動作原理

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機　　能
メインルーチン	main	ウォッチドッグタイマの初期設定を行なう。
スイッチ検出	int5ms	5ms毎にＳＷのＯＮ－ＯＦＦ検出及び、ウォッチドッグタイマのクリアを行なう。

（2）引数の説明

本タスクでは引数は使用していません。

（3）使用内部レジスタ説明

レジスタ名	機　　能	設定値	使用モジュール名
WDT_TCSR	ウォッチドッグタイマモードを設定します。	0xA57D	メインルーチン
WDT_RSTCSRW	ウォッチドッグタイマのオーバーフローで内部リセットするように設定する。	0x5A5F	
ATUC.PSCR1	チャネル0のプリスケーラをφ/6にする。	0x05	
ATU0.ITVRR2A	TCNT0 ビット13で割り込み発生(820μs)するように設定する。	0x08	
ATUC.TSTR1	TCNT0のカウント開始を設定する。	0x0001	
INT.IPRC	ATU01の割り込み優先レベルを15に設定する。	0x00F0	

（4）使用ＲＡＭ

本タスクではＲＡＭは使用していません。

プログラムリスト

```
/*****************************************************************************/
/*              内蔵ウォッチドッグタイマを使用したシステム監視               */
/*****************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                           /* 周辺レジスタ定義ﾍｯﾀﾞﾌｧｲﾙ      */
/*****************************************************************************/
/*              関数プロトタイプ宣言                                         */
/*****************************************************************************/
void main( void );
/*****************************************************************************/
/*              変数定義                                                     */
/*****************************************************************************/
#define cnt     (*(unsigned char  *)0xFFFFC000)     /* WDTｸﾘｱ回数格納        */
#define dmy_rd (*(unsigned char  *)0xFFFFC001)     /* RSTCSRのﾀﾞﾐｰﾘｰﾄﾞ用     */
/*****************************************************************************/
/*              メインルーチン                                               */
/*****************************************************************************/
void main( void )
{
    ATUC.PSCR1 = 0x05;              /* ﾌﾟﾘｽｹｰﾗ1段目　φ/6                    */
    ATU0.ITVRR2A = 0x08;            /* TCNT0 13ﾋﾞｯﾄ目で割り込み発生(4.92ms) */
    ATUC.TSTR1 = 0x01;              /* ﾁｬﾈﾙ0　ｶｳﾝﾄｽﾀｰﾄ                      */
    WDT_TCSRW = 0xA57D;             /* WDTOVF信号出力,ｵｰﾊﾞｰﾌﾛｰ周期6.6ms      */
    WDT_RSTCSRW = 0x5A5F;           /* 内部ﾘｾｯﾄ,ﾊﾟﾜｰｵﾝﾘｾｯﾄ                  */
    INTC.IPRC = 0x00F0;             /* ATU01割り込み優先ﾚﾍﾞﾙを15に設定       */
    set_imask(0x0);                 /* 割り込み許可                         */
    while(1);                       /* 無限ﾙｰﾌﾟ（割込み待ち）               */
}
/*****************************************************************************/
/*              スイッチ検出ルーチン                                         */
/*****************************************************************************/
#pragma interrupt( int5ms )
void int5ms( void )
{
    ATU0.TSR0 &= 0xFFBF;            /* ｲﾝﾀｰﾊﾞﾙ割込みﾌﾗｸﾞｸﾘｱ                  */
    dmy_rd = WDT_RSTCSRW;           /* RSTCSRのﾀﾞﾐｰﾘｰﾄﾞ                     */
    WDT_RSTCSRW = 0xA500;           /* ｵｰﾊﾞｰﾌﾛｰﾌﾗｸﾞｸﾘｱ                      */
    WDT_TCNTW = 0x5A00;             /* ｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏｸﾘｱ                      */
    cnt++;                          /* WDTｸﾘｱ回数更新                       */
    if( cnt > 1 )
    {
        while(1);
    }
}
```

2.20　クロック同期式シリアル送受信	MCU	ＳＨ７０５５	使用機能	ＳＣＩ

仕様

（1）図１に示すように、ＳＨ７０５５のSCIをクロック同期式モードで使用し、4バイトのデータを同時に送受信します。
（2）転送プロトコルは転送レート1Mbps、データ長8ビット、パリティ無しです。

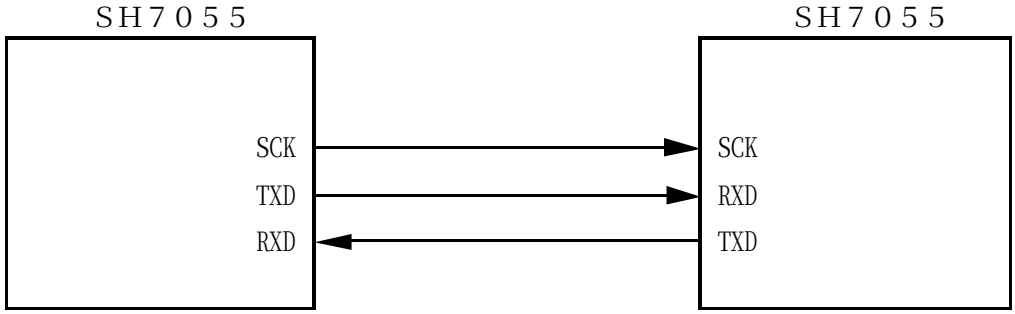


図１　ＳＨ７０５５によるクロック同期式シリアルインタフェースブロック図

使用機能説明

表１、表２に本タスク例の機能割付を示します。表１、表２に示すようにＳＨ７０５５に内蔵しているSCI、PFCの機能を割付、インタフェースを行ないます。

表１　ＳＣＩ機能割付

ＳＣＩ機能		機　　能
端子	RXD	コンソールからデータを受信する。
	TXD	コンソールへデータを送信する。
	SCK	シリアルクロックを入出力する。
レジスタ	SMR	SCIの送信フォーマットを設定する。
	SCR	SCIの割り込みの許可/禁止を設定する。
	SSR	割り込みステータスを設定する。
	RDR	コンソールから受信したデータを設定する。
	TDR	コンソールへ送信するデータを設定する。
	BRR	転送レートを設定する。

表２　ＰＦＣ機能割付

ＰＦＣレジスタ	機　　能
PBIOR	端子の入出力方向を設定する。
PCIOR	
PBCRH	端子の機能を選択する。
PCCR	

動作説明

図２に動作原理を示します。図２に示すタイミングでクロック同期式SCIを制御し、インタフェースを行なっています。

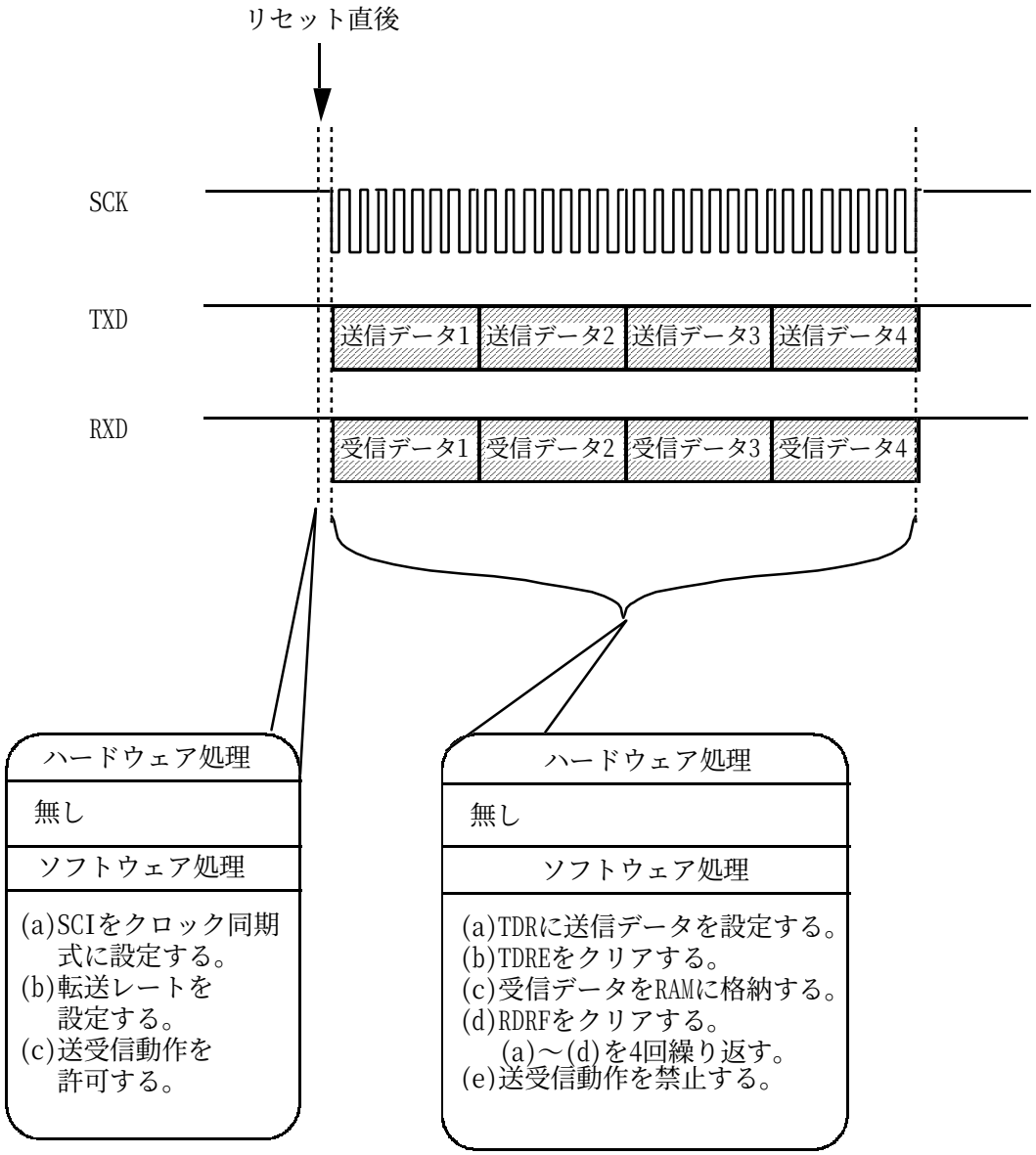


図２　インタフェース動作原理

2.20　クロック同期式シリアル送受信	MCU	SH7055	使用機能	SCI

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機　　能
メインルーチン	main	SCIの初期設定及びデータの送受信を行なう。

（2）引数の説明

本タスクでは引数は使用していません。

（3）使用内部レジスタ説明

レジスタ名	機　　能	設定値	使用モジュール名
SCI1.SMR	SCIのモード(クロック同期式)、転送フォーマット及びボーレイトジェネレータへのクロック選択(φクロック入力)を設定する。	0x80	メインルーチン
SCI1.SCR	送信・受信動作を許可する。	0x30	
SCI1.RDR	受信したデータを設定する。	rdat	
SCI1.TDR	送信するデータを設定する。	tdat	
SCI1.BRR	転送レートを1Mbpsに設定する。	0x04	
PC.PCIOR	RXD1を入力、TXD1を出力に設定する。	0x0001	
PC.PCCR	端子マルチプレクスをRXD1、TXD1の使用にする。	0x0005	
PB.PBIOR	SCK1を出力に設定する。	0x4000	
PB.PBCRH	端子マルチプレクスをSCK0の使用にする。	0x1000	

（4）使用ＲＡＭ

ラベル名	機　　能	データ長	使用モジュール名
lp	SCI送受信の回数を格納する。	unsigned char	メインルーチン
tdat	送信データを格納する。	unsigned short	
rdat	受信データを格納する。	unsigned short	

プログラムリスト

```
/************************************************************************/
/*              クロック同期式シリアル送受信                            */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ        */
/************************************************************************/
/*              関数プロトタイプ宣言                                    */
/************************************************************************/
void main( void );
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define tdat    (*(unsigned long  *)0xFFFFC000)    /* 送信ﾃﾞｰﾀ             */
#define rdat    (*(unsigned long  *)0xFFFFC004)    /* 受信ﾃﾞｰﾀ             */
/************************************************************************/
/*              SCI送受信                                               */
/************************************************************************/
void main( void )
{
    signed int lp;
    PC.PCIOR = 0x0001;              /* TXD1出力,RXD1入力                */
    PC.PCCR  = 0x0005;              /* TXD1,RXD1使用                    */
    PB.PBIOR = 0x4000;              /* SCK1出力                         */
    PB.PBCRH = 0x1000;              /* SCK1使用                         */
    SCI1.SCR = 0x00;                /* 送信・受信動作を禁止する         */
    SCI1.SMR = 0x80;                /* ｸﾛｯｸ同期式､8ﾋﾞｯﾄﾃﾞｰﾀ、ﾊﾟﾘﾃｨなし   */
    SCI1.BRR = 0x04;                /* ﾋﾞｯﾄﾚｰﾄ1Mbps                     */
    for( lp = 1; lp < 1; lp++ );    /* ｳｪｲﾄ                             */
    SCI1.SCR = 0x30;                /* 送信・受信動作を許可する         */
    tdat = 0x5511ffaa;
    for( lp = 0; lp < 4; lp++ )
    {
        while(( SCI1.SSR & 0x80 ) != 0x80 );
        SCI1.TDR = *(unsigned char *)((long)(&tdat) + lp);/* ﾃﾞｰﾀの送信  */
        SCI1.SSR &= 0x7f;                       /* 送信ﾌﾗｸﾞをｸﾘｱする    */
        while(( SCI1.SSR & 0x40 ) != 0x40 );
        *(unsigned char *)((long)(&rdat) + lp) = SCI0.RDR;/* 受信ﾃﾞｰﾀの格納 */
        SCI1.SSR &= 0xbf;                       /* 受信ﾌﾗｸﾞをｸﾘｱする    */
    }
    while(( SCI1.SSR & 0x04 ) != 0x04 );
    SCI1.SCR = 0x00;                        /* 送信・受信動作を禁止する */
}
```

2.21　ＨＣＡＮ送受信	対象	ＳＨ７０５５	使用機能	ＨＣＡＮ

仕様

（１）データフレーム（＊１）の送受信を行ないます。（SH7055を2個使用）
　　図1にデータフレームの仕様を示します。
　　(a) SOF：データフレームの開始を示します。
　　(b) アービトレーションフィールド（＊２）：”101010101010 ”とします。
　　・RTR = ”0 ”：データフレームの選択
　　(c) コントロールフィールド（＊３）：”000001 ”とします。
　　・IDE = ”0 ”：スタンダードフォーマットの選択
　　・R0 = ”0 ”：リザーブビット
　　・DLC =”0001 ”：1バイトのデータ長の設定
　　(d) データフィールド（＊４）：”10101010 ”とします。
　　(e) CRCフィールド（＊5）：HCAN0内部でCRCが自動生成されます。
　　(f) ACKフィールド（＊6）：送信側では”11 ”、受信側では”01 (正常時)”が出力されます。
　　(g) EOF：送信/受信データフレームの完了を示します。
（２）チャネルは送信、受信側のSH7055共にHCAN0を使用します。
（３）通信速度は250Kbps（40MHz動作時）とします。
（４）データ長は、1バイトとします。
（５）メッセージの送信はメールボックス1を用います。
（６）メッセージの受信はメールボックス0を用います。メッセージの受信方式としてはIdentifierを
　　マスクして一致すれば受信とします。
（７）受信データは内蔵RAMに格納します。
（８）図２にCANバスの接続例を示します。

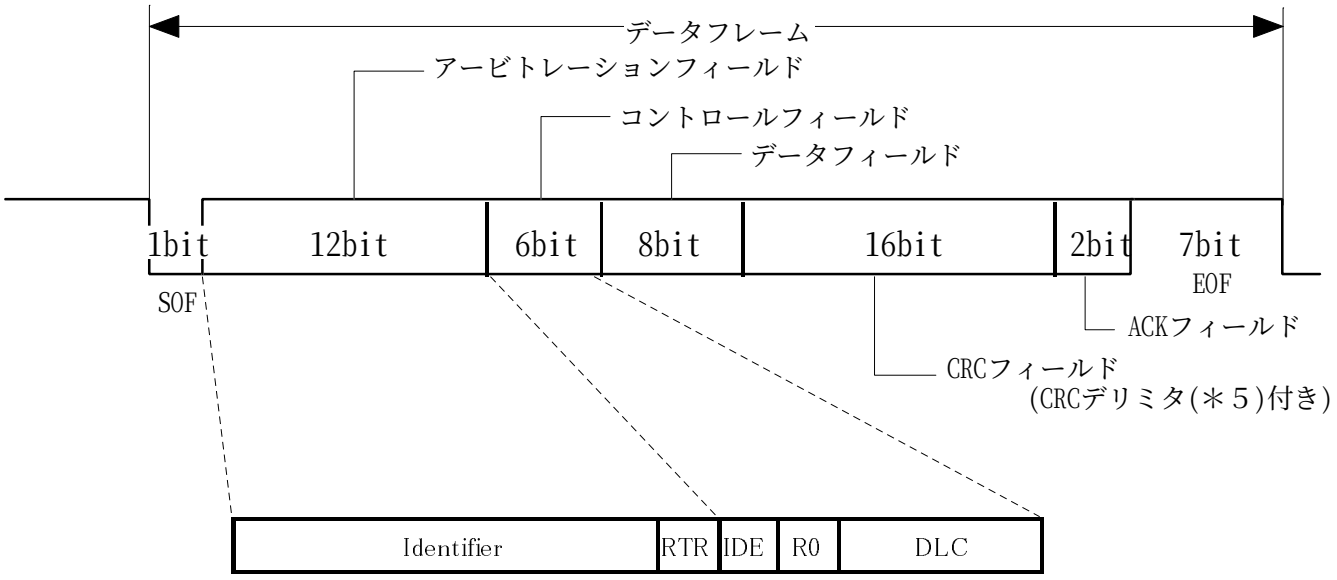
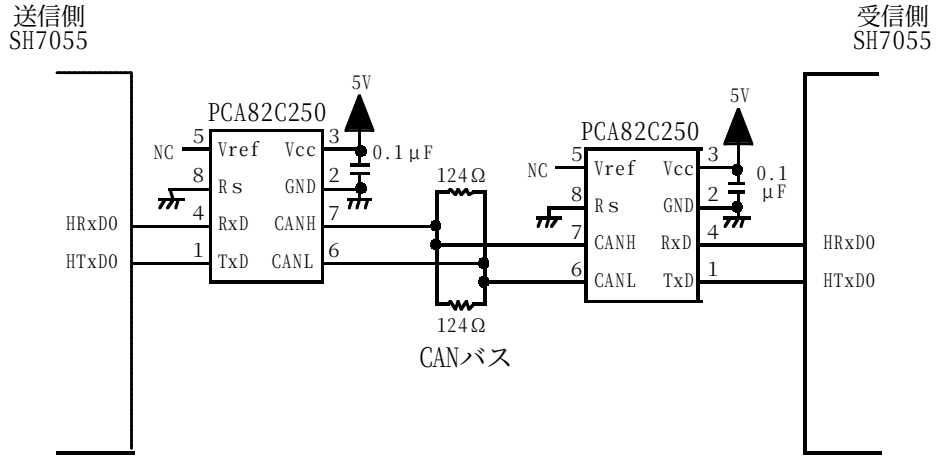


図1　データフレームの仕様

（＊__）は備考を参照して下さい。



※　本LSIとCANバスを接続するためにはバストランシーバICが必要になります。
Philips社製PCA82C50とコンパチブルなものを推奨します。

図２　SH7055を用いたCANインタフェース図

使用機能説明

表１、表２に本タスク例の機能割り付けを示します。本タスク例は表１、表２に示すようにSH7055に内蔵しているHCAN0の機能を割り付け、HCAN0の送受信を行ないます。

表１　　HCAN0機能割り付け

HCAN0レジスタ		機能
端子	HTxD0	メッセージの送信を行ないます。
	HRxD0	メッセージの受信を行ないます。
送受信共通レジスタ	IRR	各割り込み要因のステータスを示します。
	BCR	CANのボーレートプリスケーラ、ビットタイミングパラメータを設定します。
	MBCR	メールボックスの送信/受信を設定します。
	MCR	CANインタフェースの制御を行ないます。
	MC0_1 ～ MC15_8	アービトレーションフィールド 、コントロールフィールドの設定を行ないます。
	MD0_1 ～ MD15_8	データフィールドの設定を行ないます。
送信用レジスタ	TXPR	送信メッセージをメールボックスに格納後送信待ち状態を設定します。
	TXACK	対応するメールボックスの送信メッセージが正常に送信されたことを示します。
受信用レジスタ	LAFMH	受信用メールボックス0のIdentifier用フィルタマスクの設定を行ないます。
	RXPR	対応するメールボックスにデータが正常に受信されたことを示します。

表２　PFC機能割り付け

PFCレジスタ	機能
PLCRH	HTxD0、HRxD0端子の機能を設定します。

動作説明(送信時)

図３に送信時の動作原理を示します。図３に示すようにSH7055のハードウェア処理及びソフトウェア処理によりHCAN0の送信を行ないます。

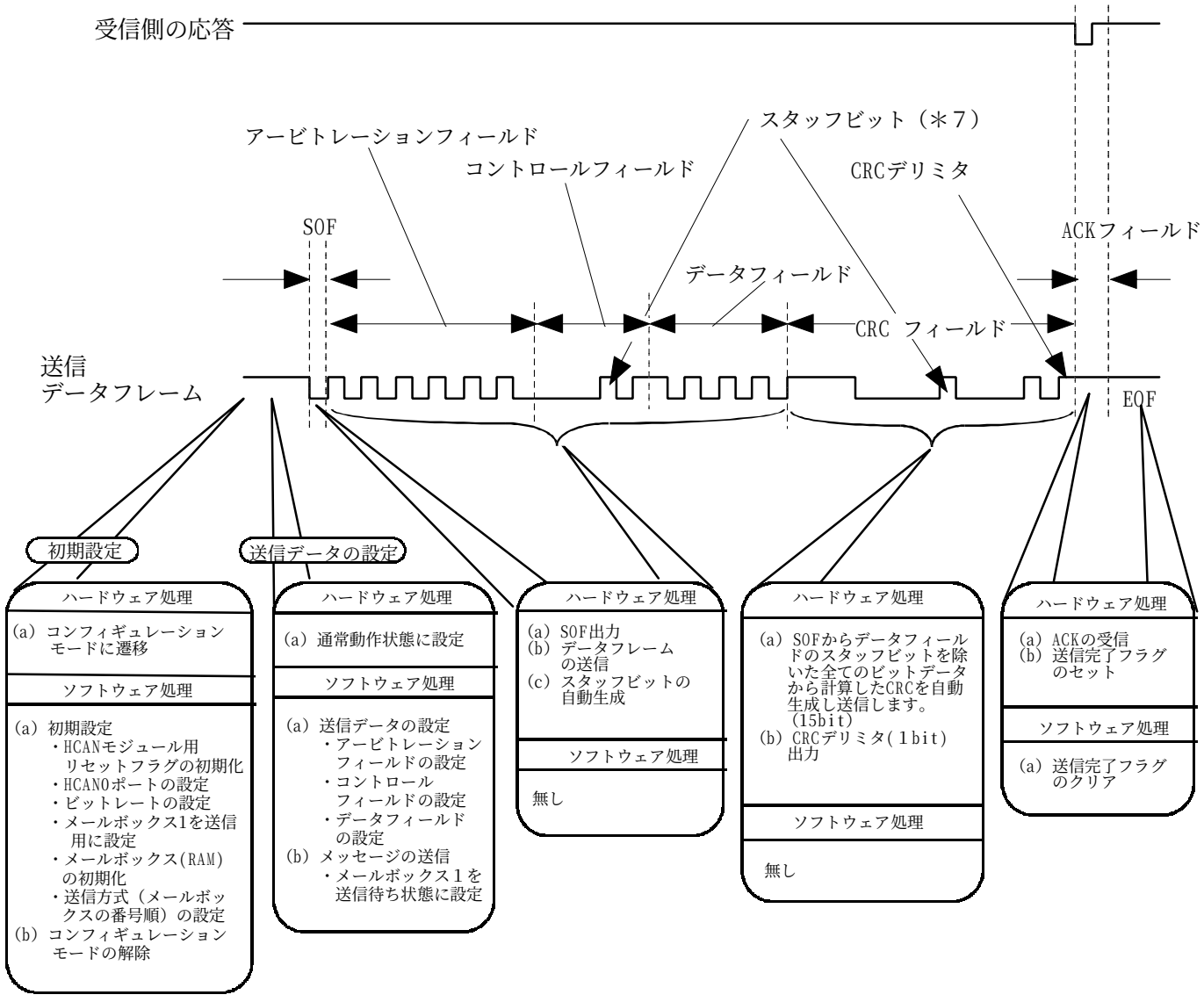


図３　HCAN送信時の動作

（＊__）は備考を参照して下さい。

動作説明(受信時)

図4に受信時の動作原理を示します。図4に示すようにSH7055のハードウェア処理及びソフトウェア処理によりHCAN0の受信を行ないます。

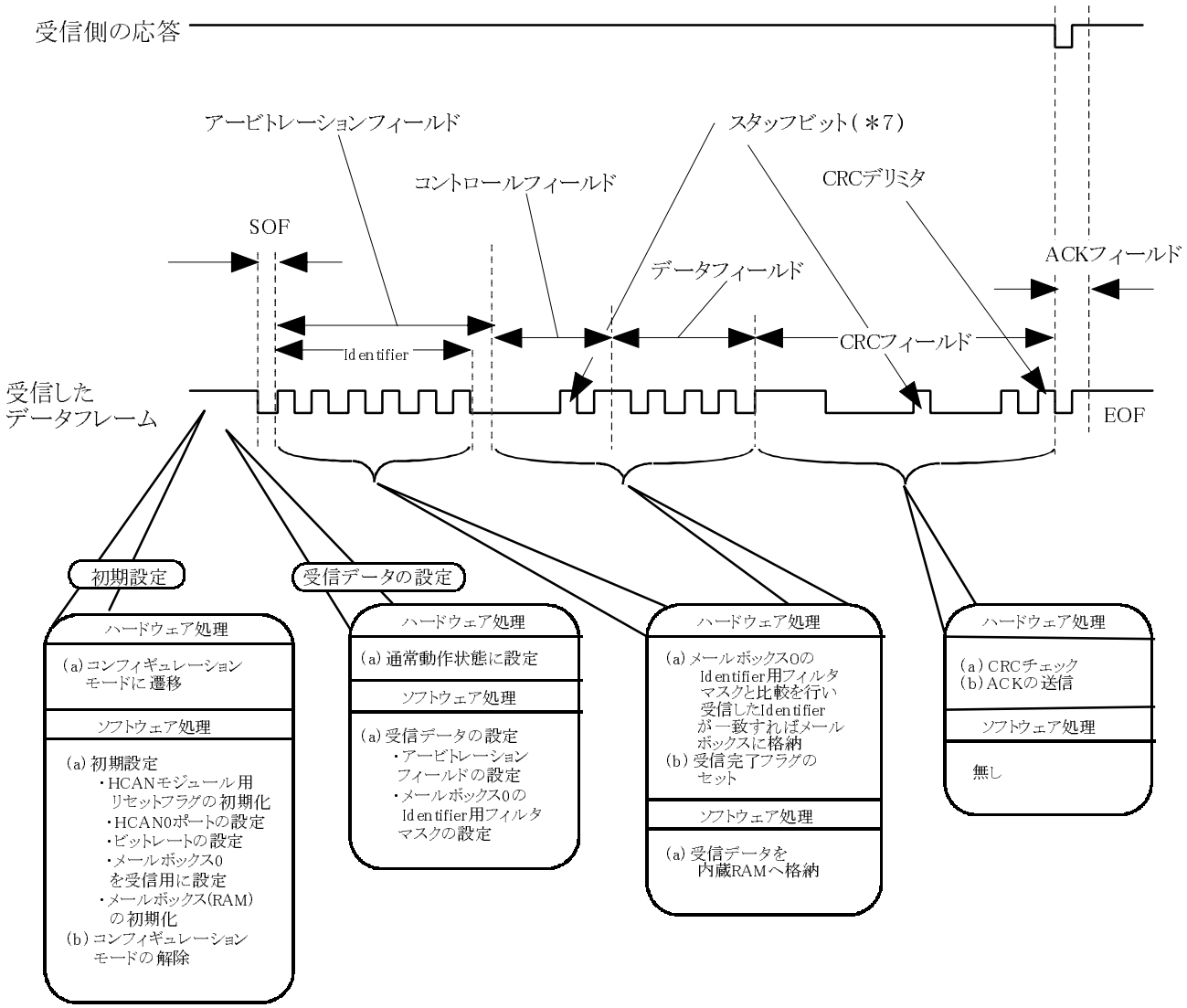


図4　ＨＣＡＮ受信時の動作

(＊　)は備考を参照して下さい。

2.21　ＨＣＡＮ送受信	対象	ＳＨ７０５５	使用機能	ＨＣＡＮ

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	HCAN0の初期設定及び送信、受信の設定を行ないます。

（2）使用変数の説明

ラベル名	機能	データ長	使用モジュール名
MAIL_BOX0	HCAN0_MD0_1のデータを格納します。	unsigned　char	メインルーチン

（3）仕様内部レジスタの説明

レジスタ名	機能	設定値	使用モジュール名
送受信共通設定			メインルーチン
PL.PLCRH	端子マルチプレクスをHTxD0、HRxD0の使用にします。	0x0050	
HCAN0_IRR	HCAN0モジュール用リセットフラグを初期化します。	0x0100	
HCAN0_BCR	HCAN0のビットレートを250Kbpsに設定します。	0x011E	
送信時の設定			
HCAN0_MBCR	メールボックス1を送信用に設定します。	0xFDFF	
HCAN0_MCR	メールボックスの番号順で送信するように設定 及び、リセットリクエストビットをクリアします。	0x04	
HCAN0_MC1_1	1バイトのデータ長を設定します。	0x01	
HCAN0_MC1_5	メールボックス１のデータフレーム、スタンダードフォーマットを選択及び、Identifierを設定します。	0xA0	
HCAN0_MC1_6	メールボックス1のIdentifierを設定します。	0xAA	
HCAN0_MD1_1	メールボックス1の送信データを設定します。	0xAA	
HCAN0_TXPR	メールボックス1を送信待ち状態に設定します。	0x0200	
受信時の設定			
HCAN0_MBCR	メールボックス0を受信用に設定します。	0x0100	
HCAN0_MCR	リセットリクエストビットをクリアします。	0xFE	
HCAN0_MC0_5	メールボックス0のデータフレーム、スタンダードフォーマットを選択及び、Identifierを設定します。	0xA0	
HCAN0_MC0_6	メールボックス0のIdentifierを設定します。	0xAA	
HCAN0_LAFMH	メールボックス0のIdentifier用フィルタマスクを設定します。	0x0000	

2.21　ＨＣＡＮ送受信	対象	ＳＨ７０５５	使用機能	ＨＣＡＮ

送信フローチャート

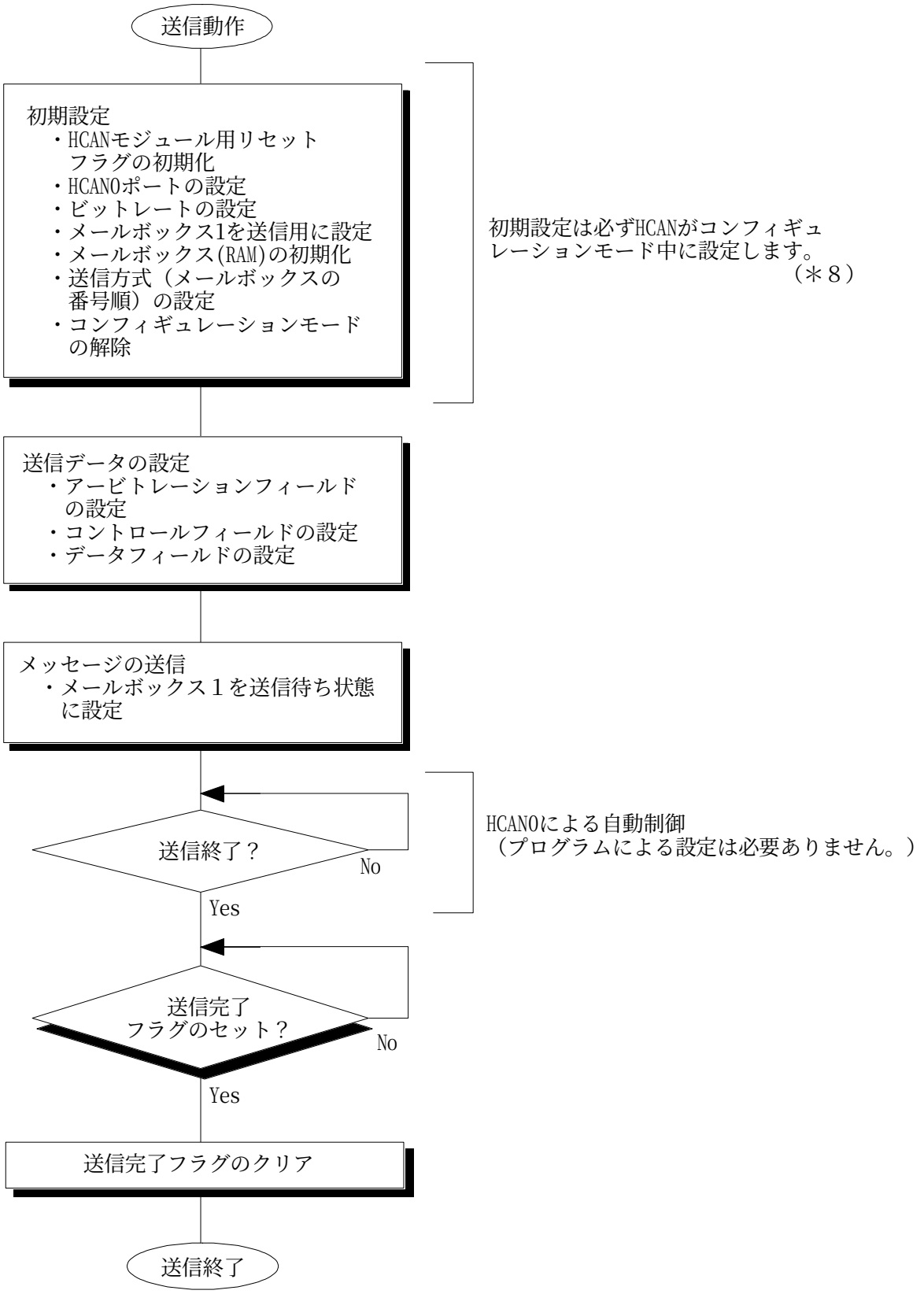


図５　送信時のフローチャート

（＊＿）は備考を参照して下さい。

送信プログラムリスト

```
/*****************************************************************************/
/*                          HCAN  チャネル0送信プログラム                    */
/*****************************************************************************/
#include <stdio.h>                      /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ                */
#include <machine.h>                    /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ                */
#include "7055.h"                       /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ              */
/*****************************************************************************/
/*                          関数プロトコル宣言                               */
/*****************************************************************************/
void main( void );
/*****************************************************************************/
/*                          メインルーチン                                   */
/*****************************************************************************/
void main(void)
{
        short COUNT;
/* 初期設定 */
        HCAN0_IRR = 0x0100;             /* HCANﾓｼﾞｭｰﾙ用ﾘｾｯﾄﾌﾗｸﾞの初期化          */
        PL.PLCRH = 0x0050;              /* HTxD0､HRxD0を選択                    */
        HCAN0_BCR = 0x011E;             /* ﾋﾞｯﾄﾚｰﾄ 250Kbps                      */
        HCAN0_MBCR = 0xFDFF;            /* ﾒｰﾙﾎﾞｯｸｽ1を送信用に設定              */
        for( COUNT = 0; COUNT < 128; COUNT++ )    /* ﾒｰﾙﾎﾞｯｸｽ(RAM)の初期化      */
                {
                        *(char*)(&HCAN0_MC0_1 + COUNT) = 0x00;
                }
        for( COUNT = 0; COUNT < 128; COUNT++ )    /* ﾒｰﾙﾎﾞｯｸｽ(RAM)の初期化      */
                {
                        *(char*)(&HCAN0_MD0_1 + COUNT) = 0x00;
                }
        HCAN0_MCR = 0x04;               /* ﾒｰﾙﾎﾞｯｸｽの番号順で送信、ｺﾝﾌｨｷﾞｭﾚｰｼｮﾝﾓｰﾄﾞの解除 */
/* 送信データの設定 */
        HCAN0_MC1_5 = 0xA0;             /* ﾃﾞｰﾀﾌﾚｰﾑ、ｽﾀﾝﾀﾞｰﾄﾞﾌｫｰﾏｯﾄ、Identifierの設定  */
        HCAN0_MC1_6 = 0xAA;             /* Identifierの設定                     */
        HCAN0_MC1_1 = 0x01;             /* ﾃﾞｰﾀ長：1ﾊﾞｲﾄ                        */
        HCAN0_MD1_1 = 0xAA;             /* 送信ﾃﾞｰﾀ：10101010                   */
/* メッセージの送信 */
        HCAN0_TXPR = 0x0200;            /* ﾒｰﾙﾎﾞｯｸｽ1を送信待ち状態に設定        */
        while((HCAN_TXACK & 0x0200) != 0x0200);  /* 送信完了待ち                */
/* 送信完了フラグのクリア */
        HCAN0_TXACK = 0x0200;           /* 送信完了ﾌﾗｸﾞのｸﾘｱ                    */
        while(1);
}
```

受信フローチャート

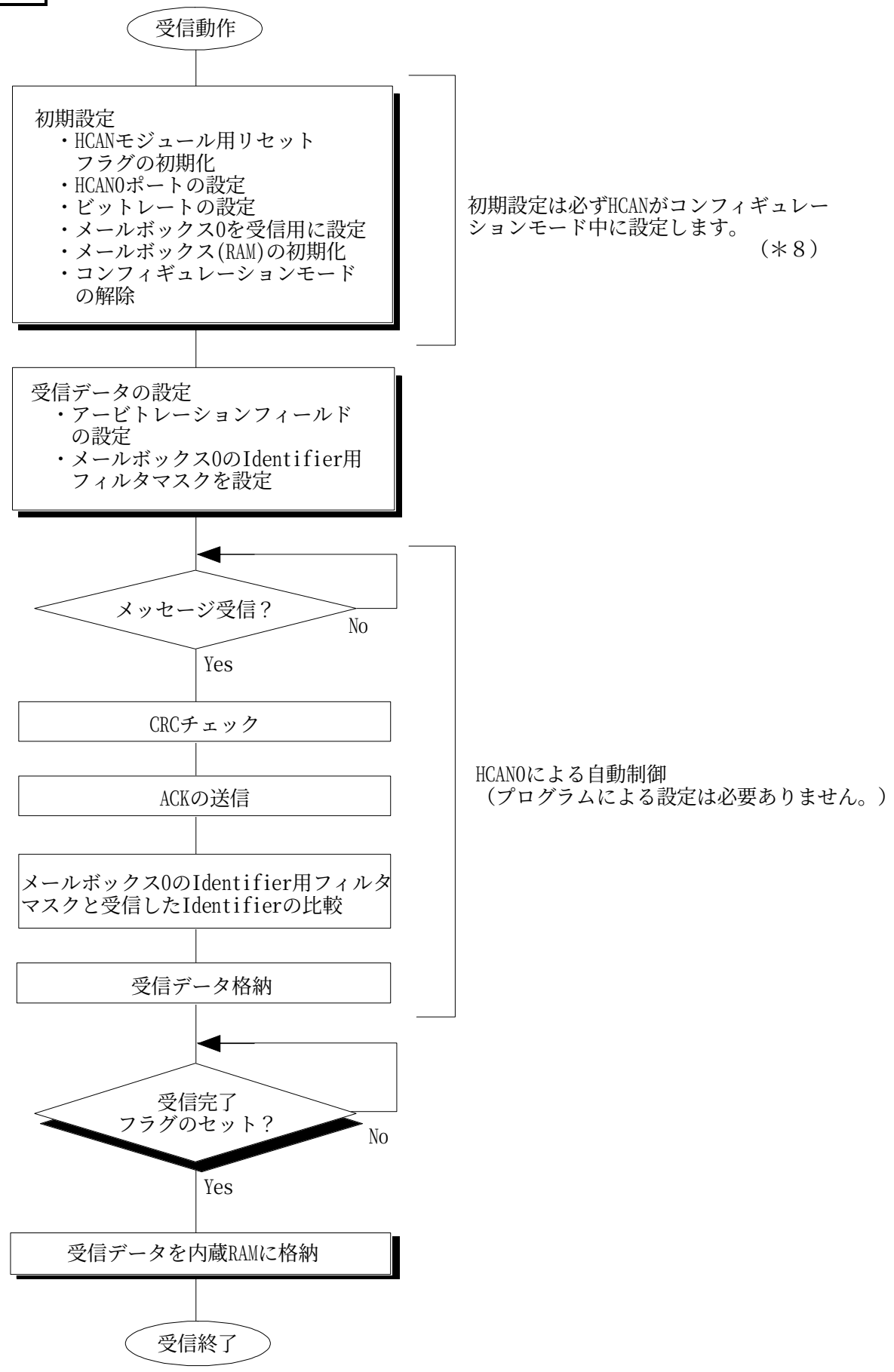


図６　受信時のフローチャート

（＊＿）は備考を参照して下さい。

受信プログラムリスト

```
/*****************************************************************************************/
/*                                HCAN  チャネル0受信プログラム                          */
/*****************************************************************************************/
#include <stdio.h>                              /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ                    */
#include <machine.h>                            /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ                    */
#include "7055.h"                               /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ                  */
/*****************************************************************************************/
/*                                関数プロトコル宣言                                     */
/*****************************************************************************************/
extern  void main( void );
/*****************************************************************************************/
/*                                定数定義                                               */
/*****************************************************************************************/
#define MAIL_BOX0(*(unsigned char*)0xFFFFC000)/* ﾒｰﾙﾎﾞｯｸｽ0受信ﾃﾞｰﾀ格納                */
/*****************************************************************************************/
/*                                メインルーチン                                         */
/*****************************************************************************************/
void main(void)
{
        short COUNT;
/* 初期設定 */
        HCAN0_IRR = 0x0100;                     /* HCAN0ﾓｼﾞｭｰﾙ用ﾘｾｯﾄﾌﾗｸﾞの初期化          */
        PL.PLCRH = 0x0050;                      /* HTxD0､HRxD0を選択                      */
        HCAN0_BCR = 0x011E;                     /* ﾋﾞｯﾄﾚｰﾄ 250Kbps                        */
        HCAN0_MBCR = 0x0100;                    /* ﾒｰﾙﾎﾞｯｸｽ0を受信に設定                  */
        for( COUNT = 0; COUNT < 128; COUNT++ )/* ﾒｰﾙﾎﾞｯｸｽ(RAM)の初期化                  */
                {
                        *(char*)(&HCAN0_MC0_1 + COUNT) = 0x00;
                }
        for( COUNT = 0; COUNT < 128; COUNT++ )/* ﾒｰﾙﾎﾞｯｸｽ (RAM)の初期化                 */
                {
                        *(char*)(&HCAN0_MD0_1 + COUNT) = 0x00;
                }
        HCAN0_MCR &= 0xFE;                      /* ｺﾝﾌｨｷﾞｭﾚｰｼｮﾝﾓｰﾄﾞの解除                 */
/* 受信データの設定 */
        HCAN0_MC0_5 = 0xA0;                     /* ﾃﾞｰﾀﾌﾚｰﾑ、ｽﾀﾝﾀﾞｰﾄﾞﾌｫｰﾏｯﾄ、Identifierの設定*/
        HCAN0_MC0_6 = 0xAA;                     /* Identifierの設定                       */
        HCAN0_LAFMH = 0x0000;                   /* ﾒｰﾙﾎﾞｯｸｽ0のIdentifier用のﾌｨﾙﾀﾏｽｸの設定  */
        while((HCAN0_RXPR & 0x0100) != 0x0100);/* 受信完了待ち                           */
/* 受信データを内蔵RAMに格納 */
        MAIL_BOX0 = HCAN0_MD0_1;                /* 受信ﾃﾞｰﾀの格納                         */
        while(1);
}
```

備考

（＊1）データフレーム　：送信元から送信先へ転送したいデータです。

（＊2）アービトレーションフィールド　：メッセージ固有のID及びデータフレームまたはリモートフレームの設定をします。

（＊3）コントロールフィールド　：転送するデータ長及びスタンダードフォーマットまたはエクステンデッドフォーマットの設定をします。

（＊4）データフィールド　：メッセージの内容(送信したいデータ)を設定します。

（＊5）CRCフィールド　：HCAN内部でSOFからデータフィールドのスタッフビットを除いた全てのビットデータからCRCが自動生成され、送信メッセージのエラーを検出します。
15bitのCRCと1bitのCRCデリミタから成ります。
CRCデリミタはCRC後に必ず”1 ”が出力されます。

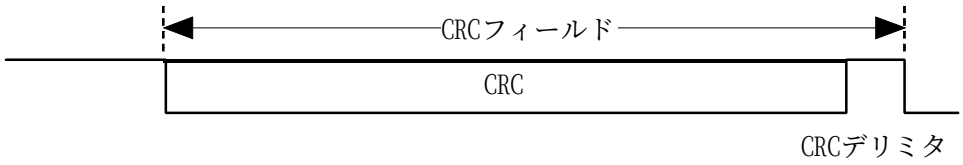


図7　CRCフィールド

CRCについて

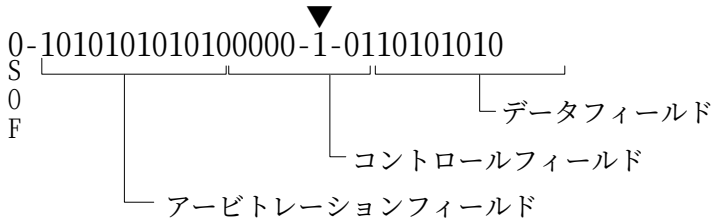
伝送しようとするデータ多項式$P(X)$にX^nを乗じ，さらにこの$X^n \cdot P(X)$を生成多項式$G(X)$で除し，余りである$R(X)$を求めます。情報を送りだす側では，$X^n \cdot P(X)$に求められた$R(X)$をチェックビットとして付加し電信情報$Tx(X)$として送りだします。これが、図3の送信データフレームです。
また，情報を受ける側では受けた受信情報$Rx(X)$を生成多項式$G(X)$で除し，余りを求めます。この余りがゼロとなった場合は，情報伝送は正しく行われたとみなし，余りが出た場合は伝送された情報に誤りがあったと判断します。

$P(X)$ = {スタッフビットを除いたSOFからデータフィールドまで}
$G(X) = X^{15}+X^{14}+X^{10}+X^8+X^7+X^4+X^3+1$
R = 15

※$G(X)$ はCRCを生成する多項式としてCANプロトコルにて規定されています。

例として図3の送信データフレームについて説明します。
設定値
SOF： 0
アービトレーションフィールド：101010101010
コントロールフィールド：000001
データフィールド：10101010

送信されるデータのSOFからデータフィールドは（▼スタッフビット）



で、対象となるデータはスタッフビットを除いた
01010101010100000110101010

即ち、$P(X) = X^{25}+X^{23}+X^{21}+X^{19}+X^{17}+X^{15}+X^8+X^7+X^5+X^3+X^1$
となります。

ここで、$P(X)$ にX^{15}を掛け、
$X^{15} \cdot P(X) = X^{40}+X^{38}+X^{36}+X^{34}+X^{32}+X^{30}+X^{23}+X^{22}+X^{20}+X^{18}+X^{16}$
とし、この値を
$G(X) = X^{15}+X^{14}+X^{10}+X^{8}+X^{7}+X^{4}+X^{3}+1$
で除します。

$X^{15} \cdot P(X) = X^{40}+X^{38}+X^{36}+X^{34}+X^{32}+X^{30}+X^{23}+X^{22}+X^{20}+X^{18}+X^{16}$

⇩

P[40:0]=10101010101000000110101010000000000000000

$G(X) = X^{15}+X^{14}+X^{10}+X^{8}+X^{7}+X^{4}+X^{3}+1$

⇩

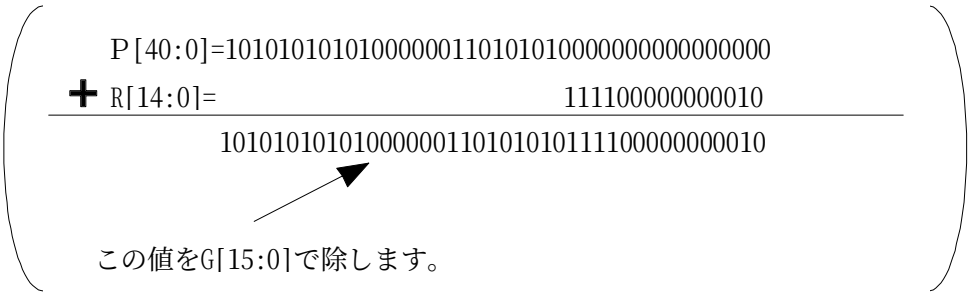
G[15:0]=1100010110011001

```
                                          商
                  ________________________________________________
1100010110011001 ) 10101010101000000110101010000000000000000
                   1100010110011001
                    1101111001110010
                    1100010110011001
                       1101111101011110
                       1100010110011001
                          1101011000111101
                          1100010110011001
                             1001110100100010
                             1100010110011001
                              1011000101110110
                              1100010110011001
                               1110100111011110
                               1100010110011001
                                 1011000100011100
                                 1100010110011001
                                  1110100100001010
                                  1100010110011001
                                    1011001001001100
                                    1100010110011001
                                     1110111110101010
                                     1100010110011001
                                       1010100011001100
                                       1100010110011001
                                        1101101010101010
                                        1100010110011001
                                           1111100110011000
                                           1100010110011001
                                             111100000000010  ◀── 余り：R[14:0]
```

除数と非除数で
EORをとります。

計算の結果、次の値がえられます。
　R[14:0]=111100000000010
つまり、これがCRCの値となり、データフィールド後に付加され電信情報$Tx(X)$として送信されます。
実際にはスタッフビット(▲)とCRCデリミタ(△)が付加され、
　1111000001000010１がCRCフィールド上（例：図3、図9）で送信されます。
　　　　▲　　　△

次に、受信した受信情報*Rx(X)*に誤りがあるか計算してみます。
*Rx(X)*とは、前記の計算により求まった余りをX^{15}・*P(X)*に加算した値であり、この値を*G(X)*で除します。



以上の計算から余りはゼロとなり情報伝達は正しいとみなされます。
また、図3、図9のCRCフィールド上のCRCの値に誤りは無く情報伝達は正しく行なわれたことが分かります。

備考

（＊６）ACKフィールド　：正常受信確認用です。
1bitのACKスロット、1bitのACKデリミタから成ります。

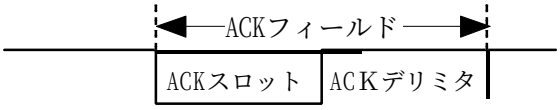


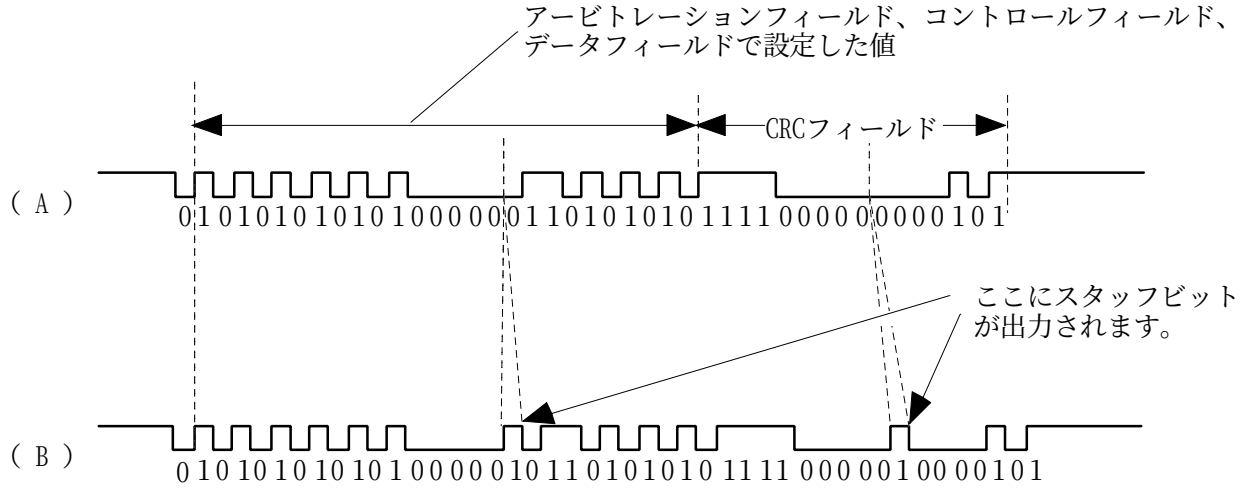
図８　ACKフィールド

・SH7055受信側がCRCチェックでエラーを検出した場合はACKスロット=Highを、検出しなかった　場合はACKスロット=Lowを出力します。

（＊７）スタッフビット　：データフレーム上でLowが5bit続いた場合、次の1bitは必ずHighが出力されます。
同様にHighが5bit続いた場合、次の1bitは必ずLowが出力されます。
　これらの様にスタッフビットが出力される場合データフレームのビット長は、スタッフビットの分だけ長くなります。

例として、設定値を
アービトレーションフィールド：”101010101010 ”
コントロールフィールド：”000001 ”
とすると、”101010101010000001 ”となり、最下位から2bit目（０が５つ続いた後）に
スタッフビット（▼）が出力されます。　　▼
　よって、CANバス上に送信される値は”1010101010100000101 ”となり、
スタッフビットの１bit分だけ長くなります。
また、図3の送信データフレームを例にしてみると図9の様にスタッフビットが2bit出力されます。

※CRCの値は（＊５）を参考にして下さい。



(A)の送信したいデータフレームに対して、CANバス上ではスタッフビットの乗ったデータフレーム(B)が送信されます。また、(B)ではデータフレームの長さもスタッフビットの分だけ長くなっています。

図９　スタッフビット

（＊８）コンフィギュレーションモード　：HCANモジュールがリセット状態のことであり、解除はマスタコントロールレジスタ（MCR）のリセットリクエストビット（MCR0）をクリアすることで行ないます。

2.22　単一モードによるA／D変換	MCU	ＳＨ７０５５	使用機能	ＡＴＵ－Ⅱ、A／D

仕様

（1）図１に示すように、1チャネルに入力される電圧をＳＨ７０５５のA/D変換器を使用し、測定します。
（2）測定結果はRAMに格納します。
（3）入力電圧は2～5Vの範囲です。

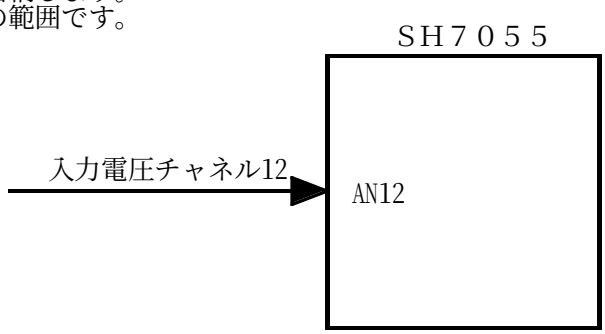


図1　ＳＨ７０５５による電圧の測定ブロック図

使用機能説明

表１、表2に本タスク例の機能割り付けを示します。本タスク例は表１、表2に示すようにＳＨ７０５５に内蔵されているATU-Ⅱ、A/D変換器、PFCの機能を割り付け、A/D変換を行ないます。

表1　ＡＴＵ-Ⅱ機能割り付け

ＡＴＵ-Ⅱ内蔵機能	機能
PSCR1	ATU-Ⅱのプリスケーラの設定をします。
ITVRR2B	TCNT0に対応したビットの設定により、割り込み要求、A/D変換の起動を制御します。
TSTR1	ATU-Ⅱチャネル0のカウンタ動作を設定します。

表2　Ａ／Ｄ機能割り付け

Ａ／Ｄ変換機能	機能
ADTRGR1	A/Dのトリガの選択を行ないます。
AD.ADCSR1	A/Dの変換モード及びA/D割り込み要求の許可／禁止、測定端子を設定します。
AD.ADCR1	A/D変換の開始及び動作クロックを設定します。
AD.ADDR12	A/D変換結果を格納します。

表3　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PA.PAIOR	端子の入出力方向を設定します。
PA.PADR	データを格納します。
PA.PACRL	端子の機能を設定します。

動作原理

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理により1チャネルのA/D変換を行ないます。

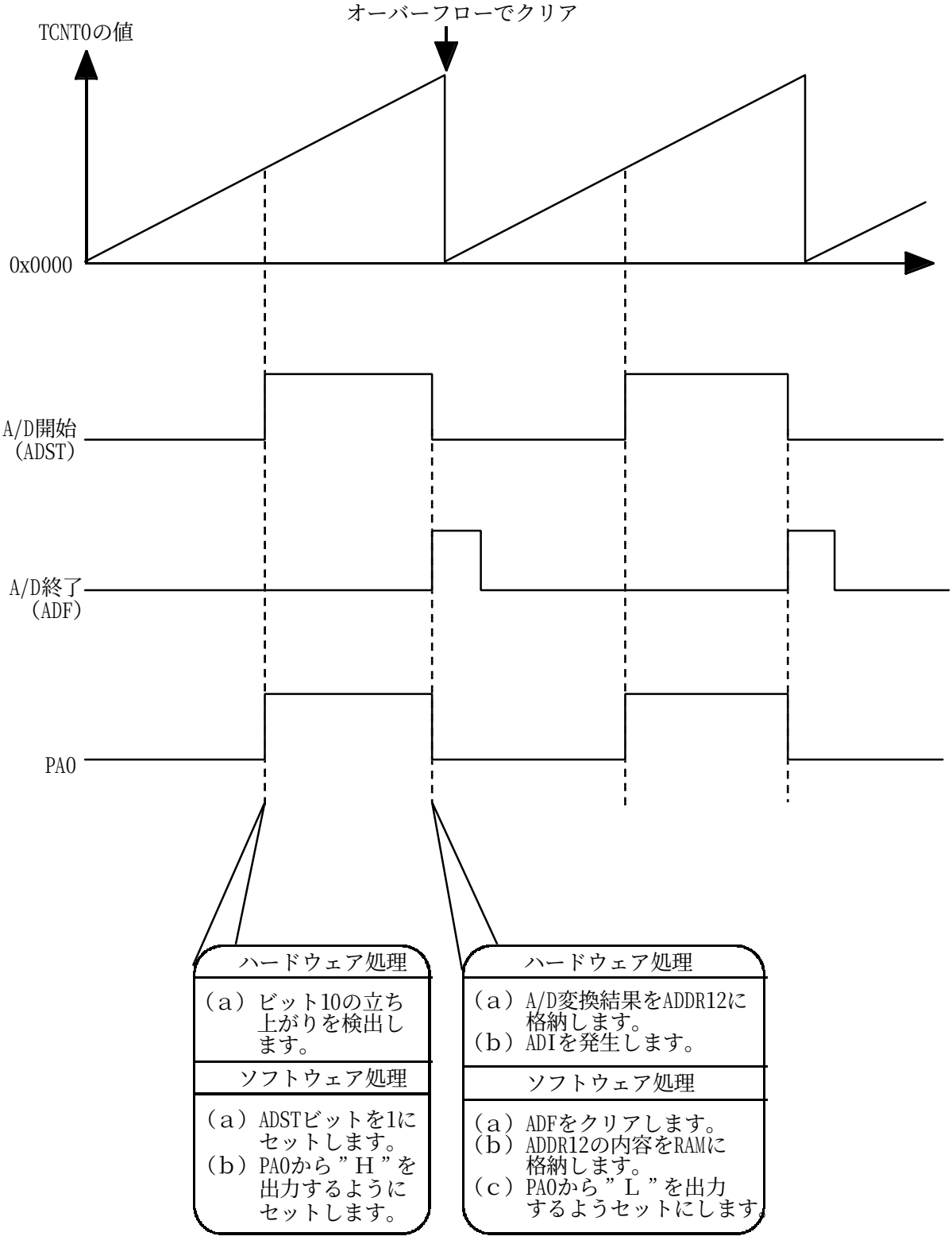


図２　Ａ／Ｄ変換動作原理

2.22　単一モードによるA／D変換	MCU	SH７０５５	使用機能	ＡＴＵ－Ⅱ、A／D

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	ATU-Ⅱ及びA/Dの初期設定を行ないます。
インターバル割り込みルーチン	ITV1	ITV1により起動し、A/D変換開始の認識フラグの設定を行ないます。
A/D変換終了割り込みルーチン	ADI1	ADF1により起動し、A/D変換結果をRAMに格納します。

（2）使用変数の説明

本タスクでは変数は使用しておりません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PA.PACRL	端子マルチプレクスをPA0端子に設定します。	0x0000	メインルーチン
PA.PAIOR	PA0を出力端子に設定します。	0x0001	
PA.PADR	PA0の初期状態を設定します。	0x0000	
ATUC.PSCR1	ATU-Ⅱのプリスケーラ1段目をPφ/10に設定します。	0x09	
ADTRGR1	ATU-Ⅱのインターバルタイマ割り込みによるA/Dの起動を設定します。	0x7F	
ATU0.ITVRR2B	TCNT0ビット10の立ち上がりによるA/D変換器の起動及び割り込みを設定します。	0x11	
AD.ADCSR1	A/D変換モードを単一モードに設定、測定端子AN12によるA/D変換終了割り込み要求を許可します。	0x40	
AD.ADCR1	ATU-Ⅱトリガによる A/D変換の開始を許可、変換時間を268ステートに設定します。	0xCF	
ATUC.TSTR1	ATU-Ⅱチャネル0のカウント開始を設定します。	0x01	
INTC.IPRC	ATU01の割り込み優先レベルを14に設定します。	0x00E0	
INTC.IPRJ	AD1の割り込み優先レベルを15に設定します。	0x00F0	
AD.ADDR12	A/D変換結果を格納します。	-	A/D変換終了割り込みルーチン

プログラムリスト

```
/************************************************************************/
/*              A/D コンバータ                                          */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ       */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main(void);
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define ad_data  (*(unsigned short *)0xFFFFC000)        /* A/Dﾃﾞｰﾀ     */
/************************************************************************/
/*              Channel_1_A/D Main Routine                              */
/************************************************************************/
void main(void){
        PA.PACRL = 0x0000;              /* PA0を汎用入出力機能に設定        */
        PA.PADR = 0x0000;               /* Low出力(初期状態)                */
        PA.PAIOR = 0x0001;              /* PA0を出力端子に設定              */
        ATUC.PSCR1 = 0x09;              /* ﾌﾟﾘｽｹｰﾗ1段目　Pφ/10(250ns)      */
        ADTRGR1 = 0x7F;                 /* ATU-Ⅱ ch0 ｲﾝﾀｰﾊﾞﾙﾀｲﾏにより起動  */
        ATU0.ITVRR2B = 0x11;            /* TCNT0のﾋﾞｯﾄ10の立ち上がりでA/D起動、
                                           割込み許可                       */
        AD.ADCSR1 = 0x40;               /* AN12 A/D終了割込み許可           */
        AD.ADCR1 = 0xCF;                /* ATU-Ⅱﾄﾘｶﾞ 変換時間268ｽﾃｰﾄ      */
        ATUC.TSTR1 = 0x01;              /* TCNT0ｶｳﾝﾄｽﾀｰﾄ                    */
        INTC.IPRC = 0x00E0;             /* ATU01割り込みﾚﾍﾞﾙ設定            */
        INTC.IPRJ = 0x00F0;             /* A/D割り込みﾚﾍﾞﾙ設定              */
        set_imask(0x0);                 /* 割込みﾏｽｸﾚﾍﾞﾙ設定               */
        while(1);                       /* 無限ﾙｰﾌﾟ(割込み待ち)             */
}
/************************************************************************/
/*              インターバル割込みルーチン                              */
/************************************************************************/
#pragma interrupt(ITV1)
void ITV1(void){
        ATU0.TSR0 &= 0xFF1F;            /* ｲﾝﾀｰﾊﾞﾙ割込みﾌﾗｸﾞｸﾘｱ              */
        PA.PADR |= 0x0001;              /* A/D開始認識用ﾌﾗｸﾞｾｯﾄ             */
}
/************************************************************************/
/*              A/D変換終了割込みルーチン                               */
/************************************************************************/
#pragma interrupt(ADI1)
void ADI1(void){
        AD.ADCSR1 &= 0x7F;              /* A/D変換終了ﾌﾗｸﾞｸﾘｱ               */
        ad_data = AD.ADDR12;            /* A/D変換ﾃﾞｰﾀ保存                  */
        PA.PADR &= 0xFFFE;              /* A/D開始認識用ﾌﾗｸﾞﾘｾｯﾄ            */
}
```

2.23　1ｻｲｸﾙｽｷｬﾝﾓｰﾄﾞによるA/D変換	MCU	ＳＨ７０５５	使用機能	Ａ／Ｄ

仕様

（１）図１に示すように、1グループ（AN0～AN11）に入力される電圧をＳＨ７０５５のA/D変換器を使用し、測定します。
（２）測定結果はRAMに格納します。
（３）入力電圧は2～5Vの範囲です。

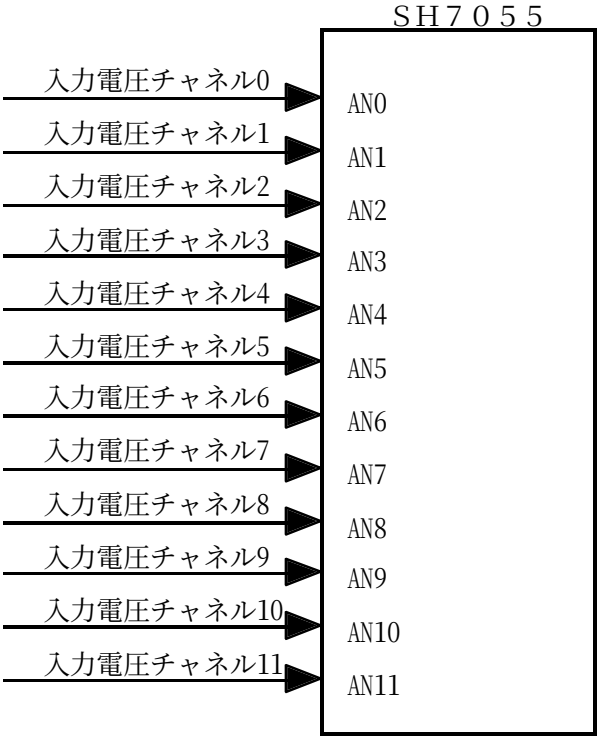


図1　ＳＨ７０５５による電圧の測定ブロック図

使用機能説明

表１に本タスク例の機能割り付けを示します。本タスク例は表１に示すようにＳＨ７０５５に内蔵されているA/D変換器の機能を割り付け、A/D変換を行ないます。

表1　Ａ／Ｄ機能割り付け

Ａ／Ｄ変換機能		機能
端子	AN0～AN11	アナログ電圧を入力します。
レジスタ	AD.ADCSR0	A/Dの変換モード及びA/D割り込み要求の許可／禁止、 測定端子を設定します。
	AD.ADCR0	A/D変換の開始及び動作クロック、スキャンモードの設定をします。
	AD.ADDR0～11	A/D変換結果を格納します。

2.23　1ｻｲｸﾙｽｷｬﾝﾓｰﾄﾞによるA/D変換	MCU	ＳＨ７０５５	使用機能	Ａ／Ｄ

動作原理

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理により12チャネルのA/D変換を行ないます。

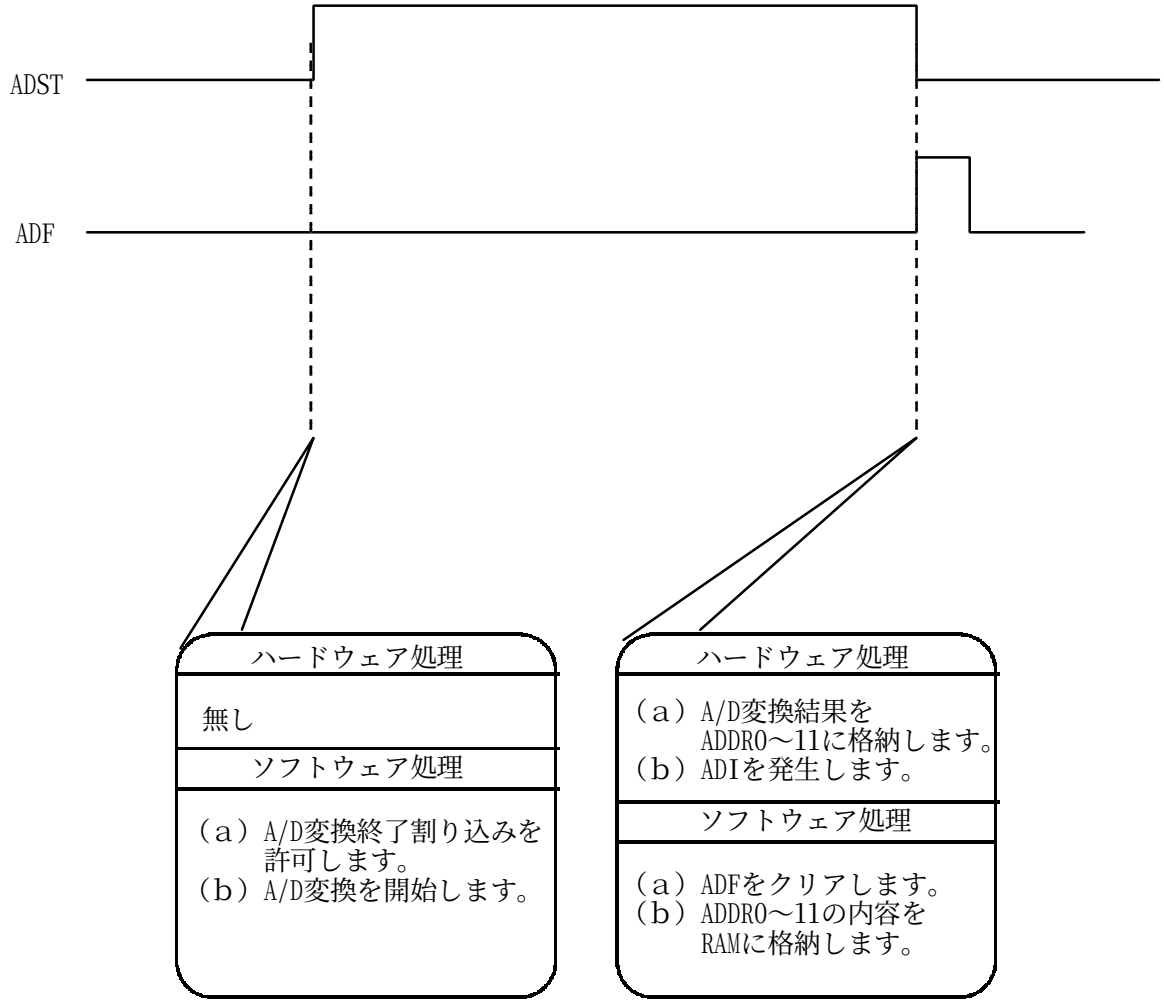


図２　Ａ／Ｄ変換動作原理

2.23　1ｻｲｸﾙｽｷｬﾝﾓｰﾄﾞによるA/D変換	MCU	SH7055	使用機能	A／D

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	A/Dの初期設定を行ないます。
A/D変換終了割り込みルーチン	ADI0	ADI0により起動し、A/D変換結果をRAMに格納します。

（2）使用変数の説明

ラベル名	機能	データ長	使用モジュール名
ADC.DATA0～11	AN0～11に入力した電圧のA/D変換データを格納します。	unsigned short	A/D変換終了割り込みルーチン
COUNT	A/D変換データの格納先頭アドレスからのオフセットを格納します。	char	

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
AD.ADCSR0	A/D変換モードをスキャンモードに設定、測定端子AN0～11によるA/D変換終了割り込み要求を許可します。	0x73	メインルーチン
AD.ADCR0	1サイクルスキャンモードによるA/D変換の開始を許可、変換時間を268ステートに設定します。	0x6F	
INTC.IPRJ	AD0の割り込み優先レベルを15に設定します。	0x0F00	
AD.ADDR0～11	A/D変換結果を格納します。	-	A/D変換終了割り込みルーチン

プログラムリスト

```
/************************************************************************/
/*              A/D0ｺﾝﾊﾞｰﾀ（1ｻｲｸﾙｽｷｬﾝﾓｰﾄﾞ）                                */
/************************************************************************/
#include <stdio.h>                              /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include <machine.h>
#include "7055.h"                               /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/************************************************************************/
/*              関数ﾌﾟﾛﾄｺﾙ宣言                                          */
/************************************************************************/
void main(void);
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
volatile struct st_adc{
        unsigned short DATA0;                           /*A/D変換ﾃﾞｰﾀ0      */
        unsigned short DATA1;                           /*A/D変換ﾃﾞｰﾀ1      */
        unsigned short DATA2;                           /*A/D変換ﾃﾞｰﾀ2      */
        unsigned short DATA3;                           /*A/D変換ﾃﾞｰﾀ3      */
        unsigned short DATA4;                           /*A/D変換ﾃﾞｰﾀ4      */
        unsigned short DATA5;                           /*A/D変換ﾃﾞｰﾀ5      */
        unsigned short DATA6;                           /*A/D変換ﾃﾞｰﾀ6      */
        unsigned short DATA7;                           /*A/D変換ﾃﾞｰﾀ7      */
        unsigned short DATA8;                           /*A/D変換ﾃﾞｰﾀ8      */
        unsigned short DATA9;                           /*A/D変換ﾃﾞｰﾀ9      */
        unsigned short DATA10;                          /*A/D変換ﾃﾞｰﾀ10     */
        unsigned short DATA11;                          /*A/D変換ﾃﾞｰﾀ11     */
};
#define ADC (*(struct st_adc *)0xFFFFC000)
/************************************************************************/
/*              ﾒｲﾝﾙｰﾁﾝ                                                */
/************************************************************************/
void main(void){
        AD.ADCSR0 = 0x73;               /* A/D割込み許可､ｽｷｬﾝﾓｰﾄﾞ測定端子   */
        AD.ADCR0 = 0x6F;                /* 測定時間268ｽﾃｰﾄ､A/D変換ｽﾀｰﾄ      */
        INTC.IPRJ = 0x0F00;             /* A/D割り込みﾚﾍﾞﾙ設定             */
        set_imask(0x0);                 /* 割込みﾏｽｸﾚﾍﾞﾙ設定               */
        while(1);                       /* 無限ﾙｰﾌﾟ(割込み待ち)             */
}
/************************************************************************/
/*              A/D変換終了割込みルーチン                               */
/************************************************************************/
#pragma interrupt(ADI0)
void ADI0(void){
        char COUNT;
        AD.ADCSR0 &= 0x7F;              /* A/D変換終了ﾌﾗｸﾞｸﾘｱ               */
        for(COUNT = 0; COUNT < 12; COUNT++){
                *(short*)(&ADC.DATA0 + COUNT) = *(short*)(&AD.ADDR0 + COUNT);
        }
}
```

2.24 連続サイクルスキャンモードによるA/D変換	MCU	ＳＨ７０５５	使用機能	Ａ／Ｄ

仕様

（１）図１に示すように、1グループ（AN0～AN11）に入力される電圧をＳＨ７０５５のA/D変換器を使用し、測定します。
（２）測定結果はRAMに格納します。
（３）入力電圧は2～5Vの範囲です。

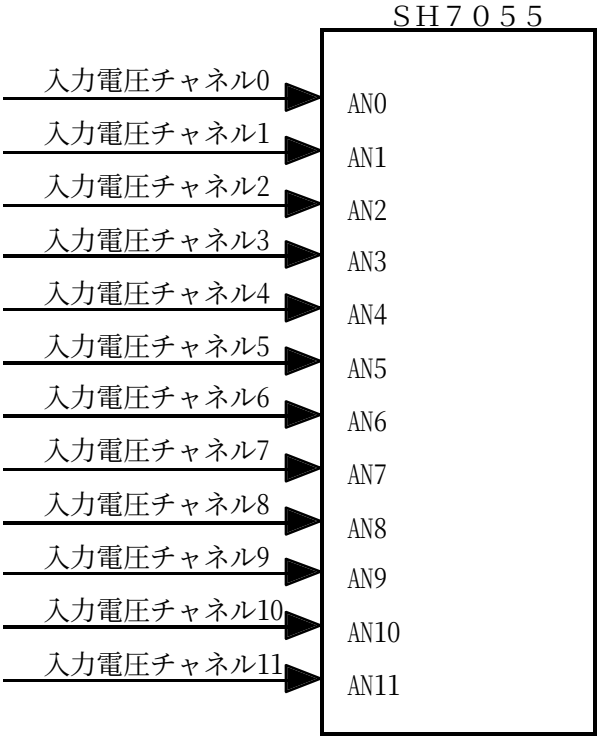


図1　ＳＨ７０５５による電圧の測定ブロック図

使用機能説明

表１に本タスク例の機能割り付けを示します。本タスク例は表１に示すようにＳＨ７０５５に内蔵されているA/D変換器の機能を割り付け、A/D変換を行ないます。

表1　Ａ／Ｄ機能割り付け

Ａ／Ｄ変換機能		機能
端子	AN0～AN11	アナログ電圧を入力します。
レジスタ	AD.ADCSR0	A/Dの変換モード及びA/D割り込み要求の許可／禁止、測定端子を設定します。
	AD.ADCR0	A/D変換の開始及び動作クロック、スキャンモードの設定をします。
	AD.ADDR0～11	A/D変換結果を格納します。

動作原理

図２に動作原理を示します。図２に示すようにＳＨ７０５５のハードウェア処理及びソフトウェアの処理により12チャネルのA/D変換を行ないます。

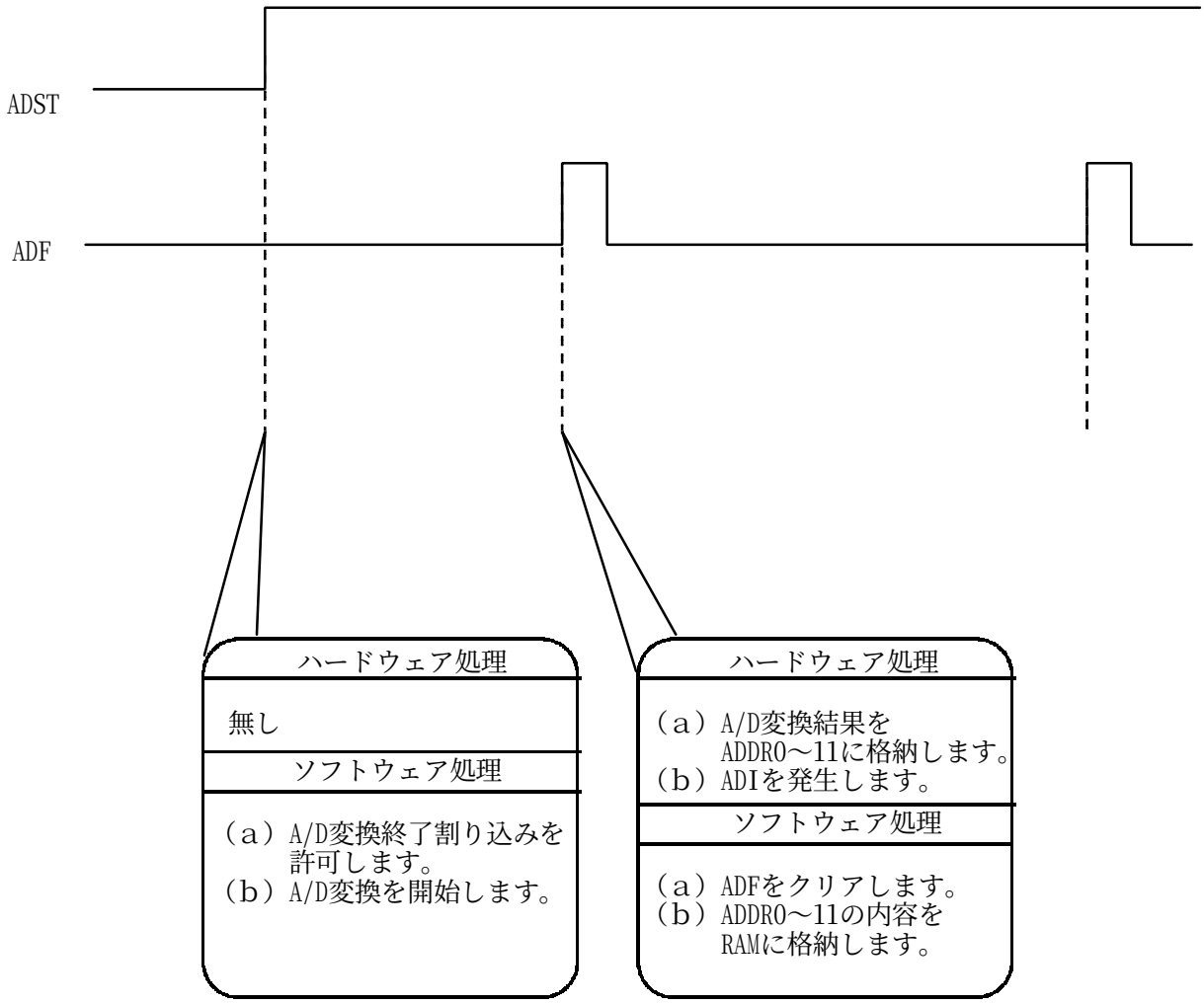


図２　Ａ／Ｄ変換動作原理

2.24 連続サイクルスキャンモードによるA/D変換	MCU	SH7055	使用機能	A／D

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	A/Dの初期設定を行ないます。
A/D変換終了割り込みルーチン	ADI0	ADI0により起動し、A/D変換結果をRAMに格納します。

（2）使用変数の説明

ラベル名	機能	データ長	使用モジュール名
ADC.DATA0～11	AN0～11に入力した電圧のA/D変換データを格納します。	unsigned short	A/D変換終了割り込みルーチン
COUNT	A/D変換データの格納先頭アドレスからのオフセットを格納します。	char	

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
AD.ADCSR0	A/D変換モードをスキャンモードに設定、測定端子AN0～11によるA/D変換終了割り込み要求を許可します。	0x73	メインルーチン
AD.ADCR0	連続サイクルスキャンモードによるA/D変換の開始を許可、変換時間を268ステートに設定します。	0x7F	
INTC.IPRJ	ADI0の割り込み優先レベルを15に設定します。	0x0F00	
AD.ADDR0～11	A/D変換結果を格納します。	-	A/D変換終了割り込みルーチン

プログラムリスト

```
/************************************************************************/
/*              A/D0コンバータ（連続スキャンモード）                    */
/************************************************************************/
#include <stdio.h>                          /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ      */
#include <machine.h>
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ     */
/************************************************************************/
/*              関数プロトコル宣言                                      */
/************************************************************************/
void main(void);
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
volatile struct st_adc{
        unsigned short DATA0;               /*A/D変換ﾃﾞｰﾀ0              */
        unsigned short DATA1;               /*A/D変換ﾃﾞｰﾀ1              */
        unsigned short DATA2;               /*A/D変換ﾃﾞｰﾀ2              */
        unsigned short DATA3;               /*A/D変換ﾃﾞｰﾀ3              */
        unsigned short DATA4;               /*A/D変換ﾃﾞｰﾀ4              */
        unsigned short DATA5;               /*A/D変換ﾃﾞｰﾀ5              */
        unsigned short DATA6;               /*A/D変換ﾃﾞｰﾀ6              */
        unsigned short DATA7;               /*A/D変換ﾃﾞｰﾀ7              */
        unsigned short DATA8;               /*A/D変換ﾃﾞｰﾀ8              */
        unsigned short DATA9;               /*A/D変換ﾃﾞｰﾀ9              */
        unsigned short DATA10;              /*A/D変換ﾃﾞｰﾀ10             */
        unsigned short DATA11;              /*A/D変換ﾃﾞｰﾀ11             */
};
#define ADC (*(struct st_adc *)0xFFFFC000)
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main(void){
        AD.ADCSR0 = 0x73;       /*A/D割込み許可､ｽｷｬﾝﾓｰﾄﾞ測定端子          */
        AD.ADCR0 = 0x7F;        /*測定時間268ｽﾃｰﾄ､A/D変換ｽﾀｰﾄ､連続ｽｷｬﾝ    */
        INTC.IPRJ = 0x0F00;     /* A/D割り込みﾚﾍﾞﾙ設定                  */
        set_imask(0x0);         /* 割込みﾏｽｸﾚﾍﾞﾙ設定                    */
        while(1);               /* 無限ﾙｰﾌﾟ(割込み待ち)                  */
}
/************************************************************************/
/*              A/D変換終了割込みルーチン                               */
/************************************************************************/
#pragma interrupt(ADI0)
void ADI0(void){
        char COUNT;
        AD.ADCSR0 &= 0x7F;                  /* A/D変換終了ﾌﾗｸﾞｸﾘｱ        */
                for(COUNT = 0; COUNT < 12; COUNT++)
                {
                     *(short*)(&ADC.DATA0 + COUNT) = *(short*)(&AD.ADDR0 + COUNT);
                }
}
```

2.25　外部トリガによるA／D変換	MCU	ＳＨ７０５５	使用機能	A／D

仕様

（1）図1に示すように、1チャネルに入力される電圧をＳＨ７０５５のA/D変換器を使用し、測定します。
（2）測定結果は2バイトのRAMに格納します。
（3）入力する電圧は0～5Vの範囲です。

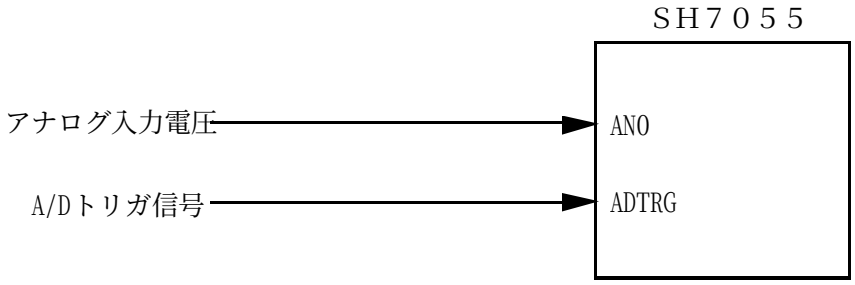


図1　ＳＨ７０５５による電圧の測定ブロック図

使用機能説明

表1、表2に本タスク例の機能割付を示します。表1、表2に示すようにＳＨ７０５５に内蔵されているA/D変換器、PFCの機能を割り付け、A/D変換を行ないます。

表1　A／D変換機能割付

A／D変換機能		機能
端子	AN0	アナログ電圧を入力する。
	ADTRG0	A/Dトリガ信号を入力する。
レジスタ	ADCSR0	A/D変換のモード（単一モード／スキャンモード）の設定、測定端子、クロックの選択をする。
	ADCR0	A/D変換器の測定の開始及び終了を設定する。
	ADDR0	A/D変換の結果を設定する。

表2　　ＰＦＣ機能割り付け

ＰＦＣレジスタ	機能
PGIOR	端子の入出力方向を設定する。
PGCR	端子の機能を選択する。

動作説明

図２に動作原理を示します。図２に示すように、ＳＨ７０５５のハードウェア処理及びソフトウェア処理により1チャネルのA/D変換を行います。

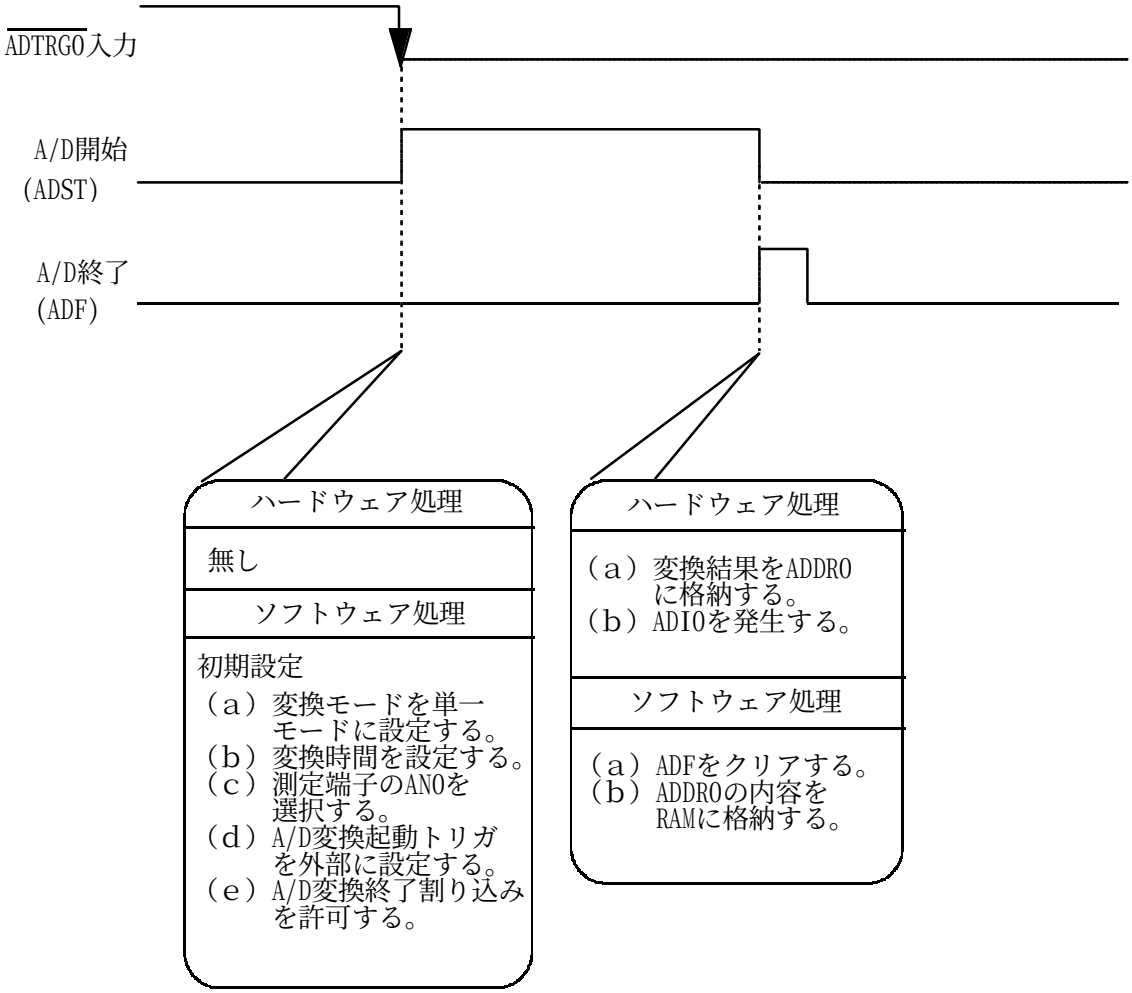


図２　A／D変換動作原理

2.25　外部トリガによるA／D変換	MCU	SH7055	使用機能	A／D

ソフトウェア説明

（1）モジュール説明

モジュール名	ラベル名	機　　能
メインルーチン	main	A/D変換器の初期設定を行なう。
A/D変換終了	adend	ADI0により起動し、結果をRAMに格納する。

（2）引数の説明

ラベル名	機　　能	データ長	使用モジュール名
ad_dat	A/D変換データを格納する。	unsigned short	A/D変換終了

（3）使用内部レジスタ説明

レジスタ名	機　　能	設定値	使用モジュール名
AD.ADCSR0	A/D変換モードを単一モードに設定、 測定端子をAN0に設定する。	0x40	メインルーチン
AD.ADCR0	A/D変換器のクロックを268ステートに設定する。	0xCF	
INTC.IPRJ	A/D0の割り込み優先レベルを設定する。	0x0F00	
PG.PGIOR	PG3を入力に設定する。	0x0000	
PG.PGCR	端子マルチプレクスをA/D変換トリガ入力に設定する。	0x0004	
AD.ADDR0	A/D変換の結果を設定する。		A/D変換終了

（4）使用ＲＡＭ

本タスクでは引数以外のＲＡＭは使用していません。

プログラムリスト

```
/************************************************************************/
/*              外部トリガによるA/D変換                                 */
/************************************************************************/
#include <machine.h>                        /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ          */
#include "7055.h"                           /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ        */
/************************************************************************/
/*              関数プロトタイプ宣言                                    */
/************************************************************************/
void main( void );
/************************************************************************/
/*              変数定義                                                */
/************************************************************************/
#define  ad_dat  (*(unsigned short *)0xFFFFC000)   /* A/D変換ﾃﾞｰﾀ格納       */
/************************************************************************/
/*              メインルーチン                                          */
/************************************************************************/
void main( void )
{
    PG.PGIOR = 0x0000;                  /* PG3を入力端子に設定             */
    PG.PGCR = 0x0080;                   /* A/Dﾄﾘｶﾞ入力端子に設定            */
    AD.ADCSR0 = 0x40;                   /* A/D割込許可,単一ﾓｰﾄﾞ,測定端子:AN0  */
    AD.ADCR0 = 0xCF;                    /* 外部ﾄﾘｶﾞ,変換時間268ｽﾃｰﾄ        */
    INTC.IPRJ = 0x0F00;                 /* A/D0割り込み優先ﾚﾍﾞﾙを15に設定    */
    set_imask(0x0);                     /* 割り込み許可                    */
    while(1);                           /* 無限ﾙｰﾌﾟ（割込み待ち）          */
}
/************************************************************************/
/*              A/D変換終了割込みルーチン                               */
/************************************************************************/
#pragma interrupt( adend)
void  adend( void )
{
    AD.ADCSR0 &= 0x7F;                  /* A/Dｴﾝﾄﾞﾌﾗｸﾞｸﾘｱ                  */
    ad_dat = AD.ADDR0;                  /* A/Dﾃﾞｰﾀの格納                    */
}
```

2.26　端子の出力遮断	MCU	SH7055	使用機能	I / O ポート

仕様

（1）図１に示すようにPOD端子にLOWレベルを入力した時、PE0端子の出力を遮断します。

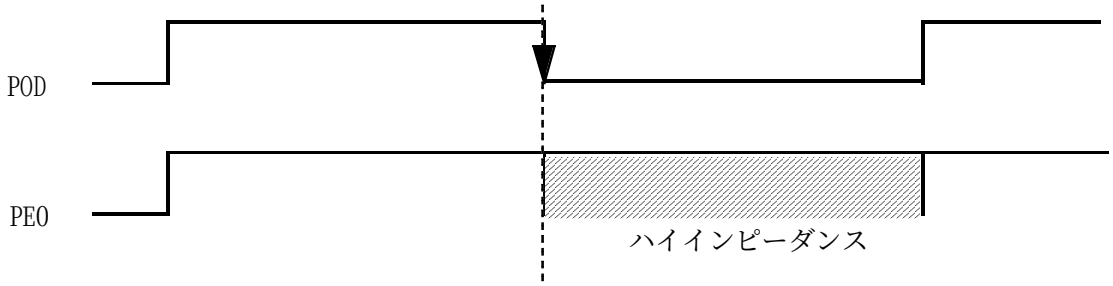


図1　端子の出力遮断動作ブロック図

使用機能説明

表1に本タスク例の機能割り付けを示します。表1に示すようにSH7055のI/0ポートの機能を割り付け、端子の出力遮断を行ないます。

表1　I / 0 ポート機能割り付け

I / 0 ポート機能	機能
PE.PEIOR	端子の入出力を設定します。
PF.PFIOR	
PE.PECR	端子の機能を選択します。
PE.PECRH	
PE.PECRL	
PE.PEDR	ポートのデータを格納します。

動作説明

図2に動作原理を示します。図2に示すように、SH7055のハードウェア処理及びソフトウェア処理により端子の出力遮断を行ないます。

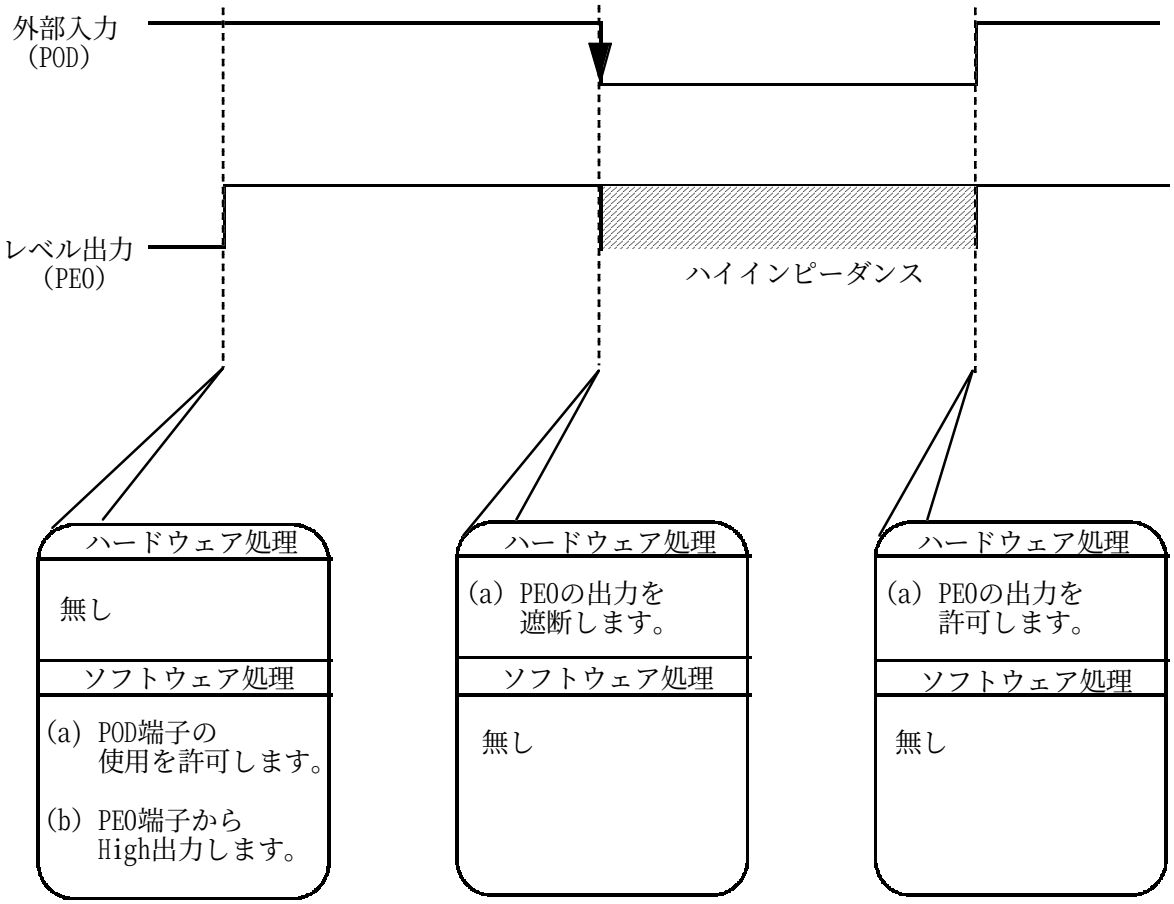


図２　端子の出力遮断動作原理

2.26　端子の出力遮断	MCU	SH7055	使用機能	I / 0 ポート

ソフトウェア説明

（1） モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	I/Oポートの初期設定を行います。

（2）使用変数の説明

本タスクでは変数は使用してません。

（3）使用内部レジスタ説明

レジスタ名	機能	設定値	使用モジュール名
PE.PEIOR	PE0を出力端子に設定します。	0x0001	メインルーチン
PF.PFIOR	PB0を入力端子に設定します。	0x0000	
PE.PECR	端子マルチプレクスをポートに設定します。	0x0000	
PE.PECRH	端子マルチプレクスをポートに設定します。	0x0000	
PE.PECRL	端子マルチプレクスをPODに設定します。	0x0800	
PE.PEDR	Highレベルを出力します。	0x0001	

プログラムリスト

```
/************************************************************************/
/*              端子の出力遮断                                          */
/************************************************************************/
#include <stdio.h>                      /* ﾗｲﾌﾞﾗﾘ関数用ﾍｯﾀﾞﾌｧｲﾙ        */
#include <machine.h>
#include "7055.h"                       /* 周辺ﾚｼﾞｽﾀ定義ﾍｯﾀﾞﾌｧｲﾙ       */
/************************************************************************/
/*              関数プロトタイプ宣言                                    */
/************************************************************************/
void main(void);
/************************************************************************/
/*              端子の出力遮断                                          */
/************************************************************************/

void main(void)
{
    PE.PEIOR = 0x0001;          /* PE0を出力端子に設定する              */
    PE.PECR  = 0x0000;          /* 端子ﾏﾙﾁﾌﾟﾚｸｽをPE0に設定する         */
    PF.PFIOR = 0x0000;          /* PODを入力端子にする                  */
    PF.PFCRH = 0x0000;          /* 端子ﾏﾙﾁﾌﾟﾚｸｽを汎用出力に設定する    */
    PF.PFCRL = 0x0800;          /* 端子ﾏﾙﾁﾌﾟﾚｸｽをPODに設定する         */
    PE.PEDR  = 0x0001;          /* Highﾚﾍﾞﾙ出力                         */
    sleep();
}
```

```
/************************************************************************/
/*   SH7055 Include File (by SH7055 Target Spec. Rev.0.1) Ver 0.0       */
/************************************************************************/
/*----------------------------------------------------------------------*/
/*            SCI                                                       */
/*----------------------------------------------------------------------*/
volatile struct st_sci              /* struct SCI0～4                    */
{
       unsigned char SMR;           /* Serial Mode Register              */
       unsigned char BRR;           /* Bit Rate Register                 */
       unsigned char SCR;           /* Serial Control Register           */
       unsigned char TDR;           /* Transmit Data Register            */
       unsigned char SSR;           /* Serial Status Register            */
       unsigned char RDR;           /* Receive Data Register             */
       unsigned char SDCR;          /* Serial Direction Control Register*/
};
#define SCI0 (*(volatile struct st_sci *)0xFFFFF000)/* SCI0 Address     */
#define SCI1 (*(volatile struct st_sci *)0xFFFFF008)/* SCI1 Address     */
#define SCI2 (*(volatile struct st_sci *)0xFFFFF010)/* SCI2 Address     */
#define SCI3 (*(volatile struct st_sci *)0xFFFFF018)/* SCI3 Address     */
#define SCI4 (*(volatile struct st_sci *)0xFFFFF020)/* SCI4 Address     */
/************************************************************************/
/*            Advanced Timer Unit-Ⅱ                                     */
/*----------------------------------------------------------------------*/
/*                  ATU(common)                                         */
/*----------------------------------------------------------------------*/
volatile struct st_atuc             /* struct ATUC                      */
{
       unsigned char TSTR2;         /* Timer Start Register 2           */
       unsigned char TSTR1;         /* Timer Start Register 1           */
       unsigned char TSTR3;         /* Timer Start Register 3           */
       unsigned char DYF403;        /* Dummy Byte                       */
       unsigned char PSCR1;         /* Prescaler Register 1             */
       unsigned char DYF405;        /* Dummy Byte                       */
       unsigned char PSCR2;         /* Prescaler Register 2             */
       unsigned char DYF407;        /* Dummy Byte                       */
       unsigned char PSCR3;         /* Prescaler Register 3             */
       unsigned char DYF409;        /* Dummy Byte                       */
       unsigned char PSCR4;         /* Prescaler Register 4             */
};
#define ATUC (*(volatile struct st_atuc *)0xFFFFF400)/* ATUC Address    */
#define TSTR12              (*(volatile unsigned short *)0xFFFFF400)
#define TSTR123             (*(volatile unsigned long  *)0xFFFFF400)
```

```
/*------------------------------------------------------------------------*/
/*                      ATU0                                              */
/*------------------------------------------------------------------------*/
volatile struct st_atu0             /* struct ATU0                        */
{
       unsigned long  ICR0D;        /* Input Capture Register D           */
       unsigned char  ITVRR1;/* Timer Interval Interrupt Request Reg. 1 */
       unsigned char  DYF425;       /* Dummy Byte                         */
       unsigned char  ITVRR2A;/* Timer Interval Interrupt Request Reg. 2A*/
       unsigned char  DYF427;       /* Dummy Byte                         */
       unsigned char  ITVRR2B;/* Timer Interval Interrupt Request Reg. 2B*/
       unsigned char  DYF429;       /* Dummy Byte                         */
       unsigned char  TIOR0;        /* Timer I/O Control Register         */
       unsigned char  DYF42B;       /* Dummy Byte                         */
       unsigned short TSR0;         /* Timer Status Register 0            */
       unsigned short TIER0;        /* Timer Interrupt Enable Register 0*/
       unsigned long  TCNT0H;       /* Free Running Counter 0             */
       unsigned long  ICR0AH;       /* Input Capture Register A           */
       unsigned long  ICR0BH;       /* Input Capture Register B           */
       unsigned long  ICR0CH;       /* Input Capture Register C           */
};
#define ATU0 (*(volatile struct st_atu0 *)0xFFFFF420)/* ATU0 Address    */
/*------------------------------------------------------------------------*/
/*                      ATU1                                              */
/*------------------------------------------------------------------------*/
volatile struct st_atu1             /* struct ATU1                        */
{
       unsigned short TCNT1A;       /* Free Running Counter 1A            */
       unsigned short TCNT1B;       /* Free Running Counter 1B            */
       unsigned short GR1A;         /* General Register 1A                */
       unsigned short GR1B;         /* General Register 1B                */
       unsigned short GR1C;         /* General Register 1C                */
       unsigned short GR1D;         /* General Register 1D                */
       unsigned short GR1E;         /* General Register 1E                */
       unsigned short GR1F;         /* General Register 1F                */
       unsigned short GR1G;         /* General Register 1G                */
       unsigned short GR1H;         /* General Register 1H                */
       unsigned short OCR1;         /* Output Compare Register 1          */
       unsigned short OSBR1;        /* Offset Base Register 1             */
       unsigned char  TIOR1B;       /* Timer I/O Control Register 1B      */
       unsigned char  TIOR1A;       /* Timer I/O Control Register 1A      */
       unsigned char  TIOR1D;       /* Timer I/O Control Register 1D      */
       unsigned char  TIOR1C;       /* Timer I/O Control Register 1C      */
       unsigned char  TCR1B;        /* Timer Control Register 1B          */
       unsigned char  TCR1A;        /* Timer Control Register 1A          */
       unsigned short TSR1A;        /* Timer Status Register 1A           */
       unsigned short TSR1B;        /* Timer Status Register 1B           */
       unsigned short TIER1A;     /* Timer Interrupt Enable Register 1A */
       unsigned short TIER1B;     /* Timer Interrupt Enable Register 1B */
       unsigned char  TRGMDR1;      /* Trigger Mode Register 1            */
};
#define ATU1 (*(volatile struct st_atu1 *)0xFFFFF440)    /* ATU1 Address*/
#define TIOR1AB             (*(volatile unsigned short *)0xFFFFF458)
#define TIOR1CD             (*(volatile unsigned short *)0xFFFFF45A)
#define TCR1AB              (*(volatile unsigned short *)0xFFFFF45C)
```

```
/*-----------------------------------------------------------------------*/
/*                    ATU2                                               */
/*-----------------------------------------------------------------------*/
volatile struct st_atu2             /* struct ATU2                       */
{
       unsigned short TCNT2A;       /* Free Running Counter 2A           */
       unsigned short TCNT2B;       /* Free Running Counter 2B           */
       unsigned short GR2A;         /* General Register 2A               */
       unsigned short GR2B;         /* General Register 2B               */
       unsigned short GR2C;         /* General Register 2C               */
       unsigned short GR2D;         /* General Register 2D               */
       unsigned short GR2E;         /* General Register 2E               */
       unsigned short GR2F;         /* General Register 2F               */
       unsigned short GR2G;         /* General Register 2G               */
       unsigned short GR2H;         /* General Register 2H               */
       unsigned short OCR2A;        /* Output Compare Register 2A        */
       unsigned short OCR2B;        /* Output Compare Register 2B        */
       unsigned short OCR2C;        /* Output Compare Register 2C        */
       unsigned short OCR2D;        /* Output Compare Register 2D        */
       unsigned short OCR2E;        /* Output Compare Register 2E        */
       unsigned short OCR2F;        /* Output Compare Register 2F        */
       unsigned short OCR2G;        /* Output Compare Register 2G        */
       unsigned short OCR2H;        /* Output Compare Register 2H        */
       unsigned short OSBR2;        /* Offset Base Register 2            */
       unsigned char  TIOR2B;       /* Timer I/O Control Register 2B     */
       unsigned char  TIOR2A;       /* Timer I/O Control Register 2A     */
       unsigned char  TIOR2D;       /* Timer I/O Control Register 2D     */
       unsigned char  TIOR2C;       /* Timer I/O Control Register 2C     */
       unsigned char  TCR2B;        /* Timer Control Register 2B         */
       unsigned char  TCR2A;        /* Timer Control Register 2A         */
       unsigned short TSR2A;        /* Timer Status Register 2A          */
       unsigned short TSR2B;        /* Timer Status Register 2B          */
       unsigned short TIER2A;     /* Timer Interrupt Enable Register 2A */
       unsigned short TIER2B;     /* Timer Interrupt Enable Register 2B */
};
#define ATU2 (*(volatile struct st_atu2 *)0xFFFFF600)/* ATU2 Address     */
#define TIOR2AB             (*(volatile unsigned short *)0xFFFFF626)
#define TIOR2CD             (*(volatile unsigned short *)0xFFFFF628)
#define TCR2AB              (*(volatile unsigned short *)0xFFFFF62A)
/*-----------------------------------------------------------------------*/
/*                    ATU3                                               */
/*-----------------------------------------------------------------------*/
volatile struct st_atu3             /* struct ATU3                       */
{
       unsigned short TSR3;         /* Timer Status Register 3           */
       unsigned short TIER3;        /* Timer Interrupt Enable Register 3*/
       unsigned char  TMDR;         /* Timer Mode Register               */
       unsigned char  DYF485;       /* Dummy Byte                        */
       unsigned short DYF486;       /* Dummy Word                        */
       unsigned long  DYF488;       /* Dummy Long                        */
       unsigned long  DYF48C;       /* Dummy Long                        */
       unsigned long  DYF490;       /* Dummy Long                        */
       unsigned long  DYF494;       /* Dummy Long                        */
       unsigned long  DYF498;       /* Dummy Long                        */
       unsigned long  DYF49C;       /* Dummy Long                        */
       unsigned short TCNT3;        /* Free Running Counter 3            */
       unsigned short GR3A;         /* General Register 3A               */
       unsigned short GR3B;         /* General Register 3B               */
       unsigned short GR3C;         /* General Register 3C               */
       unsigned short GR3D;         /* General Register 3D               */
       unsigned char  TIOR3B;       /* Timer I/O Control Register 3B     */
       unsigned char  TIOR3A;       /* Timer I/O Control Register 3A     */
       unsigned char  TCR3;         /* Timer Control Register 3          */
};
#define ATU3 (*(volatile struct st_atu3 *)0xFFFFF480)    /* ATU3 Address*/
#define TIOR3AB             (*(volatile unsigned short *)0xFFFFF4AA)
```

```
/*-----------------------------------------------------------------------*/
/*                    ATU4                                               */
/*-----------------------------------------------------------------------*/
volatile struct st_atu4             /* struct ATU4                       */
{
       unsigned short TCNT4;        /* Free Running Counter 4            */
       unsigned short GR4A;         /* General Register 4A               */
       unsigned short GR4B;         /* General Register 4B               */
       unsigned short GR4C;         /* General Register 4C               */
       unsigned short GR4D;         /* General Register 4D               */
       unsigned char TIOR4B;        /* Timer I/O Control Register 4B     */
       unsigned char TIOR4A;        /* Timer I/O Control Register 4A     */
       unsigned char TCR4;          /* Timer Control Register 4          */
};
#define ATU4 (*(volatile struct st_atu4 *)0xFFFFF4C0)    /* ATU4 Address*/
#define TIOR4AB             (*(volatile unsigned short *)0xFFFFF4CA)
/*-----------------------------------------------------------------------*/
/*                    ATU5                                               */
/*-----------------------------------------------------------------------*/
volatile struct st_atu5             /* struct ATU5                       */
{
       unsigned short TCNT5;        /* Free Running Counter 5            */
       unsigned short GR5A;         /* General Register 5A               */
       unsigned short GR5B;         /* General Register 5B               */
       unsigned short GR5C;         /* General Register 5C               */
       unsigned short GR5D;         /* General Register 5D               */
       unsigned char  TIOR5B;       /* Timer I/O Control Register 5B     */
       unsigned char  TIOR5A;       /* Timer I/O Control Register 5A     */
       unsigned char  TCR5;         /* Timer Control Register 5          */
};
#define ATU5 (*(volatile struct st_atu5 *)0xFFFFF4E0)    /* ATU5 Address*/
#define TIOR5AB             (*(volatile unsigned short *)0xFFFFF4EA)
/*-----------------------------------------------------------------------*/
/*                    ATU6                                               */
/*-----------------------------------------------------------------------*/
volatile struct st_atu6             /* struct ATU6                       */
{
       unsigned short TCNT6A;       /* Free Running Counter 6A           */
       unsigned short TCNT6B;       /* Free Running Counter 6B           */
       unsigned short TCNT6C;       /* Free Running Counter 6C           */
       unsigned short TCNT6D;       /* Free Running Counter 6D           */
       unsigned short CYLR6A;       /* Cycle Register 6A                 */
       unsigned short CYLR6B;       /* Cycle Register 6B                 */
       unsigned short CYLR6C;       /* Cycle Register 6C                 */
       unsigned short CYLR6D;       /* Cycle Register 6D                 */
       unsigned short BFR6A;        /* Buffer Register 6A                */
       unsigned short BFR6B;        /* Buffer Register 6B                */
       unsigned short BFR6C;        /* Buffer Register 6C                */
       unsigned short BFR6D;        /* Buffer Register 6D                */
       unsigned short DTR6A;        /* Duty Register 6A                  */
       unsigned short DTR6B;        /* Duty Register 6B                  */
       unsigned short DTR6C;        /* Duty Register 6C                  */
       unsigned short DTR6D;        /* Duty Register 6D                  */
       unsigned char  TCR6B;        /* Timer Control Register 6B         */
       unsigned char  TCR6A;        /* Timer Control Register 6A         */
       unsigned short TSR6;         /* Timer Status Register 6           */
       unsigned short TIER6;        /* Timer Interrupt Enable Register 6*/
       unsigned char  PMDR6;        /* PWM Mode Register 6               */
};
#define ATU6 (*(volatile struct st_atu6 *)0xFFFFF500)/* ATU6 Address    */
#define TCR6AB              (*(volatile unsigned short *)0xFFFFF520)
```

```
/*-----------------------------------------------------------------------*/
/*                      ATU7                                             */
/*-----------------------------------------------------------------------*/
volatile struct st_atu7             /* struct ATU7                       */
{
       unsigned short TCNT7A;       /* Free Running Counter 7A           */
       unsigned short TCNT7B;       /* Free Running Counter 7B           */
       unsigned short TCNT7C;       /* Free Running Counter 7C           */
       unsigned short TCNT7D;       /* Free Running Counter 7D           */
       unsigned short CYLR7A;       /* Cycle Register 7A                 */
       unsigned short CYLR7B;       /* Cycle Register 7B                 */
       unsigned short CYLR7C;       /* Cycle Register 7C                 */
       unsigned short CYLR7D;       /* Cycle Register 7D                 */
       unsigned short BFR7A;        /* Buffer Register 7A                */
       unsigned short BFR7B;        /* Buffer Register 7B                */
       unsigned short BFR7C;        /* Buffer Register 7C                */
       unsigned short BFR7D;        /* Buffer Register 7D                */
       unsigned short DTR7A;        /* Duty Register 7A                  */
       unsigned short DTR7B;        /* Duty Register 7B                  */
       unsigned short DTR7C;        /* Duty Register 7C                  */
       unsigned short DTR7D;        /* Duty Register 7D                  */
       unsigned char  TCR7B;        /* Timer Control Register 7B         */
       unsigned char  TCR7A;        /* Timer Control Register 7A         */
       unsigned short TSR7;         /* Timer Status Register 7           */
       unsigned short TIER7;        /* Timer Interrupt Enable Register 7*/
};
#define ATU7 (*(volatile struct st_atu7 *)0xFFFFF580)    /* ATU7 Address*/
#define TCR7AB              (*(volatile unsigned short *)0xFFFFF5A0)
/*-----------------------------------------------------------------------*/
/*                      ATU8                                             */
/*-----------------------------------------------------------------------*/
volatile struct st_atu8             /* struct ATU8                       */
{
       unsigned short DCNT8A;       /* Down Counter 8A                   */
       unsigned short DCNT8B;       /* Down Counter 8B                   */
       unsigned short DCNT8C;       /* Down Counter 8C                   */
       unsigned short DCNT8D;       /* Down Counter 8D                   */
       unsigned short DCNT8E;       /* Down Counter 8E                   */
       unsigned short DCNT8F;       /* Down Counter 8F                   */
       unsigned short DCNT8G;       /* Down Counter 8G                   */
       unsigned short DCNT8H;       /* Down Counter 8H                   */
       unsigned short DCNT8I;       /* Down Counter 8I                   */
       unsigned short DCNT8J;       /* Down Counter 8J                   */
       unsigned short DCNT8K;       /* Down Counter 8K                   */
       unsigned short DCNT8L;       /* Down Counter 8L                   */
       unsigned short DCNT8M;       /* Down Counter 8M                   */
       unsigned short DCNT8N;       /* Down Counter 8N                   */
       unsigned short DCNT8O;       /* Down Counter 8O                   */
       unsigned short DCNT8P;       /* Down Counter 8P                   */
       unsigned short RLDR8;        /* Reload Buffer 8                   */
       unsigned short TCNR;         /* Timer Connection Register         */
       unsigned short OTR;          /* OSP Termination Register          */
       unsigned short DSTR;         /* Down Count Start Register         */
       unsigned char  TCR8;         /* Timer Control Register 8          */
       unsigned char  DYF669;       /* Dummy Byte                        */
       unsigned short TSR8;         /* Timer Status Register 8           */
       unsigned short TIER8;        /* Timer Interrupt Enable Register 8*/
       unsigned char  RLDENR;       /* Reload Enable Register 8          */
};
#define ATU8 (*(volatile struct st_atu8 *)0xFFFFF640)    /* ATU8 Address*/
```

```
/*------------------------------------------------------------------------*/
/*                      ATU9                                              */
/*------------------------------------------------------------------------*/
volatile struct st_atu9             /* struct ATU9                        */
{
       unsigned char ECNT9A;        /* Event Counter 9A                   */
       unsigned char DYF681;        /* Dummy Byte                         */
       unsigned char ECNT9B;        /* Event Counter 9B                   */
       unsigned char DYF683;        /* Dummy Byte                         */
       unsigned char ECNT9C;        /* Event Counter 9C                   */
       unsigned char DYF685;        /* Dummy Byte                         */
       unsigned char ECNT9D;        /* Event Counter 9D                   */
       unsigned char DYF687;        /* Dummy Byte                         */
       unsigned char ECNT9E;        /* Event Counter 9E                   */
       unsigned char DYF689;        /* Dummy Byte                         */
       unsigned char ECNT9F;        /* Event Counter 9F                   */
       unsigned char DYF68B;        /* Dummy Byte                         */
       unsigned char GR9A;          /* General Register 9A                */
       unsigned char DYF68D;        /* Dummy Byte                         */
       unsigned char GR9B;          /* General Register 9B                */
       unsigned char DYF68F;        /* Dummy Byte                         */
       unsigned char GR9C;          /* General Register 9C                */
       unsigned char DYF691;        /* Dummy Byte                         */
       unsigned char GR9D;          /* General Register 9D                */
       unsigned char DYF693;        /* Dummy Byte                         */
       unsigned char GR9E;          /* General Register 9E                */
       unsigned char DYF695;        /* Dummy Byte                         */
       unsigned char GR9F;          /* General Register 9F                */
       unsigned char DYF697;        /* Dummy Byte                         */
       unsigned char TCR9A;         /* Timer Control Register 9A          */
       unsigned char DYF699;        /* Dummy Byte                         */
       unsigned char TCR9B;         /* Timer Control Register 9B          */
       unsigned char DYF69B;        /* Dummy Byte                         */
       unsigned char TCR9C;         /* Timer Control Register 9C          */
       unsigned char DYF69D;        /* Dummy Byte                         */
       unsigned short TSR9;         /* Timer Status Register 9            */
       unsigned short TIER9;        /* Timer Interrupt Enable Register 9*/
};
#define ATU9 (*(volatile struct st_atu9 *)0xFFFFF680)/* ATU9 Address    */
```

```
/*-----------------------------------------------------------------------*/
/*                    ATU10                                              */
/*-----------------------------------------------------------------------*/
volatile struct st_atu10           /* struct ATU10                       */
{
       unsigned long  TCNT10A;     /* Free Running Counter 10A           */
       unsigned char  TCNT10B;     /* Event Counter 10B                  */
       unsigned char  DYF6C5;      /* Dummy Byte                         */
       unsigned short TCNT10C;     /* Reload Counter 10C                 */
       unsigned char  TCNT10D;     /* Angle Counter 10D                  */
       unsigned char  DYF6C9;      /* Dummy Byte                         */
       unsigned short TCNT10E;     /* Angle Counter 10E                  */
       unsigned short TCNT10F;     /* Angle Counter 10F                  */
       unsigned short TCNT10G;     /* Angle Counter 10G                  */
       unsigned long  ICR10A;      /* Input Capture Register 10A         */
       unsigned long  OCR10A;      /* Output Compare Register 10A        */
       unsigned char  OCR10B;      /* Output Compare Register 10B        */
       unsigned char  DYF6D9;      /* Dummy Byte                         */
       unsigned short RLD10C;      /* Reload Register 10C                */
       unsigned short GR10G;       /* General Register 10G               */
       unsigned char  TCNT10H;     /* Noise Canceler Counter 10H         */
       unsigned char  DYF6DF;      /* Dummy Byte                         */
       unsigned char  NCR10;       /* Noise Canceler Register 10         */
       unsigned char  DYF6E1;      /* Dummy Byte                         */
       unsigned char  TIOR10;      /* Timer I/O Control Register 10      */
       unsigned char  DYF6E3;      /* Dummy Byte                         */
       unsigned char  TCR10;       /* Timer Control Register 10          */
       unsigned char  DYF6E5;      /* Dummy Byte                         */
       unsigned short TCCLR10;     /* Timer Counter Clear Register 10    */
       unsigned short TSR10;       /* Timer Status Register 10           */
       unsigned short TIER10;     /* Timer Interrupt Enable Register 10 */
};
#define ATU10 (*(volatile struct st_atu10 *)0xFFFFF6C0)/* ATU10 Adress  */
/*-----------------------------------------------------------------------*/
/*                    ATU11                                              */
/*-----------------------------------------------------------------------*/
volatile struct st_atu11           /* struct ATU11                       */
{
       unsigned short TCNT11;      /* Free Running Counter 11            */
       unsigned short GR11A;       /* General Register 11A               */
       unsigned short GR11B;       /* General Register 11B               */
       unsigned char  TIOR11;      /* Timer I/O Control Register 11      */
       unsigned char  DYF5C7;      /* Dummy Byte                         */
       unsigned char  TCR11;       /* Timer Control Register 11          */
       unsigned char  DYF5C9;      /* Dummy Byte                         */
       unsigned short TSR11;       /* Timer Status Register 11           */
       unsigned short TIER11;     /* Timer Interrupt Enable Register 11 */
};
#define ATU11 (*(volatile struct st_atu11 *)0xFFFFF5C0)/* ATU11 Adress  */
```

```
/************************************************************************/
/*            Interrupt Controller                                      */
/************************************************************************/
volatile struct st_intc     /* struct INTC                              */
{                           /*      4bit    4bit      4bit     4bit     */
       unsigned short IPRA; /*IPRA: IRQ0    IRQ1      IRQ2     IRQ3     */
       unsigned short IPRB; /*IPRB: IRQ4    IRQ5      IRQ6     IRQ7     */
       unsigned short IPRC; /*IPRC: DMAC0,1 DMAC2,3   ATU01    ATU02    */
       unsigned short IPRD; /*IPRD: ATU03   ATU04     ATU11    ATU12    */
       unsigned short IPRE; /*IPRE: ATU13   ATU21     ATU22    ATU23    */
       unsigned short IPRF; /*IPRF: ATU31   ATU32     ATU41    ATU42    */
       unsigned short IPRG; /*IPRG: ATU51   ATU52     ATU6     ATU7     */
       unsigned short IPRH; /*IPRH: ATU81   ATU82     ATU83    ATU84    */
       unsigned short IPRI; /*IPRI: ATU91   ATU92     ATU101   ATU102   */
       unsigned short IPRJ; /*IPRJ: ATU11   CMT0,A/D0 CMT1,AD1 A/D2     */
       unsigned short IPRK; /*IPRK: SCI0    SCI1      SCI2     SCI3     */
       unsigned short IPRL; /*IPRL: SCI4    HCAN0     WDT      HCAN1    */
       unsigned short ICR;  /* Interrupt Control Register               */
       unsigned short ISR;  /* Interrupt Status Register                */
};
#define INTC (*(volatile struct st_intc *)0xFFFFED00)    /* INTC Address*/
/************************************************************************/
/*            I/O Port                                                  */
/*----------------------------------------------------------------------*/
/*                  Port A                                              */
/*----------------------------------------------------------------------*/
volatile struct st_pa               /* struct PA                        */
{
       unsigned short PAIOR;        /*  Port A I/O Register             */
       unsigned short PACRH;        /*  Port A Control Register  H      */
       unsigned short PACRL;        /*  Port A Control Register  L      */
       unsigned short PADR;         /*  Port A Data Register            */
};
#define PA   (*(volatile struct st_pa   *)0xFFFFF720)    /* PA Address */
/*----------------------------------------------------------------------*/
/*                  Port B                                              */
/*----------------------------------------------------------------------*/
volatile struct st_pb               /* struct PB                        */
{
       unsigned short PBIOR;        /*  Port B I/O Register             */
       unsigned short PBCRH;        /*  Port B Control Register  H      */
       unsigned short PBCRL;        /*  Port B Control Register  L      */
       unsigned short PBIR;         /*  Port B Invert Register          */
       unsigned short PBDR;         /*  Port B Data Register            */
};
#define PB   (*(volatile struct st_pb   *)0xFFFFF730)    /* PB Address */
/*----------------------------------------------------------------------*/
/*                  Port C                                              */
/*----------------------------------------------------------------------*/
volatile struct st_pc               /* struct PC                        */
{
       unsigned short PCIOR;        /*  Port C I/O Register             */
       unsigned short PCCR;         /*  Port C Control Register         */
       unsigned short PCDR;         /*  Port C Data Register            */
};
#define PC   (*(volatile struct st_pc   *)0xFFFFF73A)    /* PC Address */
```

```
/*-----------------------------------------------------------------------*/
/*                    Port D                                             */
/*-----------------------------------------------------------------------*/
volatile struct st_pd               /* struct PD                         */
{
       unsigned short PDIOR;        /*  Port D I/O Register              */
       unsigned short PDCRH;        /*  Port D Control Register H        */
       unsigned short PDCRL;        /*  Port D Control Register L        */
       unsigned short PDDR;         /*  Port D Data Register             */
};
#define PD   (*(volatile struct st_pd   *)0xFFFFF740)     /* PD Address */
/*-----------------------------------------------------------------------*/
/*                    Port E                                             */
/*-----------------------------------------------------------------------*/
volatile struct st_pe               /* struct PE                         */
{
       unsigned short PEIOR;        /*  Port E I/O Register              */
       unsigned short PECR;         /*  Port E Control Register          */
       unsigned short PEDR;         /*  Port E Data Register             */
};
#define PE   (*(volatile struct st_pe   *)0xFFFFF750)     /* PE Address */
/*-----------------------------------------------------------------------*/
/*                    Port F                                             */
/*-----------------------------------------------------------------------*/
volatile struct st_pf               /* struct PF                         */
{
       unsigned short PFIOR;        /*  Port F I/O Register              */
       unsigned short PFCRH;        /*  Port F Control Register H        */
       unsigned short PFCRL;        /*  Port F Control Register L        */
       unsigned short PFDR;         /*  Port F Data Register             */
};
#define PF   (*(volatile struct st_pf   *)0xFFFFF748)     /* PF Address */
/*-----------------------------------------------------------------------*/
/*                    Port G                                             */
/*-----------------------------------------------------------------------*/
volatile struct st_pg               /* struct PG                         */
{
       unsigned short PGIOR;        /*  Port G I/O Register              */
       unsigned short PGCR;         /*  Port G Control Register          */
       unsigned short PGDR;         /*  Port G Data Register             */
};
#define PG   (*(volatile struct st_pg   *)0xFFFFF760)     /* PG Address */
/*-----------------------------------------------------------------------*/
/*                    Port H                                             */
/*-----------------------------------------------------------------------*/
volatile struct st_ph               /* struct PH                         */
{
       unsigned short PHIOR;        /*  Port H I/O Register              */
       unsigned short PHCR;         /*  Port H Control Register          */
       unsigned short PHDR;         /*  Port H Data Register             */
};
#define PH   (*(volatile struct st_ph   *)0xFFFFF728)     /* PH Address */
/*-----------------------------------------------------------------------*/
/*                    Port J                                             */
/*-----------------------------------------------------------------------*/
volatile struct st_pj               /* struct PJ                         */
{
       unsigned short PJIOR;        /*  Port J I/O Register              */
       unsigned short PJCRH;        /*  Port J Control Register H        */
       unsigned short PJCRL;        /*  Port J Control Register L        */
       unsigned short PJDR;         /*  Port J Data Register             */
};
#define PJ   (*(volatile struct st_pj   *)0xFFFFF766)     /* PJ Address */
```

```
/*------------------------------------------------------------------------*/
/*                    Port K                                              */
/*------------------------------------------------------------------------*/
volatile struct st_pk               /* struct PK                          */
{
       unsigned short PKIOR;        /*  Port K I/O Register               */
       unsigned short PKCRH;        /*  Port K Control Register H         */
       unsigned short PKCRL;        /*  Port K Control Register L         */
       unsigned short PKIR;         /*  Port K Invert Register            */
       unsigned short PKDR;         /*  Port K Data Register              */
};
#define PK   (*(volatile struct st_pk   *)0xFFFFF770)     /* PK Address */
/*------------------------------------------------------------------------*/
/*                    Port L                                              */
/*------------------------------------------------------------------------*/
volatile struct st_pl               /* struct PL                          */
{
       unsigned short PLIOR;        /*  Port L I/O Register               */
       unsigned short PLCRH;        /*  Port L Control Register H         */
       unsigned short PLCRL;        /*  Port L Control Register L         */
       unsigned short PLIR;         /*  Port L Invert Register            */
       unsigned short PLDR;         /*  Port L Data Register              */
};
#define PL   (*(volatile struct st_pl   *)0xFFFFF756)     /* PL Address */
/************************************************************************/
/*            Compare Match Timer                                       */
/*------------------------------------------------------------------------*/
/*                    CMT(common)                                         */
/*------------------------------------------------------------------------*/
volatile struct st_cmtc             /* struct CMTC                        */
{
       unsigned short CMSTR;      /* Compare Match Timer Start Register */
};
#define CMTC (*(volatile struct st_cmtc *)0xFFFFF710) /* CMTC Address   */
/*------------------------------------------------------------------------*/
/*                    CMT0,1                                              */
/*------------------------------------------------------------------------*/
volatile struct st_cmt              /* struct CMT                         */
{
       unsigned short CMCSR;/* Compare Match Timer Control Status Register*/
       unsigned short CMCNT;/* Compare Match Timer Counter                */
       unsigned short CMCOR;/* Compare Match Timer Constant Register      */
};
#define CMT0 (*(volatile struct st_cmt *)0xFFFFF712)     /* CMT0 Address*/
#define CMT1 (*(volatile struct st_cmt *)0xFFFFF718)     /* CMT1 Address*/
/************************************************************************/
/*            Flash                                                     */
/************************************************************************/
volatile struct st_flash            /* struct FLASH                       */
{
       unsigned char FLMCR1;        /* Flash Memory Control Register 1  */
       unsigned char FLMCR2;        /* Flash Memory Control Register 2  */
       unsigned char EBR1;          /* Entry Block Resister 1           */
       unsigned char EBR2;          /* Entry Block Resister 2           */
};
#define FLASH (*(volatile struct st_flash *)0xFFFFE800)/* FLASH Address */
/************************************************************************/
/*            Advanced Pulse Controller                                 */
/************************************************************************/
volatile struct st_apc              /*struct APC                         */
{
       unsigned short POPCR;        /*Pulse Output Port Control Register*/
};
#define APC  (*(volatile struct st_apc  *)0xFFFFF700)     /* APC Address*/
```

```
/************************************************************************/
/*            A/D Converter                                             */
/*----------------------------------------------------------------------*/
/*                    A/D                                               */
/*----------------------------------------------------------------------*/
volatile struct st_ad               /* struct AD                        */
{
       unsigned short ADDR0;        /*  A/D Data Register 0             */
       unsigned short ADDR1;        /*  A/D Data Register 1             */
       unsigned short ADDR2;        /*  A/D Data Register 2             */
       unsigned short ADDR3;        /*  A/D Data Register 3             */
       unsigned short ADDR4;        /*  A/D Data Register 4             */
       unsigned short ADDR5;        /*  A/D Data Register 5             */
       unsigned short ADDR6;        /*  A/D Data Register 6             */
       unsigned short ADDR7;        /*  A/D Data Register 7             */
       unsigned short ADDR8;        /*  A/D Data Register 8             */
       unsigned short ADDR9;        /*  A/D Data Register 9             */
       unsigned short ADDR10;       /*  A/D Data Register 10            */
       unsigned short ADDR11;       /*  A/D Data Register 11            */
       unsigned char  ADCSR0;       /*  A/D Control Status Register 0   */
       unsigned char  ADCR0;        /*  A/D Control Register 0          */
       unsigned short DYF81A;       /*  Dummy Word                      */
       unsigned long  DYF81C;       /*  Dummy Long                      */
       unsigned short ADDR12;       /*  A/D Data Register 12            */
       unsigned short ADDR13;       /*  A/D Data Register 13            */
       unsigned short ADDR14;       /*  A/D Data Register 14            */
       unsigned short ADDR15;       /*  A/D Data Register 15            */
       unsigned short ADDR16;       /*  A/D Data Register 16            */
       unsigned short ADDR17;       /*  A/D Data Register 17            */
       unsigned short ADDR18;       /*  A/D Data Register 18            */
       unsigned short ADDR19;       /*  A/D Data Register 19            */
       unsigned short ADDR20;       /*  A/D Data Register 20            */
       unsigned short ADDR21;       /*  A/D Data Register 21            */
       unsigned short ADDR22;       /*  A/D Data Register 22            */
       unsigned short ADDR23;       /*  A/D Data Register 23            */
       unsigned char  ADCSR1;       /*  A/D Control Status Register 1   */
       unsigned char  ADCR1;        /*  A/D Control Register 1          */
       unsigned short DYF83A;       /*  Dummy Word                      */
       unsigned long  DYF83C;       /*  Dummy Long                      */
       unsigned short ADDR24;       /*  A/D Data Register 24            */
       unsigned short ADDR25;       /*  A/D Data Register 25            */
       unsigned short ADDR26;       /*  A/D Data Register 26            */
       unsigned short ADDR27;       /*  A/D Data Register 27            */
       unsigned short ADDR28;       /*  A/D Data Register 28            */
       unsigned short ADDR29;       /*  A/D Data Register 29            */
       unsigned short ADDR30;       /*  A/D Data Register 30            */
       unsigned short ADDR31;       /*  A/D Data Register 31            */
};
#define AD (*(volatile struct st_ad  *)0xFFFFF800)       /* A/D Address*/
#define ADTRGR1             (*(volatile unsigned char *)0xFFFFF72E)
#define ADTRGR2             (*(volatile unsigned char *)0xFFFFF72F)
#define ADTRGR0             (*(volatile unsigned char *)0xFFFFF76E)
#define ADCSR2              (*(volatile unsigned char *)0xFFFFF858)
#define ADCR2               (*(volatile unsigned char *)0xFFFFF859)
```

```
/************************************************************************/
/*            User Break Controller                                     */
/************************************************************************/
volatile struct st_ubc              /* struct UBC                       */
{
       unsigned short UBARH;        /* User Break Address Register H    */
       unsigned short UBARL;        /* User Break Address Register L    */
       unsigned short UBAMRH;      /* User Break Address Mask Register H */
       unsigned short UBAMRL;      /* User Break Address Mask Register L */
       unsigned short UBBR;         /* User Break Bus Cycle Register    */
       unsigned short UBCR;         /* User Break Control Register      */
};
#define UBC   (*(volatile struct st_ubc  *)0xFFFFEC00)    /* UBC Address*/
#define UBAR                (*(volatile unsigned long *)0xFFFFEC00)
#define UBAMR               (*(volatile unsigned long *)0xFFFFEC04)
/************************************************************************/
/*            Direct Memory Access Controller                           */
/*----------------------------------------------------------------------*/
/*                    DMAC(common)                                      */
/*----------------------------------------------------------------------*/
volatile struct st_dmacc            /* struct DMACC                     */
{
       unsigned short DMAOR;        /*  DMA Operation Register          */
};
#define DMACC (*(volatile struct st_dmacc *)0xFFFFECB0) /* DMACC Address*/
/*----------------------------------------------------------------------*/
/*                    DMAC0～3                                          */
/*----------------------------------------------------------------------*/
volatile struct st_dmac             /* struct DMAC0                     */
{
       unsigned long SAR;           /* Source Address Register          */
       unsigned long DAR;           /* Destination Address Register     */
       unsigned long DMATCR;        /* Transfer Count Register          */
       unsigned long CHCR;          /* Channel Control Register         */
};
#define DMAC0 (*(volatile struct st_dmac *)0xFFFFECC0) /* DMAC0 Address */
#define DMAC1 (*(volatile struct st_dmac *)0xFFFFECD0) /* DMAC1 Address */
#define DMAC2 (*(volatile struct st_dmac *)0xFFFFECE0) /* DMAC2 Address */
#define DMAC3 (*(volatile struct st_dmac *)0xFFFFECF0) /* DMAC3 Address */
/************************************************************************/
/*            Bus State Controller                                      */
/************************************************************************/
volatile struct st_bsc              /* struct BSC                       */
{
       unsigned short BCR1;         /* Bus Control Register 1           */
       unsigned short BCR2;         /* Bus Control Register 2           */
       unsigned short WCR;          /* Wait Control Register            */
       unsigned short RAMER;        /* RAM Enable Register              */
};
#define BSC  (*(volatile struct st_bsc  *)0xFFFFEC20)     /* BSC Address*/
```

```
/************************************************************************/
/*          HCAN  CHANEL0                                               */
/************************************************************************/

#define HCAN0_MCR           (*(volatile unsigned char  *)0xFFFFE400)
#define HCAN0_GSR           (*(volatile unsigned char  *)0xFFFFE401)
#define HCAN0_BCR           (*(volatile unsigned short *)0xFFFFE402)
#define HCAN0_MBCR          (*(volatile unsigned short *)0xFFFFE404)
#define HCAN0_TXPR          (*(volatile unsigned short *)0xFFFFE406)
#define HCAN0_TXCR          (*(volatile unsigned short *)0xFFFFE408)
#define HCAN0_TXACK         (*(volatile unsigned short *)0xFFFFE40A)
#define HCAN0_ABACK         (*(volatile unsigned short *)0xFFFFE40C)
#define HCAN0_RXPR          (*(volatile unsigned short *)0xFFFFE40E)
#define HCAN0_RFPR          (*(volatile unsigned short *)0xFFFFE410)
#define HCAN0_IRR           (*(volatile unsigned short *)0xFFFFE412)
#define HCAN0_MBIMR         (*(volatile unsigned short *)0xFFFFE414)
#define HCAN0_IMR           (*(volatile unsigned short *)0xFFFFE416)
#define HCAN0_REC           (*(volatile unsigned char  *)0xFFFFE418)
#define HCAN0_TEC           (*(volatile unsigned char  *)0xFFFFE419)
#define HCAN0_UMSR          (*(volatile unsigned short *)0xFFFFE41A)
#define HCAN0_LAFML         (*(volatile unsigned short *)0xFFFFE41C)
#define HCAN0_LAFMH         (*(volatile unsigned short *)0xFFFFE41E)
#define HCAN0_MC0_1         (*(volatile unsigned char *)0xFFFFE420)
#define HCAN0_MC0_2         (*(volatile unsigned char *)0xFFFFE421)
#define HCAN0_MC0_3         (*(volatile unsigned char *)0xFFFFE422)
#define HCAN0_MC0_4         (*(volatile unsigned char *)0xFFFFE423)
#define HCAN0_MC0_5         (*(volatile unsigned char *)0xFFFFE424)
#define HCAN0_MC0_6         (*(volatile unsigned char *)0xFFFFE425)
#define HCAN0_MC0_7         (*(volatile unsigned char *)0xFFFFE426)
#define HCAN0_MC0_8         (*(volatile unsigned char *)0xFFFFE427)
#define HCAN0_MC1_1         (*(volatile unsigned char *)0xFFFFE428)
#define HCAN0_MC1_2         (*(volatile unsigned char *)0xFFFFE429)
#define HCAN0_MC1_3         (*(volatile unsigned char *)0xFFFFE42A)
#define HCAN0_MC1_4         (*(volatile unsigned char *)0xFFFFE42B)
#define HCAN0_MC1_5         (*(volatile unsigned char *)0xFFFFE42C)
#define HCAN0_MC1_6         (*(volatile unsigned char *)0xFFFFE42D)
#define HCAN0_MC1_7         (*(volatile unsigned char *)0xFFFFE42E)
#define HCAN0_MC1_8         (*(volatile unsigned char *)0xFFFFE42F)
#define HCAN0_MC2_1         (*(volatile unsigned char *)0xFFFFE430)
#define HCAN0_MC2_2         (*(volatile unsigned char *)0xFFFFE431)
#define HCAN0_MC2_3         (*(volatile unsigned char *)0xFFFFE432)
#define HCAN0_MC2_4         (*(volatile unsigned char *)0xFFFFE433)
#define HCAN0_MC2_5         (*(volatile unsigned char *)0xFFFFE434)
#define HCAN0_MC2_6         (*(volatile unsigned char *)0xFFFFE435)
#define HCAN0_MC2_7         (*(volatile unsigned char *)0xFFFFE436)
#define HCAN0_MC2_8         (*(volatile unsigned char *)0xFFFFE437)
```

```
#define HCAN0_MC3_1         (*(volatile unsigned char *)0xFFFFE438)
#define HCAN0_MC3_2         (*(volatile unsigned char *)0xFFFFE439)
#define HCAN0_MC3_3         (*(volatile unsigned char *)0xFFFFE43A)
#define HCAN0_MC3_4         (*(volatile unsigned char *)0xFFFFE43B)
#define HCAN0_MC3_5         (*(volatile unsigned char *)0xFFFFE43C)
#define HCAN0_MC3_6         (*(volatile unsigned char *)0xFFFFE43D)
#define HCAN0_MC3_7         (*(volatile unsigned char *)0xFFFFE43E)
#define HCAN0_MC3_8         (*(volatile unsigned char *)0xFFFFE43F)
#define HCAN0_MC4_1         (*(volatile unsigned char *)0xFFFFE440)
#define HCAN0_MC4_2         (*(volatile unsigned char *)0xFFFFE441)
#define HCAN0_MC4_3         (*(volatile unsigned char *)0xFFFFE442)
#define HCAN0_MC4_4         (*(volatile unsigned char *)0xFFFFE443)
#define HCAN0_MC4_5         (*(volatile unsigned char *)0xFFFFE444)
#define HCAN0_MC4_6         (*(volatile unsigned char *)0xFFFFE445)
#define HCAN0_MC4_7         (*(volatile unsigned char *)0xFFFFE446)
#define HCAN0_MC4_8         (*(volatile unsigned char *)0xFFFFE447)
#define HCAN0_MC5_1         (*(volatile unsigned char *)0xFFFFE448)
#define HCAN0_MC5_2         (*(volatile unsigned char *)0xFFFFE449)
#define HCAN0_MC5_3         (*(volatile unsigned char *)0xFFFFE44A)
#define HCAN0_MC5_4         (*(volatile unsigned char *)0xFFFFE44B)
#define HCAN0_MC5_5         (*(volatile unsigned char *)0xFFFFE44C)
#define HCAN0_MC5_6         (*(volatile unsigned char *)0xFFFFE44D)
#define HCAN0_MC5_7         (*(volatile unsigned char *)0xFFFFE44E)
#define HCAN0_MC5_8         (*(volatile unsigned char *)0xFFFFE44F)
#define HCAN0_MC6_1         (*(volatile unsigned char *)0xFFFFE450)
#define HCAN0_MC6_2         (*(volatile unsigned char *)0xFFFFE451)
#define HCAN0_MC6_3         (*(volatile unsigned char *)0xFFFFE452)
#define HCAN0_MC6_4         (*(volatile unsigned char *)0xFFFFE453)
#define HCAN0_MC6_5         (*(volatile unsigned char *)0xFFFFE454)
#define HCAN0_MC6_6         (*(volatile unsigned char *)0xFFFFE455)
#define HCAN0_MC6_7         (*(volatile unsigned char *)0xFFFFE456)
#define HCAN0_MC6_8         (*(volatile unsigned char *)0xFFFFE457)
#define HCAN0_MC7_1         (*(volatile unsigned char *)0xFFFFE458)
#define HCAN0_MC7_2         (*(volatile unsigned char *)0xFFFFE459)
#define HCAN0_MC7_3         (*(volatile unsigned char *)0xFFFFE45A)
#define HCAN0_MC7_4         (*(volatile unsigned char *)0xFFFFE45B)
#define HCAN0_MC7_5         (*(volatile unsigned char *)0xFFFFE45C)
#define HCAN0_MC7_6         (*(volatile unsigned char *)0xFFFFE45D)
#define HCAN0_MC7_7         (*(volatile unsigned char *)0xFFFFE45E)
#define HCAN0_MC7_8         (*(volatile unsigned char *)0xFFFFE45F)
#define HCAN0_MC8_1         (*(volatile unsigned char *)0xFFFFE460)
#define HCAN0_MC8_2         (*(volatile unsigned char *)0xFFFFE461)
#define HCAN0_MC8_3         (*(volatile unsigned char *)0xFFFFE462)
#define HCAN0_MC8_4         (*(volatile unsigned char *)0xFFFFE463)
#define HCAN0_MC8_5         (*(volatile unsigned char *)0xFFFFE464)
#define HCAN0_MC8_6         (*(volatile unsigned char *)0xFFFFE465)
#define HCAN0_MC8_7         (*(volatile unsigned char *)0xFFFFE466)
#define HCAN0_MC8_8         (*(volatile unsigned char *)0xFFFFE467)
```

```
#define HCAN0_MC9_1         (*(volatile unsigned char *)0xFFFFE468)
#define HCAN0_MC9_2         (*(volatile unsigned char *)0xFFFFE469)
#define HCAN0_MC9_3         (*(volatile unsigned char *)0xFFFFE46A)
#define HCAN0_MC9_4         (*(volatile unsigned char *)0xFFFFE46B)
#define HCAN0_MC9_5         (*(volatile unsigned char *)0xFFFFE46C)
#define HCAN0_MC9_6         (*(volatile unsigned char *)0xFFFFE46D)
#define HCAN0_MC9_7         (*(volatile unsigned char *)0xFFFFE46E)
#define HCAN0_MC9_8         (*(volatile unsigned char *)0xFFFFE46F)
#define HCAN0_MC10_1        (*(volatile unsigned char *)0xFFFFE470)
#define HCAN0_MC10_2        (*(volatile unsigned char *)0xFFFFE471)
#define HCAN0_MC10_3        (*(volatile unsigned char *)0xFFFFE472)
#define HCAN0_MC10_4        (*(volatile unsigned char *)0xFFFFE473)
#define HCAN0_MC10_5        (*(volatile unsigned char *)0xFFFFE474)
#define HCAN0_MC10_6        (*(volatile unsigned char *)0xFFFFE475)
#define HCAN0_MC10_7        (*(volatile unsigned char *)0xFFFFE476)
#define HCAN0_MC10_8        (*(volatile unsigned char *)0xFFFFE477)
#define HCAN0_MC11_1        (*(volatile unsigned char *)0xFFFFE478)
#define HCAN0_MC11_2        (*(volatile unsigned char *)0xFFFFE479)
#define HCAN0_MC11_3        (*(volatile unsigned char *)0xFFFFE47A)
#define HCAN0_MC11_4        (*(volatile unsigned char *)0xFFFFE47B)
#define HCAN0_MC11_5        (*(volatile unsigned char *)0xFFFFE47C)
#define HCAN0_MC11_6        (*(volatile unsigned char *)0xFFFFE47D)
#define HCAN0_MC11_7        (*(volatile unsigned char *)0xFFFFE47E)
#define HCAN0_MC11_8        (*(volatile unsigned char *)0xFFFFE47F)
#define HCAN0_MC12_1        (*(volatile unsigned char *)0xFFFFE480)
#define HCAN0_MC12_2        (*(volatile unsigned char *)0xFFFFE481)
#define HCAN0_MC12_3        (*(volatile unsigned char *)0xFFFFE482)
#define HCAN0_MC12_4        (*(volatile unsigned char *)0xFFFFE483)
#define HCAN0_MC12_5        (*(volatile unsigned char *)0xFFFFE484)
#define HCAN0_MC12_6        (*(volatile unsigned char *)0xFFFFE485)
#define HCAN0_MC12_7        (*(volatile unsigned char *)0xFFFFE486)
#define HCAN0_MC12_8        (*(volatile unsigned char *)0xFFFFE487)
#define HCAN0_MC13_1        (*(volatile unsigned char *)0xFFFFE488)
#define HCAN0_MC13_2        (*(volatile unsigned char *)0xFFFFE489)
#define HCAN0_MC13_3        (*(volatile unsigned char *)0xFFFFE48A)
#define HCAN0_MC13_4        (*(volatile unsigned char *)0xFFFFE48B)
#define HCAN0_MC13_5        (*(volatile unsigned char *)0xFFFFE48C)
#define HCAN0_MC13_6        (*(volatile unsigned char *)0xFFFFE48D)
#define HCAN0_MC13_7        (*(volatile unsigned char *)0xFFFFE48E)
#define HCAN0_MC13_8        (*(volatile unsigned char *)0xFFFFE48F)
#define HCAN0_MC14_1        (*(volatile unsigned char *)0xFFFFE490)
#define HCAN0_MC14_2        (*(volatile unsigned char *)0xFFFFE491)
#define HCAN0_MC14_3        (*(volatile unsigned char *)0xFFFFE492)
#define HCAN0_MC14_4        (*(volatile unsigned char *)0xFFFFE493)
#define HCAN0_MC14_5        (*(volatile unsigned char *)0xFFFFE494)
#define HCAN0_MC14_6        (*(volatile unsigned char *)0xFFFFE495)
#define HCAN0_MC14_7        (*(volatile unsigned char *)0xFFFFE496)
#define HCAN0_MC14_8        (*(volatile unsigned char *)0xFFFFE497)
```

```
#define HCAN0_MC15_1        (*(volatile unsigned char *)0xFFFFE498)
#define HCAN0_MC15_2        (*(volatile unsigned char *)0xFFFFE499)
#define HCAN0_MC15_3        (*(volatile unsigned char *)0xFFFFE49A)
#define HCAN0_MC15_4        (*(volatile unsigned char *)0xFFFFE49B)
#define HCAN0_MC15_5        (*(volatile unsigned char *)0xFFFFE49C)
#define HCAN0_MC15_6        (*(volatile unsigned char *)0xFFFFE49D)
#define HCAN0_MC15_7        (*(volatile unsigned char *)0xFFFFE49E)
#define HCAN0_MC15_8        (*(volatile unsigned char *)0xFFFFE49F)
#define HCAN0_MD0_1         (*(volatile unsigned char *)0xFFFFE4B0)
#define HCAN0_MD0_2         (*(volatile unsigned char *)0xFFFFE4B1)
#define HCAN0_MD0_3         (*(volatile unsigned char *)0xFFFFE4B2)
#define HCAN0_MD0_4         (*(volatile unsigned char *)0xFFFFE4B3)
#define HCAN0_MD0_5         (*(volatile unsigned char *)0xFFFFE4B4)
#define HCAN0_MD0_6         (*(volatile unsigned char *)0xFFFFE4B5)
#define HCAN0_MD0_7         (*(volatile unsigned char *)0xFFFFE4B6)
#define HCAN0_MD0_8         (*(volatile unsigned char *)0xFFFFE4B7)
#define HCAN0_MD1_1         (*(volatile unsigned char *)0xFFFFE4B8)
#define HCAN0_MD1_2         (*(volatile unsigned char *)0xFFFFE4B9)
#define HCAN0_MD1_3         (*(volatile unsigned char *)0xFFFFE4BA)
#define HCAN0_MD1_4         (*(volatile unsigned char *)0xFFFFE4BB)
#define HCAN0_MD1_5         (*(volatile unsigned char *)0xFFFFE4BC)
#define HCAN0_MD1_6         (*(volatile unsigned char *)0xFFFFE4BD)
#define HCAN0_MD1_7         (*(volatile unsigned char *)0xFFFFE4BE)
#define HCAN0_MD1_8         (*(volatile unsigned char *)0xFFFFE4BF)
#define HCAN0_MD2_1         (*(volatile unsigned char *)0xFFFFE4C0)
#define HCAN0_MD2_2         (*(volatile unsigned char *)0xFFFFE4C1)
#define HCAN0_MD2_3         (*(volatile unsigned char *)0xFFFFE4C2)
#define HCAN0_MD2_4         (*(volatile unsigned char *)0xFFFFE4C3)
#define HCAN0_MD2_5         (*(volatile unsigned char *)0xFFFFE4C4)
#define HCAN0_MD2_6         (*(volatile unsigned char *)0xFFFFE4C5)
#define HCAN0_MD2_7         (*(volatile unsigned char *)0xFFFFE4C6)
#define HCAN0_MD2_8         (*(volatile unsigned char *)0xFFFFE4C7)
#define HCAN0_MD3_1         (*(volatile unsigned char *)0xFFFFE4C8)
#define HCAN0_MD3_2         (*(volatile unsigned char *)0xFFFFE4C9)
#define HCAN0_MD3_3         (*(volatile unsigned char *)0xFFFFE4CA)
#define HCAN0_MD3_4         (*(volatile unsigned char *)0xFFFFE4CB)
#define HCAN0_MD3_5         (*(volatile unsigned char *)0xFFFFE4CC)
#define HCAN0_MD3_6         (*(volatile unsigned char *)0xFFFFE4CD)
#define HCAN0_MD3_7         (*(volatile unsigned char *)0xFFFFE4CE)
#define HCAN0_MD3_8         (*(volatile unsigned char *)0xFFFFE4CF)
#define HCAN0_MD4_1         (*(volatile unsigned char *)0xFFFFE4D0)
#define HCAN0_MD4_2         (*(volatile unsigned char *)0xFFFFE4D1)
#define HCAN0_MD4_3         (*(volatile unsigned char *)0xFFFFE4D2)
#define HCAN0_MD4_4         (*(volatile unsigned char *)0xFFFFE4D3)
#define HCAN0_MD4_5         (*(volatile unsigned char *)0xFFFFE4D4)
#define HCAN0_MD4_6         (*(volatile unsigned char *)0xFFFFE4D5)
#define HCAN0_MD4_7         (*(volatile unsigned char *)0xFFFFE4D6)
#define HCAN0_MD4_8         (*(volatile unsigned char *)0xFFFFE4D7)
```

```
#define HCAN0_MD5_1         (*(volatile unsigned char *)0xFFFFE4D8)
#define HCAN0_MD5_2         (*(volatile unsigned char *)0xFFFFE4D9)
#define HCAN0_MD5_3         (*(volatile unsigned char *)0xFFFFE4DA)
#define HCAN0_MD5_4         (*(volatile unsigned char *)0xFFFFE4DB)
#define HCAN0_MD5_5         (*(volatile unsigned char *)0xFFFFE4DC)
#define HCAN0_MD5_6         (*(volatile unsigned char *)0xFFFFE4DD)
#define HCAN0_MD5_7         (*(volatile unsigned char *)0xFFFFE4DE)
#define HCAN0_MD5_8         (*(volatile unsigned char *)0xFFFFE4DF)
#define HCAN0_MD6_1         (*(volatile unsigned char *)0xFFFFE4E0)
#define HCAN0_MD6_2         (*(volatile unsigned char *)0xFFFFE4E1)
#define HCAN0_MD6_3         (*(volatile unsigned char *)0xFFFFE4E2)
#define HCAN0_MD6_4         (*(volatile unsigned char *)0xFFFFE4E3)
#define HCAN0_MD6_5         (*(volatile unsigned char *)0xFFFFE4E4)
#define HCAN0_MD6_6         (*(volatile unsigned char *)0xFFFFE4E5)
#define HCAN0_MD6_7         (*(volatile unsigned char *)0xFFFFE4E6)
#define HCAN0_MD6_8         (*(volatile unsigned char *)0xFFFFE4E7)
#define HCAN0_MD7_1         (*(volatile unsigned char *)0xFFFFE4E8)
#define HCAN0_MD7_2         (*(volatile unsigned char *)0xFFFFE4E9)
#define HCAN0_MD7_3         (*(volatile unsigned char *)0xFFFFE4EA)
#define HCAN0_MD7_4         (*(volatile unsigned char *)0xFFFFE4EB)
#define HCAN0_MD7_5         (*(volatile unsigned char *)0xFFFFE4EC)
#define HCAN0_MD7_6         (*(volatile unsigned char *)0xFFFFE4ED)
#define HCAN0_MD7_7         (*(volatile unsigned char *)0xFFFFE4EE)
#define HCAN0_MD7_8         (*(volatile unsigned char *)0xFFFFE4EF)
#define HCAN0_MD8_1         (*(volatile unsigned char *)0xFFFFE4F0)
#define HCAN0_MD8_2         (*(volatile unsigned char *)0xFFFFE4F1)
#define HCAN0_MD8_3         (*(volatile unsigned char *)0xFFFFE4F2)
#define HCAN0_MD8_4         (*(volatile unsigned char *)0xFFFFE4F3)
#define HCAN0_MD8_5         (*(volatile unsigned char *)0xFFFFE4F4)
#define HCAN0_MD8_6         (*(volatile unsigned char *)0xFFFFE4F5)
#define HCAN0_MD8_7         (*(volatile unsigned char *)0xFFFFE4F6)
#define HCAN0_MD8_8         (*(volatile unsigned char *)0xFFFFE4F7)
#define HCAN0_MD9_1         (*(volatile unsigned char *)0xFFFFE4F8)
#define HCAN0_MD9_2         (*(volatile unsigned char *)0xFFFFE4F9)
#define HCAN0_MD9_3         (*(volatile unsigned char *)0xFFFFE4FA)
#define HCAN0_MD9_4         (*(volatile unsigned char *)0xFFFFE4FB)
#define HCAN0_MD9_5         (*(volatile unsigned char *)0xFFFFE4FC)
#define HCAN0_MD9_6         (*(volatile unsigned char *)0xFFFFE4FD)
#define HCAN0_MD9_7         (*(volatile unsigned char *)0xFFFFE4FE)
#define HCAN0_MD9_8         (*(volatile unsigned char *)0xFFFFE4FF)
#define HCAN0_MD10_1        (*(volatile unsigned char *)0xFFFFE500)
#define HCAN0_MD10_2        (*(volatile unsigned char *)0xFFFFE501)
#define HCAN0_MD10_3        (*(volatile unsigned char *)0xFFFFE502)
#define HCAN0_MD10_4        (*(volatile unsigned char *)0xFFFFE503)
#define HCAN0_MD10_5        (*(volatile unsigned char *)0xFFFFE504)
#define HCAN0_MD10_6        (*(volatile unsigned char *)0xFFFFE505)
#define HCAN0_MD10_7        (*(volatile unsigned char *)0xFFFFE506)
#define HCAN0_MD10_8        (*(volatile unsigned char *)0xFFFFE507)
```

```
#define HCAN0_MD11_1        (*(volatile unsigned char *)0xFFFFE508)
#define HCAN0_MD11_2        (*(volatile unsigned char *)0xFFFFE509)
#define HCAN0_MD11_3        (*(volatile unsigned char *)0xFFFFE50A)
#define HCAN0_MD11_4        (*(volatile unsigned char *)0xFFFFE50B)
#define HCAN0_MD11_5        (*(volatile unsigned char *)0xFFFFE50C)
#define HCAN0_MD11_6        (*(volatile unsigned char *)0xFFFFE50D)
#define HCAN0_MD11_7        (*(volatile unsigned char *)0xFFFFE50E)
#define HCAN0_MD11_8        (*(volatile unsigned char *)0xFFFFE50F)
#define HCAN0_MD12_1        (*(volatile unsigned char *)0xFFFFE510)
#define HCAN0_MD12_2        (*(volatile unsigned char *)0xFFFFE511)
#define HCAN0_MD12_3        (*(volatile unsigned char *)0xFFFFE512)
#define HCAN0_MD12_4        (*(volatile unsigned char *)0xFFFFE513)
#define HCAN0_MD12_5        (*(volatile unsigned char *)0xFFFFE514)
#define HCAN0_MD12_6        (*(volatile unsigned char *)0xFFFFE515)
#define HCAN0_MD12_7        (*(volatile unsigned char *)0xFFFFE516)
#define HCAN0_MD12_8        (*(volatile unsigned char *)0xFFFFE517)
#define HCAN0_MD13_1        (*(volatile unsigned char *)0xFFFFE518)
#define HCAN0_MD13_2        (*(volatile unsigned char *)0xFFFFE519)
#define HCAN0_MD13_3        (*(volatile unsigned char *)0xFFFFE51A)
#define HCAN0_MD13_4        (*(volatile unsigned char *)0xFFFFE51B)
#define HCAN0_MD13_5        (*(volatile unsigned char *)0xFFFFE51C)
#define HCAN0_MD13_6        (*(volatile unsigned char *)0xFFFFE51D)
#define HCAN0_MD13_7        (*(volatile unsigned char *)0xFFFFE51E)
#define HCAN0_MD13_8        (*(volatile unsigned char *)0xFFFFE51F)
#define HCAN0_MD14_1        (*(volatile unsigned char *)0xFFFFE520)
#define HCAN0_MD14_2        (*(volatile unsigned char *)0xFFFFE521)
#define HCAN0_MD14_3        (*(volatile unsigned char *)0xFFFFE522)
#define HCAN0_MD14_4        (*(volatile unsigned char *)0xFFFFE523)
#define HCAN0_MD14_5        (*(volatile unsigned char *)0xFFFFE524)
#define HCAN0_MD14_6        (*(volatile unsigned char *)0xFFFFE525)
#define HCAN0_MD14_7        (*(volatile unsigned char *)0xFFFFE526)
#define HCAN0_MD14_8        (*(volatile unsigned char *)0xFFFFE527)
#define HCAN0_MD15_1        (*(volatile unsigned char *)0xFFFFE528)
#define HCAN0_MD15_2        (*(volatile unsigned char *)0xFFFFE529)
#define HCAN0_MD15_3        (*(volatile unsigned char *)0xFFFFE52A)
#define HCAN0_MD15_4        (*(volatile unsigned char *)0xFFFFE52B)
#define HCAN0_MD15_5        (*(volatile unsigned char *)0xFFFFE52C)
#define HCAN0_MD15_6        (*(volatile unsigned char *)0xFFFFE52D)
#define HCAN0_MD15_7        (*(volatile unsigned char *)0xFFFFE52E)
#define HCAN0_MD15_8        (*(volatile unsigned char *)0xFFFFE52F)
```

```
/************************************************************************/
/*              HCAN  CHANEL1                                           */
/************************************************************************/
#define HCAN1_MCR           (*(volatile unsigned char  *)0xFFFFE600)
#define HCAN1_GSR           (*(volatile unsigned char  *)0xFFFFE601)
#define HCAN1_BCR           (*(volatile unsigned short *)0xFFFFE602)
#define HCAN1_MBCR          (*(volatile unsigned short *)0xFFFFE604)
#define HCAN1_TXPR          (*(volatile unsigned short *)0xFFFFE606)
#define HCAN1_TXCR          (*(volatile unsigned short *)0xFFFFE608)
#define HCAN1_TXACK         (*(volatile unsigned short *)0xFFFFE60A)
#define HCAN1_ABACK         (*(volatile unsigned short *)0xFFFFE60C)
#define HCAN1_RXPR          (*(volatile unsigned short *)0xFFFFE60E)
#define HCAN1_RFPR          (*(volatile unsigned short *)0xFFFFE610)
#define HCAN1_IRR           (*(volatile unsigned short *)0xFFFFE612)
#define HCAN1_MBIMR         (*(volatile unsigned short *)0xFFFFE614)
#define HCAN1_IMR           (*(volatile unsigned short *)0xFFFFE616)
#define HCAN1_REC           (*(volatile unsigned char  *)0xFFFFE618)
#define HCAN1_TEC           (*(volatile unsigned char  *)0xFFFFE619)
#define HCAN1_UMSR          (*(volatile unsigned short *)0xFFFFE61A)
#define HCAN1_LAFML         (*(volatile unsigned short *)0xFFFFE61C)
#define HCAN1_LAFMH         (*(volatile unsigned short *)0xFFFFE61E)
#define HCAN1_MC0_1         (*(volatile unsigned char *)0xFFFFE620)
#define HCAN1_MC0_2         (*(volatile unsigned char *)0xFFFFE621)
#define HCAN1_MC0_3         (*(volatile unsigned char *)0xFFFFE622)
#define HCAN1_MC0_4         (*(volatile unsigned char *)0xFFFFE623)
#define HCAN1_MC0_5         (*(volatile unsigned char *)0xFFFFE624)
#define HCAN1_MC0_6         (*(volatile unsigned char *)0xFFFFE625)
#define HCAN1_MC0_7         (*(volatile unsigned char *)0xFFFFE626)
#define HCAN1_MC0_8         (*(volatile unsigned char *)0xFFFFE627)
#define HCAN1_MC1_1         (*(volatile unsigned char *)0xFFFFE628)
#define HCAN1_MC1_2         (*(volatile unsigned char *)0xFFFFE629)
#define HCAN1_MC1_3         (*(volatile unsigned char *)0xFFFFE62A)
#define HCAN1_MC1_4         (*(volatile unsigned char *)0xFFFFE62B)
#define HCAN1_MC1_5         (*(volatile unsigned char *)0xFFFFE62C)
#define HCAN1_MC1_6         (*(volatile unsigned char *)0xFFFFE62D)
#define HCAN1_MC1_7         (*(volatile unsigned char *)0xFFFFE62E)
#define HCAN1_MC1_8         (*(volatile unsigned char *)0xFFFFE62F)
#define HCAN1_MC2_1         (*(volatile unsigned char *)0xFFFFE630)
#define HCAN1_MC2_2         (*(volatile unsigned char *)0xFFFFE631)
#define HCAN1_MC2_3         (*(volatile unsigned char *)0xFFFFE632)
#define HCAN1_MC2_4         (*(volatile unsigned char *)0xFFFFE633)
#define HCAN1_MC2_5         (*(volatile unsigned char *)0xFFFFE634)
#define HCAN1_MC2_6         (*(volatile unsigned char *)0xFFFFE635)
#define HCAN1_MC2_7         (*(volatile unsigned char *)0xFFFFE636)
#define HCAN1_MC2_8         (*(volatile unsigned char *)0xFFFFE637)
```

```
#define HCAN1_MC3_1         (*(volatile unsigned char *)0xFFFFE638)
#define HCAN1_MC3_2         (*(volatile unsigned char *)0xFFFFE639)
#define HCAN1_MC3_3         (*(volatile unsigned char *)0xFFFFE63A)
#define HCAN1_MC3_4         (*(volatile unsigned char *)0xFFFFE63B)
#define HCAN1_MC3_5         (*(volatile unsigned char *)0xFFFFE63C)
#define HCAN1_MC3_6         (*(volatile unsigned char *)0xFFFFE63D)
#define HCAN1_MC3_7         (*(volatile unsigned char *)0xFFFFE63E)
#define HCAN1_MC3_8         (*(volatile unsigned char *)0xFFFFE63F)
#define HCAN1_MC4_1         (*(volatile unsigned char *)0xFFFFE640)
#define HCAN1_MC4_2         (*(volatile unsigned char *)0xFFFFE641)
#define HCAN1_MC4_3         (*(volatile unsigned char *)0xFFFFE642)
#define HCAN1_MC4_4         (*(volatile unsigned char *)0xFFFFE643)
#define HCAN1_MC4_5         (*(volatile unsigned char *)0xFFFFE644)
#define HCAN1_MC4_6         (*(volatile unsigned char *)0xFFFFE645)
#define HCAN1_MC4_7         (*(volatile unsigned char *)0xFFFFE646)
#define HCAN1_MC4_8         (*(volatile unsigned char *)0xFFFFE647)
#define HCAN1_MC5_1         (*(volatile unsigned char *)0xFFFFE648)
#define HCAN1_MC5_2         (*(volatile unsigned char *)0xFFFFE649)
#define HCAN1_MC5_3         (*(volatile unsigned char *)0xFFFFE64A)
#define HCAN1_MC5_4         (*(volatile unsigned char *)0xFFFFE64B)
#define HCAN1_MC5_5         (*(volatile unsigned char *)0xFFFFE64C)
#define HCAN1_MC5_6         (*(volatile unsigned char *)0xFFFFE64D)
#define HCAN1_MC5_7         (*(volatile unsigned char *)0xFFFFE64E)
#define HCAN1_MC5_8         (*(volatile unsigned char *)0xFFFFE64F)
#define HCAN1_MC6_1         (*(volatile unsigned char *)0xFFFFE650)
#define HCAN1_MC6_2         (*(volatile unsigned char *)0xFFFFE651)
#define HCAN1_MC6_3         (*(volatile unsigned char *)0xFFFFE652)
#define HCAN1_MC6_4         (*(volatile unsigned char *)0xFFFFE653)
#define HCAN1_MC6_5         (*(volatile unsigned char *)0xFFFFE654)
#define HCAN1_MC6_6         (*(volatile unsigned char *)0xFFFFE655)
#define HCAN1_MC6_7         (*(volatile unsigned char *)0xFFFFE656)
#define HCAN1_MC6_8         (*(volatile unsigned char *)0xFFFFE657)
#define HCAN1_MC7_1         (*(volatile unsigned char *)0xFFFFE658)
#define HCAN1_MC7_2         (*(volatile unsigned char *)0xFFFFE659)
#define HCAN1_MC7_3         (*(volatile unsigned char *)0xFFFFE65A)
#define HCAN1_MC7_4         (*(volatile unsigned char *)0xFFFFE65B)
#define HCAN1_MC7_5         (*(volatile unsigned char *)0xFFFFE65C)
#define HCAN1_MC7_6         (*(volatile unsigned char *)0xFFFFE65D)
#define HCAN1_MC7_7         (*(volatile unsigned char *)0xFFFFE65E)
#define HCAN1_MC7_8         (*(volatile unsigned char *)0xFFFFE65F)
#define HCAN1_MC8_1         (*(volatile unsigned char *)0xFFFFE660)
#define HCAN1_MC8_2         (*(volatile unsigned char *)0xFFFFE661)
#define HCAN1_MC8_3         (*(volatile unsigned char *)0xFFFFE662)
#define HCAN1_MC8_4         (*(volatile unsigned char *)0xFFFFE663)
#define HCAN1_MC8_5         (*(volatile unsigned char *)0xFFFFE664)
#define HCAN1_MC8_6         (*(volatile unsigned char *)0xFFFFE665)
#define HCAN1_MC8_7         (*(volatile unsigned char *)0xFFFFE666)
#define HCAN1_MC8_8         (*(volatile unsigned char *)0xFFFFE667)
```

```
#define HCAN1_MC9_1         (*(volatile unsigned char *)0xFFFFE668)
#define HCAN1_MC9_2         (*(volatile unsigned char *)0xFFFFE669)
#define HCAN1_MC9_3         (*(volatile unsigned char *)0xFFFFE66A)
#define HCAN1_MC9_4         (*(volatile unsigned char *)0xFFFFE66B)
#define HCAN1_MC9_5         (*(volatile unsigned char *)0xFFFFE66C)
#define HCAN1_MC9_6         (*(volatile unsigned char *)0xFFFFE66D)
#define HCAN1_MC9_7         (*(volatile unsigned char *)0xFFFFE66E)
#define HCAN1_MC9_8         (*(volatile unsigned char *)0xFFFFE66F)
#define HCAN1_MC10_1        (*(volatile unsigned char *)0xFFFFE670)
#define HCAN1_MC10_2        (*(volatile unsigned char *)0xFFFFE671)
#define HCAN1_MC10_3        (*(volatile unsigned char *)0xFFFFE672)
#define HCAN1_MC10_4        (*(volatile unsigned char *)0xFFFFE673)
#define HCAN1_MC10_5        (*(volatile unsigned char *)0xFFFFE674)
#define HCAN1_MC10_6        (*(volatile unsigned char *)0xFFFFE675)
#define HCAN1_MC10_7        (*(volatile unsigned char *)0xFFFFE676)
#define HCAN1_MC10_8        (*(volatile unsigned char *)0xFFFFE677)
#define HCAN1_MC11_1        (*(volatile unsigned char *)0xFFFFE678)
#define HCAN1_MC11_2        (*(volatile unsigned char *)0xFFFFE679)
#define HCAN1_MC11_3        (*(volatile unsigned char *)0xFFFFE67A)
#define HCAN1_MC11_4        (*(volatile unsigned char *)0xFFFFE67B)
#define HCAN1_MC11_5        (*(volatile unsigned char *)0xFFFFE67C)
#define HCAN1_MC11_6        (*(volatile unsigned char *)0xFFFFE67D)
#define HCAN1_MC11_7        (*(volatile unsigned char *)0xFFFFE67E)
#define HCAN1_MC11_8        (*(volatile unsigned char *)0xFFFFE67F)
#define HCAN1_MC12_1        (*(volatile unsigned char *)0xFFFFE680)
#define HCAN1_MC12_2        (*(volatile unsigned char *)0xFFFFE681)
#define HCAN1_MC12_3        (*(volatile unsigned char *)0xFFFFE682)
#define HCAN1_MC12_4        (*(volatile unsigned char *)0xFFFFE683)
#define HCAN1_MC12_5        (*(volatile unsigned char *)0xFFFFE684)
#define HCAN1_MC12_6        (*(volatile unsigned char *)0xFFFFE685)
#define HCAN1_MC12_7        (*(volatile unsigned char *)0xFFFFE686)
#define HCAN1_MC12_8        (*(volatile unsigned char *)0xFFFFE687)
#define HCAN1_MC13_1        (*(volatile unsigned char *)0xFFFFE688)
#define HCAN1_MC13_2        (*(volatile unsigned char *)0xFFFFE689)
#define HCAN1_MC13_3        (*(volatile unsigned char *)0xFFFFE68A)
#define HCAN1_MC13_4        (*(volatile unsigned char *)0xFFFFE68B)
#define HCAN1_MC13_5        (*(volatile unsigned char *)0xFFFFE68C)
#define HCAN1_MC13_6        (*(volatile unsigned char *)0xFFFFE68D)
#define HCAN1_MC13_7        (*(volatile unsigned char *)0xFFFFE68E)
#define HCAN1_MC13_8        (*(volatile unsigned char *)0xFFFFE68F)
#define HCAN1_MC14_1        (*(volatile unsigned char *)0xFFFFE690)
#define HCAN1_MC14_2        (*(volatile unsigned char *)0xFFFFE691)
#define HCAN1_MC14_3        (*(volatile unsigned char *)0xFFFFE692)
#define HCAN1_MC14_4        (*(volatile unsigned char *)0xFFFFE693)
#define HCAN1_MC14_5        (*(volatile unsigned char *)0xFFFFE694)
#define HCAN1_MC14_6        (*(volatile unsigned char *)0xFFFFE695)
#define HCAN1_MC14_7        (*(volatile unsigned char *)0xFFFFE696)
#define HCAN1_MC14_8        (*(volatile unsigned char *)0xFFFFE697)
```

```
#define HCAN1_MC15_1        (*(volatile unsigned char *)0xFFFFE698)
#define HCAN1_MC15_2        (*(volatile unsigned char *)0xFFFFE699)
#define HCAN1_MC15_3        (*(volatile unsigned char *)0xFFFFE69A)
#define HCAN1_MC15_4        (*(volatile unsigned char *)0xFFFFE69B)
#define HCAN1_MC15_5        (*(volatile unsigned char *)0xFFFFE69C)
#define HCAN1_MC15_6        (*(volatile unsigned char *)0xFFFFE69D)
#define HCAN1_MC15_7        (*(volatile unsigned char *)0xFFFFE69E)
#define HCAN1_MC15_8        (*(volatile unsigned char *)0xFFFFE69F)
#define HCAN1_MD0_1         (*(volatile unsigned char *)0xFFFFE6B0)
#define HCAN1_MD0_2         (*(volatile unsigned char *)0xFFFFE6B1)
#define HCAN1_MD0_3         (*(volatile unsigned char *)0xFFFFE6B2)
#define HCAN1_MD0_4         (*(volatile unsigned char *)0xFFFFE6B3)
#define HCAN1_MD0_5         (*(volatile unsigned char *)0xFFFFE6B4)
#define HCAN1_MD0_6         (*(volatile unsigned char *)0xFFFFE6B5)
#define HCAN1_MD0_7         (*(volatile unsigned char *)0xFFFFE6B6)
#define HCAN1_MD0_8         (*(volatile unsigned char *)0xFFFFE6B7)
#define HCAN1_MD1_1         (*(volatile unsigned char *)0xFFFFE6B8)
#define HCAN1_MD1_2         (*(volatile unsigned char *)0xFFFFE6B9)
#define HCAN1_MD1_3         (*(volatile unsigned char *)0xFFFFE6BA)
#define HCAN1_MD1_4         (*(volatile unsigned char *)0xFFFFE6BB)
#define HCAN1_MD1_5         (*(volatile unsigned char *)0xFFFFE6BC)
#define HCAN1_MD1_6         (*(volatile unsigned char *)0xFFFFE6BD)
#define HCAN1_MD1_7         (*(volatile unsigned char *)0xFFFFE6BE)
#define HCAN1_MD1_8         (*(volatile unsigned char *)0xFFFFE6BF)
#define HCAN1_MD2_1         (*(volatile unsigned char *)0xFFFFE6C0)
#define HCAN1_MD2_2         (*(volatile unsigned char *)0xFFFFE6C1)
#define HCAN1_MD2_3         (*(volatile unsigned char *)0xFFFFE6C2)
#define HCAN1_MD2_4         (*(volatile unsigned char *)0xFFFFE6C3)
#define HCAN1_MD2_5         (*(volatile unsigned char *)0xFFFFE6C4)
#define HCAN1_MD2_6         (*(volatile unsigned char *)0xFFFFE6C5)
#define HCAN1_MD2_7         (*(volatile unsigned char *)0xFFFFE6C6)
#define HCAN1_MD2_8         (*(volatile unsigned char *)0xFFFFE6C7)
#define HCAN1_MD3_1         (*(volatile unsigned char *)0xFFFFE6C8)
#define HCAN1_MD3_2         (*(volatile unsigned char *)0xFFFFE6C9)
#define HCAN1_MD3_3         (*(volatile unsigned char *)0xFFFFE6CA)
#define HCAN1_MD3_4         (*(volatile unsigned char *)0xFFFFE6CB)
#define HCAN1_MD3_5         (*(volatile unsigned char *)0xFFFFE6CC)
#define HCAN1_MD3_6         (*(volatile unsigned char *)0xFFFFE6CD)
#define HCAN1_MD3_7         (*(volatile unsigned char *)0xFFFFE6CE)
#define HCAN1_MD3_8         (*(volatile unsigned char *)0xFFFFE6CF)
#define HCAN1_MD4_1         (*(volatile unsigned char *)0xFFFFE6D0)
#define HCAN1_MD4_2         (*(volatile unsigned char *)0xFFFFE6D1)
#define HCAN1_MD4_3         (*(volatile unsigned char *)0xFFFFE6D2)
#define HCAN1_MD4_4         (*(volatile unsigned char *)0xFFFFE6D3)
#define HCAN1_MD4_5         (*(volatile unsigned char *)0xFFFFE6D4)
#define HCAN1_MD4_6         (*(volatile unsigned char *)0xFFFFE6D5)
#define HCAN1_MD4_7         (*(volatile unsigned char *)0xFFFFE6D6)
#define HCAN1_MD4_8         (*(volatile unsigned char *)0xFFFFE6D7)
```

```
#define HCAN1_MD5_1         (*(volatile unsigned char *)0xFFFFE6D8)
#define HCAN1_MD5_2         (*(volatile unsigned char *)0xFFFFE6D9)
#define HCAN1_MD5_3         (*(volatile unsigned char *)0xFFFFE6DA)
#define HCAN1_MD5_4         (*(volatile unsigned char *)0xFFFFE6DB)
#define HCAN1_MD5_5         (*(volatile unsigned char *)0xFFFFE6DC)
#define HCAN1_MD5_6         (*(volatile unsigned char *)0xFFFFE6DD)
#define HCAN1_MD5_7         (*(volatile unsigned char *)0xFFFFE6DE)
#define HCAN1_MD5_8         (*(volatile unsigned char *)0xFFFFE6DF)
#define HCAN1_MD6_1         (*(volatile unsigned char *)0xFFFFE6E0)
#define HCAN1_MD6_2         (*(volatile unsigned char *)0xFFFFE6E1)
#define HCAN1_MD6_3         (*(volatile unsigned char *)0xFFFFE6E2)
#define HCAN1_MD6_4         (*(volatile unsigned char *)0xFFFFE6E3)
#define HCAN1_MD6_5         (*(volatile unsigned char *)0xFFFFE6E4)
#define HCAN1_MD6_6         (*(volatile unsigned char *)0xFFFFE6E5)
#define HCAN1_MD6_7         (*(volatile unsigned char *)0xFFFFE6E6)
#define HCAN1_MD6_8         (*(volatile unsigned char *)0xFFFFE6E7)
#define HCAN1_MD7_1         (*(volatile unsigned char *)0xFFFFE6E8)
#define HCAN1_MD7_2         (*(volatile unsigned char *)0xFFFFE6E9)
#define HCAN1_MD7_3         (*(volatile unsigned char *)0xFFFFE6EA)
#define HCAN1_MD7_4         (*(volatile unsigned char *)0xFFFFE6EB)
#define HCAN1_MD7_5         (*(volatile unsigned char *)0xFFFFE6EC)
#define HCAN1_MD7_6         (*(volatile unsigned char *)0xFFFFE6ED)
#define HCAN1_MD7_7         (*(volatile unsigned char *)0xFFFFE6EE)
#define HCAN1_MD7_8         (*(volatile unsigned char *)0xFFFFE6EF)
#define HCAN1_MD8_1         (*(volatile unsigned char *)0xFFFFE6F0)
#define HCAN1_MD8_2         (*(volatile unsigned char *)0xFFFFE6F1)
#define HCAN1_MD8_3         (*(volatile unsigned char *)0xFFFFE6F2)
#define HCAN1_MD8_4         (*(volatile unsigned char *)0xFFFFE6F3)
#define HCAN1_MD8_5         (*(volatile unsigned char *)0xFFFFE6F4)
#define HCAN1_MD8_6         (*(volatile unsigned char *)0xFFFFE6F5)
#define HCAN1_MD8_7         (*(volatile unsigned char *)0xFFFFE6F6)
#define HCAN1_MD8_8         (*(volatile unsigned char *)0xFFFFE6F7)
#define HCAN1_MD9_1         (*(volatile unsigned char *)0xFFFFE6F8)
#define HCAN1_MD9_2         (*(volatile unsigned char *)0xFFFFE6F9)
#define HCAN1_MD9_3         (*(volatile unsigned char *)0xFFFFE6FA)
#define HCAN1_MD9_4         (*(volatile unsigned char *)0xFFFFE6FB)
#define HCAN1_MD9_5         (*(volatile unsigned char *)0xFFFFE6FC)
#define HCAN1_MD9_6         (*(volatile unsigned char *)0xFFFFE6FD)
#define HCAN1_MD9_7         (*(volatile unsigned char *)0xFFFFE6FE)
#define HCAN1_MD9_8         (*(volatile unsigned char *)0xFFFFE6FF)
#define HCAN1_MD10_1        (*(volatile unsigned char *)0xFFFFE700)
#define HCAN1_MD10_2        (*(volatile unsigned char *)0xFFFFE701)
#define HCAN1_MD10_3        (*(volatile unsigned char *)0xFFFFE702)
#define HCAN1_MD10_4        (*(volatile unsigned char *)0xFFFFE703)
#define HCAN1_MD10_5        (*(volatile unsigned char *)0xFFFFE704)
#define HCAN1_MD10_6        (*(volatile unsigned char *)0xFFFFE705)
#define HCAN1_MD10_7        (*(volatile unsigned char *)0xFFFFE706)
#define HCAN1_MD10_8        (*(volatile unsigned char *)0xFFFFE707)
```

```
#define HCAN1_MD11_1        (*(volatile unsigned char *)0xFFFFE708)
#define HCAN1_MD11_2        (*(volatile unsigned char *)0xFFFFE709)
#define HCAN1_MD11_3        (*(volatile unsigned char *)0xFFFFE70A)
#define HCAN1_MD11_4        (*(volatile unsigned char *)0xFFFFE70B)
#define HCAN1_MD11_5        (*(volatile unsigned char *)0xFFFFE70C)
#define HCAN1_MD11_6        (*(volatile unsigned char *)0xFFFFE70D)
#define HCAN1_MD11_7        (*(volatile unsigned char *)0xFFFFE70E)
#define HCAN1_MD11_8        (*(volatile unsigned char *)0xFFFFE70F)
#define HCAN1_MD12_1        (*(volatile unsigned char *)0xFFFFE710)
#define HCAN1_MD12_2        (*(volatile unsigned char *)0xFFFFE711)
#define HCAN1_MD12_3        (*(volatile unsigned char *)0xFFFFE712)
#define HCAN1_MD12_4        (*(volatile unsigned char *)0xFFFFE713)
#define HCAN1_MD12_5        (*(volatile unsigned char *)0xFFFFE714)
#define HCAN1_MD12_6        (*(volatile unsigned char *)0xFFFFE715)
#define HCAN1_MD12_7        (*(volatile unsigned char *)0xFFFFE716)
#define HCAN1_MD12_8        (*(volatile unsigned char *)0xFFFFE717)
#define HCAN1_MD13_1        (*(volatile unsigned char *)0xFFFFE718)
#define HCAN1_MD13_2        (*(volatile unsigned char *)0xFFFFE719)
#define HCAN1_MD13_3        (*(volatile unsigned char *)0xFFFFE71A)
#define HCAN1_MD13_4        (*(volatile unsigned char *)0xFFFFE71B)
#define HCAN1_MD13_5        (*(volatile unsigned char *)0xFFFFE71C)
#define HCAN1_MD13_6        (*(volatile unsigned char *)0xFFFFE71D)
#define HCAN1_MD13_7        (*(volatile unsigned char *)0xFFFFE71E)
#define HCAN1_MD13_8        (*(volatile unsigned char *)0xFFFFE71F)
#define HCAN1_MD14_1        (*(volatile unsigned char *)0xFFFFE720)
#define HCAN1_MD14_2        (*(volatile unsigned char *)0xFFFFE721)
#define HCAN1_MD14_3        (*(volatile unsigned char *)0xFFFFE722)
#define HCAN1_MD14_4        (*(volatile unsigned char *)0xFFFFE723)
#define HCAN1_MD14_5        (*(volatile unsigned char *)0xFFFFE724)
#define HCAN1_MD14_6        (*(volatile unsigned char *)0xFFFFE725)
#define HCAN1_MD14_7        (*(volatile unsigned char *)0xFFFFE726)
#define HCAN1_MD14_8        (*(volatile unsigned char *)0xFFFFE727)
#define HCAN1_MD15_1        (*(volatile unsigned char *)0xFFFFE728)
#define HCAN1_MD15_2        (*(volatile unsigned char *)0xFFFFE729)
#define HCAN1_MD15_3        (*(volatile unsigned char *)0xFFFFE72A)
#define HCAN1_MD15_4        (*(volatile unsigned char *)0xFFFFE72B)
#define HCAN1_MD15_5        (*(volatile unsigned char *)0xFFFFE72C)
#define HCAN1_MD15_6        (*(volatile unsigned char *)0xFFFFE72D)
#define HCAN1_MD15_7        (*(volatile unsigned char *)0xFFFFE72E)
#define HCAN1_MD15_8        (*(volatile unsigned char *)0xFFFFE72F)
```

```
/************************************************************************/
/*          WDT                                                         */
/************************************************************************/
#define WDT_TCSRW           (*(volatile unsigned short *)0xFFFFEC10)
#define WDT_TCSRR           (*(volatile unsigned char  *)0xFFFFEC10)
#define WDT_TCNTW           (*(volatile unsigned short *)0xFFFFEC10)
#define WDT_TCNTR           (*(volatile unsigned char  *)0xFFFFEC11)
#define WDT_RSTCSRW         (*(volatile unsigned short *)0xFFFFEC12)
#define WDT_RSTCSRR         (*(volatile unsigned char  *)0xFFFFEC13)
/************************************************************************/
/*          H-UDI                                                       */
/************************************************************************/
#define HUDI_SDIR           (*(volatile unsigned short *)0xFFFFF7C0)
#define HUDI_SDSR           (*(volatile unsigned short *)0xFFFFF7C2)
#define HUDI_SDDR           (*(volatile unsigned long  *)0xFFFFF7C4)
#define HUDI_SDDRH          (*(volatile unsigned short *)0xFFFFF7C4)
#define HUDI_SDDRL          (*(volatile unsigned short *)0xFFFFF7C6)
/************************************************************************/
/*          低消費電力状態                                              */
/************************************************************************/
#define SBYCR               (*(volatile unsigned char  *)0xFFFFEC14)
#define SYSCR               (*(volatile unsigned char  *)0xFFFFF708)
#define MSTCRW              (*(volatile unsigned short *)0xFFFFF70A)
#define MSTCRR              (*(volatile unsigned char  *)0xFFFFF70B)
```

A. 付録

目　次

RAMモニタモード	MCU	SH7055	使用機能	AUD

仕様

（1）図1に示すように、SH7055のAUDバスをSH7050のI/0ポートで制御します。
（2）SH7050システムをRAMモニタとして、SH7055内蔵RAMに4バイトデータのライトを行ないます。
（3）SH7050システムをRAMモニタとして、SH7055内蔵RAMから4バイトデータのリードを行ないます。

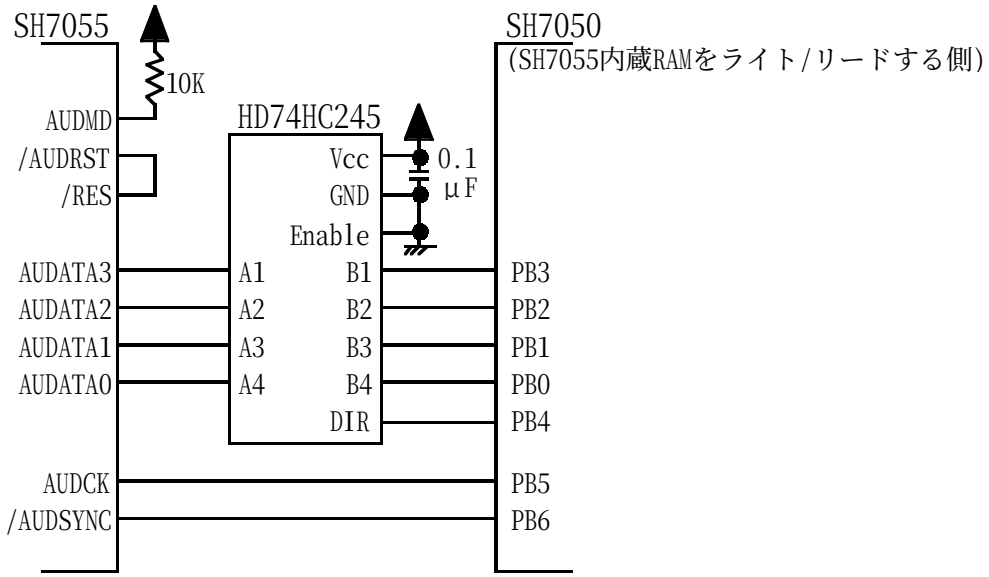


図1 インタフェース例

(注意) DIR=0(PB4出力Low) : AUDATAn ← PBn
DIR=1(PB4出力High): AUDATAn → PBn

考え方

（1）図2に本タスク例のAUDバス・タイミングを示します。

/AUDSYNCのアサート/ネゲートは、AUDCKの立ち下がりで設定することにより、セットアップ時間、ホールド時間を満足します。同様にAUDATAnの入出力は、AUDCKの立ち下がりで設定することにより、各セットアップ時間、ホールド時間を満足します。

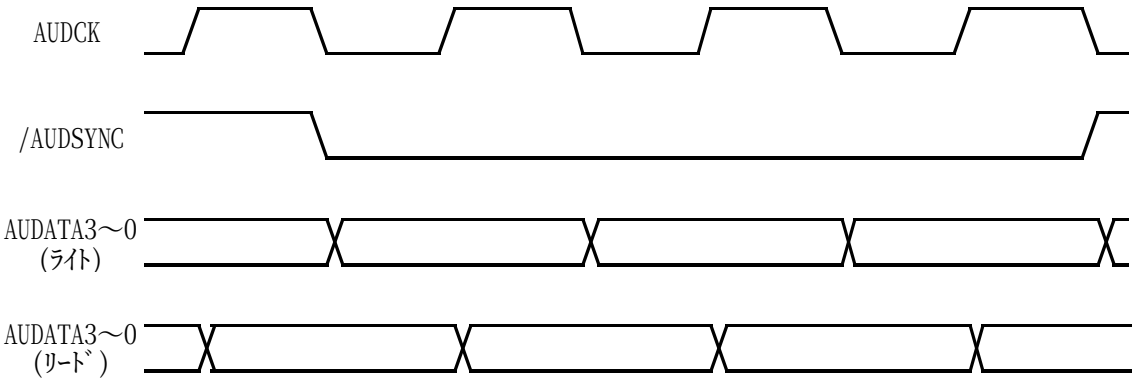


図2 AUDバス・タイミング

表１、表２に本タスク例の機能割り付け、及び設定値(入出力値)を示します。本タスク例は表１、表２にＳＨ７０５５に内蔵されているAUD及び低消費電力モードの機能を割り付けます。

表１ ＡＵＤ機能割付

ＡＵＤ内蔵機能		機能	入出力値
端子	AUDMD	AUDのモード選択入力。	Highレベル入力。 (RAMモニタモード選択)
	/AUDRST	AUDのリセット入力。	SH7055リセット端子と同期。
	AUDCK	AUDの同期クロック入力。	SH7050 PB5のソフトウェア 制御によるクロックを入力。
	/AUDSYNC	AUDのデータ先頭位置認識信号入力。	SH7050 PB6のソフトウェア 制御による信号を入力。
	AUDATA[3:0]	AUDのモニタアドレス／データ入出力。	SH7050 PB0～3の ソフトウェア制御による アドレス／データ入出力。

表２ 低消費電力モード機能割付

低消費電力 モードレジスタ	機能	設定値
SYSCR	AUDのリセット状態の設定をします。	0x01 （初期値）
MSTCR	AUDのクロック供給を設定します。	0x3C01（初期値）

ライト動作説明

図３にライト動作原理を示します。図３に示すようにＳＨ７０５０のI/Oポートにより、ＳＨ７０５５内蔵RAMに4バイトデータのライトを行ないます。

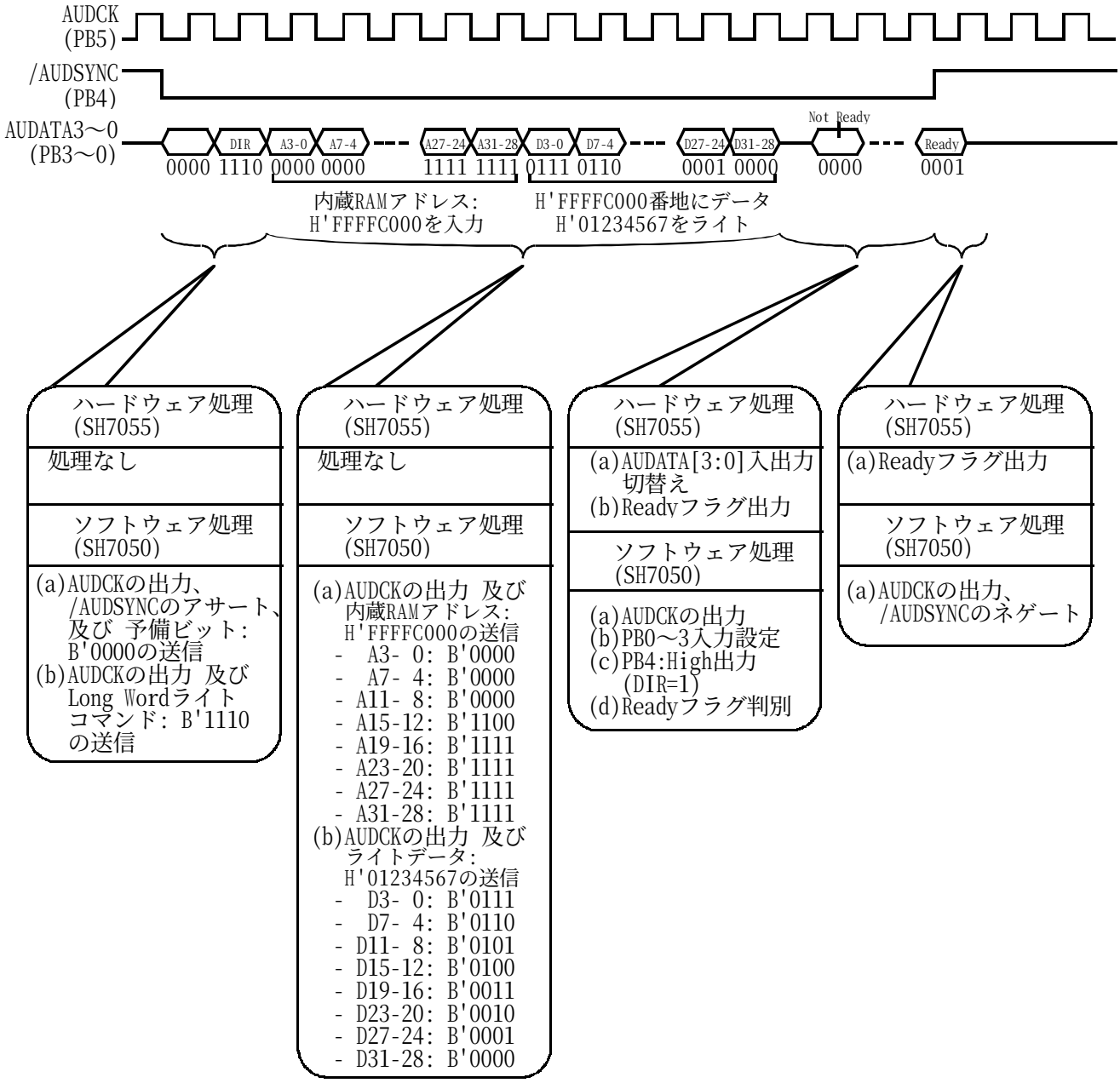


図３ AUDにおけるライト動作原理

リード動作説明

図４にリード動作原理を示します。図４に示すようにＳＨ７０５０のI/0ポートにより、ＳＨ７０５５内蔵RAMから4バイトデータのリードを行ないます。

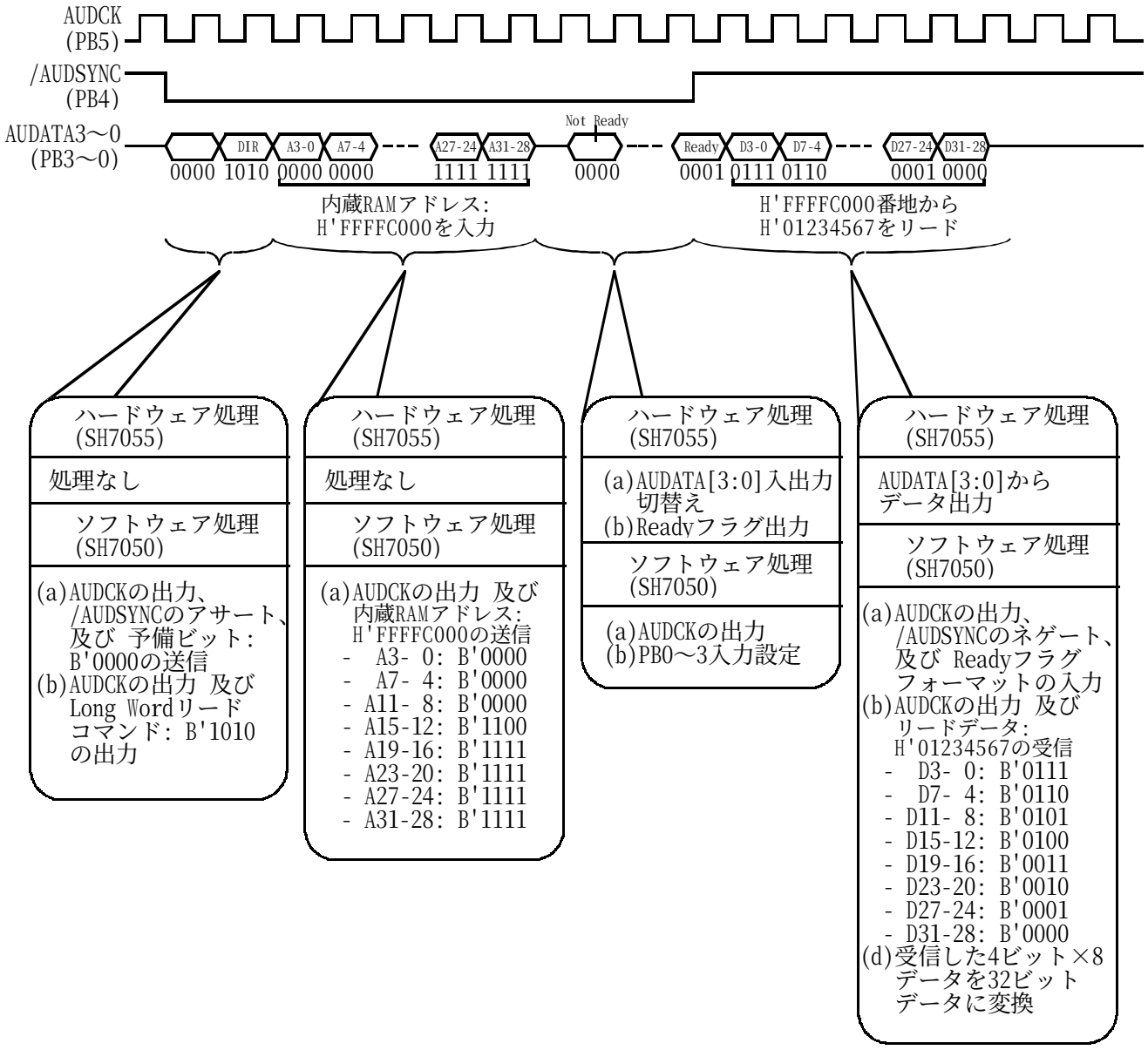


図４ AUDにおけるリード動作原理

図５、図６にユーザシステム側(本タスクではSH7050側)のデータライト制御、及びデータリード制御の一例をフローチャートで示します。

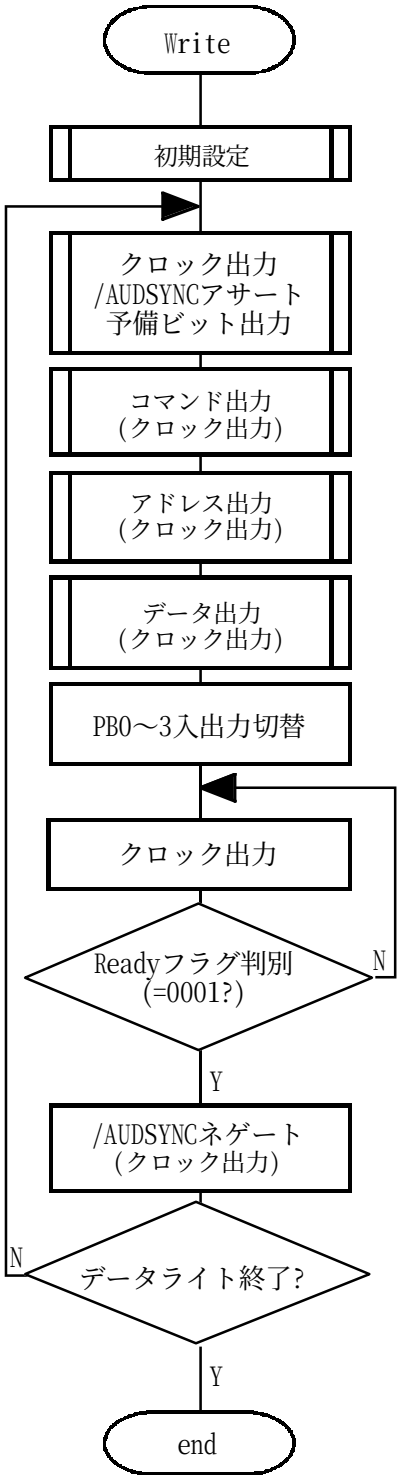


図５ データライト制御フローチャート

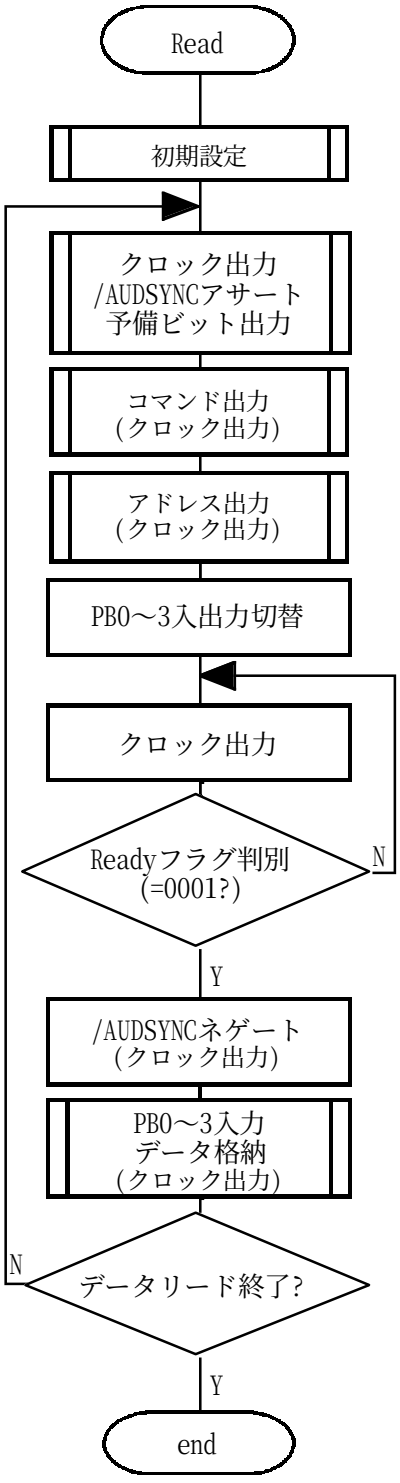
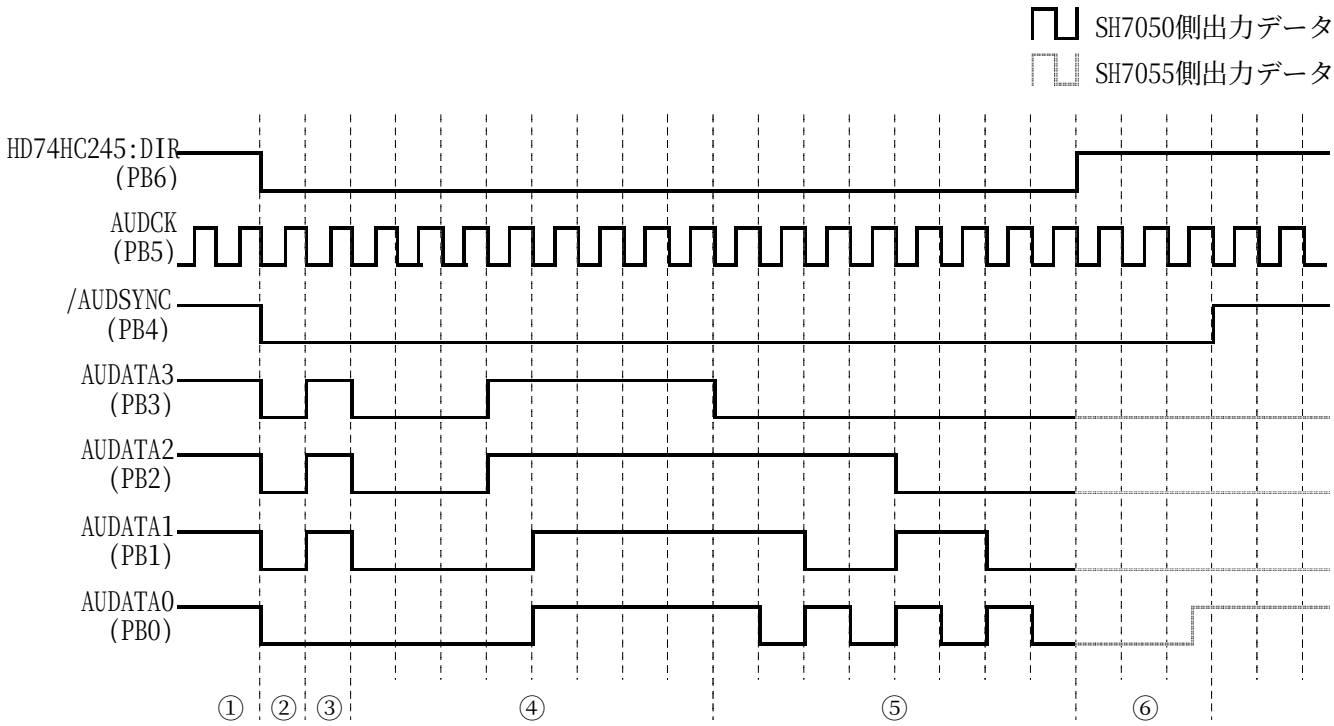


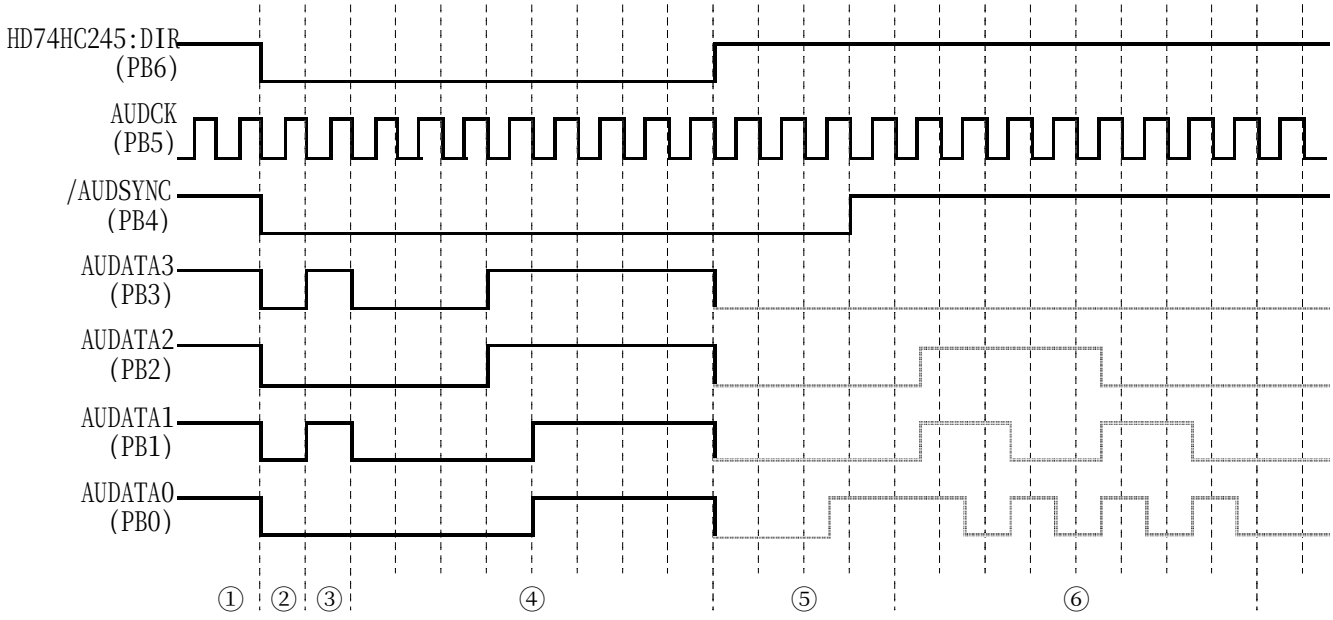
図６ データリード制御フローチャート

（2）図7、図8に本タスク例であるSH7050のI/0ポート及びSH7055のAUD内蔵機能による
タイミング波形を示します。



①：初期設定、クロック入力開始
②：予備ビット入力(B'0000)
③：DIRコマンド入力(B'1110)
④：アドレス入力(H'FFFFC000：A[3:0]→A[31:28])
⑤：データ入力(H'01234567：D[3:0]→D[31:28])
入出力切替え(HD74HC245:DIR端子レベル切替え)
⑥：Readyフラグ出力
(Ready=B'0001 → /AUDSYNCレベル切替え)

図7 データライトタイミング波形



①：初期設定、クロック入力開始
②：予備ビット入力(B'0000)
③：DIRコマンド入力(B'1010)
④：アドレス入力(H'FFFFC000：A[3:0]→A[31:28])
入出力切替え(HD74HC245:DIR端子レベル切替え)
⑤：Readyフラグ出力
(Ready=B'0001 → /AUDSYNCレベル切替え)
⑥：データ出力(H'01234567：D[3:0]→D[31:28])

図8 データリードタイミング波形

RAMによるフラッシュメモリのエミュレーション	MCU	SH7055	使用機能	ユーザプログラムモード

仕様

（1）ユーザプログラムモードでATU-Ⅱチャネル6を使用し、フラッシュメモリ上のデータ(ワードサイズ)をデューティーの変化率とした、PWMを出力します。
（2）シリアルコミュニケーションインタフェース(SCI)の受信割込みを使用し、エミュレーション機能によりフラッシュメモリ(H'2000～H'2FFF *注)とオーバーラップさせたRAM(H'FFFF6000～H'FFFF6FFF)上のデータをリアルタイムで変換します。
（3）変換データのフラッシュメモリ(H'2000～H'2FFF)への書込みを行います。
*注：フラッシュメモリのエミュレーションブロック(4kB)はRAMERにより選択します。

・フラッシュメモリ上のアプリケーションプログラム実行
・RAMERのRAMSビット、及びRAM0～2ビットにより、フラッシュメモリのEB2をエミュレーションブロックに設定
・SCI受信割込みによりオーバーラップRAM上のデータ変換を実行

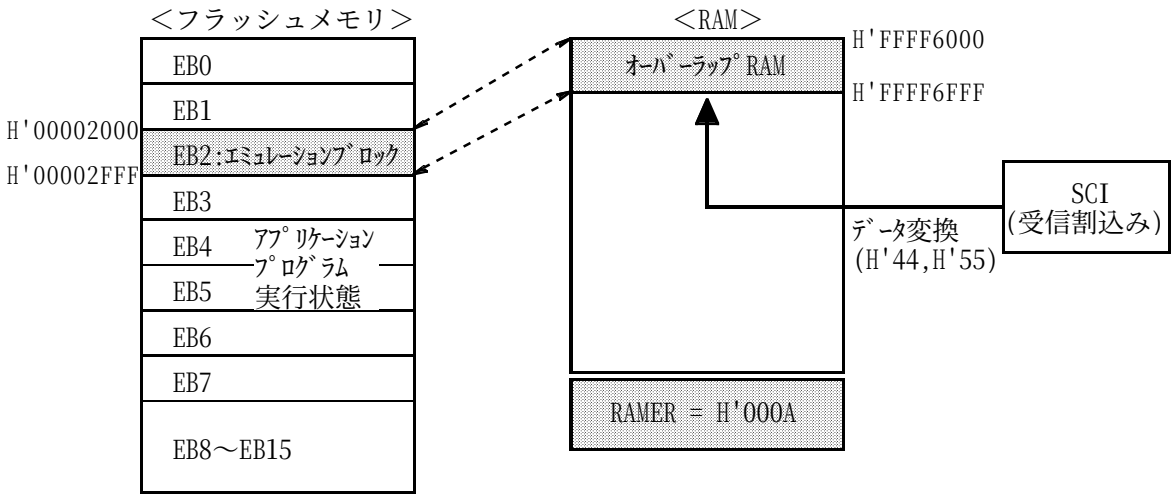


図1 RAMによるフラッシュメモリのエミュレーションブロック図

・RAM上のデータ確定後、SCI受信割込みによりRAMSビットをクリア
・フラッシュメモリ上のデータ変換プログラム(フラッシュメモリのエミュレーションブロックの消去、及びオーバーラップRAMのデータを書き込むプログラム)をRAMに転送し実行

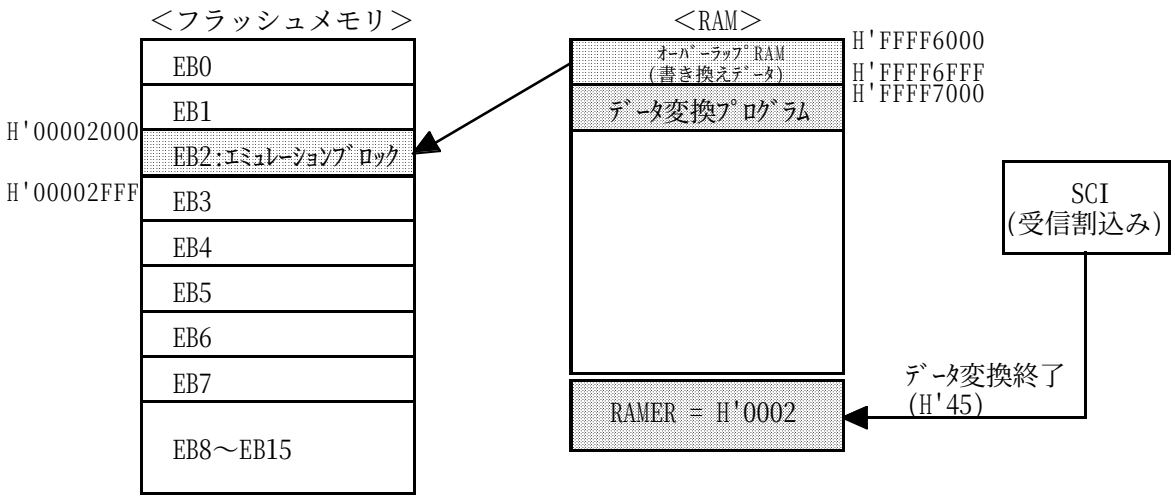
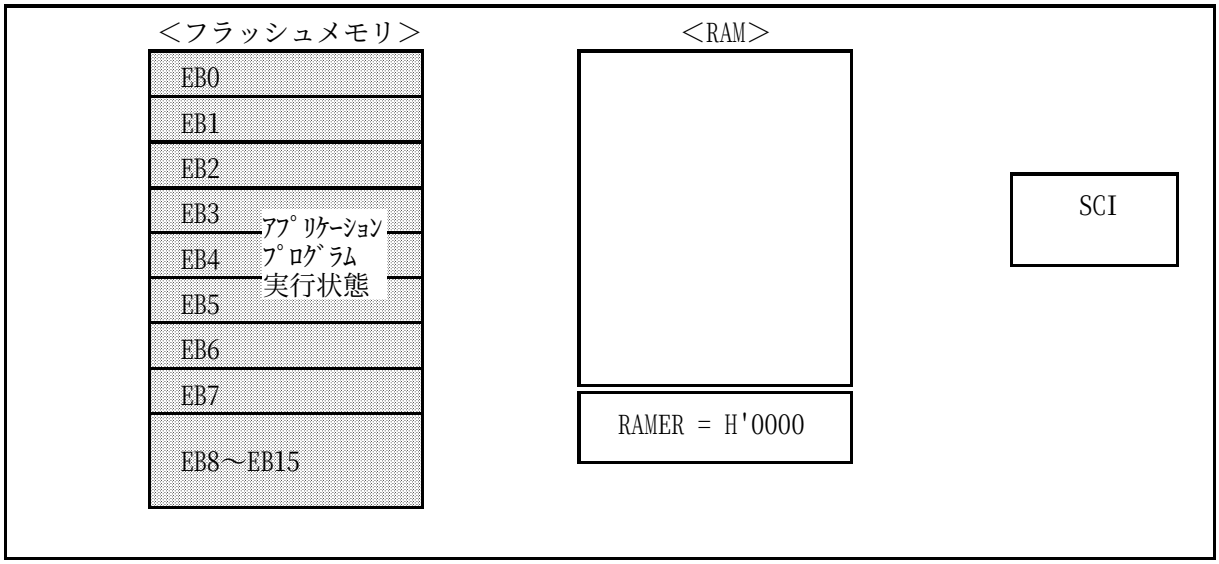


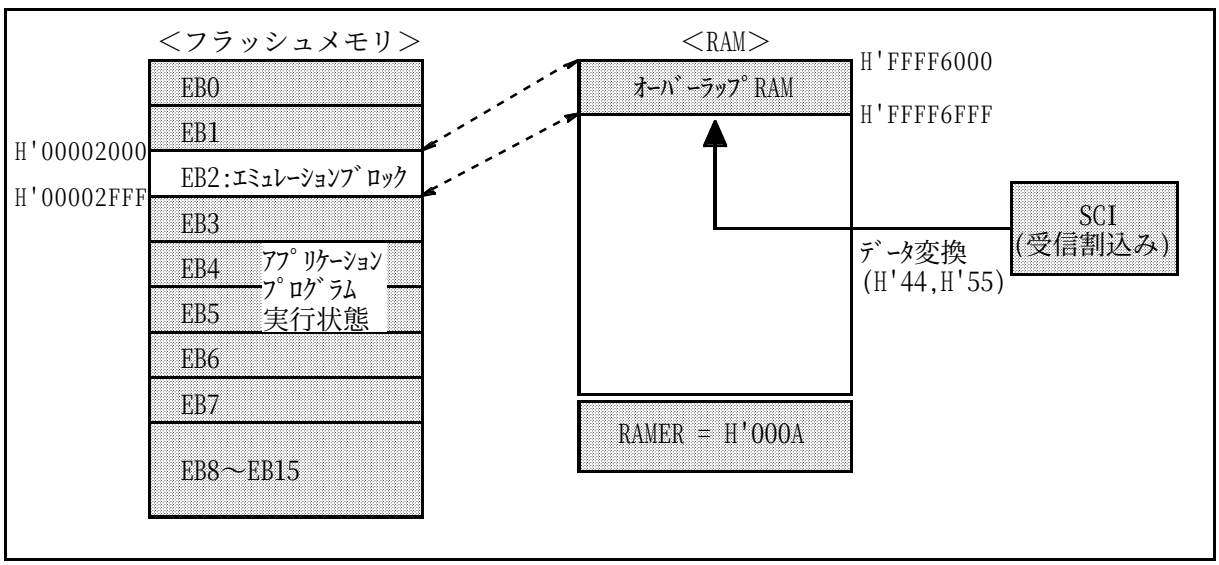
図2 エミュレーションデータのフラッシュ書込みブロック図

動作説明

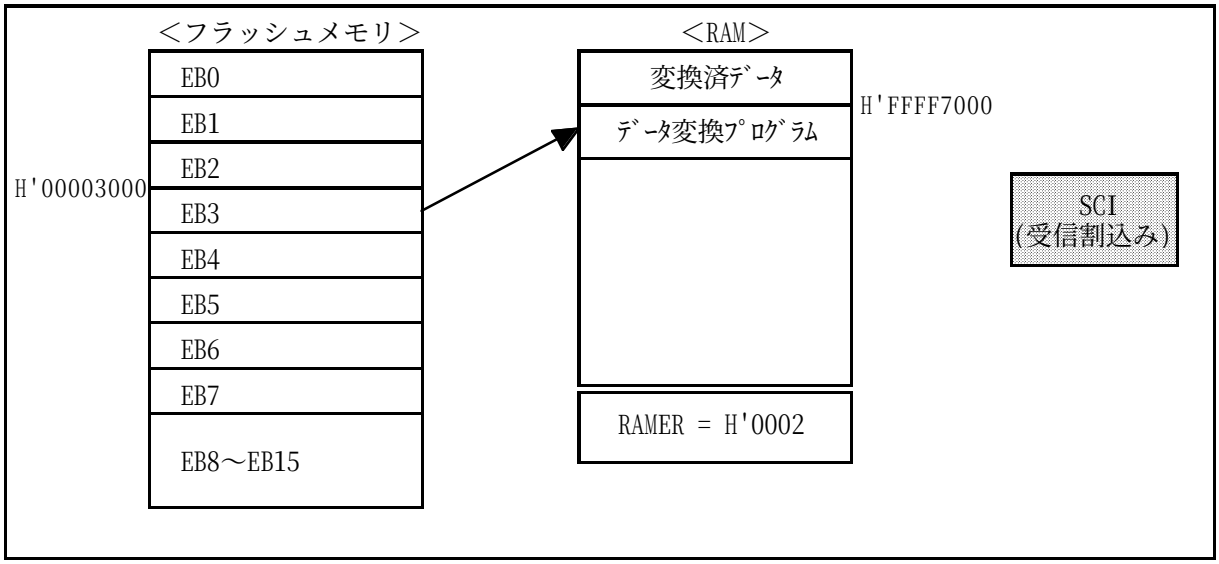
（1）フラッシュメモリ上のアプリケーションプログラムを実行。
この時T06A端子からは、H'00002000番地に格納されているデータ(DATA=H'0010)を
デューティーの変化率としたPWMを出力。



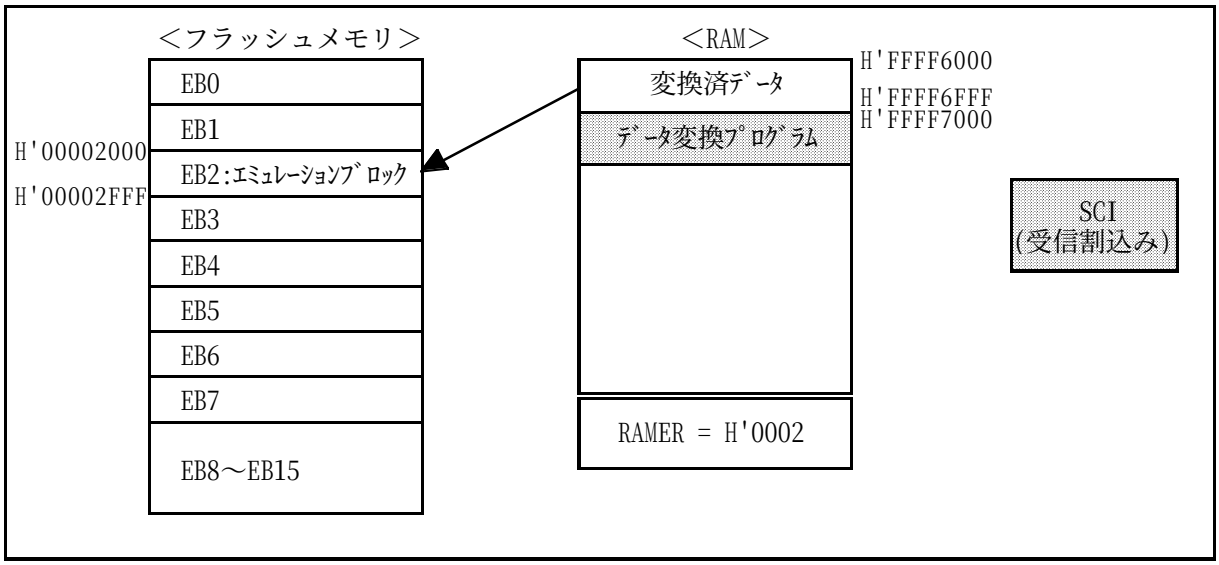
（2）SCI受信割込みによりRAMERを設定し、オーバーラップRAM上のデータを変換。
受信データ H'55：データインクリメント
H'44：データデクリメント
この時T06A端子からは、H'FFFF6000番地に格納されているデータ(EM_DATA←SCI受信割込みにより
変化)をデューティーの変化率としたPWMを出力。



（3）オーバーラップRAM上のデータ確定後、SCI受信割込みによりRAMSビット(RAMER)をクリアし、
データ変換プログラムをRAMに転送。



（4）データ変換プログラムを実行し、フラッシュメモリのエミュレーションブロックの消去、及び
RAM上の変換済データの書込みを行う。



・アプリケーションプログラム

（1）モジュール説明

モジュール名	ラベル名	機能
メイン	main	アプリケーションプログラムメインルーチン
セクションイニシャライズ	INITSCT	RAMの初期化
ポート設定	PORT_SET	I/Oポートの設定
SCI設定	SCI_SET	SCI0の設定
ATU-Ⅱ設定	ATU_SET	ATU-Ⅱチャネル6の設定
ROMtoRAMデータコピー	COPY_DATA	フラッシュメモリのエミュレーションブロックデータをオーバーラップRAMにコピー
SCI0受信割込み	RXI0	RAMERの設定、オーバーラップRAM上のデータ変換
PWMサイクル終了割込み	CMI6A	PWMデューティーの設定
データ書き換え準備	FLASH_WRITE	データ変換プログラムをRAMに転送

（2）使用内部レジスタの説明

レジスタ名	機能	使用ラベル名
PAIOR	ポートA入出力設定。	PORT_SET
PACRH	TxD0,RxD0機能選択。	PORT_SET
PBIOR	ポートB入出力設定。	PORT_SET
PBCRL	TO6A機能選択。	PORT_SET
SCR	送受信動作、及び割込み許可/禁止設定。	SCI_SET
SMR	調歩同期式,データ長8ビット,ノンパリティに設定。	SCI_SET
BRR	転送レートを9600bpsに設定。	SCI_SET
RDR	受信データ参照用に使用。	RXI0
SSR	ステータスフラグ処理。	RXI0
PSCR2	ATU-Ⅱチャネル6のプリスケーラ1段目設定。	ATU_SET
TCR6A	TCNT6Aのプリスケーラ(2段目)設定。	ATU_SET
PMDR	オンデューティー,相補PWMに設定。	ATU_SET
CYLR6A	PWMのサイクルを設定。	ATU_SET
BFR6A	PWMのデューティーバッファ。サイクル終了時にDTR6Aに転送。サイクル終了割込み時にデータにより変化。	ATU_SET CMI6A
TIER6	CMF6A(サイクル終了フラグ)による割込み許可。	CMI6A
DTR6A	PWM6Aデューティー値。	CMI6A
TSR6	ステータスフラグ処理。	CMI6A
IPRG	ATU6割込み優先レベル設定。	main
IPRK	SCI0割込み優先レベル設定。	main
TSTR2	TCNT6A動作設定。	main
RAMER	エミュレーション及びブロック選択。	RXI0
FLMCR1	FWE端子状態参照用に使用。(ユーザプログラムモード)	FLASH_WRITE

```
/************************************************************************************/
/*                      RAMｴﾐｭﾚｰｼｮﾝﾓｰﾄﾞ                                             */
/************************************************************************************/
#include <stdio.h>
#include <machine.h>
#include "7055.h"
#include "F_WRITE.h"
/************************************************************************************/
/*                      関数ﾌﾟﾛﾄﾀｲﾌﾟ宣言                                             */
/************************************************************************************/
void main(void);
void INITSCT(void);
void PORT_SET(void);
void SCI_SET(void);
void ATU_SET(void);
void FLASH_WRITE(void);
void COPY_DATA(void);
/************************************************************************************/
/*                      変数定義                                                    */
/************************************************************************************/
#define EM_DATA       (*(volatile unsigned short *)0xFFFF6000)
const unsigned short DATA = 0x0010;
unsigned char  WORK;
extern long *_B_BGN,*_B_END,*_D_BGN,*_D_END,*_D_ROM;
/************************************************************************************/
/*                      ﾒｲﾝﾙｰﾁﾝ                                                     */
/************************************************************************************/
void main(void){
        INITSCT();
        PORT_SET();                     /* I/Oﾎﾟｰﾄｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ                  */
        SCI_SET();                      /* SCI0ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ                      */
        ATU_SET();                      /* ATU-Ⅱ ch6ｲﾆｼｬﾗｲｽﾞﾙｰﾁﾝ                 */
        WORK = 0x00;                    /* ｿﾌﾄｳｪｱﾌﾗｸﾞ初期化                      */
        COPY_DATA();                    /* ｴﾐｭﾚｰｼｮﾝRAM領域初期化                 */
        INTC.IPRG      = 0x00A0;        /* ATU6割込みﾚﾍﾞﾙ設定                    */
        INTC.IPRK      = 0xF000;        /* SCI0割込みﾚﾍﾞﾙ設定                    */
        ATUC.TSTR2 = 0x01;              /* ﾀｲﾏｽﾀｰﾄ                              */
        set_imask(0x0);
        while(1);
}
void INITSCT(void){
        register long *p,;
        for(p=_B_BGN; p<_B_END; p++)
                *p=0;
}
```

```
void PORT_SET(void){
        PA.PAIOR        = 0x4000;                       /* PA15を入力,PA14を出力端子に設定   */
        PA.PACRH        = 0x5000;                       /* PA15,14をRxD0,TxD0機能に設定      */
        PB.PBIOR        = 0x0001;                       /* PB0を出力端子に設定               */
        PB.PBCRL        = 0x0001;                       /* PB0をTO6A機能に設定               */
}
void SCI_SET(void){
        signed int lp;
    SCI0.SCR = 0x00;                                    /* 送信・受信動作を禁止              */
    SCI0.SMR = 0x00;                                    /* 調歩同期式､8ﾋﾞｯﾄﾃﾞｰﾀ、ﾊﾟﾘﾃｨなし  */
    SCI0.BRR = 0x40;                                    /* ﾋﾞｯﾄﾚｰﾄ9600bps                    */
    for( lp = 1; lp < 1; lp++ );                        /* ｳｪｲﾄ                              */
    SCI0.SCR = 0x70;                                    /* 送信・受信動作・割込みを許可する。*/
}
void ATU_SET(void){
        ATUC.PSCR2      = 0x0004;                       /* ﾌｧｰｽﾄﾌﾟﾘｽｹｰﾗφ'=Pφ/5(250ns)       */
        ATU6.TCR6A      = 0x04;             /* ｾｶﾝﾄﾞﾌﾟﾘｽｹｰﾗφ"=φ'/16(4μs)        */
        ATU6.PMDR       = 0x01;             /* ｵﾝﾃﾞｭｰﾃｨｰ,相補PWM                 */
        ATU6.CYLR6A     = 0x0400;                       /* 周期8.192ms                       */
        ATU6.BFR6A      = 0x0200;                       /* ﾃﾞｭｰﾃｨｰ50%(初期値)                */
        ATU6.TIER6      = 0x0001;                       /* ｻｲｸﾙ終了割込み許可                */
}
void COPY_DATA(void){
        unsigned long *changeprg;
        unsigned long *InRAMaddress;
        changeprg = (unsigned long *)0x00002000;    /* ﾌﾗｯｼｭﾒﾓﾘｴﾐｭﾚｰｼｮﾝﾌﾞﾛｯｸ先頭ｱﾄﾞﾚｽ    */
        InRAMaddress = (unsigned long *)0xFFFF6000; /* 転送先内蔵RAM先頭ｱﾄﾞﾚｽ      */
        while(changeprg < (unsigned long *)0x00003000){     /* 転送終了ｱﾄﾞﾚｽ             */
                *InRAMaddress = *changeprg;                 /* 転送                 */
                changeprg ++;                               /* ｱﾄﾞﾚｽｲﾝｸﾘﾒﾝﾄ             */
                )InRAMaddress ++;                           /* ｱﾄﾞﾚｽｲﾝｸﾘﾒﾝﾄ             */
        }
}
```

```
/*****************************************************************************/
/*                        割込みﾙｰﾁﾝ                                         */
/*****************************************************************************/
#pragma interrupt(RXI0,CMI6A)
void RXI0(void){                                    /* SCI0受信割込みﾙｰﾁﾝ          */
        if((WORK & 0x0F) == 0x0F){
                switch(SCI0.RDR){                   /* 受信ﾃﾞｰﾀ判別               */
                        case 0x44:                  /* 0x44('D')受信              */
                                if(EM_DATA >= 0x01){
                                        EM_DATA -= 1; /* ﾃﾞｭｰﾃｨｰ変化率DOWN        */
                                }
                                break;
                        case 0x55:                  /* 0x55('U')受信              */
                                if(EM_DATA <= 0x4E){
                                        EM_DATA += 1; /* ﾃﾞｭｰﾃｨｰ変化率UP          */
                                }
                                break;
                        case 0x45:                  /* 0x45('E')受信              */
                                BSC.RAMER &= 0xFFF7; /* RAMSﾋﾞｯﾄｸﾘｱ               */
                                FLASH_WRITE();      /* ﾃﾞｰﾀ変換ﾌﾟﾛｸﾞﾗﾑを転送し実行  */
                                break;
                }
        }else{
                WORK = 0x0F;
                BSC.RAMER = 0x000A;          /* ｴﾐｭﾚｰｼｮﾝ選択(EB2:H'2000～H'2FFF)  */
        }
        SCI0.SSR &= 0xBF;
}
void CMI6A(void){                                   /* PWMｻｲｸﾙ終了割込み          */
        if((WORK & 0xF0) == 0x00){                  /* ﾃﾞｭｰﾃｨｰ増加               */
                ATU6.BFR6A += DATA;
                if(ATU6.DTR6A >= 0x03B0){
                        WORK |= 0xF0;
                }
        }else{                                      /* ﾃﾞｭｰﾃｨｰ減少               */
                ATU6.BFR6A -= DATA;
                if(ATU6.DTR6A <= 0x0050){
                        WORK &= 0x0F;
                }
        }
        ATU6.TSR6 &= 0xFFFE;
}
void FLASH_WRITE(void){
    extern void RAM_main(void);
    extern char *CopyTopAddress, *CopyEndAddress;
    char *x, *y;
    while((FLASH.FLMCR1 & 0x80) != 0x80){
        ;                                           /* FWEﾋﾞｯﾄ検証                */
    }
    for( x=CopyTopAddress, y=(char *)RAM_TOP; x<CopyEndAddress; x++, y++ ){
        *y = *x;                                    /* ﾃﾞｰﾀ変換ﾌﾟﾛｸﾞﾗﾑを内蔵RAMに転送 */
    }
    RAM_main();                                     /* ﾃﾞｰﾀ変換ﾌﾟﾛｸﾞﾗﾑ実行        */
}
```

・データ変換プログラム
　本プログラムにおけるフラッシュメモリの消去/書込みは、暫定的な数値及び仕様により作成されております。本プログラムをご使用される場合は、推奨されるアルゴリズムに沿って書き換える必要があります。

（1）モジュール説明

モジュール名	ラベル名	機能
RAMエリアメイン	RAM_main	データ変換プログラムメインルーチン
消去メイン	F_ERASE	フラッシュメモリ消去メイン
消去ベリファイ	ERASEVF	フラッシュメモリ消去ベリファイ
消去	ERASE	フラッシュメモリブロック消去
書込みメイン	F_WRITE	フラッシュメモリ書込みメイン
書込みベリファイ	PROGRAMVF	フラッシュメモリ書込みベリファイ
書込み	PROGRAM	フラッシュメモリ4kbyte書込み
ウェイト	WAIT	1μs～10msウェイト
書込みウェイト	P_WAIT	書込み50μsまたは200μsウェイト
消去ウォッチドッグ設定	SET_WDT_E	消去時のウォッチドッグタイマ設定
書込みウォッチドッグ設定	SET_WDT_P	書込み時のウォッチドッグタイマ設定
ウォッチドッグ解除	RESET_WDT	ウォッチドッグタイマの解除
ROMtoRAMコピー	ROMtoRAMcopyADDRESS	ROMからRAMへコピーする範囲を指定

（2）使用内部レジスタの説明

レジスタ名	機能	使用ラベル名
RAMER	エミュレーションブロック選択値として参照用に使用。	RAM_main
FLMCR1	消去、書込み時に各ビットを設定。 (FWEはFWE端子の状態により決定)	RAM_main, ERASEVF,ERASE, PROGRAMVF,PROGRAM
EBR1	消去ブロックを設定。(1ブロックのみ設定可能)	ERASE
RSTCSR	ウォッチドッグタイマのオーバーフローにより 内部パワーオンリセットに設定。	SET_WDT_E SET_WDT_P
TCNT	ウォッチドッグタイマカウンタ設定。	SET_WDT_E SET_WDT_P
TCSR	ウォッチドッグタイマのカウントクロック、及び 動作開始/停止設定。	SET_WDT_E SET_WDT_P RESET_WDT

```
/*****************************************************************************/
/*            ﾃﾞｰﾀ変換ﾌﾟﾛｸﾞﾗﾑ                                                 */
/*****************************************************************************/
#include "7055.h"
#include "F_WRITE.h"
/*****************************************************************************/
/*    内部関数の宣言                                                         */
/*****************************************************************************/
void RAM_main(void);
static int F_ERASE(int blk_no);
static int ERASEVF(long *erase_top , long *erase_btm);
static void ERASE(unsigned char e_bit);
static int F_WRITE(long *write_data , long *write_adr);
static void PROGRAM(int wtime, char *src ,char *dst);
static int PROGRAMVF(long *src , long *dst , long *rewrite);
static void WAIT(unsigned long t_counter);
static void P_WAIT(unsigned short Wtime);
static void SET_WDT_E();
static void SET_WDT_P();
static void RESET_WDT();

#pragma section _ROMtoRAM
/*****************************************************************************/
/*            ﾃﾞｰﾀの定義                                                     */
/*****************************************************************************/
struct DAT_BUF {
    long *w_adr;
    long *w_data;
};
static const char erase_bit[] = {
    BLK0,
    BLK1,
    BLK2,
    BLK3,
    BLK4,
    BLK5,
    BLK6,
    BLK7
};
typedef long *ShortPtr;
static const   ShortPtr erase_block[] = {
    (long *)BLK0TOP,
    (long *)BLK1TOP,
    (long *)BLK2TOP,
    (long *)BLK3TOP,
    (long *)BLK4TOP,
    (long *)BLK5TOP,
    (long *)BLK6TOP,
    (long *)BLK7TOP,
    (long *)BLK7END
};
```

```
/*****************************************************************************/
/*            RAM_main : RAMｴﾘｱﾒｲﾝﾙｰﾁﾝ                                       */
/*****************************************************************************/
void RAM_main(void) {
    int i, res;
    struct DAT_BUF dat_buf;
       i = BSC.RAMER;                       /* ﾌﾞﾛｯｸNo.(選択済)                */
    FLASH.FLMCR1 |= 0x40;                   /* SWE1ﾋﾞｯﾄｾｯﾄ : 書込み開始        */
    WAIT(0x0000000A);                       /* 1μs以上 : ｳｪｲﾄﾙｰﾁﾝ             */
    res = F_ERASE(i);
       dat_buf.w_adr = erase_block[i];      /* ｴﾐｭﾚｰｼｮﾝﾌﾞﾛｯｸ先頭ｱﾄﾞﾚｽ          */
       dat_buf.w_data = (long *)I_RAMadd;   /* ｴﾐｭﾚｰｼｮﾝ用内蔵RAM先頭ｱﾄﾞﾚｽ      */
    if(res == OK) {                         /* 消去OKなら書込み処理を実行      */
              while( dat_buf.w_adr < erase_block[i+1] ){
              res = F_WRITE( dat_buf.w_data , dat_buf.w_adr);
              if(res != OK) {
              break;                        /* 書込みｴﾗｰ                      */
              }
                     for(i=0;i<32;i++){     /* 128ﾊﾞｲﾄｱﾄﾞﾚｽｲﾝｸﾘﾒﾝﾄ            */
                            dat_buf.w_data++;
                            dat_buf.w_adr++;
                     }
              }
    }
       else{
       }
    FLASH.FLMCR1 &= 0xBF;                   /* SWE1ﾋﾞｯﾄｸﾘｱ : 書込み終了        */
    for(;;);                                /* 自己ﾙｰﾌﾟ                        */
}
```

```
/*****************************************************************************/
/*            F_ERASE：ﾌﾗｯｼｭﾒﾓﾘ消去ﾙｰﾁﾝ                                      */
/*****************************************************************************/
static int F_ERASE(int blk_no) {
    int erase_time,res;
    erase_time = 0;                         /* 消去回数初期化                  */
    res = OK;
    while((res = ERASEVF(erase_block[blk_no],erase_block[blk_no+1])) != OK) {
        ERASE(erase_bit[blk_no]);
        erase_time++;                       /* 消去回数ｲﾝｸﾘﾒﾝﾄ                 */
        if(erase_time >= MAX_erase_time) {
            res = OT;                       /* 消去ｴﾗｰ(100回以上)              */
            break;
        }
    }
    return res;
}
/*****************************************************************************/
/*            ERASEVF：消去ﾍﾞﾘﾌｧｲﾙｰﾁﾝ                                       */
/*****************************************************************************/
static int ERASEVF(long *erase_top , long *erase_btm) {
    long *erase_adr;
    int res;
    res = OK;
    FLASH.FLMCR1 |= 0x08;                   /* EV1ﾋﾞｯﾄｾｯﾄ                      */
    WAIT(0x0000003C);                       /* 6μs以上 : ｳｪｲﾄﾙｰﾁﾝ              */
    erase_adr = erase_top;
    while(erase_adr < erase_btm) {
        *erase_adr = 0xFFFFFFFF;            /* ｱﾄﾞﾚｽﾗｯﾁのためのﾀﾞﾐｰﾗｲﾄ(H'FFFFFFFF)  */
        WAIT(0x00000014);                   /* 2μs以上 : ｳｪｲﾄﾙｰﾁﾝ              */
        if(*erase_adr != 0xFFFFFFFF) {
            res = NG;                       /* 消去ｴﾗｰ                         */
            break;
        }
        erase_adr++;
    }
    FLASH.FLMCR1 &= 0xF7;                   /* EV1ﾋﾞｯﾄｸﾘｱ                      */
    WAIT(0x00000028);                       /* 4μs以上 : ｳｪｲﾄﾙｰﾁﾝ              */
    return res;
}
/*****************************************************************************/
/*            ERASE : 消去ﾙｰﾁﾝ                                               */
/*****************************************************************************/
static void ERASE(unsigned char e_bit) {
    FLASH.EBR1 |= e_bit;                    /* EBR1にｲﾚｰｽﾌﾞﾛｯｸを設定           */
    SET_WDT_E();                            /* ｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏｾｯﾄ                  */
    FLASH.FLMCR1 |= 0x20;                   /* ESU1ﾋﾞｯﾄｾｯﾄ                     */
    WAIT(0x000003E8);                       /* 100μs以上 : ｳｪｲﾄﾙｰﾁﾝ            */
    FLASH.FLMCR1 |= 0x02;                   /* E1ﾋﾞｯﾄｾｯﾄ : 消去開始            */
    WAIT(0x00018000);                       /* 10msec未満 : ｳｪｲﾄﾙｰﾁﾝ           */
    FLASH.FLMCR1 &= 0xFD;                   /* E1ﾋﾞｯﾄｸﾘｱ : 消去停止            */
    WAIT(0x00000064);                       /* 10μs以上 : ｳｪｲﾄﾙｰﾁﾝ            */
    FLASH.FLMCR1 &= 0xDF;                   /* ESU1ﾋﾞｯﾄｸﾘｱ                     */
    WAIT(0x00000064);                       /* 10μs以上 : ｳｪｲﾄﾙｰﾁﾝ            */
    RESET_WDT();                            /* ｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏ停止                */
}
```

```
/*****************************************************************************/
/*            F_WRITE : ﾌﾗｯｼｭﾒﾓﾘ書込みﾙｰﾁﾝ                                    */
/*****************************************************************************/
static int F_WRITE(long *write_data , long *write_adr) {
    long rewrite_buff[32];                  /* 再書込みﾃﾞｰﾀﾊﾞｯﾌｧ                 */
    long rewrite_buff1[32];                 /* 1回前の書込みﾃﾞｰﾀﾊﾞｯﾌｧ             */
    int i,res;
    long *src,program_time;
    src = write_data;                       /* ｴﾐｭﾚｰｼｮﾝﾃﾞｰﾀ(内蔵RAM)ｱﾄﾞﾚｽ          */
    for(i=0; i<32; i++){                    /* 128ﾊﾞｲﾄ転送                       */
        rewrite_buff[i] = *src;             /* 書込みﾃﾞｰﾀを再書込みﾃﾞｰﾀﾊﾞｯﾌｧに転送 */
        rewrite_buff1[i] = *src++;  /* 書込みﾃﾞｰﾀを1回前の書込みﾃﾞｰﾀﾊﾞｯﾌｧに転送   */
    }
    program_time = 0;                       /* 書込み回数初期化                 */
    while((res = PROGRAMVF(write_data , write_adr , rewrite_buff)) != OK){
        if(res == CONT){                    /* 書込み中                         */
                    if( program_time >= 1){
                            PROGRAM( program_time, (char *)rewrite_buff1 ,(char *)write_adr);
                            for(i=0; i<32; i++){
                                    rewrite_buff1[i] = rewrite_buff[i];
                            }       /* 再書込みﾃﾞｰﾀを1回前の書込みﾃﾞｰﾀﾊﾞｯﾌｧに転送  */
                    }
            program_time++;                 /* 書込み回数ｲﾝｸﾘﾒﾝﾄ                */
            PROGRAM( program_time, (char *)rewrite_buff ,(char *)write_adr);
            if(program_time >= MAX_program_time){
                res = NG;                   /* 書込みｴﾗｰ(1000回以上)            */
                break;
            }
        }else{
            res = NG;                       /* 書込みｴﾗｰ                        */
            break;
        }
    }
    return res;
}
```

```
/*****************************************************************************/
/*            PROGRAMVF : 書込みﾍﾞﾘﾌｧｲﾙｰﾁﾝ                                   */
/*****************************************************************************/
static int PROGRAMVF(long *src , long *dst , long *rewrite){
    long *rw_ptr,work;
    int res,i;
    FLASH.FLMCR1 |= 1;                      /* PV1ﾋﾞｯﾄｾｯﾄ                       */
    WAIT(0x00000028);                       /* 4μs以上 : ｳｪｲﾄﾙｰﾁﾝ              */
    res = OK;
    rw_ptr = rewrite;                       /* 再書込みﾃﾞｰﾀﾊﾞｯﾌｧの先頭ｱﾄﾞﾚｽをｺﾋﾟｰ  */
    for(i=0; i<32; i++) {
        *dst = 0xFFFFFFFF;                  /* ｱﾄﾞﾚｽﾗｯﾁのためのﾀﾞﾐｰﾗｲﾄ(H'FFFFFFFF)  */
        WAIT(0x00000014);                   /* 2μs : ｳｪｲﾄﾙｰﾁﾝ                  */
        work = ~*dst++;                     /* 再書込みﾃﾞｰﾀ演算                 */
        *rw_ptr++ = (*src | work);          /* 再書込みﾃﾞｰﾀを再書込みﾃﾞｰﾀﾊﾞｯﾌｧに格納 */
        if((work & *src++) != 0){
            res = NG;                       /* 書込みｴﾗｰ(初期値異常)検出        */
            break;
        }
    }
    FLASH.FLMCR1 &= 0xFB;                   /* PV1ﾋﾞｯﾄｸﾘｱ                       */
    WAIT(0x00000014);                       /* 2μs以上 : ｳｪｲﾄﾙｰﾁﾝ              */
    if(res != NG) {
        for(i=0; i<32; i++) {
            if(*rewrite++ != 0xFFFFFFFF) {  /* 再書込みﾃﾞｰﾀ検証                 */
                res = CONT;                 /* 書込み不完全                     */
                break;
            }
        }
    }
    return res;
}
/*****************************************************************************/
/*            PROGRAM : 書込みﾙｰﾁﾝ                                           */
/*****************************************************************************/
static void PROGRAM( int Wtime, char *src ,char *dst){
    int i;
    for(i=0; i<128; i++) {                  /* 書込みデータを転送               */
        *dst++ = *src++;                    /* 転送はバイト単位                 */
    }
    SET_WDT_P();                            /* ｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏｾｯﾄ                  */
    FLASH.FLMCR1 |= 0x10;                   /* PSU1ﾋﾞｯﾄｾｯﾄ                      */
    WAIT(0x000001F4);                       /* 50μs以上 : ｳｪｲﾄﾙｰﾁﾝ             */
    FLASH.FLMCR1 |= 0x01;                   /* P1ﾋﾞｯﾄｾｯﾄ                        */
    P_WAIT(Wtime);                          /* 50 or 200μs未満 : ｳｪｲﾄﾙｰﾁﾝ       */
    FLASH.FLMCR1 &= 0xFE;                   /* P1ﾋﾞｯﾄｸﾘｱ                        */
    WAIT(0x00000032);                       /* 5μs以上 : ｳｪｲﾄﾙｰﾁﾝ              */
    FLASH.FLMCR1 &= 0xEF;                   /* PSU1ﾋﾞｯﾄｸﾘｱ                      */
    WAIT(0x00000032);                       /* 5μs以上 : ｳｪｲﾄﾙｰﾁﾝ              */
    RESET_WDT();                            /* ｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏ停止                 */
}
```

```
/*****************************************************************************/
/*            WAIT：ｳｪｲﾄﾙｰﾁﾝ(SH7055F, 40MHz動作の場合)                          */
/*****************************************************************************/
static void WAIT(unsigned long t_counter){
    int i;
    for(i=0; i < t_counter; i++);
}
static void P_WAIT(unsigned short Wtime){  /* 50 or 200μs未満 : ｳｪｲﾄﾙｰﾁﾝ        */
    int j;
    if( Wtime < 4 ){
        for(j=0; j < 0x01E0; j++);         /* 50μs未満                        */
    }else{
        for(j=0; j < 0x07C0; j++);         /* 200μs未満                       */
    }
}
/*****************************************************************************/
/*            WDT：ｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏ設定ﾙｰﾁﾝ                                      */
/*****************************************************************************/
static void SET_WDT_P(){                   /* 書込み時のｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏの設定    */
       WDT_RSTCSRW = 0x5A5F;               /* 内部ﾊﾟﾜｰｵﾝﾘｾｯﾄ                   */
       WDT_TCNTW = 0x5A00;                 /* 409.6μs
    WDT_TCSRW = 0xA579;                    /* WDTｽﾀｰﾄ,φ'=φ/64                 */
}
static void SET_WDT_E(){                   /* 消去時のｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏの設定    */
       WDT_RSTCSRW = 0x5A5F;               /* 内部ﾊﾟﾜｰｵﾝﾘｾｯﾄ                   */
       WDT_TCNTW = 0x5A80;                 /* 13.2ms
    WDT_TCSRW = 0xA57E;                    /* WDTｽﾀｰﾄ,φ'=φ/4096               */
}
static void RESET_WDT(){                   /* ｳｫｯﾁﾄﾞｯｸﾞﾀｲﾏの設定解除          */
    WDT_TCSRW = 0xA55E;                    /* WDT停止                          */
}
/*****************************************************************************/
/*            ROMtoRAMcopyADDRESS                                            */
/*****************************************************************************/
#pragma section
int ROMtoRAMcopyADDRESS(){
#pragma asm
    .export _CopyTopAddress,_CopyEndAddress
_CopyTopAddress .data.l    (STARTOF P_ROMtoRAM)
_CopyEndAddress .data.l    (STARTOF P_ROMtoRAM)+(SIZEOF P_ROMtoRAM)+(SIZEOF C_ROMtoRAM)
#pragma endasm
}
```

プログラムリスト　F_WRITE.h(ﾍｯﾀﾞﾌｧｲﾙ)

```
/*****************************************************************************/
/*            定数の定義                                                     */
/*****************************************************************************/
#define BLK0 0x01                   /* EBRのﾌﾞﾛｯｸ0指定                       */
#define BLK1 0x02                   /* EBRのﾌﾞﾛｯｸ1指定                       */
#define BLK2 0x04                   /* EBRのﾌﾞﾛｯｸ2指定                       */
#define BLK3 0x08                   /* EBRのﾌﾞﾛｯｸ3指定                       */
#define BLK4 0x10                   /* EBRのﾌﾞﾛｯｸ4指定                       */
#define BLK5 0x20                   /* EBRのﾌﾞﾛｯｸ5指定                       */
#define BLK6 0x40                   /* EBRのﾌﾞﾛｯｸ6指定                       */
#define BLK7 0x80                   /* EBRのﾌﾞﾛｯｸ7指定                       */
#define BLK0TOP 0x00000000          /* 消去ﾌﾞﾛｯｸ0の先頭アドレス              */
#define BLK1TOP 0x00001000          /* 消去ﾌﾞﾛｯｸ1の先頭アドレス              */
#define BLK2TOP 0x00002000          /* 消去ﾌﾞﾛｯｸ2の先頭アドレス              */
#define BLK3TOP 0x00003000          /* 消去ﾌﾞﾛｯｸ3の先頭アドレス              */
#define BLK4TOP 0x00004000          /* 消去ﾌﾞﾛｯｸ4の先頭アドレス              */
#define BLK5TOP 0x00005000          /* 消去ﾌﾞﾛｯｸ5の先頭アドレス              */
#define BLK6TOP 0x00006000          /* 消去ﾌﾞﾛｯｸ6の先頭アドレス              */
#define BLK7TOP 0x00007000          /* 消去ﾌﾞﾛｯｸ7の先頭アドレス              */
#define BLK7END 0x00008000          /* 消去ﾌﾞﾛｯｸ7の終了アドレス              */
#define MAX_erase_time    100       /* 消去：10msec * 100回                  */
#define MAX_program_time 1000       /* 書込み：50μsec * 4回, 200μsec * 996回 */
#define I_RAMadd 0xFFFF6000         /* ｴﾐｭﾚｰｼｮﾝRAM先頭ｱﾄﾞﾚｽ                  */
#define RAM_TOP  0xFFFF7000         /* ﾃﾞｰﾀ変換ﾌﾟﾛｸﾞﾗﾑ転送先(内蔵RAM)       */
#define OK      1
#define NG      0
#define CONT    3
#define OT      2
```

メモリマップ

図３に本アプリケーションのメモリマップを示します。
また、参照用にバッチファイル例、及びサブコマンドファイル例を示します。
フラッシュメモリのデータ変換は、RAM上にコピーしたデータ変換プログラムにより行われます。

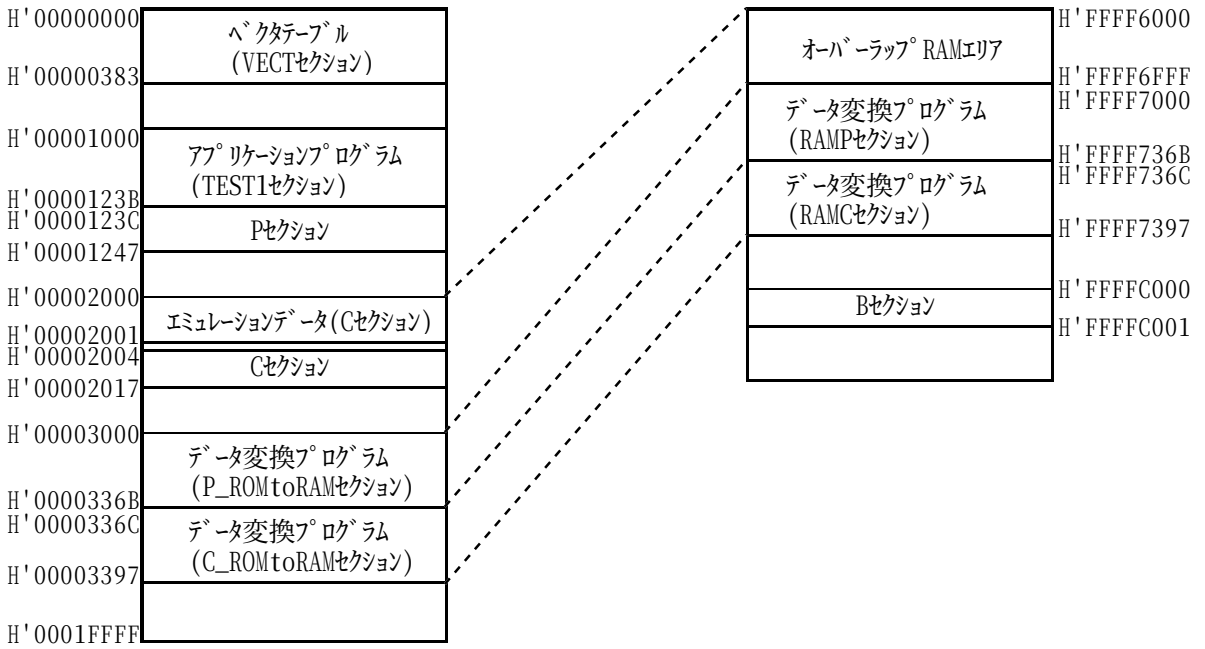


図３ メモリマップ

バッチファイル例

```
asmsh initsct.src -cp=sh2 -lis
shc vec.c -section=d=VECT -debug -cp=sh2 -l
shc TEST.c -section=p=TEST1 -debug -cp=sh2 -lis -sh=so
shc chgprg.c -debug -code=asmcode -cp=sh2 -lis -sh=so
asmsh chgprg.SRC -cp=sh2 /lis

lnk -subcommand=TEST.sub
cnvs TEST.abs
```

サブコマンドファイル例

```
debug
INPUT vec,TEST
INPUT initsct,chgprg
LIB c:\shc\lib\shclib.lib
OUTPUT TEST
ROM (C_ROMtoRAM,RAMC),(P_ROMtoRAM,RAMP)
START VECT(0),TEST1,P,D(00001000),C(00002000),P_ROMtoRAM,C_ROMtoRAM(00003000)
,RAMP,RAMC(0FFFF7000),R,B(0FFFFC000)
FORM A
PRINT TEST

EXIT
```

SH7055 内蔵I/O編 アプリケーションノート

発行年月　平成11年３月　第１版
発　　行　株式会社　日立製作所
　　半導体事業本部　統括営業本部
編　　集　株式会社　超Lメディア
　　技術ドキュメントグループ

SH7055 内蔵 I/O 編
アプリケーションノート

ルネサス エレクトロニクス株式会社
神奈川県川崎市中原区下沼部1753 〒211-8668

ADJ-502-075