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

H8/300Lシリーズ

アプリケーションノート（応用編）

H8/3644シリーズ
H8/3657シリーズ

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00　1999.02

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

H8/3644シリーズおよびH8/3657シリーズは、高速H8/300L〉CPUを核にして、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マイクロコンピュータです。

H8/300L〉CPUは、H8/300〉CPUと互換性のある命令体系を備えています。

H8/3644シリーズおよびH8/3657シリーズは、システム構成に必要な周辺機能として、5種類のタイマ、14ビットPWM、2チャネルのシリアルコミュニケーションインタフェース、A/D変換器を内蔵しており、高度な制御システムの組込み用マイコンとして活用できます。

H8/300Lシリーズ〉-H8/3644、H8/3657-〉アプリケーションノート　-応用編-〉は、H8/3644シリーズおよびH8/3657シリーズの内蔵周辺機能を組み合わせて使用した場合の動作例を示した“応用編”により構成されており、ユーザにてソフトウェア設計およ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プログラム、回路等の動作は確認しておりますが、実際にご使用になる場合は、必ず動作確認の上ご使用くださいますようにお願い致します。

目次

1. H8/300Lシリーズ -H8/3644、H8/3657- アプリケーションノート -応用編- 使用手引

目次

本アプリケーションノートは、図1に示すように2部構成になっています。

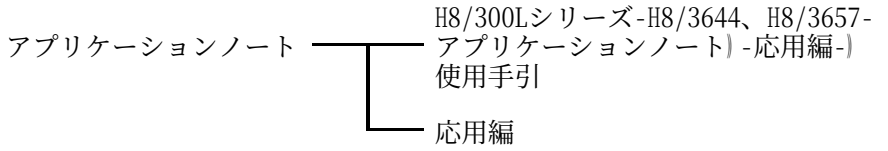


図1　アプリケーションノート構成

(1)H8/300Lシリーズ-H8/3644、H8/3657-アプリケーションノート》-応用編-》使用手引

H8/300Lシリーズ-H8/3644、H8/3657-アプリケーションノート》-》応用編-》の使用法について説明しています。

(2)応用編

H8/3644シリーズ、H8/3657シリーズの内蔵周辺機能（タイマ、シリアルコミュニケーションインタフェース、A/Dコンバータ、PWM、I/Oポート、割込み、低消費電力モード等）を組み合わせて使用した場合の使用法を簡単なタスク例をもとに説明しています。

1.1》応用編構成

応用編は図2に示す構成で内蔵周辺機能を組み合わせて使用した場合の使用法について説明しています。

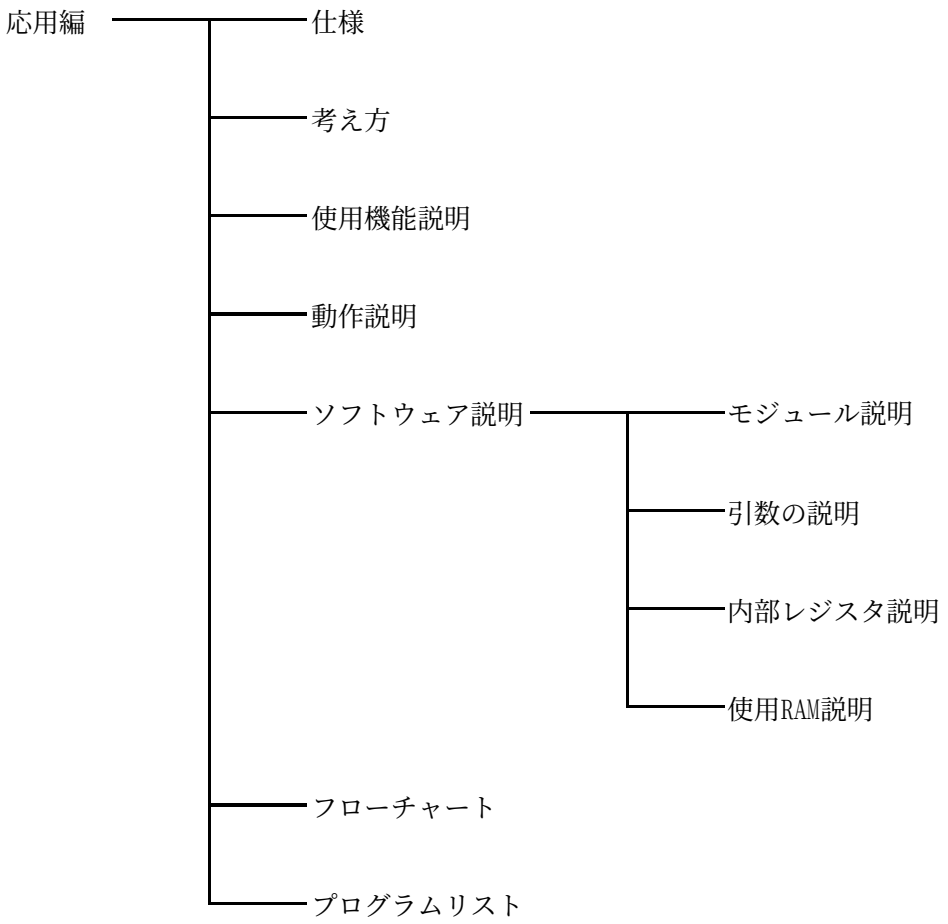


図2　応用編構成

(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) 考え方
タスク例のシステムを実現するための方法を説明しています。

(3) 使用機能説明
タスク例で使用する周辺機能の特徴および周辺機能の割付けについて説明しています。

(4) 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。

(5) 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）について説明しています。

(d) 使用RAM説明
モジュールで使用するRAMのラベル名及び機能について説明しています。

(6) フローチャート
タスク例を実行するソフトウェアについてフローチャートを使用し説明しています。

(7) 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ています。

2. 応用編

目次

2.1 EEPROMとのインタフェース

EEPROMとのインタフェース	使用機能	I/Oポート

仕様

(1) 図1に示すように、SCL（Serial Clock）とSDA（Serial Data）の2線により接続されたEEPROM（HN58X2408）との1：1のインターフェースを行ないます。
(2) ROM上のデータをEEPOROMにライトし、EEPROMにライトされたデータを再びRAMにリードします。
(3) 転送データは、ポート7_3に接続されたLEDを点灯させるプログラムです。
(4) データのLSB（最下位ビット）よりデータを送信/受信するLSBファースト方式による転送を行ないます。

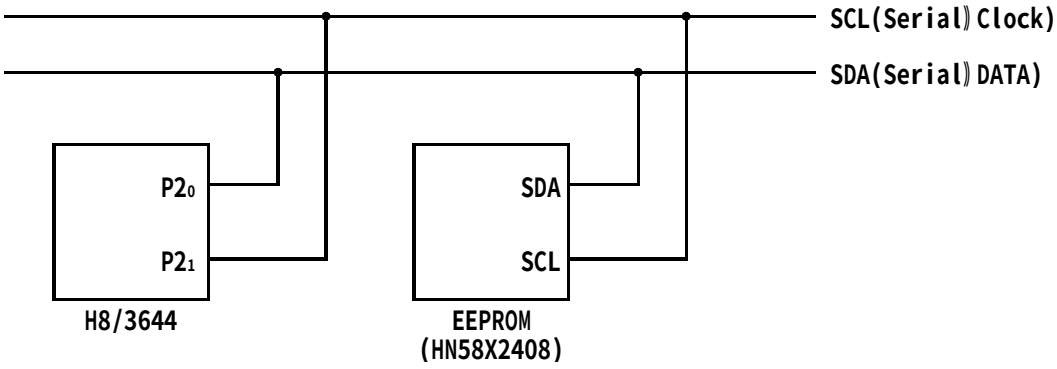


図1 EEPROMとの接続例

考え方

(1) 図2に本タスク例で使用するEEPROM（HN58X2408）のバス・タイミングを示します。

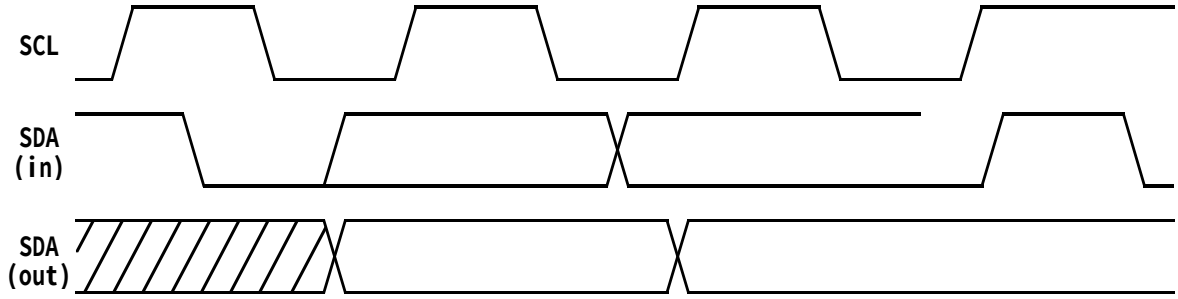


図2 EEPROM（HN58X2408）バス・タイミング

(2) 使用するEEPROMのバス・タイミングが図2に示すようなバス・タイミングのため、本タスク例では、ソフトウェアによりSCLに出力するクロックを$P2_1$端子レベルを"High"、"Low"に設定することによりシリアルクロックを生成しています。ソフトウェアにより生成したシリアルクロックに同期して$P2_0$端子より、シリアルデータを出力/入力することによりEEPROM（HN58X2408）とのインタフェースを実現しています。図3に$P2_1$端子、および$P2_0$端子レベルのタイミング波形を示します。

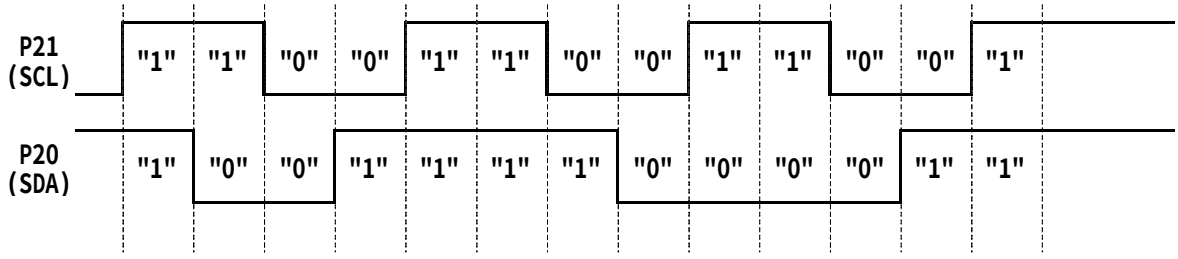


図3 $P2_1$端子、および$P2_0$端子レベルのタイミング波形

EEPROMとのインタフェース	使用機能	I/Oポート

使用機能説明

(1) 本タスク例では、図4に示すようにH8/3644とEEPROM（HN58X2408）を接続し、インタフェースを行ないます。また、表1にEEPROM（HN58X2408）のピン説明を示します。

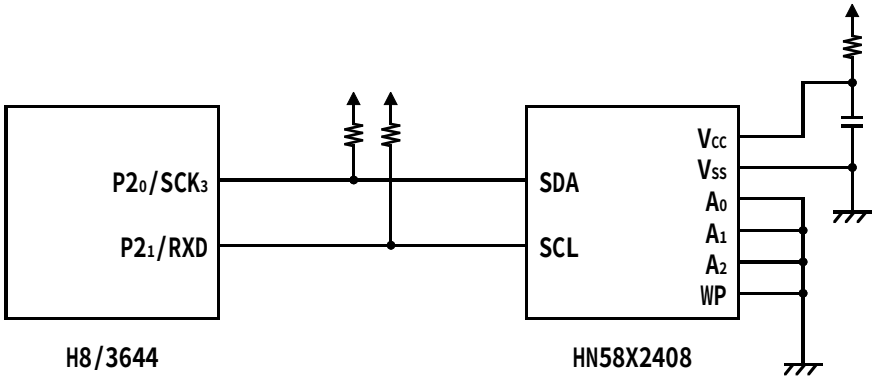


図4　H8/3644とHN58X2408の接続図

表1　HN58X2408ピン説明

Pin名称	機能
A0～A2	デバイス・アドレス
SCL	シリアル・クロック入力
SDA	シリアル・データ入力/出力
WP	ライト・プロテクト
V_{CC}	V_{CC}
V_{SS}	GND

(2) 図5に示すようなブロック図により、H8/3644とEEPROM（HN58X2408）のインタフェースを行ないます。以下にH8/3644の機能について説明します。

(a) ポート2は、3ビットの入出力ポートで、$P2_0$/SCK_3、$P2_1$/RXD、$P2_2$/TXDの3つの端子により構成されています。
(b) $P2_0$/SCK_3端子をHN58X2408のSDA端子に接続し、シリアル・データの入出力端子として使用します。
(c) $P2_1$/RXD端子をHN58X2408のSCL端子に接続し、シリアル・クロックの出力端子として使用します。
(d) ポート7は、5ビットの入出力ポートで、$P7_3$、$P7_4$/TMRIV、$P7_5$/TCIV、$P7_6$/TMOV、$P7_7$の5つの端子により構成されています。
(e) $P7_3$端子をLEDへの出力端子として使用します。
(f) $P2_0$/SCK_3端子は、シリアルコントロールレジスタ3（SCR3）のクロックイネーブル1（CKE1）を"0"に、クロックイネーブル0（CKE0）を"0"に、シリアルモードレジスタ（SMR）のコミュニケーションモード（COM）を"0"に設定することにより$P2_0$入出力端子として機能します。また、ポートコントロールレジスタ2（PCR2）のポートコントロールレジスタ2_0（$PCR2_0$）を"0"に設定すると$P2_0$入力端子として、$PCR2_0$を"1"に設定すると$P2_0$出力端子として機能します。
(g) $P2_1$/RXD端子は、SCR3のレシーブイネーブル（RE）を"0"に設定することにより$P2_1$入出力端子として機能します。また、PCR2のポートコントロールレジスタ2_1（$PCR2_1$）を"0"に設定すると$P2_1$入力端子として、$PCR2_1$を"1"に設定すると$P2_1$出力端子として機能します。
(h) $P7_3$端子は、ポートコントロールレジスタ7（PCR7）のポートコントロールレジスタ7_3（$PCR7_3$）を"1"に設定することにより$P7_3$出力端子として機能します。

EEPROMとのインタフェース	使用機能	I/Oポート

使用機能説明

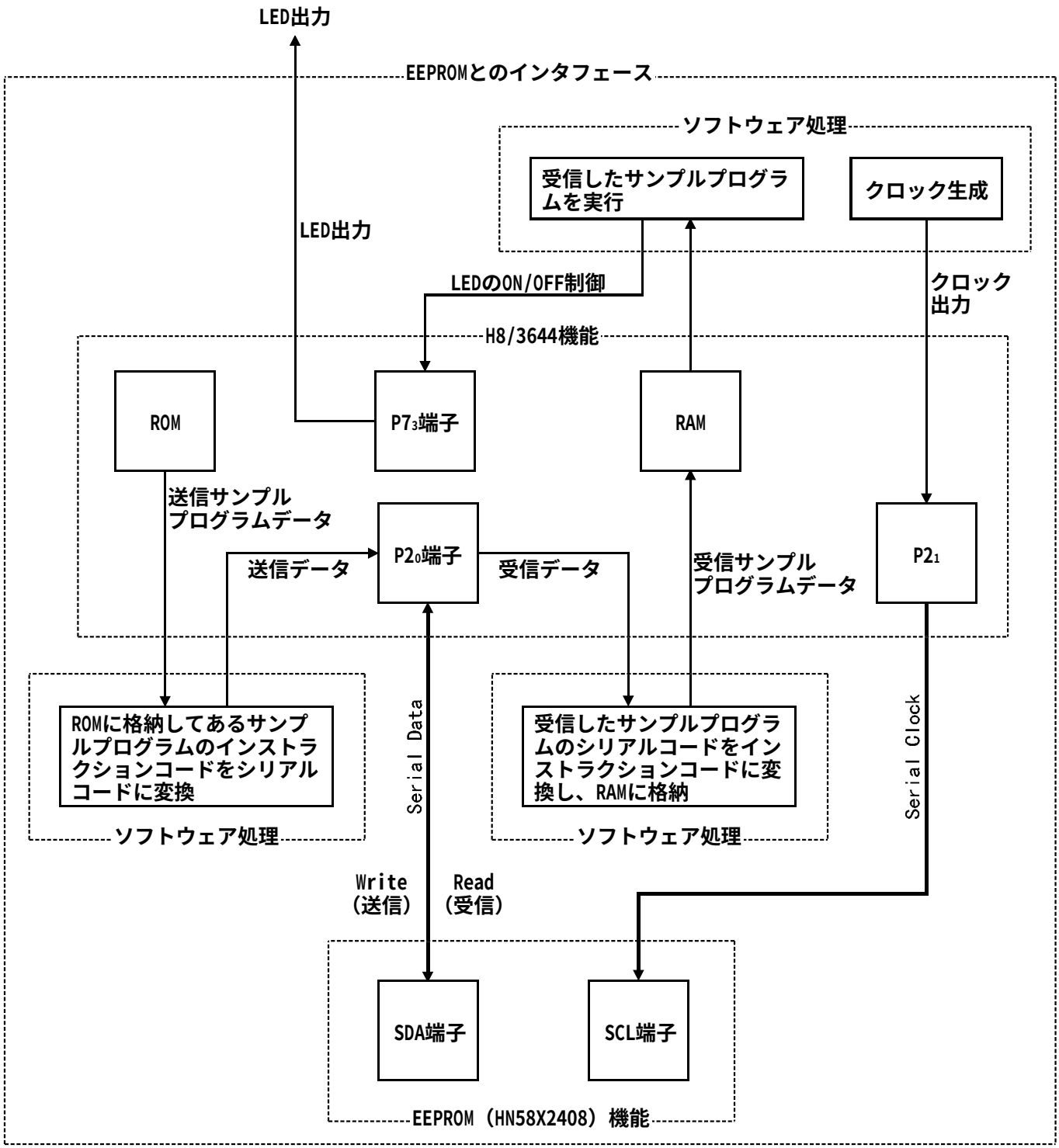


図5　EEPROMとのインタフェースのブロック図

使用機能説明

(2) 表2に本タスク例の機能割付けを示します。表2に示すようにH8/3644の機能を割付け、EEPROMとのインタフェースを行ないます。

表2　H8/3644機能割付け

H8/3644機能	機能
$P2_0$端子	シリアル・データ入出力
$P2_0$	$P2_0$端子のデータを格納
$PCR2_0$	$P2_0$入出力端子機能の設定
$P2_1$端子	シリアル・クロック出力
$P2_1$	$P2_1$端子のデータを格納
$PCR2_1$	$P2_1$入出力端子機能の設定
$P7_3$端子	LED出力
$P7_3$	$P7_3$端子のデータを格納
$PCR7_3$	$P7_3$入出力端子機能の設定

(4) 以下に本タスク例で使用しているEEPROM（HN58X2408）の仕様について説明します。

(a) 本タスク例で使用するEEPROM（HN58X2408）は、2線式シリアルインタフェースのEEPROM（電気的に書き換え可能なROM）で、最新のNMOSメモリ技術、CMOSプロセスおよび低電圧回路技術を採用し、低電源電圧動作・低消費電力・高速動作・高信頼性を実現したEEPROMです。32バイトページ書き換え機能により、データ書き換えが高速化されています。

(b) 以下に本タスク例で使用しているEEPROM（HN58X2408）の特徴を示します。
- 単一電源　：　1.8V～5.5V
- 2線式シリアルインタフェース
- 動作周波数　：　400kHz
- 消費電流
  - スタンバイ時　：　3μA（max）
  - 読み出し時　：　1mA（max）
  - 書き換え時　：　3mA（max）
- ページ書き換え　：　ページサイズ32バイト
- 書き換え時間　：　10ms（2.7V以上）/15ms（1.8V～2.7V）
- 書き換え回数　：　10^5回（ページ書き換え時）

(c) Read、Writeの動作を開始するには、SCL入力がHighの期間に、SDA入力をHighからLowにするスタート・コンディションにする必要があります。また、SDA入力がHighの期間に、SDA入力をLowからHighにすることで、ストップ・コンディションになります。Readの場合、ストップコンディションを入力するとReadが終了し、スタンバイ状態になります。Writeの場合は、ストップ・コンディション入力で書き換えデータの入力終了となり、メモリへの書き込みを書き換え時間（t_{WC}）の期間実施した後、スタンバイモードになります。図6にスタート・コンディション、ストップ・コンディションのタイミング波形を示します。

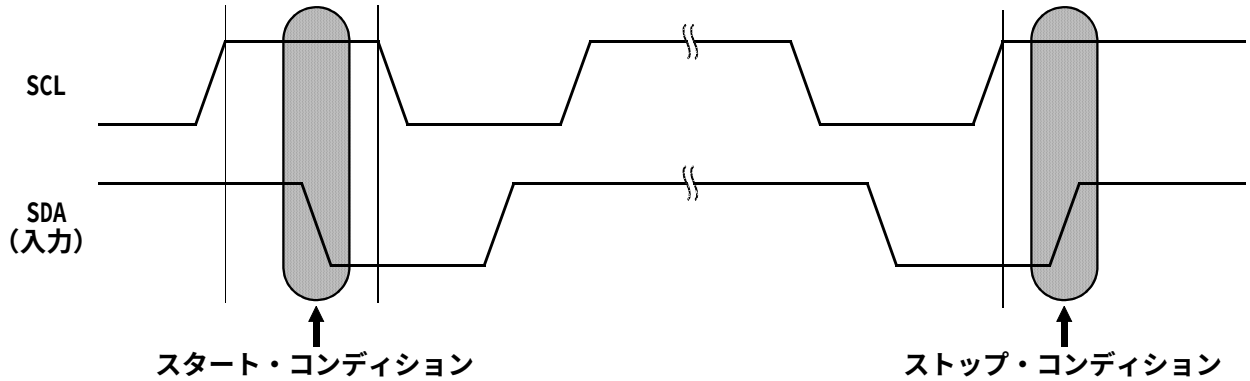


図6　スタート・コンディション、およびストップ・コンディション信号出力タイミング波形

(c) アドレス情報、Read情報等のシリアルデータは、8ビット単位で送受信が行われます。この8ビットのデータが正常に送信または受信されたことを示すのがAcknowledge信号で、SCLの9クロック目に受信側が"0"を出力します。送信側は、この9クロック目でAcknowledge信号を受信するために、バスを開放します。EEPROMから見ると、Writeの場合は全て受信となるため、8ビットの受信が完了したら、9クロック目にEEPROMからAcknowledge信号の"0"を出力します。Readの場合は、スタートコンディションの後の8ビット受信後にAcknowledge信号の"0"を出力します。これに続いて、EEPROMはReadデータを8ビット単位で出力しますが、出力後はバスを開放し、マスタ側からAcknowledge信号の"0"が送られるのを待ちます。Acknowledge信号の"0"を検出すると、EEPROMは次のアドレスのReadデータを出力します。Acknoeledge信号の"0"が検出されずにストップ・コンディションを受信すると、Read動作を終了しスタンバイ状態になります。なお、Ackowledge信号の"0"が検出されず、かつストップコンディションも送られてこない場合は、データを出力せずにバス解放状態を持続します。図7にAckowledge信号のタイミング波形を示します。

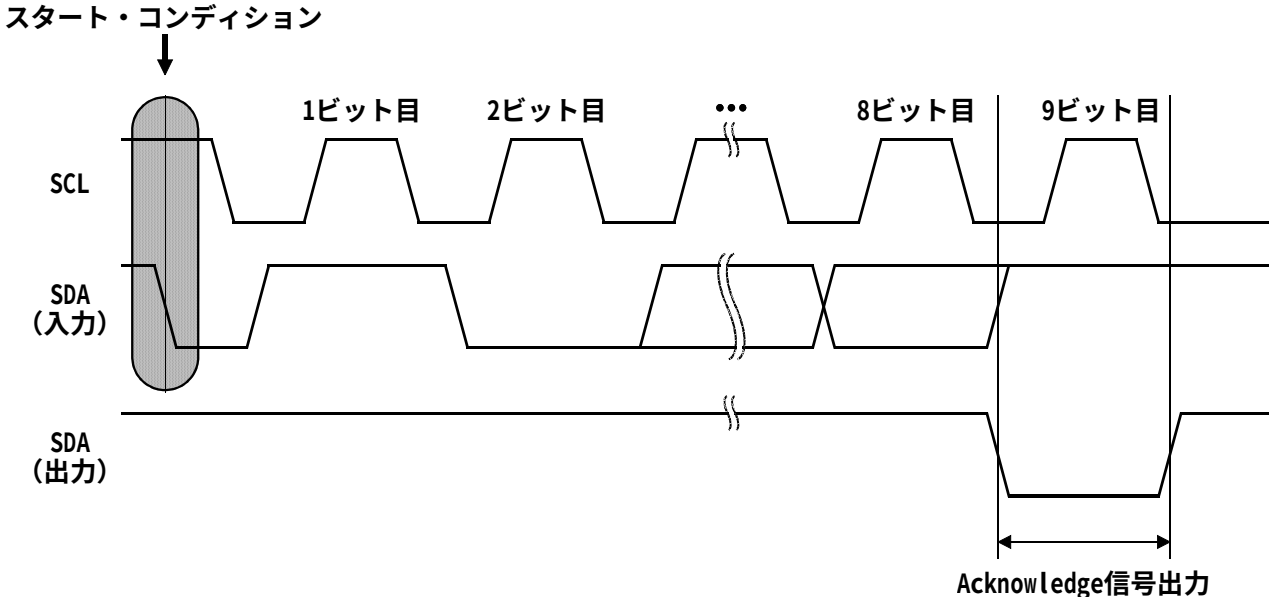


図7　Acknowledge信号出力タイミング波形

(d) スタートコンディションに続いて8ビットのデバイス・アドレス・ワードを入力します。この入力でデバイスはRead、Writeの動作を開始します。デバイス・アドレス・ワードはデバイス・コード4ビット、デバイス・アドレス・コード3ビット、Read/Writeコード1ビットの3つのコードで構成されています。デバイス・アドレス・ワードの上位4ビットはデバイス・タイプを識別するデバイス・コードで、本タスク例で使用しているEEPROM（HN58X2408）では"1010"の固定コードとなります。デバイス・コードに続けてデバイス・アドレス・コード3ビットをA2、A1、A0の順に入力します。デバイス・アドレス・コードはバスに最大8ケ接続されたデバイスのうち、どれを選択するかを決定します。本タスク例で使用するEEPROM（HN58X2408）のデバイス・アドレス・コードは"000"に設定しています。デバイス・アドレス・ワードの8ビット目はR/Wコードです。"0"入力の場合はWrite動作、"1"入力の場合はRead動作になります。なお、デバイス・コードが"1010"でない場合、もしくはデバイス・アドレス・コードが一致しない場合は、Read/Write動作に入らず、スタンバイモードになります。図8にデバイス・アドレス・ワードについて示します。

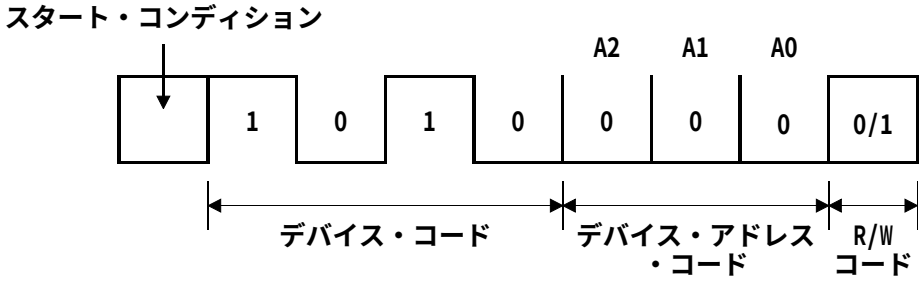


図8　デバイス・アドレス・ワード

使用機能説明

(e) 本タスク例では、32バイトまでの任意のバイト数を一度に書き換えるPage Write機能を使用してWrite動作を行ないます。スタート・コンディション→デバイス・アドレス・ワード→メモリ・アドレス（n）→Writeデータ（Dn）の順に、9ビットごとのAcknowledge信号の"0"出力を確認しながら入力します。Writeデータ（Dn+1）を入力すると、Page Writeモード入ります。Writeデータ（Dn+1）を入力した時点で、ページ内アドレス（a0～a4）は自動的にインクリメントされ（n+1）番地になります。このように、Writeデータを次々と入力することができ、Writeデータ入力ごとにページ内アドレスがインクリメントされ、最大32バイトのWriteデータを入力できます。ページ内アドレス（a0～a4）がページの最終番地に達した場合は、アドレスは"Roll Over"して、ページの先頭アドレスに戻ります。"Roll Over"した場合は、同一アドレスにWriteデータが2度（以上）入力されることになりますが、最後に入力したWriteデータが有効になります。ストップ・コンディションを入力すると、Writeデータの入力を終了し、書き換え動作に入ります。図9にPage Write動作について示します。

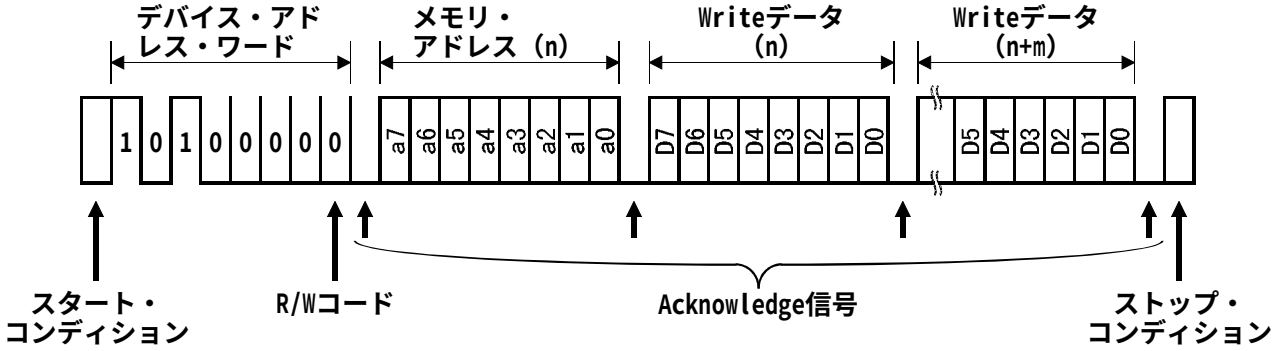


図9　Page Write動作

(f) EEPROMが書き換え中か否かを判定する機能として、Ackonowlege Pollingがあります。書き換え期間中にスタート・コンディションに続いてデバイス・アドレス・ワード8ビットを入力します。Acknowledge Pollingの場合、Read/Writeコードは"1"/"0"のどちらでもかまいません。9ビット目のAcknowledge信号で書き換え中か否かを判定します。Acknowledge信号が"1"のときは書き換え中、Acknowledge信号が"0"のときは書き換え終了を示します。Acknowledge Pollingは、Writeデータ入力後、ストップ・コンディションが入力された時点から機能します。

(g) 本タスク例では、データを連続してReadするSequential Readモードを使用してRead動作を行ないます。ダミーのWriteモードでReadするデータの先頭アドレスを入力します。8ビットのデータを出力した後、Acknowledge信号の"0"を入力すると、アドレスがインクリメントされ、次の8ビットのデータが出力されます。データ出力後にAcknowledge信号の"0"の入力を続けるとアドレスをインクリメントしながら次々とデータを出力します。アドレスが最終アドレスになった場合は、0番地に"Roll Over"します。"Roll Over"後もSequential Readが可能です。動作を終了するには、Acknowledge信号の"1"（Acknowledge信号の入力をせずに、バスを解放しても可）→ストップコンディションの順で入力します。図10にSequential Read動作について示します。

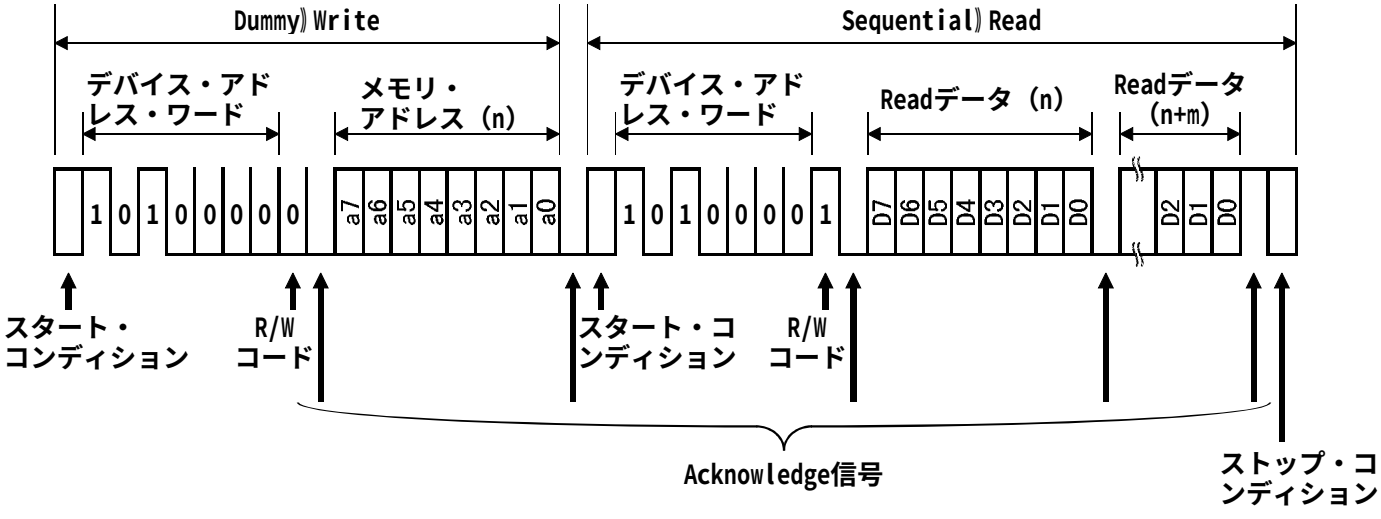


図10　Sequential Read動作

(h) EEPROM（HN58X2408）は、Write動作が終了し、Read動作に入るときには、書き換え時間（t_{WC}）の⟫期間、待機しなければなりません。書き換え時間（t_{WC}）は、5V動作時、10ms（max）です。書き⟫換え時間のタイミング波形を図11に示します。

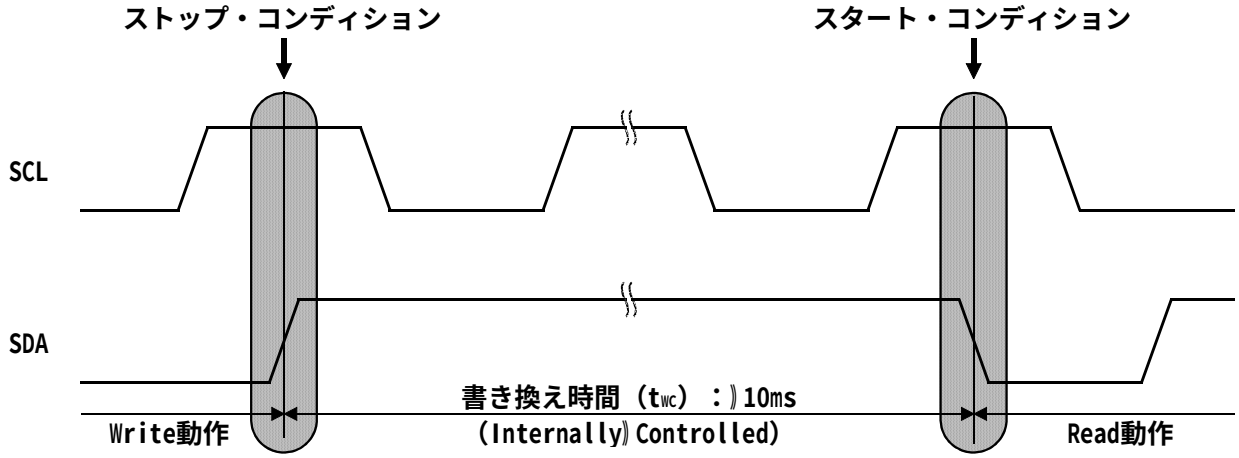
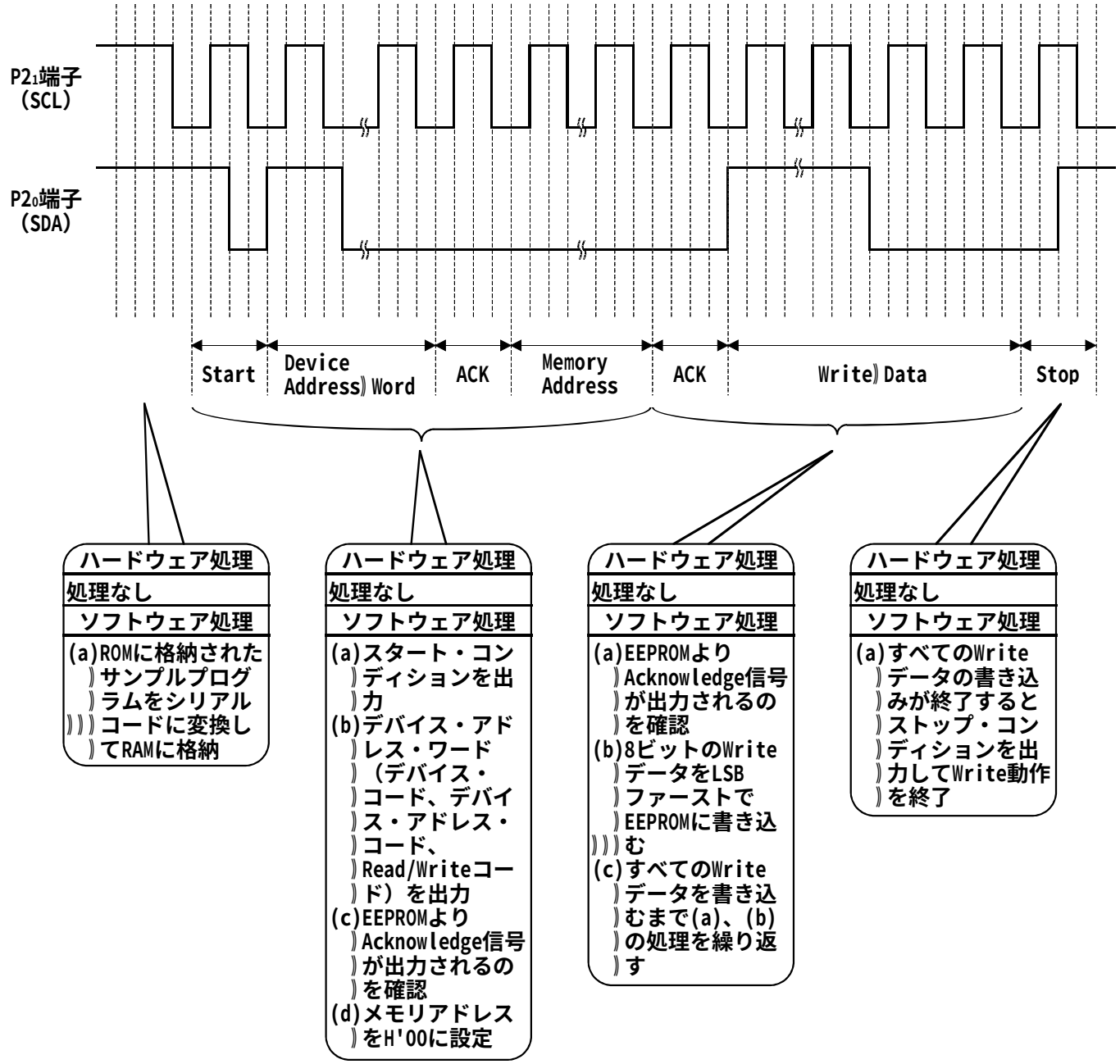


図11　書き換え時間のタイミング波形

EEPROMとのインタフェース	使用機能	I/Oポート

動作原理

（1）図12にWrite（送信）時の動作原理を示します。図12に示すようにH8/3644のハードウェア処理、およびソフトウェア処理によりEEPROMへのWrite（送信）を行ないます。

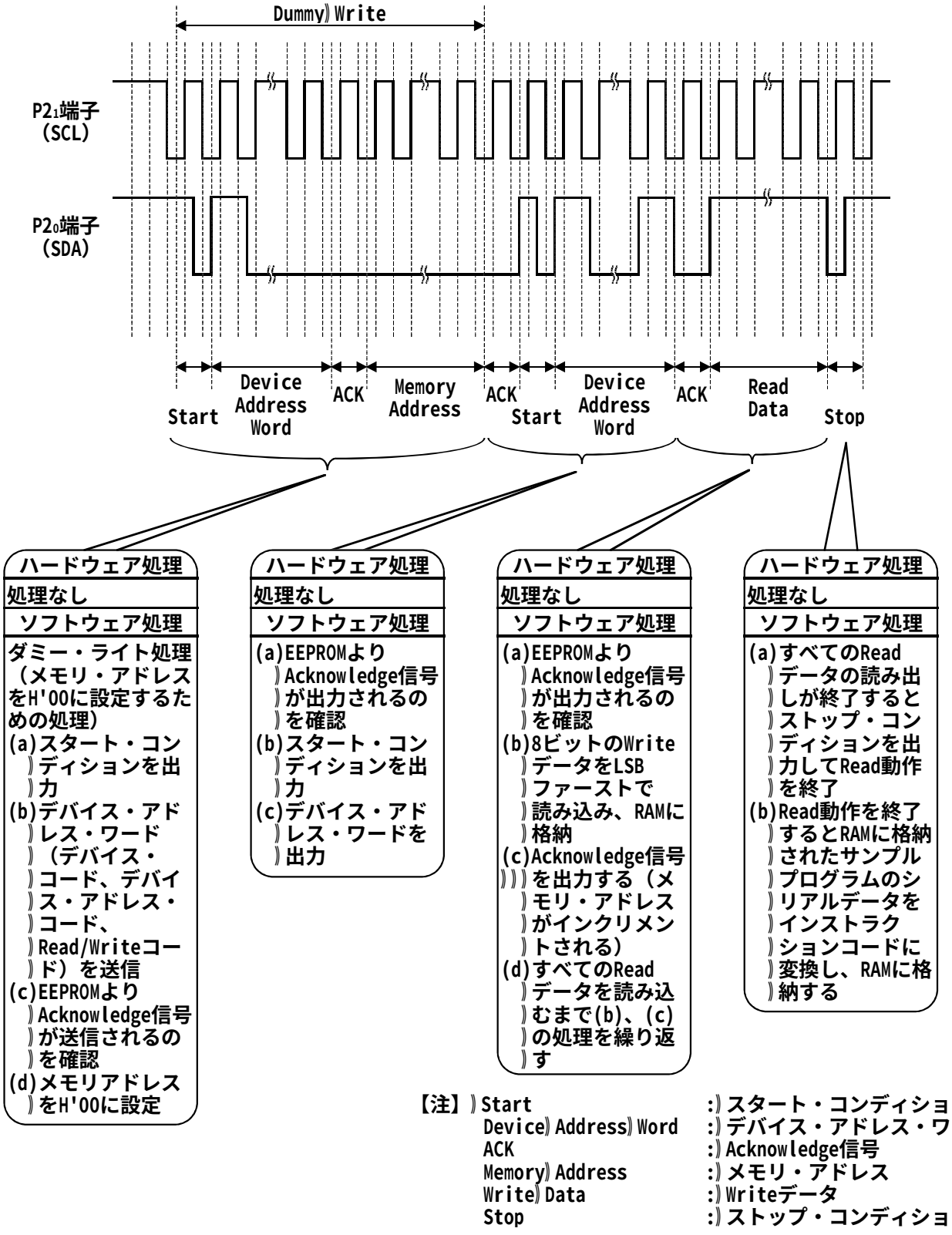


【注】Start :スタート・コンディション
Device Address Word :デバイス・アドレス・ワード
ACK :Acknowledge信号
Memory Address :メモリ・アドレス
Write Data :Writeデータ
Stop :ストップ・コンディション

図12　EEPROMへのWrite時の動作原理

動作原理

（2）図13にRead（受信）時の動作原理を示します。図13に示すようにH8/3644のハードウェア処理、およびソフトウェア処理によりEEPROMからのRead（受信）を行ないます。



【注】Start :スタート・コンディション
Device Address Word :デバイス・アドレス・ワード
ACK :Acknowledge信号
Memory Address :メモリ・アドレス
Write Data :Writeデータ
Stop :ストップ・コンディション

図13　EEPROMからのRead時の動作原理

(3) 本タスク例で使用しているポート2への出力、および入力の動作について表3に示します。表3に示すような設定により、シリアル・クロックの出力、およびシリアルデータの入出力を行ないます。

表3　$P2_1$(SCL)および$P2_0$(SDA)入出力設定

端子設定＼出力値		$P2_1$(SCL)=1 $P2_0$(SDA)=1	$P2_1$(SCL)=1 $P2_0$(SDA)=0	$P2_1$(SCL)=0 $P2_0$(SDA)=1	$P2_1$(SCL)=0 $P2_0$(SDA)=0
PDR2	$P2_1$	0	0	0	0
	$P2_0$	0	0	0	0
PCR2	$PCR2_1$	0 (入力端子機能)	0 (入力端子機能)	1 (出力端子機能)	1 (出力端子機能)
	$PCR2_0$	0 (入力端子機能)	1 (出力端子機能)	0 (入力端子機能)	1 (出力端子機能)

(3) 図14に本タスク例で使用しているH8/3644のメモリマップを示します。

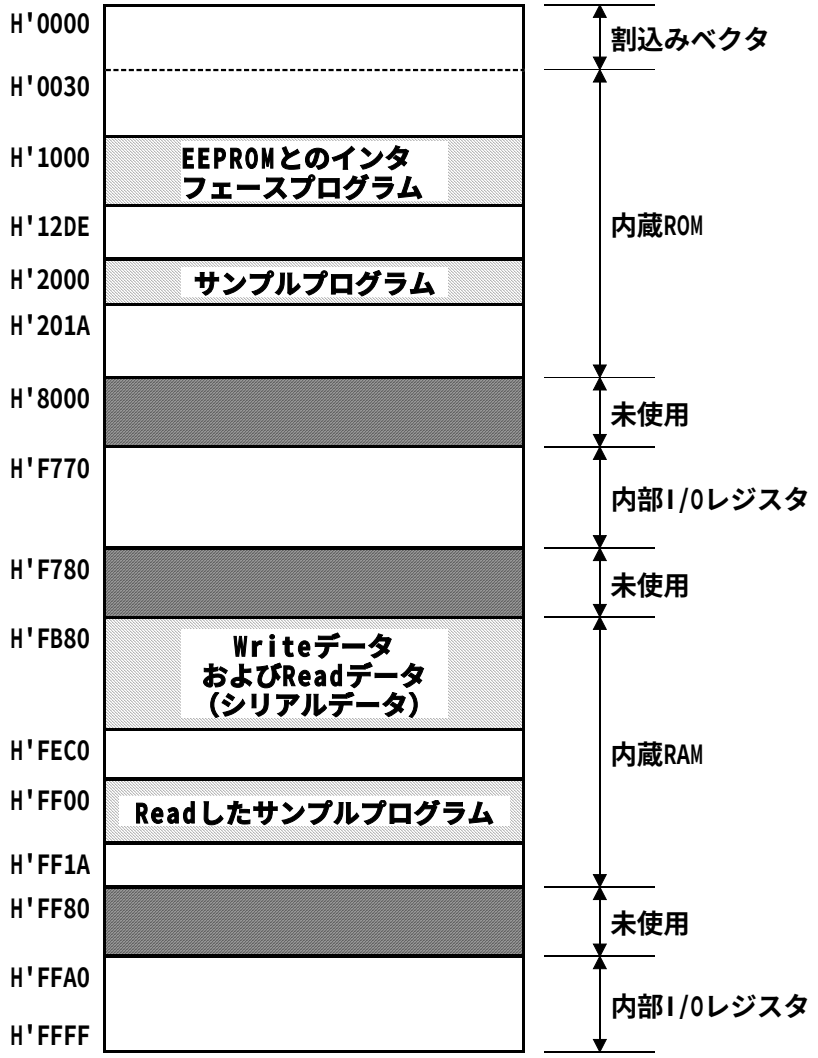


図14　本タスク例で使用するH8/3644のメモリマップ

EEPROMとのインタフェース	使用機能	I/Oポート

ソフトウェア説明

(1) モジュール説明

表4に本タスク例におけるモジュール説明を示します。

表4　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	スタックポインタのイニシャライズ、割込みの禁止、使用RAMのイニシャライズ、書き換え時間の待機、EEPROMの書き込み、読み出しの制御を行なう
シリアルコード変換	SERCODE	ROMのサンプルプログラムのデータをシリアルデータに変換し、RAMに格納する
書き込み	WRITE	RAMに格納されているサンプルプログラムのシリアルデータをEEPROMにLSBファースト方式により送信
読み出し	READ	EEPROMからサンプルプログラムのシリアルデータをLSBファースト方式で受信し、RAMに格納
インストラクションコード変換	PARCODE	RAMに格納されたサンプルプログラムのシリアルデータをインストラクションコードに変換
受信サンプルプログラム	SPLPGM	EEPROMから受信したサンプルプログラムで、$P7_3$端子に接続されたLEDを262msごとに点灯、または消灯を繰り返す
エラールーチン	ERORR	エラー処理を行なう

(2) 引数の説明

本タスク例では、引数は使用していません。

(3) 使用内部レジスタ説明

表5に本タスク例におけるH8/3644の使用内部レジスタ説明を示します。

表5　H8/3644の使用内部レジスタ説明

レジスタ名		機能	アドレス	設定値
PDR2	$P2_1$	ポートデータレジスタ2（ポートデータレジスタ2_1） ：$P2_1$が"0"のとき、$P2_1$端子のデータは"0" ：$P2_1$が"1"のとき、$P2_1$端子のデータは"1"	H'FFD5 ビット1	0/1
	$P2_0$	ポートデータレジスタ2（ポートデータレジスタ2_0） ：$P2_0$が"0"のとき、$P2_0$端子のデータは"0" ：$P2_0$が"1"のとき、$P2_0$端子のデータは"1"	H'FFD5 ビット0	0/1
PCR2	$PCR2_1$	ポートコントロールレジスタ2 （ポートコントロールレジスタ2_1） ：$PCR2_1$が"0"のとき、$P2_1$端子は入力端子として機能 ：$PCR2_1$が"1"のとき、$P2_1$端子は出力端子として機能	H'FFE5 ビット1	0/1
	$PCR2_0$	ポートコントロールレジスタ2 （ポートコントロールレジスタ2_0） ：$PCR2_0$が"0"のとき、$P2_0$端子は入力端子として機能 ：$PCR2_0$が"1"のとき、$P2_0$端子は出力端子として機能	H'FFE5 ビット0	0/1
PDR7	$P7_3$	ポートデータレジスタ7（ポートデータレジスタ7_3） ：$P7_3$が"0"のとき、$P7_3$端子のデータは"0" ：$P7_3$が"1"のとき、$P7_3$端子のデータは"1"	H'FFDA ビット3	0/1
PCR7	$PCR7_3$	ポートコントロールレジスタ7 （ポートコントロールレジスタ7_3） ：$PCR7_3$が"0"のとき、$P7_3$端子は入力端子として機能 ：$PCR7_3$が"1"のとき、$P7_3$端子は出力端子として機能	H'FFEA ビット3	1

(4) 使用RAM説明

表6に本タスク例における使用RAM説明を示します。

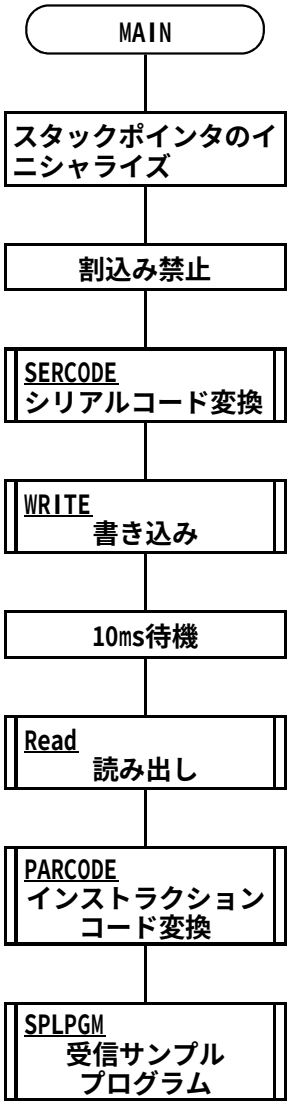
表6　使用RAM説明

ラベル名	機能	アドレス	使用モジュール名
SERAREA	サンプルプログラムのシリアルデータを格納するRAMの先頭番地を格納	H'FB80	シリアルコード変換 書き込み 読み出し インストラクションコード変換
COUNTER	サンプルプログラムのシリアルデータを格納するRAMの最終番地を格納	H'FEC0	シリアルコード変換
SPLPGM	EEPROMから読み出したサンプルプログラムのインストラクションコードを格納するRAMの先頭番地を格納	H'FF00	シリアルコード変換 インストラクションコード変換

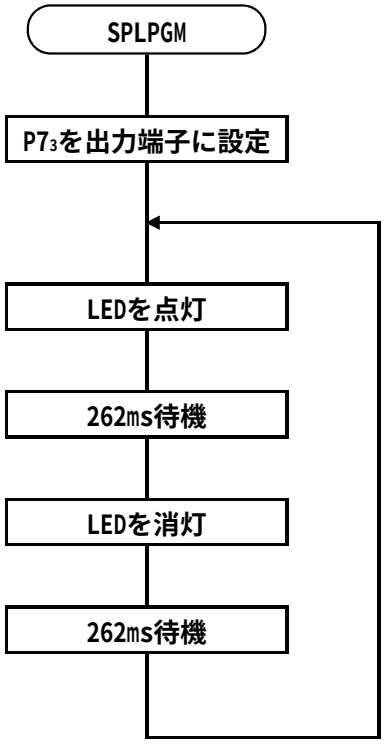
EEPROMとのインタフェース	使用機能	I/Oポート

フローチャート

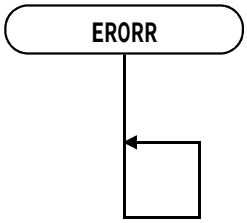
（a）メインルーチン



（b）受信サンプルプログラム

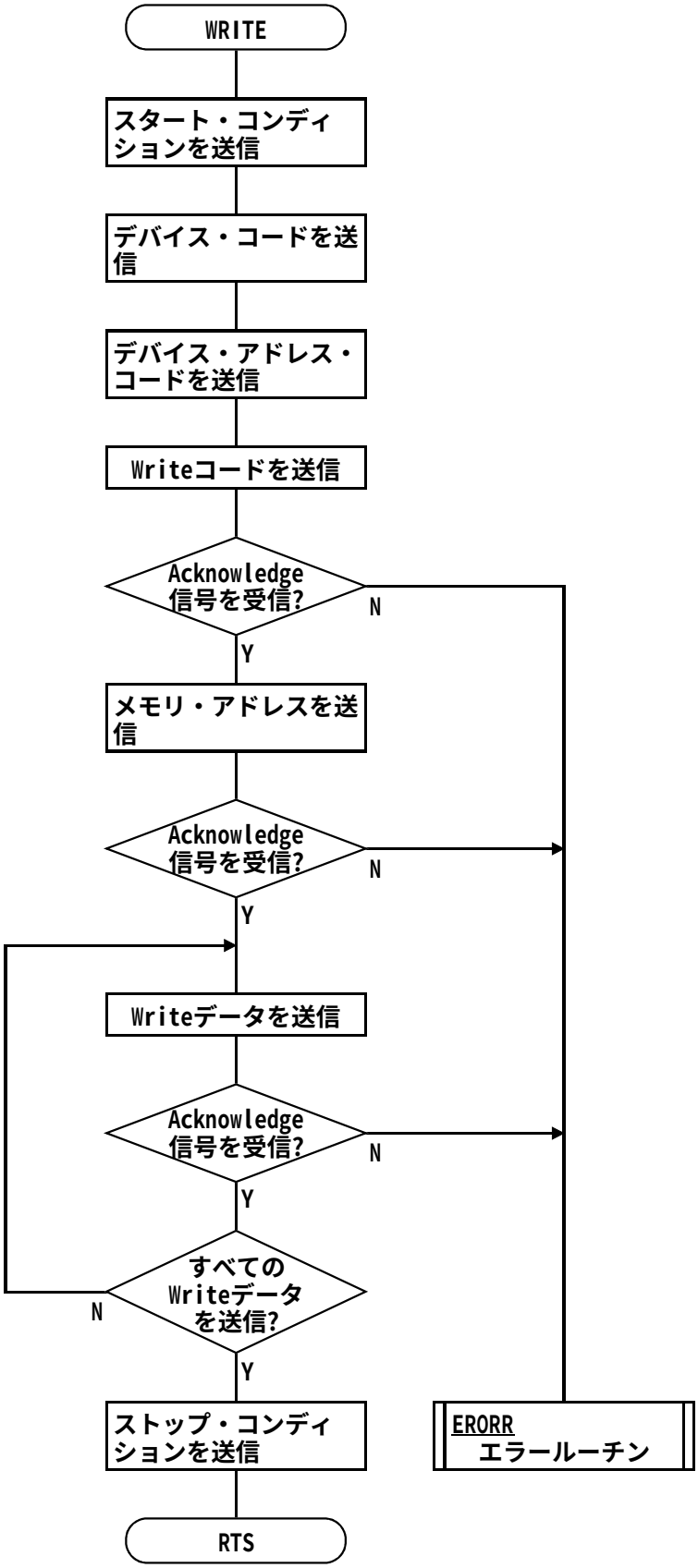


（c）エラールーチン



フローチャート

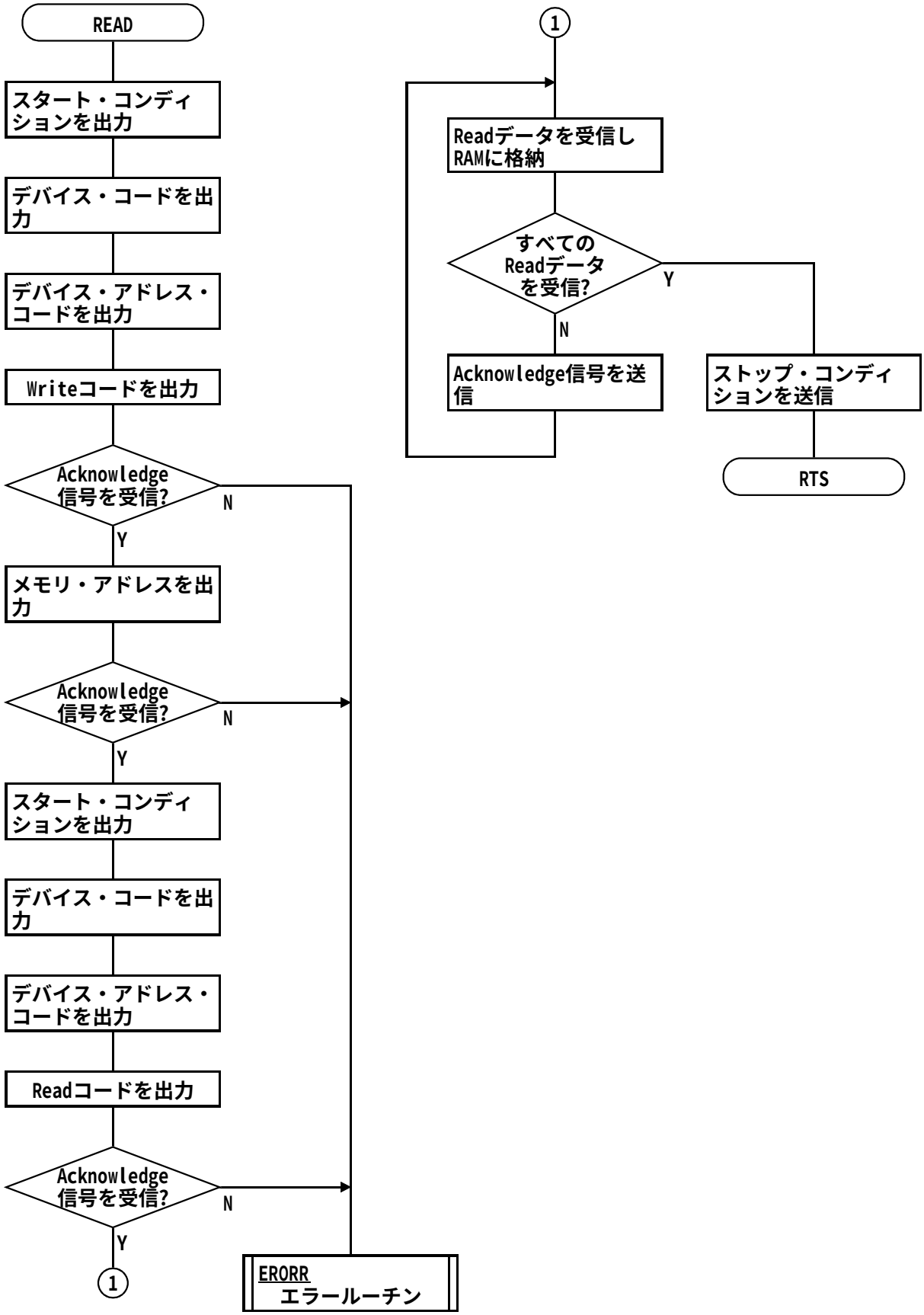
(d) 書き込み



EEPROMとのインタフェース	使用機能	I/Oポート

フローチャート

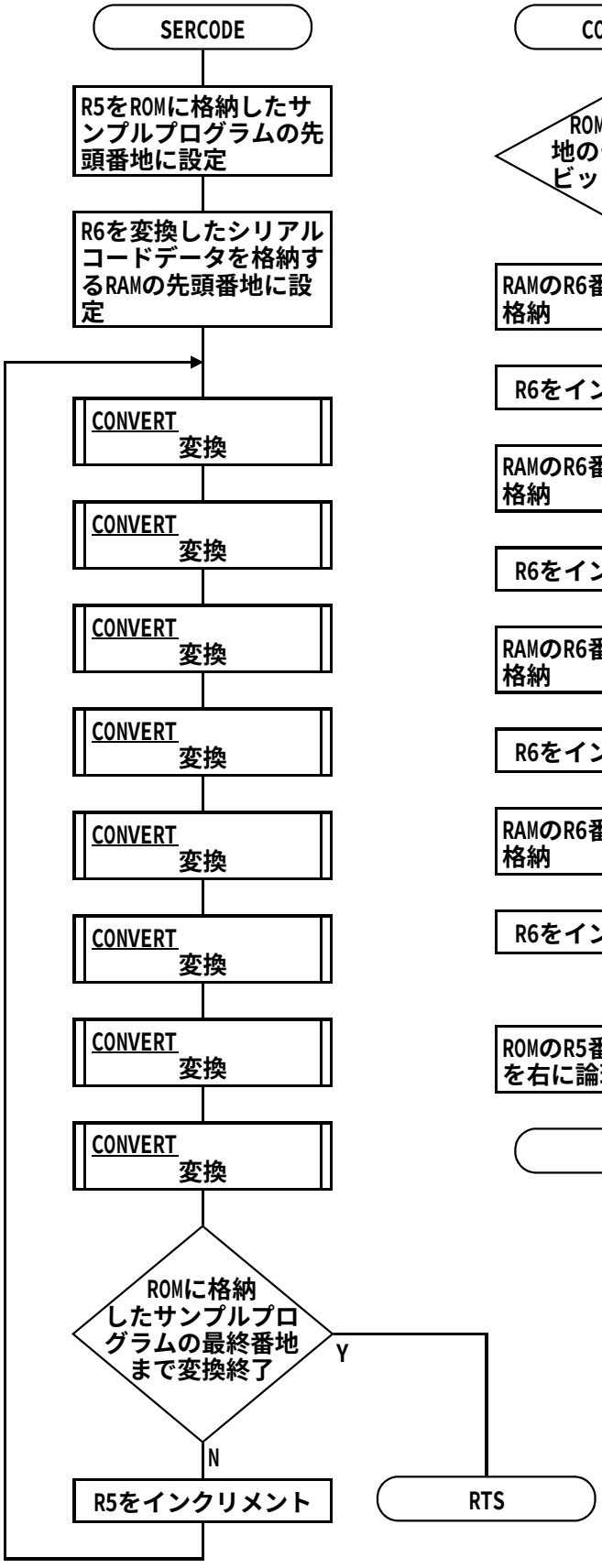
(e) 読み出し



EEPROMとのインタフェース	使用機能	I/Oポート

フローチャート

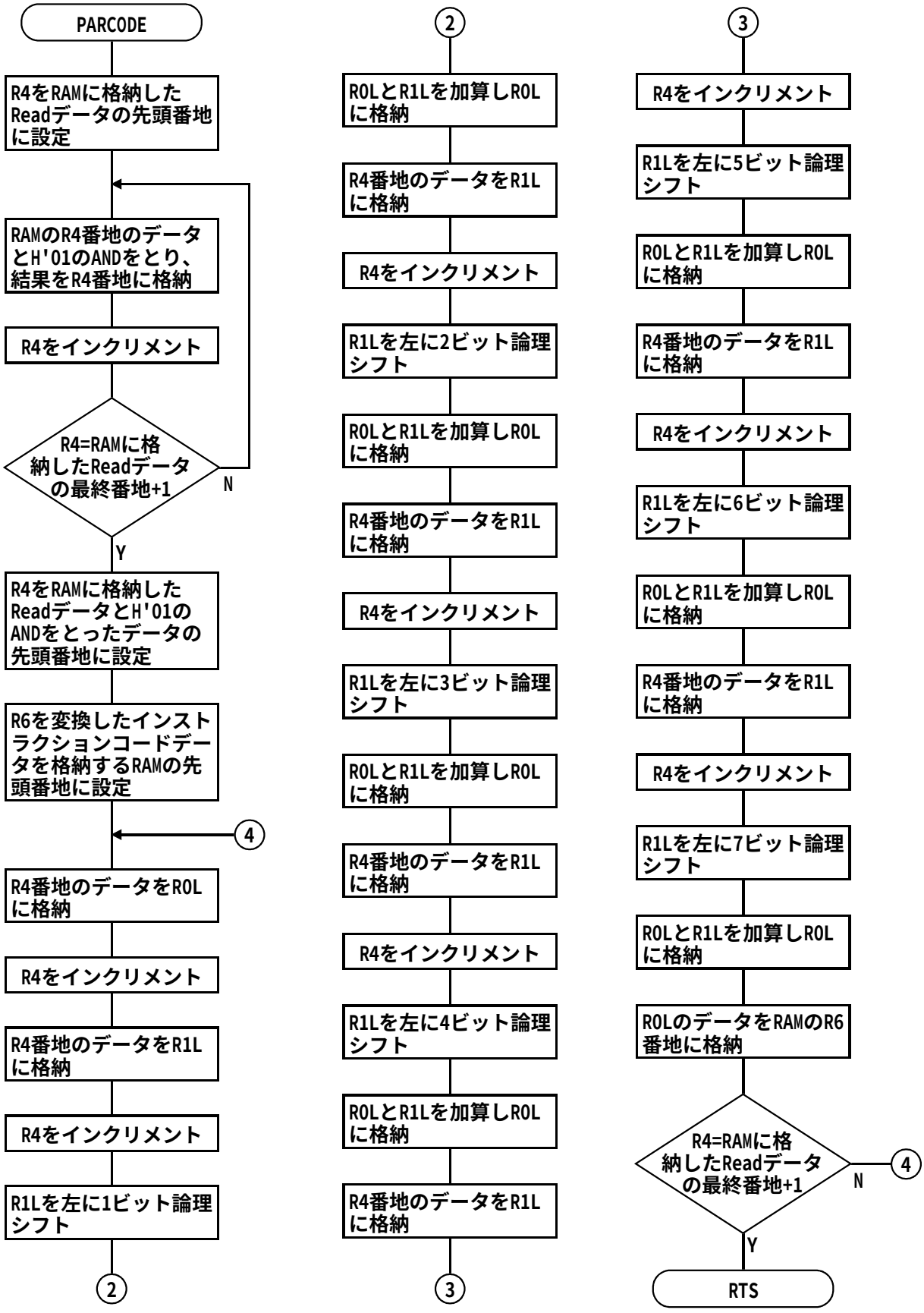
（f）シリアルコード変換



（g）変換

フローチャート

(h) インストラクションコード変換



EEPROMとのインタフェース	使用機能	I/Oポート

プログラムリスト

```
;*************************************************************
;
;               H8/3644 Application Note
;
;               'EEPROM Write & Read Control'
;
;               Function
;               : I/O Port Base
;
;               External Clock : 10MHz
;               Internal Clock : 5MHz
;
;*************************************************************
;
;*************************************************************
;
                .cpu    300L
;*************************************************************
;
;*************************************************************
;               Symbol Defnition
;*************************************************************
;
PDR2            .equ            H'FFD5          ;Port Data Register 2
PCR2            .equ            H'FFE5          ;Port Control Register 2
;*************************************************************
;               Ram Allocation
;*************************************************************
;
SERAREA         .equ            H'FB80          ;Serial code area (H'FB80-H'FEBF: SERSIZE *) 4 pulse)
COUNTER         .equ            H'FEC0          ;Counter (@H'FEC0 =) H'1A: 26 bytes)
SPLPGM          .equ            H'FF00          ;Sample program area (H'FF00-H'FF19)
STACK           .equ            H'FF80          ;Stack Pointer
SPLSIZE         .equ            H'1A            ;Sample program size (  =) 26 bytes)
SERSIZE         .equ            H'D0            ;Serial code size (  =) SPLSIZE *) 8-bit)
;*************************************************************
;               Vector Address
;*************************************************************
;
                .org            H'0000
                .data.w         MAIN            ;Reset Interrupt
                .org            H'0008
                .data.w         MAIN            ;IRQ0 Interrupt
                .data.w         MAIN            ;IRQ1 Interrupt
                .data.w         MAIN            ;IRQ2 Interrupt
                .data.w         MAIN            ;IRQ3 Interrupt
                .data.w         MAIN            ;INT0 - INT7 Interrupt
                .data.w         H'0014
                .data.w         MAIN            ;Timer A Interrupt
                .data.w         MAIN            ;Timer B1 Interrupt
                .data.w         H'0020
                .data.w         MAIN            ;Timer X Interrupt
                .data.w         MAIN            ;Timer V Interrupt
                .org            H'0026
                .data.w         MAIN            ;Sci1 Interrupt
                .org            H'002A
                .data.w         MAIN            ;Sci3 Interrupt
                .data.w         MAIN            ;A/D Converter Interrupt
                .data.w         MAIN            ;Sleep Interrupt
;*************************************************************
;               Main Program
```

EEPROMとのインタフェース	使用機能	I/Oポート

プログラムリスト

```
;*********************************************************
;
;
                .org            H'1000
MAIN            MOV.W           #STACK,SP       ;Initialize Stack Pointer
                ORC             #H'80,CCR       ;Interrupt Disable
                BSR             SERCODE         ;Convert sample program data to serial code
                JSR             @WRITE          ;Write EEPROM
                MOV.W           #1,R4           ;10ms wait as tWC spec. of EEPROM
                MOV.W           #H'208C,R5
TWCWAIT         SUB.W           R4,R5
                BNE             TWCWAIT
                JSR             @READ           ;Read EEPROM
                JSR             @PARCODE        ;Convert Serial code to 8-bit sample program data
                MOV.W           #1,R0           ;Load software time data of LED on/off period
                MOV.W           #0,R1           ;as 262ms
                MOV.W           #1,R2
                JMP             @SPLPGM         ;Execute sample program at Internal RAM
;
ERROR           BRA             ERROR           ;ERROR area
;*********************************************************
;
;               Convert sample program to serial code
;*********************************************************
;
;
SERCODE         .equ            $
                MOV.B           #0,R0L          ;Load SCL =1, SDA =1 data
                MOV.B           #1,R1L          ;Load SCL =1, SDA =0 data
                MOV.B           #2,R2L          ;Load SCL =0, SDA =1 data
                MOV.B           #3,R3L          ;Load SCL =0, SDA =0 data
                MOV.W           #SAMPLE,R5      ;Load sample program address
                MOV.W           #SERAREA,R6     ;Load serial code address
                MOV.B           #SPLSIZE,R4L    ;Load sample program size
                MOV.B           R4L,@COUNTER
NEXTCON         MOV.B           @R5+,R4L        ;Load sample program data
                BSR             CONVERT         ;Convert sample program data to serial code bit0
                BSR             CONVERT         ;Convert sample program data to serial code bit1
                BSR             CONVERT         ;Convert sample program data to serial code bit2
                BSR             CONVERT         ;Convert sample program data to serial code bit3
                BSR             CONVERT         ;Convert sample program data to serial code bit4
                BSR             CONVERT         ;Convert sample program data to serial code bit5
                BSR             CONVERT         ;Convert sample program data to serial code bit6
                BSR             CONVERT         ;Convert sample program data to serial code bit7
                MOV.B           @COUNTER,R4L
                DEC             R4L
                MOV.B           R4L,@COUNTER
                CMP.B           #0,R4L          ;@COUNTER=0?
                BNE             NEXTCON         ;No.
                RTS
;
CONVERT         .equ            $
                BTST            #0,R4L          ;If sample program data bitn is "0",
                BEQ             BITEQU0         ;it branch to BITEQU0
                MOV.B           R2L,@R6         ;Store SCL & SDA data as follows.
                ADDS            #1,R6           ;SCL = 0110
                MOV.B           R0L,@R6         ;SDA = 1111
                ADDS            #1,R6
                MOV.B           R0L,@R6
                ADDS            #1,R6
                MOV.B           R2L,@R6
                ADDS            #1,R6
                BRA             BITn
```

EEPROMとのインタフェース	使用機能	I/Oポート

プログラムリスト

```
BITEQU0         MOV.B           R3L,@R6         ;Store SCL & SDA data as follows.
                ADDS            #1,R6           ;SCL = 0110
                MOV.B           R1L,@R6         ;SDA = 0000
                ADDS            #1,R6
                MOV.B           R1L,@R6
                ADDS            #1,R6
                MOV.B           R3L,@R6
                ADDS            #1,R6
BITn            SHLR            R4L
                RTS
;*********************************************************
;               Write (Use Page Write Operation)
;*********************************************************
;
WRITE           .equ            $
                MOV.B           R0L,@PDR2       ;SCL = "0", SDA = "1"
                MOV.B           R2L,@PCR2
;
                JSR             @RW_start       ;Output "0" of Device address start bit
                JSR             @RW_H           ;Output "1" of Device address
                JSR             @RW_L           ;Output "0" of Device address
                JSR             @RW_H           ;Output "1" of Device address
                JSR             @RW_L           ;Output "0" of Device address
                JSR             @RW_L           ;Output "0" of Device address code
                JSR             @RW_L           ;Output "0" of Device address code
                JSR             @RW_L           ;Output "0" of Device address code
                JSR             @RW_L           ;Output "0" of Device address write bit
                JSR             @RW_ack         ;Input "0" of /ACK
;
                JSR             @RW_L           ;Output "0" of Memory address a7
                JSR             @RW_L           ;Output "0" of Memory address a6
                JSR             @RW_L           ;Output "0" of Memory address a5
                JSR             @RW_L           ;Output "0" of Memory address a4
                JSR             @RW_L           ;Output "0" of Memory address a3
                JSR             @RW_L           ;Output "0" of Memory address a2
                JSR             @RW_L           ;Output "0" of Memory address a1
                JSR             @RW_L           ;Output "0" of Memory address a0
                JSR             @RW_ack         ;Input "0" of /ACK
;
                MOV.W           #SERAREA,R4     ;Load serial code of sample program
                MOV.W           #SPLSIZE,R5
WRLOOP          JSR             @WR_data        ;Output Write data D0
                JSR             @WR_data        ;Output Write data D1
                JSR             @WR_data        ;Output Write data D2
                JSR             @WR_data        ;Output Write data D3
                JSR             @WR_data        ;Output Write data D4
                JSR             @WR_data        ;Output Write data D5
                JSR             @WR_data        ;Output Write data D6
                JSR             @WR_data        ;Output Write data D7
                JSR             @RW_ack         ;Input "0" of /ACK
                DEC             R5L             ;Counter=0?
                BEQ             WREND           ;No.
                BRA             WRLOOP
WREND           JSR             @RW_stop        ;Output stop bit
                RTS
;*********************************************************
;               Read (Random &) Sequential Operation of EEPROM)
;*********************************************************
;
READ            .equ            $
```

EEPROMとのインタフェース	使用機能	I/Oポート

プログラムリスト

```
                MOV.W           #SERAREA,R4)    ;Load) serial) code) address
                MOV.W           #SPLSIZE,R5)    ;Load) count) data) () =) 26) bytes)
                MOV.B)))        #0,R0L)))))     ;Load) SCL) =) 1,) SDA) =) 1) data
                MOV.B)))        #1,R1L)))))     ;Load) SCL) =) 1,) SDA) =) 0) data
                MOV.B)))        #2,R2L)))))     ;Load) SCL) =) 0,) SDA) =) 1) data
                MOV.B)))        #3,R3L)))))     ;Load) SCL) =) 0,) SDA) =) 0) data
                MOV.B           R0L,@PDR2
                MOV.B           R2L,@PCR2)))    ;SCL) =) "0",) SDA) =) "1"
;
                JSR             @RW_start)))    ;Output) "0") of) Device) address) start) bit
                JSR             @RW_H           ;Output) "1") of) Device) address
                JSR             @RW_L           ;Output) "0") of) Device) address
                JSR             @RW_H           ;Output) "1") of) Device) address
                JSR             @RW_L           ;Output) "0") of) Device) address
                JSR             @RW_L           ;Output) "0") of) Device) address) code
                JSR             @RW_L           ;Output) "0") of) Device) address) code
                JSR             @RW_L           ;Output) "0") of) Device) address) code
                JSR             @RW_L           ;Output) "0") of) Device) address) write) bit
                JSR             @RW_ack                 ;Input) "0") of) /ACK
;
                BSR             RW_L            ;Output) "0") of) Memory) address) a7
                BSR             RW_L            ;Output) "0") of) Memory) address) a6
                BSR             RW_L            ;Output) "0") of) Memory) address) a5
                BSR             RW_L            ;Output) "0") of) Memory) address) a4
                BSR             RW_L            ;Output) "0") of) Memory) address) a3
                BSR             RW_L            ;Output) "0") of) Memory) address) a2
                BSR             RW_L            ;Output) "0") of) Memory) address) a1
                BSR             RW_L            ;Output) "0") of) Memory) address) a0
                BSR             RW_ack          ;Input) "0") of) /ACK
;
                BSR             RW_start))))    ;Output) "0") of) Device) address) start) bit
                BSR             RW_H            ;Output) "1") of) Device) address
                BSR             RW_L            ;Output) "0") of) Device) address
                BSR             RW_H            ;Output) "1") of) Device) address
                BSR             RW_L            ;Output) "0") of) Device) address
                BSR             RW_L            ;Output) "0") of) Device) address) code
                BSR             RW_L            ;Output) "0") of) Device) address) code
                BSR             RW_L            ;Output) "0") of) Device) address) code
                BSR             RW_H            ;Output) "1") of) Device) address) read) bit
                BSR             RW_ack          ;Input) "0") of) /ACK
;
RDLOOP          JSR             @RD_data))))    ;Input) Read) data) D0
                JSR             @RD_data))))    ;Input) Read) data) D1
                JSR             @RD_data))))    ;Input) Read) data) D2
                JSR             @RD_data))))    ;Input) Read) data) D3
                JSR             @RD_data))))    ;Input) Read) data) D4
                JSR             @RD_data)))))   ;Input) Read) data) D5
                JSR             @RD_data))))    ;Input) Read) data) D6
                JSR             @RD_data))))    ;Input) Read) data) D7
                BSR             RW_L)))))))     ;Output) "0") of) /ACK) bit
                DEC)))))        R5L
                BEQ)))))        SERDEND
                BRA)))))        RDLOOP
SERDEND)        .equ            $
                MOV.B           R0L,@PDR2
                MOV.B           R3L,@PCR2)))    ;SCL) =) "0",) SDA) =) "0"
                BSR             RW_stop)))))    ;Output) stop) bit
                RTS
;*********************************************************
;))))            Subroutine
```

EEPROMとのインタフェース	使用機能	I/Oポート

プログラムリスト

```
;*********************************************************
;
;
RW_start        .equ            $
                MOV.B           R0L,@PDR2       ;Output "0" of Start bit
                MOV.B           R2L,@PCR2       ;SCL = "0", SDA = "1"
                MOV.B           R0L,@PDR2
                MOV.B           R0L,@PCR2       ;SCL = "1", SDA = "1"
                MOV.B           R0L,@PDR2
                MOV.B           R1L,@PCR2       ;SCL = "1", SDA = "0"
                MOV.B           R0L,@PDR2
                MOV.B           R3L,@PCR2       ;SCL = "0", SDA = "0"
                RTS
;
RW_H            .equ            $
                MOV.B           R0L,@PDR2       ;Output "1" of Device/Memory address
                MOV.B           R2L,@PCR2       ;SCL = "0", SDA = "1"
                MOV.B           R0L,@PDR2
                MOV.B           R0L,@PCR2       ;SCL = "1", SDA = "1"
                MOV.B           R0L,@PDR2
                MOV.B           R0L,@PCR2       ;SCL = "1", SDA = "1"
                MOV.B           R0L,@PDR2
                MOV.B           R2L,@PCR2       ;SCL = "0", SDA = "1"
                RTS
;
RW_L            .equ            $
                MOV.B           R0L,@PDR2       ;Output "0" of Device/Memory address or /ACK
                MOV.B           R3L,@PCR2       ;SCL = "0", SDA = "0"
                MOV.B           R0L,@PDR2
                MOV.B           R1L,@PCR2       ;SCL = "1", SDA = "0"
                MOV.B           R0L,@PDR2
                MOV.B           R1L,@PCR2       ;SCL = "1", SDA = "0"
                MOV.B           R0L,@PDR2
                MOV.B           R3L,@PCR2       ;SCL = "0", SDA = "0"
                RTS
;
RW_ack          .equ            $
                MOV.B           R0L,@PDR2       ;Input "0" of /ACK bit
                MOV.B           R2L,@PCR2
                MOV.B           R0L,@PDR2
                MOV.B           R0L,@PCR2
                BTST            #0,@PDR2        ;/ACK=0?
                BEQ             ACKOK           ;Yes.
                JMP             @ERROR
ACKOK           MOV.B           R0L,@PDR2
                MOV.B           R3L,@PCR2       ;SCL = "0", SDA = "0"
                RTS
;
RW_stop                 .equ            $
                MOV.B           R0L,@PDR2       ;Output "1" of Stop bit
                MOV.B           R1L,@PCR2       ;SCL = "1", SDA = "0"
                MOV.B           R0L,@PDR2
                MOV.B           R0L,@PCR2       ;SCL = "1", SDA = "1"
                MOV.B           R0L,@PDR2
                MOV.B           R2L,@PCR2       ;SCL = "0", SDA = "1"
                MOV.B           R0L,@PDR2
                MOV.B           R2L,@PCR2       ;SCL = "0", SDA = "1"
                MOV.B           R0L,@PDR2
                MOV.B           R0L,@PCR2       ;SCL = "1", SDA = "1"
                RTS
;
```

EEPROMとのインタフェース	使用機能	I/Oポート

プログラムリスト

```
WR_data    .equ      $
           MOV.B     @R4+,R1L          ;Output write data bitn
           MOV.B     R1L,@PCR2         ;SCL =>"0", SDA =>"Dn Output"
           MOV.B     @R4+,R1L
           MOV.B     R1L,@PCR2         ;SCL =>"1", SDA =>"Dn Output"
           MOV.B     @R4+,R1L
           MOV.B     R1L,@PCR2         ;SCL =>"1", SDA =>"Dn Output"
           MOV.B     @R4+,R1L
           MOV.B     R1L,@PCR2         ;SCL =>"0", SDA =>"Dn Output"
           RTS
;
RD_data    .equ      $
           MOV.B     R0L,@PDR2         ;Input read data bitn
           MOV.B     R2L,@PCR2         ;SCL =>"0", SDA =>"Dn input"
           MOV.B     R0L,@PDR2
           MOV.B     R0L,@PCR2         ;SCL =>"1", SDA =>"Dn input"
           MOV.B     @PDR2,R6L
           MOV.B     R6L,@R4           ;Store serial code at Internal RAM
           ADDS      #1,R4
           MOV.B     R0L,@PDR2
           MOV.B     R2L,@PCR2         ;SCL =>"0", SDA =>"Dn input"
           RTS
;
PARCODE    .equ      $
           MOV.W     #SERAREA,R4       ;Load serial code address
           MOV.B     #SERSIZE,R5L
ANDDATA    MOV.B     @R4,R0L           ;AND.B #1,@SERAREA(H'FB80-FEBF)
           AND       #H'01,R0L
           MOV.B     R0L,@R4
           ADDS      #1,R4
           DEC       R5L
           BNE       ANDDATA
           MOV.W     #SERAREA,R4       ;Load serial code address
           MOV.W     #SPLSIZE,R5
           MOV.W     #SPLPGM,R6        ;Load execution sample program address
SERPAR     MOV.B     @R4+,R0L          ;Load serial code bit0
           MOV.B     @R4+,R1L          ;Load serial code bit1
           SHLL      R1L               ;Convert bit1 code
           ADD.B     R1L,R0L           ;Calculate Bit1,0 code
           MOV.B     @R4+,R1L          ;Load serial code bit2
           SHLL      R1L               ;Convert bit2 code
           SHLL      R1L
           ADD.B     R1L,R0L           ;Calculate Bit2-0 code
           MOV.B     @R4+,R1L          ;Load serial code bit3
           SHLL      R1L               ;Convert bit3 code
           SHLL      R1L
           SHLL      R1L
           ADD.B     R1L,R0L           ;Calculate Bit3-0 code
           MOV.B     @R4+,R1L          ;Load serial code bit4
           SHLL      R1L               ;Convert bit4 code
           SHLL      R1L
           SHLL      R1L
           SHLL      R1L
           ADD.B     R1L,R0L           ;Calculate Bit4-0 code
           MOV.B     @R4+,R1L          ;Load serial code bit5
           SHLL      R1L               ;Convert bit5 code
           SHLL      R1L
           SHLL      R1L
           SHLL      R1L
           SHLL      R1L
```

プログラムリスト

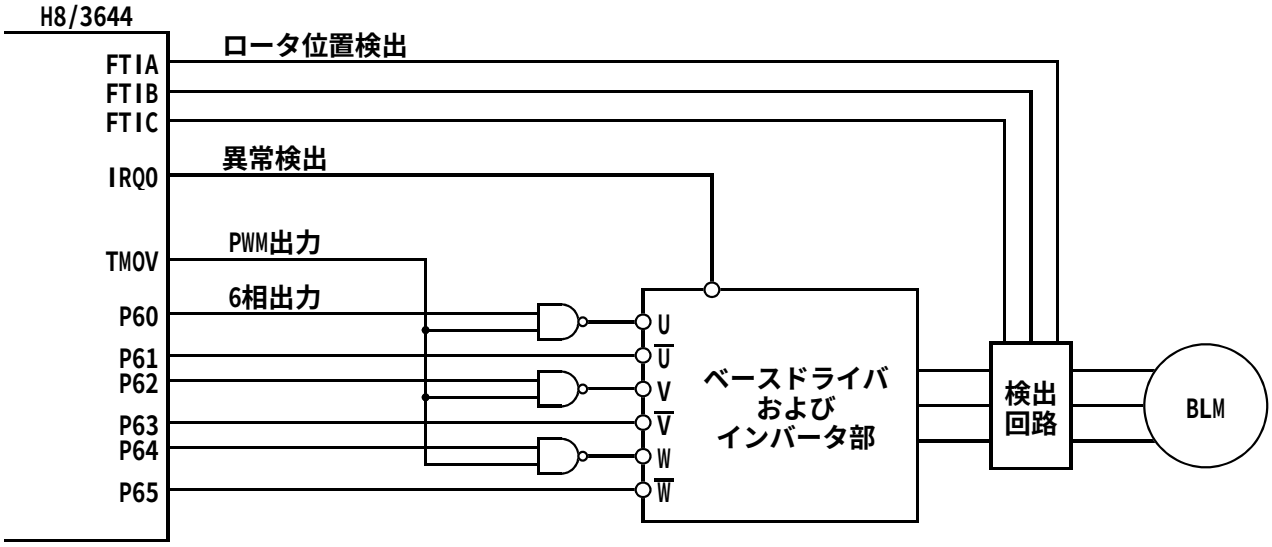
```
                ADD.B           R1L,R0L             ;Calculate Bit5-0 code
                MOV.B           @R4+,R1L            ;Load serial code bit6
                SHLL            R1L                 ;Convert bit6 code
                SHLL            R1L
                SHLL            R1L
                SHLL            R1L
                SHLL            R1L
                SHLL            R1L
                ADD.B           R1L,R0L             ;Calculate Bit6-0 code
                MOV.B           @R4+,R1L            ;Load serial code bit7
                SHLL            R1L                 ;Convert bit7 code
                SHLL            R1L
                SHLL            R1L
                SHLL            R1L
                SHLL            R1L
                SHLL            R1L
                SHLL            R1L
                ADD.B           R1L,R0L             ;Calculate Bit7-0 code
                MOV.B           R0L,@R6
                ADDS            #1,R6
                DEC             R5L
                BNE             SERPAR
                RTS
;************************************************************
;               H8/3644 Sample program of LED Control
;************************************************************
;
;
                .org            H'2000
SAMPLE          BSET            #3,@H'FFEA          ;Initialize P73 Output Port
LEDCTL          BSET            #3,@H'FFDA          ;Turn on LED
                BSR             WAIT
                BCLR            #3,@H'FFDA          ;Turn off LED
                BSR             WAIT
                BRA             LEDCTL
WAIT            SUB.W           R0,R1
                MULXU           R0L,R2
                BNE             WAIT
                RTS
;
                .end
```

2.2 DCブラシレスモータ制御

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

仕様

(1) 図1に示すように、H8/3644を使用しDCブラシレスモータを制御します。
(2) DCブラシレスモータは、ロータの磁極位置信号を検出し、各位置信号にあった回転磁界をポートから出力することで回転します。
(3) H8/3644内蔵タイマにより、PWM波形を生成し、チョッピング制御します。
(4) DCブラシレスモータの磁極位置を検出して、I/Oポートで6相のレベルを切替え、ベースドライバへ出力します。



【注】記号説明

BLM	：	DCブラスレスモータ（120°通電型）
U	：	U相
$\overline{U}$	：	$\overline{U}$相
V	：	V相
$\overline{V}$	：	$\overline{V}$相
W	：	W相
$\overline{W}$	：	$\overline{W}$相
FTIA	：	インプットキャプチャA入力端子
FTIB	：	インプットキャプチャB入力端子
FTIC	：	インプットキャプチャC入力端子
IRQ0	：	$\overline{IRQ_0}$入力端子
TMOV	：	タイマV波形出力端子
P60-P65	：	ポート6_0～ポート6_5出力端子

図1 DCブラシレスモータ制御

考え方

(1) DCブラスレスモータを制御するためのPWM波形をタイマで生成し、I/Oポートより出力します。
(2) モータの回り始めは、励磁する相を一定周期ごとに順に切換え、ポート出力します。
(3) 励磁する相の切換えを60回行なった後、モータより出力されるロータ位置信号を検出して制御する方法に遷移します。
(4) モータより出力されるロータ位置検出信号は、検出回路を介してタイマXのインプットキャプチャ端子で取り込み、割込みを発生させます。
(5) 割込みにより、回転磁界を切替え、励磁する相をチョッピング制御します。

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

使用機能説明

(1) 図2に示すようにH8/3644のタイマX（インプットキャプチャ、アウトプットコンペア機能）、タイマV（コンペアマッチ機能）、I/Oポート（ポート6機能）、IRQ（外部割込み機能）の各機能を割付け，DCブラスレスモータ制御を行ないます。

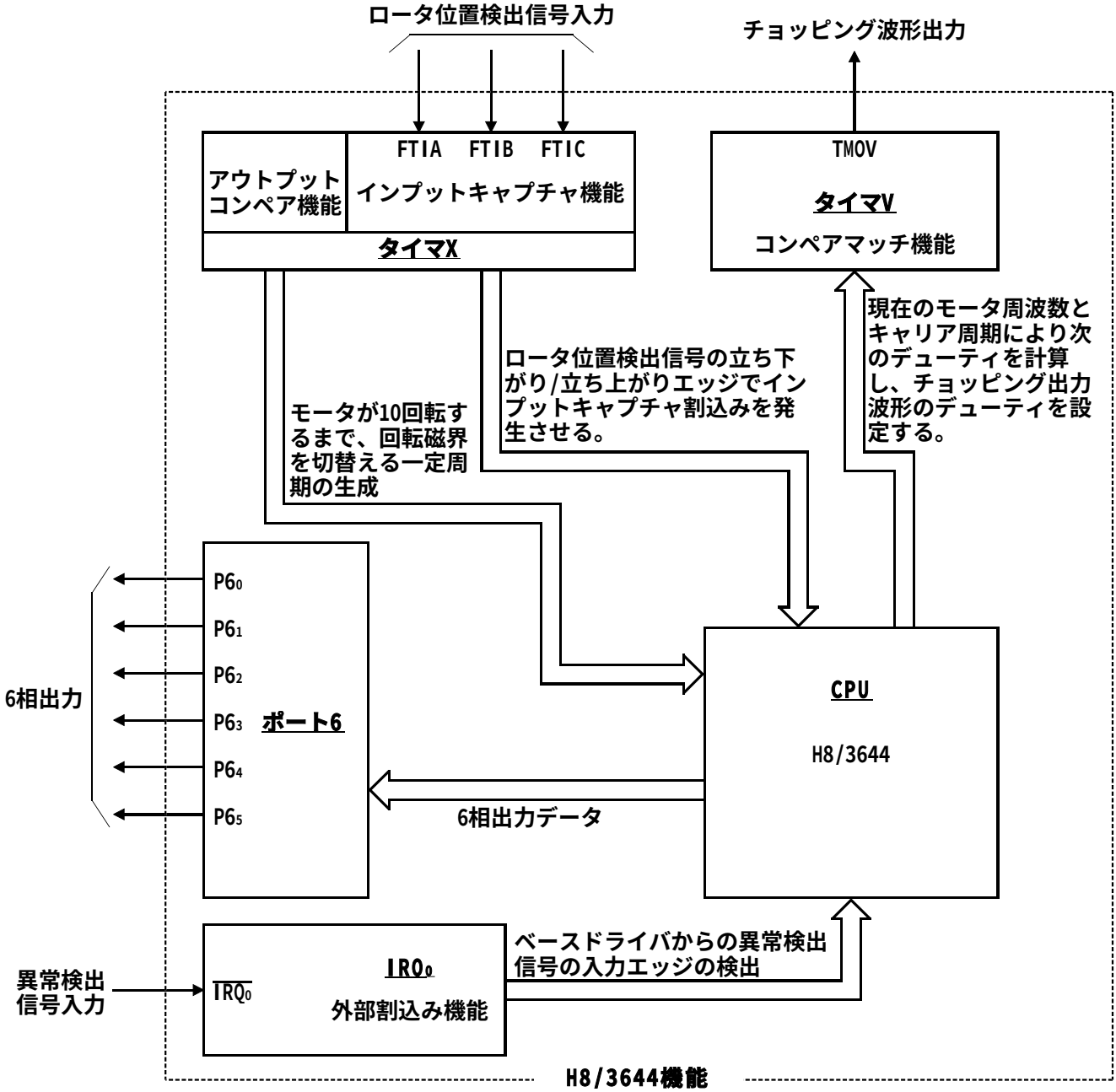


図2 DCブラスレスモータ制御ブロック図

以下にH8/3644の機能割付けについて説明します。

- タイマXインプットキャプチャ機能：ロータ位置検出信号の立ち上がり/立ち下がりエッジを検出し、CPUに割込みを要求します。
- タイマXアウトプットコンペア機能：モータが10回転するまで、回転磁界切換え周期（モータの回転周波数の60°分に相当する時間）ごとにCPUに割込みを要求します。
- タイマVコンペアマッチ機能：ドライバがONする際にチョッピング制御を行なうためのチョッピング波形の生成を行い、TMOV端子より出力します。
- ポート6機能：ドライバへ出力する6相のデータを出力します。
- IRQ外部割込み機能：ドライバからの異常検出信号によりモータを停止させます。

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

使用機能説明

(2) 以下に各機能ごとに説明します。

(a) タイマXは、ロータ位置検出信号の立ち上がり/立ち下がりエッジを検出し割込みを発生させるインプットキャプチャ機能と、初期モータ制御時の回転磁界切換え周期を測定するためのアウトプットコンペア機能で使用します。

以下にタイマXのインプットキャプチャ機能とアウトプットコンペア機能で共通に使用する機能について説明します。

・システムクロック（φ）は、10MHzのOSCクロックを2分周した5MHzのクロックで、CPUおよび周辺機能を動作させるための基準クロックです。
・プリスケーラS（PSS）は、φを入力とする13ビットのカウンタで、1サイクルごとにカウントアップします。
・フリーランニングカウンタ（FRC）は、16ビットのリード/ライト可能なアップカウンタで、入力するクロックはシステムクロックの2分周、8分周、32分周および外部クロックの計4種のクロックより選択可能です。FRCの入力クロックはTCRXにより設定します。本タスク例では、FRCの入力クロックにシステムクロックの2分周のクロックを選択しています。
・タイマコントロールレジスタX（TCRX）は、8ビットのリード/ライト可能なレジスタで、インプットキャプチャ入力A～Cの入力エッジの選択、FRCの入力クロックの選択を行ないます。
・タイマコントロール/ステータスレジスタ（TCSRX）は、8ビットのレジスタで、各割込み要求信号の制御を行ないます。
・タイマインタラプトイネーブルレジスタX（TIER）は、8ビットのリード/ライト可能なレジスタで各割込み要求の許可/禁止を制御します。本タスク例では、ICFAによる割込み要求（ICIA）、ICFBによる割込み要求（ICIB）、ICFCによる割込み要求（ICIC）を許可しています。

(b) タイマXインプットキャプチャ機能によりロータ位置検出信号の立ち上がり/立ち下がりエッジを検出し、割込みを発生させます。図3にインプットキャプチャ機能によるロータ位置検出信号の入力エッジによる割込み要求のブロック図を示します。以下にブロックについて説明します。

- インプットキャプチャ入力端子A、B、C（FTIA、FTIB、FTIC）は、ロータ位置検出信号の入力端子として機能します。
- インプットキャプチャレジスタA、B、C（ICRA、ICRB、ICRC）は、16ビットのリード専用のレジスタで、インプットキャプチャ入力信号の入力エッジが検出されると、その時のFRCの値がICRA、ICRB、ICRCに転送され、TCSRXのICFA、ICFB、ICFCが"1"にセットされます。このとき、TIERのICIAE、ICIBE、ICICEが"1"ならばCPUに割込みを要求します。

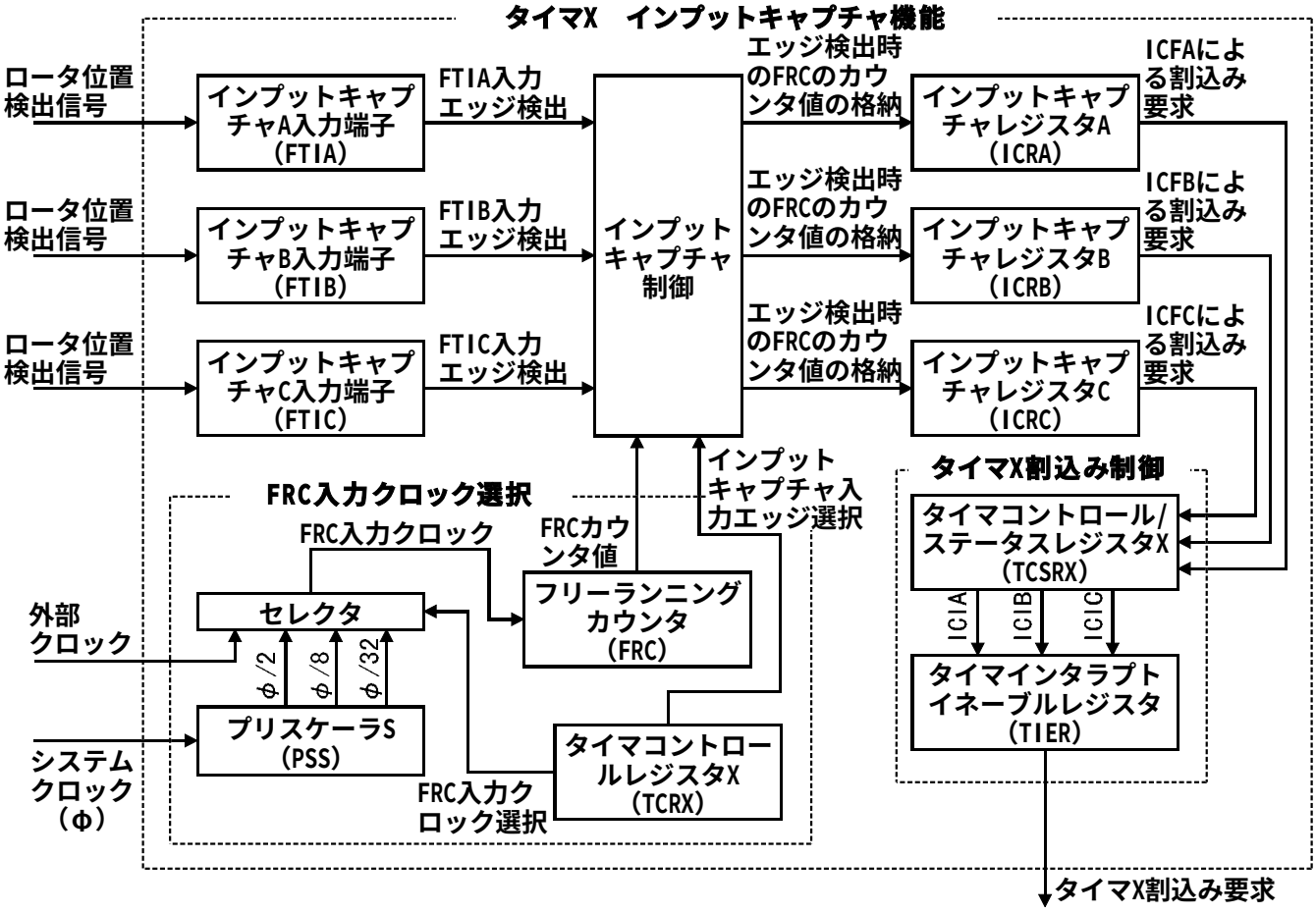


図3　インプットキャプチャ機能を使用したロータ位置検出信号の入力エッジ検出による割込み要求ブロック図

(c) タイマXアウトプットコンペア機能によりモータが10回転するまで、回転磁界切換え周期（モータの回転周波数の60°分に相当する時間）ごとにCPUに割込みを要求します。図4にタイマXアウトプットコンペア機能を使用した回転磁界切換え周期ごとの割込み要求のブロック図を示します。以下にブロック図について説明します。

・アウトプットコンペアレジスタA（OCRA）は、16ビットのリード/ライト可能なレジスタで2本のレジスタ（OCRA、OCRB）より構成されています。OCRAとOCRBのアドレスは同一になっており、これらの切換えは、TOCRのOCRSビットで行ないます。また、OCRAの内容はFRCと常に比較されており、両者の値が一致するとTCSRXのOCFAが"1"にセットされます。このとき、TIERのOCIAEが"1"ならばCPUに割込みを要求します。

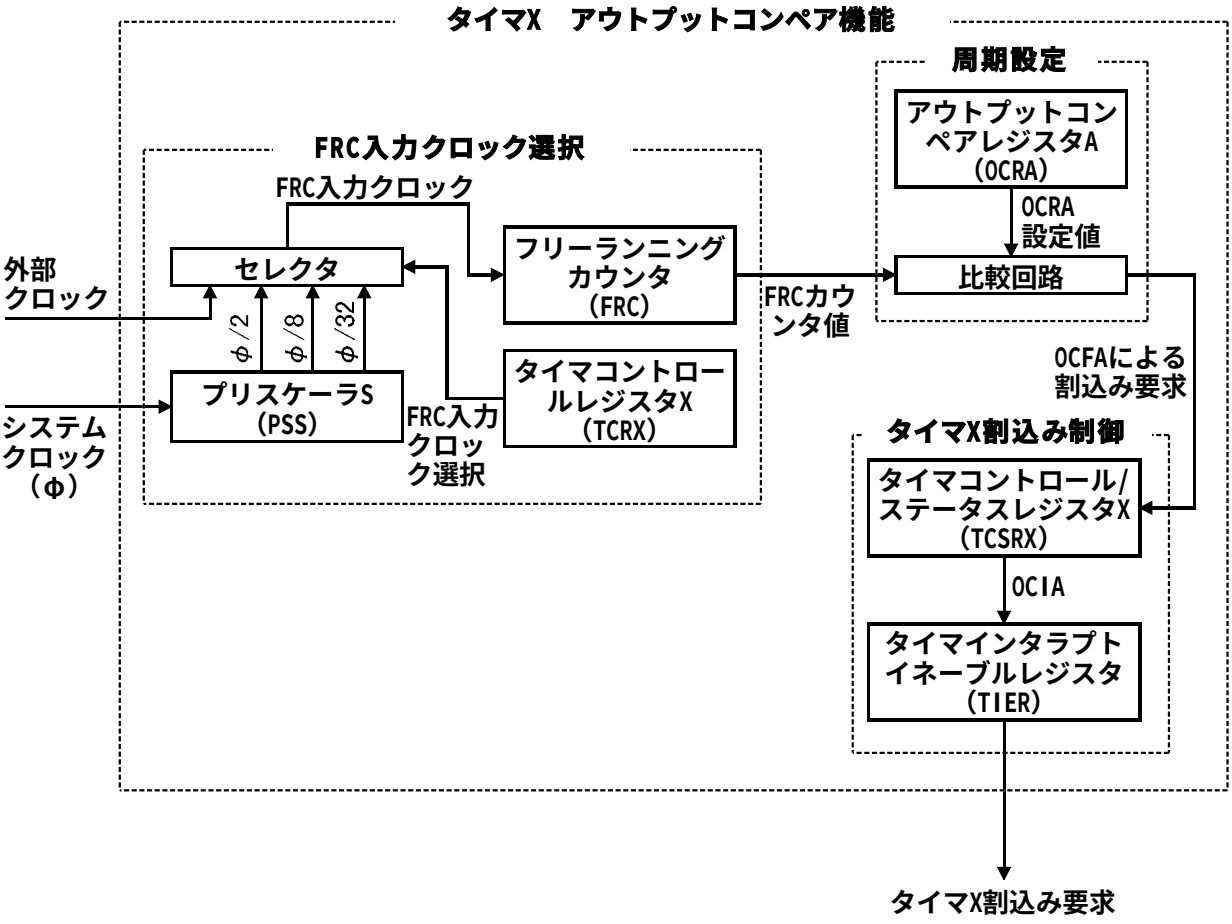


図4　タイマXアウトプットコンペア機能を使用した回転磁界切換え周期ごとの割込み要求ブロック図

(d) タイマVコンペア機能によりチョッピング波形を生成し、TMOV端子より出力します。図5にタイマVコンペアマッチ機能によるチョッピング波形出力のブロック図を示します。以下にブロック図について説明します。

・タイマカウンタV（TCNTV）は、8ビットのリード/ライト可能な8ビットカウンタで、入力する内部クロック/外部クロックによりカウントアップされます。入力するクロックは、TCRV0のCKS2～CKS0により選択します。TCNTVの値は、CPUから常にリード/ライトできます。TCNTVの値は、外部リセット入力信号またはコンペアマッチ信号A、Bにより、クリアすることができます。いずれの信号でクリアするかは、TCRV0のCCLR1、CCLR0により選択します。
・タイムコンスタントレジスタA、B（TCORA、B）は、8ビットのリード/ライト可能なレジスタです。TCORA、Bの内容はTCNTVと常に比較されており、両者の値が一致するとTCSRVのCMFA、Bが"1"にセットされます。このときTCRV0のCMIEA、Bが"1"ならばCPUに割込みを要求します。ただし、TCORA、BへのライトサイクルのT_3ステートでの比較は禁止されています。
・タイムコントロールレジスタV0（TCRV0）は、8ビットのリード/ライト可能なレジスタで、TCNTVの入力クロックの選択、TCNTVのクリア指定、および割込み要求の許可を行ないます。本タスク例では、CMFAによる割込み要求、CMFBによる割込み要求を許可し、それ以外の割込み要求を禁止しています。また、TCNTVのクリア指定はコンペアマッチAによるクリアを選択しています。TCNTVの入力クロックは、TCRV1のICKS0との組み合わせにより選択しています。
・タイマコントロール/ステータスレジスタV（TCSRV）は、8ビットのレジスタで、コンペアマッチフラグのセット、およびコンペアマッチ出力の制御を行ないます。本タスク例では、コンペアマッチBにより"0"出力、コンペアマッチAにより"1"出力を選択しています。
・タイマコントロールレジスタV1（TCRV1）は、8ビットのリード/ライト可能なレジスタで、TCNTVの入力クロックの選択を行ないます。
・タイマV出力端子（TMOV）によりチョッピング波形を出力します。

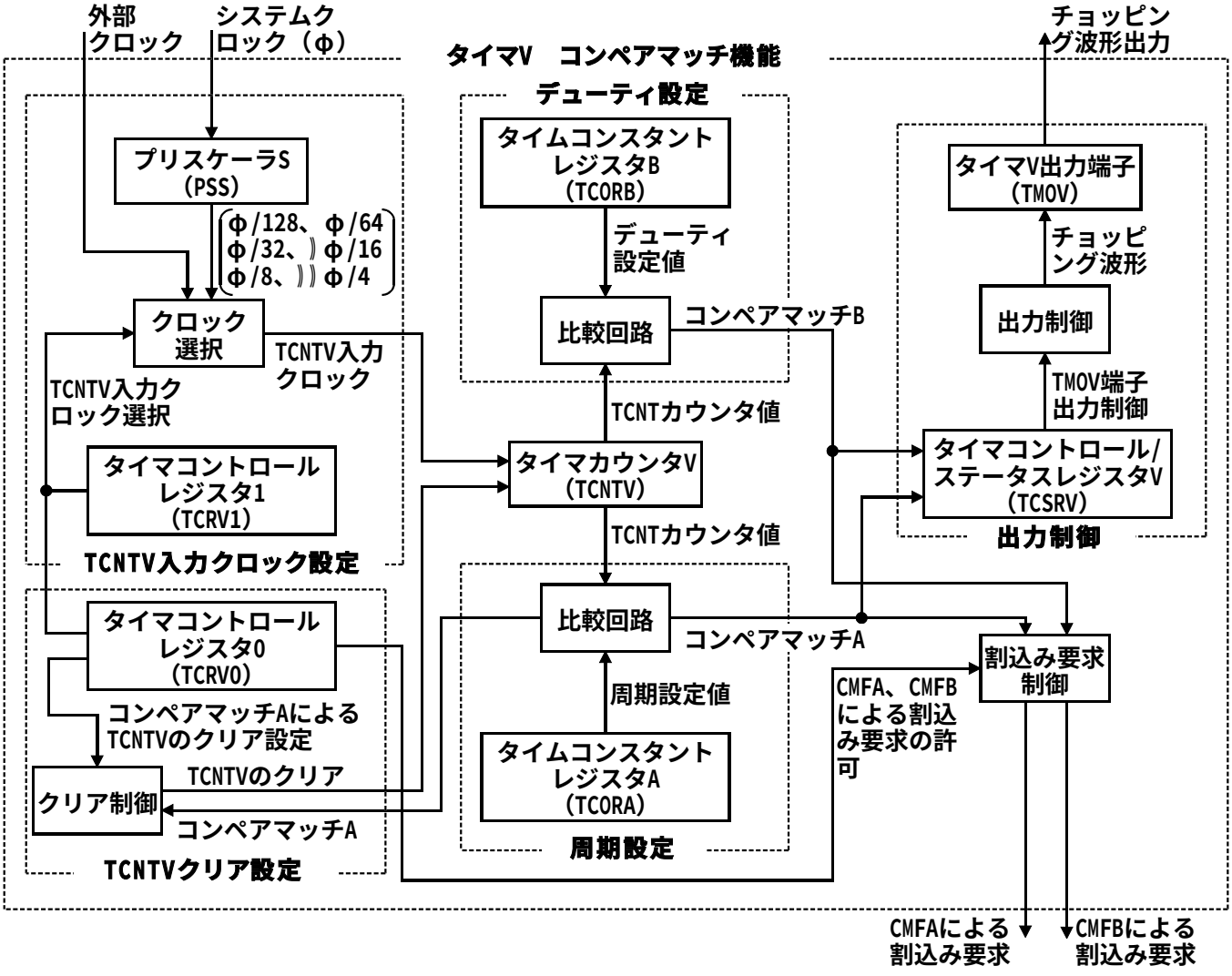


図5　タイマVコンペアマッチ機能を使用したチョッピング波形出力ブロック図

(e) ポート6機能により6相出力、およびモータ停止スイッチ入力の検出を行ないます。図6にポート6機能による6相出力、およびモータ停止信号の検出のブロック図を示します。以下にブロック図について説明します。

・ポート60端子（P60）は、ベースドライバのU相への出力端子として機能します。
・ポート61端子（P61）は、ベースドライバのU相への出力端子として機能します。
・ポート62端子（P62）は、ベースドライバのV相への出力端子として機能します。
・ポート63端子（P63）は、ベースドライバのV相への出力端子として機能します。
・ポート64端子（P64）は、ベースドライバのW相への出力端子として機能します。
・ポート65端子（P65）は、ベースドライバのW相への出力端子として機能します。
・ポートデータレジスタ6（PDR6）は、ポート6の各端子P60～P66のデータを格納する8ビットのレジスタです。PCR6が"1"のとき、ポート7のリードを行なうと、PDR6の値を直接リードします。そのため端子状態の影響を受けません。PCR6が"0"のとき、ポート6のリードを行なうと、端子状態が読み出されます。
・ポートコントロールレジスタ6（PCR6）は、ポート6の各端子P60～P66の入出力をビットごとに制御します。本タスク例では、端子P60～P65を出力端子機能に、端子P66～P67を入力端子機能に設定します。

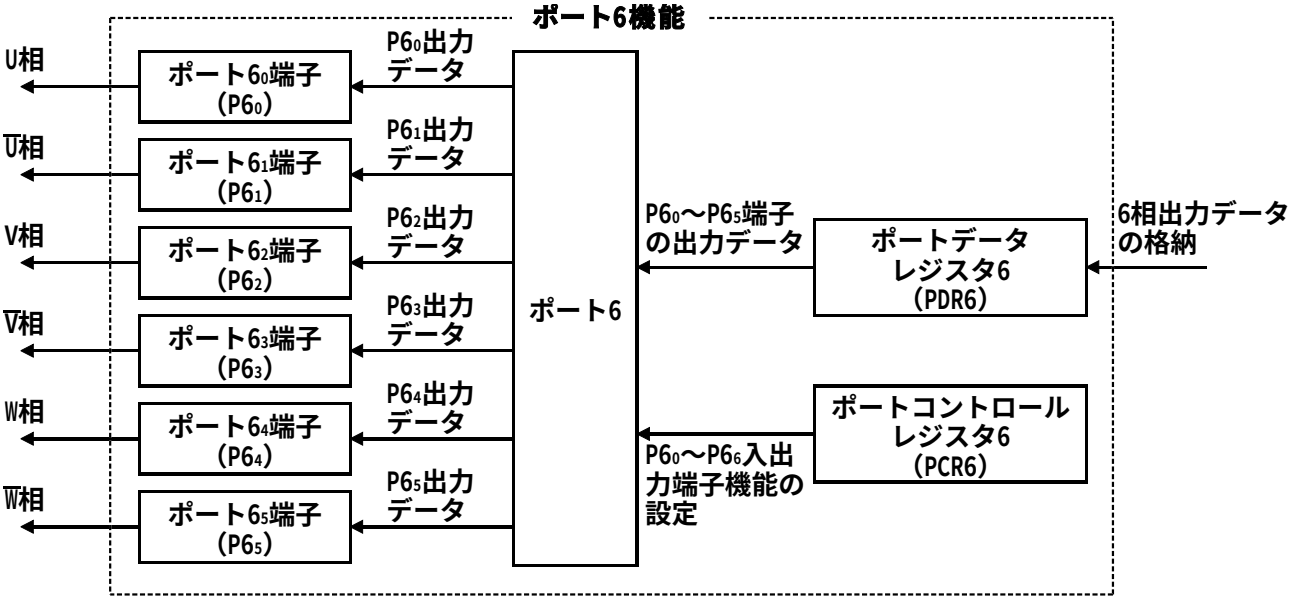


図6　ポート6を使用した6相出力ブロック図

(e) IRQ外部割込み機能により異常検出入力信号の検出を行ないます。図7にIRQ外部割込み機能を使用した異常検出入力信号の検出のブロック図を示します。以下にブロック図について説明します。

・IRQ0端子はベースドライバからの異常検出信号の入力端子として機能します。
・IRQ0端子の入力エッジの検出によりIRQ0割込みを要求します。IRQ0端子の入力エッジセンスは立ち上がり/立ち下がりより選択可能です。
・割込みエッジセレクトレジスタ1（IEGR1）は、8ビットのリード/ライト可能なレジスタで、IRQ0端子の立ち上がり/立ち下がりエッジを指定します。
・割込み許可レジスタ1（IENR1）は、8ビットのリード/ライト可能なレジスタで、割込み要求の許可/禁止を制御します。本タスク例では、IRQ0割込み要求を許可し、タイマB1、タイマA、IRQ1、IRQ2、IRQ3割込み要求は禁止します。
・割込み要求レジスタ1（IRR1）は、8ビットのリード/ライト可能なレジスタで、タイマB1、タイマA、IRQ0～IRQ3割込み要求が発生すると対応するフラグが"1"にセットされます。各割込み要求フラグは割込みが受け付けられてもオートクリアされません。各割込み要求フラグをクリアする場合は"0"をライトしてください。

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ_0

使用機能説明

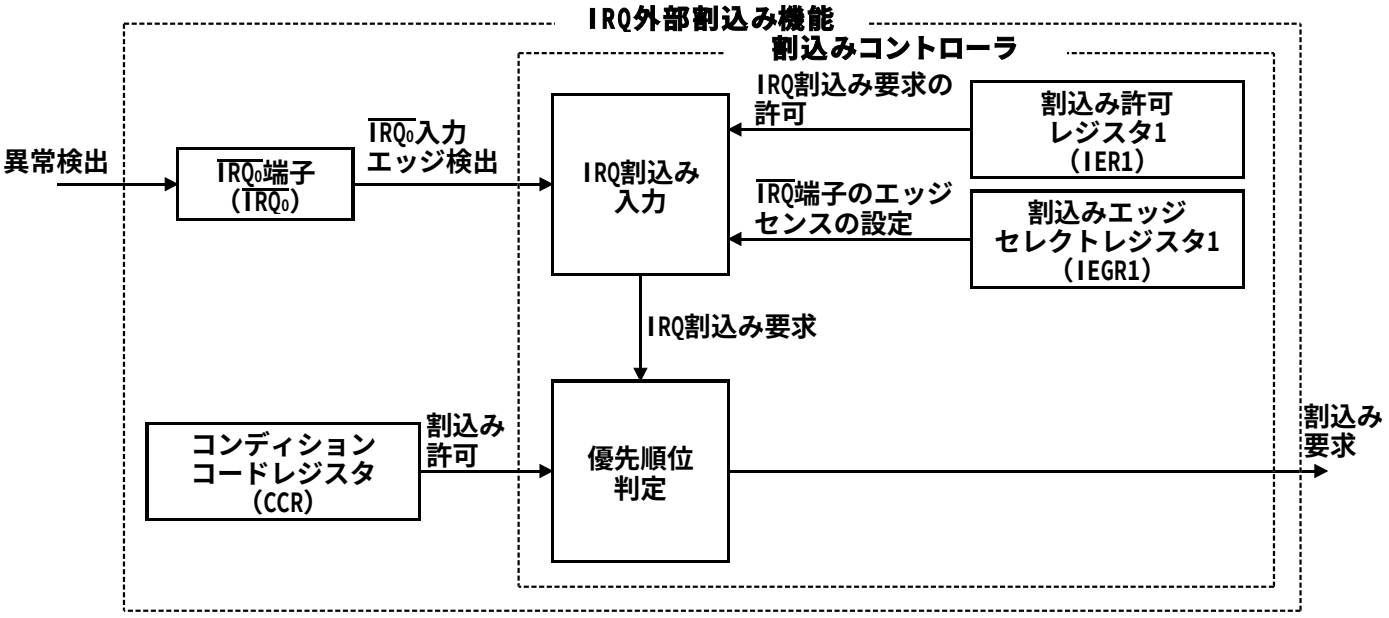


図7　IRQ外部割込み機能を使用した異常検出入力信号の検出ブロック図

(3) 表1に本タスク例の機能割付けを示します。表1に示すようにH8/3644の機能を割り付け、DCブラシレスモータ制御を行ないます。

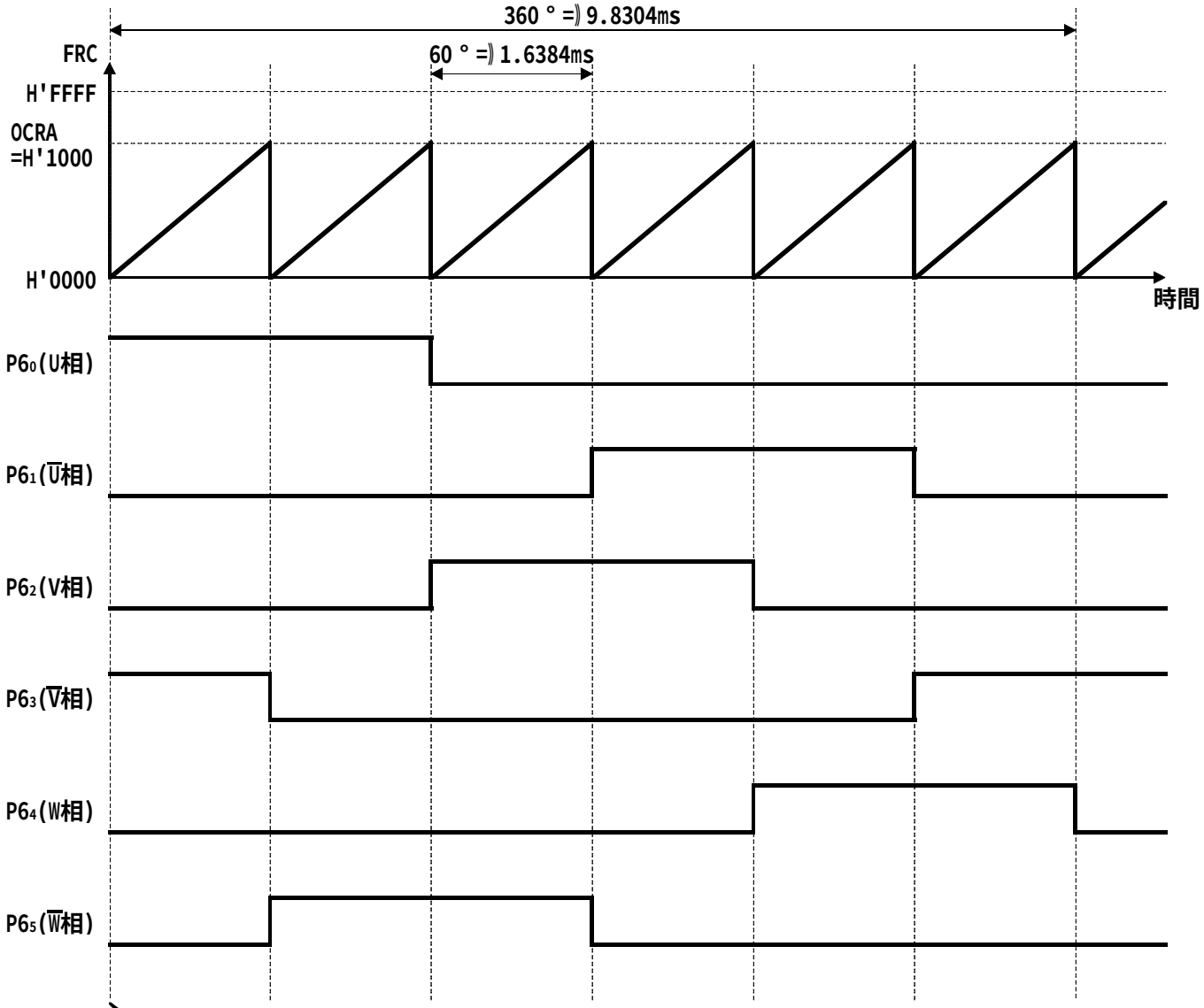
表1　H8/3644機能割付け

H8/3644機能	機能割付け
FTIA、B、C	ロータ位置検出信号入力
FRC	16ビットアップカウンタ、入力クロックはTCRXにより設定
TCRX	FRC入力クロックの設定、インプットキャプチャ入力エッジの設定
OCRA	初期モータ回転磁界切換え周期の設定
TCSRX	タイマXの割込み要求（ICIA、ICIB、ICIC、OCIA）の有無を反映
TIER	タイマXの割込み要求（ICIA、ICIB、ICIC、OCIA）許可または禁止を設定
TMOV	チョッピング波形出力
TCNTV	8ビットアップカウンタ
TCORA	チョッピング出力波形の周期を設定
TCORB	チョッピング出力波形のデューティを設定
TCRV0	TCNTV入力クロックの設定、TCNTVのクリア指定の設定、タイマVの割込み要求（CMIA、CMIB）の許可または禁止を設定
TCRV1	TCNTV入力クロックの設定
TCSRV	タイマVの割込み要求（CMIA、CMIB）の有無を反映、コンペアマッチ出力の設定
$P6_0$～$P6_5$	6相波形出力
PCR6	$P6_0$～$P6_5$出力端子機能の設定
PDR6	$P6_0$～$P6_5$出力端子のデータの格納、端子レベルの読み出し
$\overline{IRQ_0}$	異常検出信号入力
IEGR1	$\overline{IRQ_0}$端子の入力エッジの設定
IER1	IRQ_0割込み要求の有無を反映

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

動作説明

(1) 図8に初期モータ制御時（モータが10回転するまで、一定周期ごとに回転磁界を切換えモータを回転させる）の動作原理を示します。図8に示すようにH8/3644のハードウェア処理、およびソフトウェア処理により初期モータ制御時のDCブラスレスモータ制御を行ないます。



ハードウェア処理
(a)FRCカウントアップ開始
ソフトウェア処理
(a)PDR6に6相出力データを格納 (b)OCRAに初期モータ制御磁界切換え周期を設定 (c)OCFAによる割込み要求を許可

ハードウェア処理
(a)コンペアマッチA発生 (b)OCFAを"1"にセット (c)FRCをクリア
ソフトウェア処理
(a)OCFAを"0"にクリア (b)データテーブルより次の6相出力データをPDR6に格納

図8　初期モータ制御時のDCブラスレスモータ制御の動作原理

動作説明

(2) 図9にロータ位置信号の検出により回転磁界の切換えを行なうことによるDCブラスレスモータ制御の動作原理を示します。図9に示すようにH8/3644のハードウェア処理、およびソフトウェア処理によりロータ位置信号の検出によるDCブラスレスモータ制御を行ないます。

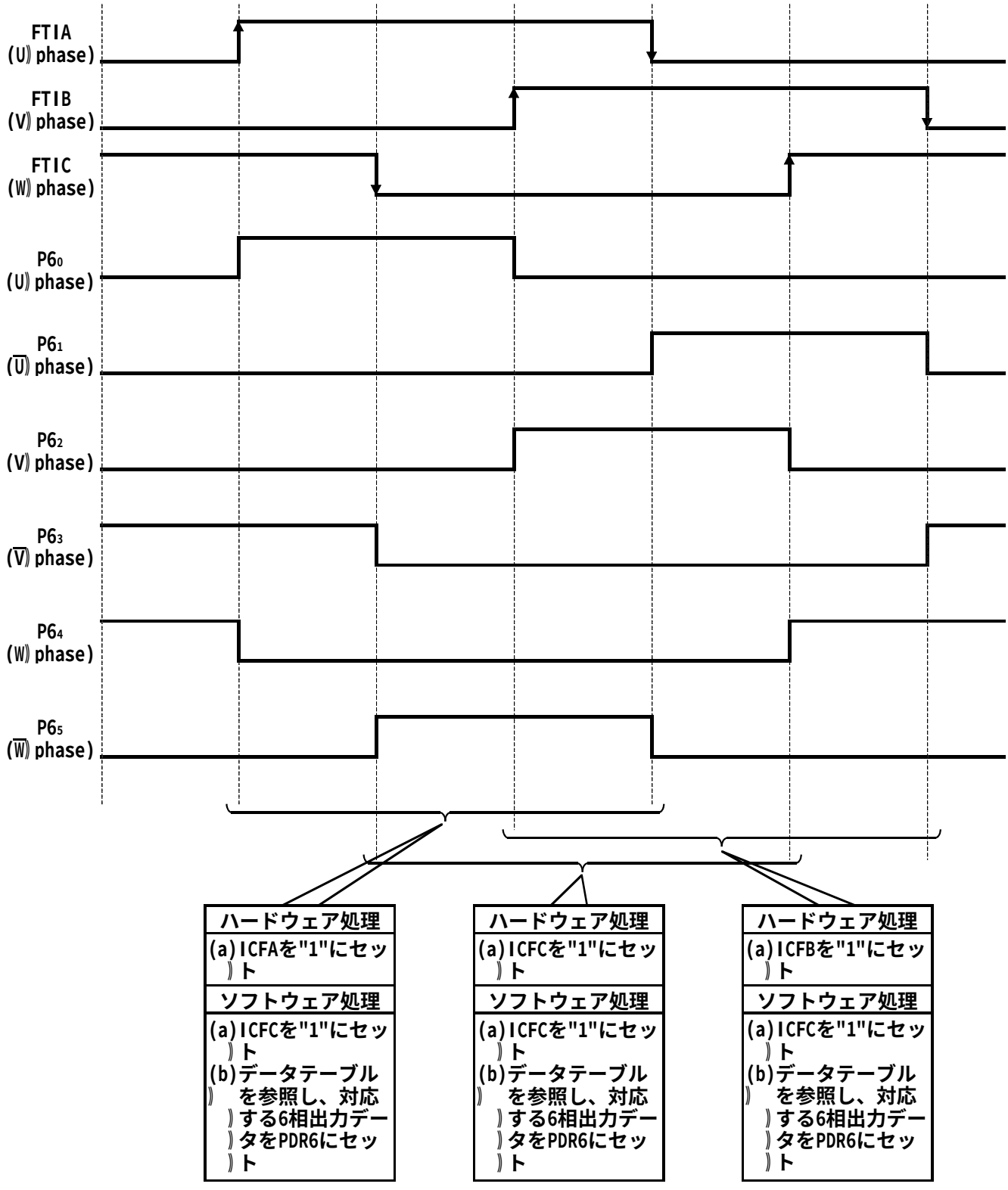


図9　ロータ位置信号検出によるDCブラシレスモータ制御の動作原理

(3) 本制御例では、ドライバがONする際にチョッピング制御を行ないます。チョッピング波形は、タイマVコンペアマッチ機能により生成し、TMOV端子から出力します。図10にチョッピング波形出力の動作原理を示します。

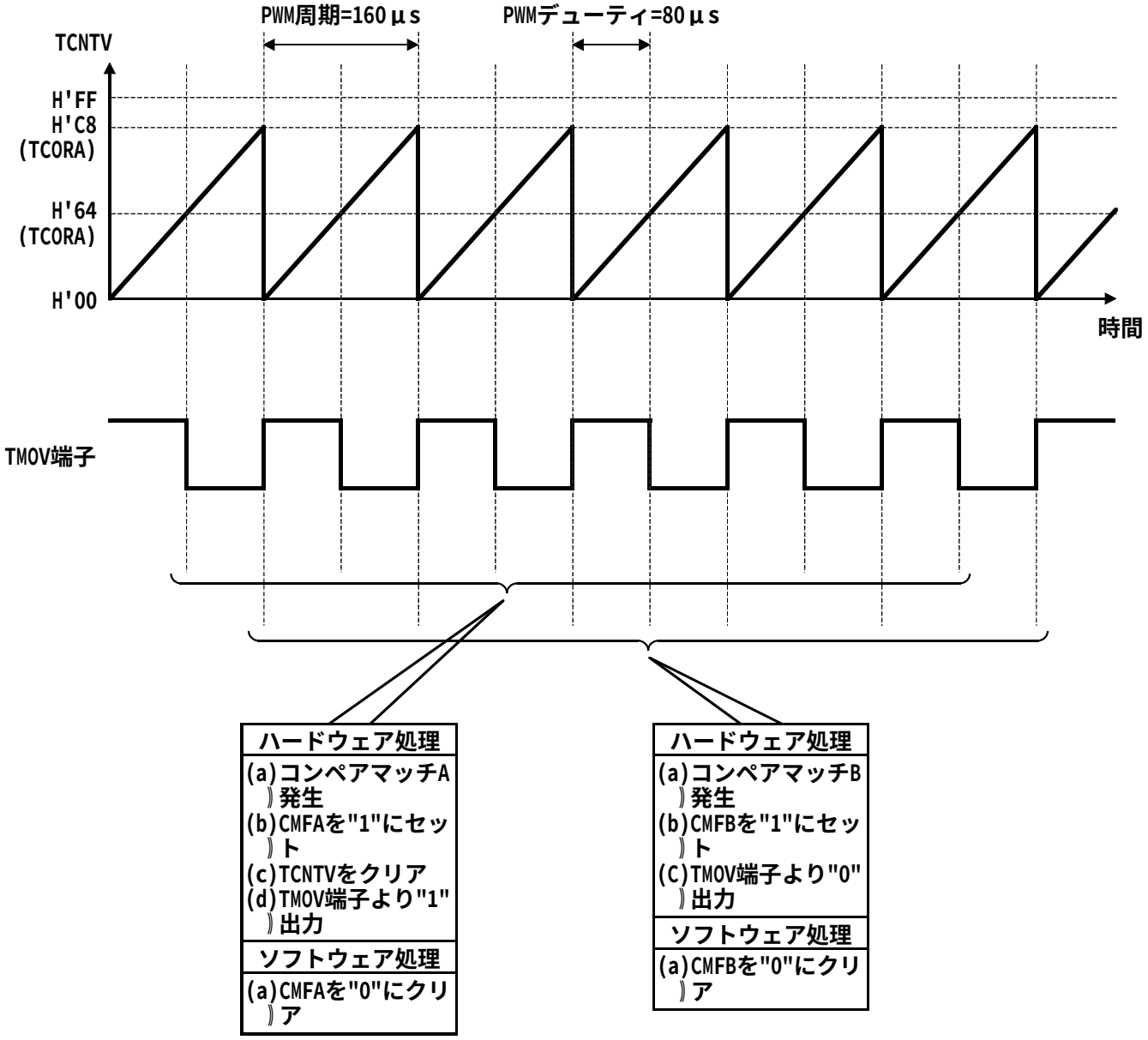


図10　チョッピング波形出力の動作原理

・位相の計算方法とPWMデューティ設定方法

コンペアマッチA割込みにより、次のデューティを計算し、TCORBに設定します。デューティの計算は、現在のモータ周波数とキャリア周期により、1キャリアでの位相の進みを求めます。

360°/(キャリア周期)/モータ周波数)=1キャリアでの位相の進み……(1)

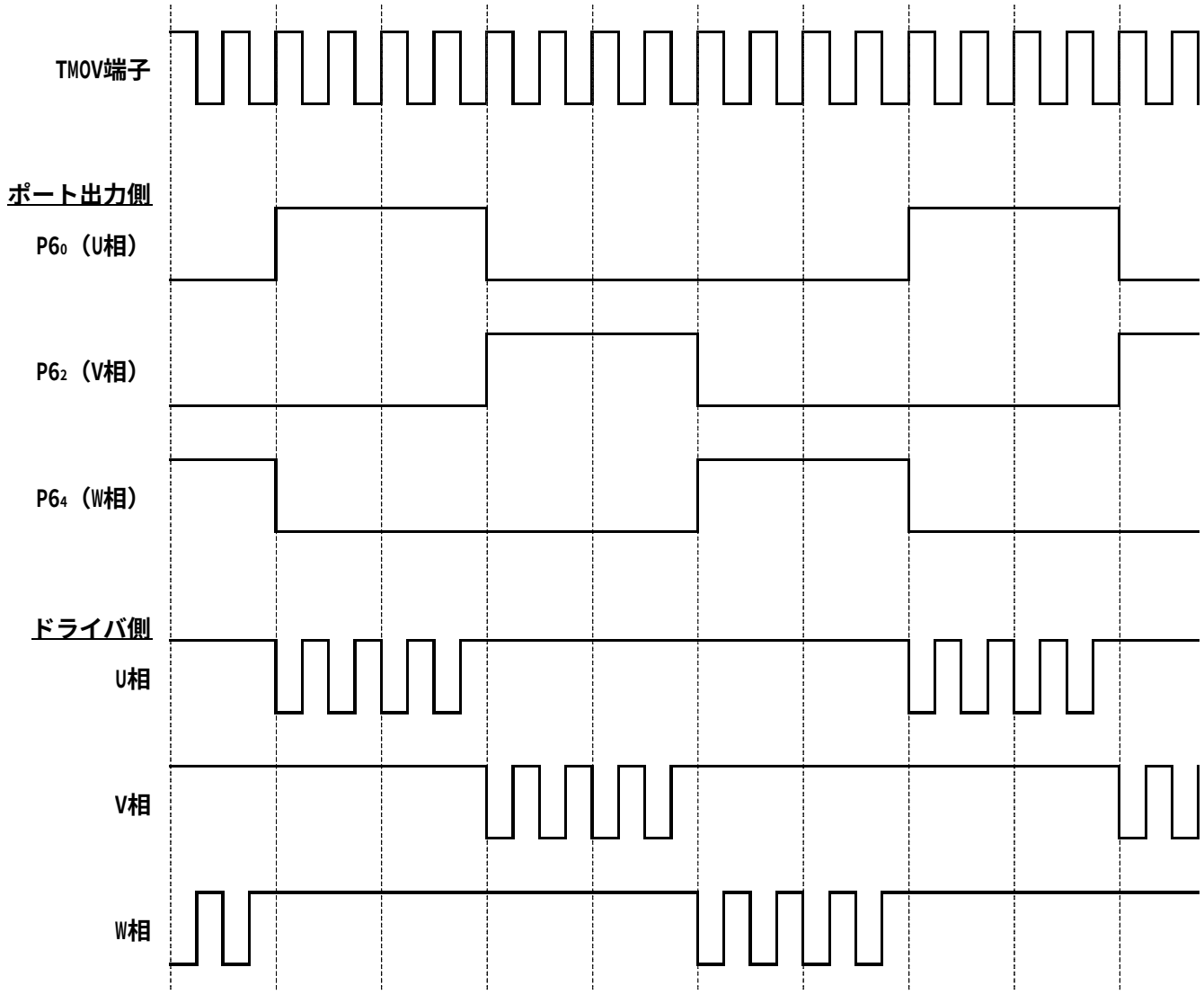
(1)式を変形すると、

(360°×モータ周波数)/キャリア周期　=1キャリアでの位相の進み

となり、計算された位相に対応するデューティをデータテーブルから参照して、TCORBに設定します。

【注意】本タスク例では、PWM周期（TCORA設定値）を160μs（H'C8）、およびPWMデューティ（TCORB設定値）を80μs（H'64）の値に固定して制御を行なっています。

(4) 6相出力の正相側に外付けのNAND回路を設け、チョッピング波形（TMOV端子）と回転磁界波形
》（ポート出力）のNANDをとり、ドライバへ出力します。図11にチョッピング波形と回転磁界波形の
》NANDで6相出力を行なう際の動作原理を示します。



【注意】ドライバは》Low》Active》信号とします。

図11　TMOV端子とポートのNANDによる6相出力の動作原理

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

ソフトウェア説明

(1) モジュール説明

表2に本タスク例におけるモジュール説明を示します。

表2　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	Reset	スタックポインタのイニシャライズ、IRQ0割込みの設定、ポートの設定、タイマXアウトプットコンペア機能の設定、使用RAMの設定、および各割込みの設定を行なう。
タイマX割込み処理ルーチン	TimerX	タイマX割込み処理ルーチンで、割込み要求フラグのクリアおよび6相出力の制御を行なう。
タイマV割込み処理ルーチン	TimerV	タイマV割込み処理ルーチンで、割込み要求フラグのクリアを行なう。
IRQ0割込み処理ルーチン	IRQ0	IRQ0割込み処理ルーチンで、割込み要求フラグのクリアおよびエラーフラグをたて、モータ停止処理を行なう。

(2) 引数の説明

表3に本タスク例における引数の説明を示します。

表3　引数の説明

引数名	機能	使用モジュール名	データ長	入出力
OCRA	初期モータ制御時、回転磁界切換え周期を設定	Reset	1ワード	入力
TCORA	チョッピング出力波形の周期を設定	TimerX	1バイト	入力
TCORB	チョッピング出力波形のデューティを設定	TimerX	1バイト	入力

(3) 使用内部レジスタ説明

表4に本タスク例における使用内部レジスタ説明を示します。

表4　使用内部レジスタ説明

レジスタ名	機能	アドレス	設定値
PDR6	ポートデータレジスタ6 ：P6nが"0"のとき、P6n端子レベルは"Low"（n=0～7） ：P6nが"1"のとき、P6n端子レベルは"High"（n=0～7）	H'FFD9	H'40
PCR6	ポートコントロールレジスタ6 ：PCR6nが"0"のとき、P6n端子は入力端子機能（n=0～7） ：PCR6nが"1"のとき、P6n端子は出力端子機能（n=0～7）	H'FFE9	H'3F
PDR7	ポートデータレジスタ6 ：P7nが"0"のとき、P7n端子レベルは"Low"（n=0～7） ：P7nが"1"のとき、P7n端子レベルは"High"（n=0～7）	H'FFDA	H'FF
PCR7	ポートコントロールレジスタ7 ：PCR7nが"0"のとき、P7n端子は入力端子機能（n=0～7） ：PCR7nが"1"のとき、P7n端子の出力端子機能（n=0～7）	H'FFEA	H'FF

表4　使用内部レジスタ説明（つづき）

レジスタ名		機能	アドレス	設定値
TCSRV	CMFB	タイマコントロール/ステータスレジスタ （コンペアマッチフラグB） ：CMFB="0"のとき、コンペアマッチBが発生していないことを示す ：CMFA="1"のとき、コンペアマッチBが発生したことを示す	H'FFB9 ビット7	0
	CMFA	タイマコントロール/ステータスレジスタ （コンペアマッチフラグA） ：CMFA="0"のとき、コンペアマッチAが発生していないことを示す ：CMFA="1"のとき、コンペアマッチAが発生したことを示す	H'FFB9 ビット6	0
	OS3 OS2 OS1 OS0	タイマコントロール/ステータスレジスタ （アウトプットセレクト3～0） ：OS3=0、OS2=1、OS1=1、OS0=0のとき、TMOV端子の出力レベルをコンペアマッチBで"0"出力、コンペアマッチAで"1"出力に設定	H'FFB9 ビット3 ビット2 ビット1 ビット0	OS3=0 OS2=1 OS1=1 OS0=0
TCORA		タイムコンスタントレジスタA ：TCORA=H'C8のとき、TCNTVがH'C8までカウントアップするとコンペアマッチAが発生します	H'FFBA	H'C8
TCORB		タイムコンスタントレジスタB ：TCORB=H'64のとき、TCNTVがH'64までカウントアップするとコンペアマッチAが発生します	H'FFBB	H'64
TIER	ICIAE	タイマインタラプトイネーブルレジスタ （インプットキャプチャインタラプトAイネーブル） ：ICIAE=0のとき、ICFAによる割込み要求を禁止 ：ICIAE=1のとき、ICFAによる割込み要求を許可	H'F770 ビット7	0
	ICIBE	タイマインタラプトイネーブルレジスタ （インプットキャプチャインタラプトBイネーブル） ：ICIBE=0のとき、ICFBによる割込み要求を禁止 ：ICIBE=1のとき、ICFBによる割込み要求を許可	H'F770 ビット6	0
	ICICE	タイマインタラプトイネーブルレジスタ （インプットキャプチャインタラプトCイネーブル） ：ICICE=0のとき、ICFCによる割込み要求を禁止 ：ICICE=1のとき、ICFCによる割込み要求を許可	H'F770 ビット5	0
	OCIAE	タイマインタラプトイネーブルレジスタ （アウトプットコンペアインタラプトイネーブルA） ：OCIAE=0のとき、OCFAによる割込み要求を禁止 ：OCIAE=1のとき、OCFAによる割込み要求を許可	H'F770 ビット3	1
TCSRX	ICFA	タイマコントロール/ステータスレジスタX （インプットキャプチャフラグA） ：ICFA=0のとき、インプットキャプチャ信号が入力されていないことを示す ：ICFA=1のとき、インプットキャプチャ信号によりFRCのカウンタ値がICRAに転送されたことを示す	H'F771 ビット7	0
	ICFB	タイマコントロール/ステータスレジスタX （インプットキャプチャフラグB） ：ICFB=0のとき、インプットキャプチャ信号が入力されていないことを示す ：ICFB=1のとき、インプットキャプチャ信号によりFRCのカウンタ値がICRBに転送されたことを示す	H'F771 ビット7	0

表4　使用内部レジスタ説明（つづき）

レジスタ名		機能	アドレス	設定値
TCSRX	ICFC	タイマコントロール/ステータスレジスタX （インプットキャプチャフラグC） ：ICFC=0のとき、インプットキャプチャ信号が入力されていないことを示す ：ICFC=1のとき、インプットキャプチャ信号によりFRCのカウンタ値がICRCに転送されたことを示す	H'F771 ビット5	0
	OCFA	タイマコントロール/ステータスレジスタX （アウトプットコンペアフラグA） ：OCFA=0のとき、FRCとOCRAがコンペアマッチしていないことを示す ：OCFA=1のとき、FRCとOCRAがコンペアマッチしたことを示す	H'F771 ビット3	0
OCRAH		アウトプットコンペアレジスタAH ：OCRAの上位8ビットで、OCRAの設定値とFRCのカウンタ値が一致すると、コンペアマッチAが発生	H'F774	H'10
OCRBH		アウトプットコンペアレジスタAL ：OCRAの下位8ビットで、OCRAの設定値とFRCのカウンタ値が一致すると、コンペアマッチAが発生	H'F775	H'00
TOCR	OCRS	タイマアウトプットコンペアコントロールレジスタ （アウトプットコンペアレジスタセレクト） ：OCRS=0のとき、OCRA、OCRBのアクセスの切換えをOCRAに設定 ：OCRS=1のとき、OCRA、OCRBのアクセスの切換えをOCRBに設定	H'F777 ビット4	0
	OEA	タイマアウトプットコンペアコントロールレジスタ （アウトプットイネーブルA） ：OEA=0のとき、アウトプットコンペアA出力を禁止 ：OEA=1のとき、アウトプットコンペアA出力を許可	H'F777 ビット3	0
TCRX	IEDGA	タイマコントロールレジスタX （インプットエッジセレクトA） ：IEDGA=0のとき、インプットキャプチャ入力Aの立ち下がりエッジでキャプチャ ：IEDGA=1のとき、インプットキャプチャ入力Aの立ち上がりエッジでキャプチャ	H'F776 ビット7	0
	IEDGB	タイマコントロールレジスタX （インプットエッジセレクトB） ：IEDGB=0のとき、インプットキャプチャ入力Bの立ち下がりエッジでキャプチャ ：IEDGB=1のとき、インプットキャプチャ入力Bの立ち上がりエッジでキャプチャ	H'F776 ビット6	0
	IEDGC	タイマコントロールレジスタX （インプットエッジセレクトC） ：IEDGC=0のとき、インプットキャプチャ入力Cの立ち下がりエッジでキャプチャ ：IEDGC=1のとき、インプットキャプチャ入力Cの立ち上がりエッジでキャプチャ	H'F776 ビット5	0
	CKS1 CKS0	タイマコントロールレジスタX （クロックセレクト1、0） ：CKS1=0、CKS0=0のとき、FRCの入力クロックはシステムクロックの2分周のクロックに設定	H'F776 ビット1 ビット0	CKS1=0 CKS0=0

表4　使用内部レジスタ説明（つづき）

レジスタ名		機能	アドレス	設定値
TCRV0	CMIEB	タイマコントロールレジスタV0 （コンペアマッチインタラプトイネーブルB） ：CMIEB=0のとき、CMFBによる割込み要求を禁止 ：CMIEB=1のとき、CMFBによる割込み要求を許可	H'FFB8 ビット7	1
	CMIEA	タイマコントロールレジスタV0 （コンペアマッチインタラプトイネーブルA） ：CMIEA=0のとき、CMFAによる割込み要求を禁止 ：CMIEA=1のとき、CMFAによる割込み要求を許可	H'FFB8 ビット6	1
	CCLR1 CCLR0	タイマコントロールレジスタV0 （カウンタクリア1、0） ：CCLR1=0、CCLR0=1のとき、コンペアマッチAによりTCNTVをクリア	H'FFB8 ビット4 ビット3	CCLR1=0 CCLR0=1
	CKS2 CKS1 CKS0	タイマコントロールレジスタV0 （クロックセレクト2～0） ：CKS2=0、CKS1=0、CKS0=1およびTCRV1のICKS0=0のとき、TCNTVはシステムクロックの4分周のクロックの立ち下がりエッジでカウント	H'FFB8 ビット2 ビット1 ビット0	CKS2=0 CKS1=0 CKS0=1
IEGR1	IEG0	インタラプトエッジセレクトレジスタ1 （IRQ0エッジセレクト） ：IEG0=0のとき、IRQ0端子入力の立ち下がりエッジを検出 ：IEG0=1のとき、IRQ0端子入力の立ち上がりエッジを検出	H'FFF2 ビット0	0
IENR1	IEN0	インタラプトイネーブルレジスタ1 （IRQ0インタラプトイネーブル） ：IEN0=0のとき、IRQ0割込み要求を禁止 ：IEN0=1のとき、IRQ0割込み要求を許可	H'FFF4 ビット0	0
IRR1	IRRI0	インタラプトリクエストレジスタ1 （IRQ0インタラプトリクエストフラグ） ：IRRI0=0のとき、IRQ0割込みが要求されていないことを示す ：IRRI0=1のとき、IRQ0割込みが要求されていることを示す	H'FFF7 ビット0	0

(4) 使用RAM説明

表5に本タスク例における使用RAM説明を示します。

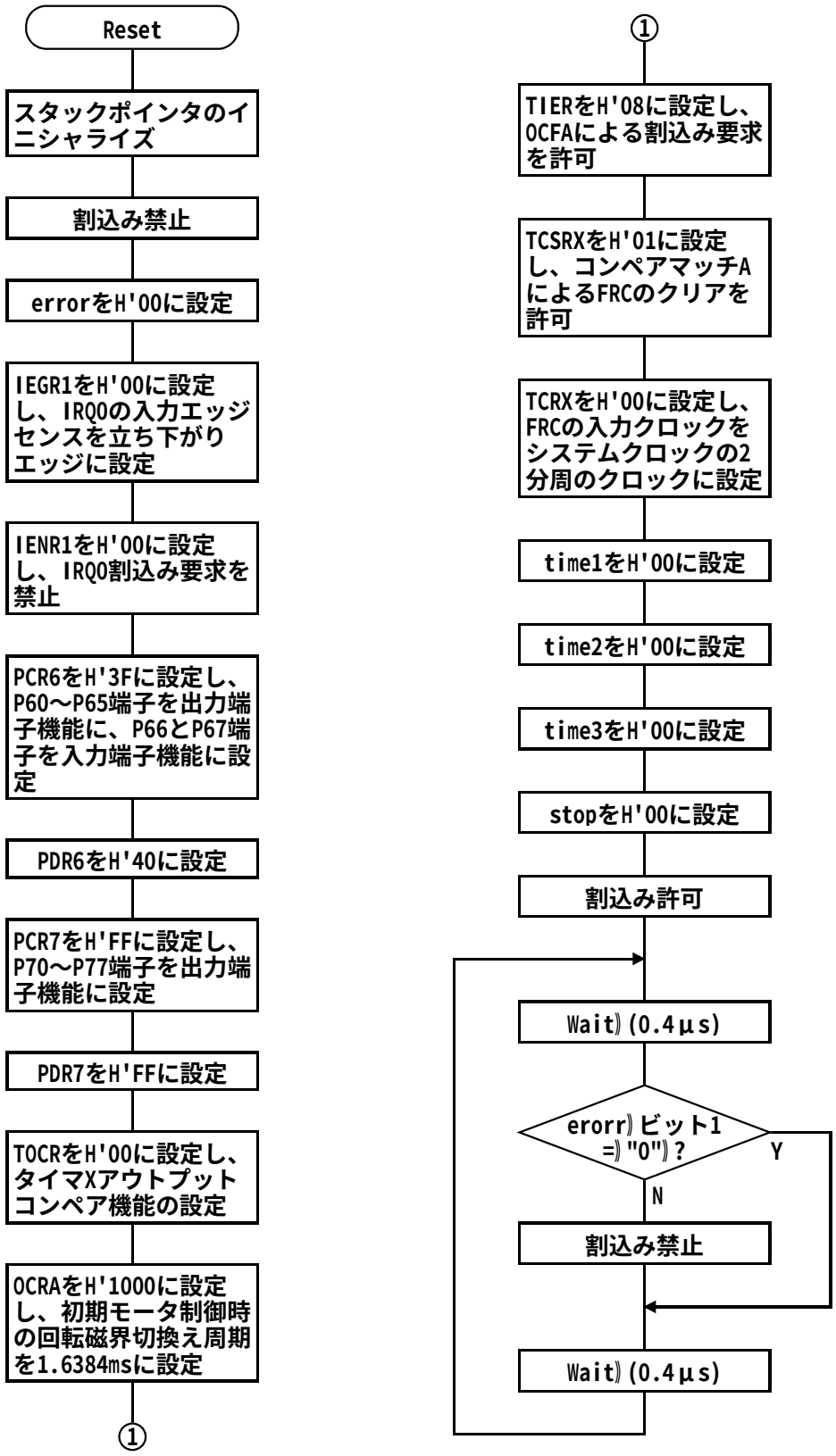
表5　使用RAM説明

ラベル名	機能	使用モジュール
time1	初期モータ制御時のデータを格納	Reset、TimerX
time2	ロータ位置検出によるモータ制御時のデータを格納	Reset、TimerX
time3	初期モータ制御終了を判定するフラグ	Reset、TimerX
erorr	エラー検出を判定するフラグ	Reset、TimerX、IRQ0
stop	ポート6のデータを格納	Reset、IRQ0

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

フローチャート

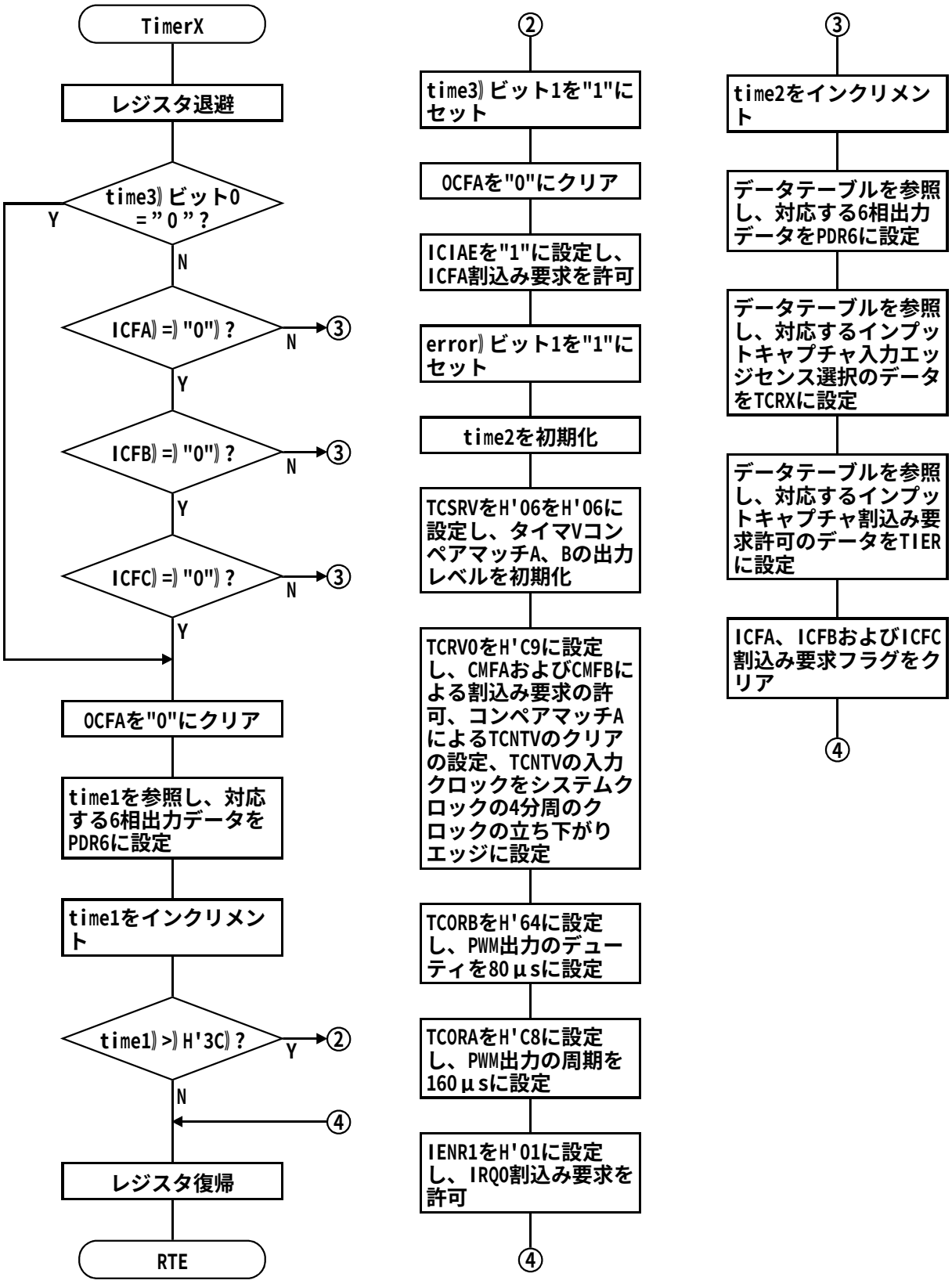
a.メインルーチン



DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

フローチャート

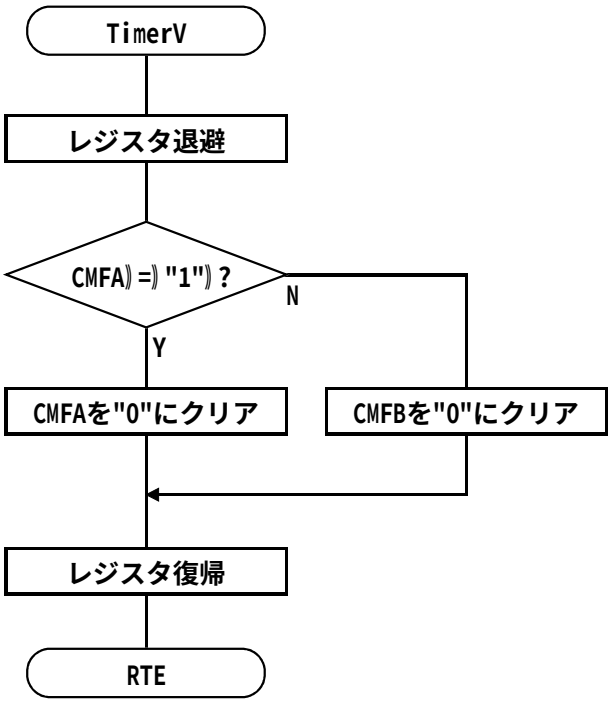
b.タイマX割込み処理ルーチン



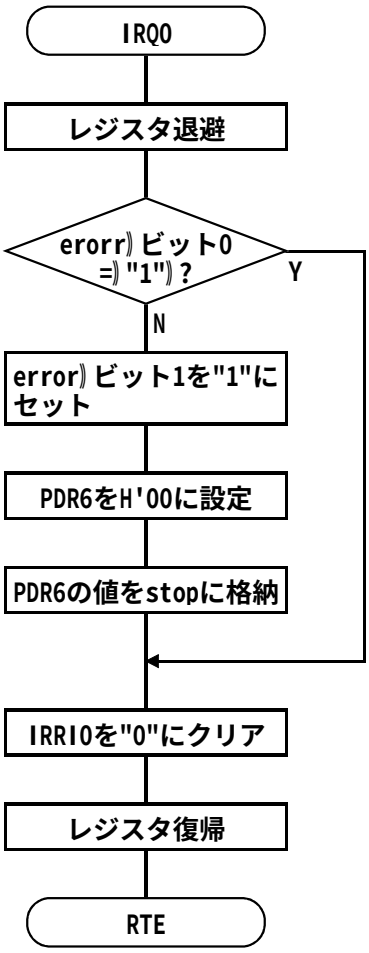
DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

フローチャート

c.タイマV割込み処理ルーチン



d.IRQ0割込み処理ルーチン



DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

プログラムリスト

```
;************************************************************
;
;               H8/300L Series -H8/3644, H8/3657-
;               Application Note
;
;               'Control the DC Brushless Motor'
;
;               Function : I/O Port
;                       Timer V Compare Match
;                       Timer X Input Capture Timer
;                       IRQ0,IRQ1 External Interrupt
;
;               External Clock : 10MHz
;               Internal Clock : 5MHz
;               Sub Clock      : 32.768kHz
;
;************************************************************
;
;************************************************************
;
                .cpu            300L
;************************************************************
;
;************************************************************
;               Symbol Defnition
;************************************************************
;
PDR6            .equ            H'FFD9          ; Port Data Register 6
PCR6            .equ            H'FFE9          ; Port Control Register 6
PCR7            .equ            H'FFEA          ; Port Control Register 7
PDR7            .equ            H'FFDA          ; Port Data Register 7
TCSRV           .equ            H'FFB9          ; Timer Control/Status Register V
TCORA           .equ            H'FFBA          ; Time Constant Register A
TCORB           .equ            H'FFBB          ; Time Constant Register B
TIER            .equ            H'F770          ; Timer Interrupt Enable Register
TCSRX           .equ            H'F771          ; Timer Control/Status Register X
OCRAH           .equ            H'F774          ; Output Compare Register AH
OCRAL           .equ            H'F775          ; Output Conpare Register AL
TOCR            .equ            H'F777          ; Timer Output Compare Control rehister
TCRX            .equ            H'F776          ; Timer Control Register X
TCRV0           .equ            H'FFB8          ; Timer Control Register V0
IEGR1           .equ            H'FFF2          ; Interrupt Edge Select Register 1
IENR1           .equ            H'FFF4          ; Interrupt Enable Register 1
IRR1            .equ            H'FFF7          ; Interrupt Request Register 1
;
;************************************************************
;               Vector Address
;************************************************************
;
;
                .section        VECTOR,COMMON,ALIGN=2
;
                .org            H'0000
                .data.w         Reset           ; Reset Interrupt
;
                .org            H'0008
                .data.w         IRQ0            ; IRQ0 Interrupt
                .data.w         Reset           ; IRQ1 Interrupt
                .data.w         Reset           ; IRQ2 Interrupt
                .data.w         Reset           ; IRQ3 Interrupt
                .data.w         Reset           ; INT0-INT7 Interrupt
;
```

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

プログラムリスト

```
                .org            H'0014
                .data.w         Reset           ;Timer A Interrupt
                .data.w         Reset           ;Timer B1 Interrupt
;
                .org            H'0020
                .data.w         TimerX          ;Timer X Interrupt
                .data.w         TimerV          ;Timer V Interrupt
;
                .org            H'0026
                .data.w         Reset           ;SCI1 Interrupt
;
                .org            H'002A
                .data.w         Reset           ;SCI3 Interrupt
                .data.w         Reset           ;A/D Converter Interrupt
                .data.w         Reset           ;Direct Transfer Interrupt
;
;*********************************************************
;               Main Routine
;*********************************************************
;
                .section        CODE,code,align=2
Reset:
                mov.w           #H'ff70,r7      ;SP configuration
                mov.b           #H'80,r0l
                orc             #H'80,ccr       ;mask all the interrupt
;
                mov.b           #0,r0l          ;IRQ0 status RAM initiation
                mov.b           r0l,@error
                mov.b           #0,r0l
                mov.b           r0l,@IEGR1      ;falling edge of IRQ0 detect
                mov.b           #0,r0l
                mov.b           r0l,@IENR1      ;mask IRQ0
                mov.b           #B'00111111,r0l
                mov.b           r0l,@PCR6       ;Port6 set P60-P65 are output P66 is input
                mov.b           #H'40,r0l
                mov.b           r0l,@PDR6       ;stop the motor at the begining of the programe
                mov.b           #H'ff,r0l
                mov.b           r0l,@PCR7       ;Port seven is output port
                mov.b           #H'ff,r0l
                mov.b           r0l,@PDR7       ;P76 output high level at the begining of the programe
                mov.b           #H'0,r0l
                mov.b           r0l,@TOCR       ;disable the compare output A and B)))))))
                mov.w           #H'1000,r1
                mov.b           r1h,@OCRAH      ;set the initial motor control timer
                mov.b           r1l,@OCRAL
                mov.b           #H'8,r0l
                mov.b           r0l,@TIER       ;enable the compare match interrupt A
                mov.b           #H'1,r0l
                mov.b           r0l,@TCSRX      ;FRC is cleared by compare match A))
                mov.b           #H'0,r0l
                mov.b           r0l,@TCRX       ;select timer for FRC
                mov.b           #H'0,r0l
                mov.b           r0l,@time1      ;RAM initialization
                mov.b           r0l,@time2
                mov.b           r0l,@time3
                mov.b           r0l,@stop
                andc            #H'7f,ccr       ;enable interrupt
wait:
                nop
                nop                             ;wait
```

プログラムリスト

```
                mov.b           @error,r0l
                btst            #1,r0l
                beq             con             ;) when) the) error) of) IRQ0) happened) mask) all) the) interrupt
                orc             #H'80,ccr
con:
                nop
                nop
                jmp             @wait
;
;*************************************************************
;
;               Timer) X) Interrupt) Routine
;*************************************************************
;
;
TimerX:
                push            r0
                push            r1              ;) keep) the) register
                push            r2
                mov.b           @time3,r0l      ;) is) the) initial) ten) times) rotating) over
                btst            #0,r0l
                beq             tim
                mov.b           @TCSRX,r0l      ;) is) the) interrupt) a) capture) interrupt
                btst            #7,r0l
                bne             Capa
                btst            #6,r0l
                bne             Capa
                btst            #5,r0l
                bne             Capa
;
;*************************************************************
;
;)              Compare) match) A) interrupt) service
;               programe) used) for) the) initial) ten
;               times) rotating) control
;*************************************************************
;
;
tim:
                mov.b           @TCSRX,r0l
                and.b           #B'11110111,r0l
                mov.b           r0l,@TCSRX      ;) clear) the) interrupt) flag
                sub.w           r2,r2
                mov.b           @time1,r2l
                mov.b           #6,r1l
                divxu           r1l,r2
                sub.w           r0,r0
                                mov.b   r2h,r0l
                mov.b           @(Control,r0),r1l
                mov.b           r1l,@PDR6       ;) search) the) control) table) for) Port) six) output
                mov.b           @time1,r0l
                add.b           #1,r0l          ;) figure) initial) rotating) time
;
                cmp.b           #60,r0l
                bhi             over            ;) is) initial) rotating) control) over) ) ) ) ) ) ) ) )
                mov.b           r0l,@time1
                jmp             @back
;
over:
                mov.b           #1,r0l          ;) set) the) initial) control) over) flag
                mov.b           r0l,@time3
                mov.b           @TCSRX,r0l
                and.b           #B'11110111,r0l
                mov.b           r0l,@TCSRX      ;) clear) the) interrupt) flag
```

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

プログラムリスト

```
;
                mov.b           #B'10000000,r0l
                mov.b           r0l,@TIER       ;) enable) the) input) capture) interrupt) A
;
                mov.b           #1,r0l          ;) set) the) IRQ0) detect) enable) flag
                mov.b           r0l,@error
                mov.b           #0,r0l
                mov.b           r0l,@time2      ;) initialize) time2
                mov.b           #B'00000110,r0l
                mov.b           r0l,@TCSRV      ;) configurate) timer) V
                mov.b           #B'11001001,r0l
                mov.b           r0l,@TCRV0
                mov.b           #100,r0l
                mov.b           r0l,@TCORB
;
                mov.b           #200,r0l
                mov.b           r0l,@TCORA      ;) set) the) compare) match) value) for) A) and) B
                mov.b           #1,r0l
                mov.b           r0l,@IENR1
;
back:
                pop             r2
                pop             r1
                pop             r0              ;) interrupt) service) programe) back
                rte
;
;*************************************************************
;               Input) capture) interrupt) service) programe
;*************************************************************
;
Capa:
                mov.b           @time2,r0l
                add.b           #1,r0l          ;) add) time2
;
                sub.b           r0h,r0h
                mov.b           #6,r1l)
                divxu           r1l,r0
                mov.b           r0h,r1l
;
                mov.b           r0h,@time2
                mov.b           #0,r0h
                mov.b           r1l,r0l
                mov.b           @(Control,r0),r1l
                mov.b           r1l,@PDR6       ;search) Control) table) for) output) data) of) Port) six
                mov.b           @(Control1,r0),r1l
                mov.b           r1l,@TCRX       ;search) Control1) table) for) setting) data) of) TIER
                mov.b           @(Control2,r0),r1l
                mov.b           r1l,@TIER       ;search) Control2) table) for) setting) data) of) TCSRX
                mov.b           @TCSRX,r0l
                and.b           #B'00011111,r0l
                mov.b           r0l,@TCSRX      ;) clear) the) interrupt) flag
                jmp             @back
;
;*************************************************************
;               Timer) V) Interrupt) Routine
;*************************************************************
;
TimerV:
                push            r0
                push            r1
```

DCブラシレスモータ制御	使用機能	I/Oポート、タイマV、タイマX、IRQ0

プログラムリスト

```
                push            r2
                mov.b           @TCSRV,r0l
                btst            #6,r0l          ;) clear) interrupt) flag
                bne             timera
                mov.b           #B'01111111,r1l
                and.b           r1l,r0l
                mov.b           r0l,@TCSRV
                jmp             @timervend
;
timera:
                mov.b           #B'10111111,r1l
                mov.b           @TCSRV,r0l      ;) clear) interrupt) flag
                and.b           r1l,r0l
                mov.b           r0l,@TCSRV
;
timervend:
))))))))       pop     r2
))))))))       pop     r1
))))))))       pop     r0
))))))))       rte
;
;*********************************************************
;               IRQ0) Interrupt) Routine
;*********************************************************
;
IRQ0:
                push            r0
                mov.b           @error,r0l      ;) is) the) initial) motor) control) over
                btst            #0,r0l
                beq             Irq0end
                mov.b           #2,r0l
                mov.b           r0l,@error      ;) set) the) error) flag
                mov.b           #B'00000000,r0l
                mov.b           r0l,@PDR6       ;) stop) the) motor
                mov.b           @PDR6,r0l       ;) keep) the) P66) status
                mov.b           r0l,@stop
Irq0end:
                mov.b           @IRR1,r0l
                mov.b           #B'11111110,r0h
                and.b           r0h,r0l
                mov.b           r0l,@IRR1       ;) clear) the) interrupt) flag
                pop             r0
                rte
;
;*********************************************************
;               DATA) table
;*********************************************************
;
                .section        DATA,data,align=2
Control:
                .DATA.B)        B'00001001      ;) data) for) the) output) of) Port) six
                .DATA.B)        B'00100001
                .DATA.B)        B'00100100
                .DATA.B)        B'00000110
                .DATA.B)        B'00010010
                .DATA.B)        B'00011000
;
Control1:
                .DATA.B)        B'10000001      ;) data) for) the) timerX) control
                .DATA.B)        B'01000001
```

プログラムリスト

```
                .DATA.B         B'00100001
                .DATA.B         B'00000001
                .DATA.B         B'00000001
                .DATA.B         B'00000001
Control2:
                .DATA.B         B'10000000
                .DATA.B         B'01000000
                .DATA.B         B'00100000
                .DATA.B         B'10000000
                .DATA.B         B'01000000
                .DATA.B         B'00100000
;
;*********************************************************
;
;               RAM configuration
;*********************************************************
;
;
                .section        Stack,stack,align=2
;
time1           .res.b          1               ;initial motor control ram
time2           .res.b          1               ;main control ram
time3           .res.b          1               ;initial motor control end flag ram
error           .res.b          1               ;error detect flag ram
stop            .res.b          1               ;P66 data
;
                .end
```

H8/300L シリーズ　アプリケーションノート（応用編）

発行年月　平成 11 年 2 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

H8/300L シリーズ
アプリケーションノート（応用編）

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-502-073