お客様各位

カタログ等資料中の旧社名の扱いについて

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

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

2010年4月1日
ルネサスエレクトロニクス株式会社

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

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

ご注意書き

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

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

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

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

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

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

RENESAS

アプリケーション・ノート

V850ES/JG2, V850ES/JJ2

32ビット・シングルチップ・マイクロコントローラ

V850ES/JG2：
μPD70F3715
μPD70F3716
μPD70F3717
μPD70F3718
μPD70F3719

V850ES/JJ2：
μPD70F3720
μPD70F3721
μPD70F3722
μPD70F3723
μPD70F3724

資料番号 U18676JJ1V0AN00（第1版）
発行年月 April 2007 NS CP(K)

〔メ　モ〕

目次要約

CMOSデバイスの一般的注意事項

① 入力端子の印加波形

入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。

CMOSデバイスの入力がノイズなどに起因して，V_{IL}（MAX.）からV_{IH}（MIN.）までの領域にとどまるような場合は，誤動作を引き起こす恐れがあります。入力レベルが固定な場合はもちろん，V_{IL}（MAX.）からV_{IH}（MIN.）までの領域を通過する遷移期間中にチャタリングノイズ等が入らないようご使用ください。

② 未使用入力の処理

CMOSデバイスの未使用端子の入力レベルは固定してください。

未使用端子入力については，CMOSデバイスの入力に何も接続しない状態で動作させるのではなく，プルアップかプルダウンによって入力レベルを固定してください。また，未使用の入出力端子が出力となる可能性（タイミングは規定しません）を考慮すると，個別に抵抗を介してV_{DD}またはGNDに接続することが有効です。

資料中に「未使用端子の処理」について記載のある製品については，その内容を守ってください。

③ 静電気対策

MOSデバイス取り扱いの際は静電気防止を心がけてください。

MOSデバイスは強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には，当社が出荷梱包に使用している導電性のトレーやマガジン・ケース，または導電性の緩衝材，金属ケースなどを利用し，組み立て工程にはアースを施してください。プラスチック板上に放置したり，端子を触ったりしないでください。

また，MOSデバイスを実装したボードについても同様の扱いをしてください。

④ 初期化以前の状態

電源投入時，MOSデバイスの初期状態は不定です。

電源投入時の端子の出力状態や入出力設定，レジスタ内容などは保証しておりません。ただし，リセット動作やモード設定で定義している項目については，これらの動作ののちに保証の対象となります。

リセット機能を持つデバイスの電源投入後は，まずリセット動作を実行してください。

⑤ 電源投入切断順序

内部動作および外部インタフェースで異なる電源を使用するデバイスの場合，原則として内部電源を投入した後に外部電源を投入してください。切断の際には，原則として外部電源を切断した後に内部電源を切断してください。逆の電源投入切断順により，内部素子に過電圧が印加され，誤動作を引き起こしたり，異常電流が流れ内部素子を劣化させたりする場合があります。

資料中に「電源投入切断シーケンス」についての記載のある製品については，その内容を守ってください。

⑥ 電源OFF時における入力信号

当該デバイスの電源がOFF状態の時に，入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により，誤動作を引き起こしたり，異常電流が流れ内部素子を劣化させたりする場合があります。

資料中に「電源OFF時における入力信号」についての記載のある製品については，その内容を守ってください。

- 本資料に記載されている内容は2007年3月現在のもので，今後，予告なく変更することがあります。量産設計の際には最新の個別データ・シート等をご参照ください。
- 文書による当社の事前の承諾なしに本資料の転載複製を禁じます。当社は，本資料の誤りに関し，一切その責を負いません。
- 当社は，本資料に記載された当社製品の使用に関連し発生した第三者の特許権，著作権その他の知的財産権の侵害等に関し，一切その責を負いません。当社は，本資料に基づき当社または第三者の特許権，著作権その他の知的財産権を何ら許諾するものではありません。
- 本資料に記載された回路，ソフトウエアおよびこれらに関する情報は，半導体製品の動作例，応用例を説明するものです。お客様の機器の設計において，回路，ソフトウエアおよびこれらに関する情報を使用する場合には，お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し，当社は，一切その責を負いません。
- 当社は，当社製品の品質，信頼性の向上に努めておりますが，当社製品の不具合が完全に発生しないことを保証するものではありません。当社製品の不具合により生じた生命，身体および財産に対する損害の危険を最小限度にするために，冗長設計，延焼対策設計，誤動作防止設計等安全設計を行ってください。
- 当社は，当社製品の品質水準を「標準水準」，「特別水準」およびお客様に品質保証プログラムを指定していただく「特定水準」に分類しております。また，各品質水準は，以下に示す用途に製品が使われることを意図しておりますので，当社製品の品質水準をご確認ください。

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

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

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

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

(注)

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

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

M8E　02.11

はじめに

注　意　1. **この説明書で使用するプログラムは，**
NEC**エレクトロニクスのホーム・ページ（**http://www.necel.co.jp/**）の**
サンプル・プログラムのページ（http://www.necel.com/micro/ja/designsupports/sampleprogram/index.html**）**
よりダウンロードしてください。

2. **このサンプル・プログラムはあくまで参考用のものであり，当社がこの動作を保証するものではありません。このサンプル・プログラムを使用する場合，お客様のセット上で十分な評価をしたうえでご使用いただきますようお願いいたします。**
3. **このサンプル・プログラムを使用する場合は，次のスタートアップ・ファイル，リンク・ディレクティブ・ファイルを参照し，必要に応じて修正してください。**
   - **スタートアップ・ファイル　　：**JG2_start.s
   - **リンク・ディレクティブ・ファイル　：**JG2_link.dir

対 象 者　**このアプリケーション・ノートは，**V850ES/JG2（μPD70F3715, 70F3716, 70F3717, 70F3718, 70F3719），V850ES/JJ2（μPD70F3720, 70F3721, 70F3722, 70F3723, 70F3724）**の機能を理解し，それらを使用した応用システムを設計するユーザを対象とします。**

目　　的　**このアプリケーション・ノートは，**V850ES/JG2, V850ES/JJ2**製品の基礎的な機能について，応用プログラムを用いてユーザに理解していただくことを目的とします。**

読 み 方　**このマニュアルの読者には，電気，論理回路，およびマイクロコンピュータに関する一般知識を必要とします。**

ハードウエア機能の詳細を理解しようとするとき
→別冊の V850ES/JJ2　**ユーザーズ・マニュアル　ハードウエア編**
V850ES/JG2　**ユーザーズ・マニュアル　ハードウエア編を参照してください。**

命令機能の詳細を理解しようとするとき
→別冊のV850ES　**ユーザーズ・マニュアル　アーキテクチャ編を参照してください。**

凡　　例　**データ表記の重み：左が上位桁，右が下位桁**
アクティブ・ロウの表記：$\overline{\text{xxx}}$**（端子，信号名称に上線）**
メモリ・マップのアドレス：上部－上位，下部－下位
注　：本文中に付けた注の説明
注意：気を付けて読んでいただきたい内容
備考：本文の補足説明
数の表記：2**進数　…** xxxx**または**xxxxB
10**進数 …** xxxx
16**進数 …** xxxxH

2のべき数を示す接頭語（アドレス空間，メモリ容量）：

K（キロ）… $2^{10} = 1024$

M（メガ）… $2^{20} = 1024^2$

G（ギガ）… $2^{30} = 1024^3$

関数一覧表は次のように構成されています。

テーマ（ハードウエア略号）

【機　　能】	テーマの説明
【関　数　名】	サンプル関数の名前
【引　き　数】	引き数の型と概要
【処　理　内　容】	サンプル関数の処理内容
【起　動　方　法】	関数の呼び出し条件
【使　用 S　F　R】	レジスタ名と設定内容
【 call　　関数】	呼び出し関数の名前と機能
【変　　数】	サンプル関数での使用変数の型，名前，概要
【割　り　込　み】	関数名
【割り込み 要 因】	名称
【フ ァ イ ル 名】	対応するサンプル・プログラム・ファイル名
【注　意　事　項】	関数使用上の注意。使い方

割り込み関数

【関　数　名】	割り込み関数の名前
【概　　要】	処理の内容
【要　　因】	割り込み名と発生条件
【 call　　関数】	なし
【変　　数】	変数名，機能
【フ ァ イ ル 名】	対応するサンプル・プログラム・ファイル名
【注　意　事　項】	なし

関連資料　関連資料は暫定版の場合がありますが，この資料では「暫定」の表示をしておりません。あらかじめご了承ください。

V850ES/JG2, V850ESJJ2**に関する資料**

資　料　名	資料番号
V850ES　**ユーザーズ・マニュアル　アーキテクチャ編**	U15943J
V850ES/JG2　**ユーザーズ・マニュアル　ハードウエア編**	U17715J
V850ES/JJ2　**ユーザーズ・マニュアル　ハードウエア編**	U17714J
V850ES/JG2, V850ES/JJ2 **アプリケーション・ノート**	**このマニュアル**

開発ツールに関する資料（ユーザーズ・マニュアル）

資　料　名		資料番号
QB-V850ESSX2　**インサーキット・エミュレータ**		U17091J
QB-V850MINI　**オンチップ・デバック・エミュレータ**		U17638J
QB-MINI2　**プログラミング機能付きオンチップ・デバック・エミュレータ**		U18371J
CA850　Ver.3.00　**Cコンパイラ・パッケージ**	**操作編**	U17293J
	C言語編	U17291J
	アセンブリー言語編	U17292J
	リンク・ディレクティブ編	U17294J
PM+　Ver.6.30　**プロジェクト・マネージャ**		U18416J
ID850QB　Ver.3.20　**統合デバッガ**	**操作編**	U17964J
SM850　Ver.2.50　**システム・シミュレータ**	**操作編**	U16218J
SM850　Ver.2.00**以上**　**システム・シミュレータ**	**外部部品ユーザ・オープン・インタフェース仕様編**	U14873J
SM+　**システム・シミュレータ**	**操作編**	U18601J
	ユーザ・オープン・インタフェース編	U18212J
RX850　Ver.3.20　**リアルタイム**OS	**基礎編**	U13430J
	インストレーション編	U17419J
	テクニカル編	U13431J
	タスク・デバッガ編	U17420J
RX850 Pro　Ver.3.20　**リアルタイム**OS	**基礎編**	U13773J
	インストレーション編	U17421J
	テクニカル編	U13772J
	タスク・デバッガ編	U17422J
AZ850　Ver.3.30　**システム・パフォーマンス・アナライザ**		U17423J
PG-FP4　**フラッシュ・メモリ・プログラマ**		U15260J

目　次

第1章　初期化処理編

【機　　能】	CPU が最初に実施する必要のある初期化処理を行います。
【関　数　名】	JG2_init_main
【引　き　数】	なし
【処 理 内 容】	VSWC レジスタ，OCDM レジスタ，WDTM2 レジスタの設定を行います。
【使 用 S F R】	なし
【call　　関数】	JG2_init_set
【変　　数】	なし
【割 り 込 み】	なし
【割り込み要因】	なし
【フ ァ イ ル 名】	JG2_init.c
【注 意 事 項】	特定レジスタへ値を設定する場合，特定のシーケンスに従う必要があります。

【関　数　名】	JG2_init_set	
【処 理 内 容】	CPU が最初に実施する必要のあるレジスタの設定を行います。	
【使 用 S F R】	DCHC0.E00	：0（DMA0 転送禁止）
	DCHC1.E11	：0（DMA1 転送禁止）
	DCHC2.E22	：0（DMA2 転送禁止）
	DCHC3.E33	：0（DMA3 転送禁止）
	PRCMD	：0x01（コマンド・レジスタへの書き込み）
	OCDM	：0x01（動作モードをオンチップ・デバッグ・モード許可に設定）
	WDTM2	：0x00（ウォッチドッグ・タイマを動作停止に設定）
【call　　関数】	なし	
【変　　数】	なし	
【フ ァ イ ル 名】	JG2_init.c	
【注 意 事 項】	・特定レジスタへのデータ設定を行う前に，DMA 転送を禁止しておく必要があるので， このサンプル・プログラムでは DMA を転送禁止にしてあります。	

図1－1　初期化処理（1）

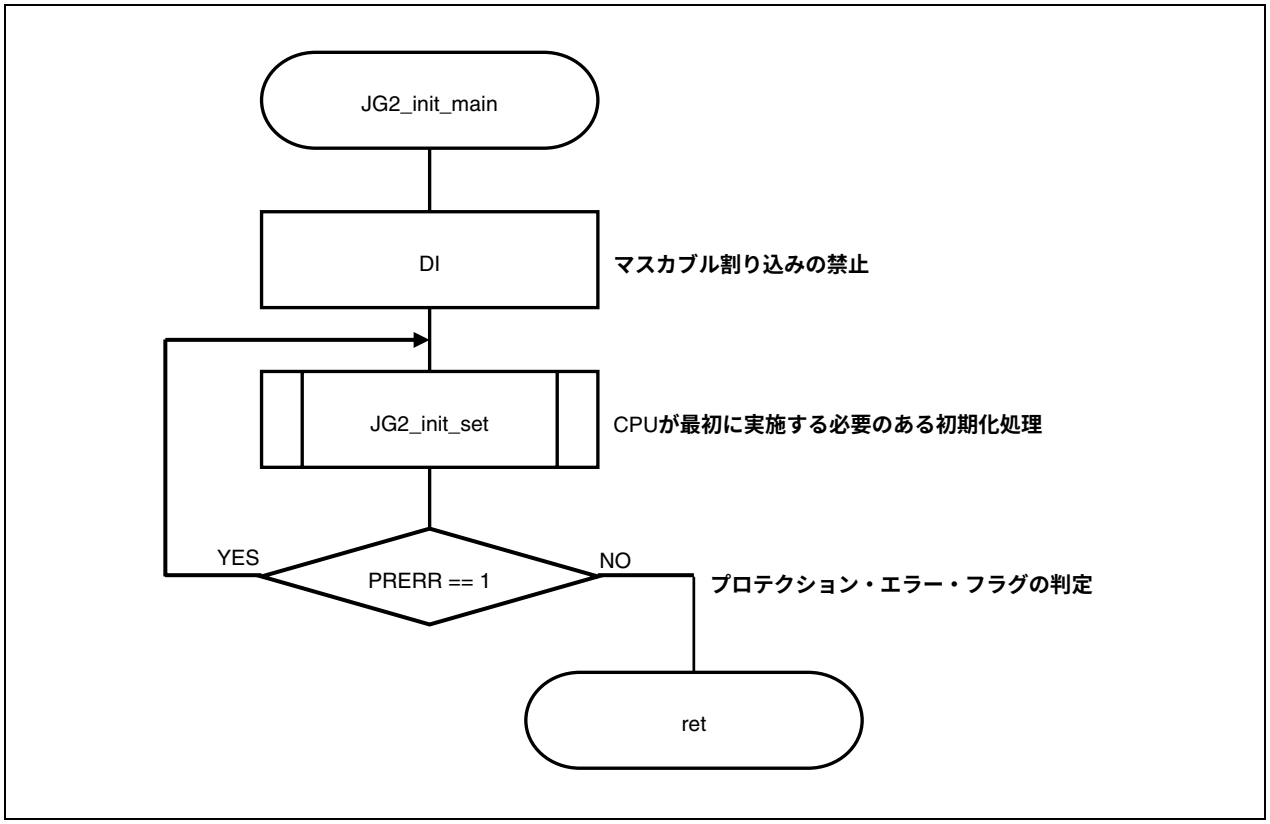
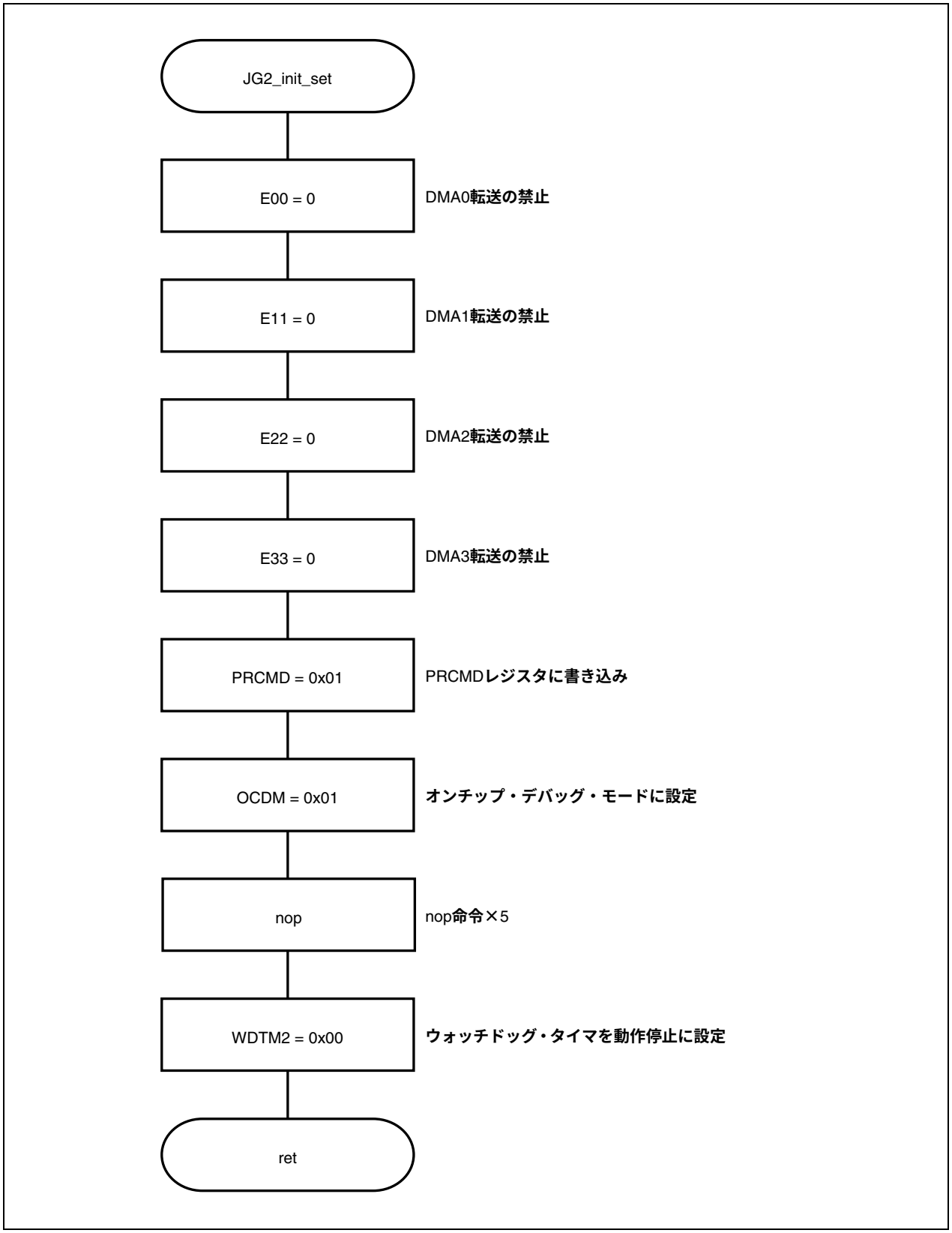

JG2_init_main
DI
マスカブル割り込みの禁止
JG2_init_set
CPUが最初に実施する必要のある初期化処理
YES
PRERR == 1
NO
プロテクション・エラー・フラグの判定
ret

図1－2　初期化処理（2）


JG2_init_set
E00 = 0
DMA0転送の禁止
E11 = 0
DMA1転送の禁止
E22 = 0
DMA2転送の禁止
E33 = 0
DMA3転送の禁止
PRCMD = 0x01
PRCMDレジスタに書き込み
OCDM = 0x01
オンチップ・デバッグ・モードに設定
nop
nop命令×5
WDTM2 = 0x00
ウォッチドッグ・タイマを動作停止に設定
ret

第2章　クロック発生機能

2. 1　PLLモード

【機　　能】	CPU 動作クロックの設定（PLL モード）と特定レジスタである PCC レジスタの設定によりクロック周波数を設定します。
【関　数　名】	clock_pll_main
【引　き　数】	なし
【処　理　内　容】	PLLCTL, PCC レジスタを設定し，PLL を使用します。
【起　動　方　法】	なし
【使 用 ＳＦＲ】	なし
【call　　関数】	clock_pll_mode，clock_pcc_mode
【変　　数】	なし
【割　り　込　み】	なし
【割り込み要因】	なし
【ファイル名】	clock_pll¥clock_pll.c
【注　意　事　項】	clock_pll_main 関数を呼び出すと EI（マスカブル割り込み要求許可）にはならないので注意してください。

【関　数　名】	clock_pll_mode
【処　理　内　容】	PLLCTL レジスタで PLL モードに設定します。
【使 用 ＳＦＲ】	PLLCTL　：0x03（PLL モード，PLL 動作に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	clock_pll¥clock_pll.c
【注　意　事　項】	なし

【関　数　名】 clock_pcc_mode

【処　理　内　容】 DMA を転送禁止にし，PCC レジスタを設定します。

【使 用 S F R】

DCHC0.E00	：0（DMA0 転送禁止）
DCHC1.E11	：0（DMA1 転送禁止）
DCHC2.E22	：0（DMA2 転送禁止）
DCHC3.E33	：0（DMA3 転送禁止）
PRCMD	：0x00（コマンド・レジスタへの書き込み）
PCC	：0x00（クロックを fxx に選択）

【call　関数】 なし

【変　数】 なし

【ファイル名】 clock_pll¥clock_pll.c

【注　意　事　項】 ・特定レジスタへのデータ設定を行う前に，DMA 転送を禁止しておく必要があるため，このサンプル・プログラムでは DMA を転送禁止にしてあります。

・PCC レジスタの設定は PLL モードに切り換えたあとに設定してください。PCC レジスタは特定レジスタなので，特定のシーケンスの組み合わせによってのみ書き込みができます。

図2－1　PLLモード（1）

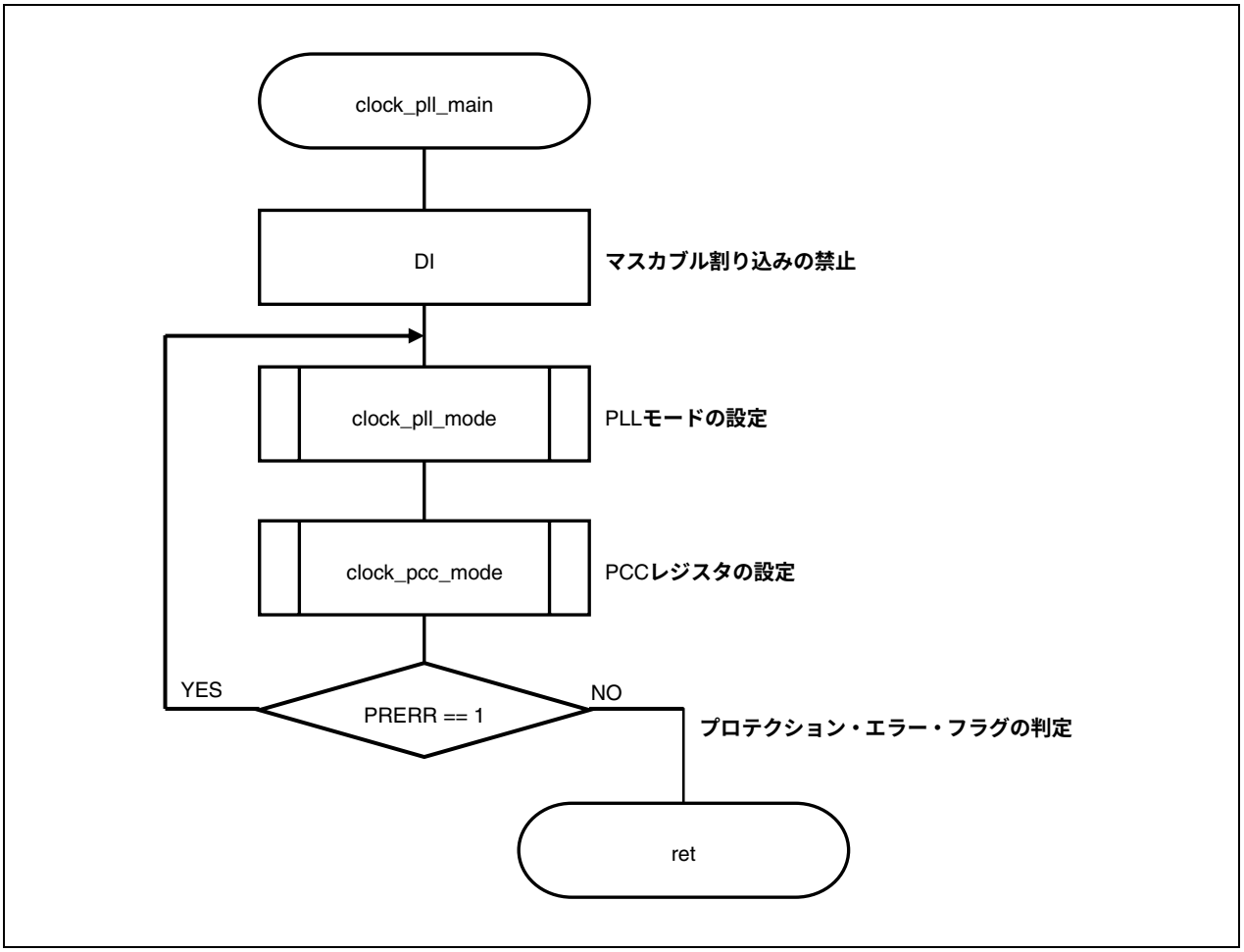
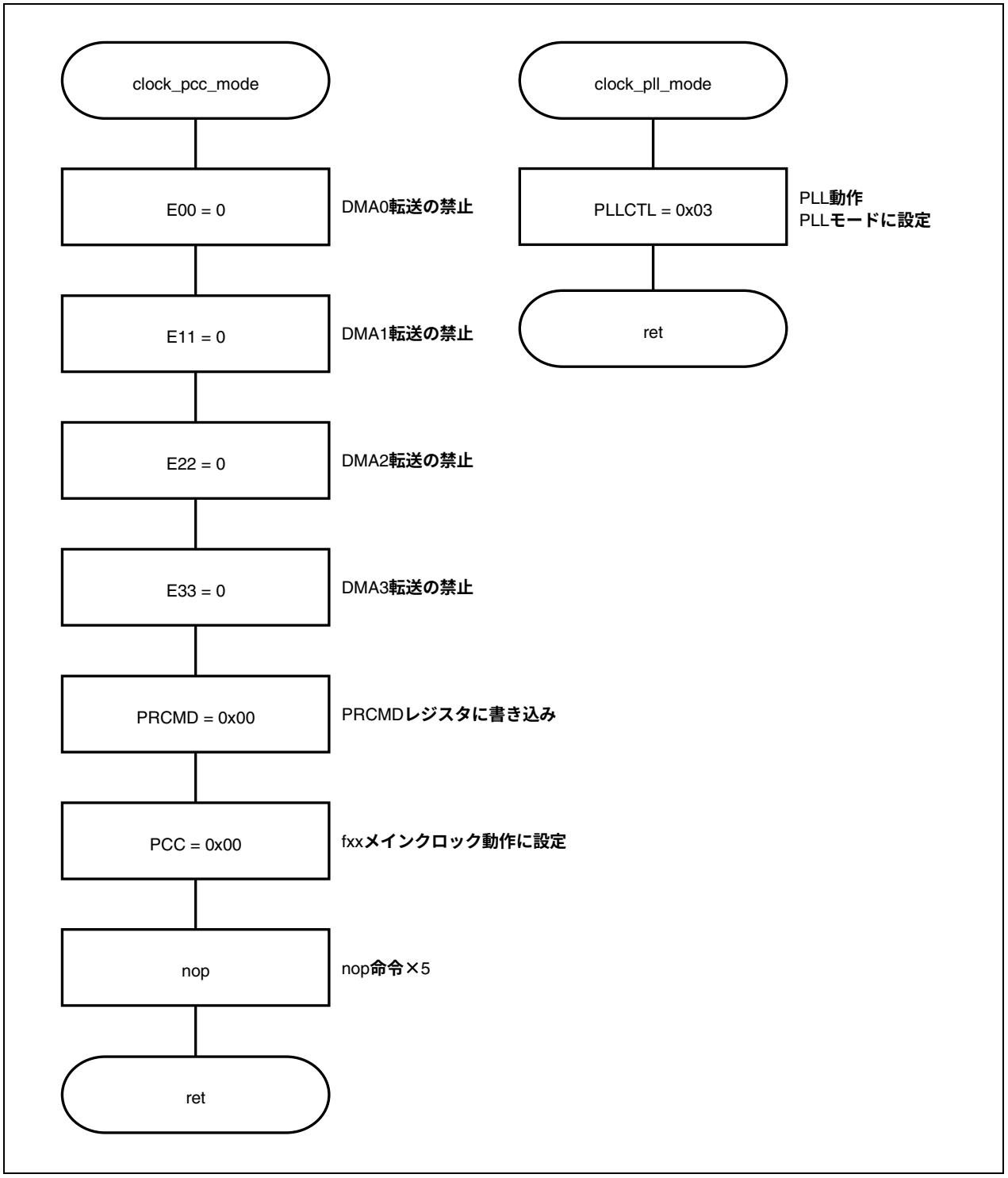


図2－2　PLLモード（2）


clock_pcc_mode
E00 = 0
DMA0転送の禁止
E11 = 0
DMA1転送の禁止
E22 = 0
DMA2転送の禁止
E33 = 0
DMA3転送の禁止
PRCMD = 0x00
PRCMDレジスタに書き込み
PCC = 0x00
fxxメインクロック動作に設定
nop
nop命令×5
ret
clock_pll_mode
PLLCTL = 0x03
PLL動作
PLLモードに設定
ret

2.2　クロック・モニタ・モード

【機　　能】　クロック・モニタ動作

【関　数　名】　clock_monitor_main

【引　き　数】　なし

【処 理 内 容】　クロック・モニタ・モード関数を呼び出し，クロック・モニタ・モードの動作を許可します。

【起 動 方 法】　なし

【使 用 S F R】　なし

【call　　関数】　clock_monitor_mode

【変　　数】　なし

【割 り 込 み】　なし

【割り込み要因】　なし

【ファイル名】　clock_monitor¥clock_monitor.c

【注 意 事 項】　一度 CLM.CLME ビット = 1 に設定した場合，リセット以外ではクリア（0）できません。

【関　数　名】　clock_monitor_mode

【処 理 内 容】　DMA 転送を禁止し，CLM レジスタでクロック・モニタの動作モードを設定します。

【使 用 S F R】

DCHC0.E00	：0（DMA0 転送禁止）
DCHC1.E11	：0（DMA1 転送禁止）
DCHC2.E22	：0（DMA2 転送禁止）
DCHC3.E33	：0（DMA3 転送禁止）
PRCMD	：0x01（コマンド・レジスタへの書き込み）
CLM	：0x01（動作許可に設定）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　clock_monitor¥clock_monitor.c

【注 意 事 項】
- 特定レジスタへのデータ設定を行う前に，DMA 転送を禁止しておく必要があるため，このサンプル・プログラムでは DMA を転送禁止にしてあります。
- CLM レジスタは特定レジスタなので，特定のシーケンスの組み合わせによってのみ書き込みができます。

図2－3　クロック・モニタ・モード（3）

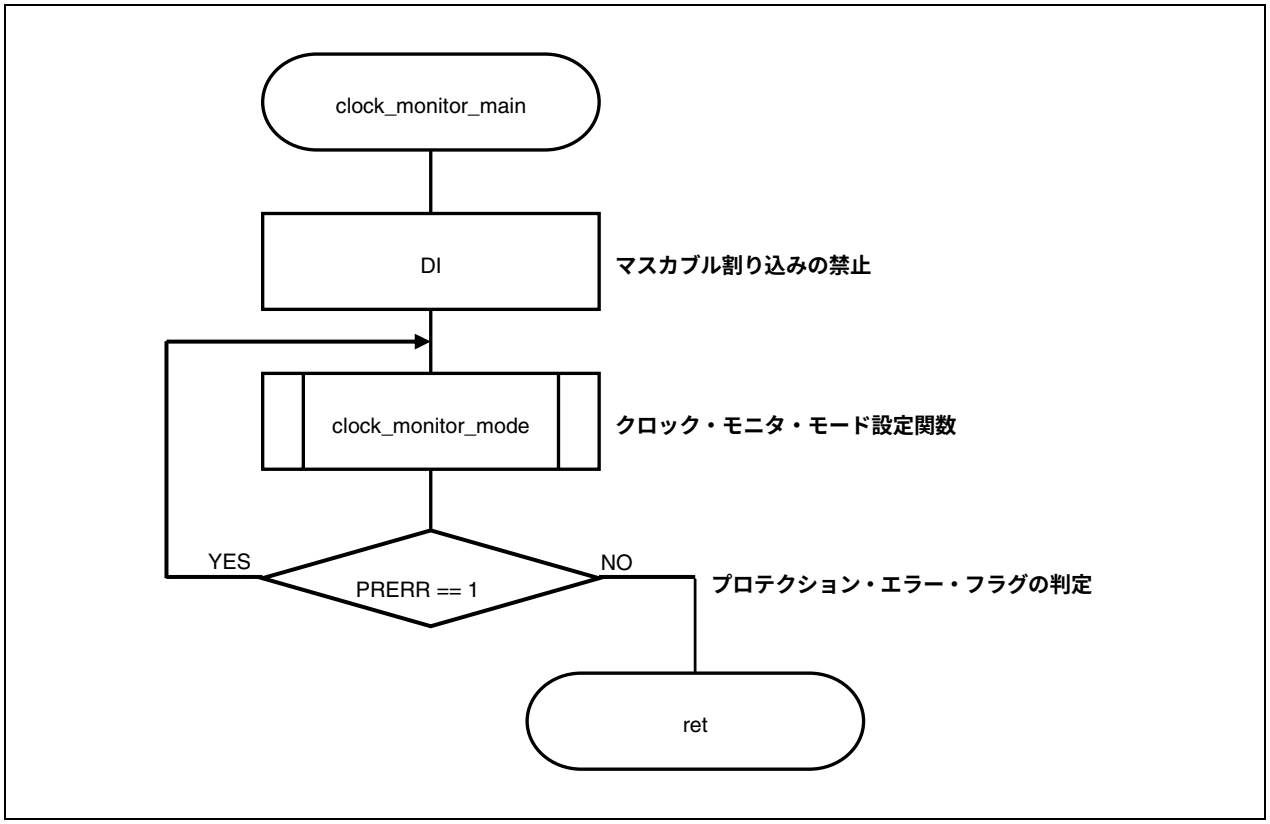
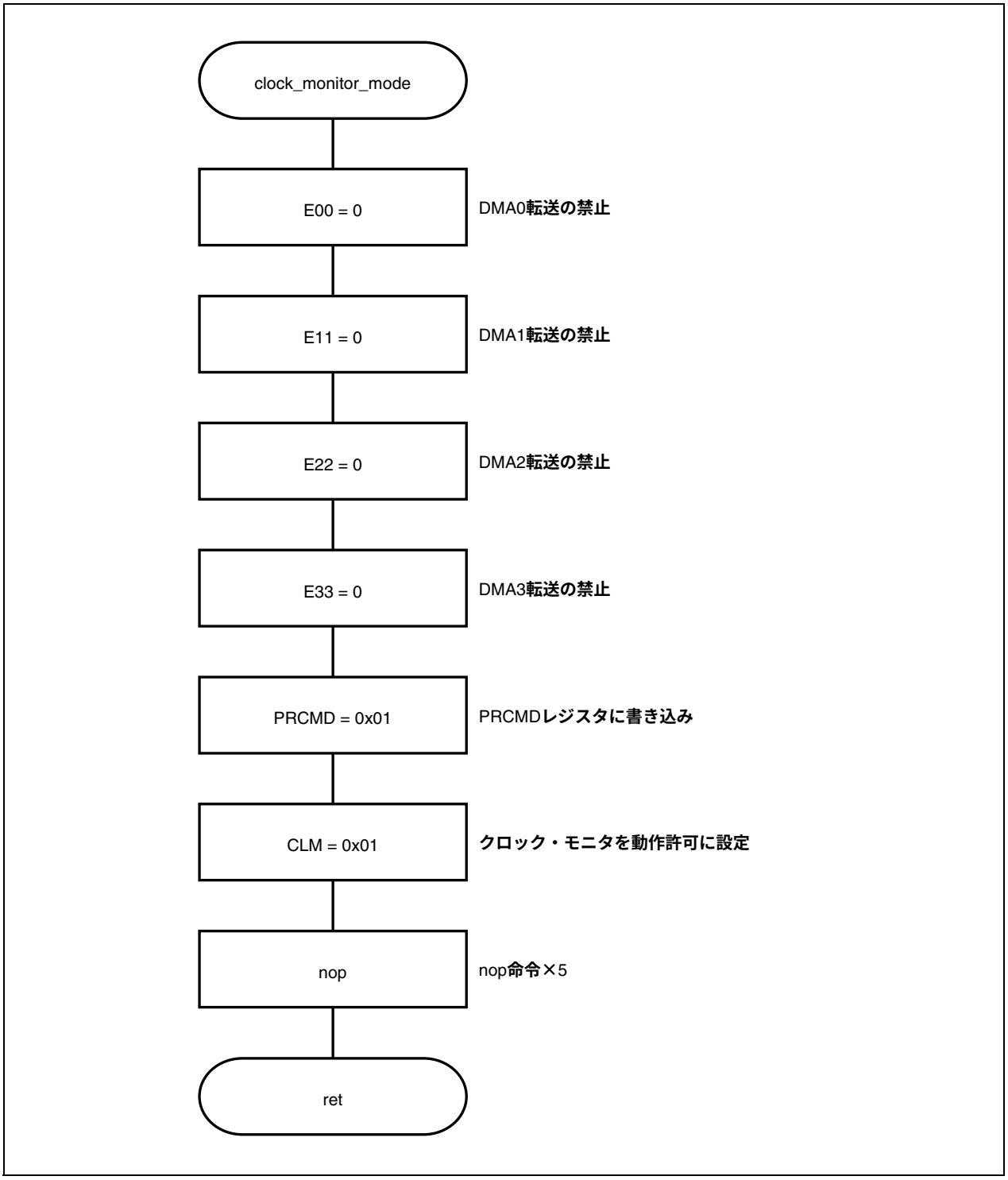


図2－4　クロック・モニタ・モード（4）



第3章　16ビット・タイマ／イベント・カウンタP（TMP）

3. 1　インターバル・タイマ・モード

【機　　能】	TP0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 TP0CCR0 レジスタで設定したインターバル間隔で割り込み要求信号（INTTP0CC0）を発生します。また，TP0CCR0 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOP00 端子出力を反転します。 TMP0-TMP5 で実現可能です。
【関　数　名】	timerp_interval_main
【引　き　数】	なし
【処 理 内 容】	fxx/32 のカウント・クロックでカウント動作を行い，カウンタの値が TP0CCR0 レジスタの値と一致した次のカウンタのタイミングで，TOP00 端子出力を反転させて割り込みを発生します。 TOP00 端子はハイ・レベル・スタートです。
【使 用 S F R】	なし
【call　　関数】	timerp_port_set，timerp_interval_set，timerp_interval_start
【変　　数】	なし
【割 り 込 み】	timerp_TP0CC0_int
【割り込み要因】	INTTP0CC0
【ファイル名】	timerp_interval¥timerp_interval.c
【注 意 事 項】	なし

【関　数　名】	timerp_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 3 を TOP00 出力端子に設定します。
【使 用 S F R】	PFC3L　：0x04（TOP00 出力端子に設定） PFCE3L　：0x04（TOP00 出力端子に設定） PMC3L　：0x04（TOP00 出力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerp_interval¥timerp_interval.c
【注 意 事 項】	なし

【関　数　名】　timerp_interval_set

【引　き　数】　なし

【処 理 内 容】　TMP0 制御レジスタの設定を行います。

【使 用 S F R】　TP0CTL0.TP0CE：0（TMP0 動作禁止）
TP0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TP0CTL1　：0x00（タイマ・モードをインターバル・タイマ・モードに設定）
TP0IOC0　：0x01（TOP00 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TP0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TP0CCMK0　：0（TMP0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_interval¥timerp_interval.c

【注 意 事 項】　なし

インターバル間隔は次に示す式で計算できます。

インターバル間隔 =　（TP0CCR0 レジスタ設定値＋1）×カウント・クロック周期

【関　数　名】　timerp_interval_start

【引　き　数】　なし

【処 理 内 容】　インターバル・タイマ・モードの起動関数です。

【起 動 方 法】　timerp_interval_set 関数のあとにコールしてください。

【使 用 S F R】　TP0CTL0.TP0CE　：1（TMP0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_interval¥timerp_interval.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】	timerp_TP0CC0_int
【概　　要】	ユーザ定義
【要　　因】	INTTP0CC0　　16 ビット・カウンタのカウント値と TP0CCR0 の一致
【call　関数】	なし
【変　　数】	なし
【ファイル名】	timerp_interval¥timerp_interval .c
【注 意 事 項】	なし

図3－1　インターバル・タイマ・モード（1）

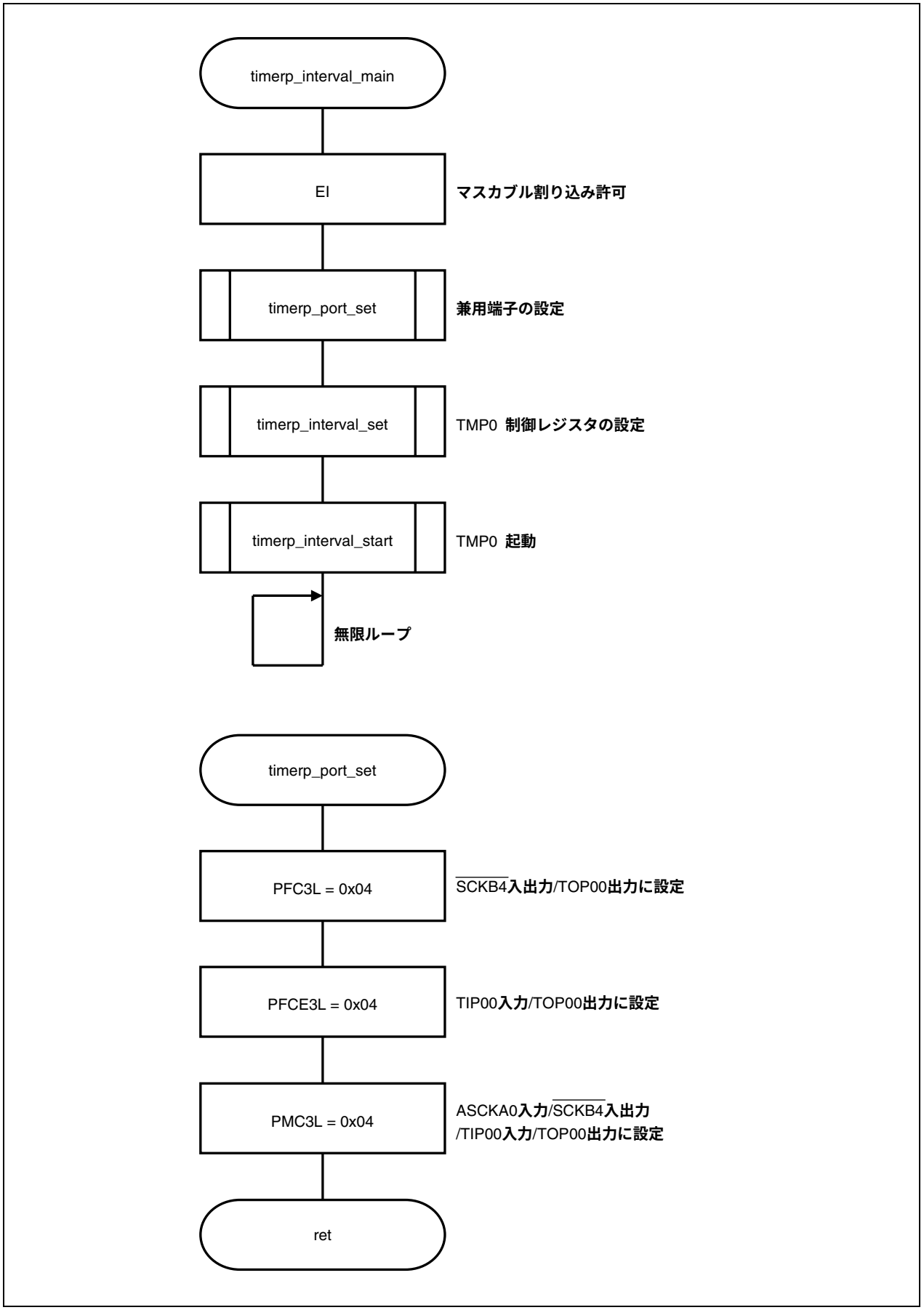
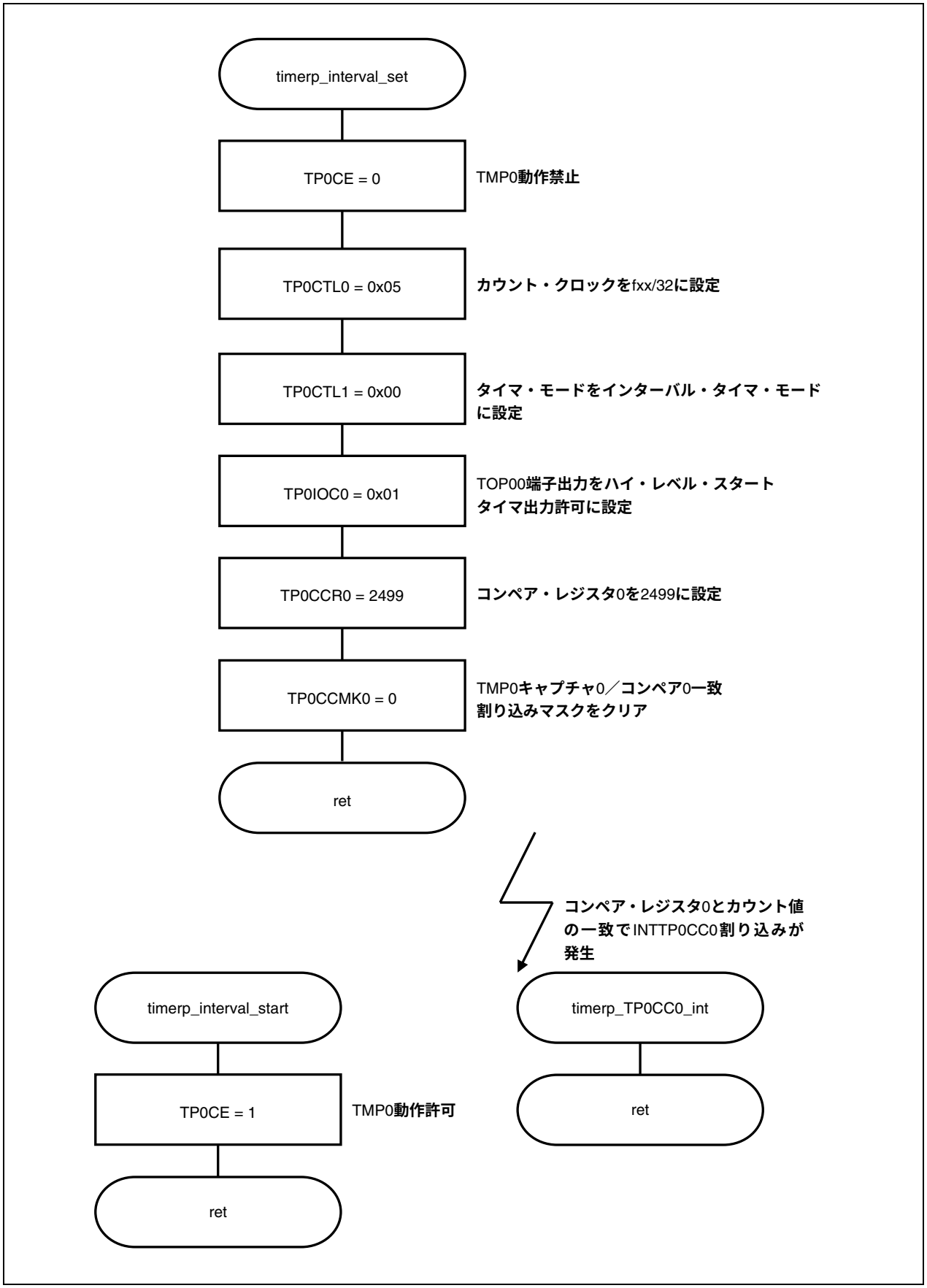


図3－2　インターバル・タイマ・モード（2）



3. 2　外部イベント・カウント・モード

【機　　能】	外部イベント・カウント入力（TIP00 端子）の有効エッジをカウントし，TP0CCR0 レジスタで設定した値とカウントが一致するごとに，割り込み要求信号（INTTP0CC0）を発生します（このとき，16 ビット・カウンタもクリアします）。 TMP0-TMP5 で実現可能です。
【関　数　名】	timerp_event_count_main
【引　き　数】	なし
【処 理 内 容】	外部イベント・カウント入力の有効エッジをカウントし，カウンタの値が TP0CCR0 レジスタの値と一致した次のカウント（TP0CCR0 レジスタに設定した値＋1 回）のタイミングで，割り込みを発生し，カウンタをクリアします。
【使 用 S F R】	なし
【call　　関数】	timerp_port_set，timerp_event_count_set，timerp_event_count_start
【変　　数】	なし
【割 り 込 み】	timerp_TP0CC0_int
【割り込み要因】	INTTP0CC0
【ファイル名】	timerp_event_count¥timerp_event_count.c,
【注 意 事 項】	なし

【関　数　名】	timerp_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 3 を TIP00 入力端子に設定します。
【使 用 S F R】	PFCE3L　：0x04（TIP00 入力端子に設定） PMC3L　：0x04（TIP00 入力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerp_interval¥timerp_event_count.c
【注 意 事 項】	なし

【関　数　名】　timerp_event_count_set

【引　き　数】　なし

【処 理 内 容】　TMP0 制御レジスタの設定を行います。

【使 用 S F R】　TP0CTL0.TP0CE：0（TMP0 動作禁止）
TP0CTL0　：0x00（カウント・クロックを fxx に設定）
TP0CTL1　：0x01（タイマ・モードを外部イベント・カウント・モードに設定）
TP0IOC2　：0x08（TIP00 端子の有効エッジを立ち下がりエッジを検出に設定）
TP0CCR0　：40（16 ビット・カウンタのコンペア・レジスタ 0 を 40 に設定）
TP0CCMK0　：0（TMP0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_event_count¥timerp_event_count.c

【注 意 事 項】　なし

【関　数　名】　timerp_event_count_start

【引　き　数】　なし

【処 理 内 容】　外部イベント・カウント・モードの起動関数です。

【起 動 方 法】　timerp_event_count_set 関数のあとにコールしてください。

【使 用 S F R】　TP0CTL0.TP0CE　：1（TMP0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_event_count¥ timerp_event_count.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】　timerp_TP0CC0_int

【概　要】　ユーザ定義

【要　因】　INTTP0CC0　16 ビット・カウンタのカウント値と TP0CCR0 の一致

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_event_count¥ timerp_event_count.c

【注 意 事 項】　なし

図3－3　外部イベント・カウント・モード（1）

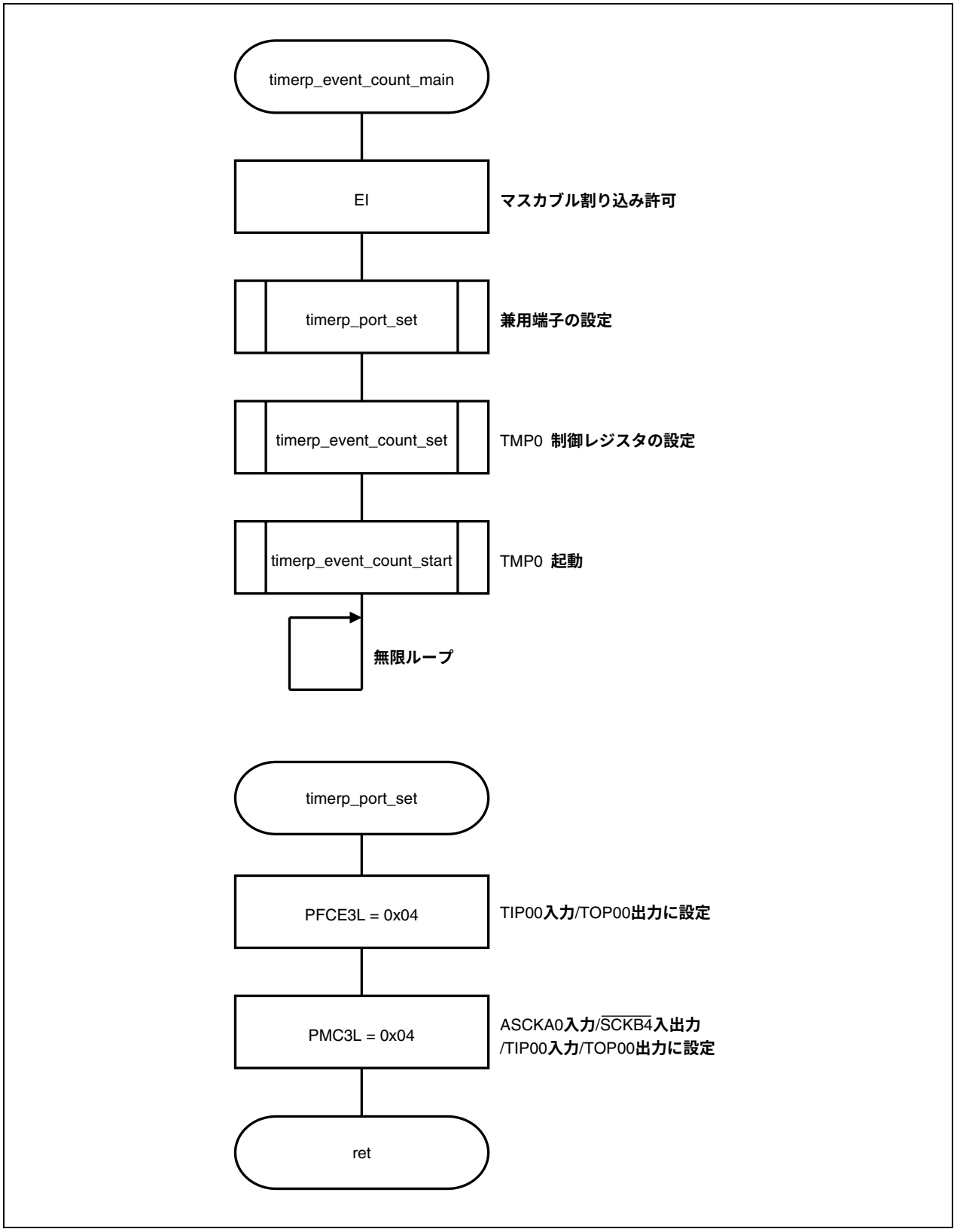
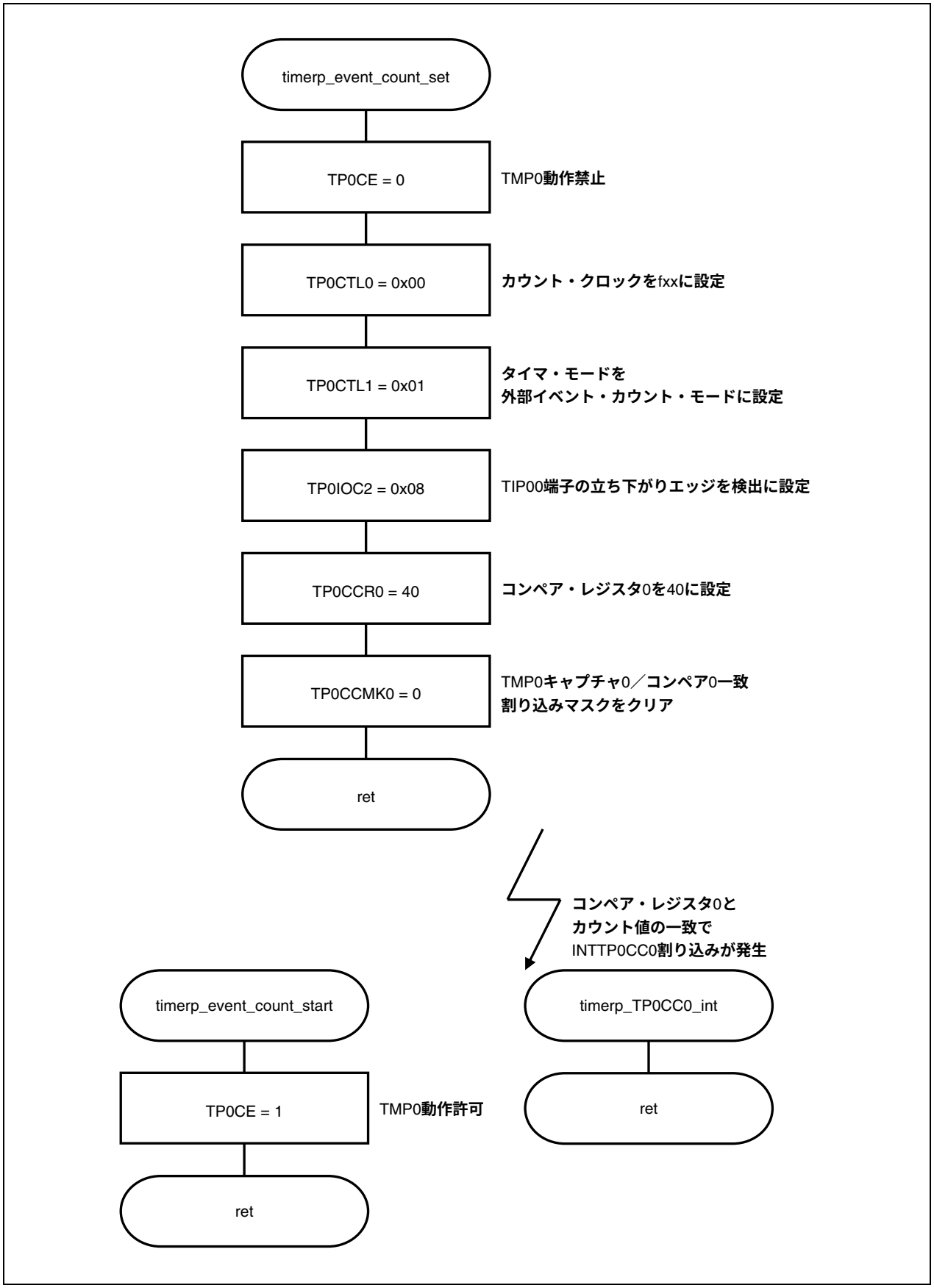


図3－4　外部イベント・カウント・モード（2）



3. 3　外部トリガ・パルス出力モード

【機　　能】	外部トリガ入力（TIP00 端子）の有効エッジを検出すると 16 ビット・カウンタの動作を開始します。 TP0CCR0 レジスタとのコンペア一致で 16 ビット・カウンタをクリアします。 TP0CCR0 レジスタ，TP0CCR1 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOP01 端子から PWM 波形を出力します。 再度トリガが発生した場合には，カウンタを 0000H にクリアし再スタートします。 TMP0-TMP5 で実現可能です。
【関　数　名】	timerp_trigger_pulse_main
【引　き　数】	なし
【処 理 内 容】	外部トリガ入力の有効エッジ検出で，fxx/32 のカウント・クロックのカウント動作を開始し，カウンタの値が TP0CCR0 レジスタの値と一致した次のカウントのタイミングで TOP01 端子出力を反転させて割り込みを発生し，カウンタをクリアします。 また，カウンタの値が TP0CCR1 レジスタの値と一致した次のカウントのタイミングで TOP01 端子出力を反転させて割り込みを発生します。 TOP01 端子はハイ・レベル・スタートです。
【使 用 S F R】	なし
【call　　関数】	timerp_port_set，timerp_trigger_pulse_set，timerp_trigger_pulse_start
【変　　数】	なし
【割 り 込 み】	timerp_TP0CC0_int timerp_TP0CC1_int
【割り込み要因】	INTTP0CC0 INTTP0CC1
【ファイル名】	timerp_trigger_pulse¥timerp_trigger_pulse.c
【注 意 事 項】	なし

【関　数　名】	timerp_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 3 を TOP01 出力端子，TIP00 入力端子に設定します。
【使 用 S F R】	PFC3L　：0x08（TOP01 出力端子，TIP00 入力端子に設定） PFCE3L　：0x04（TOP01 出力端子，TIP00 入力端子に設定） PMC3L　：0x0C（TIP00 入力端子，TOP01 出力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerp_trigger_pulse¥timerp_ trigger_pulse.c
【注 意 事 項】	なし

【関　数　名】　timerp_trigger_pulse_set

【引　き　数】　なし

【処　理　内　容】　TMP0 制御レジスタの設定を行います。

【使 用 S F R】　TP0CTL0.TP0CE：0（TMP0 動作禁止）
TP0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TP0CTL1　：0x02（タイマ・モードを外部トリガ・パルス出力モードに設定）
TP0IOC0　：0x04（TOP01 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TP0IOC2　：0x08（TIP00 端子の有効エッジを立ち下がりエッジを検出に設定）
TP0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TP0CCR1　：1249（16 ビット・カウンタのコンペア・レジスタ 1 を 1249 に設定）
TP0CCMK0　：0（TMP0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TP0CCMK1　：0（TMP0 キャプチャ 1／コンペア 1 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_trigger_pulse¥timerp_trigger_pulse.c

【注　意　事　項】　なし

PWM波形のアクティブ・レベル幅，周期，およびデューティは次に示す式で計算できます。

アクティブ・レベル幅 =（TP0CCR1 レジスタの設定値）×カウント・クロック周期

周期 =（TP0CCR0 レジスタの設定値＋1）×カウント・クロック周期

デューティ =（TP0CCR1 レジスタの設定値）／（TP0CCR0 レジスタの設定値＋1）

【関　数　名】　timerp_trigger_pulse_start

【引　き　数】　なし

【処　理　内　容】　外部トリガ・パルス出力モードの起動関数です。

【起　動　方　法】　timerp_trigger_pulse_set 関数のあとにコールしてください。

【使 用 S F R】　TP0CTL0.TP0CE　：1（TMP0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_trigger_pulse¥timerp_trigger_pulse.c

【注　意　事　項】　なし

割り込み関数

【関　数　名】 timerp_TP0CC0_int

【概　　要】 ユーザ定義

【要　　因】 INTTP0CC0　　16 ビット・カウンタのカウント値と TP0CCR0 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerp_trigger_pulse¥timerp_trigger_pulse.c

【注 意 事 項】 なし

【関　数　名】 timerp_TP0CC1_int

【概　　要】 ユーザ定義

【要　　因】 INTTP0CC1　　16 ビット・カウンタのカウント値と TP0CCR1 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerp_trigger_pulse¥timerp_trigger_pulse.c

【注 意 事 項】 なし

図3－5　外部トリガ・パルス出力モード（1）

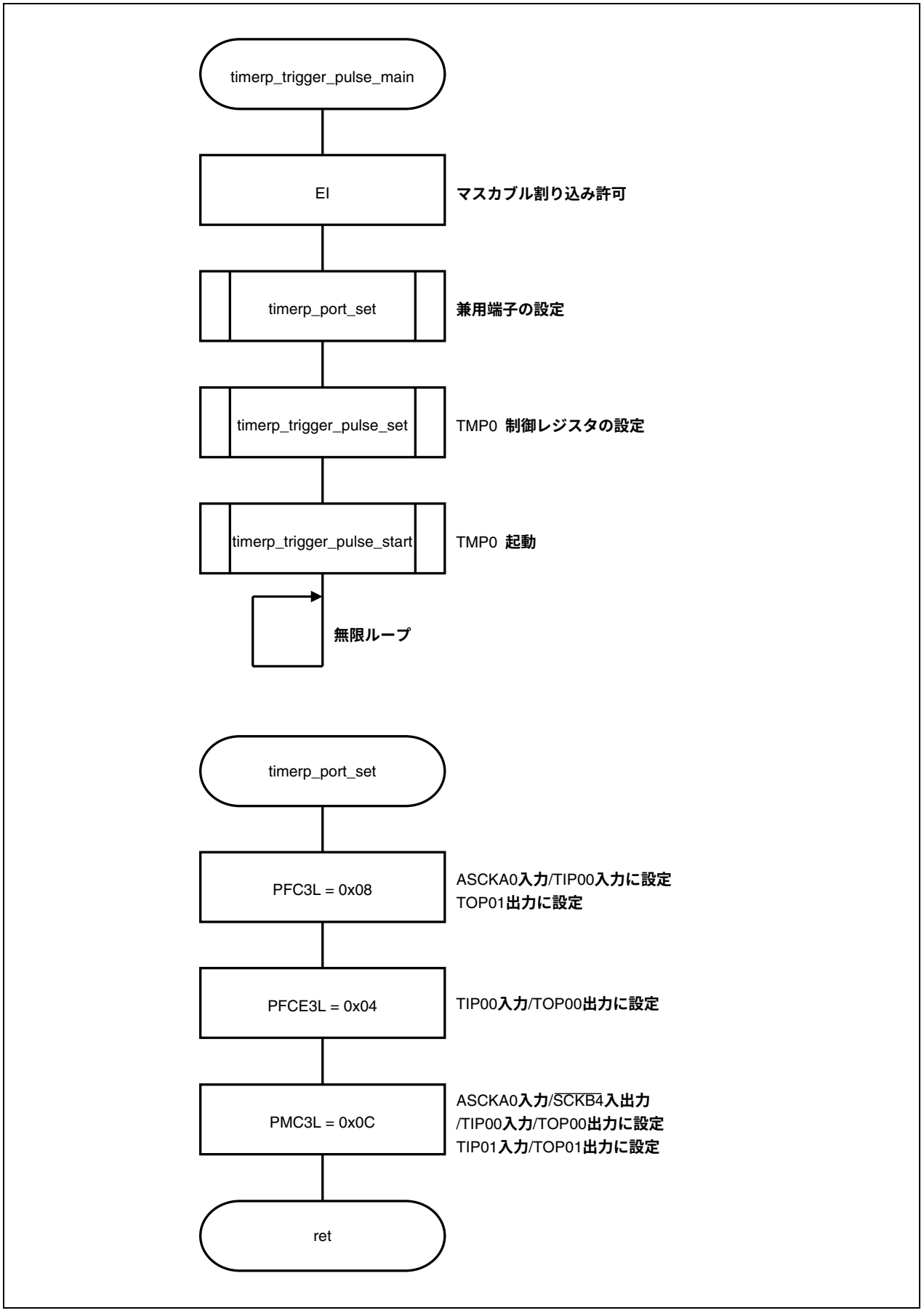


図3－6　外部トリガ・パルス出力モード（2）

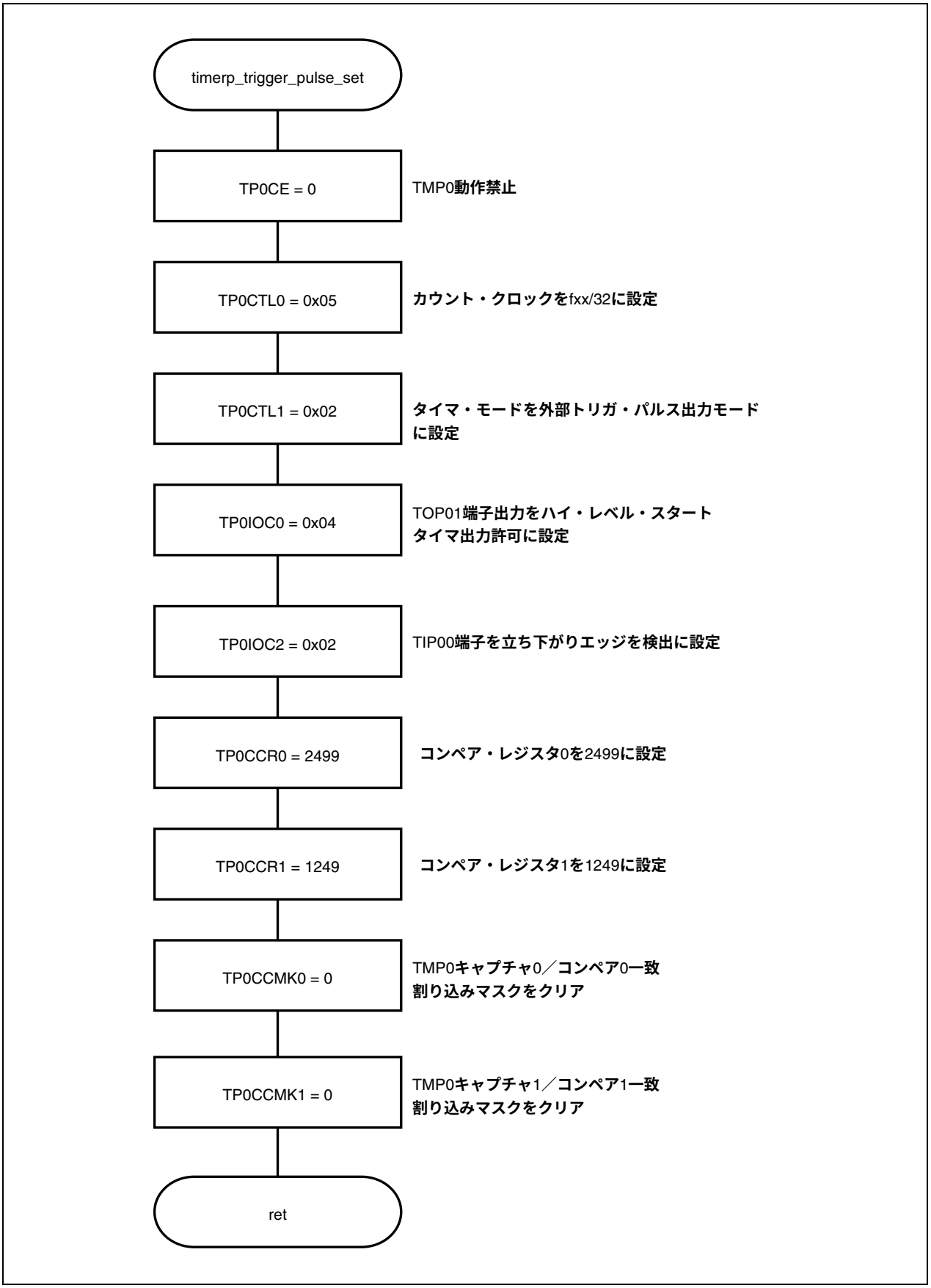
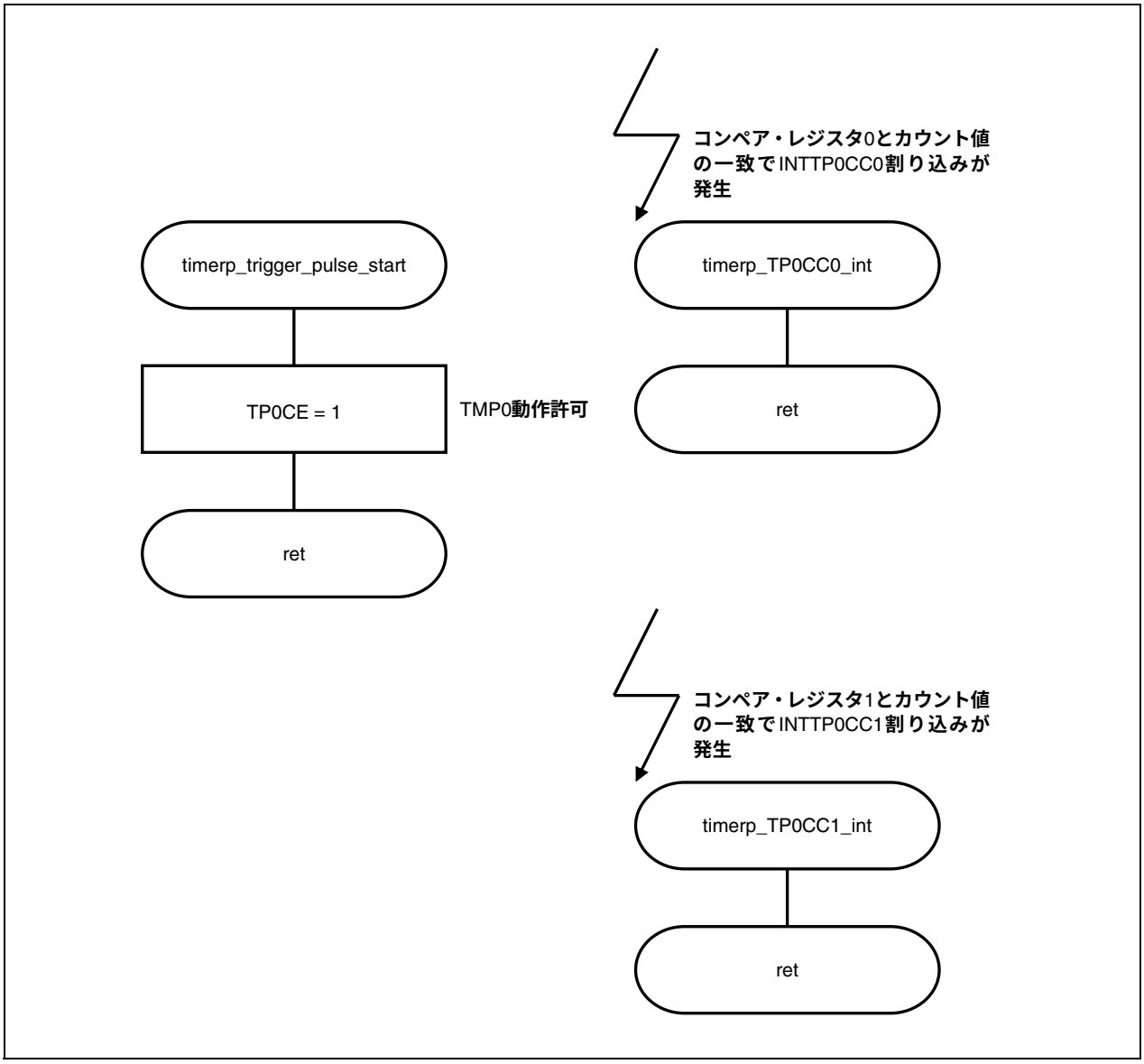


図3－7　外部トリガ・パルス出力モード（3）



3. 4　ワンショット・パルス出力モード

【機　　能】	外部トリガ入力（TIP00 端子）の有効エッジを検出すると 16 ビット・カウンタの動作を開始します。 TP0CCR1 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOP01 端子の出力をアクティブ・レベルにします。TP0CCR0 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOP01 端子の出力を反転させ，ワンショット・パルスを出力します。ワンショット・パルスを出力したあと，カウントを停止し，トリガ待ち状態となります。 ワンショット・パルス出力中に再度トリガが発生しても無視します。 TMP0-TMP5 で実現可能です。
【関　数　名】	timerp_1shot_pulse_main
【引　き　数】	なし
【処 理 内 容】	外部トリガ入力の有効エッジ検出で，fxx/32 のカウント・クロックのカウント動作を開始し，カウンタの値が TP0CCR0 レジスタの値と一致した次のカウントのタイミングで割り込みを発生し，TOP01 端子出力を反転させてカウンタをクリアし，カウント動作を停止します。また，カウンタの値が TP0CCR1 レジスタの値と一致した次のカウントのタイミングで TOP01 端子出力を反転させて割り込みを発生します。 TOP01 端子はハイ・レベル・スタートです。
【使 用 ＳＦＲ】	なし
【call　関数】	timerp_port_set，timerp_1shot_pulse_set，timerp_1shot_pulse_start
【変　　数】	なし
【割 り 込 み】	timerp_TP0CC0_int timerp_TP0CC1_int
【割り込み要因】	INTTP0CC0 INTTP0CC1
【ファイル名】	timerp_1shot_pulse¥timerp_1shot_pulse.c,
【注 意 事 項】	なし

【関　数　名】 timerp_port_set

【引　き　数】 なし

【処 理 内 容】 ポート 3 を TOP01 出力端子，TIP00 入力端子に設定します。

【使 用 S F R】 PFC3L　：0x08（TOP01 出力端子，TIP00 入力端子に設定）
PFCE3L　：0x04（TOP01 出力端子，TIP00 入力端子に設定）
PMC3L　：0x0C（TOP01 出力端子，TIP00 入力端子に設定）

【call　関数】 なし

【変　数】 なし

【ファイル名】 timerp_1shot_pulse¥timerp_1shot_pulse.c

【注 意 事 項】 なし

【関　数　名】 timerp_1shot_pulse_set

【引　き　数】 なし

【処 理 内 容】 TMP0 制御レジスタの設定を行います。

【使 用 S F R】 TP0CTL0.TP0CE：0（TMP0 動作禁止）
TP0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TP0CTL1　：0x03（タイマ・モードをワンショット・パルスモードに設定）
TP0IOC0　：0x04（TOP01 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TP0IOC2　：0x08（TIP00 端子の有効エッジを立ち下がりエッジを検出に設定）
TP0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TP0CCR1　：1249（16 ビット・カウンタのコンペア・レジスタ 1 を 1249 に設定）
TP0CCMK0　：0（TMP0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TP0CCMK1　：0（TMP0 キャプチャ 1／コンペア 1 一致割り込みマスク・クリアに設定）

【call　関数】 なし

【変　数】 なし

【ファイル名】 timerp_1shot_pulse¥timerp_1shot_pulse.c

【注 意 事 項】 なし

ワンショット・パルスの出力ディレイ期間，およびアクティブ・レベル幅は次に示す式で計算できます。

出力ディレイ期間 = （TP0CCR1 レジスタの設定値）×カウント・クロック周期

アクティブ・レベル幅 = （TP0CCR0 レジスタの設定値－TP0CCR1 レジスタの設定値+1）×カウント・クロック周期

【関　数　名】　timerp_1shot_pulse_start

【引　き　数】　なし

【処 理 内 容】　ワンショット・パルス出力モードの起動関数です。

【起 動 方 法】　timerp_1shot_pulse_set 関数のあとにコールしてください。

【使 用 S F R】　TP0CTL0.TP0CE　：1（TMP0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_1shot_pulse¥timerp_1shot_pulse.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】　timerp_TP0CC0_int

【概　要】　ユーザ定義

【要　因】　INTTP0CC0　16 ビット・カウンタのカウント値と TP0CCR0 の一致

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_1shot_pulse¥timerp_1shot_pulse.c

【注 意 事 項】　なし

【関　数　名】　timerp_TP0CC1_int

【概　要】　ユーザ定義

【要　因】　INTTP0CC1　16 ビット・カウンタのカウント値と TP0CCR1 の一致

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_1shot_pulse¥timerp_1shot_pulse.c

【注 意 事 項】　なし

図3－8　ワンショット・パルス出力モード（1）

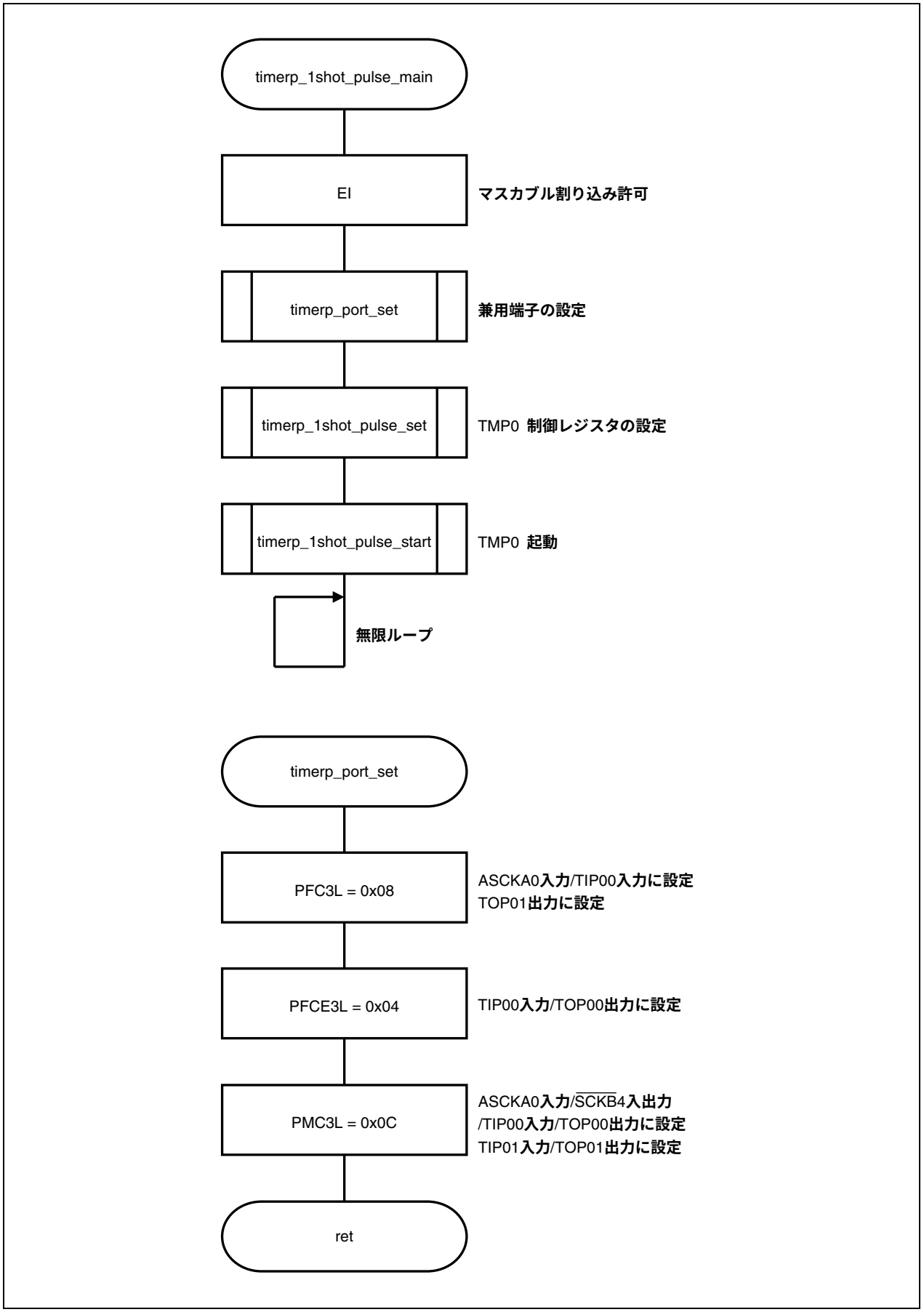


図3－9　ワンショット・パルス出力モード（2）

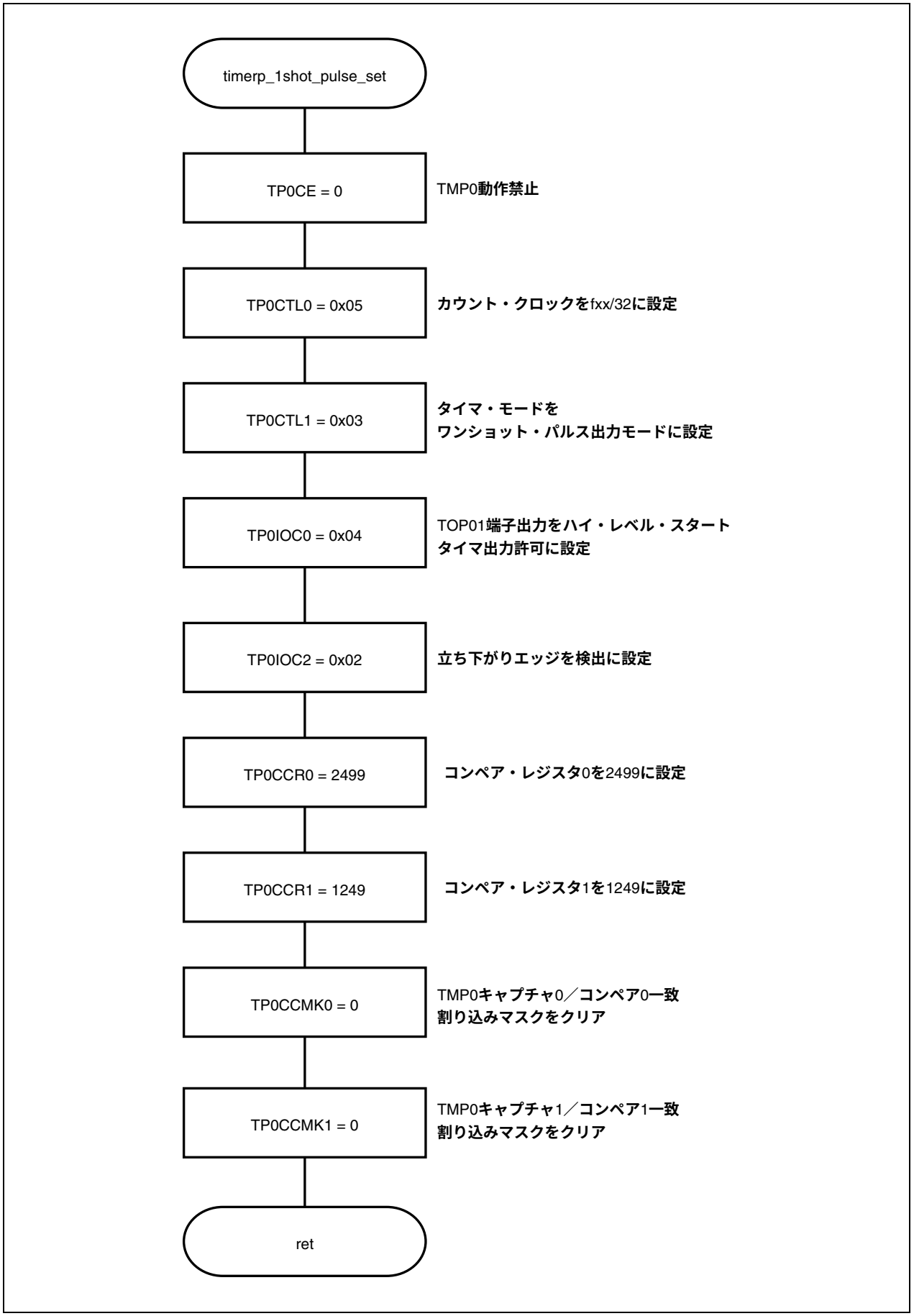
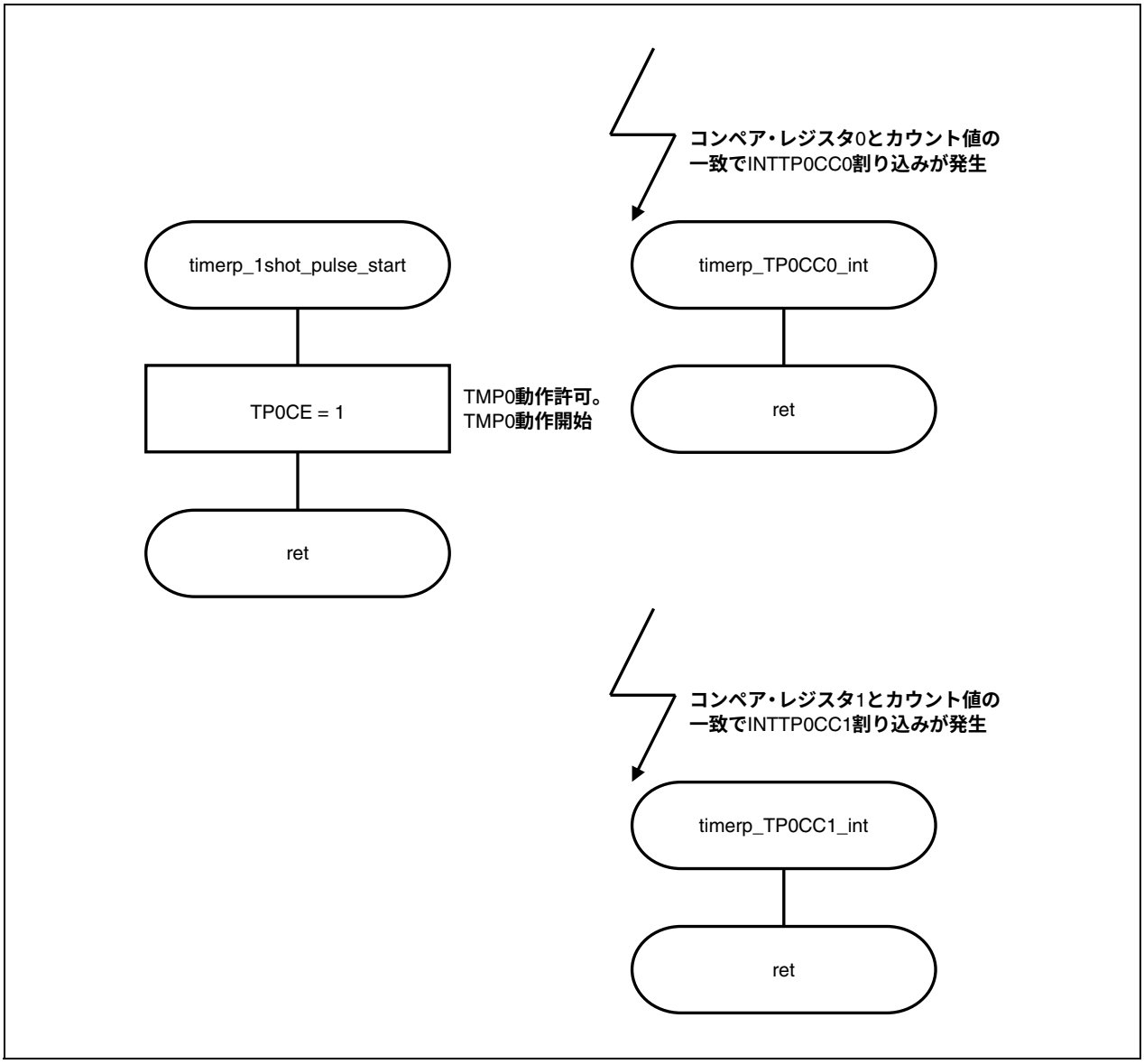


図3－10　ワンショット・パルス出力モード（3）



3. 5　PWM出力モード

【機　　能】	TP0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 TP0CCR0 レジスタとのコンペア一致で 16 ビット・カウンタをクリアし，TOP00 端子を反転し，TP0CCR0 レジスタの値＋1 を半周期とする 50 %（TP0CCR1 レジスタの設定値）／（TP0CCR0 レジスタの設定値＋1）デューティの PWM 波形を出力します。 TP0CCR1 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOP01 端子を反転します。 TOP01 端子は 16 ビット・カウンタのクリア時に反転します。 TMP0-TMP5 で実現可能です。
【関　数　名】	timerp_pwm_output_main
【引　き　数】	なし
【処 理 内 容】	fxx/32 のカウント・クロックでカウント動作を行い，カウンタの値が TP0CCR0 レジスタの値と一致した次のカウントのタイミングで TOP00, TOP01 端子出力を反転させて割り込みを発生し，カウンタをクリアします。 また，カウンタの値が TP0CCR1 レジスタの値と一致した次のカウントのタイミングで TOP01 端子出力を反転させて割り込みを発生します。 TOP00, TOP01 端子ともにハイ・レベル・スタートです。
【使 用 ＳＦＲ】	なし
【call　関数】	timerp_port_set，timerp_pwm_output_set，timerp_pwm_output_start
【変　　数】	なし
【割 り 込 み】	timerp_TP0CC0_int timerp_TP0CC1_int
【割り込み要因】	INTTP0CC0 INTTP0CC1
【ファイル名】	timerp_pwm_output¥timerp_pwm_output.c
【注 意 事 項】	なし

【関　数　名】　timerp_port_set

【引　き　数】　なし

【処　理　内　容】　ポート 3 を TOP00 出力端子，TOP01 出力端子に設定します。

【使 用 S F R】　PFC3L　：0x0C（TOP00 出力端子，TOP01 出力端子に設定）
PFCE3L　：0x04（TOP00 出力端子，TOP01 出力端子に設定）
PMC3L　：0x0C（TOP00 出力端子，TOP01 出力端子に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_pwm_output ¥timerp_pwm_output.c

【注　意　事　項】　なし

【関　数　名】　timerp_pwm_output_set

【引　き　数】　なし

【処　理　内　容】　TMP0 制御レジスタの設定を行います。

【使 用 S F R】　TP0CTL0.TP0CE：0（TMP0 動作禁止）
TP0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TP0CTL1　：0x04（タイマ・モードをワンショット・パルスモードに設定）
TP0IOC0　：0x05（TOP00 端子出力をハイ・レベル・スタート，タイマ出力許可，
TOP01 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TP0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TP0CCR1　：1249（16 ビット・カウンタのコンペア・レジスタ 1 を 1249 に設定）
TP0CCMK0　：0（TMP0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TP0CCMK1　：0（TMP0 キャプチャ 1／コンペア 1 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_pwm_output ¥timerp_pwm_output.c

【注　意　事　項】　なし

TOP01端子から出力されるPWM波形のアクティブ・レベル幅，周期，およびデューティは，次に示す式で計算できます。

アクティブ・レベル幅 = （TP0CCR1 レジスタの設定値）×カウント・クロック周期

周期 = （TP0CCR0 レジスタの設定値＋1）×カウント・クロック周期

デューティ = （TP0CCR1 レジスタの設定値）／（TP0CCR0 レジスタの設定値＋1）

【関　数　名】 timerp_pwm_output_start

【引　き　数】 なし

【処 理 内 容】 PWM 出力モードの起動関数です。

【起 動 方 法】 timerp_pwm_output_set 関数のあとにコールしてください。

【使 用 S F R】 TP0CTL0.TP0CE　：1（TMP0 動作許可）

【call　　関数】 なし

【変　　数】 なし

【ファイル名】 timerp_pwm_output ¥timerp_pwm_output.c

【注 意 事 項】 なし

割り込み関数

【関　数　名】 timerp_TP0CC0_int

【概　　要】 ユーザ定義

【要　　因】 INTTP0CC0　　16 ビット・カウンタのカウント値と TP0CCR0 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerp_pwm_output¥timerp_pwm_output.c

【注 意 事 項】 なし

【関　数　名】 timerp_TP0CC1_int

【概　　要】 ユーザ定義

【要　　因】 INTTP0CC1　　16 ビット・カウンタのカウント値と TP0CCR1 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerp_pwm_output¥timerp_pwm_output.c

【注 意 事 項】 なし

図3－11　PWM出力モード（1）

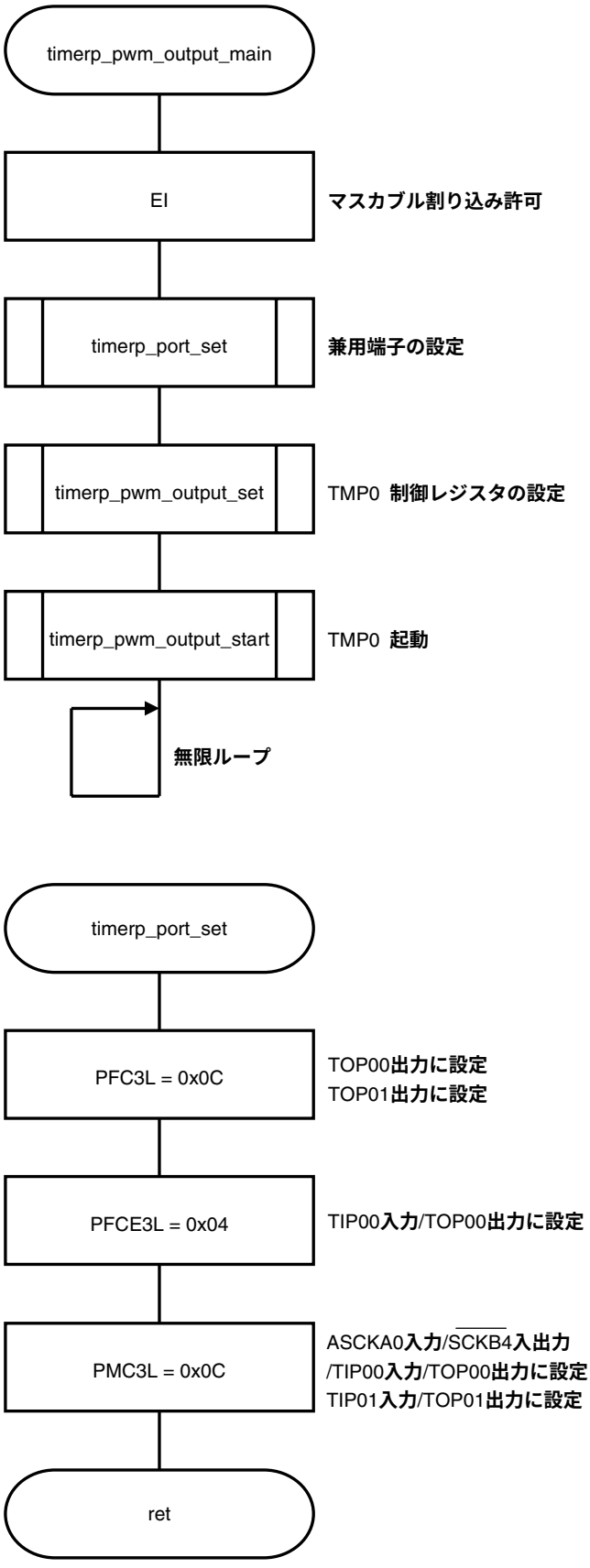


図3－12　PWM出力モード（2）

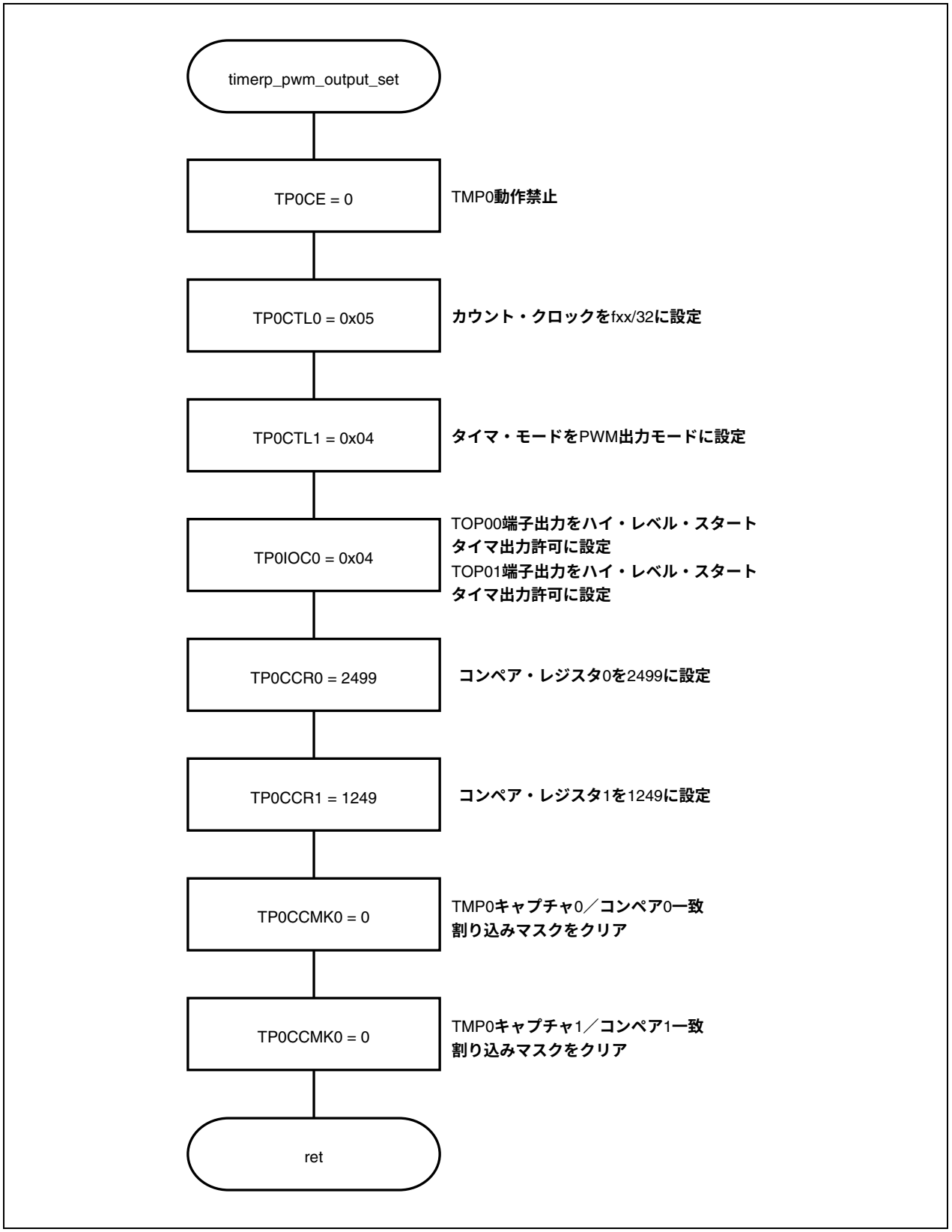
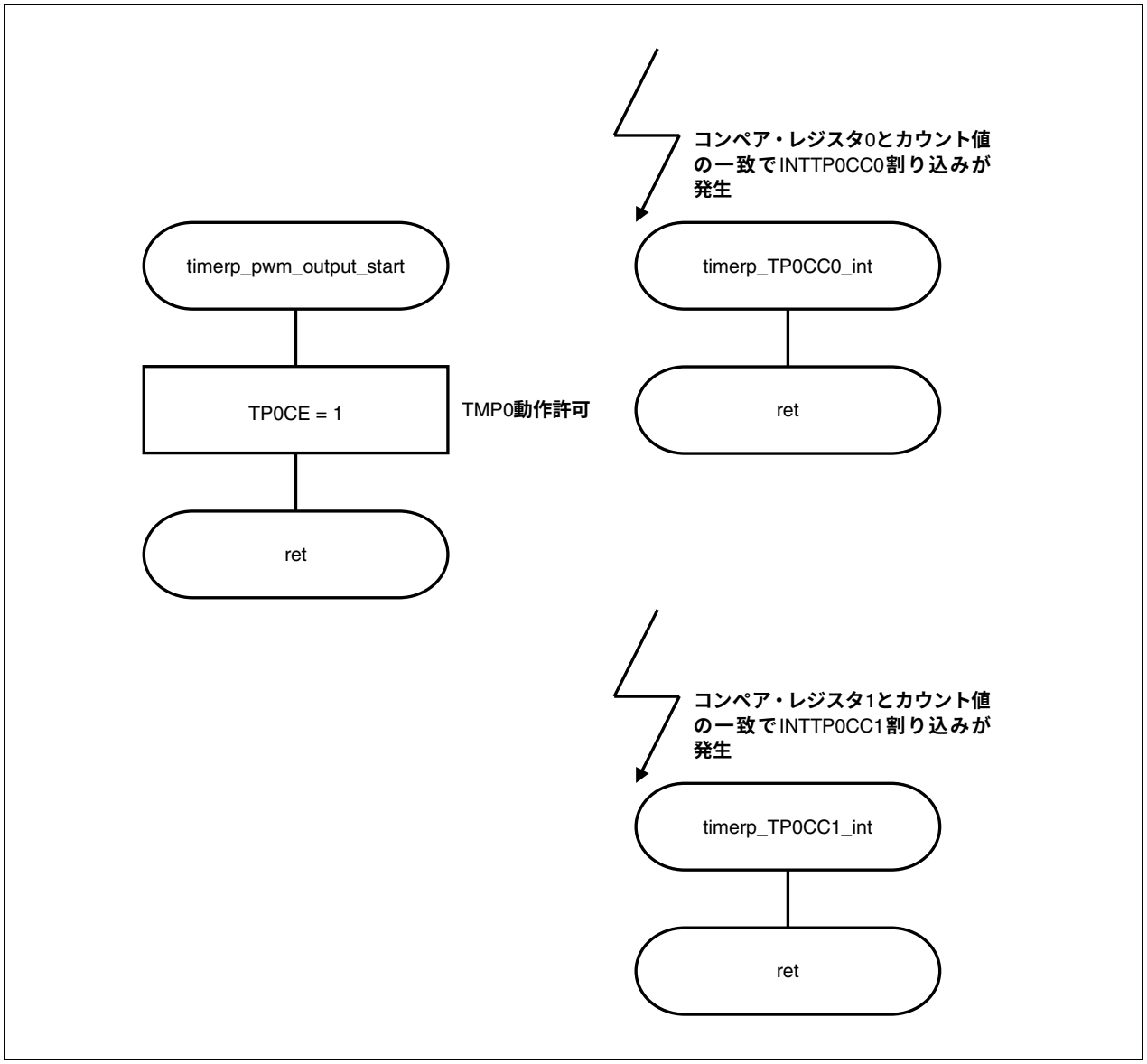


図3－13　PWM出力モード（3）



3. 6　フリー・ランニング・タイマ・モード

【機　　能】	TP0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 TP0CCR0 レジスタと 16 ビット・カウンタのカウント値とのコンペア一致で TOP00 端子の出力を反転します（コンペア機能）。TP0CCR0 レジスタと 16 ビット・カウンタのカウント値が一致してもカウント値はクリアされません。FFFFH までカウントすると，次のクロックでオーバフロー割り込み要求信号（INTTP0OV）を発生するとともに，0000H にクリアしカウント動作を継続します。 また，キャプチャ・トリガ入力（TIP01 端子）の有効エッジ検出で 16 ビット・カウンタのカウント値を格納します（キャプチャ機能）。 キャプチャ機能は TMP0-TMP5 で実現可能です。
【関　数　名】	timerp_free_running_main
【引　き　数】	なし
【処 理 内 容】	fxx/32 のカウント・クロックでカウント動作を行い，カウンタの値が TP0CCR0 レジスタの値と一致した次のカウントのタイミングで TOP00 端子出力を反転させて割り込みを発生し，カウンタをクリアします。 また，TIP01 端子からの有効エッジ検出により，カウンタの値を TP0CCR1 レジスタにキャプチャし，割り込みを発生します。 カウンタのオーバフローが検出されると，INTTP0OV 割り込みを発生します。 TOP00 端子はハイ・レベル・スタートです。
【使 用 ＳＦＲ】	なし
【call　関数】	timerp_port_set，timerp_free_running_set，timerp_free_running_start
【変　　数】	なし
【割 り 込 み】	timerp_TP0CC0_int timerp_TP0CC1_int timerp_TP0OV_int
【割り込み要因】	INTTP0CC0 INTTP0CC1 INTTP0OV
【ファイル名】	timerp_free_running¥timerp_free_running.c
【注 意 事 項】	なし

【関　数　名】 timerp_port_set

【引　き　数】 なし

【処 理 内 容】 ポート 3 を TOP00 出力端子，TIP01 入力端子に設定します。

【使 用 S F R】 PFC3L　：0x04（TOP00 出力端子，TIP01 入力端子に設定）
PFCE3L　：0x04（TOP00 出力端子，TIP01 入力端子に設定）
PMC3L　：0x0C（TIP01 入力端子，TOP00 出力端子に設定）

【call　関数】 なし

【変　数】 なし

【ファイル名】 timerp_free_running¥timerp_free_running.c

【注 意 事 項】 なし

【関　数　名】 timerp_free_running_set

【引　き　数】 なし

【処 理 内 容】 TMP0 制御レジスタの設定を行います。

【使 用 S F R】 TP0CTL0.TP0CE：0（TMP0 動作禁止）
TP0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TP0CTL1　：0x05（タイマ・モードをフリー・ランニング・タイマ・モードに設定）
TP0IOC0　：0x01（TOP00 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TP0IOC1　：0x08（TIP01 端子の有効エッジを立ち下がりエッジを検出に設定）
TP0OPT0　：0x20（TP0CCR1 をキャプチャ・レジスタに設定）
TP0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TP0CCMK0　：0（TMP0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TP0CCMK1　：0（TMP0 キャプチャ 1／コンペア 1 一致割り込みマスク・クリアに設定）
TP0OVMK　：0（TMP0 オーバフロー割り込みマスク・クリアに設定）

【call　関数】 なし

【変　数】 なし

【ファイル名】 timerp_free_running¥timerp_free_running.c

【注 意 事 項】 なし

【関　数　名】　timerp_ free_running _start

【引　き　数】　なし

【処 理 内 容】　フリー・ランニング・タイマ・モードの起動関数です。

【起 動 方 法】　timerp_free_running_set 関数のあとにコールしてください。

【使 用 ＳＦＲ】　TP0CTL0.TP0CE　：1（TMP0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_free_running¥timerp_free_running.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】　timerp_TP0CC0_int

【概　要】　ユーザ定義

【要　因】　INTTP0CC0　16 ビット・カウンタのカウント値と TP0CCR0 の一致

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_free_running¥timerp_free_running.c

【注 意 事 項】　なし

【関　数　名】　timerp_TP0CC1_int

【概　要】　ユーザ定義

【要　因】　INTTP0CC1　TIP01 端子入力の有効エッジ検出

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_free_running¥timerp_free_running.c

【注 意 事 項】　なし

【関　数　名】 timerp_TP0OV_int

【概　　要】 ユーザ定義

【要　　因】 INTTP0OV　　16 ビット・カウンタのオーバフロー発生

【call　　関数】 なし

【変　　数】 なし

【ファイル名】 timerp_free_running¥timerp_free_running.c

【注 意 事 項】 なし

図3－14　フリー・ランニング・タイマ・モード（1）

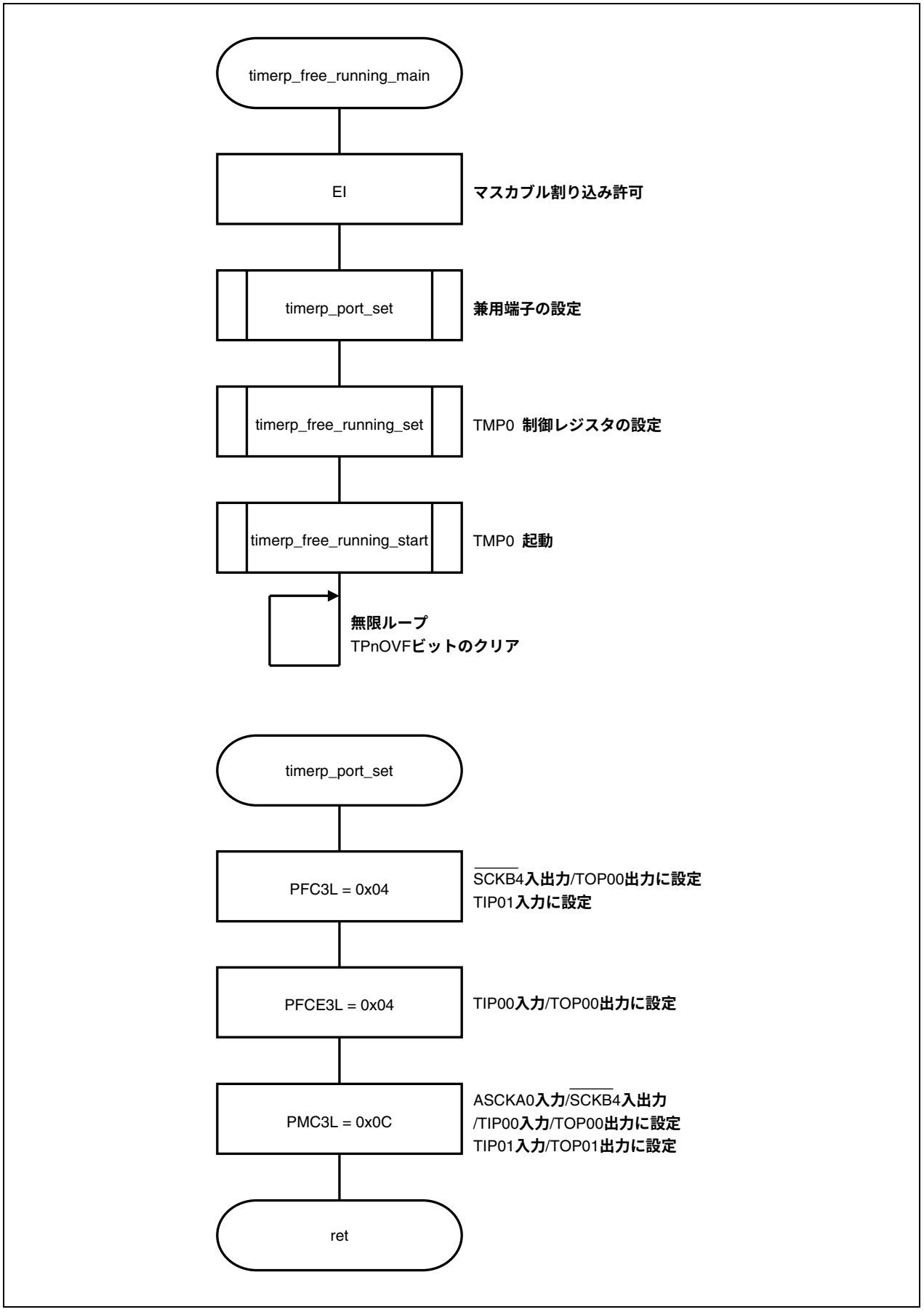


図3－15　フリー・ランニング・タイマ・モード（2）

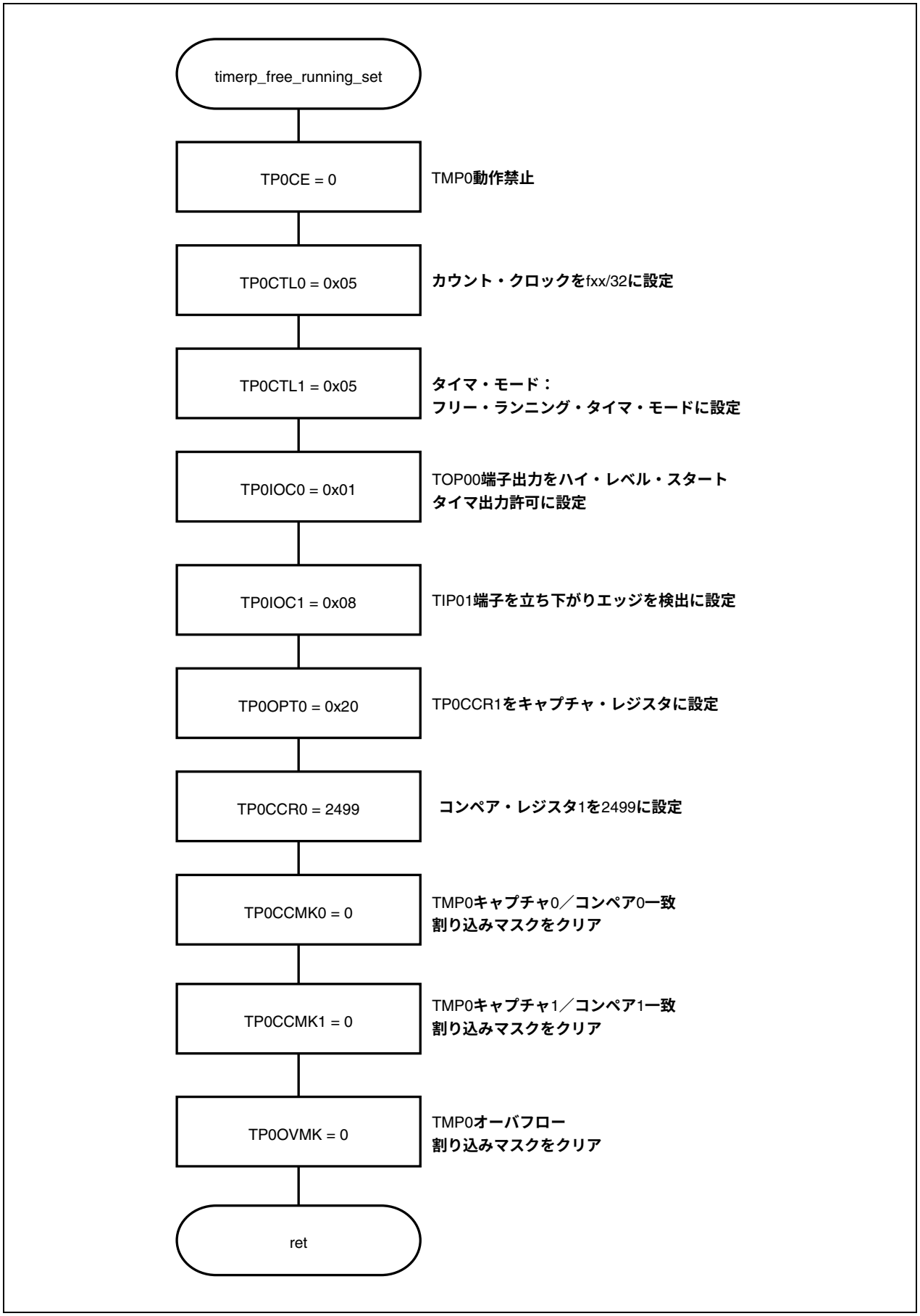
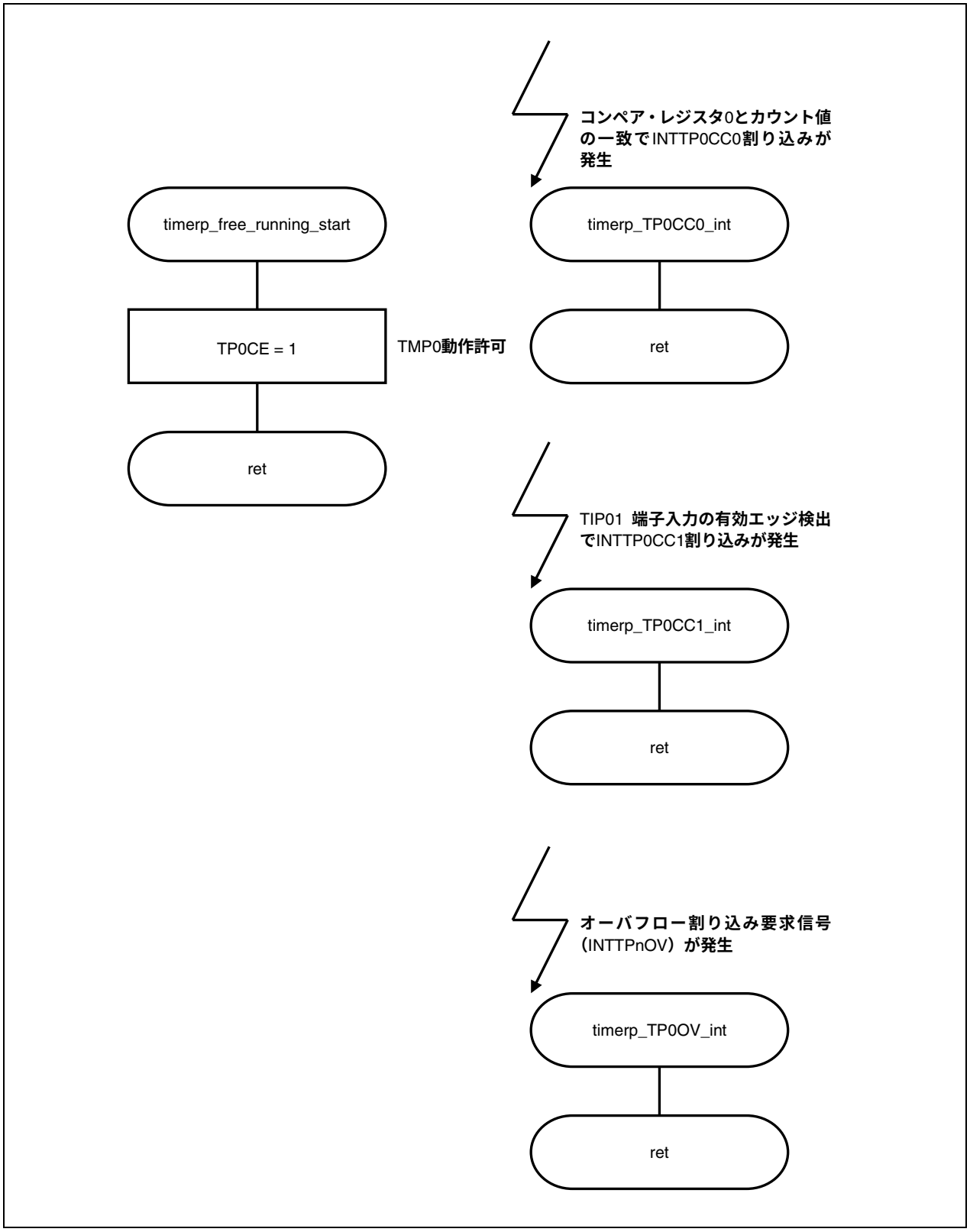


図3－16　フリー・ランニング・タイマ・モード（3）



3. 7　パルス幅測定モード

【機　　能】	TP0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 キャプチャ・トリガ入力（TIP00 端子）の有効エッジで 16 ビット・カウンタのカウント値を TP0CCR0 レジスタに格納し，16 ビット・カウンタをクリアします。 また，TIP00 端子入力の有効エッジで割り込みを発生させ，TP0CCR0 レジスタの値を読み込むことにより TIP00 端子の有効エッジ間隔を測定します。 TMP0-TMP5 で実現可能です。
【関　数　名】	timerp_pulse_measure
【引　き　数】	なし
【処 理 内 容】	fxx/32 のカウント・クロックでカウント動作を行い，TIP00 端子入力の有効エッジで 16 ビット・カウンタのカウント値を TP0CCR0 レジスタに格納して割り込みを発生し，カウンタをクリアします。 カウンタのオーバフローが検出されると，INTTP0OV 割り込みを発生します。
【使 用 S F R】	なし
【call　　関数】	timerp_port_set，timerp_pulse_measure_set，timerp_ pulse_measure_start
【変　　数】	なし
【割 り 込 み】	timerp_TP0CC0_int timerp_TP0OV_int
【割り込み要因】	INTTP0CC0 INTTP0OV
【ファイル名】	timerp_pulse_measure¥timerp_pulse_measure.c
【注 意 事 項】	なし

【関　数　名】	timerp_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 3 を TIP00 入力端子に設定します。
【使 用 S F R】	PFCE3L　：0x04　（TIP00 入力端子の設定） PMC3L　：0x04　（TIP00 入力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerp_pulse_measure¥timerp_pulse_measure.c
【注 意 事 項】	なし

【関　数　名】　timerp_free_running _set

【引　き　数】　なし

【処　理　内　容】　TMP0 制御レジスタの設定を行います。

【使 用　S　F　R】　TP0CTL0.TP0CE：0（TMP0 動作禁止）
TP0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TP0CTL1　：0x06（タイマ・モードをパルス幅測定モードに設定）
TP0IOC1　：0x02（TIP00 端子を立ち下がりエッジを検出に設定）
TP0IOC2　：0x08（TIP00 端子を立ち下がりエッジを検出に設定）
TP0CCMK0　：0（TMP0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TP0OVMK　：0（TMP0 オーバフロー割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_pulse_measure¥timerp_pulse_measure.c

【注　意　事　項】　なし

パルス幅は次に示す式で計算できます。

パルス幅 =（TP0CCR0 レジスタの値＋1）×カウント・クロック周期

16ビット・カウンタのオーバフローが検出された場合のパルス幅は次に示す式で計算できます。

パルス幅 =（TP0CCR0 レジスタの値＋0x10001）×カウント・クロック周期

【関　数　名】　timerp_pulse_measure_start

【引　き　数】　なし

【処　理　内　容】　パルス幅測定モードの起動関数です。

【起　動　方　法】　timerp_pulse_measure_set 関数のあとにコールしてください。

【使 用　S　F　R】　TP0CTL0.TP0CE　：1（TMP0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerp_pulse_measure¥timerp_pulse_measure.c

【注　意　事　項】　なし

割り込み関数

【関　数　名】	timerp_TP0CC0_int
【概　　要】	ユーザ定義
【要　　因】	INTTP0CC0　　TIP00 端子入力の有効エッジ検出
【call　関数】	なし
【変　　数】	なし
【ファイル名】	timerp_pulse_measure¥timerp_pulse_measure.c
【注意事項】	なし

【関　数　名】	timerp_TP0OV_int
【概　　要】	ユーザ定義
【要　　因】	INTTP0OV　　16 ビット・カウンタのオーバフロー発生
【call　関数】	なし
【変　　数】	なし
【ファイル名】	timerp_pulse_measure¥timerp_pulse_measure.c
【注意事項】	なし

図3－17　パルス幅測定モード（1）

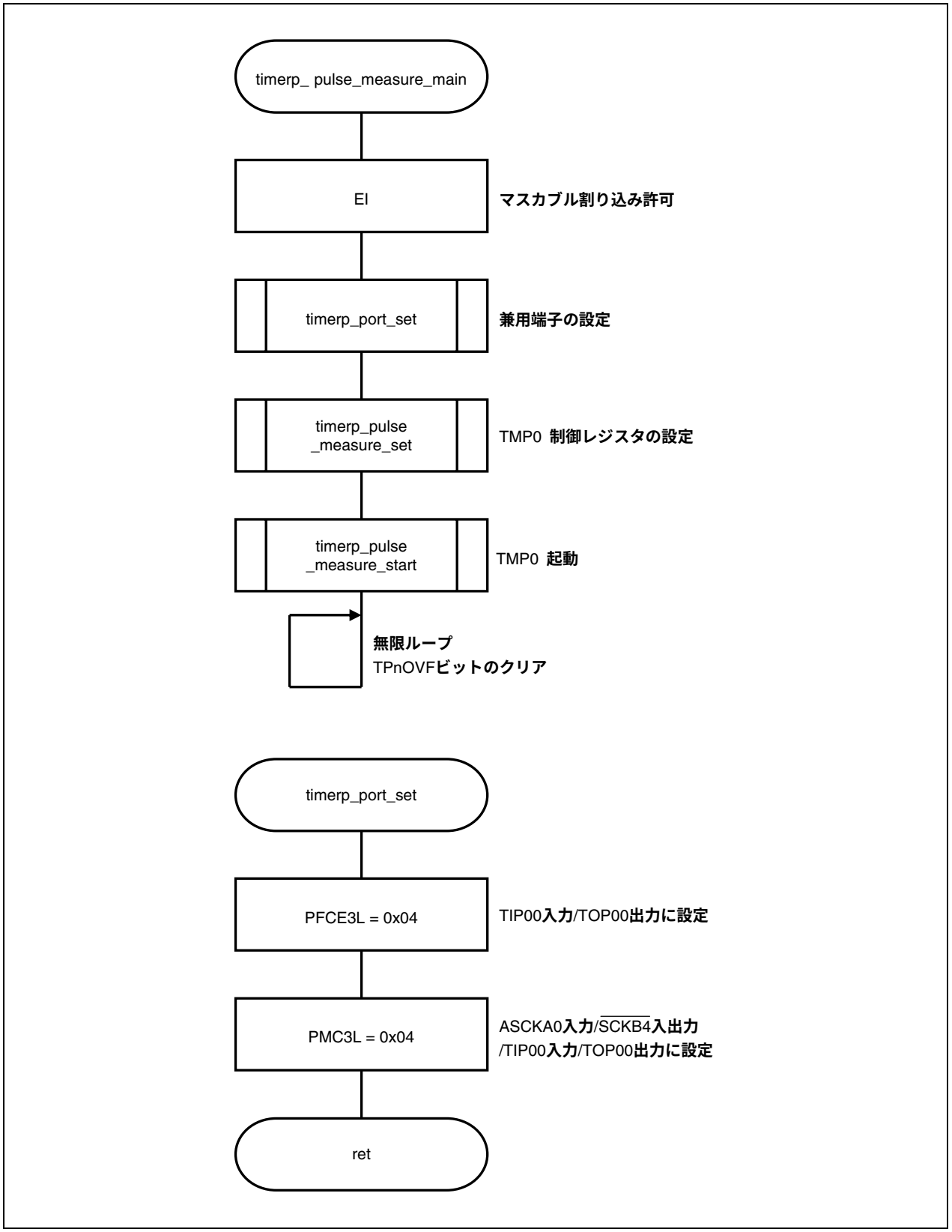


図3－18　パルス幅測定モード（2）

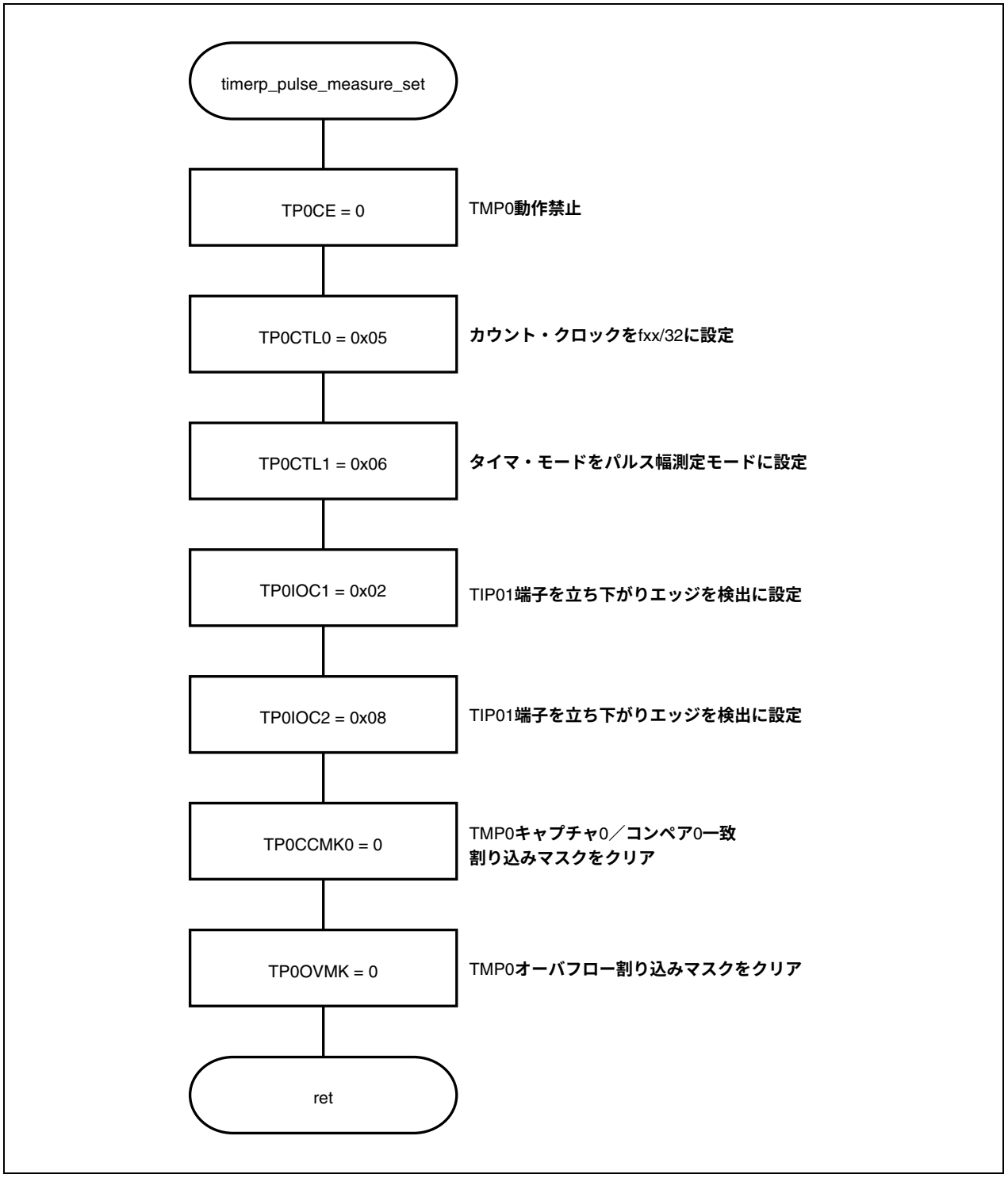
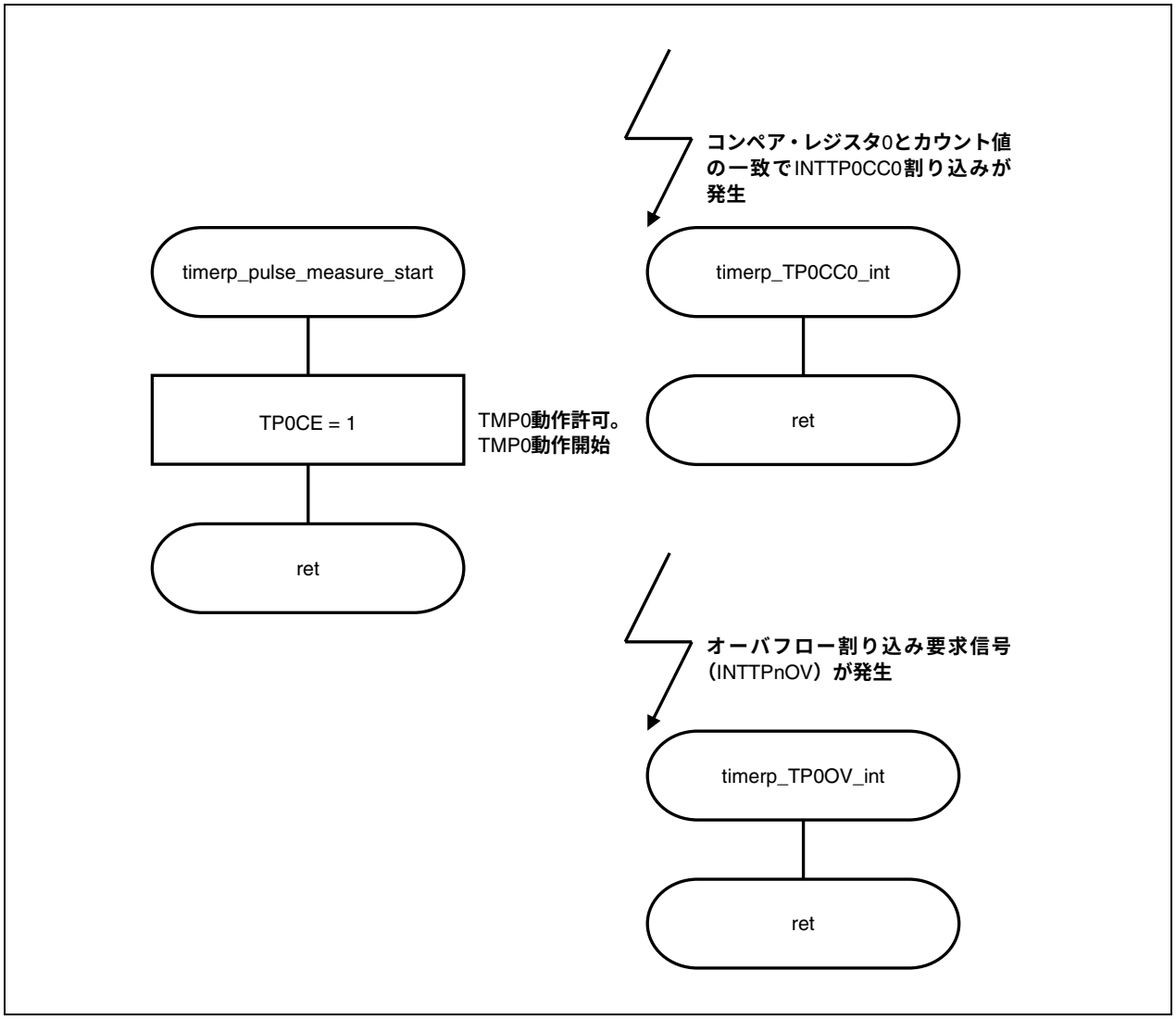


図3－19　パルス幅測定モード（3）



第4章　16ビット・タイマ／イベント・カウンタQ（TMQ）

4.1　インターバル・タイマ・モード

【機　　能】	TQ0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 TQ0CCR0 レジスタで設定したインターバル間隔で割り込み要求信号（INTTQ0CC0）を発生します。また，TQ0CCR0 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOQ00 端子出力を反転します。
【関　数　名】	timerq_interval_main
【引　き　数】	なし
【処 理 内 容】	fxx/32 のカウント・クロックでカウント動作を行い，カウンタの値が TQ0CCR0 レジスタの値と一致した次のカウンタのタイミングで，TOQ00 端子出力を反転させて割り込みを発生します。 TOQ00 端子はハイ・レベル・スタートです。
【使 用 S F R】	なし
【call　　関数】	timerq_port_set，timerq_interval_set，timerq_interval_start
【変　　数】	なし
【割 り 込 み】	timerq_TQ0CC0_int
【割り込み要因】	INTTQ0CC0
【ファイル名】	timerq_interval¥timerq_interval.c
【注 意 事 項】	なし

【関　数　名】	timerq_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 5 を TOQ00 出力端子に設定します。
【使 用 S F R】	PFC5　　：0x00（TOQ00 出力端子に設定） PFCE5　：0x08（TOQ00 出力端子に設定） PMC5　　：0x08（TOQ00 出力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_interval¥timerq_interval.c
【注 意 事 項】	なし

【関　数　名】　timerq_interval_set

【引　き　数】　なし

【処　理　内　容】　TMQ0 制御レジスタの設定を行います。

【使 用 S F R】　TQ0CTL0.TQ0CE：0（TMQ0 動作禁止）
TQ0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TQ0CTL1　：0x00（タイマ・モードをインターバル・タイマ・モードに設定）
TQ0IOC0　：0x01（TOQ00 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TQ0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TQ0CCMK0　：0（TMQ0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_interval¥timerq_interval.c

【注　意　事　項】　なし

インターバル間隔は次に示す式で計算できます。

インターバル間隔 =　（TQ0CCR0 レジスタ設定値＋1）×カウント・クロック周期

【関　数　名】　timerq_interval_start

【引　き　数】　なし

【処　理　内　容】　インターバル・タイマ・モードの起動関数です。

【起　動　方　法】　timerq_interval_set 関数のあとにコールしてください。

【使 用 S F R】　TQ0CTL0.TQ0CE　：1（TMQ0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_interval¥timerq_interval.c

【注　意　事　項】　なし

割り込み関数

【関　数　名】	timerq_TQ0CC0_int
【概　　要】	ユーザ定義
【要　　因】	INTTQ0CC0　　16 ビット・カウンタのカウント値と TQ0CCR0 の一致
【call　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_interval¥timerq_interval .c
【注意事項】	なし

図4－1　インターバル・タイマ・モード（1）

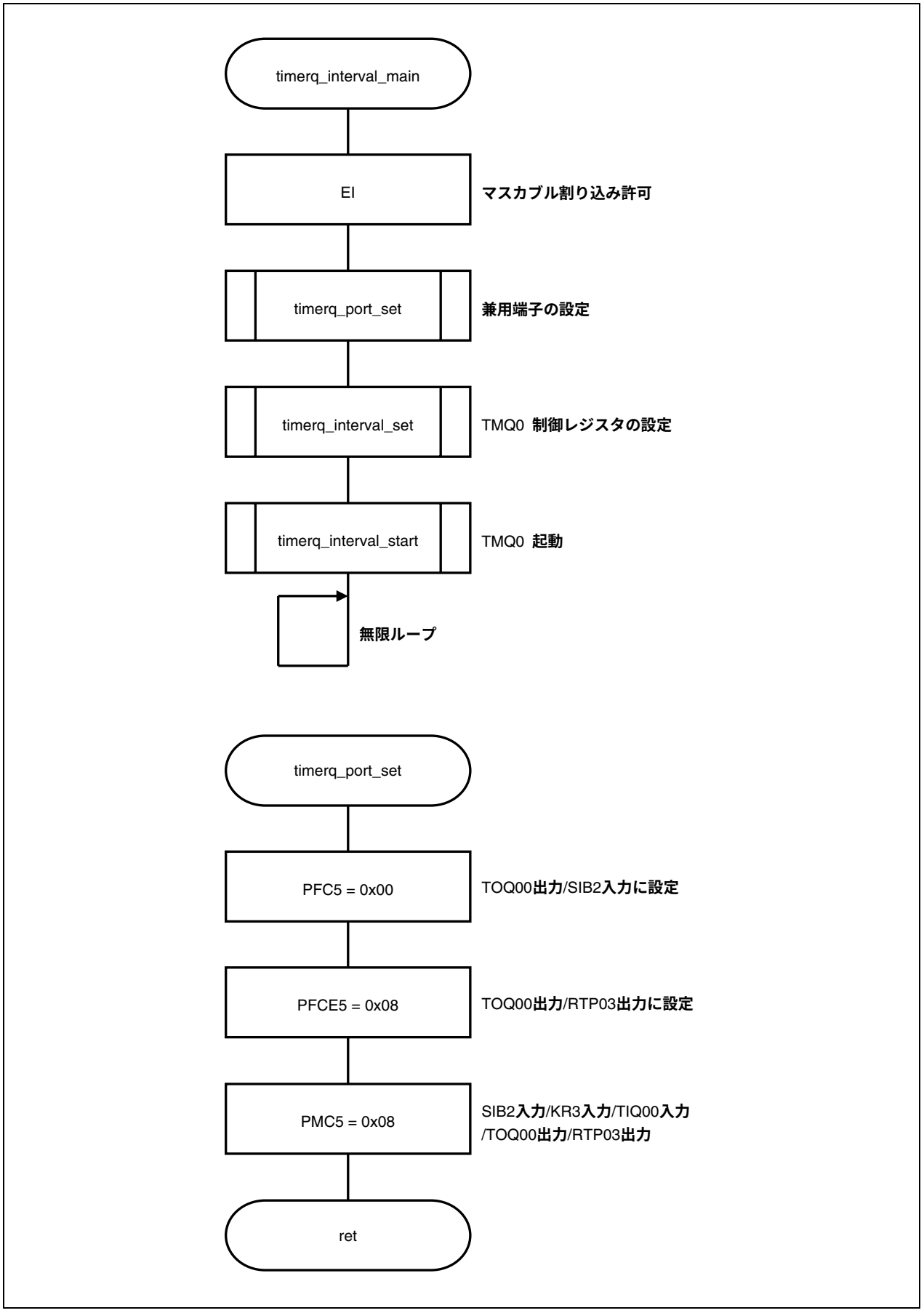
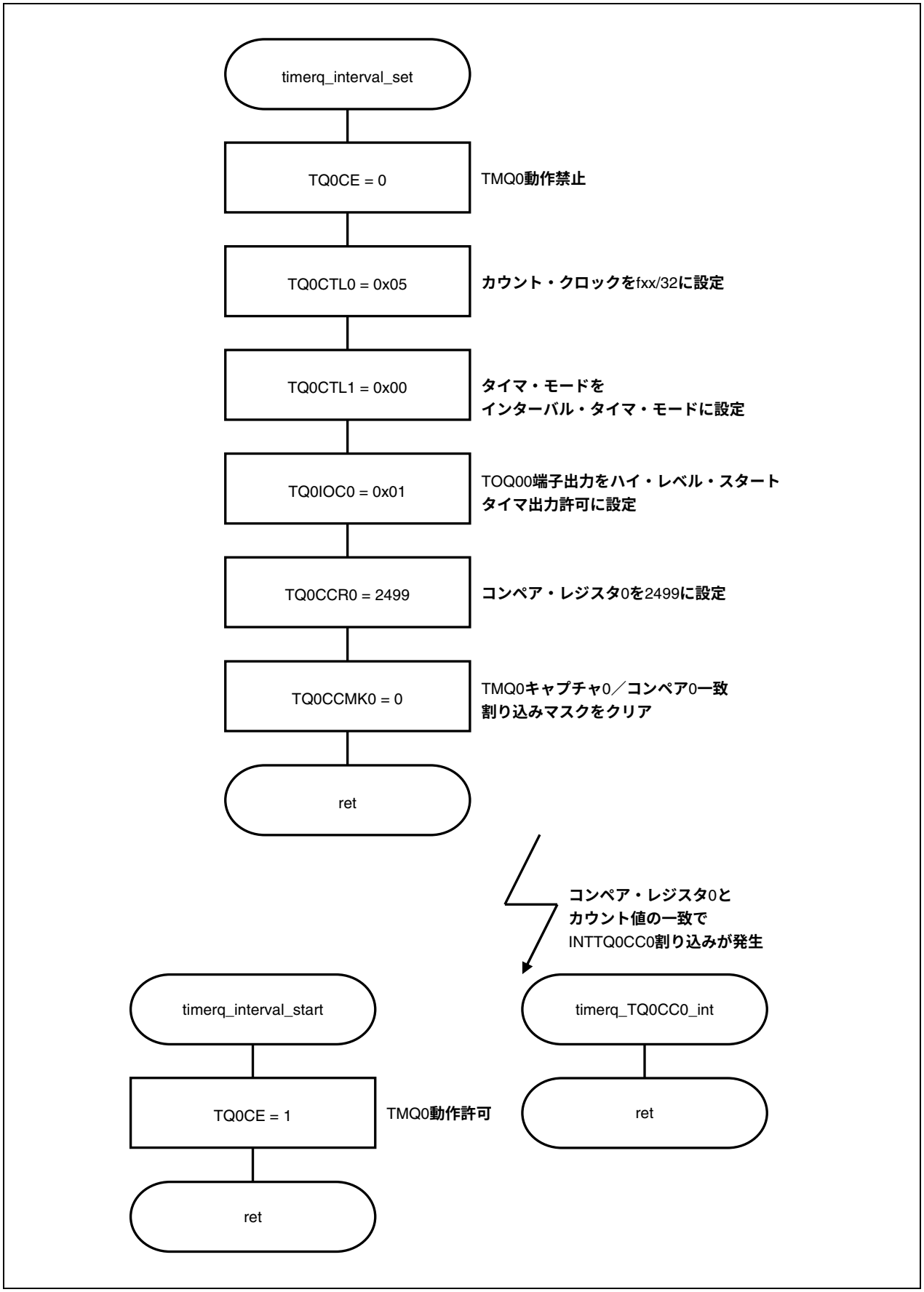


図4－2　インターバル・タイマ・モード（2）



4.2　外部イベント・カウント・モード

【機　　能】	外部イベント・カウント入力（TIQ00 端子）の有効エッジをカウントし，TQ0CCR0 レジスタで設定した値とカウントが一致するごとに，割り込み要求信号（INTTQ0CC0）を発生します（このとき，16 ビット・カウンタもクリアします）。
【関　数　名】	timerq_event_count_main
【引　き　数】	なし
【処 理 内 容】	外部イベント・カウント入力の有効エッジをカウントし，カウンタの値が TQ0CCR0 レジスタの値と一致した次のカウント（TQ0CCR0 レジスタに設定した値＋1 回）のタイミングで，割り込みを発生し，カウンタをクリアします。
【使 用 ＳＦＲ】	なし
【call　　関数】	timerq_port_set，timerq_event_count_set，timerq_event_count_start
【変　　数】	なし
【割 り 込 み】	timerq_TQ0CC0_int
【割り込み要因】	INTTQ0CC0
【ファイル名】	timerq_event_count¥timerq_event_count.c
【注 意 事 項】	なし

【関　数　名】	timerq_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 5 を TIQ00 入力端子に設定します。
【使 用 ＳＦＲ】	PFC5　　：0x08（TIQ00 入力端子に設定） PMC5　　：0x08（TIQ00 入力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_interval¥timerq_event_count.c
【注 意 事 項】	なし

【関　数　名】　timerq_event_count_set

【引　き　数】　なし

【処 理 内 容】　TMQ0 制御レジスタの設定を行います。

【使 用 S F R】　TQ0CTL0.TQ0CE：0（TMQ0 動作禁止）
TQ0CTL0　：0x00（カウント・クロックを fxx に設定）
TQ0CTL1　：0x01（タイマ・モードを外部イベント・カウント・モードに設定）
TQ0IOC2　：0x08（TIQ00 端子の有効エッジを立ち下がりエッジを検出に設定）
TQ0CCR0　：40（16 ビット・カウンタのコンペア・レジスタ 0 を 40 に設定）
TQ0CCMK0　：0（TMQ0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_event_count¥timerq_event_count.c

【注 意 事 項】　なし

【関　数　名】　timerq_event_count_start

【引　き　数】　なし

【処 理 内 容】　外部イベント・カウント・モードの起動関数です。

【起 動 方 法】　timerq_event_count_set 関数のあとにコールしてください。

【使 用 S F R】　TQ0CTL0.TQ0CE　：1（TMQ0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_event_count¥ timerq_event_count.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】　timerq_TQ0CC0_int

【概　要】　ユーザ定義

【要　因】　INTTQ0CC0　16 ビット・カウンタのカウント値と TQ0CCR0 の一致

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_event_count¥ timerq_event_count.c

【注 意 事 項】　なし

図4－3　外部イベント・カウント・モード（1）

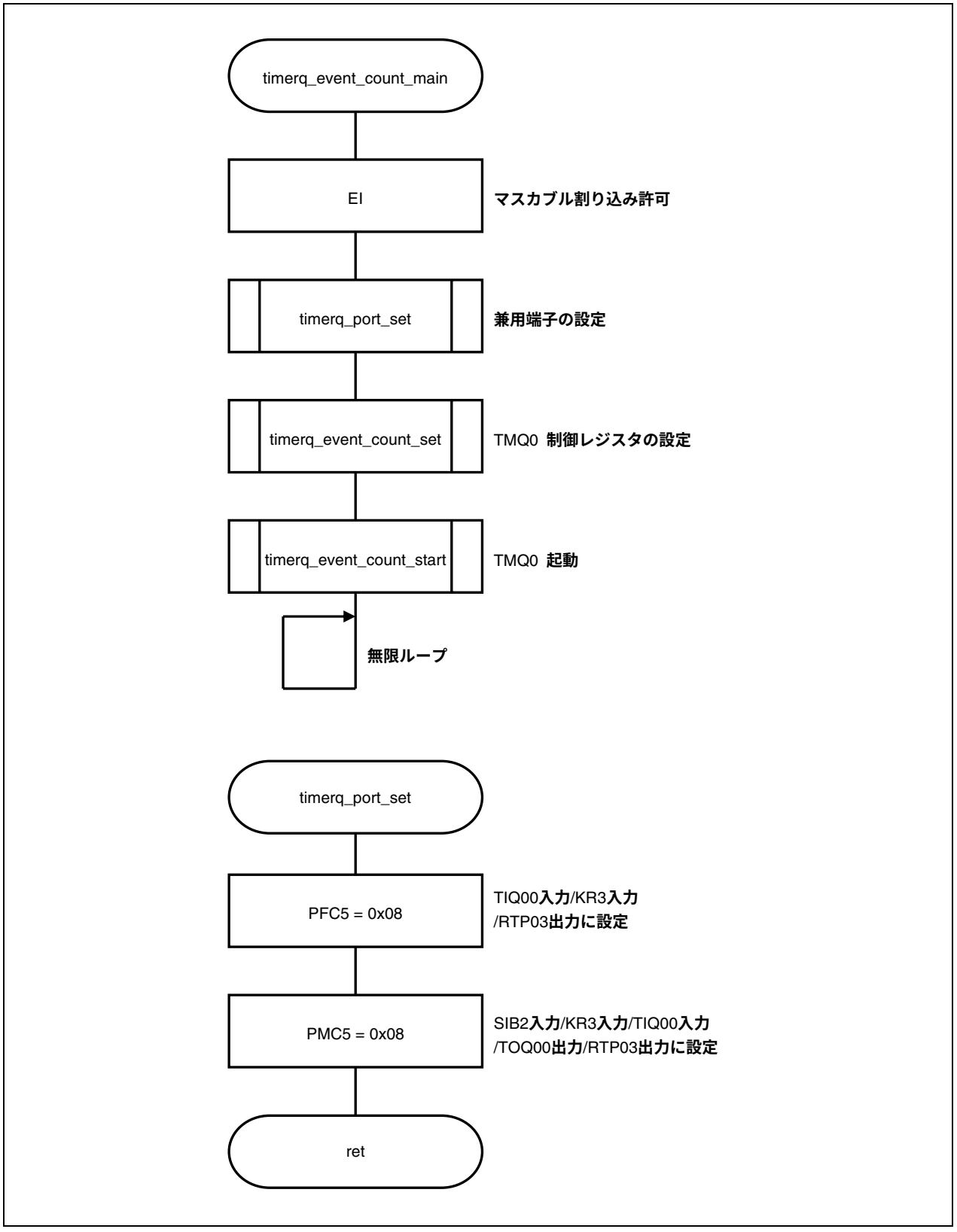
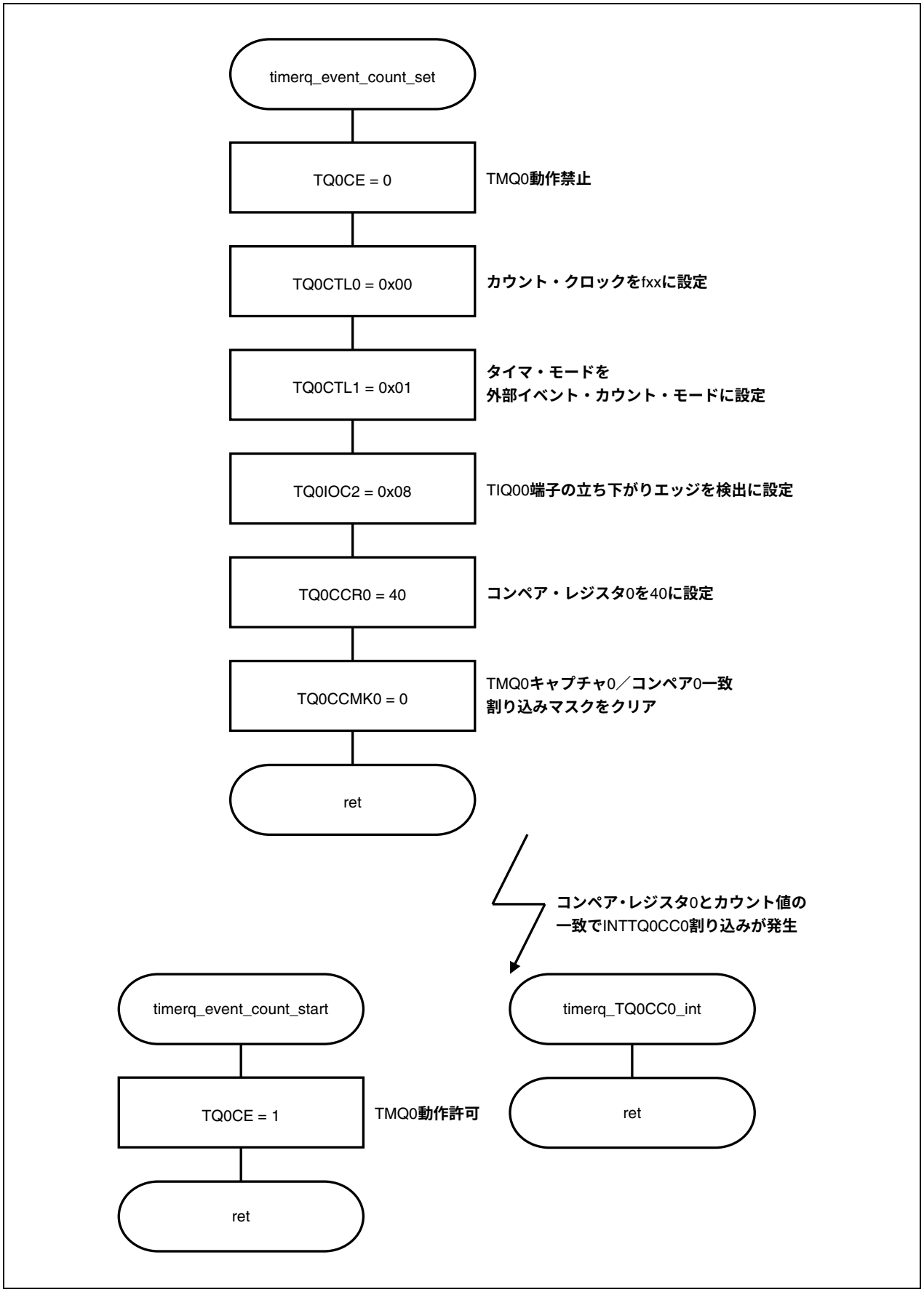


図4－4　外部イベント・カウント・モード（2）



4.3　外部トリガ・パルス出力モード

【機　　能】	外部トリガ入力（TIQ00 端子）の有効エッジを検出すると 16 ビット・カウンタの動作を開始します。 TQ0CCR0 レジスタとのコンペア一致で 16 ビット・カウンタをクリアします。 TQ0CCR0 レジスタ，TQ0CCR1 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOQ01 端子から PWM 波形を出力します。 再度トリガが発生した場合には，カウンタを 0000H にクリアし再スタートします
【関　数　名】	timerq_trigger_pulse_main
【引　き　数】	なし
【処 理 内 容】	外部トリガ入力の有効エッジ検出で，fxx/32 のカウント・クロックのカウント動作を開始し，カウンタの値が TQ0CCR0 レジスタの値と一致した次のカウントのタイミングで TOQ01 端子出力を反転させて割り込みを発生し，カウンタをクリアします。 また，カウンタの値が TQ0CCR1 レジスタの値と一致した次のカウントのタイミングで TOQ01 端子出力を反転させて割り込みを発生します。 TOQ01 端子はハイ・レベル・スタートです。
【使 用 S F R】	なし
【call　　関数】	timerq_port_set，timerq_trigger_pulse_set，timerq_trigger_pulse_start
【変　　数】	なし
【割 り 込 み】	timerq_TQ0CC0_int timerq_TQ0CC1_int
【割り込み要因】	INTTQ0CC0 INTTQ0CC1
【ファイル名】	timerq_trigger_pulse¥timerq_trigger_pulse.c
【注 意 事 項】	なし

【関　数　名】	timerq_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 5 を TOQ01 出力端子，TIQ00 入力端子に設定します。
【使 用 S F R】	PFC5　　：0x08（TOQ01 出力端子，TIQ00 入力端子に設定） PFCE5　：0x01（TOQ01 出力端子，TIQ00 入力端子に設定） PMC5　　：0x09（TIQ00 入力端子，TOQ01 出力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_trigger_pulse¥timerq_ trigger_pulse.c
【注 意 事 項】	なし

【関　数　名】　timerq_trigger_pulse_set

【引　き　数】　なし

【処 理 内 容】　TMQ0 制御レジスタの設定を行います。

【使 用 S F R】　TQ0CTL0.TQ0CE：0（TMQ0 動作禁止）
TQ0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TQ0CTL1　：0x02（タイマ・モードを外部トリガ・パルス出力モードに設定）
TQ0IOC0　：0x04（TOQ01 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TQ0IOC2　：0x08（TIQ00 端子の有効エッジを立ち下がりエッジを検出に設定）
TQ0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TQ0CCR1　：1249（16 ビット・カウンタのコンペア・レジスタ 1 を 1249 に設定）
TQ0CCMK0　：0（TMQ0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TQ0CCMK1　：0（TMQ0 キャプチャ 1／コンペア 1 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_trigger_pulse¥timerq_trigger_pulse.c

【注 意 事 項】　なし

PWM波形のアクティブ・レベル幅，周期，およびデューティは次に示す式で計算できます。

アクティブ・レベル幅 =（TQ0CCR1 レジスタの設定値）×カウント・クロック周期

周期 =（TQ0CCR0 レジスタの設定値＋1）×カウント・クロック周期

デューティ =（TQ0CCR1 レジスタの設定値）／（TQ0CCR0 レジスタの設定値＋1）

【関　数　名】　timerq_trigger_pulse_start

【引　き　数】　なし

【処 理 内 容】　外部トリガ・パルス出力モードの起動関数です。

【起 動 方 法】　timerq_trigger_pulse_set 関数のあとにコールしてください。

【使 用 S F R】　TQ0CTL0.TQ0CE　：1（TMQ0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_trigger_pulse¥timerq_trigger_pulse.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】 timerq_TQ0CC0_int

【概　　要】 ユーザ定義

【要　　因】 INTTQ0CC0　　16 ビット・カウンタのカウント値と TQ0CCR0 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerq_trigger_pulse¥timerq_trigger_pulse.c

【注 意 事 項】 なし

【関　数　名】 timerq_TQ0CC1_int

【概　　要】 ユーザ定義

【要　　因】 INTTQ0CC1　　16 ビット・カウンタのカウント値と TQ0CCR1 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerq_trigger_pulse¥timerq_trigger_pulse.c

【注 意 事 項】 なし

図4－5　外部トリガ・パルス出力モード（1）

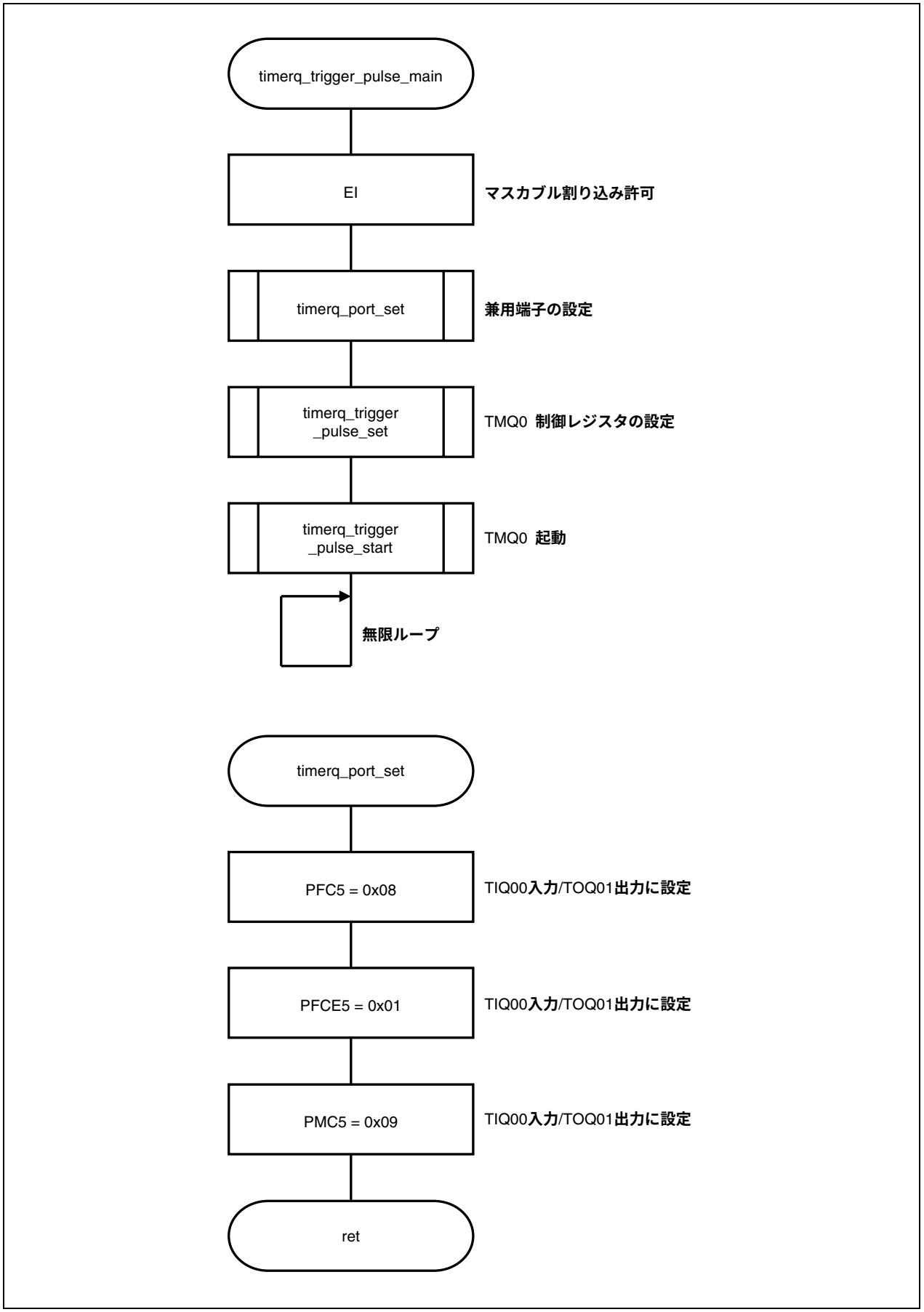


図4－6　外部トリガ・パルス出力モード（2）

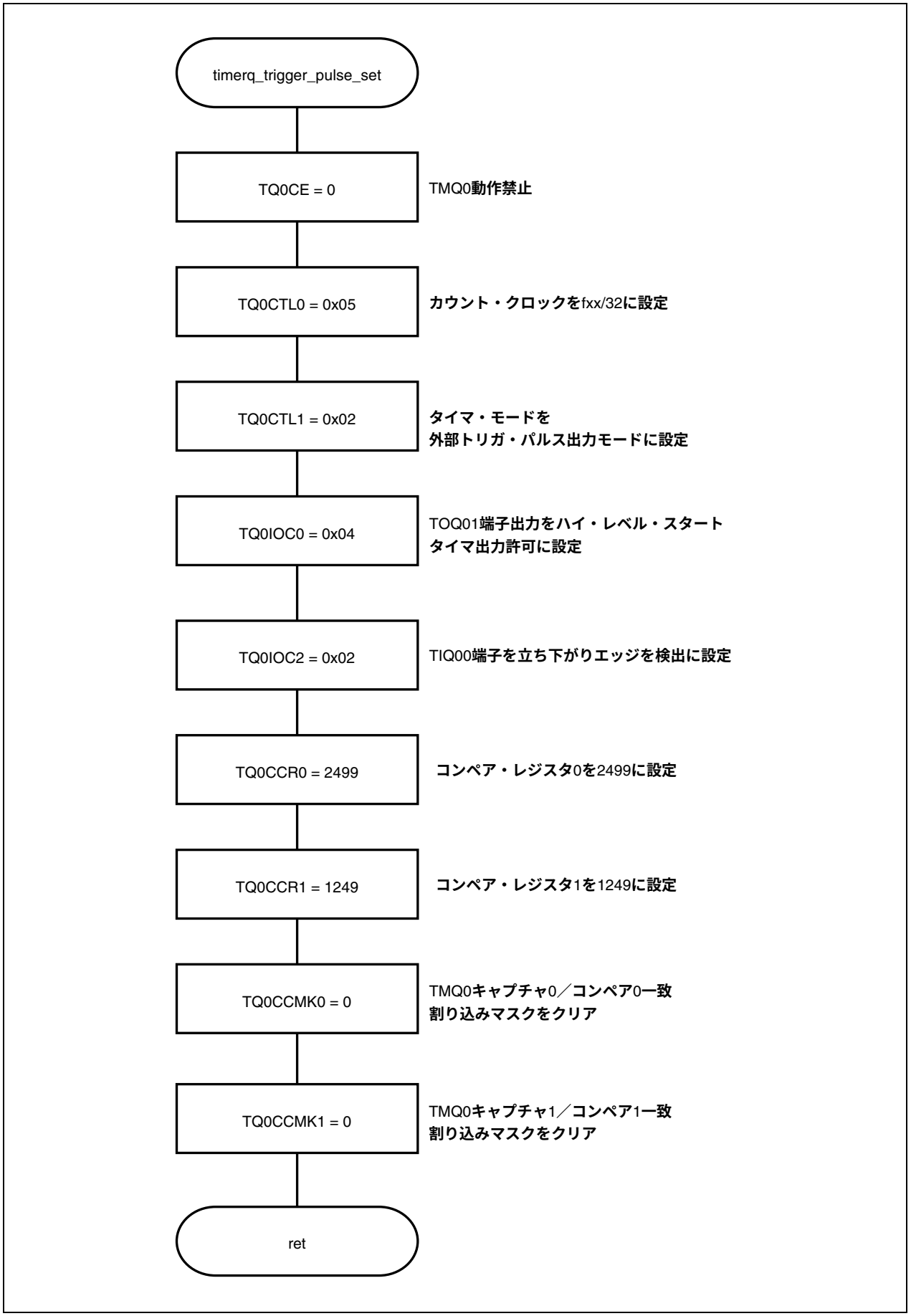
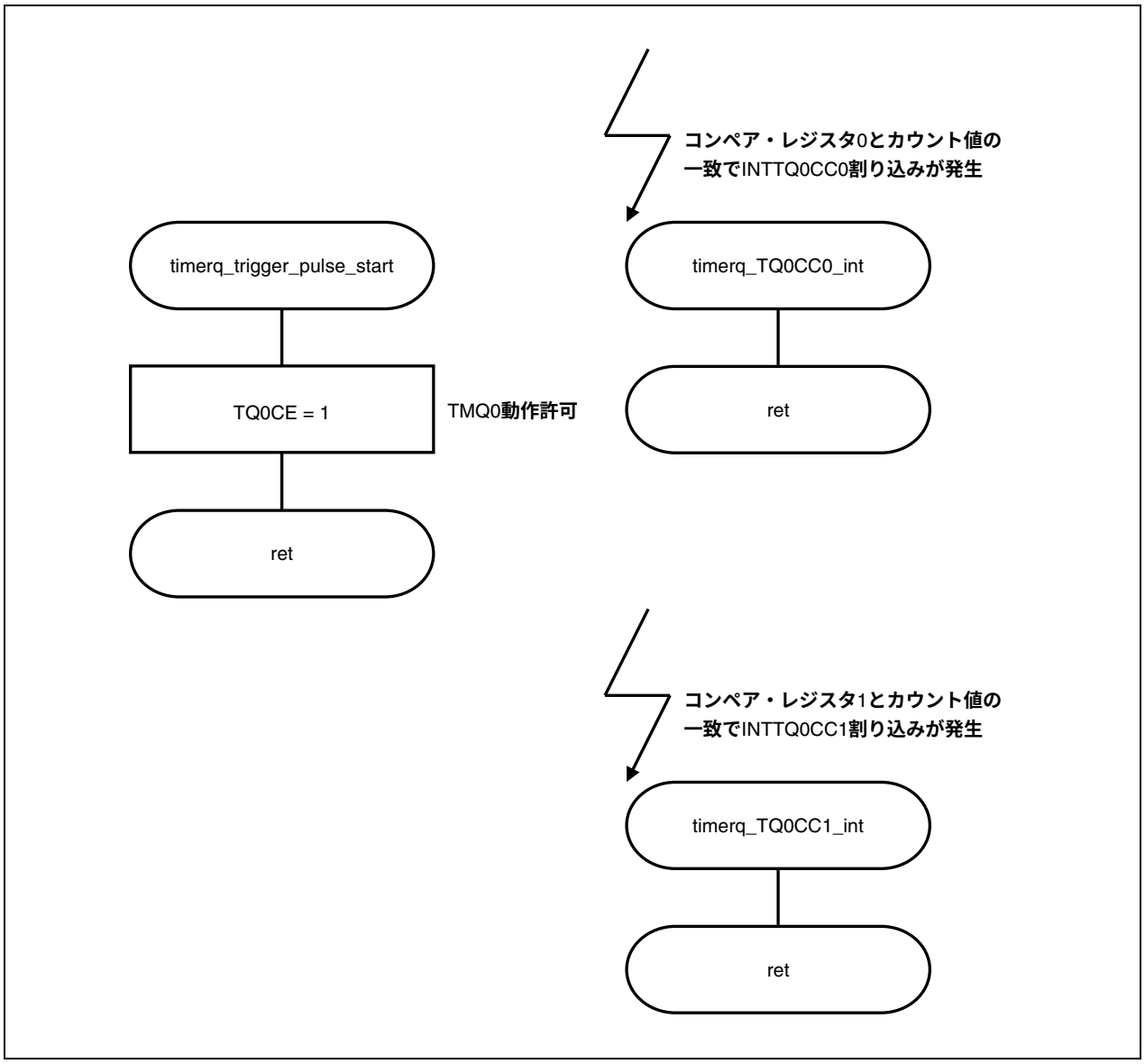


図4－7　外部トリガ・パルス出力モード（3）



4.4　ワンショット・パルス出力モード

【機　　能】	外部トリガ入力（TIQ00 端子）の有効エッジを検出すると 16 ビット・カウンタの動作を開始します。 TQ0CCR1 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOQ01 端子の出力をアクティブ・レベルにします。TQ0CCR0 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOQ01 端子の出力を反転させ，ワンショット・パルスを出力します。ワンショット・パルスを出力したあと，カウントを停止し，トリガ待ち状態となります。 ワンショット・パルス出力中に再度トリガが発生しても無視します。
【関　数　名】	timerq_1shot_pulse_main
【引　き　数】	なし
【処 理 内 容】	外部トリガ入力の有効エッジ検出で，fxx/32 のカウント・クロックのカウント動作を開始し，カウンタの値が TQ0CCR0 レジスタの値と一致した次のカウントのタイミングで割り込みを発生し，TOQ01 端子出力を反転させてカウンタをクリアし，カウント動作を停止します。また，カウンタの値が TQ0CCR1 レジスタの値と一致した次のカウントのタイミングで TOQ01 端子出力を反転させて割り込みを発生します。 TOQ01 端子はハイ・レベル・スタートです。
【使 用 S F R】	なし
【call　　関数】	timerq_port_set，timerq_1shot_pulse_set，timerq_1shot_pulse_start
【変　　数】	なし
【割 り 込 み】	timerq_TQ0CC0_int timerq_TQ0CC1_int
【割り込み要因】	INTTQ0CC0 INTTQ0CC1
【ファイル名】	timerq_1shot_pulse¥timerq_1shot_pulse.c,
【注 意 事 項】	なし

【関　数　名】	timerq_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 5 を TOQ01 出力端子，TIQ00 入力端子に設定します。
【使 用 S F R】	PFC5　　：0x08（TOQ01 出力端子，TIQ00 入力端子に設定） PFCE5　：0x01（TOQ01 出力端子，TIQ00 入力端子に設定） PMC5　　：0x09（TIQ00 入力端子，TOQ01 出力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_1shot_pulse¥timerq_1shot_pulse.c
【注 意 事 項】	なし

【関　数　名】　timerq_1shot_pulse_set

【引　き　数】　なし

【処 理 内 容】　TMQ0 制御レジスタの設定を行います。

【使 用 S F R】　TQ0CTL0.TQ0CE：0（TMQ0 動作禁止）
TQ0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TQ0CTL1　：0x03（タイマ・モードをワンショット・パルスモードに設定）
TQ0IOC0　：0x04（TOQ01 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TQ0IOC2　：0x08（TIQ00 端子の有効エッジを立ち下がりエッジを検出に設定）
TQ0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TQ0CCR1　：1249（16 ビット・カウンタのコンペア・レジスタ 1 を 1249 に設定）
TQ0CCMK0　：0（TMQ0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TQ0CCMK1　：0（TMQ0 キャプチャ 1／コンペア 1 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_1shot_pulse¥timerq_1shot_pulse.c

【注 意 事 項】　なし

ワンショット・パルスの出力ディレイ期間，およびアクティブ・レベル幅は次に示す式で計算できます。

出力ディレイ期間 =（TQ0CCR1 レジスタの設定値）×カウント・クロック周期

アクティブ・レベル幅 =（TQ0CCR0 レジスタの設定値−TQ0CCR1 レジスタの設定値+1）×カウント・クロック周期

【関　数　名】　timerq_1shot_pulse_start

【引　き　数】　なし

【処 理 内 容】　ワンショット・パルス出力モードの起動関数です。

【起 動 方 法】　timerq_1shot_pulse_set 関数のあとにコールしてください。

【使 用 S F R】　TQ0CTL0.TQ0CE　：1（TMQ0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_1shot_pulse¥timerq_1shot_pulse.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】 timerq_TQ0CC0_int

【概　　要】 ユーザ定義

【要　　因】 INTTQ0CC0　　16 ビット・カウンタのカウント値と TQ0CCR0 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerq_1shot_pulse¥timerq_1shot_pulse.c

【注 意 事 項】 なし

【関　数　名】 timerq_TQ0CC1_int

【概　　要】 ユーザ定義

【要　　因】 INTTQ0CC1　　16 ビット・カウンタのカウント値と TQ0CCR1 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerq_1shot_pulse¥timerq_1shot_pulse.c

【注 意 事 項】 なし

図4－8　ワンショット・パルス出力モード（1）

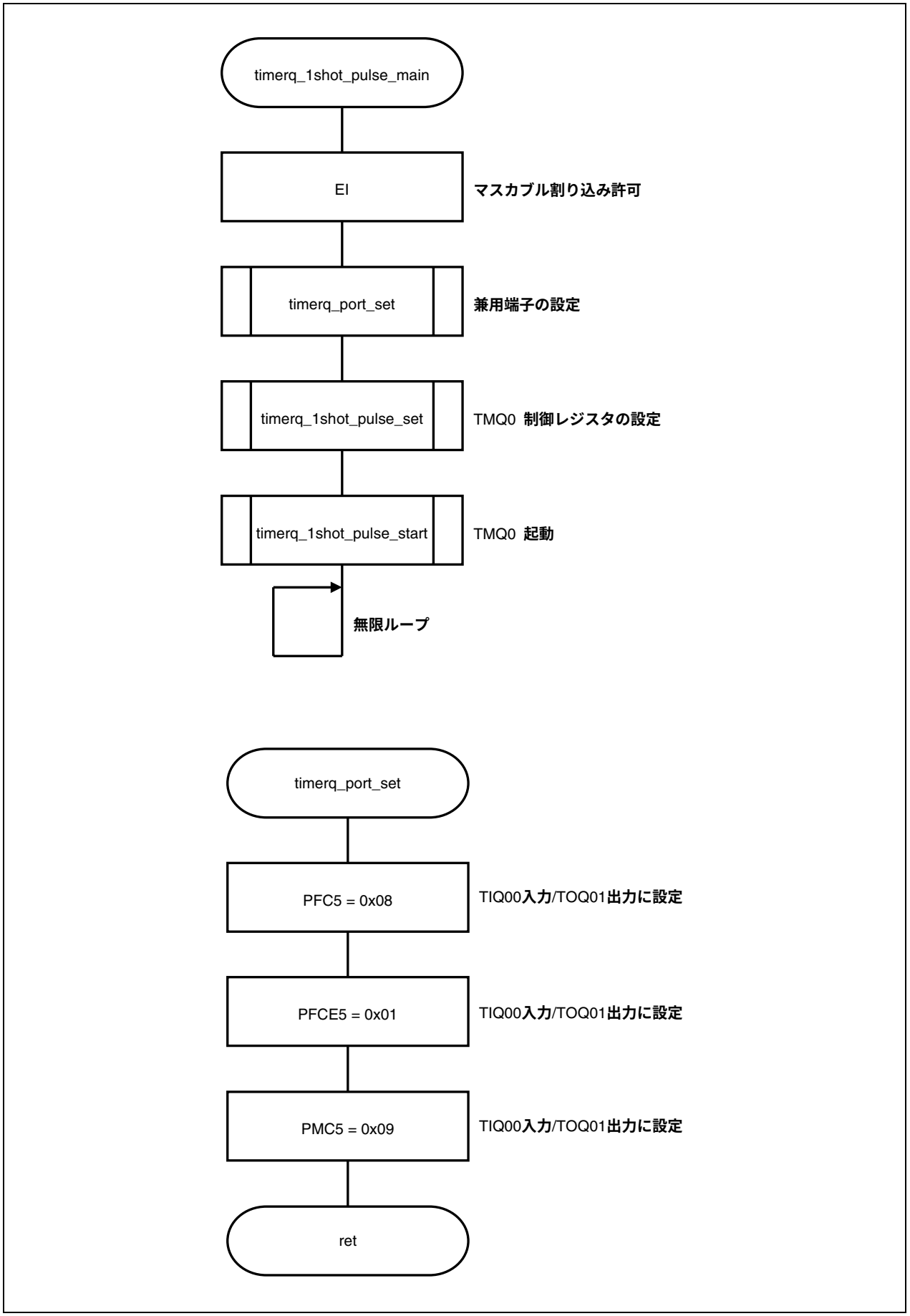


図4－9　ワンショット・パルス出力モード（2）

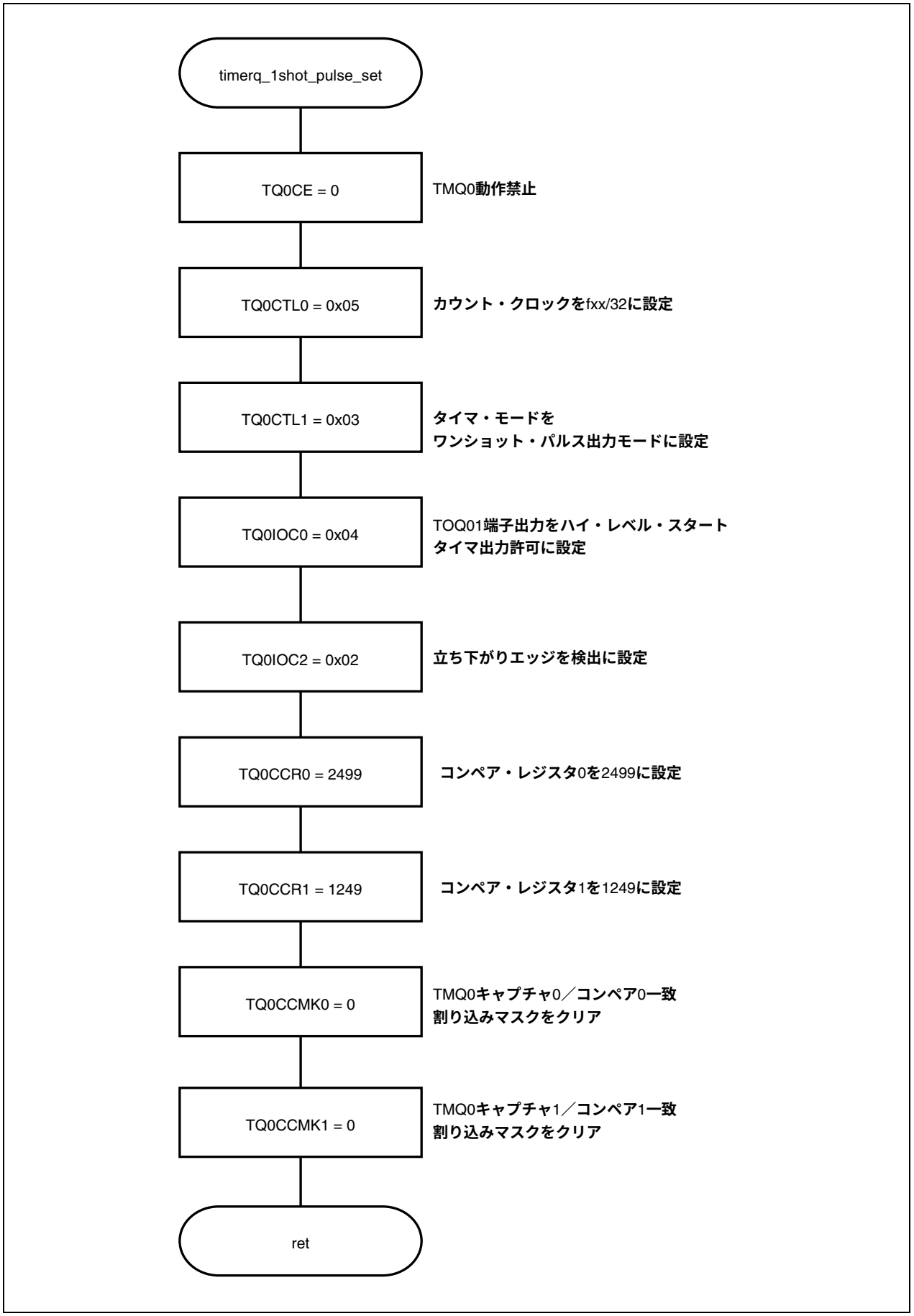
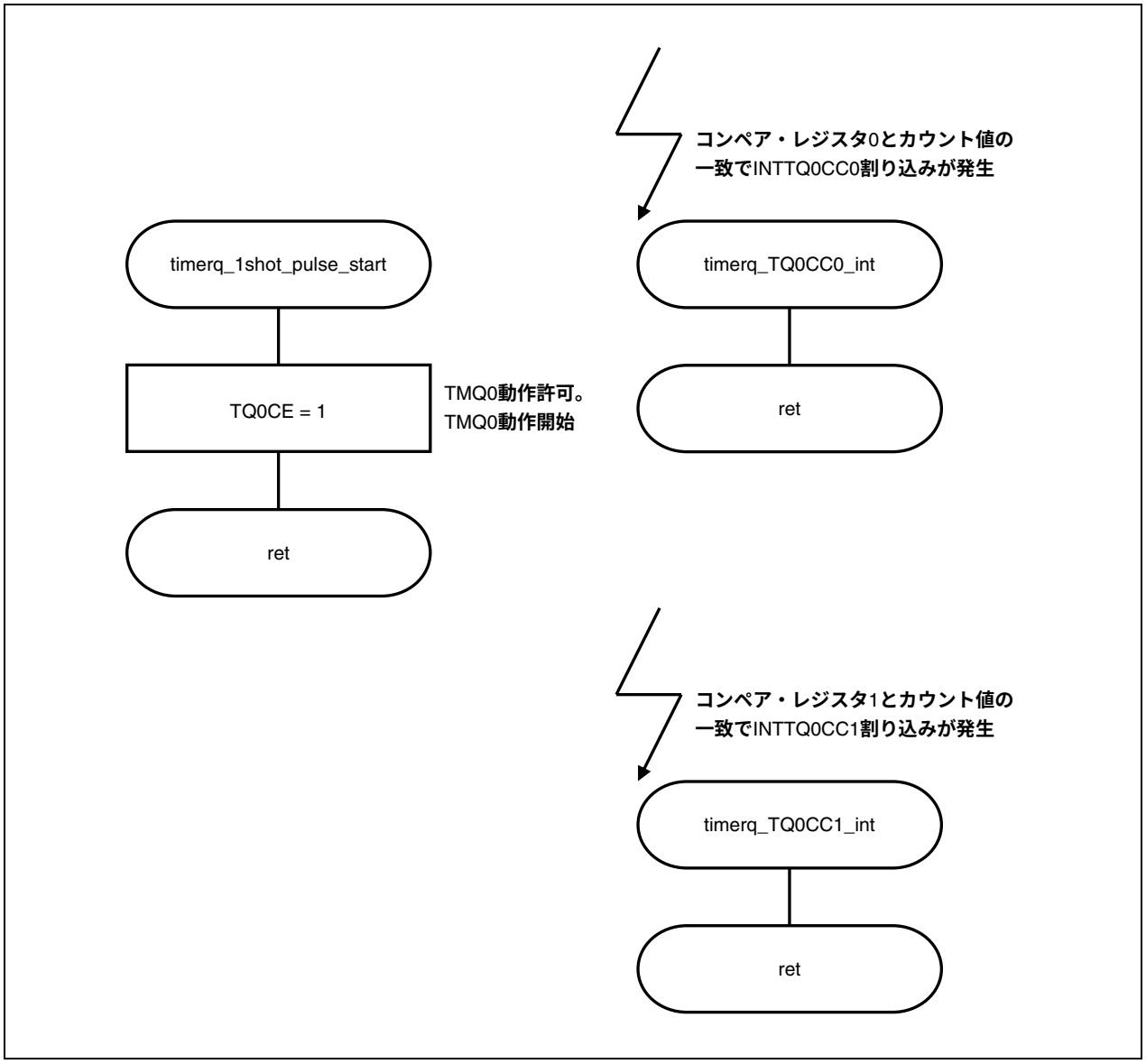


図4－10　ワンショット・パルス出力モード（3）



4. 5　PWM出力モード

【機　　能】	TQ0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 TQ0CCR0 レジスタとのコンペア一致で 16 ビット・カウンタをクリアし，TOQ00 端子を反転し，TQ0CCR0 レジスタの値＋1 を半周期とする 50 %（TQ0CCR1 レジスタの設定値）／（TQ0CCR0 レジスタの設定値＋1）デューティの PWM 波形を出力します。 TQ0CCR1 レジスタで設定した値と 16 ビット・カウンタのカウント値の一致で TOQ01 端子を反転します。 TOQ01 端子は 16 ビット・カウンタのクリア時に反転します。
【関　数　名】	timerq_pwm_output_main
【引　き　数】	なし
【処 理 内 容】	fxx/32 のカウント・クロックでカウント動作を行い，カウンタの値が TQ0CCR0 レジスタの値と一致した次のカウントのタイミングで TOQ00, TOQ01 端子出力を反転させて割り込みを発生し，カウンタをクリアします。 また，カウンタの値が TQ0CCR1 レジスタの値と一致した次のカウントのタイミングで TOQ01 端子出力を反転させて割り込みを発生します。 TOQ00, TOQ01 端子ともにハイ・レベル・スタートです。
【使 用 S F R】	なし
【call　　関数】	timerq_port_set，timerq_pwm_output_set，timerq_pwm_output_start
【変　　数】	なし
【割 り 込 み】	timerq_TQ0CC0_int timerq_TQ0CC1_int
【割り込み要因】	INTTQ0CC0 INTTQ0CC1
【ファイル名】	timerq_pwm_output¥timerq_pwm_output.c,
【注 意 事 項】	なし

【関　数　名】	timerq_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 5 を TOQ00 出力端子，TOQ01 出力端子に設定します。
【使 用 S F R】	PFC5　　：0x08（TOQ01 出力端子，TOQ00 出力端子に設定） PFCE5　：0x09（TOQ01 出力端子，TOQ00 出力端子に設定） PMC5　　：0x09（TOQ00 出力端子，TOQ01 出力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_pwm_output ¥timerq_pwm_output.c
【注 意 事 項】	なし

【関　数　名】　timerq_pwm_output_set

【引　き　数】　なし

【処 理 内 容】　TMQ0 制御レジスタの設定を行います。

【使 用 S F R】　TQ0CTL0.TQ0CE：0（TMQ0 動作禁止）
TQ0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TQ0CTL1　：0x04（タイマ・モードをワンショット・パルスモードに設定）
TQ0IOC0　：0x05（TOQ00 端子出力をハイ・レベル・スタート，タイマ出力許可，TOQ01 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TQ0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TQ0CCR1　：1249（16 ビット・カウンタのコンペア・レジスタ 1 を 1249 に設定）
TQ0CCMK0　：0（TMQ0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TQ0CCMK1　：0（TMQ0 キャプチャ 1／コンペア 1 一致割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_pwm_output ¥timerq_pwm_output.c

【注 意 事 項】　なし

TOQ01端子から出力されるPWM波形のアクティブ・レベル幅，周期，およびデューティは，次に示す式で計算できます。

アクティブ・レベル幅 = （TQ0CCR1 レジスタの設定値）×カウント・クロック周期

周期 = （TQ0CCR0 レジスタの設定値＋1）×カウント・クロック周期

デューティ = （TQ0CCR1 レジスタの設定値）／（TQ0CCR0 レジスタの設定値＋1）

【関　数　名】　timerq_pwm_output_start

【引　き　数】　なし

【処 理 内 容】　PWM 出力モードの起動関数です。

【起 動 方 法】　timerq_pwm_output_set 関数のあとにコールしてください。

【使 用 S F R】　TQ0CTL0.TQ0CE　：1（TMQ0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_pwm_output ¥timerq_pwm_output.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】 timerq_TQ0CC0_int

【概　　要】 ユーザ定義

【要　　因】 INTTQ0CC0　　16 ビット・カウンタのカウント値と TQ0CCR0 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerq_pwm_output¥timerq_pwm_output.c

【注 意 事 項】 なし

【関　数　名】 timerq_TQ0CC1_int

【概　　要】 ユーザ定義

【要　　因】 INTTQ0CC1　　16 ビット・カウンタのカウント値と TQ0CCR1 の一致

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerq_pwm_output¥timerq_pwm_output.c

【注 意 事 項】 なし

図4－11　PWM出力モード（1）

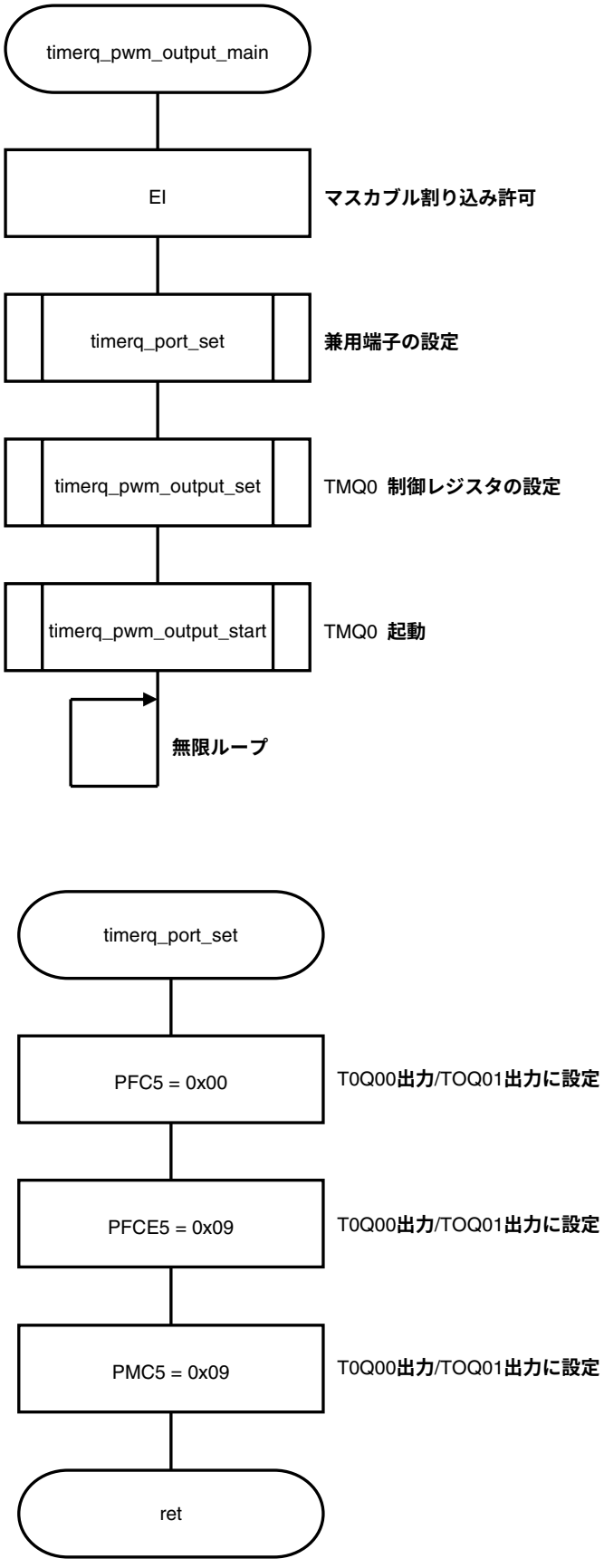


図4－12　PWM出力モード（2）

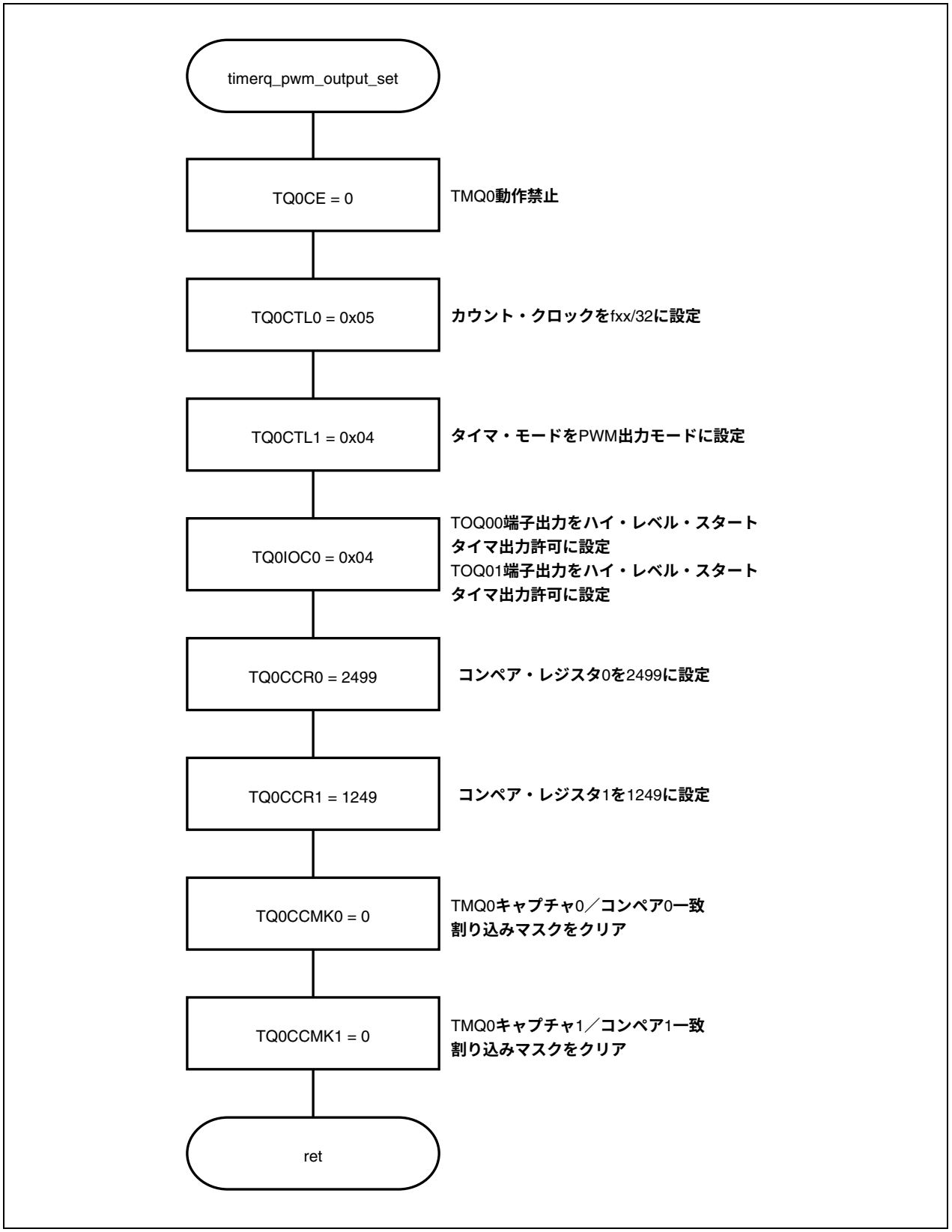
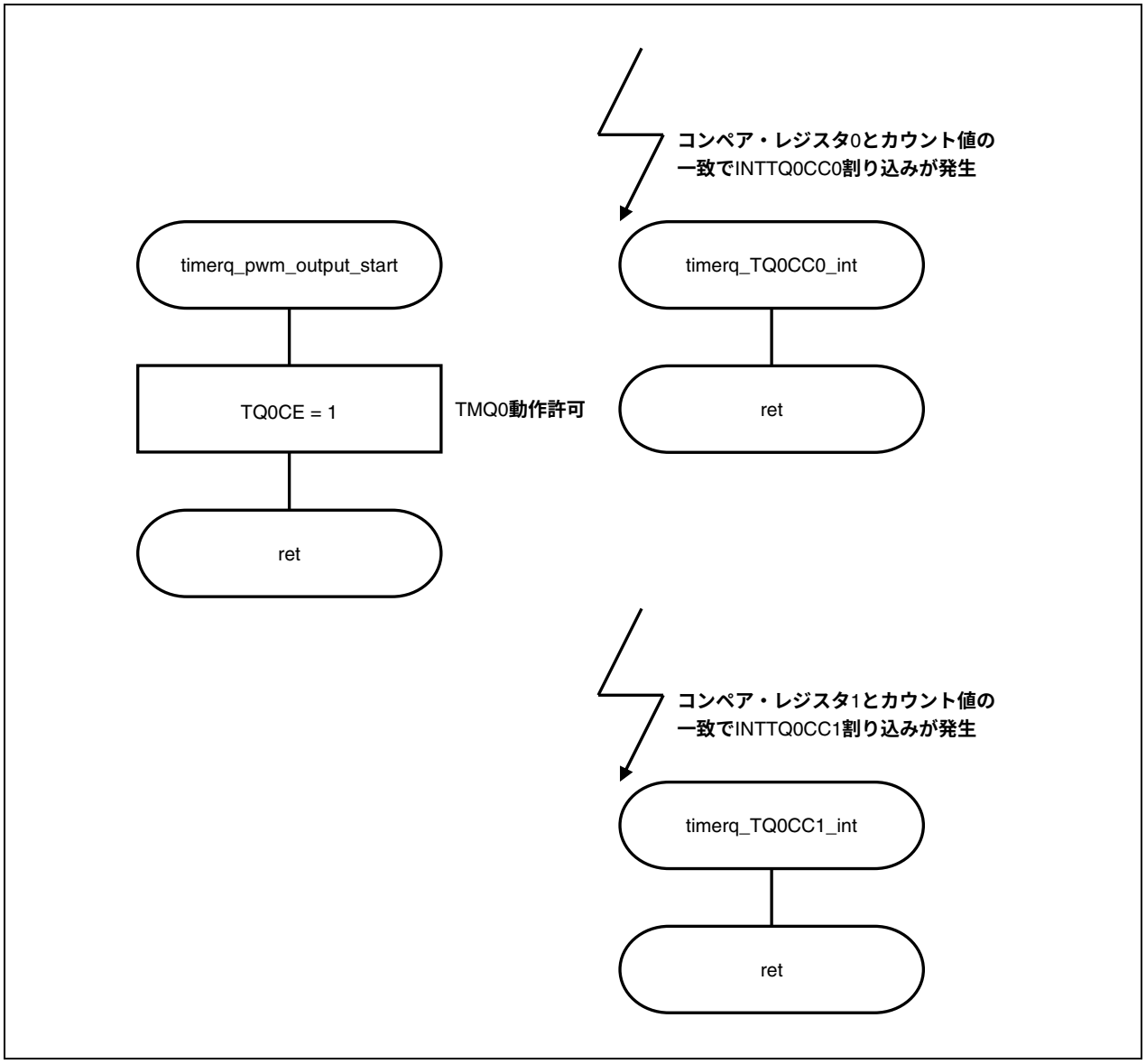


図4－13　PWM出力モード（3）



4.6　フリー・ランニング・タイマ・モード

【機　　能】	TQ0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 TQ0CCR0 レジスタと 16 ビット・カウンタのカウント値とのコンペア一致で TOQ00 端子の出力を反転します（コンペア機能）。TQ0CCR0 レジスタと 16 ビット・カウンタのカウント値が一致してもカウント値はクリアされません。FFFFH までカウントすると，次のクロックでオーバフロー割り込み要求信号（INTTQ0OV）を発生するとともに，0000H にクリアしカウント動作を継続します また，キャプチャ・トリガ入力（TIQ01 端子）の有効エッジ検出で 16 ビット・カウンタのカウント値を格納します（キャプチャ機能）。
【関　数　名】	timerq_free_running_main
【引　き　数】	なし
【処 理 内 容】	fxx/32 のカウント・クロックでカウント動作を行い，カウンタの値が TQ0CCR0 レジスタの値と一致した次のカウントのタイミングで TOQ00 端子出力を反転させて割り込みを発生し，カウンタをクリアします。 また，TIQ01 端子からの有効エッジ検出により，カウンタの値を TQ0CCR1 レジスタにキャプチャし，割り込みを発生します。 カウンタのオーバフローが検出されると，INTTQ0OV 割り込みを発生します。 TOQ00 端子はハイ・レベル・スタートです。
【使 用 ＳＦＲ】	なし
【call　関数】	timerq_port_set，timerq_free_running_set，timerq_free_running_start
【変　　数】	なし
【割 り 込 み】	timerq_TQ0CC0_int timerq_TQ0CC1_int timerq_TQ0OV_int
【割り込み要因】	INTTQ0CC0 INTTQ0CC1 INTTQ0OV
【ファイル名】	timerq_free_running¥timerq_free_running.c
【注 意 事 項】	なし

【関　数　名】　timerq_port_set

【引　き　数】　なし

【処　理　内　容】　ポート 5 を TOQ00 出力端子，TIQ01 入力端子に設定します。

【使 用 S F R】　PFC5　：0x01（TOQ00 出力端子，TIQ01 入力端子に設定）
PFCE5　：0x08（TOQ00 出力端子，TIQ01 入力端子に設定）
PMC5　：0x09（TIQ01 入力端子，TOQ00 出力端子に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_free_running¥timerq_free_running.c

【注　意　事　項】　なし

【関　数　名】　timerq_free_running_set

【引　き　数】　なし

【処　理　内　容】　TMQ0 制御レジスタの設定を行います。

【使 用 S F R】　TQ0CTL0.TQ0CE：0（TMQ0 動作禁止）
TQ0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TQ0CTL1　：0x05（タイマ・モードをフリー・ランニング・タイマ・モードに設定）
TQ0IOC0　：0x01（TOQ00 端子出力をハイ・レベル・スタート，タイマ出力許可に設定）
TQ0IOC1　：0x08（TIQ01 端子の有効エッジを立ち下がりエッジを検出に設定）
TQ0OPT0　：0x20（TQ0CCR1 をキャプチャ・レジスタに設定）
TQ0CCR0　：2499（16 ビット・カウンタのコンペア・レジスタ 0 を 2499 に設定）
TQ0CCMK0　：0（TMQ0 キャプチャ 0/コンペア 0 一致割り込みマスク・クリアに設定）
TQ0CCMK1　：0（TMQ0 キャプチャ 1/コンペア 1 一致割り込みマスク・クリアに設定）
TQ0OVMK　：0（TMQ0 オーバフロー割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_free_running¥timerq_free_running.c

【注　意　事　項】　なし

【関　数　名】　timerq_ free_running _start

【引　き　数】　なし

【処 理 内 容】　フリー・ランニング・タイマ・モードの起動関数です。

【起 動 方 法】　timerq_free_running_set 関数のあとにコールしてください。

【使 用 ＳＦＲ】　TQ0CTL0.TQ0CE　：1（TMQ0 動作許可）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　timerq_free_running¥timerq_free_running.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】　timerq_TQ0CC0_int

【概　　要】　ユーザ定義

【要　　因】　INTTQ0CC0　　16 ビット・カウンタのカウント値と TQ0CCR0 の一致

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　timerq_free_running¥timerq_free_running.c

【注 意 事 項】　なし

【関　数　名】　timerq_TQ0CC1_int

【概　　要】　ユーザ定義

【要　　因】　INTTQ0CC1　　TIQ01 端子入力の有効エッジ検出

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　timerq_free_running¥timerq_free_running.c

【注 意 事 項】　なし

【関　数　名】 timerq_TQ0OV_int

【概　　要】 ユーザ定義

【要　　因】 INTTQ0OV　　16 ビット・カウンタのオーバフロー発生

【call　関数】 なし

【変　　数】 なし

【ファイル名】 timerq_free_running¥timerq_free_running.c

【注 意 事 項】 なし

図4－14　フリー・ランニング・タイマ・モード（1）

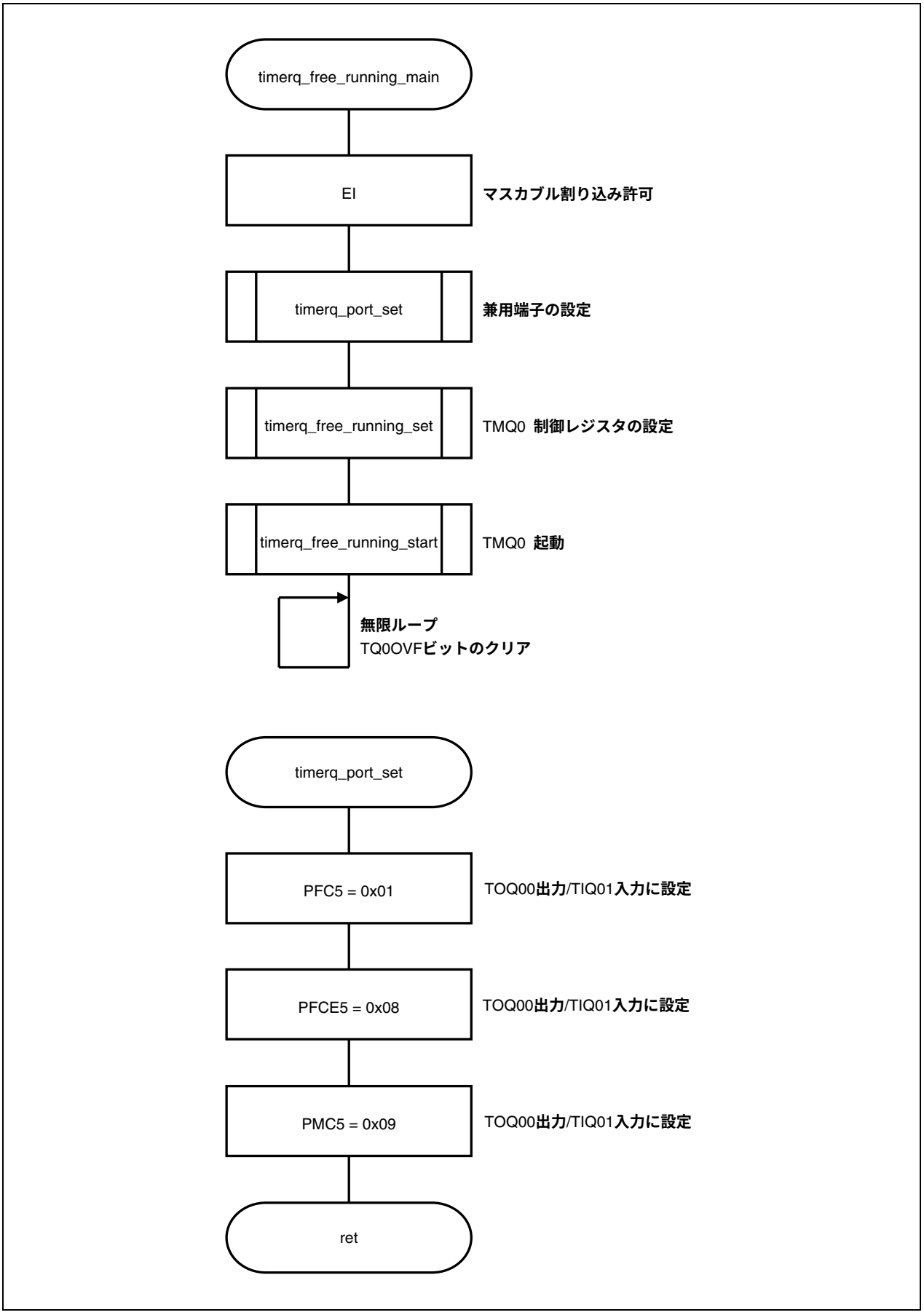


図4－15　フリー・ランニング・タイマ・モード（2）

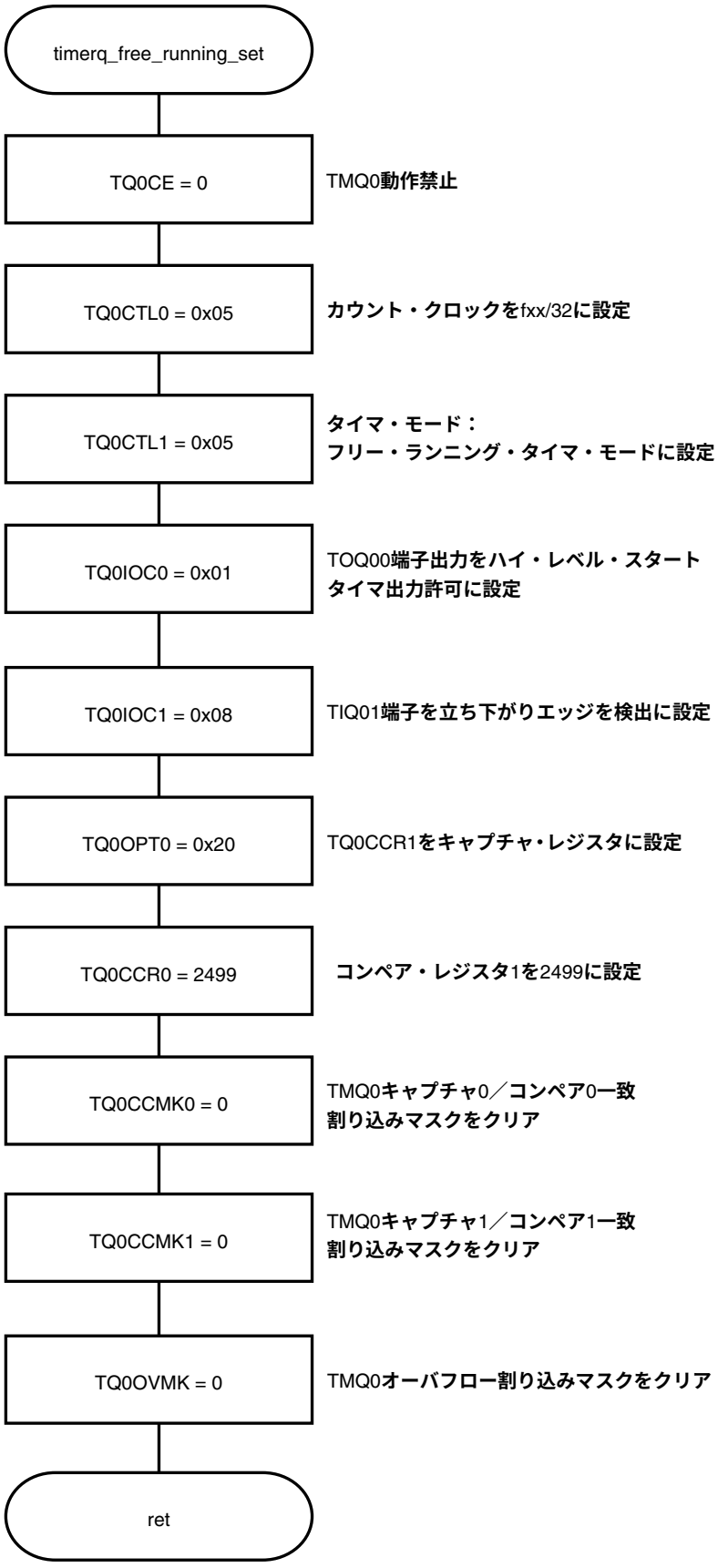
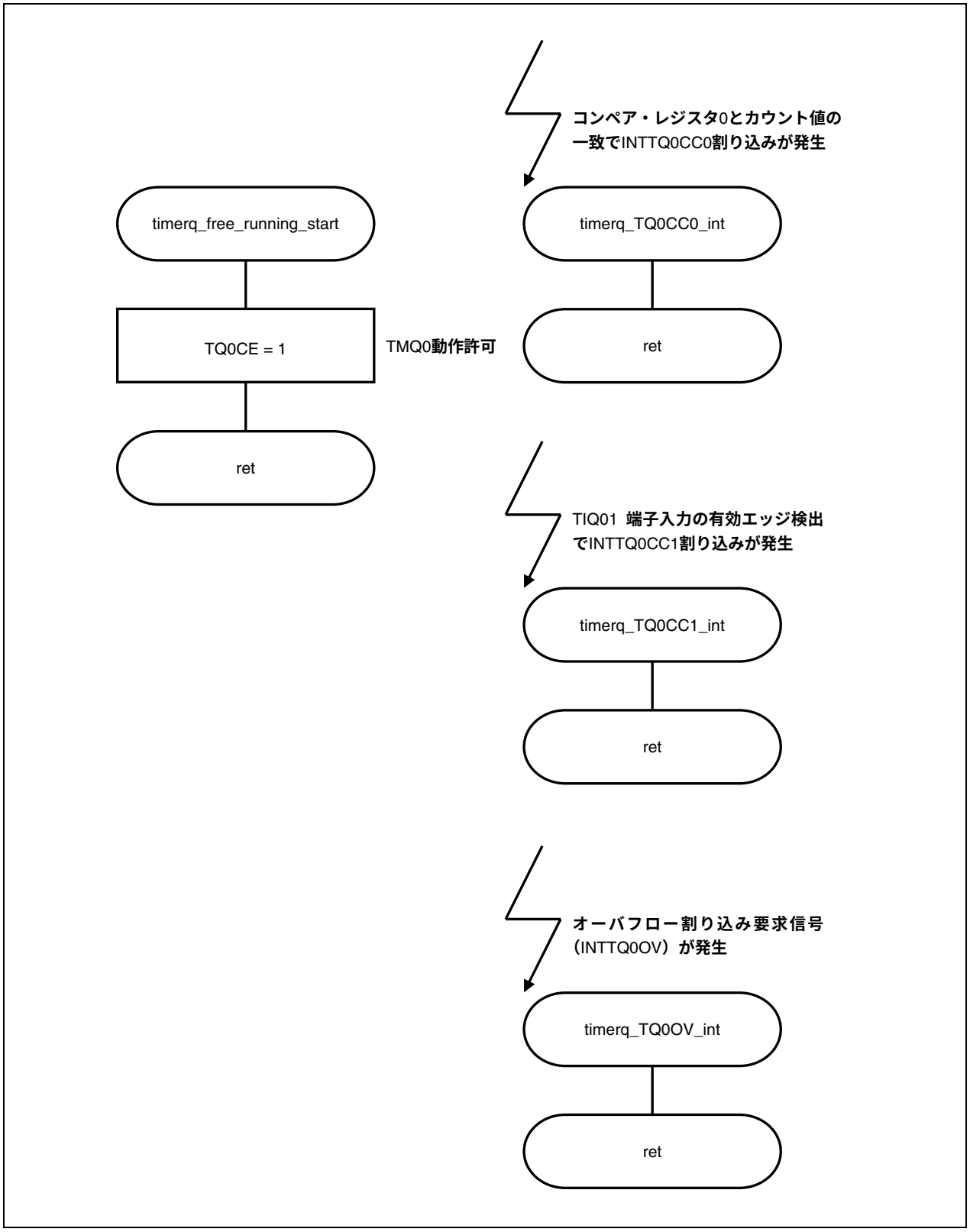


図4－16　フリー・ランニング・タイマ・モード（3）



4.7　パルス幅測定モード

【機　　能】	TQ0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 キャプチャ・トリガ入力（TIQ00 端子）の有効エッジで 16 ビット・カウンタのカウント値を TQ0CCR0 レジスタに格納し，16 ビット・カウンタをクリアします。 また，TIQ00 端子入力の有効エッジで割り込みを発生させ，TQ0CCR0 レジスタの値を読み込むことにより TIQ00 端子の有効エッジ間隔を測定します。
【関　数　名】	timerq_pulse_measure
【引　き　数】	なし
【処 理 内 容】	fxx/32 のカウント・クロックでカウント動作を行い，TIQ00 端子入力の有効エッジで 16 ビット・カウンタのカウント値を TQ0CCR0 レジスタに格納して割り込みを発生し，カウンタをクリアします。 カウンタのオーバフローが検出されると，INTTQ0OV 割り込みを発生します。
【使 用 S F R】	なし
【call　　関数】	timerq_port_set，timerq_pulse_measure_set，timerq_ pulse_measure_start
【変　　数】	なし
【割 り 込 み】	timerq_TQ0CC0_int timerq_TQ0OV_int
【割り込み要因】	INTTQ0CC0 INTTQ0OV
【ファイル名】	timerq_pulse_measure¥timerq_pulse_measure.c,
【注 意 事 項】	なし

【関　数　名】	timerq_port_set
【引　き　数】	なし
【処 理 内 容】	ポート 5 を TIQ00 入力端子に設定します。
【使 用 S F R】	PFC5　　：0x08（TIQ00 入力端子の設定） PMC5　　：0x08（TIQ00 入力端子に設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_pulse_measure¥timerq_pulse_measure.c
【注 意 事 項】	なし

【関　数　名】　timerq_free_running _set

【引　き　数】　なし

【処　理　内　容】　TMQ0 制御レジスタの設定を行います。

【使 用　S　F　R】　TQ0CTL0.TQ0CE：0（TMQ0 動作禁止）
TQ0CTL0　：0x05（カウント・クロックを fxx/32 に設定）
TQ0CTL1　：0x06（タイマ・モードをパルス幅測定モードに設定）
TQ0IOC1　：0x02（TIQ00 端子を立ち下がりエッジを検出に設定）
TQ0IOC2　：0x08（TIQ00 端子を立ち下がりエッジを検出に設定）
TQ0CCMK0　：0（TMQ0 キャプチャ 0／コンペア 0 一致割り込みマスク・クリアに設定）
TQ0OVMK　：0（TMQ0 オーバフロー割り込みマスク・クリアに設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_pulse_measure¥timerq_pulse_measure.c

【注　意　事　項】　なし

パルス幅は次に示す式で計算できます。

パルス幅 = （TQ0CCR0 レジスタの値＋1）×カウント・クロック周期

16ビット・カウンタのオーバフローが検出された場合のパルス幅は次に示す式で計算できます。

パルス幅 = （TQ0CCR0 レジスタの値＋0x10001）×カウント・クロック周期

【関　数　名】　timerq_pulse_measure_start

【引　き　数】　なし

【処　理　内　容】　パルス幅測定モードの起動関数です。

【起　動　方　法】　timerq_pulse_measure_set 関数のあとにコールしてください。

【使 用　S　F　R】　TQ0CTL0.TQ0CE　：1（TMQ0 動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　timerq_pulse_measure¥timerq_pulse_measure.c

【注　意　事　項】　なし

割り込み関数

【関　数　名】	timerq_TQ0CC0_int
【概　　要】	ユーザ定義
【要　　因】	INTTQ0CC0　　TIQ00 端子入力の有効エッジ検出
【call　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_pulse_measure¥timerq_pulse_measure.c
【注意事項】	なし

【関　数　名】	timerq_TQ0OV_int
【概　　要】	ユーザ定義
【要　　因】	INTTQ0OV　　16 ビット・カウンタのオーバフロー発生
【call　関数】	なし
【変　　数】	なし
【ファイル名】	timerq_pulse_measure¥timerq_pulse_measure.c
【注意事項】	なし

図4－17　パルス幅測定モード（1）

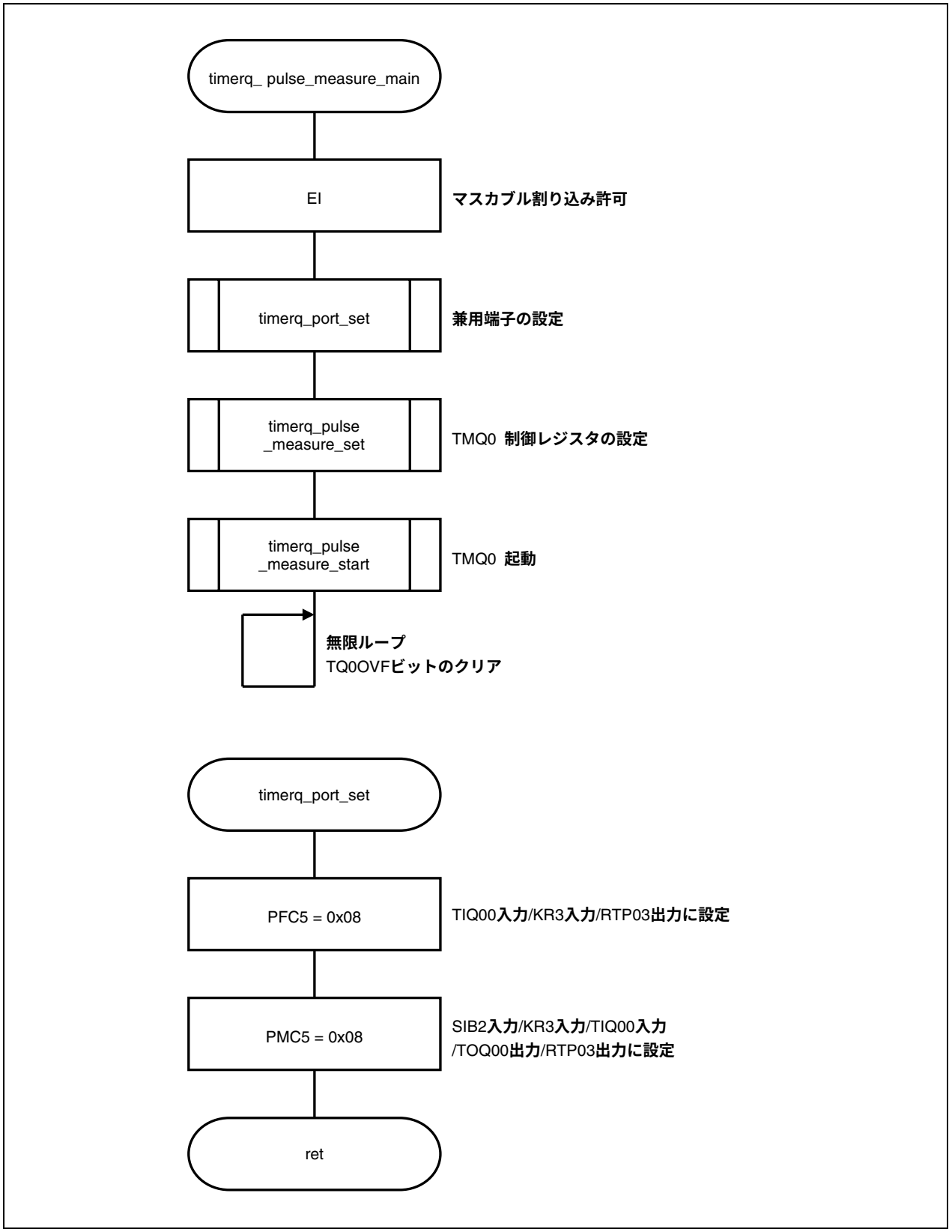


図4－18　パルス幅測定モード（2）

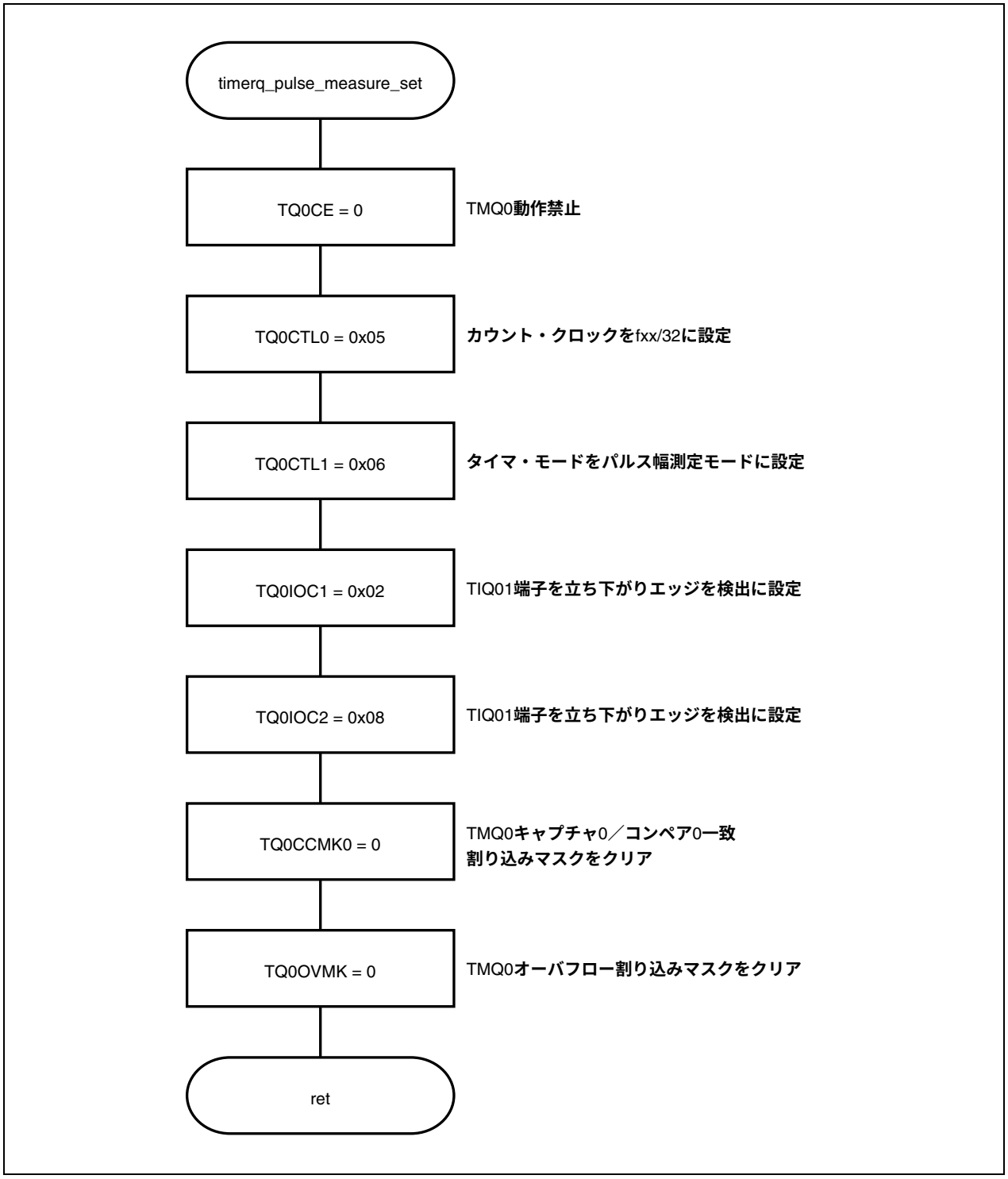
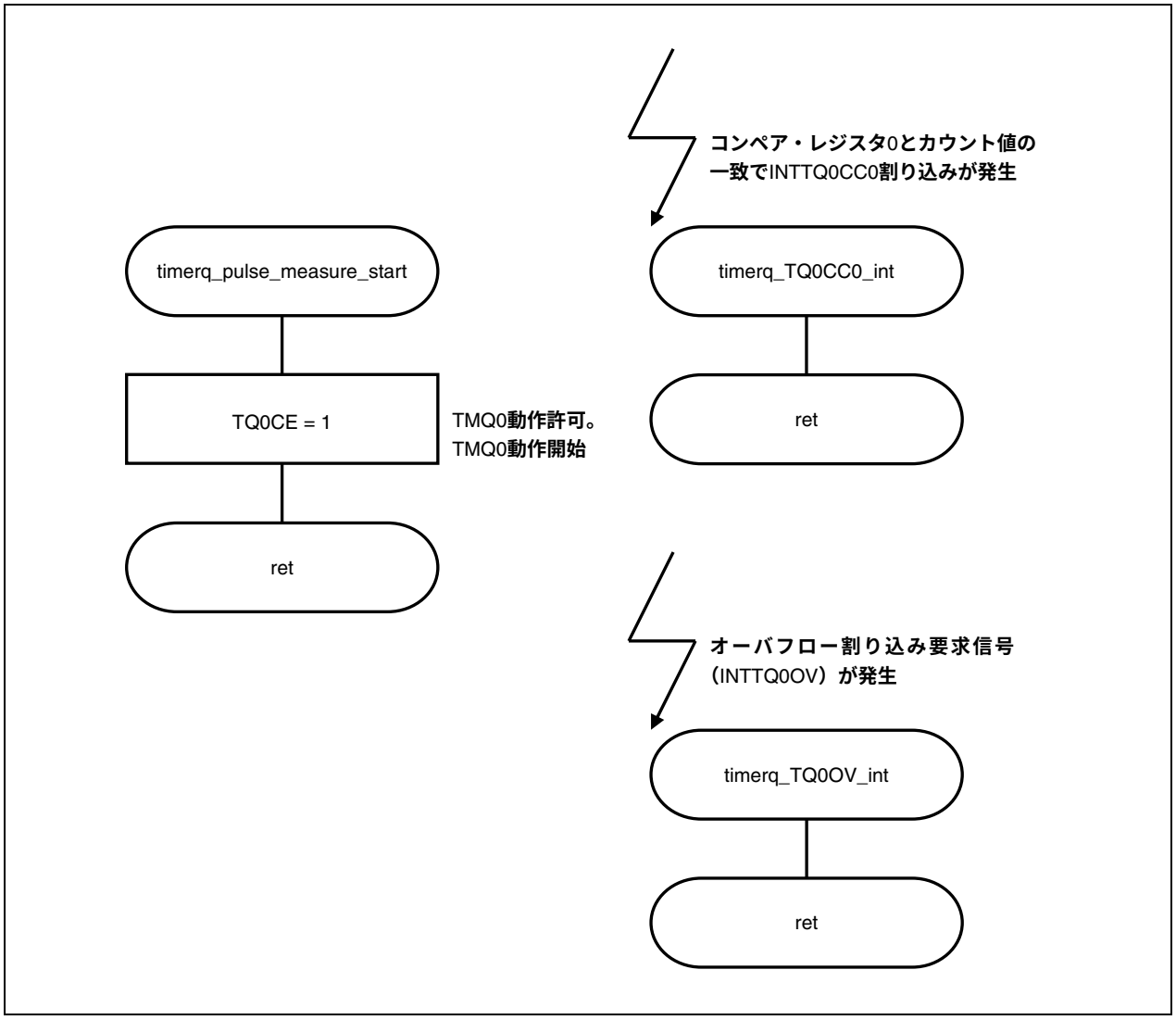


図4－19　パルス幅測定モード（3）



第5章　16ビット・インターバル・タイマM（TMM）

5. 1　インターバル・タイマ・モード

【機　　能】	TM0CE ビットをセットすることで 16 ビット・カウンタの動作を開始します。 TM0CMP0 レジスタで設定したインターバル間隔で割り込み要求信号（INTTM0EQ0）を発生します。
【関　数　名】	timerm_interval_main
【引　き　数】	なし
【処　理　内　容】	fxx/4 のカウント・クロックでカウント動作を行い，カウンタの値が TM0CMP0 レジスタの値と一致した次のカウンタのタイミングで，割り込みを発生します。
【使 用 S F R】	なし
【call　　関数】	timerm_interval_set，timerm_interval_start
【変　　数】	なし
【割　り　込　み】	timerm_TM0EQ0_int
【割り込み要因】	INTTM0EQ0
【ファイル名】	timerm_interval¥timerm_interval.c
【注　意　事　項】	なし

【関　数　名】	timerm_interval_set	
【引　き　数】	なし	
【処　理　内　容】	TMM0 制御レジスタの設定を行います。	
【使 用 S F R】	TM0CTL0.TM0CE	：0（TMM0 動作禁止）
	TM0CTL0	：0x02（カウント・クロックを fxx/4 に設定）
	TM0CMP0	：1249（16 ビット・カウンタのコンペア・レジスタ 0 を 1249 に設定）
	TM0EQMK0	：0（TMM0 コンペア一致割り込みマスク・クリアに設定）
【call　　関数】	なし	
【変　　数】	なし	
【ファイル名】	timerm_interval¥timerm_interval.c	
【注　意　事　項】	なし	

設定時間は次の式により決定します（クロック・スルー・モード（5 MHz）動作時）。

同期クロック：5 MHz = 1/5 ⇒ 0.2 μs

カウント・クロック周期（fxx/4 の場合）：0.2 μs×4 ⇒ 0.8 μs

1 ms で割り込みを発生させる場合のコンペア・レジスタの値：1000 μs/0.8 μs ⇒ 1250－1 = 1249

【関　数　名】 timerm_interval_start

【引　き　数】 なし

【処 理 内 容】 インターバル・タイマ・モードの起動関数です。

【起 動 方 法】 timerm_interval_set 関数のあとにコールしてください。

【使 用 S F R】 TM0CTL0.TM0CE ：1（TMM0 動作許可）

【call　関数】 なし

【変　数】 なし

【フ ァ イ ル 名】 timerm_interval¥timerm_interval.c

【注 意 事 項】 なし

割り込み関数

【関　数　名】 timerm_TM0EQ0_int

【概　要】 ユーザ定義

【要　因】 INTTM0EQ0　16 ビット・カウンタのカウント値と TM0CMP0 の一致

【call　関数】 なし

【変　数】 なし

【フ ァ イ ル 名】 timerm_interval¥timerm_interval.c

【注 意 事 項】 なし

図5－1　インターバル・タイマ・モード（1）

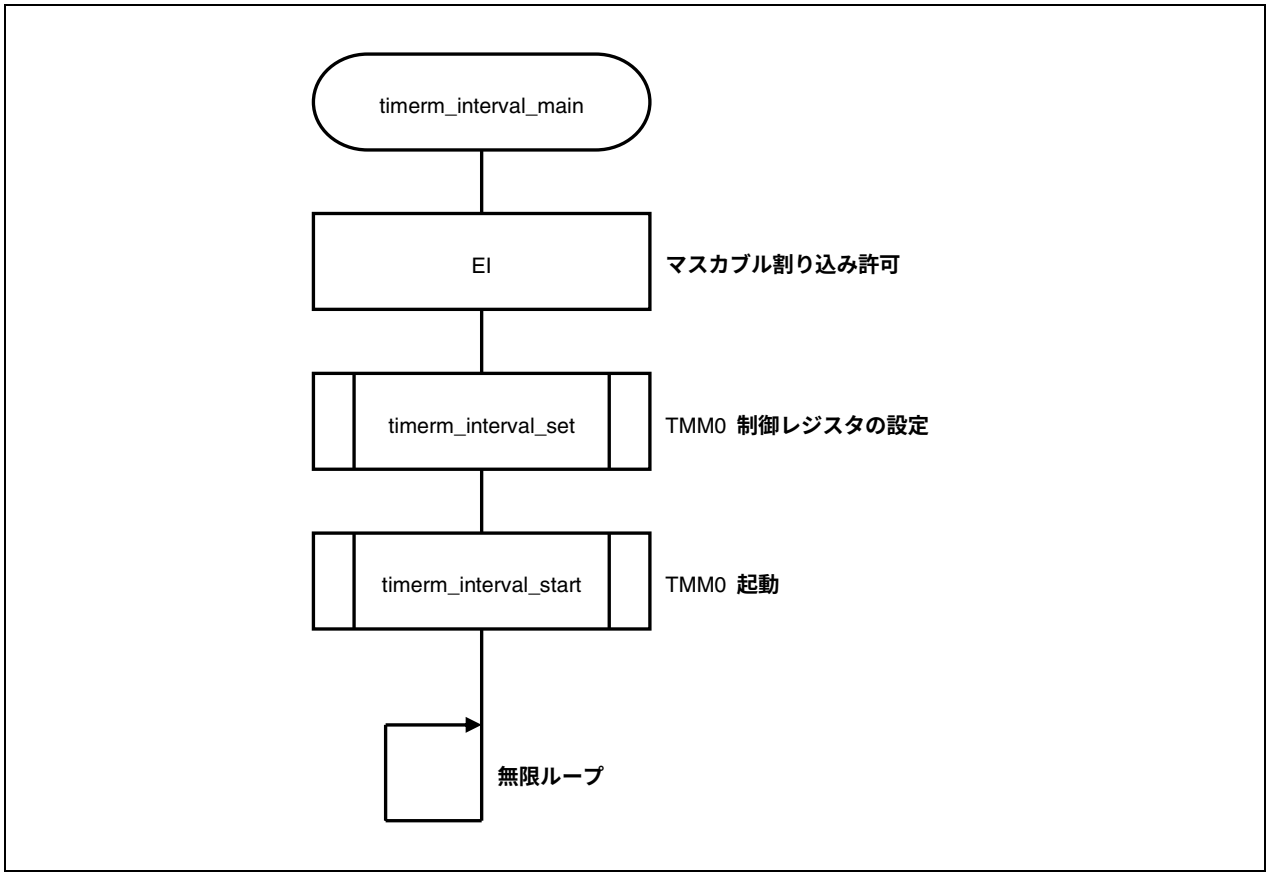
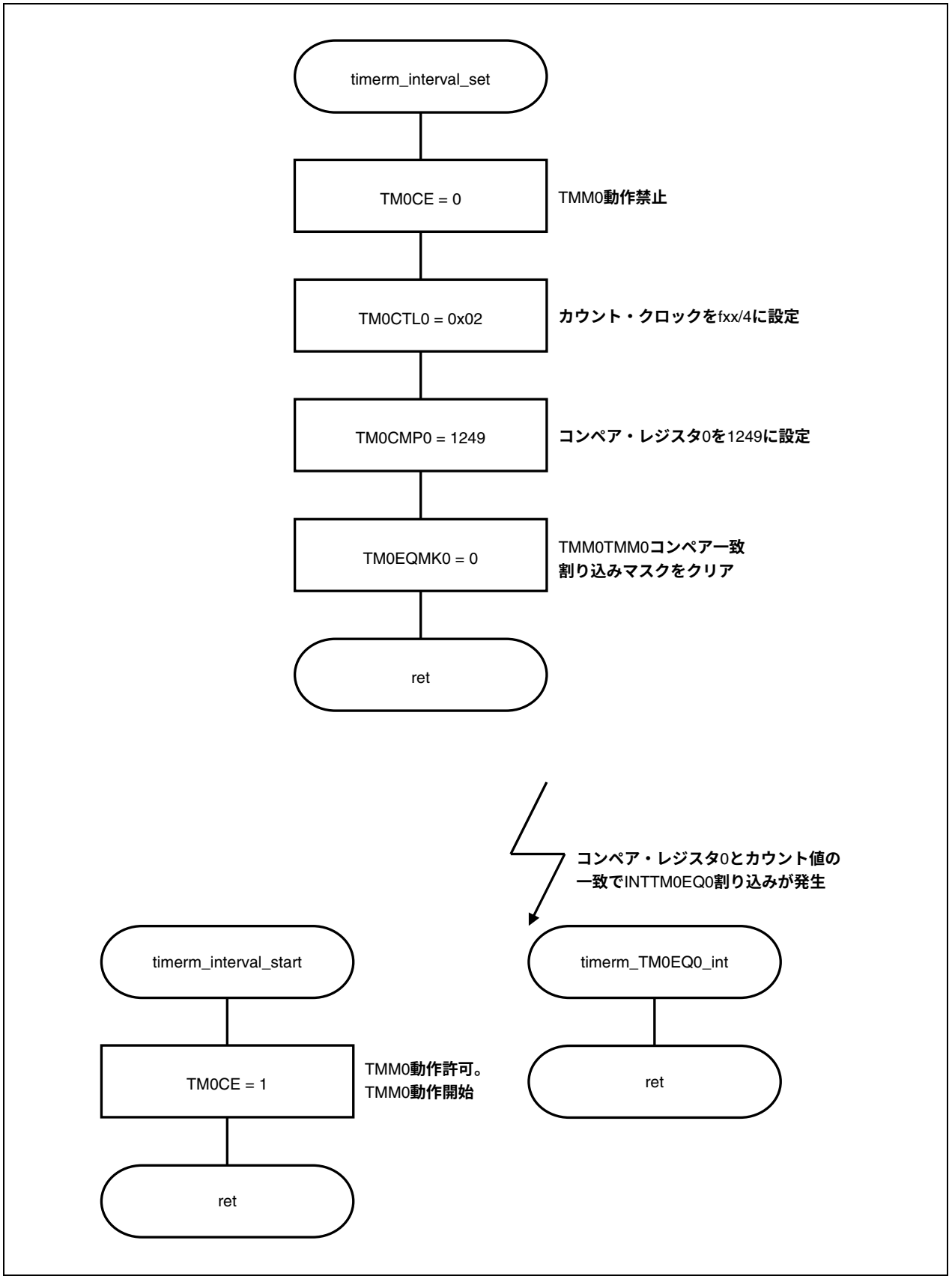


図5－2　インターバル・タイマ・モード（2）



第6章　時計タイマ機能

6.1　時計タイマ

【機　　能】	WTM.WTM1, WTM0 ビットに“11”を設定するとカウント動作がスタートします。 メイン・クロックを使用して，0.5 秒の時間間隔の時計タイマとして動作します。
【関　数　名】	timer_time_main
【引　き　数】	なし
【処　理　内　容】	0.5 秒の時間間隔で割り込み要求信号（INTWT）を発生させます。
【使 用 S F R】	なし
【call　関数】	timer_time_set，timer_time_start
【変　　数】	なし
【割　り　込　み】	timer_WT_int
【割り込み要因】	INTWT
【ファイル名】	timer_time¥timer_time.c
【注　意　事　項】	動作許可（WTM.WTM1, WTM0 ビット = 1）してから，最初の 1 回目の時計タイマ割り込み要求信号（INTWT）が発生するまで多少時間がかかります。

【関　数　名】	timer_time_set	
【引　き　数】	なし	
【処　理　内　容】	時計タイマ 制御レジスタの設定を行います。	
【使 用 S F R】	WTM.WTM0	：0（5 ビット・カウンタの動作を動作停止後クリアに設定）
	WTM.WTM1	：0（時計用タイマを動作禁止に設定）
	WTM	：0x00（時計用フラグのセット時間を 0.5 秒に設定）
	PRSCM0	：0x13（プリスケーラ・コンペア・レジスタ 0 を 19 に設定）
	PRSM0	：0x12（メイン・クロックでの動作を許可， 時計タイマ・ソース・クロックを fx/4 に設定）
	WTMK	：0（時計タイマの基準時間割り込みマスクをクリアに設定）
【call　関数】	なし	
【変　　数】	なし	
【ファイル名】	timer_time¥timer_time.c	
【注　意　事　項】	なし	

【関　数　名】　timer_time_start

【引　き　数】　なし

【処 理 内 容】　時計タイマの起動関数です。

【起 動 方 法】　timer_time_set 関数のあとにコールしてください。

【使 用 S F R】　WTM.WTM0　　：1（5 ビット・カウンタをスタートに設定）
　　WTM.WTM1　　：1（時計用タイマの動作を許可に設定）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　timer_time¥timer_time.c

【注 意 事 項】　なし

割り込み関数

【関　数　名】　timer_WT_int

【概　　要】　ユーザ定義

【要　　因】　INTWT　0.5 秒の経過

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　timer_time¥timer_time.c

【注 意 事 項】　なし

図6－1　時計タイマ（1）

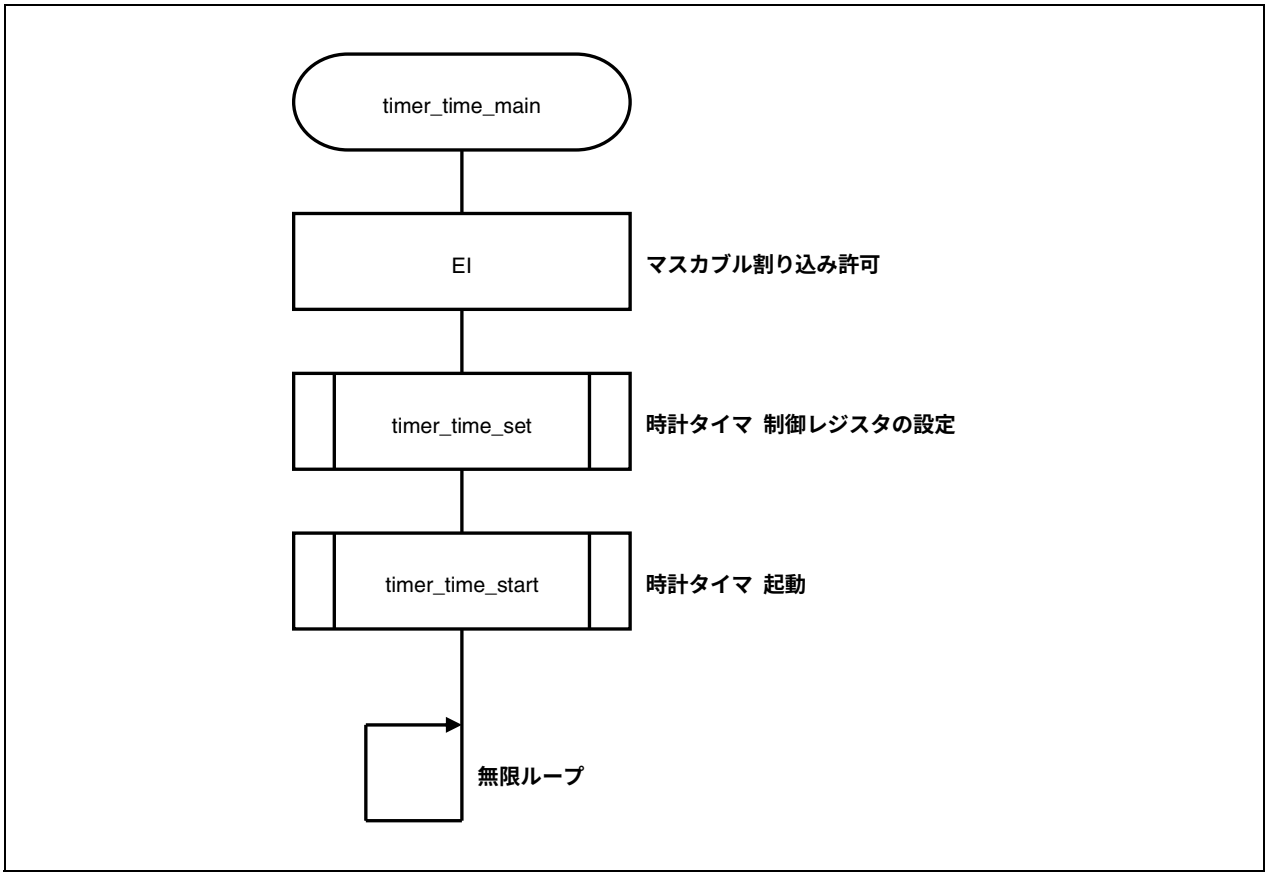
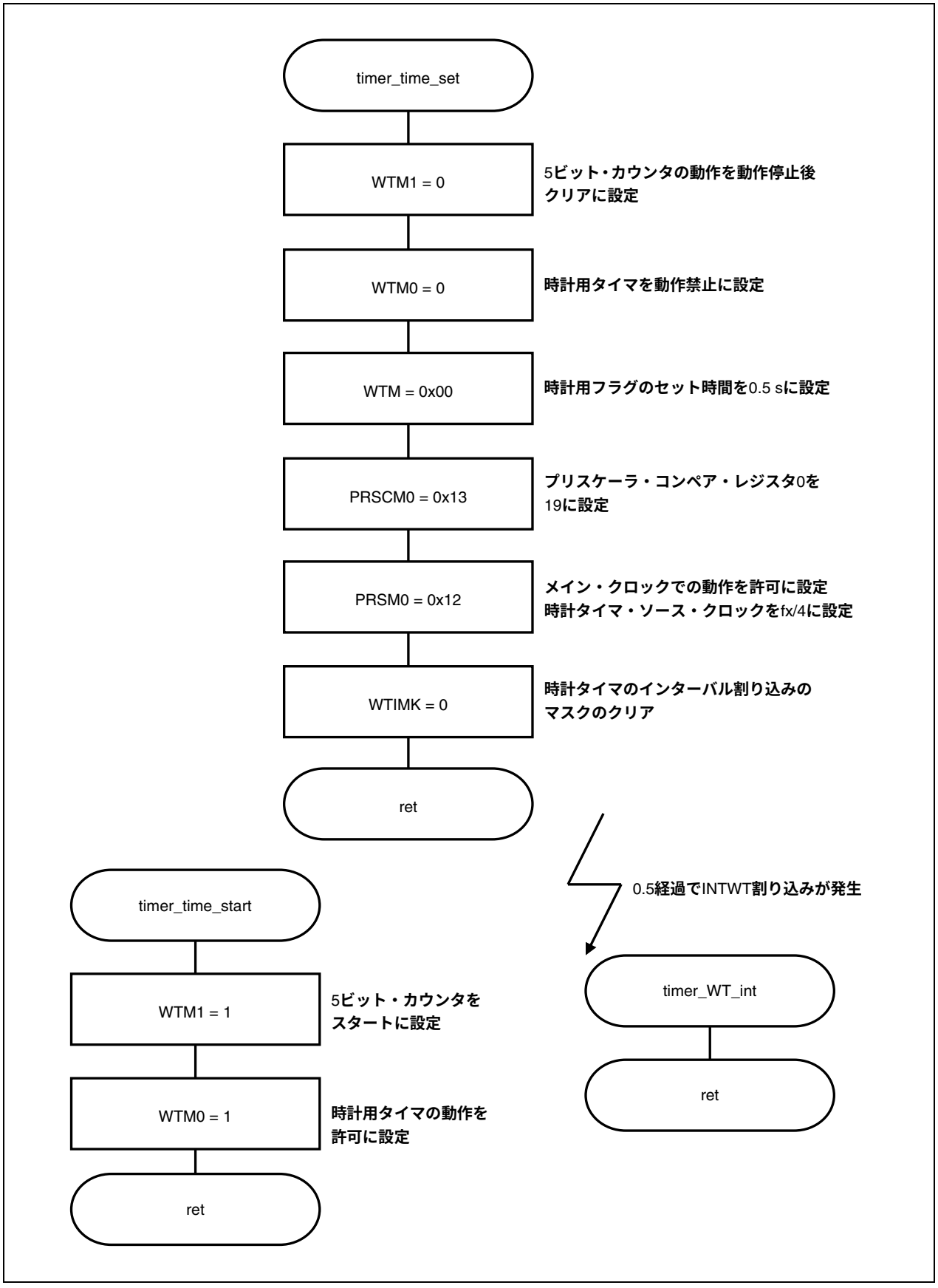

timer_time_main
EI
マスカブル割り込み許可
timer_time_set
時計タイマ 制御レジスタの設定
timer_time_start
時計タイマ 起動
無限ループ

図6－2　時計タイマ（2）


timer_time_set
WTM1 = 0
5ビット・カウンタの動作を動作停止後
クリアに設定
WTM0 = 0
時計用タイマを動作禁止に設定
WTM = 0x00
時計用フラグのセット時間を0.5 sに設定
PRSCM0 = 0x13
プリスケーラ・コンペア・レジスタ0を
19に設定
PRSM0 = 0x12
メイン・クロックでの動作を許可に設定
時計タイマ・ソース・クロックをfx/4に設定
WTIMK = 0
時計タイマのインターバル割り込みの
マスクのクリア
ret
0.5経過でINTWT割り込みが発生
timer_time_start
WTM1 = 1
5ビット・カウンタを
スタートに設定
WTM0 = 1
時計用タイマの動作を
許可に設定
ret
timer_WT_int
ret

6.2　インターバル・タイマ

【機　　能】	WTM レジスタの WTM4-WTM7 ビットで設定したカウント値をインターバルとし，繰り返し割り込み要求信号（INTWTI）を発生するインターバル・タイマとして動作します。
【関　数　名】	timer_interval_main
【引　き　数】	なし
【処　理　内　容】	インターバル時間で設定した値で割り込み要求信号（INTWTI）を発生させます。
【使 用 S F R】	なし
【call　関数】	timer_interval_set，timer_interval_start
【変　　数】	なし
【割　り　込　み】	timer_WTI_int
【割り込み要因】	INTWTI
【ファイル名】	timer_interval¥timer_interval.c
【注　意　事　項】	なし

【関　数　名】	timer_interval_set	
【引　き　数】	なし	
【処　理　内　容】	時計タイマ 制御レジスタの設定を行います。	
【使 用 S F R】	WTM.WTM0	：0（5 ビット・カウンタの動作を動作停止後クリアに設定）
	WTM.WTM1	：0（時計用タイマを動作禁止に設定）
	WTM	：0x00（プリスケーラのインターバル時間を 488 μs に設定）
	PRSCM0	：0x13（プリスケーラ・コンペア・レジスタ 0 を 19 に設定）
	PRSM0	：0x12（メイン・クロックでの動作を許可，時計タイマ・ソース・クロックを fx/4 に設定）
	WTIMK	：0（時計タイマのインターバル割り込みのマスクのクリアに設定）
【call　関数】	なし	
【変　　数】	なし	
【ファイル名】	timer_interval¥timer_interval.c	
【注　意　事　項】	なし	

【関　数　名】 timer_interval_start

【引　き　数】 なし

【処 理 内 容】 インターバル・タイマの起動関数です。

【起 動 方 法】 timer_interval_set 関数のあとにコールしてください。

【使 用 ＳＦＲ】 WTM.WTM0 ：1（5 ビット・カウンタをスタートに設定）
WTM.WTM1 ：1（時計用タイマの動作を許可に設定）

【call　関数】 なし

【変　数】 なし

【ファイル名】 timer_interval¥timer_interval.c

【注 意 事 項】 なし

割り込み関数

【関　数　名】 timer_WTI_int

【概　要】 ユーザ定義

【要　因】 INTWTI　インターバル時間の経過

【call　関数】 なし

【変　数】 なし

【ファイル名】 timer_interval¥timer_interval.c

【注 意 事 項】 なし

図6－3　インターバル・タイマ（1）

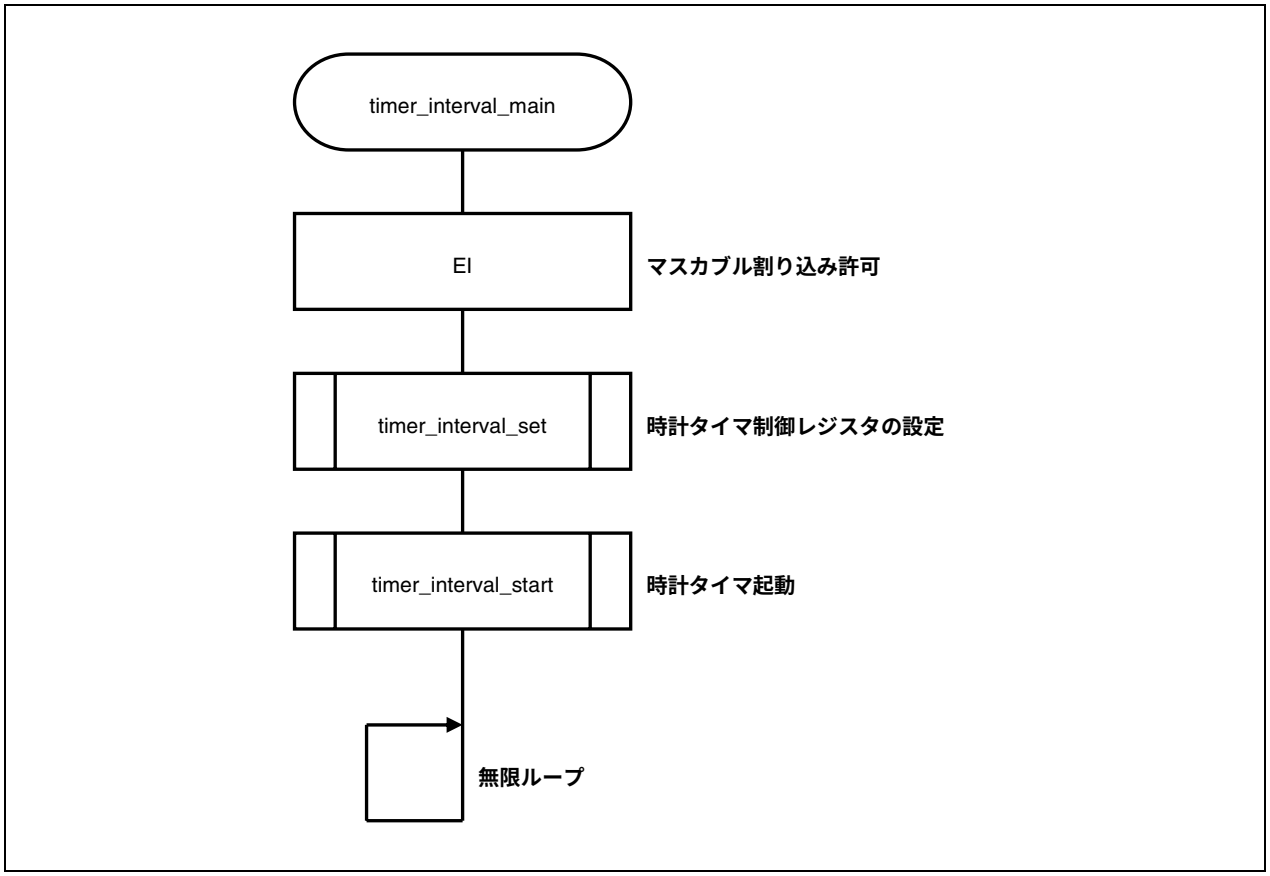
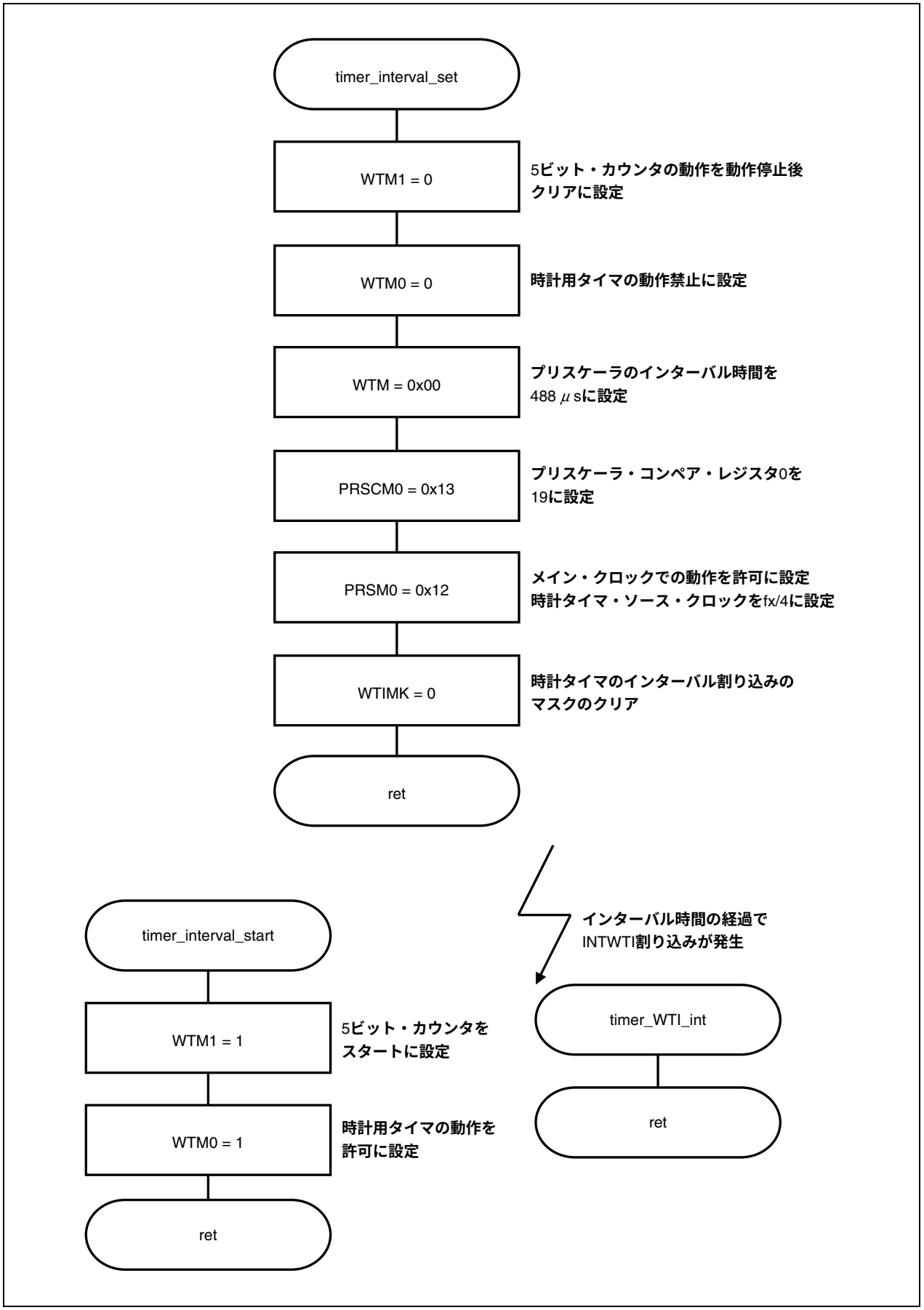


図6－4　インターバル・タイマ（2）



第7章　ウォッチドッグ・タイマ機能

7.1　リセット・モード

【機　　能】	リセット解除後に自動的にリセット・モードでスタートします。 オーバフロー時間を 2^{18}/fxx に設定し，WDTE レジスタに ACH が書き込まれず，暴走検出時間を越えてしまった場合，リセット信号（WDT2RES）を発生します。
【関　数　名】	WDT_reset_main
【引　き　数】	なし
【処　理　内　容】	WDTE レジスタに ACH が書き込まれずオーバフロー時間を越えてしまった場合，リセット信号（WDT2RES）を発生します。
【使 用 S F R】	なし
【call　　関数】	WDT_reset_set，WDT_reset_start_clr
【変　　数】	なし
【割　り　込　み】	なし
【割り込み要因】	WDT2RES
【ファイル名】	WDT_reset¥WDT_reset.c
【注　意　事　項】	WDTM2 レジスタへの書き込みは，バイト・アクセスのみリセット後に一度だけ可能です。

【関　数　名】	WDT_reset_set
【引　き　数】	なし
【処　理　内　容】	WDT 制御レジスタの設定を行います。
【使 用 S F R】	WDTM2　　：0x48（オーバフロー時間を 2^{18}/fxx に設定， 動作モードをリセット・モードに設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	WDT_reset¥WDT_reset.c
【注　意　事　項】	なし

【関　　数　　名】　WDT_reset_start_clr

【引　　き　　数】　なし

【処　理　内　容】　WDT の起動関数です。

【使 用　S　F　R】　WDTE　　：0xAC（WDT のカウンタをクリアし再スタート）

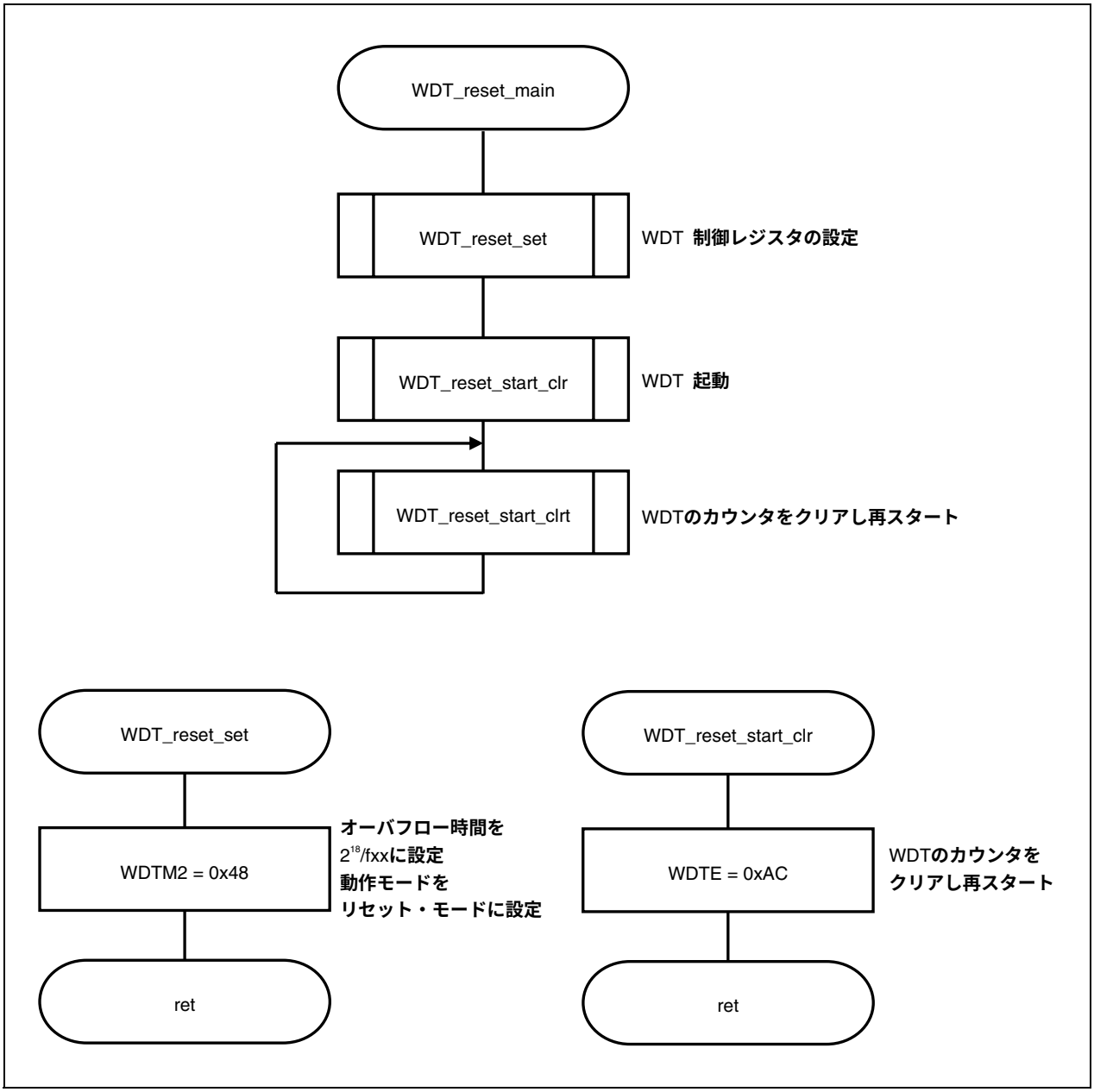
【call　　関数】　なし

【変　　数】　なし

【ファイル名】　WDT_reset¥WDT_reset.c

【注　意　事　項】　なし

図7－1　リセット・モード



7.2　ノンマスカブル割り込み要求モード

【機　　能】	リセット解除後に自動的にリセット・モードでスタートします。 オーバフロー時間を 2^{18}/fxx に設定し，WDTE レジスタに ACH が書き込まれず，暴走検出時間を越えてしまった場合，ノンマスカブル割り込み要求信号（INTWDT2）を発生します。
【関　数　名】	WDT_nmi_main
【引　き　数】	なし
【処 理 内 容】	WDTE レジスタに ACH が書き込まれずオーバフロー時間を越えてしまった場合，ノンマスカブル割り込み要求信号（INTWDT2）を発生します。
【使 用 ＳＦＲ】	なし
【call　関数】	WDT_nmi_set，WDT_nmi_start_clr
【変　　数】	なし
【割 り 込 み】	WDT_INTWDT2_int
【割り込み要因】	INTWDT2
【ファイル名】	WDT_nmi¥WDT_nmi.c
【注 意 事 項】	WDTM2 レジスタへの書き込みは，バイト・アクセスのみリセット後に一度だけ可能です。

【関　数　名】	WDT_nmi_set
【引　き　数】	なし
【処 理 内 容】	WDT 制御レジスタの設定を行います。
【使 用 ＳＦＲ】	WDTM2　：0x28（オーバフロー時間を 2^{18}/fxx に設定， 動作モードをリセット・モードに設定）
【call　関数】	なし
【変　　数】	なし
【ファイル名】	WDT_nmi¥WDT_nmi.c
【注 意 事 項】	なし

【関　　数　　名】　WDT_nmi_start_clr

【引　　き　　数】　なし

【処　理　内　容】　WDT の起動関数です。

【使 用　S　F　R】　WDTE　　：0xAC（WDT のカウンタをクリアし再スタート）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　WDT_nmi¥WDT_nmi.c

【注　意　事　項】　なし

割り込み関数

【関　　数　　名】　WDT_nmi_int

【概　　要】　ユーザ定義

【要　　因】　INTWDT2　　オーバフロー時間経過

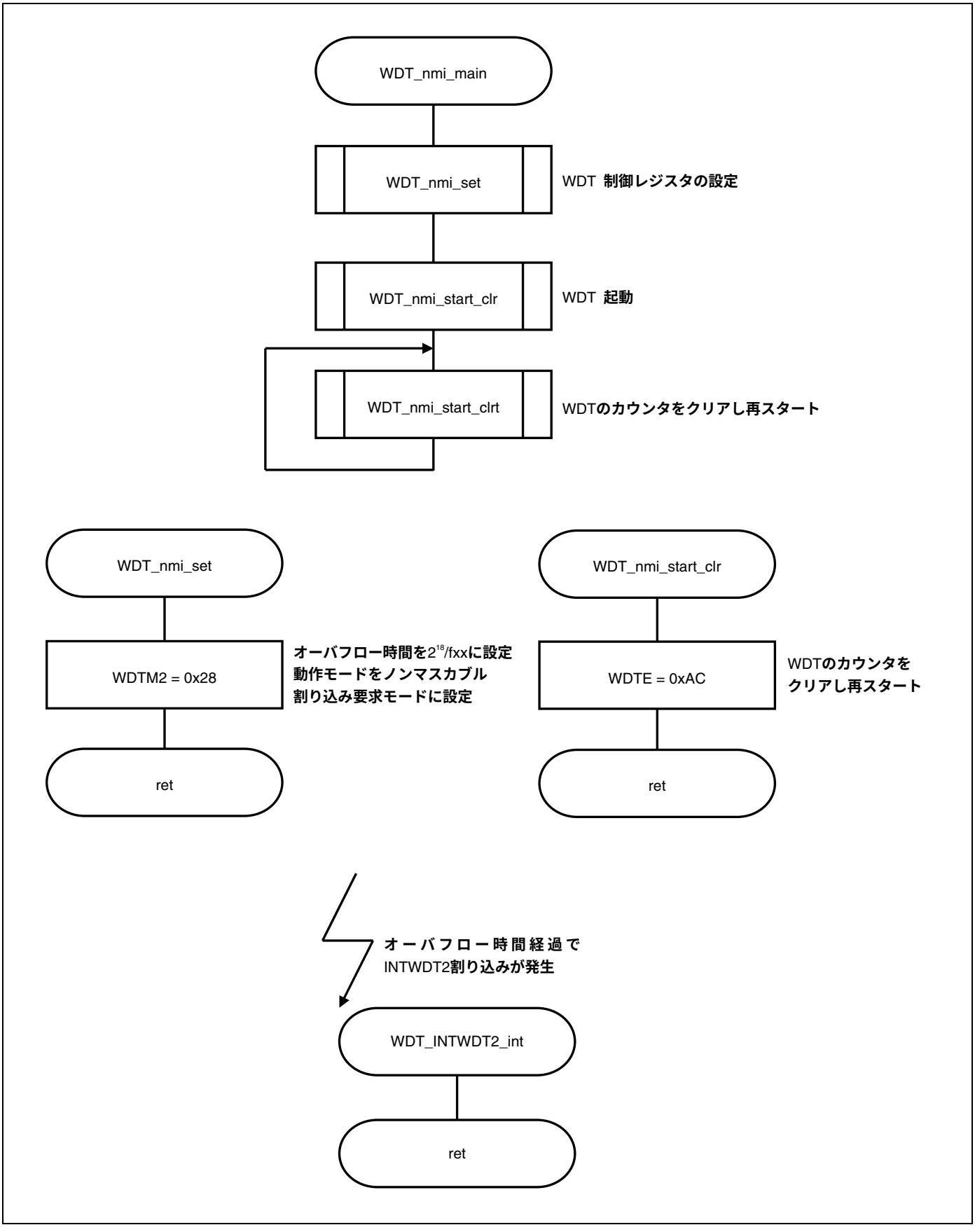
【call　　関数】　なし

【変　　数】　なし

【ファイル名】　WDT_nmi¥WDT_nmi.c

【注　意　事　項】　なし

図7－2　ノンマスカブル割り込み要求モード



第8章　A/Dコンバータ

8.1　外部トリガ・モード（連続スキャン・モード）

【機　　能】	A/D 変換動作開始タイミングをハードウエア・トリガ・モードの外部トリガ・モードに設定し，A/D 変換を行います。
【関　数　名】	ad_external_main
【引　き　数】	なし
【処 理 内 容】	ADTRG 端子からトリガが入力されると，ANI0 端子，ANI1 端子，ANI2 端子，ANI3 端子までを順に選択し，A/D 変換を連続で行い，A/D 変換結果をアナログ入力端子に対応した buf[ ]，buf_1[ ]，buf_2[ ]，buf_3[ ]に格納します。 指定したアナログ入力端子の変換動作が終了すると A/D 変換終了割り込み要求信号（INTAD）が発生します。 A/D 変換は 10 回行います。
【使 用 S F R】	ADMK　：0（A/D 変換終了割り込み要求信号（INTAD）クリア，マスク解除，優先レベル 7 に設定）
【call　関数】	ad_trgger_port_set, ad_port_set, ad_set, ad_start, ad_stop
【変　　数】	unsigned short int buf [ ]　：変換データ格納バッファ unsigned short int buf_1 [ ]　：変換データ格納バッファ unsigned short int buf _2[ ]　：変換データ格納バッファ unsigned short int buf _3[ ]　：変換データ格納バッファ volatile unsigned char count　：変換カウント変数
【割 り 込 み】	ad_int
【割り込み要因】	INTAD
【ファイル名】	ad_external_trigger¥ad_external_trigger.c
【注 意 事 項】	なし

【関　数　名】　ad_trigger_port_set

【処　理　内　容】　ポート 0 を ADTRG 入力端子に設定します。

【使 用 S F R】　PMC0　：0x08（ADTRG 入力端子に設定）
　　PFC0　：0x08（ADTRG 入力端子に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　ad_external_trigger¥ad_external_trigger.c

【関　数　名】　ad_port_set

【処　理　内　容】　アナログ入力端子の設定を行います。

【使 用 S F R】　ADA0S　：0x03（アナログ入力端子を ANI0 端子，ANI1 端子，ANI2 端子，ANI3 端子に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　ad_external_trigger¥ad_external_trigger.c

【関　数　名】　ad_set

【処　理　内　容】　A/D 変換制御レジスタの設定を行います。

【使 用 S F R】　ADA0M0　：0x16（連続スキャン・モード，有効エッジを立ち下がりエッジ，ハードウエア・トリガ・モードに設定）
　　ADA0M2　：0x00（外部トリガ・モードに設定）
　　ADA0M1　：0x80（高速変換モード，変換クロック数を 6.5 μs に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　ad_external_trigger¥ad_external_trigger.c

【関　数　名】　ad_start

【処　理　内　容】　A/D 変換動作を許可します。

【使 用 S F R】　ADA0M0.ADA0CE　：1（A/D 変換動作許可）

【call　関数】　なし

【変　数】　なし

【ファイル名】　ad_external_trigger¥ad_external_trigger.c

【関　数　名】　ad_stop

【処　理　内　容】　A/D 変換動作を停止します。

【使 用　S　F　R】　ADA0M0.ADA0CE　：0（A/D 変換動作停止）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　ad_external_trigger¥ad_external_trigger.c

割り込み関数

【関　数　名】　ad_int

【処　理　内　容】　A/D 変換結果データをバッファに格納します。

【使 用　S　F　R】　ADA0CR0　A/D 変換結果レジスタ 0
ADA0CR1　A/D 変換結果レジスタ 1
ADA0CR2　A/D 変換結果レジスタ 2
ADA0CR3　A/D 変換結果レジスタ 3

【call　　関数】　なし

【変　　数】　unsigned short int buf [ ]　：変換データ格納バッファ
unsigned short int buf _1[ ]　：変換データ格納バッファ
unsigned short int buf _2[ ]　：変換データ格納バッファ
unsigned short int buf _3[ ]　：変換データ格納バッファ
volatile unsigned char count　：変換カウント変数

【ファイル名】　ad_external_trigger¥ad_external_trigger.c

図8－1　外部トリガ・モード（連続スキャン・モード）（1）

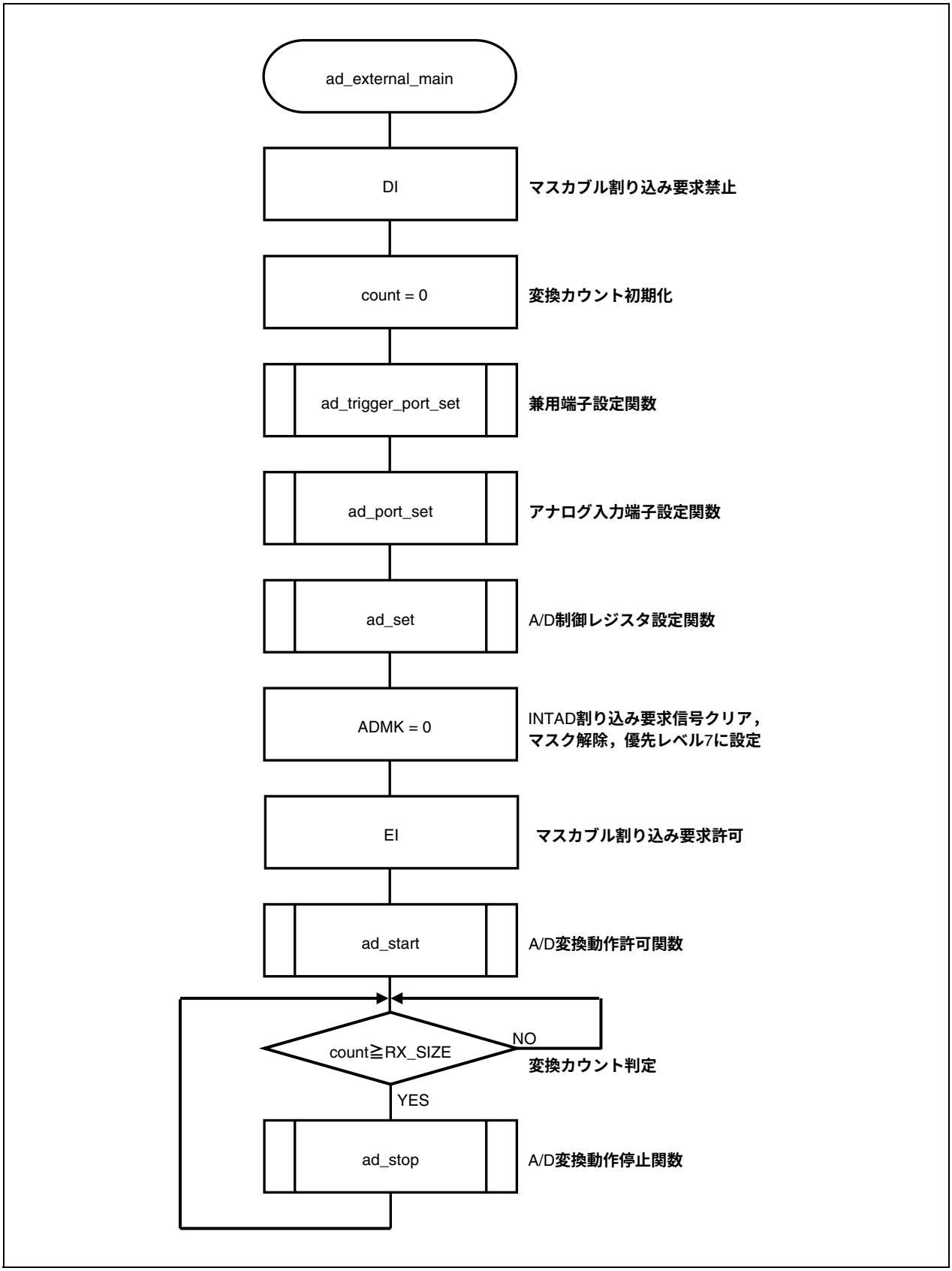


図8－2　外部トリガ・モード（連続スキャン・モード）（2）

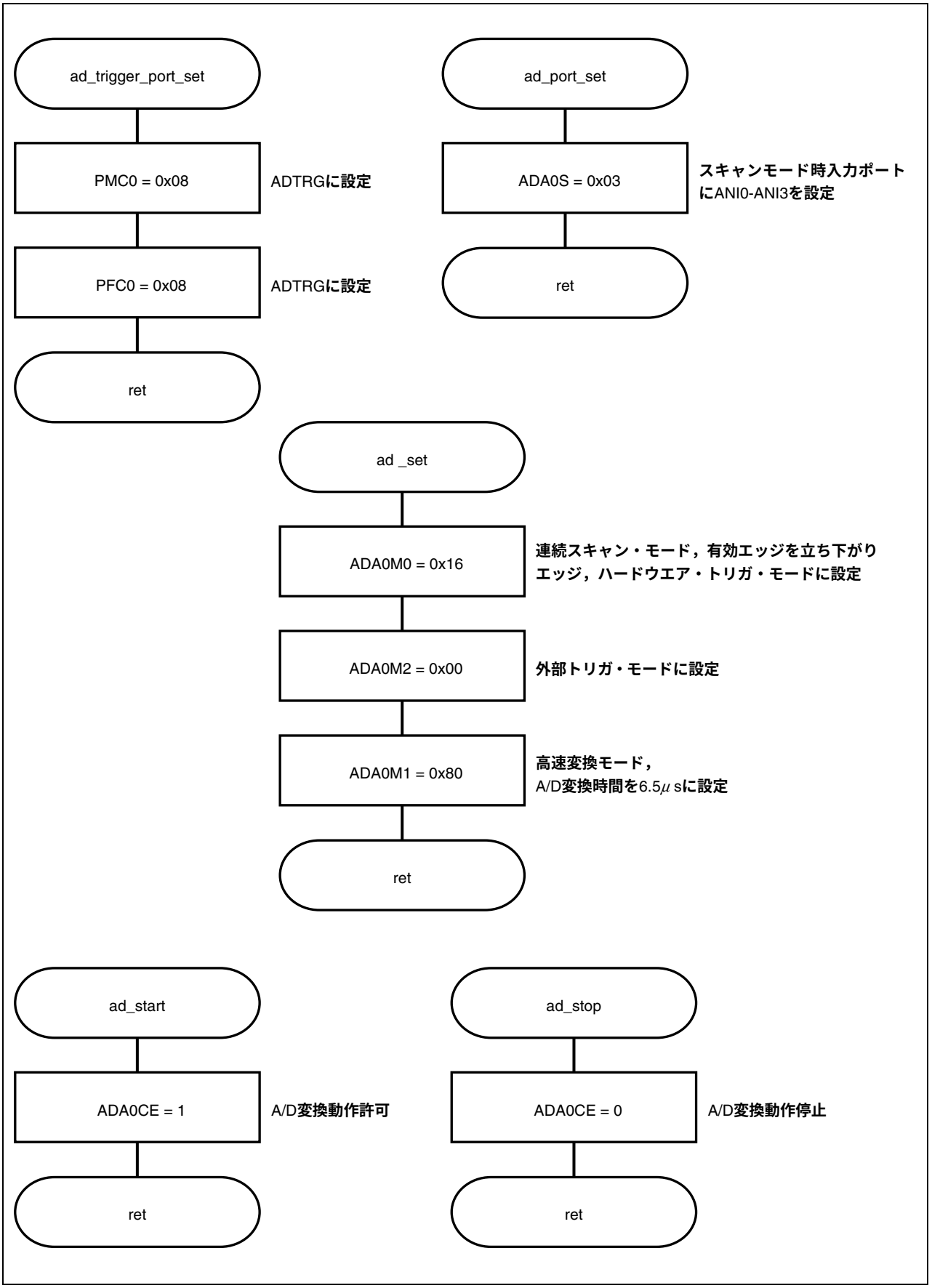
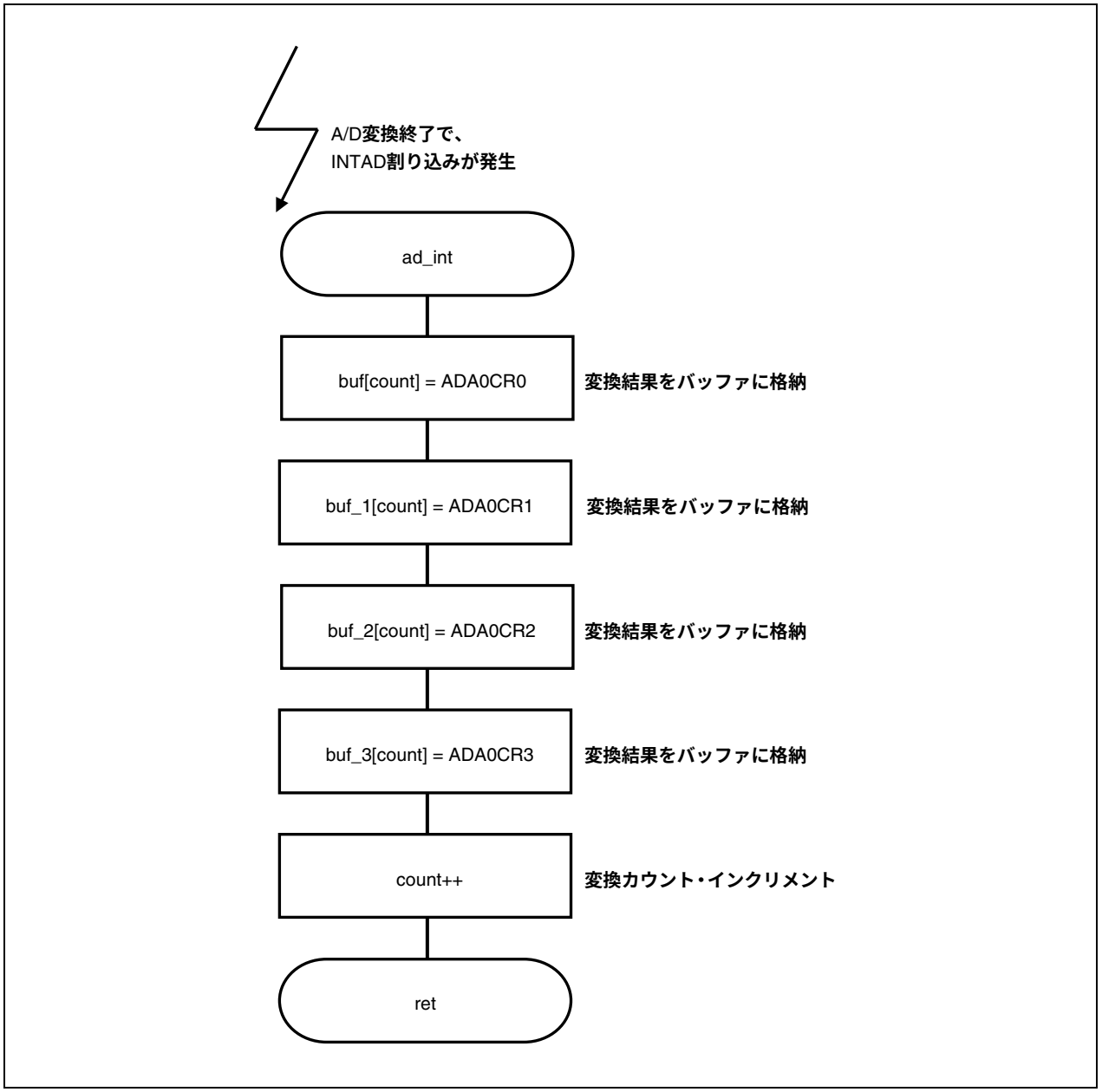


図8－3　外部トリガ・モード（連続スキャン・モード）（3）



8.2 ソフトウエア・トリガ・モード（連続セレクト・モード）

【機　　能】	A/D 変換動作開始タイミングをソフトウエア・トリガ・モードに設定し，A/D 変換を行います。
【関　数　名】	ad_software_main
【引　き　数】	なし
【処 理 内 容】	ADA0M0.ADA0CE ビットをセット（1）することで，A/D 変換を開始します。ANI0 端子を A/D 変換し，A/D 変換結果を buf[ ]に格納します。 1 回の A/D 変換終了ごとに A/D 変換終了割り込み要求信号（INTAD）が発生します。 A/D 変換は 10 回行います。
【使 用 S F R】	ADMK　：0（A/D 変換終了割り込み要求信号（INTAD）クリア，マスク解除，優先レベル 7 に設定）
【call　関数】	ad_port_set, ad_set, ad_start, ad_stop
【変　　数】	unsigned short int buf [ ]　：変換データ格納バッファ volatile unsigned char count　：変換カウント変数
【割 り 込 み】	ad_int
【割り込み要因】	INTAD
【ファイル名】	ad_software_trigger¥ad_software_trigger.c
【注 意 事 項】	なし

【関　数　名】	ad_port_set
【処 理 内 容】	アナログ入力端子の設定を行います。
【使 用 S F R】	ADA0S　：0x00（アナログ入力端子を ANI00 に設定）
【call　関数】	なし
【変　　数】	なし
【ファイル名】	ad_software_trigger¥ad_software_trigger.c

【関　数　名】　ad_set

【処　理　内　容】　A/D 変換制御レジスタの設定を行います。

【使 用　S　F　R】　ADA0M0　　：0x00（連続セレクト・モード，有効エッジをエッジ検出なし，ソフトウエア・トリガ・モードに設定）

ADA0M1　　：0x00（通常変換モード，変換クロック数を 16.25 μs に設定）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　ad_software_trigger¥ad_software_trigger.c

【関　数　名】　ad_start

【処　理　内　容】　A/D 変換動作を開始します。

【使 用　S　F　R】　ADA0M0.ADA0CE　：1（A/D 変換動作許可）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　ad_software_trigger¥ad_software_trigger.c

【関　数　名】　ad_stop

【処　理　内　容】　A/D 変換動作を停止します。

【使 用　S　F　R】　ADA0M0.ADA0CE　：0（A/D 変換動作停止）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　ad_software_trigger¥ad_software_trigger.c

割り込み関数

【関　数　名】　ad_int

【処　理　内　容】　A/D 変換結果データをバッファに格納します。

【使 用　S　F　R】　ADA0CR0　　A/D 変換結果レジスタ 0

【call　　関数】　なし

【変　　数】　unsigned short int buf [ ]　　：変換データ格納バッファ

volatile unsigned char count　　：変換カウント変数

【ファイル名】　ad_software_trigger¥ad_software_trigger.c

図8－4　ソフトウエア・トリガ・モード（連続セレクト・モード）（1）

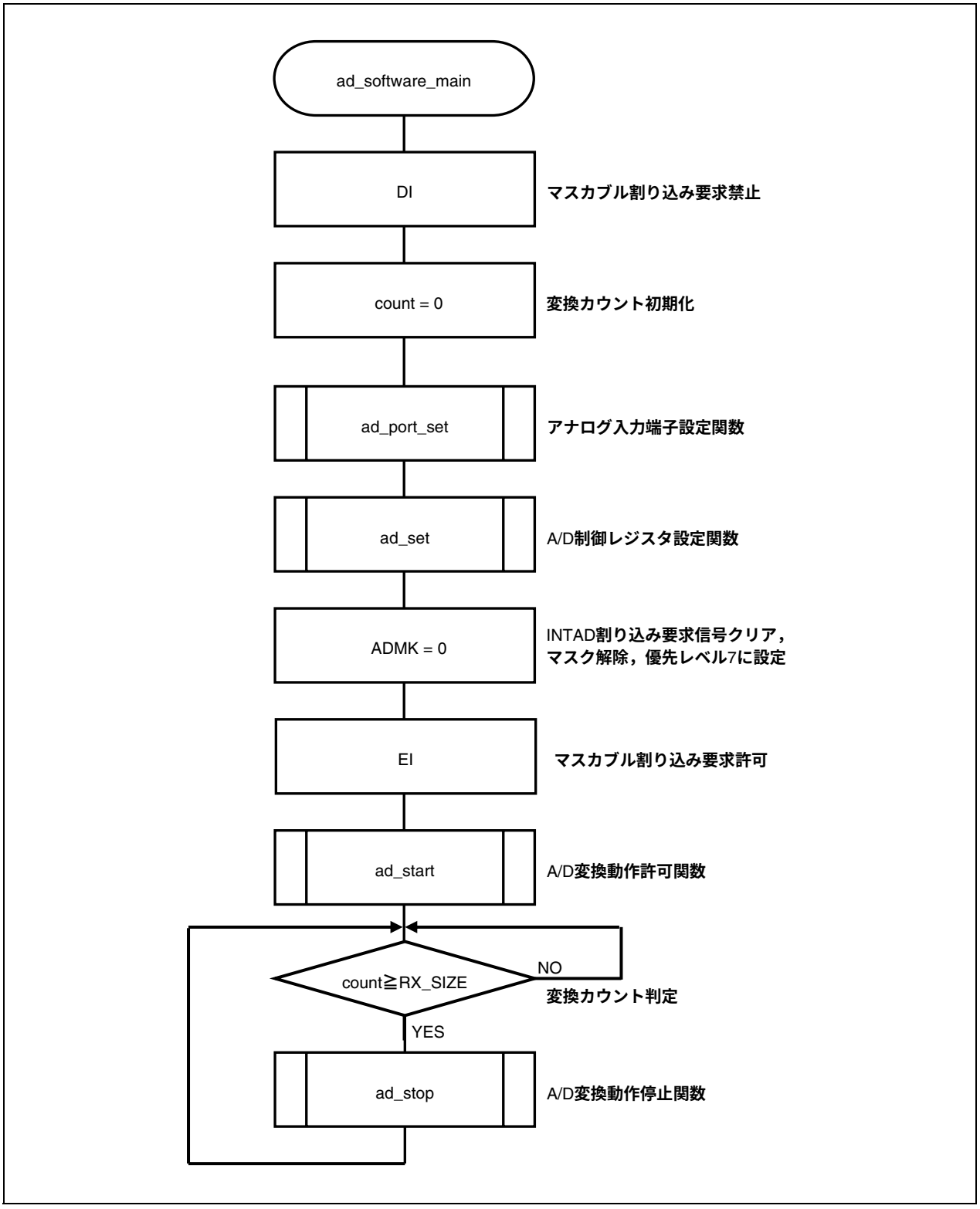
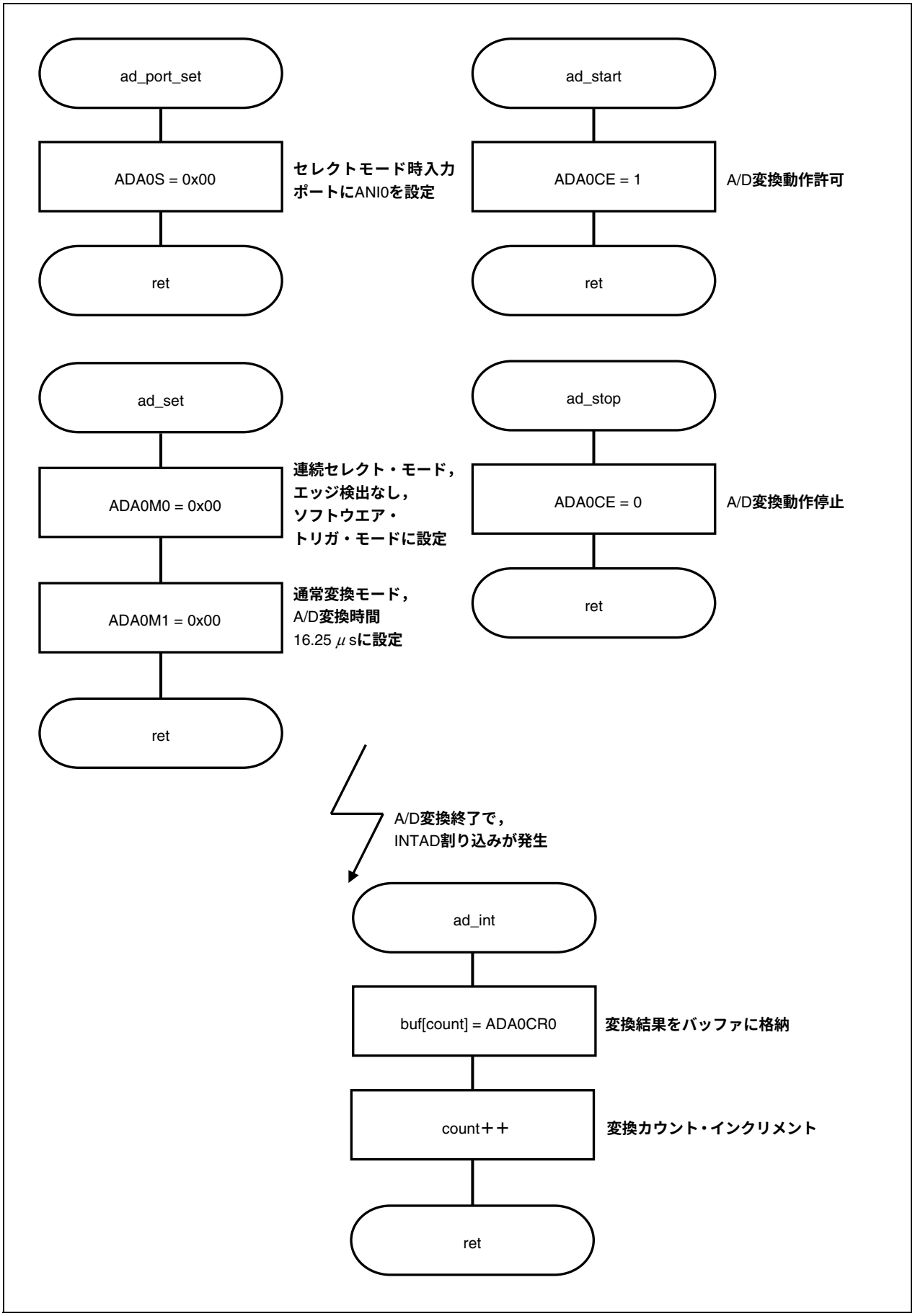


図8－5　ソフトウエア・トリガ・モード（連続セレクト・モード）（2）



8. 3　ソフトウエア・トリガ・モード（ワンショット・スキャン・モード）

【機　　能】	A/D 変換動作開始タイミングをソフトウエア・トリガ・モードに設定し，A/D 変換を行います。
【関　数　名】	ad_software1_main
【引　き　数】	なし
【処　理　内　容】	ADA0M0.ADA0CE ビットをセット（1）することで，A/D 変換を開始します。ANI0 端子，ANI1 端子，ANI2 端子，ANI3 端子までを順に選択し，A/D 変換を連続で行い，A/D 変換結果をアナログ入力端子に対応した buf[ ]，buf_1[ ]，buf_2[ ]，buf_3[ ]に格納します。 1 回の A/D 変換終了ごとに A/D 変換終了割り込み要求信号（INTAD）が発生します。 A/D 変換は 10 回行います。
【使 用 S F R】	ADMK　：0（A/D 変換終了割り込み要求信号（INTAD）クリア， マスク解除，優先レベル 7 に設定）
【call　関数】	ad_port_set, ad_set, ad_start, ad_stop
【変　　数】	unsigned short int buf [ ]　：変換データ格納バッファ unsigned short int buf_1 [ ]　：変換データ格納バッファ unsigned short int buf _2[ ]　：変換データ格納バッファ unsigned short int buf _3[ ]　：変換データ格納バッファ volatile unsigned char count　：変換カウント変数
【割　り　込　み】	ad_int
【割り込み要因】	INTAD
【ファイル名】	ad_software1_trigger¥ad_software1_trigger.c
【注　意　事　項】	なし

【関　数　名】	ad_port_set
【処　理　内　容】	アナログ入力端子の設定を行います。
【使 用 S F R】	ADA0S　：0x03（アナログ入力端子を ANI0 端子，ANI1 端子，ANI2 端子，ANI3 端子に設定）
【call　関数】	なし
【変　　数】	なし
【ファイル名】	ad_software1_trigger¥ad_software1_trigger.c

【関　数　名】	ad_set
【処　理　内　容】	A/D 変換制御レジスタの設定を行います。
【使 用 S F R】	ADA0M0　：0x30（ワンショット・スキャン・モード，有効エッジをエッジ検出なし，ソフトウエア・トリガ・モードに設定）
	ADA0M1　：0x00（通常変換モード，変換クロック数を 16.25 μs に設定）
【call　関数】	なし
【変　数】	なし
【ファイル名】	ad_software1_trigger¥ad_software1_trigger.c

【関　数　名】	ad_start
【処　理　内　容】	A/D 変換動作を開始します。
【使 用 S F R】	ADA0M0.ADA0CE　：1（A/D 変換動作許可）
【call　関数】	なし
【変　数】	なし
【ファイル名】	ad_software1_trigger¥ad_software1_trigger.c

【関　数　名】	ad_stop
【処　理　内　容】	A/D 変換動作を停止します。
【使 用 S F R】	ADA0M0.ADA0CE　：0（A/D 変換動作停止）
【call　関数】	なし
【変　数】	なし
【ファイル名】	ad_software1_trigger¥ad_software1_trigger.c

割り込み関数

【関　数　名】	ad_int	
【処　理　内　容】	A/D 変換結果データをバッファに格納します。	
【使 用　S　F　R】	ADA0CR0	A/D 変換結果レジスタ 0
	ADA0CR1	A/D 変換結果レジスタ 1
	ADA0CR2	A/D 変換結果レジスタ 2
	ADA0CR3	A/D 変換結果レジスタ 3
【call　関数】	なし	
【変　数】	unsigned short int buf [ ]	：変換データ格納バッファ
	unsigned short int buf _1[ ]	：変換データ格納バッファ
	unsigned short int buf _2[ ]	：変換データ格納バッファ
	unsigned short int buf _3[ ]	：変換データ格納バッファ
	volatile unsigned char count	：変換カウント変数
【フ ァ イ ル 名】	ad_software1_trigger¥ad_software1_trigger.c	

図8－6　ソフトウエア・トリガ・モード（ワンショット・スキャン・モード）（1）

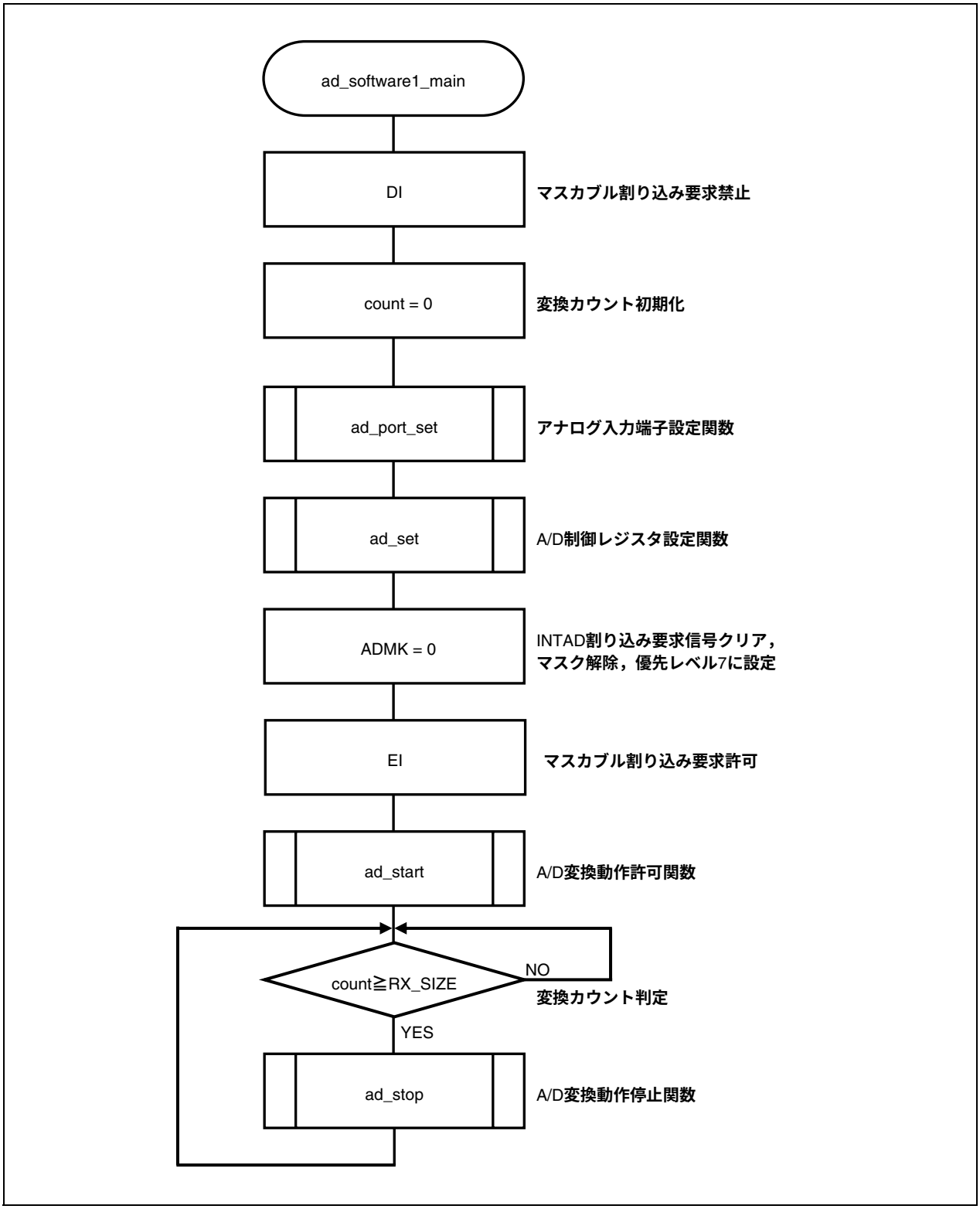


図8－7　ソフトウエア・トリガ・モード（ワンショット・スキャン・モード）（2）

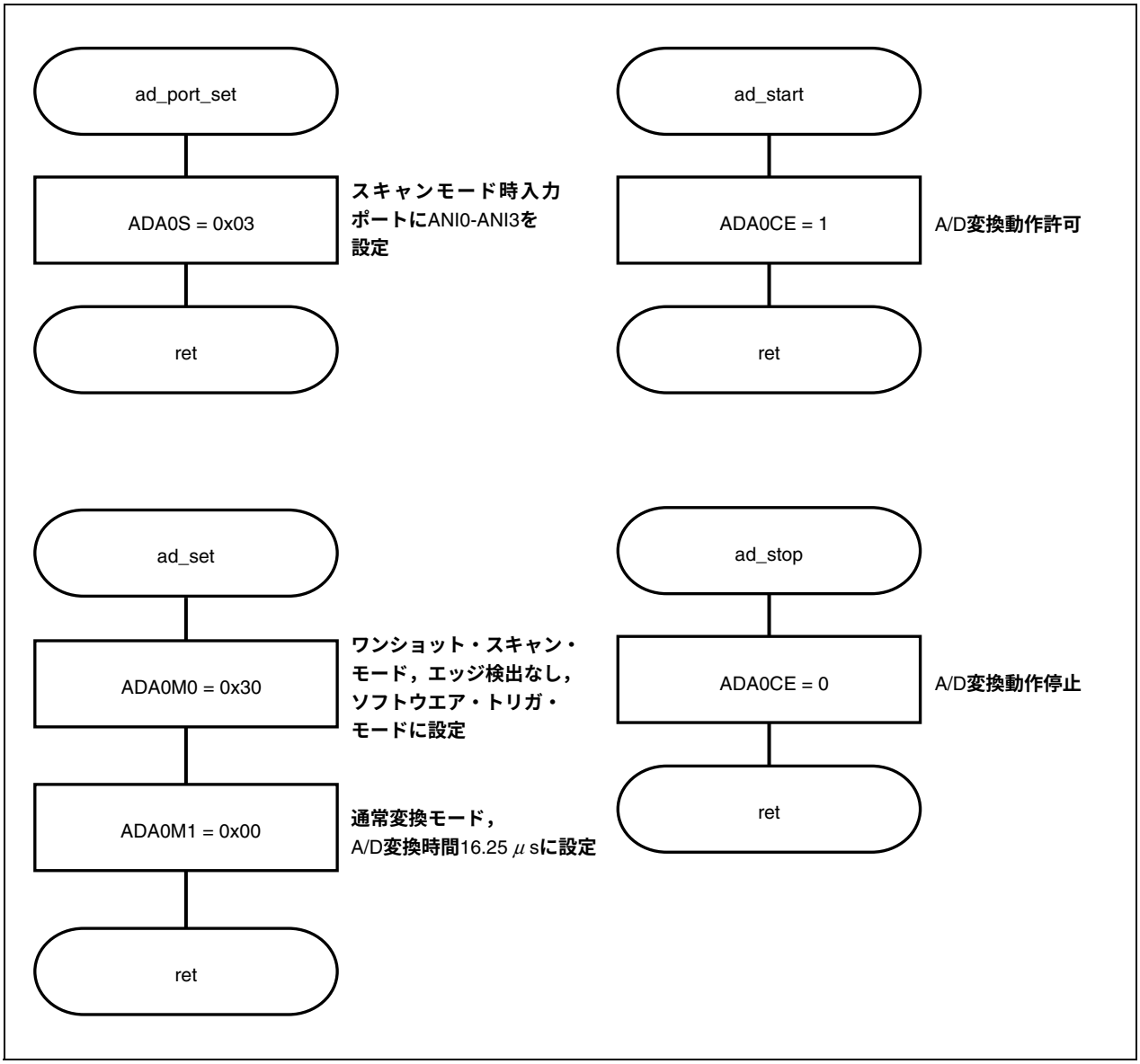
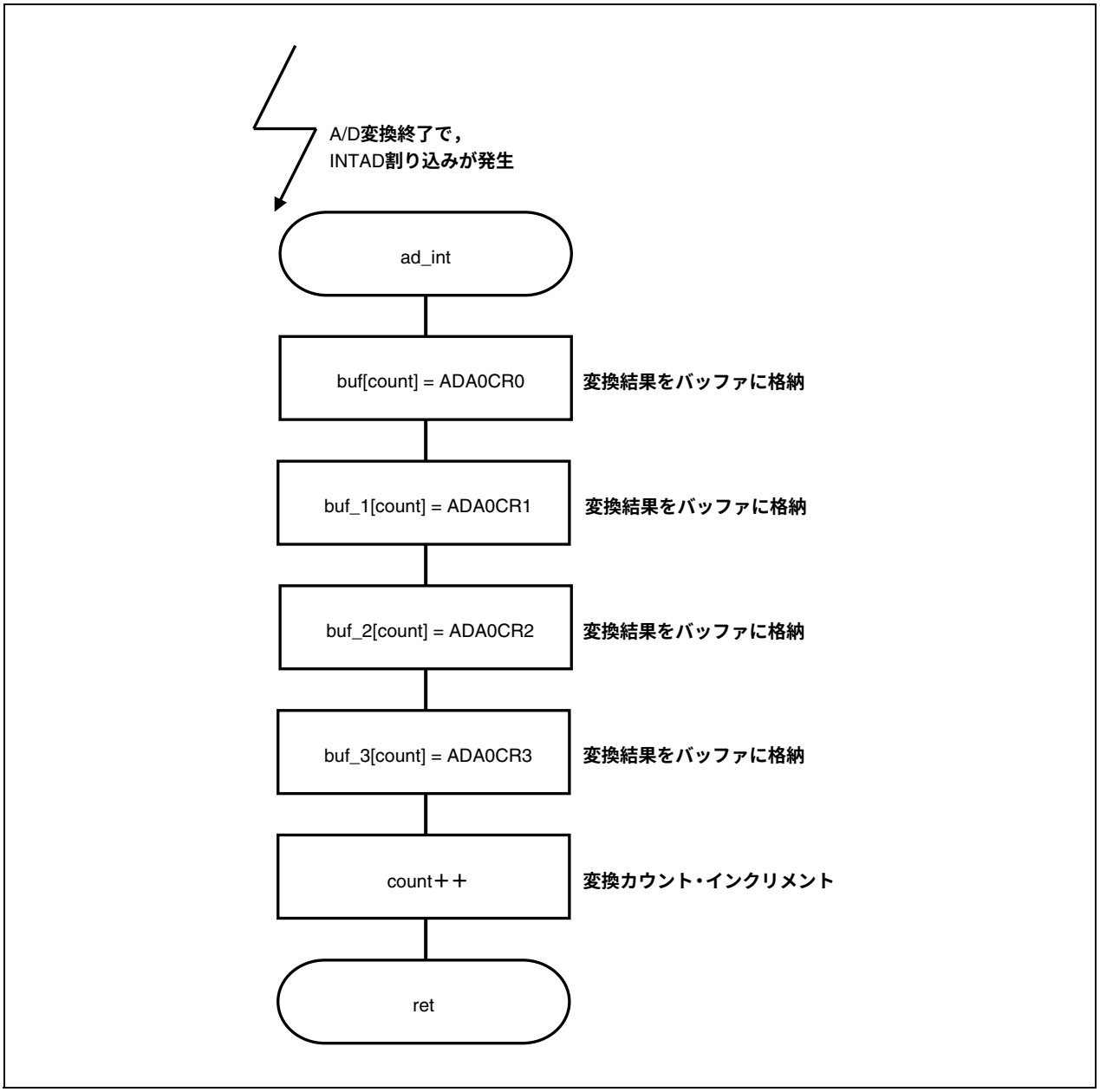


図8－8　ソフトウエア・トリガ・モード（ワンショット・スキャン・モード）（3）



8.4　タイマ・トリガ・モード（ワンショット・セレクト・モード）

【機　　能】	A/D 変換動作開始タイミングをハードウエア・トリガ・モードのタイマ・トリガ・モード 0 に設定し，A/D 変換を行います。
【関　数　名】	ad_timer_main
【引　き　数】	なし
【処　理　内　容】	タイマのコンペア一致割り込み要求信号（INTTP2CC0）が入力されると，A/D 変換を開始します。ANI00 端子を A/D 変換し，A/D 変換結果を buf[ ]に格納します。 1 回の A/D 変換終了ごとに A/D 変換終了割り込み要求信号（INTAD）が発生します。 A/D 変換は 10 回行います。
【使 用 S F R】	ADMK　：0（A/D 変換終了割り込み要求信号（INTAD）クリア， マスク解除，優先レベル 7 に設定）
【call　関数】	Ad_port_set, ad_set, ad_start, ad_timer_trigger, ad_stop
【変　　数】	unsigned short int buf [ ]　：変換データ格納バッファ volatile unsigned char count　：変換カウント変数
【割　り　込　み】	ad_int
【割り込み要因】	INTAD
【ファイル名】	ad_timer_trigger¥ad_timer_trigger.c
【注　意　事　項】	タイマのコンペア一致割り込み要求信号（INTTP2CC0）発生方法は第 3 章　16 ビット・タイマ／イベント・カウンタ P（TMP）を参照してください。

【関　数　名】	ad_port_set
【処　理　内　容】	アナログ入力端子の設定を行います。
【使 用 S F R】	ADA0S　：0x00（アナログ入力端子を ANI0 端子に設定）
【call　関数】	なし
【変　　数】	なし
【ファイル名】	ad_timer_trigger¥ad_timer_trigger.c

【関　数　名】 ad_set

【処　理　内　容】 A/D 変換制御レジスタの設定を行います。

【使 用 S F R】 ADA0M0 ：0x22（ワンショット・セレクト・モード，有効エッジ検出なし，ハードウエア・トリガ・モードに設定）
ADA0M2 ：0x01（タイマ・トリガ・モード 0 に設定）
ADA0M1 ：0x80（高速変換モード，変換クロック数を 6.5 μs に設定）

【call　関数】 なし

【変　数】 なし

【ファイル名】 ad_timer_trigger¥ad_timer_trigger.c

【関　数　名】 ad_start

【処　理　内　容】 A/D 変換動作を許可します。

【使 用 S F R】 ADA0M0.ADA0CE ：1（A/D 変換動作許可）

【call　関数】 なし

【変　数】 なし

【ファイル名】 ad_timer_trigger¥ad_timer_trigger.c

【関　数　名】 ad_stop

【処　理　内　容】 A/D 変換動作を停止します。

【使 用 S F R】 ADA0M0.ADA0CE ：0（A/D 変換動作停止）

【call　関数】 なし

【変　数】 なし

【ファイル名】 ad_timer_trigger¥ad_timer_trigger.c

割り込み関数

【関　数　名】 ad_int

【処　理　内　容】 A/D 変換結果データをバッファに格納します。

【使 用 S F R】 ADA0CR0 A/D 変換結果レジスタ 0

【call　関数】 なし

【変　数】 unsigned short int buf [ ] ：変換データ格納バッファ
volatile unsigned char count ：変換カウント変数

【ファイル名】 ad_timer_trigger¥ad_timer_trigger.c

図8－9　タイマ・トリガ・モード（ワンショット・セレクト・モード）（1）

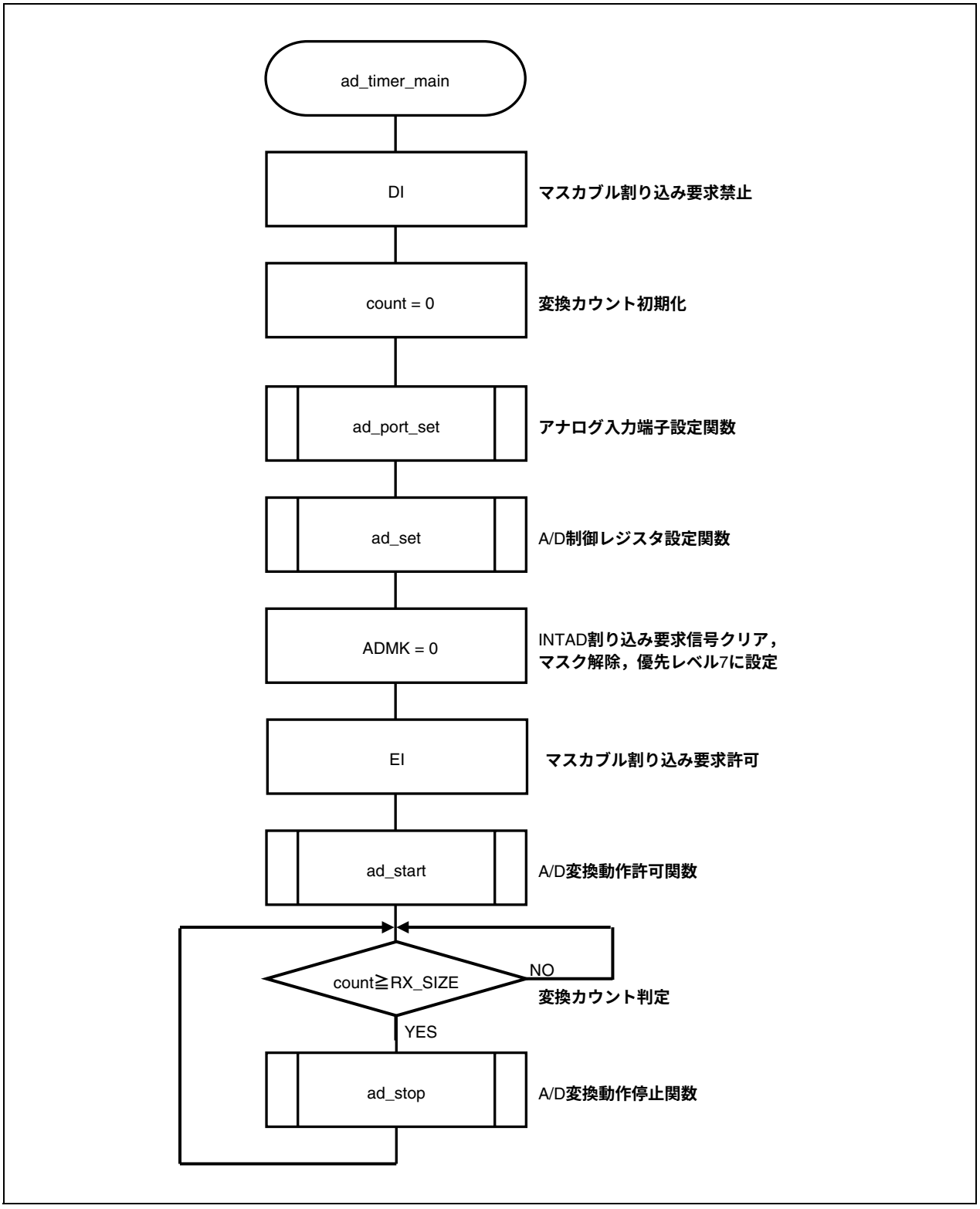
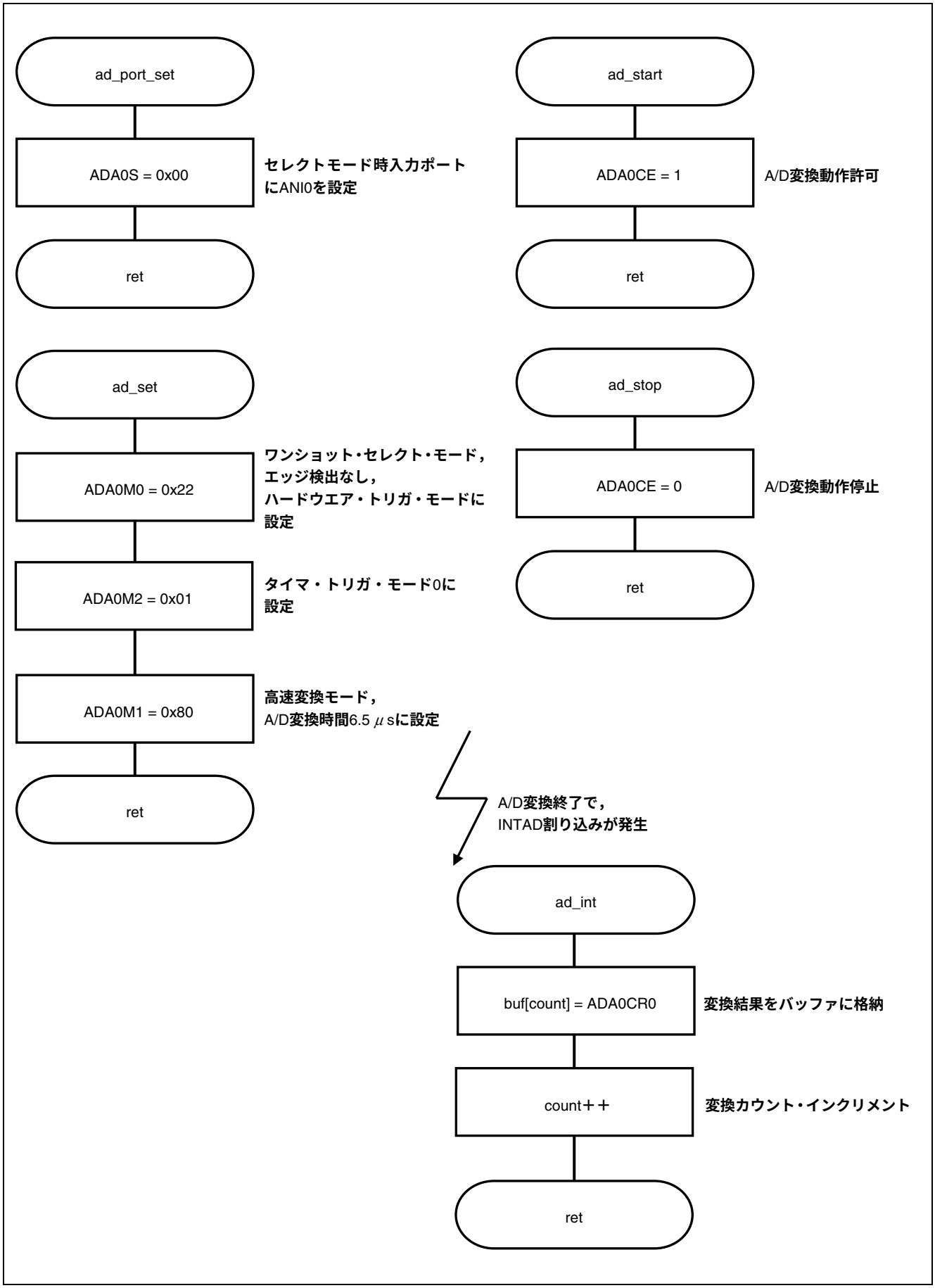


図8－10　タイマ・トリガ・モード（ワンショット・セレクト・モード）（2）



第9章　D/Aコンバータ

9.1　通常モード

【機　　能】	DA0CS0 レジスタへのライト動作を起動トリガとして，D/A 変換を行います。
【関　数　名】	da_normal_main
【引　き　数】	なし
【処　理　内　容】	DA0CS0 レジスタへのライト動作を起動トリガとして，D/A 変換を行い，DA0CS0 レジスタ設定した値を ANO0 端子に出力します。
【使 用 S F R】	なし
【call　　関数】	da_port_set, da_normal_set, da_normal_start,
【変　　数】	なし
【割　り　込　み】	なし
【割り込み要因】	なし
【ファイル名】	da_normal¥da_normal.c
【注　意　事　項】	なし

【関　数　名】	da_port_set	
【処　理　内　容】	ポート 1 を出力端子に設定いします。	
【使 用 S F R】	PM1	：0xFE（ANO0 を出力端子に指定）
【call　　関数】	なし	
【変　　数】	なし	
【ファイル名】	da_normal¥da_normal.c	

【関　数　名】	da_normal_set	
【処　理　内　容】	ANO0 端子に出力するアナログ電圧値を設定します。	
【使 用 S F R】	DA0CS0	：0x80（ANO0 端子に出力するアナログ電圧値を 0x80 に設定）
	DA0M	：0x00（通常モードに設定）
【call　　関数】	なし	
【変　　数】	なし	
【ファイル名】	da_normal¥da_normal.c	

【関　　数　　名】　da_normal_start

【処　理　内　容】　D/A 変換動作を開始します。

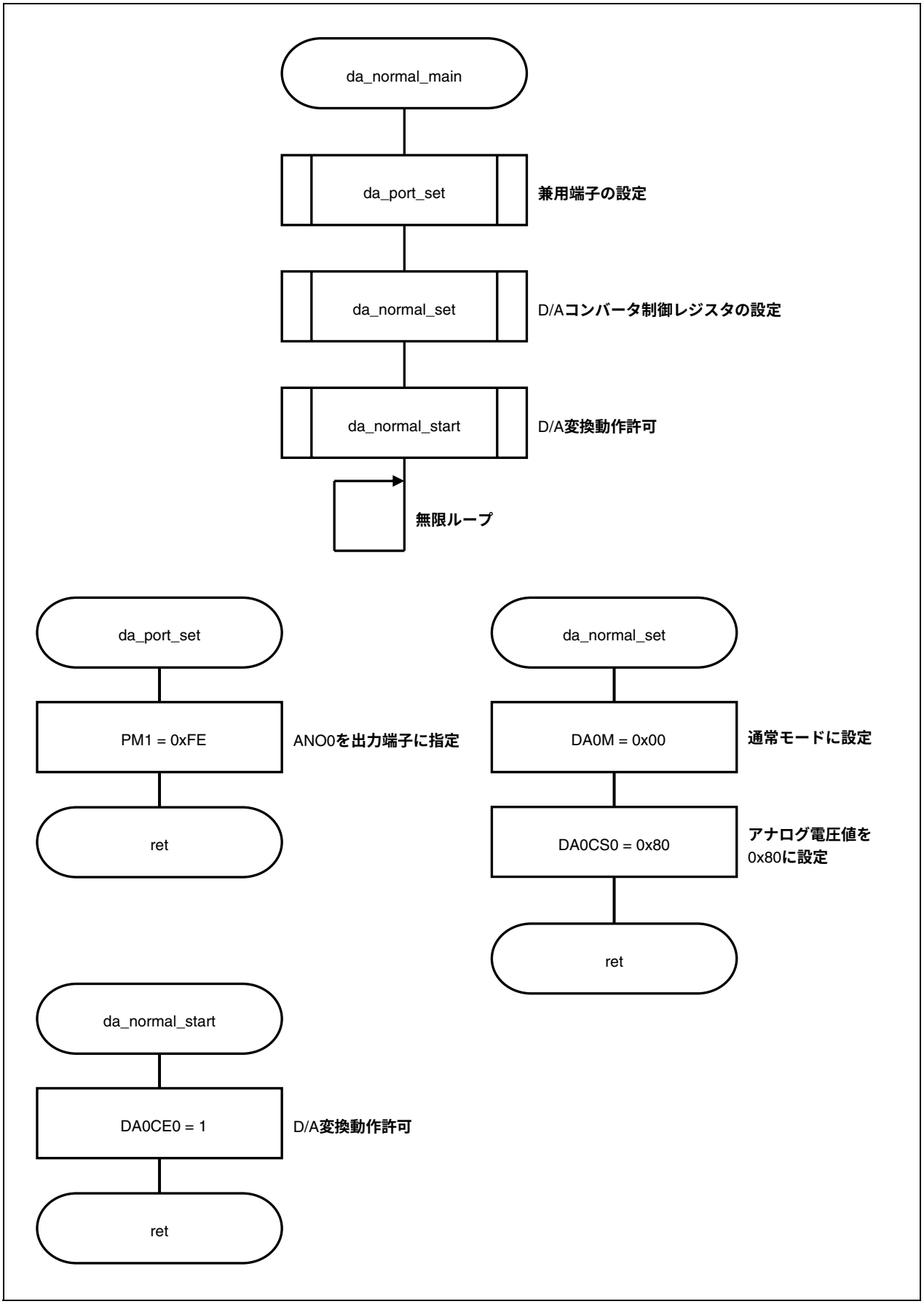
【使 用　S　F　R】　DA0CE0　　：1（D/A 変換動作許可）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　da_normal¥da_normal.c

図9－1　通常モード


da_normal_main
da_port_set
兼用端子の設定
da_normal_set
D/Aコンバータ制御レジスタの設定
da_normal_start
D/A変換動作許可
無限ループ
da_port_set
PM1 = 0xFE
ANO0を出力端子に指定
ret
da_normal_set
DA0M = 0x00
通常モードに設定
DA0CS0 = 0x80
アナログ電圧値を
0x80に設定
ret
da_normal_start
DA0CE0 = 1
D/A変換動作許可
ret

9. 2　リアルタイム出力モード

【機　　能】	TMP2 の割り込み要求信号（INTTP2CC0）を起動トリガとして，D/A 変換を行います。
【関　数　名】	da_realtime_main
【引　き　数】	なし
【処　理　内　容】	DA0CS0 レジスタへのライト動作を起動トリガとして，D/A 変換を行い，DA0CS0 レジスタ設定した値を ANO0 端子に出力します。
【使 用 ＳＦＲ】	なし
【call　　関数】	da_port_set, da_realtime_set, da_realtime_start,
【変　　数】	なし
【割　り　込　み】	なし
【割り込み要因】	なし
【ファイル名】	da_realtime¥da_realtime.c
【注　意　事　項】	タイマのコンペア一致割り込み要求信号（INTTP2CC0）発生方法は第 3 章　16 ビット・タイマ／イベント･カウンタ P（TMP）を参照してください。

【関　数　名】	da_port_set	
【処　理　内　容】	ポート 1 を出力端子に設定いします。	
【使 用 ＳＦＲ】	PM1	：0xFE（ANO0 を出力端子に指定）
【call　　関数】	なし	
【変　　数】	なし	
【ファイル名】	da_realtime¥da_realtime.c	

【関　数　名】	da_ realtime_set	
【処　理　内　容】	ANO0 端子に出力するアナログ電圧値を設定します。	
【使 用 ＳＦＲ】	DA0CS0	：0x80（ANO0 端子に出力するアナログ電圧値を 0x80 に設定）
	DA0M	：0x00（通常モードに設定）
【call　　関数】	なし	
【変　　数】	なし	
【ファイル名】	da_realtime¥da_realtime.c	

【関　　数　　名】　da_ realtime_start

【処　理　内　容】　D/A 変換動作を開始します。

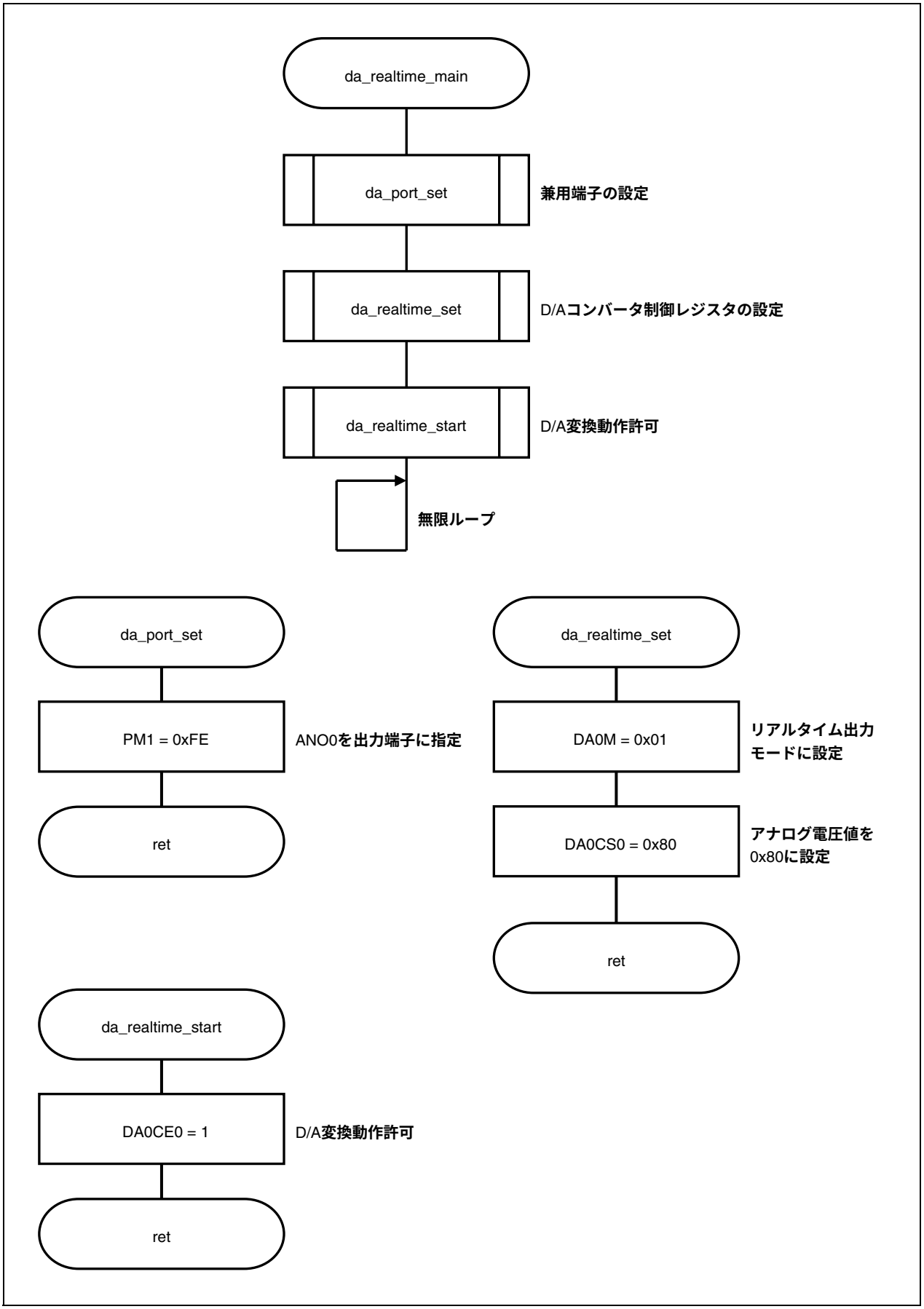
【使 用　S　F　R】　DA0CE0　　：1（D/A 変換動作許可）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　da_realtime¥da_realtime.c

図9－2　リアルタイム出力モード



第10章　アシンクロナス・シリアル・インタフェースA（UARTA）

10. 1　アシンクロナス・シリアル・インタフェースA（UARTAn）（n = 0-2）

【機　　能】	UARTA0 連続送受信
【関　数　名】	uarta_main
【引　き　数】	なし
【処　理　内　容】	UARTA0 を使用し，送受信を 10 回行います。受信したデータは buf_rx[ ]に格納し，送信データは buf_tx[ ]に格納します。
【使 用　S　F　R】	UA0TMK　：0（UARTA0 送信許可割り込み要求信号（INTUA0T）クリア，マスク解除，優先レベル 7 に設定） UA0RMK　：0（UARTA0 受信終了割り込み要求信号（INTUA0R）クリア，マスク解除，優先レベル 7 に設定）
【call　　関数】	uarta_port_set, uarta_set, uarta_start, uarta_receive_end, uarta_send_end, uarta_end
【変　　数】	unsigned char buf_tx[ ]　：送信データ格納バッファ unsigned char buf_rx[ ]　：受信データ格納バッファ volatile unsigned char count_tx　：送信カウント変数 volatile unsigned char count_rx　：受信カウント変数 unsigned char count　：転送データ生成変数
【割　り　込　み】	uarta_int_send, uarta_int_receive
【割り込み要因】	INTUA0T, INTUA0R
【ファイル名】	UARTA¥uarta.c
【注　意　事　項】	なし

【関　数　名】	uarta_port_set
【処　理　内　容】	兼用端子を UARTA0 入出力端子に設定します。
【使 用　S　F　R】	PMC3　：0x03（TXDA0 出力，RXDA0 入力設定）
【call　　関数】	なし
【変　　数】	なし
【ファイル名】	UARTA¥uarta.c
【注　意　事　項】	なし

【関　数　名】 uarta_set

【処　理　内　容】 UARTA0 制御レジスタの設定を行います。ボー・レートは 9600 bps に設定します。

【使 用 S F R】 UA0CTL1 ：0x01（ボー・レートを 9600 bps に設定）
UA0CTL2 ：0x82（ボー・レートを 9600 bps に設定）
UA0OPT0 ：0x14（転送データ通常出力，転送データ通常入力に設定）
UA0CTL0 ：0x9A（UARTA0 動作許可，LSB ファースト，奇数パリティを出力データ，キャラクタ長 8 ビット，ストップ・ビット長 1 ビットに設定）

【call　関数】 なし

【変　数】 なし

【ファイル名】 UARTA¥uarta.c

【注　意　事　項】 なし

【関　数　名】 uarta_start

【処　理　内　容】 送受信を許可し，UA0TX レジスタにデータを書き込みます。

【使 用 S F R】 UA0CTL0.UA0TXE ：1（送信動作許可）
UA0CTL0.UA0RXE ：1（受信動作許可）
UA0TX 送信データ・レジスタ

【call　関数】 なし

【変　数】 unsigned char buf_tx[ ] ：送信データ格納バッファ

【ファイル名】 UARTA¥uarta.c

【注　意　事　項】 UA0CTL0.UA0PWR ビット = 1 にしてから UA0RXE ビット = 1，UA0TXE ビット = 1 にしてください。

【関　数　名】 uarta_receive_end

【処　理　内　容】 受信動作禁止にします。

【使 用 S F R】 UA0CTL0.UA0RXE ：0（受信動作禁止）

【call　関数】 なし

【変　数】 なし

【ファイル名】 UARTA¥uarta.c

【注　意　事　項】 なし

【関　　数　　名】　uarta_send_end

【処　理　内　容】　送信動作禁止にします。

【使 用　S　F　R】　UA0CTL0.UA0TXE　：0（送信動作禁止）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　UARTA¥uarta.c

【注　意　事　項】　なし

【関　　数　　名】　uarta_end

【処　理　内　容】　UARTA0 を動作禁止にします。

【使 用　S　F　R】　UA0CTL0.UA0PWR　：0（UARTA0 動作禁止）

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　UARTA¥uarta.c

【注　意　事　項】　停止時は UA0RXE ビット = 0，UA0TXE ビット = 0 にしてから UA0PWR ビット = 0 にしてください。

割り込み関数

【関　数　名】 uarta_int_send

【処 理 内 容】 送信データを送信データ・レジスタにライトします。
送信回数とカウント数が一致すれば，送信動作を停止します。

【使 用 S F R】 UA0TX　　送信データ・レジスタ

【call　　関数】 なし

【変　　数】 unsigned char buf_tx[ ]　　：送信データ格納バッファ
volatile unsigned char count_tx　：送信カウント変数

【ファイル名】 UARTA¥uarta.c

【注 意 事 項】 なし

【関　数　名】 uarta_int_recive

【処 理 内 容】 受信データ・レジスタの受信データをバッファにに格納します。
受信回数とカウント数が一致すれば，受信動作を停止し，受信動作を禁止します。

【使 用 S F R】 UA0RX　　受信データ・レジスタ

【call　　関数】 なし

【変　　数】 unsigned char buf_rx[ ]　　：受信データ格納バッファ
volatile unsigned char count_rx　：受信カウント変数

【ファイル名】 UARTA¥uarta.c

【注 意 事 項】 なし

図10－1　アシンクロナス・シリアル・インタフェースA（UARTAn）（1）

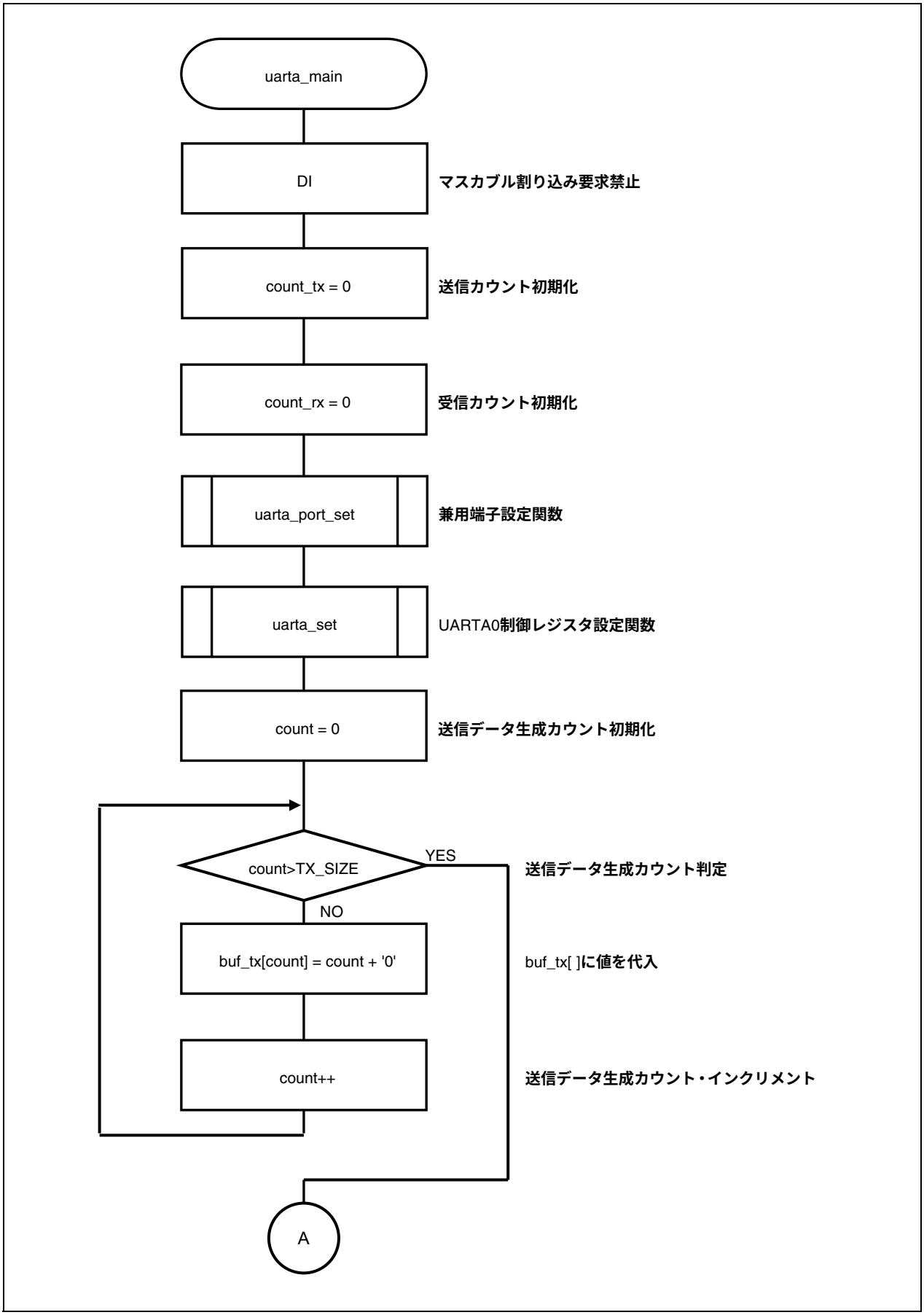


図10－2　アシンクロナス・シリアル・インタフェースA（UARTAn）（2）

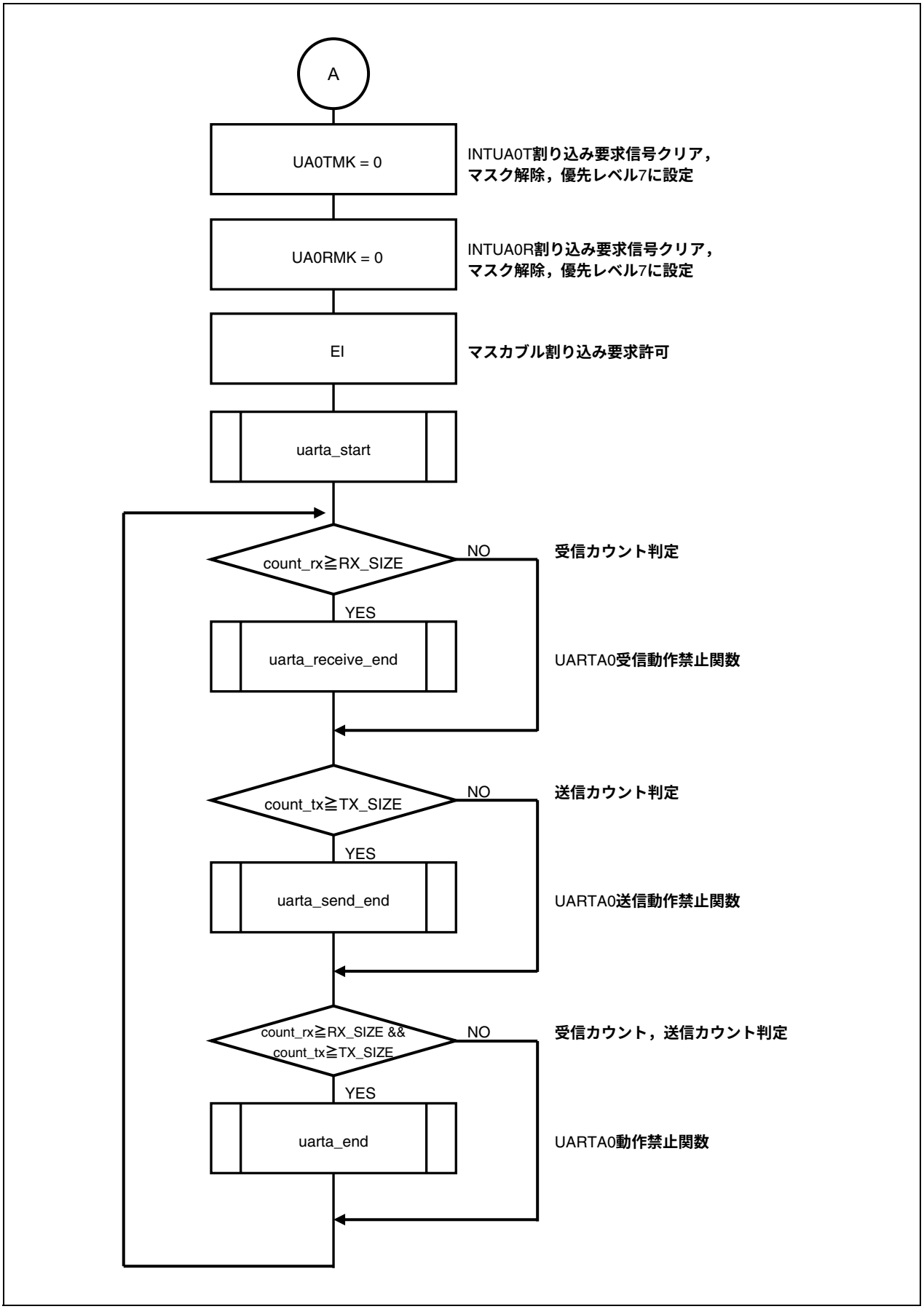


図10－3　アシンクロナス・シリアル・インタフェースA（UARTAn）（3）

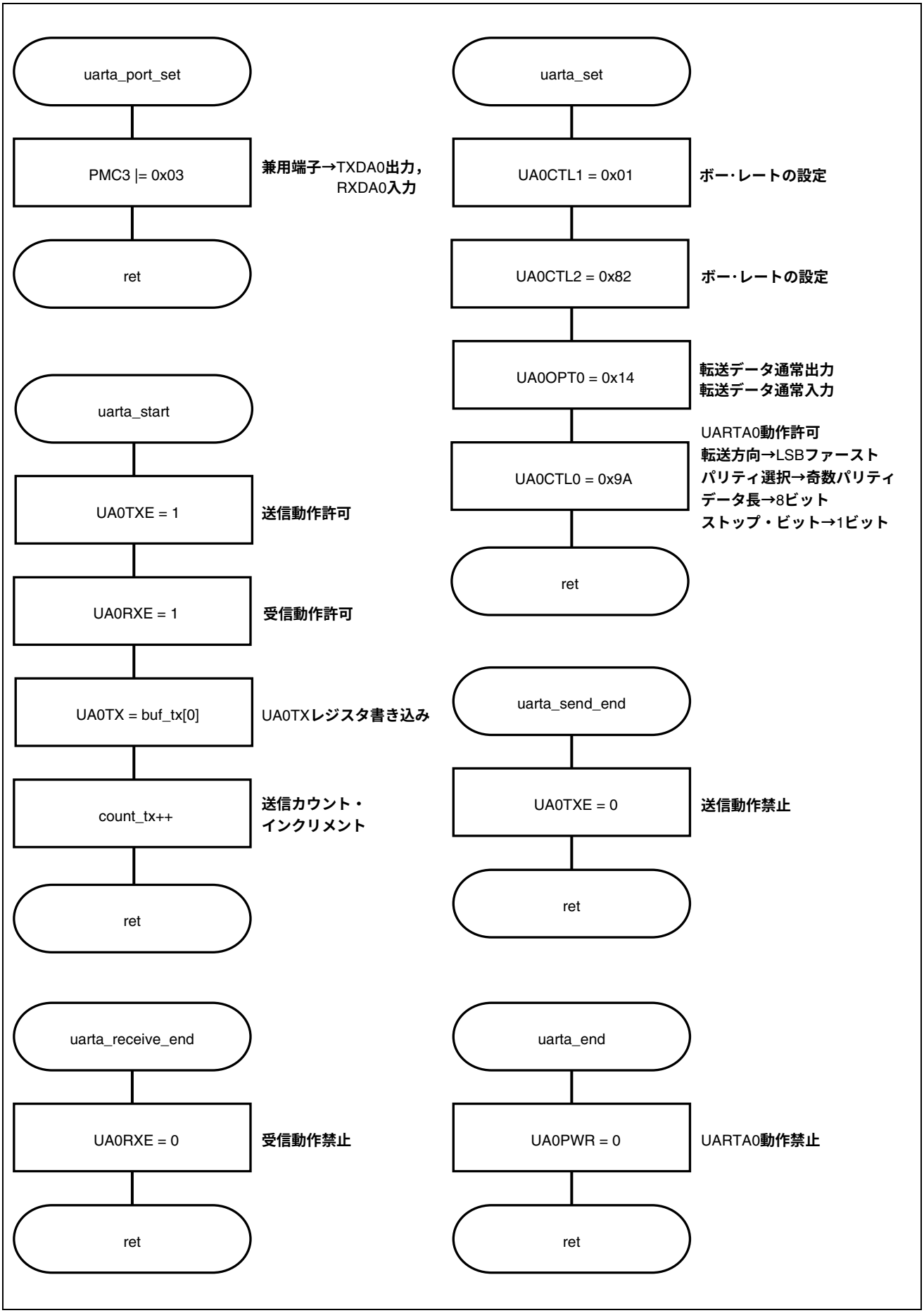
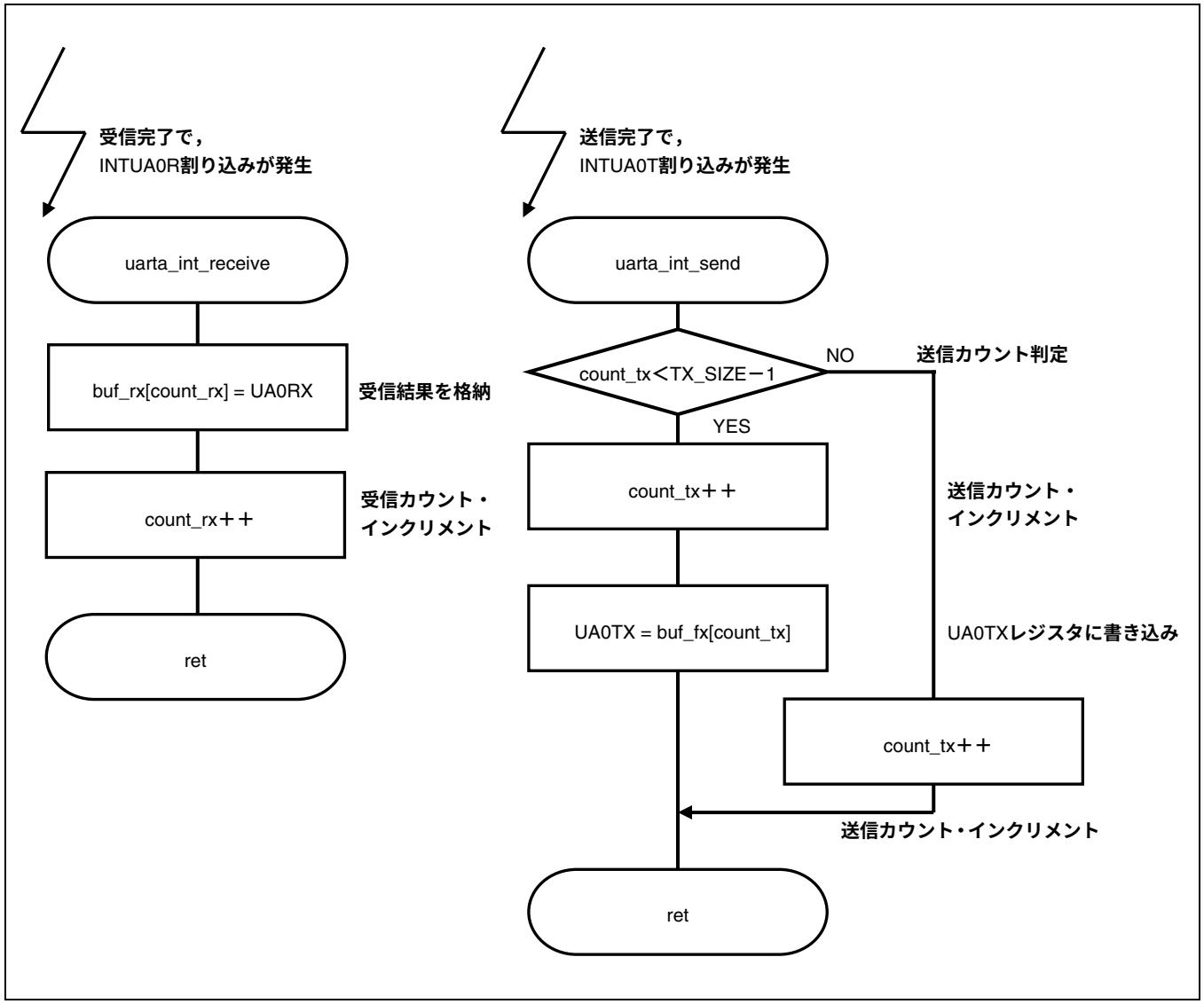


図10－4　アシンクロナス・シリアル・インタフェースA（UARTAn）（4）



第11章　3線式可変長シリアルI/O（CSIB）

11.1　連続転送モード（マスタ・モード，送受信モード）

【機　　能】	連続転送モードで，通信モードをマスタ・モード，転送方向モードを MSB ファーストに設定し，送受信を各 10 回行います。 通信起動トリガを有効にし，通信クロック = fxx/64，転送データ長を 8 ビットに設定します。
【関　数　名】	csib1_main
【引　き　数】	なし
【処　理　内　容】	送信回数カウント（count_tx）を初期値 0 に設定します。受信回数カウント（count_rx）を初期値 0 に設定し，各設定関数を呼び出したあと，送受信を開始します。
【使 用 S F R】	CB0RMK　：0（CSIB0 受信終了割り込み（INTCB0R）処理許可） CB0TMK　：0（CSIB0 送信許可割り込み（INTCB0T）処理許可） CB0STR.CB0TSF　通信状態フラグ
【call　関数】	csib_port, csib_set, csib_start, csib_end
【変　　数】	unsigned char buf_tx[ ]　：送信データ格納バッファ unsigned char buf_rx[ ]　：受信データ格納バッファ volatile unsigned char count_tx　：送信回数カウント変数 volatile unsigned char count_rx　：受信回数カウント変数 unsigned char count　：送信データ生成変数
【割 り 込 み】	csib_int_send, csib_int_receive
【割り込み要因】	INTCB0R, INTCB0T
【ファイル名】	csib1¥csib1.c
【注 意 事 項】	なし

【関　数　名】	csib_port
【処　理　内　容】	ポート 4 を CSIB0 入出力端子に設定します。
【使 用 S F R】	PMC4　：0x07（$\overline{\text{SCKB0}}$ 入出力，SOB0 出力，SIB0 入力設定）
【call　関数】	なし
【変　　数】	なし
【ファイル名】	csib1¥csib1.c
【注 意 事 項】	なし

【関　数　名】　csib_set

【処　理　内　容】　CSIB0 制御レジスタの設定を行います。

【使 用　S　F　R】　CB0CTL1　：0x05（通信タイプ 1，通信クロック = fxx/64 に設定）
CB0CTL2　：0x00（転送データ長を 8 ビットに設定）
CB0CTL0　：0x63（CSIB0 送信動作許可，受信動作許可，MSB ファースト，連続転送モード，通信起動トリガ有効に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　csib1¥csib1.c

【注　意　事　項】　CB0CTL0 レジスタは CB0PWR ビット = 0 時のみ CB0TXE, CB0RXE ビットの書き換え可能です。ただし，同時に CB0PWR ビット = 1 とするのは可能です。

【関　数　名】　csib_start

【処　理　内　容】　CSIB0 動作許可し，送信データ・レジスタに値をライトします。

【使 用　S　F　R】　CB0CTL0.CB0PWR　：1（CSIB0 動作許可）
CB0TX　送信データ・レジスタ

【call　関数】　なし

【変　数】　unsigned char buf_tx[ ]　：送信データ格納バッファ
volatile char count_tx　：送信カウント変数

【ファイル名】　csib1¥csib1.c

【注　意　事　項】　なし

【関　数　名】　csib_end

【処　理　内　容】　CSIB0 動作，送受信動作を禁止します。

【使 用　S　F　R】　CB0CTL0.CB0PWR　：0（CSIB0 動作禁止）
CB0CTL0.CB0RXE　：0（CSIB0 受信動作禁止）
CB0CTL0.CB0TXE　：0（CSIB0 送信動作禁止）

【call　関数】　なし

【変　数】　なし

【ファイル名】　csib1¥csib1.c

【注　意　事　項】　なし

割り込み関数

【関　数　名】	csib_int_send
【処　理　内　容】	次のデータを送信するための新たなデータをセットします。
【使 用　S　F　R】	CB0TX　　送信データ・レジスタ
【call　関数】	なし
【変　数】	unsigned char buf_tx[ ]　：送信データ格納バッファ volatile unsigned char count_tx　：送信カウント変数
【ファイル名】	csib1¥csib1.c
【注　意　事　項】	なし

【関　数　名】	csib_int_receive
【処　理　内　容】	受信したデータをバッファに格納します。
【使 用　S　F　R】	CB0RX　　受信データ・レジスタ
【call　関数】	なし
【変　数】	unsigned char buf_rx[ ]　：受信データ格納バッファ volatile unsigned char count_rx　：受信カウント変数
【ファイル名】	csib1¥csib1.c
【注　意　事　項】	なし

図11－1　連続転送モード（マスタ・モード，送受信モード）（1）

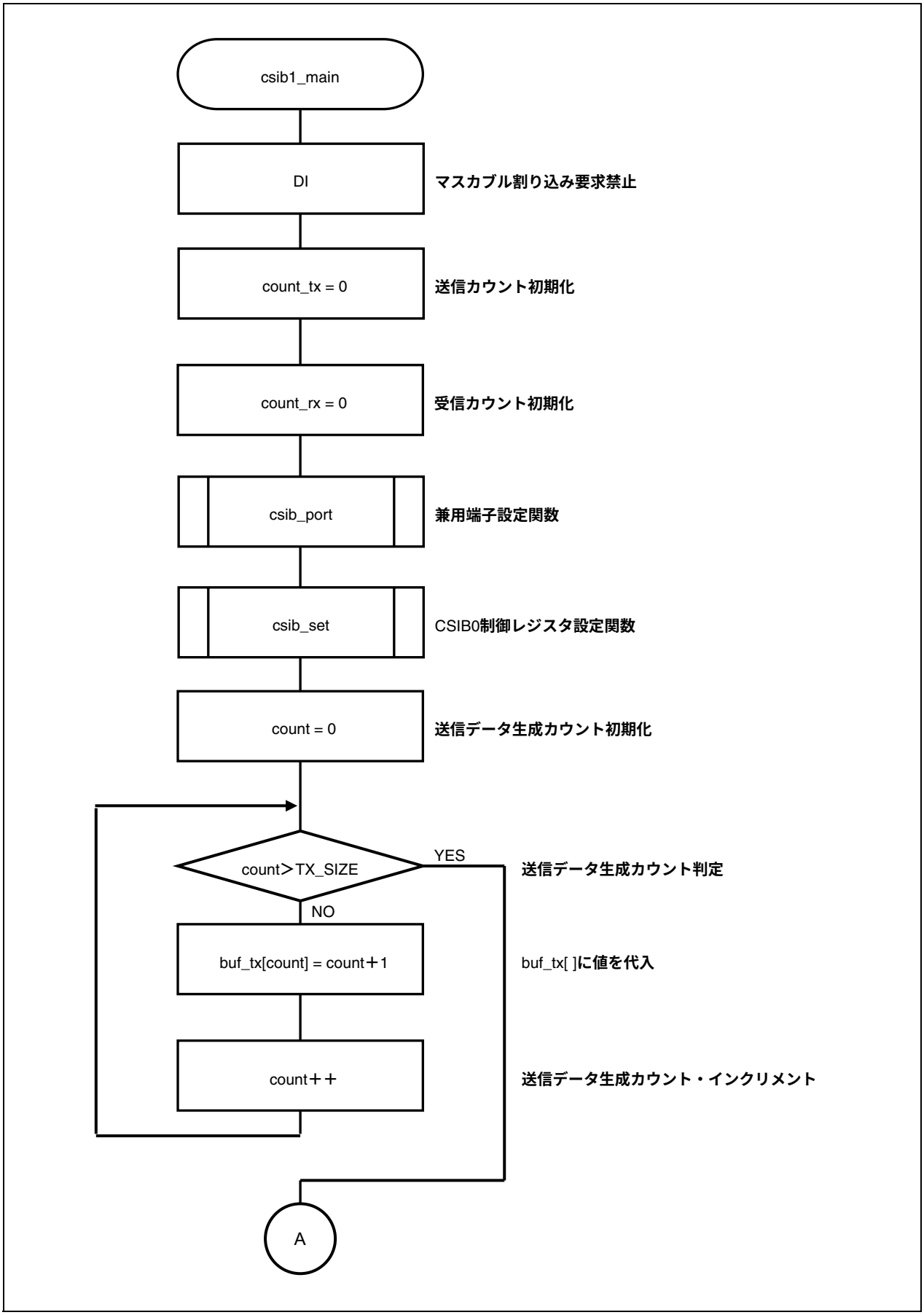


図11－2　連続転送モード（マスタ・モード，送受信モード）（2）

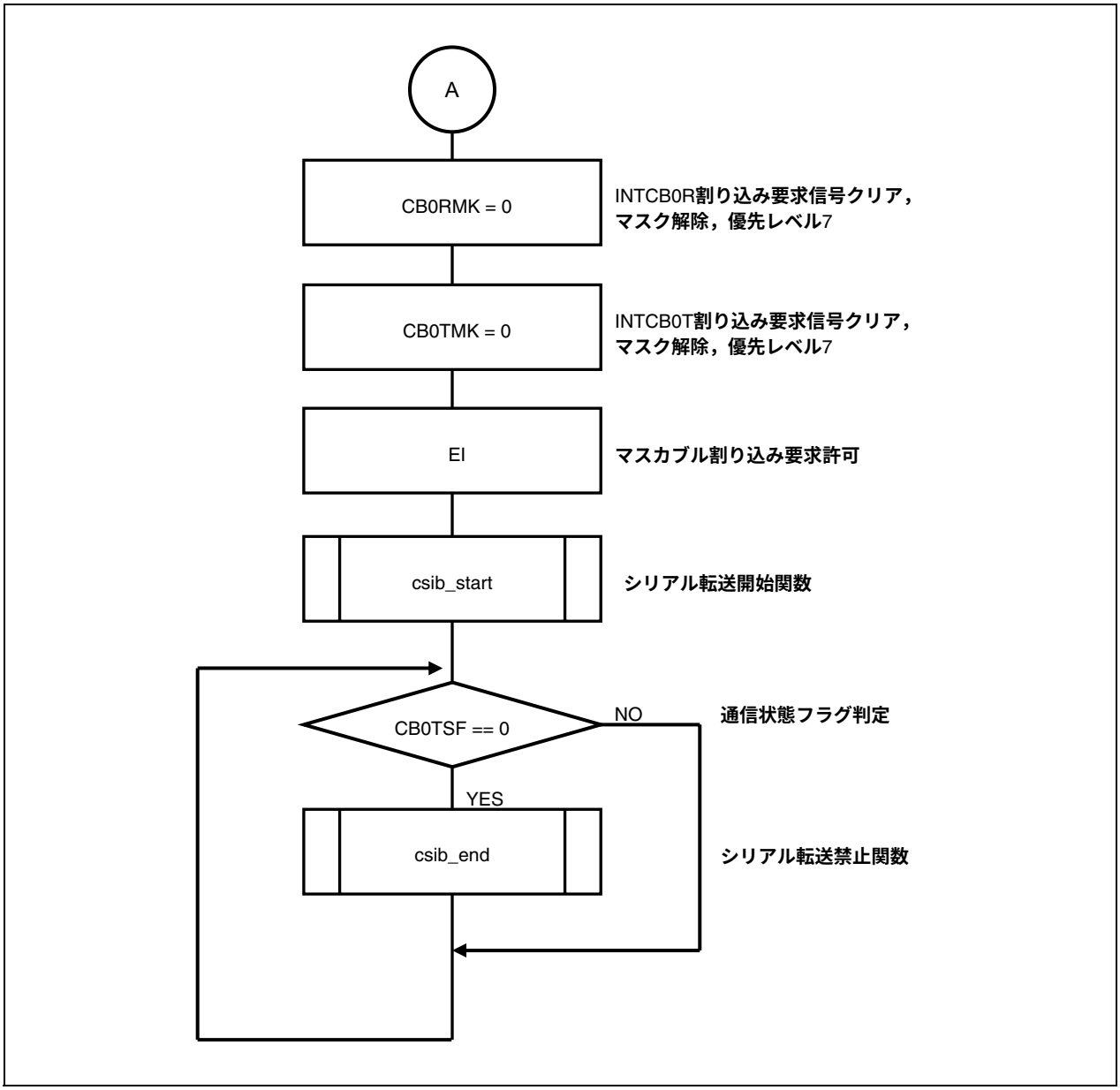


図11－3　連続転送モード（マスタ・モード，送受信モード）（3）

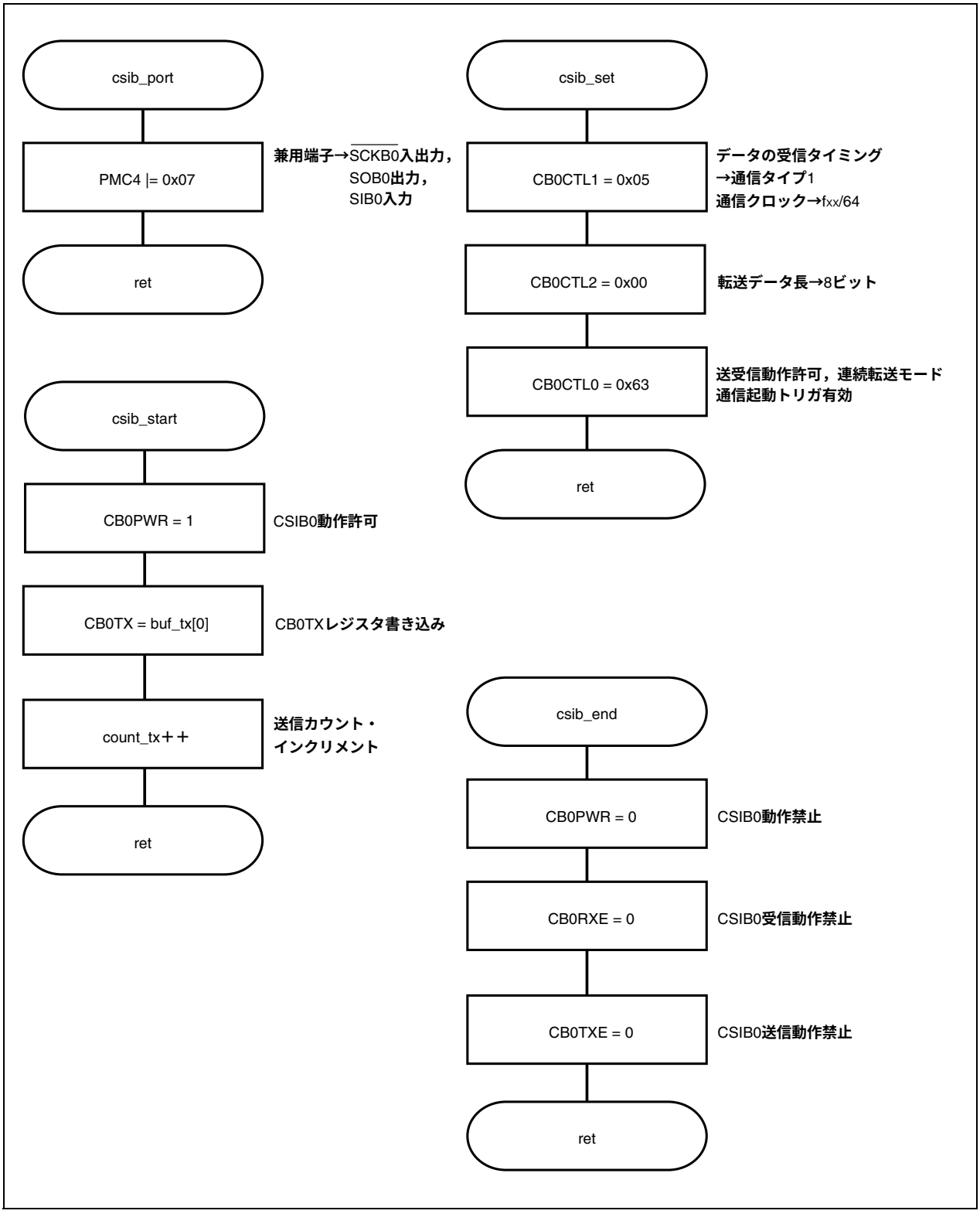
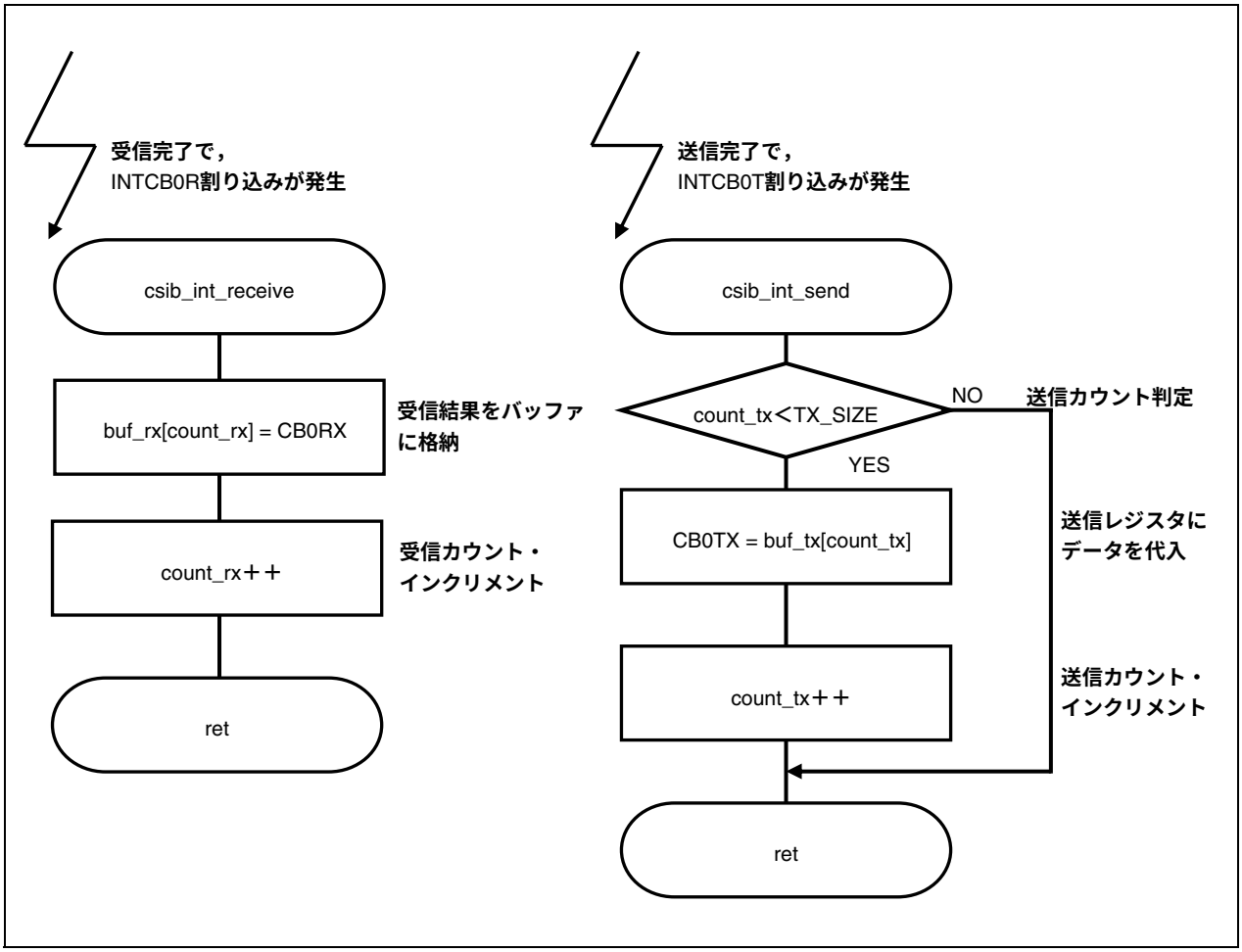


図11－4　連続転送モード（マスタ・モード，送受信モード）（4）



11.2　連続転送モード（スレーブ・モード，送受信モード）

【機　　能】	連続転送モードで，通信モードをスレーブ・モード，転送方向モードを MSB ファーストに設定し，送受信を各 10 回ずつ行います。 通信起動トリガを有効にし，通信クロックを外部クロックに，転送データ長を 8 ビットに設定します。
【関　数　名】	csib2_main
【引　き　数】	なし
【処　理　内　容】	送信回数カウント（count_tx）を初期値 0 に設定します。受信回数カウント（count_rx）を初期値 0 に設定し，各設定関数を呼び出したあと，送受信を開始します。
【使 用 S F R】	CB0RMK　：0（CSIB0 受信終了割り込み（INTCB0R）処理許可） CB0TMK　：0（CSIB0 送信許可割り込み（INTCB0T）処理許可） CB0STR.CB0TSF　通信状態フラグ
【call　関数】	csib_port, csib_set, csib_start, csib_end
【変　　数】	unsigned char buf_tx[ ]　：送信データ格納バッファ unsigned char buf_rx[ ]　：受信データ格納バッファ volatile unsigned char count_tx　：送信カウント変数 volatile unsigned char count_rx　：受信カウント変数 unsigned char count　：送信データ生成変数
【割 り 込 み】	csib_int_send, csib_int_receive
【割り込み要因】	INTCB0T, INTCB0R
【ファイル名】	csib2¥csib2.c
【注 意 事 項】	なし

【関　数　名】	csib_port
【処　理　内　容】	ポート 4 を CSIB0 入出力端子に設定します。
【使 用 S F R】	PMC4　：0x07（$\overline{\text{SCKB0}}$ 入出力，SOB0 出力，SIB0 入力設定）
【call　関数】	なし
【変　　数】	なし
【ファイル名】	csib2¥csib2.c
【注 意 事 項】	なし

【関　数　名】　csib_set

【処　理　内　容】　CSIB0 制御レジスタの設定を行います。

【使 用 S F R】　CB0CTL1　：0x07（通信タイプ 1，通信クロックを外部クロックに設定）
CB0CTL2　：0x00（転送データ長を 8 ビットに設定）
CB0CTL0　：0x63（CSIB0 送信動作許可，受信動作許可，MSB ファースト，連続転送モード，通信起動トリガ有効に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　csib2¥csib2.c

【注　意　事　項】　CB0CTL0 レジスタは CB0PWR ビット = 0 時のみ CB0TXE, CB0RXE ビットの書き換え可能です。ただし，同時に CB0PWR ビット = 1 とするのは可能です。

【関　数　名】　csib_start

【処　理　内　容】　CSIB0 動作許可し，送信データ・レジスタに値をライトします。

【使 用 S F R】　CB0CTL0.CB0PWR　：1（CSIB0 動作許可）
CB0TX　送信データ・レジスタ

【call　関数】　なし

【変　数】　unsigned char buf_tx[ ]　：送信データ格納バッファ
volatile unsigned char count_tx　：送信カウント変数

【ファイル名】　csib2¥csib2.c

【注　意　事　項】　なし

【関　数　名】　csib_end

【処　理　内　容】　CSIB0 動作，送受信動作を禁止します。

【使 用 S F R】　CB0CTL0.CB0PWR　：0（CSIB0 動作禁止）
CB0CTL0.CS0RXE　：0（CSIB0 受信動作禁止）
CB0CTL0.CB0TXE　：0（CSIB0 送信動作禁止）

【call　関数】　なし

【変　数】　なし

【ファイル名】　csib2¥csib2.c

【注　意　事　項】　なし

割り込み関数

【関　数　名】	csib_int_send
【処　理　内　容】	次のデータを送信するための新たなデータをセットします。
【使 用 S F R】	CB0TX　　送信データ・レジスタ
【call　関数】	なし
【変　数】	unsigned char buf_tx[ ]　：送信データ格納バッファ volatile unsigned char count_tx　：送信カウント変数
【ファイル名】	csib2¥csib2.c
【注　意　事　項】	なし

【関　数　名】	csib_int_receive
【処　理　内　容】	受信したデータをバッファに格納します。
【使 用 S F R】	CB0RX　　受信データ・レジスタ
【call　関数】	なし
【変　数】	unsigned char buf_rx[ ]　：受信データ格納バッファ volatile unsigned char count_rx　：受信カウント変数
【ファイル名】	csib2¥csib2.c
【注　意　事　項】	なし

図11－5　連続転送モード（スレーブ・モード，送受信モード）（1）

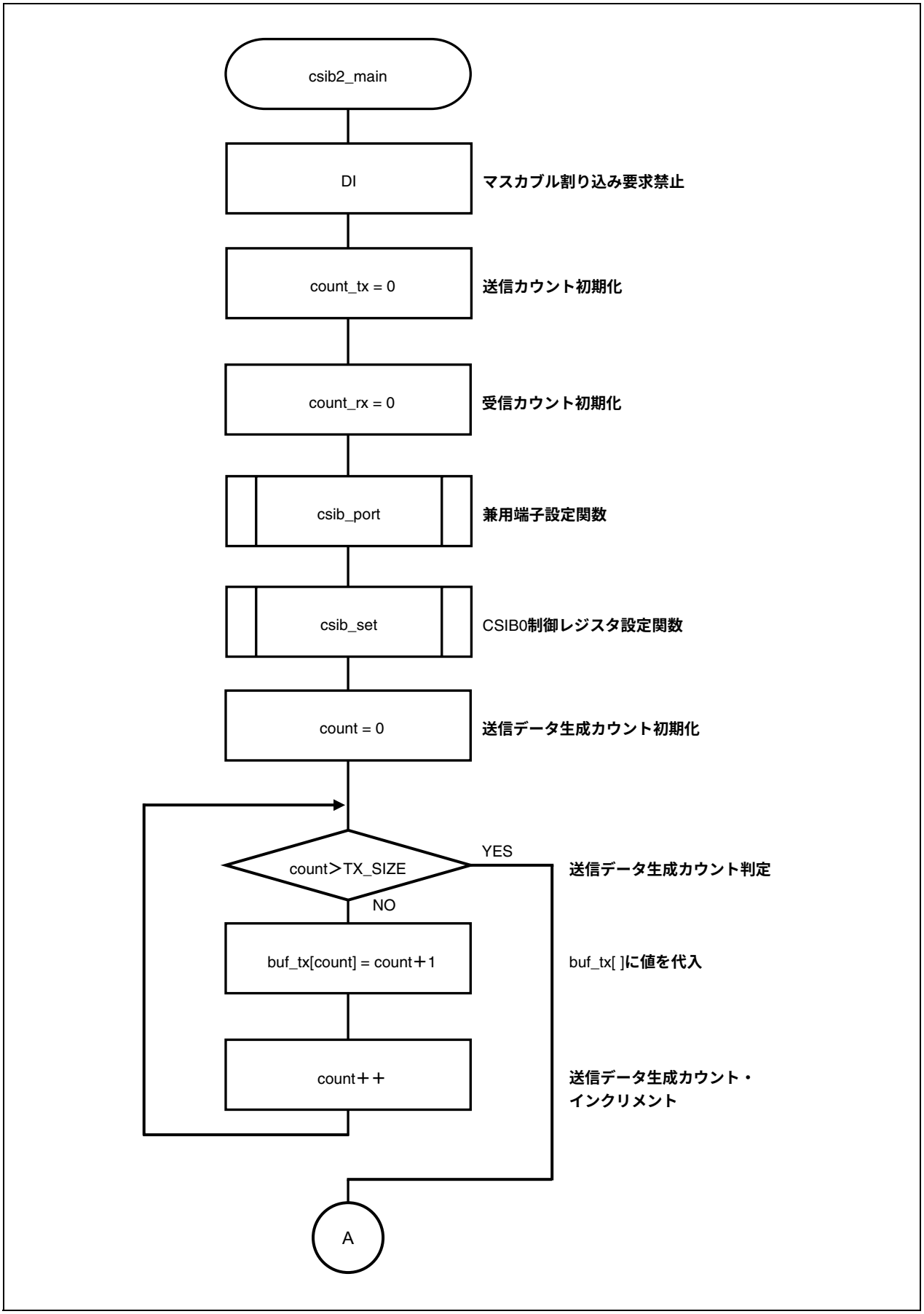


図11－6　連続転送モード（スレーブ・モード，送受信モード）（2）

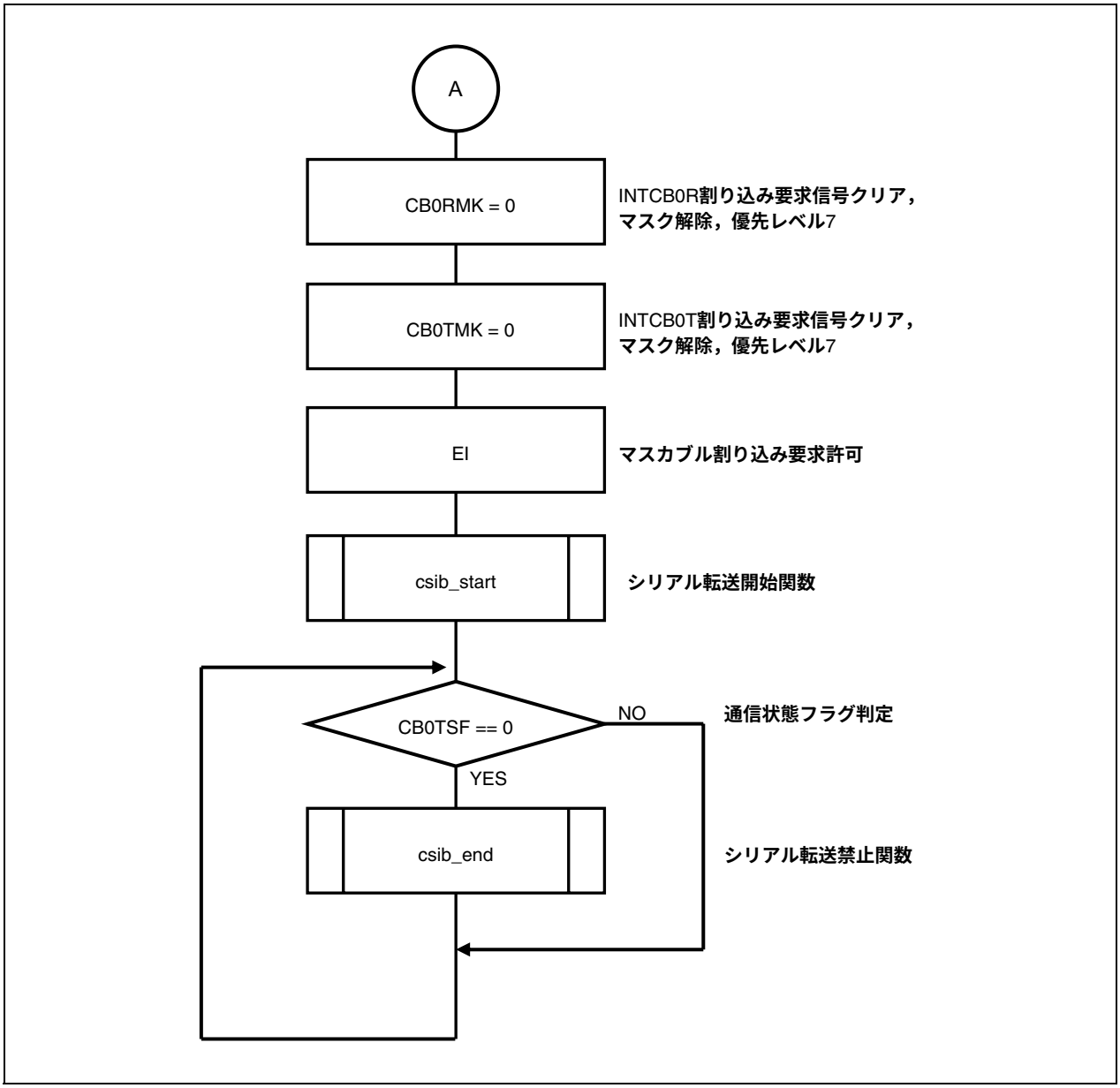


図11－7　連続転送モード（スレーブ・モード，送受信モード）（3）

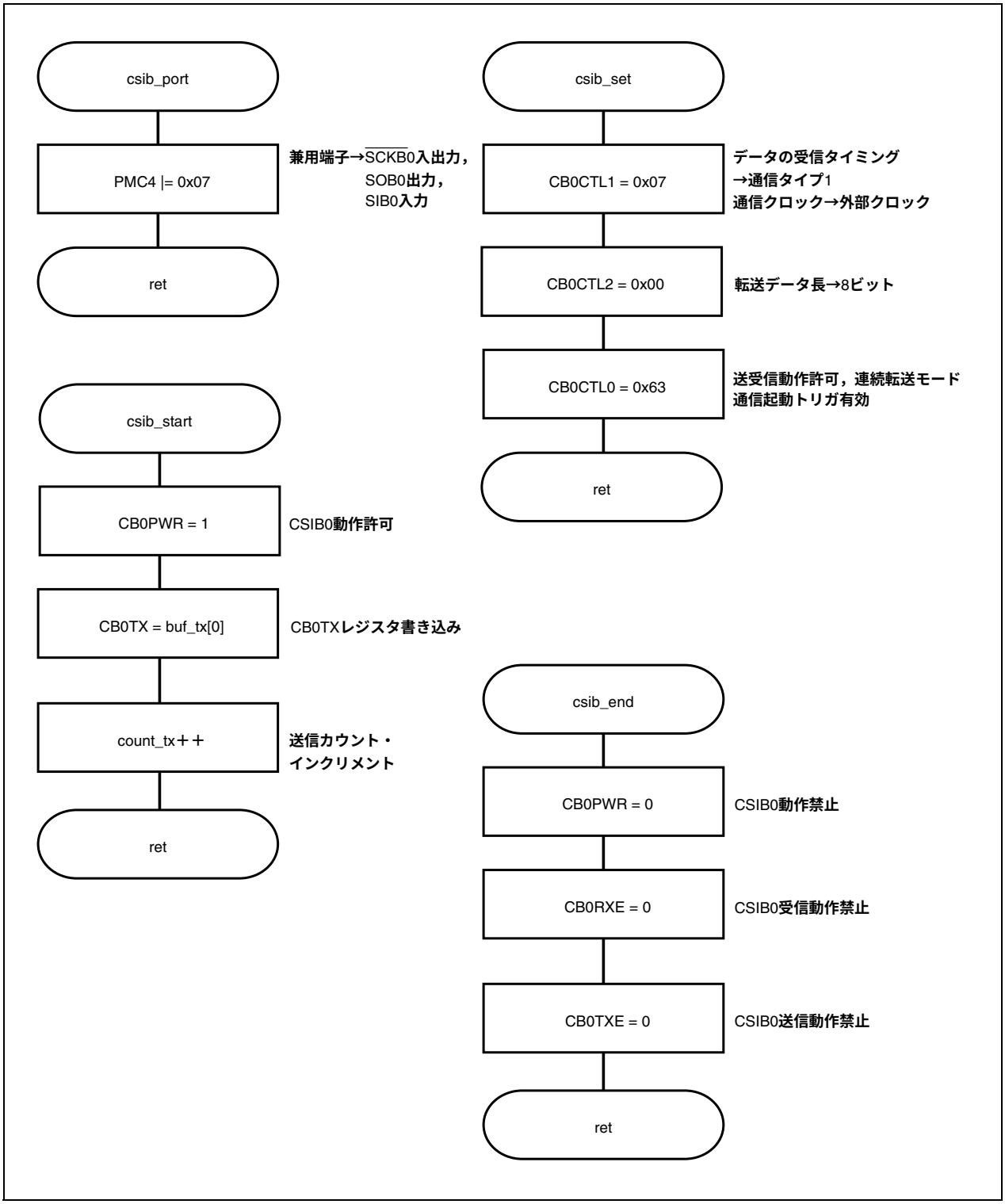
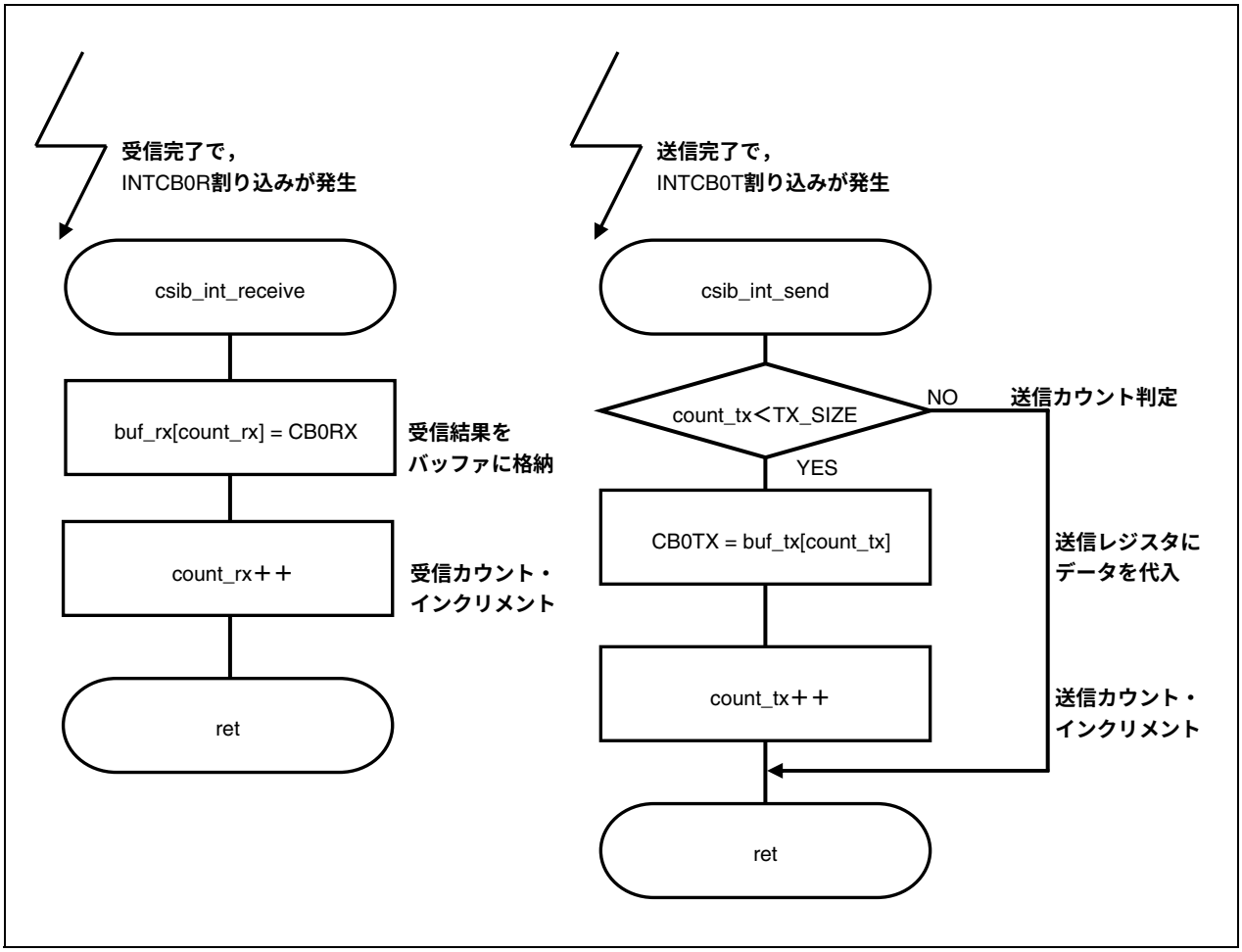


図11－8　連続転送モード（スレーブ・モード，送受信モード）（4）



11.3　シングル転送モード（マスタ・モード，送信モード）

【機　　能】	シングル転送モードで，通信モードをマスタ・モード，転送方向モードを MSB ファーストに設定し，10 回のデータ送信を行います。 通信起動トリガを有効にし，通信クロック = fxx/64，転送データ長を 8 ビットに設定します。
【関　数　名】	csib３_main
【引　き　数】	なし
【処 理 内 容】	送信回数カウント（count_tx）を初期値 0 に設定します。各設定関数を呼び出したあと，送信を開始します。
【使 用 S F R】	CB0RMK　：0（CSIB0 受信終了割り込み要求信号（INTCB0R）クリア，マスク解除，優先レベル 7 に設定）
【call　関数】	csib_port, csib_set, csib_start, csib_end
【変　　数】	unsigned char buf_tx[ ]　：送信データ格納バッファ volatile unsigned char count_tx　：送信カウント変数 unsigned char count　：送信データ生成変数
【割 り 込 み】	csib_int_send
【割り込み要因】	INTCB0R
【ファイル名】	csib3¥csib3.c
【注 意 事 項】	なし

【関　数　名】	csib_port
【処 理 内 容】	ポート 4 を CSIB0 入出力端子に設定します。
【使 用 S F R】	PMC4　：0x06（$\overline{\text{SCKB0}}$ 入出力，SOB0 出力設定）
【call　関数】	なし
【変　　数】	なし
【ファイル名】	csib3¥csib3.c
【注 意 事 項】	なし

【関　数　名】　csib_set

【処　理　内　容】　CSIB0 制御レジスタの設定を行います。

【使 用 S F R】　CB0CTL1　：0x05（通信タイプ 1，通信クロック = fxx/64 に設定）
CB0CTL2　：0x00（転送データ長を 8 ビットに設定）
CB0CTL0　：0x41（CSIB0 送信動作許可，MSB ファースト，シングル転送モード，通信起動トリガ有効に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　csib3¥csib3.c

【注　意　事　項】　CB0CTL0 レジスタは CB0PWR ビット = 0 時のみ CB0TXE, CB0RXE ビットの書き換え可能です。ただし，同時に CB0PWR ビット = 1 とするのは可能です。

【関　数　名】　csib_start

【処　理　内　容】　CSIB0 動作許可し，送信データ・レジスタに値をライトします。

【使 用 S F R】　CB0CTL0.CB0PWR　：1（CSIB0 動作許可）
CB0TX　送信データ・レジスタ

【call　関数】　なし

【変　数】　unsigned char buf_tx[ ]　：送信データ格納バッファ
volatile unsigned char count_tx　：送信カウント変数

【ファイル名】　csib3¥csib3.c

【注　意　事　項】　なし

【関　数　名】　csib_end

【処　理　内　容】　CSIB0 動作，送信動作を禁止します。

【使 用 S F R】　CB0CTL0.CB0PWR　：0（CSIB0 動作禁止）
CB0CTL0.CB0TXE　：0（CSIB0 送信動作禁止）

【call　関数】　なし

【変　数】　なし

【ファイル名】　csib3¥csib3.c

【注　意　事　項】　なし

割り込み関数

【関　数　名】	csib_int_send
【処　理　内　容】	次のデータを送信するための新たなデータをセットします。
【使 用 S F R】	CB0TX　　送信データ・レジスタ
【call　関数】	なし
【変　数】	unsigned char buf_tx[ ]　：送信データ格納バッファ volatile unsigned char count_tx　：送信カウント変数
【フ ァ イ ル 名】	csib3¥csib3.c
【注　意　事　項】	なし

図11－9　シングル転送モード（マスタ・モード，送信モード）（1）

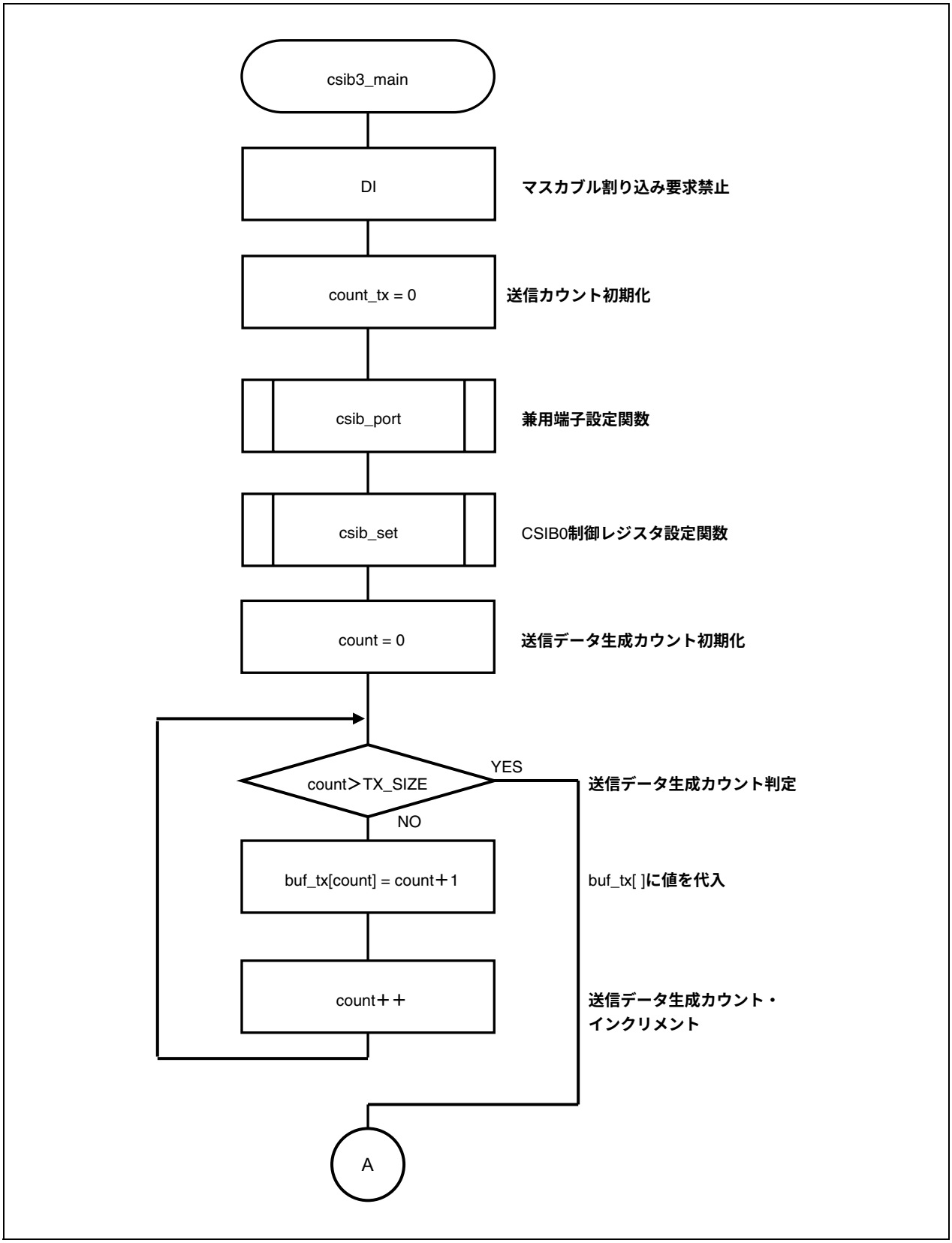


図11－10　シングル転送モード（マスタ・モード，送信モード）（2）

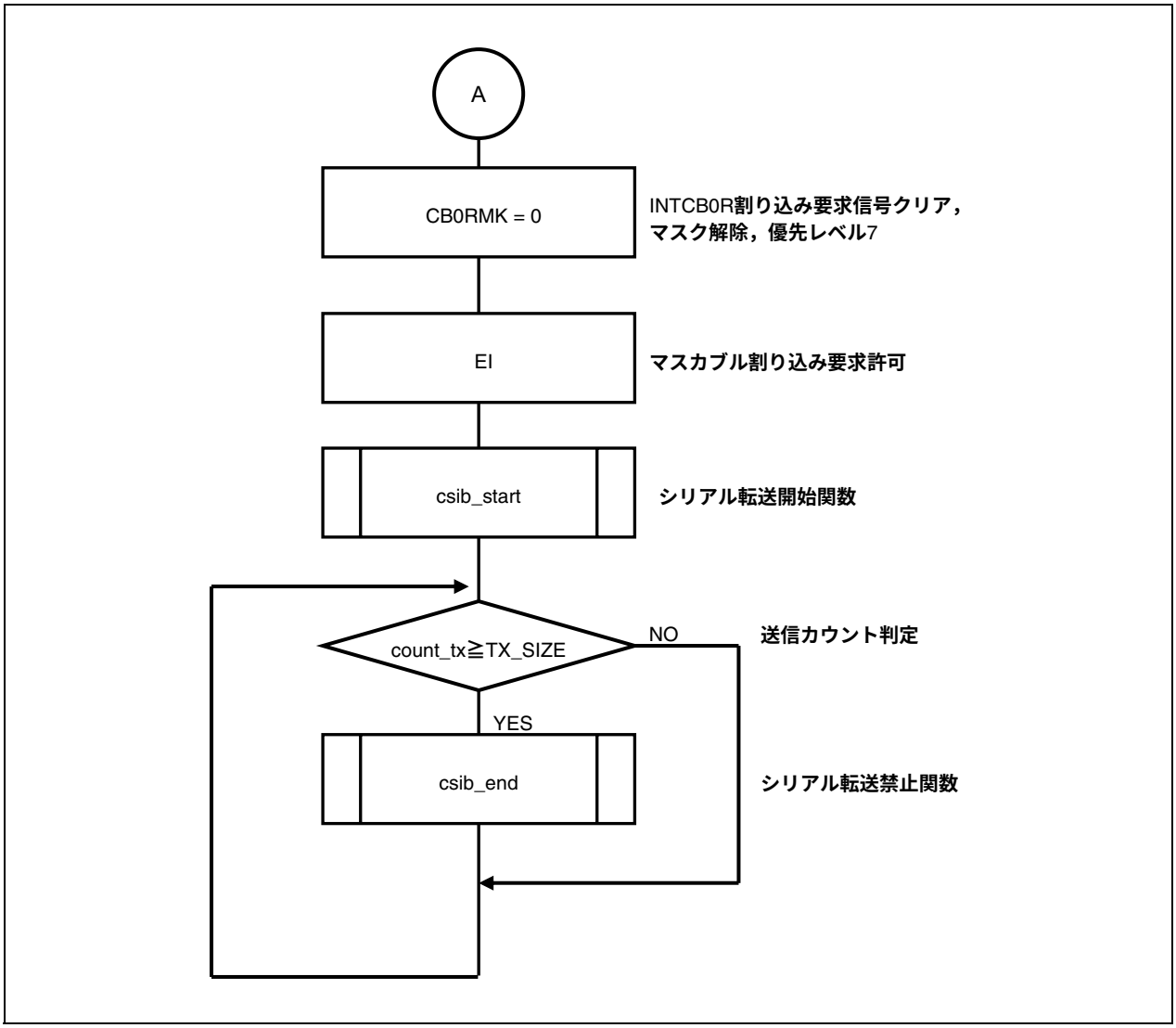


図11－11　シングル転送モード（マスタ・モード，送信モード）（3）

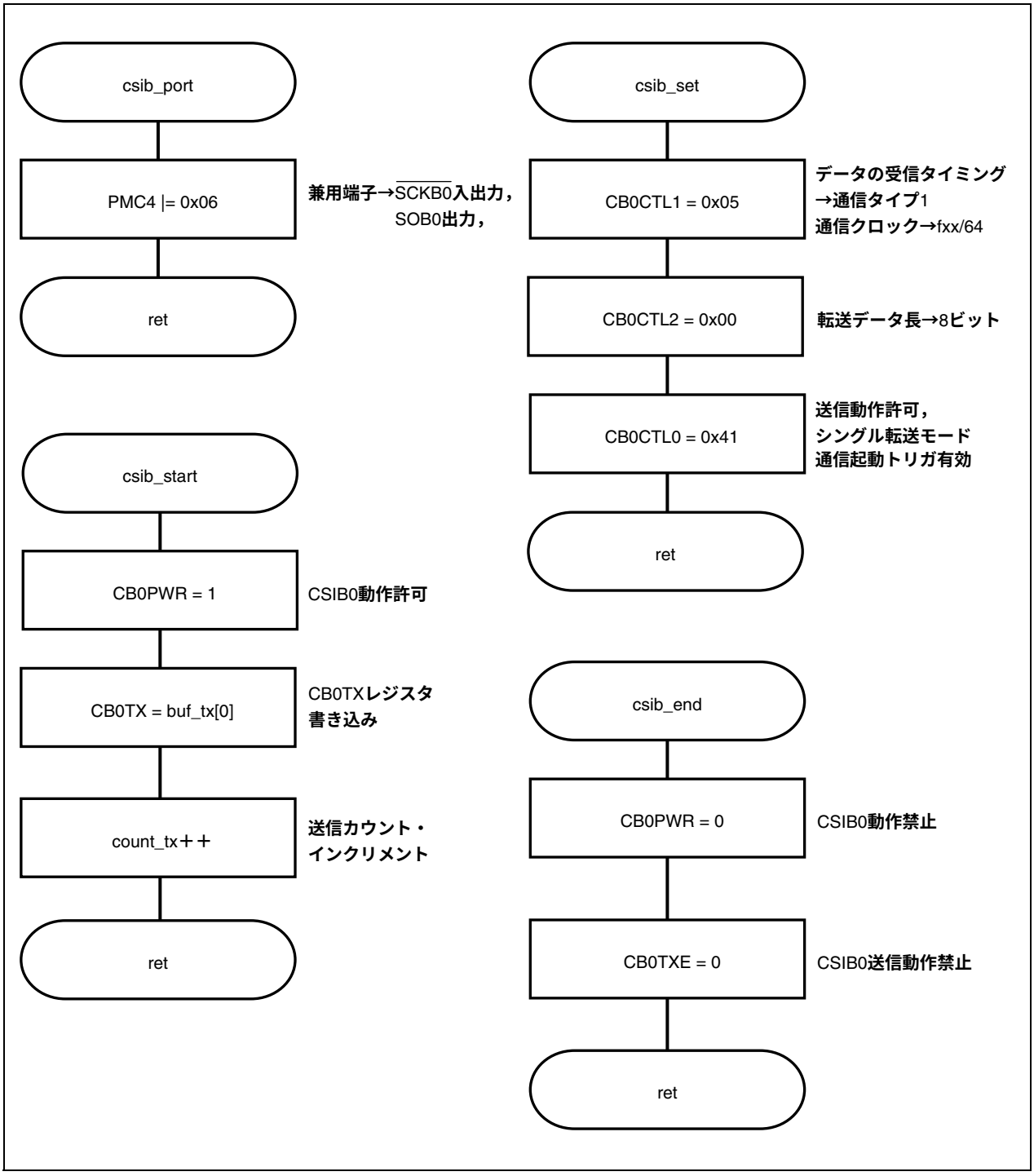
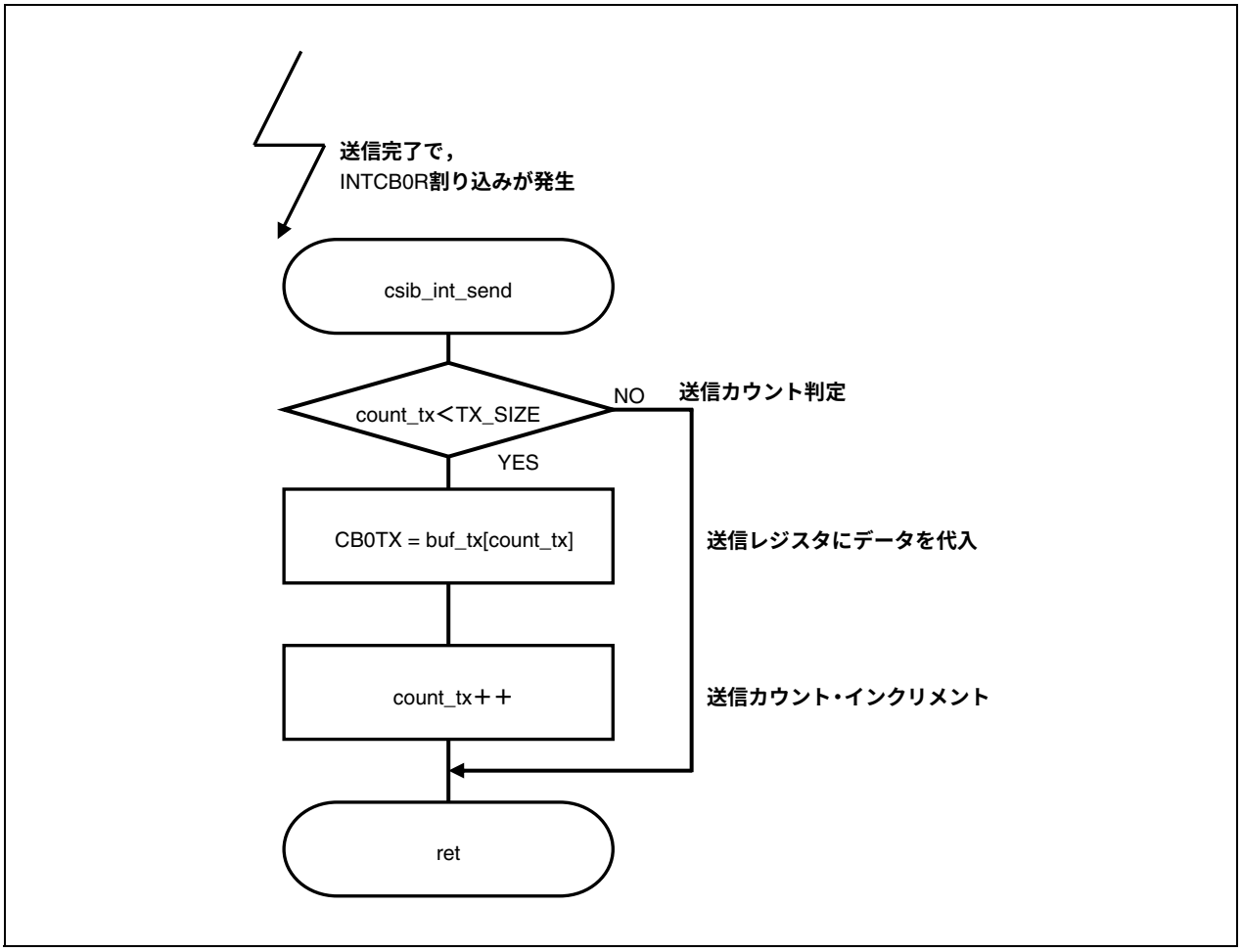


図11－12　シングル転送モード（マスタ・モード，送信モード）（4）



11.4　シングル転送モード（スレーブ・モード，受信モード）

【機　　能】	シングル転送モードで，通信モードをスレーブ・モード，転送方向モードを MSB ファーストに設定し，10 回のデータ受信を行います。 通信起動トリガを有効にし，通信クロックは外部クロック，転送データ長を 8 ビットに設定します。
【関　数　名】	csib4_main
【引　き　数】	なし
【処　理　内　容】	受信回数カウント（count_rx）を初期値 0 に設定します。各設定関数を呼び出したあと，受信を開始します。
【使 用　S　F　R】	CB0RMK　：0（CSIB0 受信終了割り込み要求信号（INTCB0R）クリア，マスク解除，優先レベル 7 に設定）
【call　関数】	csib_port, csib_set, csib_start, csib_end
【変　　数】	unsigned char buf_rx[ ]　：受信データ格納バッファ volatile unsigned char count_rx　：受信カウント変数
【割　り　込　み】	csib_int_receive
【割り込み要因】	INTCB0R
【ファイル名】	csib4¥csib4.c
【注　意　事　項】	なし

【関　数　名】	csib_port
【処　理　内　容】	ポート 4 を CSIB0 入出力端子に設定します。
【使 用　S　F　R】	PMC4　：0x05（$\overline{\text{SCKB0}}$ 入出力，SIB0 入力設定）
【call　関数】	なし
【変　　数】	なし
【ファイル名】	csib4¥csib4.c
【注　意　事　項】	なし

【関　数　名】　csib_set

【処　理　内　容】　CSIB0 制御レジスタの設定を行います。

【使 用　S　F　R】　CB0CTL1　：0x07（通信タイプ 1，外部クロックに設定）
CB0CTL2　：0x00（転送データ長を 8 ビットに設定）
CB0CTL0　：0x21（CSIB 受信動作許可，MSB ファースト，シングル転送モード，通信起動トリガ有効に設定）

【call　関数】　なし

【変　数】　なし

【ファイル名】　csib4¥csib4.c

【注　意　事　項】　CB0CTL0 レジスタは CB0PWR ビット = 0 時のみ CB0TXE, CB0RXE ビットの書き換え可能です。ただし，同時に CB0PWR ビット = 1 とするのは可能です。

【関　数　名】　csib_start

【処　理　内　容】　CSIB0 動作許可し，受信データ・レジスタをダミー・リードします。

【使 用　S　F　R】　CB0CTL0.CB0PWR　：1（CSIB0 動作許可）
CB0RX　受信データ・レジスタ

【call　関数】　なし

【変　数】　unsigned char buf_rx[ ]　：受信データ格納バッファ

【ファイル名】　csib4¥csib4.c

【注　意　事　項】　なし

【関　数　名】　csib_end

【処　理　内　容】　CSIB0 動作，受信動作を禁止します。

【使 用　S　F　R】　CB0CTL0.CB0PWR　：0（CSIB0 動作禁止）
CB0CTL0.CB0RXE　：0（CSIB0 受信動作禁止）

【call　関数】　なし

【変　数】　なし

【ファイル名】　csib4¥csib4.c

【注　意　事　項】　なし

割り込み関数

【関　数　名】	csib_int_receive
【処　理　内　容】	受信したデータをバッファに格納します。
【使 用 S F R】	CB0RX　受信データ・レジスタ CB0CTL0.CB0SCE　：0（通信起動トリガ無効）
【call　関数】	なし
【変　数】	unsigned char buf_rx[ ]　：受信データ格納バッファ volatile unsigned char count_rx　：受信カウント変数
【ファイル名】	csib4¥csib4.c
【注　意　事　項】	なし

図11－13　シングル転送モード（スレーブ・モード，受信モード）（1）

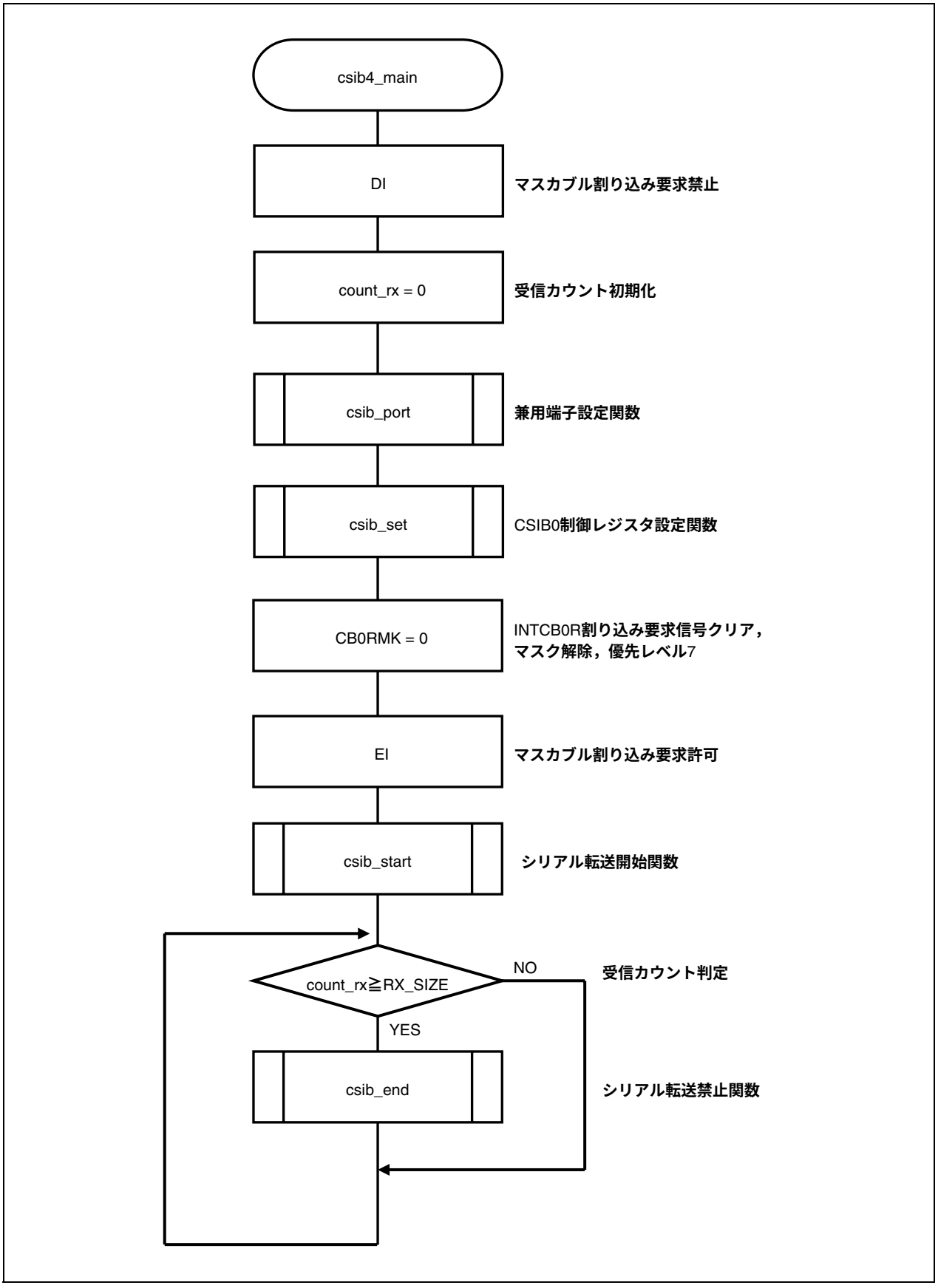


図11－14　シングル転送モード（スレーブ・モード，受信モード）（2）

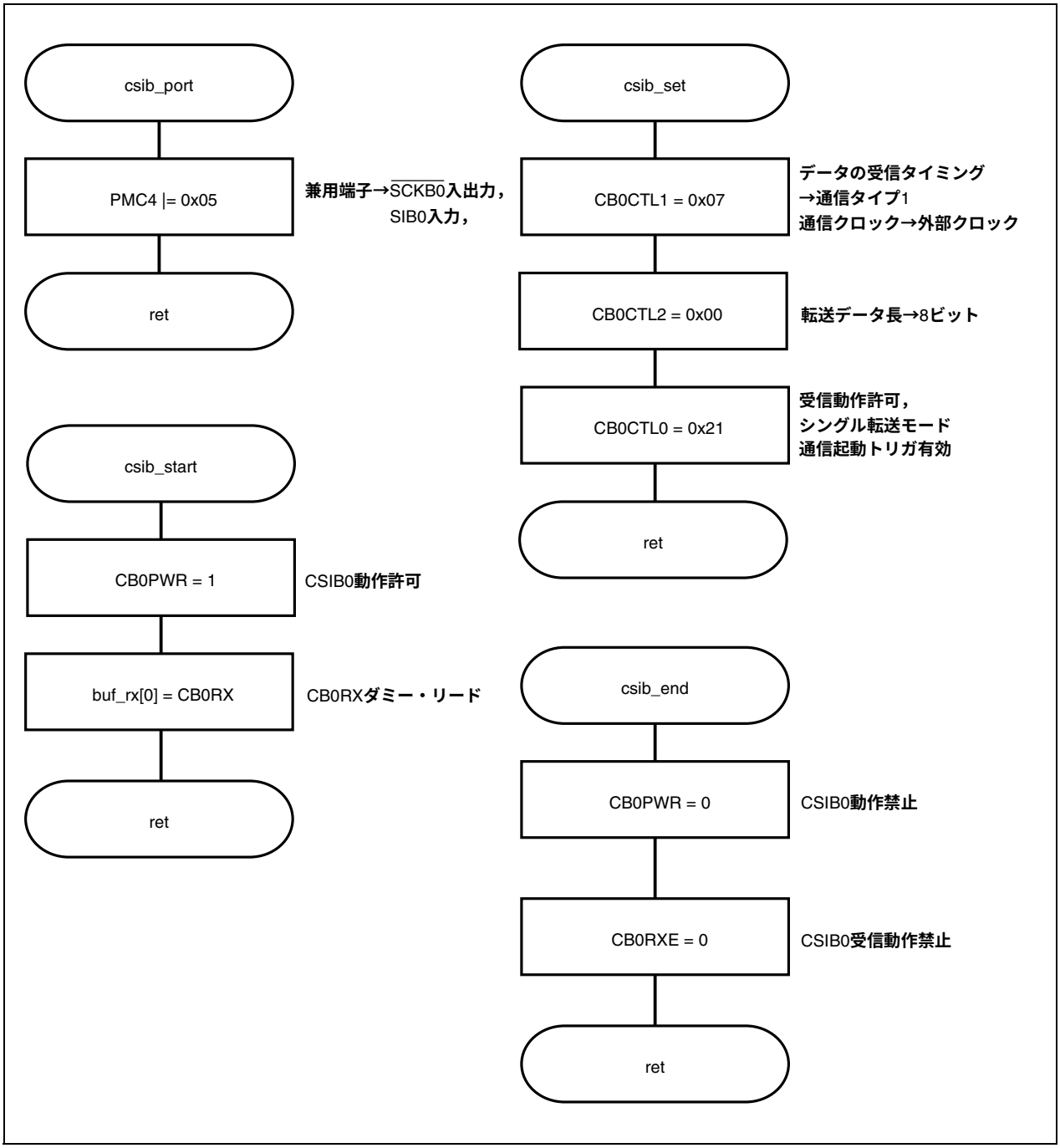
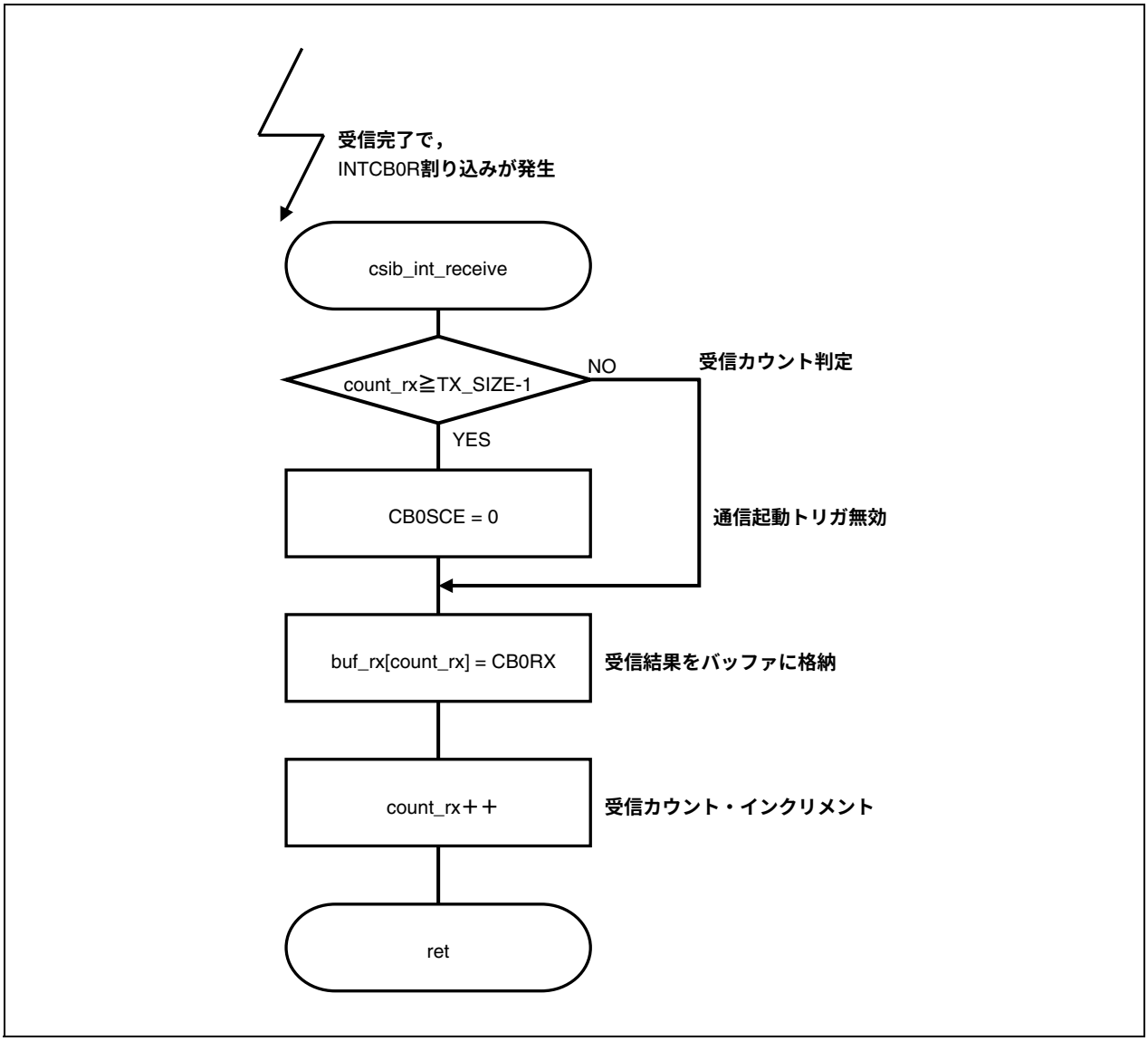


図11－15　シングル転送モード（スレーブ・モード，受信モード）（3）



第12章　割り込み／例外処理機能

12. 1　外部割り込み

【機　　能】	INTP3 を使用して外部割り込みを実現します。
【関　数　名】	ex_int_main
【引　き　数】	なし
【処　理　内　容】	・初期設定関数を呼び出し，割り込みの各設定を行います。 ・割り込みを許可します。
【使 用 ＳＦＲ】	なし
【call　　関数】	ex_int_init, ex_int_end
【変　　数】	unsigned char flag
【割　り　込　み】	ex_int_p3_handler
【割り込み要因】	INTP3
【ファイル名】	ex_intrrupt¥ex_interrupt.c
【注　意　事　項】	なし

【関　数　名】	ex_int_init
【処　理　内　容】	外部割り込みの初期設定を行います。
【使 用 ＳＦＲ】	なし
【call　　関数】	ex_int_p3_init, ex_int_interrupt_init
【変　　数】	なし
【ファイル名】	ex_intrrupt¥ex_interrupt.c
【注　意　事　項】	なし

【関　数　名】 ex_int_p3_init

【処 理 内 容】 P06 端子を外部マスカブル割り込み要求入力に設定し，エッジ検出の設定を行います。

【使 用 S F R】

PMC0	：0x40（P06 端子を外部マスカブル割り込み要求入力（INTP3）に設定
INTR0	：0x40（有効エッジを立ち上がりエッジに設定）
INTF0	：0x00（有効エッジを立ち上がりエッジに設定）

【call　関数】 なし

【変　数】 なし

【ファイル名】 ex_intrrupt¥ex_interrupt.c

【注 意 事 項】 なし

【関　数　名】 ex_int_interrupt_init

【処 理 内 容】 割り込みの初期設定を行います。

【使 用 S F R】

PIC3	：0x45（INTP3 割り込み要求信号の優先順位をレベル 5 に設定）
PIC3. PIF3	：0（INTP3 割り込み要求フラグクリア）
PIC3.PMK3	：0（INTP3 割り込み処理許可）

【call　関数】 なし

【変　数】 なし

【ファイル名】 ex_intrrupt¥ex_interrupt.c

【注 意 事 項】 なし

【関　数　名】 ex_in_end

【処 理 内 容】 外部割り込み発生後，端子をポート・モード，エッジ指定をエッジ検出なし，割り込み禁止に設定します。

【使 用 S F R】

INTR0	：0x00（エッジ検出なし設定）
INTF0	：0x00（エッジ検出なし設定）
PMC0	：0x00（P06 端子をポート・モードに設定）
PIC3.PMK3	：1（INTP3 割り込み処理禁止）

【call　関数】 なし

【変　数】 なし

【ファイル名】 ex_intrrupt¥ex_interrupt.c

【注 意 事 項】 なし

割り込み関数

【関　　数　　名】 ex_int_p3_handler

【処　理　内　容】 **割り込み発生確認のため，特に処理内容はありません。**

【使 用 ＳＦＲ】 **なし**

【call　　関数】 **なし**

【変　　数】 unsigned char flag

【ファイル名】 ex_intrrupt¥ex_interrupt.c

【注　意　事　項】 **なし**

図12－1　外部割り込み（1）

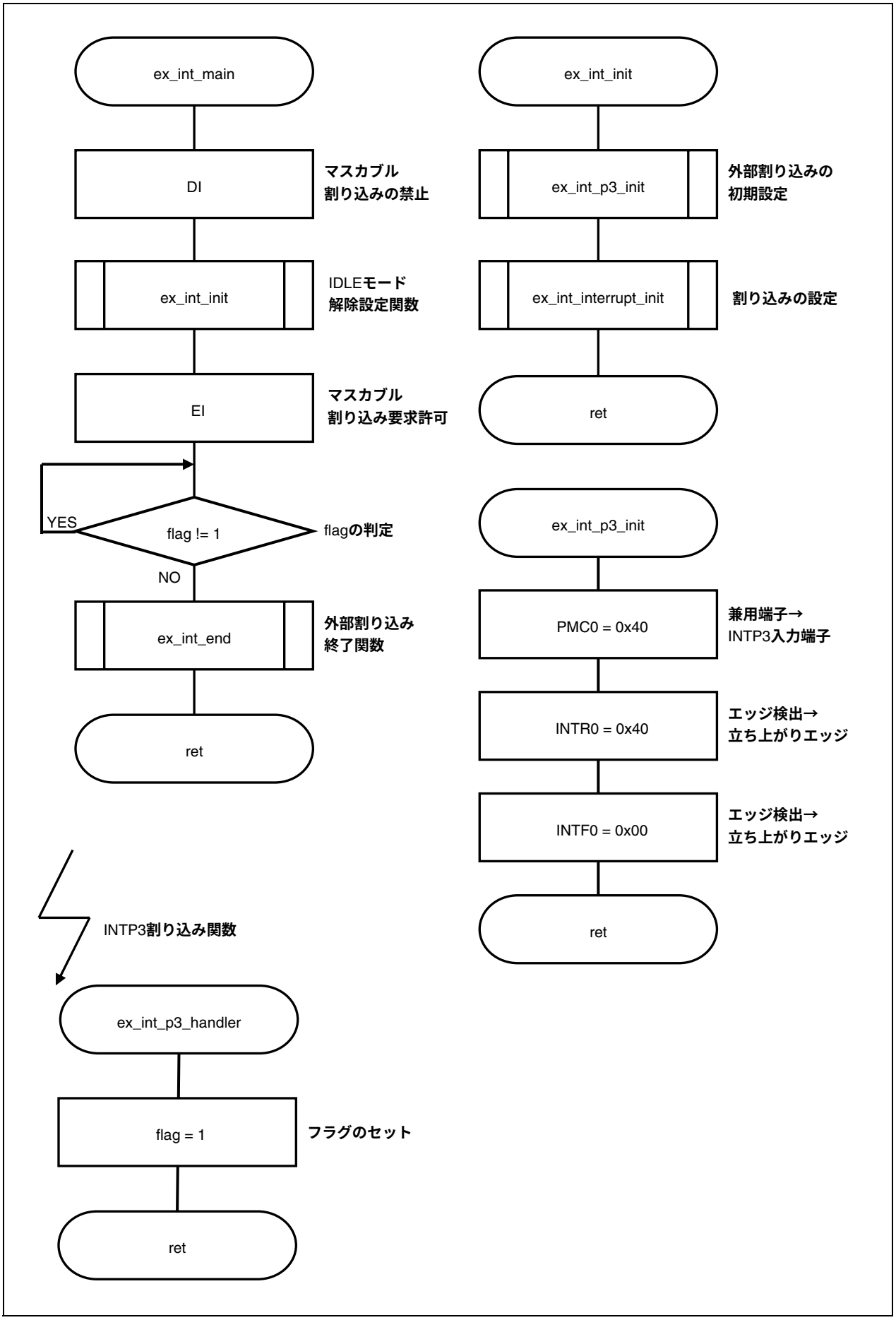
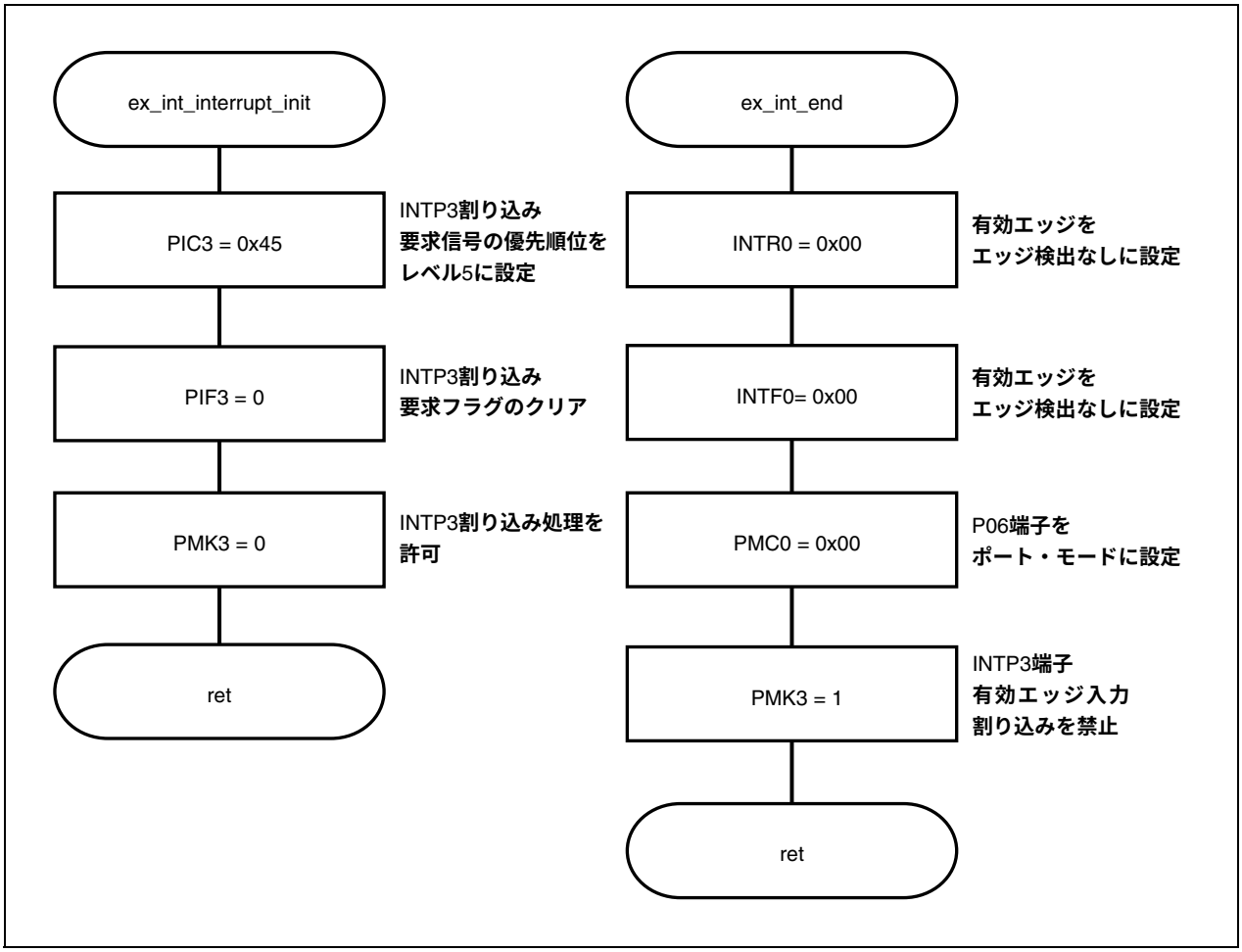


図12ー2　外部割り込み（2）



第13章　スタンバイ機能

13. 1　HALTモード

【機　　能】	スタンバイ機能（HALT モード）
【関　数　名】	halt_main
【引　き　数】	なし
【処　理　内　容】	通常動作モードから HALT モードに移行します。HALT モードは外部割り込み要求信号により解除されます。
【起　動　方　法】	なし
【使 用 ＳＦＲ】	なし
【call　　関数】	halt_init, halt_mode
【変　　数】	なし
【割　り　込　み】	external_int
【割り込み要因】	INTP3
【ファイル名】	halt¥halt.c
【注　意　事　項】	なし

【関　数　名】	halt_init	
【処　理　内　容】	HALT モードを解除するための外部割り込み要求信号（INTP3）を設定します。	
【使 用 ＳＦＲ】	PMC0	：0x40（兼用端子設定）
	INTR0	：0x40（有効エッジを立ち上がりエッジに設定）
	PIC3.PIF3	：0（INTP3 割り込み要求フラグのクリア）
	PIC3.PMK3	：0（INTP3 割り込み処理許可）
【call　　関数】	なし	
【変　　数】	なし	
【ファイル名】	halt¥halt.c	
【注　意　事　項】	なし	

【関　　数　　名】　halt_mode

【処　理　内　容】　HALT 命令を実行します。

【使 用 ＳＦＲ】　なし

【call　　関数】　なし

【変　　数】　なし

【ファイル名】　halt¥halt.c

【注　意　事　項】　HALT 命令のあとには，NOP 命令を 5 命令以上挿入してください。

割り込み関数

【関　　数　　名】　external_int

【処　理　内　容】　外部割り込み発生確認のため，特に処理内容はありません。

【使 用 ＳＦＲ】　なし

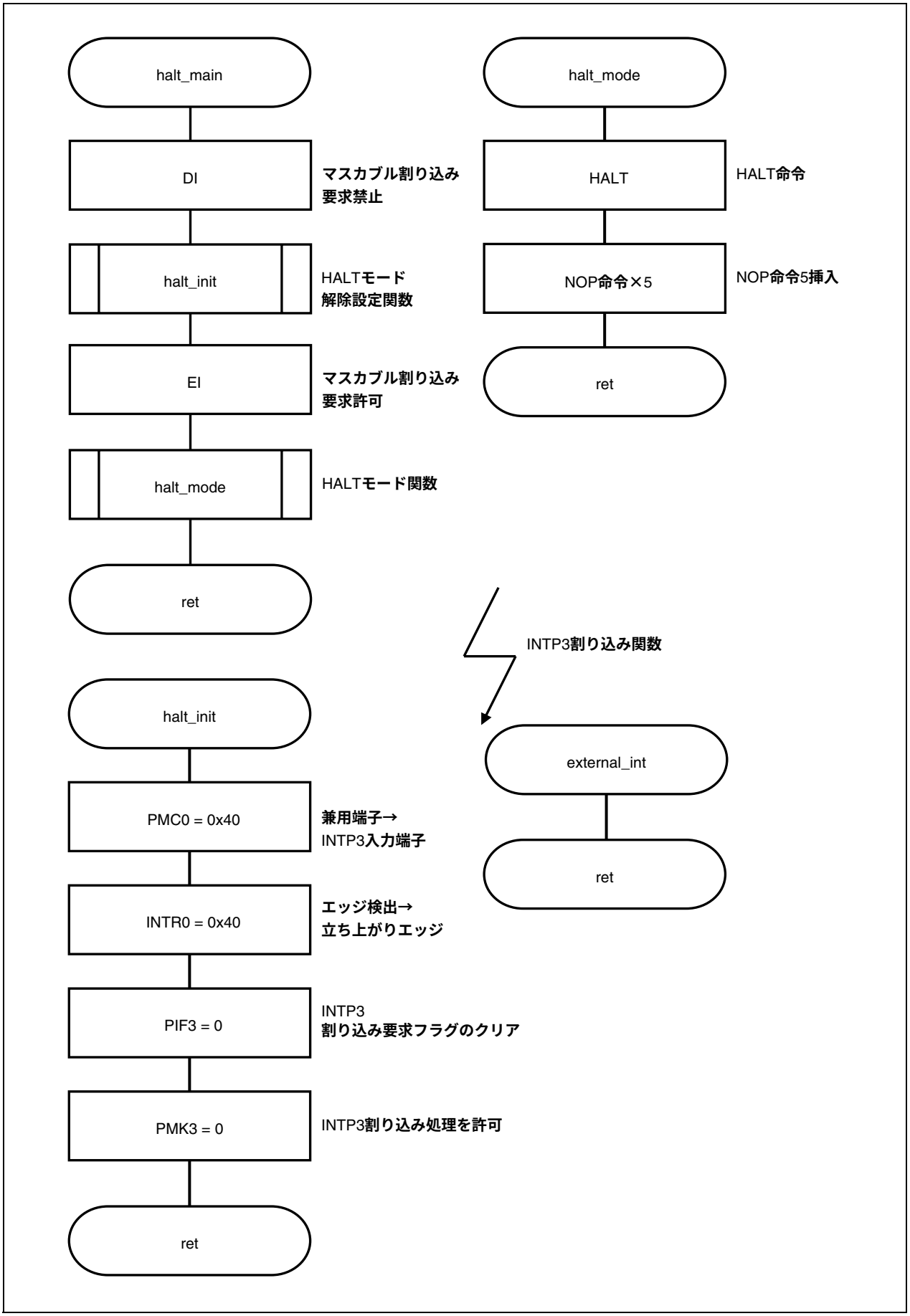
【call　　関数】　なし

【変　　数】　なし

【ファイル名】　halt¥halt.c

【注　意　事　項】　なし

図13ー1　HALTモード



13. 2　IDLEモード

【機　　能】	スタンバイ機能（IDLE モード）
【関　数　名】	idle_main
【引　き　数】	なし
【処　理　内　容】	通常動作モードから IDLE モードに移行します。IDLE モードは外部割り込み要求信号により解除されます。
【起　動　方　法】	なし
【使 用 S F R】	なし
【call　　関数】	idle_init, idle_mode
【変　　数】	なし
【割　り　込　み】	external_int
【割り込み要因】	INTP3
【ファイル名】	idle¥idle.c
【注　意　事　項】	なし

【関　数　名】	idle_init	
【処　理　内　容】	IDLE モードを解除するための外部割り込み要求信号（INTP3）を設定します。	
【使 用 S F R】	PMC0	：0x40（兼用端子設定）
	INTR0	：0x40（有効エッジを立ち上がりエッジに設定）
	PIC3.PIF3	：0（INTP3 割り込み要求フラグのクリア）
	PIC3.PMK3	：0（INTP3 割り込み処理許可）
【call　　関数】	なし	
【変　　数】	なし	
【ファイル名】	idle¥idle.c	
【注　意　事　項】	なし	

【関　数　名】	idle_mode
【処　理　内　容】	すべての DMA 転送を禁止にし，IDLE モードに移行します。
【使 用 S F R】	DCHC0.E00　：0（DMA0 転送禁止）
	DCHC1.E11　：0（DMA1 転送禁止）
	DCHC2.E22　：0（DMA2 転送禁止）
	DCHC3.E33　：0（DMA3 転送禁止）
	PSMR　：0x00（IDLE モードに設定）
	PRCMD　：0x02（コマンド・レジスタへの書き込み 特定レジスタへの書き込み時に使用）
	PSC　：0x02（スタンバイ・モード解除許可，スタンバイ・モード設定）
【call　関数】	なし
【変　数】	なし
【ファイル名】	idle¥idle.c
【注　意　事　項】	・特定レジスタへのデータ設定を行う前に，DMA 転送を禁止しておく必要があるので，このサンプル・プログラムでは DMA を転送禁止にしてあります。 ・PCC, PSC レジスタは特定レジスタなので，特定のシーケンスの組み合わせによってのみ書き込みができます。

割り込み関数

【関　数　名】	external_int
【処　理　内　容】	外部割り込み発生確認のため，特に処理内容はありません。
【使 用 S F R】	なし
【call　関数】	なし
【変　数】	なし
【ファイル名】	idle¥idle.c
【注　意　事　項】	なし

図13－2　IDLEモード（1）

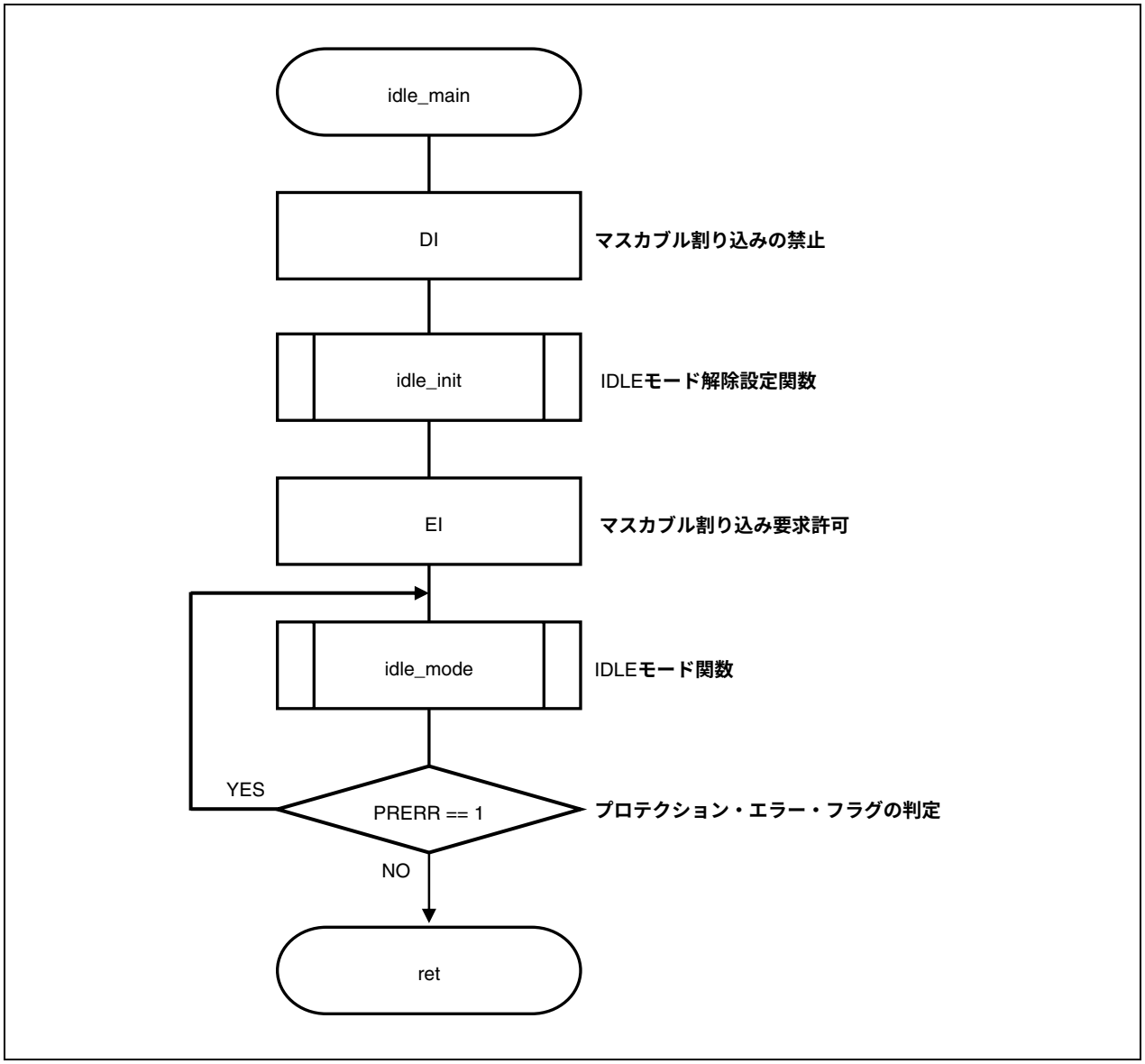


図13－3　IDLEモード（2）

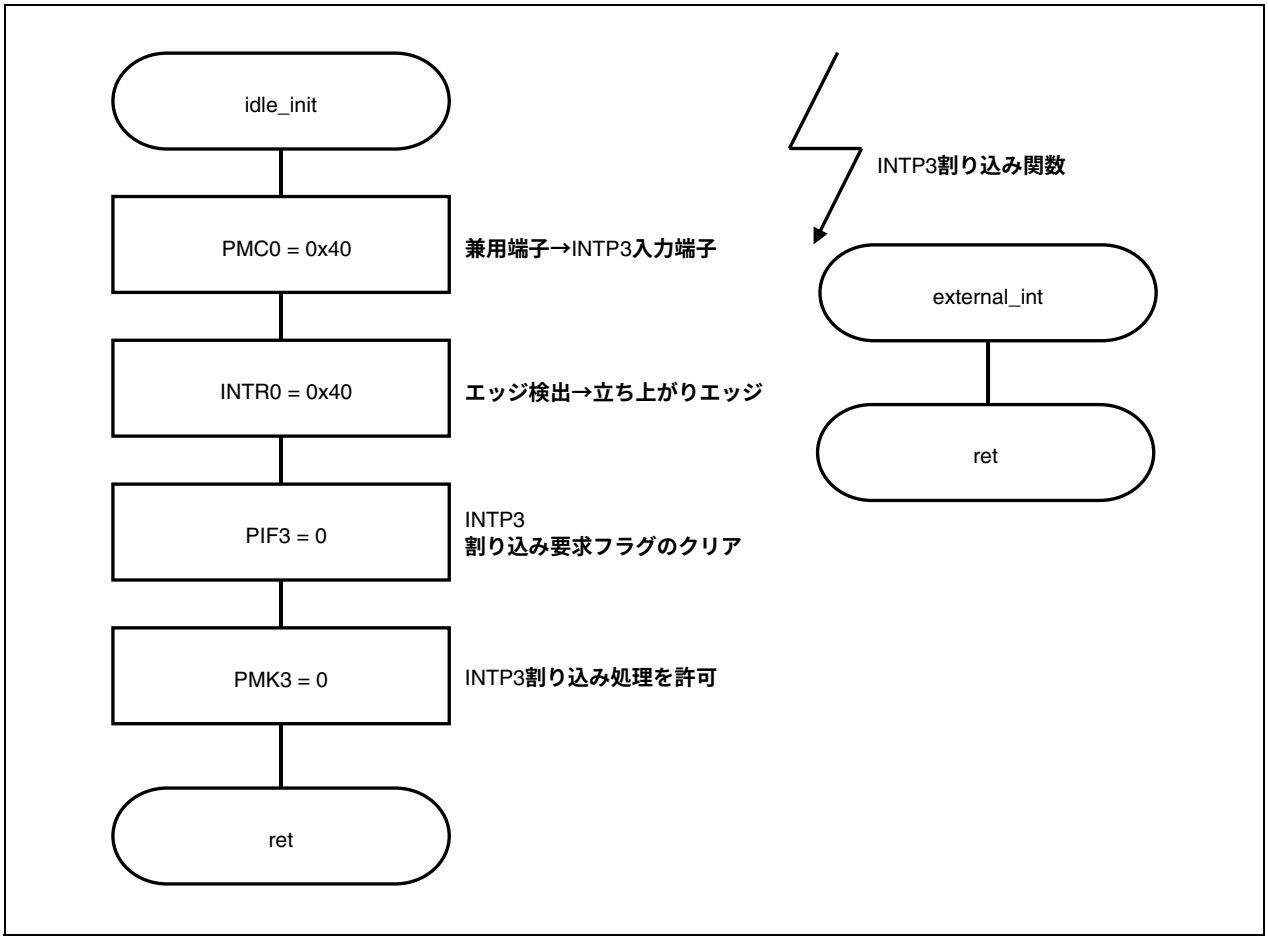
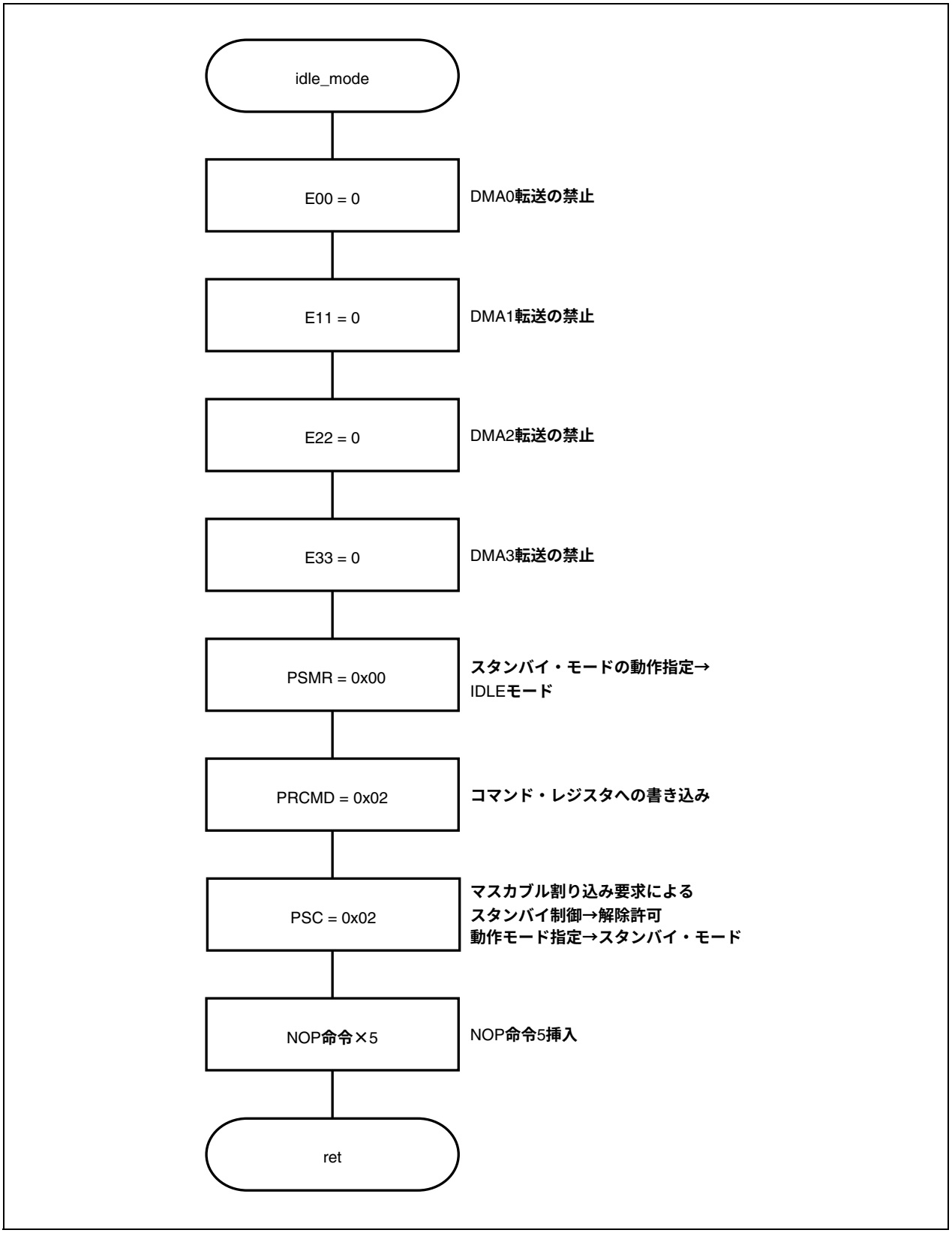


図13－4　IDLEモード（3）



13. 3　STOPモード

【機　　能】	スタンバイ機能（STOP モード）
【関　数　名】	stop_main
【引　き　数】	なし
【処　理　内　容】	通常動作モードから STOP モードに移行します。STOP モードは外部割り込み要求信号により解除されます。
【起　動　方　法】	なし
【使 用 S F R】	なし
【call　　関数】	stop_init, stop_mode
【変　　数】	なし
【割　り　込　み】	なし
【割り込み要因】	なし
【ファイル名】	stop¥stop.c
【注　意　事　項】	なし

【関　数　名】	stop_init	
【処　理　内　容】	STOP モードを解除するための外部割り込み要求信号（INTP3）を設定します。	
【使 用 S F R】	OSTS	：0x06（発振安定時間 13.107 ms）
	PMC0	：0x40（兼用端子設定）
	INTR0	：0x40（有効エッジを立ち上がりエッジに設定）
	PIC3.PIF3	：0（INTP3 割り込み要求フラグのクリア）
	PIC3.PMK3	：0（INTP3 割り込み処理許可）
【call　　関数】	なし	
【変　　数】	なし	
【ファイル名】	stop¥stop.c	
【注　意　事　項】	なし	

【関　数　名】	stop_mode
【処　理　内　容】	すべての DMA 転送を禁止にし，STOP モードを実行します。
【使 用 S F R】	DCHC0.E00　：0（DMA0 転送禁止） DCHC1.E11　：0（DMA1 転送禁止） DCHC2.E22　：0（DMA2 転送禁止） DCHC3.E33　：0（DMA3 転送禁止） PSMR　：0x01（STOP モードに設定） PRCMD　：0x02（コマンド・レジスタへの書き込み，特定レジスタへの書き込み時に使用） PSC　：0x02（スタンバイ・モード解除許可，スタンバイ・モード設定）
【call　関数】	なし
【変　数】	なし
【ファイル名】	stop¥stop.c
【注　意　事　項】	・特定レジスタへのデータ設定を行う前に，DMA 転送を禁止しておく必要があるので，このサンプル・プログラムでは DMA を転送禁止にしてあります。 ・PSC レジスタは特定レジスタなので，特定のシーケンスの組み合わせによってのみ書き込みができます。 ・STOP モードに設定する場合には，PSMR.PSM0 ビット，PSC.STB ビットの順序で設定してください。

割り込み関数

【関　数　名】	external_int
【処　理　内　容】	外部割り込み発生確認のため，特に処理内容はありません。
【使 用 S F R】	なし
【call　関数】	なし
【変　数】	なし
【ファイル名】	stop¥stop.c
【注　意　事　項】	なし

図13－5　STOPモード（1）

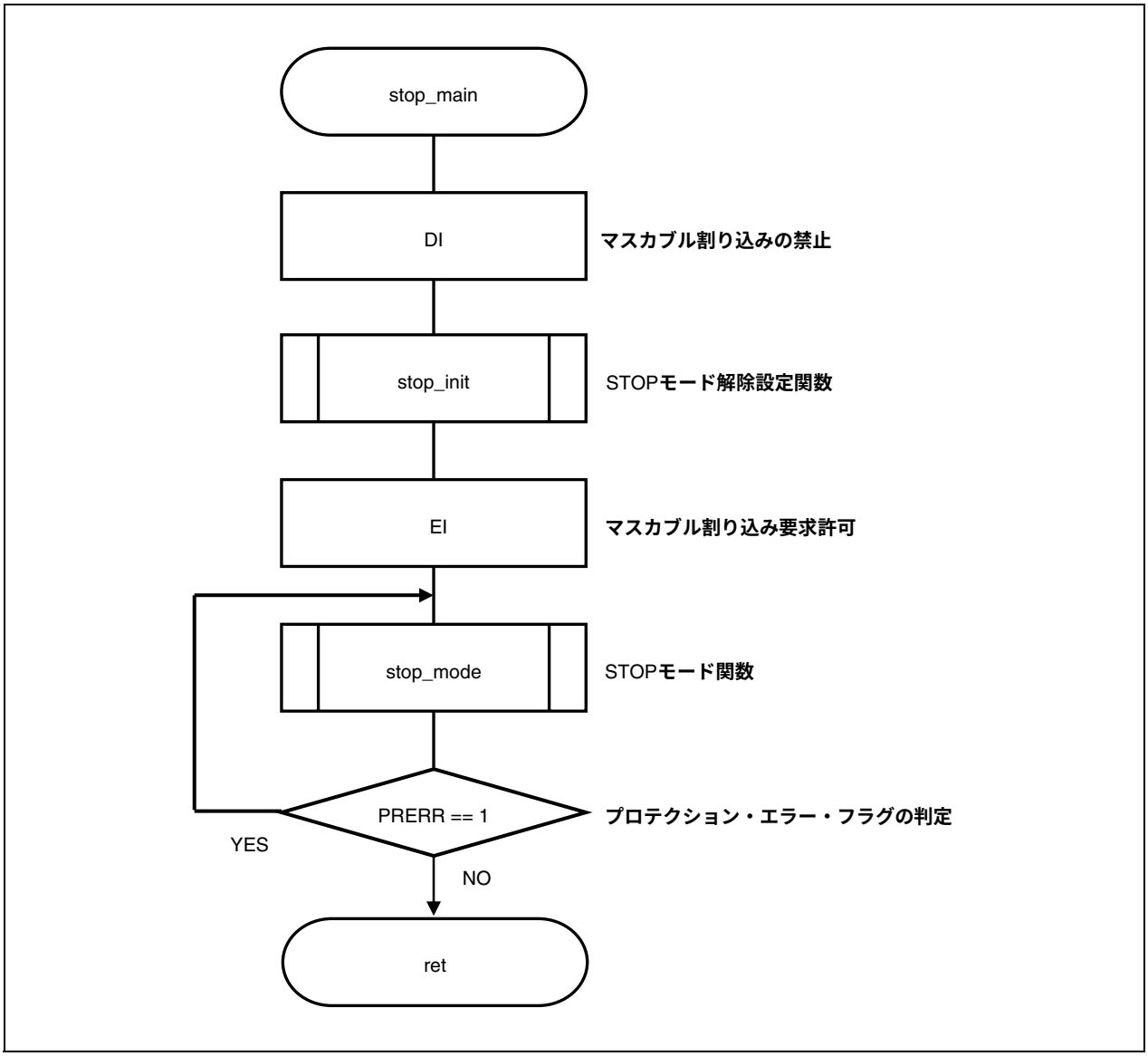
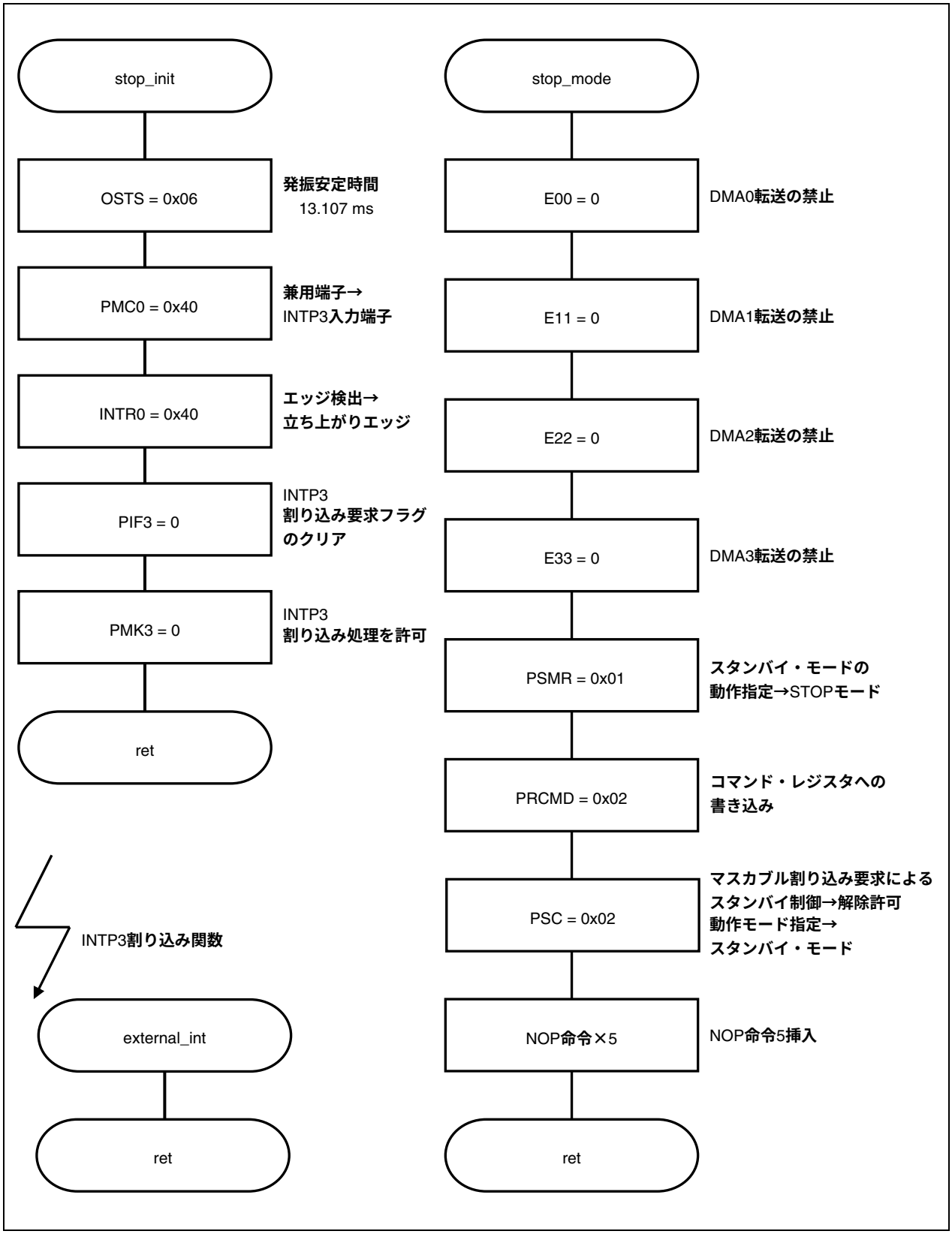


図13－6　STOPモード（2）

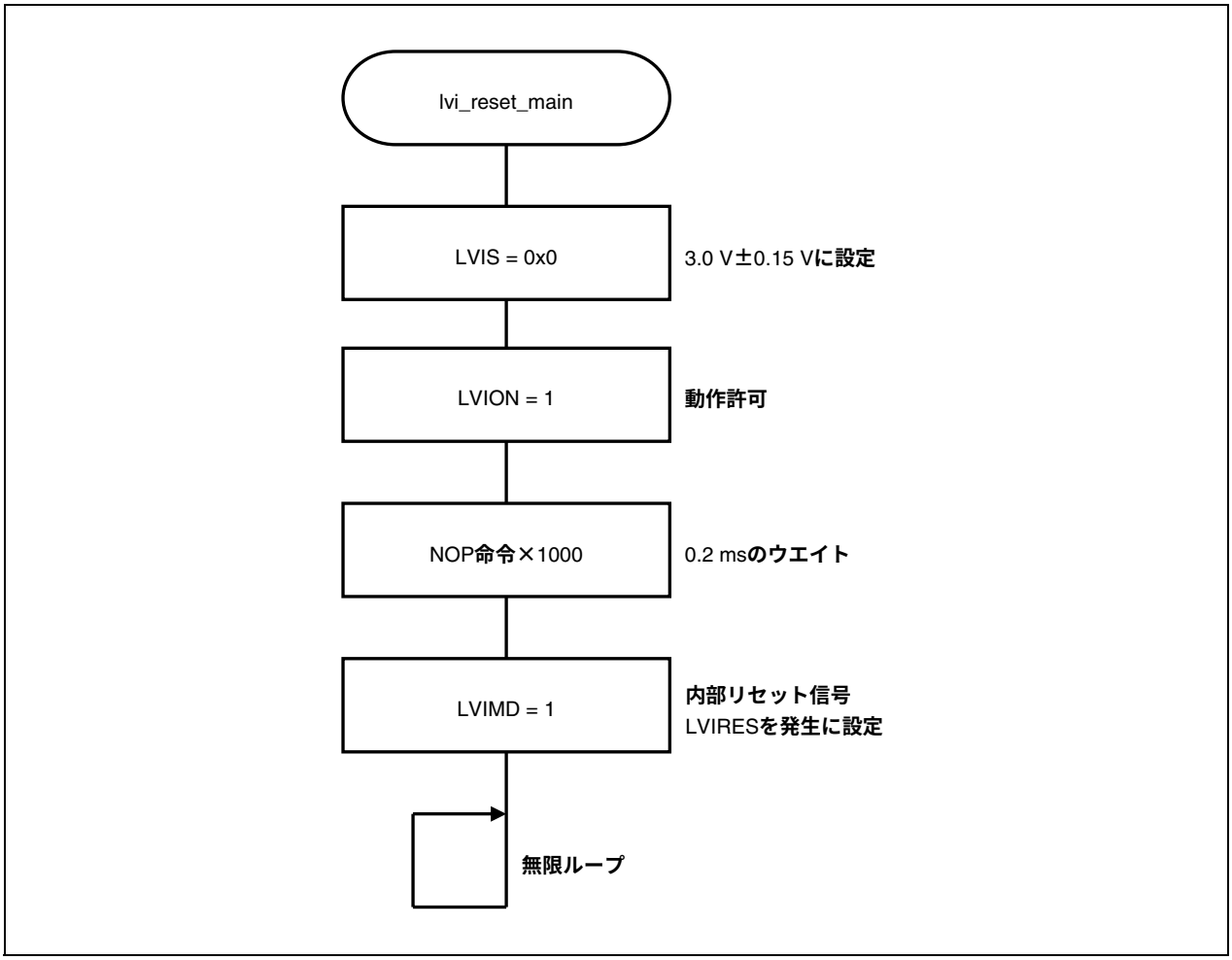


第14章　低電圧検出回路（LVI）

14. 1　内部リセット・モード

【機　　能】	電源電圧（V_{DD}）と検出電圧（V_{LVI}）を比較し V_{DD}＜V_{LVI} になったとき，内部リセット信号を発生します。
【関　数　名】	lvi_reset_main
【引　き　数】	なし
【処　理　内　容】	LVIS レジスタ，LVION レジスタ，LVIMD レジスタを設定し，V_{DD}＜V_{LVI} になったとき，内部リセット信号を発生させます。
【使 用　S　F　R】	LVIS　：0x0（検出レベルを 3.0 V±0.15 V に設定） LVIM.LVION　：1（動作許可に設定） LVIS.LVIMD　：1（低電圧検出で内部リセット信号 LVIRES を発生に設定）
【 call　関数】	なし
【変　　数】	なし
【割　り　込　み】	なし
【割り込み要因】	なし
【ファイル名】	lvi_reset¥lvi_reset.c
【注　意　事　項】	LVIMD ビット = 1 に設定した場合，LVI 以外のリセット要求が発生するまで，LVIM, LVIS レジスタの変更はできません。

図14－1　内部リセット・モード



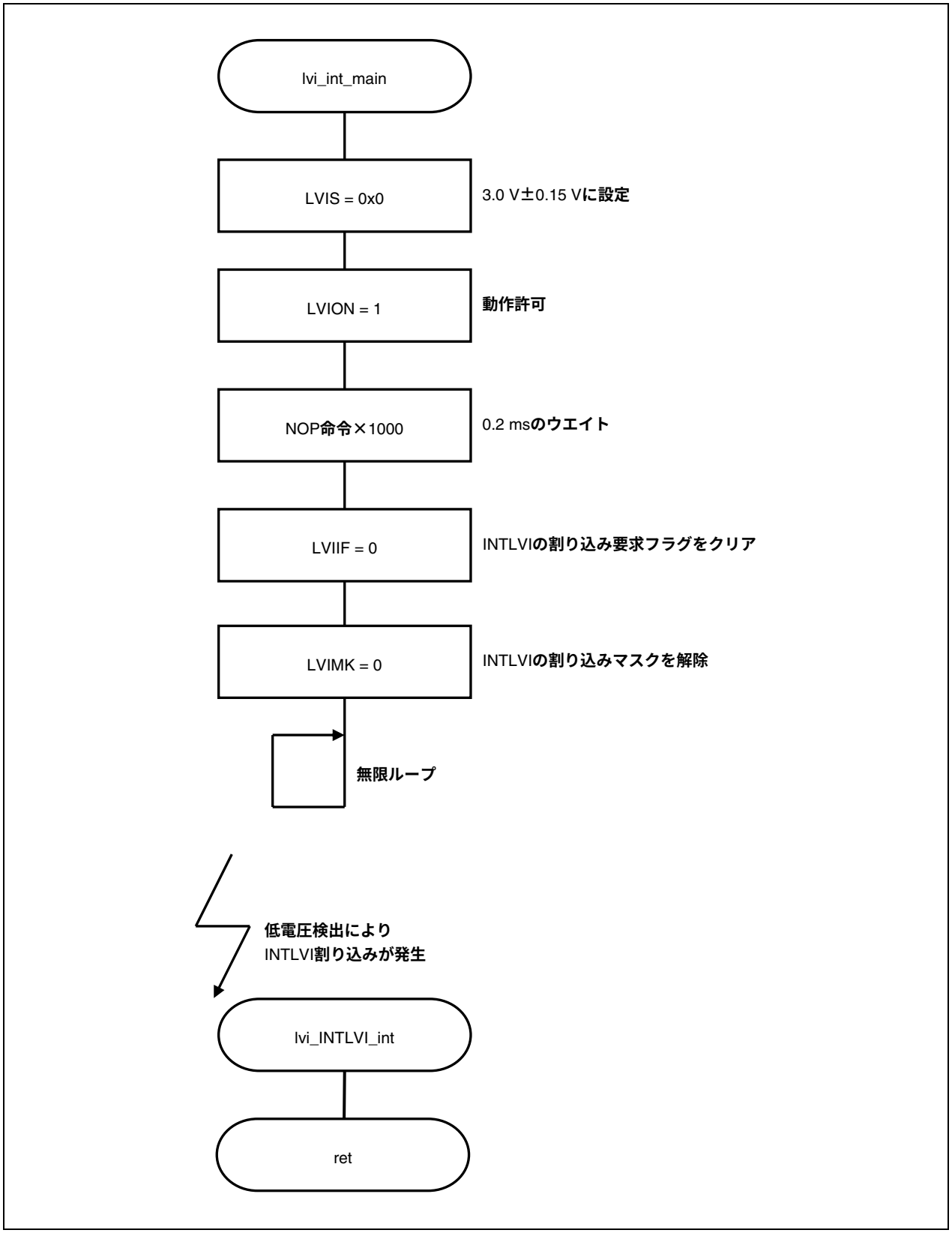
14. 2　内部割り込みモード

【機　　能】	電源電圧（V_{DD}）と検出電圧（V_{LVI}）を比較し V_{DD}＜V_{LVI} になったとき，内部割り込み信号（INTLVI）を発生します。
【関　数　名】	lvi_int_main
【引　き　数】	なし
【処 理 内 容】	LVIS レジスタ，LVION レジスタ，LVIMD レジスタを設定し，V_{DD}＜V_{LVI} になったとき，内部割り込み信号を発生させます。
【使 用 S F R】	LVIS　：0x0（検出レベルを 3.0 V±0.15 V に設定） LVIM.LVION　：1（動作許可に設定） LVIIF　：0（INTLVI の割り込み要求フラグをクリア） LVIMK　：0（INTLVI の割り込みマスクをクリア）
【 call　関数】	なし
【変　　数】	なし
【割 り 込 み】	lvi_INTLVI_int
【割り込み要因】	INTLVI
【ファイル名】	lvi_int¥lvi_int.c
【注 意 事 項】	なし

割り込み関数

【関　数　名】	lvi_INTLVI_int	
【概　　要】	ユーザ定義	
【要　　因】	INTLVI	低電圧の検出
【call　関数】	なし	
【変　　数】	なし	
【ファイル名】	lvi_int¥lvi_int.c	
【注 意 事 項】	なし	

図14－2　内部割り込みモード



【発　行】
NECエレクトロニクス株式会社
〒211-8668　神奈川県川崎市中原区下沼部1753
電話（代表）：044(435)5111

お問い合わせ先

【ホームページ】

NECエレクトロニクスの情報がインターネットでご覧になれます。
URL(アドレス)　http://www.necel.co.jp/

【営業関係，技術関係お問い合わせ先】

半導体ホットライン
（電話：午前 9:00～12:00，午後 1:00～5:00）
電　話　：044−435−9494
E-mail　：info@necel.com

【資料請求先】

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