User's Manual

32

RZ/N1D グループ、RZ/N1S グループ、RZ/N1L グループ

ユーザーズマニュアル　周辺機能 2 編

RZ ファミリ
RZ/N シリーズ

ルネサス エレクトロニクス
www.renesas.com

Rev.1.20　2021.12

ご注意書き

1. 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。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、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 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。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

   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、産業用ロボット等

   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合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いますが、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、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等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、お客様の機器・システムとしての出荷保証を行ってください。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、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の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。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. 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・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
13. 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禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

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直接的、間接的に支配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。

(Rev.5.0-1　2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品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、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、CMOS 製品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。
2. 電源投入時の処置
   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、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「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。
5. クロックについて
   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
6. 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 製品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。入力レベルが固定の場合はもちろん、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などが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
8. 製品間の相違について
   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

このマニュアルの使い方

1. 目的と対象者

このマニュアルは、本マイコンのハードウェア機能と電気的特性をユーザに理解していただくためのマニュアルです。本マイコンを用いた応用システムを設計するユーザを対象にしています。このマニュアルを使用するには、電気回路、論理回路、マイクロコンピュータに関する基本的な知識が必要です。

このマニュアルは、大きく分類すると、製品の概要、CPU、システム制御機能、周辺機能、電気的特性、使用上の注意で構成されています。

本マイコンは、注意事項を十分確認の上、使用してください。注意事項は、各章の本文中、各章の最後、注意事項の章に記載しています。

改訂記録は旧版の記載内容に対して訂正または追加した主な箇所をまとめたものです。改訂内容すべてを記録したものではありません。詳細は、このマニュアルの本文でご確認ください。

RZ/N1 グループでは次のドキュメントを用意しています。

■ RZ/N1 関連ドキュメント一覧

ドキュメント名	資料番号
RZ/N1D グループ、RZ/N1S グループ、RZ/N1L グループ　データシート	R01DS0323JJ****
RZ/N1D グループ、RZ/N1S グループ、RZ/N1L グループ　ユーザーズマニュアル アーキテクチャ・製品データ編	R01UH0750JJ****
RZ/N1D グループ、RZ/N1S グループ、RZ/N1L グループ　ユーザーズマニュアル システム・周辺機能 1 編	R01UH0751JJ****
RZ/N1D グループ、RZ/N1S グループ、RZ/N1L グループ　ユーザーズマニュアル 周辺機能 2 編	R01UH0752JJ**** （本ユーザーズマニュアル）
RZ/N1D グループ、RZ/N1S グループ、RZ/N1L グループ　ユーザーズマニュアル R-IN Engine・イーサネット編	R01UH0753JJ****
RZ/N1D グループ、RZ/N1S グループ、RZ/N1L グループ　ユーザーズマニュアル PWMTimer	R01UH0913JJ****

2. レジスタの表記

各章において「レジスタの説明」には、ビットの並びを示すビット配置図とビットに設定する内容を説明するビット機能表があります。使用する記号、用語を以下に説明します。

X.X.X　[レジスタ名]

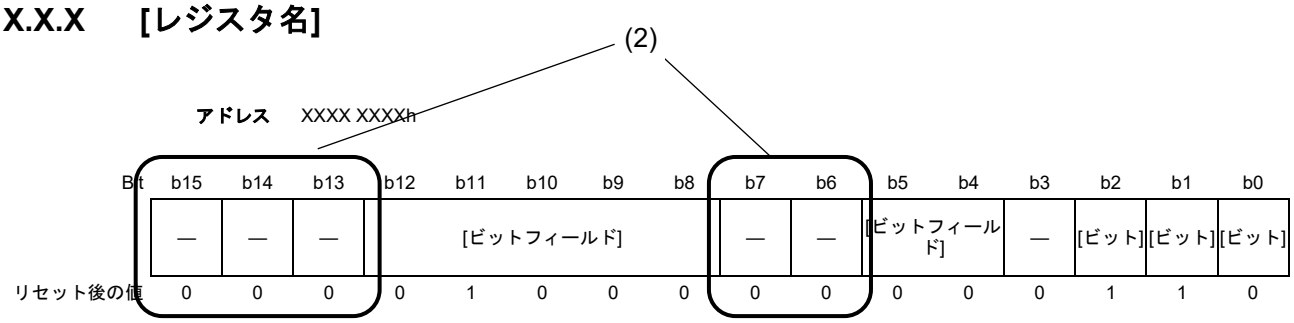


表 X.X　　[レジスタ名]レジスタの内容

ビット位置	ビット名	機能	R/W
b12～b8	[ビットフィールド]	[説明]	R/W
b5～b4	[ビットフィールド]	[説明] 2'b00：Hi-Z 2'b01：L 出力 上記以外：設定禁止 (3)	R/W
b2	[ビット]	[説明] 1'b0：Hi-Z 1'b1：出力（デフォルト値）	R/W
b1	[ビット]	[説明] 1'b0：Hi-Z 1'b1：出力（デフォルト値）	R/W (1)
b0	[ビット]	[説明] 1'b0：Hi-Z 1'b1：出力	R/W

(1) R/W　：読み出し／書き込みとも有効です。

R/(W)　：読み出し／書き込みとも有効ですが、書き込みには制限があります。
制限の内容については、各レジスタの説明や注記を参照ください。

R　：読み出しのみ有効です。書き込みは無効になります。

W　：書き込みのみ有効です。読み出した値は保証されません。

(2) 予約ビットです。書き込みを行う場合には、指定された値を書き込んでください。指定以外の値を書き込んだ場合の動作は保証されません。

(3) 設定禁止。設定した場合の動作は保証されません。

3. 略語および略称の説明

略語／略称	フルスペル
AHB	Arm Advanced High-performance Bus
APB	Arm Advanced Peripheral Bus
AXI	Arm Advanced eXtensible Interface
bps	bits per second
CA7	Arm Cortex-A7 module
CM3	Arm Cortex-M3 module
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
Hi-Z	High Impedance
HSR	High-availability Seamless Redundancy
HW-RTOS	Hard Ware Real Time OS
I/O	Input/Output
INTC	Interrupt Controller
LSB	Least Significant Bit
MSB	Most Significant Bit
NC	Non-Connect
NoC	Network-on-Chip
PLL	Phase Locked Loop
PWM	Pulse Width Modulation
UART	Universal Asynchronous Receiver/Transmitter
OTP	One Time Programmable
PTP	Precision Time Protocol
PRP	Parallel Redundancy Protocol
SoC	System On Chip

4. アクセスサイズの表記

アクセスサイズ：

8 ビット＝バイト

16 ビット＝ハーフワード

32 ビット＝ワード

CAN（Controller Area Network）：ドイツ Robert Bosch GmbH により開発された車載ネットワーク規格です。

目　次

第1章　UART

1.1　概要

RZ/N1 の PG0（ペリフェラルグループ 0）サブシステムおよび PG1（ペリフェラルグループ 1）サブシステムは UART を合計 8 ブロック搭載しています。

各 UART には、DMA 機能を除き、同じ機能が搭載されています。

- PG0 の UART r（縮小版）　：　UART1、UART2、UART3
- PG1 の UART f（フル）　　：　UART4、UART5、UART6、UART7、UART8

各 UART は、以下の機能を有しています。

- 16550 UART に基づく以下の機能
  - 1 キャラクタあたりのデータビット数（5～8）、任意選択のパリティビット（奇数または偶数を選択）、ストップビット数（1、1.5、または 2）などのキャラクタのプロパティを設定可能
  - ラインブレークの生成と検出
  - 割り込みの優先識別
- 16×8（16×8 ビット幅）送信 FIFO と 16×8 受信 FIFO
- RS485 および MODBUS®の拡張機能
- FIFO 許可／禁止設定可能
- UART クロックのソースは以下の 2 つが可能
  1. プログラマブル周波数クロック（7.81MHz～83.33MHz）（メイン PLL）
  2. 固定 48MHz クロック（USBPLL）

  PG0 に属するプログラマブル整数分周器は UART1～3 で共有され、PG1 に属するものは UART4～8 で共有されます。
- プログラマブルボーレート生成（最高速度：UART_SCLK/16）
- 誤スタートビットの検出
- プログラマブルなハードウェアフロー制御
- ソフトウェアオーバーヘッドを減らしソフトウェアプログラマブルリセットを実現するためのシャドーレジスタ
- 16750 標準に準拠する自動フロー制御モード
- 送信保持レジスタエンプティ（THRE）割り込みモード
- ビジー機能
- モデム制御および自動フロー制御機能を拡張テストできるループバックモード
- 追加の FIFO ステータスレジスタ
- モデムラインとステータスラインを個別に制御可能
- 2 つのレシーバタイムアウトにより、MODBUS リンクでのフレーム間時間処理をサポート

- 2 つのトランシーバタイムアウトにより、MODBUS リンクでのフレーム間時間処理をサポート
- DE（データイネーブル）信号による「2 線式」インタフェース上の半二重管理
- TXD、RXD、CTS_N、RTS_N、DTR_N、DSR_N、DCD_N、RI_N（GPIO 端子上でマルチプレクス）をサポート

UART フル（UART4～8）の追加機能は以下です。

- DMA 接続
  - 周辺機能フローコントローラモード
  - DMA チャネルは 2 本使用可（送信用に 1 本、受信用に 1 本）

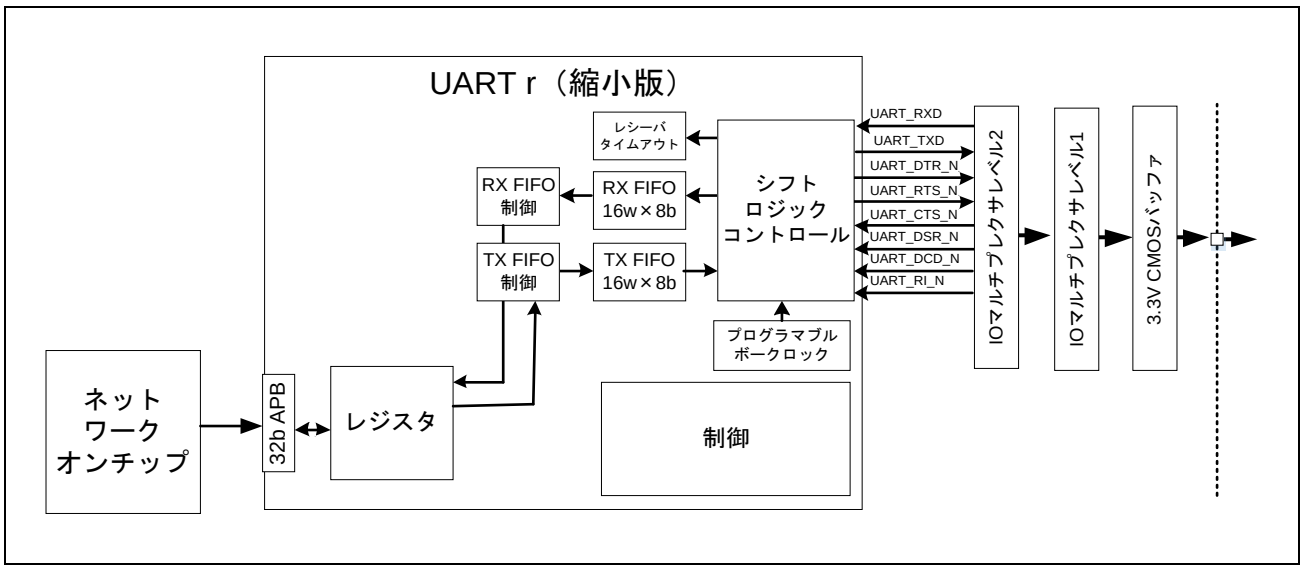


図 1.1　　縮小版 UART の概要図（UART1～3）

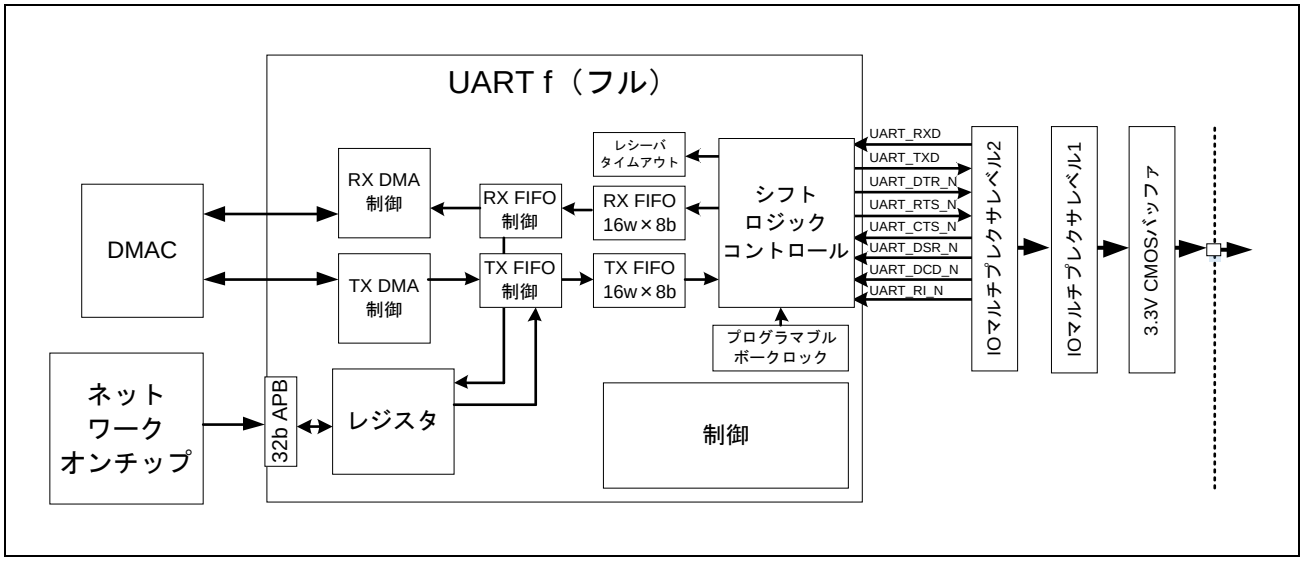


図 1.2　　フル版 UART の概要図（UART4～8）

1.2　信号インタフェース

信号名	入力／出力	説明
クロック		
UART[m]_PCLK	入力	内部バスクロック（APB）
UART[m]_SCLK	入力	シリアルリファレンスクロック
割り込み		
UART[m]_Int	出力	レベル検出割り込み出力、アクティブ High
外部信号		
UART[m]_RXD	入力	受信データ
UART[m]_TXD	出力	送信データ
UART[m]_CTS_N	入力	送信可モデム状態
UART[m]_DSR_N	入力	データセットレディモデム状態
UART[m]_DCD_N	入力	データキャリア検出モデム状態
UART[m]_RI_N	入力	被呼表示モデム状態
UART[m]_DTR_N	出力	モデム制御データ端末準備完了
UART[m]_RTS_N	出力	モデム制御送信要求（全二重モード） 送信データイネーブル（半二重モード）

備考　m＝1～8

本章では主にインデックスを省いたスタイルを使用します。

例）UART_PCLK

1.3　レジスタマップ

1.3.1　レジスタマップ UART1

表 1.1　レジスタマップ UART1

アドレス	レジスタシンボル	レジスタ名
4006 0000h	（bUart_DLAB＝0）rUart_RBR_THR	受信バッファ／送信保持レジスタ
	（bUart_DLAB＝1）rUart_DLL	除数ラッチ（Low）
4006 0004h	（bUart_DLAB＝0）rUart_IER	割り込み許可レジスタ
	（bUart_DLAB＝1）rUart_DLH	除数ラッチ（High）
4006 0008h	（書き込み時）rUart_FCR	FIFO コントロールレジスタ
	（読み出し時）rUart_IIR	割り込み識別レジスタ
4006 000Ch	rUart_LCR	ラインコントロールレジスタ
4006 0010h	rUart_MCR	モデムコントロールレジスタ
4006 0014h	rUart_LSR	ラインステータスレジスタ
4006 0018h	rUart_MSR	モデムステータスレジスタ
4006 001Ch	rUart_SCR	スクラッチパッドレジスタ
4006 0030h	rUart_SRBR_STHR	シャドー受信バッファ／送信保持レジスタ注 1
4006 0070h	rUart_FAR	FIFO アクセスレジスタ
4006 0074h	rUart_TFR	送信 FIFO 読み出し
4006 0078h	rUart_RFW	受信 FIFO 書き込み
4006 007Ch	rUart_USR	UART ステータスレジスタ
4006 0080h	rUart_TFL	送信 FIFO レベル
4006 0084h	rUart_RFL	受信 FIFO レベル
4006 0088h	rUart_SRR	ソフトウェアリセットレジスタ
4006 008Ch	rUart_SRTS	シャドー送信要求
4006 0090h	rUart_SBCR	シャドーブレークコントロールレジスタ
4006 0098h	rUart_SFE	シャドーFIFO 許可
4006 009Ch	rUart_SRT	シャドーRCVR トリガ
4006 00A0h	rUart_STET	シャドーTX エンプティトリガ
4006 00A4h	rUart_HTX	送信停止
4006 0100h	rUart_TO	タイムアウトカウンタコンフィグレーションレジスタ
4006 0104h	rUart_CTRLTO	タイムアウトコントロールレジスタ
4006 0108h	rUart_STATUSTO	タイムアウトカウンタステータスレジスタ

注1.　アドレスは、4006 0030h～4006 006Ch に割り当てられています。

1.3.2　レジスタマップ UART2

表 1.2　レジスタマップ UART2

アドレス	レジスタシンボル	レジスタ名
4006 1000h	（bUart_DLAB＝0）rUart_RBR_THR	受信バッファ／送信保持レジスタ
	（bUart_DLAB＝1）rUart_DLL	除数ラッチ（Low）
4006 1004h	（bUart_DLAB＝0）rUart_IER	割り込み許可レジスタ
	（bUart_DLAB＝1）rUart_DLH	除数ラッチ（High）
4006 1008h	（書き込み時）rUart_FCR	FIFO コントロールレジスタ
	（読み出し時）rUart_IIR	割り込み識別レジスタ
4006 100Ch	rUart_LCR	ラインコントロールレジスタ
4006 1010h	rUart_MCR	モデムコントロールレジスタ
4006 1014h	rUart_LSR	ラインステータスレジスタ
4006 1018h	rUart_MSR	モデムステータスレジスタ
4006 101Ch	rUart_SCR	スクラッチパッドレジスタ
4006 1030h	rUart_SRBR_STHR	シャドー受信バッファ／送信保持レジスタ[注 1]
4006 1070h	rUart_FAR	FIFO アクセスレジスタ
4006 1074h	rUart_TFR	送信 FIFO 読み出し
4006 1078h	rUart_RFW	受信 FIFO 書き込み
4006 107Ch	rUart_USR	UART ステータスレジスタ
4006 1080h	rUart_TFL	送信 FIFO レベル
4006 1084h	rUart_RFL	受信 FIFO レベル
4006 1088h	rUart_SRR	ソフトウェアリセットレジスタ
4006 108Ch	rUart_SRTS	シャドー送信要求
4006 1090h	rUart_SBCR	シャドーブレークコントロールレジスタ
4006 1098h	rUart_SFE	シャドーFIFO 許可
4006 109Ch	rUart_SRT	シャドーRCVR トリガ
4006 10A0h	rUart_STET	シャドーTX エンプティトリガ
4006 10A4h	rUart_HTX	送信停止
4006 1100h	rUart_TO	タイムアウトカウンタコンフィグレーションレジスタ
4006 1104h	rUart_CTRLTO	タイムアウトコントロールレジスタ
4006 1108h	rUart_STATUSTO	タイムアウトカウンタステータスレジスタ

注1.　アドレスは、4006 1030h～4006 106Ch に割り当てられています。

1.3.3　レジスタマップ UART3

表 1.3　レジスタマップ UART3

アドレス	レジスタシンボル	レジスタ名
4006 2000h	（bUart_DLAB＝0）rUart_RBR_THR	受信バッファ／送信保持レジスタ
	（bUart_DLAB＝1）rUart_DLL	除数ラッチ（Low）
4006 2004h	（bUart_DLAB＝0）rUart_IER	割り込み許可レジスタ
	（bUart_DLAB＝1）rUart_DLH	除数ラッチ（High）
4006 2008h	（書き込み時）rUart_FCR	FIFO コントロールレジスタ
	（読み出し時）rUart_IIR	割り込み識別レジスタ
4006 200Ch	rUart_LCR	ラインコントロールレジスタ
4006 2010h	rUart_MCR	モデムコントロールレジスタ
4006 2014h	rUart_LSR	ラインステータスレジスタ
4006 2018h	rUart_MSR	モデムステータスレジスタ
4006 201Ch	rUart_SCR	スクラッチパッドレジスタ
4006 2030h	rUart_SRBR_STHR	シャドー受信バッファ／送信保持レジスタ注 1
4006 2070h	rUart_FAR	FIFO アクセスレジスタ
4006 2074h	rUart_TFR	送信 FIFO 読み出し
4006 2078h	rUart_RFW	受信 FIFO 書き込み
4006 207Ch	rUart_USR	UART ステータスレジスタ
4006 2080h	rUart_TFL	送信 FIFO レベル
4006 2084h	rUart_RFL	受信 FIFO レベル
4006 2088h	rUart_SRR	ソフトウェアリセットレジスタ
4006 208Ch	rUart_SRTS	シャドー送信要求
4006 2090h	rUart_SBCR	シャドーブレークコントロールレジスタ
4006 2098h	rUart_SFE	シャドーFIFO 許可
4006 209Ch	rUart_SRT	シャドーRCVR トリガ
4006 20A0h	rUart_STET	シャドーTX エンプティトリガ
4006 20A4h	rUart_HTX	送信停止
4006 2100h	rUart_TO	タイムアウトカウンタコンフィグレーションレジスタ
4006 2104h	rUart_CTRLTO	タイムアウトコントロールレジスタ
4006 2108h	rUart_STATUSTO	タイムアウトカウンタステータスレジスタ

注1.　アドレスは、4006 2030h～4006 206Ch に割り当てられています。

1.3.4　レジスタマップ UART4

表 1.4　レジスタマップ UART4

アドレス	レジスタシンボル	レジスタ名
5000 0000h	（bUart_DLAB＝0）rUart_RBR_THR	受信バッファ／送信保持レジスタ
	（bUart_DLAB＝1）rUart_DLL	除数ラッチ（Low）
5000 0004h	（bUart_DLAB＝0）rUart_IER	割り込み許可レジスタ
	（bUart_DLAB＝1）rUart_DLH	除数ラッチ（High）
5000 0008h	（書き込み時）rUart_FCR	FIFO コントロールレジスタ
	（読み出し時）rUart_IIR	割り込み識別レジスタ
5000 000Ch	rUart_LCR	ラインコントロールレジスタ
5000 0010h	rUart_MCR	モデムコントロールレジスタ
5000 0014h	rUart_LSR	ラインステータスレジスタ
5000 0018h	rUart_MSR	モデムステータスレジスタ
5000 001Ch	rUart_SCR	スクラッチパッドレジスタ
5000 0030h	rUart_SRBR_STHR	シャドー受信バッファ／送信保持レジスタ[注 1]
5000 0070h	rUart_FAR	FIFO アクセスレジスタ
5000 0074h	rUart_TFR	送信 FIFO 読み出し
5000 0078h	rUart_RFW	受信 FIFO 書き込み
5000 007Ch	rUart_USR	UART ステータスレジスタ
5000 0080h	rUart_TFL	送信 FIFO レベル
5000 0084h	rUart_RFL	受信 FIFO レベル
5000 0088h	rUart_SRR	ソフトウェアリセットレジスタ
5000 008Ch	rUart_SRTS	シャドー送信要求
5000 0090h	rUart_SBCR	シャドーブレークコントロールレジスタ
5000 0098h	rUart_SFE	シャドーFIFO 許可
5000 009Ch	rUart_SRT	シャドーRCVR トリガ
5000 00A0h	rUart_STET	シャドーTX エンプティトリガ
5000 00A4h	rUart_HTX	送信停止
5000 00A8h	rUart_DMASA	DMA ソフトウェアアクノリッジ
5000 0100h	rUart_TO	タイムアウトカウンタコンフィグレーションレジスタ
5000 0104h	rUart_CTRLTO	タイムアウトコントロールレジスタ
5000 0108h	rUart_STATUSTO	タイムアウトカウンタステータスレジスタ
5000 010Ch	rUart_TDMACR	DMA コントロールレジスタ（送信モード）
5000 0110h	rUart_RDMACR	DMA コントロールレジスタ（受信モード）

注1.　アドレスは、5000 0030h～5000 006Ch に割り当てられています。

1.3.5　レジスタマップ UART5

表 1.5　レジスタマップ UART5

アドレス	レジスタシンボル	レジスタ名
5000 1000h	（bUart_DLAB＝0）rUart_RBR_THR	受信バッファ／送信保持レジスタ
	（bUart_DLAB＝1）rUart_DLL	除数ラッチ（Low）
5000 1004h	（bUart_DLAB＝0）rUart_IER	割り込み許可レジスタ
	（bUart_DLAB＝1）rUart_DLH	除数ラッチ（High）
5000 1008h	（書き込み時）rUart_FCR	FIFO コントロールレジスタ
	（読み出し時）rUart_IIR	割り込み識別レジスタ
5000 100Ch	rUart_LCR	ラインコントロールレジスタ
5000 1010h	rUart_MCR	モデムコントロールレジスタ
5000 1014h	rUart_LSR	ラインステータスレジスタ
5000 1018h	rUart_MSR	モデムステータスレジスタ
5000 101Ch	rUart_SCR	スクラッチパッドレジスタ
5000 1030h	rUart_SRBR_STHR	シャドー受信バッファ／送信保持レジスタ注 1
5000 1070h	rUart_FAR	FIFO アクセスレジスタ
5000 1074h	rUart_TFR	送信 FIFO 読み出し
5000 1078h	rUart_RFW	受信 FIFO 書き込み
5000 107Ch	rUart_USR	UART ステータスレジスタ
5000 1080h	rUart_TFL	送信 FIFO レベル
5000 1084h	rUart_RFL	受信 FIFO レベル
5000 1088h	rUart_SRR	ソフトウェアリセットレジスタ
5000 108Ch	rUart_SRTS	シャドー送信要求
5000 1090h	rUart_SBCR	シャドーブレークコントロールレジスタ
5000 1098h	rUart_SFE	シャドーFIFO 許可
5000 109Ch	rUart_SRT	シャドーRCVR トリガ
5000 10A0h	rUart_STET	シャドーTX エンプティトリガ
5000 10A4h	rUart_HTX	送信停止
5000 10A8h	rUart_DMASA	DMA ソフトウェアアクノリッジ
5000 1100h	rUart_TO	タイムアウトカウンタコンフィグレーションレジスタ
5000 1104h	rUart_CTRLTO	タイムアウトコントロールレジスタ
5000 1108h	rUart_STATUSTO	タイムアウトカウンタステータスレジスタ
5000 110Ch	rUart_TDMACR	DMA コントロールレジスタ（送信モード）
5000 1110h	rUart_RDMACR	DMA コントロールレジスタ（受信モード）

注1.　アドレスは、5000 1030h～5000 106Ch に割り当てられています。

1.3.6　レジスタマップ UART6

表 1.6　レジスタマップ UART6

アドレス	レジスタシンボル	レジスタ名
5000 2000h	（bUart_DLAB＝0）rUart_RBR_THR	受信バッファ／送信保持レジスタ
	（bUart_DLAB＝1）rUart_DLL	除数ラッチ（Low）
5000 2004h	（bUart_DLAB＝0）rUart_IER	割り込み許可レジスタ
	（bUart_DLAB＝1）rUart_DLH	除数ラッチ（High）
5000 2008h	（書き込み時）rUart_FCR	FIFO コントロールレジスタ
	（読み出し時）rUart_IIR	割り込み識別レジスタ
5000 200Ch	rUart_LCR	ラインコントロールレジスタ
5000 2010h	rUart_MCR	モデムコントロールレジスタ
5000 2014h	rUart_LSR	ラインステータスレジスタ
5000 2018h	rUart_MSR	モデムステータスレジスタ
5000 201Ch	rUart_SCR	スクラッチパッドレジスタ
5000 2030h	rUart_SRBR_STHR	シャドー受信バッファ／送信保持レジスタ注 1
5000 2070h	rUart_FAR	FIFO アクセスレジスタ
5000 2074h	rUart_TFR	送信 FIFO 読み出し
5000 2078h	rUart_RFW	受信 FIFO 書き込み
5000 207Ch	rUart_USR	UART ステータスレジスタ
5000 2080h	rUart_TFL	送信 FIFO レベル
5000 2084h	rUart_RFL	受信 FIFO レベル
5000 2088h	rUart_SRR	ソフトウェアリセットレジスタ
5000 208Ch	rUart_SRTS	シャドー送信要求
5000 2090h	rUart_SBCR	シャドーブレークコントロールレジスタ
5000 2098h	rUart_SFE	シャドーFIFO 許可
5000 209Ch	rUart_SRT	シャドーRCVR トリガ
5000 20A0h	rUart_STET	シャドーTX エンプティトリガ
5000 20A4h	rUart_HTX	送信停止
5000 20A8h	rUart_DMASA	DMA ソフトウェアアクノリッジ
5000 2100h	rUart_TO	タイムアウトカウンタコンフィグレーションレジスタ
5000 2104h	rUart_CTRLTO	タイムアウトコントロールレジスタ
5000 2108h	rUart_STATUSTO	タイムアウトカウンタステータスレジスタ
5000 210Ch	rUart_TDMACR	DMA コントロールレジスタ（送信モード）
5000 2110h	rUart_RDMACR	DMA コントロールレジスタ（受信モード）

注1.　アドレスは、5000 2030h～5000 206Ch に割り当てられています。

1.3.7　レジスタマップ UART7

表 1.7　レジスタマップ UART7

アドレス	レジスタシンボル	レジスタ名
5000 3000h	（bUart_DLAB＝0）rUart_RBR_THR	受信バッファ／送信保持レジスタ
	（bUart_DLAB＝1）rUart_DLL	除数ラッチ（Low）
5000 3004h	（bUart_DLAB＝0）rUart_IER	割り込み許可レジスタ
	（bUart_DLAB＝1）rUart_DLH	除数ラッチ（High）
5000 3008h	（書き込み時）rUart_FCR	FIFO コントロールレジスタ
	（読み出し時）rUart_IIR	割り込み識別レジスタ
5000 300Ch	rUart_LCR	ラインコントロールレジスタ
5000 3010h	rUart_MCR	モデムコントロールレジスタ
5000 3014h	rUart_LSR	ラインステータスレジスタ
5000 3018h	rUart_MSR	モデムステータスレジスタ
5000 301Ch	rUart_SCR	スクラッチパッドレジスタ
5000 3030h	rUart_SRBR_STHR	シャドー受信バッファ／送信保持レジスタ注 1
5000 3070h	rUart_FAR	FIFO アクセスレジスタ
5000 3074h	rUart_TFR	送信 FIFO 読み出し
5000 3078h	rUart_RFW	受信 FIFO 書き込み
5000 307Ch	rUart_USR	UART ステータスレジスタ
5000 3080h	rUart_TFL	送信 FIFO レベル
5000 3084h	rUart_RFL	受信 FIFO レベル
5000 3088h	rUart_SRR	ソフトウェアリセットレジスタ
5000 308Ch	rUart_SRTS	シャドー送信要求
5000 3090h	rUart_SBCR	シャドーブレークコントロールレジスタ
5000 3098h	rUart_SFE	シャドーFIFO 許可
5000 309Ch	rUart_SRT	シャドーRCVR トリガ
5000 30A0h	rUart_STET	シャドーTX エンプティトリガ
5000 30A4h	rUart_HTX	送信停止
5000 30A8h	rUart_DMASA	DMA ソフトウェアアクノリッジ
5000 3100h	rUart_TO	タイムアウトカウンタコンフィグレーションレジスタ
5000 3104h	rUart_CTRLTO	タイムアウトコントロールレジスタ
5000 3108h	rUart_STATUSTO	タイムアウトカウンタステータスレジスタ
5000 310Ch	rUart_TDMACR	DMA コントロールレジスタ（送信モード）
5000 3110h	rUart_RDMACR	DMA コントロールレジスタ（受信モード）

注1.　アドレスは、5000 3030h～5000 306Ch に割り当てられています。

1.3.8　レジスタマップ UART8

表 1.8　レジスタマップ UART8

アドレス	レジスタシンボル	レジスタ名
5000 4000h	（bUart_DLAB＝0）rUart_RBR_THR	受信バッファ／送信保持レジスタ
	（bUart_DLAB＝1）rUart_DLL	除数ラッチ（Low）
5000 4004h	（bUart_DLAB＝0）rUart_IER	割り込み許可レジスタ
	（bUart_DLAB＝1）rUart_DLH	除数ラッチ（High）
5000 4008h	（書き込み時）rUart_FCR	FIFO コントロールレジスタ
	（読み出し時）rUart_IIR	割り込み識別レジスタ
5000 400Ch	rUart_LCR	ラインコントロールレジスタ
5000 4010h	rUart_MCR	モデムコントロールレジスタ
5000 4014h	rUart_LSR	ラインステータスレジスタ
5000 4018h	rUart_MSR	モデムステータスレジスタ
5000 401Ch	rUart_SCR	スクラッチパッドレジスタ
5000 4030h	rUart_SRBR_STHR	シャドー受信バッファ／送信保持レジスタ注 1
5000 4070h	rUart_FAR	FIFO アクセスレジスタ
5000 4074h	rUart_TFR	送信 FIFO 読み出し
5000 4078h	rUart_RFW	受信 FIFO 書き込み
5000 407Ch	rUart_USR	UART ステータスレジスタ
5000 4080h	rUart_TFL	送信 FIFO レベル
5000 4084h	rUart_RFL	受信 FIFO レベル
5000 4088h	rUart_SRR	ソフトウェアリセットレジスタ
5000 408Ch	rUart_SRTS	シャドー送信要求
5000 4090h	rUart_SBCR	シャドーブレークコントロールレジスタ
5000 4098h	rUart_SFE	シャドーFIFO 許可
5000 409Ch	rUart_SRT	シャドーRCVR トリガ
5000 40A0h	rUart_STET	シャドーTX エンプティトリガ
5000 40A4h	rUart_HTX	送信停止
5000 40A8h	rUart_DMASA	DMA ソフトウェアアクノリッジ
5000 4100h	rUart_TO	タイムアウトカウンタコンフィグレーションレジスタ
5000 4104h	rUart_CTRLTO	タイムアウトコントロールレジスタ
5000 4108h	rUart_STATUSTO	タイムアウトカウンタステータスレジスタ
5000 410Ch	rUart_TDMACR	DMA コントロールレジスタ（送信モード）
5000 4110h	rUart_RDMACR	DMA コントロールレジスタ（受信モード）

注1.　アドレスは、5000 4030h～5000 406Ch に割り当てられています。

1.4　レジスタの説明

1.4.1　rUart_DLL ― 除数ラッチ（Low）

- 依存関係：bUart_DLAB ビット＝1

アドレス　4006 0000h（UART1）
4006 1000h（UART2）
4006 2000h（UART3）
5000 0000h（UART4）
5000 1000h（UART5）
5000 2000h（UART6）
5000 3000h（UART7）
5000 4000h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_DLL							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.9　rUart_DLL レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	bUart_DLL	16 ビットの下位 8 ビット UART 用のボーレート除数を示す除数ラッチレジスタ 本レジスタは、（rUart_LCR レジスタの）bUart_DLAB ビットがセットされ、UART がビジーでなく、（rUart_USR レジスタの）bUart_BUSY ビットが 0 の場合のみアクセス可能です。 ボークロックは、以下のように、UART_SCLK をボーレート除数値×16 で分周したものです。 ボークロック＝UART_SCLK／（16×ボーレート除数） **備考**　ボーレート除数（bUart_DLL および bUart_DLH）が 0 の場合、ボークロックは無効で、シリアル通信は実行されません。 **注意**　いったん bUart_DLL または bUart_DLH を設定した場合、最も遅いクロックの少なくとも 8 クロックサイクル分経過してからデータの送受信を行ってください。	R/W

1.4.2　rUart_DLH ― 除数ラッチ（High）

- 依存関係：bUart_DLAB ビット＝1

アドレス　4006 0004h（UART1）
4006 1004h（UART2）
4006 2004h（UART3）
5000 0004h（UART4）
5000 1004h（UART5）
5000 2004h（UART6）
5000 3004h（UART7）
5000 4004h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_DLH							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.10　rUart_DLH レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	bUart_DLH	16 ビットの上位 8 ビット UART 用のボーレート除数を示す除数ラッチレジスタ 本レジスタは、（rUart_LCR レジスタの）bUart_DLAB ビットがセットされ、UART がビジーでなく、（rUart_USR レジスタの）bUart_BUSY ビットが 0 の場合のみアクセス可能です。 ボークロックは、「**1.4.1 rUart_DLL ― 除数ラッチ（Low）**」を参照してください。	R/W

1.4.3　rUart_IIR ― 割り込み識別レジスタ

● 以下アドレス読み出し時

アドレス　4006 0008h（UART1）
4006 1008h（UART2）
4006 2008h（UART3）
5000 0008h（UART4）
5000 1008h（UART5）
5000 2008h（UART6）
5000 3008h（UART7）
5000 4008h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_FIFOSE		―	―	bUart_IID			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

表 1.11　rUart_IIR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7、b6	bUart_FIFOSE	FIFO 許可 FIFO が許可か禁止かを示します。 2’b00＝禁止 2’b11＝許可	R
b5、b4	予約ビット		R
b3～b0	bUart_IID	割り込み ID 最優先保留割り込みを示します。割り込みには以下の種別があります。 4’b0000＝モデム状態 4’b0001＝保留中の割り込みなし 4’b0010＝THR エンプティ 4’b0100＝受信データあり 4’b0101＝レシーバタイムアウト 4’b0110＝レシーバラインステータス 4’b0111＝ビジー検出 4’b1100＝キャラクタタイムアウト 割り込みの優先順位は、6 つのレベルに分けられます（詳細については、「**表 1.41 割り込み制御機能**」参照）。 **備考**　bUart_IID のビット[3]は、FIFO が許可されている場合のみ割り込みが発生することを示し、キャラクタタイムアウト条件割り込みを識別する際に使用されます。	R

1.4.4　rUart_RBR_THR ― 受信バッファ／送信保持レジスタ

- 依存関係：bUart_DLAB ビット＝0

アドレス　4006 0000h（UART1）
4006 1000h（UART2）
4006 2000h（UART3）
5000 0000h（UART4）
5000 1000h（UART5）
5000 2000h（UART6）
5000 3000h（UART7）
5000 4000h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_RBR_THR							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.12　rUart_RBR_THR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	bUart_RBR_THR	本レジスタ読み出し時：受信バッファレジスタ rUart_RBR ● シリアル入力ポート UART_RXD で受信したデータバイト 本レジスタのデータは、ラインステータスレジスタ（rUart_LSR）のデータ準備完了（bUart_DR）ビットがセットされている場合のみ有効です。 ● FIFO が禁止されている場合（bUart_FIFOE＝0）、rUart_RBR のデータは、次のデータが到着する前に読み出す必要があります。読み出されない場合、オーバーランエラーになります。 ● FIFO が許可されている場合（bUart_FIFOE＝1）、本レジスタは、受信 FIFO の先頭にアクセスします。受信 FIFO がフルで、本レジスタが次のデータキャラクタ到着前に読み出されない場合は、FIFO 内にすでにあるデータが保持され、着信データは失われ、オーバーランエラーが発生します。 本レジスタ書き込み時：送信保持レジスタ rUart_THR ● シリアル出力ポート UART_TXD で送信するデータ ● FIFO が禁止されており（bUart_FIFOE＝0）、bUart_THRE がセットされている時に rUart_THR に 1 キャラクタ書き込むと、bUart_THRE がクリアされます。bUart_THRE が再びセットされる前に rUart_THR に追加書き込みがあると、rUart_THR データは上書きされます。 ● FIFO が許可されており（bUart_FIFOE＝1）、bUart_THRE がセットされている場合、FIFO がフルになる前に 16 キャラクタのデータを rUart_THR に書き込み可能です。FIFO がフルの時に更にデータを書き込もうとすると書き込みデータは失われます。「**1.5.1.8　プログラマブル THRE 割り込み**」を参照してください。	R/W

1.4.5　rUart_IER ― 割り込み許可レジスタ

- 依存関係：bUart_DLAB ビット＝0

アドレス　4006 0004h（UART1）
4006 1004h（UART2）
4006 2004h（UART3）
5000 0004h（UART4）
5000 1004h（UART5）
5000 2004h（UART6）
5000 3004h（UART7）
5000 4004h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bUart_ETIMEOUT3	bUart_ETIMEOUT2	bUart_ETIMEOUT1	bUart_ETIMEOUT0	bUart_PTIME	―	―	―	bUart_EDSSI	bUart_ELSI	bUart_ETBEI	bUart_ERBFI
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.13　rUart_IER レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b12	予約ビット		R
b11	bUart_ETIMEOUT3	レシーバまたはトランシーバタイムアウト n（n＝0～3）割り込みを許可 トランシーバタイムアウト n（n＝3）を許可 本ビットは、レシーバ（n＝0、1）またはトランシーバタイムアウト n（n＝2、3）割り込みを許可／禁止する際に使用されます。 これは、優先順位 6 番目の割り込みになります。 各タイムアウト n（n＝0～3）につき、以下の設定となります。 1’b0＝禁止 1’b1＝許可 「**1.5.1.10(1) レシーバタイムアウト**」を参照してください。 「**1.5.1.10(2) トランシーバタイムアウト**」を参照してください。	R/W
b10	bUart_ETIMEOUT2	トランシーバタイムアウト n（n＝2）を許可 詳細については上記を参照してください。	R/W
b9	bUart_ETIMEOUT1	レシーバタイムアウト n（n＝1）を許可 詳細については上記を参照してください。	R/W
b8	bUart_ETIMEOUT0	レシーバタイムアウト n（n＝0）を許可 詳細については上記を参照してください。	R/W
b7	bUart_PTIME	プログラマブル THRE 割り込みモード許可 本ビットは、THRE 割り込み発生を許可／禁止する際に使用されます。 1’b0＝禁止 1’b1＝許可 「**1.5.1.8 プログラマブル THRE 割り込み**」を参照してください。	R/W
b6～b4	予約ビット		R
b3	bUart_EDSSI	モデム状態割り込みを許可 本ビットは、モデム状態割り込み発生を許可／禁止する際に使用されます。 これは、優先順位 4 番目の割り込みになります。 1’b0＝禁止 1’b1＝許可	R/W

表 1.13　rUart_IER レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b2	bUart_ELSI	レシーバラインステータス割り込み許可 本ビットは、レシーバラインステータス割り込み発生を許可／禁止する際に使用されます。 これは、最優先の割り込みになります。 1'b0＝禁止 1'b1＝許可	R/W
b1	bUart_ETBEI	送信保持レジスタエンプティ割り込み許可 本ビットは、送信保持レジスタエンプティ割り込み発生を許可／禁止する際に使用されます。 これは、優先順位 3 番目の割り込みになります。 1'b0＝禁止 1'b1＝許可	R/W
b0	bUart_ERBFI	受信データあり割り込み許可 本ビットは、受信データあり割り込みおよびキャラクタタイムアウト割り込み（FIFO 許可の場合）を許可／禁止する際に使用されます。 これは、優先順位 2 番目の割り込みになります。 1'b0＝禁止 1'b1＝許可	R/W

1.4.6　rUart_FCR ― FIFO コントロールレジスタ

● 以下アドレス書き込み時

アドレス　4006 0008h（UART1）
4006 1008h（UART2）
4006 2008h（UART3）
5000 0008h（UART4）
5000 1008h（UART5）
5000 2008h（UART6）
5000 3008h（UART7）
5000 4008h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_RCVR		bUart_TET		―	bUart_XFIFOR	bUart_RFIFOR	bUart_FIFOE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.14　rUart_FCR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7、b6	bUart_RCVR	受信 FIFO トリガ 本ビットは、受信データあり割り込みが発生する受信 FIFO のトリガレベルを選択する際に使用されます。 以下のトリガレベルがサポートされています。 2’b00＝FIFO 内に 1 キャラクタ 2’b01＝FIFO 1/4 フル 2’b10＝FIFO 1/2 フル 2’b11＝FIFO フルまであと 2 キャラクタ	W
b5、b4	bUart_TET	送信 FIFO エンプティトリガ 本ビットは、THRE 割り込みが発生するエンプティしきい値レベルを指定する際に使用されます。 以下のトリガレベルがサポートされています。 2’b00＝FIFO エンプティ 2’b01＝FIFO 内に 2 キャラクタ 2’b10＝FIFO 1/4 フル 2’b11＝FIFO 1/2 フル	W
b3	予約ビット		R
b2	bUart_XFIFOR	送信 FIFO リセット 本ビットに 1 をセットすると、送信 FIFO の制御部分をリセットし、FIFO をエンプティとして処理します。 **備考**）　本ビットは、「自己クリア」します。本ビットをクリアする必要はありません。	W
b1	bUart_RFIFOR	受信 FIFO リセット 本ビットに 1 をセットすると、受信 FIFO の制御部分をリセットし、FIFO をエンプティとして処理します。 **備考**）　本ビットは、「自己クリア」します。本ビットをクリアする必要はありません。	W

表 1.14　rUart_FCR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bUart_FIFOE	FIFO 許可 本ビットは、送信 FIFO および受信 FIFO を許可／禁止します。 本ビットの値が変わると FIFO の送受信制御部分がリセットされます。 1'b0＝禁止 1'b1＝許可	W

1.4.7　rUart_LCR ― ラインコントロールレジスタ

アドレス　4006 000Ch（UART1）
4006 100Ch（UART2）
4006 200Ch（UART3）
5000 000Ch（UART4）
5000 100Ch（UART5）
5000 200Ch（UART6）
5000 300Ch（UART7）
5000 400Ch（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_DLAB	bUart_BC	bUart_StickParity	bUart_EPS	bUart_PEN	bUart_STOP	bUart_DLS	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.15　rUart_LCR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7	bUart_DLAB	除数ラッチアクセスビット UART がビジー以外のとき（bUart_BUSY＝0）のみ書き込み可能です。 本ビットは、ボーレート除数（bUart_DLL および bUart_DLH）の読み出しおよび書き込みを許可し、UART のボーレートを設定する際に使用されます。 本ビットは、他のレジスタにアクセスするため、ボーレートの初期設定後にクリアする必要があります。 1’b0＝除数ラッチアクセス無効 1’b1＝除数ラッチアクセス有効	R/W
b6	bUart_BC	ブレークコントロールビット 本ビットは、ブレーク条件を受信デバイスに送る際に使用されます。 1 の時、シリアル出力が強制的にスペース（論理 0）状態にされます。bUart_LB によるループバックモード以外では、UART_TXD ラインは、bUart_BC がクリアされるまで Low のままです。 ループバックモード時は、ブレーク条件はレシーバに内部ループバックされます。	R/W
b5	bUart_StickParity	スティックパリティビット UART がビジー以外のとき（bUart_BUSY＝0）のみ書き込み可能です。 本ビットに 1 をセットすると、パリティ有効時（bUart_PEN＝1）に、パリティビットを bUart_EPS の反転した値として、送信時はパリティの付加、受信時はチェックをします。 bUart_EPS＝1：パリティビット値 0 bUart_EPS＝0：パリティビット値 1	R/W
b4	bUart_EPS	偶数パリティ選択 UART がビジー以外のとき（bUart_BUSY＝0）のみ書き込み可能です。 本ビットは、パリティが有効にされているとき（bUart_PEN＝1）、偶数パリティか奇数パリティかを選択する際に使用されます。 偶数パリティでは、パリティ値と送受信キャラクタを合わせて、1 の数の合計が偶数個になるようにパリティ値を決定します。 1’b0＝奇数パリティ 1’b1＝偶数パリティ	R/W

表 1.15　rUart_LCR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3	bUart_PEN	パリティ有効 UART がビジー以外のとき（bUart_BUSY＝0）のみ書き込み可能です。 本ビットは、それぞれ送信および受信シリアルキャラクタ内でパリティの生成および検出を有効／無効する際に使用されます。 1’b0＝パリティ無効 1’b1＝パリティ有効	R/W
b2	bUart_STOP	ストップビット数 UART がビジー以外のとき（bUart_BUSY＝0）のみ書き込み可能です。 本ビットは、周辺デバイスが送受信する 1 キャラクタあたりのストップビット数を選択する際に使用されます。 ● 0 の場合、1 個のストップビットがシリアルデータ内で送信されます。 ● 1 の場合で、データビットが 5 に設定されている（bUart_DLS＝0）場合、1.5 個のストップビットが送信されます。これ以外の場合、2 個のストップビットが送信されます。 選択されたストップビット数に関係なく、レシーバは最初のストップビットのみをチェックすることに注意してください。 1’b0＝1 ストップビット 1’b1＝bUart_DLS が 0 の場合 1.5 ストップビット。それ以外は 2 ストップビット **備考）**設定および送信方向のボーレート除数値により挿入されるアイドル時間のために、ストップビット期間はより長く見える場合があります。送信転送間のアイドル時間の詳細については、「**1.5.1.5 連続キャラクタストリーム送信**」を参照してください。	R/W
b1、b0	bUart_DLS	データ長選択 UART がビジー以外のとき（bUart_BUSY＝0）のみ書き込み可能です。 本ビットは、周辺デバイスが送受信する 1 キャラクタあたりのデータビット数を選択する際に使用されます。 選択可能なビット数は以下のとおりです。 2’b00＝5 ビット 2’b01＝6 ビット 2’b10＝7 ビット 2’b11＝8 ビット	R/W

1.4.8　rUart_MCR ― モデムコントロールレジスタ

アドレス　4006 0010h（UART1）
4006 1010h（UART2）
4006 2010h（UART3）
5000 0010h（UART4）
5000 1010h（UART5）
5000 2010h（UART6）
5000 3010h（UART7）
5000 4010h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	bUart_AFCE	bUart_LB	bUart_OUT2	bUart_OUT1	bUart_RTS	bUart_DTR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.16　rUart_MCR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b6	予約ビット		R
b5	bUart_AFCE	自動フロー制御許可 FIFO が許可されており（bUart_FIFOE ビットがセット）、自動フロー制御イネーブル（bUart_AFCE ビットがセット）の場合、自動フロー制御機能が、「**1.5.1.7 自動フロー制御**」に記載のとおり、有効になります。 1’b0＝自動フロー制御モード禁止 1’b1＝自動フロー制御モード許可	R/W
b4	bUart_LB	ループバックビット 本ビットは、テストのため UART を診断モードにする際に使用されます。 シリアルデータが UART_RXD ラインに内部的にループバックされる一方、UART_TXD ラインのデータは High に保持されます。本モード中、すべての割り込みは完全に機能します。また、ループバックモードでは、モデム制御入力（UART_DSR_N、UART_CTS_N、UART_RI_N、UART_DCD_N）は切断され、モデム制御出力（UART_DTR_N、UART_RTS_N、UART_OUT1_N、UART_OUT2_N）は内部的に入力にループバックされます。 1’b0＝ループバックモード無効 1’b1＝ループバックモード有効	R/W
b3	bUart_OUT2	本ビットは、ユーザ指定の Output2（UART_OUT2_N）出力を直接制御する際に使用されます。 この位置に書き込まれた値は反転され、以下のように UART_OUT2_N に出力されます。 1’b0＝UART_OUT2_N デアサート（論理 1） 1’b1＝UART_OUT2_N アサート（論理 0） **注意**　本出力端子は、ピン配列には接続されていません。ループバックモードで使用されます。 **備考**　ループバックモード（bUart_LB＝1）時、UART_OUT2_N 出力は非アクティブな High に保持され、本位置の値が内部的に入力にループバックされます。	R/W

表 1.16　rUart_MCR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b2	bUart_OUT1	本ビットは、ユーザ指定の Output1（UART_OUT1_N）出力を直接制御する際に使用されます。 この位置に書き込まれた値は反転され、以下のように UART_OUT1_N に出力されます。 1'b0＝UART_OUT1_N デアサート（論理 1） 1'b1＝UART_OUT1_N アサート（論理 0） **注意**） 本出力端子は、ピン配列には接続されていません。ループバックモードで使用されます。 **備考**） ループバックモード（bUart_LB＝1）時、UART_OUT1_N 出力は非アクティブな High に保持され、本位置の値が内部的に入力にループバックされます。	R/W
b1	bUart_RTS	送信要求 本ビットは、送信要求（UART_RTS_N）出力を直接制御する際に使用されます。 送信要求（UART_RTS_N）出力は、モデムまたはデータセットに、UART がデータを交換する準備ができていることを知らせる際に使用されます。 自動 RTS フロー制御が有効でない（bUart_AFCE＝0）場合、bUart_RTS を High にプログラムすることにより UART_RTS_N 信号が Low に設定されます。 自動フロー制御（bUart_AFCE＝1）かつ FIFO 許可（bUart_FIFOE＝1）のとき、UART_RTS_N 出力は同じように制御されますが、「受信 FIFO 準フルトリガ」によりゲートされます。ここで「準フル」とは、FIFO に利用可能なスロットが 2 つあることを指します（しきい値を越えると UART_RTS_N は非アクティブな High になります）。 bUart_RTS を Low クリアすると UART_RTS_N 信号はデアサートされます。 「**1.5.1.7 自動フロー制御**」を参照してください。 **備考**） ループバックモード（bUart_LB＝1）時、UART_RTS_N 出力は非アクティブな High に保持され、本位置の値が内部的に入力にループバックされます。	R/W
b0	bUart_DTR	データ端末準備完了 本ビットは、データ端末準備完了（UART_DTR_N）出力を直接制御する際に使用されます。 この位置に書き込まれた値は反転され、以下のように UART_DTR_N に出力されます。 1'b0＝UART_DTR_N デアサート（論理 1） 1'b1＝UART_DTR_N アサート（論理 0） データ端末準備完了出力は、モデムまたはデータセットに、UART が通信を確立する準備ができていることを知らせる際に使用されます。 **備考**） ループバックモード（bUart_LB＝1）時、UART_DTR_N 出力は非アクティブな High に保持され、本位置の値が内部的に入力にループバックされます。	R/W

1.4.9　rUart_LSR ― ラインステータスレジスタ

アドレス　4006 0014h（UART1）
4006 1014h（UART2）
4006 2014h（UART3）
5000 0014h（UART4）
5000 1014h（UART5）
5000 2014h（UART6）
5000 3014h（UART7）
5000 4014h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_RFE	bUart_TEMT	bUart_THRE	bUart_BI	bUart_FE	bUart_PE	bUart_OE	bUart_DR
リセット後の値	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

表 1.17　rUart_LSR レジスタの内容（1/3）

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7	bUart_RFE	レシーバ FIFO エラービット 本ビットは FIFO が許可されている時（bUart_FIFOE＝1）のみ関係します。 受信 FIFO に少なくとも 1 つのパリティエラー、フレーミングエラー、またはブレーク表示があることを示すために使用されます。 1’b0＝受信 FIFO にエラーなし 1’b1＝受信 FIFO にエラーあり rUart_LSR が読み出され、エラーのあるキャラクタが受信 FIFO の先頭にあり、FIFO にそれ以降エラーがない場合、本ビットはクリアされます。	R
b6	bUart_TEMT	トランスミッタエンプティビット FIFO 許可（bUart_FIFOE＝1）時、本ビットは、トランスミッタシフトレジスタと送信 FIFO の両方がエンプティのときセットされます。 FIFO 禁止時、本ビットは、トランスミット保持レジスタ（rUart_THR）とトランスミッタシフトレジスタの両方がエンプティのときセットされます。	R
b5	bUart_THRE	送信保持レジスタエンプティビット THRE モード無効（bUart_PTIME＝0）時、本ビットは FIFO が許可されているかどうかに関係なく rUart_THR または送信 FIFO がエンプティであることを示します。 本ビットは、データが rUart_THR または送信 FIFO からトランスミッタシフトレジスタに転送され、rUart_THR または送信 FIFO に新しいデータが書き込まれていない場合セットされます。 これは、また、THRE 割り込みが有効な場合（bUart_ETBEI）、THRE 割り込みを発生させます。 THRE モードで FIFO が許可されている（bUart_PTIME＝1 かつ bUart_FIFOE＝1）場合、機能は切り替えられ、送信 FIFO がフルで、もはや THRE 割り込みを制御していないことを示します。この場合、THRE 割り込みは bUart_TET しきい値設定で制御されます。 詳細は「**1.5.1.8　プログラマブル THRE 割り込み**」を参照してください。	R

表 1.17　rUart_LSR レジスタの内容（2/3）

ビット位置	ビット名	機能	R/W
b4	bUart_BI	ブレーク割り込みビット 本ビットは、シリアル入力データにブレークシーケンスが検出されたことを示すために使用されます。 シリアル入力である UART_RXD がスタート時間＋データビット＋パリティ＋ストップビットの合計時間より長く論理“0”の状態に保持された場合にセットされます。 シリアル入力のブレーク条件により、すべて 0 から成る 1 キャラクタを UART が受信します。 FIFO モード中（bUart_FIFOE＝1）、ブレーク条件に関連するキャラクタは、FIFO 内を移動し、そのキャラクタが FIFO の先頭に来ると明らかになります。 rUart_LSR を読み出すと bUart_BI ビットがクリアされます。 非 FIFO モードでは、bUart_BI 表示はただちに発生し、rUart_LSR が読み出されるまで継続します。 **備考**　ブレーク条件受信時 FIFO がフルの場合、FIFO オーバーランエラーが発生します。ブレーク条件およびブレーク条件に関連するすべての情報（パリティエラーおよびフレーミングエラー）が廃棄され、ブレークキャラクタが受信されたという情報はなくなります。	R
b3	bUart_FE	フレーミングエラービット 本ビットは、レシーバ内でフレーミングエラーが発生したことを示すために使用されます。レシーバが受信データ内で有効なストップビットを検出しない場合、フレーミングエラーが発生します。 FIFO モード（bUart_FIFOE＝1）では、フレーミングエラーは受信したキャラクタに関連しているので、フレーミングエラーがあるキャラクタが FIFO の先頭に来たときに明らかになります。 フレーミングエラーが発生すると、UART は再同期を試行します。UART はエラーが次のキャラクタのスタートビットによるものであると判断し、それ以外のビット（データ、および／またはパリティおよびストップ）の受信を継続することにより再同期を試行します。 ただし、ブレーク割り込み bUart_BI ビットで示されるブレーク割り込みが発生すると、フレーミングエラーbUart_FE ビットがセットされます。これは、キャラクタの期間よりも長く UART_RXD 入力を論理 0 に保持することによりブレークキャラクタが暗黙的にフレーミングエラーを発生させるために起こります。 1’b0＝フレーミングエラーなし 1’b1＝フレーミングエラーあり rUart_LSR を読み出すと rUart_FE ビットがクリアされます。	R
b2	bUart_PE	パリティエラービット パリティ有効 bUart_PEN ビットがセットされている場合、本ビットは、レシーバでパリティエラーが発生したことを示すために使用されます。 FIFO モード（bUart_FIFOE＝1）では、パリティエラーは受信したキャラクタに関連しているので、パリティエラーがあるキャラクタが FIFO の先頭に来たときに明らかになります。 ただし、ブレーク割り込み bUart_BI ビットで示されるブレーク割り込みが発生すると、パリティ生成および検出が有効（bUart_PEN＝1）で、パリティが奇数に設定（bUart_EPS＝0）されている場合、パリティエラービットがセットされます。 1’b0＝パリティエラーなし 1’b1＝パリティエラーあり rUart_LSR を読み出すと bUart_PE ビットがクリアされます。	R

表 1.17　rUart_LSR レジスタの内容（3/3）

ビット位置	ビット名	機能	R/W
b1	bUart_OE	オーバーランエラービット 本ビットは、オーバーランエラーが発生したことを示すために使用されます。本エラーは、前のデータが読み出される前に新しいデータキャラクタを受信した場合に発生します。 非 FIFO モード（bUart_FIFOE＝0）では、前のデータが rUart_RBR から読み出される前にレシーバに新しいキャラクタが到着すると、bUart_OE ビットがセットされます。このとき、rUart_RBR 内のデータは上書きされます。 FIFO モードでは、FIFO がフルのとき新しいキャラクタがレシーバに到着するとオーバーランエラーが発生します。FIFO 内のデータは保持され、レシーバシフトレジスタ内のデータは失われます。 1’b0＝オーバーランエラーなし 1’b1＝オーバーランエラーあり rUart_LSR を読み出すと bUart_OE ビットがクリアされます。	R
b0	bUart_DR	データ準備完了ビット 本ビットは、レシーバの rUart_RBR または受信 FIFO 内に少なくとも 1 キャラクタがあることを示します。 1’b0＝データ準備未完了 1’b1＝データ準備完了 本ビットは、非 FIFO モードで rUart_RBR が読み出されるもしくは FIFO モードで受信 FIFO がエンプティの場合クリアされます。	R

1.4.10　rUart_MSR ― モデムステータスレジスタ

アドレス　4006 0018h（UART1）
4006 1018h（UART2）
4006 2018h（UART3）
5000 0018h（UART4）
5000 1018h（UART5）
5000 2018h（UART6）
5000 3018h（UART7）
5000 4018h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_DCD	bUart_RI	bUart_DSR	bUart_CTS	bUart_DDCD	bUart_TERI	bUart_DDSR	bUart_DCTS
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.18　rUart_MSR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7	bUart_DCD	データキャリア検出 本ビットは、モデム制御ライン UART_DCD_N の現在の状態を示すために使用されます。 本ビットは、UART_DCD_N の反転になります。 データキャリア検出入力（UART_DCD_N）のアサートは、キャリアがモデムまたはデータセットにより検出されたことを示します。 1'b0＝UART_DCD_N 入力のデアサート（論理 1） 1'b1＝UART_DCD_N 入力のアサート（論理 0） ループバックモード時（bUart_LB＝1）、bUart_DCD は bUart_OUT2 と同じになります。	R
b6	bUart_RI	被呼表示 本ビットは、モデム制御ライン UART_RI_N の現在の状態を示すために使用されます。本ビットは、UART_RI_N の反転になります。 被呼表示入力（UART_RI_N）のアサートは、電話呼び出し信号がモデムまたはデータセットにより受信されたことを示します。 1'b0＝UART_RI_N 入力のデアサート（論理 1） 1'b1＝UART_RI_N 入力のアサート（論理 0） ループバックモード時（bUart_LB＝1）、bUart_RI は bUart_OUT1 と同じになります。	R
b5	bUart_DSR	データセットレディ 本ビットは、モデム制御ライン UART_DSR_N の現在の状態を示すために使用されます。本ビットは、UART_DSR_N の反転になります。 データセットレディ入力（UART_DSR_N）のアサートは、モデムまたはデータセットが UART と通信を確立する準備ができていることを示します。 1'b0＝UART_DSR_N 入力のデアサート（論理 1） 1'b1＝UART_DSR_N 入力のアサート（論理 0） ループバックモード時（bUart_LB＝1）、bUart_DSR は bUart_DTR と同じになります。	R

表 1.18　rUart_MSR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b4	bUart_CTS	送信可 本ビットは、モデム制御ライン UART_CTS_N の現在の状態を示すために使用されます。本ビットは、UART_CTS_N の反転になります。 送信可入力（UART_CTS_N）のアサートは、モデムまたはデータセットが UART とデータを交換する準備ができていることを示します。 1'b0＝UART_CTS_N 入力のデアサート（論理 1） 1'b1＝UART_CTS_N 入力のアサート（論理 0） ループバックモード時（bUart_LB＝1）、bUart_CTS は bUart_RTS と同じになります。	R
b3	bUart_DDCD	デルタデータキャリア検出 本ビットは、最後の rUart_MSR 読み出し以降モデム制御ライン UART_DCD_N が変更されたことを示すために使用されます。 1'b0＝rUart_MSR の最終読み出し以降 UART_DCD_N の変更なし 1'b1＝rUart_MSR の最終読み出し以降 UART_DCD_N の変更あり rUart_MSR を読み出すと bUart_DDCD ビットがクリアされます。ループバックモード時（bUart_LB＝1）、bUart_DDCD は bUart_OUT2 の変化を反映します。 **備考**　bUart_DDCD ビットがセットされておらず、UART_DCD_N 信号がアサートされ（Low）、リセットが起きる（ソフトウェアその他）と、リセット解除後 bUart_DDCD ビットがセットされます。	R
b2	bUart_TERI	被呼表示の立ち下がりエッジ 本ビットは、rUart_MSR が最後に読み出されてから、入力 UART_RI_N に（アクティブの Low から非アクティブの High 状態への）変化が起きたことを示すために使用されます。 1'b0＝rUart_MSR の最終読み出し以降 UART_RI_N の変更なし 1'b1＝rUart_MSR の最終読み出し以降 UART_RI_N の変更あり rUart_MSR を読み出すと bUart_TERI ビットがクリアされます。ループバックモード時（bUart_LB＝1）、bUart_TERI は、bUart_OUT1 の High から Low への状態変化を反映します。	R
b1	bUart_DDSR	デルタデータセットレディ 本ビットは、最後の rUart_MSR 読み出し以降モデム制御ライン UART_DSR_N が変更されたことを示すために使用されます。 1'b0＝rUart_MSR の最終読み出し以降 UART_DSR_N の変更なし 1'b1＝rUart_MSR の最終読み出し以降 UART_DSR_N の変更あり rUart_MSR を読み出すと bUart_DDSR ビットがクリアされます。ループバックモード時（bUart_LB＝1）、bUart_DDSR は bUart_DTR の変化を反映します。 **備考**　bUart_DDSR ビットがセットされておらず、UART_DSR_N 信号がアサートされ（Low）、リセットが起きる（ソフトウェアその他）と、リセット解除後 bUart_DDSR ビットがセットされます。	R
b0	bUart_DCTS	デルタ送信可 本ビットは、最後の rUart_MSR 読み出し以降モデム制御ライン UART_CTS_N が変更されたことを示すために使用されます。 1'b0＝rUart_MSR の最終読み出し以降 UART_CTS_N の変更なし 1'b1＝rUart_MSR の最終読み出し以降 UART_CTS_N の変更あり rUart_MSR を読み出すと bUart_DCTS ビットがクリアされます。ループバックモード時（bUart_LB＝1）、bUart_DCTS は bUart_RTS の変化を反映します。 **備考**　bUart_DCTS ビットがセットされておらず、UART_CTS_N 信号がアサートされ（Low）、リセットが起きる（ソフトウェアその他）と、リセット解除後 bUart_DCTS ビットがセットされます。	R

1.4.11　rUart_SCR ― スクラッチパッドレジスタ

アドレス　4006 001Ch（UART1）
4006 101Ch（UART2）
4006 201Ch（UART3）
5000 001Ch（UART4）
5000 101Ch（UART5）
5000 201Ch（UART6）
5000 301Ch（UART7）
5000 401Ch（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_SCR							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.19　rUart_SCR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	bUart_SCR	本レジスタは、プログラマが一時的な保存空間として使用するためのものです。 UART では決まった用途はありません。	R/W

1.4.12　rUart_SRBR_STHR ― シャドー受信バッファ／送信保持レジスタ

- 依存関係：bUart_DLAB ビット＝0

アドレス　4006 0030h（UART1）
4006 1030h（UART2）
4006 2030h（UART3）
5000 0030h（UART4）
5000 1030h（UART5）
5000 2030h（UART6）
5000 3030h（UART7）
5000 4030h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_SRBR_STHR							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.20　rUart_SRBR_STHR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	bUart_SRBR_STHR	本レジスタ読み出し時：シャドー受信バッファレジスタ（rUart_SRBR） ● 本レジスタは rUart_RBR のシャドーであり、マスタからのバーストアクセスを受け付けるために 16 個の 32 ビット位置が割り当てられています。 本レジスタには、シリアル入力ポート UART_RXD で受信したデータバイトが入れられます。 本レジスタのデータは、ラインステータスレジスタ（rUart_LSR）のデータ準備完了（bUart_DR ビット）がセットされている場合のみ有効です。 ● FIFO が禁止されている場合（bUart_FIFOE＝0）、rUart_RBR のデータは、次のデータが到着する前に読み出す必要があります。読み出されない場合、上書きされ、オーバーランエラーになります。 ● FIFO が許可されている場合（bUart_FIFOE＝1）、本レジスタは、受信 FIFO の先頭にアクセスします。受信 FIFO がフルで、本レジスタが次のデータキャラクタ到着前に読み出されない場合は、FIFO 内にすでにあるデータが保持され、受信データは失われます。 オーバーランエラーも発生します。 本レジスタ書き込み時：シャドー送信保持レジスタ（rUart_STHR） ● 本レジスタは rUart_THR のシャドーであり、マスタからのバーストアクセスを受け付けるために 16 個の 32 ビット位置が割り当てられています。 本レジスタには、シリアル出力ポート UART_TXD で送信するデータが入れられます。データを rUart_THR に書き込むのは、rUart_LSR レジスタの THR エンプティ（bUart_THRE）ビットがセットされたときのみとします。 ● FIFO が禁止されており（bUart_FIFOE＝0）、bUart_THRE がセットされている時に rUart_THR に 1 キャラクタ書き込むと、bUart_THRE がクリアされます。bUart_THRE が再びセットされる前に rUart_THR に追加書き込みがあると、rUart_THR データは上書きされます。 ● FIFO が許可されており（bUart_FIFOE＝1）、bUart_THRE がセットされている場合、FIFO がフルになる前に 16 キャラクタのデータを rUart_THR に書き込み可能です。FIFO がフルの時に更にデータを書き込もうとすると書き込みデータは失われます。	R/W

1.4.13　rUart_FAR ― FIFO アクセスレジスタ

アドレス　4006 0070h（UART1）
4006 1070h（UART2）
4006 2070h（UART3）
5000 0070h（UART4）
5000 1070h（UART5）
5000 2070h（UART6）
5000 3070h（UART7）
5000 4070h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bUart_FAR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.21　rUart_FAR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	bUart_FAR	FIFO アクセスレジスタ 本レジスタは、テスト用に FIFO アクセスを有効にするために使用されます。その結果、FIFO が許可されている場合、マスタは受信 FIFO に書き込みができ、送信 FIFO から読み出し可能です。 FIFO が許可されていない場合、マスタによる rUart_RBR への書き込みおよびマスタによる rUart_THR からの読み出しが可能です。 1’b0＝FIFO アクセスモード禁止 1’b1＝FIFO アクセスモード許可 **備考**　FIFO アクセスモードを有効／無効にすると、受信 FIFO と送信 FIFO の制御部分がリセットされ、FIFO はエンプティと見なされます。	R/W

1.4.14　rUart_TFR ― 送信 FIFO 読み出し

アドレス　4006 0074h（UART1）
4006 1074h（UART2）
4006 2074h（UART3）
5000 0074h（UART4）
5000 1074h（UART5）
5000 2074h（UART6）
5000 3074h（UART7）
5000 4074h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bUart_TFR							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.22　rUart_TFR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	bUart_TFR	送信 FIFO 読み出し これらのビットは、FIFO アクセスモードが許可されている（bUart_FAR＝1）場合のみ有効です。 FIFO が許可（bUart_FIFOE）されている場合、本レジスタを読み出すと、送信 FIFO の先頭にあるデータが読み出されます。連続して読み出しがあるたびに送信 FIFO がポップされ、現在 FIFO の先頭にある次のデータ値が読み出されます。 FIFO が許可されていない場合、本レジスタを読み出すと、rUart_THR レジスタのデータが読み出されます。	R

1.4.15　rUart_RFW — 受信 FIFO 書き込み

アドレス　4006 0078h（UART1）
4006 1078h（UART2）
4006 2078h（UART3）
5000 0078h（UART4）
5000 1078h（UART5）
5000 2078h（UART6）
5000 3078h（UART7）
5000 4078h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	bUart_RFFE	bUart_RFPE	bUart_RFWD							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.23　rUart_RFW レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b10	予約ビット		R
b9	bUart_RFFE	受信 FIFO フレーミングエラー このビットは、FIFO アクセスモードが許可（bUart_FAR＝1）の場合のみ有効です。 FIFO が許可されている場合、本ビットは、フレーミングエラー検出情報を受信 FIFO に書き込む際に使用されます。 FIFO が許可されていない場合、本ビットは、フレーミングエラー検出情報を rUart_RBR レジスタに書き込む際に使用されます。 **備考）** 本ビットはまた受信 FIFO のブレーク条件でアクティブとなります。	W
b8	bUart_RFPE	受信 FIFO パリティエラー このビットは、FIFO アクセスモードが許可（bUart_FAR＝1）の場合のみ有効です。 FIFO が許可されている場合、本ビットは、パリティエラー検出情報を受信 FIFO に書き込む際に使用されます。 FIFO が許可されていない場合、本ビットは、パリティエラー検出情報を rUart_RBR レジスタに書き込む際に使用されます。	W
b7～b0	bUart_RFWD	受信 FIFO 書き込みデータ これらのビットは、FIFO アクセスモードが許可（bUart_FAR＝1）の場合のみ有効です。 FIFO が許可されている場合、bUart_RFWD に書き込まれたデータは受信 FIFO に入力されます。連続して書き込みがあるたびに新しいデータが受信 FIFO の次の書き込み位置に押し出されます。 FIFO が許可されていない場合、bUart_RFWD に書き込まれたデータは rUart_RBR レジスタに入力されます。	W

1.4.16　rUart_USR ― UART ステータスレジスタ

アドレス　4006 007Ch（UART1）
4006 107Ch（UART2）
4006 207Ch（UART3）
5000 007Ch（UART4）
5000 107Ch（UART5）
5000 207Ch（UART6）
5000 307Ch（UART7）
5000 407Ch（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bUart_RFF	bUart_RFNE	bUart_TFE	bUart_TFNF	bUart_BUSY
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

表 1.24　rUart_USR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット		R
b4	bUart_RFF	受信 FIFO フル 本ビットは、受信 FIFO が完全にフルであることを示すために使用されます。 1'b0＝受信 FIFO はフルでない 1'b1＝受信 FIFO はフルである 本ビットは、受信 FIFO がもはやフルではなくなったときクリアされます。	R
b3	bUart_RFNE	受信 FIFO はエンプティでない 本ビットは、受信 FIFO に 1 つ以上のエントリがあることを示すために使用されます。 1'b0＝受信 FIFO はエンプティである 1'b1＝受信 FIFO はエンプティではない 本ビットは、受信 FIFO がエンプティになったときクリアされます。	R
b2	bUart_TFE	送信 FIFO エンプティ 本ビットは、送信 FIFO が完全にエンプティであることを示すために使用されます。 1'b0＝送信 FIFO はエンプティではない 1'b1＝送信 FIFO はエンプティである 本ビットは、送信 FIFO がもはやエンプティではなくなったときクリアされます。	R
b1	bUart_TFNF	送信 FIFO はフルでない 本ビットは、送信 FIFO がフルでないことを示すために使用されます。 1'b0＝送信 FIFO はフルである 1'b1＝送信 FIFO はフルではない 本ビットは、送信 FIFO がフルになったときクリアされます。	R

表 1.24　rUart_USR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bUart_BUSY	UART ビジー 本ビットは、シリアル転送が進行中であることを示し、クリアされると、UART がアイドルまたは非アクティブであることを示します。 1'b0＝UART はアイドルまたは非アクティブ 1'b1＝UART はビジー（データ転送中） 本ビットは、以下の条件のいずれかでセットされます。 ● シリアルインタフェース上で送信進行中 ● rUart_THR に送信データがある。このとき、除数ラッチアクセスビットが 0（bUart_DLAB＝0）のとき、FIFO アクセスモードは使用中でなく（bUart_FAR＝0）、ボーレート除数は 0 以外（{rUart_DLH, rUart_DLL} !＝0）です。 ● インタフェース上で受信進行中 ● rUart_RBR に受信データがある。このとき、FIFO アクセスモードは使用中でない（bUart_FAR＝0）。 **備考）** 他のデバイスから新たにキャラクタが送信されていても bUart_BUSY ビットはクリアされる可能性があります。 すなわち、UART が rUart_RBR および bUart_THR にデータを持っておらず、送信が進行中でないときに、新しいキャラクタのスタートビットがちょうど UART に到着したばかりの場合です。 これは、ビット期間の中間まで有効なスタートが検出されず、この期間がプログラムされたボーレート除数に依存するためです。 また、本ビットのアサートは、より遅いクロック UART_SCLK および UART_PCLK の数サイクル分遅延します。	R

1.4.17　rUart_TFL ― 送信 FIFO レベル

アドレス　4006 0080h（UART1）
4006 1080h（UART2）
4006 2080h（UART3）
5000 0080h（UART4）
5000 1080h（UART5）
5000 2080h（UART6）
5000 3080h（UART7）
5000 4080h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bUart_TFL				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.25　rUart_TFL レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット		R
b4～b0	bUart_TFL	送信 FIFO レベル 本ビットは、送信 FIFO のデータエントリ数を表示します。	R

1.4.18　rUart_RFL ― 受信 FIFO レベル

アドレス　4006 0084h（UART1）
4006 1084h（UART2）
4006 2084h（UART3）
5000 0084h（UART4）
5000 1084h（UART5）
5000 2084h（UART6）
5000 3084h（UART7）
5000 4084h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bUart_RFL				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.26　rUart_RFL レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット		R
b4～b0	bUart_RFL	受信 FIFO レベル 本ビットは、受信 FIFO のデータエントリ数を表示します。	R

1.4.19　rUart_SRR ― ソフトウェアリセットレジスタ

アドレス　4006 0088h（UART1）
4006 1088h（UART2）
4006 2088h（UART3）
5000 0088h（UART4）
5000 1088h（UART5）
5000 2088h（UART6）
5000 3088h（UART7）
5000 4088h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	bUart_XFR	bUart_RFR	bUart_UR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.27　rUart_SRR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b3	予約ビット		R
b2	bUart_XFR	送信 FIFO リセット 本ビットは、送信 FIFO リセットビット（bUart_XFIFOR）のシャドーになります。本ビットに 1 をセットすると、rUart_FCR 値を操作しなくても、送信 FIFO の制御部分をリセットし、FIFO をエンプティとして処理します。 **備考**）　本ビットは、「自己クリア」します。本ビットをクリアする必要はありません。	W
b1	bUart_RFR	受信 FIFO リセット 本ビットは、受信 FIFO リセットビット（bUart_RFIFOR）のシャドーになります。本ビットに 1 をセットすると、rUart_FCR 値を操作しなくても、受信 FIFO の制御部分をリセットし、FIFO をエンプティとして処理します。 **備考**）　本ビットは、「自己クリア」します。本ビットをクリアする必要はありません。	W
b0	bUart_UR	UART リセット 本ビットに 1 をセットすると、UART を非同期リセットし、リセットアサートを同期で解除します。UART_SCLK と UART_PCLK の両ドメインともリセットされます。	W

1.4.20　rUart_SRTS ― シャドー送信要求

アドレス　4006 008Ch（UART1）
4006 108Ch（UART2）
4006 208Ch（UART3）
5000 008Ch（UART4）
5000 108Ch（UART5）
5000 208Ch（UART6）
5000 308Ch（UART7）
5000 408Ch（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bUart_SRTS
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.28　rUart_SRTS レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	bUart_SRTS	シャドー送信要求 本ビットは、bUart_RTS ビットのシャドーで、rUart_MCR のリードモディファイライトが不要になります。本ビットは、送信要求（UART_RTS_N）出力を直接制御する際に使用されます。送信要求（UART_RTS_N）出力は、モデムまたはデータセットに UART がデータを交換する準備ができていることを知らせるために使用されます。 自動 RTS フロー制御が有効でない（bUart_AFCE＝0）場合、bUart_RTS を High にプログラムすることにより UART_RTS_N 信号が Low に設定されます。 自動フロー制御（bUart_AFCE＝1）かつ FIFO 許可（bUart_FIFOE＝1）のとき、UART_RTS_N 出力は同じように制御されますが、ただし「受信 FIFO 準フルトリガ」によりゲートされます。ここで「準フル」とは、FIFO に利用可能なスロットが 2 つあることを指します（しきい値を超えると UART_RTS_N は非アクティブの High になります）。 **備考）** ループバックモード（bUart_LB＝1）時、UART_RTS_N 出力は非アクティブな High に保持され、本位置の値が内部的に入力にループバックされます。	R/W

1.4.21　rUart_SBCR — シャドーブレークコントロールレジスタ

アドレス　4006 0090h（UART1）
4006 1090h（UART2）
4006 2090h（UART3）
5000 0090h（UART4）
5000 1090h（UART5）
5000 2090h（UART6）
5000 3090h（UART7）
5000 4090h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	bUart_SBCR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.29　rUart_SBCR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	bUart_SBCR	シャドーブレークコントロールビット 本ビットは、ブレークビット（bUart_BC）のシャドーで、rUart_MCR のリードモディファイライトが不要になります。本ビットは、ブレーク条件を受信デバイスに送る際に使用されます。 1 の時、シリアル出力が強制的にスペース（論理 0）状態にされます。ループバックモード以外では、bUart_LB の決定に従い、UART_TXD ラインは、ブレークビットがクリアされるまで Low のままです。 ループバックモード時は、ブレーク条件はレシーバに内部ループバックされます。	R/W

1.4.22　rUart_SFE ― シャドーFIFO 許可

アドレス　4006 0098h（UART1）
4006 1098h（UART2）
4006 2098h（UART3）
5000 0098h（UART4）
5000 1098h（UART5）
5000 2098h（UART6）
5000 3098h（UART7）
5000 4098h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bUart_SFE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.30　rUart_SFE レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	bUart_SFE	シャドーFIFO 許可 本ビットは、FIFO 許可ビット（bUart_FIFOE）のシャドーになります。本ビットは、以前 rUart_FCR に書き込まれた値を保存し、本値をマスクしなければならない負担を取り除き、その結果 FIFO 許可ビットのみが更新されるようにするために使用されます。 本ビットは、送信 FIFO および受信 FIFO を許可／禁止します。本ビットが許可に設定された後 0（禁止）になると、FIFO の送信制御部分と受信制御部分の両方がリセットされます。 1’b0＝禁止 1’b1＝許可	R/W

1.4.23　rUart_SRT ― シャドーRCVR トリガ

アドレス　4006 009Ch（UART1）
4006 109Ch（UART2）
4006 209Ch（UART3）
5000 009Ch（UART4）
5000 109Ch（UART5）
5000 209Ch（UART6）
5000 309Ch（UART7）
5000 409Ch（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bUart_SRCVR	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.31　rUart_SRT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット		R
b1、b0	bUart_SRCVR	シャドー受信 FIFO トリガ 本ビットは、受信 FIFO トリガビット（bUart_RCVR）のシャドーになります。本ビットは、以前 rUart_FCR に書き込まれた値を保存し、本値をマスクしなければならない負担を取り除き、その結果受信 FIFO トリガビットのみが更新されるようにするために使用されます。 本ビットは、受信データあり割り込みが発生する受信 FIFO のトリガレベルを選択するために使用されます。 以下のトリガレベルがサポートされています。 2'b00＝FIFO 内に 1 キャラクタ 2'b01＝FIFO 1/4 フル 2'b10＝FIFO 1/2 フル 2'b11＝FIFO フルまであと 2 キャラクタ	R/W

1.4.24　rUart_STET ― シャドーTX エンプティトリガ

アドレス　4006 00A0h（UART1）
4006 10A0h（UART2）
4006 20A0h（UART3）
5000 00A0h（UART4）
5000 10A0h（UART5）
5000 20A0h（UART6）
5000 30A0h（UART7）
5000 40A0h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bUart_STET	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.32　rUart_STET レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット		R
b1、b0	bUart_STET	シャドー送信エンプティトリガ 本ビットは、送信エンプティトリガビット（bUart_TET）のシャドーになります。 本ビットは、送信エンプティトリガビットのみが更新されるように、以前 rUart_FCR に書き込まれた値を保存し、本値をマスクしなければならない負担を取り除くために使用されます。 本ビットは、モードがアクティブのとき、THRE 割り込みが発生するエンプティしきい値レベルを指定するために使用されます。 以下のトリガレベルがサポートされています。 2'b00＝FIFO エンプティ 2'b01＝FIFO 内に 2 キャラクタ 2'b10＝FIFO 1/4 フル 2'b11＝FIFO 1/2 フル	R/W

1.4.25　rUart_HTX ― 送信停止

アドレス　4006 00A4h（UART1）
4006 10A4h（UART2）
4006 20A4h（UART3）
5000 00A4h（UART4）
5000 10A4h（UART5）
5000 20A4h（UART6）
5000 30A4h（UART7）
5000 40A4h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bUart_HTX
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.33　rUart_HTX レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	bUart_HTX	送信停止 本レジスタはテストのために送信を停止する際に使用されます。その結果、FIFO が許可のとき、送信 FIFO はマスタによりフルにされます。 1’b0＝送信停止禁止 1’b1＝送信停止許可 **備考）** FIFO が許可されていない場合は、bUart_HTX レジスタの設定は動作に影響を与えません。	R/W

1.4.26　rUart_DMASA ― DMA ソフトウェアアクノリッジ

- UART4～8 のみが対象

アドレス　5000 00A8h（UART4）
5000 10A8h（UART5）
5000 20A8h（UART6）
5000 30A8h（UART7）
5000 40A8h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bUart_DMASA
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.34　rUart_DMASA レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	bUart_DMASA	本レジスタは、エラーのため転送中止が必要な場合 DMA ソフトウェア ACK 応答を実行する際に使用されます。本ビットに 1 をセットすると、ACK 応答が実行されます。 たとえば、DMA がチャネルを無効にした場合、UART は要求をクリアしなければなりません。これにより、送受信要求信号はデアサートされます。 **備考）** 本ビットは、「自己クリア」されます。本ビットをクリアする必要はありません。	W

1.4.27　rUart_TO — タイムアウトカウンタコンフィグレーションレジスタ

レシーバまたはトランシーバが新しいキャラクタを待つ、タイムアウト遅延期間

アドレス　4006 0100h（UART1）
4006 1100h（UART2）
4006 2100h（UART3）
5000 0100h（UART4）
5000 1100h（UART5）
5000 2100h（UART6）
5000 3100h（UART7）
5000 4100h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bUart_TO3								bUart_TO2							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bUart_TO1								bUart_TO0							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.35　rUart_TO レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b24	bUart_TO3	bUart_TO[n]（n＝0～3） タイムアウト n 値（n＝3） 4 つのタイムアウトがあります。 ● 受信用に 2 つ（bUart_TO0～1）。UART_RXD 上のアイドル状態（サイレント期間検出）専用。 ● 送信用に 2 つ（bUart_TO2～3）。UART_TXD 上のアイドル状態（サイレント期間検出）専用。 各タイムアウト（n＝0～3）につき、以下の設定となります。 8'h0 : - タイムアウト無効 8'h1～8'hff : - タイムアウトは有効で、タイムアウト遅延は、 bUart_TO[n]×「ボークロック周期」 「**1.5.1.1 UART（RS232）シリアルプロトコル**」を参照してください。 「**1.5.1.10(2) トランシーバタイムアウト**」を参照してください。 「**1.5.1.10(1) レシーバタイムアウト**」を参照してください。 ボークロックは、ボーレート除数（bUart_DLL および bUart_DLH）で設定します。 タイムアウト遅延期間は、レシーバまたはトランシーバが UART_RXD（タイムアウト n（n＝0、1））または UART_TXD（タイムアウト n（n＝2～3））上で新しいキャラクタを待つ時間です。 bUart_TO[n]フィールドが 0 に設定されている場合、タイムアウトカウンタのクロックは停止され、カウンタは現在の値を保持します。rUart_STATUSTO の bUart_TIMEOUTInt[n]ビットは現在の値を保持します。 bUart_TO[n]フィールドが 0 に設定されていない場合、レシーバカウンタ n（n＝0、1）またはトランシーバカウンタ n（n＝2、3）は、bUart_TO[n]内の設定値を 8 ビットカウンタにロードします。	R/W

表 1.35　rUart_TO レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
		このカウンタ[n]は、各ビット周期でデクリメントされ、新しいキャラクタが受信（n＝0、1）または送信（n＝2、3）されるたびに再ロードされます。 カウンタ[n]が 0 になるとタイムアウトカウンタステータスレジスタ（rUart_STATUSTO）の bUart_TIMEOUTInt[n]ビットが立ち上がり、割り込みを起動します（マスクされていない場合）。カウンタは 0 になると LOAD コマンドを受け取るまでロック状態になります。 「**図 1.14　レシーバタイムアウトの概要**」を参照してください。 「**図 1.15　トランシーバタイムアウトの概要**」を参照してください。 「**図 1.16　レシーバ＆トランシーバのタイムアウト（n＝0～3）タイミング**」を参照してください。 「**1.5.1.10(1)　レシーバタイムアウト**」を参照してください。 「**1.5.1.10(2)　トランシーバタイムアウト**」を参照してください。	
b23～b16	bUart_TO2	タイムアウト n 値（n＝2） 詳細については上記を参照してください。	R/W
b15～b8	bUart_TO1	タイムアウト n 値（n＝1） 詳細については上記を参照してください。	R/W
b7～b0	bUart_TO0	タイムアウト n 値（n＝0） 詳細については上記を参照してください。	R/W

1.4.28　rUart_CTRLTO ― タイムアウトコントロールレジスタ

アドレス　4006 0104h（UART1）
4006 1104h（UART2）
4006 2104h（UART3）
5000 0104h（UART4）
5000 1104h（UART5）
5000 2104h（UART6）
5000 3104h（UART7）
5000 4104h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	bUart_TG							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	bUart_EnableFilteringRXD	bUart_EnableDE	bUart_REARMTO3	bUart_REARMTO2	bUart_REARMTO1	bUart_REARMTO0	bUart_STARTTO3	bUart_STARTTO2	bUart_STARTTO1	bUart_STARTTO0
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.36　rUart_CTRLTO レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b24	予約ビット		R
b23～b16	bUart_TG	タイムガード値 本 LSI では、使用できません。 初期値のまま使用してください。	R/W
b15～b10	予約ビット		R
b9	bUart_EnableFilteringRXD	半二重モードで UART_RXD のフィルタリングを許可 本 LSI では、使用できません。 初期値のまま使用してください。	R/W
b8	bUart_EnableDE	外部端子 UART_RTS_N のマルチプレクシングを許可 本 LSI では、使用できません。 初期値のまま使用してください。	R/W
b7	bUart_REARMTO3	bUart_REARMTO[n]（n＝0～3）。リアームタイムアウト リアームタイムアウト n 値（n＝3） 各タイムアウト（n＝0～3）につき、以下の設定となります。 0＝無効 1＝タイムアウトカウンタをリスタート bUart_REARMTO[n]（n＝0～3）（立ち上がり検出）の起動後、カウンタはただちに値 bUart_TO[n]（n＝0～3）からカウントダウンを開始します。 「**図 1.14 レシーバタイムアウトの概要**」を参照してください。 「**図 1.15 トランシーバタイムアウトの概要**」を参照してください。 「**1.5.1.10(1) レシーバタイムアウト**」を参照してください。 「**1.5.1.10(2) トランシーバタイムアウト**」を参照してください。	R/W
b6	bUart_REARMTO2	リアームタイムアウト n 値（n＝2） 詳細については上記を参照してください。	R/W
b5	bUart_REARMTO1	リアームタイムアウト n 値（n＝1） 詳細については上記を参照してください。	R/W
b4	bUart_REARMTO0	リアームタイムアウト n 値（n＝0） 詳細については上記を参照してください。	R/W

表 1.36　rUart_CTRLTO レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3	bUart_STARTTO3	bUart_STARTTO[n]（n＝0～3）。スタートタイムアウト スタートタイムアウト n 値（n＝3） 各タイムアウト（n＝0～3）につき、以下の設定となります。 0＝無効 1＝タイムアウトカウンタの初期化を開始 bUart_STARTTO[n]（n＝0～3）（立ち上がり検出）の起動後、タイムアウトカウンタを初期化し、ロック状態にします。 「**図 1.14 レシーバタイムアウトの概要**」を参照してください。 「**図 1.15 トランシーバタイムアウトの概要**」を参照してください。 「**1.5.1.10(1) レシーバタイムアウト**」を参照してください。 「**1.5.1.10(2) トランシーバタイムアウト**」を参照してください。	R/W
b2	bUart_STARTTO2	スタートタイムアウト n 値（n＝2） 詳細については上記を参照してください。	R/W
b1	bUart_STARTTO1	スタートタイムアウト n 値（n＝1） 詳細については上記を参照してください。	R/W
b0	bUart_STARTTO0	スタートタイムアウト n 値（n＝0） 詳細については上記を参照してください。	R/W

1.4.29　rUart_STATUSTO ― タイムアウトカウンタステータスレジスタ

ソフトウェアドライバ（アプリケーション）は、各タイムアウトの状態を判断するために行う割り込みサービスルーチンまたはポーリング中、本レジスタを読み出します。

アドレス　4006 0108h（UART1）
4006 1108h（UART2）
4006 2108h（UART3）
5000 0108h（UART4）
5000 1108h（UART5）
5000 2108h（UART6）
5000 3108h（UART7）
5000 4108h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	bUart_DE	bUart_TIMEOUTStatus3	bUart_TIMEOUTStatus2	bUart_TIMEOUTStatus1	bUart_TIMEOUTStatus0	bUart_TIMEOUTInt3	bUart_TIMEOUTInt2	bUart_TIMEOUTInt1	bUart_TIMEOUTInt0
リセット後の値	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

表 1.37　rUart_STATUSTO レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b9	予約ビット		R
b8	bUart_DE	本レジスタは、内部送信データイネーブルの値を示します。 本 LSI では、使用できません。読み出し値は無視してください。	R
b7	bUart_TIMEOUTStatus3	bUart_TIMEOUTStatus[n]（n＝0～3）。タイムアウト検出状態 タイムアウト n 検出状態（n＝3） 本レジスタは、通常、割り込みサービスルーチンまたはポーリング中にソフトウェアドライバにより読み出されます。 4 つのタイムアウトがあります（bUart_TO[n]（n＝0～3））。 ● 受信用に 2 つ（bUart_TO0～1）。UART_RXD 上のアイドル状態（サイレント期間検出）専用。 ● 送信用に 2 つ（bUart_TO2～3）。UART_TXD 上のアイドル状態（サイレント期間検出）専用。 各タイムアウト n（n＝0～3）につき、以下の設定となります。 1'b0＝タイムアウトカウンタ[n]の値が“0”以外 1'b1＝タイムアウトカウンタ[n]の値が“0” 「**1.5.1.10 MODBUS 管理用トランシーバ＆レシーバタイムアウト**」を参照してください。 「**図 1.14 レシーバタイムアウトの概要**」を参照してください。 「**図 1.15 トランシーバタイムアウトの概要**」を参照してください。 「**図 1.16 レシーバ＆トランシーバのタイムアウト（n＝0～3）タイミング**」を参照してください。 「**1.5.1.10(1) レシーバタイムアウト**」を参照してください。 「**1.5.1.10(2) トランシーバタイムアウト**」を参照してください。	R
b6	bUart_TIMEOUTStatus2	タイムアウト n 検出状態（n＝2） 詳細については上記を参照してください。	R
b5	bUart_TIMEOUTStatus1	タイムアウト n 検出状態（n＝1） 詳細については上記を参照してください。	R
b4	bUart_TIMEOUTStatus0	タイムアウト n 検出状態（n＝0） 詳細については上記を参照してください。	R

表 1.37　rUart_STATUSTO レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3	bUart_TIMEOUTInt3	bUart_TIMEOUTInt[n]（n＝0～3）。タイムアウト検出割り込み タイムアウト n 検出割り込み（n＝3） 本レジスタは、通常、割り込みサービスルーチンまたはポーリング中にソフトウェアドライバにより読み出されます。本レジスタのフィールドのほとんどは、ホストに割り込みを起こさせます。 4 つのタイムアウトがあります（bUart_TO[n]（n＝0～3））。 ● 受信用に 2 つ（bUart_TO0～1）。UART_RXD 上のアイドル状態（サイレント期間検出）専用。 ● 送信用に 2 つ（bUart_TO2～3）。UART_TXD 上のアイドル状態（サイレント期間検出）専用。 各タイムアウト n（n＝0～3）につき、以下の設定となります。 1’b0＝最後のタイムアウト開始コマンド以来タイムアウトなし。タイムアウトからアクティブな割り込みはありません。 1’b1＝最後のタイムアウト開始コマンド以来タイムアウトあり。本ビットが High のとき、割り込み信号 UART_Int も High です。各フィールド（ビット[3:0]）は、rUart_IER レジスタの該当ビットをマスクすることによりマスク可能です。「**1.5.1.6 割り込み**」を参照してください。 本レジスタでは、ビットは、読み出しによってはクリアされません。本レジスタの（非予約）ビット（ビット[3:0]）に 1’b1 を書き込むとビットがクリアされますが、1’b0 を書き込んでも影響はありません。 各タイムアウト n（n＝0～3）は、rUart_CTRLTO レジスタの該当ビットにより開始またはリセット可能です。 「**1.5.1.10 MODBUS 管理用トランシーバ&レシーバタイムアウト**」を参照してください。 「**図 1.14 レシーバタイムアウトの概要**」を参照してください。 「**図 1.15 トランシーバタイムアウトの概要**」を参照してください。 「**図 1.16 レシーバ&トランシーバのタイムアウト（n＝0～3）タイミング**」を参照してください。 「**1.5.1.10(1) レシーバタイムアウト**」を参照してください。 「**1.5.1.10(2) トランシーバタイムアウト**」を参照してください。	R/W
b2	bUart_TIMEOUTInt2	タイムアウト n 検出割り込み（n＝2） 詳細については上記を参照してください。	R/W
b1	bUart_TIMEOUTInt1	タイムアウト n 検出割り込み（n＝1） 詳細については上記を参照してください。	R/W
b0	bUart_TIMEOUTInt0	タイムアウト n 検出割り込み（n＝0） 詳細については上記を参照してください。	R/W

1.4.30　rUart_TDMACR ― DMA コントロールレジスタ（送信モード）

- UART4～8 のみが対象

注意

上記の UART のみが、DMA コントローラとの DMA 要求を管理可能です。

アドレス　5000 010Ch（UART4）
5000 110Ch（UART5）
5000 210Ch（UART6）
5000 310Ch（UART7）
5000 410Ch（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	bUart_CURRENT_DEST_BLOCK_SIZE												
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bUart_DEST_BLOCK_SIZE													bUart_DEST_BURST_SIZE		bUart_TDMAE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.38　rUart_TDMACR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b29	予約ビット		R
b28～b16	bUart_CURRENT_DEST_BLOCK_SIZE	DEST_BLOCK_SIZE の残数 本フィールドはブロック転送が終了するたびにデクリメントされます。 ファームウェアが以下の動作を行うと、bUart_DEST_BLOCK_SIZE 値が bUart_CURRENT_DEST_BLOCK_SIZE にリロードされます。 bUart_TDMAE ビットを“1”にセット（立ち上がり）	R
b15～b3	bUart_DEST_BLOCK_SIZE	DEST_BLOCK_SIZE 送信 FIFO の宛先ブロック転送サイズ UART はフローコントローラです。したがってユーザは、DMA モードが許可される前または許可されると同時に本フィールドに書き込みを行う必要があります。DEST_BLOCK_SIZE に設定された数字は、各ブロック転送で実行するシングルトランザクションの総数を示します。シングルトランザクションのサイズは 1 バイトです。 いったん転送が開始されると、bUart_DEST_BLOCK_SIZE を読み出すことでブロック転送を完了させるために送信 FIFO へ書き込まれるデータバイト総数が示されます。 13’d0＝0 バイトの転送またはブロック転送終了 13’d1＝1 バイトの転送 13’d2＝2 バイトの転送 … … 13’d8191＝8191 バイトの転送	R/W

表 1.38　rUart_TDMACR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b2、b1	bUart_DEST_BURST_SIZE	DEST_BURST_SIZE 送信 FIFO 内の宛先バーストトランザクションサイズ UART はフローコントローラです。したがってユーザは、DMA モードが許可される前または許可されると同時に本フィールドに書き込みを行う必要があります。DMA 要求時、送信バーストトランザクション要求があるたびに送信 FIFO に書き込むデータバイト数です。 2'b00＝1 バイト 2'b01＝4 バイト 2'b10＝8 バイト 2'b11＝予約（未使用）	R/W
b0	bUart_TDMAE	送信 DMA 許可／禁止 1'b0＝送信モードで DMA を禁止 1'b1＝送信モードで DMA を許可 送信 FIFO の最後の転送完了後（送信 FIFO に DEST_BLOCK_SIZE バイトが書き込まれる）、送信モードでの DMA を禁止するために bUart_TDMAE はハードウェアにより自動的にクリアされます。 したがって、ソフトウェアは本ビットのポーリングを行い、本チャネルがいつ空きになり、次の DMA 転送ができるかを判別可能です。 禁止されている場合、DMA 要求はアサートされません。 許可されている場合、UART は DMA コントローラとの DMA 要求を管理します。 **注意）** DMA 転送中に本ビットがクリアされると、現在の転送（バーストまたはシングル）が終了してから DMA モードが停止します。DMA ブロック転送を完了するには、bUart_DEST_BLOCK_SIZE に bUart_CURRENT_DEST_BLOCK_SIZE を、bUart_DEST_BURST_SIZE に適切な値を書き込みます。bUart_CURRENT_DEST_BLOCK_SIZE 値は、現在の転送が終了しているときのみ一致します。	R/W

1.4.31　rUart_RDMACR ― DMA コントロールレジスタ（受信モード）

- UART4～8 のみが対象

注意

上記の UART のみが、DMA コントローラとの DMA 要求を管理可能です。

アドレス　5000 0110h（UART4）
5000 1110h（UART5）
5000 2110h（UART6）
5000 3110h（UART7）
5000 4110h（UART8）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	bUart_CURRENT_SRC_BLOCK_SIZE												
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bUart_SRC_BLOCK_SIZE													bUart_SRC_BURST_SIZE		bUart_RDMAE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 1.39　rUart_RDMACR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b29	予約ビット		R
b28～b16	bUart_CURRENT_SRC_BLOCK_SIZE	SRC_BLOCK_SIZE の残数 本フィールドはブロック転送が終了するたびにデクリメントされます。 ファームウェアが以下の動作を行うと、bUart_SRC_BLOCK_SIZE 値が bUart_CURRENT_SRC_BLOCK_SIZE にリロードされます。 bUart_RDMAE ビットを“1”にセット（立ち上がり）	R
b15～b3	bUart_SRC_BLOCK_SIZE	SRC_BLOCK_SIZE 受信 FIFO の送信元ブロック転送サイズ UART はフローコントローラです。したがってユーザは、DMA モードが許可される前または許可されると同時に本フィールドに書き込みを行う必要があります。 SRC_BLOCK_SIZE に設定された数字は、各ブロック転送で実行するシングルトランザクションの総数を示します。シングルトランザクションのサイズは 1 バイトです。 13’d0＝0 バイトの転送またはブロック転送終了 13’d1＝1 バイトの転送 13’d2＝2 バイトの転送 … … 13’d8191＝8191 バイトの転送	R/W
b2、b1	bUart_SRC_BURST_SIZE	SRC_BURST_SIZE 受信 FIFO の送信元バーストトランザクションサイズ UART はフローコントローラです。したがってユーザは、DMA モードが許可される前または許可されると同時に本フィールドに書き込みを行う必要があります。DMA 要求時、受信バーストトランザクション要求があるたびに受信 FIFO に読み出されるデータバイト数です。 2’b00＝1 バイト 2’b01＝4 バイト 2’b10＝8 バイト 2’b11＝予約（未使用）	R/W

表 1.40　rUart_RDMACR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bUart_RDMAE	受信 DMA 許可／禁止 1'b0＝受信モードで DMA を禁止 1'b1＝受信モードで DMA を許可 受信 FIFO の最後の転送完了後（受信 FIFO に SRC_BLOCK_SIZE バイトを読み出す）、受信モードでの DMA を禁止するために bUart_RDMAE はハードウェアにより自動的にクリアされます。したがって、ソフトウェアは本ビットのポーリングを行い、本チャネルがいつ空きになり、次の DMA 転送ができるかを判別可能です。 禁止されている場合、DMA 要求はアサートされません。 許可されている場合、UART は DMA コントローラとの DMA 要求を管理します。 **注意）** DMA 転送中に本ビットがクリアされると、現在の転送（バーストまたはシングル）が終了してから DMA モードが停止します。DMA ブロック転送を完了するには、bUart_SRC_BLOCK_SIZE に bUart_CURRENT_SRC_BLOCK_SIZE を、bUart_SRC_BURST_SIZE に適切な値を書き込みます。bUart_CURRENT_SRC_BLOCK_SIZE 値は、現在の転送が終了しているときのみ一致します。	R/W

1.5　動作説明

1.5.1　主要機能ブロックの説明

1.5.1.1　UART（RS232）シリアルプロトコル

UART と選択デバイス間のシリアル通信は調歩同期式なので、始まりと終わりを示すためにスタートビットとストップビットがシリアルデータに付加されます。これらのビットを利用することにより 2 つのデバイスの同期が可能です。このスタートビットとストップビットを付加したシリアルデータの構造は、下図に示すようにキャラクタと呼ばれます。

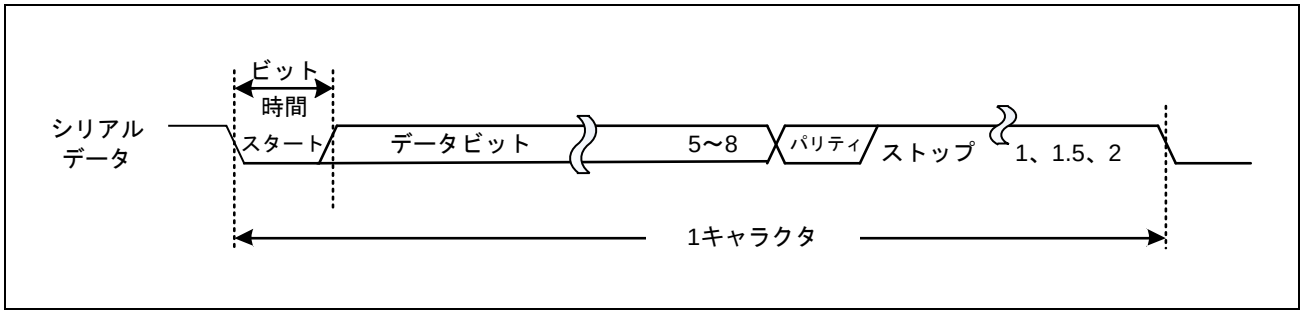


図 1.3　シリアルデータのフォーマット

シリアルキャラクタにはパリティビットを付加可能です。本ビットはキャラクタ構造において最終データビットの後かつストップビットの前に置かれ、本ビットにより UART は、受信データの簡単なエラーチェックが可能です。

シリアルキャラクタの特性制御には、UART ラインコントロールレジスタ（rUart_LCR）が使用されます。データワードの個々のビットは、スタートビット後、最下位ビット（LSB）から順に送信されます。その後、オプションのパリティビットとストップビット（1、1.5、または 2 ビット）が続きます。

注　意

- **送信転送間のアイドル時間の詳細については、「1.5.1.5 連続キャラクタストリーム送信」を参照してください。**
- **ストップビット期間は、以下の理由により、より長く見える場合があります。**
  a）設定によりキャラクタ間に挿入されるアイドル時間
  b）送信方向のボーレート除数値

ビット送信期間は、全く同じです。1.5 個のストップビットが使用される場合のハーフストップビットが唯一の例外です。本期間は、ビット期間またはビット時間と呼ばれ、1 ビット時間は、ボークロック周期になります。内部動作については、基本クロック（ボークロックの 16 倍の周波数）で行います。

ライン安定のため、スタートビット検出後、レシーバはビット時間のほぼ中央でシリアル入力データのサンプリングを行います。各ビットが送信される基本クロック数は正確に分かっているので、サンプリング用の中間点計算は難しいことではなく、スタートビットの中間点サンプル後、ボークロックごとになります。

シリアル入力のデバウンスサンプリングにより、誤ったスタートビットの検出を避けることができます。短いグリッチは、デバウンスで除去されるので、ライン上で遷移が検出されることはありません。グリッチがデバウンスにより排除されないくらい大きい場合、立ち下がりが検出されます。しかしながら、スタートビットが検出されるのは、ハーフビット時間経過後ラインが再び Low になった場合のみです。

シリアルキャラクタの最初の数ビットのサンプリングポイントを下図に示します。

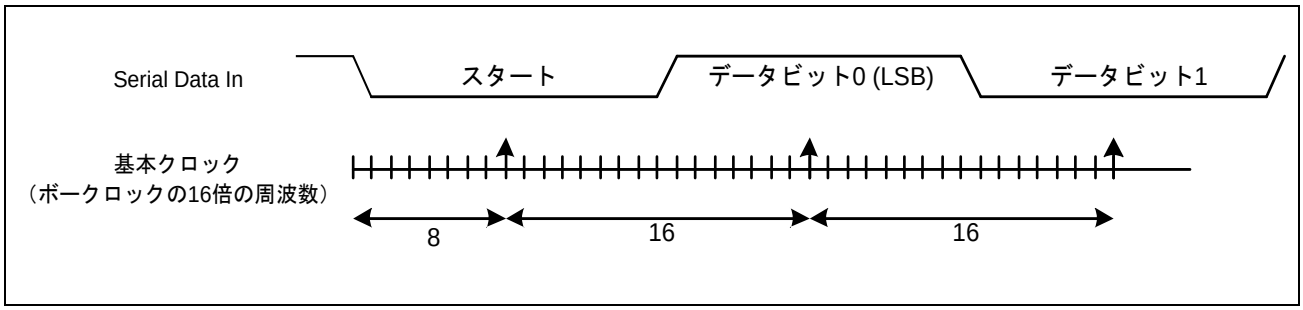


図 1.4　　レシーバシリアルデータサンプル

標準の 16550 規格の一部として、ボークロックモジュールはタイミング情報を提供します。UART のボークロックは、UART_SCLK とボーレート除数（rUart_DLH および rUart_DLL）により制御されます。ボークロックは、以下のように、UART_SCLK をボーレート除数値×16 で分周したものです。

$$\text{ボークロック} = \frac{\text{UART\_SCLK}}{16 \times \text{ボーレート除数}}$$

1.5.1.2　19200 ボーのボーレート許容値

19200 ボーで受信時のボーレート許容値の評価（UART_SCLK：48MHz）

受信時の UART ボーレート設定

- ファームウェアは、rUart_DLL レジスタおよび rUart_DLH レジスタに値 16’d156 を書き込む必要があります。
  - （48/16）／156＝0.0192307 → 19230 ボー
  - ボーレート設定の誤差は＋0.16%

受信付き送信時の最大 UART ボーレート設定

- ファームウェアは、rUart_DLL レジスタおよび rUart_DLH レジスタに値 16’d151 を書き込む必要があります。
  - （48/16）／151 → 19867 ボー
  - ボーレート設定の誤差は＋3.47%

受信付き送信時の最小 UART ボーレート設定

- ファームウェアは、rUart_DLL レジスタおよび rUart_DLH レジスタに値 16’d161 を書き込む必要があります。
  - （48/16）／161 → 18633 ボー
  - ボーレート設定の誤差は－2.95%

1.5.1.3　FIFO 管理

CPU 割り込みを減らすために、送信データ、および受信データをバッファするプログラマブルな FIFO 許可／禁止を備えた 2 つの別々の FIFO（16×8 ビット）が用意されています。これにより CPU は、単一のデータバイトを受信するたびに UART にアクセスする必要がありません。

テスト用にプログラマブル FIFO アクセスモードがあります。これにより、受信 FIFO の CPU による書き込みと、送信 FIFO の CPU による読み出しが可能になります。

FIFO アクセスモードは、FIFO アクセスレジスタ（rUart_FAR レジスタ）で有効になります。FIFO アクセスモードが有効になると、送信 FIFO と受信 FIFO の制御部分はリセットされ、FIFO はエンプティと見なされます。

本モードでは通常どおり送信 FIFO にデータの書き込みが可能ですが、シリアル送信は発生しません（通常動作停止）ので、FIFO からデータは出て行きません。送信 FIFO に書き込まれたデータは、送信 FIFO 読み出し（rUart_TFR レジスタ）でリードバック可能で、読み出される場合は送信 FIFO の先頭データとなります。

同様に、受信 FIFO からのデータ読み出しも通常どおり可能です。本モードでは、UART の通常動作は停止しているので、データは後でリードバック可能なように受信 FIFO に書き込む必要があります。

データの受信 FIFO への書き込みは受信 FIFO 書き込み（rUart_RFW レジスタ）により行われます。10 ビットレジスタの上位 2 ビット（bUart_RFFE&bUart_RFPE）は、受信 FIFO にフレーミングエラー検出情報およびパリティエラー検出情報を書き込む際に使用されます。

ビット bUart_RFFE をセットするとフレーミングエラーを示し、bUart_RFPE はパリティエラーを示します。これらのビットは、受信バッファレジスタを介してのリードバックはできませんが、ラインステータスレジスタを読み出し、問題のデータが受信 FIFO の先頭にあるときに対応するビットをチェックすることにより、確認可能です。

1.5.1.4　クロック管理

UART は、2 つの非同期ドメインクロックを使用します。

- UART_PCLK　　サブシステム内の APB クロックドメイン
- UART_SCLK　　シリアルリファレンスクロック

2 つのシステムクロック境界間ですべての制御およびデータの同期をとるために、同期モジュールが実装されています。2 つのドメインクロック間で追加遅延時間が生成されます。

以下の事項に留意ください。

- 最初にシリアルコントロールレジスタを設定後、シリアルデータを送受信する前には多少の時間が必要です。
- シリアルクロックモジュールは、新しいレジスタ値を参照して、それぞれのステートマシンをリセットするために時間が必要です。本合計時間は、2 つのシステムクロックのうち遅い方で 8 クロックサイクル分を超えないことが保証されています。初期設定後、本最大時間経過前のデータ送受信は禁止です。

1.5.1.5　連続キャラクタストリーム送信

本項では以下について説明します。

- UART が、キャラクタ間にアイドル時間を置かずに、シリアルインタフェース上にキャラクタを連続して送信するシナリオ
- 連続するキャラクタ間にワーストケースのアイドル時間があるシナリオ

送信 FIFO に複数のデータエントリがある場合、UART は、FIFO にあるキャラクタを連続してシリアルバスに送信します。2 つのシステムクロック境界間ですべての制御およびデータの同期をとるため、2 つのドメインクロック間に追加遅延時間があります。本遅延により、現在のストップビットの終わりと次のスタートビットの間にアイドル期間が発生する可能性があります。本アイドル期間はシリアルバス上において延長されたストップビット期間のように見えます。

これは、別のキャラクタが送信の準備ができているかどうかを問い合わせるとき、UART_SCLK ドメインのトランスミッタと UART_PCLK ドメインの TX FIFO 間で同期遅延があるためです。トランスミッタは、現在のストップビットが終わる 1 基本クロック（ボークロックの 16 倍の周波数）前にハンドシェイクを開始します。同期遅延（sync_delay）の期間は、以下の式で求められます。

sync_delay＝4×UART_SCLK＋5×UART_PCLK 周期

sync_delay 期間が、1 基本クロック（ボークロックの 16 倍の周波数）の周期より長い場合、ストップビットの終わりと次のスタートビットの間にアイドル期間が挿入されます。アイドル期間の挿入を防ぐには、以下の条件が成立する必要があります。

sync_delay ≦ 基本クロックの周期

- 挿入されるアイドル期間のワーストケースタイミングは、以下の式で求められます。

  worst_case_idle_duration＝sync_delay＋（15×基本クロックの周期）

  ストップビット期間全体を求めるには、worst_case_idle_duration を、設定したストップビット期間に足します。

1.5.1.6　割り込み

UART_Int 割り込みは、優先順位付けされた割り込み種別のいずれかが許可されアクティブになった場合に発生します。rUart_IER レジスタにより、以下の割り込み種別を許可可能です。

- レシーバライン状態
- 受信データあり
- キャラクタタイムアウト（FIFO モードのみ）
- しきい値以下の送信保持レジスタエンプティ（プログラマブル THRE 割り込みモード時）
- モデム状態
- ビジー検出表示
- UART_RXD でのレシーバタイムアウト 0、1
- UART_TXD でのトランシーバタイムアウト 2、3

割り込み種別の詳細については、「**表 1.41　割り込み制御機能**」を参照してください。

割り込み発生時、マスタはその割り込みの要因を判断するために rUart_IIR レジスタにアクセスします。

FIFO モード（bUart_FIFOE＝1）では、エラーがあるキャラクタが FIFO の先頭に来たときに、ブレーク割り込み、フレーミングエラー、パリティエラーといった要因のレシーバライン状態割り込みが明らかになります。

表 1.41　割り込み制御機能

bUart_IID	優先順位	割り込み種別	割り込み要因	割り込み許可	割り込みリセット制御
4'b0001	—	なし	なし	—	—
4'b0110	最優先	レシーバライン状態	オーバーラン／パリティ／フレーミングエラー、またはブレーク割り込み FIFO モード（bUart_FIFOE＝1）では、エラーがあるキャラクタが FIFO の先頭に来たときに、ブレーク割り込み、フレーミングエラー、パリティエラーといった要因のレシーバライン状態割り込みが明らかになります。	bUart_ELSI	ラインステータスレジスタの読み出し
4'b0100	2 番目	受信データあり	受信データあり（FIFO 禁止）または受信 FIFO トリガレベル到達（FIFO 許可）	bUart_ERBFI	受信バッファレジスタ（FIFO 禁止）の読み出し、または FIFO がトリガレベル未満（FIFO 許可）
4'b1100	2 番目	キャラクタタイムアウト表示	最後の 4 キャラクタ期間中受信 FIFO とのキャラクタ入出力はなく、この間に受信 FIFO 内に少なくとも 1 キャラクタあり（FIFO モードのみ）	bUart_ERBFI	受信バッファレジスタの読み出し
4'b0010	3 番目	送信保持レジスタエンプティ もしくは 送信 FIFO レベルがしきい値と同じかそれ以下	送信保持レジスタエンプティ（プログラマブル THRE モード無効）もしくは送信 FIFO レベルがしきい値以下（プログラマブル THRE モード有効） 割り込み識別レジスタを読み出すことによる THRE 割り込み解除後は、割り込み要因が常に成立していても、内部マスクが THRE 割り込みをデアサートします。本割り込みは、割り込み要因が常に成立する場合、各転送の開始時に再アサートされます。	bUart_ETBEI bUart_PTIME	（THRE が割り込み要因の場合）割り込み識別レジスタの読み出し、送信保持レジスタへの書き込み（FIFO またはプログラマブル THRE モード無効）または送信 FIFO レベルがしきい値超過（FIFO およびプログラマブル THRE モード有効）
4'b0000	4 番目	モデム状態	「送信可」、「データセットレディ」、「被呼表示」、または「データキャリア検出」 **備考**　自動フロー制御モードが有効の場合、bUart_CTS（bUart_DCTS セット）が変更されても割り込みは発生しません。	bUart_EDSSI	モデムステータスレジスタの読み出し
4'b0111	5 番目	ビジー検出表示	UART がビジー中（bUart_BUSY が 1 にセット）にマスタがラインコントロールレジスタに書き込みを試みた。	—	UART ステータスレジスタの読み出し
4'b0101	6 番目	レシーバトランシーバタイムアウト	本機能は、UART_RXD ラインまたは UART_TXD ライン上でアイドル状態を検出します。タイムアウト検出時、ステータスレジスタ（rUart_STATUSTO）の bUart_TIMEOUTInt[n]（n＝0～3）ビットが立ち上がり、割り込みが発生することにより、ドライバにフレームの終わりを示します。 「**1.5.1.10 MODBUS 管理用トランシーバ＆レシーバタイムアウト**」を参照してください。	bUart_ETIMEOUT[n]（n＝0～3）	タイムアウトカウンタステータスレジスタ（rUart_STATUSTO）の読み出し

1.5.1.7　自動フロー制御

UART は、16750 互換の自動 RTS および自動 CTS シリアルデータフロー制御モードを設定できます。自動フロー制御モードは、モデムコントロールレジスタ（rUart_MCR）の bUart_AFCE ビットにより有効にできます。

自動 RTS は、以下のときアクティブになります。

- rUart_MCR レジスタの bUart_RTS ビットおよび bUart_AFCE ビットが両方ともセット
- FIFO が許可（bUart_FIFOE ビットがセット）

自動 RTS が有効（アクティブ）のとき、FIFO が準フルになると、UART_RTS_N 出力が強制的に非アクティブ（High）になります。ここで「準フル」とは、FIFO で使用可能なスロットが 2 つということを意味します。UART_RTS_N が別の UART デバイスの UART_CTS_N 入力に接続されている場合、そのもう一方の UART は受信 FIFO に空き領域ができるまで（完全にエンプティになるまで）シリアルデータの送信を停止します。

以下の受信 FIFO しきい値が選択可能です。

- 1
- 1/4
- 1/2
- 「フルまであと 2 キャラクタ」

UART_RTS_N が非アクティブになっても、（データがすでに UART のトランスミッタブロックに入ってしまうと）もう 1 個キャラクタが UART に送信されるので、しきい値を「フルまであと 2 キャラクタ」に設定することで、1 キャラクタの余裕をもって FIFO を最大限に利用できます。

レシーババッファレジスタ（rUart_RBR）を読み出すことにより受信 FIFO が完全にエンプティになると、UART_RTS_N は再びアクティブ（Low）になり、もう一方の UART にデータ送信を続けるように伝えます。

他の項目がすべて選択され、正しく rUart_MCR レジスタがセットされても、bUart_FIFOE により FIFO が禁止されると、自動フロー制御も無効になることに注意してください。自動 RTS が無効の場合、UART_RTS_N は bUart_RTS によってのみ制御されます。

自動 RTS 動作のタイミング図を以下に示します。

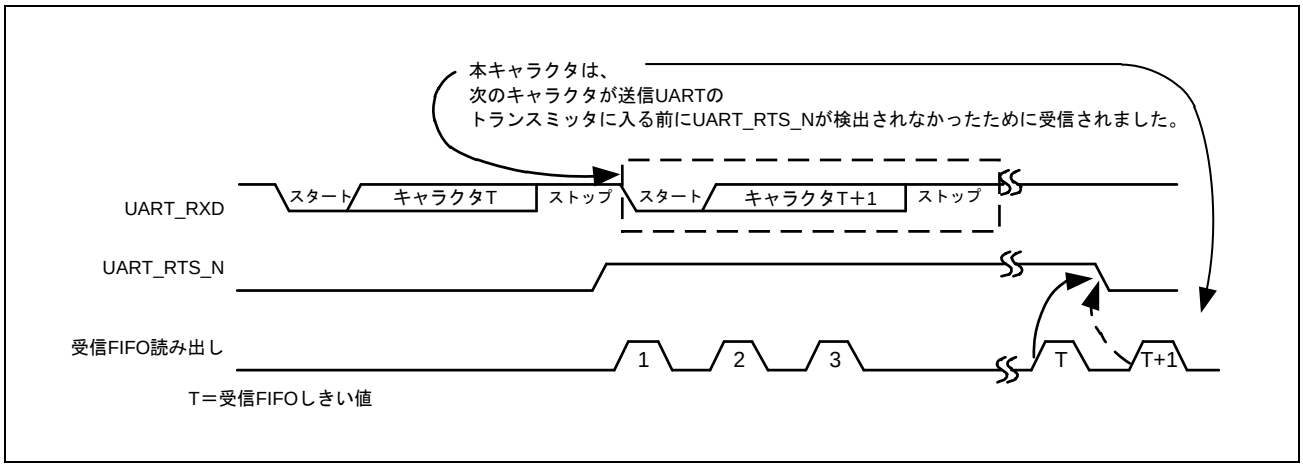


図 1.5　自動 RTS タイミング

自動 CTS は、以下のときアクティブになります。

- rUart_MCR レジスタの bUart_AFCE ビットがセット
- FIFO が許可（bUart_FIFOE ビットがセット）

自動 CTS 有効（アクティブ）時、UART トランスミッタは UART_CTS_N 入力が非アクティブ（High）になると無効になります。これにより、受信側 UART の FIFO のオーバーフローが防止されます。

直前のストップビットの中間前で UART_CTS_N 入力がアクティブの場合、トランスミッタが無効にされる前に別のキャラクタが送信されます。トランスミッタが無効であっても、送信 FIFO は引き続き書き込み可能なため、オーバーフローする可能性もあります。

したがって、本モード使用時は以下の点に留意してください。

- 送信 FIFO がフルかどうか（bUart_TFNF＝0）をチェックするために UART ステータスレジスタが読み出し可能
- 現在の FIFO レベルが rUart_TFL レジスタを介して読み出し可能
- ラインステータスレジスタ（rUart_LSR）を介して「FIFO フル」状態にアクセスするには、プログラマブル THRE 割り込みモードが有効でなければならない

ソフトウェアは、トランスミッタ FIFO への各書き込みの前に「FIFO フル」状態をポーリングすることが可能です。詳細は、「**1.5.1.8 プログラマブル THRE 割り込み**」を参照してください。UART_CTS_N 入力が再びアクティブ（Low）になると、送信が再開します。

他の項目がすべて選択され、bUart_FIFOE により FIFO が禁止されると、自動フロー制御も無効になることに注意してください。自動 CTS が無効のとき、トランスミッタは UART_CTS_N の影響を受けません。

自動 CTS 動作を示すタイミング図を以下に示します。

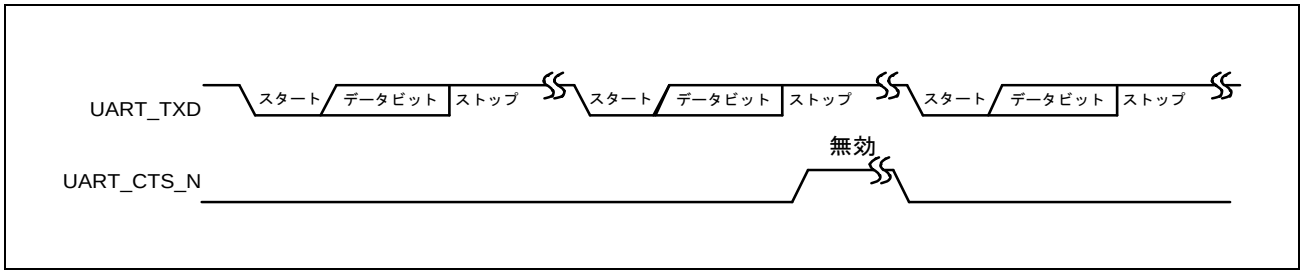


図 1.6　自動 CTS タイミング

1.5.1.8　プログラマブル THRE 割り込み

UART には、システムのパフォーマンスを向上させるプログラマブル THRE 割り込みモードがあり、割り込み許可レジスタ（rUart_IER の bUart_PTIME）を介して有効にすることが可能です。

下図に示すように、FIFO、プログラマブル THRE 割り込みモードおよび THRE 割り込みが有効時、THRE 割り込みおよび UART DMA 要求は、設定された送信 FIFO エンプティしきい値レベル以下になるとアクティブになります。

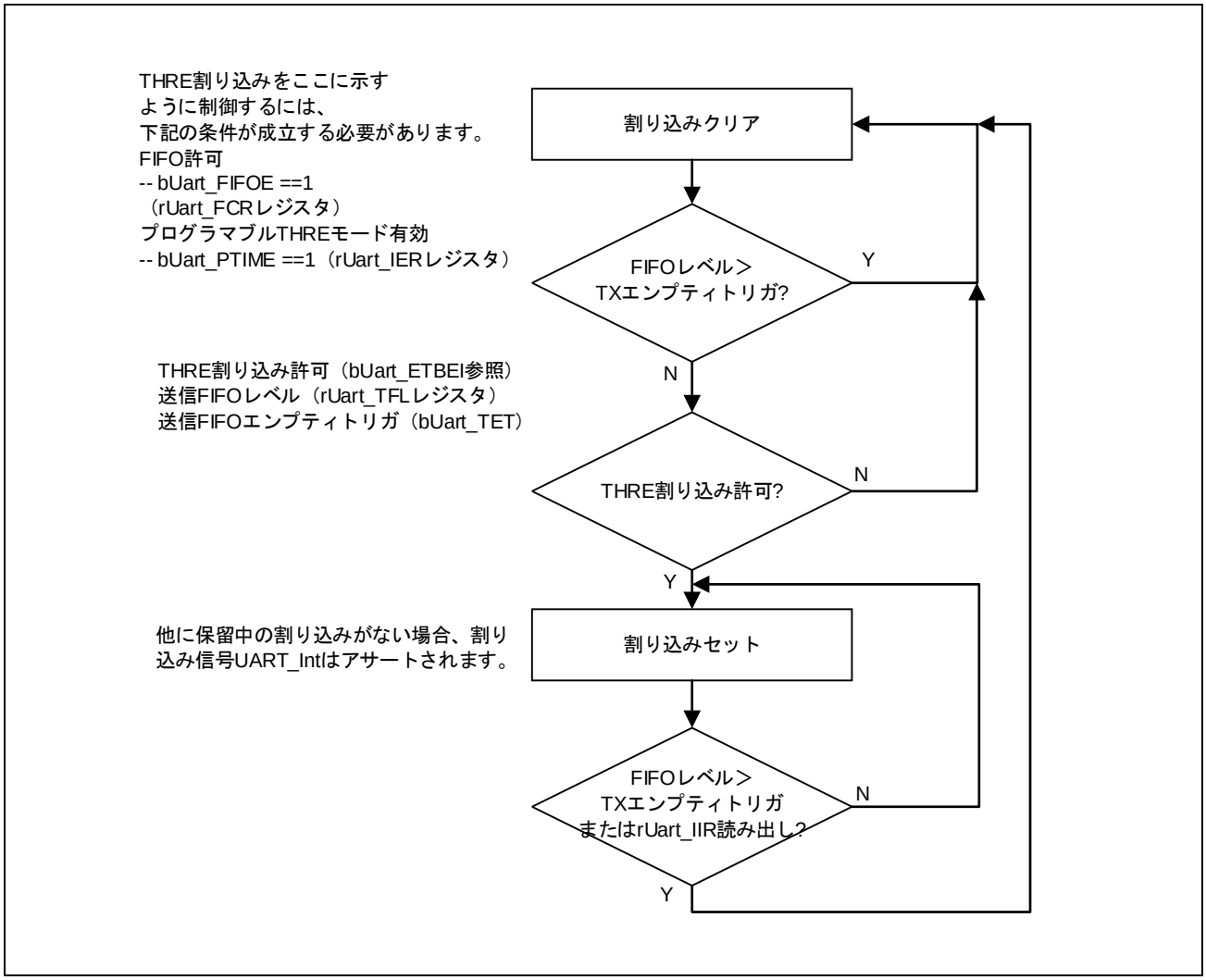


図 1.7　割り込み生成、プログラマブル THRE 割り込みモード＆FIFO 許可のフローチャート

本しきい値トリガレベル（送信 FIFO エンプティトリガ）は bUart_TET ビットに設定されます。エンプティしきい値としては、エンプティ、2、1/4、および 1/2 があります。しきい値設定の詳細については rUart_FCR レジスタを参照してください。

最適のしきい値選択は、システムが新規の送信シーケンスを適時開始できるかどうかにかかっています。送信 FIFO がエンプティになるのを防ぐことによりシステムパフォーマンスは向上します。

割り込みの変更以外に、ラインステータスレジスタ（rUart_LSR レジスタ）の bUart_THRE ビットは、送信 FIFO エンプティ表示から FIFO フルに機能を切り替えることも行います。これによりソフトウェアは、別のキャラクタを書き込む前に bUart_THRE をポーリングすることにより送信シーケンスのたびに FIFO をデータで満たすことができます。その後のフローでは、FIFO が完全にエンプティになるのを待つのではなく、割り込みが発生し送信すべきデータがあるたびに送信 FIFO にデータを満たすことができるようになりま

す。FIFO がエンプティになるまで待つということは、システムがビジーですぐに応答できない場合パフォーマンスの低下につながります。自動フロー制御とともに本モードを有効にするとシステム効率を更に向上させることが可能です。

他の項目がすべて選択され有効の場合でも、bUart_FIFOE ビットにより FIFO が禁止されると、プログラマブル THRE 割り込みモードも無効になります。他の項目により無効の場合、THRE 割り込みおよび bUart_THRE は通常どおり機能し、THR または FIFO はエンプティを意味します。

プログラマブル THRE 割り込みモード以外の場合の THRE 割り込み生成のフローチャートを下図に示します。

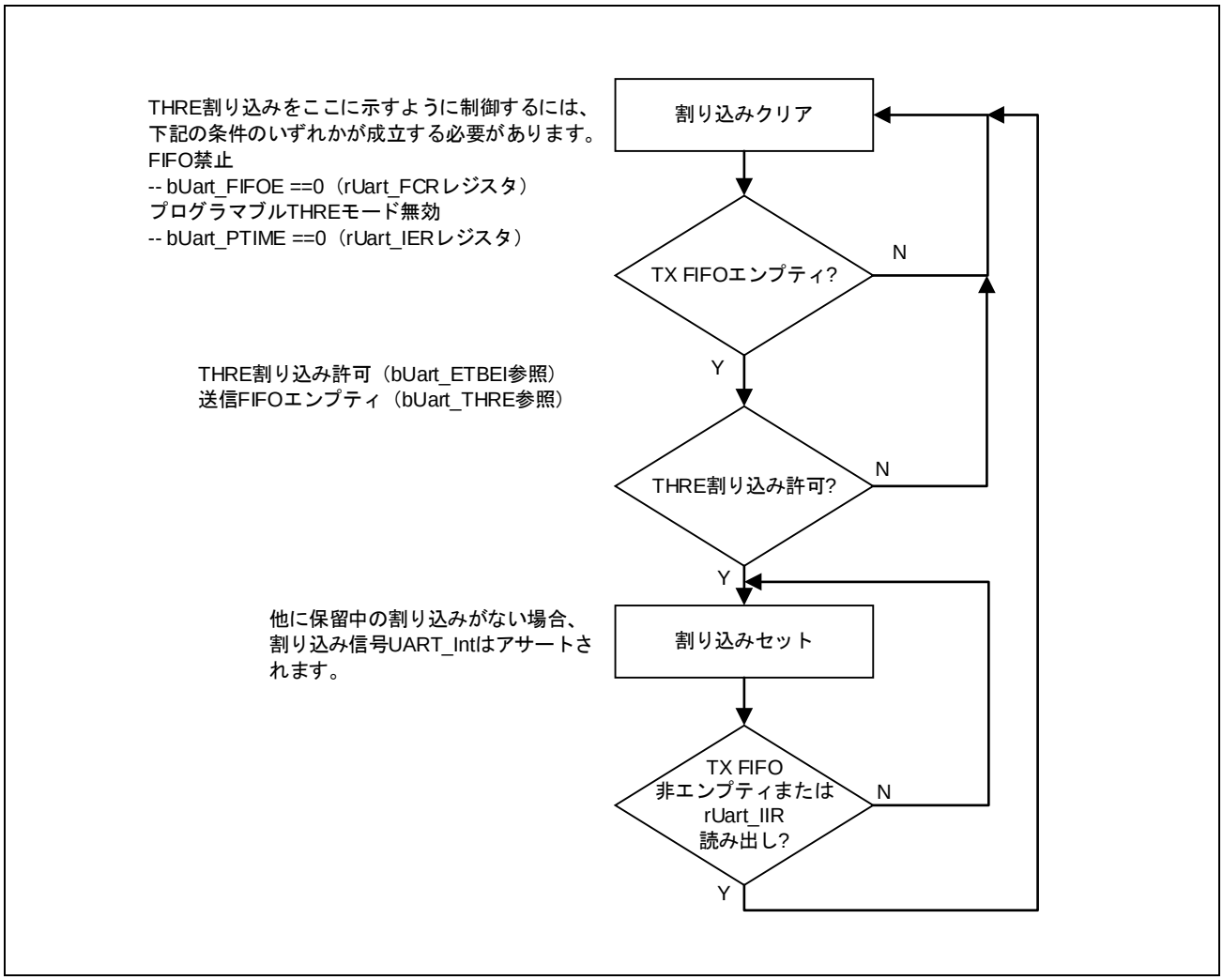


図 1.8　　割り込み生成、プログラマブル THRE 割り込みモードまたは FIFO 禁止のフローチャート

rUart_IIR を読み出すことによる THRE 割り込み解除後、割り込み要因が常に成立していても、内部マスクが THRE 割り込みをデアサートします。本割り込みは、割り込み要因が常に成立する場合、各転送の開始時に再アサートされます。

FIFO が bUart_FIFOE（rUart_FCR レジスタ）を介して禁止されている場合、プログラマブル THRE 割り込みモードも無効になります。

プログラマブル THRE 割り込みモードが無効のときも、THRE 割り込みおよびラインステータスレジスタ（rUart_LSR レジスタ）の bUart_THRE ビットは正常に機能します（両方とも rUart_THR または送信 FIFO がエンプティを反映します）。

1.5.1.9　DMA 管理（UART4、5、6、7、8 のみ）

(1) DMA 動作の概要

これらの UART は、2 本の DMA チャネルを使用します（送信データ用に 1 本、受信データ用に 1 本）。DMA コントローラは、周辺機能フローコントローラモードに設定する必要があります。

これらの UART は、データの読み出し準備ができたときまたは送信 FIFO がエンプティになったことを示すために追加の DMA インタフェース信号を設定できます。

- 送信 DMA 要求は以下の条件でアサートされます。
  - 送信保持レジスタ（rUart_THR）が非 FIFO モードでエンプティのとき。非 FIFO モードでは DEST_BURST_SIZE は、2’b00（1 バイト）にプログラムされていなければなりません。
  - プログラマブル THRE 割り込みモードが無効で、送信 FIFO が、FIFO モード時にエンプティのとき
  - プログラマブル THRE 割り込みモードが有効で、送信 FIFO が、設定されたしきい値以下のとき
- 受信 DMA 要求は以下の条件でアサートされます。
  - 非 FIFO モードで受信バッファレジスタ（rUart_RBR）に単一のキャラクタがあるとき。非 FIFO モードでは SRC_BURST_SIZE は、2’b00（1 バイト）にプログラムされていなければなりません。
  - 受信 FIFO が、FIFO モードで、設定されたトリガレベル以上のとき

DMA ハンドシェイク信号を使用する場合、UART は、要求の完了を識別し要求を削除する際に内部状態およびレベル値に依存する必要はありません。その代わり、DMA 送信および受信要求のデアサートは、それぞれ DMA ACK 応答のアサートにより制御されます。

UART の DMA 信号が設定されている場合、データフロー（転送長）の管理は UART にあり、設定されたバーストトランザクション長&ブロックサイズにより制御されます。

FIFO およびプログラマブル THRE 割り込みモードが設定された UART に対するハンドシェイク信号は、以下の DMA フローで説明します。

DMA コントローラは、周辺機能フローコントローラモードに設定する必要があります。

UART は送受信するデータ項目数（ブロックサイズ）が設定されている必要があります。これは、送信 FIFO および受信 FIFO それぞれについて、UART の rUart_TDMACR レジスタおよび rUart_RDMACR レジスタの DEST_BLOCK_SIZE フィールドおよび SRC_BLOCK_SIZE フィールドに設定されます。ブロックは複数のトランザクションに分割され、それぞれのトランザクションは UART からの要求で開始されます。

DMA コントローラおよび UART は、各 DMA 要求に対してバーストトランザクションにより転送されるデータバイト数（この場合 UART FIFO エントリ）も設定する必要があります。これは、バーストトランザクション長と呼ばれ、以下に設定されます。

DMA コントローラ：

送信 FIFO および受信 FIFO それぞれについて、DMA CTL[n]レジスタの DEST_MSIZE フィールドおよび SRC_MSIZE フィールド

UART：

送信 FIFO および受信 FIFO それぞれについて、UART の rUart_TDMACR レジスタの DEST_BURST_SIZE フィールドおよび rUart_RDMACR レジスタの SRC_BURST_SIZE フィールド

注　意

バーストトランザクションサイズは、DMA コントローラおよび UART ともに同じ値でなければなりません。

(2) 送信ウォーターマークレベルおよび送信 FIFO アンダーフロー

UART シリアル転送中、送信 FIFO 内のエントリ数が rUart_FCR レジスタの送信エンプティトリガ（bUart_TET）の復号化されたレベル以下の場合、送信 FIFO 要求が DMA に対して行われます。

本レベルをウォーターマークレベルと言います。DMA は、送信 FIFO バッファに、CTL[n].DEST_MSIZE＝DEST_BURST_SIZE 長のバーストデータを書き込むことにより応答します。

データは、送信 FIFO がシリアル転送を連続的に行うために十分な頻度で DMA からフェッチされる必要があります。すなわち、FIFO エンプティになりそうなとき、DMA 要求がトリガされなければなりません。トリガされないと、FIFO にデータがなくなってしまいます（アンダーフロー）。このような状況が起こらないようにするために、ソフトウェアはウォーターマークレベルを正しく設定する必要があります。

(3) 送信ウォーターマークレベルの選択

想定した例を考えてみます。

DEST_BURST_SIZE＝DMAC.CTL[n].DEST_MSIZE＝FIFO_DEPTH − bUart_TET

ここで、DMA バーストで転送されるデータアイテムの数は、送信 FIFO 内の空きスペースと同じです。2 つのウォーターマークレベル設定について考えてみます。

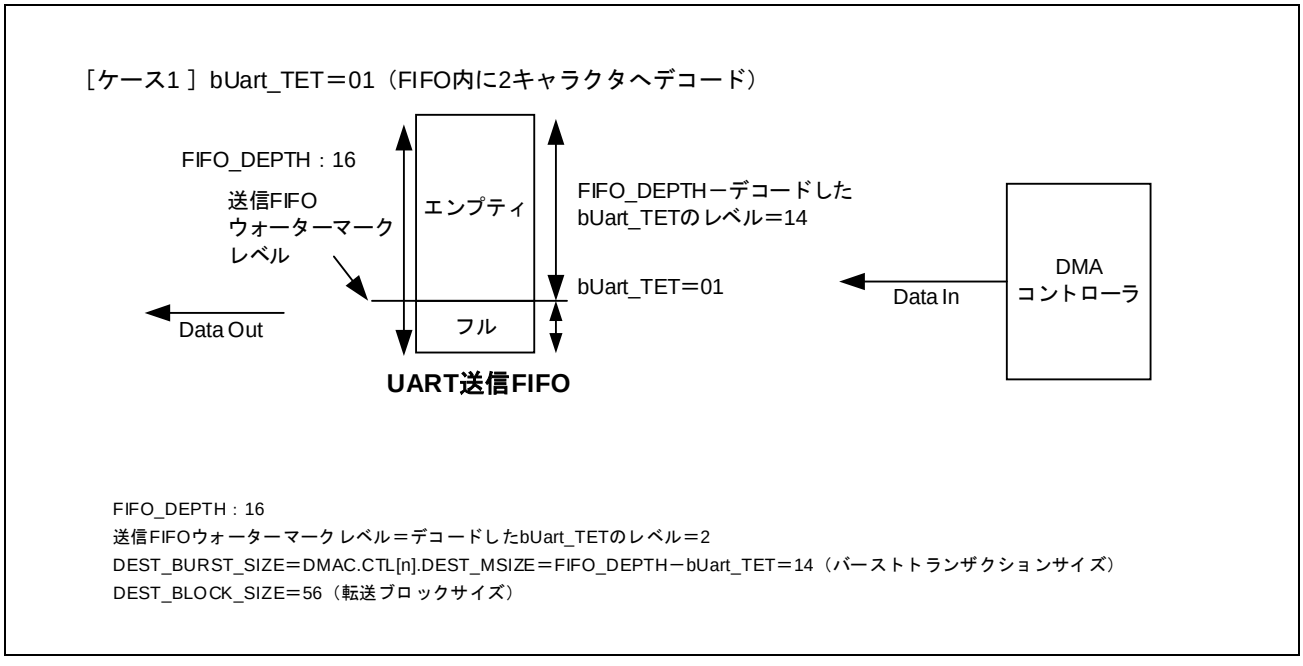


図 1.9　　ケース 1：送信ウォーターマークレベル

したがって、必要なバーストトランザクション数は、バーストあたりのデータアイテム数で割ったブロックサイズとなります。

DEST_BLOCK_SIZE/DEST_BURST_SIZE＝56/14＝4

DMA ブロック転送でのバーストトランザクション数は 4 です。しかし、bUart_TET の復号化されたレベルであるウォーターマークレベルは極めて低くなります。したがって、UART シリアル送信ラインがデータを送信しなければならないにもかかわらず、送信 FIFO にデータが残っていない場合、UART アンダーフローが起きる可能性が高くなります。これは、DMA が、送信 FIFO が空きになる前に DMA 要求を処理する時間がなかったために起こります。

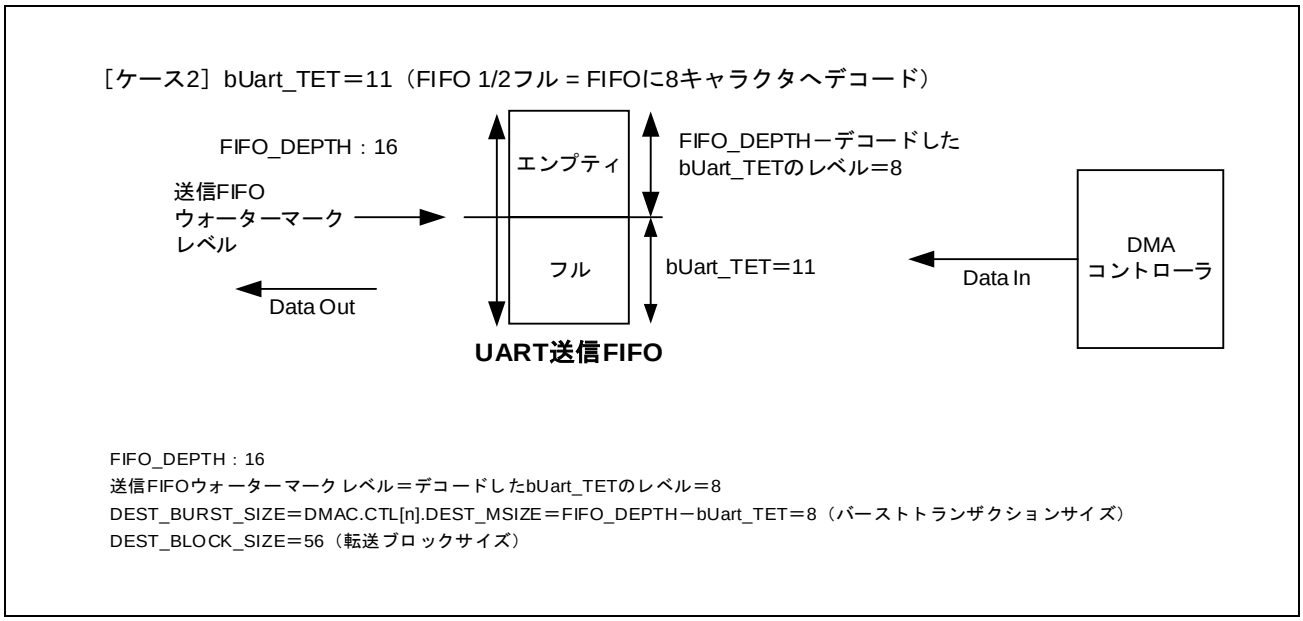


図 1.10　　ケース 2：送信ウォーターマークレベル

ブロック n 内のバーストトランザクション数：

DEST_BLOCK_SIZE/DEST_BURST_SIZE＝56/8＝7

本ブロック転送では、DMA ブロック転送のバーストトランザクション数は 7 です。しかし、bUart_TET の復号化されたレベルであるウォーターマークレベルは高くなります。したがって、DMA コントローラは UART の送信 FIFO が空きになる前に宛先バーストトランザクション要求を処理する時間が十分にあるので、UART アンダーフローが起こる可能性は低くなります。

したがって、2 番目のケースは、1 ブロックにつき送信可能なバーストトランザクション数が多くなり、アンダーフローの確率が低くなります。これは、前のケースよりも 1 ブロックにつきより多くの要求バーストを送ることになり、バス使用率が悪くなります。

したがって、ウォーターマークレベル選択の目的は、アンダーフローになる確率を許容可能なレベルに抑えながら、ブロックあたりのトランザクション数を最小限に抑えることにあります。実際、これは、UART が送信するデータ速度に DMA が宛先バースト要求に応答できる速度を合わせる機能です。

たとえば、チャネルを DMA の最優先チャネルに格上げし、DMA マスタインタフェースをバスレイヤの最優先マスタに格上げすると、DMA コントローラがバーストトランザクション要求に応答可能な速度を上げることになります。これは、逆に、ユーザがウォーターマークレベルを下げることを可能にします。ウォーターマークレベルを下げることによりバス効率を上げることができます。

(4) DEST_MSIZE の選択および送信 FIFO オーバーフロー

UART 送信 FIFO に宛先バースト要求を処理するための十分なスペースがない場合、オーバーフローとなる可能性があります。

したがって、適した動作のためには、以下のように設定する必要があります。

- DMAC.CTL[n].DEST_MSIZE＝DEST_BURST_SIZE＝4
- bUart_TET＝2’b11（FIFO 1/2 フルに設定）

もしくは

- DMAC.CTL[n].DEST_MSIZE＝DEST_BURST_SIZE＝8
- bUart_TET＝2’b11（FIFO 1/2 フルに設定）

注　意

UART が DMA バースト転送中、UART シリアル送信ライン上で 1 つ以上のデータ送信に成功した場合、送信 FIFO は、転送完了時にフルになりません。

(5) 受信ウォーターマークレベルおよび受信 FIFO オーバーフロー

UART シリアル転送中、受信 FIFO 内のエントリ数が rUart_FCR のレシーバトリガ（bUart_RCVR）の復号化されたレベル以上になると、受信 FIFO 要求が DMAC に対して行われます。本レベルをウォーターマークレベルと言います。

DMAC は、受信 FIFO バッファから、CTL[n].SRC_MSIZE＝SRC_BURST_SIZE 長のバーストデータを読み出すことにより応答します。

データは、受信 FIFO がシリアル転送を連続的に受け入れるために十分な頻度で DMAC によりフェッチされる必要があります。すなわち、FIFO にデータが満たされ始めると、DMA 転送が要求されます。さもなければ、FIFO はデータでフルになってしまいます（オーバーフロー）。このような状況が起こらないようにするために、ウォーターマークレベルを正しく設定する必要があります。

(6) 受信ウォーターマークレベルの選択

上述の送信ウォーターマークレベルの選択と同様に、bUart_RCVR の復号化されたレベルである受信ウォーターマークレベルは、オーバーフローの確率を最小限に抑えるように設定する必要があります。これは、ブロックあたりに必要な DMA バーストトランザクション数とオーバーフローが起こる確率のトレードオフになります。

(7) SRC_MSIZE の選択および受信 FIFO アンダーフロー

送信元バースト要求を処理する十分なデータがない場合、アンダーフローとなる可能性があります。

したがって、適した動作のためには、以下のように設定する必要があります。

- DMAC.CTL[n].SRC_MSIZE＝SRC_BURST_SIZE＝4
- bUart_RCVR= 2’b01（FIFO 1/4 フルに設定）

もしくは

- DMAC.CTL[n].SRC_MSIZE＝SRC_BURST_SIZE＝8
- bUart_RCVR＝2’b10（FIFO 1/2 フルに設定）

注　意

UART がバースト転送中、UART シリアル受信ライン上で 1 つ以上のデータ受信に成功した場合、受信 FIFO は、送信元バーストトランザクション完了時にエンプティになりません。

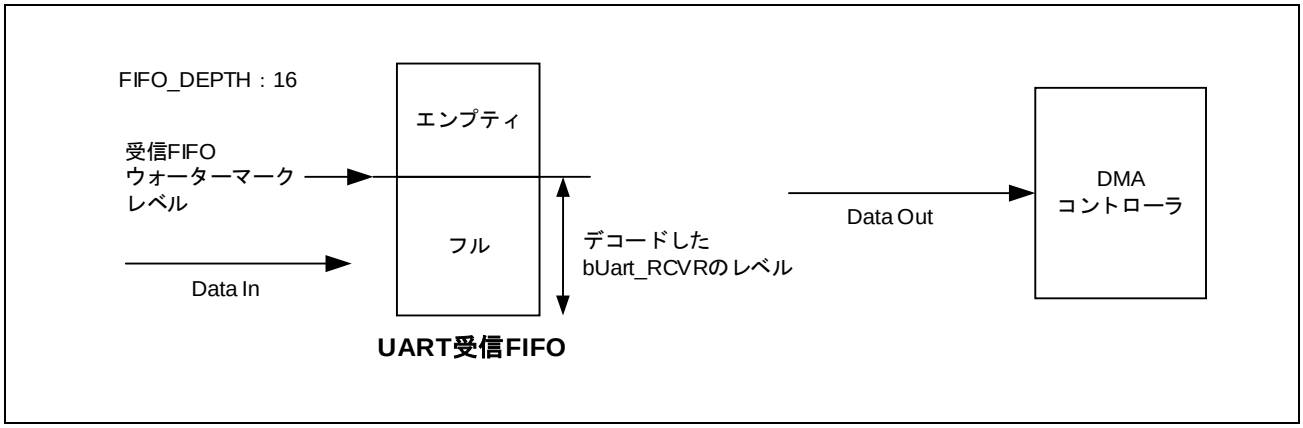


図 1.11　　ケース 3：受信ウォーターマークレベル

(8)　受信モードで DMA 接続した UART でのデッドロックの可能性

DMA バーストトランザクション長が UART 受信 FIFO しきい値（rUart_FCR レジスタの bUart_RCVR ビット）と同じ場合、UART_RTS_N がデアサートされた後にキャラクタが受信されるとデッドロックになる危険があります。

UART は、受信 FIFO がしきい値に達した場合、UART_RTS_N をデアサートします。しかしながら、ラインの相手コンポーネントは、UART_CTS_N 入力のデアサートを検出する前に新しいキャラクタを送信し始めることが可能です。これが起こると、キャラクタ送信は正常終了します。これは、追加のキャラクタを受信し受信 FIFO に入れる（フルの場合を除く）ことを意味します。

UART_RTS_N がデアサートされるのと同時に、UART は、受信 DMA 要求をアサートし、DMA コントローラに DMA バーストトランザクションを要求します。DMA コントローラがバーストトランザクション（受信 FIFO しきい値と同じ長さ）を完了すると、受信 FIFO にキャラクタが 1 つ残り、UART_RTS_N が再びアサートされないようにします。

UART は DMA コントローラに DMA シングルトランザクションを要求します。しかしながら、DMAC はシングルトランザクション領域で動作している場合を除き、シングルトランザクション要求を無視します。

その後デッドロック状態になります。

- UART_RTS_N 信号はデアサートされているので、UART はこれ以上キャラクタを受信しません。
  受信 FIFO を再びしきい値レベルまで満たし、DMAC に新しいバーストトランザクション要求を行うために、受信 FIFO にデータを入れることはできません。シングルトランザクション要求のみ生成可能です。
- DMAC は、シングルトランザクション領域に達しないかぎり、シングルトランザクション要求を無視し、受信 FIFO からの読み出しを行いません。
  受信 FIFO はエンプティにすることはできないので、UART_RTS_N 信号は再びアサートされません。

デッドロック状態は、以下により回避可能です。

- 受信 FIFO しきい値レベルが DMA バーストトランザクションサイズより小さい値に設定されている場合。
  このことにより、もう一つ追加でキャラクタが受信され、それに従って UART_RTS_N がアサートされたかどうかに関係なく、受信 FIFO は、DMA バーストトランザクションが完了すると常にエンプティになります。
- DMA ブロックサイズが DMA バーストトランザクション長の 2 倍より小さい値に設定されている場合。
  このことにより、DMA バーストトランザクション完了後、DMAC は必ずシングルトランザクション領域に入ります。次に DMAC は UART からのシングルトランザクション要求を受け入れ、受信 FIFO を空にします。

本デッドロック状態は、通常の動作条件では、頻繁には起こらないと考えられます。

この場合タイムアウト割り込みが発生し、本デッドロック状態を検出するために使用可能です。

1.5.1.10　MODBUS 管理用トランシーバ&レシーバタイムアウト

MODBUS シリアルラインプロトコルは、マスタ／スレーブプロトコルです。本プロトコルは、OSI モデルの第 2 層で実行されます。

マスタ／スレーブ型のシステムは、明示的なコマンドを複数の「スレーブ」ノードのいずれかに発行し、応答を処理する 1 つのマスタノードから構成されます。スレーブノードは、通常マスタノードからの要求がある場合を除いてデータを送信することはなく、他のスレーブとの通信も行いません。

物理レベルでは、MODBUS over Serial Line システムは別々の物理インタフェース（RS485、RS232）を使用する可能性があります。送信データイネーブル（DE）信号は UART[m]_RTS_N もしくは GPIO を使用して、ソフトウェアによって制御します。

2 線式インタフェースが最も一般的です。アドオンオプションで、TIA/EIA-485（RS485）の 4 線式インタフェースも実装可能です。短いポイントツーポイント通信のみ必要な場合は、TIA/EIA-232-E（RS232）シリアルインタフェースも使用可能です。

通常の 2 線式インタフェースを下図に示します。

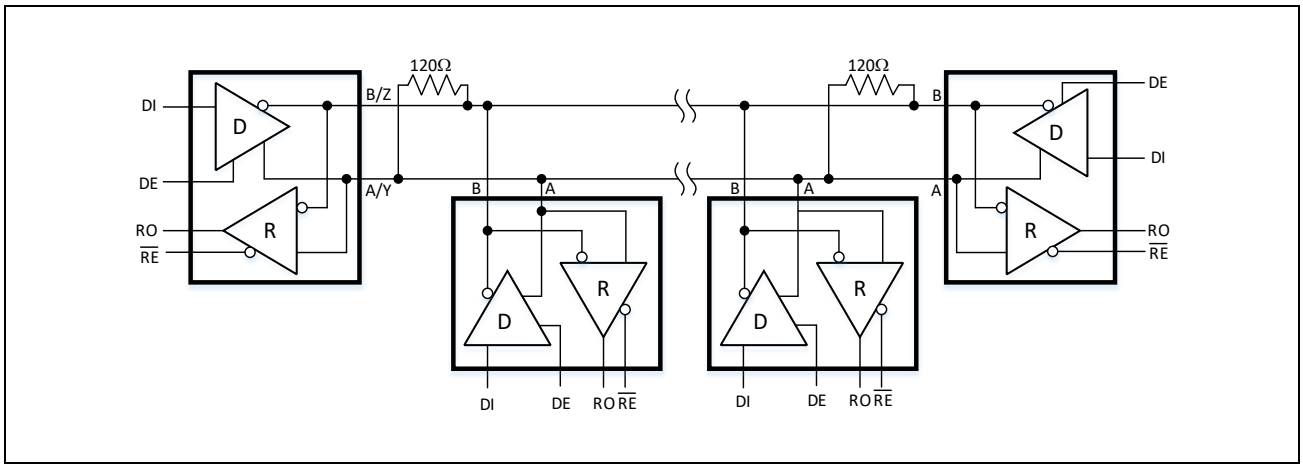


図 1.12　　通常の半二重 RS485 ネットワーク

MODBUS メッセージは、送信デバイスにより、開始ポイントと終了ポイントが分かっているフレームに入れられます。これにより、新規フレームを受信するデバイスは、メッセージの始まりで動作開始し、メッセージが完了した場合それを認識できます。不完全なメッセージを検出し、エラー結果を設定する必要があります。

RTU モードでは、メッセージフレーム間は、少なくとも 3.5 キャラクタ時間のサイレント区間で分離されます。以下の各項では、本時間間隔を t3.5 と呼称します。下図参照。

メッセージフレーム全体は、連続したキャラクタの流れとして送信される必要があります。

2 つのキャラクタ間に 1.5 キャラクタ時間（t1.5）以上のサイレント区間がある場合、そのメッセージフレームは不完全と宣言され、レシーバはそのようなメッセージフレームを廃棄しなければなりません。

［備考］

RTU 受信ドライバの実装は、t1.5 タイマおよび t3.5 タイマによる頻繁な割り込みの管理を意味します。高速の通信ボーレートの場合、これは CPU への重い負荷になります。

したがって、ボーレートが 19200 ボー以下の場合は、これら 2 つのタイマを厳格に守ります。ボーレートが 19200 ボーを超える場合は、2 つのタイマに固定値を使用します。キャラクタ間タイムアウト（t1.5）には 750µs、フレーム間遅延（t3.5）には 1.750ms を推奨します。

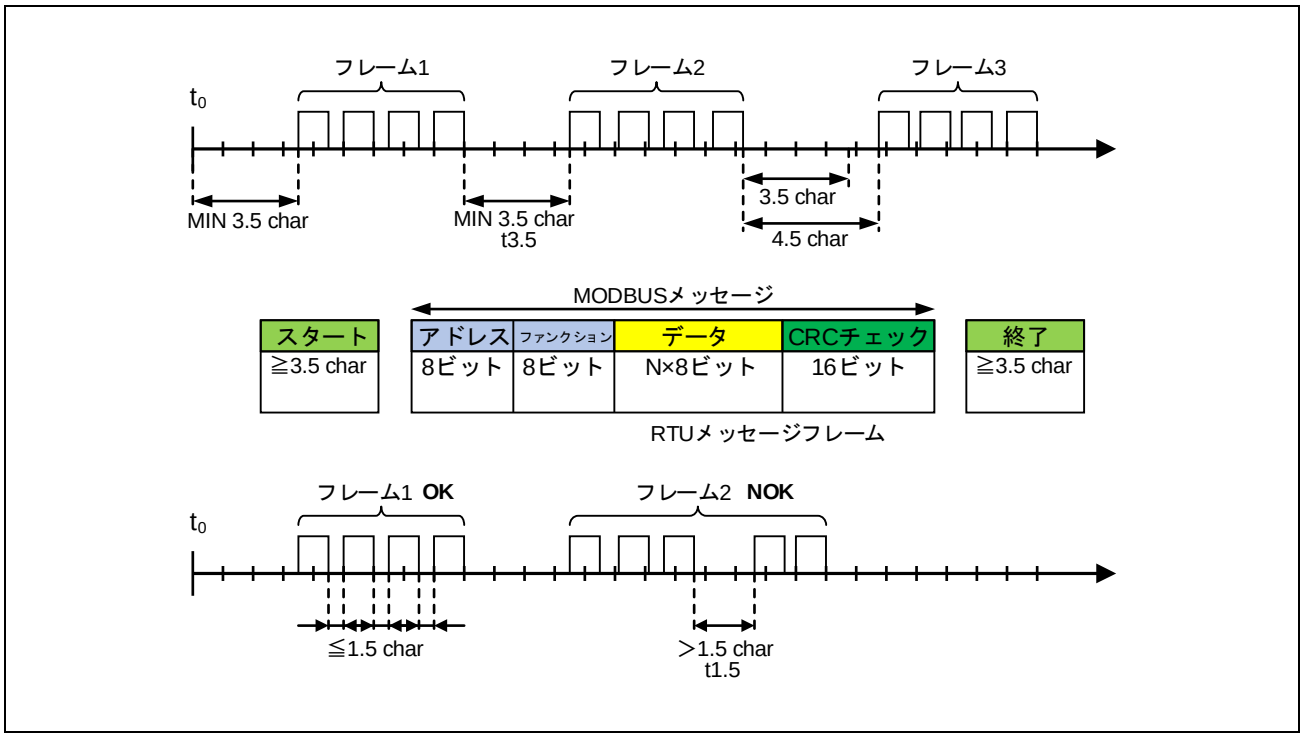


図 1.13　　MODBUS フレーム＆制限タイミング

注　意

正常に動作するには、以下の式が成立しなければなりません。

UART_PCLK ≧ 4×（UART_SCLK／ボーレート除数）

UART_PCLK＝7.5MHz かつ UART_SCLK＝83.33MHz の場合

- 最小ボーレート除数値＝（83.33／7.5）×4＝45
- 最大ボーレートは、UART_SCLK／（16×45）＝115K ボー

(1) レシーバタイムアウト

レシーバタイムアウトは、MODBUS リンクでのフレーム間時間（t1.5、t3.5）処理をサポートします。

2 つのレシーバタイムアウトがあります。

本機能は、UART_RXD ライン上でアイドル状態を検出します（サイレント期間検出）。

（レシーバが新しいキャラクタを待つ）タイムアウト遅延期間は、タイムアウトレジスタ（rUart_TO）の bUart_TO[n]（n＝0、1）フィールド内に設定されます。本時間は、rUart_TO[n]×「ボークロック周期」と等しくなります。

タイムアウト検出時（タイムアウトカウンタ[n]（n＝0、1）での 0 値）、ステータスレジスタ（rUart_STATUSTO）の bUart_TIMEOUTInt[n]（n＝0、1）ビットが立ち上がり、（マスクされていない場合）割り込みを発生することにより、ドライバにフレームの終わりを表示します。

「**1.5.1.1 UART（RS232）シリアルプロトコル**」を参照してください。

ボークロックは、ボーレート除数（bUart_DLL および bUart_DLH）で設定されます。

注　意

rUart_DLL および／または rUart_DLH レジスタがビジー状態中に書き込まれると、タイムアウト機能は、rUart_DLL レジスタおよび rUart_DLH レジスタが非ビジー状態の時に書き込まれるまで正常に動作しません。

bUart_TO[n]（n＝0、1）フィールドが 0 に設定されている場合、レシーバタイムアウトは無効で、タイムアウトは検出されません。rUart_STATUSTO の bUart_TIMEOUTInt[n]（n＝0、1）ビットは、0 のままです。

bUart_TO[n]（n＝0、1）フィールドが 0 に設定されていない場合、カウンタ[n]は、bUart_TO[n]（n＝0、1）に設定されている値を 8 ビットカウンタに入力します。本カウンタは、各ビット期間でデクリメントされ、新しいキャラクタが受信されるたびに再ロードされます。カウンタが 0 になるとステータスレジスタ（rUart_STATUSTO）の bUart_TIMEOUTInt[n]（n＝0、1）ビットが立ち上がり、割り込みを起動します（マスクされていない場合）。

カウンタは 0 になると LOAD コマンドを受け取るまでロック状態になります（以下概要参照）。ステータスレジスタ（rUart_STATUSTO）の bUart_TIMEOUTStatus[n]（n＝0、1）ビットは、タイムアウトカウンタ[n]（n＝0、1）の状態を見るために使用可能です。本レジスタは、通常、割り込みサービスルーチンまたはポーリング中にソフトウェアドライバにより読み出されます。

- 1’b0：タイムアウトカウンタの値が“0”以外
- 1’b1：タイムアウトカウンタの値が“0”

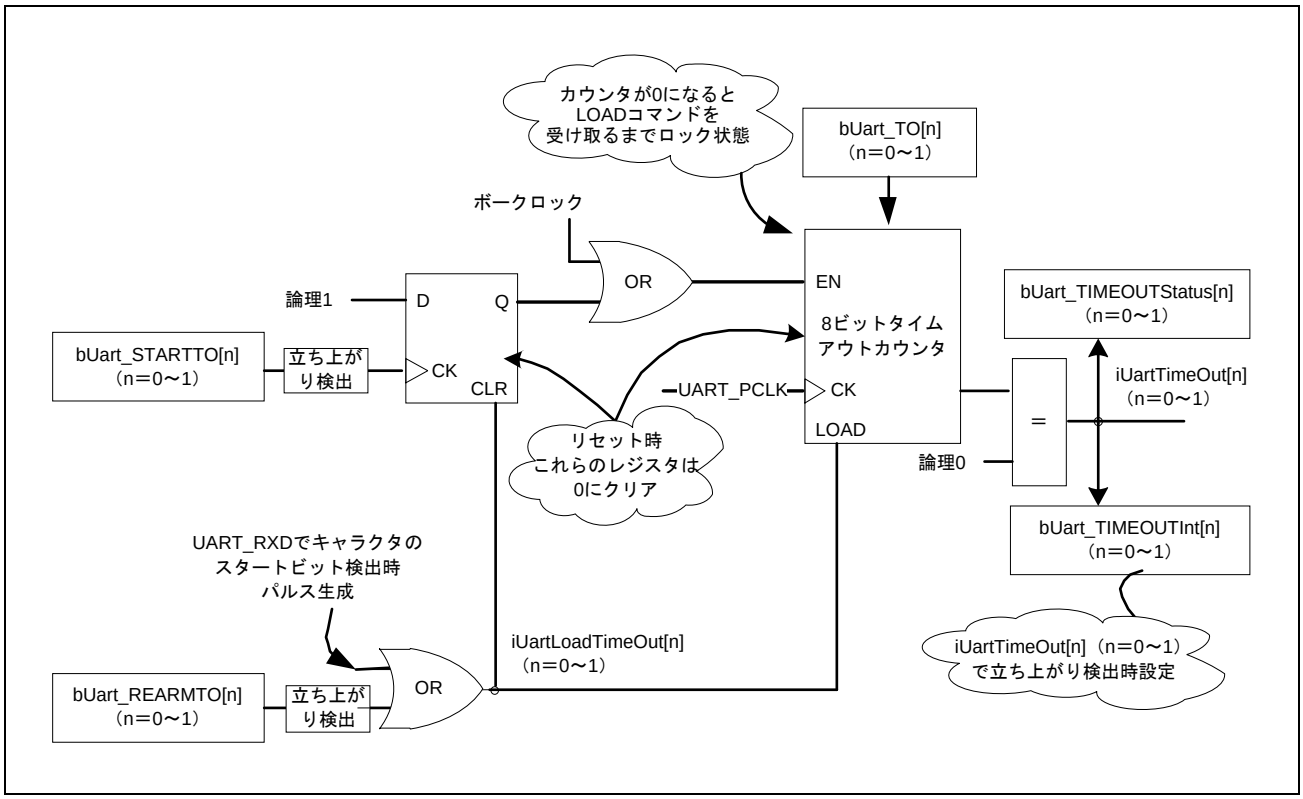


図 1.14　　レシーバタイムアウトの概要

bUart_STARTTO[n]（n＝0、1）（立ち上がり検出）により、タイムアウトカウンタは 0 になるまで UART_PCLK でカウントダウンされ、最初のキャラクタを受信するまで停止します。本機能により、UART_RXD でのアイドル状態検出時の次フレームの開始待ちが可能になります。

キャラクタのスタートビット検出あるいはソフトウェアで bUart_REARMTO[n]（n＝0、1）を 1 にセット（立ち上がり検出）することで、タイムアウトカウンタは、bUart_TO[n]（n＝0、1）の値からカウントダウンを開始します。

詳細については、「**図 1.16　レシーバ＆トランシーバのタイムアウト（n＝0～3）タイミング**」を参照してください。

(2) トランシーバタイムアウト

トランシーバタイムアウトは、MODBUS リンクでのフレーム間時間（t1.5、t3.5）処理をサポートします。

2 つのトランシーバタイムアウトがあります。

本機能は、UART_TXD ライン上でアイドル状態を検出します（サイレント期間検出）。

（トランシーバが新しいキャラクタを待つ）タイムアウト遅延期間は、タイムアウトレジスタ（rUart_TO）の bUart_TO[n]（n＝2、3）フィールド内に設定されます。本時間は、rUart_TO[n]×「ボークロック周期」と等しくなります。

タイムアウト検出時（タイムアウトカウンタ[n]（n＝2、3）での 0 値）、ステータスレジスタ（rUart_STATUSTO）の bUart_TIMEOUTInt[n]（n＝2、3）ビットが立ち上がり、（マスクされていない場合）割り込みを発生することにより、ドライバにフレームの終わりを表示します。

bUart_TO[n]（n＝2、3）フィールドが 0 に設定されている場合、タイムアウトカウンタのクロックは停止され、カウンタは現在の値を保持します。rUart_STATUSTO の bUart_TIMEOUTInt[n]（n＝2、3）ビットは現在の値を保持します。

bUart_TO[n]（n＝2、3）フィールドが 0 に設定されていない場合、カウンタ[n]は、bUart_TO[n]（n＝2、3）に設定されている値を 8 ビットカウンタに入力します。本カウンタは、各ビット期間でデクリメントされ、新しいキャラクタが送信されるたびに再ロードされます。カウンタ[n]が 0 になるとステータスレジスタ（rUart_STATUSTO）の bUart_TIMEOUTInt[n]（n＝2、3）ビットが立ち上がり、割り込みを起動します（マスクされていない場合）。

カウンタ[n]は 0 になると LOAD コマンドを受け取るまでロック状態になります（以下概要参照）。ステータスレジスタ（rUart_STATUSTO）の bUart_TIMEOUTStatus[n]（n＝2、3）ビットは、タイムアウトカウンタ[n]（n＝2、3）の状態を見るために使用可能です。

- 1’b0：タイムアウトカウンタの値が“0”以外
- 1’b1：タイムアウトカウンタの値が“0”

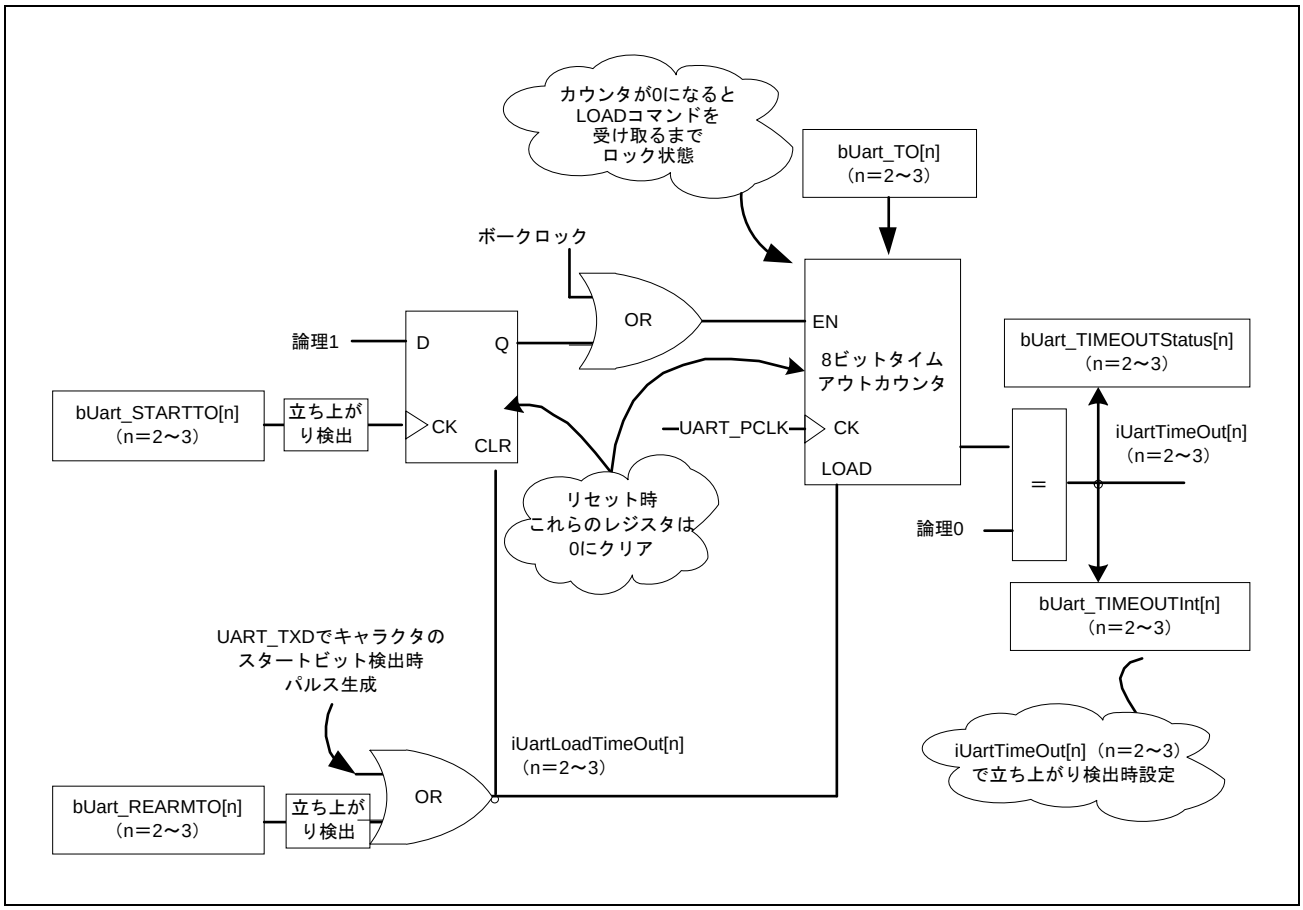


図 1.15　　トランシーバタイムアウトの概要

bUart_STARTTO[n]（n＝2、3）（立ち上がり検出）により、タイムアウトカウンタは 0 になるまで UART_PCLK でカウントダウンされ、最初のキャラクタを送信するまで停止します。本機能により、UART_TXD でのアイドル状態検出時の次フレームの開始待ちが可能になります。

キャラクタのスタートビット検出あるいはソフトウェアで bUart_REARMTO[n]（n＝2、3）を 1 にセット（立ち上がり検出）することで、タイムアウトカウンタは、bUart_TO[n]（n＝2、3）の値からカウントダウンを開始します。

詳細については、「**図 1.16　レシーバ＆トランシーバのタイムアウト（n＝0～3）タイミング**」を参照してください。

(3) タイムアウトカウンタのタイミング

タイムアウトカウンタ[n]（n＝0～3）のタイミングを下図に示します。

スタートビット検出により、rUart_TO レジスタの設定値がロードされ、カウントダウンを開始します。

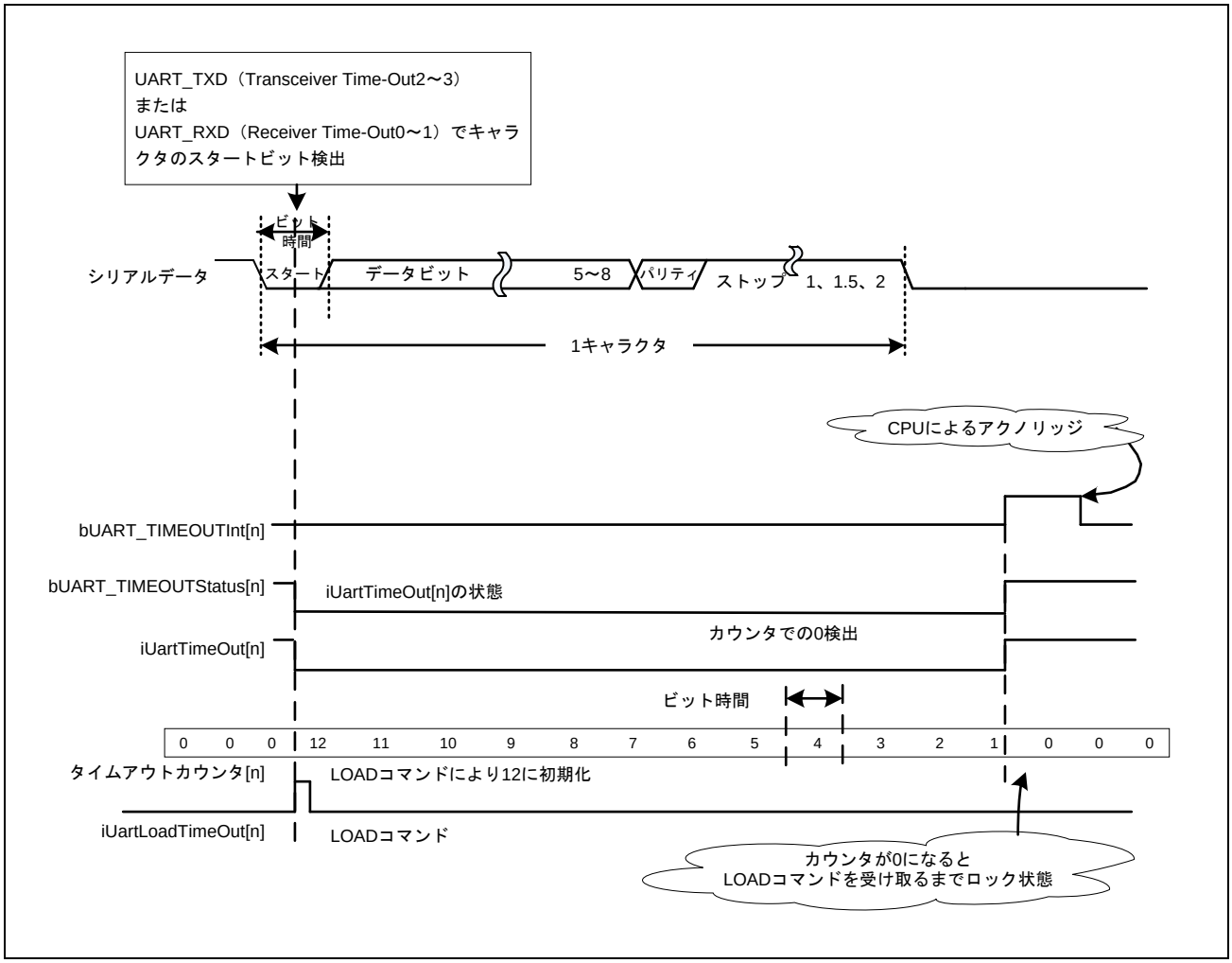


図 1.16　レシーバ＆トランシーバのタイムアウト（n＝0～3）タイミング

1.5.2　使用上の注意事項

「2 線式」インタフェース使用時、半二重モードで「DE（データイネーブル）」信号を管理する必要があります。

内部の UART_RTS_N 信号を使用して、データイネーブル信号を接続することが可能です。内部の iUART_DE 信号からの出力は使用できません。

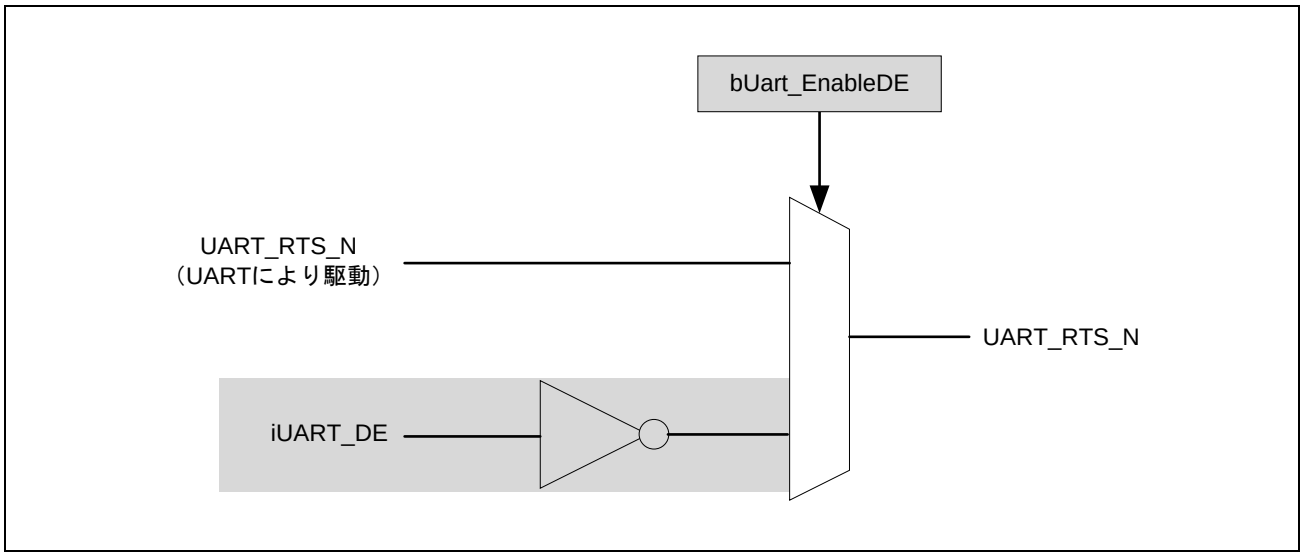


図 1.17　半二重モードでの UART_RTS_N 管理

第2章　SPI

2.1　概要

RZ/N1 では、SPI マスタ専用の 4 ブロックおよび SPI スレーブ専用の 2 ブロックを提供します。

- マスタモード
  - SPI1、SPI2、SPI3、SPI4 のみ
- スレーブモード
  - SPI5、SPI6 のみ
- 送信 FIFO（TX FIFO）16 ビットの 16 ワード
- 受信 FIFO（RX FIFO）16 ビットの 16 ワード
- シリアルクロックビットレート、データ転送のシリアルビットレートの動的制御
  - マスタモードのみ
- プログラマブルなフレームデータサイズ（4～16 ビット）
- スレーブセレクト数
  - SPI マスタモード 4 本、SPI1、SPI2、SPI3、SPI4
  - SPI スレーブモード 1 本、SPI5、SPI6
- 選択可能なシリアルインタフェース動作
  - モトローラ SPI
  - テキサスインスツルメンツ同期式シリアルプロトコル
  - ナショナルセミコンダクターMicrowire
- DMA 接続
  - 周辺機能フローコントローラモード
  - DMA チャネルは 2 本使用可（送信用に 1 本、受信用に 1 本）

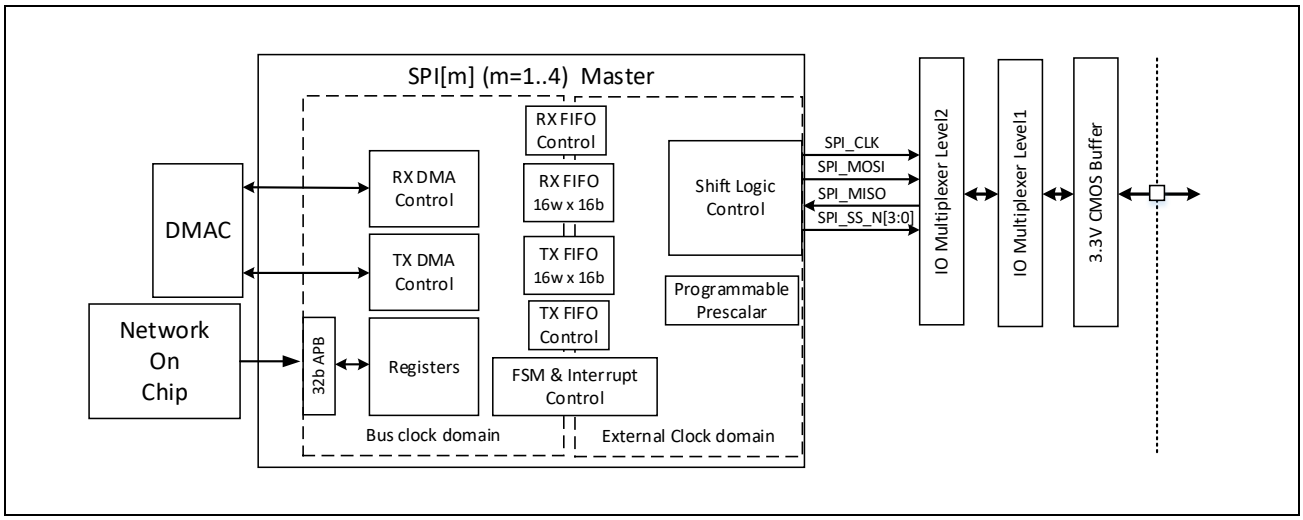


図 2.1　　SPI マスタの概要

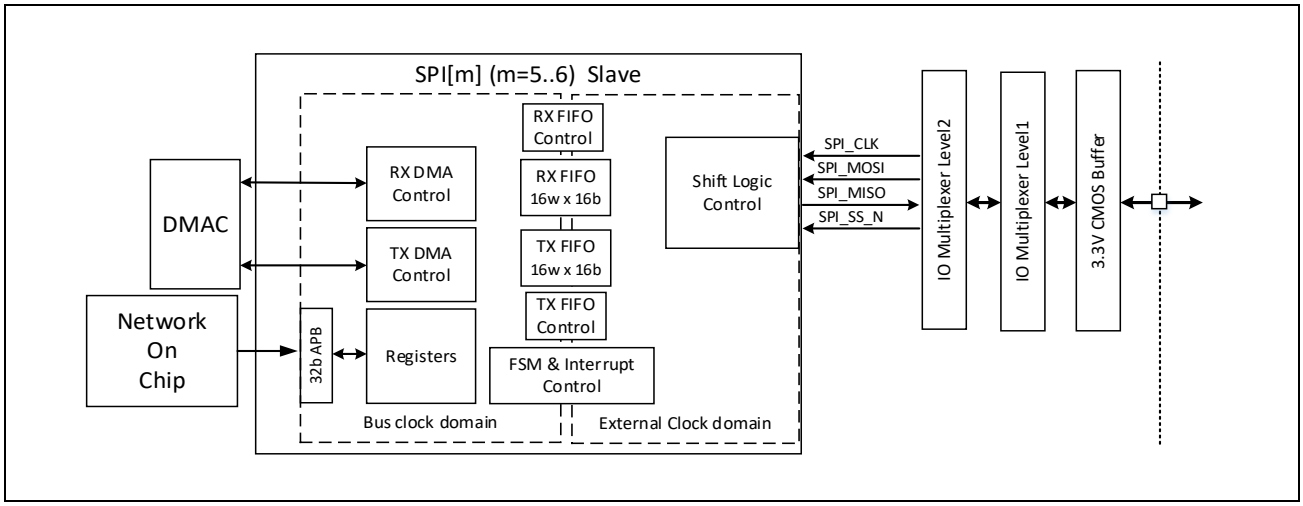


図 2.2　　SPI スレーブの概要

2.2　信号インタフェース

信号名	入出力	説明
クロック		
SPI[m]_PCLK	入力	内部バスクロック（APB）
SPI[m]_SCLK	入力	シリアルリファレンスクロック注 1
割り込み		
SPI[m]_Int	出力	レベル検出割り込み出力、アクティブ High
外部信号（SPI マスタモード）		
SPI[m]_CLK (M)	出力	シリアルクロック
SPI[m]_MOSI (M)	出力	マスタ送信データ
SPI[m]_MISO (M)	入力	マスタ受信データ
SPI[m]_SS_N[3:0] (M)	出力	スレーブセレクト（チップセレクト）、アクティブ Low
外部信号（SPI スレーブモード）		
SPI[m]_CLK (S)	入力	シリアルクロック
SPI[m]_MOSI (S)	入力	スレーブ受信データ
SPI[m]_MISO (S)	出力	スレーブ送信データ
SPI[m]_SS_N (S)	入力	スレーブセレクト（チップセレクト）、アクティブ Low

備考　m＝1～6

本章ではインデックスを省いたスタイルを使用します。

例）SPI_PCLK

注1.　入力周波数　7.81MHz～125MHz

SPI[m]_SCLK（m＝1～4）と SPI[m]_SCLK（m＝5、6）は異なる周波数設定が可能です。

2.3　レジスタマップ

2.3.1　レジスタマップ SPI1（マスタ）

表 2.1　レジスタマップ SPI1（マスタ）

アドレス	レジスタシンボル	レジスタ名
5000 5000h	rSpi_CTRLR0	コントロールレジスタ 0
5000 5004h	rSpi_CTRLR1	コントロールレジスタ 1
5000 5008h	rSpi_SSIENR	イネーブルレジスタ
5000 500Ch	rSpi_MWCR	Microwire コントロールレジスタ
5000 5010h	rSpi_SER	スレーブイネーブルレジスタ
5000 5014h	rSpi_BAUDR	ボーレート選択
5000 5018h	rSpi_TXFTLR	送信 FIFO しきい値レベル
5000 501Ch	rSpi_RXFTLR	受信 FIFO しきい値レベル
5000 5020h	rSpi_TXFLR	送信 FIFO レベルレジスタ
5000 5024h	rSpi_RXFLR	受信 FIFO レベルレジスタ
5000 5028h	rSpi_SR	ステータスレジスタ
5000 502Ch	rSpi_IMR	割り込みマスクレジスタ
5000 5030h	rSpi_ISR	割り込みステータスレジスタ
5000 5034h	rSpi_RISR	ロウ（raw）割り込みステータスレジスタ
5000 5038h	rSpi_TXOICR	送信 FIFO オーバーフロー割り込みクリアレジスタ
5000 503Ch	rSpi_RXOICR	受信 FIFO オーバーフロー割り込みクリアレジスタ
5000 5040h	rSpi_RXUICR	受信 FIFO アンダーフロー割り込みクリアレジスタ
5000 5048h	rSpi_ICR	割り込みクリアレジスタ
5000 504Ch	rSpi_DMACR	DMA コントロールレジスタ
5000 5050h	rSpi_DMATDLR	DMA 送信データレベル
5000 5054h	rSpi_DMARDLR	DMA 受信データレベル
5000 5060h	rSpi_DR	データレジスタ
5000 50F0h	rSpi_RX_SAMPLE_DLY	RXD サンプル遅延レジスタ
5000 5100h	rSpi_TDMACR	DMA コントロールレジスタ（送信モード）
5000 5104h	rSpi_RDMACR	DMA コントロールレジスタ（受信モード）

2.3.2　レジスタマップ SPI2（マスタ）

表 2.2　レジスタマップ SPI2（マスタ）

アドレス	レジスタシンボル	レジスタ名
5000 6000h	rSpi_CTRLR0	コントロールレジスタ 0
5000 6004h	rSpi_CTRLR1	コントロールレジスタ 1
5000 6008h	rSpi_SSIENR	イネーブルレジスタ
5000 600Ch	rSpi_MWCR	Microwire コントロールレジスタ
5000 6010h	rSpi_SER	スレーブイネーブルレジスタ
5000 6014h	rSpi_BAUDR	ボーレート選択
5000 6018h	rSpi_TXFTLR	送信 FIFO しきい値レベル
5000 601Ch	rSpi_RXFTLR	受信 FIFO しきい値レベル
5000 6020h	rSpi_TXFLR	送信 FIFO レベルレジスタ
5000 6024h	rSpi_RXFLR	受信 FIFO レベルレジスタ
5000 6028h	rSpi_SR	ステータスレジスタ
5000 602Ch	rSpi_IMR	割り込みマスクレジスタ
5000 6030h	rSpi_ISR	割り込みステータスレジスタ
5000 6034h	rSpi_RISR	ロウ（raw）割り込みステータスレジスタ
5000 6038h	rSpi_TXOICR	送信 FIFO オーバーフロー割り込みクリアレジスタ
5000 603Ch	rSpi_RXOICR	受信 FIFO オーバーフロー割り込みクリアレジスタ
5000 6040h	rSpi_RXUICR	受信 FIFO アンダーフロー割り込みクリアレジスタ
5000 6048h	rSpi_ICR	割り込みクリアレジスタ
5000 604Ch	rSpi_DMACR	DMA コントロールレジスタ
5000 6050h	rSpi_DMATDLR	DMA 送信データレベル
5000 6054h	rSpi_DMARDLR	DMA 受信データレベル
5000 6060h	rSpi_DR	データレジスタ
5000 60F0h	rSpi_RX_SAMPLE_DLY	RXD サンプル遅延レジスタ
5000 6100h	rSpi_TDMACR	DMA コントロールレジスタ（送信モード）
5000 6104h	rSpi_RDMACR	DMA コントロールレジスタ（受信モード）

2.3.3　レジスタマップ SPI3（マスタ）

表 2.3　レジスタマップ SPI3（マスタ）

アドレス	レジスタシンボル	レジスタ名
5000 7000h	rSpi_CTRLR0	コントロールレジスタ 0
5000 7004h	rSpi_CTRLR1	コントロールレジスタ 1
5000 7008h	rSpi_SSIENR	イネーブルレジスタ
5000 700Ch	rSpi_MWCR	Microwire コントロールレジスタ
5000 7010h	rSpi_SER	スレーブイネーブルレジスタ
5000 7014h	rSpi_BAUDR	ボーレート選択
5000 7018h	rSpi_TXFTLR	送信 FIFO しきい値レベル
5000 701Ch	rSpi_RXFTLR	受信 FIFO しきい値レベル
5000 7020h	rSpi_TXFLR	送信 FIFO レベルレジスタ
5000 7024h	rSpi_RXFLR	受信 FIFO レベルレジスタ
5000 7028h	rSpi_SR	ステータスレジスタ
5000 702Ch	rSpi_IMR	割り込みマスクレジスタ
5000 7030h	rSpi_ISR	割り込みステータスレジスタ
5000 7034h	rSpi_RISR	ロウ（raw）割り込みステータスレジスタ
5000 7038h	rSpi_TXOICR	送信 FIFO オーバーフロー割り込みクリアレジスタ
5000 703Ch	rSpi_RXOICR	受信 FIFO オーバーフロー割り込みクリアレジスタ
5000 7040h	rSpi_RXUICR	受信 FIFO アンダーフロー割り込みクリアレジスタ
5000 7048h	rSpi_ICR	割り込みクリアレジスタ
5000 704Ch	rSpi_DMACR	DMA コントロールレジスタ
5000 7050h	rSpi_DMATDLR	DMA 送信データレベル
5000 7054h	rSpi_DMARDLR	DMA 受信データレベル
5000 7060h	rSpi_DR	データレジスタ
5000 70F0h	rSpi_RX_SAMPLE_DLY	RXD サンプル遅延レジスタ
5000 7100h	rSpi_TDMACR	DMA コントロールレジスタ（送信モード）
5000 7104h	rSpi_RDMACR	DMA コントロールレジスタ（受信モード）

2.3.4　レジスタマップ SPI4（マスタ）

表 2.4　レジスタマップ SPI4（マスタ）

アドレス	レジスタシンボル	レジスタ名
5000 8000h	rSpi_CTRLR0	コントロールレジスタ 0
5000 8004h	rSpi_CTRLR1	コントロールレジスタ 1
5000 8008h	rSpi_SSIENR	イネーブルレジスタ
5000 800Ch	rSpi_MWCR	Microwire コントロールレジスタ
5000 8010h	rSpi_SER	スレーブイネーブルレジスタ
5000 8014h	rSpi_BAUDR	ボーレート選択
5000 8018h	rSpi_TXFTLR	送信 FIFO しきい値レベル
5000 801Ch	rSpi_RXFTLR	受信 FIFO しきい値レベル
5000 8020h	rSpi_TXFLR	送信 FIFO レベルレジスタ
5000 8024h	rSpi_RXFLR	受信 FIFO レベルレジスタ
5000 8028h	rSpi_SR	ステータスレジスタ
5000 802Ch	rSpi_IMR	割り込みマスクレジスタ
5000 8030h	rSpi_ISR	割り込みステータスレジスタ
5000 8034h	rSpi_RISR	ロウ（raw）割り込みステータスレジスタ
5000 8038h	rSpi_TXOICR	送信 FIFO オーバーフロー割り込みクリアレジスタ
5000 803Ch	rSpi_RXOICR	受信 FIFO オーバーフロー割り込みクリアレジスタ
5000 8040h	rSpi_RXUICR	受信 FIFO アンダーフロー割り込みクリアレジスタ
5000 8048h	rSpi_ICR	割り込みクリアレジスタ
5000 804Ch	rSpi_DMACR	DMA コントロールレジスタ
5000 8050h	rSpi_DMATDLR	DMA 送信データレベル
5000 8054h	rSpi_DMARDLR	DMA 受信データレベル
5000 8060h	rSpi_DR	データレジスタ
5000 80F0h	rSpi_RX_SAMPLE_DLY	RXD サンプル遅延レジスタ
5000 8100h	rSpi_TDMACR	DMA コントロールレジスタ（送信モード）
5000 8104h	rSpi_RDMACR	DMA コントロールレジスタ（受信モード）

2.3.5　レジスタマップ SPI5（スレーブ）

表 2.5　レジスタマップ SPI5（スレーブ）

アドレス	レジスタシンボル	レジスタ名
5000 9000h	rSpi_CTRLR0	コントロールレジスタ 0
5000 9008h	rSpi_SSIENR	イネーブルレジスタ
5000 900Ch	rSpi_MWCR	Microwire コントロールレジスタ
5000 9018h	rSpi_TXFTLR	送信 FIFO しきい値レベル
5000 901Ch	rSpi_RXFTLR	受信 FIFO しきい値レベル
5000 9020h	rSpi_TXFLR	送信 FIFO レベルレジスタ
5000 9024h	rSpi_RXFLR	受信 FIFO レベルレジスタ
5000 9028h	rSpi_SR	ステータスレジスタ
5000 902Ch	rSpi_IMR	割り込みマスクレジスタ
5000 9030h	rSpi_ISR	割り込みステータスレジスタ
5000 9034h	rSpi_RISR	ロウ（raw）割り込みステータスレジスタ
5000 9038h	rSpi_TXOICR	送信 FIFO オーバーフロー割り込みクリアレジスタ
5000 903Ch	rSpi_RXOICR	受信 FIFO オーバーフロー割り込みクリアレジスタ
5000 9040h	rSpi_RXUICR	受信 FIFO アンダーフロー割り込みクリアレジスタ
5000 9048h	rSpi_ICR	割り込みクリアレジスタ
5000 904Ch	rSpi_DMACR	DMA コントロールレジスタ
5000 9050h	rSpi_DMATDLR	DMA 送信データレベル
5000 9054h	rSpi_DMARDLR	DMA 受信データレベル
5000 9060h	rSpi_DR	データレジスタ
5000 9100h	rSpi_TDMACR	DMA コントロールレジスタ（送信モード）
5000 9104h	rSpi_RDMACR	DMA コントロールレジスタ（受信モード）

2.3.6　レジスタマップ SPI6（スレーブ）

表 2.6　レジスタマップ SPI6（スレーブ）

アドレス	レジスタシンボル	レジスタ名
5000 A000h	rSpi_CTRLR0	コントロールレジスタ 0
5000 A008h	rSpi_SSIENR	イネーブルレジスタ
5000 A00Ch	rSpi_MWCR	Microwire コントロールレジスタ
5000 A018h	rSpi_TXFTLR	送信 FIFO しきい値レベル
5000 A01Ch	rSpi_RXFTLR	受信 FIFO しきい値レベル
5000 A020h	rSpi_TXFLR	送信 FIFO レベルレジスタ
5000 A024h	rSpi_RXFLR	受信 FIFO レベルレジスタ
5000 A028h	rSpi_SR	ステータスレジスタ
5000 A02Ch	rSpi_IMR	割り込みマスクレジスタ
5000 A030h	rSpi_ISR	割り込みステータスレジスタ
5000 A034h	rSpi_RISR	ロウ（raw）割り込みステータスレジスタ
5000 A038h	rSpi_TXOICR	送信 FIFO オーバーフロー割り込みクリアレジスタ
5000 A03Ch	rSpi_RXOICR	受信 FIFO オーバーフロー割り込みクリアレジスタ
5000 A040h	rSpi_RXUICR	受信 FIFO アンダーフロー割り込みクリアレジスタ
5000 A048h	rSpi_ICR	割り込みクリアレジスタ
5000 A04Ch	rSpi_DMACR	DMA コントロールレジスタ
5000 A050h	rSpi_DMATDLR	DMA 送信データレベル
5000 A054h	rSpi_DMARDLR	DMA 受信データレベル
5000 A060h	rSpi_DR	データレジスタ
5000 A100h	rSpi_TDMACR	DMA コントロールレジスタ（送信モード）
5000 A104h	rSpi_RDMACR	DMA コントロールレジスタ（受信モード）

2.4　レジスタの説明

2.4.1　rSpi_CTRLR0 ― コントロールレジスタ 0

本レジスタはシリアルデータ転送を制御します。bSpi_SSIENR がセットされ SPI コントローラが有効な場合は、本レジスタに書き込むことはできません。

アドレス　5000 5000h（SPI1）
5000 6000h（SPI2）
5000 7000h（SPI3）
5000 8000h（SPI4）
5000 9000h（SPI5）
5000 A000h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bSpi_CFS				bSpi_SRL	bSpi_SLV_OE	bSpi_TMOD		bSpi_SCPOL	bSpi_SCPH	bSpi_FRF		bSpi_DFS			
リセット後の値	0	0	0	0	0	0	0	0	1	1	0	0	0	1	1	1

表 2.7　rSpi_CTRLR0 レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	読むと 0 が読み出されます。	R
b15～b12	bSpi_CFS	コントロールフレームサイズ Microwire フレームフォーマットのコントロールワードの長さを選択します。 4’b0000：1 ビットのコントロールワード 4’b0001：2 ビットのコントロールワード … … … … 4’b1110：15 ビットのコントロールワード 4’b1111：16 ビットのコントロールワード	R/W
b11	bSpi_SRL	シフトレジスタループ テスト用途のみに使用。セットした場合、送信シフトレジスタ出力を受信シフトレジスタ入力に接続します。SPI スレーブおよび SPI マスタの両方で使用できます。 0：通常モードで動作 1：テストモードで動作 SPI スレーブがテストモードとして設定された場合、SPI_CLK 信号および SPI_SS_N 信号は、外部ソースによって提供される必要があります。本モードでは、スレーブはループバックするものがないため、これらの信号を生成できません。	R/W
b10	bSpi_SLV_OE	スレーブ出力イネーブル（SPI スレーブのみ） 本ビットは SPI スレーブからの SPI_MISO 出力の設定を有効または無効にします。 bSpi_SLV_OE＝1 の場合、SPI_MISO 出力はアクティブにできず、常にフローティング状態になります。 マスタがブロードキャストモードで送信する（マスタはすべてのスレーブデバイスにデータを送信する）場合に役立ちます。1 つのスレーブのみが、マスタの SPI_MISO ライン上のデータに対応することが可能です。 本ビットはリセット後に有効になりますが、本デバイスでデータの応答を行いたくない場合は、ソフトウェアによって無効にする必要があります（ブロードキャストモード使用時）。 1’b0：スレーブ SPI_MISO は有効 1’b1：スレーブ SPI_MISO は無効	R/W

表 2.7　rSpi_CTRLR0 レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b9、b8	bSpi_TMOD	転送モード シリアル通信用の転送モードを選択します。本フィールドはデュプレックス設定には影響を与えません。受信データまたは送信データのどちらが有効かを示すのみです。 送信のみモードでは、外部デバイスから受信したデータは有効ではなく、受信 FIFO メモリにも保存されません。そのため、次の転送時に上書きされます。 受信のみモードでは、送信データは有効ではありません。送信 FIFO に最初に書き込まれた後、同一のコントロールワードが転送期間中再送信されます。 送受信モードでは、送信データと受信データの両方が有効です。送信 FIFO がエンプティになるまで転送は続行します。外部デバイスから受信したデータは、受信 FIFO メモリに格納され、ホストプロセッサによってアクセスできます。 EEPROM 読み出しモードでは、制御データが送信されている間、受信データは有効ではありません。すべての制御データが EEPROM に送信されたときに、受信データが有効になり、送信データは無効になります。送信 FIFO のすべてのデータは、本モードで制御データとみなされます。本転送モードは SP I コントローラがマスタの場合のみ有効です。 2’b00：送受信 2’b01：送信のみ 2’b10：受信のみ 2’b11：EEPROM 読み出し	R/W
b7	bSpi_SCPOL	シリアルクロック極性 フレームフォーマット（bSpi_FRF）がモトローラ SPI に設定されている場合に有効です。 非アクティブなシリアルクロックの極性を選択するために使用され、SPI コントローラがシリアルバスでデータ転送をアクティブに行っていないときに、非アクティブに保たれます。 1’b0：シリアルクロックの非アクティブ状態は Low です 1’b1：シリアルクロックの非アクティブ状態は High です	R/W
b6	bSpi_SCPH	シリアルクロック位相 フレームフォーマット（bSpi_FRF）がモトローラ SPI に設定されている場合に有効です。シリアルクロック位相は、スレーブセレクト信号とシリアルクロックの関係を選択します。 bSpi_SCPH＝0 の場合、データはシリアルクロックの最初のエッジでキャプチャされます。 bSpi_SCPH＝1 の場合、スレーブセレクトラインがアクティブ Low になった後、シリアルクロックは 1 つ目にトグルされ、データはシリアルクロックの 2 番目のエッジでキャプチャされます。 1’b0：シリアルクロックはデータビットの真ん中で最初にトグルします。 1’b1：シリアルクロックはデータビットのスタート時点で最初にトグルします。	R/W
b5、b4	bSpi_FRF	フレームフォーマット 2’b00：モトローラシリアルペリフェラルインタフェース 2’b01：テキサスインスツルメンツ同期式シリアルプロトコル 2’b10：ナショナルセミコンダクターMicrowire 2’b11：予約	R/W
b3～b0	bSpi_DFS	データフレームサイズ データフレーム長を選択します。データフレームサイズが 16 ビット未満になるようにプログラミングされている場合、受信データは受信ロジックによって自動的に右詰めされ、受信 FIFO の上位側に 0 がパディングされます。 送信 FIFO に書き込む前に送信データを右詰めする必要があります。送信ロジックはデータ送信時に未使用の上位ビットを無視します。 4’b0000：予約 - 未定義の動作 4’b0001：予約 - 未定義の動作 4’b0010：予約 - 未定義の動作 4’b0011：4 ビットのシリアルデータ転送 4’b0100：5 ビットのシリアルデータ転送 … … … … 4’b1110：15 ビットのシリアルデータ転送 4’b1111：16 ビットのシリアルデータ転送	R/W

2.4.2　rSpi_CTRLR1 ― コントロールレジスタ 1

bSpi_SSIENR がセットされ SPI コントローラが有効な場合は、本レジスタに書き込むことはできません。

注　意

本レジスタは SPI マスタのみに存在します。連続受信は、フレームサイズによらず最大 64KB です。

アドレス　5000 5004h（SPI1）
5000 6004h（SPI2）
5000 7004h（SPI3）
5000 8004h（SPI4）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bSpi_NDF															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.8　rSpi_CTRLR1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	読むと 0 が読み出されます。	R
b15～b0	bSpi_NDF	データフレーム数 bSpi_TMOD＝2’b10 または bSpi_TMOD＝2’b11 の場合、本レジスタフィールドは、SPI コントローラにより連続して受信されるデータフレームの数を設定します。 SPI コントローラは、データフレームの数が本レジスタ値に 1 を足した値に等しくなるまで連続してシリアルデータを受信し、連続転送で最大 64KB まで受信できます。	R/W

2.4.3　rSpi_SSIENR ― イネーブルレジスタ

アドレス　5000 5008h（SPI1）
5000 6008h（SPI2）
5000 7008h（SPI3）
5000 8008h（SPI4）
5000 9008h（SPI5）
5000 A008h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_SSIENR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.9　rSpi_SSIENR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	読むと 0 が読み出されます。	R
b0	bSpi_SSIENR	SPI コントローラ有効 すべての SPI 動作を有効または無効にします。 無効な場合、すべてのシリアル転送はただちに停止されます。送信 FIFO バッファおよび受信 FIFO バッファはクリアされます。有効な場合は、SPI コントロールレジスタの一部をプログラムできません。 1'b0：無効 1'b1：有効	R/W

2.4.4　rSpi_MWCR ― Microwire コントロールレジスタ

bSpi_SSIENR がセットされ SPI コントローラが有効な場合は、本レジスタに書き込むことはできません。

アドレス　5000 500Ch（SPI1）
5000 600Ch（SPI2）
5000 700Ch（SPI3）
5000 800Ch（SPI4）
5000 900Ch（SPI5）
5000 A00Ch（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_MWHS	bSpi_MDD	bSpi_MWMOD
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.10　rSpi_MWCR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b3	予約ビット	読むと 0 が読み出されます。	R
b2	bSpi_MWHS	Microwire ハンドシェイク（SPI マスタのみ） Microwire プロトコル用の“ビジー／レディ”ハンドシェイクを有効および無効にするために使用されます。 有効の場合、最終データ／制御ビットの転送後、レジスタ rSpi_SR のビジー状態をクリアする前に、SPI コントローラはターゲットスレーブからレディ状態をチェックします。 0：ハンドシェイクインタフェースは無効 1：ハンドシェイクインタフェースは有効	R/W
b1	bSpi_MDD	Microwire コントロール Microwire シリアルプロトコルの使用時にデータワードの方向を定義します。 0：データワードは外部シリアルデバイスから SPI コントローラによって受信される 1：データワードは外部シリアルデバイスに SPI コントローラから送信される	R/W
b0	bSpi_MWMOD	Microwire 転送モード Microwire 転送がシーケンシャルか非シーケンシャルかを定義します。シーケンシャルモードが使用される場合、データワードの 1 ブロックを送信または受信するには、1 つのコントロールワードのみ必要です。 非シーケンシャルモードが使用される場合、送信または受信されるデータワードそれぞれに対して 1 つのコントロールワードが存在する必要があります。 0：非シーケンシャル転送 1：シーケンシャル転送	R/W

2.4.5　rSpi_SER ― スレーブイネーブルレジスタ

SPI コントローラがビジー状態の場合は、本レジスタに書き込まないでください。「**2.5.3 ハードウェアモードあるいはソフトウェアモードによるスレーブセレクトラインの制御**」を参照してください。

注　意

本レジスタは SPI マスタのみに存在します。

アドレス　5000 5010h（SPI1）
5000 6010h（SPI2）
5000 7010h（SPI3）
5000 8010h（SPI4）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bSpi_CtrlSS				bSpi_SoftwareSS				bSpi_HardwareSS			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.11　rSpi_SER レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b12	予約ビット	読むと 0 が読み出されます。	R
b11～b8	bSpi_CtrlSS	スレーブセレクトモード有効 SPI マスタからの個々のスレーブセレクト出力ラインを有効にします。最大 4 つのスレーブセレクト出力信号を SPI マスタで使用可能です。 各 bSpi_CtrlSS[3:0]ビットでは、ハードウェアモードまたはソフトウェアモードで SPI マスタからのスレーブのそれぞれのセレクトライン SPI_SS_N[3:0]のコントロールを許可します。 0：ハードウェアモード • これらのライン SPI_SS_N[3:0]は、シリアライザモジュールからそれぞれ bSpi_HardwareSS[3:0]ビットによってコントロールされ、シリアル転送が開始するときにアクティブ Low になります。 1：ソフトウェアモード • これらのライン SPI_SS_N[3:0]はそれぞれ bSpi_SoftwareSS[3:0]ビットによって直接コントロールされます。	R/W
b7～b4	bSpi_SoftwareSS	ソフトウェアモード：スレーブセレクト有効フラグ SPI マスタからの個々のスレーブセレクト出力ラインを有効にします。最大 4 つのスレーブセレクト出力信号を SPI マスタで使用可能です。 本レジスタの各 bSpi_SoftwareSS[3:0]ビットは、SPI マスタから各スレーブへのセレクトライン SPI_SS_N[3:0]に対応します。 bSpi_CtrlSS[3:0]がラインそれぞれに対して 1 にセットされたとき、本モードはアクティブになります。マスタからの対応するスレーブセレクト SPI_SS_N[3:0]ラインは、周辺デバイスに直接接続され、bSpi_SoftwareSS[3:0]によってそれぞれコントロールされます。 **注意）** 転送を開始させるために、bSpi_HardwareSS[3:0]で 1 つのスレーブを選択してください。 1：選択済み 0：選択されていない	R/W

表 2.11　rSpi_SER レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3～b0	bSpi_HardwareSS	ハードウェアモード：スレーブセレクト有効フラグ レジスタは、SPI マスタからのスレーブセレクト出力ラインをそれぞれ有効にします。最大 4 つのスレーブセレクト出力信号が SPI マスタで使用可能です。 本レジスタの各 bSpi_HardwareSS[3:0]ビットは、SPI マスタから各スレーブへのセレクトライン SPI_SS_N[3:0]に対応します。 bSpi_CtrlSS[3:0]がラインそれぞれに対して 0 にクリアされたとき、本モードはアクティブになります。本モードでは、本レジスタのビットが 1 にセットされた場合、対応するスレーブセレクト SPI_SS_N[3:0]は、シリアル転送が開始されたときにそれぞれアクティブ Low になります。これらのラインはシリアライザモジュールによってコントロールされます。 本レジスタのビットのセットまたはクリアは、転送が開始するまでは、対応するスレーブセレクト出力に影響を与えないことに注意する必要があります。転送を開始する前に、マスタが通信するスレーブデバイスに対応する本レジスタのビットを有効にする必要があります。ブロードキャストモード（マスタはすべてのスレーブデバイスにデータを送信する）で動作しない場合は、本フィールドの 1 つのビットのみがセットされる必要があります。 1：選択済み 0：選択されていない	R/W

2.4.6　rSpi_BAUDR — ボーレート選択

bSpi_SSIENR がセットされ SPI コントローラが有効な場合は、本レジスタに書き込むことはできません。

注　意

本レジスタは SPI マスタのみに存在します。

アドレス　5000 5014h（SPI1）
5000 6014h（SPI2）
5000 7014h（SPI3）
5000 8014h（SPI4）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bSpi_SCKDV															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.12　rSpi_BAUDR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	読むと 0 が読み出されます。	R
b15～b0	bSpi_SCKDV	SPI クロック分周器 本レジスタの 16 ビットフィールドは、SPI_CLK 分周値を定義します。 本フィールドの LSB は、常に 0 にセットされており、書き込み動作によって影響を受けません。これにより、本レジスタで偶数値が保持されます。値が 0 の場合、シリアル出力クロック（SPI_CLK）は無効です。 SPI_CLK の周波数は以下の式から導かれます。 周波数（SPI_CLK）＝周波数（SPI_SCLK）／bSpi_SCKDV ここで、bSpi_SCKDV は 2～65534 の間の偶数です。	R/W

2.4.7　rSpi_TXFTLR ― 送信 FIFO しきい値レベル

本レジスタは送信 FIFO メモリのしきい値をコントロールします。

bSpi_SSIENR がセットされ SPI コントローラが有効な場合は、本レジスタに書き込まないでください。

アドレス　5000 5018h（SPI1）
5000 6018h（SPI2）
5000 7018h（SPI3）
5000 8018h（SPI4）
5000 9018h（SPI5）
5000 A018h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_TFT			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.13　rSpi_TXFTLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b4	予約ビット	読むと 0 が読み出されます。	R
b3～b0	bSpi_TFT	送信 FIFO しきい値 送信 FIFO コントローラが割り込みをトリガするエントリのレベル（上限）をコントロールします。 本値を FIFO の深さ以上の値にセットしようとすると、本フィールドは書き込まれず、現在の値を保持します。 送信 FIFO エントリ数が本値以下の場合、送信 FIFO エンプティ割り込みがトリガされます。 4’d0：送信 FIFO データエントリ数が 0 の場合、iSpi_TXE_Int はアサートされます 4’d1：送信 FIFO データエントリ数が 1 以下の場合、iSpi_TXE_Int はアサートされます … … … … … … … … 4’d14：送信 FIFO データエントリ数が 14 以下の場合、iSpi_TXE_Int はアサートされます 4’d15：送信 FIFO データエントリ数が 15 以下の場合、iSpi_TXE_Int はアサートされます	R/W

2.4.8　rSpi_RXFTLR ― 受信 FIFO しきい値レベル

本レジスタは受信 FIFO メモリのしきい値をコントロールします。
bSpi_SSIENR がセットされ SPI コントローラが有効な場合は、本レジスタに書き込まないでください。

アドレス　5000 501Ch（SPI1）
5000 601Ch（SPI2）
5000 701Ch（SPI3）
5000 801Ch（SPI4）
5000 901Ch（SPI5）
5000 A01Ch（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_RFT			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.14　rSpi_RXFTLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b4	予約ビット	読むと 0 が読み出されます。	R
b3～b0	bSpi_RFT	受信 FIFO しきい値 受信 FIFO コントローラが割り込みをトリガするエントリのレベル（下限）をコントロールします。 本値を FIFO の深さを超えた値にセットしようとすると、本フィールドは書き込まれず、現在の値を保持します。 受信 FIFO エントリの数が本値プラス 1 以上の場合、受信 FIFO フル割り込みがトリガされます。 4’d0：受信 FIFO に 1 以上のデータエントリが存在する場合、iSpi_RXF_Int はアサートされます 4’d1：受信 FIFO に 2 以上のデータエントリが存在する場合、iSpi_RXF_Int はアサートされます … … … … … … … … 4’d14：受信 FIFO に 15 以上のデータエントリが存在する場合、iSpi_RXF_Int はアサートされます 4’d15：受信 FIFO に 16 以上のデータエントリが存在する場合、iSpi_RXF_Int はアサートされます	R/W

2.4.9　rSpi_TXFLR ― 送信 FIFO レベルレジスタ

本レジスタは、送信 FIFO メモリの有効なデータエントリ数を示します。

アドレス　5000 5020h（SPI1）
5000 6020h（SPI2）
5000 7020h（SPI3）
5000 8020h（SPI4）
5000 9020h（SPI5）
5000 A020h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bSpi_TXTFL				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.15　rSpi_TXFLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	読むと 0 が読み出されます。	R
b4～b0	bSpi_TXTFL	送信 FIFO レベル 送信 FIFO の有効なデータエントリ数	R

2.4.10　rSpi_RXFLR ― 受信 FIFO レベルレジスタ

本レジスタは、受信 FIFO メモリの有効なデータエントリ数を示します。本レジスタは、いつでも読み出しが可能です。

アドレス　5000 5024h（SPI1）
5000 6024h（SPI2）
5000 7024h（SPI3）
5000 8024h（SPI4）
5000 9024h（SPI5）
5000 A024h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bSpi_RXTFL				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.16　rSpi_RXFLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	読むと 0 が読み出されます。	R
b4～b0	bSpi_RXTFL	受信 FIFO レベル 受信 FIFO の有効なデータエントリ数	R

2.4.11　rSpi_SR ― ステータスレジスタ

ステータスレジスタは、いつでも読み出しが可能です。

アドレス　5000 5028h（SPI1）
5000 6028h（SPI2）
5000 7028h（SPI3）
5000 8028h（SPI4）
5000 9028h（SPI5）
5000 A028h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	bSpi_TXE	bSpi_RFF	bSpi_RFNE	bSpi_TFE	bSpi_TFNF	bSpi_BUSY
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

表 2.17　rSpi_SR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b6	予約ビット	読むと 0 が読み出されます。	R
b5	bSpi_TXE	送信エラー（SPI スレーブのみ） 転送が開始されるときに送信 FIFO がエンプティの場合にセットされます。 以前の送信データが、SPI_MISO ラインで再送信されます。 本ビットは読み出し時にクリアされます。 0：エラーなし 1：送信エラー	R
b4	bSpi_RFF	受信 FIFO フル 受信 FIFO が完全にフルの場合、本ビットがセットされます。 受信 FIFO に 1 つ以上のエンプティが含まれる場合、本ビットはクリアされます。 0：受信 FIFO は空き 1：受信 FIFO はフル	R
b3	bSpi_RFNE	受信 FIFO は非エンプティ 受信 FIFO が 1 つ以上のエントリを含むときにセットされ、受信 FIFO がエンプティのときにクリアされます。 受信 FIFO を完全にエンプティにする際に、本ビットはソフトウェアによってポーリングが可能です。 0：受信 FIFO はエンプティ 1：受信 FIFO は非エンプティ	R
b2	bSpi_TFE	送信 FIFO エンプティ 送信 FIFO が完全にエンプティな場合、本ビットがセットされます。 送信 FIFO に 1 つ以上の有効なエントリが含まれる場合、本ビットはクリアされます。本ビットは割り込みを要求しません。 0：送信 FIFO は非エンプティ 1：送信 FIFO はエンプティ	R
b1	bSpi_TFNF	送信 FIFO 空き 送信 FIFO が 1 つ以上のエンプティを含む場合にセットされ、FIFO がフルの場合にクリアされます。 0：送信 FIFO はフル 1：送信 FIFO は空き	R

表 2.17　rSpi_SR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bSpi_BUSY	SPI ビジーフラグ セットされた場合、シリアル転送が進行中であることを示し、クリアされると、SPI コントローラがアイドルまたは無効であることを示します。 0：SPI コントローラはアイドルまたは無効 1：SPI コントローラはアクティブにデータを転送中	R

2.4.12　rSpi_IMR ― 割り込みマスクレジスタ

アドレス　5000 502Ch（SPI1）
5000 602Ch（SPI2）
5000 702Ch（SPI3）
5000 802Ch（SPI4）
5000 902Ch（SPI5）
5000 A02Ch（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bSpi_RXFIM	bSpi_RXOIM	bSpi_RXUIM	bSpi_TXOIM	bSpi_TXEIM
リセット後の値	0	0	0	0	0	0	0	0	0	0	1/0	1	1	1	1	1

表 2.18　rSpi_IMR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b6	予約ビット	読むと 0 が読み出されます。	R
b5	予約ビット	値は、SPI マスタには 1、SPI スレーブには 0 にリセットされます。初期値を保持してください。	R/W
b4	bSpi_RXFIM	受信 FIFO フル割り込みマスク 1’b0：iSpi_RXF_Int 割り込み禁止 1’b1：iSpi_RXF_Int 割り込み許可	R/W
b3	bSpi_RXOIM	受信 FIFO オーバーフロー割り込みマスク 1’b0：iSpi_RXO_Int 割り込み禁止 1’b1：iSpi_RXO_Int 割り込み許可	R/W
b2	bSpi_RXUIM	受信 FIFO アンダーフロー割り込みマスク 1’b0：iSpi_RXU_Int 割り込み禁止 1’b1：iSpi_RXU_Int 割り込み許可	R/W
b1	bSpi_TXOIM	送信 FIFO オーバーフロー割り込みマスク 1’b0：iSpi_TXO_Int 割り込み禁止 1’b1：iSpi_TXO_Int 割り込み許可	R/W
b0	bSpi_TXEIM	送信 FIFO エンプティ割り込みマスク 1’b0：iSpi_TXE_Int 割り込み禁止 1’b1：iSpi_TXE_Int 割り込み許可	R/W

2.4.13　rSpi_ISR ― 割り込みステータスレジスタ

本レジスタは、SPI 割り込みのマスク後のステータスをレポートします。

アドレス　5000 5030h（SPI1）
5000 6030h（SPI2）
5000 7030h（SPI3）
5000 8030h（SPI4）
5000 9030h（SPI5）
5000 A030h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bSpi_RXFIS	bSpi_RXOIS	bSpi_RXUIS	bSpi_TXOIS	bSpi_TXEIS
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.19　rSpi_ISR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	読むと 0 が読み出されます。	R
b4	bSpi_RXFIS	受信 FIFO フル割り込みステータス 1'b0：iSpi_RXF_Int 割り込みなし（マスク後） 1'b1：iSpi_RXF_Int 割り込みあり（マスク後）	R
b3	bSpi_RXOIS	受信 FIFO オーバーフロー割り込みステータス 1'b0：iSpi_RXO_Int 割り込みなし（マスク後） 1'b1：iSpi_RXO_Int 割り込みあり（マスク後）	R
b2	bSpi_RXUIS	受信 FIFO アンダーフロー割り込みステータス 1'b0：iSpi_RXU_Int 割り込みなし（マスク後） 1'b1：iSpi_RXU_Int 割り込みあり（マスク後）	R
b1	bSpi_TXOIS	送信 FIFO オーバーフロー割り込みステータス 1'b0：iSpi_TXO_Int 割り込みなし（マスク後） 1'b1：iSpi_TXO_Int 割り込みあり（マスク後）	R
b0	bSpi_TXEIS	送信 FIFO エンプティ割り込みステータス 1'b0：iSpi_TXE_Int 割り込みなし（マスク後） 1'b1：iSpi_TXE_Int 割り込みあり（マスク後）	R

2.4.14　rSpi_RISR ― ロウ（raw）割り込みステータスレジスタ

本レジスタは、SPI 割り込みのマスク前のステータスをレポートします。

アドレス　5000 5034h（SPI1）
5000 6034h（SPI2）
5000 7034h（SPI3）
5000 8034h（SPI4）
5000 9034h（SPI5）
5000 A034h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bSpi_RXFIR	bSpi_RXOIR	bSpi_RXUIR	bSpi_TXOIR	bSpi_TXEIR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.20　rSpi_RISR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	読むと 0 が読み出されます。	R
b4	bSpi_RXFIR	受信 FIFO フル割り込みステータス 1'b0：iSpi_RXF_Int 割り込みなし（マスク前） 1'b1：iSpi_RXF_Int 割り込みあり（マスク前）	R
b3	bSpi_RXOIR	受信 FIFO オーバーフロー割り込みステータス 1'b0：iSpi_RXO_Int 割り込みなし（マスク前） 1'b1：iSpi_RXO_Int 割り込みあり（マスク前）	R
b2	bSpi_RXUIR	受信 FIFO アンダーフロー割り込みステータス 1'b0：iSpi_RXU_Int 割り込みなし（マスク前） 1'b1：iSpi_RXU_Int 割り込みあり（マスク前）	R
b1	bSpi_TXOIR	送信 FIFO オーバーフロー割り込みステータス 1'b0：iSpi_TXO_Int 割り込みなし（マスク前） 1'b1：iSpi_TXO_Int 割り込みあり（マスク前）	R
b0	bSpi_TXEIR	送信 FIFO エンプティ割り込みステータス 1'b0：iSpi_TXE_Int 割り込みなし（マスク前） 1'b1：iSpi_TXE_Int 割り込みあり（マスク前）	R

2.4.15　rSpi_TXOICR ― 送信 FIFO オーバーフロー割り込みクリアレジスタ

アドレス　5000 5038h（SPI1）
5000 6038h（SPI2）
5000 7038h（SPI3）
5000 8038h（SPI4）
5000 9038h（SPI5）
5000 A038h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_TXOICR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.21　rSpi_TXOICR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	読むと 0 が読み出されます。	R
b0	bSpi_TXOICR	送信 FIFO オーバーフロー割り込みをクリア 本レジスタは、割り込みのステータスを反映します。 本レジスタからの読み出しによって、iSpi_TXO_Int 割り込みがクリアされ、書き込みは無効です。	R

2.4.16　rSpi_RXOICR ― 受信 FIFO オーバーフロー割り込みクリアレジスタ

アドレス　5000 503Ch（SPI1）
5000 603Ch（SPI2）
5000 703Ch（SPI3）
5000 803Ch（SPI4）
5000 903Ch（SPI5）
5000 A03Ch（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_RXOICR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.22　rSpi_RXOICR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	読むと 0 が読み出されます。	R
b0	bSpi_RXOICR	受信 FIFO オーバーフロー割り込みをクリア 本レジスタは、割り込みのステータスを反映します。 本レジスタからの読み出しによって、iSpi_RXO_Int 割り込みがクリアされ、書き込みは無効です。	R

2.4.17　rSpi_RXUICR ― 受信 FIFO アンダーフロー割り込みクリアレジスタ

アドレス　5000 5040h（SPI1）
5000 6040h（SPI2）
5000 7040h（SPI3）
5000 8040h（SPI4）
5000 9040h（SPI5）
5000 A040h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_RXUICR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.23　rSpi_RXUICR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	読むと 0 が読み出されます。	R
b0	bSpi_RXUICR	受信 FIFO アンダーフロー割り込みをクリア 本レジスタは、割り込みのステータスを反映します。 本レジスタからの読み出しによって、iSpi_RXU_Int 割り込みがクリアされ、書き込みは無効です。	R

2.4.18　rSpi_ICR ― 割り込みクリアレジスタ

アドレス　5000 5048h（SPI1）
5000 6048h（SPI2）
5000 7048h（SPI3）
5000 8048h（SPI4）
5000 9048h（SPI5）
5000 A048h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_ICR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.24　rSpi_ICR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	読むと 0 が読み出されます。	R
b0	bSpi_ICR	割り込みをクリア 以下の任意の割り込みがアクティブの場合に本レジスタはセットされます。 読み出しは、割り込み iSpi_TXO_Int、iSpi_RXU_Int、および iSpi_RXO_Int 割り込みをクリアします。本レジスタへの書き込みは無効です。	R

2.4.19　rSpi_DMACR ― DMA コントロールレジスタ

アドレス　5000 504Ch（SPI1）
5000 604Ch（SPI2）
5000 704Ch（SPI3）
5000 804Ch（SPI4）
5000 904Ch（SPI5）
5000 A04Ch（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_TDMAE	bSpi_RDMAE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.25　rSpi_DMACR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bSpi_TDMAE	送信 DMA 許可 本ビットは送信 FIFO DMA チャネルを許可／禁止します。 0：送信 DMA 禁止 1：送信 DMA 許可	R/W
b0	bSpi_RDMAE	受信 DMA 許可 本ビットは受信 FIFO DMA チャネルを許可／禁止します。 0：受信 DMA 禁止 1：受信 DMA 許可	R/W

2.4.20　rSpi_DMATDLR ― DMA 送信データレベル

アドレス　5000 5050h（SPI1）
5000 6050h（SPI2）
5000 7050h（SPI3）
5000 8050h（SPI4）
5000 9050h（SPI5）
5000 A050h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_DMATDLR			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.26　rSpi_DMATDLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b4	予約ビット	読むと 0 が読み出されます。	R
b3～b0	bSpi_DMATDLR	送信データレベル 本フィールドは、送信ロジックが DMA リクエストを実行するレベルを制御します。 そのレベルはウォーターマークレベルと等しく、送信 FIFO の有効なデータエントリ数が本フィールド値以下で、bSpi_TDMAE＝1 の場合に、DMA リクエストが生成されます。 4’d0：送信 FIFO データエントリ数が 0 の場合、DMA リクエストはアサートされます 4’d1：送信 FIFO データエントリ数が 1 以下の場合、DMA リクエストはアサートされます … … … … … … … … 4’d14：送信 FIFO データエントリ数が 14 以下の場合、DMA リクエストはアサートされます 4’d15：送信 FIFO データエントリ数が 15 以下の場合、DMA リクエストはアサートされます	R/W

2.4.21　rSpi_DMARDLR ― DMA 受信データレベル

アドレス　5000 5054h（SPI1）
5000 6054h（SPI2）
5000 7054h（SPI3）
5000 8054h（SPI4）
5000 9054h（SPI5）
5000 A054h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	bSpi_DMARDLR			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.27　rSpi_DMARDLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b4	予約ビット	読むと 0 が読み出されます。	R
b3～b0	bSpi_DMARDLR	受信データレベル 本フィールドは、受信ロジックが DMA リクエストを実行するレベルを制御します。ウォーターマークレベル＝bSpi_DMARDLR＋1、すなわち、受信 FIFO の有効なデータエントリ数が本フィールド値＋1 以上で、bSpi_RDMAE＝1 の場合に、DMA リクエストが生成されます。 4’d0：受信 FIFO に 1 以上のデータエントリが存在する場合、DMA リクエストはアサートされます 4’d1：受信 FIFO に 2 以上のデータエントリが存在する場合、DMA リクエストはアサートされます … … … … … … … … 4’d14：受信 FIFO に 15 以上のデータエントリが存在する場合、DMA リクエストはアサートされます 4’d15：受信 FIFO に 16 以上のデータエントリが存在する場合、DMA リクエストはアサートされます	R/W

2.4.22　rSpi_DR ― データレジスタ

SPI データレジスタは送信／受信 FIFO 用の 16 ビットの読み出し／書き込みバッファです。レジスタが読み出されると、受信 FIFO バッファのデータがアクセスされます。書き込まれると、データは送信 FIFO バッファに移動します。bSpi_SSIENR＝1 の場合のみ書き込みが可能です。FIFO は bSpi_SSIENR＝0 の場合にリセットされます。

備　考

SPI コントローラの rSpi_DR レジスタは、AHB バースト転送を機能させるために、メモリマップ上で 32 ビット 36 個分を占有します。これらのアドレス位置のいずれかへの書き込みは、APB バスから送信 FIFO にデータを追加するのと同じ効果となります。これらの位置のいずれかからの読み出しは、受信 FIFO からデータが APB バスに取り出されるのと同じ効果となります。SPI コントローラ上の FIFO バッファはアドレス指定できません。
データレジスタ（DR）は、AHB マスタ（DMA コントローラやプロセッサなど）からアクセスされる場合、AHB 転送タイプはバーストである可能性があります。AHB バースト転送中、アドレスはバーストの各ビート後にインクリメントされます。

アドレス　5000 5060h（SPI1）
5000 6060h（SPI2）
5000 7060h（SPI3）
5000 8060h（SPI4）
5000 9060h（SPI5）
5000 A060h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bSpi_DR															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.28　rSpi_DR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	読むと 0 が読み出されます。	R
b15～b0	bSpi_DR	データレジスタ 本レジスタへの書き込み時、データを右詰めする必要があります。データの読み出しは、自動的に右詰めされます。 読み出し：受信 FIFO バッファ 書き込み：送信 FIFO バッファ **注意**　DMA モードでは、bSpi_TDMAE と bSpi_TDMAE1、あるいは bSpi_RDMAE と bSpi_RDMAE1 のいずれかまたは両方が 1 にセットされ、DMA コントローラには以下のパラメータでプログラムされる必要があります。 ● 1 つのアドレスのみが rSpi_DR レジスタ用に使用されます。（たとえば 12'h060） DMA は 16 ビットワードモードのみで設定されます。	R/W

2.4.23　rSpi_RX_SAMPLE_DLY ― RXD サンプル遅延レジスタ

注　意

本レジスタは SPI マスタのみに存在します。

アドレス　5000 50F0h（SPI1）
5000 60F0h（SPI2）
5000 70F0h（SPI3）
5000 80F0h（SPI4）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bSpi_RX_Sample_Delay							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.29　rSpi_RX_SAMPLE_DLY レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bSpi_RX_Sample_Delay	受信データ（SPI_MISO）サンプル遅延 本レジスタは、SPI_MISO 入力信号のサンプルを遅延させるために使用されます。値は、SPI_MISO 信号サンプルの SPI_SCLK 遅延を表します。 「**2.5.5 データ入力サンプル遅延**」を参照してください。 **備考）** 本レジスタは、内部シフトレジスタの段数（64）を超える値でプログラムされた場合、ゼロ（0）遅延が SPI_MISO のサンプルに適用されます。	R/W

2.4.24　rSpi_TDMACR ― DMA コントロールレジスタ（送信モード）

アドレス　5000 5100h（SPI1）
5000 6100h（SPI2）
5000 7100h（SPI3）
5000 8100h（SPI4）
5000 9100h（SPI5）
5000 A100h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	bSpi_CURRENT_DEST_BLOCK_SIZE												
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bSpi_DEST_BLOCK_SIZE													bSpi_DEST_BURST_SIZE		bSpi_TDMAE1
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.30　rSpi_TDMACR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31	予約ビット	読むと 0 が読み出されます。	R
b30、b29	予約ビット	初期値を保持してください。	R/W
b28～b16	bSpi_CURRENT_DEST_BLOCK_SIZE	DEST_BLOCK_SIZE の現在の残数 本フィールドはブロック転送が終了するたびにデクリメントされます。 ファームウェアが以下の動作を行うと、bSpi_DEST_BLOCK_SIZE 値が bSpi_CURRENT_DEST_BLOCK_SIZE にリロードされます。 bSpi_TDMAE1 ビットを“1”にセット（立ち上がり）	R
b15～b3	bSpi_DEST_BLOCK_SIZE	DEST_BLOCK_SIZE 送信 FIFO の宛先ブロック転送サイズ SPI コントローラがフローコントローラです。したがってユーザは、DMA モードが許可される前または許可されると同時に本フィールドに書き込みを行う必要があります。 DEST_BLOCK_SIZE に設定された数字は、各ブロック転送で実行するシングルトランザクションの総数を示します。 シングルトランザクションのサイズは 16 ビットワードです。 いったん転送が開始すると、bSpi_DEST_BLOCK_SIZE を読み出すことでブロック転送を完了させるために送信 FIFO へ書き込まれるデータバイトの総数が分かります。 13’d0：0 ワードの転送またはブロック転送終了 13’d1：1 ワードの転送 13’d2：2 ワードの転送 … … … … 13’d8191：8191 ワードの転送	R/W
b2、b1	bSpi_DEST_BURST_SIZE	DEST_BURST_SIZE 送信 FIFO の宛先バーストトランザクションサイズ SPI コントローラがフローコントローラです。したがってユーザは、DMA モードが許可される前または許可されると同時に本フィールドに書き込みを行う必要があります。 2’b00：1 ワード 2’b01：4 ワード 2’b10：8 ワード 2’b11：予約（未使用）	R/W

表 2.30　rSpi_TDMACR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bSpi_TDMAE1	送信 DMA 許可／禁止 0：送信モードで DMA を禁止 1：送信モードで DMA を許可 送信 FIFO の最後の転送完了後（送信 FIFO に DEST_BLOCK_SIZE ワードを書き込む）、送信モードでの DMA を禁止するために bSpi_TDMAE1 はハードウェアにより自動的にクリアされます。 したがって、ソフトウェアは本ビットのポーリングを行い、本チャネルがいつ空きになり次の DMA 転送ができるかを判別可能です。 **注意）** ● 本ビットを有効にする前に、ソフトウェアは、DMA モードを有効にするために rSpi_DMACR レジスタの bSpi_TDMAE を有効にする必要があります。その後、DMA モードは bSpi_TDMAE1 でのみコントロールされます。 ● DMA 転送中に本ビットがクリアされると、現在の転送（バーストまたはシングル）が終了してから DMA モードが停止します。DMA ブロック転送を完了するには、bSpi_DEST_BLOCK_SIZE に bSpi_CURRENT_DEST_BLOCK_SIZE を、bSpi_DEST_BURST_SIZE に適切な値を書き込みます。bSpi_CURRENT_DEST_BLOCK_SIZE 値は、現在の転送が終了しているときのみ一致します。	R/W

2.4.25　rSpi_RDMACR ― DMA コントロールレジスタ（受信モード）

アドレス　5000 5104h（SPI1）
5000 6104h（SPI2）
5000 7104h（SPI3）
5000 8104h（SPI4）
5000 9104h（SPI5）
5000 A104h（SPI6）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	bSpi_CURRENT_SRC_BLOCK_SIZE												
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bSpi_SRC_BLOCK_SIZE													bSpi_SRC_BURST_SIZE		bSpi_RDMAE1
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 2.31　rSpi_RDMACR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31	予約ビット	読むと 0 が読み出されます。	R
b30、b29	予約ビット	初期値を保持してください。	R/W
b28～b16	bSpi_CURRENT_SRC_BLOCK_SIZE	SRC_BLOCK_SIZE の現在の残数 本フィールドはブロック転送が終了するたびにデクリメントされます。 ファームウェアが以下の動作を行うと、bSpi_SRC_BLOCK_SIZE 値が bSpi_CURRENT_SRC_BLOCK_SIZE にリロードされます。 bSpi_RDMAE1 ビットを“1”にセット（立ち上がり）	R
b15～b3	bSpi_SRC_BLOCK_SIZE	SRC_BLOCK_SIZE 受信 FIFO の送信元ブロック転送サイズ SPI コントローラがフローコントローラです。 したがってユーザは、DMA モードが許可される前または許可されると同時に本フィールドに書き込みを行う必要があります。 SRC_BLOCK_SIZE に設定された数字は、各ブロック転送で実行するシングルトランザクションの総数を示します。 シングルトランザクションのサイズは 16 ビットワードです。 13’d0：0 ワードの転送またはブロック転送終了 13’d1：1 ワードの転送 13’d2：2 ワードの転送 … … … … 13’d8191：8191 ワードの転送	R/W
b2、b1	bSpi_SRC_BURST_SIZE	SRC_BURST_SIZE 受信 FIFO の送信元バーストトランザクションサイズ SPI コントローラがフローコントローラです。 したがってユーザは、DMA モードが許可される前または許可されると同時に本フィールドに書き込みを行う必要があります。 2’b00：1 ワード 2’b01：4 ワード 2’b10：8 ワード 2’b11：予約（未使用）	R/W

表 2.31　rSpi_RDMACR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bSpi_RDMAE1	受信 DMA 許可／禁止 0：受信モードで DMA を禁止 1：受信モードで DMA を許可 受信 FIFO の最後の転送完了後（受信 FIFO から SRC_BLOCK_SIZE ワードを読み出す）、受信モードでの DMA を禁止するために bSpi_RDMAE1 はハードウェアにより自動的にクリアされます。 したがって、ソフトウェアは本ビットのポーリングを行い、本チャネルがいつ空きになり次の DMA 転送ができるかを判別できます。 **注意）** ● 本ビットを有効にする前に、ソフトウェアは、DMA モードを有効にするために rSpi_DMACR レジスタの bSpi_RDMAE を有効にする必要があります。その後、DMA モードは bSpi_RDMAE1 でのみコントロールされます。 ● DMA 転送中に本ビットがクリアされると、現在の転送（バーストまたはシングル）が終了してから DMA モードが停止します。DMA ブロック転送を完了するには、bSpi_SRC_BLOCK_SIZE に bSpi_CURRENT_SRC_BLOCK_SIZE を、bSpi_SRC_BURST_SIZE に適切な値を書き込みます。bSpi_CURRENT_SRC_BLOCK_SIZE 値は、現在の転送が終了しているときのみ一致します。	R/W

2.5　動作説明

2.5.1　概要

SPI コントローラを接続しているシリアルマスタあるいはシリアルスレーブの周辺デバイスは以下のインタフェースの少なくとも 1 つを有する必要があります。

- モトローラシリアルペリフェラルインタフェース
  - モトローラの 4 線、全二重シリアルプロトコル。シリアルクロック位相およびシリアルクロック極性の 4 つの組み合わせが可能です。クロック位相（bSpi_SCPH）は、シリアル転送がスレーブセレクト信号の立ち下りエッジで開始するか、シリアルクロックの最初のエッジで開始するかを決定します。SPI コントローラがアイドルまたは無効の場合は、スレーブセレクトラインは、High に保持されます。詳細については、「**2.5.9 モトローラシリアルペリフェラルインタフェース**」を参照してください。
- テキサスインスツルメンツシリアルプロトコル（SSP）
  - 4 線、全二重シリアルプロトコル。SPI プロトコルおよび Microwire プロトコルに使用されるスレーブセレクトラインは、SSP プロトコルのフレームインジケータとしても機能します。詳細については、「**2.5.10 テキサスインスツルメンツ同期式シリアルプロトコル**」を参照してください。
- ナショナルセミコンダクターMicrowire
  - シリアルマスタからターゲットのシリアルスレーブへ送信されるコントロールワードを使用する、半二重のシリアルプロトコル。詳細については、「**2.5.11 ナショナルセミコンダクターMicrowire**」を参照してください。

使用されるプロトコルを選択するために、コントロールレジスタ 0（rSpi_CTRLR0）の bSpi_FRF（フレームフォーマット）をプログラムすることが可能です。SPI コントローラによってサポートされるシリアルプロトコルにより、ハードウェアかソフトウェアのいずれかを使用してシリアルスレーブは選択されます。

ハードウェアで実装された場合、シリアルスレーブは、専用のハードウェアセレクトラインの制御下で選択されます。シリアルマスタから生成されるセレクトラインの数は、バス上に存在するシリアルスレーブの数に等しい値となります。シリアルマスタデバイスは、データ転送が始まる前に、ターゲットのシリアルスレーブのセレクトラインをアサートします。アーキテクチャを「**2.5.2 SPI マスタと SPI スレーブ間の一般的な接続**」に示します。

ソフトウェアで実装された場合は、各シリアルスレーブの入力セレクトラインは、シリアルマスタ上の単一のスレーブセレクト出力信号のいずれかを発信元にする必要があります（スレーブセレクト出力を 1 つ持つようにマスタを設定する必要があります）。ソフトウェアドメインの主なプログラムはターゲットのスレーブデバイスの選択を制御することです。このアーキテクチャは、「**2.5.2 SPI マスタと SPI スレーブ間の一般的な接続**」で示されます。ソフトウェアは、どのスレーブが、マスタデバイスからのシリアル転送要求に応答するかを制御するために、すべてのスレーブで rSpi_SSIENR レジスタを使用することが可能です。

2.5.2　SPI マスタと SPI スレーブ間の一般的な接続

SPI コントローラは、シリアルマスタとシリアルスレーブ周辺デバイス間のシリアル通信を可能にします。SPI マスタはすべてのシリアル転送を制御します。

下図では、SPI マスタと他の SPI スレーブデバイスとの接続例を示します。SPI マスタによって生成され制御されるシリアルクロックは、SPI_CLK ラインで出力されます。SPI コントローラが無効（bSpi_SSIENR＝0）の場合、シリアル転送はできず、動作シリアルプトロコルに従って “非アクティブ”状態に保たれます。

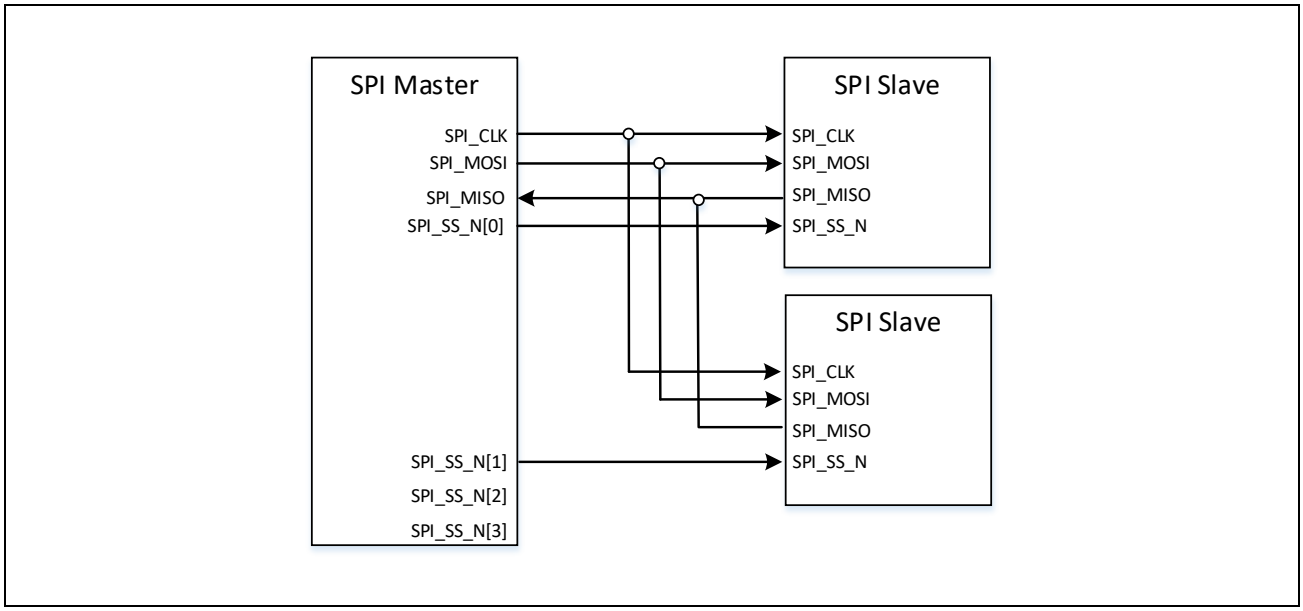


図 2.3　SPI マスタと SPI スレーブ間の一般的な接続

2.5.3　ハードウェアモードあるいはソフトウェアモードによるスレーブセレクトラインの制御

SPI コントローラは、シリアルマスタとシリアルスレーブ周辺デバイス間のシリアル通信を可能にし、2 つの基本モードで制御できます。

- ハードウェアモードで制御されるセレクトライン
- ソフトウェアモードで制御されるセレクトライン

rSpi_SER レジスタの以下のビットを参照してください。

- bSpi_CtrlSS[3:0]
- bSpi_HardwareSS[3:0]
- bSpi_SoftwareSS[3:0]

注　意

- SPI_SS_N (M)が、2 つの SPI データフレーム（各データフレーム長は 4～16 ビット）間でトグルする必要があり、ソフトウェアモードで制御する場合、ソフトウェアスレーブセレクトコントロールビット（bSpi_SoftwareSS[3:0]）により適切にトグルさせてください。
- モトローラシリアルペリフェラルインタフェースのフレームフォーマットで rSpi_CTRLR0 レジスタの bSpi_SCPH が 0 の場合、2 つの SPI データフレーム間で、SPI_SS_N (S)をトグルする必要があります。

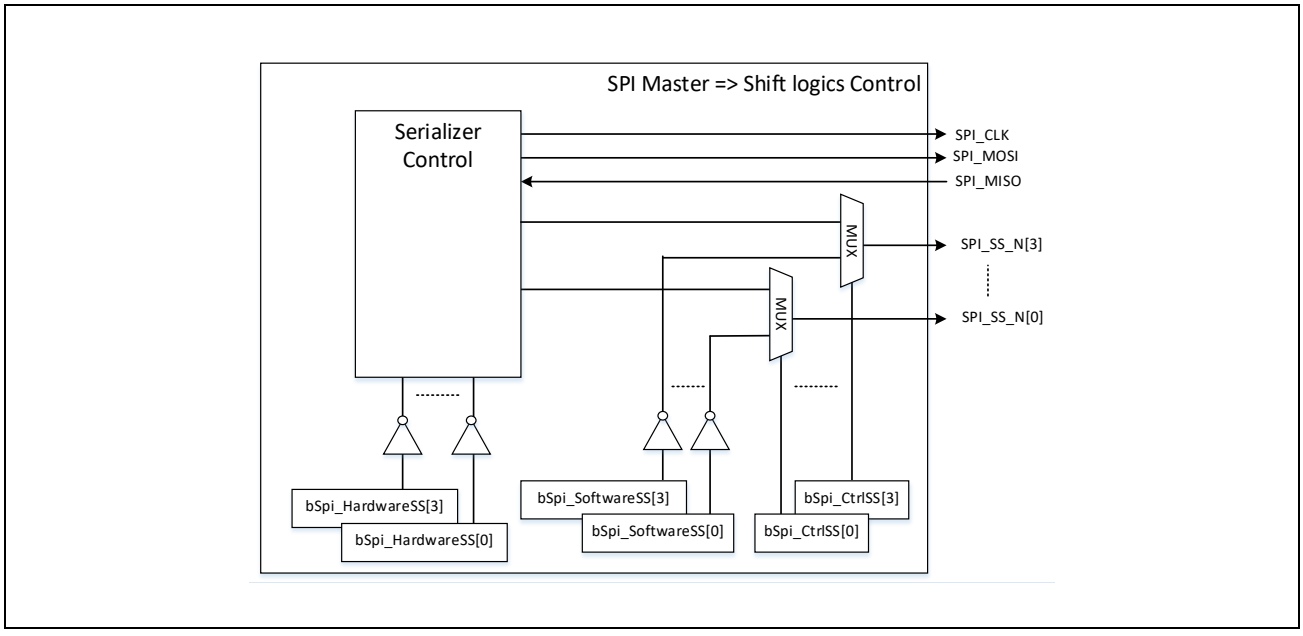


図 2.4　ハードウェアモードあるいはソフトウェアモードによるスレーブセレクトラインの制御

2.5.4　プログラマブルプリスケーラクロック

SPI_SCLK（シリアルリファレンスクロック、最大 125MHz）は、SPI_PCLK（APB）の周波数以下である必要があります。それにより、シリアルリファレンスクロック SPI_SCLK ドメインからの制御信号が SPI_PCLK ドメインと同期されることを保証します。SPI_PCLK および SPI_SCLK が非同期の場合、同期ロジックは 1 つのクロックドメインから他方へ制御信号を転送します。

シリアルクロック（SPI_CLK）とシリアルリファレンスクロック（SPI_SCLK）間の推奨周波数比の制限は、以下のようになります。

- マスタ：
  SPI_SCLK ≧ 2×（最大 SPI_CLK 出力）
- スレーブ（受信のみ）：
  SPI_SCLK ≧ 8×（最大 SPI_CLK 入力）
- スレーブ：
  SPI_SCLK ≧ 10×（最大 SPI_CLK 入力）

注　意

実際の最大周波数（SPI_CLK）は SPI の AC 仕様内で設定される必要があります。

2.5.5　データ入力サンプル遅延

SPI マスタは、SPI_MISO(M)信号のデフォルトのサンプルタイムを遅延させる追加ロジックを含みます。本追加ロジックは、シリアルバス上で実現可能な最大周波数を上げることに役立ちます。

マスタからの SPI_CLK(M)信号とスレーブからの SPI_MISO 信号における往復ルーティング遅延により、SPI_MISO（マスタ入力）信号のタイミングが通常のサンプリング時間から変わる可能性があります。以下の図は本状況を示します。

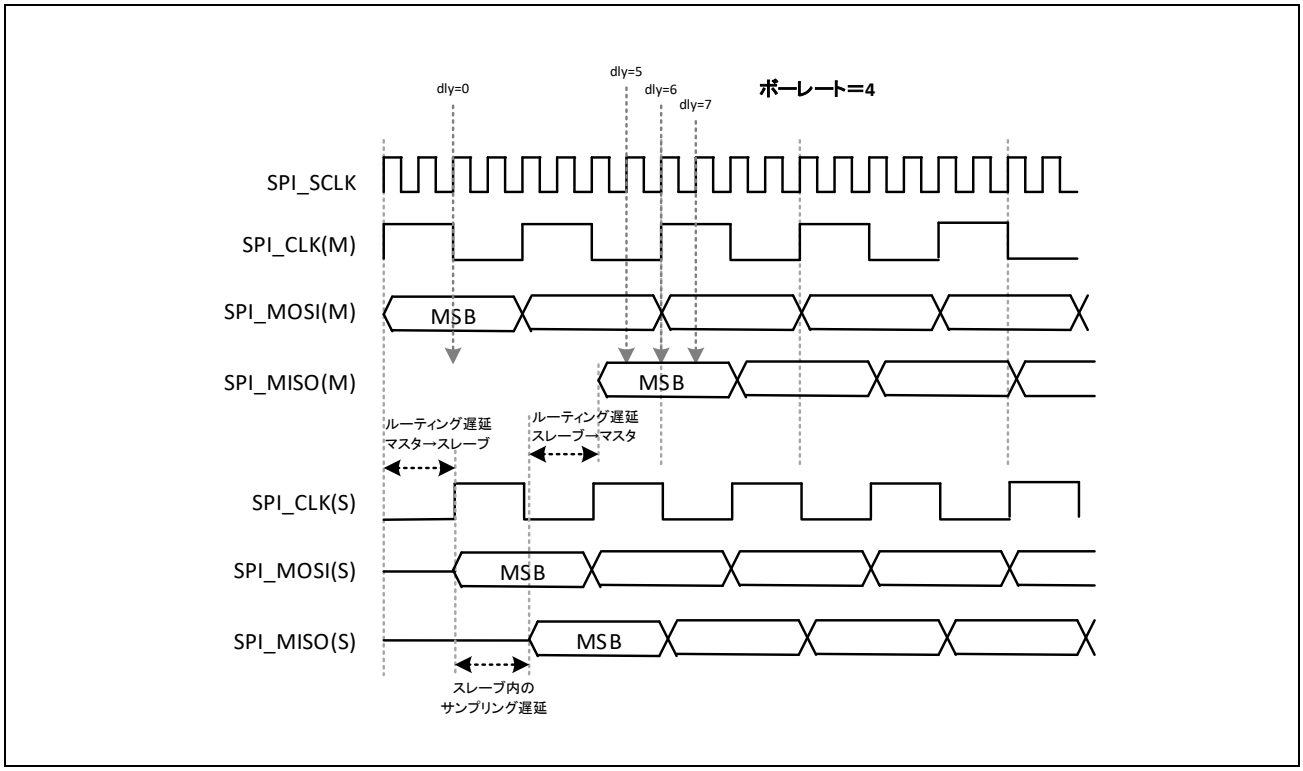


図 2.5　SPI データ入力サンプル遅延

注　意

dly＝0 の参照は、フォーマットの種類によって異なります。上記の図は、シリアルクロックの 2 番目のエッジでキャプチャします（たとえば、モトローラフォーマットでは、bSpi_SCPH＝1）。

SPI_MISO(S)信号データをシリアルバスで駆動するために、スレーブはマスタからの SPI_CLK(S)信号をストローブとして使用します。スレーブデバイスによる SPI_CLK(M)信号のルーティング遅延とサンプリング遅延により、マスタが SPI_MISO 信号をサンプリングする際、正しい値で安定化していない可能性があります。図「SPI データ入力サンプル遅延」では、SPI_MISO 信号でのルーティング遅延によって、マスタがポートをサンプリングする際に、デフォルト時間では誤った SPI_MISO 値になる一例を示します。

SPI_MISO 信号のサンプリング時間をデフォルトから変更する目的で、遅延値を動的にプログラム可能です。

サンプル遅延ロジックには、SPI_SCLK サイクル単位の分解能があります。ソフトウェアは、マスタによって正しいデータが受信されるまで、スレーブから連続的に読み出し、マスタの「データ入力サンプル遅延」値をインクリメントするようなループをコーディングすることでシリアルバスを「トレーニング」することが可能です。

2.5.6　送信 FIFO および受信 FIFO およびコントロール

送信 FIFO および受信 FIFO バッファの深さは 16 ワードです。送信 FIFO および受信 FIFO バッファの幅は 16 ビットに固定されています。16 ビット未満のデータフレームは、送信 FIFO バッファに書き込まれるときに右詰めされている必要があります。シフトコントロールロジックは、受信 FIFO バッファで受信データを自動的に右詰めします。

FIFO バッファの各データエントリには単一のデータフレームが含まれます。複数のデータフレームを単一の FIFO 位置に格納することは不可能です。たとえば、2 つの 8 ビットデータフレームを単一の FIFO 位置に格納できません。8 ビットデータフレームが必要な場合、シリアルシフタでデータを送信する際、FIFO エントリの上位 8 ビットは無視され使用されません。

SPI コントローラが無効（bSpi_SSIENR＝0）であるか、SPI モジュールがリセットされる場合、送信 FIFO バッファと受信 FIFO バッファはクリアされます。

送信 FIFO は CPU が SPI データレジスタ（rSpi_DR）に書き込むことでロードされます。データは、シフトコントロールロジックによって送信シフトレジスタに送信 FIFO から取り出されます。FIFO のエントリ数が FIFO しきい値以下の場合に、送信 FIFO は FIFO エンプティ割り込み要求（iSpi_TXE_Int）を生成します。

rSpi_TXFTLR で設定されたしきい値は、割り込みが生成されるときの FIFO エントリのレベルを決定します。しきい値によって、送信 FIFO がエンプティに近いということをプロセッサに早めに表示することが可能です。送信 FIFO オーバーフロー割り込み（iSpi_TXO_Int）は、すでにフルの送信 FIFO にデータを書き込もうとする場合に生成されます。

データは、SPI データレジスタ（rSpi_DR）への APB 読み出しコマンドによって受信 FIFO から取り出されます。受信 FIFO はシフトコントロールロジックによって受信シフトレジスタからロードされます。FIFO のエントリ数が FIFO しきい値＋1 以上の場合に、受信 FIFO は FIFO フル割り込み要求（iSpi_RXF_Int）を生成します。rSpi_RXFTLR で設定されたしきい値は、割り込みが生成されるときの FIFO エントリのレベルを決定します。

しきい値によって、受信 FIFO がフルに近いということをプロセッサに早めに表示することが可能です。受信 FIFO オーバーフロー割り込み（iSpi_RXO_Int）は、受信シフトロジックが完全にフルの受信 FIFO にデータをロードしようとするときに生成されます。ただし、新しく受信したデータは失われます。受信 FIFO アンダーフロー割り込み（iSpi_RXU_Int）は、エンプティの受信 FIFO から読み出そうとするときに生成されます。本割り込みは、読み出しデータが無効であるとプロセッサに警告するものです。

2.5.7　割り込み管理機能

SPI コントローラは、マスク可能な各割り込み要求や、その組み合わせをサポートします。組み合せの割り込み要求は、マスク後のほかのすべての SPI 割り込みの OR 演算の結果です。

SPI 割り込みを以下に示します。

- 送信 FIFO エンプティ割り込みステータス（iSpi_TXE_Int）
  - 送信 FIFO がしきい値以下のときに、アンダーランを防止するためのサービスを必要とする場合にセットします。
  - ソフトウェアプログラマブルレジスタで設定されたしきい値は、割り込みが生成されるときの送信 FIFO エントリのレベルを決定します。
  - 本割り込みは、データが送信 FIFO バッファに書き込まれ、しきい値レベルを超えるときに、ハードウェアによってクリアされます。
- 送信 FIFO オーバーフロー割り込み許可（iSpi_TXO_Int）
  - 完全にフルになった後、CPU アクセスが送信 FIFO に書き込もうとするときにセットされます。セットされると、CPU から書き込まれたデータは破棄されます。
  - 本割り込みは、送信 FIFO オーバーフロー割り込みクリアレジスタ（rSpi_TXOICR）を読み出すまで、セットされたままです。
- 受信 FIFO フル割り込み（iSpi_RXF_Int）
  - 受信 FIFO がしきい値以上で、オーバーフローを防止するためのサービスを必要とする場合にセットします。
  - ソフトウェアプログラマブルレジスタで設定されたしきい値は、割り込みが生成されるときの受信 FIFO エントリのレベルを決定します。
  - 本割り込みは、データが受信 FIFO バッファから読み出されて、しきい値レベル未満のときに、ハードウェアによってクリアされます。
- 受信 FIFO オーバーフロー割り込み（iSpi_RXO_Int）
  - 完全にフルになった後、受信ロジックが受信 FIFO にデータを配置しようとするときにセットされます。セットされると、新しく受信したデータは破棄されます。
  - 本割り込みは、受信 FIFO オーバーフロー割り込みクリアレジスタ（rSpi_RXOICR）を読み出すまで、セットされたままです。
- 受信 FIFO アンダーフロー割り込み（iSpi_RXU_Int）
  - CPU が受信 FIFO から読み出そうとして、それがエンプティのときにセットされます。
  - セットされると、受信 FIFO から 0 がリードバックされます。
  - 本割り込みは、受信 FIFO アンダーフロー割り込みクリアレジスタ（rSpi_RXUICR）を読み出すまで、セットされたままです。
- 割り込み要求の組合せ（SPI_Int）
  - マスク後のすべての上記割り込み要求の OR 演算の結果。
  - 本割り込み信号をマスクするには、他のすべての SPI 割り込み要求をマスクする必要があります。

2.5.8　転送モード

SPI シリアルトランザクションを実行するには、次の 4 つの転送モードがあります。

- 送受信。「**2.5.8.1 送受信モード**」を参照してください。
- 送信のみ。「**2.5.8.2 送信のみモード**」を参照してください。
- 受信のみ。「**2.5.8.3 受信のみモード**」を参照してください。
- EEPROM 読み出し。「**2.5.8.4 EEPROM 読み出しモード**」を参照してください。

転送モードは、コントロールレジスタ 0（rSpi_CTRLR0）の bSpi_TMOD に書き込むことで設定されます。

注　意

- **転送モード設定は、シリアル転送のデュプレックスに影響を与えません。**
- **bSpi_TMOD は Microwire 転送で無視され、rSpi_MWCR レジスタによって制御されます。**

2.5.8.1　送受信モード

bSpi_TMOD＝2’b00 の場合、送信と受信の両方のロジックが有効です。

データ転送は、選択されたフレームフォーマット（シリアルプロトコル）に従って通常どおり行われます。送信データは送信 FIFO から取り出され、SPI_MOSI ライン経由でターゲットデバイスに送信され SPI_MISO ライン上のデータで返答します。ターゲットデバイスからの受信データは受信シフトレジスタから受信 FIFO へ各データフレームの終わりで移動します。

2.5.8.2　送信のみモード

bSpi_TMOD＝2’b01 の場合、受信データは無効で、受信 FIFO に格納されることはありません。

データ転送は、選択されたフレームフォーマット（シリアルプロトコル）に従って通常どおり行われます。送信データは送信 FIFO から取り出され、SPI_MOSI ライン経由でターゲットデバイスに送信され SPI_MISO ライン上のデータで返答します。データフレームの終わりに、受信シフトレジスタは、新しく受信したデータを受信 FIFO にロードしません。受信シフトレジスタのデータは、次の転送によって上書きされます。本モードに入ったときは、受信ロジック由来の割り込みをマスクする必要があります。

2.5.8.3　受信のみモード

bSpi_TMOD＝2’b10 の場合、送信データは無効です。

SPI コントローラがスレーブのとき、受信のみモードでは送信 FIFO からデータは取り出されません。SPI_MISO 出力は、送信中、一定のロジックレベルで保持されます。データ転送は、選択されたフレームフォーマット（シリアルプロトコル）に従って通常どおり行われます。

ターゲットデバイスからの受信データは受信シフトレジスタから受信 FIFO へ各データフレームの終わりで移動します。本モードに入ったときは、送信ロジック由来の割り込みをマスクする必要があります。

2.5.8.4　EEPROM 読み出しモード

本転送モードは、マスタモードのみで有効です。

bSpi_TMOD＝2’b11 のとき、送信データは、EEPROM デバイスへのアドレスとオペコードのいずれかまたは両方を送信するために使用されます。一般的に、ここでは 3 つのデータフレーム（8 ビットオペコード、8 ビットの上位アドレス、8 ビットの下位アドレス）を得ます。オペコードとアドレスの送信中、受信ロジックによりデータはキャプチャされません（SPI マスタがその SPI_MOSI ライン上でデータ送信中であるかぎり、SPI_MISO ライン上のデータは無視されます）。送信 FIFO がエンプティになるまで SPI マスタはデータの送信を続行します。

このため、EEPROM へオペコードとアドレスを供給するために、送信 FIFO に十分なデータフレームのみを持つ必要があります。送信 FIFO に、必要以上に多くのデータフレームがある場合は、読み出しデータは失われます。

送信 FIFO がエンプティになるとき（すべての制御情報が送信済み）、受信ライン上のデータ（SPI_MISO）は有効で、受信 FIFO に格納されます。SPI_MOSI 出力は一定のロジックレベルに保持されます。シリアル転送は、SPI マスタによって受信するデータフレーム数が rSpi_CTRLR1 レジスタの bSpi_NDF＋1 に一致するまで続行します。

注　意

- EEPROM 読み出しモードは、マスタモードのみで有効です。
- SPI コントローラがテキサスインスツルメンツプロトコルで設定される場合、EEPROM 読み出しモードはサポートされません。

2.5.9　モトローラシリアルペリフェラルインタフェース

モトローラシリアルペリフェラルインタフェースは、4 線の全二重のシリアルリンクです。

シリアルクロック位相およびシリアルクロック極性の 4 つの組み合わせが可能です

- クロック位相（bSpi_SCPH）は、シリアル転送がスレーブセレクト信号の立ち下りエッジで開始するか、シリアルクロックの最初のエッジで開始するかを決定します。SPI リンクがアイドルまたは無効の場合は、スレーブセレクトラインは、High に保持されます。モトローラフォーマットでは、クロック極性（bSpi_SCPOL）コンフィグレーションビットは、シリアルクロックの非アクティブ状態が High か Low かを決定します。データを送信するには、両方の SPI 周辺デバイスは同じシリアルクロック位相（bSpi_SCPH）であり、クロック極性（bSpi_SCPOL）値が同一である必要があります。データフレームは 4～16 ビットの長さが可能です。bSpi_SCPH＝0 の場合、データ転送がスレーブセレクト信号の立ち下りエッジで開始します。最初のデータビットは、シリアルクロックの最初のエッジでマスタ周辺デバイスおよびスレーブ周辺デバイスによってキャプチャされるので、最初のシリアルクロックエッジの前に、有効データが SPI_MISO ラインおよび SPI_MOSI ライン上に存在する必要があります。
  下図は、bSpi_SCPH＝0 の場合の単一 SPI 転送のタイミング図です。シリアルクロックは、bSpi_SCPOL＝0 かつ bSpi_SCPOL＝1 に対して示します。

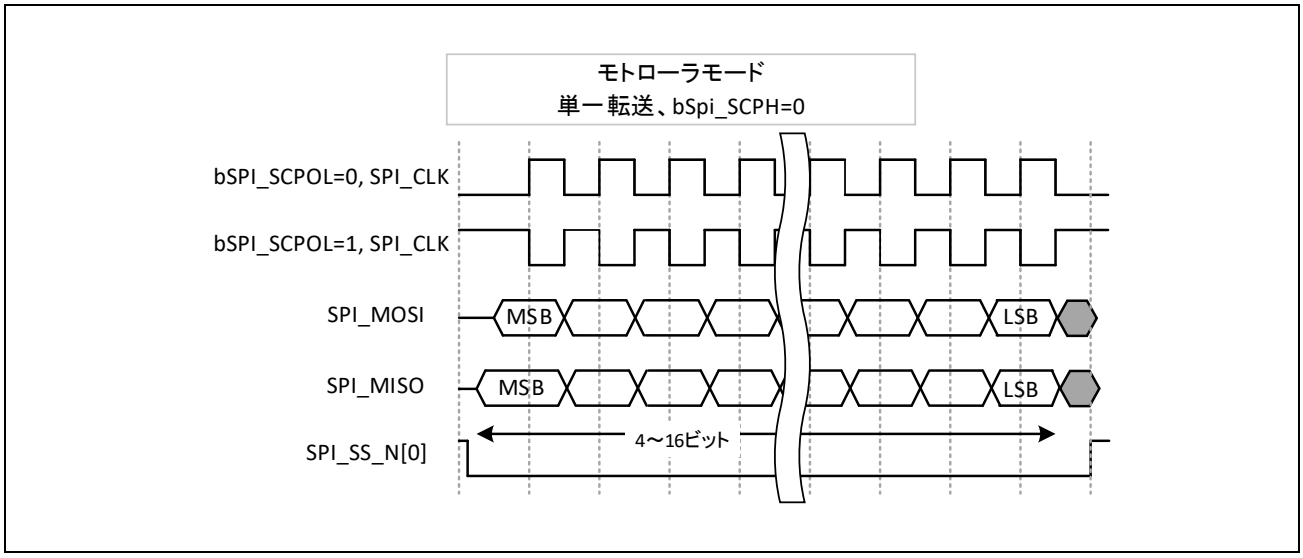


図 2.6　モトローラモード、単一転送、bSPI_SCPH＝0

- bSpi_SCPH＝0 の場合、本 SPI スレーブはスレーブセレクト信号の立ち下がりエッジでデータ転送を開始するため、連続データ転送では、次のデータフレームの開始前にスレーブセレクト信号がトグルする必要があります。本 SPI マスタは、ハードウェアモードで制御されるセレクトラインをデータフレーム間でトグル出力し、スレーブセレクト信号が非アクティブ High の間、シリアルクロックをデフォルト値に保持します。
  下図は、bSpi_SCPH＝0 の場合の連続 SPI 転送のタイミング図です。

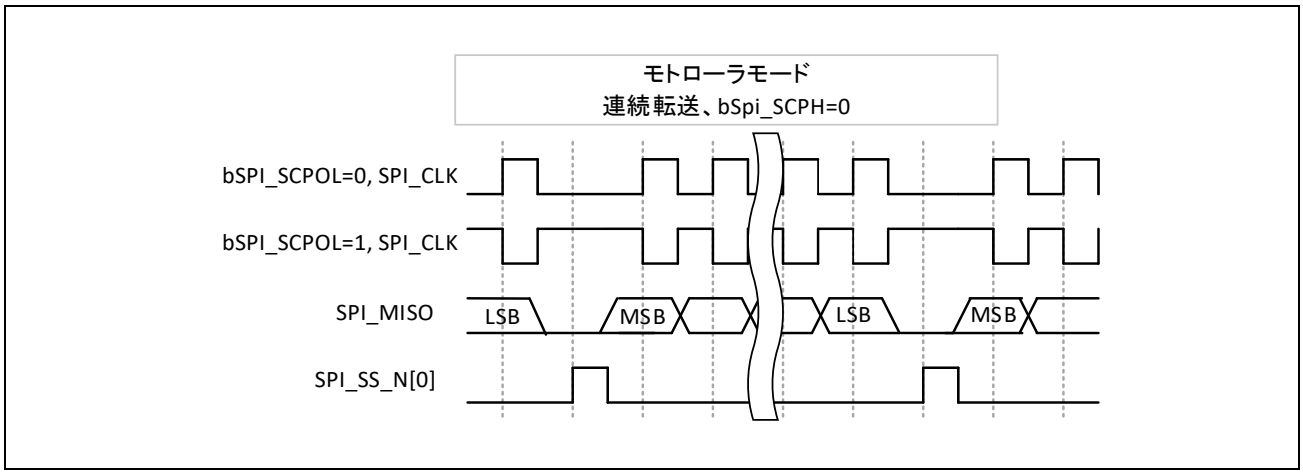


図 2.7　　モトローラモード、連続転送、bSpi_SCPH＝0

- bSpi_SCPH＝1 の場合、マスタ周辺デバイスとスレーブ周辺デバイスは両方とも、スレーブセレクトラインがアクティブ Low になった後に、最初のシリアルクロックエッジでデータの送信を開始します。最初のデータビットは、2 番目のシリアルクロックエッジでキャプチャされます。データは、シリアルクロックの最初のエッジでマスタ周辺デバイスおよびスレーブ周辺デバイスによって伝搬されます。データフレームの連続転送中、スレーブセレクトラインは、最終フレームの最終ビットがキャプチャされるまでアクティブ Low のままである可能性があります。
  下図は、bSpi_SCPH＝1 の場合の単一 SPI 転送のタイミング図です。

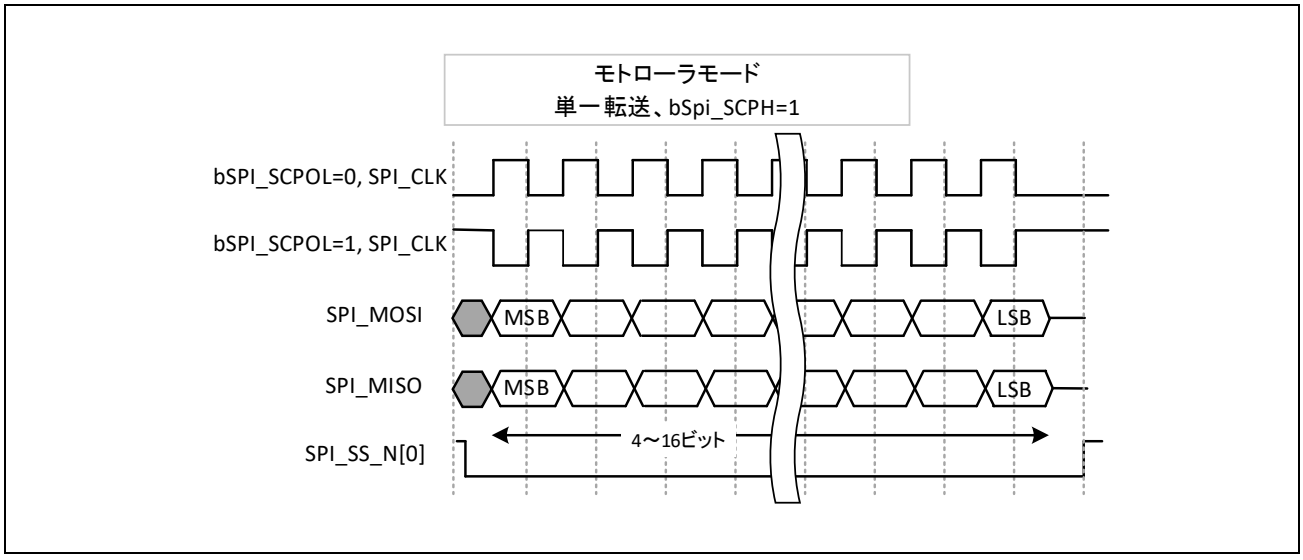


図 2.8　　モトローラモード、単一転送、bSpi_SCPH＝1

- 連続データフレームは、現在のフレームの LSB のすぐ後に続く次のフレームの MSB とともに、単一のフレームと同じ方法で転送されます。スレーブセレクト信号は、転送の間アクティブ Low に保たれます。
  下図は、bSpi_SCPH＝1 の場合の連続 SPI 転送のタイミング図です。

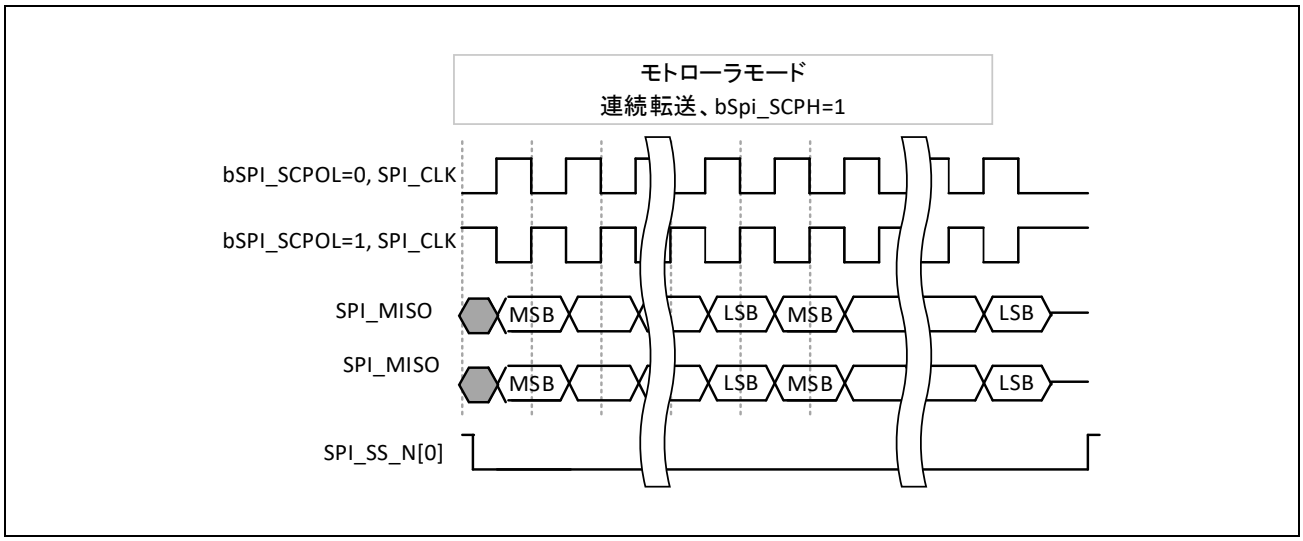


図 2.9　　モトローラモード、連続転送、bSpi_SCPH＝1

2.5.10　テキサスインスツルメンツ同期式シリアルプロトコル

データ転送は、1 シリアルクロック周期の間、フレームインジケータラインの SPI_SS_N[0]をアサートすることで始まります。

送信されるデータは、1 シリアルクロックサイクル後に SPI_MOSI ライン上で駆動され、同様にスレーブからのデータは SPI_MISO ライン上で駆動されます。データは、シリアルクロック SPI_CLK の立ち上がりエッジで伝搬され、立ち下がりエッジでキャプチャされます。データフレームの長さは 4～16 ビットの範囲です。

下図に、単一のテキサスシリアル転送用のタイミングの図を示します。

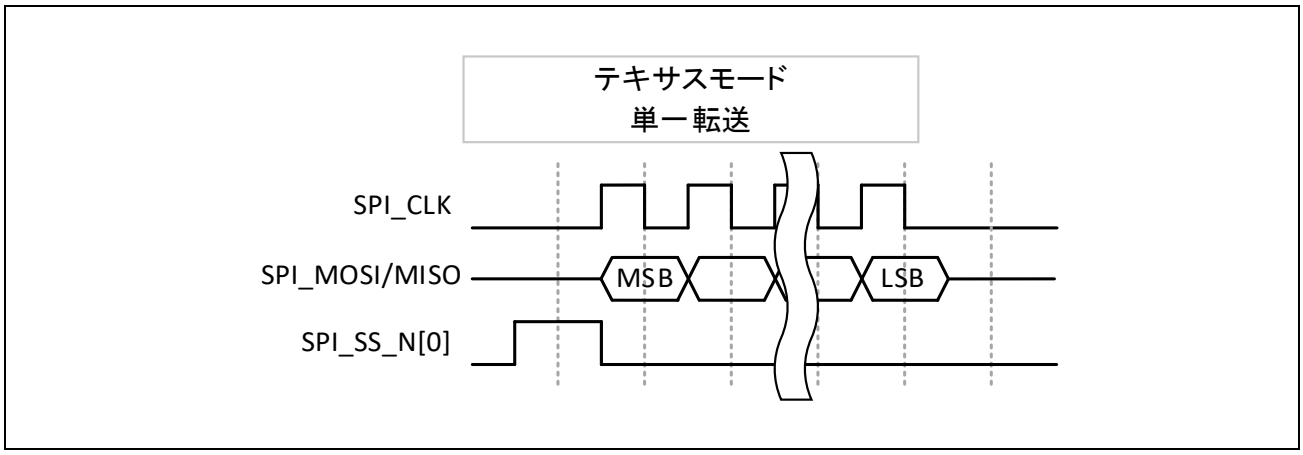


図 2.10　テキサスモード、単一転送

連続したデータフレームは、単一のデータフレームと同じ方法で転送されます。フレームインジケータは、現在の転送の LSB と同じサイクル中に 1 クロック周期の間アサートされ、別のデータフレームが続くことを意味します。

下図に、連続したテキサスシリアル転送用のタイミングを示します。

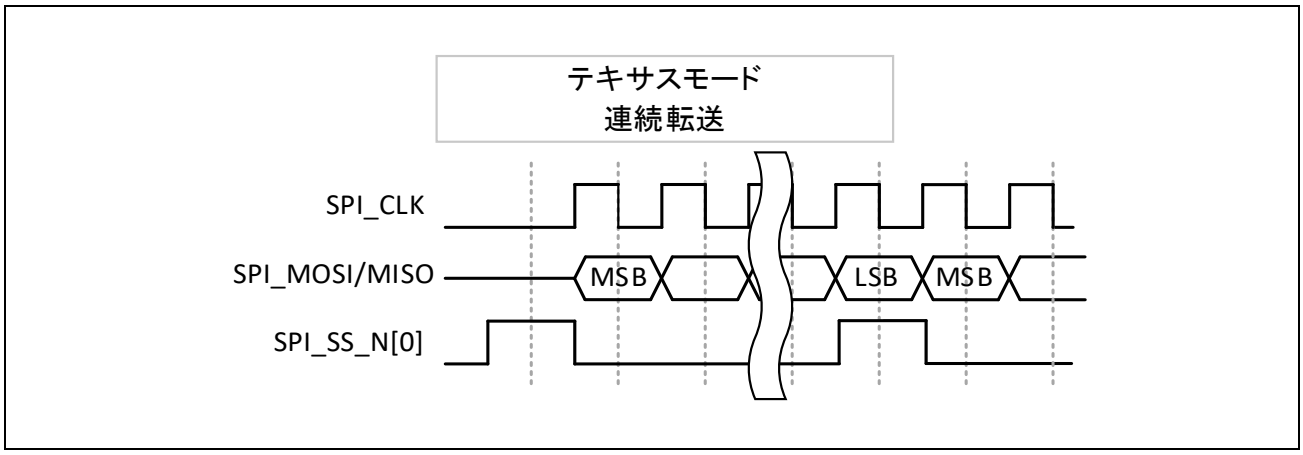


図 2.11　テキサスモード、連続転送

2.5.11　ナショナルセミコンダクターMicrowire

本モードでは、SPI コントローラがマスタの場合、データ転送は、スレーブセレクト信号 SPI_SS_N[0]の立ち下がりエッジで開始します。シリアルクロック SPI_CLK 周期の後、コントロールの最初のビットが SPI_MOSI ライン上で送信されます。

コントロールワードの長さは、1～16 ビットの範囲が可能であり、rSpi_CTRLR0 レジスタの bSpi_CFS（ビット[15:12]）の書き込みで設定されます。コントロールワードの残りが SPI マスタによって送信されます（SPI_CLK の立ち下がりエッジで伝搬されます）。本送信中、シリアルマスタの SPI_MISO ライン上にデータは存在しません。

データワードの方向は、Microwire コントロールレジスタ（rSpi_MWCR）の bSpi_MDD（ビット[1]）によって制御され、bSpi_MDD＝0 の場合、SPI マスタが外部シリアルスレーブからのデータを受信することを意味します。コントロールワードの LSB が送信された後の 1 クロックサイクル期間、スレーブ周辺デバイスはダミーの 0 ビットを応答し、それに続いて 4～16 ビットの長さのデータフレームを応答可能です。データは、シリアルクロックの立ち下がりエッジで伝搬され、立ち上がりエッジでキャプチャされます。

スレーブセレクトは、転送の間アクティブ Low に保持され、データ転送後半クロックサイクルでデアサートされます。下図に、外部シリアルスレーブからの単一の SPI マスタ読み出しのタイミングを示します。

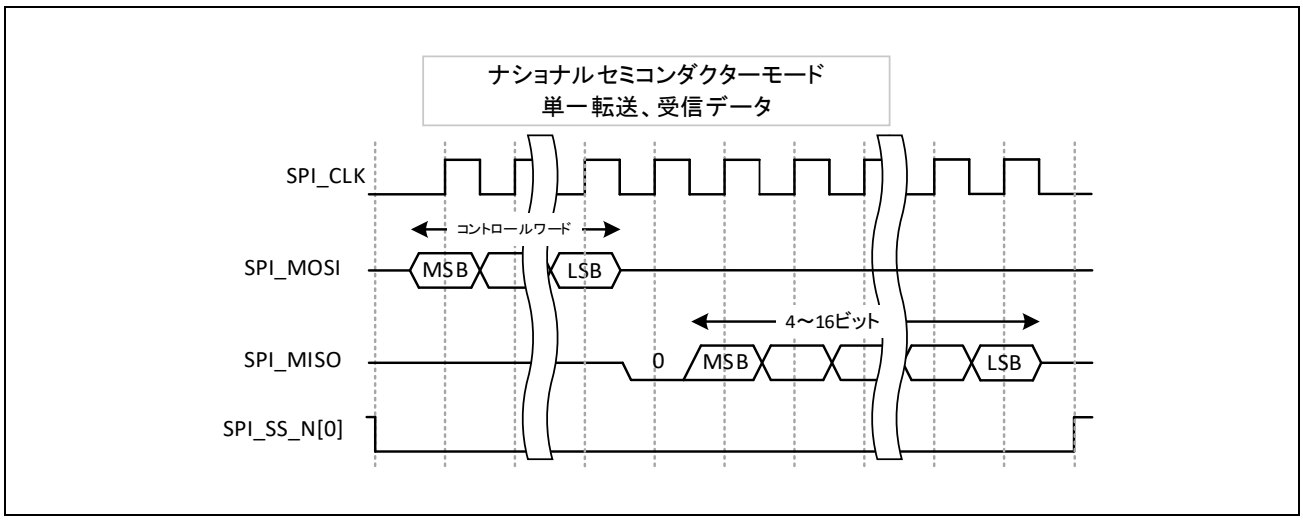


図 2.12　ナショナルセミコンダクターモード、単一転送、受信データ

Microwire プロトコルからの連続転送は、シーケンシャルまたは非シーケンシャルであることが可能で、rSpi_MWCR の bSpi_MWMOD（ビット[0]）によって制御されます。

非シーケンシャル連続転送は、現在のデータワードの LSB のすぐ後に続く次の転送用のコントロールワードとともに、以下の図で示したように行われます。

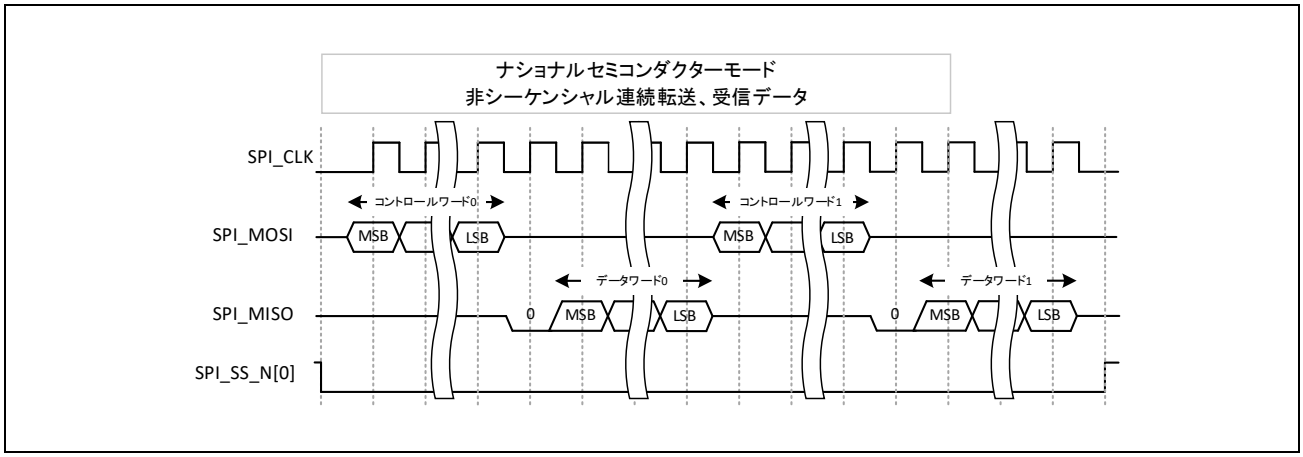


図 2.13　　ナショナルセミコンダクターモード、非シーケンシャル連続転送、受信データ

シーケンシャル連続転送の間、1 つのコントロールワードのみが SPI マスタから転送されます。転送は、シーケンシャルリードのない動作と同じ方法で開始されますが、さらにデータを読み出すために転送は継続されます。スレーブデバイスは、アドレスポインタを次の位置へ自動的にインクリメントし、その位置からデータを提供し続けます。この方法でどのような数も読み出すことが可能です。SPI マスタは、受信したワード数が rSpi_CTRLR1＋1 の値に等しくなったときに、この転送を終了します。

下図のタイミング図と例で、外部スレーブデバイスからの 2 つのデータフレームの連続シーケンシャル読み出しを示します。

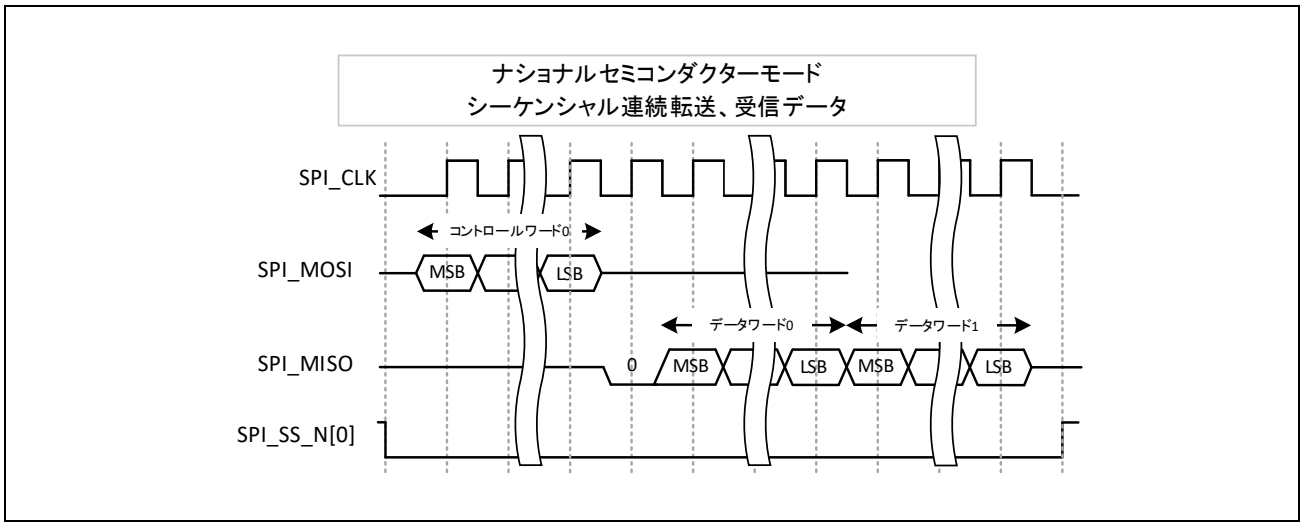


図 2.14　　ナショナルセミコンダクターモード、シーケンシャル連続転送、受信データ

bSpi_MDD＝1 の場合、これは、SPI マスタがデータを外部シリアルスレーブに送信することを示します。コントロールワードの LSB が送信されたすぐ後に、SPI マスタはスレーブ周辺デバイスにデータフレームの送信を続けます。下図に、外部シリアルスレーブへの単一シリアルマスタ書き込みタイミングを示します。

注　意

bSpi_MDD＝1 かつ bSpi_MWMOD＝1 の場合、SPI コントローラは連続シーケンシャル Microwire 書き出しをサポートしません。

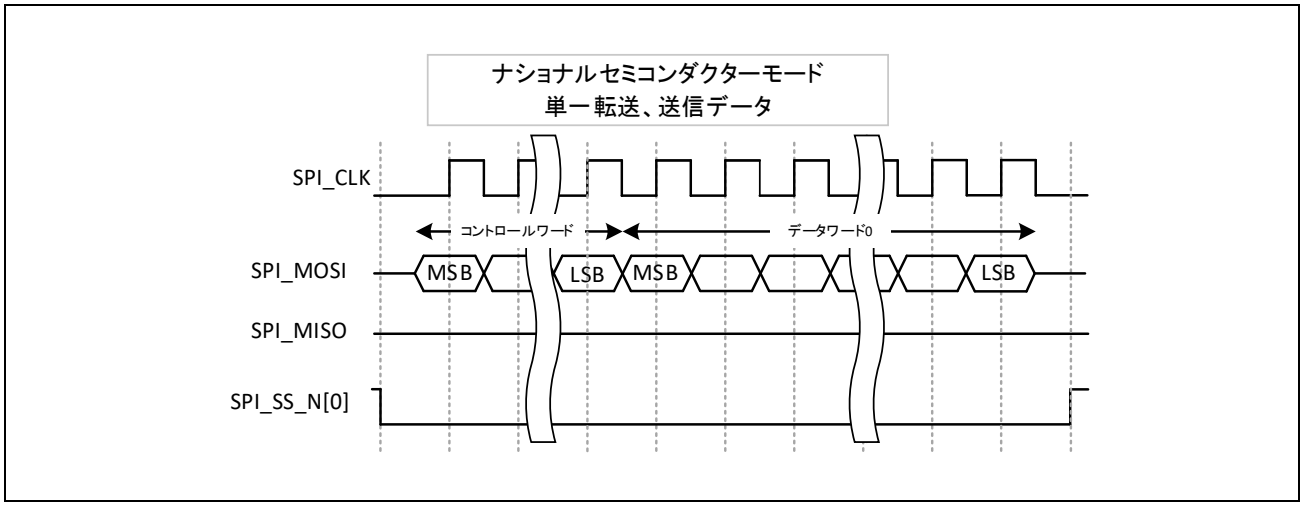


図 2.15　　ナショナルセミコンダクターモード、単一転送、送信データ

連続転送は、現在のデータワードの LSB のすぐ後に続く次の転送用のコントロールワードとともに、以下の図で示したように行われます。

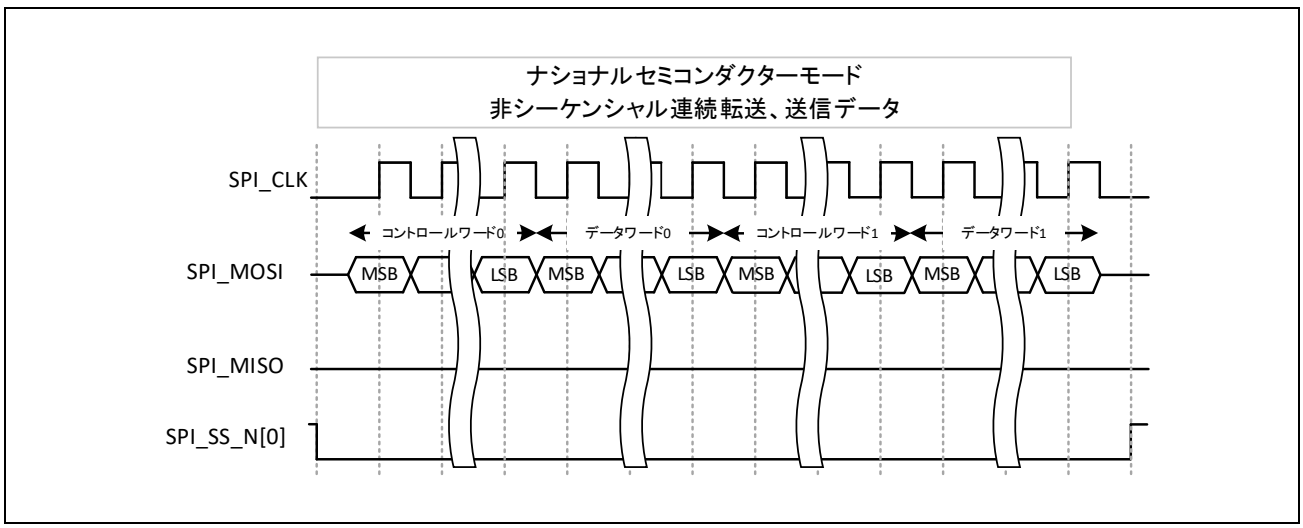


図 2.16　　ナショナルセミコンダクターモード、非シーケンシャル連続転送、送信データ

Microwire ハンドシェイクインタフェースを外部シリアルスレーブデバイスへの SPI マスタ書き込み動作のために有効にできます。ハンドシェイクインタフェースを有効にするには、rSpi_MWCR レジスタの bSpi_MWHS（ビット[2]）に 1 を書き込む必要があります。bSpi_MWHS が 1 にセットされた場合、転送を完了させる前、あるいは、連続転送の次のコントロールワードを送信する前に、SPI マスタは、スレーブデバイスがレディ状態になったかどうかをチェックします。

下図で、ハンドシェイクインタフェースが有効な連続 Microwire 転送の例を示します。

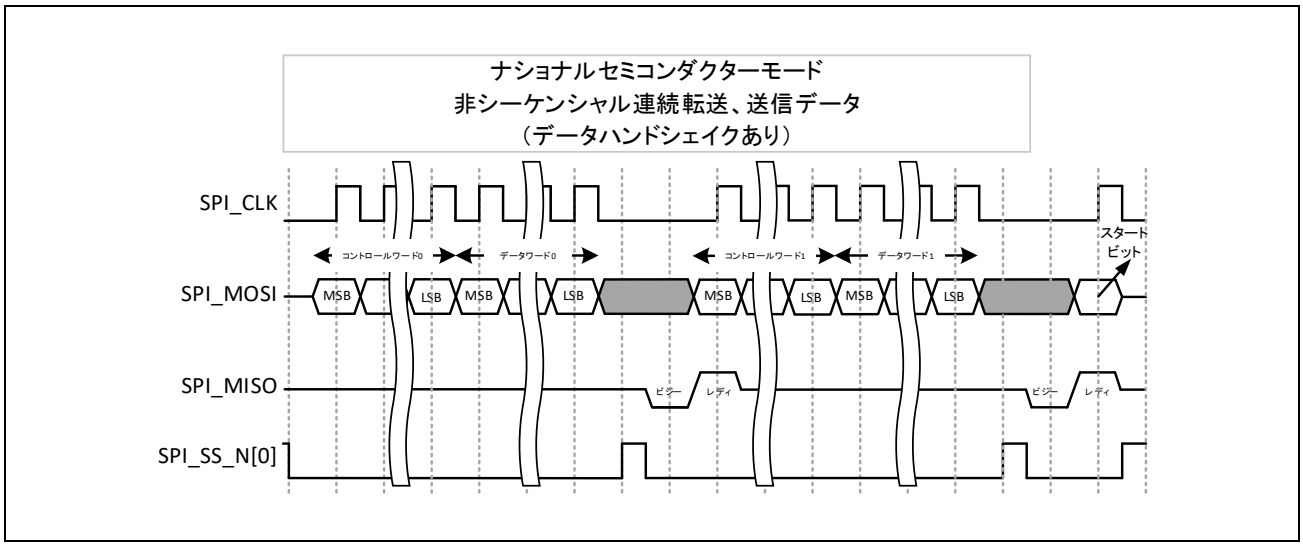


図 2.17　ナショナルセミコンダクターモード、非シーケンシャル連続転送、送信データ、ハンドシェイク

最初のワードがシリアルスレーブデバイスに送信された後、SPI マスタは SPI_MISO 入力をポーリングし、スレーブデバイスがレディ状態になるのを待ちます。レディ状態を受け入れてから、SPI マスタは次のコントロールワードの送信を開始します。最後のデータフレームの送信が完了した後、SPI マスタはスタートビットを送信して、転送完了前にスレーブデバイスのレディ状態をクリアします。

（後にデータが続いていない）コントロールワードを SPI マスタからシリアルスレーブデバイスに送信するには、送信 FIFO バッファに 1 つのエントリのみが存在する必要があります。連続転送では、SPI コントローラのシフトロジックは 2 番目のコントロールワードをデータワードとして扱うため、2 つのコントロールワードを送信できません。SPI マスタが 1 つのコントロールワードのみを送信する場合、bSpi_MDD（rSpi_MWCR レジスタのビット[1]）は 1 にセットされる必要があります。

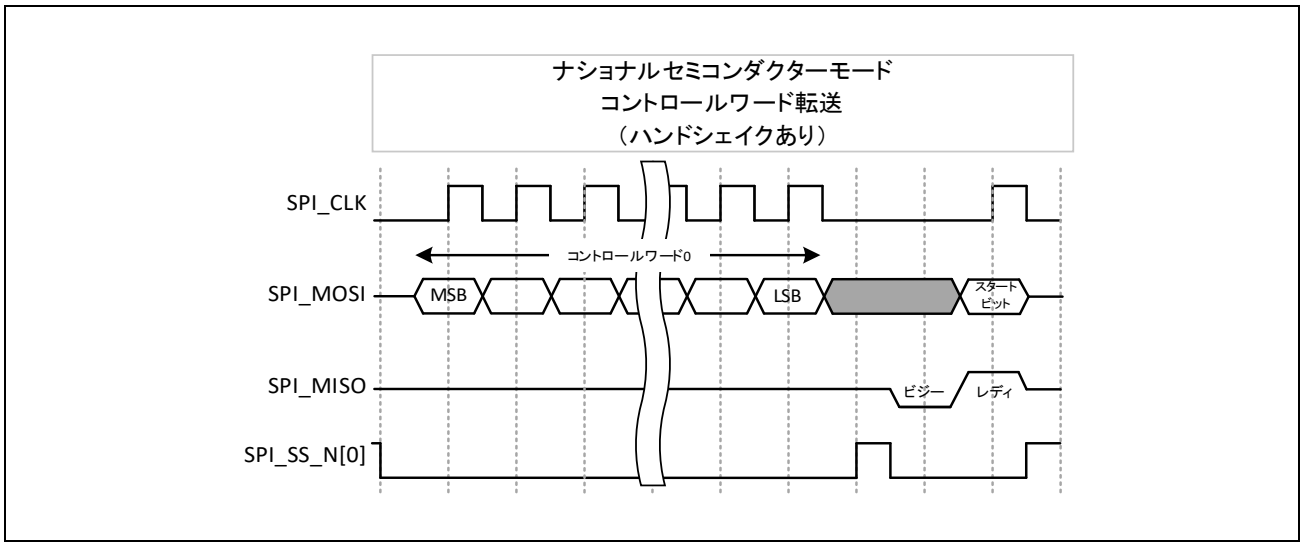


図 2.18　ナショナルセミコンダクターモード、コントロールワード転送、ハンドシェイク

SPI コントローラがスレーブの場合、データ転送は、スレーブセレクト信号（SPI_SS_N[0]）の立ち下がりエッジで開始します。シリアルクロック（SPI_CLK）半周期の後、コントロールの最初のビットが SPI_MOSI ラインに出力されます。コントロールワードの長さは、1～16 ビットの範囲が可能で、rSpi_CTRLR0 レジスタの bSpi_CFS で設定されます。

bSpi_CFS は、シリアルマスタからのコントロールワードサイズに設定する必要があります。コントロールワードの残りが SPI スレーブによって受信されます（SPI_CLK の立ち上がりエッジでキャプチャされます）。本受信中、シリアルスレーブの SPI_MISO ラインにデータは出力されません。

データワードの方向は、Spi_MWCR レジスタの bSpi_MDD（ビット[1]）によって制御され、bSpi_MDD＝0 の場合、SPI スレーブが外部シリアルマスタからのデータを受信することを意味します。コントロールワードが送信されたすぐ後に、シリアルマスタは、SPI_MOSI 上にデータフレームを出力し始めます。データは、シリアルクロックの立ち下がりエッジで伝搬され、立ち上がりエッジでキャプチャされます。スレーブセレクトは、転送の間アクティブ Low に保持され、データ転送後半クロックサイクルでデアサートされます。

下図に、外部シリアルマスタからの単一 SPI スレーブ読み出しタイミングを示します。

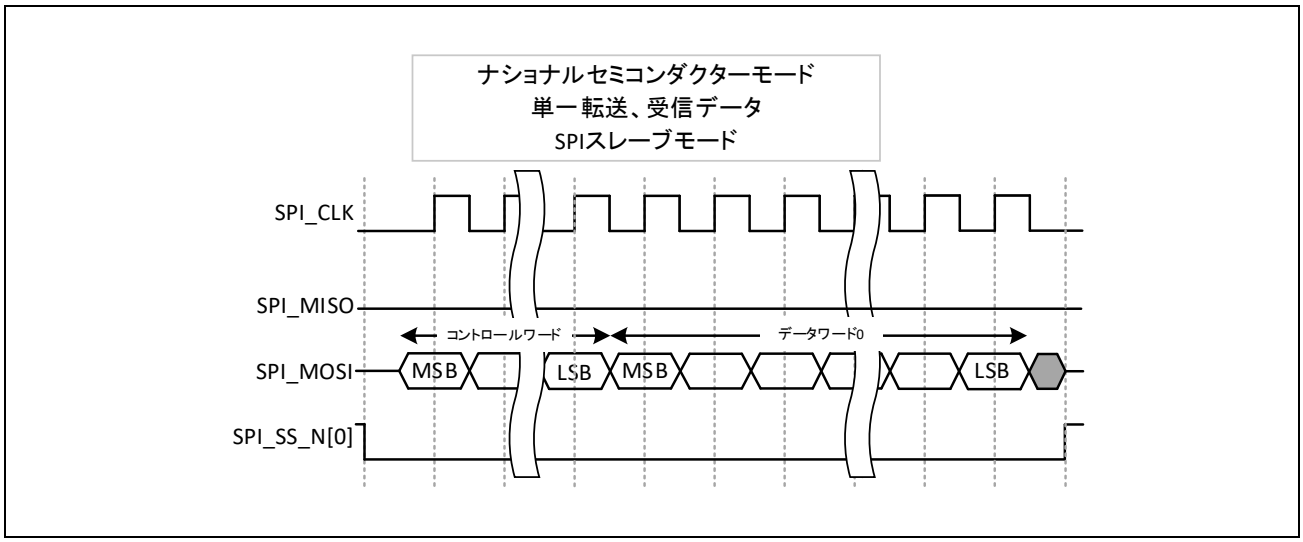


図 2.19　ナショナルセミコンダクターモード、単一転送、受信データ、スレーブモードの SPI コントローラ

bSpi_MDD＝1 の場合、これは、SPI スレーブがデータを外部シリアルマスタに送信することを示します。コントロールワードの LSB が送信されたすぐ後に、SPI スレーブはダミーの 0 ビットとそれに続く 4～16 ビットのデータフレームを SPI_MISO ラインで送信します。

下図に、外部シリアルマスタへの単一の SPI スレーブ送信のタイミングを示します。

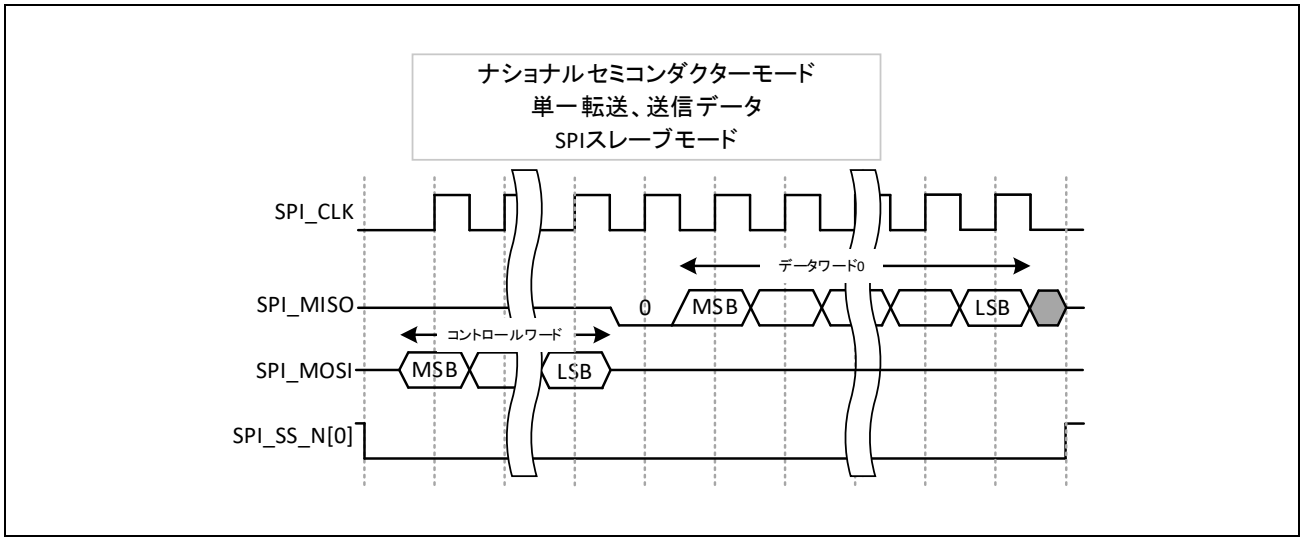


図 2.20　ナショナルセミコンダクターモード、単一転送、送信データ、スレーブモードの SPI コントローラ

SPI スレーブの連続転送は、SPI マスタで指定されたものと同じ方法で行われます。SPI スレーブは、ビジー期間がないためハンドシェイクインタフェースをサポートしません。

2.5.12　DMA 制御

SPI コントローラには DMA 機能があります。また、転送を要求したり制御したりするための、DMA コントローラへの要求インタフェースがあります。DMA からのデータ転送または DMA へのデータ転送の実行に APB バスが使用されます。効率性のため、DMA は可能なかぎりデータを常に DMA バーストトランザクションを使用して転送します。

注　意

本モードでは、DMA コントローラは、周辺機能フローコントローラモードに設定する必要があります。

SPI コントローラは、2 本の DMA チャネルを使用します（送信データ用に 1 本、受信データ用に 1 本）。SPI コントローラには次の DMA レジスタがあります。

- DMA 動作を許可する rSpi_DMACR コントロールレジスタ
- rSpi_TDMACR、rSpi_RDMACR：以下の用途のコントロールレジスタ
  - DMA 動作コンフィグレーション（ブロックサイズおよびバーストトランザクションサイズ）
  - DMA 転送の開始と停止
- rSpi_DMATDLR：DMA 要求が生成される送信 FIFO レベルを設定するレジスタ
- rSpi_DMARDLR：DMA 要求が生成される受信 FIFO レベルを設定するレジスタ

SPI コントローラ上で DMA コントローラインタフェースを有効にし、送信要求インタフェースを有効にするには：

- rSpi_DMACR レジスタの bSpi_TDMAE に 1 を書き込む
- rSpi_TDMACR レジスタの bSpi_TDMAE1 に 1 を書き込む

SPI コントローラ上で DMA コントローラインタフェースを有効にし、受信要求インタフェースを有効にするには：

- rSpi_DMACR レジスタの bSpi_RDMAE に 1 を書き込む
- rSpi_RDMACR レジスタの bSpi_RDMAE1 に 1 を書き込む

ブロックサイズとバーストトランザクション長を設定するには：

- 送信 FIFO および受信 FIFO 用の rSpi_TDMACR レジスタおよび rSpi_RDMACR レジスタの DEST_BLOCK_SIZE フィールド／SRC_BLOCK_SIZE フィールドにそれぞれブロックサイズを設定します。
- 送信 FIFO および受信 FIFO 用の rSpi_TDMACR レジスタおよび rSpi_RDMACR レジスタの DEST_BURST_SIZE フィールド／SRC_BURST_SIZE フィールドにそれぞれトランザクション長を設定します。

2.5.12.1　DMA 動作の概要

SPI コントローラはプロセッサにより、SPI コントローラが送信あるいは受信するデータ項目数（ブロックサイズ）が設定されている必要があります。これは、送信 FIFO および受信 FIFO それぞれについて、SPI コントローラの rSpi_TDMACR レジスタおよび rSpi_RDMACR レジスタの DEST_BLOCK_SIZE フィールドおよび SRC_BLOCK_SIZE フィールドに設定されます。ブロックは複数のトランザクションに分割され、それぞれのトランザクションは SPI コントローラからの要求で開始されます。

DMA コントローラ&SPI コントローラは、各 DMA 要求に対してバーストトランザクションにより転送されるデータバイト数も設定する必要があります。これは、バーストトランザクション長と呼ばれ、以下のように設定されます。

DMA コントローラ：

送信 FIFO および受信 FIFO それぞれについて、DMAC.CTL[n]レジスタの DEST_MSIZE フィールドおよび SRC_MSIZE フィールド

SPI コントローラ：

送信 FIFO および受信 FIFO それぞれについて、SPI コントローラの rSpi_TDMACR レジスタの DEST_BURST_SIZE フィールドおよび rSpi_RDMACR レジスタの SRC_BURST_SIZE フィールド

注　意

- バーストトランザクションサイズは、DMA コントローラおよび SPI コントローラともに同じ値でなければなりません。
- DMA コントローラのソースおよび宛先転送幅の設定である、DMAC.CTL[n].SRC_TR_WIDTH および DMAC.CTL[n].DST_TR_WIDTH は、SPI FIFO が 16 ビット幅であるため、3'b001 に設定される必要があります。

2.5.12.2　送信ウォーターマークレベルおよび送信 FIFO アンダーフロー

SPI シリアル転送中、送信 FIFO 内のエントリ数が、ウォーターマークレベルとして知られる DMA 送信データレベルレジスタ rSpi_DMATDLR の値以下の場合、送信 FIFO 要求が DMA コントローラに対して行われます。DMA は、送信 FIFO バッファに、DMAC.CTL[n].DEST_MSIZE＝DEST_BURST_SIZE 長のバーストデータを書き込むみます。

データは、送信 FIFO がシリアル転送を連続的に行うために十分な頻度で DMA からフェッチされる必要があります。すなわち、FIFO エンプティになりそうなとき、別の DMA 要求がトリガされなければなりません。さもなければアンダーフローします。このような状況が起こらないようにするために、ユーザはウォーターマークレベルを正しく設定する必要があります。

2.5.12.3　送信ウォーターマークレベルの選択

以下の例を考えてみます。

- DEST_BURST_SIZE＝DMAC.CTL[n].DEST_MSIZE＝FIFO_DEPTH − bSpi_DMATDLR
- ここで、DMA バーストで転送されるデータアイテムの数は、送信 FIFO 内の空きスペースと同じです。

2 つのウォーターマークレベル設定について考えてみます。

- DEST_BLOCK_SIZE/DEST_BURST_SIZE＝42/14＝3
- DMA ブロック転送でのバーストトランザクション数は 3 です。しかし、ウォーターマークレベルの bSpi_DMATDLR は極めて低くなります。したがって、SPI シリアル送信ラインがデータを送信しなければならないにもかかわらず送信 FIFO にデータが残っていない場合、SPI アンダーフローが起きる可能性が高くなります。これは、送信 FIFO がエンプティになる前に、DMA コントローラが DMA 要求を処理する時間がなかったために起こります。

以下の図で、必要なバーストトランザクション数は、バーストあたりのデータアイテム数で割ったブロックサイズとなります。

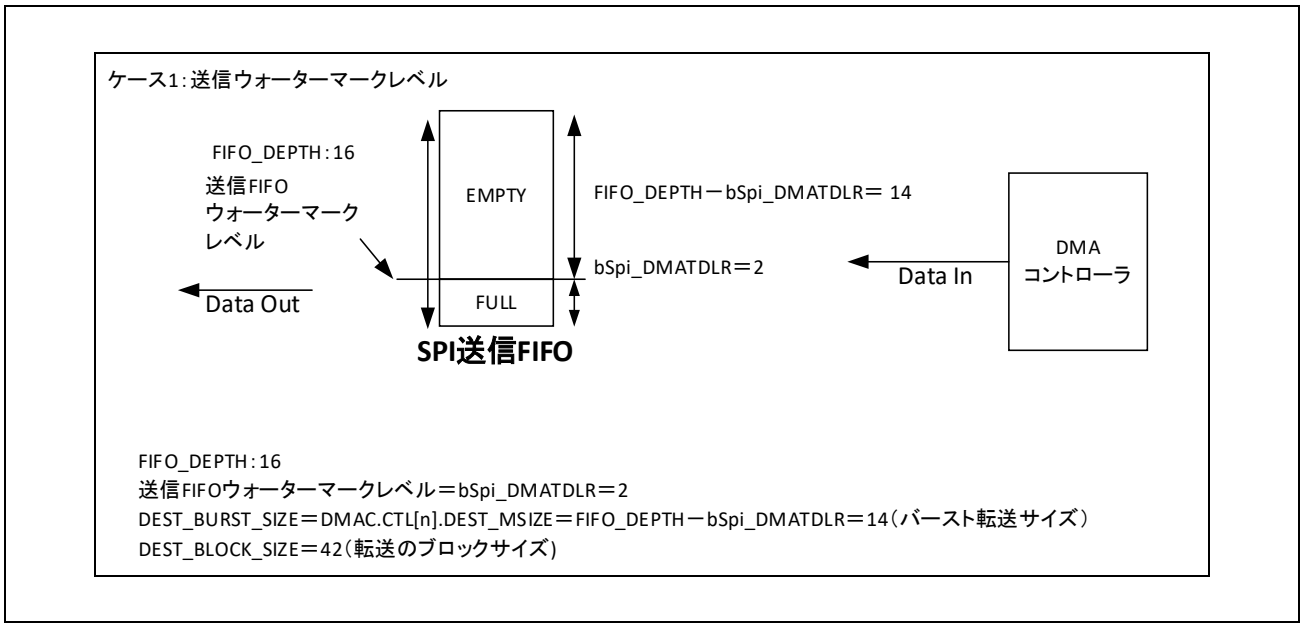


図 2.21　SPI ケース 1：送信ウォーターマークレベル

2 番目のケースでのブロックのバーストトランザクション数：

- DEST_BLOCK_SIZE/DEST_BURST_SIZE＝42/2＝21
- 本ブロック転送では、DMA ブロック転送に 21 個のバーストトランザクションがあります。しかしウォーターマークレベルの bSpi_DMATDLR は極めて高くなります。したがって、DMA コントローラは SPI 送信 FIFO がエンプティになる前に宛先バーストトランザクション要求を処理する時間が十分にあるので、SPI アンダーフローが起こる可能性は低くなります。
- したがって、2 番目のケースは、1 ブロックにつき送信可能なバーストトランザクション数が多くなり、アンダーフローの確率が低くなります。これは、前のケースよりも 1 ブロックにつきより多くの要求バーストを送ることになり、バス効率が悪くなります。

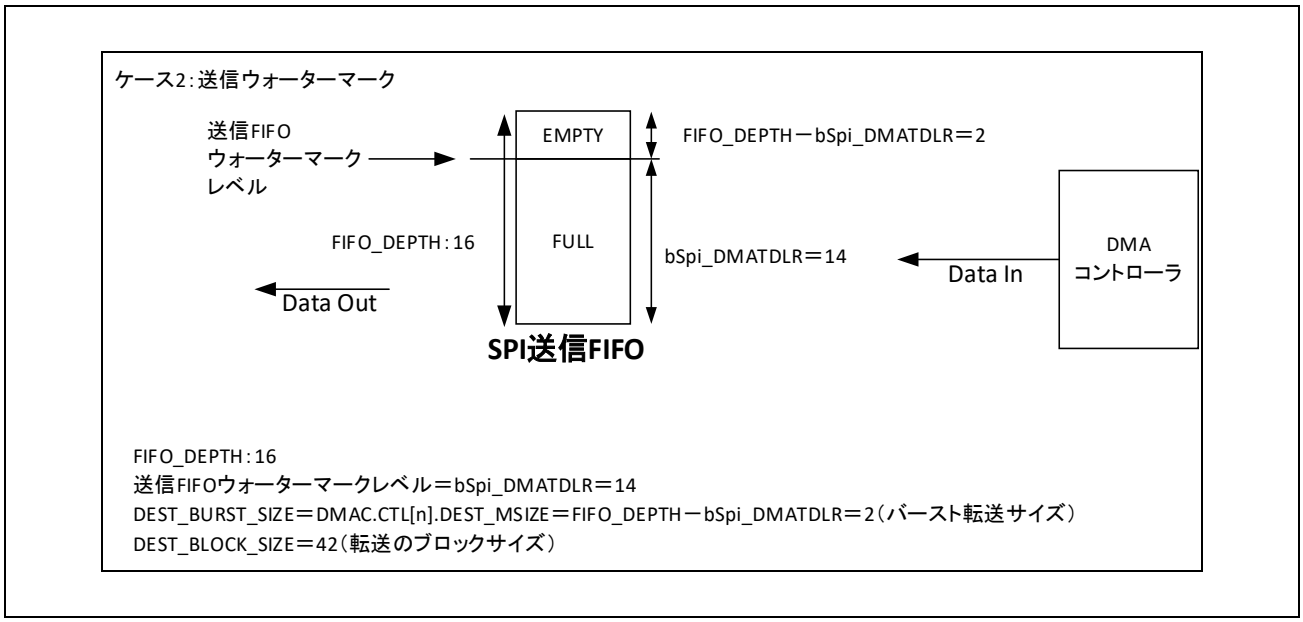


図 2.22　SPI ケース 2：送信ウォーターマークレベル

したがって、ウォーターマークレベル選択の目的は、アンダーフローになる確率を許容可能なレベルに抑えながら、ブロックあたりのトランザクション数を最小限に抑えることにあります。実際、これは、SPI コントローラが送信するデータ速度に DMA コントローラが宛先バースト要求に応答できる速度を合わせる機能です。

たとえば、チャネルを DMA の最優先チャネルに格上げし、DMA マスタインタフェースをバスレイヤの最優先マスタに格上げすると、DMA コントローラがバーストトランザクション要求に応答可能な速度を上げることになります。これは、逆に、ユーザがウォーターマークレベルを下げることを可能にします。ウォーターマークレベルを下げることによりバス効率を上げることができます。

2.5.12.4　DEST_MSIZE の選択および送信 FIFO オーバーフロー

SPI 送信 FIFO に宛先バースト要求を処理するための十分なスペースがない場合、アンダーフローとなる可能性があります。

したがって、適した動作のためには、以下のように設定する必要があります。

- DMAC.CTL[n].DEST_MSIZE＝DEST_BURST_SIZE＝4
- bSpi_DMATDLR＝8 に設定

もしくは

- DMAC.CTL[n].DEST_MSIZE＝DEST_BURST_SIZE＝8
- bSpi_DMATDLR＝8 に設定

2.5.12.5　受信ウォーターマークレベルおよび受信 FIFO オーバーフロー

SPI シリアル転送中、受信 FIFO 内のエントリ数が DMA 受信データレベルレジスタ、つまり bSpi_DMARDLR＋1 以上の場合、受信 FIFO 要求が DMA コントローラに対して行われます。

本レベルをウォーターマークレベルと言います。DMA コントローラは、受信 FIFO バッファから、SRC_BURST_SIZE＝DMAC.CTL[n].SRC_MSIZE 長のバーストデータを読み出すことにより応答します。

データは、受信 FIFO がシリアル転送を連続的に受け入れるために十分な頻度で DMA からフェッチされる必要があります。すなわち、FIFO がフルになりそうなとき、別の DMA 転送が要求されなければなりません。さもなければオーバーフローします。このような状況が起こらないようにするために、ユーザはウォーターマークレベルを正しく設定する必要があります。

2.5.12.6　受信ウォーターマークレベルの選択

上述の送信ウォーターマークレベルの選択と同様に、受信ウォーターマークレベル bSpi_DMARDLR＋1 は、以下の図に示すようにオーバーフローの確率を最小限に抑えるように設定する必要があります。

これは、ブロックあたりに必要な DMA バーストトランザクション数とオーバーフローが起こる確率のトレードオフになります。

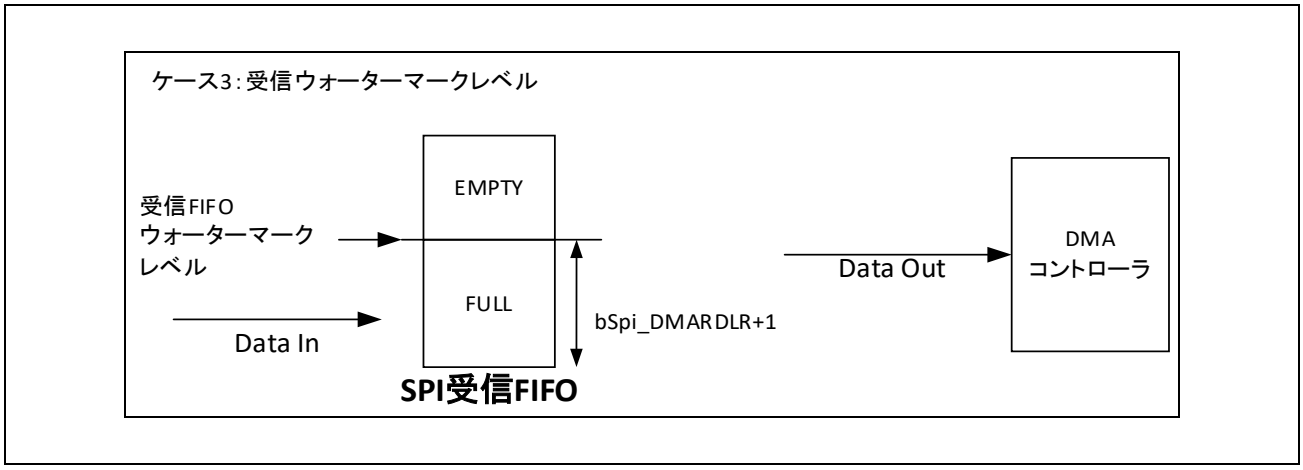


図 2.23　SPI ケース 3：受信ウォーターマークレベル

2.5.12.7　SRC_MSIZE の選択および受信 FIFO アンダーフロー

送信元バースト要求を処理するために十分なデータがない場合アンダーフローとなる可能性があります。

したがって、適した動作のためには、以下のように設定する必要があります。

- DMAC.CTL[n].SRC_MSIZE＝SRC_BURST_SIZE＝4
- bSpi_DMARDLR＝3 に設定

もしくは

- DMAC.CTL[n].SRC_MSIZE＝SRC_BURST_SIZE＝8
- bSpi_DMARDLR＝7 に設定

2.6　使用上の注意事項

2.6.1　プログラミング上の注意事項

注　意

- データが送信 FIFO に書き込まれる場合は、BUSY ステータスはセットされていません。本ビットは、ターゲットスレーブが選択され、転送中の場合のみセットされます。
- データを送信 FIFO に書き込んだ後、シフトロジックは、SPI_CLK 信号のポジティブエッジまで、シリアル転送を開始しません。ポジティブエッジ待機の遅延は、シリアル転送のボーレートによって異なります。
- BUSY ステータスをポーリングする前に、bSpi_TFE ステータス（1 を待機）を最初にポーリングするか、rSpi_BAUDR×SPI_PCLK クロックサイクルを待つ必要があります。
- BUSY ステータス（rSpi_SR.bSpi_BUSY）をポーリングすることにより、シリアル転送が完了したかどうかを判断することが可能です。

2.6.1.1　モトローラ&テキサスモードでのマスタ SPI のプログラミング

モトローラシリアルペリフェラルインタフェースおよびテキサスインスツルメンツ同期式シリアルプロトコルの項では、それぞれのシリアルプロトコルについて説明しています。シリアル転送の前後で送受信の FIFO データが構造化される方法を説明しています。

転送モードが、送受信モードまたは送信のみモードの場合（それぞれ bSpi_TMOD＝2’b00 あるいは bSpi_TMOD＝2’b01）、送信 FIFO がエンプティのときに、転送はシフトコントロールロジックによって終了します。

連続データ転送では、すべてのデータが送信される前に、送信 FIFO バッファがエンプティにならないことを保証する必要があります。送信 FIFO しきい値レベル（rSpi_TXFTLR）は、プロセッサに早めに割り込み（iSpi_TXE_Int）、送信 FIFO バッファがほとんどエンプティであることを示すために使用されます。DMA が APB アクセスに対して使用される場合、送信データレベル（rSpi_DMATDLR）は、DMA コントローラに早めに要求を出し、送信 FIFO がほとんどエンプティであることを示すために使用できます。その後、FIFO は再び満たされたデータでシリアル転送を続行可能です。bSpi_HardwareSS によって、シリアルスレーブを有効にする前に、データブロック（最低 2 つの FIFO エントリ）を送信 FIFO に書き込むこともできます。これによって、シリアル送信は、送信 FIFO にあるデータフレームの数が連続転送を構成する数になるまで開始しません。

転送モードが受信のみモードの場合（bSpi_TMOD＝2’b10）、シリアルスレーブが選択されたときに、シリアル転送は 1 つの“ダミー”データワードを送信 FIFO に書き込むことによって開始されます。SPI からの SPI_MOSI 出力が、シリアル転送の期間中一定のロジックレベルに保持されます。データは送信 FIFO から、開始時に一度だけ取り出され、シリアル転送の期間中エンプティであり続ける可能性があります。シリアル転送の終わりは、コントロールレジスタ 1（rSpi_CTRLR1）の“データフレーム数”（bSpi_NDF）フィールドでコントロールされます。

たとえば、24 のデータフレームをシリアルスレーブ周辺デバイスから受信する場合、bSpi_NDF フィールドを値 23 でプログラムする必要があり、受信ロジックは、受信したフレーム数が bSpi_NDF の値＋1 に等しくなったときに、シリアル転送を終了させます。本転送モードは、送信 FIFO がバスを使用しないため、APB バスの帯域幅を増加させます。受信 FIFO バッファは、オーバーフローを防ぐために、受信 FIFO が FIFO フル割り込み要求を生成するたびに読み出される必要があります。

転送モードが EEPROM 読み出しモードの場合（bSpi_TMOD＝2’b11）、シリアル転送は、シリアルスレーブ（EEPROM）が選択されたときに、オペコードかアドレスのいずれかまたは両方を送信 FIFO に書き込むことで開始されます。オペコードおよびアドレスが EEPROM デバイスに送信され、その後、読み出しデータ

が EEPROM デバイスから受信され、受信 FIFO に格納されます。シリアル転送の終わりは、コントロールレジスタ 1（rSpi_CTRLR1）の bSpi_NDF でコントロールされます。

注　意

SPI コントローラがテキサスモードで設定される場合に EEPROM 読み出しモードは、サポートされません。

受信 FIFO しきい値レベル（rSpi_RXFTLR）は、受信 FIFO がフルに近いことを早めに表示するために使用することが可能です。DMA が APB アクセスに対して使用される場合、受信データレベル（bSpi_DMARDLR）は、DMA コントローラに早めに要求を出し、受信 FIFO がほとんどフルであることを示すために使用できます。

SPI マスタからのシリアル転送を完了させるための一般的なソフトウェアフローを以下に示します。

1. SPI コントローラが有効な場合、SPI イネーブル（bSpi_SSIENR）に 0 を書き込むことで、無効にします。
2. 転送用に SPI コントロールレジスタを設定します。これらのレジスタは任意の順序で設定できます。
   - コントロールレジスタ 0（rSpi_CTRLR0）に書き込みます。
     （モトローラ転送では、シリアルクロック極性パラメータおよびシリアルクロック位相パラメータは、ターゲットスレーブデバイスに対して同一に設定される必要があります）
   - 転送モードが受信のみモードの場合、コントロールレジスタ 1（rSpi_CTRLR1）を転送フレーム数マイナス 1 で書き込みます。たとえば、4 つのデータフレームを受信する場合は、このレジスタに 3 を書き込みます。
   - rSpi_BAUDR レジスタに書き込みます。
     転送のボーレートを設定します。
   - rSpi_TXFTLR レジスタおよび rSpi_RXFTLR レジスタに書き込みます。
     FIFO しきい値レベルを設定します。
   - rSpi_IMR レジスタに書き込みます。
     割り込みマスクを設定します。
   - rSpi_SER レジスタに書き込みます。
     スレーブイネーブルレジスタに書き込み、選択用のターゲットスレーブを有効にすることが可能です。スレーブがここで有効になると、転送は、1 つの有効データエントリが送信 FIFO に存在すると同時に開始します。データレジスタ（bSpi_DR）に書き込まれる際に有効なスレーブがない場合、転送はスレーブが有効になるまで開始しません。
3. bSpi_SSIENR に 1 を書き込むことで SPI コントローラを有効にします。
4. ターゲットスレーブへの送信用のデータを送信 FIFO（rSpi_DR）に書き込みます。
   - この時点で rSpi_SER レジスタに有効なスレーブがない場合は、転送を開始するために有効にしてください。
5. bSpi_BUSY ステータスをポーリングして、転送の完了を待ちます。bSpi_BUSY ステータスはすぐにポーリングできません。詳細については、「**2.6.1　プログラミング上の注意事項**」を参照してください。
   - 送信 FIFO エンプティ割り込み要求が発生した場合、送信 FIFO（rSpi_DR）に書き込んでください。
   - 受信 FIFO フル割り込み要求が発生した場合、受信 FIFO（rSpi_DR）を読み出してください。

6. 転送は、送信 FIFO がエンプティの場合、シフトコントロールロジックによって停止します。
   - 転送モードが受信のみモードの場合（bSpi_TMOD＝2’b10）、指定されたフレーム数が受信されたとき、シフトコントロールロジックによって転送は停止します。
   - 転送が完了すると、ビジー状態は 0 にリセットされます。
7. 転送モードが送信のみモードではない場合（bSpi_TMOD != 2’b01）、受信 FIFO がエンプティになるまで読み出してください。
8. bSpi_SSIENR に 0 を書き込むことで SPI コントローラを無効にします。

下図では、SPI マスタモトローラおよびテキサスモード転送の開始について一般的なソフトウェアフローを示します。図では、シリアルマスタコンポーネント内のハードウェアフローも示します。

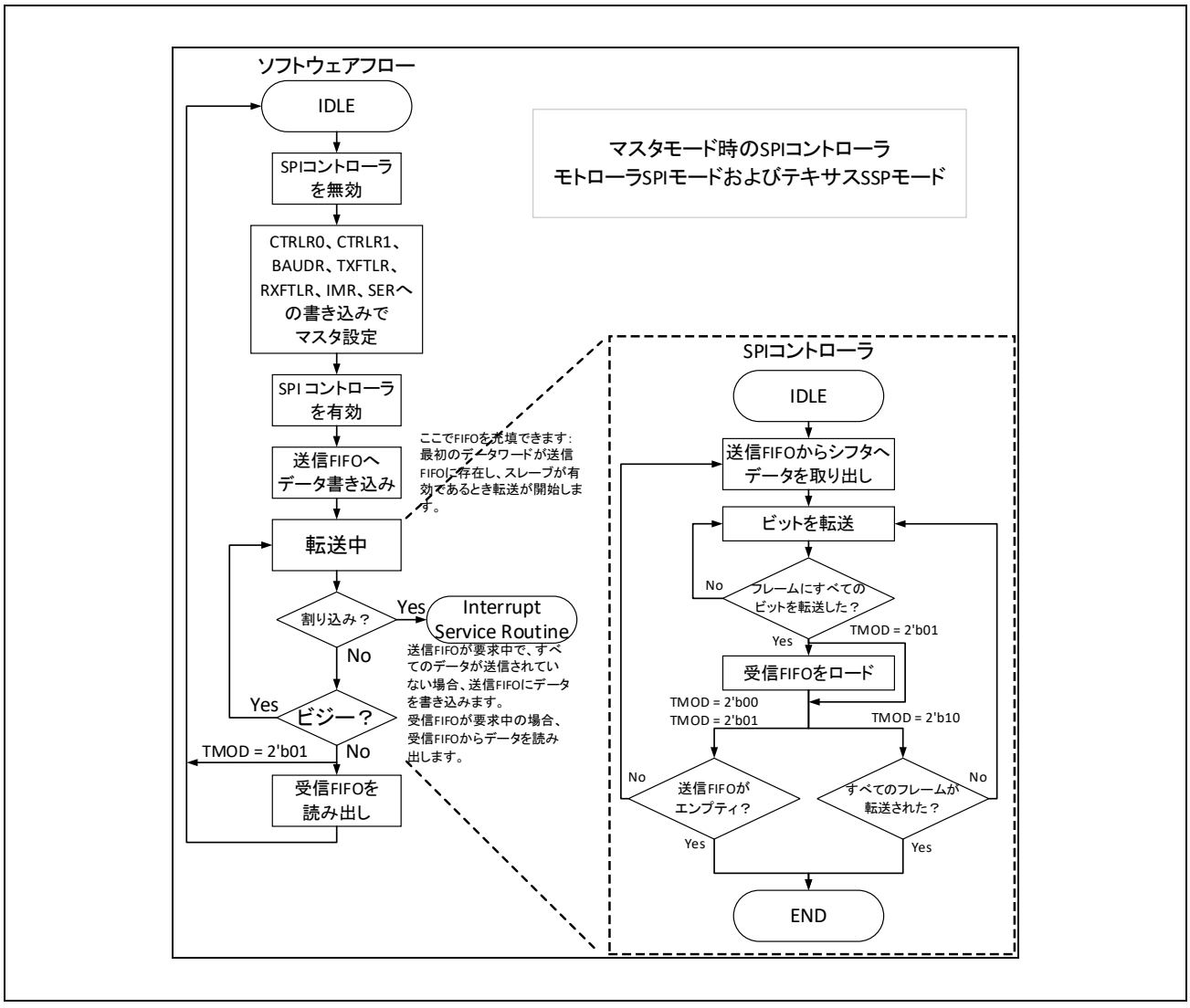


図 2.24　マスタモード時の SPI コントローラ、モトローラおよびテキサスモード

2.6.1.2　ナショナルセミコンダクターモードでのマスタ SPI のプログラミング

ナショナルセミコンダクターMicrowire の項では、Microwire シリアルプロトコルを詳しく説明しています。シリアル転送の前後で FIFO 送受信データが構造化される方法を説明しています。SPI マスタからの Microwire シリアル転送は、Microwire コントロールレジスタ（rSpi_MWCR）によって制御されます。bSpi_MWHS は Microwire ハンドシェイクインタフェースを有効または無効にします。bSpi_MDD はデータフレームの方向を制御します（コントロールフレームは常に、マスタによって送信され、スレーブによって受信されます）。bSpi_MWMOD は、転送がシーケンシャルか非シーケンシャルかを定義します。

すべての Microwire 転送は、送信 FIFO に少なくとも 1 つのコントロールワードがありスレーブが有効な場合に、SPI マスタによって開始されます。SPI マスタがデータフレームを送信するとき（bSpi_MDD＝1）、送信 FIFO がエンプティになるとシフトロジックによって転送が終了します。

SPI マスタがデータフレームを受信するとき（bSpi_MDD＝0）、転送の終了は、bSpi_MWMOD の設定によって異なります。転送は、非シーケンシャルの場合（bSpi_MWMOD＝0）、スレーブからのデータフレームをシフト後、送信 FIFO がエンプティになったときに終了します。転送は、シーケンシャルの場合（bSpi_MWMOD＝1）、受信したデータフレーム数が rSpi_CTRLR1 レジスタ＋1 に等しくなった場合に、シフトロジックによって終了します。

SPI マスタ上のハンドシェイクインタフェースが有効な場合（bSpi_MWHS＝1）、ターゲットスレーブのステータスが送信後ポーリングされます。スレーブがレディ状態を示した場合のみ、SPI マスタは転送を完了させ、ビジー状態をクリアします。転送が連続している場合、スレーブデバイスがレディ状態を返すまで、次のコントロール／データフレームは送信されません。

SPI マスタからの Microwire シリアル転送を完了させるための一般的なソフトウェアフローを以下に示します。

1. SPI が有効な場合、bSpi_SSIENR に 0 を書き込むことで、無効にします。
2. 転送用の SPI コントロールレジスタを設定します。これらのレジスタは任意の順序で設定できます。
   - 転送パラメータを設定するには rSpi_CTRLR0 に書き込みます。
     転送がシーケンシャルで、SPI マスタがデータを受信する場合、転送のフレーム数マイナス 1 で rSpi_CTRLR1 に書き込みます。たとえば、4 つのデータフレームを受信する場合は、このレジスタに 3 を書き込みます。
   - rSpi_BAUDR に書き込みます。
     転送のボーレートを設定します。
   - rSpi_TXFTLR レジスタおよび rSpi_RXFTLR レジスタに書き込みます
     FIFO しきい値レベルを設定します。
   - rSpi_IMR レジスタに書き込みます。
     割り込みマスクを設定します。
   - rSpi_SER レジスタに書き込みます。
     選択用のターゲットスレーブを有効にします。スレーブがここで有効になると、転送は、送信 FIFO に 1 つの有効データエントリが書き込まれると開始します。bSpi_DR レジスタに書き込まれる際に有効なスレーブがない場合、転送はスレーブが有効になるまで開始しません。
3. bSpi_SSIENR レジスタに 1 を書き込むことで SPI コントローラを有効にします。
4. SPI マスタがデータを送信する場合、コントロールワードとデータワードを送信 FIFO（rSpi_DR）に書き込みます。
   - SPI マスタがデータを受信する場合、コントロールワードを送信 FIFO に書き込みます。
   - この時点で rSpi_SER レジスタに有効なスレーブがない場合は、転送を開始するために有効にしてください。

5. bSpi_BUSY ステータスをポーリングして、転送の完了を待ちます。bSpi_BUSY ステータスはすぐにはポーリングできません。詳細については、「**2.6.1 プログラミング上の注意事項**」を参照してください。
   - 送信 FIFO エンプティ割り込み要求が発生した場合、送信 FIFO（rSpi_DR）に書き込んでください。
   - 受信 FIFO フル割り込み要求が発生した場合、受信 FIFO（rSpi_DR）を読み出してください。
6. 転送は、送信 FIFO がエンプティの場合、シフトコントロールロジックによって停止します。転送モードがシーケンシャルで SPI マスタがデータを受信する場合、指定されたデータフレーム数を受信したとき、シフトコントロールロジックによって転送は停止します。送信が行われたとき、ビジー状態は 0 にリセットされます。
7. SPI マスタがデータを受信する場合、受信 FIFO をエンプティになるまで読み出してください。
8. bSpi_SSIENR に 0 を書き込むことで SPI コントローラを無効にします。

下図では、SPI マスタナショナルセミコンダクター仕様の Microwire シリアル転送の開始について一般的なソフトウェアフローを示します。図では、シリアルマスタコンポーネント内のハードウェアフローも示します。

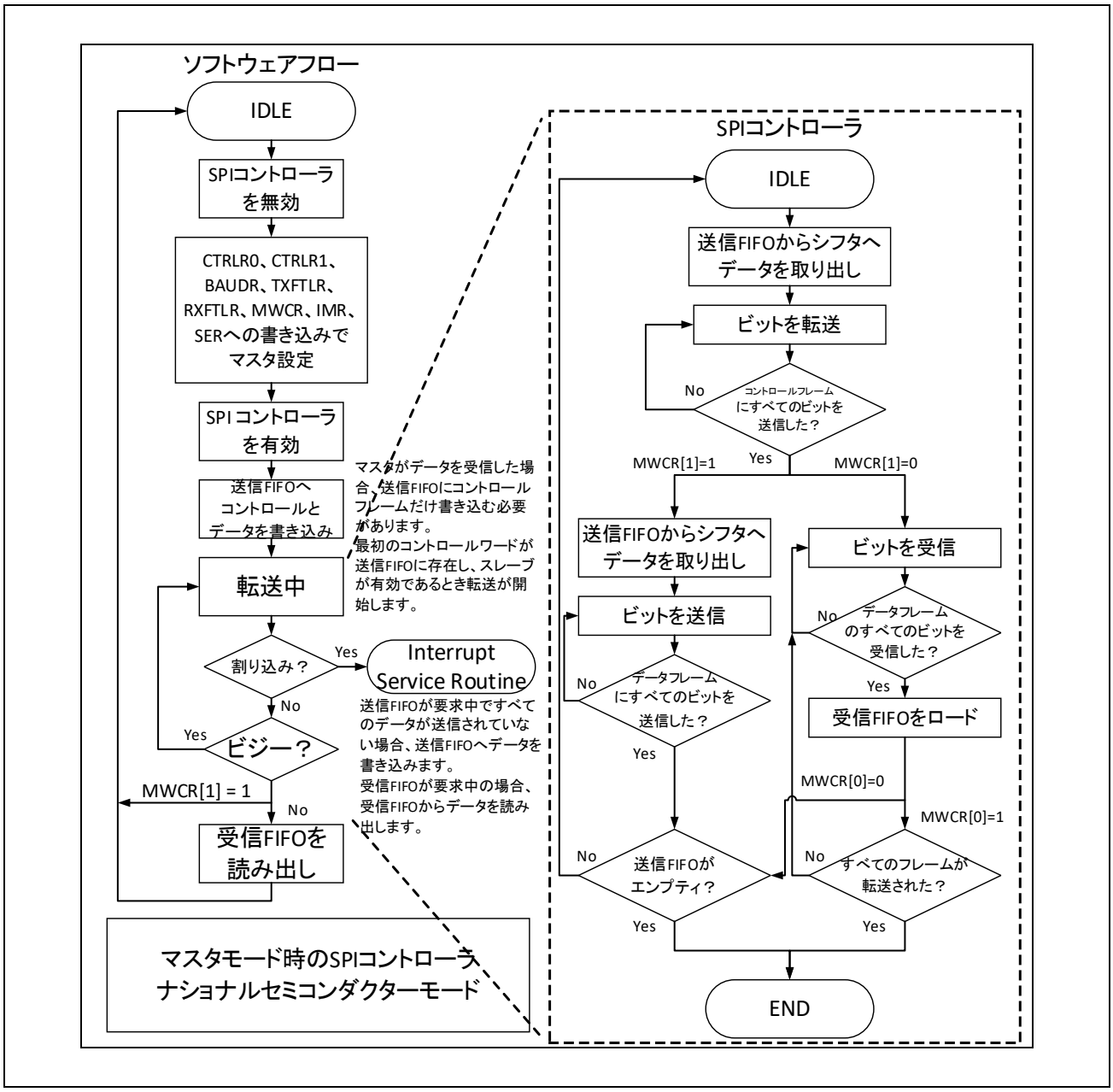


図 2.25　マスタモード時の SPI コントローラ、ナショナルセミコンダクターモード

2.6.1.3　モトローラおよびテキサスモードでのスレーブ SPI のプログラミング

モトローラシリアルペリフェラルインタフェースおよびテキサスインスツルメンツ同期式シリアルプロトコルの項では、それぞれのシリアルプロトコルについて説明しています。シリアル転送の前後で送受信の FIFO データが構造化される方法を説明しています。

SPI スレーブが受信のみモードの場合（bSpi_TMOD＝10）、送信シフトレジスタに現在あるデータが、スレーブデバイスが選択されるたびに再送信されるため、送信 FIFO に有効なデータを含む必要はありません。本モードを使用する場合は、送信 FIFO エンプティ割り込みをマスクする必要があります。

SPI スレーブがデータをマスタに送信する場合、シリアルマスタデバイスによって転送が開始される前に、送信 FIFO にデータが存在することを確認する必要があります。マスタが SPI スレーブへの転送を開始する場合、送信 FIFO にデータがないときに、エラーフラグ（bSpi_TXE）が SPI ステータスレジスタにセットされ、以前送信されたデータフレームが SPI_MISO で再送信されます。

連続データ転送では、すべてのデータが送信される前に、送信 FIFO バッファがエンプティにならないことを保証する必要があります。送信 FIFO しきい値レベルレジスタ（rSpi_TXFTLR）は、プロセッサに早めに割り込み（iSpi_TXE_Int）、送信 FIFO バッファがほとんどエンプティであることを示すために使用することが可能です。

DMA コントローラが APB アクセスに対して使用される場合、DMA 送信データレベルレジスタ（rSpi_DMATDLR）は、DMA コントローラに早めに要求を出し、送信 FIFO がほとんどエンプティであることを示すために使用できます。その後、FIFO は再び満たされたデータでシリアル転送を続行可能です。受信 FIFO バッファは、オーバーフローを防ぐために、受信 FIFO が FIFO フル割り込み要求を生成するたびに読み出される必要があります。受信 FIFO しきい値レベルレジスタ（rSpi_RXFTLR）は、受信 FIFO がフルに近いことを早めに表示するために使用することが可能です。

DMA コントローラが APB アクセスに対して使用される場合、DMA 受信データレベルレジスタ（rSpi_DMARDLR）は、DMA コントローラに早めに要求を出し、受信 FIFO がほとんどフルであることを示すために使用できます。

SPI スレーブへのシリアルマスタからの連続シリアル転送を完了させるための一般的なソフトウェアフローを以下に示します。

1. SPI コントローラが有効な場合、bSpi_SSIENR に 0 を書き込むことで、無効にします。
2. 転送用の SPI コントロールレジスタを設定します。これらのレジスタは任意の順序で設定できます。
   - rSpi_CTRLR0 に書き込みます。
     SPI 転送には、bSpi_SCPH および bSpi_SCPOL はマスタデバイスと同一になるように設定される必要があります。
   - rSpi_TXFTLR レジスタおよび rSpi_RXFTLR レジスタに書き込みます
     FIFO しきい値レベルを設定します。
   - rSpi_IMR レジスタに書き込みます。
     割り込みマスクを設定します。
3. bSpi_SSIENR レジスタに 1 を書き込むことで SPI コントローラを有効にします。
4. 転送モードが送受信モードの場合（bSpi_TMOD＝2’b00）または送信のみモードの場合（bSpi_TMOD＝2’b01）、マスタへの送信用のデータを送信 FIFO（rSpi_DR）に書き込みます。転送モードが受信のみモードの場合（bSpi_TMOD＝2’b10）、データを送信 FIFO に書き込む必要はなく、送信シフトレジスタの現在の値が再送信されます。
5. SPI スレーブはシリアル転送準備完了。シリアルマスタデバイスによって SPI スレーブが選択されると、転送が開始します。
6. 転送が途中のとき、bSpi_BUSY ステータスで転送ステータスを見るためにポーリングできます。

- 送信 FIFO エンプティ割り込み要求が発生した場合、送信 FIFO（rSpi_DR）に書き込んでください。
- 受信 FIFO フル割り込み要求が発生した場合、受信 FIFO（rSpi_DR）を読み出してください。

7. SPI スレーブへのシリアルマスタ選択入力を解除したときに、転送が終了します。
   - 送信が完了したとき、bSpi_BUSY 状態は 0 にリセットされます。
8. 転送モードが送信のみモードではない場合（bSpi_TMOD != 2’b01）、受信 FIFO がエンプティになるまで読み出します。
9. bSpi_SSIENR に 0 を書き込むことで SPI コントローラを無効にします。

下図では、SPI スレーブモトローラおよびテキサスモードシリアル転送について一般的なソフトウェアフローを示します。図では、シリアル-スレーブコンポーネント内のハードウェアフローも示します。

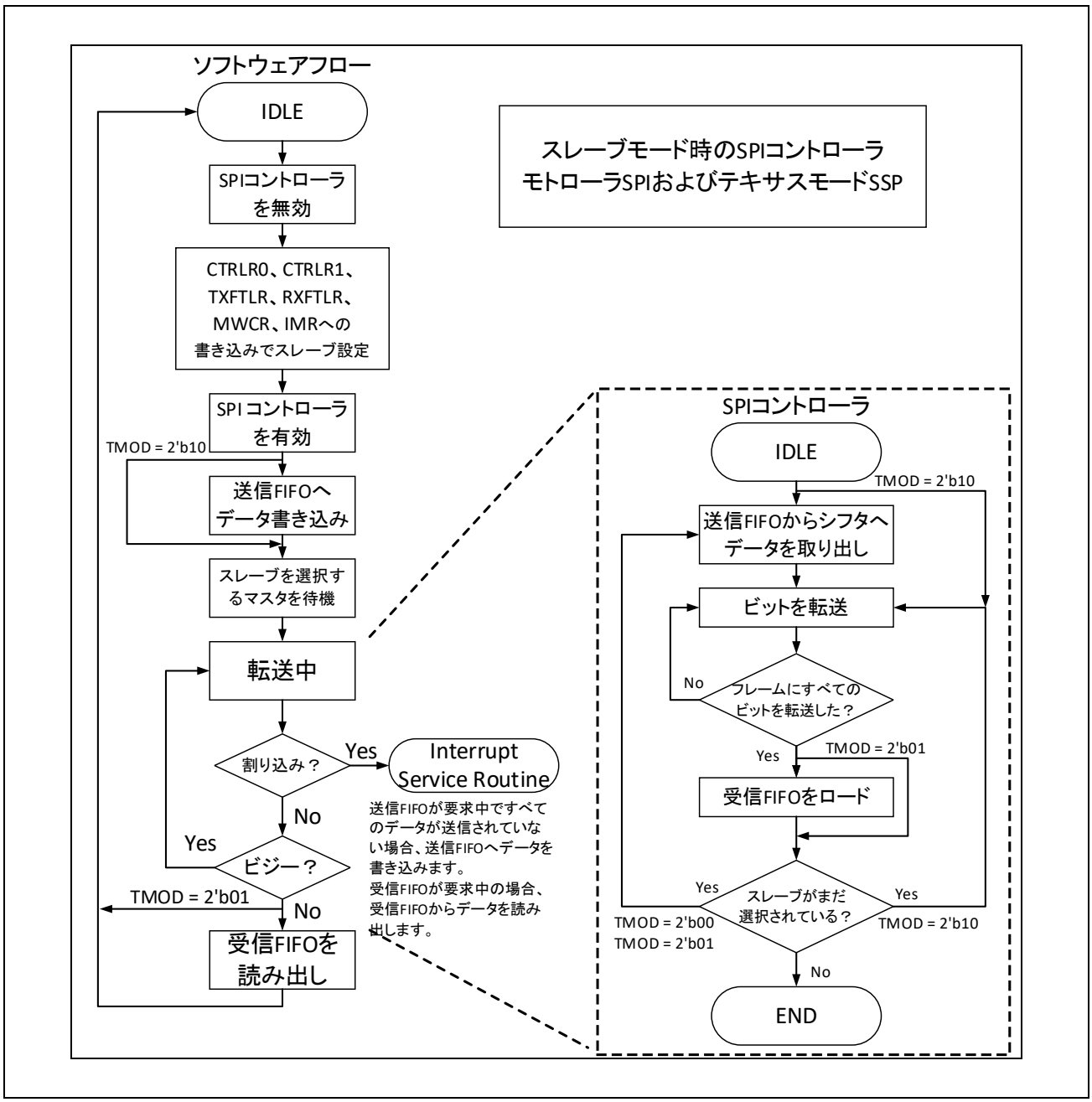


図 2.26　　スレーブモード時の SPI コントローラ、モトローラおよびテキサスモード

2.6.1.4　ナショナルセミコンダクターモードでのスレーブ SPI のプログラミング

ナショナルセミコンダクターMicrowire の項では、Microwire シリアルプロトコルを詳しく説明しています。タイミング図を含め、シリアル転送の前後で FIFO 送受信データが構造化される方法を説明しています。

SPI スレーブの場合、Microwire プロトコルは SPI プロトコルと同じ方法で動作します。SPI スレーブデバイスによるコントロールフレームのデコードはありません。

「**図 2.26 スレーブモード時の SPI コントローラ、モトローラおよびテキサスモード**」を参照してください。

第3章　I2C

3.1　概要

- 2 ユニット
- 2 種類のスピード：
  - 標準モード（0～100kb/s）
  - ファストモード（最大 400kb/s）
- 2 つの独立した FIFO：8×8 ビット送信 FIFO および 8×8 ビット受信 FIFO
- マスタまたはスレーブの I2C 動作
- 7 ビットまたは 10 ビットのアドレス形式
- 7 ビットまたは 10 ビットの結合フォーマットでの転送
- バルク送信モード
- 送受信バッファ
- 割り込みモードまたはポーリングモードの動作
- プログラマブル SDA ホールド時間（$t_{HD;DAT}$）
- シリアルリファレンスクロック
  - （最大 83.33MHz のプログラマブル周波数）
  - I2C1 および I2C2 は同じプログラマブル整数分周器を共有

両方の I2C モジュールのインタフェースおよび他のブロックへの接続を下図に示します。

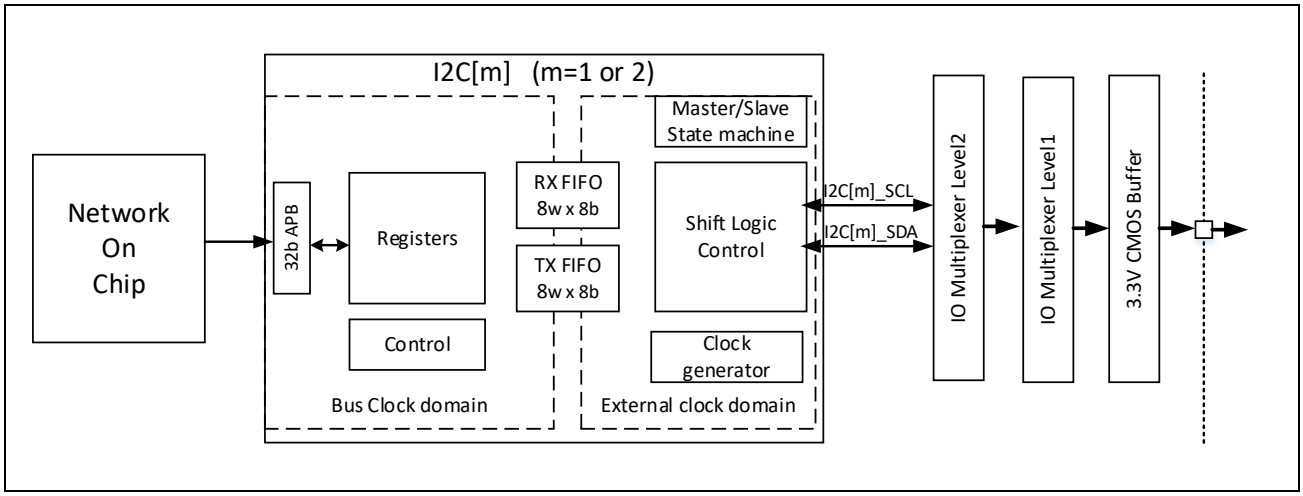


図 3.1　I2C インタフェースおよび接続

3.2　信号インタフェース

信号名	入出力	説明
クロック		
I2C[m]_PCLK	入力	内部バスクロック（APB）
I2C[m]_SCLK	入力	シリアルリファレンスクロック
割り込み		
I2C[m]_Int	出力	レベル検出割り込み出力、アクティブ High
外部信号		
I2C[m]_SCL	入出力	シリアルクロック（SCL）
I2C[m]_SDA	入出力	シリアルデータ（SDA）

備考　m＝1 または m＝2
本章では主にインデックスを省いたスタイルを使用します。
例）I2C_PCLK

3.3　レジスタマップ

3.3.1　I2C1 レジスタマップ

表 3.1　I2C1 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
4006 3000h	IC_CON	I2C コントロールレジスタ
4006 3004h	IC_TAR	I2C 対象アドレスレジスタ
4006 3008h	IC_SAR	I2C スレーブアドレスレジスタ
4006 3010h	IC_DATA_CMD	I2C Rx/Tx データバッファおよびコマンドレジスタ
4006 3014h	IC_SS_SCL_HCNT	標準モード I2C クロック SCL High カウントレジスタ
4006 3018h	IC_SS_SCL_LCNT	標準モード I2C クロック SCL Low カウントレジスタ
4006 301Ch	IC_FS_SCL_HCNT	ファストモード I2C クロック SCL High カウントレジスタ
4006 3020h	IC_FS_SCL_LCNT	ファストモード I2C クロック SCL Low カウントレジスタ
4006 302Ch	IC_INTR_STAT	I2C 割り込みステータスレジスタ
4006 3030h	IC_INTR_MASK	I2C 割り込みマスクレジスタ
4006 3034h	IC_RAW_INTR_STAT	I2C ロウ（raw）割り込みステータスレジスタ
4006 3038h	IC_RX_TL	I2C 受信 FIFO しきい値レジスタ
4006 303Ch	IC_TX_TL	I2C 送信 FIFO しきい値レジスタ
4006 3040h	IC_CLR_INTR	結合および個別の割り込みクリアレジスタ
4006 3044h	IC_CLR_RX_UNDER	RX_UNDER 割り込みクリアレジスタ
4006 3048h	IC_CLR_RX_OVER	RX_OVER 割り込みクリアレジスタ
4006 304Ch	IC_CLR_TX_OVER	TX_OVER 割り込みクリアレジスタ
4006 3050h	IC_CLR_RD_REQ	RD_REQ 割り込みクリアレジスタ
4006 3054h	IC_CLR_TX_ABRT	TX_ABRT 割り込みクリアレジスタ
4006 3058h	IC_CLR_RX_DONE	RX_DONE 割り込みクリアレジスタ
4006 305Ch	IC_CLR_ACTIVITY	ACTIVITY 割り込みクリアレジスタ
4006 3060h	IC_CLR_STOP_DET	STOP_DET 割り込みクリアレジスタ
4006 3064h	IC_CLR_START_DET	START_DET 割り込みクリアレジスタ
4006 3068h	IC_CLR_GEN_CALL	GEN_CALL 割り込みクリアレジスタ
4006 306Ch	IC_ENABLE	I2C イネーブルレジスタ
4006 3070h	IC_STATUS	I2C ステータスレジスタ
4006 3074h	IC_TXFLR	I2C 送信 FIFO レベルレジスタ
4006 3078h	IC_RXFLR	I2C 受信 FIFO レベルレジスタ
4006 307Ch	IC_SDA_HOLD	I2C SDA ホールド時間長レジスタ
4006 3080h	IC_TX_ABRT_SOURCE	I2C 送信アボート要因レジスタ
4006 3084h	IC_SLV_DATA_NACK_ONLY	スレーブデータ NACK 生成レジスタ
4006 3094h	IC_SDA_SETUP	I2C SDA セットアップレジスタ
4006 3098h	IC_ACK_GENERAL_CALL	I2C ACK ゼネラルコールレジスタ
4006 309Ch	IC_ENABLE_STATUS	I2C イネーブルステータスレジスタ
4006 30A0h	IC_FS_SPKLEN	I2C Sm、Fm スパイク抑止制限
4006 30A8h	IC_CLR_RESTART_DET	RESTART_DET 割り込みクリアレジスタ
4006 30F4h	IC_COMP_PARAM_1	コンポーネントパラメータレジスタ 1

3.3.2　I2C2 レジスタマップ

表 3.2　I2C2 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
4006 4000h	IC_CON	I2C コントロールレジスタ
4006 4004h	IC_TAR	I2C 対象アドレスレジスタ
4006 4008h	IC_SAR	I2C スレーブアドレスレジスタ
4006 4010h	IC_DATA_CMD	I2C Rx/Tx データバッファおよびコマンドレジスタ
4006 4014h	IC_SS_SCL_HCNT	標準モード I2C クロック SCL High カウントレジスタ
4006 4018h	IC_SS_SCL_LCNT	標準モード I2C クロック SCL Low カウントレジスタ
4006 401Ch	IC_FS_SCL_HCNT	ファストモード I2C クロック SCL High カウントレジスタ
4006 4020h	IC_FS_SCL_LCNT	ファストモード I2C クロック SCL Low カウントレジスタ
4006 402Ch	IC_INTR_STAT	I2C 割り込みステータスレジスタ
4006 4030h	IC_INTR_MASK	I2C 割り込みマスクレジスタ
4006 4034h	IC_RAW_INTR_STAT	I2C ロウ（raw）割り込みステータスレジスタ
4006 4038h	IC_RX_TL	I2C 受信 FIFO しきい値レジスタ
4006 403Ch	IC_TX_TL	I2C 送信 FIFO しきい値レジスタ
4006 4040h	IC_CLR_INTR	結合および個別の割り込みクリアレジスタ
4006 4044h	IC_CLR_RX_UNDER	RX_UNDER 割り込みクリアレジスタ
4006 4048h	IC_CLR_RX_OVER	RX_OVER 割り込みクリアレジスタ
4006 404Ch	IC_CLR_TX_OVER	TX_OVER 割り込みクリアレジスタ
4006 4050h	IC_CLR_RD_REQ	RD_REQ 割り込みクリアレジスタ
4006 4054h	IC_CLR_TX_ABRT	TX_ABRT 割り込みクリアレジスタ
4006 4058h	IC_CLR_RX_DONE	RX_DONE 割り込みクリアレジスタ
4006 405Ch	IC_CLR_ACTIVITY	ACTIVITY 割り込みクリアレジスタ
4006 4060h	IC_CLR_STOP_DET	STOP_DET 割り込みクリアレジスタ
4006 4064h	IC_CLR_START_DET	START_DET 割り込みクリアレジスタ
4006 4068h	IC_CLR_GEN_CALL	GEN_CALL 割り込みクリアレジスタ
4006 406Ch	IC_ENABLE	I2C イネーブルレジスタ
4006 4070h	IC_STATUS	I2C ステータスレジスタ
4006 4074h	IC_TXFLR	I2C 送信 FIFO レベルレジスタ
4006 4078h	IC_RXFLR	I2C 受信 FIFO レベルレジスタ
4006 407Ch	IC_SDA_HOLD	I2C SDA ホールド時間長レジスタ
4006 4080h	IC_TX_ABRT_SOURCE	I2C 送信アボート要因レジスタ
4006 4084h	IC_SLV_DATA_NACK_ONLY	スレーブデータ NACK 生成レジスタ
4006 4094h	IC_SDA_SETUP	I2C SDA セットアップレジスタ
4006 4098h	IC_ACK_GENERAL_CALL	I2C ACK ゼネラルコールレジスタ
4006 409Ch	IC_ENABLE_STATUS	I2C イネーブルステータスレジスタ
4006 40A0h	IC_FS_SPKLEN	I2C Sm、Fm スパイク抑止制限
4006 40A8h	IC_CLR_RESTART_DET	RESTART_DET 割り込みクリアレジスタ
4006 40F4h	IC_COMP_PARAM_1	コンポーネントパラメータレジスタ 1

3.4　レジスタの説明

3.4.1　IC_CON ― I2C コントロールレジスタ

アドレス　4006 3000h (I2C1)
4006 4000h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	RX_FIFO_FULL_HLD_CTRL	TX_EMPTY_CTRL	STOP_DET_IFADDRESSED	IC_SLAVE_DISABLE	IC_RESTART_EN	IC_10BITADDR_MASTER_rd_only	IC_10BITADDR_SLAVE	SPEED		MASTER_MODE
リセット後の値	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1

表 3.3　IC_CON レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b10	予約ビット		R
b9	RX_FIFO_FULL_HLD_CTRL	本ビットは、Rx FIFO がその RX_BUFFER_DEPTH（深さ＝8）に対して物理的にフルの場合、I2C コントローラがバスをホールドすべきかどうかを制御します。 0：バスのホールド無効 1：バスのホールド有効	R/W
b8	TX_EMPTY_CTRL	本ビットは、IC_RAW_INTR_STAT レジスタで説明されるように、TX_EMPTY 割り込みの生成を制御します。	R/W
b7	STOP_DET_IFADDRESSED	スレーブモード時： 1：アドレス指定されるときのみ STOP_DET 割り込みを発行します。 0：アドレス指定されるかどうかにかかわらず、STOP_DET を発行します。 **備考**　ゼネラルコールアドレス中、STOP_DET_IF_ADDRESSED＝1'b1 の場合、スレーブが ACK 生成によりゼネラルコールアドレスに応答する場合であっても、本スレーブは STOP_DET 割り込みを発行しません。送信アドレスがスレーブアドレス（SAR）に一致する場合のみ STOP_DET 割り込みが生成されます。	R/W
b6	IC_SLAVE_DISABLE	本ビットは I2C コントローラがスレーブを無効にするかどうかを制御します。 本ビットがセットされた場合（スレーブは無効）、I2C コントローラはマスタとしてのみ機能し、スレーブを必要とするアクションを実行しません。 0：スレーブは有効 1：スレーブは無効 リセット適用後、スレーブは無効になります。 **備考**　本ビットに 0 が書き込まれた場合、ソフトウェアは、必ず、ビット 0 にも 0 を書き込む必要があります。	R/W

表 3.3　IC_CON レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b5	IC_RESTART_EN	マスタとして動作時に RESTART コンディションが送信されるかどうかを決定します。一部の古いスレーブは、RESTART コンディションをサポートしませんが、いくつかの I2C コントローラの動作で RESTART コンディションが使用されます。 0：無効 1：有効 RESTART が無効の場合、マスタは以下の機能の実行を禁止されます。 ● 転送内での方向変更（スプリット） ● START BYTE の送信 ● 7 ビットアドレス形式モードでの結合フォーマットの転送 ● 10 ビットアドレスでの読み出し動作 ● 転送ごとの複数バイトの送信 STOP コンディションに続く RESTART コンディションと以降の START コンディションを置き換えることで、スプリット動作が複数の I2C コントローラ転送に分割されます。上記の動作が実行されると、IC_RAW_INTR_STAT レジスタのビット 6（TX_ABRT）がセットされます。	R/W
b4	IC_10BITADDR_MASTER_rd_only	ビットのアドレス形式モードを示す読み出し専用ビット：本ビットの機能は IC_TAR レジスタのビット 12 によって処理されます。 0：7 ビットのアドレス形式 1：10 ビットのアドレス形式	R
b3	IC_10BITADDR_SLAVE	スレーブとして動作するとき、本ビットは、I2C コントローラが 7 ビットアドレスまたは 10 ビットアドレスのどちらに対応するかを制御します。 0：7 ビットのアドレス形式。I2C コントローラは、10 ビットアドレス形式を含むトランザクションを無視します。IC_SAR レジスタの下位 7 ビットのみが比較されます。 1：10 ビットのアドレス形式。I2C コントローラは、IC_SAR レジスタの 10 ビットすべてに一致する 10 ビットアドレス形式転送のみに対応します。	R/W
b2、b1	SPEED	本ビットは I2C コントローラが動作するスピードを制御します。その設定は、I2C コントローラのマスタモードでの動作時のみ関係します。ハードウェアは、ソフトウェアでプログラミングされた不正な値から保護します。 1：標準モード（≦100kb/s） 2：ファストモード（≦400kb/s）	R/W
b0	MASTER_MODE	本ビットは I2C マスタコントローラが有効かどうかを制御します。 0：マスタは無効 1：マスタは有効 **備考**　本ビットに“1”が書き込まれた場合、ソフトウェアは、必ず、ビット 6 にも“1”を書き込む必要があります。	R/W

3.4.2　IC_TAR — I2C 対象アドレスレジスタ

以下の条件の任意の組み合わせが真の場合にかぎり、すべてのビットを動的に更新することが可能です。

I2C コントローラが有効ではない（IC_ENABLE[0]が 0 にセット）
　または
（I2C コントローラは有効である（IC_ENABLE[0]＝1）
　かつ
I2C コントローラは任意のマスタ（tx、rx）動作にかかわっていない（IC_STATUS[5]＝0）
　かつ
I2C コントローラはマスタモードでの動作が有効にされている（IC_CON[0]＝1）
　かつ
TX FIFO にエントリがない（IC_STATUS[2]＝1））

アドレス　4006 3004h (I2C1)
　　4006 4004h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	IC_10BIT ADDR_M ASTER	SPECIA L	GC_OR _STAR T	IC_TAR									
リセット後の値	X	X	X	1	0	0	0	0	0	1	0	1	0	1	0	1

表 3.4　IC_TAR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b13	予約ビット		R
b12	IC_10BITADDR_MASTER	本ビットは、マスタとしての動作時に、I2C コントローラが 7 ビットアドレス形式モードまたは 10 ビットアドレス形式モードのどちらで転送を開始するかを制御します。 0：7 ビットのアドレス形式 1：10 ビットのアドレス形式	R/W
b11	SPECIAL	本ビットは、ソフトウェアがゼネラルコールまたは START BYTE コマンドを実行するかどうかを指定します。 0：ビット 10 の GC_OR_START を無視し、IC_TAR を通常使用 1：GC_OR_START ビットで指定された特別な I2C コマンドを実行	R/W
b10	GC_OR_START	ビット 11（SPECIAL）が 1 にセットされている場合、本ビットは、ゼネラルコールまたは START Byte コマンドが I2C コントローラによって実行されるかどうかを指定します。 0：ゼネラルコール発行後のゼネラルコールアドレス、書き込みのみ実行できます。読み出しコマンドを発行しようとすると、IC_RAW_INTR_STAT レジスタのビット 6（TX_ABRT）がセットされます。I2C コントローラは、SPECIAL ビット値（ビット 11）がクリアされるまで、ゼネラルコールモードで保持されます。 1：START BYTE	R/W
b9～b0	IC_TAR	これは、任意のマスタトランザクションのターゲットアドレスです。ゼネラルコール送信時、これらのビットは無視されます。START BYTE を生成するには、CPU はこれらのビットに一度のみ書き込む必要があります。 IC_TAR および IC_SAR が同一の場合、ループバックになりますが、FIFO はマスタおよびスレーブの間で共有されるので完全なループバックは可能ではありません。1 方向のループバックモードのみサポートされ（単信）、二重化されません。マスタはマスタ自身に送信できません。スレーブへのみ送信できます。	R/W

3.4.3　IC_SAR ― I2C スレーブアドレスレジスタ

アドレス　4006 3008h (I2C1)
4006 4008h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	IC_SAR									
リセット後の値	X	X	X	X	X	X	0	0	0	1	0	1	0	1	0	1

表 3.5　IC_SAR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b10	予約ビット		R
b9～b0	IC_SAR	I2C コントローラがスレーブとして動作するときに、IC_SAR はスレーブアドレスを保持します。7 ビットアドレスでは IC_SAR[6:0]のみが使用されます。 本レジスタは、I2C インタフェースが無効（IC_ENABLE[0]=0）の場合のみ書き込み可能です。他の場合の書き込みは無効になります。 **備考）** 予約アドレス（0x00～0x07、または 0x78～0x7f）はいずれも禁止されます。IC_SAR または IC_TAR を予約値にプログラムした場合は、デバイスの正しい動作は保証されません。	R/W

3.4.4　IC_DATA_CMD ― I2C Rx/Tx データバッファおよびコマンドレジスタ

これは、TX FIFO を満たすときに CPU が書き込み、RX FIFO からバイトの取得時に CPU が読み出すレジスタです。

アドレス　4006 3010h (I2C1)
4006 4010h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	RESTART	STOP	CMD	DAT							
リセット後の値	X	X	X	X	X	0	0	0	0	0	0	0	0	0	0	0

表 3.6　IC_DATA_CMD レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b11	予約ビット		R
b10	RESTART	本ビットはバイトが送信または受信される前に RESTART が発行されるかどうかを制御します。 1：IC_RESTART_EN が 1 の場合、データが送信／受信（CMD の値に従う）される前に RESTART が発行されます。転送方向が以前のコマンドから変化したかしないかにかかわらず、IC_RESTART_EN が 0 の場合、代わりに START が続く STOP が発行されます。 0：IC_RESTART_EN が 1 の場合、転送方向が以前のコマンドから変化した場合のみ、RESTART が発行されます。IC_RESTART_EN が 0 の場合、代わりに START が続く STOP が発行されます。	W
b9	STOP	本ビットはバイトが送信または受信された後に STOP が発行されるかどうかを制御します。 1：Tx FIFO がエンプティかどうかにかかわらず、本バイトの後に STOP が発行されます。Tx FIFO がエンプティではない場合、マスタは、START を発行し、バスのアービトレーションを行って、ただちに新しい転送を開始しようとします。 0：Tx FIFO がエンプティかどうかにかかわらず、本バイトの後に STOP が発行されません。Tx FIFO がエンプティではない場合、マスタは、CMD ビットの値に従ってデータバイトを送信／受信することで現在の転送を続行します。Tx FIFO がエンプティの場合、マスタは SCL ラインを Low ホールドし、新しいコマンドが Tx FIFO で有効になるまでバスをストールさせます。	W

表 3.6　IC_DATA_CMD レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b8	CMD	本ビットは読み出しまたは書き込みが実行されるかどうかを制御します。 本ビットは、I2C コントローラがスレーブとして動作するときの方向を制御しません。マスタとして動作するときの方向のみ制御します。 1：読み出し 0：書き込み コマンドが TX FIFO に入力される場合、本ビットは書き込みコマンドと読み出しコマンドを区別します。スレーブレシーバモードでは、本レジスタへの書き込みは必要ではないため、本ビットは“don't care”です。スレーブトランスミッタモードでは、“0”は DAT または IC_DATA_CMD[7:0]として CPU データが送信されることを示します。 ● **本ビットのプログラム時は次のことに注意してください：** ゼネラルコールコマンドが送信された後に読み出し動作を実行しようとすると、IC_TAR レジスタのビット 11（SPECIAL）がクリアされないかぎり、TX_ABRT 割り込み（IC_RAW_INTR_STAT レジスタのビット 6）が発生します。 RD_REQ 割り込みの受信後、本ビットに“1”が書き込まれた場合、TX_ABRT 割り込みが発生します。 **備考）** マスタ I2C 読み出し転送が I2C コントローラ上で行われようとする間、I2C コントローラをアドレス指定するリモート I2C マスタが原因で RD_REQ 割り込みが同時に発生する可能性があります。このような場合、I2C コントローラは IC_DATA_CMD 書き込みを無視し、TX_ABRT 割り込みを生成して、RD_REQ 割り込み処理を待ちます。	W
b7～b0	DAT	本レジスタは、I2C バス上の送信データ、または受信データを含みます。 本レジスタに書き込みをし、読み出しを実行しようとすると、ビット[7:0]（DAT）は I2C コントローラによって無視されます。ただし、本レジスタを読み出すときに、これらのビットは I2C コントローラインタフェース上で受信したデータの値を返します。	R/W

3.4.5　IC_SS_SCL_HCNT ― 標準モード I2C クロック SCL High カウントレジスタ

アドレス　4006 3014h (I2C1)
4006 4014h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	IC_SS_SCL_HCNT															
リセット後の値	0	0	0	0	0	0	0	1	0	1	0	0	1	1	1	0

表 3.7　IC_SS_SCL_HCNT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット		R
b15～b0	IC_SS_SCL_HCNT	本レジスタは、適切な入出力タイミングを確保するために任意の I2C バストランザクションが行われる前に設定される必要があります。本レジスタは標準モードの SCL クロックが High である期間のカウント数をセットします。 本レジスタは、I2C インタフェースが無効の場合（IC_ENABLE[0]＝0）のみ書き込み可能です。他の場合の書き込みは無効になります。 有効な最小値は 6 です。ハードウェアはこの値未満の値の書き込みを防ぎ、書き込まれた場合は、6 がセットされることになります。 **備考）** 本レジスタは、65525 を超える値の設定ができません。I2C コントローラは、本カウンタが IC_SS_SCL_HCNT＋10 の値に達するときに I2C バスのアイドルコンディションにフラグを立てるために 16 ビットカウンタを使用します。	R/W

3.4.6　IC_SS_SCL_LCNT ― 標準モード I2C クロック SCL Low カウントレジスタ

アドレス　4006 3018h (I2C1)
4006 4018h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	IC_SS_SCL_LCNT															
リセット後の値	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0

表 3.8　IC_SS_SCL_LCNT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット		R
b15～b0	IC_SS_SCL_LCNT	本レジスタは、適切な入出力タイミングを確保するために任意の I2C バストランザクションが行われる前に設定される必要があります。本レジスタは標準モードの SCL クロックが Low である期間のカウント数をセットします。 本レジスタは、I2C インタフェースが無効の場合（IC_ENABLE[0]＝0）のみ書き込み可能です。他の場合の書き込みは無効になります。 有効な最小値は 8 です。ハードウェアはこの値未満の値の書き込みを防ぎ、書き込まれた場合は、8 がセットされることになります。	R/W

3.4.7　IC_FS_SCL_HCNT ― ファストモード I2C クロック SCL High カウントレジスタ

アドレス　4006 301Ch (I2C1)
4006 401Ch (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	IC_FS_SCL_HCNT															
リセット後の値	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0

表 3.9　IC_FS_SCL_HCNT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット		R
b15～b0	IC_FS_SCL_HCNT	本レジスタは、適切な入出力タイミングを確保するために任意の I2C バストランザクションが行われる前に設定される必要があります。本レジスタはファストモードの SCL クロックが High である期間のカウント数をセットします。 本レジスタは、I2C インタフェースが無効の場合（IC_ENABLE[0]＝0）のみ書き込み可能です。他の場合の書き込みは無効になります。 有効な最小値は 6 です。ハードウェアはこの値未満の値の書き込みを防ぎ、書き込まれた場合は、6 がセットされることになります。	R/W

3.4.8　IC_FS_SCL_LCNT ― ファストモード I2C クロック SCL Low カウントレジスタ

アドレス　4006 3020h (I2C1)
4006 4020h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	IC_FS_SCL_LCNT															
リセット後の値	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	1

表 3.10　IC_FS_SCL_LCNT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット		R
b15～b0	IC_FS_SCL_LCNT	本レジスタは、適切な入出力タイミングを確保するために任意の I2C バストランザクションが行われる前に設定される必要があります。本レジスタはファストモードの SCL クロックが Low である期間のカウント数をセットします。 本レジスタは、I2C インタフェースが無効の場合（IC_ENABLE[0]＝0）のみ書き込み可能です。他の場合の書き込みは無効になります。 有効な最小値は 8 です。ハードウェアはこの値未満の値の書き込みを防ぎ、書き込まれた場合は、8 がセットされることになります。	R/W

3.4.9　IC_INTR_STAT ― I2C 割り込みステータスレジスタ

本レジスタの各ビットには、IC_INTR_MASK レジスタに対応するマスクビットがあります。これらのビットは、一致する割り込みクリアレジスタの読み出しでクリアされます。これらのビットのマスクされていない raw ステータスは IC_RAW_INTR_STAT レジスタで利用可能です。

アドレス　4006 302Ch (I2C1)
4006 402Ch (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	R_MASTER_ON_HOLD	R_RESTART_DET	R_GEN_CALL	R_START_DET	R_STOP_DET	R_ACTIVITY	R_RX_DONE	R_TX_ABRT	R_RD_REQ	R_TX_EMPTY	R_TX_OVER	R_RX_FULL	R_RX_OVER	R_RX_UNDER
リセット後の値	X	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 3.11　IC_INTR_STAT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット	これらのビットのステータスは、下記レジスタを参照ください。 「**3.4.11 IC_RAW_INTR_STAT ― I2C ロウ（raw）割り込みステータスレジスタ**」	R
b13	R_MASTER_ON_HOLD		R
b12	R_RESTART_DET		R
b11	R_GEN_CALL		R
b10	R_START_DET		R
b9	R_STOP_DET		R
b8	R_ACTIVITY		R
b7	R_RX_DONE		R
b6	R_TX_ABRT		R
b5	R_RD_REQ		R
b4	R_TX_EMPTY		R
b3	R_TX_OVER		R
b2	R_RX_FULL		R
b1	R_RX_OVER		R
b0	R_RX_UNDER		R

3.4.10　IC_INTR_MASK ― I2C 割り込みマスクレジスタ

これらのビットは、対応する割り込みステータスビットをマスクします。0 の値はビットが割り込みを生成しないようにします。

アドレス　4006 3030h (I2C1)
4006 4030h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	M_MASTER_ON_HOLD	M_RESTART_DET	M_GEN_CALL	M_START_DET	M_STOP_DET	M_ACTIVITY	M_RX_DONE	M_TX_ABRT	M_RD_REQ	M_TX_EMPTY	M_TX_OVER	M_RX_FULL	M_RX_OVER	M_RX_UNDER
リセット後の値	X	X	0	0	1	0	0	0	1	1	1	1	1	1	1	1

表 3.12　IC_INTR_MASK レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット		R
b13	M_MASTER_ON_HOLD	IC_INTR_STAT レジスタの R_MASTER_ON_HOLD 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b12	M_RESTART_DET	IC_INTR_STAT レジスタの R_RESTART_DET 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b11	M_GEN_CALL	IC_INTR_STAT レジスタの R_GEN_CALL 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b10	M_START_DET	IC_INTR_STAT レジスタの R_START_DET 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b9	M_STOP_DET	IC_INTR_STAT レジスタの R_STOP_DET 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b8	M_ACTIVITY	IC_INTR_STAT レジスタの R_ACTIVITY 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b7	M_RX_DONE	IC_INTR_STAT レジスタの R_RX_DONE 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b6	M_TX_ABRT	IC_INTR_STAT レジスタの R_TX_ABRT 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b5	M_RD_REQ	IC_INTR_STAT レジスタの R_RD_REQ 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b4	M_TX_EMPTY	IC_INTR_STAT レジスタの R_TX_EMPTY 割り込のためのをマスクビット 0：割り込み禁止 1：割り込み許可	R/W

表 3.12　IC_INTR_MASK レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3	M_TX_OVER	IC_INTR_STAT レジスタの R_TX_OVER 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b2	M_RX_FULL	IC_INTR_STAT レジスタの R_RX_FULL 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b1	M_RX_OVER	IC_INTR_STAT レジスタの R_RX_OVER 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W
b0	M_RX_UNDER	IC_INTR_STAT レジスタの R_RX_UNDER 割り込みのためのマスクビット 0：割り込み禁止 1：割り込み許可	R/W

3.4.11　IC_RAW_INTR_STAT ― I2C ロウ（raw）割り込みステータスレジスタ

IC_INTR_STAT レジスタとは異なり、これらのビットはマスクされる前の値を示します。

アドレス　4006 3034h (I2C1)
4006 4034h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	MASTER_ON_HOLD	RESTART_DET	GEN_CALL	START_DET	STOP_DET	RAW_INTR_ACTIVITY	RX_DONE	TX_ABRT	RD_REQ	TX_EMPTY	TX_OVER	RX_FULL	RX_OVER	RX_UNDER
リセット後の値	X	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 3.13　IC_RAW_INTR_STAT レジスタの内容（1/3）

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット		R
b13	MASTER_ON_HOLD	マスタがバスをホールドし、かつ TX FIFO がエンプティ状態の時に、本ビットは 1 にセットされます。	R
b12	RESTART_DET	I2C コントローラがスレーブモードで動作中で、スレーブがアドレス指定されているときに、RESTART コンディションが I2C インタフェース上で発生したとき、本ビットは 1 にセットされます。 **備考）** 次の例外では、RESTART_DET 割り込みが生成されません。START Byte 転送で、I2C プロトコル定義のフォーマットに従い RESTART がアドレスフィールドの前に来る場合、スレーブはアドレス指定されていないため、RESTART_DET 割り込みを生成しません。	R
b11	GEN_CALL	ゼネラルコールアドレスを受信し、ACK 応答されるときのみセットされます。I2C コントローラを無効にするか、IC_CLR_GEN_CALL レジスタのビット 0 を CPU が読み出してクリアするまで、セットされたままです。I2C コントローラは Rx バッファに受信データを格納します。	R
b10	START_DET	I2C コントローラがスレーブモードまたはマスタモードで動作しているかにかかわらず、START コンディションまたは RESTART コンディションが I2C インタフェース上で発生したとき、本ビットは 1 にセットされます。	R
b9	STOP_DET	STOP_DET 割り込みステータスの動作は、IC_CON レジスタの STOP_DET_IFADDRESSED の選択によって異なります。 STOP_DET_IFADDRESSED＝0 の場合 I2C コントローラがスレーブモードまたはマスタモードで動作しているかにかかわらず、STOP コンディションが I2C インタフェース上で発生したとき、本ビットは 1 にセットされます。 スレーブモードでは、スレーブがアドレス指定されているかどうかにかかわらず、1 にセットされます。 STOP_DET_IFADDRESSED＝1 の場合 マスタモード（MASTER_MODE＝1'b1）では、STOP コンディションが I2C インタフェース上で発生したとき、本ビットは 1 にセットされます。 スレーブモード（MASTER_MODE＝1'b0）では、スレーブがアドレス指定される場合のみ STOP_DET 割り込みが生成されます。 **備考）**ゼネラルコールアドレス中、STOP_DET_IFADDRESSED＝1'b1 の場合、スレーブが ACK を生成してゼネラルコールアドレスに応答する場合であっても、本スレーブは STOP_DET 割り込みを発生しません。送信アドレスがスレーブアドレス（SAR）に一致する場合のみ STOP_DET 割り込みが生成されます。	R

表 3.13　IC_RAW_INTR_STAT レジスタの内容（2/3）

ビット位置	ビット名	機能	R/W
b8	RAW_INTR_ACTIVITY	本ビットは、I2C コントローラアクティビティのキャプチャで 1 にセットされ、クリアされるまで値を保持します。クリアするには次の 4 つの方法があります。 ● I2C コントローラの無効化 ● IC_CLR_ACTIVITY レジスタの読み出し ● IC_CLR_INTR レジスタの読み出し ● システムリセット 本ビットがセットされると、4 つの方法のいずれかによりクリアされるまでセットされたままです。I2C コントローラがアイドルの場合でも、本ビットはクリアされるまでセットされたままで、バス上にアクティビティが存在したことを示します。	R
b7	RX_DONE	I2C コントローラがスレーブトランスミッタとして動作する場合、マスタが送信されたバイトを確認しない場合に本ビットは 1 に設定されます。これは、送信の最終バイトで発生し、送信が完了したことを示します。	R
b6	TX_ABRT	本ビットは、I2C コントローラが I2C トランスミッタとして、送信 FIFO の内容で意図した動作を完了できないとき、1 をセットします。本状況は、I2C マスタまたは I2C スレーブの両方のケースで発生する可能性があり、“送信アボート”と呼ばれます。 本ビットが 1 にセットされた場合、IC_TX_ABRT_SOURCE レジスタは送信アボートが発生した理由を示します。 **備考）** I2C コントローラは、IC_TX_ABRT_SOURCE レジスタによって追跡される任意のイベントによって送信アボートが発生するときはいつでも、TX_FIFO のみをフラッシュ／リセット／エンプティにします。TX FIFO は、IC_CLR_TX_ABRT レジスタが読み出されるまで、このフラッシュされた状態を維持します。本読み出しが実行されることで、Tx FIFO はデータを APB インタフェースから受け入れることができます。Tx FIFO および Rx FIFO は送信アボートでフラッシュされます。	R
b5	RD_REQ	I2C コントローラがスレーブとして動作中に他の I2C マスタがデータを読み出そうとしている場合に、本ビットは 1 にセットされます。I2C コントローラは、本割り込みが処理されるまで、I2C バスを待機状態（SCL＝0）で保持します。これは、データ転送するよう問い合わせているリモートマスタによってスレーブが既にアドレス指定されていることを示します。プロセッサは本割り込みに応答する必要があり、その後、リクエストされたデータを IC_DATA_CMD レジスタに書き込みます。プロセッサが IC_CLR_RD_REQ レジスタを読み出した直後に、本ビットは 0 にセットされます。	R
b4	TX_EMPTY	TX_EMPTY 割り込みステータスの動作は、IC_CON レジスタの TX_EMPTY_CTRL の選択によって異なります。 TX_EMPTY_CTRL＝0 の場合： 送信バッファが IC_TX_TL レジスタでセットされたしきい値以下の場合に本ビットは 1 にセットされます。 TX_EMPTY_CTRL＝1 の場合： 送信バッファが IC_TX_TL レジスタでセットされたしきい値以下で、最近ポップされたコマンドのアドレス／データ送信が完了した場合に本ビットは 1 にセットされます。 バッファレベルがしきい値を超えると、ハードウェアによって自動的にクリアされます。IC_ENABLE[0]が 0 にセットされている場合、TX FIFO はフラッシュされ、リセット状態が保持されます。TX FIFO は内部にデータを持たないように見えており、マスタまたはスレーブのステートマシンがアクティブな場合は、本ビットは 1 にセットされます。アクティブでない場合は、IC_ENABLE_STATUS.IC_EN＝0 になり、本ビットは 0 になります。	R
b3	TX_OVER	送信バッファが 8 の深さまで満たされた状態で送信中、プロセッサが IC_DATA_CMD レジスタへの書き込みによって別の I2C コマンドを発行しようとしたときにセットされます。モジュールが無効になると、本ビットはマスタまたはスレーブのステートマシンがアイドルになるまでは値を維持し、IC_ENABLE_STATUS.IC_EN が 0 になってから、本割り込みがクリアされます。	R

表 3.13　IC_RAW_INTR_STAT レジスタの内容（3/3）

ビット位置	ビット名	機能	R/W
b2	RX_FULL	受信バッファが IC_RX_TL レジスタの RX_TL しきい値以上の場合にセットされます。バッファレベルがしきい値を下回ると、ハードウェアによって自動的にクリアされます。モジュールが無効な場合（IC_ENABLE[0]＝0）、RX FIFO はフラッシュされ、リセット状態が保持されます。このとき RX FIFO はフルではありません。本ビットは、アクティブかどうかにかかわらず、IC_ENABLE ビット 0 が 0 に設定されると、クリアされます。	R
b1	RX_OVER	受信バッファが深さ 8 までフルに満たされた状態で、追加データが外部 I2C デバイスから受信された場合にセットされます。I2C コントローラはこのことを認識しますが、FIFO がフルになった後に受信したデータバイトは失われます。モジュールを無効にすると（IC_ENABLE[0]＝0）、本ビットはマスタまたはスレーブのステートマシンがアイドルになるまでは値を維持します。IC_ENABLE_STATUS.IC_EN が 0 になると、本割り込みがクリアされます。 **備考）** IC_CON[9]ビット（RX_FIFO_FULL_HLD_CTRL）が High に設定されている場合、Rx FIFO はオーバーフローになることはないため、RX_OVER 割り込みは決して 1 にセットされません。	R
b0	RX_UNDER	IC_DATA_CMD レジスタから受信バッファがエンプティであるときにプロセッサが受信バッファを読み出そうとすると、1 にセットされます。モジュールを無効にすると（IC_ENABLE[0]＝0）、本ビットはマスタまたはスレーブのステートマシンがアイドルになるまでは値を維持します。IC_ENABLE_STATUS.IC_EN が 0 になると、本割り込みがクリアされます。	R

3.4.12　IC_RX_TL ― I2C 受信 FIFO しきい値レジスタ

アドレス　4006 3038h (I2C1)
4006 4038h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	RX_TL							
リセット後の値	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

表 3.14　IC_RX_TL レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	RX_TL	受信 FIFO しきい値レベル RX_FULL 割り込み（IC_RAW_INTR_STAT レジスタのビット 2）をトリガするエントリのレベル（下限）を制御します。有効な範囲は 0～255 ですが、バッファの容量より大きい値に設定しようとすると、実際の値はバッファの最大容量に設定されます。 0 の値は、1 エントリのしきい値を設定し、255 の値は、256 エントリのしきい値を設定します。	R/W

3.4.13　IC_TX_TL ― I2C 送信 FIFO しきい値レジスタ

アドレス　4006 303Ch (I2C1)
4006 403Ch (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	TX_TL							
リセット後の値	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

表 3.15　IC_TX_TL レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	TX_TL	送信 FIFO しきい値レベル TX_EMPTY 割り込み（IC_RAW_INTR_STAT レジスタのビット 4）をトリガするエントリのレベル（上限）を制御します。有効な範囲は 0～255 であり、バッファの容量より大きい値に設定しようとすると、実際の値はバッファの最大容量に設定されます。 0 の値は、0 エントリのしきい値を設定し、255 の値は、255 エントリのしきい値を設定します。	R/W

3.4.14　IC_CLR_INTR ― 結合および個別の割り込みクリアレジスタ

アドレス　4006 3040h (I2C1)
4006 4040h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_INTR
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.16　IC_CLR_INTR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_INTR	本レジスタを読み出すことで、統合された割り込み、すべての個々の割り込み、および IC_TX_ABRT_SOURCE レジスタをクリアします。本ビットはハードウェアでクリア可能な割り込みをクリアしませんが、ソフトウェアでクリア可能な割り込みをクリアします。IC_TX_ABRT_SOURCE をクリアする例外については、IC_TX_ABRT_SOURCE レジスタのビット 9 を参照してください。	R

3.4.15　IC_CLR_RX_UNDER ― RX_UNDER 割り込みクリアレジスタ

アドレス　4006 3044h (I2C1)
4006 4044h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_RX_UNDER
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.17　IC_CLR_RX_UNDER レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_RX_UNDER	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの RX_UNDER 割り込み（ビット 0）をクリアします。	R

3.4.16　IC_CLR_RX_OVER ― RX_OVER 割り込みクリアレジスタ

アドレス　4006 3048h (I2C1)
4006 4048h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_RX_OVER
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.18　IC_CLR_RX_OVER レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_RX_OVER	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの RX_OVER 割り込み（ビット 1）をクリアします。	R

3.4.17　IC_CLR_TX_OVER ― TX_OVER 割り込みクリアレジスタ

アドレス　4006 304Ch (I2C1)
4006 404Ch (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_TX_OVER
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.19　IC_CLR_TX_OVER レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_TX_OVER	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの TX_OVER 割り込み（ビット 3）をクリアします。	R

3.4.18　IC_CLR_RD_REQ ― RD_REQ 割り込みクリアレジスタ

アドレス　4006 3050h (I2C1)
4006 4050h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_RD_REQ
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.20　IC_CLR_RD_REQ レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_RD_REQ	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの RD_REQ 割り込み（ビット 5）をクリアします。	R

3.4.19　IC_CLR_TX_ABRT ― TX_ABRT 割り込みクリアレジスタ

アドレス　4006 3054h (I2C1)
4006 4054h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_TX_ABRT
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.21　IC_CLR_TX_ABRT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_TX_ABRT	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの TX_ABRT 割り込み（ビット 6）および IC_TX_ABRT_SOURCE レジスタをクリアします。これは、フラッシュ済み／リセット状態から TX FIFO を開放し、TX FIFO へさらに書き込めるようになります。 IC_TX_ABRT_SOURCE をクリアする例外については、IC_TX_ABRT_SOURCE レジスタのビット 9 を参照してください。	R

3.4.20　IC_CLR_RX_DONE ― RX_DONE 割り込みクリアレジスタ

アドレス　4006 3058h (I2C1)
4006 4058h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_RX_DONE
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.22　IC_CLR_RX_DONE レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_RX_DONE	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの RX_DONE 割り込み（ビット 7）をクリアします。	R

3.4.21　IC_CLR_ACTIVITY ― ACTIVITY 割り込みクリアレジスタ

アドレス　4006 305Ch (I2C1)
4006 405Ch (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_ACTIVITY
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.23　IC_CLR_ACTIVITY レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_ACTIVITY	本レジスタの読み出しは、I2C コントローラがアクティブではない場合に、ACTIVITY 割り込みをクリアします。I2C モジュールがバス上でまだアクティブである場合、ACTIVITY 割り込みビットはセットされたままです。モジュールが無効の場合、およびバス上にそれ以上のアクティビティがない場合、ハードウェアによって自動的にクリアされます。本レジスタの値を読み出すことで、IC_RAW_INTR_STAT レジスタの ACTIVITY 割り込み（ビット 8）のステータスを取得します。	R

3.4.22　IC_CLR_STOP_DET ― STOP_DET 割り込みクリアレジスタ

アドレス　4006 3060h (I2C1)
4006 4060h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_STOP_DET
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.24　IC_CLR_STOP_DET レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_STOP_DET	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの STOP_DET 割り込み（ビット 9）をクリアします。	R

3.4.23　IC_CLR_START_DET ― START_DET 割り込みクリアレジスタ

アドレス　4006 3064h (I2C1)
4006 4064h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_START_DET
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.25　IC_CLR_START_DET レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_START_DET	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの START_DET 割り込み（ビット 10）をクリアします。	R

3.4.24　IC_CLR_GEN_CALL ― GEN_CALL 割り込みクリアレジスタ

アドレス　4006 3068h (I2C1)
4006 4068h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_GEN_CALL
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.26　IC_CLR_GEN_CALL レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_GEN_CALL	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの GEN_CALL 割り込み（ビット 11）をクリアします。	R

3.4.25　IC_ENABLE ― I2C イネーブルレジスタ

アドレス　4006 306Ch (I2C1)
4006 406Ch (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	ABORT	ENABLE
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0

表 3.27　IC_ENABLE レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット		R
b1	ABORT	セットされると、コントローラは転送アボートを開始します。 0：ABORT は開始されないか、ABORT が完了済みです 1：ABORT 動作が進行中です ソフトウェアは、本ビットを設定することによりマスタモードで I2C 転送をアボートできます。ENABLE が既に設定されている場合、ソフトウェアは本ビットを設定できます。そうではない場合、コントローラは ABORT ビットへの書き込みを無視します。ソフトウェアは一度セットされた ABORT ビットをクリアできません。ABORT 開始で、コントローラは STOP を発行し、現在の転送が完了した後に Tx FIFO をフラッシュし、アボート動作後に TX_ABORT 割り込みが設定されます。アボート動作後、ABORT ビットは自動的にクリアされます。	R/W
b0	ENABLE	I2C コントローラが有効かどうかを制御します。 0：I2C コントローラを無効化（TX FIFO および RX FIFO は消去された状態で保持されます） 1：I2C コントローラを有効化 ソフトウェアは I2C コントローラがアクティブの場合でも無効にすることが可能です。ただし、I2C コントローラが適切に無効にされていることを確認してください。 I2C コントローラが無効になると、以下が発生します： ● TX FIFO および RX FIFO がフラッシュされます。 ● IC_INTR_STAT レジスタのステータスビットは、I2C コントローラがアイドル状態になるまでアクティブのままです。 モジュールが送信中の場合、現在の転送が完了した後、送信バッファの内容を削除すると同時に停止します。モジュールが受信中の場合、I2C コントローラは現在の転送を、現在のバイトの終わりで停止し、転送に応答しません。	R/W

3.4.26　IC_STATUS ― I2C ステータスレジスタ

これは、転送ステータスおよび FIFO ステータスを示す、読み出し専用のレジスタです。ステータスレジスタは、いつでも読み出しが可能です。本レジスタのいずれのビットも割り込みを要求しません。

IC_ENABLE レジスタのビット 0 に 0 を書き込むことで I2C コントローラが無効の場合：

- ビット 1 およびビット 2 は、1 に設定されます。
- ビット 3 およびビット 4 は、0 に設定されます。

マスタまたはスレーブのステートマシンがアイドルで IC_ENABLE_STATUS.IC_EN＝0 の場合：

- ビット 5 およびビット 6 は、0 に設定されます。

アドレス　4006 3070h (I2C1)
4006 4070h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	SLV_ACTIVITY	MST_ACTIVITY	RFF	RFNE	TFE	TFNF	IC_STATUS_ACTIVITY
リセット後の値	X	X	X	X	X	X	X	X	X	0	0	0	0	1	1	0

表 3.28　IC_STATUS レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b7	予約ビット		R
b6	SLV_ACTIVITY	スレーブ FSM アクティビティステータス スレーブ FSM（Finite State Machine）がアイドル状態ではない場合、本ビットはセットされます。 0：スレーブ FSM はアイドル状態のため、I2C コントローラのスレーブはアクティブではありません 1：スレーブ FSM はアイドル状態ではないため、I2C コントローラのスレーブはアクティブです	R
b5	MST_ACTIVITY	マスタ FSM アクティビティステータス マスタ FSM（Finite State Machine）がアイドル状態ではない場合、本ビットはセットされます。 0：マスタ FSM はアイドル状態のため、I2C コントローラのマスタはアクティブではありません 1：マスタ FSM はアイドル状態ではないため、I2C コントローラのマスタはアクティブです **備考**　IC_STATUS[0]、つまり IC_STATUS_ACTIVITY ビットは、SLV_ACTIVITY および MST_ACTIVITY ビットの OR 演算です。	R
b4	RFF	受信 FIFO フル 受信 FIFO が完全にフルな場合、本ビットがセットされます。 受信 FIFO に 1 つ以上のエンプティが含まれる場合、本ビットはクリアされます。 0：受信 FIFO は空き 1：受信 FIFO はフル	R

表 3.28　IC_STATUS レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3	RFNE	受信 FIFO は非エンプティ 本ビットは、受信 FIFO が 1 つ以上のエントリを含むときにセットされ、受信 FIFO がエンプティのときにクリアされます。 0：受信 FIFO はエンプティ 1：受信 FIFO は非エンプティ	R
b2	TFE	送信 FIFO エンプティ 送信 FIFO が完全にエンプティな場合、本ビットがセットされます。 1 つ以上の有効なエントリが含まれる場合、本ビットはクリアされます。本ビットフィールドは、割り込みを要求しません。 0：送信 FIFO は非エンプティ 1：送信 FIFO はエンプティ	R
b1	TFNF	送信 FIFO 空き 送信 FIFO が 1 つ以上のエンプティを含む場合にセットされ、FIFO がフルの場合にクリアされます。 0：送信 FIFO はフル 1：送信 FIFO はフルでない	R
b0	IC_STATUS_ACTIVITY	I2C アクティビティステータス 0：I2C コントローラはアクティブではありません 1：I2C コントローラはアクティブです	R

3.4.27　IC_TXFLR ― I2C 送信 FIFO レベルレジスタ

本レジスタは、送信 FIFO バッファの有効なデータエントリ数を示します。次の場合は常にクリアされます。

- I2C コントローラが無効の場合
- 送信アボートがある、つまり TX_ABRT ビットが IC_RAW_INTR_STAT レジスタでセットされている場合
- スレーブバルク送信モードがアボートされている場合

レジスタは、データが送信 FIFO に格納されたときにインクリメントし、送信 FIFO からデータが取得されたときにデクリメントします。

アドレス　4006 3074h (I2C1)
4006 4074h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	TXFLR			
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0

表 3.29　IC_TXFLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b4	予約ビット		R
b3～b0	TXFLR	送信 FIFO レベル 送信 FIFO の有効なデータエントリ数	R

3.4.28　IC_RXFLR ― I2C 受信 FIFO レベルレジスタ

本レジスタは、受信 FIFO バッファの有効なデータエントリ数を示します。次の場合は常にクリアされます。

- I2C コントローラが無効の場合
- IC_TX_ABRT_SOURCE でトラックされるイベントのいずれかで発生した送信アボートがある場合

レジスタは、データが受信 FIFO に格納されたときにインクリメントし、受信 FIFO からデータが取得されたときにデクリメントします。

アドレス　4006 3078h (I2C1)
4006 4078h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	RXFLR			
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0

表 3.30　IC_RXFLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b4	予約ビット		R
b3～b0	RXFLR	受信 FIFO レベル 受信 FIFO の有効なデータエントリ数	R

3.4.29　IC_SDA_HOLD ― I2C SDA ホールド時間長レジスタ

本レジスタのビット[15:0]は、スレーブモードおよびマスタモードで送信中の SDA ホールド時間の制御に使用されます（SCL が High から Low へ遷移後）。本レジスタのビット[23:16]は、マスタモードまたはスレーブモードのいずれかのレシーバで SCL が High の場合に、SDA 遷移（存在する場合）を延長するために使用されます。本レジスタへの書き込みは、IC_ENABLE[0]＝0 の場合のみ成功します。本レジスタの値は、I2C_SCLK 周期単位です。IC_SDA_TX_HOLD に設定される値は、各モードの最小ホールド時間（マスタモード 1 サイクル、スレーブモード 7 サイクル）を超える必要があります。送信中の設定 SDA ホールド時間（IC_SDA_TX_HOLD）は、SCL のどの Low 期間タイミングも超えることはできません。よって、設定値は N_SCL_LOW-2 より大きくすることはできません。ここで N_SCL_LOW は I2C_SCLK サイクルで計測される SCL 周期の Low 期間です。

アドレス　4006 307Ch (I2C1)
4006 407Ch (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	IC_SDA_RX_HOLD							
リセット後の値	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	IC_SDA_TX_HOLD															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

表 3.31　IC_SDA_HOLD レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b24	予約ビット		R
b23～b16	IC_SDA_RX_HOLD	I2C コントローラがレシーバとして動作する場合に、I2C_SCLK 周期単位で要求される SDA ホールド時間を設定します。	R/W
b15～b0	IC_SDA_TX_HOLD	I2C コントローラがトランスミッタとして動作する場合に、I2C_SCLK 周期単位で要求される SDA ホールド時間を設定します。	R/W

3.4.30　IC_TX_ABRT_SOURCE ― I2C 送信アボート要因レジスタ

本レジスタにはTX_ABRTビットの要因を示す32ビットがあります。ビット9以外は、本レジスタはIC_CLR_TX_ABRTレジスタまたはIC_CLR_INTRレジスタが読み出されるといつでもクリアされます。ビット9をクリアするには、ABRT_SBYTE_NORSTRTの要因を最初に修正する必要があります。RESTARTを有効化（IC_CON[5]＝1）、SPECIALビットをクリア（IC_TAR[11]）、またはGC_OR_STARTビットをクリア（IC_TAR[10]）する必要があります。

ABRT_SBYTE_NORSTRTの要因を修正すると、本ビットは本レジスタのほかのビットと同じ方法でクリアできます。本ビットをクリアする前に、ABRT_SBYTE_NORSTRTの要因が修正されない場合は、ビット9は1サイクルクリアされてから再アサートされます。

アドレス　4006 3080h (I2C1)
4006 4080h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TX_FLUSH_CNT									—	—	—	—	—	—	ABRT_USER_ABRT
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ABRT_SLVRD_INTX	ABRT_SLV_ARBLOST	ABRT_SLVFLUSH_TXFIFO	ARB_LOST	ABRT_MASTER_DIS	ABRT_10B_RD_NORSTRT	ABRT_SBYTE_NORSTRT	—	ABRT_SBYTE_ACKDET	—	ABRT_GCALL_READ	ABRT_GCALL_NOACK	ABRT_TXDATA_NOACK	ABRT_10ADDR2_NOACK	ABRT_10ADDR1_NOACK	ABRT_7B_ADDR_NOACK
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 3.32　IC_TX_ABRT_SOURCE レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b23	TX_FLUSH_CNT	本フィールドは、TX_ABRT 割り込みでフラッシュされる Tx FIFO データコマンドの数を示します。 I2C コントローラが無効なときは常にクリアされます。 I2C コントローラの役割：マスタトランスミッタまたはスレーブトランスミッタ	R
b22～b17	予約ビット		R
b16	ABRT_USER_ABRT	これはマスタモードのみのビットです。 1：マスタは転送アボートを検出しました（IC_ENABLE[1]）。 I2C コントローラの役割：マスタトランスミッタ	R
b15	ABRT_SLVRD_INTX	1：プロセッサが、リモートマスタに送信するデータとしてスレーブモード要求に応答し、IC_DATA_CMD レジスタの CMD（ビット 8）に 1 を書き込む場合 I2C コントローラの役割：スレーブトランスミッタ	R
b14	ABRT_SLV_ARBLOST	1：スレーブは、リモートマスタへのデータ送信中にバスの所有権を失いました。IC_TX_ABRT_SOURCE[12]が同時にセットされます。 **備考**　スレーブがバスを“所有”しない場合でも、バス上で不具合が発生する可能性があります。これがフェイルセーフチェックです。たとえば、Low から High への SCL 遷移でデータを送信中に、データバス上にあるものが、送信が想定されているものではない場合、I2C コントローラはバスを所有しなくなります。 I2C コントローラの役割：スレーブトランスミッタ	R
b13	ABRT_SLVFLUSH_TXFIFO	1：スレーブが読み出しコマンドを受信する際、TX FIFO にデータが存在する場合、スレーブは TX_ABRT 割り込みを発行して TX FIFO の古いデータをフラッシュします。 I2C コントローラの役割：スレーブトランスミッタ	R

表 3.32　IC_TX_ABRT_SOURCE レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b12	ARB_LOST	1：マスタが調停に失敗。もしくは IC_TX_ABRT_SOURCE[14]もセットされている場合は、スレーブトランスミッタが調停に失敗。 I2C コントローラの役割：マスタトランスミッタまたはスレーブトランスミッタ	R
b11	ABRT_MASTER_DIS	1：ユーザは、マスタモードを無効にすることでマスタ動作を初期化しようとします。 I2C コントローラの役割：マスタトランスミッタまたはマスタレシーバ	R
b10	ABRT_10B_RD_NORSTRT	1：RESTART は無効化（IC_RESTART_EN ビット（IC_CON[5])＝0）され、マスタは読み出しコマンドを 10 ビットアドレス形式モードで送信します。 I2C コントローラの役割：マスタレシーバ	R
b9	ABRT_SBYTE_NORSTRT	ビット 9 をクリアするには、ABRT_SBYTE_NORSTRT の要因をまず修正する必要があります。RESTART を有効化（IC_CON[5]＝1）、SPECIAL ビットをクリア（IC_TAR[11]）、または GC_OR_START ビットをクリア IC_TAR[10]）する必要があります。ABRT_SBYTE_NORSTRT の要因を修正すると、本ビットは本レジスタのほかのビットと同じ方法でクリアできます。本ビットをクリアする前に、ABRT_SBYTE_NORSTRT の要因が修正されない場合、ビット 9 は 1 サイクル間クリアされてから再アサートされます。 1：RESTART が無効化（IC_RESTART_EN ビット（IC_CON[5])＝0）された状態で START Byte を送信。 I2C コントローラの役割：マスタ	R
b8	予約ビット		R
b7	ABRT_SBYTE_ACKDET	1：マスタが START Byte を送信し、その START Byte が ACK 応答されました（誤った動作）。 I2C コントローラの役割：マスタ	R
b6	予約ビット		R
b5	ABRT_GCALL_READ	1：マスタモードの I2C コントローラはゼネラルコールを送信しましたが、ゼネラルコールに続いてバスから読み出すようにプログラムされました。 I2C コントローラの役割：マスタトランスミッタ	R
b4	ABRT_GCALL_NOACK	1：マスタモードの I2C コントローラはゼネラルコールを送信しましたがどのスレーブにも ACK 応答されませんでした。 I2C コントローラの役割：マスタトランスミッタ	R
b3	ABRT_TXDATA_NOACK	1：これはマスタモードのみのビットです。マスタはアドレスの ACK 応答を受信しますが、アドレスに続くデータバイトを送信したとき、リモートスレーブからの ACK 応答を受信しませんでした。 I2C コントローラの役割：マスタトランスミッタ	R
b2	ABRT_10ADDR2_NOACK	1：マスタは 10 ビットのアドレスモードで、10 ビットアドレスのセカンドアドレスバイトはどのスレーブにも ACK 応答されませんでした。 I2C コントローラの役割：マスタトランスミッタまたはマスタレシーバ	R
b1	ABRT_10ADDR1_NOACK	1：マスタは 10 ビットのアドレスモードで、最初の 10 ビットアドレスバイトはどのスレーブにも ACK 応答されませんでした。 I2C コントローラの役割：マスタトランスミッタまたはマスタレシーバ	R
b0	ABRT_7B_ADDR_NOACK	1：マスタは 7 ビットのアドレスモードで、送信されたアドレスはどのスレーブにも ACK 応答されませんでした。 I2C コントローラの役割：マスタトランスミッタまたはマスタレシーバ	R

3.4.31　IC_SLV_DATA_NACK_ONLY ― スレーブデータ NACK 生成レジスタ

レジスタは、I2C コントローラがスレーブレシーバとして動作するときに、転送データ用の NACK を生成するために使用されます。

次の条件の両方が満たされた場合に、本レジスタ上で書き込みが可能です。

- I2C コントローラが無効（IC_ENABLE[0]＝0）
- スレーブが非アクティブ（IC_STATUS[6]＝0）

アドレス　4006 3084h (I2C1)
4006 4084h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	NACK
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.33　IC_SLV_DATA_NACK_ONLY レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	NACK	NACK 生成 本 NACK 生成は、I2C コントローラがスレーブレシーバの場合のみ発生します。本レジスタが 1 の値にセットされた場合、データバイトが受信された後のみ、NACK を生成できます。そのため、データ転送はアボートされ、受信したデータは受信バッファに入りません。 レジスタが 0 の値にセットされた場合、通常の基準により NACK/ACK を生成します。 1：データバイト受信後に NACK を生成 0：NACK/ACK を通常生成	R/W

3.4.32　IC_SDA_SETUP ― I2C SDA セットアップレジスタ

本レジスタは、スレーブトランスミッタ動作で I2C コントローラが読み出し要求に対応するときに、SDA 変化に関連して SCL が立ち上がるエッジの遅延時間（I2C_SCLK 周期単位）を制御します。関連する I2C としての要求は、I2C バス仕様で定義される $t_{SU;DAT}$ です。本レジスタは 2 以上の値でプログラムされる必要があります。

本レジスタへの書き込みは、IC_ENABLE[0]＝0 の場合のみ成功します。

アドレス　4006 3094h (I2C1)
4006 4094h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	SDA_SETUP							
リセット後の値	X	X	X	X	X	X	X	X	0	1	1	0	0	1	0	0

表 3.34　IC_SDA_SETUP レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	SDA_SETUP	SDA セットアップ 要求された遅延が 1000ns の場合、10MHz の I2C_SCLK 周波数に対して、IC_SDA_SETUP は 11 の値で設定されることが推奨されます。IC_SDA_SETUP の最小値は 2 です。	R/W

3.4.33　IC_ACK_GENERAL_CALL ― I2C ACK ゼネラルコールレジスタ

本レジスタは、I2C ゼネラルコールアドレスを受信したときに、I2C コントローラが ACK または NACK のどちらで応答するかを制御します。

備　考

本レジスタは I2C コントローラがスレーブモードの場合にのみ適用可能です。

アドレス　4006 3098h (I2C1)
4006 4098h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	ACK_GEN_CALL
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

表 3.35　IC_ACK_GENERAL_CALL レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	ACK_GEN_CALL	ACK ゼネラルコール 1 にセットすると、I2C コントローラは、ゼネラルコールを受信するときに ACK で応答します。そうでない場合は、I2C コントローラは NACK で応答します。	R/W

3.4.34　IC_ENABLE_STATUS ― I2C イネーブルステータスレジスタ

本レジスタは、IC_ENABLE[0]レジスタが1から0にセットされたとき、つまりI2Cコントローラが無効にされたときに、I2Cコントローラハードウェアステータスのレポートに使用されます。

IC_ENABLE[0]が1に設定されている場合、ビット[2:1]は強制的に0になり、ビット0は強制的に1になります。

IC_ENABLE[0]が0に設定されている場合、ビット[2:1]はビット0が“0”のときのみ有効です。

備　考

IC_ENABLE[0]に“0”で書き込まれたとき、I2C コントローラが無効になるのは I2C バスのアクティビティ次第のため、ビット 0 が“0”になるまで遅延が発生します。

アドレス　4006 309Ch (I2C1)
4006 409Ch (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	SLV_RX_DATA_LOST	SLV_DISABLED_WHILE_BUSY	IC_EN
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0

表 3.36　IC_ENABLE_STATUS レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b3	予約ビット		R
b2	SLV_RX_DATA_LOST	スレーブ受信データ消失 本ビットは、I2C 転送でなくとも 1 つのデータバイトを受信した状態で、IC_ENABLE のビット 0 が 1 から 0 にセットされたために、スレーブレシーバ動作がアボートされたことを示します。 1 が読み出されるとき、データバイトに NACK で応答したとしても、I2C コントローラは、アボートされた I2C 転送（アドレスが一致する）に関わっており、データフェーズであったことが想定されます。 **備考）** ● I2C コントローラが転送に NACK を送る前にリモート I2C マスタが STOP コンディションで転送を終了し、IC_ENABLE[0]が 0 にセットされていた場合も、本ビットは 1 にセットされます。 0 が読み出されたとき、I2C コントローラは、スレーブレシーバ転送のデータフェーズにアクティブに関わることなく無効にされたと想定されます。 ● IC_EN（ビット 0）が 0 のとき、CPU は本ビットを安全に読み出すことが可能です。	R

表 3.36　IC_ENABLE_STATUS レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b1	SLV_DISABLED_WHILE_BUSY	ビジー中にスレーブが無効（送信、受信） 本ビットは、IC_ENABLE レジスタのビット 0 を 1 から 0 に設定する際に、スレーブ動作がアボートされたかどうかを示します。本ビットは、次の場合に CPU が 0 を IC_ENABLE レジスタに書き込むと、セットされます：（a）I2C コントローラがスレーブトランスミッタ動作でアドレスバイトをリモートマスタから受信中、または（b）リモートマスタからのスレーブレシーバ動作のアドレスおよびデータバイトである。 1 として読み出されるとき、I2C アドレスが I2C コントローラ（IC_SAR レジスタ）に設定されているスレーブアドレスに一致したかどうか、または、IC_ENABLE が 0 に設定されたがそれが有効になる前に転送が完了したかどうかにかかわらず、I2C コントローラは、I2C 転送の任意の部分で NACK を強制したと考えられます。 **備考）** ● I2C コントローラが NACK を送る前にリモート I2C マスタが STOP コンディションで転送を終了し、IC_ENABLE[0]が 0 に設定された場合も、本ビットは 1 にセットされます。0 が読み出される場合は、I2C コントローラは、マスタアクティビティがあるとき、または I2C バスがアイドルのときに、無効にされたと考えられます。 ● IC_EN（ビット 0）が 0 のときに、CPU は本ビットを安全に読み出すことが可能です。	R
b0	IC_EN	I2C インタフェースステータス 1 が読み出される場合、I2C コントローラはイネーブルステートです。 0 が読み出される場合、I2C コントローラは完全に非アクティブです。 **備考）** CPU は本ビットをいつでも読み出せます。本ビットが 0 のとき、CPU は SLV_RX_DATA_LOST（ビット 2）および SLV_DISABLED_WHILE_BUSY（ビット 1）を安全に読み出すことが可能です。	R

3.4.35　IC_FS_SPKLEN ― I2C Sm、Fm スパイク抑止制限

本レジスタは、コンポーネントが標準モード、ファストモードで動作中に、スパイク抑制ロジックでフィルタ処理される最長スパイクに関して、I2C_SCLK サイクル単位の期間を格納します。関連する I2C としての要求は、I2C バス仕様で定義される t_{SP} です。本レジスタの最小値は 1 です。

アドレス　4006 30A0h (I2C1)
4006 40A0h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	IC_FS_SPKLEN							
リセット後の値	X	X	X	X	X	X	X	X	0	0	0	0	0	1	0	1

表 3.37　IC_FS_SPKLEN レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット		R
b7～b0	IC_FS_SPKLEN	本レジスタは、安定動作を確保するために任意の I2C バストランザクションが行われる前に送信される必要があります。本レジスタは、スパイク抑制ロジックでフィルタ処理される SCL ラインまたは SDA ラインでの最長スパイクに関して I2C_SCLK サイクル単位の期間を設定します。 本レジスタは、I2C インタフェースが無効の場合（IC_ENABLE[0]＝0）のみ書き込み可能です。他の場合の書き込みは無効になります。 有効な最小値は 1 です。ハードウェアはこの値未満の場合は、1 をセットします。	R/W

3.4.36　IC_CLR_RESTART_DET ― RESTART_DET 割り込みクリアレジスタ

アドレス　4006 30A8h (I2C1)
4006 40A8h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	CLR_RESTART_DET
リセット後の値	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

表 3.38　IC_CLR_RESTART_DET レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット		R
b0	CLR_RESTART_DET	本レジスタを読み出すことで、IC_RAW_INTR_STAT レジスタの RESTART_DET 割り込み（ビット 12）をクリアします。	R

3.4.37　IC_COMP_PARAM_1 ― コンポーネントパラメータレジスタ 1

これは、I2C コントローラパラメータ設定に関するエンコード情報を含む定数読み出しのみのレジスタです。

アドレス　4006 30F4h (I2C1)
4006 40F4h (I2C2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	TX_BUFFER_DEPTH							
リセット後の値	X	X	X	X	X	X	X	X	0	0	0	0	0	1	1	1

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RX_BUFFER_DEPTH								ADD_ENCODED_PARAMS	HAS_DMA	INTR_IO	HC_COUNT_VALUES	MAX_SPEED_MODE		APB_DATA_WIDTH	
リセット後の値	0	0	0	0	0	1	1	1	1	0	1	0	1	0	1	0

表 3.39　IC_COMP_PARAM_1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b24	予約ビット		R
b23～b16	TX_BUFFER_DEPTH	送信バッファの深さです。 0x7：バッファの深さは 8 です。	R
b15～b8	RX_BUFFER_DEPTH	受信バッファの深さです。 0x7：バッファの深さは 8 です。	R
b7	ADD_ENCODED_PARAMS	エンコード情報が付加されたかどうかを示します。 1：これらのエンコードパラメータをソフトウェア経由で読み出し可能です。	R
b6	HAS_DMA	DMA 要求インタフェース信号が利用可能かどうかを示します。 0：不可	R
b5	INTR_IO	割り込みが個々に行われるか、あるいは 1 つに統合して行われるかを示します。 1：統合	R
b4	HC_COUNT_VALUES	*CNT レジスタが書き込み可能か、読み出しのみかを示します。 0：書き込み可能	R
b3、b2	MAX_SPEED_MODE	I2C コントローラでサポートされる I2C モードを示します。 2：ファストモード	R
b1、b0	APB_DATA_WIDTH	I2C コントローラが使用している APB データバスの幅を示します。 2：32 ビット	R

3.5　動作モード

I2C コントローラは I2C マスタまたは I2C スレーブのどちらかでしか動作できず、同時に両方はできないことに注意してください。IC_CON レジスタのビット 6（IC_SLAVE_DISABLE）およびビット 0（MASTER_MODE）がそれぞれ 0 と 1 にセットされることはありません。

3.5.1　スレーブモード動作

3.5.1.1　初期設定

I2C コントローラをスレーブとして使用するには、以下の手順を実行します。

(1) IC_ENABLE レジスタのビット 0 に“0”を書き込むことで I2C を無効にします。

(2) IC_SAR レジスタ（ビット[9:0]）を書き込んでスレーブアドレスをセットします。これは、I2C コントローラが応答するアドレスです。

(3) IC_CON レジスタへの書き込みで、サポートされるアドレス形式の種類（ビット 3 のセットにより、7 ビットまたは 10 ビット）を指定します。ビット 6（IC_SLAVE_DISABLE）に“0”を書き込み、ビット 0（MASTER_MODE）に“0”を書き込み、スレーブの I2C コントローラを有効にします。

備　考

スレーブおよびマスタは、7 ビットアドレスまたは 10 ビットアドレスの種類を揃えてプログラムする必要はありません。たとえば、スレーブは 7 ビットアドレス形式で、マスタは 10 ビットアドレス形式でも可能であり、その逆も可能です。

(4) IC_ENABLE レジスタのビット 0 に“1”を書き込むことで I2C コントローラを有効にします。

注　意

- I2C バスがアイドルの場合のみ、I2C スレーブをリセット解除することを推奨します。転送がバス上で継続しているときにリセットアサート解除すると、SDL および SCL の同期化回路がリセット値からバス上の実際の値に変化します。これにより、SCL が 1 の間に SDA が 1 から 0 へトグルすると、偽の START コンディションが I2C スレーブによって検出されてしまいます。
- これは、I2C コントローラを IC_SLAVE_DISABLE＝1 かつ MASTER_MODE＝1 と設定することでも回避でき、これによってスレーブインタフェースはリセット後に無効になります。内部 SDA および SCL がバス上の値と同期化された後、IC_CON[0]＝0 かつ IC_CON[6]＝0 とプログラミングすることでスレーブを有効にできます。これには、リセットのアサート解除後、I2C_SCLK で約 6 サイクルかかります。

3.5.1.2　シングルバイトのスレーブトランスミッタ動作

バス上の別の I2C マスタデバイスが I2C コントローラをアドレス指定してデータを要求するとき、I2C コントローラはスレーブトランスミッタとして動作し、次の手順が発生します。

(5) 他の I2C マスタデバイスが、I2C コントローラの IC_SAR レジスタのスレーブアドレスと一致するアドレスで I2C 転送を開始します。

(6) I2C コントローラは、アドレスを確認し、転送方向によりスレーブトランスミッタとして動作することを識別します。

(7) I2C コントローラは、RD_REQ 割り込みをアサートし（IC_RAW_INTR_STAT レジスタのビット 5）、SCL ラインを Low にホールドします。ソフトウェアが応答するまでウェイトステートになります。IC_INTR_MASK[5]レジスタ（M_RD_REQ ビットフィールド）が 0 に設定されて RD_REQ 割り込みがマスクされる場合、ハードウェアおよび／またはソフトウェアのタイミングルーチンにより、IC_RAW_INTR_STAT レジスタの定期的な読み出しの実行を CPU に指示することを推奨します。

- IC_RAW_INTR_STAT[5]（R_RD_REQ ビットフィールド）が 1 にセットされている場合、RD_REQ 割り込みがアサートされているのと同じ処理が必要です。
- ソフトウェアは、その後、I2C マスタの要求を満たす必要があります。
- タイミング間隔は、I2C コントローラが処理できる最速の SCL クロック周期のほぼ 10 倍である必要があります。たとえば、400kb/s に対して、タイミング間隔は 25µs です。

備　考

I2C バス上で転送されるデータのシングルバイトに必要な時間に近いため、10 という値がここで推奨されています。

(8) 読み出し要求を受信する際に Tx FIFO にデータが残っている場合は、I2C コントローラは TX_ABRT 割り込み（IC_RAW_INTR_STAT レジスタのビット 6）をアサートして TX FIFO から古いデータをフラッシュします。

備　考

TX_ABRT イベントが発生するときは、常に Tx FIFO が強制的にフラッシュ状態／リセット状態になるため、Tx FIFO に書き込む前に、IC_CLR_TX_ABRT レジスタを読み込んで I2C コントローラをこの状態から復帰させることが重要です。詳細については、IC_RAW_INTR_STAT レジスタを参照してください。

IC_INTR_MASK[6]レジスタ（M_TX_ABRT ビットフィールド）が 0 に設定されたことにより TX_ABRT 割り込みがマスクされた場合、タイミングルーチン（前のステップで説明）の再利用、または同様のものを IC_RAW_INTR_STAT レジスタの読み出しに使用することを推奨します。

- ビット 6（R_TX_ABRT）が 1 にセットされている場合、TX_ABRT 割り込みがアサートされるのと同等の処理が必要です。
- ソフトウェアからはさらなるアクションはありません。
- タイミング間隔は、IC_RAW_INTR_STAT[5]レジスタについて前述のステップで説明したものと同様です。

(9) ソフトウェアは、（ビット 8 に“0”を書き込んだ上で）IC_DATA_CMD レジスタに書き込み対象データを書き込みます。

(10) ソフトウェアは、IC_RAW_INTR_STAT レジスタの RD_REQ 割り込みおよび TX_ABRT 割り込み（それぞれビット 5 およびビット 6）をクリアする必要があります。

(11) I2C コントローラは SCL を解放し、バイトを送信します。

(12) マスタは、RESTART コンディションを発行して I2C バスをホールドしたり、または STOP コンディションを発行してバスを解放したりすることが可能です。

3.5.1.3　シングルバイトのスレーブレシーバ動作

バス上の別の I2C マスタデバイスが I2C コントローラをアドレス指定してデータを送信するとき、I2C コントローラはスレーブレシーバトランスミッタとして動作し、次の手順が発生します。

(1) 他の I2C マスタデバイスが、IC_SAR レジスタの I2C スレーブアドレスと一致するアドレスで I2C 転送を開始します。

(2) I2C コントローラは、アドレスを確認し、転送方向によりスレーブレシーバとして動作することを識別します。

(3) I2C コントローラは送信されたバイトを受信バッファに格納します。

(4) I2C コントローラは RX_FULL 割り込みをアサートします（IC_RAW_INTR_STAT[2]レジスタ）。IC_INTR_MASK[2]レジスタを 0 に設定、または IC_RX_TL を 0 より大きい値にすることで、RX_FULL 割り込みがマスクされる場合、（「シングルバイトのスレーブトランスミッタ動作」と同様に）タイミングルーチンで、IC_STATUS レジスタの定期的な読み出しを実装してください。IC_STATUS レジスタを読み出してビット 3（RFNE）が 1 の場合、RX_FULL 割り込みがアサートされているときと同じ処理が必要です。

(5) ソフトウェアは IC_DATA_CMD レジスタ（ビット[7:0]）からバイトを読み出すことができます。

(6) 他のマスタデバイスは、RESTART コンディションを発行して I2C バスをホールドしたり、または STOP 条件を発行してバスを解放したりすることが可能です。

3.5.1.4　バルク転送のスレーブ転送動作

標準の I2C 製品では、すべてのトランザクションはシングルバイトトランザクションであり、プログラマは、スレーブ TX FIFO へ 1 バイトを書き込むことでリモートマスタの読み出し要求に応答します。スレーブ（スレーブトランスミッタ）がリモートマスタ（マスタレシーバ）から読み出し要求（RD_REQ）される場合、少なくとも 1 つのエントリがスレーブトランスミッタの TX FIFO に格納されている必要があります。I2C コントローラは、TX FIFO でより多くのデータを扱うことで割り込みを発行することなく、後続の読み出し要求がより多くのデータを取得できるようにします。これによって、TX FIFO にエントリを 1 つしか格納できないという制限により、データに対する割り込みが毎回起き、相当なレイテンシが発生する可能性を削減します。

本モードは、I2C コントローラがスレーブトランスミッタとして動作する場合のみ発生します。リモートマスタがスレーブトランスミッタによって送信されたデータを確認し、スレーブの TX FIFO にデータが存在しない場合、I2C コントローラは、読み出し要求割り込み（RD_REQ）を発生させている間は I2C SCL ラインを Low にホールドし、データが TX FIFO に書き込まれるのを待機してからデータをリモートマスタに送信します。

IC_INTR_MASK レジスタのビット 5（M_RD_REQ）が 0 に設定されているために、RD_REQ 割り込みがマスクされる場合、IC_RAW_INTR_STAT レジスタの定期的な読み出しをするためにタイミングルーチンの使用を推奨します。IC_RAW_INTR_STAT レジスタのビット 5（RD_REQ）が 1 にセットされている場合、本項で参照される RD_REQ 割り込みと同等に扱われる必要があります。本タイミングルーチンは、「シングルバイトのスレーブトランスミッタ動作」で説明されるものと同様です。

RD_REQ 割り込みは、読み出し要求で発生し、割り込みと同様に、割り込み処理ルーチン（ISR）終了時にクリアされます。ISR は、Tx FIFO への 1 バイト以上の書き込みを許可します。これらのバイトをマスタへ送信する際、マスタが最後のバイトを確認している場合、マスタはさらにデータを要求するために、スレーブは RD_REQ を再び発生させる必要があります。

リモートマスタが n バイトのパケットを要求することを知っている場合は、事前に Tx FIFO に n バイト書き込んでおくと、リモートマスタはデータを連続ストリームとして受信できます。I2C スレーブは、リモートマスタが送信データを確認し、Tx FIFO に有効なデータがあるかぎり、データをリモートマスタに送信し続けます。SCL ライン Low をホールドしたり、RD_REQ を再生成したりする必要はありません。

リモートマスタが I2C コントローラから n バイトを受信する際、プログラマが Tx FIFO へ n より大きいバイト数を書き込んでいる場合、スレーブは n バイトを送信し終わると、Tx FIFO をクリアして超過バイトを無視します。

I2C コントローラは、このような例で Tx FIFO のクリアを示すための送信アボート（TX_ABRT）イベントを生成します。ACK/NACK が期待されるときに、NACK が受信された場合、リモートマスタには必要なすべてのデータがあります。このとき、Tx FIFO に残されたデータをクリアするために、スレーブのステートマシン内でフラグが立ち上がり、FIFO が存在しているプロセッサバスクロックドメインに転送され、Tx FIFO の内容がクリアされます。

3.5.2　マスタモード動作

3.5.2.1　初期設定

I2C コントローラをマスタとして使用するには、以下の手順を実行します。

(1) IC_ENABLE レジスタのビット 0 に 0 を書き込むことで、I2C コントローラを無効にします。

(2) スレーブ動作でサポートされる最大スピードモード（ビット[2:1]）の設定を IC_CON レジスタへ書き込みます。

(3) IC_TAR レジスタに、アドレス指定する I2C デバイスのアドレスを書き込みます。ゼネラルコールまたは START BYTE コマンドのどちらで I2C コントローラが実行するかも指定します。7/10 ビットアドレス形式モードのどちらで転送を開始するかは、IC_10BITADDR_MASTER ビット（ビット 12）で指定されます。

(4) IC_ENABLE レジスタのビット 0 に 1 を書き込むことで、I2C コントローラを有効にします。

(5) ここで、IC_DATA_CMD レジスタに転送方向および送信されるデータを書き込みます。I2C コントローラが有効になる前に IC_DATA_CMD レジスタに書き込まれる場合、バッファがクリアされたままとなるため、データおよびコマンドは失われます。

3.5.2.2　動的 IC_TAR または IC_10BITADDR_MASTER の更新

I2C コントローラは、IC_TAR レジスタの IC_TAR（ビット[9:0]）および IC_10BITADDR_MASTER（ビット 12）ビットフィールドの動的更新をサポートします。既存の TAR アドレスを使用する他のコマンドが Tx FIFO にないことをソフトウェアが保証する場合、IC_TAR レジスタに動的に書き込むことが可能です。ソフトウェアがこれを保証できない場合は、次の条件を満たした上で、IC_TAR を再設定する必要があります。

- I2C コントローラが有効ではない（IC_ENABLE[0]＝0）
  または
  （I2C コントローラが有効である（IC_ENABLE[0]＝1）
  かつ
  I2C コントローラは任意のマスタ（tx、rx）動作にかかわっていない（IC_STATUS[5]＝0）
  かつ
  I2C コントローラはマスタモードで動作することを有効にされている（IC_CON[0]＝1）
  かつ
  TX FIFO にエントリがない（IC_STATUS[2]＝1）

次の条件を満たす場合のみ、バス制御権を失うことなく、TAR アドレスを動的に変更できます。

- I2C コントローラが有効である（IC_ENABLE[0]＝1）かつ
  I2C コントローラはマスタモードで動作することを有効にされている（IC_CON[0]＝1）かつ
  Tx FIFO にエントリがなくマスタが HOLD 状態（IC_INTR_STAT[13]＝1）

備　考

I2C コントローラは、次の条件のいずれかが真の場合に、TAR アドレスを使用します。

- セットした RESTART ビットまたは STOP ビットがコマンドにある場合
- 書き込みコマンドに続く読み出しコマンドで方向が変更された場合、またはその逆。

次の START または RESTART が発生したときのみ、更新された TAR アドレスが有効になります。

3.5.2.3　マスタ送信およびマスタ受信

I2C コントローラは、動的な読み出しおよび書き込みの切り替えをサポートします。データを送信するために、IC_DATA_CMD レジスタの下位バイトにデータを書き込みます。I2C 書き込み動作は、CMD ビット[8]に 0 を書き込む必要があります。その後、読み出しコマンドが実行される場合は、IC_DATA_CMD レジスタの下位バイトはドントケアで構いませんが、CMD ビットに 1 が書き込まれる必要があります。マスタの I2C コントローラは、コマンドが送信 FIFO に存在するかぎり、転送を続けます。送信 FIFO がエンプティになると、マスタは IC_DATA_CMD[9]が 1 にセットされているかどうかチェックします。1 にセットされている場合は現在の転送完了後に STOP コンディションが発行され、0 にセットされている場合は次のコマンドが送信 FIFO に書き込まれるまで、SCL Low をホールドします。

3.5.3　I2C コントローラの無効化

IC_ENABLE レジスタのビット 0 が 1 から 0 に設定される応答として、I2C コントローラがシャットダウンしたとき問題が発生したかどうかをソフトウェアが明確に判断できるように IC_ENABLE_STATUS レジスタがあります。

備　考

処理中の現在のコマンドが STOP ビットを 1 にセットした場合のみ（IC_ENABLE がデアサートするとき）、I2C マスタを無効にできます。
STOP ビットをセットせずにコマンド処理中に I2C マスタを無効にしようとすると、I2C マスタはアクティブ状態を継続し、新しいコマンドが Tx FIFO で受信されるまで SCL ライン Low をホールドします。

3.5.3.1　処理手順

1. システムで使用され、I2C コントローラがサポートする、最速の I2C 転送スピードの信号期間の 10 倍に等しいタイマー間隔（ti2c_poll）を定義してください。たとえば、最速の I2C 転送スピードが 400kb/s の場合、この ti2c_poll は 25µs です。
2. 最大のタイムアウトパラメータ MAX_T_POLL_COUNT を定義し、ポーリング動作がこの最大値を超える場合に、エラーが報告されるようにします。
3. 保留中の転送が完了することを許可しつつ、スレッド／プロセス／関数によるブロックで、ソフトウェアによって I2C マスタトランザクションが開始されることを防いでください。
   I2C コントローラが、I2C スレーブとしてのみ動作するようにプログラミングされる場合、本手順は無視できます。
4. 変数 POLL_COUNT を 0 に初期化します。
5. IC_ENABLE レジスタのビット 0 を 0 に設定します。
6. IC_ENABLE_STATUS レジスタを読み出し、IC_EN ビット（ビット 0）をチェックします。
   POLL_COUNT を 1 つインクリメントします。POLL_COUNT ≧ MAX_T_POLL_COUNT の場合、関連するエラーコードで終了します。
7. IC_ENABLE_STATUS[0]が 1 の場合、ti2c_poll の間スリープし、前の手順に戻ります。そうでない場合は、関連する正常終了コードで終了します。

3.5.4　I2C 転送のアボート

IC_ENABLE レジスタの ABORT 制御ビットにより、Tx FIFO から発行された転送コマンドを完了する前に、ソフトウェアが I2C バスを放棄することが可能です。ABORT 要求の応答で、コントローラは I2C バス上に STOP コンディションを発行し、Tx FIFO フラッシュが続きます。転送のアボートは、マスタモード動作の場合のみ許可されます。

3.5.4.1　処理手順

8. Tx FIFO（IC_DATA_CMD）に新しいコマンドで満たすのを停止します。
9. IC_ENABLE レジスタのビット 1（ABORT）を 1 にセットします。
10. TX_ABRT 割り込みを待機します。
11. IC_TX_ABRT_SOURCE レジスタを読み出し、ABRT_USER_ABRT が要因になっていることを確認します。

3.6　I2C コントローラのプログラミング

3.6.1　スパイク抑制

I2Cコントローラには、標準モード／ファストモードのI2Cバス仕様（t_{SP}）によって課される要件に一致するプログラマブルスパイク抑制ロジックを含みます。
本ロジックは、入力信号（SCLおよびSDA）をモニタするカウンタをベースにしており、内部的にサンプリングする前に、所定のI2C_SCLKサイクルの間安定かどうかをチェックします。信号（SCLおよびSDA）ごとに1つの独立したカウンタがあります。I2C_SCLKサイクル数は、ユーザによってプログラムでき、I2C_SCLKの周波数と関連するスパイク長仕様を考慮して計算する必要があります。
各カウンタは、入力信号の値が変更したときに常に開始されます。入力信号の挙動によって、次のうちどちらかが発生します。

- カウンタがそのカウント制限値に達するまで、入力信号が変更しないままである。このとき、内部信号は入力値で更新され、カウンタはリセットされて停止します。カウンタは、入力信号で新しい変化が検出されるまで再スタートしません。
- カウンタがそのカウント制限値に達する前に、入力信号が再度変化すると、カウンタはリセットされ停止しますが、内部信号は更新されません。カウンタは、入力信号で新しい変化が検出されるまで停止したままになります。

I2Cバス仕様では、標準モードとファストモードの最大スパイク長は50nsです。レジスタIC_FS_SPKLENは、標準モードおよびファストモードの最大スパイク長を保持します。
レジスタは8ビット幅であり、APBインタフェース経由でアクセス可能です。ただし、I2Cコントローラが無効の場合のみ書き込みが可能です。レジスタにプログラミング可能な最小値は1であり、1より小さい値をプログラミングしようとすると、値1が書き込まれます。
たとえば、10nsのI2C_SCLK周期に対して、使用されるカウント上限値は5です（50nsスパイク抑制）。
IC_FS_SPKLENレジスタにプログラム可能な最小値は1であるため、スパイク長仕様をI2C_SCLKの低周波数周期が超えることがあります。10MHz（100ns周期）のI2C_SCLKの簡単な例を考えると、この場合、プログラム可能な最小スパイク長は100nsであり、この長さまでスパイクが抑制されることを意味します。

3.6.2　I2C_SCLK 周波数設定

I2C コントローラがマスタとして設定されている場合、適切な I/O タイミングを確認するため、I2C バストランザクションが行われる前に、*CNT レジスタを設定する必要があります。*CNT レジスタとは次のものです。

- IC_SS_SCL_HCNT
- IC_SS_SCL_LCNT
- IC_FS_SCL_HCNT
- IC_FS_SCL_LCNT

備　考

I2C スレーブとしてのみ動作するように I2C コントローラが有効な場合、任意の*CNT レジスタをプログラムする必要はありません。これらのレジスタは、I2C マスタとしての動作の SCL タイミング要求を決定するためだけに使用されます。

3.6.2.1　最小の High カウントおよび Low カウント

I2C コントローラが I2C マスタとして動作するとき、送信および受信両方の転送に対し次のようになります。

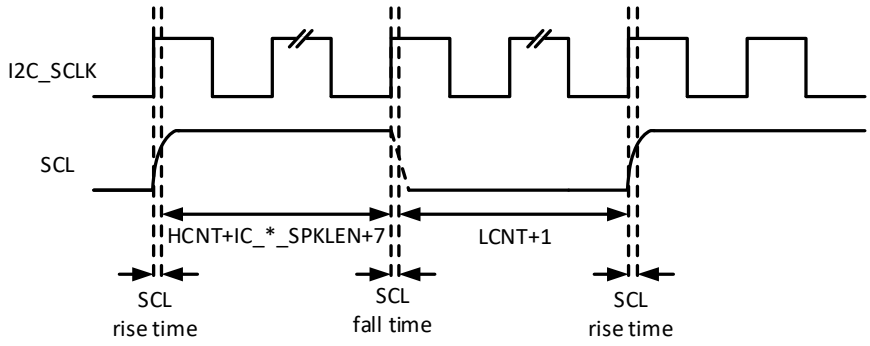
- IC_SS_SCL_LCNT レジスタおよび IC_FS_SCL_LCNT レジスタ値は、IC_FS_SPKLEN＋7 より大きい必要があります。
- IC_SS_SCL_HCNT レジスタおよび IC_FS_SCL_HCNT レジスタ値は、IC_FS_SPKLEN＋5 より大きい必要があります。

I2C コントローラの High カウントおよび Low カウントの詳細については以下のとおりです。

- *_LCNT レジスタに対する IC_FS_SPKLEN＋7 の最小値は、SCL のネガティブエッジ後、SDA を動作させるために I2C コントローラが必要な時間に起因します。
- *_HCNT レジスタに対する IC_FS_SPKLEN＋5 の最小値は、SCL の High 期間中、SDA をサンプリングするために I2C コントローラが必要な時間に起因します。
- I2C コントローラは、SCL クロックの Low 期間を生成するために、プログラムされた*_LCNT 値に 1 サイクルを追加します。これは、（*_LCNT＋1）までカウントする SCL Low のカウントロジックに起因します。
- I2C コントローラは、SCL クロックの High 期間を生成するために、プログラムされた*_HCNT 値に IC_FS_SPKLEN＋7 サイクルを追加します。これは、次の要因のためです。
  - （*_HCNT＋1）までの SCL High カウントのカウンタロジック
  - SCL ラインに適用されるデジタルフィルタリングは、I2C_SCLK で（IC_FS_SPKLEN＋2）サイクルの遅延を発生させます。
  - 本フィルタリングには、SDA エッジおよび SCL エッジ上のメタスタビリティ除去およびプログラマブルスパイク抑制を含みます。
  - I2C コントローラによって SCL が 1 から 0 に駆動される（つまり、SCL High 時間が完了）場合は常に、I2C_SCLK で 3 サイクル分の内部ロジック遅延が発生します。したがって、I2C コントローラが可能な SCL Low 時間の最小値は、I2C_SCLK で 9 周期（7＋1＋1）、最小 SCL High タイムは I2C_SCLK で 13 周期（6＋1＋3＋3）です。

備　考

I2C コントローラマスタにより生成される SCL の High 時間および Low 時間の合計は、下図に示すように、SCL ラインの立ち上がり時間および立ち下り時間によっても影響を受けます。SCL の立ち上がり時間および立ち下がり時間のパラメータは、IO ドライバの特性、プルアップレジスタ値、SCL ラインの全体の容量などの外部要因によって変化します。これらの特性は I2C コントローラの制御外です。



SCL_High_time = [(HCNT + IC_*_SPKLEN + 7) * I2C_SCLK] + SCL_Fall_time
SCL_Low_time = [(LCNT + 1) * I2C_SCLK] − SCL_Fall_time + SCL_Rise_time

図 3.2　SCL High 時間および SCL Low 時間

下の表は、High カウントおよび Low カウントの値のあるすべてのモードでの最小の I2C_SCLK 値を示します。SCL_Rise_time および SCL_Fall_time は考慮されません。

表 3.40　最小の High カウントおよび Low カウント

スピードモード	最小の I2C_SCLK 周波数（MHz）	最小の IC_FS_SPKLEN	IC_*_SCL_LCNT	SCL Low 時間（µs）	IC_*_SCL_HCNT	SCL High 時間（µs）
標準モード	2.7	1	12	4.7	6	5.2
ファストモード	12	1	15	1.33	6	1.16

備考　I2C_SCLK は I2C_PCLK より速いか同じである必要があります。

3.6.3　SDA ホールド時間

I2C プロトコル仕様は、標準モードおよびファストモードで SDA 信号（$t_{HD;DAT}$）でのホールド時間 300ns を必要とします。

SCL および SDA 信号のボード遅延は、ホールド時間の要件を I2C マスタでは満たすものの、I2C スレーブでは満たさないこと（またはその逆）を意味する可能性があります。各アプリケーションは異なるボード遅延を取り扱うため、I2C コントローラにはソフトウェアプログラマブルレジスタ（IC_SDA_HOLD）があり、SDA ホールド時間の動的調整が可能です。

ビット[15:0]は、スレーブモードおよびマスタモード両方での送信中に SDA のホールド時間を制御するために使用されます（SCL が High から Low へ遷移後）。

ビット[23:16]は、マスタモードまたはスレーブモードでレシーバでの SCL が High の場合はいつでも、SDA 遷移（存在する場合）を延長するために使用されます。

スピードモードによって異なる SDL ホールド時間が必要な場合、スピードモードが変更されるときに、IC_SDA_HOLD レジスタを再プログラミングする必要があります。I2C コントローラが無効（IC_ENABLE[0]＝0）の場合のみ、IC_SDA_HOLD レジスタをプログラムできます。

3.6.3.1　レシーバでの SDA ホールドタイミング

I2C コントローラがレシーバとして動作するとき、I2C プロトコルによって、デバイスは内部的に SDA ラインをホールドして、SCL のロジック 1 およびロジック 0 間の未定義ギャップを調整します。

IC_SDA_RX_HOLD は、I2C コントローラが受信 SDA ラインに適用する内部ホールド時間を変えるために使用できます。IC_SDA_RX_HOLD レジスタの各値は、I2C_SCLK の 1 周期単位を示します。
IC_SDA_RX_HOLD の最小値は 0 です。本ホールド時間は、SCL が High の場合のみ適用可能です。SCL が内部的に Low になった後、レシーバは SDA を延長しません。

3 以上の IC_SDA_RX_HOLD を持つレシーバとしての I2C プロトコルを下図に示します。
IC_SDA_RX_HOLD が 3 より大きい場合、SCL が内部的に Low になるため、I2C コントローラは I2C_SCLK で 3 サイクルを超えて SDA をホールドしません。

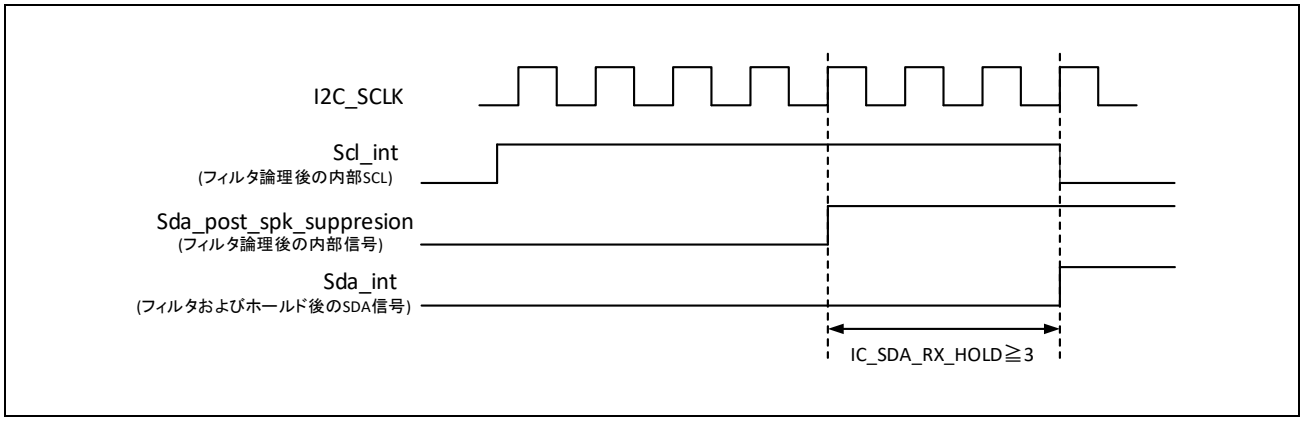


図 3.3　レシーバでの SDA ホールドタイミング

対応するスピードモード用にレジスタでプログラム可能な IC_SDA_RX_HOLD の最大値は、以下の式から導かれます。

標準モード

最大 IC_SDA_RX_HOLD＝IC_SS_SCL_HCNT － IC_FS_SPKLEN － 3

ファストモード

最大 IC_SDA_RX_HOLD＝IC_FS_SCL_HCNT － IC_FS_SPKLEN － 3

上記の最大値は、マスタモードに適用可能です。スレーブモードでは、IC_SDA_RX_HOLD が最大の SCL 立ち下がり時間（標準モードおよびファストモードでの t_f）を超えないことを確認してください。

3.6.3.2　トランスミッタでの SDA ホールドタイミング

IC_SDA_TX_HOLD レジスタは、I2C コントローラによって生成された SDA 信号のタイミングを変えるために使用できます。IC_SDA_TX_HOLD レジスタの各値は、I2C_SCLK 単位となります。

I2C コントローラがマスタモードで動作するとき、最小 $t_{HD;DAT}$ タイミングは I2C_SCLK で 1 周期です。よって、IC_SDA_TX_HOLD が 0 の値である場合でも、SCL をロジック 0 で出力後、I2C コントローラは I2C_SCLK で 1 サイクル間、SDA を出力します。IC_SDA_TX_HOLD のそれ以外のすべての値に対して、以下が適用されます。

- SCL をロジック 0 で出力した後、I2C_SCLK で IC_SDA_TX_HOLD サイクルの間 SDA の出力を行います。

I2C コントローラがスレーブモードで動作するとき、最小 $t_{HD;DAT}$ タイミングは I2C_SCLK で（IC_FS_SPKLEN＋7）周期です。本遅延により、SCL サンプルでの同期化およびスパイク抑制が可能になります。よって、IC_SDA_TX_HOLD に（IC_FS_SPKLEN＋7）未満の値がある場合でも、SCL がロジック 0 に遷移した後、I2C コントローラは I2C_SCLK で（IC_FS_SPKLEN＋7）サイクルの間 SDA を出力します。IC_SDA_TX_HOLD のそれ以外のすべての値に対して、以下が適用されます。

- SCL がロジック 0 に遷移した後、I2C_SCLK で IC_SDA_TX_HOLD サイクルの間 SDA の出力を行います。

備　考

プログラミングされる SDA ホールド時間は、どのタイミングでも SCL の Low 期間を超えることはできません。よって、プログラミングされる値は N_SCL_LOW-2 より大きくすることはできません。ここで N_SCL_LOW は I2C_SCLK サイクル単位の SCL 周期の Low 期間です。

第4章　Basic GPIO

4.1　概要

RZ/N1 には、3 つの Basic GPIO（BGPIO）モジュールがあります。

BGPIO は下記の機能に対応しています。

- 下記の最大 6 ポートが個別に設定可能です。
  - BGPIO1 ポート A、BGPIO1 ポート B、BGPIO2 ポート A、BGPIO2 ポート B、BGPIO3 ポート A、BGPIO3 ポート B
- 信号ごとに個別のデータレジスタおよびデータ方向レジスタ
- 独立して制御可能な信号ビット
- 選択型割り込みモードは BGPIO1 ポート A、BGPIO2 ポート A、および BGPIO3 ポート A のみ（ポートごとに 32 の割り込み）
  - 任意の信号ビットで外部信号を割り込み要因として受け入れるように本モードを設定可能です。
  - 割り込みの種類は、下記のいずれかに設定可能です。
    a）レベル（アクティブ High）
    b）レベル（アクティブ Low）
    c）立ち上がりエッジ
    d）立ち下がりエッジ
- BGPIO1、2、3 ポート A からのすべての割り込みは、プログラマブルラッパーによってルーティングされ、Cortex®-A7 および Cortex®-M3 に直接ルーティングされる 8 つの割り込みを抽出します。
- 専用ラッパー回路付きのトリガ同期動作は、割り込みによって制御されるリアルタイム動作を可能にします。GPIO 信号のステータスは、オンチップ周辺機能からの割り込みと同期して更新されます。トリガ同期制御モードは、レジスタ設定で有効／無効にすることが可能です。

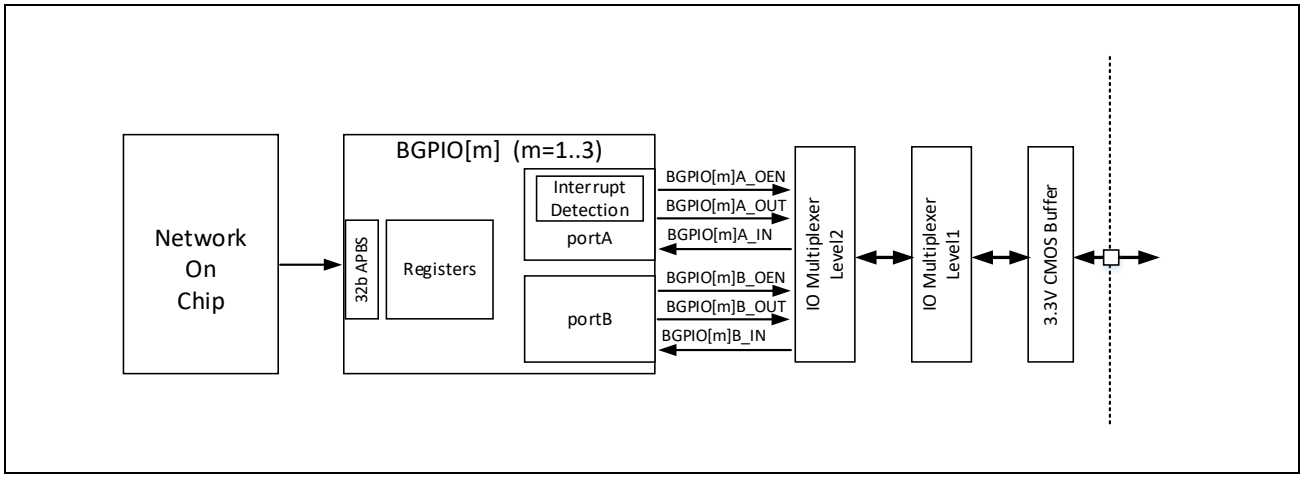


図 4.1　BGPIO の概要

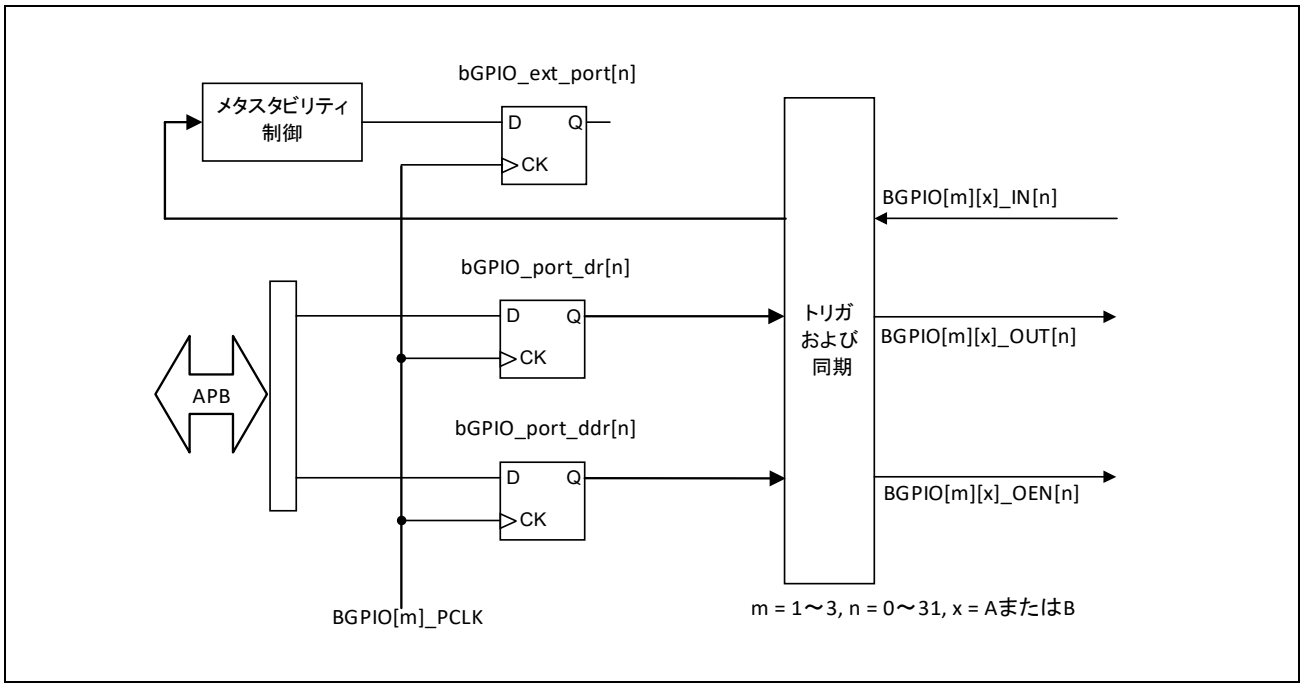


図 4.2　BGPIO の概要（ポート管理機能）

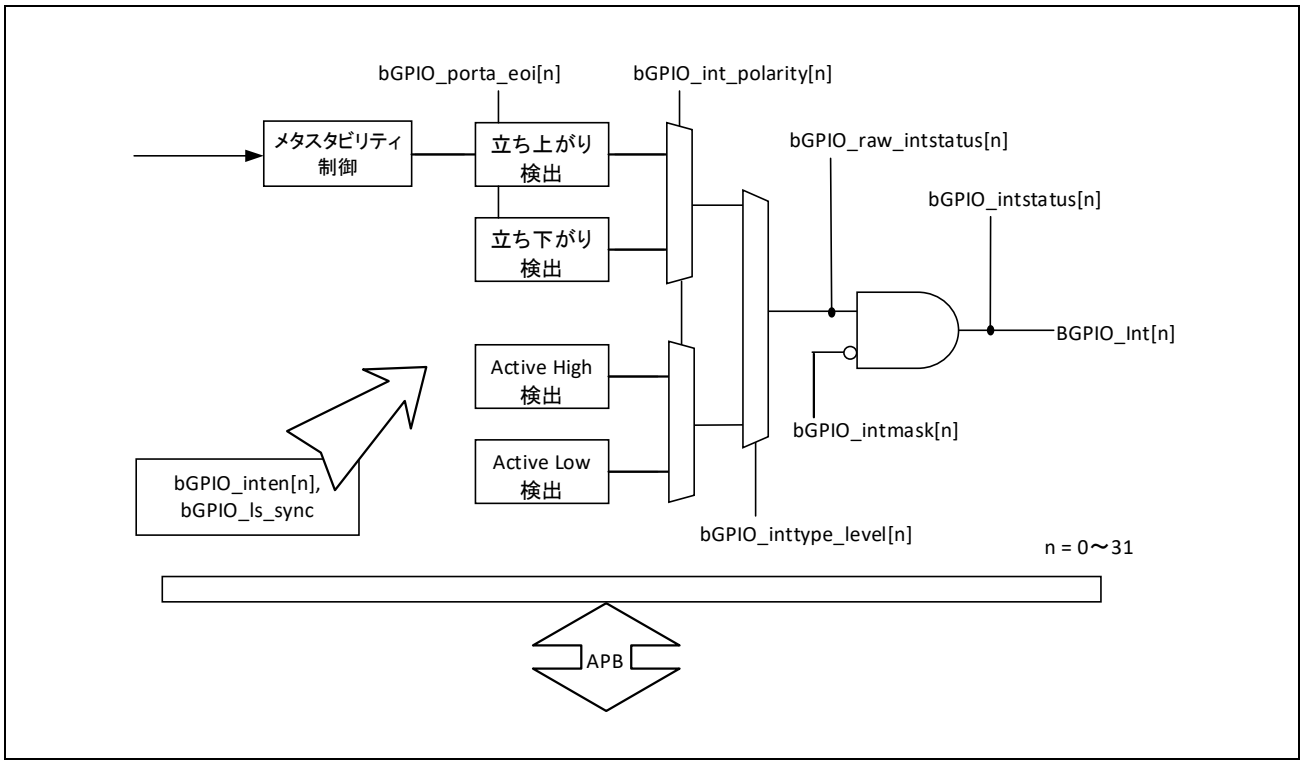


図 4.3　BGPIO の概要（割り込み管理機能）

4.2　信号インタフェース

表 4.1　BGPIO 信号インタフェース

信号名	入力 出力	説明
クロック		
BGPIO[m]_PCLK	入力	内部バスクロック（APB）
割り込み		
BGPIO[m]_Int[31:0]	出力	ポート A のレベル検出割り込み出力、アクティブ High
外部 GPIO 端子インタフェース信号		
BGPIO1A_OEN[31:0] BGPIO1B_OEN[31:0] BGPIO2A_OEN[31:0] BGPIO2B_OEN[31:0] BGPIO3A_OEN[31:0] BGPIO3B_OEN[9:0]	出力	GPIO 出力イネーブル
BGPIO1A_IN[31:0] BGPIO1B_IN[31:0] BGPIO2A_IN[31:0] BGPIO2B_IN[31:0] BGPIO3A_IN[31:0] BGPIO3B_IN[9:0]	入力	GPIO 入力
BGPIO1A_OUT[31:0] BGPIO1B_OUT[31:0] BGPIO2A_OUT[31:0] BGPIO2B_OUT[31:0] BGPIO3A_OUT[31:0] BGPIO3B_OUT[9:0]	出力	GPIO 出力
GPIO トリガコントロール信号		
CFG_GPIOT_PTEN1A[31:0] CFG_GPIOT_PTEN1B[31:0] CFG_GPIOT_PTEN2A[31:0] CFG_GPIOT_PTEN2B[31:0] CFG_GPIOT_PTEN3A[31:0] CFG_GPIOT_PTEN3B[9:0]	入力	GPIO トリガイネーブル
GPIO_TRIGGER[3:0]	入力	GPIO トリガ信号

備考　m＝1～3

出力イネーブル、入力、出力は、IO マルチプレクシングロジックによってルーティングされます。

4.3　レジスタマップ

4.3.1　レジスタマップ BGPIO1

表 4.2　Basic GPIO1 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5000 B000h	rGPIO_swporta_dr	GPIO ポート A データ出力レジスタ
5000 B004h	rGPIO_swporta_ddr	GPIO ポート A データ方向レジスタ
5000 B00Ch	rGPIO_swportb_dr	GPIO ポート B データ出力レジスタ
5000 B010h	rGPIO_swportb_ddr	GPIO ポート B データ方向レジスタ
5000 B030h	rGPIO_inten	GPIO ポート A 割り込み許可レジスタ
5000 B034h	rGPIO_intmask	GPIO ポート A 割り込みマスクレジスタ
5000 B038h	rGPIO_inttype_level	GPIO ポート A 割り込みレベルレジスタ
5000 B03Ch	rGPIO_int_polarity	GPIO ポート A 割り込み極性レジスタ
5000 B040h	rGPIO_intstatus	GPIO ポート A 割り込みステータス
5000 B044h	rGPIO_raw_intstatus	GPIO ポート A ロウ（raw）割り込みステータス（マスク前）
5000 B04Ch	rGPIO_porta_eoi	GPIO ポート A 割り込みクリアレジスタ
5000 B050h	rGPIO_ext_porta	GPIO ポート A データ入力レジスタ
5000 B054h	rGPIO_ext_portb	GPIO ポート B データ入力レジスタ
5000 B060h	rGPIO_ls_sync	GPIO ポート A レベル検出同期化イネーブルレジスタ

4.3.2　レジスタマップ BGPIO2

表 4.3　Basic GPIO2 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5000 C000h	rGPIO_swporta_dr	GPIO ポート A データ出力レジスタ
5000 C004h	rGPIO_swporta_ddr	GPIO ポート A データ方向レジスタ
5000 C00Ch	rGPIO_swportb_dr	GPIO ポート B データ出力レジスタ
5000 C010h	rGPIO_swportb_ddr	GPIO ポート B データ方向レジスタ
5000 C030h	rGPIO_inten	GPIO ポート A 割り込み許可レジスタ
5000 C034h	rGPIO_intmask	GPIO ポート A 割り込みマスクレジスタ
5000 C038h	rGPIO_inttype_level	GPIO ポート A 割り込みレベルレジスタ
5000 C03Ch	rGPIO_int_polarity	GPIO ポート A 割り込み極性レジスタ
5000 C040h	rGPIO_intstatus	GPIO ポート A 割り込みステータス
5000 C044h	rGPIO_raw_intstatus	GPIO ポート A ロウ（raw）割り込みステータス（マスク前）
5000 C04Ch	rGPIO_porta_eoi	GPIO ポート A 割り込みクリアレジスタ
5000 C050h	rGPIO_ext_porta	GPIO ポート A データ入力レジスタ
5000 C054h	rGPIO_ext_portb	GPIO ポート B データ入力レジスタ
5000 C060h	rGPIO_ls_sync	GPIO ポート A レベル検出同期化イネーブルレジスタ

4.3.3　レジスタマップ BGPIO3

表 4.4　Basic GPIO3 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5000 D000h	rGPIO_swporta_dr	GPIO ポート A データ出力レジスタ
5000 D004h	rGPIO_swporta_ddr	GPIO ポート A データ方向レジスタ
5000 D00Ch	rGPIO_swportb_dr	GPIO ポート B データ出力レジスタ
5000 D010h	rGPIO_swportb_ddr	GPIO ポート B データ方向レジスタ
5000 D030h	rGPIO_inten	GPIO ポート A 割り込み許可レジスタ
5000 D034h	rGPIO_intmask	GPIO ポート A 割り込みマスクレジスタ
5000 D038h	rGPIO_inttype_level	GPIO ポート A 割り込みレベルレジスタ
5000 D03Ch	rGPIO_int_polarity	GPIO ポート A 割り込み極性レジスタ
5000 D040h	rGPIO_intstatus	GPIO ポート A 割り込みステータス
5000 D044h	rGPIO_raw_intstatus	GPIO ポート A ロウ（raw）割り込みステータス（マスク前）
5000 D04Ch	rGPIO_porta_eoi	GPIO ポート A 割り込みクリアレジスタ
5000 D050h	rGPIO_ext_porta	GPIO ポート A データ入力レジスタ
5000 D054h	rGPIO_ext_portb	GPIO ポート B データ入力レジスタ
5000 D060h	rGPIO_ls_sync	GPIO ポート A レベル検出同期化イネーブルレジスタ

4.4　レジスタの説明

4.4.1　rGPIO_swporta_dr ― GPIO ポート A データ出力レジスタ

アドレス　5000 B000h (BGPIO1)
5000 C000h (BGPIO2)
5000 D000h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_port_dr															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_port_dr															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.5　rGPIO_swporta_dr レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_port_dr	各 n ビット（n＝0～31）に対して、対応するポートのデータ方向ビットが出力モードに設定されている場合、本レジスタに書き込まれた値は外部端子 BGPIO[m]A[n]での出力になります。 読み出される値は本レジスタに最後に書き込まれた値と等しくなります。	R/W

4.4.2　rGPIO_swporta_ddr ― GPIO ポート A データ方向レジスタ

アドレス　5000 B004h (BGPIO1)
5000 C004h (BGPIO2)
5000 D004h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_port_ddr															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_port_ddr															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.6　rGPIO_swporta_ddr レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_port_ddr	各 n ビット（n＝0～31）に対して、本レジスタに書き込まれた値は、外部端子 BGPIO[m]A[n]で対応するデータビットの方向を個別に制御します。 1'b0：入力（リセット後のデフォルト） 1'b1：出力	R/W

4.4.3　rGPIO_swportb_dr — GPIO ポート B データ出力レジスタ

アドレス　5000 B00Ch (BGPIO1)
5000 C00Ch (BGPIO2)
5000 D00Ch (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_port_dr															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_port_dr															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.7　rGPIO_swportb_dr レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_port_dr	各 n ビット（n＝0～31）に対して、対応するポートのデータ方向ビットが出力モードに設定されている場合、本レジスタに書き込まれた値は外部端子 BGPIO[m]B[n]での出力になります。 読み出される値は本レジスタに最後に書き込まれた値と等しくなります。	R/W

4.4.4　rGPIO_swportb_ddr — GPIO ポート B データ方向レジスタ

アドレス　5000 B010h (BGPIO1)
5000 C010h (BGPIO2)
5000 D010h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_port_ddr															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_port_ddr															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.8　rGPIO_swportb_ddr レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_port_ddr	各 n ビット（n＝0～31）に対して、本レジスタに書き込まれた値は、外部端子 BGPIO[m]B[n]で対応するデータビットの方向を個別に制御します。 1'b0：入力（リセット後のデフォルト） 1'b1：出力	R/W

4.4.5　rGPIO_inten ― GPIO ポート A 割り込み許可レジスタ

アドレス　5000 B030h (BGPIO1)
5000 C030h (BGPIO2)
5000 D030h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_inten															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_inten															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.9　rGPIO_inten レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_inten	ポート A の各 n ビット（n＝0～31）に対して、割り込み用に設定することを許可します。デフォルトでは、割り込みの生成は無効です。 本レジスタのビットに 1 が書き込まれている場合は、ポート A の対応するビットが割り込みになるように設定され、そうでない場合は、ポート A は通常の GPIO 信号として動作します。 対応するデータ方向レジスタが出力に設定されている場合、ポート A の対応するビットで割り込みは禁止です。 1'b0：ポート A を通常の GPIO 信号として設定する（デフォルト） 1'b1：ポート A を割り込みとして設定する	R/W

4.4.6　rGPIO_intmask — GPIO ポート A 割り込みマスクレジスタ

アドレス　5000 B034h (BGPIO1)
5000 C034h (BGPIO2)
5000 D034h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_intmask															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_intmask															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.10　rGPIO_intmask レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_intmask	各 n ビット（n＝0～31）に対して、ポート A での割り込みを制御します。デフォルトでは、すべての割り込みはマスク解除されています。本レジスタのビットに 1 が書き込まれている場合は、本信号の割り込み生成機能をマスクしますが、そうでない場合は、マスクされません。マスク解除されたステータスは、マスク後も結果ステータスと同じように読み出すことが可能です。 1’b0：割り込み許可（デフォルト） 1’b1：割り込み禁止	R/W

4.4.7　rGPIO_inttype_level — GPIO ポート A 割り込みレベルレジスタ

アドレス　5000 B038h (BGPIO1)
5000 C038h (BGPIO2)
5000 D038h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_inttype_levell															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_inttype_levell															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.11　rGPIO_inttype_level レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_inttype_levell	各 n ビット（n＝0～31）に対して、ポート A で発生可能な割り込みの種類を制御します。 本レジスタのビットに 0 が書き込まれている場合は、割り込みの種類はレベル検出に設定され、そうでない場合はエッジ検出です。 1’b0：レベル検出（デフォルト） 1’b1：エッジ検出	R/W

4.4.8　rGPIO_int_polarity ― GPIO ポート A 割り込み極性レジスタ

アドレス　5000 B03Ch (BGPIO1)
5000 C03Ch (BGPIO2)
5000 D03Ch (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_int_polarity															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_int_polarity															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.12　rGPIO_int_polarity レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_int_polarity	各 n ビット（n＝0～31）に対して、ポート A の入力で発生可能なエッジまたはレベル検出の極性を制御します。本レジスタのビットに 0 が書き込まれている場合、割り込みの種類は立ち下がりエッジまたはアクティブ Low 検出に設定されますが、そうでない場合は、立ち上がりエッジまたはアクティブ High 検出です。 1’b0：アクティブ Low（デフォルト） 1’b1：アクティブ High	R/W

4.4.9　rGPIO_intstatus — GPIO ポート A 割り込みステータス

アドレス　5000 B040h (BGPIO1)
5000 C040h (BGPIO2)
5000 D040h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_intstatus															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_intstatus															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

表 4.13　rGPIO_intstatus レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_intstatus	各 n ビット（n＝0～31）に対するポート A の割り込みステータス 1’b0：割り込みなし 1’b1：割り込み発生	R

4.4.10　rGPIO_raw_intstatus — GPIO ポート A ロウ（raw）割り込みステータス（マスク前）

アドレス　5000 B044h (BGPIO1)
5000 C044h (BGPIO2)
5000 D044h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_raw_intstatus															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_raw_intstatus															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.14　rGPIO_raw_intstatus レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_raw_intstatus	各 n ビット（n＝0～31）に対するポート A のロウ（raw）割り込みステータス（ビットマスク前） 1’b0：割り込み要求なし 1’b1：割り込み要求あり	R

4.4.11　rGPIO_porta_eoi ― GPIO ポート A 割り込みクリアレジスタ

アドレス　5000 B04Ch (BGPIO1)
5000 C04Ch (BGPIO2)
5000 D04Ch (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_porta_eoi															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_porta_eoi															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.15　rGPIO_porta_eoi レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_porta_eoi	各 n ビットに対して（n＝0～31）、ポート A でのエッジタイプ割り込みのクリアを制御します。本レジスタの対応するビットに 1 が書き込まれると、割り込みはクリアされます。 ポート A が割り込み用に設定されていない場合は、割り込みはクリアされます。 1'b0：割り込みクリアなし（デフォルト） 1'b1：割り込みをクリア	W

4.4.12　rGPIO_ext_porta ― GPIO ポート A データ入力レジスタ

アドレス　5000 B050h (BGPIO1)
5000 C050h (BGPIO2)
5000 D050h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_ext_port															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_ext_port															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.16　rGPIO_ext_porta レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_ext_port	各 n ビットに対して（n＝0～31）、ポートが入力として設定されている場合、本レジスタの読み出しは、外部端子 BGPIO[m]A[n]の値を読み出します。 ポートのデータ方向が出力に設定されている場合は、BGPIO がトリガモードの場合を除いて、本レジスタの読み出しはポートのデータ出力レジスタを読み出します。	R

4.4.13　rGPIO_ext_portb ― GPIO ポート B データ入力レジスタ

アドレス　5000 B054h (BGPIO1)
5000 C054h (BGPIO2)
5000 D054h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bGPIO_ext_port															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bGPIO_ext_port															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.17　rGPIO_ext_portb レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bGPIO_ext_port	各 n ビットに対して（n＝0～31）、ポートが入力として設定されている場合、本レジスタの読み出しは、外部端子 BGPIO[m]B[n]の値を読み出します。 ポートのデータ方向が出力に設定されている場合は、BGPIO がトリガモードの場合を除いて、本レジスタの読み出しはポートのデータ出力レジスタを読み出します。	R

4.4.14　rGPIO_ls_sync ― GPIO ポート A レベル検出同期化イネーブルレジスタ

アドレス　5000 B060h (BGPIO1)
5000 C060h (BGPIO2)
5000 D060h (BGPIO3)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bGPIO_ ls_sync
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 4.18　rGPIO_ls_sync レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	0 が読み出されます。	R
b0	bGPIO_ls_sync	本レジスタに 1 を書き込むと、すべてのレベル検出割り込みがクロックに同期されます。 1'b0：BGPIO_PCLK との同期なし（デフォルト） 1'b1：BGPIO_PCLK との同期あり **注意）** 割り込みコントローラへの割り込みラインでのグリッチを避けるために、ファームウェアは本ビットを 1 に設定する必要があります。	R/W

4.5　動作説明

4.5.1　主要機能ブロックの説明

4.5.1.1　データおよび制御フロー

BGPIO は、外部出力データおよび外部入出力の方向を制御します。メモリマップされたレジスタを使用して外部入力データの読み出しも可能です。

ポート A およびポート B それぞれの各ビットは個別に制御可能です。信号のデータおよび制御フローを「**図 4.2 BGPIO の概要（ポート管理機能）**」に示します。

- ポート方向制御

  rGPIO_swporta_ddr.bGPIO_port_ddr[n]および rGPIO_swportb_ddr.bGPIO_port_ddr[n]

- ポート出力データ

  rGPIO_swporta_dr.bGPIO_port_dr[n]および rGPIO_swportb_dr.bGPIO_port_dr[n]

- ポート入力データ（読み出しのみ）

  rGPIO_ext_porta.bGPIO_ext_port[n]および rGPIO_ext_portb.bGPIO_ext_port[n]

注　意

有効な BGPIO ポートおよびレジスタはデバイス（RZ/N1D、N1S、または N1L）によって異なります。
たとえば、RZ/N1D-400 において BGPIO3 ポート B に関連するレジスタはビット 9～0 が有効です。

4.5.1.2　割り込み（ポート A のみ）

ポート A は、任意の信号ビットで外部信号を割り込み要因として受け入れるように設定可能です。割り込みの種類は、下記のいずれかに設定可能です。

- レベル（アクティブ High）
- レベル（アクティブ Low）
- 立ち上がりエッジ
- 立ち下がりエッジ

ポート A の各 n ビット（n＝0～31）に対して、割り込みは bGPIO_intmask[n]レジスタをプログラミングすることでマスク可能です。割り込みステータスは、マスク前（ロウ（raw）ステータスと呼ばれる）およびマスク後を読み出し可能です。

ポート A が割り込み用に設定されている場合、データ方向を「入力」に設定する必要があり、ソフトウェアで割り込みをラッチできるように設定します。データ方向レジスタを出力モードに再プログラミングする場合、保留中の割り込みは失われません。ただし、新しい割り込みは生成されません。

「**図 4.3 BGPIO の概要（割り込み管理機能）**」に割り込みの生成方法およびデータフロー方法を示します。

エッジ検出された割り込みに対して、対応するビットの bGPIO_porta_eoi[n]ビットに 1 を書き込むことで割り込みはクリアされます。本書き込みは、割り込みステータスおよびロウ（raw）ステータスレジスタもクリアします。bGPIO_porta_eoi[n]レジスタに書き込む前に、割り込み要因をクリアすることが推奨されます。bGPIO_porta_eoi[n]ビットへの書き込みはレベル検出割り込みでは影響ありません。

レベル検出割り込みが割り込みを発生させる場合、ソフトウェアは、割り込み要因が消えるまで、bGPIO_raw_intstatus[n]レジスタをポーリングすることが可能です。そうでない場合は、割り込みをマスクするために bGPIO_intmask[n]レジスタに書き込むことが可能です。

ソフトウェアが複数の保留中の割り込み要求を見つけるために bGPIO_intstatus[n]レジスタを読み出す場合、それらの保留中割り込み要求の優先順位付けはプロセッサ次第です。

bGPIO_porta_eoi[n]レジスタに複数の 1 を書き込むことで同時クリア可能なエッジ検出割り込みの数に制限はありません。

新しい割り込みが検出されるのと同時に、ソフトウェアが既存の割り込みをクリアするために 1 を書き込む場合があります。そのような場合は、割り込みクリアレジスタへの書き込みは、最初の割り込みしかクリアしません。割り込みのセットの方がクリアより優先順位が高いため、2 番目の割り込みは失われません。

4.5.1.3　Cortex-A7 および Cortex-M3 でルーティングされるプログラマブル割り込み

BGPIO1、2、3 ポート A からのすべての割り込みは、Cortex-A7 および Cortex-M3 へ直接ルーティングされる 8 つの割り込み GPIO_Int[7:0]を抽出するために、システムコントロールで GPIO 割り込みマルチプレクサによって選択されます。マルチプレクサは ConfigSys2 の rGPIOs_Level2_Gpio_Int_[n]によって設定されます。

各割り込み GPIO_Int[n]（n＝0～7）に対して：

- 割り込み要因を、32×3 の考えられる割り込み要因（ポート A のみ）から選択します。
  - BGPIO1_Int[31:0]、BGPIO2_Int[31:0]、または BGPIO3_Int[31:0]

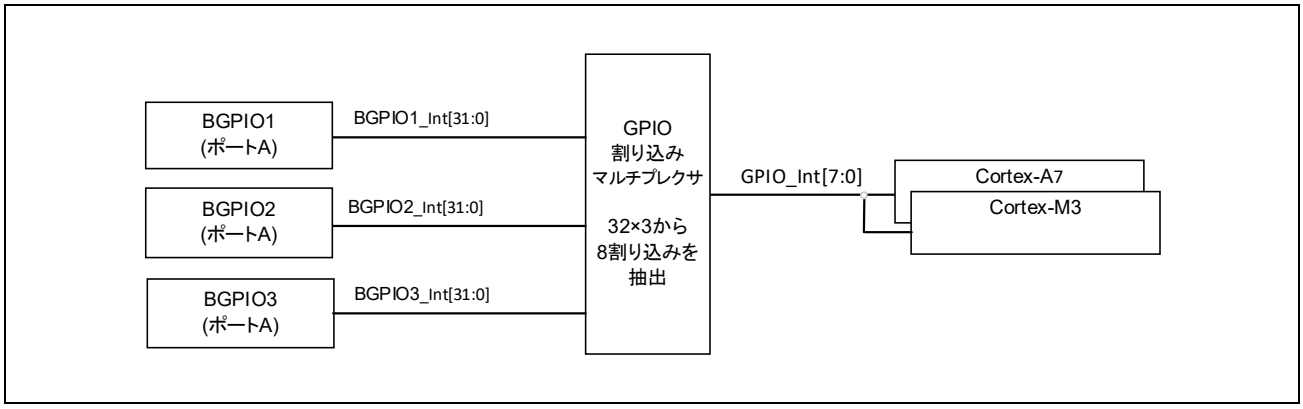


図 4.4　Cortex-A7 および Cortex-M3 でルーティングされるプログラマブル割り込み

4.5.1.4　トリガ同期式動作

専用ラッパー回路で管理されるトリガ同期式動作は、システムコントローラで設定される割り込みソースによって制御され、リアルタイム動作を可能にします。GPIO 端子の GPIO データ信号（入力、出力）は、オンチップ周辺機能からの割り込みソースに同期して更新されます。トリガ同期制御モードは、レジスタ設定経由で有効／無効にすることが可能です。GPIO 出力イネーブルは常に直接接続されていることに注意してください（同期機能なし）。

「**図 4.5 同期方式およびイベントでのキャプチャ**」を参照してください。

- トリガ同期制御モード 無効（トランスペアレントモード）：
  - 入力／出力信号および出力イネーブル信号は直接接続され、キャプチャもラッチもなく、GPIO データ信号は BGPIO 信号ごとに直接駆動されます。
  - システムコントロールの CFG_GPIOT_PTEN_mj レジスタにあるビット[n]を“0”にします。
    a）m：1、2、3 → リファレンスポート BGPIO1、BGPIO2、BGPIO3 によります
    b）j：A、B → 現在のポートが A または B によります
    c）n＝0～31 → 現在のポートアドレス形式のビット番号
- トリガ同期制御モード 有効：
  - BGPIO 入力信号および出力信号は、システムコントローラ内で選択されたオンチップ周辺機能からの割り込み（GPIO_TRIGGER[3:0]信号）に同期して更新されます。
  - BGPIO_PCLK に同期される出力および入力信号のキャプチャは GPIO_TRIGGER[3:0]信号の立ち上がりエッジ検出によって開始されます。
  - GPIO_TRIGGER[3:0]信号の立ち上がりエッジが検出されない場合、BGPIO 入力信号と出力信号は保持され、最後のキャプチャから変更はありません。
  - システムコントロールの CFG_GPIOT_PTEN_mj レジスタにあるビット[n]を“1”にします。
  - 出力イネーブル信号ではキャプチャおよびラッチ機能は使用できません（直接接続）。

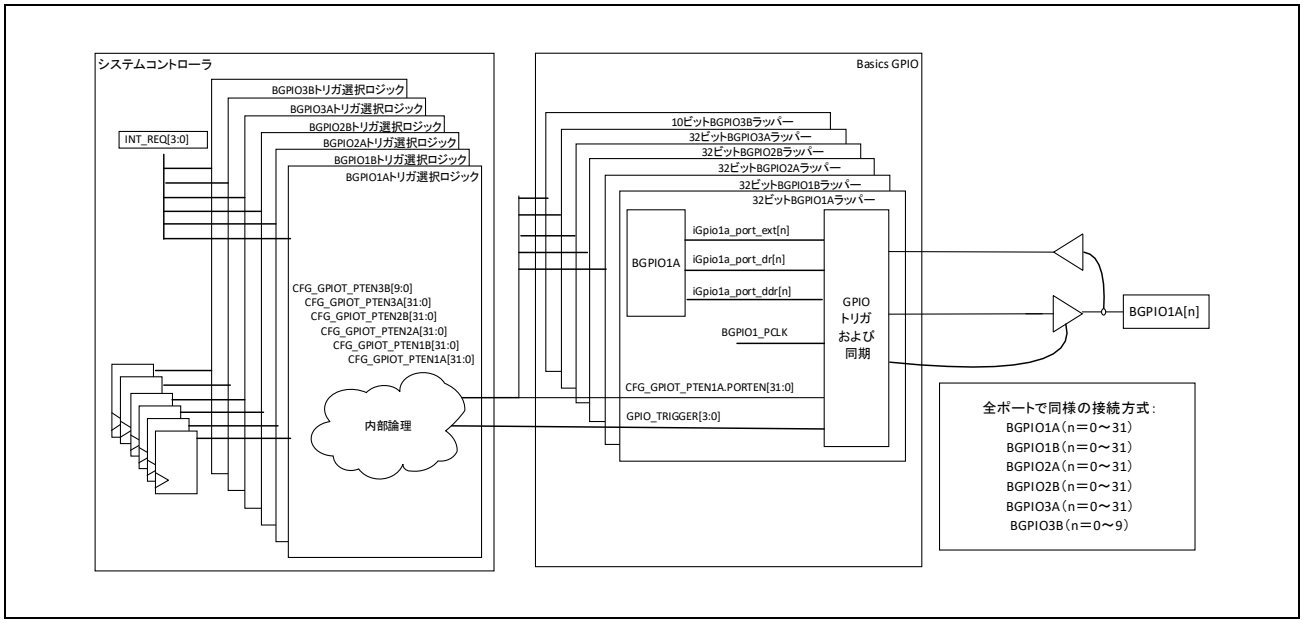


図 4.5　同期方式およびイベントでのキャプチャ

表 4.19　トリガ同期式動作割り当てライン

BGPIO 信号名	GPIO トリガ 信号名	PTEN 信号名
BGPIO1A		
BGPIO1A[7:0]	GPIO_TRIGGER[0]	CFG_GPIOT_PTEN1A.PORTEN[7:0]
BGPIO1A[15:8]	GPIO_TRIGGER[1]	CFG_GPIOT_PTEN1A.PORTEN[15:8]
BGPIO1A[23:16]	GPIO_TRIGGER[2]	CFG_GPIOT_PTEN1A.PORTEN[23:16]
BGPIO1A[31:24]	GPIO_TRIGGER[3]	CFG_GPIOT_PTEN1A.PORTEN[31:24]
BGPIO1B		
BGPIO1B[7:0]	GPIO_TRIGGER[0]	CFG_GPIOT_PTEN1B.PORTEN[7:0]
BGPIO1B[15:8]	GPIO_TRIGGER[1]	CFG_GPIOT_PTEN1B.PORTEN[15:8]
BGPIO1B[23:16]	GPIO_TRIGGER[2]	CFG_GPIOT_PTEN1B.PORTEN[23:16]
BGPIO1B[31:24]	GPIO_TRIGGER[3]	CFG_GPIOT_PTEN1B.PORTEN[31:24]
BGPIO2A		
BGPIO2A[7:0]	GPIO_TRIGGER[0]	CFG_GPIOT_PTEN2A.PORTEN[7:0]
BGPIO2A[15:8]	GPIO_TRIGGER[1]	CFG_GPIOT_PTEN2A.PORTEN[15:8]
BGPIO2A[23:16]	GPIO_TRIGGER[2]	CFG_GPIOT_PTEN2A.PORTEN[23:16]
BGPIO2A[31:24]	GPIO_TRIGGER[3]	CFG_GPIOT_PTEN2A.PORTEN[31:24]
BGPIO2B		
BGPIO2B[7:0]	GPIO_TRIGGER[0]	CFG_GPIOT_PTEN2B.PORTEN[7:0]
BGPIO2B[15:8]	GPIO_TRIGGER[1]	CFG_GPIOT_PTEN2B.PORTEN[15:8]
BGPIO2B[23:16]	GPIO_TRIGGER[2]	CFG_GPIOT_PTEN2B.PORTEN[23:16]
BGPIO2B[31:24]	GPIO_TRIGGER[3]	CFG_GPIOT_PTEN2B.PORTEN[31:24]
BGPIO3A		
BGPIO3A[7:0]	GPIO_TRIGGER[0]	CFG_GPIOT_PTEN3A.PORTEN[7:0]
BGPIO3A[15:8]	GPIO_TRIGGER[1]	CFG_GPIOT_PTEN3A.PORTEN[15:8]
BGPIO3A[23:16]	GPIO_TRIGGER[2]	CFG_GPIOT_PTEN3A.PORTEN[23:16]
BGPIO3A[31:24]	GPIO_TRIGGER[3]	CFG_GPIOT_PTEN3A.PORTEN[31:24]
BGPIO3B		
BGPIO3B[7:0]	GPIO_TRIGGER[0]	CFG_GPIOT_PTEN3B.PORTEN[7:0]
BGPIO3B[9:8]	GPIO_TRIGGER[1]	CFG_GPIOT_PTEN3B.PORTEN[10:8]

表 4.20　トリガ同期式動作基本機能

GPIO トリガモード	GPIO トリガ	BGPIO 信号のステータス
CFG_GPIOT_PTEN[m]A CFG_GPIOT_PTEN[m]B	GPIO_TRIGGER[3:0]	BGPIO[m]A BGPIO[m]B
1’b0	1’bx	トランスペアレントモード
1’b1	立ち上がりエッジ検出時	トリガ同期制御モード、イベントキャプチャ
1’b1	立ち上がり検出なし	トリガ同期制御モード、最後のキャプチャイベントから変更なし

注：　m＝1～3

4.6　使用上の注意事項

4.6.1　プログラミング上の注意事項

割り込み機能、エッジ検出またはレベル検出の割り込み、および割り込み極性についての BGPIO レジスタのプログラミングは、割り込みコントローラへの割り込みライン上のグリッチを防ぐために、BGPIO1 ポート A、BGPIO2 ポート A、および BGPIO3 ポート A の割り込みを許可する前に完了させる必要があります。

注　項

- グリッチを防ぐためには、rGPIO_ls_sync レジスタで bGPIO_ls_sync ビットを 1 にプログラムします。
- 割り込みクリアレジスタへの書き込みは、エッジ検出割り込みをクリアし、レベル検出割り込みに影響はありません。

第5章　タイマブロック

ペリフェラルグループ 1（PG1）サブシステムには、8 つのサブタイマを持つ 2 つのタイマブロックがあります。

5.1　概要

- 2 ユニット
- 16 ビットプログラマブルサブタイマ×6（0～5）
- 32 ビットプログラマブルサブタイマ×2（6～7）
- 各サブタイマは単一の割り込みを生成
- 2 つの時間ベースから選択可能なプリスケーラ
  - 25MHz または 1MHz
- 2 つの機能モード
  - 自動リロードモード
    割り込みを事前に設定された時間で起動。カウンタは自動的にクリアされ再開します。
  - シングルショットモード
    割り込みを事前に設定された時間で起動。カウンタは停止後、無効になります。
- DMA 接続
  - DMAC フローコントローラモードを使用
  - 割り込みタイミングで DMA トランザクションを開始
  - サブタイマ（6～7）でのみ使用可能

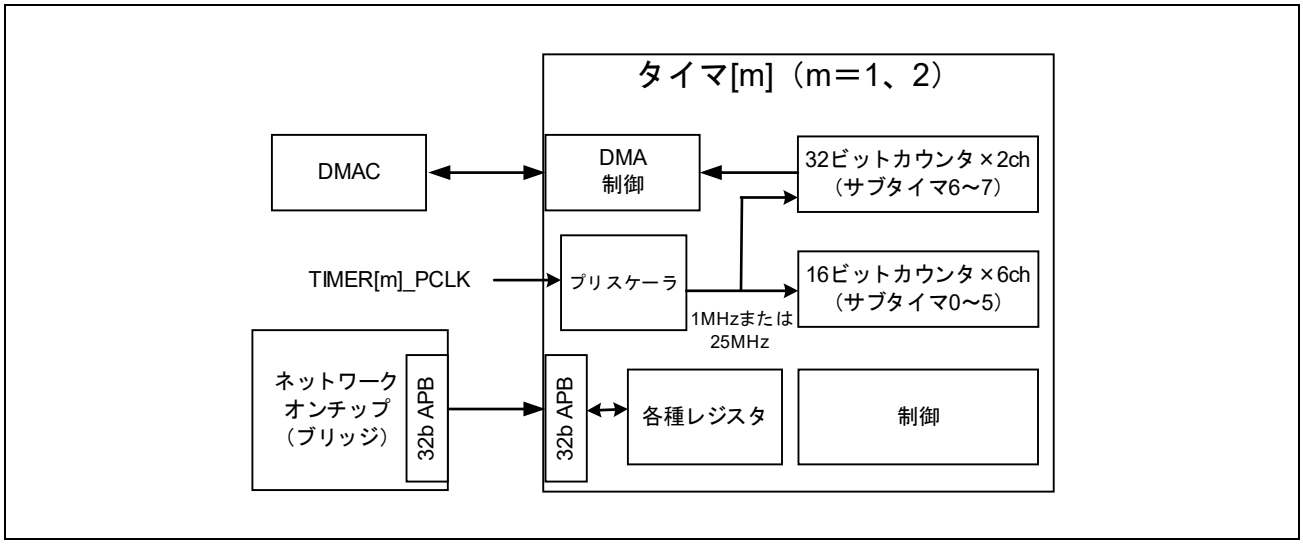


図 5.1　タイマインタフェースおよび接続

5.2　信号インタフェース

信号名	入力 出力	説明
クロック		
TIMER[m]_PCLK	入力	内部バスクロック（APB）およびプリスケーラ入力クロック（25MHz）
割り込み		
TIMER[m]_Int[7:0]	出力	各サブタイマ用レベル検出割り込み出力（アクティブ High）

備考　m＝1 または 2

本章では、インデックスを省いた記述スタイルを使用します。

例）TIMER_PCLK

5.3　レジスタマップ

5.3.1　タイマ 1 レジスタマップ

表 5.1　タイマ 1 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5100 1000h＋20h×n	rTimerLoadCount_[n]（n＝0～7）	サブタイマ[n]のプリセット値
5100 1004h＋20h×n	rTimerCurrentCount_[n]（n＝0～7）	サブタイマ[n]の現在値
5100 1008h＋20h×n	rTimerControl_[n]（n＝0～7）	サブタイマ[n]の制御モード
5100 100Ch＋20h×n	rTimerClearInt_[n]（n＝0～7）	サブタイマ[n]の割り込みをクリア
5100 1010h＋20h×n	rTimerStatusInt0_[n]（n＝0～7）	サブタイマ[n]のマスク前の割り込み状態
5100 1014h＋20h×n	rTimerStatusInt1_[n]（n＝0～7）	サブタイマ[n]のマスク後の割り込み状態
5100 1100h	rTimerAllClearInt	すべての割り込みをクリア
5100 1104h	rTimerAllStatusInt0	マスク前のすべての割り込み状態
5100 1108h	rTimerAllStatusInt1	マスク後のすべての割り込み状態
5100 110Ch	rTimer_DMA_Pending	タイマ DMA 要求状態
5100 1110h	rTimer_DMA_PendingOvf	タイマ DMA オーバーフロー状態
5100 1114h	rTimer_DMA_PendingClrOvf	タイマ DMA オーバーフロークリア

5.3.2　タイマ 2 レジスタマップ

表 5.2　タイマ 2 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5100 2000h＋20h×n	rTimerLoadCount_[n]（n＝0～7）	サブタイマ[n]のプリセット値
5100 2004h＋20h×n	rTimerCurrentCount_[n]（n＝0～7）	サブタイマ[n]の現在値
5100 2008h＋20h×n	rTimerControl_[n]（n＝0～7）	サブタイマ[n]の制御モード
5100 200Ch＋20h×n	rTimerClearInt_[n]（n＝0～7）	サブタイマ[n]の割り込みをクリア
5100 2010h＋20h×n	rTimerStatusInt0_[n]（n＝0～7）	サブタイマ[n]のマスク前の割り込み状態
5100 2014h＋20h×n	rTimerStatusInt1_[n] n＝0～7）	サブタイマ[n]のマスク後の割り込み状態
5100 2100h	rTimerAllClearInt	すべての割り込みをクリア
5100 2104h	rTimerAllStatusInt0	マスク前のすべての割り込み状態
5100 2108h	rTimerAllStatusInt1	マスク後のすべての割り込み状態
5100 210Ch	rTimer_DMA_Pending	タイマ DMA 要求状態
5100 2110h	rTimer_DMA_PendingOvf	タイマ DMA オーバーフロー状態
5100 2114h	rTimer_DMA_PendingClrOvf	タイマ DMA オーバーフロークリア

5.4　レジスタの説明

5.4.1　rTimerLoadCount_[n] ― サブタイマ[n]（n＝0～5）のプリセット値

アドレス　5100 1000h＋20h×n（タイマ 1）
5100 2000h＋20h×n（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bTimerLoadCount															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.3　rTimerLoadCount_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bTimerLoadCount	プリセット値 タイマは 0 からプリセット値までカウントアップします。 **注意）** ● プリセット値は、0 以外とします。プリセット値が 0 の場合、タイマは動作しません。 ● タイマ動作時（bTimerEnable を“1”にセット）に CPU が新しいプリセット値を書き込むと、ただちに認識されます。 ● 自動リロードモード時、周期サイクルは（bTimerLoadCount＋1）となります。 タイマ動作時ファームウェアが bTimerLoadCount を変更した場合は、「**5.6 使用上の注意事項**」を参照してください。	R/W

5.4.2　rTimerLoadCount_[n] ― サブタイマ[n]（n＝6～7）のプリセット値

アドレス　5100 1000h＋20h×n（タイマ 1）
5100 2000h＋20h×n（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bTimerLoadCount															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bTimerLoadCount															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.4　rTimerLoadCount_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bTimerLoadCount	プリセット値 タイマは 0 からプリセット値までカウントアップします。 **注意）** ● プリセット値は、0 以外とします。プリセット値が 0 の場合、タイマは動作しません。 ● タイマ動作時（bTimerEnable を“1”にセット）に CPU が新しいプリセット値を書き込むと、ただちに認識されます。 ● 自動リロードモード時、周期サイクルは（bTimerLoadCount＋1）となります。 タイマ動作時ファームウェアが bTimerLoadCount を変更した場合は、「**5.6 使用上の注意事項**」を参照してください。	R/W

5.4.3　rTimerCurrentCount_[n] ― サブタイマ[n]（n＝0～5）の現在値

アドレス　5100 1004h＋20h×n（タイマ 1）
5100 2004h＋20h×n（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bTimerCurrentCount															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.5　rTimerCurrentCount_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bTimerCurrentCount	タイマの現在値 あるタイマの現在値がプリセット値と同じ場合、 　割り込みがトリガされます。 　bTimerEnable が 1 にセットされるとタイマはクリアされます。	R

5.4.4　rTimerCurrentCount_[n] ― サブタイマ[n]（n＝6～7）の現在値

アドレス　5100 1004h＋20h×n（タイマ 1）
5100 2004h＋20h×n（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bTimerCurrentCount															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bTimerCurrentCount															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.6　rTimerCurrentCount_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bTimerCurrentCount	タイマの現在値 あるタイマの現在値がプリセット値と同じ場合、 　割り込みがトリガされます。 　bTimerEnable が 1 にセットされるとタイマはクリアされます。	R

5.4.5　rTimerControl_[n] ― サブタイマ[n]（n＝0～7）の制御モード

アドレス　5100 1008h＋20h×n（タイマ 1）
5100 2008h＋20h×n（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bTimer DMAEnable	bTimer MaskInt	bTimer Enable	bTimer Mode	bTimer Prescaler
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.7　rTimerControl_[n]レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	使用しない	R
b4	bTimerDMAEnable	DMA チャネルイネーブル 1：DMA チャネル有効 0：DMA チャネル無効 ・現在のアクセスに対する DMA ACK 応答受信後は、すべての現在の DMA 要求は強制的にアイドル状態となります。 **備考）** ● rTimerControl_6 および rTimerControl_7 でのみ対応しています。 ● rTimerControl_0～rTimerControl_5 では使用されず予約となります。	R/W
b3	bTimerMaskInt	サブタイマの割り込みマスク 0：割り込み禁止 1：割り込み許可	R/W
b2	bTimerEnable	0：タイマを無効とし、現在値を保持 ・タイマを停止し、bTimerCurrentCount をインクリメントせず、現在値のままとする。 1：タイマをリセットし開始する ・bTimerMode および bTimerPrescaler の状態をサンプリング ・プリスケーラカウンタをリセット ・bTimerCurrentCount をリセット ・インクリメントモードでタイマを開始 **注意）** タイマをリセットするには、ユーザは、bTimerEnable に“0”を書き込むことによりタイマを無効にした後、bTimerEnable に“1”を書き込むことによりタイマを有効にします。	R/W
b1	bTimerMode	本ビットは、下記エンコードに従ってタイマの動作モードを選択できます。 0：シングルショット ・タイマはプリセット値までインクリメントされます。 ・プリセット時間になると、割り込みが起動され、タイマは停止され無効になります。 1：自動リロード ・タイマはプリセット値までインクリメントされます。 ・プリセット時間になると、割り込みが起動され、タイマは自動的にクリアされ再開します。 ファームウェアが bTimerEnable を“1”にする（立ち上がり）と、bTimerMode がサンプリングされます。	R/W

表 5.7　rTimerControl_[n]レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bTimerPrescaler	本ビットは、下記のエンコードに従ってプリスケーラ設定を制御します。 1：タイムベース 1MHz（1μs） 0：タイムベース 25MHz ファームウェアが bTimerEnable を“1”にする（立ち上がり）と、bTimerPrescaler がサンプリングされます。	R/W

5.4.6　rTimerClearInt_[n] ― サブタイマ[n]（n＝0～7）の割り込みをクリア

アドレス　5100 100Ch＋20h×n（タイマ 1）
5100 200Ch＋20h×n（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bTimer ClearInt
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.8　rTimerClearInt_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	使用しない	R
b0	bTimerClearInt	本レジスタを読み出すと 0 が読み出され、タイマの割り込みをクリアします。	R

5.4.7　rTimerStatusInt0_[n] ― サブタイマ[n]のマスク前の割り込み状態（n＝0～7）

アドレス　5100 1010h＋20h×n（タイマ 1）
5100 2010h＋20h×n（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bTimer StatusInt0
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.9　rTimerStatusInt0_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	使用しない	R
b0	bTimerStatusInt0	TIMER_Int[n]の立ち上がりで割り込みが生成されます。 サブタイマのマスク前の割り込み状態 0：サブタイマに対して CPU 割り込みがアクティブではない 1：サブタイマに対して割り込みビットがセットされた 本レジスタを読み出しても、アクティブな割り込みをクリアしません。	R

5.4.8　rTimerStatusInt1_[n] — サブタイマ[n]のマスク後の割り込み状態（n＝0～7）

アドレス　5100 1014h＋20h×n（タイマ 1）
5100 2014h＋20h×n（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	bTimerStatusInt1
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.10　rTimerStatusInt1_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b1	予約ビット	使用しない	R
b0	bTimerStatusInt1	TIMER_Int[n]の立ち上がりで割り込みが生成されます。 サブタイマのマスク後の割り込み状態 0：サブタイマに対して CPU 割り込みがアクティブではないまたは割り込みがマスクされている 1：サブタイマに対して CPU 割り込みビットがセットされており、マスクされていない 本レジスタを読み出しても、アクティブな割り込みをクリアしません。	R

5.4.9　rTimerAllClearInt — すべての割り込みをクリア

アドレス　5100 1100h（タイマ 1）
5100 2100h（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	bTimerAllClearInt							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.11　rTimerAllClearInt レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	使用しない	R
b7～b0	bTimerAllClearInt	本レジスタを読み出すとすべて 0 が読み出され、すべてのアクティブな割り込みがクリアされます。	R

5.4.10　rTimerAllStatusInt0 ― マスク前のすべての割り込み状態

アドレス　5100 1104h（タイマ 1）
5100 2104h（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bTimerAllStatusInt0							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.12　rTimerAllStatusInt0 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	使用しない	R
b7～b0	bTimerAllStatusInt0	すべてのサブタイマのマスク前の割り込み状態 ビット割り当ては下記のとおり。 　ビット 7：サブタイマ 7 専用 　… 　ビット 0：サブタイマ 0 専用 各ビットとも下記のようになります。 　0：対応するサブタイマ CPU 割り込みは非アクティブ 　1：対応するサブタイマ CPU 割り込みは bTimerMaskInt ビット次第 本レジスタを読み出しても、アクティブな割り込みをクリアしません。	R

5.4.11　rTimerAllStatusInt1 ― マスク後のすべての割り込み状態

アドレス　5100 1108h（タイマ 1）
5100 2108h（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bTimerAllStatusInt1							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.13　rTimerAllStatusInt1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	使用しない	R
b7～b0	bTimerAllStatusInt1	すべてのサブタイマのマスク後の割り込み状態 ビット割り当ては下記のとおり。 　ビット 7：サブタイマ 7 専用 　… 　ビット 0：サブタイマ 0 専用 各ビットとも下記のようになります。 　0：対応するサブタイマ CPU 割り込みは非アクティブ 　1：対応するサブタイマ CPU 割り込みがアクティブ 本レジスタを読み出しても、アクティブな割り込みをクリアしません。	R

5.4.12　rTimer_DMA_Pending — タイマ DMA 要求状態

DMA 機能は、サブタイマ 6 およびサブタイマ 7 でのみ実装されます。

TIMER DMA 要求は、起動後、DMA 転送終了検出まで実行されます。

アドレス　5100 110Ch（タイマ 1）
5100 210Ch（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	bTimer_DMA_Running_7	bTimer_DMA_Running_6	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.14　rTimer_DMA_Pending レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	使用しない	R
b7	bTimer_DMA_Running_7	サブタイマ 7 の DMA 動作状態 サブタイマ 7 は、設定値からカウントアップし、カウントがプリセット値に達すると割り込みを生成します。 割り込みの立ち上がりが検出されると、TIMER DMA 要求が起動され、DMA 転送終了検出まで実行されます。 1：DMA 要求実行中 0：同一チャネルでの TIMER DMA 要求なし **備考）** ● 本チャネルで DMA 転送終了が検出されると本ビットは自動的にクリアされます。 ● 同じサイクルで設定要求とクリア要求を本ビットが受け取ると、それらの要因にかかわらず、本 bTimer_DMA_Running_7 ビットはクリアされ、設定要求は無視されます。この場合、DMA 要求プロセスは再開されません。 ● この場合、rTimer_DMA_PendingOvf レジスタのオーバーフロービット bTimer_DMA_RunningOvf_7 が、以前セットされたどうかに関係なくセットされます。	R
b6	bTimer_DMA_Running_6	bTimer_DMA_Running_7 と同様	R
b5～b0	予約ビット	使用しない	R

5.4.13　rTimer_DMA_PendingOvf ― タイマ DMA オーバーフロー状態

DMA 機能は、サブタイマ 6 およびサブタイマ 7 でのみ実装されます。

前の要求が実行されている状態で新しい TIMER DMA 要求が生成されたことを示します。

アドレス　5100 1110h（タイマ 1）
5100 2110h（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bTimer_DMA_RunningOvf_7	bTimer_DMA_RunningOvf_6	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.15　rTimer_DMA_PendingOvf レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	使用しない	R
b7	bTimer_DMA_RunningOvf_7	サブタイマ 7 の DMA 要求について TIMER DMA が実行されているときのオーバーフロー 前の要求がサブタイマ 7 で実行されている状態で新しい TIMER DMA 要求が生成されたことを示します。 1：DMA 要求での TIMER DMA オーバーフロー 0：TIMER DMA のオーバーフローなし **備考）** ● オーバーフロー状態になっても、TIMER DMA 転送処理は中止されません。単に、DMA 要求が処理されなかったことを示します。 ● つまり、最初の DMA 要求ブロックは、DMA 終了まで通常処理されますが、トリガされた 2 番目の DMA 転送ブロックは処理されません。	R
b6	bTimer_DMA_RunningOvf_6	bTimer_DMA_RunningOvf_7 と同様	R
b5～b0	予約ビット	使用しない	R

5.4.14　rTimer_DMA_PendingClrOvf — タイマ DMA オーバーフロークリア

DMA 機能は、サブタイマ 6 およびサブタイマ 7 でのみ実装されます。

アドレス　5100 1114h（タイマ 1）
5100 2114h（タイマ 2）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	bTimer_DMA_RunningClrOvf_7	bTimer_DMA_RunningClrOvf_6	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 5.16　rTimer_DMA_PendingClrOvf レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	使用しない	R
b7	bTimer_DMA_RunningClrOvf_7	サブタイマ 7 の DMA 要求について TIMER DMA が実行されているときのオーバーフローをクリア 1：オーバーフロービットをクリア 0：何もしません **備考）** ● ハードウェアが rTimer_DMA_PendingOvf レジスタのオーバーフロービット bTimer_DMA_RunningOvf_7 をセットしようとするのと同時に、ソフトウェアが本ビットに 1 を書き込もうとする場合、ハードウェアが優先され、bTimer_DMA_RunningOvf_7 ビットがセットされます。 ● 常に 0 が読めます。	W
b6	bTimer_DMA_RunningClrOvf_6	bTimer_DMA_RunningClrOvf_7 と同様	W
b5～b0	予約ビット	使用しない	R

5.5　動作説明

5.5.1　プリスケーラカウンタ

各サブタイマは、25MHz または 1MHz に接続された独立クロック入力を有しています。プリスケーラカウンタは、タイマをインクリメントするためのクロックを生成します。

bTimerPrescaler 信号により 2 つのクロックから選択可能です。

25MHz

- タイマは直接 TIMER_PCLK（25MHz）を使用します。

1MHz

- 25 分周器により TIMER_PCLK（25MHz）から新しいクロックを生成します。
- 本分周器は、アップモード（0～24 までカウント）のカウンタです。
- bTimerEnable を“1”にすると、リセット値（0）にプリスケーラカウンタを初期化し、そのあとに新しいサイクル（0、1、2～22、23、24）を開始します。
- bTimerEnable の立ち上がりにより、プリスケーラカウンタは“0”にリセットされます。

5.5.2　カウンタ 16 ビットまたは 32 ビット

サブタイマは、0 から設定値までカウントアップし、カウントがプリセット値に達すると割り込みが発生します。各サブタイマでは、単一の割り込みが発生し、その割り込みは個々のタイマ割り込みのいずれかがアクティブの場合は常にアクティブです。すべての割り込みステータスレジスタはいつでもアクセス可能です。

各サブタイマのプリセット値は、専用の rTimerLoadCount_[n]レジスタにロードされます。

タイマは 2 つのイベントでクリア可能

- リセット、または無効後の有効化。
- タイマがプリセット値までカウントアップ。

2 つの機能モードを実装

- 自動リロードモード：
  - タイマはプリセット値までインクリメントされます。
  - プリセット時間になると、割り込みが起動され、タイマは自動的にクリアされ再開します。
- シングルショットモード：
  - タイマはプリセット値までインクリメントされます。
  - プリセット時間になると、割り込みが起動され、タイマは停止され無効になります。

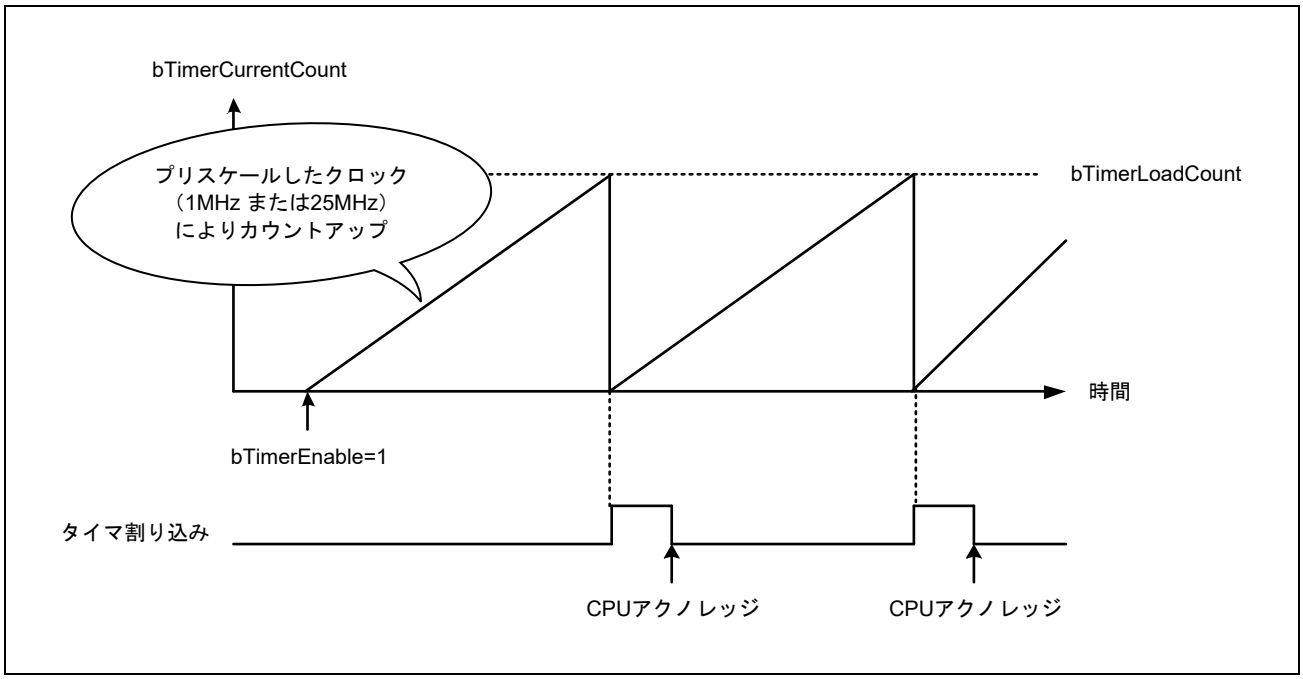


図 5.2　　タイマモード（bTimerMode）＝自動リロード

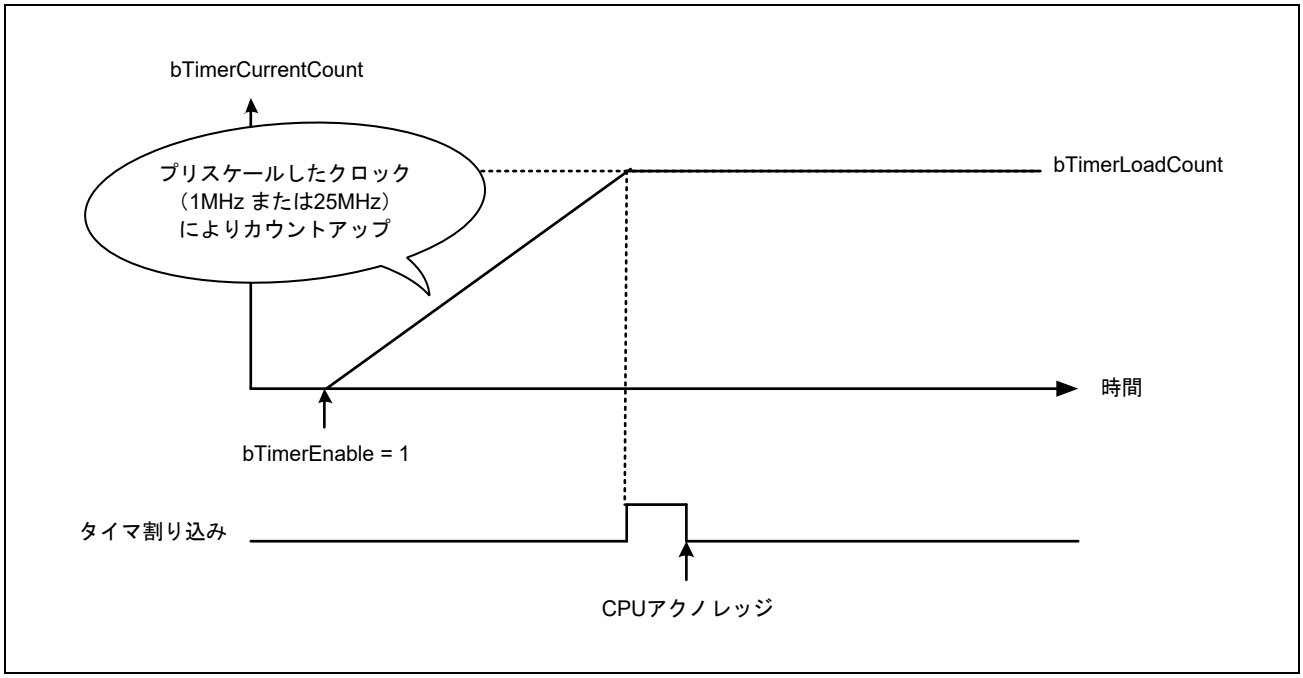


図 5.3　　タイマモード（bTimerMode）＝シングルショット

タイマを無効／有効にできます。

［無効］

- タイマインクリメントを停止し、bTimerCurrentCount は現在値のままとする。

［有効］

- プリスケーラカウンタをクリア
- カレントカウンタ bTimerCurrentCount をクリア
- タイマインクリメントを再開

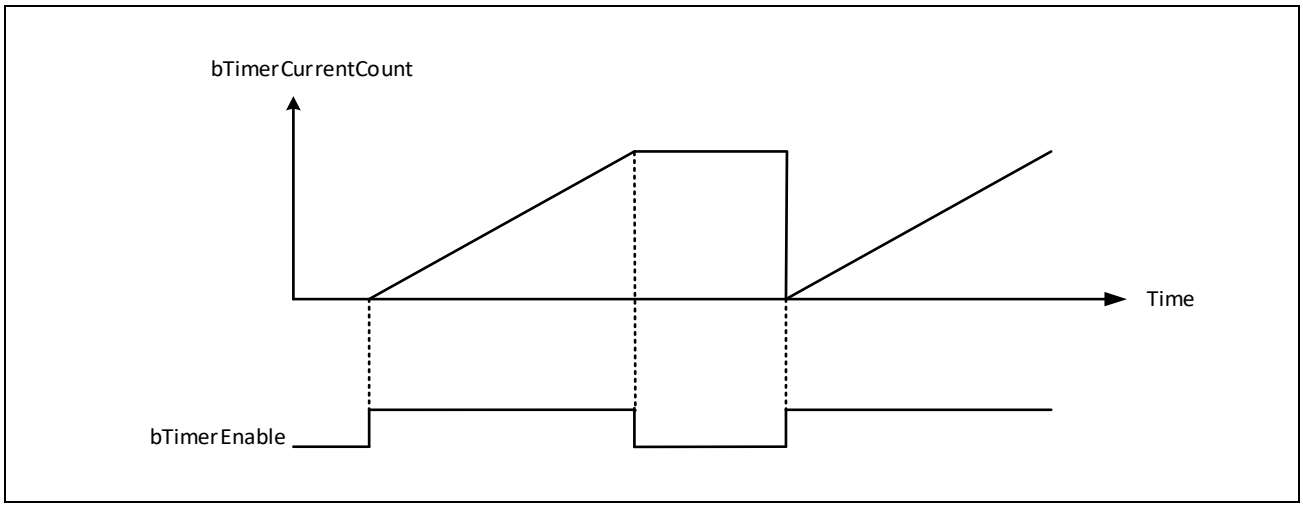


図 5.4　　bTimerEnable のオン／オフ動作

5.5.3　割り込み

タイマは、設定値までカウントアップし、カウントがプリセット値に達すると割り込みを生成します。各サブタイマでは、単一の割り込みが生成され、その割り込みは個々のタイマ割り込みのいずれかがアクティブの場合は常にアクティブです。

割り込みは、bTimerClearInt ビットの CPU 読み出しによってのみ ACK 応答可能です。

bTimerLoadCount＝5 の場合の例を下記に示します。

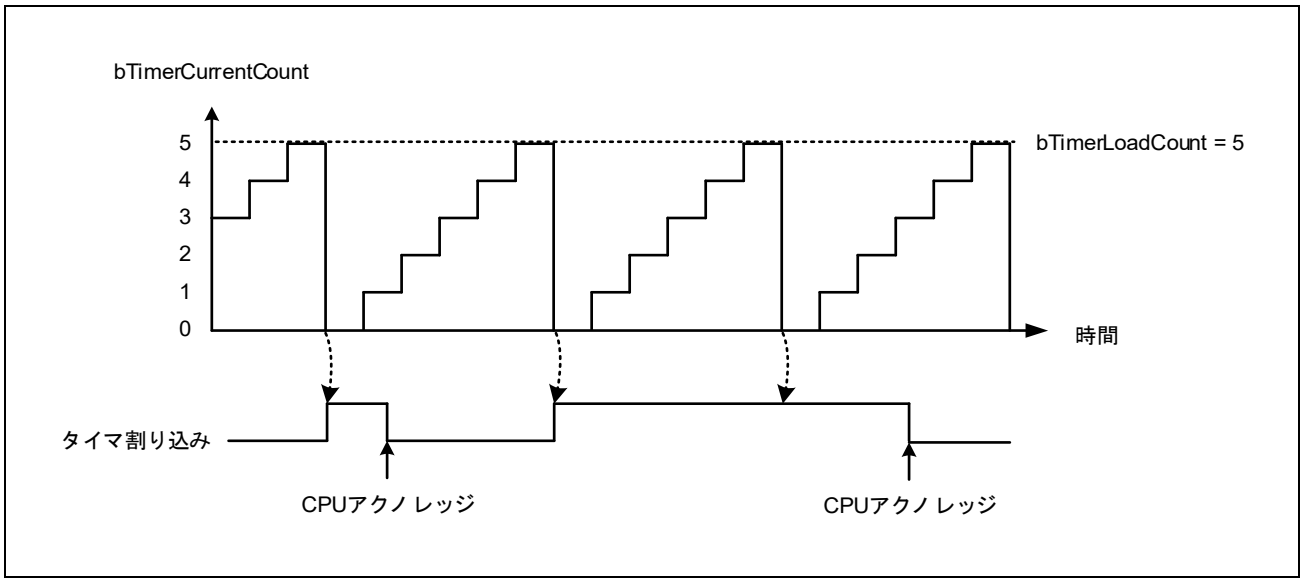


図 5.5　タイマ割り込み

5.5.4　DMA 制御

TIMER にはオプションの DMA 機能があります。TIMER ブロックは、転送を要求および制御するために DMA コントローラとの DMA 要求インタフェースを有しています。本モードで、DMA コントローラは、DMAC フローコントローラモードに設定する必要があります。DMA は効率化のために可能な場合常に DMA バーストトランザクションを使ってデータを転送します。

TIMER は、送信元から送信先へのメモリブロック転送のために 2 つの DMA チャネルを有しています。

各タイマブロックについて、TIMER DMA フロー制御は下記の DMA ビットにより管理されます。

- rTimerControl_[n]（n＝6 または 7）レジスタの bTimerDMAEnable ビット
  - サブタイマ[n]（n＝6 または 7）の DMA チャネルを有効／無効にします。
- サブタイマ[n]（n＝6 または 7）の割り込み
  - サブタイマ[n]は、設定値からカウントアップし、カウントがプリセット値に達すると割り込みを生成します。
  - 割り込みの立ち上がりが検出されると、TIMER DMA 要求が起動され、DMA 転送終了検出まで実行されます。
- rTimer_DMA_Pending レジスタの bTimer_DMA_Running_[n]（n＝6 または 7）ビット
  - TIMER DMA 要求は、起動後、DMA 転送終了検出まで実行されます。

TIMER の DMA コントローラインタフェースを有効にし、DMA 要求インタフェースを有効にするには

- DMA コントローラを設定します。
  - 送信元アドレスおよび送信先アドレス
  - 送信元および送信先のバーストサイズ
  - 転送ブロックサイズ
  - DMAC フローコントローラモード
  - チャネル割り当て
  - 1 ブロックのみ
  - 割り込み設定
- rTimerControl_[n]（n＝6 または 7）レジスタの bTimerDMAEnable ビットをセットし、TIMER DMA チャネルを有効にします。
  - サブタイマ[n]でタイマ割り込みの立ち上がりが検出されると、TIMER DMA 要求が起動され、DMA 転送終了検出まで実行されます。
- rTimer_DMA_Pending レジスタの bTimer_DMA_Running_[n]（n＝6 または 7）ビットは、実行中の DMA トランザクションの現在の状態を示します。

注　意

DMA コントローラは、DMAC フローコントローラモードに設定する必要があります。というのも TIMER は、転送ブロックのサイズがわからないからです。

5.6　使用上の注意事項

注　意

TIMER モジュールの初期化、ロード、および有効化時に同期問題が起こらないようにするために、以下の基本手順に従ってください。

(1)　rTimerControl_[n]によりサブタイマを初期化

- bTimerEnable に“0”を書き込むことによりサブタイマを無効にする
- bTimerMaskInt に“0”を書き込むことによりサブタイマの割り込みをマスクする

(2)　rTimerClearInt_[n]により起こりうるサブタイマの割り込みをリセットする

- bTimerClearInt の“0”を読み出すことにより割り込みを解除する

(3)　rTimerLoadCount_[n]によりサブタイマ値を初期化

- bTimerLoadCount の 16/32 ビットに値を書き込む
- 決して 0 を書き込まないでください。サブタイマが動作しなくなります。

(4)　rTimerControl_[n]レジスタによりサブタイマを初期化

- タイマモードを選択
- プリスケーラモードを選択
- bTimerMaskInt に“1”を書き込むことによりサブタイマの割り込みをマスクしない
- bTimerEnable に“1”を書き込むことによりサブタイマを有効にする

注　意

サブタイマ動作時にファームウェアが bTimerLoadCount を変更した場合の注意事項

- たとえば、bTimerCurrentCount＝“6”かつ bTimerLoadCount＝“10”でサブタイマを有効にします。
- ファームウェアが bTimerLoadCount＝“5”に新しい値をロードすると、プリセットが検出され、割り込みがトリガされます。
- すなわち、サブタイマが有効で、ファームウェアが bTimerCurrentCount より小さい 0 以外の新しい値を bTimerLoadCount にロードすると、プリセットが検出され、割り込みがトリガされます。
- その後、サブタイマの動作は bTimerMode に依存します（自動リロードモードまたはシングルショットモード）。

第6章　CAN

6.1　概要

RZ/N1 には、“同期フレーム”送信メカニズムのある CAN コントローラの 2 つのインスタンスがあります。

CAN 2.0 向け機能

- フル機能の CAN 2.0 – 2.0A（CAN 1.2 に相当）と 2.0B の両方をサポート
- 11 ビット識別子と 29 ビット識別子の両方をサポート
- 125Kbps 未満から、1Mbps を超えるビットレートをサポート
- 64 バイトの受信 FIFO
- アクセプタンスフィルタ機能
- ソフトウェア駆動のビットレート検出（ホットプラグインサポートの提供）
- シングルショット送信オプション、リッスンオンリモード、“自己”メッセージの受信
- ビット位置情報付きのアービトレーションロスト割り込み
- 読み出し／書き込みエラーカウンタ
- 最終エラーレジスタ
- プログラマブルエラーリミットワーニング
- PeliCAN モードでの Philips SJA1000 との幅広い互換性

追加機能

CANopen®向け“同期フレーム”の定期送信

- 以下を構成するための、プログラマブルタイムベース（ビット期間単位）：
  - “同期フレーム”送出の周期（“同期フレーム”の送出を非アクティブ化することが可能です）
  - マスクフレーム時間
  - パッシブエラー検出システム

2.0A 仕様（CAN 1.2 に相当）は、標準メッセージフォーマット（11 ビット識別子）を対象とします。2.0B 仕様は、標準メッセージフォーマットと拡張メッセージフォーマットの両方（11 ビット識別子と 29 ビット識別子の両方）を対象とします。

CAN コントローラは、PeliCAN モードで動作する Philips SJA1000 との幅広い互換性がありますが、いくつか例外があります（詳細については、「CAN コントローラとリファレンス Philips SJA1000 デバイスとの違い」の項を参照してください）。

本仕様は、BOSCH CAN 仕様バージョン 2.0（以降、CAN 2.0 仕様と呼ぶ）と合わせて読む必要があります。

注　意

- CAN 2.0 仕様は、CAN バス上で、ドミナントビットが論理値“0”の間、レセシブビットは論理値“1”であるという規則を使用します。
- この規則は本仕様でも使用されます。

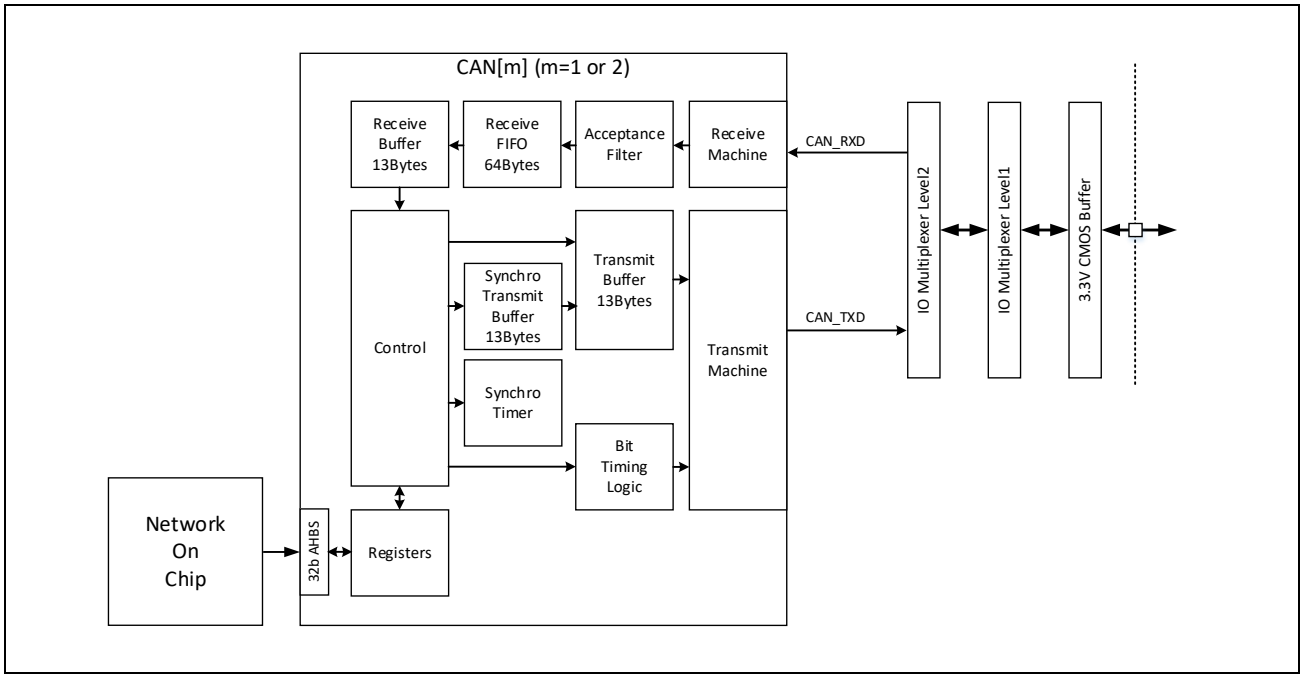


図 6.1　　CAN の概要

6.2　信号インタフェース

信号名	入力 出力	説明
クロック		
CAN[m]_HCLK	入力	内部バスクロック（AHB）、48MHz 固定
割り込み		
CAN[m]_Int	出力	レベル検出割り込み出力、アクティブ High
外部信号		
CAN[m]_RXD	入力	受信データ
CAN[m]_TXD	出力	送信データ

備考　m＝1 または m＝2
本章ではインデックスを省いたスタイルを使用します。
例）CAN_HCLK

6.3　レジスタマップ

6.3.1　レジスタマップ（CAN1）

表 6.1　CAN1 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5210 4000h	rCan_MOD	コンフィグレーションモードレジスタ
5210 4004h	rCan_CMR	コマンドレジスタ
5210 4008h	rCan_SR	コントローラステータスレジスタ
5210 400Ch	rCan_IR	割り込みレジスタ
5210 4010h	rCan_IER	割り込みイベントレジスタ
5210 4018h	rCan_BTR0	バスタイミングレジスタ 0
5210 401Ch	rCan_BTR1	バスタイミングレジスタ 1
5210 4020h	rCan_OCR	出力コントロールレジスタ
5210 402Ch	rCan_ALC	アービトレーションロストキャプチャレジスタ
5210 4030h	rCan_ECC	エラーコードキャプチャレジスタ
5210 4034h	rCan_EWLR	エラーワーニングリミットレジスタ
5210 4038h	rCan_RXERR	受信エラーカウンタレジスタ
5210 403Ch	rCan_TXERR	送信エラーカウンタレジスタ
5210 4040h～ 5210 4070h	rCan_WrTransmitBuffer	送信バッファ書き込みレジスタ
	rCan_RdReceiveBuffer	受信バッファ読み出しレジスタ
5210 4040h＋4h×n	rCan_ACR[n]（n＝0～3）	アクセプタンスコードフィルタ[n]レジスタ
5210 4050h＋4h×n	rCan_AMR[n]（n＝0～3）	アクセプタンスマスクフィルタ[n]レジスタ
5210 4074h	rCan_RMC	受信メッセージカウンタレジスタ
5210 4078h	rCan_RBSA	受信バッファスタートアドレスレジスタ
5210 4080h～ 5210 417Ch	rCan_ReceiveFifo	受信 FIFO レジスタ
5210 4180h～ 5210 41B0h	rCan_RdTransmitBuffer	送信バッファ読み出しレジスタ
5210 4440h～ 5210 4470h	rCan_SyncTransmitBuffer	同期フレーム送信バッファレジスタ
5210 4480h	rCan_SyncPeriod	同期フレーム送信タイムウィンドウレジスタ
5210 4488h	rCan_SyncStatusInt	同期フレーム割り込みステータスレジスタ
5210 448Ch	rCan_SyncMaskInt	同期フレームマスク割り込みレジスタ
5210 4490h	rCan_SyncClearInt	同期フレームクリア割り込みレジスタ
5210 4494h	rCan_SyncStatus	同期フレームステータスコンフィグレーションレジスタ
5210 4498h	rCan_SyncClearSetRunStop	同期フレーム生成レジスタ
5210 44A0h	rCan_SyncPassiveError	同期パッシブエラー検出レジスタ

6.3.2　レジスタマップ（CAN2）

表 6.2　CAN2 レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5210 5000h	rCan_MOD	コンフィグレーションモードレジスタ
5210 5004h	rCan_CMR	コマンドレジスタ
5210 5008h	rCan_SR	コントローラステータスレジスタ
5210 500Ch	rCan_IR	割り込みレジスタ
5210 5010h	rCan_IER	割り込みイベントレジスタ
5210 5018h	rCan_BTR0	バスタイミングレジスタ 0
5210 501Ch	rCan_BTR1	バスタイミングレジスタ 1
5210 5020h	rCan_OCR	出力コントロールレジスタ
5210 502Ch	rCan_ALC	アービトレーションロストキャプチャレジスタ
5210 5030h	rCan_ECC	エラーコードキャプチャレジスタ
5210 5034h	rCan_EWLR	エラーワーニングリミットレジスタ
5210 5038h	rCan_RXERR	受信エラーカウンタレジスタ
5210 503Ch	rCan_TXERR	送信エラーカウンタレジスタ
5210 5040h～ 5210 5070h	rCan_WrTransmitBuffer	送信バッファ書き込みレジスタ
	rCan_RdReceiveBuffer	受信バッファ読み出しレジスタ
5210 5040h＋4h×n	rCan_ACR[n]（n＝0～3）	アクセプタンスコードフィルタ[n]レジスタ
5210 5050h＋4h×n	rCan_AMR[n]（n＝0～3）	アクセプタンスマスクフィルタ[n]レジスタ
5210 5074h	rCan_RMC	受信メッセージカウンタレジスタ
5210 5078h	rCan_RBSA	受信バッファスタートアドレスレジスタ
5210 5080h～ 5210 517Ch	rCan_ReceiveFifo	受信 FIFO レジスタ
5210 5180h～ 5210 51B0h	rCan_RdTransmitBuffer	送信バッファ読み出しレジスタ
5210 5440h～ 5210 5470h	rCan_SyncTransmitBuffer	同期フレーム送信バッファレジスタ
5210 5480h	rCan_SyncPeriod	同期フレーム送信タイムウィンドウレジスタ
5210 5488h	rCan_SyncStatusInt	同期フレーム割り込みステータスレジスタ
5210 548Ch	rCan_SyncMaskInt	同期フレームマスク割り込みレジスタ
5210 5490h	rCan_SyncClearInt	同期フレームクリア割り込みレジスタ
5210 5494h	rCan_SyncStatus	同期フレームステータスコンフィグレーションレジスタ
5210 5498h	rCan_SyncClearSetRunStop	同期フレーム生成レジスタ
5210 54A0h	rCan_SyncPassiveError	同期パッシブエラー検出レジスタ

6.4　レジスタの説明

6.4.1　rCan_MOD ― コンフィグレーションモードレジスタ

下記のモードでの CAN コントローラの動作の設定に使用されます。

- スリープモード
- リッスンオンリモード
- アクセプタンスフィルタモード
- リセットモード
- セルフテストモード

アドレス　5210 4000h (CAN1)
5210 5000h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bCan_SM	bCan_AFM	bCan_STM	bCan_LOM	bCan_RM
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

表 6.3　rCan_MOD レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	読むと 0 が読み出されます。	R
b4	bCan_SM	スリープモード 動作モードでのみ書き込むことが可能です。 「**6.5.6 スリープモード**」を参照してください。 バスアクティビティがなく、保留中の割り込みがない場合、CAN コントローラはスリープモードにすることが可能です。 この機能は、ファームウェアの互換性のために実装され、省電力には影響ありません。 1：スリープモード コントローラは、保留中の CAN 割り込みがなく、バスアクティビティがない場合、スリープモードに入ります。バスアクティビティがあるか、割り込みが保留中の場合は、ウェイクアップ手順が実行されます。 0：ウェイクアップ（通常動作） スリープ中の場合、CAN コントローラはウェイクアップします。 ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	R/W
b3	bCan_AFM	アクセプタンスフィルタモード 「**6.5.7 アクセプタンスフィルタ処理**」を参照してください。 1：シングルフィルタ 受信データは 1 つの 4 バイトフィルタを使用してフィルタされる 0：デュアルフィルタ 受信データは 2 つのより短いフィルタを使用してフィルタされる	R/W
b2	bCan_STM	セルフテストモード 「**6.5.5 セルフ受信**」を参照してください。 1：セルフテスト有効 本モードでは、セルフ受信要求コマンドを使用して、バスでの他のアクティブノードなしにフルノードテストが可能です。CAN コントローラは、ACK 応答が受信されない場合でも、正常な送信を実行します。 0：通常動作 正常な送信に ACK 応答が必要です。	R/W

表 6.3　rCan_MOD レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b1	bCan_LOM	リッスンオンリモード 「**6.5.5 セルフ受信**」を参照してください。 1：リッスンオンリモード有効 本モードでは、メッセージが正常に受信された場合でも、CAN コントローラは ACK 応答を CAN バスに送信しません。 0：通常動作 エラーカウンタは現在の値で停止します。	R/W
b0	bCan_RM	リセットモード 「**6.5.14 リセットモード**」を参照してください。 1：リセットモード選択 現在送信中または受信中のメッセージが中断され、リセットモードに入ります。 0：通常動作 本ビットの“1 から 0 へ”の遷移でコントローラは動作モードに戻ります。 ［1 セット条件］ • “バスオフ”に切り替え（“1”を bCan_BS にセット）	R/W

6.4.2　rCan_CMR ― コマンドレジスタ

コマンドレジスタ内の 1 つ以上のビットをセットすることで、CAN コントローラの転送レイヤ内でアクションを開始します。

CAN コントローラで可能なアクションは次のとおりです。

- セルフ受信の要求
- データオーバーランのクリア
- 受信バッファの解放
- 送信の中断
- 送信の要求

注　意

- レジスタは書き込みのみです。読み出しの場合、すべてのビットは“0”を返します。
- コマンドビット bCan_AT および bCan_TR を同時にセットすると、エラーやアービトレーションロストの発生時に再送信のない、送信メッセージのシングルショット送信になります。
- コマンドビット bCan_AT および bCan_SRR を同時にセットすると、エラーやアービトレーションロストの発生時に再送信のない、セルフ受信機能を使用した送信メッセージのシングルショット送信になります。
- bCan_SRR および bCan_TR が同時にセットされた場合、bCan_SRR ビットは無視されます。
- 前のコマンドで作成された送信要求は、bCan_TR ビット（送信要求）を“0”にセットしてもキャンセルできません。要求された送信は bCan_AT ビット（送信中断）を“1”にセットすることでのみキャンセルできます。

アドレス　5210 4004h (CAN1)
5210 5004h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	bCan_SRR	bCan_CDO	bCan_RRB	bCan_AT	bCan_TR
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.4　rCan_CMR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	読むと 0 が読み出されます。	R
b4	bCan_SRR	セルフ受信要求 1：メッセージが同時に送受信される場合にセット ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	W

表 6.4　rCan_CMR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3	bCan_CDO	データオーバーランのクリア 1：bCan_DOS ビット（データオーバーランステータス）によって生成されたデータオーバーラン条件をクリアするためにセット。bCan_DOS ビット（データオーバーランステータス）がセットされている間は、これ以上のデータオーバーラン割り込みは生成されません。 ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	W
b2	bCan_RRB	受信バッファの解放 1：“受信バッファ”を解放するためセット ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	W
b1	bCan_AT	送信の中断 1：次の送信要求がまだ実行されていない場合、この送信要求をキャンセルするためにセット ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	W
b0	bCan_TR	送信の要求 1：メッセージが送信される場合にセット ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	W

6.4.3　rCan_SR ― コントローラステータスレジスタ

注　意

- 受信ステータス（bCan_RS）ビットおよび送信ステータス（bCan_TS）ビットの両方が“0”の場合、CAN バスはアイドルです。
- 両方のビット（bCan_RS および bCan_TS）が“1”の場合、コントローラは再びアイドルになるのを待ちます。
- ハードウェアリセット後、バスフリーシーケンス（11 個の連続したレセシブビット）が検出されるとアイドル状態に入ります。
- “バスオフ”イベント後、アイドル状態に入る前に 128 個のバスフリーシーケンスが受信される必要があります。

アドレス　5210 4008h (CAN1)
5210 5008h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_BS	bCan_ES	bCan_TS	bCan_RS	bCan_TCS	bCan_TBS	bCan_DOS	bCan_RBS
リセット後の値	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0

表 6.5　rCan_SR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7	bCan_BS	バス状態 1：CAN コントローラは“バスオフ”状態にあり、バスアクティビティに関与していません 0：CAN コントローラはバスアクティビティに関与しています	R
b6	bCan_ES	エラー状態 1：少なくとも 1 つのエラーカウンタが、エラーワーニングリミットレジスタ（rCan_EWLR レジスタの bCan_EWLR ビット）で定義された CPU ワーニングリミットに達したか、それを超えました 0：両方のエラーカウンタがワーニングリミットを下回ります	R
b5	bCan_TS	送信ステータス 1：CAN コントローラはメッセージを送信中です 0：送信中のメッセージはありません ［1 セット条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	R
b4	bCan_RS	受信ステータス 1：CAN コントローラはメッセージを受信中です 0：現在は何も受信していません ［1 セット条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	R
b3	bCan_TCS	送信完了ステータス 1：最後に要求された送信は、正常に終了しました 0：最後に要求された送信は、まだ終了していません	R

表 6.5　rCan_SR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b2	bCan_TBS	送信バッファステータス 1：送信バッファの解放 CPU は“送信バッファ”にメッセージを書き込むことができます。 0：送信バッファはロック中 メッセージが送信待機中か送信中であるため、CPU は“送信バッファ”にアクセスできません。 ［1 セット条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	R
b1	bCan_DOS	データオーバーランステータス 1：データオーバーラン 受信 FIFO のメッセージに対する容量が十分にないため、メッセージが失われました。 0：最後のデータオーバーランクリアコマンドが発行された後に、 データオーバーランが発生していません（rCan_CMR レジスタの bCan_CDO ビット） オーバーラン条件は、メッセージ全体が受信された場合のみ表示されます。 メッセージがエラーのため完了しなかった場合、オーバーラン条件は表示されません。 ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	R
b0	bCan_RBS	受信バッファステータス 1：受信バッファノットエンプティ 1 つ以上の完全なメッセージが、“受信バッファ”経由で“受信 FIFO”からの読み出し可能です。 0：受信バッファエンプティ 現在読み出し可能なメッセージはありません。 ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	R

6.4.4　rCan_IR ― 割り込みレジスタ

注　意

割り込みレジスタは、読み出し専用です。また、レジスタ読み出し後、受信割り込みビットを除いたすべての割り込みビットがリセットされます。

アドレス　5210 400Ch (CAN1)
5210 500Ch (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_BEI	bCan_ALI	bCan_EPI	bCan_WUI	bCan_DOI	bCan_EI	bCan_TI	bCan_RI
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.6　rCan_IR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7	bCan_BEI	バスエラー割り込み 1：割り込みイネーブルレジスタ（rCan_IER）内で bCan_BEIE ビットがセットされている場合、CAN コントローラが CAN バス上でエラーを検出するとセット 0：割り込みなし ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット） ● 本レジスタからの読み出し	R
b6	bCan_ALI	アービトレーションロスト割り込み 1：割り込みイネーブルレジスタ（rCan_IER）内で bCan_ALIE ビットがセットされている場合、CAN コントローラがアービトレーションに負けてレシーバになるとセット 0：割り込みなし ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット） ● 本レジスタからの読み出し	R
b5	bCan_EPI	エラーパッシブ割り込み 1：割り込みイネーブルレジスタ（rCan_IER）内の bCan_EPIE ビットがセットされている場合、エラーパッシブ状態の後、または、少なくとも 1 つのエラーがプロトコル定義の 127 レベルを超えた場合、CAN コントローラがアクティブエラー状態に再び入るときにセット 0：割り込みなし ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット） ● 本レジスタからの読み出し	R

表 6.6　rCan_IR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b4	bCan_WUI	ウェイクアップ割り込み 1：割り込みイネーブルレジスタ（rCan_IER）内で bCan_WUIE ビットがセットされている場合、CAN コントローラがスリープの間バスアクティビティが検出されるとセット。ウェイクアップ割り込みは、CAN コントローラがバスアクティビティに関与中か、CAN 割り込みが保留中の場合に、CPU がスリープモード（bCan_SM）ビットをセットしようとする場合にも生成されます。 0：割り込みなし ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット） ● 本レジスタからの読み出し	R
b3	bCan_DOI	データオーバーラン割り込み 1：割り込みイネーブルレジスタ（rCan_IER）内の bCan_DOIE ビットがセットされている場合、データオーバーランステータスビット（bCan_DOS）の“0 から 1 へ”の遷移でセット 0：割り込みなし ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット） ● 本レジスタからの読み出し	R
b2	bCan_EI	エラーワーニング割り込み 1：割り込みイネーブルレジスタ（rCan_IER）内の bCan_EIE ビットがセットされている場合、バスステータスビットあるいはエラーステータスビット（bCan_BS、bCan_ES）が変更（セットまたはクリア）されるたびにセット 0：割り込みなし ［0 クリア条件］ ● 本レジスタからの読み出し **備考）** “バスオフ”条件のためにリセットモードに入った場合、エラーワーニング割り込みがセットされます（許可されている場合）。	R
b1	bCan_TI	送信割り込み 1：割り込みイネーブルレジスタ（rCan_IER）内で bCan_TIE ビットがセットされている場合、送信バッファステータス（bCan_TBS）が“0 から 1 に”（解放）変化するときは常にセット 0：割り込みなし ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット） ● 本レジスタからの読み出し	R
b0	bCan_RI	受信割り込み 1：割り込みイネーブルレジスタ（rCan_IER）内で bCan_RIE ビットがセットされている場合、受信バッファに 1 つ以上のメッセージがあるときは常にセット。受信バッファに読み出すデータがそれ以上ない場合に、受信バッファ解放コマンド（bCan_RRB）が発行されたときクリアされます。 0：割り込みなし ［0 クリア条件］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット） bCan_RI ビット（有効な場合）は、受信バッファステータスビット（bCan_RBS）をミラーリングします。そのため、割り込みレジスタが読み出されたときに自動的にクリアされません。	R

6.4.5　rCan_IER — 割り込みイベントレジスタ

アドレス　5210 4010h (CAN1)
5210 5010h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	bCan_BEIE	bCan_ALIE	bCan_EPIE	bCan_WUIE	bCan_DOIE	bCan_EIE	bCan_TIE	bCan_RIE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.7　rCan_IER レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7	bCan_BEIE	バスエラー割り込み許可 1：許可。バスエラーが検出されたときに割り込みが発生します。 0：マスク。割り込みは禁止です。	R/W
b6	bCan_ALIE	アービトレーションロスト割り込み許可 1：許可。CAN コントローラがアービトレーションに負けたときに割り込みが発生します。 0：マスク。割り込みは禁止です。	R/W
b5	bCan_EPIE	エラーパッシブ割り込み許可 1：許可。割り込みは、CAN コントローラのエラーステータスがエラーアクティブからエラーパッシブへ、またはその逆の変更があったときに発生します。 0：マスク。割り込みは禁止です。	R/W
b4	bCan_WUIE	ウェイクアップ割り込みイネーブル 1：許可。スリーピング CAN コントローラがウェイクアップするときに割り込みが発生します。 0：マスク。割り込みは禁止です。	R/W
b3	bCan_DOIE	データオーバーラン割り込み許可 1：許可。データオーバーランステータスビット（bCan_DOS）がセットされているとき、割り込みが発生します。 0：マスク。割り込みは禁止です。	R/W
b2	bCan_EIE	エラーワーニング割り込み許可 1：許可。バスステータスビットまたはエラーステータスビット（bCan_BS、bCan_ES）を変更するときに割り込みが発生します。 0：マスク。割り込みは禁止です。	R/W
b1	bCan_TIE	送信割り込み許可 1：許可。メッセージが正常に送信されたか、“送信バッファ”が再びアクセス可能になったときに、割り込みが発生します。 0：マスク。割り込みは禁止です。	R/W
b0	bCan_RIE	受信割り込み許可 1：許可。受信バッファステータス（bCan_RBS）が“0”から“1”（“フル”）になったときに割り込みが発生します。 0：マスク。割り込みは禁止です。 受信割り込み許可ビットは受信割り込みビットと外部割込み出力 CAN_Int に直接影響があります。 bCan_RIE がクリアされる場合、他の割り込みが保留中でなければ、CAN_Int は即時非アクティブ（Low）になります。	R/W

6.4.6　rCan_BTR0 ― バスタイミングレジスタ 0

注　意

バスタイミングレジスタ 0 は、同期ジャンプ幅（SJW）とボーレートプリスケーラ（BRP）の値を定義します。本レジスタは、リセットモードのみで書き込み可能で、動作モードでは読み出し専用です。

アドレス　5210 4018h (CAN1)
5210 5018h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_SJW		bCan_BRP					
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.8　rCan_BTR0 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7、b6	bCan_SJW	同期ジャンプサイズ 同期ジャンプ幅は、現在の送信の関連する信号エッジ（レセシブからドミナントへ）で再同期するためにビット期間を短くしたり長くしたりすることのできる Time Quanta の最大数を定義します。 SJW＝Tq×(2×bCan_SJW[1]＋bCan_SJW[0]＋1)	R/W
b5～b0	bCan_BRP	ボーレートプリスケーラ ボーレートプリスケーラは、CAN 用クロックの“Time Quantum”Tq を、CAN_HCLK 周期の倍数として定義します。 CAN 用クロックの Time Quantum は以下で指定されます。 Tq＝2×Tperiod (CAN_HCLK)× (32×bCan_BRP[5]＋16×bCan_BRP[4] ＋8×bCan_BRP[3]＋ 4×bCan_BRP[2] ＋2×bCan_BRP[1]＋bCan_BRP[0]＋1) Tperiod (CAN_HCLK)＝CAN_HCLK の周期 「**6.5.13 ビット期間パラメータとバスタイミングパラメータ**」を参照してください。	R/W

6.4.7　rCan_BTR1 — バスタイミングレジスタ 1

注　意

バスタイミングレジスタ 1 は、ビット期間の長さ、サンプルポイントの位置、および、各サンプルポイントで取得されるサンプル数を定義します。
本レジスタは、リセットモードのみで書き込み可能で、動作モードでは読み出し専用です。

アドレス　5210 401Ch (CAN1)
5210 501Ch (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	bCan_SAM	bCan_TSEG2			bCan_TSEG1			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.9　rCan_BTR1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7	bCan_SAM	サンプルモード 1：バスは 3 回サンプリングされる。これは、低速または中速のバス（クラス A またはクラス B）で推奨されます。 0：バスは 1 度サンプリングされる。これは高速バス（SAE クラス C）で推奨されます。	R/W
b6～b4	bCan_TSEG2	bCan_TSEG1 および bCan_TSEG2 は、サンプルが取得されるまでの Time Quanta 数およびサンプルが取得される後の Time Quanta 数を指定することでビット期間の長さを定義します。 「**6.5.13 ビット期間パラメータとバスタイミングパラメータ**」を参照してください。 「**図 6.3 CAN：ビット期間、Tsyncseg、Tseg1、Tseg2 の一般的な構造**」を参照してください。 Tsyncseg＝1×Tq Tseg2＝Tq×(4×bCan_TSEG2[2]＋2×bCan_TSEG2[1]＋bCan_TSEG2[0]＋1)	R/W
b3～b0	bCan_TSEG1	bCan_TSEG1 および bCan_TSEG2 は、サンプルが取得されるまでの Time Quanta 数およびサンプルが取得される後の Time Quanta 数を指定することでビット期間の長さを定義します。 「**6.5.13 ビット期間パラメータとバスタイミングパラメータ**」を参照してください。 「**図 6.3 CAN：ビット期間、Tsyncseg、Tseg1、Tseg2 の一般的な構造**」を参照してください。 Tsyncseg＝1×Tq Tseg1＝Tq×(8×bCan_TSEG1[3]＋4×bCan_TSEG1[2]＋2×bCan_TSEG1[1]＋bCan_TSEG1[0]＋1)	R/W

6.4.8　rCan_OCR ― 出力コントロールレジスタ

注　意

本レジスタは、リセットモードのみで書き込み可能で、動作モードでは読み出し専用です。

アドレス　5210 4020h (CAN1)
5210 5020h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bCan_OCMODE	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.10　rCan_OCR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット	読むと 0 が読み出されます。	R
b1、b0	bCan_OCMODE	出力コントロールレジスタにより、出力ドライバ構成の選択ができます。 1 つのモード“通常出力”のみ実装されています。 本レジスタ経由の SJA1000 で利用可能な追加のドライバ構成は、CAN コントローラでサポートされていません。 通常出力モードでは、ビットシーケンスは CAN_TXD に送信されます。 2’b00：予約 2’b01：予約 2’b10：通常モード 2’b11：予約	R/W

6.4.9　rCan_ALC ― アービトレーションロストキャプチャレジスタ

アドレス　5210 402Ch (CAN1)
5210 502Ch (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bCan_ALC				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.11　rCan_ALC レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	読むと 0 が読み出されます。	R
b4～b0	bCan_ALC	バスアービトレーションロストの場合、アービトレーションロスト割り込み（bCan_ALI）が生成され（この割り込みを許可する場合、“1”を bCan_ALIE ビットにセット）、フレームの現在のビット位置は本レジスタにキャプチャされます。 本レジスタの内容はレジスタ読み出しされるまで保持されます。 キャプチャメカニズムが再びアクティブになります。 5’b0_0000：識別子の第 1 ビット（ID.28）でのアービトレーションロスト 5’b0_0001：識別子の第 2 ビット（ID.27）でのアービトレーションロスト … … … 5’b0_1001：識別子の第 10 ビット（ID.19）でのアービトレーションロスト 5’b0_1010：識別子の第 11 ビット（ID.18）でのアービトレーションロスト 5’b0_1011：SRTR ビットでのアービトレーションロスト（標準フレームフォーマットメッセージでの RTR ビット） 5’b0_1100：IDE ビットでのアービトレーションロスト 以下は、拡張フレームフォーマットメッセージのみ 5’b0_1101：識別子の第 12 ビット（ID.17）でのアービトレーションロスト 5’b0_1110：識別子の第 13 ビット（ID.16）でのアービトレーションロスト 5’b0_1111：識別子の第 14 ビット（ID.15）でのアービトレーションロスト 5’b1_0000：識別子の第 15 ビット（ID.14）でのアービトレーションロスト 5’b1_0001：識別子の第 16 ビット（ID.13）でのアービトレーションロスト … … … 5’b1_1101：識別子の第 28 ビット（ID.1）でのアービトレーションロスト 5’b1_1110：識別子の第 29 ビット（ID.0）でのアービトレーションロスト 5’b1_1111：RTR ビットでのアービトレーションロスト 「**6.5.9 バスアービトレーション（バスの調停）**」を参照してください。	R

6.4.10　rCan_ECC ― エラーコードキャプチャレジスタ

アドレス　5210 4030h (CAN1)
5210 5030h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_ECC_Code		bCan_ECC_Direction	bCan_ECC_Segment				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.12　rCan_ECC レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b6	bCan_ECC_Code	バスエラーが発生したとき、バスエラー割り込み（bCan_BEI）が生成され（この割り込みを許可する場合、“1”を bCan_BEIE ビットにセット）、フレームの現在のビット位置は本エラーコードキャプチャレジスタにキャプチャされます。本レジスタの内容はレジスタ読み出しされるまで保持されます。 キャプチャメカニズムが再びアクティブになります。 エラーコード： 2’b00：ビットエラー 2’b01：フォームエラー 2’b10：スタッフエラー 2’b11：別の種類のエラー 「**6.5.10 エラー処理**」を参照してください。	R
b5	bCan_ECC_Direction	エラーの方向： 1：受信中に発生したエラー 0：送信中に発生したエラー 「**6.5.10 エラー処理**」を参照してください。	R

表 6.12　rCan_ECC レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b4～b0	bCan_ECC_Segment	エラーセグメントコード： 5'b0_0000：未使用 5'b0_0001：未使用 5'b0_0010：ID.21～ID.28 5'b0_0011：フレームのスタート 5'b0_0100：SRTR ビット 5'b0_0101：IDE ビット 5'b0_0110：ID.18～ID.20 5'b0_0111：ID.13～ID.17 5'b0_1000：CRC シーケンス 5'b0_1001：予約ビット 0 5'b0_1010：データフィールド 5'b0_1011：データ長コード 5'b0_1100：RTR ビット 5'b0_1101：予約ビット 1 5'b0_1110：ID.0～ID.4 5'b0_1111：ID.5～ID.12 5'b1_0000：未使用 5'b1_0001：アクティブエラーフラグ 5'b1_0010：休止 5'b1_0011：ドミナントビット許容 5'b1_0100：未使用 5'b1_0101：未使用 5'b1_0110：パッシブエラーフラグ 5'b1_0111：エラーデリミタ 5'b1_1000：CRC デリミタ 5'b1_1001：ACK 応答 5'b1_1010：フレームの終了 5'b1_1011：ACK 応答デリミタ 5'b1_1100：オーバーロードフラグ 5'b1_1101：未使用 5'b1_1110：未使用 5'b1_1111：未使用 「**6.5.10 エラー処理**」を参照してください。	R

6.4.11　rCan_EWLR ― エラーワーニングリミットレジスタ

注　意

本レジスタは、リセットモードのみで書き込み可能で、動作モードでは読み出し専用です。

アドレス　5210 4034h (CAN1)
5210 5034h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_EWLR							
リセット後の値	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

表 6.13　rCan_EWLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_EWLR	エラーワーニングリミット 受信または送信のいずれかでのワーニングが生成されるエラー数。 送信エラーカウンタ（bCan_TXERR ビット）あるいは受信エラーカウンタ（bCan_RXERR ビット）が本値を超えると、ステータスレジスタ（rCan_SR）のエラーステータス（bCan_ES）ビットがセットされ、エラーワーニング割り込み（bCan_EI）が生成されます（この割り込みを許可する場合、“1”を bCan_EIE にセット）。 **「6.5.10 エラー処理」**を参照してください。 リセットモード内で行われた変更は、動作モードに戻るときにのみ有効になることにも注意する必要があります。	R/W

6.4.12　rCan_RXERR ― 受信エラーカウンタレジスタ

注　意

本レジスタは、リセットモードのみで書き込み可能で、動作モードでは読み出し専用です。

アドレス　5210 4038h (CAN1)
5210 5038h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_RXERR							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.14　rCan_RXERR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_RXERR	受信エラーカウンタ 本カウンタは、CAN 2.0 仕様で定義されたルールに沿って、エラーが受信ビットストリームで発生したときにインクリメントされ、メッセージがエラーなしで受信されたときにデクリメントされます。 関連付けられた送信エラーカウンタ（bCan_TXERR）とともに、CAN バス上で発生する送信の質を示します。 カウンタの 2 つのレベルが特定のイベントをトリガします。 ● カウンタがエラーワーニングリミットレジスタ（bCan_EWLR）でセットしたレベルに到達したとき、（以前、送信エラーカウンタ（bCan_TXERR ビット）によってトリガされていないかぎり）エラーワーニング割り込み（bCan_EI）が生成されます（この割り込みを許可する場合、“1”を bCan_EIE にセット）。 ● カウンタが 127 を超えると、デバイスは CAN 2.0 仕様に従って“エラーパッシブ”状態になります（以前、送信エラーカウンタによってトリガされていないかぎり）。また、最終アクティブエラーが送信され、エラーパッシブ割り込み（bCan_EPI）も生成されます（この割り込みを許可する場合、“1”を bCan_EPIE ビットにセット）。 “バスオフ”イベントが発生すると、カウンタは自動的に“0”にセットされます（bCan_BS に“1”をセット）。 **「6.5.10　エラー処理」**を参照してください。 ただし、CAN コントローラが“バスオフ”状態にあるときに、本レジスタへの書き込みは影響ないことに注意する必要があります。また、リセットモードで行われた変更は、動作モードに戻ったときにのみ有効になることにも注意する必要があります。 **備考**　“バスオフ”条件のためにリセットモードに入った場合（bCan_BS に“1”をセット）、受信エラーカウンタはクリアされます。また、送信エラーカウンタは 127 に初期化され、11 個の連続したレセシブビットが 128 回発生した時間に相当する、CAN 定義のバスオフリカバリ時間のカウントダウンに使用されます。	R/W

6.4.13　rCan_TXERR ― 送信エラーカウンタレジスタ

注　意

本レジスタは、リセットモードのみで書き込み可能で、動作モードでは読み出し専用です。

アドレス　5210 403Ch (CAN1)
5210 503Ch (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_TXERR							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.15　rCan_TXERR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_TXERR	送信エラーカウンタ 本カウンタは、CAN 2.0 仕様で定義されたルールに沿って、送信エラーが発生したときにインクリメントされ、メッセージがエラーなしで送信されたときにデクリメントされます。関連付けられた受信エラーカウンタ（bCan_RXERR）とともに、CAN バス上で発生する送信の質を示します。 カウンタの 3 つのレベルが特定のイベントをトリガします。 ● カウンタがエラーワーニングリミットレジスタ（bCan_EWLR）でセットしたレベルに到達したとき、（以前、受信エラーカウンタ（bCan_RXERR ビット）によってトリガされていないかぎり）エラーワーニング割り込み（bCan_EI）が生成されます（この割り込みを許可する場合、“1”を bCan_EIE にセット）。 ● カウンタが 127 を超えると、デバイスは、CAN 2.0 仕様に従って“エラーパッシブ”状態になります（以前、受信エラーカウンタによってトリガされていないかぎり）。また、最終アクティブエラーが送信され、エラーパッシブ割り込み（bCan_EPI）が生成されます（この割り込みを許可する場合、“1”を bCan_EPIE ビットにセット）。 ● カウンタが 255 を超える場合、CAN 2.0 仕様に従ってデバイスは“バスオフ”状態になり、自動的にリセットモードになります（1 ノードのみ CAN バスにあるときのスタートアップ中以外）。エラーワーニング割り込み（bCan_EI）も生成されます（この割り込みを許可する場合、“1”を bCan_EPIE ビットにセット）。 “バスオフ”イベントの後、CAN コントローラが CAN バスでさらに送信を行う（11 個の連続レセシブビットの“バスフリー”のシーケンス 128 回）前に、最小限のプロトコル定義の時間をカウントするためにレジスタは 127 に初期化されます。 この時間の間の送信エラーカウンタの読み出しにより、“バスオフ”リカバリ状態になります。 「**6.5.10 エラー処理**」を参照してください。	R/W

表 6.15　rCan_TXERR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
		備考） ● “バスオフ”リカバリが完了する（bCan_TXERR が 0 を超える）前にリセットモードに再び入った場合、CAN コントローラが動作モードに戻るまで、“バスオフ”は bCan_TXERR が凍結した状態でアクティブのままです。 ● “バスオフ”状態の間、0～254 の値を bCan_TXERR へ書き込むと、“バスオフ”フラグをクリアします。CAN コントローラは、リセットモードがクリアされた後に 1 つのバスフリーシーケンスのみを待ちます。 ● 255 を bCan_TXERR にリセットモードで書き込むと、CPU 駆動の“バスオフ”イベントを開始します。“バスオフ”イベントがバスエラーによって強制的に発行されたかのように実行されたとき、CAN コントローラが動作モードに戻るまで、新しい bCan_TXERR 値に対応してエラーが何も発生しないか、バスステータスの変更が発生します。このことは、リセットモードに再び入ったことを意味し、送信エラーカウンタが 127 に初期化、受信エラーカウンタはクリアされ、関連するステータスと割り込みレジスタのビットがセットされます。リセットモードのクリアは、プロトコル定義のバスオフリカバリシーケンスを実行します（128 回のバスフリー信号を待ちます）。 ● “バスオフ”条件のためにリセットモードに入った場合（bCan_BS に“1”をセット）、受信エラーカウンタはクリアされます。また、送信エラーカウンタは 127 に初期化され、11 個の連続したレセシブビットが 128 回発生する時間に相当する、CAN 定義のバスオフリカバリ時間のカウントダウンに使用されます。	

6.4.14　rCan_WrTransmitBuffer ― 送信バッファ書き込みレジスタ

注　意

本レジスタは書き込みのみであり、動作モードでのみ書き込むことが可能です。

アドレス　5210 4040h～5210 4070h (CAN1)
5210 5040h～5210 5070h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_WrTransmitBuffer							
リセット後の値	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X

表 6.16　rCan_WrTransmitBuffer レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_WrTransmitBuffer	送信バッファの書き込み フレームデータの CAN バスへの送信を許可します。送信バッファには 13 バイトの長さがあります。最大 8 データバイトの送信メッセージ 1 つに対応します。 「**6.5.3 送信**」を参照してください。 ● 送信バッファへの書き込みのみのアクセスが、 CAN オフセット 12'h040～12'h070 を使用する動作モードで提供されます。 ● 送信バッファへの読み出しアクセスは、 CAN オフセット 12'h180～12'h1B0 の使用で可能です。 送信バッファのグローバルレイアウトについては、「**6.5.11 送信バッファレイアウト**」を参照してください。 “標準フレームフォーマット”（SFF）メッセージと“拡張フレームフォーマット”（EFF）メッセージを区別することが重要です。	W

6.4.15　rCan_RdReceiveBuffer ― 受信バッファ読み出しレジスタ

アドレス　5210 4040h～5210 4070h (CAN1)
5210 5040h～5210 5070h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_RdReceiveBuffer							
リセット後の値	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X

表 6.17　rCan_RdReceiveBuffer レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_RdReceiveBuffer	受信バッファの読み出し 受信バッファは、CPU が受信 FIFO にアクセスするためのウィンドウを提供します。送信バッファと同様に、受信バッファも 13 バイトの長さがあります（最大 8 データバイトの 1 受信メッセージに対応可能）。「**6.5.4 受信**」を参照してください。 ● 受信バッファへの読み出しのみのアクセスが、 CAN オフセット 12'h040～12'h070 を使用する動作モードで提供されます。 受信バッファのグローバルレイアウトについては、「**6.5.12 受信バッファレイアウト**」を参照してください。 “標準フレームフォーマット”（SFF）メッセージと“拡張フレームフォーマット”（EFF）メッセージを区別することが重要です。	R

6.4.16　rCan_ACR[n] ― アクセプタンスコードフィルタ[n]レジスタ（n＝0～3）

注　意

このレジスタはリセットモードで書き込みのみ可能です。

アドレス　5210 4040h＋4h×n (CAN1)
5210 5040h＋4h×n (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_ACR							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.18　rCan_ACR[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_ACR	アクセプタンスコードフィルタ これらの 8 ビットのレジスタは、rCan_AMR[n]によって提供されたマスクと一緒に受信データをフィルタリングするときのアクセプタンスフィルタが使用したビットパターンを保持します。 これらのビットパターンが適用される方法は、シングルフィルタとデュアルフィルタのいずれが使用されるかによって、また、データが標準フレームフォーマット（SFF）か拡張フレームフォーマット（EFF）のいずれであるかによって異なります。 「**6.5.7 アクセプタンスフィルタ処理**」を参照してください。	R/W

6.4.17　rCan_AMR[n] ― アクセプタンスマスクフィルタ[n]レジスタ（n＝0～3）

注　意

このレジスタはリセットモードで書き込みのみ可能です。

アドレス　5210 4050h＋4h×n (CAN1)
5210 5050h＋4h×n (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_AMR							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.19　rCan_AMR[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_AMR	アクセプタンスマスクフィルタ これらの 8 ビットレジスタは、受信したデータをフィルタするときにアクセプタンスフィルタで適用されたマスクパターンを保持します。 ● これらのレジスタでの“0”は、対応するアクセプタンスコードレジスタ rCan_ACR[n]でのビット値に一致する必要のある受信データバイトのビットを特定します。 ● “1”は個々のビットを“don't care”としてマークします。 これらのマスクによって選択される受信データのビットは、シングルフィルタとデュアルフィルタのいずれが使用されるかによって、また、データが標準フレームフォーマット（SFF）か拡張フレームフォーマット（EFF）のいずれであるかによって異なります。 「**6.5.7 アクセプタンスフィルタ処理**」を参照してください。	R/W

6.4.18　rCan_RMC ― 受信メッセージカウンタレジスタ

アドレス　5210 4074h (CAN1)
5210 5074h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bCan_RMC				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.20　rCan_RMC レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	読むと 0 が読み出されます。	R
b4～b0	bCan_RMC	受信メッセージカウンタ 受信メッセージカウンタレジスタは、受信 FIFO で現在利用可能なメッセージ数を保持します。受信イベントごとに自動的にインクリメントされ、受信バッファ解放コマンドごとにデクリメントされます。 「**6.5.4 受信**」を参照してください。 ［“0”にクリア］ ● ソフトウェアのリセット（“1”を bCan_RM にセット） ● “バスオフ”に切り替え（“1”を bCan_BS にセット）	R

6.4.19　rCan_RBSA ― 受信バッファスタートアドレスレジスタ

注　意

このレジスタはリセットモードで書き込みのみ可能です。

アドレス　5210 4078h (CAN1)
5210 5078h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	bCan_RBSA					
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.21　rCan_RBSA レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b6	予約ビット	読むと 0 が読み出されます。	R
b5～b0	bCan_RBSA	受信バッファスタートアドレス 受信バッファスタートアドレスレジスタは、64 バイトの受信 FIFO 内の RX FIFO 読み出しポインタの現在の位置を 0～63 の間の値として保持します。 位置 0 は、CAN オフセット 12'h080 にマップします。 位置 63 は、CAN オフセット 12'h17C にマップします。 「**6.5.4 受信**」を参照してください。 本レジスタは、ソフトウェアリセットによる変更はありません（FIFO の内容も変更しません）。ただし、ソフトウェアリセットは、RX FIFO 書き込みポインタを RX FIFO 読み出しポインタの値へセットするため、ソフトウェアリセット後に受信バッファがアクセスしたデータは、受信 FIFO に保持される次のメッセージで上書きされます。	R

6.4.20　rCan_ReceiveFifo ― 受信 FIFO レジスタ

注　意

このレジスタはリセットモードで書き込みのみ可能です。

アドレス　5210 4080h～5210 417Ch (CAN1)
5210 5080h～5210 517Ch (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_ReceiveFifo							
リセット後の値	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X

表 6.22　rCan_ReceiveFifo レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_ReceiveFifo	受信 FIFO、64 バイト コントローラが受信したデータは、最初に“アクセプタンスフィルタ”によってフィルタリングされてから、“受信 FIFO”に渡されます。 “受信 FIFO”は 64 バイトの容量があり、最大 5 つのフル“拡張フレームフォーマット”メッセージのスペースがあり、循環方式で使用されます。 受信バッファ開始アドレスレジスタ（rCan_RBSA）は、64 バイトの受信 FIFO 内の RX FIFO 読み出しポインタの現在の位置を 0～63 の間の値として保持します。 位置 0 は、CAN オフセット 12’h080 にマップします。 位置 63 は、CAN オフセット 12’h17C にマップします。 「**6.5.4 受信**」を参照してください。	R/W

6.4.21　rCan_RdTransmitBuffer ― 送信バッファ読み出しレジスタ

アドレス　5210 4180h～5210 41B0h (CAN1)
5210 5180h～5210 51B0h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_RdTransmitBuffer							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.23　rCan_RdTransmitBuffer レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_RdTransmitBuffer	送信バッファの読み出し 送信バッファはリードバックが可能です。「**6.5.3 送信**」を参照してください。 ● 送信バッファへの書き込みのみのアクセスが、 CAN オフセット 12'h040～12'h070 を使用する動作モードで提供されます。 ● 送信バッファへの読み出しアクセスは、 CAN オフセット 12'h180～12'h1B0 の使用で可能です。 送信バッファのグローバルレイアウトについては、「**6.5.11 送信バッファレイアウト**」を参照してください。	R

6.4.22　rCan_SyncTransmitBuffer ― 同期フレーム送信バッファレジスタ

アドレス　5210 4440h～5210 4470h (CAN1)
5210 5440h～5210 5470h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bCan_SyncTransmitBuffer							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.24　rCan_SyncTransmitBuffer レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bCan_SyncTransmitBuffer	同期フレーム送信バッファ 本バッファは CPU によって“同期フレーム”の内容で書き込まれます。専用のタイムウィンドウで CAN バス上の“同期フレーム”の送信を許可します。 “同期フレーム”専用の本タイムウィンドウは、下記のアクションの実行を CAN コントローラに許可します。 ● CPU によって rCan_WrTransmitBuffer に書き込まれた最終フレームの送信 ● 本フレーム上の潜在的なリトライの管理（アービトレーションロスト） ● rCan_SyncTransmitBuffer で初期化された“同期フレーム”の送信 本タイムウィンドウは、rCan_SyncStatus レジスタの bCan_SyncMaskFrame ビットで管理されます。タイムウィンドウのサイズは、rCan_SyncPeriod レジスタのタイマ bCan_SyncMaskFrameTime ビットでコントロールされます。 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 送信バッファは 13 バイトの長さです。最大データ 8 バイトの送信メッセージ 1 つに対応します。送信バッファのグローバルレイアウトについては、「**6.5.11 送信バッファレイアウト**」を参照してください。	R/W

6.4.23　rCan_SyncPeriod ― 同期フレーム送信タイムウィンドウレジスタ

ビット期間単位での“同期フレーム”のタイムパラメータにより、下記の制御が可能になります。

- “同期フレーム”周期
- CAN バス上の“同期フレーム”の送信専用のタイムウィンドウ

アドレス　5210 4480h (CAN1)
5210 5480h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	bCan_SyncMaskFrameTime										—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bCan_SyncPeriod															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.25　rCan_SyncPeriod レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31、b30	予約ビット	読むと 0 が読み出されます。	R
b29～b20	bCan_SyncMaskFrameTime	CAN バス上の“同期フレーム”の送信専用のビット期間単位でのタイムウィンドウ ● 本タイマは下記の間の時間を設定します。 – bCan_SyncMaskFrame ビットの立ち上がりエッジ – “同期フレーム”の次の送出 ● “同期フレーム”専用の本タイムウィンドウは、下記のアクションの実行を CAN コントローラに許可します。 – CPU によって rCan_WrTransmitBuffer に書き込まれた最終フレームの送信 – 本フレーム上の潜在的なリトライの管理（アービトレーションロスト） – rCan_SyncTransmitBuffer で初期化された“同期フレーム”の送信 ● 本タイムウィンドウのサイズを計算するには、下記を考慮に入れる必要があります。 – CAN バス上で、rCan_WrTransmitBuffer に含まれる潜在的な最終フレームをリトライなしで送信するために必要な時間。 – CAN バス上で 2 回または 3 回のリトライの管理に必要な時間（アービトレーションロスト）。「**6.5.9 バスアービトレーション（バスの調停）**」を参照してください。 – CAN コントローラが、rCan_SyncTransmitBuffer で初期化された“同期フレーム”を、rCan_WrTransmitBuffer レジスタにコピーし、rCan_CMR レジスタの送信リクエスト（bCan_TR または bCan_SRR）を送信するために必要な時間（8 ビット期間単位かかる）。 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。 「**図 6.8 CANopen：周期的同期フレームの送出専用のタイムウィンドウ**」を参照してください。 10’h0：1 ビット期間 10’h1：2 ビット期間 10’h2：3 ビット期間 … … … 10’h3FF：1024 ビット期間	R/W

表 6.25　rCan_SyncPeriod レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
		● 常に下記を満たす必要があります。 – bCan_SyncPeriod は、8 を超える bCan_SyncMaskFrameTime より大きい。 ビット期間は rCan_BTR0 レジスタおよび rCan_BTR1 レジスタで定義されます。 「**図 6.3 CAN：ビット期間、Tsyncseg、Tseg1、Tseg2 の一般的な構造**」を参照してください。	
b19～b16	予約ビット	読むと 0 が読み出されます。	R
b15～b0	bCan_SyncPeriod	ビット期間単位での“同期フレーム”の周期 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 16’h0：1 ビット期間 16’h1：2 ビット期間 16’h2：3 ビット期間 … … … 16’hFFFF：65536 ビット期間 ビット期間は rCan_BTR0 レジスタおよび rCan_BTR1 レジスタで定義されます。 「**図 6.3 CAN：ビット期間、Tsyncseg、Tseg1、Tseg2 の一般的な構造**」を参照してください。	R/W

6.4.24　rCan_SyncStatusInt ― 同期フレーム割り込みステータスレジスタ

割り込みレジスタにより割り込みの要因が特定できます。本レジスタの 1 つ以上のビットがセットされると、CAN コントローラは CPU に割り込みを送信します。本レジスタは、“同期フレーム”管理専用です。

- オーバーラン“同期フレーム”割り込みのステータス
- “同期フレーム”割り込みのステータス

アドレス　5210 4488h (CAN1)
5210 5488h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	bCan_OverrunSyncFrameInt	bCan_SendSyncFrameInt
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.26　rCan_SyncStatusInt レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bCan_OverrunSyncFrameInt	オーバーラン“同期フレーム”割り込みのステータス ● 下記のいずれかの場合に割り込みが発生します。 – bCan_SyncMaskFrameTime ビットと bCan_SyncPeriod ビットで定義されたタイムウィンドウが正しく初期化されていない – CAN バス上の多くの繰り返しフレーム（アービトレーションロスト）。 「**6.5.9 バスアービトレーション（バスの調停）**」を参照してください。 **通常の場合：** – オーバーラン“同期フレーム”割り込みなし – “同期フレーム”の送出がタイムベース bCan_SyncPeriod と同期している **オーバーランの場合：** – オーバーラン“同期フレーム”割り込みが発生 – “同期フレーム”の送出がタイムベース bCan_SyncPeriod と同期していない – CAN バス上の必要なすべての繰り返しのある rCan_SyncTransmitBuffer レジスタに含まれる現在のフレームの送出を終える十分な時間を取るために“同期フレーム”の送信を遅らせます。 1：bCan_SyncRunStop ビットが“1”にセットされている場合（“同期フレーム”の送出が有効）、 本ビットは次のすべてを満たす場合にセットされます。 • “同期フレーム”の送信直後 • bCan_SyncMaskFrame ビットおよび bCan_SyncMaskFrameTime ビットでコントロールされるタイムウィンドウに、“同期フレーム”送信に必要な時間が不足している場合 0：割り込みは生成されません bCanOverrunSyncFrameClearInt ビットに“1”を書き込むことで、“0”にクリアされます。 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。 「**図 6.8 CANopen：周期的同期フレームの送出専用のタイムウィンドウ**」を参照してください。	R

表 6.26　rCan_SyncStatusInt レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bCan_SendSyncFrameInt	“同期フレーム”割り込みのステータス 1：bCan_SyncRunStop ビットが“1”にセットされている場合（“同期フレーム”の送出が有効）、 “同期フレーム”が送信されるとただちに本ビットがセットされます。 0：割り込みは生成されません bCan_SendSyncFrameClearInt ビットに“1”を書き込むことで“0”にクリアされます。 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。	R

6.4.25　rCan_SyncMaskInt ― 同期フレームマスク割り込みレジスタ

本レジスタは、割り込みを発生させるイベントを選択するために使用します。本レジスタは、“同期フレーム”管理専用です。

アドレス　5210 448Ch (CAN1)
5210 548Ch (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bCan_OverrunSyncFrameMaskInt	bCan_SendSyncFrameMaskInt
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.27　rCan_SyncMaskInt レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bCan_OverrunSyncFrameMaskInt	オーバーラン“同期フレーム”割り込みの許可または禁止 1：マスクなし。オーバーラン“同期フレーム”割り込みの許可 0：マスクあり。オーバーラン“同期フレーム”割り込みの禁止 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。 「**図 6.8 CANopen：周期的同期フレームの送出専用のタイムウィンドウ**」を参照してください。	R/W
b0	bCan_SendSyncFrameMaskInt	“同期フレーム”割り込みの許可または禁止 1：マスクなし。“同期フレーム”割り込みの許可 0：マスクあり。“同期フレーム”割り込みの禁止 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。	R/W

6.4.26　rCan_SyncClearInt ― 同期フレームクリア割り込みレジスタ

アドレス　5210 4490h (CAN1)
5210 5490h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bCan_OverrunSyncFrameClearInt	bCan_SendSyncFrameClearInt
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.28　rCan_SyncClearInt レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bCan_OverrunSyncFrameClearInt	オーバーラン“同期フレーム”割り込みの ACK 応答 ［1 の書き込み］ 　オーバーラン“同期フレーム”割り込みの ACK 応答 　bCan_OverrunSyncFrameInt ビットを“0”でクリアします。 ［0 の書き込み］ 　アクションなし 読むと 0 が読み出されます。 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。 「**図 6.8 CANopen：周期的同期フレームの送出専用のタイムウィンドウ**」を参照してください。	W
b0	bCan_SendSyncFrameClearInt	“同期フレーム”割り込みの ACK 応答 ［1 の書き込み］ 　“同期フレーム”割り込みの ACK 応答 　bCan_SendSyncFrameInt ビットを“0”でクリアします。 ［0 の書き込み］ 　アクションなし 読むと 0 が読み出されます。 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。	W

6.4.27　rCan_SyncStatus — 同期フレームステータスコンフィグレーションレジスタ

“同期フレーム”管理専用のステータス。下記のモードでの CAN コントローラの動作の設定に使用されます。

- ランモードおよびストップモード
- マスクフレームウィンドウステータス

アドレス　5210 4494h (CAN1)
5210 5494h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	bCan_TimerOnlyMode	bCan_TimerOnlyIfBusOff	bCan_SyncMode	bCan_SyncMaskFrame	—	—	bCan_SyncRunStop
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.29　rCan_SyncStatus レジスタの内容（1/3）

ビット位置	ビット名	機能	R/W
b31～b7	予約ビット	読むと 0 が読み出されます。	R
b6	bCan_TimerOnlyMode	0：CAN コントローラはフルモードで稼働中 1：CAN コントローラは TimerOnlyMode で稼働中。同期フレームの開始なし。CAN コントローラは同期フレームの割り込みのみを送信します。 本フィールドはハードウェアによってセットされます。	R/W
b5	bCan_TimerOnlyIfBusOff	0：CAN コントローラが“バスオフ”条件を検出すると、同期フレームシステムすべてを無効化 1：CAN コントローラが“バスオフ”条件を検出すると、“TimerOnlyMode”に切り替え **注意）** ● CAN コントローラは、同期フレームの開始タイミングの前に、8 ビット期間単位中“バスオフ”条件のみ検出できます。 ● **「図 6.8 CANopen：周期的同期フレームの送出専用のタイムウィンドウ」**を参照してください。 ● システムがバスオフを検出するには、バスオフステータスをクリアする前に、同期フレーム割り込みを待つことを推奨します。	R/W
b4	bCan_SyncMode	“同期フレーム”を送信するために使用されるリソース 2 つの機能モードが使用可能です。 （1）送信要求モード （2）セルフ受信要求モード 1：コマンドを bCan_SRR に書き込んで“同期フレーム”を送信（セルフ受信要求モード） 0：コマンドを bCan_TR に書き込んで“同期フレーム”を送信（送信要求モード） **「6.5.15 同期フレーム」**を参照してください。 **「図 6.8 CANopen：周期的同期フレームの送出専用のタイムウィンドウ」**を参照してください。	R/W

表 6.29　rCan_SyncStatus レジスタの内容（2/3）

ビット位置	ビット名	機能	R/W
b3	bCan_SyncMaskFrame	“同期フレーム”の送出用に予約されているタイムウィンドウ rCan_WrTransmitBuffer レジスタでの CPU アクセスを有効化または無効化。 rCan_WrTransmitBuffer に新しい CAN フレームを書き込む前に、CPU によってポーリングされる必要があります。 rCan_WrTransmitBuffer に最終 CAN フレームを書き込む前に、CPU は bCan_SyncMaskFrame ビットで“0”を読み出す必要があります。 1：“同期フレーム”の送出用に予約されているタイムウィンドウ CPU は、現在の CAN フレームの rCan_WrTransmitBuffer レジスタへの書き込みを終了できますが、バッファに新しい CAN フレームを書き込むことはできません。 0：標準 CAN フレームの送出に予約されたウィンドウ rCan_WrTransmitBuffer のすべての CPU アクセスが許可されます。 本ビットは専用タイマにより管理されます。ウィンドウのサイズは、rCan_SyncPeriod レジスタの bCan_SyncMaskFrameTime ビットでプログラム可能です。 “同期フレーム”専用の本タイムウィンドウにより、CAN コントローラは下記のアクションが実行できます。 – CPU によって rCan_WrTransmitBuffer に書き込まれた最終フレームの送信 – 本フレーム上の潜在的なリトライの管理（アービトレーションロスト） – rCan_SyncTransmitBuffer で初期化された“同期フレーム”の送信 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。 「**図 6.8 CANopen：周期的同期フレームの送出専用のタイムウィンドウ**」を参照してください。	R
b2、b1	予約ビット	読むと 0 が読み出されます。	R

表 6.29　rCan_SyncStatus レジスタの内容（3/3）

ビット位置	ビット名	機能	R/W
b0	bCan_SyncRunStop	“同期フレーム”の送出状態 特定のシーケンスにより、本ビットをセットまたはクリアできます。 1：“ランモード”。“同期フレーム”送出の有効化（ラン） bCan_SyncMaskFrame の生成 “同期フレーム”が送信されたときに、割り込みが発生します。 “同期フレーム”の設定は、以下のレジスタの CPU による初期化、ランに入る前に有効にする必要があります： - bCan_SyncMode（送信モードの使用） - rCan_SyncTransmitBuffer（同期フレームデータ） - bCan_SyncPeriod（“同期フレーム”の周期） - bCan_SyncMaskFrameTime（タイムベースのマスクフレーム） 0：“ストップモード”。“同期フレーム”送出の無効化（ストップ） 以下のすべてのビットがクリアされます。 - bCan_SyncRunStop ビット - bCan_SyncMaskFrame ビット “同期フレーム”管理用のすべてのタイマ 本レジスタに書き込むには、特定のシーケンスが必要です。 – rCan_SyncClearSetRunStop に 32’h2052_756E で書き込む “Run モード”に切り替え – rCan_SyncClearSetRunStop に 32’h5374_6F70 で書き込む “ストップモード”に切り替え これらの値のみが受け入れられ、他の値は無視されます。 **注意）** ● “同期フレーム”を送信する前に、CAN コントローラは、bCan_BS ビットのステータス（バスステータス）を確認します。 ● bCan_BS が“1”に読み出される場合（“バスオフ”モード）、CAN コントローラは、“0”へ bCan_SyncRunStop ビットをクリアします。 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.5 CANopen：周期的同期フレームの開始と停止**」を参照してください。 「**図 6.8 CANopen：周期的同期フレームの送出専用のタイムウィンドウ**」を参照してください。	R

6.4.28　rCan_SyncClearSetRunStop ― 同期フレーム生成レジスタ

アドレス　5210 4498h (CAN1)
5210 5498h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bCan_SyncClearSetRunStop															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bCan_SyncClearSetRunStop															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.30　rCan_SyncClearSetRunStop レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bCan_SyncClearSetRunStop	“同期フレーム”送出の実行またはストップ 特定のシーケンスにより、rCan_SyncStatus レジスタの bCan_SyncRunStop ビットをセットまたはクリアできます。 本レジスタに書き込むには、特定のシーケンスが必要です。 – rCan_SyncClearSetRunStop に 32’h2052_756E で書き込む “Run モード”に切り替え “同期フレーム”の送出の有効化 “1”を bCan_SyncRunStop ビットにセット bCan_SyncMaskFrame の生成 “同期フレーム”が送信されたときに、割り込みの生成 – rCan_SyncClearSetRunStop に 32’h5374_6F70 で書き込む “ストップモード”に切り替え “同期フレーム”の送出の無効化 “0”へ bCan_SyncRunStop ビットをクリア “0”へ bCan_SyncMaskFrame ビットをクリア “同期フレーム”管理用のすべてのタイマのクリア これらの値のみが受け入れられ、他の値は無視されます。 “同期フレーム”設定を変更する前に、ファームウェアは本レジスタで“ストップモード”に変更する必要があります。設定を書き込んだ後、ファームウェアは“同期フレーム”メカニズムをアクティブにするために本レジスタで“Run モード”に必要があります。 「**6.5.15 同期フレーム**」を参照してください。 「**図 6.5 CANopen：周期的同期フレームの開始と停止**」を参照してください。	W

6.4.29　rCan_SyncPassiveError ― 同期パッシブエラー検出レジスタ

アドレス　5210 44A0h (CAN1)
5210 54A0h (CAN2)

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bCan_SyncPassiveError															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 6.31　rCan_SyncPassiveError レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	読むと 0 が読み出されます。	R
b15～b0	bCan_SyncPassiveError	CAN モジュールのパッシブエラー検出システムの設定および有効化 ● “0” を bCan_SyncPassiveError に書き込むことで、“パッシブエラー検出システム” を無効にします。 ● bCan_SyncPassiveError に他の値を書き込むと、“パッシブエラー検出システム” が有効になります。“同期フレーム” の開始が遅れ、次の “同期フレーム” の前に残るビット期間数が bCan_SyncPassiveError より小さい場合に、bCan_TimerOnlyMode は自動的にセットされます。 「**図 6.7 CANopen：パッシブエラー検出システムのトリガ**」を参照してください。	R/W

6.5　動作説明

6.5.1　主な機能の説明

送信メッセージは、CPU によって CAN バス上の送信用の“送信バッファ”に入れられます。

デバイスが受信したメッセージは、最初に“アクセプタンスフィルタ”によってフィルタリングされてから、“受信 FIFO”に入れられます。CPU は、“受信バッファ”と呼ばれる 13 バイトのウィンドウ経由で“受信 FIFO”にアクセスします。“受信 FIFO”と合わせて“受信バッファ”を使用することで、1 つのメッセージが受信中に、CPU は別のメッセージを処理することが可能です。“受信 FIFO”は、全体の長さが 64 バイトあり、循環方式で使用されます。これによって、一度に最大 5 つの“拡張フレームフォーマット”メッセージに対応することが可能になります。

“ビットタイミングロジック”ブロックは、デバイスのボーレートに対応し、プログラム可能です。サポートされるボーレートの範囲は、内部バスクロックの周波数によって異なり、BOSCH 仕様によって選択される 125Kbps～1Mbps より広いボーレート範囲に容易に対応することが可能です。

CAN バスへのインタフェースは、送信用の信号 CAN_TXD や受信用の信号 CAN_RXD によって提供されます。

6.5.2　動作モード

CAN コントローラには、下記の 2 種類の主な動作モードがあります。

- データが送受信可能な“動作モード”
- バスタイミングパラメータとメッセージアクセプタンスフィルタがセットできる“リセットモード”。リセットモードでも、受信エラーおよび送信エラーのカウンタ（bCan_RXERR、bCan_TXERR）、およびエラーワーニングリミット（bCan_EWLR）の変更が可能です。

ハードウェアリセットを実行するか、モードレジスタ（rCan_MOD）の bCan_RM ビットを“1”にセットすることで、リセットモードが選択されます。

CAN コントローラは bCan_RM ビットをクリアすることで動作モードに戻ります。

“リセットモード”から“動作モード”へ切り替えるとき、bCan_TS ビットおよび bCan_RS ビットが 1 に等しい状態で、CAN コントローラが“アイドル待ち”の状態にあります。システムの使用準備が整う（“アイドル”状態になる）のは、これら 2 つのビットが 0 に戻るときのみです。

CAN コントローラは、動作モードまたはリセットモードのいずれかの時にモードレジスタ（rCan_MOD）経由で選択可能な、“リッスンオンリモード”と“セルフテストモード”もサポートします。

- “リッスンオンリモード”では、コントローラはデータを受信することのみ可能です。送信はできません。コントローラは、正常に受信されたデータの任意の ACK 応答を送信することはありません。“エラーパッシブ”も強制されます（「**6.5.10　エラー処理**」を参照してください）。
- “セルフテストモード”では、コントローラは自らのセルフ受信機能を使用して、任意のリモートノードからの ACK 応答を探すことなくメッセージを送信および受信します。

“リセットモード”の詳細については、「**6.5.14　リセットモード**」を参照してください。

“リッスンオンリ”モードと“セルフテスト”モードの詳細については、rCan_MOD レジスタの説明を参照してください。

注　意

コントローラは、任意のデータの送信または受信をするために、リセットモードから遷移する必要があります。

6.5.3　送信

メッセージを送信するために、CAN コントローラは、自身の動作モード（bCan_RM＝0）である必要があります。

送信割り込みおよびエラーワーニング割り込みの許可を推奨しますが、通常はアービトレーションロスト割り込みとバスエラー割り込みを許可する必要はありません。バスアービトレーションに負けた場合、またはメッセージの送信中に送信エラーが発生した場合に、CAN コントローラは、メッセージの送信を自動的に再度試みるためです。

送信されるデータは、“標準フレームフォーマット（SFF）”または“拡張フレームフォーマット（EFF）”のいずれかで、CAN の“送信バッファ”に書き込まれます。送信バッファは、CAN オフセット 12’h040 と 12’h070 の間の 13 バイトで構成されます。これにより、最大 8 バイトデータを含む 1 つのメッセージフレームのスペースを提供します。

- “送信バッファ”の書き込み：rCan_WrTransmitBuffer レジスタを参照。
- “送信バッファ”の読み出し：rCan_RdTransmitBuffer レジスタを参照。
- “送信バッファ”のグローバルレイアウトについては、「**6.5.11 送信バッファレイアウト**」を参照してください。

注　意

データをバッファに書き込む前に、送信バッファステータス（bCan_TBS）をチェックして、バッファが“解放”（bCan_TBS＝“1”）されたかどうか確かめる必要があります。バッファがロックされているとき（bCan_TBS＝“0”）、このバッファに書き込まれたデータは、いかなる通知もなく単に失われます。

“送信バッファ”に書き込まれたデータの送信は、次のいずれかを発行することによって開始されます。

- コマンドレジスタ経由の送信リクエスト（bCan_TR＝“1”のセットによる）
- または、メッセージのセルフ受信が必要な場合のセルフ受信要求（bCan_SRR＝“1”）（詳細については、「**6.5.5 セルフ受信**」を参照）。

CAN コントローラは、その後、バス上でデータを送信する準備として一連のステップを開始し（適切な 15 ビット CRC の生成を含む）、CAN バスがアイドルになるのを待ち送信を開始します。送信が開始されると、送信ステータス（bCan_TS）は“1”に変化し、送信要求ビットはクリアされます。ビットシーケンスが CAN_TXD に出力されます。

一度送信が進行中になると、CPU は単に送信割り込み（bCan_TI）が発生（この割り込みを許可する場合、“1”を bCan_TIE にセット）するのを待つか、または送信が終了したことを示すために送信バッファが解放される（“1”を bCan_TBS にセット）のを待つ必要があります。

バスアービトレーションに負けた場合（「**6.5.9 バスアービトレーション（バスの調停）**」を参照）または、メッセージの送信中に送信エラーが発生した場合（「**6.5.10 エラー処理**」を参照）、CAN コントローラは自動的にメッセージを再送信しようとします。

各フレームで 15 ビットの巡回冗長検査（CRC）が送信されます。CRC は、送信されるフレームの SOF（フレームの開始）、アービトレーション、コントロールフィールド、およびデータフィールドから生成されます。

詳細は、CAN 2.0 仕様を参照してください。

メッセージの送信がまだ始まっていない場合、この送信は、コマンドレジスタ（bCan_AT＝“1”）経由で送信の中断コマンドを発行することで中断できます。送信が開始された後は、メッセージを中断することはできません。

注　意

- 元のメッセージが正常に送信されたか中断されたかを判別するには、送信バッファステータスビット（bCan_TBS）が“1”にセットされるのを待つか、送信割り込みが生成される（bCan_TI）のを待って（この割り込みを許可する場合、bCan_TIE ビットをセット）から、送信完了ステータスビット（bCan_TCS）を確認してください。（送信割り込みは、メッセージが中断した場合でも生成されます。送信バッファステータスビットが“解放”に変更されるためです。）
- 送信要求（bCan_TR＝“1”）と同時に送信の中断を発行すると、現在のメッセージの“シングルショット”送信になり、送信エラーやバスアービトレーションロストの場合に再試行しません。

6.5.4　受信

メッセージを受信するために、CAN コントローラは、自身の動作モード（bCan_RM＝0）である必要があります。

受信は、CAN コントローラが「フレームの開始」を検出することよって開始されます。

コントローラが受信したデータは、最初に“アクセプタンスフィルタ”によってフィルタリングされてから、“受信 FIFO”に渡されます。“アクセプタンスフィルタ”は、“アクセプタンスフィルタ”レジスタに保持されるものと一致する識別子ビットを持つメッセージのみを渡します。

“受信 FIFO”へのデータ配置が開始されると、ステータスレジスタ（rCan_SR）の受信ステータスビット（bCan_RS）が“1”になります。その後、データが受信されると、受信バッファステータスビット（bCan_RBS）が“1”になり、受信割り込み（bCan_RI）が生成されます（この割り込みを許可する場合、bCan_RIE ビットをセット）。

“受信 FIFO”は 64 バイトの容量があり、これにより最大 5 つのフル“拡張フレームフォーマット（EFF）”メッセージのスペースを確保し、循環方式で使用されます。受信中のデータに対して FIFO に十分なスペースがない場合、ステータスレジスタ（rCan_SR）のデータオーバーランステータスビット（bCan_DOS）がセットされ、受信中のデータフレームは廃棄されます。データオーバーラン割り込み bCan_DOI も生成されます（この割り込みを許可する場合、bCan_DOIE ビットをセット）。

“受信 FIFO”に入れられるデータは、“受信バッファ”と呼ばれる 13 バイトのウィンドウ経由で読み出されます。本ウィンドウは、CAN オフセット 12’h040～12’h070 に配置され、“送信バッファ”と同じアドレス空間を占有します。“送信バッファ”と同様、最大 8 バイトのデータを含む 1 つのメッセージに対して十分な幅です。

各メッセージは“受信 FIFO”から読み出されるので、ホスト CPU は、コマンドレジスタ（rCan_CMR）経由で受信バッファ解放コマンドを発行（bCan_RRB＝“1”をセット）することで現在 FIFO 上にあるウィンドウを解放する必要があります。別のメッセージが“受信 FIFO”で読み出されるのを待っている場合、“受信バッファ”経由でただちに利用可能になります。待機しているメッセージがない場合、受信ステータスビット（bCan_RS）、受信割り込みビット（bCan_RI）、および受信バッファステータスビット（bCan_RBS）がすべてクリアされます。

- “受信バッファ”での読み出し：rCan_RdReceiveBuffer レジスタを参照。
- “受信バッファ”のグローバルレイアウトについては、「**6.5.12 受信バッファレイアウト**」を参照してください。

受信 FIFO で現在利用可能なメッセージ数のカウントは、受信メッセージカウンタレジスタ（rCan_RMC レジスタ参照）によって提供されます。

受信したデータは RX FIFO 書き込みポインタによって指示される位置に書き込まれ、このポインタは各バイトが書き込まれるときに更新されます。データが現在読み出されている位置は、RX FIFO 読み出しポインタ（rCan_RBSA）によって指定されます。

書き込みポインタが読み出しポインタと一致する場合、受信中のデータ用の FIFO にスペースはありません。受信中のデータフレームが廃棄され、ステータスレジスタ（rCan_SR）のデータオーバーランステータス（bCan_DOS）ビットがセットされ、データオーバーラン割り込み bCan_DOI が生成されます（この割り込みを許可する場合、“1”を bCan_DOIE ビットにセット）。このデータ消失からの復旧は、CPU 次第です。

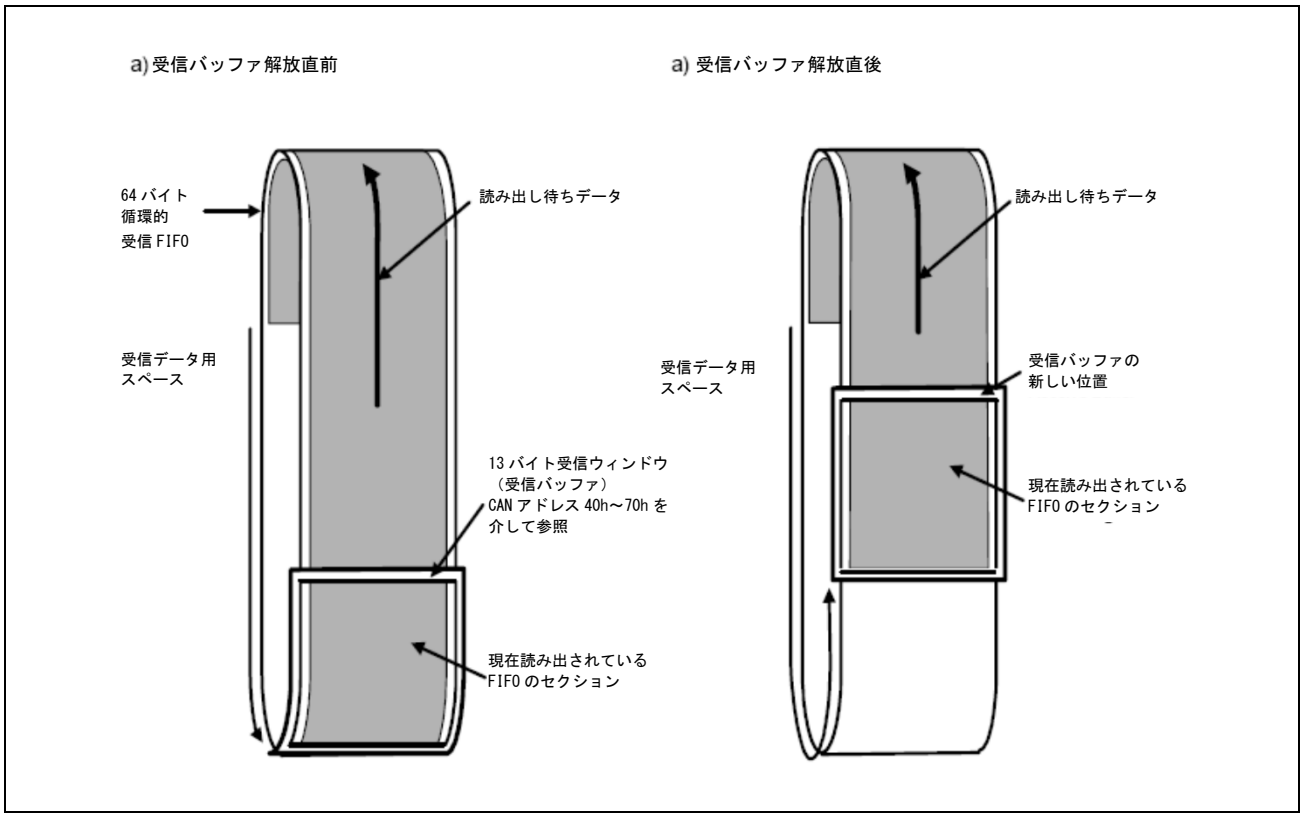


図 6.2　　CAN：受信バッファウィンドウまたは受信 FIFO

6.5.5　セルフ受信

CAN コントローラの機能により、他の CAN ノードへのメッセージ送信と CAN コントローラでのメッセージ受信を同時に行うことができます。

現在の送信メッセージのセルフ受信は、コマンドレジスタ（rCan_CMR）経由で、セルフ受信要求を発行（bCan_SRR ビット＝“1”をセット）することで選択されます。

CAN コントローラは正しい動作に必要な送受信割り込みを自動的に生成します。

注　意

- 通常の送信と同時にセルフ受信が要求される場合（bCan_SRR と bCan_TR を同時にセット）、セルフ受信要求が無視されます。
- セルフ受信機能の特別なバージョンが、リモートノードからの ACK 応答を必要とせずに、テストメッセージの送信と受信が両方なされる CAN コントローラのセルフテストモードによって提供されます。これによって、バス上の別のアクティブノードを必要としない、CAN コントローラを含むノードの完全なテストが可能です。
- rCan_MOD レジスタを参照してください。

6.5.6　スリープモード

バスアクティビティがなく、保留中の割り込みがない場合、CAN コントローラをスリープモードにすることが可能です。この機能は、ファームウェアの互換性のために実装され、省電力には影響ありません。

スリープモードは、モードレジスタ（bCan_SM）のスリープモードビットを“1”にセットすることで選択されます。CAN_TXD はスリープモードで High になります。下記のイベントのいずれかによって、CAN コントローラがスリープモードから“ウェイクアップ”になります。

- スリープモード（bCan_SM）ビットを“0”にセット
- CAN バス入力（CAN_RXD）でのアクティビティ

ウェイクアップすると、CAN コントローラはウェイクアップ割り込み（bCan_WUI）を生成します（この割り込みを許可する場合、bCan_WUIE ビットをセット）。

注　意

- **CAN コントローラがバスアクティビティによってウェイクアップされる場合、バス上で 11 個のレセシブビットのバスフリーシーケンスを検出する後まで、メッセージを受信しません。**
- **CAN コントローラがリセットモードの間、スリープモードを選択することができないことに注意してください。**

6.5.7　アクセプタンスフィルタ処理

CAN ネットワーク内では、すべてのノードが、バス上で送信されたすべてのメッセージを受信します。

自身に関連のないメッセージをノードが無視できるように、CAN コントローラは、受信データに 4 バイトのアクセプタンスフィルタを適用することで受信メッセージの事前フィルタリングができます。フィルタと一致する識別子ビット付きのメッセージのみが受信 FIFO に渡されます。

通常メッセージフィルタリングは、識別子全体をベースにしています。識別子は、受信メッセージが標準フレームフォーマットか拡張フレームフォーマットかによって、11 ビットまたは 29 ビットの長さが可能です。ただし、CAN コントローラでは、オプションのマスクレジスタが、特別な識別子ビットを“don’t care”にセットすることで、各グループの識別子付きのメッセージが受信され、受信 FIFO に入れることができます。

8 ビットのアクセプタンスコードレジスタ 4 つ（一致するビットパターンを保持する rCan_ACR0～3）と、アクセプタンスコードビットパターンを“don’t care”として特別なビットをマスクする 8 ビットのアクセプタンスマスクレジスタ 4 つ（rCan_AMR0～3）を一緒に使用して、フィルタ処理が実行されます。両方のレジスタセットは、各受信メッセージの最初の 4 バイトへのシングル 32 ビットフィルタとして、またはメッセージの最初の 2 バイトへの 2 つの別々の 16 ビットフィルタとしてのいずれかとして適用されます。2 つのフィルタが適用される場合、テストされた識別子ビットが少なくとも 1 つのフィルタと一致すれば、メッセージは受け入れられます。

フィルタの正確な適用は、データが標準フレームフォーマットか拡張フレームフォーマットのいずれであるか、あるいは、フィルタが 1 つまたは 2 つ適用されるかどうかによります。

CAN コントローラは、受信データストリームをフィルタし、必要なビットパターンの識別子を持っていないメッセージを廃棄します。

メッセージ識別子が一致するビットパターンは、アクセプタンスコードレジスタ（rCan_ACR0～3）に保持され、アクセプタンスマスクレジスタ（rCan_AMR0～3）で保持される値でマスクされます。

- rCan_AMR0～3 の“0”は、メッセージ識別子に一致する必要のある rCan_ACR0～3 の対応する位置でビットを識別します。
- “1”はそれぞれのビットを“don’t care”として識別します。

rCan_ACR0～3 として保持されるビットパターンは、シングル 4 バイトフィルタとして使用できるか、2 つのより短いフィルタとして使用できます。モードレジスタ（rCan_MOD）の bCan_AFM ビット経由で、選択が行われます。

- bCan_AFM＝“1”の場合、シングルフィルタが適用されます。
- bCan_AFM＝“0”の場合、2 つのフィルタが適用されます。2 つのフィルタが使用される場合、識別子がいずれかのフィルタに一致する場合に受信メッセージは受け入れられます。

rCan_ACR0～3 で定義されるビットパターンの適用方法は、以下のように、受信メッセージが標準フレームフォーマット（SFF）か拡張フレームフォーマット（EFF）かによってさらに異ります。

表 6.32　標準フレームフォーマット、シングルフィルタ

標準フレームフォーマット（SFF）		アクセプタンスフィルタ条件	
CAN オフセット	受信バッファフィールド	フィルタコード	フィルタマスク
12’h040	RX フレーム情報		
12’h044	RX 識別子 1 ID[28,27,26,25,24,23,22,21]	フィルタ 1：rCan_ACR0[7:0]	フィルタ 1：rCan_AMR0[7:0]
12’h048	RX 識別子 2 ID[20,19,18],RTR,X,X,X,X X：フィルタなし	フィルタ 1：rCan_ACR1[7:4] 使用しない：rCan_ACR1[3:0]	フィルタ 1：rCan_AMR1[7:4] 使用しない：rCan_AMR1[3:0]
12’h04C	RX データバイト 1	フィルタ 1：rCan_ACR2[7:0]	フィルタ 1：rCan_AMR2[7:0]
12’h050	RX データバイト 2	フィルタ 1：rCan_ACR3[7:0]	フィルタ 1：rCan_AMR3[7:0]

表 6.33　標準フレームフォーマット、デュアルフィルタ

標準フレームフォーマット（SFF）		アクセプタンスフィルタ条件	
CAN オフセット	受信バッファフィールド	フィルタコード	フィルタマスク
12’h040	RX フレーム情報		
12’h044	RX 識別子 1 ID[28,27,26,25,24,23,22,21]	フィルタ 1：rCan_ACR0[7:0] フィルタ 2：rCan_ACR2[7:0]	フィルタ 1：rCan_AMR0[7:0] フィルタ 2：rCan_AMR2[7:0]
12’h048	RX 識別子 2 ID[20,19,18],RTR,X,X,X,X X：フィルタなし	フィルタ 1：rCan_ACR1[7:4] フィルタ 2：rCan_ACR3[7:4]	フィルタ 1：rCan_AMR1[7:4] フィルタ 2：rCan_AMR3[7:4]
12’h04C	RX データバイト 1 バイト[7:4] バイト[3:0]	フィルタ 1：rCan_ACR1[3:0] フィルタ 1：rCan_ACR3[3:0]	フィルタ 1：rCan_AMR1[3:0] フィルタ 1：rCan_AMR3[3:0]
12’h050	RX データバイト 2 フィルタなし		

表 6.34　拡張フレームフォーマット、シングルフィルタ

拡張フレームフォーマット（EFF）		アクセプタンスフィルタ条件	
CAN オフセット	受信バッファフィールド	フィルタコード	フィルタマスク
12’h040	RX フレーム情報		
12’h044	RX 識別子 1 ID[28,27,26,25,24,23,22,21]	フィルタ 1：rCan_ACR0[7:0]	フィルタ 1：rCan_AMR0[7:0]
12’h048	RX 識別子 2 ID[20,19,18,17,16,15,14,13]	フィルタ 1：rCan_ACR1[7:0]	フィルタ 1：rCan_AMR1[7:0]
12’h04C	RX 識別子 3 ID[12,11,10,9,8,7,6,5]	フィルタ 1：rCan_ACR2[7:0]	フィルタ 1：rCan_AMR2[7:0]
12’h050	RX 識別子 4 ID[4,3,2,1,0],RTR,X,X X：フィルタなし	フィルタ 1：rCan_ACR3[7:2] 使用しない：rCan_ACR3[1:0]	フィルタ 1：rCan_AMR3[7:2] 使用しない：rCan_AMR3[1:0]

表 6.35　拡張フレームフォーマット、デュアルフィルタ

拡張フレームフォーマット（EFF）		アクセプタンスフィルタ条件	
CAN オフセット	受信バッファフィールド	フィルタコード	フィルタマスク
12’h040	RX フレーム情報		
12’h044	RX 識別子 1 ID[28,27,26,25,24,23,22,21]	フィルタ 1：rCan_ACR0[7:0] フィルタ 2：rCan_ACR2[7:0]	フィルタ 1：rCan_AMR0[7:0] フィルタ 2：rCan_AMR2[7:0]
12’h048	RX 識別子 2 ID[20,19,18,17,16,15,14,13]	フィルタ 1：rCan_ACR1[7:0] フィルタ 2：rCan_ACR3[7:0]	フィルタ 1：rCan_AMR1[7:0] フィルタ 2：rCan_AMR3[7:0]
12’h04C	RX 識別子 3 ID[12,11,10,9,8,7,6,5] フィルタなし		
12’h050	RX 識別子 4 ID[4,3,2,1,0],RTR,X,X フィルタなし		

6.5.8　割り込み生成

以下のいずれかの条件が発生したとき、CAN コントローラは割り込みの生成をサポートします。

- CAN コントローラがスリープモードである間のバスアクティビティ（ウェイクアップ割り込み、bCan_WUI）
- メッセージの受信（受信割り込み、bCan_RI）
- 現在の送信の完了（送信割り込み、bCan_TI）
- フルになっている FIFO 経由での受信データの消失（データオーバーラン割り込み、bCan_DOI）
- CAN バス上のアービトレーションロスト（アービトレーションロスト割り込み、bCan_ALI）
- CAN バス上のエラー（バスエラー割り込み、bCan_BEI）
- CAN コントローラが“エラーパッシブ”状態から遷移（エラーパッシブ割り込み、bCan_EPI）
- エラー数がエラーワーニングリミット（rCan_EWLR）を超えるか、またはエラー数に起因するデバイスの“バスオフ”状態（エラーワーニング割り込み、bCan_EI）

“同期フレーム”送信メカニズムを使用する場合のみ

- “同期フレーム”送信の完了
  - “同期フレーム”割り込みのステータス、bCan_SendSyncFrameInt
- “同期フレーム”の送信のオーバーラン
  - rCan_SyncTransmitBuffer レジスタに含まれる現在のフレームの送出を終える十分な時間を取るために“同期フレーム”の送信を遅らせます。
  - オーバーラン“同期フレーム”割り込みのステータス、bCan_OverrunSyncFrameInt

ハードウェアリセットに続く、これらの割り込みは無効です。よって、ユーザは割り込みイネーブルレジスタ（rCan_IER および rCan_SyncMaskInt）で必要な割り込みを許可する必要があります。ただし、許可された割り込みの選択は、ソフトウェアリセットによる影響がありません（モードレジスタ（rCan_MOD）の、bCan_RM＝“1”）。

注　意

- **通常、CPU は、アービトレーションロストイベントや個々のバスエラーを監視する必要はありません。それらが発生した場合に CAN コントローラが自動的に送信を再試行するためです。**
- **割り込みレジスタ（rCan_IR）の読み出しは、受信割り込み（bCan_RI）を除く、レジスタ内のすべての割り込みを自動的にクリアします。**

次の各項では、各タイプの割り込みに応答するアクションについて説明します。

6.5.8.1　受信割り込み

受信割り込みの生成は、受信 FIFO のメッセージを読み出すことが可能であることを示します。メッセージは、受信バッファと呼ばれる受信 FIFO 上の 13 バイトウィンドウ経由で読み出され、CAN オフセット 12’h040～12’h070 に位置します（rCan_RdReceiveBuffer レジスタを参照）。

受信バッファ経由で現在アクセス可能なメッセージが読み出されると、CPU は、受信バッファ解放コマンド（bCan_RRB＝“1”）を発行することで、現在 FIFO 上にあるウィンドウを解放する必要があります。RX FIFO 読み出しポインタ（つまり受信バッファスタートアドレス）はその後、次のメッセージが開始する受信 FIFO の位置に移動します。

この位置に読み出されていないメッセージがある場合、ただちに受信バッファ経由で読み出すことができるようになります。メッセージがない場合、受信割り込みビット（bCan_RI）および受信バッファステータスビット（bCan_RBS）はクリアされます。

6.5.8.2　送信割り込み

送信割り込み（bCan_TI）の生成は、送信バッファが別の送信メッセージを受信する準備ができたことを示します。本割り込みに対する応答は、単に送信されるデータがそれ以上あるかどうかによって異なります。データがまだ存在する場合、送信の項で説明される送信手順が繰り返される必要があります。データがない場合、割り込みは無視できます。

6.5.8.3　エラーワーニング割り込み

エラーワーニング割り込み（bCan_EI）の生成は、送信エラーの数（bCan_TXERR）または受信エラーの数（bCan_RXERR）がエラーワーニングリミットレジスタ（rCan_EWLR）に保持される EWL 値を超えたか、または送信エラーの数（bCan_TXERR）が 255 を超えたため CAN コントローラが“バスオフ”状態になったことのいずれかを示します。

受信エラーの数は、rCan_RXERR レジスタに保持され、送信エラーの数は rCan_TXERR レジスタに保持されます。

CAN コントローラが“バスオフ”状態になった場合、バスステータスビット（bCan_BS）は“1”（バスオフ）にセットされます。さらに、リセットモードビット（bCan_RM）がセットされ、ソフトウェアリセットを発生させ、CAN コントローラをリセットモードにし、ホスト CPU がモードレジスタ（bCan_RM）でリセットモードビットをクリアするまでリセットモードのままです。

さらに、動作モードに戻るときに、CAN コントローラは、再び“バスオン”になる前に、11 個の連続するレセシブビットであるバスフリーシーケンス（CAN プロトコル定義の最小時間）が 128 回発生するのを待ちます。

注　意

- この間、“バスオン”までの進捗は rCan_TXERR レジスタの読み出しによって監視できます。リセットモードから遷移するときは、レジスタは 127 に初期設定されます。その後、必要な数のバスフリーシーケンスによって 0 になるまでカウントダウンします（0 になるとデバイスは再び“バスオン”になることが可能です）。
- EWL 値を超過した結果として割り込みが生成される場合、本割り込みの発生に応じてどのアクションを取るかはプログラマ次第です。

6.5.8.4　データオーバーラン割り込み

データオーバーラン割り込み（bCan_DOI）は、受信メッセージに必要な格納ペースが受信 FIFO の空きバイト数より大きい場合のみ生成されます。データオーバーランステータスビット（bCan_DOS）もセットされます。

必要なストレージスペースは、受信メッセージの RTR ビット、FF ビット、および DLC ビットによって決定します。各ビットは以下を定義します。

- メッセージがリモート送信要求かどうか
- 標準フレームフォーマットメッセージか拡張フレームフォーマットメッセージのどちらであるか
- メッセージに含まれるバイト数

必要なスペースの評価は、メッセージが受信された後に行われます。メッセージの格納に十分なスペースがない場合は、メッセージは失われます。

メッセージが失われた場合に行われる復旧は、システム設計によって異なります。ただし、大量のデータオーバーランイベントを経験すると、データトラフィック量は予想を下回ることを意味し、また、システムが受信メッセージに対するより大きなメモリバッファが用意されていると助かることを意味します。

注　意

データオーバーランイベントを処理する割り込みルーチンは、データオーバーランステータスビット（bCan_CDO）がセットされている間は更なるデータオーバーラン割り込み（bCan_DOI）が生成されないため、データオーバーランクリアコマンド（bCan_CDO＝1）を発行して、データオーバーランステータスビットをクリアすることで、終了する必要があります。

6.5.8.5　ウェイクアップ割り込み

この機能は、ファームウェアの互換性のために実装され、省電力には影響ありません。

- ウェイクアップ割り込み（bCan_WUI）は、CAN コントローラがスリープモードからウェイクアップしたときに生成されます。

下記のイベントのいずれかによって、CAN コントローラがスリープモードから“ウェイクアップ”になります。

- スリープモードビット（bCan_SM）のクリア
- CAN バス入力（CAN_RXD）でのアクティビティ
- 「**6.5.6 スリープモード**」を参照してください。

6.5.8.6　エラーパッシブ割り込み

受信エラー（rCan_RXERR）カウンタおよび送信エラー（rCan_TXERR）カウンタは、受信エラーまたは送信エラーが発生するたびに、それぞれ自動的に 1 つずつインクリメントされ、受信または送信に成功するたびに 1 ずつデクリメントされます。

受信エラーおよび送信エラーの累計が 127 を超えた場合、CAN コントローラは、さらにエラーがカウントされ続けるものの、個々の割り込みは生成されない状態になります。この状態は、“エラーパッシブ”と呼ばれ、エラーパッシブ割り込み（bCan_EPI）を生成する（この割り込みを許可する場合、“1”を bCan_EPIE にセット）ことによってエラーパッシブ状態に入ったことを知らせます。

CAN コントローラは、エラーカウントが 127 を超えるかぎり、エラーパッシブ状態であり続けます。送信エラーカウント（rCan_TXERR）は、127 を超えている間はインクリメントおよびデクリメントされ続けます。ただし、受信エラーカウント（rCan_RXERR）は、各メッセージが正常に受信されたことよって自動的に 119 と 127 の間の値に減り、CAN コントローラはエラーパッシブ状態から抜ける可能性があります。

CAN コントローラがエラーパッシブ状態を抜けるときに、更なるエラーパッシブ割り込み（bCan_EPI）が生成されます。

6.5.8.7　アービトレーションロスト割り込み

アービトレーションロスト割り込み（bCan_ALI）の生成は、メッセージの送信中、CAN コントローラが CAN バスの制御を失ったことを意味します。

通常、CAN コントローラが自動的に現在のメッセージの送信を再試行するため、特別なアクションは必要ありません。ただし、シングルショット送信が行われる場合は、アービトレーションに負けたという事実に注意してください（「**6.5.3 送信**」を参照）。

アービトレーションロストビット位置は、アービトレーションロストキャプチャ（rCan_ALC）レジスタに保持されます。本ビットの詳細については、「**6.5.9 バスアービトレーション（バスの調停）**」を参照してください。

注　意

アービトレーションロストキャプチャレジスタ（rCan_ALC）に保持されるデータは、本レジスタが読み出されるまでクリアされません。結果として、以前保持されたデータが rCan_ALC レジスタから読み出されるまで、アービトレーションロストに関する更なる情報は保持できません。

6.5.8.8　バスエラー割り込み

バスエラー割り込み（bCan_BEI）の生成は、CAN バス上の送信エラーの発生を示します。通常、CAN コントローラは、バスエラーが発生した受信メッセージを自動的に廃棄し、バスエラーが発生した送信メッセージを自動的に再送信するので、特別なアクションは必要ありません。

ただし、バスエラーの追加の情報が必要な場合、エラーのタイプ（ビット／フォーム／スタッフ／その他）および各エラーの位置がエラーコードキャプチャレジスタ（rCan_ECC）でキャプチャされ、本レジスタが読み出されるまで保持されます。

一方、そのようなエラーが頻発する場合は、対策がとられるべきであるということを示すので、CAN コントローラは 2 つのエラーカウンタを保持します。

- 1 つは受信エラー用（rCan_RXERR）
- もう 1 つは、送信エラー用（rCan_TXERR）

これらはエラーが発生すると常に自動的にインクリメントされます。いずれかのカウンタがエラーワーニングリミットレジスタ（rCan_EWLR）値を超えるか、エラーワーニング割り込み（bCan_EI）が生成される場合（この割り込みを許可する場合、“1”を bCan_EIE にセット）、いずれかのカウンタが 127 カウントを超える場合は、エラーパッシブ割り込み（bCan_EPI）が生成されます（この割り込みを許可する場合、“1”を bCan_EPIE にセット）。

送信エラー数が 255 を超えた結果、CAN コントローラが“バスオフ”状態になる場合、エラーワーニング割り込み（bCan_EI）も生成されます。

これらの追加の割り込みは、上記で説明されます。

6.5.8.9　送信“同期フレーム”割り込み

注　意

“同期フレーム”送信メカニズムを使用する場合のみ。

“同期フレーム”送信の完了

- bCan_SyncRunStop ビットが“1”にセットされている場合（“同期フレーム”の送出が有効）、“同期フレーム”が送信されるときに割り込み（bCan_SendSyncFrameInt ビット）が生成されます（この割り込みを許可する場合、“1”を bCan_SendSyncFrameMaskInt ビットにセット）。
- bCan_SendSyncFrameClearInt ビットに“1”を書き込むことで、割り込みを ACK 応答してください。
- 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。

6.5.8.10　送信オーバーラン“同期フレーム”割り込み

注　意

“同期フレーム”送信メカニズムを使用する場合のみ。

“同期フレーム”の送信のオーバーラン

- rCan_WrTransmitBuffer レジスタに含まれる現在のフレームの送出を終える十分な時間を取るために“同期フレーム”の送信を遅らせます。
- “同期フレーム”が送信されるときに、割り込み（bCan_OverrunSyncFrameInt ビット）が生成され（この割り込みを許可する場合、“1”を bCan_OverrunSyncFrameMaskInt ビットにセット）、割り込み（bCan_SendSyncFrameInt ビット）が生成されます（この割り込みを許可する場合、“1”を bCan_SendSyncFrameMaskInt ビットにセット）。
- bCanOverrunSyncFrameClearInt ビットおよび bCan_SendSyncFrameClearInt ビットに“1”を書き込むことで、割り込みを ACK 応答してください。
- 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。

6.5.9　バスアービトレーション（バスの調停）

CAN コミュニケーションでは、バスアイドル状態中に最初に送信を開始したユニットが送信の権限を得ます。2 つ以上のユニットが同時に送信する場合、双方のアービトレーションフィールドのビットを比較することで、競合に対して調停され、他のユニットより多くのドミナントビットを持つユニットが送信権限を得ます。

それを管理するリソースは次のとおりです。バスの制御権を持つ CAN ノードは、常に最も小さい識別子を持つ CAN ノードです。アービトレーションに負けた CAN ノードは制御権を取り下げる必要があり、CAN バスがアイドルになるまで、CAN バスを再びコントロールしようとしません。

CAN コントローラを含むデバイスがアービトレーションに負けると、アービトレーションロスト割り込み（“1”を bCan_ALI ビットにセット）が発生し（この割り込みを許可する場合、rCan_IER レジスタの bCan_ALIE ビットをセット）、アービトレーションロストビット位置は、rCan_ALC レジスタの bCan_ALC ビットで保持されます。

本ビットの詳細については、bCan_ALC ビットを参照してください。

注　意

- アービトレーションロストキャプチャレジスタに保持されるデータは、本レジスタが読み出されるまでクリアされません。
- 結果として、以前保持されたデータがレジスタから読み出されるまで、アービトレーションロストに関する更なる情報が保持できません。

6.5.10　エラー処理

受信と送信でのエラーは、CAN 2.0B プロトコル仕様に従って処理されます。

エラー検出の主なルールは次のとおりです。

- すべてのユニットはエラーを検出できます。
  - エラー検出機能
- エラーを検出したユニットは、すべての他のユニットに同時にエラーを通知するために、ただちにエラーフレームを送信します。
  - エラー通知機能
- メッセージ送信ユニットがエラーを検出した場合、ユニットは強制的に送信を中断します。そのあと、メッセージが正常に送信されるまで、繰り返し再送信を試みます。
  - エラー回復機能

エラーのタイプ（ビット、フォーム、スタッフ、その他）およびメッセージフレームの中にあるエラーの位置が、エラーコードキャプチャ（rCan_ECC）レジスタ（bCan_ECC_Code ビット、bCan_ECC_Direction ビット、bCan_ECC_Segment ビット）でキャプチャされます。これらは本レジスタが読み出されるまで保持されます。

ビットエラー：

- 任意のビットの出力レベルおよびバス上のそのレベルが一致しないときに、本エラーが検出されます。

スタッフエラー：

- ビットスタッフィングフィールドで 6 つの連続したビットに対して同じレベルが検出されるときに本エラーが検出されます。

CRC エラー：

- 受信 CRC シーケンスが、受信データから計算された CRC の結果とは異なるとき、本エラーが検出されます。

フォームエラー：

- 固定フォームビットフィールド（CRC デリミタ、ACK デリミタ、EOF フィールド）が 1 つ以上不正なビットを含むときに、本エラーが検出されます。

ACK エラー：

- 送信ユニットが ACK スロットでレセシブビットを検出したとき、本エラーが検出されます。

CAN コントローラは次の 2 つのエラーカウンタを持っています。

- 受信エラー用（bCan_RXERR）
- 送信エラー用（bCan_TXERR）

これらは、エラーが発生すると、CAN 2.0B 仕様に従って自動的にインクリメントされます。

受信や送信が成功した場合も、CAN 2.0 仕様に従ってカウンタをデクリメントします。

CAN コントローラは、ワーニングが生成される必要のある受信または送信でのエラー数を表す値である、エラーワーニングリミット（bCan_EWLR）レジスタも含みます。bCan_EWLR のデフォルト値は 8’d96 です。

送信エラーカウンタ（bCan_TXERR）あるいは受信エラーカウンタ（bCan_RXERR）が本値を超えると、ステータスレジスタ（rCan_SR）のエラーステータス（bCan_ES）ビットがセットされ、エラーワーニング割り込み（bCan_EI）が生成されます（この割り込みを許可する場合、“1”を bCan_EIE にセット）。

どちらかのカウンタが 127 を超える場合、（CAN プロトコルで定義されるように）CAN コントローラは“エラーパッシブ”状態になり、エラーパッシブ（“1”を bCan_EPI ビットにセット）割り込み（この割り込みを許可する場合、“1”を bCan_EPIE ビットにセット）を発行します。

送信エラーカウンタが 255（カウンタリミット）を超えると、バスステータスビット（bCan_BS）が“1”（“バスオフ”）にセットされ、CAN コントローラはリセットモードにセットされ、エラーワーニング割り込み（bCan_EI）が発生します（この割り込みを許可する場合、“1”を bCan_EIE にセット）。

その後、ホスト CPU がモードレジスタのリセットモードビット（bCan_RM ビット）をクリアするまで、CAN コントローラはリセットモードを維持します。さらに、動作モードに戻るときに、CAN コントローラは、再び“バスオン”になる前に、バスフリーシーケンス（CAN プロトコル定義の最小時間）が 128 回発生するのを待ちます。

下の表は、bCan_RXERR カウンタおよび bCan_TXERR カウンタをインクリメントまたはデクリメントする主なルールの概要です。

表 6.36　送受信エラーカウンタのインクリメント／デクリメント

エラーイベント	アクション
レシーバがエラーを検出	bCan_RXERR を 1 ずつインクリメント
エラーフラグが送信された後、最初のビットとして、レシーバがドミナントビットを検出	bCan_RXERR を 8 ずつインクリメント
レシーバがアクティブエラーフラグまたはオーバロードフラグを送信中にビットエラーを検出	bCan_RXERR を 8 ずつインクリメント
メッセージの正常受信	bCan_RXERR を 1 ずつデクリメント
カウントが以前エラーパッシブトリガレベルの 127 を超えた状態での、メッセージの正常受信	bCan_RXERR を自動的に 119 と 127 の間の値にセット
アクティブエラーフラグまたはオーバロードフラグを送信後に、14 番目の連続ドミナントビットを受信	この時点および 8 個の連続ドミナントビットの各追加シーケンス後の両方で、bCan_RXERR および bCan_TXERR を 8 ずつインクリメント
パッシブエラーフラグを送信後に 8 番目の連続ドミナントビットを受信	
トランスミッタがエラーフラグを送信	bCan_TXERR を 8 ずつインクリメント
トランスミッタがアクティブエラーフラグまたはオーバロードフラグ送信中にビットエラーを検出	bCan_TXERR を 8 ずつインクリメント
トランスミッタが正常にメッセージを送信	bCan_TXERR を 1 ずつデクリメント

6.5.11　送信バッファレイアウト

送信バッファは、ディスクリプタフィールドとデータフィールドに分けられます。ディスクリプタフィールドの最初のバイトはフレーム情報を保持します。ここに、フレームフォーマット（SFF または EFF）、リモートフレームまたはデータフレームのどちらであるか、およびデータ長が記述されます。これに、SFF の場合は 2 つの識別子バイトまたは EFF メッセージの場合は 4 つの識別子バイトのいずれかが続きます。
データフィールドは最大 8 データバイトを含みます。

表 6.37　送信フレームフォーマットの説明

標準フレームフォーマット（SFF）		拡張フレームフォーマット（EFF）	
CAN オフセット	フィールド	CAN オフセット	フィールド
12’h040	TX フレーム情報	12’h040	TX フレーム情報
12’h044	TX 識別子 1	12’h044	TX 識別子 1
12’h048	TX 識別子 2	12’h048	TX 識別子 2
12’h04C	TX データバイト 1	12’h04C	TX 識別子 3
12’h050	TX データバイト 2	12’h050	TX 識別子 4
12’h054	TX データバイト 3	12’h054	TX データバイト 1
12’h058	TX データバイト 4	12’h058	TX データバイト 2
12’h05C	TX データバイト 5	12’h05C	TX データバイト 3
12’h060	TX データバイト 6	12’h060	TX データバイト 4
12’h064	TX データバイト 7	12’h064	TX データバイト 5
12’h068	TX データバイト 8	12’h068	TX データバイト 6
12’h06C	使用しない	12’h06C	TX データバイト 7
12’h070	使用しない	12’h070	TX データバイト 8

6.5.11.1　送信バッファのディスクリプタフィールド

送信バッファのディスクリプタフィールドのビットレイアウトを下記に示します。最初に SFF、次に EFF のビットレイアウトを示します。ディスクリプタフィールドの異なる要素は、後述の項で説明します。

- FF：フレームフォーマット（FF）を参照
- RTR：リモート要求（RTR）を参照
- DLC：データ長コード（DLC）を参照
- ID：識別子（ID）を参照

表 6.38　標準送信フレーム（SFF）フォーマットの説明

CAN オフセット	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
12'h040	FF	RTR	X 注 1	X 注 1	DLC.3	DLC.2	DLC.1	DLC.0
12'h044	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
12'h048	ID.20	ID.19	ID.18	X 注 2	X 注 1	X 注 1	X 注 1	X 注 1

表 6.39　拡張送信フレーム（EFF）フォーマットの説明

CAN オフセット	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
12'h040	FF	RTR	X 注 1	X 注 1	DLC.3	DLC.2	DLC.1	DLC.0
12'h044	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
12'h048	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13
12'h04C	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5
12'h050	ID.4	ID.3	ID.2	ID.1	ID.0	X 注 2	X 注 1	X 注 1

注1.　値は Don't care ですが、セルフ受信またはセルフテストのオプションを使用する場合、受信バッファと互換性を持たせるため "0" にすることを推奨します。

注2.　値は Don't care ですが、セルフ受信またはセルフテストのオプションが使用される場合、受信バッファで使用される RTR ビットと一致させることを推奨します。

6.5.11.2　フレームフォーマット（FF）

送信されるフレームフォーマットのタイプを FF ビットで選択します。

- "1" は拡張フレームフォーマット（EFF）を選択します
- "0" は標準フレームフォーマット（SFF）を選択します

6.5.11.3　リモート要求（RTR）

RTR ビットは、（CAN プロトコルで定義されているように）リモートフレームかデータフレームのどちらかといった、フレームの識別に使用されます。

- "1" は、リモートフレーム（別のノードからのデータの要求）を示します
- "0" は、データフレームを示します

6.5.11.4　データ長コード（DLC）

DLC[3:0]ビットは、送信されているメッセージに含まれるデータバイト数の指定に使用されます。

1 フレームに含まれるデータバイトの最大数は 8 であるため、8 より大きい DLC [3:0]の値は、自動的に 8 として解釈されます。

リモートフレーム送信の場合、データバイトはローカルホストから送信されませんが、2 つの CAN コントローラがリモートフレーム送信を同じ識別子で同時に開始する場合、バスエラーを避けるためにリモートフレームのデータ長が指定されることに注意してください。

6.5.11.5　識別子（ID）

識別子は、アクセプタンスフィルタリング用にレシーバで使用されるメッセージ名として機能し、バスアクセスの優先順位も決定します。

識別子のバイナリ値が低いほど、優先度が高くなります。

標準フレームフォーマット（SFF）では、識別子は 11 ビット（ID.28～ID.18）で構成されます。

拡張フレームフォーマット（EFF）メッセージでは、識別子は 29 ビット（ID.28～ID.0）で構成されます。ID.28 は最上位ビットであり、バスで最初に送信されます。

6.5.11.6　データフィールド

データフィールドは、データ長コードで定義されたデータバイト数を構成する必要があります。CAN オフセット 12’h04C（SFF）もしくは CAN オフセット 12’h054（EFF）にあるデータバイト 1 の最上位ビットが最初に送信されます。

6.5.12　受信バッファレイアウト

受信バッファレイアウトは、前の項で説明した送信バッファと同様です。実際、使用される構成は、特に送信バッファのレイアウトと互換性を持たせるように選択されています。

ここでも、標準フレームフォーマット（SFF）メッセージと拡張フレームフォーマット（EFF）メッセージを区別することが重要です。

受信バッファは、ディスクリプタとデータフィールドに分けられます。ディスクリプタフィールドの最初のバイトはフレーム情報を保持します。ここに、フレームフォーマット（SFF または EFF）、リモートフレームまたはデータフレームのどちらであるか、およびデータ長が記述されます。これに、SFF の場合は 2 つの識別子バイトまたは EFF メッセージの場合は 4 つの識別子バイトのいずれかが続きます。データフィールドは最大 8 データバイトを含みます。

表 6.40　受信フレームフォーマットの説明

標準フレームフォーマット（SFF）		拡張フレームフォーマット（EFF）	
CAN オフセット	フィールド	CAN オフセット	フィールド
12’h040	RX フレーム情報	12’h040	RX フレーム情報
12’h044	RX 識別子 1	12’h044	RX 識別子 1
12’h048	RX 識別子 2	12’h048	RX 識別子 2
12’h04C	RX データバイト 1	12’h04C	RX 識別子 3
12’h050	RX データバイト 2	12’h050	RX 識別子 4
12’h054	RX データバイト 3	12’h054	RX データバイト 1
12’h058	RX データバイト 4	12’h058	RX データバイト 2
12’h05C	RX データバイト 5	12’h05C	RX データバイト 3
12’h060	RX データバイト 6	12’h060	RX データバイト 4
12’h064	RX データバイト 7	12’h064	RX データバイト 5
12’h068	RX データバイト 8	12’h068	RX データバイト 6
12’h06C	使用しない	12’h06C	RX データバイト 7
12’h070	使用しない	12’h070	RX データバイト 8

受信バッファのディスクリプタフィールドのビットレイアウトを下記に示します。最初に SFF、次に EFF のビットレイアウトを示します。ディスクリプタフィールドの異なる要素は、後述の項で説明します。

- FF：フレームフォーマット（FF）を参照
- RTR：リモート要求（RTR）を参照
- DLC：データ長コード（DLC）を参照
- ID：識別子（ID）を参照

表 6.41　拡張送信フレーム（EFF）フォーマットの説明

CAN オフセット	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
12’h040	FF	RTR	0	0	DLC.3	DLC.2	DLC.1	DLC.0
12’h044	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
12’h048	ID.20	ID.19	ID.18	RTR	0	0	0	0

表 6.42　拡張受信フレーム（EFF）フォーマットの説明

CAN オフセット	ビット 7	ビット 6	ビット 5	ビット 4	ビット 3	ビット 2	ビット 1	ビット 0
12'h040	FF	RTR	0	0	DLC.3	DLC.2	DLC.1	DLC.0
12'h044	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
12'h048	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13
12'h04C	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5
12'h050	ID.4	ID.3	ID.2	ID.1	ID.0	RTR	0	0

注　意

フレーム情報バイト（CAN オフセット 12'h040）で受信されたデータ長コードは、送信されたデータの長さを示し、8 バイトより大きい場合があります。ただし、データバイトの最大数は 8 です。

6.5.13　ビット期間パラメータとバスタイミングパラメータ

バスタイミングパラメータは CAN コントローラを CAN バスで使用されるビットレートに設定し、各ビット期間内に受信したビットストリームがサンプリングされるポイントを設定します。ビットストリームの再同期化によって他のノードで生成されたビットレートの変動を CAN コントローラが補正できる度合いを指定します。

他のノードで生成されたビットレートの変動、およびバス上と CAN ノード内両方の物理的な遅延時間に対応するために、ビット期間は、次のもので構成されます。

- 同期セグメント
  - 同期セグメントは、ビットエッジの到着が期待されるビット期間の部分を示します。
- 伝播セグメント
  - 伝播セグメントは、物理的な遅延時間を補正できるビット時間の部分を示します。
- 2 つの位相バッファ
  - 2 つの位相バッファは、サンプリングポイントを囲んでおり、ビットエッジが同期セグメント外に到着するときに、受信ビットストリームの再同期化のために必要に応じて短くしたり長くしたりされます。

これらの各セグメントの長さは、“Time Quanta”(Tq)の数として定義されます。

- 同期セグメントは常に、1Tq です。
- 伝播セグメントは 1Tq～8Tq です。
- 2 つの位相バッファは 1Tq～8Tq です。

位相バッファを長くしたり短くしたりできる最大数も、同期ジャンプ幅として定義されます。これは、1Tq～4Tq に限られ、2 つの位相バッファのいずれかよりも長くなることはできません。

CAN コントローラで使用されるタイミングパラメータは、2 つのバスタイミングレジスタ（rCan_BTR0 および rCan_BTR1）経由で選択されます。それとともに、ビット期間の構造も定義します。

- 両方のレジスタはリセットモードで書き込みのみ可能です。動作モードでは、読み出しのみが可能です。

rCan_BTR0 は以下を定義します。

- ボーレートプリスケーラ（bCan_BRP）は、CAN 用クロックの“Time Quantum”Tq を、CAN_HCLK 周期の倍数として定義します。
- 現在の送信と再同期するためにビット期間を短くしたり長くしたりすることのできる、Time Quanta 数（bCan_SJW）

rCan_BTR1 は以下を定義します。

- サンプルが取得されるまでの Time Quanta 数およびサンプルが取得される後の Time Quanta 数（bCan_TSEG1 および bCan_TSEG2）
  - bCan_TSEG1 は同期セグメントとサンプルポイントの間の時間を表します（つまり、伝播セグメントプラス 1st 位相バッファ）。
  - bCan_TSEG2 は、サンプルポイントとビット期間の終わりとの間の時間を表します（つまり、2nd 位相バッファ）。
- 取得されるサンプル数（bCan_SAM）、1 または 3

ボーレートプリスケーラは、CAN 用クロックの“Time Quantum”Tq を、CAN_HCLK 周期の倍数として定義します。

CAN 用クロックの Time Quantum は以下で指定されます。

Tq＝2×Tperiod (CAN_HCLK)×(32×bCan_BRP[5]＋16×bCan_BRP[4]
＋8×bCan_BRP[3]＋4×bCan_BRP[2]＋2×bCan_BRP[1]＋bCan_BRP[0]＋1)

Tseg1 および Tseg2 は、サンプルが取得されるまでの Time Quanta 数およびサンプルが取得される後の Time Quanta 数を指定することでビット期間の長さを定義します。

Tsyncseg＝1×Tq
Tseg1＝Tq×(8×bCan_TSEG1[3]＋4×bCan_TSEG1[2]＋2×bCan_TSEG1[1]＋bCan_TSEG1[0]＋1)
Tseg2＝Tq×(4×bCan_TSEG2[2]＋2×bCan_TSEG2[1]＋bCan_TSEG2[0]＋1)

注　意

- **理論上、これらのレジスタ設定により、4Tq と 25Tq の間のビット期間を定義できます。**
- **ただし、実際使用されるビット期間は、BOSCH 標準に従うことが必要で、長さ 8Tq と 25Tq の間のビット期間を定義します。**

確実に CAN 2.0 仕様に合った使い方をするため、次の簡単なルールに従ってください。

- “再同期ジャンプ幅”（SJW）を 1Tq～4Tq の間に設定
- “伝播セグメント”を 1Tq～8Tq に設定
- “1st 位相バッファ”を“SJW”～8Tq に設定
- “2nd 位相バッファ”を、“1st 位相バッファ”から“2nd 位相バッファ”＋2Tq の間で設定

CAN コントローラレジスタへの適用：

- bCan_SJW＝SJW－1
- bCan_TSEG1＝“伝播セグメント”＋“1st 位相バッファ”－1
- bCan_TSEG2＝“2nd 位相バッファ”－1

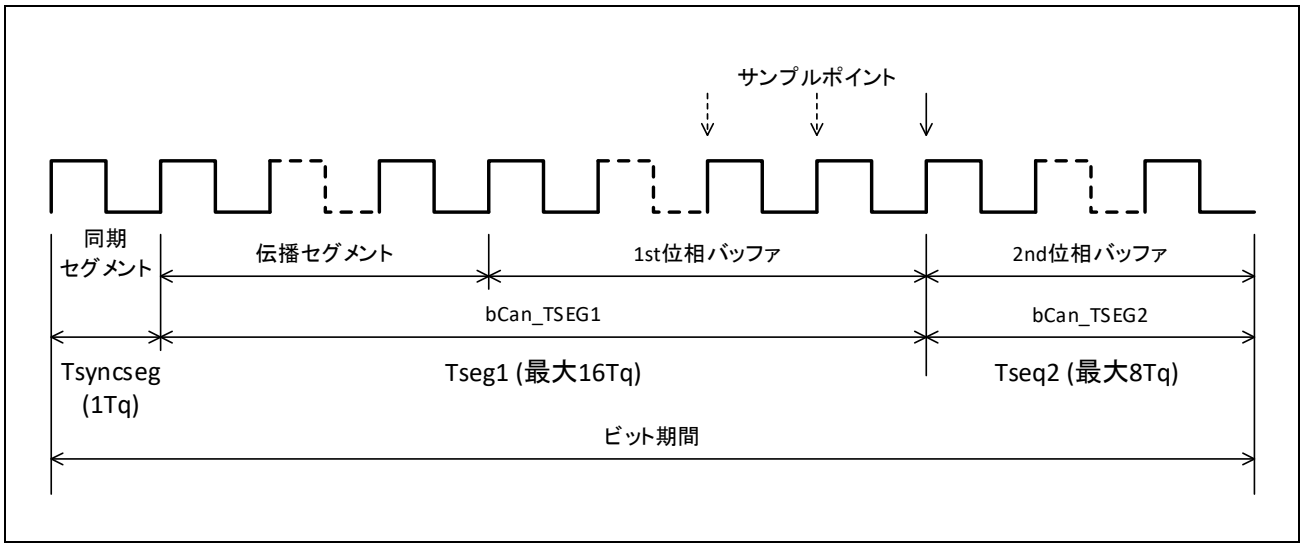


図 6.3　CAN：ビット期間、Tsyncseg、Tseg1、Tseg2 の一般的な構造

6.5.14　リセットモード

CAN コントローラを動作させるための設定は、リセットモード中にセットされます。CAN コントローラはハードウェアリセットの直後もしくはソフトウェアリセット（モードレジスタ（rCan_MOD）のリセットモードビット（bCan_RM）が“1”（ソフトウェアリセット）にセットされた結果）によってリセットモードに移行します。

リセットモード中、CAN コントローラの下記の動作をセットすることが可能です。

- 適用されるバスタイミングパラメータ（CAN バス上で使用されるボーレートを選択）
- 受信したメッセージに適用されるアクセプタンスフィルタ
- 必要な割り込み
- 目標エラーワーニングリミット
- 必要な出力モード

CAN コントローラはリセットモードビット（bCan_RM）をクリアすることで動作モードに戻ります。CAN コントローラは、（リセットがホスト CPU かハードウェアによってトリガされた場合）バス上で 11 個のレセシブビットの“バスフリー”シーケンスを検出するまで、またはバスステータスが“バスオフ”に移行することに起因する場合は、本シーケンスを 128 回検出するまで、データを送信も受信もしないことにも注意する必要があります（rCan_TXERR レジスタを参照）。

動作中：

- モードレジスタ（bCan_RM）でリセットモードを設定することで、メッセージの現在の送信／受信の中断が発生し、CAN コントローラがリセットモードになります（これは、内部バスクロックの次のポジティブエッジで発生します）。
- たとえば送信エラーカウンタ（rCan_TXERR）が 255 を超えた場合などに、バスステータスが“バスオフ”（bCan_BS が“1”にセット）に移行すると、CAN コントローラもリセットモードに移行します。

注　意

- “バスオフ”条件のためにリセットモードに入った場合、エラーワーニング割り込み（bCan_EI）が設定されます（この割り込みを許可する場合、bCan_EIE ビットをセット）。
- “バスオフ”条件のためにリセットモードに入った場合、受信エラーカウンタ（rCan_RXERR）はクリアされます。また、送信エラーカウンタ（rCan_TXERR）は 127 に初期化され、11 個の連続したレセシブビットが 128 回発生する時間に相当する、CAN 定義のバスオフリカバリ時間のカウントダウンに使用されます。
- CAN コントローラは、データが送受信される前に、リセットモードから遷移する必要があります。

6.5.15　同期フレーム

“同期フレーム”送信メカニズムを使用する場合のみ：

CAN リンクを使用することでドライブとの通信が可能になります。構成は 8 ドライブに制限され、目的とする動作は、2ms 以内で 4 ドライブをリフレッシュ（または 4ms 以内で 8 ドライブ）すること、さらに、同期フレーム（“同期フレーム”）でそれらを同期することです。“同期フレーム”は、可能なかぎり小さなジッタ（上限は 70µs）で定期的に送信される必要があります。

“同期フレーム”は、データのない標準フレームです。CAN バス上での長さは、44 ビットで、次のように分割します：

- 1 ビット：スタートビット（暗黙的）
- 12 ビット：アービトレーションフィールド
- 6 ビット：コントロールフィールド
- 0 ビット：データフィールド（最大 64 ビットまで可能）
- 16 ビット：CRC フィールド（暗黙的）
- 2 ビット：ACK フィールド（暗黙的）
- 7 ビット：フレームの終わり（暗黙的）
- ＋フレーム内容に従ったビットスタッフィング

6.5.15.1　CANopen 同期フレーム設定

“同期フレーム”に関して、CAN コントローラの主な機能は以下のとおりです。

- ビット期間単位に周期的な専用タイムベース（プログラム可能）でコントロールされる“同期フレーム”の送信：
  - “同期フレーム”周期（rCan_SyncPeriod レジスタの bCan_SyncPeriod ビット)
  - “同期フレーム”専用のタイムウィンドウ（rCan_SyncPeriod レジスタの bCan_SyncMaskFrameTime ビット）
  - CAN コントローラが以下のアクションを実行できるようにする：
    CPU によって rCan_WrTransmitBuffer レジスタに書き込まれた最後のフレームの送信
    本フレームでの潜在的な再試行の管理（アービトレーションロスト）
    「**6.5.9 バスアービトレーション（バスの調停）**」を参照してください。
    rCan_SyncTransmitBuffer レジスタで初期設定された“同期フレーム”の送信
  - 「**図 6.4 CANopen：周期的同期フレームの送出**」を参照してください。
- 送信される“同期フレーム”は、CAN の“同期フレーム送信バッファ”（rCan_SyncTransmitBuffer レジスタ）に書き込まれ、bCan_SyncRunStop ビット（“同期フレーム”送出の開始と終了）と bCan_SyncMaskFrame ビット（“同期フレーム”送出用に予約されたタイムウィンドウ）によってコントロールされます。
  - 「**図 6.5 CANopen：周期的同期フレームの開始と停止**」を参照してください。
- “同期フレーム”が送信されると以下によってコントロールされる CPU への割り込み（CAN_Int）を送信します：
  - rCan_SyncStatusInt レジスタでの割り込みステータス
  - rCan_SyncMaskInt レジスタでの割り込みマスク
  - rCan_SyncClearInt レジスタでの割り込み ACK 応答

- 「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。

● 特定の書き込みシーケンスでコントロールされる機能モード

- rCan_SyncClearSetRunStop への 32’h2052_756E の書き込み
  “RUN モード”
  “同期フレーム”の送出の有効化
  “1”を bCan_SyncRunStop ビットにセット
  bCan_SyncMaskFrame の生成
  “同期フレーム”が送信されるときに割り込みが発生します
- rCan_SyncClearSetRunStop への 32’h5374_6F70 の書き込み
  “ストップモード”
  “同期フレーム”の送出の無効化
  “0”へ bCan_SyncRunStop ビットをクリア
  “0”へ bCan_SyncMaskFrame ビットをクリア
  “同期フレーム”管理用のタイマをすべてクリア

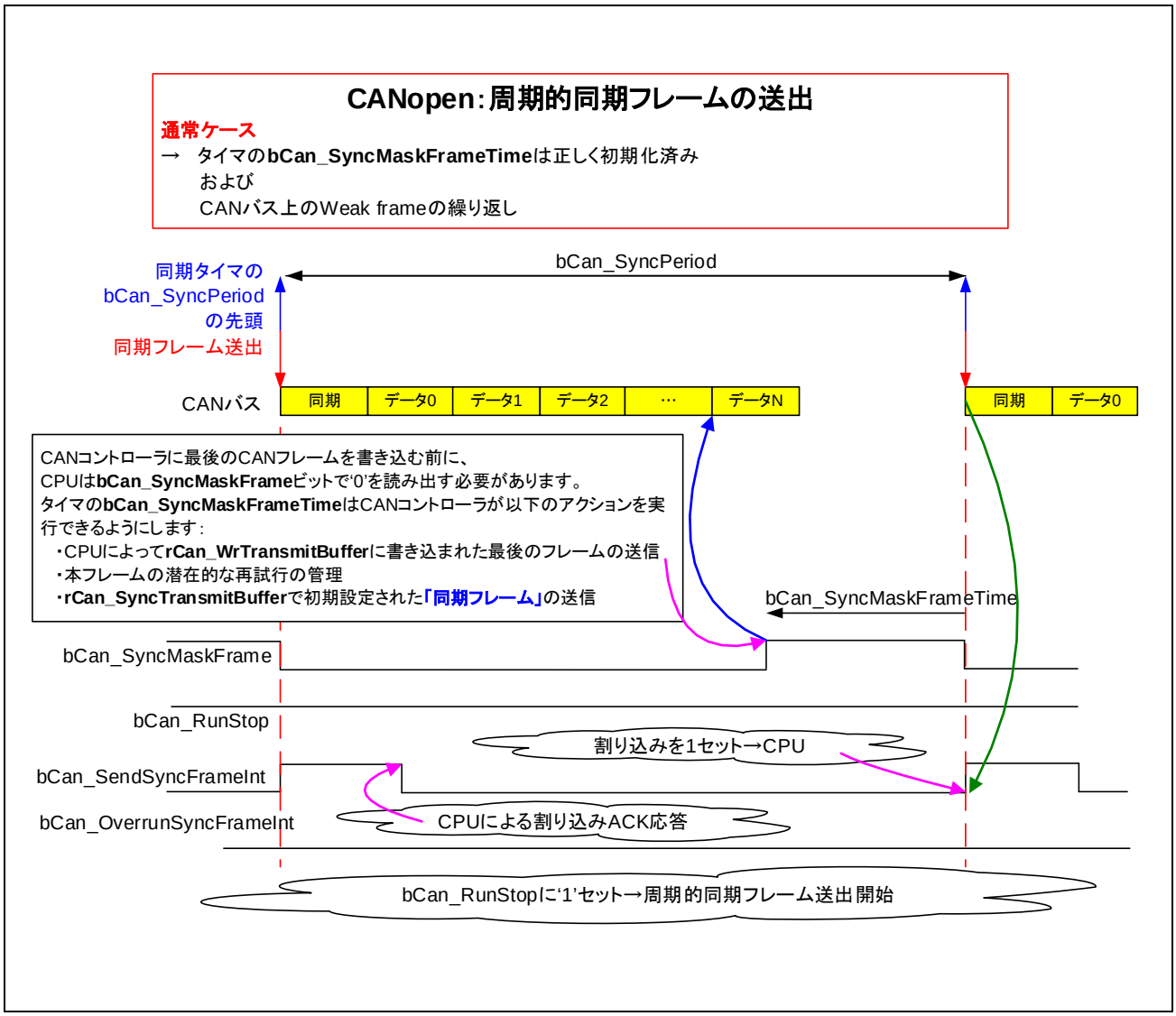


図 6.4　CANopen：周期的同期フレームの送出

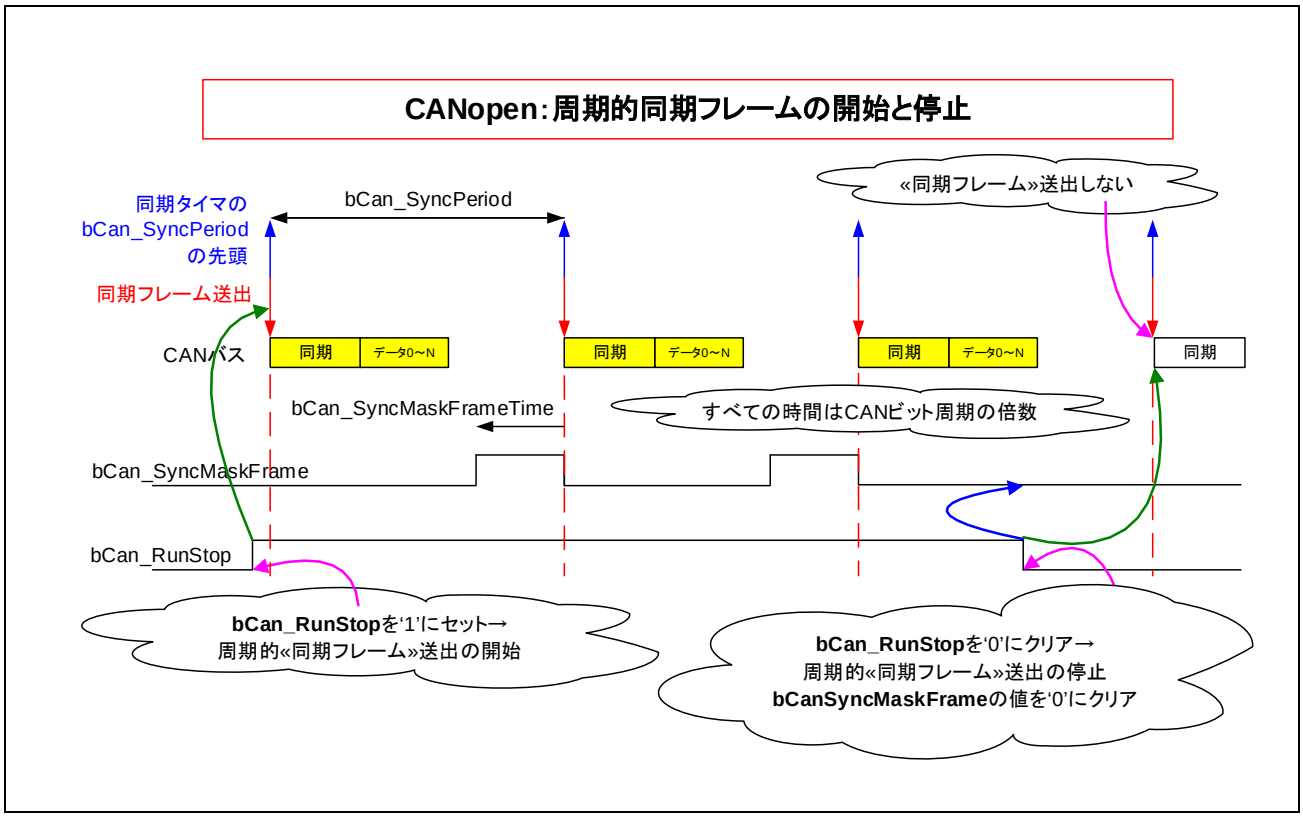


図 6.5　　CANopen：周期的同期フレームの開始と停止

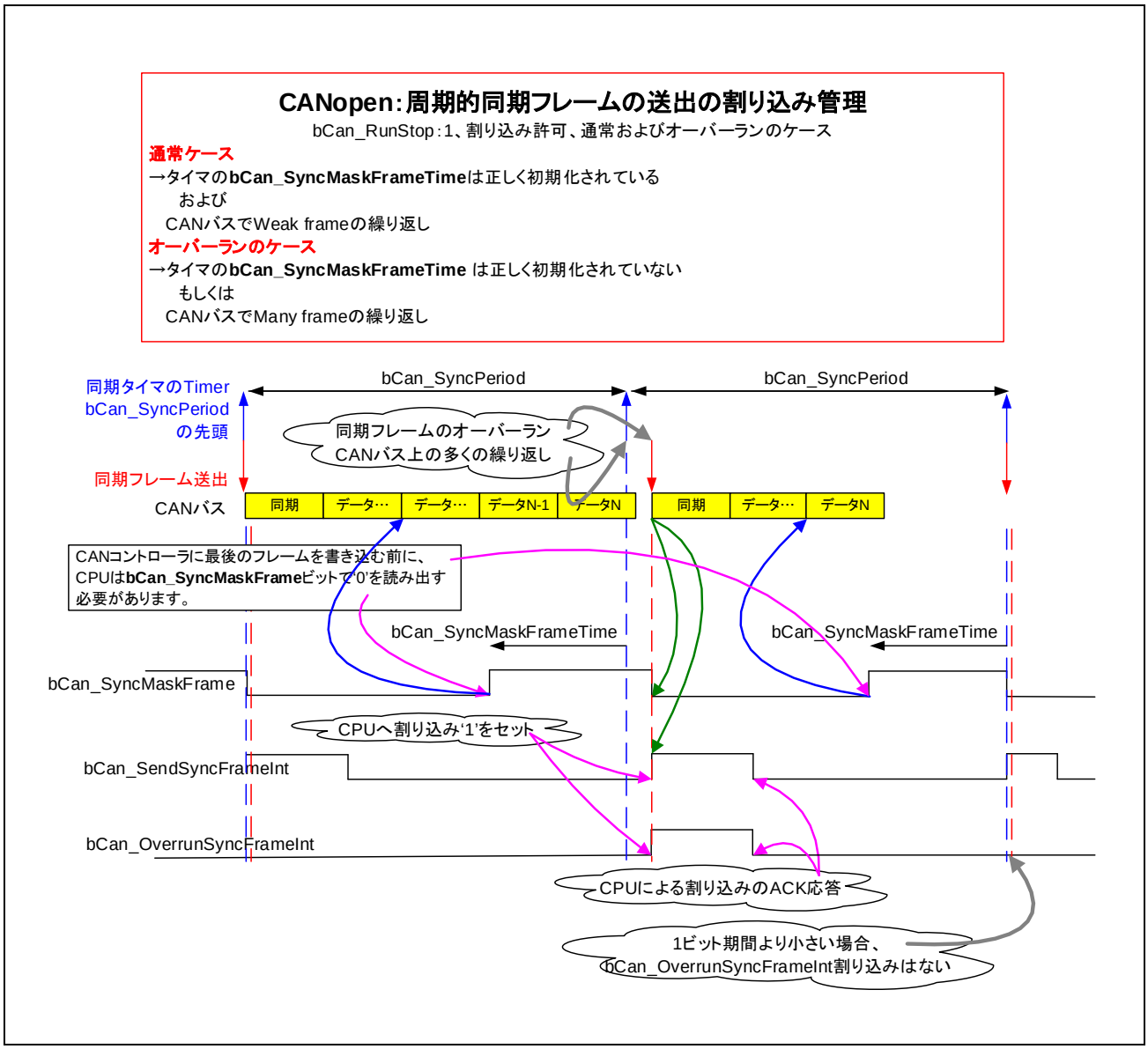


図 6.6　　CANopen：周期的同期フレームの割り込み管理

補助機能の実装により、ユーザは、ネットワーク上で“同期フレーム”を送信することなく、同期フレーム割り込みのみを持つこともできます。

本機能を有効にするには、3 つの方法があります：

- rCan_SyncStatus レジスタで、bCan_TimerOnlyMode をセットする。このオプションの使用によって同期フレームの設定と開始を非アクティブ化できますが、割り込みシステムは有効のままです。bCan_SyncMaskFrameTime が有効なときはいつでも本モードで、CAN コントローラにアクセスできます。
- rCan_SyncStatus レジスタの bCan_TimerOnlyIfBusOff をセットする。このオプションの使用によって、rCan_SR レジスタの bCan_BS がセットされている場合のみ、同期フレームの設定と開始を非アクティブ化します。そのようなことが発生した場合、システムは自動的に rCan_SyncStatus の bCan_TimerOnlyMode をセットします。
- rCan_SyncPassiveError レジスタを“0”以外の値にセットする。“パッシブエラー検出システム”がアクティブな場合は、“同期フレーム”の遅延した開始と、現在の“同期周期”の終わりの間のビット期間の

数をチェックします。この数が bCan_SyncPassiveError より小さいと、ただちにシステムは自動的に rCan_SyncStatus の bCan_TimerOnlyMode をセットします。

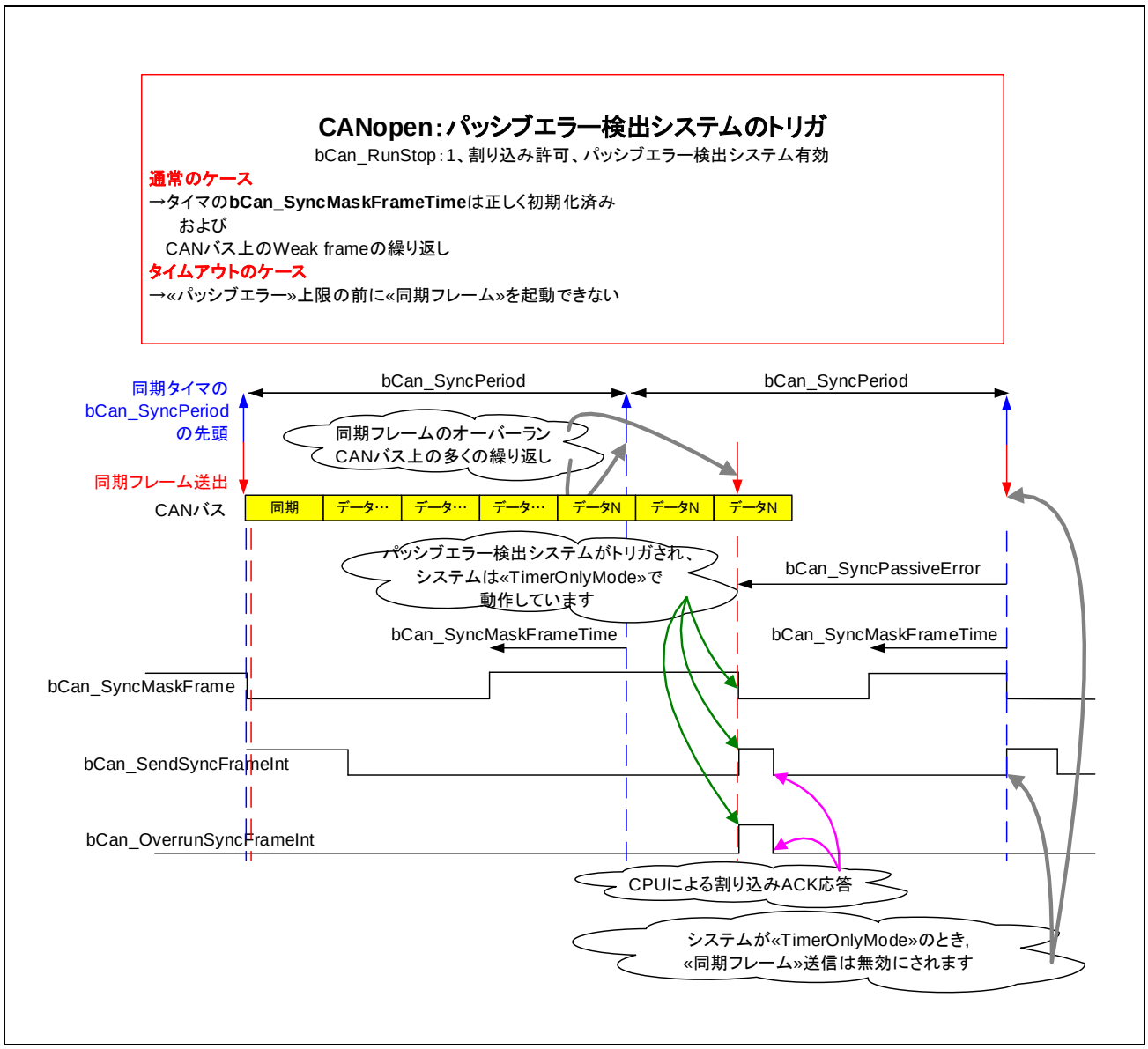


図 6.7　CANopen：パッシブエラー検出システムのトリガ

6.5.15.2　CANopen：“同期フレーム”の送出

“同期フレーム”用のタイムウィンドウの考察

本タイムウィンドウにより、CAN コントローラは下記のアクションを実行できます。

- CPU によって rCan_WrTransmitBuffer レジスタに書き込まれた最終フレームの送信
- 本フレーム上の潜在的なリトライの管理（アービトレーションロスト）
- rCan_SyncTransmitBuffer レジスタで初期化された“同期フレーム”の送信

本タイムウィンドウのサイズを計算するには、ユーザは下記を考慮に入れる必要があります。

- CAN バスで、rCan_WrTransmitBuffer に含まれる潜在的な最終フレームをリトライなしで送信するために必要な時間
- CAN バス上で 2 回または 3 回のリトライの管理に必要な時間（アービトレーションロスト）
  - 「**6.5.9 バスアービトレーション（バスの調停）**」を参照してください。
- CAN コントローラが、rCan_SyncTransmitBuffer で初期化された“同期フレーム”を、rCan_WrTransmitBuffer レジスタにコピーし、rCan_CMR レジスタの書き込みコマンドを送信するために必要な時間
  8 ビット期間単位かかります。

“同期フレーム”を送信する前、CAN コントローラは次のアクションを実行します（“同期周期”の終わりの前に、8 ビット期間単位の開始）：

［ステップ 1］bCan_BS ビットの読み出し（バスステータス）

1’b0 ：“バスオン”モード。CAN コントローラはバスアクティビティに関与しています。ステップ 2 へ進んでください。

1’b1 ：“バスオフ”モード
bCan_TimerOnlyIfBusOff が“0”にクリアされている場合、“0”へ bCan_SyncRunStop ビットをクリアしてください。すべての“同期フレーム”システムは無効にされ、送信は中断されます。
bCan_TimerOnlyIfBusOff が“1”にセットされている場合、CAN コントローラが自動的に“1”を bCan_TimerOnlyMode ビットにセットします。“同期フレーム”送信システムが無効にされ、送信は中断され、割り込みシステムがまだ有効です。

［ステップ 2］bCan_TBS ビットの読み出し（送信バッファステータス）

1’b1 ：送信バッファ解放。バッファは新しいフレームを書き込むことができます。ステップ 3 へ進んでください。

1’b0 ：送信バッファロック。送信フレームは実行中です。ステップ 1 に戻ってください。

［ステップ 3］rCan_SyncTransmitBuffer から rCan_WrTransmitBuffer へ、“同期フレーム”をコピー

“同期周期”の終わりで bCan_TR ビットもしくは bCan_SRR ビット（bCan_SyncMode ビットに依存）に書き込むことで、“同期フレーム”を送信します。

［ステップ 4］bCan_BS ビットの読み出し（バスステータス）

1’b0 ：“バスオン”モード。“同期フレーム”割り込みが送信され、bCan_SyncMaskFrame ビットが無効になる前に、bCan_TS が“1”にセットされるまで待ちます。

1’b1 ：“バスオフ”モード
bCan_TimerOnlyIfBusOff が“0”にクリアされている場合、“0”へ bCan_SyncRunStop ビットをクリアしてください。すべての“同期フレーム”システムは無効にされ、送信は中断されます。
bCan_TimerOnlyIfBusOff が“1”にセットされている場合、CAN コントローラが自動的に“1”を bCan_TimerOnlyMode ビットにセットします。“同期フレーム”送信システムが無効にされ、送信は中断され、割り込みシステムがまだ有効です。

注　意

- “同期フレーム”システムが有効で、“TimerOnlyMode”ではない場合、CPU は、“同期周期”の最終の 8 ビット期間 CAN コントローラへの書き込みアクセスをしてはなりません。タイムベース bCan_SyncMaskFrameTime の設定は、8 ビット期間単位に、bCan_SyncMaskFrame での最終チェックへのアクセス（読み出し／書き込み）シーケンスリンクを実行するために必要な時間を足した時間以上でなければなりません。
- “パッシブエラー検出システム”もステップ 1 とステップ 4 でチェックされます。
- ステップ 1 とステップ 4 の間のバスオフ検出は、bCan_TimerOnlyMode が“1”にセットされているときは無効です。
- “同期フレーム”送信に必要な時間が不足している場合、オーバーラン“同期フレーム”割り込みが発生します。次の状態であるか確認してください：
  - bCan_SyncMaskFrameTime ビットで定義されたタイムウィンドウが正しく初期化されている。
  - CAN バスの整合性、CAN バス上の多くの繰り返しフレーム（アービトレーションロスト）
  - 他の CAN マスタが“同期周期”の終わりにフレームを送信していない。

「**6.5.9 バスアービトレーション（バスの調停）**」を参照してください。
「**図 6.6 CANopen：周期的同期フレームの割り込み管理**」を参照してください。

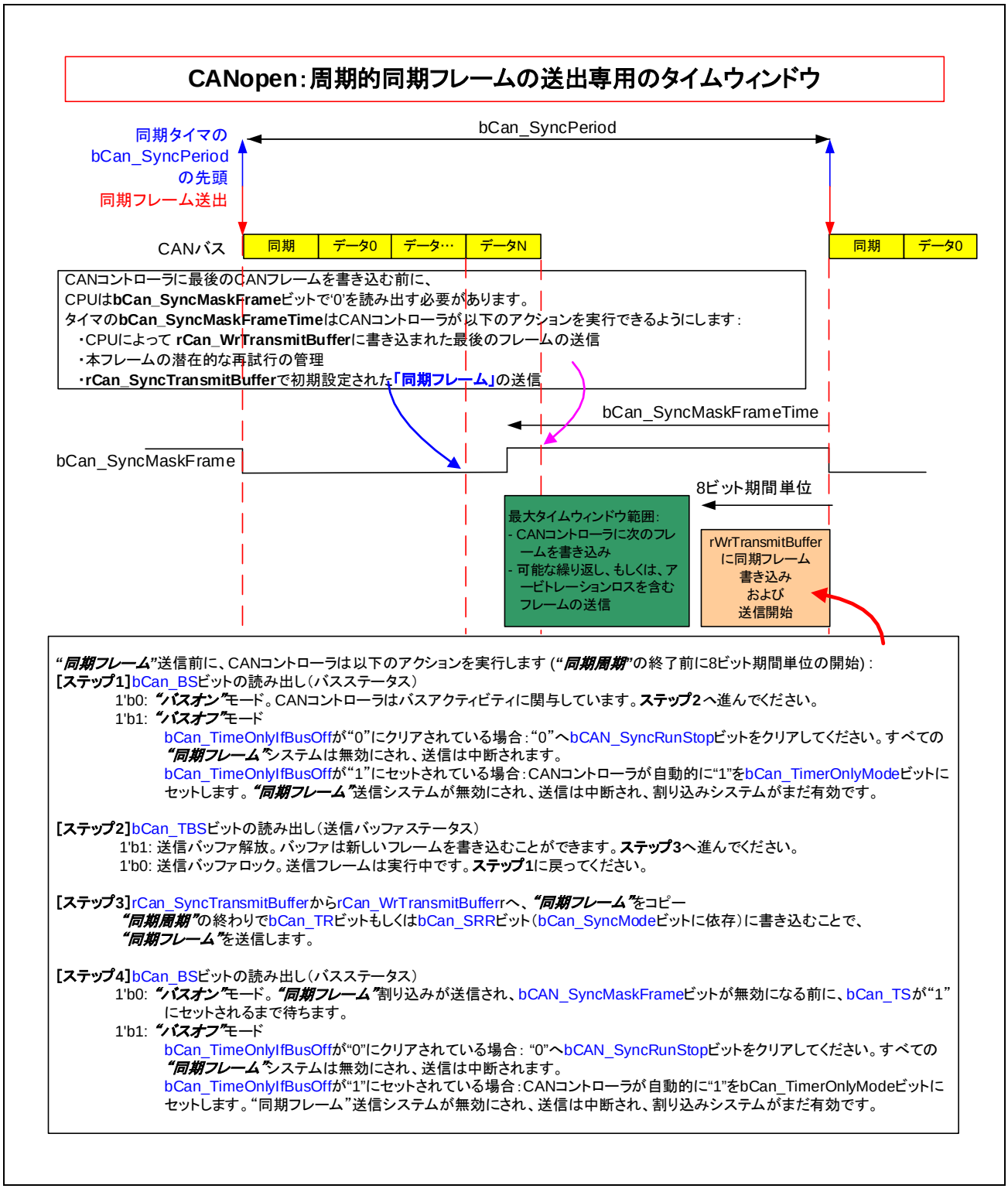


図 6.8　CANopen：周期的同期フレームの送出専用のタイムウィンドウ

6.5.16　CAN コントローラとリファレンス Philips SJA1000 デバイスの違い

本項では、本 CAN コントローラとリファレンスの Philips SJA1000 デバイスとの違いをまとめます。

動作モード：

- リファレンスデバイスには、BasicCAN と PeliCAN と呼ばれる 2 種類の動作モードがあります。本 CAN コントローラには単一の動作モードがあり、SJA1000 の PeliCAN モードと広く互換性があります。
- 本 CAN コントローラと Philips SJA1000 の PeliCAN モードとの違いを以下に示します。

送信されたメッセージの扱い

- 本 CAN コントローラはリファレンスデバイスと異なり送信されたメッセージを受信バッファにコピーしません。
- 代わりに、送信されたメッセージを CAN オフセット 12’h180～12’h1B0 からリードバックできます。

モードレジスタ：

- リファレンスデバイスでは、rCan_MOD[3:1]への書き込みアクセスはリセットモードに制限されます。本 CAN コントローラでは、これらのビットはリセットモードか動作モードのいずれかから書き込むことが可能です。

出力コントロールレジスタ：

- 本 CAN コントローラは、リファレンスデバイスと異なり rCan_OCR[7:2]経由で選択された出力電圧レベルと極性オプションをサポートしません。
- 本 CAN コントローラで、これらのビットは予約済みであり、読み出されると 0 が返されます。
- rCan_OCR[1:0]経由で選択できる出力モードは通常出力モードのみに制限されることに注意してください。

クロック分周器レジスタ：

- 本レジスタはサポートされていません。

AHB インタフェース：

- リファレンスデバイスのすべてのレジスタのアドレスはバイトアラインです。
- 本 CAN コントローラでは、すべてのレジスタのアドレスは 32 ビットアラインであり、ビット 31～24 のすべては 0 が読み出されます。

スリープモード：

- リファレンスデバイスは、スリープモードに入り内部ゲーティングでデバイスを無効にします。本 CAN コントローラでは代わりに、コアのほとんどを駆動する CAN_HCLK 信号をゲーティングできます。

6.6　特記事項

本ドキュメントの CAN の項は、Mentor Graphics Corporation の著作権と情報が含まれます。このことを考慮して、本ドキュメントのいかなるコピーも、ここに含まれるすべての著作権と所有権通知を保持します。疑義を避けるために付言すると、ここに記載されているいかなる部分も、暗示的、禁反言、その他により、次に該当するいかなるライセンスまたは権利を付与するものではありません。（i）Mentor Graphics またはいかなる第三者の特許権または商標、あるいは（ii）いかなる Mentor Graphics Corporation の著作権。

第7章　ADC コントローラおよび 12 ビット A/D コンバータ

7.1　概要

ADC コントローラの特長

- ソフトウェア変換要求
- 2 レベルの優先順位
- 同じ優先順位の変換要求が同時発生した場合のラウンドロビン管理
- 優先順位の高い変換要求を、動作中の優先順位の低い変換の終了時に挿入
- トリガが発生すると、動作中の変換の終了時に S&H（サンプル&ホールド）チャネルをサンプリングし、その後、グローバルラウンドロビン方式に従って変換
- DMA 接続
  - 選択した「変換終了」信号に基づく DMA トランザクションの開始
  - 2 チャネル排他使用可
- チャネルごとの変換終了情報を集約したステータスレジスタ
- 仮想チャネル機能

ADC コアの特長

- 最大 2 ユニット
- 分解能：12 ビット
- サンプリングレート：0.0625MSPS～1MSPS
- 逐次比較方式
- 最大変換時間：21ADC_CLK
- アナログ入力
  コアあたり 8 チャネル（5 標準チャネル ＋3 サンプル&ホールドチャネル）
- チャネルごとに独自の変換起動用入力トリガを装備。トリガは ADC コントローラで管理
- DNL：±1.0LSB（最大）（VAIN＝0.0V～AVDD、f_{CLK}＝20MHz の場合）
- INL：±4.0LSB（最大）（VAIN＝0.0V～AVDD、f_{CLK}＝20MHz の場合）
- 入力リーク電流：約 0.2μA（1 入力あたり）
- 低消費電力モード
- ADC クロック周波数：4MHz～20MHz

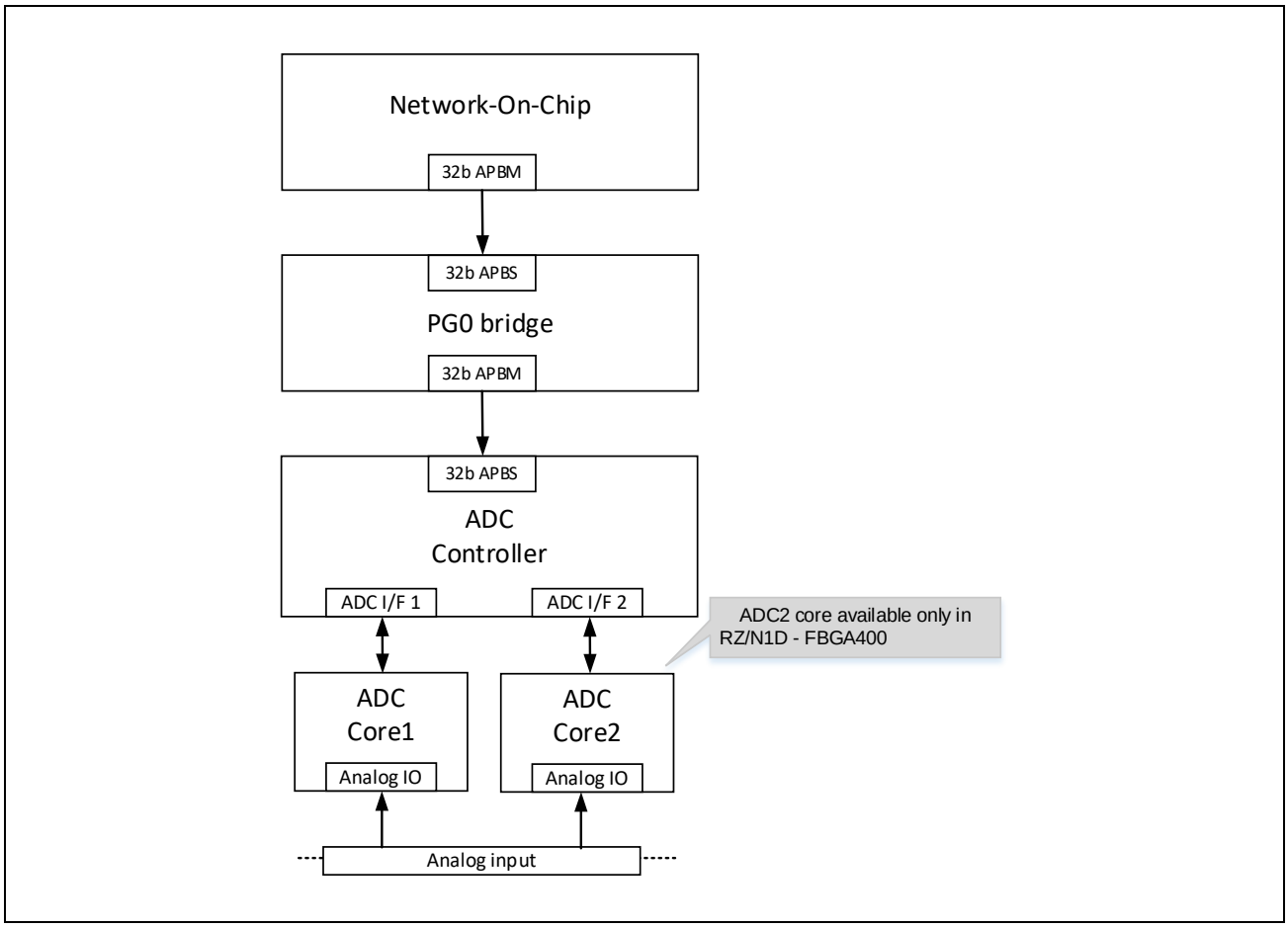


図 7.1　　ADC のインタフェースおよび接続

7.1.1　アナログバッファ

ADC コア同士は、端子数削減のため、アナログ端子のいくつかを共有しています。

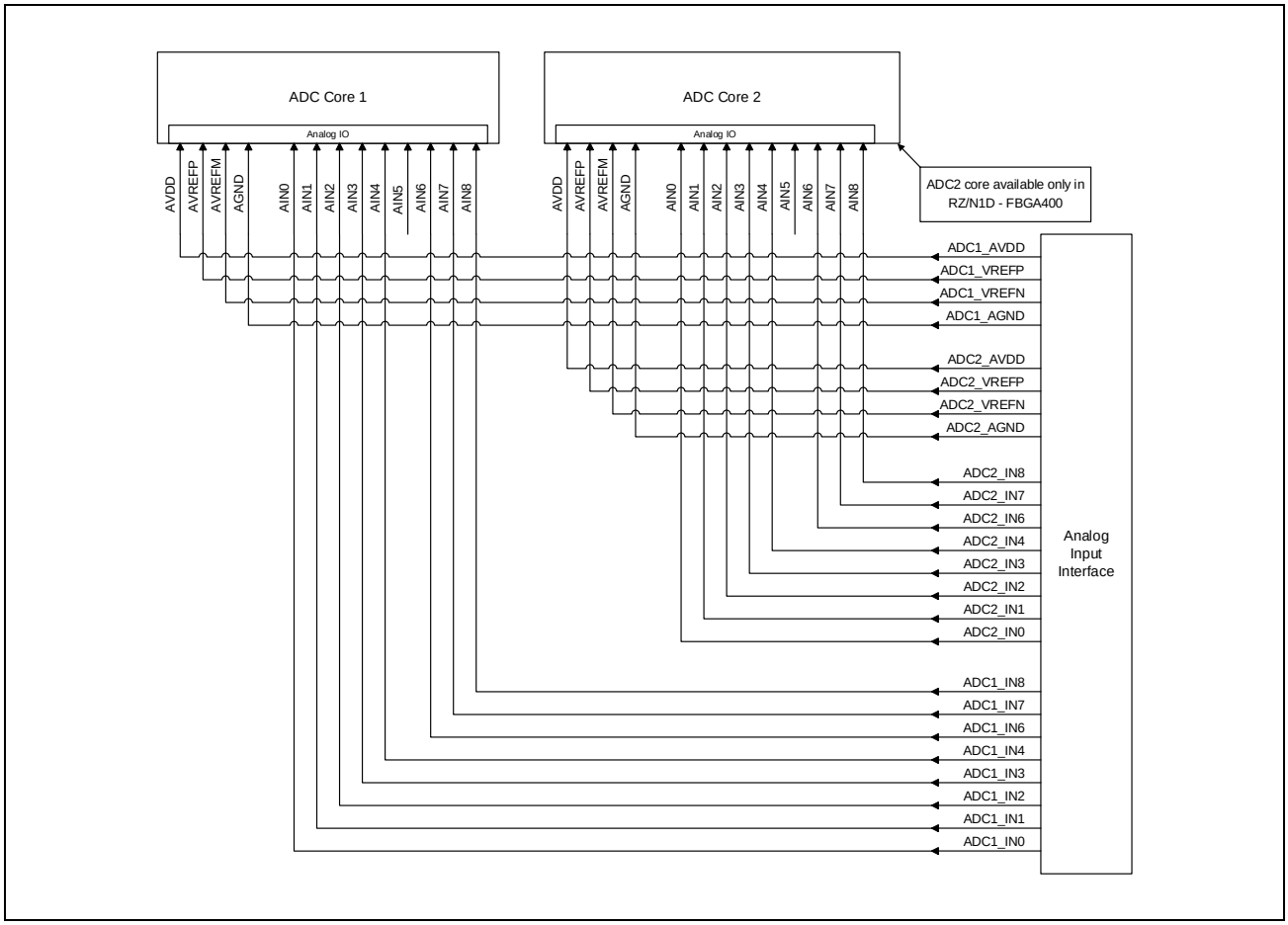


図 7.2　　ADC コアのアナログ信号

7.2　信号インタフェース

信号名	入力 出力	説明
クロック		
ADC_PCLK	入力	内部バスクロック（APB）
ADC_CLK	入力	ADC クロック
割り込み		
ADC_Int	出力	レベル検出割り込み出力、アクティブ High
外部信号		
ADC1_IN[8:6,4:0]	入力	ADC1 用のアナログ入力端子
ADC2_IN[8:6,4:0]	入力	ADC2 用のアナログ入力端子

7.3　レジスタマップ

注　意

利用可能な ADC の個数はパッケージによって異なります。
　RZ/N1D FBGA400：ADC1 および ADC2、その他：ADC1 のみ

7.3.1　レジスタマップ ADC1

注　意

すべてのパッケージで利用可能です。

表 7.1　A/D コンバータのレジスタマップ

アドレス	レジスタシンボル	レジスタ名
4006 5000h	rADC_INTSTATUS0	マスキング前割り込みステータス
4006 5004h	rADC_INTSTATUS1	マスキング後割り込みステータス
4006 5008h	rADC_INTCLR	割り込みクリア
4006 500Ch	rADC_INTMASK	割り込みマスク
4006 5010h	rADC_INTOVFSTATUS0	マスキング前割り込みオーバーフロー
4006 5014h	rADC_INTOVFSTATUS1	マスキング後割り込みオーバーフロー
4006 5018h	rADC_INTCLROVF	割り込みオーバーフロークリア
4006 501Ch	rADC_INTOVFMASK	割り込みオーバーフローマスク
4006 5020h	rADC_PENDING	動作開始保留中
4006 5024h	rADC_PENDINGOVF	動作開始保留中オーバーフロー
4006 5028h	rADC_PENDINGCLROVF	動作開始オーバーフロークリア
4006 502Ch	rADC_CONTROL	ADC 制御
4006 5030h	rADC_FORCE	ADC 要求
4006 5034h	rADC_SETFORCE	ADC 要求セット
4006 5038h	rADC_CLRFORCE	ADC 要求クリア
4006 503Ch	rADC_PRIORITY	ADC 優先モード
4006 5040h	rADC_CONFIG	ADC コンフィグレーション
4006 50ACh	rADC_ACQS	ADC サンプル＆ホールド制御
4006 50B0h＋4h×n	rADC_MASKLOCK[n] n＝0～3	マスクデータロック[n]
4006 50C0h＋4h×n	rADC_VC[n] n＝0～15	仮想チャネル[n]用 ADC コントロールレジスタ
4006 5100h＋4h×n	rADC1_DATA[n] n＝0～15	仮想チャネル[n]ADC1 変換データ
4006 5180h＋4h×n	rADC1_DATALOCK[n] n＝0～15	ADC1 データロック[n]レジスタ

7.3.2　レジスタマップ ADC2

注　意

パッケージ FBGA400 でのみ利用可能です。

表 7.2　ADC2 のレジスタマップ

アドレス	レジスタシンボル	レジスタ名
4006 5140h＋4h×n	rADC2_DATA[n] n＝0～15	仮想チャネル[n]ADC2 変換データ
4006 51C0h＋4h×n	rADC2_DATALOCK[n] n＝0～15	ADC2 データロック[n]レジスタ

7.4　レジスタの説明

7.4.1　レジスタの説明 ADC1

7.4.1.1　rADC_INTSTATUS0 ― マスキング前割り込みステータス

アドレス　4006 5000h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_INTSTATUS0_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.3　rADC_INTSTATUS0 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_INTSTATUS0_VC	仮想チャネル ADC_VC[n]（n＝0～15）のマスキング前の割り込みステータス bADC_INTSTATUS0_VC[n]ビット＝1 かつ rADC_INTMASK レジスタの対応するマスクビットがセットされている場合に ADC_Int 信号がアサートされます。 ビット[n]：bADC_INTSTATUS0_VC[n]ビットは、iADC_EOC_VC[n]の立ち上がりで発生するトリガです。 0：割り込みなし 1：割り込みセット済み **備考）** ● ADC_VC[n]が連続モード（bADC_Continuous ビット＝1）に設定されている場合、本フラグビットがセットされていても、選択した iADC_EOC_VC[n]イベントが発生すると必ず追加のトリガパルスが生成されます。 ● 各 bADC_INTSTATUS0_VC[n]ビットがセットされ、かつ選択した追加の iADC_EOC_VC[n]トリガが生成されると、bADC_INTOVFSTATUS0_VC[n]ビットで ADC_VC[n]割り込みオーバーフローイベントが発生します。 ● 本レジスタを読み出しても、アクティブな割り込みはクリアされません。	R

7.4.1.2　rADC_INTSTATUS1 ― マスキング後割り込みステータス

アドレス　4006 5004h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_INTSTATUS1_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.4　rADC_INTSTATUS1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_INTSTATUS1_VC	仮想チャネル ADC_VC[n]（n＝0～15）のマスキング後の割り込みステータス 本ビットフィールドは、bADC_INTMASK ビットと bADC_INTSTATUS0_VC ビットの論理積を示します。 ビット[n]：bADC_INTSTATUS1_VC[n]ビットは、iADC_EOC_VC[n]の立ち上がりで発生するトリガです。 0：割り込みなし 1：割り込みセット済み	R

7.4.1.3　rADC_INTCLR ― 割り込みクリア

アドレス　4006 5008h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_INTCLR_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.5　rADC_INTCLR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_INTCLR_VC	仮想チャネル ADC_VC[n]（n＝0～15）の割り込みステータスのクリア rADC_INTSTATUS0 レジスタと rADC_INTSTATUS1 レジスタの各割り込みビットをクリアします。 ビット[n]：bADC_INTCLR_VC[n]ビットは、bADC_INTSTATUSx_VC[n]ビット（x＝0、1）をクリアします。 0：何もしません 1：bADC_INTSTATUS0_VC[n]ビットおよび bADC_INTSTATUS1_VC[n]ビットの各割り込みをクリア **備考）** ● ハードウェアがレジスタの bADC_INTSTATUS0_VC[n]ビットをセットしようとしているとき、同じクロックサイクルでソフトウェアが本ビット n をセットしようとすると、ハードウェアの方が優先されて、bADC_INTSTATUS0_VC[n]ビットがセットされます。 この場合、bADC_INTSTATUS0_VC[n]ビットが以前にセットされたか否かにかかわらず、レジスタの各オーバーフロービット（bADC_INTOVFSTATUS0_VC[n]）は影響を受けません。 ● 読むと常に 0 が読み出されます。	W

7.4.1.4　rADC_INTMASK ― 割り込みマスク

アドレス　4006 500Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_INTMASK_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.6　rADC_INTMASK レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_INTMASK_VC	仮想チャネル ADC_VC[n]（n＝0～15）の割り込みステータスのマスク ビット[n]：bADC_INTMASK_VC[n]ビットは、bADC_INTSTATUS0_VC[n]ビットをマスクします。 1：割り込み許可 0：割り込み禁止	R/W

7.4.1.5　rADC_INTOVFSTATUS0 ― マスキング前割り込みオーバーフロー

アドレス　4006 5010h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_INTOVFSTATUS0_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.7　rADC_INTOVFSTATUS0 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_INTOVFSTATUS0_VC	仮想チャネル ADC_VC[n]（n＝0～15）のマスキング前の割り込みオーバーフロー 各 bADC_INTSTATUS0_VC[n]ビットがセットされ、かつ選択した追加の iADC_EOC_VC[n]トリガが生成されると、オーバーフロー状態が発生します。 rADC_INTOVFMASK レジスタの対応するマスクビットの bADC_INTOVFMASK_VC[n]が 1 である場合に、ADC_Int 信号がアサートされます。 ビット[n]：bADC_INTOVFSTATUS0_VC[n]ビットは、bADC_INTSTATUS0_VC[n]ビットの割り込みオーバーフロービットです。 0：割り込みオーバーフローイベントの検出なし 1：可能性のある割り込みオーバーフローを検出 **備考）** 本オーバーフロービットは、連続モードビット（bADC_Continuous）の状態を考慮しません。オーバーフロー状態は、このモード選択とは無関係に発生します。	R

7.4.1.6　rADC_INTOVFSTATUS1 ― マスキング後割り込みオーバーフロー

アドレス　4006 5014h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_INTOVFSTATUS1_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.8　rADC_INTOVFSTATUS1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_INTOVFSTATUS1_VC	仮想チャネル ADC_VC[n]（n＝0～15）のマスキング後の割り込みオーバーフロー 本ビットフィールドは、bADC_INTOVFMASK ビットと bADC_INTOVFSTATUS1_VC ビットの論理積を示します。 ビット[n]：bADC_INTOVFSTATUS1_VC[n]ビットは、bADC_INTSTATUS1_VC[n]ビットの割り込みオーバーフロービットです。 0：割り込みオーバーフローイベントの検出なし 1：可能性のある割り込みオーバーフローが検出されて、対応する bADC_INTOVFMASK＝1 **備考）** 本オーバーフロービットは、連続モードビット（bADC_Continuous）の状態を考慮しません。オーバーフロー状態は、このモード選択とは無関係に発生します。	R

7.4.1.7　rADC_INTCLROVF ― 割り込みオーバーフロークリア

アドレス　4006 5018h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_INTCLROVF_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.9　rADC_INTCLROVF レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_INTCLROVF_VC	仮想チャネル ADC_VC[n]（n＝0～15）の割り込みオーバーフローのクリア rADC_INTOVFSTATUS0 レジスタと rADC_INTOVFSTATUS1 レジスタの各割り込みビットをクリアします。 ビット[n]：bADC_INTCLROVF_VC[n]ビットは、bADC_INTOVFSTATUSx_VC[n]ビット（x＝0、1）をクリアします。 0：何もしません 1：bADC_INTOVFSTATUSx_VC[n]ビット（x＝0、1）の各割り込みをクリア **備考）** ● ハードウェアがレジスタのオーバーフロービット（bADC_INTOVFSTATUS0_VC[n]）をセットしようとしているとき、同じクロックサイクルでソフトウェアが本ビット n をセットしようとすると、ハードウェアの方が優先されて、bADC_INTOVFSTATUS0_VC[n]ビットがセットされます。 ● 読むと常に 0 が読み出されます。	W

7.4.1.8　rADC_INTOVFMASK ― 割り込みオーバーフローマスク

アドレス　4006 501Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_INTOVFMASK_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.10　rADC_INTOVFMASK レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_INTOVFMASK_VC	仮想チャネル ADC_VC[n]（n＝0～15）の割り込みオーバーフローのマスク ビット[n]：bADC_INTOVFMASK_VC[n]ビットは、bADC_INTOVFSTATUS0_VC[n]ビットをマスクします。 1：割り込み許可 0：割り込み禁止	R/W

7.4.1.9　rADC_PENDING ― 動作開始保留中

アドレス　4006 5020h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bADC_DMA1_RUNNING	bADC_DMA0_RUNNING	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_PENDING_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.11　rADC_PENDING レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31	bADC_DMA1_RUNNING	ADC DMA チャネル 1 実行中 ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択されて、下記のすべての条件が満たされた場合： (1) イベント検出「データロックレジスタへのデータコピー」 (2) bADC_Mode ビット（rADC_VC[n]レジスタ）＝2'b11 の設定 (3) bADC_DMA_Request[1:0]ビット（rADC_VC[n]レジスタ）が 2'b11 に設定 ADC DMA チャネル 1 要求が起動して、DMA 転送終了の検出まで実行されます。 1：DMA 要求実行中 0：同一チャネルでの DMA 要求なし **備考）** ● 本チャネルで DMA 転送終了が検出されると、本ビットは自動的にクリアされます。 ● 本ビットが同じサイクルでセット要求とクリア要求の両方を受け取った結果、競合が生じた場合、どちらの要求元とも無関係に、本ビット（bADC_DMA1_RUNNING）はセットされ、クリア要求は無視されます。この場合、本ビットが以前にセットされたか否かにかかわらず、rADC_PENDINGOVF レジスタのオーバーフロービット（bADC_DMA1_RUNNINGOVF）は影響を受けません。	R
b30	bADC_DMA0_RUNNING	ADC DMA チャネル 0 実行中 bADC_DMA1_RUNNING と同じ	R
b29～b16	予約ビット	使用しない	R

表 7.11　rADC_PENDING レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b15～b0	bADC_PENDING_VC	仮想チャネル ADC_VC[n]（n＝0～15）の動作開始保留中 ビット[n]：bADC_PENDING_VC[n]ビットは、ADC_VC[n]でのイベント受信時にセットされます。 1：イベント受信済み。以下の場合にセットされます。 - rADC_VC[n]の bADC_TrigSel ビットで選択したトリガの立ち上がり検出 - CPU が bADC_SETFORCE_VC ビットのセットで bADC_FORCE_VC[n] ビットを 1 に変更 - ADC_VC[n]に対する ADC 要求（変換、サンプル＆ホールド、その他の動作）が保留中 0：イベント受信なし - 保留中の動作なし **備考）** ● 各 ADC_VC[n]の動作が開始されると、本ビットは自動的にクリアされます。 ● 本ビットが同じサイクルでセット要求とクリア要求の両方を受け取った結果、競合が生じた場合、どちらの要求元とも無関係に、本ビット（bADC_PENDING_VC[n]）はセットされ、クリア要求は無視されます。この場合、本ビットが以前にセットされたか否かにかかわらず、rADC_PENDINGOVF レジスタのオーバーフロービット（bADC_PENDINGOVF_VC[n]）は影響を受けません。 ● 要求された変換またはサンプル＆ホールド動作は、bADC_Mode ビットの状態に依存します。 ● rADC_CONFIG レジスタへの CPU 書き込みによって以下の動作を開始すると、0 にクリアされます。 – bADC_POWER_DOWN（低消費電力モード有効または無効） – bADC_SAMPLE_HOLD_ENABLE（各チャネルのサンプル＆ホールド機能有効または無効）	R

7.4.1.10　rADC_PENDINGOVF ― 動作開始保留中オーバーフロー

アドレス　4006 5024h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bADC_DMA1_RUNNINGOVF	bADC_DMA0_RUNNINGOVF	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_PENDINGOVF_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.12　rADC_PENDINGOVF レジスタの内容

ビット位置	ビット名	機能	R/W
b31	bADC_DMA1_RUNNINGOVF	ADC DMA チャネル 1 実行中のオーバーフロー 以前の要求がすでに実行されている状態で、新たに bADC_DMA_Request[1:0]ビット＝2'b11 が発生したことを示します。 1：ADC_DMA1 で ADC DMA オーバーフロー発生 0：ADC オーバーフローなし **備考）** ● オーバーフロー状態が、ADC DMA 転送の処理を中断させることはありません。それは単に、DMA 要求が失敗したことを示しています。 すなわち、最初の DMA 要求ブロックは、DMA 終了まで正常に処理されますが、トリガされた 2 番目の DMA 転送ブロックは処理されません。	R
b30	bADC_DMA0_RUNNINGOVF	ADC DMA チャネル 0 実行中のオーバーフロー 以前の要求がすでに実行されている状態で、新たに bADC_DMA_Request[1:0]ビット＝2'b10 が発生したことを示します。 1：ADC_DMA0 で ADC DMA オーバーフロー発生 0：ADC オーバーフローなし **備考）** ● オーバーフロー状態が、ADC DMA 転送の処理を中断させることはありません。それは単に、DMA 要求が失敗したことを示しています。 すなわち、最初の DMA 要求ブロックは、DMA 終了まで正常に処理されますが、トリガされた 2 番目の DMA 転送ブロックは処理されません。	R
b29～b16	予約ビット	使用しない	R
b15～b0	bADC_PENDINGOVF_VC	仮想チャネル ADC_VC[n]（n＝0～15）の動作開始保留中のオーバーフロー 既存イベントがすでに保留中の状態で、新たに ADC_VC[n]イベントが発生したことを示します。 ビット[n]：bADC_PENDINGOVF_VC[n]ビットは、ADC_VC[n]イベントのオーバーフロービットです。 1：イベントオーバーフロー発生 0：イベントオーバーフローなし **備考）** オーバーフロー状態が、ADC_VC[n]イベントの処理を中断させることはありません。それは単に、トリガが失敗したことを示しています。	R

7.4.1.11　rADC_PENDINGCLROVF ― 動作開始オーバーフロークリア

アドレス　4006 5028h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bADC_DMA1_RUNNINGCLROVF	bADC_DMA0_RUNNINGCLROVF	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_PENDINGCLROVF_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.13　rADC_PENDINGCLROVF レジスタの内容

ビット位置	ビット名	機能	R/W
b31	bADC_DMA1_RUNNINGCLROVF	ADC DMA チャネル 1 実行中のオーバーフローのクリア 1 を書き込むと、bADC_DMA1_RUNNINGOVF ビットがクリアされます。 1：bADC_DMA1_RUNNINGOVF ビットをクリア 0：何もしません **備考）** ● ハードウェアが rADC_PENDINGOVF レジスタのオーバーフロービット（bADC_DMA1_RUNNINGOVF）をセットしようとしているとき、同じクロックサイクルでソフトウェアが本ビットをセットしようとすると、ハードウェアの方が優先されて、bADC_DMA1_RUNNINGOVF ビットがセットされます。 ● 読むと常に 0 が読み出されます。	W
b30	bADC_DMA0_RUNNINGCLROVF	ADC DMA チャネル 0 実行中のオーバーフローのクリア bADC_DMA1_RUNNINGCLROVF と同じ	W
b29～b16	予約ビット	使用しない	R
b15～b0	bADC_PENDINGCLROVF_VC	仮想チャネル ADC_VC[n]（n＝0～15）の動作開始オーバーフローのクリア 1 を書き込むと、各 ADC_VC[n]の保留中オーバーフローフラグがクリアされます。 ビット[n]：bADC_PENDINGCLROVF_VC[n]ビットは、ADC_VC[n]イベントのオーバーフローをクリアします。 1：rADC_PENDINGOVF レジスタの各 bADC_PENDINGOVF_VC[n]ビットをクリア 0：何もしません **備考）** ● ハードウェアが rADC_PENDINGOVF レジスタのオーバーフロービット（bADC_PENDINGOVF_VC[n]）をセットしようとしているとき、同じクロックサイクルでソフトウェアが本ビット n をセットしようとすると、ハードウェアの方が優先されて、bADC_PENDINGOVF_VC[n]ビットがセットされます。 ● 読むと常に 0 が読み出されます。	W

7.4.1.12　rADC_CONTROL — ADC 制御

アドレス　4006 502Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	bADC_BUSY	bADC_VC_BUSY	bADC_VC_RUN				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.14　rADC_CONTROL レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b7	予約ビット	使用しない	R
b6	bADC_BUSY	ADC コンフィグレーションの変更実行中 保留中の全 ADC 動作が停止します。 ● rADC_CONFIG レジスタへの CPU 書き込みによって以下の動作を開始すると、1 にセットされます。 bADC_POWER_DOWN - 低消費電力モード有効または無効 bADC_SAMPLE_HOLD_ENABLE - 各チャネルのサンプル＆ホールド機能有効または無効 ● 以下の動作が終了すると、0 にクリアされます。 bADC_POWER_DOWN - 低消費電力モード有効または無効 bADC_SAMPLE_HOLD_ENABLE - 各チャネルのサンプル＆ホールド機能有効または無効 1：ADC コンフィグレーションの変更実行中 0：ADC コンフィグレーションの変更終了 **備考）** ● ステートマシンによるハードウェアリセット（コンフィグレーションの変更終了）までは、CPU が本ビットをポーリングしている必要があります。 ● CPU 読み出し値が 0 になると、仮想チャネルのコンフィグレーションを開始できるようになります。	R
b5	bADC_VC_BUSY	ADC_VC ビジー ADC_VC が実行中の場合にセットされ、アイドル状態（イベント待ち）の場合にクリアされます。 0：ADC_VC は次の仮想チャネルの実行に利用可能 1：ADC_VC はビジー状態であり、別の仮想チャネルの実行は不可	R
b4～b0	bADC_VC_RUN	ADC_VC チャネルステータス bADC_VC_BUSY ビット＝0 の場合、最後に実行された ADC_VC の値を保持します。 bADC_VC_BUSY ビット＝1 の場合、現在処理中の ADC_VC を反映します。 5'h00：ADC_VC0 が現在処理中であるか、または最後に実行された ADC_VC … … 5'h0F：ADC_VC15 が現在処理中であるか、または最後に実行された ADC_VC 5'h1x：予約	R

7.4.1.13　rADC_FORCE ― ADC 要求

アドレス　4006 5030h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_FORCE_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.15　rADC_FORCE レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_FORCE_VC	仮想チャネル ADC_VC[n]（n＝0～15）の動作強制開始 rADC_SETFORCE レジスタで 1 を書き込むと、rADC_PENDING レジスタの各フラグビット（bADC_PENDING_VC[n]）が強制的に 1 になります。本ビットは、ソフトウェア起動による動作を開始するために利用できます。 各仮想チャネル ADC_VC[n]の動作強制開始は、以下の割り当てビットに直接関係しています。 ビット[n]：bADC_FORCE_VC[n]ビットは、動作開始ビット（bADC_PENDING_VC[n]）をセットします。 1：ADC_VC[n]が優先された時点で、ADC 要求を強制的に起動 0：何もしません **備考）** ● ハードウェアが rADC_PENDING レジスタの bADC_PENDING_VC[n]ビットをクリアしようとしているとき、同じクロックサイクルでソフトウェアが本ビットをセットしようとすると、ソフトウェアの方が優先されて、bADC_PENDING_VC[n]ビットがセットされます。 この場合、bADC_PENDING_VC[n]ビットが以前にセットされたか否かにかかわらず、rADC_PENDINGOVF レジスタのオーバーフロービットは影響を受けません。 ● 各仮想チャネル ADC_VC[n]（n＝0～15）が単一モードである（bADC_Continuous ビットを 1'b0 にクリアした）場合 – ADC 動作（変換、サンプル＆ホールド、その他）は 1 回限り実行されます。選択したチャネルでの動作が終了すると、bADC_FORCE_VC[n]ビットは自動的に 0 にクリアされます。 ● 各仮想チャネル ADC_VC[n]（n＝0～15）が連続モードである（bADC_Continuous ビットを 1'b1 にセットした）場合 – ADC 動作は選択した複数チャネルに対して順番に連続して実行されます。これは、rADC_CLRFORCE レジスタへの CPU 書き込みによって、bADC_FORCE_VC[n]ビットがクリアされるまで続きます。 ● rADC_CONFIG レジスタへの CPU 書き込みによって以下の動作を開始すると、0 にクリアされます。 – bADC_POWER_DOWN（低消費電力モード有効または無効） – bADC_SAMPLE_HOLD_ENABLE（各チャネルのサンプル＆ホールド機能有効または無効）	R

7.4.1.14　rADC_SETFORCE ― ADC 要求セット

アドレス　4006 5034h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_SETFORCE_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.16　rADC_SETFORCE レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_SETFORCE_VC	仮想チャネル ADC_VC[n]（n＝0～15）の動作強制開始のセット rADC_FORCE レジスタの各動作強制開始ビットをセットします。 ビット[n]：bADC_SETFORCE_VC[n]ビットは、bADC_FORCE_VC[n]ビットをセットします。 1：rADC_FORCE レジスタの各 bADC_FORCE_VC[n]ビットをセット 0：何もしません。0 の書き込みは無視されます。 **備考）** 読むと常に 0 が読み出されます。	W

7.4.1.15　rADC_CLRFORCE ― ADC 要求クリア

アドレス　4006 5038h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_CLRFORCE_VC															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.17　rADC_CLRFORCE レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	使用しない	R
b15～b0	bADC_CLRFORCE_VC	仮想チャネル ADC_VC[n]（n＝0～15）の動作強制開始のクリア rADC_FORCE レジスタの各動作強制開始ビットをクリアします。 ビット[n]：bADC_CLRFORCE_VC[n]ビットは、bADC_FORCE_VC[n]ビットをクリアします。 1：rADC_FORCE レジスタの各 bADC_FORCE_VC[n]ビットをクリア 0：何もしません。0 の書き込みは無視されます。 **備考**）　読むと常に 0 が読み出されます。	W

7.4.1.16　rADC_PRIORITY ― ADC 優先モード

アドレス　4006 503Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	bADC_RR_Pointer					bADC_Priority				
リセット後の値	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

表 7.18　rADC_PRIORITY レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b10	予約ビット	使用しない	R
b9～b5	bADC_RR_Pointer	ADC_VC[n]ラウンドロビンポインタ（n＝0～15） 変換順序を決定するため、ラウンドロビン方式によって使用される最後に変換されたラウンドロビン ADC_VC[n]の値を保持します。 5’h00： – ADC_VC0 が最後に変換されたラウンドロビン ADC_VC – ADC_VC1 が最優先のラウンドロビン順位となる 5’h01： – ADC_VC1 が最後に変換されたラウンドロビン ADC_VC – ADC_VC2 が最優先のラウンドロビン順位となる … 5’h0E： – ADC_VC14 が最後に変換されたラウンドロビン ADC_VC – ADC_VC14 が最優先のラウンドロビン順位となる 5’h0F： – ADC_VC15 が最後に変換されたラウンドロビン ADC_VC – ADC_VC0 が最優先のラウンドロビン順位となる 5’h10： – どの ADC_VC も変換されていないことを示すために値をリセット。ADC_VC0 が最優先のラウンドロビン順位となる – bADC_Priority ビットに書き込みがなされたとき、この値に設定されます。変換が現在進行中であれば、それが終了した後、新しい優先順位になります。 – デバイスがリセットされたとき、または rADC_CONFIG レジスタへの CPU 書き込みによって以下の動作を開始したとき、この値に設定されます。 ・bADC_POWER_DOWN（低消費電力モード有効または無効） ・bADC_SAMPLE_HOLD_ENABLE（各チャネルのサンプル＆ホールド機能有効または無効） ・変換が現在進行中であれば、それが終了した後、新しい優先順位になります。 5’h11～1F： – 無効な値	R

表 7.18　rADC_PRIORITY レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b4～b0	bADC_Priority	ADC_VC[n]優先順位（n＝0～15） ADC_VC[n]に対して優先モードのカットオフポイントとラウンドロビン調停を決定します。 5’h00：ADC_VC の優先順位は全チャネルをラウンドロビン方式で処理 5’h01：VC0 を高優先順位とし、残りのチャネルはラウンドロビン方式 5’h02：VC0～1 を高優先順位とし、VC2～15 はラウンドロビン方式 5’h03：VC0～2 を高優先順位とし、VC3～15 はラウンドロビン方式 5’h04：VC0～3 を高優先順位とし、VC4～15 はラウンドロビン方式 5’h05：VC0～4 を高優先順位とし、VC5～15 はラウンドロビン方式 5’h06：VC0～5 を高優先順位とし、VC6～15 はラウンドロビン方式 5’h07：VC0～6 を高優先順位とし、VC7～15 はラウンドロビン方式 5’h08：VC0～7 を高優先順位とし、VC8～15 はラウンドロビン方式 5’h09：VC0～8 を高優先順位とし、VC9～15 はラウンドロビン方式 5’h0A：VC0～9 を高優先順位とし、VC10～15 はラウンドロビン方式 5’h0B：VC0～10 を高優先順位とし、VC11～15 はラウンドロビン方式 5’h0C：VC0～11 を高優先順位とし、VC12～15 はラウンドロビン方式 5’h0D：VC0～12 を高優先順位とし、VC13～15 はラウンドロビン方式 5’h0E：VC0～13 を高優先順位とし、VC14～15 はラウンドロビン方式 5’h0F：VC0～14 を高優先順位とし、VC15 はラウンドロビン方式 5’h10：すべての VC[n]（n＝0～15）が高優先順位であり、VC[n]の番号で調停される 5’h11～1F：その他は無効な選択	R/W

7.4.1.17　rADC_CONFIG ― ADC コンフィグレーション

アドレス　4006 5040h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	bADC_DMA	bADC_POWER_DOWN	bADC_SAMPLE_HOLD_ENABLE		
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

表 7.19　rADC_CONFIG レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b5	予約ビット	使用しない	R
b4	bADC_DMA	DMA チャネル許可 1：DMA チャネル 0 および 1 許可 0：DMA チャネル 0 および 1 禁止 現在のアクセスに対する DMA ACK 応答受信後は、現在の全 DMA 要求が強制的にアイドル状態となります。	R/W
b3	bADC_POWER_DOWN	低消費電力モード有効または無効 低消費電力モードでは、ADC は使用できません。 本ビットは、仮想チャネル ADC_VC[n]（n＝0～15）の設定を行う前にクリアする必要があります。 1：アナログ A/D コンバータ ADC1 および ADC2 を低消費電力モードに設定 0：アナログ A/D コンバータ ADC1 および ADC2 を動作モードに設定 CPU 書き込みが検出されたとき、以下の設定がなされます。 (1) ADC コンフィグレーションの変更が終了するまで、bADC_BUSY ビットをセット (2) すべての rADC_VC[n]レジスタ（n＝0～15）の bADC_TrigEnable ビットを 0 にクリア（トリガ無効） (3) rADC_PENDING レジスタの bADC_PENDING_VC[n]ビット（n＝0～15）（仮想チャネル ADC_VC[n]の動作保留中）を 0 にクリア (4) rADC_FORCE レジスタの bADC_FORCE_VC[n]ビット（n＝0～15）（仮想チャネル ADC_VC[n]の動作強制開始）を 0 にクリア (5) rADC_PRIORITY レジスタの bADC_RR_Pointer ビットを 5'h10 に設定（ADC_VC0 が最優先のラウンドロビン順位となります。変換が現在進行中であれば、それが終了した後、新しい優先順位になります） (6) 現在の動作の終了時に、仮想ステートマシンに対してアイドル状態を強制	R/W

表 7.19　rADC_CONFIG レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b2～b0	bADC_SAMPLE_HOLD_ENABLE	各チャネルのサンプル＆ホールド機能有効または無効 仮想マシン ADC_VC[n]（n＝0～15）によって、ADC1 コンバータと ADC2 コンバータが結合され、両 ADC に対して同じ機能（サンプル＆ホールド機能有効または無効）が設定されます。 サンプル＆ホールド機能は、以下の割り当てビットに直接関係しています。 ビット[0]の設定 - ADC1 および ADC2 の物理チャネル 6 ビット[1]の設定 - ADC1 および ADC2 の物理チャネル 7 ビット[2]の設定 - ADC1 および ADC2 の物理チャネル 8 0：アナログ入力はマルチプレクサ回路に直結し、サンプル＆ホールド回路は使用しない（スルーモード） 1：サンプル＆ホールド機能は利用可能 CPU 書き込みが検出されたとき、以下の設定がなされます。 (1) ADC コンフィグレーションの変更が終了するまで、bADC_BUSY ビットをセット (2) すべての rADC_VC[n]レジスタ（n＝0～15）の bADC_TrigEnable ビットを 0 にクリア（トリガ無効） (3) rADC_PENDING レジスタの bADC_PENDING_VC[n]ビット（n＝0～15）（ADC_VC[n]の動作保留中）を 0 にクリア (4) rADC_FORCE レジスタの bADC_FORCE_VC[n]ビット（n＝0～15）（ADC_VC[n]の動作強制開始）を 0 にクリア (5) 現在の動作の終了時に、仮想ステートマシンに対してアイドル状態を強制	R/W

7.4.1.18　rADC_ACQS ― ADC サンプル＆ホールド制御

アドレス　4006 50ACh

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	bADC_TSHOH			bADC_TSHSET		bADC_TSHSAMP				
リセット後の値	0	0	0	0	0	0	1	1	0	1	0	0	0	1	1	0

表 7.20　rADC_ACQS レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b10	予約ビット	読むと“0”が読み出されます。	R
b9、b7	bADC_TSHOH	ADC サンプル＆ホールドの SHOUT～SHCNT ホールド時間：t_{SHOH} ADC1 および ADC2 用サンプル＆ホールド回路において、SHCNT の立ち下がりから SHOUT の立ち上がりまでの間に 300ns のホールド時間を確保するための ADC_CLK クロック数を指定します。 t_{SHOH}：最小値は 300ns 3’b000：無効な値 3’b001：無効な値 3’b010：2 サイクル（4MHz ≦ f_{ADC_CLK} ≦ 6.66MHz のときに設定） 3’b011：3 サイクル（6.66MHz ＜ f_{ADC_CLK} ≦ 10MHz のときに設定） … … 3’b110：6 サイクル（16.66MHz ＜ f_{ADC_CLK} ≦ 20MHz のときに設定） 3’b111：無効な値	R/W
b6、b5	bADC_TSHSET	ADC サンプル＆ホールドの SHOUT～CONV セットアップ時間：t_{SHSET} からの遅延 ADC1 および ADC2 用サンプル＆ホールド回路において、SHOUT から CONV までの間に 100ns のセットアップ時間を確保するための ADC_CLK クロック数を指定します。 t_{SHSET}：最小値は 100ns 2’b00：無効な値 2’b01：1 サイクル（4MHz ≦ f_{ADC_CLK} ＜ 10MHz のときに設定） 2’b10：2 サイクル（10MHz ≦ f_{ADC_CLK} ≦ 20MHz のときに設定） 2’b11：無効な値	R/W
b4～b0	bADC_TSHSAMP	ADC サンプル＆ホールドのサンプリング時間：t_{SHSAMP} ADC1 および ADC2 用サンプル＆ホールド回路において、ウィンドウサンプリング時間を確保するための ADC_CLK クロック数を指定します。 t_{SHSAMP}：最小値は 300ns 5’h0：無効な値 5’h1：無効な値 5’h2：2 サイクル 5’h3：3 サイクル … … 5’h7：7 サイクル 5’h8：8 サイクル … … 5’h1f：31 サイクル	R/W

7.4.1.19　rADC_MASKLOCK[n] ― マスクデータロック[n]（n＝0～3）

アドレス　4006 50B0h＋4h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC_MASKLOCK															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.21　rADC_MASKLOCK[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b16	予約ビット	読むと“0”が読み出されます。	R
b15～b0	bADC_MASKLOCK	マスクデータロック 下記の各レジスタからのコピーを許可または禁止します。 ● rADC1_DATA[n]レジスタから rADC1_DATALOCK[n]レジスタ（n＝0～15） ● rADC2_DATA[n]レジスタから rADC2_DATALOCK[n]レジスタ（n＝0～15） ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択されて、下記のすべての条件が満たされた場合： ● イベント検出「データロックレジスタへのデータコピー」 ● bADC_Mode ビット（rADC_VC[n]レジスタ）＝2’b11 の設定 下記の動作が実行されます： ● bADC1_ChannelSel は、各 rADC1_DATA[n]レジスタから rADC1_DATALOCK[n]レジスタ（n＝0～15）へのデータコピーを許可または禁止するために使用する rADC_MASKLOCK0～3 レジスタを選択します。 ● bADC2_ChannelSel は、各 rADC2_DATA[n]レジスタから rADC2_DATALOCK[n]レジスタ（n＝0～15）へのデータコピーを許可または禁止するために使用する rADC_MASKLOCK0～3 レジスタを選択します。 各 bADC_MASKLOCK[n]ビットは下記の割り当てビットに直接関連付けられます。 ● ビット[n]：bADC_MASKLOCK[n]ビットは、下記レジスタのマスクデータロックです。 - rADC1_DATA[n]レジスタ（bADC1_ChannelSel で選択した場合） - rADC2_DATA[n]レジスタ（bADC2_ChannelSel で選択した場合） 0：何もしません。コピーを禁止 1：bADC1_ChannelSel、rADC1_DATA[n]から rADC1_DATALOCK[n]へのコピー許可 bADC2_ChannelSel、rADC2_DATA[n]から rADC2_DATALOCK[n]へのコピー許可	R/W

7.4.1.20　rADC_VC[n] ― 仮想チャネル[n]用 ADC コントロールレジスタ（n＝0～15）

アドレス　4006 50C0h＋4h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	bADC_DMA_Request		bADC2_Enable
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bADC1_Enable	bADC_Continuous	bADC_TrigEnable	bADC_TrigSel					bADC_Mode		bADC2_ChannelSel			bADC1_ChannelSel		
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.22　rADC_VC[n]レジスタの内容（1/5）

ビット位置	ビット名	機能	R/W
b31～b19	予約ビット	読むと“0”が読み出されます。	R
b18、b17	bADC_DMA_Request	仮想チャネル ADC_VC[n]（n＝0～15）の ADC DMA 要求 ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択されて、下記のすべての条件が満たされた場合： ● イベント検出「データロックレジスタへのデータコピー」 ● bADC_Mode ビット（rADC_VC[n]レジスタ）＝2’b11 の設定 ● bADC_DMA_Request[1]ビットが 1 に設定 ● rADC_CONFIG レジスタの bADC_DMA ビットが有効 ADC DMA 要求が起動して、DMA 転送終了の検出まで実行されます。選択される ADC DMA チャネルは、bADC_DMA_Request[0]ビットの状態に依存します。 2’b00：ADC DMA 禁止 - DMA 接続なし 2’b10：ADC DMA0 許可 - rADC_PENDING レジスタの bADC_DMA0_RUNNING ビットをセット 2’b11：ADC DMA1 許可 - rADC_PENDING レジスタの bADC_DMA1_RUNNING ビットをセット	R/W
b16	bADC2_Enable	仮想チャネル ADC_VC[n]（n＝0～15）の ADC2 有効 セットした場合、ADC2 の変換、サンプル＆ホールド、その他の動作が許可されます。 各仮想チャネル ADC_VC[n]（n＝0～15）に対して 0：ADC2 無効 – 選択した ADC_VC[n]イベントが設定された場合（トリガの立ち上がり検出、または CPU が bADC_SETFORCE_VC ビットのセットにより bADC_FORCE_VC[n]ビットを 1 に変更）、かつ、ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択されます。bADC_Mode、bADC2_ChannelSel で選択および制御される機能（変換、サンプル＆ホールド、その他の動作）は、ADC2 では実行されません。 1：ADC2 有効 – 選択した ADC_VC[n]イベントが設定された場合（トリガの立ち上がり検出、または CPU が bADC_SETFORCE_VC ビットのセットにより bADC_FORCE_VC[n]ビットを 1 に変更）、かつ、ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択されます。bADC_Mode、bADC2_ChannelSel で選択および制御される機能（変換、サンプル＆ホールド、その他の動作）は、ADC2 で実行される必要があります。 **備考）** 割り込みのみのモードではない場合、かつ ADC1 と ADC2 が両方とも無効な場合、それでも専用のコマンド終了（EoC）信号は生成されます。	R/W

表 7.22　rADC_VC[n]レジスタの内容（2/5）

ビット位置	ビット名	機能	R/W
		注意） ● ADC2 が実装されていない場合（たとえば、RZ/N1S）、本ビットは仮想チャネルに影響を与えません。ハードウェアによって“0”に設定されます。 ADC2 が接続されていない場合（RZ/N1D-324）、物理チャネルは“0”にゲートされるため、本チャネルでは変換もサンプル＆ホールドも不可能です。	
b15	bADC1_Enable	仮想チャネル ADC_VC[n]（n＝0～15）の ADC1 有効 bADC2_Enable と同じ	R/W
b14	bADC_Continuous	仮想チャネル ADC_VC[n]（n＝0～15）の ADC 連続モード 各仮想チャネル ADC_VC[n]（n＝0～15）に対して 0：単一モード 1：連続モード **備考）** **単一モードの場合：** ● ADC 動作（変換、サンプル＆ホールド、その他）は 1 回限り実行されます。 ● bADC_TrigEnable ビットは、選択したチャネルでの動作が終了すると、自動的に 0 にクリアされます。 ● bADC_FORCE_VC[n]ビットは、選択したチャネルでの動作が終了すると、自動的に 0 にクリアされます。 **連続モードの場合：** ● ADC 動作は選択した複数チャネルに対して順番に連続して実行されます。これは、bADC_TrigEnable ビットおよび bADC_FORCE_VC[n]ビットが、ソフトウェア／ハードウェアリセットまたはファームウェア ADC リセットによってクリアされるまで続きます。 ● 選択したチャネルでの動作が終了したとき、bADC_TrigEnable ビットおよび bADC_FORCE_VC[n]ビットに変化はありません。 **すべての場合において、以下のすべての条件が満たされたとき、ADC_VC[n]の単一モード、連続モード、またはその他の動作が実行されます。** ● 選択した専用の ADC_VC[n]イベントが設定されている（トリガの立ち上がり検出、または CPU が bADC_SETFORCE_VC ビットのセットにより bADC_FORCE_VC[n]ビットを 1 に変更） ● ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択されている ● ADC2 および ADC1 のそれぞれ bADC2_Enable ビットおよび bADC1_Enable ビットが有効になっている	R/W
b13	bADC_TrigEnable	仮想チャネル ADC_VC[n]（n＝0～15）のトリガ有効 各仮想チャネル ADC_VC[n]（n＝0～15）に対して 0：トリガ無効 – 仮想チャネルはアイドル状態であり、bADC_TrigSel ビットで選択したトリガで起動できません。 1：トリガ有効 – 本ビットがセットされた場合、ADC_VC[n]の実行が ADC ステートマシンで選択されていると、bADC_TrigSel ビットで設定したトリガ条件によって動作（変換、サンプル＆ホールド、その他）を開始します。 **備考）** ● 単一モード（bADC_Continuous ビット＝0）の場合、選択したチャネルで動作が終了すると、本ビットは自動的に 0 にクリアされます。 ● rADC_CONFIG レジスタへの CPU 書き込みによって以下の動作を開始すると、0 にクリアされます。 – bADC_POWER_DOWN（低消費電力モード有効または無効） – bADC_SAMPLE_HOLD_ENABLE（各チャネルのサンプル＆ホールド機能有効または無効）	R/W

表 7.22　rADC_VC[n]レジスタの内容（3/5）

ビット位置	ビット名	機能	R/W
b12～b8	bADC_TrigSel	仮想チャネル ADC_VC[n]（n＝0～15）のトリガ選択 ADC ステートマシンによって ADC_VC[n]が優先された時点で開始する動作（変換、サンプル＆ホールド、その他）を開始するためのトリガを設定します。 ADC_VC[n]イベントの開始には、立ち上がりエッジのトリガのみが使用されます。 仮想チャネル ADC_VC[n]（n＝0～15）のトリガ選択： 5’h00～5’h0f：使用しない 5’h10：iADC_EOC_VC0 を選択 … … 5’h1f：iADC_EOC_VC15 を選択 **備考**　iADC_EOC_VC[n]の各立ち上がりエッジが bADC_INTSTATUS0_VC[n]（n＝0～15）用のトリガになります。	R/W
b7、b6	bADC_Mode	仮想チャネル ADC_VC[n]（n＝0～15）の ADC モード 仮想チャネルの機能モードを選択します。 以下で選択する機能は、ADC2 用の bADC2_Enable ビットと ADC1 用の bADC1_Enable ビットの状態に依存します。詳細については、bADC1_ChannelSel ビットフィールドと bADC2_ChannelSel ビットフィールドの説明を参照してください。 ADC2 および ADC1 のそれぞれ bADC2_Enable ビットおよび bADC1_Enable ビットが有効になっている場合に、仮想チャネル ADC_VC[n]（n＝0～15）のモードを選択します。 **［選択イベント：物理チャネルの変換］** ● 2’b00：物理チャネルの変換 – bADC1_ChannelSel ビットフィールドと bADC2_ChannelSel ビットフィールドは、ADC ステートマシンによって ADC_VC[n]の実行が優先された時点で変換する物理チャネルを選択します。 – コマンド終了時に、結果値が rADC1_DATA[n]レジスタおよび rADC2_DATA[n]レジスタに格納されます。 – 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。 **［選択イベント：サンプル＆ホールド、物理チャネル 6～8 のみ］** ● 2’b01：サンプル＆ホールド – bADC1_ChannelSel ビットフィールドと bADC2_ChannelSel ビットフィールドは、ADC ステートマシンによって ADC_VC[n]の実行が優先された時点でサンプル＆ホールドする物理チャネルを選択します。 – 本機能は、物理チャネル ADC1_IN[8:6]および ADC2_IN[8:6]でのみ利用可能です。 – 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。 **注意）** ADC1 および ADC2 のサンプル＆ホールドに関する注意事項 ● この機能は、物理チャネル 6～8 でのみ利用可能です。 ● 物理チャネル 0～4 に対して本フィールドを 2’b01 にしても、何の影響もありません。 **［選択イベント：割り込みのみおよびコマンド終了］** ● 2’b10： – 物理チャネルに対して何もしません。 – 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。	R/W

表 7.22　rADC_VC[n]レジスタの内容（4/5）

ビット位置	ビット名	機能	R/W
		[選択イベント：データロックレジスタへのデータコピー] ● 2'b11： – 物理チャネルに対して何もしません。 – bADC1_ChannelSel ビットフィールドと bADC2_ChannelSel ビットフィールドでは、ADC ステートマシンによって ADC_VC[n]の実行が優先された時点で、rADC[m]_DATA0～15 から rADC[m]_DATALOCK0～15（n＝0,1）へのコピーを許可または禁止するために使用するマスクデータロック（rADC_MASKLOCK0～3 レジスタ）を選択します。 – 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。 – 可能性のある ADC DMA 要求が起動して、DMA 転送終了の検出まで実行されます。選択される ADC DMA チャネルは、bADC_DMA_Request[1:0]ビットの状態に依存します。	
b5～b3	bADC2_ChannelSel	仮想チャネル ADC_VC[n]（n＝0～15）の ADC2 機能 bADC2_Enable ビットおよび bADC_Mode ビットに応じて、いくつかの機能が利用可能です。 **[選択イベント：なし]** ● bADC2_Enable＝0 の場合： – 何もしません **[選択イベント：物理チャネルの変換]** ● bADC2_Enable＝1 かつ bADC_Mode＝2'b00 の場合 – ADC ステートマシンが ADC_VC[n]の実行を受信したときに変換される物理チャネル ADC2_IN[8:6,4:0]を選択します。 – コマンド終了時に、結果値が rADC2_DATA[n]レジスタに格納されます。 – 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。 ● 変換される物理チャネルを ADC2_ChannelSel ビットフィールドで選択： 3'b000：物理チャネル ADC2_IN0 を選択 … … 3'b100：物理チャネル ADC2_IN4 を選択 3'b101：物理チャネル ADC2_IN6 を選択 … … 3'b111：物理チャネル ADC2_IN8 を選択 **[選択イベント：サンプル＆ホールド、物理チャネル 6～8 のみ]** ● bADC2_Enable＝1 かつ bADC_Mode＝2'b01 の場合 – ADC ステートマシンが ADC_VC[n]の実行を受信したときにサンプル＆ホールドされる物理チャネル ADC2_IN[8:6]を選択します。 – 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。 – サンプル＆ホールドされる物理チャネルを bADC2_ChannelSel ビットフィールドで選択： ● bADC2_ChannelSel[2]： 1：物理チャネル ADC2_IN8 に対してサンプル＆ホールド 0：サンプル＆ホールドなし ● bADC2_ChannelSel[1]： 1：物理チャネル ADC2_IN7 に対してサンプル＆ホールド 0：サンプル＆ホールドなし ● bADC2_ChannelSel[0]： 1：物理チャネル ADC2_IN6 に対してサンプル＆ホールド 0：サンプル＆ホールドなし **[例]** ADC2_ChannelSel[2:0]＝3'b101 の場合 ADC2_IN8 と ADC2_IN6 に対して同時サンプル＆ホールド	R/W

表 7.22　rADC_VC[n]レジスタの内容（5/5）

ビット位置	ビット名	機能	R/W
		［選択イベント：割り込みのみおよびコマンド終了］ ● bADC2_Enable＝1 かつ bADC_Mode＝2’b10 の場合： – 物理チャネルに対して何もしません。 – 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。 ● bADC2_ChannelSel： 3’bxxx：予約 **［選択イベント：データロックレジスタへのデータコピー］** ● bADC2_Enable＝1’b1 かつ bADC_Mode＝2’b11 の場合： – 物理チャネルに対して何もしません。 – bADC2_ChannelSel ビットフィールドでは、ADC ステートマシンによって ADC_VC[n]の実行が優先された時点で、各 rADC2_DATA0～15 レジスタから rADC2_DATALOCK0～15 レジスタへのコピーを許可または禁止するために使用するマスクデータロック（rADC_MASKLOCK0～3 レジスタ）を選択します。 – 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。 – 可能性のある ADC DMA 要求が起動して、DMA 転送終了の検出まで実行されます。選択される ADC DMA チャネルは、bADC_DMA_Request[1:0]ビットの状態に依存します。 ● マスクデータロックレジスタを bADC2_ChannelSel ビットフィールドで選択： 3’b000：rADC_MASKLOCK0 レジスタのマスクを選択 3’b001：rADC_MASKLOCK1 レジスタのマスクを選択 3’b010：rADC_MASKLOCK2 レジスタのマスクを選択 3’b011：rADC_MASKLOCK3 レジスタのマスクを選択 3’b1xx：予約 **注意）** ● ADC2 が実装されていない場合（たとえば、RZ/N1S）、これらのビットは搭載されていないため、読むと 0 が読み出されます。 ● ADC2 が接続されていない場合（RZ/N1D-324）、物理チャネルは“0”にゲートされるため、本チャネルでは変換もサンプル＆ホールドも不可能です。	
b2～b0	bADC1_ChannelSel	仮想チャネル ADC_VC[n]（n＝0～15）の ADC1 機能 bADC2_ChannelSel と同じ	R/W

7.4.1.21　rADC1_DATA[n] ― 仮想チャネル[n]ADC1 変換データ（n＝0～15）

アドレス　4006 5100h＋4h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bADC1_DATA_Update	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bADC1_DATA											
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.23　rADC1_DATA[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31	bADC1_DATA_Update	rADC1_DATA[n]レジスタから各 rADC1_DATALOCK[n]（n＝0～15）レジスタへの最後のコピー以降にデータの更新あり 各仮想チャネル ADC_VC[n]（n＝0～15）に対して： 0：bADC1_DATA でデータ更新なし 1：bADC1_DATA でデータ更新あり **備考**　本ビットは、マスク有効時に rADC1_DATA[n]レジスタから各 rADC1_DATALOCK[n]レジスタへのコピーが実行中であれば 0 にクリアされます。	R
b30～b12	予約ビット	読むと“0”が読み出されます。	R
b11～b0	bADC1_DATA	仮想チャネル ADC_VC[n]（n＝0～15）の ADC1 変換データ ADC1 が仮想チャネル ADC_VC[n]で必要な変換を完了すると、そのデジタル変換値は対応する rADC1_DATA[n]（n＝0～15）レジスタに保存されます。 **［例］** ● 物理チャネル ADC1_IN0 を変換するように ADC_VC5 が設定されている場合、その変換の最終結果は rADC1_DATA5 に保存されます。	R

7.4.1.22　rADC1_DATALOCK[n] ― ADC1 データロック[n]レジスタ（n＝0～15）

アドレス　4006 5180h＋4h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bADC1_DATALOCK_Update	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bADC1_DATALOCK											
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.24　rADC1_DATALOCK[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31	bADC1_DATALOCK_Update	rADC1_DATA[n]レジスタ（n＝0～15）の bADC1_DATA_Update ビットのコピーロック イベント「データロックレジスタへのデータコピー」が ADC_VC ステートマシンによって実行中であるとき、本レジスタは rADC_MASKLOCK0～3 レジスタの管理下で更新されます。 **備考**）bADC1_DATALOCK_Update ビットへ bADC1_DATA_Update のコピーが完了すると（コピーのマスクなし）、bADC1_DATA_Update ビットはハードウェアでクリアされます。	R
b30～b12	予約ビット	読むと“0”が読み出されます。	R
b11～b0	bADC1_DATALOCK	rADC1_DATA[n]レジスタ（n＝0～15）の bADC1_DATA ビットフィールドのコピーロック イベント「データロックレジスタへのデータコピー」が ADC_VC ステートマシンによって実行中であるとき、本レジスタは rADC_MASKLOCK0～3 レジスタの管理下で更新されます。 ロックされたデータは、rADC1_DATA[n]レジスタで実行中の新たな取得と並行して、CPU および DMA による読み出しが可能です。	R

7.4.2　レジスタの説明 ADC2

注　意

これらのレジスタは RZ/N1D にのみ実装されています。その他では読むと 0 が読み出されます。

7.4.2.1　rADC2_DATA[n] ― 仮想チャネル[n]ADC2 変換データ（n＝0～15）

アドレス　4006 5140h＋4h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bADC2_DATA_Update	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bADC2_DATA											
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.25　rADC2_DATA[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31	bADC2_DATA_Update	rADC2_DATA[n]レジスタから各 rADC2_DATALOCK[n]（n＝0～15）レジスタへの最後のコピー以降にデータの更新あり 各仮想チャネル ADC_VC[n]（n＝0～15）に対して： 0：bADC2_DATA ではデータ更新なし 1：bADC2_DATA ではデータ更新あり **備考**　本ビットは、マスク有効時に rADC2_DATA[n]レジスタから各 rADC2_DATALOCK[n]ロックレジスタへのコピーが実行中であれば 0 にクリアされます。	R
b30～b12	予約ビット	読むと“0”が読み出されます。	R
b11～b0	bADC2_DATA	仮想チャネル ADC_VC[n]（n＝0～15）の ADC2 変換データ ADC2 が仮想チャネル ADC_VC[n]で必要な変換を完了すると、そのデジタル変換値は対応する rADC2_DATA[n]（n＝0～15）レジスタに保存されます。 **［例］** ● 物理チャネル ADC2_IN3 をサンプルするように ADC_VC7 が設定されている場合、その変換の最終結果は rADC2_DATA7 に保存されます。	R

7.4.2.2　rADC2_DATALOCK[n] ― ADC2 データロック[n]レジスタ（n＝0～15）

アドレス　4006 51C0h＋4h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bADC2_DATALOCK_Update	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bADC2_DATALOCK											
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 7.26　rADC2_DATALOCK[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31	bADC2_DATALOCK_Update	rADC2_DATA[n]レジスタ（n＝0～15）の bADC2_DATA_Update ビットのコピーロック イベント「データロックレジスタへのデータコピー」が ADC_VC ステートマシンによって実行中であるとき、本レジスタは rADC_MASKLOCK0～3 レジスタの管理下で更新されます。 **備考**　bADC2_DATALOCK_Update ビットへ bADC2_DATA_Update のコピーが完了すると（コピーのマスクなし）、bADC2_DATA_Update ビットはハードウェアでクリアされます。	R
b30～b12	予約ビット	読むと“0”が読み出されます。	R
b11～b0	bADC2_DATALOCK	rADC2_DATA[n]レジスタ（n＝0～15）の bADC2_DATA ビットフィールドのコピーロック イベント「データロックレジスタへのデータコピー」が ADC_VC ステートマシンによって実行中であるとき、本レジスタは rADC_MASKLOCK0～3 レジスタの管理下で更新されます。 ロックされたデータは、rADC2_DATA[n]レジスタで実行中の新たな取得と並行して、CPU および DMA による読み出しが可能です。	R

7.5　動作説明

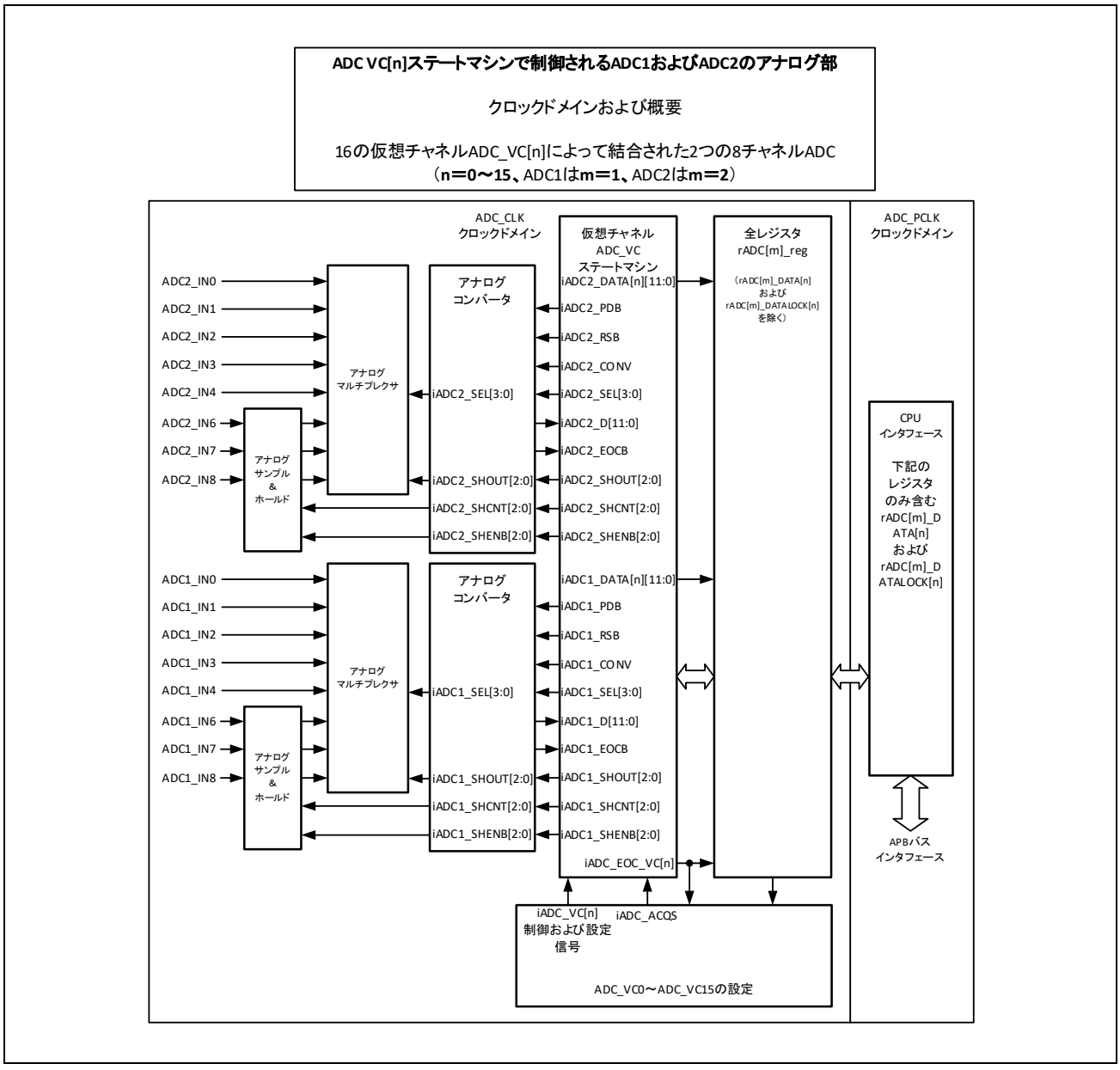


図 7.3　　ADC コントローラの概要

7.5.1　仮想チャネル ADC_VC の動作原理

標準的な ADC タイプとは異なり、本 ADC はシーケンサベースではありません。代わりに、本 ADC は ADC_VC（仮想チャネル）をベースとしています。ADC_VC という言葉は、シングルチャネルまたはデュアルチャネルでの動作を定義するコンフィグレーションセットを表します。このセットには、以下の設定項目が含まれています。

- 動作を開始するトリガ要因
- 処理対象の物理チャネル
- 変換モードまたはサンプル&ホールドモード
- 選択した「変換終了」信号に基づく DMA トランザクション
- 選択した「変換終了」信号に基づく割り込みの設定
- データロックレジスタの更新

それぞれの ADC_VC は個別に設定され、トリガ、チャネル、およびモードの任意の組み合わせが可能です。

同じトリガ、チャネル、および／またはサンプル&ホールドに対して、必要に応じて複数の ADC_VC を設定できます。これによって、チャネルごとに異なるトリガを用いた個別サンプリングから、1 つのトリガを用いた同一チャネルのオーバーサンプリング、そして 1 つのトリガを用いた異なるチャネルの一連の独自変換処理の作成まで、さまざまな変換動作を非常に柔軟に設定できるようになります。

ADC_VC[n]の構成例：

ADC_VC[n]のトリガ要因は、rADC_VC[n]レジスタの bADC_TrigSel ビットフィールド、bADC_TrigEnable ビットフィールド、および bADC_Continuous ビットフィールドの組み合わせとして設定されます。
rADC_FORCE レジスタを用いると、ソフトウェアによって ADC_VC[n]イベントを強制することも可能です。物理チャネルは、rADC_VC[n]レジスタの bADC1_ChannelSel ビットフィールド、bADC2_ChannelSel ビットフィールド、および bADC_Mode ビットフィールドで設定されます。また、サンプル&ホールドウィンドウのサイズは、rADC_ACQS レジスタの bADC_TSHSAMP ビットで設定されます。

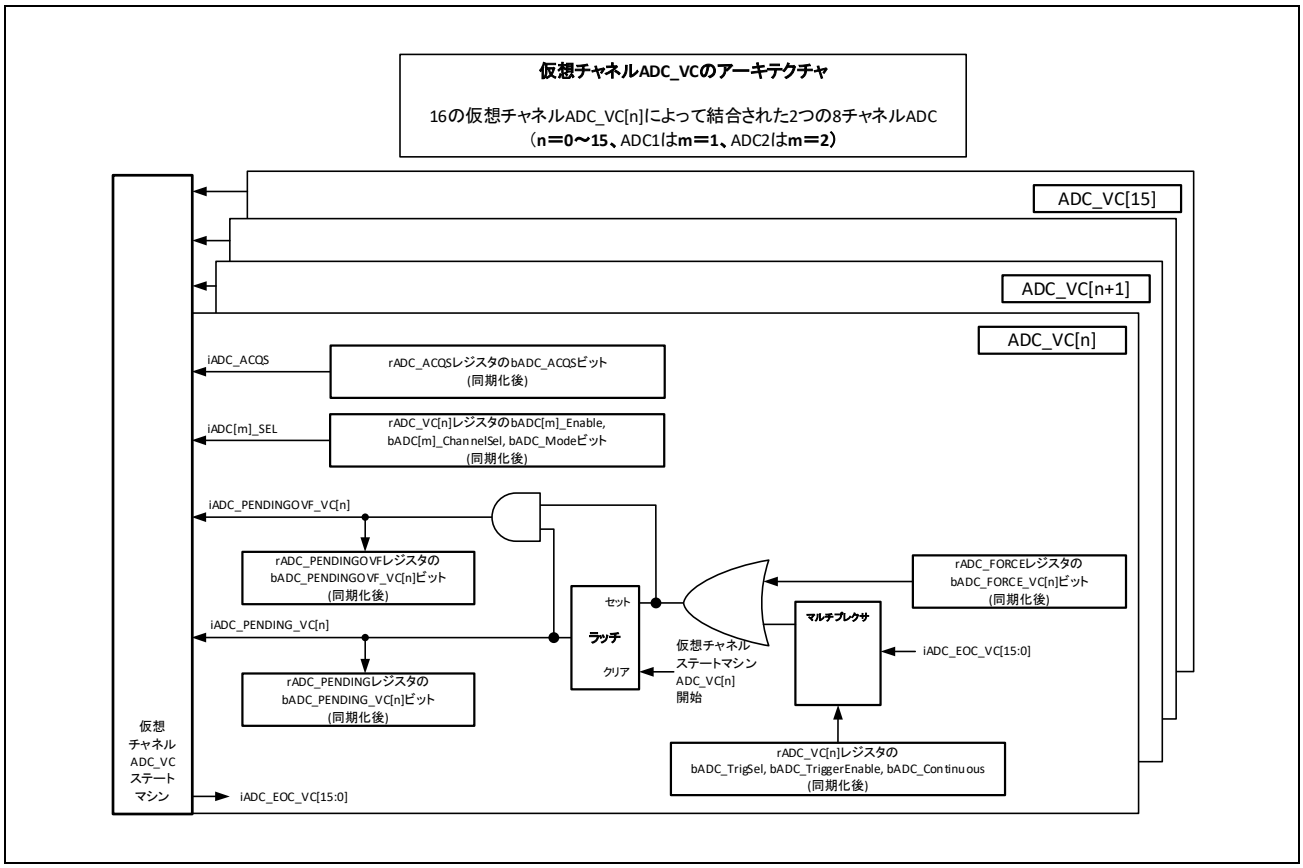


図 7.4　　仮想チャネル ADC_VC のアーキテクチャ

たとえば、iADC_EOC_VC4 が設定されたときにチャネル ADC1_IN1 で単一変換が行われるように構成することが可能です。そのために、仮想チャネル ADC_VC を使用します。次にその rADC_VC[n]レジスタを用いて ADC_VC の 1 つをセットアップします。どの ADC_VC[n]を選択しても違いはないので、ここでは、例として ADC_VC8 を使用することにします。

また、どの ADC トリガを選択しても違いはないので、iADC_EOC_VC4 を使用することにします。許容される最も速い ADC 用のサンプルウィンドウは 6 サイクルです。サンプルウィンドウに最速時間、変換するチャネルには ADC1_IN1 チャネルを選択すると、bADC_TSHSAMP ビットフィールドに 6、bADC_TrigSel ビットフィールドに 5’h14、bADC_TrigEnable ビットと bADC1_Enable ビットには有効モードをそれぞれ指定します。

結果として、各レジスタに書き込む値は以下のようになります。

rADC_VC8：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b0（ADC2 無効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h14（トリガ iADC_EOC_VC4 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h1（ADC1_IN1 チャネルを選択）
- bADC2_ChannelSel：3’hx（使用しない）

rADC_ACQS：

- bADC_TSHSAMP：5’h6（最小値）

このように設定すると、物理チャネル ADC1_IN1 の単一変換は iADC_EOC_VC4 イベントを使用して開始され、結果値は rADC1_DATA8 レジスタに格納されます。

同時に、同じトリガを用いて 4X のオーバーサンプリングで物理チャネル ADC1_IN2 を変換することにします。そのために、ここでは ADC_VC9～12 を使用します。

結果として、各レジスタに書き込む値は以下のようになります。

rADC_VC8：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b0（ADC2 無効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h14（トリガ iADC_EOC_VC4 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h1（ADC1_IN1 チャネルを選択）
- bADC2_ChannelSel：3’hx（使用しない）

rADC_VC9～12（同じ値で構成）：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b0（ADC2 無効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h14（トリガ iADC_EOC_VC4 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h2（ADC1_IN2 チャネルを選択）
- bADC2_ChannelSel：3’hx（使用しない）

rADC_ACQS：

- bADC_TSHSAMP：5’h6（最小値）

rADC_PRIORITY：

- bADC_Priority：5’h0
  ADC_VC0～15 のすべてに対してラウンドロビン調停

このように設定すると、物理チャネル ADC1_IN1 の単一変換は iADC_EOC_VC4 イベントを使用して開始され、結果値は rADC1_DATA8 レジスタに格納されます。

また、物理チャネル ADC1_IN2 の 4 回の変換（4X のオーバーサンプリング）は iADC_EOC_VC4 イベントを使用して開始され、結果値は rADC1_DATA9～12 レジスタに格納されます。

さらに同時に、iADC_EOC_VC7 を使用して開始された同時サンプル＆ホールドによって物理チャネル ADC1_IN[8:6]と ADC2_IN[8:6]を変換することにします。そのために、ここではサンプル＆ホールドのイベ

ント用に ADC_VC0 を使用し、ADC1_IN[8:6]および ADC2_IN[8:6]の変換用に ADC_VC1～3 を使用します。ADC2 用のサンプルウィンドウは 12 サイクルです。

この取得に対しては、高優先順位モードを設定します。

結果として、各レジスタに書き込む値は以下のようになります。

rADC_VC8：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b0（ADC2 無効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h14（トリガ iADC_EOC_VC4 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h1（ADC1_IN1 チャネルを選択）
- bADC2_ChannelSel：3’hx（使用しない）

rADC_VC9～12（同じ値で構成）：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b0（ADC2 無効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h14（トリガ iADC_EOC_VC4 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h2（ADC1_IN2 チャネルを選択）
- bADC2_ChannelSel：3’hx（使用しない）

rADC_VC0：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b1（ADC2 有効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h17（トリガ iADC_EOC_VC7 を選択）
- bADC_Mode：2’b01（サンプル＆ホールド）
- bADC1_ChannelSel：3’h7（ADC1_IN[8:6]チャネルを選択）
- bADC2_ChannelSel：3’h7（ADC2_IN[8:6]チャネルを選択）

rADC_VC1：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b1（ADC2 有効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）

- bADC_TrigSel：5’h17（トリガ iADC_EOC_VC7 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h5（ADC1_IN6 チャネルを選択）
- bADC2_ChannelSel：3’h5（ADC2_IN6 チャネルを選択）

rADC_VC2：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b1（ADC2 有効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h17（トリガ iADC_EOC_VC7 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h6（ADC1_IN7 チャネルを選択）
- bADC2_ChannelSel：3’h6（ADC2_IN7 チャネルを選択）

rADC_VC3：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b1（ADC2 有効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h17（トリガ iADC_EOC_VC7 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h7（ADC1_IN8 チャネルを選択）
- bADC2_ChannelSel：3’h7（ADC2_IN8 チャネルを選択）

rADC_ACQS：

- bADC_TSHSAMP：5’h0C

rADC_PRIORITY：

- bADC_Priority：5’h4
  ADC_VC0～3 を高優先順位とし、ADC_VC4～15 はラウンドロビン方式

このように設定すると、物理チャネル ADC1_IN1 の単一変換は iADC_EOC_VC4 イベントを使用して開始され、結果値は rADC1_DATA8 レジスタに格納されます。

また、物理チャネル ADC1_IN2 の 4 回の変換（4X のオーバーサンプリング）は iADC_EOC_VC4 イベントを使用して開始され、結果値は rADC1_DATA9～12 レジスタに格納されます。

サンプル&ホールドは iADC_EOC_VC7 イベントを使用して開始されます。その後、物理チャネル ADC1_IN[8:6]および ADC2_IN[8:6]の 3 回の変換が（前回のサンプル&ホールド値に基づいて）開始され、その結果値が rADC1_DATA1～3 レジスタと rADC2_DATA1～3 レジスタに格納されます。

上記の例においては、1 つの ADC インスタンスが他のインスタンスに影響を与えないよう制御できます。すなわち、それらを完全に同時使用することが可能です。共通部分は外部の iADC_EOC_VC7 だけです。

7.5.2　電気的 ADC モデルおよびサンプルの取得

外部ドライバごとに、アナログ信号を迅速かつ効果的に駆動する能力が異なります。回路によっては、電荷を ADC のサンプリングコンデンサへ正しく転送するために長時間要するものもあります。

これに対処するため、本 ADC ではサンプルウィンドウの長さに対する制御が可能になっています。rADC_ACQS レジスタの bADC_TSHSAMP ビットフィールドは 5 ビットフィールドであり、これによって ADC ごとにサンプル&ホールド（S/H）ウィンドウのサイズ t_{SHSAMP} が決定されます。

このフィールドに書き込む値は、ADC のサンプリングウィンドウに必要なサイクル数です。許容されるサンプルサイクル数の最小時間値は 300ns です。合計サンプリング時間は、ADC の変換時間にサンプルウィンドウのサイズを加算することによって求めることができます。

以下に示す図には、入力ブロックの等価回路と、外部回路から生じるサンプリング誤差が詳細に説明されています。ここで、

- アナログ入力等価抵抗（R_{IN}）＝マルチプレクサのオン抵抗 + サンプリング SW のオン抵抗
- アナログ入力等価容量（C_{IN}）＝内部寄生容量 + サンプリング容量
- 信号源のインピーダンス（R_s）＝信号源の出力抵抗 + 外部抵抗（ローパスフィルタ）
- 外付け容量（C_e）＝外部寄生容量＋外付け容量（ローパスフィルタ）

電気的特性に記載されている ADC 特性の精度には、外部回路の影響によって生じたサンプリング誤差は含まれていません。

アナログ入力等価抵抗（R_{IN}）を介してアナログ入力等価容量（C_{IN}）に電荷をチャージする際は、サンプリング時の入力電圧とサンプリング時間（t_{SHSAMP}）を 0.1LSB の範囲内に抑える必要があります。

信号源のインピーダンス（R_s）が大きいと、サンプリングのためのチャージ時間が不足して、サンプリング誤差が生じます。そのため、信号源のインピーダンスは（$R_s < 300\Omega$, $C_e < 15$ pF）を満たすように減少させてください。

（t_{SHSAMP}）：サンプル時間ウィンドウ：ADC_CLK 周波数の 6 クロック周期

- 最小値：300ns（ADC_CLK＝20MHz の場合）
- 最大値：1500ns（ADC_CLK＝4MHz の場合）

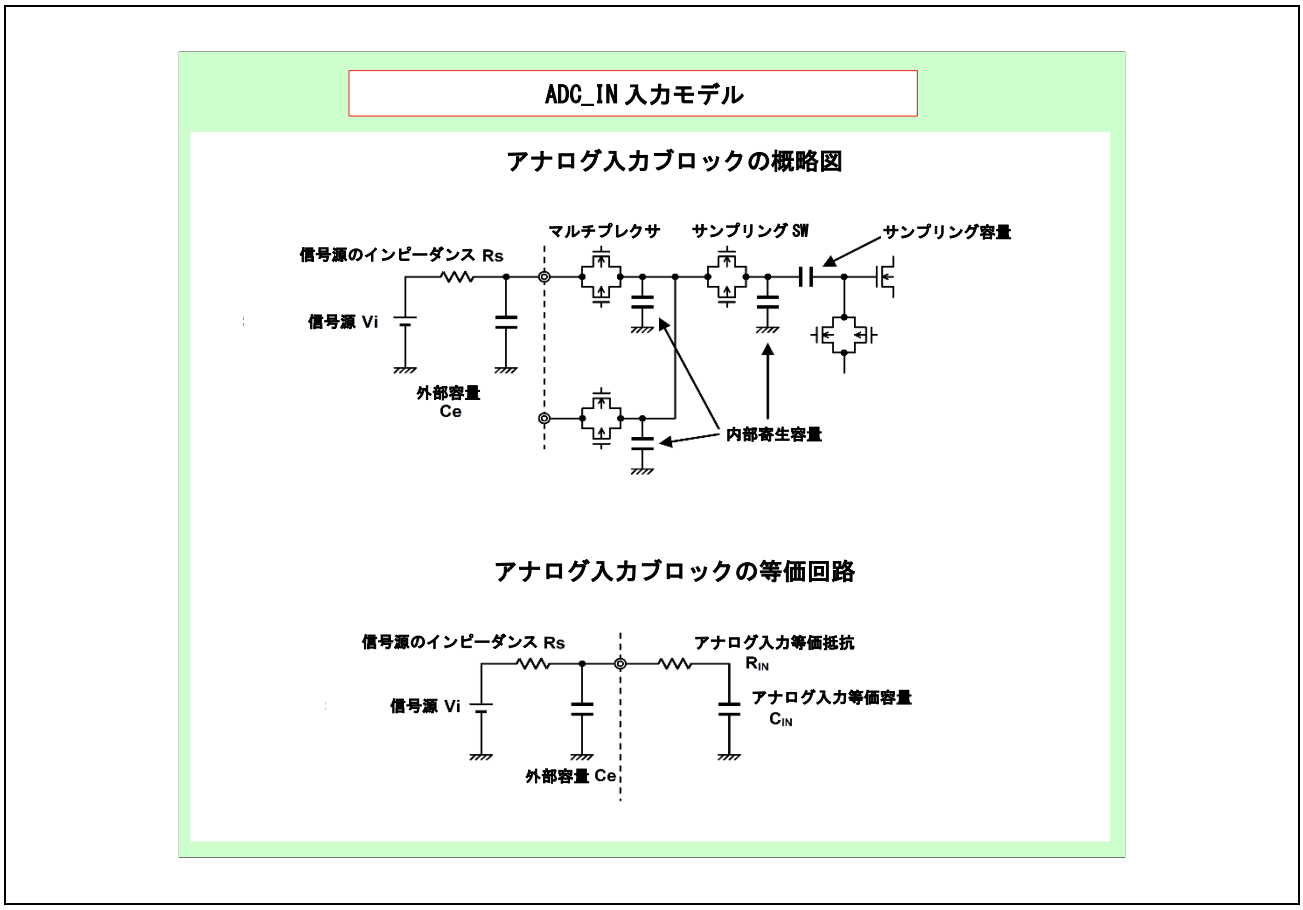


図 7.5　　ADC アナログ入力モデル

不要な高周波信号（たとえば、ノイズ）を除去するため、アナログ入力にローパスフィルタを挿入することが推奨されます。ローパスフィルタの RC パラメータ（R_s および C_e）は、応用機器の特性に合わせて選択してください。

AC 信号を扱う応用機器では、信号源のインピーダンスを十分低くする必要があります。AC 特性が重要でない場合は、大容量の外付け容量（C_e）を追加するとサンプリング誤差が低減されます。

サンプリング誤差の概算値は次式で得られます。

$$\text{Sampling error} \fallingdotseq \frac{C_{IN}}{Ce + C_{IN}} \times 4096\ [\text{LSB}]$$

Ce があまり大きくない場合、サンプリング誤差の概算値は次式で得られます。

$$\text{Sampling error} \fallingdotseq \frac{C_{IN}}{Ce + C_{IN}} \times \exp\left(-\frac{t_{AS}}{Rs \times Ce + (R_s + R_{IN}) \times C_{IN}}\right) \times 4096\ [\text{LSB}]$$

スイッチングマルチプレクサを用いて周期的な繰り返しの A/D 変換を行う場合は、信号源のインピーダンス（R_s）と外付け容量（C_e）のもとで、次式に示すようなサンプリング誤差が生じます。

$$\text{Sampling error} \fallingdotseq \left(\frac{C_{IN}}{Ce + C_{IN}} + \frac{R_S \times C_{IN}}{t_{SC}}\right) \times 4096 \text{ [LSB]}$$

t_{SC}：マルチプレクサ走査の周期。マルチプレクサ走査とは、A/D コンバータのスイッチング入力による周期的な繰り返しの A/D 変換を意味します。

たとえば、ADC1_IN0 から ADC1_IN8 まで連続して A/D 変換を行い、ADC1_IN8 の後に ADC1_IN0 に戻ります。

7.5.3　トリガ選択およびイベント管理

各 ADC_VC[n]（n＝0～15）は、さまざまな入力トリガの 1 つで開始するように設定できます。必要であれば、同じチャネルに複数の ADC_VC[n]を設定することも可能です。

使用できるトリガは以下のとおりです。

- rADC_FORCE レジスタの bADC_FORCE_VC[n]ビットへの CPU 書き込み（bADC_SETFORCE_VC ビットのセット）：
  - ADC_VC[n]が優先された時点で、ADC 要求（設定に応じて、変換、サンプル＆ホールド、またはその他の動作）の開始を強制します。
- iADC_EOC_VC[n]イベント、ADC_VC[n]の動作終了：
  - 動作終了時に ADC_VC[n]ステートマシンによって駆動されます。
  - このモードは、ある ADC_VC から別の ADC_VC へと連続した変換ストリームが必要な場合に役立ちます。
  - rADC_VC[n]レジスタの bADC_TrigEnable ビットおよび bADC_TrigSel ビットで設定します。
- 単一モード：
  - 単一モードの場合、ADC 動作（変換、サンプル＆ホールド、その他）は 1 回限り実行されます。
  - bADC_TrigEnable ビットは、選択したチャネルでの動作が終了すると、自動的に 0 にクリアされます。
  - bADC_FORCE_VC[n]ビットは、選択したチャネルでの動作が終了すると、自動的に 0 にクリアされます。
  - rADC_VC[n]レジスタの bADC_Continuous ビット＝1’b0 で設定します。
- 連続モード：
  - 連続モードの場合、ADC 動作は選択した複数チャネルに対して順番に連続して実行されます。これは、bADC_TrigEnable ビットおよび bADC_FORCE_VC[n]ビットが、ソフトウェア／ハードウェアリセットまたはファームウェア ADC リセットによってクリアされるまで続きます。
  - 選択したチャネルでの動作が終了したとき、bADC_TrigEnable ビットおよび bADC_FORCE_VC[n]ビットに変化はありません。
  - このモードは、ADC_VC から連続する変換ストリームが必要な場合に役立ちます。
  - rADC_VC[n]レジスタの bADC_Continuous ビット＝1’b1 で設定します。

以下の場合に、仮想チャネル ADC_VC[n]の rADC_PENDING レジスタの bADC_PENDING_VC[n]ビットでイベントが受信されます。

- bADC_TrigEnable ビット＝1’b1 の設定
  - 仮想チャネル ADC_VC[n]の bADC_TrigSel ビットフィールドで選択したトリガの立ち上がり検出
- rADC_FORCE レジスタの bADC_FORCE_VC[n]ビットへの CPU 書き込み（bADC_SETFORCE_VC ビットのセット）
  - ADC_VC[n]が優先された時点で、開始のための ADC 要求（設定に応じて、変換、サンプル&ホールド、またはその他の動作）を強制します。
- ADC_VC[n]に対する ADC イベント（変換、サンプル&ホールド、その他の動作）が保留中

以下の場合に、rADC_PENDING レジスタで保留中のイベント（変換、サンプル&ホールド、その他の動作）は、ADC1 および／または ADC2 で実行する必要があります。

- ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択された場合
  - bADC1_Enable ビットが 1 になっていれば、bADC_Mode と bADC1_ChannelSel で制御される各動作が ADC1 で実行されます。
  - bADC2_Enable ビットが 1 になっていれば、bADC_Mode と bADC2_ChannelSel で制御される各動作が ADC2 で実行されます
- 同時に、rADC_PENDING レジスタで保留中のイベントに関して、以下の操作が行われます。
  - 各 ADC_VC[n]の動作が開始されると、rADC_PENDING レジスタの bADC_PENDING_VC[n]ビットは自動的にクリアされます。
  - 単一モードの場合、ADC 動作（変換、サンプル&ホールド、その他）は 1 回限り実行されます。
    選択したチャネルでの動作が終了すると、bADC_TrigEnable ビットは自動的に 0 にクリアされます。
    選択したチャネルでの動作が終了すると、bADC_FORCE_VC[n]ビットは自動的に 0 にクリアされます。
  - 連続モードの場合、ADC 動作は選択した複数チャネルに対して順番に連続して実行されます。これは、bADC_TrigEnable ビットおよび bADC_FORCE_VC[n]ビットが、ソフトウェア／ハードウェアリセットまたはファームウェア ADC リセットによってクリアされるまで続きます。

以下の場合に、仮想チャネル ADC_VC[n]の rADC_PENDINGOVF レジスタの bADC_PENDINGOVF_VC[n]ビットでイベントオーバーフローが受信されます。

- 既存イベントがすでに保留中の状態で、新たに ADC_VC[n]イベントが発生した場合
- オーバーフロー状態が、ADC_VC[n]イベントの処理を中断させることはありません。それは単に、トリガが失敗したことを示しています。
- rADC_PENDINGCLROVF レジスタの bADC_PENDINGCLROVF_VC[n]ビットを 1 に書き込みすると、bADC_PENDINGOVF_VC[n]ビットがクリアされます。

「**図 7.4 仮想チャネル ADC_VC のアーキテクチャ**」を参照してください。

7.5.4　物理チャネルの選択

各 ADC_VC[n]（n＝0～15）は、利用可能な ADC[m]_IN[8:6,4:0]入力チャネル（m＝1、2）のいずれかを変換するように設定できます。

- ADC_VC[n]を変換モード（bADC_Mode ビット＝2’b00）に設定した場合
  - rADC_VC[n]レジスタの bADC[m]_ChannelSel フィールド（bADC1_ChannelSel および bADC2_ChannelSel）で、変換する物理チャネルを定義します。
  - bADC1_Enable ビット＝1 であれば ADC1 で、bADC2_Enable ビット＝1 であれば ADC2 で、変換が実行されます。
  - コマンド終了時に、結果値が rADC[m]_DATA[n]レジスタ（rADC1_DATA[n]および rADC2_DATA[n]）に格納されます。
- ADC_VC[n]を同時サンプリングモード（bADC_Mode ビット＝2’b01）に設定した場合
  - rADC_VC[n]レジスタの bADC[m]_ChannelSel フィールド（bADC1_ChannelSel および bADC2_ChannelSel）で、サンプル＆ホールド動作を処理する物理チャネルを定義します。この場合、サンプル＆ホールド動作は、各 ADC の物理チャネル 6～8 でのみ利用可能です。
  - bADC1_Enable ビット＝1 であれば ADC1 で、bADC2_Enable ビット＝1 であれば ADC2 で、サンプル＆ホールドが実行されます。

「**図 7.4 仮想チャネル ADC_VC のアーキテクチャ**」を参照してください。

7.5.5　ADC 動作の優先順位

複数の ADC_VC[n]（n＝0～15）フラグが同時にセットされると、それらの変換順序は 2 種類の方式のいずれかで決定されます。デフォルトの方式はラウンドロビンです。この方式では、どの ADC_VC[n]も他より高い固有の優先順位を持つことはありません。優先順位はラウンドロビンポインタ（bADC_RR_Pointer ビット）に依存します。

rADC_PRIORITY レジスタに表示された bADC_RR_Pointer は、最後に変換された ADC_VC[n]を指し示しています。そして、bADC_RR_Pointer の値よりも 1 つ大きな値の ADC_VC[n]に対して最高の優先順位が与えられます。ADC_VC15 の次は ADC_VC0 に戻ります。

値の 0 は変換がすでに行われたことを意味するため、リセット時の値には 16 が用いられます。
bADC_RR_Pointer の値が 16 であると、ADC_VC0 に最高の優先順位が与えられます。

bADC_RR_Pointer は、下記のいずれかの条件でリセットされます。

- デバイスリセットが行われたとき
- rADC_PRIORITY レジスタに書き込みがなされたとき
- rADC_CONFIG レジスタへの CPU 書き込みによって以下の動作を開始したとき
  - bADC_POWER_DOWN（低消費電力モード有効または無効）
  - bADC_SAMPLE_HOLD_ENABLE（各チャネルのサンプル＆ホールド機能有効または無効）
  - 変換が現在進行中であれば、それが終了した後、新しい優先順位になります。

ラウンドロビン順位の決定方法の例を下図に示します。

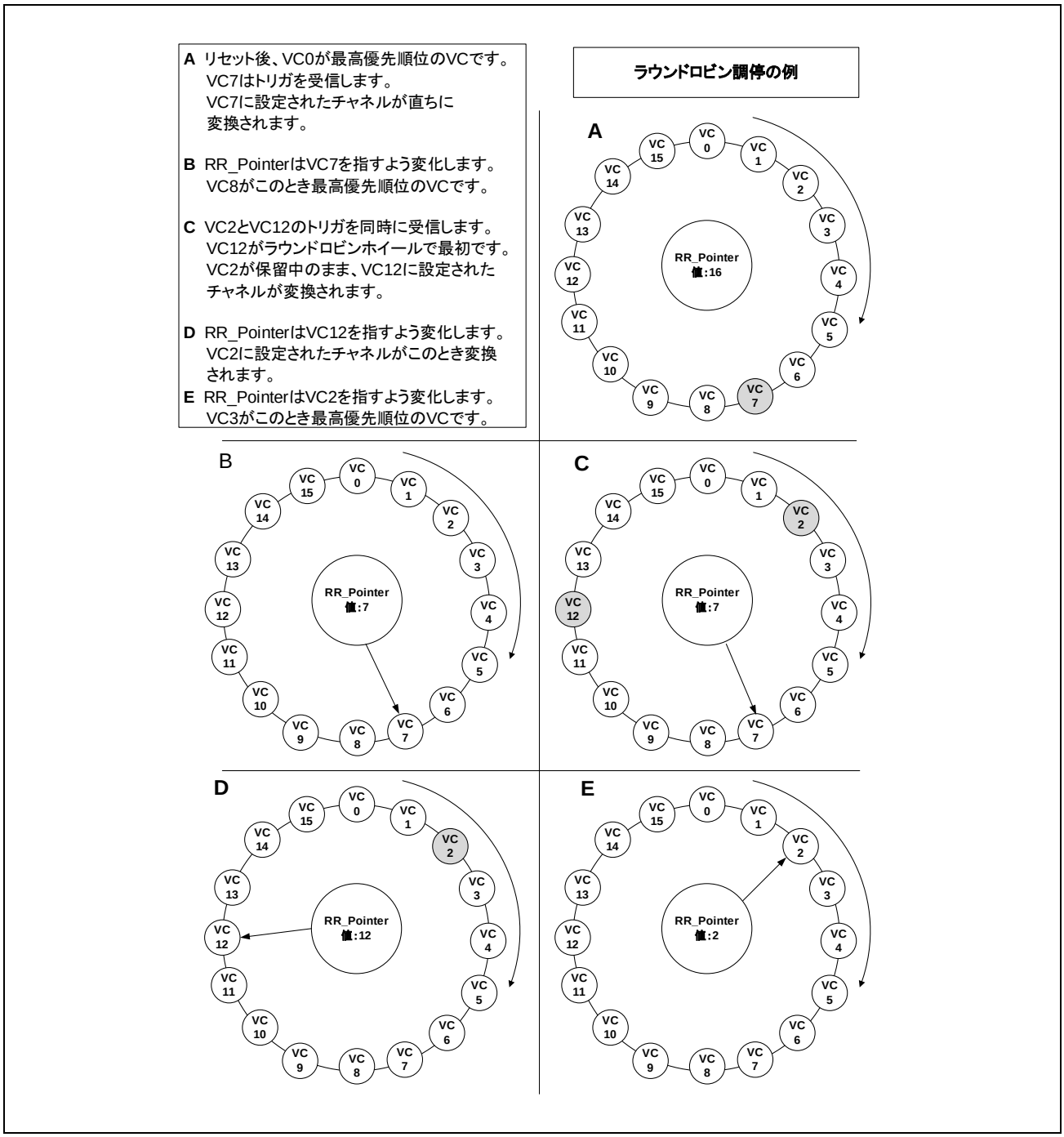


図 7.6　　ADC のラウンドロビン調停の例

rADC_PRIORITY レジスタの bADC_Priority フィールドを使用すると、1 つ以上の ADC_VC に対して高優先順位モードを割り当てることが可能です。高優先順位モードが割り当てられた ADC_VC[n]は、現在の変換が完了するとラウンドロビンホイールを中断し、自分自身を次の変換対象にします。その変換が完了すると、ラウンドロビンホイールが中断していた地点から再開します。高優先順位モードの 2 つの ADC_VC が同時にトリガされた場合は、番号の小さい方の ADC_VC が優先されます。

高優先順位モードの割り当ては、ADC_VC0 に始まって昇順に続きます。bADC_Priority ビットフィールドに書き込む値は、高優先順位モードとしない最初の ADC_VC 番号です。たとえば bADC_Priority ビットに 4 を書き込むと、ADC_VC0～3 が高優先順位モードとなり、番号の最も若い ADC_VC0 が最優先されます。

高優先順位モードの ADC_VC[n]の使用例を下図に示します。

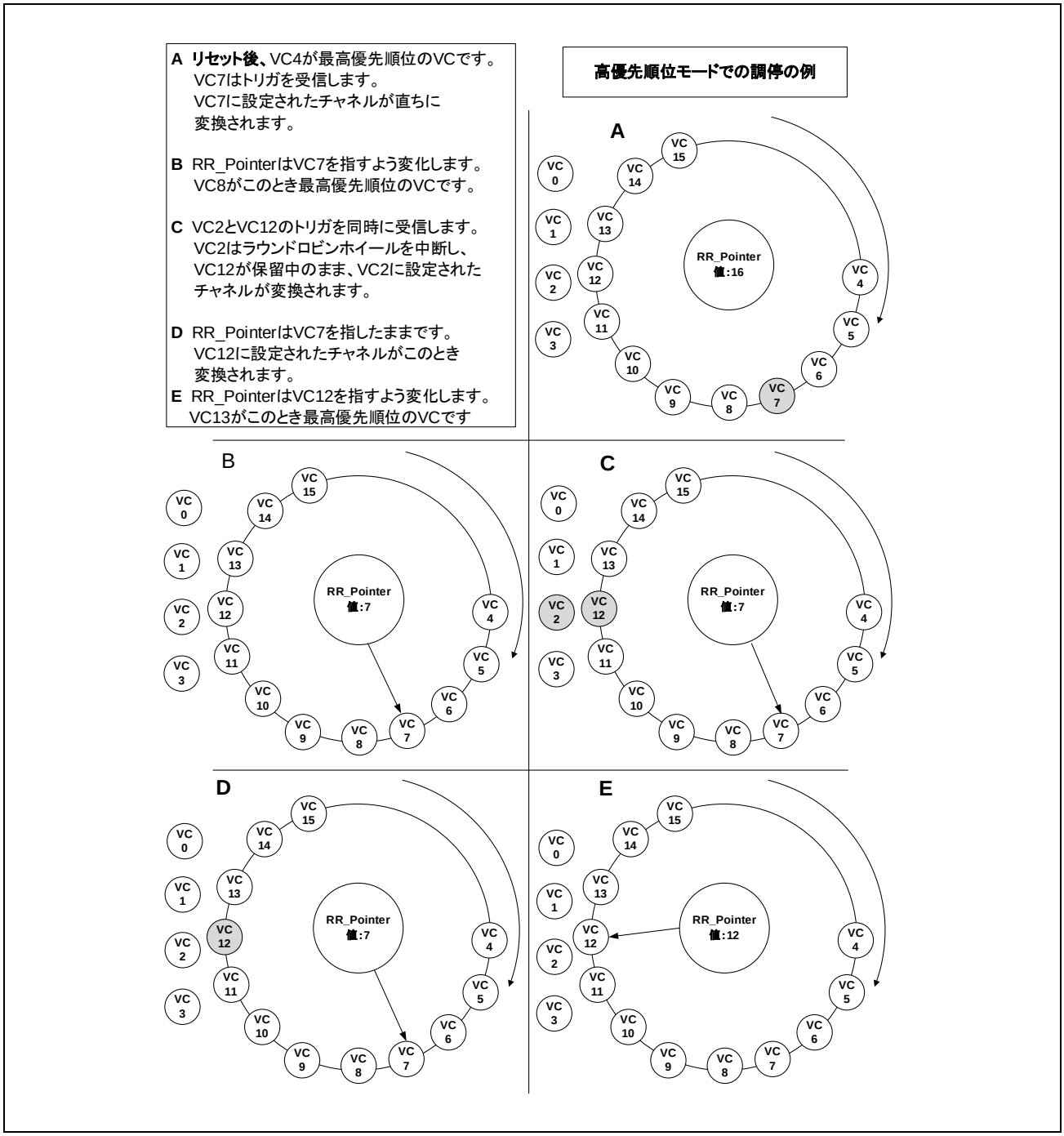


図 7.7　　ADC 高優先順位モードでの調停の例

7.5.6　同時サンプル＆ホールド

一部の応用機器では、2 つ以上の信号のサンプリング間の遅延を最小限に抑えることが重要になります。各 ADC には、3 つの異なるチャネルの同時サンプリングを可能にする三重サンプル＆ホールド回路が内蔵されています。

- ADC_VC[n]を同時サンプリングモード（bADC_Mode ビット＝2’b01）に設定した場合、rADC_VC[n]レジスタの bADC[m]_ChannelSel ビットフィールドでは、サンプル＆ホールド動作を処理するためのチャネルを指定します。
- これらの機能は、各 ADC の物理チャネル 6～8 でのみ利用可能です。

「**図 7.4 仮想チャネル ADC_VC のアーキテクチャ**」および「**図 7.3 ADC コントローラの概要**」を参照してください。

下記の例では、iADC_EOC_VC7 を使用して開始した同時サンプル＆ホールドによって物理チャネル ADC1_IN[8:6]と ADC2_IN[8:6]を変換することにします。そのために、ここではサンプル＆ホールドのイベント用に ADC_VC0 を使用し、ADC1_IN[8:6]および ADC2_IN[8:6]の変換用に ADC_VC1～3 を使用します。ADC2 用のサンプルウィンドウは 12 サイクルです。

この取得に対しては、高優先順位モードを設定します。

結果として、各レジスタに書き込む値は以下のようになります。

rADC_VC0：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b1（ADC2 有効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h17（トリガ iADC_EOC_VC7 を選択）
- bADC_Mode：2’b01（サンプル＆ホールド）
- bADC1_ChannelSel：3’h7（ADC1_IN[8:6]チャネルを選択）
- bADC2_ChannelSel：3’h7（ADC2_IN[8:6]チャネルを選択）

rADC_VC1：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b1（ADC2 有効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h17（トリガ iADC_EOC_VC7 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h5（ADC1_IN6 チャネルを選択）
- bADC2_ChannelSel：3’h5（ADC2_IN6 チャネルを選択）

rADC_VC2：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b1（ADC2 有効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h17（トリガ iADC_EOC_VC7 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h6（ADC1_IN7 チャネルを選択）
- bADC2_ChannelSel：3’h6（ADC2_IN7 チャネルを選択）

rADC_VC3：

- bADC1_Enable：1’b1（ADC1 有効）
- bADC2_Enable：1’b1（ADC2 有効）
- bADC_Continuous：1’b0（単一変換）
- bADC_TrigEnable：1’b1（トリガ有効）
- bADC_TrigSel：5’h17（トリガ iADC_EOC_VC7 を選択）
- bADC_Mode：2’b00（物理チャネルの変換）
- bADC1_ChannelSel：3’h7（ADC1_IN8 チャネルを選択）
- bADC2_ChannelSel：3’h7（ADC2_IN8 チャネルを選択）

rADC_ACQS：

- bADC_TSHSAMP：5’h0C

rADC_PRIORITY：

- bADC_Priority：5’h4
  ADC_VC0～3 を高優先順位とし、ADC_VC4～15 はラウンドロビン方式
  「**図 7.7 ADC 高優先順位モードでの調停の例**」を参照してください。

サンプル&ホールドは iADC_EOC_VC7 イベントを使用して開始されます。その後、物理チャネル ADC1_IN[8:6]および ADC2_IN[8:6]の 3 回の変換が（前回のサンプル&ホールド値に基づいて）開始され、その結果値が rADC1_DATA1～3 レジスタと rADC2_DATA1～3 レジスタに格納されます。

上記の例においては、1 つの ADC インスタンスが他のインスタンスに影響を与えないよう制御できます。すなわち、それらを完全に同時使用することが可能です。共通部分は外部の iADC_EOC_VC7 だけです。

結合動作は以下のとおりです。

- トリガ iADC_EOC_VC7 が、同時に、ADC_VC0 に対してのサンプル&ホールド処理、および、ADC_VC1～3 に対しての変換を開始します。
- 調停後、より高い優先順位の ADC_VC0 が選択され、物理チャネル ADC1_IN[8:6]および ADC2_IN[8:6]上で同時サンプル&ホールド動作を実行します。
- 調停後、より高い優先順位の ADC_VC1 が選択され、物理チャネル 6 上で各 ADC のデータを変換します。
- 調停後、より高い優先順位の ADC_VC2 が選択され、物理チャネル 7 上で各 ADC のデータを変換します。
- 調停後、より高い優先順位の ADC_VC3 が選択され、物理チャネル 8 上で各 ADC のデータを変換します。
- 各変換が完了すると、結果データが ADC[m]_DATA1～3 レジスタ（m＝1、2）に格納されます。
- すべての動作（サンプル&ホールドおよび変換）の終了時に、割り込みを生成するように ADC_VC3 を設定することが可能です。

7.5.7　コマンド終了（EOC）および割り込み動作

独立した 16 個の ADC_VC[n]コンフィグレーションセットがあり、それらは 16 個の iADC_EOC_VC[n]パルス（ADC_VC[n]の動作終了信号）で管理されます。

すなわち、iADC_EOC_VC[n]の立ち上がりエッジによって、rADC_INTSTATUS0 レジスタおよび rADC_INTSTATUS1 レジスタの bADC_INTSTATUS0_VC[n]ビットおよび bADC_INTSTATUS1_VC[n]ビットに割り込みが設定されます。

- bADC_INTSTATUS0_VC[n]ビット：
  - 仮想チャネル ADC_VC[n]（n＝0～15）のマスキング前の割り込みステータス
- bADC_INTSTATUS1_VC[n]ビット：
  - 仮想チャネル ADC_VC[n]（n＝0～15）のマスキング後の割り込みステータス
  - 本レジスタの 1 つのビットが High であるとき、割り込みが ADC_Int に現れます。
- bADC_INTMASK_VC[n]ビット：
  - rADC_INTSTATUS0 レジスタの割り込みビットをマスクします。
- bADC_INTCLR_VC[n]ビット：
  - rADC_INTSTATUS0 レジスタと rADC_INTSTATUS1 レジスタの各割り込みビットをクリアします。

さらに、各 bADC_INTSTATUS0_VC[n]ビットがセットされ、かつ選択した追加の iADC_EOC_VC[n]トリガが生成されると、rADC_INTOVFSTATUS0 レジスタおよび rADC_INTOVFSTATUS1 レジスタの bADC_INTOVFSTATUS0_VC[n]ビットおよび bADC_INTOVFSTATUS1_VC[n]ビットにオーバーフロー状態が発生します。

- bADC_INTOVFSTATUS0_VC[n]ビット：
  - 仮想チャネル ADC_VC[n]（n＝0～15）のマスキング前の割り込みオーバーフローステータス
- bADC_INTOVFSTATUS1_VC[n]ビット：
  - 仮想チャネル ADC_VC[n]（n＝0～15）のマスキング後の割り込みオーバーフローステータス
  - 本レジスタの 1 つのビットが High であるとき、割り込みが ADC_Int に現れます。
- bADC_INTOVFMASK_VC[n]ビット：
  - rADC_INTOVFSTATUS0 レジスタと rADC_INTOVFSTATUS1 レジスタの各割り込みビットをマスクします。
- bADC_INTCLROVF_VC[n]ビット：
  - rADC_INTOVFSTATUS0 レジスタと rADC_INTOVFSTATUS1 レジスタの各割り込みビットをクリアします。

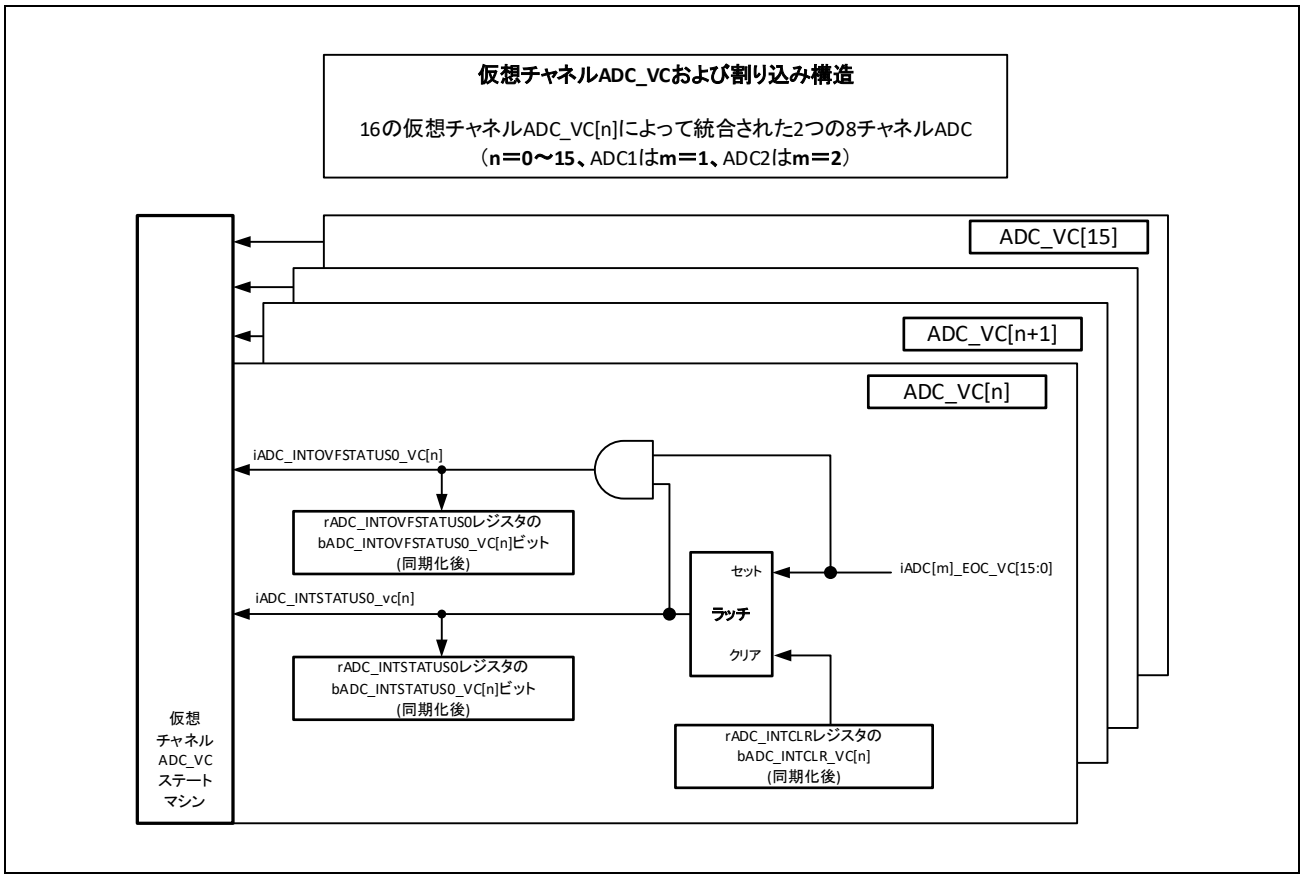


図 7.8　　ADC 割り込み構造

7.5.8　データロックレジスタへのデータコピー

各 ADC_VC[n]（ADC1 m:1、ADC2 m:2、n＝0～15）は、ADC_VC[n]が「データロックレジスタへのデータコピー」モード（bADC_Mode＝2’b11）に設定されているとき、rADC[m]_DATA0～15 レジスタから rADC[m]_DATALOCK0～15 レジスタへデータをコピーするように設定できます。

以下の場合に、rADC_PENDING レジスタで保留中のイベント（データロックレジスタへのデータコピー動作）は、ADC1 および／または ADC2 で実行する必要があります。

- ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択された場合
  - bADC1_Enable ビットが 1 になっていれば、bADC_Mode ビットフィールド（2’b11 に設定）と bADC1_ChannelSel ビットフィールドで制御される各動作が ADC1 で実行されます。
  - bADC2_Enable ビットが 1 になっていれば、bADC_Mode ビットフィールド（2’b11 に設定）と bADC2_ChannelSel ビットフィールドで制御される各動作が ADC2 で実行されます
- bADC1_ChannelSel ビットフィールドと bADC2_ChannelSel ビットフィールドでは、rADC[m]_DATA0～15 から rADC[m]_DATALOCK0～15（m＝1、2）へのコピーを許可または禁止するために使用するマスクデータロック（rADC_MASKLOCK0～3 レジスタ）を選択します。
- 割り込み管理または他の ADC_VC[n]のトリガ入力のために、ADC_VC[n]のコマンド終了 iADC_EOC_VC[n]が生成されます。
- 可能性のある ADC DMA 要求が開始され、DMA 転送終了の検出まで実行されます。選択した ADC DMA チャネルは、bADC_DMA_Request[1:0]ビットの状態に依存します。

詳細については、下図を参照してください。

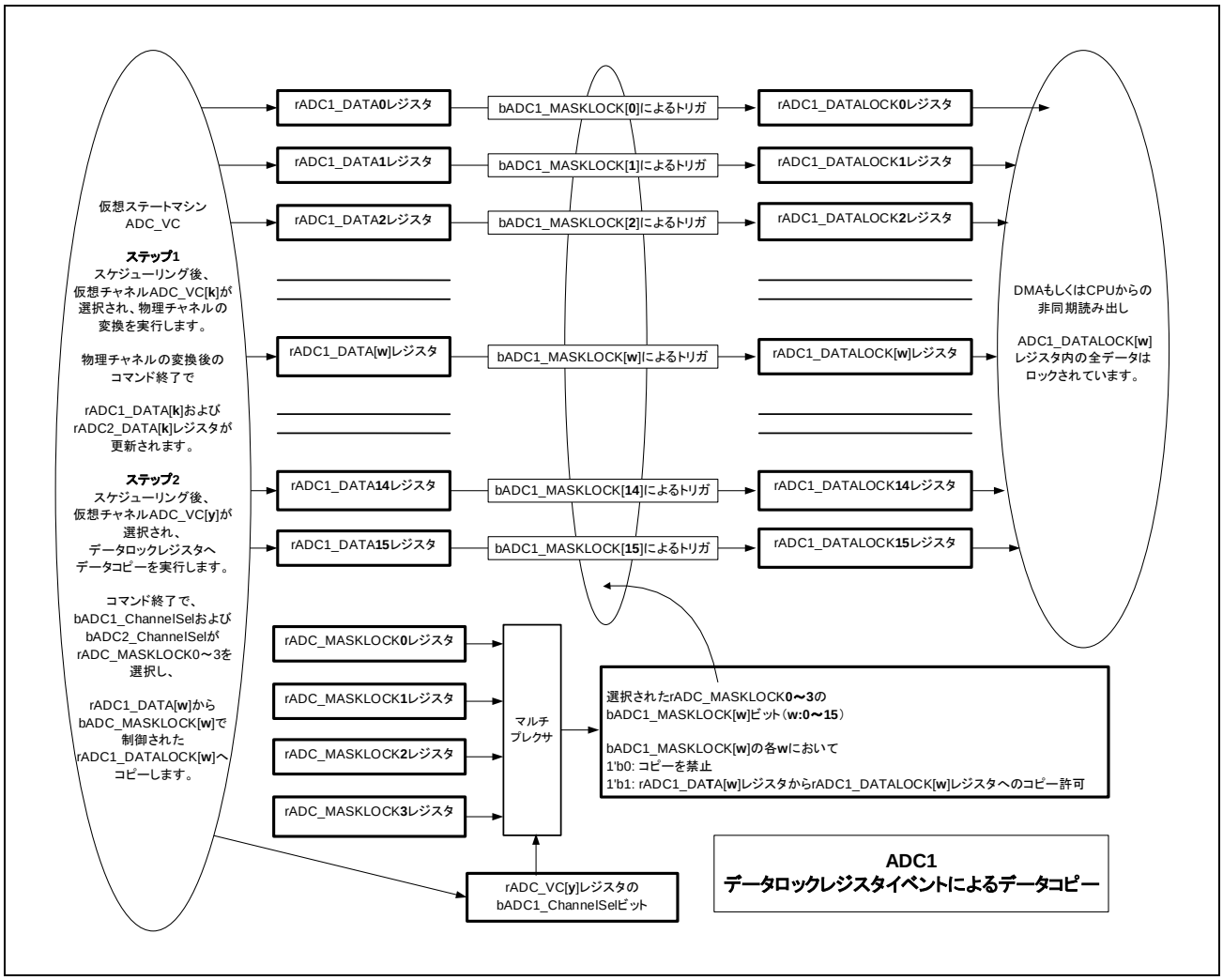


図 7.9　　ADC データロックレジスタへのデータコピー

7.5.9　タイミング

7.5.9.1　3 チャネルでの基本的 A/D 変換

サンプル&ホールド機能を使用しない A/D 変換は以下のように 3 ステップで実行されます。

- 物理チャネル 0 での A/D 変換、仮想チャネル ADC_VC1 による制御
- 物理チャネル 1 での A/D 変換、仮想チャネル ADC_VC2 による制御
- 物理チャネル 2 での A/D 変換、仮想チャネル ADC_VC3 による制御
- A/D 変換はチャネルごとに実行（rADC_PRIORITY レジスタでの優先順位の設定に依存）
- 「コマンド終了」検出時の割り込みの設定
- CPU による ACK 応答
- 各 A/D 変換は、クロックの立ち上がりエッジ時に iADC[m]_CONV（m＝1、2、ADC に依存）の設定によって起動

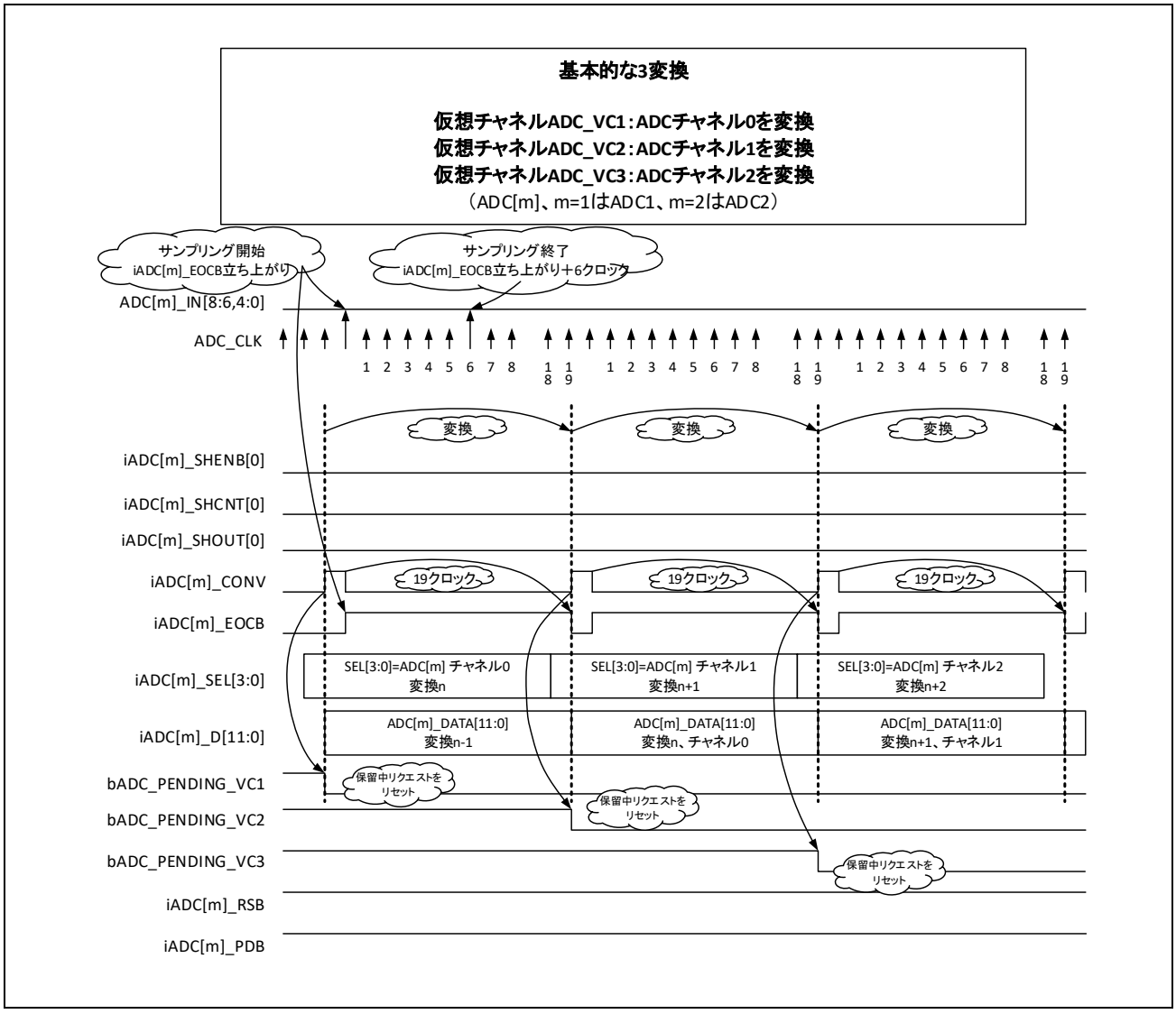


図 7.10　3 チャネルでの基本的 A/D 変換（サンプル＆ホールドなし）

7.5.9.2　サンプル&ホールドに続く 1 チャネルでの A/D 変換

サンプル&ホールド機能を使用した A/D 変換は以下のように 2 ステップで実行されます。

- 物理チャネル 6 でのサンプル&ホールド動作（A/D 変換なし）
  - 仮想チャネル ADC_VC0 による制御
- 物理チャネル 6 での A/D 変換
  - 仮想チャネル ADC_VC1 による制御
- A/D 変換はチャネルごとに実行（rADC_PRIORITY レジスタでの優先順位の設定に依存）
- 「コマンド終了」検出時の割り込みの設定
- CPU による ACK 応答
- A/D 変換は、クロックの立ち上がりエッジ時に iADC[m]_CONV（m＝1、2）の設定によって起動
- ADC を低消費電力モードから動作モードに変更した場合、CPU は ADC が安定するために必要な復帰時間（1μs）を待つ必要があります（rADC_CONTROL レジスタの bADC_BUSY ビットから 0 を読み出してください）。
- サンプル&ホールド回路による 1 回サンプリングで、A/D 変換を 1 回だけ実行できます。別の A/D 変換を実行したい場合は、再度サンプリングする必要があります。
- サンプル&ホールドモードのホールド時間は最大 15ms であり、これはファームウェアで管理する必要があります。

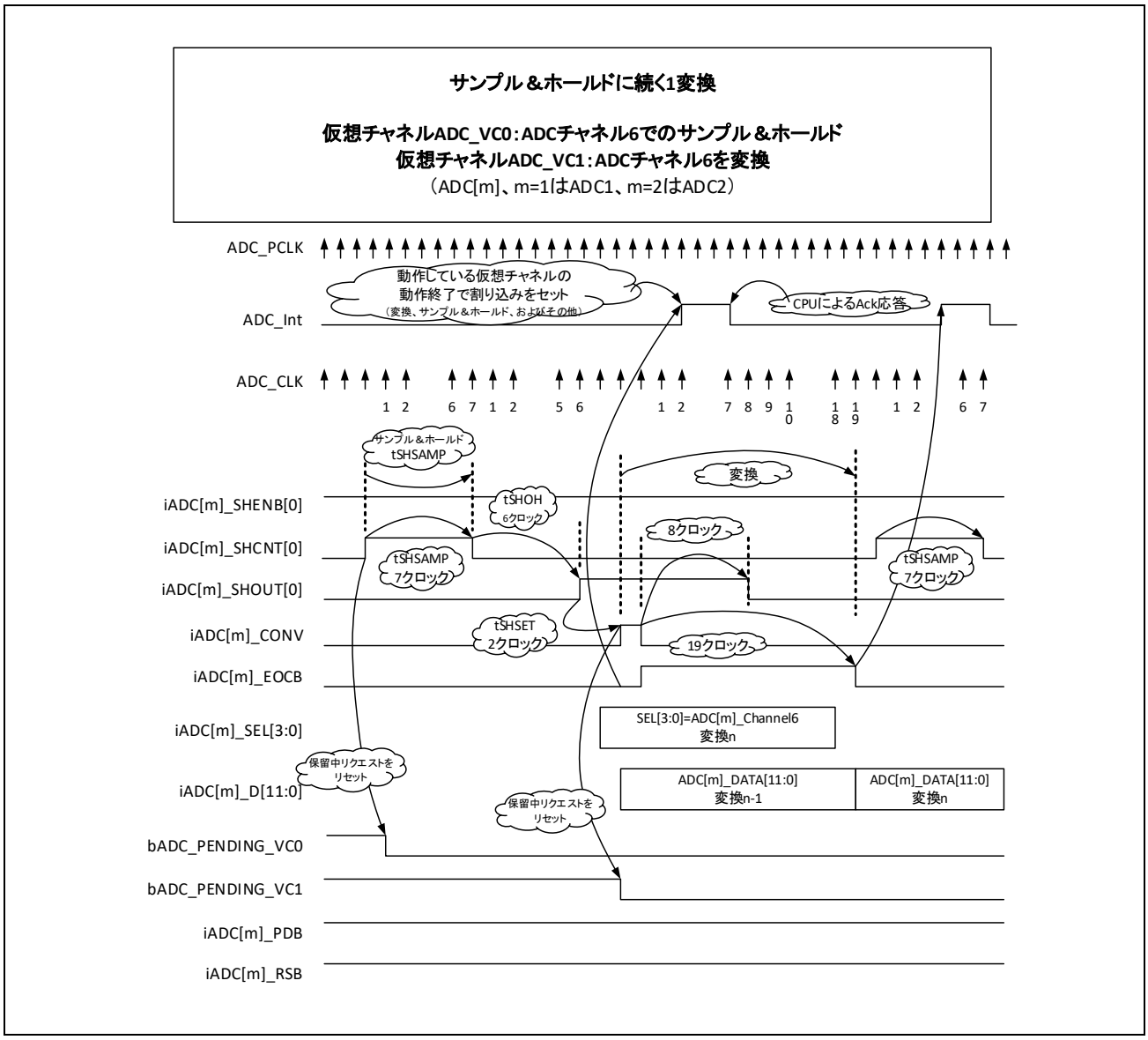


図 7.11　　サンプル＆ホールドに続く 1 チャネルでの A/D 変換

7.5.9.3　サンプル&ホールドに続く 3 チャネルでの A/D 変換

サンプル&ホールド機能を使用した A/D 変換は以下のように 4 ステップで実行されます。

- 物理チャネル 6～8 でのサンプル&ホールド動作（A/D 変換なし）
  - 仮想チャネル ADC_VC0 による制御
- 物理チャネル 6 での A/D 変換、仮想チャネル ADC_VC1 による制御
- 物理チャネル 7 での A/D 変換、仮想チャネル ADC_VC2 による制御
- 物理チャネル 8 での A/D 変換、仮想チャネル ADC_VC3 による制御
- A/D 変換はチャネルごとに実行（rADC_PRIORITY レジスタでの優先順位の設定に依存）
- 「コマンド終了」検出時の割り込みの設定
- CPU による ACK 応答
- A/D 変換は、クロックの立ち上がりエッジ時に iADC[m]_CONV（m＝1、2）の設定によって起動
- ADC を低消費電力モードから動作モードに変更した場合、CPU は ADC が安定するために必要な復帰時間（1μs）を待つ必要があります（rADC_CONTROL レジスタの bADC_BUSY ビットから 0 を読み出してください）。
- サンプル&ホールド回路による 1 回サンプリングで、A/D 変換を 3 回だけ実行できます。別の A/D 変換を実行したい場合は、再度サンプリングする必要があります。
- サンプル&ホールドモードのホールド時間は最大 15ms であり、これはファームウェアで管理する必要があります。

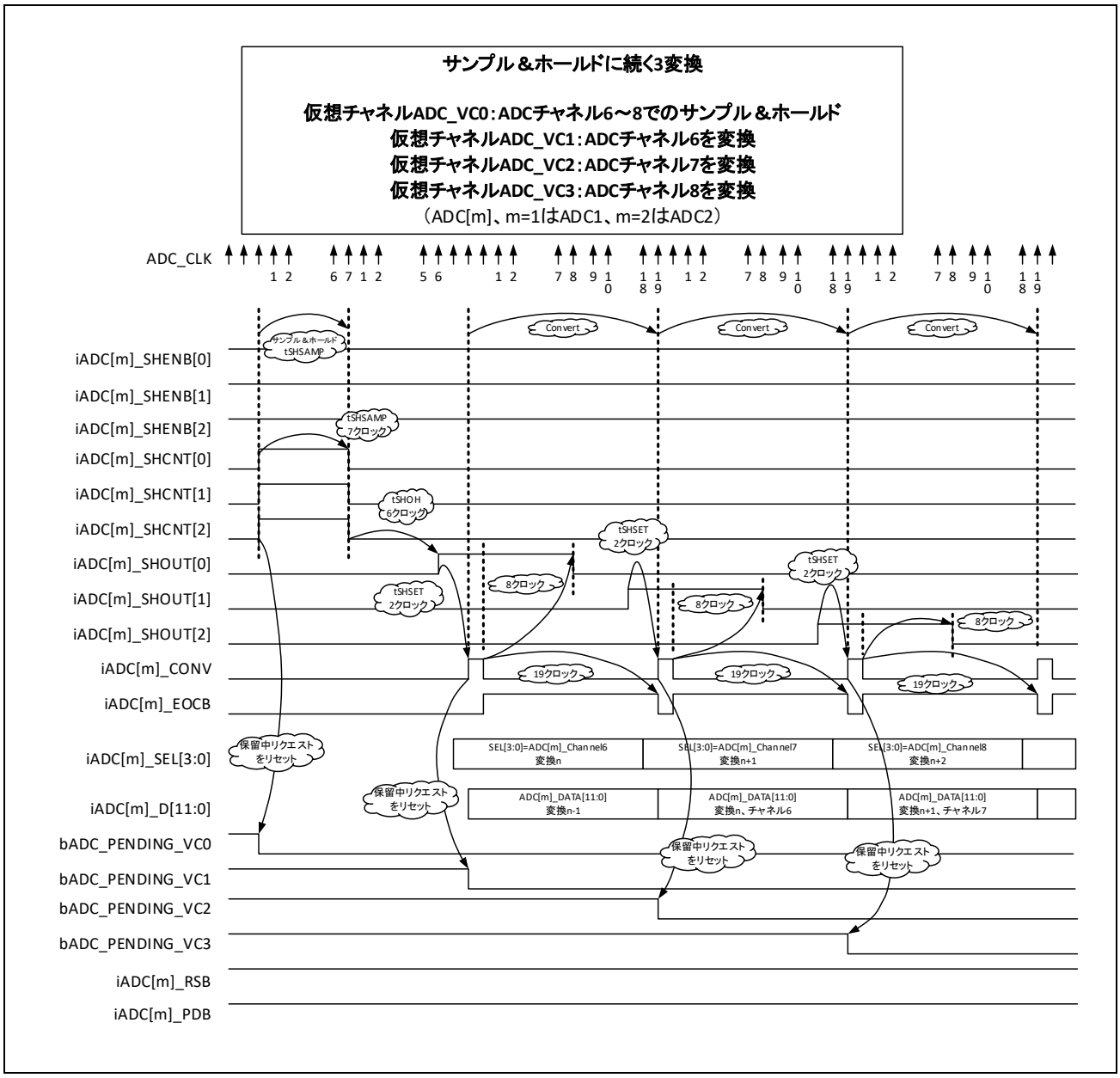


図 7.12　　サンプル＆ホールドに続く 3 チャネルでの A/D 変換

7.5.9.4　低消費電力モード

低消費電力モードのシーケンスは、以下のステップで実行されます。

- 物理チャネル 0 での現在の A/D 変換
  - 仮想チャネル ADC_VC1 による制御
- CPU 書き込みによって、rADC_CONFIG レジスタの bADC_POWER_DOWN ビットを 1 にセット
  - iADC[m]_PDB 信号と iADC[m]_RSB 信号（m＝1、2）を 0 にクリア
  - rADC_CONTROL レジスタの bADC_BUSY ビットを 1 にセット
- すべての rADC_VC[n]レジスタ（n＝0～15）の bADC_TrigEnable ビットを 0 にクリア（トリガ無効）
  - rADC_PENDING レジスタの bADC_PENDING_VC[n]ビット（n＝0～15）（仮想チャネル ADC_VC[n]の動作保留中）を 0 にクリア
  - rADC_FORCE レジスタの bADC_FORCE_VC[n]ビット（n＝0～15）（仮想チャネル ADC_VC[n]の動作強制開始）を 0 にクリア
  - 現在実行中の変換を停止させて、現在の動作の終了時に、仮想ステートマシンに対してアイドル状態を強制
- rADC_CONTROL レジスタの bADC_BUSY ビットから 0 が読み出されるまで（コンフィグレーションの変更が終了するまで）、CPU によって bADC_BUSY ビットをポーリング

- CPU 書き込みによって、rADC_CONFIG レジスタの bADC_POWER_DOWN ビットを 0 にセット
  - iADC[m]_PDB 信号と iADC[m]_RSB 信号（m＝1、2）を 1 にセット
  - rADC_CONTROL レジスタの bADC_BUSY ビットから 0 が読み出されるまで、bADC_BUSY ビットは 1 にセット（復帰時間 ≧ 1μs）
  - 復帰時間の経過後、ADC1 および ADC2 を動作モードに設定
- この時点で、CPU は全仮想チャネル ADC_VC[n]（n＝0～15）を再構成している必要があります。
- 物理チャネル 4 での A/D 変換
  - 仮想チャネル ADC_VC2 による制御
- A/D 変換はチャネルごとに実行（rADC_PRIORITY レジスタでの優先順位の設定に依存）
- 「コマンド終了」検出時の割り込みの設定
- CPU による ACK 応答
- A/D 変換は、クロックの立ち上がりエッジ時に iADC[m]_CONV（m＝1、2）の設定によって開始

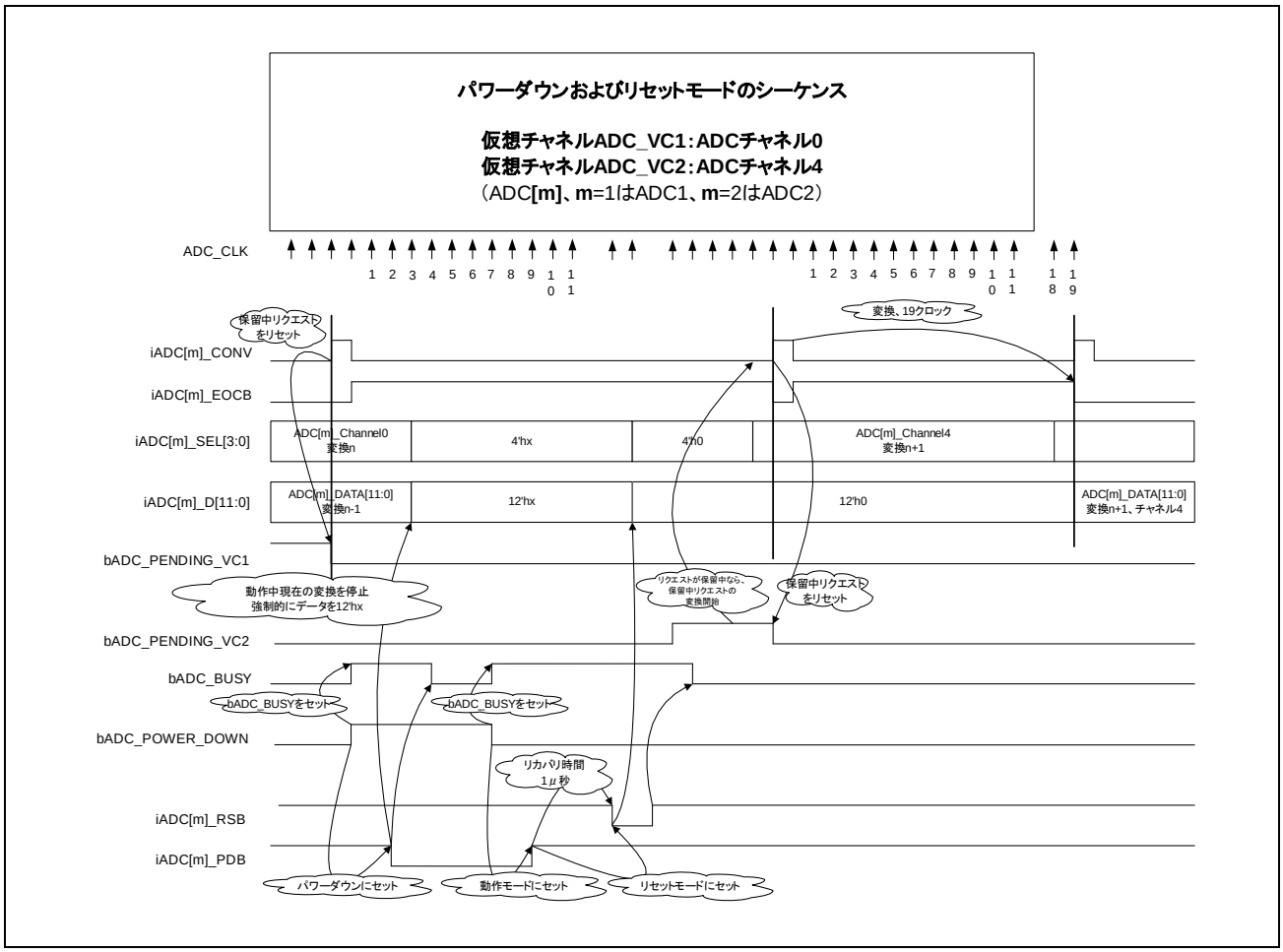


図 7.13　　ADC 低消費電力モード

7.5.9.5　A/D 変換速度

以前の各図より、A/D 変換速度を計算できます。

表 7.27　　A/D 変換速度

モード	ADC クロック周期の数	1 回の変換あたりの ADC クロック周期の数（平均値）
基本の A/D 変換	20	20
3 チャネル同時サンプル＆ホールド	(7＋6＋2)＋3×(20)	25
3 チャネル独立サンプル＆ホールド	3×((7＋6＋2)＋20))	35

7.5.10　DMA 制御

本 ADC コントローラには DMA 機能があります。この機能は、転送の要求および制御のために、DMA コントローラに対するハンドシェイクインタフェースをサポートしています。DMA へのデータ転送には APB バスが用いられます。このモードでは、DMA コントローラは DMAC フローコントローラモードに設定する必要があります。DMA は効率性を向上させるため、可能な場合は常に DMA バーストトランザクションを使用してデータを転送します。

本 ADC コントローラは、2 つの DMA チャネルを使用して ADC の変換値（rADC[m]_DATALOCK[n]レジスタ（n＝0～15、m＝1、2））を転送します。

DMA ADC のフロー制御は下記の DMA ビットによって管理されます。

- rADC_CONFIG レジスタの bADC_DMA ビット
  - DMA チャネル 0 および 1 を許可または禁止
- rADC_VC[n]レジスタ（n＝0～15）の bADC_DMA_Request ビット
  - ADC_VC ステートマシンによるスケジューリングの終了後、ADC_VC[n]の実行が選択されて、イベント「データロックレジスタへのデータコピー」が検出された場合、ADC DMA 要求が開始され、DMA 転送終了の検出まで実行されます。選択した ADC DMA チャネルは、bADC_DMA_Request[1:0]ビットの状態に依存します。
- rADC_PENDING レジスタの bADC_DMA0_RUNNING ビットおよび bADC_DMA1_RUNNING ビット
  - ADC DMA 要求が開始され、DMA 転送終了の検出まで実行されます。
- rADC1_DATALOCK[n]レジスタおよび rADC2_DATALOCK[n]レジスタ（n＝0～15）
  - DMA または CPU 読み出しのために、rADC1_DATA[n]レジスタ（n＝0～15）の bADC1_DATA ビットのコピーロック
  - DMA または CPU 読み出しのために、rADC2_DATA[n]レジスタ（n＝0～15）の bADC2_DATA ビットのコピーロック
  - 「**図 7.9 ADC データロックレジスタへのデータコピー**」を参照してください。

ADC の DMA コントローラインタフェースを有効にして、ハンドシェイクインタフェースを有効にするには、以下の設定を行います。

- DMA コントローラのコンフィグレーション（必須）
  - 転送元アドレスおよび転送先アドレス
  - 転送元および転送先のバーストサイズ
  - 転送ブロックサイズ
  - DMAC フローコントローラモード
  - チャネル割り当て
  - 1 ブロックのみ
  - 割り込み
- ADC DMA チャネルを許可するため、rADC_CONFIG レジスタの bADC_DMA ビットをセット
  - イベント「データロックレジスタへのデータコピー」を検出したとき、rADC_VC[n]レジスタ（n＝0～15）の bADC_DMA_Request ビットで設定されたチャネルの ADC DMA 要求が開始され、DMA 転送終了の検出まで実行されます。

- rADC_PENDING レジスタの bADC_DMA0_RUNNING ビットと bADC_DMA1_RUNNING ビットは、実行中の DMA トランザクションの現在の状態を示します。
- DMA 転送には、rADC1_DATALOCK[n]レジスタおよび rADC2_DATALOCK[n]レジスタ（n＝0～15）のみを使用
  - rADC1_DATA[n]レジスタおよび rADC2_DATA[n]レジスタのコピーロック
  - 「**図 7.9 ADC データロックレジスタへのデータコピー**」を参照してください。

7.5.10.1　DMA 動作の概要

注　意

- ADC は転送されるブロックサイズを認識していないため、DMA コントローラは DMAC フローコントローラモードに設定する必要があります。
- ADC は 32 ビット幅のみをサポートします。

バーストモードおよびシングルモードですべてのトランザクションを正しく管理するために推奨される値

- APB バスのサイズ：32 ビット
- バーストラインサイズ：4×32 ビット～16×32 ビット

たとえば、DMA コントローラ内にプログラムされたブロックサイズが 12 で、バーストトランザクション長が 4 に設定されている場合、ブロックサイズはバーストトランザクション長の倍数です。そのため、DMA ブロック転送は一連のバーストトランザクションで構成されます。

ADC がこのチャネルに対して受信要求を行うと、4 個のデータ項目が rADC1_DATALOCK[n]レジスタと rADC2_DATALOCK[n]レジスタから読み出されます。12 個すべてのデータ項目を読み出す前に、この DMA チャネルに対して 3 回の独立した要求を行う必要があります。

注　意

ADC レジスタは 32 ビット幅であるため、DMAC 転送元の転送幅（DMAC.CTL[n].SRC_TR_WIDTH）の設定値は 3’b010 にする必要があります。

7.6　使用上の注意事項

7.6.1　制約事項

DMA を 2 チャネル同時に使用すると、DMA 要求が正常に動作しない場合があります。

このとき、rADC_PENDING レジスタの bADC_DMA1_RUNNING/bADC_DMA0_RUNNING ビットおよび rADC_PENDINGOVF レジスタの bADC_DMA1_RUNNINGOVF/bADC_DMA0_RUNNINGOVF ビットは正常ではない可能性があります。このため DMA を使用する場合は、1 チャネルのみ有効にしてください。

第8章　LCD コントローラ

8.1　概要

RZ/N1 の PG4（ペリフェラルグループ 4）は LCD コントローラを搭載しています。

- 広い範囲の LCD パネル解像度を設定可能
- 1 ポート TFT LCD パネル用インタフェース
  - 18 ビットデジタル（6 ビット／色）
  - 24 ビットデジタル（8 ビット／色）
- フレームバッファのピクセルあたりのビット数（bpp）を設定可能
  - カラーパレットを介して 1、2、4、8bpp を 18 ビット LCD ピクセルにマッピング
  - 16、18bpp は 18 ビット LCD ピクセルを直接駆動
  - 24bpp は 24 ビット LCD ピクセルを直接駆動
- カラーパレット RAM：256 ワード×16 ビット
- 出力フォーマットを設定可能：
  - RGB 6:6:6、5:6:5、または 5:5:5（18 ビットデジタルインタフェース）
  - RGB 5:6:5（24 ビットデジタルインタフェース）（パレットを介した 8bpp による）
  - RGB 8:8:8（24 ビットデジタルインタフェース）
- ハードウェア点滅サポート
- LCD パネル LED バックライト輝度制御用に 2 つのパルス幅変換モジュール
  - LCD_PWM[1:0]
- パワーアップおよびパワーダウンシーケンスのサポート
- 統合 DMA
- サポートされる LCD パネルの特性

パネルインタフェース	ピクセルあたりのビット数	パレットサイズ	色数
18/24 ビット（6 ビット／色）	1	2 エントリ×16 ビット	2
18/24 ビット（6 ビット／色）	2	4 エントリ×16 ビット	4
18/24 ビット（6 ビット／色）	4	16 エントリ×16 ビット	16
18/24 ビット（6 ビット／色）	8	256 エントリ×16 ビット	256
18/24 ビット（6 ビット／色）	16	—	32768／65536
18/24 ビット（6 ビット／色）	18	—	262144
24 ビット（8 ビット／色）	8	256 エントリ×16 ビット	256
24 ビット（8 ビット／色）	24	—	16777216

- 2 つの FIFO メモリ
  - 出力 FIFO：16 ワード、24 ビット
  - 入力 FIFO：1K ワード、64 ビット

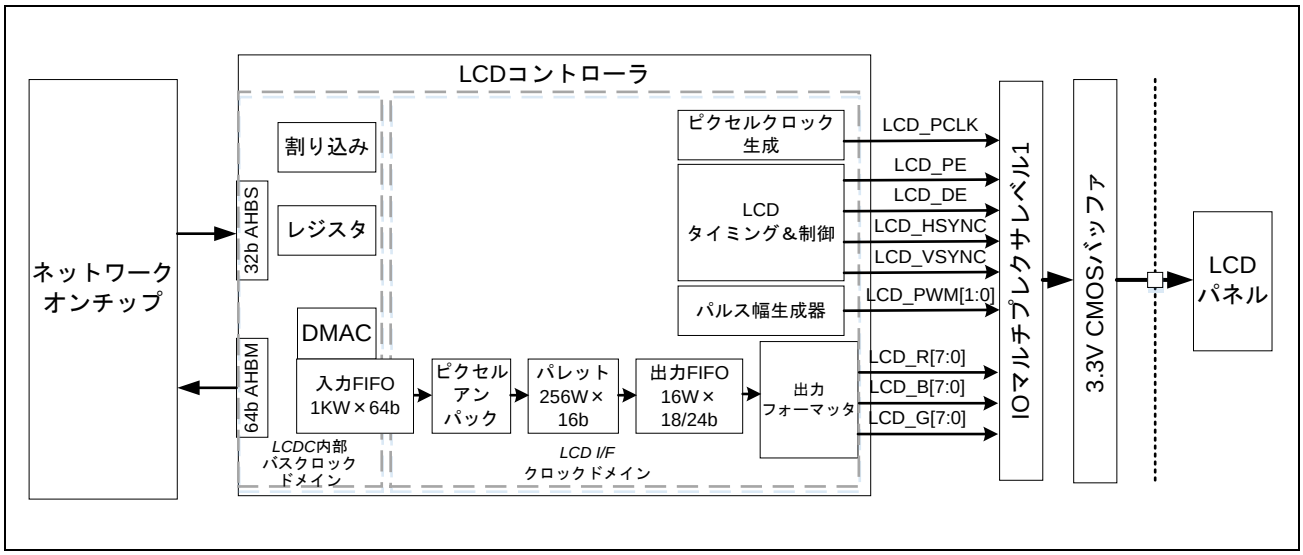


図 8.1　　LCD コントローラの概要

8.2　信号インタフェース

表 8.1　LCD 信号インタフェース

信号名	入力 出力	説明
クロック		
LCD_HCLK	入力	内部バスクロック（AHB）
LCD_ECLK	入力	リファレンスクロック（ピクセルクロックドメイン）
割り込み		
LCDC_Int	出力	レベル検出割り込み出力、アクティブ High
外部信号		
LCD_PCLK	出力	ピクセルクロック
LCD_HSYNC	出力	水平同期パルス
LCD_VSYNC	出力	垂直同期パルス
LCD_DE	出力	データイネーブル
LCD_PE	出力	パワーイネーブル
LCD_PWM[1:0]	出力	LCD LED パルス幅変換
LCD_R[7:0]	出力	赤データ　#プログラマブル赤／青入れ替えモード
LCD_G[7:0]	出力	緑データ
LCD_B[7:0]	出力	青データ　#プログラマブル赤／青入れ替えモード

8.3　レジスタマップ

表 8.2　レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5300 4000h	rLcd_CR1	コントロールレジスタ 1
5300 4008h	rLcd_HTR	水平方向タイミングレジスタ
5300 400Ch	rLcd_VTR1	垂直方向 1 タイミングレジスタ
5300 4010h	rLcd_VTR2	垂直方向 2 タイミングレジスタ
5300 4014h	rLcd_PCTR	ピクセルクロックタイミングレジスタ
5300 4018h	rLcd_ISR	マスキング前の割り込みステータスレジスタ
5300 401Ch	rLcd_IMR	割り込みマスクレジスタ
5300 4020h	rLcd_IVR	マスキング後の割り込みステータスレジスタ
5300 4024h	rLcd_ISCR	割り込みスキャンコンペアレジスタ
5300 4028h	rLcd_DBAR	フレームバッファメモリの DMA 開始ベースアドレス
5300 402Ch	rLcd_DCAR	進行中の DMA カレントベースアドレス
5300 4030h	rLcd_DEAR	DMA 終了アドレス
5300 4034h	rLcd_PWMFR_0	PWM0 周波数レジスタ
5300 4038h	rLcd_PWMDCR_0	PWM0 デューティサイクルレジスタ
5300 4044h	rLcd_HVTER	水平方向および垂直方向タイミング拡張レジスタ
5300 4048h	rLcd_HPPLOR	水平方向ラインあたりピクセル数オーバーライド制御
5300 404Ch	rLcd_PWMFR_1	PWM1 周波数レジスタ
5300 4050h	rLcd_PWMDCR_1	PWM1 デューティサイクルレジスタ
5300 41F8h	rLcd_GPIOR	点滅コントロール
5300 41FCh	rLcd_CIR	コア識別レジスタ

8.3.1　コーディングパレット（パレットレジスタ）マップ

注　意

- コーディングパレットは、選択されたカラーモードに依存します。
- コーディングパレットのアドレスにマッピングされたシンボルは、各モードで異なります。
- 下表に、すべてのカラーモードのコーディングパレットのシンボル（名前）を示します。

アドレス	レジスタシンボル	レジスタ名
5300 4200h～ 5300 43FCh	rLcd_PAL_RGB_555	RGB 5:5:5 モード時のコーディングパレット

アドレス	レジスタシンボル	レジスタ名
5300 4200h～ 5300 43FCh	rLcd_PAL_RGB_565	RGB 5:6:5 モード時のコーディングパレット

アドレス	レジスタシンボル	レジスタ名
5300 4200h～ 5300 43FCh	rLcd_PAL_BGR_555	BGR 5:5:5 モード時のコーディングパレット

アドレス	レジスタシンボル	レジスタ名
5300 4200h～ 5300 43FCh	rLcd_PAL_BGR_565	BGR 5:6:5 モード時のコーディングパレット

8.4　レジスタの説明

8.4.1　rLcd_CR1 ― コントロールレジスタ 1

アドレス　5300 4000h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	bLcd_FBP	bLcd_LPS	bLcd_FDW	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_PSS	bLcd_OPS			bLcd_VSP	bLcd_HSP	bLcd_PCP	bLcd_DEP	bLcd_EBO	bLcd_EPO	bLcd_RGB	bLcd_BPP			bLcd_LPE	bLcd_LCE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.3　rLcd_CR1 レジスタの内容（1/3）

ビット位置	ビット名	機能	R/W
b31～b20	予約ビット	読むと 0 が読み出されます。	R
b19	bLcd_FBP	フレームバッファ 24bpp パックワード 0：パックなし。32 ビットフレームバッファワードあたり 24bpp。上位 8 ビットは未使用。 1：24bpp パック。3 つのフレームバッファワード内に 4 つの 24bpp ピクセルをパック。 **注意**　パックモード（bLcd_FBP＝1）は、24bpp モード（bLcd_BPP＝3'b110）のときだけ使用できます。	R/W
b18	bLcd_LPS	LCD ポート選択 0：LCD ポート出力（デフォルト） 1：予約	R/W
b17、b16	bLcd_FDW	FIFO DMA 要求バーストサイズ DMA 読み出し要求のバーストサイズ設定をワード単位で行うことが可能です。 基本的なワードサイズは 64 ビットです。 2'b00：FIFO が 4 ワード以上の余裕がある場合の 4 ビートバーストに対する FIFO DMA 要求 2'b01：FIFO が 8 ワード以上の余裕がある場合の 8 ビートバーストに対する FIFO DMA 要求 2'b10：FIFO が 16 ワードの余裕がある場合の 16 ビートバーストに対する FIFO DMA 要求 2'b11：予約 帯域幅の都合上 2'b10 が推奨されます。	R/W
b15	bLcd_PSS	パレットロードソース選択 初期値のままとしてください。本コントローラは、パレットロードソース選択をサポートしていません。 0：パレットをスレーブバスからロード（パレットレジスタ経由の CPU 転送を使用） 1：（使用禁止）	R/W

表 8.3　rLcd_CR1 レジスタの内容（2/3）

ビット位置	ビット名	機能	R/W
b14～b12	bLcd_OPS	出力ピクセル選択 ビット[12]：bLcd_OPS0 0：16bpp 出力フォーマット RGB 5:6:5 または BGR 5:6:5 1：16bpp 出力フォーマット RGB 5:5:5 または BGR 5:5:5 ビット[13]：Lcd_OPS1。bLcd_BPP＝1、2、4、8、16、18bpp の場合有効 0：LCD パネルとの 3×8 ビット RGB インタフェースの LSB に RGB 5:6:5 または RGB 5:5:5 を指定 LCD_R,G,B[5:0] LCD_R,G,B[5:1] 1：LCD パネルとの 3×8 ビット RGB インタフェースの MSB に RGB 5:6:5 または RGB 5:5:5 を指定 LCD_R,G,B[7:2] LCD_R,G,B[7:3] ビット[14]：bLcd_OPS2：予約	R/W
b11	bLcd_VSP	垂直同期極性 0：LCD_VSYNC 信号はアクティブ High 1：LCD_VSYNC 信号はアクティブ Low	R/W
b10	bLcd_HSP	水平同期極性 0：LCD_HSYNC 信号はアクティブ High 1：LCD_HSYNC 信号はアクティブ Low	R/W
b9	bLcd_PCP	ピクセルクロック極性 0：LCD_PCLK の立ち上がりで駆動される出力データ信号 1：LCD_PCLK の立ち下がりで駆動される出力データ信号	R/W
b8	bLcd_DEP	データイネーブル極性 0：アクティブ表示モード時 LCD_DE 信号は、アクティブ Low 1：アクティブ表示モード時 LCD_DE 信号は、アクティブ High	R/W
b7	bLcd_EBO	パレットにおけるビッグエンディアンまたはリトルエンディアンバイト並び順モード 0：リトルエンディアン 1：ビッグエンディアン	R/W
b6	bLcd_EPO	バイト内ビッグエンディアンまたはリトルエンディアンピクセル並び順 バイト内のピクセルエンディアン並び順を選択（ピクセル 1、2、4bpp のみ） 0：リトルエンディアン 1：ビッグエンディアン	R/W
b5	bLcd_RGB	パレットの RGB または BGR フォーマットモードを選択 0：RGB モード。パレットからの赤と青のデータを入れ替えない 1：BGR モード。出力フォーマッタにおいて赤と青のデータを入れ替え	R/W
b4～b2	bLcd_BPP	LCD のピクセルあたりのビット数 3'b000：1bpp（パレットの 16 ビットエントリ×2 を使用） 3'b001：2bpp（パレットの 16 ビットエントリ×4 を使用） 3'b010：4bpp（パレットの 16 ビットエントリ×16 を使用） 3'b011：8bpp（パレットの 16 ビットエントリ×256 を使用） 3'b100：16bpp（パレットルックアップテーブルを使用しない） 3'b101：18bpp（パレットルックアップテーブルを使用しない） 3'b110：24bpp（パレットルックアップテーブルを使用しない） 3'b111：予約	R/W
b1	bLcd_LPE	LCD 電源イネーブル 出力 LCD_PE を直接駆動 通常 LCD パネルへの電源をイネーブルする際に使用されます。 0：LCD 電源ディスエーブル 1：LCD 電源イネーブル	R/W

表 8.3　rLcd_CR1 レジスタの内容（3/3）

ビット位置	ビット名	機能	R/W
b0	bLcd_LCE	LCD コントローライネーブル 0：LCD コントローラディスエーブル 1：LCD コントローライネーブル パネル信号はアクティブレベルに解放され、最初のフレームがフェッチされ、表示が開始されます。 **備考）** ● bLcd_LCE＝0 の場合、CPU は下記を初期化するものとします。 水平方向および垂直方向タイミング rLcd_HTR、rLcd_VTR1、rLcd_VTR2 レジスタ 拡張水平方向および垂直方向タイミング rLcd_HPPLOR および rLcd_HVTER レジスタ rLcd_PCTR レジスタのピクセルクロック RESET は非アクティブ（リセットなし状態）に解除 DMA コントローラレジスタ（rLcd_DBAR） パレットレジスタ ● bLcd_LCE＝0 の場合、LCD パネル信号 LCD_PCLK、LCD_HSYNC、LCD_VSYNC、LCD_DE、LCD_R[7:0]、LCD_G[7:0]、および LCD_B[7:0]は論理 0 に保持されます。 ● bLcd_LCE が 1 から 0 になると、LCD タイミングユニットは現在のフレーム表示が終わるのを待ち、パネル信号を強制的に論理 0 にします。 **注意）** bLcd_LCE はいったん 1 になると、表示動作中ずっと 1 のままとします。 bLcd_LCE は、パワーダウンの直前のみ Low にすることが可能です。 bLcd_LCE は、1 にした後、（コントローラを再構成するために）0 にクリアし、その後再び 1 にすることはできません。	R/W

8.4.2　rLcd_HTR ― 水平方向タイミングレジスタ

タイミングについては、「**図 8.2 LCD 水平方向タイミング**」および「**8.5.3 タイミング＆制御タイミング＆制御**」を参照してください。

アドレス　5300 4008h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bLcd_HSW								bLcd_HBP							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_PPL								bLcd_HFP							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.4　rLcd_HTR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b24	bLcd_HSW	水平同期幅 ● LCD_PCLK クロック周期単位の LCD_HSYNC 幅 ● 有効値：0～255 ● 目的の LCD_HSYNC 同期幅－1 を設定 **備考**　bLcd_HSW の 2 ビット拡張については、rLcd_HVTER レジスタの bLcd_HSWE（水平同期幅拡張）を参照してください。	R/W
b23～b16	bLcd_HBP	水平バックポーチ ● 水平同期幅の終わりから表示対象ラインの最初のピクセルまたは LCD_DE アクティブまで待機する LCD_PCLK クロック周期数 ● 有効値：0～255 **備考**　bLcd_HBP の 2 ビット拡張については、rLcd_HVTER レジスタの bLcd_HBPE（水平バックポーチ拡張）を参照してください。	R/W
b15～b8	bLcd_PPL	水平方向ラインあたりのピクセル数 ● ラインあたりのピクセル数 ● 実際のラインあたりのピクセル数＝16×bLcd_PPL ● ラインあたりの最大ピクセル数＝16×255＝4080 ● 有効値：1～255 rLcd_HPPLOR レジスタの bLcd_HPOE（水平方向ラインあたりピクセル数オーバーライド）が 1 になると、bLcd_HPPLO フィールドが bLcd_PLL フィールドをオーバーライドします。 bLcd_HPPLO は、16 で割り切れないラインあたりのピクセル数を持つパネルに使用されます。	R/W
b7～b0	bLcd_HFP	水平フロントポーチ ● LCD_DE アクティブ期間の終わりから LCD_HSYNC の始まりまでの、追加する LCD_PCLK クロック周期数 ● 有効値：0～255 **備考**　bLcd_HFP の 2 ビット拡張については、rLcd_HVTER レジスタの bLcd_HFPE（水平フロントポーチ拡張）を参照してください。	R/W

8.4.3　rLcd_VTR1 ― 垂直方向 1 タイミングレジスタ

タイミングについては、「**図 8.3 LCD 垂直方向タイミング**」および「**8.5.3 タイミング&制御**」を参照してください。

アドレス　5300 400Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	bLcd_VBP							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_VFP								bLcd_VSW							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.5　rLcd_VTR1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b24	予約ビット	読むと 0 が読み出されます。	R
b23～b16	bLcd_VBP	垂直バックポーチ ● 垂直同期幅の終わりから表示対象アクティブピクセルのある最初のラインの始まりまで待機する LCD_HSYNC ライン期間数 ● 有効値：0～255 **備考**　bLcd_VBP の 2 ビット拡張については、rLcd_HVTER レジスタの bLcd_VBPE（垂直バックポーチ拡張）を参照してください。	R/W
b15～b8	bLcd_VFP	垂直フロントポーチ ● フレームの終わりから LCD_VSYNC アクティブ期間までの LCD_HSYNC ライン期間数 ● 有効値：0～255 **備考**　bLcd_VFP の 2 ビット拡張については、rLcd_HVTER レジスタの bLcd_VFPE（垂直フロントポーチ拡張）を参照してください。	R/W
b7～b0	bLcd_VSW	垂直同期幅 ● LCD_HSYNC ライン期間中の LCD_VSYNC パルス幅 ● 有効値：0～255 **備考**　bLcd_VSW の 2 ビット拡張については、rLcd_HVTER レジスタの bLcd_VSWE（垂直同期幅拡張）を参照してください。	R/W

8.4.4　rLcd_VTR2 ― 垂直方向 2 タイミングレジスタ

タイミングについては、「**図 8.3 LCD 垂直方向タイミング**」および「**8.5.3 タイミング&制御**」を参照してください。

アドレス　5300 4010h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bLcd_LPP											
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.6　rLcd_VTR2 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b12	予約ビット	読むと 0 が読み出されます。	R
b11～b0	bLcd_LPP	パネルあたりのライン数 ● パネル内フレームあたりのアクティブライン数 ● 有効値：1～4095 ライン	R/W

8.4.5　rLcd_PCTR ― ピクセルクロックタイミングレジスタ

アドレス　5300 4014h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	bLcd_PCR	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.7　rLcd_PCTR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b11	予約ビット	読むと 0 が読み出されます。	R
b10	bLcd_PCR	ピクセルクロックドメインリセット 0：ピクセルクロック RESET はアクティブのまま（リセット状態） ピクセルクロックドメインの論理はすべてリセットに保持 1：ピクセルクロック RESET は非アクティブ（リセットなし状態）に解除 この場合、コンフィグレーションレジスタを設定後に LCD コントローラを起動することが可能です。 bLcd_PCR＝0 の場合、ピクセルクロックドメインの論理はすべてリセットに保持されます。ピクセルクロックドメインのリファレンスクロック LCD_ECLK は、システムコントローラにおいて LCD 解像度により必要とされる周波数で設定するものとします。クロックが設定され安定すると、ピクセルクロックドメインのリセットは、bLcd_PCR を 1 にして書き込むことによりデアサート可能です。	R/W
b9～b0	予約ビット	読むと 0 が読み出されます。	R

8.4.6　rLcd_ISR ― マスキング前の割り込みステータスレジスタ

本ビットに 1 を書き込むとクリアします。0 を書き込んでも何もしません。

アドレス　5300 4018h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	bLcd_LDD	bLcd_BAU	bLcd_VCT	bLcd_MBE	bLcd_FER	bLcd_IFO	bLcd_IFU	bLcd_OFO	bLcd_OFU
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.8　rLcd_ISR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b9	予約ビット	読むと 0 が読み出されます。	R
b8	bLcd_LDD	マスキング（AND 論理）前の LCD コントローラディスエーブル完了割り込み状態 rLcd_CR1 レジスタの bLcd_LCE が 1 から 0 になると（LCD コントローラアクティブから非アクティブモードへ）、LCD タイミングユニットは、LCD コントローラが非アクティブモードになり、パネルへの出力信号が強制的に論理 0 にされる前に、表示中の現在のフレームが終わるのを待ちます。LCD コントローラが非アクティブモードになると、bLcd_LDD は 1 になり割り込みが発生します（マスクされていない場合）。 0：LCD コントローラは未ディスエーブル 1：LCD コントローラはディスエーブルされ、（マスクされていない場合）LCDC_Int が発生	R/W
b7	bLcd_BAU	マスキング（AND 論理）前の bLcd_DCAR の DMA ベースアドレスフィールドによる更新割り込み状態 0：DMA ベースアドレスフィールド（bLcd_DBAR）はカレントアドレスフィールド（bLcd_DCAR）に転送されない 1：DMA ベースアドレスフィールド（bLcd_DBAR）はカレントアドレスフィールド（bLcd_DCAR）に転送され、（マスクされていない場合）LCDC_Int が発生	R/W
b6	bLcd_VCT	マスキング（AND 論理）前の垂直コンペアによる割り込み状態 0：垂直スキャンコンペア設定時トリガ未発生 1：垂直スキャンコンペア設定時トリガ発生し（マスクされていない場合）LCDC_Int が発生 垂直スキャンコンペア設定トリガについては、rLcd_ISCR レジスタの定義を参照してください。	R/W
b5	bLcd_MBE	マスキング（AND 論理）前の DMA マスタ AHB バスエラー割り込み状態 0：エラーなし 1：DMA マスタ AHB バスにエラーが発生し（マスクされていない場合）LCDC_Int が発生	R/W
b4	bLcd_FER	マスキング（AND 論理）前の入出力 FIFO エラー（アンダーラン、オーバーラン）割り込み状態 0：エラーなし 1：bLcd_OFU、bLcd_OFO、bLcd_IFU、bLcd_IFO エラービットのいずれかがアクティブ 1 になると（OR 論理）、bLcd_FER もセットされ、（マスクされていない場合）LCDC_Int が発生	R/W
b3	bLcd_IFO	マスキング（AND 論理）前の入力 FIFO オーバーラン割り込み状態 0：エラーなし 1：入力 FIFO オーバーランにより（マスクされていない場合）LCDC_Int が発生	R/W
b2	bLcd_IFU	マスキング前の入力 FIFO アンダーラン割り込み状態（AND 論理） 0：エラーなし 1：入力 FIFO アンダーランにより（マスクされていない場合）LCDC_Int が発生	R/W

表 8.8　　rLcd_ISR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b1	bLcd_OFO	マスキング（AND ロジック）前の出力 FIFO オーバーラン割り込み状態 0：エラーなし 1：出力 FIFO オーバーランにより（マスクされていない場合）LCDC_Int が発生	R/W
b0	bLcd_OFU	マスキング（AND 論理）前の出力 FIFO アンダーラン割り込み状態 0：エラーなし 1：出力 FIFO アンダーランにより（マスクされていない場合）LCDC_Int が発生	R/W

8.4.7　rLcd_IMR ― 割り込みマスクレジスタ

アドレス　5300 401Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	bLcd_LDDM	bLcd_BAUM	bLcd_VCTM	bLcd_MBEM	bLcd_FERM	bLcd_IFOM	bLcd_IFUM	bLcd_OFOM	bLcd_OFUM
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.9　rLcd_IMR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b9	予約ビット	読むと 0 が読み出されます。	R
b8	bLcd_LDDM	LCD コントローラディスエーブル完了、マスク有効／無効 0：LDD 割り込みを禁止 1：LDD 割り込みを許可（bLcd_LDD により駆動）	R/W
b7	bLcd_BAUM	bLcd_DCAR の DMA ベースアドレスフィールドによる更新、マスク有効／無効 0：BAU 割り込みを禁止 1：BAU 割り込みを許可（bLcd_BAU により駆動）	R/W
b6	bLcd_VCTM	垂直コンペアトリガ起動、マスク有効／無効 0：VCT 割り込みを禁止 1：VCT 割り込みを許可（bLcd_VCT により駆動）	R/W
b5	bLcd_MBEM	DMA マスタ AHB バスエラー、マスク有効／無効 0：MBE 割り込みを禁止 1：MBE 割り込みを許可（bLcd_MBE により駆動）	R/W
b4	bLcd_FERM	入出力 FIFO エラー、アンダーランまたはオーバーラン、マスク有効／無効 0：FER 割り込みを禁止 1：FER 割り込みを許可（bLcd_FER により駆動）	R/W
b3	bLcd_IFOM	入力 FIFO オーバーラン、マスク有効／無効 0：IFO 割り込みを禁止 1：IFO 割り込みを許可（bLcd_IFO により駆動）	R/W
b2	bLcd_IFUM	入力 FIFO アンダーラン、マスク有効／無効 0：IFU 割り込みを禁止 1：IFU 割り込みを許可（bLcd_IFU により駆動）	R/W
b1	bLcd_OFOM	出力 FIFO オーバーラン、マスク有効／無効 0：OFO 割り込みを禁止 1：OFO 割り込みを許可（bLcd_OFO により駆動）	R/W
b0	bLcd_OFUM	出力 FIFO アンダーラン、マスク有効／無効 0：OFU 割り込みを禁止 1：OFU 割り込みを許可（bLcd_OFU により駆動）	R/W

8.4.8　rLcd_IVR ― マスキング後の割り込みステータスレジスタ

アドレス　5300 4020h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	bLcd_LDDV	bLcd_BAUV	bLcd_VCTV	bLcd_MBEV	bLcd_FERV	bLcd_IFOV	bLcd_IFUV	bLcd_OFOV	bLcd_OFUV
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.10　rLcd_IVR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b9	予約ビット	読むと 0 が読み出されます。	R
b8	bLcd_LDDV	マスキング（AND 論理）後の LCD コントローラディスエーブル完了割り込み状態 rLcd_CR1 レジスタの bLcd_LCE が 1 から 0 になると（LCD コントローラアクティブから非アクティブモードへ）、表示中の現在のフレームが終わるのを待った後、LCD タイミングユニットは、LCD コントローラが非アクティブモードになり、パネルへの出力信号が強制的に論理 0 にされます。LCD コントローラが非アクティブモードになると、割り込みがマスクされていない場合、bLcd_LDDV は 1 になり割り込みが発生します。 0：LCD コントローラは未ディスエーブル 1：LCD コントローラはディスエーブルされ、LCDC_Int が発生	R
b7	bLcd_BAUV	マスキング（AND 論理）後の bLcd_DCAR の DMA ベースアドレスフィールドによる更新割り込み状態 0：DMA ベースアドレスフィールド（bLcd_DBAR）はカレントアドレスフィールド（bLcd_DCAR）に転送されない 1：DMA ベースアドレスフィールド（bLcd_DBAR）はカレントアドレスフィールド（bLcd_DCAR）に転送され、LCDC_Int が発生	R
b6	bLcd_VCTV	マスキング（AND 論理）後の垂直コンペアによる割り込み状態 0：垂直スキャンコンペア設定時トリガ未発生 1：垂直スキャンコンペア設定時トリガ発生し LCDC_Int が発生 垂直スキャンコンペア設定トリガについては、rLcd_ISCR レジスタの定義を参照してください。	R
b5	bLcd_MBEV	マスキング（AND 論理）後の DMA マスタ AHB バスエラー割り込み状態 0：エラーなし 1：DMA マスタ AHB バスにエラーが発生し LCDC_Int が発生	R
b4	bLcd_FERV	マスキング（AND ロジック）後の入出力 FIFO エラー（アンダーラン、オーバーラン）割り込み状態 0：エラーなし 1：bLcd_OFUV、bLcd_OFOV、bLcd_IFUV、bLcd_IFOV エラービットのいずれかがアクティブ 1 になると（OR 論理）、bLcd_FERV もセットされ、LCDC_Int が発生	R
b3	bLcd_IFOV	マスキング（AND 論理）後の入力 FIFO オーバーラン割り込み状態 0：エラーなし 1：入力 FIFO オーバーランにより LCDC_Int が発生	R
b2	bLcd_IFUV	マスキング（AND 論理）後の入力 FIFO アンダーラン割り込み状態 0：エラーなし 1：入力 FIFO アンダーランにより LCDC_Int が発生	R
b1	bLcd_OFOV	マスキング（AND 論理）後の出力 FIFO オーバーラン割り込み状態 0：エラーなし 1：出力 FIFO オーバーランにより LCDC_Int が発生	R

表 8.10　rLcd_IVR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b0	bLcd_OFUV	マスキング（AND 論理）後の出力 FIFO アンダーラン割り込み状態 0：エラーなし 1：出力 FIFO アンダーランにより LCDC_Int が発生	R

8.4.9　rLcd_ISCR ― 割り込みスキャンコンペアレジスタ

アドレス　5300 4024h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	bLcd_VSC		
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.11　rLcd_ISCR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b3	予約ビット	読むと 0 が読み出されます。	R
b2～b0	bLcd_VSC	垂直スキャンコンペア 3'b000：bLcd_VSC 非アクティブ 3'b100：垂直同期幅（bLcd_VSW）パルス開始 3'b101：垂直バックポーチ（bLcd_VBP）開始 3'b110：アクティブフレームウィンドウ開始 3'b111：垂直フロントポーチ（bLcd_VFP）開始 3'b001、3'b010、3'b011：予約	R/W

8.4.10　rLcd_DBAR ― フレームバッファメモリの DMA 開始ベースアドレス

アドレス　5300 4028h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bLcd_DBAR															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_DBAR													—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.12　rLcd_DBAR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b3	bLcd_DBAR	DMA ベースアドレス フレームバッファメモリの開始アドレス 本値は、各垂直フレーム時間の開始時 bLcd_DCAR に転送されます。 LCD コントローラが（bLcd_LCE を 0 にすることにより）ディスエーブルされるか、bLcd_DBAR が bLcd_DCAR レジスタに転送されたことを意味する bLcd_BAU が 1 になった直後に bLcd_DBAR は書き込まれるものとします。	R/W
b2～b0	予約ビット	読むと 0 が読み出されます。	R

8.4.11　rLcd_DCAR ― 進行中の DMA カレントベースアドレス

アドレス　5300 402Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bLcd_DCAR															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_DCAR															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.13　rLcd_DCAR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bLcd_DCAR	DMA カレントアドレス DMA コントローラがフレームバッファメモリからのデータにアクセス中のカレントアドレス。絶えず、入力 FIFO にロードする次の 64 ビットのフレームバッファワードを反映します。bLcd_DBAR は、各垂直フレーム時間の開始時 bLcd_DCAR に転送されます。	R

8.4.12　rLcd_DEAR ― DMA 終了アドレス

アドレス　5300 4030h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bLcd_DEAR															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_DEAR															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.14　rLcd_DEAR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bLcd_DEAR	DMA 終了アドレス フレームバッファメモリの終了アドレス。読み出し対象の最終フレームバッファメモリアドレスを判断する際にレジスタ bLcd_DCAR と比較されます。 **注意）** bLcd_DEAR 値は、転送対象のブロックの終わりを＋1＋8 にして設定する必要があります。 **［例］** ● ブロックの開始は、32’h5000_0000 ● ブロックの終わりは、32’h5000_FFFF ● ブロックサイズ：64KB (32’h1_0000) 設定は下記のようになります。 ● bLcd_DBAR：32’h5000_0000 ● bLcd_DEAR：32’h5001_0008 (+1+8)	R/W

8.4.13　rLcd_PWMFR_0 ― PWM0 周波数レジスタ

アドレス　5300 4034h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	bLcd_PWMFCE_0	bLcd_PWMFCD_0					
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_PWMFCD_0															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.15　rLcd_PWMFR_0 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b23	予約ビット	読むと 0 が読み出されます。	R
b22	bLcd_PWMFCE_0	PWM0 周波数クロックイネーブル 0：LCD_PWM[0]非アクティブ 1：LCD_PWM[0]アクティブ **備考）** ● 本 PWM0 ユニットの出力が BLINK 論理を駆動します。 ● bLcd_BlinkOn が“1”で、Lcd_GPIOR レジスタの bLcd_BlinkMode で管理される場合、本モードはアクティブです。	R/W
b21～b0	bLcd_PWMFCD_0	PWM0 周波数クロック（PWM_CLK）分周器 LCD_ECLK クロックより生成される内部 PWM0 ユニットの周波数 0：LCD_ECLK／（256×1） 1：LCD_ECLK／（256×2） 2：LCD_ECLK／（256×3） …　… **備考）** ● 本 PWM0 ユニットの出力が BLINK 論理を駆動します。 ● bLcd_BlinkOn が“1”で、Lcd_GPIOR レジスタの bLcd_BlinkMode で管理される場合、本モードはアクティブです。	R/W

8.4.14　rLcd_PWMDCR_0 ― PWM0 デューティサイクルレジスタ

アドレス　5300 4038h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bLcd_PWMDC_0							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.16　rLcd_PWMDCR_0 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bLcd_PWMDC_0	PWM0 デューティサイクルレジスタ LCD_PWM[0]出力のデューティサイクル 0：1／256 1：2／256 2：3／256 …　… **備考）** ● 本 PWM0 ユニットの出力が BLINK 論理を駆動します。 ● bLcd_BlinkOn が“1”で、Lcd_GPIOR レジスタの bLcd_BlinkMode で管理される場合、本モードはアクティブです。	R/W

8.4.15　rLcd_HVTER ― 水平方向および垂直方向タイミング拡張レジスタ

タイミングについては、「**図 8.3 LCD 垂直方向タイミング**」、「**図 8.2 LCD 水平方向タイミング**」、および「**8.5.3 タイミング&制御**」を参照してください。

アドレス　5300 4044h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	bLcd_VSWE		―	―	―	―	―	―	bLcd_HSWE	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bLcd_VBPE		―	―	bLcd_VFPE		―	―	bLcd_HBPE		―	―	bLcd_HFPE	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.17　rLcd_HVTER レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b26	予約ビット	読むと 0 が読み出されます。	R
b25、b24	bLcd_VSWE	垂直同期幅拡張 rLcd_VTR1 レジスタの bLcd_VSW（垂直同期幅）に最上位の 2 ビットとして付加され、完全な垂直同期幅値を定義する 10 ビットフィールドを形成します。	R/W
b23～b18	予約ビット	読むと 0 が読み出されます。	R
b17、b16	bLcd_HSWE	水平同期幅拡張 rLcd_HTR レジスタの bLcd_HSW（水平同期幅）に最上位の 2 ビットとして付加され、完全な水平同期幅値を定義する 10 ビットフィールドを形成します。	R/W
b15、b14	予約ビット	読むと 0 が読み出されます。	R
b13、b12	bLcd_VBPE	垂直バックポーチ拡張 rLcd_VTR1 レジスタの bLcd_VBP（垂直バックポーチ）に最上位の 2 ビットとして付加され、完全な垂直バックポーチ値を定義する 10 ビットフィールドを形成します。	R/W
b11、b10	予約ビット	読むと 0 が読み出されます。	R
b9、b8	bLcd_VFPE	垂直フロントポーチ拡張 rLcd_VTR1 レジスタの bLcd_VFP（垂直フロントポーチ）に最上位の 2 ビットとして付加され、完全な垂直フロントポーチ値を定義する 10 ビットフィールドを形成します。	R/W
b7、b6	予約ビット	読むと 0 が読み出されます。	R
b5、b4	bLcd_HBPE	水平バックポーチ拡張 rLcd_HTR レジスタの bLcd_HBP（水平バックポーチ）に最上位の 2 ビットとして付加され、完全な水平バックポーチ値を定義する 10 ビットフィールドを形成します。	R/W
b3、b2	予約ビット	読むと 0 が読み出されます。	R
b1、b0	bLcd_HFPE	水平フロントポーチ拡張 rLcd_HTR レジスタの bLcd_HFP（水平フロントポーチ）に最上位の 2 ビットとして付加され、完全な水平フロントポーチ値を定義する 10 ビットフィールドを形成します。	R/W

8.4.16　rLcd_HPPLOR ― 水平方向ラインあたりピクセル数オーバーライド制御

タイミングについては、「**図 8.2 LCD 水平方向タイミング**」および「**8.5.3 タイミング&制御**」を参照してください。

アドレス　5300 4048h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bLcd_HPOE	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bLcd_HPPLO											
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.18　rLcd_HPPLOR レジスタの内容

ビット位置	ビット名	機能	R/W
b31	bLcd_HPOE	水平方向ラインあたりピクセル数オーバーライド許可 0：水平方向ラインあたりピクセル数オーバーライド禁止 rLcd_HTR レジスタの bLcd_PPL フィールドが使用されます。 ラインあたりのピクセル数が 16 で割り切れるパネルに使用されます。 1：水平方向ラインあたりピクセル数オーバーライド許可 rLcd_HPPLOR の bLcd_HPPLO フィールドは bLcd_PPL フィールドをオーバーライドします。 ラインあたりのピクセル数が 16 で割り切れないパネルに使用されます。	R/W
b30～b12	予約ビット	読むと 0 が読み出されます。	R
b11～b0	bLcd_HPPLO	水平方向ラインあたりピクセル数オーバーライド ラインあたりの実際のピクセル数 bLcd_HPOE が 1 の場合の rLcd_HTR レジスタの bLcd_PPL フィールドをオーバーライドします。 ラインあたりのピクセル数が 16 で割り切れないパネルの場合、bLcd_HPOE を 1 に、bLcd_HPPLO をラインあたりの正確なピクセル数に設定してください。 したがって、800×600 パネルの場合、bLcd_HPPLO を 800 に設定してご使用ください。 ラインあたりのピクセル数有効値：1～4095	R/W

8.4.17　rLcd_PWMFR_1 ― PWM1 周波数レジスタ

アドレス　5300 404Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	bLcd_PWMFCE_1	bLcd_PWMFCD_1					
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_PWMFCD_1															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.19　rLcd_PWMFR_1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b23	予約ビット	読むと 0 が読み出されます。	R
b22	bLcd_PWMFCE_1	PWM1 周波数クロックイネーブル 0：LCD_PWM[1]非アクティブ 1：LCD_PWM[1]アクティブ	R/W
b21～b0	bLcd_PWMFCD_1	PWM1 周波数クロック（PWM_CLK）分周器 LCD_ECLK クロックより生成される内部 PWM1 ユニットの周波数 0：LCD_ECLK／（256×1） 1：LCD_ECLK／（256×2） 2：LCD_ECLK／（256×3） …　…	R/W

8.4.18　rLcd_PWMDCR_1 ― PWM1 デューティサイクルレジスタ

アドレス　5300 4050h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	bLcd_PWMDC_1							
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.20　rLcd_PWMDCR_1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b8	予約ビット	読むと 0 が読み出されます。	R
b7～b0	bLcd_PWMDC_1	PWM1 デューティサイクルレジスタ LCD_PWM[1]出力のデューティサイクル 0：1／256 1：2／256 2：3／256 …　…	R/W

8.4.19　rLcd_GPIOR ― 点滅コントロール

アドレス　5300 41F8h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bLcd_BlinkMode	bLcd_BlinkOn
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.21　rLcd_GPIOR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bLcd_BlinkMode	点滅の輝度モード 1：半輝度点滅、オン／オフ間切り替え [ON] LCD_R[7:0]＝{R7,R6,R5,R4,R3,R2,R1,R1}、R1 値は、LCD_R[1:0]に複製される。 LCD_G[7:0]＝{G7,G6,G5,G4,G3,G2,G1,G0} LCD_B[7:0]＝{B7,B6,B5,B4,B3,B2,B1,B1}、B1 値は、LCD_B[1:0]に複製される。 [OFF] LCD_R[7:0]＝{0,R7,R6,R5,R4,R3,R2,R1} LCD_G[7:0]＝{0,G7,G6,G5,G4,G3,G2,G1} LCD_B[7:0]＝{0,B7,B6,B5,B4,B3,B2,B1} 0：暗点滅、オン／オフ間切り替え [ON] LCD_R[7:0]＝{R7,R6,R5,R4,R3,R2,R1,R1}、R1 値は、LCD_R[1:0]に複製される。 LCD_G[7:0]＝{G7,G6,G5,G4,G3,G2,G1,G0} LCD_B[7:0]＝{B7,B6,B5,B4,B3,B2,B1,B1}、B1 値は、LCD_B[1:0]に複製される。 [OFF] LCD_R[7:0]＝8’b00000000 LCD_G[7:0]＝8’b00000000 LCD_B[7:0]＝8’b00000000 bLcd_BlinkOn が 1 のとき点滅モードはアクティブです。 点滅頻度は、rLcd_PWMFR_0 レジスタの bLcd_PWMFCD_0 および bLcd_PWMFCE_0 によります。	R/W
b0	bLcd_BlinkOn	点滅機能有効／無効 0：点滅モード無効 1：点滅モード有効	R/W

8.4.20　rLcd_CIR ― コア識別レジスタ

アドレス　5300 41FCh

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	bLcd_MN							
リセット後の値	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_BW				bLcd_BI				bLcd_REV							
リセット後の値	0	1	0	0	0	0	0	1	0	0	0	0	1	1	1	1

表 8.22　rLcd_CIR レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b24	予約ビット	読むと 0 が読み出されます。	R
b23～b16	bLcd_MN	コア識別レジスタ ― モデル番号 8'h90：モデル番号	R
b15～b12	bLcd_BW	コア識別レジスタ ― DMA マスタバスのビット幅 4'h4：64 ビット	R
b11～b8	bLcd_BI	コア識別レジスタ ― DMA マスタバスインタフェース 4'h1：AHB_BUS	R
b7～b0	bLcd_REV	コア識別レジスタ ― レビジョン 8'h0F：バージョン 1.15	R

8.4.21　コーディングパレット（パレットレジスタ）の説明

注　意

- コーディングパレットは、選択されたカラーモードに依存します。
- コーディングパレットのアドレスにマッピングされたシンボルは、各モードで異なります。
- また、各フィールドの意味（機能）は、カラーモードに依存します。
- 下表にすべてのカラーモードに関する情報を示します。

「**図 8.7 LCD 出力フォーマット指定 ― bLcd_BPP：1、2、4、8**」を参照してください。
「**図 8.8 LCD 出力フォーマット指定 ― bLcd_BPP：16、18、24**」を参照してください。
「**8.5.8 パレットルックアップテーブル**」を参照してください。
「**8.5.9 出力 FIFO およびフォーマッタ**」を参照してください。

8.4.21.1　rLcd_PAL_RGB_555 ― RGB 5:5:5 モード時のコーディングパレット

コーディングパレット設定：rLcd_CR1 レジスタにおいて、bLcd_RGB＝0、bLcd_OPS0＝1

アドレス　5300 4200h ～ 5300 43FCh

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	bLcd_RED1					bLcd_GREEN1					bLcd_BLUE1				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	bLcd_RED0					bLcd_GREEN0					bLcd_BLUE0				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.23　rLcd_PAL_RGB_555 レジスタの内容

ビット位置	ビット名	機能	R/W
b31	予約ビット	読むと 0 が読み出されます。	R
b30～b26	bLcd_RED1	赤データパレット：5 ビット	R/W
b25～b21	bLcd_GREEN1	緑データパレット：5 ビット	R/W
b20～b16	bLcd_BLUE1	青データパレット：5 ビット	R/W
b15	予約ビット	読むと 0 が読み出されます。	R
b14～b10	bLcd_RED0	赤データパレット：5 ビット	R/W
b9～b5	bLcd_GREEN0	緑データパレット：5 ビット	R/W
b4～b0	bLcd_BLUE0	青データパレット：5 ビット	R/W

8.4.21.2　rLcd_PAL_RGB_565 ― RGB 5:6:5 モード時のコーディングパレット

コーディングパレット設定：rLcd_CR1 レジスタにおいて、bLcd_RGB＝0、bLcd_OPS0＝0

アドレス　5300 4200h ～ 5300 43FCh

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bLcd_RED1					bLcd_GREEN1						bLcd_BLUE1				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_RED0					bLcd_GREEN0						bLcd_BLUE0				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.24　rLcd_PAL_RGB_565 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b27	bLcd_RED1	赤データパレット：5 ビット	R/W
b26～b21	bLcd_GREEN1	緑データパレット：6 ビット	R/W
b20～b16	bLcd_BLUE1	青データパレット：5 ビット	R/W
b15～b11	bLcd_RED0	赤データパレット：5 ビット	R/W
b10～b5	bLcd_GREEN0	緑データパレット：6 ビット	R/W
b4～b0	bLcd_BLUE0	青データパレット：5 ビット	R/W

8.4.21.3　rLcd_PAL_BGR_555 ― BGR 5:5:5 モード時のコーディングパレット

コーディングパレット設定：rLcd_CR1 レジスタにおいて、bLcd_RGB＝1、bLcd_OPS0＝1

アドレス　5300 4200h ～ 5300 43FCh

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	bLcd_BLUE1					bLcd_GREEN1					bLcd_RED1				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	bLcd_BLUE0					bLcd_GREEN0					bLcd_RED0				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.25　rLcd_PAL_BGR_555 レジスタの内容

ビット位置	ビット名	機能	R/W
b31	予約ビット	読むと 0 が読み出されます。	R
b30～b26	bLcd_BLUE1	青データパレット：5 ビット	R/W
b25～b21	bLcd_GREEN1	緑データパレット：5 ビット	R/W
b20～b16	bLcd_RED1	赤データパレット：5 ビット	R/W
b15	予約ビット	読むと 0 が読み出されます。	R
b14～b10	bLcd_BLUE0	青データパレット：5 ビット	R/W
b9～b5	bLcd_GREEN0	緑データパレット：5 ビット	R/W
b4～b0	bLcd_RED0	赤データパレット：5 ビット	R/W

8.4.21.4　rLcd_PAL_BGR_565 ― BGR 5:6:5 モード時のコーディングパレット

コーディングパレット設定：rLcd_CR1 レジスタにおいて、bLcd_RGB＝1、bLcd_OPS0＝0

アドレス　5300 4200h ～ 5300 43FCh

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bLcd_BLUE1					bLcd_GREEN1						bLcd_RED1				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bLcd_BLUE0					bLcd_GREEN0						bLcd_RED0				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 8.26　rLcd_PAL_BGR_565 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b27	bLcd_BLUE1	青データパレット：5 ビット	R/W
b26～b21	bLcd_GREEN1	緑データパレット：6 ビット	R/W
b20～b16	bLcd_RED1	赤データパレット：5 ビット	R/W
b15～b11	bLcd_BLUE0	青データパレット：5 ビット	R/W
b10～b5	bLcd_GREEN0	緑データパレット：6 ビット	R/W
b4～b0	bLcd_RED0	赤データパレット：5 ビット	R/W

8.5　動作説明

8.5.1　主要機能の説明

LCD コントローラ概要図に LCD コントローラのアーキテクチャを示します。

LCD コントローラは、まず、スレーブバスインタフェースを介してプロセッサにより初期化されます。本インタフェースは、LCD コントローラが「応答のみ可能で、バストランザクションを開始しない」読み出し／書き込みインタフェースです。

コントロールレジスタおよびステータスレジスタについては最低限下記を設定してください。

- 水平方向および垂直方向タイミング信号用のタイミングレジスタ（rLcd_HTR、rLcd_VTR1、および rLcd_VTR2）
- DMA ベースアドレスレジスタ（rLcd_DBAR）
- ピクセルクロックタイミングレジスタ（rLcd_PCTR）

その後、コントロールレジスタ（rLcd_CR1）の bLcd_LCE をセットすると、LCD コントローラが動作を開始し、フレームバッファメモリにアクセスし、データ処理後ディスプレイに送信します。

パレットが使用される場合は、まずそのパレットをロードする必要があります。パレットは、スレーブインタフェースを介してプロセッサにより静的にロードします。

各フレームの開始は、垂直同期信号と同時の、タイミング&コントロールユニットからの内部開始同期信号により開始します。本開始同期信号は DMA コントローラを起動し、DMA マスタインタフェースを介してフレームバッファメモリデータへのアクセスを開始します。本開始同期信号はまたピクセルアンパックを起動し、入力 FIFO の読み出し側からのデータ受け入れを開始します。

マスタインタフェースは、マスタ AHB バスとの読み出しトランザクションを起動します。バス使用率を上げるために、（64 ビットワードで）4、8、16 ワードのバースト読み出し長を選択設定できます。受信したフレームデータは入力 FIFO に書き込まれます。FIFO は 2 つのクロックドメインのインタフェースとなります。すなわち、LCD パネルは LCD_ECLK（公差を含む）で動作しており、一方プロセッサシステムバスは LCD_HCLK で動作しています。

ピクセルアンパックは、64 ビットフレームバッファワードから 1、2、4、8、16、18、または 24bpp（bit per pixel）ワードをアンパックします。bpp 設定およびパレットが使用されているかどうかにより、ピクセルデータは、（カラールックアップテーブル経由で変換された）パレット経由またはピクセルアンパックから直接出力 FIFO に送られます。

出力 FIFO キューは、LCD パネルタイミング信号との同期用にピクセルを準備します。FIFO の出力は、出力フォーマッタを駆動し、出力フォーマッタは 18/24 ビットパネルインタフェース用に RGB データをフォーマットします。

各項で、ピクセルデータフローの各ブロック処理の詳細情報を記述します。

8.5.2　帯域制限

サポートされる LCD パネルに関する主な制限は、利用可能なメモリ帯域です。

LCD コントローラは、フレームバッファメモリサイズおよびシステム相互接続帯域要件を決定する、ユーザ設定可能な LCD パネル解像度およびエンコードされたピクセルサイズを各種サポートします。LCD フレームバッファメモリサイズの表に、各 LCD パネル解像度対フレームバッファ bpp 選択、および必要なメモリサイズの一覧を示します。LCD 帯域の表には、対応する必要なフレームバッファ帯域を示します。

表 8.27　LCD 帯域

	QVGA	WQVGA	VGA	WVGA	SVGA	WSVGA	XGA
水平方向	320	480	640	800	800	1024	1024
垂直方向	240	272	480	480	600	600	768
ピクセルあたりのビット数	帯域（MB／秒、60Hz 時）						
1	0.6	1.0	2.3	2.9	3.6	4.6	5.9
2	1.2	2.0	4.6	5.8	7.2	9.2	11.8
4	2.3	3.9	9.2	11.5	14.4	18.4	23.6
8	4.6	7.8	18.4	23.0	28.8	36.8	47.2
16	9.2	15.7	36.9	46.1	57.6	73.7	94.4 注 1
24（パック）	13.8	23.5	55.3	69.1	86.4 注 1	110.5 注 1	141.6 注 1、注 2
18 もしくは 24（パックなし）	18.4	31.3	73.7	92.2 注 1	115.2 注 1	147.4 注 1、注 2	188.7 注 1、注 2

注1.　4 バースト転送時にアンダーフローが発生する可能性があります。

注2.　8 バースト転送時にアンダーフローが発生する可能性があります。

表 8.28　LCD フレームバッファメモリサイズ

	QVGA	WQVGA	VGA	WVGA	SVGA	WSVGA	XGA
水平方向	320	480	640	800	800	1024	1024
垂直方向	240	272	480	480	600	600	768
ピクセルあたりのビット数	フレームバッファメモリ要件（kiB）						
1	9.4	15.9	37.5	46.9	58.6	75.0	96.0
2	18.8	31.9	75.0	93.8	117.2	150.0	192.0
4	37.5	63.8	150.0	187.5	234.4	300.0	384.0
8	75.0	127.5	300.0	375.0	468.8	600.0	768.0
16	150.0	255.0	600.0	750.0	937.5	1200.0	1536.0
24（パック）	225.0	382.5	900.0	1125.0	1406.3	1800.0	2304.0
18 もしくは 24（パックなし）	300.0	510.0	1200.0	1500.0	1875.0	2400.0	3072.0

8.5.3　タイミング＆制御

タイミング＆制御ユニットは、水平方向タイミングレジスタ（rLcd_HTR）および水平方向タイミングレジスタ 1 & 2（rLcd_VTR1、rLcd_VTR2）、そしてオプションで水平方向および垂直方向タイミング拡張レジスタ（rLcd_HVTER）および水平方向タイミングオーバーライドレジスタ（rLcd_HPPLOR）を利用し、タイミング信号 LCD_VSYNC、LCD_HSYNC、および LCD_DE を生成し、LCD パネルに送ります。タイミングユニットは、コントロールレジスタ 1（rLcd_CR1）の bLcd_LCE がアクティブになるまで非アクティブのままです。この時点で、タイミング&制御ユニットは、bLcd_LCE がデアサートされるまで動作します。そのときタイミングユニットは、現フレームの終わりまで動作を継続し、現フレームが終わりになるとシャットダウンします。タイミングユニットは、bLcd_LCE を再アサートすることにより再起動可能ですが、ディスプレイへの電源供給はしばしば初期状態に戻す必要があります。これは、LCD パネルの外部電源のイネーブル用に接続されたコントロールレジスタ 1（rLcd_CR1）の bLcd_LPE により可能です。

bLcd_LCE は、電源シーケンスでも役割を果たします。立ち上げ時、bLcd_LCE が非アクティブの間、タイミング信号の LCD_PCLK、LCD_HSYNC、LCD_VSYNC、LCD_DE、およびデータ信号の LCD_R[7:0]、LCD_G[7:0]、LCD_B[7:0]は論理 0 に保持されます。bLcd_LCE シャットダウン時、表示中の現在のフレームが完了しタイミングユニットが停止した後、これらの信号は強制的に論理 0 にされます。この場合、LCD パネルの電源は安全に落とすことができます。

タイミングユニットは、割り込み bLcd_VCT を発生します。この割り込みは、垂直スキャン期間中 4 つのタイミングトリガポイントのいずれかで発生します。トリガポイントは、割り込みスキャンコンペアレジスタ（rLcd_ISCR）を介して設定されます。

下図に、LCD のさまざまなパラメータ定義を示します。

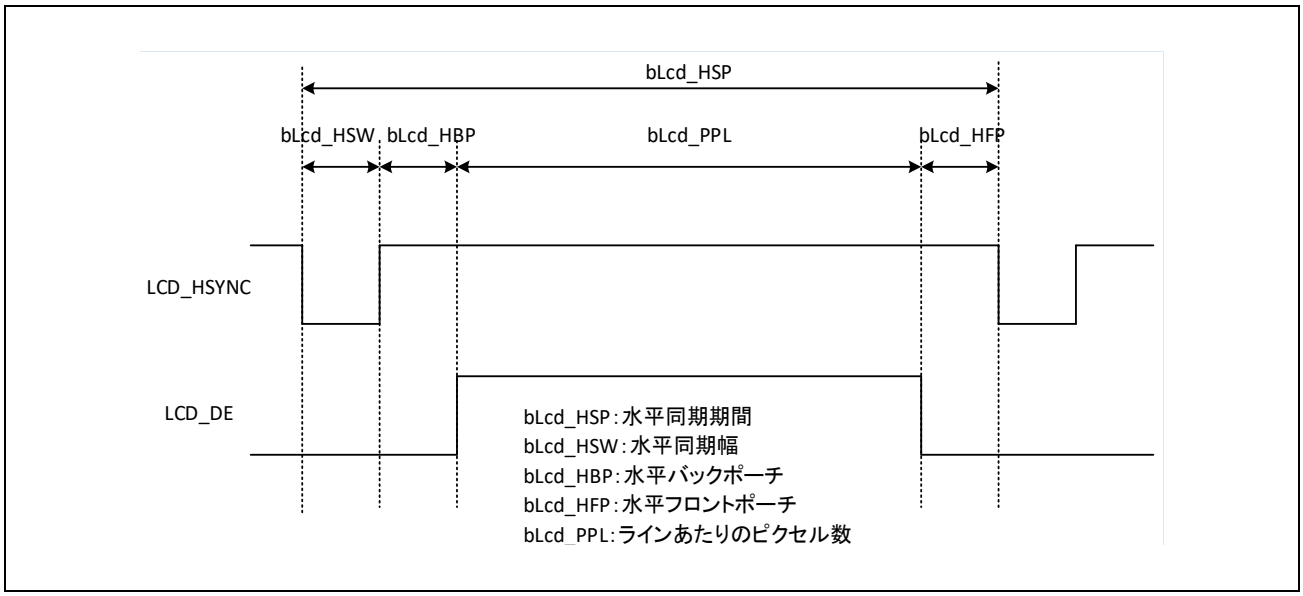


図 8.2　　LCD 水平方向タイミング

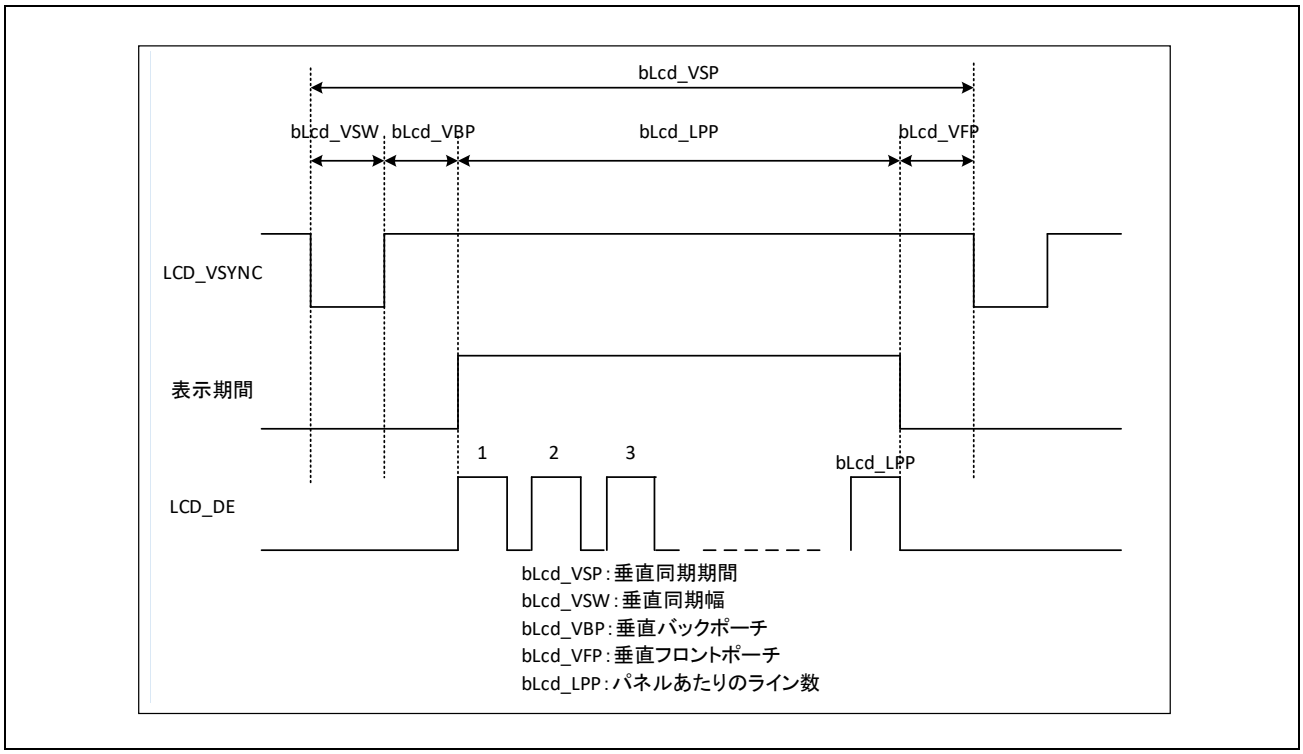


図 8.3　　LCD 垂直方向タイミング

8.5.4　DMA コントローラとメモリインタフェース

DMA コントローラは、DMA ベースアドレスレジスタ（rLcd_DBAR）から DMA カレントアドレスレジスタ（rLcd_DCAR）への転送および最初のメモリ転送トランザクションの開始により、内部フレーム開始パルスを介して初期化されます。バーストに含まれるワード数は、コントロールレジスタ 1（rLcd_CR1）の bLcd_FDW により設定されます。bLcd_FDW および FIFO 内のエンプティワード数に基づき、DMA コントローラからマスタインタフェースへサービス要求が発行されると、フレームバッファ読み出しが開始されます。

ソフトウェアは、rLcd_DEAR レジスタに、フレームバッファ終了アドレスを設定する必要があります。DMA コントローラは、rLcd_DCAR のカレントアドレスが rLcd_DEAR と同じになるまでフレームバッファワードを読み続けます。DMA コントローラは次のフレームで停止し、rLcd_DBAR 内のアドレスで始まるフレームバッファから読み出しを開始します。

8.5.5　フレームバッファの構成

フレームバッファメモリは外部メモリ内に存在します（DDR または組み込み型 SRAM）。DMA マスタインタフェースに結び付いているフレームバッファは、LCD パネルに表示するためのエンコードされたもしくはエンコードなしのピクセルを提供します。

注　意

- DMA マスタインタフェース接続は、64 ビットのデータバス幅があります。
- **「図 8.1 LCD コントローラの概要」**を参照してください。

8.5.6　入力 FIFO

入力 FIFO のサイズは、深さ 1K ワード×幅 64 ビットのメモリです。**「図 8.1 LCD コントローラの概要」**を参照してください。

DMA コントローラとマスタインタフェースは、バスクロック（LCD_HCLK）ドメインの書き込み側を制御し、一方ピクセルアンパックは、ピクセルクロック（LCD_ECLK）ドメインの読み出し側を制御します。アドレスポインタが、FIFO エンプティおよびフルフラグ計算に使用され、同様に空きワード数が 4 つ、8 つ、または 16 ある場合に使用されます。

コントロールレジスタ 1（rLcd_CR1）の bLcd_FDW および FIFO の空きワード数に基づき、メモリに対する 4、8、または 16 ワードバーストサービス要求が DMA コントローラからマスタインタフェースに発行されます。

空きがない状態での FIFO 書き込み、または有効なデータがない状態での読み出しがあった場合には、割り込み bLcd_IFO（入力 FIFO – オーバーラン）および bLcd_IFU（入力 FIFO – アンダーラン）が発生します。設計上、書き込み側は FIFO をオーバーランできません。読み出し側アンパック論理はアンダーランを引き起こす場合がありますが、これはマスタバス帯域の不足あるいはフレームバッファメモリ応答に起因するものです。その結果、アンパック論理がデータを要求した場合、入力 FIFO はエンプティになります。

マスタバス帯域の不足あるいはフレームバッファメモリ応答に起因する FIFO アンダーランの最初の表示は、bLcd_OFU（出力 FIFO – アンダーラン）です。

8.5.7　ピクセルアンパック

ピクセルアンパックは、入力 FIFO から 32 ビットデータを読み出し、コントロールレジスタ 1（rLcd_CR1）の bLcd_BPP に従い 1、2、4、8、16、18、または 24bpp データを抽出します。1、2、4、8bpp はエントリをパレットにインデックスするエンコードされたピクセルで、一方 16、18、24bpp は、出力フォーマッタ経由で LCD パネルを直接駆動するエンコードされていないピクセルであるということに注意してください。LCD コントローラは、ビッグエンディアン、リトルエンディアンの両データフォーマットをサポートします。

各フレームで、タイミング＆制御ユニットからの内部開始同期パルスによりピクセルアンパックが初期化され、各ワードが読み出し側に現れると入力 FIFO のキューからそれらのワードを取り出し始めます。

下表に、エンディアンと bLcd_BPP 設定の組み合わせに応じた入力 FIFO 内の各フレームバッファワードデータの構成を示します。3 つのサポートされているデータフォーマットのおのおのについて、ピクセルアンパックは、データワードから適切な表示ピクセルを抽出します。

以下に、その 3 つのデータ種別を示します。

- LEB_LEP：リトルエンディアンピクセルバイトに置かれたリトルエンディアンフレームバッファバイト
  - bLcd_EBO＝0 かつ bLcd_EPO＝0
- BEB_BEP：ビッグエンディアンピクセルバイトに置かれたビッグエンディアンフレームバッファバイト
  - bLcd_EBO＝1 かつ bLcd_EPO＝x
  - 1、2、4、8、16bpp の場合のみ違いがあります。
- LEB_BEP：ビッグエンディアンピクセルバイトに置かれたリトルエンディアンフレームバッファバイト
  - bLcd_EBO＝0 かつ bLcd_EPO＝1
  - 1、2、4bpp の場合のみ違いがあります。

LCDピクセルアンパックおよびデータ構造
リトルエンディアンピクセルバイト(LEP)に置かれたリトルエンディアンフレームバッファバイト(LEB)
bLcd_EBO=0, bLcd_EPO=0

BPP	入力FIFOリード側出力ビット															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	p31	p30	p29	p28	p27	p26	p25	p24	p23	p22	p21	p20	p19	p18	p17	p16
2	p15		p14		p13		p12		p11		p10		p9		p8	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	p7				p6				p5				p4			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	p3								p2							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
16	p1															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
18	p0															
	-	-	-	-	-	-	-	-	-	-	-	-	-	-	17	16
24	p0															
	-	-	-	-	-	-	-	-	23	22	21	20	19	18	17	16

BPP	入力FIFOリード側出力ビット															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	p15	p14	p13	p12	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0
2	p7		p6		p5		p4		p3		p2		p1		p0	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	p3				p2				p1				p0			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	p1								p0							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
16	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
18	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

図 8.4　　ピクセルアンパック ― リトルエンディアンピクセルバイトに置かれたリトルエンディアンフレームバッファバイト

LCDピクセルアンパックおよびデータ構造
ビッグエンディアンピクセルバイト(BEP)に置かれたビッグエンディアンフレームバッファバイト(BEB)
bLcd_EBO=1, bLcd_EPO=x

BPP	入力FIFOリード側出力ビット															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	p0	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15
2	p0		p1		p2		p3		p4		p5		p6		p7	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	p0				p1				p2				p3			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	p0								p1							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
16	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
18	p0															
	-	-	-	-	-	-	-	-	-	-	-	-	-	-	17	16
24	p0															
	-	-	-	-	-	-	-	-	23	22	21	20	19	18	17	16

BPP	入力FIFOリード側出力ビット															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25	p26	p27	p28	p29	p30	p31
2	p8		p9		p10		p11		p12		p13		p14		p15	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	p4				p5				p6				p7			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	p2								p3							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
16	p1															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
18	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

図 8.5　　ピクセルアンパック ― ビッグエンディアンピクセルバイトに置かれたビッグエンディアンフレームバッファバイト

LCDピクセルアンパックおよびデータ構造
ビッグエンディアンピクセルバイト(BEP)に置かれたリトルエンディアンフレームバッファバイト(LEB)
bLcd_EBO=0, bLcd_EPO=1

BPP	入力FIFOリード側出力ビット															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	p24	p25	p26	p27	p28	p29	p30	p31	p16	p17	p18	p19	p20	p21	p22	p23
2	p12		p13		p14		p15		p8		p9		p10		p11	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	p6				p7				p4				p5			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	p3								p2							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
16	p1															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
18	p0															
	-	-	-	-	-	-	-	-	-	-	-	-	-	-	17	16
24	p0															
	-	-	-	-	-	-	-	-	23	22	21	20	19	18	17	16

BPP	入力FIFOリード側出力ビット															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	p8	p9	p10	p11	p12	p13	p14	p15	p0	p1	p2	p3	p4	p5	p6	p7
2	p4		p5		p6		p7		p0		p1		p2		p3	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	p2				p3				p0				p1			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	p1								p0							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
16	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
18	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

図 8.6　　ピクセルアンパック ― ビッグエンディアンピクセルバイトに置かれたリトルエンディアンフレームバッファバイト

8.5.8　パレットルックアップテーブル

パレットは、256 エントリ×16 ビットのルックアップテーブルで、2 ポートの 128 エントリ×32 ビット RAM として実装されます。1 つのポートがバスクロック（LCD_HCLK）ドメインと結びついています。スレーブインタフェースがプロセッサを介してパレットにデータを埋めます。2 番目のポートは、ピクセルクロック（LCD_ECLK）ドメインに結びついており、パレット RAM の内容をピクセルアンパックのエンコードされたピクセル出力によってインデックスすることが可能です。パレットの出力は出力フォーマッタに流し込まれます。

注　意

- DMA マスタインタフェース接続は、64 ビットのデータバス幅があります。
- **「図 8.1 LCD コントローラの概要」**を参照してください。

32 ビットパレットエントリのうち、どちらの 16 ビットを選択するかは、エンディアンの設定とインデックス用のエンコードされたピクセル入力の最下位ビットによって決定されます。コントロールレジスタ 1（rLcd_CR1）の bLcd_LPE によりエンディアンの設定が決定されます。

- リトルエンディアンモードで入力インデックスのエンコードされたピクセル最下位ビットが 0 の場合、下位 16 ビットのパレットエントリを選択
  - 「**図 8.6 ピクセルアンパック ― ビッグエンディアンピクセルバイトに置かれたリトルエンディアンフレームバッファバイト**」を参照してください。
- ビッグエンディアンモードで入力インデックスのエンコードされたピクセル最下位ビットが 0 の場合、上位 16 ビットのパレットエントリを選択
  - 「**図 8.5 ピクセルアンパック ― ビッグエンディアンピクセルバイトに置かれたビッグエンディアンフレームバッファバイト**」を参照してください。

パレットデータワード（rLcd_PAL_BGR_555、rLcd_PAL_BGR_565、rLcd_PAL_RGB_555、rLcd_PAL_RGB_565 の各レジスタ）の内部フォーマットを「**8.4.21 コーディングパレット（パレットレジスタ）の説明**」に示します。4 つのフォーマットがあり、それらは下表「bLcd_OPS および bLcd_RGB による LCD パレットデータワード解釈」により解釈されます。

bLcd_OPS および bLcd_RGB は、コントロールレジスタ 1（rLcd_CR1）にあります。

- bLcd_OPS[0]は、5 ビットまたは 6 ビットの緑ビット選択を制御します。
- bLcd_RGB は、赤または青フィールドの RGB または BGR の入れ替えを制御します。

表 8.29　bLcd_OPS および bLcd_RGB による LCD パレットデータワード解釈

bLcd_OPS[0]、bLcd_RGB	RGB フォーマット	赤／青の入れ替え／入れ替えなし
2'b00	RGB 5:6:5	入れ替えなし
2'b01	BGR 5:6:5	入れ替え
2'b10	RGB 5:5:5	入れ替えなし
2'b11	BGR 5:5:5	入れ替え

8.5.9　出力 FIFO およびフォーマッタ

出力フォーマッタは、16 ワード×24 ビットメモリから構成される出力 FIFO を備えています。bpp 設定により、入力選択はピクセルアンパック（16、18、24bpp）またはパレット（1、2、4、8bpp）のいずれかになります。

出力 FIFO は、ピクセルアンパックのスレーブで、ピクセルアンパックは直接またはパレット経由でピクセルを出力 FIFO に駆動します。出力 FIFO は、別のピクセルをキューに受け入れ不可のときバックプレッシャーパイプラインフリーズ機能を提供します。この機能により LCD コントローラは、フレームの開始時にフレームバッファデータをプリフェッチし、入力 FIFO と出力 FIFO の両方をデータで満たし、最初の行が表示可能になるまでフリーズします。タイミング＆制御ユニットからの最初のデータイネーブル（LCD_DE）信号がアクティブになると、出力 FIFO の読み出し側は、アクティブな各水平ライン期間の残りの間、連続的に読み出しを行います。これらの読み出しの結果、アンパックユニット、次に DMA コントローラが再起動され、要求に従ってフレームバッファデータにアクセスします。

出力フォーマッタは、下表に示すコントロールレジスタ 1（rLcd_CR1）に定義されている bLcd_BPP、bLcd_OPS、および bLcd_RGB に従って出力 FIFO から読み出したピクセルを解釈します。

BPP: bLcd_BPP
OPS: bLcd_OPS
RGB: bLcd_RGB

LCD出力フォーマット
BPP: 1, 2, 4, 8 (パレット)

BPP	OPS[1:0]	RGB	RGB フォーマット	BGR フォーマット	端子	備考
1	00	0	5:6:5	-	LCD_R[5:1] LCD_G[5:0] LCD_B[5:1]	パレットで定義されたRGB
2	00	0		-		パレットで定義されたRGB
4	00	0		-		パレットで定義されたRGB
8	00	0		-		パレットで定義されたRGB
1	10	0	5:6:5	-	LCD_R[7:3] LCD_G[7:2] LCD_B[7:3]	パレットで定義されたRGB
2	10	0		-		パレットで定義されたRGB
4	10	0		-		パレットで定義されたRGB
8	10	0		-		パレットで定義されたRGB
1	01	0	5:5:5	-	LCD_R[5:1] LCD_G[5:1] LCD_B[5:1]	パレットで定義されたRGB
2	01	0		-		パレットで定義されたRGB
4	01	0		-		パレットで定義されたRGB
8	01	0		-		パレットで定義されたRGB
1	11	0	5:5:5	-	LCD_R[7:3] LCD_G[7:3] LCD_B[7:3]	パレットで定義されたRGB
2	11	0		-		パレットで定義されたRGB
4	11	0		-		パレットで定義されたRGB
8	11	0		-		パレットで定義されたRGB
1	00	1	-	5:6:5	LCD_R[5:1] LCD_G[5:1] LCD_B[5:1]	パレットでBとRを入れ替え
2	00	1	-			パレットでBとRを入れ替え
4	00	1	-			パレットでBとRを入れ替え
8	00	1	-			パレットでBとRを入れ替え
1	10	1	-	5:6:5	LCD_R[7:3] LCD_G[7:3] LCD_B[7:3]	パレットでBとRを入れ替え
2	10	1	-			パレットでBとRを入れ替え
4	10	1	-			パレットでBとRを入れ替え
8	10	1	-			パレットでBとRを入れ替え
1	01	1	-	5:5:5	LCD_R[5:1] LCD_G[5:1] LCD_B[5:1]	パレットでBとRを入れ替え
2	01	1	-			パレットでBとRを入れ替え
4	01	1	-			パレットでBとRを入れ替え
8	01	1	-			パレットでBとRを入れ替え
1	11	1	-	5:5:5	LCD_R[7:3] LCD_G[7:3] LCD_B[7:3]	パレットでBとRを入れ替え
2	11	1	-			パレットでBとRを入れ替え
4	11	1	-			パレットでBとRを入れ替え
8	11	1	-			パレットでBとRを入れ替え

図 8.7　　LCD 出力フォーマット指定 ― bLcd_BPP：1、2、4、8

BPP: bLcd_BPP
OPS: bLcd_OPS
RGB: bLcd_RGB

LCD出力フォーマット
BPP: 16, 18, 24 (フレームバッファ)

BPP	OPS[1:0]	RGB	RGB フォーマット	BGR フォーマット	端子	備考
16	00	-	5:6:5	-	LCD_R[5:1] LCD_G[5:0] LCD_B[5:1]	フレームバッファ直接出力
16	01	-	5:5:5	-	LCD_R[5:1] LCD_G[5:1] LCD_B[5:1]	フレームバッファ直接出力
16	10	-	5:6:5	-	LCD_R[7:3] LCD_G[7:2] LCD_B[7:3]	フレームバッファ直接出力
16	11	-	5:5:5	-	LCD_R[7:3] LCD_G[7:3] LCD_B[7:3]	フレームバッファ直接出力
18	0-	-	6:6:6	-	LCD_R[5:0] LCD_G[5:0] LCD_B[5:0]	フレームバッファ直接出力
18	1-	-	6:6:6	-	LCD_R[7:2] LCD_G[7:2] LCD_B[7:2]	フレームバッファ直接出力
24	-	-	8:8:8	-	LCD_R[7:0] LCD_G[7:0] LCD_B[7:0]	フレームバッファ直接出力

図 8.8　　LCD 出力フォーマット指定 ― bLcd_BPP：16、18、24

注　意

- 18bpp 未満かつ bLcd_OPS[1]＝0 の場合、LCD パネルへの 18 ビット RGB インタフェースの最上位ビットにマッピングされます。未使用の出力ピンは 0 に駆動されます。
- 18bpp 未満かつ bLcd_OPS[1]＝1 の場合、LCD パネルへの 24 ビット RGB インタフェースの最上位ビットにマッピングされます。未使用の出力ピンは 0 に駆動されます。

出力 FIFO の書き込み側と読み出し側の両方ともピクセルクロック（LCD_ECLK）ドメインにあります。アドレスポインタが、FIFO エンプティおよびフルフラグ計算に使用され、同様にルックアヘッドパイプラインフリーズ信号用に使用されます。

空きがない状態での FIFO 書き込み、または有効なデータがない状態での読み出しがあった場合には、割り込み bLcd_OFO（出力 FIFO – オーバーラン）および bLcd_OFU（出力 FIFO – アンダーラン）が発生します。（バックプレッシャーパイプラインフリーズ機能のため）書き込み側論理は設計上オーバーランできません。読み出し側には bLcd_OFU 割り込みが発生する場合があります。これは、フレームバッファデータにアクセスする際にバス帯域が不十分なことに起因する可能性があります。

8.5.10　コンフィグレーションレジスタの初期化

コンフィグレーションレジスタの変更または初期化用に、LCD コントローラは下記のシーケンスサポートを提供します。

(1) bLcd_LCE をクリアしてください（LCD ディスエーブル）。LCD タイミングユニットは現在のフレーム表示が終わるのを待ち、パネル信号を強制的に論理 0 にします。LCD コントローラコアは内部で下記の信号を論理 0 に保持します。

- LCD_VSYNC
- LCD_HSYNC
- LCD_DE
- LCD_PCLK
- LCD_R[7:0]
- LCD_B[7:0]
- LCD_G[7:0]

(2) LCD の誤動作を避けるために、コンフィグレーションレジスタはすべてこの段階で設定する必要があります。コンフィグレーションレジスタは下記のとおりです。

- rLcd_CR1 レジスタ。ただし、bLcd_LCE（0 にクリア済み、つまり LCD ディスエーブル）および bLcd_LPE（要求される電源モードシーケンスに依存）を除く。
- rLcd_HTR、rLcd_VTR1 および rLcd_VTR2 レジスタ（水平方向および垂直方向タイミング）
- rLcd_HPPLOR および rLcd_HVTER レジスタ（拡張水平方向および垂直方向タイミング）
- rLcd_PCTR レジスタのピクセルクロック RESET は非アクティブ（リセット状態なし）に解除
- rLcd_PWMDCR_0、rLcd_PWMDCR_1、rLcd_PWMFR_0、rLcd_PWMFR_1 PWM0&1 レジスタ（PWM 制御）
- rLcd_DBAR レジスタ（DMA ベースアドレス）
- パレットレジスタ初期化

(3) コンフィグレーションレジスタ初期化後、コントロールレジスタ 1（rLcd_CR1）の bLcd_LCE がオンに設定されます。bLcd_LCE がオンになると、(1)に挙げた LCD パネルに送られる信号は、設定されたアクティブレベルに駆動可能となり、最初のフレームのフェッチおよび表示が開始されます。

8.5.11　割り込み

割り込みステータスレジスタ（rLcd_ISR）、割り込みマスクレジスタ（rLcd_IMR）、および割り込みベクタレジスタ（rLcd_IVR）の 3 つの割り込みレジスタはお互いに連携しています。rLcd_ISR と rLcd_IMR は読み出し／書き込みレジスタですが、rLcd_IVR は読み出し専用です。

内部で発生した割り込みは、rLcd_ISR の対応するビットをセットします。rLcd_IMR においてエラーに対応するマスクビットがセットされている場合、rLcd_IVR レジスタの対応するエラービットがセットされ、プロセッサに対する割り込みが発生します。個々の割り込みを処理するべきかどうかを判断するために、プロセッサ割り込みハンドラが rLcd_IVR を読み出すことにより、応答します。割り込み応答が終了すると、プログラムは、rLcd_ISR 内の対応する割り込みビットに論理 1 を書き込むことにより、rLcd_ISR 内の割り込みをリセットできます。rLcd_IMR レジスタを介して、プログラムはどの割り込みを許可するかをフルコントロールできます。

8.5.12　電源シーケンス

LCD コントローラは、下記のパワーアップシーケンスをサポートします。

(4) LCD コントローラを搭載する VLSI デバイスおよび LCD パネルに電源が供給されます。LCD コントローラは内部で下記の信号を論理 0 に保持します。

- LCD_VSYNC
- LCD_HSYNC
- LCD_DE
- LCD_PCLK
- LCD_R[7:0]
- LCD_B[7:0]
- LCD_G[7:0]

(5) LCD の誤動作を避けるために、コンフィグレーションレジスタはすべてこの段階で設定する必要があります（任意設定）。コンフィグレーションレジスタは下記のとおりです。

- rLcd_CR1 レジスタ
- rLcd_HTR、rLcd_VTR1 および rLcd_VTR2 レジスタ（水平方向および垂直方向タイミング）
- rLcd_HPPLOR および rLcd_HVTER レジスタ（拡張水平方向および垂直方向タイミング）
- rLcd_PCTR レジスタのピクセルクロック RESET は非アクティブ（リセット状態なし）に解除
- rLcd_PWMDCR_0、rLcd_PWMDCR_1、rLcd_PWMFR_0、rLcd_PWMFR_1 PWM0&1 レジスタ（PWM 制御）
- rLcd_DBAR レジスタ（DMA ベースアドレス）
- パレットレジスタ初期化

(6) LCD パネルにより規定されプロセッサタイマにより制御される時間が経過した後、コントロールレジスタ 1（rLcd_CR1）の bLcd_LPE がオンに設定されます。bLcd_LPE がオンになると、(1)に挙げた LCD パネルに送られる信号は、設定されたアクティブレベルに駆動可能となります。

LCD コントローラは、下記のパワーダウンシーケンスをサポートします。

(1) コントロールレジスタ 1（rLcd_CR1）の bLcd_LPE はオフに設定されます。

(2) 表示中の現在のフレームが完了すると、上記の LCD パネルへの各信号は強制的に 0 にされます。

(3) このとき、LCD パネルへの各信号は強制的に 0 にされ、割り込み bLcd_LDD が発生し、フレーム完了を知らせます。LCD パネルにより規定された時間が経過した後、ディスプレイの電源は切断されます。

LCD パネルの外部電源へのイネーブルとして接続されているコントロールレジスタ 1（rLcd_CR1）の bLcd_LPE を、LCD パネルの電源をイネーブル／ディスエーブルするために使用することが可能です。

8.5.13　フレームバッファ 24bpp パックワード

LCD コントローラは、32 ビットフレームバッファワード内のパック化なしまたはパック化した 24 ビットピクセルで 24bpp の LCD パネルを駆動することが可能です。フレームバッファワードは、32 ビットワード内の 24 ビットピクセル用に下記のように構成されます。

表 8.30　LCD フレームバッファの構成（bLcd_FBP＝0 かつ bLcd_BPP＝3’b110）

フレームバッファ ベースアドレス	フレームバッファの中味 パック化なしデータ 24bpp モード： ●bLcd_FBP＝0 かつ bLcd_BPP＝3’b110	
	ビット[31:24]	ビット[23:0]
32’h0	使用しない	ピクセル 0（24 ビットピクセルデータ：RGB のみ）
32’h4	使用しない	ピクセル 1（24 ビットピクセルデータ：RGB のみ）
....	....	....

このように各 32 ビットフレームバッファワードの最上位バイトは使用されません。

本構成では、入れ替えなしモード（RGB のみ）が使用できます。

コントロールレジスタ 1（rLcd_CR1）の bLcd_LPE がセットされている場合、LCD コントローラには、32 ビットフレームバッファワード内の 24bpp ピクセルを下表に従ってパック可能なプログラミングモードがあります。

表 8.31　LCD フレームバッファの構成（bLcd_FBP＝1 かつ bLcd_BPP＝3’b110）

フレームバッファ ベースアドレス	フレームバッファの中味 パック化データ 24bpp モード： ●bLcd_FBP＝1 かつ bLcd_BPP＝3’b110（下記の注意事項参照）			
	ビット[31:24]	ビット[23:0]	ビット[15:8]	ビット[7:0]
32’h0	P1-バイト 0：B	P0-バイト 2：R	P0-バイト 1：G	P0-バイト 0：B
32’h4	P2-バイト 1：G	P2-バイト 0：B	P1-バイト 2：R	P1-バイト 1：G
32’h8	P3-バイト 2：R	P3-バイト 1：G	P3-バイト 0：B	P2-バイト 2：R
....	....	....		

本構成では、入れ替えなしモード（RGB のみ）が使用できます。

（ピクセル内で）Pn＝ピクセル n となります。したがって、32 ビットフレームバッファワードが 3 つ連続するごとに、LCD コントローラは 4 つの 24 ビットピクセルをアンパックすることが可能です。

注　意

パック化モード（bLcd_FBP＝1）は、24bpp モード（bLcd_BPP＝3’b110）の時だけ使用できます。

ファームウェアが本制限事項を順守しない場合、LCD コントローラ機能は保証されません。

8.5.14　パルス幅変調

TFT LCD パネルのバックライト用 LED の出現とともに、2 つのパルス幅変換（PWM0&1）モジュールが LCD コントローラに追加されました。通常、DC/DC コンバータは LED に一定の電流を供給し、コンバータは輝度入力を備えています。輝度入力を PWM 信号で変調することにより、パネルが消費する電力と輝度のトレードオフを行います。

PWM0 および PWM1 モジュールのリファレンスクロックは LCD_ECLK になります。希望の PWM 周波数は、PWM 周波数クロック分周器レジスタ（rLcd_PWMFR_0 および rLcd_PWMFR_1）で設定され、この PWM 周波数クロック分周器レジスタは、PWM デューティサイクルレジスタ（rLcd_PWMDCR_0 および rLcd_PWMDCR_1）でパルス幅を変調されます。

コントロール／ステータスレジスタはスレーブバス LCD_HCLK によりクロック同期され、一方 PWM モジュールは LCD_ECLK によりクロック同期されるので、まずクロックドメインクロス論理が CSR 信号を選択した PWM_CLK ドメインに転送し、次に PWM 開始時最初の PWM_CLK サイクルが保持レジスタに転送されます。このように PWM は、周波数およびデューティサイクルを動的に変更することが可能です。

8.5.15　点滅機能

点滅機能の主な目的は画像（ピクチャ）を点滅させることです。

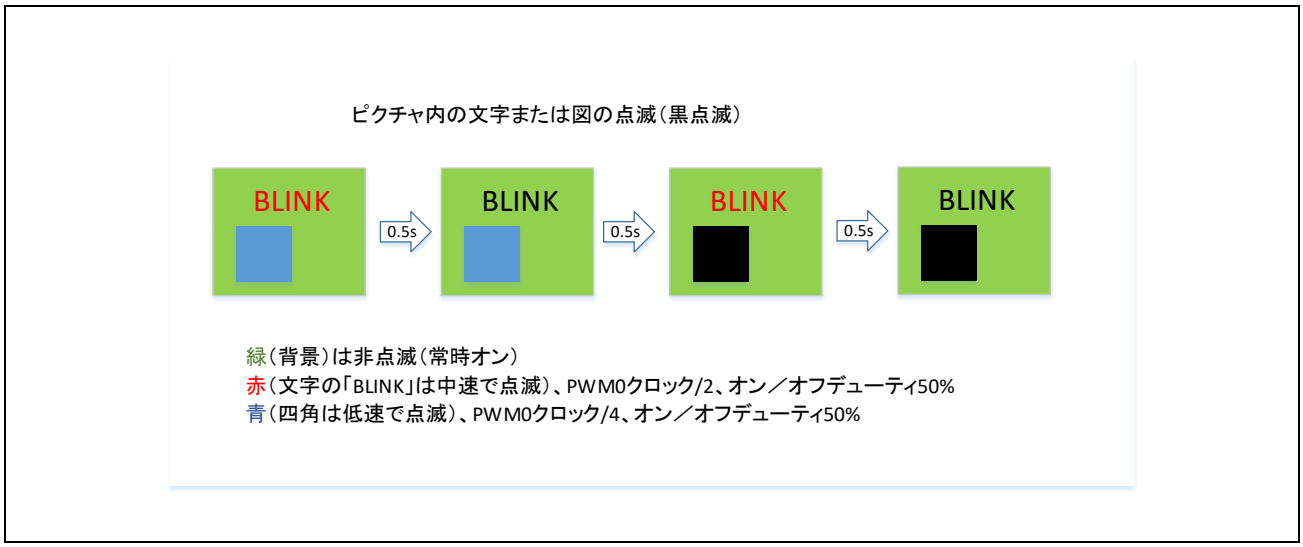


図 8.9　　LCD 点滅の主要原理

点滅 BL[1:0]属性管理の図では、（DDR/SRAM 内の）メモリフレームバッファにおいて、点滅機能をアクティブにした場合またはアクティブにしない場合の各ピクセル（24 ビット上の R,G,B）のマッピングを示します。

点滅機能は下記のように制御されます。

- 点滅の有効または無効は rLcd_GPIOR レジスタの bLcd_BlinkOn により制御されます。
  - LCD_R[7:0]、LCD_G[7:0]、LCD_B[7:0]に提示されるピクセルデータは、点滅機能が有効か無効かによります。
  - 「**図 8.10　点滅 BL[1:0]属性管理**」を参照してください。
- 点滅の輝度モードは rLcd_GPIOR レジスタの bLcd_BlinkMode により制御されます。
  - 点滅頻度は rLcd_PWMFR_0 レジスタの bLcd_PWMFCD_0 および bLcd_PWMFCE_0 によります。

- この場合、本 PWM0 ユニットの出力が BLINK 論理を駆動します。
- 点滅機能は 24bpp モードでのみ使用することが可能です。

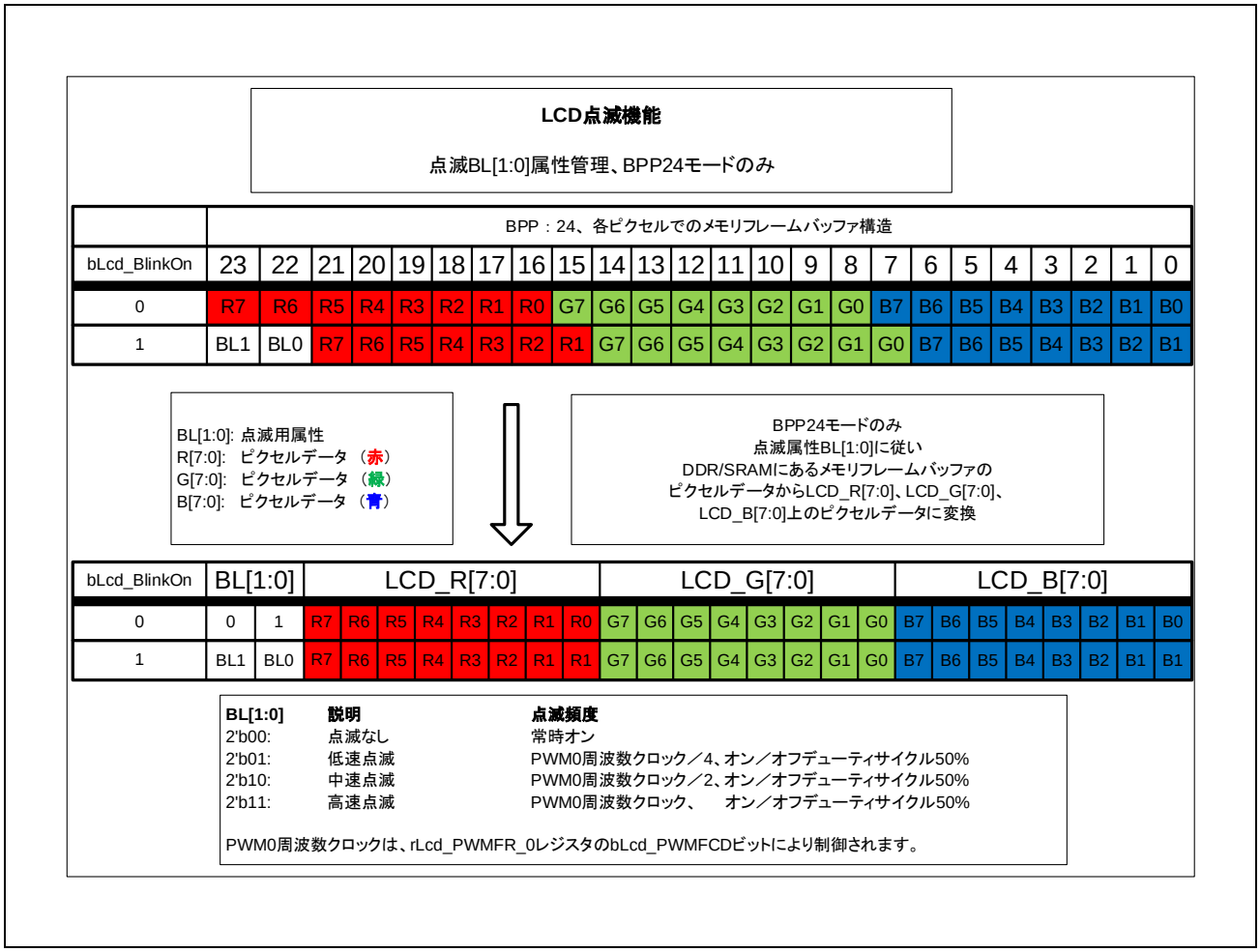


図 8.10　　点滅 BL[1:0]属性管理

8.5.16　制限事項

LCD コントローラには次のような制限事項があります。

- ラインあたり偶数のピクセルを使用することとします（奇数ピクセルの使用は堅く禁止します）。
- LCD コントローラと DDR2/3 間の帯域制限により、アンダーランの入出力 FIFO を検出することが可能です。アンダーラン状態を解決するには、
  - bLcd_FDW 値を増やす（DMA 読み出し要求のバーストサイズ設定をワード単位で）
  - 目標 LCD 解像度が DDR2/3 または SRAM で使用可能な帯域と互換性があるか確認する
    1) bpp（bit per pixel）およびリフレッシュ速度に依存
    2) 最高周波数：83.3MHz
    3) LCD と DDR2/3 間の帯域制限
    「**8.5.2 帯域制限**」を参照してください。
- 帯域問題を避けるために推奨される bLcd_FDW パラメータの設定
  - bLcd_FDW＝4 は、低解像度に対してのみ使用
  - bLcd_FDW＝16 は、高解像度に対してのみ使用
- ピクセルパック化モード
  - パック化モード（bLcd_FBP＝1）は 24bpp モード（bLcd_BPP＝3’b110）の時だけ使用できます。
  - ファームウェアが本制限事項を順守しない場合、パック化モード機能は保証されず、RGB 出力は間違っている可能性があります。
  - 本モードでは、入れ替えモードは使用できません。
- 点滅機能
  - 点滅機能は、24bpp モード（bLcd_BPP＝3’b110）の時だけ使用できます。
  - ファームウェアが本制限事項を順守しない場合、点滅機能は保証されず、RGB 出力は間違っている可能性があります。
  - 点滅機能は、PWM0 ユニットのすべてのリソースを再利用します。
    PWM0 ユニットは、BLINK 周波数を駆動します。

第9章　セマフォ

9.1　概要

セマフォとは、すべての内部共有リソース（バッファプール、メモリ領域、およびペリフェラル）の排他的アクセスを制御するハードウェアロックメカニズムを設定するために使用される R/W レジスタ群です。

プロセッサは、共有リソースを使用する前にハードウェアロックセマフォでリソースの所有者になっている必要があります。

（外部 CPU を含めて）4 つの CPU でリソースを共有し、64 個のハードウェアロックセマフォを利用できます。

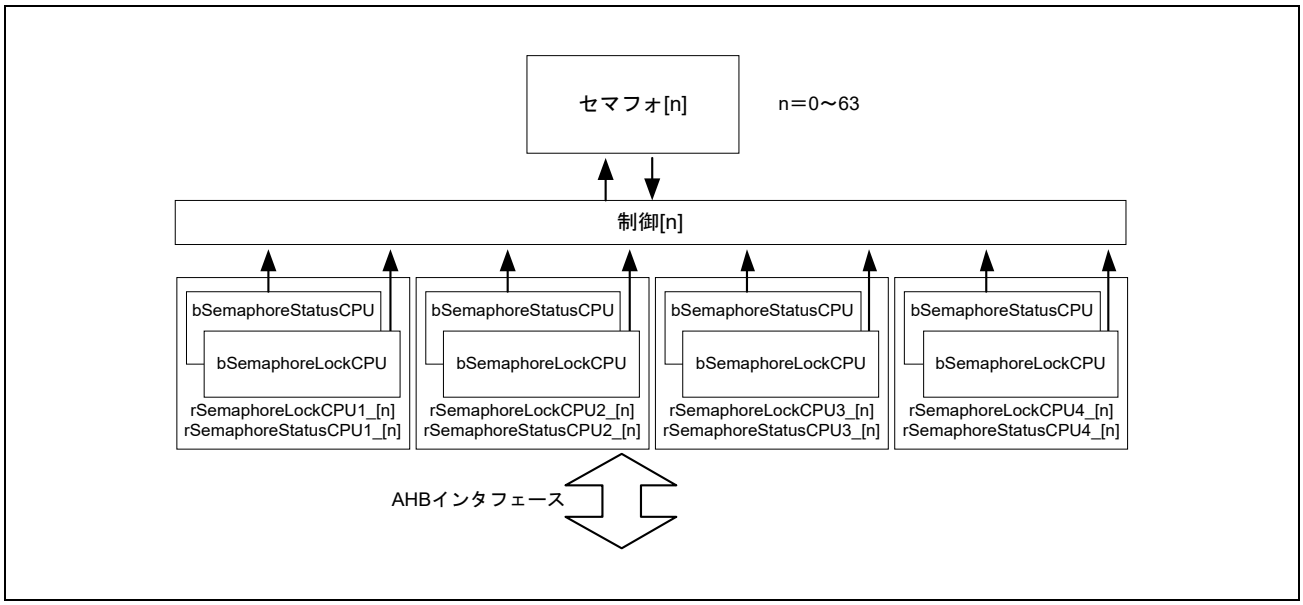


図 9.1　セマフォの概要

9.2　信号インタフェース

表 9.1　信号インタフェース

信号名	入力 出力	説明
クロック		
SEMAP_HCLK	入力	内部バスクロック（AHB）

9.3　レジスタマップ

表 9.2　レジスタマップ

アドレス	レジスタシンボル	レジスタ名
5300 0000h＋10h×n	rSemaphoreLockCPU1_[n]（n＝0～63）	セマフォロック CPU1 レジスタ[n]
5300 0004h＋10h×n	rSemaphoreStatusCPU1_[n]（n＝0～63）	セマフォステータス CPU1 レジスタ[n]
5300 1000h＋10h×n	rSemaphoreLockCPU2_[n]（n＝0～63）	セマフォロック CPU2 レジスタ[n]
5300 1004h＋10h×n	rSemaphoreStatusCPU2_[n]（n＝0～63）	セマフォステータス CPU2 レジスタ[n]
5300 2000h＋10h×n	rSemaphoreLockCPU3_[n]（n＝0～63）	セマフォロック CPU3 レジスタ[n]
5300 2004h＋10h×n	rSemaphoreStatusCPU3_[n]（n＝0～63）	セマフォステータス CPU3 レジスタ[n]
5300 3000h＋10h×n	rSemaphoreLockCPU4_[n]（n＝0～63）	セマフォロック CPU4 レジスタ[n]
5300 3004h＋10h×n	rSemaphoreStatusCPU4_[n]（n＝0～63）	セマフォステータス CPU4 レジスタ[n]

9.4　レジスタの説明

9.4.1　rSemaphoreLockCPU[m]_[n] ― セマフォロック CPU[m]レジスタ[n]

CPU[m]（m＝1～4）、セマフォ[n]（n＝0～63）

CPU[m]はアドレスによって区別されるもので、実際の CPU とは関係ありません。

セマフォがフリーの状態で、このレジスタを読み出すとその CPU に対して予約が行われます。

セマフォの解放は、予約された CPU が 3’b000 を書き込む場合のみ可能です。

アドレス　rSemaphoreLockCPU1_[n]：5300 0000h＋10h×n
rSemaphoreLockCPU2_[n]：5300 1000h＋10h×n
rSemaphoreLockCPU3_[n]：5300 2000h＋10h×n
rSemaphoreLockCPU4_[n]：5300 3000h＋10h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	bSemaphoreLockCPU		
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 9.3　rSemaphoreLockCPU[m]_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b3	予約ビット		R
b2～b0	bSemaphoreLockCPU	各セマフォ[n]について： CPU による読み出し 3’b000 セマフォはフリー CPU のハードウェア予約受付 3’b100 セマフォは CPU1 により予約済み 3’b101 セマフォは CPU2 により予約済み 3’b110 セマフォは CPU3 により予約済み 3’b111 セマフォは CPU4 により予約済み CPU による書き込み 3’b000 セマフォ解放 3’b000 以外 何もしない	R/W

9.4.2　rSemaphoreStatusCPU[m]_[n] ― セマフォステータス CPU[m]レジスタ[n]

CPU[m]（m＝1～4）、セマフォ[n]（n＝0～63）

アドレス　rSemaphoreStatusCPU1_[n]：5300 0004h＋10h×n
rSemaphoreStatusCPU2_[n]：5300 1004h＋10h×n
rSemaphoreStatusCPU3_[n]：5300 2004h＋10h×n
rSemaphoreStatusCPU4_[n]：5300 3004h＋10h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	bSemaphoreStatusCPU		
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 9.4　rSemaphoreStatusCPU[m]_[n]レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b3	予約ビット		R
b2～b0	bSemaphoreStatusCPU	各セマフォ[n]について： セマフォの現在値の読み出し 3’b000 セマフォはフリー 3’b001、3’b010、3’b011 予約 3’b100 セマフォは CPU1 により予約済み 3’b101 セマフォは CPU2 により予約済み 3’b110 セマフォは CPU3 により予約済み 3’b111 セマフォは CPU4 により予約済み	R

9.5　動作説明

9.5.1　セマフォ[n]（n＝0～63）

本ステートマシンは、セマフォの予約および解放を管理します。

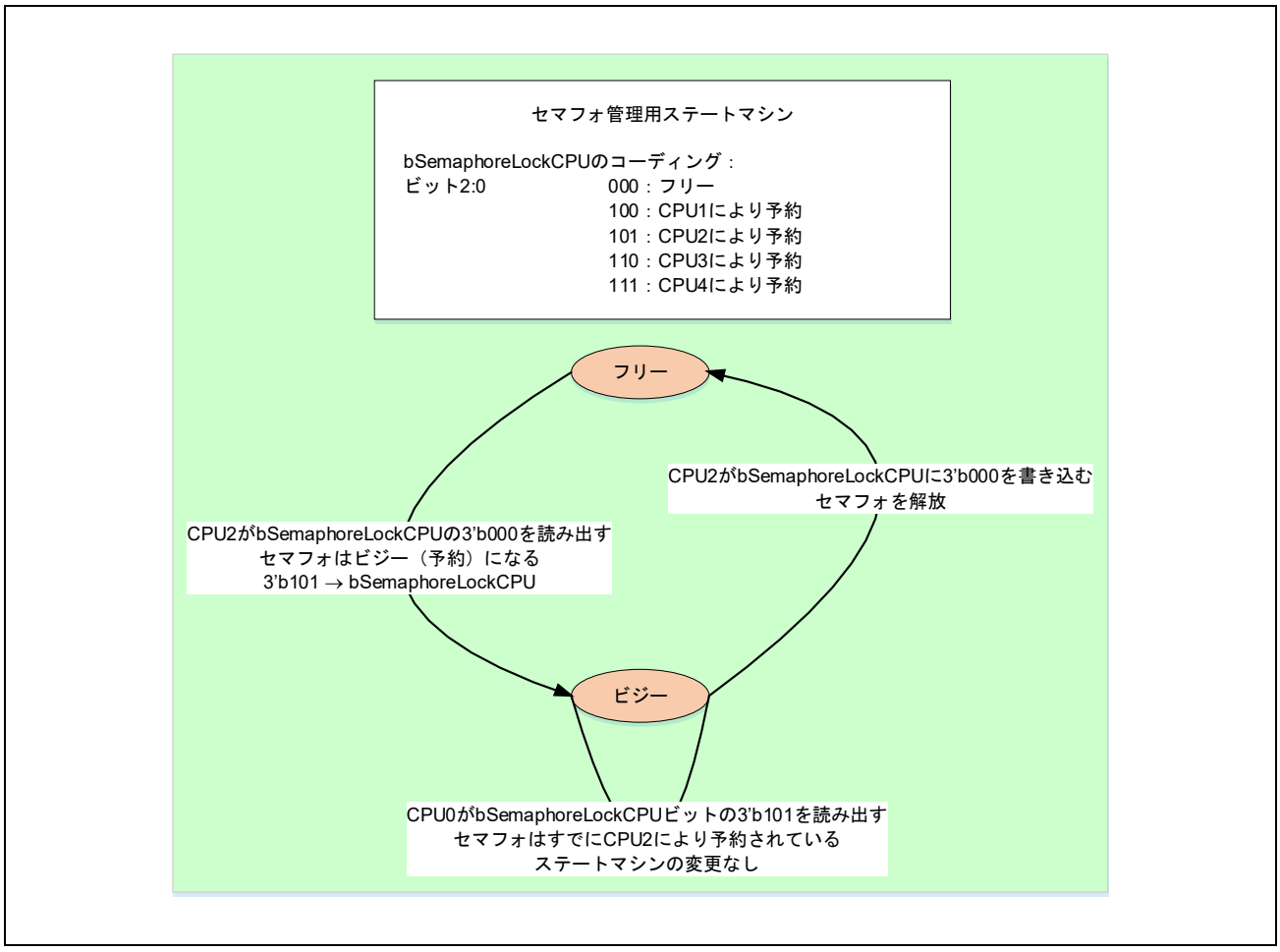


図 9.2　セマフォステートマシン

9.5.2　CPU 識別とアドレスの復号化

同一のセマフォ[n]は、アクセス用に 4 つの別々のアドレスを持っています。

セマフォ[n]は、レジスタによりアクセス可能です。

- rSemaphoreLockCPU1_[n]
- rSemaphoreLockCPU2_[n]
- rSemaphoreLockCPU3_[n]
- rSemaphoreLockCPU4_[n]

セマフォステートマシンは、本アドレスを使って bSemaphoreLockCPU にセマフォ識別を書き込みます。

- 3’b100：CPU1 により予約されているセマフォ
- 3’b101：CPU2 により予約されているセマフォ
- 3’b110：CPU3 により予約されているセマフォ
- 3’b111：CPU4 により予約されているセマフォ

セマフォ識別を上手に管理するには、各 CPU は、セマフォ内アクセス用に別々のアドレスを使用する必要があります。

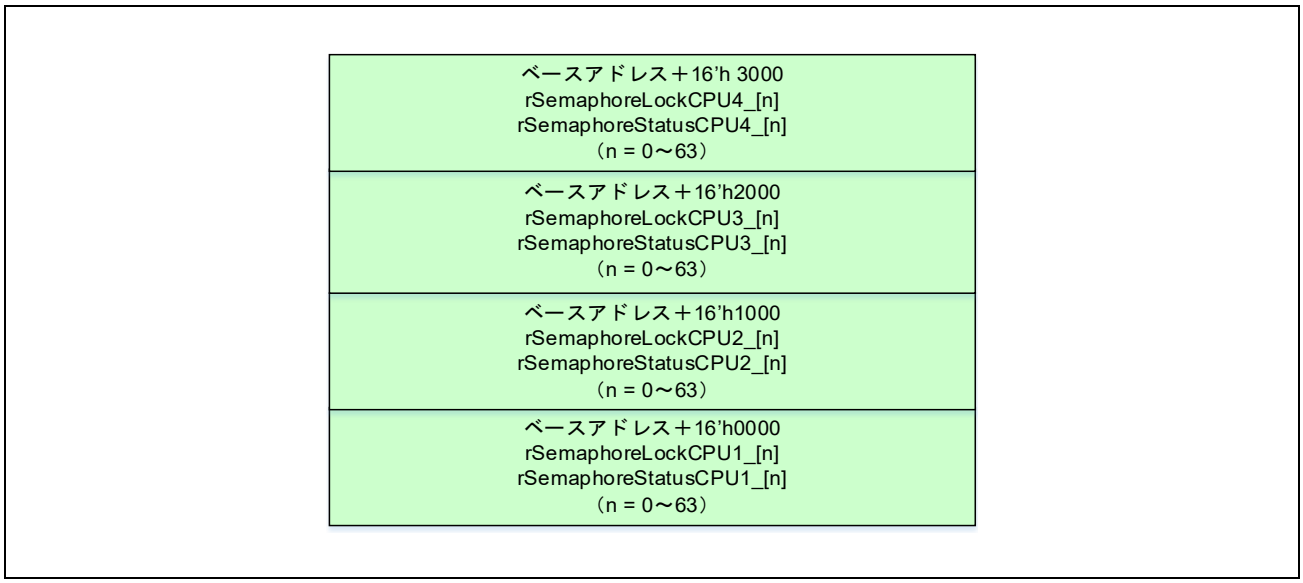


図 9.3　CPU 識別用のセマフォアドレスブロック

9.6　使用上の注意事項

複数の CPU 間でセマフォを使用するには、基本手順は以下のとおりです。

CPU1 と CPU2 によるセマフォ 55 の例

(1) CPU1 が rSemaphoreLockCPU1_55 レジスタの bSemaphoreLockCPU ビットを読み出す

- 3’b000 かどうかの確認
  - 読み出し値が 3’b000 である場合、セマフォ 55 は CPU1 用に予約されます。CPU1 は、すべての内部共有リソース（バッファプール、メモリ領域、およびペリフェラル）に排他的にアクセス可能です。
  - 読み出し値が 3’b101 である場合、セマフォ 55 はすでに CPU2 用に予約済みで、CPU1 は待機しなければなりません。

(2) CPU2 が rSemaphoreLockCPU2_55 レジスタの bSemaphoreLockCPU ビットを読み出す

- 3’b000 かどうかの確認
  - 読み出し値が 3’b000 である場合、セマフォ 55 は CPU2 用に予約されます。CPU2 は、すべての内部共有リソースに排他的にアクセス可能です。
  - 読み出し値が 3’b100 である場合、セマフォ 55 はすでに CPU1 用に予約済みで、CPU2 は待機しなければなりません。

この例では、CPU1 と CPU2 は、Semaphore55 を使いリソースを共有します。

- CPU1 は、rSemaphoreLockCPU1_55 レジスタを使って Semaphore55 を管理します。
- CPU2 は、rSemaphoreLockCPU2_55 レジスタを使って Semaphore55 を管理します。

Semaphore55 は CPU1 と CPU2 に異なるアドレスを提供し、ハードウェアはこれらのセマフォアドレスを使って CPU を識別します。

セマフォ機能は、下記の割り当てで使用することとします。

- CPU1 は、セマフォ[n]を管理するために rSemaphoreLockCPU1_[n]のみを使用
- CPU2 は、セマフォ[n]を管理するために rSemaphoreLockCPU2_[n]のみを使用
- CPU3 は、セマフォ[n]を管理するために rSemaphoreLockCPU3_[n]のみを使用
- CPU4 は、セマフォ[n]を管理するために rSemaphoreLockCPU4_[n]のみを使用

注　意

異なる CPU が同じアドレスを使用してセマフォ 55 にアクセスすべきではありません。

第10章　外部バスインタフェース（MSEBI）

10.1　概要

MSEBI はプログラマブルです。4 つのチップセレクトがあり、アドレス／データ／制御データがマルチプレクスされます。32 ビットデータバスは、8 ビット、16 ビット、および 32 ビットの外部デバイスと接続できるように構成することが可能です。

MSEBI は、以下の 2 つの独立したブロックで構成されています。

- マスタ
- スレーブ

マスタモードの特長：

外部バスインタフェース（MSEBI）は、外付けの周辺非同期デバイスおよび同期デバイスにアクセスするための制御信号を管理します。読み出し制御信号と書き込み制御信号が分離されているため、非同期式のメモリ接続および周辺デバイス接続が可能です。また、スレーブからの外部ウェイト要求受信機能と DMA 接続機能も搭載されています。同じボード上で、同期式モードと非同期式モード、および、データバス幅（32 ビット、16 ビット、または 8 ビット）の混在が可能です。

スレーブモードの特長：

外部バスインタフェース（MSEBI）は、同期式モードのみサポートします。また、マスタへの外部ウェイト要求送信機能とマスタへの外部 DMA 要求機能も搭載されています。

異なる構成の 3 つの基本モードが利用可能です。

- データバス上の 32 ビットマルチプレクス（MSEBI モード 32）
- データバス上の 16 ビットマルチプレクス（MSEBI モード 16）
- データバス上の 8 ビットマルチプレクス（MSEBI モード 8）

チップセレクト（CS0_N～CS3_N）は、構成に応じて最大 4GB までのプログラマブルアドレスを設定できます。

それぞれスレーブモードとマスタモードにおいて、CPU は MSEBI インタフェースによって使用されない端子のすべてを切断できます。本機能により、未使用端子は汎用入出力端子として利用できるようになります。

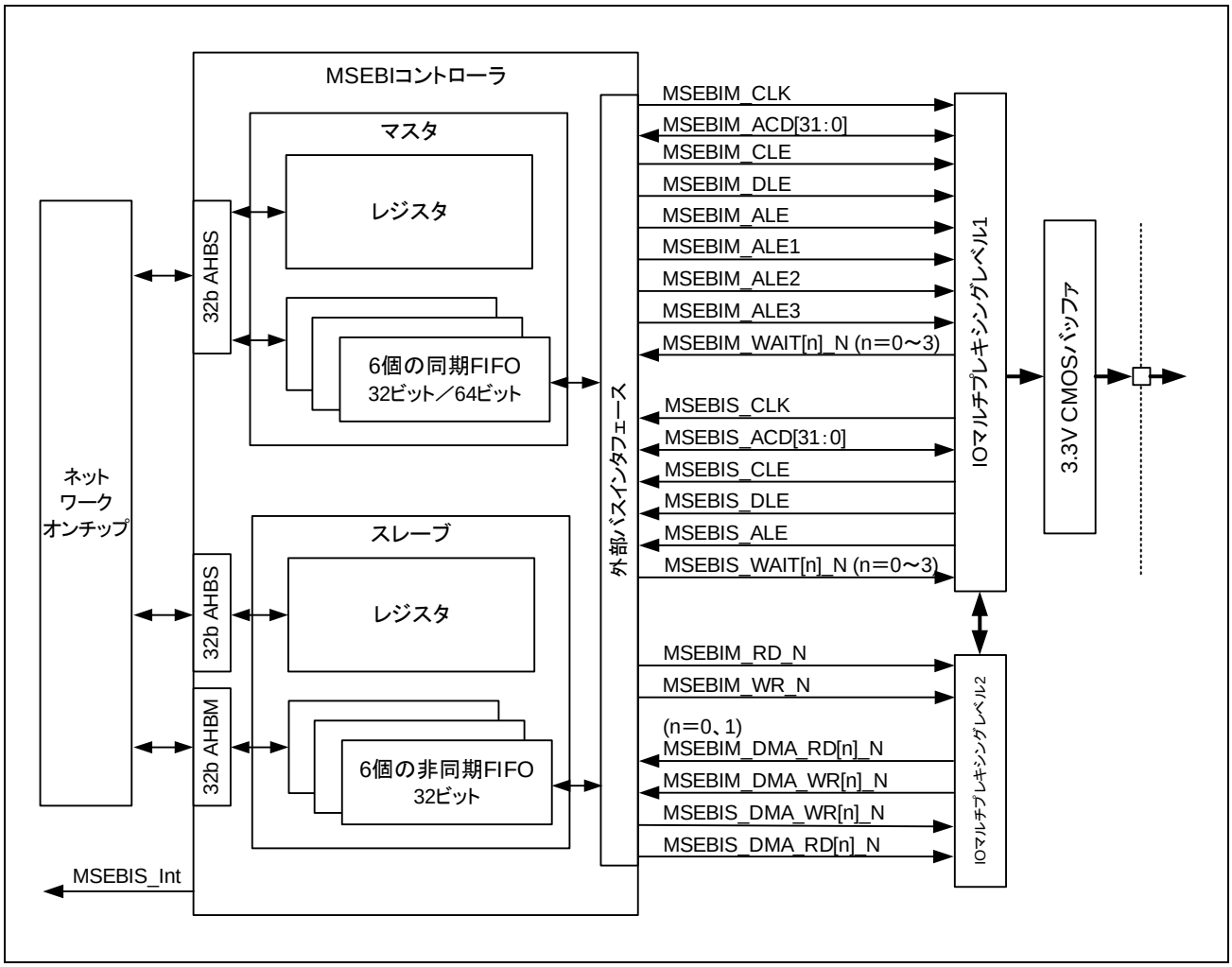


図 10.1　　MSEBI の概要

主な特長は以下のとおりです。

- 8 ビット、16 ビット、および 32 ビットからデータバス幅を選択可能
- マスタ部とスレーブ部の独立したイネーブル制御
- 非同期式モードと同期式モード
- マルチ DLE モード
- バーストモード
- DMA 接続
  - 周辺機能フローコントローラモード
  - 4 本の DMA チャネルを利用可能（外部要求受信可能）
- マスタモード時の FIFO サイズ：
  - マスタ用の CPU 送受信 FIFO：2×32 ワード×32 ビット
  - マスタ用の DMA 送受信 FIFO：4×32 ワード×64 ビット
- スレーブモード時の FIFO サイズ：
  - スレーブ用の CPU 送受信 FIFO：2×32 ワード×32 ビット
  - スレーブ用の DMA 送受信 FIFO：4×32 ワード×32 ビット
- 最大 4 本のチップセレクトライン
- 2B～4GB のプログラマブルアドレス機能
- プログラム可能なセットアップ時間
- プログラム可能なホールド時間
- 外部ウェイト要求（許可／禁止の設定可）

10.1.1　信号インタフェース

表 10.1　信号インタフェース

信号名	入力 出力	説明
クロック		
MSEBIM_HCLK	入力	内部 AHB バスクロック（マスタ）
MSEBIS_HCLK	入力	内部 AHB バスクロック（スレーブ）
割り込み		
MSEBIS_Int	出力	レベル検出割り込み出力、アクティブ High
外部信号（マスタモード）		
MSEBIM_ACD[31:0]	入出力	アドレス、制御データ、およびデータのマルチプレクス
MSEBIM_CLK	出力	設定可能なグローバルクロック（全タイミングの基準）
MSEBIM_ALE	出力	アドレスラッチイネーブル（アクティブ High）
MSEBIM_ALE1	出力	アドレスラッチイネーブル（アクティブ High）
MSEBIM_ALE2	出力	アドレスラッチイネーブル（アクティブ High）
MSEBIM_ALE3	出力	アドレスラッチイネーブル（アクティブ High）
MSEBIM_CLE	出力	アドレス＆コントロールラッチイネーブル（アクティブ High）
MSEBIM_DLE	出力	データラッチイネーブル（アクティブ High）
MSEBIM_RD_N	出力	読み出しイネーブル（アクティブ Low）
MSEBIM_WR_N	出力	書き込みイネーブル（アクティブ Low）
MSEBIM_WAIT[n]_N	入力	現在のサイクルへのウェイト挿入（n＝0～3）（アクティブ Low）
MSEBIM_DMA_RD[n]_N	入力	読み出しモードにおける MSEBI_CS[n]_N の DMA 要求（n＝0、1）
MSEBIM_DMA_WR[n]_N	入力	書き込みモードにおける MSEBI_CS[n]_N の DMA 要求（n＝0、1）
外部信号（スレーブモード）		
MSEBIS_ACD[31:0]	入出力	アドレス、制御データ、およびデータのマルチプレクス
MSEBIS_CLK	入力	外部バスのマスタが供給するクロック
MSEBIS_ALE	入力	アドレスラッチイネーブル（アクティブ High）
MSEBIS_CLE	入力	アドレス＆コントロールラッチイネーブル（アクティブ High）
MSEBIS_DLE	入力	データラッチイネーブル（アクティブ High）
MSEBIS_WAIT[n]_N	出力	現在のサイクルへのウェイト挿入（n=0～3）（アクティブ Low）
MSEBIS_DMA_RD[n]_N	出力	読み出しモードにおける MSEBI_CS[n]_N の DMA 要求（n＝0、1）
MSEBIS_DMA_WR[n]_N	出力	書き込みモードにおける MSEBI_CS[n]_N の DMA 要求（n＝0、1）

備考　GPIO Multiplexed 端子名では、インデックス[n]は最後に付きます。
例）MSEBIM_WAIT_N[n]、MSEBIS_DMA_RD_N[n]

10.1.2　CS[n]の MSEBI マスタアドレス割り当て（CPU）

次の表に記載されている領域にライトまたはリードすると、CPU 送信 FIFO にコマンドが格納され、アドレスに従って MSEBI_CS[n]_N（n=0..3）に対する MSEBI アクセスが生成されます。

表 10.2　CS[n]のアドレス割り当て（CPU）

CS[n]	ベースアドレス
CS0	6000 0000h
CS1	6800 0000h
CS2	7000 0000h
CS3	7800 0000h

10.1.3　マルチプレクス信号インタフェース

MSEBIM_ACD に対するデータマルチプレクサは、MSEBIM_ALE、MSEBIM_CLE、MSEBIM_DLE によって制御され、MSEBIM_CLK の立ち上がりエッジでラッチされます。関連する信号についての詳細が下表に示されています。

アクセスはバスサイズに依存することに注意してください。詳細は下記の表を参照してください。

- 「**表 10.4 MSEBI モード 32、ACD31～0 に対するマルチプレクサ機能**」
- 「**表 10.6 MSEBI モード 16、ACD15～0 に対するマルチプレクサ機能**」
- 「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」

表 10.3　マルチプレクス信号インタフェース（1/2）

信号名	説明
MSEBIM_D[31:16]	データ（モード 32 でのみ使用）
MSEBIM_D[15:8]	データ（モード 32 またはモード 16 でのみ使用）
MSEBIM_D[7:0]	データ
MSEBIM_A[31:0]	アドレス
MSEBI_BE[n]_N	バイトイネーブル（アクティブ Low） ● モード 32（n＝0～3） BE3_N　BE2_N　BE1_N　BE0_N 0　0　0　0　ワード（32 ビット） 1　1　0　0　下位ハーフワード（16 ビット） 0　0　1　1　上位ハーフワード（16 ビット） 1　1　1　0　下位バイト 1　1　0　1　下位バイト＋1 1　0　1　1　下位バイト＋2 0　1　1　1　上位バイト（下位バイト＋3） 1　1　1　1　予約 ● モード 16（n＝0、1） BE1_N　BE0_N 0　0　ハーフワード（16 ビット） 1　0　下位バイト 0　1　上位バイト 1　1　予約 ● モード 8：本モードではバイトイネーブルなし
MSEBI_CS[n]_N	チップセレクト：それぞれ 4GB のプログラマブルアドレス（アクティブ Low） ● アドレス機能について： すべてのチップセレクト（n＝0～3）は、その構成に応じたプログラマブルアドレス機能を設定できます。オプションの ALE フェーズ数を減らすことで、帯域幅を広げて待ち時間を削減したい場合、拡張アドレス機能モードが有効です。

表 10.3　MSEBI マルチプレクス信号インタフェース（2/2）

信号名	説明
MSEBI_CSREG_N	割り込み（マスタからスレーブへ）とステータスを管理するグローバルレジスタへのアクセス（アクティブ Low） **注意）** マスタとスレーブのどちらのモードにおいても、マスタがスレーブの共有レジスタに（MSEBI_CSREG_N を用いて）アクセスしているときは、プリフェッチを行わないでください。 下記のデコーディング表には、特定のチップセレクト（n＝0～3）の共有レジスタにアクセスする方法が示されています。 MSEBI_CS[n]_N　MSEBI_CSREG_N　CS[n]_N アクセスタイプ 0　0　CS[n]_N の共有レジスタにアクセス 0　1　CS[n]_N のメモリ空間にアクセス 1　0　予約 1　1　アクセスなし
MSEBI_DMA_N	現在の要求のイニシエータ（DMA または CPU）を識別します。CS[n]_N（n＝0～3）と対応している必要があります。 詳細については、「**表 10.47 スレーブの要求イニシエータの検出**」を参照してください。
MSEBI_R/W_N	読み出し／書き込み制御 1：読み出しアクセス 0：書き込みアクセス

10.1.3.1　モード 32 のマルチプレクサ

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。

表 10.4　MSEBI モード 32、ACD31～0 に対するマルチプレクサ機能

MSEBI_ACD マルチプレクスバス	ADDRESS ステージ ALE による制御（オプション）	CONTROL ステージ CLE による制御	DATA ステージ DLE による制御
MSEBI(x)_ALE シリアルモードのみ	1'b1	1'b0	1'b0
MSEBIM_ALE パラレルモードのみ	1'b1	1'b0	1'b0
MSEBIM_ALE1 パラレルモードのみ	1'b0	1'b0	1'b0
MSEBIM_ALE2 パラレルモードのみ	1'b0	1'b0	1'b0
MSEBIM_ALE3 パラレルモードのみ	1'b0	1'b0	1'b0
MSEBI(x)_CLE	1'b0	1'b1	1'b0
MSEBI(x)_DLE	1'b0	1'b0	1'b1
ACD 上の割り当て			
MSEBI(x)_ACD[31:11]	1'b0	MSEBI_A[22:2]	MSEBI_D[31:11]
MSEBI(x)_ACD10	1'b0	MSEBI_BE3_N	MSEBI_D10
MSEBI(x)_ACD9	1'b0	MSEBI_BE2_N	MSEBI_D9
MSEBI(x)_ACD8	MSEBI_A31	MSEBI_BE1_N	MSEBI_D8
MSEBI(x)_ACD7	MSEBI_A30	MSEBI_BE0_N	MSEBI_D7
MSEBI(x)_ACD6	MSEBI_A29	MSEBI_R/W_N	MSEBI_D6
MSEBI(x)_ACD5	MSEBI_A28	MSEBI_DMA_N	MSEBI_D5
MSEBI(x)_ACD4	MSEBI_A27	MSEBI_CSREG_N	MSEBI_D4
MSEBI(x)_ACD3	MSEBI_A26	MSEBI_CS3_N	MSEBI_D3
MSEBI(x)_ACD2	MSEBI_A25	MSEBI_CS2_N	MSEBI_D2
MSEBI(x)_ACD1	MSEBI_A24	MSEBI_CS1_N	MSEBI_D1
MSEBI(x)_ACD0	MSEBI_A23	MSEBI_CS0_N	MSEBI_D0

表 10.5　MSEBI モード 32、チップセレクト管理

ALE フェーズなしで利用可能な拡張アドレス機能			
4 チップセレクト 8MB	3 チップセレクト 16MB	2 チップセレクト 32MB	1 チップセレクト 64MB
MSEBI_CS3_N	MSEBI_A23	MSEBI_A23	MSEBI_A23
MSEBI_CS2_N	MSEBI_CS2_N	MSEBI_A24	MSEBI_A24
MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_A25
MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N

ALE フェーズは、レイテンシを削減する設定が各チップセレクト CS[n]_N（n＝0～3）ごとに可能です。

マスタモードの場合

CPU（n＝0～3）

- MSEBI_A26～MSEBI_A2 は、CPU のアドレスが使用されます（CS[n]_N ごとに 128MB）。
- MSEBI_A31～MSEBI_A27 は、下記のレジスタによって指定されます。
  - rMSEBIM_CONFIG_CS[n]_N

DMA（n＝0、1）

- MSEBI_A31～MSEBI_A2 は、下記のレジスタによって指定されます。
  - rMSEBIM_ADDRDMA_CURRENTREAD_CS[n]_N
  - rMSEBIM_ADDRDMA_CURRENTWRITE_CS[n]_N

10.1.3.2　モード 16 のマルチプレクサ

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。

ALE 列に関する注記：

- 複数の ALE ステージが用いられる場合、ALE 列のシンボル#k は、マスタが ALE フェーズをスレーブに送信する際の順番を表します（#1 が最初の ALE、#2 が 2 番目の ALE、以下同様）。

表 10.6　MSEBI モード 16、ACD15～0 に対するマルチプレクサ機能

MSEBI_ACD マルチプレクスバス	ADDRESS ステージ ALE による制御 （オプション、#2）	ADDRESS ステージ ALE による制御 （オプション、#1）	CONTROL ステージ CLE による制御	DATA ステージ DLE による制御
MSEBI(x)_ALE シリアルモードのみ	1’b1	1’b1	1’b0	1’b0
MSEBIM_ALE パラレルモードのみ	1’b0	1’b1	1’b0	1’b0
MSEBIM_ALE1 パラレルモードのみ	1’b1	1’b0	1’b0	1’b0
MSEBIM_ALE2 パラレルモードのみ	1’b0	1’b0	1’b0	1’b0
MSEBIM_ALE3 パラレルモードのみ	1’b0	1’b0	1’b0	1’b0
MSEBI(x)_CLE	1’b0	1’b0	1’b1	1’b0
MSEBI(x)_DLE	1’b0	1’b0	1’b0	1’b1
ACD 上の割り当て				
MSEBI(x)_ACD[15:9]	1’b0	MSEBI_A[23:17]	MSEBI_A[7:1]	MSEBI_D[15:9]
MSEBI(x)_ACD8	1’b0	MSEBI_A16	MSEBI_BE1_N	MSEBI_D8
MSEBI(x)_ACD7	MSEBI_A31	MSEBI_A15	MSEBI_BE0_N	MSEBI_D7
MSEBI(x)_ACD6	MSEBI_A30	MSEBI_A14	MSEBI_R/W_N	MSEBI_D6
MSEBI(x)_ACD5	MSEBI_A29	MSEBI_A13	MSEBI_DMA_N	MSEBI_D5
MSEBI(x)_ACD4	MSEBI_A28	MSEBI_A12	MSEBI_CSREG_N	MSEBI_D4
MSEBI(x)_ACD3	MSEBI_A27	MSEBI_A11	MSEBI_CS3_N	MSEBI_D3
MSEBI(x)_ACD2	MSEBI_A26	MSEBI_A10	MSEBI_CS2_N	MSEBI_D2
MSEBI(x)_ACD1	MSEBI_A25	MSEBI_A9	MSEBI_CS1_N	MSEBI_D1
MSEBI(x)_ACD0	MSEBI_A24	MSEBI_A8	MSEBI_CS0_N	MSEBI_D0

表 10.7　MSEBI モード 16、チップセレクト管理

ALE フェーズなしで利用可能な拡張アドレス機能			
4 チップセレクト 256B	3 チップセレクト 512B	2 チップセレクト 1KB	1 チップセレクト 2KB
MSEBI_CS3_N	MSEBI_A8	MSEBI_A8	MSEBI_A8
MSEBI_CS2_N	MSEBI_CS2_N	MSEBI_A9	MSEBI_A9
MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_A10
MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N

1 つの ALE フェーズで利用可能な拡張アドレス機能			
4 チップセレクト 16MB	3 チップセレクト 32MB	2 チップセレクト 64MB	1 チップセレクト 128MB
MSEBI_CS3_N	MSEBI_A24	MSEBI_A24	MSEBI_A24
MSEBI_CS2_N	MSEBI_CS2_N	MSEBI_A25	MSEBI_A25
MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_A26
MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N

ALE フェーズは、レイテンシを削減する設定が各チップセレクト CS[n]_N（n＝0～3）ごとに可能です。

マスタモードの場合

CPU（n＝0～3）

- MSEBI_A26～MSEBI_A1 は、CPU のアドレスが使用されます（CS[n]_N ごとに 128MB）。
- MSEBI_A31～MSEBI_A27 は、下記のレジスタによって指定されます。
  - rMSEBIM_CONFIG_CS[n]_N

DMA（n＝0、1）

- MSEBI_A31～MSEBI_A1 は、下記のレジスタによって指定されます。
  - rMSEBIM_ADDRDMA_CURRENTREAD_CS[n]_N（n＝0、1）
  - rMSEBIM_ADDRDMA_CURRENTWRITE_CS[n]_N（n＝0、1）

10.1.3.3　モード 8 のマルチプレクサ

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。

ALE 列に関する注記：

- 複数の ALE ステージが用いられる場合、ALE 列のシンボル#k は、マスタが ALE フェーズをスレーブに送信する際の順番を表します（#1 が最初の ALE、#2 が 2 番目の ALE、以下同様）。

表 10.8　MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能

MSEBI_ACD マルチプレクスバス	ADDRESS ステージ ALE による制御（オプション、#4）	ADDRESS ステージ ALE による制御（オプション、#3）	ADDRESS ステージ ALE による制御（オプション、#2）	ADDRESS ステージ ALE による制御（オプション、#1）	CONTROL ステージ CLE による制御	DATA ステージ DLE による制御
MSEBI(x)_ALE シリアルモードのみ	1'b1	1'b1	1'b1	1'b1	1'b0	1'b0
MSEBIM_ALE パラレルモードのみ	1'b0	1'b0	1'b0	1'b1	1'b0	1'b0
MSEBIM_ALE1 パラレルモードのみ	1'b0	1'b0	1'b1	1'b0	1'b0	1'b0
MSEBIM_ALE2 パラレルモードのみ	1'b0	1'b1	1'b0	1'b0	1'b0	1'b0
MSEBIM_ALE3 パラレルモードのみ	1'b1	1'b0	1'b0	1'b0	1'b0	1'b0
MSEBI(x)_CLE	1'b0	1'b0	1'b0	1'b0	1'b1	1'b0
MSEBI(x)_DLE	1'b0	1'b0	1'b0	1'b0	1'b0	1'b1
ACD 上の割り当て						
MSEBI(x)_ACD7	1'b0	MSEBI_A24	MSEBI_A16	MSEBI_A8	MSEBI_A0	MSEBI_D7
MSEBI(x)_ACD6	MSEBI_A31	MSEBI_A23	MSEBI_A15	MSEBI_A7	MSEBI_R/W_N	MSEBI_D6
MSEBI(x)_ACD5	MSEBI_A30	MSEBI_A22	MSEBI_A14	MSEBI_A6	MSEBI_DMA_N	MSEBI_D5
MSEBI(x)_ACD4	MSEBI_A29	MSEBI_A21	MSEBI_A13	MSEBI_A5	MSEBI_CSREG_N	MSEBI_D4
MSEBI(x)_ACD3	MSEBI_A28	MSEBI_A20	MSEBI_A12	MSEBI_A4	MSEBI_CS3_N	MSEBI_D3
MSEBI(x)_ACD2	MSEBI_A27	MSEBI_A19	MSEBI_A11	MSEBI_A3	MSEBI_CS2_N	MSEBI_D2
MSEBI(x)_ACD1	MSEBI_A26	MSEBI_A18	MSEBI_A10	MSEBI_A2	MSEBI_CS1_N	MSEBI_D1
MSEBI(x)_ACD0	MSEBI_A25	MSEBI_A17	MSEBI_A9	MSEBI_A1	MSEBI_CS0_N	MSEBI_D0

表 10.9　MSEBI モード 8、チップセレクト管理

ALE フェーズなしで利用可能な拡張アドレス機能			
4 チップセレクト 2B	3 チップセレクト 4B	2 チップセレクト 8B	1 チップセレクト 16B
MSEBI_CS3_N	MSEBI_A1	MSEBI_A1	MSEBI_A1
MSEBI_CS2_N	MSEBI_CS2_N	MSEBI_A2	MSEBI_A2
MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_A3
MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N

1 つの ALE フェーズで利用な拡張アドレス機能			
4 チップセレクト 512B	3 チップセレクト 1KB	2 チップセレクト 2KB	1 チップセレクト 4KB
MSEBI_CS3_N	MSEBI_A9	MSEBI_A9	MSEBI_A9
MSEBI_CS2_N	MSEBI_CS2_N	MSEBI_A10	MSEBI_A10
MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_A11
MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N

2 つの ALE フェーズで利用可能な拡張アドレス機能			
4 チップセレクト 128KB	3 チップセレクト 256KB	2 チップセレクト 512KB	1 チップセレクト 1MB
MSEBI_CS3_N	MSEBI_A17	MSEBI_A17	MSEBI_A17
MSEBI_CS2_N	MSEBI_CS2_N	MSEBI_A18	MSEBI_A18
MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_A19
MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N

3 つの ALE フェーズで利用可能な拡張アドレス機能			
4 チップセレクト 32MB	3 チップセレクト 64MB	2 チップセレクト 128MB	1 チップセレクト 256MB
MSEBI_CS3_N	MSEBI_A25	MSEBI_A25	MSEBI_A25
MSEBI_CS2_N	MSEBI_CS2_N	MSEBI_A26	MSEBI_A26
MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_CS1_N	MSEBI_A27
MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N	MSEBI_CS0_N

ALE フェーズは、レイテンシを削減する設定が各チップセレクト CS[n]_N（n＝0～3）ごとに可能です。

マスタモードの場合

CPU（n＝0～3）

- MSEBI_A26～MSEBI_A0 は、CPU のアドレスが使用されます（CS[n]_N ごとに 128MB）。
- MSEBI_A31～MSEBI_A27 は、下記のレジスタによって指定されます。
  - rMSEBIM_CONFIG_CS[n]_N

DMA（n＝0、1）

- MSEBI_A31～MSEBI_A0 は、下記のレジスタによって指定されます。
  - rMSEBIM_ADDRDMA_CURRENTREAD_CS[n]_N（n＝0、1）
  - rMSEBIM_ADDRDMA_CURRENTWRITE_CS[n]_N（n＝0、1）

10.2　レジスタマップ

10.2.1　レジスタマップ：CPU から MSEBI マスタ

表 10.10　レジスタマップ：CPU から MSEBI マスタ

アドレス	レジスタシンボル	レジスタ名
400C 0000h＋100h×n	rMSEBIM_CYCLESIZE_CS[n]_N（n＝0～3）	チップセレクトサイクルサイズレジスタ
400C 0004h＋100h×n	rMSEBIM_SETUPHOLD_CS[n]_N（n＝0～3）	チップセレクトセットアップホールドレジスタ
400C 0008h＋100h×n	rMSEBIM_TDMACR_CS[n]_N（n＝0、1）	DMA 送信コントロール＆ステータスレジスタ
400C 000Ch＋100h×n	rMSEBIM_RDMACR_CS[n]_N（n＝0、1）	DMA 受信コントロール＆ステータスレジスタ
400C 0010h＋100h×n	rMSEBIM_ADDRDMA_READ_CS[n]_N（n＝0、1）	DMA 読み出しアドレスレジスタ
400C 0014h＋100h×n	rMSEBIM_ADDRDMA_CURRENTREAD_CS[n]_N（n＝0、1）	DMA カレント読み出しアドレスレジスタ
400C 0018h＋100h×n	rMSEBIM_ADDRDMA_WRITE_CS[n]_N（n＝0、1）	DMA 書き込みアドレスレジスタ
400C 001Ch＋100h×n	rMSEBIM_ADDRDMA_CURRENTWRITE_CS[n]_N（n＝0、1）	DMA カレント書き込みアドレスレジスタ
400C 0020h＋100h×n	rMSEBIM_DMATDLR_CS[n]_N（n＝0、1）	DMA 送信データレベルレジスタ
400C 0024h＋100h×n	rMSEBIM_DMARDLR_CS[n]_N（n＝0、1）	DMA 受信データレベルレジスタ
400C 0060h＋100h×n	rMSEBIM_CONFIG_CS[n]_N（n＝0～3）	チップセレクトコンフィグレジスタ
400C 0800h	rMSEBIM_CONFIG	コモンコンフィグレジスタ
400C 0808h	rMSEBIM_CPU_FIFOREAD_FLUSH	フラッシュ受信 FIFO レジスタ

10.2.2　レジスタマップ：DMA から MSEBI マスタ

表 10.11　レジスタマップ：DMA から MSEBI マスタ

アドレス	レジスタシンボル	レジスタ名
4008 0000h＋20000h×n	rMSEBIM_DMA_FIFOREAD_CS[n]_N（n＝0、1）	DMA 受信 FIFO（32KB）
4009 0000h＋20000h×n	rMSEBIM_DMA_FIFOWRITE_CS[n]_N（n＝0、1）	DMA 送信 FIFO（64KB）

10.2.3　レジスタマップ：CPU から MSEBI スレーブ

表 10.12　レジスタマップ：CPU から MSEBI スレーブ

アドレス	レジスタシンボル	レジスタ名
400C 2000h＋100h×n	rMSEBIS_CYCLESIZE_CS[n]_N（n＝0～3）	チップセレクトサイクルサイズレジスタ
400C 2004h＋100h×n	rMSEBIS_SETUPHOLD_CS[n]_N（n＝0～3）	チップセレクトセットアップホールドレジスタ
400C 2008h＋100h×n	rMSEBIS_MMU_ADDR_CS[n]_N（n＝0～3）	MMU ベースアドレスレジスタ
400C 200Ch＋100h×n	rMSEBIS_MMU_ADDR_MASK_CS[n]_N（n＝0～3）	MMU アドレスマスクレジスタ
400C 2010h＋100h×n	rMSEBIS_DMATX_REQ_CS[n]_N（n＝0、1）	DMA 送信要求レジスタ
400C 2014h＋100h×n	rMSEBIS_DMARX_REQ_CS[n]_N（n＝0、1）	DMA 受信要求レジスタ
400C 2018h＋100h×n	rMSEBIS_DMATDLR_CS[n]_N（n＝0、1）	DMA 送信データレベルレジスタ
400C 201Ch＋100h×n	rMSEBIS_DMARDLR_CS[n]_N（n＝0、1）	DMA 受信データレベルレジスタ
400C 2060h＋100h×n	rMSEBIS_CONFIG_CS[n]_N（n＝0～3）	チップセレクトコンフィグレジスタ
400C 2800h	rMSEBIS_CONFIG	コモンコンフィグレジスタ
400C 2804h	rMSEBIS_STATUS_INT0	割り込みステータスレジスタ
400C 2808h	rMSEBIS_STATUS_INT1	マスク後割り込みステータスレジスタ
400C 280Ch	rMSEBIS_MASK_INT	割り込みマスクレジスタ
400C 2810h	rMSEBIS_CLR_INT	割り込みクリアレジスタ
400C 2814h	rMSEBIS_EOB_ADDR	ブロック終了アドレスレジスタ

10.2.4　レジスタマップ：MSEBI から MSEBI スレーブ

表 10.13　レジスタマップ：MSEBI から MSEBI スレーブ

CS[n]注1	アドレス	レジスタシンボル	レジスタ名
CS0	400C 1000h	rMSEBIS_INT	スレーブ割り込みレジスタ
	400C 1004h	rMSEBIS_STATUS	スレーブステータスレジスタ
	400C 1008h＋4h×n	rMSEBIS_ID_CS[n]_N（n＝0～3）	スレーブ ID レジスタ
CS1	400C 1400h	rMSEBIS_INT	スレーブ割り込みレジスタ
	400C 1404h	rMSEBIS_STATUS	スレーブステータスレジスタ
	400C 1408h＋4h×n	rMSEBIS_ID_CS[n]_N（n＝0～3）	スレーブ ID レジスタ
CS2	400C 1800h	rMSEBIS_INT	スレーブ割り込みレジスタ
	400C 1804h	rMSEBIS_STATUS	スレーブステータスレジスタ
	400C 1808h＋4h×n	rMSEBIS_ID_CS[n]_N（n＝0～3）	スレーブ ID レジスタ
CS3	400C 1C00h	rMSEBIS_INT	スレーブ割り込みレジスタ
	400C 1C04h	rMSEBIS_STATUS	スレーブステータスレジスタ
	400C 1C08h＋4h×n	rMSEBIS_ID_CS[n]_N（n＝0～3）	スレーブ ID レジスタ

注1.　MSEBI マスタは、この表に応じて MSEBI_CS[n]_N（n＝0～3）により、各 MSEBI スレーブのレジスタにアクセスが可能です。

10.3　レジスタの説明

10.3.1　レジスタの説明：CPU から MSEBI マスタ

10.3.1.1　rMSEBIM_CYCLESIZE_CS[n]_N ― チップセレクトサイクルサイズレジスタ（n＝0～3）

注　意

設定を切り替える前に、CPU 受信 FIFO をフラッシュ（rMSEBIM_CPU_FIFOREAD_FLUSH レジスタにパターン 32'h0808 を CPU 書き込み）することによって、以前のアクセスが保留中でないようにする必要があります。すべての FIFO がエンプティ状態、かつ DMA コントローラは停止していなければいけません。

アドレス　400C 0000h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_WRDLEDATA_NB								bMSEBIM_RDDLEDATA_NB							
リセット後の値	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIM_WRDLEDATA_B		―	―	bMSEBIM_RDDLEDATA_B		―	―	―	―	―	―	bMSEBIM_CLEDATA	bMSEBIM_ALEDATA
リセット後の値	0	0	1	1	0	0	1	1	0	0	0	0	0	0	1	1

表 10.14　rMSEBIM_CYCLESIZE_CS[n]_N レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b24	bMSEBIM_WRDLEDATA_NB	非バーストモード時のラッチデータフェーズのサイズ（WRDLEDATA_NB） 以下の場合にのみ使用されます。 書き込みサイクル時 シングルアクセスまたはバーストサイクルの最初のアクセス（同期式モードのみ） MSEBIM_DLE が High である期間（MSEBIM_CLK） 8'h00：1 MSEBIM_CLK 8'h01：2 MSEBIM_CLK … … … 8'hFE：255 MSEBIM_CLK 8'hFF：256 MSEBIM_CLK 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b23～b16	bMSEBIM_RDDLEDATA_NB	非バーストモード時のラッチデータフェーズのサイズ（RDDLEDATA_NB） 以下の場合にのみ使用されます。 読み出しサイクル時 シングルアクセスまたはバーストサイクルの最初のアクセス（同期式モードのみ） MSEBIM_DLE が High である期間（MSEBIM_CLK） 8'h00：1 MSEBIM_CLK 8'h01：2 MSEBIM_CLK … … … 8'hFE：255 MSEBIM_CLK 8'hFF：256 MSEBIM_CLK	R/W
b15、b14	予約ビット	読むと 0 が読み出されます。	R

表 10.14　rMSEBIM_CYCLESIZE_CS[n]_N レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b13、b12	bMSEBIM_WRDLEDATA_B	バーストモード時のラッチデータフェーズのサイズ（WRDLEDATA_B） 以下の場合にのみ使用されます。 書き込みサイクル時 バーストが許可された同期式モードのみ MSEBIM_DLE が High である期間（MSEBIM_CLK） 2'b00：1 MSEBIM_CLK 2'b01：2 MSEBIM_CLK 2'b10：3 MSEBIM_CLK 2'b11：4 MSEBIM_CLK 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b11、b10	予約ビット	読むと 0 が読み出されます。	R
b9、b8	bMSEBIM_RDDLEDATA_B	バーストモード時のラッチデータフェーズのサイズ（RDDLEDATA_B） 以下の場合にのみ使用します。 読み出しサイクル時 バーストが許可された同期式モードのみ MSEBIM_DLE が High である期間（MSEBIM_CLK） 2'b00：1 MSEBIM_CLK 2'b01：2 MSEBIM_CLK 2'b10：3 MSEBIM_CLK 2'b11：4 MSEBIM_CLK	R/W
b7～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bMSEBIM_CLEDATA	コントロールラッチフェーズのサイズ（CLEDATA） MSEBIM_CLE フェーズの期間（MSEBIM_CLK）： 0：1 MSEBIM_CLK（1 MSEBIM_CLK の期間中 High） 1：2 MSEBIM_CLK（1 MSEBIM_CLK の期間中 High、その後 MSEBIM_CLK で 1 サイクルの期間中 Low） **注意**　非同期式モードでは、外部ラッチのホールド時間を確保するため、MSEBI_CLE フェーズの長さは 2 を使用してください。	R/W
b0	bMSEBIM_ALEDATA	アドレスラッチフェーズのサイズ（ALEDATA） MSEBIM_ALE フェーズの期間（MSEBIM_CLK）： 0：1 MSEBIM_CLK（1 MSEBIM_CLK の期間中 High） 1：2 MSEBIM_CLK（1 MSEBIM_CLK の期間中 High、その後 MSEBIM_CLK で 1 サイクルの期間中 Low） **注意**　ALE フェーズがない場合、本パラメータは使用されません。 非同期式モードでは、外部ラッチのホールド時間を確保するため、MSEBI_ALE フェーズの長さは 2 を使用してください。 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W

10.3.1.2　rMSEBIM_SETUPHOLD_CS[n]_N ― チップセレクトセットアップホールドレジスタ（n＝0～3）

注　意

設定を切り替える前に、CPU 受信 FIFO をフラッシュ（rMSEBIM_CPU_FIFOREAD_FLUSH レジスタにパターン 32'h0808 を CPU 書き込み）することによって、以前のアクセスが保留中でないようにする必要があります。すべての FIFO がエンプティ状態、かつ DMA コントローラは停止していなければいけません。

アドレス　400C 0004h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	bMSEBIM_WRDLEHOLD						―	―	bMSEBIM_RDDLEHOLD					
リセット後の値	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIM_WRDLESETUP						―	―	bMSEBIM_RDDLESETUP					
リセット後の値	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1

表 10.15　rMSEBIM_SETUPHOLD_CS[n]_N レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b30	予約ビット	読むと 0 が読み出されます。	R
b29～b24	bMSEBIM_WRDLEHOLD	ホールドデータフェーズのサイズ（WRDLEHOLD） 以下の場合にのみ使用されます。 書き込みサイクル時 ホールドフェーズの期間（MSEBIM_CLK） 6'h00：0 MSEBIM_CLK 6'h01：1 MSEBIM_CLK … … … 6'h3E：62 MSEBIM_CLK 6'h3F：63 MSEBIM_CLK 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b23～b22	予約ビット	読むと 0 が読み出されます。	R
b21～b16	bMSEBIM_RDDLEHOLD	ホールドデータフェーズのサイズ（RDDLEHOLD） 以下の場合にのみ使用されます。 読み出しサイクル時 ホールドフェーズの期間（MSEBIM_CLK） 6'h00：0 MSEBIM_CLK 6'h01：1 MSEBIM_CLK … … … 6'h3E：62 MSEBIM_CLK 6'h3F：63 MSEBIM_CLK **注意）** 周辺デバイスへの読み出しアクセスには（バス競合を避けるため）長さ“0”を使用しないでください。	R/W
b15、b14	予約ビット	読むと 0 が読み出されます。	R

表 10.15　rMSEBIM_SETUPHOLD_CS[n]_N レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b13～b8	bMSEBIM_WRDLESETUP	セットアップデータフェーズのサイズ（WRDLESETUP） 以下の場合にのみ使用されます。 ● 書き込みサイクル時 セットアップフェーズの期間（MSEBIM_CLK） 6’h00：0 MSEBIM_CLK 6’h01：1 MSEBIM_CLK … … … 6’h3E：62 MSEBIM_CLK 6’h3F：63 MSEBIM_CLK	R/W
b7、b6	予約ビット	読むと 0 が読み出されます。	R
b5～b0	bMSEBIM_RDDLESETUP	セットアップデータフェーズのサイズ（RDDLESETUP） 以下の場合にのみ使用されます。 ● 読み出しサイクル時 セットアップフェーズの期間（MSEBIM_CLK） 6’h00：0 MSEBIM_CLK 6’h01：1 MSEBIM_CLK … … … 6’h3E：62 MSEBIM_CLK 6’h3F：63 MSEBIM_CLK **注意）** 周辺デバイスへの読み出しアクセスには（バス競合を避けるため）長さ“0”を使用しないでください。 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W

10.3.1.3　rMSEBIM_TDMACR_CS[n]_N ― DMA 送信コントロール&ステータスレジスタ（n＝0、1）

アドレス　400C 0008h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	bMSEBIM_SINGLE_DEST_WIDTH	bMSEBIM_CURRENT_DEST_BLOCK_SIZE													bMSEBIM_DEST_BLOCK_SIZE
リセット後の値	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_DEST_BLOCK_SIZE												bMSEBIM_DEST_BURST_SIZE			bMSEBIM_TDMAE1
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.16　rMSEBIM_TDMACR_CS[n]_N レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31	予約ビット	読むと 0 が読み出されます。	R
b30	bMSEBIM_SINGLE_DEST_WIDTH	シングルトランザクションのサイズ 本値に従って DMAC.CTL[ch].DST_TR_WIDTH レジスタ（ch＝0～7）をプログラミングする必要があります。 1'b1：64 ビット幅 **注意）** MSEBI インタフェースを使用する場合、DMAC.CTL[ch].DST_TR_WIDTH フィールドに上記以外の値（8、16、32 ビットなど）をプログラミングしないでください。	R
b29～b17	bMSEBIM_CURRENT_DEST_BLOCK_SIZE	DEST_BLOCK_SIZE の現在値 いったん転送が始まると、bMSEBIM_CURRENT_DEST_BLOCK_SIZE ビットを読み出すことで、現在のブロック転送を完了させるために DMA 送信 FIFO に書き込まれるシングルトランザクションの総数が分かります。 ファームウェアが下記の動作を行うと、bMSEBIM_DEST_BLOCK_SIZE ビット値が bMSEBIM_CURRENT_DEST_BLOCK_SIZE ビットにリロードされます。 bMSEBIM_TDMAE1 ビットに“1'b1”をセット（立ち上がり）	R
b16～b4	bMSEBIM_DEST_BLOCK_SIZE	DEST_BLOCK_SIZE DMA 送信 FIFO の宛先ブロック転送サイズ MSEBI はフローコントローラです。 DMA モードが許可される前または許可されると同時に、本フィールドに書き込みを行う必要があります。DEST_BLOCK_SIZE にプログラミングする数値は、ブロック転送ごとに実行するシングルトランザクションの総数を意味します。シングルトランザクションのサイズは bMSEBIM_SINGLE_DEST_WIDTH ビットで決まります。 0：転送するシングルトランザクション数は 0、またはブロック転送終了 1：転送するシングルトランザクション数は 1 2：転送するシングルトランザクション数は 2 … … … 8191：転送するシングルトランザクション数は 8191 **備考）** 最大の要素数（最大 DEST_BLOCK_SIZE＝8191 のシングル要素）を転送できるようにするには、転送の開始前に、DMAC レジスタ DMAC.DAR[ch]（ch＝0～7）の宛先ポインタに FIFO のベースアドレスを設定する必要があります。	R/W

表 10.16　rMSEBIM_TDMACR_CS[n]_N レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3～b1	bMSEBIM_DEST_BURST_SIZE	DEST_BURST_SIZE DMA 送信 FIFO の宛先バーストトランザクションサイズ MSEBI はフローコントローラです。 DMA モードが許可される前または許可されると同時に、本フィールドに書き込みを行う必要があります。送信バーストトランザクション要求が生成されるたびに、DMA 送信 FIFO に書き込むシングルトランザクション数を指定します。 3'b000：シングルトランザクション数は 1 3'b001：シングルトランザクション数は 4（推奨値） 3'b010：シングルトランザクション数は 8 3'b011：シングルトランザクション数は 16 3'b100：シングルトランザクション数は 32 3'b101：予約（使用しない） 3'b11x：予約（使用しない）	R/W
b0	bMSEBIM_TDMAE1	送信 DMA 許可／禁止 0：送信モードでの DMA を禁止 1：送信モードでの DMA を許可 bMSEBIM_TDMAE1 ビットの立ち上がりで、DMA 送信 FIFO をフラッシュします。 送信 FIFO の最後のトランザクションが完了し（DEST_BLOCK_SIZE 分のシングルトランザクションが DMA 送信 FIFO に書き込まれ）、さらに DMA 送信 FIFO 内の全データが MSEBI バスに出力されると（FIFO がエンプティになると）、bMSEBIM_TDMAE1 ビットがハードウェアによって自動的にクリアされて、送信モードでの DMA が禁止されます。そのため、ソフトウェアで本ビットをポーリングすれば、いつ当該チャネルが解放されて次の DMA 転送が可能になるかを判定できます。 **注意）** ● DMA が外部端子 MSEBIM_DMA_WR0_N または MSEBIM_DMA_WR1_N（使用する MSEBI_CS_N に依存）によって制御される場合： 本ビットを「許可」に設定する前に、ソフトウェアで rMSEBIM_DMATDLR_CS0_N または rMSEBIM_DMATDLR_CS1_N レジスタ（使用する MSEBI_CS_N に依存）の bMSEBIM_USE_EXT_WRDMA_REQ ビットを「許可」に設定して、外部端子による DMA モードの制御を許可する必要があります。 それ以降、DMA モードは bMSEBIM_TDMAE1 ビットと外部端子 MSEBIM_DMA_WR0_N または MSEBIM_DMA_WR1_N（使用する MSEBI_CS_N に依存）によって制御されます。 「**図 10.53 MSEBI マスタ：外部 DMA 要求**」を参照してください。 ● DMA が外部端子 MSEBIM_DMA_WR0_N または MSEBIM_DMA_WR1_N によって制御されない場合： 本ビットを「許可」に設定する前に、ソフトウェアで rMSEBIM_DMATDLR_CS0_N または rMSEBIM_DMATDLR_CS1_N レジスタ（使用する MSEBI_CS_N に依存）の bMSEBIM_USE_EXT_WRDMA_REQ ビットを「禁止」に設定する必要があります。 それ以降、DMA モードは bMSEBIM_TDMAE1 ビットのみによって制御されます。 ● DMA 転送中に本ビットがクリアされると、DMA モードが停止する前に現在の転送（バースト転送またはシングル転送）が終了します。 bMSEBIM_CURRENT_DEST_BLOCK_SIZE ビット値は、現在の転送が終了した場合にのみ矛盾のない値になります。 DMA ブロック転送を完了させるには： – DMA 送信 FIFO がエンプティであることを（bMSEBIM_DMA_TRANSMIT_FIFOLEVEL ビットで）確認します。 – bMSEBIM_ADDRDMA_WRITE ビットに、bMSEBIM_ADDRDMA_CURRENTWRITE ビット値をリロードします。 – bMSEBIM_DEST_BLOCK_SIZE ビットに、bMSEBIM_CURRENT_DEST_BLOCK_SIZE ビット値をリロードします。 – 必要に応じて、bMSEBIM_DEST_BURST_SIZE ビットを更新します。 – bMSEBIM_TDMAE1 ビットを 1 にします。	R/W

10.3.1.4　rMSEBIM_RDMACR_CS[n]_N ― DMA 受信コントロール&ステータスレジスタ（n＝0、1）

アドレス　400C 000Ch＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	bMSEBIM_SINGLE_SRC_WIDTH	bMSEBIM_CURRENT_SRC_BLOCK_SIZE													bMSEBIM_SRC_BLOCK_SIZE
リセット後の値	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_SRC_BLOCK_SIZE												bMSEBIM_SRC_BURST_SIZE			bMSEBIM_RDMAE1
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.17　rMSEBIM_RDMACR_CS[n]_N レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31	予約ビット	読むと 0 が読み出されます。	R
b30	bMSEBIM_SINGLE_SRC_WIDTH	シングルトランザクションのサイズ 本値に従って DMAC.CTL[ch].SRC_TR_WIDTH レジスタ（ch＝0～7）をプログラミングする必要があります。 1'b1：64 ビット幅 **注意）** MSEBI インタフェースを使用する場合、DMAC.CTL[ch].SRC_TR_WIDTH フィールドに上記以外の値（8、16、32 ビットなど）をプログラミングしないでください。	R
b29～b17	bMSEBIM_CURRENT_SRC_BLOCK_SIZE	SRC_BLOCK_SIZE の現在値 いったん転送が始まると、bMSEBIM_CURRENT_SRC_BLOCK_SIZE ビットを読み出すことで、現在のブロック転送を完了させるために DMA 受信 FIFO を読み出すシングルトランザクションの総数が分かります。 ファームウェアが下記の動作を行うと、bMSEBIM_SRC_BLOCK_SIZE ビット値が bMSEBIM_CURRENT_SRC_BLOCK_SIZE ビットにリロードされます。 bMSEBIM_RDMAE1 ビットに“1”をセット（立ち上がり）	R
b16～b4	bMSEBIM_SRC_BLOCK_SIZE	SRC_BLOCK_SIZE DMA 受信 FIFO の送信元ブロックサイズ MSEBI はフローコントローラです。 DMA モードが許可される前または許可されると同時に、本フィールドに書き込みを行う必要があります。SRC_BLOCK_SIZE にプログラミングする数値は、ブロック転送ごとに実行するシングルトランザクションの総数を意味します。シングルトランザクションのサイズは bMSEBIM_SINGLE_SRC_WIDTH ビットで決まります。 0：転送するシングルトランザクション数は 0、またはブロック転送終了 1：転送するシングルトランザクション数は 1 2：転送するシングルトランザクション数は 2 … … … 8191：転送するシングルトランザクション数は 8191 **備考）** 最大の要素数（最大 DEST_BLOCK_SIZE＝8191 のシングル要素）を転送できるようにするには、転送の開始前に、DMAC レジスタ DMAC.SAR[ch]（ch＝0～7）の送信元ポインタに FIFO のベースアドレスを設定する必要があります。	R/W

表 10.17　rMSEBIM_RDMACR_CS[n]_N レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3～b1	bMSEBIM_SRC_BURST_SIZE	SRC_BURST_SIZE DMA 受信 FIFO の送信元バーストトランザクションサイズ MSEBI はフローコントローラです。 DMA モードが許可される前または許可されると同時に、本フィールドに書き込みを行う必要があります。CS0_N または CS1_N に対して、n＝0 または n＝2 で受信バーストトランザクション要求が生成されるたびに、DMA 受信 FIFO を読み出すシングルトランザクション数を指定します。 3’b000：シングルトランザクション数は 1 3’b001：シングルトランザクション数は 4（推奨値） 3’b010：シングルトランザクション数は 8 3’b011：シングルトランザクション数は 16 3’b100：シングルトランザクション数は 32 3’b101：予約（使用しない） 3’b11x：予約（使用しない）	R/W
b0	bMSEBIM_RDMAE1	受信 DMA 許可／禁止 0：受信モードでの DMA を禁止 1：受信モードでの DMA を許可 bMSEBIM_RDMAE1 ビットの立ち上がりは、DMA 受信 FIFO をフラッシュします。 DMA 受信 FIFO からの最後のトランザクションが完了すると（SRC_BLOCK_SIZE 分のシングルトランザクションが DMA 受信 FIFO から読み出されると）、bMSEBIM_RDMAE1 ビットがハードウェアによって自動的にクリアされて、受信モードでの DMA が禁止されます。DMA コントローラが停止して、DMA 受信 FIFO がフラッシュされます。そのため、ソフトウェアで本ビットをポーリングすれば、いつ当該チャネルが解放されて次の DMA 転送が可能になるかを判定できます。 **注意）** ● DMA が外部端子 MSEBIM_DMA_RD0_N または MSEBIM_DMA_RD1_N（使用する MSEBI_CS_N に依存）によって制御される場合：本ビットを「許可」に設定する前に、ソフトウェアで rMSEBIM_DMARDLR_CS0_N または rMSEBIM_DMARDLR_CS1_N レジスタ（使用する MSEBI_CS_N に依存）の bMSEBIM_USE_EXT_RDDMA_REQ ビットを「許可」に設定して、外部端子による DMA モードの制御を許可する必要があります。それ以降、DMA モードは bMSEBIM_RDMAE1 ビットと外部端子 MSEBIM_DMA_RD0_N または MSEBIM_DMA_RD1_N（使用する MSEBI_CS_N に依存）によって制御されます。 「**図 10.53 MSEBI マスタ：外部 DMA 要求**」を参照してください。 ● DMA が外部端子 MSEBIM_DMA_RD0_N または MSEBIM_DMA_RD1_N によって制御されない場合：本ビットを「許可」に設定する前に、ソフトウェアで rMSEBIM_DMARDLR_CS0_N または rMSEBIM_DMARDLR_CS1_N レジスタ（使用する MSEBI_CS_N に依存）の bMSEBIM_USE_EXT_RDDMA_REQ ビットを「禁止」に設定する必要があります。それ以降、DMA モードは bMSEBIM_RDMAE1 ビットのみによって制御されます。 ● DMA 転送中に本ビットがクリアされると、DMA モードが停止する前に現在の転送（バースト転送またはシングル転送）が終了します。bMSEBIM_CURRENT_SRC_BLOCK_SIZE ビット値は、現在の転送が終了した場合にのみ矛盾のない値になります。 DMA ブロック転送を完了させるには： – DMA 受信 FIFO がエンプティであることを（bMSEBIM_DMA_RECEIVE_FIFOLEVEL ビットで）確認します。 – bMSEBIM_ADDRDMA_READ ビットに、bMSEBIM_ADDRDMA_CURRENTREAD ビット値をリロードします。 – bMSEBIM_SRC_BLOCK_SIZE ビットに、bMSEBIM_CURRENT_SRC_BLOCK_SIZE ビット値をリロードします。 – 必要に応じて、bMSEBIM_SRC_BURST_SIZE ビットを更新します。 – bMSEBIM_RDMAE1 ビットを 1 にします。	R/W

10.3.1.5　rMSEBIM_ADDRDMA_READ_CS[n]_N ― DMA 読み出しアドレスレジスタ（n＝0、1）

アドレス　400C 0010h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_ADDRDMA_READ															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_ADDRDMA_READ													bMSEBIM_ADDRDMA_READ_2	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.18　rMSEBIM_ADDRDMA_READ_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b3	bMSEBIM_ADDRDMA_READ	DMA 読み出しアクセス用アドレス これは、ファームウェアが bMSEBIM_RDMAE1 ビットに“1”をセット（立ち上がり）したとき、MSEBI バスから DMA 受信 FIFO へ DMA 転送を開始するため、DMA コントローラによって使用される先頭ブロックアドレスです。 **「図 10.49 MSEBI：バーストモード、DMA 受信 FIFO とバスインタフェースの結合」**を参照してください。	R/W
b2	bMSEBIM_ADDRDMA_READ_2	本ビットは先頭ブロックアドレスの定義で無視されます。	R/W
b1、b0	予約ビット	これらのビット[1:0]は 0 にリセットされます。	R

10.3.1.6　rMSEBIM_ADDRDMA_CURRENTREAD_CS[n]_N ― DMA カレント読み出しアドレスレジスタ（n＝0、1）

アドレス　400C 0014h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_ADDRDMA_CURRENTREAD															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_ADDRDMA_CURRENTREAD															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.19　rMSEBIM_ADDRDMA_CURRENTREAD_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bMSEBIM_ADDRDMA_CURRENTREAD	DMA 読み出しアクセス用カレントアドレス これは、MSEBI バスから DMA 受信 FIFO へデータを読み出すために、DMA コントローラによって使用されるアドレスです。MSEBI アクセスの終了時、本値はバス上に読み出したパックデータのサイズで更新されます。 **「図 10.49 MSEBI：バーストモード、DMA 受信 FIFO とバスインタフェースの結合」**を参照してください。 ファームウェアが bMSEBIM_RDMAE1 ビットに“1”をセット（立ち上がり）したとき、bMSEBIM_ADDRDMA_READ ビット値が bMSEBIM_ADDRDMA_CURRENTREAD ビットにリロードされます。リロード時に下記の動作が行われます。 ● bMSEBIM_SINGLE_SRC_WIDTH＝1 であると、ビット[2:0]が 0 にリセットされます（64 ビットアライメント）。	R

10.3.1.7　rMSEBIM_ADDRDMA_WRITE_CS[n]_N ― DMA 書き込みアドレスレジスタ（n＝0、1）

アドレス　400C 0018h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_ADDRDMA_WRITE															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_ADDRDMA_WRITE													bMSEBIM_ADDRDMA_WRITE_2	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.20　rMSEBIM_ADDRDMA_WRITE_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b3	bMSEBIM_ADDRDMA_WRITE	DMA 書き込みアクセス用アドレス これは、ファームウェアが bMSEBIM_TDMAE1 ビットに“1”をセット（立ち上がり）したとき、DMA 送信 FIFO から MSEBI バスへ DMA 転送を開始するため、DMA コントローラによって使用される先頭ブロックアドレスです。 ビット[1:0]は 0 にリセットされます（32 ビットアライメント）。 **「図 10.48 MSEBI：バーストモード、DMA 送信 FIFO とバスインタフェースの結合」**を参照してください。	R/W
b2	bMSEBIM_ADDRDMA_WRITE_2	bMSEBIM_SINGLE_DEST_WIDTH ビットに依存します。 ● bMSEBIM_SINGLE_DEST_WIDTH＝1（64 ビットシングルトランザクション）の場合、本ビットは先頭ブロックアドレスの定義で無視されます（64 ビットアライメント）。	R/W
b1、b0	予約ビット	これらのビット[1:0]は 0 にリセットされます（32 ビットアライメント）。	R

10.3.1.8　rMSEBIM_ADDRDMA_CURRENTWRITE_CS[n]_N ― DMA カレント書き込みアドレスレジスタ（n＝0、1）

アドレス　400C 001Ch＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_ADDRDMA_CURRENTWRITE															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_ADDRDMA_CURRENTWRITE															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.21　rMSEBIM_ADDRDMA_CURRENTWRITE_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bMSEBIM_ADDRDMA_CURRENTWRITE	DMA 書き込みアクセス用カレントアドレス これは、DMA 送信 FIFO から MSEBI バスへデータを書き込むために、DMA コントローラによって使用されるアドレスです。 MSEBI アクセスの終了時、本値はバス上に書き込まれたパックデータのサイズで更新されます。 **「図 10.48 MSEBI：バーストモード、DMA 送信 FIFO とバスインタフェースの結合」**を参照してください。 ファームウェアが bMSEBIM_TDMAE1 ビットに“1”をセット（立ち上がり）したとき、bMSEBIM_ADDRDMA_WRITE ビット値が bMSEBIM_ADDRDMA_CURRENTWRITE ビットにリロードされます。リロード時に下記の動作が行われます。 ● bMSEBIM_SINGLE_DEST_WIDTH＝1 であると、ビット[2:0]が 0 にリセットされます（64 ビットアライメント）。	R

10.3.1.9　rMSEBIM_DMATDLR_CS[n]_N ― DMA 送信データレベルレジスタ（n＝0、1）

注　意

設定を切り替える前に、CPU 受信 FIFO をフラッシュ（rMSEBIM_CPU_FIFOREAD_FLUSH レジスタにパターン 32'h0808 を CPU 書き込み）することによって、以前のアクセスが保留中でないようにする必要があります。すべての FIFO がエンプティ状態、かつ DMA コントローラは停止していなければいけません。

アドレス　400C 0020h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bMSEBIM_USE_EXT_WRDMA_REQ
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_BURST_SIZEMAX_DMAWRITE			―	bMSEBIM_DMA_TRANSMIT_FIFOLEVEL						―	bMSEBIM_DMATDLR				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.22　rMSEBIM_DMATDLR_CS[n]_N レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31	予約ビット	初期値を保持	R/W
b30～b17	予約ビット	読むと 0 が読み出されます。	R
b16	bMSEBIM_USE_EXT_WRDMA_REQ	送信モード時、外部端子 MSEBIM_DMA_WR0_N または MSEBIM_DMA_WR1_N（使用する CS_N に依存）による DMA チャネル制御を許可します。本モードでは、DMA モードは同時に bMSEBIM_TDMAE1 ビットによっても制御されます。 送信 DMA モード許可／禁止 1'b0：外部端子による DMA 制御を禁止。DMA 転送は、bMSEBIM_TDMAE1 ビットの立ち上がり時に即座に開始 1'b1：外部端子 MSEBIM_DMA_WR0_N または MSEBIM_DMA_WR1_N（使用する CS_N に依存）による DMA 制御を許可。DMA 転送は、bMSEBIM_TDMAE1 ビットの立ち上がり検出後に外部 DMA 要求が 0 になったとき開始 「**図 10.53　MSEBI マスタ：外部 DMA 要求**」を参照してください。	R/W
b15～b13	bMSEBIM_BURST_SIZEMAX_DMAWRITE	すべての CS[n]_N（n＝0、1）に対して DMA 送信 FIFO から MSEBI バスへの書き込みアクセス時に許容される最大バーストサイズ ラウンドロビンアービタによって使用されます。 「**図 10.45　MSEBI：ラウンドロビン優先順位**」を参照してください。 3'b000：1 ワード 3'b001：2 ワード 3'b010：4 ワード 3'b011：8 ワード 3'b100：16 ワード 3'b101：無制限 3'b11x：予約 1 ワードは、モード 32 デバイスでは 32 ビット幅、モード 16 デバイスでは 16 ビット幅、モード 8 デバイスでは 8 ビット幅です。	R/W
b12	予約ビット	読むと 0 が読み出されます。	R

表 10.22　rMSEBIM_DMATDLR_CS[n]_N レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b11～b6	bMSEBIM_DMA_TRANSMIT_FIFOLEVEL	DMA 送信 FIFO レベル DMA 送信 FIFO 内にある有効データエントリの数が格納されます。 6'd0：データエントリ数 0（DMA 送信 FIFO はエンプティ） 6'd1：データエントリ数 1、またはアクティビティがハンドシェイク上および／または MSEBI バス上に存在 6'd2：データエントリ数 2 … … … 6'd32：データエントリ数 32（DMA 送信 FIFO はフル） **備考）** 1 データエントリ＝1 ワード（64 ビット）	R
b5	予約ビット	読むと 0 が読み出されます。	R
b4～b0	bMSEBIM_DMATDLR	DMA 送信 FIFO データレベル 本ビットフィールドは、DMA 要求が送信ロジックによって生成されるときのレベルを制御します。これはウォーターマークレベルに一致します。すなわち、DMA 送信 FIFO 内にある有効データエントリの数が本ビットフィールド値以下であり、かつ bMSEBIM_TDMAE1＝1 であると、DMA 要求信号が発生します。 フィールドデコードについては、下記を参照してください。 6'd0：DMA 送信 FIFO 内にデータエントリが存在しない場合、DMA 要求がアサートされる 6'd1：DMA 送信 FIFO 内に 1 個以下のデータエントリが存在する場合、DMA 要求がアサートされる … … … 6'd31：DMA 送信 FIFO 内に 31 個以下のデータエントリが存在する場合、DMA 要求がアサートされる 最適動作のための推奨値は以下のとおりです（ch＝0～7）。 DMAC.CTL[ch].DST_TR_WIDTH = 3（64 ビット） DMAC.CTL[ch].DEST_MSIZE = 1（シングルトランザクション数は 4） bMSEBIM_SINGLE_DEST_WIDTH = 1（64 ビット） bMSEBIM_DEST_BURST_SIZE = 1（シングルトランザクション数は 4） bMSEBIM_DMATDLR = 28 **「図 10.48 MSEBI：バーストモード、DMA 送信 FIFO とバスインタフェースの結合」**を参照してください。	R/W

10.3.1.10　rMSEBIM_DMARDLR_CS[n]_N ― DMA 受信データレベルレジスタ（n＝0、1）

注　意

設定を切り替える前に、CPU 受信 FIFO をフラッシュ（rMSEBIM_CPU_FIFOREAD_FLUSH レジスタにパターン 32'h0808 を CPU 書き込み）することによって、以前のアクセスが保留中でないようにする必要があります。すべての FIFO がエンプティ状態、かつ DMA コントローラは停止していなければいけません。

アドレス　400C 0024h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bMSEBIM_USE_EXT_RDDMA_REQ
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_BURST_SIZEMAX_DMAREAD			―	bMSEBIM_DMA_RECEIVE_FIFOLEVEL						―	bMSEBIM_DMARDLR				
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.23　rMSEBIM_DMARDLR_CS[n]_N レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b30	予約ビット	初期値を保持	R/W
b29～b17	予約ビット	読むと 0 が読み出されます。	R
b16	bMSEBIM_USE_EXT_RDDMA_REQ	受信モード時、外部端子 MSEBIM_DMA_RD0_N または MSEBIM_DMA_RD1_N（使用する CS_N に依存）による DMA チャネル制御を許可します。本モードでは、DMA モードは同時に bMSEBIM_RDMAE1 ビットによっても制御されます。 受信 DMA モード許可／禁止 1'b0：外部端子による DMA 制御を禁止。DMA 転送は、bMSEBIM_RDMAE1 ビットの立ち上がり時に即座に開始 1'b1：外部端子 MSEBIM_DMA_RD0_N または MSEBIM_DMA_RD1_N（使用する CS_N に依存）による DMA 制御を許可。DMA 転送は、bMSEBIM_RDMAE1 ビットの立ち上がり検出後に外部 DMA 要求が 0 になったとき開始 「**図 10.53　MSEBI マスタ：外部 DMA 要求**」を参照してください。	R/W
b15～b13	bMSEBIM_BURST_SIZEMAX_DMAREAD	すべての CS[n]_N（n＝0、1）に対して MSEBI バスから DMA 受信 FIFO への読み出しアクセス時に許容される最大バーストサイズ。ラウンドロビンアービタによって使用されます。 「**図 10.45　MSEBI：ラウンドロビン優先順位**」を参照してください。 3'b000：1 ワード 3'b001：2 ワード 3'b010：4 ワード 3'b011：8 ワード 3'b100：16 ワード 3'b101：無制限 3'b11x：予約 1 ワードは、モード 32 デバイスでは 32 ビット幅、モード 16 デバイスでは 16 ビット幅、モード 8 デバイスでは 8 ビット幅です。	R/W
b12	予約ビット	読むと 0 が読み出されます。	R

表 10.23　rMSEBIM_DMARDLR_CS[n]_N レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b11～b6	bMSEBIM_DMA_RECEIVE_FIFOLEVEL	DMA 受信 FIFO レベル DMA 受信 FIFO 内にある有効データエントリの数が格納されます。 6'd0：データエントリ数 0（DMA 受信 FIFO はエンプティ） 6'd1：データエントリ数 1、またはアクティビティがハンドシェイク上および／または MSEBI バス上に存在 6'd2：データエントリ数 2 … … … 6'd32：データエントリ数 32（DMA 受信 FIFO はフル） ここで、1 データエントリ＝1 ワード（64 ビット）	R
b5	予約ビット	読むと 0 が読み出されます。	R
b4～b0	bMSEBIM_DMARDLR	DMA 受信 FIFO データレベル 本ビットフィールドは、DMA 要求が受信ロジックによって生成されるときのレベルを制御します。ウォーターマークレベル＝bMSEBIM_DMARDLR＋1 です。すなわち、DMA 受信 FIFO 内にある有効データエントリの数が本ビットフィールド値＋1 以上であり、かつ bMSEBIM_RDMAE1＝1 であると、DMA 要求が発生します。 フィールドデコードについては、下記を参照してください。 6'd0：受信 FIFO 内に 1 個以上のデータエントリが存在する場合、DMA 要求がアサートされる 6'd1：受信 FIFO 内に 2 個以上のデータエントリが存在する場合、DMA 要求がアサートされる … … … 6'd31：受信 FIFO 内に 32 個以上のデータエントリが存在する場合、DMA 要求がアサートされる 最適動作のための推奨値は以下のとおりです（ch＝0～7）。 DMAC.CTL[ch].SRC_TR_WIDTH　= 3（64 ビット） DMAC.CTL[ch].SRC_MSIZE　= 1（シングルトランザクション数は 4） bMSEBIM_SINGLE_SRC_WIDTH　= 1（64 ビット） bMSEBIM_SRC_BURST_SIZE　= 1（シングルトランザクション数は 4） bMSEBIM_DMARDLR　= 3 **「図 10.49 MSEBI：バーストモード、DMA 受信 FIFO とバスインタフェースの結合」**を参照してください。	R/W

10.3.1.11　rMSEBIM_CONFIG_CS[n]_N ― チップセレクトコンフィグレジスタ（n＝0～3）

注　意

設定を切り替える前に、CPU 受信 FIFO をフラッシュ（rMSEBIM_CPU_FIFOREAD_FLUSH レジスタにパターン 32'h0808 を CPU 書き込み）することによって、以前のアクセスが保留中でないようにする必要があります。すべての FIFO がエンプティ状態、かつ DMA コントローラは停止していなければいけません。

アドレス　400C 0060h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_EXTEND_ADDR					—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_MULTI_DLE	bMSEBIM_CS[n]N_ROUTING_CS3_N	bMSEBIM_CS[n]N_ROUTING_CS2_N	bMSEBIM_CS0N_ROUTING_CS1_N	bMSEBIM_ALE_MODE	bMSEBIM_ALE_NUMBER			bMSEBIM_BURST_ENABLE	bMSEBIM_MODE_WAIT		—	—	bMSEBIM_CONFIG		
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.24　rMSEBIM_CONFIG_CS[n]_N レジスタの内容（1/4）

ビット位置	ビット名	機能	R/W
b31～b27	bMSEBIM_EXTEND_ADDR	ユーザの使用例に合わせて、レジスタによって MSEBI_A27 から最大 MSEBI_A31 までアドレス機能を拡張します。 以下の割り当てビットが使用されます。 ビット 31：MSEBI_A31 ビット 30：MSEBI_A30 ビット 29：MSEBI_A29 ビット 28：MSEBI_A28 ビット 27：MSEBI_A27 「**10.1.3 マルチプレクス信号インタフェース**」を参照してください。	R/W
b26～b16	予約ビット	読むと 0 が読み出されます。	R
b15	bMSEBIM_MULTI_DLE	マルチ DLE モードを使用すると、FPGA のコンフィグレーションを高速かつ少ない端子数で実施できるようになります。 書き込みバーストアクセス中に、MSEBI_DLE は各データ有効信号の立ち上がりを提示することで、外部 FPGA のコンフィグレーション方式を可能にします。 1'b0：マルチ DLE モードを禁止 1'b1：マルチ DLE モードを許可 **注意）** マルチ DLE モードでは、書き込みアクセスのみがサポートされています。ウェイト信号は無視されます。 「**10.4.6.2(3) マスタのマルチ DLE**」を参照してください。	R/W

表 10.24　rMSEBIM_CONFIG_CS[n]_N レジスタの内容（2/4）

ビット位置	ビット名	機能	R/W
b14	bMSEBIM_CS[n]N_ROUTING_CS3_N	チップセレクト MSEBI_CS0_N、MSEBI_CS1_N、MSEBI_CS2_N によるアクセス中に、MSEBI_CS3_N をアドレスビットとして使用することにより、ユーザの使用例に合わせてアドレス機能を拡張できます。アドレスビットの割り当ては、ALE フェーズの数に依存します。 これらのビットは下記レジスタでは使用されないため、読むと 0 が読み出されます。 rMSEBIM_CONFIG_CS3_N ビットは以下のように管理されます。 1'b0：本ライン上ではアドレスルーティングなし 1'b1：本ライン上でアドレスルーティングを許可 「**10.1.3 マルチプレクス信号インタフェース**」を参照してください。	R/W
b13	bMSEBIM_CS[n]N_ROUTING_CS2_N	チップセレクト MSEBI_CS0_N、MSEBI_CS1_N によるアクセス中に、MSEBI_CS2_N をアドレスビットとして使用することにより、ユーザの使用例に合わせてアドレス機能を拡張できます。アドレスビットの割り当ては、ALE フェーズの数に依存します。 これらのビットは下記レジスタでは使用されないため、読むと 0 が読み出されます。 rMSEBIM_CONFIG_CS2_N rMSEBIM_CONFIG_CS3_N ビットは以下のように管理されます。 1'b0：本ライン上ではアドレスルーティングなし 1'b1：本ライン上でアドレスルーティングを許可 「**10.1.3 マルチプレクス信号インタフェース**」を参照してください。	R/W
b12	bMSEBIM_CS0N_ROUTING_CS1_N	チップセレクト MSEBI_CS0_N によるアクセス中に、MSEBI_CS1_N をアドレスビットとして使用することにより、ユーザの使用例に合わせてアドレス機能を拡張できます。アドレスビットの割り当ては、ALE フェーズの数に依存します。 これらのビットは下記レジスタでは使用されないため、読むと 0 が読み出されます rMSEBIM_CONFIG_CS1_N rMSEBIM_CONFIG_CS2_N rMSEBIM_CONFIG_CS3_N ビットは以下のように管理されます。 1'b0：本ライン上ではアドレスルーティングなし 1'b1：本ライン上でアドレスルーティングを許可 「**10.1.3 マルチプレクス信号インタフェース**」を参照してください。	R/W

表 10.24　rMSEBIM_CONFIG_CS[n]_N レジスタの内容（3/4）

ビット位置	ビット名	機能	R/W
b11	bMSEBIM_ALE_MODE	MSEBI_ALE はシリアルモードまたはパラレルモードで管理されます。 ● 1'b0：シリアルモード – 全 ALE サイクルで MSEBIM_ALE のみを使用 – 同期式インタフェースではシリアルモードを推奨 ● 1'b1：パラレルモード – 非同期式インタフェースではパラレルモードを推奨。この場合、外部ラッチ（74x16373 タイプ）の直結が可能となるため、ボード上の個別部品のコストを削減できます。 – モード 32 ALE サイクルでは MSEBIM_ALE を使用 – モード 16 最初の ALE サイクルでは MSEBIM_ALE を使用 2 番目の ALE サイクルでは MSEBIM_ALE1 を使用 – モード 8 最初の ALE サイクルでは MSEBIM_ALE を使用 2 番目の ALE サイクルでは MSEBIM_ALE1 を使用 3 番目の ALE サイクルでは MSEBIM_ALE2 を使用 4 番目の ALE サイクルでは MSEBIM_ALE3 を使用 「**10.1.3 マルチプレクス信号インタフェース**」および「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b10～b8	bMSEBIM_ALE_NUMBER	周辺デバイスのアドレス指定に使用される MSEBI_ALE フェーズの数 3'b000：0 MSEBI_ALE を使用 3'b001：1 MSEBI_ALE を使用 3'b010：2 MSEBI_ALE を使用 3'b011：3 MSEBI_ALE を使用 3'b100：4 MSEBI_ALE を使用 3'b101：予約 3'b11x：予約 「**10.1.3 マルチプレクス信号インタフェース**」および「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b7	bMSEBIM_BURST_ENABLE	読み出し／書き込みアクセス時のバーストモードを許可します。 バーストモードは同期式モードでのみ利用可能です。 非同期式モードでは、本ビットは無視されます。 バーストを許可する場合 1'b0：バーストを禁止（シングルアクセス） 1'b1：バーストを許可（シングルおよびバーストアクセス） **注意**　マスタがスレーブの共有レジスタに（MSEBI_CSREG_N を用いて）アクセスしているときは、プリフェッチを行わないでください。 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b6、b5	bMSEBIM_MODE_WAIT	各 MSEBI_CS[n]_N（n＝0～3）に対して MSEBI インタフェースは下記の 3 つの基本機能に設定できます。 2'b00：MSEBIM_WAIT[n]_N 端子に対するウェイト管理なし。専用の外部端子 MSEBIM_WAIT[n]_N は使用されない 2'b01：MSEBIM_WAIT[n]_N 端子に対するウェイト管理。専用の外部端子 MSEBIM_WAIT[n]_N が監視および管理される。 2'b10：MSEBIM_WAIT0_N 端子に対するウェイト管理。選択した MSEBI_CS[n]_N に対して、1 つの外部共通端子 MSEBIM_WAIT0_N のみが監視および管理される。 2'b11：予約 各 MSEBI_CS[n]_N に対して、端子 MSEBIM_WAIT[n]_N が、bMSEBIM_CONFIG ビットの設定に応じて、同期式モードまたは非同期式モードで管理されます。 2'b10 のモードでは、すべての MSEBI_CS[n]_N に対して、1 つの外部端子 MSEBIM_WAIT0_N だけが使用可能です。この場合、使用する外部端子の本数が削減されます。 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W

表 10.24　rMSEBIM_CONFIG_CS[n]_N レジスタの内容（4/4）

ビット位置	ビット名	機能	R/W
b4、b3	予約ビット	初期値を保持	R/W
b2～b0	bMSEBIM_CONFIG	MSEBI インタフェースは下記の 6 つの基本機能に設定できます。 3'b000：非同期式、16 ビット、マルチプレクス、モード 16、非バースト 3'b001：同期式、16 ビット、マルチプレクス、モード 16、バースト可 3'b010：非同期式、32 ビット、マルチプレクス、モード 32、非バースト 3'b011：同期式、32 ビット、マルチプレクス、モード 32、バースト可 3'b100：非同期式、8 ビット、マルチプレクス、モード 8、非バースト 3'b101：同期式、8 ビット、マルチプレクス、モード 8、バースト可 3'b110：予約 3'b111：予約 「**10.1.3 マルチプレクス信号インタフェース**」および「**10.4.4 MSEBI タイミング**」を参照してください。	R/W

10.3.1.12　rMSEBIM_CONFIG ― コモンコンフィグレジスタ

注　意

設定を切り替える前に、CPU 受信 FIFO をフラッシュ（rMSEBIM_CPU_FIFOREAD_FLUSH レジスタにパターン 32'h0808 を CPU 書き込み）することによって、以前のアクセスが保留中でないようにする必要があります。すべての FIFO がエンプティ状態、かつ DMA コントローラは停止していなければいけません。

アドレス　400C 0800h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	bMSEBIM_CPU_RECEIVE_FIFOLEVEL						bMSEBIM_CPU_TRANSMIT_FIFOLEVEL
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_CPU_TRANSMIT_FIFOLEVEL					bMSEBIM_BURST_SIZEMAX_CPUREAD			bMSEBIM_BURST_SIZEMAX_CPUWRITE			bMSEBIM_CLKENABLE	bMSEBIM_CLKH		bMSEBIM_CLKL	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

表 10.25　rMSEBIM_CONFIG レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b30	予約ビット	初期値を保持	R/W
b29～b23	予約ビット	読むと 0 が読み出されます。	R
b22～b17	bMSEBIM_CPU_RECEIVE_FIFOLEVEL	CPU 受信 FIFO レベル CPU 受信 FIFO 内にある有効データエントリの数が格納されます。 6'd0：データエントリ数 0（CPU 受信 FIFO はエンプティ） 6'd1：データエントリ数 1、またはアクティビティが MSEBI バス上に存在 6'd2：データエントリ数 2 … … … 6'd31：データエントリ数 31 6'd32：データエントリ数 32（CPU 受信 FIFO はフル） ここで、1 データエントリ＝1 ワード（32 ビット）	R
b16～b11	bMSEBIM_CPU_TRANSMIT_FIFOLEVEL	CPU 送信 FIFO レベル CPU 送信 FIFO 内にある有効データエントリの数が格納されます。 6'd0：データエントリ数 0（CPU 送信 FIFO はエンプティ） 6'd1：データエントリ数 1、またはアクティビティが MSEBI バス上に存在 6'd2：データエントリ数 2 … … … 6'd31：データエントリ数 31 6'd32：データエントリ数 32（CPU 送信 FIFO はフル） ここで、1 データエントリ＝1 ワード（32 ビット）	R

表 10.25　rMSEBIM_CONFIG レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b10～b8	bMSEBIM_BURST_SIZEMAX_CPUREAD	MSEBI バスから CPU 受信 FIFO への読み出しアクセス時に許容される最大バーストサイズ。ラウンドロビンアービタによって使用されます。 「**図 10.45 MSEBI：ラウンドロビン優先順位**」を参照してください。 3'b000：1 ワード 3'b001：2 ワード 3'b010：4 ワード 3'b011：8 ワード 3'b100：16 ワード 3'b101：無制限 3'b11x：予約 1 ワードは、モード 32 デバイスでは 32 ビット幅、モード 16 デバイスでは 16 ビット幅、モード 8 デバイスでは 8 ビット幅です。 **備考**　本ビットは、CPU 受信 FIFO のプリフェッチ動作中に読み出される最大ワード数を制御するためにも使用されます。	R/W
b7～b5	bMSEBIM_BURST_SIZEMAX_CPUWRITE	CPU 送信 FIFO から MSEBI バスへの書き込みアクセス時に許容される最大バーストサイズ。ラウンドロビンアービタによって使用されます。 「**図 10.45 MSEBI：ラウンドロビン優先順位**」を参照してください。 3'b000：1 ワード 3'b001：2 ワード 3'b010：4 ワード 3'b011：8 ワード 3'b100：16 ワード 3'b101：無制限 3'b11x：予約 1 ワードは、モード 32 デバイスでは 32 ビット幅、モード 16 デバイスでは 16 ビット幅、モード 8 デバイスでは 8 ビット幅です。	R/W
b4	bMSEBIM_CLKENABLE	MSEBIM_CLK 端子出力を有効にします。 1'b0：クロック出力を無効。MSEBIM_CLK を 0 にリセット 1'b1：クロック出力を有効（別途 IO マルチプレキシング設定要） クロックを無効／有効にする際にグリッチは発生しません。（無効時はレベル 0 です）	R/W
b3、b2	bMSEBIM_CLKH	MSEBIM_CLK クロック周期の設定 MSEBIM_CLK のレベルが High である期間（MSEBIM_HCLK） 2'b00：1 MSEBIM_HCLK 2'b01：2 MSEBIM_HCLK 2'b10：3 MSEBIM_HCLK 2'b11：4 MSEBIM_HCLK	R/W
b1、b0	bMSEBIM_CLKL	MSEBIM_CLK クロック周期の設定 MSEBIM_CLK のレベルが Low である期間（MSEBIM_HCLK） 2'b00：1 MSEBIM_HCLK 2'b01：2 MSEBIM_HCLK 2'b10：3 MSEBIM_HCLK 2'b11：4 MSEBIM_HCLK	R/W

10.3.1.13　rMSEBIM_CPU_FIFOREAD_FLUSH ― フラッシュ受信 FIFO レジスタ

アドレス　400C 0808h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_CPU_FIFOREAD_FLUSH															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_CPU_FIFOREAD_FLUSH															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.26　rMSEBIM_CPU_FIFOREAD_FLUSH レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bMSEBIM_CPU_FIFOREAD_FLUSH	CPU によるパターン 32'h0808 の書き込みは CPU 受信 FIFO をフラッシュします。 32'h0808 以外のパターンを書き込んでも何もしません。 **備考）** 読むと常に 0 が読み出されます。 「**10.4.6 MSEBI マスタモード**」を参照してください。	W

10.3.2　レジスタの説明：DMA から MSEBI マスタ

10.3.2.1　rMSEBIM_DMA_FIFOREAD_CS[n]_N ― DMA 受信 FIFO（64KB）（n＝0、1）

注　意

CPU ハングアップを引き起こす恐れがあるため、CPU アクセスは禁止です。本メモリ領域は DMA コントローラ専用です。

アドレス　4008 0000h＋20000h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_DMA_FIFOREAD															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_DMA_FIFOREAD															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.27　rMSEBIM_DMA_FIFOREAD_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bMSEBIM_DMA_FIFOREAD	DMA 受信 FIFO 本レジスタを読み出すと、受信 FIFO の先頭にあるデータが読み出されます。連続して読み出すたびに受信 FIFO がポップされて、次のデータが DMA 受信 FIFO の先頭に来ます。 **「図 10.49 MSEBI：バーストモード、DMA 受信 FIFO とバスインタフェースの結合」**を参照してください。	R

10.3.2.2　rMSEBIM_DMA_FIFOWRITE_CS[n]_N ― DMA 送信 FIFO（64KB）（n＝0、1）

注　意

CPU ハングアップを引き起こす恐れがあるため、CPU アクセスは禁止です。本メモリ領域は DMA コントローラ専用です。

アドレス　4009 0000h＋20000h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIM_DMA_FIFOWRITE															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIM_DMA_FIFOWRITE															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.28　rMSEBIM_DMA_FIFOWRITE_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bMSEBIM_DMA_FIFOWRITE	DMA 送信 FIFO 本レジスタに書き込むと、送信 FIFO 内にデータがプッシュされます。連続して書き込むたびに新しいデータが DMA 送信 FIFO の次の書き込み位置にプッシュされます。 **「図 10.48 MSEBI：バーストモード、DMA 送信 FIFO とバスインタフェースの結合」**を参照してください。	W

10.3.3　レジスタの説明：CPU から MSEBI スレーブ

10.3.3.1　rMSEBIS_CYCLESIZE_CS[n]_N ― チップセレクトサイクルサイズレジスタ（n＝0～3）

注　意

設定を切り替える前に、（対応するチップセレクトの）bMSEBIS_BUSY＝0 を読み出すことによって、アクセスが進行中でないことを確認する必要があります。

本レジスタを更新する場合は、（対応するチップセレクトの）bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。

アドレス　400C 2000h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIS_WRDLEDATA_NB								bMSEBIS_RDDLEDATA_NB							
リセット後の値	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIS_WRDLEDATA_B		―	―	bMSEBIS_RDDLEDATA_B		―	―	―	―	―	―	bMSEBIS_CLEDATA	―
リセット後の値	0	0	1	1	0	0	1	1	0	0	0	0	0	0	1	1

表 10.29　rMSEBIS_CYCLESIZE_CS[n]_N レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b24	bMSEBIS_WRDLEDATA_NB	非バーストモード時のラッチデータフェーズのサイズ（WRDLEDATA_NB） 以下の場合にのみ使用されます。 ● 書き込みサイクル時 ● シングルアクセスまたはバーストサイクルの最初のアクセス MSEBIS_DLE が High である期間（MSEBIS_CLK） 8’h00：1 MSEBIS_CLK 8’h01：2 MSEBIS_CLK … … … 8’hFE：255 MSEBIS_CLK 8’hFF：256 MSEBIS_CLK 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b23～b16	bMSEBIS_RDDLEDATA_NB	非バーストモード時のラッチデータフェーズのサイズ（RDDLEDATA_NB） 以下の場合にのみ使用されます。 ● 読み出しサイクル時 ● シングルアクセスまたはバーストサイクルの最初のアクセス MSEBIS_DLE が High である期間（MSEBIS_CLK） 8’h00：1 MSEBIS_CLK 8’h01：2 MSEBIS_CLK … … … 8’hFE：255 MSEBIS_CLK 8’hFF：256 MSEBIS_CLK	R/W
b15、b14	予約ビット	読むと 0 が読み出されます。	R

表 10.29　rMSEBIS_CYCLESIZE_CS[n]_N レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b13、b12	bMSEBIS_WRDLEDATA_B	バーストモード時のラッチデータフェーズのサイズ（WRDLEDATA_B） 以下の場合にのみ使用されます。 ● 書き込みサイクル時 ● バースト許可 MSEBIS_DLE が High である期間（MSEBIS_CLK） 2’b00：1 MSEBIS_CLK 2’b01：2 MSEBIS_CLK 2’b10：3 MSEBIS_CLK 2’b11：4 MSEBIS_CLK 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b11、b10	予約ビット	読むと 0 が読み出されます。	R
b9、b8	bMSEBIS_RDDLEDATA_B	バーストモード時のラッチデータフェーズのサイズ（RDDLEDATA_B） 以下の場合にのみ使用します。 ● 読み出しサイクル時 ● バースト許可 MSEBIS_DLE が High である期間（MSEBIS_CLK） 2’b00：1 MSEBIS_CLK 2’b01：2 MSEBIS_CLK 2’b10：3 MSEBIS_CLK 2’b11：4 MSEBIS_CLK	R/W
b7～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bMSEBIS_CLEDATA	コントロールラッチフェーズのサイズ（CLEDATA） MSEBIS_CLE の期間（MSEBIS_CLK）： 0：1 MSEBIS_CLK（MSEBIS_CLK で 1 サイクルの期間中 High） 1：2 MSEBIS_CLK（MSEBIS_CLK で 1 サイクルの期間中 High、その後 MSEBIS_CLK で 1 サイクルの期間中 Low）	R/W
b0	予約ビット	初期値を維持してください。	R/W

10.3.3.2　rMSEBIS_SETUPHOLD_CS[n]_N ― チップセレクトセットアップホールドレジスタ（n＝0～3）

注　意

設定を切り替える前に、（対応するチップセレクトの）bMSEBIS_BUSY＝0 を読み出すことによって、アクセスが進行中でないことを確認する必要があります。

本レジスタを更新する場合は、（対応するチップセレクトの）bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。

アドレス　400C 2004h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIS_WRDLESETUP						―	―	bMSEBIS_RDDLESETUP					
リセット後の値	0	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1

表 10.30　rMSEBIS_SETUPHOLD_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット	読むと 0 が読み出されます。	R
b13～b8	bMSEBIS_WRDLESETUP	セットアップデータフェーズのサイズ（WRDLESETUP） 以下の場合にのみ使用されます。 ● 書き込みサイクル時 セットアップフェーズの期間（MSEBIS_CLK） 6'h00：予約 6'h01：1 MSEBIS_CLK … … … 6'h3E：62 MSEBIS_CLK 6'h3F：63 MSEBIS_CLK	R/W
b7、b6	予約ビット	読むと 0 が読み出されます。	R
b5～b0	bMSEBIS_RDDLESETUP	セットアップデータフェーズのサイズ（RDDLESETUP） 以下の場合にのみ使用されます。 ● 読み出しサイクル時 セットアップフェーズの期間（MSEBIS_CLK） 6'h00：予約 6'h01：1 MSEBIS_CLK … … … 6'h3E：62 MSEBIS_CLK 6'h3F：63 MSEBIS_CLK 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W

10.3.3.3　rMSEBIS_MMU_ADDR_CS[n]_N ― MMU ベースアドレスレジスタ（n＝0～3）

注　意

設定を切り替える前に、（対応するチップセレクトの）bMSEBIS_BUSY＝0 を読み出すことによって、アクセスが進行中でないことを確認する必要があります。

本レジスタを更新する場合は、（対応するチップセレクトの）bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。

アドレス　400C 2008h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIS_MMU_ADDR															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIS_MMU_ADDR				―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.31　rMSEBIS_MMU_ADDR_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b12	bMSEBIS_MMU_ADDR	bMSEBIS_ADDR_MODE ビットが MMU モードに設定されている場合、MSEBI スレーブコントローラはアドレス変換用に本パラメータを使用します。 MMU モードにおいては、bMSEBIS_MMU_ADDR が変換用のベースアドレスとなります。 「**10.4.7.6 MSEBI スレーブ：アドレス指定モード**」を参照してください。	R/W
b11～b0	予約ビット	読むと 0 が読み出されます。	R

10.3.3.4　rMSEBIS_MMU_ADDR_MASK_CS[n]_N ― MMU アドレスマスクレジスタ（n＝0～3）

注　意

設定を切り替える前に、（対応するチップセレクトの）bMSEBIS_BUSY＝0 を読み出すことによって、アクセスが進行中でないことを確認する必要があります。

本レジスタを更新する場合は、（対応するチップセレクトの）bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。

アドレス　400C 200Ch＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIS_MMU_ADDR_MASK															
リセット後の値	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIS_MMU_ADDR_MASK				―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

表 10.32　rMSEBIS_MMU_ADDR_MASK_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b12	bMSEBIS_MMU_ADDR_MASK	bMSEBIS_ADDR_MODE ビットが MMU モードに設定されている場合、MSEBI スレーブコントローラはアドレス変換用に本パラメータを使用します。 MMU モードにおいては、bMSEBIS_MMU_ADDR が変換用のアドレスのマスクになります。 「**10.4.7.6 MSEBI スレーブ：アドレス指定モード**」を参照してください。	R/W
b11～b0	予約ビット	読むと 12'hFFF が読み出されます。	R

10.3.3.5　rMSEBIS_DMATX_REQ_CS[n]_N ― DMA 送信要求レジスタ（n＝0、1）

アドレス　400C 2010h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bMSEBIS_DMATX_ENABLE	bMSEBIS_DMATX_FORCE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

表 10.33　rMSEBIS_DMATX_REQ_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bMSEBIS_DMATX_ENABLE	各 MSEBI_CS[n]_N（n＝0、1）に対して MSEBI スレーブコントローラの DMA TX チャネルの設定状態を示します。 本ビットフィールドは、必ずチップセレクトの設定終了時にセットする必要があります。 本ビットは以下の目的に使用されます。 ● MSEBI_CS[n]_N を MSEBI_DMA_N に結合させたとき、チップセレクトをアクティブ／非アクティブに設定します。DMA TX FIFO がエンプティでなければ、エンプティになるまで読み出されます。 「**10.4.7.3 MSEBI スレーブ：要求イニシエータの検出**」を参照してください。 許可ビットは以下のように管理されます。 1’b0：MSEBI スレーブコントローラの DMA TX チャネルは、マスタからの要求を受信できる状態にありません。CLE フェーズ中に MSEBI_CS[n]_N と MSEBI_DMA_N が 0 になると、現在のアクセスは無視されます。 1’b1：MSEBI スレーブコントローラの DMA TX チャネルは、マスタからの要求を受信できる状態にあります。CLE フェーズ中に MSEBI_CS[n]_N と MSEBI_DMA_N が 0 になると、現在のアクセスが実行されます。 **注意**　チップセレクト（MSEBI_CS[n]_N）はアクティブ状態（専用の bMSEBIS_CS_ENABLE ビット＝1）に設定する必要があります。0 にクリアされていると、現在のアクセスは無視されます。	R/W
b0	bMSEBIS_DMATX_FORCE	各 MSEBI_CS[n]_N（n＝0、1）に対して DMA フロー制御が有効（bMSEBIS_DMATX_FLOW_CTRL ビット＝1）の場合、bMSEBIS_DMATX_FORCE ビットは下記の DMA フロー制御端子を駆動します。 MSEBIS_DMA_WR[n]_N（n＝0、1） 1’b0：MSEBIS_DMA_WR[n]_N を 0 に設定 1’b1：MSEBIS_DMA_WR[n]_N を 1 に設定	R/W

10.3.3.6　rMSEBIS_DMARX_REQ_CS[n]_N ― DMA 受信要求レジスタ（n＝0、1）

アドレス　400C 2014h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	bMSEBIS_DMARX_ENABLE	bMSEBIS_DMARX_FORCE
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

表 10.34　rMSEBIS_DMARX_REQ_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b2	予約ビット	読むと 0 が読み出されます。	R
b1	bMSEBIS_DMARX_ENABLE	各 MSEBI_CS[n]_N（n＝0、1）に対して MSEBI スレーブコントローラの DMA RX チャネルの設定状態を示します。 本ビットフィールドは、必ずチップセレクトの設定終了時にセットする必要があります。 本ビットは以下の目的に使用されます。 ● 対応する FIFO のフラッシュ ● MSEBI_CS[n]_N を MSEBI_DMA_N に結合させたとき、チップセレクトをアクティブ／非アクティブに設定 「**10.4.7.3 MSEBI スレーブ：要求イニシエータの検出**」を参照してください。 許可ビットは以下のように管理されます。 1’b0：MSEBI スレーブコントローラの DMA RX チャネルは、マスタからの要求を受信できる状態にありません。CLE フェーズ中に MSEBI_CS[n]_N と MSEBI_DMA_N が 0 になると、現在のアクセスは無視されます。 1’b1：MSEBI スレーブコントローラの DMA RX チャネルは、マスタからの要求を受信できる状態にあります。CLE フェーズ中に MSEBI_CS[n]_N と MSEBI_DMA_N が 0 になると、現在のアクセスが実行されます。 **注意**　チップセレクト（MSEBI_CS[n]_N）はアクティブ状態（専用の bMSEBIS_CS_ENABLE ビット＝1）に設定する必要があります。0 にクリアされていると、現在のアクセスは無視されます。	R/W
b0	bMSEBIS_DMARX_FORCE	各 MSEBI_CS[n]_N（n＝0、1）に対して DMA フロー制御が有効（bMSEBIS_DMARX_FLOW_CTRL ビット＝1）の場合、bMSEBIS_DMARX_FORCE ビットは下記の DMA フロー制御信号を駆動します。 MSEBIS_DMA_RD[n]_N（n＝0、1） 1’b0：MSEBIS_DMA_RD[n]_N を 0 に設定 1’b1：MSEBIS_DMA_RD[n]_N を 1 に設定	R/W

10.3.3.7　rMSEBIS_DMATDLR_CS[n]_N ― DMA 送信データレベルレジスタ（n＝0、1）

注　意

本レジスタを更新する場合は、（対応するチップセレクトの）bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。

アドレス　400C 2018h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIS_DMATX_FIFO_LVL						―	―	―	―	bMSEBIS_DMATX_FLOW_CTRL	bMSEBIS_DMATX_OPT_BURST	bMSEBIS_DMATX_MAX_BURST	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.35　rMSEBIS_DMATDLR_CS[n]_N レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31	予約ビット	初期値を保持	R/W
b30～b14	予約ビット	読むと 0 が読み出されます。	R
b13～b8	bMSEBIS_DMATX_FIFO_LVL	DMA TX[n]（n＝0、1）FIFO レベル 下記の状態で MSEBI_CS[n]_N の MSEBI バスから書き込み要求があると、本 FIFO は AHB バスを用いてメモリに書き込むデータを保持します。 MSEBI_DMA_N＝DMA モード 「**表 10.47　スレーブの要求イニシエータの検出**」を参照してください。 bMSEBIS_DMATX_FIFO_LVL ビットには、DMA TX[n]（n＝0、1）FIFO 内にある有効データエントリの数が格納されます。 6'd0：データエントリ数 0（DMA TX[n] FIFO はエンプティ） 6'd1：データエントリ数 1、またはアクティビティが AHB バス上に存在 6'd2：データエントリ数 2 … … … 6'd31：データエントリ数 31 6'd32：データエントリ数 32（DMA TX[n] FIFO はフル） ここで、1 データエントリ＝1 ワード（32 ビット） 「**10.4.7.2(2)(a)　MSEBI マスタの DMA TX FIFO からの要求に対するスレーブの DMA FIFO**」を参照してください。	R
b7～b4	予約ビット	読むと 0 が読み出されます。	R
b3	bMSEBIS_DMATX_FLOW_CTRL	MSEBIS_DMA_WR[n]_N（n＝0、1）信号に対する DMA フロー制御を有効にします。 「**10.4.7.2(3)　スレーブの DMA フロー制御信号**」を参照してください。 1'b0：DMA フロー制御は無効 1'b1：DMA フロー制御は有効	R/W
b2	bMSEBIS_DMATX_OPT_BURST	MSEBI スレーブコントローラは、書き込み要求を NoC へ送信する前に（「ブロック終了イベント」を受信するまでの間に）、バーストの準備を行えるだけの十分なデータがスレーブの DMA TX[n]（n＝0、1）FIFO 内に格納されるのを待ちます。 本オプションは、NoC AHB の帯域幅を最適化します。 以下のように、DMA TX[n]（n＝0、1）FIFO の最適化を有効にします。 1'b0：バーストサイズの最適化は無効 1'b1：バーストサイズの最適化は有効 「**10.4.7.2　MSEBI スレーブ：バーストモード**」および「**10.4.7.2(2)(a)　MSEBI マスタの DMA TX FIFO からの要求に対するスレーブの DMA FIFO**」を参照してください。	R/W

表 10.35　rMSEBIS_DMATDLR_CS[n]_N レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b1、b0	bMSEBIS_DMATX_MAX_BURST	MSEBI スレーブコントローラは、シングル書き込みアクセスをバーストにまとめることにより、スレーブの DMA TX[n]（n＝0、1）FIFO から AHB マスタポートへのアクセスを最適化できます。バーストの最大サイズの設定が可能です。バーストサイズは以下の値によって制御されます。 2’b00：最大 1 ワード 2’b01：最大 4 ワード 2’b10：最大 8 ワード 2’b11：最大 16 ワード 「**10.4.7.2 MSEBI スレーブ：バーストモード**」および「**10.4.7.2(2)(a) MSEBI マスタの DMA TX FIFO からの要求に対するスレーブの DMA FIFO**」を参照してください。	R/W

10.3.3.8　rMSEBIS_DMARDLR_CS[n]_N ― DMA 受信データレベルレジスタ（n＝0、1）

注　意

本レジスタを更新する場合は、（対応するチップセレクトの）bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。

アドレス　400C 201Ch＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIS_DMARX_FIFO_L						―	―	―	―	―	bMSEBIS_DMARX_FLOW_CTRL	bMSEBIS_DMARX_MAX_BURST	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.36　rMSEBIS_DMARDLR_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31	予約ビット	初期値を保持	R/W
b30～b14	予約ビット	読むと 0 が読み出されます。	R
b13～b8	bMSEBIS_DMARX_FIFO_L	DMA RX[n]（n＝0、1）FIFO レベル 下記の状態で MSEBI_CS[n]N（n＝0、1）の MSEBI バスから読み出し要求があると、本 FIFO は AHB バスに読み出されるデータを保持します。 MSEBI_DMA_N＝DMA モード bMSEBIS_DMARX_FIFO_LVL ビットには、DMA RX[n]（n＝0、1）FIFO 内にある有効データエントリの数が格納されます。 6’d0：データエントリ数 0（DMA RX[n] FIFO はエンプティ） 6’d1：データエントリ数 1、またはアクティビティが MSEBI バス上に存在 6’d2：データエントリ数 2 … … … 6’d31：データエントリ数 31 6’d32：データエントリ数 32（DMA RX[n] FIFO はフル） ここで、1 データエントリ＝1 ワード（32 ビット） **「表 10.47 スレーブの要求イニシエータの検出」**および**「図 10.62 MSEBI マスタの DMA RX FIFO からの要求に対するスレーブの DMA FIFO」**を参照してください。	R
b7～b3	予約ビット	読むと 0 が読み出されます。	R
b2	bMSEBIS_DMARX_FLOW_CTRL	MSEBIS_DMA_RD[n]_N（n＝0、1）信号に対する DMA フロー制御を有効にします。 1’b0：DMA フロー制御は無効 1’b1：DMA フロー制御は有効	R/W
b1、b0	bMSEBIS_DMARX_MAX_BURST	MSEBI スレーブコントローラは、AHB マスタポートのプリフェッチ機構を用いることで、MSEBI バスのマスタの DMA RX[n] FIFO（n＝0、1）から来る読み出しアクセスの待ち時間を最適化できます。 バーストプリフェッチのサイズは以下の値に設定できます。 2’b00：最大 1 ワード 2’b01：最大 4 ワード 2’b10：最大 8 ワード 2’b11：最大 16 ワード **「10.4.7.2 MSEBI スレーブ：バーストモード」**および**「10.4.7.2(2)(b) MSEBI マスタの DMA RX FIFO からの要求に対するスレーブの DMA FIFO」**を参照してください。	R/W

10.3.3.9　rMSEBIS_CONFIG_CS[n]_N ― チップセレクトコンフィグレジスタ（n＝0～3）

注　意

設定を切り替える前に、（対応するチップセレクトの）bMSEBIS_BUSY＝0 を読み出すことによって、アクセスが進行中でないことを確認する必要があります。

本レジスタを更新する場合は、（対応するチップセレクトの）bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。

アドレス　400C 2060h＋100h×n

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	bMSEBIS_CS[n]N_ROUTING_CS3_N	bMSEBIS_CS[n]N_ROUTING_CS2_N	bMSEBIS_CS[n]N_ROUTING_CS1_N	―	―	bMSEBIS_CS_ENABLE	bMSEBIS_ADDR_MODE	bMSEBIS_BURST_ENABLE	bMSEBIS_MODE_WAIT	―	bMSEBIS_WEN	bMSEBIS_BUSY	bMSEBIS_CONFIG		
リセット後の値	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

表 10.37　rMSEBIS_CONFIG_CS[n]_N レジスタの内容（1/4）

ビット位置	ビット名	機能	R/W
b31～b15	予約ビット	読むと 0 が読み出されます。	R
b14	bMSEBIS_CS[n]N_ROUTING_CS3_N	下記チップセレクトに対するアクセス中に、 MSEBI_CS0_N MSEBI_CS1_N MSEBI_CS2_N MSEBI_CS3_N をアドレスビットとして使用することにより、ユーザの使用例に合わせてアドレス機能を拡張できます。アドレスビットの割り当ては、ALE フェーズの数に依存します。 本ビットは下記レジスタでは使用されないため、読むと 0 が読み出されます。 rMSEBIS_CONFIG_CS3_N ビットは以下のように管理されます。 1'b0：本ライン上ではアドレスルーティングなし 1'b1：本ライン上でアドレスルーティングを許可 「**10.1.3 マルチプレクス信号インタフェース**」を参照してください。	R/W
b13	bMSEBIS_CS[n]N_ROUTING_CS2_N	下記チップセレクトによるアクセス中に、 MSEBI_CS0_N MSEBI_CS1_N MSEBI_CS2_N をアドレスビットとして使用することにより、ユーザの使用例に合わせてアドレス機能を拡張できます。アドレスビットの割り当ては、ALE フェーズの数に依存します。 これらのビットは下記レジスタでは使用されないため、読むと 0 が読み出されます。 rMSEBIS_CONFIG_CS2_N rMSEBIS_CONFIG_CS3_N ビットは以下のように管理されます。 1'b0：本ライン上ではアドレスルーティングなし 1'b1：本ライン上でアドレスルーティングを許可 「**10.1.3 マルチプレクス信号インタフェース**」を参照してください。	R/W

表 10.37　rMSEBIS_CONFIG_CS[n]_N レジスタの内容（2/4）

ビット位置	ビット名	機能	R/W
b12	bMSEBIS_CS[n]N_ROUTING_CS1_N	下記チップセレクトによるアクセス中に、 MSEBI_CS0_N MSEBI_CS1_N をアドレスビットとして使用することにより、ユーザの使用例に合わせてアドレス機能を拡張できます。アドレスビットの割り当ては、ALE フェーズの数に依存します。 これらのビットは下記レジスタでは使用されないため、読むと 0 が読み出されます。 rMSEBIS_CONFIG_CS1_N rMSEBIS_CONFIG_CS2_N rMSEBIS_CONFIG_CS3_N ビットは以下のように管理されます。 1'b0：本ライン上ではアドレスルーティングなし 1'b1：本ライン上でアドレスルーティングを許可 「**10.1.3 マルチプレクス信号インタフェース**」を参照してください。	R/W
b11、b10	予約ビット	読むと 0 が読み出されます。	R
b9	bMSEBIS_CS_ENABLE	MSEBI スレーブコントローラの CPU チャネルの設定状態を示します。 本ビットフィールドは、スレーブの CPU が MSEBI_CS[n]_N（n＝0～3）専用のコンフィグレーションレジスタグループに対して設定を完了したとき、スレーブの CPU によってセットされます。 許可ビットは以下のように管理されます。 ● 1'b0：MSEBI スレーブコントローラの CPU チャネルは、マスタからの要求を受信できる状態にない – チップセレクト（MSEBI_CS[n]_N）を非アクティブ状態に設定 – rMSEBIS_ID_CS[n]_N レジスタに通知されるステータス値を制御 – MSEBI_CS[n]_N 専用のコンフィグレーションレジスタグループをアンロック（CPU による書き込みが可能） ● 1'b1：MSEBI スレーブコントローラの CPU チャネルは、マスタからの要求を受信できる状態にある – チップセレクト（MSEBI_CS[n]_N）をアクティブ状態に設定 – rMSEBIS_ID_CS[n]_N レジスタに通知されるステータス値を制御 – MSEBI_CS[n]_N 専用のコンフィグレーションレジスタグループをロック（CPU による書き込みが不可） bMSEBIS_CS_ENABLE ビットの立ち下がりが検出されると、rMSEBIS_ID_CS レジスタ内部で専用の bMSEBIS_ERROR_CS_CONFIGURATION ビットがセットされます。 以下のコンフィグレーションレジスタへのアクセスがロック／アンロックされます。 ● rMSEBIS_CYCLESIZE_CS[n]_N（n＝0～3） ● rMSEBIS_SETUPHOLD_CS[n]_N（n＝0～3） ● rMSEBIS_MMU_ADDR_CS[n]_N（n＝0～3） ● rMSEBIS_MMU_MASK_ADDR_CS[n]_N（n＝0～3） ● rMSEBIS_DMATDLR_CS[n]_N（n＝0、1） ● rMSEBIS_DMARDLR_CS[n]_N（n＝0、1） ● rMSEBIS_CONFIG_CS[n]_N（n＝0～3） ● rMSEBIS_CONFIG ● rMSEBIS_EOB_ADDR **備考）** MSEBI_CS[n]_N（n＝0～3）に対して、bMSEBIS_CS_ENABLE ビットがクリアされた場合： ● 専用の要求は、CLE および DLE フェーズのデコーディングでは考慮されません。 ● CPU および DMA FIFO 内部の専用の要求は実行されません。 ● MSEBI_CS[n]_N に対するアクセスが進行中であれば、データおよび MSEBI マスタ・スレーブ間の同期が失われます。 「**10.4.7.8 MSEBI スレーブ：コンフィグレーションレジスタ＆同期**」を参照してください。	R/W

表 10.37　rMSEBIS_CONFIG_CS[n]_N レジスタの内容（3/4）

ビット位置	ビット名	機能	R/W
b8	bMSEBIS_ADDR_MODE	MSEBI スレーブインタフェースは下記の 2 つの基本機能に設定できます。 1'b0：ダイレクトモードによるアドレス管理 1'b1：MMU モードによるアドレス管理 「**10.4.7.6 MSEBI スレーブ：アドレス指定モード**」を参照してください。	R/W
b7	bMSEBIS_BURST_ENABLE	各 MSEBI_CS[n]_N（n＝0～3）において、AHB マスタポートに対する読み出し／書き込みアクセス時のバーストモードを許可します。 バーストサイズは、rMSEBIS_CONFIG レジスタにおける最大バーストサイズの設定パラメータによって制限されます。 バーストを禁止すると、本チップセレクトに対するプリフェッチも禁止されます。 1'b0：バーストを禁止（シングルアクセスのみ） 1'b1：バーストを許可（シングルおよびバーストアクセス） 「**10.4.7.2 MSEBI スレーブ：バーストモード**」および「**10.4.4 MSEBI タイミング**」を参照してください。 **注意）** マスタがスレーブの共有レジスタに（MSEBI_CSREG_N を用いて）アクセスしているときは、プリフェッチを行わないでください。	R/W
b6、b5	bMSEBIS_MODE_WAIT	各 MSEBI_CS[n]_N（n＝0～3）に対して、MSEBI スレーブインタフェースは下記の 3 つの基本機能に設定できます。 2'b00：予約 2'b01：MSEBIS_WAIT[n]_N 端子に対するウェイト管理。専用の外部端子 MSEBIS_WAIT[n]_N が駆動されます。 2'b10：MSEBIS_WAIT0_N 端子に対するウェイト管理。選択した MSEBI_CS[n]_N に対して、1 つの外部共通端子 MSEBIS_WAIT0_N のみが駆動されます。 2'b11：予約 2'b10 のモードでは、すべての MSEBI_CS[n]_N に対して、1 つの外部端子 MSEBIS_WAIT0_N だけが使用可能です。この場合、必要な外部端子の本数が削減されます。 「**10.4.4 MSEBI タイミング**」を参照してください。	R/W
b4	予約ビット	初期値を保持	R/W
b3	bMSEBIS_WEN	本ビットは、デバイスに対する書き込みアクセス権の管理を許可します。 bMSEBIS_WEN ビットが 0 のときに書き込みアクセスが発生すると、rMSEBIS_STATUS レジスタにエラーフラグがセットされます。 アクセス権は以下のように設定されます。 1'b0：デバイスへの書き込みを禁止 1'b1：デバイスへの書き込みを許可 「**10.4.7.7 MSEBI スレーブ：書き込み保護**」を参照してください。	R/W

表 10.37　rMSEBIS_CONFIG_CS[n]_N レジスタの内容（4/4）

ビット位置	ビット名	機能	R/W
b2	bMSEBIS_BUSY	各 MSEBI_CS[n]_N（n＝0～3）において、MSEBI スレーブコントローラの通信ステータスを示します。bMSEBIS_BUSY ビットは、下記条件がすべて満たされるまで 1 にセットされています。 ● CPU または DMA モードにおいて、チップセレクト（MSEBI_CS[n]_N）で進行中の MSEBI バスアクセスがない ● 進行中の AHB バスアクセスがない ● DMA モード：AHB バスから DMA RX FIFO への読み出し – DMA が禁止（bMSEBIS_DMARX_ENABLE ビットがクリア）されている場合、FIFO がフラッシュされエンプティ状態です。 – DMA が許可（bMSEBIS_DMARX_ENABLE ビットがセット）されている場合、DMA RX FIFO はエンプティ状態であり、進行中の AHB アクセスはありません。 ● DMA モード：DMA TX FIFO から AHB バスへの書き込み – DMA TX FIFO はエンプティ状態であり（必要な「ブロック終了イベント」を待たない場合）、かつ進行中の AHB アクセスがありません。 ● CPU モード：CPU 受信 FIFO から AHB バスへの書き込み – CPU 受信 FIFO 内に保留中の要求がありません。 – 進行中の AHB アクセスがありません。 ● CPU モード：AHB バスから CPU 送信 FIFO へ – CPU 送信 FIFO 内に保留中の要求がありません。 – 進行中の AHB アクセスがありません。 1’b0：本チップセレクト（MSEBI_CS[n]_N）に対して MSEBI スレーブコントローラはアイドル状態 1’b1：本チップセレクト（MSEBI_CS[n]_N）で要求が現在も進行中 以下の図を参照してください。 **「図 10.62 MSEBI マスタの DMA RX FIFO からの要求に対するスレーブの DMA FIFO」** **「図 10.61 MSEBI マスタの DMA TX FIFO からの要求に対するスレーブの DMA FIFO」** **「図 10.58 MSEBI スレーブの CPU FIFO の例 1」** **「図 10.59 MSEBI スレーブの CPU FIFO の例 1、8 ビットモード」** **「図 10.60 MSEBI スレーブの CPU FIFO の例 2」**	R
b1、b0	bMSEBIS_CONFIG	MSEBI スレーブインタフェースは下記の 3 つの基本機能に設定できます。 2’b00：同期式、16 ビット、マルチプレクス、モード 16、バースト可 2’b01：同期式、32 ビット、マルチプレクス、モード 32、バースト可 2’b10：同期式、8 ビット、マルチプレクス、モード 8、バースト可 2’b11：予約ビット 下記の項を参照してください。 **「10.1.3 マルチプレクス信号インタフェース」** **「10.4.4 MSEBI タイミング」**	R/W

10.3.3.10　rMSEBIS_CONFIG ― コモンコンフィグレジスタ

注　意

設定を切り替える前に、（対応するチップセレクトの）bMSEBIS_BUSY＝0 を読み出すことによって、アクセスが進行中でないことを確認する必要があります。

本レジスタを更新する場合は、すべての MSEBI_CS[n]_N チップセレクトの bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。（いつでもアクセス可能な bMSEBIS_TIMEOUT_REG_ACCESS ビットは例外です）。

アドレス　400C 2800h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	bMSEBIS_WAIT_CONF	—	bMSEBIS_TIMEOUT_REG_ACCESS	bMSEBIS_TIMEOUT_REG_ACCESS_DELAY				bMSEBIS_AHB_MASTER_CACHE	bMSEBIS_AHB_MASTER_BUF	bMSEBIS_CPUTX_FIFO_LVL					
リセット後の値	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	bMSEBIS_CPURX_FIFO_LVL						—	—	—	—	bMSEBIS_BURST_SIZEMAX_CPUWRITE		bMSEBIS_BURST_SIZEMAX_CPUREAD	
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.38　rMSEBIS_CONFIG レジスタの内容（1/4）

ビット位置	ビット名	機能	R/W
b31	予約ビット	初期値を保持	R/W
b30	bMSEBIS_WAIT_CONF	1 を設定してください。	R/W
b29	予約ビット	初期値を保持	R/W
b28	bMSEBIS_TIMEOUT_REG_ACCESS	スレーブのコンフィグレーションレジスタに対する書き込みアクセスにおいて、同期機構の動作中のタイムアウトを表示します。 読み出しアクセス時： 1'b0：タイムアウトの検出なし 1'b1：タイムアウトの検出あり 書き込みアクセス時（いつでもクリア可能）： 1'b0：0 の書き込みは無効 1'b1：1 の書き込みはビットをクリア 本ビットのクリアを可能にするには、最初に同期機構がタイムアウト条件を終了させる必要があります。そうしないと、本フラグは 1'b1 のままです。本状態を終了するには、以下の操作を行います。 ● 同期機構が正しく終了できるように、MSEBIS_CLK クロックの起動を待ちます。 ● 起動した最後のチップセレクトを停止させます。 その後、本フラグのクリアが可能になります。 本レジスタの bMSEBIS_TIMEOUT_REG_ACCESS_DELAY ビットフィールドを参照してください。	R/W

表 10.38　rMSEBIS_CONFIG レジスタの内容（2/4）

ビット位置	ビット名	機能	R/W
b27～b24	bMSEBIS_TIMEOUT_REG_ACCESS_DELAY	bMSEBIS_TIMEOUT_REG_ACCESS_DELAY ビットは、チップセレクト[n]（n＝0～3）用のコンフィグレーションレジスタがアンロックされたとき、タイムアウトがトリガされるまでの遅延（MSEBIS_HCLK クロックサイクル数）を定義します。下記ビットでチップセレクト[n]を有効に設定したとき、コンフィグレーションレジスタへの書き込みアクセスがロックされます。 bMSEBIS_CS_ENABLE コンフィグレーションレジスタに書き込む場合、事前に bMSEBIS_CS_ENABLE ビットをクリアして、レジスタをアンロックする必要があります。MSEBI スレーブコントローラがアンロックコマンドを受信すると、内部同期機構の動作が完了するまでの間、コンフィグレーションレジスタに対する AHB アクセスは保留されます。 ● 同期は、MSEBIS_CLK クロックで 2 サイクル＋MSEBIS_HCLK クロックで 2 サイクルだけ継続します。MSEBIS_CLK クロックがバスのマスタによって切断されると、内部同期機構は動作を完了することができません。AHB のデッドロックを防ぐために、MSEBI スレーブコントローラは MSEBIS_HCLK に基づくタイムアウトを使用します。 タイムアウトの遅延は以下のように設定できます。 4'h0：4 MSEBIS_HCLK クロックサイクル後にタイムアウト 4'h1：8 MSEBIS_HCLK クロックサイクル後にタイムアウト … … … 4'hE：60 MSEBIS_HCLK クロックサイクル後にタイムアウト 4'hF：64 MSEBIS_HCLK クロックサイクル後にタイムアウト タイムアウト後の動作は以下のとおりです。 ● AHB バスが解放されます。 ● 同期は続行されます。 ● コンフィグレーションレジスタへの書き込みアクセスは、同期機構の終了までロックされます。 ● bMSEBIS_TIMEOUT_REG_ACCESS ビットにフラグがセットされます。 「**10.4.7.8 MSEBI スレーブ：コンフィグレーションレジスタ &同期**」を参照してください。	R/W
b23	bMSEBIS_AHB_MASTER_CACHE	スレーブの AHB マスタポートからの要求を設定することにより、キャッシュの使用が可能になります。 本パラメータは AHB 信号（HPROT[3]）を駆動します。 以下のように設定します。 1'd0：データはキャッシュ可能でない 1'd1：データはキャッシュ可能	R/W
b22	bMSEBIS_AHB_MASTER_BUF	スレーブの AHB マスタポートからの要求を設定することにより、バッファ可能なアクセスの使用が可能になります。本パラメータは AHB 信号（HPROT[2]）を駆動します。 以下のように設定します。 1'd0：データはバッファ可能でない 1'd1：データはバッファ可能	R/W

表 10.38　rMSEBIS_CONFIG レジスタの内容（3/4）

ビット位置	ビット名	機能	R/W
b21～b16	bMSEBIS_CPUTX_FIFO_LVL	CPU 送信 FIFO レベル MSEBI_DMA_N＝CPU モードの状態で MSEBI バスから読み出し要求があると、本 FIFO は AHB バスに読み出されるデータを保持します。 「**表 10.47 スレーブの要求イニシエータの検出**」を参照してください。 bMSEBIS_CPUTX_FIFO_LVL ビットフィールドには、CPU 送信 FIFO 内にある有効データエントリの数が格納されます。 6'd0：データエントリ数 0（CPU 送信 FIFO はエンプティ） 6'd1：データエントリ数 1、またはアクティビティが AHB バス上に存在 6'd2：データエントリ数 2 … … … 6'd31：データエントリ数 31 6'd32：データエントリ数 32（CPU 送信 FIFO はフル） 1 個のデータエントリのサイズは、MSEBI バスのサイズとアクセスタイプ（バーストまたは非バースト）に依存して、8/16/32 ビットになります。 「**図 10.58MSEBI スレーブの CPU FIFO の例 1**」、「**図 10.59 MSEBI スレーブの CPU FIFO の例 1、8 ビットモード**」、および「**図 10.60 MSEBI スレーブの CPU FIFO の例 2**」を参照してください。	R
b15、b14	予約ビット	読むと 0 が読み出されます。	R
b13～b8	bMSEBIS_CPURX_FIFO_LVL	CPU 受信 FIFO レベル 本 FIFO は、MSEBI_DMA_N＝CPU モードの状態で MSEBI バスから受信したすべての要求を保持します。 「**表 10.47 スレーブの要求イニシエータの検出**」を参照してください。 bMSEBIS_CPURX_FIFO_LVL ビットには、CPU 受信 FIFO 内にある有効データエントリの数が格納されます。 6'd0：データエントリ数 0（CPU 受信 FIFO はエンプティ） 6'd1：データエントリ数 1、またはアクティビティが MSEBI バス上に存在 6'd2：データエントリ数 2 … … … 6'd31：データエントリ数 31 6'd32：データエントリ数 32（CPU 受信 FIFO はフル） 1 個のデータエントリのサイズは、MSEBI バスのサイズに依存して、8/16/32 ビットになります。 「**図 10.58 MSEBI スレーブの CPU FIFO の例 1**」、「**図 10.59 MSEBI スレーブの CPU FIFO の例 1、8 ビットモード**」、および「**図 10.60 MSEBI スレーブの CPU FIFO の例 2**」を参照してください。	R
b7～b4	予約ビット	読むと 0 が読み出されます。	R
b3、b2	bMSEBIS_BURST_SIZEMAX_CPUWRITE	MSEBI スレーブコントローラは、シングル書き込みアクセスをバーストにまとめることにより、スレーブの CPU 受信 FIFO から AHB マスタポートへのアクセスを最適化できます。 バーストの最大サイズの設定が可能です。 バーストサイズは以下の値によって制御されます。 2'b00：最大 1 ワード 2'b01：最大 4 ワード 2'b10：最大 8 ワード 2'b11：最大 16 ワード 「**10.4.7.2 MSEBI スレーブ：バーストモード**」および「**10.4.7.2(1) スレーブの CPU FIFO**」を参照してください。	R/W

表 10.38　rMSEBIS_CONFIG レジスタの内容（4/4）

ビット位置	ビット名	機能	R/W
b1、b0	bMSEBIS_BURST_SIZEMAX_CPUREAD	MSEBI スレーブコントローラは、AHB マスタポートのプリフェッチ機構を用いることで、MSEBI バスのマスタの CPU 送信 FIFO から来る読み出しアクセスの待ち時間を最適化できます。 プリフェッチバーストのサイズは以下の値に設定できます。 2’b00：最大 1 ワード 2’b01：最大 4 ワード 2’b10：最大 8 ワード 2’b11：最大 16 ワード 「**10.4.7.2 MSEBI スレーブ：バーストモード**」および「**10.4.7.2(1) スレーブの CPU FIFO**」を参照してください。	R/W

10.3.3.11　rMSEBIS_STATUS_INT0 ― 割り込みステータスレジスタ

本レジスタのビットは、コントローラのスレーブ部で発生した「ブロック終了イベント」割り込みを参照しています。

これらのビットには、マスク（rMSEBIS_MASK_INT）が適用される前の MSEBI スレーブの割り込み状態が格納されます。MSEBI バスのマスタが下記レジスタの対応するビットに 1 を書き込んだとき、それぞれのビットが個別にセットされます。

- rMSEBIS_INT

「**10.4.5 MSEBI 割り込み**」を参照してください。

アドレス　400C 2804h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIS_INT0_DMATX		―	―	bMSEBIS_INT0_DMARX		bMSEBIS_INT0_CPUTX				bMSEBIS_INT0_CPURX			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.39　rMSEBIS_STATUS_INT0 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット	読むと 0 が読み出されます。	R
b13、b12	bMSEBIS_INT0_DMATX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 1'b0：ブロック終了未検出 1'b1：ブロック終了を検出	R
b11、b10	予約ビット	読むと 0 が読み出されます。	R
b9、b8	bMSEBIS_INT0_DMARX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 1'b0：ブロック終了未検出 1'b1：ブロック終了を検出	R
b7～b4	bMSEBIS_INT0_CPUTX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 1'b0：ブロック終了未検出 1'b1：ブロック終了を検出	R
b3～b0	bMSEBIS_INT0_CPURX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 1'b0：ブロック終了未検出 1'b1：ブロック終了を検出	R

10.3.3.12　rMSEBIS_STATUS_INT1 ― マスク後割り込みステータスレジスタ

本レジスタのビットは、コントローラのスレーブ部で発生した「ブロック終了イベント」割り込みを参照しています。

これらのビットには、rMSEBIS_STATUS_INT0 レジスタにマスク（rMSEBIS_MASK_INT）が適用された後の MSEBI スレーブの割り込み状態が格納されます。それぞれのビットは、下記レジスタの対応するビット間の論理積の結果です。

- rMSEBIS_MASK_INT
- rMSEBIS_STATUS_INT0

「**10.4.5 MSEBI 割り込み**」を参照してください。

アドレス　400C 2808h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	bMSEBIS_INT1_DMATX		—	—	bMSEBIS_INT1_DMARX		bMSEBIS_INT1_CPUTX				bMSEBIS_INT1_CPURX			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.40　rMSEBIS_STATUS_INT1 レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット	読むと 0 が読み出されます。	R
b13、b12	bMSEBIS_INT1_DMATX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 1'b0：ブロック終了割り込み未検出またはマスク済み 1'b1：ブロック終了を検出かつマスクなし	R
b11、b10	予約ビット	読むと 0 が読み出されます。	R
b9、b8	bMSEBIS_INT1_DMARX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 1'b0：ブロック終了割り込み未検出またはマスク済み 1'b1：ブロック終了を検出かつマスクなし	R
b7～b4	bMSEBIS_INT1_CPUTX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 1'b0：ブロック終了割り込み未検出またはマスク済み 1'b1：ブロック終了を検出かつマスクなし	R
b3～b0	bMSEBIS_INT1_CPURX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 1'b0：ブロック終了割り込み未検出またはマスク済み 1'b1：ブロック終了を検出かつマスクなし	R

10.3.3.13　rMSEBIS_MASK_INT ― 割り込みマスクレジスタ

本レジスタには、MSEBI スレーブ割り込み専用の割り込みマスクが格納されます。

それぞれのビットは rMSEBIS_STATUS_INT0 レジスタのビットを参照しています。

アドレス　400C 280Ch

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIS_MASK_INT_DMATX		―	―	bMSEBIS_MASK_INT_DMARX		bMSEBIS_MASK_INT_CPUTX				bMSEBIS_MASK_INT_CPURX			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.41　rMSEBIS_MASK_INT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット	読むと 0 が読み出されます。	R
b13、b12	bMSEBIS_MASK_INT_DMATX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 1'b0：ブロック終了割り込み禁止 1'b1：ブロック終了割り込み許可	R/W
b11、b10	予約ビット	読むと 0 が読み出されます。	R
b9、b8	bMSEBIS_MASK_INT_DMARX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 1'b0：ブロック終了割り込み禁止 1'b1：ブロック終了割り込み許可	R/W
b7～b4	bMSEBIS_MASK_INT_CPUTX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 1'b0：ブロック終了割り込み禁止 1'b1：ブロック終了割り込み許可	R/W
b3～b0	bMSEBIS_MASK_INT_CPURX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 1'b0：ブロック終了割り込み禁止 1'b1：ブロック終了割り込み許可	R/W

10.3.3.14　rMSEBIS_CLR_INT ― 割り込みクリアレジスタ

CPU が本レジスタのいずれかのビットに 1 を書き込むと、rMSEBIS_STATUS_INT0 レジスタの対応する割り込み要因がクリアされます。

備　考

本レジスタを用いて割り込みをクリアすると、rMSEBIS_INT レジスタの対応するビットがクリアされます。

アドレス　400C 2810h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIS_CLR_INT_DMATX		―	―	bMSEBIS_CLR_INT_DMARX		bMSEBIS_CLR_INT_CPUTX				bMSEBIS_CLR_INT_CPURX			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.42　rMSEBIS_CLR_INT レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット	読むと 0 が読み出されます。	R
b13、b12	bMSEBIS_CLR_INT_DMATX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 1 を書き込むと割り込みステータスビットがクリアされます。	R/W
b11、b10	予約ビット	読むと 0 が読み出されます。	R
b9、b8	bMSEBIS_CLR_INT_DMARX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 1 を書き込むと割り込みステータスビットがクリアされます。	R/W
b7～b4	bMSEBIS_CLR_INT_CPUTX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 1 を書き込むと割り込みステータスビットがクリアされます。	R/W
b3～b0	bMSEBIS_CLR_INT_CPURX	本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 1 を書き込むと割り込みステータスビットがクリアされます。	R/W

10.3.3.15　rMSEBIS_EOB_ADDR ― ブロック終了アドレスレジスタ

備　考

CS[n]_N（n＝0～3）のディスクリプタに対するアドレスは自動的に計算されます。
「**表 10.43 rMSEBIS_EOB_ADDR レジスタの内容**」を参照してください。

注　意

設定を切り替える前に、（対応するチップセレクトの）bMSEBIS_BUSY＝0 を読み出すことによって、アクセスが進行中でないことを確認する必要があります。本レジスタを更新する場合は、すべての MSEBI_CS[n]_N チップセレクトの bMSEBIS_CS_ENABLE ビットをクリアしてから更新する必要があります。

アドレス　400C 2814h

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIS_EOB_ADDR															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIS_EOB_ADDR														—	—
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.43　rMSEBIS_EOB_ADDR レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b2	bMSEBIS_EOB_ADDR	MSEBI_CS[n]_N に対して、メモリへの書き込み転送を完了するために使用されるディスクリプタのアドレスが格納されます。 ● CPU モード：n＝0～3 ● DMA モード：n＝0、1 本アドレスは、以下のように、ディスクリプタのアドレスを計算するためにも使用されます。 CPU 用の rMSEBIS_EOB_ADDR の場合 rMSEBIS_EOB_ADDR：　MSEBI_CS0_N を参照 rMSEBIS_EOB_ADDR＋32'h04：MSEBI_CS1_N を参照 rMSEBIS_EOB_ADDR＋32'h08：MSEBI_CS2_N を参照 rMSEBIS_EOB_ADDR＋32'h0C：MSEBI_CS3_N を参照 DMA 用の rMSEBIS_EOB_ADDR の場合 rMSEBIS_EOB_ADDR＋32'h10：MSEBI_CS0_N を参照 rMSEBIS_EOB_ADDR＋32'h14：MSEBI_CS1_N を参照 MSEBI バスのマスタから「ブロック終了イベント」を受信したとき、MSEBI スレーブコントローラは「rMSEBIS_EOB_ADDR＋オフセット」のアドレス位置にブロック転送ディスクリプタを書き込みます（AHB バスを介して 0x1234_5678 を書き込みます）。MSEBI マスタは、下記の状態に依存して、rMSEBIS_INT レジスタの bMSEBIS_SET_INT_CPUTX ビットまたは bMSEBIS_SET_INT_DMATX ビットに書き込みます。 ● 転送のイニシエータ（CPU または DMA） ● 送受信側（TX） ● 転送に用いられた MSEBI_CS[n]_N（n＝0～3）	R/W

表 10.43　rMSEBIS_EOB_ADDR レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
		備考） スレーブメモリ上のディスクリプタを読み出した後、そのディスクリプタを 32'h0 にクリアし、さらに bMSEBIS_SET_INT_CPUTX ビットまたは bMSEBIS_SET_INT_DMATX ビットをリセットして、次の転送を可能にする必要があります。 「**10.4.5.3 MSEBI 割り込み：スレーブによるブロック終了検出**」を参照してください。	
b1、b0	予約ビット	これらのビット[1:0]は 0 にリセットされます（32 ビットアライメント）。	R

10.3.4　レジスタの説明：MSEBI から MSEBI スレーブ

10.3.4.1　rMSEBIS_INT ― スレーブ割り込みレジスタ

注　意

本レジスタは CPU からアクセスできません。MSEBI バスアクセス用に予約されています。MSEBI マスタは、「**10.2.4 レジスタマップ：MSEBI から MSEBI スレーブ**」に応じて MSEBI_CS[n]_N（n＝0～3）により、各 MSEBI スレーブ内の本レジスタにアクセスが可能です。

アドレス　400C 1000h（MSEBI_CS0_N アクセス）
400C 1400h（MSEBI_CS1_N アクセス）
400C 1800h（MSEBI_CS2_N アクセス）
400C 1C00h（MSEBI_CS3_N アクセス）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	bMSEBIS_SET_INT_DMATX		―	―	bMSEBIS_SET_INT_DMARX		bMSEBIS_SET_INT_CPUTX				bMSEBIS_SET_INT_CPURX			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.44　rMSEBIS_INT レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b14	予約ビット	読むと 0 が読み出されます。	R
b13、b12	bMSEBIS_SET_INT_DMATX	下記の MSEBI スレーブ割り込みを設定します。 DMA TX チャネルでの「ブロック終了イベント」検出 本フィールドに書き込む場合： ● 最初に、ブロック転送ディスクリプタがメモリ上の対応する MSEBI_CS[n]_N（n＝0、1）のアドレス（下記レジスタで指定された DMA モードでのオフセット位置）に書き込まれます。 – MSEBIS_EOB_ADDR – 「**図 10.42 MSEBI スレーブ：MSEBI CPU マスタからの書き込みブロック転送の終了**」を参照してください。 1’b0：0 の書き込みは無効 1’b1：1 の書き込みは、rMSEBIS_STATUS_INT0 レジスタの対応する「ブロック終了イベント」フラグをセット rMSEBIS_STATUS_INT0 レジスタの対応するビットの値が読み出されます。 本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 rMSEBIS_CLR_INT レジスタの対応するビットをクリアすると、本ビットフィールドがクリアされます。	R/W
b11、b10	予約ビット	読むと 0 が読み出されます。	R

表 10.44　rMSEBIS_INT レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b9、b8	bMSEBIS_SET_INT_DMARX	下記の MSEBI スレーブ割り込みを設定します。 DMA RX チャネルでの「ブロック終了イベント」検出 本レジスタに書き込む場合： 1'b0：0 の書き込みは無効 1'b1：1 の書き込みは、rMSEBIS_STATUS_INT0 レジスタの対応する「ブロック終了イベント」フラグをセット rMSEBIS_STATUS_INT0 レジスタの対応するビットの値が読み出されます。 本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0、1）を参照しています。 rMSEBIS_CLR_INT レジスタの対応するビットをクリアすると、本ビットフィールドもクリアされます。	R/W
b7～b4	bMSEBIS_SET_INT_CPUTX	下記の MSEBI スレーブ割り込みを設定します。 MSEBI マスタの CPU TX チャネルでの「ブロック終了イベント」検出 本フィールドに書き込む場合： ● 最初に、ブロック転送ディスクリプタがメモリ上の対応する MSEBI_CS[n]_N（n＝0～3）のアドレス（下記レジスタで指定された CPU モードでのオフセット位置）に書き込まれます。 – rMSEBIS_EOB_ADDR – 「**図 10.42 MSEBI スレーブ：MSEBI CPU マスタからの書き込みブロック転送の終了**」を参照してください。 1'b0：0 の書き込みは無効 1'b1：1 を書き込むと、rMSEBIS_STATUS_INT0 レジスタの対応する「ブロック終了イベント」フラグがセットされる rMSEBIS_STATUS_INT0 レジスタの対応するビットの値が読み出されます。 本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 rMSEBIS_CLR_INT レジスタの対応するビットをクリアすると、本ビットフィールドがクリアされます。	R/W
b3～b0	bMSEBIS_SET_INT_CPURX	下記の MSEBI スレーブ割り込みを設定します。 MSEBI マスタの CPU RX チャネルでの「ブロック終了イベント」検出 本レジスタに書き込む場合： 1'b0：0 の書き込みは無効 1'b1：1 を書き込むと、rMSEBIS_STATUS_INT0 レジスタの対応する「ブロック終了イベント」フラグがセットされる rMSEBIS_STATUS_INT0 レジスタの対応するビットの値が読み出されます。 本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。 rMSEBIS_CLR_INT レジスタの対応するビットをクリアすると、本ビットフィールドがクリアされます。	R/W

10.3.4.2　rMSEBIS_STATUS ― スレーブステータスレジスタ

注　意

本レジスタは CPU からアクセスできません。MSEBI バスアクセス用に予約されています。MSEBI マスタは、「**10.2.4 レジスタマップ : MSEBI から MSEBI スレーブ**」に応じて MSEBI_CS[n]_N（n＝0～3）により、各 MSEBI スレーブ内の本レジスタにアクセスが可能です。

アドレス　400C 1004h（MSEBI_CS0_N アクセス）
400C 1404h（MSEBI_CS1_N アクセス）
400C 1804h（MSEBI_CS2_N アクセス）
400C 1C04h（MSEBI_CS3_N アクセス）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―	―
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	―	―	―	―	bMSEBIS_ERROR_CS_CONFIGURATION				bMSEBIS_ERROR_WEN				bMSEBIS_ERROR_ADDR			
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.45　rMSEBIS_STATUS レジスタの内容（1/2）

ビット位置	ビット名	機能	R/W
b31～b12	予約ビット	読むと 0 が読み出されます。	R
b11～b8	bMSEBIS_ERROR_CS_CONFIGURATION	MSEBI スレーブコントローラのエラー状態を示します。 下記の潜在的問題がある状態で、MSEBI_CS[n]_N（n＝0～3）専用のコンフィグレーションレジスタグループ内で設定の変更が検出されています。 ● MSEBI のマスタとスレーブの間で交換されたデータが失われた可能性 ● MSEBIS のマスタとスレーブの同期が失われた可能性 rMSEBIS_CONFIG_CS[n]_N レジスタの bMSEBIS_CS_ENABLE ビットフィールドを参照してください。 MSEBI バスのマスタによって読み出す場合： 1’b0：エラーの検出なし 1’b1：MSEBI スレーブコントローラが設定の変更を検出し、データと同期が失われた可能性あり MSEBI バスのマスタによってクリアする場合： 1’b0 の書き込みは無効です。 1’b1 の書き込みはフラグをクリアします。 本ビットは、新しい設定がスレーブに正しくロードされたことを MSEBI マスタによってチェックするために利用できます。 本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。	R/W
b7～b4	bMSEBIS_ERROR_WEN	MSEBI スレーブコントローラのエラー状態を示します。 スレーブで書き込み保護が有効になっているとき、バスのマスタからの書き込みアクセスが検出されています。rMSEBIS_CONFIG_CS[n]_N レジスタの bMSEBIS_WEN ビットフィールドを参照してください。 MSEBI バスのマスタによって読み出す場合： 1’b0：エラーの検出なし 1’b1：MSEBI スレーブコントローラが書き込みアクセス時にエラーを検出 MSEBI バスのマスタによってクリアする場合： 1’b0 の書き込みは無効です。 1’b1 の書き込みはフラグをクリアします。 本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。	R/W

表 10.45　rMSEBIS_STATUS レジスタの内容（2/2）

ビット位置	ビット名	機能	R/W
b3～b0	bMSEBIS_ERROR_ADDR	MSEBI スレーブコントローラのエラー状態を示します。 AHB マスタポートに対するアクセス中に、NoC が不正なアドレスレンジに起因したエラーを通知しています。これは、バスのマスタのエラーか、またはアドレス指定モードの設定上のエラーが原因かもしれません。rMSEBIS_CONFIG_CS[n]_N レジスタの bMSEBIS_ADDR_MODE ビットフィールドを参照してください。 MSEBI バスのマスタによって読み出す場合： 1'b0：エラーの検出なし 1'b1：MSEBI スレーブコントローラが不正アドレスへのアクセスを検出 MSEBI バスのマスタによってクリアする場合： 1'b0 の書き込みは無効です。 1'b1 の書き込みはフラグをクリアします。 本フィールドのビット[n]は MSEBI_CS[n]_N（n＝0～3）を参照しています。	R/W

10.3.4.3　rMSEBIS_ID_CS[n]_N ― スレーブ ID レジスタ（n＝0～3）

チップセレクトが利用可能かどうかを判定するために、マスタによって使用されます。

注　意

本レジスタは CPU からアクセスできません。MSEBI バスアクセス用に予約されています。MSEBI マスタは、「**10.2.4 レジスタマップ：MSEBI から MSEBI スレーブ**」に応じて MSEBI_CS[n]_N（n＝0～3）により、各 MSEBI スレーブ内の本レジスタにアクセスが可能です。

アドレス　400C 1008h＋4h×n（MSEBI_CS0_N アクセス）
400C 1408h＋4h×n（MSEBI_CS1_N アクセス）
400C 1808h＋4h×n（MSEBI_CS2_N アクセス）
400C 1C08h＋4h×n（MSEBI_CS3_N アクセス）

ビット	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	bMSEBIS_ID															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ビット	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bMSEBIS_ID															
リセット後の値	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

表 10.46　rMSEBIS_ID_CS[n]_N レジスタの内容

ビット位置	ビット名	機能	R/W
b31～b0	bMSEBIS_ID	本レジスタは、チップセレクトが通信可能な状態にあるときを判定するため、バスのマスタによって読み出されます。 bMSEBIS_CS_ENABLE＝1 の場合： ● マスタが rMSEBIS_ID_CS[n]_N（n＝0～3）レジスタを読むと、0x1234_FEDn（n＝0～3）が読み出されます。 bMSEBIS_CS_ENABLE＝0 の場合： ● マスタが rMSEBIS_ID_CS[n]_N（n＝0～3）レジスタを読むと、0x0 が読み出されます。 「**10.4.7.5 MSEBI スレーブ：チップセレクトの設定状態**」を参照してください。	R

10.4　動作説明

10.4.1　AHB インタフェース

10.4.1.1　AHB スレーブインタフェース

MSEBI マスタと MSEBI スレーブはそれぞれ独立した AHB スレーブを持っています。

MSEBI マスタ用

MSEBI は 8/16/32 ビットの AHB アクセスをサポートしています。そのため、1 回の AHB 32 ビットアクセスで、1 回の MSEBI アクセス（モード 32）または 2 回の MSEBI アクセス（モード 16）、または 4 回の MSEBI アクセス（モード 8）を発生させることができます。AHB バーストアクセス時は、AHB アクセスサイズを MSEBI バスサイズ以上にしてください。

MSEBI スレーブ用

MSEBI は 8/16/32 ビットの AHB アクセスをサポートしています。

10.4.1.2　AHB マスタインタフェース（MSEBI スレーブのみ）

バーストサイズ（4/8/16 ワード）は下記のレジスタで管理できます。

- rMSEBIS_CONFIG（CPU のみ）
- rMSEBIS_DMARDLR_CS[n]_N（DMA 受信、n＝0、1）
- rMSEBIS_DMATDLR_CS[n]_N（DMA 送信、n＝0、1）

AHB アクセスの属性は下記のビットで管理できます。

- bMSEBIS_AHB_MASTER_BUF
- bMSEBIS_AHB_MASTER_CACHE

AHB マスタは、不正アドレスが生じた場合に、NoC からの「エラー」応答を管理できます。

rMSEBIS_STATUS レジスタの bMSEBIS_ERROR_ADDR ビットを参照してください。

10.4.2　デバイス接続の使用例

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。

信号名	説明
MSEBI(x)_ACD[31:0]	アドレス、制御データ、およびデータは 32/16/8 ビットでマルチプレクスされ、MSEBIM_ALE、MSEBIM_CLE、MSEBIM_DLE によって制御されます。また、MSEBIM_CLK クロックの立ち上がりエッジでラッチされます。
MSEBI(x)_CLK	● マスタモードでは、本クロックは以下の設定が可能です。 MSEBIM_HCLK の 2、3、4、5、6、7、または 8 分周 ● スレーブモードでは、本クロックはバスのマスタによって供給されます。 **注意）** クロックのデューティ比は 50%でない場合があります。
MSEBI(x)_WAIT[n]_N	**マスタモードの場合** ● MSEBIM_WAIT[n]_N は、bMSEBIM_CONFIG ビットを用いて同期式モードまたは非同期式モードに設定できます。 ● MSEBIM_WAIT[n]_N は、MSEBI_CS[n]_N（n＝0～3）が Low かつ MSEBIM_DLE が High のとき、下記のいずれかの場合に VALID サブフェーズ中に参照されます。 – 同期式モードにおけるバーストアクセスの最初のアクセス、または非同期式／同期式モードにおける非バーストアクセスに対して、RDDLEDATA_NB/WRDLEDATA_NB の期間が過ぎた場合 – 同期式モードにおける最初のアクセス後のすべてのバーストアクセスに対して、RDDLEDATA_B/WRDLEDATA_B の期間が過ぎた場合 **スレーブモードの場合** ＜同期式モードのみ＞ ● MSEBIS_WAIT[n]_N は、MSEBI_CS[n]_N（n＝0～3）が Low かつ MSEBIS_DLE が High のとき、下記のいずれかの場合に VALID サブフェーズ中に生成されます。 – バーストアクセスの最初のアクセス、または非バーストアクセスに対して、RDDLEDATA_NB/WRDLEDATA_NB の期間が過ぎた場合 – 同期式モードにおける最初のアクセス後のすべてのバーストアクセスに対して、RDDLEDATA_B/WRDLEDATA_B の期間が過ぎた場合
MSEBI(x)_CLE	アドレス＆コントロールラッチイネーブル（アクティブ High）
MSEBI(x)_DLE	データラッチイネーブル（アクティブ High） 最初のデータ転送は、RDDLEDATA_NB（読み出し時）または WRDLEDATA_NB（書き込み時）の期間実行され、その後（バーストモードでのみ）、MSEBI_DLE がアクティブ状態を維持していれば、RDDLEDATA_B（読み出し時）または WRDLEDATA_B（書き込み時）によってプログラミングされたバースト間隔で連続転送が実行されます。バーストはリニアアドレスインクリメント方式であるため、長さは不定です。
MSEBI(x)_ALE	アドレスラッチイネーブル（アクティブ High）
MSEBIM_ALE1	アドレスラッチイネーブル（アクティブ High） パラレルモードのモード 8 またはモード 16 で動作する、マスタに対してのみ使用されます。
MSEBIM_ALE2	アドレスラッチイネーブル（アクティブ High） パラレルモードのモード 8 で動作する、マスタに対してのみ使用されます。
MSEBIM_ALE3	アドレスラッチイネーブル（アクティブ High） パラレルモードのモード 8 で動作する、マスタに対してのみ使用されます。
MSEBIM_RD_N	読み出しイネーブル（アクティブ Low） 読み出し時に限って、MSEBI_DLE の反転と同一です。 オプション機能であり、非同期式モードで動作するマスタに対してのみ使用されます。
MSEBIM_WR_N	書き込みイネーブル（アクティブ Low） 書き込み時に限って、MSEBI_DLE の反転と同一です。 オプション機能であり、非同期式モードで動作するマスタに対してのみ使用されます。

10.4.2.1　1 デバイス、モード 32、同期式

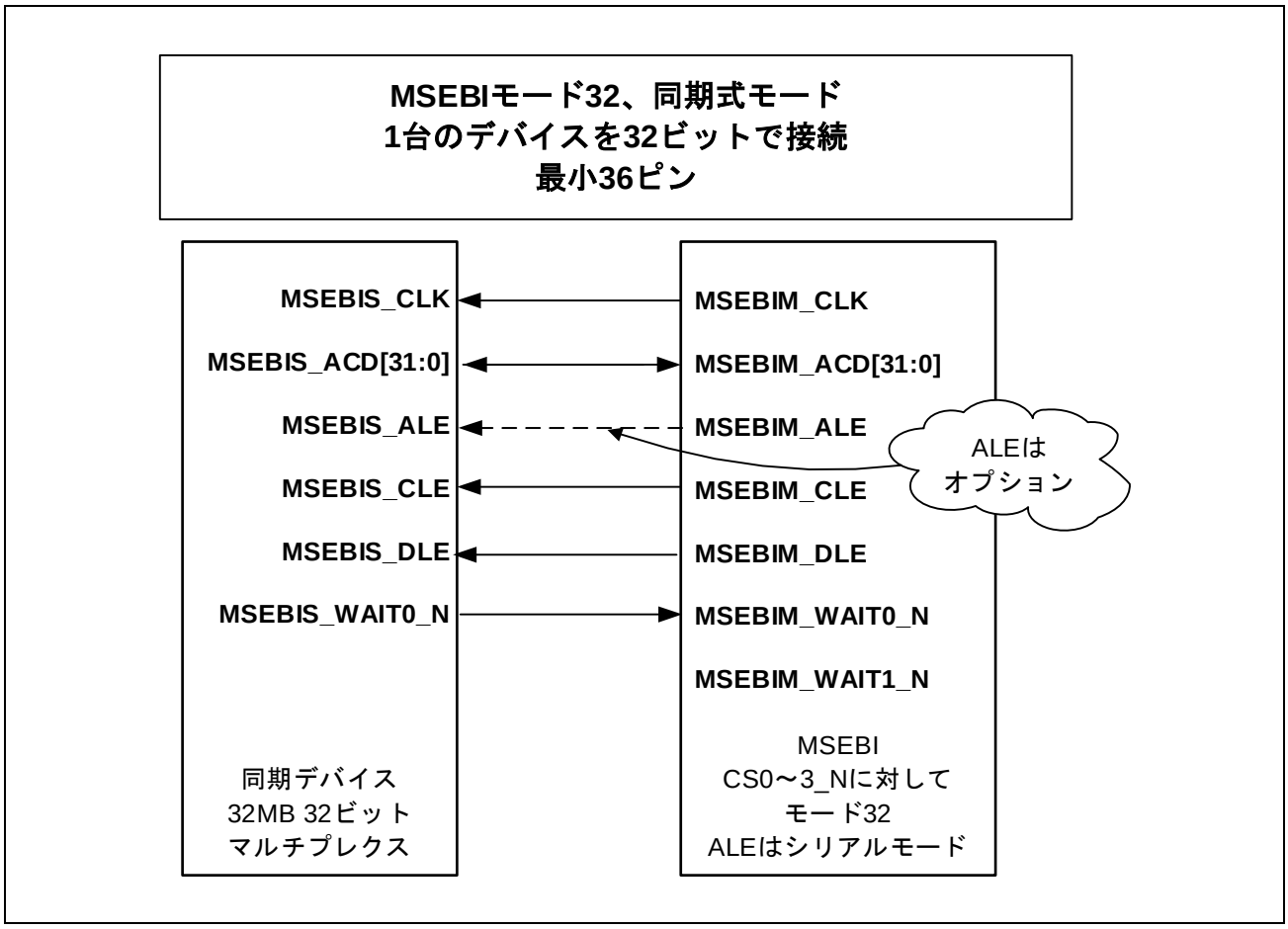


図 10.2　1 デバイス、モード 32、同期式

10.4.2.2　1 デバイス、モード 16、同期式

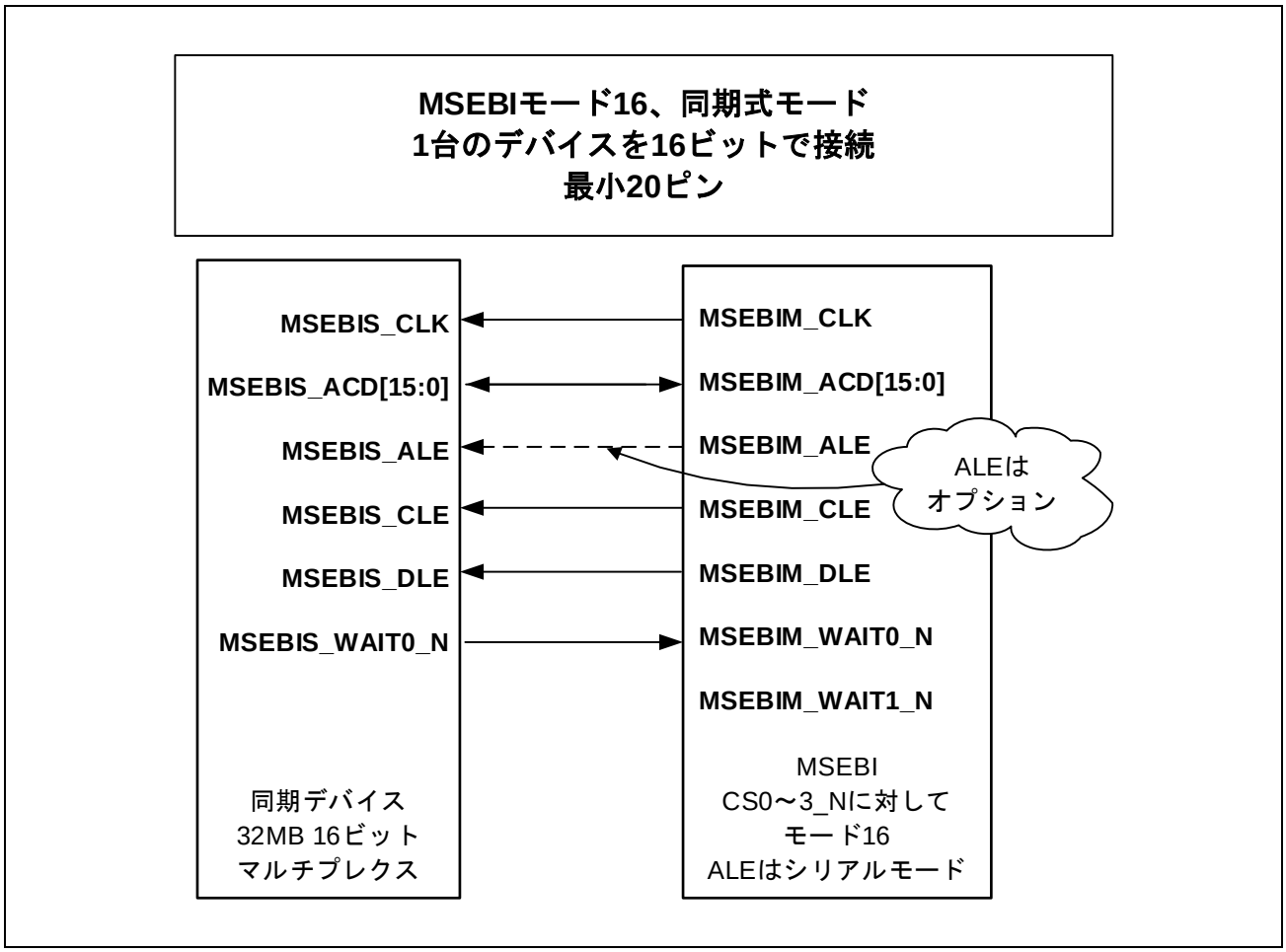


図 10.3　1 デバイス、モード 16、同期式

10.4.2.3　1 デバイス、モード 8、同期式

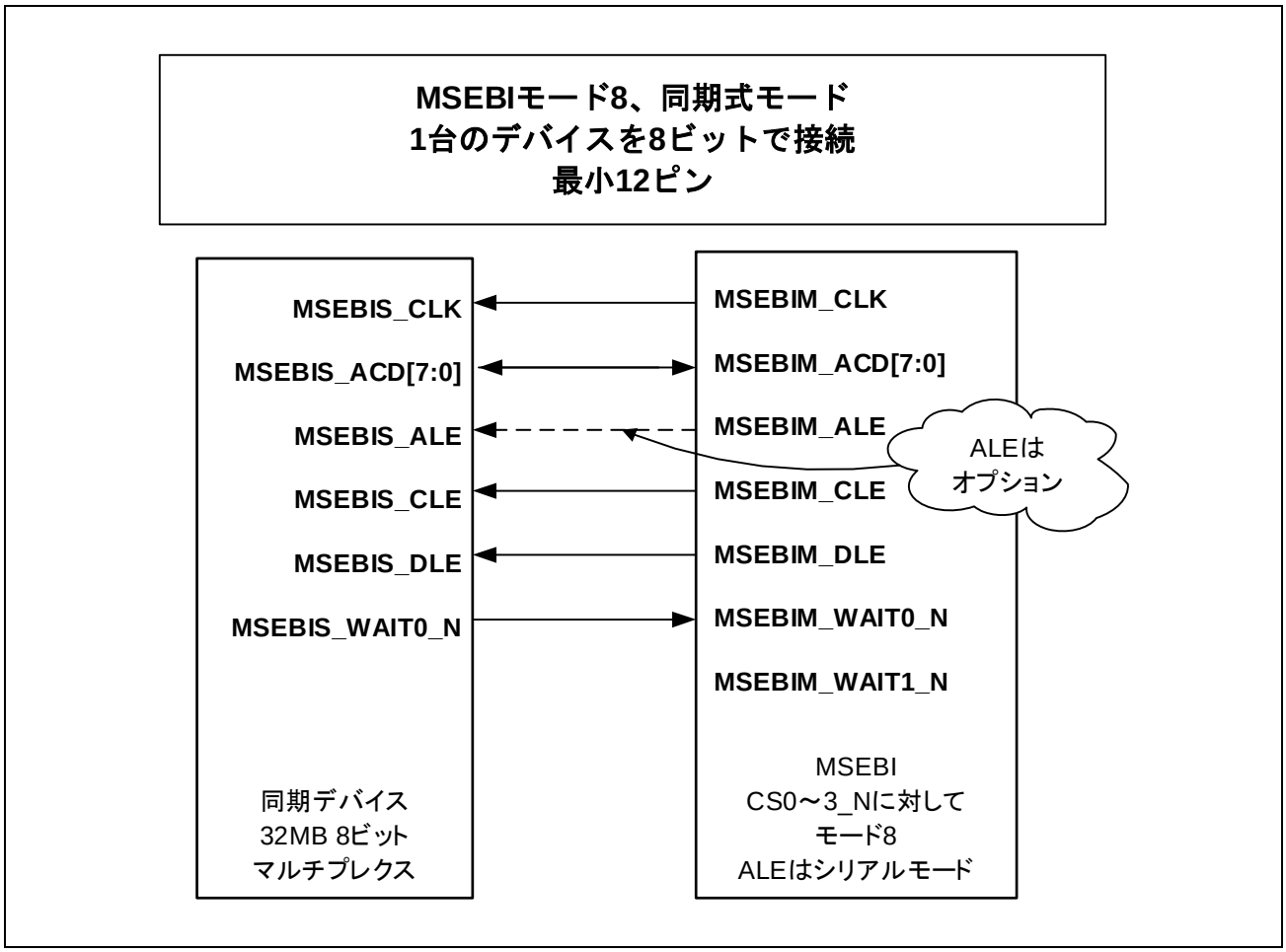


図 10.4　1 デバイス、モード 8、同期式

10.4.2.4　3 デバイス、モード 8/16/32、同期式

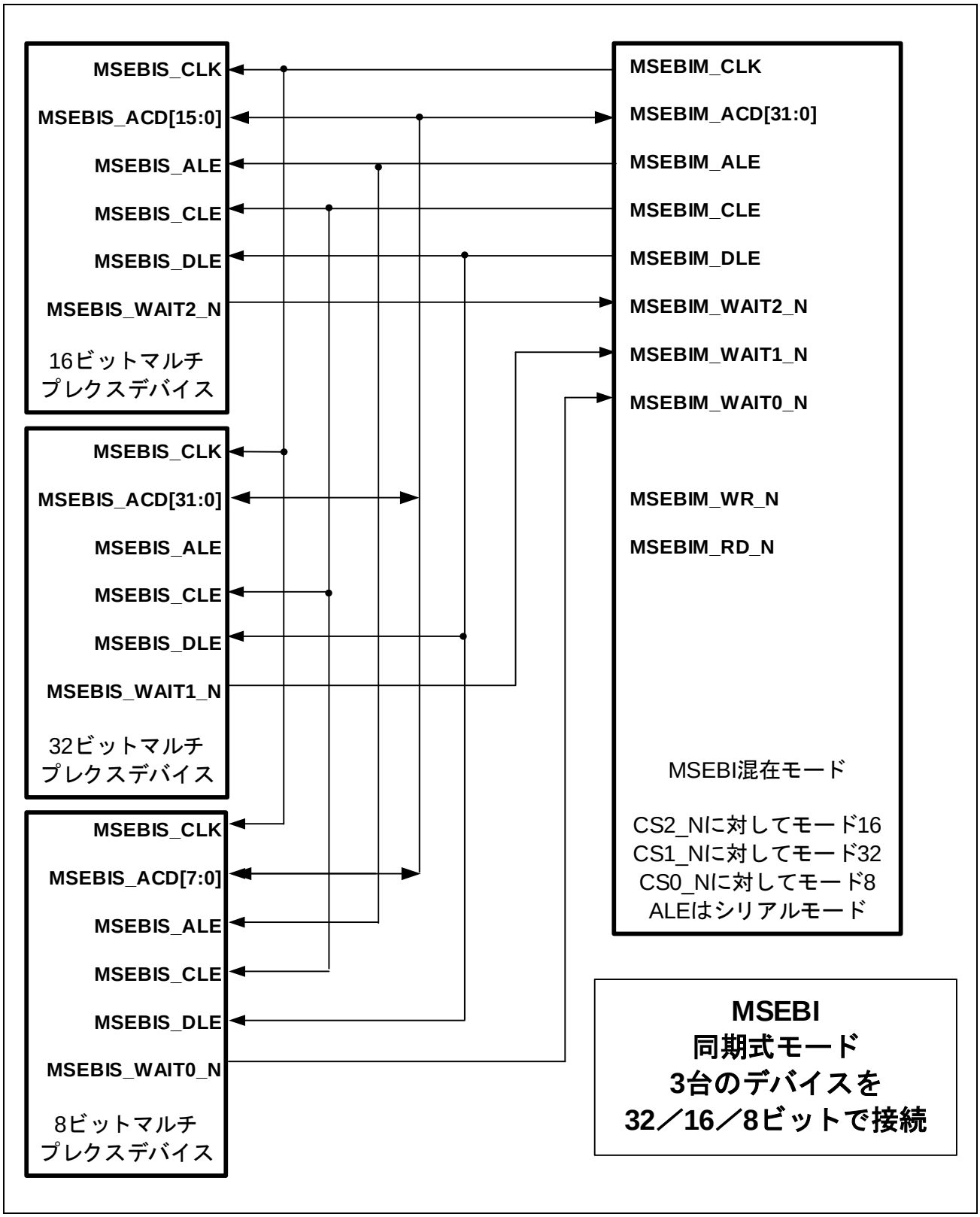


図 10.5　　3 デバイス、モード 8/16/32、同期式

10.4.2.5　3 デバイス、モード 8/16/32、非同期式

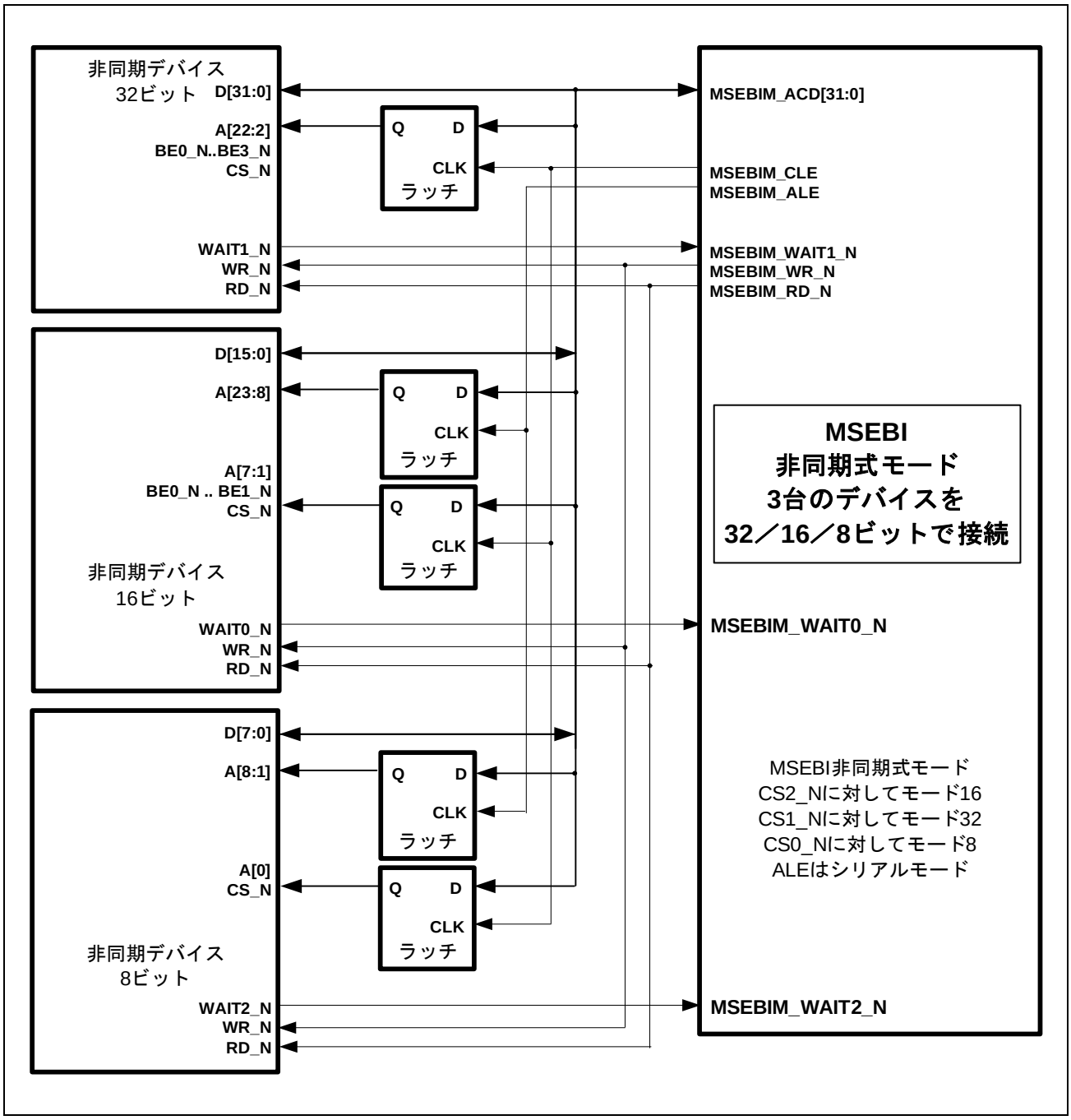


図 10.6　3 デバイス、モード 8/16/32、非同期式

10.4.2.6　3 デバイス、モード 8/16/32、同期式と非同期式の混在

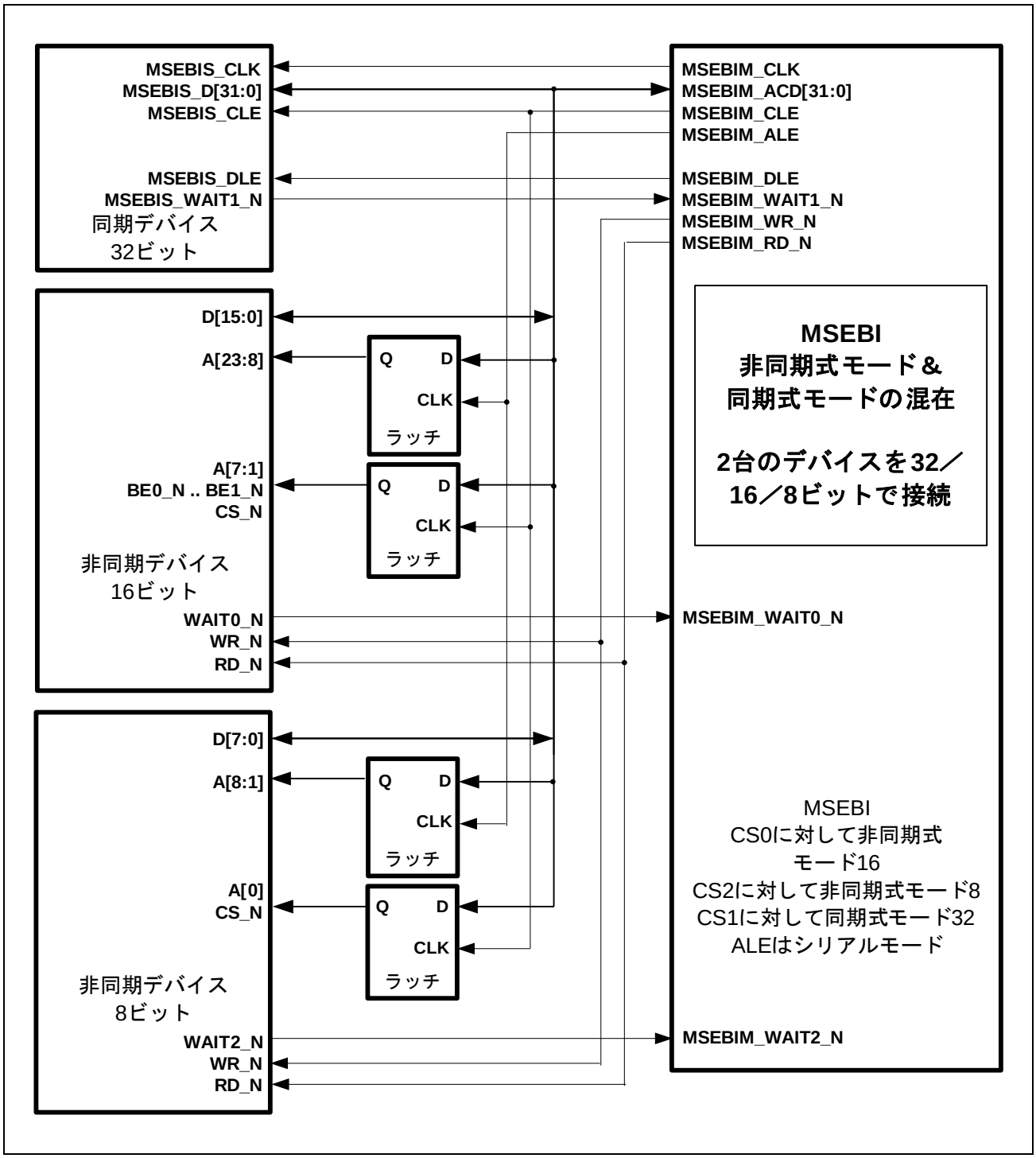


図 10.7　　3 デバイス、モード 8/16/32、同期式と非同期式の混在

10.4.2.7　1 デバイス、モード 8、非同期式、パラレルモードの ALE

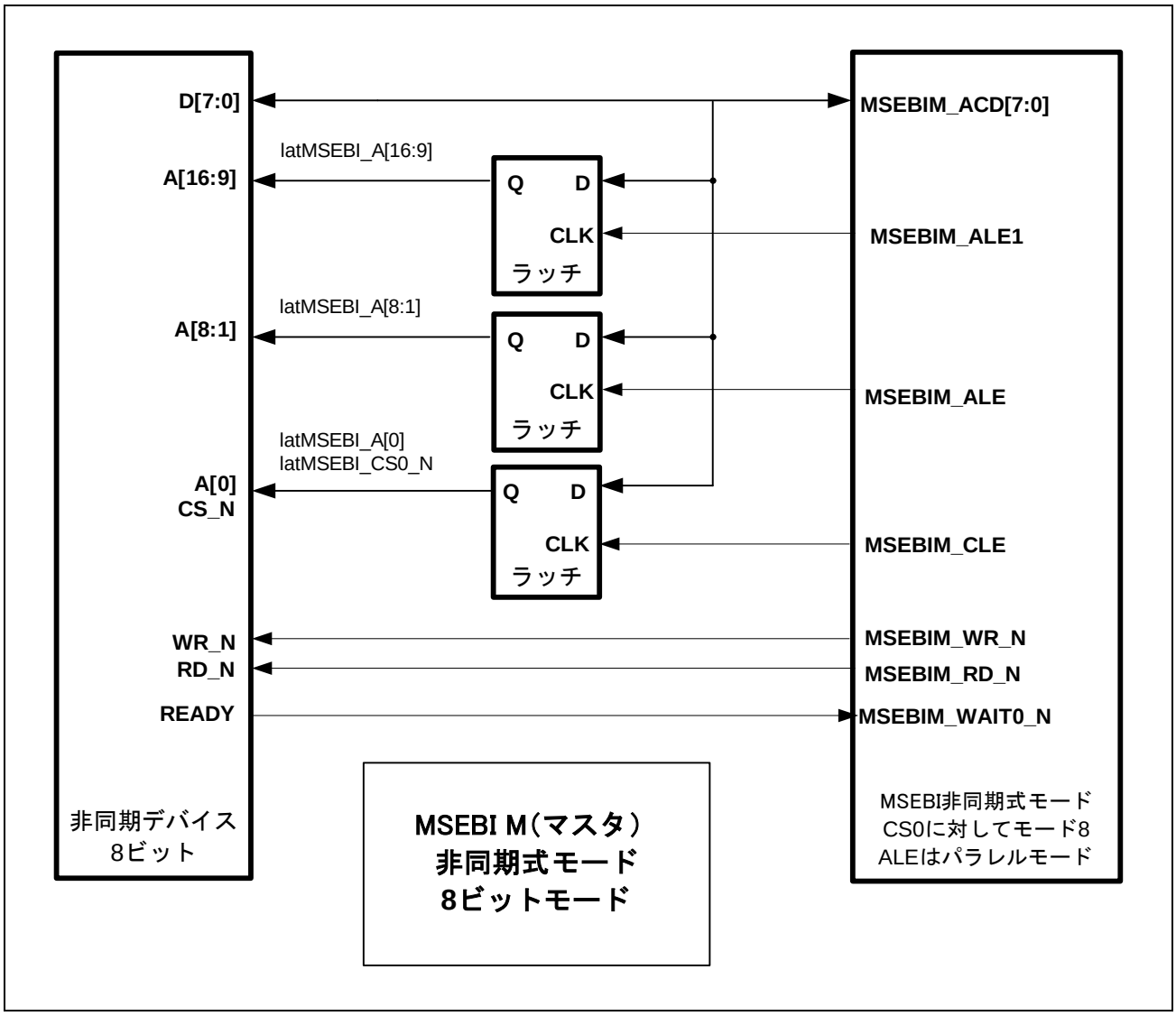


図 10.8　　1 デバイス、モード 8、非同期式、パラレルモードの ALE

10.4.3　ADDRESS、CONTROL、および DATA フェーズの基本原理

アクセスは以下の 3 つの主要フェーズで成り立っています。

- アドレスラッチ（ADDRESS フェーズ）は、MSEBIM_ALE（マスタの場合）または MSEBIS_ALE（スレーブの場合）によって制御されますが、本フェーズはオプションです。
- コントロールラッチ（CONTROL フェーズ）は MSEBIM_CLE（マスタの場合）または MSEBIS_CLE（スレーブの場合）によって制御されます。
- データ転送（DATA フェーズ）は MSEBIM_DLE（マスタの場合）または MSEBIS_DLE（スレーブの場合）によって制御されます。

下記の信号がマスタによって制御および設定されます。

- MSEBIM_CLK
- MSEBIM_ALE
- MSEBIM_CLE
- MSEBIM_DLE

下記の信号がバスのマスタによって駆動されます。スレーブ側の設定パラメータは、マスタのパラメータに従う必要があります。

- MSEBIS_CLK
- MSEBIS_ALE
- MSEBIS_CLE
- MSEBIS_DLE

注　意

同期に関する制限から、MSEBIS_CLK の周波数（MSEBI マスタによって発生）は、MSEBIS_HCLK（AHB バスクロック）の周波数より低くなくてはいけません。

10.4.3.1　アドレスラッチ ALE フェーズ（ADDRESS）

MSEBIM_ALE、MSEBIM_ALE1、MSEBIM_ALE2、MSEBIM_ALE3 は使用するモードに従って、アドレスの一部をバスからラッチするために用いられます。

本フェーズはオプションであり、アドレスビットの割り当ては、使用する MSEBI_ALE フェーズの数で決まります（モードに従って 0～4）。

- 「**表 10.4 MSEBI モード 32、ACD31～0 に対するマルチプレクサ機能**」を参照してください。
- 「**表 10.6 MSEBI モード 16、ACD15～0 に対するマルチプレクサ機能**」を参照してください。
- 「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」を参照してください。

マスタモード時のみ

本フェーズの長さは、CS[n]_N（n＝0～3）ごとに MSEBIM_CLK の 1～2 周期に設定が可能です（マスタによってのみ設定されます）。

非同期式モードでは、MSEBIM_ALE、MSEBIM_ALE1、MSEBIM_ALE2、MSEBIM_ALE3 をラッチの入力に接続することにより、外部ラッチ（74x16373 タイプ）を使用してアドレス部のラッチが可能です。外部ラッチのホールド時間を確保するため、MSEBI_ALE フェーズの長さは 2 を使用するか、または同期ラッチを使用してください。MSEBIM_ALE、MSEBIM_ALE1、MSEBIM_ALE2、MSEBIM_ALE3 は、本フェーズの第 1 クロック周期中にのみ駆動されます。

本フェーズはオプションです。アドレスビットの割り当ては、使用する MSEBI_ALE フェーズの数で決まります。これによって、アドレス機能に最大限の柔軟性をもたせることが可能です。

MSEBI_ALE は、bMSEBIM_ALE_MODE ビットを用いてシリアルモードまたはパラレルモードに設定できます。

- 同期式インタフェースではシリアルモードが推奨されます。
  - 「**図 10.15 MSEBI タイミング、非同期式モード、読み出し、ウェイトなし、非バースト、ALE 数 2（シリアルモード）**」および「**図 10.17 MSEBI タイミング、非同期式モード、書き込み、ウェイトなし、非バースト、ALE 数 2（シリアルモード）**」を参照してください。
- パラレルモードでは非同期式インタフェースと外部ラッチ（74x16373 タイプ）接続が推奨されます。これによってボードのコストが最小限に抑えられます。
  - 「**図 10.16 MSEBI タイミング、非同期式モード、読み出し、ウェイトなし、非バースト、ALE 数 2（パラレルモード）**」および「**図 10.18 MSEBI タイミング、非同期式モード、書き込み、ウェイトなし、非バースト、ALE 数 2（パラレルモード）**」を参照してください。

《モード 8 での CPU によるアクセスの例》

［ケース 1］

MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A8～A1
- 2 回目のアクセス：A16～A9
- 3 回目のアクセス：A24～A17
- 4 回目のアクセス：A31～A25
- A0 は MSEBIM_CLE フェーズで生成され、4GB のアドレス機能が可能になります。
- 4 つのチップセレクト CS[n]_N（n＝0～3）が利用可能です。

- MSEBI_A26～MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A27 はレジスタによって指定されます

「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」を参照してください。

［ケース 2］

3 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A8～A1
- 2 回目のアクセス：A16～A9
- 3 回目のアクセス：A24～A17
- A0 は MSEBIM_CLE フェーズで生成され、32MB のアドレス機能が可能になります。
- 4 つのチップセレクト CS[n]_N（n＝0～3）が利用可能です。
- MSEBI_A24～MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A25 は本構成では使用されません。

「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」を参照してください。

［ケース 3］

2 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A8～A1
- 2 回目のアクセス：A16～A9
- A0 は MSEBIM_CLE フェーズで生成され、128KB のアドレス機能が可能になります。
- 4 つのチップセレクト CS[n]_N（n＝0～3）が利用可能です。
- MSEBI_A16～MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A17 は本構成では使用されません。

「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」を参照してください。

［ケース 4］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A8～A1
- A0 は MSEBIM_CLE フェーズで生成され、512B のアドレス機能が可能になります。
- 4 つのチップセレクト CS[n]_N（n＝0～3）が利用可能です。
- MSEBI_A8～MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A9 は本構成では使用されません。

「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」を参照してください。

［ケース 5］

MSEBI_ALE フェーズなし：

- A0 は MSEBIM_CLE フェーズで生成され、2B のアドレス機能が可能になります。

- 4 つのチップセレクト CS[n]_N（n＝0～3）が利用可能です。
- MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A1 は本構成では使用されません。

「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」を参照してください。

各 CS[n]_N（n＝1～3）をアドレスビットとして使用することにより、ユーザの使用例に合わせてアドレス機能を拡張できます。

以下の表には、すべてのアドレス機能の詳細が示されています。

- 「**表 10.5 MSEBI モード 32、チップセレクト管理**」を参照してください。
- 「**表 10.7 MSEBI モード 16、チップセレクト管理**」を参照してください。
- 「**表 10.9 MSEBI モード 8、チップセレクト管理**」を参照してください。

《モード 8 での CPU によるアクセスの例》

［ケース 1］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A8～A1
- A0 は MSEBIM_CLE フェーズで生成され、512B のアドレス機能が可能になります。
- 4 つのチップセレクト CS[n]_N（n＝0～3）が利用可能です。
- MSEBI_A8～MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A9 は本構成では使用されません。

「**表 10.9 MSEBI モード 8、チップセレクト管理**」を参照してください。

［ケース 2］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A8～A1
- CS3_N は自動的に A9 をマップするよう設定
- A0 は MSEBIM_CLE フェーズで生成され、1KB のアドレス機能が可能になります。
- 3 つのチップセレクト CS[n]_N（n＝0～2）が利用可能です。
- MSEBI_A9～MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A10 は本構成では使用されません。

「**表 10.9 MSEBI モード 8、チップセレクト管理**」を参照してください。

［ケース 3］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A8～A1
- CS3_N は自動的に A9 をマップするよう設定
- CS2_N は自動的に A10 をマップするよう設定

- A0 は MSEBIM_CLE フェーズで生成され、2KB のアドレス機能が可能になります。
- 2 つのチップセレクト CS[n]_N（n＝0、1）が利用可能です。
- MSEBI_A10～MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A11 は本構成では使用されません。

「**表 10.9 MSEBI モード 8、チップセレクト管理**」を参照してください。

［ケース 4］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A8～A1
- CS3_N は自動的に A9 をマップするよう設定
- CS2_N は自動的に A10 をマップするよう設定
- CS1_N は自動的に A11 をマップするよう設定
- A0 は MSEBIM_CLE フェーズで生成され、4KB のアドレス機能が可能になります。
- 1 つのチップセレクト CS0_N が利用可能です。
- MSEBI_A11～MSEBI_A0 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A12 は本構成では使用されません。

「**表 10.9 MSEBI モード 8、チップセレクト管理**」を参照してください。

《モード 16 での CPU によるアクセスの例》

［ケース 1］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A23～A8
- A7～A1 は MSEBIM_CLE フェーズで生成され、16MB のアドレス機能が可能になります。
- 4 つのチップセレクト CS[n]_N（n＝0～3）が利用可能です。
- MSEBI_A23～MSEBI_A1 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A24 は本構成では使用されません。

「**表 10.7 MSEBI モード 16、チップセレクト管理**」を参照してください。

［ケース 2］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A23～A8
- CS3_N は自動的に A24 をマップするよう設定
- A7～A1 は MSEBIM_CLE フェーズで生成され、32MB のアドレス機能が可能になります。
- 3 つのチップセレクト CS[n]_N（n＝0～2）が利用可能です。
- MSEBI_A24～MSEBI_A1 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A25 は本構成では使用されません。

「**表 10.7 MSEBI モード 16、チップセレクト管理**」を参照してください。

［ケース 3］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A23～A8
- CS3_N は自動的に A24 をマップするよう設定
- CS2_N は自動的に A25 をマップするよう設定
- A7～A1 は MSEBIM_CLE フェーズで生成され、64MB のアドレス機能が可能になります。
- 2 つのチップセレクト CS[n]_N（n＝0、1）が利用可能です。
- MSEBI_A25～MSEBI_A1 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A26 は本構成では使用されません。

「**表 10.7 MSEBI モード 16、チップセレクト管理**」を参照してください。

［ケース 4］

1 つの MSEBI_ALE フェーズによって以下のラッチが可能になります。

- 1 回目のアクセス：A23～A8
- CS3_N は自動的に A24 をマップするよう設定
- CS2_N は自動的に A25 をマップするよう設定
- CS1_N は自動的に A26 をマップするよう設定
- A7～A1 は MSEBIM_CLE フェーズで生成され、128MB のアドレス機能が可能になります。
- 1 つのチップセレクト CS0_N が利用可能です。
- MSEBI_A26～MSEBI_A1 は CPU のアドレスが使用されます。
- MSEBI_A31～MSEBI_A27 は本構成では使用されません。

「**表 10.7 MSEBI モード 16、チップセレクト管理**」を参照してください。

(1) ALE フェーズの動作に関する注意点

マスタの場合

ALE フェーズの数はマスタのアドレス指定範囲に直接影響を与えます。

ALE 数が不足しているため、MSEBI バスが CPU の指定するアドレスに到達できない場合、必然的にラップ効果が生じます。

たとえば、マスタが 8 ビットで ALE フェーズ数 1（再ルーティングなし）に設定された場合、マスタが MSEBI バスを介してアクセスできるのは 512B のデータ（[A8:A0]すなわち 0x000001FF～0x00000000 の範囲）です。

CPU がアドレス 0x000002F4（756 番目のバイト）を MSEBI バス上に送出しようとしても、ALE フェーズを増やさなければ、それは不可能です（アドレスは 0x000000F4 になり、ラップが発生しています）。

スレーブの場合

マスタが MSEBI バス上にバーストを生成するよう設定された場合、そのアドレスは、バースト開始時に一度だけスレーブに送信されます。

その後、スレーブはバーストのビートごとに、独自に対応するアドレスを生成します。

それを用いると、マスタは、ALE フェーズ数に基づく正常アドレスレンジの外にあるデータにもアクセスすることが可能です。

しかし、この可能性は信頼できないので使用してはいけません。なぜなら、FIFO 内のデータが十分な数となり、バーストが許可されると、ただちにマスタによってバーストが生成されるためです（最大バースト長のみが設定可能であり、最小バースト長は設定できません）。

バーストのサイズ（さらには、バーストが発生するか否かさえも）予測するのは不可能なので、アドレスレンジ外のデータにアクセスすることは推奨されません。

［例 1］

- マスタが 8 ビットで ALE フェーズ数 1（再ルーティングなし）に設定された場合、マスタが MSEBI バスを介してアクセスできるのは 512B のデータ（[A8:A0]すなわち 0x000001FF～0x00000000 の範囲）です。
- CPU がアドレス 0x000001FF でバーストを開始した場合、スレーブはビートごとに正しくアドレスを管理します。すなわち、スレーブはアドレス 0x000001FF からスタートして、次にアドレス 0x00000200、0x00000201、そして最後に 0x00000202 を生成します。
- バーストを使用しなかった場合は、アドレス 0x000001FF、0x00000000、0x00000001、0x00000002 というように、ラッピングしたシングルアクセスを行う結果になります。

［例 2］

- マスタが 16 ビットで ALE フェーズ数 1（再ルーティングなし）に設定された場合、マスタが MSEBI バスを介してアクセスできるのは 16MB のデータ（[A23:A0]すなわち 0x00FFFFFF～0x00000000 の範囲）です。
- CPU がアドレス 0x00FFFFFF でバーストを開始した場合、スレーブはビートごとに正しくアドレスを管理します。すなわち、スレーブはアドレス 0x00FFFFFF からスタートして、次にアドレス 0x01000000、0x01000001、そして最後に 0x01000002 を生成します。
- バーストを使用しなかった場合は、アドレス 0x00FFFFFF、0x00000000、0x00000001、0x00000002 というように、ラッピングしたシングルアクセスを行う結果になります。

10.4.3.2　コントロールラッチ CLE フェーズ（CONTROL）

アドレスバスの下位部をバスからラッチするため、および R/W_N、BE[n]_N、DMA_N、CSREG_N、CS[n]_N（n＝0～3）（使用するモードに依存）を制御するために、MSEBIM_CLE/MSEBIS_CLE が用いられます。

- 「**表 10.4 MSEBI モード 32、ACD31～0 に対するマルチプレクサ機能**」を参照してください。
- 「**表 10.6 MSEBI モード 16、ACD15～0 に対するマルチプレクサ機能**」を参照してください。
- 「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」を参照してください。

マスタモード時のみ

本フェーズの長さは、CS[n]_N ごとに MSEBIM_CLK/MSEBIS_CLK の 1 または 2 周期に設定が可能です。

非同期式モードでは、MSEBIM_CLE をラッチの入力に接続することにより、外部ラッチ（74x16373 タイプ）を使用してすべての信号のラッチが可能です。外部ラッチのホールド時間を確保するため、MSEBI_CLE フェーズの長さは 2 を使用するか、または同期ラッチを使用してください。MSEBI_CLE は、本フェーズの第 1 クロック周期中にのみ駆動されます。

10.4.3.3　データフェーズ：SETUP＋VALID＋HOLD（DATA）

データフェーズは 3 つのサブフェーズ：SETUP、VALID、および HOLD で成り立っています。

(1) SETUP

［書き込みコマンド時］

SETUP サブフェーズは、書き込みコマンドの開始前に、アドレス、制御データ、データのセットアップ時間を拡張するために使用されます。
rMSEBIM_SETUPHOLD_CS[n]_N レジスタの bMSEBIM_WRDLESETUP ビット、および rMSEBIS_SETUPHOLD_CS[n]_N レジスタの bMSEBIS_WRDLESETUP ビットを参照してください。

［読み出しコマンド時］

SETUP サブフェーズは、読み出しコマンドの開始前に、アドレス、制御信号のセットアップ時間を拡張するために使用されます。本サブフェーズ期間中、外部バッファとのバス競合を避けるため、データバスはフローティングになります。これはターンアラウンド状態に似ています（rMSEBIM_SETUPHOLD_CS[n]_N レジスタの bMSEBIM_RDDLESETUP ビット／rMSEBIS_SETUPHOLD_CS[n]_N レジスタの bMSEBIS_RDDLESETUP ビットを参照してください）。

本フェーズの長さは、CS[n]_N（n＝0～3）ごとに MSEBIM_CLK の 0～63 周期に設定が可能です。

注　意

外部周辺デバイスへの読み出しアクセスには（バス競合を避けるため）長さ”0”を使用しないでください。

(2) VALID

マスタ時

VALID サブフェーズの期間中、MSEBIM_DLE と MSEBIM_RD_N（非同期式モードのみ）、または MSEBIM_WR_N（非同期式モードのみ）はアクティブに駆動されます。

スレーブ時

VALID サブフェーズの期間中、MSEBIS_DLE がアサートされます。

両方に対して（n＝0～3）

シングル転送モードでは、最後の VALID クロック周期の終了時にデータがサンプリングされます。本フェーズの長さは、CS[n]_N ごとに MSEBIM_CLK/MSEBIS_CLK の 1～256 周期に設定が可能です。
マスタについては、rMSEBIM_CYCLESIZE_CS[n]_N レジスタの bMSEBIM_RDDLEDATA_NB ビットを参照してください。
スレーブについては、rMSEBIS_CYCLESIZE_CS[n]_N レジスタの bMSEBIS_RDDLEDATA_NB ビットと bMSEBIS_WRDLEDATA_NB ビットを参照してください。

バーストモード（同期式モードのみ）では、プログラミングされた長さの終了時に、シングル転送モードと同様に最初のデータがサンプリングされます。その後、本フェーズは CS[n]_N ごとに、バーストサイクルの各アクセスに対して 1～4 MSEBIM_CLK の RDDLEDATA_B および WRDLEDATA_B（または、スレーブに対する RDDLEDATA_B および WRDLEDATA_B）分だけ延長されます。そして、バースト周期のクロック間隔ごとに、次のデータがサンプリングされます。アドレスはリニアにインクリメントされ、決して 1kB 境界を超えません。

インクリメントには、下記の値が使用されます。

- モード 32：4 バイト
- モード 16：2 バイト
- モード 8：1 バイト

スレーブではバースト読み出しで、データをプリフェッチできます。（そのため、たとえば FIFO に対してバースト転送を行わないでください）。

注　意

バースト長は 1/2/4（モードに依存）～1 キロバイトで不定であり、決して 1kB 境界を超えません。

バースト転送は DMA/CPU によって中断することが可能です。

外部信号 MSEBIM_WAIT[n]_N/MSEBIS_WAIT[n]_N（n＝0～3）が実装されている場合は、ウェイト信号が出力されている限り VALID サブフェーズは一時停止します。

VALID サブフェーズ以外では、ウェイト信号は無視されます。

(3) HOLD

［書き込みコマンド時］

HOLD サブフェーズは、書き込みコマンドの終了後、アドレス、制御データ、データのホールド時間を維持するために使用されます。

マスタについては、rMSEBIM_SETUPHOLD_CS[n]_N レジスタの bMSEBIM_WRDLEHOLD ビットを参照してください。

［読み出しコマンド時］

HOLD サブフェーズは、外部バッファとのバス競合を避けるため、データバスをフローティングにするために使用されます。
これはターンアラウンド状態に似ています。

マスタについては、rMSEBIM_SETUPHOLD_CS[n]_N レジスタの bMSEBIM_RDDLEHOLD ビットを参照してください。

本サブフェーズは読み出しコマンド時に推奨されますが、書き込みコマンドでも次のサイクルを遅らせるために使用できます。本フェーズの終了時の読み出しサイクル中に、周辺デバイスはそのデータバスをフローティングに戻していなければいけません。

マスタ時のみ

本フェーズの長さは、CS[n]_N（n＝0～3）ごとに、MSEBIM_CLK の 0～63 周期に設定が可能です。

注　意

外部周辺デバイスへの読み出しアクセスには（バス競合を避けるため）長さ”0”を使用しないでください。

10.4.4　MSEBI タイミング

10.4.4.1　非同期式モード、ALE 数 1

マスタモード時のみ

非同期式モードを管理する場合、MSEBI インタフェースには以下の機能を設定する必要があります（n＝0～3）。

- 非同期式モードを許可（bMSEBIM_CONFIG ビットをセット）
- バーストモードは常に禁止され、bMSEBIM_BURST_ENABLE ビットは無視されます。
- 外部信号 MSEBIM_WAIT[n]_N がバス上の外部スレーブによって生成され、内部の MSEBIM_HCLK クロックと同期されます（MSEBIM_HCLK クロック 1～3 周期分のレイテンシの可能性があります。また、MSEBI_CS[n]_N が Low かつ MSEBIM_DLE が High のとき、かつ RDDLEDATA_NB（読み出し時）または WRDLEDATA_NB（書き込み時）の期間が終了した VALID サブフェーズ終了時に参照されます）。
- MSEBI マスタバスは、非同期式モードにおいて下記のすべての信号を制御します。
  - MSEBIM_CLK
  - MSEBIM_ALE
  - MSEBIM_CLE
  - MSEBIM_DLE
  - MSEBIM_RD_N
  - MSEBIM_WR_N

備　考

タイミングパラメータは、マスタとスレーブの間で適合している必要があります。

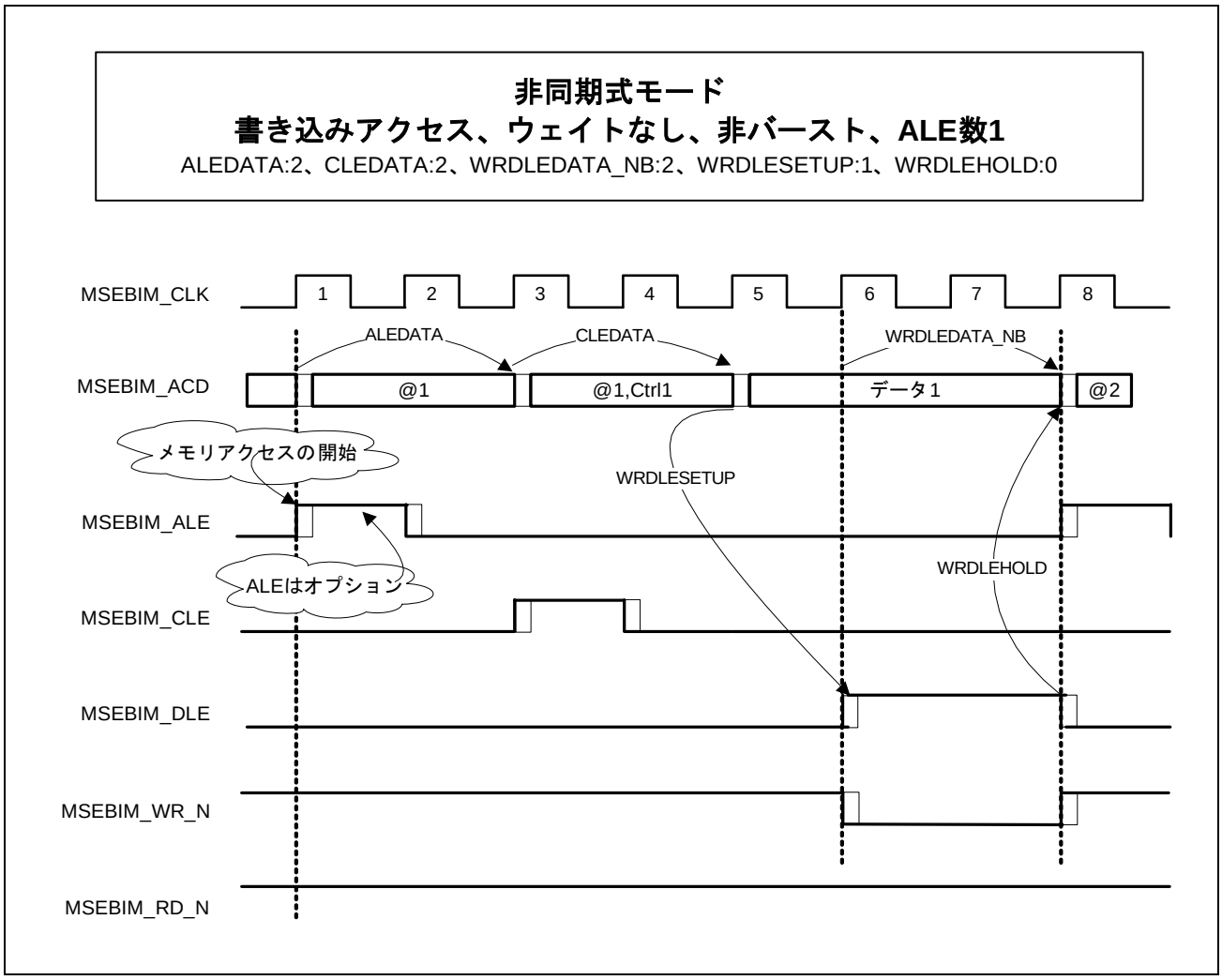


図 10.9　　MSEBI タイミング、非同期式モード、書き込み 1、ウェイトなし、非バースト、ALE 数 1

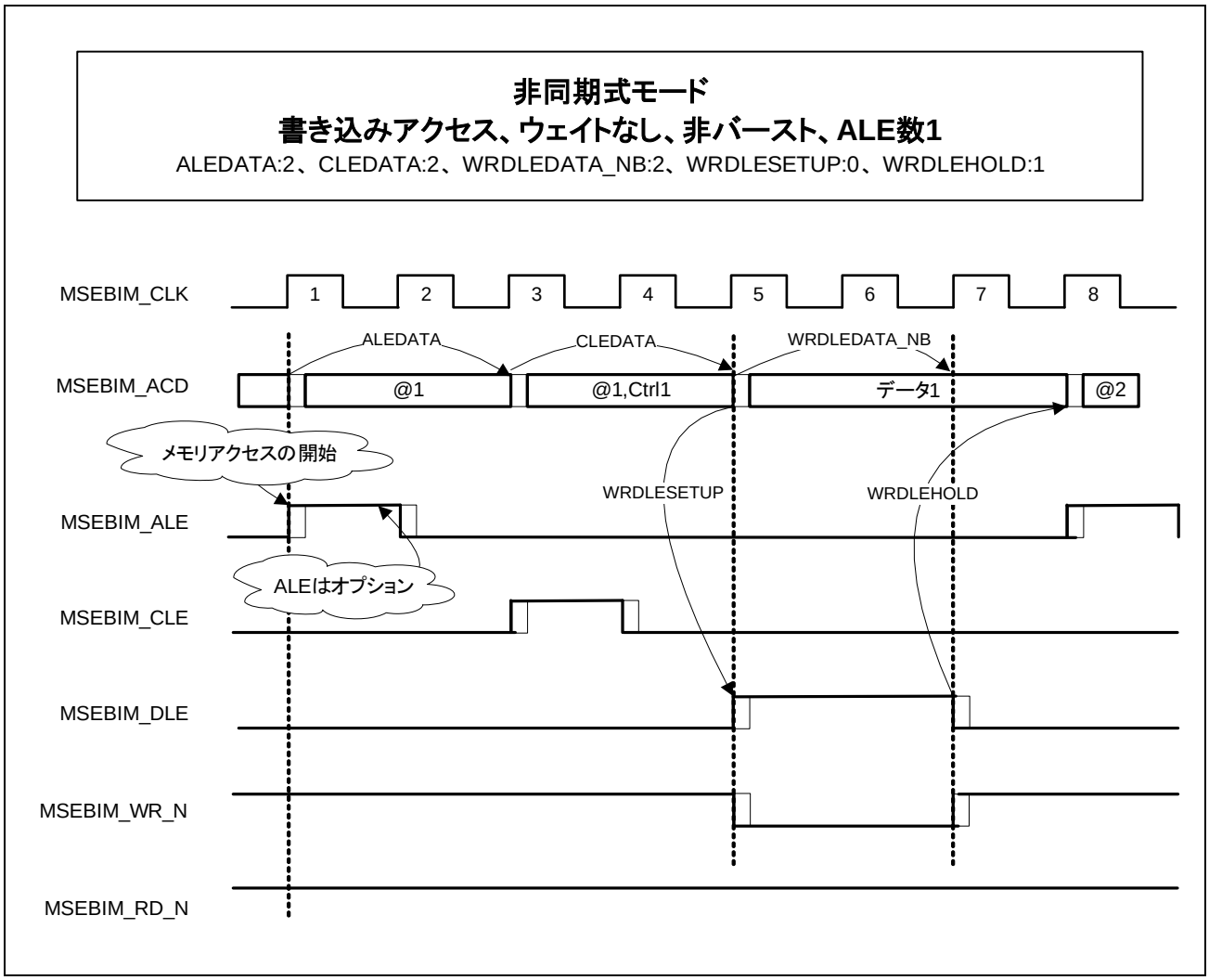


図 10.10　　MSEBI タイミング、非同期式モード、書き込み 2、ウェイトなし、非バースト、ALE 数 1

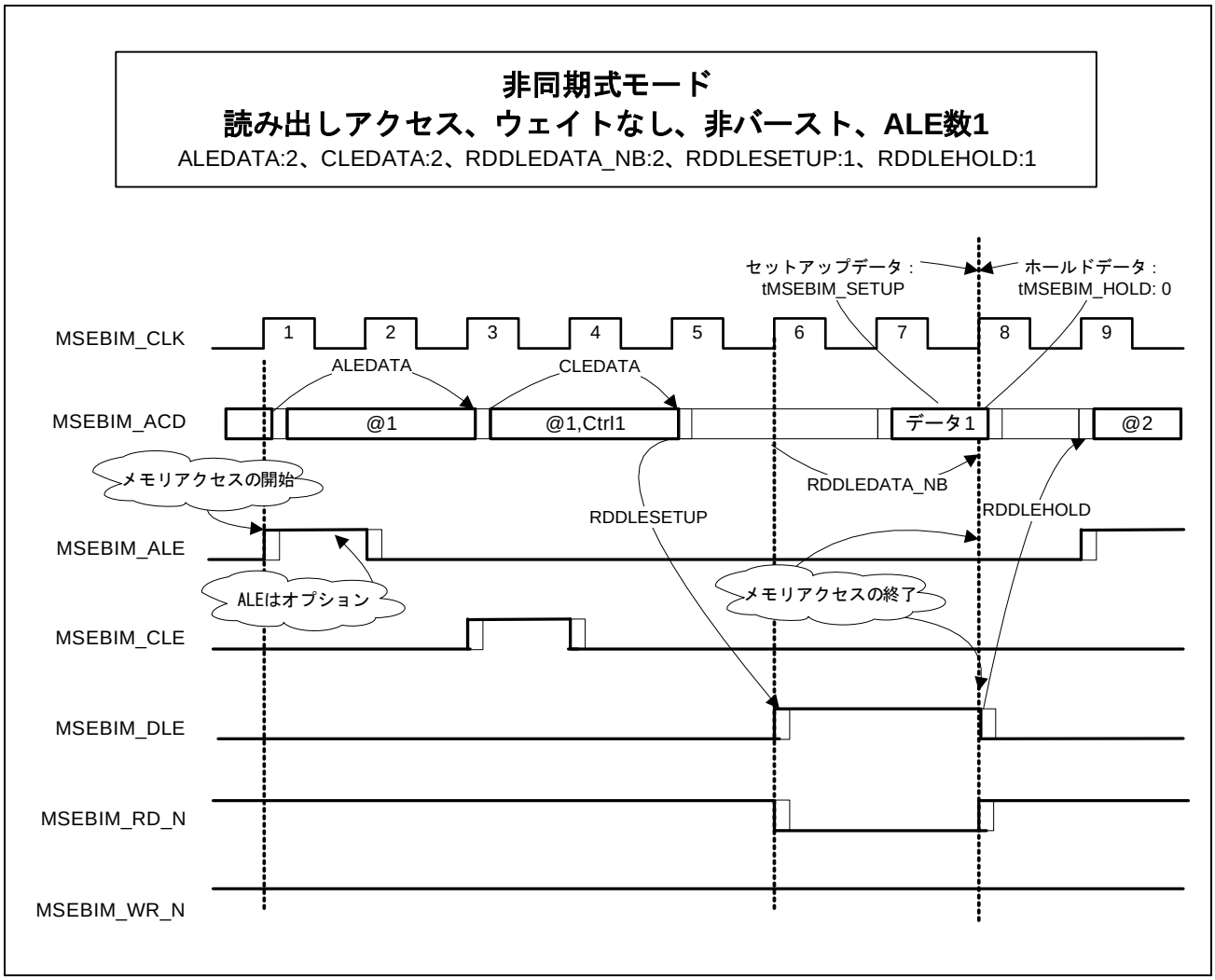


図 10.11　　MSEBI タイミング、非同期式モード、読み出し 1、ウェイトなし、非バースト、ALE 数 1

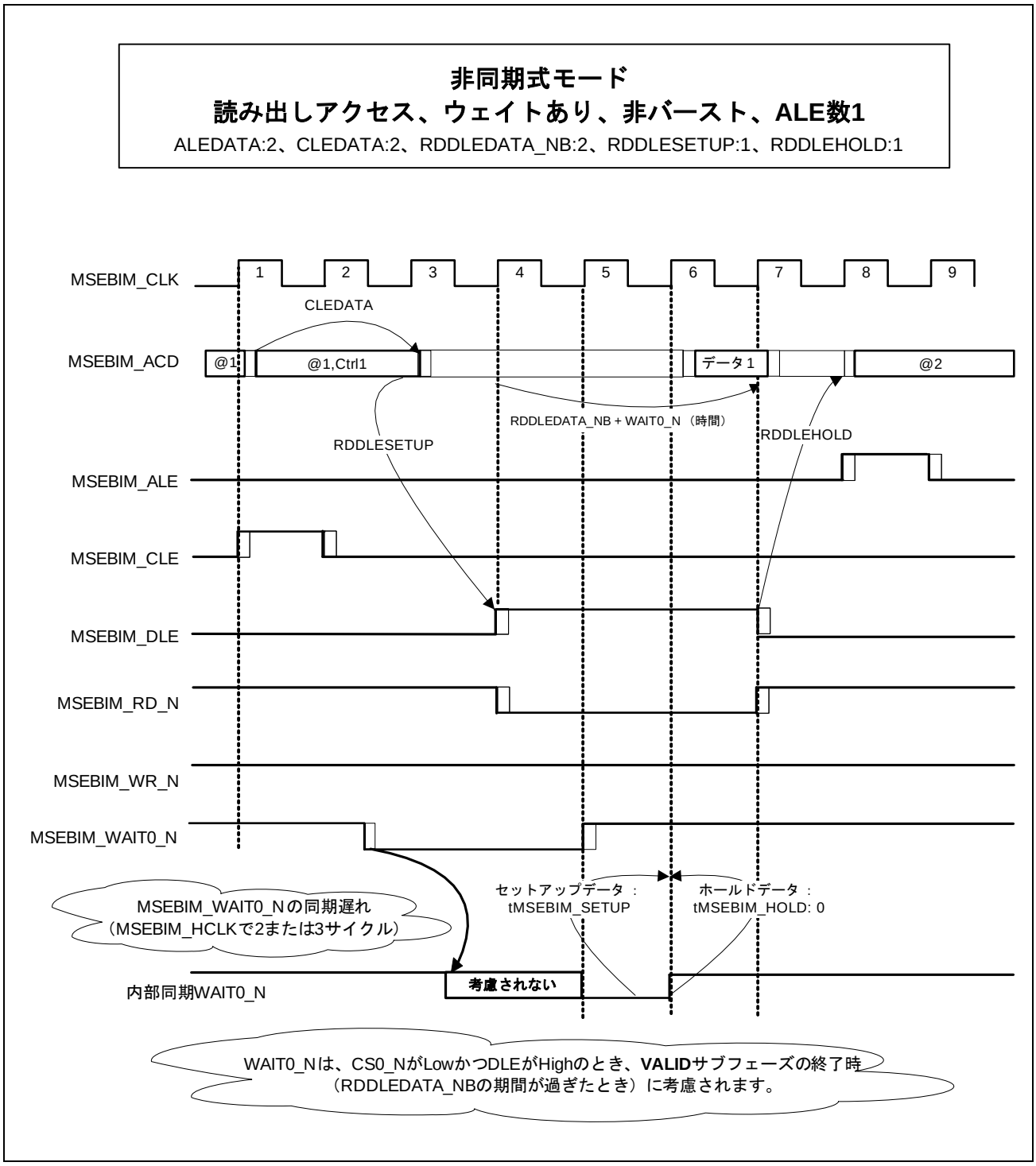


図 10.12　MSEBI タイミング、非同期式モード、読み出し 2、ウェイトあり、非バースト、ALE 数 1

10.4.4.2　非同期式モード、ALE なし、MSEBI マスタのみ

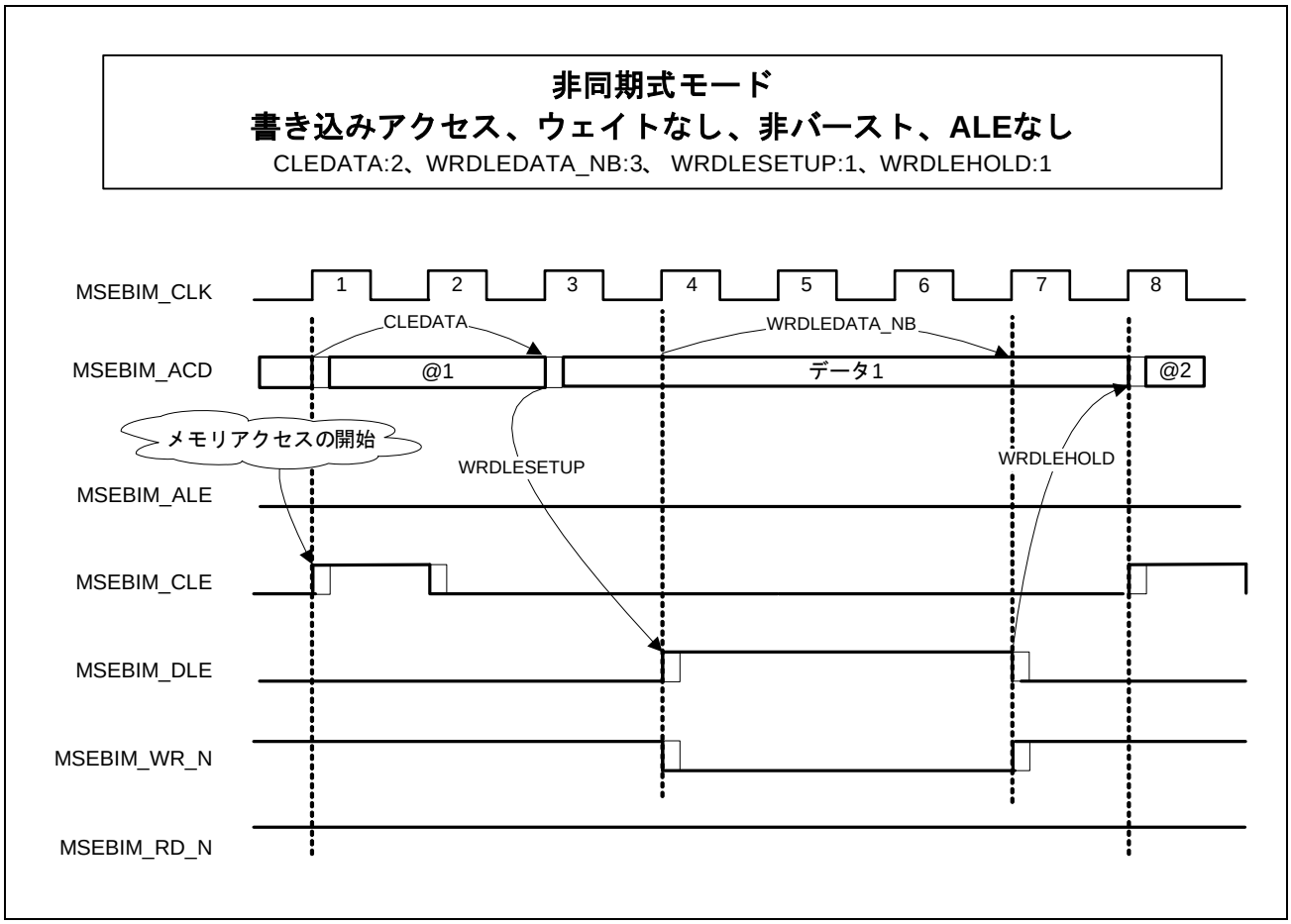


図 10.13　MSEBI タイミング、非同期式モード、書き込み 1、ウェイトなし、非バースト、ALE なし

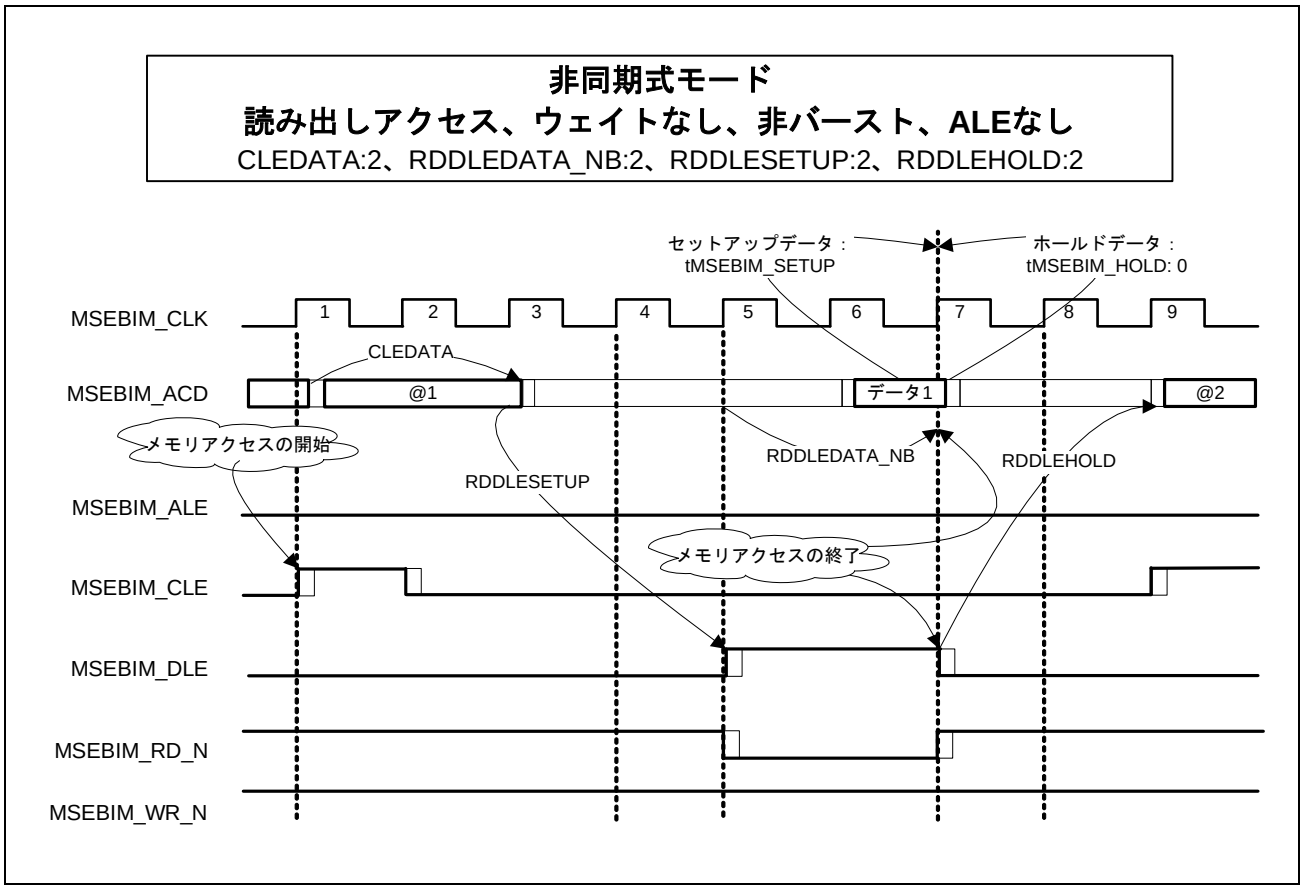


図 10.14　　MSEBI タイミング、非同期式モード、読み出し 1、ウェイトなし、非バースト、ALE なし

10.4.4.3　非同期式モード、ALE 数 2

マスタモード時のみ

非同期式モードを管理する場合、MSEBI インタフェースには以下の機能を設定する必要があります（n＝0～3）。

- 非同期式モードを許可（bMSEBIM_CONFIG ビットをセット）
- バーストモードは常に禁止され、bMSEBIM_BURST_ENABLE ビットは無視されます。
- 外部信号 MSEBIM_WAIT[n]_N がバス上の外部スレーブによって生成され、内部の MSEBIM_HCLK クロックと同期されます（MSEBIM_HCLK クロック 1～3 周期分のレイテンシの可能性があります。また、MSEBI_CS[n]_N が Low かつ MSEBIM_DLE が High のとき、かつ RDDLEDATA_NB（読み出し時）または WRDLEDATA_NB（書き込み時）の期間が終了した VALID サブフェーズ終了時に参照されます）。
- MSEBI マスタバスは、非同期式モードにおいて下記のすべての信号を制御します。
  - MSEBIM_CLK
  - MSEBIM_ALE（オプションで MSEBIM_ALE1）、下記の 2 モードが利用可能です。
    シリアルモード：ALE は同じライン（MSEBIM_ALE）で生成されます。
    パラレルモード：各 ALE は異なるライン（MSEBIM_ALE、MSEBIM_ALE1）にルーティングされます。
  - MSEBIM_CLE
  - MSEBIM_DLE
  - MSEBIM_RD_N
  - MSEBIM_WR_N

備　考

タイミングパラメータは、マスタとスレーブの間で適合している必要があります。

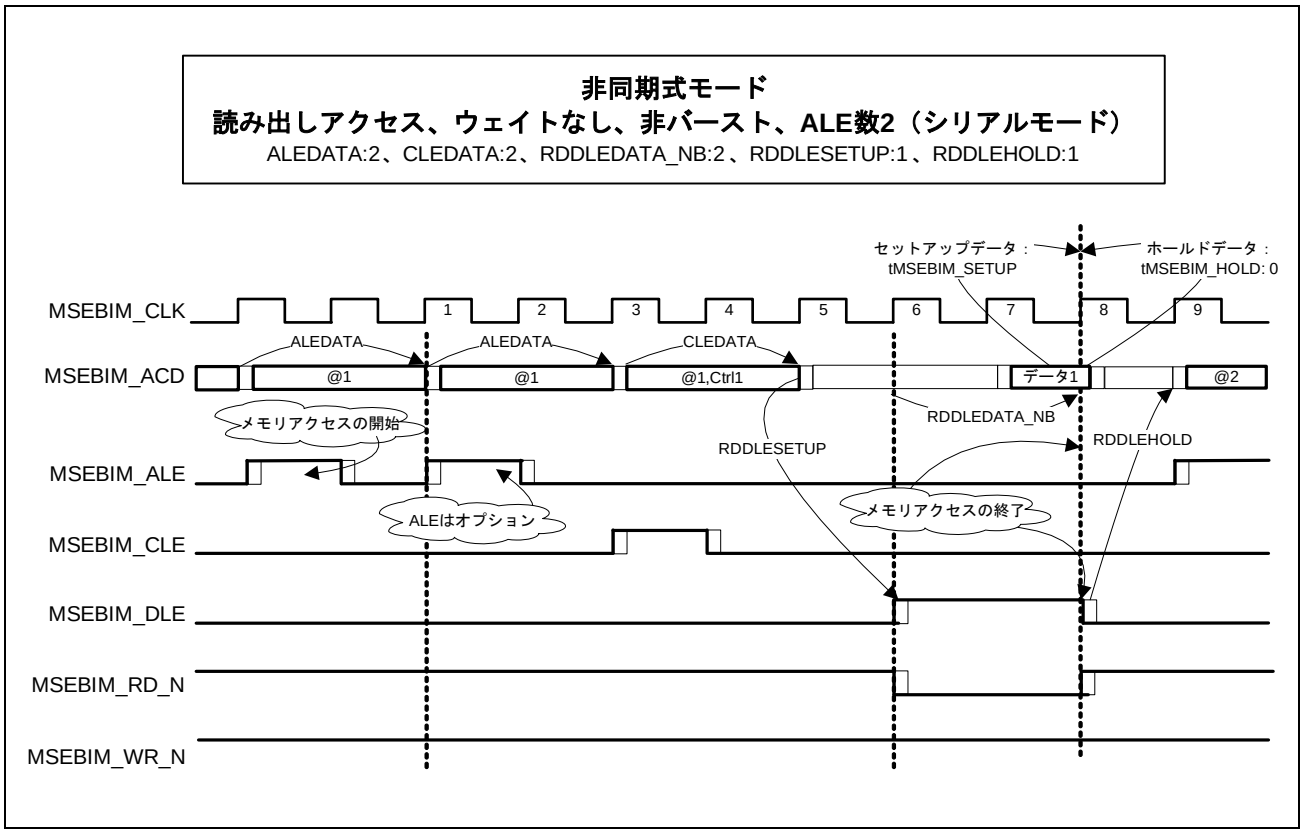


図 10.15　MSEBI タイミング、非同期式モード、読み出し、ウェイトなし、非バースト、ALE 数 2（シリアルモード）

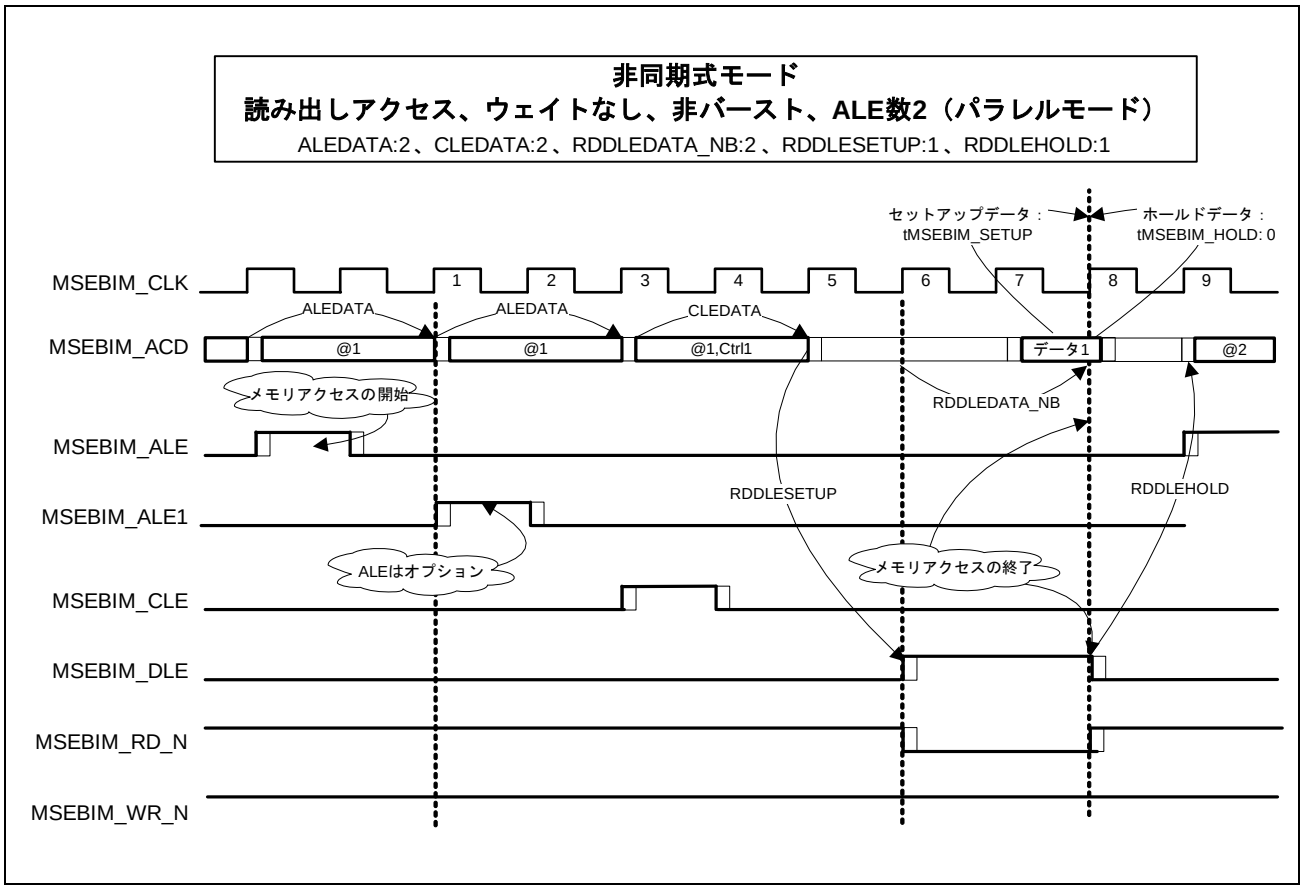


図 10.16　MSEBI タイミング、非同期式モード、読み出し、ウェイトなし、非バースト、ALE 数 2（パラレルモード）

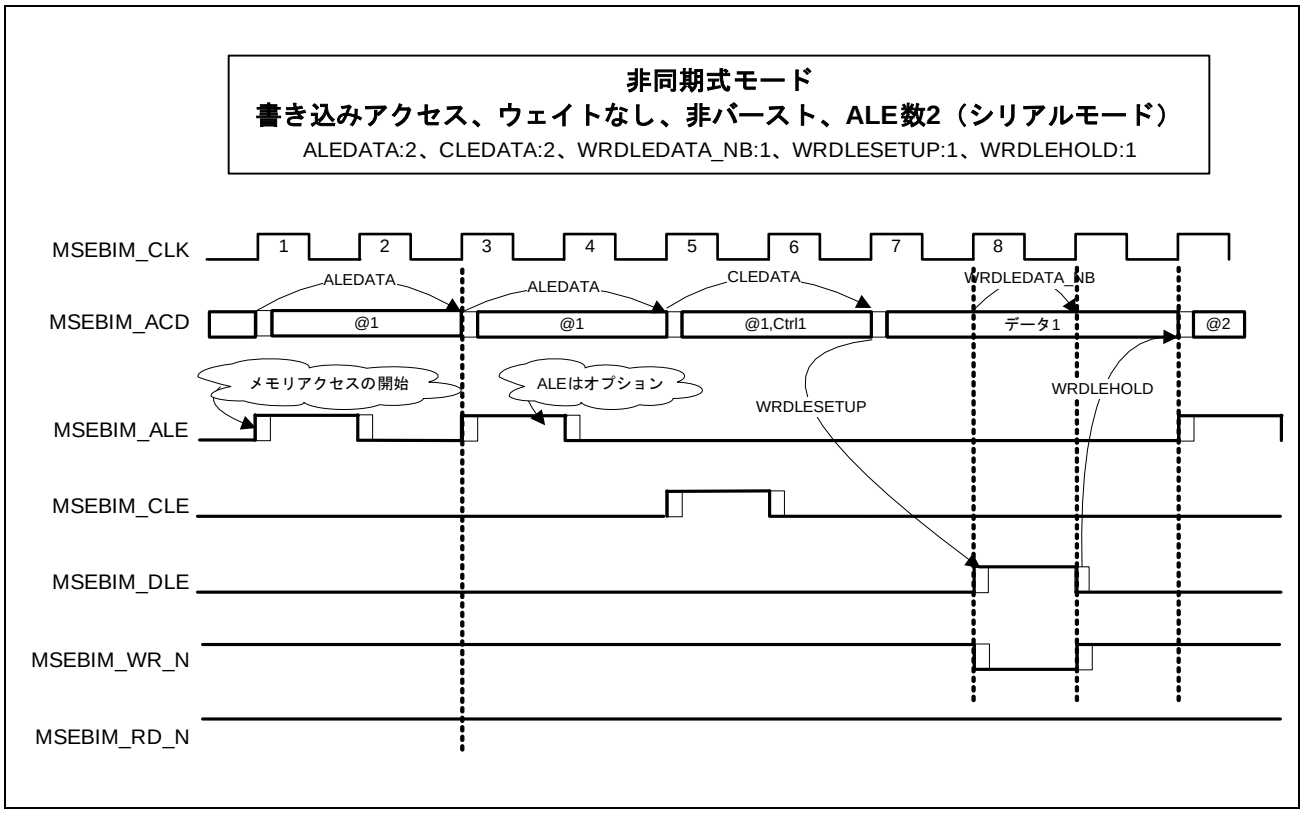


図 10.17　MSEBI タイミング、非同期式モード、書き込み、ウェイトなし、非バースト、ALE 数 2（シリアルモード）

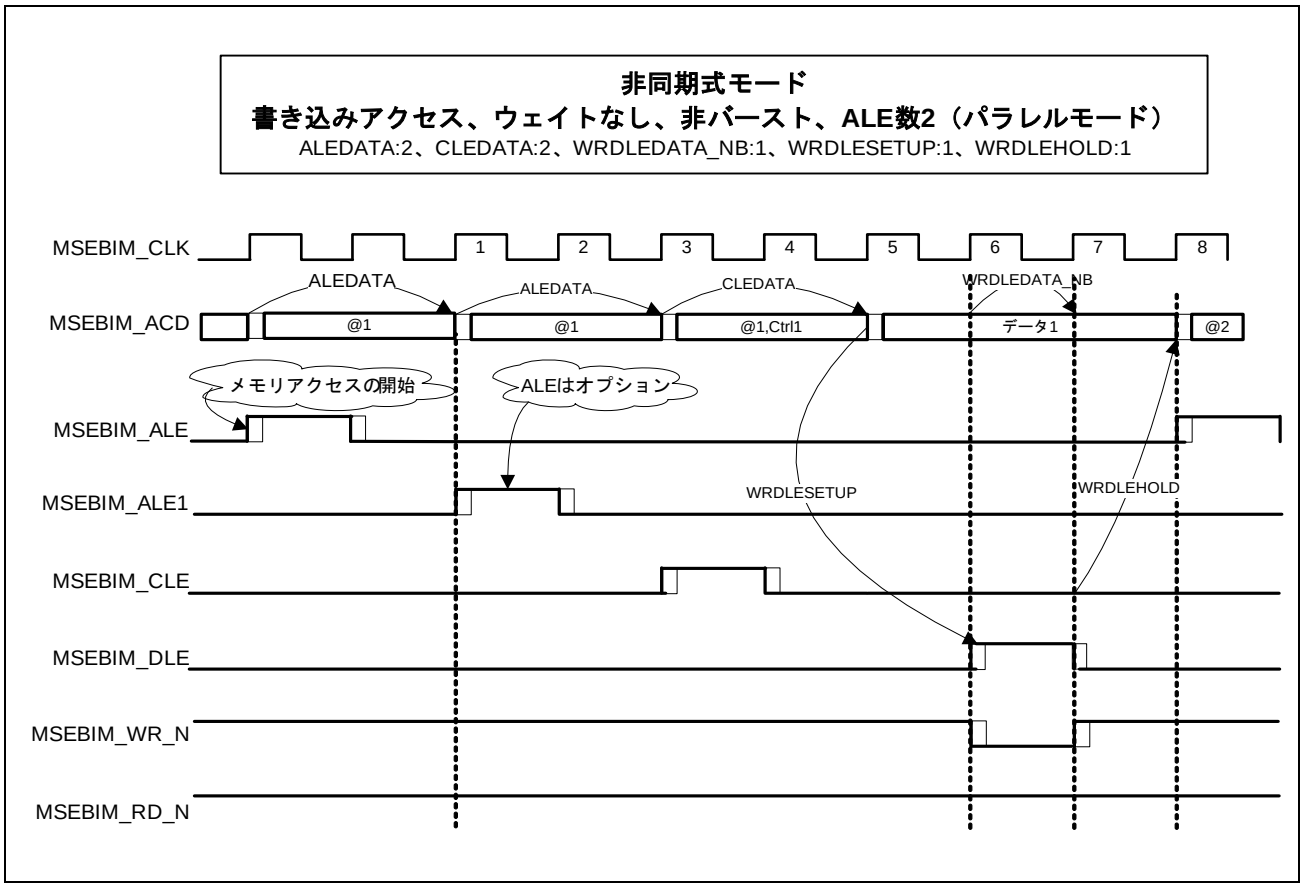


図 10.18　MSEBI タイミング、非同期式モード、書き込み、ウェイトなし、非バースト、ALE 数 2（パラレルモード）

10.4.4.4　同期式モード、非バースト、ALE 数 1

同期式モードを管理する場合、MSEBI インタフェースには以下の機能を設定する必要があります（n＝0～3）。

- 同期式モードを許可（マスタ時 bMSEBIM_CONFIG ビット、またはスレーブ時 bMSEBIS_CONFIG ビットを設定）
- バーストモードは、マスタ側で bMSEBIM_BURST_ENABLE ビットを用いて禁止できます。
- 外部信号 MSEBIM_WAIT[n]_N がバス上の外部スレーブによって生成され、MSEBIM_CLK クロックと同期されます。本信号は、MSEBI_CS[n]_N が Low かつ MSEBIM_DLE が High のとき、かつ RDDLEDATA_NB（読み出し時）または WRDLEDATA_NB（書き込み時）の期間が終了した場合の VALID サブフェーズ中にマスタによって参照されます。
- MSEBI マスタバスは、同期式モードにおいて下記のすべての信号を制御します。
  - MSEBIM_CLK
  - MSEBIM_ALE
  - MSEBIM_CLE
  - MSEBIM_DLE
- また、下記の信号を 1 にセットします。
  - MSEBIM_RD_N
  - MSEBIM_WR_N
- バス上の MSEBI スレーブデバイスは、下記の信号を使用します。
  - MSEBIS_CLK
  - MSEBIS_ALE
  - MSEBIS_CLE
  - MSEBIS_DLE
- また、下記の信号を生成します。
  - MSEBIM_WAIT[n]_N

備　考

タイミングパラメータは、マスタとスレーブの間で適合している必要があります。

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

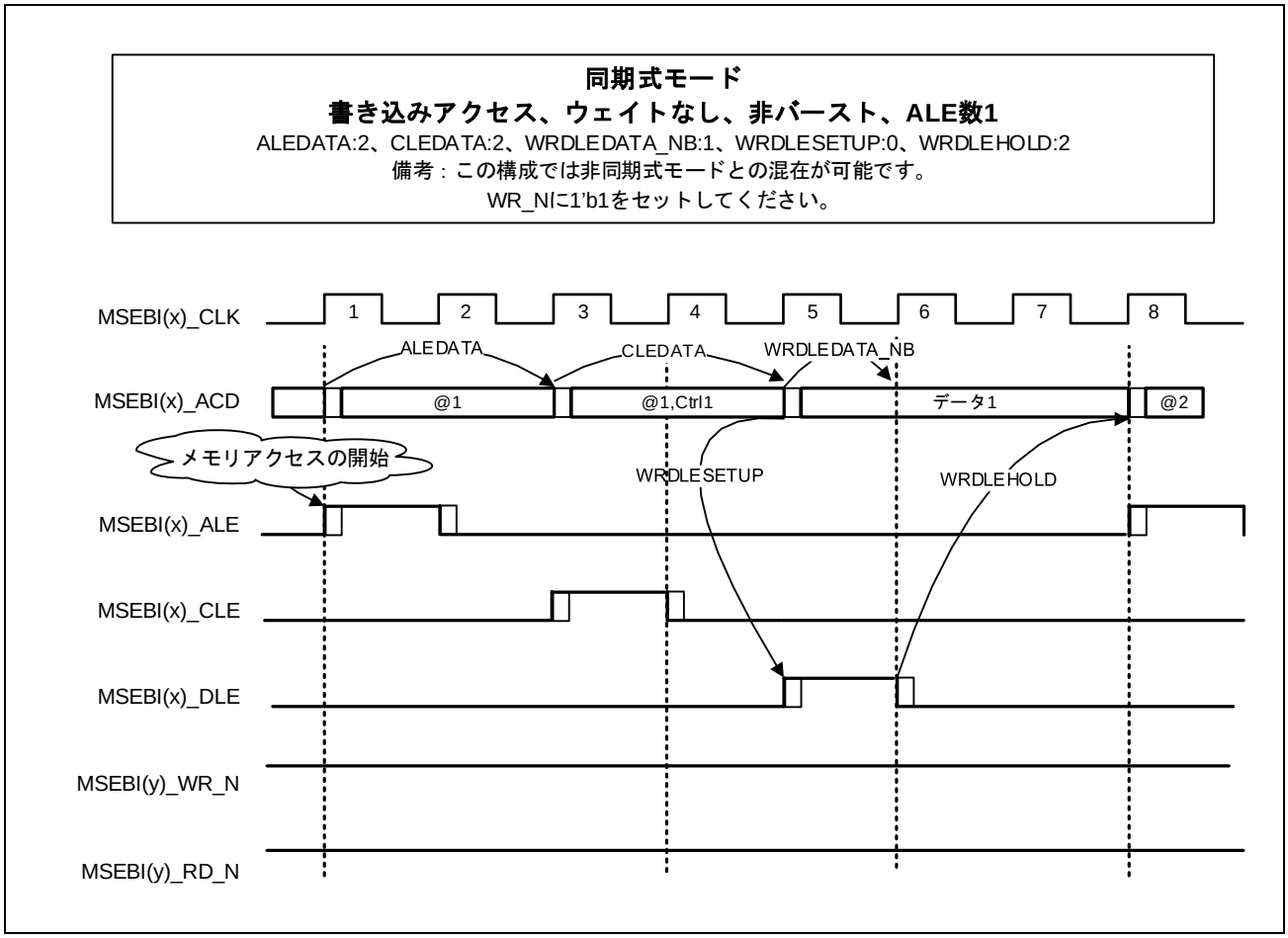


図 10.19　　MSEBI タイミング、同期式モード、書き込み 1、ウェイトなし、非バースト、ALE 数 1

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

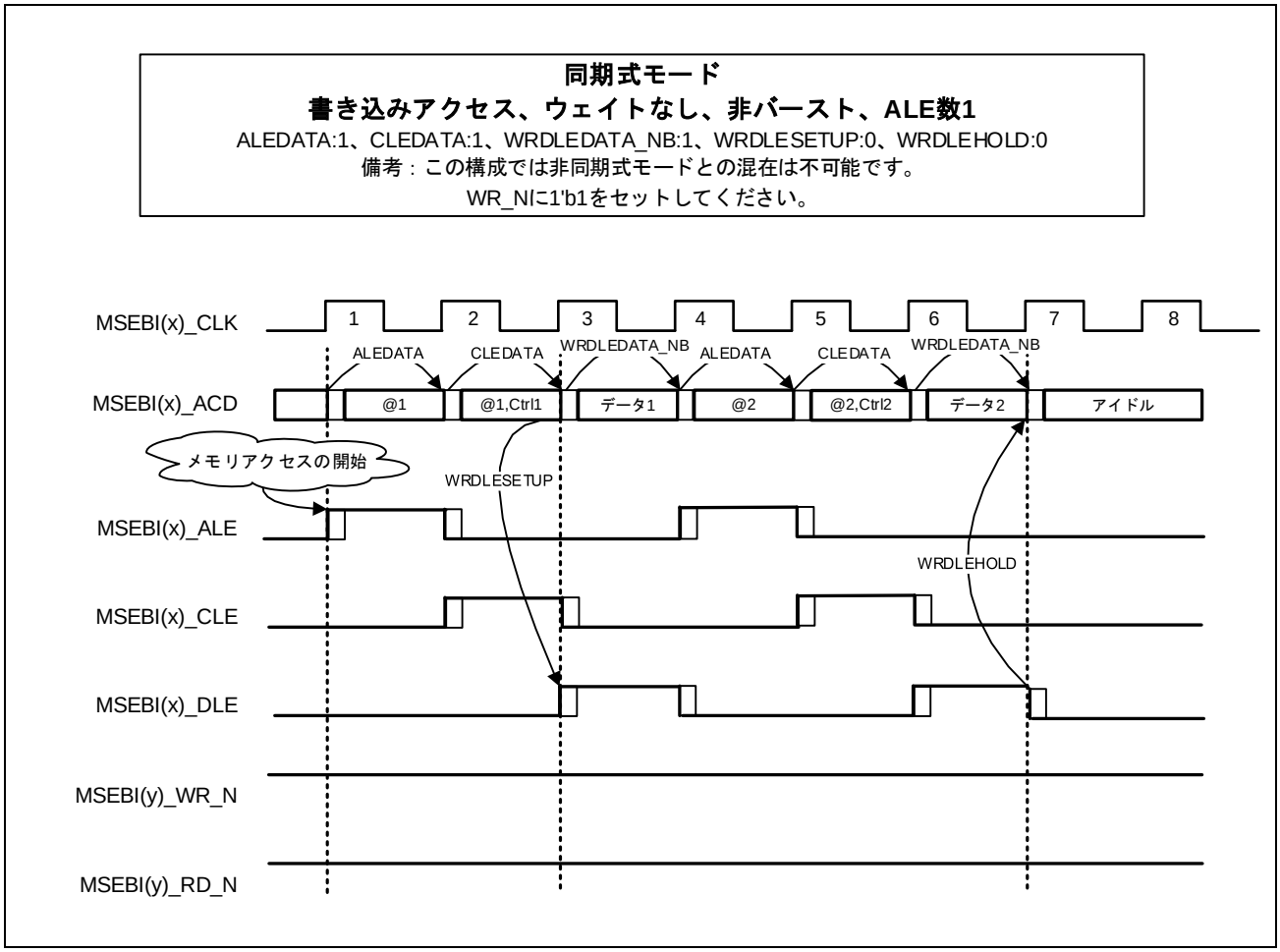


図 10.20　　MSEBI タイミング、同期式モード、書き込み 2、ウェイトなし、非バースト、ALE 数 1

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

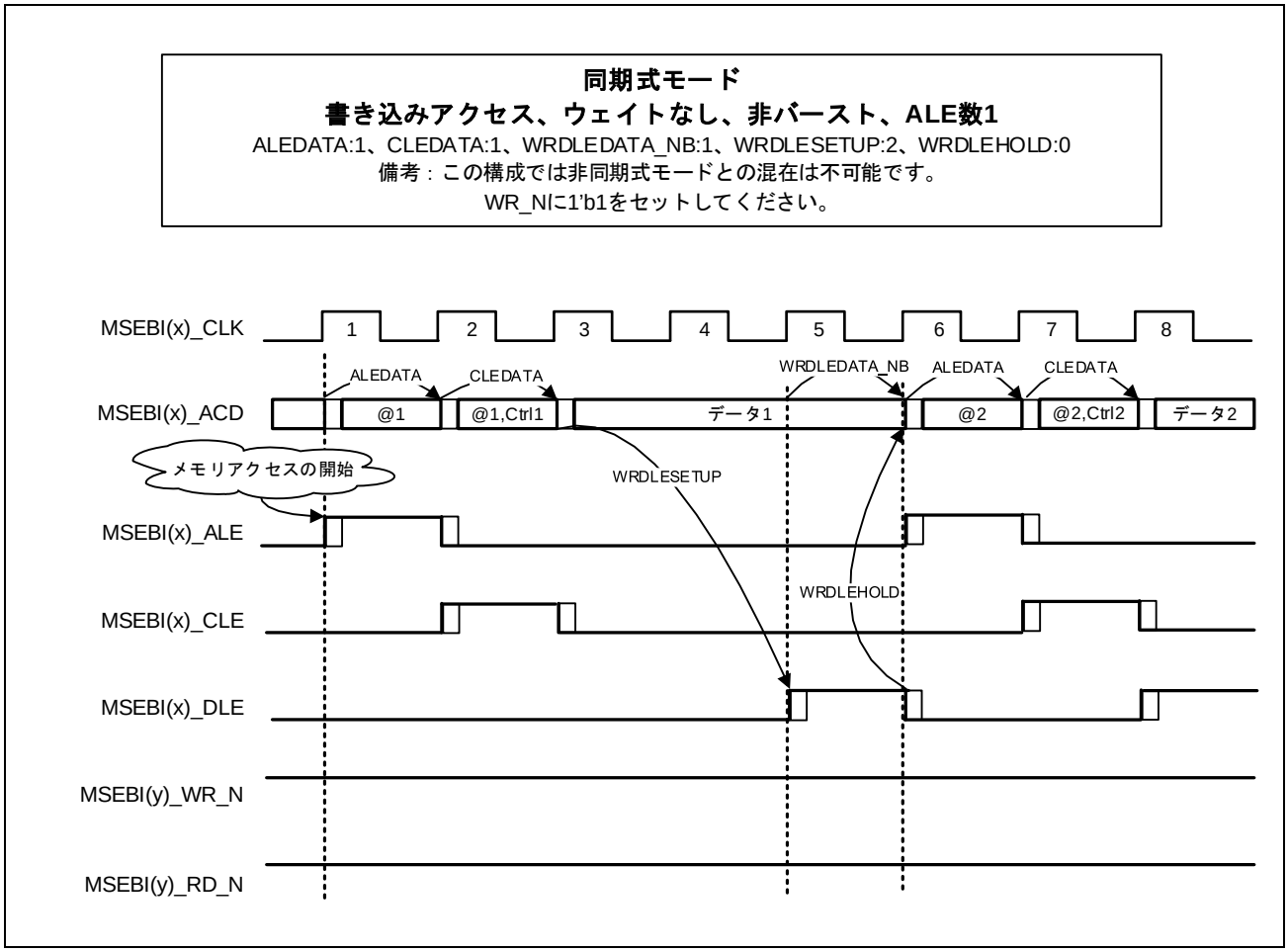


図 10.21　　MSEBI タイミング、同期式モード、書き込み 3、ウェイトなし、非バースト、ALE 数 1

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

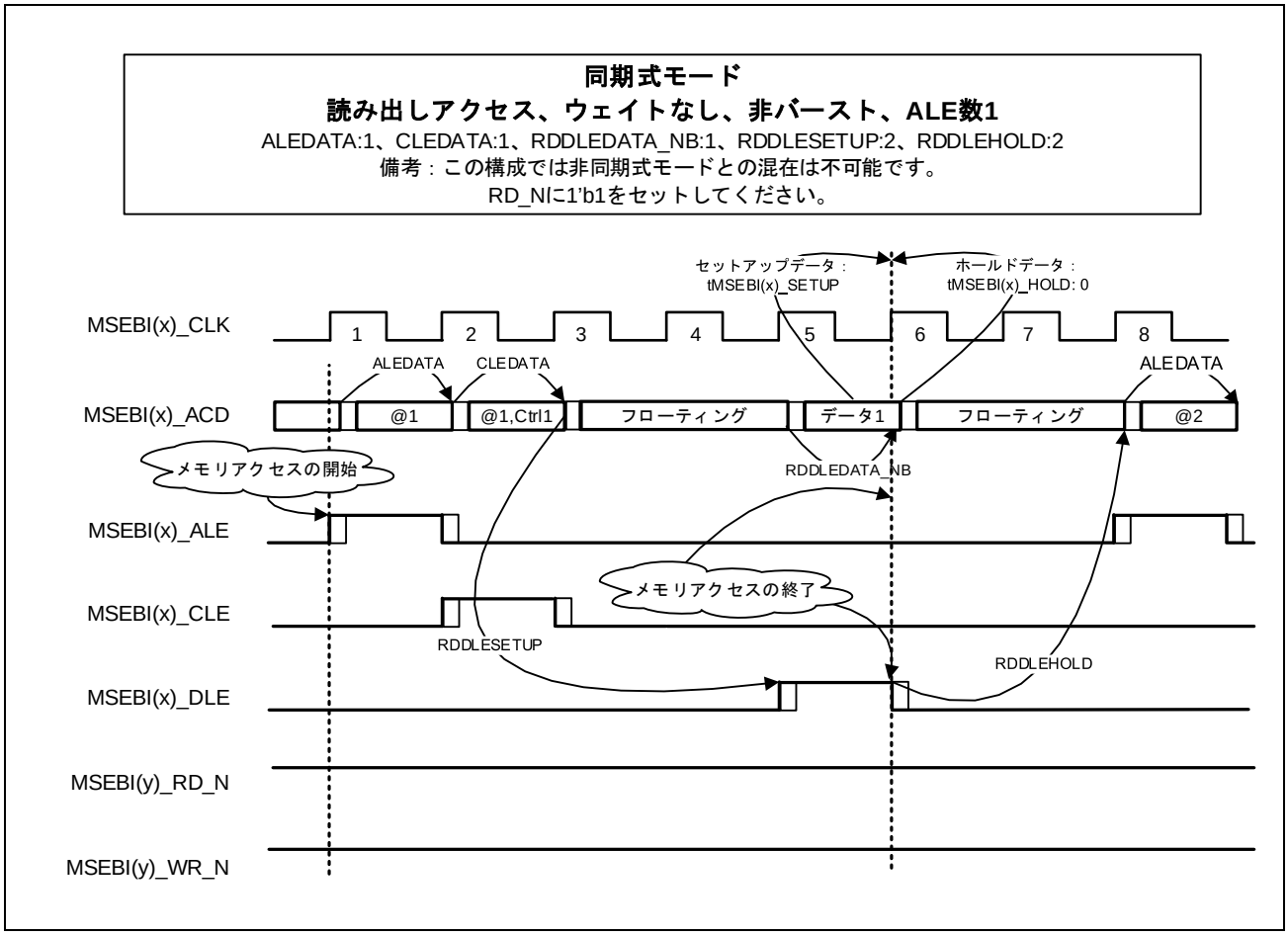


図 10.22　　MSEBI タイミング、同期式モード、読み出し 1、ウェイトなし、非バースト、ALE 数 1

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

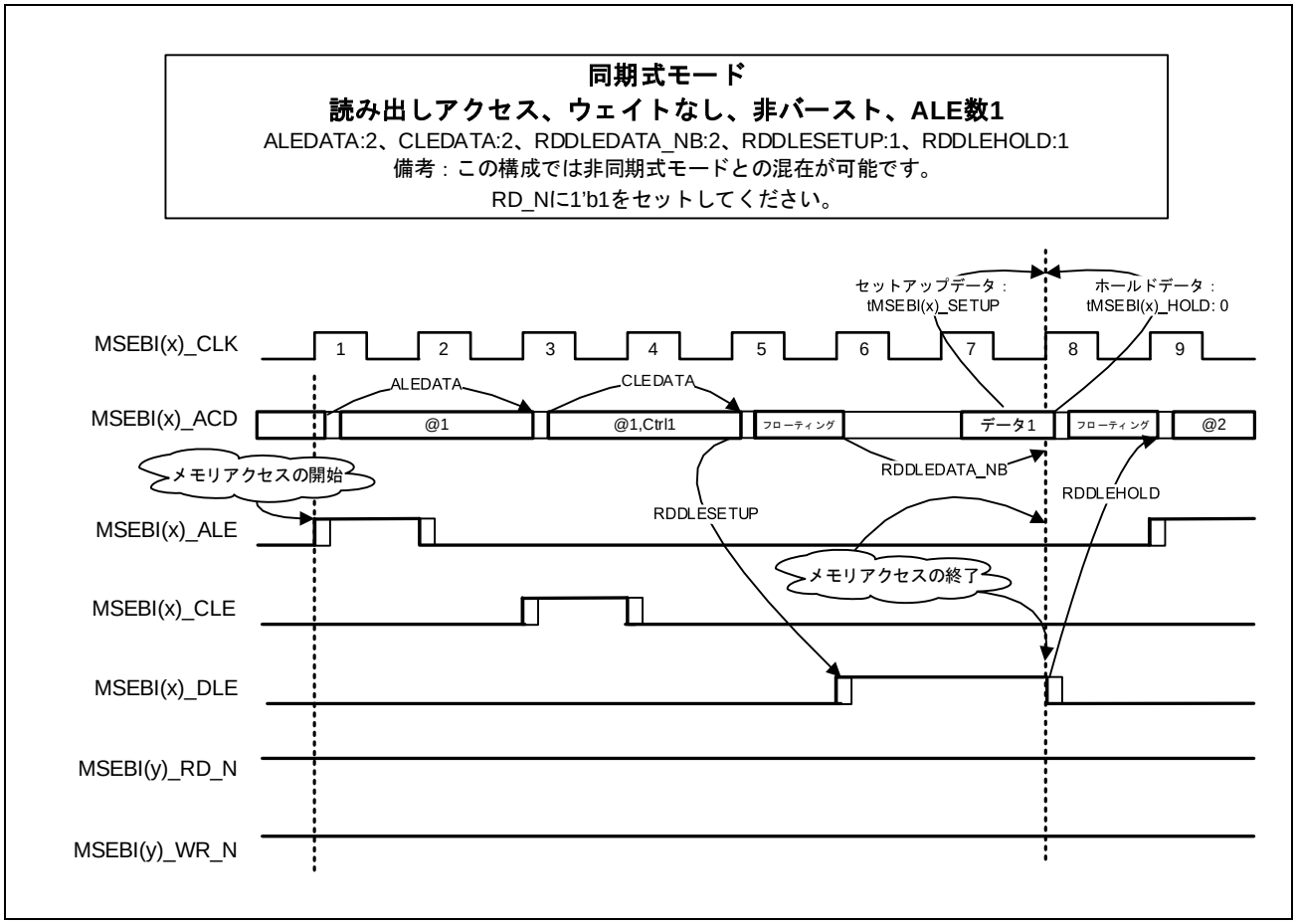


図 10.23　　MSEBI タイミング、同期式モード、読み出し 2、ウェイトなし、非バースト、ALE 数 1

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

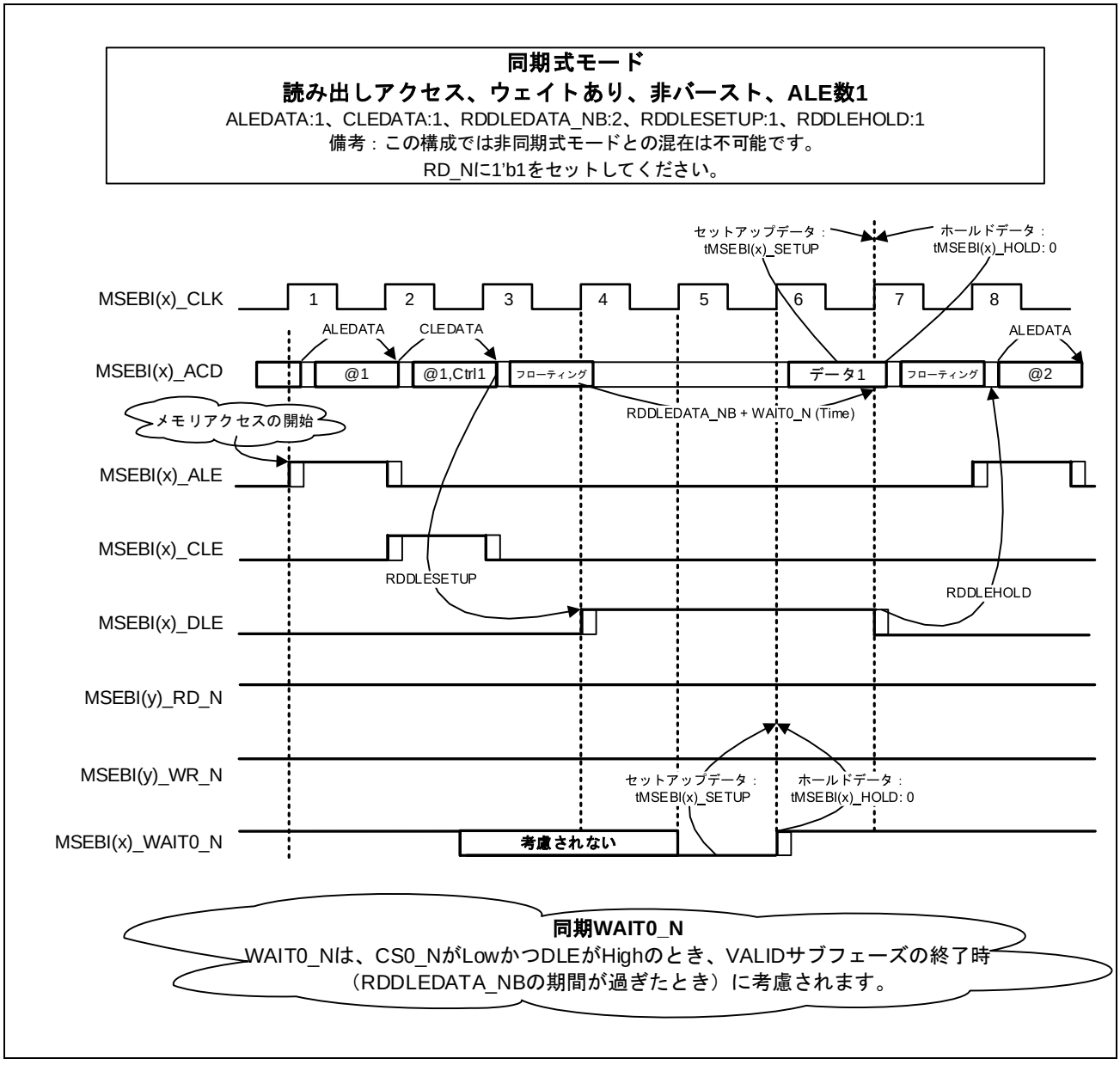


図 10.24　　MSEBI タイミング、同期式モード、読み出し 3、ウェイトあり、非バースト、ALE 数 1

10.4.4.5　同期式モード、非バースト、ALE なし

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

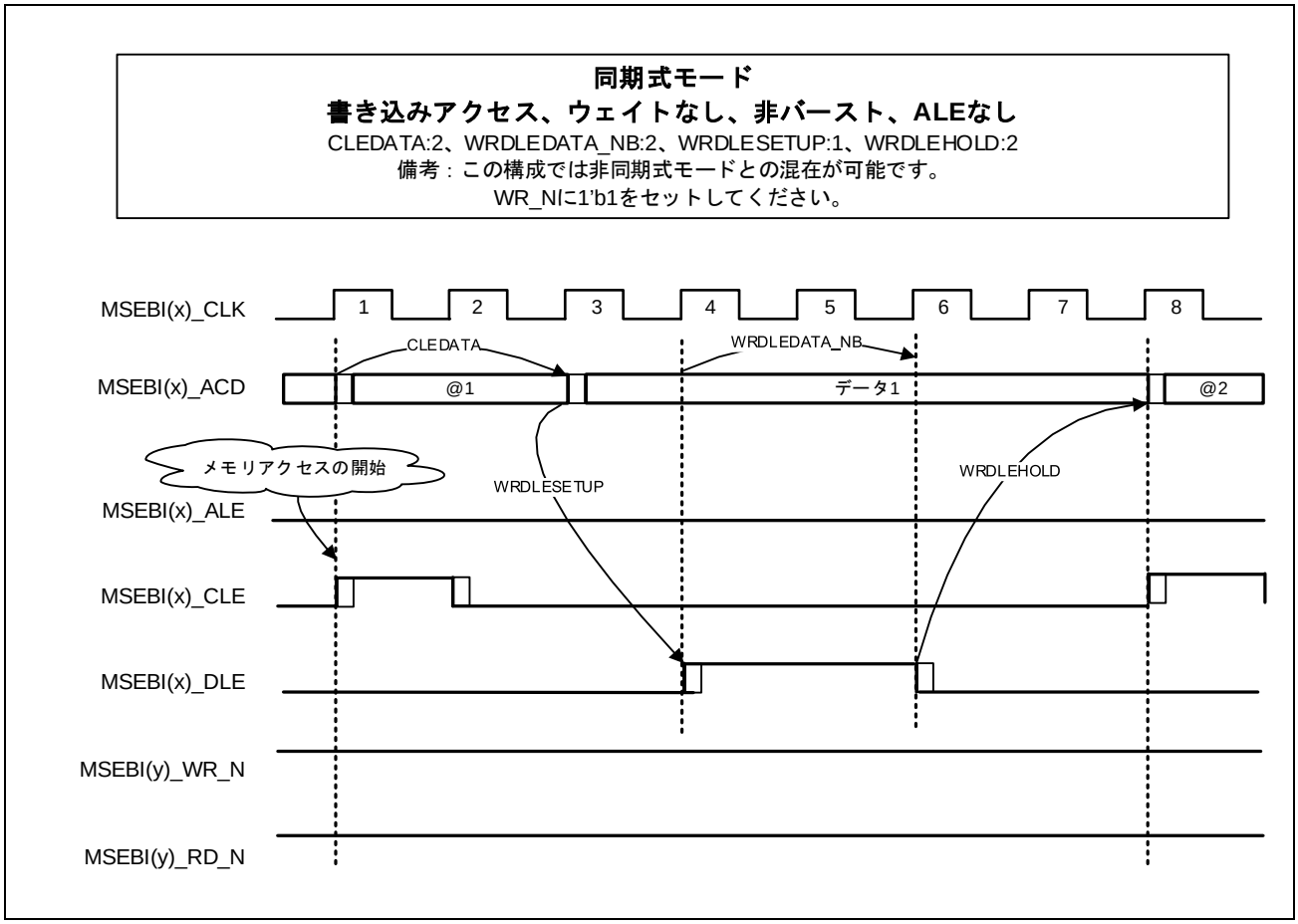


図 10.25　MSEBI タイミング、同期式モード、書き込み 1、ウェイトなし、非バースト、ALE なし

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

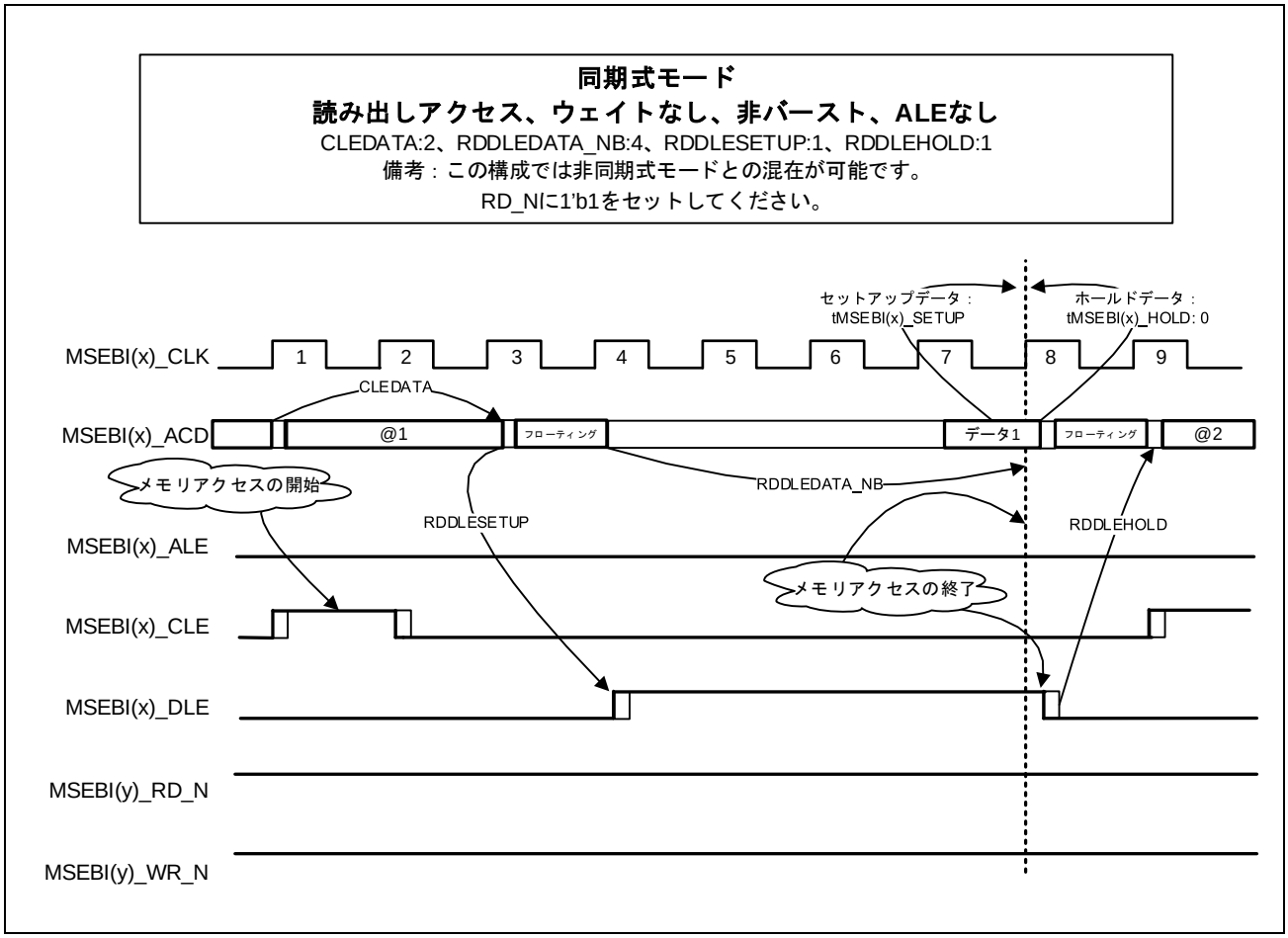


図 10.26　　MSEBI タイミング、同期式モード、読み出し 1、ウェイトなし、非バースト、ALE なし

10.4.4.6　同期式モード、非バースト、複数 ALE

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

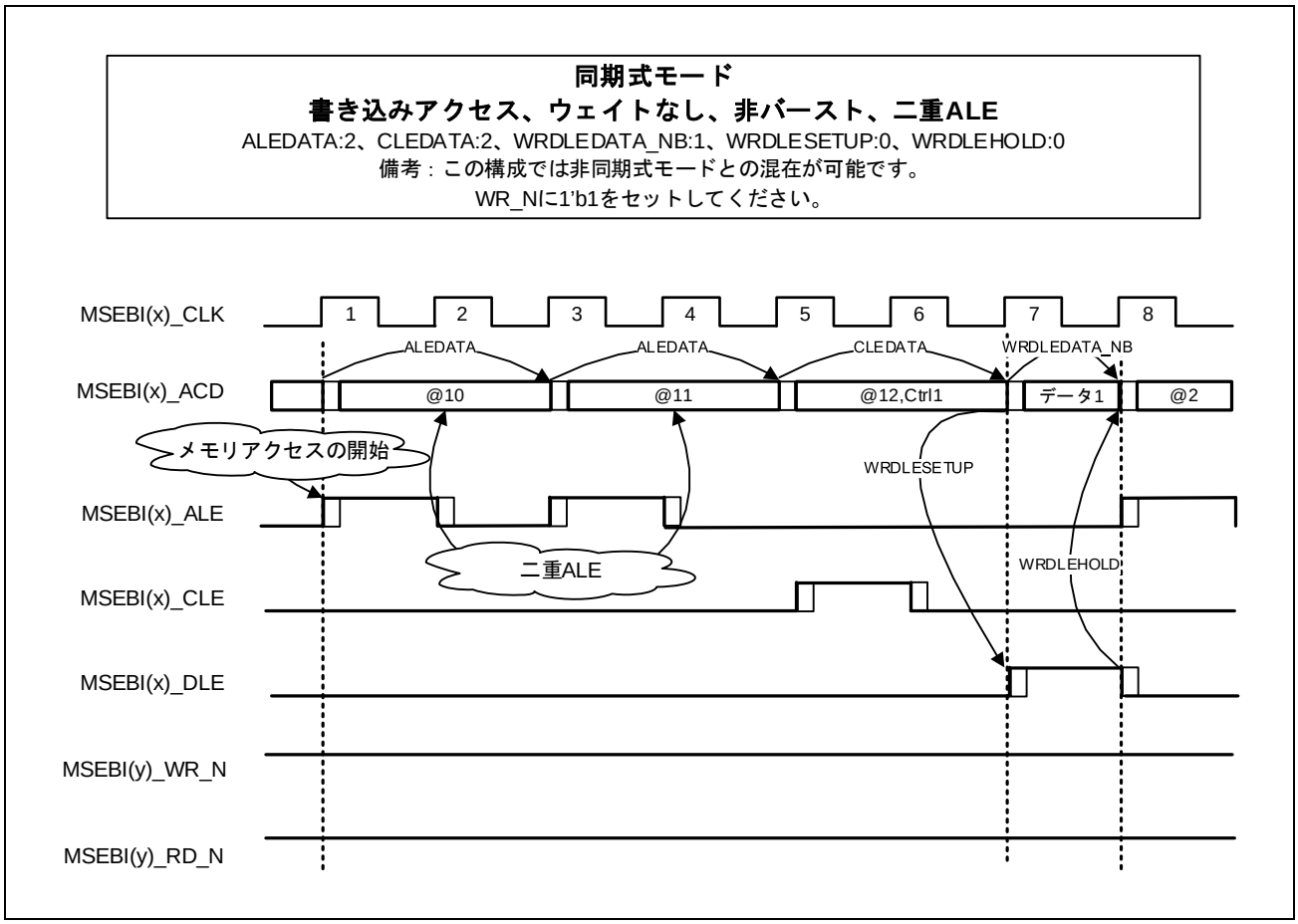


図 10.27　MSEBI タイミング、同期式モード、書き込み 1、ウェイトなし、非バースト、二重 ALE

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

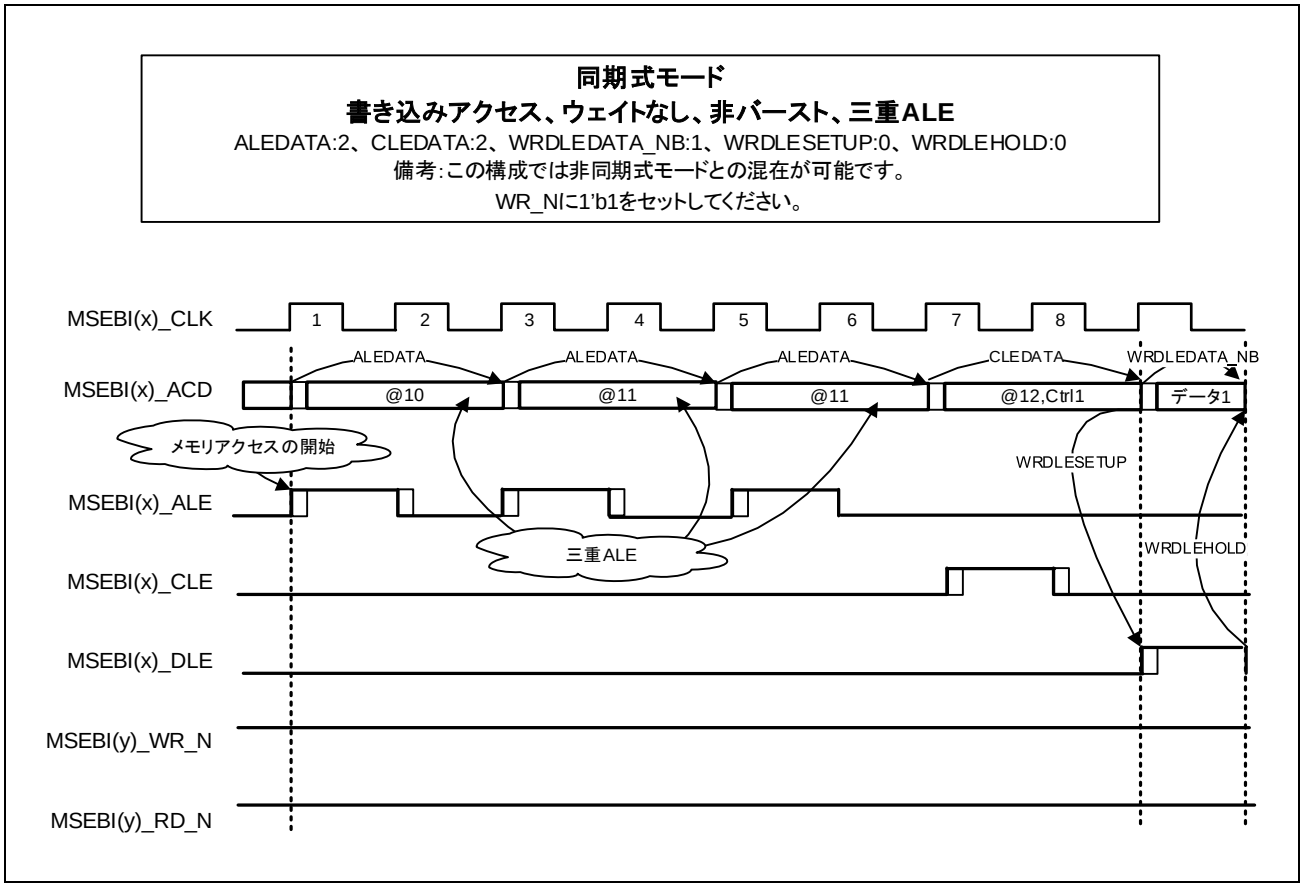


図 10.28　　MSEBI タイミング、同期式モード、書き込み 2、ウェイトなし、非バースト、三重 ALE

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

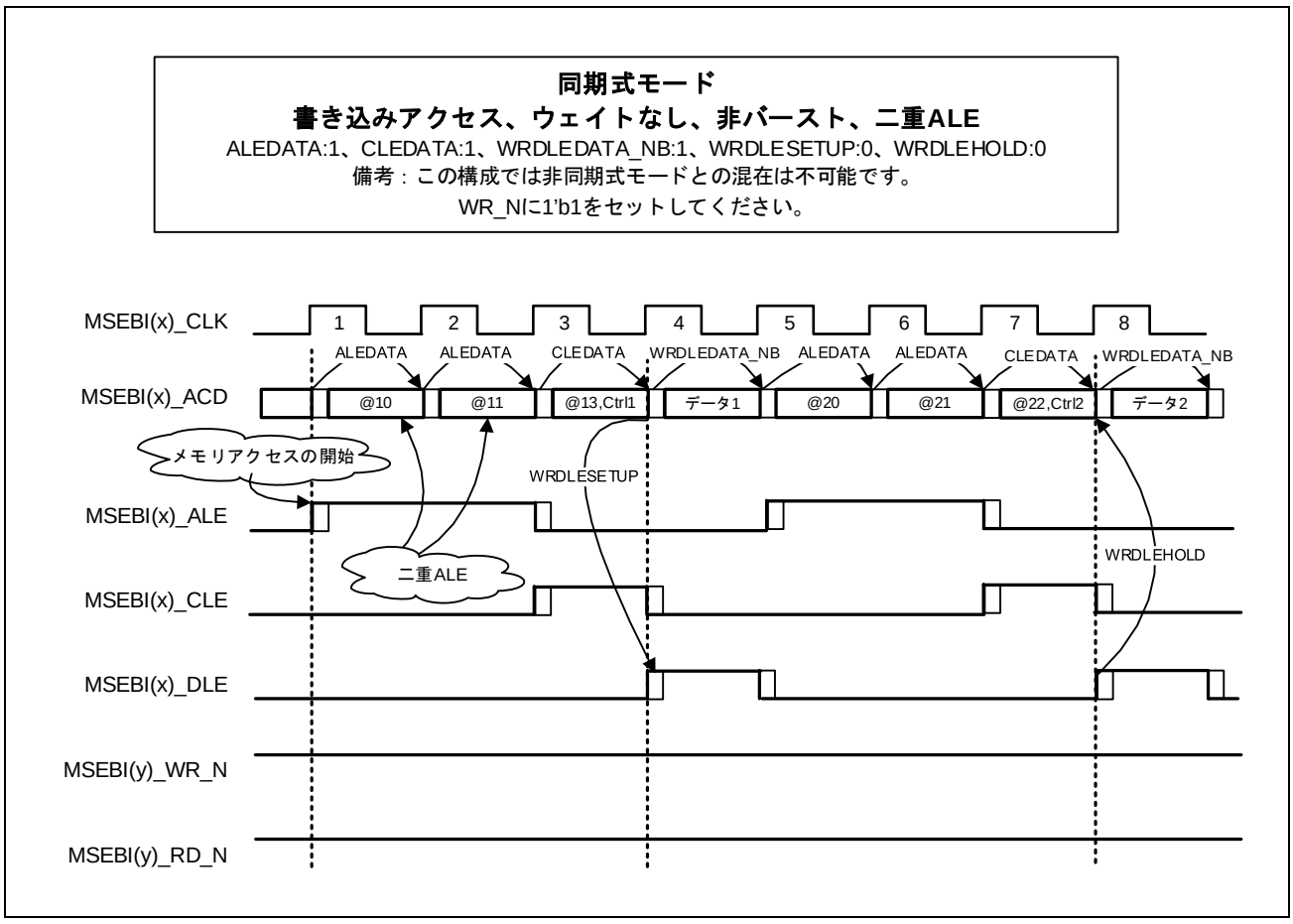


図 10.29　MSEBI タイミング、同期式モード、書き込み 3、ウェイトなし、非バースト、二重 ALE

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

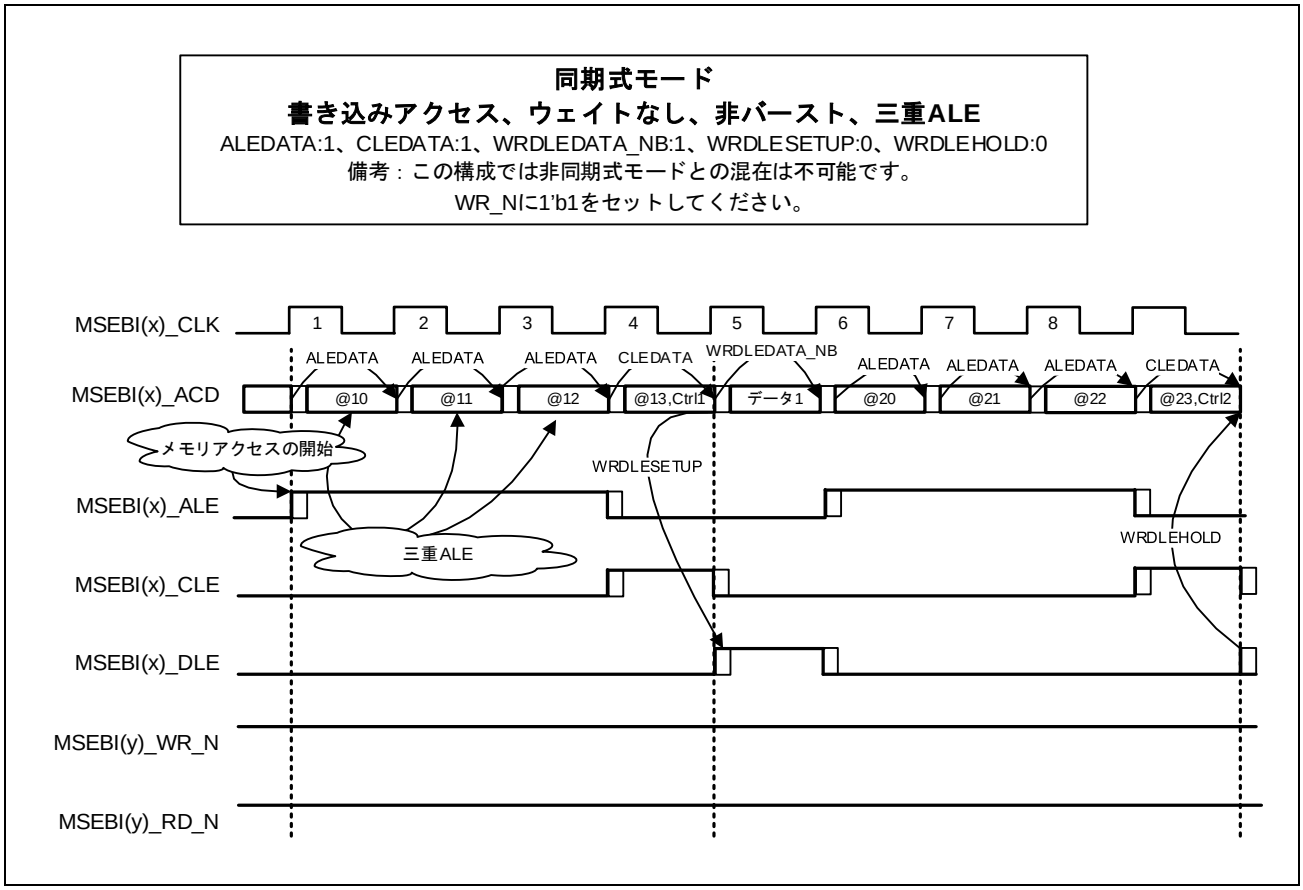


図 10.30　　MSEBI タイミング、同期式モード、書き込み 4、ウェイトなし、非バースト、三重 ALE

10.4.4.7　同期式モード、バースト、ALE 数 1

同期式モードを管理する場合、MSEBI インタフェースには以下の機能を設定する必要があります（n＝0～3）。

- 同期式モードを許可（マスタ時 bMSEBIM_CONFIG ビット、またはスレーブ時 bMSEBIS_CONFIG ビットをセット）
- バーストモードは、マスタ側で bMSEBIM_BURST_ENABLE ビットを用いて許可されます。
- 外部信号 MSEBIM_WAIT[n]_N がバス上の外部スレーブによって生成され、MSEBIM_CLK クロックと同期されます。本信号は、バーストアクセスの最初のアクセスに対して RDDLEDATA_NB（読み出し時）または WRDLEDATA_NB（書き込み時）の期間が過ぎた場合、あるいは最初のアクセス後のすべてのバーストアクセスに対して RDDLEDATA_B（読み出し時）または WRDLEDATA_B（書き込み時）の期間が過ぎた場合の VALID サブフェーズ終了時に MSEBI_CS[n]_N が Low かつ MSEBIM_DLE が High のとき、マスタによって参照されます。
- MSEBI マスタバスは、同期式モードにおいて下記のすべての信号を制御します。
  - MSEBIM_CLK
  - MSEBIM_ALE
  - MSEBIM_CLE
  - MSEBIM_DLE
- また、下記の信号を 1 にセットします。
  - MSEBIM_RD_N
  - MSEBIM_WR_N
- バス上の MSEBI スレーブデバイスは、下記の信号を使用します。
  - MSEBIS_CLK
  - MSEBIS_ALE
  - MSEBIS_CLE
  - MSEBIS_DLE
- また、下記の信号を生成します。
  - MSEBIM_WAIT[n]_N

備　考

タイミングパラメータは、マスタとスレーブの間で適合している必要があります。

- MSEBI バスでの最大バーストサイズは、下記ビットによってマスタ側で設定されます。
  - CPU：bMSEBIM_BURST_SIZEMAX_CPUREAD ビット、および bMSEBIM_BURST_SIZEMAX_CPUWRITE ビット
  - DMA：bMSEBIM_BURST_SIZEMAX_DMAREAD ビット、および bMSEBIM_BURST_SIZEMAX_DMAWRITE ビット

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

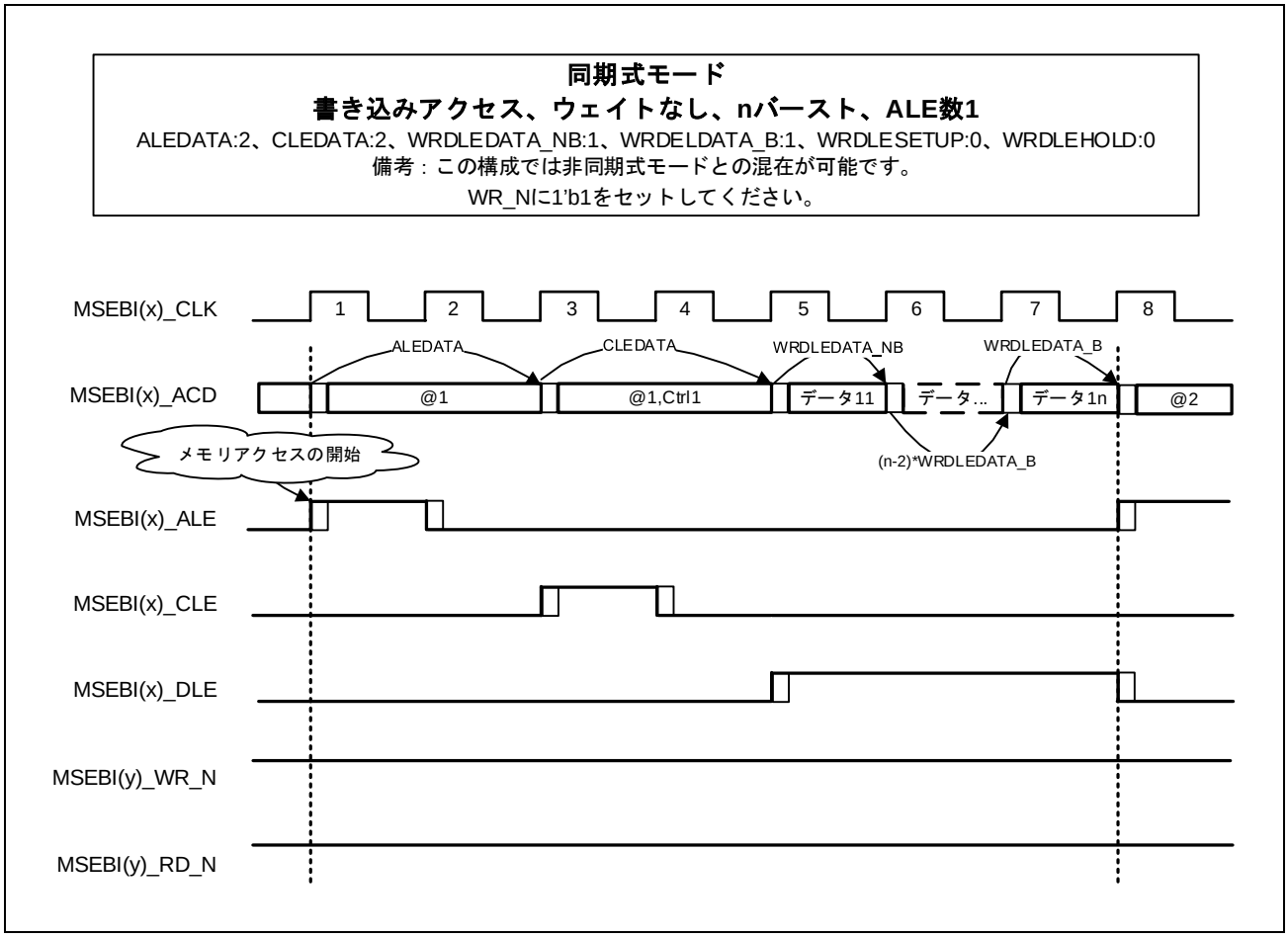


図 10.31　MSEBI タイミング、同期式モード、書き込み 1、ウェイトなし、n バースト、ALE 数 1

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

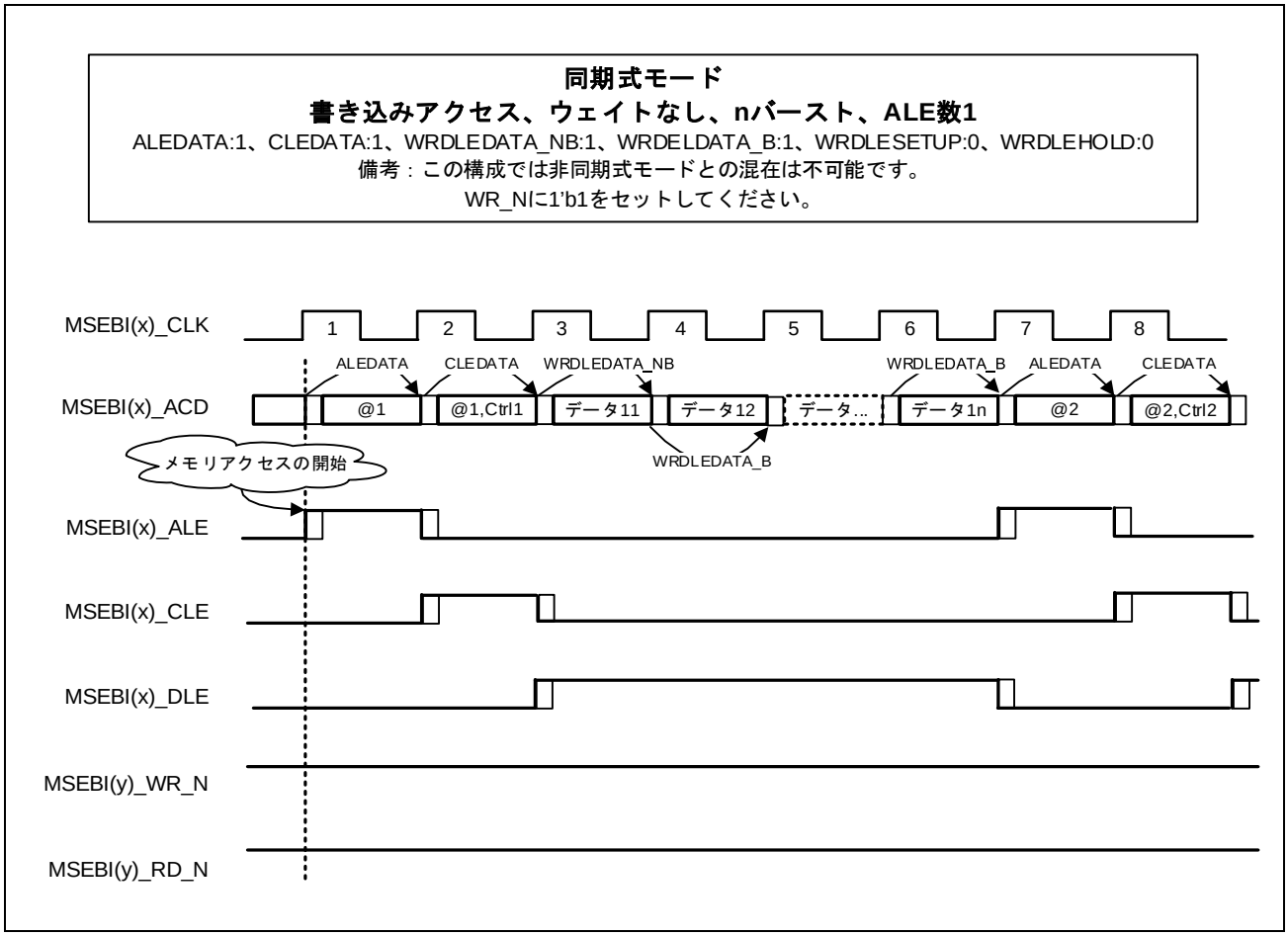


図 10.32　MSEBI タイミング、同期式モード、書き込み 2、ウェイトなし、n バースト、ALE 数 1

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

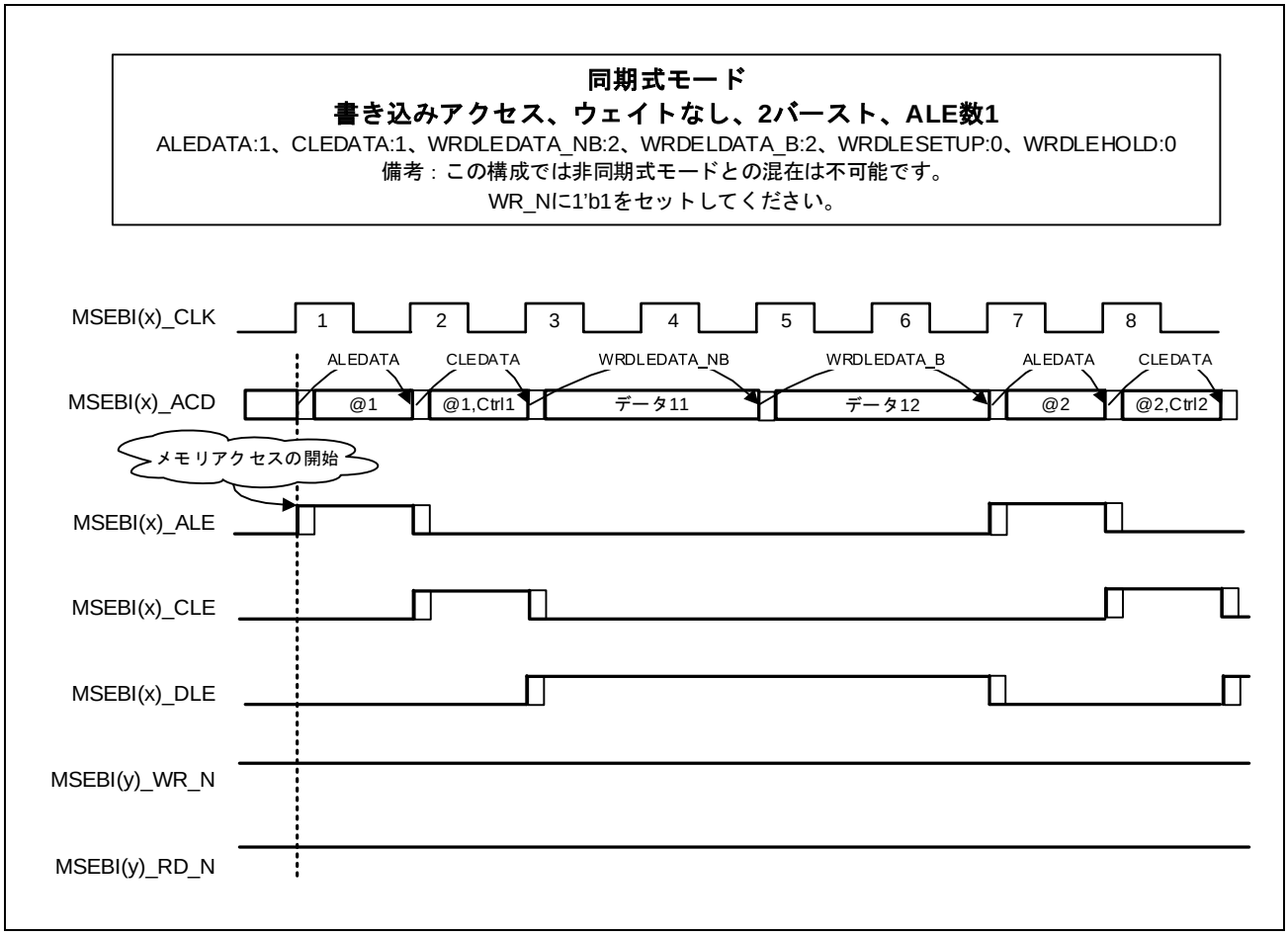


図 10.33　MSEBI タイミング、同期式モード、書き込み 3、ウェイトなし、2 バースト、ALE 数 1

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

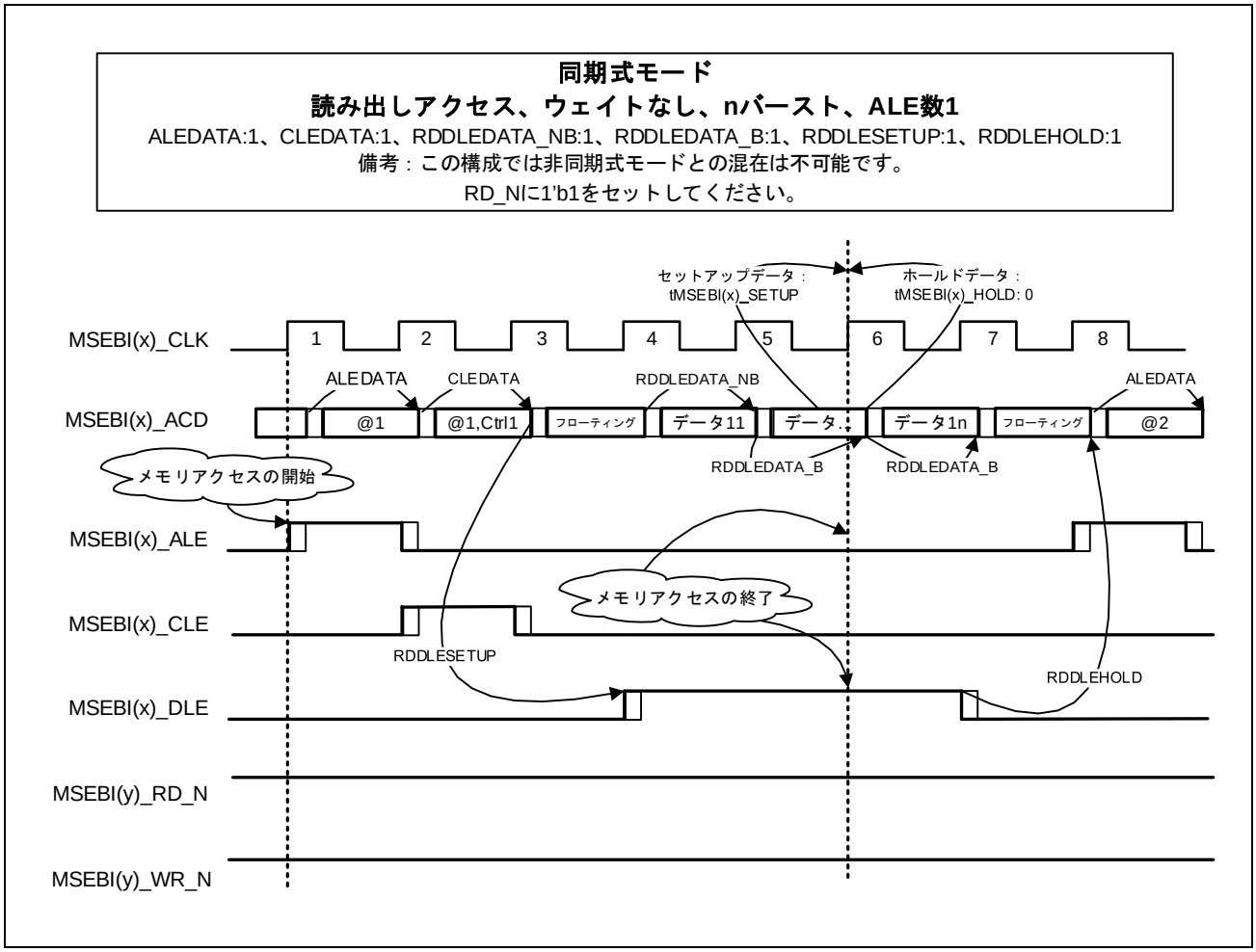


図 10.34　　MSEBI タイミング、同期式モード、読み出し 1、ウェイトなし、n バースト

10.4.4.8　同期式モード、バースト、ALE なし

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

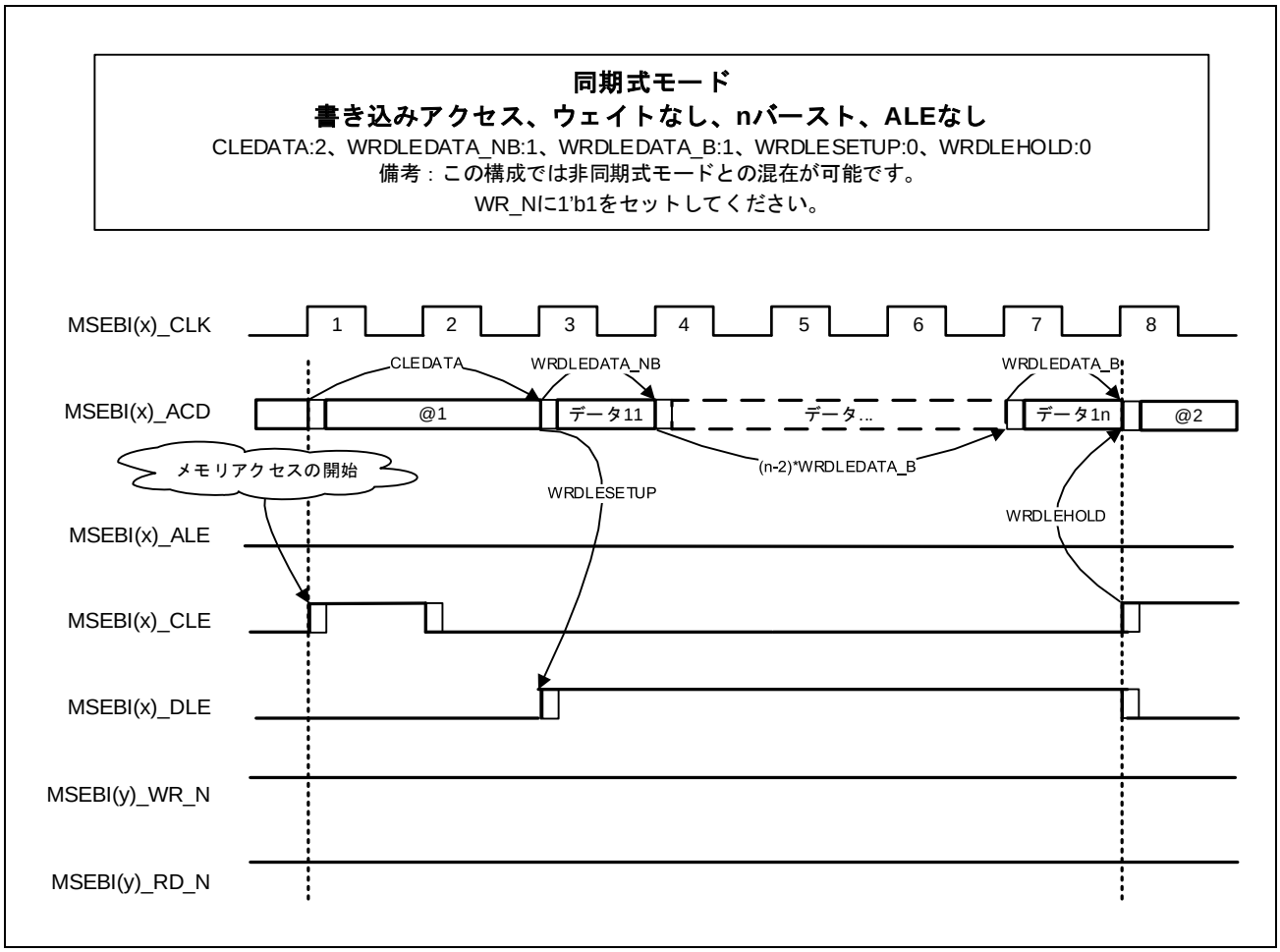


図 10.35　MSEBI タイミング、同期式モード、書き込み 1、ウェイトなし、バースト、ALE なし

信号名に関する注記：

- マスタインタフェースの場合、MSEBI(x)は MSEBIM を表します。
- スレーブインタフェースの場合、MSEBI(x)は MSEBIS を表します。
- MSEBI(y)は MSEBIM のみを表します。

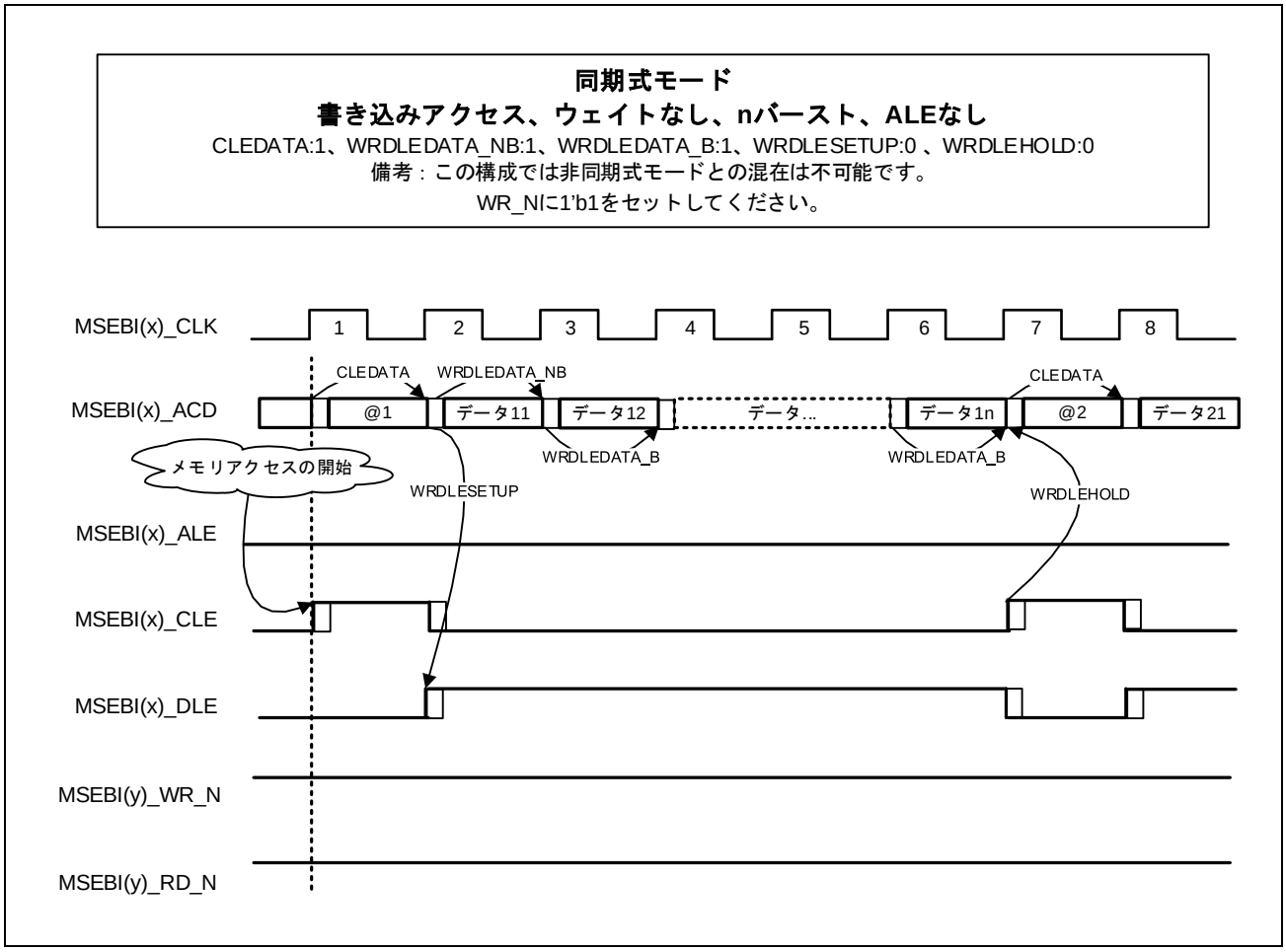


図 10.36　MSEBI タイミング、同期式モード、書き込み 2、ウェイトなし、バースト、ALE なし

10.4.5　MSEBI 割り込み

10.4.5.1　MSEBI 割り込み：概要

MSEBI は 1 本の割り込みライン MSEBIS_Int を備えています。

本割り込みは、スレーブによる「ブロック終了イベント」検出に用いられます。

下記の各項を参照してください。

- レジスタによる割り込みの設定（MSEBI を通じたマスタの CPU によるアクセス）
  - 「**10.3.4.1 rMSEBIS_INT ― スレーブ割り込みレジスタ**」
- レジスタによる割り込みステータスの読み出し（CPU によるアクセス）
  - 「**10.3.3.11 rMSEBIS_STATUS_INT0 ― 割り込みステータスレジスタ**」
  - 「**10.3.3.12 rMSEBIS_STATUS_INT1 ― マスク後割り込みステータスレジスタ**」
- レジスタによる割り込みのマスク（CPU によるアクセス）
  - 「**10.3.3.13 rMSEBIS_MASK_INT ― 割り込みマスクレジスタ**」
- レジスタによる割り込みのクリア（CPU によるアクセス）
  - 「**10.3.3.14 rMSEBIS_CLR_INT ― 割り込みクリアレジスタ**」
- メモリへの書き込み転送を完了するために使用されるディスクリプタのアドレスを格納（CPU によるアクセス）
  - 「**10.3.3.15 rMSEBIS_EOB_ADDR ― ブロック終了アドレスレジスタ**」
  - CPU の MSEBI_CS[n]_N（n＝0～3）ディスクリプタと DMA の MSEBI_CS[n]_N（n＝0、1）ディスクリプタのアドレスは、自動的に計算されます（レジスタの説明を参照してください）。

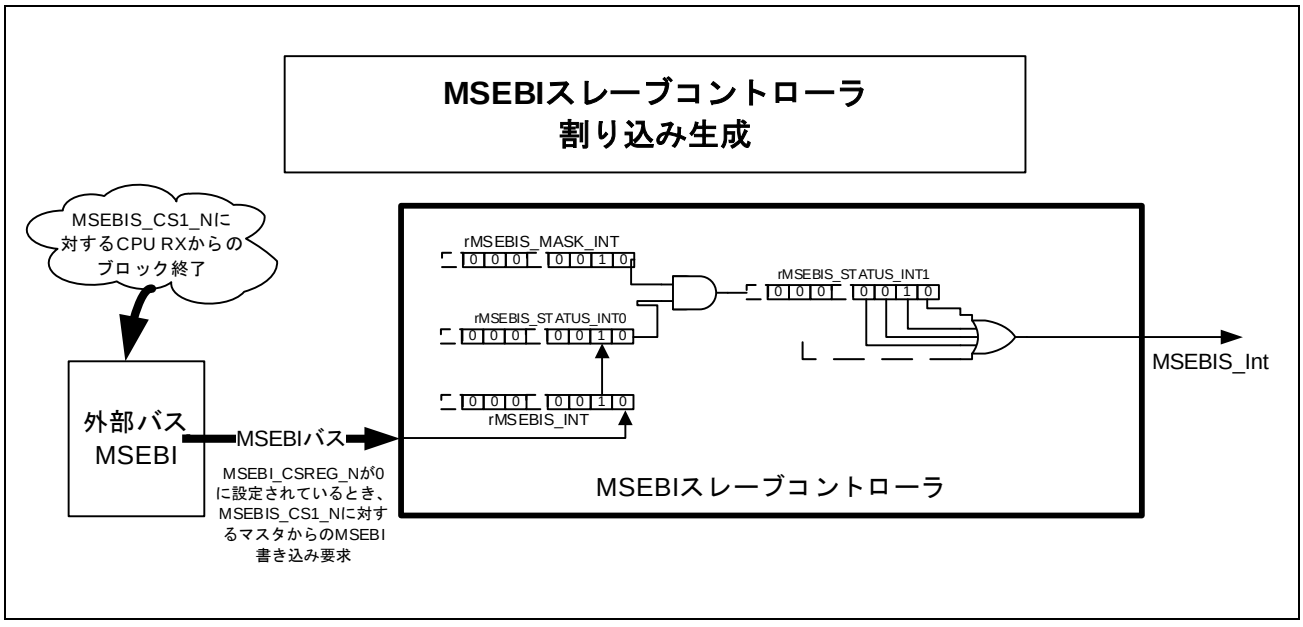


図 10.37　MSEBI スレーブの割り込み生成

10.4.5.2　MSEBI 割り込み：マスタによるブロック終了検出

以下の項では、マスタ側でのブロック終了検出の管理方法について説明します。

DMA TX[n] FIFO（n＝0、1）を用いた転送

- DMA によるブロック転送の終了時に、DMA は割り込みを発生させることで、転送の最後のデータが DMA TX[n] FIFO にプッシュされたことを示すことができます。
  - 「**図 10.48 MSEBI：バーストモード、DMA 送信 FIFO とバスインタフェースの結合**」を参照してください。
- CPU は bMSEBIM_TDMAE1 をポーリングすることにより、本割り込みを管理します。本ステータスビットは、FIFO 内の全データが MSEBI バスに送信され、かつ bMSEBIM_DEST_BLOCK_SIZE 分のシングル要素が送信された後に、自動的にクリアされます。
  - 「**10.4.6.3 MSEBI マスタ：DMA 制御**」を参照してください。

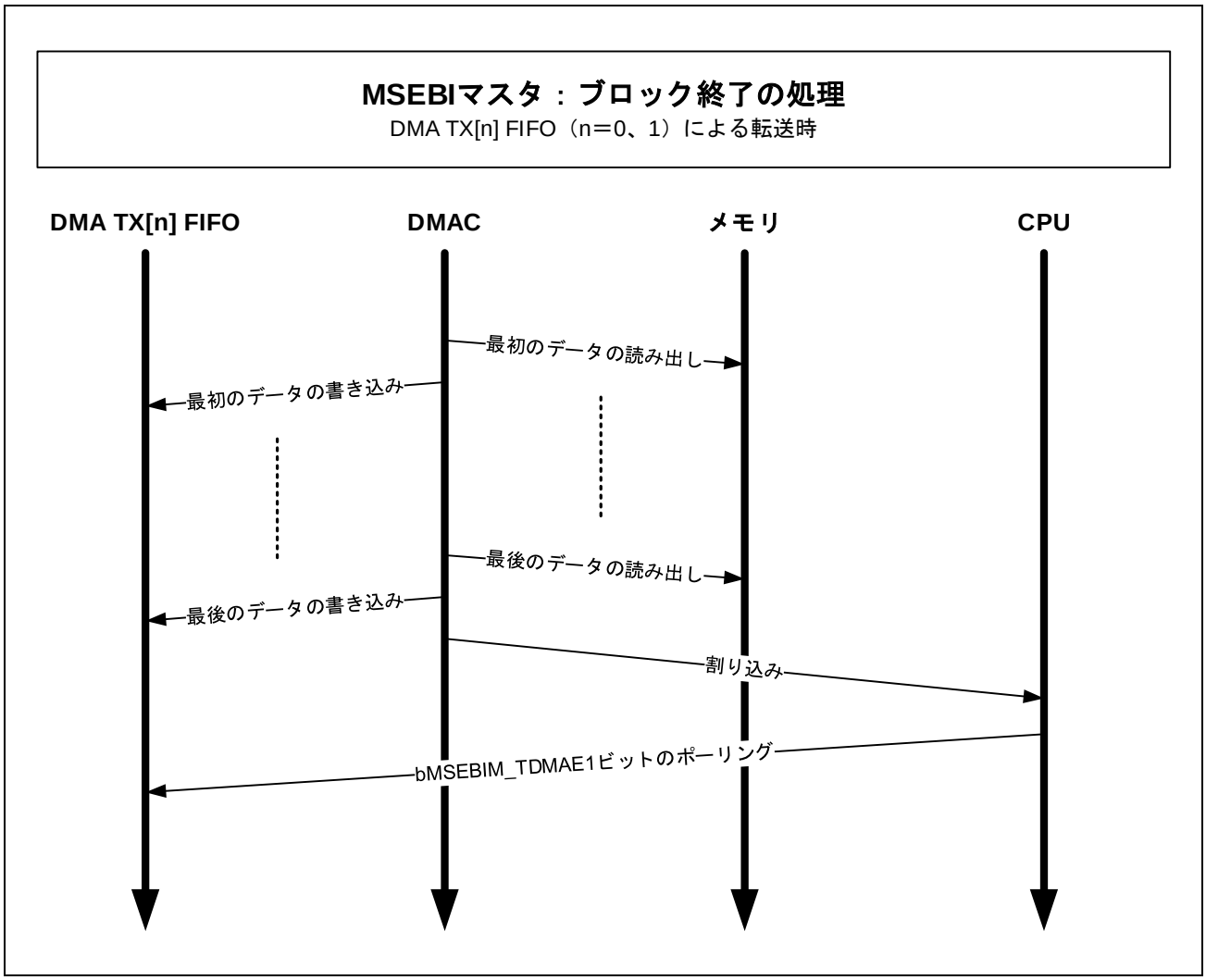


図 10.38　MSEBI マスタ：DMA TX によるブロック転送の終了

DMA RX[n] FIFO（n＝0、1）を用いた転送

- DMA によるブロック転送の終了時に、DMA は割り込みを発生させることで、転送の最後の読み出しデータがメモリに送信されたことを示すことができます。
  - 「**図 10.49 MSEBI：バーストモード、DMA 受信 FIFO とバスインタフェースの結合**」を参照してください。
- CPU は、すべてのデータがメモリ上で実際に利用可能なこと（書き込みバッファに格納されていないこと）を、それらを使用する前に確認する必要があります。これには DMA のライトバック機能を利用できます（ch＝0～7）。

  ライトバック機能とは、ブロック転送の終了時に、DMA 転送のステータス（DMAC.CTL[ch].DONE）が DMA によってメモリにコピーされる動作です。本コピーは、DMAC.LLP[ch]によって指示された位置に、LLI（linked list item）と呼ばれる仕組みで実行されます。

  CPU は、LLI の DMAC.CTL[ch].DONE ビットが 1 になるまで、メモリ上のステータスの位置をポーリングします。下記の理由から、転送の完了が保証されます。
  （1）書き込みアクセス（特定のイニシエータからのアクセス）の順序が NoC で保証されるため
  （2）ライトバックコマンドが、転送の最後の書き込み後に DMA によって送信されるため
  - 「**10.4.6.3 MSEBI マスタ：DMA 制御**」を参照してください。

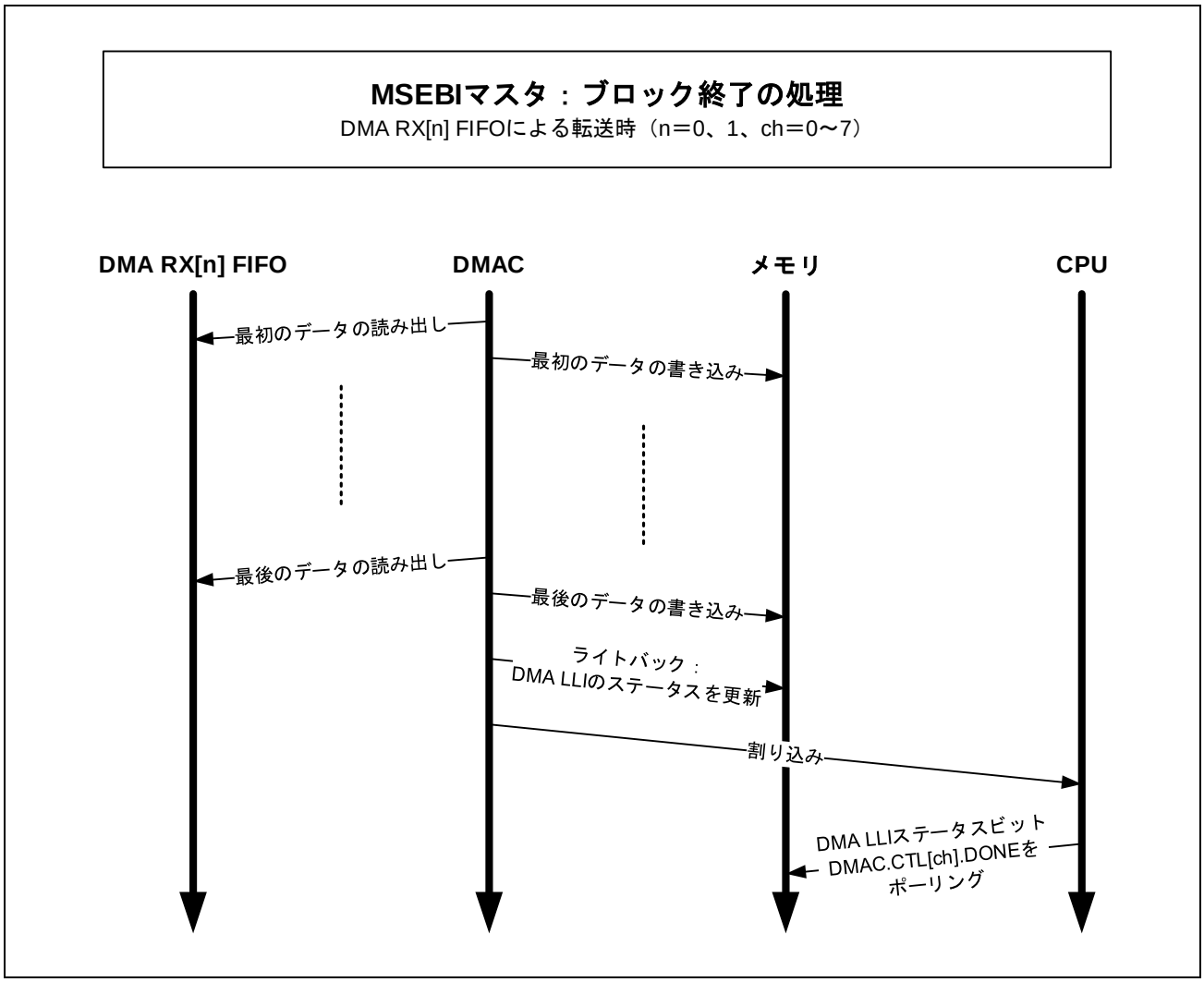


図 10.39　MSEBI マスタ：DMA RX によるブロック転送の終了

CPU を用いた書き込み転送

- CPU によるブロック転送の終了時に、MSEBI コントローラへの最後の書き込み終了は、コマンドが CPU 送信 FIFO にプッシュされたことを示します。
  - 「**図 10.46 MSEBI：バーストモード、CPU 送受信 FIFO とバスインタフェースの結合、例 1**」および「**図 10.47 MSEBI：バーストモード、CPU 送受信 FIFO とバスインタフェースの結合、例 2**」を参照してください。
- CPU は、コマンドが確実にバス上に送信されるように、下記の方法でブロック終了を管理します。
  FIFO レベルビット（bMSEBIM_CPU_TRANSMIT_FIFOLEVEL）をポーリングする方法

  あるいは、MSEBI スレーブでステータスレジスタを読み出す方法
  （1）読み出しおよび書き込み転送は、それらが受信された順序で実行されます。
  （2）MSEBI スレーブで書き込みを強制するには、読み出しを実行する必要があります。読み出すことによって、CPU 送信 FIFO から書き込みコマンドおよび読み出しコマンドが排出されるからです。

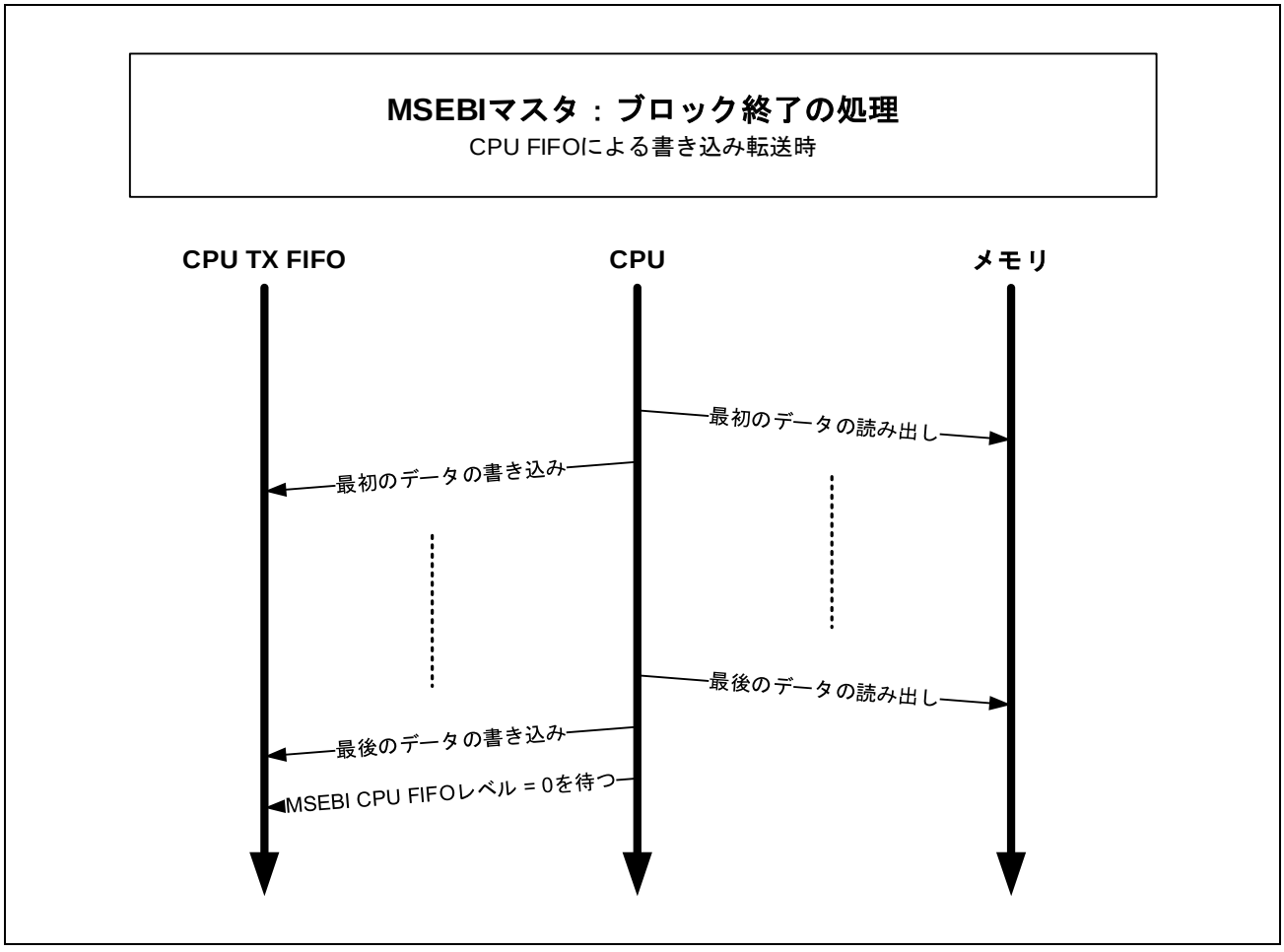


図 10.40　MSEBI マスタ：CPU FIFO による書き込みブロック転送の終了

CPU を用いた読み出し転送

- CPU によるブロック転送の終了時に、読み出しデータは CPU によってメモリにコピーされます。特定のアクションは不要です。
  - 「**図 10.46 MSEBI：バーストモード、CPU 送受信 FIFO とバスインタフェースの結合、例 1**」および「**図 10.47 な MSEBI：バーストモード、CPU 送受信 FIFO とバスインタフェースの結合、例 2**」を参照してください。

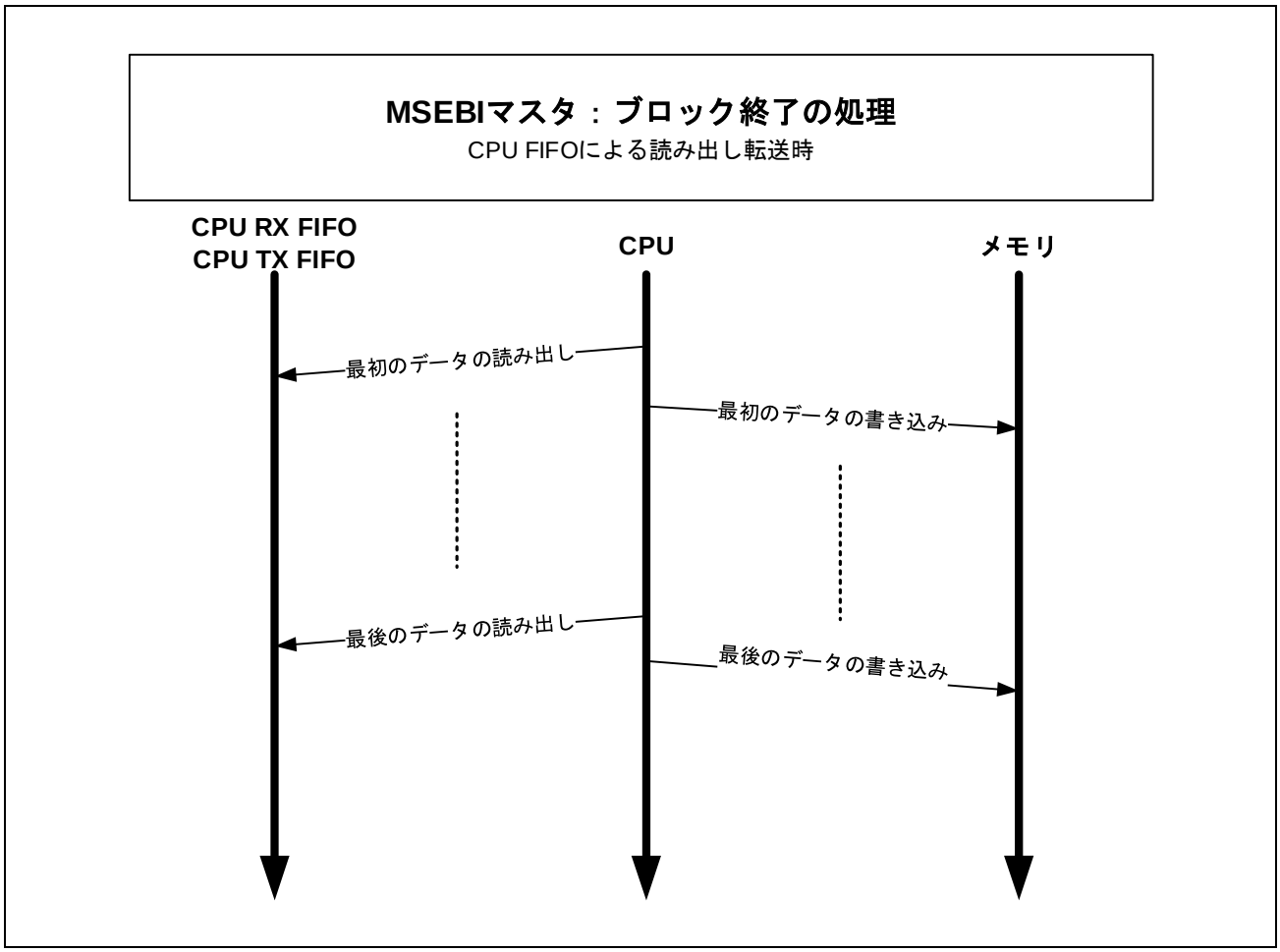


図 10.41　MSEBI マスタ：CPU FIFO による読み出しブロック転送の終了

10.4.5.3　MSEBI 割り込み：スレーブによるブロック終了検出

MSEBI マスタと MSEBI スレーブデバイスの間でブロック転送が終了すると、スレーブデバイスにブロックの完了を通知できます。

以下の項では、スレーブに書き込みブロック転送の終了を検出させるためのシーケンスについて説明します（n＝0、1）。

- MSEBI マスタは、MSEBI マスタコントローラの CPU 部または DMA 部を用いてデータブロックをスレーブに転送します。
  - 下記の図を参照してください。
    「**図 10.58　MSEBI スレーブの CPU FIFO の例 1**」
    「**図 10.61　MSEBI マスタの DMA TX FIFO からの要求に対するスレーブの DMA FIFO**」
- 転送終了時に、マスタ側のファームウェアは、以下のようにして、ブロックが完全にスレーブメモリに書き込まれたことを確認する必要があります。
  - マスタ側のファームウェアは、rMSEBIS_INT レジスタの bMSEBIS_SET_INT_CPUTX ビットまたは bMSEBIS_SET_INT_DMATX ビットに書き込むことによって、スレーブに「ブロック終了イベント」を送信します（「**10.4.7.4　MSEBI スレーブ：マスタによるレジスタアクセス**」を参照してください）。
    rMSEBIS_INT レジスタに書き込む値は、以下の状態に依存します。
    （1）転送のイニシエータ（CPU/DMA）
    （2）送受信側（TX）
    （3）転送に用いられた MSEBI_CS[n]_N
- 「ブロック終了イベント」を受信すると、MSEBI スレーブは、対応する FIFO（CPU 受信 FIFO または DMA 送信 FIFO）にブロック転送ディスクリプタをプッシュします。その位置は、rMSEBIS_EOB_ADDR レジスタで指示されたメモリ上のアドレスです（MSEBI_CS[n]_N とアクセスタイプ（CPU/DMA）によって決まるオフセットが適用されます）。
- MSEBI スレーブは、rMSEBIS_STATUS_INT0 レジスタに割り込みを設定します。
- CPU は、メモリ上のディスクリプタをポーリングしてから、ブロック転送ディスクリプタを 0 にリセットし、さらに rMSEBIS_CLR_INT レジスタで割り込みをクリアすることにより、本割り込みを管理します。

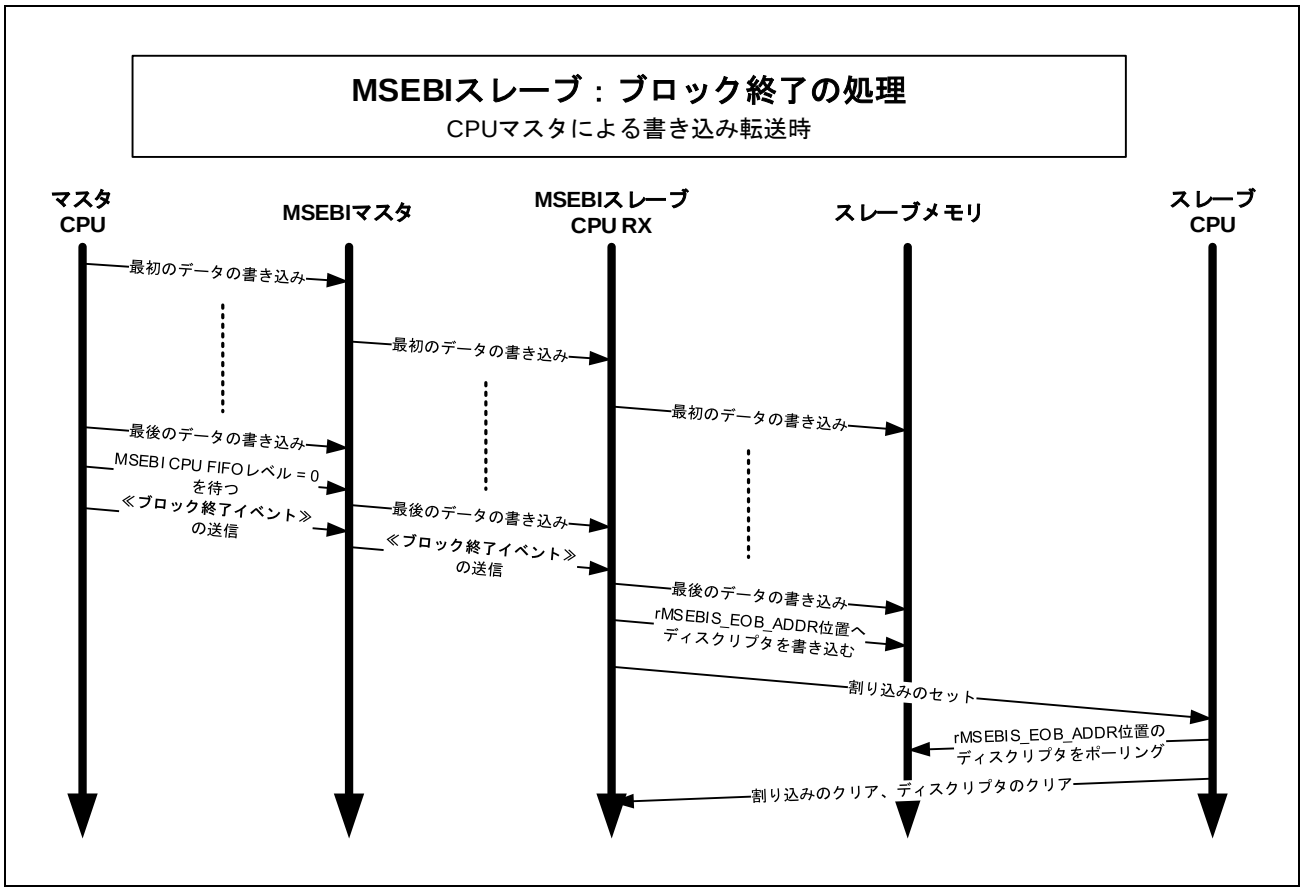


図 10.42　MSEBI スレーブ：MSEBI CPU マスタからの書き込みブロック転送の終了

以下の項では、スレーブに読み出しブロック転送の終了を検出させるためのシーケンスについて説明します（n＝0～3）。

- MSEBI マスタは、MSEBI マスタコントローラの CPU 部または DMA 部を用いてデータブロックをスレーブから読み出します。
  - 下記の図を参照してください。
    「**図 10.58 MSEBI スレーブの CPU FIFO の例 1**」
    「**図 10.62 MSEBI マスタの DMA RX FIFO からの要求に対するスレーブの DMA FIFO**」
- 転送終了時に、マスタ側のファームウェアは、rMSEBIS_INT レジスタに書き込むことによって、スレーブに「ブロック終了イベント」を送信します（「**10.4.7.4 MSEBI スレーブ：マスタによるレジスタアクセス**」を参照してください）。
  - rMSEBIS_INT レジスタの bMSEBIS_SET_INT_CPURX ビットまたは bMSEBIS_SET_INT_DMARX ビットに書き込む値は、以下の状態に依存します。
    （1）転送のイニシエータ（CPU/DMA）
    （2）送受信側（RX）
    （3）転送に用いられた MSEBI_CS[n]_N
- MSEBI スレーブは、rMSEBIS_STATUS_INT0 レジスタに割り込みを設定します。
- スレーブの CPU は、rMSEBIS_CLR_INT レジスタで割り込みをクリアすることにより、本割り込みを管理します。

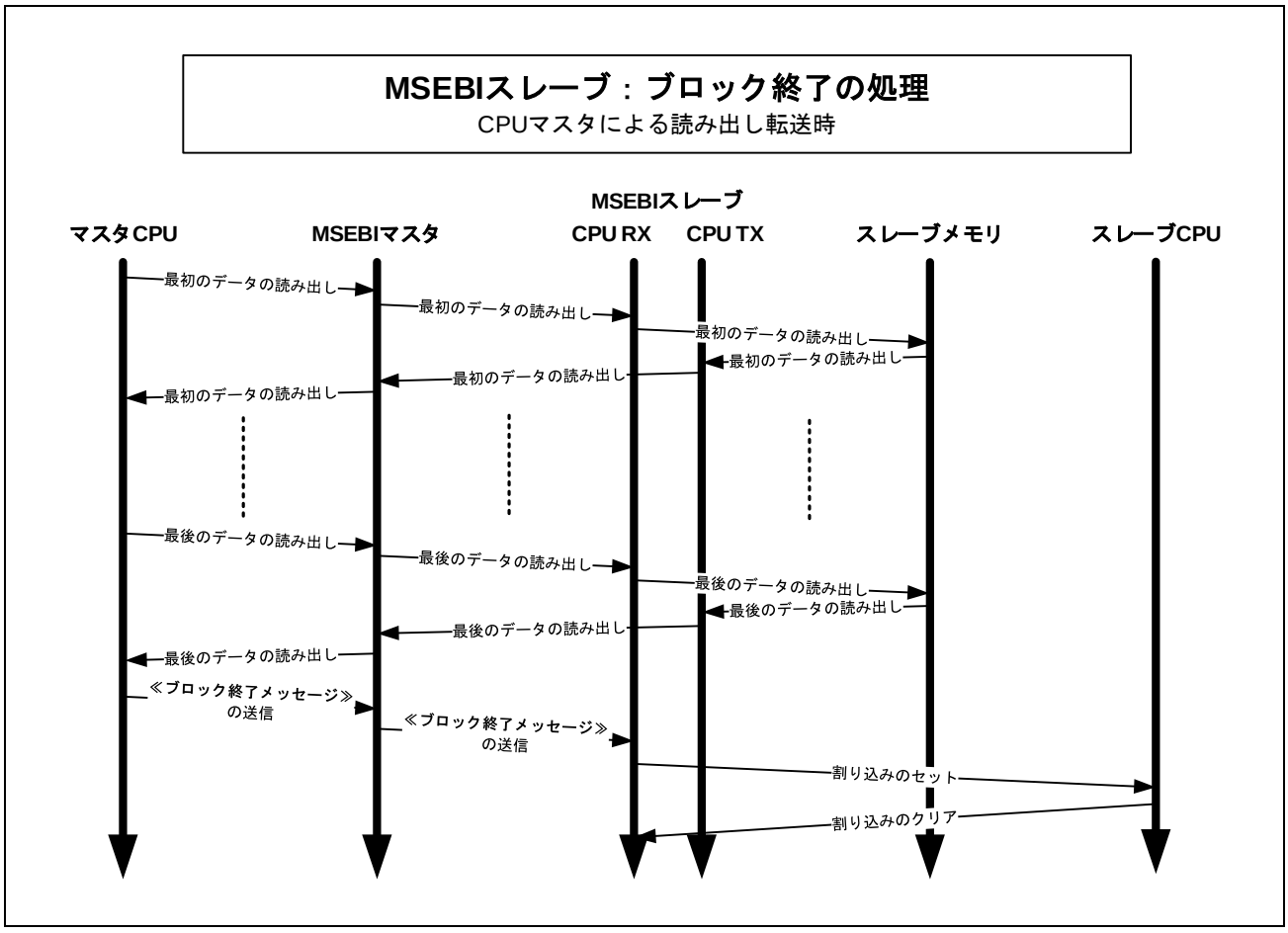


図 10.43　MSEBI スレーブ：MSEBI CPU マスタからの読み出しブロック転送の終了

10.4.6　MSEBI マスタモード

10.4.6.1　マスタモードの概要

主な特長は以下のとおりです。

- 非同期式モードと同期式モード
- 8 ビット、16 ビット、および 32 ビットから選択可能な MSEBI インタフェース幅
- マルチ DLE モード
- MSEBI インタフェースでのバーストモード
- AHB 上での 1-4-8-16 バーストアクセスのサポート
- CPU 受信 FIFO に対するプリフェッチ機能
- 4 チャネルの DMA 接続（外部要求受信可能）
- マスタ用の CPU 送受信 FIFO：2×32 ワード×32 ビット
- マスタ用の DMA 送受信 FIFO：4×32 ワード×64 ビット、設定可能なウォーターマークレベル
- 最大 4 本のチップセレクトライン
- 2B～4GB のプログラマブルアドレス機能
- プログラム可能なセットアップ時間の読み出し／書き込み
- プログラム可能なホールド時間の読み出し／書き込み
- 外部ウェイト要求（許可／禁止の設定可）

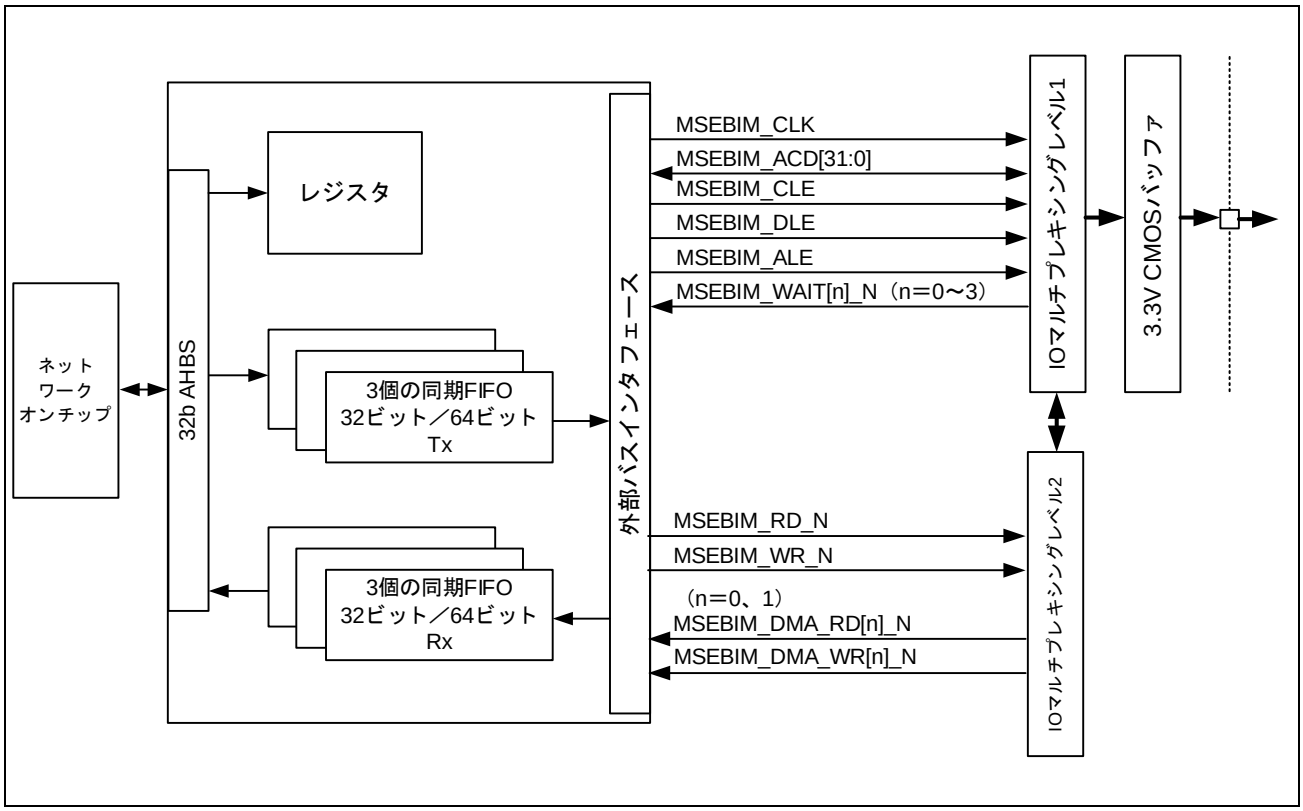


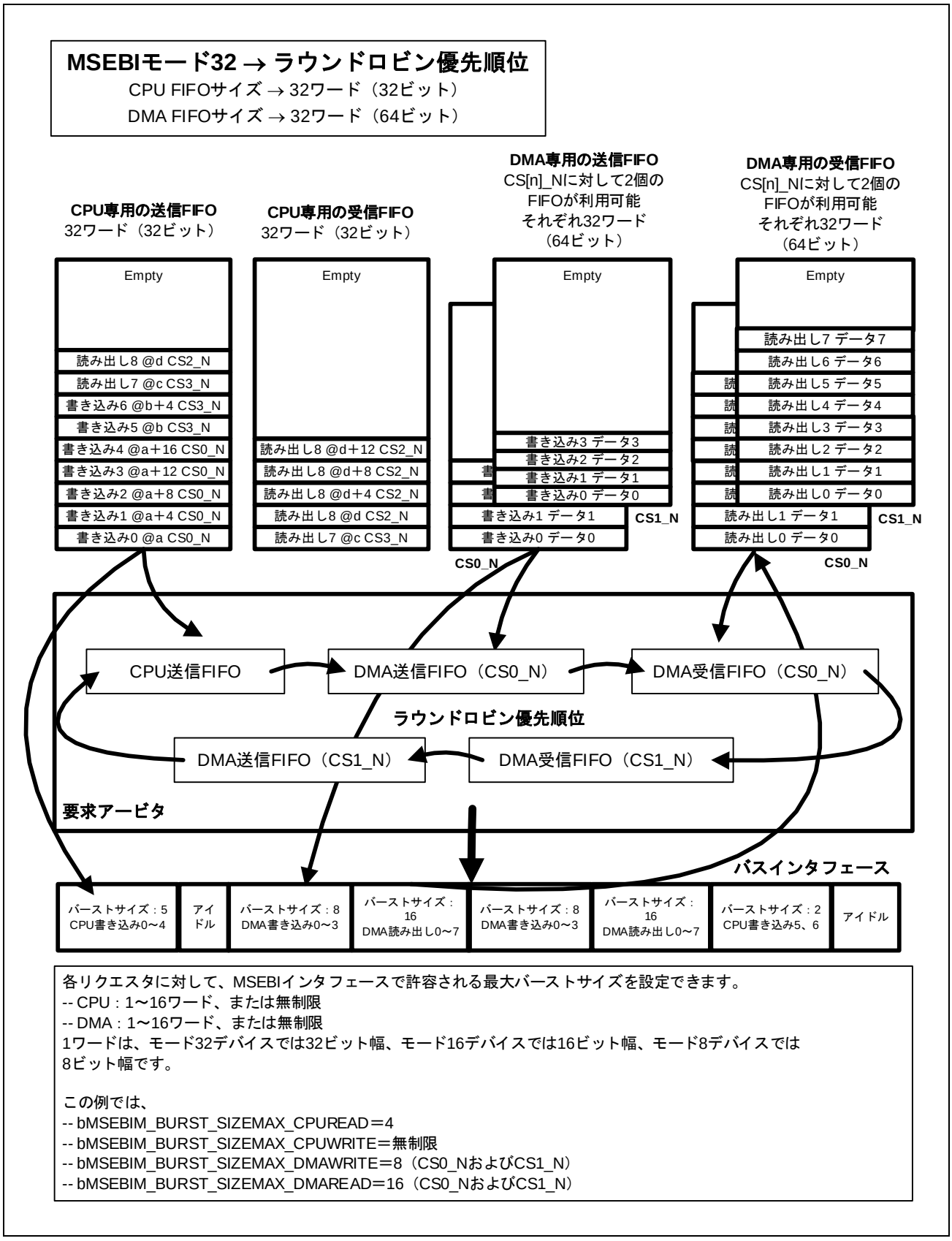
図 10.44　マスタモードの概要

10.4.6.2　MSEBI マスタ：バーストモード

バーストアクセスを管理する場合、MSEBI インタフェースには以下の機能を設定する必要があります。

- 同期式モードを許可（bMSEBIM_CONFIG ビットを設定）
- バーストモードを許可（bMSEBIM_BURST_ENABLE ビットを 1’b1 にセット）
- バーストは各 CS[n]_N（n＝0～3）に対して許可されます。
- バーストはチップセレクトの構成に応じて許可されます。アクセスサイズは下記条件に適合させる必要があります。
  - MSEBI をモード 32 に設定した場合：ワードアクセス
  - MSEBI をモード 16 に設定した場合：ワードおよびハーフワードアクセス
  - MSEBI をモード 8 に設定した場合：全アクセスタイプ
- バーストサイクルを生成できるのは、下記の 5 つのリクエスタです。
  - CPU
  - CS0_N に対する DMA 送信 FIFO
  - CS1_N に対する DMA 送信 FIFO
  - CS0_N に対する DMA 受信 FIFO
  - CS1_N に対する DMA 受信 FIFO
- 各リクエスタ間では、アービタがラウンドロビン優先順位を管理します。
- プリフェッチモードは、下記の設定がされている場合にのみ利用可能です。
  - 各 CS[n]_N（n＝0～3）に対して、bMSEBIM_BURST_ENABLE ビットでバーストモードを許可
  - CPU モードでは、bMSEBIM_BURST_SIZEMAX_CPUREAD ビットを 1 より大きな値に設定
  - DMA モードでは、bMSEBIM_BURST_SIZEMAX_DMAREAD ビットを 1 より大きな値に設定
- MSEBI は可能な限りバーストを生成します。
- バーストサイズは不定です。
  - モード 32 の場合：4～1 キロバイトの範囲であり、1kB 境界を超えません。
  - モード 16 の場合：2～1 キロバイトの範囲であり、1kB 境界を超えません。
  - モード 8 の場合：1～1 キロバイトの範囲であり、1kB 境界を超えません。
- 最大バーストサイズは下記ビットで設定します。
  - CPU：bMSEBIM_BURST_SIZEMAX_CPUREAD ビット、および bMSEBIM_BURST_SIZEMAX_CPUWRITE ビット

    CPU プリフェッチ：bMSEBIM_BURST_SIZEMAX_CPUREAD ビットは、プリフェッチ動作中に読み出される最大ワード数を制御するためにも使用されます。
  - DMA：bMSEBIM_BURST_SIZEMAX_DMAREAD ビット、および bMSEBIM_BURST_SIZEMAX_DMAWRITE ビット
- FIFO サイズ
  - CPU：送受信 FIFO に対して 32 ワード×32 ビット
  - DMA：CS[n]_N（n＝0、1）ごとに、送受信 FIFO に対して 32 ワード×64 ビット



各リクエスタに対して、MSEBIインタフェースで許容される最大バーストサイズを設定できます。
-- CPU：1～16ワード、または無制限
-- DMA：1～16ワード、または無制限
1ワードは、モード32デバイスでは32ビット幅、モード16デバイスでは16ビット幅、モード8デバイスでは8ビット幅です。

この例では、
-- bMSEBIM_BURST_SIZEMAX_CPUREAD＝4
-- bMSEBIM_BURST_SIZEMAX_CPUWRITE＝無制限
-- bMSEBIM_BURST_SIZEMAX_DMAWRITE＝8（CS0_NおよびCS1_N）
-- bMSEBIM_BURST_SIZEMAX_DMAREAD＝16（CS0_NおよびCS1_N）

図 10.45　MSEBI：ラウンドロビン優先順位

バーストアクセスの主な特長は以下のとおりです。

- CPU FIFO の場合、どのチップセレクトにアクセスするときも、アクセス順序を厳格に守る必要があります。
- DMA 送信 FIFO（CS0_N 専用）の場合、チップセレクト CS0_N ではアクセス順序を厳格に守る必要があります。
- DMA 送信 FIFO（CS1_N 専用）の場合、チップセレクト CS1_N ではアクセス順序を厳格に守る必要があります。
- DMA 受信 FIFO（CS0_N 専用）の場合、チップセレクト CS0_N ではアクセス順序を厳格に守る必要があります。
- DMA 受信 FIFO（CS1_N 専用）の場合、チップセレクト CS1_N ではアクセス順序を厳格に守る必要があります。
- 各 FIFO（CPU FIFO と 4 つの DMA FIFO）間では、アクセス順序は守られません。
- バスインタフェースは、バーストアクセスをいつでも停止させることが可能です。
- 外部バス上のバーストは、キャッシュ可能なデータまたはキャッシュ可能でないデータに対して生成され、AHB バーストの情報は使用されません。
- CPU のすべての要求（シングル、バースト、読み出し、書き込み）が FIFO に格納されます。
- 書き込み時、バスインタフェースは常に（FIFO の内容に応じて）可能な限り大きなバーストサイズを生成しようと試みます（ただし最大許容バーストサイズで制限されます）。
- 読み出し時、現在の読み出し要求後に FIFO がエンプティであれば、FIFO がフルになるまで、あるいは bMSEBIM_BURST_SIZEMAX_CPUREAD ビットの最大許容バーストサイズで制限されるまで、バスインタフェースは以降の CPU 読み出し要求がプリフェッチできることを期待して、プリフェッチモードでバーストアクセスを生成します。
  - アドレスは、8/16/32 ビットモードのそれぞれで 1/2/4 ずつリニアにインクリメントされ、決して 1kB 境界を超えません。

(1) マスタの CPU FIFO

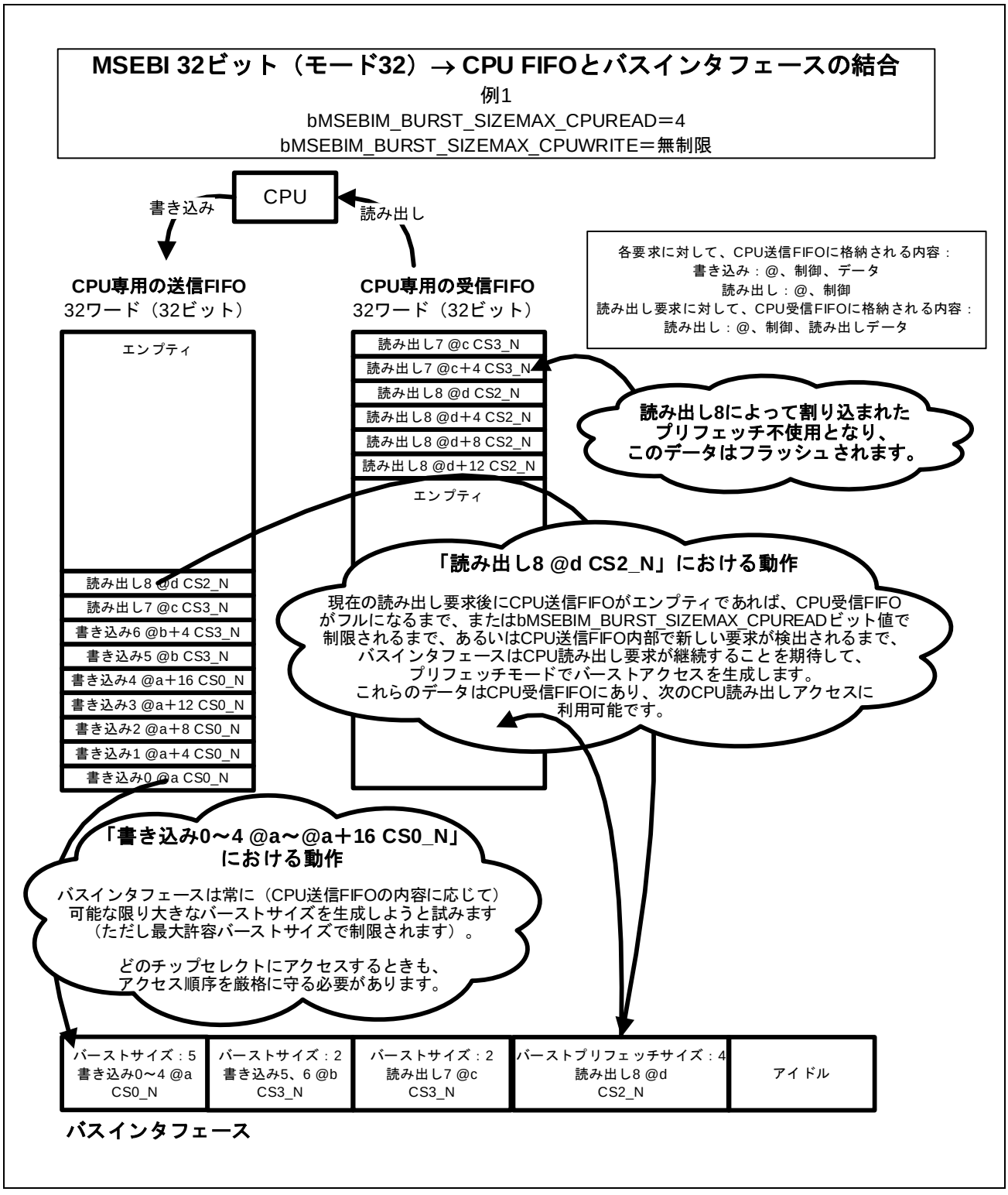


図 10.46　MSEBI：バーストモード、CPU 送受信 FIFO とバスインタフェースの結合、例 1

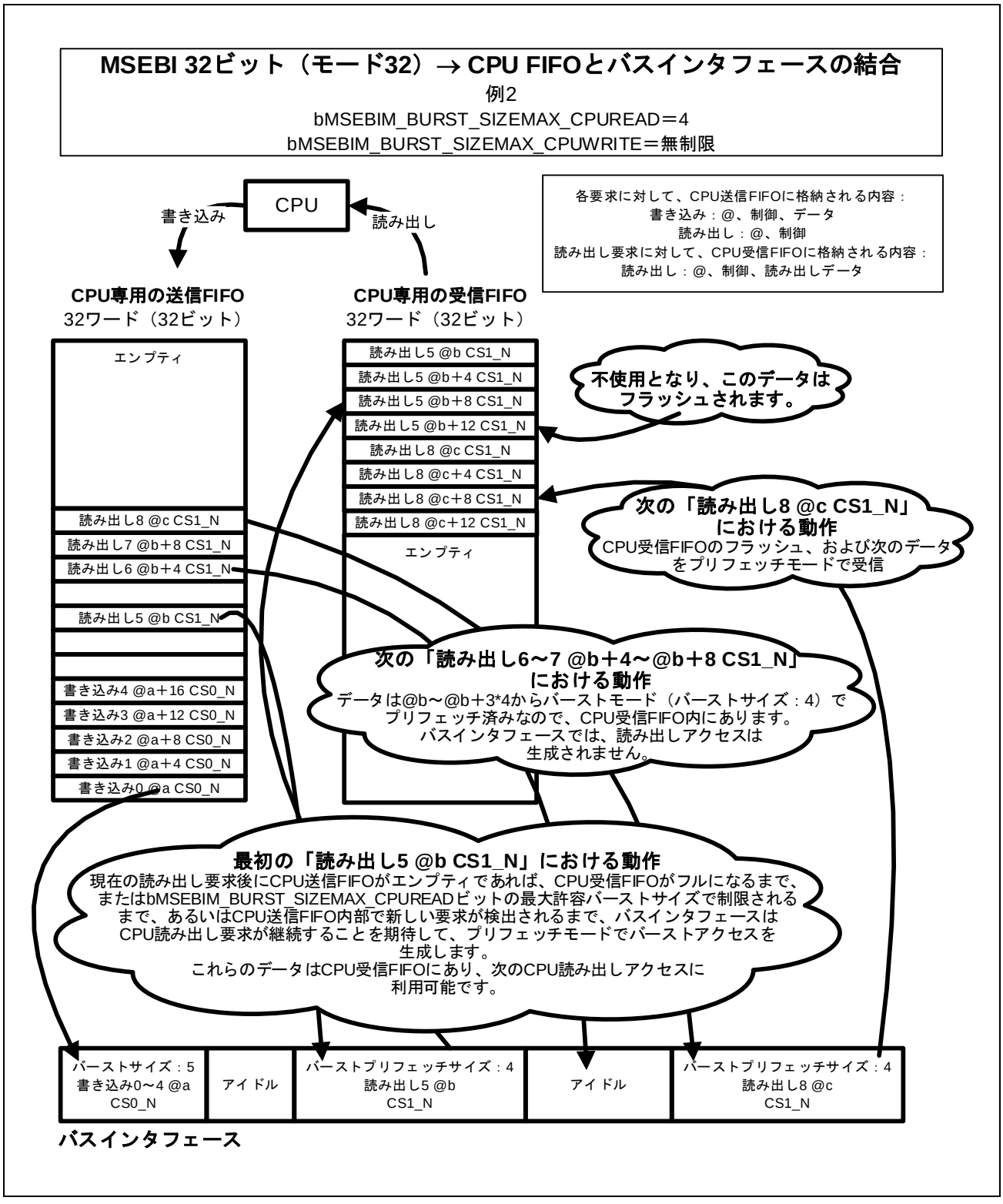


図 10.47　　MSEBI：バーストモード、CPU 送受信 FIFO とバスインタフェースの結合、例 2

(2) マスタの DMA FIFO

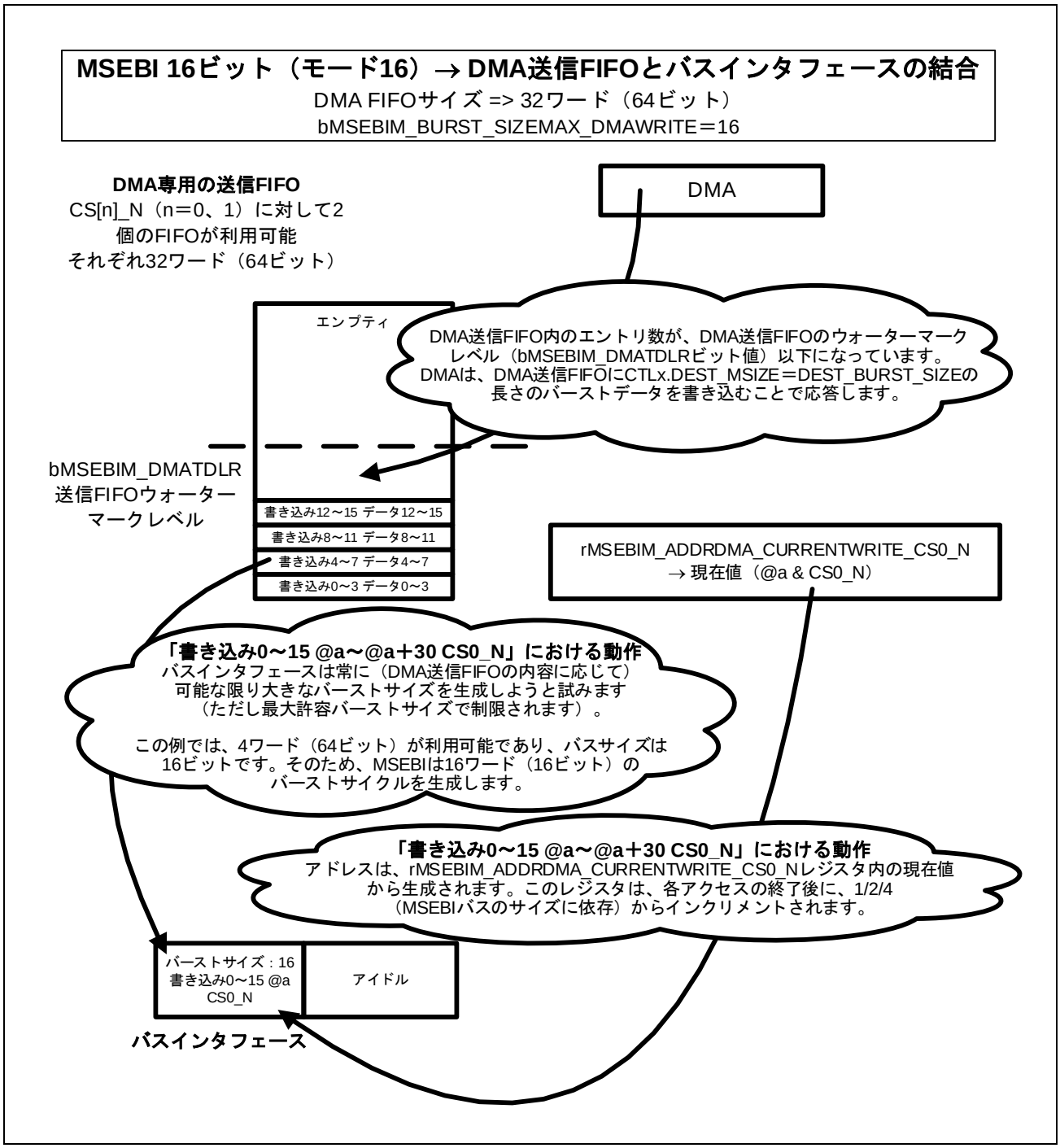


図 10.48　MSEBI：バーストモード、DMA 送信 FIFO とバスインタフェースの結合

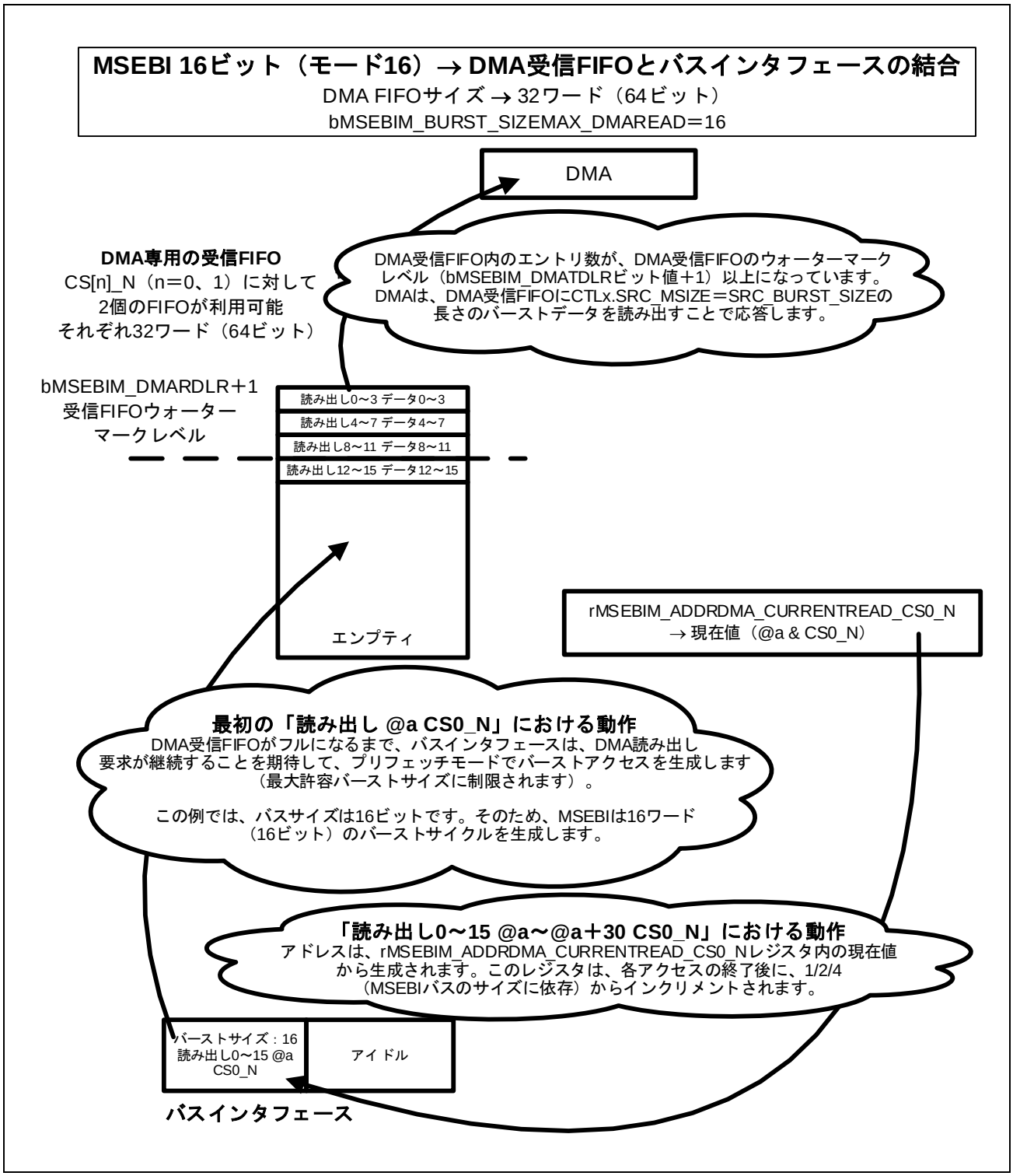


図 10.49　MSEBI：バーストモード、DMA 受信 FIFO とバスインタフェースの結合

(3) マスタのマルチ DLE

マルチ DLE モードは、チップセレクトごとに（bMSEBIM_MULTI_DLE ビットで）個別に設定可能です。ただし以下の制限があります。

- 書き込みのみをサポート
- ウェイト管理なし
- WRDLEDATA_NB/WRDLEDATA_B は、MSEBIM_CLK で 2 サイクル以上であること

本モードは、下記の可能性がある場合、外部 FPGA のコンフィグレーションに使用されることができます。

- 外部フラッシュではコストがかかり、メンテナンスが困難な場合
- JTAG やパッシブシリアルでは低速すぎる場合
- パラレルアクセスで、MSEBI_DLE 信号をクロックとして用いると、MSEBI バスによって駆動が可能な場合
  - 本方式の利点は以下のとおりです。
    ＜JTAG 方式より高速＞
    標準モードでは、FPGA は RZ/N1 と通信を行う必要があります。MSEBI は、RZ/N1 と FPGA の間の標準の通信に利用できますが、FPGA のプログラミングにも使用するなら、他のバスを実装する必要がなくなります（PCB 上の端子数やスペースを節減できます）。
  - こうした MSEBI の標準的な使用法を超えた使い方には不便さもあります。それは、シングルアクセスしかサポートされない点です。

マルチ DLE モードの基本的な考え方は、MSEBI をバーストモードで使用して、外部 FPGA のコンフィグレーションを行えるようにすることです。下図に示すように、マルチ DLE モードによって、FPGA のコンフィグレーションインタフェースの管理が可能になります。MSEBIM_DLE 信号は、コンフィグレーションクロック入力として使用され、さらに各データの有効信号の立ち上がりも提示します。MSEBIM_CLK、MSEBIM_ALE、MSEBIM_CLE の各信号は使用されません。通信は、ターゲットの FPGA に依存して 8 または 16 ビットで行われます。

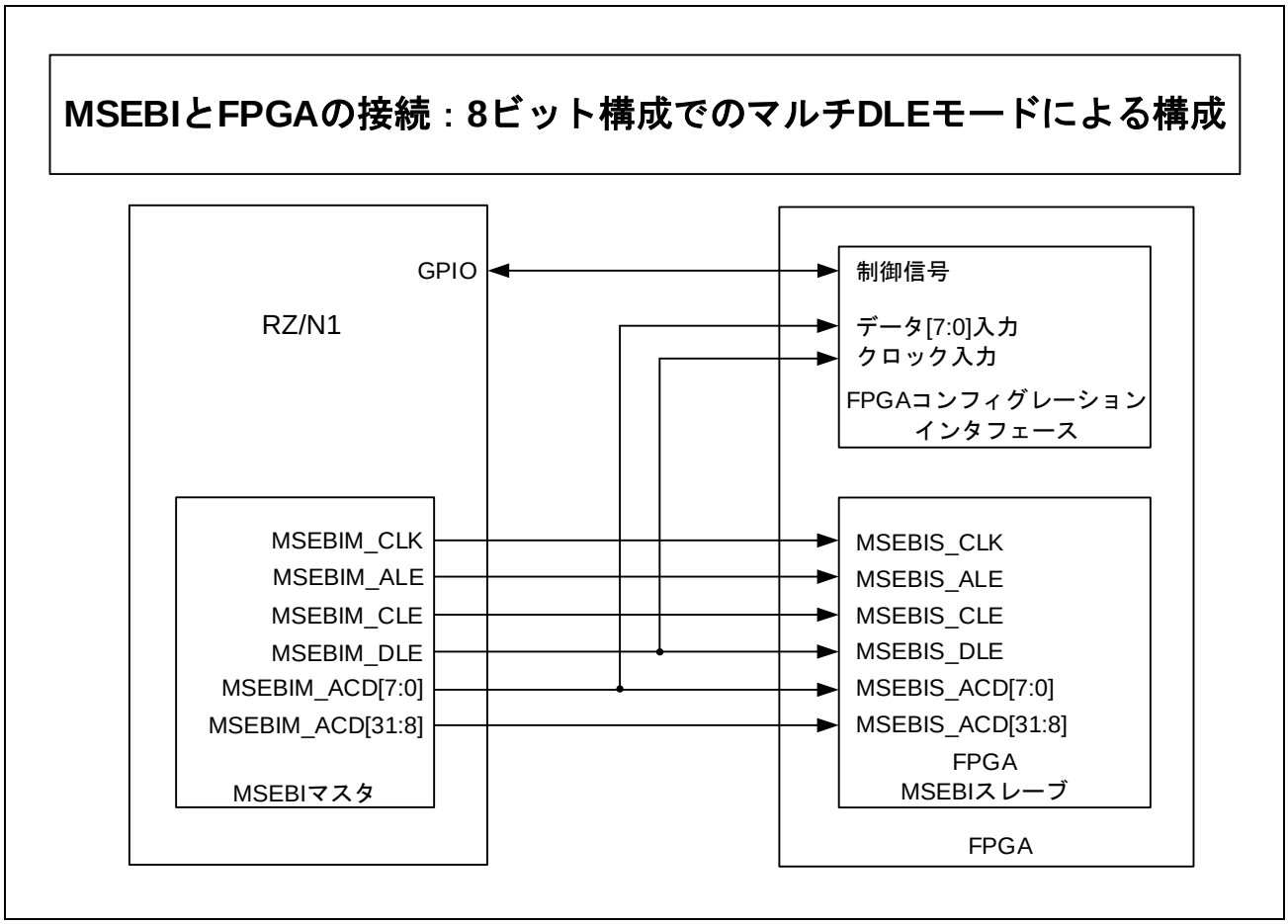


図 10.50　　MSEBI とコンフィグレーションのための FPGA の接続：8 ビット設定でのマルチ DLE モード

注　意

- FPGA のコンフィグレーション後に MSEBI を標準モードで使用できるようにする場合、コンフィグレーションクロック入力はコンフィグレーション専用端子となっているため、MSEBI_DLE 信号は並行して FPGA の他の IO にルーティングする必要があります。
- ターゲットの FPGA によっては、一部のコンフィグレーションデータ端子がコンフィグレーション専用なので、いくつかの MSEBI_ACD ビットは並行して FPGA の他の IO にルーティングする必要があります。

マルチ DLE モードでは、MSEBI_DLE 信号は WRDLEDATA_NB または WRDLEDATA_B の最後のサイクルでセットされます。下図を参照してください。それぞれ WRDLEDATA_NB と WRDLEDATA_B が 2 および 3 に設定されています。

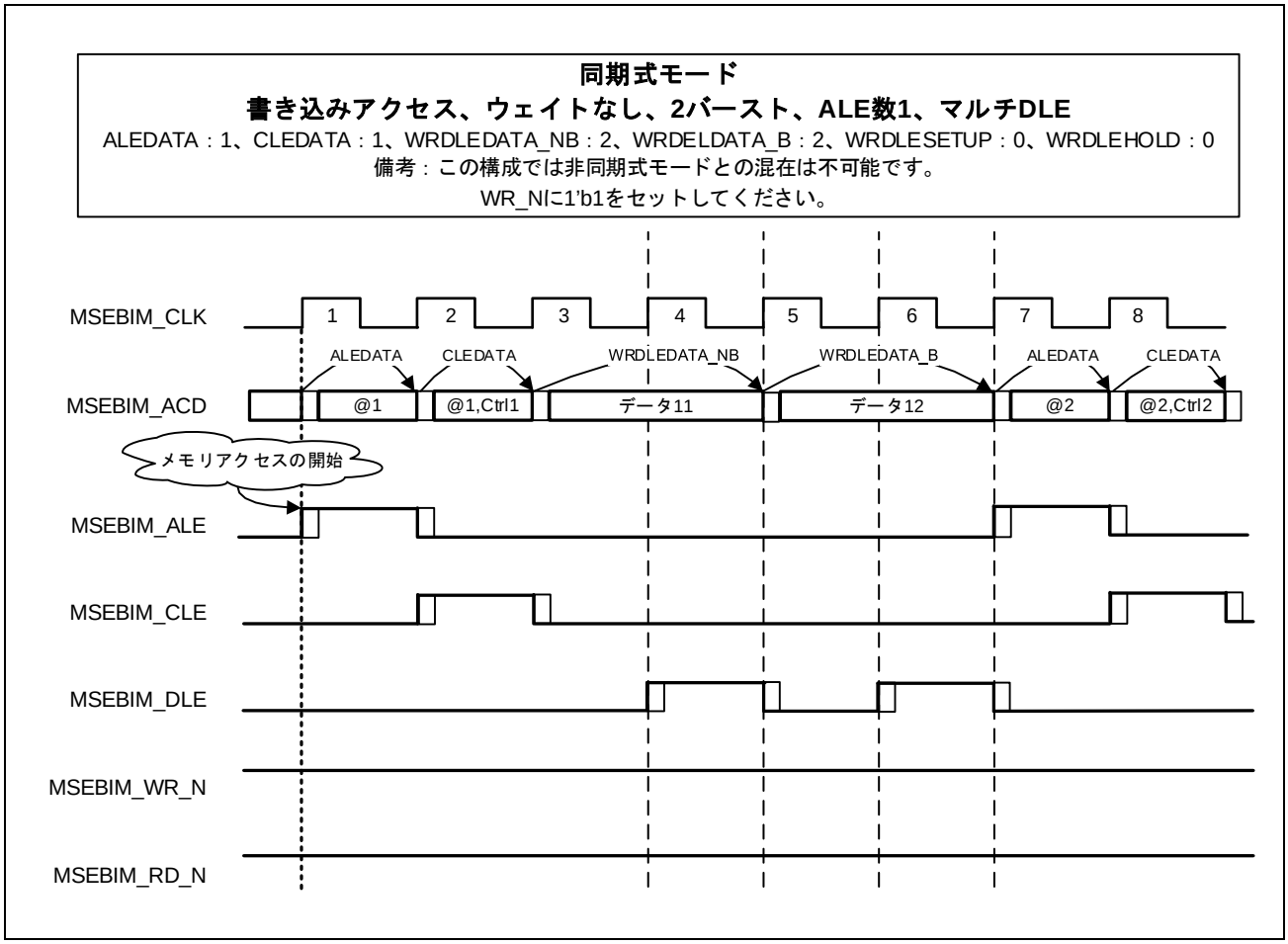


図 10.51　　タイミング 1、同期式モード、バースト、マルチ DLE

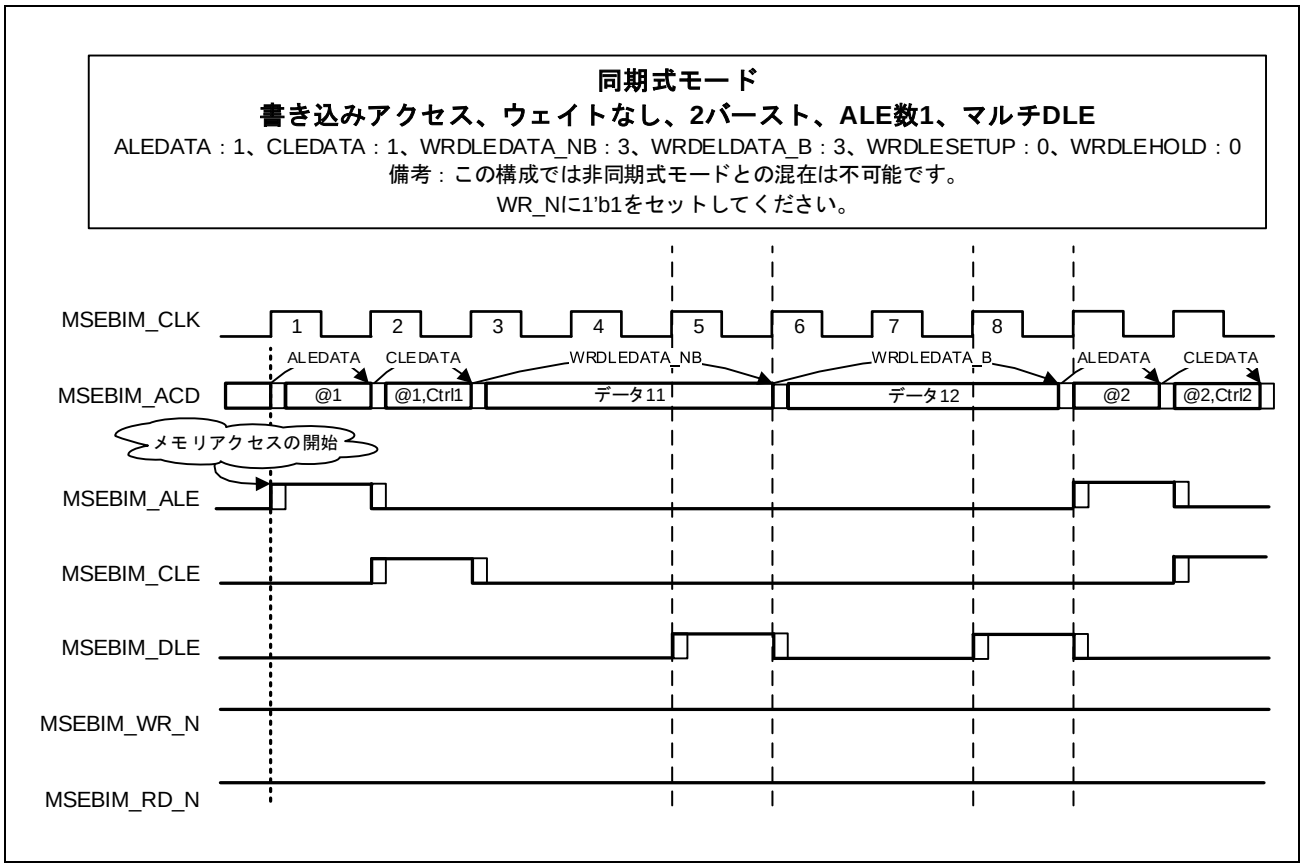


図 10.52　　タイミング 2、同期式モード、バースト、マルチ DLE

10.4.6.3　MSEBI マスタ：DMA 制御

本 MSEBI コントローラには、DMA 機能があります。本機能は、転送の要求および制御のために、DMA コントローラに対するハンドシェイクインタフェースをサポートしています。DMA との間のデータ転送には AHB バスが用いられます。本モードでは、DMA コントローラは周辺機能フローコントローラモードに設定する必要があります。DMA は効率性を考えて、可能な場合は常に DMA バーストトランザクションを使用してデータを転送します。

MSEBI コントローラは 4 本の DMA チャネルを使用します。2 本は送信データ用で、後の 2 本は受信データ用です（CS0_N と CS1_N のそれぞれに対して 1 本ずつあります）。

MSEBI には以下の DMA レジスタがあります。

- bMSEBIM_USE_EXT_RDDMA_REQ ビットおよび bMSEBIM_USE_EXT_WRDMA_REQ ビット：外部端子の DMA 制御動作を許可するコントロールレジスタ
- rMSEBIM_TDMACR_CS0_N、rMSEBIM_RDMACR_CS0_N、rMSEBIM_TDMACR_CS1_N、および rMSEBIM_RDMACR_CS1_N の各レジスタ：DMA 動作の設定（ブロックサイズ、バースト要求サイズ）と、DMA 転送の開始と停止のためのコントロールレジスタ
- bMSEBIM_DMATDLR ビット（送信ウォーターマークレベル）：DMA 要求が生成されるときの DMA 送信 FIFO レベルを設定するレジスタ
- bMSEBIM_DMARDLR ビット（受信ウォーターマークレベル）：DMA 要求が生成されるときの DMA 受信 FIFO レベルを設定するレジスタ
- bMSEBIM_BURST_SIZEMAX_DMAWRITE ビットおよび bMSEBIM_BURST_SIZEMAX_DMAREAD ビット：書き込み／読み出しアクセス時の最大許容バーストサイズ

バーストアクセスを管理する場合、MSEBI インタフェースには以下の機能を設定する必要があります。

- 同期式モードを許可（bMSEBIM_CONFIG ビットをセット）
- バーストモードを許可（bMSEBIM_BURST_ENABLE ビットを 1’b1 にセット）

最大バーストサイズは下記ビットで設定します。

- DMA：bMSEBIM_BURST_SIZEMAX_DMAREAD ビット、および bMSEBIM_BURST_SIZEMAX_DMAWRITE ビット

MSEBI の DMA コントローラインタフェースを有効にし、さらに送信ハンドシェイクインタフェースを有効にするには、以下の設定を行います（n＝0、1）。

- MSEBI は 64 ビットのシングルトランザクションのサイズをサポートしています。（rMSEBIM_TDMACR_CS[n]_N レジスタの bMSEBIM_SINGLE_DEST_WIDTH フィールドを参照してください）
- プロセッサが送信すべきシングルトランザクション数（ブロックサイズ）を MSEBI へ設定する必要があります。そのためには、DMA 送信 FIFO に対して、MSEBI の rMSEBIM_TDMACR_CS[n]_N レジスタの bMSEBIM_DEST_BLOCK_SIZE フィールドを設定します。
- MSEBI は、DMA FIFO 用に予約されたメモリ範囲内にあるアドレスで受信したすべての AHB 要求を、FIFO へのプッシュとして使用します（「**10.3.2.2 rMSEBIM_DMA_FIFOWRITE_CS[n]_N ― DMA 送信 FIFO（64KB）（n＝0、1）**」を参照してください）。最大の要素数（最大 DEST_BLOCK_SIZE＝8191 のシングル要素）を転送できるようにするには、DMAC レジスタ（DMAC.DAR[ch]）の宛先ポインタに FIFO のベースアドレスを設定する必要があります。

- プロセッサがバースト転送のサイズ（バーストサイズ）を MSEBI へ設定する必要があります。そのためには、DMA 送信 FIFO に対して、MSEBI の rMSEBIM_TDMACR_CS[n]_N レジスタの bMSEBIM_DEST_BURST_SIZE フィールドを設定します。
- rMSEBIM_DMATDLR_CS[n]_N レジスタの bMSEBIM_USE_EXT_WRDMA_REQ ビットフィールドに 1 を書いて、外部 DMA 要求を許可します。
- DMA 書き込みアクセス用アドレス（rMSEBIM_ADDRDMA_WRITE_CS[n]_N レジスタ）を書き込みます。これは、ファームウェアが bMSEBIM_TDMAE1 ビットに “1”をセット（立ち上がり）したとき、DMA 送信 FIFO から MSEBI バスへ DMA 転送を開始するため、DMA コントローラによって使用される先頭ブロックアドレスです。
- 送信ウォーターマークレベル（bMSEBIM_DMATDLR ビット）を設定します。本ビットフィールドは、DMA バースト要求が送信ロジックによって生成されるときのレベルを制御します。
- rMSEBIM_TDMACR_CS[n]_N レジスタの bMSEBIM_TDMAE1 ビットフィールドに 1 を書き込みます。この時点で、DMA 送信チャネルは、すべてのデータ（DEST_BLOCK_SIZE 分のシングルトランザクション）が転送されるまで実行中となります。すべてのデータ（DEST_BLOCK_SIZE 分のシングルトランザクション）が DMA 送信 FIFO から MSEBI バスに転送されると（FIFO がエンプティになると）、DMA コントローラは停止します（bMSEBIM_TDMAE1 ビットが 0 にリセットされます）。

備　考

bMSEBIM_TDMAE1 ビットの立ち上がりは、DMA 送信 FIFO をフラッシュします。

bMSEBIM_USE_EXT_WRDMA_REQ ビットに 1 を書いて、外部 DMA 要求を許可した場合：

bMSEBIM_TDMAE1 ビットが 0 から 1（DMA 開始）になると、MSEBI コントローラは、MSEBIM_DMA_WR0_N または MSEBIM_DMA_WR1_N（使用する CS_N に依存）が 0 にリセットされるのを待ちます。

このとき、DMA 送信 FIFO 内のエントリ数がウォーターマークレベル（bMSEBIM_DMATDLR ビット値）以下であれば、ただちに DMA コントローラに対して DMA 送信 FIFO 要求が出されます。

DMA は、DMA 送信 FIFO バッファにバーストデータを書き込むことで応答します。

このとき、DMA 送信 FIFO は読み出され、バーストモードで読み出しデータが MSEBI バスに出力されます。

bMSEBIM_USE_EXT_WRDMA_REQ ビットに 0 を書いて、外部 DMA 要求を禁止した場合：

同じように機能しますが、外部 DMA 要求を使用しません（MSEBIM_DMA_WR0_N または MSEBIM_DMA_WR1_N が 0 にリセットされます）。

DMA 転送は即座に開始されます。

MSEBI の DMA コントローラインタフェースを有効にし、さらに受信ハンドシェイクインタフェースを有効にするには、以下の設定を行います（n＝0、1）。

- MSEBI は 64 ビットのシングルトランザクションのサイズをサポートしています。（rMSEBIM_RDMACR_CS[n]_N レジスタの bMSEBIM_SINGLE_SRC_WIDTH フィールドを参照してください）
- プロセッサによって MSEBI のプログラミングを行い、受信すべきシングルトランザクション数（ブロックサイズ）を設定する必要があります。そのためには、DMA 受信 FIFO に対して、MSEBI の rMSEBIM_RDMACR_CS[n]_N レジスタの bMSEBIM_SRC_BLOCK_SIZE フィールドを設定します。

- MSEBI は、DMA FIFO 用に予約されたメモリ範囲内にあるアドレスで受信したすべての AHB 要求を、FIFO の読み出しとして使用します（「**10.3.2.1 rMSEBIM_DMA_FIFOREAD_CS[n]_N ― DMA 受信 FIFO（64KB）（n＝0、1）**」を参照してください）。最大の要素数（最大 DEST_BLOCK_SIZE＝8191 のシングル要素）を転送できるようにするには、DMAC レジスタ（SAR[ch]）の送信元ポインタに FIFO のベースアドレスを設定する必要があります。
- プロセッサによって MSEBI のプログラミングを行い、バースト転送のサイズ（バーストサイズ）を設定する必要があります。そのためには、DMA 受信 FIFO に対して、MSEBI の rMSEBIM_RDMACR_CS[n]_N レジスタの bMSEBIM_SRC_BURST_SIZE フィールドを設定します。
- rMSEBIM_DMARDLR_CS[n]_N レジスタの bMSEBIM_USE_EXT_RDDMA_REQ ビットフィールドに 1 を書いて、外部 DMA 要求を許可します。
- DMA 読み出しアクセス用アドレス（rMSEBIM_ADDRDMA_READ_CS[n]_N レジスタ）を書き込みます。これは、ファームウェアが bMSEBIM_RDMAE1 ビットに“1”をセット（立ち上がり）したとき、MSEBI バスから DMA 受信 FIFO へ DMA 転送を開始するため、DMA コントローラによって使用される先頭ブロックアドレスです。
- 受信ウォーターマークレベル（bMSEBIM_DMARDLR ビット）を設定します。本ビットフィールドは、DMA バースト要求が受信ロジックによって生成されるときのレベルを制御します。
- rMSEBIM_RDMACR_CS[n]_N レジスタの bMSEBIM_RDMAE1 ビットフィールドに 1 を書き込みます。この時点で、DMA 受信チャネルは、すべてのデータ（SRC_BLOCK_SIZE 分のシングルトランザクション）が転送されるまで実行中となります。すべてのデータが転送されて、bMSEBIM_RDMAE1 ビットが 0 にリセットされると、DMA コントローラが停止して、DMA 受信 FIFO がフラッシュされます。

備　考

bMSEBIM_RDMAE1 ビットの立ち上がりは、DMA 受信 FIFO をフラッシュします。

bMSEBIM_USE_EXT_RDDMA_REQ ビットに 1 を書いて、外部 DMA 要求を許可した場合：

bMSEBIM_RDMAE1 ビットが 0 から 1（DMA 開始）になると、MSEBI コントローラは、MSEBIM_DMA_RD0_N または MSEBIM_DMA_RD1_N（使用する CS_N に依存）が 0 にリセットされるのを待ちます。

このとき、データが MSEBI バスから入力され、MSEBI コントローラは DMA 受信 FIFO にデータを書き込みます。

DMA 受信 FIFO 内のエントリ数がウォーターマークレベル（bMSEBIM_DMARDLR ビット値＋1）以上であると、DMA は、DMA 受信 FIFO バッファからバーストデータを読み出すことで応答します。

bMSEBIM_USE_EXT_RDDMA_REQ ビットに 0 を書いて、外部 DMA 要求を禁止した場合：

同じように機能しますが、外部 DMA 要求を使用しません（MSEBIM_DMA_RD0_N または MSEBIM_DMA_RD1_N が 0 にリセットされます）。

DMA 転送は即座に開始されます。

(1) DMA 動作の概要

DMA コントローラは、周辺機能フローコントローラモードに設定する必要があります。MSEBI がフローコントローラであり、転送されるブロックサイズを識別する必要があるからです（詳細については、下記を参照してください。ここで、n＝0、1 です）。

プロセッサによって MSEBI のプログラミングを行い、送受信すべきデータ項目数（ブロックサイズ）を設定する必要があります。そのためには、DMA 送信 FIFO と DMA 受信 FIFO のそれぞれに対して、MSEBI の rMSEBIM_TDMACR_CS[n]_N レジスタの bMSEBIM_DEST_BLOCK_SIZE フィールド、および rMSEBIM_RDMACR_CS[n]_N レジスタの bMSEBIM_SRC_BLOCK_SIZE フィールドを設定します。ブロックは複数のトランザクションに分割され、それぞれのトランザクションは MSEBI からの要求で開始されます。

さらに、DMA コントローラと MSEBI のプログラミングを行い、DMA 要求ごとにバーストによって転送すべきシングルトランザクション数も設定する必要があります。これは、バーストトランザクション長とも呼ばれています。プログラミングは下記フィールドに対して行います。

- DMA コントローラ（ch＝0～7）：　送信 FIFO と受信 FIFO のそれぞれに対して、DMAC.CTL[ch]レジスタの DEST_MSIZE フィールドおよび SRC_MSIZE フィールド
- MSEBI：　DMA 送信 FIFO と DMA 受信 FIFO のそれぞれに対して、MSEBI の rMSEBIM_TDMACR_CS[n]_N レジスタの bMSEBIM_DEST_BURST_SIZE フィールド、および rMSEBIM_RDMACR_CS[n]_N レジスタの bMSEBIM_SRC_BURST_SIZE フィールド

DMA コントローラで管理される AHB バスのサイズを必ず考慮に入れてください。これは、DMA 送信 FIFO と DMA 受信 FIFO のそれぞれに対して、DMAC.CTL[ch]レジスタの DST_TR_WIDTH フィールドおよび SRC_TR_WIDTH フィールドを用いて設定できます。

MSEBI は 64 ビット幅をサポートしています（bMSEBIM_SINGLE_DEST_WIDTH および bMSEBIM_SINGLE_SRC_WIDTH ビットを参照してください）。

バーストモードとシングルモードで、すべてのトランザクションを正しく管理するための推奨値：

- AHB バスのサイズ：64 ビット
- バーストラインのサイズ：4×64 ビット

注　意

バーストトランザクションサイズは、DMAC と MSEBI で同じ値にする必要があります。

(2) 外部 DMA 要求

DMA 転送フレームを開始するために、4 本の入力が利用可能です。

- MSEBIM_DMA_RD0_N：CS0_N に対する受信フレームの開始および停止
- MSEBIM_DMA_RD1_N：CS1_N に対する受信フレームの開始および停止
- MSEBIM_DMA_WR0_N：CS0_N に対する送信フレームの開始および停止
- MSEBIM_DMA_WR1_N：CS1_N に対する送信フレームの開始および停止

すべての外部要求は非同期式です。

使用前（n＝0、1）：

下記レジスタを用いて、DMA 読み出しのためそれぞれの外部要求（MSEBIM_DMA_RD[n]_N）を許可する必要があります。

- rMSEBIM_DMARDLR_CS[n]_N レジスタの bMSEBIM_USE_EXT_RDDMA_REQ ビット
- rMSEBIM_RDMACR_CS[n]_N レジスタの bMSEBIM_RDMAE1 ビット

チャネルを有効にした後、MSEBIM_DMA_RD[n]_N にレベル“0”が検出されると、MSEBI バスから DMA 受信 FIFO へ DMA 読み出しサイクルが開始されます。

DMA 転送中に、MSEBIM_DMA_RD[n]_N が“1”になると、DMA 読み出しサイクルが中断します。これは、DMA は DMA 受信 FIFO からデータをさらに読み出すことはできても、読み出し要求は MSEBI バスに送信されないことを意味しています。

転送の中断中に、MSEBIM_DMA_RD[n]_N が“0”にリセットされると、転送が再開されて、次の読み出しコマンドが MSEBI バスに送信されます。

DMA 受信 FIFO において最後の転送が完了すると（SRC_BLOCK_SIZE 分のデータ項目が DMA 受信 FIFO 内に読み出されると）、bMSEBIM_RDMAE1 ビットがハードウェアによって自動的にクリアされて、受信モードでの DMA が禁止されます。そのため、ソフトウェアで本ビットをポーリングすれば、いつ当該チャネルが解放されて次の DMA 転送が可能になるかを判定できます。

下記レジスタを用いて、DMA 書き込みのためそれぞれの外部要求（MSEBIM_DMA_WR[n]_N）を許可する必要があります。

- rMSEBIM_DMATDLR_CS[n]_N レジスタの bMSEBIM_USE_EXT_WRDMA_REQ ビット
- rMSEBIM_TDMACR_CS[n]_N レジスタの bMSEBIM_TDMAE1 ビット

チャネルを有効にした後、MSEBIM_DMA_WR[n]_N にレベル“0”が検出されると、DMA 送信 FIFO から DMA コントローラへ DMA 書き込み要求サイクルが開始されます。

DMA 転送中に、MSEBIM_DMA_WR[n]_N が“1”になると、DMA 書き込みサイクルが中断します。これは、DMA は DMA 送信 FIFO にデータをさらに書き込むことはできても、書き込み要求は MSEBI バスに送信されないことを意味しています。

転送の中断中に、MSEBIM_DMA_WR[n]_N が“0”にリセットされると、転送が再開されて、次の書き込みコマンドが MSEBI バスに送信されます。

DMA 送信 FIFO において最後の転送が完了すると（DEST_BLOCK_SIZE 分のデータ項目が DMA 送信 FIFO 内に読み出されると）、bMSEBIM_TDMAE1 ビットがハードウェアによって自動的にクリアされて、送信モードでの DMA が禁止されます。そのため、ソフトウェアで本ビットをポーリングすれば、いつ当該チャネルが解放されて次の DMA 転送が可能になるかを判定できます。

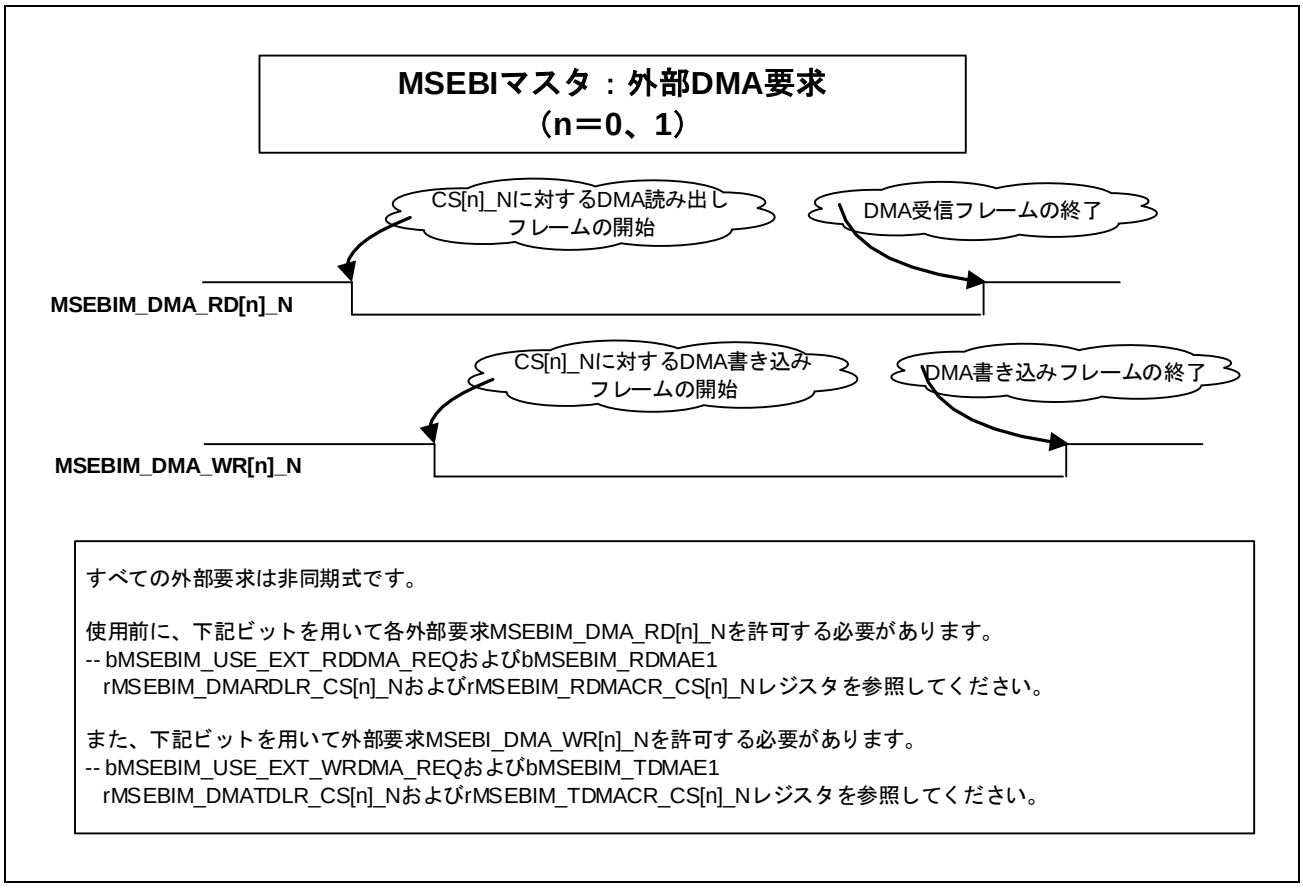


図 10.53　MSEBI マスタ：外部 DMA 要求

(3) 送信ウォーターマークレベルと送信 FIFO アンダーフロー

MSEBI バスが転送を実行しているとき、送信 FIFO 内のエントリ数が DMA 送信データレベルレジスタの bMSEBIM_DMATDLR ビット値以下になると、ただちに DMA コントローラに対して送信 FIFO 要求が出されます。DMA は、DMAC.CTL[ch].DEST_MSIZE＝DEST_BURST_SIZE の長さのバーストデータを送信 FIFO バッファに書き込むことで応答します。送信 FIFO がシリアル転送を連続して実行できるだけの十分な頻度で、データが DMA からフェッチされる必要があります。すなわち、FIFO がエンプティになりそうであれば、次の DMA 要求がトリガされなければいけません。次の DMA 要求がトリガされないと、FIFO はデータ切れ（アンダーフロー）になります。これを防止するため、ウォーターマークレベルを正確に設定する必要があります。

(4) 送信ウォーターマークレベルの選択

次式が仮定される場合の例について考えてみましょう。

DEST_BURST_SIZE＝DMAC.CTL[ch].DEST_MSIZE＝FIFO_DEPTH－bMSEBIM_DMATDLR

ここでは、DMA バーストで転送されるデータ項目数が、送信 FIFO 内の空きスペースに一致しています。以下では、2 つの異なるウォーターマークレベルの設定値について検討してみます。

下図において、必要とされるバーストトランザクション数は、次式のように、ブロックサイズをバーストあたりのデータ項目数で割った値です。

DEST_BLOCK_SIZE / DEST_BURST_SIZE＝120/30＝4

DMA ブロック転送におけるバーストトランザクション数は 4 になります。しかし、ウォーターマークレベル（bMSEBIM_DMATDLR ビット値）はかなり低い値です。そのため、MSEBI バスがデータを送信しようとしても、送信 FIFO 内にはデータが残っていない可能性があり、MSEBI がアンダーフローを引き起こす確率は高くなります。すなわち DMA には、送信 FIFO がエンプティとなる前に、DMA 要求に対応する時間的余裕がありません。

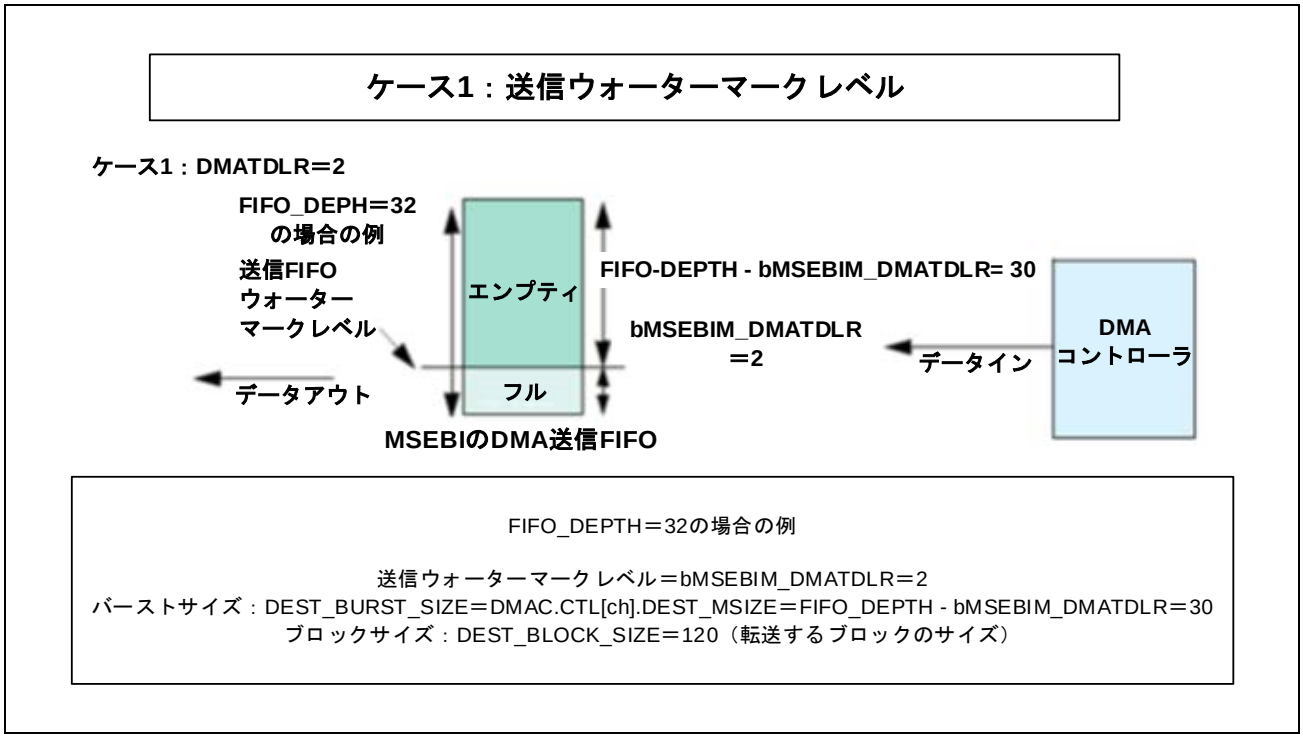


図 10.54　MSEBI ケース 1：送信ウォーターマークレベル

2 番目のケースでは、ブロック転送におけるバーストトランザクション数は以下のようになります。

DEST_BLOCK_SIZE / DEST_BURST_SIZE＝120/2＝60

今回のブロック転送では、DMA ブロック転送における宛先バーストトランザクション数は 60 になります。しかし、ウォーターマークレベル（bMSEBIM_DMATDLR ビット値）は非常に高い値です。DMA コントローラには、送信 FIFO がエンプティとなる前に、宛先バーストトランザクション要求に対応できるだけの十分な時間があるので、MSEBI がアンダーフローを引き起こす確率は低くなります。このように 2 番目のケースでは、ブロックあたりのバーストトランザクション数が増えるかわりに、アンダーフローの発生確率が下がります。一方で、ブロックあたりの要求バースト量が増える可能性があるため、最初のケースよりもバスの利用効率が低下します。

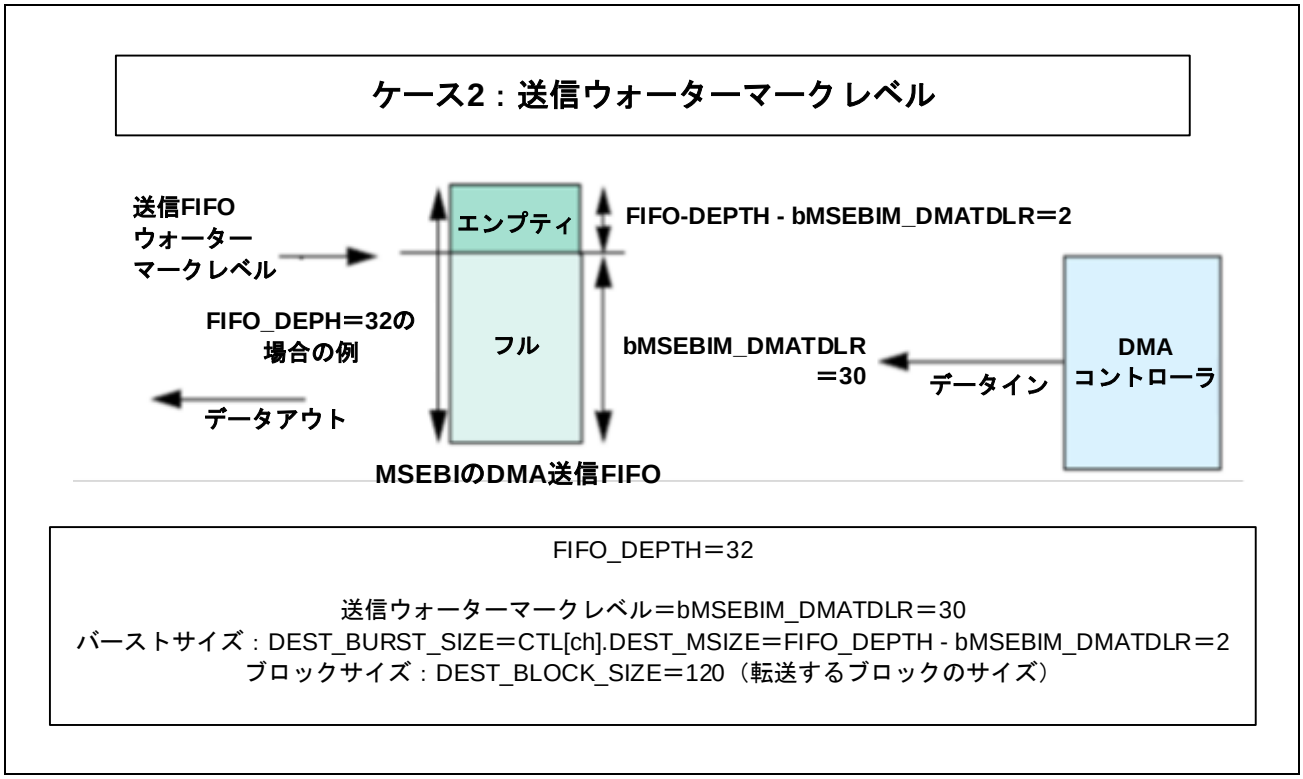


図 10.55　MSEBI ケース 2：送信ウォーターマークレベル

したがって、ウォーターマークレベル選択時の目標は、アンダーフローになる確率を許容できるレベルに維持しつつ、ブロックあたりのトランザクション数を最小限に抑えることにあります。これは実際上、MSEBI がデータを送信する速度と、DMA が宛先バースト要求に応答する速度との比の問題です。

たとえば、DMA において当該チャネルを最優先チャネルとし、さらにバスレイヤにおいて DMA マスタインタフェースを最優先マスタとすれば、DMA コントローラがバーストトランザクション要求に応答できる速度が上がります。結果として、ウォーターマークレベルを下げることが可能となり、アンダーフローの発生確率に妥協することなく、バスの利用効率を改善できるようになります。

(5) DEST_MSIZE の選択と送信 FIFO オーバーフロー

MSEBI の送信 FIFO 内に宛先バースト要求に対応できるだけの十分なスペースがない場合、オーバーフローを引き起こす可能性があります。

したがって、適した動作のためには、以下のように設定する必要があります（ch＝0～7）。

- DMAC.CTL[ch].DST_TR_WIDTH　＝　3（64 ビット）
- DMAC.CTL[ch].DEST_MSIZE　＝　1（シングルトランザクション数 4）
- bMSEBIM_SINGLE_DEST_WIDTH　＝　1（64 ビット）
- bMSEBIM_DEST_BURST_SIZE　＝　1（シングルトランザクション数 4）
- bMSEBIM_DMATDLR　＝　28

(6) 受信ウォーターマークレベルと受信 FIFO オーバーフロー

MSEBI バスが転送を実行しているとき、受信 FIFO 内のエントリ数が DMA 受信データレベル（bMSEBIM_DMARDLR ビット値＋1）以上になると、ただちに DMA コントローラに対して受信 FIFO 要求が出されます。

これをウォーターマークレベルと呼びます。DMA コントローラ（ch＝0～7）は、SRC_BURST_SIZE＝DMAC.CTL[ch].SRC_MSIZE の長さのバーストデータを受信 FIFO バッファに読み出すことで応答します。

受信 FIFO が MSEBI バスの転送を連続して受け入れられるだけの十分な頻度で、データが DMA によってフェッチされる必要があります。すなわち、FIFO がフルになりそうであれば、次の DMA 転送が要求されなければいけません。次の DMA 転送が要求されないと、FIFO はデータ溢れ（オーバーフロー）になります。これを防止するため、ウォーターマークレベルを正確に設定する必要があります。

(7) 受信ウォーターマークレベルの選択

前述の「送信ウォーターマークレベルの選択」の場合と同様に、受信ウォーターマークレベル（bMSEBIM_DMARDLR ビット値＋1）に対しても、下図に示すように、オーバーフローの発生確率を最小限に抑えるように設定する必要があります。これは、必要とされるブロックあたりの DMA バーストトランザクション数と、オーバーフローの発生確率とのトレードオフになります。

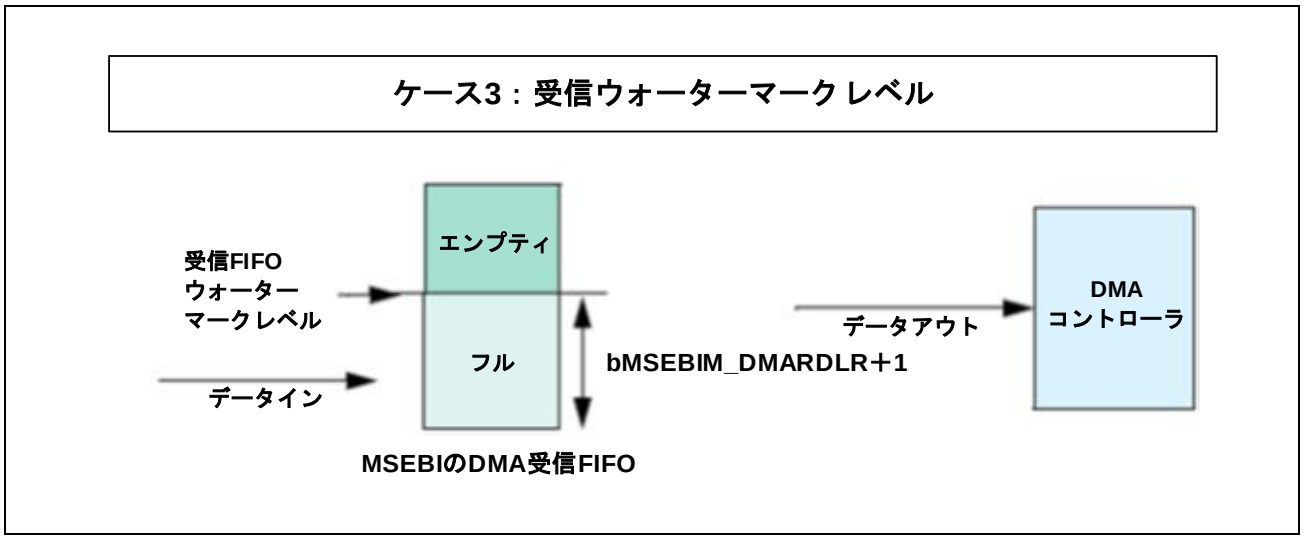


図 10.56　MSEBI ケース 3：受信ウォーターマークレベル

(8) SRC_MSIZE の選択と受信 FIFO アンダーフロー

送信元バースト要求に対応できるだけの十分なデータが MSEBI の受信 FIFO 内に存在していない場合、アンダーフローを引き起こす可能性があります。

したがって、適した動作のためには、以下のように設定する必要があります（ch＝0～7）。

- DMAC.CTL[ch].SRC_TR_WIDTH = 3（64 ビット）
- DMAC.CTL[ch].SRC_MSIZE = 1（シングルトランザクション数 4）
- bMSEBIM_SINGLE_SRC_WIDTH = 1（64 ビット）
- bMSEBIM_SRC_BURST_SIZE = 1（シングルトランザクション数 4）
- bMSEBIM_DMARDLR = 3

10.4.7　MSEBI スレーブモード

10.4.7.1　スレーブモードの概要

スレーブモードの主な特長は以下のとおりです。

- 同期式モードのみ
- 8 ビット、16 ビット、および 32 ビットから選択可能な MSEBI インタフェース幅
- AHB 上での 1-4-8-16 バーストアクセスのサポート
- 4 本の DMA フロー制御信号（外部要求送信）使用可能
- 1 本の割り込みライン（「ブロック終了」検出用）
- 2 種類のアドレス指定モード
  - ダイレクトモード
  - MMU モード
- デバイスへの書き込みを防止する書き込み保護ビット
- MSEBI からの要求を管理する 6 つの 32 ビット FIFO
- CPU 送信 FIFO と DMA RX FIFO 用のプリフェッチ機能
- DMA TX FIFO 用に最適化されたバースト機能
- 最大 4 本のチップセレクトライン
- バスのマスタからアクセスされる 6 つの共有レジスタ
  - エラーを管理するレジスタが 1 つ
  - ブロック終了割り込みを管理するレジスタが 1 つ
  - チップセレクトごとに ID レジスタが 1 つ
    チップセレクトが利用可能かどうかを判定するためにマスタが使用

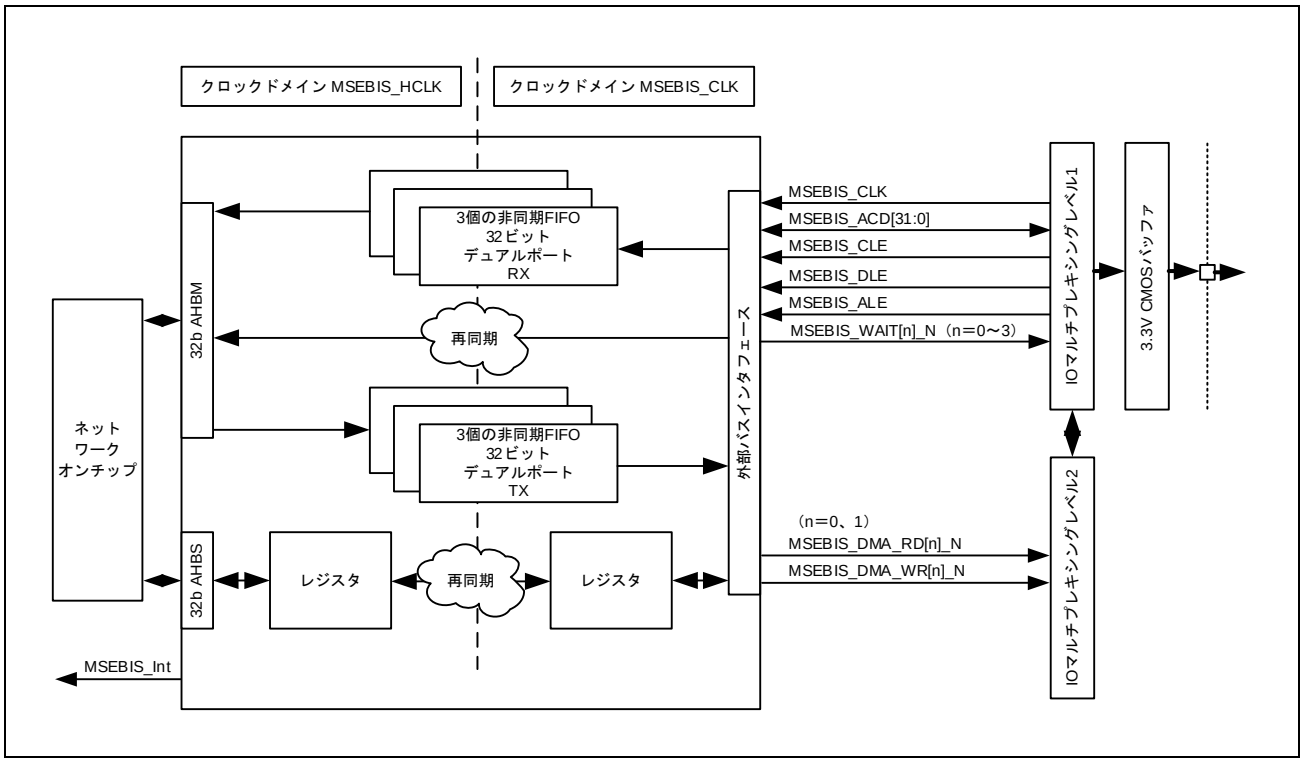


図 10.57　　スレーブモードの概要

10.4.7.2　MSEBI スレーブ：バーストモード

AHB バスに対するバーストアクセスを管理するため、MSEBI スレーブインタフェースは以下の一連のルールを使用します。

- 各 MSEBI_CS[n]_N（n＝0～3）に対してバーストモードを許可（bMSEBIS_BURST_ENABLE ビットを 1’b1 にセット）
- バーストサイクルを生成できるのは、下記の 5 つのリクエスタです。
  - CPU 受信 FIFO
    MSEBI バスのマスタ CPU 部からのコマンドを保持
  - CS0_N に対する DMA 送信 FIFO
    バスのマスタ DMA TX 0 部からのコマンドを保持
  - CS1_N に対する DMA 送信 FIFO
    バスのマスタ DMA TX 1 部からのコマンドを保持
  - CS0_N に対する DMA 受信 FIFO
    バスのマスタ DMA RX 0 部からのコマンドを管理
  - CS1_N に対する DMA 受信 FIFO
    バスのマスタ DMA RX 1 部からのコマンドを管理
- 要求のイニシエータの検出には特殊なロジックが用いられます。「**10.4.7.3 MSEBI スレーブ：要求イニシエータの検出**」を参照してください。
- 各要求間では、アービタがラウンドロビン順位を管理します。
- プリフェッチモードは、下記条件のすべてが満たされた場合にのみ利用可能です。
  - 各 MSEBI_CS[n]_N（n＝0～3）に対して、bMSEBIS_BURST_ENABLE ビットでバーストモードを許可
  - CPU モードでは、bMSEBIS_BURST_SIZEMAX_CPUREAD ビットを 1 より大きな値に設定
  - DMA モードでは、bMSEBIS_DMARX_MAX_BURST ビットを 1 より大きな値に設定
- 最大バーストサイズは下記ビットで設定します。
  - CPU：　bMSEBIS_BURST_SIZEMAX_CPUWRITE ビット、および
    bMSEBIS_BURST_SIZEMAX_CPUREAD ビット

    CPU プリフェッチ：　bMSEBIS_BURST_SIZEMAX_CPUREAD ビットは、プリフェッチ動作中に読み出される最大ワード数を制御するためにも使用されます。
  - DMA：rMSEBIS_DMARDLR_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMARX_MAX_BURST ビット、および rMSEBIS_DMATDLR_CS[n]_N（n＝0、1）レジスタの
    bMSEBIS_DMATX_MAX_BURST ビット

    DMA プリフェッチ：　rMSEBIS_DMARDLR_CS[n]_N（n＝0、1）レジスタの
    bMSEBIS_DMARX_MAX_BURST ビットは、プリフェッチ動作中に読み出される最大ワード数を制御するためにも使用されます。
- MSEBI は書き込みバーストのみを生成します。
- FIFO サイズ
  - CPU：送受信 FIFO に対して 32 ワード×32 ビット
  - DMA：MSEBI_CS[n]_N（n＝0、1）ごとに、送受信 FIFO に対して 32 ワード×32 ビット
- AHB バーストアクセスは 1kB 境界を超えません。

(1) スレーブの CPU FIFO

受信 FIFO：

- ID＝CPU の場合、MSEBI バスからのすべてのトランザクションは、受信 FIFO に格納されます。
- トランザクションの順序は厳格に守られます。
- MSEBI スレーブコントローラは、常に（FIFO の内容に応じて）可能な限り大きなサイズのバーストを AHB バス上に生成しようと試みます（ただし、bMSEBIS_BURST_SIZEMAX_CPUWRITE ビットの最大許容バーストサイズで制限されます）。
  - MSEBI スレーブコントローラはバーストのみを生成します。
  - アドレスは、リニアに 4 ずつインクリメントされ、決して 1kB 境界を超えません。
- MSEBI_CS[n]_N（n＝0～3）に対するアクセスは、専用の bMSEBIS_CS_ENABLE ビットがセットされている場合にのみ許可されます。ビットがクリアされていると、MSEBI_CS[n]_N に対するアクセスは無視されます。

送信 FIFO：

- 読み出し要求からのデータは送信 FIFO に格納されます。
- MSEBI バスからの読み出しコマンドに対して、プリフェッチが許可されている場合、かつ現在の読み出し要求後に FIFO がエンプティであれば、バスインタフェースは MSEBI の読み出し要求が継続することを期待して、プリフェッチモードでバーストアクセスを生成します（ただし、bMSEBIS_BURST_SIZEMAX_CPUREAD ビット値で制限されます）。
  - MSEBI スレーブコントローラはプリフェッチ動作のみを生成します。
  - アドレスは、リニアに 4 ずつインクリメントされ、決して 1kB 境界を超えません。
- 読み出し要求が非インクリメンタルアドレスを付加したとき、FIFO の内容がフラッシュされます。
- MSEBI_CS[n]_N（n＝0～3）に対するアクセスは、専用の bMSEBIS_CS_ENABLE ビットがセットされている場合にのみ許可されます。ビットがクリアされていると、MSEBI_CS[n]_N に対するアクセスは無視されます。

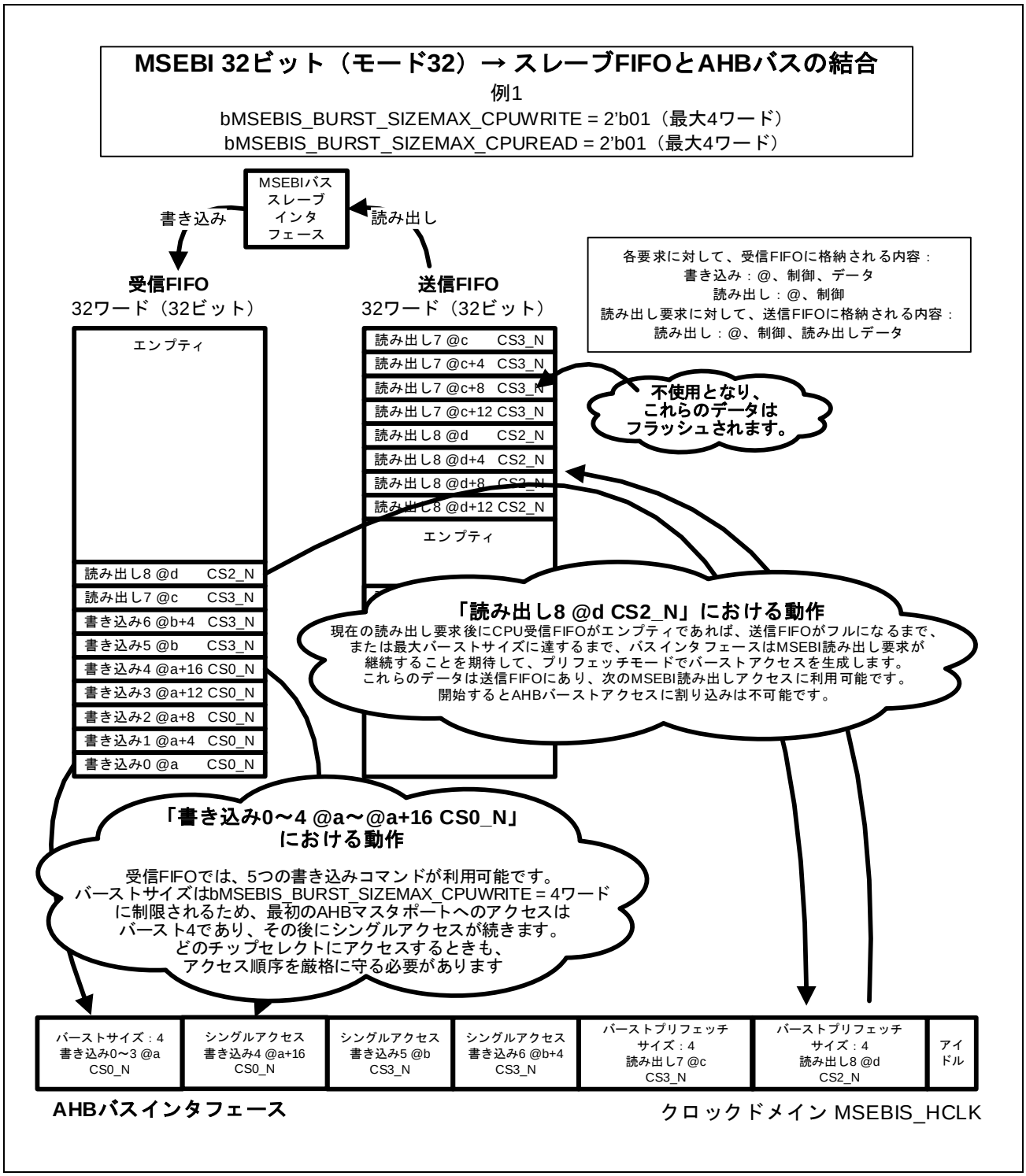


図 10.58　　MSEBI スレーブの CPU FIFO の例 1

備　考

上図において、すべてのアクセスは 32 ビットにアラインされるものとします。

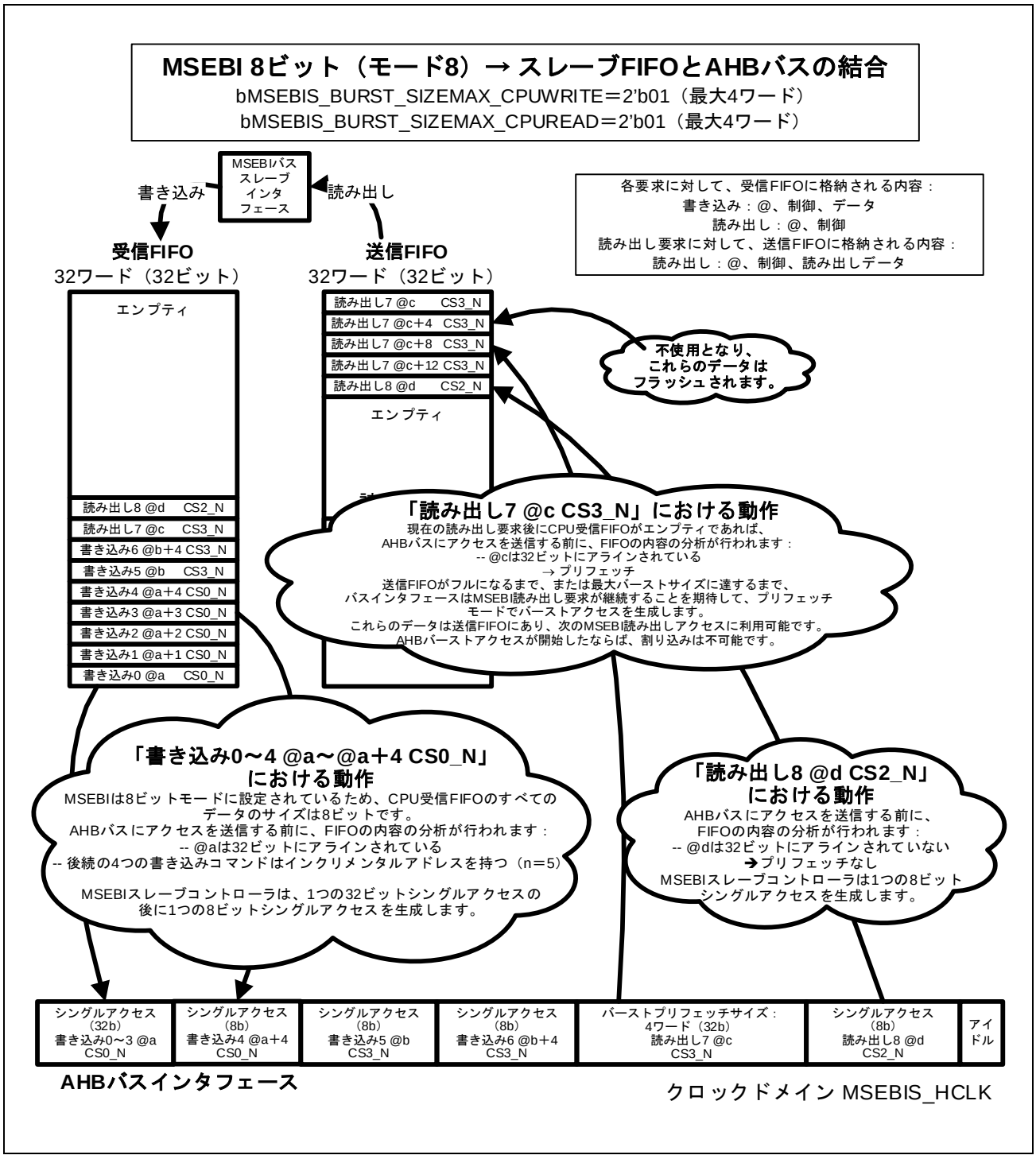


図 10.59　　MSEBI スレーブの CPU FIFO の例 1、8 ビットモード

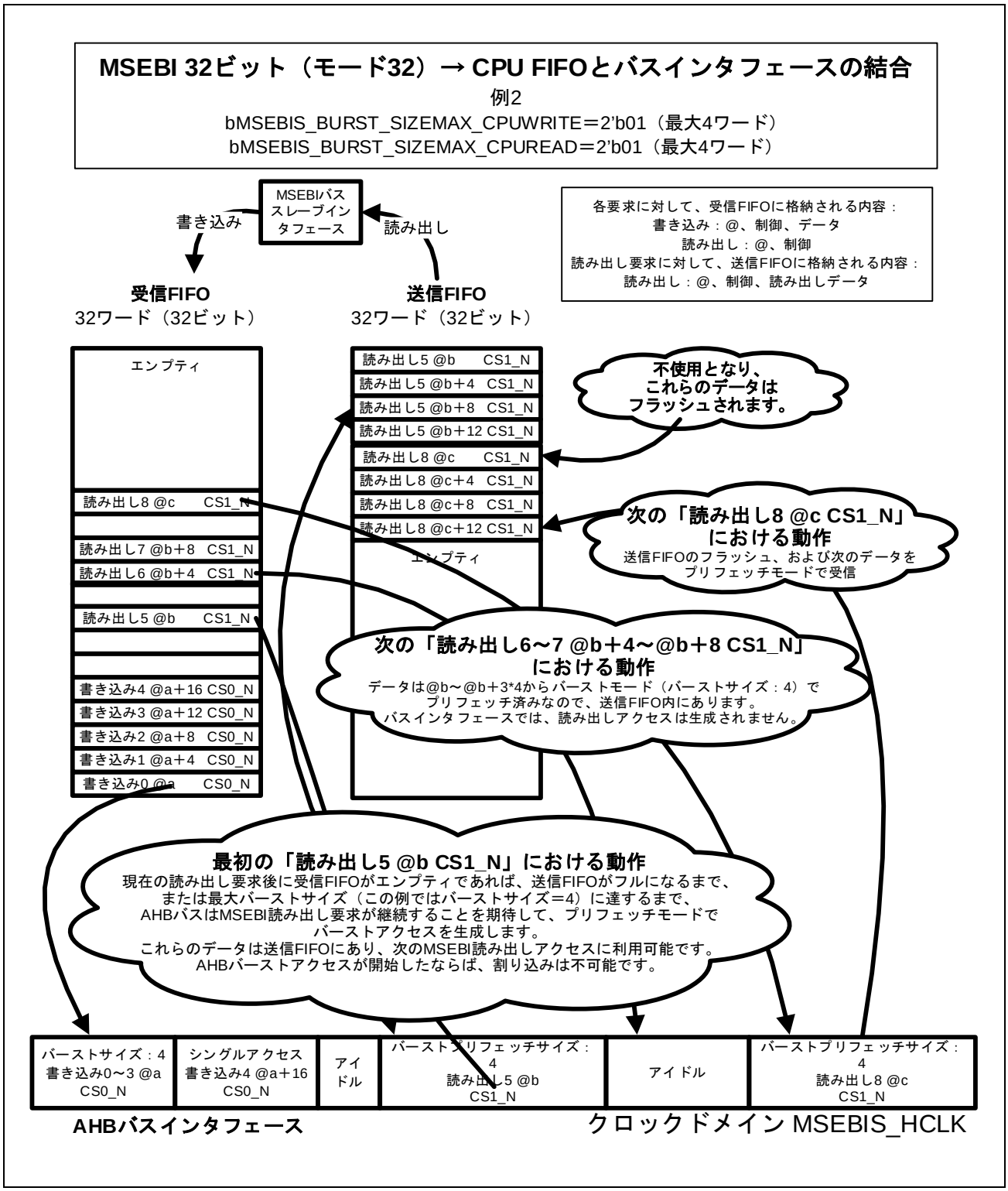


図 10.60　MSEBI スレーブの CPU FIFO の例 2

備　考

上図において、すべてのアクセスは 32 ビットにアラインされるものとします。

(2) スレーブの DMA FIFO

(a) MSEBI マスタの DMA TX FIFO からの要求に対するスレーブの DMA FIFO

- ID＝DMA の場合、MSEBI バスからのすべての書き込み要求は DMA 送信 FIFO に格納されます。
- トランザクションの順序は厳格に守られます。
- MSEBI スレーブコントローラは、常に（FIFO の内容に応じて）可能な限り大きなサイズのバーストを AHB バス上に生成しようと試みます（ただし、rMSEBIS_DMATDLR_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMATX_MAX_BURST ビットの最大許容バーストサイズで制限されます）。
  - MSEBI スレーブコントローラはバーストのみを生成します。
  - アドレスは、リニアに 1/2/4 ずつインクリメントされ、決して 1kB 境界を超えません。
- MSEBI は、バーストサイズ（rMSEBIS_DMATDLR_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMATX_MAX_BURST ビット）を送信できるように、十分な数の要求が FIFO に格納されるまで待機することで、バースト要求を最適化することが可能です。このモードは、rMSEBIS_DMATDLR_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMATX_OPT_BURST ビットで選択します。
- rMSEBIS_DMATX_REQ_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMATX_ENABLE ビットが 0 に設定されたとき、FIFO がエンプティになるまでその内容が読み出されます。

注　意

MSEBI スレーブが NoC の占有率を低減するように最適化されている場合（bMSEBIS_DMATX_OPT_BURST＝1）、書き込み要求は DMA TX FIFO でグループ化されます。書き込むブロックのサイズが BURST_SIZE ワード（32 ビット）の倍数でなければ、「ブロック終了イベント」を受信するまで、ブロック転送は完了しません（「ブロック終了イベント」の受信後、FIFO がエンプティになるまでの間、そのチャネル上で最適化は無効とみなされます。「**10.4.5.3 MSEBI 割り込み：スレーブによるブロック終了検出**」を参照してください）。

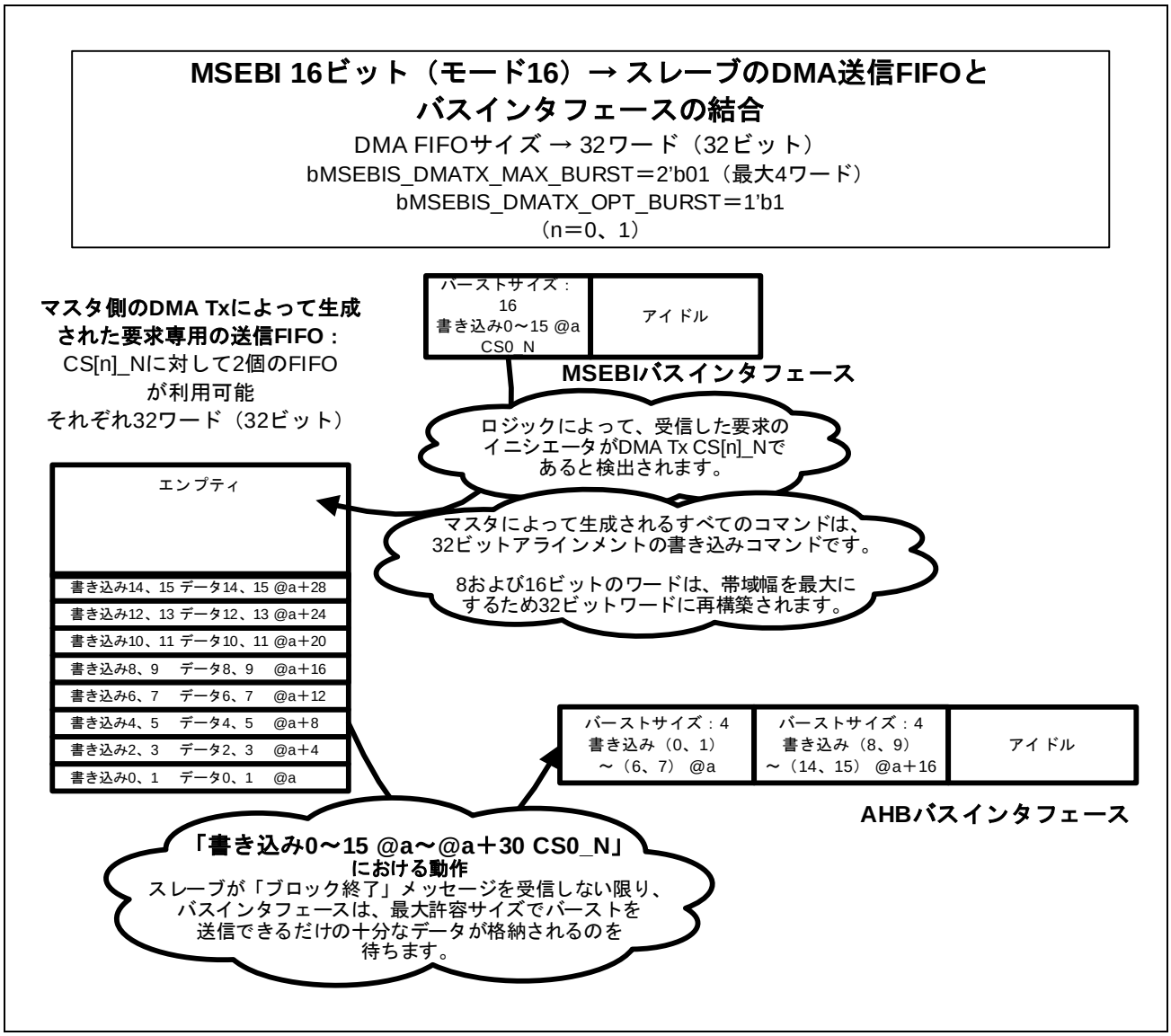


図 10.61　MSEBI マスタの DMA TX FIFO からの要求に対するスレーブの DMA FIFO

(b) MSEBI マスタの DMA RX FIFO からの要求に対するスレーブの DMA FIFO

- ID＝DMA の場合、MSEBI バスからのすべての読み出し要求は DMA 受信 FIFO に格納されます。
- トランザクションの順序は厳格に守られます。
- MSEBI バスからの読み出しコマンドに対して、プリフェッチが許可されている場合、かつ現在の読み出し要求後に FIFO がエンプティであれば、バスインタフェースは本チャネル上で MSEBI の読み出し要求が継続することを期待して、プリフェッチモードでバーストアクセスを生成します（ただし、rMSEBIS_DMARDLR_CS[n]_N（n＝0、1）レジスタの bMSEBIS_BURST_SIZEMAX_CPUREAD ビット値で制限されます）。
  - MSEBI スレーブコントローラはプリフェッチ動作のみを生成します。
  - アドレスは、リニアに 4 ずつインクリメントされ、決して 1kB 境界を超えません。
- プリフェッチ動作の終了後、AHB バスに対して読み出しバーストを生成できるだけの十分なスペースが FIFO 内に存在すれば（FIFO がエンプティのときは、2 つのプリフェッチ動作間で MSEBI バスのウェイトサイクルを避けるために）、MSEBI スレーブコントローラはただちにプリフェッチモードで次のバーストを送信します。このタイプのプリフェッチは、下記条件のすべてが満たされたときに生成されます。
  - 最初のプリフェッチ動作が発生済みの場合
  - 最後のプリフェッチ動作以降に、FIFO がフラッシュされていない場合
  - 少なくとも FIFO 内に、rMSEBIS_DMARDLR_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMARX_MAX_BURST ビットで指定されたサイズ分のスペースが存在する場合
- 読み出し要求が非インクリメンタルアドレスを付加したとき、または rMSEBIS_DMARX_REQ_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMARX_ENABLE ビットが 0 に設定されたとき、FIFO の内容がフラッシュされます。

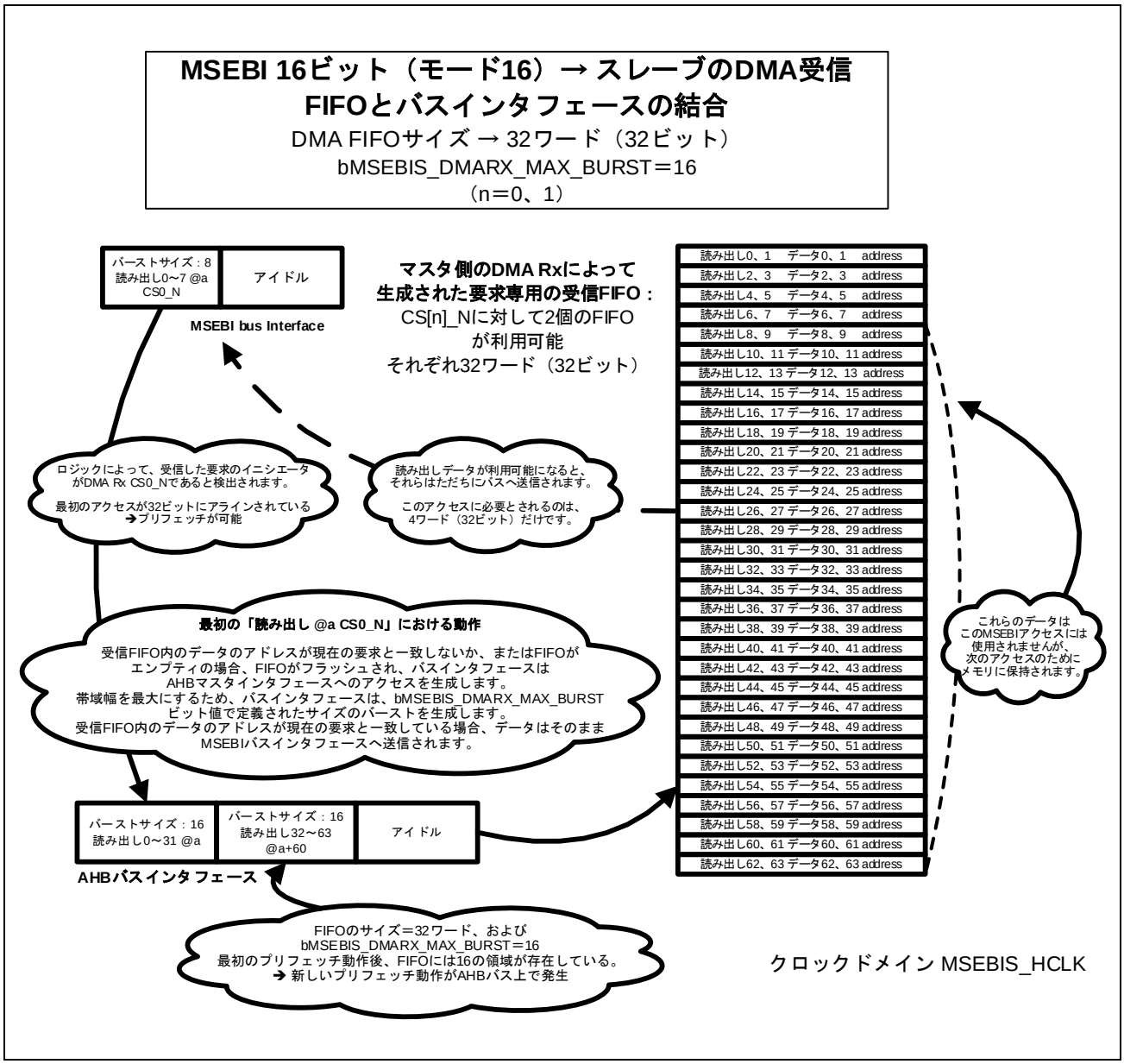


図 10.62　　MSEBI マスタの DMA RX FIFO からの要求に対するスレーブの DMA FIFO

(3) スレーブの DMA フロー制御信号

MSEBI スレーブコントローラは、下記の DMA フロー制御信号を管理できます。

- MSEBIS_DMA_RD0_N
- MSEBIS_DMA_RD1_N
- MSEBIS_DMA_WR0_N
- MSEBIS_DMA_WR1_N

MSEBIS_DMA_RD0_N 端子と MSEBIS_DMA_RD1_N 端子のフロー制御は、rMSEBIS_DMARDLR_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMARX_FLOW_CTRL ビットフィールドで有効にできます。

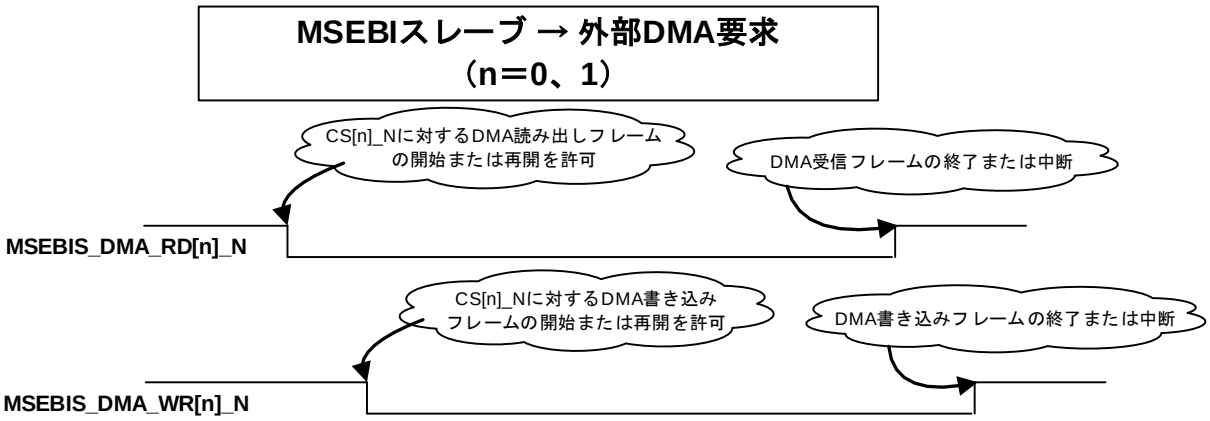
MSEBIM_DMA_WR0_N 端子と MSEBIM_DMA_WR1_N 端子のフロー制御は、rMSEBIS_DMATDLR_CS[n]_N（n＝0、1）レジスタの bMSEBIS_DMATX_FLOW_CTRL ビットフィールドで有効にできます。

MSEBI コントローラは、DMA FIFO ごとに 1 つの DMA フロー制御信号を供給します。

フロー制御を有効にすると、MSEBIS_DMA_RD[n]_N 信号と MSEBIS_DMA_WR[n]_N 信号は、rMSEBIS_DMARX_REQ_CS[n]_N レジスタと rMSEBIS_DMATX_REQ_CS[n]_N レジスタの下記ビットで駆動されます。

- bMSEBIS_DMARX_FORCE
- bMSEBIS_DMATX_FORCE

フロー制御信号の動作についての詳細は、マスタ部の「**10.4.6.3(2) 外部 DMA 要求**」の説明を参照してください。



MSEBIスレーブコントローラの使用前に、rMSEBIS_DMARDLR_CS[n]_Nレジスタの下記ビットで外部要求MSEBIS_DMA_RD[n]_Nを設定する必要があります。
-- bMSEBIS_DMARX_ENABLEを1にセット
-- bMSEBIS_DMARX_FLOW_CTRLを1にセット

また、rMSEBIS_DMATDLR_CS[n]_Nレジスタの下記ビットで外部要求MSEBIS_DMA_WR[n]_Nを設定する必要があります。
-- bMSEBIS_DMATX_ENABLEを1にセット
-- bMSEBIS_DMATX_FLOW_CTRLを1にセット

許可されている場合、MSEBIS_DMA_RD[n]_N/MSEBIS_DMA_WR[n]_Nは下記ビットで駆動されます。
-- bMSEBIS_DMARX_FORCE
-- bMSEBIS_DMATX_FORCE

クロックドメイン MSEBIS_HCLK

図 10.63　MSEBI スレーブ：外部 DMA 要求

(4) スレーブ FIFO アービタ：ラウンドロビン

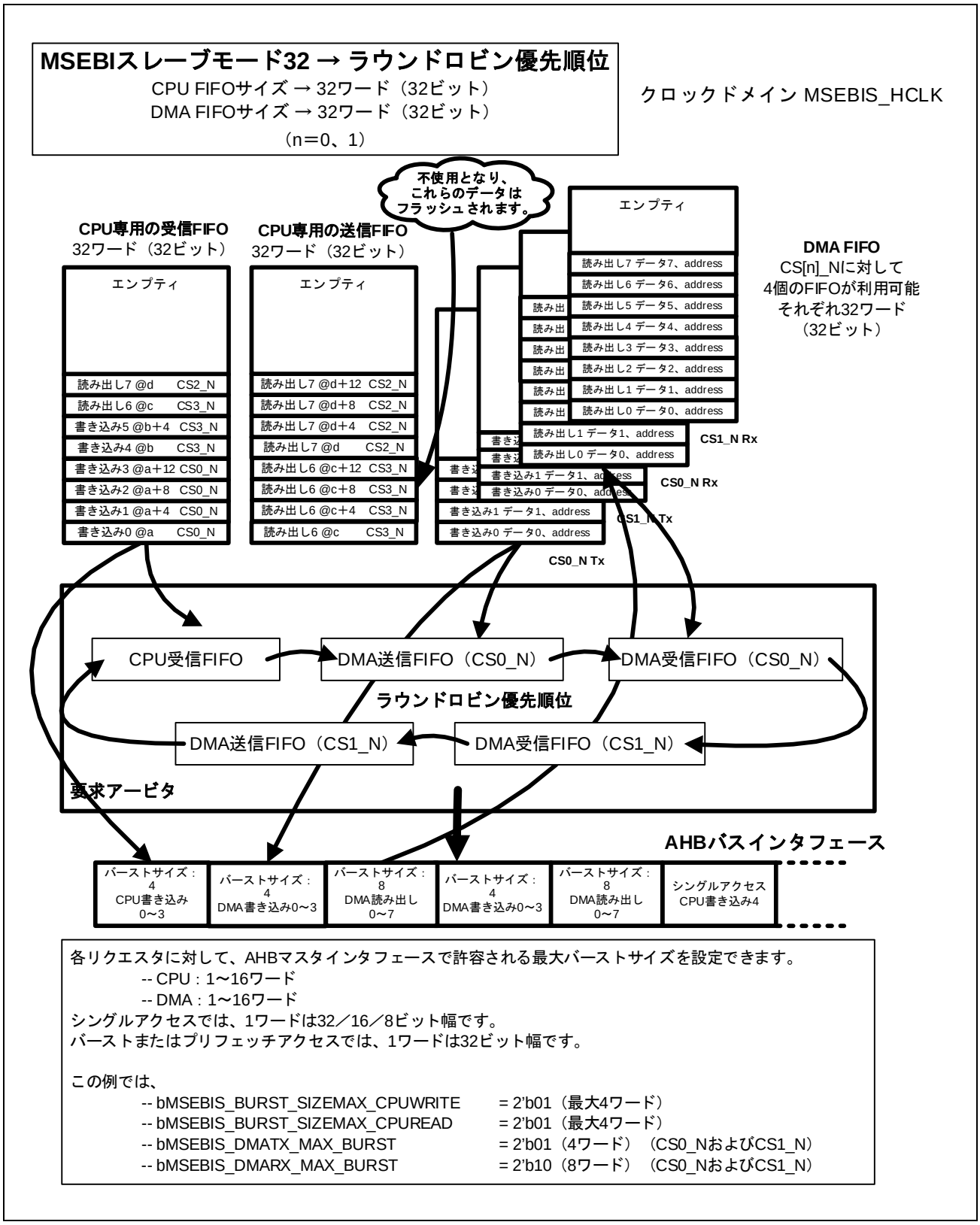


図 10.64　スレーブ FIFO アービタ：ラウンドロビン

10.4.7.3　MSEBI スレーブ：要求イニシエータの検出

要求のイニシエータは、CLE フェーズ中に MSEBI_DMA_N ビットによって検出されます。

本情報は、受信した要求を的確な FIFO へルーティングして、転送を最適化するために使用されます（同じイニシエータからの要求をグループ化することにより、バーストの発生確率が高まります）。

以下の表を参照してください。

- 「**表 10.4 MSEBI モード 32、ACD31～0 に対するマルチプレクサ機能**」
- 「**表 10.6 MSEBI モード 16、ACD15～0 に対するマルチプレクサ機能**」
- 「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」

表 10.47　スレーブの要求イニシエータの検出

イニシエータ	MSEBI_DMA_N	MSEBI_CS1_N, MSEBI_CS0_N	要求タイプ	ターゲット FIFO
CPU	1	X, X	X	CPU 受信
DMA RX CS0	0	1, 0	読み出し	DMA RX CS0
DMA TX CS0	0	1, 0	書き込み	DMA TX CS0
DMA RX CS1	0	0, 1	読み出し	DMA RX CS1
DMA TX CS1	0	0, 1	書き込み	DMA TX CS1

10.4.7.4　MSEBI スレーブ：マスタによるレジスタアクセス

バス上の MSEBI マスタは、6 つのスレーブ共有レジスタにアクセスできます。各レジスタのアドレスに従い、MSEBI_CS[n]_N（n＝0～3）に対応した MSEBI アクセスが生成されます。

- rMSEBIS_INT
- rMSEBIS_STATUS
- rMSEBIS_ID_CS[n]_N（n＝0～3）

MSEBI_CS[n]_N がアクティブなチップセレクトであるとき、スレーブの共有レジスタへアクセスできるのは、下記条件が満たされた場合です。

- チップセレクト[n]へのアクセスの MSEBI_CLE フェーズ中に、コマンドによって MSEBI_CSREG_N と MSEBI_CS[n]_N が同時に 0 に設定された場合
- 下記の設定によって、チップセレクトがアクティブなデバイス上で要求が管理される場合
  - bMSEBIS_CS_ENABLE＝1

下表は、スレーブの共有レジスタにアクセスするための、MSEBI_CSREG_N と MSEBI_CS[n]_N の条件を説明しています。

表 10.48　MSEBI スレーブの共有レジスタへのアクセス

MSEBI_CS[n]_N	MSEBI_CSREG_N	CS[n]_N アクセスタイプ
0	0	CS[n]_N の共有レジスタにアクセス
0	1	CS[n]_N のメモリ空間にアクセス
1	0	予約
1	1	アクセスなし

以下の表を参照してください。

- 「**表 10.4 MSEBI モード 32、ACD31～0 に対するマルチプレクサ機能**」
- 「**表 10.6 MSEBI モード 16、ACD15～0 に対するマルチプレクサ機能**」
- 「**表 10.8 MSEBI モード 8、ACD7～0 に対するマルチプレクサ機能**」

10.4.7.5　MSEBI スレーブ：チップセレクトの設定状態

MSEBI スレーブコントローラのすべてのチップセレクトは、その設定状態を MSEBI バスのマスタに伝達できます（n＝0～3）。

- 本機能の主な目的は、MSEBI バスの設定を変更した後、その（スレーブデバイス上の）チップセレクトが MSEBI バスからの要求を受信できる状態にあると MSEBI バスのマスタに通知することです。
- 設定状態は、MSEBI バスを通じて、スレーブの共有レジスタ（rMSEBIS_ID_CS[n]_N）を読み出すことにより入手できます。
  - スレーブの共有レジスタについての詳細は、「**10.4.7.4 MSEBI スレーブ：マスタによるレジスタアクセス**」を参照してください。
- MSEBI バスのマスタは、すべてのチップセレクトが下記の状態であれば、スレーブの共有レジスタにアクセスできます。
  - MSEBI_CSREG_N と MSEBI_CS[n]_N が 1’b0 になっている場合
  - 「**10.4.7.4 MSEBI スレーブ：マスタによるレジスタアクセス**」を参照してください。
- 対応するスレーブの共有レジスタから正しい値を読み出すことにより、チップセレクトの設定の有効性を確認できます。
  - rMSEBIS_ID_CS[n]_N レジスタから 0x1234_FEDn が読み出された場合、そのチップセレクト[n]の設定は有効です。
  - その他の値は、そのチップセレクト[n]が通信可能な状態にないことを意味します。

10.4.7.6　MSEBI スレーブ：アドレス指定モード

MSEBI スレーブは、新たに受信したトランザクションのアドレスを、以下の 2 種類の方法で管理できます。

- ダイレクトアクセス：スレーブによってデコードされたアドレスをそのまま使用して、AHB マスタポートに新たなトランザクションを発生させます。
- MMU モードアクセス：スレーブによってデコードされたアドレスにベースオフセット（4KB にアラインされた値）を加算してから、AHB マスタポートに新たなトランザクションを発生させます。

以下のレジスタを参照してください。

- 「**10.3.3.4 rMSEBIS_MMU_ADDR_MASK_CS[n]_N — MMU アドレスマスクレジスタ（n＝0～3）**」、および「**10.3.3.3 rMSEBIS_MMU_ADDR_CS[n]_N — MMU ベースアドレスレジスタ（n＝0～3）**」

MMU モードは一部の ALE フェーズを節減できる可能性があるため、パフォーマンスが向上します。

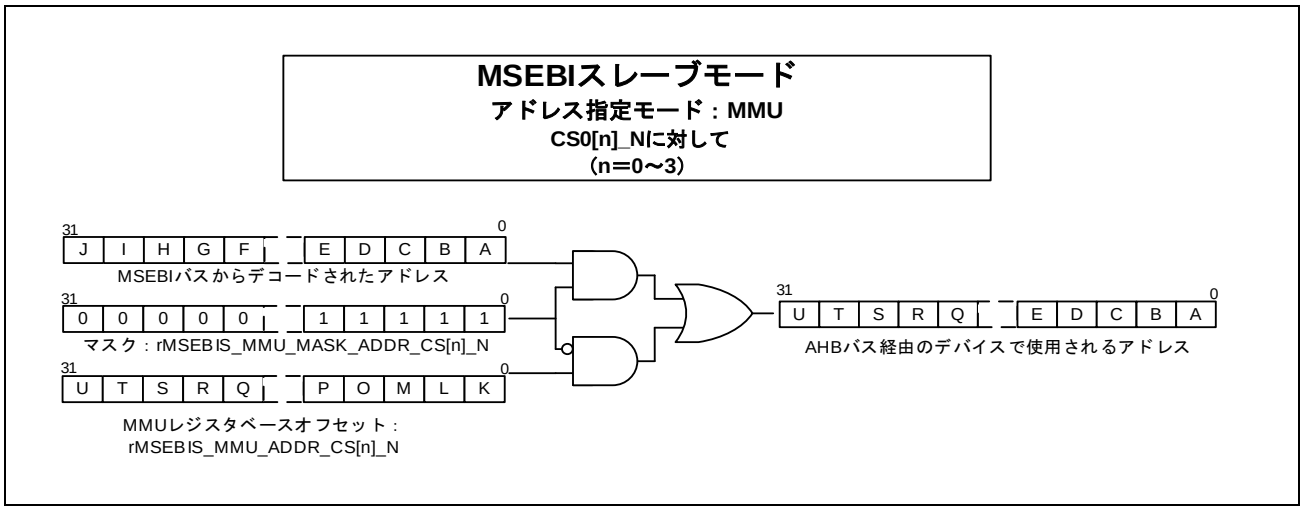


図 10.65　スレーブの MMU モードのアドレス指定

アドレス指定モードの選択は、bMSEBIS_ADDR_MODE ビットで行います。

10.4.7.7　MSEBI スレーブ：書き込み保護

書き込み保護ビットを設定すると、スレーブのメモリ空間への書き込みはすべて禁止されます。

書き込み保護は、bMSEBIS_WEN ビットで設定し、MSEBI_CS[n]_N（n＝0～3）ごとに管理できます。

書き込み保護が有効（bMSEBIS_WEN＝0）になっているとき、チップセレクトに対して MSEBI から書き込み要求が来ると、対応する rMSEBIS_STATUS レジスタのエラーフラグ（bMSEBIS_ERROR_WEN）がセットされます。本ステータスレジスタは、バスのマスタからアクセス可能です。

バスのマスタが rMSEBIS_STATUS レジスタの対応するビットに 1’b1 を書き込むと、bMSEBIS_ERROR_WEN ビットがクリアされます。

rMSEBIS_STATUS レジスタは CPU からアクセスできません。MSEBI マスタのバスアクセス用に予約されています。

10.4.7.8　MSEBI スレーブ：コンフィグレーションレジスタ &同期

すべてのレジスタは、AHB のクロックドメイン（MSEBIS_HCLK）上にありますが、一部の設定フィールド（フェーズ遅れ、ルーティング機能、ウェイト設定など）は、MSEBI のクロックドメイン（MSEBIS_CLK）上にあります。

MSEBI_CS[n]_N によって管理されるすべてのレジスタにおいて設定を更新する場合は、これらのフィールドを以下のように同期させる必要があります。

- 最初の AHB アクセス時に、bMSEBIS_CS_ENABLE ビットが CPU によってクリアされます。
- 同期機構の動作が終了するまでの間、下記の状態になります。
  - AHB バスからのデータは、MSEBI スレーブのレジスタにコピーされません。
  - AHB はウェイト状態になります（最大遅延時間はタイムアウトによって制御されます）。
- 同期機構の動作が完了すると、コンフィグレーションレジスタへの書き込みが許可されて、MSEBIS_CLK クロックに同期させるフィールドの値は（副作用を防止するため）デフォルト値に設定されます。
- AHB からの設定（bMSEBIS_CS_ENABLE ビットのセット）の終了後、レジスタへの書き込みアクセスがロックされて、MSEBIS_CLK クロックに同期させるフィールドの値はレジスタによって指定されます。

備　考

異なる MSEBIS_CS[m]_N（m≠n）上で管理されるコンフィグレーションレジスタは変更されません。
MSEBIS_CS[m]_N へのアクセスは正常に実行し続けます。

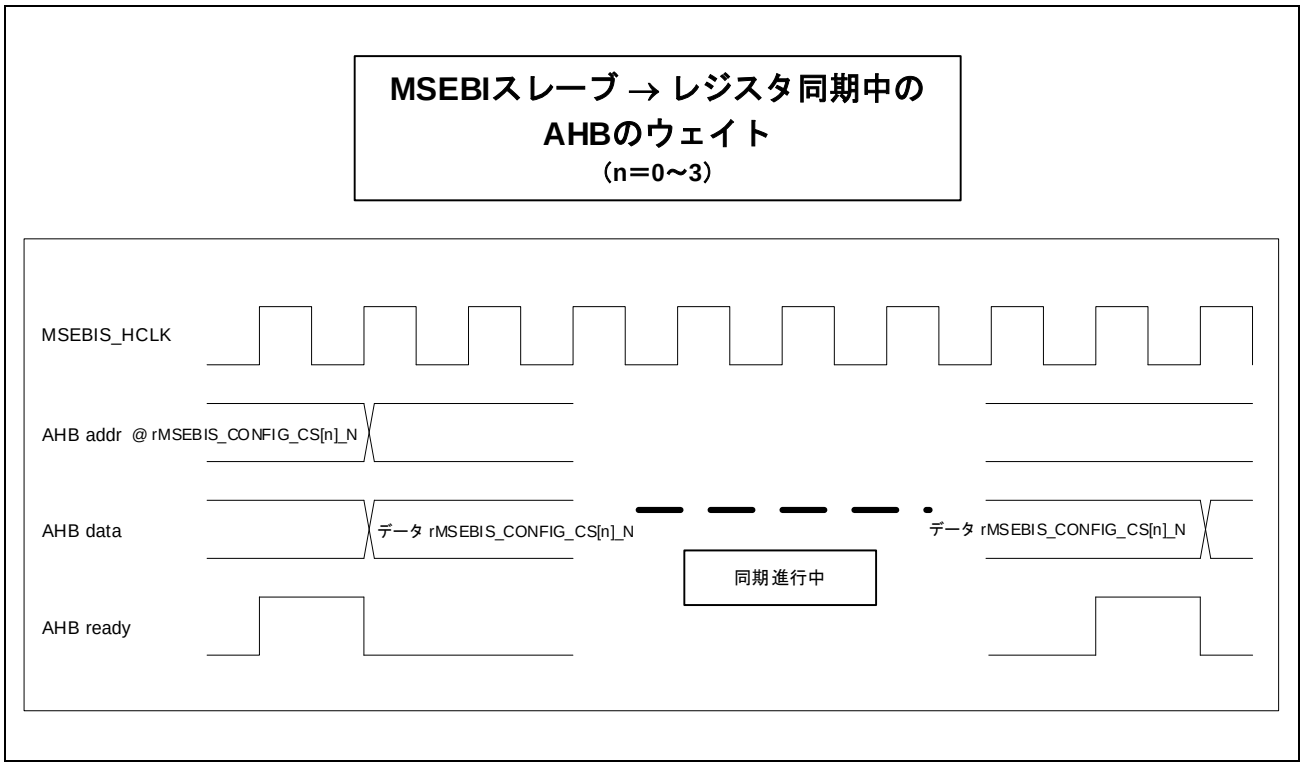


図 10.66　レジスタ同期機構の動作中における AHB のウェイト

10.5　使用上の注意事項

DMA 転送中に DMA コントローラがダウンした場合、以下の回復手順を実行する必要があります。

1. MSEBI マスタにおいて現在の DMA 転送を停止させます。
2. DMA RX コントロールレジスタ（rMSEBIM_RDMACR_CS[n]_N（n＝0、1））のバーストサイズ値をリセットし、シングルアクセスのみを許可します。
3. DMARDLR レジスタ（rMSEBIM_DMARDLR_CS[n]_N（n＝0、1））をリセットします。
4. 可能であれば、初期パラメータで DMA コントローラの動作を再開し、DMA MSEBI FIFO をアンロックします。バーストサイズ値（SRC_MSIZE または DEST_MSIZE）は 1（シングルアクセス）にします。
5. DMA 受信 FIFO がエンプティであることを確認します。
6. DMA コントローラを無効／停止にします。
7. MSEBI マスタにおいて DMA 転送を再開します。

改訂記録	RZ/N1Dグループ、RZ/N1Sグループ、RZ/N1Lグループ ユーザーズマニュアル　周辺機能2編

Rev.	発行日	改訂内容	
		ページ	ポイント
0.90	2018.03.30	—	初版発行
0.95	2018.10.19	全章	すべての章で表記修正、および体裁修正
		全章	すべての章でクロック表記統一
		6	このマニュアルの使い方、3. 略語および略称の説明、INTC、OTP　語句修正
		23～30	1.3.1～8 レジスタマップ UART1～8、表 1.1～8 レジスタマップ UART1～8　注釈追加
		31	1.4.1 rUart_DLL — 除数ラッチ（Low）、bUart_DLL　説明修正
		37	1.4.6 rUart_FCR — FIFO コントロールレジスタ、bUart_TET　説明修正
		46、47	1.4.10 rUart_MSR — モデムステータスレジスタ、機能（補数→反転）　説明修正
		51	1.4.14 rUart_TFR — 送信 FIFO 読み出し、bUart_TFR　表現修正
		57	1.4.19 rUart_SRR — ソフトウェアリセットレジスタ、bUart_XFR、bUart_RFR、bUart_UR　説明修正
		58	1.4.20 rUart_SRTS — シャドー送信要求、bUart_SRTS　説明修正
		59	1.4.21 rUart_SBCR — シャドーブレークコントロールレジスタ、bUart_SBCR　説明修正
		77	1.5.1.3 FIFO 管理（バイト→ビット）　説明修正
		77	1.5.1.4 クロック管理　説明修正
		79	1.5.1.6 割り込み、表 1.41 割り込み制御機能、bUart_IID（4'b0100→4'b1100）　値修正
		80	1.5.1.7 自動フロー制御（rUart_MCR ビット→rUart_MCR レジスタ）　語句修正
		85、86	1.5.1.9 DMA 管理（UART4、5、6、7、8 のみ）、(3) 送信ウォーターマークレベルの選択 説明修正
		87	1.5.1.9 DMA 管理（UART4、5、6、7、8 のみ）、(4) DEST_MSIZE の選択および送信 FIFO オーバーフロー　説明修正
		87	1.5.1.9 DMA 管理（UART4、5、6、7、8 のみ）、(7) SRC_MSIZE の選択および受信 FIFO アンダーフロー　説明修正
		—	第 2 章 SPI、全般的に（ポップ→データを取り出す、プッシュ→データを追加）　説明修正
		100	2.1 概要、受信 FIFO、スレーブセレクト数、（削除：ボーレートリファレンスクロック）説明修正
		102	2.2 信号インタフェース　説明修正
		110	2.4.1 rSpi_CTRLR0 — コントロールレジスタ 0、bSpi_TMOD、bSpi_SCPH　説明修正
		111	2.4.2 rSpi_CTRLR1 — コントロールレジスタ 1　注意修正
		114、115	2.4.5 rSpi_SER — スレーブイネーブルレジスタ、bSpi_CtrlSS、bSpi_SoftwareSS、bSpi_HardwareSS　説明修正
		119	2.4.9 rSpi_TXFLR — 送信 FIFO レベルレジスタ、bSpi_TXTFL（b3～b0→b4～b0）　説明修正
		120	2.4.10 rSpi_RXFLR — 受信 FIFO レベルレジスタ、bSpi_RXTFL（b3～b0→b4～b0）説明修正
		139	2.5.1 概要　説明修正
		140	2.5.2 SPI マスタと SPI スレーブ間の一般的な接続（シリアルビットレートクロック→シリアルクロック）　語句修正
		141	2.5.3 ハードウェアモードあるいはソフトウェアモードによるスレーブセレクトラインの制御　注意修正
		142	2.5.4 プログラマブルプリスケーラクロック　説明修正
		142	2.5.4 プログラマブルプリスケーラクロック、図 2.5 SPI マスタモード、最大クロック比 図削除
		142	2.5.4 プログラマブルプリスケーラクロック、図 2.6 SPI スレーブモード、最大クロック比　図削除

Rev.	発行日	改訂内容	
		ページ	ポイント
0.95	2018.10.19	143	2.5.5 データ入力サンプル遅延　説明修正
		146	2.5.8.3 受信のみモード（SPI_MOSI→SPI_MISO、その他）　説明修正
		148～150	2.5.9 モトローラシリアルペリフェラルインタフェース　説明修正
		157	2.5.11 ナショナルセミコンダクターMicrowire、図 2.19 ナショナルセミコンダクターモード、単一転送、受信データ、スレーブモードの SPI コントローラ　図修正
		157	2.5.11 ナショナルセミコンダクターMicrowire、図 2.20 ナショナルセミコンダクターモード、単一転送、送信データ、スレーブモードの SPI コントローラ　図修正
		158	2.5.12 DMA 制御（rSpi_DMACR→rSpi_TDMACR）　語句修正
		161	2.5.12.4 DEST_MSIZE の選択および送信 FIFO オーバーフロー　説明修正
		162	2.5.12.7 SRC_MSIZE の選択および受信 FIFO アンダーフロー　説明修正
		163、164	2.6.1.1 モトローラ&テキサスモードでのマスタ SPI のプログラミング　説明修正
		169	2.6.1.3 モトローラおよびテキサスモードでのスレーブ SPI のプログラミング　説明修正
		179	3.4.2 IC_TAR ― I2C 対象アドレスレジスタ、IC_TAR　説明修正
		198	3.4.19 IC_CLR_TX_ABRT ― TX_ABRT 割り込みクリアレジスタ、CLR_TX_ABRT　説明修正
		202、203	3.4.26 IC_STATUS ― I2C ステータスレジスタ、（ビット 10→4）、MST_ACTIVITY、TFNF　説明修正
		209	3.4.31 IC_SLV_DATA_NACK_ONLY ― スレーブデータ NACK 生成レジスタ、NACK　説明修正
		210	3.4.32 IC_SDA_SETUP ― I2C SDA セットアップレジスタ　説明修正
		214	3.4.35 IC_FS_SPKLEN ― I2C Sm、Fm スパイク抑止制限　説明修正
		217	3.5.1.1 初期設定、(4)　備考　説明削除
		219	3.5.1.3 シングルバイトのスレーブレシーバ動作、(4)　（IC_TX_TL→IC_RX_TL、その他）　説明修正
		219、220	3.5.1.4 バルク転送のスレーブ転送動作（R_RD_REQ→RD_REQ、その他）　説明修正
		221	3.5.2.1 初期設定、(2)、(3)　説明修正
		225	3.6.1 スパイク抑制（周波数→周波数周期）　説明修正
		230	4.1 概要（端子→信号）　表現修正
		230	4.1 概要、図 4.1 BGPIO の概要（32b AHBS→32b APBS）　図修正
		241	4.4.12 rGPIO_ext_porta ― GPIO ポート A データ入力レジスタ、bGPIO_ext_port（データレジスタ→データ出力レジスタ）　説明修正
		242	4.4.13 rGPIO_ext_portb ― GPIO ポート B データ入力レジスタ、bGPIO_ext_port（データレジスタ→データ出力レジスタ）　説明修正
		243	4.5.1.1 データおよび制御フロー　説明修正
		245	4.5.1.4 トリガ同期式動作　説明修正
		245	4.5.1.4 トリガ同期式動作、図 4.5 同期方式およびイベントでのキャプチャ　図修正
		246	4.5.1.4 トリガ同期式動作、表 4.19 トリガ同期式動作割り当てライン、表 4.20 トリガ同期式動作基本機能（端子→信号）　表現修正
		247	4.6.1 プログラミング上の注意事項　注意修正
		251	5.4.1 rTimerLoadCount_[n] ― サブタイマ[n]（n＝0～5）のプリセット値、bTimerLoadCount　注意修正
		252	5.4.2 rTimerLoadCount_[n] ― サブタイマ[n]（n＝6～7）のプリセット値、bTimerLoadCount　注意修正
		262	5.4.14 rTimer_DMA_PendingClrOvf ― タイマ DMA オーバーフロークリア（bTimer_DMA_RunnningClrOvf_6→bTimer_DMA_RunningClrOvf_6）　語句修正
		263	5.5.2 カウンタ 16 ビットまたは 32 ビット　説明修正
		265	5.5.2 カウンタ 16 ビットまたは 32 ビット、図 5.4 bTimerEnable のオン／オフ動作　タイトル修正
		266	5.5.3 割り込み（設定値から→設定値まで）　説明修正

Rev.	発行日	改訂内容	
		ページ	ポイント
0.95	2018.10.19	266	5.5.3 割り込み、図 5.5 タイマ割り込み（タイマ割り込みの立ち上がりタイミング） 図修正
		267	5.5.4 DMA 制御 説明修正
		—	第 6 章 CAN、全般的に（記録→保持） 語句修正
		269	6.1 概要（ビット位置を記録する→ビット位置情報付きの） 表現修正
		279	6.4.3 rCan_SR — コントローラステータスレジスタ、bCan_RBS（フル→ノットエンプティ） 説明修正
		295	6.4.16 rCan_ACR[n] — アクセプタンスコードフィルタ[n]レジスタ（n＝0～3）、bCan_ACR（rCan_AMR0→rCan_AMR[n]） 説明修正
		296	6.4.17 rCan_AMR[n] — アクセプタンスマスクフィルタ[n]レジスタ（n＝0～3）、bCan_AMR（rCan_ACR0→rCan_ACR[n]） 説明修正
		312	6.4.29 rCan_SyncPassiveError — 同期パッシブエラー検出レジスタ、bCan_SyncPassiveError 説明修正
		323	6.5.8.6 エラーパッシブ割り込み（遷移する→抜ける） 説明修正
		323	6.5.8.7 アービトレーションロスト割り込み 表現修正
		325	6.5.9 バスアービトレーション（バスの調停） 表現修正
		327	6.5.10 エラー処理、表 6.36 送受信エラーカウンタのインクリメント／デクリメント（エラー→エラーフラグ、オーバロード→オーバロードフラグ） 語句修正
		344	6.5.16 CAN コントローラとリファレンス Philips SJA1000 デバイスの違い、スリープモード： 説明修正
		346	7.1 概要 説明削除
		356	7.4.1.5 rADC_INTOVFSTATUS0 — マスキング前割り込みオーバーフロー、bADC_INTOVFSTATUS0_VC 説明修正
		365	7.4.1.13 rADC_FORCE — ADC 要求、bADC_FORCE_VC 説明修正
		369	7.4.1.16 rADC_PRIORITY — ADC 優先モード、bADC_Priority 表現修正
		380	7.4.1.22 rADC1_DATALOCK[n] — ADC1 データロック[n]レジスタ（n＝0～15）、bADC1_DATALOCK_Update（ADC1_DATA_Update→bADC1_DATA_Update ） 表現修正
		399	7.5.6 同時サンプル＆ホールド（ADC_VC1～4→ADC_VC1～3） 説明修正
		412	7.5.10.1 DMA 動作の概要（DMAC の転送元および転送先の転送幅→DMAC 転送元の転送幅） 注意修正
		417、419	8.4.1 rLcd_CR1 — コントロールレジスタ 1、bLcd_PSS、bLcd_LCE 説明修正
		433	8.4.13 rLcd_PWMFR_0 — PWM0 周波数レジスタ、bLcd_PWMFCD_0 説明修正
		434	8.4.14 rLcd_PWMDCR_0 — PWM0 デューティサイクルレジスタ、bLcd_PWMDC_0 説明修正
		437	8.4.17 rLcd_PWMFR_1 — PWM1 周波数レジスタ、bLcd_PWMFCD_1 説明修正
		438	8.4.18 rLcd_PWMDCR_1 — PWM1 デューティサイクルレジスタ、bLcd_PWMDC_1 説明修正
		445	8.5.1 主要機能の説明 説明修正
		449	8.5.4 DMA コントローラとメモリインタフェース 説明削除
		449	8.5.5 フレームバッファの構成 説明削除
		449	8.5.5 フレームバッファの構成（表 8.29 LCD フレームバッファのパレットロードサポート～表 8.34 LCD フレームバッファの構成（bLcd_PSS＝1 かつ bLcd_BPP＝3'b011） 表削除
		452	8.5.7 ピクセルアンパック、図 8.5 ピクセルアンパック — ビッグエンディアンピクセルバイトに置かれたビッグエンディアンフレームバッファバイト（Lcd_EPO=1→Lcd_EPO=x） 図修正
		454	8.5.8 パレットルックアップテーブル 説明修正
		457	8.5.9 出力 FIFO およびフォーマッタ、図 8.8 LCD 出力フォーマット指定 — bLcd_BPP：16、18、24（パレットで定義された RGB→フレームバッファ直接出力） 図修正

Rev.	発行日	改訂内容	
		ページ	ポイント
0.95	2018.10.19	462	8.5.15 点滅機能、図 8.10 点滅 BL[1:0]属性管理（クロック→クロック/4、クロック/4→クロック） 図修正
		463	8.5.16 制限事項、DMA によるパレットのロード 説明削除
		466	9.4.1 rSemaphoreLockCPU[m]_[n] — セマフォロック CPU[m]レジスタ[n] 説明追加
		467	9.4.2 rSemaphoreStatusCPU[m]_[n] — セマフォステータス CPU[m]レジスタ[n] 表現修正
		470	9.6 使用上の注意事項 注意修正
		474	10.1.2 CS[n]の MSEBI マスタアドレス割り当て（CPU）（表 10.2 CS[n]のアドレス割り当て（CPU）） 項追加
		524	10.3.3.10 rMSEBIS_CONFIG — コモンコンフィグレジスタ、bMSEBIS_WAIT_CONF（R→R/W） 修正
		538	10.3.4.3 rMSEBIS_ID_CS[n]_N — スレーブ ID レジスタ（n＝0～3）、bMSEBIS_ID（0x1234_FED0→0x1234_FEDn（n＝0～3）） 説明修正
		545	10.4.2.5 3 デバイス、モード 8/16/32、非同期式、図 10.6 3 デバイス、モード 8/16/32、非同期式 図修正
		546	10.4.2.6 3 デバイス、モード 8/16/32、同期式と非同期式の混在、図 10.7 3 デバイス、モード 8/16/32、同期式と非同期式の混在 図修正
		547	10.4.2.7 1 デバイス、モード 8、非同期式、パラレルモードの ALE、図 10.8 1 デバイス、モード 8、非同期式、パラレルモードの ALE（READY_N →READY） 図修正
		547	10.4.2.7 2 デバイス、モード 8/16、同期式、マルチマスタ（図 10.8 2 デバイス、モード 8/16、同期式、マルチマスタ） 項削除
		547	10.4.2.8 3 デバイス、モード 8、同期式／非同期式、マルチマスタ（図 10.9 3 デバイス、モード 8、同期式／非同期式、マルチマスタ） 項削除
		559～589 608～609	10.4.4.1 非同期式モード、ALE 数 1、図 10.9 MSEBI タイミング、非同期式モード、書き込み 1、ウェイトなし、非バースト、ALE 数 1（MSEBI(x)_HCLK 削除、読み出し図：セットアップデータとホールドデータの基準を MSEBI(x)_CLK に変更） 図修正 10.4.4.1～10.4.4.7、10.4.6.2（図 10.10～36、図 10.51～52）10.4.4.1 非同期式モード、ALE 数 1、図 10.9 MSEBI タイミング、非同期式モード、書き込み 1、ウェイトなし、非バースト、ALE 数 1 と同様な修正
		571～582 584～589	10.4.4.4 同期式モード、非バースト、ALE 数 1、信号名に関する注記： 説明修正 10.4.4.5～10.4.4.8 10.4.4.4 同期式モード、非バースト、ALE 数 1 と同様な修正
		592	10.4.5.2 MSEBI 割り込み：マスタによるブロック終了検出、図 10.39 MSEBI マスタ：DMA RX によるブロック転送の終了（チェック→ポーリング） 図修正
		595	10.4.5.3 MSEBI 割り込み：スレーブによるブロック終了検出（チェック→ポーリング） 説明修正
		596	10.4.5.3 MSEBI 割り込み：スレーブによるブロック終了検出、図 10.42 MSEBI スレーブ：MSEBI CPU マスタからの書き込みブロック転送の終了（チェック→ポーリング） 図修正
		600	10.4.6.2 MSEBI マスタ：バーストモード、図 10.45 MSEBI：ラウンドロビン優先順位 図修正
		605	10.4.6.2 MSEBI マスタ：バーストモード、(2)、図 10.49 MSEBI：バーストモード、DMA 受信 FIFO とバスインタフェースの結合（書き出し→読み出し） 図修正
		616	10.4.6.3 MSEBI マスタ：DMA 制御、(4)、図 10.54 MSEBI ケース 1：送信ウォーターマークレベル 図修正
		618	10.4.6.3 MSEBI マスタ：DMA 制御、(5) DEST_MSIZE の選択と送信 FIFO オーバーフロー 説明修正
		619	10.4.6.3 MSEBI マスタ：DMA 制御、(8) SRC_MSIZE の選択と受信 FIFO アンダーフロー 説明修正
		624	10.4.7.2 MSEBI スレーブ：バーストモード、(1)、図 10.58 MSEBI スレーブの CPU FIFO の例 1（書き込み 0～15→書き込み 0～3） 図修正
		626	10.4.7.2 MSEBI スレーブ：バーストモード、(1)、図 10.60 MSEBI スレーブの CPU FIFO の例 2（書き込み 0～15→書き込み 0～3） 図修正

Rev.	発行日	改訂内容	
		ページ	ポイント
0.95	2018.10.19	630	10.4.7.2 MSEBI スレーブ：バーストモード、(2)、図 10.62 MSEBI マスタの DMA RX FIFO からの要求に対するスレーブの DMA FIFO（点線追加：使用されないデータ範囲）図修正
		632	10.4.7.2 MSEBI スレーブ：バーストモード、(3)、図 10.63 MSEBI スレーブ：外部 DMA 要求（bMSEBIS_DMARX_FORCE、bMSEBIS_DMATX_FORCE）　説明追加
1.00	2019.03.29	—	すべての章で表記修正、および体裁修正
		31	1.4.1 rUart_DLL — 除数ラッチ（Low）、bUart_DLL　説明修正
		32	1.4.2 rUart_DLH — 除数ラッチ（High）、bUart_DLH　説明修正
		37～38	1.4.6 rUart_FCR — FIFO コントロールレジスタ　説明修正
		39～40	1.4.7 rUart_LCR — ラインコントロールレジスタ　説明修正
		41～42	1.4.8 rUart_MCR — モデムコントロールレジスタ、bUart_LB、bUart_RTS　説明修正
		54	1.4.16 rUart_USR — UART ステータスレジスタ、bUart_BUSY　説明修正
		57	1.4.19 rUart_SRR — ソフトウェアリセットレジスタ　説明修正
		60	1.4.22 rUart_SFE — シャドーFIFO 許可、bUart_SFE　説明修正
		64	1.4.26 rUart_DMASA — DMA ソフトウェアアクノリッジ、bUart_DMASA　説明修正
		65	1.4.27 rUart_TO — タイムアウトカウンタコンフィグレーションレジスタ、bUart_TO3 説明修正
		67	1.4.28 rUart_CTRLTO — タイムアウトコントロールレジスタ、bUart_TG　説明修正
		69	1.4.29 rUart_STATUSTO — タイムアウトカウンタステータスレジスタ、bUart_TIMEOUTStatus3　説明修正
		75～76	1.5.1.1 UART（RS232）シリアルプロトコル　説明修正
		76	1.5.1.1 UART（RS232）シリアルプロトコル、図 1.4 レシーバシリアルデータサンプル 図修正
		76	1.5.1.2 19200 ボーのボーレート許容値　説明追加
		78	1.5.1.5 連続キャラクタストリーム送信　説明修正
		82	1.5.1.8 プログラマブル THRE 割り込み、図 1.7 割り込み生成、プログラマブル THRE 割り込みモード&FIFO 許可のフローチャート　図修正
		84	1.5.1.9 DMA 管理（UART4、5、6、7、8 のみ）、(1) DMA 動作の概要　説明修正
		91～95	1.5.1.10 MODBUS 管理用トランシーバ&レシーバタイムアウト　説明修正
		93	1.5.1.10 MODBUS 管理用トランシーバ&レシーバタイムアウト、図 1.14 レシーバタイムアウトの概要　図修正
		94	1.5.1.10 MODBUS 管理用トランシーバ&レシーバタイムアウト、図 1.15 レシーバ&トランシーバのタイムアウト（n＝0～3）タイミング　図修正
		95	1.5.1.10 MODBUS 管理用トランシーバ&レシーバタイムアウト、図 1.16 トランシーバタイムアウトの概要　図修正
		97	1.5.1.11 半二重モード管理、　説明修正
		97	1.5.1.11 半二重モード管理、図 1.17 トランシーバタイムガードの概要　図修正
		98	1.5.1.11 半二重モード管理、図 1.18 半二重モードでのデータイネーブルの自動生成　図修正
		101	2.1 概要、図 2.2 SPI スレーブの概要　図修正
		110	2.4.1 rSpi_CTRLR0 — コントロールレジスタ 0、bSpi_SCPH　説明修正
		112	2.4.3 rSpi_SSIENR — イネーブルレジスタ、bSpi_SSIENR　説明修正
		114	2.4.5 rSpi_SER — スレーブイネーブルレジスタ　説明修正
		117	2.4.7 rSpi_TXFTLR — 送信 FIFO しきい値レベル　説明修正
		118	2.4.8 rSpi_RXFTLR — 受信 FIFO しきい値レベル　説明修正
		133	2.4.22 rSpi_DR — データレジスタ、bSpi_DR　説明削除
		155	2.5.11 ナショナルセミコンダクターMicrowire　説明削除
		168	2.6.1.2 ナショナルセミコンダクターモードでのマスタ SPI のプログラミング、図 2.25 マスタモード時の SPI コントローラ、ナショナルセミコンダクターモード　図修正

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2019.03.29	171	2.6.1.3 モトローラおよびテキサスモードでのスレーブ SPI のプログラミング、図 2.26 スレーブモード時の SPI コントローラ、モトローラおよびテキサスモード　図修正
		177	3.4.1 IC_CON — I2C コントロールレジスタ、RX_FIFO_FULL_HLD_CTRL　説明修正
		187	3.4.9 IC_INTR_STAT — I2C 割り込みステータスレジスタ、ALL_BITS　説明修正
		188～189	3.4.10 IC_INTR_MASK — I2C 割り込みマスクレジスタ、ALL_BITS　説明修正
		190～192	3.4.11 IC_RAW_INTR_STAT — I2C ロウ（raw）割り込みステータスレジスタ　説明修正
		202	3.4.26 IC_STATUS — I2C ステータスレジスタ、IC_STATUS_ACTIVITY　説明修正
		206	3.4.30 IC_TX_ABRT_SOURCE — I2C 送信アボート要因レジスタ、ABRT_USER_ABRT 説明修正
		237	4.4.7 rGPIO_inttype_level — GPIO ポート A 割り込みレベルレジスタ、bGPIO_inttype_levell　語句修正
		239	4.4.9 rGPIO_intstatus — GPIO ポート A 割り込みステータス、bGPIO_intstatus　説明修正
		239	4.4.10 rGPIO_raw_intstatus — GPIO ポート A ロウ（raw）割り込みステータス（マスク前）、bGPIO_raw_intstatus　説明修正
		252	5.4.3 rTimerCurrentCount_[n] — サブタイマ[n]（n＝0～5）の現在値、bTimerCurrentCount　説明修正
		252	5.4.4 rTimerCurrentCount_[n] — サブタイマ[n]（n＝6～7）の現在値、bTimerCurrentCount　説明修正
		345	7.1 概要、DMA 接続　説明修正
		410	7.5.10 DMA 制御　説明修正
		412	7.6 使用上の注意事項、7.6.1 制約事項　項追加
		448	8.5.3 タイミング＆制御、図 8.2 LCD 水平方向タイミング　図修正
		474	10.1.1 信号インタフェース、表 10.1 信号インタフェース　注釈追加
		493	10.3.1.5 rMSEBIM_ADDRDMA_READ_CS[n]_N — DMA 読み出しアドレスレジスタ（n＝0、1）、bMSEBIM_ADDRDMA_READ_2　説明削除
		539	10.4.1.1 AHB スレーブインタフェース、MSEBI マスタ用　説明追加
		556	10.4.3.3 データフェーズ：SETUP＋VALID＋HOLD（DATA）、（1kB 境界）　説明修正
		599、601	10.4.6.2 MSEBI マスタ：バーストモード、（1kB 境界）　説明修正
		622～623 627、629	10.4.7.2 MSEBI スレーブ：バーストモード、（1kB 境界）　説明修正
1.10	2020.09.30	484	10.2.4 レジスタマップ：MSEBI から MSEBI スレーブ、表 10.13 レジスタマップ：MSEBI から MSEBI スレーブ　説明追記
		534	10.3.4.1 rMSEBIS_INT — スレーブ割り込みレジスタ、アドレス　説明追記
		536	10.3.4.2 rMSEBIS_STATUS — スレーブステータスレジスタ、アドレス　説明追記
		538	10.3.4.3 rMSEBIS_ID_CS[n]_N — スレーブ ID レジスタ（n＝0～3）、アドレス　説明追記
		634	10.4.7.4 MSEBI スレーブ：マスタによるレジスタアクセス　説明修正
1.20	2021.12.29	—	すべての章で表記修正、および体裁修正
		22	1.2 信号インタフェース、UART[m]_RTS_N　説明修正
		67、68	1.4.28 rUart_CTRLTO — タイムアウトコントロールレジスタ　説明修正
		69	1.4.29 rUart_STATUSTO — タイムアウトカウンタステータスレジスタ、bUart_DE　説明修正
		90	1.5.1.10 MODBUS 管理用トランシーバ＆レシーバタイムアウト　説明追加
		93	1.5.1.10 MODBUS 管理用トランシーバ＆レシーバタイムアウト、(1) レシーバタイムアウト、図 1.14 レシーバタイムアウトの概要　図修正
		93	1.5.1.10 MODBUS 管理用トランシーバ＆レシーバタイムアウト、(1) レシーバタイムアウト　説明修正

Rev.	発行日	改訂内容	
		ページ	ポイント
1.20	2021.12.29	96	1.5.1.10 MODBUS 管理用トランシーバ&レシーバタイムアウト、(3) タイムアウトカウンタのタイミング、図 1.15 レシーバ&トランシーバのタイムアウト（n＝0～3）タイミング図移動
		95	1.5.1.10 MODBUS 管理用トランシーバ&レシーバタイムアウト、(2) トランシーバタイムアウト、図 1.16 トランシーバタイムアウトの概要　図修正
		95	1.5.1.10 MODBUS 管理用トランシーバ&レシーバタイムアウト、(2) トランシーバタイムアウト　説明修正
		—	1.5.1.11 半二重モード管理　項削除
		97	1.5.2 使用上の注意事項　項追加
		161	2.6.1 プログラミング上の注意事項、注意　説明修正

RZ/N1Dグループ、RZ/N1Sグループ、RZ/N1Lグループ
ユーザーズマニュアル　周辺機能2編

発行年月日　2018 年 03 月 30 日　Rev.0.90
2021 年 12 月 29 日　Rev.1.20

発行　ルネサス エレクトロニクス株式会社
〒135-0061　東京都江東区豊洲 3-2-24（豊洲フォレシア）

RZ/N1D グループ、RZ/N1S グループ、RZ/N1L グループ

Renesas Electronics Corporation

R01UH0752JJ0120