Gowin BSRAM & SSRAM

ユーザーガイド

UG285-1.3.5J, 2023-01-05

免責事項

当社は、GOWINSEMI Terms and Conditions of Sale（GOWINSEMI取引条件）に規定されている内容を除き、（明示的か又は黙示的かに拘わらず）いかなる保証もせず、また、知的財産権や材料の使用によりあなたのハードウェア、ソフトウェア、データ、又は財産が被った損害についても責任を負いません。本文書における全ての情報は、予備的情報として取り扱われなければなりません。当社は、事前の通知なく、いつでも本文書の内容を変更することができます。本文書を参照する何れの団体及び個人も、最新の文書やエラッタ（不具合情報）については、当社に問い合わせる必要があります。

バージョン履歴

日付	バージョン	説明
2016/05/17	1.05J	初版。
2016/07/15	1.06J	図面を更新。
2016/10/27	1.07J	GW2AR シリーズ FPGA 製品へのサポートを追加。
2017/05/03	1.08J	● BSRAM のタイミング図を更新し、ROM、バイトイネーブル信号、バイトパリティ、パワーアップ状況、出力レジスタリセット、位置制約を追加。 ● 付録 A を追加。
2018/05/31	1.09J	● 第三章 ポートとパラメータ紹介を追加。 ● メモリ拡張を追加。 ● A.3 読み出し/書き込みの注意事項を更新。
2019/04/03	1.1J	表 A.1 書き込みの注意事項のリストを更新。
2020/08/17	1.2J	マニュアルの構造を最適化。
2021/06/21	1.3J	IP 呼び出しの図面を更新し、Help 内容を削除。
2021/10/12	1.3.1J	RESET の説明を更新。
2022/07/22	1.3.2J	デバイスの情報を更新。
2022/08/11	1.3.3J	デバイスのバージョン情報を更新。
2022/11/11	1.3.4J	GW1NS-2 を削除。
2023/01/03	1.3.5J	IP 呼び出しの図面の更新、Device Version オプションを追加。

目次

図一覧

表一覧

1 本マニュアルについて

1.1 マニュアル内容

このマニュアルは、主に GOWIN セミコンダクターの BSRAM と SSRAM の特性、動作モード、プリミティブ、及び IP の呼び出しなどについて説明します。

1.2 関連ドキュメント

GOWIN セミコンダクターのウェブサイト www.gowinsemi.com/ja から、以下の関連ドキュメントがダウンロード、参考できます：

- GW1N シリーズ FPGA 製品データシート(DS100)
- GW1NR シリーズ FPGA 製品データシート(DS117)
- GW2A シリーズ FPGA 製品データシート（DS102）
- GW2AR シリーズ FPGA 製品データシート(DS226)
- Gowin ソフトウェア ユーザーガイド（SUG100）

1.3 用語、略語

本マニュアルで使用される用語、略語、及びその意味については、表 1-1 を参照してください。

表 1-1 用語、略語

用語、略語	正式名称	意味
BSRAM	Block SRAM	ブロック SRAM
CFU	Configurable Function Unit	コンフィギャラブル機能ユニット
CST	Constraints	物理制約ファイル
DP	True Dual Port 16K Block SRAM	16K デュアルポート BSRAM
ROM	Read-Only Memory	読み出し専用メモリ
SDP	Semi Dual Port 16K Block SRAM	16K セミ・デュアルポート BSRAM
SP	Single Port 16K Block SRAM	16K シングルポート BSRAM
SSRAM	Shadow SRAM	分散 SRAM

1.4 テクニカル・サポートとフィードバック

GOWIN セミコンダクターは、包括的な技術サポートをご提供しています。使用に関するご質問、ご意見については、直接弊社までお問い合わせください。

Web サイト：www.gowinsemi.com/ja

E-mail：support@gowinsemi.com

2 概要

Gowin FPGA 製品には、ブロック SRAM（BSRAM）や分散 SRAM（SSRAM）などの豊富なメモリリソースがあります。

各 BSRAM は最大 18Kbits に構成でき、データ幅やアドレスの深さも構成可能です。各 BSRAM には、A ポートと B ポートの 2 つの独立したポートがあります。2 つのポートには独立したクロック、アドレス、データ、及び制御信号があるため、個別に読み出し/書き込みを行うことができます。また、この 2 つのポートは 1 つのメモリ領域を共有します。

Gowin FPGA の基本構成要素であるコンフィギャラブル機能ユニット (CFU) は、アプリケーションシナリオに応じて、16 x 4 ビットの SRAM または ROM（ROM16）を含む SSRAM として構成できます。

2.1 BSRAM の特性

- 1 つの BSRAM の最大容量は 18Kbits
- クロック周波数は最大 380MHz(Read-before-write モードの場合は 230MHz)
- シングルポートモード（SP）をサポート
- デュアルポートモード（DP）をサポート
- セミ・デュアルポートモード（SDP）をサポート
- 読み出し専用モード（ROM）をサポート
- 最大 36 ビットのデータ幅をサポート
- デュアルポートモードとセミ・デュアルポートモードは、独立した読み出し/書き込みクロックと独立したデータ幅をサポート
- 読み出しはレジスタ出力またはバイパス出力をサポート
- 書き込みは Normal モード、read-before-write モード、および write-through モードをサポート

2.2 BSRAM のモード

各 BSRAM は 16Kbits または 18Kbits に構成でき、構成可能なデータ幅及びアドレス深さは表 2-1 に示す通りです。

表 2-1 BSRAM の構成モード一覧

容量	シングルポートモード	デュアルポートモード	セミ・デュアルポートモード	ROM モード
16Kbits	16K x 1	16K x 1	16K x 1	16K x 1
	8K x 2	8K x 2	8K x 2	8K x 2
	4K x 4	4K x 4	4K x 4	4K x 4
	2K x 8	2K x 8	2K x 8	2K x 8
	1K x 16	1K x 16	1K x 16	1K x 16
	512 x 32	-	512 x 32	512 x 32
18Kbits	2K x 9	2K x 9	2K x 9	2K x 9
	1K x 18	1K x 18	1K x 18	1K x 18
	512 x 36	-	512 x 36	512 x 36

各 BSRAM のアドレスバス幅は 14 ビット（すなわち、AD[13:0]）であるため、最大アドレス深さは 16,384 になります。データ幅とアドレス幅の対応関係は表 2-2 に示す通りです。

表 2-2 BSRAM のデータ幅とアドレス幅の対応関係

容量	構成モード	データ幅	アドレス深さ	アドレス幅
16Kbits	16K x 1	[0:0]	16,384	[13:0]
	8K x 2	[1:0]	8,192	[13:1]
	4K x 4	[3:0]	4,096	[13:2]
	2K x 8	[7:0]	2,048	[13:3]
	1K x 16	[15:0]	1,024	[13:4]
	512 x 32	[31:0]	512	[13:5]
18Kbits	2K x 9	[8:0]	2,048	[13:3]
	1K x 18	[17:0]	1,024	[13:4]
	512 x 36	[35:0]	512	[13:5]

デュアルポートとセミ・デュアルポートモードの書き込みクロック及び読み出しクロックは独立しており、独立した読み出し/書き込みのデータ幅がサポートされています。デュアルポートモードでは、A ポートと B ポートがサポートするデータ幅は表 2-3 に示す通りです。セミ・デュアルポートモードでは、A ポートと B ポートがサポートするデータ幅は表 2-4 に示す通りです。

表 2-3 デュアルポートモードにおけるデータ幅構成

容量	B ポート	A ポート						
		16K x 1	8K x 2	4K x 4	2K x 8	1K x 16	2K x 9	1K x 18
16Kbits	16K x 1	Yes	Yes	Yes	Yes	Yes	N/A	N/A
	8K x 2	Yes	Yes	Yes	Yes	Yes	N/A	N/A
	4K x 4	Yes	Yes	Yes	Yes	Yes	N/A	N/A
	2K x 8	Yes	Yes	Yes	Yes	Yes	N/A	N/A
	1K x 16	Yes	Yes	Yes	Yes	Yes	N/A	N/A
18Kbits	2K x 9	N/A	N/A	N/A	N/A	N/A	Yes	Yes
	1K x 18	N/A	N/A	N/A	N/A	N/A	Yes	Yes

表 2-4 セミ・デュアルポートモードにおけるデータ幅構成

容量	B ポート	A ポート								
		16K x 1	8K x 2	4K x 4	2K x 8	1K x 16	512x32	2K x 9	1K x 18	512 x 36
16Kbits	16K x 1	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A
	8K x 2	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A
	4K x 4	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A
	2K x 8	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A
	1K x 16	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A
	512 x 32	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A
18Kbits	2K x 9	N/A	N/A	N/A	N/A	N/A	N/A	Yes	Yes	Yes
	1K x 18	N/A	N/A	N/A	N/A	N/A	N/A	Yes	Yes	Yes

3 BSRAM プリミティブ

Block Memory は、静的アクセス機能を備えたブロック状の SRAM です。BSRAM の特性によれば、シングルポートモード（SP/SPX9）、デュアルポートモード（DPB/DPX9B）、セミ・デュアルポートモード（SDPB/SDPX9B）、および読み出し専用モード（pROM/pROMX9）に分類できます。

注記：

- GW1N-9/GW1N-1S/GW1NR-9/GW1NS-4 はデュアルポートモードをサポートしません。
- GW1N-9/GW1NR-9/GW1NS-4 の場合、32/36 ビット幅の SP/SPX9 は 2 つの SP/SPX9 によって実装されるため、2 つの BSRAM を占有します。
- GW1NZ-1/GW1NZ-1C は 1/2/4/8/9 ビット幅のデュアルポートモードをサポートしません。
- GW1N-4D/GW1NR-4D/GW2AN-18X/GW2AN-9X は 1/2/4/8/9 ビット幅のデュアルポートモードでの read-before-write モードをサポートしません。

3.1 デュアルポートモード

プリミティブの紹介

DPB/DPX9B(True Dual Port 16K Block SRAM/True Dual Port 18K Block SRAM)：16K/18K デュアルポート BSRAM。

機能の説明

DPB/DPX9B はそれぞれメモリ領域が 16K bit/18K bit であるデュアルポートモードの BSRAM です。A ポートと B ポートは個別に読み出し/書き込みを実現できます。2 つの読み出しモード（bypass モードと pipeline モード）と 3 つの書き込みモード（Normal モード、write-through モード、read-before-write モード）がサポートされます。

- 読み出しモード

パラメータの READ_MODE0、READ_MODE1 は、A および B ポート出力 pipeline レジスタを有効または無効にするために使用されます。出力 pipeline レジスタを使用する場合、読み出しには追加の遅延期間が必要です。

- 書き込みモード

Normal モード、write-through モード、および read-before-write モードがあります。A ポートおよび B ポートの書き込みモードは、それぞれパラメータ WRITE_MODE0 および WRITE_MODE1 によって構成されます。異なるモードに対応する内部タイミング波形を図 3-1 から図 3-6 に示します。

図 3-1 DPB/DPX9B Normal 書き込みモードのタイミング図 （Bypass 読み出しモード）

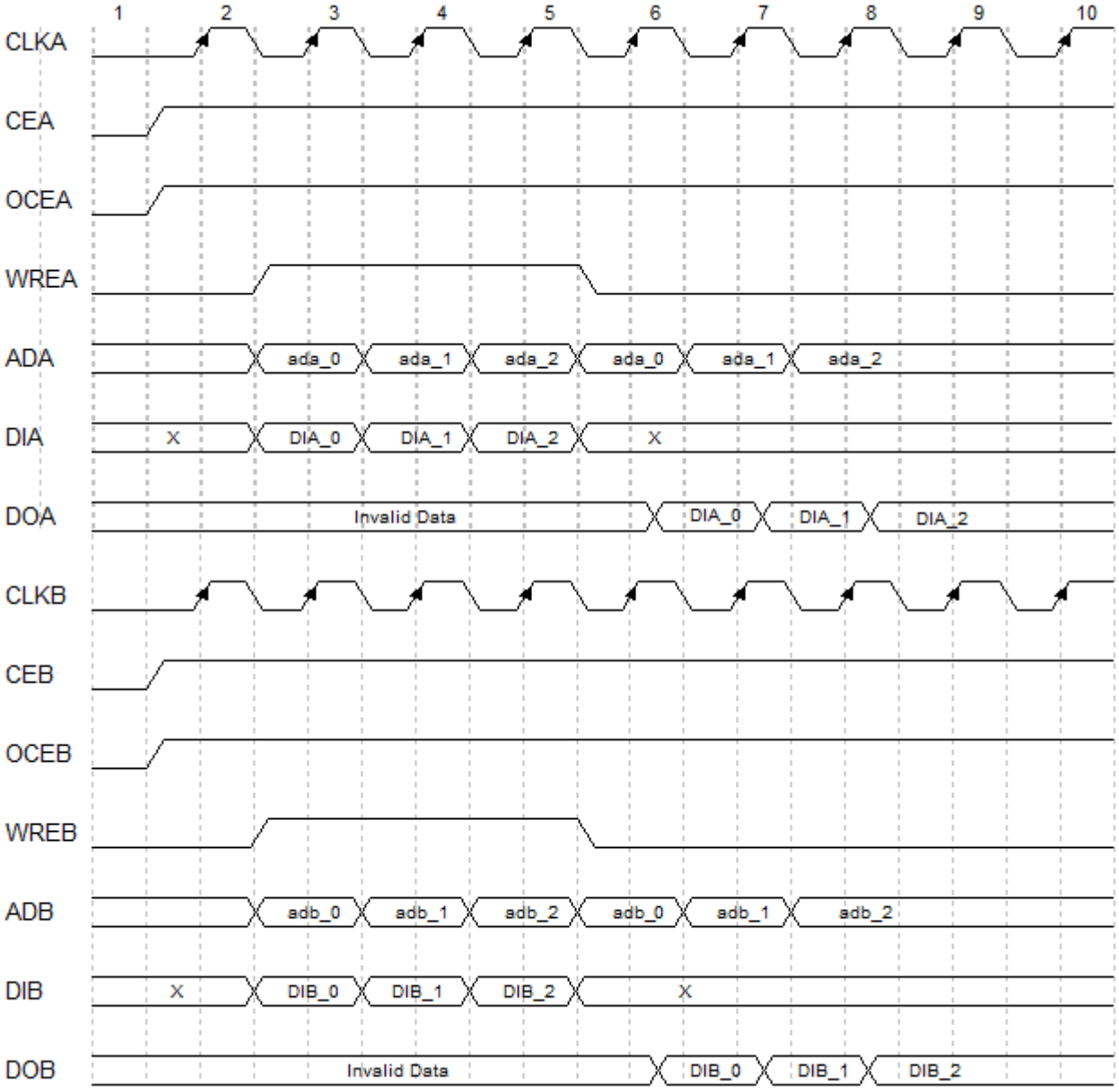


図 3-2 DPB/DPX9B Normal 書き込みモードのタイミング図（Pipeline 読み出しモード）

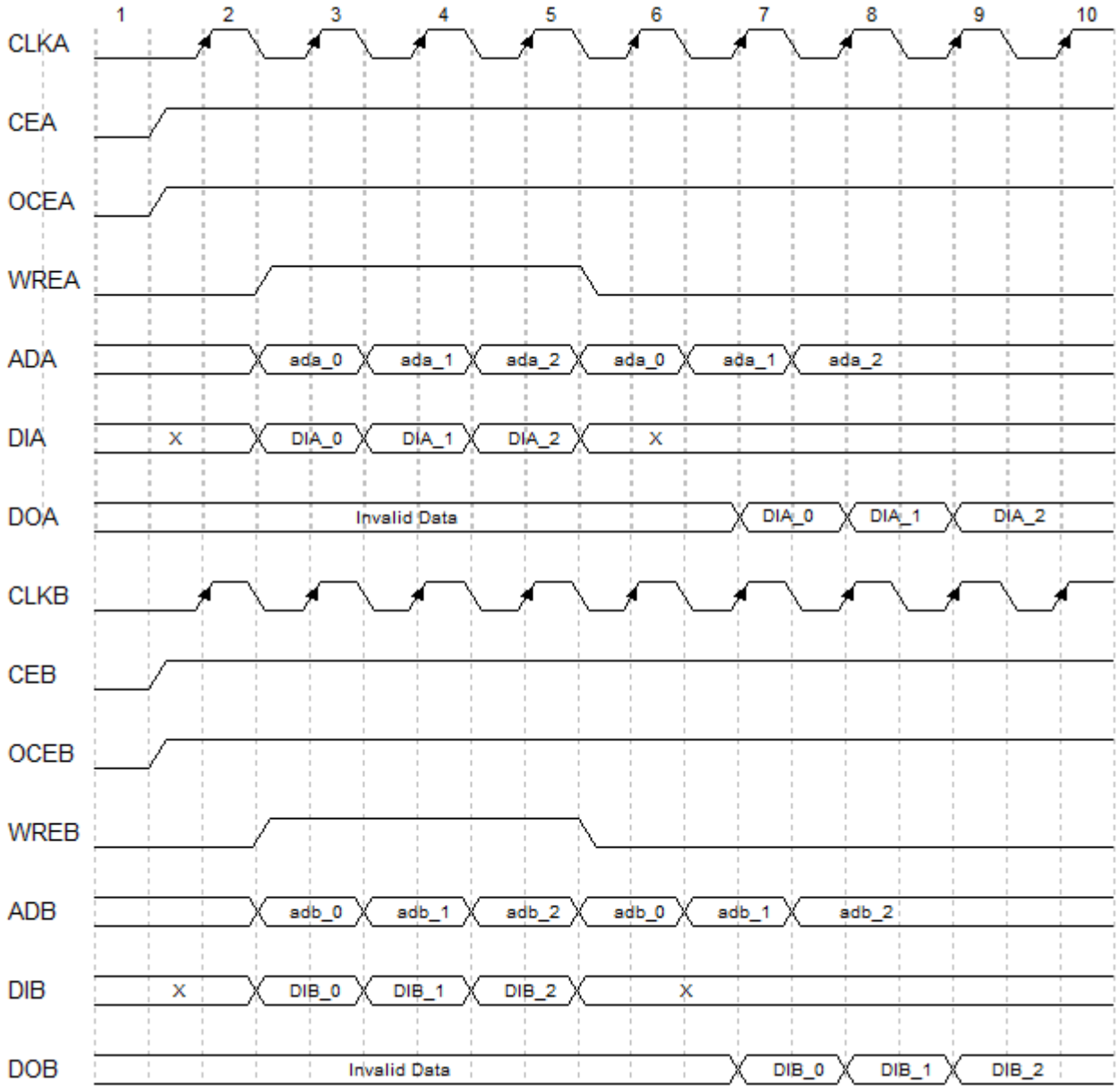


図 3-3 DPB/DPX9B Write-through 書き込みモードのタイミング図 （Bypass 読み出しモード）

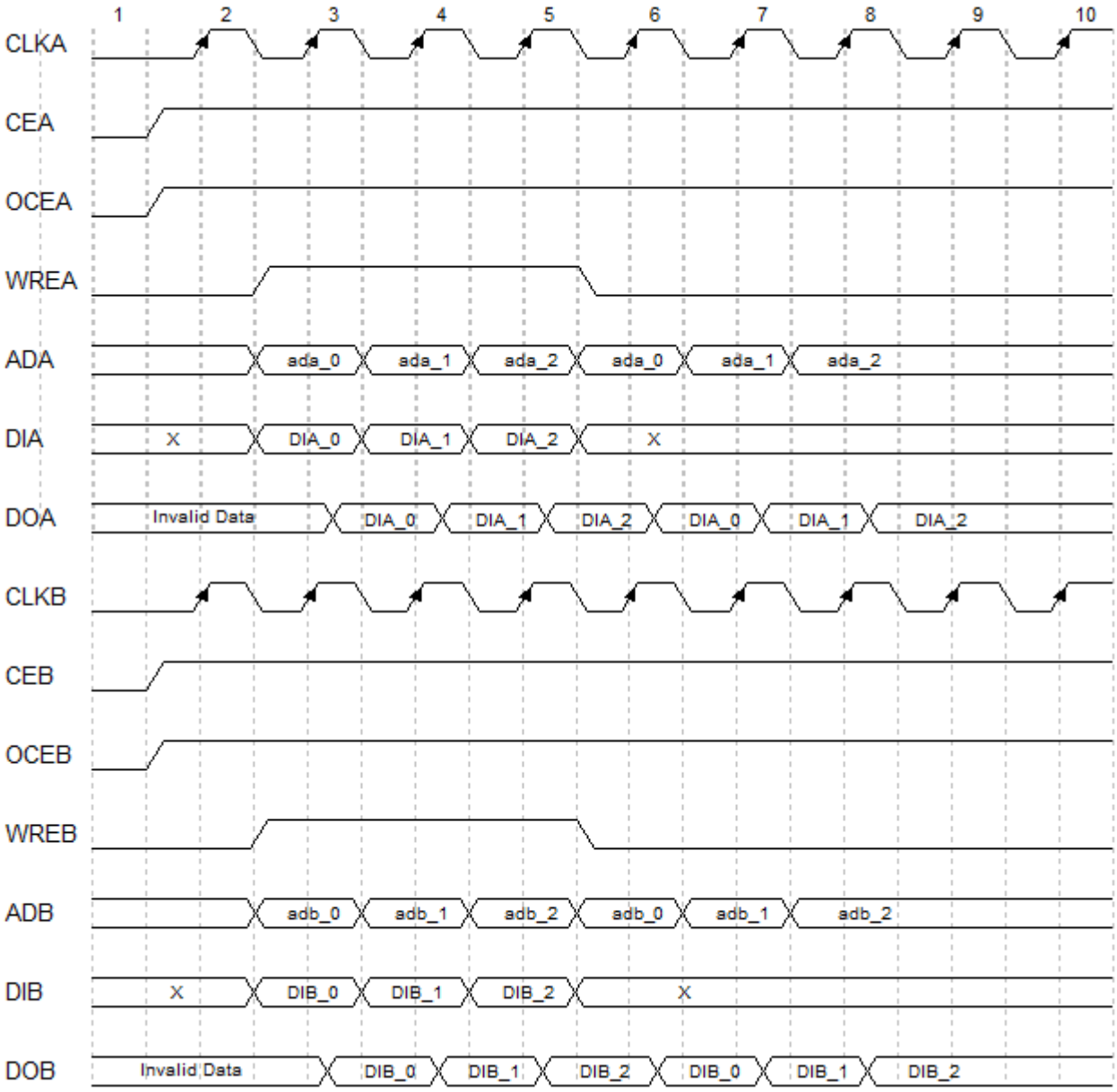


図 3-4 DPB/DPX9B Write-through 書き込みモードのタイミング図（Pipeline 読み出しモード）

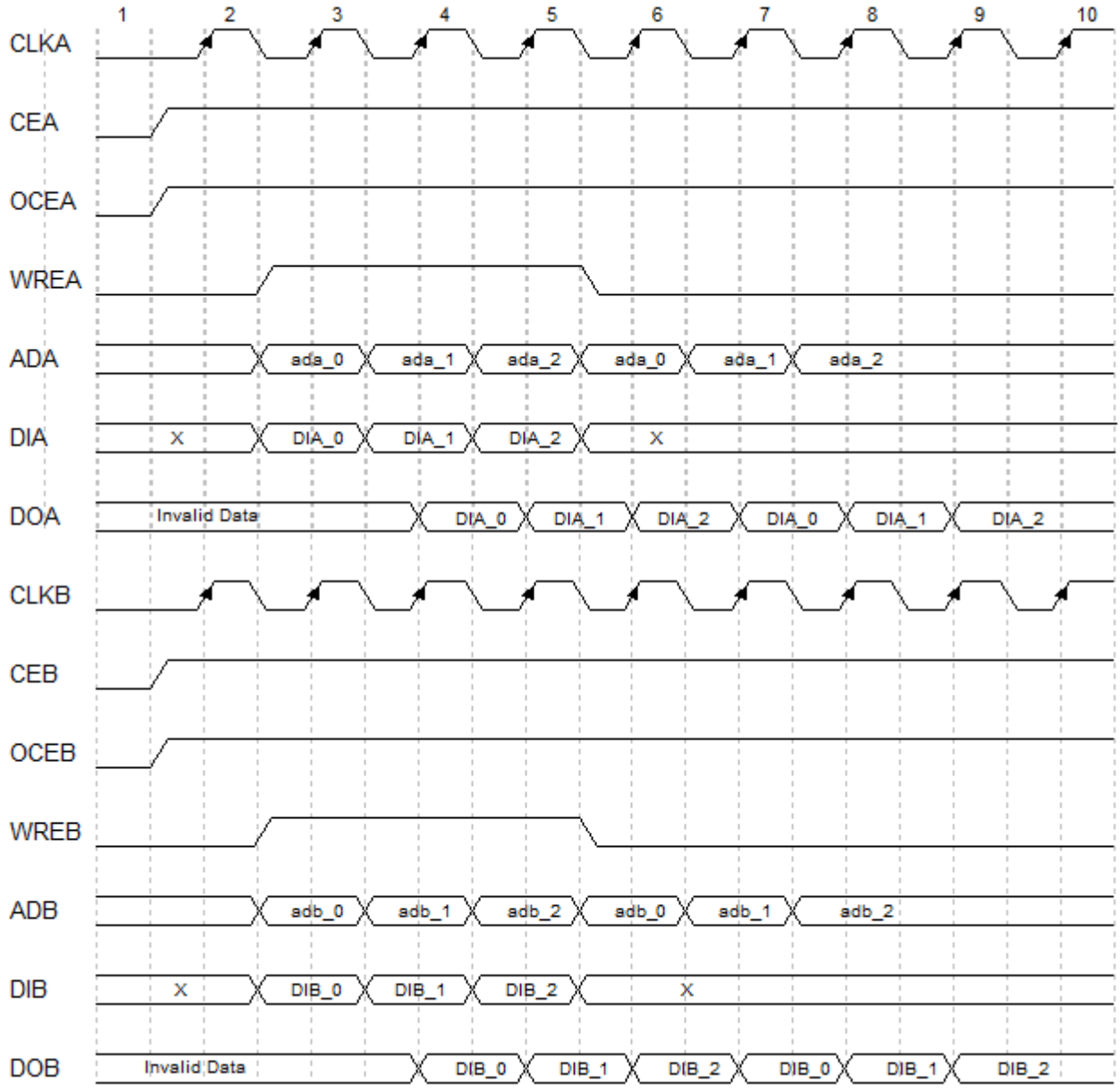


図 3-5 DPB/DPX9B Read-before-write 書き込みモードのタイミング図 （Bypass 読み出しモード）

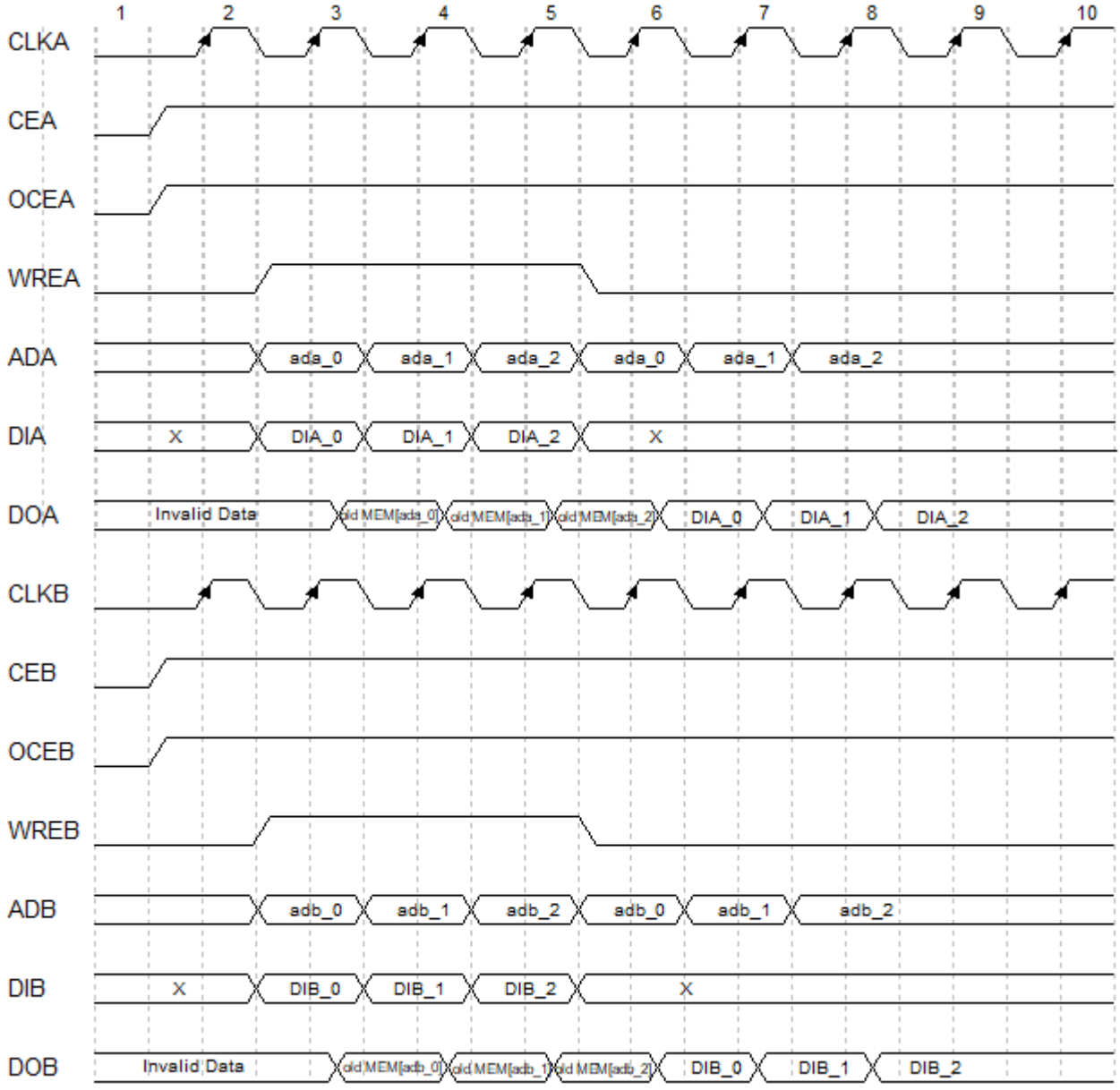
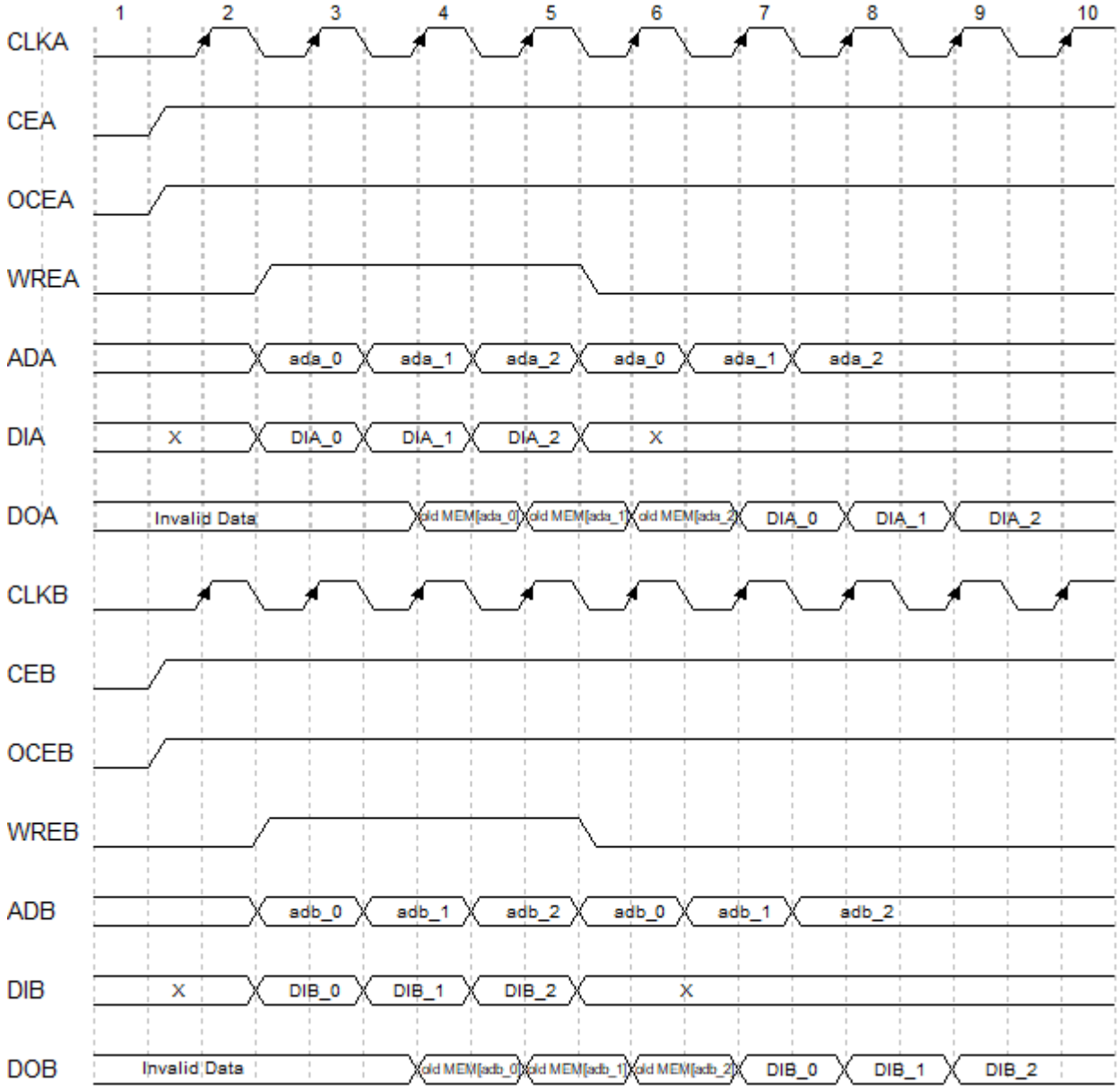


図 3-6 DPB/DPX9B Read-before-write 書き込みモードのタイミング図（Pipeline 読み出しモード）



対応関係

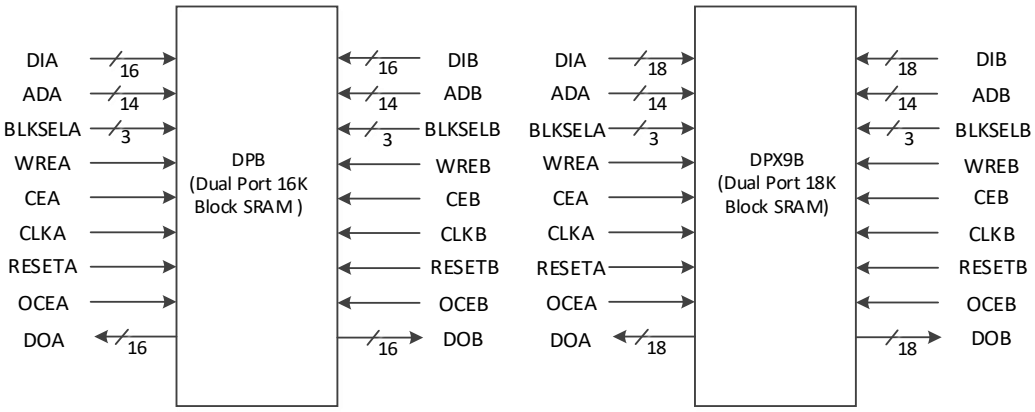
表 3-1 DPB/DPX9B データ幅とアドレス深さの対応関係

デュアルポートモード	BSRAM 容量	データ幅	アドレス深さ
DPB	16Kbits	1	14
		2	13
		4	12
		8	11
		16	10
DPX9B	18Kbits	9	11

デュアルポートモード	BSRAM 容量	データ幅	アドレス深さ
		18	10

ポート図

図 3-7 DPB/DPX9B のポート図



ポートの説明

表 3-2 DPB/DPX9B のポートの説明

ポート名	I/O	説明
DOA[15:0]/DOA[17:0]	出力	A ポートデータ出力
DOB[15:0]/DOB[17:0]	出力	B ポートデータ出力
DIA[15:0]/DIA[17:0]	入力	A ポートデータ入力
DIB[15:0]/DIB[17:0]	入力	B ポートデータ入力
ADA[13:0]	入力	A ポートアドレス入力
ADB[13:0]	入力	B ポートアドレス入力
WREA	入力	A ポート書き込みイネーブル入力 1：書き込み 0：読み出し
WREB	入力	B ポート書き込みイネーブル入力 1：書き込み 0：読み出し
CEA	入力	A ポートクロックイネーブル信号、アクティブ High
CEB	入力	B ポートクロックイネーブル信号、アクティブ High
CLKA	入力	A ポートクロック入力
CLKB	入力	B ポートクロック入力
RESETA	入力	A ポートリセット入力。同期リセットおよび非同期リセットをサポート、アクティブ High。RESETA は、メモリ内の値をリセットするのではなく、レジスタをリセットします

ポート名	I/O	説明
RESETB	入力	B ポートリセット入力。同期リセットおよび非同期リセットをサポート、アクティブ High。RESETB は、メモリ内の値をリセットするのではなく、レジスタをリセットします
OCEA	入力	A ポート出力クロックイネーブル信号。A ポートの pipeline モードに使用、bypass モードには無効
OCEB	入力	B ポート出力クロックイネーブル信号。B ポートの pipeline モードに使用、bypass モードには無効
BLKSELA[2:0]	入力	A ポートブロック選択信号。容量拡張のために複数の BSRAM をカスケード接続する際に使用
BLKSELB[2:0]	入力	B ポートブロック選択信号。容量拡張のために複数の BSRAM をカスケード接続する際に使用

パラメータの説明

表 3-3 DPB/DPX9B のパラメータの説明

パラメータ名	パラメータのタイプ	値の範囲	デフォルト値	説明
READ_MODE0	Integer	1'b0,1'b1	1'b0	A ポート読み出しモード構成 1'b0:bypass モード 1'b1:pipeline モード
READ_MODE1	Integer	1'b0,1'b1	1'b0	B ポート読み出しモード構成 1'b0:bypass モード 1'b1:pipeline モード
WRITE_MODE0	Integer	2'b00,2'b01,2'b10	2'b00	A ポート書き込みモード構成 2'b00: Normal モード 2'b01: write-through モード 2'b10: read-before-write モード
WRITE_MODE1	Integer	2'b00,2'b01,2'b10	2'b00	B ポート書き込みモード構成 2'b00: Normal モード 2'b01: write-through モード 2'b10: read-before-write モード
BIT_WIDTH_0	Integer	DPB:1,2,4,8,16 DPX9B:9,18	DPB:16 DPX9B:18	A ポートデータ幅構成
BIT_WIDTH_1	Integer	DPB:1,2,4,8,16 DPX9B:9,18	DPB:16 DPB:18	B ポートデータ幅構成
BLK_SEL_0	Integer	3'b000~3'b111	3'b000	A ポートブロック選択のパラメータの設定。ポート BLKSELA の値と同じ場合にこの BSRAM が選択されま

パラメータ名	パラメータのタイプ	値の範囲	デフォルト値	説明
				す。IP Core Generator を使用してメモリ拡張を行う場合、拡張は自動的に実行されます。
BLK_SEL_1	Integer	3'b000~3'b111	3'b000	B ポートブロック選択のパラメータの設定。ポート BLKSELB の値と同じ場合にこの BSRAM が選択されます。IP Core Generator を使用してメモリ拡張を行う場合、拡張は自動的に実行されます。
RESET_MODE	String	"SYNC","ASYNC"	"SYNC"	リセットモードの構成 SYNC：同期リセット ASYNC：非同期リセット
INIT_RAM_00 ~ INIT_RAM_3F	Integer	DPB:256'h0 … 0~256'h1…1 DPX9B:288'h0 … 0~288'h1…1	DPB:256'h0…0 DPX9B:288'h0…0	BSRAM メモリセルの初期化データを設定するために使用されます

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。

DPB のインスタンス化を例に説明します：

Verilog でのインスタンス化：

```
DPB bram_dpb_0 (
        .DOA({doa[15:8],doa[7:0]}),
        .DOB({doa[15:8],dob[7:0]}),
    .CLKA(clka),
    .OCEA(ocea),
    .CEA(cea),
    .RESETA(reseta),
    .WREA(wrea),
    .CLKB(clkb),
    .OCEB(oceb),
    .CEB(ceb),
    .RESETB(resetb),
    .WREB(wreb),
```

```
        .BLKSELA({3'b000}),
        .BLKSELB({3'b000}),
        .ADA({ada[10:0],3'b000}),
          .DIA({{8{1'b0}},dia[7:0]})
        .ADB({adb[10:0],3'b000}),
          .DIB({{8{1'b0}},dib[7:0]})
    );
      defparam bram_dpb_0.READ_MODE0 = 1'b0;
      defparam bram_dpb_0.READ_MODE1 = 1'b0;
      defparam bram_dpb_0.WRITE_MODE0 = 2'b00;
      defparam bram_dpb_0.WRITE_MODE1 = 2'b00;
      defparam bram_dpb_0.BIT_WIDTH_0 = 8;
      defparam bram_dpb_0.BIT_WIDTH_1 = 8;
      defparam bram_dpb_0.BLK_SEL_0 = 3'b000;
      defparam bram_dpb_0.BLK_SEL_1 = 3'b000;
      defparam bram_dpb_0.RESET_MODE = "SYNC";
      defparam bram_dpb_0.INIT_RAM_00 =
256'h00A000000000000B00A000000000000B00A000000000000B00A00
0000000000B;
      defparam bram_dpb_0.INIT_RAM_3E =
256'h00A000000000000B00A000000000000B00A000000000000B00A00
0000000000B;
      defparam bram_dpb_0.INIT_RAM_3F =
256'h00A000000000000B00A000000000000B00A000000000000B00A00
0000000000B;
```

VHDL でのインスタンス化：

```
  COMPONENT  DPB
            GENERIC  (
                      BIT_WIDTH_0:integer:=16;
                      BIT_WIDTH_1:integer:=16;
                      READ_MODE0:bit:='0';
                      READ_MODE1:bit:='0';
                      WRITE_MODE0:bit_vector:="00";
                      WRITE_MODE1:bit_vector:="00";
                      BLK_SEL_0:bit_vector:="000";
                      BLK_SEL_1:bit_vector:="000";
                      RESET_MODE:string:="SYNC";
```

```
                    INIT_RAM_00:bit_vector:=X"000000000000000
0000000000000000000000000000000000000000000000000";
                    INIT_RAM_01:bit_vector:=X"000000000000000
0000000000000000000000000000000000000000000000000";
                    INIT_RAM_3F:bit_vector:=X"000000000000000
0000000000000000000000000000000000000000000000000"
        );
    PORT (
                    DOA,DOB:OUT std_logic_vector(15 downto 0):
=conv_std_logic_vector(0,16);
                    CLKA,CLKB,CEA,CEB,OCEA,OCEB,RESETA,
RESETB,WREA,WREB:IN std_logic;
                    ADA,ADB:IN std_logic_vector(13 downto 0);
                    BLKSELA:IN std_logic_vector(2 downto 0);
                    BLKSELB:IN std_logic_vector(2 downto 0);
                    DIA,DIB:IN std_logic_vector(15 downto 0)
        );
  END COMPONENT;
  uut:DPB
      GENERIC MAP(
                    BIT_WIDTH_0=>16,
                    BIT_WIDTH_1=>16,
                    READ_MODE0=>'0',
                    READ_MODE1=>'0',
                    WRITE_MODE0=>"00",
                    WRITE_MODE1=>"00",
                    BLK_SEL_0=>"000",
                    BLK_SEL_1=>"000",
                    RESET_MODE=>"SYNC",
  INIT_RAM_00=>X"000000000000000000000000000000000000000000000
0000000000000000000",
  INIT_RAM_01=>X"000000000000000000000000000000000000000000000
0000000000000000000",
  INIT_RAM_3F=>X"000000000000000000000000000000000000000000000
0000000000000000000"
      )
      PORT MAP(
```

```
        DOA=>doa,
        DOB=>dob,
        CLKA=>clka,
        CLKB=>clkb,
        CEA=>ceb,
        CEB=>ceb,
        OCEA=>ocea,
        OCEB=>oceb,
        RESETA=>reseta,
        RESETB=>resetb,
        WREA=>wrea,
        WREB=>wreb,
        ADA=>ada,
        ADB=>adb,
        BLKSELA=>blksela,
        BLKSELB=>blkselb,
        DIA=>dia,
        DIB=>dib
    );
```

3.2 シングルポートモード

プリミティブの紹介

SP/SPX9(Single Port 16K BSRAM/Single Port 18K B- SRAM)：16K/18K のシングルポート BSRAM。

機能の説明

SP/SPX9 のメモリ容量は 16K bit/18K bit で、動作モードはシングルポートモードです。1 つのクロックでシングルポートの読み出し/書き込みを制御し、2 つの読み出しモード（bypass モードと pipeline モード）と 3 つの書き込みモード（Normal モード、write-through モード、read-before-write モード）をサポートします。

- 読み出しモード

パラメータの READ_MODE は、出力 pipeline レジスタを有効または無効にするために使用されます。出力 pipeline レジスタを使用する場合、読み出しには追加の遅延期間が必要です。

- 書き込みモード

書き込みモードには、Normal モード、write-through モード、および read-before-write モードがあり、パラメータの WRITE_MODE により構成されます。

シングルポート BSRAM の各読み出し書き込みモードに対応する内部タイミング波形については、デュアルポート BSRAM の場合のタイミング図である図 3-1～ 図 3-6 を参照してください。

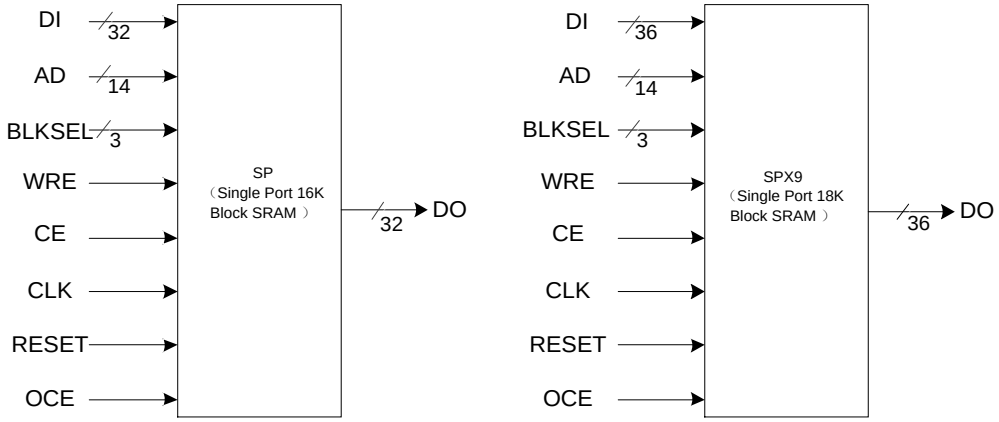
対応関係

表 3-4 SP/SPX9 データ幅とアドレス深さの対応関係

シングルポートモード	BSRAM 容量	データ幅	アドレス深さ
SP	16Kbits	1	14
		2	13
		4	12
		8	11
		16	10
		32	9
SPX9	18Kbits	9	11
		18	10
		36	9

ポート図

図 3-8 SP/SPX9 のポート図



ポートの説明

表 3-5 SP/SPX9 のポートの説明

ポート名	I/O	説明
DO[31:0]/DO[35:0]	出力	データ出力信号
DI[31:0]/DI[35:0]	入力	データ入力
AD[13:0]	入力	アドレス入力

ポート名	I/O	説明
WRE	入力	書き込みイネーブル入力 1：書き込み 0：読み出し
CE	入力	クロックイネーブル入力、アクティブ High
CLK	入力	クロック入力
RESET	入力	リセット入力。同期リセットおよび非同期リセットをサポート、アクティブ High。RESET は、メモリ内の値をリセットするのではなく、レジスタをリセットします
OCE	入力	出力クロックイネーブル信号。pipeline モードに使用、bypass モードには無効
BLKSEL[2:0]	入力	BSRAM ブロック選択信号。容量拡張のために複数の BSRAM をカスケード接続する際に使用

パラメータの説明

表 3-6 SP/SPX9 のパラメータの説明

パラメータ名	パラメータのタイプ	値の範囲	デフォルト値	説明
READ_MODE	Integer	1'b0,1'b1	1'b0	読み出しモード構成 1'b0:bypass モード 1'b1:pipeline モード
WRITE_MODE	Integer	2'b00,2'b01,2'b10	2'b00	書き込みモード構成 2'b00: Normal モード 2'b01:write-through モード; 2'b10: read-before-write モード
BIT_WIDTH	Integer	SP:1,2,4,8,16,32 SPX9:9,18,36	SP:32 SPX9:36	データ幅構成
BLK_SEL	Integer	3'b000~3'b111	3'b000	BSRAM ブロック選択のパラメータの設定。ポート BLKSEL の値と同じ場合にこの BSRAM が選択されます。IP Core Generator を使用してメモリ拡張を行う場合、拡張は自動的に実行されます。
RESET_MODE	String	"SYNC","ASYNC"	"SYNC"	リセットモードの構成 SYNC：同期リセット ASYNC：非同期リセット
INIT_RAM_00~ INIT_RAM_3F	Integer	SP:256'h0…0~256'h1…1 SPX9:288'h0 … 0~288'h1…1	SP:256'h0…0 SPX9:288'h0…0	BSRAM の初期化データを設定するために使用されます

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。SP のインスタンス化を例に説明します：

Verilog でのインスタンス化：

```
SP bram_sp_0 (
    .DO({dout[31:8], dout[7:0]}),
    .CLK(clk),
    .OCE(oce),
    .CE(ce),
    .RESET(reset),
    .WRE(wre),
    .BLKSEL({3'b000}),
    .AD({ad[10:0], 3'b000}),
    .DI({{24{1'b0}}, din[7:0]})
);
defparam bram_sp_0.READ_MODE = 1'b0;
defparam bram_sp_0.WRITE_MODE = 2'b00;
defparam bram_sp_0.BIT_WIDTH = 8;
defparam bram_sp_0.BLK_SEL = 3'b000;
defparam bram_sp_0.RESET_MODE = "SYNC";
  defparam bram_sp_0.INIT_RAM_00 =
  256'h00A000000000000B00A000000000000B00A000000000000B00
  A000000000000B;
  defparam bram_sp_0.INIT_RAM_01 =
  256'h00A000000000000B00A000000000000B00A000000000000B00
  A000000000000B;
  defparam bram_sp_0.INIT_RAM_3F =
  256'h00A000000000000B00A000000000000B00A000000000000B00
  A000000000000B;
```

VHDL でのインスタンス化：

```
  COMPONENT SP
        GENERIC(
              BIT_WIDTH:integer:=32;
              READ_MODE:bit:='0';
              WRITE_MODE:bit_vector:="01";
```

```
                        BLK_SEL:bit_vector:="000";
                        RESET_MODE:string:="SYNC";
                        INIT_RAM_00:bit_vector:=X"00A000000000000B
00A000000000000B00A000000000000B00A000000000000B ";
                        INIT_RAM_01:bit_vector:=X"00A000000000000B
00A000000000000B00A000000000000B00A000000000000B ";
                        INIT_RAM_3F:bit_vector:=X"00A000000000000B
00A000000000000B00A000000000000B00A000000000000B "
                );
                  PORT(
                        DO:OUT std_logic_vector(31 downto 0):=conv_
std_logic_vector(0,32);
                        CLK,CE,OCE,RESET,WRE:IN std_logic;
                        AD:IN std_logic_vector(13 downto 0);
                        BLKSEL:IN std_logic_vector(2 downto 0);
                        DI:IN std_logic_vector(31 downto 0)
                 );
        END COMPONENT;
        uut:SP
             GENERIC MAP(
                           BIT_WIDTH=>32,
                           READ_MODE=>'0',
                           WRITE_MODE=>"01",
                           BLK_SEL=>"000",
                           RESET_MODE=>"SYNC",
                           INIT_RAM_00=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B ",
                           INIT_RAM_01=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B ",
                           INIT_RAM_02=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B ",
                           INIT_RAM_3F=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B "
                )
              PORT MAP (
                   DO=>dout,
                   CLK=>clk,
```

```
        OCE=>oce,
        CE=>ce,
        RESET=>reset,
        WRE=>wre,
        BLKSEL=>blksel,
        AD=>ad,
        DI=>din
    );
```

3.3 セミ・デュアルポートモード

プリミティブの紹介

SDPB/SDPX9B(Semi Dual Port 16K Block SRAM /Semi Dual Port 18K Block SRAM)：16K/18K のセミ・デュアルポート BSRAM。

機能の説明

SDPB/SDPX9B のメモリ領域はそれぞれ 16K bit/18K bit で、その動作モードはセミ・デュアルポートモードです、ポート A は書き込み、ポート B は読み出しに使用されます。2 つの読み出しモード（bypass モードと pipeline モード）と 1 つの書き込みモード（Normal モード）がサポートされます。

- 読み出しモード

パラメータの READ_MODE は、出力 pipeline レジスタを有効または無効にするために使用されます。出力 pipeline レジスタを使用する場合、読み出しには追加の遅延期間が必要です。

- 書き込みモード

SDPB/SDPX9B ポート A は書き込み、ポート B は読み出しを実行します。Normal モードをサポートします。

セミ・デュアルポート BSRAM の各モードに対応する内部タイミング波形を図 3-9 および図 3-10 に示します。

図 3-9 セミ・デュアルポート BSRAM Normal 書き込みモードのタイミング図（Bypass 読み出しモード）

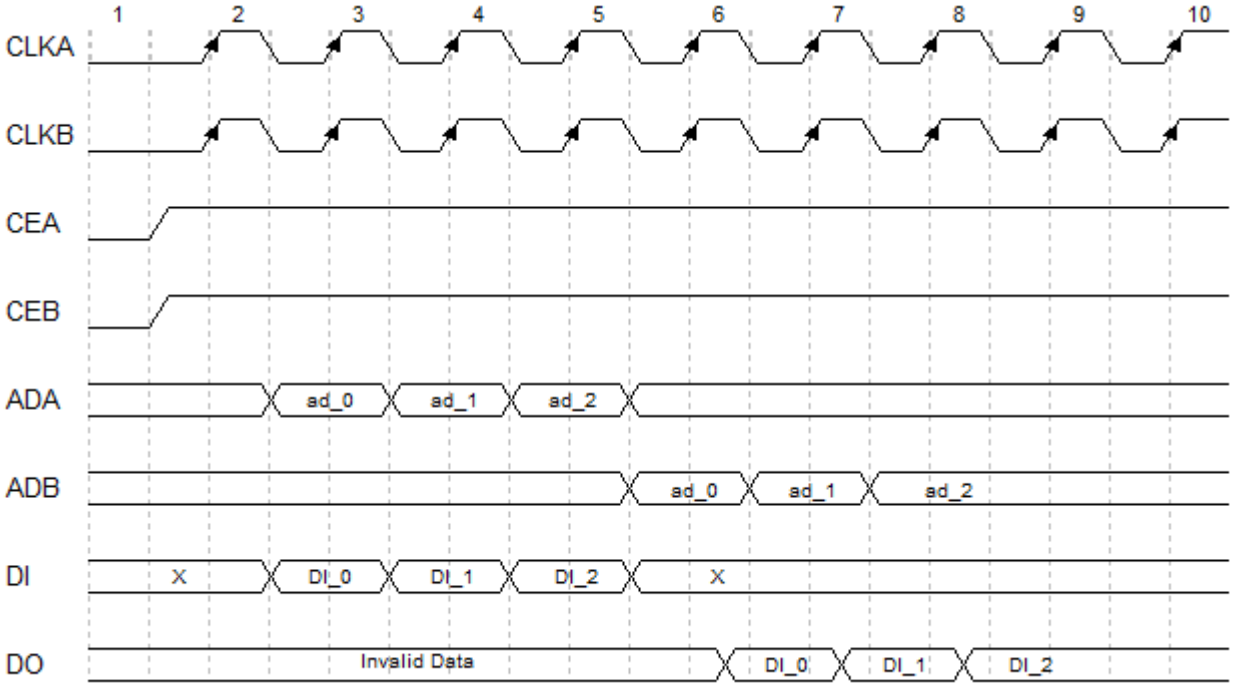
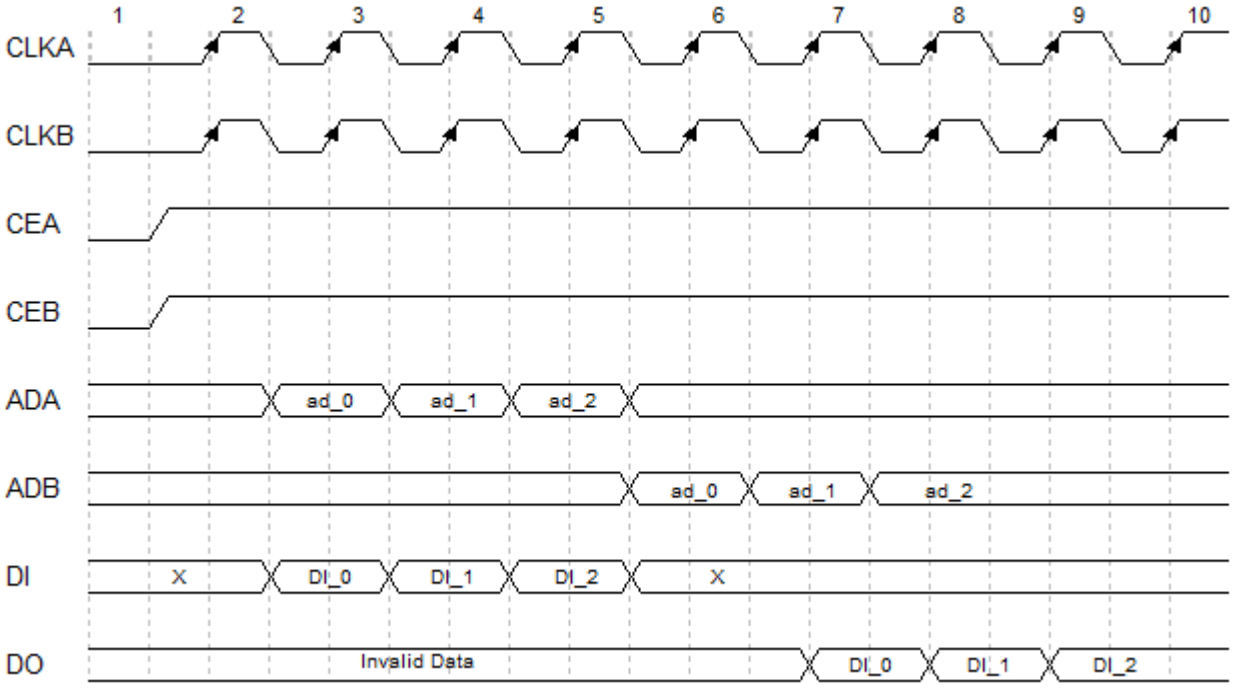


図 3-10 セミ・デュアルポート BSRAM Normal 書き込みモードのタイミング図（Pipeline 読み出しモード）



対応関係

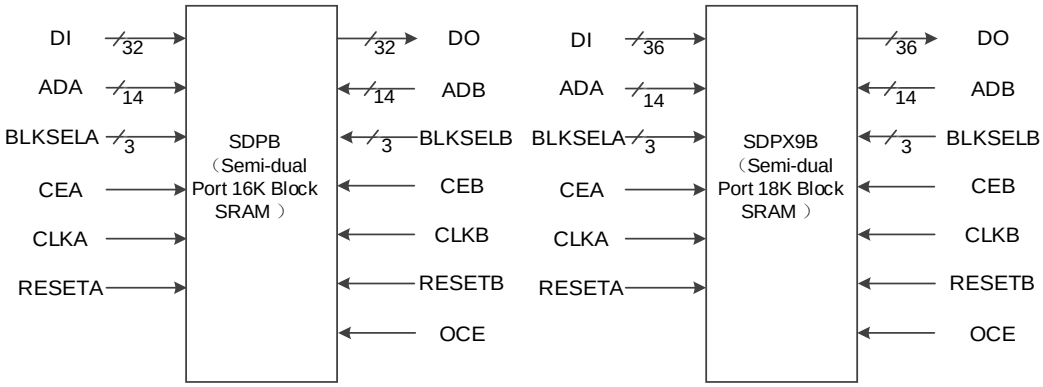
表 3-7 SDPB/SDPX9B データ幅とアドレス深さの対応関係

セミ・デュアルポートモード	BSRAM 容量	データ幅	アドレス深さ
SDPB	16Kbits	1	14

セミ・デュアルポートモード	BSRAM 容量	データ幅	アドレス深さ
		2	13
		4	12
		8	11
		16	10
		32	9
SDPX9B	18Kbits	9	11
		18	10
		36	9

ポート図

図 3-11 SDPB/SDPX9B のポート図



ポートの説明

表 3-8 SDPB/SDPX9B のポートの説明

ポート名	I/O	説明
DO[31:0]/DO[35:0]	出力	データ出力信号
DI[31:0]/DI[35:0]	入力	データ入力信号
ADA[13:0]	入力	A ポートアドレス入力信号
ADB[13:0]	入力	B ポートアドレス入力信号
CEA	入力	A ポートクロックイネーブル信号、アクティブ High
CEB	入力	B ポートクロックイネーブル信号、アクティブ High
CLKA	入力	A ポートクロック入力信号
CLKB	入力	B ポートクロック入力信号
RESETA	入力	A ポートリセット入力信号。同期リセットおよび非同期リセットをサポート、アクティブ High。RESETA は、メモリ内の値をリセットするのではなく、レジスタをリセットします

ポート名	I/O	説明
RESETB	入力	B ポートリセット入力信号。同期リセットおよび非同期リセットをサポート、アクティブ High。RESETB は、メモリ内の値をリセットするのではなく、レジスタをリセットします
OCE	入力	出力クロックイネーブル信号。pipeline モードに使用、bypass モードには無効
BLKSELA[2:0]	入力	A ポートブロック選択信号。容量拡張のために複数の BSRAM をカスケード接続する際に使用
BLKSELB[2:0]	入力	B ポートブロック選択信号。容量拡張のために複数の BSRAM をカスケード接続する際に使用

パラメータの説明

表 3-9 SDPB/SDPX9B のパラメータの説明

パラメータ名	パラメータのタイプ	値の範囲	デフォルト値	説明
READ_MODE	Integer	1'b0,1'b1	1'b0	読み出しモード構成 1'b0:bypass モード 1'b1:pipeline モード
BIT_WIDTH_0	Integer	SDPB:1,2,4,8,16,32 SDPX9B:9,18,36	SDPB:32 SDPX9B:36	A ポートデータ幅構成
BIT_WIDTH_1	Integer	SDPB:1,2,4,8,16,32 SDPX9B:9,18,36	SDPB:32 SDPX9B:36	B ポートデータ幅構成
BLK_SEL_0	Integer	3'b000~3'b111	3'b000	A ポートブロック選択のパラメータの設定。ポート BLKSEL の値と同じ場合にこの BSRAM が選択されます。IP Core Generator を使用してメモリ拡張を行う場合、ソフトウェアは自動的に拡張処理を行います。
BLK_SEL_1	Integer	3'b000~3'b111	3'b000	B ポートブロック選択のパラメータの設定。ポート BLKSEL の値と同じ場合にこの BSRAM が選択されます。IP Core Generator を使用してメモリ拡張を行う場合、拡張は自動的に実行されます。
RESET_MODE	String	"SYNC","ASYNC"	"SYNC"	リセットモードの構成 SYNC：同期リセット ASYNC：非同期リセット
INIT_RAM_00~ INIT_RAM_3F	Integer	SDPB:256'h0 … 0~256'h1…1	SDPB:256'h0…0 SDPX9B:288'h	BSRAM の初期化データを設定するために使用され

パラメータ名	パラメータのタイプ	値の範囲	デフォルト値	説明
		SDPX9B:288'h0 … 0~288'h1…1	0…0	ます

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。SDPB のインスタンス化を例に説明します：

Verilog でのインスタンス化：

```
SDPB bram_sdpb_0 (
        .DO({dout[31:16],dout[15:0]}),
    .CLKA(clka),
    .CEA(cea),
    .RESETA(reseta),
    .CLKB(clkb),
    .CEB(ceb),
    .RESETB(resetb),
    .OCE(oce),
    .BLKSELA({3'b000}),
    .BLKSELB({3'b000}),
    .ADA({ada[9:0], 2'b00, byte_en[1:0]}),
        .DI({{16{1'b0}},din[15:0]}),
    .ADB({adb[9:0],4'b0000})
);
defparam bram_sdpb_0.READ_MODE = 1'b1;
defparam bram_sdpb_0.BIT_WIDTH_0 = 16;
defparam bram_sdpb_0.BIT_WIDTH_1 = 16;
defparam bram_sdpb_0.BLK_SEL_0 = 3'b000;
defparam bram_sdpb_0.BLK_SEL_1 = 3'b000;
defparam bram_sdpb_0.RESET_MODE = "SYNC";
  defparam bram_sdpb_0.INIT_RAM_00 =
  256'h00A000000000000B00A000000000000B00A000000000000B00
  A000000000000B;
  defparam bram_sdpb_0.INIT_RAM_3F =
  256'h00A000000000000B00A000000000000B00A000000000000B00
  A000000000000B;
```

VHDL でのインスタンス化：

```
    COMPONENT SDPB
        GENERIC(
            BIT_WIDTH_0:integer:=16;
            BIT_WIDTH_1:integer:=16;
            READ_MODE:bit:='0';
            BLK_SEL_0:bit_vector:="000";
            BLK_SEL_1:bit_vector:="000";
            RESET_MODE:string:="SYNC";
            INIT_RAM_00:bit_vector:=X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B";
            INIT_RAM_01:bit_vector:=X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B";
            INIT_RAM_3F:bit_vector:=X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B"
        );
        PORT(
            DO:OUT std_logic_vector(31 downto 0):=conv_std_logic_vector(0,32);
            CLKA,CLKB,CEA,CEB:IN std_logic;
            OCE,RESETA,RESETB:IN std_logic;
            ADA,ADB:IN std_logic_vector(13 downto 0);
            BLKSELA:IN std_logic_vector(2 downto 0);
            BLKSELB:IN std_logic_vector(2 downto 0);
            DI:IN std_logic_vector(31 downto 0)
        );
    END COMPONENT;
  uut:SDPB
      GENERIC MAP(
            BIT_WIDTH_0=>16,
            BIT_WIDTH_1=>16,
            READ_MODE=>'0',
            BLK_SEL_0=>"000",
            BLK_SEL_1=>"000",
            RESET_MODE=>"SYNC",
            INIT_RAM_00=>X"00A000000000000B00A00
```

```
0000000000B00A000000000000B00A000000000000B",
                    INIT_RAM_01=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B",
                    INIT_RAM_3F=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B"
                    )
        PORT MAP(
            DO=>dout,
            CLKA=>clka,
            CEA=>cea,
            RESETA=>reseta,
            CLKB=>clkb,
            CEB=>ceb,
            RESETB=>resetb,
            OCE=>oce,
            BLKSELA=>blksela,
            BLKSELB=>blkselb,
            ADA=>ada,
            DI=>din,
            ADB=>adb
        );
```

3.4 ROM モード

プリミティブの紹介

pROM/pROMX9(16K/18K Block ROM)：16K/18K のブロック ROM。

機能の説明

pROM/pROMX9 はメモリ領域がそれぞれ 16K bit/18K bit である読み出し専用メモリで、2 つの読み出しモード（bypass モードと pipeline モード）をサポートします。

パラメータの READ_MODE は、出力 pipeline レジスタを有効または無効にするために使用されます。出力 pipeline レジスタを使用する場合、読み出しには追加の遅延期間が必要です。

ROM の各読み出しモードに対応する内部タイミング波形については、セミ・デュアルポート BSRAM の B ポートのタイミングである図 3-12 および図 3-13 を参照してください。

図 3-12 ROM のタイミング図（Bypass モード）

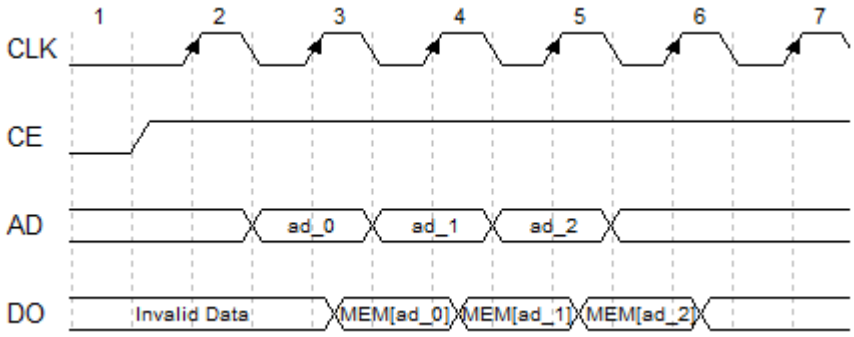
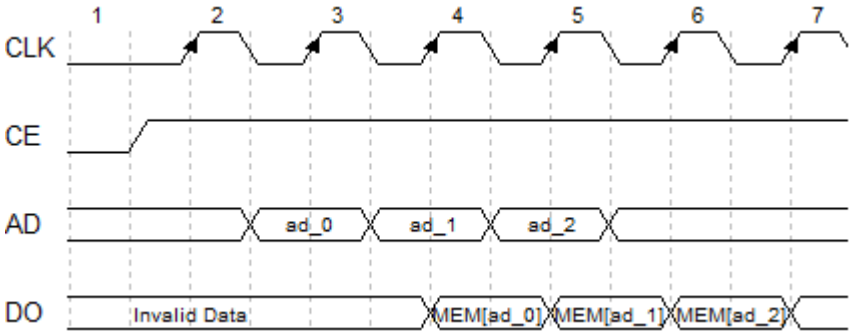


図 3-13 ROM のタイミング図（Pipeline モード）



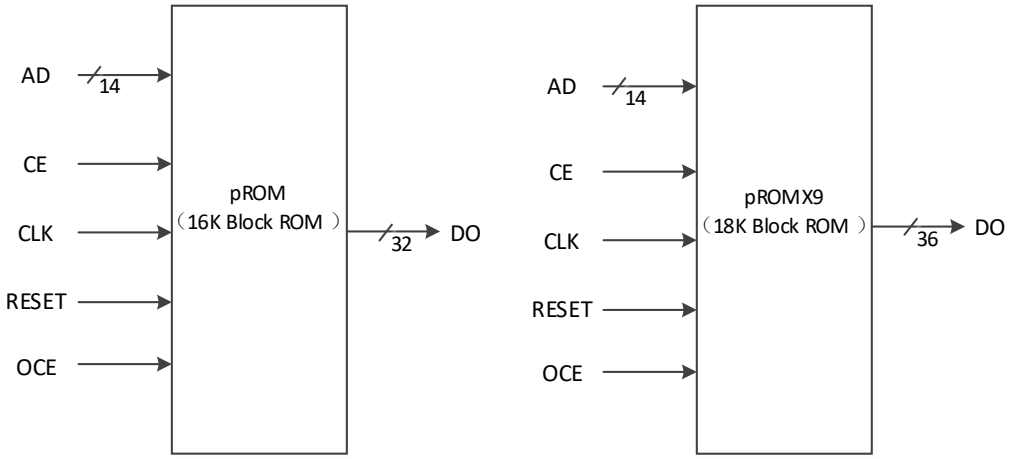
対応関係

表 3-10 pROM/pROMX9 データ幅とアドレス深さの対応関係

ROM モード	BSRAM 容量	データ幅	アドレス深さ
pROM	16Kbits	1	14
		2	13
		4	12
		8	11
		16	10
		32	9
pROMX9	18Kbits	9	11
		18	10
		36	9

ポート図

図 3-14 pROM/pROMX9 のポート図



ポートの説明

表 3-11 pROM/pROMX9 のポートの説明

ポート名	I/O	説明
DO[31:0]/DO[35:0]	出力	データ出力信号
AD[13:0]	入力	アドレス入力
CE	入力	クロックイネーブル入力、アクティブ High
CLK	入力	クロック入力
RESET	入力	リセット入力。同期リセットおよび非同期リセットをサポート、アクティブ High。RESET は、メモリ内の値をリセットするのではなく、レジスタをリセットします
OCE	入力	出力クロックイネーブル信号。pipeline モードに使用、bypass モードには無効

パラメータの説明

表 3-12 pROM/pROMX9 のパラメータの説明

パラメータ名	パラメータのタイプ	値の範囲	デフォルト値	説明
READ_MODE	Integer	1'b0,1'b1	1'b0	読み出しモード構成 1'b0:bypass モード 1'b1:pipeline モード
BIT_WIDTH	Integer	pROM:1,2,4,8,16,32 pROMX9:9,18,36	pROM:32 pROMX9:36	データ幅構成
RESET_MODE	String	"SYNC","ASYNC"	"SYNC"	リセットモードの構成 SYNC：同期リセット

パラメータ名	パラメータのタイプ	値の範囲	デフォルト値	説明
				ASYNC:非同期リセット
INIT_RAM_00~ INIT_RAM_3F	Integer	pROM:256'h0…0~256'h1…1 pROMX9:288'h0 … 0~288'h1…1	pROM:256'h0…0 pROMX9:288'h0…0	BSRAM の初期化データを設定するために使用されます

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。pROM のインスタンス化を例に説明します：

Verilog でのインスタンス化：

```
pROM bram_prom_0 (
    .DO({dout[31:8],dout[7:0]}),
    .CLK(clk),
    .OCE(oce),
    .CE(ce),
    .RESET(reset),
    .AD({ad[10:0],3'b000})
);
defparam bram_prom_0.READ_MODE = 1'b0;
defparam bram_prom_0.BIT_WIDTH = 8;
defparam bram_prom_0.RESET_MODE = "SYNC";
  defparam bram_prom_0.INIT_RAM_00 =
  256'h9C23645D0F78986FFC3E36E141541B95C19F2F7164085E63
  1A819860D8FF0000;
  defparam bram_prom_0.INIT_RAM_01 =
  256'h0000000000000000000000000000000000000000000000000
  000FFFFFFBDCF;
```

VHDL でのインスタンス化：

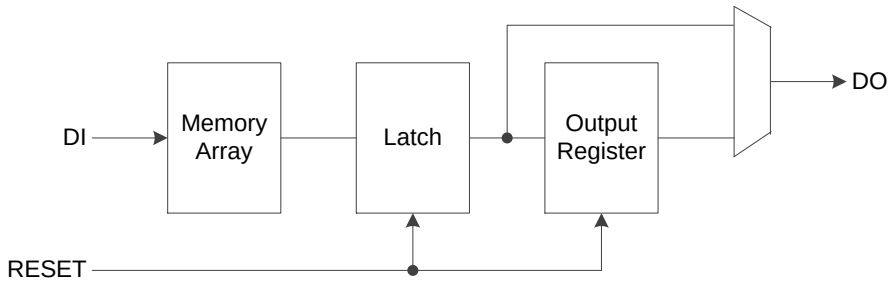
```
COMPONENT pROM
    GENERIC(
        BIT_WIDTH:integer:=1;
        READ_MODE:bit:='0';
        RESET_MODE:string:="SYNC";
        INIT_RAM_00:bit_vector:=X"9C23645D0F78986FF
C3E36E141541B95C19F2F7164085E631A819860D8FF0000";
```

```
                INIT_RAM_01:bit_vector:=X"000000000000000000
0000000000000000000000000000000000000FFFFFFBDCF"
        );
    PORT(
                DO:OUT std_logic_vector(31 downto 0):=conv_std
_logic_vector(0,32);
                CLK,CE,OCE,RESET:IN std_logic;
                AD:IN std_logic_vector(13 downto 0)
        );
  END COMPONENT;
  uut:pROM
    GENERIC MAP(
                BIT_WIDTH=>1,
                READ_MODE=>'0',
                RESET_MODE=>"SYNC",
                INIT_RAM_00=>X"9C23645D0F78986FFC3E36
E141541B95C19F2F7164085E631A819860D8FF0000",
                INIT_RAM_01=>X"000000000000000000000000000
0000000000000000000000000000000FFFFFFBDCF "
    )
    PORT MAP(
            DO=>do,
            AD=>ad,
            CLK=>clk,
            CE=>ce,
            OCE=>oce,
            RESET=>reset
        );
```

4 BSRAM 出力リセット

出力モジュールは RESET 信号をサポートし、リセットされると、0 を出力することになります。そのブロック図は図 4-1 に示す通りです。

図 4-1 出力リセットのブロック図



RESET 信号が有効な場合（アクティブ High）、出力ポートは 0 を出力します。

RESET は同期リセット及び非同期リセットをサポートします。ユーザーが直接プリミティブを呼び出しする際、パラメータ RESET_MODE を設定する必要があります。IP Core Generator を使用する場合、GUI でリセットモードを選択できます。詳細については 6 IP の呼び出しを参照してください。

RESET 信号はラッチ及び出力レジスタをリセットします。そのため、RESET 信号を有効にした場合、ユーザーがレジスタ出力モードまたはバイパス出力モードを使用しているかに関わらず、出力はすべて 0 になります。

注記：

書き込みの際、RESET 信号は 0（無効状態）でなければなりません。

図 4-2、図 4-3、図 4-4、及び図 4-5 は各モードにおけるリセットタイミング図です。そのうち、DO_RAM はメモリアレイのデータであり、DO は出力ポートのデータです。

レジスタ出力モード：

- 同期リセットが有効な場合、CLK の立ち上がりエッジで DO がリセッ

トされます。

- 非同期リセットが有効な場合、DO はリセットされます。
- リセットが無効で OCE が有効な場合、DO は "DO_RAM"を出力します。
- リセットが無効で OCE が無効な場合、DO は前の出力データを保持します。

バイパス出力モード：

- 同期リセットが有効な場合、CLK の立ち上がりエッジで DO がリセットされます。
- 非同期リセットが有効な場合、DO はリセットされます。
- リセットが無効な場合、OCE が有効か無効かにかかわらず、DO は "DO_RAM"を出力します。

図 4-2 同期リセットのタイミング図（Pipeline モード）

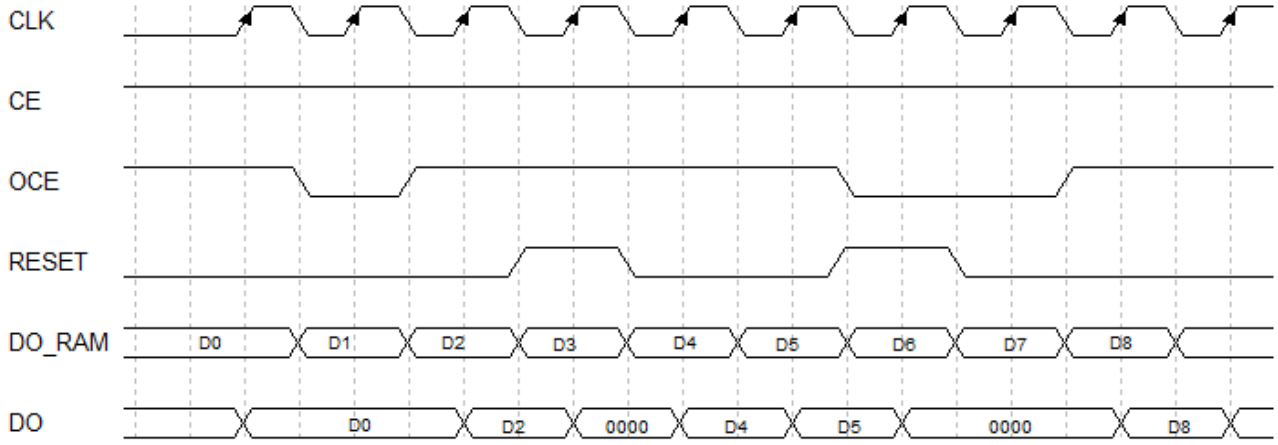


図 4-3 同期リセットのタイミング図（Bypass モード）

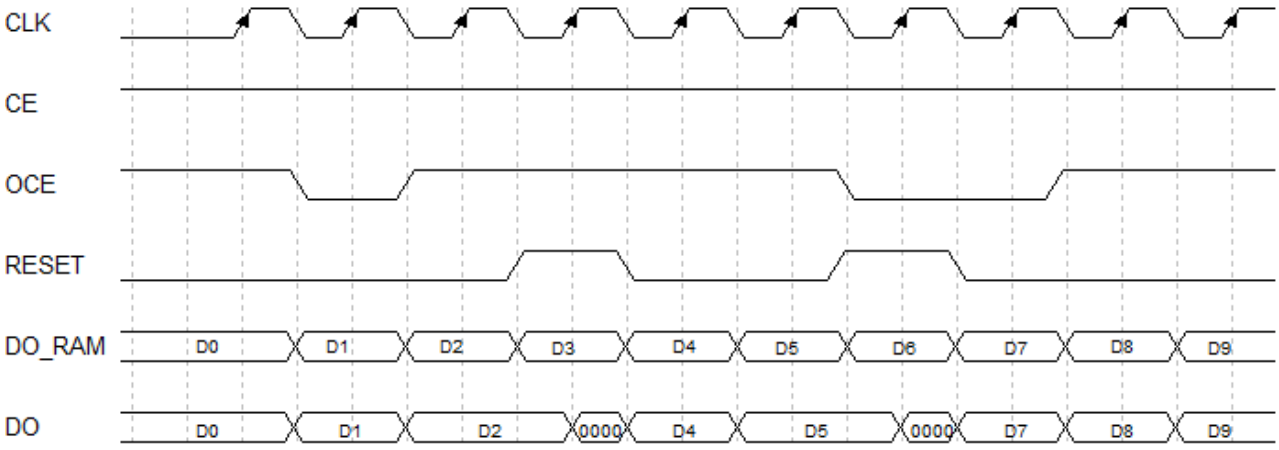


図 4-4 非同期リセットのタイミング図（Pipeline モード）

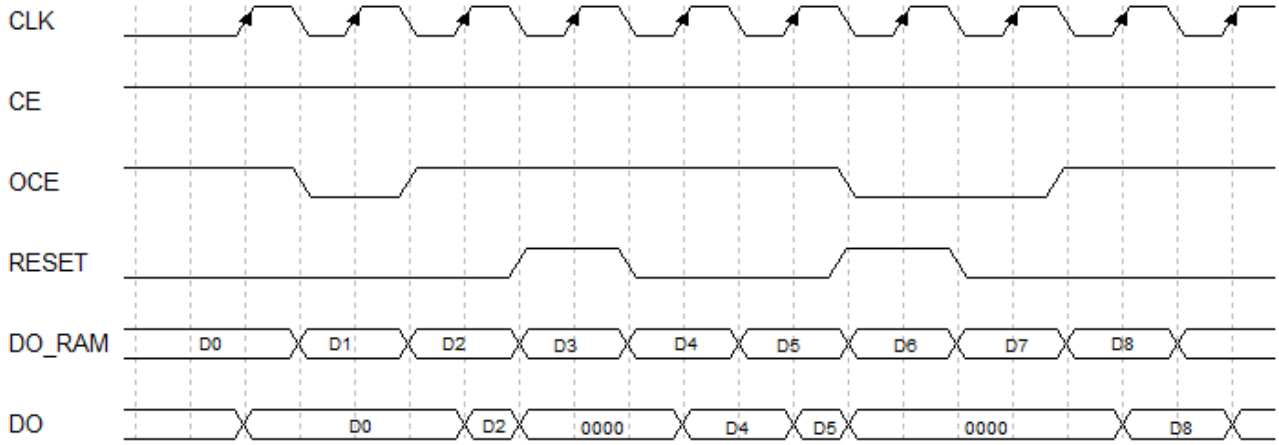
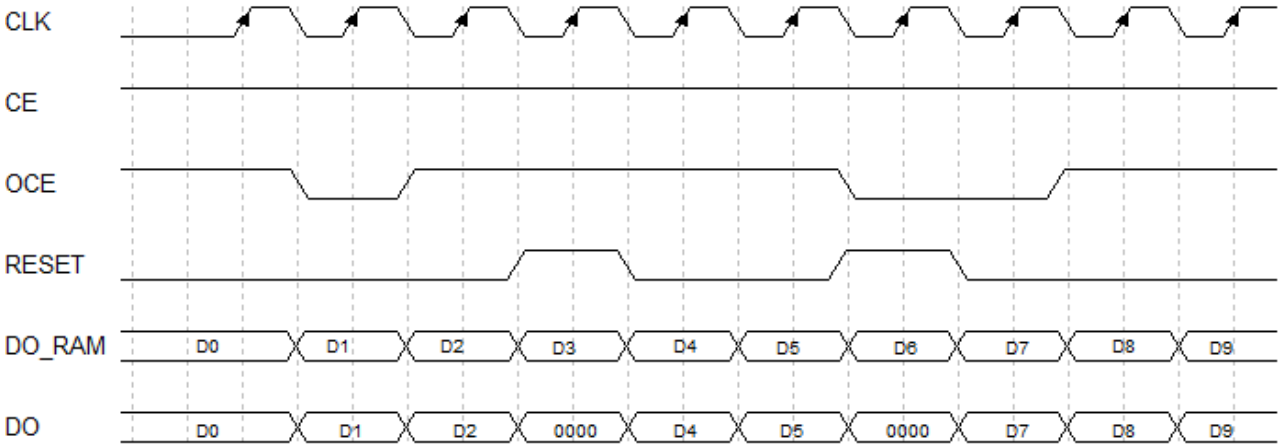


図 4-5 非同期リセットのタイミング図（Bypass モード）



5 SSRAM プリミティブ

Shadow SRAM は分散 SRAM で、シングルポートモード、セミ・デュアルポートモード、および読み出し専用モードに構成できます（表 5-1）。

表 5-1 SSRAM モード

プリミティブ	説明
RAM16S1	アドレス深さが 16、データ幅が 1 のシングルポート SSRAM
RAM16S2	アドレス深さが 16、データ幅が 2 のシングルポート SSRAM
RAM16S4	アドレス深さが 16、データ幅が 4 のシングルポート SSRAM
RAM16SDP1	アドレス深さが 16、データ幅が 1 のセミ・デュアルポート SSRAM
RAM16SDP2	アドレス深さが 16、データ幅が 2 のセミ・デュアルポート SSRAM
RAM16SDP4	アドレス深さが 16、データ幅が 4 のセミ・デュアルポート SSRAM
ROM16	アドレス深さが 16、データ幅が 1 の ROM

注記：

GW1N-1、GW1N-1S、GW1N-4、GW1N-4B、GW1NR-1、GW1NR-4、GW1NR-4B、GW1NRF-4B、GW1NS-4、GW1NS-4C、GW1NSER-4C、GW1NSR-4、GW1NSR-4C、GW1N-4D、GW1NR-4D デバイスは SSRAM をサポートしません。

5.1 RAM16S1

プリミティブの紹介

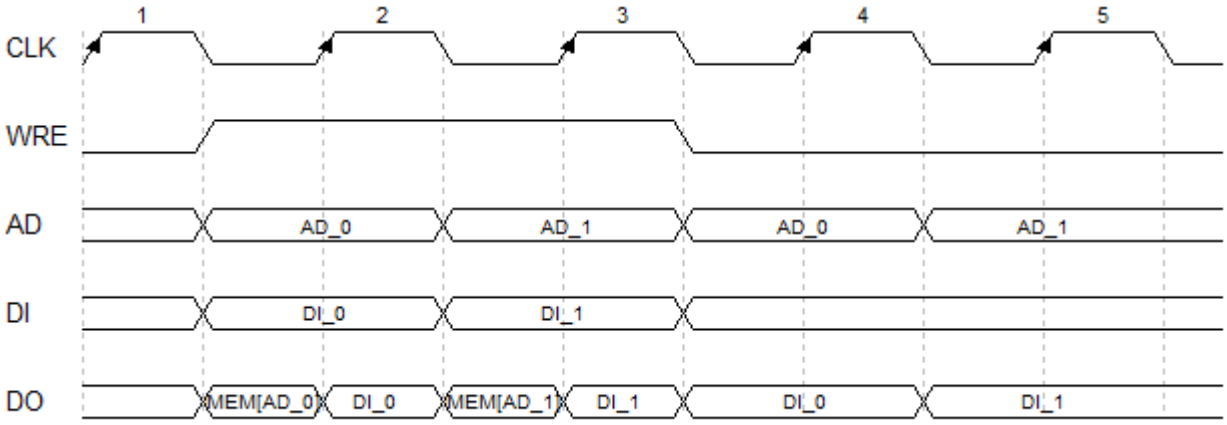
RAM16S1(16-Deep by 1-Wide Single-port SSRAM)はアドレス深さが 16、データ幅が 1 のシングルポート SSRAM です。

機能の説明

RAM16S1 はデータ幅が 1 のシングルポート SSRAM で、その読み出しアドレスと書き込みアドレスは同じです。WRE が High のときに書き込みが実行されます。この場合、CLK の立ち上がりエッジでデータがメモリにロードされます。読み出し操作では、アドレスに対応するデータが出力されます。つまり、CFU の LUT によって構成される SSRAM は、同期的に書き込まれ、非同期的に読み出されます。ただし、必要な場合は、各 LUT に関連付けられたレジスタを使用して同期読み出しも実装できます。その

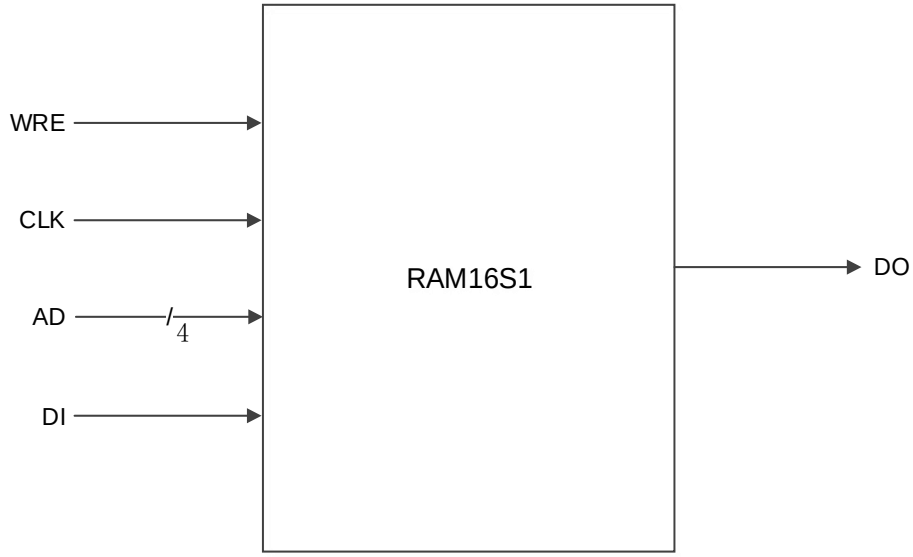
タイミング図を図 5-1 に示します。

図 5-1 RAM16S1 モードのタイミング図



ポート図

図 5-2 RAM16S1 のポート図



ポートの説明

表 5-2 RAM16S1 のポートの説明

ポート	I/O	説明
DI	入力	データ入力
CLK	入力	クロック入力
WRE	入力	書き込みイネーブル入力
AD[3:0]	入力	アドレス入力
DO	出力	データ出力

パラメータの説明

表 5-3 RAM16S1 のパラメータの説明

パラメータ	範囲	デフォルト	説明
INIT_0	16'h0000~16'hffff	16'h0000	RAM16S1 の初期値

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。

Verilog でのインスタンス化：

```
RAM16S1 instName(
        .DI(DI),
        .WRE(WRE),
        .CLK(CLK),
        .AD(AD[3:0]),
        .DO(DOUT)
);
defparam instName.INIT_0=16'h1100;
```

VHDL でのインスタンス化：

```
COMPONENT RAM16S1
        GENERIC (INIT:bit_vector:=X"0000");
        PORT(
                DO:OUT std_logic;
                DI:IN std_logic;
                CLK:IN std_logic;
                WRE:IN std_logic;
                AD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:RAM16S1
      GENERIC MAP(INIT=>X"0000")
      PORT MAP (
            DO=>DOUT,
            DI=>DI,
            CLK=>CLK,
            WRE=>WRE,
```

```
        AD=>AD
    );
```

5.2 RAM16S2

プリミティブの紹介

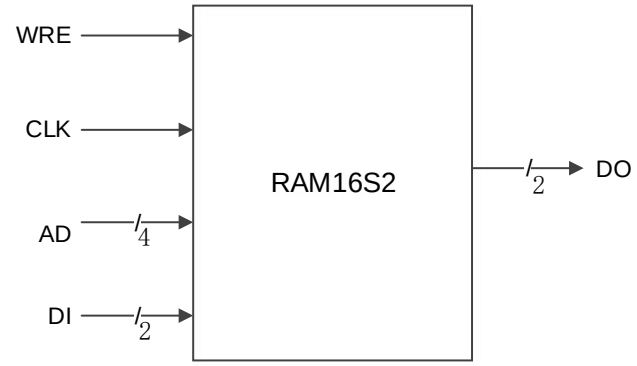
RAM16S2(16-Deep by 2-Wide Single-port SSRAM)はアドレス深さが 16、データ幅が 2 のシングルポート SSRAM です。

機能の説明

RAM16S2 はデータ幅が 2 のシングルポート SSRAM で、その読み出しアドレスと書き込みアドレスは同じです。WRE が High のときに書き込みが実行されます。この場合、CLK の立ち上がりエッジでデータがメモリにロードされます。読み出し操作では、アドレスに対応するデータが出力されます。つまり、CFU の LUT によって構成される SSRAM は、同期的に書き込まれ、非同期的に読み出されます。ただし、必要な場合は、各 LUT に関連付けられたレジスタを使用して同期読み出しも実装できます。そのタイミング図を図 5-1 に示します。

ポート図

図 5-3 RAM16S2 のポート図



ポートの説明

表 5-4 RAM16S2 のポート図

ポート	I/O	説明
DI[1:0]	入力	データ入力
CLK	入力	クロック入力
WRE	入力	書き込みイネーブル入力
AD[3:0]	入力	アドレス入力
DO[1:0]	出力	データ出力

パラメータの説明

表 5-5 RAM16S2 のパラメータの説明

パラメータ	範囲	デフォルト	説明
INIT_0~ INIT_1	16'h0000~16'hffff	16'h0000	RAM16S2 の初期値

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。

Verilog でのインスタンス化：

```
RAM16S2 instName(
        .DI(DI[1:0]),
        .WRE(WRE),
        .CLK(CLK),
        .AD(AD[3:0]),
        .DO(DOUT[1:0])
);
defparam instName.INIT_0=16'h0790;
defparam instName.INIT_1=16'h0f00;
```

VHDL でのインスタンス化：

```
COMPONENT RAM16S2
        GENERIC (INIT_0:bit_vector:=X"0000";
                        INIT_1:bit_vector:=X"0000"
            );
        PORT(
                DO:OUT std_logic_vector(1 downto 0);
                DI:IN std_logic_vector(1 downto 0);
                CLK:IN std_logic;
                WRE:IN std_logic;
                AD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:RAM16S2
        GENERIC MAP(INIT_0=>X"0000",
                            INIT_1=>X"0000"
            )
        PORT MAP (
```

```
        DO=>DOUT,
        DI=>DI,
        CLK=>CLK,
        WRE=>WRE,
        AD=>AD
    );
```

5.3 RAM16S4

プリミティブの紹介

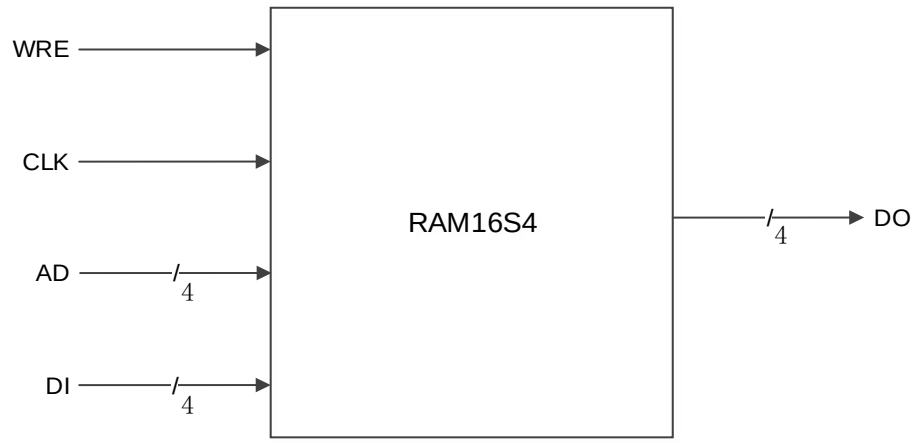
RAM16S4(16-Deep by 4-Wide Single-port SSRAM)はアドレス深さが 16、データ幅が 4 のシングルポート SSRAM です。

機能の説明

RAM16S4 はデータ幅が 4 のシングルポート SSRAM で、その読み出しアドレスと書き込みアドレスは同じです。WRE が High のときに書き込みが実行されます。この場合、CLK の立ち上がりエッジでデータがメモリにロードされます。読み出し操作では、アドレスに対応するデータが出力されます。つまり、CFU の LUT によって構成される SSRAM は、同期的に書き込まれ、非同期的に読み出されます。ただし、必要な場合は、各 LUT に関連付けられたレジスタを使用して同期読み出しも実装できます。そのタイミング図を図 5-1 に示します。

ポート図

図 5-4 RAM16S4 のポート図



ポートの説明

表 5-6 RAM16S4 のポート図

ポート	I/O	説明
DI[3:0]	入力	データ入力

ポート	I/O	説明
CLK	入力	クロック入力
WRE	入力	書き込みイネーブル入力
AD[3:0]	入力	アドレス入力
DO[3:0]	出力	データ出力

パラメータの説明

表 5-7 RAM16S4 のパラメータの説明

パラメータ	範囲	デフォルト	説明
INIT_0~ INIT_3	16'h0000~16'hffff	16'h0000	RAM16S4 の初期値

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。

Verilog でのインスタンス化：

```
RAM16S4 instName(
        .DI(DI[3:0]),
        .WRE(WRE),
        .CLK(CLK),
        .AD(AD[3:0]),
        .DO(DOUT[3:0])
);
defparam instName.INIT_0=16'h0450;
defparam instName.INIT_1=16'h1ac3;
defparam instName.INIT_2=16'h1240;
defparam instName.INIT_3=16'h045c;
```

VHDL でのインスタンス化：

```
COMPONENT RAM16S4
    GENERIC (INIT_0:bit_vector:=X"0000";
                INIT_1:bit_vector:=X"0000";
                INIT_2:bit_vector:=X"0000";
                INIT_3:bit_vector:=X"0000"
        );
    PORT(
        DO:OUT std_logic_vector(3 downto 0);
```

```
            DI:IN std_logic_vector(3 downto 0);
            CLK:IN std_logic;
            WRE:IN std_logic;
            AD:IN std_logic_vector(3 downto 0)
        );
    END COMPONENT;
    uut:RAM16S4
        GENERIC MAP(INIT_0=>X"0000",
                    INIT_1=>X"0000",
                    INIT_2=>X"0000",
                    INIT_3=>X"0000"
            )
        PORT MAP (
            DO=>DOUT,
            DI=>DI,
            CLK=>CLK,
            WRE=>WRE,
            AD=>AD
        );
```

5.4 RAM16SDP1

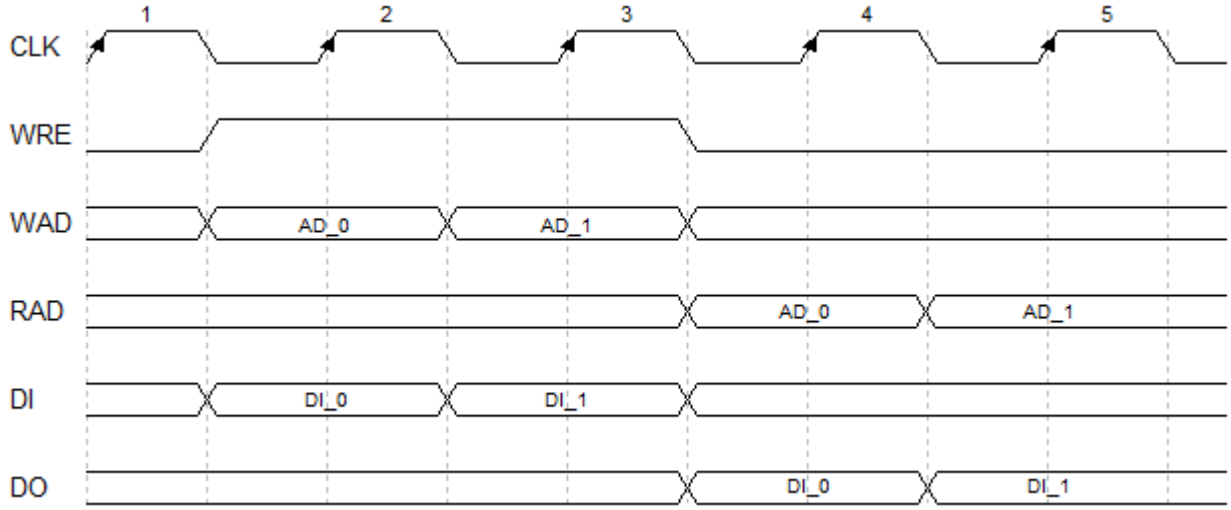
プリミティブの紹介

RAM16SDP1(16-Deep by 1-Wide Semi Dual-port SSRAM)はアドレス深さが 16、データ幅が 1 のセミ・デュアルポート SSRAM です。

機能の説明

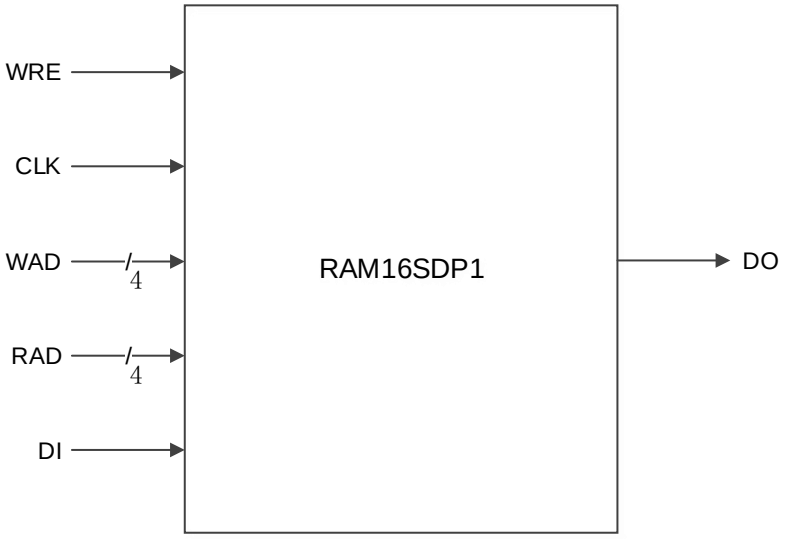
RAM16SDP1 には、書き込みアドレス WAD および読み出しアドレス RAD があります。この 2 つのアドレスポートは非同期です。WRE が High のときに書き込みが実行されます。この場合、CLK の立ち上がりエッジでデータがメモリにロードされます。読み出し操作では、読み出しアドレスに対応するデータが出力されます。そのタイミング図を図 5-5 に示します。

図 5-5 RAM16SDP1 モードのタイミング図



ポート図

図 5-6 RAM16SDP1 のポート図



ポートの説明

表 5-8 RAM16SDP1 のポート図

ポート	I/O	説明
DI	入力	データ入力
CLK	入力	クロック入力
WRE	入力	書き込みイネーブル入力
WAD[3:0]	入力	書き込みアドレス信号
RAD[3:0]	入力	読み出しアドレス信号
DO	出力	データ出力

パラメータの説明

表 5-9 RAM16SDP1 のパラメータの説明

パラメータ	範囲	デフォルト	説明
INIT_0	16'h0000~16'hffff	16'h0000	RAM16SDP1 の初期値

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。

Verilog でのインスタンス化：

```
RAM16SDP1 instName(
        .DI(DI),
        .WRE(WRE),
        .CLK(CLK),
        .WAD(WAD[3:0]),
        .RAD(RAD[3:0]),
        .DO(DOUT)
);
defparam instName.INIT_0=16'h0100;
```

VHDL でのインスタンス化：

```
COMPONENT RAM16SDP1
      GENERIC (INIT_0:bit_vector:=X"0000");
      PORT(
            DO:OUT std_logic;
            DI:IN std_logic;
            CLK:IN std_logic;
            WRE:IN std_logic;
            WAD:IN std_logic_vector(3 downto 0);
            RAD:IN std_logic_vector(3 downto 0)
      );
END COMPONENT;
uut:RAM16SDP1
      GENERIC MAP(INIT_0=>X"0000")
      PORT MAP (
            DO=>DOUT,
```

```
        DI=>DI,
        CLK=>CLK,
        WRE=>WRE,
      WAD=>WAD,
        RAD=>RAD
);
```

5.5 RAM16SDP2

プリミティブの紹介

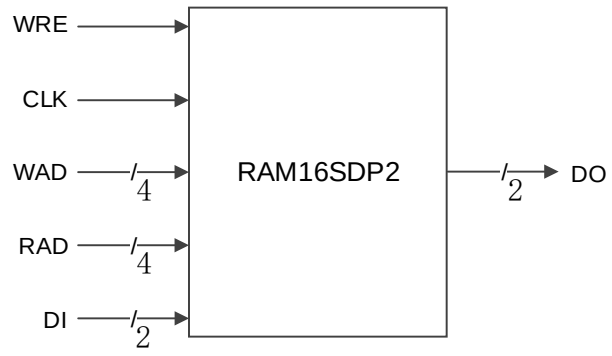
RAM16SDP2(16-Deep by 2-Wide Semi Dual-port SSRAM)はアドレス深さが 16、データ幅が 2 のセミ・デュアルポート SSRAM です。

機能の説明

RAM16SDP2 には、書き込みアドレス WAD および読み出しアドレス RAD があります。この 2 つのアドレスポートは非同期です。WRE が High のときに書き込みが実行されます。この場合、CLK の立ち上がりエッジでデータがメモリにロードされます。読み出し操作では、読み出しアドレスに対応するデータが出力されます。そのタイミング図を図 5-5 に示します。

ポート図

図 5-7 RAM16SDP2 のポート図



ポートの説明

表 5-10 RAM16SDP2 のポート図

ポート	I/O	説明
DI[1:0]	入力	データ入力
CLK	入力	クロック入力
WRE	入力	書き込みイネーブル入力
WAD[3:0]	入力	書き込みアドレス信号
RAD[3:0]	入力	読み出しアドレス信号
DO[1:0]	出力	データ出力

パラメータの説明

表 5-11 RAM16SDP2 のパラメータの説明

パラメータ	範囲	デフォルト	説明
INIT_0~ INIT_1	16'h0000~16'hffff	16'h0000	RAM16SDP2 の初期値

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。

Verilog でのインスタンス化：

```
RAM16SDP2 instName(
        .DI(DI[1:0]),
        .WRE(WRE),
        .CLK(CLK),
        .WAD(WAD[3:0]),
        .RAD(RAD[3:0]),
        .DO(DOUT[1:0])
);
defparam instName.INIT_0=16'h5600;
defparam instName.INIT_1=16'h0af0;
```

VHDL でのインスタンス化：

```
COMPONENT RAM16SDP2
        GENERIC (INIT_0:bit_vector:=X"0000";
                        INIT_1:bit_vector:=X"0000"
            );
        PORT(
                DO:OUT std_logic_vector(1 downto 0);
                DI:IN std_logic_vector(1 downto 0);
                CLK:IN std_logic;
                WRE:IN std_logic;
                WAD:IN std_logic_vector(3 downto 0);
                RAD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:RAM16SDP2
```

```
    GENERIC MAP(INIT_0=>X"0000",
                INIT_1=>X"0000"
        )
    PORT MAP (
        DO=>DOUT,
        DI=>DI,
        CLK=>CLK,
        WRE=>WRE,
      WAD=>WAD,
        RAD=>RAD
);
```

5.6 RAM16SDP4

プリミティブの紹介

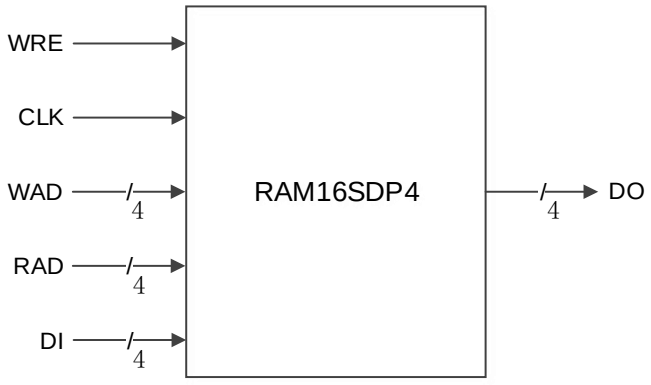
RAM16SDP4(16-Deep by 4-Wide Semi Dual-port SSRAM)はアドレス深さが 16、データ幅が 4 のセミ・デュアルポート SSRAM です。

機能の説明

RAM16SDP4 には、書き込みアドレス WAD および読み出しアドレス RAD があります。この 2 つのアドレスポートは非同期です。WRE が High のときに書き込みが実行されます。この場合、CLK の立ち上がりエッジでデータがメモリにロードされます。読み出し操作では、読み出しアドレスに対応するデータが出力されます。そのタイミング図を図 5-5 に示します。

ポート図

図 5-8 RAM16SDP4 のポート図



ポートの説明

表 5-12 RAM16SDP4 のポート図

ポート	I/O	説明
DI[3:0]	入力	データ入力

ポート	I/O	説明
CLK	入力	クロック入力
WRE	入力	書き込みイネーブル入力
WAD[3:0]	入力	書き込みアドレス信号
RAD[3:0]	入力	読み出しアドレス信号
DO[3:0]	出力	データ出力

パラメータの説明

表 5-13 RAM16SDP4 のパラメータの説明

パラメータ	範囲	デフォルト	説明
INIT_0~ INIT_3	16'h0000~16'hffff	16'h0000	LUT4 の初期値

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。

Verilog でのインスタンス化：

```
RAM16SDP4 instName(
        .DI(DI[3:0]),
        .WRE(WRE),
        .CLK(CLK),
        .WAD(WAD[3:0]),
        .RAD(RAD[3:0]),
        .DO(DOUT[3:0])
);
defparam instName.INIT_0=16'h0340;
defparam instName.INIT_1=16'h9065;
defparam instName.INIT_2=16'hac12;
defparam instName.INIT_3=16'h034c;
```

VHDL でのインスタンス化：

```
COMPONENT RAM16SDP2
    GENERIC (INIT_0:bit_vector:=X"0000";
                INIT_1:bit_vector:=X"0000";
                INIT_2:bit_vector:=X"0000";
                INIT_3:bit_vector:=X"0000";
        );
```

```
        PORT(
                DO:OUT std_logic_vector(3 downto 0);
                DI:IN std_logic_vector(3 downto 0);
                CLK:IN std_logic;
                WRE:IN std_logic;
                WAD:IN std_logic_vector(3 downto 0);
                RAD:IN std_logic_vector(3 downto 0)
        );
    END COMPONENT;
    uut:RAM16SDP2
        GENERIC MAP(INIT_0=>X"0000",
                                INIT_1=>X"0000",
                                INIT_2=>X"0000",
                                INIT_3=>X"0000"
            )
        PORT MAP (
                DO=>DOUT,
                DI=>DI,
                CLK=>CLK,
                WRE=>WRE,
               WAD=>WAD,
                RAD=>RAD
        );
```

5.7 ROM16

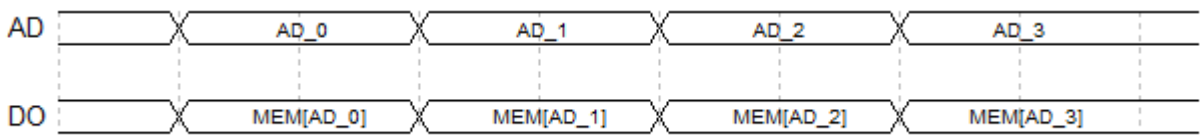
プリミティブの紹介

ROM16 はアドレス深さが 16、データ幅が 1 の読み出し専用メモリで、メモリの内容は INIT によって初期化されます。

機能の説明

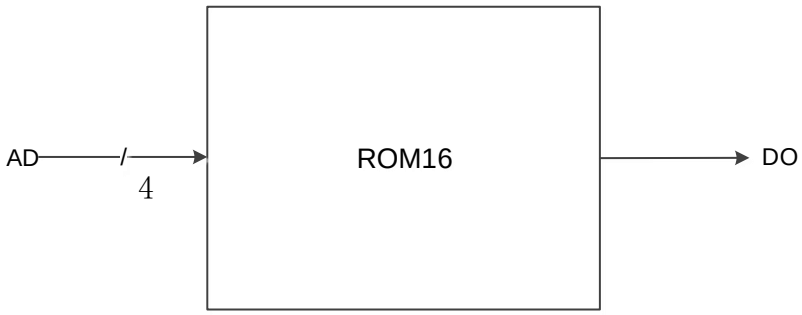
ROM16 の読み出し操作では、アドレスに対応するデータが出力されます。そのタイミング図を図 5-9 に示します。

図 5-9 ROM16 モードのタイミング図



ポート図

図 5-10 ROM16 のポート図



ポートの説明

表 5-14 ROM16 のポート図

ポート	I/O	説明
AD[3:0]	入力	アドレス入力
DO	出力	データ出力

パラメータの説明

表 5-15 ROM16 のパラメータの説明

パラメータ	範囲	デフォルト	説明
INIT_0	16'h0000~16'hffff	16'h0000	ROM16 の初期値

プリミティブのインスタンス化

プリミティブを直接インスタンス化するか、IP Core Generator で生成できます。詳しくは、6 IP の呼び出しを参照してください。

Verilog でのインスタンス化：

```
ROM16 instName (
        .AD(AD[3:0]),
        .DO(DOUT)
);
defparam instName.INIT_0=16'hfc00;
```

VHDL でのインスタンス化：

```
COMPONENT ROM16
        GENERIC (INIT:bit_vector:=X"0000");
        PORT(
                DO:OUT std_logic;
```

```
            AD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:ROM16
    GENERIC MAP(INIT=>X"0000")
    PORT MAP (
        DO=>DOUT,
        AD=>AD
    );
```

6 IP の呼び出し

Gowin ソフトウェアの IP Core Generator は、IP コアの呼び出しをサポートします。ユーザーは GUI でデータ幅、アドレス深さ、書き込みモード、及び読み出しモードを設定して IP モジュールを生成することができます。それ以外にも、BSRAM、SSRAM を実装する方法は 2 つあります。1 つは、Gowin ソフトウェアのプリミティブライブラリのファイルを呼び出し、ポート及びパラメータを設定して IP モジュールを生成する方法です。もう 1 つは、合成時、合成ツールで自動的に BSRAM、SSRAM モードに合成する方法です。

IP Core Generator では、BSRAM モジュールはシングルポートモード、セミ・デュアルポートモード、デュアルポートモード、および読み出し専用モードをサポートし、SSRAM モジュールはシングルポートモード、セミ・デュアルポートモード、および読み出し専用モードをサポートします。以下では、デュアルポートモードの BSRAM とシングルポートモードの SSRAM の呼び出しを紹介します。

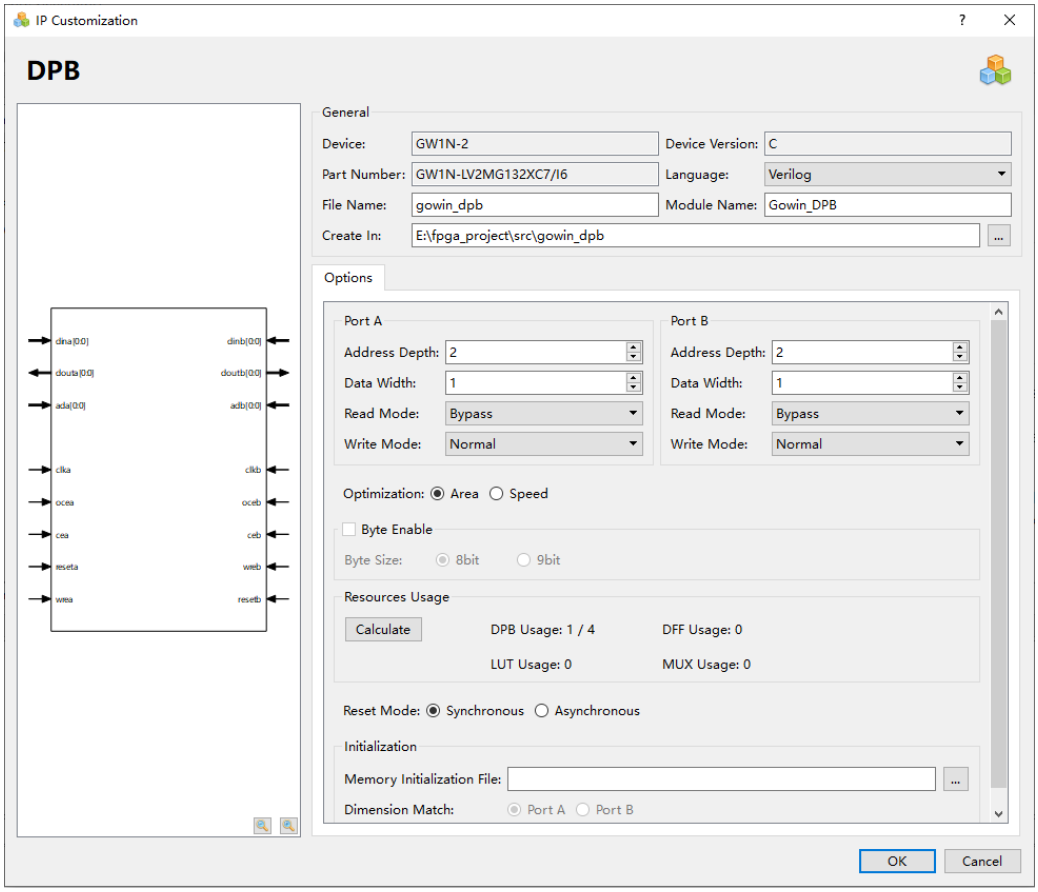
6.1 デュアルポートモードの BSRAM

デュアルポートモードの BSRAM は、プリミティブの DPB および DPX9B により実装できます。IP Core Generator のインターフェースで"DPB"をクリックすると、右側に DPB の概要が表示されます。

IP の構成

IP Core Generator インターフェースで"DPB"をダブルクリックすると、DPB の"IP Customization"ウィンドウがポップアップします。このウィンドウには General 構成タブ、Options 構成タブ、およびポート図があります（図 6-1）。

図 6-1 DPB の IP Customization ウィンドウの構造



1. General 構成タブ。General 構成タブは、IP ファイルの構成に使用されます。

- Device：対象デバイス。
- Device Version：デバイスのバージョン。
- Part Number：パーツ番号。
- Language : IP を実現するハードウェア記述言語。右側のドロップダウンリストからターゲット言語（Verilog または VHDL）を選択します。
- Module Name : 生成される IP ファイルのモジュール名。右側のテキストボックスで編集できます。Module Name をプリミティブ名と同じにすることはできません。同じ場合、エラーメッセージがポップアップします。
- File Name：生成される IP ファイルのファイル名。右側のテキストボックスで再編集できます。
- Create In：生成される IP ファイルのパス。右側のテキストボックスでパスを直接編集するか、テキストボックスの右側にある選択ボタンを使用してパスを選択できます。

2. Options 構成タブ。Options 構成タブは IP のカスタマイズに使用されます。図 6-1 に示すように、A ポートと B ポートがあります。

- Data Width & Address Depth：アドレス深さ（Address Depth）とデータ幅（Data Width）を構成します。構成されたアドレス深さとデータ幅を 1 つのモジュールで実装できない場合、IP Core は複数モジュールの組み合わせで実現します。
- Resource Usage：現在の構成で使用される Block Ram、DFF、LUT、MUX の数を計算し、表示します。
- Read/Write Mode：読み出し/書き込みモードを構成します。DPB は以下のモードをサポートします。
  - 2 つの読み出しモード：Bypass と Pipeline。
  - 3 つの書き込みモード:Normal、Write-Through、Read-before-Write。
- Reset Mode：リセットモード（同期モード"Synchronous"または非同期モード"Asynchronous"）を選択します。
- Initialization：初期値を構成します。初期値は、バイナリ、16 進数、またはアドレス付き 16 進数の形式で初期化ファイルに書き込まれます。"Memory Initialization File"で選択される初期化ファイルは手動で入力するか、Gowin ソフトウェアの"File > New > Memory Initialization File"をクリックすることにより生成できます。生成方法の詳細は『Gowin ソフトウェア ユーザーガイド(SUG100)』を参照してください。初期化ファイルの形式については 7 初期化ファイルを参照してください。

  注記：

  - Options 構成タブでは、DPB の Port A と Port B のアドレス深さ、データ幅、および読み出し/書き込みモードを個別に構成できます。
  - DPB の Port A と Port B は同じ BSRAM に対して読み出しと書き込みを行うため、Port A と Port B の Address Depth*Data Width は同じでなければなりません。
  - Options 構成の初期化ファイル（Memory initialization File）内にあるデータの幅は Dimension Match で選択した Port のデータ幅と一致しなければなりません。
  - Port A と Port B の Address Depth*Data Width の結果が一致しない場合、Error メッセージがポップアップします。
  - データ幅が一致しない場合、生成される DPB インスタンスの Init 値はデフォルトで 0 となり、そして Output ウィンドウで以下のメッセージがポップアップします：Error (MG2105): Initial values' width is unequal to user's width。

3. ポート図

- ポート図は、現在の IP Core の構成結果を表示し、入力・出力ポートのビット幅は Options 構成に従ってリアルタイムで更新されます（図 6-1）。
- Options 構成の Port A と Port B アドレス深さ"Address Depth"の構成はアドレスのビット幅に影響し、データ幅"Data Width"の構成は入力データと出力データのビット幅に影響します。

生成されるファイル

IP の構成が完了したら、構成ファイルの"File Name"によって命名された 3 つのファイルが生成されます：

- "gowin_dpb.v"は完全な verilog モジュールです。
- gowin_dpb_tmp.v は IP のテンプレートファイルです。
- "gowin_dpb.ipc"は IP の構成ファイルです。

注記：

VHDL が設計の言語として選択されている場合、生成される最初の 2 つのファイル名のサフィックスは.vhd になります。

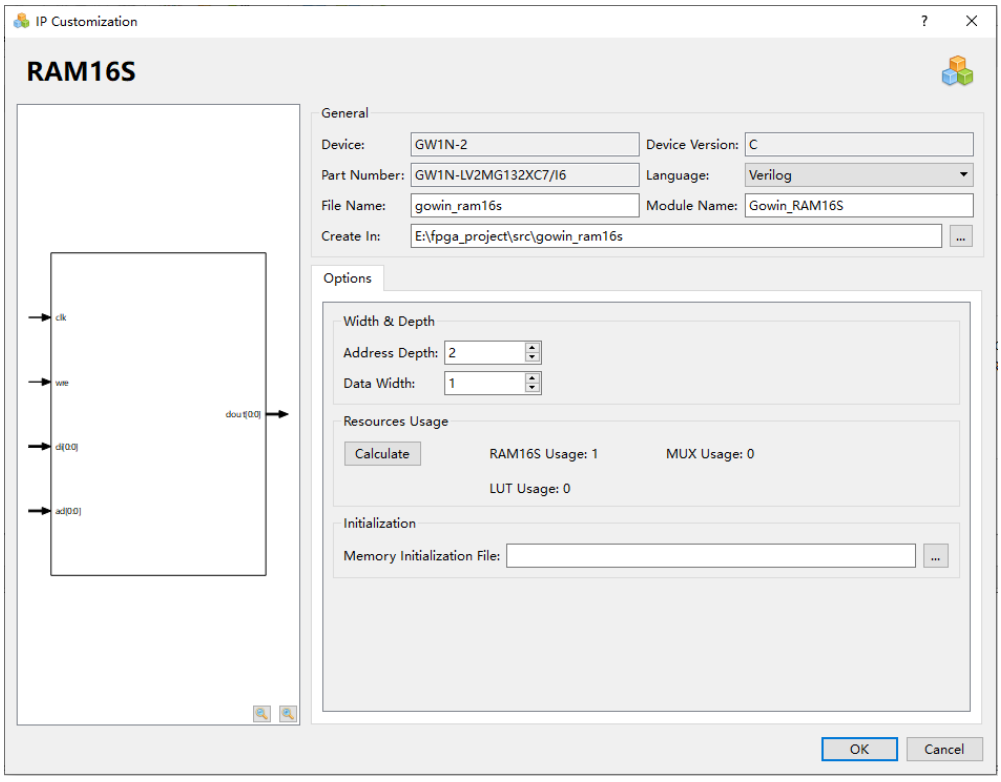
6.2 シングルポートモードの SSRAM

RAM16S（シングルポートモードの SSRAM）は、プリミティブの RAM16S1、RAM16S2、および RAM16S4 により実装できます。IP Core Generator のインターフェースで RAM16S をクリックすると、右側に ALU54 の概要が表示されます。

IP の構成

IP Core Generator インターフェースで RAM16S をダブルクリックすると、RAM16S の"IP Customization"ウィンドウがポップアップします。このウィンドウには General 構成タブ、Options 構成タブ、およびポート図があります（図 6-2）。

図 6-2 RAM16S の IP Customization ウィンドウの構造



1. General 構成タブ。General 構成タブは、IP ファイルの構成に使用されます。RAM16S の General 構成タブの使用はデュアルポートモードの BSRAM と同様です。詳細については、6.1 デュアルポートモードの BSRAM を参照してください。

2. Options 構成タブ。Options 構成タブは IP のカスタマイズに使用されます。Options 構成タブを図 6-2 に示します。RAM16S の Options 構成タブの使用はデュアルポートモードの BSRAM と同様です。詳細については、6.1 デュアルポートモードの BSRAM を参照してください。
3. ポート図
   - ポート図は、現在の IP Core の構成結果を表示し、入力・出力ポートのビット幅は Options 構成に従ってリアルタイムで更新されます（図 6-2）。
   - Options 構成のアドレス深さ"Address Depth"の構成はアドレスのビット幅に影響し、データ幅"Data Width"の構成は入力データと出力データのビット幅に影響します。

生成されるファイル

IP の構成が完了したら、構成ファイルの"File Name"によって命名された 3 つのファイルが生成されます：

- "gowin_ram16s.v"は完全な verilog モジュールです。
- gowin_ram16s_tmp.v は IP のテンプレートファイルです。
- "gowin_ram16s.ipc"は IP の構成ファイルです。

注記：

VHDL が設計の言語として選択されている場合、生成される最初の 2 つのファイル名のサフィックスは.vhd になります。

7 初期化ファイル

BSRAM および SSRAM では、メモリの各ビットを 0 または 1 に初期化できます。初期値は、バイナリ、16 進数、またはアドレス付き 16 進数の形式で初期化ファイルに書き込まれます。

7.1 バイナリ形式（Bin File）

Bin ファイルはバイナリ数 0 と 1 から成るテキストファイルです。行の数はメモリのアドレス深さ、列の数はメモリのデータ幅を表します。

```
#File_format=Bin
#Address_depth=16
#Data_width=32
00001100000100000000100100010000
10000000010010000100000001000000
01000000100000001000000010000000
00100000100001001100000011000000
```

7.2 16 進数形式（Hex File）

Hex ファイルは Bin ファイルと同様で、16 進数の 0~F で構成されます。行の数はメモリのアドレス深さを表し、各行のデータのバイナリ数はメモリのデータ幅を表します。

```
#File_format=Hex
#Address_depth=8
#Data_width=16
3A40
A28E
0B52
1C49
D602
```

0801

03E6

4C18

7.3 アドレス 16 進法形式（Address-Hex File）

Address-Hex ファイルは、データ記録を有するアドレスとデータを記録します。アドレスとデータはすべて 16 進法数の 0~F から成り、各行のコロンの前はアドレスで、コロンの後はデータです。ファイルでは、書き込みデータおよびそのアドレスのみ記録し、記録のないアドレスのデータはデフォルトで 0 です。

```
#File_format=AddrHex
#Address_depth=256
#Data_width=16
9:FFFF
23:00E0
2a:001F
30:1E00
```


GOWIN高云
智慧逻辑 定制未来