



LatticeXP2 メモリ使用ガイド

はじめに

このテクニカルノートはLatticeXP2™ デバイスファミリのメモリ使用法について議論します。ispLEVERでこれらのデバイスファミリ用のユーザタグ(Tag)、組み込みブロックRAM (EBR)とPFUベースのメモリを集積する際におけるガイドとして、設計エンジニアに用いられることを意図しています。

これらデバイスのアーキテクチャは、FPGAオンチップメモリを用いるアプリケーションのためのリソースを提供します。sysMEM組み込みブロックRAM (EBR)とPFUベースの分散メモリは相互に補います。シングルポートRAM、デュアルポートRAM、擬似デュアルポートRAM、FIFOおよびROMメモリはEBRを用いることで構成できます。LUTとPFUは、分散シングルポートRAM、分散デュアルポートRAM、および分散ROMを実装することができます。ユーザ・タグメモリはデバイスによってサイズは異なりますが、各デバイスにあります。

ユーザ・タグメモリとEBRブロックRAM、及びPFU RAMの機能はプリミティブと呼ばれ、本テクニカルノートの後ろで説明されます。設計者はispLEVERソフトウェアのIPexpress™ ツールによって、2つの異なる方法でメモリ・プリミティブを利用することができます。IPexpressで所望のメモリタイプとサイズの指定ができます。IPexpressはメモリ・プリミティブの1つかそれ以上を用いることによって、必要なメモリを実装するネットリストを生成します。

このドキュメントの残りは、IPexpress、メモリモジュール、およびメモリプリミティブの活用方法について議論します。

LatticeXP2デバイスにおけるメモリ

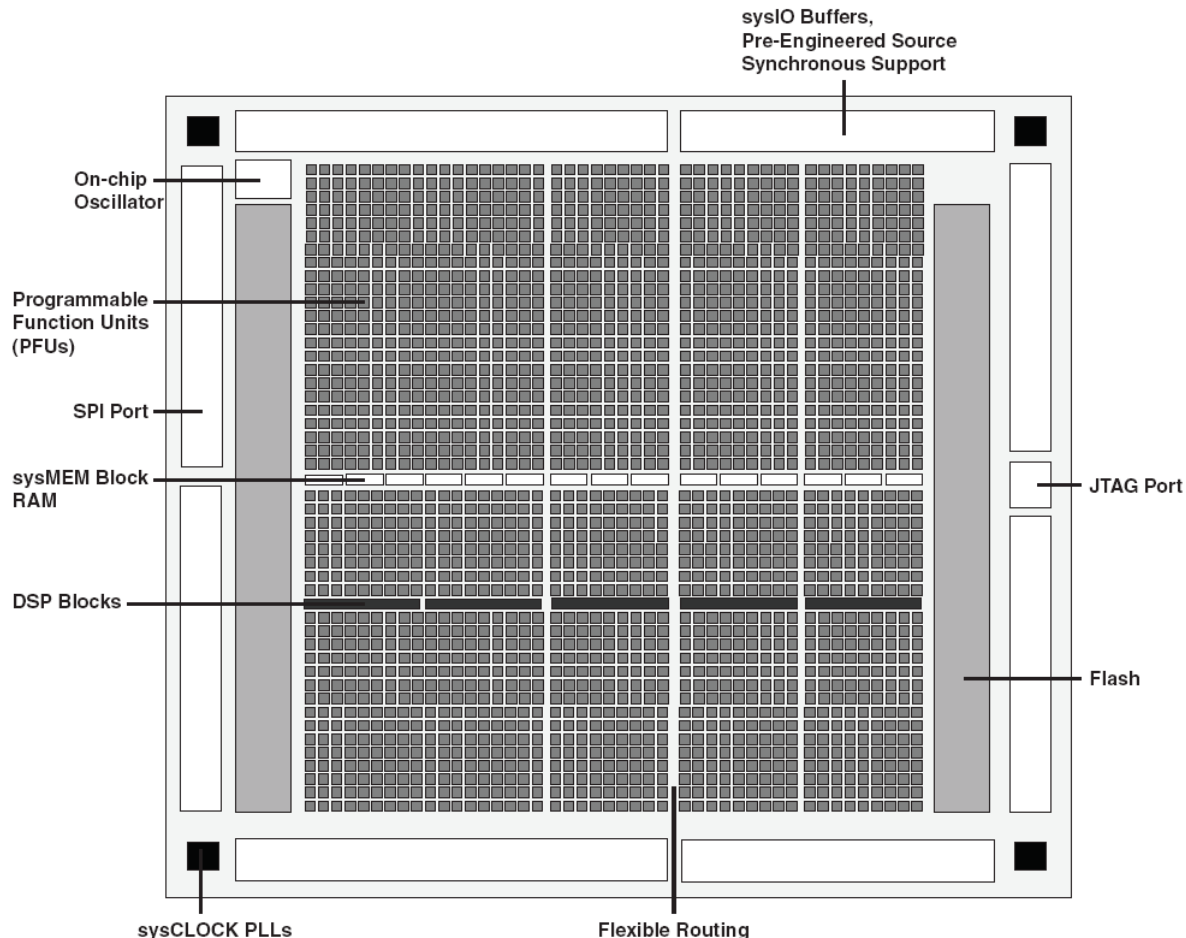
2種類の論理ブロック、即ちプログラマブル・ファンクションユニット(PFU)、およびRAM対応なしのプログラマブル・ファンクションユニット(PFF)があります。PFUはロジック、演算、RAM、ROM、およびレジスタ機能のためのビルディングブロックを含みます。PFFブロックはロジック、演算、およびROM機能のためのビルディングブロックを含んでいます。複雑なデザインを短時間に、効率的に実装できる柔軟性のためにPFUとPFFブロックが共に最適化されています。論理ブロックは2次元配列でアレンジされています。同じタイプのブロックが（横方向の）列単位で用いられます。

sysMEM EBRブロックはLatticeXP2ファミリには最大2列含まれています。sysMEM EBRは18Kの大きさで専用の高速メモリブロックです。各sysMEMブロックはさまざまな深さと幅でRAMやROMを構成することができます。LatticeXP2はユーザタグ専用の列があり、最大451バイトです。

表 10-1 LatticeXP2 メモリの規模

パラメータ	XP2-5	XP2-8	XP2-17	XP2-30	XP2-40
EBR列数	1	1	1	1	2
EBRブロック数	9	12	15	21	48
EBRサイズ (bit)	165888	221184	276480	387072	884736
分散RAMサイズ (bit)	10368	18432	34560	64512	82944
全メモリサイズ (bit)	176256	239616	311040	451584	967680

図 10-1 簡略化したLatticeXP2デバイスブロック図 (上面図)



IPexpressの活用

設計者は、IPexpressを活用することで、種々メモリを容易に指定することができます。これらのモジュールは、必要に応じて汎用の配線とLUTと共に1つかそれ以上のメモリ・プリミティブを用いることで構成されます。利用できるモジュールは以下の通りです。

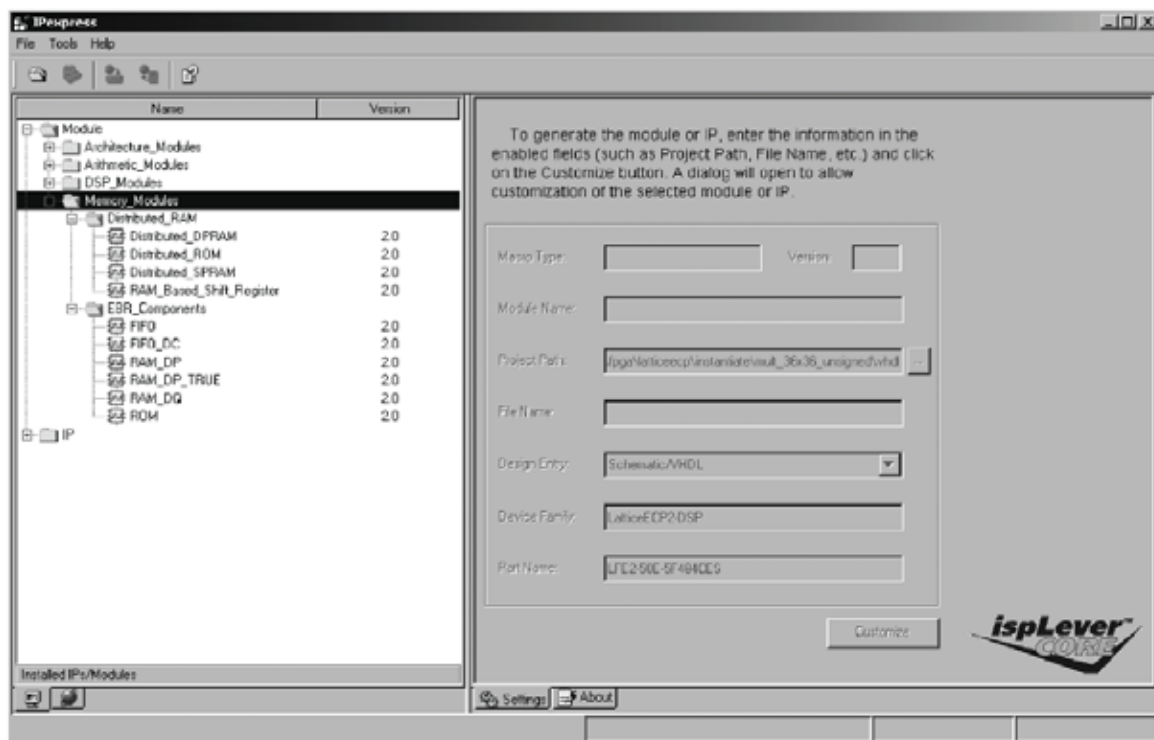
- ・ シングルポートRAM (RAM_DQ) ; EBRベース
- ・ デュアルポートRAM (RAM_DP_TRUE) ; EBRベース
- ・ 擬似デュアルポートRAM (RAM_DP) ; EBRベース
- ・ リードオンリ・メモリ(ROM) ; EBRベース
- ・ ファーストイン・ファーストアウト・メモリ(FIFO_DC) ; EBRベース
- ・ 分散シングルポートRAM (Distributed_SPRAM) ; PFUベース
- ・ 分散デュアルポートRAM (Distributed_DPRAM) ; PFUベース
- ・ 分散ROM (Distributed_ROM) ; PFU/PFFベース
- ・ ユーザ・タグメモリ(SSPIA) ; TAGベース

IPexpressフロー

これらのメモリのどれかを生成するには、まずLatticeXP2デバイス用のプロジェクトを作成(オープン)します。

プロジェクト・ナビゲータから、**Tools > IPexpress**を選択します。或いは、LatticeXP2デバイスがターゲットになっている場合、ツールバー内のボタンをクリックします。図10-2で示されるようにIPexpressウィンドウがオープンします。

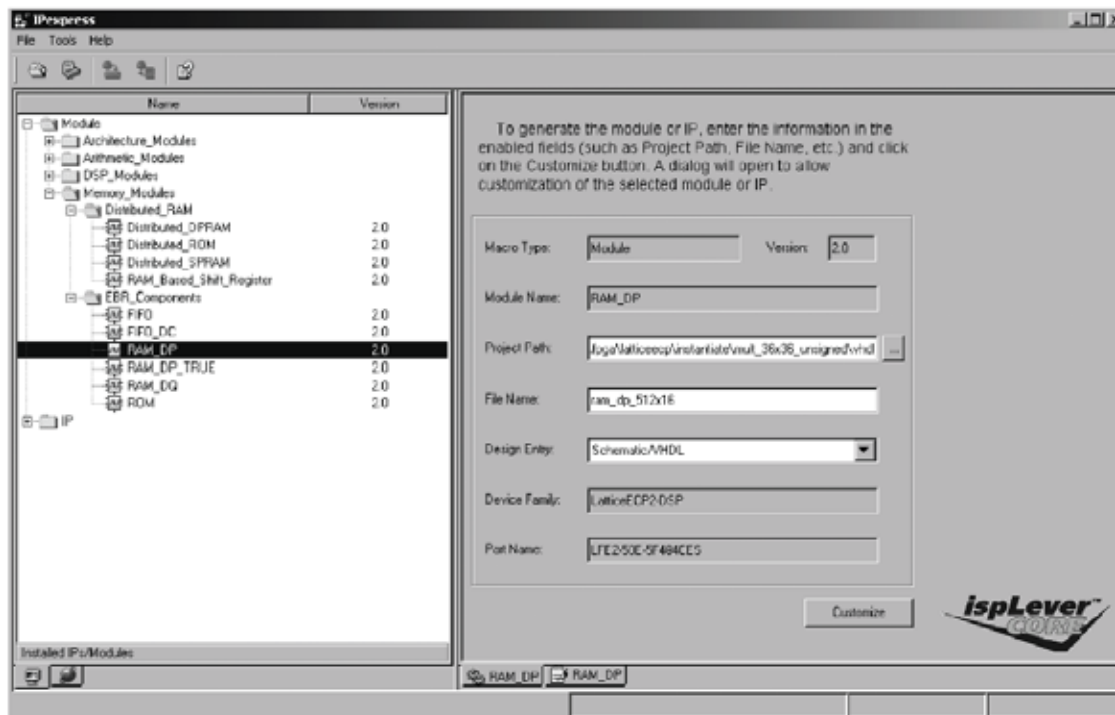
図 10-2 IPexpress メインウィンドウ



このウィンドウの左枠には、モジュールツリー(Module Tree)があります。図10-2に見られるように、EBRベースのメモリモジュールは**EBR_Components**の下に、PFUベースの分散メモリモジュールは**Storage_Components**の下にあります。

サイズ512 x 16のEBRベースの擬似デュアルポートRAM生成に関する例を見ます。まず**EBR_Components**下の**RAM_DP**を選択します。右枠は図10-3で示されるようになります。

図 10-3 IPexpress を用いることで擬似デュアルポートRAM (RAM_DP) を生成する例



ウィンドウの右枠には、デバイスファミリー(**Device Family**)、マクロタイプ(**Macro Type**)、カテゴリ(**Category**)、およびモジュール名(**Module Name**)のような、選択されたデバイスとモジュールに依存するオプションがあります。IPexpressでこれらを変えることはできません。

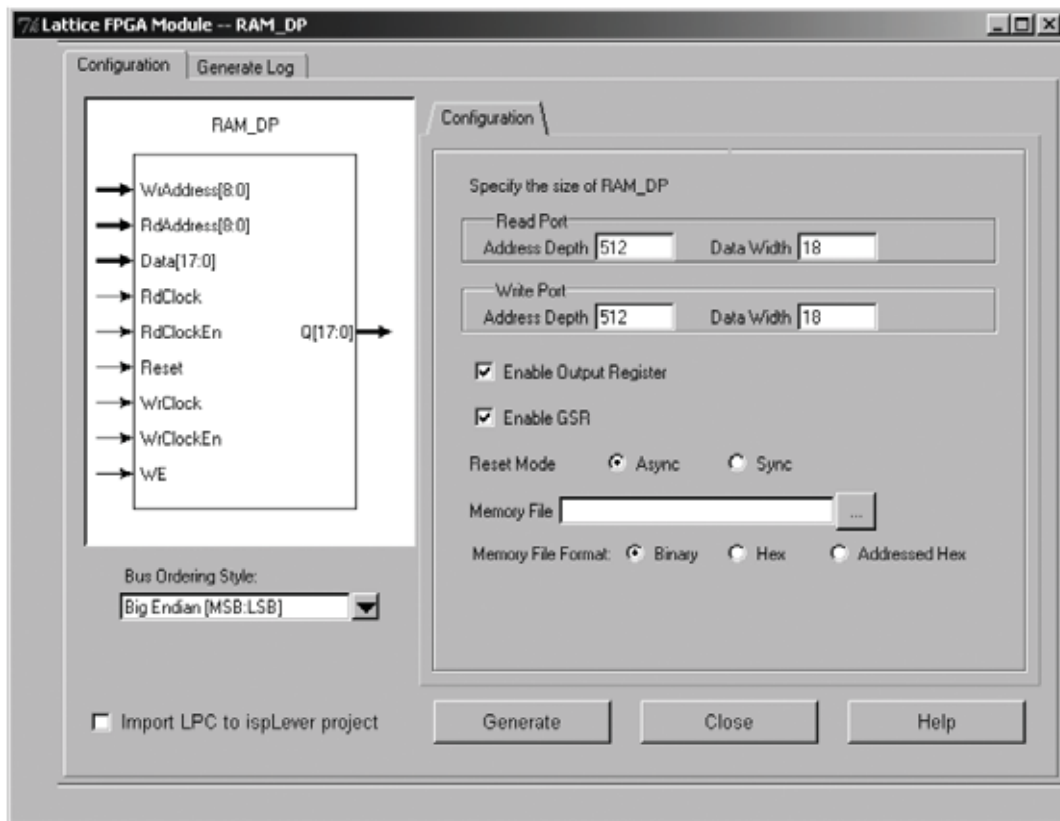
ユーザはプロジェクトパス (**Project Path**) ブラウズ(**Browse**)ボタンをクリックすることによって、生成するモジュールのファイルが置かれるディレクトリを変えることができます。

モジュール名 (**Module Name**) テキストボックスで、ユーザは生成しようとしているモジュールのエンティティ名を指定します。ユーザはこれを入力しなければなりません。

設計入力 (**Design Entry**, VerilogかVHDL) はデフォルトではプロジェクトタイプと同じです。プロジェクトがVHDLプロジェクトの場合、選択された設計入力オプションは "Schematic/VHDL" (日本語版註; 又は "VHDL" も可) で、プロジェクトタイプがVerilog-HDLでは、"Schematic/Verilog-HDL" (日本語版註; 又は "Verilog HDL" も可) です。

デバイス (**Device**) プルダウンメニューでユーザは同一ファミリー内、この場合LatticeXP2、で異なるデバイスを選択することが可能です。カスタマイズ(**Customize**)ボタンをクリックすることで、図10-4に示す別のウィンドウが開き、ここでユーザはRAMのカスタマイズできます。

図 10-4 擬似デュアルポートRAM (RAM_DP) モジュールのカスタマイズ例 ~ Configuration タブ



このウィンドウの左側はモジュールのブロック図を示します。右側にはコンフィグレーション (**Configuration**) タブがあり、RAM_DP をカスタマイズ (アドレス・ポートサイズとデータ幅を指定) するためのオプションを選択できます。

ユーザはテキストボックスに値を入力することで、リードポートとライトポートそれぞれのアドレスの深さとデータ幅を指定します。この例では、サイズ 512x16 の擬似デュアルポート RAM を生成しています。また、ユーザは擬似デュアルポートと真の (True) デュアルポート RAM の場合で異なるポート幅の RAM を作成することができます。

EBR ベースの RAM においてはクロックに同期した書き込みのみをサポートするため、データ入力とアドレス制御には常にレジスタがあります。チェックボックス **Enable Output Registers** は読み出しデータポートに出力レジスタを挿入します。EBR ベースの RAM で、出力レジスタはオプションです。

ユーザは非同期リセットか同期リセットかのリセットモード (**Reset Mode**) を選択できます。また GSR (Global Set Reset) を許可 (**Enable GSR**) するかどうかを選択することができます。

ユーザは **Memory File** で指定するメモリファイルを用意し、メモリを事前に (コンフィグレーション時に) 初期化することもできます。このファイルを RAM に用意するのはオプションですが、ROM の場合は必須です。ファイルはバイナリ、16 進 (Hex) または Addresses Hex 形式のいずれかです。このテクニカルノートの “初期設定ファイル” セクションでこれらの形式の詳細について議論します。

ここでユーザは、カスタマイズしたモジュールを生成するために **Generate** ボタンをクリックすることができます。Verilog が VHDL のネットリストが生成され、指定されたディレクトリに置かれます。ユーザはこのネットリストをその設計内に取り込むことができます。

別の重要なボタンは **Load Parameters** ボタンです。IPexpress は <module_name>.lpc ファイルの中で指定されたパラメータを格納します。このファイルはモジュールと共に生成されます。ユーザは Load Parameters

ボタンをクリックすることで、以前に生成されたモジュールのパラメータをロードし、確認するか或いはその変更をすることができます。

モジュールがいったん生成されると、ユーザはその設計内で *.lpc か Verilog-HDL/VHDL ファイルをトップレベル・モジュールでインスタンスすることができます。

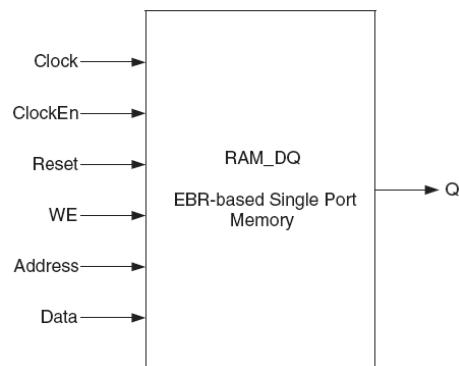
本テクニカルノートでは後に、EBRと分散メモリの種々メモリモジュールについて議論します。

メモリモジュール

シングルポートRAM (RAM_DQ) ~ EBRベース

シングルポートRAM (RAM_DQ) としてLatticeXP2デバイス内のEBRブロックを構成することができます。IPexpressは、ユーザの設計要件に従ったメモリサイズ用に、EDIFネットリストと共にVerilog-HDLかVHDLを生成します。図10-5に生成されるメモリモジュールを示します。

図 10-5 IPexpressによって生成されるシングルポート・メモリモジュール



デバイスには多くのEBRブロックがあります。IPexpress GUI内でユーザによって指定されたメモリサイズを生成するために、これらEBRブロックやプリミティブを使用し連結します。1つのEBRブロックより小さいメモリでは、モジュールは1つのEBRブロックで作成されます。指定されたメモリが1つのEBRブロックより大きい場合では、そのサイズを作成するのに必要とされる深さと幅で、複数のEBRブロックを連結して生成されます。

シングルポートRAMモードで、ポートへの入力データとアドレスにはメモリアレイの入力にレジスタがあります。メモリの出力データへのレジスタはオプションです。

シングルポートメモリの様々なポートとそれらの定義は表10-2に含まれており、IPexpressによって生成されたモジュールとEBR RAM_DQプリミティブの対応するポートを記載します。

表 10-2 EBRベースのシングルポートメモリのポート定義

生成されたモジュールでのポート名	EBRブロックプリミティブのポート名	記述	アクティブなステート
Clock	CLK	クロック	立ち上がりエッジ
ClockEn	CE	クロック・イネーブル	Highアクティブ
Address	AD[x:0]	アドレスバス	—
Data	DI[y:0]	入力データ	—
Q	DO[y:0]	出力データ	—
WE	WE	ライトイネーブル	Highアクティブ
Reset	RST	リセット	Highアクティブ
—	CS[2:0]	チップセレクト	—

リセット(RST)はRAMの入出力レジスタをリセットするだけで、メモリの内容をリセットしません。

EBRプリミティブにあるCS (チップセレクト) ポートは、メモリが複数のEBRブロックを連結される必要があるときに役立ちます。この場合3ビットのCS信号はアドレスのMSBとなり、8つのメモリまで容易に連結できます。ユーザによって指定されたメモリサイズが8つ以上のEBRブロック以上を必要とする場合、ソフトウェアは自動的にPFUで(EBRブロックの外部)で実装される追加アドレスデコード・ロジックを生成します。

それぞれのEBRブロックは18,432ビットのRAMより成り、それぞれのアドレス幅(x)とデータ幅(y)が表10-3に示されています。

表10-3 LatticeXP2 デバイス、16Kメモリのシングルポートメモリ・サイズ

シングルポート メモリサイズ	入力データ	出力データ	アドレス[MSB:LSB]
16K x 1	DI	DO	AD[13:0]
8K x 2	DI[1:0]	DO[1:0]	AD[12:0]
4K x 4	DI[3:0]	DO[3:0]	AD[11:0]
2K x 9	DI[8:0]	DO[8:0]	AD[10:0]
1K x 18	DI[17:0]	DO[17:0]	AD[9:0]
512 x 36	DI[35:0]	DO[35:0]	AD[8:0]

表10-4はシングルポートメモリ(RAM_DQ)で利用できる様々なアトリビュートを示します。これらの幾つかはIPexpress GUIを通してユーザが選択可能です。詳細なアトリビュート定義については、付録Aを参照してください。

表10-4 LatticeXP2 デバイスのシングルポートRAMアトリビュート

アトリビュート	記述	値	デフォルト値	IPexpressを通して ユーザ選択可能
Address depth	アドレス深さ	16K, 8K, 4K, 2N, 1K, 512		YES
Data Width	データ幅	1, 2, 4, 9, 18, 36	1	YES
Enable Output Registers	レジスタモード(パイプライン)	NOREG, OUTREG	NOREG	YES
Enable GSR	グローバル・セット/リセット	ENABLE, DISABLE	ENABLE	YES
Reset Mode	リセットタイプを選択	ASYNC, SYNC	ASYNC	YES
Memory File Format		BINARY, HEX, ADDRESSED HEX		YES
Write Mode	リード/ライト・モード	NORMAL, WRITETHROUGH	NORMAL	YES
Chip Select Decode	チップセレクト・デコード	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
Init Value	初期化値	0x00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00....0xFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFF	0x00000000 0000000000 0000000000 0000000000 0000000000 0000000000 0000000000 0000000000 0000000000 0000000000 0000000000 0000000000 00	NO

シングルポートRAM(RAM_DQ)を構成できるモードは、ノーマル(NORMAL)、またはライトスルー(WRITE THROUGH) のいずれかです。これらモードは、書き込み動作のすぐ後に読み出し動作が同じメモリアドレ

スに対して続いた場合、どのようなデータがメモリのポートQから出て来るかに影響します。

さらに、シングルポートRAMの出力レジスタをイネーブルするかどうかを選択することができます。図10-6から図10-9は、これらのオプションに対するシングルポートRAMの内部タイミング波形をそれぞれ示しています。

図10-6 シングルポートRAMタイミング波形 ~ NORMAL モード、出力レジスタなし

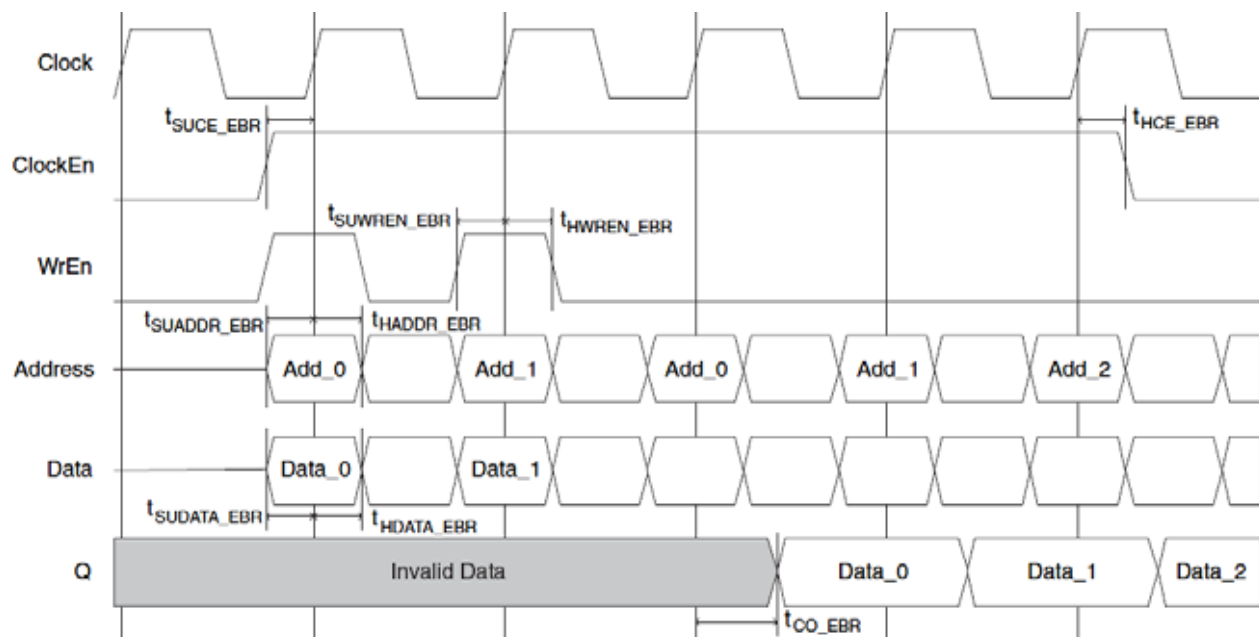


図10-7 シングルポートRAMタイミング波形 ~ NORMAL モード、出力レジスタあり

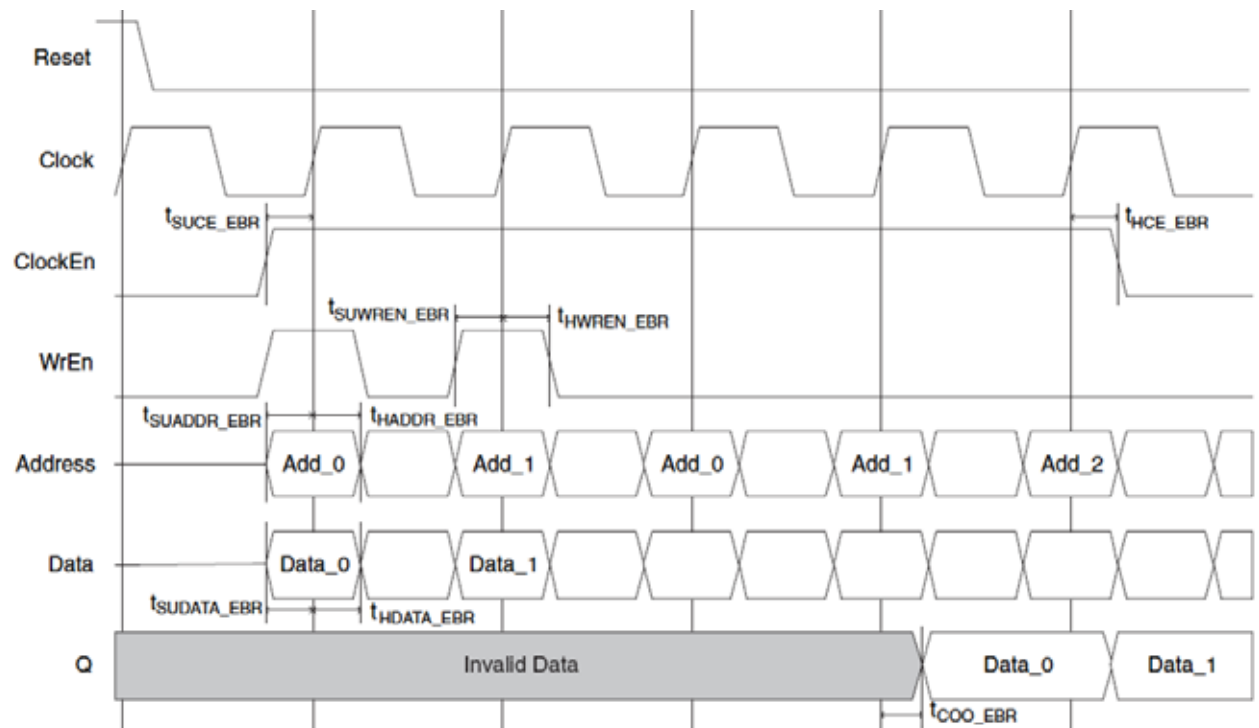


図10-8 シングルポートRAMタイミング波形 ~ WRITE THROUGH モード、出力レジスタなし

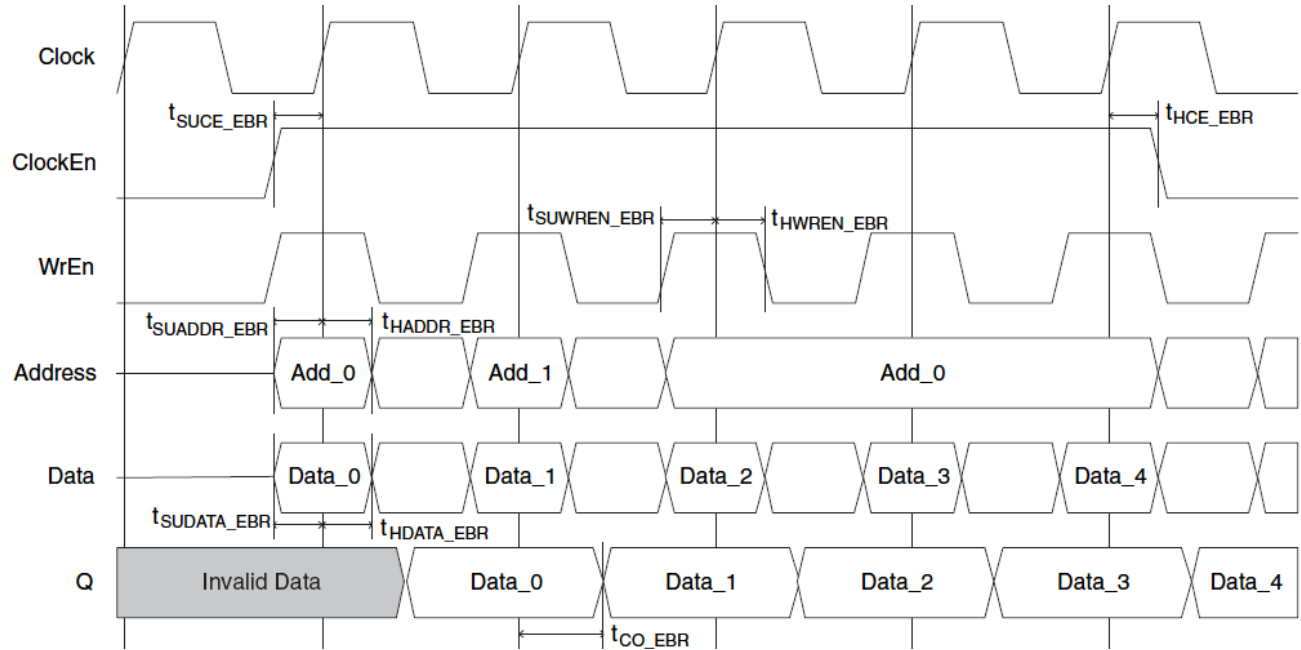
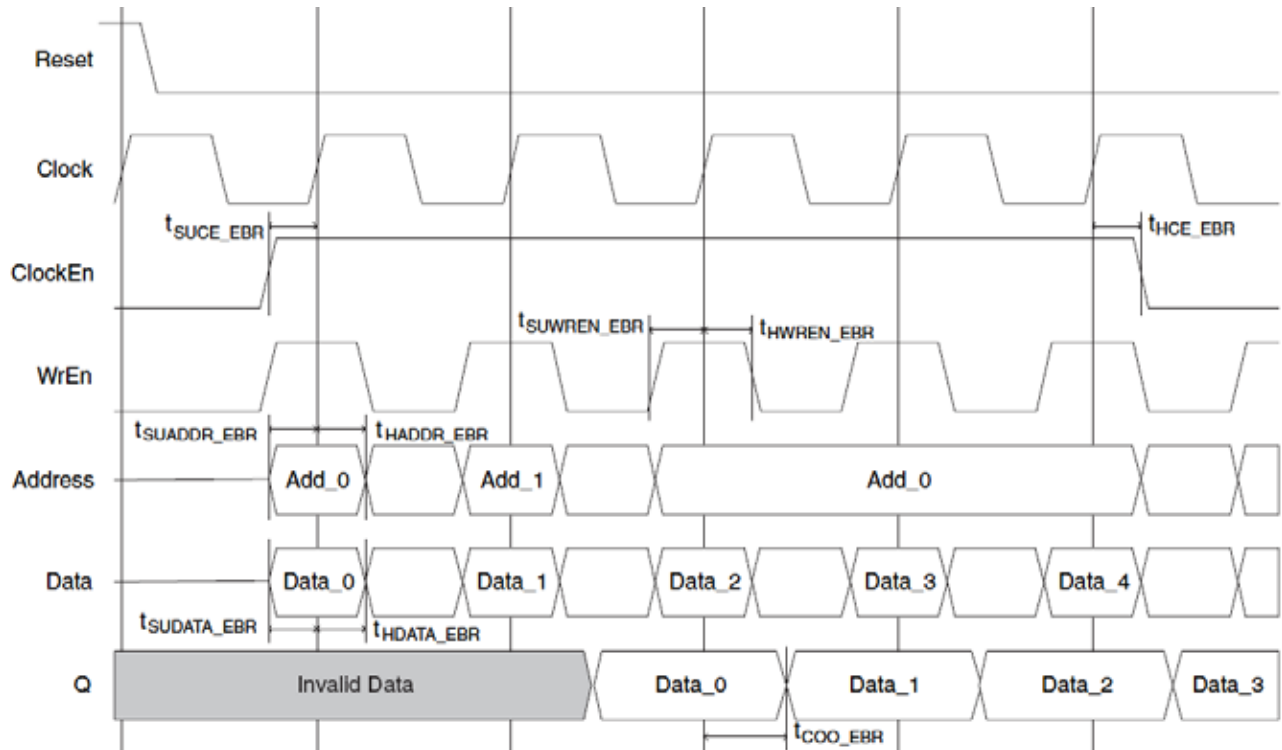


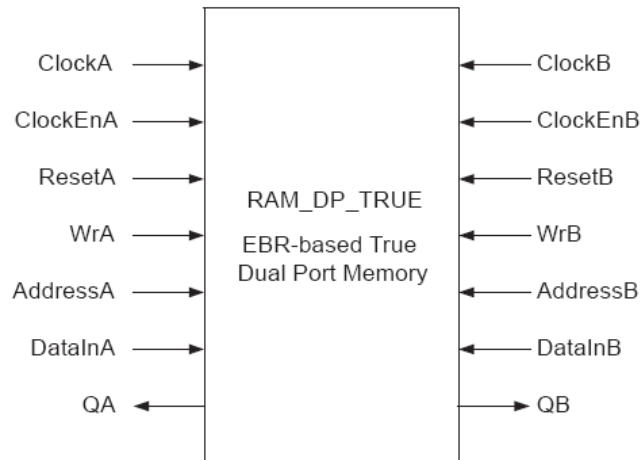
図10-9 シングルポートRAMタイミング波形 ~ WRITE THROUGH モード、出力レジスタあり



真の(True)デュアルポートRAM (RAM_DP_TRUE) ; EBRベース

True-Dual Port RAM (RAM_DP_TRUE) としてLatticeXP2デバイスにおけるEBRブロックを構成することができます。IPexpressは、ユーザの設計要件に従ったメモリサイズ用に、EDIFネットリストと共にVerilog-HDLかVHDLを生成します。図10-10に生成されるメモリモジュールを示します。

図10-10 IPexpress で生成される真のデュアルポート・メモリモジュール



生成されたモジュールはRAM_DP_TRUEプリミティブを利用します。1つのEBRブロックより小さいメモリでは、モジュールは1つのEBRブロックで作成されます。指定されたメモリが1つのEBRブロックより大きい場合は、そのサイズを作成するのに必要とされる深さと幅で、複数のEBRブロックを連結して生成されます。

真のデュアルポートRAMモードで、入力データとアドレスポートにはメモリアレイの入力レジスタがあります。メモリの出力データへのレジスタはオプションです。

真のデュアルポート・メモリの種々ポートとそれらの定義は表10-5に示されています。IPexpressによって生成されたモジュールとEBR RAM_DP_TRUEプリミティブの対応するポートを記載します。

表10-5 EBRベースの真のデュアルポート・メモリのポート定義

生成されたモジュールのポート名	EBRブロック・プリミティブのポート名	記述	アクティブなステート
ClockA, ClockB	CLKA, CLKB	ポートA/ポートBのクロック	立ち上がりエッジ
ClockEnA, ClockEnB	CEA, CEB	ポートA/ポートBのクロック・イネーブル	Highアクティブ
AddressA, AddressB	ADA[x1:0], ADB[x2:0]	ポートA/ポートBのアドレスバス	—
DataA, DataB	DIA[y1:0], DIB[y2:0]	ポートA/ポートBの入力データ	—
QA, QB	DOA[y1:0], DOB[y2:0]	ポートA/ポートBの出力データ	—
WEA, WEB	WEA, WEB	ポートA/ポートBのライトイネーブル	Highアクティブ
ResetA, ResetB	RSTA, RSTB	ポートA/ポートBのリセット	Highアクティブ
—	CSA[2:0], CSB[2:0]	各ポートのチップセレクト	—

リセット(RST)はRAMの入出力レジスタをリセットするだけで、メモリの内容をリセットしません。

EBRプリミティブにあるCS (チップセレクト) ポートは、メモリが複数のEBRブロックを連結される必要があるときに役立ちます。この場合3ビットのCS信号はアドレスのMSBとなり、8つのメモリまで容易に連結できます。ユーザによって指定されたメモリサイズが8つ以上のEBRブロック以上を必要とする場合、ソフトウェアは自動的にPFUで(EBRブロックの外部)で実装される追加アドレス・デコード・ロジックを生成

します。

それぞれのEBRブロックは18,432ビットのRAMより成り、それぞれのアドレス幅(x)とデータ幅(y)が表10-6に示されています

表10-6 LatticeXP2デバイス、16Kメモリの真のデュアルポート・メモリサイズ

デュアルポート メモリサイズ	入力データ ポートA	入力データ ポートB	出力データ ポートA	出力データ ポート B	アドレスポート A [MSB:LSB]	アドレスポート B [MSB:LSB]
16K x 1	DIA	DIB	DOA	DOB	ADA[13:0]	ADB[13:0]
8K x 2	DIA[1:0]	DIB[1:0]	DOA[1:0]	DOB[1:0]	ADA[12:0]	ADB[12:0]
4K x 4	DIA[3:0]	DIB[3:0]	DOA[3:0]	DOB[3:0]	ADA[11:0]	ADB[11:0]
2K x 9	DIA[8:0]	DIB[8:0]	DOA[8:0]	DOB[8:0]	ADA[10:0]	ADB[10:0]
1K x 18	DIA[17:0]	DIB[17:0]	DOA[17:0]	DOB[17:0]	ADA[9:0]	ADB[9:0]

表10-7は真のデュアルポートMemory(RAM_DP_TRUE)で利用できる様々なアトリビュートを示します。これらのいくつかはIPexpress GUIを通してユーザの選択が可能です。詳細なアトリビュート定義については、付録Aを参照してください。

表10-7 LatticeXP2用の真のデュアルポートRAMアトリビュート

アトリビュート	記述	値	デフォルト値	IPexpressでユーザ 選択可能
Port A Address depth	ポートAのアドレス深さ	16K, 8K, 4K, 2K, 1K		YES
Port A Data Width	ポートAのデータ幅	1, 2, 4, 9, 18	1	YES
Port B Address depth	ポートBのアドレス深さ	16K, 8K, 4K, 2K, 1K		YES
Port B Data Width	ポートBのデータ幅	1, 2, 4, 9, 18	1	YES
Port A Enable Output Register	ポートAのレジスタモード(パイプライン)	NOREG, OUTREG	NOREG	YES
Port B Enable Output Register	ポートBのレジスタモード(パイプライン)	NOREG, OUTREG	NOREG	YES
Enable GSR	グローバルセット・リセットをイネーブル	ENABLE, DISABLE	ENABLE	YES
Reset Mode	リセットタイプの選択	ASYNC, SYNC	ASYNC	YES
Memory File Format		BINARY, HEX, ADDRESSED HEX		YES
Port A Write Mode	ポートAのRead / Writeモード	NORMAL, WRITETHROUGH	NORMAL	YES
Port B Write Mode	ポートBのRead / Writeモード	NORMAL, WRITETHROUGH	NORMAL	YES
Chip Select Decode for Port A	ポートAのチップセレクト・デコード	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
Chip Select Decode for Port B	ポートBのチップセレクト・デコード	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
Init Value	初期化値	0x00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 000000000000000000000000....0xFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0x00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000	NO

真のデュアルポートRAM (True Dual Port RAM ; RAM_DP_TRUE) を構成できるモードも、ノーマル、またはライトスルーのいずれかです。これらモードはいずれも、書き込み動作のすぐ後に読み出し動作が同じメモリアドレスに対して続いた場合、どのようなデータがメモリのポートQから出て来るかに影響します。WRITEモードと真のデュアルポートの制約についてはAppendix Aにより詳しく記載されています。

さらに、真のデュアルポートRAMの出力レジスタをイネーブルするかどうかを選択することができます。図10-11から図10-14はこれらのオプションに対する真のデュアルポートRAMの内部タイミング波形をそれぞれ示しています。

図10-11 真のデュアルポートRAMタイミング波形 ~ NORMAL モード、出力レジスタなし

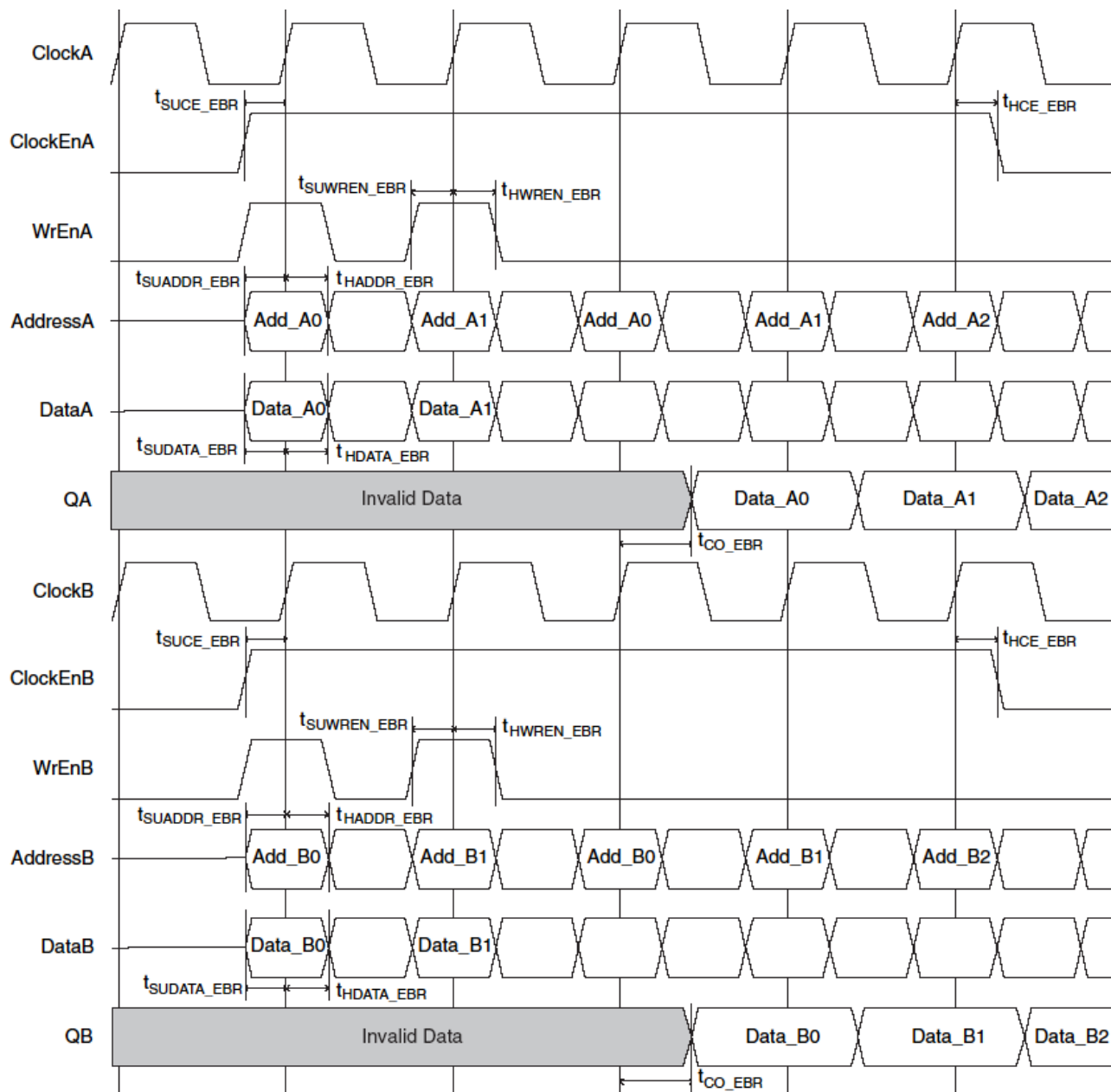


図10-12 真のデュアルポートRAMタイミング波形 ~ NORMAL モード、出力レジスタあり

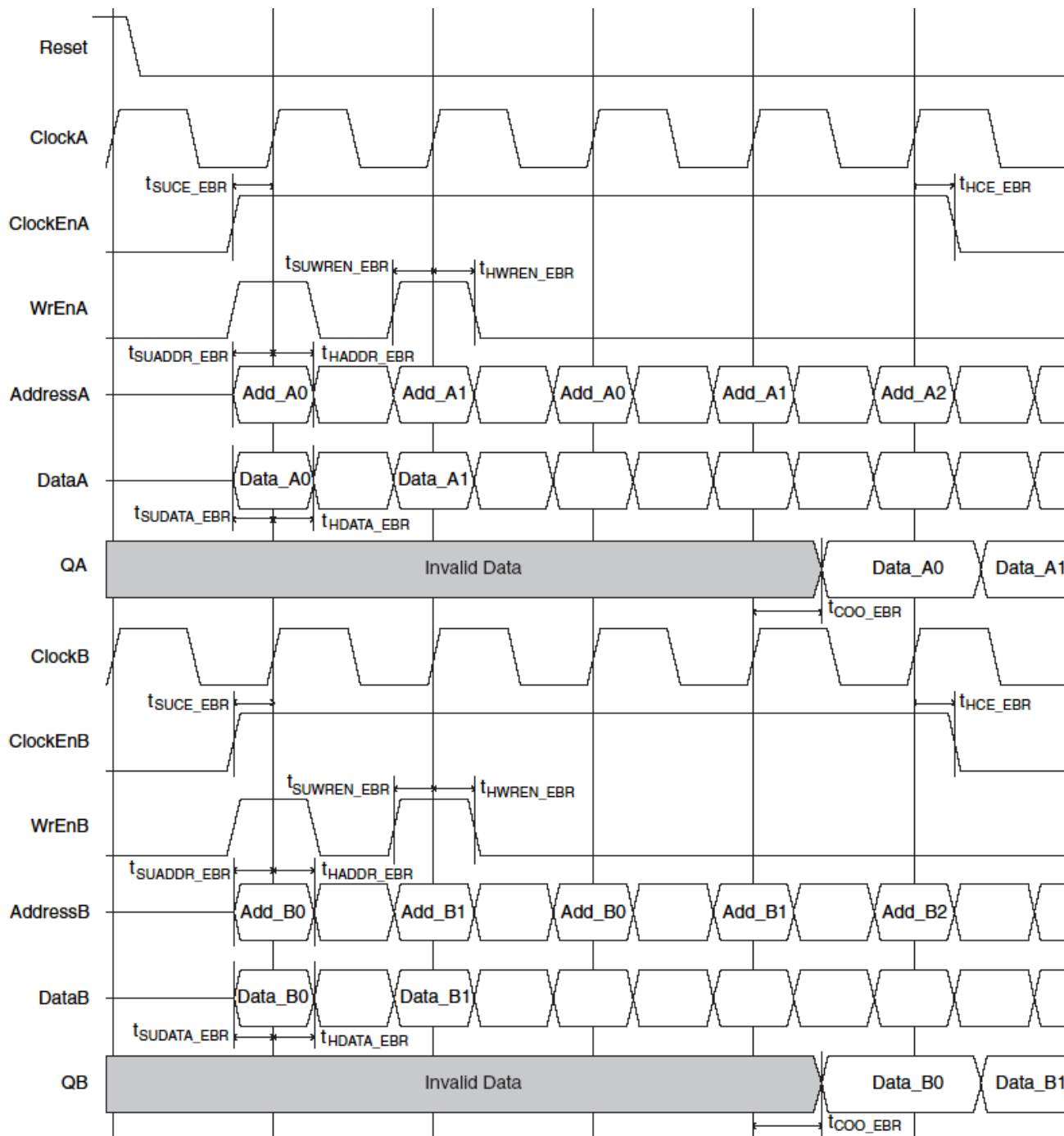


図10-13 真のデュアルポートRAMタイミング波形 ~ WRITE THROUGH モード、出力レジスタなし

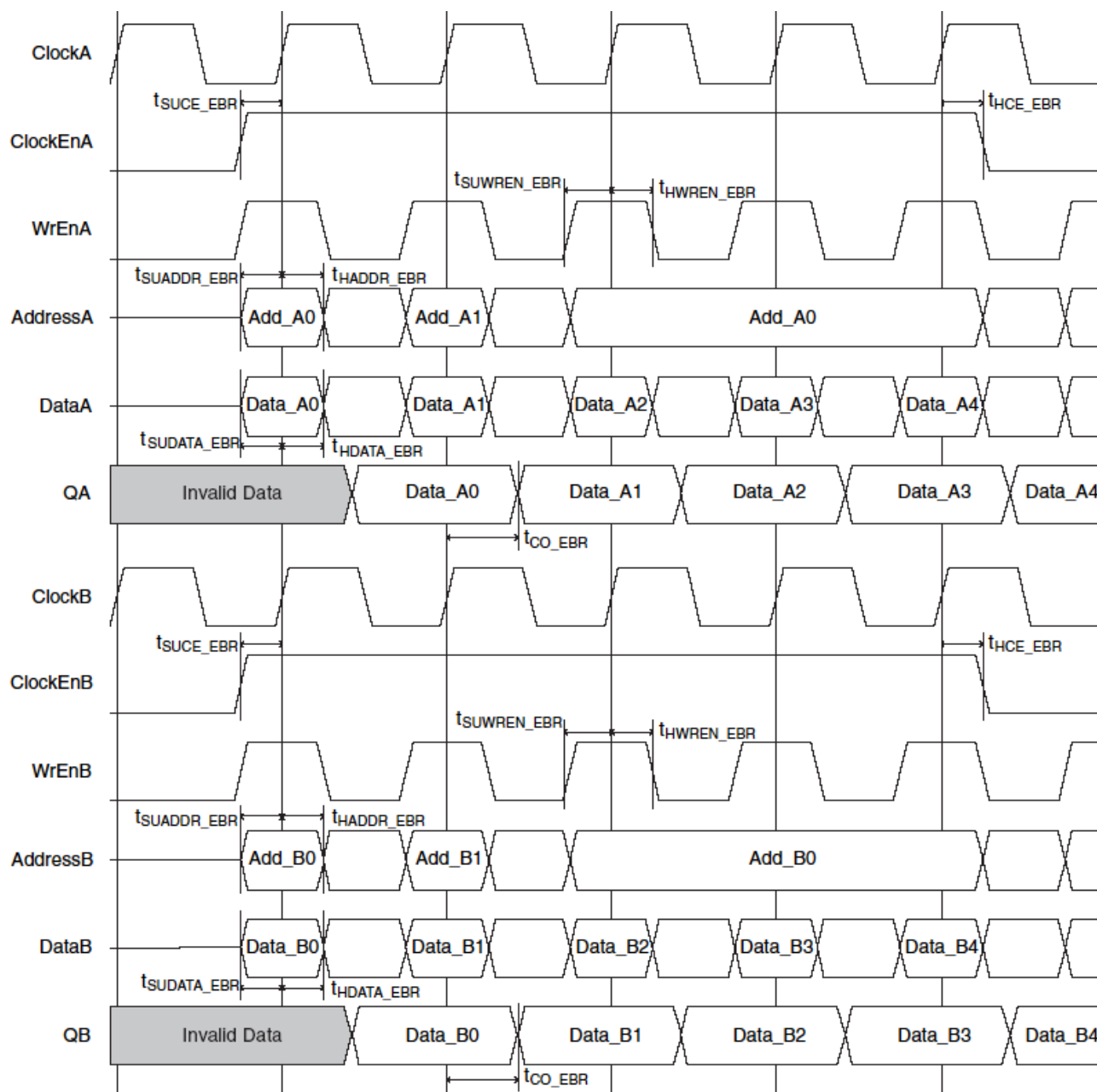
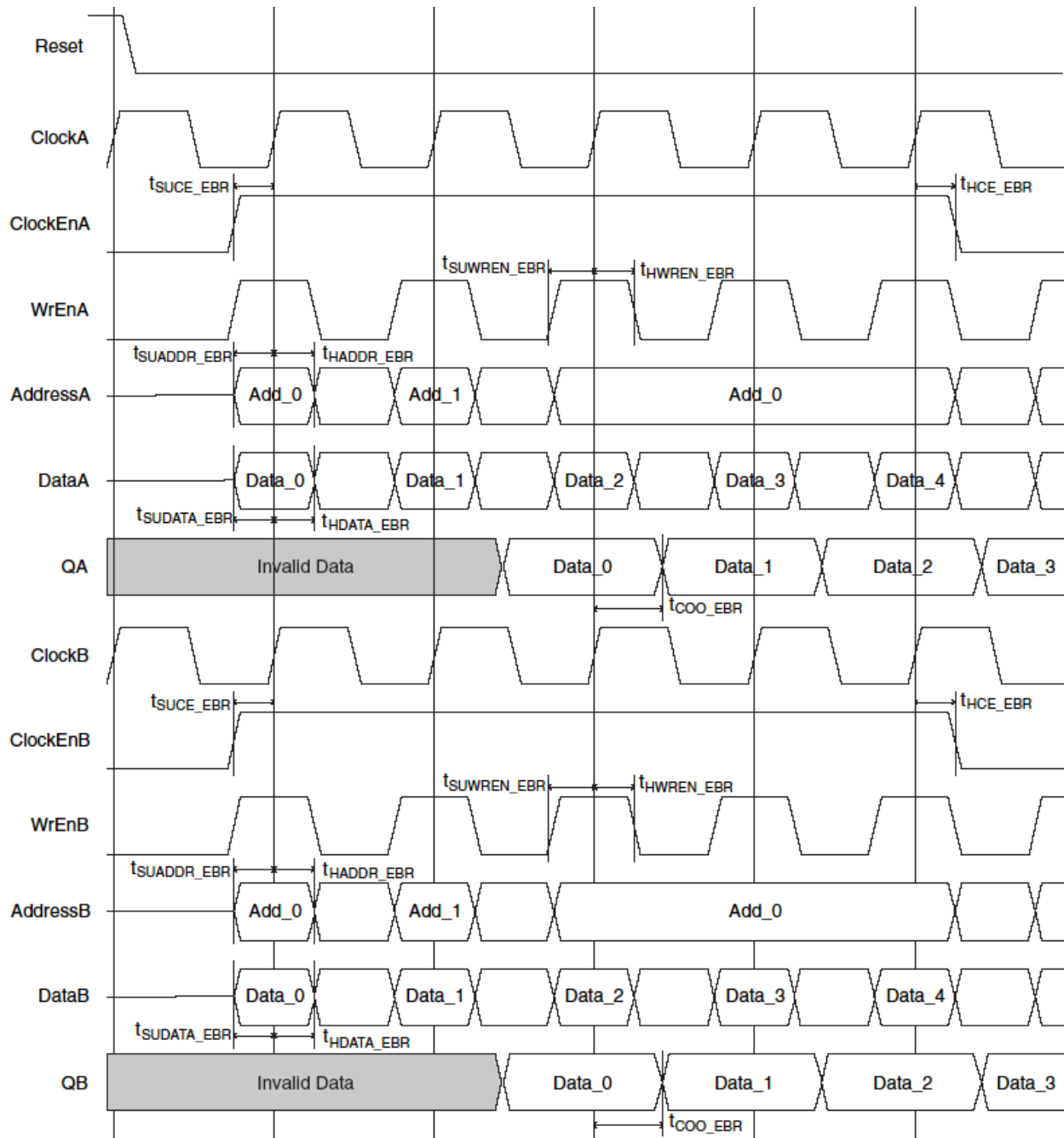


図10-14 真のデュアルポートRAMタイミング波形 ~ WRITE THROUGH モード、出力レジスタあり

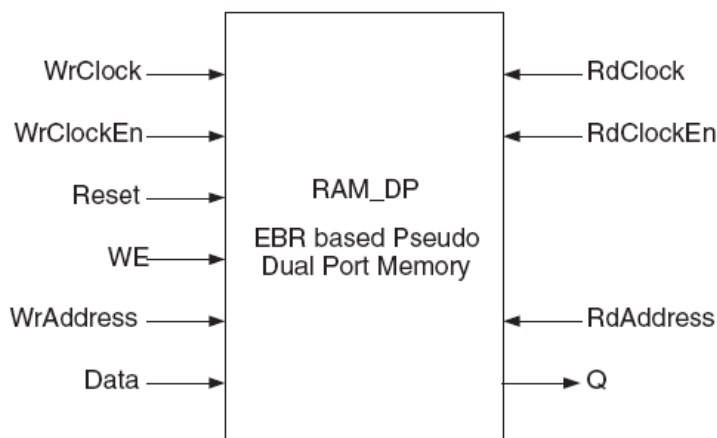


擬似デュアルポートRAM (RAM_DP) ~ EBRベース

LatticeXP2デバイス内のEBRブロックを擬似デュアルポートRAMがRAM_DPとして構成することができます。IPexpressは、ユーザの設計要件に従ったメモリサイズ用に、EDIFネットリストと共にVerilog-HDLかVHDLを生成します。

図10-15に生成されるメモリモジュールを示します。

図10-15 IPexpress で生成される擬似デュアルポート・メモリモジュール



生成されたモジュールはこれらEBRブロックかプリミティブを利用します。1つのEBRブロックより小さいメモリでは、モジュールは1つのEBRブロックで作成されます。指定されたメモリが1つのEBRブロックより大きい場合は、そのサイズを作成するのに必要とされる深さと幅で、複数のEBRブロックを連結して生成されます。

擬似デュアルポートRAMモードで、入力データとアドレスポートにはメモリアレイの入力にレジスタがあります。メモリの出力データへのレジスタはオプションです。

擬似デュアルポート・メモリの様々なポートとそれらの定義は表10-8に含まれています。表はモジュール・マネージャによって生成されたモジュールとEBR RAM_DPプリミティブの対応するポートを記載します。

表10-8 EBRベース擬似デュアルポート・メモリのポート定義

生成されたモジュールのポート名	EBRブロック・プリミティブのポート名	記述	アクティブなステート
RdAddress	ADR[x1:0]	リードアドレス	—
WrAddress	ADW[x2:0]	ライトアドレス	—
RdClock	CLKR	リードクロック	立ち上がりエッジ
WrClock	CLKW	ライトクロック	立ち上がりエッジ
RdClockEn	CER	リードクロック・イネーブル	Highアクティブ
WrClockEn	CEW	ライトクロック・イネーブル	Highアクティブ
Q	DO[y1:0]	リードデータ	—
Data	DI[y2:0]	ライトデータ	—
WE	WE	ライトイネーブル	Highアクティブ
Reset	RST	リセット	Highアクティブ
—	CS[2:0]	チップセレクト	—

リセット(RST)はRAMの入出力レジスタをリセットするだけで、メモリの内容をリセットしません。EBRプリミティブにあるCS(チップセレクト)ポートは、メモリが複数のEBRブロックを連結される必要があるときに役立ちます。この場合3ビットのCS信号はアドレスのMSBとなり、8つのメモリまで容易に連結できます。ユーザによって指定されたメモリサイズが8つ以上のEBRブロック以上を必要とする場合、ソフトウェアは自動的にPFUで(EBRブロックの外部)で実装される追加アドレス・デコード・ロジックを生成します。

それぞれのEBRブロックは18,432ビットのRAMより成り、それぞれのアドレス幅(x)とデータ幅(y)が表10-9に示されています。

表10-9 LatticeXP2 デバイス、16Kメモリの擬似デュアルポート・メモリサイズ

擬似デュアルポート・メモリサイズ	入力データポートA	入力データポートB	出力データポートA	出力データポートB	リード・アドレス・ポートA [MSB:LSB]	ライト・アドレス・ポートB [MSB:LSB]
16K x 1	DIA	DIB	DOA	DOB	RAD[13:0]	WAD[13:0]
8K x 2	DIA[1:0]	DIB[1:0]	DOA[1:0]	DOB[1:0]	RAD[12:0]	WAD[12:0]
4K x 4	DIA[3:0]	DIB[3:0]	DOA[3:0]	DOB[3:0]	RAD[11:0]	WAD[11:0]
2K x 9	DIA[8:0]	DIB[8:0]	DOA[8:0]	DOB[8:0]	RAD[10:0]	WAD[10:0]
1K x 18	DIA[17:0]	DIB[17:0]	DOA[17:0]	DOB[17:0]	RAD[9:0]	WAD[9:0]
512 x 36	DIA[35:0]	DIB[35:0]	DOA[35:0]	DOB[35:0]	RAD[8:0]	WAD[8:0]

表10-10は擬似デュアルポート・メモリ(RAM_DP)で利用できる様々なアトリビュートを示します。これらのいくつかはIPexpress GUIを通してユーザの選択が可能です。詳細なアトリビュート定義について、付録Aを参照してください。

表10-10 LatticeXP2 デバイスの擬似デュアルポートRAMアトリビュート

アトリビュート	記述	値	デフォルト値	IPexpressによってユーザが選択可能
Read Port Address depth	リードポートのアドレス深さ	16K, 8K, 4K, 2K, 1K		YES
Read Port Data Width	リードポートのデータ幅	1, 2, 4, 9, 18	1	YES
Write Port Address depth	ライトポートのアドレス深さ	16K, 8K, 4K, 2K, 1K		YES
Write Port Data Width	ライトポートのデータワード幅	1, 2, 4, 9, 18	1	YES
Write Port Enable Output Register	ライトポートのレジスタモード(パイプライン)	NOREG, OUTREG	NOREG	YES
Enable GSR	グローバルセット・リセットのイネーブル	ENABLE, DISABLE	ENABLE	YES
Reset Mode	リセットタイプの選択	ASYN, SYNC	ASYN	YES
Memory File Format		BINARY, HEX, ADDRESSED HEX		YES
Read Port Write Mode	リードポートのRead / Writeモード	NORMAL	NORMAL	YES
Write Port Write Mode	ライトポートのRead / Writeモード	NORMAL	NORMAL	YES
Chip Select Decode for Read Port	リードポートのチップセレクト・デコード	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
Chip Select Decode for Write Port	ライトポートのチップセレクト・デコード	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
Init Value	初期化値	0x00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000....0xFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0x00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 0000	NO

擬似デュアルポートRAM (RAM_DP) の出力レジスタをイネーブルするかどうか選択できます。図10-16から図10-17はこれらのオプションに対する擬似デュアルポートRAMの内部タイミング波形をそれぞれ示しています。

図10-16 擬似デュアルポートRAMタイミング波形 ~ 出力レジスタなし

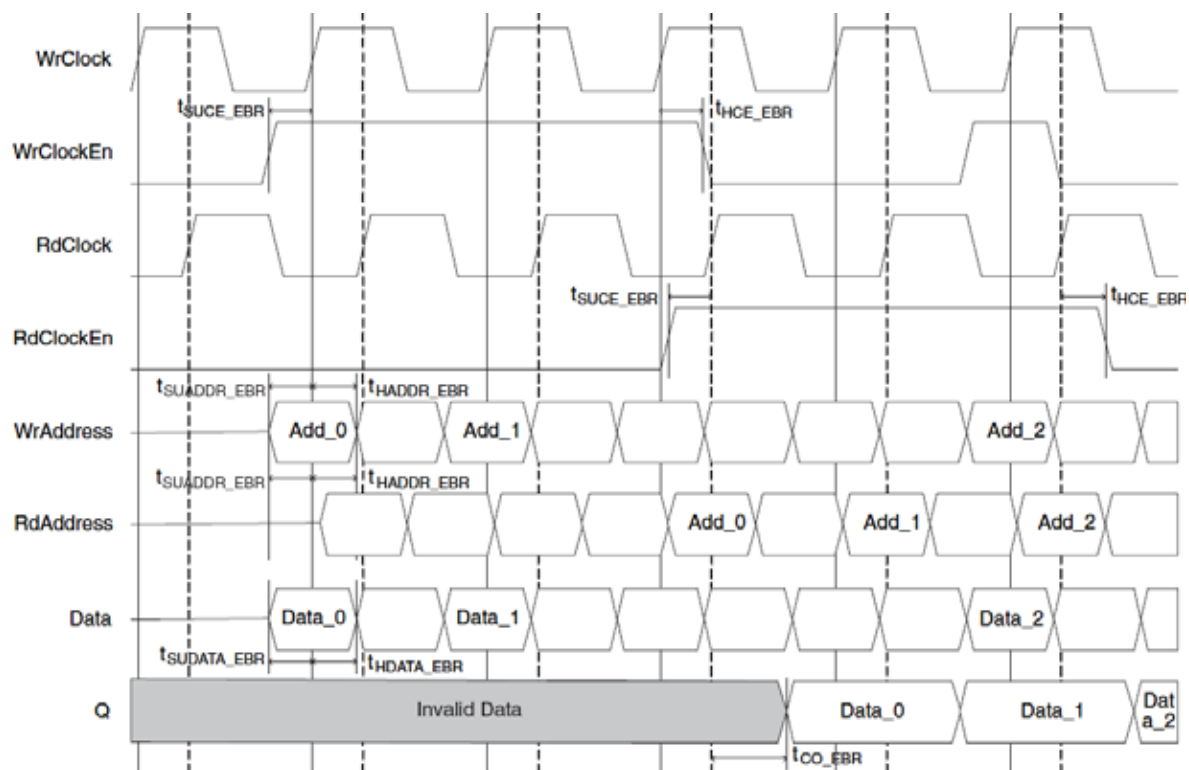
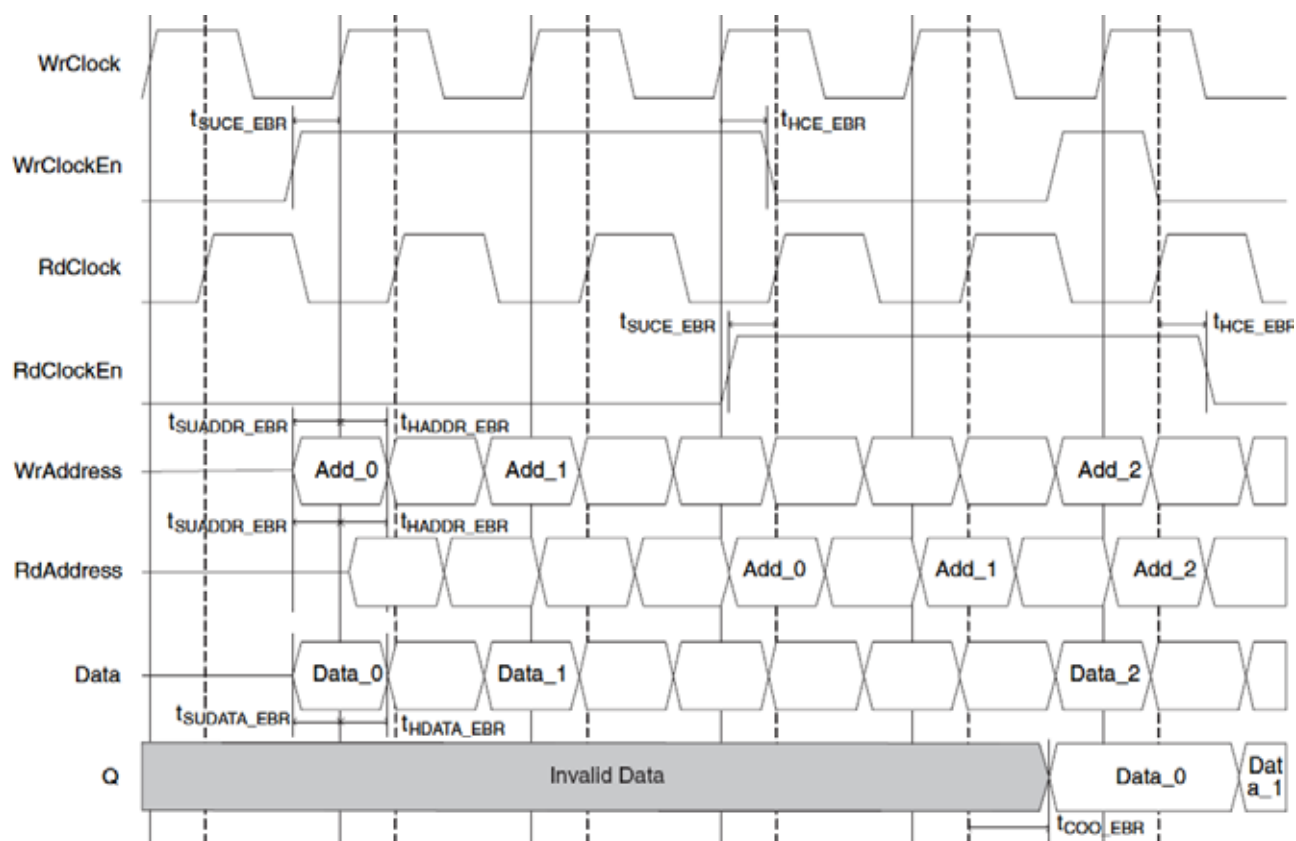


図10-17 擬似デュアルポートRAMタイミング波形 ~ 出力レジスタあり

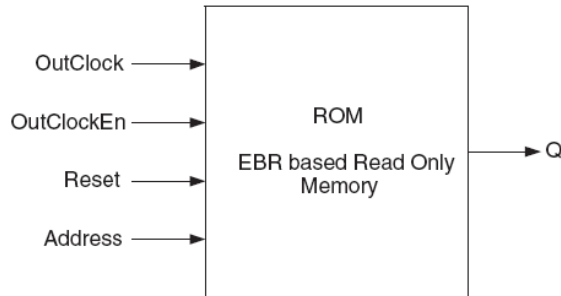


リードオンリ・メモリ(ROM) ; EBRベース

LatticeXP2デバイスのEBRブロックをリードオンリ・メモリ (ROM) として構成することができます。IPexpressは、ユーザが設計要件に従ったメモリサイズ用に、EDIFネットリストと共にVerilog-HDLかVHDLを生成することを可能にします。ユーザはROM内容を初期化ファイルの所定フォーマットで用意しなければなりません。

IPexpressは図10-18で示されるようなメモリモジュールを生成します。

図10-18 IPexpressによって生成されるROM ; リードオンリ・メモリ・モジュール



生成されたモジュールはこれらのEBRブロックかプリミティブを利用します。1つのEBRブロックより小さいメモリでは、モジュールは1つのEBRブロックで作成されます。指定されたメモリが1つのEBRブロックより大きい場合では、そのサイズを作成するのに必要とされる深さと幅で、複数のEBRブロックを連結して生成されます。

ROMモードで、アドレスポートはメモリアレイの入力レジスタがあります。メモリの出力データへのレジスタはオプションです。

ROMの様々なポートとそれらの定義は表10-11に示されます。表はIPexpressによって生成されたモジュールとROMプリミティブの対応するポートを記載します。

表10-11 EBRベースのROMポート定義

生成されたモジュールの ポート名	EBRブロック・プリミ ティブでのポート名	記述	アクティブなステート
Address	AD[x:0]	リードアドレス	—
OutClock	CLK	クロック	立ち上がりエッジ
OutClockEn	CE	クロック・イネーブル	Highアクティブ
Reset	RST	リセット	Highアクティブ
—	CS[2:0]	チップセレクト	—

リセット(RST)はRAMの入出力レジスタをリセットするだけで、メモリの内容をリセットしません。

EBRプリミティブにあるCS (チップセレクト) ポートは、メモリが複数のEBRブロックを連結される必要があるときに役立ちます。この場合3ビットのCS信号はアドレスのMSBとなり、8つのメモリまで容易に連結できます。ユーザによって指定されたメモリサイズが8つ以上のEBRブロック以上を必要とする場合、ソフトウェアは自動的にPFUで(EBRブロックへの外部)で実装される追加アドレス・デコード・ロジックを生成します。

IPexpressを用いてROMを生成する際に、ユーザはROMの内容を事前に初期化するために初期化ファイルを用意しなければなりません。これらのファイルは *.mem(拡張子)で、バイナリ、16進(Hex)またはAddresses Hex(ORCA)形式のいずれかです。初期化ファイルについては、このテクニカルノートの " メモリの初期化 " セクションで詳細に議論します。

ユーザはリードオンリ・メモリ(ROM)の出力レジスタをイネーブルするかどうか選択できます。図10-23から図10-24はこれらのオプションに対するROMの内部タイミング波形をそれぞれ示しています。

それぞれのEBRブロックは18,432ビットのRAMから成ります。各EBRブロックのアドレス値(x)とデータ値(y)の組み合わせは表10-12の通りです。

表10-12 LatticeXP2の16KメモリのROMサイズ

ROM	出力データ	アドレスポート {MSB:LSB}
16K x 1	DOA	WAD[13:0]
8K x 2	DOA[1:0]	WAD[12:0]
4K x 4	DOA[3:0]	WAD[11:0]
2K x 9	DOA[8:0]	WAD[10:0]
1K x 18	DOA[17:0]	WAD[9:0]
512 x 36	DOA[35:0]	WAD[8:0]

表10-13はリードオンリ・メモリ(ROM)で利用できる種々アトリビュートを示します。これらのアトリビュートのいくつかはIPexpress GUIを通してユーザが選択可能です。詳細なアトリビュート定義についてはAppendix Aを参照してください。

表10-13 LatticeXP2のROMアトリビュート定義

アトリビュート	記述	値	デフォルト値	IPexpressを通して 選択ユーザが可能
Address Depth	アドレス深さ	16K, 8K, 4K, 2K, 1K, 512		YES
Data Width	データ幅	1, 2, 4, 9, 18, 36	1	YES
Enable Output Register	レジスタモード(パイプライン)	NOREG, OUTREG	NOREG	YES
Enable GSR	グローバルセット・リセットのイネーブル	ENABLE, DISABLE	ENABLE	YES
Reset Mode	リセットタイプの選択	ASYN, SYNC	ASYN	YES
Memory File Format		BINARY, HEX, ADDRESSSED HEX		YES
Chip Select Decode	チップセレクト・デコード	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO

図10-19 ROMタイミング波形 ~ 出力レジスタなし

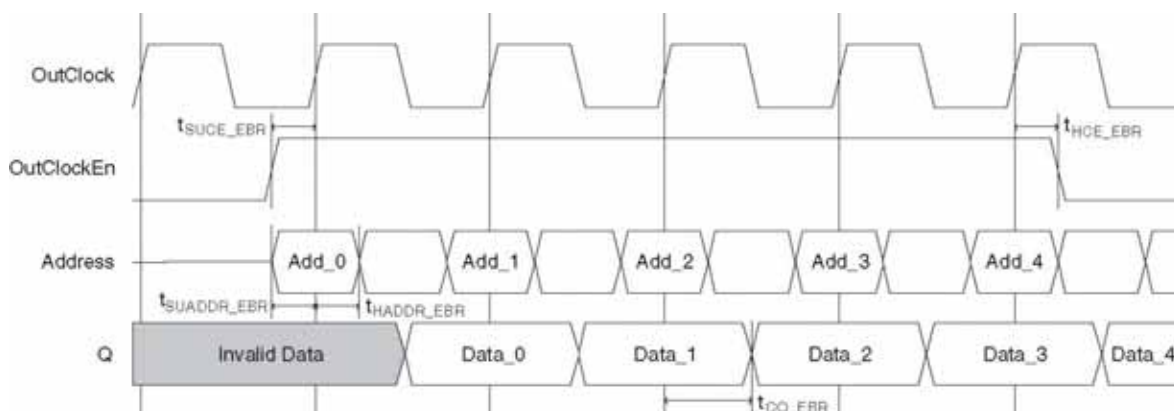
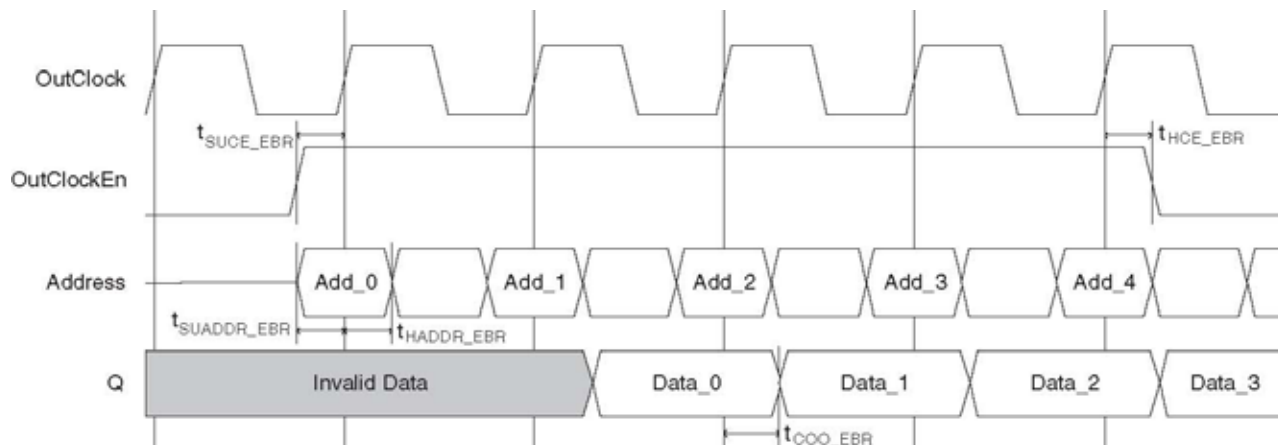


図10-20 ROMタイミング波形 ~ 出力レジスタあり



ファーストイン・ファーストアウト(FIFO, FIFO_DC) ~ EBRベース

FIFOはLatticeECP/EC、LatticeECP2/M、LatticeXP、およびMachXOの或るデバイスではサポートされていません。

ハードウェアには、シングルポート(RAM_DQ)、擬似デュアルポート(RAM_DP)、及び(真の)デュアルポート(RAM_DP_TRUE) RAMに構成することができる組み込みブロックRAM (EBR)があります。ユーザは擬似デュアルポートRAMを用いて、ispLEVERデザイン・ソフトウェアのIPexpressポイントツールを活用して、FIFOとFIFO_DCを構築できます。

これらFIFOは出力レジスタあり(パイプライン)、又はなし(非パイプライン)としてそれぞれ構成することができます。パイプライン・モードでは、ユーザにはRdEn信号で出力レジスタをイネーブルする付加的なオプションがあります。以下のセクションでFIFOの動作について議論します。

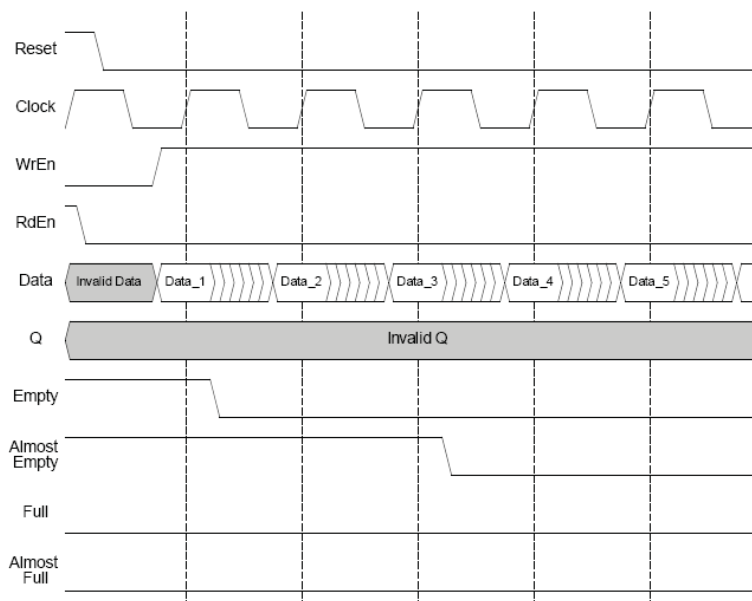
ファーストイン・ファーストアウト(FIFO) メモリ

FIFO (または単一クロックFIFO) はエミュレートFIFOです。アドレスとフラグのロジック回路はRAM周りのFPGAファブリックに実装されます。

FIFOで利用できるポートは以下の通りです。

- Reset
- Clock
- WrEn
- RdEn
- Data
- Q
- Full フラグ
- Almost Full フラグ
- Empty フラグ
- Almost Empty フラグ

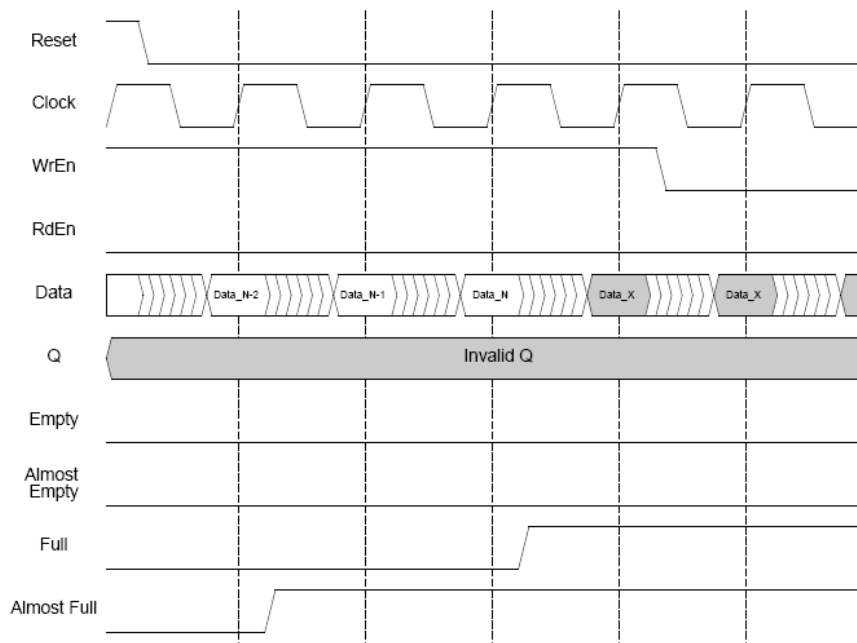
まず最初に、非パイプラインの(出力レジスタのない)FIFOについて議論します。図10-21は、いつ空(empty)の状態で、またいつデータが書き始められるかというFIFO動作を示します。

図10-21 出力レジスタのないIFIFO、データライト・サイクルの初め

FIFOに書き始めるためにはWrEn信号はHighでなければなりません。初めはEmptyとAlmost EmptyフラグがHighで、またFullとAlmost FullはLowです。

最初のデータがFIFOに書かれると、FIFOはもう空ではないのでEmptyフラグはネゲートされ(Lowになり)ます。この図では、Almost Emptyのフラグ設定が3 (アドレス位置の3)であると想定しています。したがって、3番目のアドレス位置が満たされると、Almost Emptyフラグはネゲートされます。

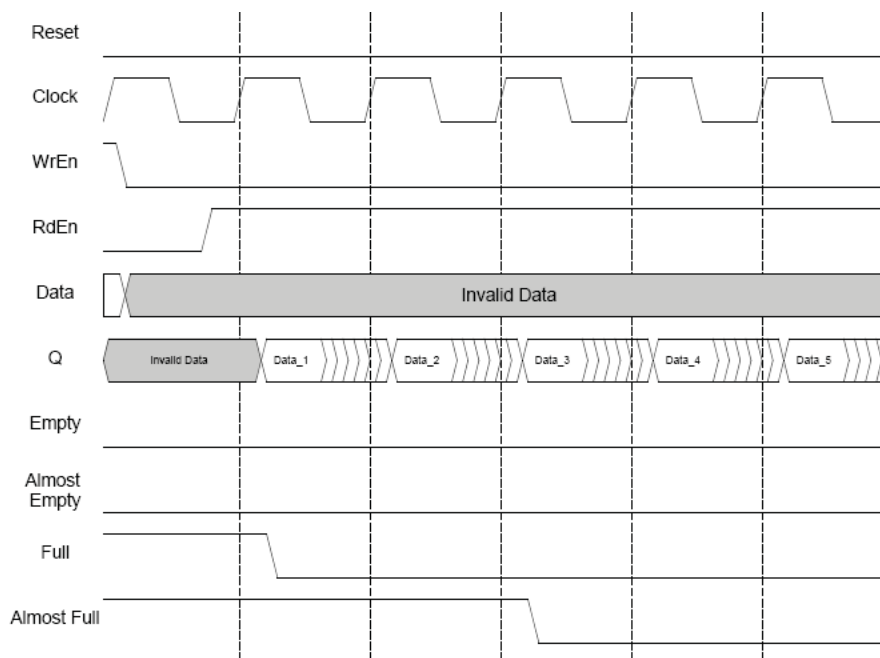
今、FIFOを満たすために書き続けると仮定します。FIFOが満たされると、Almost FullとFullの両フラグがアサートされます。図10-22はこれらのフラグの動きを示します。この図ではFIFOの深さが 'N' であると想定しています。

図10-22 出力レジスタのないIFIFO、データライト・サイクルの終わり

この場合、Almost FullフラグはFIFOが満たされる2つ前の位置にあります。FIFOのN-2位置に書かれるとAlmost Fullフラグはアサートされ、そして最終ワードが書かれるとFullフラグがアサートされます。FIFOがフルであるため(FullフラグがHigh)、Data_Xデータ入力は書かれません。

次にFIFOの内容が読み出されるとき波形を見ます。図10-23はリードサイクルの始まりを示します。RdEnがHighになり、データリードが始まります。示されているようにFullとAlmost Fullフラグはネゲートされます。

図10-23 出力レジスタのないIFIFO、データリード・サイクルの初め



同様に、データが読み出されてFIFOが空になると、Almost EmptyとEmptyフラグがアサートされます。

図10-24 出力レジスタのないIFIFO、データリード・サイクルの終わり

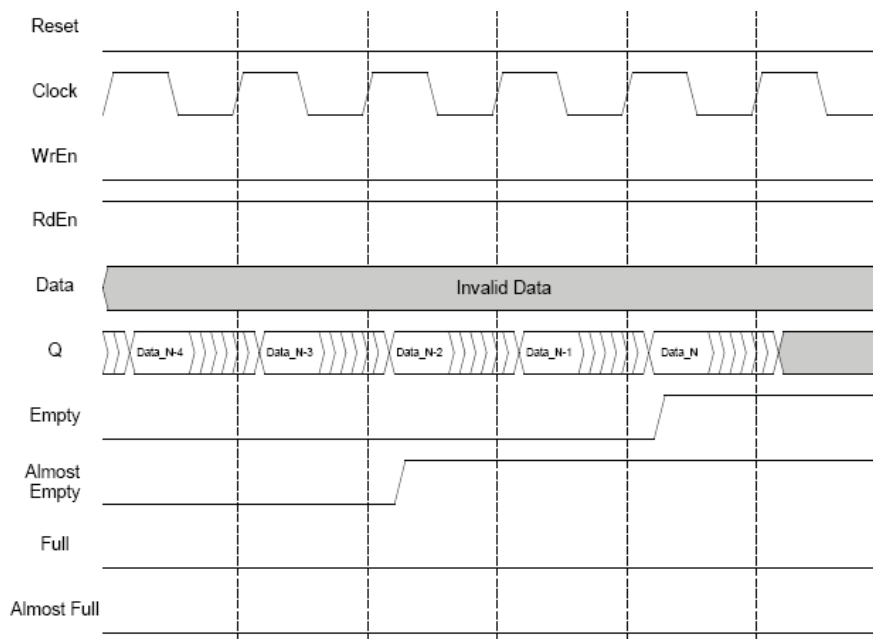


図10-21～10-24は出力レジスタのないFIFO（非パイプラインFIFO）の振舞いを示します。パイプライン・レジスタを有効にすると、出力データは1クロック周期遅れます。また、出力レジスタをRdEn信号によってイネーブルする付加的なオプションがあります。

図10-25から10-28は出力レジスタありFIFOとRdEnによる出力レジスタのイネーブルがあるFIFOの類似波形を示します。留意されるべきは、フラグが出力レジスタなしのFIFOと同様のタイミングでアサートされ、またネゲートされることです。1クロック周期遅れるのはデータ出力 'Q' のみです。

図10-25 出力レジスタのあるFIFO、データライト・サイクルの初め

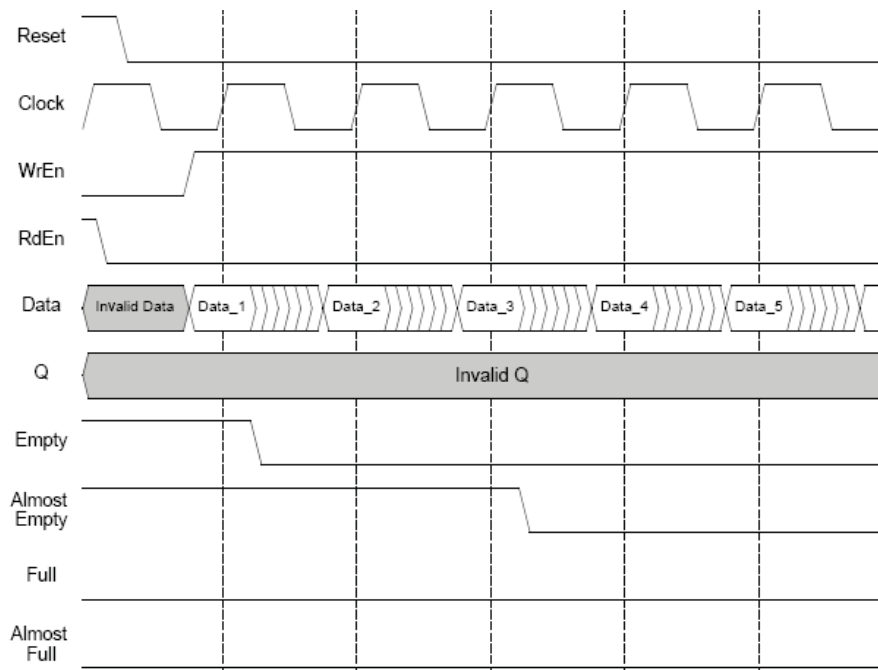


図10-26 出力レジスタのあるFIFO、データライト・サイクルの終わり

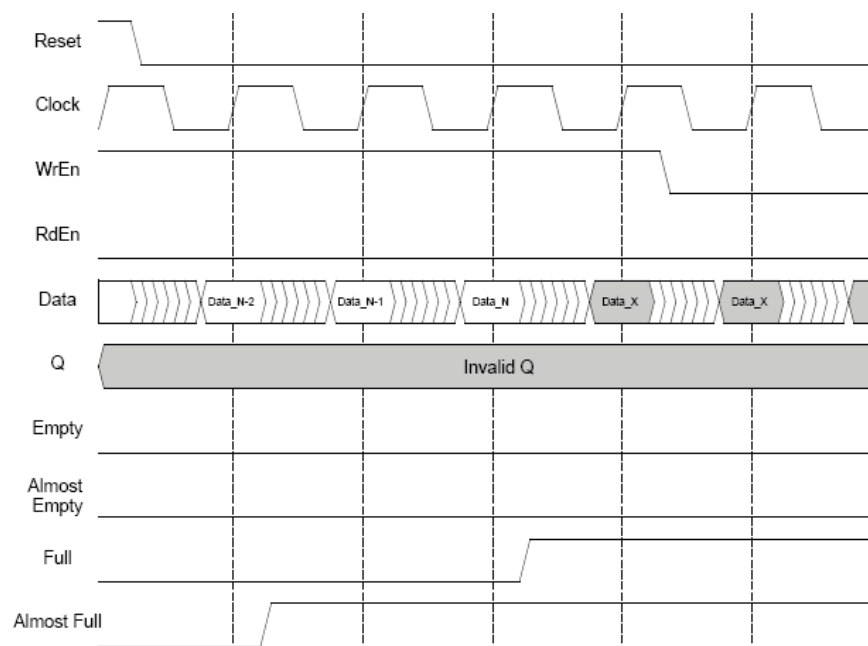
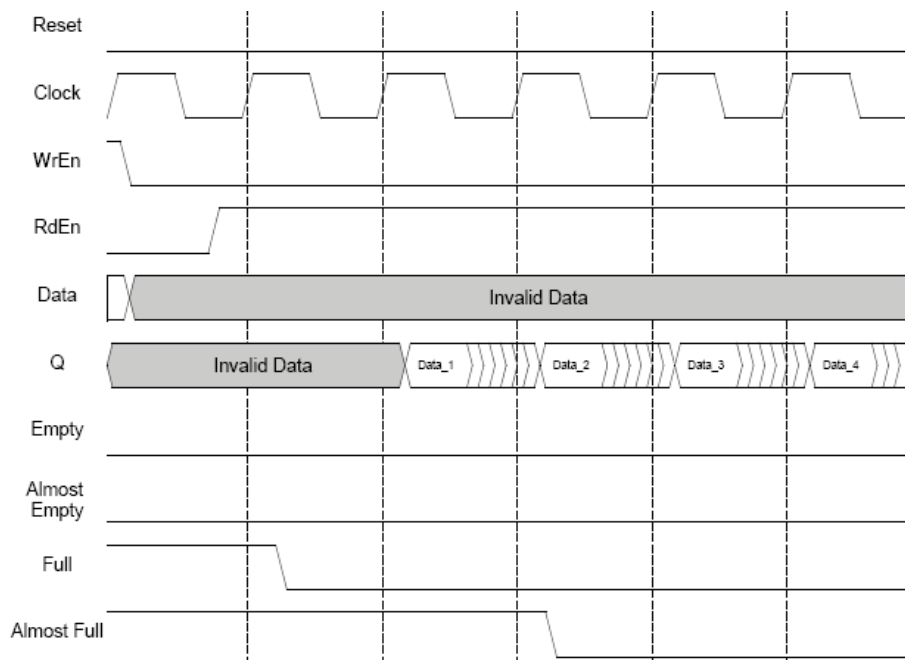
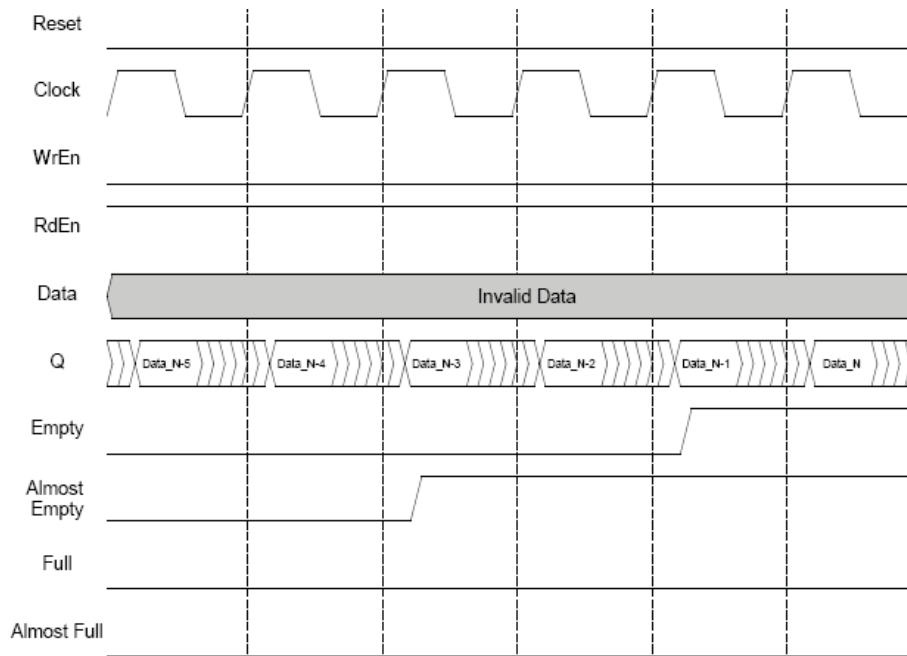
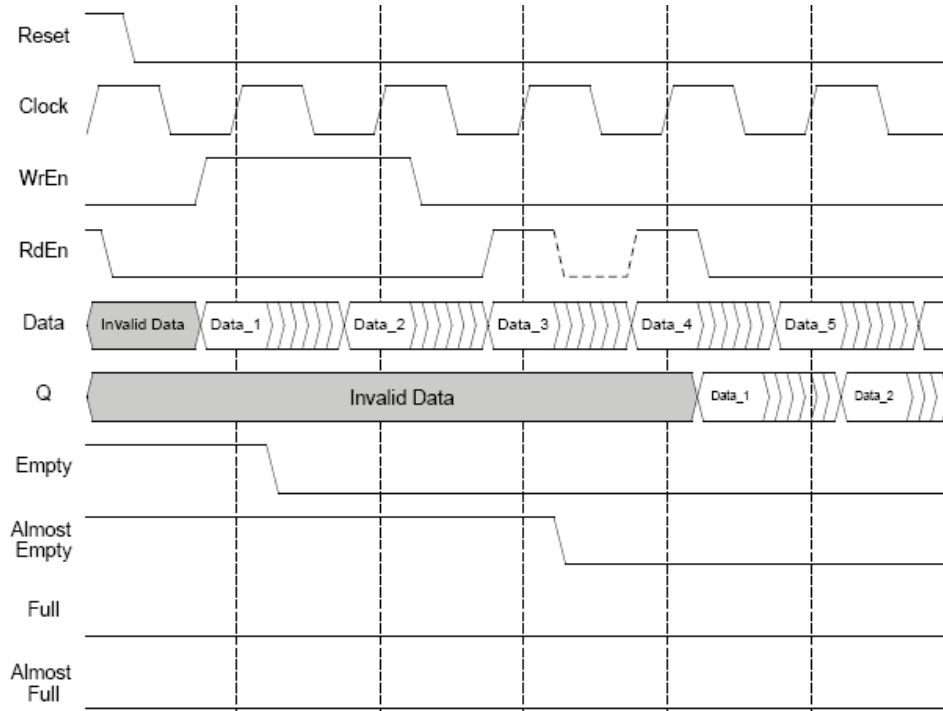


図10-27 出力レジスタのあるFIFO、データリード・サイクルの初め**図10-28 出力レジスタのあるFIFO、データリード・サイクルの終わり**

そして最後に、RdEnで出力レジスタをイネーブルするオプションを選択すると、(非パイプラインのFIFOと比べて)データ出力をさらに1クロック周期遅らせます。また、RdEnもそのクロック周期の間Highであるべきで、さもなければデータはRdEnがTrueになる時の余分なクロック周期がかかります。

図10-29 出力レジスタと、出力レジスタにRdEnのあるFIFO**デュアルクロック・ファーストイン・ファーストアウト(FIFO_DC) メモリ**

FIFO_DC (またはデュアルクロックFIFO) もまたエミュレートFIFOです。FIFOと同様に、アドレスとフラグのロジック回路はRAM周りのFPGAファブリックに実装されます。FIFO_DCで利用できるポートは以下の通りです。

- Reset
- RPRreset
- WrClock
- RdClock
- WrEn
- RdEn
- Data
- Q
- Full フラグ
- Almost Full フラグ
- Empty フラグ
- Almost Empty フラグ

FIFO_DCフラグ

エミュレートFIFOとしてのFIFO_DCは、フラグがブロックRAM周りのFPGAロジックで実装される必要があります。2本のクロックのために、フラグがリードクロックからライトクロックに、また同様に逆方向にそれぞれクロックドメインをまたぐ必要があります。これはフラグのアサートやネゲートの際にレイテンシ(遅延)を加えます。レイテンシは(アサートかネゲートのどちらか)一方の場合だけ避けることができ、或いはこれら場合の間で分散することもできます。

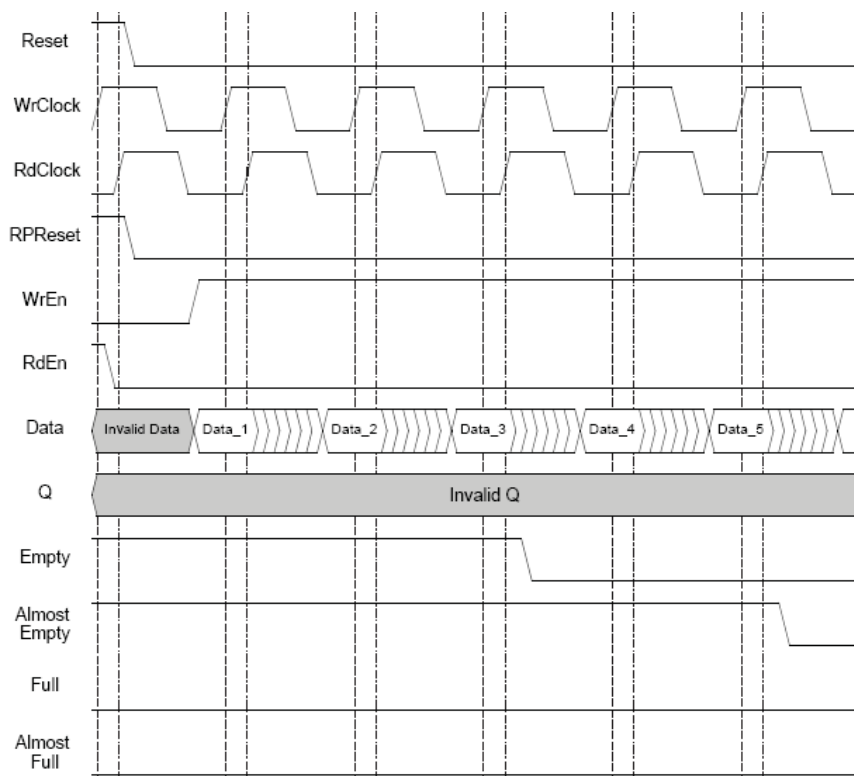
現行のエミュレートFIFO_DCでは、これらフラグのアサート時にレイテンシが全くなく、ラティスはこれが何よりも重要であると考えています。これらのフラグが真になる必要があるとき、レイテンシが全くありません。しかしながら、上述したように2クロックドメインで動作するフラグロジックの設計のために、ネゲートの際にはレイテンシがあります。

今、FIFO_DCを満たすために書き始めると仮定します。書き込み動作はWrClockとWrEnによって制御されます。しかしながら、EmptyとAlmost Emptyフラグのネゲートには付加的なRdClock周期がかかります。

他方、FullとAlmost FullのネゲートはFIFO_DCからのデータの読み出しが必要です。データを読んだ後、フラグが出力されるのに付加的なWrClock周期がかかります。

以上を理解した上で、出力レジスタなしFIFO_DCの波形を見ましょう。図10-30は、いつ空(empty)の状態か、またいつデータが書き始められるかというFIFO_DC動作を示します。

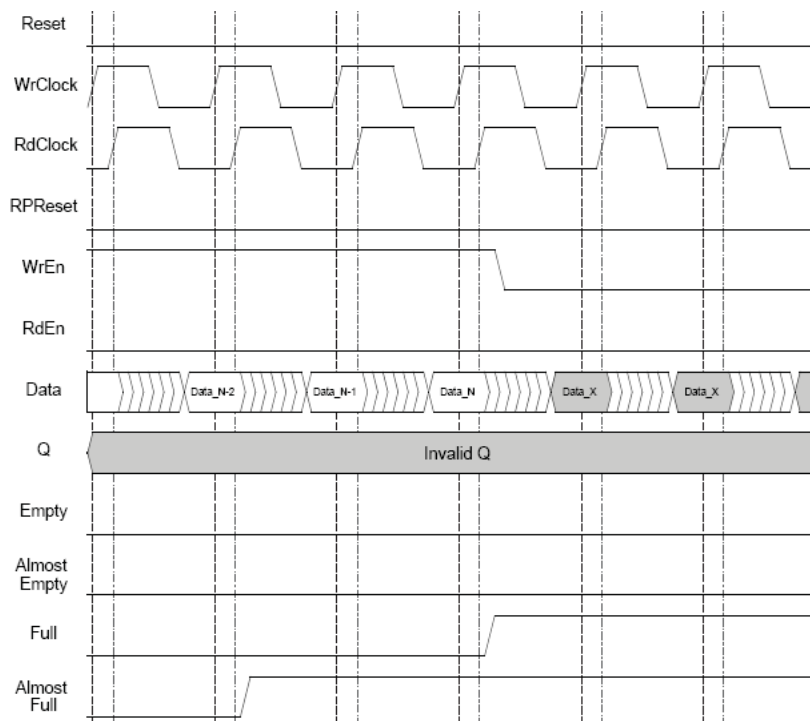
図10-30 出力レジスタのないFIFO_DC、データライト・サイクルの初め



FIFO_DCに書き始めるためにはWrEn信号はHighでなければなりません。初めはEmptyとAlmost EmptyフラグはHighで、またFullとAlmost FullはLowです。

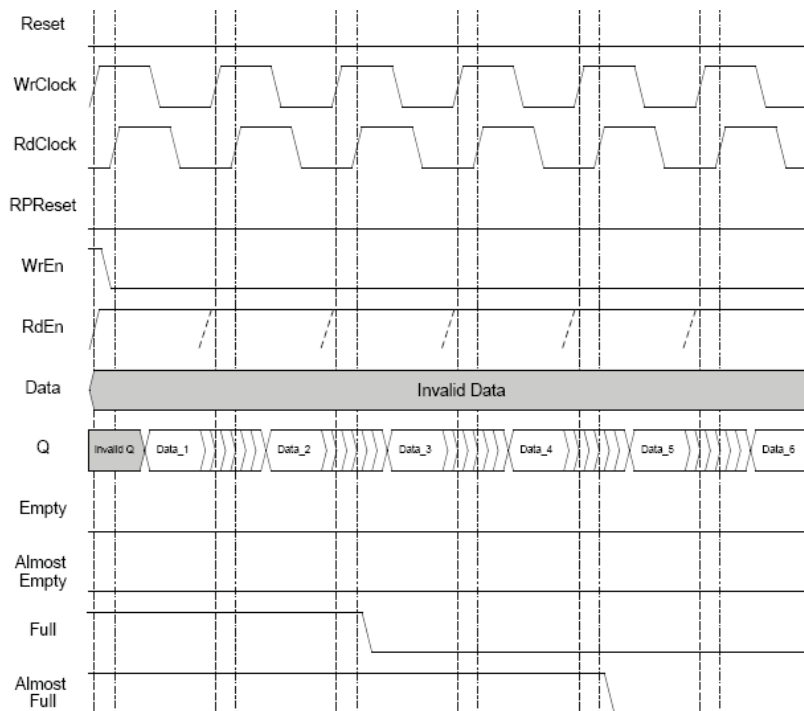
最初のデータがFIFOに書かれると、FIFO_DCはもう空ではないのでEmptyフラグはネゲートされ(Lowになります)。この図では、Almost Emptyのフラグ設定が3 (アドレス位置の3)であると想定しています。したがって、3番目のアドレス位置が満たされるとAlmost Emptyフラグはネゲートされます。

今、FIFO_DCを満たすために書き続けると仮定します。FIFO_DCが満たされると、Almost FullとFullの両フラグがアサートされます。図10-31はこれらフラグの動作を示します。この図ではFIFO_DCの深さが 'N' であると想定しています。

図10-31 出力レジスタのないIFIFO_DC、データライト・サイクルの終わり

この場合、Almost FullフラグはFIFO_DCが満たされる2つ前の位置にあります。FIFO_DCのN-2位置に書かれるとAlmost Fullフラグはアサートされ、そして最終ワードが書かれるとFullフラグがアサートされます。

FIFO_DCがフルであるため(FullフラグがHigh)、Data_Xデータ入力は書かれません。これらのフラグはすぐにアサートされ、そのレイテンシが全くないことに留意してください。

図10-32 出力レジスタのないIFIFO_DC、データリード・サイクルの初め

次にFIFO_DCの内容が読み出されときの波形を見ます。図10-32はリードサイクルの始まりを示します。RdEnがHighになり、データリードが始まります。示すようにAlmost FullとFullの両フラグがネゲートされます。この場合、ネゲートが2クロック周期遅れることに注意してください。

同様に、データが読み出されてFIFO_DCが空になると、Almost EmptyとEmptyフラグがアサートされます。

図10-33 出力レジスタのないFIFO_DC、データリード・サイクルの終わり

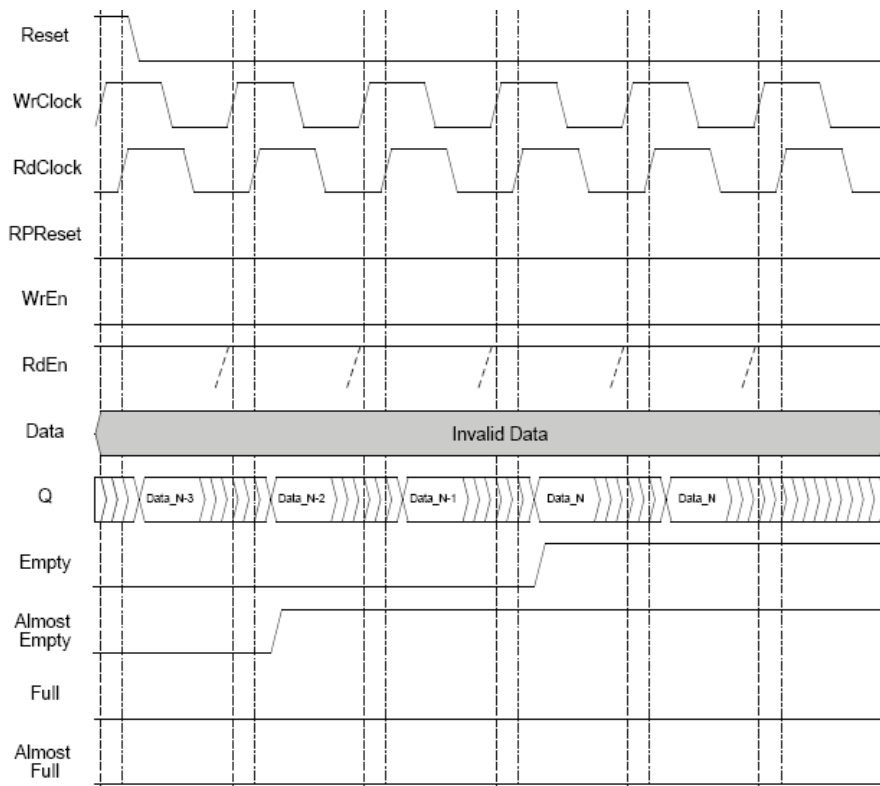


図10-33は出力レジスタのないFIFO_DC(非パイプラインFIFO_DC)の振舞いを示します。パイプライン・レジスタを有効にすると、出力データは1クロック周期遅れます。また、出力レジスタをRdEn信号によってイネーブルする付加的なオプションがあります。

図10-34から10-37は出力レジスタありFIFO_DCと、RdEnによる出力レジスタのイネーブルがないFIFO_DCの類似波形を示します。留意されるべきは、フラグが出力レジスタなしのFIFO_DCと同様のタイミングでアサートされ、またネゲートされることです。1クロック周期遅れるのはデータ出力 'Q' のみです。

図10-34 出力レジスタのあるFIFO_DC、データライト・サイクルの初め

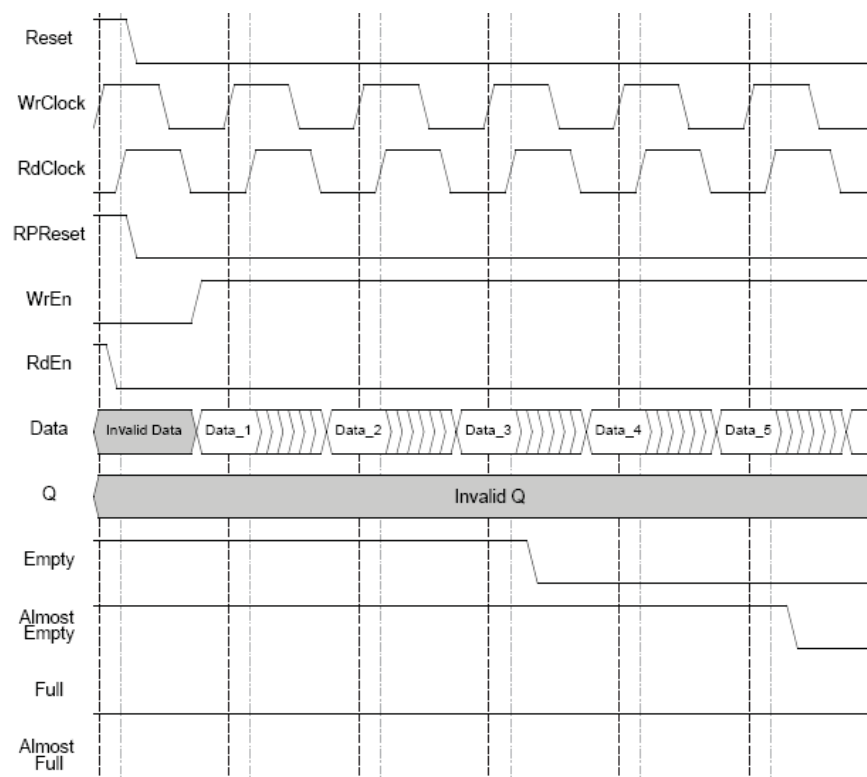


図10-35 出力レジスタのあるFIFO_DC、データライト・サイクルの終わり

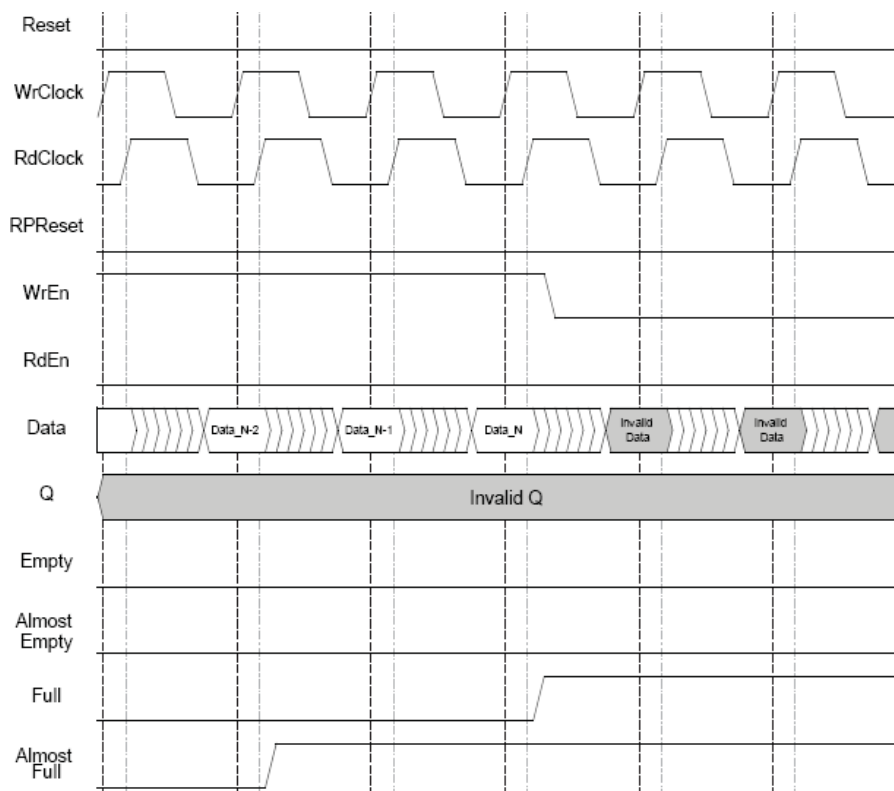


図10-36 出力レジスタのあるFIFO_DC、データリード・サイクルの初め

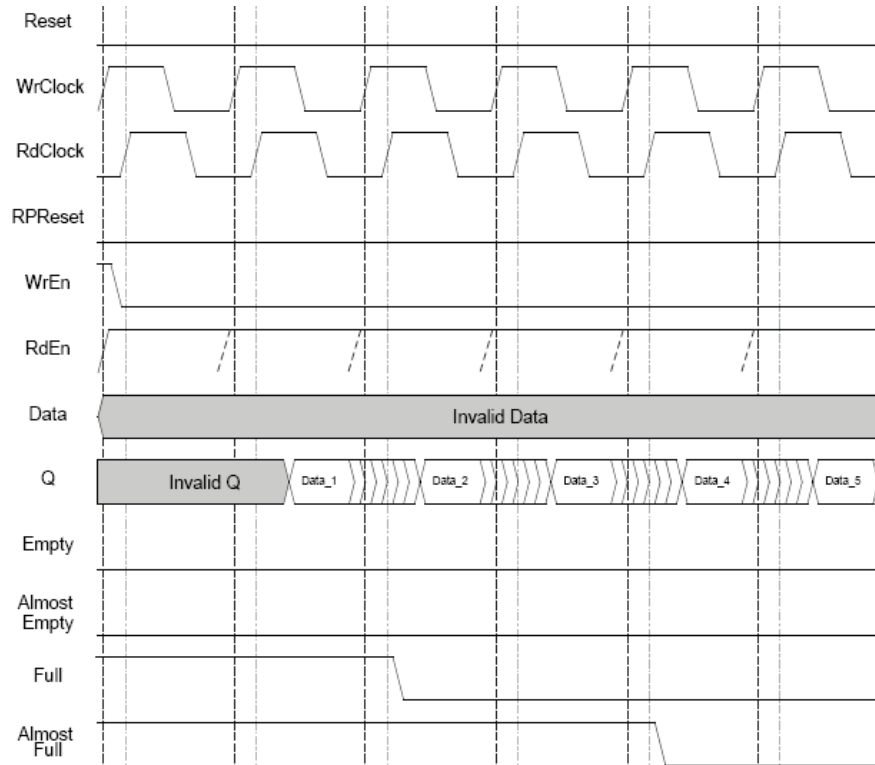
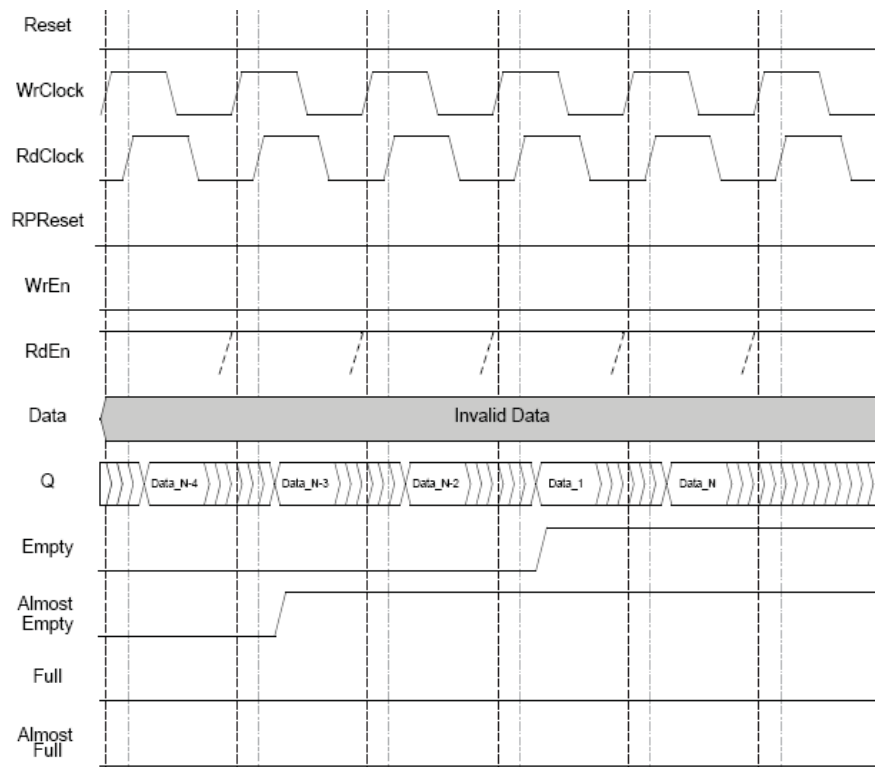


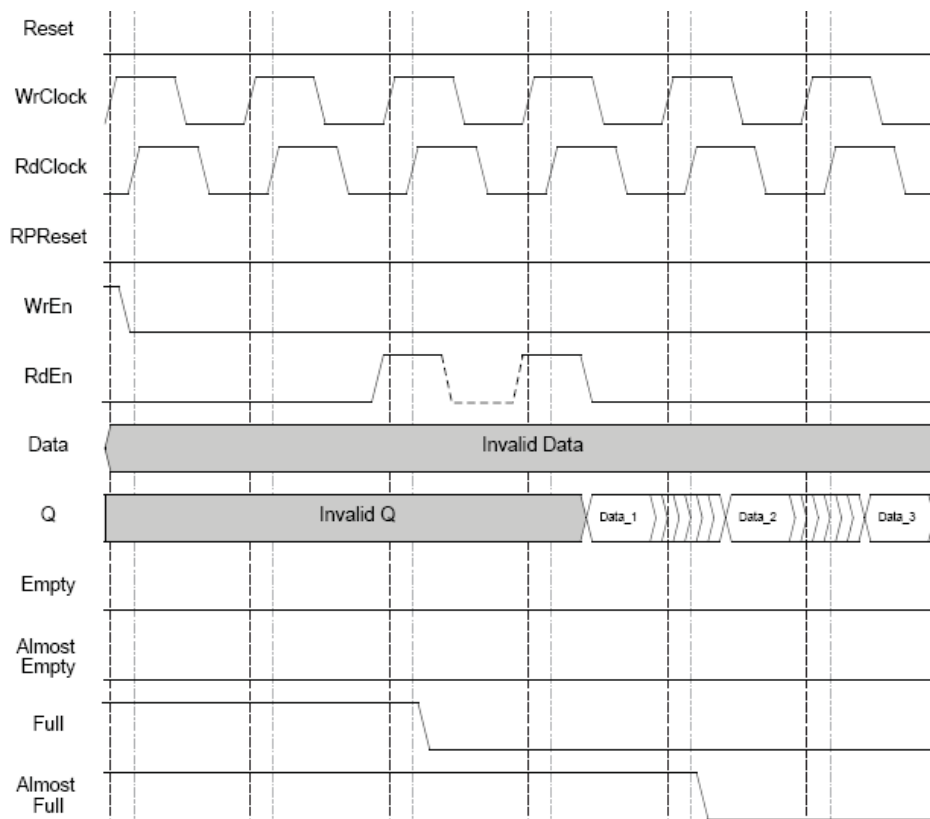
図10-37 出力レジスタのあるFIFO_DC、データリード・サイクルの終わり



そして最後に、RdEnで出力レジスタをイネーブルするオプションを選択すると、(非パイプラインの

FIFO_DCと比べて)データ出力をさらに1クロック周期遅れます。また、RdEnもそのクロック周期の間Highであるべきで、さもなければデータはRdEnが真になる時の付加的なクロック周期がかかります。

図10-38 出力レジスタと、出力レジスタにRdEnのあるFIFO_DC

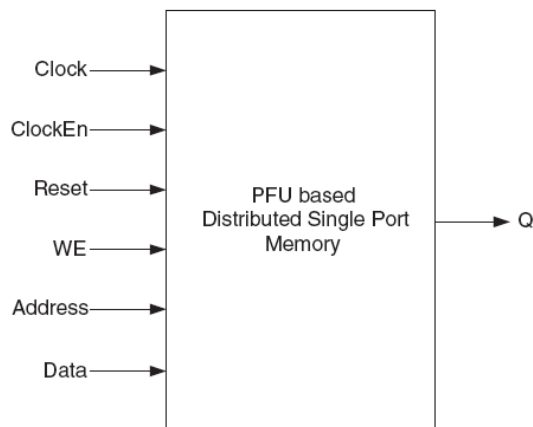


分散シングルポートRAM (Distributed_SPRAM) ; PFUベース

PFUベースの分散シングルポートRAMは、PFUで利用する4入力LUT(ルックアップ・テーブル)を用いることで生成されます。より大きい分散メモリサイズを作成するためにこれらのLUTを連結することができます。

図10-39はIPexpressによって生成される分散シングルポートRAMモジュールを示します。

図10-39 IPexpress によって生成される分散シングルポートRAMモジュール



生成されたモジュールはPFU内にある4入力LUTを利用します。Clock、ClockEn、およびResetのような追加

ロジックは、PFU内にあるリソースを利用することによって生成されます。

リードクロック(RdClock)やリードクロック・イネーブル(RdClockEn)などのポートはハードウェア・プリミティブでは利用できません。IPexpress内のコンフィグレーションで、ユーザが出力レジスタをイネーブルすると、これに対応したモジュールが生成されます。

メモリの各ポートとその定義は表10-14にまとめられています。この表のリストはIPexpressで生成されたモジュールとプリミティブに対応します。

表10-14 PFUベース分散シングルポートRAMのポート定義

生成されたモジュールのポート名	PFUプリミティブのポート名	記述	アクティブなステート
Clock	CK	クロック	立ち上がりエッジ
ClockEn	-	クロック・イネーブル	Highアクティブ
Reset	-	リセット	Highアクティブ
WE	WRE	ライトイネーブル	Highアクティブ
Address	AD[3:0]	アドレス	—
Data	DI[1:0]	データ入力	—
Q	DO[1:0]	データ出力	—

Clock Enable(ClockEn)などのポートはハードウェア・プリミティブにはありません。これらは、ユーザが出力レジスタをイネーブルしする設定を選択すると、IPexpressによって生成されます。

分散シングルポートRAM(Distributed SPRAM)の出力レジスタをイネーブルするかどうかのオプションがあります。これらに対する内部タイミング波形を、それぞれ図10-40から図10-41に示します。

図10-40 PFUベース分散シングルポートRAMタイミング波形 ~ 出力レジスタなし

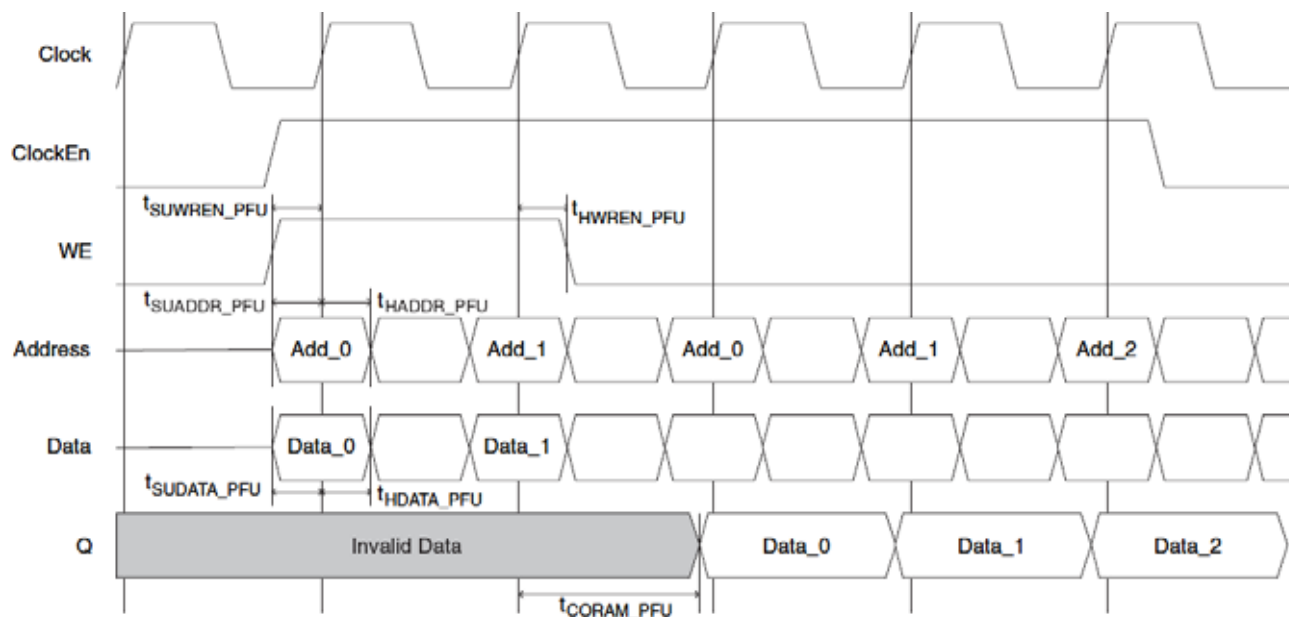
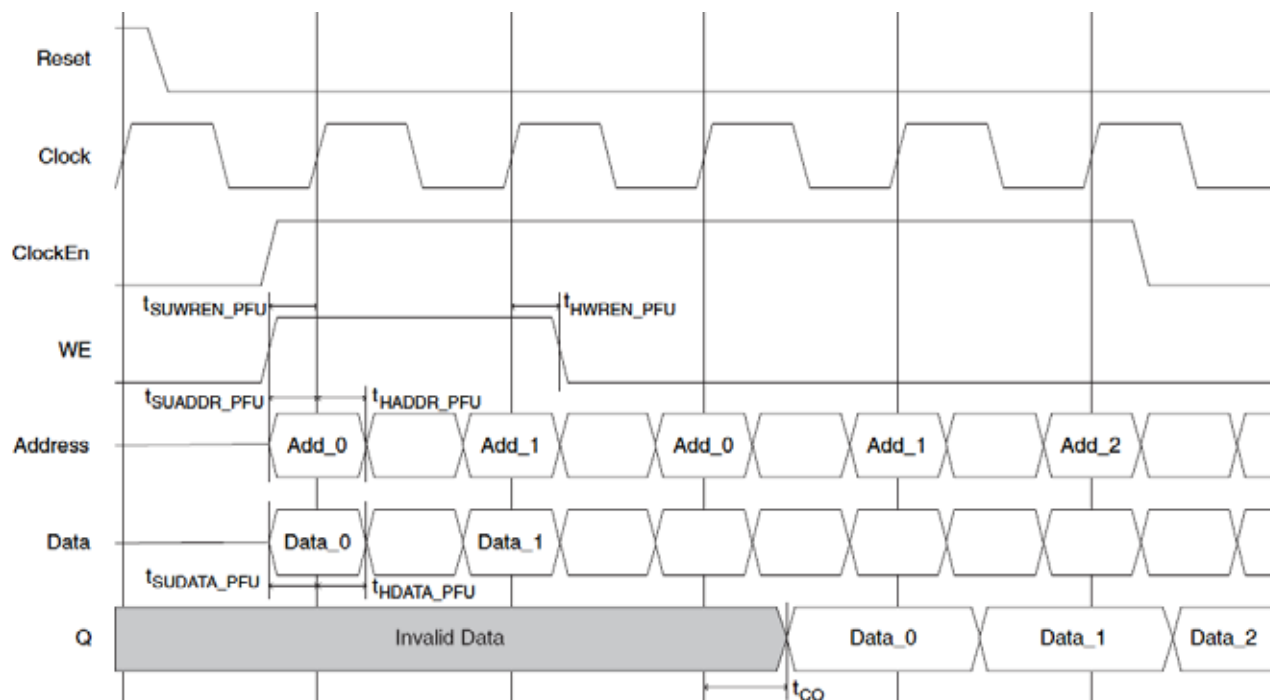


図 10-41 PFUベース分散シングルポートRAMタイミング波形 ~ 出力レジスタあり

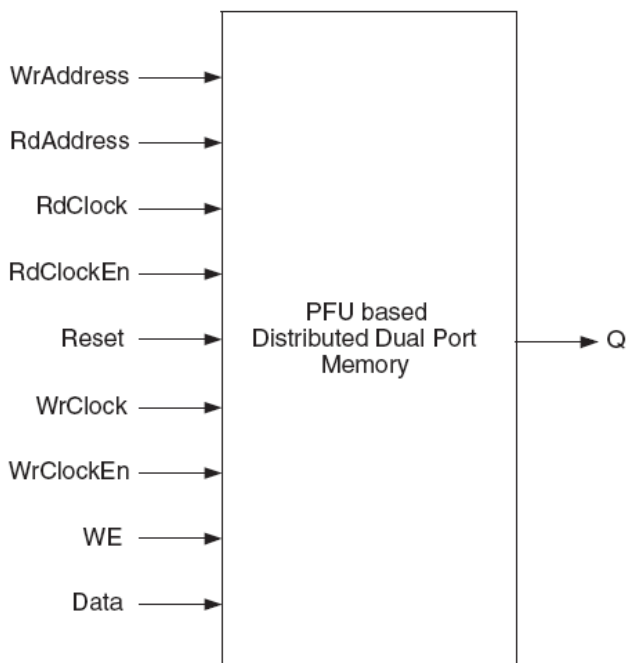


分散デュアルポートRAM (Distributed_DPRAM) ~ PFUベース

また、PFUベースの分散デュアルポートRAMは、PFUで利用する4入力LUTを用いることで生成されます。より大きい分散メモリサイズを作成するためにこれらのLUTを連結することができます。

図10-42はIPexpressによって生成される分散シングルポートRAMモジュールを示します。

図10-42 IPexpressによって生成される分散デュアルポートRAMモジュール



生成されたモジュールはPFU内にある4入力LUTを利用します。Clocks、Clock Enables、およびResetのための追加ロジックは、PFUにあるリソースを利用することによって生成されます。

リードクロック(RdClock)やリードクロック・イネーブル(RdClockEn)などのポートはハードウェア・プリミティブでは利用できません。モジュール・マネージャ内のコンフィグレーションで、ユーザが出力レジスタをイネーブルすると、これに対応したモジュールが生成されます。

メモリの種々ポートとそれらの定義は表10-15に示されています。表はIPexpressによって生成されたモジュールとプリミティブの対応するポートを記載します。

表10-15 PFUベースの分散デュアルポートRAMのポート定義

生成モジュールのポート名	PFUプリミティブのポート名	記述	アクティブなステート
WrAddress	WAD[3:0]	ライトアドレス	—
RdAddress	RAD[3:0]	リードアドレス	—
RdClock	—	リードクロック	立ち上がりエッジ
RdClockEn	—	リードクロック・イネーブル	Highアクティブ
WrClock	WCK	ライトクロック	立ち上がりエッジ
WrClockEn	—	ライトクロック・イネーブル	Highアクティブ
WE	WRE	ライトイネーブル	Highアクティブ
Data	DI[1:0]	データ入力	—
Q	RDO[1:0]	データ出力	—

分散デュアルポートRAM(Distributed DPRAM)の出力レジスタをイネーブルするかどうかのオプションがあります。これらに対する内部タイミング波形を、それぞれ図10-43から図10-44に示します。

図10-43 PFUベース分散デュアルポートRAMタイミング波形 ~ 出力レジスタなし

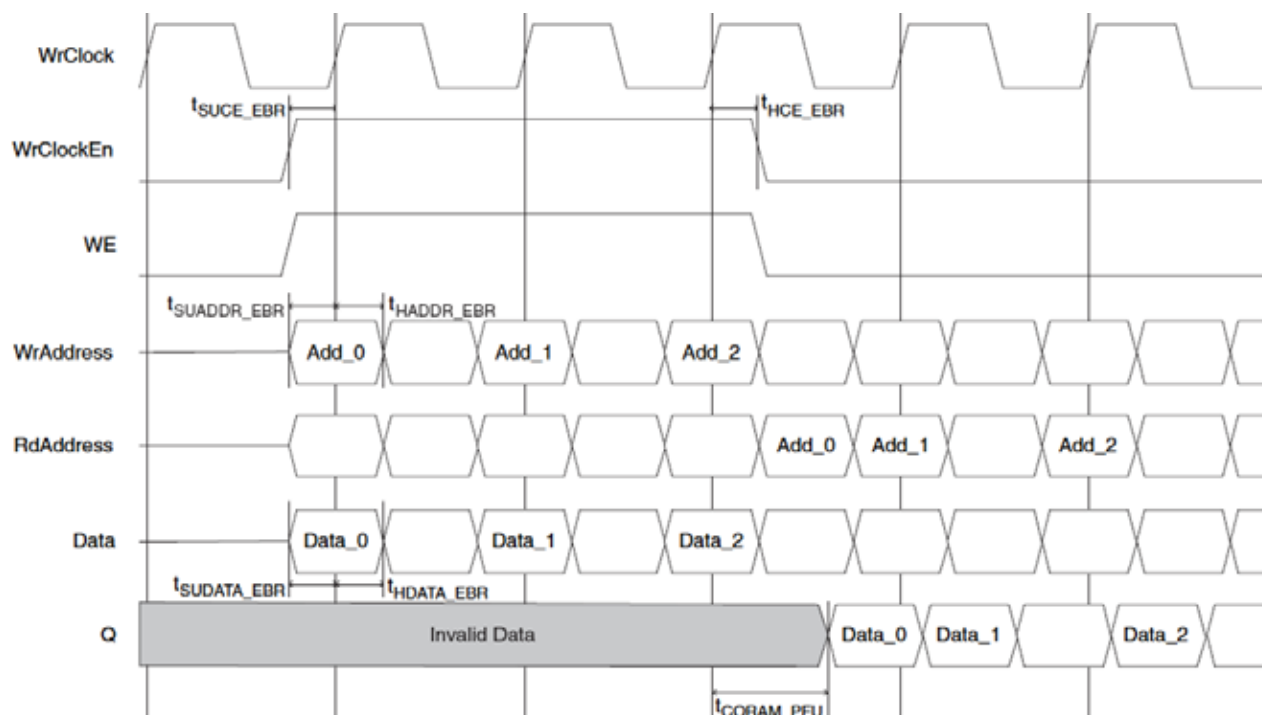
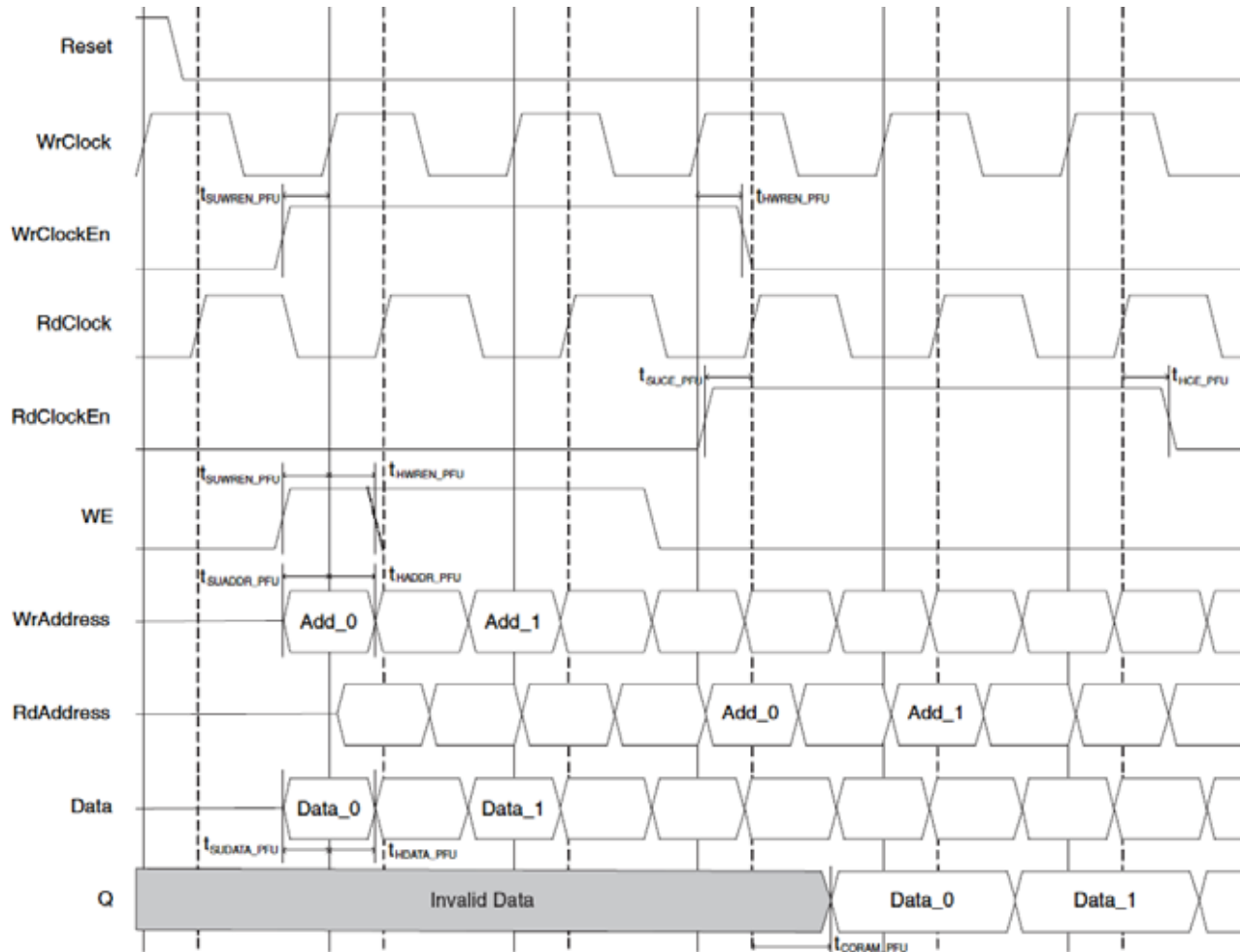


図10-44 PFUベース分散デュアルポートRAMタイミング波形 ~ 出力レジスタあり

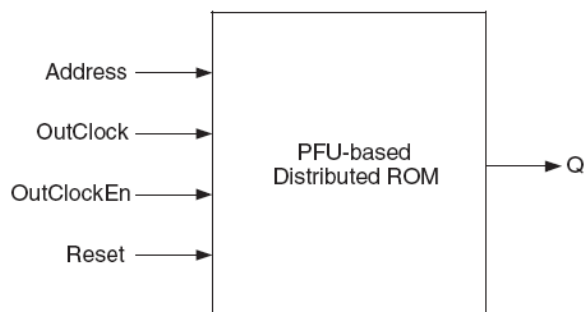


分散ROM (Distributed_ROM) ~ PFUベース

また、PFUベースの分散ROMは、PFUにある4入力LUTを用いることで作成されます。より大きい分散メモリサイズを作成するためにこれらのLUTを連結することができます。

図10-45はIPexpressによって生成される分散シングルポートRAMモジュールを示します。

図10-45 IPexpressによって生成される分散ROM



生成されるモジュールはPFUにある4入力LUTを利用します。ClockやResetなどの追加ロジックはPFUにあるリソースを活用することで生成できます。

Out Clock(OutClock)やOut Clock Enable(OutClockEn)などのポートはハードウェア・プリミティブで利用できません。IPexpress内のコンフィグレーションで、ユーザが出力レジスタをイネーブルすると、これに対応したモジュールが生成されます。

メモリの種々ポートとそれらの定義は表10-16に示されています。表はIPexpressによって生成されたモジュールとプリミティブの対応するポートを記載します。

表10-16 PFUベースの分散ROMポート定義

生成されたモジュールのポート名	PFUプリミティブのポート名	記述	アクティブなステート
Address	AD[3:0]	アドレス	—
OutClock	—	出力クロック	立ち上がりエッジ
OutClockEn	—	出力クロック・イネーブル	Highアクティブ
Reset	—	リセット	Highアクティブ
Q	DO	データ出力	—

分散ROM(Distributed ROM)の出力レジスタをイネーブルするかどうかのオプションに対する内部タイミング波形を、それぞれ図10-46から図10-47に示します。

図10-46 PFUベース分散ROMタイミング波形 ~ 出力レジスタなし

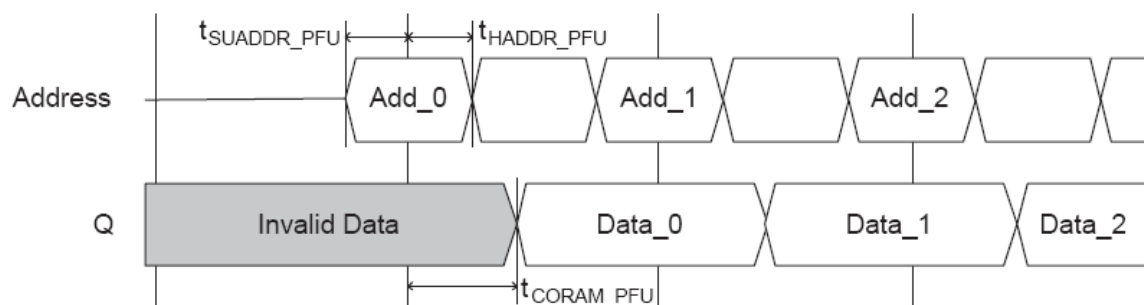
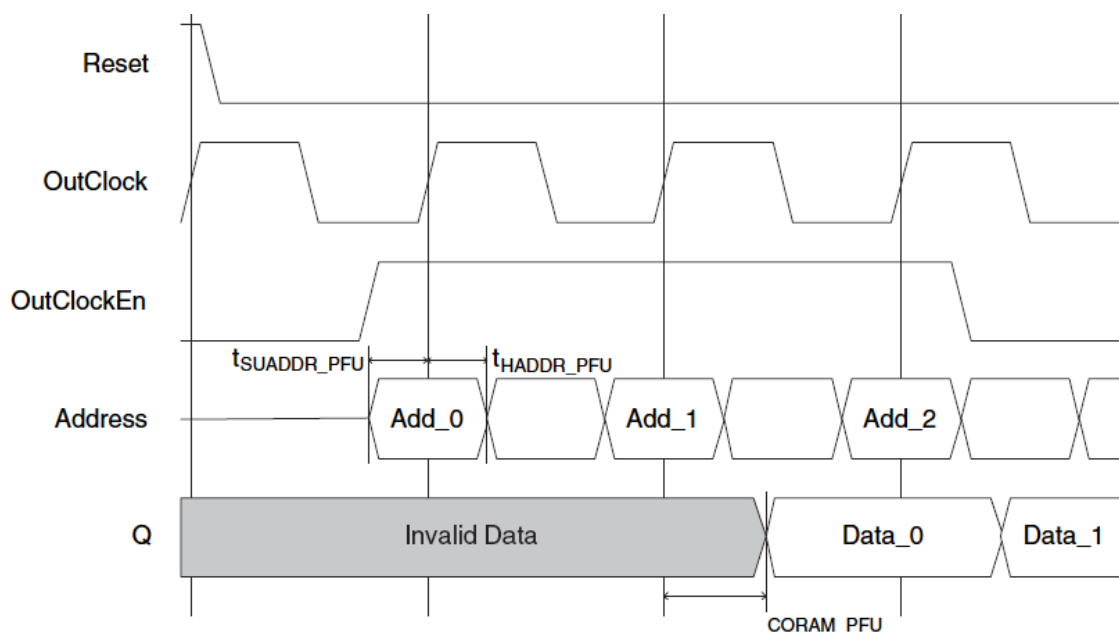


図10-47 PFUベース分散ROMタイミング波形 ~ 出力レジスタあり



ユーザ・タグメモリ

タグメモリは不揮発性記憶に用いることができるオンチップのフラッシュ領域です。あたかも外付けSPIフラッシュであるかのように、ユーザ・デザインからアクセスされます。SPIバス動作モード0 (0、0)と3 (1、1)の両方がサポートされます。

図10-48 SSPIA プリミティブ



表10-17 ユーザ・タグメモリ

プリミティブのポート名	記述
SI	データ入力
SO	データ出力
CLK	クロック
CS	チップセレクト

信号記述

タグメモリの基本仕様

各LatticeXP2デバイスにはタグメモリが1ページあります。

ページサイズは56 ~ 451バイトの範囲です。

表10-18 タグメモリの規模

デバイス	タグメモリ・サイズ [bit]	タグメモリ・サイズ [Byte]
XP2-5	632	79
XP2-8	768	96
XP2-17	2184	273
XP2-30	2640	330
XP2-40	3384	423

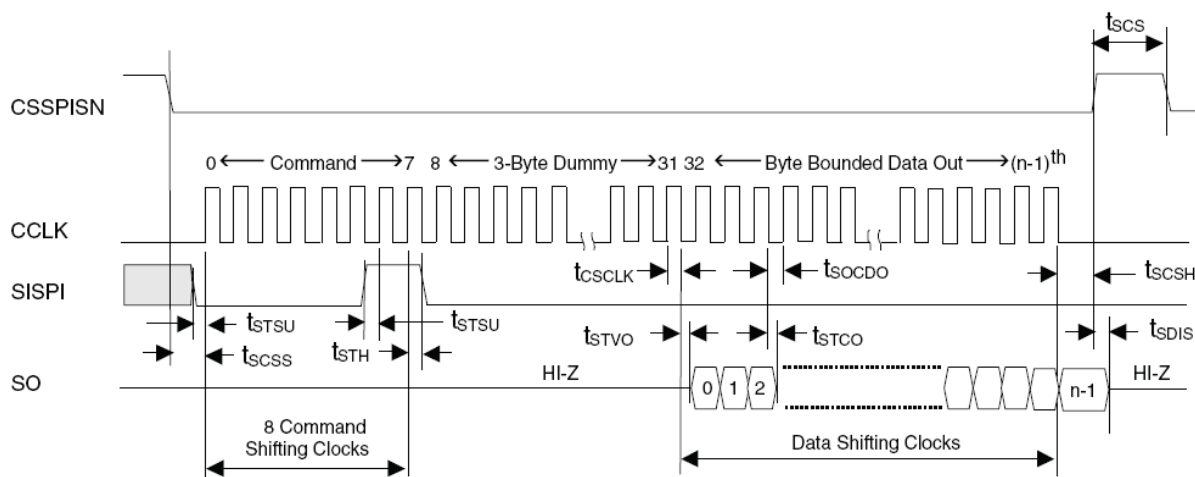
注: 最初にWRITE_ENがコマンドをロードせずにREAD_TAGコマンドが発行されると、デバイスはタグフラッシュからデータ・シフトレジスタまでデータを転送するために、最大を20usecの余分な時間を必要とします。

表10-19 タイミング仕様

シンボル	パラメータ	Min.	Max.	単位
f_{MAXSPI}	スレーブSPI CCLKクロック周波数		25	MHz
t_{RF}	クロック / データ入力のRise / Fall時間		20	ns
t_{CSCLK}	スレーブSPI CCLKクロックのHigh期間	20		ns
t_{SOCDO}	スレーブSPI CCLKクロックのLow期間	20		ns
t_{SCS}	CSSPINのHigh期間	25		ns
t_{SCSS}	CSSPINのセットアップ時間	25		ns
t_{SCSH}	CSSPINのホールド時間	25		ns
t_{TSU}	スレーブSPIデータ入力のセットアップ時間	5		ns
t_{STH}	スレーブSPIデータ入力のホールド時間	5		ns
t_{STVO}	スレーブSPI出力有効 (WRITE_ENの後)		20	ns
	スレーブSPI出力有効 (WRITE_ENなし) (1)	3	20	us
t_{STCO}	スレーブSPI出力ホールド時間	0		ns
t_{SDIS}	スレーブSPI出力ディセーブル時間		100	ns

注；最初にWRITE_ENがコマンドをロードせずにREAD_TAGコマンドが発行されると、デバイスはタグフラッシュからデータ・シフトレジスタまでデータを転送するために、最大を20usecの余分な時間を必要とします。

図10-49 一般的なタイミング・ダイアグラム



SSPIAモジュールは内部モジュールであるため、他のソフトモジュール一般のI/Oと同様にこれらのI/Oを扱うことができます。したがって、他の内部モジュールにそれらを接続したり、またはI/Oパッドに出すことができます。sysCONFIGポートピンへの接続が推奨です。

表10-20 コマンドの使い方

コマンド名	OPCODE	バイト1~3 ^{1,2}	データ	遅延時間	記述
READ_TAG	0x4E	ダミー	出力	5usec min.	タグメモリのリード
PROGRAM_TAG	0x8E	ダミー	入力	1ms min., 25ms max.	タグメモリのプログラム
ERASE_TAG	0x0E	ダミー		100ms min., 1000ms max.	タグメモリの消去

- データバイトは最上位ビット(MSB)が最初にシフトされます。
- バイト1~3は、デバイスがコマンドを実行するために付加的なタイミングを与えるための、ダミークロックです。

これらのダミークロック期間のSIピン上のデータは、図示されるような0x00である必要はなく、どのような

値でも構いません。

概要

LatticeXP2ファミリ・デバイスはインスタントオンで、また常時利用できるスタンドアロンのタグメモリ機能を持つ設計です。

タグメモリは、組み込まれているシリアル・ペリフェラル・インターフェイス(SPI)ポートや、JTAGポートからアクセスしやすい単一ページのフラッシュ不揮発性メモリとして構成されています。

スタンドアロン・タグメモリは、重要なデータや、ボードのシリアル番号、ボード改版ログ、およびプログラム化パターン識別などのためのスクラッチパッド（メモ書き用）メモリとして理想的です。

LatticeXP2デバイスファミリへのタグメモリの集積は、チップ数とボード面積を節約します。また、前世代の小規模SPI EEPROMデバイスの置き替えに用いることもできます。

組み込みSPIインターフェイスがあるため、デバイスのSPIインターフェイスを有効にする目的で、最初にコンフィグレーション・フラッシュにプログラムする必要はありません。ラティスからデバイスを出荷するとき、既にインターフェイスがイネーブルされており、ボードテストの時間を節約します。

タグメモリ・フラッシュはLatticeXP2デバイスに集積されていますが、組み込みSPIインターフェイスのために、タグメモリは独立した機能として維持され、また常にアクセスできます。

また、フィールド・アップグレードにも組み込みSPIインターフェイスは重要です。デバイスが新しいパターンにフィールド・アップグレードされても、重要なデータをタグメモリで保持し、かつそのアクセスを保証することができます。

インスタントオン機能は、ラティスからデバイスを出荷するときSPIインターフェイスをイネーブルすることによって、達成されます。コンフィグレーション・フラッシュと異なり、デバイスのスタンダード / アドバンスト・セキュリティ設定は、タグメモリのアクセス性に対しては無効です。したがって、タグメモリは常にアクセス可能です。

また他のフラッシュ・フューズと同様に、JTAGポートでIEEE 1532準拠のプログラミング・フローを用いることで、生産時のプログラミング支援やシステムデバッグのためにタグメモリをプログラムすることができます。

ピン記述

以下で説明されるピンは専用ピンではありません。タグメモリ機能が不要である場合、これらピンは通常のユーザI/Oになります。タグメモリ機能が必要な婆、内部SPIインターフェイスによってコアを介してタグメモリにアクセスすることができます。内部SPIインターフェイスは、タグメモリを先進のアプリケーションがサポートできるようにします。

例えば:

1. I²C - SPIトランスレータを使用して、SPIタグメモリをI²Cタグメモリ・デバイスになるように変換する
2. 4系統のSPIインターフェイスを他の4本のユーザI/Oに接続する

選択にはispLEVERデザインツールを用います。デフォルトでは、外部SPIインターフェイスがイネーブルされ、そしてタグメモリが選択されます。

以下に示すピンの機能記述は、内部と外部の両SPIインターフェイスに適用されます。

シリアルデータ入力 (SI)

SPI Serial Data Inputピンは、デバイスにコマンドとデータをシリアルで(シフトされ)書き込む手段を提供します。データはシリアルクロック(CLK)入力ピンの立ち上がりエッジでラッチされます。

シリアルデータ出力 (SO)

SPI Serial Data Outputピンは、デバイスからステータスとデータをシリアルに読み出す(シフトアウト)手段を提供します。データはシリアルクロック(CLK)入力ピンの立ち下がりエッジでシフトアウトされます。

シリアルクロック (CLK)

SPI Serial Clock Inputピンはシリアル入力/出力動作にタイミングを提供します。

チップセレクト (CS)

SPI Chip Selectピンは、SPIインターフェイス動作をイネーブル（選択）したりディセーブル（非選択）します。Chip SelectがHighのとき、SPIインターフェイスは非選択で、Serial Data Output (SO)ピンはハイインピーダンスです。それがLowにされるとSPIインターフェイスは選択され、コマンドをデバイスに書き、そしてデータをデバイスから読み出すことができます。パワーアップ後、新しいコマンドを受け付けるためにはCSはHighからLowに遷移しなければなりません。

SPI動作

SPIモード

SPIインターフェイスは4本の信号から成るSPI互換のバスを通してアクセスできます。これらはシリアルクロック(CLK)、チップセレクト(CS)、シリアルデータ入力(SI)、そしてシリアルデータ出力(SO)です。SPIバス動作モード0 (0, 0)と3 (1, 1)の両方がサポートされます。Mode 0とMode 3の主な違いは、SPIマスターがスタンバイで、データがデバイスのSPIインターフェイスに転送されていないとき、CLKピンの通常状態に関係があります。通常Mode 0ではCLKはLowで、Mode 3ではHighです。どちらの場合でも、SIピンのデータ入力は立ち上がりエッジ時だけサンプルされます。SOピンのデータはCLKの立ち下がりエッジのみで出力されます。

ステータス・レジスタ

SPIインターフェイスは、タグメモリ・フラッシュ・プログラミングをサポートするために必要な1ビットのステータス・レジスタにアクセスすることができます。

プログラミング完了ステータス・レジスタ；これはポーリング用の 1 ビットのステータス・レジスタです。プログラミングか消去操作が完了すると、ステータスビットは 1 にセットされます。さもなければ 0 にクリアされ、まだプログラミングや消去時間が必要なことを示します。

コマンド

表10-21 コマンド

コマンド名	バイト1 (Opcode)	バイト2	バイト3	バイト4	バイト5	バイト6	n-バイト
READ_ID	0x98	0x00	0x00	0x00	(D0-D7)	(D8-D15)	(D24-D31)
WRITE_EN	0xAC	0x00	0x00	0x00			
WRITE_DIS	0x78	0x00	0x00	0x00			
ERASE_TAG	0x0E	0x00	0x00	0x00			
PROGRAM_TAG	0x8E	0x00	0x00	0x00	D7-D0	次バイト	最終バイト
READ_TAG	0x4E	0x00	0x00	0x00	(D7-D0)	(次バイト)	(続く)
STATUS	0x4A	0x00	0x00	0x00	(b1xxxxxxx or b0xxxxxxx)		

注；1. データバイトは最上位ビット(MSB)が最初にシフトされます。括弧内のデータがあるバイトフィールドは、SOピン

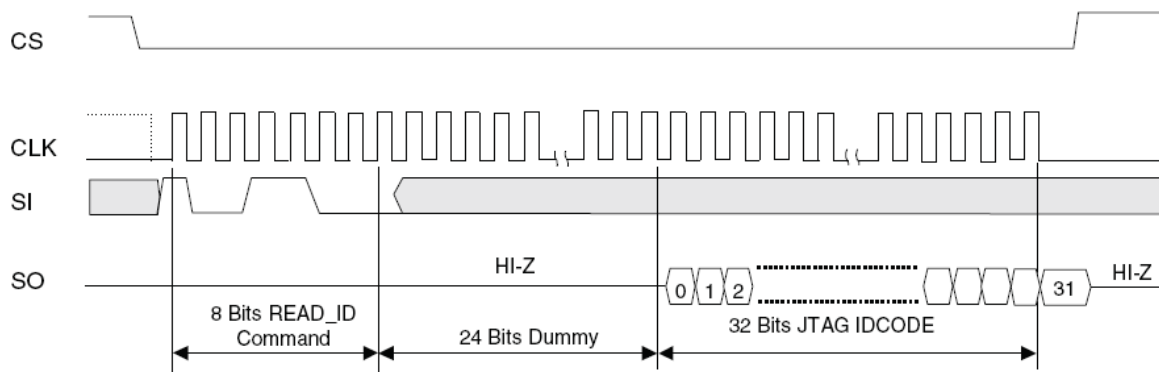
から読まれることを示します。

2. バイト2～4は、デバイスがコマンドを実行するために付加的なタイミングを提供するダミークロックです。これらのダミークロックの期間SIピンに入力されるデータは、図示されるように0x00である必要はなく、どんな値でも構いません。
3. READ_IDコマンドはデバイスの32ビットJTAG IDCODEを読み出します。その結果、SOピンにシフトアウトされる先頭ビットはJTAG IDCODEのビット0で、最終ビットはビット31です。
4. PROGRAM_TAGコマンドはページプログラミングのみをサポートします。タグメモリにシフト入力されるプログラミング・データは、タグメモリのページサイズと同一サイズでなければなりません。シフトサイズが少なかったり、多かったりした場合、タグメモリにプログラムされるデータのエラーの原因になります。n-Byteカラムに示す“最終バイト”は、チップセレクトがHighになりプログラミング動作が起動される前に、データ最終バイトがデバイスにシフト入力されなければならないことを示します。
5. 単一ビットのステータス・レジスタから読まれるSTATUSコマンド。レジスタから読まれると、先頭ビットだけが有効で、他のビットはダミーのため無視します。

READ_ID (98h)

READ_IDコマンドはデバイスのSOピンからIEEE1149.1JTAG IDCODEを捕捉します。このコマンドは、通常SPIバスとコミュニケーションが確立されたかどうかを確かめるのに用いられます。8ビットのREAD_IDコマンドが受け取られた後、デバイスはSIピンのデータを無視します。SOピンは、IDCODEの最初のビットを出力するためにクロック31の立ち下がりエッジでイネーブルされます。IDCODEの32ビットがシフトアウトされた後に、付加的なクロックによりSOからダミーデータをシフトアウトします。

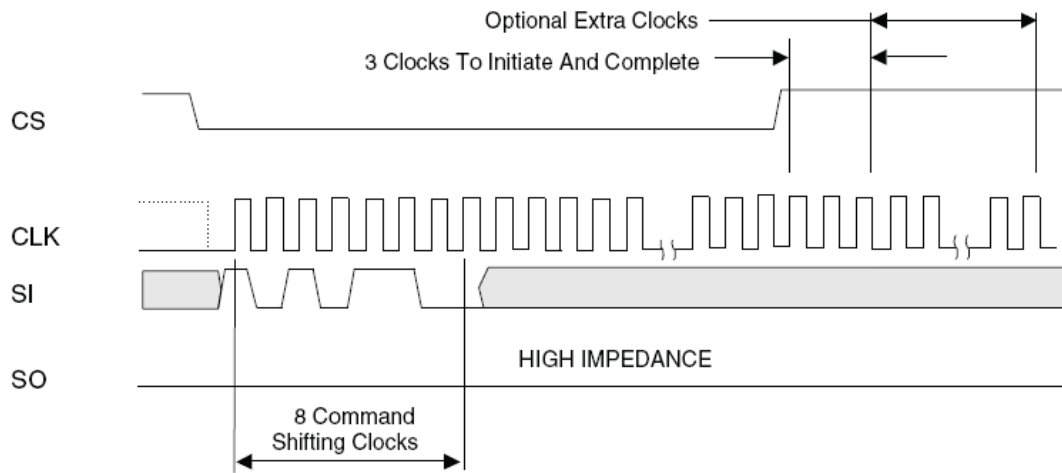
図10-50 READ_ID 波形



WRITE_EN (ACh)

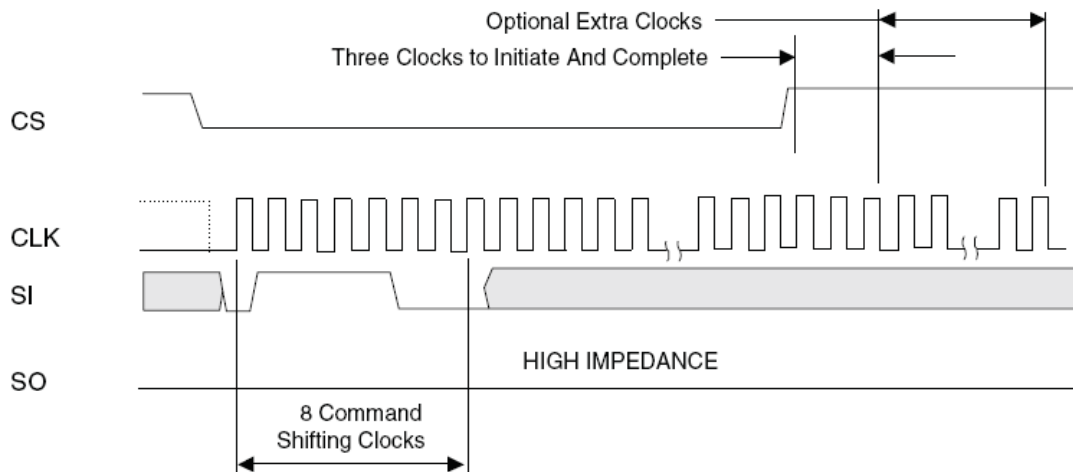
WRITE_ENコマンドはプログラミングのためにタグメモリをイネーブルします。WRITE_ENコマンドが最初にデバイスに入力されていない場合、PROGRAM_TAG、ERASE_TAG、およびSTATUSコマンドは有効になりません。これは、誤った消去やプログラミングからタグメモリを保護するためです。

チップセレクト・ピンが24番目のダミークロック後にLowからHighにドライブされるとき、コマンドは実行されます。チップセレクト・ピンをHighにする前に提示されると、どんな余分なダミークロックも無視されます。チップセレクト・ピンがLowからHighにされた後に、最低3クロックがコマンドの実行を完了するために必要です。このコマンドの効果はWRITE_DISコマンドで強制終了されます。

図10-51 WRITE_EN波形**WRITE_DIS (78h)**

WRITE_DISコマンドはタグメモリのプログラミングを無効にしますが、READ_TAGとREAD_IDコマンドは無効にしません。

チップセレクト・ピンが24番目のダミークロック後にLowからHighにドライブされるとき、コマンドは実行されます。チップセレクト・ピンをHighにドライブする前に供給される余分なダミークロックは無視されます。チップセレクト・ピンがLowからHighにドライブされた後に、最低3クロックがコマンドの実行を完了するために必要です。

図10-52 WRITE_DIS波形**ERASE_TAG (0Eh)**

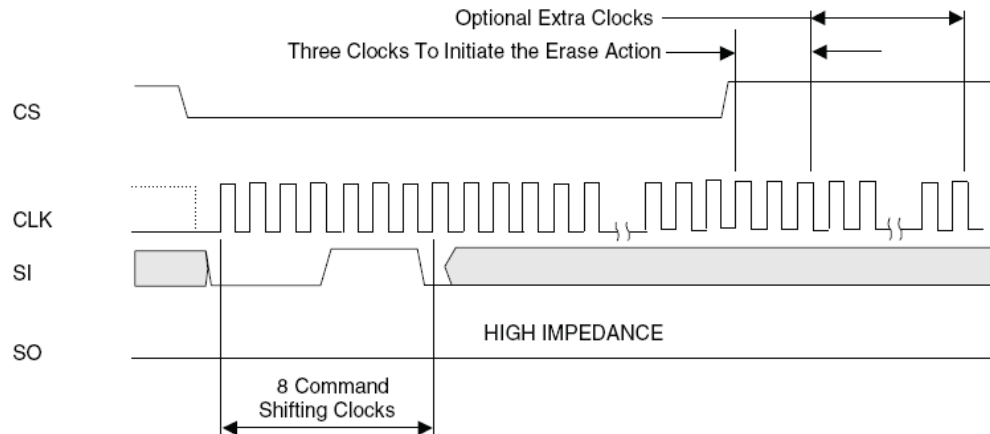
ERASE_TAGコマンドは、コマンドWRITE_ENがデバイスにシフトされた後にイネーブルされて、実行されます。ERASE_TAGコマンドは、すべてのタグメモリ・フラッシュセルを消去します。

24番目のダミークロックの後チップセレクト・ピンがLowからHighにドライブされると、コマンドは実行されます。チップセレクト・ピンをHighにする前に供給される余分なダミークロックは無視されます。チップセレクト・ピンがLowからHighにドライブされた後に、最低3クロックが消去動作の起動に必要です。3クロック後の付加的なクロックは任意です。消去動作がいったん開始されると、それは完了するまで実行され

ます。消去動作を強制終了するための機構は全くありません。

このコマンドは、消去動作が始まるとSTATUSビットを 0 にクリアします。消去が完了し成功すると、プログラミング・エンジンはステータスビットを 1 にセットします。

図10-53 ERASE_TAG 波形



PROGRAM_TAG (8Eh)

PROGRAM_TAGは、コマンドWRITE_ENがデバイスにシフトされた後にイネーブルされて、実行されます。PROGRAM_TAGコマンドはタグメモリ・ページ全部を一度にプログラムします。

コマンドがSIピンからデバイスにシフト入力されて、24ダミークロックが続いた後に、タグメモリのコラムデコーダは、プログラミング・データがシリアルにシフトされるデータバッファとして役立ちます。シフト方向は示されているように左から右です。SOピンに最も近くシフト出力される最初のビットは、シフトレジスタの最も右側にあります。データバッファはFIFOシリアル・データシフトのように機能します。ビット0が最初に読みだされることを確実にするために、データ・シフトレジスタの最も右側にデータビット0をシフトしなければなりません。これを達成するためには、データバッファは完全に満たさなければなりません。その結果、データバッファに過剰に入力するとデータバッファのオーバーフローを引き起こし、データの喪失となります。

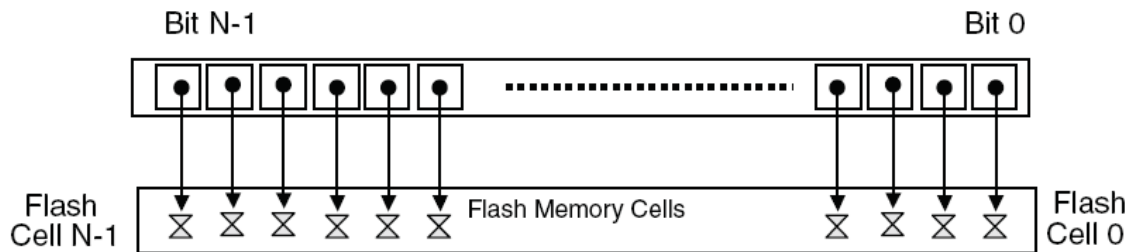
このコマンド中、SOピンはHIGHZ (ハイインピーダンス) 状態のままです。

図10-54 シフトビットの順序



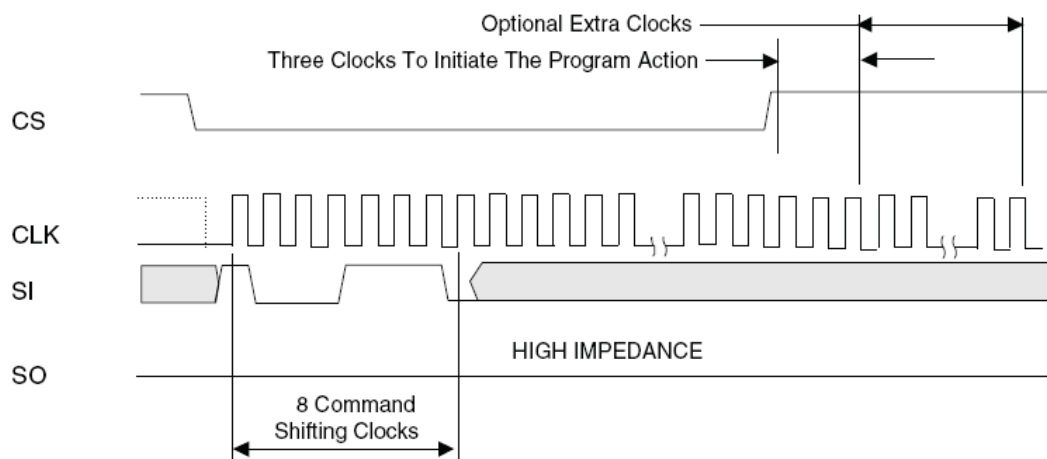
データバッファが最大1ページのデータで満たされて、チップセレクト・ピンをHighドライブすると、データシフトは終了します。チップセレクト・ピンがLowからHighにドライブされた後に、最低3クロックがプログラミング動作を起動するために必要です。プログラミング動作では、データバッファ内容がデータバッファからタグメモリ・フラッシュセルの中に並列にコピーされます。プログラミング動作が始まるとき、ステータスビットは 0 にクリアされます。プログラミング動作が完了し成功すると、ステータスビットは 1 にセットされます。

図10-55 データバッファのフラッシュセルへのマッピング



プログラミング動作が完了すると、STATUSビットは 1 にセットされます。プログラミング動作が完了すると、全く同じイメージがタグメモリ・ブロックのフラッシュセル内に書かれます。

図10-56 PROGRAM_TAG 波形



READ_TAG (4Eh)

READ_TAGコマンドは、フラッシュセル内にプログラムされたデータをデータバッファに転送するために、フラッシュプログラミング・エンジンをイネーブルします。転送動作は8ビットのオペコードの後の3番目のダミークロックで始まります。21個のダミークロックから得られる遅延時間は、フラッシュセル・データをデータバッファに転送するために必要な時間です。転送動作が一度開始されると、動作し続けるためのクロックは不要です。クロックカウントは、SOピンをイネーブルするために必要であるだけです。フラッシュ回路がまだイネーブルされていないと、デバイスが転送を起動する前にフラッシュ回路をイネーブルするために付加的な遅延時間を必要とします。この付加的な遅延はダミークロック3番目の後と24番目の前に与えなければなりません。

WRITE_ENコマンドがREAD_TAGコマンドに先行されているなら、それは高速読み出しです。デバイスは、フラッシュ回路をイネーブルするための付加的な遅延は必要としません。

転送が起動された後で、かつSOピンをイネーブルする前の20個のダミークロックは、遅延クロックと考えられます。

$$\text{遅延時間} = 20 \times 1/(\text{周波数})$$

フラッシュ回路をイネーブルするための付加的な遅延時間を含む転送遅延時間は最小5usecです。連続クロックの必要がある場合、クロック周波数を2.5MHzに設定することができます。

データバッファに捕捉された全データがシフトアウトされる時、付加的なクロックはダミーデータをシフトアウトします。READ_TAGコマンドがデバイスにシフト入力されると、SIピンはデータバッファ入力に接続されていません。データバッファのデータはSOの方向にシフトアウトされますが、ダミーデータはデータバッファにシフト入力されます。シフトし過ぎが起こるとき、結果として不定値のダミーデータがシフトアウトされます。

図10-57 読み出し順序

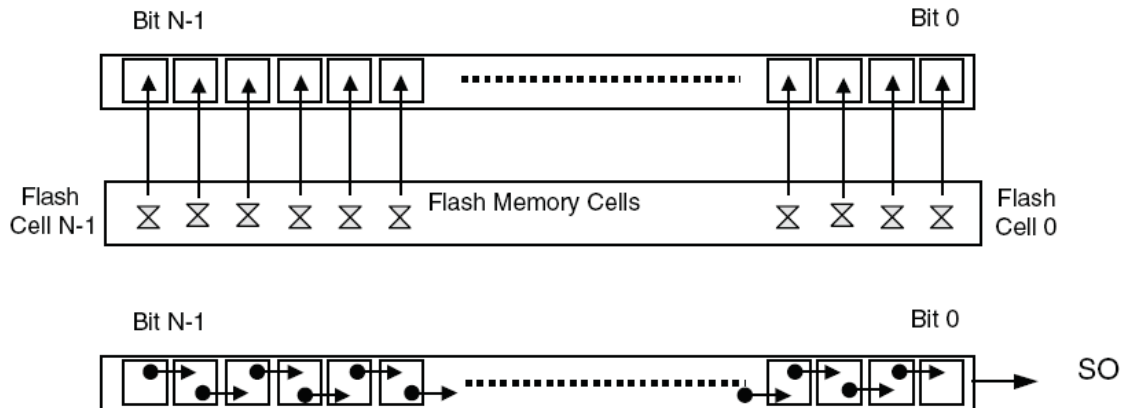
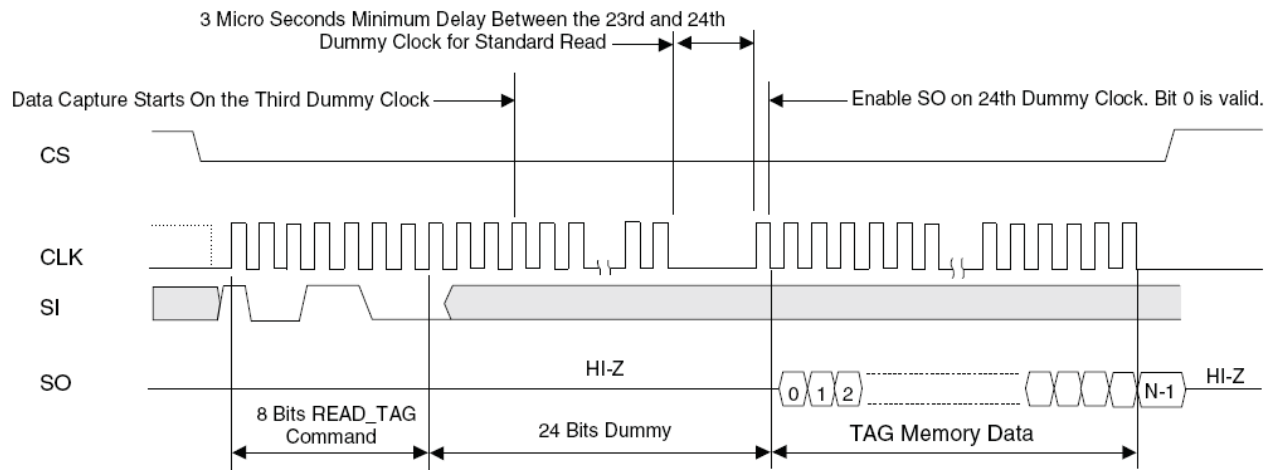


図10-58 READ_TAG 波形



STATUS (4Ah)

STATUSコマンドは単一ビットのステータス・レジスタの読み出しを可能にします。最初にWRITE_ENコマンドがデバイスにシフト入力された後であれば、いつでもこのコマンドをロードすることができます。このコマンドはプログラミングや消去動作を強制終了しません。これはプログラミングや消去動作の進行状態を報告するために用いられます。

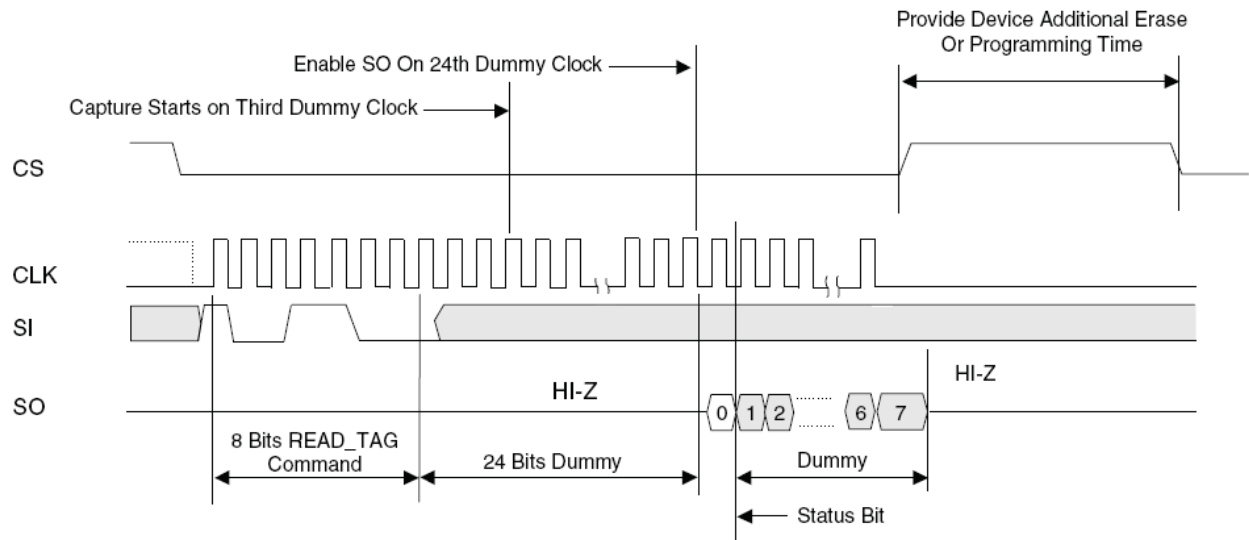
ステータス・レジスタの実サイズは単一ビットです。付加的なデータシフト・クロックが供給されている場合、ダミーデータはSOピンにシフトアウトされます。ステータスビットを得るためにコマンドが再びデバイスにシフト入力し、次に読み出すことができます。

コマンドをシフトする合間の期間、チップセレクト・ピンをHighにして、かつCLKピンをLowに保持するこ

とで、プログラミングや消去のための付加的時間を提供します。チップセレクト・ピンをHighに保っている間、クロックを供給することは任意です。

プログラミングや消去の最大時間が経過した後、まだステータスビットが 1 にセットされていない場合、消去やプログラミングは失敗です。

図10-59 STATUS波形



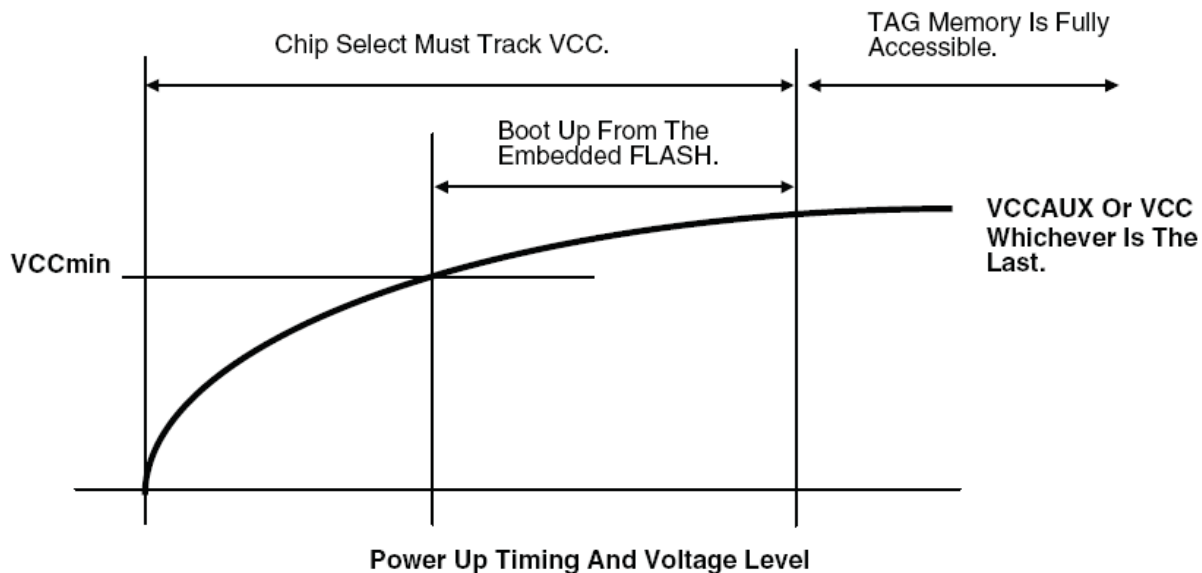
仕様とタイミング・ダイヤグラム

パワーアップ

内部の組み込みフラッシュか外付けSPIフラッシュPROMからの起動が完了すると、タグメモリが利用できます。組み込みフラッシュがブランクだと起動しません。タグメモリにアクセスする前に、組み込みフラッシュがプログラムしてある場合と同じ遅延時間待つことを推奨します。外付けSPIフラッシュから起動する場合、タグメモリにアクセスする前に、より長い遅延時間待つか、またはDONEピンがHighかどうかをまずチェックする必要があります。

SPIインターフェイスをリセットするためには、チップセレクト・ピンがLowからHighへ遷移することが必要です。パワーアップの際にCLKピンがVCCに追随する要件によって、LowからHighへ遷移は保証されます。SPIインターフェイスをリセットするもう一つの方法は、最初のコマンドをデバイスにシフト入力する前に、チップセレクト・ピンをHigh、Low、そしてHighにドライブすることです。

図10-60 デバイスのパワーアップ波形



タグメモリの使用

タグメモリは以下の例外を除き、スレーブSPIインターフェイスの殆どの時間で利用できます。

- ・SRAMヒューズがJTAGポートや、スレーブSPIインターフェイス、またはリフレッシュでアクセスされているとき
- ・JTAG BSCANテストが行われている間
- ・パーシスタンス(persistent)ヒューズがプログラムされていて(OFFに設定)、スレーブSPIインターフェイスが無効の場合

ACタイミング

- ・最大25MHzのCLK
- ・最小5usecのリードコマンド遅延
- ・VCCminから最初のコマンドのシフトするまで最小2msecの遅延

プログラミング・タイミング

- ・最大1秒の消去時間
- ・最大5msecのプログラミング時間

JTAGインターフェイスからのプログラミング

タグメモリのみをプログラムする .VMEファイルをispVM Systemソフトウェアで生成することができます。これらの .VMEファイルは標準のispVMEフローに従って扱われます。

メモリの初期化

EBRベースのROMおよびRAMモードで、メモリアレイ内の各ビットのパワーオン後の状態を指定することが可能です。メモリアレイにおける各ビットは 0 か 1 かどちらかの値を持つことができます。

初期化ファイル形式

初期化ファイルはASCIIファイルで、ユーザはどのテキストエディタでも使用できます。IPexpressは3タイプのメモリフォーマットをサポートします。

- ・バイナリファイル

- ・ 16進(Hex)ファイル
- ・ アドレス付き16進(Addressed Hex)

メモリ初期化ファイルのファイル名は *.mem (<file_name>.mem)です。各列は、特定のメモリ領域に格納される値について表現します。キャラクタ数(または、コラム数)は各アドレス(または、メモリモジュールの幅)のビット数を表わします。

初期化ファイルは、主にROMを構成するために用いられます。RAMモードのEBRもまた、メモリ内容をブロードするためにオプションとしてこの初期化ファイルを用いることができます。

バイナリファイル

ファイルは、本質的には0と1のテキストファイルです。列はワード数を、コラムがメモリ幅を示します。

```
メモリサイズ 20x32
00100000001000000000100000001000000
00000000100000000010000000100000001
0000000100000000100000001000000010
0000000110000000110000001100000011
000001000000001000000010000000100
000001010000001010000010100000101
000001100000001100000011000000110
000001110000001110000011100000111
00001000010010000000100001001000
00001001010010010010000100101001001
0000101001001010100000101001001010
000010110100101100000101101001011
00001100000011000000110000001100
00001101001011010000110100101101
00001110001111100000111000111110
00001111001111110000111100111111
000100000000100000001000000010000
00010001000100010001000100010001
000100100001001000001001000010010
000100110001001100001001100010011
```

16進ファイル

16進ファイルは本質的には、類似の列・コラムでアレンジされた16進キャラクタのテキストファイルです。ファイル内の列の数はアドレスの位置の数と同じで、各列がメモリ位置の内容を示しています。

```
メモリサイズ 8x16
A001
0B03
1004
CE06
0007
040A
0017
02A4
```

アドレス付き16進

アドレス付き16進はアドレスとデータによる行から成ります。各行はアドレスから始まり、コロン、そしていろいろなデータが続きます。memfileの形式は “アドレス: データ データ データ データ...” で、アドレスとデータは16進数です。

```
-A0 : 03 F3 3E 4F
-B2 : 3B 9F
```

最初の行はアドレスA0に03を置き、アドレスA1にF3、アドレスA2に3E、およびアドレスA3に4Fを置きます。第二行はアドレスB2に3Bを、アドレスB3に9Fを置きます。

アドレスとデータの値に制限は全くありません。値の範囲は `addr_width` と `data_width` 値に基づいて自動的にチェックされます。アドレスかデータ値に誤りがあれば、エラーメッセージがプリント（表示）されます。ユーザはすべてのアドレスの位置でデータを指定する必要はありません。あるアドレスで、データが指定されない場合、その位置のデータは 0 に初期化されます。IPexpressは、論理合成フローとシミュレーション・フローで共にメモリの初期化を可能にします。

FlashBak™ 機能

LatticeXP2ファミリはFlashBak機能を提案します。これはユーザコマンドによりEBR内のデータをフラッシュメモリに格納（保存）する方法です。FlashBakモジュール（STFAプリミティブ）は単一コマンドで二つの操作というプロセスを持ちます（図10-61参照）。FlashBakが起動されると、フラッシュを消去するための `erase-UFM-Flash` 信号がイネーブルされて、`transfer-to-operation` 動作が続きます。転送が完了すると、フラッシュ・コントローラはユーザロジックに対して `transfer-done` 信号を返します。FlashBak動作の期間、EBRはアクセスできません。IspLEVER GUIでは、通常のEBR RAM構成と、シャドー（UFM）EBR RAM構成の間に違いはありません。STFA（FlashBak）プリミティブがデザインにあるかどうかでEBR RAMの構成が決定されます。FlashBakはソフトエラー検出（SED）マクロが “Always” モードで動作している場合、FlashBakは用いることができません。アドレッシング機能はなく、EBR全てをフラッシュにダンプするだけなので、単一のSTFAモジュールが必要です。複数使用は不要、或いは禁止されています。

図10-61 FlashBak プリミティブ

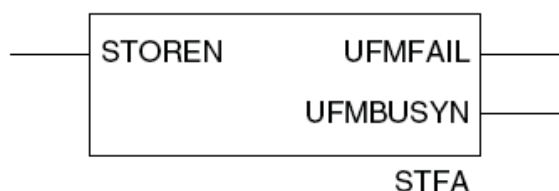


図10-62 FlashBak 波形

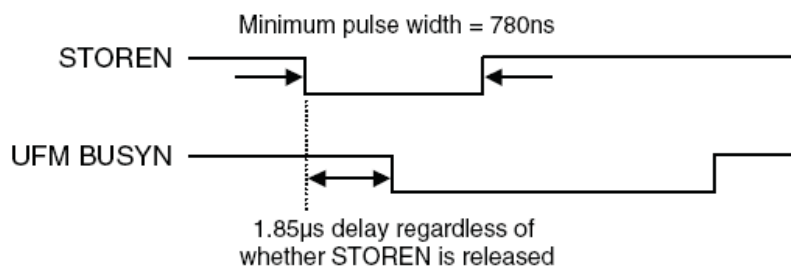


表10-22 STFAポート定義

ポート名	対応するハードウェア・ポート名	I/O	記 述
STOREN	storecmn	I	EBR内容のフラッシュへの保存を起動
UFMFAIL	ufm_fail	O	フラッシュに保存操作は失敗（不成功）
UFMBUSYN	fl_busyn	O	フラッシュがビジー状態かどうかを示す

テクニカル・サポート支援

ホットライン: 1-800-LATTICE (North America)

+1-408-826-6002 (Outside North America)

e-mail: techsupport@latticesemi.com

インターネット: www.latticesemi.com

変更履歴（日本語版）

Rev.#	日付	変更箇所
1.7J	Jan. 2009	日本語版新規発行

付録A.アトリビュート定義

DATA_WIDTH

データ幅はRAMとFIFO要素に関連しています。DATA_WIDTHアトリビュートは各ワードのビット数を定義します。それは各メモリモジュールで、RAMサイズ表で定義されるように値を取ります。

REGMODE

REGMODEアトリビュートが、メモリ内のパイプライン化をイネーブルするのに用いられます。このアトリビュートはRAMとFIFO要素に関連します。REGMODEアトリビュートはNOREGかOUTREGモードパラメータを取り、出力パイプライン・レジスタをディセーブル、またはイネーブルします。

RESETMODE

RESETMODEアトリビュートで、ユーザはメモリ内のリセット方法を選択します。このアトリビュートはブロックRAM要素に関連しています。RESETMODEは2つのパラメータを取り、SYNCとASYNCSYNCは、メモリリセットがクロックと同期することを意味します。ASYNCSYNCは、メモリリセットがクロックに非同期であることを意味します。

CSDECODE

CSDECODEアトリビュートは、ブロックRAM要素に関連します。EBRプリミティブで利用できるポートのCSは、メモリが複数のEBRブロックを連結される必要があるときに役に立ちます。複数のEBRブロックが連結されるとき、CS信号はアドレスのMSBとなります。CSは3ビットバスなので、容易に8つのメモリを連結することができます。CSDECODEは以下のパラメータを取ります; "000", "001", "010", "011", "100", "101", "110", そして "111" です。CSDECODE値はCS[2:0]のデコード値を決定します。擬似デュアルポートRAM用に、CSDECODE_Wは書き込み時のチップセレクト・デコード、CSDECODE_Rが読み出し時のチップセレクト・デコードです。CSDECODE_AとCSDECODE_Bは真のデュアルポートRAM要素に用いられて、AとBポートにそれぞれ関連します。

WRITEMODE

WRITEMODEアトリビュートはブロックRAM要素に関連しています。それはNORMAL、およびWRITETHROUGHモードのパラメータを取ります。

NORMALモードで、書き込み動作の間は出力データを変えないか、または更新しません。本モードは全てのデータ幅でサポートされます。

WRITETHROUGHモードで、ライトサイクルの間、入力データで出力データを更新します。本モードは全てのデータ幅でサポートされます。

WRITEMODE_AとWRITEMODE_BはデュアルポートRAM要素に用いられて、真のデュアルポートRAMの場合にはAとBポートに関連します。

全ての（真のデュアルポート・モジュールの）モードで、一方のポートからのリードアクセスと、他方のポートからのライトアクセスを同じアドレスに対して行うことは勧められません。この場合読み出しデータは不定になり得ます。また、同じアドレスに同時ライトアクセスも勧められません（一方からHighを書き、他方からLowを書いた場合など、そのアドレスにストアされるデータは不定になります）。

この状況が起こりうる場合は、これを検出するために以下のような制御ロジックを実装することを推奨します。

- 1, リードデータが無効な可能性があることを示すステータス信号を設ける、或いは
- 2, 両ポートからの（同一アドレスへの）同時アクセスを防ぐ制御ロジックを実装する

GSR

GSRアトリビュートは、RAM要素のためにグローバルなセット/リセットをイネーブルするか、またはディセーブルするために用いられます。