


Gowin プログラマブル汎用 IO(GPIO) ユーザーガイド

UG289-2.1.3J, 2023-04-20

著作権について(2023)

著作権に関する全ての権利は、**Guangdong Gowin Semiconductor Corporation** に留保されています。

 **Gowin**、**Arora**、**LittleBee**、及び**GOWINSEMI**は、当社により、中国、米国特許商標庁、及びその他の国において登録されています。商標又はサービスマークとして特定されたその他全ての文字やロゴは、それぞれの権利者に帰属しています。何れの団体及び個人も、当社の書面による許可を得ず、本文書の内容の一部もしくは全部を、いかなる視聴覚的、電子的、機械的、複写、録音等の手段によりもしくは形式により、伝搬又は複製をしてはなりません。

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。
2016/08/02	1.07J	GW2A シリーズ FPGA 製品のサポートを追加。
2016/10/27	1.08J	GW2AR シリーズ FPGA 製品のサポートを追加。
2017/09/01	1.09J	GW1N-6/9 の特性と GW1NR の情報を更新。
2017/10/12	1.10J	IDES16/OSER16 の関連備考情報を追加。
2017/12/12	1.2J	<ul style="list-style-type: none"> ● IDDR/ODDR RESET 信号を削除。 ● LVDS の説明を更新。 ● memory 付きの入力/出力の説明を追加。
2018/04/08	1.3J	第 7 章の図表を更新。
2020/05/14	1.4J	<ul style="list-style-type: none"> ● 3.6GPIO プリミティブを更新。 ● GW1N-6/GW1NR-6 デバイスの情報を削除。
2020/08/27	1.5J	<ul style="list-style-type: none"> ● マニュアルの構造を最適化。 ● 4 入出力ロジックと 5 IP の呼び出しを追加。
2021/01/07	1.6J	IODELAYB モジュールの内容を更新。
2021/02/02	1.7J	<ul style="list-style-type: none"> ● MIPI_IBUF_HS,MIPI_IBUF_LP の説明を追加。 ● GW2AN-55C、GW1NR-2 のサポートを追加。
2021/03/25	1.8J	<ul style="list-style-type: none"> ● GW1NZ-2 デバイスの情報を削除。 ● MIPI_OBUF、MIPI_OBUF_A をサポートするデバイスを更新。
2021/06/21	1.9J	<ul style="list-style-type: none"> ● デバイス(GW1N-2B、GW1N-1P5、GW1N-1P5B、GW1NR-2B、GW2AN-18X、GW2AN-9X)のサポートを追加。 ● IP 呼び出しの図面を更新。
2021/10/21	1.9.1J	GPIO 規格の説明を更新。
2021/11/23	1.9.2J	入力ロジックの図面を更新。
2022/01/24	2.0J	<ul style="list-style-type: none"> ● 入出力バッファの説明を更新 ● コード例のフォーマットを微調整。
2022/06/02	2.01J	終端抵抗の説明を更新。
2022/07/22	2.0.2J	OSER4 の説明を更新。
2022/08/11	2.0.3J	デバイスのバージョン情報を更新。
2022/11/11	2.1J	<ul style="list-style-type: none"> ● GW1NS-2、GW1NS-2C、GW1NSE-2C、GW1NSR-2、および GW1NSR-2C デバイスを削除。 ● セクション 3.6.14 ELVDS_IBUF_MIPI を追加。
2023/01/05	2.1.1J	<ul style="list-style-type: none"> ● IP 呼び出しの図面を更新、"Device Version"オプションを追加。 ● 差動バッファの構成の情報を更新。
2023/02/28	2.1.2J	Slew Rate の情報を削除。
2023/04/20	2.1.3J	<ul style="list-style-type: none"> ● 「3.3 電源供給の要件」の説明を更新

日付	バージョン	説明
		● 表 3-10 MIPI_IBUF 対応デバイスを更新。

目次

目次	i
図一覧	iv
表一覧	vi
1 本マニュアルについて	1
1.1 マニュアル内容	1
1.2 関連ドキュメント	1
1.3 用語、略語	2
1.4 テクニカル・サポートとフィードバック	2
2 GPIO の概要	3
3 入出力バッファ	5
3.1 GPIO 規格	5
3.2 GPIO バンク	6
3.3 電源供給の要件	6
3.3.1 LVCMOS バッファの構成	6
3.3.2 差動バッファの構成	7
3.4 エミュレート差動回路終端方式	7
3.4.1 エミュレート LVDS	7
3.4.2 エミュレート LVPECL	8
3.4.3 エミュレート RSDS	8
3.4.4 エミュレート BLVDS	8
3.5 GPIO の構成	9
3.5.1 位置	9
3.5.2 レベル規格	9
3.5.3 ドライブ強度	9
3.5.4 プルアップ/ダウンモード	9
3.5.5 リファレンス電圧	9
3.5.6 ヒステリシス	10
3.5.7 オープンドレイン	10

3.5.8 シングルエンド終端抵抗	10
3.5.9 差動終端抵抗	10
3.6 GPIO プリミティブ	10
3.6.1 IBUF	10
3.6.2 OBUF	11
3.6.3 TBUF	12
3.6.4 IOBUF	13
3.6.5 LVDS Input Buffer	15
3.6.6 LVDS Output Buffer	17
3.6.7 LVDS Tristate Buffer	19
3.6.8 LVDS Inout Buffer	21
3.6.9 MIPI_IBUF	23
3.6.10 MIPI_OBUF	25
3.6.11 MIPI_OBUF_A	27
3.6.12 I3C_IOBUF	29
3.6.13 MIPI_IBUF_HS/MIPI_IBUF_LP	30
3.6.14 ELVDS_IBUF_MIPI	33
4 入出力ロジック	35
4.1 SDR モード	36
4.2 DDR モードの入出力ロジック	36
4.2.1 IDDR	36
4.2.2 IDDR_C	39
4.2.3 IDES4	41
4.2.4 IDES8	44
4.2.5 IDES10	47
4.2.6 IVIDEO	50
4.2.7 IDES16	53
4.2.8 IDDR_MEM	57
4.2.9 IDES4_MEM	60
4.2.10 IDES8_MEM	63
4.3 DDR モードの出力ロジック	67
4.3.1 ODDR	67
4.3.2 ODDRC	70
4.3.3 OSER4	73
4.3.4 OSER8	77
4.3.5 OSER10	81
4.3.6 OVIDEO	84
4.3.7 OSER16	87
4.3.8 ODDR_MEM	90

4.3.9 OSER4_MEM.....	94
4.3.10 OSER8_MEM.....	99
4.4 遅延モジュール	104
4.4.1 IODELAY	104
4.4.2 IODELAYC	106
4.4.3 IODELAYB.....	109
4.5 サンプリングモジュール	113
5 IP の呼び出し.....	116
5.1 IP の構成	116
5.2 生成されるファイル	118

図一覧

図 2-1 入出力ブロックの構成説明図	4
図 3-1 LVDS25E の外部終端	8
図 3-2 LVPECL の外部終端	8
図 3-3 RSDS の外部終端	8
図 3-4 BLVDS の外部終端	9
図 3-5 IBUF のポート図	10
図 3-6 OBUF のポート図	11
図 3-7 TBUF のポート図	12
図 3-8 IOBUF のポート図	14
図 3-9 TLVDS_IBUF/ELVDS_IBUF のポート図	15
図 3-10 TLVDS_OBUF/ELVDS_OBUF のポート図	17
図 3-11 TLVDS_TBUF/ELVDS_TBUF のポート図	19
図 3-12 TLVDS_IOBUF/ELVDS_IOBUF のポート図	21
図 3-13 MIPI_IBUF のポート図	23
図 3-14 MIPI_OBUF のポート図	26
図 3-15 MIPI_OBUF_A のポート図	27
図 3-16 I3C_IOBUF のポート図	29
図 3-17 MIPI_IBUF_HS/MIPI_IBUF_LP のポート図	31
図 3-18 ELVDS_IBUF_MIPI のポート図	33
図 4-1 入出力ロジックの説明図 –出力	35
図 4-2 入出力ロジックの説明図 –入力	36
図 4-3 IDDR のブロック図	37
図 4-4 IDDR のタイミング図	37
図 4-5 IDDR のポート図	37
図 4-6 IDDRRC のポート図	39
図 4-7 CALIB のタイミングの例	41
図 4-8 IDES4 のポート図	42
図 4-9 IDES8 のポート図	44
図 4-10 IDES10 のポート図	47

図 4-11 IVIDEO のポート図	50
図 4-12 IDER16 のポート図	54
図 4-13 IDDR_MEM のポート図	58
図 4-14 IDER4_MEM のポート図	61
図 4-15 IDER8_MEM のポート図	64
図 4-16 ODDR のブロック図	68
図 4-17 ODDR のタイミング図	68
図 4-18 ODDR のポート図	68
図 4-19 ODDRC のブロック図	71
図 4-20 ODDRC のポート図	71
図 4-21 OSER4 のブロック図	74
図 4-22 OSER4 のポート図	74
図 4-23 OSER8 のブロック図	77
図 4-24 OSER8 のポート図	78
図 4-25 OSER10 のポート図	81
図 4-26 OVIDEO のポート図	84
図 4-27 OSER16 のポート図	87
図 4-28 ODDR_MEM のブロック図	91
図 4-29 ODDR_MEM のポート図	91
図 4-30 OSER4_MEM のブロック図	95
図 4-31 OSER4_MEM のポート図	95
図 4-32 OSER8_MEM のブロック図	99
図 4-33 OSER8_MEM のポート図	100
図 4-34 IODELAY のポート図	104
図 4-35 IODELAYC のポート図	106
図 4-36 IODELAYB の構造	110
図 4-37 IODELAYB のポート図	110
図 4-38 IEM のポート図	113
図 5-1 DDR IP の構成ウィンドウ	116

表一覧

表 1-1 用語、略語	2
表 3-1 IBUF のポートの説明	10
表 3-2 OBUF のポートの説明	11
表 3-3 TBUF のポートの説明	13
表 3-4 IOBUF のポートの説明	14
表 3-5 TLVDS_IBUF/ELVDS_IBUF のポートの説明	15
表 3-6 TLVDS_OBUF/ELVDS_OBUF のポートの説明	17
表 3-7 TLVDS_TBUF/ELVDS_TBUF のポートの説明	19
表 3-8 TLVDS_IOBUF 対応デバイス	21
表 3-9 TLVDS_IOBUF/ELVDS_IOBUF のポートの説明	21
表 3-10 MIPI_IBUF 対応デバイス	23
表 3-11 MIPI_IBUF のポートの説明	23
表 3-12 MIPI_OBUF 対応デバイス	25
表 3-13 MIPI_OBUF のポートの説明	26
表 3-14 MIPI_OBUF_A 対応デバイス(追加)	27
表 3-15 MIPI_OBUF_A のポートの説明	27
表 3-16 I3C_IOBUF 対応デバイス	29
表 3-17 I3C_IOBUF のポート図	29
表 3-18 MIPI_IBUF_HS/MIPI_IBUF_LP 対応デバイス	30
表 3-19 MIPI_IBUF_HS のポートの説明	31
表 3-20 MIPI_IBUF_LP のポートの説明	31
表 3-21 MIPI_IBUF のポートの説明	33
表 4-1 IDDR のポートの説明	37
表 4-2 IDDR のパラメータの説明	38
表 4-3 IDDRC のポートの説明	39
表 4-4 IDDRC のパラメータの説明	39
表 4-5 IDES4 のポートの説明	42
表 4-6 IDES4 のパラメータの説明	42
表 4-7 IDES8 のポートの説明	44

表 4-8 IDES8 のパラメータの説明.....	45
表 4-9 IDES10 のポートの説明.....	47
表 4-10 IDES10 のパラメータの説明	48
表 4-11 IVIDEO のポートの説明	51
表 4-12 IVIDEO のパラメータの説明.....	51
表 4-13 IDES16 対応デバイス	53
表 4-14 IDES16 のポートの説明.....	54
表 4-15 IDES16 のパラメータの説明	54
表 4-16 IDDR_MEM 対応デバイス.....	57
表 4-17 IDDR_MEM のポートの説明	58
表 4-18 IDDR_MEM のパラメータの説明.....	58
表 4-19 IDES4_MEM 対応デバイス	60
表 4-20 IDES4_MEM のポートの説明	61
表 4-21 IDES4_MEM のパラメータの説明	61
表 4-22 IDES8_MEM 対応デバイス	64
表 4-23 IDES8_MEM のポートの説明	65
表 4-24 IDES8_MEM のパラメータの説明	65
表 4-25 ODDR のポートの説明.....	69
表 4-26 ODDR のパラメータの説明	69
表 4-27 ODDRC のポートの説明.....	71
表 4-28 ODDRC のパラメータの説明.....	72
表 4-29 OSER4 のポートの説明	74
表 4-30 OSER4 のパラメータの説明.....	75
表 4-31 OSER8 のポートの説明	78
表 4-32 OSER8 のパラメータの説明.....	78
表 4-33 OSER10 のポートの説明.....	82
表 4-34 OSER10 のパラメータの説明.....	82
表 4-35 OVIDEO のポートの説明	84
表 4-36 OVIDEO のパラメータの説明	85
表 4-37 OSER16 対応デバイス.....	87
表 4-38 OSER16 のポートの説明	87
表 4-39 OSER16 のパラメータの説明.....	88
表 4-40 ODDR_MEM 対応デバイス	90
表 4-41 ODDR_MEM のポートの説明	91
表 4-42 ODDR_MEM のパラメータの説明	92
表 4-43 OSER4_MEM 対応デバイス	94
表 4-44 OSER4_MEM のポートの説明	95

表 4-45 OSER4_MEM のパラメータの説明	96
表 4-46 OSER8_MEM 対応デバイス	99
表 4-47 OSER8_MEM のポートの説明	100
表 4-48 OSER8_MEM のパラメータの説明	100
表 4-49 IODELAY のポートの説明	104
表 4-50 IODELAY のパラメータの説明	105
表 4-51 IODELAYC 対応デバイス	106
表 4-52 IODELAYC のポートの説明	107
表 4-53 IODELAYC のパラメータの説明	107
表 4-54 IODELAYB 対応デバイス	109
表 4-55 IODELAYB のポートの説明	110
表 4-56 IODELAYB のパラメータの説明	111
表 4-57 IEM のポートの説明	113
表 4-58 IEM のパラメータの説明	113

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、Gowin セミコンダクターFPGA 製品でサポートされる入出力バッファのレベル規格、バンクの配置、および入出力ロジックの機能について説明します。また、ユーザーが GPIO を使いこなせるよう GPIO の構造と Gowin ソフトウェアの使用法も説明されています。

1.2 関連ドキュメント

GOWIN セミコンダクターの公式 Web 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](#))
- GW1NS シリーズ FPGA 製品データシート([DS821](#))
- GW1NZ シリーズ FPGA 製品データシート([DS841](#))
- GW1NSR シリーズ FPGA 製品データシート([DS861](#))
- GW1NSE シリーズ安全 FPGA 製品データシート([DS871](#))
- GW1NSER シリーズ安全 FPGA 製品データシート([DS881](#))
- GW1NRF シリーズ Bluetooth FPGA 製品データシート([DS891](#))
- 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](#))
- GW2ANR シリーズ FPGA 製品データシート([DS961](#))
- GW2AN シリーズ FPGA 製品データシート([DS971](#))

1.3 用語、略語

表 1-1 に、本マニュアルで使用される用語、略語、及びその意味を示します。

表 1-1 用語、略語

用語、略語	正式名称	意味
Bus Keeper	Bus Keeper	バスキーパ(バスホールドラッチ)
CFU	Configurable Function Unit	コンフィギュラブル機能ユニット
CRU	Configurable Routing Unit	コンフィギュラブル配線ユニット
DDR	Double Data Rate	ダブルデータレート
DES	Deserializer	デシリアライザ
ELDO	Emulated LVDS Output	エミュレート LVDS 出力(電圧出力)
GPIO	Gowin Programmable Input/Output	Gowin プログラマブル汎用 IO
IO Buffer	Input/Output Buffer	入出力バッファ
IO Logic	Input/Output Logic	入出力ロジック
IOB	Input/Output Block	入出力ブロック
Open Drain	Open Drain	オープンドレイン
SDR	Single Data Rate	シングルデータレート
SER	Serializer	シリアライザ
TLDO	True LVDS Output	True LVDS 出力(電流出力)

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

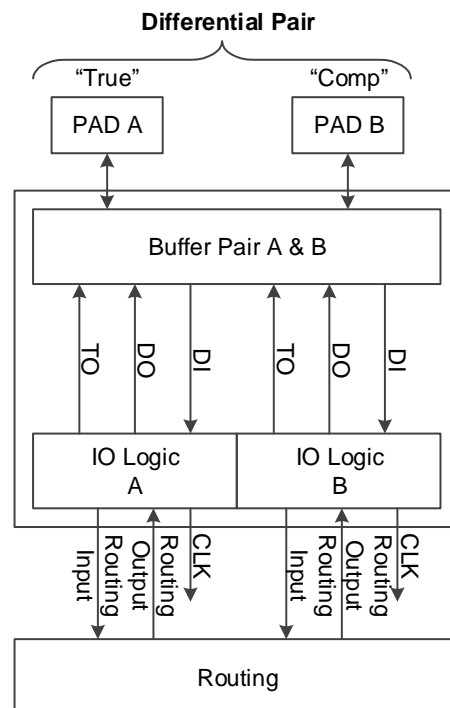
2 GPIO の概要

GOWIN セミコンダクターFPGA 製品の GPIO には、シングルエンドレベル規格から差動レベル規格まで、さまざまな外部バス、メモリデバイス、ビデオアプリケーションなどとの接続を容易にするための業界のさまざまなレベル規格をサポートする柔軟性があります。

GOWIN セミコンダクターFPGA 製品の GPIO の基本要素は、入出力バッファ(IO Buffer)、入出力ロジック(IO Logic)、および対応するコンフィギュラブル配線ユニットなどで構成される出力ブロック(IOB)です。そのうちコンフィギュラブル配線ユニットは、コンフィギュラブル機能ユニット(CFU)内のコンフィギュラブル配線ユニット(CRU)と同様です。

図 2-1 示すように、各入出力ブロックには、A および B とマークされる 2 つの入出力ピンがあります。それらは 1 つの差動信号ペアを構成するか、シングルエンド信号として個別に使用することができます。入出力バッファは、主にさまざまなシングルエンドレベル規格および差動レベル規格のサポートに使用されます。入出力ロジックは、シリアル-パラレル変換、パラレル-シリアル変換、遅延制御、およびバイトアライメントなどの機能を統合し、主に高速データ伝送に使用されます。コンフィギュラブル配線ユニットは、入出力ブロックと他のオンチップリソースとの間の相互接続に使用されます。

図 2-1 入出力ブロックの構成説明図



GOWIN セミコンダクターFPGA シリーズ製品の入出力ブロックの特徴:

- Bank 毎に供給される V_{CCIO}
- LVCMOS、PCI、LVTTL、LVDS、SSTL、及び HSTL 等複数の規格をサポート
- 一部のデバイス^[1]が MIPI 規格および MIPI I3C OpenDrain/PushPull 変換をサポート
- 入力信号のヒステリシスオプションを提供
- 出力信号のドライブ強度オプションを提供
- 各ピンに独立したバスホールド、プルアップ/プルダウン抵抗、及びオープンドレイン出力オプションを提供
- ホットプラグをサポート
- 入出力ロジックは、シングルデータレート(SDR)、ダブルデータレート(DDR)など、多くのモードをサポート

注記:

[1] MIPI、I3C をサポートするデバイスについては、[3.6.9 MIPI_IBUF](#)、[3.6.10 MIPI_OBUF](#)、および [3.6.12 I3C_IOBUF](#) を参照して下さい。

3 入出力バッファ

3.1 GPIO 規格

GOWIN セミコンダクターFPGA 製品は、シングルエンド規格と差動規格をサポートしています。シングルエンド規格では、内部のピン電圧をリファレンス電圧として使用するか、任意のピンを外部リファレンス電圧入力ピンとして使用することができます。Gowin FPGA のすべてのバンクは差動入力をサポートしています。エミュレート LVDS 差動出力は外部終端抵抗および差動 LVCMOS バッファ出力により実装されます。さらに、True LVDS 差動出力および差動入力終端をサポートする特定のバンクがあります。詳細については、[3.2 GPIO バンク](#)を参照してください。

GOWIN セミコンダクターFPGA 製品でサポートされている GPIO 規格については、対応するデータシートの“I/O 規格”セクションを参照してください。

- 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](#))
- GW1NS シリーズ FPGA 製品データシート([DS821](#))
- GW1NZ シリーズ FPGA 製品データシート([DS841](#))
- GW1NSR シリーズ FPGA 製品データシート([DS861](#))
- GW1NSE シリーズ安全 FPGA 製品データシート([DS871](#))
- GW1NSER シリーズ安全 FPGA 製品データシート([DS881](#))
- GW1NRF シリーズ Bluetooth FPGA 製品データシート([DS891](#))
- 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](#))
- GW2ANR シリーズ FPGA 製品データシート([DS961](#))
- GW2AN-18X & 9X FPGA 製品データシート([DS971](#))
- GW2AN-55 FPGA 製品データシート([DS976](#))

3.2 GPIO バンク

GPIO の汎用属性：

- すべてのバンクはエミュレート LVDS 差動出力をサポートしますが、外部抵抗ネットワークが必要です。
- すべてのバンクは、プルアップ、プルダウン、およびバスホールド設定をサポートしています。
- 各バンクは 1 つのピン電圧をサポートします。
- 各バンクは、外部ピンまたは内部リファレンス電圧発生器からの 1 つのリファレンス電圧信号をサポートしています。

3.3 電源供給の要件

コア電圧(V_{CC})とピン電圧(V_{CCIO})が特定のしきい値に達すると、内部パワーオンリセット信号(PoR)がアサートされ、GOWIN セミコンダクター FPGA 製品のコアロジックがアクティブになります。コンフィギュレーション中、デバイスのすべての GPIO は内部の弱いプルアップ^[1]を持ち、コンフィギュレーション後、I/O の状態はユーザーデザインおよび制約によって決定されます。コンフィギュレーション関連 I/O の状態はコンフィギュレーションモードにより異なります。GOWIN セミコンダクターFPGA 製品には、コア電圧とピン電圧のパワーオン/パワーオフ・シーケンス要件はありません。

注記：

GW2AN-18X/9X の場合は内部の弱いプルダウンです。

各バンクは 1 つのリファレンス電圧(V_{REF})入力をサポートします。バンク内の任意のピンをリファレンス電圧入力ピンとして構成できます。SSTL や HSTL などの規格をサポートするために、リファレンス電圧はピン電圧の半分に設定します。また、入力リファレンス電圧は、内部リファレンス電圧発生器によって生成することもできます。各バンクにはリファレンス電圧バスが 1 つしかないため、1 つのバンクの内部リファレンス電圧発生器と外部リファレンス電圧入力ピンを同時に有効にすることはできません。

GOWIN セミコンダクターFPGA 製品の GPIO バッファには、A および B とマークされる 2 つの入出力ピンがあります。ピン A は差動信号の T(True)側に対応し、ピン B は差動信号の C(Comp)側に対応します。

3.3.1 LVCMOS バッファの構成

すべての GPIO には、アプリケーションに応じて複数のモードに構成できる LVCMOS バッファが含まれています。各 LVCMOS バッファは、弱いプルアップ、弱いプルダウン、およびバスホールドに構成できます。弱いプルアップおよび弱いプルダウンは、ワイヤード AND、ワイヤード OR 等のロジック制御に広く適用できる固定特徴を提供します。バスホールドは最小の電力消費で信号の前の状態をラッチし、バスホールドをオフにすると入力リーク電流が減少します。

すべての LVCMOS バッファは、プログラマブルなドライブ強度を持っています。各規格のドライブ強度オプションについては、対応するデータシートの“I/O 規格”セクションを参照してください。GOWIN セミコンダクターFPGA 製品のプログラム可能なドライブ強度は各設定の最低限のドライブ強度を保証するだけです。

ヒステリシス設定は、主にノイズの多い場合のレベルの急激変動を防ぐために使用され、すべての LVCMOS バッファはヒステリシスオプションをサポートします。

差動ペアが 2 つのシングルエンドピンとして構成されている場合、ピン間の相対遅延が最小になり、信号の一貫性が最高になります。

3.3.2 差動バッファの構成

GPIO バッファが差動モードに構成された場合、入力ヒステリシスとバスホールド特性は無効になります。

次の GW1N シリーズ製品および GW2A シリーズ製品は、オンチップのプログラマブル 100Ω 差動入力終端抵抗をサポートします。

- GW1N-4、GW1NR-4、GW1NRF-4B、GW1N-9、GW1NR-9、GW1N-1、GW1NR-1 のバンク 0。
- GW1N-1S、GW1NS-4、GW1NS-4C、GW1NSR-4、GW1NSR-4C、GW1NSER-4C、GW2A-18、GW2A-55、GW2AN-55、GW2ANR-18、GW2AR-18 のバンク 0 と 1。
- GW1N-2、GW1NR-2、GW1N-1P5 のバンク 2。
- GW2AN-18X、GW2AN-9X のバンク 4 と 5。

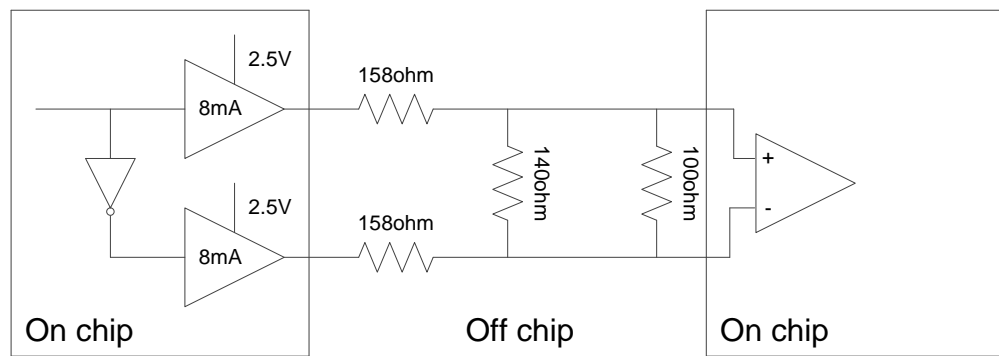
すべてのシングルエンド GPIO バッファペアは、LVPECL33E、MLVDS25E、BLVDS25E などのエミュレート LVDS 差動出力規格準拠のように構成できます。同時に、外部終端を追加する必要があります。

3.4 エミュレート差動回路終端方式

3.4.1 エミュレート LVDS

GOWIN セミコンダクターFPGA 製品は、LVCMOS 出力と外部終端を利用して、互換性のある LVDS 出力規格を構築することができます。その外部終端方式は下図に示すとおりです。

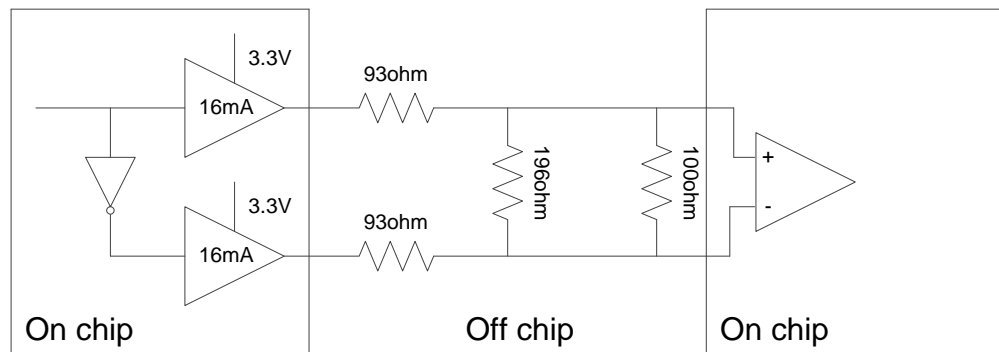
図 3-1 LVDS25E の外部終端



3.4.2 エミュレート LVPECL

GOWIN セミコンダクターFPGA 製品は、LVCMOS 出力と外部終端を利用して、互換性のある LVPECL 出力規格を構築することができます。その外部終端方式は下図に示すとおりです。

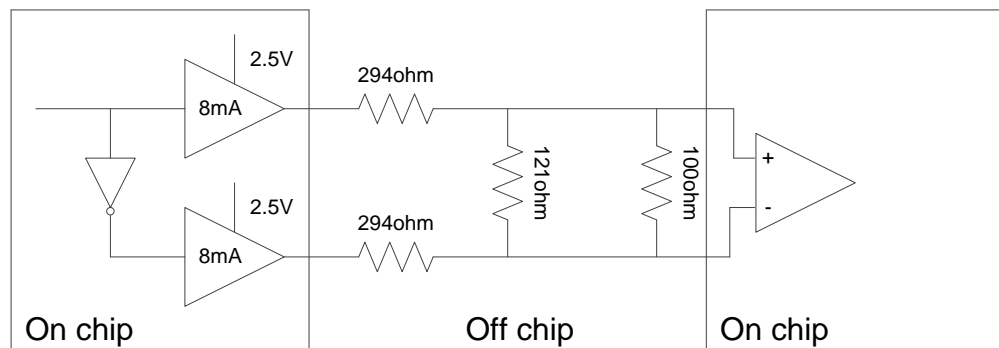
図 3-2 LVPECL の外部終端



3.4.3 エミュレート RSDS

GOWIN セミコンダクターFPGA 製品は、LVCMOS 出力と外部終端を利用して、互換性のある RSDS 出力規格を構築することができます。その外部終端方式は下図に示すとおりです。

図 3-3 RSDS の外部終端

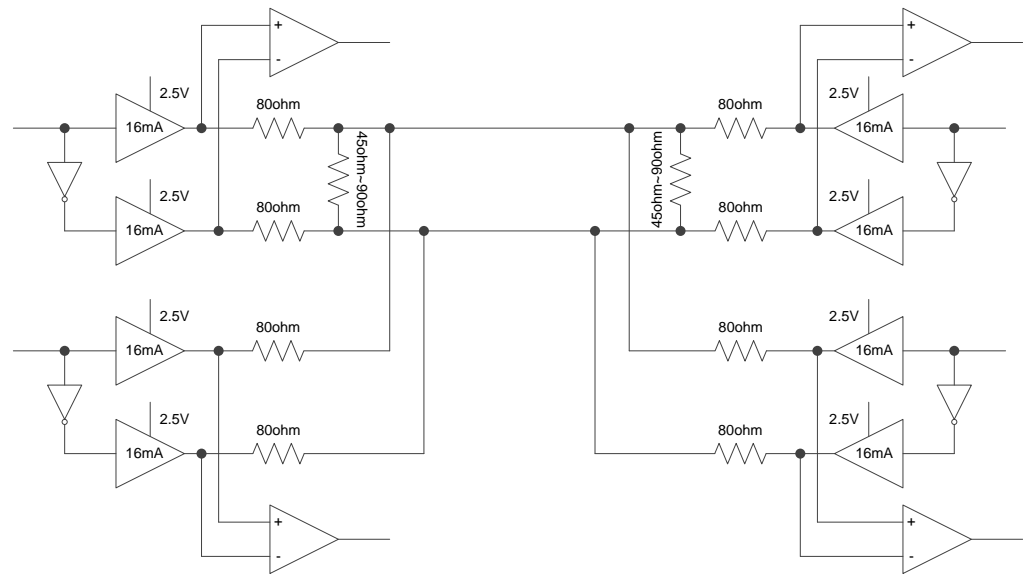


3.4.4 エミュレート BLVDS

GOWIN セミコンダクターFPGA 製品は、LVCMOS 出力と外部終端を利用して、互換性のある BLVDS 出力規格を構築することができます。その

外部終端方式は下図に示すとおりです。

図 3-4 BLVDS の外部終端



3.5 GPIO の構成

Gowin ソフトウェアの Floorplanner を使用して GPIO の位置と属性を設定するか、CST ファイルをカスタマイズしてそれを実現できます。以下は、CST ファイルによりサポートされている物理制約について説明します。

3.5.1 位置

GPIO の物理位置をロックします。

```
IO_LOC "xxx" H4 exclusive;
```

3.5.2 レベル規格

GPIO の規格を設定します。

```
IO_PORT "xxx" IO_TYPE=LVCMS18D;
```

3.5.3 ドライブ強度

出力ピンまたは双方向ピンのドライブ強度を設定します。

```
IO_PORT "xxx" DRIVE=12;
```

3.5.4 プルアップ/ダウンモード

プルアップ/ダウンモードを設定します。UP : プルアップ、DOWN : プルダウン、KEEPER : バスホールド、NONE : ハイインピーダンス。

```
IO_PORT "xxx" PULL_MODE=DOWN;
```

3.5.5 リファレンス電圧

GPIO のリファレンス電圧を設定します。リファレンス電圧は外部ピンまたは内部リファレンス電圧発生器から提供されます。

```
IO_PORT "xxx" VREF=VREF1_LOAD;
```

3.5.6 ヒステリシス

入力ピンまたは双方向ピンのためにヒステリシスを設定します。小さい順：NONE->H2L->L2H->HIGH。

```
IO_PORT "xxx" HYSTERESIS=L2H;
```

3.5.7 オープンドレイン

出力ピンまたは双方向ピンのためにオープンドレインを開閉し、ON/OFF オプションを提供します。

```
IO_PORT "xxx" OPEN_DRAIN=ON;
```

3.5.8 シングルエンド終端抵抗

シングルエンド信号のために終端抵抗を設定し、OFF 及び ON オプションを提供します。

```
IO_PORT "xxx" SINGLE_RESISTOR=ON;
```

3.5.9 差動終端抵抗

差動信号のために終端抵抗を設定し、OFF 及び ON オプションを提供します。

```
IO_PORT "xxx" Diff_RESISTOR=ON;
```

3.6 GPIO プリミティブ

IO Buffer は、機能によって通常の Buffer、エミュレート LVDS(ELVDS)、および True LVDS(TLVDS)に分類できます。

3.6.1 IBUF

プリミティブの紹介

IBUF(Input Buffer)は入力バッファです。

ポート図

図 3-5 IBUF のポート図



ポートの説明

表 3-1 IBUF のポートの説明

ポート	I/O	説明
I	入力	データ入力信号

ポート	I/O	説明
O	出力	データ出力

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
IBUF uut(
    .O(O),
    .I(I)
);
```

VHDL でのインスタンス化 :

```
COMPONENT IBUF
    PORT (
        O:OUT std_logic;
        I:IN std_logic
    );
END COMPONENT;
uut:IBUF
    PORT MAP(
        O=>O,
        I=>I
    );
```

3.6.2 OBUF

プリミティブの紹介

OBUF(Output Buffer)は出力バッファです。

ポート図

図 3-6 OBUF のポート図



ポートの説明

表 3-2 OBUF のポートの説明

ポート	I/O	説明
I	入力	データ入力信号
O	出力	データ出力

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
OBUF uut(
    .O(O),
    .I(I)
);
```

VHDL でのインスタンス化 :

```
COMPONENT OBUF
    PORT (
        O:OUT std_logic;
        I:IN std_logic
    );
END COMPONENT;
uut:OBUF
    PORT MAP(
        O=>O,
        I=>I
    );
```

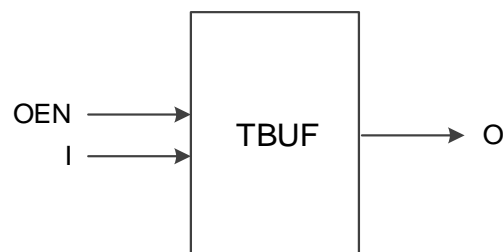
3.6.3 TBUF

プリミティブの紹介

TBUF(Output Buffer with Tristate Control)はトライステートバッファで、アクティブ Low です。

ポート図

図 3-7 TBUF のポート図



ポートの説明

表 3-3 TBUF のポートの説明

ポート	I/O	説明
I	入力	データ入力信号
OEN	入力	トライステート出力イネーブル
O	出力	データ出力

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
TBUF uut(
    .O(O),
    .I(I),
    .OEN(OEN)
);
```

VHDL でのインスタンス化 :

```
COMPONENT TBUF
    PORT (
        O:OUT std_logic;
        I:IN std_logic;
        OEN:IN std_logic
    );
END COMPONENT;

uut:TBUF
    PORT MAP(
        O=>O,
        I=>I,
        OEN=> OEN
    );
```

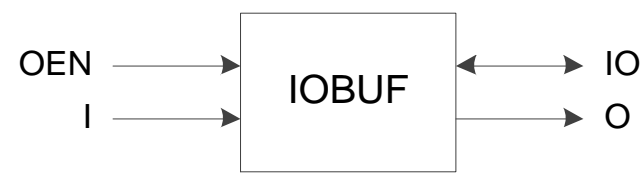
3.6.4 IOBUF

プリミティブの紹介

IOBUF(Bi-Directional Buffer)は双方向バッファです。OEN が High の場合は入力バッファとして使用され、Low の場合は出力バッファとして使用されます。

ポート図

図 3-8 IOBUF のポート図



ポートの説明

表 3-4 IOBUF のポートの説明

ポート	I/O	説明
I	入力	データ入力信号
OEN	入力	トリステート出力イネーブル
IO	双方向	入出力信号
O	出力	データ出力信号

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
IOBUF uut(  
    .O(O),  
    .IO(IO),  
    .I(I),  
    .OEN(OEN)  
);
```

VHDL でのインスタンス化 :

```
COMPONENT IOBUF  
    PORT (  
        O:OUT std_logic;  
        IO:INOUT std_logic;  
        I:IN std_logic;  
        OEN:IN std_logic  
    );  
END COMPONENT;  
uut:IOBUF  
    PORT MAP(  

```

```

O=>O,
IO=>IO,
I=>I,
OEN=> OEN
);

```

3.6.5 LVDS Input Buffer

プリミティブの紹介

LVDS の差動入力、TLVDS_IBUF と ELVDS_IBUF の 2 種類があります。

TLVDS_IBUF(True LVDS Input Buffer)はトゥルー差動入力バッファです。

注記：

GW1NZ-1、GW1N-1S デバイスは TLVDS_IBUF をサポートしません。

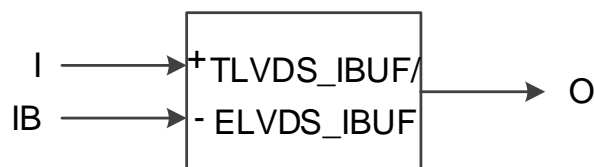
ELVDS_IBUF(Emulated LVDS Input Buffer)はエミュレート差動入力バッファです。

注記：

GW1NZ-1 デバイスは ELVDS_IBUF をサポートしません。

ポート図

図 3-9 TLVDS_IBUF/ELVDS_IBUF のポート図



ポートの説明

表 3-5 TLVDS_IBUF/ELVDS_IBUF のポートの説明

ポート	I/O	説明
I	入力	差動入力 A
IB	入力	差動入力 B
O	出力	データ出力

プリミティブのインスタンス化

例 1

Verilog でのインスタンス化：

```

TLVDS_IBUF uut(
    .O(O),

```

```

        .I(I),
        .IB(IB)
    );
VHDL でのインスタンス化 :
COMPONENT TLVDS_IBUF
    PORT (
        O:OUT std_logic;
        I:IN std_logic;
        IB:IN std_logic
    );
END COMPONENT;
uut:TLVDS_IBUF
    PORT MAP(
        O=>O,
        I=>I,
        IB=> IB
    );

```

例 2

Verilog でのインスタンス化 :

```

ELVDS_IBUF uut(
    .O(O),
    .I(I),
    .IB(IB)
);

```

VHDL でのインスタンス化 :

```

COMPONENT ELVDS_IBUF
    PORT (
        O:OUT std_logic;
        I:IN std_logic;
        IB:IN std_logic
    );
END COMPONENT;
uut:ELVDS_IBUF
    PORT MAP(
        O=>O,

```

```

        I=>I,
        IB=> IB
    );

```

3.6.6 LVDS Ouput Buffer

プリミティブの紹介

LVDS の差動出力は、TLVDS_OBUF と ELVDS_OBUF の 2 種類があります。

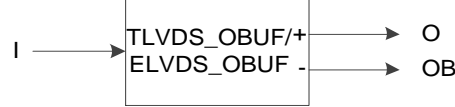
TLVDS_OBUF(True LVDS Output Buffer)はトゥルー差動出力バッファです。

注記：
GW1N-1、GW1NR-1、GW1NZ-1、GW1N-1S は TLVDS_OBUF をサポートしません。

ELVDS_OBUF(Emulated LVDS Output Buffer)はエミュレート差動出力バッファです。

ポート図

図 3-10 TLVDS_OBUF/ELVDS_OBUF のポート図



ポートの説明

表 3-6 TLVDS_OBUF/ELVDS_OBUF のポートの説明

ポート	I/O	説明
I	入力	データ入力信号
OB	出力	差動出力 B
O	出力	差動出力 A

プリミティブのインスタンス化

例 1

Verilog でのインスタンス化：

```

TLVDS_OBUF uut(
    .O(O),
    .OB(OB),
    .I(I)
);

```

VHDL でのインスタンス化：

```

COMPONENT TLVDS_OBUF
  PORT (
    O:OUT std_logic;
    OB:OUT std_logic;
    I:IN std_logic
  );
END COMPONENT;
uut:TLVDS_OBUF
  PORT MAP(
    O=>O,
    OB=>OB,
    I=> I
  );

```

例 2

Verilog でのインスタンス化 :

```

ELVDS_OBUF uut(
  .O(O),
  .OB(OB),
  .I(I)
);

```

VHDL でのインスタンス化 :

```

COMPONENT ELVDS_OBUF
  PORT (
    O:OUT std_logic;
    OB:OUT std_logic;
    I:IN std_logic
  );
END COMPONENT;
uut:ELVDS_OBUF
  PORT MAP(
    O=>O,
    OB=>OB,
    I=> I
  );

```

3.6.7 LVDS Tristate Buffer

プリミティブの紹介

LVDS トライステート差動出力には TLVDS_TBUF と ELVDS_TBUF の 2 種類があります。

TLVDS_TBUF(True LVDS Tristate Buffer)はトゥルー差動トライステートバッファで、アクティブ Low です。

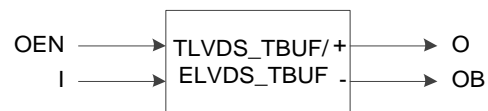
注記：

GW1N-1、GW1NR-1、GW1NZ-1、GW1N-1S は TLVDS_TBUF をサポートしません。

ELVDS_TBUF(Emulated LVDS Tristate Buffer)はエミュレート差動トライステートバッファで、アクティブ Low です。

ポート図

図 3-11 TLVDS_TBUF/ELVDS_TBUF のポート図



ポートの説明

表 3-7 TLVDS_TBUF/ELVDS_TBUF のポートの説明

ポート	I/O	説明
I	入力	データ入力信号
OEN	入力	トライステート出力イネーブル
OB	出力	差動出力 B
O	出力	差動出力 A

プリミティブのインスタンス化

例 1

Verilog でのインスタンス化：

```

TLVDS_TBUF uut(
    .O(O),
    .OB(OB),
    .I(I),
    .OEN(OEN)
);
  
```

VHDL でのインスタンス化：

```

COMPONENT TLVDS_TBUF
    PORT (
  
```

```

        O:OUT std_logic;
        OB:OUT std_logic;
        I:IN std_logic;
        OEN:IN std_logic

    );
END COMPONENT;
uut:TLVDS_TBUF
    PORT MAP(
        O=>O,
        OB=>OB,
        I=> I,
        OEN=>OEN
    );

```

例 2

Verilog でのインスタンス化 :

```

ELVDS_TBUF uut(
    .O(O),
    .OB(OB),
    .I(I),
    .OEN(OEN)
);

```

VHDL でのインスタンス化 :

```

COMPONENT ELVDS_TBUF
    PORT (
        O:OUT std_logic;
        OB:OUT std_logic;
        I:IN std_logic;
        OEN:IN std_logic

    );
END COMPONENT;
uut:ELVDS_TBUF
    PORT MAP(
        O=>O,
        OB=>OB,
        I=> I,

```


OEN=>OEN

);

3.6.8 LVDS Inout Buffer

プリミティブの紹介

LVDS 差動入出力は、TLVDS_IOBUF と ELVDS_IOBUF の 2 種類あります。

TLVDS_IOBUF(True LVDS Bi-Directional Buffer)は、トウルー双方向バッファです。OEN が High の場合は、トウルー差動入力バッファとして使用され、OEN が Low の場合は、トウルー差動出力バッファとして使用されます。

サポートされるデバイス

表 3-8 TLVDS_IOBUF 対応デバイス

ファミリー	シリーズ	デバイス
Arora ファミリー	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C
LittleBee® ファミリー	GW1N	GW1N-4, GW1N-4B, GW1N-4D
	GW1NR	GW1NR-4, GW1NR-4B, GW1NR-4D
	GW1NRF	GW1NRF-4B

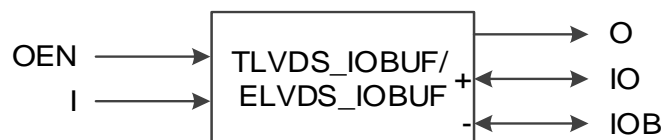
ELVDS_IOBUF(Emulated LVDS Bi-Directional Buffer)はエミュレート差動双方向バッファです。OEN が High の場合はエミュレート差動入力バッファとして使用され、OEN が Low の場合はエミュレート差動出力バッファとして使用されます。

注記：

GW1NZ-1 デバイスは ELVDS_IOBUF をサポートしません。

ポート図

図 3-12 TLVDS_IOBUF/ELVDS_IOBUF のポート図



ポートの説明

表 3-9 TLVDS_IOBUF/ELVDS_IOBUF のポートの説明

ポート	I/O	説明
I	入力	データ入力信号

ポート	I/O	説明
OEN	入力	トライステート出力イネーブル
O	出力	データ出力
IOB	双方向	差動入出力 B
IO	双方向	差動入出力 A

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
ELVDS_IOBUF uut(
    .O(O),
    .IO(IO),
    .IOB(IOB),
    .I(I),
    .OEN(OEN)
);
```

VHDL でのインスタンス化 :

```
COMPONENT ELVDS_IOBUF
    PORT (
        O:OUT std_logic;
        IO:INOUT std_logic;
        IOB:INOUT std_logic;
        I:IN std_logic;
        OEN:IN std_logic
    );
END COMPONENT;
uut:ELVDS_IOBUF
    PORT MAP(
        O=>O,
        IO=>IO,
        IOB=>IOB,
        I=> I,
        OEN=>OEN
    );
```

3.6.9 MIPI_IBUF

プリミティブの紹介

MIPI_IBUF(MIPI Input Buffer)は、抵抗の動的構成をサポートする HS 入力モードと、LP 双方向モードとの 2 つの動作モードがあります。

サポートされるデバイス

表 3-10 MIPI_IBUF 対応デバイス

ファミリー	シリーズ	デバイス
LittleBee®	GW1N	GW1N-9, GW1N-9C, GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B, GW1N-1S
	GW1NR	GW1NR-9, GW1NR-9C, GW1NR-2, GW1NR-2B
	GW1NS	GW1NS-4, GW1NS-4C
	GW1NSER	GW1NSER-4C
	GW1NSR	GW1NSR-4, GW1NSR-4C
Arora	GW2AN	GW2AN-18X, GW2AN-9X

機能の説明

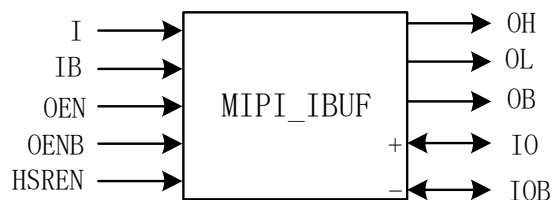
MIPI_IBUF は LP、HS モードをサポートします。IO、IOB は pad に接続されます。

LP モード: 双方向をサポートします。OEN が Low の場合、I は入力で、IO は出力です。OEN が High の場合、IO は入力で、OL は出力です。OENB が Low の場合、IB は入力で、IOB は出力です。OENB が High の場合、IOB は出力で、OB は出力です。

HS モード: IO と IOB は差動入力で、OH は出力です。HSREN は終端抵抗を制御します。

ポート図

図 3-13 MIPI_IBUF のポート図



ポートの説明

表 3-11 MIPI_IBUF のポートの説明

ポート	I/O	説明
I	入力	LP モードでは、OEN が Low の場合、I は入力です。
IB	入力	LP モードでは、OENB が Low の場合、IB は入力です。

ポート	I/O	説明
HSREN	入力	HS モードで終端抵抗を制御します。
OEN	入力	LP モードでトライステートを制御します。
OENB	入力	LP モードでトライステートを制御します。
OH	出力	HS モードではデータ出力です。
OL	出力	LP モードでは、OEN が High の場合、OL は出力です。
OB	出力	LP モードでは、OENB が High の場合、OB は出力です。
IO	双方向	<ul style="list-style-type: none"> ● LP モードでは、OEN が Low の場合、IO は出力で、OEN が High の場合、IO は入力です。 ● HS モードでは、IO は入力です。
IOB	双方向	<ul style="list-style-type: none"> ● LP モードでは、OENB が Low の場合、IOB は出力で、OENB が High の場合、IOB は入力です。 ● HS モードでは、IOB は入力です。

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```

MIPI_IBUF uut(
    .OH(OH),
    .OL(OL),
    .OB(OB),
    .IO(IO),
    .IOB(IOB),
    .I(I),
    .IB(IB),
    .OEN(OEN),
    .OENB(OENB),
    HSREN(HSREN)
);

```

VHDL でのインスタンス化 :

```

COMPONENT MIPI_IBUF
PORT (
    OH:OUT std_logic;
    OL: OUT std_logic;
    OB:OUT std_logic;
    IO:INOUT std_logic;
    IOB:INOUT std_logic;

```

```

        I:IN std_logic;
        IB:IN std_logic;
        OEN:IN std_logic;
        OENB:IN std_logic;
        HSREN:IN std_logic
    );
END COMPONENT;
uut: MIPI_IBUF
    PORT MAP(
        OH=>OH,
        OL=>OL,
        OB=>OB,
        IO=>IO,
        IOB=>IOB,
        I=>I,
        IB=>IB,
        OEN=>OEN,
        OENB=>OENB,
        HSREN=>HSREN
    );

```

3.6.10 MIPI_OBUF

プリミティブの紹介

MIPI_OBUF には HS と LP の 2 つの動作モードがあります。

MIPI_OBUF(MIPI Output Buffer)は、MIPI 出力バッファです。MODESEL が High の場合は、(HS)MIPI 高速出力バッファとして使用され、MODESEL が Low の場合は、(LP)MIPI 低消費電力出力バッファとして使用されます。

サポートされるデバイス

表 3-12 MIPI_OBUF 対応デバイス

ファミリー	シリーズ	デバイス
LittleBee® ファミリー	GW1N	GW1N-9, GW1N-9C
	GW1NR	GW1NR-9, GW1NR-9C
	GW1NS	GW1NS-4, GW1NS-4C
	GW1NSER	GW1NSER-4C
	GW1NSR	GW1NSR-4, GW1NSR-4C

ポート図

図 3-14 MIPI_OBUF のポート図



ポートの説明

表 3-13 MIPI_OBUF のポートの説明

ポート	I/O	説明
I	入力	HS モードまたは LP モードにおけるデータ入力 A
IB	入力	LP モードにおけるデータ入力 B
MODESEL	入力	モード選択(HS または LP)
O	出力	データ出力 A。(HS モードでは差動出力 A、LP モードではシングルエンド出力 A)
OB	出力	データ出力 B。(HS モードでは差動出力 B、LP モードではシングルエンド出力 B)

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
MIPI_OBUF uut(
    .O(O),
    .OB(OB),
    .I(I),
    .IB(IB),
    .MODESEL(MODESEL)
);
```

VHDL でのインスタンス化 :

```
COMPONENT MIPI_OBUF
    PORT (
        O:OUT std_logic;
        OB:OUT std_logic;
        I:IN std_logic;
        IB:IN std_logic;
        MODESEL:IN std_logic
    );
END COMPONENT;
```

```

uut: MIPI_OBUF
  PORT MAP(
    O=>O,
    OB=>OB,
    I=>I,
    IB=>IB,
    MDOESEL=>MODESEL
  );

```

3.6.11 MIPI_OBUF_A

プリミティブの紹介

MIPI_OBUF_A には HS と LP の 2 種類の動作モードがあります。

MIPI_OBUF_A(MIPI Output Buffer with IL Signal)は、MIPI 出力バッファです。MODESEL が High の場合は(HS)MIPI 高速出力バッファとして使用され、MODESEL が Low の場合は(LP)MIPI 低消費電力出力バッファとして使用されます。MIPI_OBUF と比較して、LP モードにおける入力 A として IL ポートが追加されています。

サポートされるデバイス

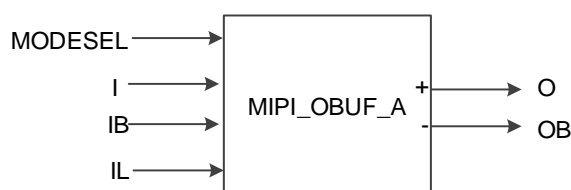
MIPI_OBUF_A 対応デバイスについては、および表 3-14 を参照してください。

表 3-14 MIPI_OBUF_A 対応デバイス(追加)

ファミリー	シリーズ	デバイス
LittleBee® ファミリー	GW1N	GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B
	GW1NR	GW1NR-2, GW1NR-2B

ポート図

図 3-15 MIPI_OBUF_A のポート図



ポートの説明

表 3-15 MIPI_OBUF_A のポートの説明

ポート	I/O	説明
I	入力	HS モードにおけるデータ入力 A

ポート	I/O	説明
IB	入力	LP モードにおけるデータ入力 B
IL	入力	LP モードにおけるデータ入力 A
MODESEL	入力	モード選択(HS または LP)
O	出力	データ出力 A。(HS モードでは差動出力 A、LP モードではシングルエンド出力 A)
OB	出力	データ出力 B。(HS モードでは差動出力 B、LP モードではシングルエンド出力 B)

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```

MIPI_OBUF_A uut(
    .O(O),
    .OB(OB),
    .I(I),
    .IB(IB),
    .IL(IL),
    .MODESEL(MODESEL)
);

```

VHDL でのインスタンス化 :

```

COMPONENT MIPI_OBUF_A
    PORT (
        O:OUT std_logic;
        OB:OUT std_logic;
        I:IN std_logic;
        IB:IN std_logic;
        IL: IN std_logic;
        MODESEL:IN std_logic
    );
END COMPONENT;

uut: MIPI_OBUF_A
    PORT MAP(
        O=>O,
        OB=>OB,
        I=>I,
        IB=>IB,
        IL=>IL,

```


MDOESEL=>MODESEL

);

3.6.12 I3C_IOBUF

プリミティブの紹介

I3C_IOBUF には Normal と I3C の 2 つの動作モードがあります。

I3C_IOBUF(I3C Bi-Directional Buffer)は、I3C 双方向バッファです。
MODESEL が High の場合は I3C 双方向バッファとして使用され、
MODESEL が Low の場合は通常の双方向バッファとして使用されます。

サポートされるデバイス

表 3-16 I3C_IOBUF 対応デバイス

ファミリ	シリーズ	デバイス
LittleBee® ファミリ	GW1N	GW1N-9, GW1N-9C
	GW1NR	GW1NR-9, GW1NR-9C
	GW1NS	GW1NS-4, GW1NS-4C
	GW1NSER	GW1NSER-4C
	GW1NSR	GW1NSR-4, GW1NSR-4C

ポート図

図 3-16 I3C_IOBUF のポート図



ポートの説明

表 3-17 I3C_IOBUF のポート図

ポート	I/O	説明
I	入力	データ入力信号
IO	双方向	入出力信号
MODESEL	入力	モード選択(Normal または I3C)
O	出力	データ出力

プリミティブのインスタンス化

Verilog でのインスタンス化 :

I3C_IOBUF uut(

```
.O(O),
.IO(IO),
.I(I),
.MODESEL(MODESEL)
);
VHDL でのインスタンス化 :
COMPONENT I3C_IOBUF
  PORT (
    O:OUT std_logic;
    IO:INOUT std_logic;
    I:IN std_logic;
    MODESEL:IN std_logic
  );
END COMPONENT;
uut: I3C_IOBUF
  PORT MAP(
    O=>O,
    IO=>IO,
    I=>I,
    MDOESEL=>MODESEL
  );
```

3.6.13 MIPI_IBUF_HS/MIPI_IBUF_LP

プリミティブの紹介

MIPI_IBUF_HS は差動入力で HS モードを実装し、MIPI_IBUF_LP はシングルエンド入力で LP モードを実装します。

サポートされるデバイス

表 3-18 MIPI_IBUF_HS/MIPI_IBUF_LP 対応デバイス

ファミリー	シリーズ	デバイス
LittleBee® ファミリー	GW1NR	GW1NR-2

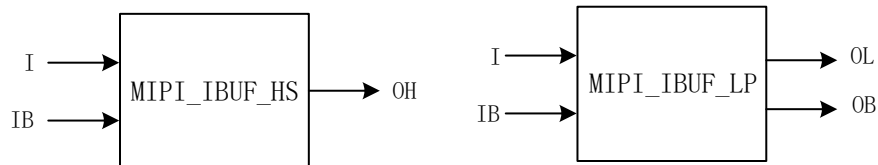
機能の説明

ユーザーは、MIPI_IBUF_HS と MIPI_IBUF_LP の組み合わせで HS モードと LP モードを実装でき、Floorplanner を使用してその位置を制約できます。MIPI_IBUF_HS の入力 I と MIPI_IBUF_LP の I は同じ信号に接続する必要があり、MIPI_IBUF_HS の入力 IB と MIPI_IBUF_LP の IB は同じ信

号に接続する必要があります。

ポート図

図 3-17 MIPI_IBUF_HS/MIPI_IBUF_LP のポート図



ポートの説明

表 3-19 MIPI_IBUF_HS のポートの説明

ポート	I/O	説明
I	入力	HS モードにおける差動入力 A
IB	入力	HS モードにおける差動入力 B
OH	出力	HS モードにおけるデータ出力

表 3-20 MIPI_IBUF_LP のポートの説明

ポート	I/O	説明
I	入力	LP モードにおけるシングルエンド入力 A
IB	入力	LP モードにおけるシングルエンド入力 B
OL	出力	LP モードにおける出力 A
OB	出力	LP モードにおける出力 B

接続ルール

- MIPI_IBUF_HS の出力 OH は lologic(入出力ロジック)に接続できます。
- MIPI_IBUF_LP の出力 OL と OB は lologic に接続できません。

プリミティブのインスタンス化

Verilog でのインスタンス化：

```

MIPI_IBUF_HS hs (
    .OH(OH),
    .I(I),
    .IB(IB)
);
MIPI_IBUF_LP lp (
    .OL(OL),
    .OB(OB),
    .I(I),

```

```
.IB(IB)
);
```

VHDL でのインスタンス化 :

```
COMPONENT MIPI_IBUF_HS
  PORT (
    OH:OUT std_logic;
    I:IN std_logic;
    IB:IN std_logic
  );
END COMPONENT;
COMPONENT MIPI_IBUF_LP
  PORT (
    OL: OUT std_logic;
    OB:OUT std_logic;
    I:IN std_logic;
    IB:IN std_logic
  );
END COMPONENT;
hs: MIPI_IBUF_HS
  PORT MAP(
    OH=>OH,
    I=>I,
    IB=>IB
  );
lp: MIPI_IBUF_LP
  PORT MAP(
    OL=>OL,
    OB=>OB,
    I=>I,
    IB=>IB
  );
```

3.6.14 ELVDS_IBUF_MIPi

プリミティブの紹介

ELVDS_IBUF_MIPi (Emulated LVDS Input MIPi Buffer) は、HS 入力モードと LP 入力モードの 2 つの動作モードを同時に有効にすることをサポートします。A ポートは HS モードをサポートし、B ポートは LP モードのみを実装します。

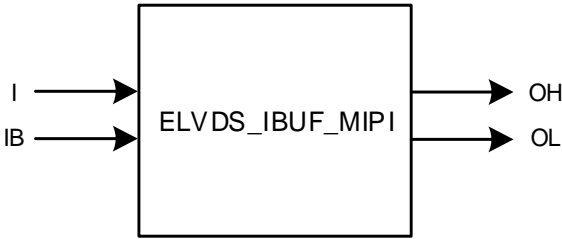
機能の説明

ELVDS_IBUF_MIPi は LP モードと HS モードをサポートします。I と IB は pad に接続されます。

- LP モード : IB は入力で、OL は出力です。
- HS モード : I と IB は差動入力で、OH は出力です。

ポート図

図 3-18 ELVDS_IBUF_MIPi のポート図



ポートの紹介

表 3-21 MIPi_IBUF のポートの説明

ポート	I/O	説明
I	Input	HSモードでは、Iは入力です。
IB	Input	LPモードでは、IBは入力です。
OH	Output	HSモードにおけるデータ出力信号
OL	Output	LPモードにおけるデータ出力信号

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
ELVDS_IBUF_MIPi uut(  
    .OH(OH),  
    .OL(OL),  
    .I(I),  
    .IB(IB)  
);
```

Vhdl でのインスタンス化 :

```
COMPONENT ELVDS_IBUF_MIPi
```

```
    PORT (  
        OH:OUT std_logic;  
        OL: OUT std_logic;  
        I:IN std_logic;  
        IB:IN std_logic  
    );  
END COMPONENT;  
uut: ELVDS_IBUF_MIP1  
    PORT MAP(  
        OH=>OH,  
        OL=>OL,  
        I=>I,  
        IB=>IB  
    );
```

4 入出力ロジック

GOWIN セミコンダクターFPGA 製品の入出力ロジックは、SDR、DDR などの動作モードをサポートします。各動作モードでは、ピン制御(または I/O 差動信号ペア)はまた出力信号、入力信号、INOUT 信号、及びトライステート出力信号(トライステート制御付きの出力信号)に設定できます。

注記：

- GW1N-1、GW1NR-1、GW1NZ-1 デバイスの IOL6、IOR6 ピンは IO ロジックをサポートしません。
- GW1N-2、GW1NR-2、GW1N-1P5、GW1N-2B、GW1N-1P5B、および GW1NR-2B デバイスの IOT2、IOT3A ピンは IO ロジックをサポートしません。
- GW1N-4、GW1N-4B、GW1NR-4、GW1NR-4B、GW1NRF-4B、GW1N-4D、および GW1NR-4D デバイスの IOL10、IOR10 ピンは IO ロジックをサポートしません。

図 4-1 は GOWIN セミコンダクターFPGA 製品の入出力ロジックの出力を示します。

図 4-1 入出力ロジックの説明図 -出力

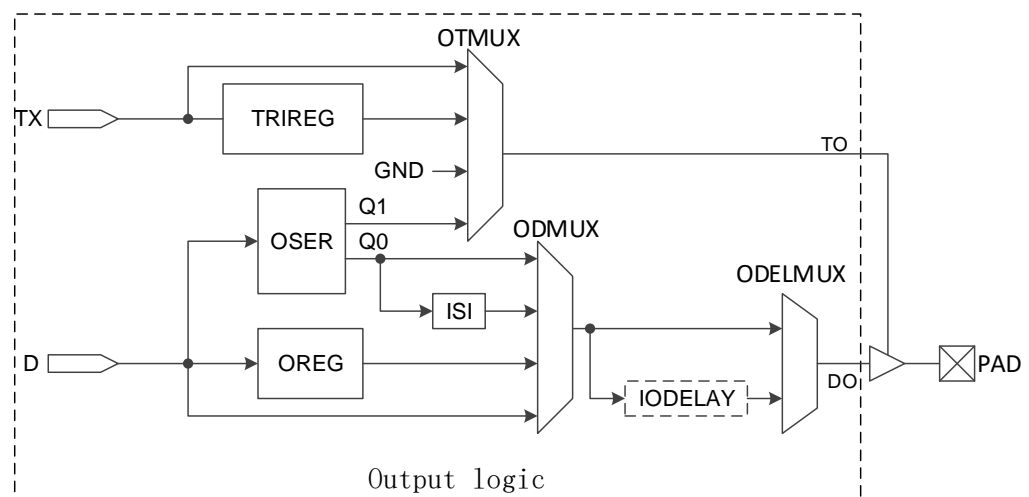
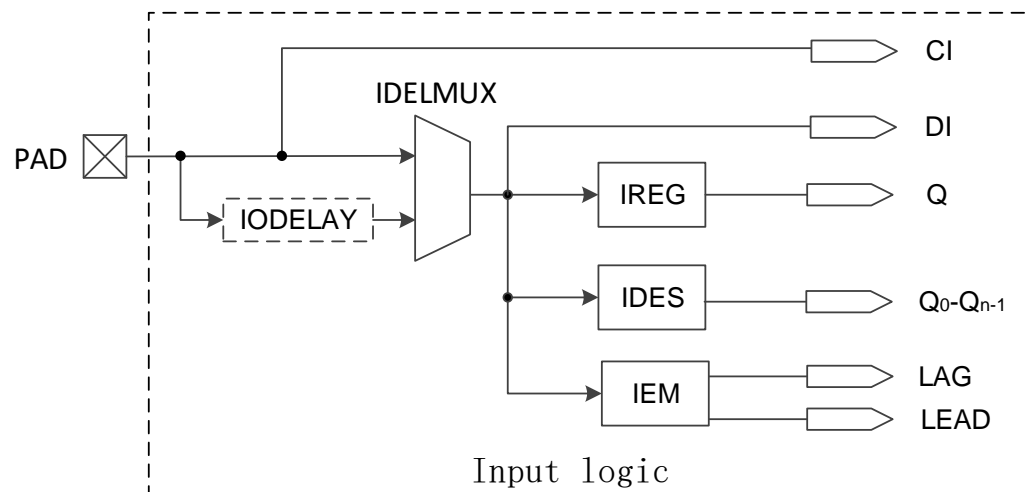


図 4-2 は GOWIN セミコンダクターFPGA 製品の入出力ロジックの入力を示します。

図 4-2 入出力ロジックの説明図 -入力



注記：

CI は GCLK 入力信号であり、ファブリックに接続できません。DI はファブリックに直接入力されます。

4.1 SDR モード

入出力ロジックは SDR モードをサポートし、入力レジスタ(IREG)、出力レジスタ(OREG)、およびトライステート制御レジスタ(TRIREG)を提供します。その機能は CFU の FF/LATCH と同様です。FF/LATCH の入力 D が Buffer/IODELAY によって駆動され、Buffer/IODELAY が他の Iologic を駆動しない場合、または FF/LATCH の出力 Q が Buffer/IODELAY のみを駆動し、Buffer が MIPI バッファでない場合、FF/LATCH は IOLOGIC として使用できます。

4.2 DDR モードの入出力ロジック

4.2.1 IDDR

プリミティブの紹介

IDDR(Dual Data Rate Input)は、ダブルデータレートの実現します。

機能の説明

IDDR モードでは、出力データは同じクロックエッジで FPGA ロジックに提供されます。IDDR モードのブロック図は、タイミング図は図 4-4 に示すとおりです。

図 4-3 IDDR のブロック図

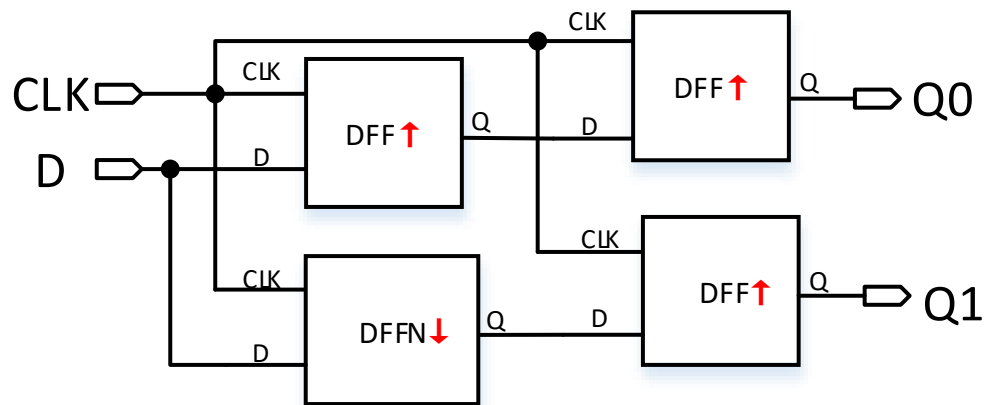
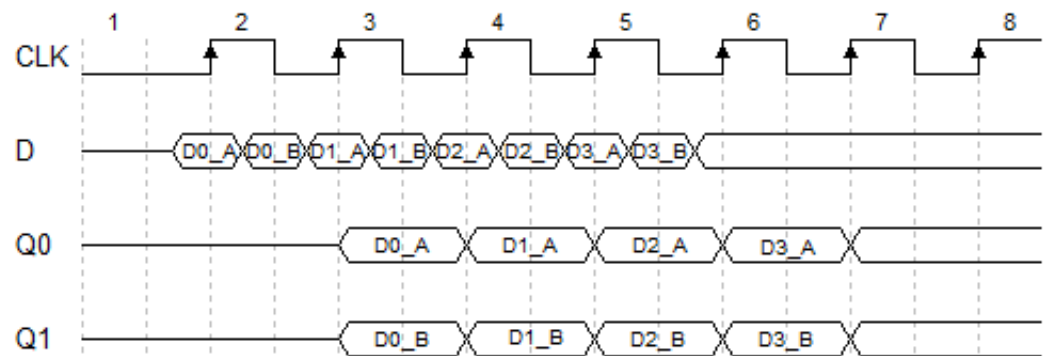


図 4-4 IDDR のタイミング図



ポート図

図 4-5 IDDR のポート図



ポートの説明

表 4-1 IDDR のポートの説明

ポート名	I/O	説明
D	入力	IDDR データ入力信号
CLK	入力	クロック入力信号
Q0, Q1	出力	IDDR データ出力

パラメータの説明

表 4-2 IDDR のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
Q0_INIT	1'b0	1'b0	Q0 出力の初期値
Q1_INIT	1'b0	1'b0	Q1 出力の初期値

接続ルール

IDDR のデータ入力 D は、IBUF から直接、または IODELAY モジュールを介して出力 DO から取得できます。

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、[5 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：

```
IDDR uut(
    .Q0(Q0),
    .Q1(Q1),
    .D(D),
    .CLK(CLK)
);
defparam uut.Q0_INIT = 1'b0;
defparam uut.Q1_INIT = 1'b0;
```

VHDL でのインスタンス化：

```
COMPONENT IDDR
    GENERIC (Q0_INIT:bit:= '0';
             Q1_INIT:bit:= '0'
    );
    PORT(
        Q0:OUT std_logic;
        Q1:OUT std_logic;
        D:IN std_logic;
        CLK:IN std_logic
    );
END COMPONENT;
uut:IDDR
    GENERIC MAP (Q0_INIT=>'0',
```

```
Q1_INIT=>'0'

)
PORT MAP (
    Q0=>Q0,
    Q1=>Q1,
    D=>D,
    CLK=>CLK
);
```

4.2.2 IDDRC

プリミティブの紹介

IDDRC(Dual Data Rate Input with Asynchronous Clear)は IDDR に比べて、非同期リセット機能をさらに備えています。

機能の説明

IDDRC モードでは、出力データは同じクロックエッジで FPGA ロジックに提供されます。

ポート図

図 4-6 IDDRC のポート図



ポートの説明

表 4-3 IDDRC のポートの説明

ポート名	I/O	説明
D	入力	IDDRC データ入力信号
CLK	入力	クロック入力信号
CLEAR	入力	非同期クリア入力、アクティブ High
Q0, Q1	出力	IDDRC データ出力

パラメータの説明

表 4-4 IDDRC のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
Q0_INIT	1'b0	1'b0	Q0 出力の初期値
Q1_INIT	1'b0	1'b0	Q1 出力の初期値

接続ルール

IDDRC のデータ入力 D は、IBUF から直接、または IODELAY モジュールを介して出力 DO から取得できます。

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、[5 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 :

```
IDDRC uut(
    .Q0(Q0),
    .Q1(Q1),
    .D(D),
    .CLK(CLK),
    .CLEAR(CLEAR)
);
defparam uut.Q0_INIT = 1'b0;
defparam uut.Q1_INIT = 1'b0;
```

VHDL でのインスタンス化 :

```
COMPONENT IDDRC
    GENERIC (Q0_INIT:bit:= '0';
             Q1_INIT:bit:= '0'
    );
    PORT(
        Q0:OUT std_logic;
        Q1:OUT std_logic;
        D:IN std_logic;
        CLEAR:IN std_logic;
        CLK:IN std_logic
    );
END COMPONENT;
uut:IDDRC
    GENERIC MAP (Q0_INIT=>'0',
                 Q1_INIT=>'0'
    )
    PORT MAP (
```

```

Q0=>Q0,
Q1=>Q1,
D=>D,
CLEAR=>CLEAR,
CLK=>CLK
);

```

4.2.3 IDES4

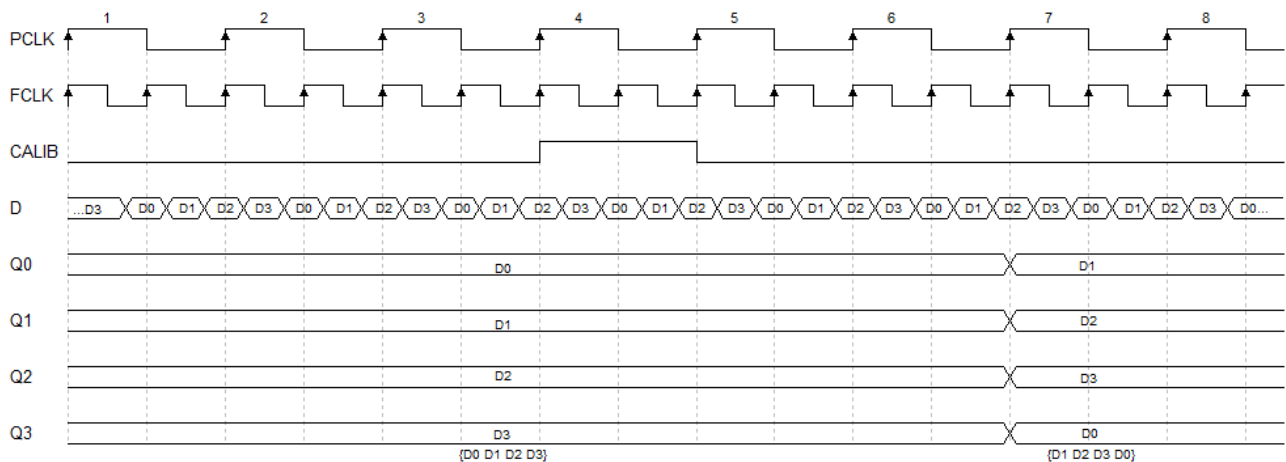
プリミティブの紹介

IDES4(1 to 4 Deserializer)は 1 ビットのシリアル入力、4 ビットの平行出力のデシリアライザです。

機能の説明

IDES4 モードでは、データは 1 : 4 デシリアライズされ、出力データは同じクロックエッジで FPGA ロジックに提供されます。**CALIB** によって出力シーケンスを調整することをサポートします。データはパルスごとに 1 ビットシフトされ、4 回シフトすると、データ出力はシフト前のデータと同じになります。**CALIB** のタイミングの例を図 4-7 に示します。

図 4-7 CALIB のタイミングの例



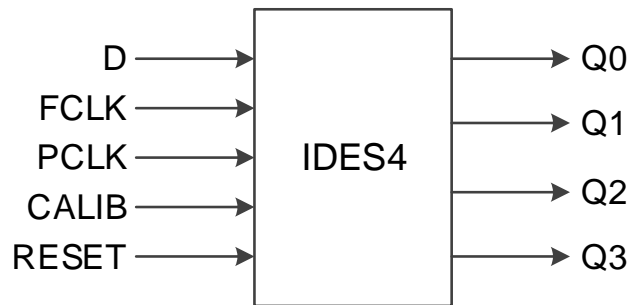
注記：

この例の **CALIB** 信号のパルス幅とタイミングは参照用であり、必要に応じて調整できます。パルス幅は T_{PCLK} 以上である必要があります。

PCLK は通常、FCLK 分周によって得られます： $f_{PCLK} = 1/2 f_{FCLK}$ 。

ポート図

図 4-8 IDES4 のポート図



ポートの説明

表 4-5 IDES4 のポートの説明

ポート名	I/O	説明
D	入力	IDES4 データ入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
CALIB	入力	出力シーケンスの調整に使用、アクティブ High
RESET	入力	非同期リセット入力、アクティブ High
Q3~Q0	出力	IDES4 データ出力信号

パラメータの説明

表 4-6 IDES4 のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

IDES4 のデータ入力 D は、IBUF から直接、または IODELAY モジュールを介して出力 DO から取得できます。

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、[5 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：

```
IDES4 uut(
```

```

        .Q0(Q0),
        .Q1(Q1),
        .Q2(Q2),
        .Q3(Q3),
        .D(D),
        .FCLK(FCLK),
        .PCLK(PCLK),
        .CALIB(CALIB),
        .RESET(RESET)
    );
    defparam uut.GSREN="false";
    defparam uut.LSREN ="true";
VHDL でのインスタンス化 :
COMPONENT IDES4
    GENERIC (GSREN:string:="false";
             LSREN:string:="true"
    );
    PORT(
        Q0:OUT std_logic;
        Q1:OUT std_logic;
        Q2:OUT std_logic;
        Q3:OUT std_logic;
        D:IN std_logic;
        FCLK:IN std_logic;
        PCLK:IN std_logic;
        CALIB:IN std_logic;
        RESET:IN std_logic
    );
END COMPONENT;
uut:IDES4
    GENERIC MAP (GSREN=>"false",
                 LSREN=>"true"
    )
    PORT MAP (
        Q0=>Q0,

```

```

Q1=>Q1,
Q2=>Q2,
Q3=>Q3,
D=>D,
FCLK=>FCLK,
PCLK=>PCLK,
CALIB=>CALIB,
RESET=>RESET
);

```

4.2.4 IDES8

プリミティブの紹介

IDES8(1 to 8 Deserializer)は 1 ビットのシリアル入力、8 ビットの平行出力のデシリアライザです。

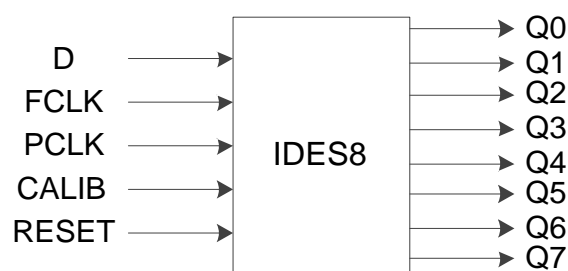
機能の説明

IDES8 モードでは、データは 1 : 8 デシリアライズされ、出力データは同じクロックエッジで FPGA ロジックに提供されます。CALIB によって出力シーケンスを調整することをサポートします。データはパルスごとに 1 ビットシフトされ、8 回シフトすると、データ出力はシフト前のデータと同じになります。

PCLK は通常、FCLK 分周によって得られます： $f_{PCLK} = 1/4 f_{FCLK}$ 。

ポート図

図 4-9 IDES8 のポート図



ポートの説明

表 4-7 IDES8 のポートの説明

ポート名	I/O	説明
D	入力	IDES8 データ入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
CALIB	入力	出力シーケンスの調整に使用、アクティブ

ポート名	I/O	説明
		High
RESET	入力	非同期リセット入力、アクティブ High
Q7~Q0	出力	IDES8 データ出力信号

パラメータの説明

表 4-8 IDES8 のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

IDES8 のデータ入力 D は、IBUF から直接取得するか、IODELAY モジュールを介してその出力 DO から取得することができます。

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、[5 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：

```
IDES8 uut(
    .Q0(Q0),
    .Q1(Q1),
    .Q2(Q2),
    .Q3(Q3),
    .Q4(Q4),
    .Q5(Q5),
    .Q6(Q6),
    .Q7(Q7),
    .D(D),
    .FCLK(FCLK),
    .PCLK(PCLK),
    .CALIB(CALIB),
    .RESET(RESET)
);
```

```
defparam uut.GSREN="false";
```

```
defparam uut.LSREN ="true";
```

VHDL でのインスタンス化 :

```
COMPONENT IDES8
```

```
    GENERIC (GSREN:string:="false";
```

```
              LSREN:string:="true"
```

```
    );
```

```
    PORT(
```

```
        Q0:OUT std_logic;
```

```
        Q1:OUT std_logic;
```

```
        Q2:OUT std_logic;
```

```
        Q3:OUT std_logic;
```

```
        Q4:OUT std_logic;
```

```
        Q5:OUT std_logic;
```

```
        Q6:OUT std_logic;
```

```
        Q7:OUT std_logic;
```

```
        D:IN std_logic;
```

```
        FCLK:IN std_logic;
```

```
        PCLK:IN std_logic;
```

```
        CALIB:IN std_logic;
```

```
        RESET:IN std_logic
```

```
    );
```

```
END COMPONENT;
```

```
uut:IDES8
```

```
    GENERIC MAP (GSREN=>"false",
```

```
                  LSREN=>"true"
```

```
    )
```

```
    PORT MAP (
```

```
        Q0=>Q0,
```

```
        Q1=>Q1,
```

```
        Q2=>Q2,
```

```
        Q3=>Q3,
```

```
        Q4=>Q4,
```

```
        Q5=>Q5,
```

```
        Q6=>Q6,
```

```

Q7=>Q7,
D=>D,
FCLK=>FCLK,
PCLK=>PCLK,
CALIB=>CALIB,
RESET=>RESET

```

```
);
```

4.2.5 IDDES10

プリミティブの紹介

IDDES10(1 to 10 Deserializer)は 1 ビットのシリアル入力、10 ビットのパラレル出力のデシリアライザです。

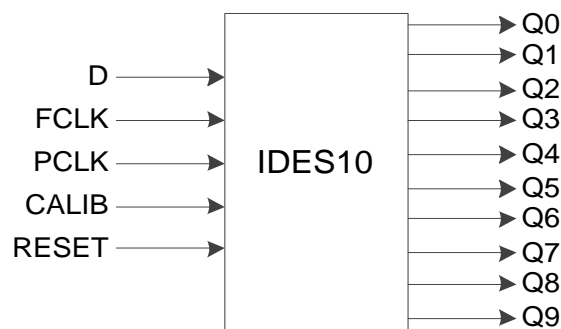
機能の説明

IDDES10 モードでは、データは 1 : 10 デシリアライズされ、出力データは同じクロックエッジで **FPGA** ロジックに提供されます。**CALIB** によって出力シーケンスを調整することをサポートします。データはパルスごとに 1 ビットシフトされ、10 回シフトすると、データ出力はシフト前のデータと同じになります。

PCLK は通常、**FCLK** 分周によって得られます： $f_{PCLK} = 1/5 f_{FCLK}$ 。

ポート図

図 4-10 IDDES10 のポート図



ポートの説明

表 4-9 IDDES10 のポートの説明

ポート名	I/O	説明
D	入力	IDDES10 データ入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
CALIB	入力	出力シーケンスの調整に使用、アクティブ High

ポート名	I/O	説明
RESET	入力	非同期リセット入力、アクティブ High
Q9~Q0	出力	IDES10 データ出力信号

パラメータの説明

表 4-10 IDES10 のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

IDES10 のデータ入力 D は、IBUF から直接取得するか、IODELAY モジュールを介してその出力 DO から取得することができます。

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、[5 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 :

```
IDES10 uut(
    .Q0(Q0),
    .Q1(Q1),
    .Q2(Q2),
    .Q3(Q3),
    .Q4(Q4),
    .Q5(Q5),
    .Q6(Q6),
    .Q7(Q7),
    .Q8(Q8),
    .Q9(Q9),
    .D(D),
    .FCLK(FCLK),
    .PCLK(PCLK),
    .CALIB(CALIB),
    .RESET(RESET)
```

```

);
defparam uut.GSREN="false";
defparam uut.LSREN="true";
VHDL でのインスタンス化 :
COMPONENT IDES10
    GENERIC (GSREN:string:="false";
             LSREN:string:="true"
    );
    PORT(
        Q0:OUT std_logic;
        Q1:OUT std_logic;
        Q2:OUT std_logic;
        Q3:OUT std_logic;
        Q4:OUT std_logic;
        Q5:OUT std_logic;
        Q6:OUT std_logic;
        Q7:OUT std_logic;
        Q8:OUT std_logic;
        Q9:OUT std_logic;
        D:IN std_logic;
        FCLK:IN std_logic;
        PCLK:IN std_logic;
        CALIB:IN std_logic;
        RESET:IN std_logic
    );
END COMPONENT;
uut:IDES10
    GENERIC MAP (GSREN=>"false",
                 LSREN=>"true"
    )
    PORT MAP (
        Q0=>Q0,
        Q1=>Q1,
        Q2=>Q2,
        Q3=>Q3,

```

```

Q4=>Q4,
Q5=>Q5,
Q6=>Q6,
Q7=>Q7,
Q8=>Q8,
Q9=>Q9,
D=>D,
FCLK=>FCLK,
PCLK=>PCLK,
CALIB=>CALIB,
RESET=>RESET

```

```

);

```

4.2.6 IVIDEO

プリミティブの紹介

IVIDEO(1 to 7 Deserializer)は 1 ビットのシリアル入力、7 ビットのパラレル出力のデシリアライザです。

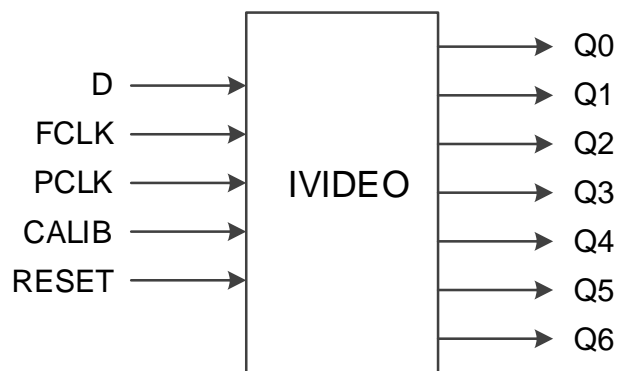
機能の説明

IVIDEO モードでは、データは 1 : 7 デシリアライズされ、出力データは同じクロックエッジで FPGA ロジックに提供されます。CALIB によって出力シーケンスを調整することをサポートします。データはパルスごとに 2 ビットシフトされ、7 回シフトすると、データ出力はシフト前のデータと同じになります。

PCLK は通常、FCLK 分周によって得られます： $f_{PCLK} = 1/3.5 f_{FCLK}$ 。

ポート図

図 4-11 IVIDEO のポート図



ポートの説明

表 4-11 IVIDEO のポートの説明

ポート名	I/O	説明
D	入力	IVIDEO データ入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
CALIB	入力	出力シーケンスの調整に使用、アクティブ High
RESET	入力	非同期リセット入力、アクティブ High
Q6~Q0	出力	IVIDEO データ出力

パラメータの説明

表 4-12 IVIDEO のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

IVIDEO のデータ入力 D は、IBUF から直接、または IODELAY モジュールを介して出力 DO から取得できます。

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、[5 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：

```
IVIDEO uut(
    .Q0(Q0),
    .Q1(Q1),
    .Q2(Q2),
    .Q3(Q3),
    .Q4(Q4),
    .Q5(Q5),
    .Q6(Q6),
    .D(D),
    .FCLK(FCLK),
    .PCLK(PCLK),
```

```

        .CALIB(CALIB),
        .RESET(RESET)
    );
    defparam uut.GSREN="false";
    defparam uut.LSREN ="true";
VHDL でのインスタンス化 :
    COMPONENT IVIDEO
        GENERIC (GSREN:string:="false";
                 LSREN:string:="true"

        );
        PORT(
            Q0:OUT std_logic;
            Q1:OUT std_logic;
            Q2:OUT std_logic;
            Q3:OUT std_logic;
            Q4:OUT std_logic;
            Q5:OUT std_logic;
            Q6:OUT std_logic;
            D:IN std_logic;
            FCLK:IN std_logic;
            PCLK:IN std_logic;
            CALIB:IN std_logic;
            RESET:IN std_logic

        );
    END COMPONENT;
    uut:IVIDEO
        GENERIC MAP (GSREN=>"false",
                     LSREN=>"true"

        )
        PORT MAP (
            Q0=>Q0,
            Q1=>Q1,
            Q2=>Q2,
            Q3=>Q3,
            Q4=>Q4,

```



```

Q5=>Q5,
Q6=>Q6,
D=>D,
FCLK=>FCLK,
PCLK=>PCLK,
CALIB=>CALIB,
RESET=>RESET
);

```

4.2.7 IDES16

プリミティブの紹介

IDES16(1 to 16 Deserializer)は 1 ビットのシリアル入力、16 ビットのパラレル出力のデシリアライザです。

サポートされるデバイス

表 4-13 IDES16 対応デバイス

ファミリー	シリーズ	デバイス
LittleBee® ファミリー	GW1N	GW1N-1S, GW1N-9, GW1N-9C, GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B
	GW1NR	GW1NR-9, GW1NR-9C, GW1NR-2, GW1NR-2B
	GW1NS	GW1NS-4, GW1NS-4C
	GW1NSER	GW1NSER-4C
	GW1NSR	GW1NSR-4, GW1NSR-4C

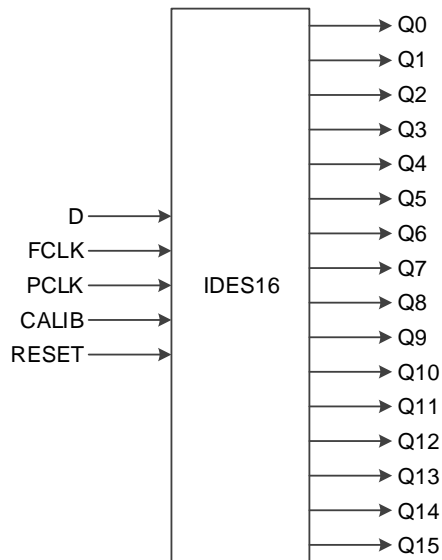
機能の説明

IDES16 モードでは、データは 1 : 16 デシリアライズされ、出力データは同じクロックエッジで **FPGA** ロジックに提供されます。**CALIB** によって出力シーケンスを調整することをサポートします。データはパルスごとに 1 ビットシフトされ、16 回シフトすると、データ出力はシフト前のデータと同じになります。

PCLK は通常、**FCLK** 分周によって得られます： $f_{PCLK} = 1/8 f_{FCLK}$ 。

ポート図

図 4-12 IDDES16 のポート図



ポートの説明

表 4-14 IDDES16 のポートの説明

ポート名	I/O	説明
D	入力	IDDES16 データ入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
CALIB	入力	出力シーケンスの調整に使用、アクティブ High
RESET	入力	非同期リセット入力、アクティブ High
Q15~Q0	出力	IDDES16 データ出力信号

パラメータの説明

表 4-15 IDDES16 のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

IDDES16 のデータ入力 D は、IBUF から直接取得するか、IODELAY モジュールを介してその出力 DO から取得することができます。

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、[5 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 :

```
IDES16 uut(  
    .Q0(Q0),  
    .Q1(Q1),  
    .Q2(Q2),  
    .Q3(Q3),  
    .Q4(Q4),  
    .Q5(Q5),  
    .Q6(Q6),  
    .Q7(Q7),  
    .Q8(Q8),  
    .Q9(Q9),  
    .Q10(Q10),  
    .Q11(Q11),  
    .Q12(Q12),  
    .Q13(Q13),  
    .Q14(Q14),  
    .Q15(Q15),  
    .D(D),  
    .FCLK(FCLK),  
    .PCLK(PCLK),  
    .CALIB(CALIB),  
    .RESET(RESET)  
);  
defparam uut.GSREN="false";  
defparam uut.LSREN ="true";
```

VHDL でのインスタンス化 :

```
COMPONENT IDES16  
    GENERIC (GSREN:string:="false";  
             LSREN:string:="true"  
    );  
PORT(  
    
```

```
Q0:OUT std_logic;
Q1:OUT std_logic;
Q2:OUT std_logic;
Q3:OUT std_logic;
Q4:OUT std_logic;
Q5:OUT std_logic;
Q6:OUT std_logic;
Q7:OUT std_logic;
Q8:OUT std_logic;
Q9:OUT std_logic;
Q10:OUT std_logic;
Q11:OUT std_logic;
Q12:OUT std_logic;
Q13:OUT std_logic;
Q14:OUT std_logic;
Q15:OUT std_logic;
D:IN std_logic;
FCLK:IN std_logic;
PCLK:IN std_logic;
CALIB:IN std_logic;
RESET:IN std_logic

);
END COMPONENT;
uut:IDES16
    GENERIC MAP (GSREN=>"false",
                  LSREN=>"true"
    )
    PORT MAP (
        Q0=>Q0,
        Q1=>Q1,
        Q2=>Q2,
        Q3=>Q3,
        Q4=>Q4,
        Q5=>Q5,
        Q6=>Q6,
```

```

Q7=>Q7,
Q8=>Q8,
Q9=>Q9,
Q10=>Q10,
Q11=>Q11,
Q12=>Q12,
Q13=>Q13,
Q14=>Q14,
Q15=>Q15,
D=>D,
FCLK=>FCLK,
PCLK=>PCLK,
CALIB=>CALIB,
RESET=>RESET

```

```
);
```

4.2.8 IDDR_MEM

プリミティブの紹介

IDDR_MEM(Dual Data Rate Input with Memory)は、memory 付きのダブルデータレートの実現します。

サポートされるデバイス

表 4-16 IDDR_MEM 対応デバイス

ファミリー	シリーズ	デバイス
Arora ファミリー	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C

機能の説明

IDDR_MEM 出力データは同じクロックエッジで FPGA ロジックに提供されます。IDDR_MEM には DQS が必要です。ICLK は DQS の出力信号 DQSR90 に接続され、ICLK のクロックエッジに従ってデータを IDDR_MEM に入力します。WADDR[2:0]は DQS の出力信号 WPOINT に接続されます。RADDR[2:0]は DQS の出力信号 RPOINT に接続されます。

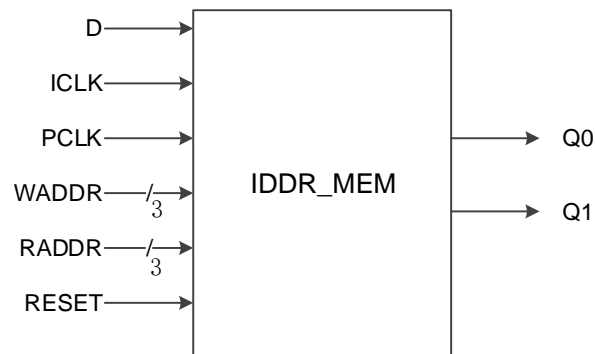
PCLK と ICLK の周波数関係は： $f_{PCLK} = f_{ICLK}$ 。

PCLK と ICLK の間には一定の位相関係があり、位相関係は DQS の

DLLSTEP 値により決定できます。

ポート図

図 4-13 IDDR_MEM のポート図



ポートの説明

表 4-17 IDDR_MEM のポートの説明

ポート名	I/O	説明
D	入力	IDDR_MEM データ入力信号
ICLK	入力	DQS モジュールの DQSR90 からのクロック入力信号
PCLK	入力	プライマリクロック入力信号
WADDR[2:0]	入力	DQS モジュールの WPOINT からの書き込みアドレス
RADDR[2:0]	入力	DQS モジュールの RPOINT からの読み出しアドレス
RESET	入力	非同期リセット入力、アクティブ High
Q1~Q0	出力	IDDR_MEM データ出力

パラメータの説明

表 4-18 IDDR_MEM のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

- IDDR_MEM のデータ入力 D は、IBUF から直接取得するか、IODELAY モジュールを介してその出力 DO から取得することができます。

- ICLK は DQS モジュールの DQSR90 からのものである必要があります。
- WADDR[2:0] は DQS モジュールの WPOINT からのものである必要があります。
- RADDR[2:0] は DQS モジュールの RPOINT からのものである必要があります。

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
IDDR_MEM iddr_mem_inst(  
    .Q0(q0),  
    .Q1(q1),  
    .D(d),  
    .ICLK (iclk),  
    .PCLK(pclk),  
    .WADDR(waddr[2:0]),  
    .RADDR(raddr[2:0]),  
    .RESET(reset)  
);
```

```
defparam uut.GSREN="false";
```

```
defparam uut.LSREN ="true";
```

VHDL でのインスタンス化 :

```
COMPONENT IDDR_MEM  
    GENERIC (GSREN:string:="false";  
             LSREN:string:="true"  
    );  
PORT(  
    Q0:OUT std_logic;  
    Q1:OUT std_logic;  
    D:IN std_logic;  
    ICLK:IN std_logic;  
    PCLK:IN std_logic;  
    WADDR:IN std_logic_vector(2 downto 0);  
    RADDR:IN std_logic_vector(2 downto 0);  
    RESET:IN std_logic  
);  
END COMPONENT;
```

```

uut:IDDR_MEM
    GENERIC MAP (GSREN=>"false",
                  LSREN=>"true"
    )
    PORT MAP (
        Q0=>q0,
        Q1=>q1,
        D=>d,
        ICLK=>iclk,
        PCLK=>pclk,
        WADDR=>waddr,
        RADDR=>raddr,
        RESET=>reset
    );

```

4.2.9 IDES4_MEM

プリミティブの紹介

IDES4_MEM(1 to 4 Deserializer with Memory) は、メモリ機能付きの 1:4 デシリアライザで、1 ビットのシリアル入力を 4 ビットの平行出力に変換できます。

サポートされるデバイス

表 4-19 IDES4_MEM 対応デバイス

ファミリー	シリーズ	デバイス
Arora ファミリー	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C

機能の説明

IDES4_MEM 实现 1: IDES4_MEM モードでは、データは 1 : 4 デシリアライズされ、出力データは同じクロックエッジで FPGA ロジックに提供されます。CALIB によって出力シーケンスを調整することをサポートします。データはパルスごとに 1 ビットシフトされ、4 回シフトすると、データ出力はシフト前のデータと同じになります。

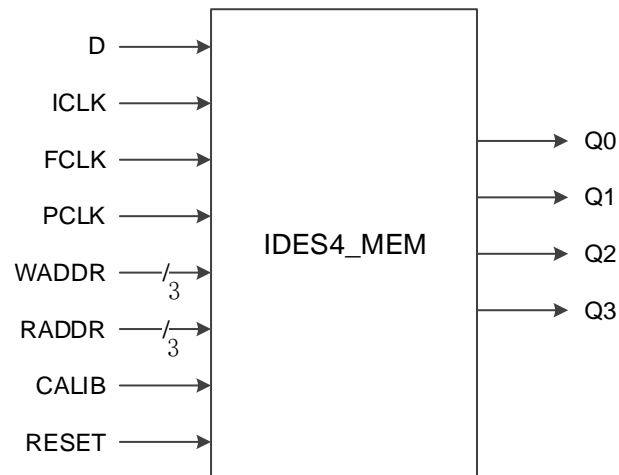
IDES4_MEM には DQS が必要です。ICLK は DQS の出力信号 DQSR90 に接続され、ICLK のクロックエッジに従ってデータを IDES4_MEM に入力します。WADDR[2:0] は DQS の出力信号 WPOINT に接続されます。RADDR[2:0] は DQS の出力信号 RPOINT に接続されます。

PCLK、FCLK、および ICLK の周波数関係は： $f_{PCLK} = 1/2 f_{FCLK} = 1/2 f_{ICLK}$ 。

FCLK と ICLK の間には一定の位相関係があり、位相関係は DQS の DLLSTEP 値により決定できます。

ポート図

図 4-14 IDES4_MEM のポート図



ポートの説明

表 4-20 IDES4_MEM のポートの説明

ポート名	I/O	説明
D	入力	IDES4_MEM データ入力信号
ICLK	入力	DQS モジュールの DQSR90 からのクロック入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
WADDR[2:0]	入力	DQS モジュールの WPOINT からの書き込みアドレス
RADDR[2:0]	入力	DQS モジュールの RPOINT からの読み出しアドレス
CALIB	入力	出力シーケンスの調整に使用、アクティブ High
RESET	入力	非同期リセット入力、アクティブ High
Q3~Q0	出力	IDES4_MEM データ出力信号

パラメータの説明

表 4-21 IDES4_MEM のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする

LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする
-------	-----------------	--------	-----------------------

接続ルール

- IDES4_MEM のデータ入力 D は、IBUF から直接取得するか、IODELAY モジュールを介してその出力 DO から取得することができます。
- ICLK は DQS モジュールの DQSR90 からのものである必要があります。
- WADDR[2:0]は DQS モジュールの WPOINT からのものである必要があります。
- RADDR[2:0]は DQS モジュールの RPOINT からのものである必要があります。

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```

IDES4_MEM ides4_mem_inst(
    .Q0(q0),
    .Q1(q1),
    .Q2(q2),
    .Q3(q3),
    .D(d),
    .ICLK(iclk),
    .FCLK(fclk),
    .PCLK(pclk),
    .WADDR(waddr[2:0]),
    .RADDR(raddr[2:0]),
    .CALIB(calib),
    .RESET(reset)
);
defparam uut.GSREN="false";
defparam uut.LSREN ="true";

```

VHDL でのインスタンス化 :

```

COMPONENT IDES4_MEM
    GENERIC (GSREN:string:="false";
             LSREN:string:="true"
    );
PORT(

```

```

        Q0:OUT std_logic;
        Q1:OUT std_logic;
        Q2:OUT std_logic;
        Q3:OUT std_logic;
        D:IN std_logic;
        ICLK:IN std_logic;
        FCLK:IN std_logic;
        PCLK:IN std_logic;
        WADDR:IN std_logic_vector(2 downto 0);
        RADDR:IN std_logic_vector(2 downto 0);
        CALIB:IN std_logic;
        RESET:IN std_logic

    );
END COMPONENT;
uut:IDES4_MEM
    GENERIC MAP (GSREN=>"false",
                  LSREN=>"true"
    )
    PORT MAP (
        Q0=>q0,
        Q1=>q1,
        Q2=>q2,
        Q3=>q3,
        D=>d,
        ICLK=>iclk,
        FCLK=>fclk,
        PCLK=>pclk,
        WADDR=>waddr,
        RADDR=>raddr,
        CALIB=>calib,
        RESET=>reset
    );

```

4.2.10 IDES8_MEM

プリミティブの紹介

IDES8_MEM(1 to 8 Deserializer with Memory) は、メモリ機能付きの 1:8

デシリアライザで、1 ビットのシリアル入力を 8 ビットの平行出力に変換できます。

サポートされるデバイス

表 4-22 IDES8_MEM 対応デバイス

ファミリー	シリーズ	デバイス
Arora ファミリー	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C

機能の説明

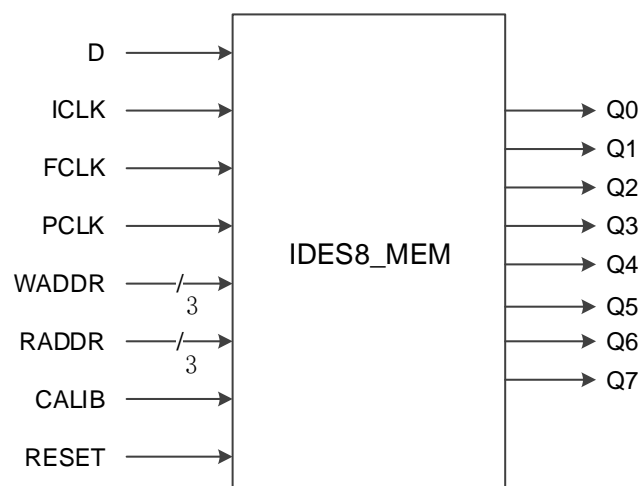
IDES8_MEM 実現 1: IDES8_MEM モードでは、データは 1 : 8 デシリアライズされ、出力データは同じクロックエッジで **FPGA** ロジックに提供されます。**CALIB** によって出力シーケンスを調整することをサポートします。データはパルスごとに 1 ビットシフトされ、8 回シフトすると、データ出力はシフト前のデータと同じになります。IDES8_MEM には **DQS** が必要です。**ICLK** は **DQS** の出力信号 **DQSR90** に接続され、**ICLK** のクロックエッジに従ってデータを IDES8_MEM に入力します。**WADDR[2:0]** は **DQS** の出力信号 **WPOINT** に接続されます。**RADDR[2:0]** は **DQS** の出力信号 **RPOINT** に接続されます。

PCLK、**FCLK**、および **TCLK** の周波数関係は： $f_{PCLK} = 1/4 f_{FCLK} = 1/4 f_{ICLK}$ 。

FCLK と **ICLK** の間には一定の位相関係があり、位相関係は **DQS** の **DLLSTEP** 値により決定できます。

ポート図

図 4-15 IDES8_MEM のポート図



ポートの説明

表 4-23 IDES8_MEM のポートの説明

ポート名	I/O	説明
D	入力	IDES8_MEM データ入力信号
ICLK	入力	DQS モジュールの DQSR90 からのクロック入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
WADDR[2:0]	入力	DQS モジュールの WPOINT からの書き込みアドレス
RADDR[2:0]	入力	DQS モジュールの RPOINT からの読み出しアドレス
CALIB	入力	出力シーケンスの調整に使用、アクティブ High
RESET	入力	非同期リセット入力、アクティブ High
Q7~Q0	出力	IDES8_MEM データ出力信号

パラメータの説明

表 4-24 IDES8_MEM のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

- IDES8_MEM のデータ入力 D は、IBUF から直接、または IODELAY モジュールを介して出力 DO から取得できます。
- ICLK は DQS モジュールの DQSR90 からのものである必要があります。
- WADDR[2:0] は DQS モジュールの WPOINT からのものである必要があります。
- RADDR[2:0] は DQS モジュールの RPOINT からのものである必要があります。

プリミティブのインスタンス化

Verilog でのインスタンス化：

```

IDES8_MEM ides8_mem_inst(
    .Q0(q0),
    .Q1(q1),
    .Q2(q2),

```

```
.Q3(q3),  
.Q4(q4),  
.Q5(q5),  
.Q6(q6),  
.Q7(q7),  
.D(d),  
.ICLK(iclk),  
.FCLK(fclk),  
.PCLK(pclk),  
.WADDR(waddr[2:0]),  
.RADDR(raddr[2:0]),  
.CALIB(calib),  
.RESET(reset)  
);
```

```
defparam uut.GSREN="false";
```

```
defparam uut.LSREN ="true";
```

VHDL でのインスタンス化 :

```
COMPONENT IDES8_MEM
```

```
    GENERIC (GSREN:string:="false";
```

```
             LSREN:string:="true"
```

```
    );
```

```
    PORT(  
        Q0:OUT std_logic;
```

```
        Q1:OUT std_logic;
```

```
        Q2:OUT std_logic;
```

```
        Q3:OUT std_logic;
```

```
        Q4:OUT std_logic;
```

```
        Q5:OUT std_logic;
```

```
        Q6:OUT std_logic;
```

```
        Q7:OUT std_logic;
```

```
        D:IN std_logic;
```

```
        ICLK:IN std_logic;
```

```
        FCLK:IN std_logic;
```

```
        PCLK:IN std_logic;
```

```
        WADDR:IN std_logic_vector(2 downto 0);
```

```

        RADDR:IN std_logic_vector(2 downto 0);
        CALIB:IN std_logic;
        RESET:IN std_logic

    );
END COMPONENT;
uut:IDES8_MEM
    GENERIC MAP (GSREN=>"false",
                  LSREN=>"true"
    )
    PORT MAP (
        Q0=>q0,
        Q1=>q1,
        Q2=>q2,
        Q3=>q3,
        Q4=>q4,
        Q5=>q5,
        Q6=>q6,
        Q7=>q7,
        D=>d,
        ICLK=>iclk,
        FCLK=>fclk,
        PCLK=>pclk,
        WADDR=>waddr,
        RADDR=>raddr,
        CALIB=>calib,
        RESET=>reset
    );

```

4.3 DDR モードの出力ロジック

4.3.1 ODDR

プリミティブの紹介

ODDR(Dual Data Rate Output)は、ダブルデータレートの実現します。

機能の説明

ODDR モードは、FPGA デバイスからダブルデータレート信号を送信す

るために使用されます。Q0 はダブルデータレートでのデータ出力で、Q1 は Q0 に接続される IOBUF/TBUF の OEN 信号に使用されます。ODDR モードのブロック図は図 4-16、タイミング図は図 4-17 に示すとおりです。

図 4-16 ODDR のブロック図

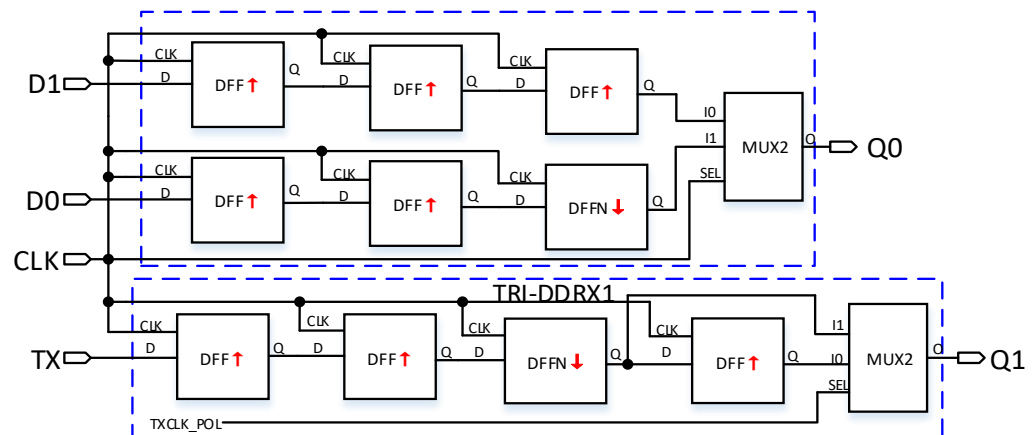
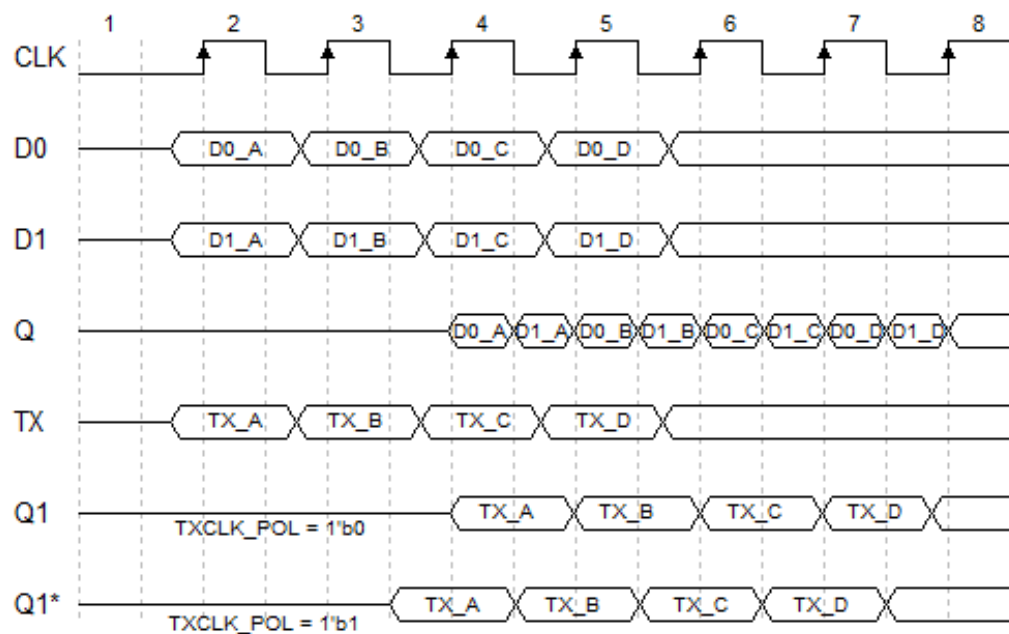


図 4-17 ODDR のタイミング図



ポート図

図 4-18 ODDR のポート図



ポートの説明

表 4-25 ODDR のポートの説明

ポート名	I/O	説明
D0, D1	入力	ODDR データ入力信号
TX	入力	TRI-DDRX1 を通じて Q1 を生成
CLK	入力	クロック入力信号
Q0	出力	ODDR データ出力
Q1	出力	ODDR トライステートイネーブル制御出力、Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにすることができます。

パラメータの説明

表 4-26 ODDR のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
TXCLK_POL	1'b0, 1'b1	1'b0	Q1 出力クロックの極性制御 1'b0:Q1 立ち上がりエッジで出力; 1'b1:Q1 立ち下がりエッジで出力
INIT	1'b0	1'b0	ODDR 出力の初期値

接続ルール

- Q0 は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。
- Q1 は、Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにする必要があります。

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、5 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：

```
ODDR uut(
    .Q0(Q0),
    .Q1(Q1),
    .D0(D0),
    .D1(D1),
    .TX(TX),
```

```

        .CLK(CLK)
    );
    defparam uut.INIT=1'b0;
    defparam uut.TXCLK_POL=1'b0;
VHDL でのインスタンス化 :
    COMPONENT ODDR
        GENERIC (CONSTANT INIT: std_logic=>'0';
                  TXCLK_POL:bit=>'0'
        );
        PORT(
            Q0:OUT std_logic;
            Q1:OUT std_logic;
            D0:IN std_logic;
            D1:IN std_logic;
            TX:IN std_logic;
            CLK:IN std_logic
        );
    END COMPONENT;
    uut:ODDR
        GENERIC MAP (INIT=>'0',
                     TXCLK_POL=>'0'
        )
        PORT MAP (
            Q0=>Q0,
            Q1=>Q1,
            D0=>D0,
            D1=>D1,
            TX=>TX,
            CLK=>CLK
        );

```

4.3.2 ODDRC

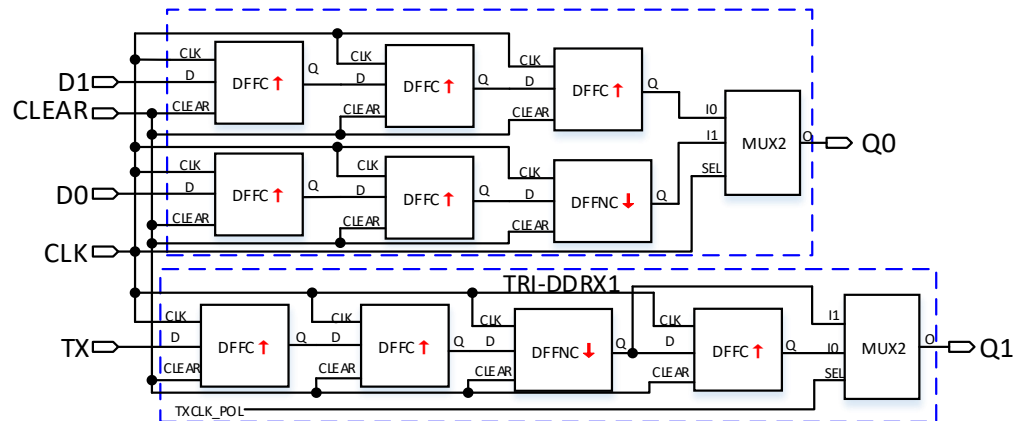
プリミティブの紹介

ODDRC(Dual Data Rate Output with Asynchronous Clear)は ODDR に比べて、非同期リセット機能をさらに備えています。

機能の説明

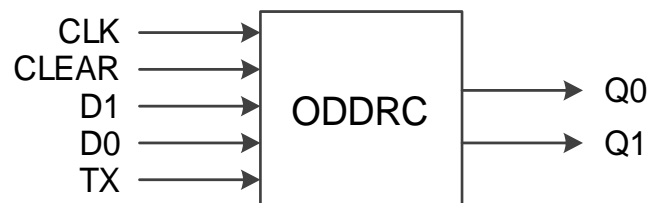
ODDRC モードは、FPGA デバイスからダブルデータレート信号を送信するために使用されます。Q0 はダブルデータレートのデータ出力で、Q1 は Q0 に接続される IOBUF/TBUF の OEN 信号に使用されます。そのブロック図を図 4-19 に示します。

図 4-19 ODDRC のブロック図



ポート図

図 4-20 ODDRC のポート図



ポートの説明

表 4-27 ODDRC のポートの説明

ポート名	I/O	説明
D0, D1	入力	ODDRC データ入力信号
TX	入力	TRI-DDRX1 を通じて Q1 を生成
CLK	入力	クロック入力信号
CLEAR	入力	非同期クリア入力、アクティブ High
Q0	出力	ODDRC データ出力
Q1	出力	ODDRC トライステートイネーブル制御出力。 Q0 に接続される IOBUF/TBUF の OEN 信号に 接続するか、フローティングのままにすることが できます。

パラメータの説明

表 4-28 ODDRC のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
TXCLK_POL	1'b0, 1'b1	1'b0	Q1 出力クロックの極性制御 1'b0:Q1 立ち上がりエッジで出力; 1'b1:Q1 立ち下がりエッジで出力
INIT	1'b0	1'b0	ODDRC 出力の初期値

接続ルール

- Q0 は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。
- Q1 は、Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにする必要があります。

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、[5 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 :

```
ODDRC uut(
    .Q0(Q0),
    .Q1(Q1),
    .D0(D0),
    .D1(D1),
    .TX(TX),
    .CLK(CLK),
    .CLEAR(CLEAR)
);
defparam uut.INIT=1'b0;
defparam uut.TXCLK_POL=1'b0;
```

VHDL でのインスタンス化 :

```
COMPONENT ODDRC
    GENERIC (CONSTANT INIT : std_logic := '0';
             TXCLK_POL : bit := '0'
    );
    PORT(
```

```

        Q0:OUT std_logic;
        Q1:OUT std_logic;
        D0:IN std_logic;
        D1:IN std_logic;
        TX:IN std_logic;
        CLK:IN std_logic;
        CLEAR:IN std_logic

    );
END COMPONENT;
uut:ODDRC
    GENERIC MAP (INIT=>'0',
                  TXCLK_POL=>'0'
    )
    PORT MAP (
        Q0=>Q0,
        Q1=>Q1,
        D0=>D0,
        D1=>D1,
        TX=>TX,
        CLK=>CLK,
        CLEAR=>CLEAR
    );

```

4.3.3 OSER4

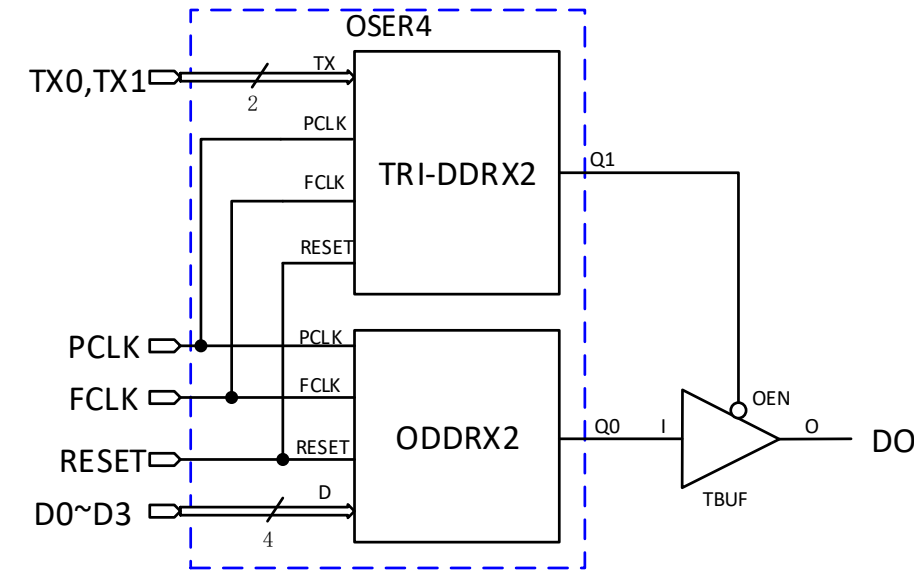
プリミティブの紹介

OSER4(4 to 1 Serializer)は 4 ビットの平行入力、1 ビットのシリアル出力のシリアライザです。

機能の説明

OSER4 モードでは、データは 4:1 シリアライズされます。Q0 は OSER4 データのシリアル出力で、Q1 は Q0 に接続される IOBUF/TBUF の OEN 信号に使用されます。TX0 と TX1 は、主に TRI-DDRX2 を介して IOBUF/TBUF の OEN 信号を生成するために使用されます。TX0 と TX1 は、データ D0 ～D3 と同期して DDR を通過できます。TX0 と TX1 は DDR を介して Q1 (IOBUF/TBUF の OEN に接続) として出力され、D0～D3 は DDR を介して Q0 (IOBUF/TBUF のデータ入力 I に接続) として出力されます。データは D0、D1、D2、D3 の順序で出力されます。そのブロック図を図 4-21 に示します。

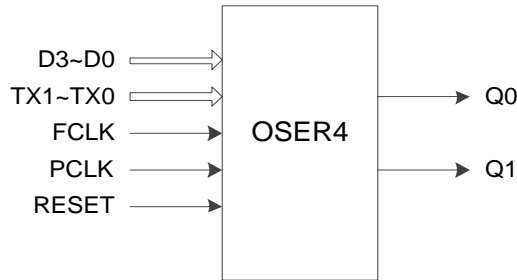
図 4-21 OSER4 のブロック図



PCLK は通常、FCLK 分周によって得られます： $f_{PCLK} = 1/2 f_{FCLK}$ 。

ポート図

図 4-22 OSER4 のポート図



ポートの説明

表 4-29 OSER4 のポートの説明

ポート名	I/O	説明
D3~D0	入力	OSER4 データ入力信号
TX1~TX0	入力	TRI-DDRX2 を通じて Q1 を生成
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
RESET	入力	非同期リセット入力、アクティブ High
Q0	出力	OSER4 データ出力信号
Q1	出力	OSER4 トライステートイネーブル制御出力。 Q0 に接続される IOBUF/TBUF の OEN 信号に 接続するか、フローティングのままにすること ができます。

パラメータの説明

表 4-30 OSER4 のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする
TXCLK_POL	1'b0, 1'b1	1'b0	Q1 出力クロックの極性制御 1'b0:立ち上がりエッジで出力 1'b1:立ち下がりエッジで出力
HWL	"false", "true"	"false"	OSER4 データ d_up0/1 タイミング関係制御 "false": d_up1 は d_up0 より 1 サイクル先です。 "true": d_up1 と d_up0 のタイミングは同じです。

接続ルール

- Q0 は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。
- Q1 は、Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにする必要があります。

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、5 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：

```
OSER4 uut(
    .Q0(Q0),
    .Q1(Q1),
    .D0(D0),
    .D1(D1),
    .D2(D2),
    .D3(D3),
    .TX0(TX0),
    .TX1(TX1),
    .PCLK(PCLK),
    .FCLK(FCLK),
```

```

        .RESET(RESET)
    );
    defparam uut.GSREN="false";
    defparam uut.LSREN ="true";
    defparam uut.HWL ="false";
    defparam uut.TXCLK_POL =1'b0;
VHDL でのインスタンス化 :
    COMPONENT OSER4
        GENERIC (GSREN:string:="false";
                  LSREN:string:="true";
                  HWL:string:="false";
                  TXCLK_POL:bit:='0'
                );
        PORT(
            Q0:OUT std_logic;
            Q1:OUT std_logic;
            D0:IN std_logic;
            D1:IN std_logic;
            D2:IN std_logic;
            D3:IN std_logic;
            TX0:IN std_logic;
            TX1:IN std_logic;
            FCLK:IN std_logic;
            PCLK:IN std_logic;
            RESET:IN std_logic
        );
    END COMPONENT;
    uut:OSER4
        GENERIC MAP (GSREN=>"false",
                     LSREN=>"true",
                     HWL=>"false",
                     TXCLK_POL=>'0'
                    )
        PORT MAP (
            Q0=>Q0,

```



```

Q1=>Q1,
D0=>D0,
D1=>D1,
D2=>D2,
D3=>D3,
TX0=>TX0,
TX1=>TX1,
FCLK=>FCLK,
PCLK=>PCLK,
RESET=>RESET

```

```
);
```

4.3.4 OSER8

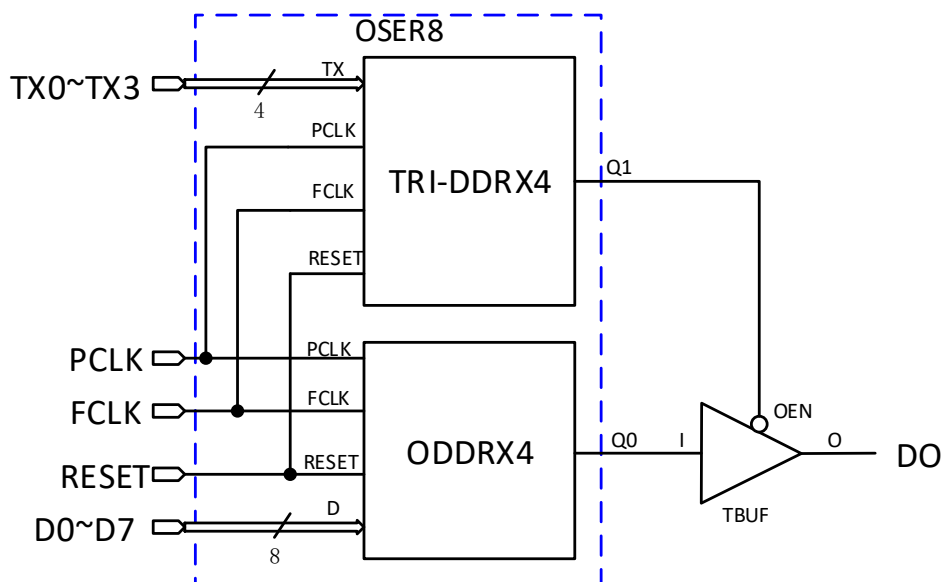
プリミティブの紹介

OSER8(8 to 1 Serializer)は 8 ビットの平行入力、1 ビットのシリアル出力のシリアライザです。

機能の説明

OSER8 モードでは、データは 8:1 シリアライズされます。Q0 は OSER8 データのシリアル出力で、Q1 は Q0 に接続される IOBUF/TBUF の OEN 信号に使用されます。そのブロック図を図 4-23 に示します。

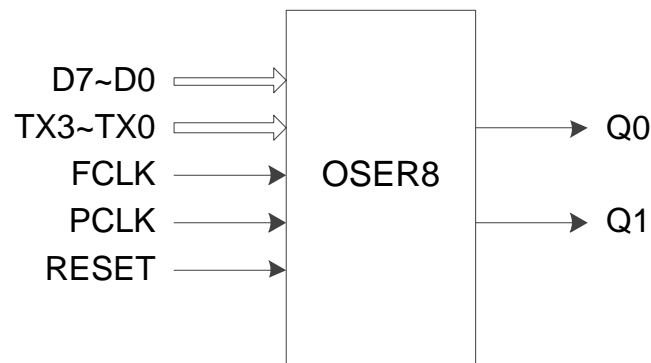
図 4-23 OSER8 のブロック図



PCLK は通常、FCLK 分周によって得られます： $f_{PCLK} = 1/4 f_{FCLK}$ 。

ポート図

図 4-24 OSER8 のポート図



ポートの説明

表 4-31 OSER8 のポートの説明

ポート名	I/O	説明
D7~D0	入力	OSER8 データ入力信号
TX3~TX0	入力	TRI-DDRX4 を通じて Q1 を生成
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
RESET	入力	非同期リセット入力、アクティブ High
Q0	出力	OSER8 データ出力信号
Q1	出力	OSER8 トライステートイネーブル制御出力。Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにすることができます。

パラメータの説明

表 4-32 OSER8 のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする
TXCLK_POL	1'b0, 1'b1	1'b0	Q1 出力クロックの極性制御 <ul style="list-style-type: none"> ● 1'b0: 立ち上がりエッジで出力 ● 1'b1: 立ち下がりエッジで出力
HWL	"false", "true"	"false"	OSER8 データ d_up0/1 タイミング関係制御 <ul style="list-style-type: none"> ● "false": d_up1 は d_up0 より 1 サイクル先です。 ● "true": d_up1 と d_up0 のタイミ

パラメータ名	値の範囲	デフォルト値	説明
			ングは同じです。

接続ルール

- Q0 は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。
- Q1 は、Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにする必要があります。

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、5 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：

```
OSER8 uut(
    .Q0(Q0),
    .Q1(Q1),
    .D0(D0),
    .D1(D1),
    .D2(D2),
    .D3(D3),
    .D4(D4),
    .D5(D5),
    .D6(D6),
    .D7(D7),
    .TX0(TX0),
    .TX1(TX1),
    .TX2(TX2),
    .TX3(TX3),
    .PCLK(PCLK),
    .FCLK(FCLK),
    .RESET(RESET)
);

defparam uut.GSREN="false";
defparam uut.LSREN ="true";
defparam uut.HWL ="false";
```

```

        defparam uut.TXCLK_POL = 1'b0;
VHDL でのインスタンス化 :
COMPONENT OSER8
    GENERIC (GSREN:string:="false";
              LSREN:string:="true";
              HWL:string:="false";
              TXCLK_POL:bit:='0'
    );
    PORT(
        Q0:OUT std_logic;
        Q1:OUT std_logic;
        D0:IN std_logic;
        D1:IN std_logic;
        D2:IN std_logic;
        D3:IN std_logic;
        D4:IN std_logic;
        D5:IN std_logic;
        D6:IN std_logic;
        D7:IN std_logic;
        TX0:IN std_logic;
        TX1:IN std_logic;
        TX2:IN std_logic;
        TX3:IN std_logic;
        FCLK:IN std_logic;
        PCLK:IN std_logic;
        RESET:IN std_logic
    );
END COMPONENT;
uut:OSER8
    GENERIC MAP (GSREN=>"false",
                  LSREN=>"true",
                  HWL=>"false",
                  TXCLK_POL=>'0'
    )
    PORT MAP (

```

```

Q0=>Q0,
Q1=>Q1,
D0=>D0,
D1=>D1,
D2=>D2,
D3=>D3,
D4=>D4,
D5=>D5,
D6=>D6,
D7=>D7,
TX0=>TX0,
TX1=>TX1,
TX2=>TX2,
TX3=>TX3,
FCLK=>FCLK,
PCLK=>PCLK,
RESET=>RESET

```

```
);
```

4.3.5 OSER10

プリミティブの紹介

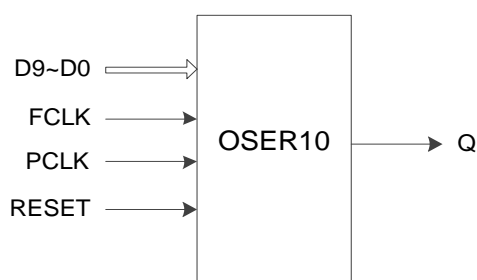
OSER10(10 to 1 Serializer)は 10 ビットの平行入力、1 ビットのシリアル出力のシリアライザです。

機能の説明

OSER10 モードでは、データは **10:1** シリアライズされます。**PCLK** は通常、**FCLK** 分周によって得られます： $f_{PCLK} = 1/5 f_{FCLK}$ 。

ポート図

図 4-25 OSER10 のポート図



ポートの説明

表 4-33 OSER10 のポートの説明

ポート名	I/O	説明
D9~D0	入力	OSER10 データ入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
RESET	入力	非同期リセット入力、アクティブ High
Q	出力	OSER10 データ出力信号

パラメータの説明

表 4-34 OSER10 のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

Q は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。

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、[5 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：

```
OSER10 uut(
    .Q(Q),
    .D0(D0),
    .D1(D1),
    .D2(D2),
    .D3(D3),
    .D4(D4),
    .D5(D5),
    .D6(D6),
    .D7(D7),
    .D8(D8),
    .D9(D9),
    .PCLK(PCLK),
```

```

        .FCLK(FCLK),
        .RESET(RESET)
    );
    defparam uut.GSREN="false";
    defparam uut.LSREN ="true";
VHDL でのインスタンス化 :
    COMPONENT OSER10
        GENERIC (GSREN:string:="false";
                  LSREN:string:="true"
        );
        PORT(
            Q:OUT std_logic;
            D0:IN std_logic;
            D1:IN std_logic;
            D2:IN std_logic;
            D3:IN std_logic;
            D4:IN std_logic;
            D5:IN std_logic;
            D6:IN std_logic;
            D7:IN std_logic;
            D8:IN std_logic;
            D9:IN std_logic;
            FCLK:IN std_logic;
            PCLK:IN std_logic;
            RESET:IN std_logic
        );
    END COMPONENT;
    uut:OSER10
        GENERIC MAP (GSREN=>"false",
                     LSREN=>"true"
        )
        PORT MAP (
            Q=>Q,
            D0=>D0,
            D1=>D1,

```

```
D2=>D2,  
D3=>D3,  
D4=>D4,  
D5=>D5,  
D6=>D6,  
D7=>D7,  
D8=>D8,  
D9=>D9,  
FCLK=>FCLK,  
PCLK=>PCLK,  
RESET=>RESET  
  
);
```

4.3.6 OVIDEO

プリミティブの紹介

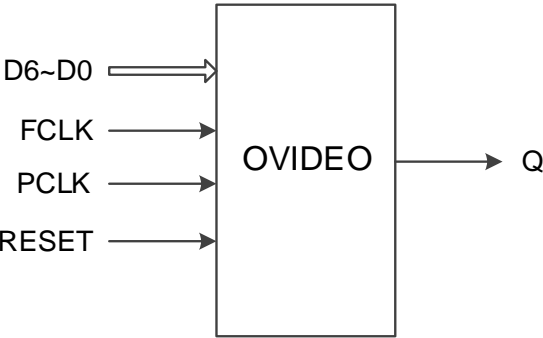
OVIDEO(7 to 1 Serializer)は 7 ビットの平行入力、1 ビットのシリアル出力のシリアライザです。

機能の説明

OVIDEO モードでは、データは 7:1 シリアライズされます。PCLK は通常、FCLK 分周によって得られます： $f_{PCLK} = 1/3.5 f_{FCLK}$ 。

ポート図

図 4-26 OVIDEO のポート図



ポートの説明

表 4-35 OVIDEO のポートの説明

ポート名	I/O	説明
D6~D0	入力	OVIDEO データ入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号

ポート名	I/O	説明
RESET	入力	非同期リセット入力、アクティブ High
Q	出力	OVIDEO データ出力

パラメータの説明

表 4-36 OVIDEO のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

Q は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。

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、[5 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 :

```
OVIDEO uut(
    .Q(Q),
    .D0(D0),
    .D1(D1),
    .D2(D2),
    .D3(D3),
    .D4(D4),
    .D5(D5),
    .D6(D6),
    .PCLK(PCLK),
    .FCLK(FCLK),
    .RESET(RESET)
);
defparam uut.GSREN="false";
defparam uut.LSREN="true";
```

VHDL でのインスタンス化 :

```
COMPONENT OVIDEO
    GENERIC (GSREN:string:="false";
             LSREN:string:="true"
    );
    PORT(
        Q:OUT std_logic;
        D0:IN std_logic;
        D1:IN std_logic;
        D2:IN std_logic;
        D3:IN std_logic;
        D4:IN std_logic;
        D5:IN std_logic;
        D6:IN std_logic;
        FCLK:IN std_logic;
        PCLK:IN std_logic;
        RESET:IN std_logic
    );
END COMPONENT;

uut:OVIDEO
    GENERIC MAP (GSREN=>"false",
                 LSREN=>"true"
    )
    PORT MAP (
        Q=>Q,
        D0=>D0,
        D1=>D1,
        D2=>D2,
        D3=>D3,
        D4=>D4,
        D5=>D5,
        D6=>D6,
        FCLK=>FCLK,
        PCLK=>PCLK,
        RESET=>RESET
    );
```

);

4.3.7 OSER16

プリミティブの紹介

OSER16(16 to 1 Serializer)は 16 ビットのパラレル入力、1 ビットのシリアル出力のシリアライザです。

サポートされるデバイス

表 4-37 OSER16 対応デバイス

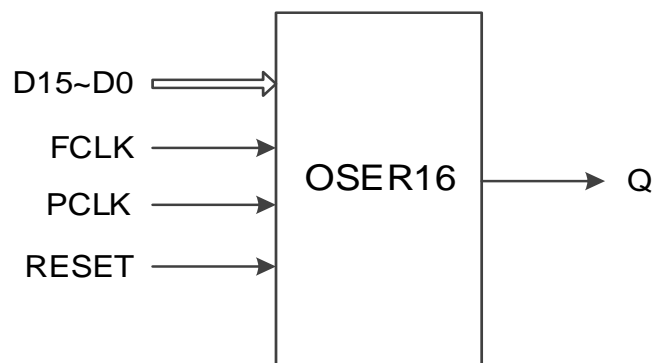
ファミリー	シリーズ	デバイス
LittleBee® ファミリー	GW1N	GW1N-1S, GW1N-9, GW1N-9C, GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B
	GW1NR	GW1NR-9, GW1NR-9C, GW1NR-2, GW1NR-2B
	GW1NS	GW1NS-4, GW1NS-4C
	GW1NSER	GW1NSER-4C
	GW1NSR	GW1NSR-4, GW1NSR-4C

機能の説明

OSER16 モードでは、データは 16:1 シリアライズされます。PCLK は通常、FCLK 分周によって得られます： $f_{PCLK} = 1/8 f_{FCLK}$ 。

ポート図

図 4-27 OSER16 のポート図



ポートの説明

表 4-38 OSER16 のポートの説明

ポート名	I/O	説明
D15~D0	入力	OSER16 データ入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
RESET	入力	非同期リセット入力、アクティブ High
Q	出力	OSER16 データ出力信号

パラメータの説明

表 4-39 OSER16 のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

接続ルール

Q は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。

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、[5 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 :

```
OSER16 uut(
    .Q(Q),
    .D0(D0),
    .D1(D1),
    .D2(D2),
    .D3(D3),
    .D4(D4),
    .D5(D5),
    .D6(D6),
    .D7(D7),
    .D8(D8),
    .D9(D9),
    .D10(D10),
    .D11(D11),
    .D12(D12),
    .D13(D13),
    .D14(D14),
    .D15(D15),
    .PCLK(PCLK),
    .FCLK(FCLK),
```

```

        .RESET(RESET)
    );
    defparam uut.GSREN="false";
    defparam uut.LSREN ="true";
VHDL でのインスタンス化 :
    COMPONENT OSER16
        GENERIC (GSREN:string:="false";
                 LSREN:string:="true"
        );
    PORT(
        Q:OUT std_logic;
        D0:IN std_logic;
        D1:IN std_logic;
        D2:IN std_logic;
        D3:IN std_logic;
        D4:IN std_logic;
        D5:IN std_logic;
        D6:IN std_logic;
        D7:IN std_logic;
        D8:IN std_logic;
        D9:IN std_logic;
        D10:IN std_logic;
        D11:IN std_logic;
        D12:IN std_logic;
        D13:IN std_logic;
        D14:IN std_logic;
        D15:IN std_logic;
        FCLK:IN std_logic;
        PCLK:IN std_logic;
        RESET:IN std_logic
    );
END COMPONENT;
uut:OSER16
    GENERIC MAP (GSREN=>"false",
                 LSREN=>"true"

```

```

)
PORT MAP (
    Q=>Q,
    D0=>D0,
    D1=>D1,
    D2=>D2,
    D3=>D3,
    D4=>D4,
    D5=>D5,
    D6=>D6,
    D7=>D7,
    D8=>D8,
    D9=>D9,
    D10=>D10,
    D11=>D11,
    D12=>D12,
    D13=>D13,
    D14=>D14,
    D15=>D15,
    FCLK=>FCLK,
    PCLK=>PCLK,
    RESET=>RESET
);

```

4.3.8 ODDR_MEM

プリミティブの紹介

ODDR_MEM(Dual Data Rate Output with Memory)は、memory 付きのダブルデータレートの実現します。

サポートされるデバイス

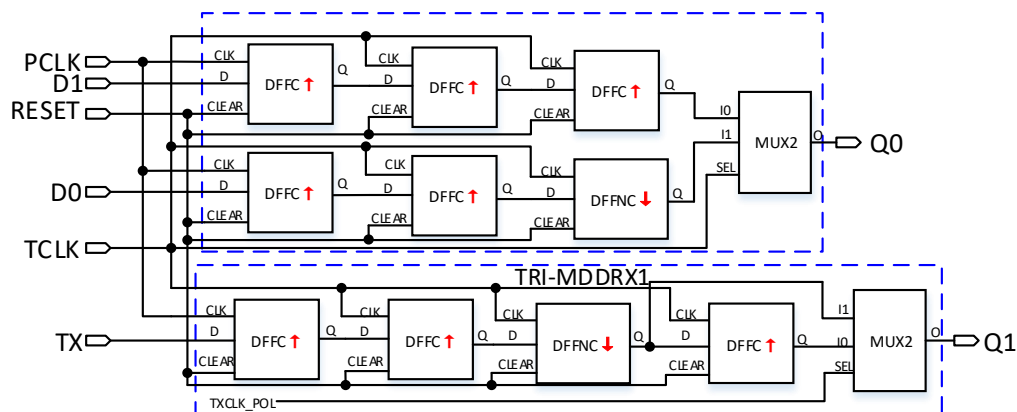
表 4-40 ODDR_MEM 対応デバイス

ファミリー	シリーズ	デバイス
Arora	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C

機能の説明

ODDR_MEM モードは、FPGA デバイスからダブルデータレート信号を転送するために使用されます。ODDR と異なり、ODDR_MEM には DQS が必要です。TCLK を DQS の出力信号 DQSW0 または DQSW270 に接続したあと、TCLK のクロックエッジに基づいてデータを ODDR_MEM から出力します。ODDR_MEM の Q0 はダブルデータレートのデータ出力で、Q1 は Q0 に接続される IOBUF/TBUF の OEN 信号に使用されます。そのブロック図を図 4-28 に示します。

図 4-28 ODDR_MEM のブロック図

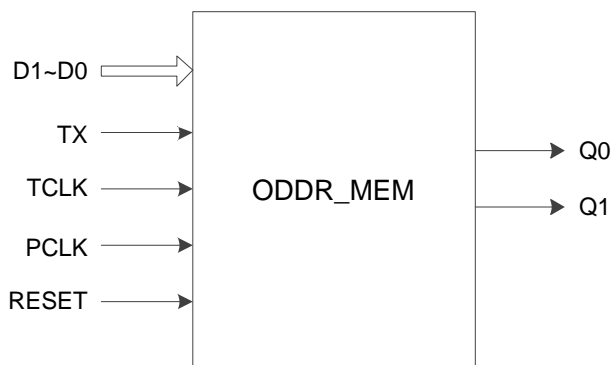


PCLK と TCLK の周波数関係は： $f_{PCLK} = f_{TCLK}$ 。

PCLK と TCLK の間には一定の位相関係があり、位相関係は DQS の DLLSTEP 値および WSTEP 値により決定できます。

ポート図

図 4-29 ODDR_MEM のポート図



ポートの説明

表 4-41 ODDR_MEM のポートの説明

ポート名	I/O	説明
D1~D0	入力	ODDR_MEM データ入力信号
TX	入力	TRI-MDDR1 を通じて Q1 を生成

ポート名	I/O	説明
TCLK	入力	DQS モジュールの DQSW0 または DQSW270 からのクロック入力信号
PCLK	入力	プライマリクロック入力信号
RESET	入力	非同期リセット入力、アクティブ High
Q0	出力	ODDR_MEM データ出力
Q1	出力	ODDR_MEM トライステートイネーブル制御出力。 Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにすることができます。

パラメータの説明

表 4-42 ODDR_MEM のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする
TXCLK_POL	1'b0, 1'b1	1'b0	Q1 出力クロックの極性制御 <ul style="list-style-type: none"> ● 1'b0: 立ち上がりエッジで出力 ● 1'b1: 立ち下がりエッジで出力
TCLK_SOURCE	"DQSW", "DQSW270"	"DQSW"	TCLK ソースの選択 <ul style="list-style-type: none"> ● "DQSW": DQS モジュールの DQSW0 から。 ● DQSW270": DQS モジュールの DQSW270 から

接続ルール

- Q0 は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。
- Q1 は、Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにする必要があります。
- TCLK は、DQS モジュールの DQSW0 または DQSW270 から取得し、対応するパラメータを構成する必要があります。

プリミティブのインスタンス化

Verilog でのインスタンス化：

```
ODDR_MEM oddr_mem_inst(
```



```

        .Q0(q0),
        .Q1(q1),
        .D0(d0),
        .D1(d1),
        .TX(tx),
        .TCLK(tclk),
        .PCLK(pclk),
        .RESET(reset)
    );
    defparam uut.GSREN="false";
    defparam uut.LSREN="true";
    defparam uut.TCLK_SOURCE="DQSW";
    defparam uut.TXCLK_POL=1'b0;

```

VHDL でのインスタンス化 :

```

COMPONENT ODDR_MEM
    GENERIC (GSREN:string:="false";
             LSREN:string:="true";
             TXCLK_POL:bit:='0';
             TCLK_SOURCE:string:="DQSW"
    );
    PORT(
        Q0:OUT std_logic;
        Q1:OUT std_logic;
        D0:IN std_logic;
        D1:IN std_logic;
        TX:IN std_logic;
        TCLK:IN std_logic;
        PCLK:IN std_logic;
        RESET:IN std_logic
    );
END COMPONENT;
uut:ODDR_MEM
    GENERIC MAP (GSREN=>"false",
                 LSREN=>"true",
                 TXCLK_POL=>'0',

```

```
                                TCLK_SOURCE=>"DQSW"
                                )
                                PORT MAP (
                                    Q0=>q0,
                                    Q1=>q1,
                                    D0=>d0,
                                    D1=>d1,
                                    TX=>tx,
                                    TCLK=>tclk,
                                    PCLK=>pclk,
                                    RESET=>reset
                                );
```

4.3.9 OSER4_MEM

プリミティブの紹介

OSER4_MEM(4 to 1 Serializer with Memory) は、メモリ機能付きの 4:1 シリアライザで、4 ビットの平行入力を 1 ビットのシリアル出力に変換できます。

サポートされるデバイス

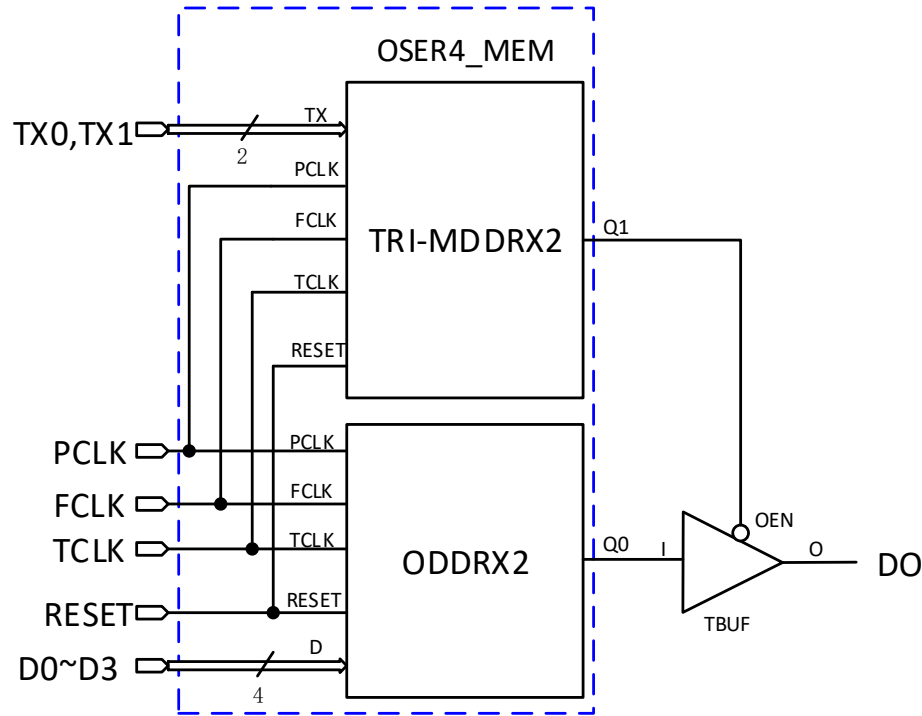
表 4-43 OSER4_MEM 対応デバイス

ファミリー	シリーズ	デバイス
Arora ファミリー	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C

機能の説明

OSER4_MEM モードでは、データは 4:1 シリアライズされます。OSER4 とは異なり、OSER4_MEM には DQS が必要です。TCLK を DQS の出力信号 DQSW0 または DQSW270 に接続したあと、TCLK のクロックエッジに基づいてデータを ODDR_MEM から出力します。OSER4_MEM の Q0 はデータのシリアル出力で、Q1 は Q0 に接続される IOBUF/TBUF の OEN 信号に使用されます。そのブロック図を図 4-30 に示します。

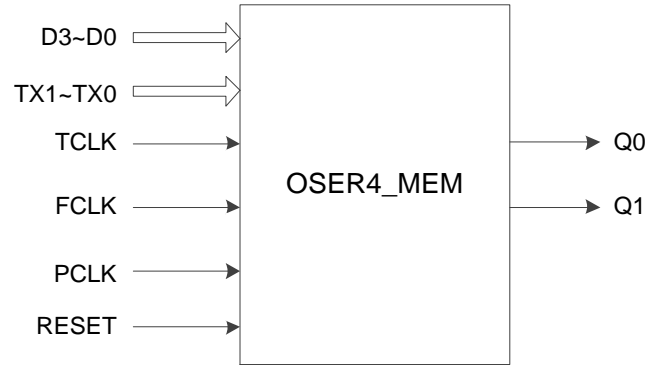
図 4-30 OSER4_MEM のブロック図



PCLK、FCLK、およびTCLKの周波数関係は： $f_{PCLK} = 1/2 f_{FCLK} = 1/2 f_{TCLK}$ 。
FCLK と TCLK の間には一定の位相関係があり、位相関係は DQS の DLLSTEP 値および WSTEP 値により決定できます。

ポート図

図 4-31 OSER4_MEM のポート図



ポートの説明

表 4-44 OSER4_MEM のポートの説明

ポート名	I/O	説明
D3~D0	入力	OSER4_MEM データ入力信号
TX1~TX0	入力	TRI-MDDR2 を通じて Q1 を生成

ポート名	I/O	説明
TCLK	入力	DQS モジュールの DQSW0 または DQSW270 からのクロック入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
RESET	入力	非同期リセット入力、アクティブ High
Q0	出力	OSER4_MEM データ出力信号
Q1	出力	OSER4_MEM トライステートイネーブル制御出力。Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにすることができます。

パラメータの説明

表 4-45 OSER4_MEM のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする
TXCLK_POL	1'b0, 1'b1	1'b0	Q1 出力クロックの極性制御 <ul style="list-style-type: none"> ● 1'b0: 立ち上がりエッジで出力 ● 1'b1: 立ち下がりエッジで出力
TCLK_SOURCE	"DQSW", "DQSW270"	" DQSW "	TCLK ソースの選択 <ul style="list-style-type: none"> ● "DQSW": DQS モジュールの DQSW0 から。 ● "DQSW270": DQS モジュールの DQSW270 から
HWL	"false", "true"	"false"	OSER4_MEM データ d_up0/1 タイミング関係制御 <ul style="list-style-type: none"> ● "false": d_up1 は d_up0 より 1 サイクル先です。 ● "true": d_up1 と d_up0 のタイミングは同じです。

接続ルール

- Q0 は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。
- Q1 は、Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにする必要があります。
- TCLK は、DQS モジュールの DQSW0 または DQSW270 から取得し、対応するパラメータを構成する必要があります。

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
OSER4_MEM oser4_mem_inst(
    .Q0(q0),
    .Q1(q1),
    .D0(d0),
    .D1(d1),
    .D2(d2),
    .D3(d3),
    .TX0(tx0),
    .TX1(tx1),
    .TCLK(tclk),
    .FCLK(fclk),
    .PCLK(pclk),
    .RESET(reset)
);

defparam uut.GSREN="false";
defparam uut.LSREN="true";
defparam uut.HWL="false";
defparam uut.TCLK_SOURCE="DQSW";
defparam uut.TXCLK_POL=1'b0;
```

VHDL でのインスタンス化 :

```
COMPONENT OSER4_MEM
    GENERIC (GSREN:string:="false";
             LSREN:string:="true";
             HWL:string:="false";
             TXCLK_POL:bit:='0';
             TCLK_SOURCE:string:="DQSW"
```

```
);  
PORT(  
    Q0:OUT std_logic;  
    Q1:OUT std_logic;  
    D0:IN std_logic;  
    D1:IN std_logic;  
    D2:IN std_logic;  
    D3:IN std_logic;  
    TX0:IN std_logic;  
    TX1:IN std_logic;  
    TCLK:IN std_logic;  
    FCLK:IN std_logic;  
    PCLK:IN std_logic;  
    RESET:IN std_logic  
);  
END COMPONENT;  
uut:OSER4_MEM  
    GENERIC MAP (GSREN=>"false",  
                  LSREN=>"true",  
                  HWL=>"false",  
                  TXCLK_POL=>'0',  
                  TCLK_SOURCE=>"DQSW"  
    )  
PORT MAP (  
    Q0=>q0,  
    Q1=>q1,  
    D0=>d0,  
    D1=>d1,  
    D2=>d2,  
    D3=>d3,  
    TX0=>tx0,  
    TX1=>tx1,  
    TCLK=>tclk,  
    FCLK=>fclk,  
    PCLK=>pclk,
```

RESET=>reset

);

4.3.10 OSER8_MEM

プリミティブの紹介

OSER8_MEM(8 to 1 Serializer with Memory) は、メモリ機能付きの 8:1 シリアライザで、8 ビットの平行入力を 1 ビットのシリアル出力に変換できます。

サポートされるデバイス

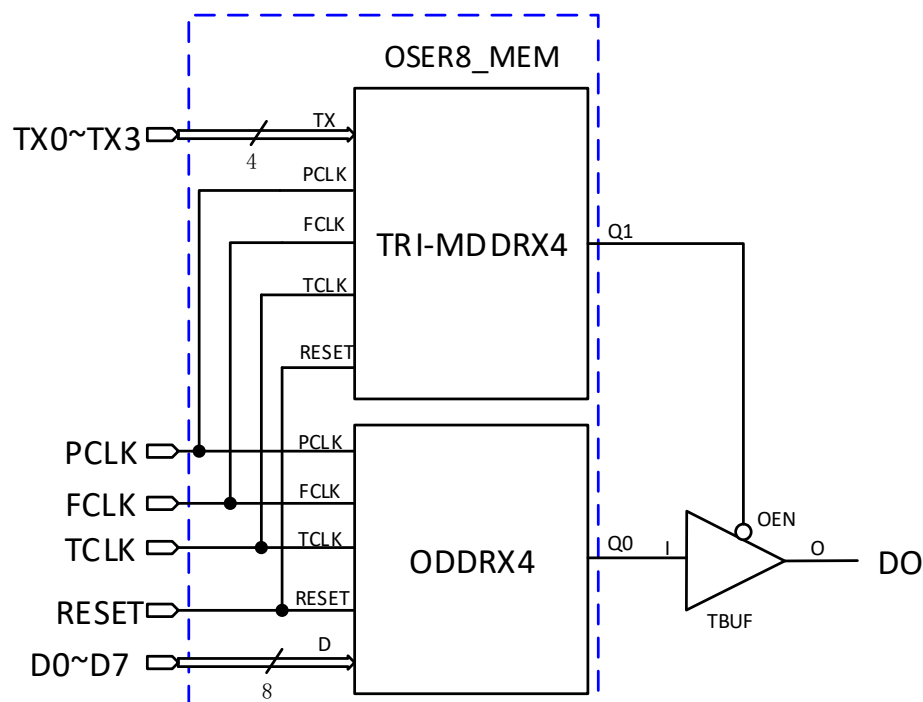
表 4-46 OSER8_MEM 対応デバイス

ファミリー	シリーズ	デバイス
Arora ファミリー	GW2A	GW2A-18, GW2A-18C, GW2A-55, GW2A-55C
	GW2AN	GW2AN-55C
	GW2AR	GW2AR-18, GW2AR-18C
	GW2ANR	GW2ANR-18C

機能の説明

OSER8_MEM モードでは、データは 8:1 シリアライズされます。OSER8 とは異なり、OSER8_MEM には DQS が必要です。TCLK を DQS の出力信号 DQSW0 または DQSW270 に接続したあと、TCLK のクロックエッジに基づいてデータを OSER8_MEM から出力します。OSER8_MEM の Q0 はデータのシリアル出力で、Q1 は Q0 に接続される IOBUF/TBUF の OEN 信号に使用されます。そのブロック図を図 4-32 に示します。

図 4-32 OSER8_MEM のブロック図

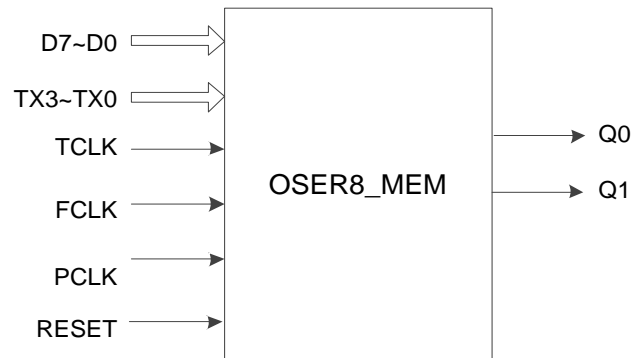


PCLK、FCLK、およびTCLKの周波数関係は： $f_{PCLK} = 1/4 f_{FCLK} = 1/4 f_{TCLK}$ 。

FCLK と TCLK の間には一定の位相関係があり、位相関係は DQS の DLLSTEP 値および WSTEP 値により決定できます。

ポート図

図 4-33 OSER8_MEM のポート図



ポートの説明

表 4-47 OSER8_MEM のポートの説明

ポート名	I/O	説明
D7~D0	入力	OSER8_MEM データ入力信号
TX3~TX0	入力	TRI-MDDR4 を通じて Q1 を生成
TCLK	入力	DQS モジュールの DQSW0 または DQSW270 からのクロック入力信号
FCLK	入力	高速クロック入力信号
PCLK	入力	プライマリクロック入力信号
RESET	入力	非同期リセット入力、アクティブ High
Q0	出力	OSER8_MEM データ出力信号
Q1	出力	OSER8_MEM トライステートイネーブル制御出力。Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにすることができます。

パラメータの説明

表 4-48 OSER8_MEM のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
GSREN	"false", "true"	"false"	グローバルリセット GSR を有効にする

パラメータ名	値の範囲	デフォルト値	説明
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする
TXCLK_POL	1'b0, 1'b1	1'b0	Q1 出力クロックの極性制御 <ul style="list-style-type: none"> ● 1'b0: 立ち上がりエッジで出力 ● 1'b1: 立ち下がりエッジで出力
TCLK_SOURCE	"DQSW", "DQSW270"	" DQSW "	TCLK ソースの選択 <ul style="list-style-type: none"> ● "DQSW": DQS モジュールの DQSW0 から。 ● DQSW270": DQS モジュールの DQSW270 から
HWL	"false", "true"	"false"	OSER8_MEM データ d_up0/1 タイミング関係制御 <ul style="list-style-type: none"> ● "false": d_up1 は d_up0 より 1 サイクル先です。 ● "true": d_up1 と d_up0 のタイミングは同じです。

接続ルール

- Q0 は OBUF に直接接続するか、IODELAY モジュールを介してその入力ポート DI に接続できます。
- Q1 は、Q0 に接続される IOBUF/TBUF の OEN 信号に接続するか、フローティングのままにする必要があります。
- TCLK は、DQS モジュールの DQSW0 または DQSW270 から取得し、対応するパラメータを構成する必要があります。

プリミティブのインスタンス化

Verilog でのインスタンス化：

```
OSER8_MEM oser8_mem_inst(
    .Q0(q0),
    .Q1(q1),
    .D0(d0),
    .D1(d1),
    .D2(d2),
```

```

        .D3(d3),
        .D4 (d4),
        .D5 (d5),
        .D6 (d6),
        .D7 (d7),
        .TX0 (tx0),
        .TX1 (tx1),
        .TX2 (tx2),
        .TX3 (tx3),
        .TCLK (tclk),
        .FCLK (fclk),
        .PCLK (pclk),
        .RESET(reset)
    );
    defparam uut.GSREN="false";
    defparam uut.LSREN ="true";
    defparam uut.HWL ="false";
    defparam uut.TCLK_SOURCE ="DQSW";
    defparam uut.TXCLK_POL=1'b0;

```

VHDL でのインスタンス化 :

```

COMPONENT OSER8_MEM
    GENERIC (GSREN:string:="false";
             LSREN:string:="true";
             HWL:string:="false";
             TXCLK_POL:bit:='0';
             TCLK_SOURCE:string:="DQSW"
    );
    PORT(
        Q0:OUT std_logic;
        Q1:OUT std_logic;
        D0:IN std_logic;
        D1:IN std_logic;
        D2:IN std_logic;
        D3:IN std_logic;
        D4:IN std_logic;

```

```
D5:IN std_logic;
D6:IN std_logic;
D7:IN std_logic;
TX0:IN std_logic;
TX1:IN std_logic;
TX2:IN std_logic;
TX3:IN std_logic;
TCLK:IN std_logic;
FCLK:IN std_logic;
PCLK:IN std_logic;
RESET:IN std_logic

);
END COMPONENT;
uut:OSER8_MEM
  GENERIC MAP (GSREN=>"false",
               LSREN=>"true",
               HWL=>"false",
               TXCLK_POL=>'0',
               TCLK_SOURCE=>"DQSW"
  )
  PORT MAP (
    Q0=>q0,
    Q1=>q1,
    D0=>d0,
    D1=>d1,
    D2=>d2,
    D3=>d3,
    D4=>d4,
    D5=>d5,
    D6=>d6,
    D7=>d7,
    TX0=>tx0,
    TX1=>tx1,
    TX2=>tx2,
    TX3=>tx3,
```

```

TCLK=>tclk,
FCLK=>fclk,
PCLK=>pclk,
RESET=>reset

```

```

);

```

4.4 遅延モジュール

4.4.1 IODELAY

プリミティブの紹介

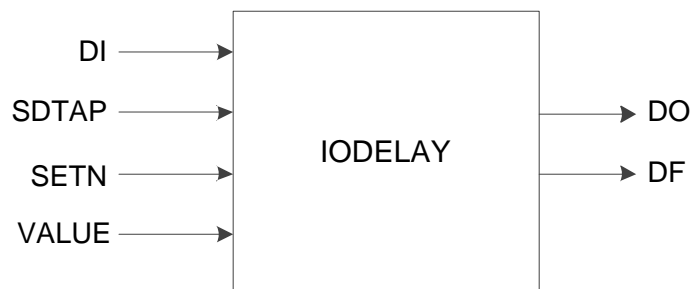
IODELAY(Input/Output delay)は IO モジュールに含まれるプログラム可能な遅延モジュールです。

機能の説明

各 I/O には、合計 128 ステップ*約 30ps(GW1N シリーズの場合)または 128 ステップ*約 18ps(GW2A シリーズの場合)の遅延を提供する IODELAY モジュールが含まれます。IODELAY は、I/O ロジックの入力または出力に使用できますが、同時に使用することはできません。

ポート図

図 4-34 IODELAY のポート図



ポートの説明

表 4-49 IODELAY のポートの説明

ポート名	I/O	説明
DI	入力	データ入力信号
SDTAP	入力	静的/動的遅延選択 <ul style="list-style-type: none"> ● 0 : 遅延を静的に調整 ● 1 : 遅延を動的に調整
SETN	入力	遅延の動的調整の方向を設定 <ul style="list-style-type: none"> ● 0: 遅延を増やす ● 1: 遅延を減らす
VALUE	入力	VALUE は立ち下がりエッジの場合の動的調整され

ポート名	I/O	説明
		た遅延値で、パルスごとに 1 遅延ステップ移動
DO	出力	データ出力
DF	出力	動的遅延調整の under-flow または over-flow を示す出力フラグビット

パラメータの説明

表 4-50 IODELAY のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
C_STATIC_DLY	0~127	0	静的遅延ステップサイズの制御

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```
IODELAY iodelay_inst(
    .DO(dout),
    .DF(df),
    .DI(di),
    .SDTAP(sdtap),
    .SETN(setn),
    .VALUE(value)
);
defparam iodelay_inst.C_STATIC_DLY=0;
```

VHDL でのインスタンス化 :

```
COMPONENT IODELAY
    GENERIC (C_STATIC_DLY:integer:=0
    );
    PORT(
        DO:OUT std_logic;
        DF:OUT std_logic;
        DI:IN std_logic;
        SDTAP:IN std_logic;
        SETN:IN std_logic;
        VALUE:IN std_logic
    );
```

```
END COMPONENT;
uut:IODELAY
    GENERIC MAP (C_STATIC_DLY=>0
    )
    PORT MAP (
        DO=>dout,
        DF=>df,
        DI=>di,
        SDTAP=>sdtap,
        SETN=>setn,
        VALUE=>value
    );
```

4.4.2 IODELAYC

プリミティブの紹介

IODELAYC(Input/Output delay)入出力遅延は IO モジュールに含まれるプログラム可能な遅延モジュールです。

サポートされるデバイス

表 4-51 IODELAYC 対応デバイス

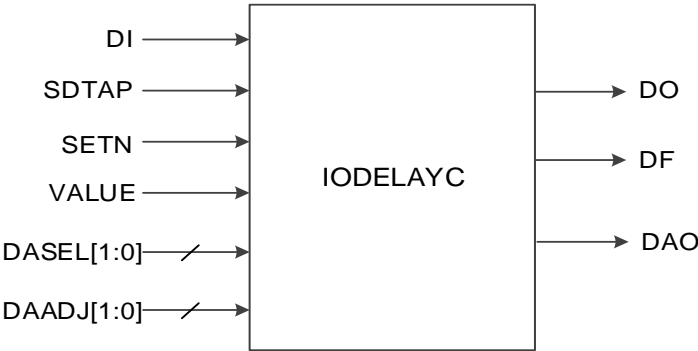
ファミリー	シリーズ	デバイス
LittleBee®	GW1N	GW1N-9C
	GW1NR	GW1NR-9C

機能の説明

各 I/O には、合計 128 ステップの遅延を提供する IODELAYC モジュールが含まれます。IODELAY と比較して、より多くの遅延調整オプションが追加されています。IODELAYC は、I/O ロジックの入力にのみ使用できます。

ポート図

図 4-35 IODELAYC のポート図



ポートの説明

表 4-52 IODELAYC のポートの説明

ポート名	I/O	説明
DI	入力	データ入力信号
SDTAP	入力	静的/動的遅延選択 <ul style="list-style-type: none"> ● 0 : 遅延を静的に調整 ● 1 : 遅延を動的に調整
SETN	入力	遅延の動的調整の方向を設定 <ul style="list-style-type: none"> ● 0: 遅延を増やす ● 1: 遅延を減らす
VALUE	入力	VALUE は立ち下がりエッジの場合の動的調整された遅延値で、パルスごとに 1 遅延ステップ移動
DASEL[1:0]	入力	DAO 遅延モードの動的制御
DAADJ[1:0]	入力	DO に対する DAO の遅延値の動的制御
DO	出力	データ出力
DAO	出力	データ遅延調整出力
DF	出力	動的遅延調整の under-flow または over-flow を示す出力フラグビット

パラメータの説明

表 4-53 IODELAYC のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
C_STATIC_DLY	0~127	0	静的遅延ステップサイズの制御
DYN_DA_SEL	“true” / “false”	false	<ul style="list-style-type: none"> ● false : DA_SEL パラメータで DAO 遅延モードを静的に制御 ● true: DASEL 信号で DAO 遅延モードを動的に制御
DA_SEL	2'b00~2'b11	2'b00	DAO 遅延モードの静的制御

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```

IODELAYC iodelayc_inst(
    .DO(dout),
    .DAO(douta),

```

```

        .DF(df),
        .DI(di),
        .SDTAP(sdtap),
        .SETN(setn),
        .VALUE(value),
        .DASEL(dasel),
        .DAADJ(daadj)
    );
defparam iodelayc_inst.C_STATIC_DLY=0;
defparam iodelayc_inst.DYN_DA_SEL="true";
defparam iodelayc_inst.DA_SEL=2'b01;
VHDL でのインスタンス化 :
COMPONENT IODELAYC
    GENERIC (C_STATIC_DLY:integer:=0;
             DYN_DA_SEL:string:="false";
             DA_SEL:bit_vector:="00"
    );
    PORT(
        DO:OUT std_logic;
        DAO:OUT std_logic;
        DF:OUT std_logic;
        DI:IN std_logic;
        SDTAP:IN std_logic;
        SETN:IN std_logic;
        VALUE:IN std_logic;
        DASEL : IN std_logic_vector(1 downto 0);
        DAADJ : IN std_logic_vector(1 downto 0)
    );
END COMPONENT;
 uut:IODELAYC
    GENERIC MAP (C_STATIC_DLY=>0,
                 DYN_DA_SEL=>"true",
                 DA_SEL=>"01"
    )
    PORT MAP (

```



```
DO=>dout,  
    DAO=>dout,  
DF=>df,  
DI=>di,  
SDTAP=>sdtap,  
SETN=>setn,  
VALUE=>value,  
DASEL=>dasel,  
DAADJ=>daadj  
);
```

4.4.3 IODELAYB

プリミティブの紹介

IODELAYB(Input/Output delay)入出力遅延はIOモジュールに含まれるプログラム可能な遅延モジュールです。

サポートされるデバイス

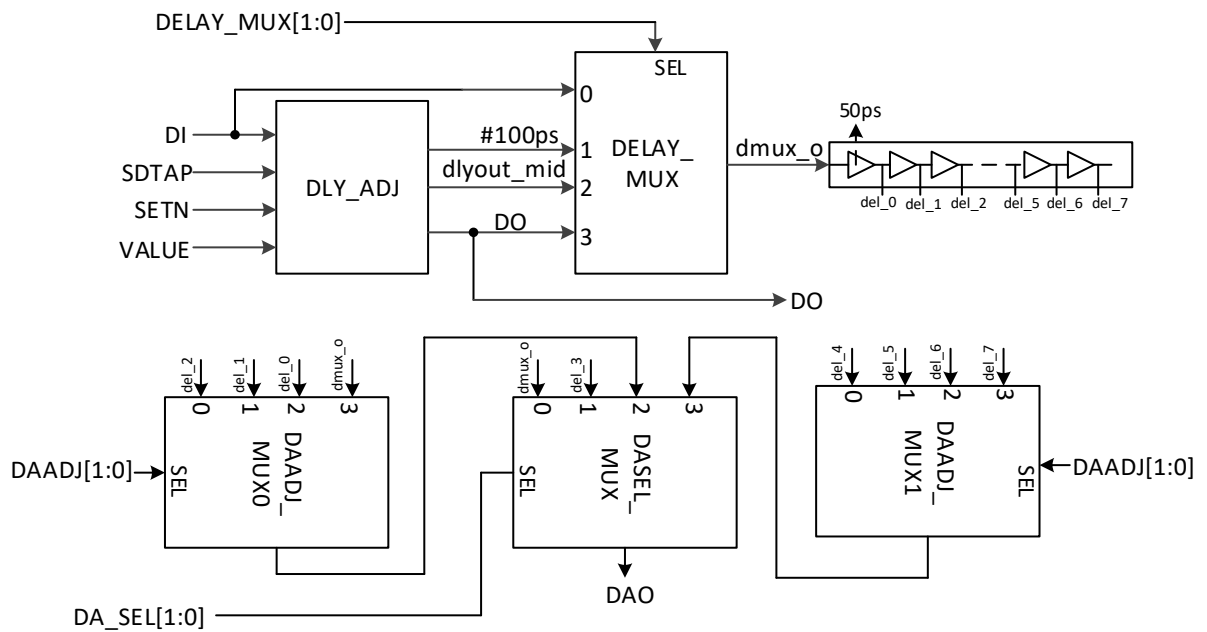
表 4-54 IODELAYB 対応デバイス

ファミリー	シリーズ	デバイス
LittleBee®	GW1N	GW1N-2, GW1N-1P5, GW1N-2B, GW1N-1P5B
	GW1NR	GW1NR-2, GW1NR-2B

機能の説明

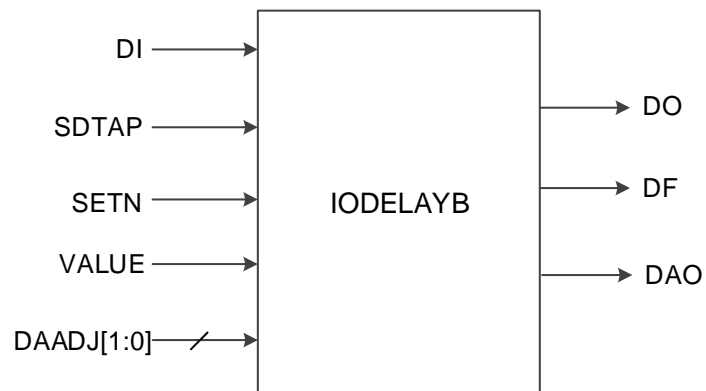
各 I/O には、合計 128 ステップの遅延を提供する IODELAYB モジュールが含まれます。IODELAY と比較して、より多くの遅延調整オプションが追加されています。そのブロック図を図 4-36 に示します。IODELAYB は、I/O ロジックの入力にのみ使用できます。

図 4-36 IODELAYB の構造



ポート図

図 4-37 IODELAYB のポート図



ポートの説明

表 4-55 IODELAYB のポートの説明

ポート名	I/O	説明
DI	入力	データ入力信号
SDTAP	入力	静的/動的遅延選択 ● 0 : 遅延を静的に調整 ● 1 : 遅延を動的に調整
SETN	入力	遅延の動的調整の方向を設定 ● 0: 遅延を増やす ● 1: 遅延を減らす
VALUE	入力	VALUE は立ち下がりエッジの場合の動的調整された遅延値で、パルスごとに 1 遅延ステップ移動。

ポート名	I/O	説明
DAADJ[1:0]	入力	DO に対する DAO の遅延値の動的制御
DO	出力	データ出力
DAO	出力	データ遅延調整出力
DF	出力	動的遅延調整の under-flow または over-flow を示す出力フラグビット。

パラメータの説明

表 4-56 IODELAYB のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
C_STATIC_DLY	0~127	0	静的遅延ステップサイズの制御
DELAY_MUX	2'b00~2'b11	2'b00	Delay MUX の選択 <ul style="list-style-type: none"> ● 2'b00:dmux_o=DI; ● 2'b01:#100ps dmux_o=DI; ● 2'b10:dmux_o=dlyout_mid; ● 2'b11:dmux_o=DO。
DA_SEL	2'b00~2'b11	2'b00	DAO 遅延モードの静的制御

注記：

IODELAYB を使用する場合、パラメータ DELAY_MUX と DA_SEL の関係は次のとおりです。

- DELAY_MUX:2/3 -> DA_SEL:0/1。つまり、DELAY_MUX が 2 または 3 の場合、DA_SEL は 0 または 1 になります。
- DELAY_MUX:0/1 -> DA_SEL:0/2/3。つまり、DELAY_MUX が 0 または 1 の場合、DA_SEL は 0、2、または 3 になります。

接続ルール

DO は IDDR/IDES に接続できず、DAO は IDDR/IDES のデータ入力にのみ接続できます。

プリミティブのインスタンス化

Verilog でのインスタンス化：

```
IODELAYB iodelayb_inst(
    .DO(dout),
    .DAO(douta),
    .DF(df),
    .DI(di),
    .SDTAP(sdtap),
    .SETN(setn),
    .VALUE(value),
```

```

        .DAADJ(daadj)
    );
    defparam iodelayb_inst.C_STATIC_DLY=0;
    defparam iodelayb_inst.DELAY_MUX = 2'b00;
    defparam iodelayb_inst.DA_SEL=2'b00;
VHDL でのインスタンス化 :
    COMPONENT IODELAYB
        GENERIC (C_STATIC_DLY:integer:=0;
                 DELAY_MUX : bit_vector := "00";
                 DA_SEL:bit_vector:= "00"
                );
        PORT(
            DO:OUT std_logic;
            DAO:OUT std_logic;
            DF:OUT std_logic;
            DI:IN std_logic;
            SDTAP:IN std_logic;
            SETN:IN std_logic;
            VALUE:IN std_logic;
            DAADJ : IN std_logic_vector(1 downto 0)
        );
    END COMPONENT;
    uut:IODELAYB
        GENERIC MAP (C_STATIC_DLY=>0,
                     DELAY_MUX =>"00",
                     DA_SEL=>"00"
                    )
        PORT MAP (
            DO=>dout,
            DAO=>douta,
            DF=>df,
            DI=>di,
            SDTAP=>sdtap,
            SETN=>setn,
            VALUE=>value,

```

```
DAADJ=>daadj
);
```

4.5 サンプルングモジュール

プリミティブの紹介

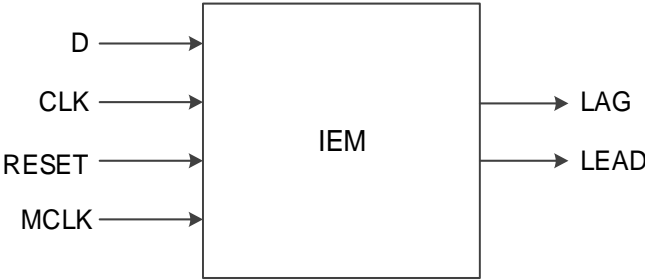
IEM(Input Edge Monitor)は、IO モジュールに含まれるサンプルングモジュールです。

機能の説明

IEM は、データエッジをサンプルングするために使用され、遅延モジュールと併用して動的サンプルングウィンドウを調整できます。DDR モードで使用されます。

ポート図

図 4-38 IEM のポート図



ポートの説明

表 4-57 IEM のポートの説明

ポート名	I/O	説明
D	入力	データ入力信号
CLK	入力	クロック入力信号
RESET	入力	非同期リセット入力、アクティブ High
MCLK	入力	IEM 検出クロック。ユーザーロジックから取得でき、出力フラグに使用されます。
LAG	出力	IEM エッジ比較 LAG 出力フラグ
LEAD	出力	IEM エッジ比較 LEAD 出力フラグ

パラメータの説明

表 4-58 IEM のパラメータの説明

パラメータ名	値の範囲	デフォルト値	説明
WINSIZE	"SMALL","MIDSMALL", "MIDLARGE","LARGE"	"SMALL"	ウィンドウサイズの設定
GSREN	"false", "true"	"false"	グローバルリセット GSR

パラメータ名	値の範囲	デフォルト値	説明
			を有効にする
LSREN	"false", "true"	"true"	ローカルリセット RESET を有効にする

プリミティブのインスタンス化

Verilog でのインスタンス化 :

```

IEM iem_inst(
    .LAG(lag),
    .LEAD(lead),
    .D(d),
    .CLK(clk),
    .MCLK(mclk),
    .RESET(reset)
);

defparam iodelay_inst.WINSIZE = "SMALL";
defparam iodelay_inst.GSREN = "false";
defparam iodelay_inst.LSREN = "true";

```

VHDL でのインスタンス化 :

```

COMPONENT IEM
    GENERIC (WINSIZE:string:="SMALL";
             GSREN:string:="false";
             LSREN:string:="true"
    );
    PORT(
        LAG:OUT std_logic;
        LEAD:OUT std_logic;
        D:IN std_logic;
        CLK:IN std_logic;
        MCLK:IN std_logic;
        RESET:IN std_logic
    );
END COMPONENT;

uut:IEM

```

```
    GENERIC MAP (WINSIZE=>"SMALL",  
                  GSREN=>"false",  
                  LSREN=>"true"  
    )  
    PORT MAP (  
        LAG=>lag,  
        LEAD=>lead,  
        D=>d,  
        CLK=>clk,  
        MCLK=>mclk,  
        RESET=>reset  
    );
```

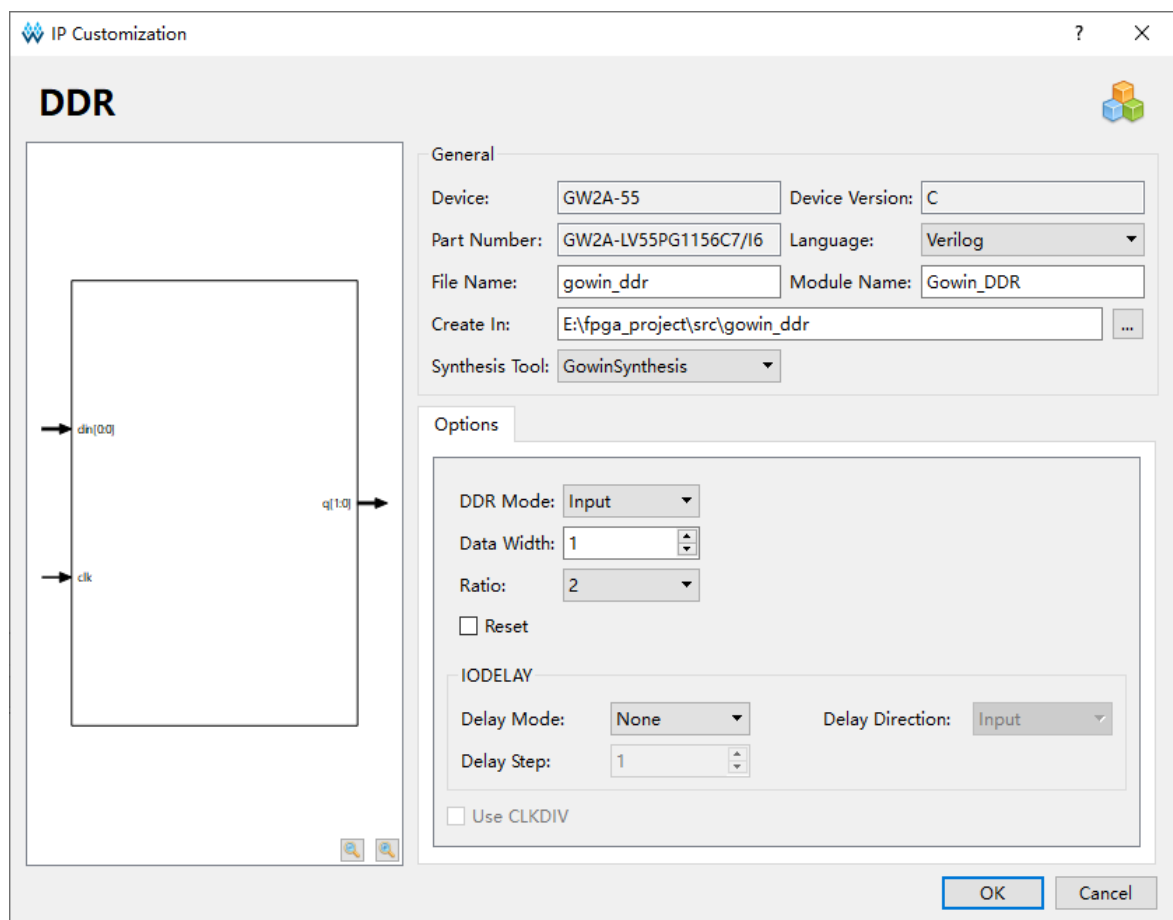
5 IP の呼び出し

現在、DDR のみがサポートされます。IP Core Generator のインターフェースで DDR をクリックすると、右側に DDR の概要が表示されます。

5.1 IP の構成

IP Core Generator インターフェースで “DDR” をダブルクリックすると、“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” 構成タブおよびポート図があります(図 5-1)。

図 5-1 DDR IP の構成ウィンドウ



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。
- **Synthesis Tool** : 合成ツールを選択します。
- **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 5-1)。

- **DDR Mode** : “Input” (入力)、“Output” (出力)、“Tristate” (トライステート)、および “Bidirectional” (双方向)を含む 4 つの DDR モードがあります。
- **Data Width** : データ幅 : DDR のデータ幅(1~64)を構成します。
- **Ratio** : DDR データ変換の比率(2、4、7、8、10、16 を含む)を構成します。
- **Reset** : Ratio が 2 に選択されている場合、このオプションを有効にするか無効にするかを選択できます。有効にすると、IDDRC または ODDRC がインスタンス化されます。
- **IODELAY** : DDR に遅延モジュールを使用するかどうかを構成します。
 - **Delay Mode** : 遅延モードを構成します。“None” は IODELAY を使用しないことを意味し、“Dynamic” は IODELAY を使用して遅延ステップ数を動的に調整することを意味し、“Static” は IODELAY を使用して遅延ステップ数を静的に調整することを意味します。
 - **Delay Step** : 遅延を静的に調整するためのステップ数(1~

128)を選択します。

- Delay Direction : Bidirectional という DDR Mode で IODELAY を使用する場合は、IODELAY を入力側または出力側に接続するかを選択します。
- Use CLKDIV : 有効にすると、CLKDIV はインスタンス化され、クロック信号 fclk は周波数分割されます。Ratio が 2 の場合、チェックできません。

3. ポート図

ポート図は、IP Core の構成結果を表示します(図 5-1)。

5.2 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_dds.v” は完全な verilog モジュールです。
- “gowin_dds_tmp.v” は IP のテンプレートファイルです。
- gowin_dds.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。

