SiTCPとSiTCP-XGの説明

2021/10/29

株式会社BeeBeans Technologies
営業部　細谷　力

Contents

会社紹介

会社名	株式会社 Bee Beans Technologies（英文社名Bee Beans Technologies Co.,Ltd.）
事業内容	研究機関、大学等で開発された実験用装置の販売と導入支援
	客先常駐による各実験の準備業務と実施とデータ処理の補助業務
	監視、制御及び解析ソフトウェア、WEB アプリケーションの設計・製作
	アナログ及びデジタル電子回路の設計・製作
	FPGA 論理回路の設計・製作
事業所	本社　つくば市大穂109 東海事業所　那珂郡東海村白方162-1　IQBRC A308
設立	2006年9月

自己紹介

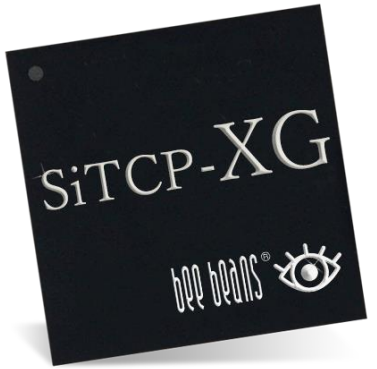
名前	細谷　力　（ほそや　りき：英文名　Ricky Hosoya）
入社	2021年8月1日（コロナ対応で出社制限中に初出勤）
所属	本社　営業部
	担当営業：KEK・JAEA・QST　その他（宮本担当分の提出書類作成、宮本担当分の議事録作成、宮本担当分のetc..）
趣味	映画：鬼滅の刃・無限列車編（炭次郎が「俺の家族がそんなこと言うわけないだろォ！！」のセリフは心が震えた） TENET　（時間が逆行可能な世界における順行側の名も無き男の戦いを描いた作品。続編が楽しみ） パラサイト 半地下の家族　（コメディとサスペンスの切り替えがうまい。ラスト30分の展開が怖い） 写経：般若心経　空の思想/全ての物事は変化し続け、変化し続けるが物事の本質は変わらず存在する 何か物理の探求に相通じるところがありそうな気がします ダンス：HipHop　Clip WalkからKick&OpenやSlideの軽い動きから突然のSwipeとBackspinで決めるのが好きです

10月末で試用期間が終了し、この発表が研修最後のいわば「卒業発表」です。
ご聴講よろしくお願いいたします。

SiTCP/SiTCP-XGとは

- Xilinx社のFPGA上で実現されたハードウェアによるTCP/IPの実装です。
- ソフトウェアを介在させずに1Gbps/10Gbpsまでの高速なデータ転送を可能にするサーバーを構築できます。
- FPGAの対応状況は以下の通り（2021/10/01現在）

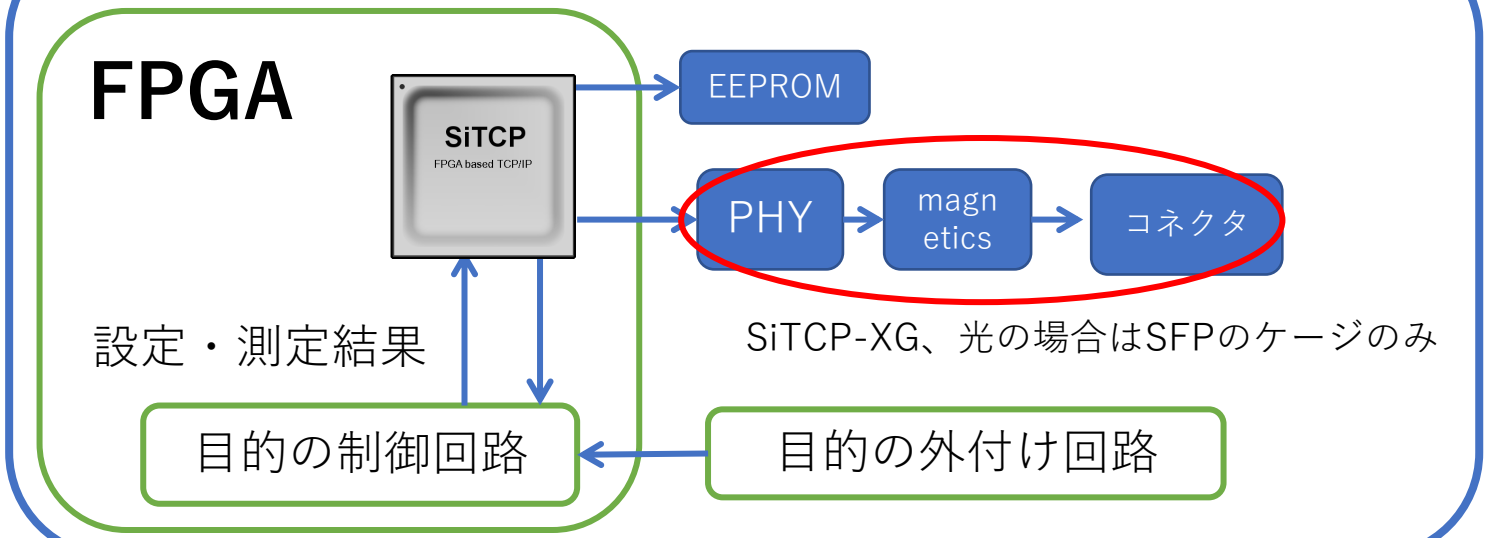
Spartan6/7
Artix7
Kintex7
Virtex4/5/6/7
Kintex_UltraScale
Virtex_UltraScalePlus



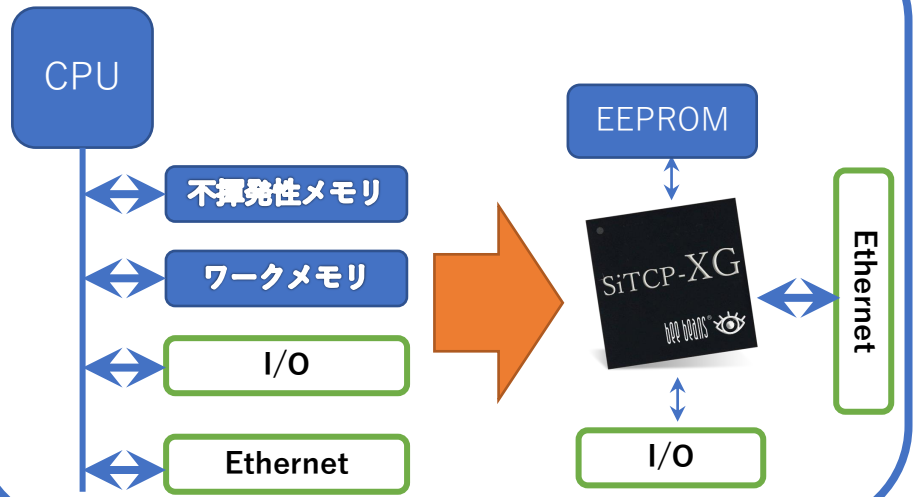
Kintex7
Virtex7

SiTCP/SiTCP-XGの特徴

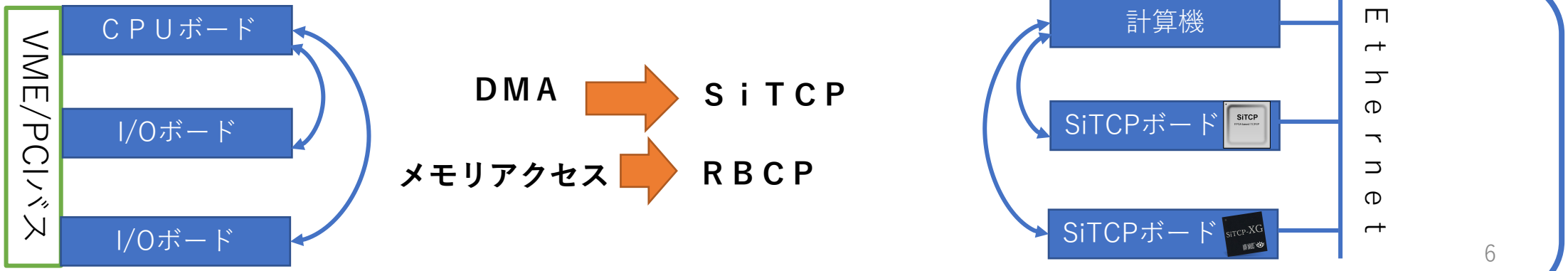
回路規模が小さい



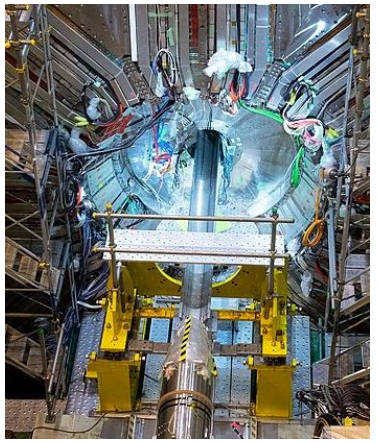
ＣＰＵが不要



外部のFPGAをまるで計算機内のI/Oの様に扱える



SiTCP/SiTCP-XGの適用シーン



広帯域転送であることを利用して

- 物理実験での大容量・多チャンネル事象の同時転送
  - 加速器、粒子衝突実験、etc
- 画像診断データ伝送
  - 医療系画像診断（放射線、MRI、etc）X線非破壊検査
- 高解像度映像伝送

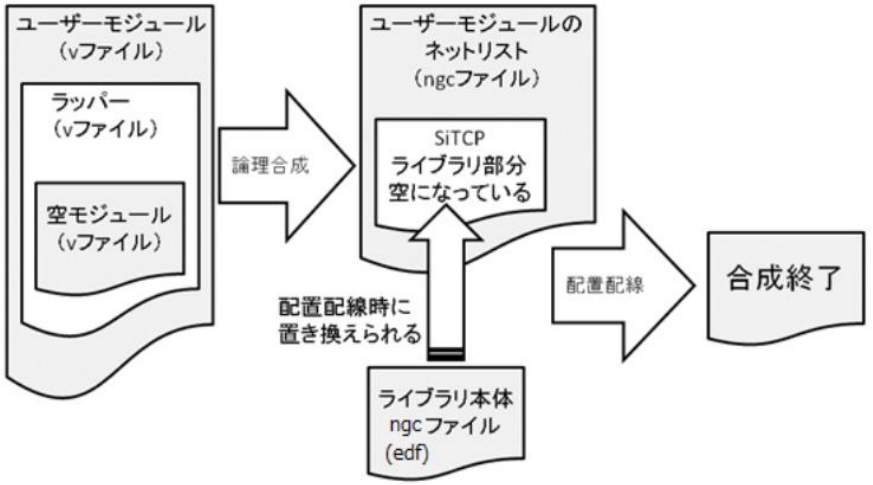
小規模回路であることを利用して

- 機器組み込みTCP/IPの実装
  - 測定器の設定や出力、計測器入出力etc

あらゆる機器にTCP/IPを提供

SiTCP 実装の概要
（ライブラリの使用方法）

- SiTCP ライブラリは以下の 4 つのファイルで構成されます。
  - ライブラリ本体： ngc/edf ファイル
  - 入出力定義モジュール: v ファイル
  - ラッパー:v ファイル
  - タイマーモジュール:TIMER.V



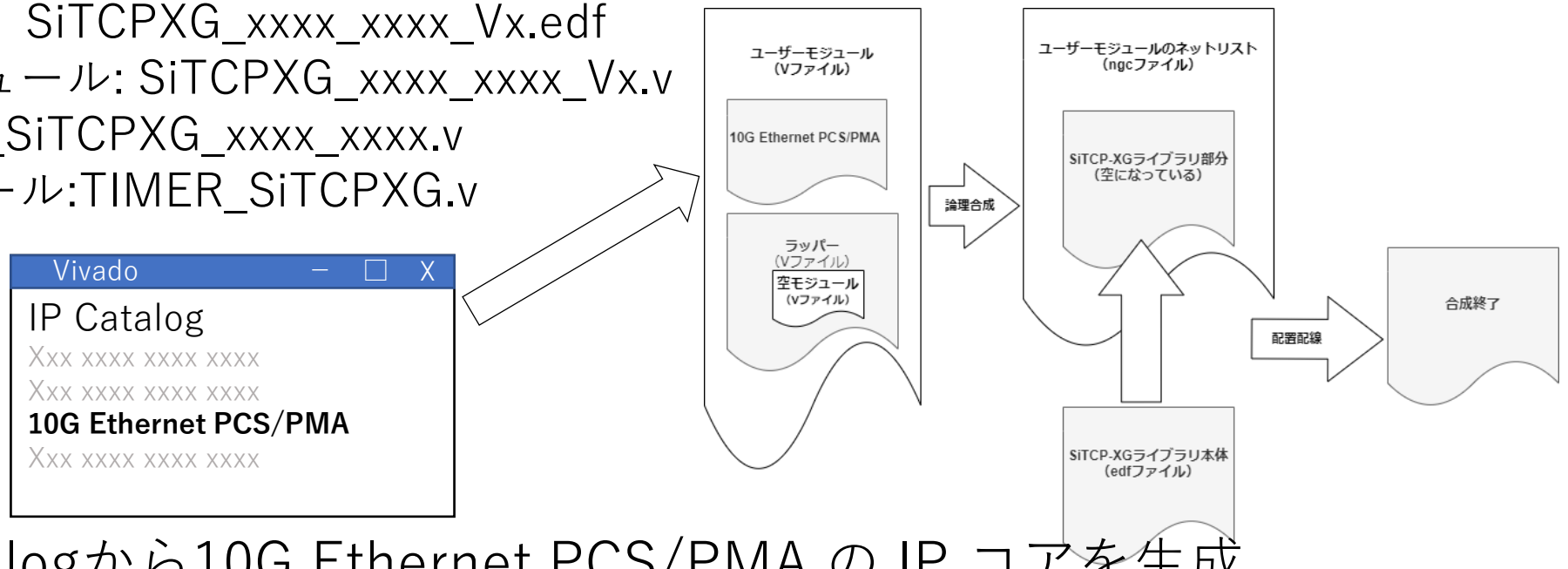
実装手順

1. ユーザーモジュールをHDLで記述する
2. ラッパーファイルをユーザーモジュールに追記（インスタンシエーション）
3. ISE/VivadoでSiTCPのngc/edfとともに合成

（詳しくは　SiTCP 説明書 (https://www.sitcp.net/doc/SiTCP.pdf)をご確認ください

SiTCP-XG　実装の概要

- SiTCP-XGライブラリは以下の 4 つのファイルで構成されます。
  - ライブラリ本体： SiTCPXG_xxxx_xxxx_Vx.edf
  - 入出力定義モジュール: SiTCPXG_xxxx_xxxx_Vx.v
  - ラッパー:WRAP_SiTCPXG_xxxx_xxxx.v
  - タイマーモジュール:TIMER_SiTCPXG.v



実装手順

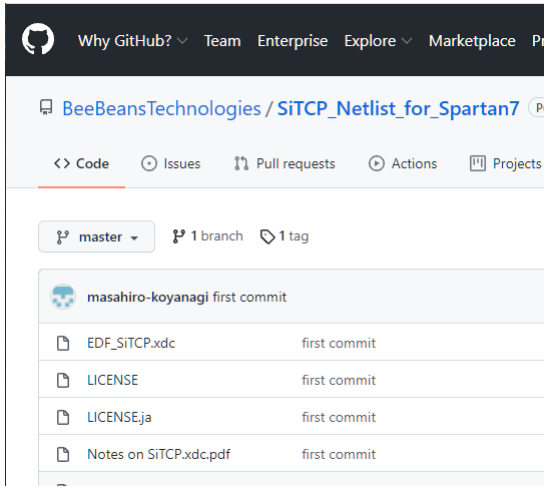
1. VivadoのIP Catalogから10G Ethernet PCS/PMA の IP コアを生成
2. ユーザーモジュールをHDLで記述する
3. ラッパーファイルをユーザーモジュールに追記（インスタンシエーション）
4. VivadoでSiTCPのedfとともに合成

詳しくはSiTCPXG 説明書 (https://www.bbtech.co.jp/download-files/sitcp/SiTCPXG_Manual_1.3.pdf)をご確認ください

SiTCP実装図解

準備

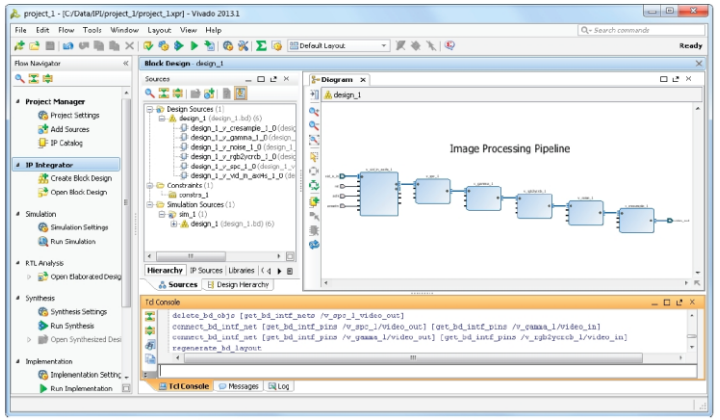
SiTCPをGithubからダウンロード



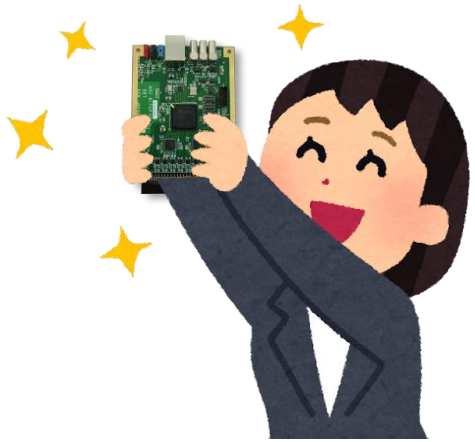
ユーザー回路の記述

- Wrapperのインスタンシエーション

```
generate
    genvar JMPR_var ;
     for (JMPR_var=0;JMPR_var<4;JM
        IBUF
        PULLUP
     end
endgenerate
```

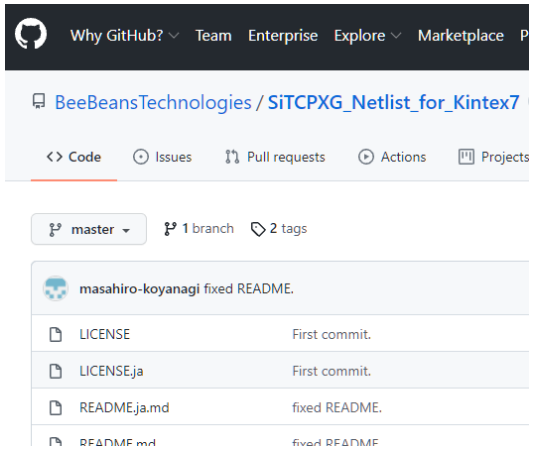


合成

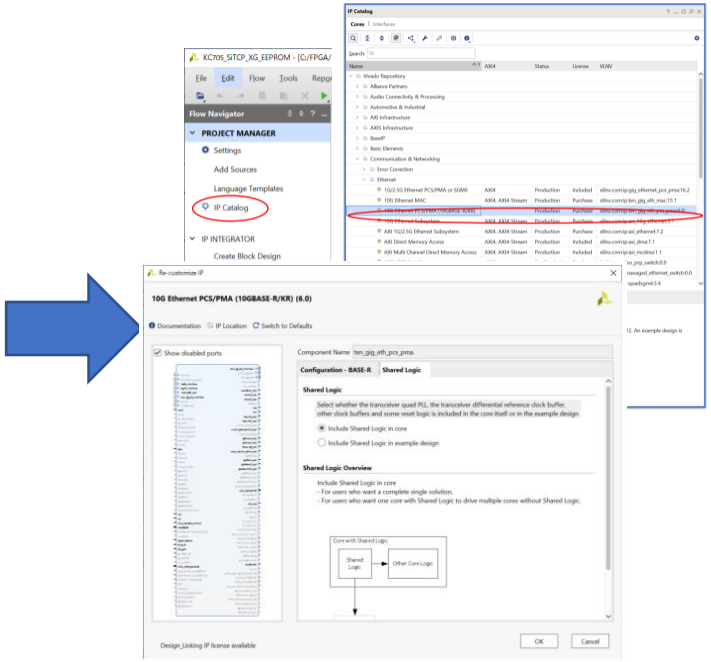


SiTCP-XG実装図解

準備1
SiTCPをGithubからダウンロード

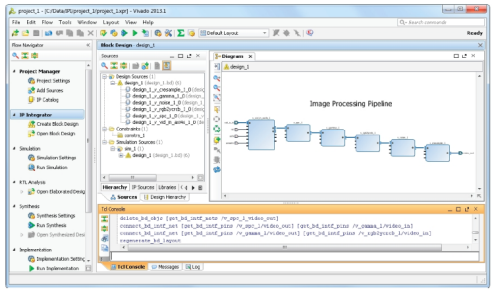


準備2　IPカタログからコアを生成

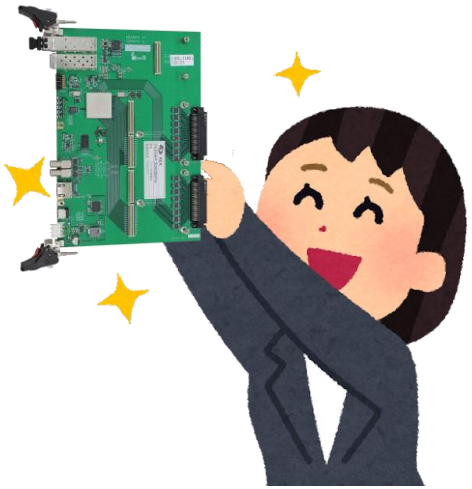


ユーザー回路の記述
- 10G Ethernet PCS/PMA
- Wrapperのインスタンシエーション

```
generate
    genvar JMPR_var ;
    for (JMPR_var=0;JMPR_var<4;JMI
        IBUF
        PULLUP
    end
endgenerate
```



合成



SiTCPライブラリV11での追加項目

- SiTCP Forum（ https://www.bbtech.co.jp/sitcp-forum/#/discussion/32/sitcp-v11-changes ）で少し触れていますが、SiTCPライブラリV11では内部レジスタ 0xFFFF_FF10のbit6～bit3に以下の機能を追加しています

  - bit6：Client ARP
  - bit5：Dup Ack
  - bit4：MIF Initializer
  - bit3：MAC flow control

  ※いずれも初期値は0

bit6：Client ARP／bit5：Dup Ack

- Client ARP
  - クライアントモード時、ライブラリV10以前は接続先サーバーのMACアドレスを0xFFFF_FF30～0xFFFF_FF35に設定していましたが、このbitを1にすると設定が不要となります
    - ✓ SiTCPからARPリクエストを行い、返送されたARPリプライから取得したMACアドレスを次回以降の通信に使用します

- Dup Ack
  - このbitを1にすると、SiTCP は接続先からのデータ受信後、次に何らかのパケットを受け取るまでACKパケットを約2ms周期で送信します（ライブラリV10以前の仕様）
  - 0の場合は接続先のデータ送信に対しACKを1個のみ送信します

bit4：MIF Initializer／bit3：MAC flow control

- MIF Initializer
  - 1にするとSiTCPを使用したPHYの（ベンダ部分を除く）MIF初期化が可能になります
  - 具体的な設定内容は「SiTCP内部レジスタ解説書」をご確認ください。

  https://www.bbtech.co.jp/download-files/sitcp/SiTCP_Register_Manual_1.0.2.pdf
- MAC flow control
  - 1に設定すると、スイッチングハブ等から送出されるIEEE802.3XのPAUSEフレームに対応します（PAUSEフレーム送出機能はありません）

0xFFFF_FF10 その他のbitに関する補足事項

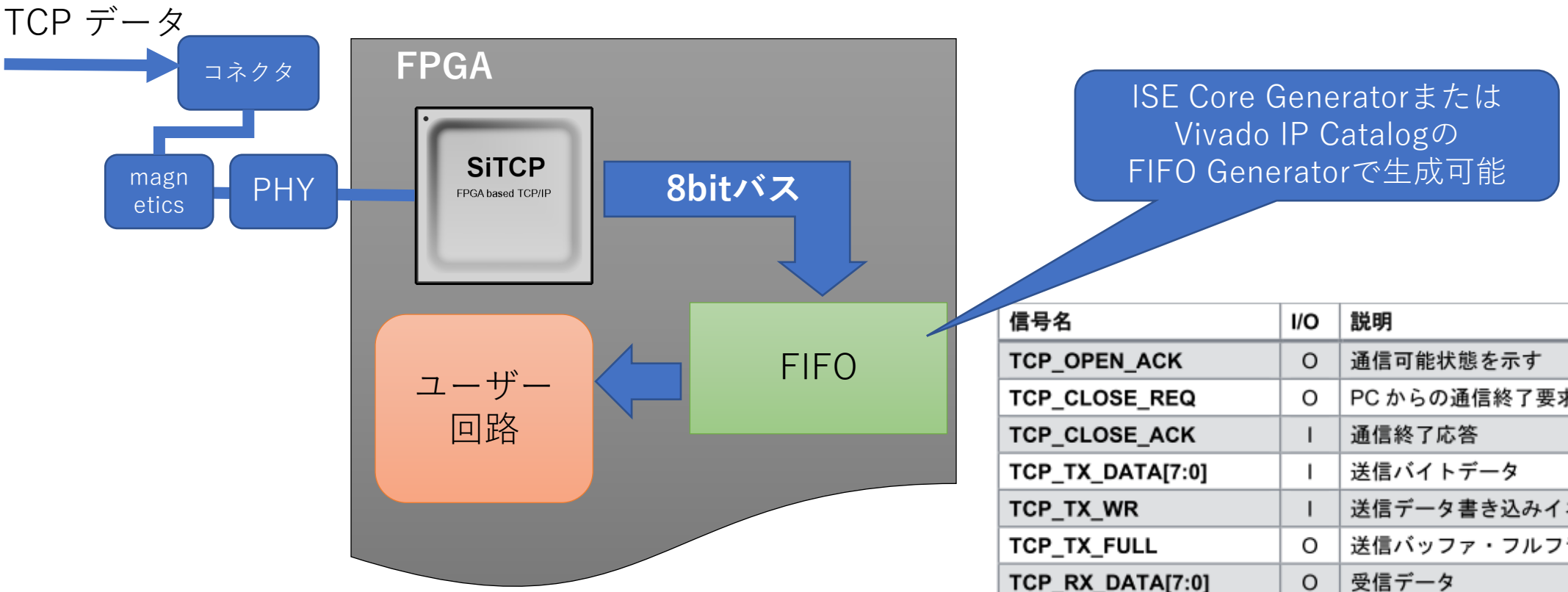
- bit2：Keep alive packet（初期値0）
  - セッション確立後、有効なパケットを受信しないまま一定時間経過すると(※)、SiTCPがセッションを切断します

    ※　0xFFFF_FF2A～0xFFFF_FF2Bのタイマの働きによるものです
  - ⇒セッションを保持する場合にはこのbitを1に設定してください

- bit0：Nagle buffering（初期値1）
  - 1に設定されている時、SiTCPは送信データサイズがMSS以上になるか、または書き込み開始から約4ms経過後にパケットを送出します
  - データサイズが小さく（～数十Byte）、応答時間を短くしたい場合は0に設定すると改善されることがあります

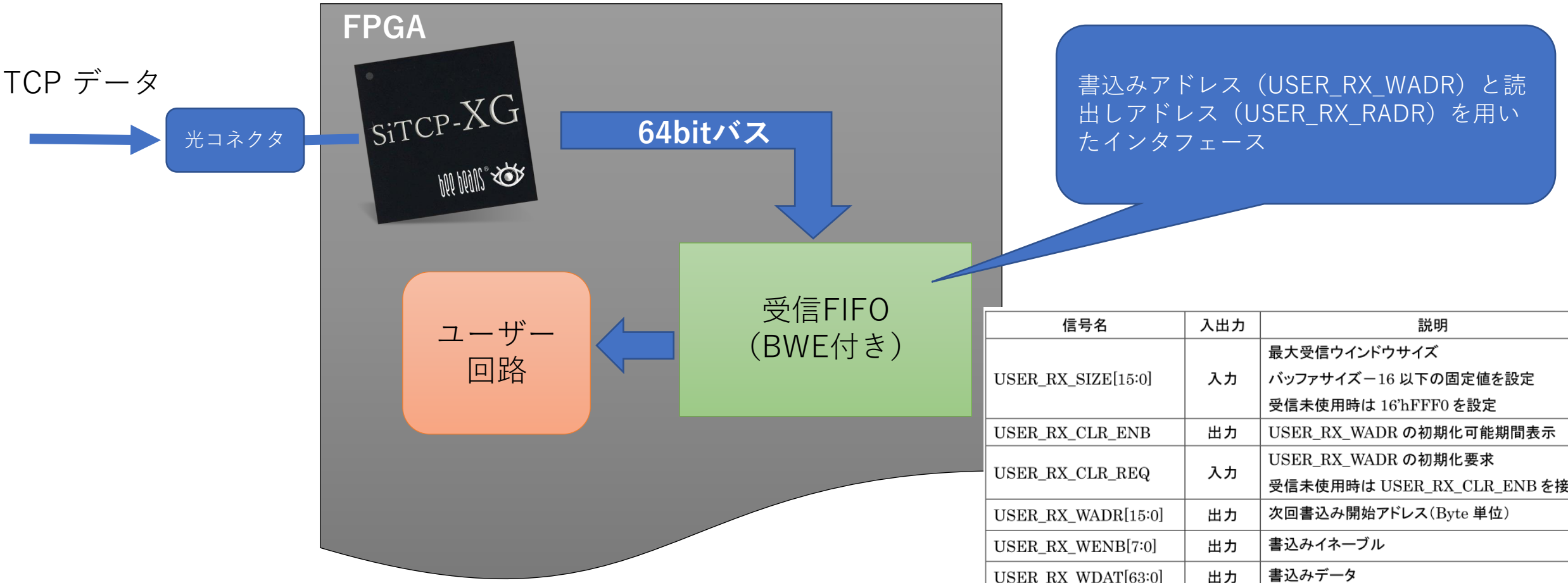
SiTCP TCP 受信

受信機能を使用する場合には SiTCP とユーザー回路の間に FIFO メモリを挿入する必要があります。



信号名	I/O	説明
TCP_OPEN_ACK	O	通信可能状態を示す
TCP_CLOSE_REQ	O	PC からの通信終了要求を受信した事を示す
TCP_CLOSE_ACK	I	通信終了応答
TCP_TX_DATA[7:0]	I	送信バイトデータ
TCP_TX_WR	I	送信データ書き込みイネーブル
TCP_TX_FULL	O	送信バッファ・フルフラグ
TCP_RX_DATA[7:0]	O	受信データ
TCP_RX_WR	O	受信データ有効
TCP_RX_WC[15:0]	I	受信バッファに格納されているデータバイト数

SiTCP-XG TCP 受信



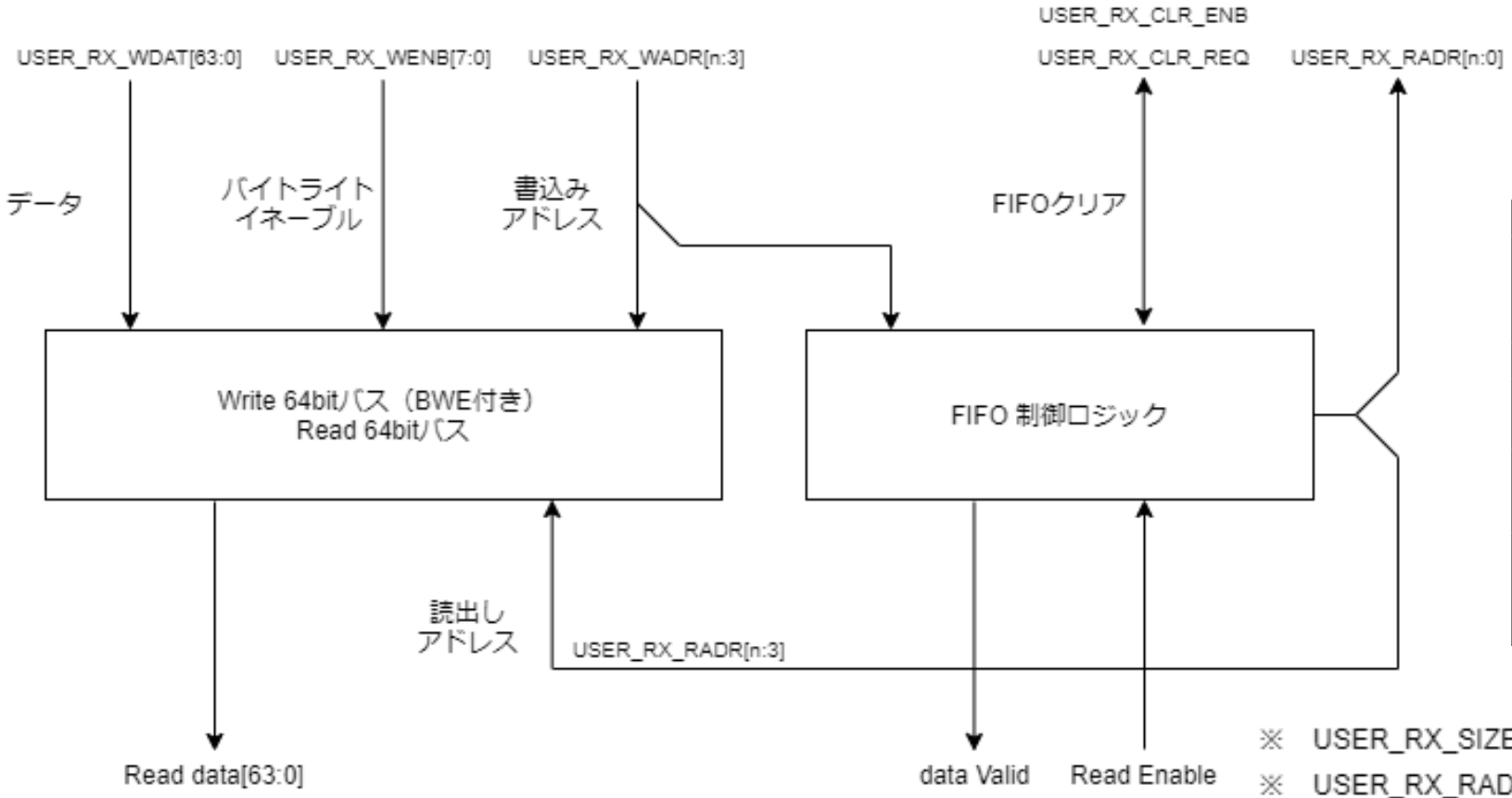
信号名	入出力	説明
USER_RX_SIZE[15:0]	入力	最大受信ウインドウサイズ バッファサイズ－16 以下の固定値を設定 受信未使用時は 16'hFFF0 を設定
USER_RX_CLR_ENB	出力	USER_RX_WADR の初期化可能期間表示
USER_RX_CLR_REQ	入力	USER_RX_WADR の初期化要求 受信未使用時は USER_RX_CLR_ENB を接続
USER_RX_WADR[15:0]	出力	次回書込み開始アドレス（Byte 単位）
USER_RX_WENB[7:0]	出力	書込みイネーブル
USER_RX_WDAT[63:0]	出力	書込みデータ
USER_RX_RADR[15:0]	入力	読出しアドレス 未使用時は USER_RX_WADR[15:0]を接続

SiTCP-XG 受信FIFO

- 固定バス（回路が単純であり、使用がしやすい）
  - 64bit
  - 32bit
  - 16bit
  - 8bit
- 可変バス（最大レートを出しながら、任意長のデータを受け取れる。）
  - 64bit

SiTCP-XG　受信　64bit固定バス

（RxBufferSize = "LongLong"）

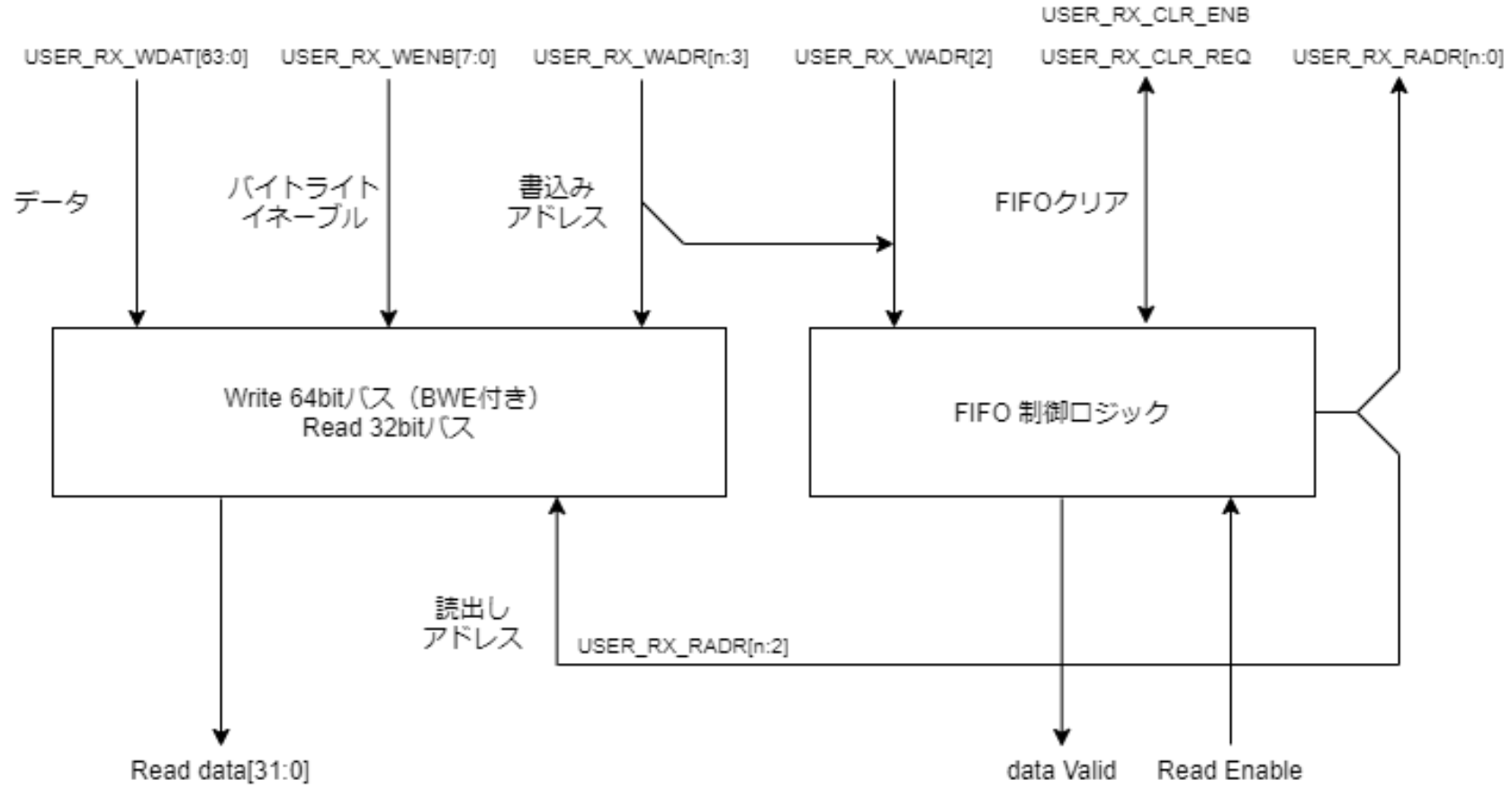


メモリ容量に応じた固定値
を入力してください

信号名	入出力	説明
USER_RX_SIZE[15:0]	入力	最大受信ウインドウサイズ バッファサイズー16 以下の固定値を設定 受信未使用時は 16'hFFF0 を設定
USER_RX_CLR_ENB	出力	USER_RX_WADR の初期化可能期間表示
USER_RX_CLR_REQ	入力	USER_RX_WADR の初期化要求 受信未使用時は USER_RX_CLR_ENB を接続
USER_RX_WADR[15:0]	出力	次回書込み開始アドレス（Byte 単位）
USER_RX_WENB[7:0]	出力	書込みイネーブル
USER_RX_WDAT[63:0]	出力	書込みデータ
USER_RX_RADR[15:0]	入力	読出しアドレス 未使用時は USER_RX_WADR[15:0]を接続

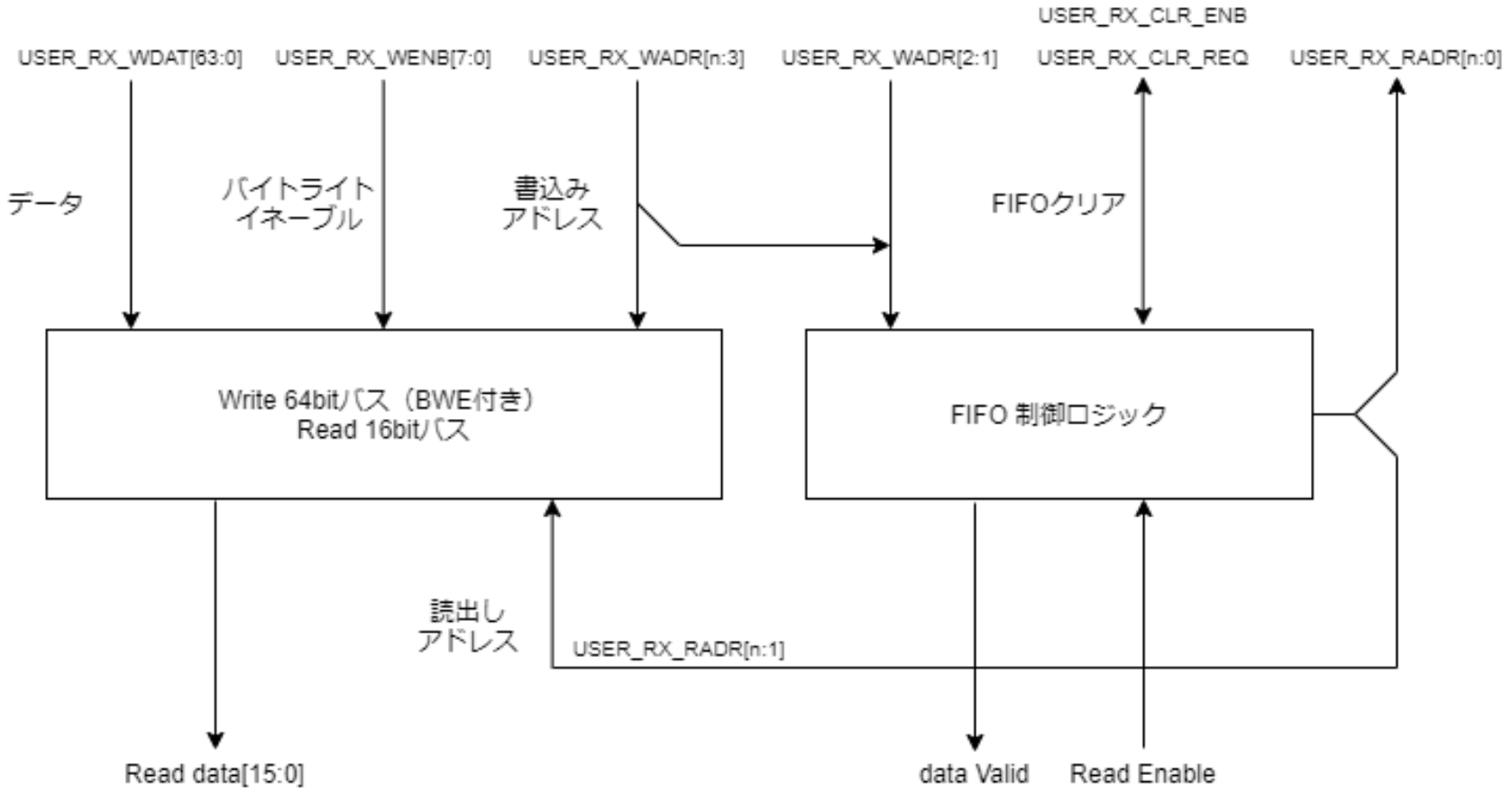
※　USER_RX_SIZEには、メモリ容量に応じた固定値を入力
※　USER_RX_RADR[2:0]= **3'b000**
※　USER_RX_RADRはUSER_RX_CLR_REQ＝1の時にUSER_RX_WADRをコピー
※　RX_CLR_REQ＝1でUSER_RX_WADR[2:0]を3'b000にクリア
※　Empty = (USER_RX_RADR[n:3] == USER_RX_WADR[n:3])

SiTCP-XG　受信　32bit固定バス(RxBufferSize = "LongWord")



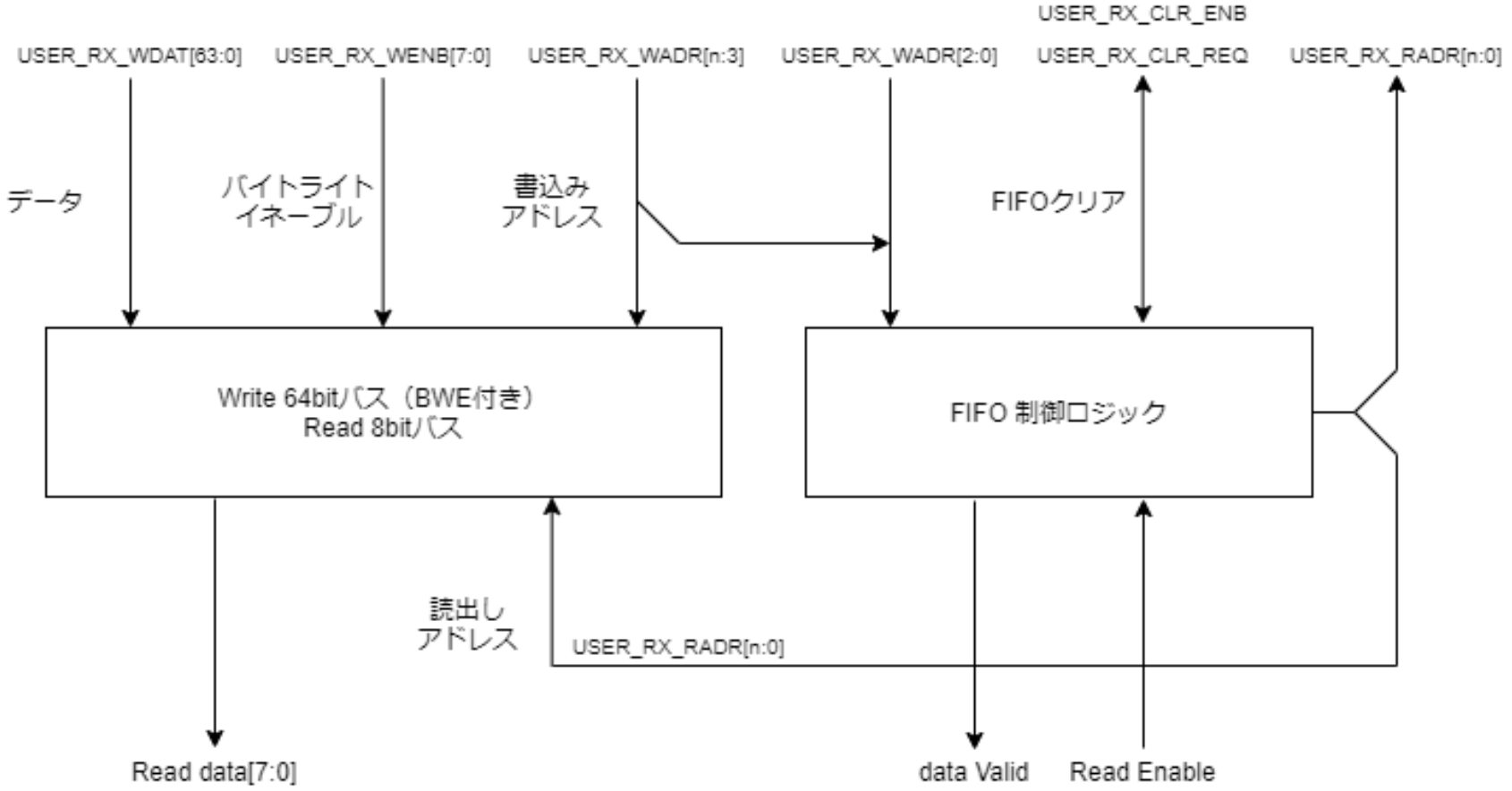
※　USER_RX_SIZEには、メモリ容量に応じた固定値を入力すること
※　USER_RX_RADR[1:0]= **2'b000**
※　USER_RX_RADRはUSER_RX_CLR_REQ＝1の時にUSER_RX_WADRをコピーすること
※　SiTCPXGは、USER_RX_CLR_REQ＝1でUSER_RX_WADR[2:0]を3'b000にクリアする
※　Empty = (USER_RX_RADR[n:2] == USER_RX_WADR[n:2])となる

SiTCP-XG　受信　16bit固定バス(RxBufferSize = "Word")



※　USER_RX_SIZEには、メモリ容量に応じた固定値を入力すること
※　USER_RX_RADR[0]= **1'b0**
※　USER_RX_RADRはUSER_RX_CLR_REQ＝1の時にUSER_RX_WADRをコピーすること
※　SiTCPXGは、USER_RX_CLR_REQ＝1でUSER_RX_WADR[2:0]を3'b000にクリアする
※　Empty = (USER_RX_RADR[n:1] == USER_RX_WADR[n:1])となる

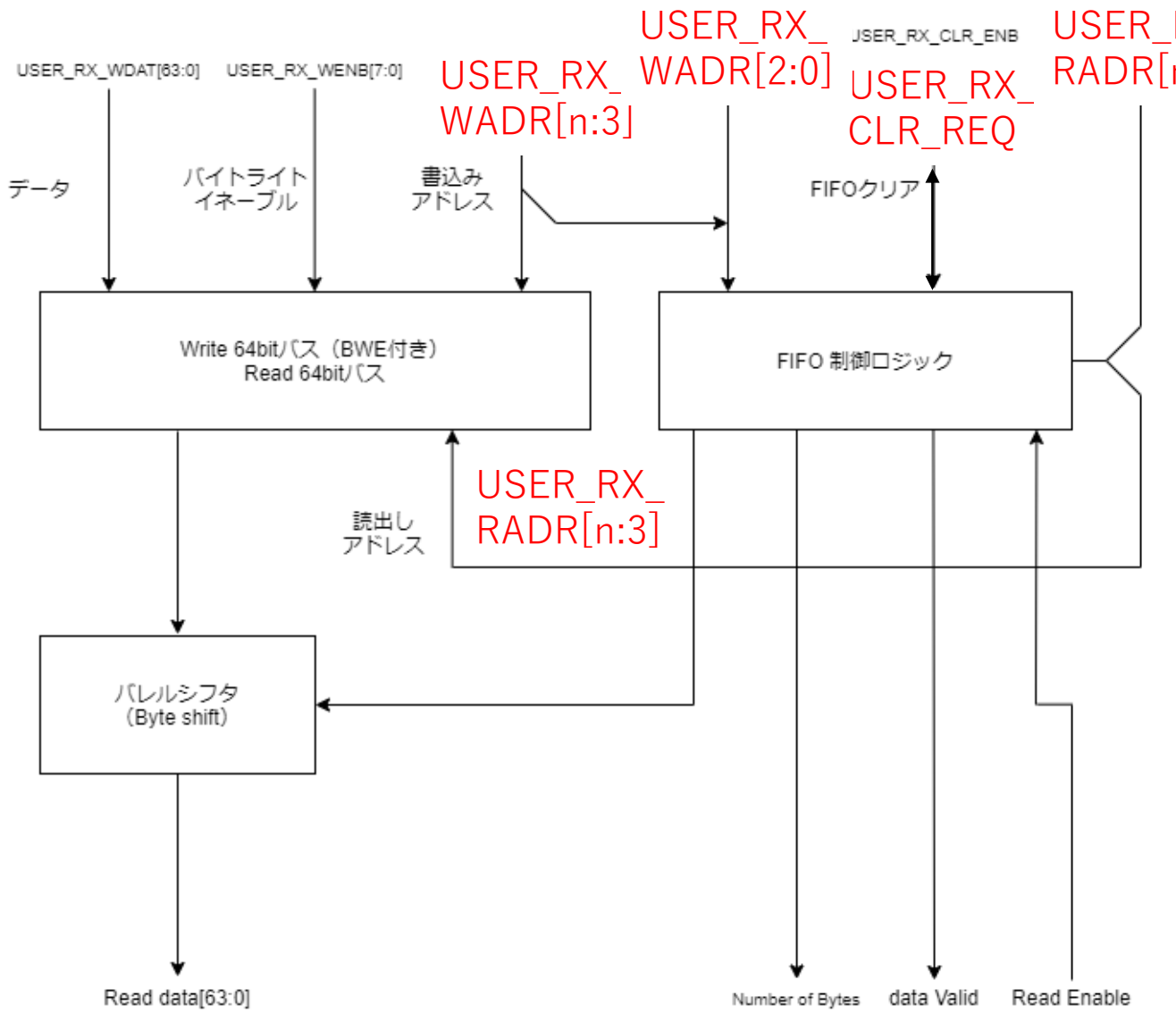
SiTCP-XG　受信　8bit固定バス （RxBufferSize = “Byte"）



※　USER_RX_SIZEには、メモリ容量に応じた固定値を入力すること
※　USER_RX_RADRはUSER_RX_CLR_REQ＝1の時にUSER_RX_WADRをコピーすること
※　SiTCPXGは、USER_RX_CLR_REQ＝1でUSER_RX_WADR[2:0]を3’b000にクリアする
※　Empty = (USER_RX_RADR[n:0] == USER_RX_WADR[n:0])となる

SiTCP-XG　受信　64bit可変バス

（RxBufferSize = "LongLong"）



・USER_RX_SIZEには、メモリ容量に応じた固定値を入力すること

・USER_RX_RADR[2:0]は、出力するByte数に応じて更新

・USER_RX_RADRはUSER_RX_CLR_REQ＝1の時にUSER_RX_WADRをコピーすること

・SiTCPXGは、USER_RX_CLR_REQ＝1でUSER_RX_WADR[2:0]を3’b000にクリアする

・Empty = (USER_RX_RADR[n:0] == USER_RX_WADR[n:0])となる

可変バス時の受信FIFOモジュール（記述例）

ポート宣言

```
module
        SiTCPXG_RX_FIFO_SAMPLE(
                /* [System] */
                input   wire            CLK156M                 // Tx clock
                input   wire            FIFO_REN                // User FIFO read enable
                output  reg     [63:0]  FIFO_D                  // User FIFO data[63:0]
                output  reg     [ 3:0]  FIFO_B                  // Byte length of User FIFO data
                /* [SiTCP-XG I/F] */
                input   wire            SiTCPXG_ESTABLISHED     // Establish of a session
                output  wire    [15:0]  SiTCPXG_RX_SIZE         // Receive buffer size(byte) caution:
                                                                //  Set a value of 4000 or more and
                                                                // (memory size-16) or less
                input   wire            SiTCPXG_RX_CLR_ENB      // Receive buffer Clear Enable
                output  wire            SiTCPXG_RX_CLR_REQ      // Receive buffer Clear Request
                output  wire    [15:0]  SiTCPXG_RX_RADR         // Receive buffer read address in bytes
                                                                // (unused upper bits are set to 0)
                input   wire    [15:0]  SiTCPXG_RX_WADR         // Receive buffer write address in bytes
                                                                //  (lower 3 bits are not connected to memory)
                input   wire    [ 7:0]  SiTCPXG_RX_WENB         // Receive buffer byte write enable
                input   wire    [63:0]  SiTCPXG_RX_WDAT         // Receive buffer write data
        );
```

可変バス時の受信FIFOモジュール（記述例）

Wire reg宣言

```
reg         [11:0]    MEM_RAD          ;
wire        [ 3:0]    MEM_REN          ;
reg         [ 3:0]    P0_MEM_LEN       ;
reg         [ 2:0]    P0_MEM_POS       ;
reg         [ 3:0]    P1_MEM_LEN       ;
reg         [ 2:0]    P1_MEM_POS       ;
wire        [63:0]    MEM_RDT          ;
```

可変バス時の受信FIFOモジュール（記述例）

FIFO制御ロジック

```
assign          SiTCPXG_RX_SIZE[15:0]       = 16'd4000;
assign          SiTCPXG_RX_CLR_REQ          = SiTCPXG_RX_CLR_ENB & ~SiTCPXG_ESTABLISHED;
assign          SiTCPXG_RX_RADR[15:0]       = {4'b0000,MEM_RAD[11:0]};
assign          MEM_REN[3]                  = FIFO_REN?                 (MEM_RAD[11:3] != SiTCPXG_RX_WADR[11:3]):1'b0;
assign          MEM_REN[2:0]                = FIFO_REN?                 SiTCPXG_RX_WADR[2:0]:MEM_RAD[2:0];

always @(posedge CLK156M) begin
                MEM_RAD[ 2:0]       <= SiTCPXG_RX_CLR_REQ    ?SiTCPXG_RX_WADR[2:0]:    (MEM_REN[3]?3 'b000:    MEM_REN[2:0]);
                MEM_RAD[11:3]       <= SiTCPXG_RX_CLR_REQ?    SiTCPXG_RX_WADR[11:3]:  (MEM_RAD[11:3] + {8'd0,MEM_REN[3]});
                P0_MEM_LEN[3]       <= MEM_REN[3] & (MEM_RAD[2:0] == 3'd0);
                P0_MEM_LEN[2:0]     <= (MEM_REN[3]?             3'd0:                   MEM_REN[2:0]) - MEM_RAD[2:0];
                P0_MEM_POS[2:0]     <= MEM_RAD[2:0];
                P1_MEM_LEN[3:0]     <= P0_MEM_LEN[3:0];
                P1_MEM_POS[2:0]     <= P0_MEM_POS[2:0];
        end
```

可変バス時の受信FIFOモジュール（記述例）

FIFOメモリ

```
BRAM_SDP_4K_64B_64B             RX_FIFO_MEM(
                        .clka                   (CLK156M),                      // input wire clka
                        .wea                    (SiTCPXG_RX_WENB[7:0]),         // input wire [7 : 0] wea
                        .addra                  (SiTCPXG_RX_WADR[11:3]),        // input wire [8 : 0] addra
                        .dina                   (SiTCPXG_RX_WDAT[63:0]),        // input wire [63 : 0] dina
                        .clkb                   (CLK156M),                      // input wire clkb
                        .addrb                  (MEM_RAD[11:3]),                // input wire [8 : 0] addrb
                        .doutb                  (MEM_RDT[63:0])                 // output wire [63 : 0] doutb
                );
```

バレルシフタ

```
always @(posedge CLK156M) begin
                        FIFO_D[63:0]            <= (
                                ((P1_MEM_POS[2:0] == 3'd0)?     MEM_RDT[63:0]:                  64'd0)|
                                ((P1_MEM_POS[2:0] == 3'd1)?     {MEM_RDT[55:0],MEM_RDT[ 7:0]}:  64'd0)|
                                ((P1_MEM_POS[2:0] == 3'd2)?     {MEM_RDT[47:0],MEM_RDT[15:0]}:  64'd0)|
                                ((P1_MEM_POS[2:0] == 3'd3)?     {MEM_RDT[39:0],MEM_RDT[23:0]}:  64'd0)|
                                ((P1_MEM_POS[2:0] == 3'd4)?     {MEM_RDT[31:0],MEM_RDT[31:0]}:  64'd0)|
                                ((P1_MEM_POS[2:0] == 3'd5)?     {MEM_RDT[23:0],MEM_RDT[39:0]}:  64'd0)|
                                ((P1_MEM_POS[2:0] == 3'd6)?     {MEM_RDT[15:0],MEM_RDT[47:0]}:  64'd0)|
                                ((P1_MEM_POS[2:0] == 3'd7)?     {MEM_RDT[ 7:0],MEM_RDT[55:0]}:  64'd0)
                        );
                        FIFO_B[3:0]             <= P1_MEM_LEN[3:0];
        end

endmodule
```

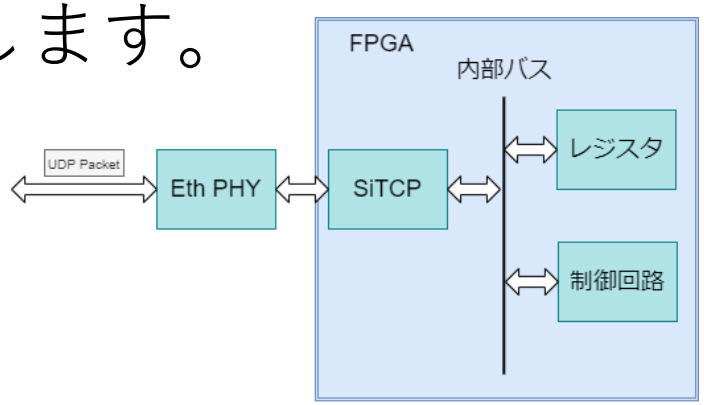
SiTCP-XGの新機能まとめ

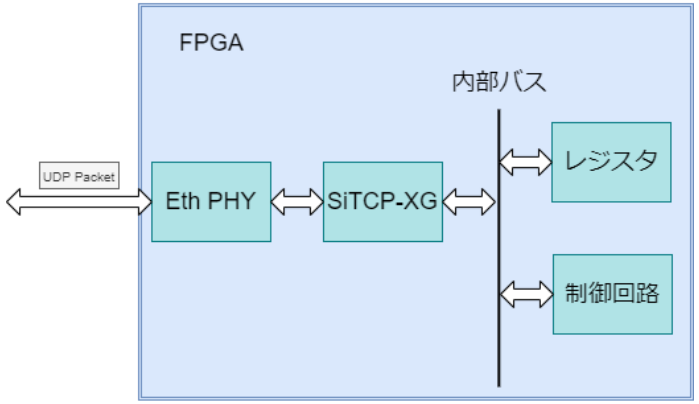
- Window Scalingに対応
  - Control register(0xFFFFFF10)で指定します。（１：有効　0：無効）
  - 現在送信バッファサイズは128KBに対応しております。

- 送信シェーパを実装
  - Transmission rate register (0xFFFFFF40～0xFFFFFF41)で設定可能です。
  - アルゴリズムはleaky bucketでLine Rateです。
  - 受け側に応じて1～10,000 の範囲で設定して下さい。

RBCP

UDPのコマンドでバスマスタとして動作するインタフェースです。SiTCP/SiTCP-XGがバスマスタとなっている内部バスを制御します。



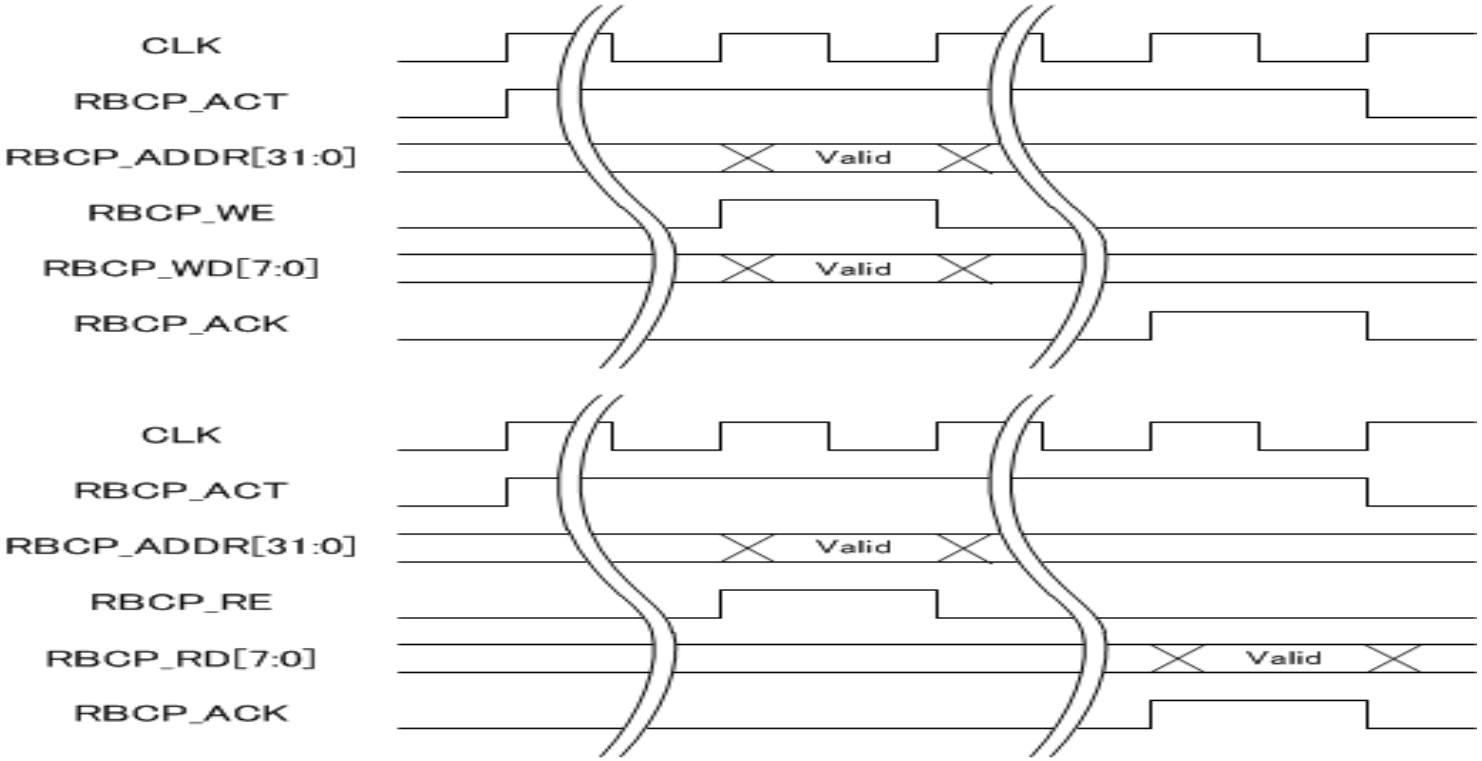


信号名	I/O	説明
CLK	I	SiTCP:システムクロック SiTCP-XG:XGMII クロック
RBCT_ACT	O	バスが動作している事を示す
RBCT_ADDR[31:0]	O	アクセス中のアドレス
RBCT_WE	O	ライトイネーブル
RBCT_WD[7:0]	O	ライトデータ
RBCT_RE	O	リードイネーブル
RBCT_RD[7:0]	I	リードデータ
RBCT_ACK	I	アクセス応答

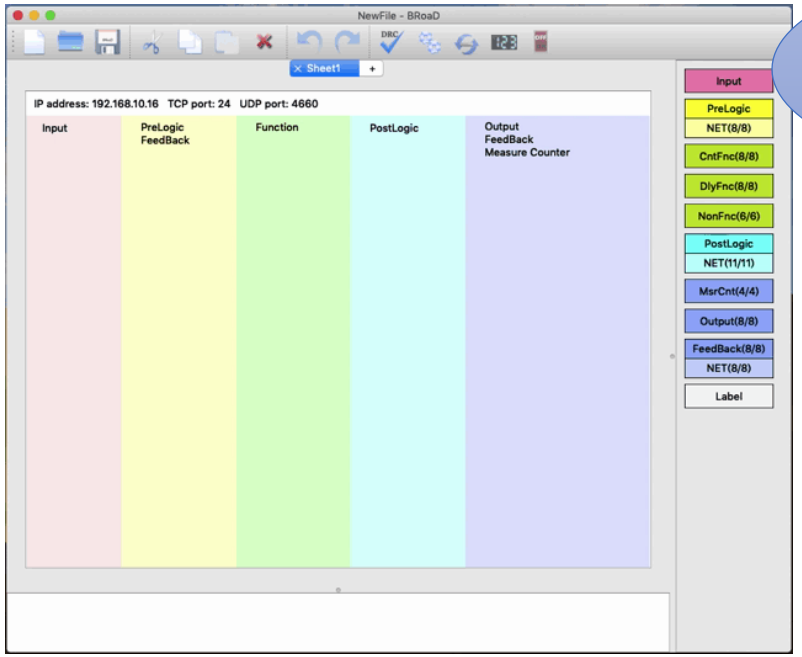
注意！
RBCPアドレス0xFFFF0000から0xFFFFFFFFはSiTCP内部レジスタ用に予約されており使用できません。上記のアドレス以外を使用してください。

RBCPによる低速のメモリマップド・インタフェース

- バスマスタとして、装置内のＩ/Ｏにアクセスできる
- ＴＣＰと独立に機能できる



実装例1　BRoaD/BRoaD III

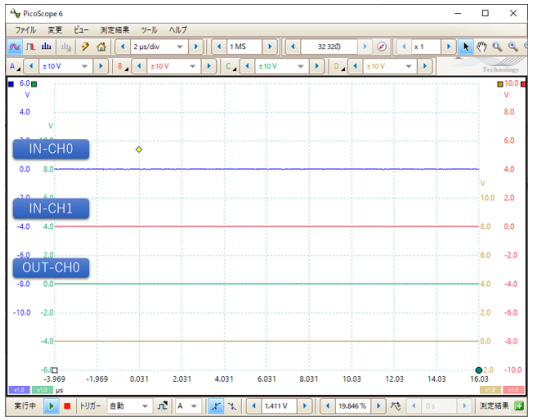


アプリ上で配線を指定→ダウンロード

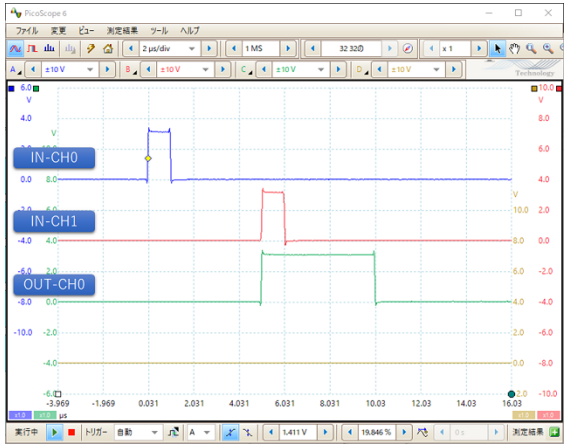
FPGA内に部品を用意

RBCPで書込

FPGA回路は書き換えず真理値表などデータ転送のみで実現、高速です。



ダウンロード
←前　　後→

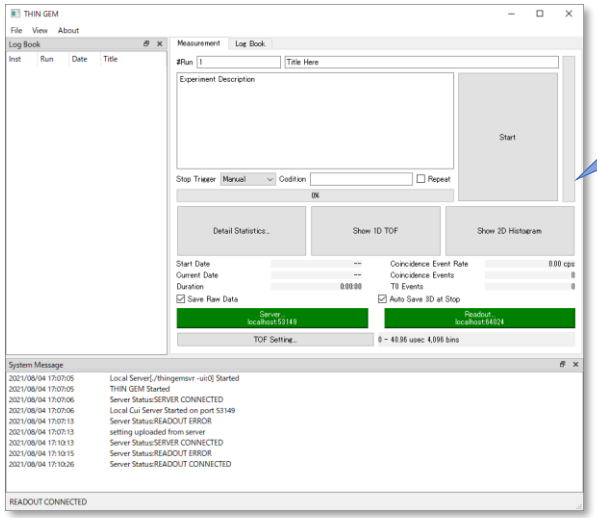


実装例2　Thin-GEM

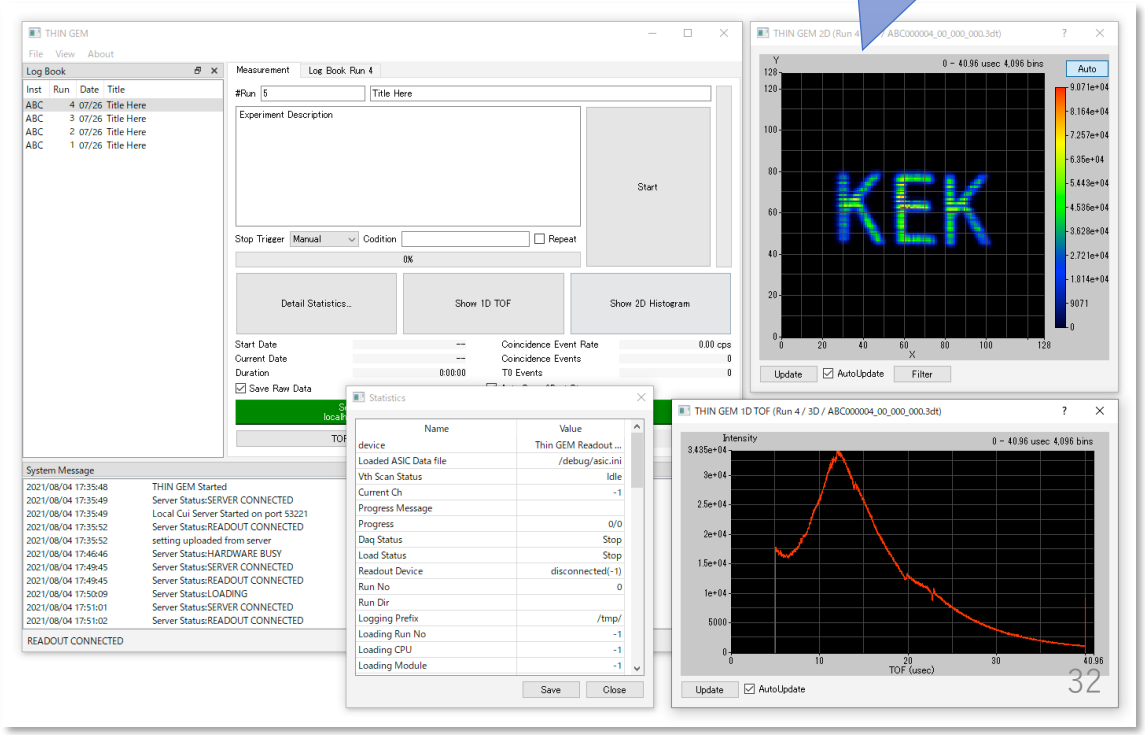
RBCPで書込

TCPで読出

アプリで設定
・IPアドレス
・イベントセッティング
・Vth offset　etc..



測定結果を受信
・2Dヒストグラム
・1D　TOF ヒストグラム



まとめ

Slice LUTs (Slice)：3,741(936)
Slice Registers（CLB Flip-Flops）:4,233(4,233)
RAMB36 （Total Block RAM）:11.5(414kb)

- SiTCP/SiTCP-XGの導入メリット
  - サイズの小ささ
    - SiTCPは3000Slice@Spartan3 程度/SiTCP-XGで1500Slice@Kintex程度です。
  - 実装がかんたん
    - WRAPPERが用意されており、記述が簡単です。
  - 導入コストが安い
    - 普及しているLANの市販のスイッチやNICがそのまま利用できます。
  - 充実したサポート体制
    - ユーザーフォーラム内のディスカッションが活発です。
    - 他ユーザーの返信が無い場合でも「中の人」が答えてくれます。

SiTCP vs SiTCP-XG

	SiTCP	SiTCP-XG
対応FPGA	Spartan6/7 Artix7 Kintex7 Virtex4/5/6/7 Kintex_UltraScale Virtex_UltraScalePlus	Kintex7 Virtex7
対応規格※1	10BASE-T 100BASE-TX 1000BASE-T 1000BASE-SX 1000BASE-LX	10GBASE-SR 10GBASE-LR
PHYインタフェース	MII GMII RGMII※2 SGMII※2	**XGMII**
クロック	送信クロック(GMII_TX_CLK) 受信クロック(GMII_RX_CLK) システムクロック(CLK)	XGMIIクロック（XGMII_CLOCK）※3
クロック周波数	送受信クロックはPHY依存 システムクロックは任意※4	156.25MHz
IEEE802.3X PAUSE対応	対応	未対応
MDIOインタフェース	内蔵	未対応
RBCPインタフェース	ACT/ADDR[31:0]/WD[7:0]/WE/RE /ACK/LOC_RD[7:0]	同左※5
TCP最大性能※6	MII：11.8MByte/s以上 GMII：118MByte/s以上	1.14GByte/s以上
TCPデータインタフェース	8bitバス	8/16/24/32/40/48/56/**64**bitバス※7
TCP window scale option	未対応	対応
送信バッファ	32 k Byte※8	128kByte
受信バッファ※9	最大64kByte	最大64kByte

※1 対応規格は参考です。PHYインタフェースに適合すれば接続できます。
※2 SiTCPコアの外にGMIIから変換する回路を設けることで対応できます。
※3 XGMIIは送受信共通クロックです。
※4 GMIIでは130MHz以上、MIIでは15MHz以上を推奨です。
※5 SiTCPと互換です。SiTCPのクロックはシステムクロックですがSiTCP-XGはXGMIIクロックとなります。
※6 相手側の性能が十分あり、通信遅延時間が十分小さい場合の値です。
※7 最大送信性能を出すためには64bitバスが必要です。
※8 標準のコアの場合です。理論最大では64kbyteです。
※9 受信バッファはコアに含まれません

SiTCP/SiTCP-XG関連サイト

- SiTCPホーム　[内田智久博士作成・当社保守]
  https://www.sitcp.net/
  - SiTCPの開発経緯や概要・導入にあたっての資料など
- SiTCPについて
  https://www.bbtech.co.jp/sitcp/
  - SiTCPについての概要とリンク集
- SiTCP 関連 ダウンロード
  https://www.bbtech.co.jp/download-files/sitcp/index.html
  - 関連ドキュメントやツールなどのダウンロード
- SiTCP Forum
  https://www.bbtech.co.jp/sitcp-forum/#/discussions
  - SiTCP/SiTCP-XGに関する入門者用の参考記事
  - 質問と回答や様々なディスカッション
  - 障害情報やトラブルシューティング
- BeeBeans Technologies GitHub
  https://github.com/BeeBeansTechnologies
  - 各種FPGA用のNetlist
  - 評価基板用のサンプルコード
  - SiTCPデバイス用ソフトウェア開発サポートライブラリ