

# MITOUJTAG

## BASIC

---

## JTAG ロジックアナライザ

## 操作マニュアル

このマニュアルは、MITOUJTAG の機能である JTAG ロジックアナライザの使用方法について記載されています。実際にご利用いただく前に、ぜひお読みください。

# 目次

<b>第1章</b>	<b>JTAGロジックアナライザの概要</b>	<b>2</b>
1.1	JTAGロジックアナライザとは	2
1.2	起動までの準備	3
1.3	起動方法	4
<b>第2章</b>	<b>バウンダリスキャン・モードでの使用</b>	<b>5</b>
2.1	基本操作	5
2.2	信号名の設定	6
2.3	信号のバス化	9
2.4	カーソルと時間軸機能	12
2.5	波形の保存と復元	14
<b>第3章</b>	<b>BLOGANAモードでの使用</b>	<b>15</b>
3.1	BLOGANAモードとは	15
3.2	ロジアナIPコアの挿入	18
3.3	BLOGANAモジュールの信号設定方法	21
3.4	トリガ機能	24



## 通常モードと BLOGANA モードの比較

通常モードと BLOGANA モードの特徴を表 1 に示します。ご使用の目的に応じて使い分けてください。

表 1 JTAG ロジックアナライザの各動作モードの特徴

	通常モード	BLOGANA モード
サンプリング速度	約 2000 サンプリング/秒	数 100M サンプリング/秒
観測可能信号数	無制限 (数 1000 個) I/O 端子の信号	最大 72 本 FPGA 内部信号
データ長	無制限(10 万サンプル程度)	512 サンプル、または 1024 サンプル
対象デバイス	各社 CPLD/FPGA/CPU メーカー、品種は問わず	XILINX FPGA (Spartan3/3E/3A/3AN Virtex 2,2Pro,4,5)
システムクロック	不要	必要
ユーザリソース	使用せず	ブロック RAM と BSCAN および、 コンポーネント(USER1 命令)を使用
FPGA のコンフィ ギュレーション	不要	必要

## 1.2 起動までの準備

JTAG ロジックアナライザを実行するためには、そのデバイスの BSDL ファイルが必要です。パッケージファイルおよび回路の設計データ(ネットリスト等)は必要ありません。

また、信号名を定義するため、その FPGA や CPLD を作成する時に使用した UCF ファイルまたは QSF ファイルをご用意ください。BLOGANA モードで使用する場合には、ソース VHDL ファイルも必要です。

表 2 JTAG ロジックアナライザの使用に必要なファイル

通常モード	BLOGANA モード
UCF ファイル(XILINX FPGA/CPLD の場合) QSF ファイル(ALTERA FPGA/CPLD の場合) BSDL ファイル	ソース VHDL ファイルまたは Verilog ファイル

FPGA や CPLD デバイスで使用する場合には、JTAG ロジックアナライザの起動前に MITOUJTAG のメイン画面で、JTAG デバイスの絵の上で右クリックをし、「ピン定義の読み込み」を実行して、UCF ファイルまたは QSF ファイルを読み込んでください。

これで、JTAG ロジックアナライザ使用時に信号名が表示されるようになります。



図 3 JTAG ロジックアナライザ起動前にピン定義を読み込む

### 1.3 起動方法

JTAG ロジックアナライザを起動するには、メイン画面上で、ツールバーの中の



ボタンを押します。

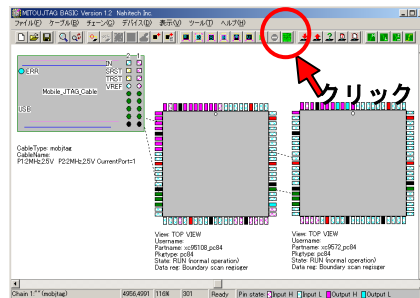


図 4 JTAG ロジックアナライザの起動方法

## 第2章 バウンダリスキャン・モードでの使用

### 2.1 基本操作

#### サンプリングの開始と停止







サンプリングの開始と停止を行うには、次の表に示す 4 つのボタンを用います。 ボタンを押して信号のおおよその形をべた後、 ボタンで高速にキャプチャするという手順で操作するとよいでしょう。

表 3 動作の開始・停止ボタン

アイコン	説明
	FPGA に内蔵されたブロック RAM を使用するモード (BLOGANA モード) へ移行します。
	高速にサンプリングします。(キャプチャ・モード)
	サンプリングしながら表示を同時に行います。(ウォッチ・モード)
	サンプリングを停止します。

#### 信号の状態の読み方

表示された波形のうち、赤い線で描かれている信号は、その端子から信号が出力されていることを表しています。また、緑の線は入力を、黄色の線は衝突状態、灰色の線はハイインピーダンス状態になっていることを表しています。

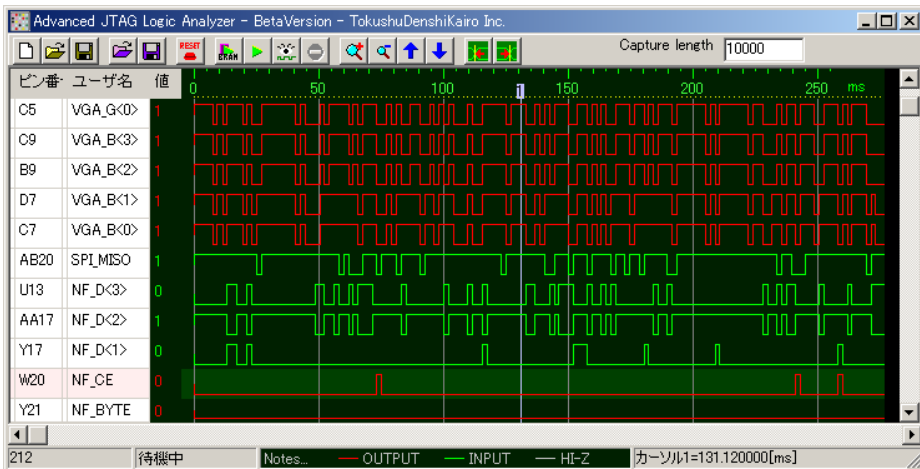


図 5 赤の線は出力信号、緑の線は入力信号を表す

## データ量の設定

デフォルトの設定では、10,000 ポイントのデータをサンプリングすると、キャプチャあるいはウォッチ動作を終了します。取得されるデータの量を変更するには、図 6 に示したボックス内の値を変更します。



図 6 データ量の設定

多量のデータを取得すると非常に多くのメモリを消費しますのでご注意ください。

## 2.2 信号名の設定

### ピン名とユーザ名

画面の「ピン名」と書かれた欄には、BSDL ファイル中に記載された信号名が表示されます。一般に、CPU など用途が決まったデバイスの場合には、個々の I/O 端子に A0 や RD などの意味を持った名前が付いているので、「ピン名」で I/O 端子の意味が把握できます。

それに対して FPGA や CPLD では端子の機能はユーザが決めるので、デフォルトの「ピン名」は意味を持ちません。「ユーザ名」を設定することができるようになっています。

「ユーザ名」を設定するには、MITOUJTAG のメイン画面で、FPGA を設計した際のピン定義ファイル (UCF ファイルまたは QSF ファイル、4 ページ参照) を読み込んでください。もしくは、目的の信号のマス目にカーソルを合わせ、そのままキーボードから入力してください。



図 7 信号の「ユーザ名」

### 信号名の調整

JTAG ロジックアナライザは数百本の信号を同時に観測することができますが、起動直後にはデバイスの持つすべての信号が表示されてしまいます。このため、目的の信号が探しにくく感じる場合があります。

目的の信号を観測しやすくするには以下に示す方法で、表示する信号を並べ替えたり、バス化してください。

## 信号の並べ替え（自動整列）

一番上のマス目（「ピン番号」「ピン名」「ユーザ名」と書かれた灰色のセル）をクリックすると、信号名を①ピン番号、②BSDL ファイル中の信号名、③ユーザ設定信号名の降順あるいは昇順にすばやく整列させることができます。

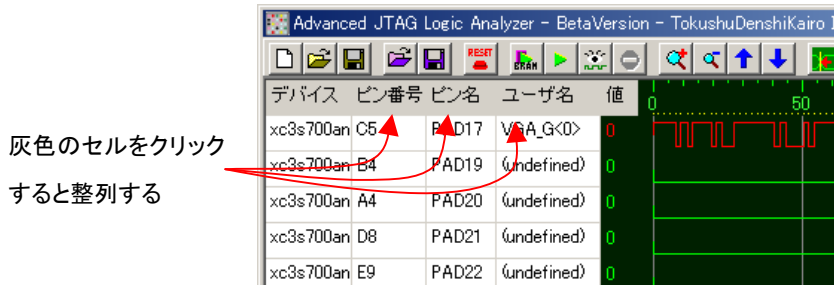


図 8 信号を「ピン番号」「ピン名」「ユーザ名」で整列すると見やすくなる

信号名が表示される場所の「ピン番号」の欄をクリックすると、ピン番号順に整列され、「ピン名」欄をクリックすると BSDL ファイルに記載された信号名順に整列され、ユーザ名をクリックすると、ユーザの定義した信号名 (UCF ファイルまたは QSF ファイル中で指定) 順に整列されます。このようにして見やすい順序に並べ替えてください。

## 信号の並べ替え（手動調整）

以下の手順で、信号を 1 つずつ上下に動かし、目的の順序に並べ替えることができます。

① 右の図のように、SIG\_1、SIG\_2、SIG\_3、SIG\_4、SIG\_5、SIG\_X の合計 6 本の信号が観測されている場合を想定します。

ここで、SIG\_X を観測対象から外し、SIG\_3 と SIG\_4 の順番を入れ替えたいとします。

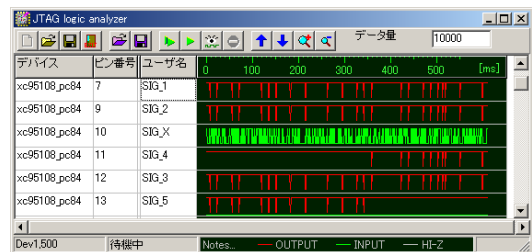


図 9 信号の並べ替え手順 1

② まず、マウスまたはカーソルキーを操作して SIG\_X の行にカーソルをあわせませす。ここで、DEL キーを押すと、SIG\_X 信号が観測対象から外れます。

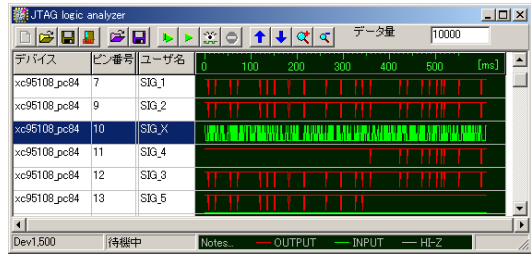


図 10 信号の並べ替え手順 2

③ 次に、マウスまたはカーソルキーを操作して SIG\_4 の行にカーソルをあわせませす。ここで、ツールバーの ↓ ボタンを押すと、SIG\_4 信号が一つ下に移動し、SIG\_3 と順番が入れ替わります。

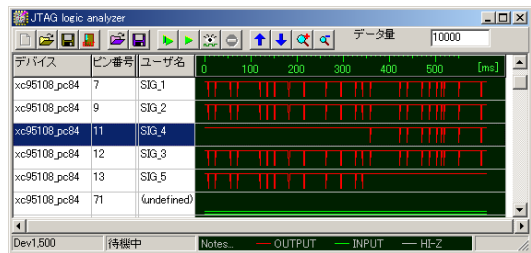


図 11 信号の並べ替え手順 3

④ ↑ と ↓ ボタン、および DEL キーを利用して、観測したい信号だけを順番に並べませす。

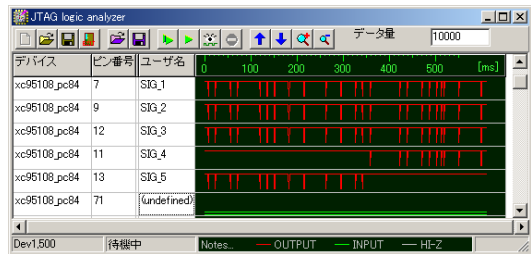


図 12 信号の並べ替え手順 4

## 2.3 信号のバス化

JTAG ロジックアナライザでは、複数の信号を束ねてバスとして扱い、表示を 16 進や 10 進数の読みやすい形にすることができます。

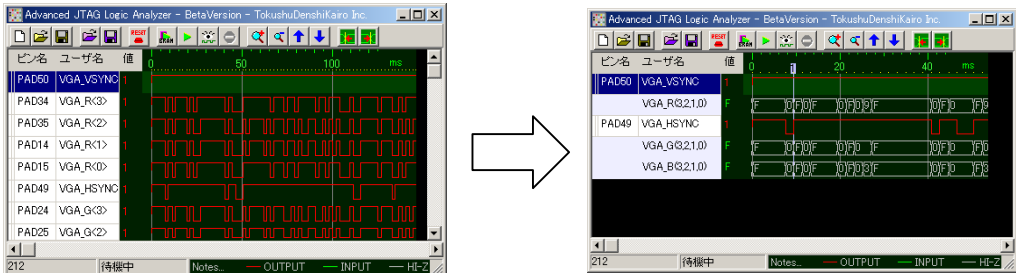


図 13 信号のバス化によって波形が見やすくなる

信号をバス化するには、以下の手順で行います。

JTAG ロジックアナライザを起動した状態では、右の図のように右の図のように、信号の順序は整列されておらず、すべての信号は 1 ビットの単体の信号として見えています。

このような場合、最初に「ピン名」または「ユーザ名」のマスをダブルクリックして、昇順または降順に整列させます。



図 14 信号のバス化手順 1

例えば右の図で、LED<7>~LED<0>と表示された信号をバス化したいとします。このような場合、まず、一番上に表示されている LED<7>をマウスで左クリックします。

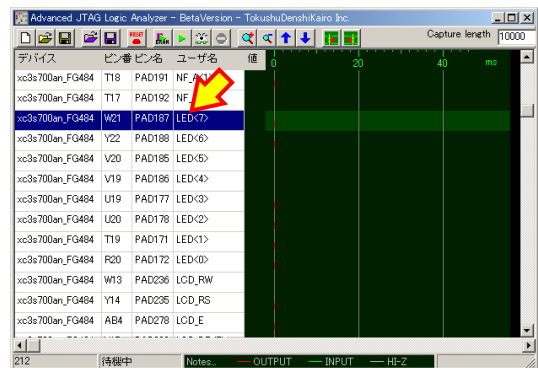


図 15 信号のバス化手順 2

次に、LED<7>を、SHIFT キーを押しながらマウスで左クリックします。

なお、CTRL キーを押しながら信号名をクリックすると、SHIFT キーのような範囲選択ではなく、1個ずつ信号を選択することができます。

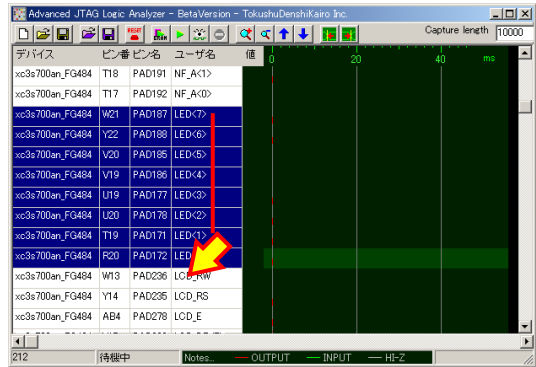


図 16 信号のバス化手順 3

バス化したい信号を選択したら、SHIFT キーまたは CTRL キーを押しながら右クリックしてください。右の図のようなプルダウンメニューが開きます。

このプルダウンメニューで、「結合」を選択すると、選択された信号がバスとして結合されます。「不可視にする」を選択すると、その信号が見えないように削除されます。

削除された信号は「すべての信号を表示」で再び観測することができます。

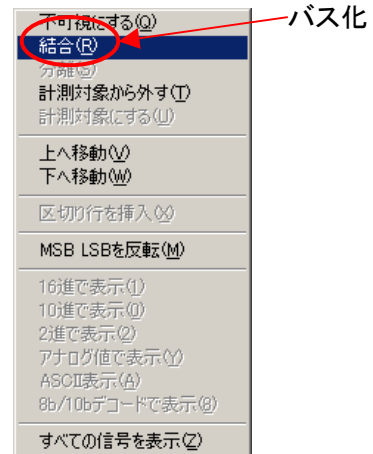


図 17 信号のバス化手順 4

信号を結合すると、右の図のようにバスとして表示されます。

デフォルトでは2進数で表示するようになっています。

バス化された信号の表示が正しくないと感じた場合は「MSBとLSBを反転」を選択してください。

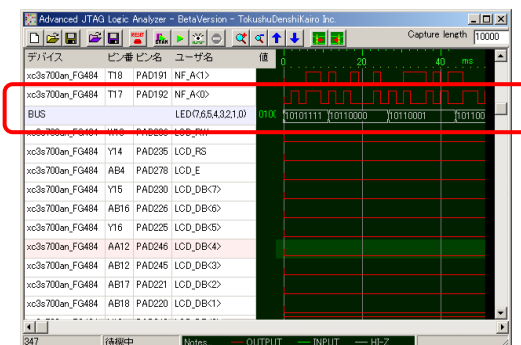


図 18 信号のバス化手順 5

ここで再度、信号名の上で右クリックをしてください。右の図のプルダウンメニューが開きます。

16進で表示させたい場合は「16進で表示」を、10進で表示させたい場合は「10進で表示」を選択してください。

通信の解析などで、8b/10b デコードで表示させたい場合は「8b/10b デコードで表示」を選択してください。

8b/10b デコードで表示する場合、9ビット分の信号をバス化し、「K b7 b6 b5 b4 b3 b2 b1 b0」の順に並ぶようにしてください。

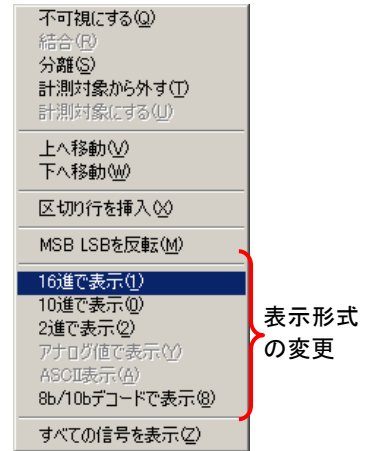


図 19 信号のバス化手順 6

## 2.4 カーソルと時間軸機能

### カーソル機能とは

カーソル機能とは、波形の任意の時刻に目印を設定する機能です。カーソルは 2 ヶ所まで設定することができます。2 つのカーソルをそれぞれカーソル1、カーソル2と呼びます。

### カーソルの設定方法

波形ウィンドウ上でマウスクリックをすると、その時刻にカーソルが設定されます。

左クリックでカーソル1が、右クリックでカーソル2が設定されます。設定されたカーソルはもう一度マウスクリックすると削除されます。

また、カーソルを設定すると、画面右下のステータスバーに各カーソルが置かれた点の時刻と、カーソル間の差分の時間が表示されます。

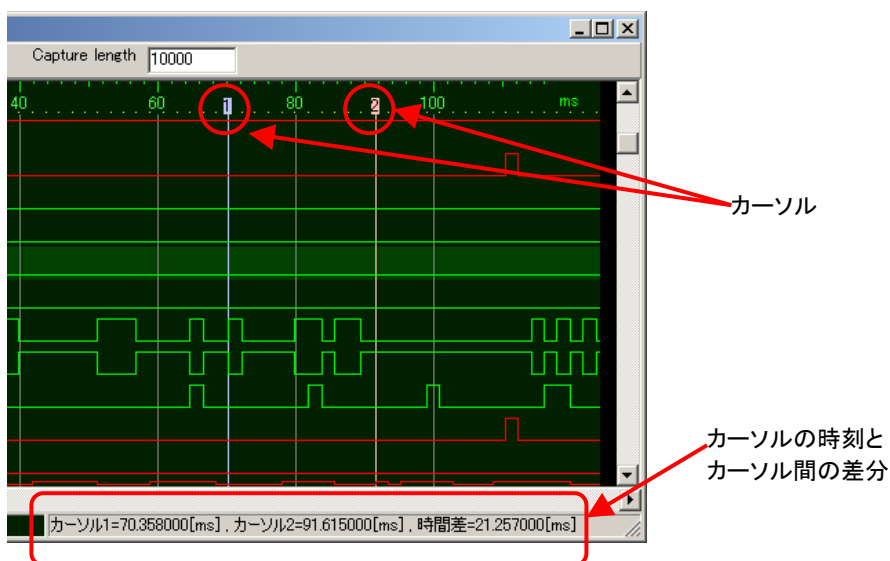


図 20 カーソル機能

## サンプリングタイミングの表示

画面上の時間軸表示の部分に、サンプリングされたタイミングが黄色い点で図示されます。

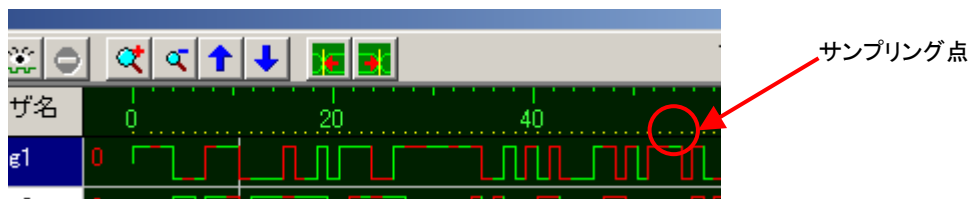


図 21 サンプリング点の表示

## 信号の遷移点の検索

下の図に示したボタンをクリックすると、現在のカーソルの位置から前または後ろに、信号の遷移するタイミングを検索することができます。



信号の遷移を左側に向かって検索

信号の遷移を右側に向かって検索

## 時間軸の拡大と縮小

キャプチャされた波形を、時間軸に対して拡大・縮小、あるいはスクロールして表示することができます。拡大・縮小を行うには、次の表に示したボタンを押します。

表 5 時間軸の拡大と縮小ボタン

アイコン	説明
	時間軸を拡大して表示します。
	時間軸を縮小して表示します。

時間軸方向にスクロールして表示するには、画面下にあるスクロールバーを操作します。

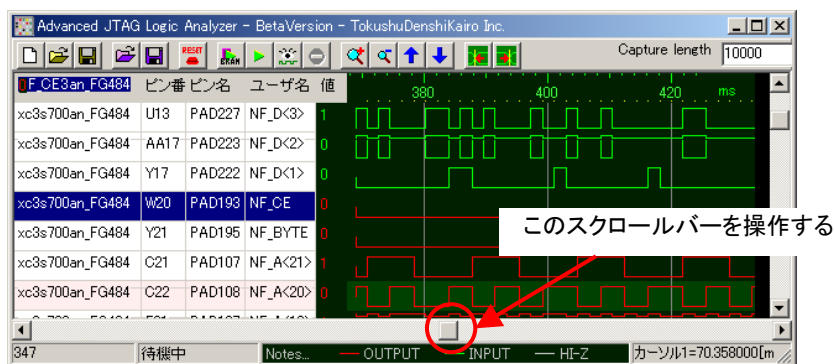


図 22 時間軸のスクロール方法

## 2.5 波形の保存と復元





信号名やその順序などの設定と、取得されたデータをファイルに保存し、いつでも好きなときに読み出すことができます。

保存した波形は、無償のビューワで参照することも可能です。

例えば、MITOUJTAG を使ってサンプリングした波形を、他社や社内の他の部署に送って、無償のビューワで見てもらおうといった使い方が可能になります。波形を取得するエンジニアと、解析するエンジニアを分けることができるようになり、デバッグの可能性が広がります。

状態の保存と復元を行うためには、次の表に示したボタンをクリックしてください。

表 6 ファイルの保存と復元に関するボタン

アイコン	説明
	保存された波形データと、信号の設定をファイルから読み込みます。
	波形データと信号の設定をファイルに保存します。
	信号の設定をファイルから読み込みます。
	信号の設定をファイルに保存します。

# 第 3 章 BLOGANA モードでの使用

## 3.1 BLOGANA モードとは

### BLOGANA モードとは

BlockRAM モードのロジックアナライザ(BlockRAM Logic Analyzer。以後 BLOGANA と略す)というのは、サンプリングした波形データを XILINX FPGA 内の内蔵 RAM に溜め込み、その RAM の内容を JTAG 経由で読み出して表示する機能です。FPGA の最高動作速度に迫る速度で動作します。例えば、Spartan3 ならば 200MHz 程度、Virtex4/5 ならば 500MHz 程度まで動作します。

通常モード(バウンダリ・スキャン・モード)の JTAG ロジックアナライザでは不可能だった、数百 MHz でのサンプリングが可能になります。BLOGANA モードは、ステートマシンの動作や周辺チップとのインタフェースを細かく調べるのに適しています。

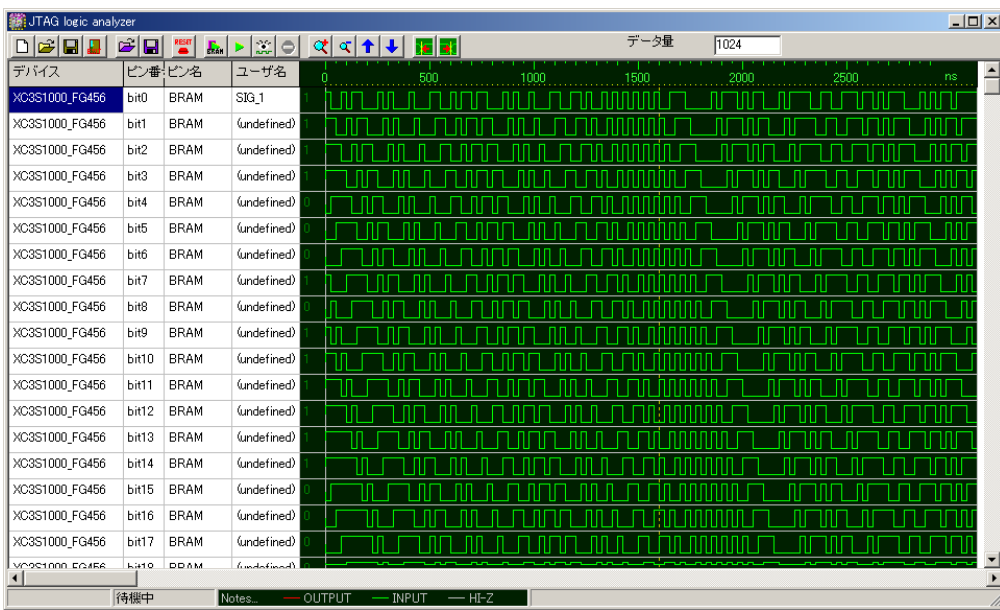



図 23 BLOGANA モードの JTAG ロジックアナライザ



BLOGANA モードは、XILINX 社の FPGA 専用です。  
XILINX 製の CPLD デバイスや、ALTERA 製デバイス、その他のメーカーの FPGA/CPLD または CPU などではご利用いただけません。

## BLOGANA モードの制約

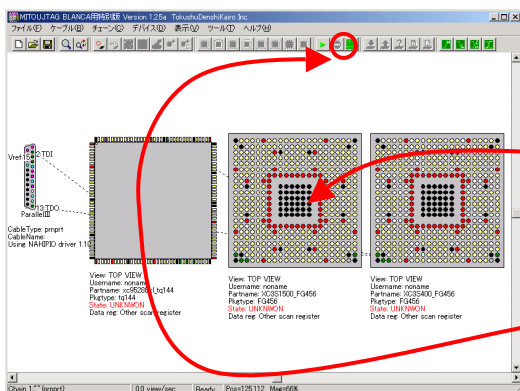
BLOGANA モードを使うには、FPGA にロジアナ用の IP コア(以下、BLOGANA モジュール)を埋め込む必要があります。また、動作にはクロックが必要です。

BLOGANA モードでは同時にサンプリングできるビット幅や最大サンプリング長は、使用する BlockRAM の数に左右されます。MITOUJTAG 1.5 では BlockRAM を最高 4 つまで使用して、最大で 1024 ポイントのサンプリングが可能です。

ビット幅は最大 72 ビットとなります。BlockRAM の数の制限は今後のバージョンアップで改善される可能性があるので、こまめにサービスパックダウンロードページへアクセスしてください。

## 起動の方法

BLOGANA モードを使用するためには、まず目的の FPGA に BLOGANA モジュールを組み込んで論理合成し、コンフィギュレーションします。その FPGA を MITOUJTAG で認識させ、通常モードの JTAG ロジックアナライザを起動します。




① 通常の手順で MITOUJTAG を起動する。

② FPGA には BLOGANA モジュールをインプリメントしたデザインファイルでコンフィギュレーションしておく。

③ ロジアナ起動ボタンを押す。

図 24 BLOGANA を起動するまでの手順

JTAG ロジックアナライザが起動したら、 ボタンを押します。FPGA 内に組み込んだ BLOGANA モジュールが MITOUJTAG から認識されると、JTAG ロジックアナライザは BLOGANA モードに移行し、次の図 25 に示すダイアログが開きます。

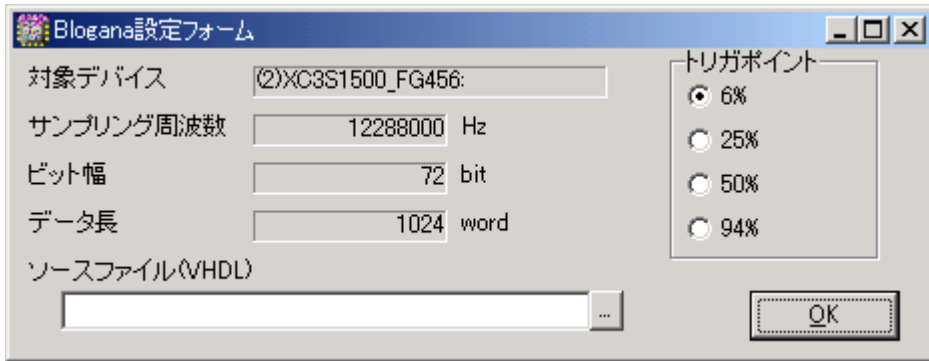


図 25 BLOGANA モードの設定ダイアログ

「ソースファイル」と書かれた入力ボックスには、BLOGANA モジュールを組み込んだ VHDL のソースファイルを指定し、OK ボタンを押します。

もし、BLOGANA モジュールを組み込んだデバイスが見つからない場合には、次のダイアログが表示されますので、ケーブルの接続や書き込まれたデザインファイルを確認してください。

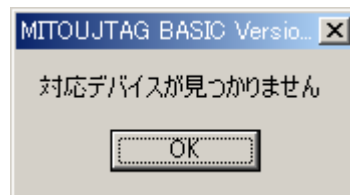


図 26 BLOGANA モジュールを実装した FPGA が見つからない場合

## 3.2 ロジアナ IP コアの挿入

### BLOGANA モジュールのソースファイルの所在

BLOGANA を使うためには XILINX ISE で論理合成する際に、プロジェクトファイルに BLOGANA モジュールを追加する必要があります。MITOUJTAG をインストールすると、VHDL 版 BLOGANA モジュールのソースが

```
"C:\Program Files\TokudenKairo\Mitoutag\vhdl\blogana.vhdl"
```

にインストールされていますので、これらのファイルをユーザプロジェクトのフォルダにコピーしてください。また、このファイルは、Windows のスタートメニューから、「プログラム→MITOUJTAG→関連 IP」を選択してもアクセスすることができます。

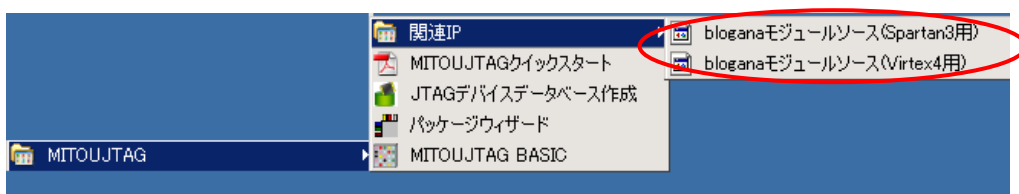


図 27 スタートメニューから BLOGANA ソースへアクセスする

### BLOGANA コアを ISE プロジェクトへ追加する

BLOGANA モジュールを、XILINX FPGA の合成プロセスに組み込むには、ISE を起動したらソース・ツリーで右クリックをして、プルダウンメニューから「Add Source」を実行します。

その後を開くダイアログで、BLOGANA モジュールのソースファイル "blogana.vhdl" を選択し、ISE プロジェクトに追加します。

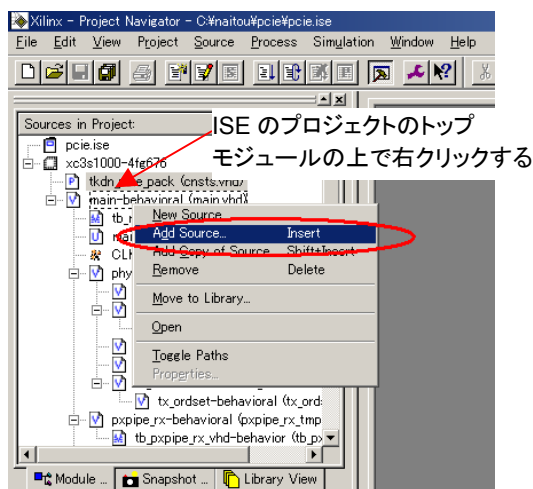


図 28 ISE プロジェクトへの BLOGANA モジュールの追加

## コンポーネント宣言の追加とインスタシエート

お客様が作成された VHDL ソースにおいて、architecture 文に次のようにコンポーネント宣言と signal 宣言を追加します。

signal 文での信号名は、必ず下記のとおりになるようにしてください。

### リスト 1 architecture 文への BLOGANA モジュールの追加

```
component blogana is
  Port ( CLK      : in std_logic;
        TRIG     : in std_logic;
        DIN      : in std_logic_vector (71 downto 0); -- 必ず 72 本
        SAMP_FREQ : in integer range 0 to 2147483647; -- サンプリング周波数
        WIDTH72   : in std_logic;                    -- 1:72 ビット 0:36 ビット
        LENGTH1024 : in std_logic;                   -- 1:1024 ワード 0:512 ワード
        BUSY      : out std_logic;                   -- 必ずどこかに出力すること
  );
end component;

signal BLOGANA_TRIG : std_logic;
signal BLOGANA_DIN  : std_logic_vector (71 downto 0);
signal BLOGANA_BUSY : std_logic;
```

次にソースファイルの begin 文の後ろ、いわゆる VHDL の本文を記述する部分に、port map 文を使って BLOGANA モジュールをインスタシエートします。BLOGANA はデバッグ用の回路なので、VHDL の最後の部分に書き加えると見やすいでしょう。

### リスト 2 BLOGANA モジュールのインスタシエート

```
INST_BLOGANA : blogana port map
( CLK      => CLK,
  TRIG     => BLOGANA_TRIG,
  DIN      => BLOGANA_DIN,
  SAMP_FREQ => 125000000,
  WIDTH72  => '1',
  LENGTH1024 => '1',
  BUSY     => BLOGANA_BUSY
);
```

リスト1のCLK入力にはサンプリングクロックをつなぎます。DINとTRIGはキャプチャする信号とトリガ入力をつなぎます。

SAMP\_FREQにはクロックの周波数を指定します。WIDTH72とLENGTH1024はサンプリングの幅と長さを指定します。SAMP\_FREQとWIDTH72とLENGTH1024は定数を指定してください。



SAMP\_FREQ や WIDTH72、LENGTH1024 には必ず定数を指定してください。BLOGANA モジュール内の冗長な回路が論理圧縮されます。

### **BLOGANA の BUSY 信号は FPGA の外に出す**

BUSYはBLOGANAの動作中に'1'になる信号です。この信号はテストポイントやLEDを通じて、必ずFPGAの外に出力される信号に反映させるようにしてください。BLOGANAモジュールは基本的に入力ポートしか必要ないコンポーネントであるため、ISEの論理合成プログラム(XST)は出力ポートがない入力だけのモジュールと判断して、無条件に圧縮して削除してしまいます。実際にはBLOGANAモジュールの内部に置かれたBSCAN\_SPARTAN3というプリミティブを通じて、JTAGポートからFPGAの外に信号を出力しているのですが、XSTはそれを理解しません。

この問題を避けるために、BLOGANAからはダミーのBUSY出力信号を取り出しています。このBLOGANA\_BUSY信号が削除されないようにしなければなりません。例えば、FPGAの外に出るテストポイントの信号を用意し、

```
TP <= BLOGANA_BUSY xor (misc(1) and ...)
```

というように、他の信号とのXORやANDを取って出力しても良いでしょう。



BUSY信号をFPGAの外に出力しないと、ISEの論理合成プログラムはBLOGANAモジュールを削除してしまいます。

### 3.3 BLOGANA モジュールの信号設定方法

#### BLOGANA の信号

BLOGANA モジュールの入出力仕様を次の表に示します。

表 7 BLOGANA モジュールの入出力信号

名前	型	方向	機能
CLK	STD_LOGIC	IN	サンプリングクロック。
TRIG	STD_LOGIC	IN	動作開始トリガ。'1'で動作開始。
DIN	STD_LOGIC_VECTOR	IN	解析されるデータ。
WIDTH72	STD_LOGIC	IN	'0'なら 36 ビットモード。 '1'なら 72 ビットモード。
LENGTH1024	INTEGER	IN	'0'ならデータ長は 512。 '0'ならデータ長は 1024。
BUSY	STD_LOGIC	OUT	計測中は'1'が出力される。 この信号は何らかの形で FPGA 外に取り出すこと。

“SAMP\_FREQ”信号は、回路の動作には影響を与えませんが、この値は FPGA 内にハードコーディングされて、BLOGANA モジュールの認識の際に読み出されます。したがって、SAMP\_FREQ には、CLK に入力するクロックの周波数を正しく入力してください。

“TRIG”信号は、この信号が'0'→'1'に遷移するとBLOGANAがキャプチャ動作を開始します。BLOGANA にはプログラマブルなトリガ機能は内蔵されていないので、高度なトリガ機能をご利用いただくには、FPGA 内のロジックを利用してユーザレベルでトリガ信号を生成してください。(3.4 節を参照)

“DIN”信号には、キャプチャされる信号を入力します。この信号は常に 72 ビット幅です。36 ビットモードで使用する場合は、下位 36 ビットが使用されます。

“WIDTH72”信号と“LENGTH1024”信号はビット幅とキャプチャされるデータの長さを指定します。

BLOGANA モジュールを VHDL ソースに埋め込む例を次のリスト 3 に示します。このソースファイルを参照して、BLOGANA モジュールをお客様のソースファイルの下位モジュールとして組み込んでください。

### リスト 3 Blogana コンポーネントの使用例

```

architecture Behavioral of sample is
  component blogana is
    Port ( CLK          : in std_logic;
          TRIG         : in std_logic;
          DIN          : in std_logic_vector(71 downto 0); -- 必ず 72 本
          SAMP_FREQ    : in integer range 0 to 2147483647; -- サンプリング周波数
          WIDTH72      : in std_logic;                    -- 1:72 ビット 0:36 ビット
          LENGTH1024   : in std_logic;                    -- 1:1024 ワード 0:512 ワード
          BUSY         : out std_logic                    -- 必ずどこかに出力すること
    );
  end component;
  signal BLOGANA_TRIG : std_logic;
  signal BLOGANA_DIN  : std_logic_vector (71 downto 0);
  signal BLOGANA_BUSY : std_logic;

  ... (中略) ...   その他の信号の定義

begin

  ... (中略) ...   VHDL によるロジックの記述

-- BLOGANA モジュールのインスタンス化
  INST_BLOGANA : blogana port map
  (  CLK      => CLK,
    TRIG      => BLOGANA_TRIG,
    DIN       => BLOGANA_DIN,
    SAMP_FREQ => 125000000,
    WIDTH72   => '1',
    LENGTH1024 => '1',
    BUSY      => BLOGANA_BUSY
  );
-- BLOGANA で見たい信号を接続する
  BLOGANA_TRIG <= SYS_AccStart;
  BLOGANA_DIN(0) <= SYS_Reset;
  BLOGANA_DIN(1) <= SYS_ReadWrite;
  BLOGANA_DIN(2) <= SYS_AccStart;
  BLOGANA_DIN(3) <= SYS_AccReady;
  BLOGANA_DIN(4) <= SYS_Interrupt;
  BLOGANA_DIN(5) <= SYS_nNMI;
  BLOGANA_DIN(6) <= SD_nCS_node;
  BLOGANA_DIN(7) <= SD_nRAS_node;
  BLOGANA_DIN(8) <= SD_nCAS_node;
  BLOGANA_DIN(9) <= SD_nWE_node;
  BLOGANA_DIN(25 downto 10) <= SYS_Address(31 downto 16);
  BLOGANA_DIN(29 downto 26) <= SYS_ByteEnable(3 downto 0);
end Behavioral;

```

## BLOGANA\_DIN バスについて

前頁のリスト 3 において、SYS\_ReadWrite や SYS\_Address といった信号は FPGA 内部の信号です。これらの信号をリスト 3 の例のように BLOGANA\_DIN という信号を経由して BLOGANA モジュールに接続します。

つまり、観たい信号を BLOGANA\_DIN の適切なビットに割り当てて、BLOGANA\_DIN 全体を BLOGANA モジュールに接続します。

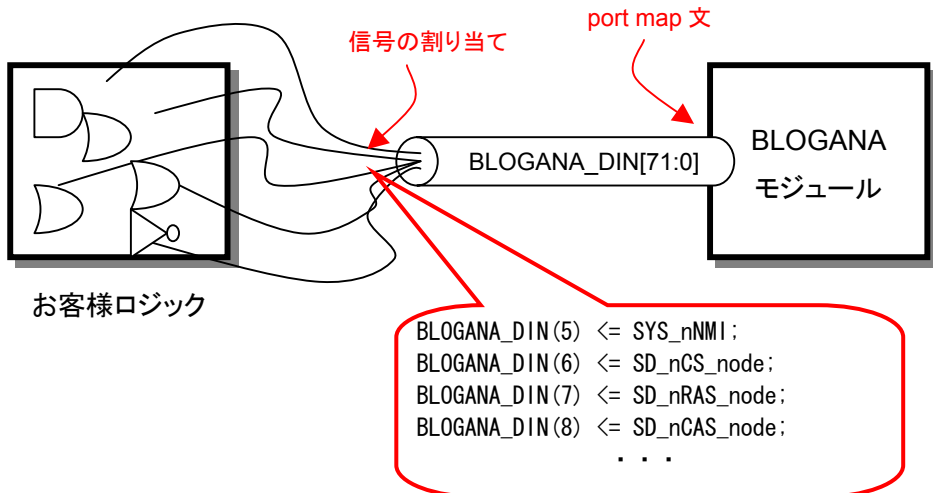


図 29 BLOGANA\_DIN バスを経由して BLOGANA モジュールへ接続する

上の図 29 のように BLOGANA\_DIN 信号を設定すると、MITOUJTAG ソフトウェアはお客様の作成した VHDL ファイルを解析して、BLOGANA\_DIN 信号に接続されている信号名を自動的に調べます。

そして、右の図 30 のようにバスの各信号名が設定され、ユーザが手動で設定する手間を省くことができます。

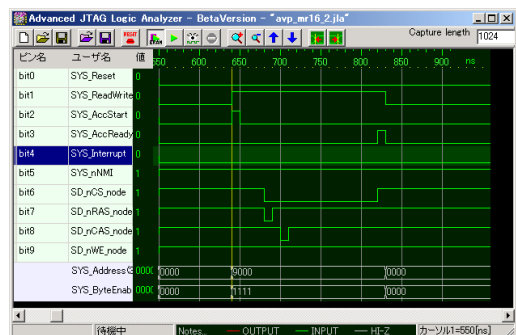


図 30 自動的に信号名が設定された

### 3.4 トリガ機能

#### BLOGANA の動作状態

JTAG ロジックアナライザを起動した直後は、BLOGANA モジュールはアイドル状態になっています。ここでキャプチャ開始ボタンが押されると、トリガ待ち状態に入ります。トリガ待ち状態で、トリガ入力信号(BLOGANA\_TRIG)が'1'になった場合、キャプチャ動作が開始されます。その後、規定のデータ数に達するとデータをパソコンに転送して、再びアイドル状態に戻ります。

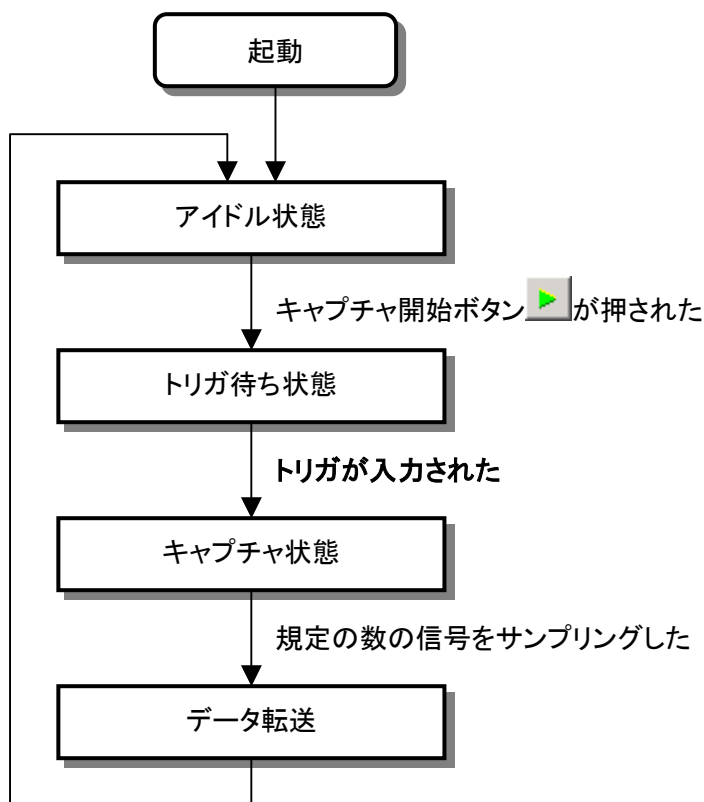


図 31 BLOGANA の動作状態

#### トリガの動作条件

BLOGANA のトリガ入力は、'1'になったらキャプチャを開始するという単純なものです。トリガを使用しない場合には、BLOGANA\_TRIG 信号に常に'1'を設定してください。

逆に、複雑な条件をトリガにしたい場合は、そのトリガ・ロジックを自分で書く必要があります。次のリスト 4 のように when 文を用いると良いでしょう。

#### リスト4 複雑なトリガ条件の指定方法

```
BLOGANA_TRIG <= '1' when (TLP_SEQ_NUM /= 0) else '0';
```

#### トリガポイントとは

TRIG 信号が入力されたタイミングと、実際に波形が取り込まれるタイミングの関係を以下の図 32～34 のように調整することができます。

トリガポイントを 6%に設定した場合には、トリガが入る 64 サンプル前からキャプチャされた信号を表示します。このモードは、トリガ後の波形を主に見たいときに使います。

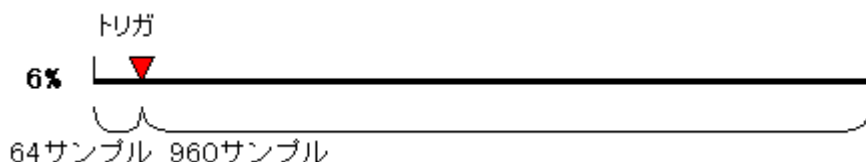


図 32 トリガポイント 6%時の動作

トリガポイントを 50%に設定した場合には、トリガが入る前と後を半々に、キャプチャされた信号を表示します。

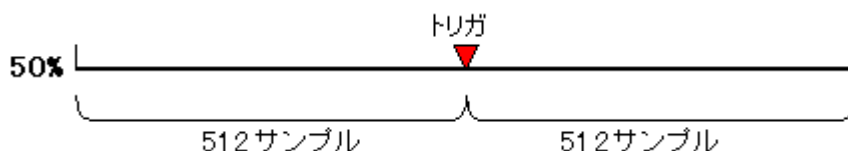


図 33 トリガポイント 50%時の動作

トリガポイントを 96%に設定した場合には、トリガが入る前を 96%、トリガ後を 6%の割合でキャプチャし、表示します。このモードは、トリガ前の波形を主に見たいときに使います。

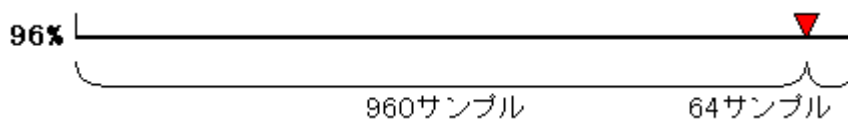



図 34 トリガポイント 96%時の動作

## トリガポイントの設定

BLOGANA モードボタン  を押すと、下の図のダイアログが開きます。

このダイアログでは、BLOGANA モジュールを埋め込んだソースファイルを指定するとともに、トリガポイントを設定することができます。

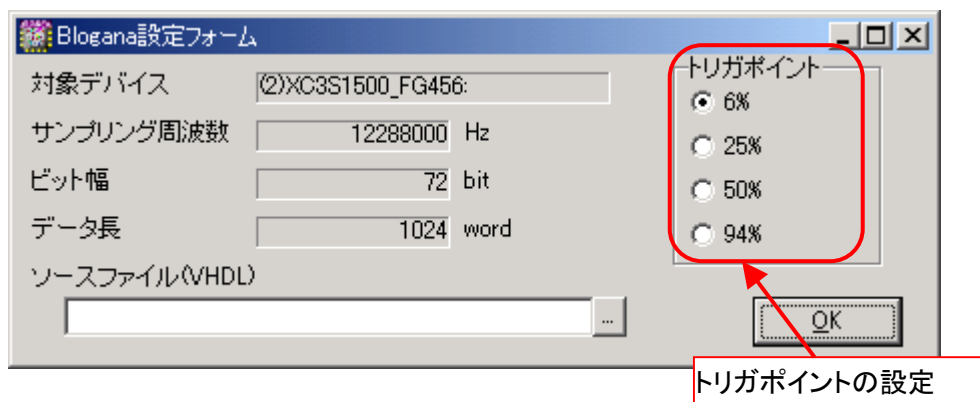


図 35 トリガポイントの設定

---

『JTAG ロジックアナライザ 操作マニュアル』

平成 20 年 7 月 9 日 第 1 版発行

特殊電子回路株式会社

(C)Copyright 2003-2008 特殊電子回路(株) All rights reserved.

無断転載を禁じます