

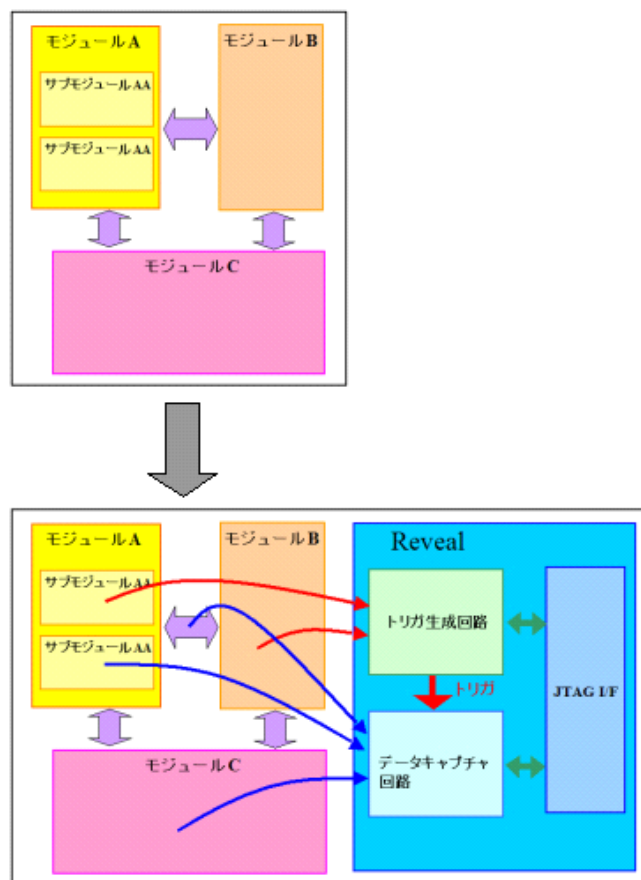
## 第 11 章 Reveal オンチップ・ロジックアナライザー

### 11.1 概要

#### 11.1.1 Reveal の機能

Reveal は ボード上でデバイス内部の動作確認をするためのツールです。ユーザー回路と一緒に Reveal コア（オンチップ・ロジックアナライザー機能= Reveal で生成された回路）を組み込むことにより、JTAG 経由でデバイス内部の信号を観測することができます。組み込む回路とトリガー設定を Reveal Inserter（以下”インサーター”）で、信号の観測を Reveal (Logic) Analyzer（以下”アナライザー”または LA）で行います。

図 11-1. Reveal コア挿入の例



Reveal コアは以下の 3 つの回路ブロックで構成されます。

#### トリガー生成回路：

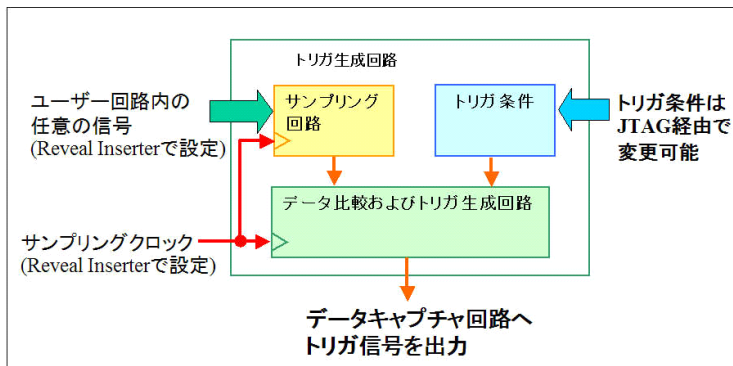
ユーザーが指定した観測信号（以下”トレース信号”）をサンプリングします。JTAG 経由でトリガー信号生成の指示が入力されると、サンプルしたデータとトリガー条件の比較を行い、条件を満たした場合にキャプチャー回路に動作開始を指示するトリガー信号を生成します（図 11-2）。

註：本 Lattice Diamond 日本語マニュアルは、日本語による理解のため一助として提供しています。作成にあたっては各トピックについて可能な限り正確を期しておりますが、必ずしも網羅的あるいは最新でない可能性や、オリジナル英語版オンラインヘルプや各種ドキュメントと不一致がある可能性があります。疑義が生じた場合は技術サポート担当者にお問い合わせ頂くか、または最新の英語オリジナル・ソースを参照するようお願い致します。

トリガー生成条件は、JTAG 経由で変更可能です。

トリガー信号生成のためにサンプルする信号の選択、トレース信号をサンプリングするためのクロック、トリガー生成の初期条件はインサーター（後述）で設定します。

図 11-2. トリガー生成回路の構成

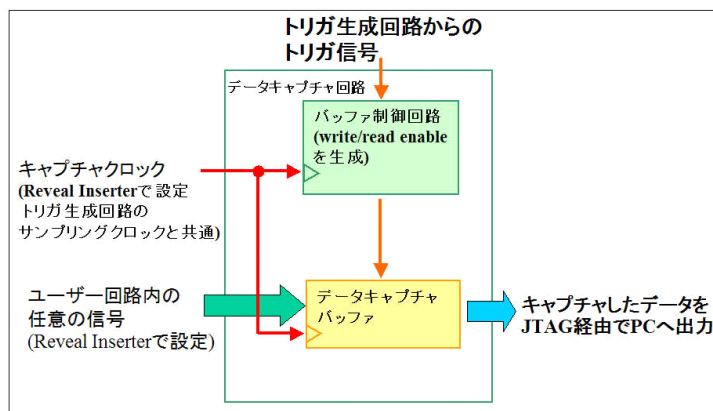


**データキャプチャー回路：**

トリガー生成回路からのトリガー信号を受けて、トレース信号をバッファに保存します。これら信号は、JTAG 経由で PC にロードされ、第 11.3 節で後述するように Waveform 画面に表示されます。

トレース信号の選択、選択した信号をキャプチャーするためのクロック、キャプチャーするためのバッファの深さ等はインサーターで設定します。

図 11-3. データキャプチャー回路の構成



**JTAG I/F 回路 (JTAG Server)：**

JTAG ポートからのトリガー生成回路とデータキャプチャー回路にアクセスするためのインターフェイス回路です。これは予め FPGA 内に組み込まれているもので、デバイスのプログラミング等でも使用されます。従って FPGA 内のリソースは消費しません。

Reveal コアは、ユーザー回路にトリガー生成回路とデータキャプチャー回路を追加（ツールが自動的に処理）するため、必要なリソース（特に EBR）が増加します。

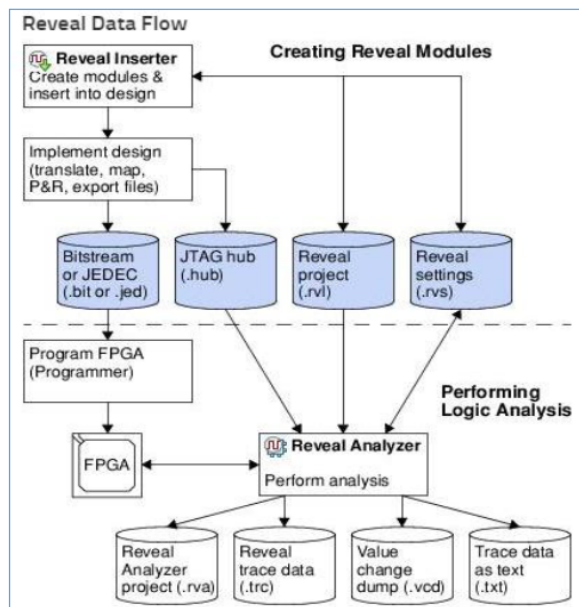
**11.1.2 実装手順**

図 11-4 は Reveal を実装する場合の手順です。Reveal はインサーター とアナライザーから構成されます。デザインエントリー後に インサーターを使用してトリガー条件やトレース信号用バッファのサイズ等の設定を行い、ユーザー回路に挿入します。

その後、通常のデザインフローに従って論理合成から PAR および書き込みデータ生成までを実行し、PC

と PCB をダウンロードケーブルで JTAG 接続してプログラミングします。LA を起動し、PC 上で信号のキャプチャー開始を指示すると、動作を開始します。トリガ条件が満たされてバッファに保存されると、トレース信号波形が表示されます。

図 11-4. Reveal 実装の手順



## 11.2 インサーター (Reveal Inserter)


### 11.2.1 サンプリング対象の信号

インサーターは、デザインファイルが全て VHDL または Verilog HDL の場合は、HDL ソース内の信号リストからサンプリング対象を選択することができます。混在言語もサポートします。また、バス信号も保持しますので、そのまま認識できます。

ただし、以下のように幾つかの制限がありますのでご注意ください (オンラインヘルプ: "Testing and Debugging On-Chip" → "Creating Reveal Modules" → "About Reveal Inserter" → "Limitations")

- 変数 (if-then-else の中や case 文中など) は対象やトリガーに指定できません
- 二次元以上のアレイ、およびアレイ内の function は認識されません
- Verilog: 未宣言の wire は階層ツリーに表示されません。1 ビットでも wire 宣言する必要があります
- VHDL: 単一モジュールの Entity と Architecture は同一ファイルに記述されている必要があります
- VHDL: 論理合成のアトリビュート指定は Architecture 内ではなく Entity 記述内で行う必要があります
- VHDL: 論理合成のアトリビュート指定 syn\_keep / syn\_preserve は string ではなく boolean として定義する必要があります
- VHDL: generate 記述内の信号は対象やトリガーに指定できません
- VHDL: ユーザー定義の enum/integer/boolean は対象やトリガーに指定できません

### 11.2.2 インサーターの起動

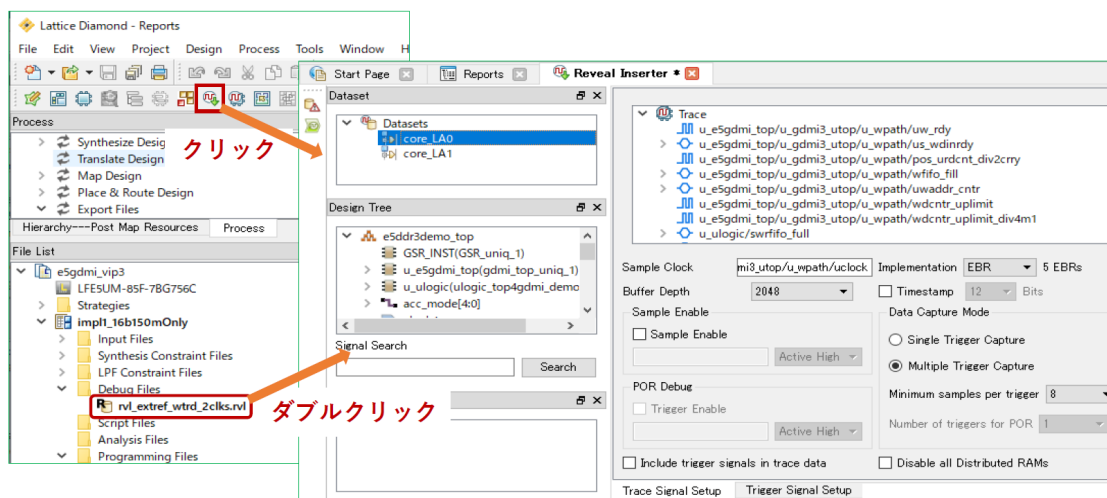
プロジェクトで初回にインサーターを起動する前に、最低限 Diamond プロジェクトを作成し、デザインソース (HDL もしくは EDIF) をインポートしておく必要があります。そして  アイコンをクリックする

か、メニューから [Tools] → [Reveal Inserter] と選択してインサーターを起動します (図 11-5)。

一旦 Reveal プロジェクトを作成しファイルリスト・ビュー (File List) の ”Debug File” 部にインサーターのプロジェクト・ファイル (.rvl) をインポートしてあれば、これをダブルクリックすることでもインサーターを起動できます。

ただし、プロジェクトにインポートされている RTL ソースの言語が統一されている場合は、論理合成実行前でもインサーターで作業ができますが、混在言語の場合は論理合成実行後でないとい作業できません。

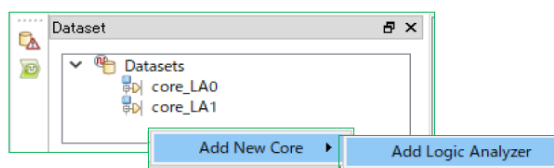
図 11-5. インサーターの起動



### 11.2.3 Reveal コアの追加

インサーター起動時には、自動的に Reveal コアが 1 つ作成されます。Reveal コアは一系統のクロックで動作するため、サンプルに必要なクロック系統が複数ある場合は、複数の Reveal コアを使用する必要があります。Reveal コアを追加する場合は、インサーターの Dataset 枠内のどこかで右クリックすると表示されるメニューから [Add New Core] → [Add Logic Analyzer] を選択します (図 11-6)。

図 11-6. Reveal コアの追加



Reveal コアの名称は変更することができます。変更する Reveal コアを選択した状態で右クリックすると表示されるメニューから [Rename Core] を選択して、適切な名称にします。なお、図 11-6 において、PCS (SERDES) 搭載のデバイスでは [Add Logic Analyzer] 以外に [Add SERDES Debug] も選択対象として表示されます (第 11.4 節参照)。

Reveal コアを削除する場合は、対象とする Reveal コアを選択した状態で、キーボードの Del キーを押すか、Reveal コアを選択した状態で右クリックすると表示されるメニューから [Remove Core] を選択します。

### 11.2.4 インサーター作業の注意点

作成済み Reveal プロジェクトを編集・更新する際の注意点をまとめます (新規の場合は該当しません)。

#### RTL ソースファイルを修正する場合：

インサーターを起動して GUI を表示している状態でも、或いはインサーターを起動する前でも構いま

せん。修正・編集を終了したら、必ず Diamond メニューから [Design] → [Refresh Design] を選択して修正を反映させます。RTL 内の信号（ノード）名を変更した場合なども該当します。

”リフレッシュ”しないと、後述の DRC でエラーになります。

### RTL ソースファイルを差し替えたい場合：

RTL 記述の推敲段階で、モジュール（エンティティ）名を同じままで RTL 記述の異なる複数のファイルを作成して差し替えて検討することは、しばしばあり得るアプローチです。

ファイルを差し替える前に、インサーターを起動して GUI を表示している状態の場合は、一旦クローズします。ファイル名やモジュール（エンティティ）名が同じか異なるかには依存しませんが、（以降に記述する）トレース対象やトリガとして指定している信号が含まれる RTL ソースファイルの場合は、クローズすることを強く推奨します。差し替えた後に、上と同様 [Design] → [Refresh Design] を実行してから、インサーターを起動します。

インサーターを開いた状態でファイルを差し替えると、トレース信号やトリガ信号の設定を含めて初期状態になったり、予期しない振る舞いになり、ほぼ全ての設定を初めからやり直す状況になり得ます。ファイルの削除（インプリメンテーションから除外）も同じです。

## 11.2.5 トレース信号の設定

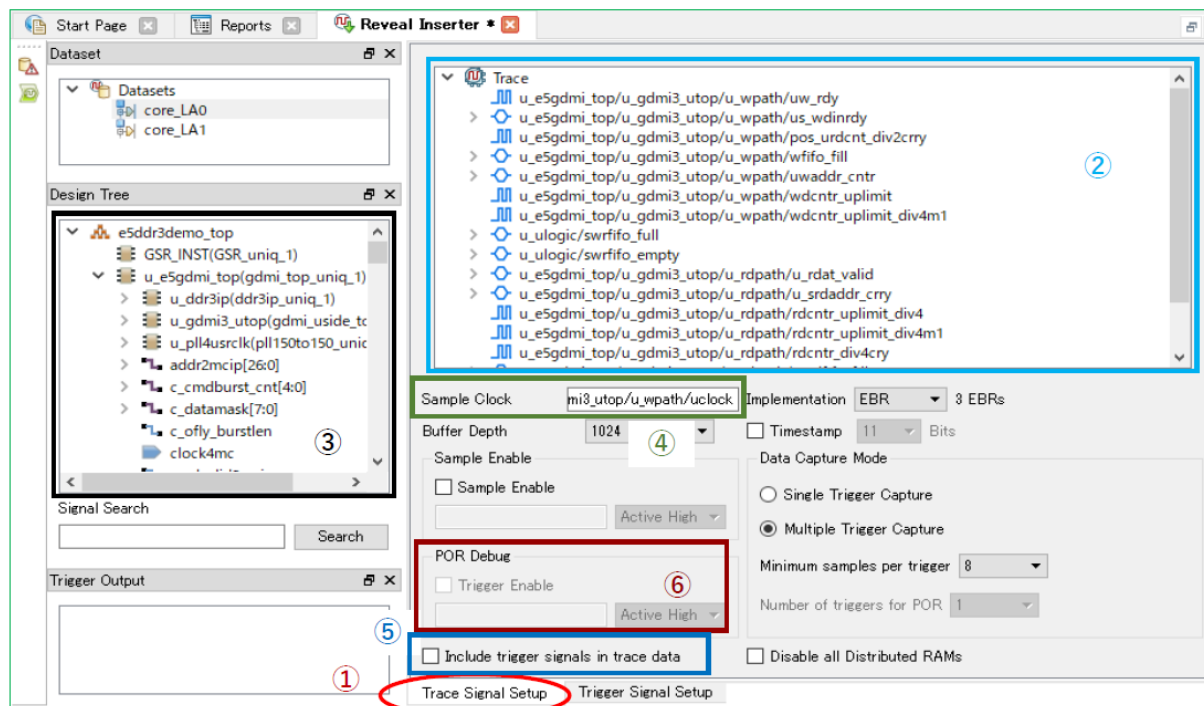
### 11.2.5.1 トレース信号とクロックの選択

トレース信号を指定するために [Trace Signal Setup] タブを選択します（図 11-7 ①）。

#### クロックの選択

トレース信号をサンプルするクロックを“Design Tree” 枠（図 11-7 ③）で選択し、「Sample Clock」ル（図 11-7 ④）にドラッグ&ドロップします。サンプルクロックは、観測する信号と同期していて、かつ TCK (JTAG クロック) より高速の信号を選択するようにします。これより遅いと何らかのエラーが起きる可能性があります。またクロックは間欠的ではなく、連続動作している必要があります。

図 11-7. インサーターのトレース信号設定



## トレース信号の選択

トレース信号を”Design Tree” 枠 (図 11-7 ③) で選択し、”Trace” 信号枠 (図 11-7 ②) にドラッグ & ドロップします。Reveal コアあたり最大 512 本まで選択することができます。

Trace 枠に表示されている順序でキャプチャー波形が表示されます。信号の表示順を変えたい場合は、Trace 枠内で信号名をドラッグ & ドロップすることで上下順を変更します。トレース信号を除外する場合は、その信号を選択してキーボードの”Delete” キーを押します。

なお、トリガーとして選択した信号もトレースする場合は「Include trigger signal in trace data」(図 11-7 ⑤) にチェックを入れます。

## パワーオンリセット (POR) 直後のデバッグ

図 11-7 ⑥部「POR Debug」で「Trigger Enable」をチェックすることで、コンフィグレーション完了後にデバイスの内部グローバル・リセット信号 POR が解除される直後のデバッグが可能になりました。

後述の”Trigger Expression (TE)” を二つ以上設定済の場合は、「POR Debug」をイネーブルする選択ができません。TE を一つにしてからイネーブルします。TE が一つもないと DRC エラーになります。

「Trigger Enable」は一般的な”Sample Enable” と全く同じ動作です。チェックした後は、入力が有効になるトリガー信号とその有効レベルを、その下のセルにドラッグ & ドロップで指定します。

Trigger Enable 信号の極性指定には注意が必要です。トレース信号を取り込む状態でのレベルを High / Low 指定します。例えば、Low Active の外部リセットを指定する場合、ユーザー回路の動作はこのリセットが High の状態なので、極性指定は [Active High] とします。

波形キャプチャの操作としては、アナライザーを起動する前にデバイスの電源をオフオンして内部 POR 信号がトグルするようにします。その後アナライザーを起動すると、その時点で既に内部メモリにダンプされているトレース信号が表示されます。

## 観測信号のバス作成

デザイン内のバス信号の観測は、他の信号と同様に扱うことができます。ただし、展開すると LSB が最上部に MSB が最下部になっているのが分かります。逆にしたい場合は手動でドラッグして意図する順序にするしかありません。

また個別の信号をバスのようにグループ化することができます。複数の対象信号を選択後に右クリックすると現れるメニューから”Group Trace Data” を選択します。”BusNN” (NN は番号) のような名称の信号が追加されますので、ダブルクリックして名称を適宜変更します。

### 11.2.5.2 サンプル数の設定

「Sample Clock」の下にある「Buffer Depth」でトレース信号のキャプチャーに使用するバッファの深さ (ワード数 or 有効サンプル数) を指定します。右側の下向き矢印をクリックする表示される候補から、適切なサンプル数を選択します。

サンプル数が多くなると、必要になるメモリーリソースも増加し、メモリー不足になることがあります。サンプル数は必要最低限にすることを推奨します。

### 11.2.5.3 データを保存するメモリーリソースの選択

「Sample Clock」の下にある「Implementation」で、サンプルしたトレース信号を保存するためのメモリー・タイプを選択します。右側の下向き矢印をクリックすると表示される候補 (EBR か ”DistRAM”) から指定します。EBR はブロック RAM、”DistRAM” はスライスを使用して実現する分散 RAM です。メモリータイプを選択すると、その右側に必要となるリソース数が表示されます。

### 11.2.5.4 タイムスタンプの設定

「Timestamp」にチェックを入れると、波形をキャプチャーした際にタイムスタンプ (起動時からのカウント値) も表示されます。タイムスタンプのビット幅はその右側で選択します。

### 11.2.5.5 サンプル・イネーブルの追加

特定の信号をサンプル・イネーブルとして使用する際は、「Sample Enable」にチェックを入れた後、その下のセルにイネーブル信号を「Design Tree」枠からドラッグ&ドロップします（図 11-8）。

信号入力セルの右側には極性選択があり、イネーブル極性を選択することができます。[Active High] を選択すると、イネーブル信号が High レベルの期間のみ、[Active Low] を選択すると、Low レベルの期間のみ、トレース信号のサンプリングが行われます。イネーブル信号が指定した極性でない場合は、信号のサンプリングが行われないため、トリガーもかかりません。

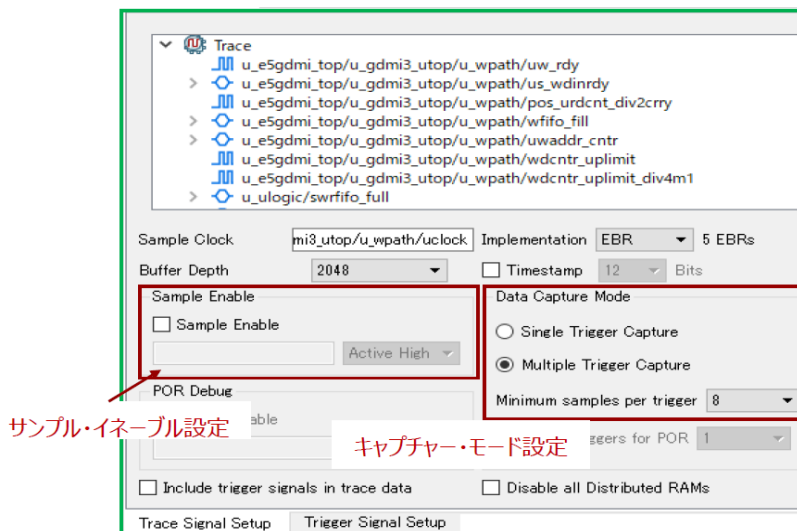
### 11.2.5.6 データキャプチャー・モードの選択

「Data Capture Mode」部でどちらかをチェックしてキャプチャー・モードの選択を行います(図 11-8 右下)。

- Single Trigger Capture : 1度だけトリガー条件成立後の信号をキャプチャー
- Multiple Trigger Capture : 指定した回数分だけトリガー条件成立後の信号をキャプチャー

Single モードでは、信号のキャプチャーが1度のみであり、トリガー条件成立後に Buffer Depth 分のサンプルをキャプチャーします。

図 11-8. サンプル・イネーブルとキャプチャー・モードの設定



Multiple モードでは、指定した回数分だけトリガー条件成立後にデータのキャプチャーを行います。バッファを分割して使用するため、指定トリガー回数が増えるとトリガー成立ごとにキャプチャーできるサンプル数は  $(\text{buffer depth} \div \text{指定トリガー回数})$  となります。回数が2のべき乗でない場合でも分割は常に2のべき乗値に丸められます。

図 11-9. 観測波形例 : Single モード (左)、Multiple モード (右)

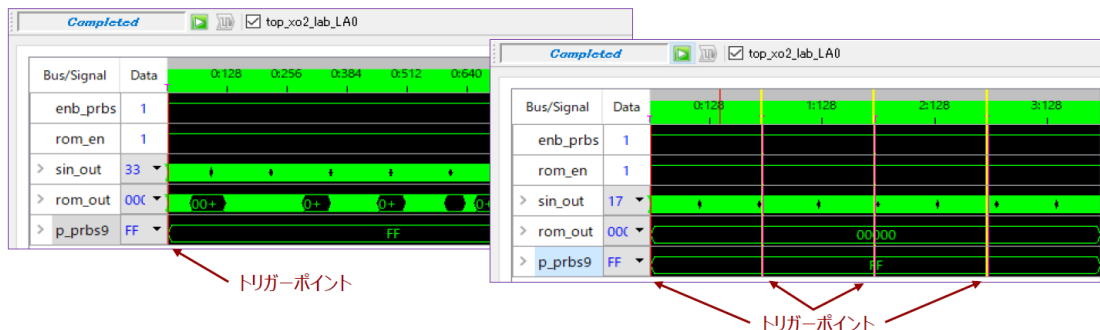


図 11-9 は Single モードと Multiple モードの波形キャプチャー例です。どちらも Buffer Depth の設定は 1024 です。Single モードではトリガーポイントは 1 つなので、その後に 1024 サンプル分の波形をキャプチャーしています。Multiple モード（トリガー回数 4 回に設定）では、トリガーポイント 4 つとそれに続く 256 サンプル（1024 buffer depth ÷ 4 トリガー）分の波形が表示されています。”Buffer Depth” ÷ ”Minimum samples per trigger” が最大トリガー回数となります。

Multiple モードを選択した場合は、「Minimum samples per trigger」（図 11-8 右側）で、トリガー条件成立後にキャプチャーする最小サンプルデータ数が設定できます。

例：Buffer Depth=512 Minimum samples per trigger=16

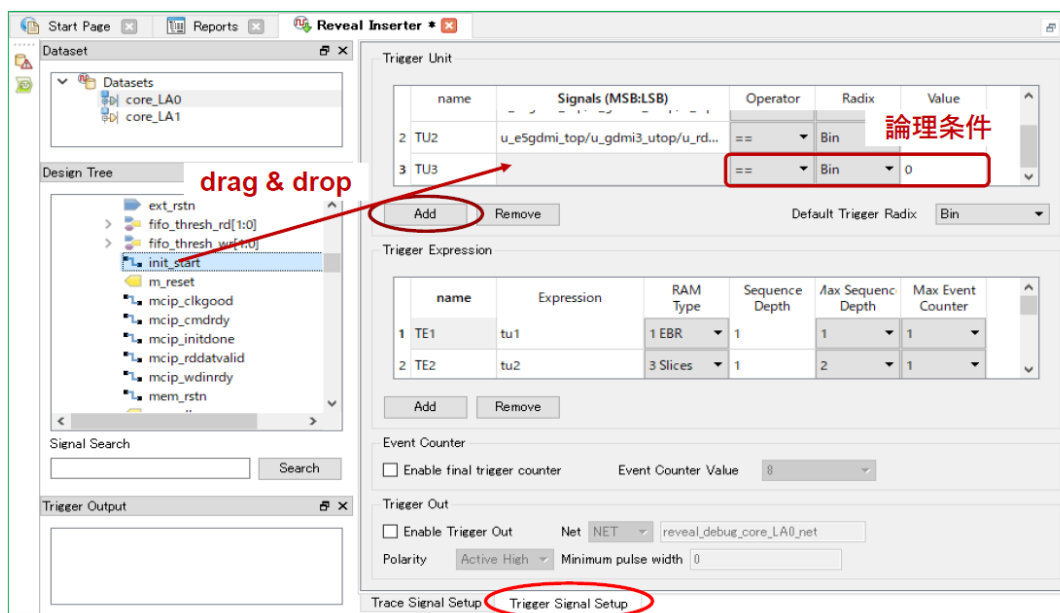
512/16 = 32 最大 32 回の繰り返しトリガーが可能

- ・ 32 回のトリガーにした場合、トリガーあたりのサンプル数は 16
- ・ 16 回のトリガーにした場合、トリガーあたりのサンプル数は 16 もしくは 32 から選択
- ・ 4 回のトリガーにした場合、トリガーあたりのサンプル数は 16,32,64 もしくは 128 から選択

### 11.2.6 トリガー条件の設定

トリガー条件は、信号とその論理的条件の組み合わせである TU（Trigger Unit）と、単一の TU か複数の TU を組み合わせで定義する TE（Trigger Expression）の二つで設定します。トリガー条件の設定には、まず [Trigger Signal Setup] タブを選択します（図 11-10）。

図 11-10. トリガー条件の設定



#### 11.2.6.1 TU（Trigger Unit）の設定

まず”Trigger Unit”部の『Add』ボタンをクリックして、空（Signalsセルがブランク）のTUを追加します（図 11-10 は TU3 を追加した後の図）。TUは単一の Reveal コアに対して最大 16 個作成できます。

##### Signalsセルの設定

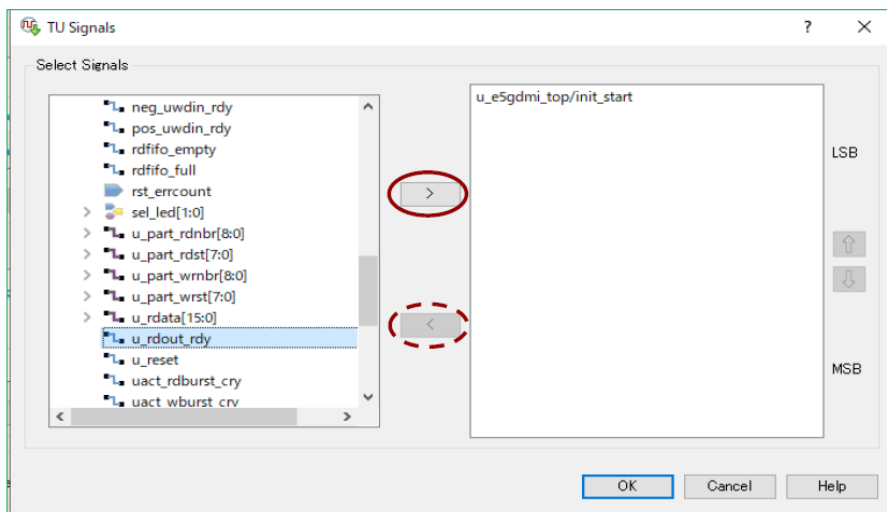
[Signals] カラムには、トリガー条件に使用する信号を指定します。”Design Tree”からドラッグ&ドロップします（図 11-10）。

或いは当該 TU の [Signals] カラムをダブルクリックすると、信号を選択するウィンドウが起動します（図 11-11）。ウィンドウ左側の枠内で信号を選択し、ウィンドウ中央の『>』ボタンをクリックすると



選択した信号が右側の枠に追加されます。複数の信号を選択した場合、右側の枠では上側が LSB、下側が MSB として扱われます。右側の枠内で信号を選択し枠の右側の上下矢印ボタンをクリックして、信号の並び順を変更することができます。必要な信号を全て選択後、『OK』ボタンをクリックするとウィンドウが閉じ、選択した信号が [Signal] カラムに表示されます。

図 11-11. TU 信号選択ウィンドウ



### Operator カラムの設定

TU に指定した信号と [Value] カラム値の関係論理式です。表 11-1 は選択肢とその定義です。実機デバッグの作業中に、再フィッティングすることなく動的に変更することができます。

表 11-1. Operate カラムの選択肢

選択肢	評価内容
==	選択した信号と Value セルの値が等しい場合に真
!=	選択した信号と Value セルの値が等しくない場合に真
>	選択した信号が Value セルの値より大きな場合に真
>=	選択した信号が Value セルの値より大きいか等しい場合に真
<	選択した信号が Value セルの値より小さい場合に真
<=	選択した信号が Value セルの値より小さいか等しい場合に真
rising edge	選択した信号の内、対応する Value セルの値が '1' に設定されている信号の立ち上がりが検出された場合に真
falling edge	選択した信号の内、対応する Value セルの値が '1' に設定されている信号の立ち下がりが検出された場合に真
serial compare	選択した信号が Value セルの値と同じ順番で観測された場合に真（選択した信号がシングル bit の場合のみ選択可）

### Radix カラムの設定

[Radix] カラムの表記フォーマットです。表 11-2 は選択肢とその定義です。実機デバッグの作業中に、再フィッティングすることなく動的に変更することができます。

### Value カラムの設定

TU 信号の論理値です。"Radix" 指定の表記で記述します。不適切な値が入力されると、赤字で表示されます。実機デバッグの作業中に、再フィッティングすることなく動的に変更できます。整数型や列挙型で論理合成前にビット幅や決まっていな信号や、状態を表すビットパターンが決まってい

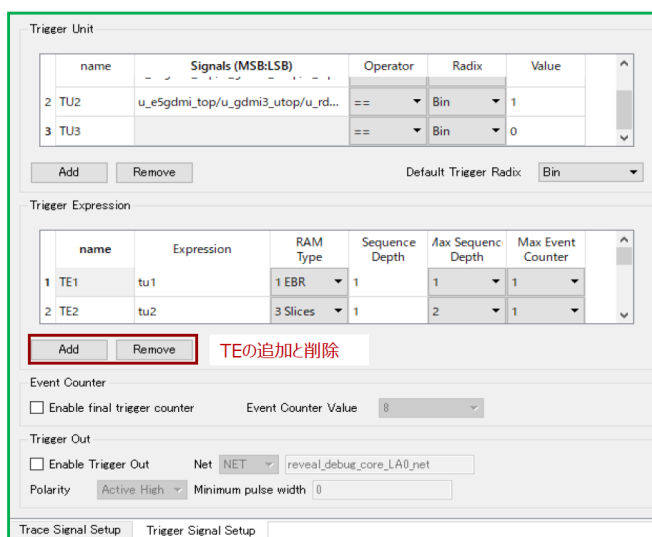
ない信号は、Reveal が自動的にソース内のタイプや値を読み込んでその値を表示します。

表 11-2. Radix カラムの選択肢

選択肢	表記フォーマット
Bin	2 進数で表記
Oct	8 進数で表記
Dec	10 進数で表記
Hex	16 進数で表記
REVEAL_ENUM...	VHDL ソースに列挙型等で記述されている変数のタイプ
REVCEAL_INT...	整数型で記述されている変数のタイプ

### 11.2.6.2 TE (Trigger Expression) の設定

図 11-12. TE 設定



最初に Trigger Expression 部の『Add』ボタン (図 11-12) をクリックして、空 (Expression カラムが空白) の TE を追加します (既にある場合はスキップしても良い)。

#### Expression カラム

[Expression] カラムには、トリガー条件として単一の TU のみを記述するか、複数 TU の論理式を記述します。表 11-3 は使用できる演算子とその内容の一覧です。

前述の TU 各カラムと異なり、実機デバッグの作業中に、動的に変更することはできません。

TE の記述例を示します。

- TU1 & TU2 :TU1 と TU2 が同時に真の場合にトリガー生成
- TU1 | TU2 :TU1 もしくは TU2 が真の場合にトリガー生成
- !TU3 :TU3 が真でない場合にトリガー生成
- TU1 & !TU4 :TU1 が真、かつ TU4 が偽の場合にトリガー生成
- TU3 ^ TU1 :TU3 もしくは TU1 の一方が真で他方が偽の場合にトリガー生成
- TU1 next TU2 :TU1 が真になった次のサンプリングで TU2 が真の場合にトリガー生成
- TU1 then TU2 :TU1 が真になった次のサンプリング以降に TU2 が真の場合にトリガー生成
- TU5 #2 :TU5 が 2 回真になった場合にトリガー生成
- TU5 ## :TU5 が 2 回連続して真になった場合にトリガー生成

表 11-3. Expression カラム内で使用できる演算子

演算子	演算子の意味
&	前後の TU の and 論理
	前後の TU の or 論理
^	前後の TU の exor 論理
!	TU の負論理
()	括弧内の論理式を優先
next	前の TU 成立後、次のサンプルで後の TU が成立。”Max Sequence Depth” カラムを設定
then	前の TU 成立後、後の TU 成立。”Max Sequence Depth” カラムを設定
##	前に指定された TU が、後に指定された回数連続して成立。回数の指定は ”Max Event Counter” カラムで与える
#	前に指定された TU が、後に指定された回数成立。回数の指定は ”Max Event Counter” カラムで与える

### RAM Type カラム

トリガー生成回路内で使用するメモリタイプを選択で、EBR (ブロックメモリー) もしくは Slices (分散メモリー) のどちらかを選択します。カラム右側の矢印をクリックすると、選択肢が表示されます。タイプの左側に表示されているのは現在の設定で必要とされるリソース数です。

### Max Sequence Depth カラム

TE で設定するシーケンス段数 (then または next 演算子で接続できる項数) の上限を設定します。”Sequence Depth” カラムに、”Expression” カラムの記述から自動的に算出されたシーケンス段数が表示されますが、この値より小さいと DRC エラーになります。

実機デバッグの作業中にこの値を変更することはできません。

### Max Event Counter カラム

TE で設定する繰り返し回数 (# または ## で指定できる値) の上限を設定します。TE で # または ## 指定した数値より小さいと DRC エラーになります。

実機デバッグの作業中にこの値を変更することはできません。

### Trigger Out 設定

”Trigger Out” 部の「Enable Trigger Out」にチェックを入れると、Reveal コア内で生成されたトリガー信号を外部で観測できるようになります。

## 11.2.7 デザインルール・チェック


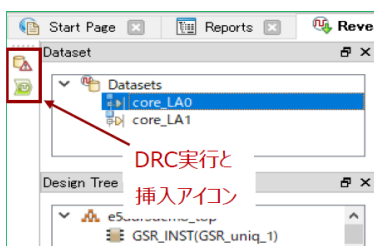

各設定が完了したら、必ずウィンドウ左上 (図 11-13) のアイコン  をクリックすることでデザインルール・チェック (DRC) を実行し、設定内容に問題がないかどうかを確認します。チェック結果としてエラーの有無と Reveal 挿入に必要なリソース数が Diamond の Output ウィンドウに表示されます。エラーがなければ、必ず保存しておきます。

図 11-13. DRC 実行と Reveal コア挿入アイコン

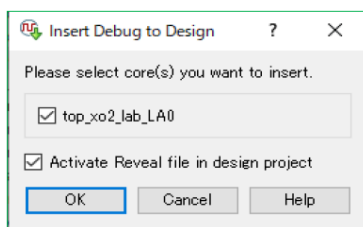


## 11.2.8 Reveal コアをデザインに挿入

各設定が完了して DRC がパスしたら、“Insert Debug” アイコン  (図 11-13) をクリックし Reveal コアをユーザーデザインに挿入します。アイコンをクリックすると、図 11-14 のような挿入する Reveal コアの選択ウィンドウが起動します (単一コアのみがある場合は、一つしか表示されません)。デザインに挿入する Reveal コアの左側のボックスをチェックし、『OK』をクリックすると、Reveal インサーター・プロジェクト・ファイル (\*.rvl) がプロジェクト・フォルダーの下に作成されます。また、ファイルリスト・ビューの “Debug Files” セクションに自動的にインポートされて、ボード表示されます (図 11-5 例の左下部)。

以上でインサーターでの作業は完了です。この後は、通常のデザインフローに従って論理合成から PAR、書き込みデータ作成までを実行します。

図 11-14. 挿入する Reveal コアの選択ウィンドウ



**【重要】**インプリメンテーションごとに複数の異なる \*.rvl ファイルをインポートすることが可能ですが、アクティブな (論理合成以降の処理に組み込まれる) ファイルは一つのみです。一方、後述のインサーター・ファイル \*.rva はプロジェクトとして一つのみ有効で、複数共存することはできません。アクティブなコアを切り替えてデバッグできますが、切り替えると論理合成から再実行されます。複数の \*.rvl を作成・インポートする場合は、Reveal インサーター・プロジェクト名を別名にする必要があります。また、\*.rvl/\*.rva ファイルはプロジェクト・フォルダー直下になければなりません。

## 11.2.9 Reveal コアの非アクティブ化

複数の Reveal コア \*.rvl をインプリメンテーションにインポートしている場合、アクティブなコアは一つですので、それ以外は非アクティブにする必要があります。また、デバッグ完了後に Reveal コアを一つも含まないデザインにする場合も、\*.rvl をインポートしたままで非アクティブ化します。太字で表示されるアクティブな \*.rvl をクリックして選択し、右クリックすると表示されるメニューから [Set as Inactive] を選択すると、Reveal コアは非アクティブ化 (細字で表示) されます。非アクティブな Reveal コアをアクティブ化する場合は、同様に Reveal コアを右クリックして [Set as Active] を選択します。

## 11.2.10 Reveal コア挿入と論理合成について

インサーターでトリガーまたはトレース信号として選択された HDL 内のノードは、最適化されずに論理合成後もネットリスト内に残ります。このため、Reveal コア挿入の有・無によって、観測信号周辺の回路構成やタイミングが変わり得ることにご留意ください。

## 11.3 アナライザー (Reveal Logic Analyzer)

### 11.3.1 スタンドアローン版アナライザー

Reveal Logic Analyzer (LA) は、Lattice Diamond のインストール時に同時にインストールされますが、スタンドアローン版も用意されています。こちらはライセンスが不要のため、Lattice Diamond をインストールしていない PC 上で LA のみを使用する、というような使い方ができます。以下の URL から選択してダウンロードできます。

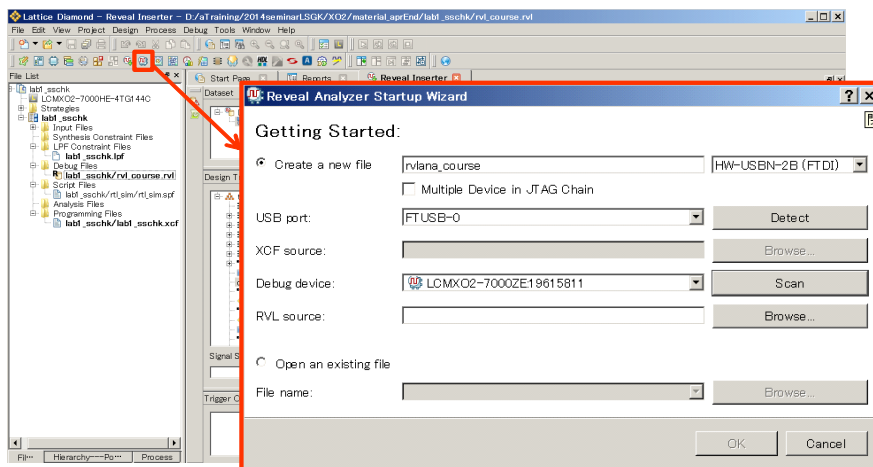
<https://www.latticesemi.com/en/Products/DesignSoftwareAndIP/FPGAandLDS/LatticeDiamond>

### 11.3.2 アナライザ作業の準備

LA はインサーターとは異なるユーザー・インターフェイスです。LA を起動する前に電源が投入された PCB と LA で作業する PC をダウンロードケーブルで接続し、Reveal コアが挿入されたデザインで FPGA がコンフィグされていることが必要です。

また、対象 FPGA の JTAG チェーンに複数のデバイスがある場合、プログラマーの設定ファイル \*.xcf が必要です。Diamond プロジェクトのファイルリスト・ビューの ”Programming Files” セクションにインポートしておきます。

図 11-15. LA の起動



LA プロジェクト作成時に、”Inserter ファイル (インサーターで作成したファイル) 内の ID” と ”デバイスに実装されている Reveal コアの ID” の比較を行います。これらが一致しないと起動が成功しませんのでご注意ください。

### 11.3.3 アナライザ作業開始時の注意点

**インサーター・プロジェクト・ファイル (\*.rvl) を差し替えた場合：**


デバッグ作業では、複数のインサーター・ファイルを作成し、デザインに挿入するアクティブな Reveal コアをファイルごと差し替える手法を取ることは、しばしば取られるアプローチです。

アナライザ (LA) の設定ファイル \*.rva は Diamond プロジェクトとして一つのみ有効であり、複数共存することはできません。すでに LA 作業を行った後の差し替えでは、生成済み \*.rva と差し替えたコアが一致しないと作業に支障が出ますので、三つの LA 用ファイル (\*.rva/\*.rvs/\*.trc) を手動であらかじめ削除しておきます。

**回復できないエラーに遭遇した場合：**

まれに LA での作業で対処方法がわからない問題に遭遇し、自力では解決できないケースがあり得ます。そのような場合は LA を終了し、上と同様に三つの LA 用ファイル (\*.rva/\*.rvs/\*.trc) を手動で削除します。その後改めて次以降に記述する新規 LA 起動ウィザードからやり直すことを推奨します。

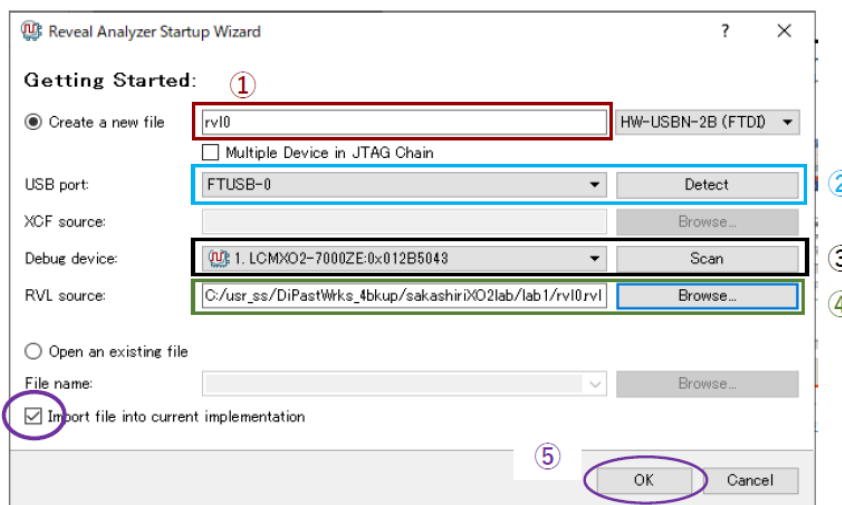
### 11.3.4 アナライザの起動

LA の起動は Diamond メニューバーの下にあるアイコン  をクリックします (図 11-15)。LA は起動時に Diamond プロジェクト・フォルダー下にある LA プロジェクト・ファイル (\*.rva) の検索を行います。初回の LA 起動時は、自動的に新しい LA プロジェクトの作成プロセスに進みます (図 11-15)。

新規に LA プロジェクトを作成する場合は、表示される ”Reveal Analyzer Startup Wizard” で「Create a new file」にチェックが入っていますので、その右のセルに任意の LA プロジェクト名を入力します (図 11-16 ①)。

次に「USB Port」で PC とボードを接続しているケーブルのポートを選択します。右側の『Detect』ボタン (図 11-16 ②) をクリックすると、接続されているケーブルが自動的に検出されてセルに表示されます。

図 11-16. LA プロジェクト作成 Wizard



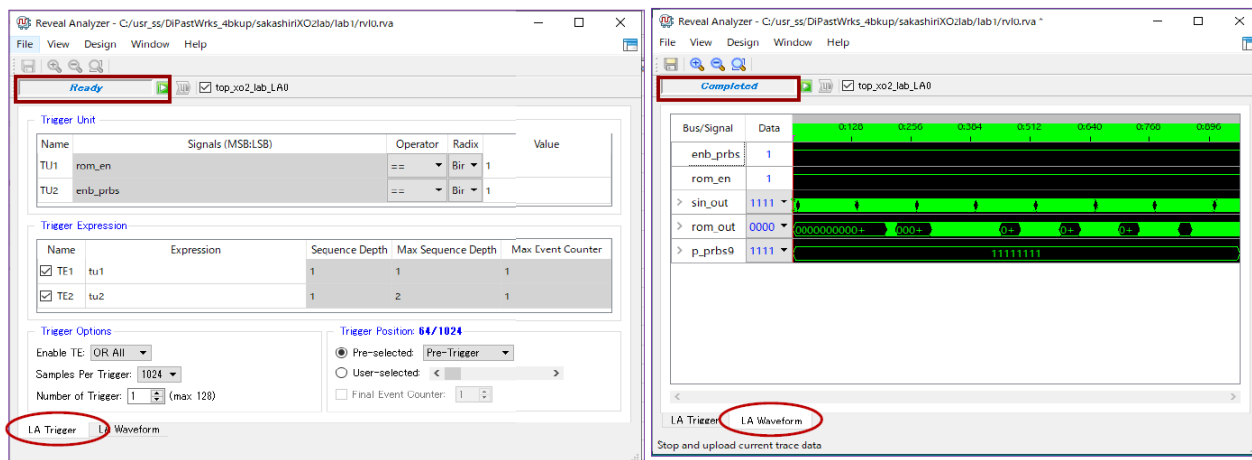
次は「Debug device」セルに入る、デバッグ対象デバイスの検出です。右側の『Scan』ボタン (図 11-16 ③) をクリックすることで、JTAG チェーンに接続されているデバイスが検出され、「Debug Device」セルで選択できるようになります。デバイスのリストは JTAG チェーンに複数のデバイスがある場合は、その先頭にあるものから順に表示されますので、この中から対象デバイスを選択します。JTAG チェーン内のデバイスに Reveal コアが含まれているものが全くない場合、エラーメッセージが出力されます。

次に④の「RVL source」セルで右側の『Browse』ボタンをクリックして、作成済みのアクティブな Reveal インサーター・ファイル (\*.rvl) を選択します。

最後に対象 FPGA の JTAG チェーンに複数のデバイスがある場合のみ、「XCF source」セルに当該 \*.xcf ファイルを選択します。④ RVL source 選択まで行くと、プロジェクト・フォルダー下の \*.xcf リストが表示されるようになりますので、適切なものを選択します。単一デバイスのみ接続されている場合は不要です。xcf ファイルについては第 14.1.8 項をご参照ください。

これらの設定が完了するとウィンドウ下部の『OK』ボタンがアクティブになりますので、クリックして準備完了です。

図 11-17. LA のタブ表示



## 11.3.5 アナライザーのウィンドウ表示操作

### 11.3.5.1 ウィンドウ構成と基本操作

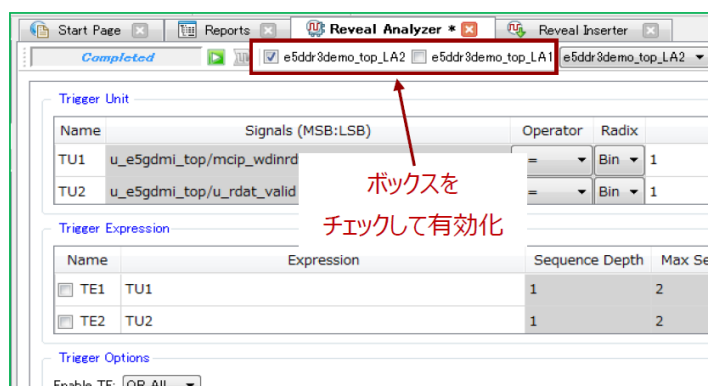
LA にはトリガー条件を設定する [LA Trigger] タブと、波形を表示する [LA Waveform] タブの二つがあります。複数の Reveal コアが実装されている場合でも、タブ表示は変わりません。

起動直後に動作準備が完了していれば、[LA Trigger] タブ選択状態で左上ステータスが "Ready" になります。第 11.3.7 項でも記述するように、キャプチャ開始アイコンが有効になっている状態で、これをクリックし波形キャプチャーを開始し、完了すると図 11-17 右のように [LA Waveform] タブ表示に切り替わり、ステータスも "Completed" になります。

### 11.3.5.2 Reveal コアが複数の場合のコア選択

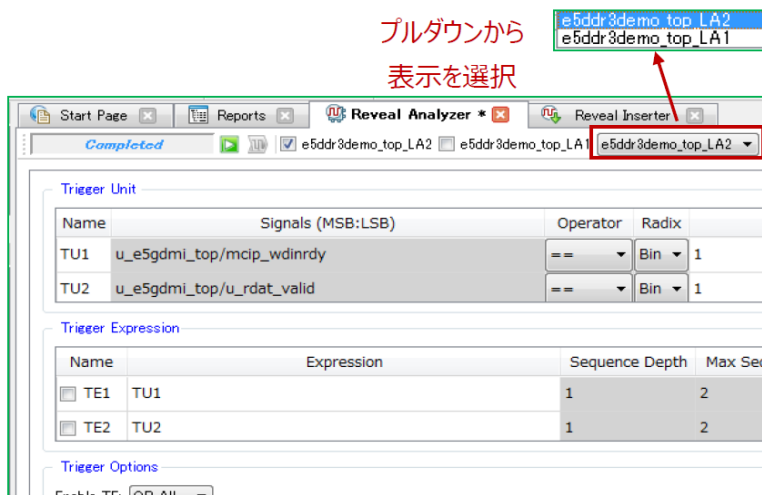
複数の Reveal コアが実装されている場合、ウィンドウ上部に Reveal コア名が表示されます。LA 名の左側にはチェックボックスがあり、チェックの入っている Reveal コアが波形キャプチャーの対象となります (図 11-18)。

図 11-18. 動作させる Reveal コアを選択



アイコン をクリックして LA の実行を開始すると (第 11.3.7 項)、イネーブルされている Reveal コア全てで波形キャプチャーが完了するまで再実行できません。トリガ条件が成立するのを待っている状態で動作を停止する場合には、実行開始アイコン表示が停止アイコン に変わりますので、これをクリックします。

図 11-19. 表示する Reveal コアを選択

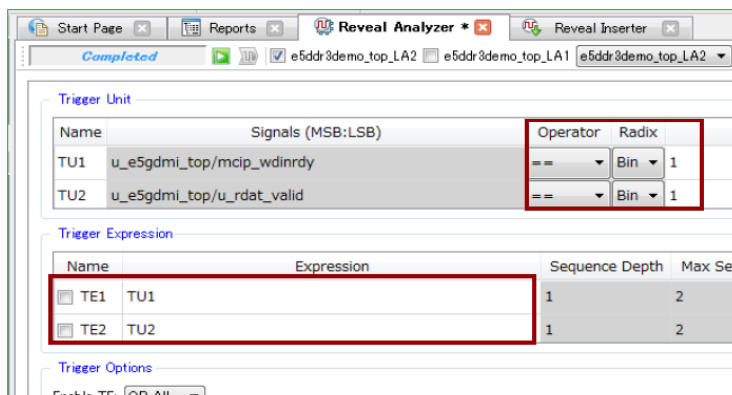


### 11.3.5.3 Reveal コアが複数の場合の表示切り替え

LA で表示できるトリガー設定とキャプチャー波形はそれぞれ Reveal コア一つ分のみです。複数の Reveal コアを実装している場合、ウィンドウ上部にある Reveal コアを選択して表示を切り替えます (図 11-19)。

### 11.3.6 トリガー設定の編集

図 11-20. LA トリガー条件の変更可能箇所



LA 起動時には、インサーターで設定したトリガー条件が設定されていますが、その一部は LA で変更することができます。TU および TE の追加はできません。また TU に設定されているトリガー条件の対象となる信号を LA 上で変更することもできません。

#### TU/TE 自体の編集

TU/TE 名を含め、ウィンドウ上でグレーアウトされていないものは変更ができます (図 11-20)。

- TU の Operator (条件) および Value (比較値)
- TE の Expression (TU の条件式)。“Max sequence Depth” と “Event count” の範囲内でのみ可能
- 有効な TE の選択 (TE 名の左側のチェックボックス)

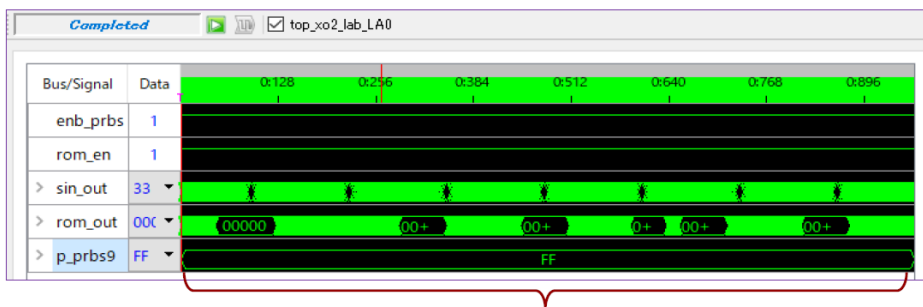
例：変更前：TU1 => 変更後：TU1 & TU2 & TU3

#### 有効 TU の選択

複数の TE が設定されている場合、有効な TE の指定ができます。Name セルにチェックが入っている TE のみが有効 (イネーブル) です。

トリガー発生条件を選択することができます。“Trigger Options” 部の「Enable TE」セルでトリガーがかかる条件を選択できます。[OR All] の場合はイネーブルされている TE の 1 つ以上が生起した場合に、[And All] の場合はイネーブルされている全ての TE が生起した場合に、それぞれトリガーがかかります。

図 11-21. Single モード時の波形表示例 (Number of Trigger=1)



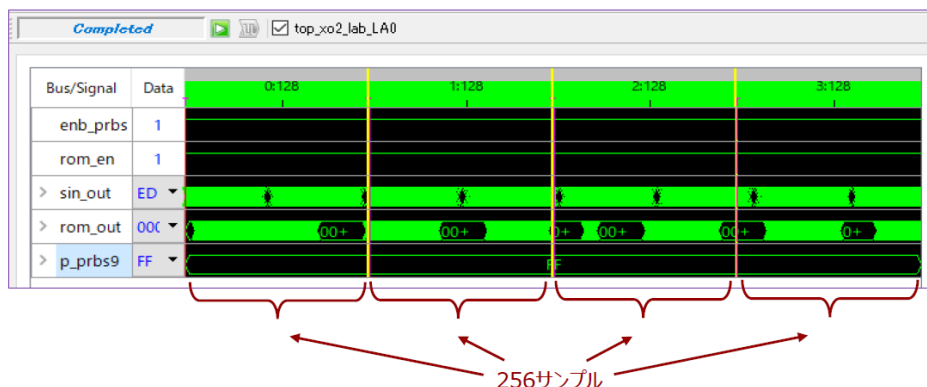
1024サンプル



### Multiple トリガーモードの設定

インサーターの設定時に、キャプチャー・モード（第 11.2.5.6 項）として Multiple モードを選択した場合、バッファを複数の領域に分割し、複数回のトリガーに対する波形キャプチャーを行うことができます（図 11-22）。通常を選択しない場合は図 11-21 のようになります。

図 11-22. Multiple モード時の波形表示例 (Number of Trigger=4)



設定は Trigger Options 部の「Samples Per Trigger」（トリガーごとのサンプル数）と「Number of Trigger」（トリガー数）で行います。それぞれ最大値はインサーターでの設定で決まります。

Multiple モードの場合、指定した回数トリガー条件が成立するまで波形は表示されません。また、マニュアルトリガーを掛ける場合も複数回のボタンクリックが必要です。

### トリガー表示位置の設定




波形表示の際に、トリガー条件が成立したサンプルの表示位置は「Trigger Position」で設定します。

「Pre-selected」では予め3つの表示位置決められています。

- ・ Pre-Trigger : トリガー位置は左（トータルサンプル数 × 1/16）
- ・ Center-Trigger : トリガー位置は中央（トータルサンプル数 × 8/16）
- ・ Post-Trigger : トリガー位置は右（トータルサンプル数 × 15/16）

「User-selected」でノブをドラッグすることで、表示位置を任意に設定することができます。Multiple モードでは設定にこの関わらず、Pre-Trigger 固定です。

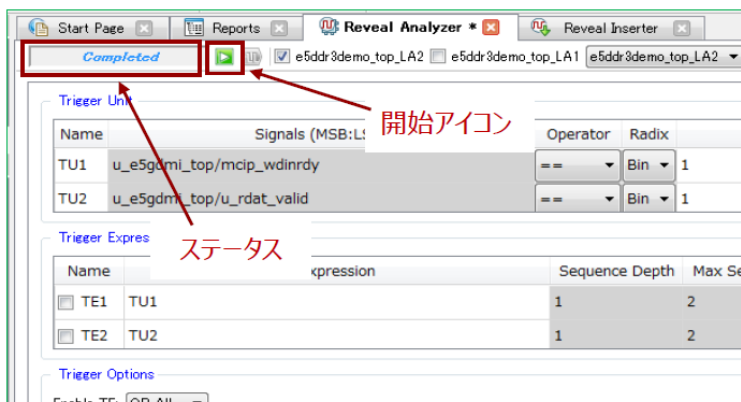
### 11.3.7 波形キャプチャーの開始

トリガー条件の設定を完了後、波形キャプチャーを開始するために、ウィンドウ左上にあるアイコン  を（図 11-23）クリックします。最初はデバイスへのトリガー条件の書き込み等が行われますので、トリガー条件の生起や波形のキャプチャー開始まである程度の時間が必要になります。トリガー待ちの時間は、開始アイコンが停止アイコン  に変わります（これをクリックして強制終了できます）。また、マニュアル・トリガーのアイコン  がアクティブになります（通常時はグレーアウト）。

LA の状態は実行アイコンの左側セルに表示されます。「Connecting」はコアとの通信と有効トリガ待ちを、「Completed」は波形表示完了を意味します。

以降に記述する表記指定なども含めて、トリガー条件など種々設定変更している場合、必ず「保存」しておきます。繰り返し行うことの多いデバッグ作業ですので、毎回同じ操作をする必要がなくなります。LA で作業後に保存まで行くと、Diamond プロジェクト・フォルダー下に三つのファイル（\*.rva/\*.rvs/\*.trc）が生成されます。LA 用これらファイルは、インプリメンテーションごとではなく、Diamond プロジェクトで一つのみ存在できます。

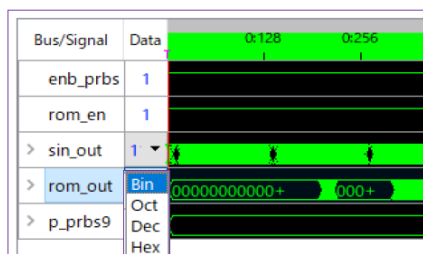
図 11-23. 波形キャプチャー開始アイコンとステータス



### 11.3.8 多ビット信号の表記指定

トリガー条件が生じ、所定の波形がキャプチャーされると [Waveform View] タブに表示されます。多ビット (バス) 信号の場合は [Data] カラムをクリックし、表示フォーマットを Hex/Oct/Dec/Bin から選択できます。

図 11-24. 表示フォーマットの変更

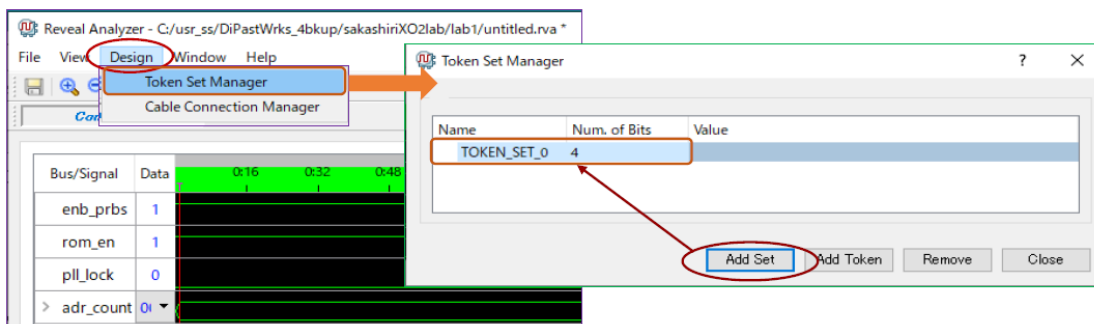


波形の拡大や縮小はウィンドウ上部のアイコンで行います。拡大・縮小ともに、カーソル基準ではなく、表示波形の中央を基準に行われます。

### 11.3.9 トークンセット・マネージャー

LA の波形表示において、特定のデータパターンを文字列に置き換えて表示することができます。データパターンと文字列の対応付けを設定するのが ” トークンセット・マネージャー ” (Token Set Manager) です。デタッチした LA のメニューバーから [Design] → [Token Set Manager] を選択して起動します (図 11-25)。

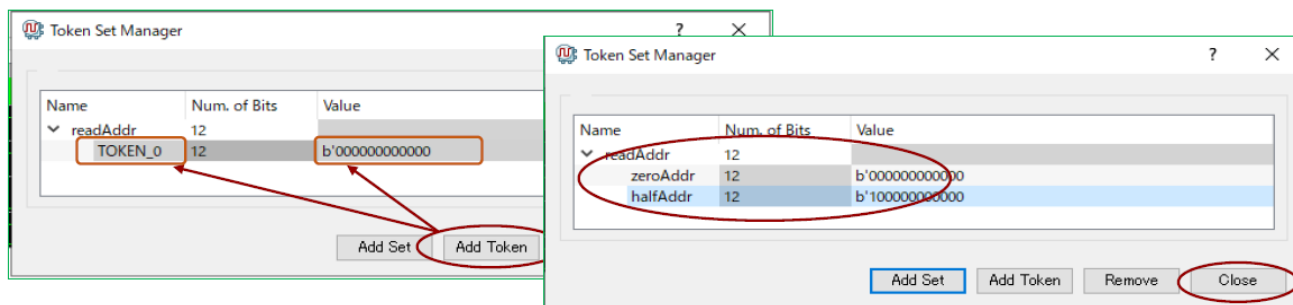
図 11-25. トークンセット・マネージャーの起動



表示されるウィンドウで、最初に 『Add Set』 ボタンをクリックして ” トークンセット ” を生成します。これは対応付けのグループを意味します。[Name] カラム、[Num. of Bits] カラムをクリックして、任意のトークン

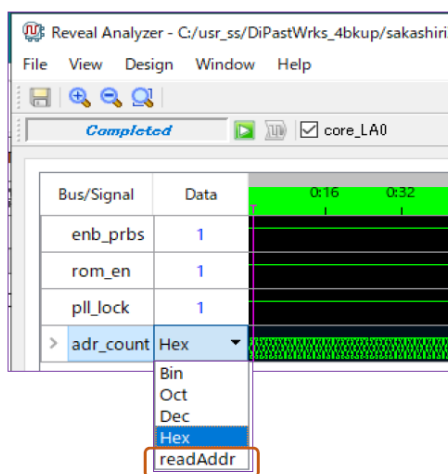
ンセット名とトレース信号のビット幅をそれぞれ入力します (図 11-26)。ビット幅が不一致だと、後述のように表示フォーマット指定ができませんので、ご注意ください。

図 11-26. トークンの追加・定義例



次に『Add Token』ボタンをクリックして、データパターンと文字列の対応づける ”トークン” を作成します。通常は一つのトークンセットに複数定義します。[Name] と [Value] カラムをクリックして、任意の名称と値を入力します (ビット幅はトークンセットで設定したもから変更できません)。全てのトークンを定義したら『Close』をクリックしてトークンセット・マネージャーを終了します。

図 11-27. 追加したトークンセット表記の選択例



トークンセット作成後、波形表示でデータの表示形式として作成したトークンセットが選択できるようになります (図 11-27)。トークンセットが選択肢として現れるのはビット幅が一致しているトレース信号のみです。またトークンセットを選択した場合、トークンに一致しないデータパターンについては、値が何も表示されなくなります。出現し得る全パターンのトークンセットを定義するようにし、また適用するトレース信号も注意深く決定するようにします。

## 11.4 Reveal による SERDES デバッグ

第 11.2.3 節で言及したとおり、PCS/SERDES を搭載する ECP5 には Reveal の SERDES デバッグ機能が対応しています。ロジックアナライザーとの共存も可能です。

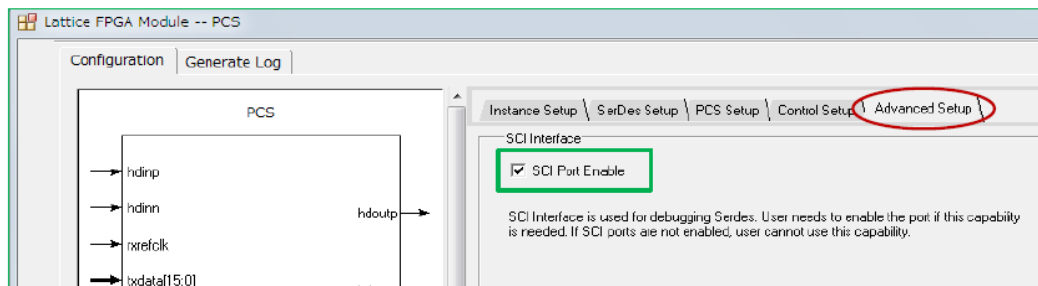
### 11.4.1 Clarity による PCS 生成時の設定

Reveal によるデバッグを可能にするためには、Clarity Designer で PCS/SERDES マクロを生成する際に、一点留意すべき事項が二点あります。

第一に、パラメータ設定 (Configuration) 時に [Advanced Setup] タブがありますが、「SCI Port Enable」オ

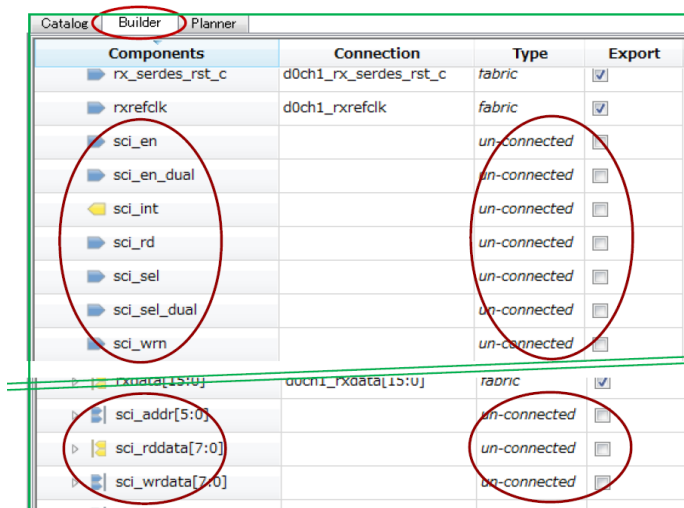
プシオンを必ずチェック（イネーブル）しておきます（図 11-28）。このインターフェイスは、Reveal が JTAG サーバーと通信する際に使用することになります。

図 11-28. Clarity での PCS SCI ポート設定



第二に、Clarity Designer の [Builder] タブでの設定です。イネーブルした SCI 各ポートを上位モジュールでの接続用に出すか、出さないかの設定を Builder で行うことができます。この用途では、のよようにすべての入出力を "un-connected" にします（"Export" しない：図 11-29）。Reveal インサーターがそれらの接続処理を自動的にバックグラウンドで行います。

図 11-29. Clarity での SCI ポートの Export 設定

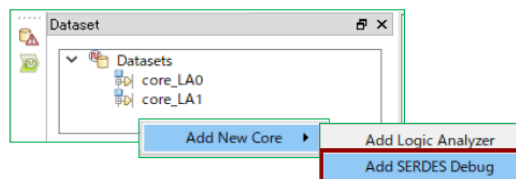


### 11.4.2 SERDES デバッグ・コアの挿入

インサーターの左上 "Dataset" 枠で右クリックすると表示されるメニューから "Add New Core" → "Add SERDES Debug" を選択します（図 11-30）。

インサーターでの設定はロジックアナライザーと比較して非常に簡潔で、二つしかありません。サンプル用クロックと、リセット信号を "Design Tree" からドラッグ&ドロップで指定します（図 11-31）。

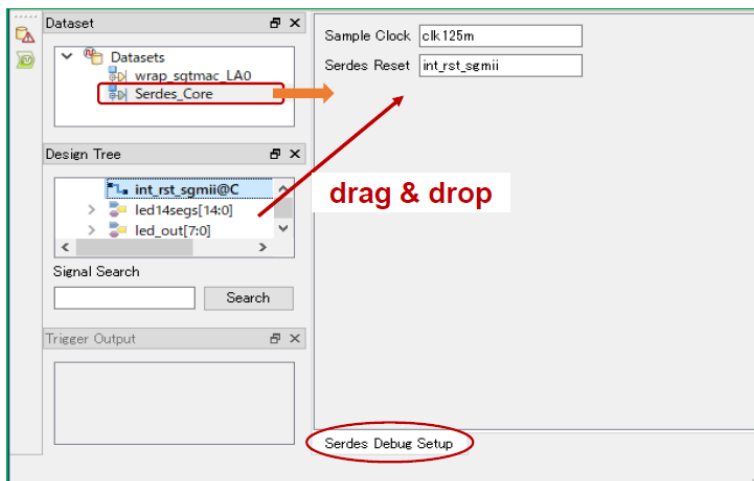
図 11-30. SERDES デバッグ・コアの挿入



指定が完了したら、ロジックアナライザーと同様に DRC アイコンをクリックして、挿入アイコンをクリ

ックします。

図 11-31. インサーターの SERDES セットアップ



### 11.4.3 SERDES アナライザー・タブ表示

ロジックアナライザーと同様の手順でFPGAをコンフィグレーション後にRevealアナライザーを起動します。

図 11-32. SERDES アナライザー・タブ表示の例

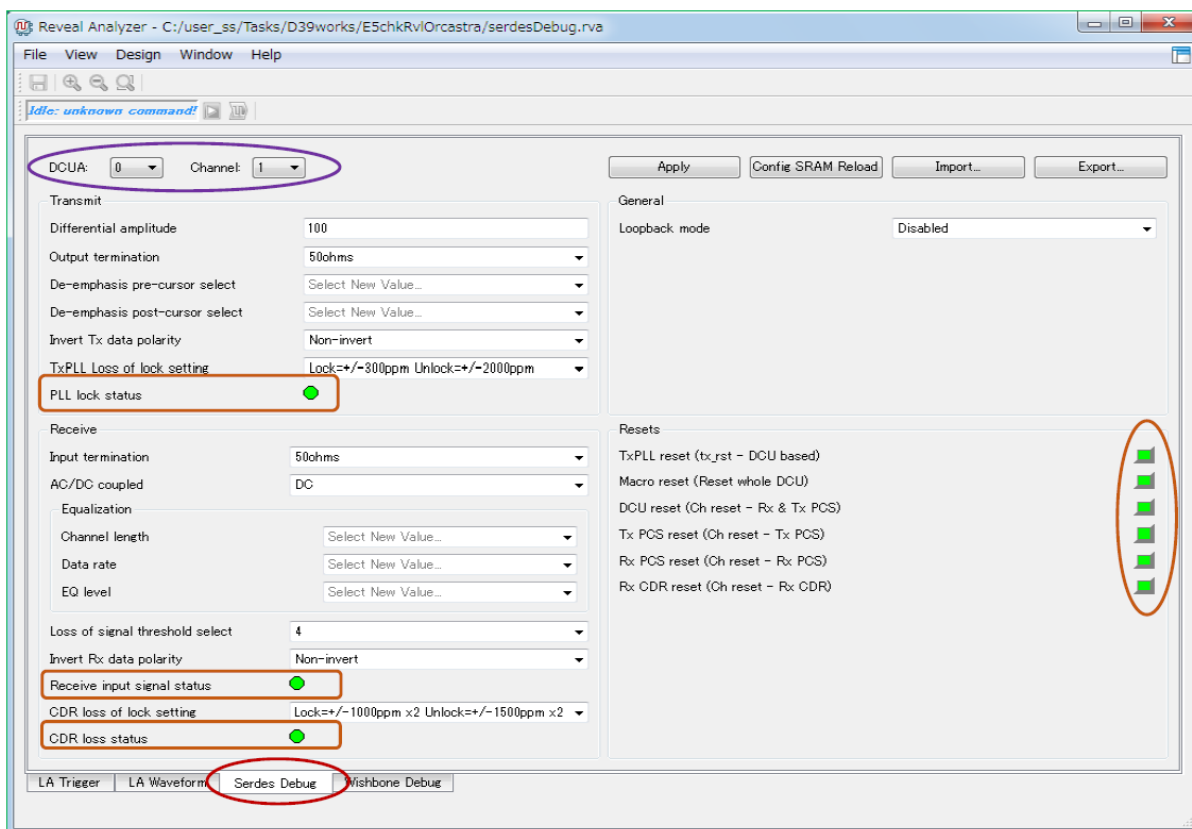


図 11-32 の例のように、正常動作だと Tx-PLL や Rx 入力信号の有無と CDR の各ステータスが緑色で表示

されます。

PCS 固有の各パラメータはそれぞれ有効な値がプルダウンから選択でき、右上部の『Apply』ボタンをクリックすることで、コンフィグレーションされている設定と異なるものに変更できます。

右下部は各種リセット信号の状態で、緑色はネゲートを示します。それぞれのインジケータはボタンになっていて、クリックする度に赤色・緑色が交互に変わります。リセットを実際に印加する操作は次の通りです (アサート→ネゲート)。

赤色にする→ Apply をクリック→緑色にする→ Apply をクリック

また、記述は割愛しますが、[Wishbone Debug] タブ表示では、より高度な内部レジスタへのアクセスも可能です。

--- \*\*\* ---