

Quartus[®] Prime はじめてガイド Signal Tap ロジック・アナライザの使い方

Ver.17.1

2018年1月 Rev.1

ELSENA

ELSENA,Inc.





Quartus Prime はじめてガイド Signal Tap ロジック・アナライザの使い方

<u>目次</u>

1. はじめに	3
2. 使用環境	4
2-1. 開発ソフトウェア	4
2-2. 通信ケーブル	4
2-3. 対応デバイス	4
3. 概要	5
3-1. 必要な FPGA の内部リソース	5
3-2. 観測できない信号	5
3-3. 作業フロー	6
3-4. 検証をする前に	6
4. 操作手順	7
4-1. デザインに Signal Tap を追加	7
4-1-1. 新規 STP ファイルを起動	7
4-1-2. デザインに STP ファイルを追加	9
4-2. Signal Tap の設定	10
4-2-1. クロックの設定	
4-2-2. 信号の登録	
4-2-3. サンプル容量の指定	13
4-2-4. RAM タイプの指定	13
4-2-5. バッファ・モードの選択	
4-3. トリガの定義	19
4-3-1. トリガ・コンディションの設定	19
4-3-2. トリガ・フロー・コントロールの設定	22
4-3-3. トリガ位置の指定	22
4-4. デザインのコンパイル	23
4-5. FPGA プログラミング	24
4-6. Signal Tap の実行	
4-7. キャプチャされたデータの表示・解析	25
4-7-1. タイム・バー	25
4-7-2. 検出データの変換	26
4-7-3. データ・ログの保存	27
改版履歴	



ELSENA

1. <u>はじめに</u>

この「Quartus Prime はじめてガイド」シリーズは、Quartus® Prime 開発ソフトウェア(以下、Quartus Prime) をはじめてご利用になるユーザ向けの資料です。

本資料は、FPGA を開発するフローの中で、主に下図の赤枠内の開発フェーズで参考になります。



現在の FPGA は大規模で高性能となっている反面、パッケージや未使用 I/O ピンの制限などにより、デバッ グは困難になっています。インテルではその現状を緩和するために、デザインを FPGA 上で動作させながら未 使用の I/O ピンを活用することなく内部信号の動作を検証できるオンチップ・デバッグ環境 "Signal Tap ロジッ ク・アナライザ" (以下、Signal Tap)を提供しています。

この Signal Tap 機能は、Quartus Prime に標準搭載されたオンチップ・デバッグ・ツールで、ロジック・アナラー ザやオシロスコープなどの外部測定装置を使用せずに、ボード上で動作する FPGA の内部信号の状況をキャ プチャおよび表示することができます。



無償の Signal Tap 用 IP コアをユーザ・デザインに組み込みデバイス内部に配置配線し、データをプログラミング後ボード上で動作させます。キャプチャされたデータはデバイスの内部メモリ領域に一度格納され、その後、 ボード上の JTAG ピンから通信ケーブルを経由して Quartus Prime に転送され、パソコンの画面上に波形表示



されます。Signal Tap は、1 つの JTAG チェーン内に複数の FPGA デバイスを含んだボード環境でも実行できま す。

詳しくは、下記ドキュメントをご覧ください。

[Design Debugging with the Signal Tap Logic Analyzer] (Quartus Prime Standard Edition)

[Design Debugging with the Signal Tap Logic Analyzer] (Quartus Prime Pro Edition)

なお この Signal Tap は、Quartus Prime 17.0 以前のバージョンで搭載されていた SignalTap II と同じ機能で す。ver.17.1 より名称が変更されました。

2. 使用環境

Signal Tap を使用するには、以下の環境が必要です。

2-1. 開発ソフトウェア

対応する開発ソフトウェアは以下のとおりです。

- Quartus Prime Lite Edition
- Quartus Prime Standard Edition
- Quartus Prime Pro Edition
- ※ Quartus Prime Lite Edition の場合は、16.0 以前のバージョンを使用する際 TalkBack 機能を有効にする 必要があります。詳細はこちらの <u>TIPS</u> をご覧ください。
- ※ Quartus Prime Programmer and Tools の場合は、解析のみ可能です。デバッグ用デザインの作成やコン パイルなどは行えません。
- 2-2. 通信ケーブル

対応する通信ケーブルは以下のとおりです。

- ・ インテル FPGA ダウンロード・ケーブル II (旧 USB-Blaster™ II)
- ・ インテル FPGA ダウンロード・ケーブル (旧 USB-Blaster)
- · インテル FPGA イーサネット・ケーブル (旧 EthernetBlaster II)
- ※ ダウンロード・ケーブルがオンボード(内蔵)タイプの評価ボードを使用する 場合は、これらの通信ケーブルは不要です。
- 2-3. 対応デバイス

対応するデバイスは以下のとおりです。

- · Stratix® シリーズ
- · Arria® シリーズ
- ・ Cyclone® シリーズ
- · MAX[®] 10







Intel' Quartus' Prime

Design Software

- ※ 内蔵メモリを持たない CPLD デバイス(MAX® V など)は Signal Tap 機能を使用できません。
- ※ 通信用として、ボード上に 10 pin コネクタを用意し、FPGA の JTAG ポートを接続しておく必要がありま す。

3. 概要

3-1. 必要な FPGA の内部リソース

Signal Tap でデバッグするためには、FPGA 内部に、ロジック・アナライザ・メガファンクションを構築するための "ロジック・リソース" と、キャプチャしたデータを格納するための "内部メモリ・ブロック" が必要です。その ため、ユーザ回路をインプリメントした残りの空きリソースがどのくらいあるかによって、構成できる Signal Tap の仕様、キャプチャできる信号本数やサンプリング数が変わります。

Quartus Prime の Signal Tap ロジック・アナライザ・ファイル (*.stp) には、構築した Signal Tap ファンクション がどのくらいのロジック・リソースとメモリ・ブロックを必要とするかを自動的に推計する機能が備わっており、 Instance Manager ペイン内 (STP ファイルの左上) で確認ができますので、この参考値を確認してください。な お、実際のリソース使用率と 10% 程度異なる場合があります。

Instance Manager: 📉 🎉) 🔳 🛄 🚺	nvalid JTAG configur	ation			
Instance	Status	LEs: 575	Memory: 24576	Small: 0/29040	Medium: 3/462	Large: 0/16
🚷 auto_signaltap_0	Not running	575 cells	24576 bits	0 blocks	3 blocks	0 blocks
	ロジック ・必要 リソ・ no-fit	♪・リソース リソースが空きロ ースを超えたり と表示される。	↓ メモリ・リソ ジック・ 易合は	ース メモリ・フ ・インス が指す 算出さ ・必要リ 数を起 れる。	ブロックのタイプ別 、タンスを作成する 定したメモリ・ブロ られる。 リソース数が空き。 超えた場合は、Car	のリソース数 5際に、ユーザ ックのタイプで メモリ・リソース n't fit と表示さ

3-2. 観測できない信号

デザインの配置配線後 (post-fitting) に存在する信号すべてが Signal Tap で観測できるわけではありません。 以下の信号は観測することはできません。

- ・ JTAG (TCK、TDI、TDO、TMS) コントロール信号
- ・キャリーチェーンのキャリーアウト信号
- ・ 配置配線後の出力ピン(双方向ピンを含む)
 - ~ 出力ピンの信号を観測するには、出力ピンを駆動するレジスタまたはバッファをタップします。
- ・ ALTGXB IP コアのインスタンシェーションの任意ポート
- ・ LVDS のシリアライザ/デシリアライザ (SERDES) ブロックからのデータ出力
- ・ DDR / DDR2 デザインの DQ、DQS 信号





3-3. 作業フロー

Signal Tap の作業フローは以下のとおりです。



3-4. 検証をする前に

Signal Tap はユーザ・デザインの内部信号を観測するには、事前に観測用のトリガとして用いたい信号の指定 と、観測したい内部信号の指定を行います。そのため、最低限 Analysis and Elaboration を実行している必要が あります。(推奨はフル・コンパイルを実行)

Analysis and Elaboration は以下のメニューで実行します。

Processing メニュー ト Start ト Start Analysis and Elaboration

ただし、インクリメンタル・コンパイルやラピッド・リコンパイルを採用したコンパイルを実行しているプロジェクト を Signal Tap で検証する場合には、該当するコンパイル手法で実行を完了しておいてください。



4. 操作手順

4-1. デザインに Signal Tap を追加

Signal Tap の IP コアは、ユーザ・ロジックと共にデバイスへ実装させるため、ユーザ・デザインに追加する必要があります。追加方法は以下の2通りです。

- Signal Tap ファイル (*.stp) を作成し、ユーザ・デザインに自動でインスタンスする
- IP Catalog から Altera SignalTap II Logic Analyzer を選択および作成し、ユーザ・デザインに手動でインスタンスする

この資料では、Signal Tap ファイル (以下、STP ファイル)を使用する方法をご紹介します。

4-1-1. 新規 STP ファイルを起動

File メニュー ▶ New をクリックし、Verification/Debugging Files カテゴリ内より "Signal Tap Logic Analyzer File" を選択します。

プロジェクトに既存登録されている STP ファイルがない場合には、Tools メニュー ➤ Signal Tap Logic Analyzer からも新規 STP ファイルが起動します。

S New	×
EDIF File	*
Qsys System File	
State Machine File	
SystemVerilog HDL File	
Tcl Script File	
Verilog HDL File	
VHDL File	
4 Memory Files	
Hexadecimal (Intel-Format) File	
Memory Initialization File	
 Verification/Debugging Files 	
In-System Sources and Probes File	=
Logic Analyzer Interface File	
Signal Tap Logic Analyzer File	
University Program VWF	
4 Other Files	
AHDL Include File	
Block Symbol File	
Chain Description File	
Synopsys Design Constraints File	
Text File	-
OK Cancel Help	



[→] プロジェクトに STP ファイルが既存登録されて いる場合は、その STP ファイルが起動します。





6 つのペインで構成された STP ファイルが表示されます。

📮 🤊 ୯ 🏁 🐽 ice Manager: 📉 👂 (Invalid JTAG	configuration					×	JTAG Chain Confi	iguration: No devices detected	
ce auto_signaltap_0	Status Not running	Enabled LEs	0 Memory: 0	Small: NA	Medium: NA NA	Large: NA NA		Hardware: Device: No	JTAG Chain Configuration	Setup. Scan Ch
o_signaltap_0 Node e Alias uble-click to add nodes	Name	Lock mode: Data Enable O	Allow all changes Trigger Enable Trigger Co 0 1 1 Basic /	▼ nditions AND ▼				Signal Configurat Clock Data Sample depth:	128 RAM type: Auto	
			Node List					Nodes Alloc Pipeline Factor Storage qualit Type:	Signal Configuration	
Data 😹 Setup	× 🗐 C	bata Log: 🔄 auto_signalta	<u>_</u> 0							
Hierarchy Di	splay						Data Log		I	

Instance Manager ペインに auto_signaltap_0 という名称の Signal Tap IP のインスタンスが自動的に作成され ます。インスタンス名を変更する際は、インスタンスを選択して右クリック ➤ Rename Instance より行います。

Instance Manager: 📉 😥 🔳 🛄 Invalid JTAG configuration						
Instance	Status	Enabled	LEs:			
auto_signaltap_0	Create Instance	Del	736			
	Detete instance	50				
Image: Image	Rename Instance	F2				

Signal Tap IP コアは、1 つの FPGA に複数インスタンスすることが可能です。Signal Tap IP コアを追加する場合 は、Instance Manager ペイン内で右クリックし、Create Instance を選択してください。複数インスタンスすると、 Signal Tap IP コアを構成するためのロジック・エレメントやメモリの消費が増加するため、空きリソース容量の範囲 で作成する必要があります。

各インスタンスは、別々のクロックを用いられることや同一クロック・ドメインに対して検証する信号ごとにイン スタンスを割り当てることができるほか、個々にデバッグが実行できるので、ユーザのニーズに合ったデバッグ 方法が実現できます。操作がアクティブなインスタンスは、アイコンの波形が赤く表示され、ボックスで囲われま す。各インスタンスの操作切り替えは、Hierarchy Display ペインのタブにより指定します。

	Instance Manager: 📉 😥 🔳	Invalid JTAG	configurati	on	
	Instance	Status	Enabled	LEs: 1067	Memory: 6144
	auto_signaltap_0	Not running	V	556 cells	5120 bits
ſ	🔶 🛃 auto_signaltap_1	Not running	V	511 cells	1024 bits
	—— 非アクティブなインスタンス —— アクティブなインスタンス		Er ⊐ t	- nabled オプションを ンパイル時プロジェ れます。	と有効にしたインスク ェクト(デザイン)にイ



4-1-2. デザインに STP ファイルを追加

Signal Tap IP を駆動させるためのクロックやサンプリング信号の登録、トリガ条件の設定などがまだ行われて いませんが、先に STP ファイルに名前を付け保存し、プロジェクトに登録しておきます。

ユーザ・デザインと Signal Tap IP のポート接続は、STP ファイルを作成しプロジェクトに登録することで Quartus Prime が自動的にユーザ・デザインと Signal Tap IP を接続してくれます。

STP ファイル内の File メニュー ➤ Save As... において、保存するフォルダおよびファイル名(任意)を指定し、Add file to current project オプションを有効にして "保存(S)" ボタンをクリックします。

🕥 Quartus Prime 🛛 🕰	
Input "Data and Trigger" is empty	STP ファイルに何も入力していない場合には Input "Data and Triager" is empty とかけージ(左図)が
ОК	表示されます。

Do you want to enable Signal Tap File "<任意ファイル名>.stp" for the current project? とメッセージが表示されます。Yes ボタンをクリックし、STP ファイルをプロジェクトに登録します。

🕞 Quar	tus Prime
	Do you want to enable Signal Tap File " .stp" for the current project?
	Yes No Cancel

この操作により、Signal Tap Logic Analyzer オプション(Assignments メニュー ➤ Settings)を自動的に有 効にし、ユーザ・デザインにインスタンスする STP ファイルを指定したことになります

General	Signal Tap Logic Analyzer
Files	Specify compilation options for the Signal Tap Logic Analyzer.
Libraries	
IP Settings	Enable Signal Tap Logic Analyzer
Design Templates	Signal Tap File name: Sfile name> stp
Operating Settings and Conditions	alline reported the market and and and and
Compilation Process Settings	
EDA Tool Settings	
Compiler Settings	
TimeQuest Timing Analyzer	
Assembler	
Design Assistant	



4-2. Signal Tap の設定

STP ファイルに、Signal Tap IP 用クロックや観測する信号、およびトリガとして用いる信号などを登録します。

4-2-1. クロックの設定

Signal Tap IP のクロックを指定します。

Signal Tap IP はクロックの立ち上がりでデータをサンプリングします。立ち下りクロックでのサンプリングはサ ポートされていません。このクロックは、デザイン内のどの信号でも選択することができますが、最良の結果を 得るため、サンプリングする信号に同期した、かつグローバル化されたクロックの使用を推奨しています。また、 サンプリングするデータに対し2倍以上の周波数のクロックを割り当てると良いでしょう。

- ① STP ファイルの Setup タブをクリックします。
- ② Signal Configuration ペインの Clock 欄右のブラウズ・ボタンをクリックし、Node Finder を表示させます。

Signal Configuration:			>	×		
Clock				•		
Data				E		
Sample depth: 128	🟸 Node Fin	der	· · · · · · · · · · · · · · · · · · ·			×
	Named:	*				List
	Options					
	Filter:	Signal Tap: post-fitting				Customize
	Look in:	st_demo			▼ Include subentities	Ilierarchy view
	Matching	Nodes:			Nodes Found:	
		Name	Assignments		Name	Assignments
	<	ш		~ × ×	۲	•
					OF	Cancel

- ③ Node Finder ウィンドウ内 Filter 欄のプルダウン・リストから、Signal Tap IP 用クロックとして採用する信号のタイプ(以下参照)を選択し、List ボタンをクリックします。
 - ・ インクリメンタル・コンパイルを採用していない場合 ⇒ Signal Tap: pre-synthesis を選択
 - ・ インクリメンタル・コンパイルを採用している場合 ⇒ Signal Tap: post-fitting を選択

選択するクロックがユーザ・デザインにおいて深い階層下にある場合は、Look in 機能により階層を指定して信号を検索することが可能です。

Look in 欄右のブラウズ・ボタンをクリックし、Select Hierarchy Level ダイアログ・ボックスから目的のエン ティティを選択すると、指定した階層名が Look in 欄に表示されます。その後 Node Finder の List ボタ ンをクリックすると、目的の信号が見つけやすくなります。

Node Finder ウィンドウの Named 欄にはワイルドカードの使用が可能ですので、組み合わせて検索 するとより検索がスムーズになります。



- ④ 検出された Matching Nodes リストの信号から、クロックにアサインしたい信号をダブルクリックで指定し、 Nodes Found リストに選出します。
- ⑤ OK ボタンをクリックし、Signal Tap IP のクロックを登録します。

エントリされた信号の文字色は、Post-fitting 信号は青、Pre-synthesis 信号は黒で表示されます。

Signal (Configuration:
Clock:	component pll_altpll:auto_generated wire_pll1_clk[1]~clkctrl

4-2-2. 信号の登録

観測する内部信号を選択します。選択した信号はトリガ条件にも使用できます。ただし Signal Tap IP 用のクロックに指定した信号は観測できません。

- ① STP ファイルの Node List の空白部分をダブルクリックし、Node Finder を起動します。
- ② Node Finder ウィンドウ内 Filter 欄のプルダウン・リストにおいて、以下のいずれかを指定します。
 - ・ インクリメンタル・コンパイルを採用していない場合 ⇒ Signal Tap: pre-synthesis を選択
 - ・ インクリメンタル・コンパイルを採用している場合 ⇒ Signal Tap: post-fitting を選択

選択するクロックがユーザ・デザインにおいて深い階層下にある場合は、Look in 機能により階層を指定して信号を検索することが可能です。

Look in 欄右のブラウズ・ボタンをクリックし、Select Hierarchy Level ダイアログ・ボックスから目的のエン ティティを選択すると、指定した階層名が Look in 欄に表示されます。その後 Node Finder の List ボタ ンをクリックすると、目的の信号が見つけやすくなります。

Node Finder ウィンドウの Named 欄にはワイルドカードの使用が可能ですので、組み合わせて検索 するとより検索がスムーズになります。

Note: STP ファイルに観測したい、あるいはトリガとして用いたい内部信号 (Signal Tap: post-fitting) を選出する際、Node Finder を使用せずに Technology Map Viewer (Post-Fitting) を活用することも可能です。ユーザ・デザインの内部信 号を視覚的に検索し、STP ファイルにインポートさせます。詳細は、下記 TIPS をご参考ください。

【TIPS】 SignalTap® II でキャプチャしたい内部信号を簡単に STP ファイルに登録する方法

後出された Matching Nodes リストの信号から、選出したい信号名を Nodes Found リストに登録します。
 キーボードの Shift または Ctrl キーを使用して、一度に複数の信号を選択することができます。

Node Finder にリストアップされる信号名の左側に表示されるピンスタブ(マーク)は、信号の種類を表しています。

ピンスタブ	種類
	入力ピン信号
out	出力ピン信号
	組み合わせ信号
R.	レジスタ出力信号
-	バス信号(ピンスタブが重なっている)



また、カラムを右クリックで選択し項目を追加することができます。Type では信号の種類、Creator で はユーザ作成の信号名か Quartus Prime のコンパイラが生成した信号名かを確認することができます。

1	Matching Nodes:			E		Nodes Fou	ind:
	Name	Assignments	Туре	Creator			
	🖕 Equal0~0	Unassigned	Combinational	Compiler generated	\mathbf{V}	Assignments	
	-QUARTUD_GND~I	Unassigned	Combinational	Compiler generated	×,	Type Creator	
	bcounter.bcounter_inst C c				<u> </u>		
		Unassigned	Combinational	Compiler generated		>>	
	✓ cnt	onassigned	Compinational	compiler generated		<	
	k cnt[0]	Unassigned	Registered	User entered		<<	
	👆 cnt[1]	Unassigned	Registered	User entered			

④ Insert ボタンをクリックし、信号を登録します。Node List エリアに表示された信号の文字色は、Post-fitting 信号は青色、Pre-synthesis 信号は黒色を示しています。コンパイル後に存在しない信号は赤色で表示さ れ、検証に使用できません、STP ファイルから削除してください。

trigger: 🛙	רער הער היו	18 DB#7#2 #1	Lock mode:	🔒 Allow all ch	ll changes	
		Node	Data Enable	Trigger Enable	Trigger Condition	
Туре	Alias	Name	18	18	1 🗹 Basic AND	
*	One	무unter:Count_Ones cntr_dsi:auto_generated counter_reg_bit[30]	信号	をバス化し、ツ	リー表示する	
×.		unter:Count_Ones cntr_dsi:auto_generated counter_reg_bit[3]	ح_ ا⊷	ができる。(信-	号を複数選択	
×		unter:Count_Ones cntr_dsi:auto_generated counter_reg_bit[2]	L.7	与クリック ▶ G	roup)	
*		unter:Count_Ones cntr_dsi:auto_generated counter_reg_bit[1]			88	
*			V	V		
👘 🌜	Ten	unt unt	1	V	Xh	
5		Prescale Alias 欄をダフルクリックし、別名をつけることができる) o 🔽	V		
5		Prescaler:inst2 AndFinal~5	V	V		
5		Prescaler:inst2 AndFinal~4				
5		svnseg:Decode_Ones a~12 🤸 ドラッグ&ドロップで順番	の入れ替え	が可能。		
<u> </u>		svnseg:Decode_Ones b~3	v	v		
<u> </u>		synseg:Decode Onesic~1	V	V		
🥦 Data	86 S	Setup				

ユーザ・デザインにおいて多くの信号は、Quartus Prime のコンパイラにより論理合成また配置配線で 最適化が図られます。これにより、RTL で記述した信号名とは異なってしまい、ユーザは Signal Tap で検 証したい信号を検索することが困難になるかもしれません。インクリメンタル・コンパイルを使用した場合 も同様です。その場合、先にも記載した Technology Map Viewer の活用が有効的です。

その他の対策として、論理合成および配置配線時に信号を保持させるアトリビュートを採用する方法が あります。

- ・ Keep : 指定した組み合わせ信号が削除されないようにします。
- · Preserve : 指定したレジスタが削除されないようにします。

このアトリビュートは、ユーザ・デザインに対してオプションとして採用する方法と HDL 記述に直接宣言する方法のいずれかで適用させます。ただしこの方法はコンパイラによる最適化が行われないため、リソースの使用率が増加したり、タイミング性能が低下する場合があります。これらのアトリビュートの使用については、メーカのドキュメント [Intel Quartus Prime Integrated Synthesis] を参照してください。



STP ファイルに登録した信号には Enable オプションがあり、各々使用方法を指定することが可能です。 いずれのオプションも、デフォルトは有効(ON)です。

- Data Enable : このオプションをディセーブル(OFF)にすると、指定した信号のデータを非表示に します。例えば、信号はトリガに用いるのみで、その信号自体のデータは表示しなくて良い場合に 設定します。これにより、Signal Tap IP で使用される FPGA 内部メモリの使用率を削減することが できます。
- Trigger Enable: このオプションをディセーブル(OFF)にすると、トリガ機能を無効にします。キャプ チャしたデータのみを表示し、その信号をトリガの一部として使用しない場合に設定します。

	Node	Data Enable	Trigger Enable	Trigger Condition
Alias	Name	18	18	1 🗹 Basic AND
One		V	V	Xh
Ten		V	V	Xh
	Prescaler:inst2 AndFinal~6	V	V	
	Prescaler:inst2 AndFinal~5	v	V	
	Prescaler:inst2 AndFinal~4	V		
	Alias One Ten	Node Alias Name One Imagenerated/counter_reg_bit[30] Ten Imagenerated/counter_reg_bit[30] Ten Imagenerated/counter_reg_bit[30] Prescaler:inst2/AndFinal~6 Prescaler:inst2/AndFinal~5 Imagenerated/counter_reg_bit[30] Imagenerated/counter_reg_bit[30]	NodeData EnableAliasName18OneImmediatedicounter_reg_bit[3]Immediatedicounter_reg_bit[3]TenImmediatedicounter_reg_bit[3]Immediatedicounter_	NodeData EnableAliasName18One

4-2-3. サンプル容量の指定

サンプル容量とは、各信号に対してキャプチャおよび格納されるサンプルの数です。Signal Configuration ペイン内 Data エリアの Sample depth において、プルダウン・リストから格納する希望サンプル数を選択します。サンプル容量の範囲は 0~128K です。

FPGA に搭載されたメモリ・リソースの空き容量が、選択したサンプル・バッファ・サイズではデザインをコンパイルできない場合があります。そのときはサンプル容量を少なくしてリソースの使用率を低減してください。

Signal Configurati	ion:				
Clock: clock~inp	outclkctrl				
Data					
Sample depth:	2 К	> •	RAM type:	М20К/М10К/М9К/М4К	-
Segmented:	256 512	^			-
Nodes Allocated	1 K		Manual:	18	*
Pipeline Factor:	2 K 4 K				•

4-2-4. RAM タイプの指定

観測データを格納する内部メモリ・ブロックの種類を指定します。Signal Configuration ペイン内 Data エリアの RAM type において、プルダウン・リストから選択します。(ターゲット・デバイスのアーキテクチャにより、RAM タ イプが指定できないファミリもあります。)

もし、ユーザ・デザインに大容量メモリ・アプリケーションがある場合、Signal Tap で使用する RAM タイプを小 規模あるいは中規模メモリ・ブロックに割り当てることで、大規模なメモリ・ブロックをユーザ・デザインのために 有効活用することができます。





Signal Configurati	on:			
Clock: clock~inp	utclkctrl			
Data				
Sample depth:	2 К	RAM type:	М20К/М10К/М9К/М4К	$\overline{}$
Segmented:	2 1 K sample segments		Auto MLAB/LUTRAM/M512	
Nodes Allocated	l: 🔘 Auto	O Manual:	M20K/M10K/M9K/M4K	
Pipeline Factor:	0		M144K/M-RAM	•

ーつのインスタンスで RAM タイプを混在させることはできません。インスタンスを複数作成した場合は、イン スタンスごとに RAM タイプの指定が可能です。インスタンスを複数作成する方法は、本資料 <u>4-1-1. STP 新規</u> <u>STP ファイルを起動</u> をご覧ください。

RAM タイプを指定すると、Instance Manager ペインに使用されるメモリの容量とブロック数が自動計算され表示されます。

In	stance Ma	inager: (R 🔊 🔳	Invalid JTAG c	onfiguration							×	JTAG Chain Configura
Ins	itance 🕄 auto 🕈 auto	_signalta _signalta	ap_0 ap_1	Status Not running Not running	Enabled LE	s: 1434 8 cells 6 cells	Memory: 94208 36864 bits 57344 bits	Small: 0/0 0 blocks 0 blocks	Medium: 12/30 5 blocks 7 blocks	Large: 0/0 0 blocks 0 blocks			Hardware: Device: None Dete
	trigger: 20	017/11/1	16 09:47:42 #1		Lock mode	: 🥂 Allow all c	hanges	▼			s	ignal C	Configuration:
۱ſ			Node		Data Enabl	Trigger Enable	Trigger Condition	ons					
I	Туре	Alias		Name	18	18	1 🗹 Basic AND					lock:	clock~inputclkctrl
	*	One		d counter_reg_bit[3	0]	V	Xh				^	Data	
	đ	Ten		d counter_reg_bit[3	0]	V	Xh						
	•		Prescaler:inst2	2 AndFinal~6	\checkmark							Samp	ole depth: 2 K
	-		Prescaler:inst2	2 AndFinal~5	\checkmark	V						Se	egmented: 2 1 K samp
	5		Prescaler:inst2	2 AndFinal~4	V	V							

4-2-5. バッファ・モードの選択

キャプチャしたデータ・バッファの収集方法を選択でき、データ収集に必要なメモリ容量を潜在的に低減することができます。収集方法は以下の2つです。

Circular (Non-segmented) バッファ	デフォルトのバッファ・タイプ。全てのメモリ領域を一つの FIFO とみなし、ロジック・アナライザが定義したトリガが発生するま で、連続的にバッファを満たします。トリガ発生後もバッファがい っぱいになるまでデータのキャプチャを継続します。
Segmented バッファ	メモリ領域をセグメントに分割し、個々のセグメントは単体の Circular バッファのように機能します。周期的に発生するトリガを キャプチャするのに適した方法です。
Pos Newly Captured	t-Trigger Pre-Trigger







バッファ・モードは、Signal Configuration ペイン内 Data エリアの Segmented オプションで選択します。

Data		
Sample depth: 8 K 🔻	RAM type:	М20К/М10К/М9К/М4К 🔻
Segmented: 24K san	• OFF → • ON →	Circular バッファ・モード Segmented バッファ・モード

この資料では、Circular バッファ・モードに関して説明します。Segmented バッファ・モードの詳細は、下記ドキュメントをご覧ください。

[Design Debugging with the Signal Tap Logic Analyzer] (Quartus Prime Standard Edition)

[Design Debugging with the Signal Tap Logic Analyzer] (Quartus Prime Pro Edition)

Circular バッファ・モードでは、Storage Qualifier 機能を選択して、限られたバッファ・スペースをより有効的に 活用できます。Signal Configuration ペイン内 Storage qualifier エリアで指定できるオプションの種類は以下のと おりです。

Storage qualifier:				
Туре:	Continuous 🗸			
la sub sub t	Continuous			
input port:	in_ Input port			
Nodes Alloo	X Transitional			
	👯 Conditional			
Record of	🚟 Start/Stop			
Disable	State-based			

Storage Qualifie	er の種類
Continuous	全サンプルをストア。
Input port	デザイン内の指定した信号がクロック・エッジに伴い High のときにサンプルをス トア。(バッファのライトイネーブルとして使用)
Transitional	選択された信号の遷移時にサンプルをストア。
Conditional	選択した信号のブール式が真のときサンプルをストア。
Start/Stop	スタートに指定した信号が真のときからストップに指定した信号が真になるまでサ ンプルをストア。
State-based	ステートマシンのフローを制御し、ステート状態が真のときにサンプルをストア。 (詳細は <u>Trigger Condition Flow Control</u> をご覧ください。)



FISE

また、Storage qualifier エリアに設けられたオプションを併用することも可能です。

Storage qualifier:							
Type: 🚟 Start/Stop							
Input port: auto_stp_external_storage_qualifier							
Nodes Alloo	Nodes Allocated: Auto Manu Record data discontinuities オプション 法取りて支援また(第555年までする) 						
Record of	data discontinuities		・Disable storage qualifier オプション				
Disable	storage qualifier		検証実行時に Storage Qualifier 機能を無効にする				

以下に、各 Storage Qualifier Type の波形サンプル(State-based を除く)を表示します。State-based モードに ついては、Trigger Condition Flow Control をご覧ください。



指定した信号の論理値が High 時のデータをストアします。

Storage qualifier:					
Type: Input port					
Input port: auto_stp_external_storage_qualifier					
Nodes Allo	Nodes Allocated: Auto Manual:				
Record data discontinuities					
Disable storage qualifier					

Continuous Mode

											click to inse	ert time bar					
	N	iode	0							~ ~ ~ ~							
Туре	Alias	Name		2	2	. 4	6	. 8	10	12	. 14	16	18	20	22	24	26
0		⊕ data_out	(1Eh	(1Fh)	1h (02h	X03h X04	h (05h (06h	(07h)(08h)	(09h)(0Ah	(OBh (OCh	(ODh (OEh)	(0Fh (10h)	(11h)(12h)	(13h)(14h)	(15h)(16h)	17h (18h)	19h (1Ah)
	Storage Qualifier	data_out[7]															

Input Port Storage Qualifier

													click to inse	ert time ba						
	ŀ	lode	0		2	i.			4	,	6				7		9+			
Туре	Alias	Name		0	1	2 4	6		8	10	17	12	14	16	. 18		20	. 22	24	26
		⊞ data_out	(10)	1011	h)(07h	XOAhXOBhXO	Ch (ODh (OEh XOFH	1)(10h)	01h	7h (0A	h(08h)	OCh (ODh)	OEh XOFH	(10h)	01h (0	7h XOAP	08h/00	h)(0Dh)(0B	h XOFh X10
-	Storage Gualifier	data_out[7]																		

◆ Transitional モード

モニタ対象に指定した信号のいずれかが変化した場合にデータをストアします。



Quartus Prime はじめてガイド – Signal Tap ロジック・アナライザの使い方

ES

	Node	Data Enable	Trigger Enable	Storage Enable	Storage Qualifie	Trigger Conditions	Signal comparation.
ype Alias	Name	7	8	9	Transitional	1 ✓ Basic OR 💲	Clock: I[0].mod 1]clk
*	d altsyncram1 address_reg_a[0]		4			100	
*	d altsyncram1 address_reg_a[1]	4		4	~		Data
R	I[0].mod_1 CRC[3].mycrc lfsr_q[0]	4	4	2	4		Sample depth: 128 RAM type: Auto
R	I[0].mod_1 CRC[3].mycrc lfsr_q[1]	4	4	1	4	12	Sample deput. 120 • Rouri type. Hato
1	I[0].mod_1 CRC[3].mycrc lfsr_q[2]		~	4		8	Segmented: 2 64 sample segments
R.	I[0].mod_1 CRC[3].mycrc Ifsr_q[3]		4	4	4	111	
8.	I[0].mod_1 CRC[3].mycrc lfsr_q[4]		4	4	4		Nodes Allocated: Auto Manual: 7
R.	I[0].mod_1 CRC[3].mycrc lfsr_q[5]		4	4	4	1	Photo Photo I
R	I[0].mod_1 CRC[3].mycrc lfsr_q[6]	Y	Y	4	4	100 H	Pipeline Factor: 1
L.	I[0].mod_1 CRC[3].mycrc lfsr_q[7]	4					Storage qualifier:
5	mod_1 CRC[3].mycrc lfsr_c[0]~0	4					
C	mod_1 CRC[3].mycrc lfsr_c[0]~4	4		v	4		Type: X Transitional 🗘

・ モニタする信号を選択

Continuous Mode

														c	lick to	insert	time b	ar .							
		Node	0																						
Туре	Alias	Name	-2	-	1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
62			ID	LE	(00h	X 011	1 X 02	h (03	h X04h	X 05h	X 06h	X07h	X 081	X 09	h X 0.	Ah X						IDLE			_

Transitional Mode

													click	to inse	ert time	bar									
	No	de	0	1											2	8									
Туре	Alias	Name	-2 .	1	9	1 :	2 :	3	4 1	5	6 1	7 8	5	1	0 1	12	13	3 1	4 1	5 '	16 1	17	18	19	20 2
			IDLE	00h	01h	02h	(03h)	(04h	(05h	(06h	(07h	(08h)	(09h)	(OAh)	(IDLE)	00h (01h)	02h	(03h)	(04h	(05h	06h	X 07h	1 (08h	(09h)

◆ Conditional モード

特定の条件成立時にデータをストアします。

trigger: 20	7/04/11 15:21:05 #1	Lock mode:	Allow a	ll changes	\$		Signal Configuration:	
	Node	Data Enable	Trigger Enal	ole Storage Enable	Storage Qualifier	Trigger Conditions		-
ype Alias	Name	11	8	9	Basic AND 😂	1 🖌 Basic OR 💲	Pipeline Factor: 1	÷
¥.	d altsyncram1 address_reg_a[0]		4			88	Storage qualifier:	
	d altsyncram1 address_reg_a[1]	4		~				_
	[0].mod_1 CRC[3].mycrc lfsr_q[0]	4	~	~			Type: 🕸 Conditional	\$
R	[0].mod_1 CRC[3].mycrc lfsr_q[1]	4	~	4	0			
	[0].mod_1 CRC[3].mycrc lfsr_q[2]	4	4	4	2		input port: [auto_ctp_external_storage_dualmer]	
	[0].mod_1 CRC[3].mycrc[lfsr_q[3]	Y	4	Y	5		Nodes Allocated: @ Auto O Manual: Q	-
•	[0].mod_1 CRC[3].mycrc lfsr_q[4]	4	~	*	1		Houes Allocated. 19 Auto 10 Mardat.	12
-	i[0].mod_1 CRC[3].mycrc lfsr_q[5]	~	4	4	x	88	Record data discontinuities	
•	[0].mod_1 CRC[3].mycrc lfsr_q[6]	4	4					
	I[0].mod_1 CRC[3].mycrc lfsr_q[7]	~					Disable storage qualifier	
	mod_1 CRC[3].mycrc lfsr_c[0]~0	4						
	mod_1 CRC[3].mycrc lfsr_c[0]~4	4				.	Trigger	

信号の Storage Enable が有効になっていない場合、信号は Storage Qualifier 条件の一部になることはできません。

Continuous Mode

													c	ick to i	nsert tir	ne bar											
	No	de	0																								
Туре	Alias	Name	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
		⊞- data_out[05]	(01h	X 02	h (03h	X 04h	X 05ł	1 (06ł	h X 07	h (08ł	1 X 09ł	XOA	h (OB	h X OC	h (OD)	h X OEH) (OFF	X 2DH	n X 2EP	1 (01	h (02	n X 03	1 (04	h (05	h X 06	h (07	h)
0	Storage Qualifier 1	data_out[6]																									-
	Storage Qualifier 2	data_out[7]																									_





Conditional Mode

※ Storage Qualifier 条件は、data_out[6] AND data_out[7] を評価するように設定した場合

												cl	ck to i	sert tin	e bar											
	No	de	1							2								3								4
Туре	Alias	Name	0	1 :	2 3	4	4 5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0		⊕ data_out[05]	07h	(08h)	(09h)	0Ah	(OBh)	0Ch X0	Dh 🗙 OEh	(07h	Xoa	h X 09t	XOA	h X OBH	Xoci	XOD	h X OEH	X 07h	X 08	h X 09ł	1 XOAH	XOB	h X OC	n X Ot	h X OE	ĩ
0	Storage Qualifier 1	data_out[6]																								
0	Storage Qualifier 2	data_out[7]																								

◆ Start/Stop モード

Start/Stop 2 つの条件を設定し、成立時にデータをストアします。なお、Stop 条件は Start 条件よりも優先されます。

trigger: 201	7/04/11 15:21:05 #1	Lock mode:	Allow	all changes		/		Signal Configuration:	2
	Node	Data Enable	igger Enab	Storage Enable	Stor	age Qualifier	Trigger Conditions		
Type Allas	Name	4	5	4	Start Basic AND	🗘 Stop Basic AND 🖨	1 Basic OR 🖨	Pipeline Factor: 1	
R	_ss[1]~DUPLICATE	3	4	1	100	1	88	Storage qualifier:	
R	ss[3]~DUPLICATE	-	*	2	0	355	202	and the second sec	
<u>L</u>		•	1	~	x	1	题	Type: 😤 Start/Stop	
R.	s[10]-DUPLICATE		4				122	Internet in the second s	10.11.4.53
5		•	*	4	1		122	input port anna_ana_external_storage_	
								Record data discontinuities Disable storage qualifier	4
								Trigger	

Storage Qualifier Enable

Continuous Mode

													clic	k to in	sertt	ine ba	e :								
		Node	0																						
Туре	Alias	Name	-2	-1	0	1	2 3		1	5 6	6	7	8	9	10	11	12	13	14	15	16	17	18	19	2
1		⊞- data_out[05]		DLE	(01h	(02h	(03h)	(04h)	(05h)	(06h)	(07h	(08h	(09h	XOAH	XOE	h) (D	h)(0	Dh X	Eh XO	Fh X			E	ALE.	_
0	STOP	data_out[6]													٦.										
	START	data_out[7]				1												_							
_																									

Start/Stop Storage Qualifier:

										sick	to inse	rtim	a loar										
		Node	0	1	2								3	1	4								
Type	Alias	Name	-3	7 -1	0 1	2 3	4	5 6	7	8	9		0 1	1	12	13	14	15	16	17	18	19	20
0		[e. data_out]05]	(OA)	X ODA X OEh	01h X 02	h (03h)	(04h) (05h	X 05h X	07h)	OBh X	09h)	(IAh	X ODn	(DEh	XOIN	02	h)(03	h X04	h (05	h X CE	h Xut	h X0	Ch X Of
0	STOP	data_out(6)				1001 1010		1				-	1				-	1111	-	1000	1.000	1.000	
0	START	data_out[7]										_		_									





4-3. トリガの定義

モニタする信号のトリガ条件を指定します。

Signal Configuration ペインの Trigger Conditions でトリガ条件を複数指定できます。最大10レベルのトリガ・コンディションを設定可能です。



4-3-1. トリガ・コンディションの設定

必要なデータをキャプチャするため、モニタする信号がそのデータを表示する条件(トリガ)を指定します。

Setup タブの Node List 内、Trigger Conditions 項目のプルダウン・リストにより、トリガ・コンディション・タイプ を選択します。タイプは4つあります。

	Trigg	er Conditions		
1 🔽 Basic AND 🛛 🔻	2 🔽	Basic ANE 🔻	3 🗸	Basic ANE 🔻
Xh		Basic AND		Xh
		Basic OR Comparison		
		Advanced		
58		55		58

Basic AND / Basic OR	トリガに適用したい各信号にトリガ・パターンを指定
Comparison	バス信号上で簡単な比較条件を指定し、バスの複数のグループ化ビットを
	予想される整数値と比較
Advanced	専用エディタに GUI で演算子などを描き複雑なトリガ式(トリガ条件)を作成

◆ Basic AND / Basic OR トリガ

最もシンプルな種類のトリガで、追加した各信号のトリガ・パターンを設定する必要があります。

トリガ・パターンを設定するには、Trigger Levels カラムで右クリックし、希望のパターンをクリックします。トリ ガ・パターンは以下のいずれかに設定できます。

- Don't Care ••• 未定義(デフォルト)
- Falling Edge ・・・ 立ち下りエッジ

• Low

- Rising Edge ・・・ 立ち上がりエッジ
- High Eith
 - Either Edge ••• 立ち上がり または 立ち下りエッジ



ESENA

		Node	Data Enable	Trigger Enable	Trigger Con	ditions	
Туре	Alias	Name	10	10	1 🔽 Basic Al	VD 🔻	
5		📮unter:Count_Ones cntr_a7i:auto_generated counter_reg_bit[30]	v	V	Yh		
*		unter:Count_Ones cntr_a7i:auto_generated counter_reg_bit[3]	V	V)	
*		unter:Count_Ones cntr_a7i:auto_generated counter_reg_bit[2]	v	V			AND/ OK
*		unter:Count_Ones cntr_a7i:auto_generated counter_reg_bit[1]					AND
*			パターン	が指定する信号	を 🔛		OR
-		📮unter:Count_Tens cntr_a7i:auto_generated counter_reg_bit[30]	選択	して右クリック	Xh		NAND
*		ounter:Count_Tens cntr_a7i:auto_generated counter_reg_bit[3]					NOR
*		ounter:Count_Tens cntr_a7i:auto_generated counter_reg_bit[2]	v	✓			XOR
*		ounter:Count_Tens cntr_a7i:auto_generated counter_reg_bit[1]	V	V			XUOR
*		ounter:Count_Tens cntr_a7i:auto_generated counter_reg_bit[0]	v	v			ANOR
		svnseg:Decode_Ones a~13	V	V			TRUE
<u> </u>		svnseg:Decode_Tens a~13	V	\checkmark			FALSE
							Compare
						<u></u>	Don't Care
						<u> </u>	Low
						\mathcal{L}^{-}	Falling Edge
						1	Rising Edge
						1	High
						x	Either Edge
							-
	_						Insert Value
涛 Data		Setup					

バス信号の場合は、グループ表示されたバス信号の Trigger Levels カラムを右クリック選択し、Insert Value から値をまとめて入力することが可能です。

🟸 Insert	t Value				X	
Value:	A	F	Radix: Hexadec	imal	-	
			ОК	Cancel	Help	
		Node	Data Enable	Trigger Enable	Trigger Conditions	
Туре	Alias	Name	8	6	1 🗹 Basic AND 🔻	
- \		₽ed counter_reg_bit[30]	1	V	Ah	
*		ted counter_reg_bit[3]	1	V	1	
*		ted counter_reg_bit[2]	1	V	0	Ν
*		ted counter_reg_bit[1]	1	V	ダブルクリックし	てパターンを直接入力
*		ted counter_reg_bit[0]	v	V	することも可能。	
a		₽ed counter_reg_bit[30]	1		バス信号の Radi	x 変更は、Node 欄の
*		ted counter_reg_bit[3]	1		バス信号名部分	を右クリックで選択し、
*		ted counter_reg_bit[2]	1		Bus Display Forma	at ► フォーマット・タイ
*		ted counter_reg_bit[1]	v		ノを进択。	
*		ted counter_reg_bit[0]	v			
<u> </u>		svnseg:Decode_Ones a~13		V]
- -		svnseg:Decode_Tens a~13		V]

選出した信号をトリガとして使用せずにデータの観測のみの場合には、Trigger Enable オプションを OFF します。同様に、選出した信号をトリガとしてのみ使用し、データの観測は必要がない場合には、Data Enable オプションは OFF にします。これにより、Signal Tap IP を構成するために必要なリソースを節約することが可能です。

EISE

◆ Comparison トリガ

バス信号上で簡単な比較条件を指定することによって、バスの複数のグループ化ビットを予想される整数 値と比較できます。Comparison トリガは Basic OR トリガに含まれるすべてのトリガ条件を保持します。

	Node			Trigger Enable	Trigger Con	ditions]
Туре	Alias	Name	14	14	1 Comparise	on 🔻	
a		t_Tens cntr_a7i:auto_generated counter_reg_bit[30]	V	V	Basic AND)	
- S		svnseg:Decode_Ones a~13	V	V	Comparise	n	ID / OR
- S		svnseg:Decode_Tens a~13	V	V	Advanced		ID
		⊡t_Ones cntr_a7i:auto_generated counter_reg_bit[30]	V	\checkmark	Xh	0	R
<u></u>		ount_Ones cntr_a7i:auto_generated counter_comb_bita0	V	\checkmark		N	AND
<u></u>		ount_Ones cntr_a7i:auto_generated counter_comb_bita1	V	\checkmark		N	OR
<u></u>		ount_Ones cntr_a7i:auto_generated counter_comb_bita2	V	\checkmark			
<u></u>		ount_Ones cntr_a7i:auto_generated counter_comb_bita3	V	\checkmark		X	JK
						TR FA	(UE ALSE
						Co	ompare
						🗱 Da	on't Care
						<u> </u>	w
						∖ Fa	alling Edge
						🦵 Ri	sing Edge
						1 Hi	iah
						X Eif	ther Edge
						In	sert Value

Comparison トリガは以下 2 つの比較タイプをサポートしています。

Single-value comparison	Interval check
M Compare	Compare
Node: Ipm counter:Count Tens cntr a7i:auto generated counter reg_bit[30] Comparison type: Single-value comparison	Node: lpm_counter.Count_Tens cntr_a7i:auto_generated counter_reg_bit[3.0] Comparison type: Interval check •
Single-value comparison	Single-value comparison
Operand: Value: 0	バス信号の値が定義した間隔に 限定されているかどうかを検証
バス信号の値と指定した数値を比較	Interval check
Lower bound value: 0	Lower bound value: 0
Exclusive	 Exclusive
Inclusive	Inclusive
Upper bound value: 15	Upper bound value: 15
Exclusive	Exclusive
Inclusive	Inclusive
Ok Cancel	Ok Cancel

◆ Advanced トリガ

トリガ・コンディション・タイプを Advanced に切り替えると、Advanced Trigger タブが追加され専用エディタが 起動します。このエディタに Object や信号を用いて演算式を作成し、その演算結果がトリガ条件として使用さ れます。



Advanced トリガの詳細は、Advanced Trigger Conditions をご覧ください。

4-3-2.トリガ・フロー・コントロールの設定

トリガ・フロー・コントロールを設定し、作成したトリガ条件を正確に制御することができます。

Signal Configuration ペイン内 Trigger 項目の Trigger flow control プルダウン・リストにより選択します。タイプは以下の2つです。

- Sequential (デフォルト)
- State-Based

Sequential トリガは、取得バッファがキャプチャを終了する前に満たす必要のある最大トリガレベルを定義できます。

State-Based トリガについては、Trigger Condition Flow Control をご覧ください。

4-3-3.トリガ位置の指定

トリガ・イベントの前後に取得されるデータの量を指定します。

Pre trigger position	トリガの後に発生した信号の状態を保存します。(トリガ前 12%、トリガ後
	88%)
Center trigger position	トリガ前とトリガ後のデータを半分ずつ保存します。(トリガ前 50%、トリガ
	後 50%)
Post trigger position	トリガの前に発生した信号の状態を保存します。(トリガ前 88%、トリガ後
	12%)







4-4. デザインのコンパイル

STP ファイルの Processing メニュー ➤ Start Compilation または Start Rapid Recompile(該当する FPGA の み実行可能)を選択し、Signal Tap ロジック・アナライザ IP を組み込んだユーザ・デザインをコンパイルします。



Note: Signal Tap Logic Analyzer オプション (Assignments メニュー ► Settings) に設定された STP ファイルがユーザ・デザインに インスタンスされます。

General	Signal Tap Logic Analyzer
Files	Specify compilation options for the Signal Tap Logic Analyzer.
Libraries	
IP Settings	Enable Signal Tap Logic Analyzer
Design Templates	Signal Tap File name: stille name> sto
Operating Settings and Conditions	ongran rap the name one name step
$\sigma^{\sigma}=\cdots=\sigma^{11}=10^{1}, \sigma=\sigma^{\sigma}=\sigma^{-1}, \sigma=\sigma^{-1}, \sigma=\sigma^{-1},$	
Assembler	
Design Assistant	
Signal Tap Logic Analyzer	
Logic Analyzer Interface	

Note: コンパイル完了後に STP ファイルの仕様変更を行った場合、それが Signal Tap IP に必要なリソースへ影響する変更であ るときのみ再コンパイルが必要です。STP ファイルの Instance Manager ペインにもメッセージが表示されます。

🖷 📒 つ ぐ 斗 🏙 🕨	r ()			
Instance Manager: 🍡 💫 🔳	Start Rapid Re	ecompile to l	Reroute the design	
Instance	Status	Enabled	LEs: 679	Memor
🔝 auto_signaltap_0	Not running	\checkmark	679 cells	53248



4-5. FPGA プログラミング

デバッグを行う FPGA へ、コンパイルにより生成された SOF ファイルをダウンロードします。ダウンロードの 操作は、STP ファイルで行います。

- パソコンとダウンロード・ケーブル、ダウンロード・ケーブルと FPGA の搭載されたボードを接続し、ボードの電源を投入します。
- STP ファイルの JTAG Chain Configuration ペインの Hardware プルダウン・リスト、または Setup ボタンを クリックし、使用するダウンロード・ケーブルを選択します。
- ③ Scan Chain ボタンをクリック後、Device プルダウン・リストからターゲットの FPGA を選択します。
- ④ SOF Manager 右端のブラウズ・ボタンをクリックし、使用する STP ファイルを選択します。
- ⑤ SOF Manager 中央にある Program Device ボタンをクリックし、書き込みを実行します。

JTAG Chain Configuration: JTAG ready	×
Hardware: USB-Blaster [USB-0]	▼ Setup
Device: @1: 10M08SA(.JES)/10M08SC (0x031820DD)	▼ Scan Chain
>> SOF Manager	.sof

⑥ Messages ウィンドウに書き込みが完了したことを表すメッセージが確認できます。



4-6. Signal Tap の実行

FPGA に SOFファイルが転送されると、すぐにSignal Tap IP を含んだユーザ・ロジックが動作し始め、STP に設定したトリガ条件に応じて、FPGA の内部メモリに指定した信号のデータが保存されています。

STP ファイルの Run Analysis ボタンをクリックし、メモリに格納されたデータを波形ウィンドウに表示させます。

Run Analysis ボタン	で 緑 😁 🕨	Stop Analysis ボタン
Instance Manag	er. 🔁 🔊 🔳	Ready to acquire
Instance		Status Enal
		Autorun Analysis ボタン



ES



Туре	Alias	Name	-8	{	5 ₋ -	4		2	, q		2		4	, 6		8
.		📮r_a7i:auto_generated counter_reg_bit[30]			7	h					\square					
*•		tr_a7i:auto_generated counter_reg_bit[3]														
*•		tr_a7i:auto_generated counter_reg_bit[2]														
*		tr_a7i:auto_generated counter_reg_bit[1]														
*•		tr_a7i:auto_generated counter_reg_bit[0]														
*•		svnseg:Decode_Ones a~13														
*•		svnseg:Decode_Tens a~13														
يل		📮r_a7i:auto_generated counter_reg_bit[30]			8h					<u>9h</u>	\square					
*•		tr_a7i:auto_generated counter_reg_bit[3]														
*•		tr_a7i:auto_generated counter_reg_bit[2]														
*•		tr_a7i:auto_generated counter_reg_bit[1]														
*•		tr_a7i:auto_generated counter_reg_bit[0]														
*•		cntr_a7i:auto_generated counter_comb_bita0														
*•		cntr_a7i:auto_generated counter_comb_bita1								•	Լ					
*•		cntr_a7i:auto_generated counter_comb_bita2									1	リガ	ポシ	ション	· _	
*•		cntr_a7i:auto_generated counter_comb_bita3														
							_									
			•													
🥦 Data	- 20	Setup														

4-7. キャプチャされたデータの表示・解析

検出されたデータを解析する際に便利な機能がいくつかありますので、代表的なものを以下にご紹介します。

4-7-1. タイム・バー

波形表示ウィンドウに、タイム・バーを挿入することができます。

波形ウィンドウ上部(cick to insert time bar と表示された部分)の挿入したい場所をマウスでクリックします。



タイム・バーは 🔙 耐 ボタンで左右へ移動可能です。マウスでタイム・バーをドラッグ&ドロップしても移動で きます。



4-7-2. 検出データの変換

検出したデータをリスト・ファイルにすることができます。また、別のファイル・フォーマットに変換することも可 能です。

<u>リスト・ファイルの生成</u>

- STP ファイル上の File メニュー ➤ Create Signal Tap List File を選択します。
- プロジェクト・フォルダに "<STP ファイル名>_<Signal Tap インスタンス名>.txt" が生成され、Quartus Prime のメイン画面にリスト・ファイルが表示されます。



別のファイル・フォーマットに変換

- ① STP ファイル上の File メニュー ➤ Export を選択します。
- ② Export ウィンドウが表示されます。
 - ・ File name: 保存するファイル名を入力し、ブラウズ・ボタンで保存フォルダを指定します。
 - Export format: プルダウン・リストから変換するフォーマットを選択します。
 - Comma Separated Values (.csv)
 - Table File (.tbl)
 - Value Change Dump (.vcd)
 - BMP Image File (.bmp)
 - JPEG Image File (.jpg)
 - Clock signal: 使用したクロック信号名が表示 されます。(自動)
 - ・ Clock period: クロック周期を入力します。
 - ③ 指定したファイルが生成されます。

🟸 Export		×
Specify settings	for exporting Signal Tap waveform data	
File name:	stp_debug	
Export format:	Comma Separated Values - (*.csv)	-
Clock signal:	clock~inputclkctrl	
Clock period:	1 s	-
	OK Cancel Help	



4-7-3. データ・ログの保存

STP ファイルで使用したトリガ設定や、指定したトリガ条件で検出した波形データをログとして保存することができます。

- ① STP ファイルの Data log ペイン内にある Data Log オプションを有効にします。
- 右のボタン(Save to Data Log)をクリックすると、その時点で STP ファイルに表示されている各種情報が保存されます。
 - ・ signal_set: Signal Configuration ペインに登録されたログ
 - ・ trigger:トリガ設定のログ
 - ・ log: キャプチャされた波形のログ

これらはすべて、保存した日時が末尾に記録されます。

🔽 Data Log: 🖳				
auto_signaltap_0				
signal_set: 2018/01/11 12:14:00 #0				
Itrigger: 2018/01/11 12:14:00 #1				
🐵 log: Trig @ 2018/01/11 12:26:39 (0:0:0.8 elapsed)				
🖸 log: Trig @ 2018/01/1	Delete Item	Dal]	
妃 log: Trig @ 2018/01/1	Delete item	Dei		
 ▲ 100 trigger: 2018/01/11 12:28: Iog: Trig @ 2018/01/1 	Rename Item	F2		
	Select All Items	ログ名をす	这更可能	

複数ログを保存した場合、表示させたいログをダブルクリックで選択してください。表示されているログのア イコンが枠で囲われます。

Data Log オプションを有効にした後に Signal Tap を実行(Run Analysis ボタンをクリック)した場合は、その 度にキャプチャされた波形のログが追加記録されます。



改版履歴

Revision	年月	概要
1	2018年1月	初版

免責およびご利用上の注意

弊社より資料を入手されましたお客様におかれましては、下記の使用上の注意を一読いただいた上でご使用ください。

- 1. 本資料は非売品です。許可無く転売することや無断複製することを禁じます。
- 2. 本資料は予告なく変更することがあります。
- 本資料の作成には万全を期していますが、万一ご不明な点や誤り、記載漏れなどお気づきの点がありましたら、本資料を入手されました下記代理店までご一報いただければ幸いです。
 株式会社マクニカ アルティマ カンパニー https://www.alt.macnica.co.ip/ 技術情報サイト アルティマ技術データベース http://www.alt.macnica.co.ip/ 技術情報サイト アルティマ技術データベース http://www.alt.macnica.co.ip/ 技術情報サイト ETS https://www.alt.macnica.co.ip/elspear/members/
- 4. 本資料で取り扱っている回路、技術、プログラムに関して運用した結果の影響については、責任を負いかねますのであらかじめご了承ください。
- 5. 本資料は製品を利用する際の補助的な資料です。製品をご使用になる際は、各メーカ発行の英語版の資料もあわせてご利用ください。