

ALTERA®



SOPC
WORLD
2 0 0 2



SOPC
WORLD
2 0 0 2

SignalTap II



SignalTap[®] II

ALTERA[®]

Agenda

- SignalTap II Interface in Quartus II S/W
- SignalTap II Demonstration
 - Initial Compilation
 - Second Compilation:
Using SignalTap II Logic Analyzer
- Conclusions

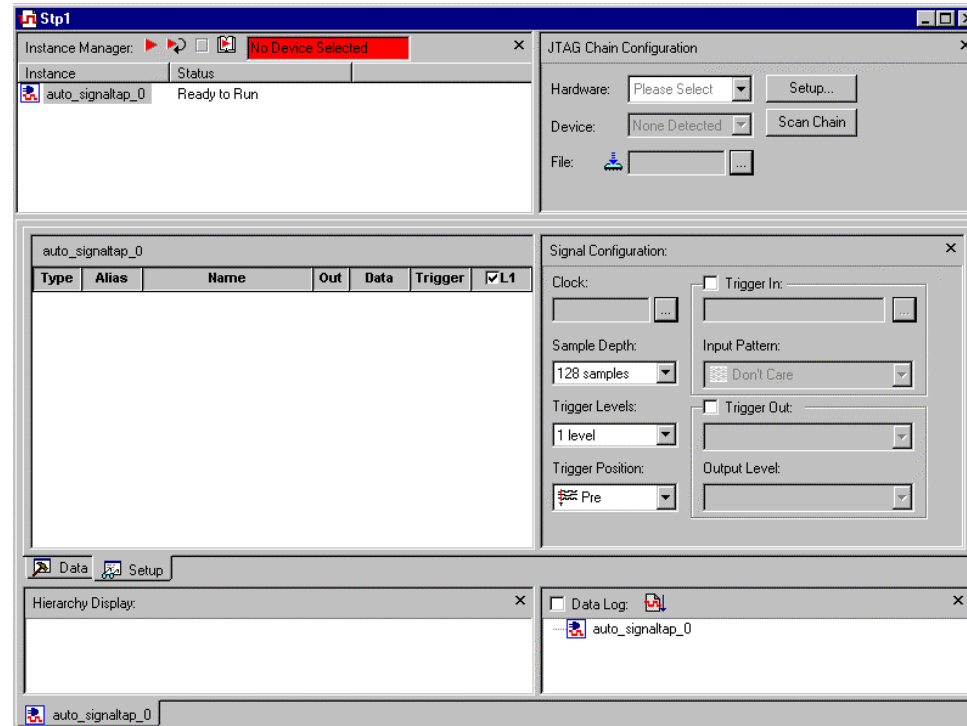


SOPC
WORLD
2 0 0 2

SignalTap II Interface in Quartus II Software

SignalTap II File (.stp)

- Creating SignalTap[®] II Logic Analyzer File
 - Choose New (File Menu)
 - Click the Other Files Tab & Select SignalTap II File
 - Click OK



SignalTap II File

Instance Manager

The screenshot displays the SignalTap II File application window, titled "Stp1". The interface is divided into several panels:

- Instance Manager:** Located at the top left, it shows a table with columns "Instance" and "Status". One instance, "auto_sigtaltap_0", is listed with a status of "Ready to Run". A red banner at the top of this panel reads "No Device Selected".
- JTAG Chain Configuration:** Located at the top right, it contains fields for "Hardware" (set to "Please Select"), "Device" (set to "None Detected"), and "File". Buttons for "Setup...", "Scan Chain", and a file selection icon are also present.
- Signal Configuration:** Located in the middle right, it includes settings for "Clock", "Sample Depth" (set to "128 samples"), "Trigger Levels" (set to "1 level"), and "Trigger Position" (set to "Pre"). It also has sections for "Trigger In" and "Trigger Out" with associated input/output patterns and levels.
- Signal Viewer:** Located in the middle left, it displays a table with columns "Type", "Alias", "Name", "Out", "Data", "Trigger", and "L1". The table is currently empty.
- Hierarchy Display:** Located at the bottom left, it shows a tree view of the design hierarchy.
- Data Log:** Located at the bottom right, it shows a list of data logs, including "auto_sigtaltap_0".

Red arrows point from the external labels to their respective panels in the interface.

JTAG Chain Configuratio

Signal Viewer

Signal Configuratio

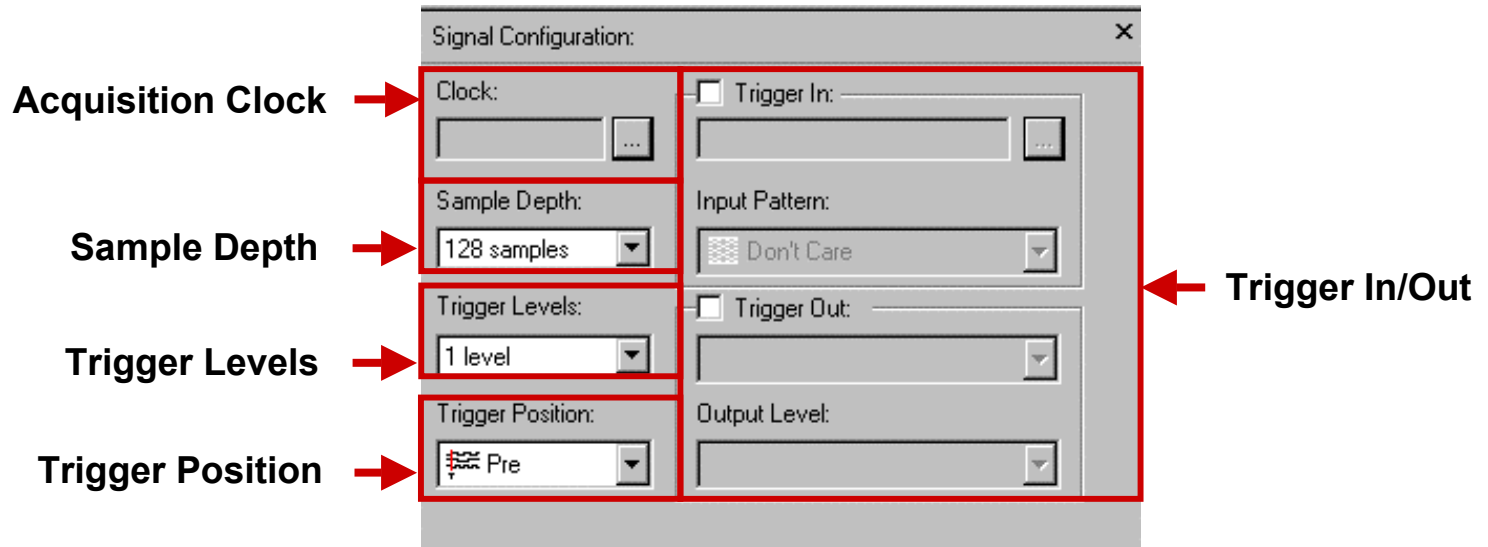
Hierarchy Display

Data Log

SignalTap II File: Acquisition Clock

■ Acquisition Clock

- For Best Results, Assign Only Global Clock as the SignalTap II Clock Signal
- Without Assigning Clock Signal, Quartus II Software Creates Clock Pin `auto_stp_external_clock`



SignalTap II File: Trigger Pattern

■ Sample Depth

- Set Number of Samples Stored for Each Input Signal
 - 0 to 128K Sample Depth

■ Trigger Levels

- Configure Analyzer with up to 10 Trigger Levels

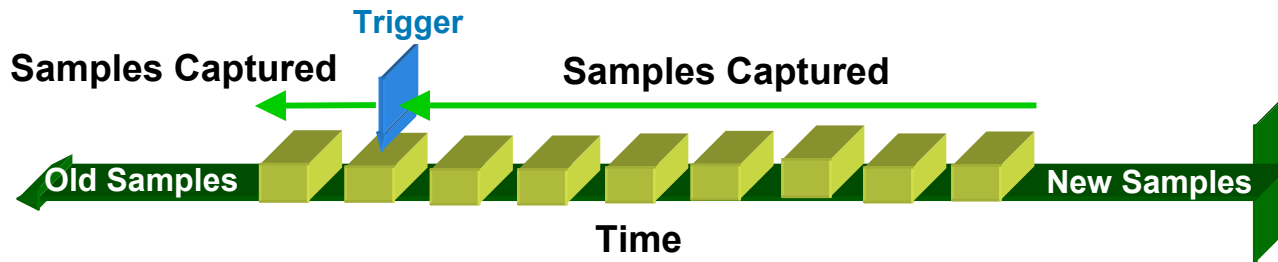
■ Trigger Position

- Specify Amount of Data Captured by SignalTap II Logic Analyzer that Should be Acquired before the Trigger as well as Amount that Should be Acquired after the Trigger

Trigger Position

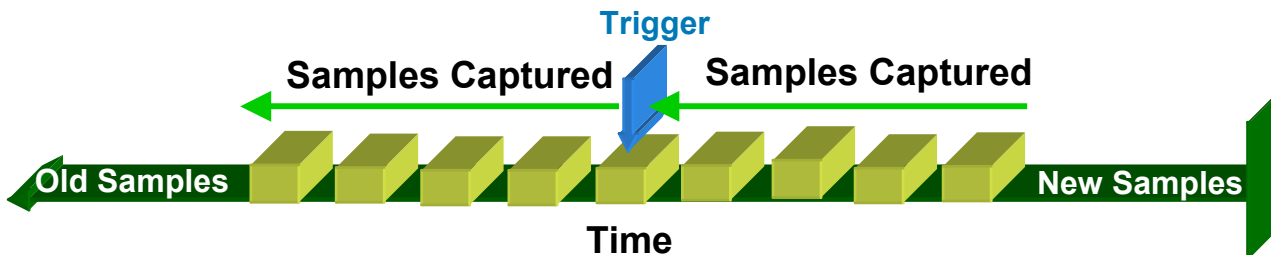
■ Pre-Trigger

- Captures Signals that Occur Immediately after Triggering (12% Pre-Trigger, 88% Post-Trigger)



■ Center Trigger

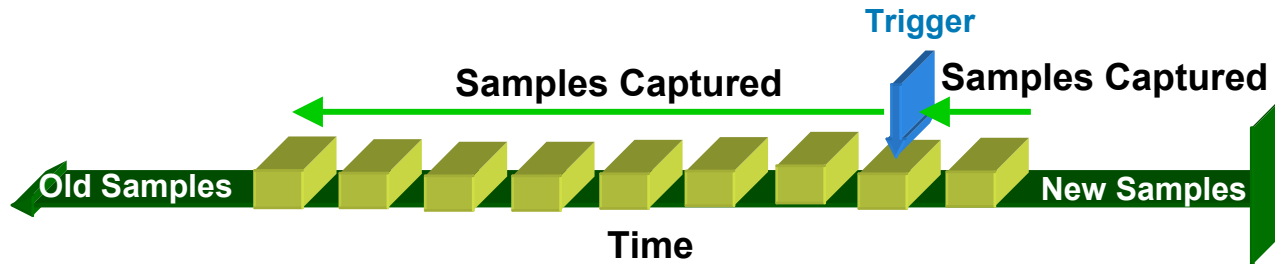
- Captures Signals before & after Triggering (50% Pre-Trigger, 50% Post-Trigger)



Trigger Position

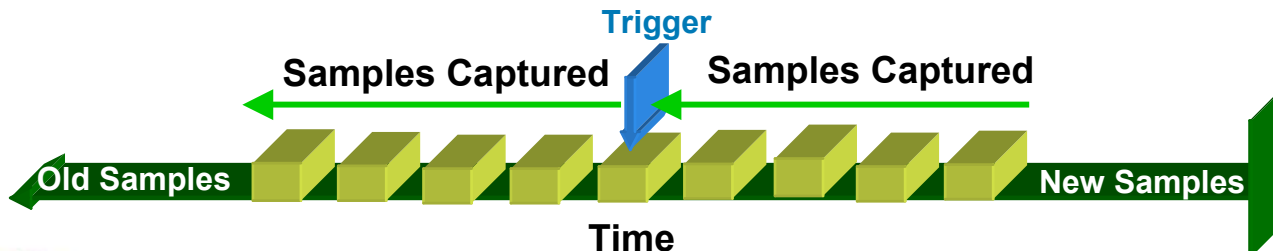
■ Post-Trigger

- Captures Signal that Occur Immediately before Triggering (88% Pre-Trigger, 12% Post-Trigger)



■ Continuous Trigger

- Captures Signals Indefinitely Until Stopped Manually (Useful When Using the Trigger Out Feature)



SignalTap II File:Trigger In/Out

■ Trigger In












- Any I/O Pin Can Trigger the SignalTap II Analyzer
- Pin auto_stp_trigger_in_0 Generated in Device
- Trigger Input Can be Set to Recognize High, Low, Rising Edge, Falling Edge, Either Edge, or Don't Care Condition


■ Trigger Out

- Spare I/O Pin that Is Set as Trigger Output Signal That Indicates When Trigger Pattern Occurs
- Pin auto_stp_trigger_out_0 Generated in Device
- Output Pulse Polarity Is Programmable

SignalTap II File: Debug Ports

- Routing SignalTap II Signal to Spare I/O Pin for Capture by Logic Analyzer
- Quartus II Software Automatically Generates a Pin
 - Debug Port Pin Name Is **stp_debug_out_1_n**
 - n Is a Number Representing the Order in Which the Debug Port Pin Occurs in the Signal List

Type	Alias	Name	Out	Data	Trigger	✓ L1
		CNT_ONE_ENABLE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
		CNT_ONE0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—
		CNT_ONE1				—
		CNT_ONE2				—
		CNT_ONE3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—

 Debug Port On
Debug Port Off

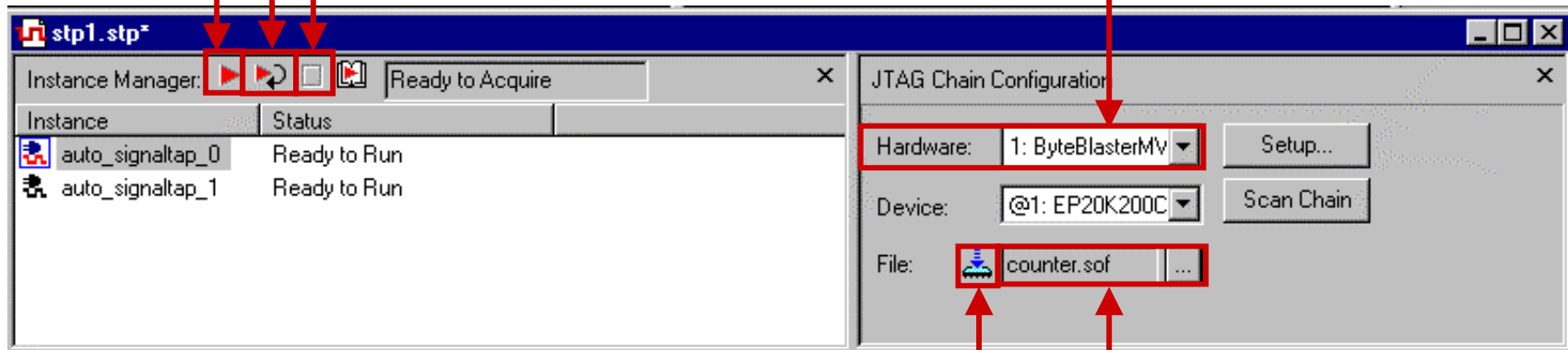
SignalTap II File: Control Settings

Stop the Logic Analyzer

Run Continuous

Run the Embedded
Logic Analyzer


Specify Communication
Hardware

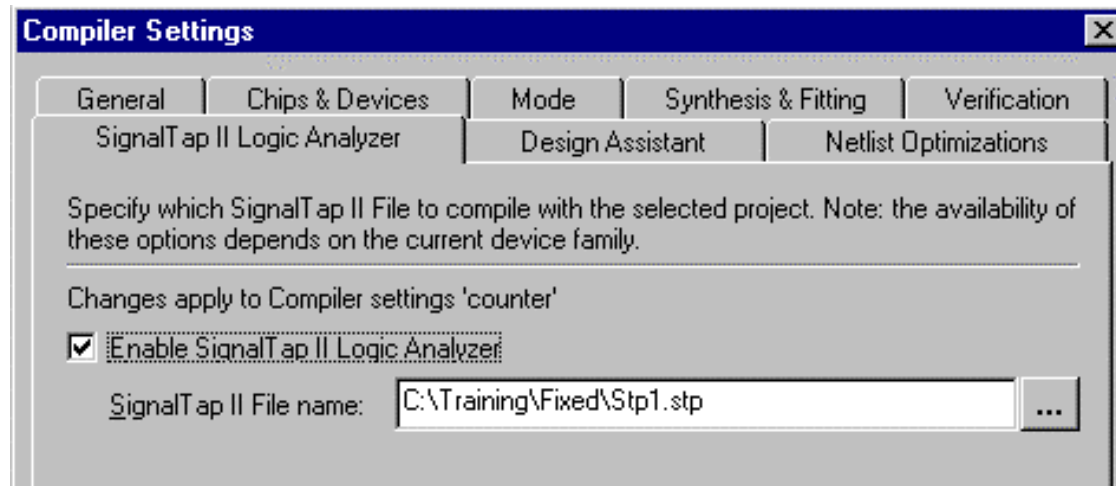


Specify Configuration
File (.sof or .cdf)

Download SOF file

Specify SignalTap II File

- Choose Compiler Settings (Processing Menu)
- Click the SignalTap II Logic Analyzer Tab
- Turn on Enable SignalTap II Logic Analyzer
- In the SignalTap II File Name box:
 - Type the Name of the SignalTap II File (.stp) for Compilation
 - Or Select a File Name with Browse Button 



Project Compilation

- Design Recompile Required When Any of These Parameters Change
 - Acquisition Clock
 - Number of Channels
 - Sample Depth
 - Debug Ports
 - Trigger in/out Ports
- Design Recompile Not Required When
 - Changing Trigger Pattern or Position
 - Modifying Trigger Levels
 - Starting or Stopping SignalTap II Logic Analyzer
- Internal Nodes to Be Analyzed Should be Inputs, Outputs, Register Outputs, or Memory Outputs

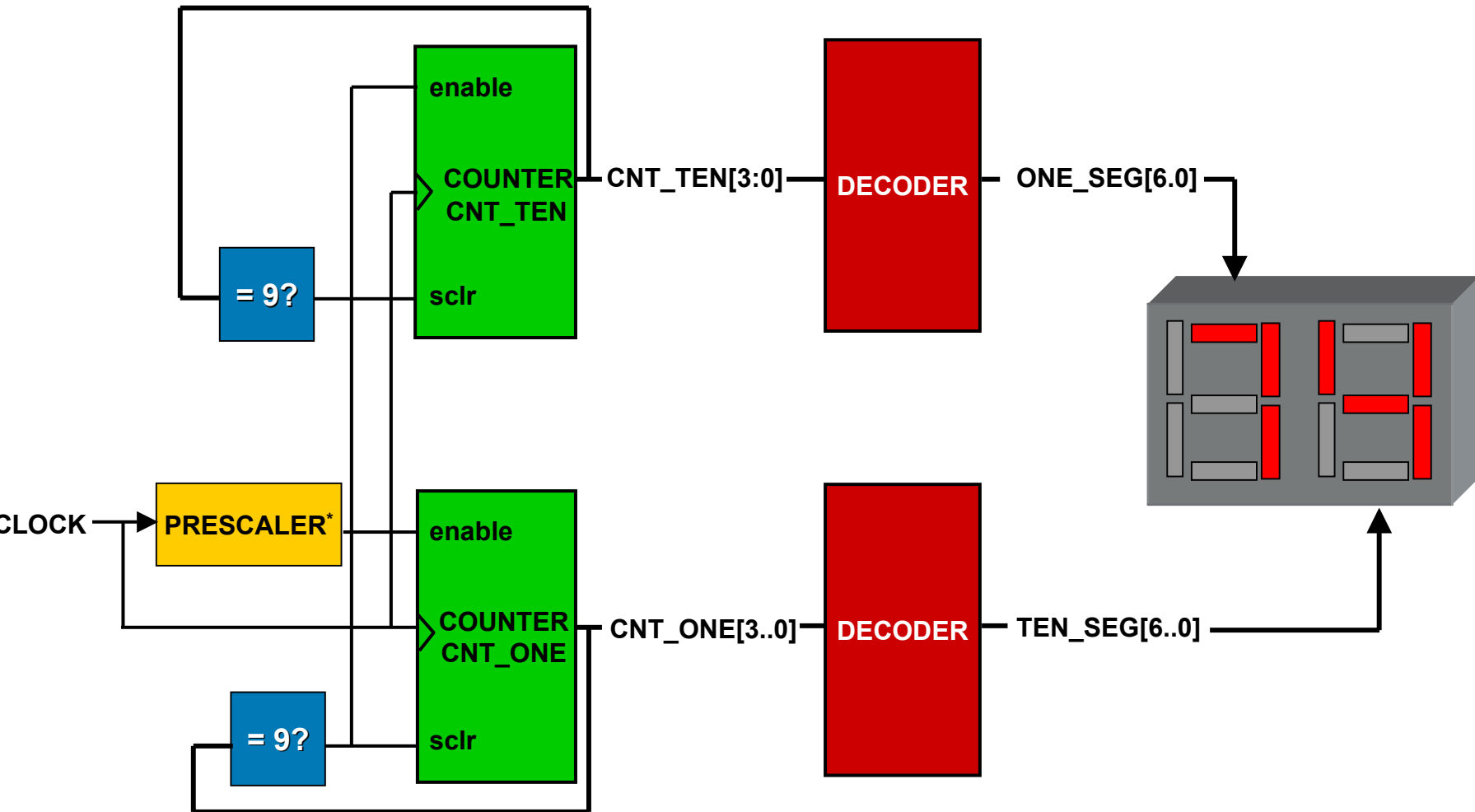
SignalTap II Demonstration

- Description of the Test Case
- Initial Compilation
 - Observing the Malfunction
- Second Compilation
 - Creating the Embedded Logic Analyzer
 - Debugging Design
- Third Compilation
 - Fixing the Design Issue

Design: Counter from 00 to 99

- Function: Two 4-Bit Counters Cascaded to Drive a Pair of Seven-Segment LEDs that Count from 0 to 99
- Top Level Name: Counter
- Target Device : EP20K200EFC484-2X
- Tools
 - Synthesis: LeonardoSpectrum
 - Fitter: Quartus II Version. 2.1
- Place Source Design Files in **C:\Training\Synthesis** Directory
- Place pins.tcl File in **C:\Training** Directory

Counter Block Diagram



* : The Prescaler Generates a Counter Enable with a Frequency of about 1 Hz

Synthesize Design (1/2)

- Launch LeonardoSpectrum Tool

- Select Mode Quick Setup



- Working Directory: C:\Training\Synthesis
- Open Files: counter.vhd
- Technology: APEX 20KE
- Device: EP20K200EFC484
- Speed Grade: -2X
- Output File: C:\Training\Synthesis\counter.edf

- Click on Run Flow

Synthesize Design (2/2)

Quick Setup →

Technology →

Device →

Speed Grade →

Output File →

Source File →

Working Directory →

Quick Setup

Run the entire flow from this one condensed page. Specify your source file(s), technology and desired frequency, then press Run Flow.

Technology

- Altera
 - ACEX 1K
 - APEX 20K
 - APEX 20KC
 - APEX 20KE
 - APEX II
 - Excalibur Arm
 - FLEX 10K
 - FLEX 10KA
 - FLEX 10KB
 - FLEX 10KE
 - FLEX 6K

Input

counter.vhd

Device:

EP20K200EFC484

Speed Grade:

-2X

Open files:

Working Directory:

Constraints

Clock Frequency

Mhz

Optimize Effort

Fastest Runtime

High Effort

Output

Output File: C:\Training\Synthesis\counter.edf

Place And Route

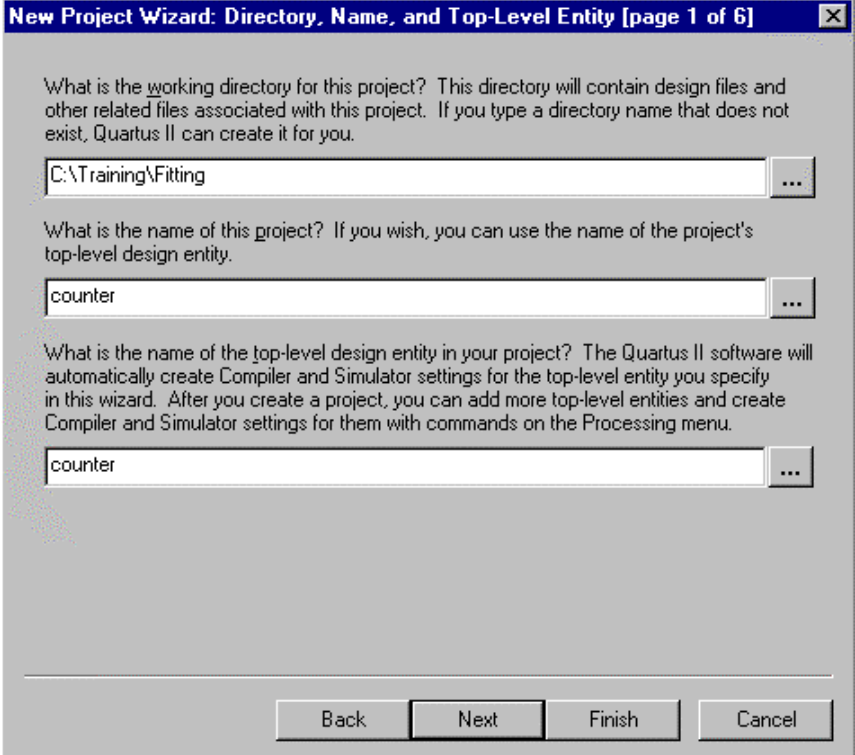
☐ Run Integrated Place and Route

Run Flow

Help

Create Quartus II Project

- Launch the Quartus II Software
- Choose New Project Wizard (File menu) to Create a New Quartus II Project
 - Directory Name: **C:\Training\Fitting**
 - Project Name: **counter**
 - Top Level Name: **counter**
- Click Next



The screenshot shows the 'New Project Wizard: Directory, Name, and Top-Level Entity [page 1 of 6]' dialog box. It contains three text input fields with '...' buttons to the right. The first field is for the working directory, containing 'C:\Training\Fitting'. The second field is for the project name, containing 'counter'. The third field is for the top-level design entity name, also containing 'counter'. Below the fields are four buttons: 'Back', 'Next', 'Finish', and 'Cancel'.

New Project Wizard: Directory, Name, and Top-Level Entity [page 1 of 6]

What is the working directory for this project? This directory will contain design files and other related files associated with this project. If you type a directory name that does not exist, Quartus II can create it for you.

C:\Training\Fitting

What is the name of this project? If you wish, you can use the name of the project's top-level design entity.

counter

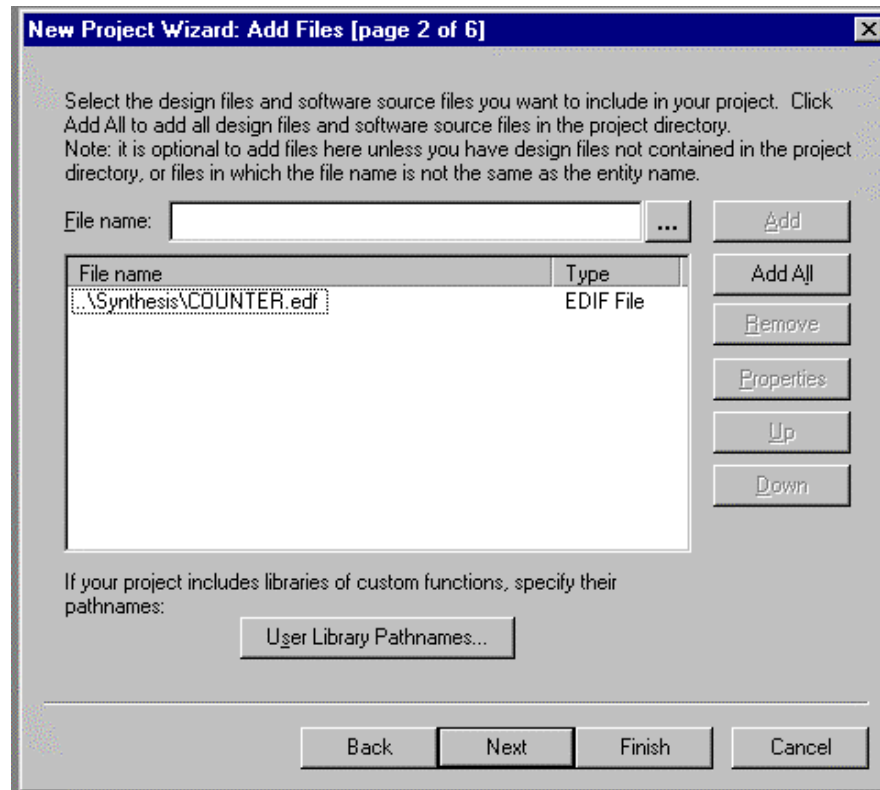
What is the name of the top-level design entity in your project? The Quartus II software will automatically create Compiler and Simulator settings for the top-level entity you specify in this wizard. After you create a project, you can add more top-level entities and create Compiler and Simulator settings for them with commands on the Processing menu.

counter

Back Next Finish Cancel

Create Quartus II Project

- Add Following Files in Project
 - C:\Training\Synthesis\counter.edf
- Click Next
- Click Finish



Create Project: Set EDA Tool

- Select LeonardoSpectrum Tool (Level 1) as Design Entry/Synthesis Tool
- Click Next

Specify the other EDA tools -- in addition to the Quartus II software -- that you will use on this project.

EDA tools

Tool type	Tool name
Design entry/synthesis	Leonardo Spectrum(Level 1)
Simulation	<NONE>
Timing analysis	<NONE>
Board-level	<NONE>
Formal verification	<NONE>
Resynthesis	<NONE>

Tool settings

Tool type: Design entry/synthesis

Tool name: Leonardo Spectrum(Level 1)

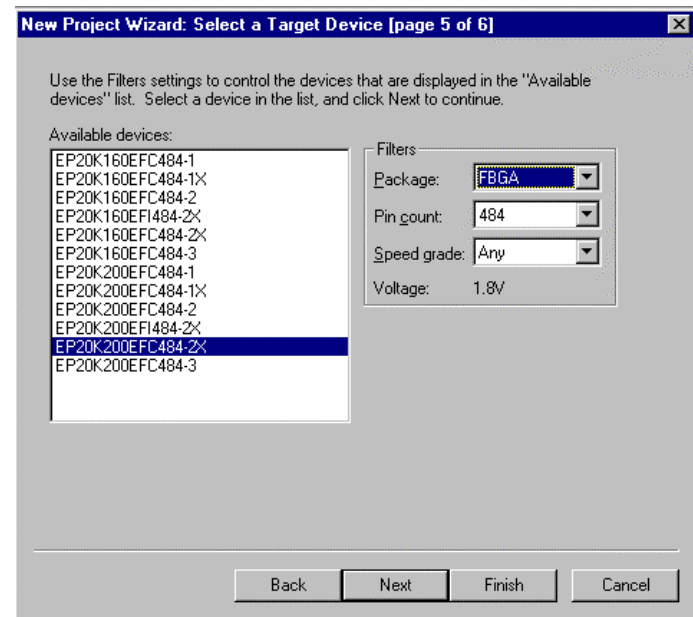
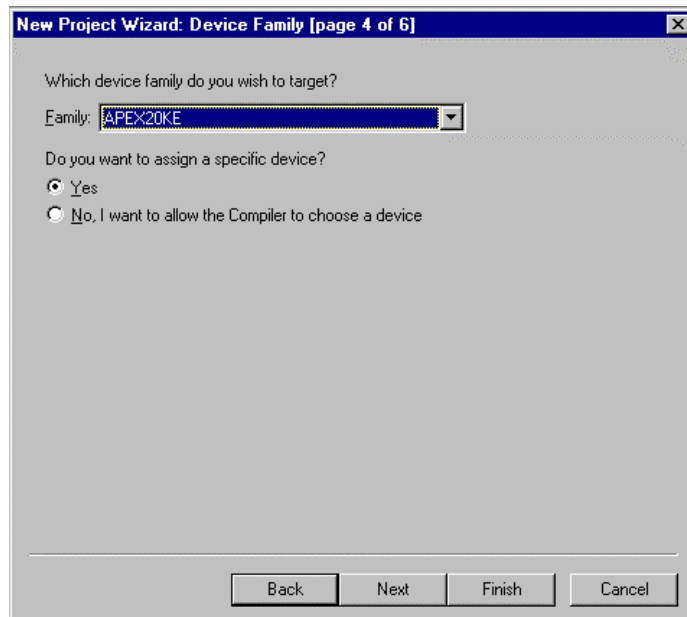
☐ Generate a compilable netlist automatically from the source files when they change

Settings... Advanced

Back Next Finish Cancel

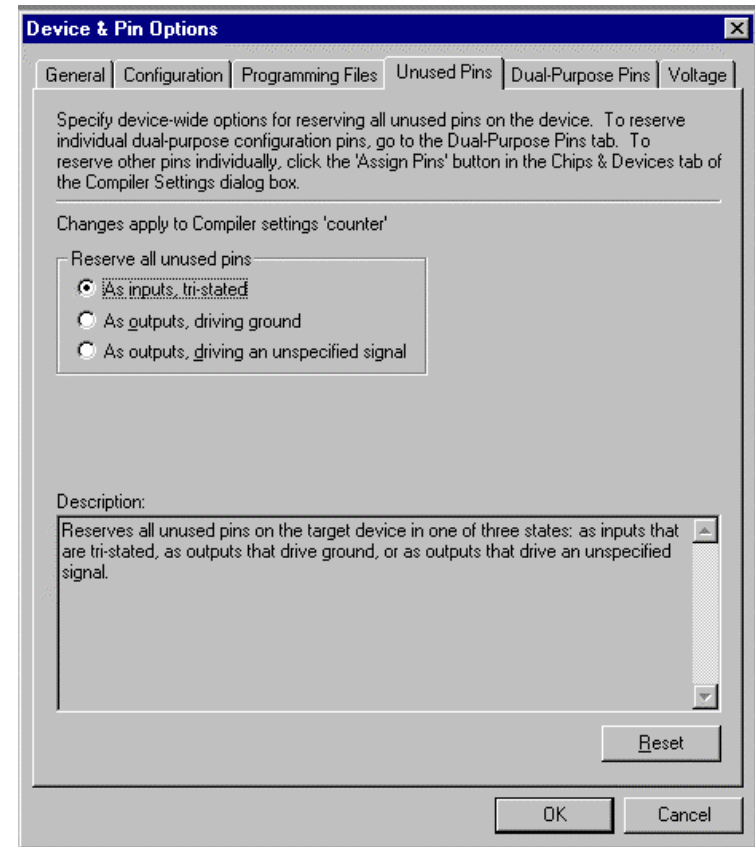
Create Project: Select Device

- Select APEX & Yes to Assign Specific Device
- Click Next
- Select EP20K200EFC484-2X
- Click Finish



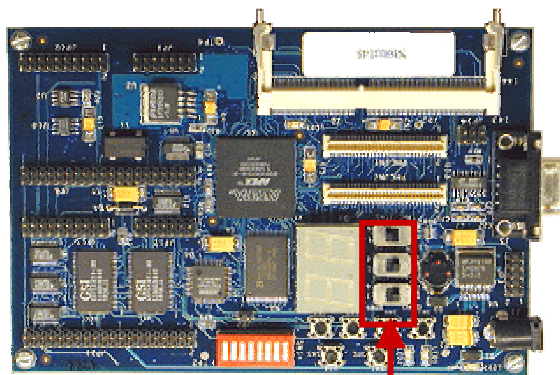
Set All Unused I/Os as Tri-Stated

- Choose Compiler Settings (Processing Menu)
- Click the Chips & Devices Tab
- Click Device & Pin Options Button
- Click the Unused Pins Tab
 - Select **As inputs, tri-stated**

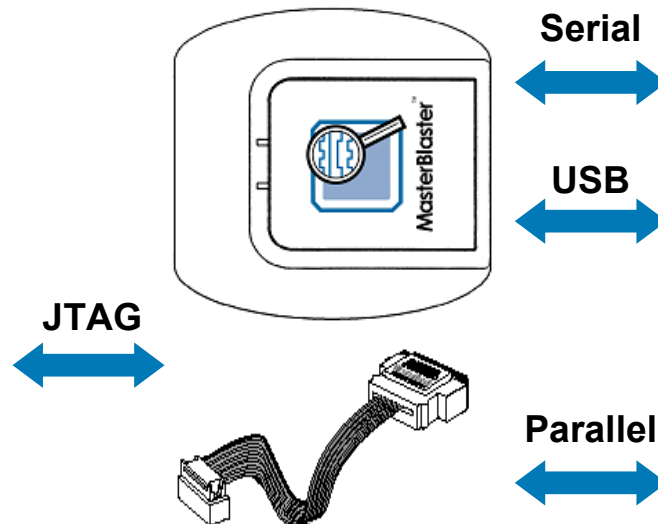


Hardware Setup

- Nios Demo Board with APEX EP20K200EFC484-2X Device
- DC Power Supply
- Device Download Cable (MasterBlaster or ByteBlasterMV Cable)



SW10 « JTAG »
SW9 « JTAG »
SW8 « APEX »

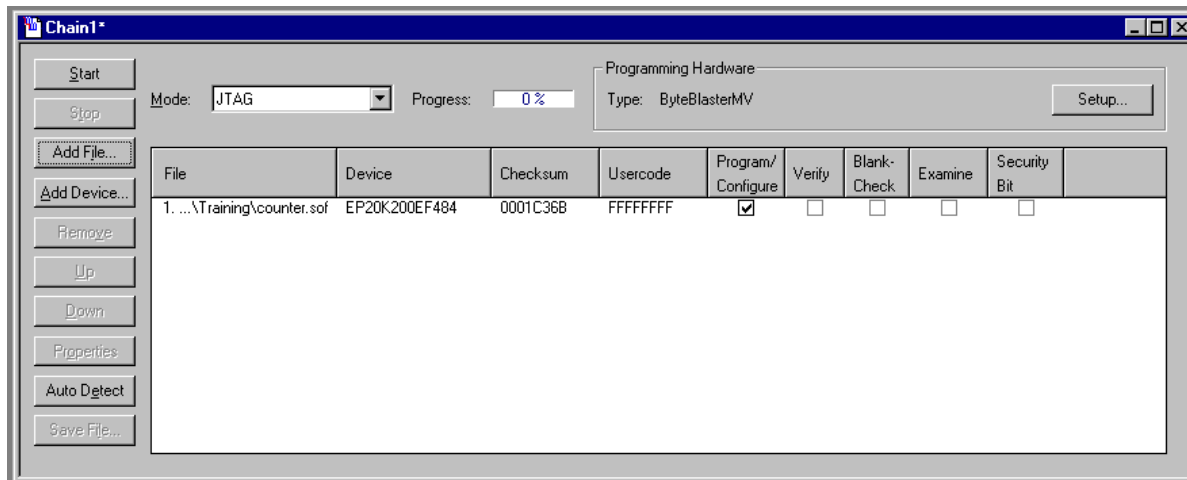


Serial
USB
Parallel



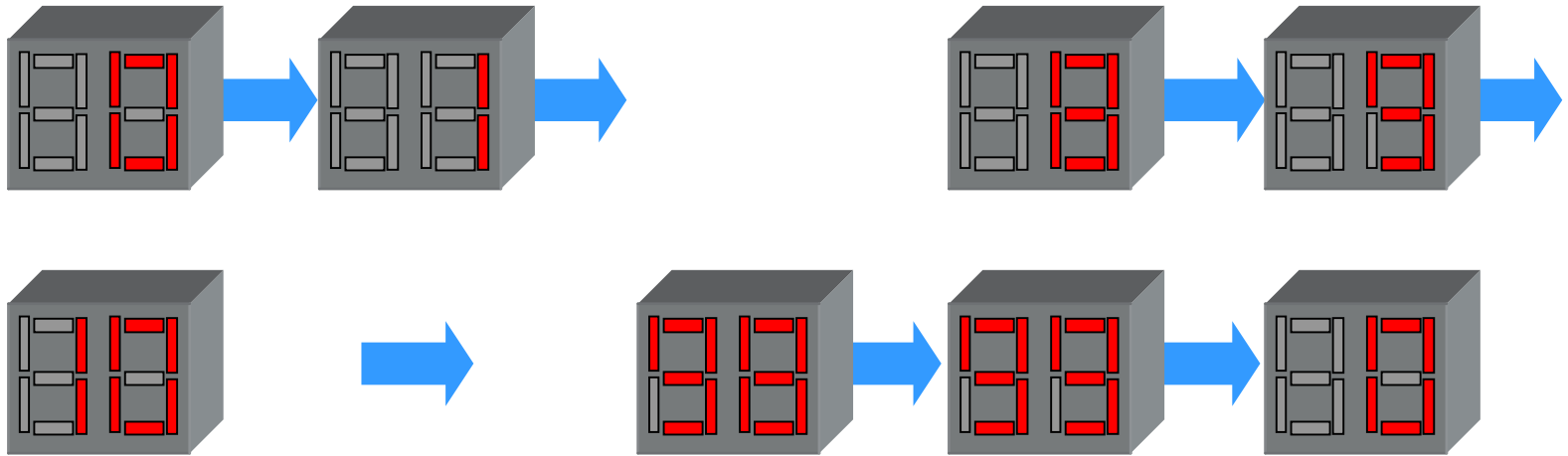
Compile Project & Configuration

- Choose Start Compilation (Processing menu)
- Choose Open Programmer (Processing menu)
 - Click Add File
 - Select the File counter.sof
 - Programming Hardware : ByteBlasterMV
 - Mode : JTAG
 - Check Option Program/Configure
- Click Start



Observing the Malfunction

- Number 9 Never Appears on the Least & Most Significant Digit



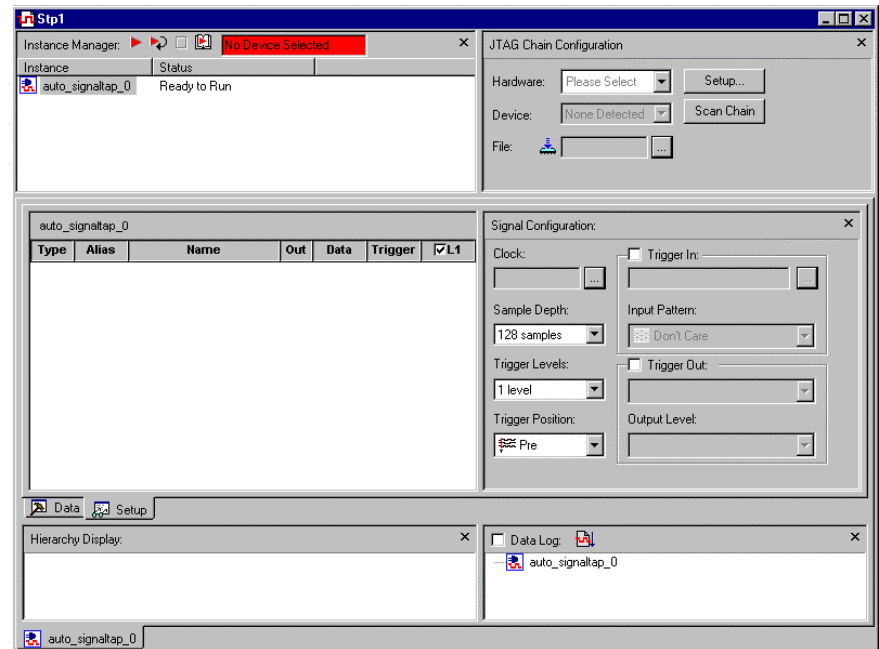
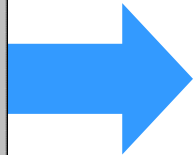
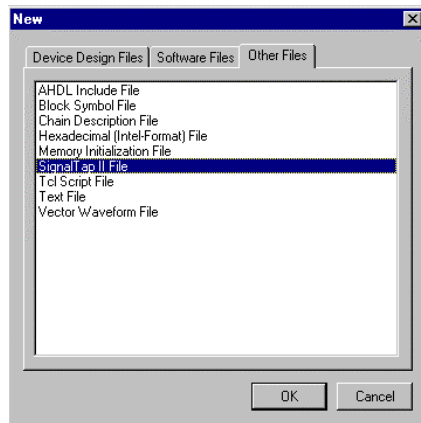
- SignalTap II Analyzer Used to Monitor Control Signals of LEDs

Essential Steps

- Configure SignalTap II Logic Analyzer
 - Select Nodes for Analysis
 - Select Acquisition Clock
 - Set Sample Depth & Trigger Options
 - Enable the Logic Analyzer & Compile
- Configure Device
- Set the Trigger Pattern
- Run the SignalTap II Logic Analyzer

Configure Logic Analyzer

- Choose New (File Menu)
- Click the Other Files Tab & Select SignalTap II File
- Click OK

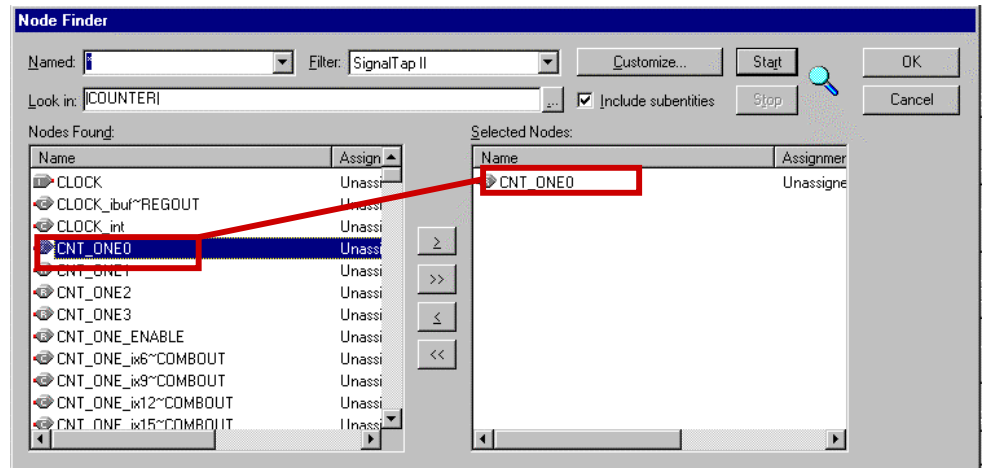


- Or Select 

- Save the SignalTap II File as **analysis.stp**

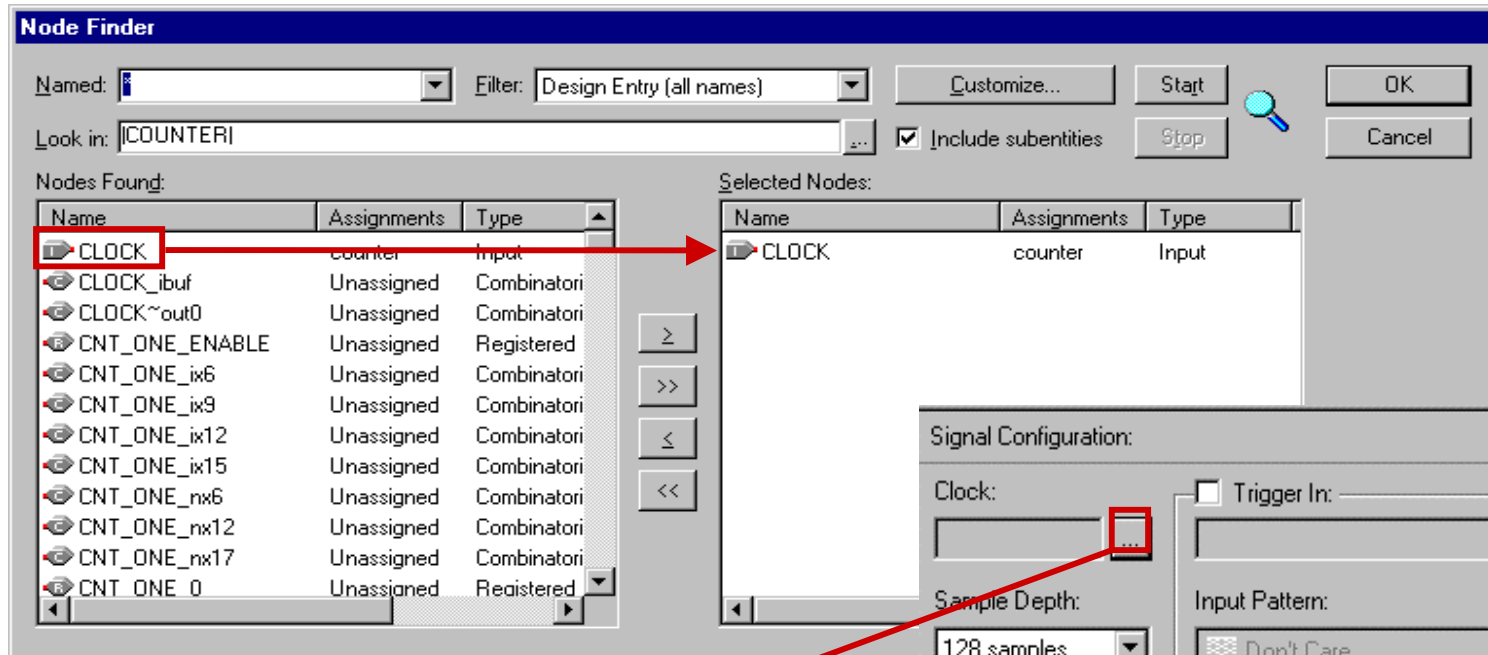
Analyzer 1: Select Nodes

- Open Node Finder Window
 - Double Click on the Signal Viewer
- Click Start to List All Pins & Internal Nodes
- Add Signal CNT_ONE_0 to Selected Nodes List
- Repeat
 - CNT_ONE_1
 - CNT_ONE_2
 - CNT_ONE_3
 - CNT_ONE_ENABLE
- Click OK

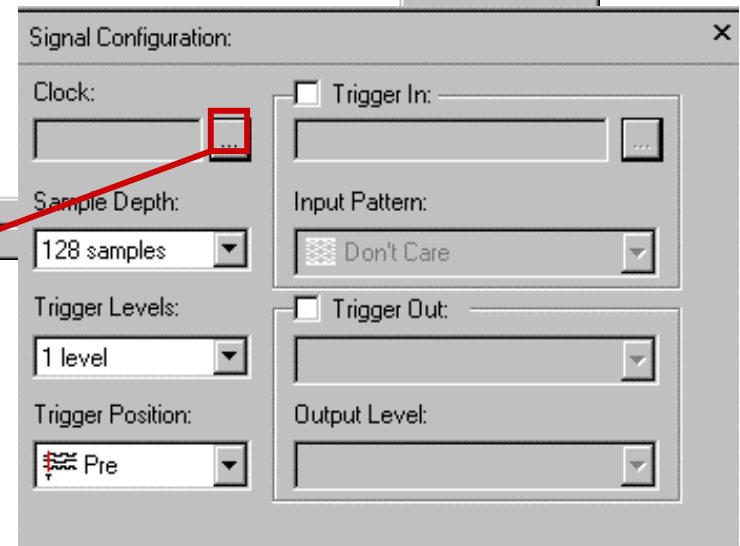


Analyzer 1: Acquisition Clock

■ Add Clock Input Pin from Node Finder



Open Node
Finder for Clock



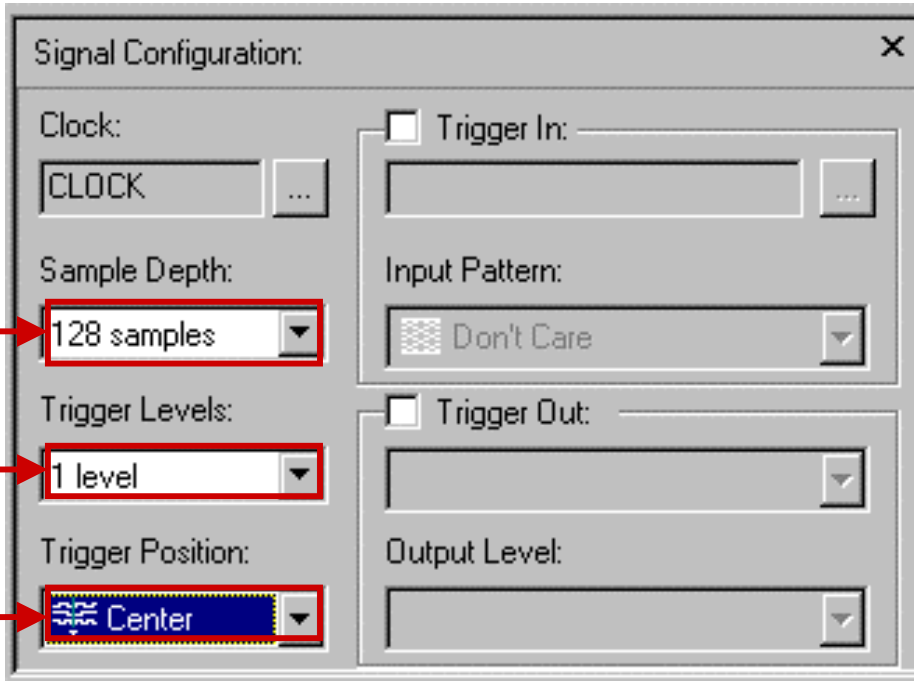
Configure Analyzer 1

- Configure Logic Analyzer 1 (auto_signaltap_0) as Follows

128 Samples

Single Level

Center



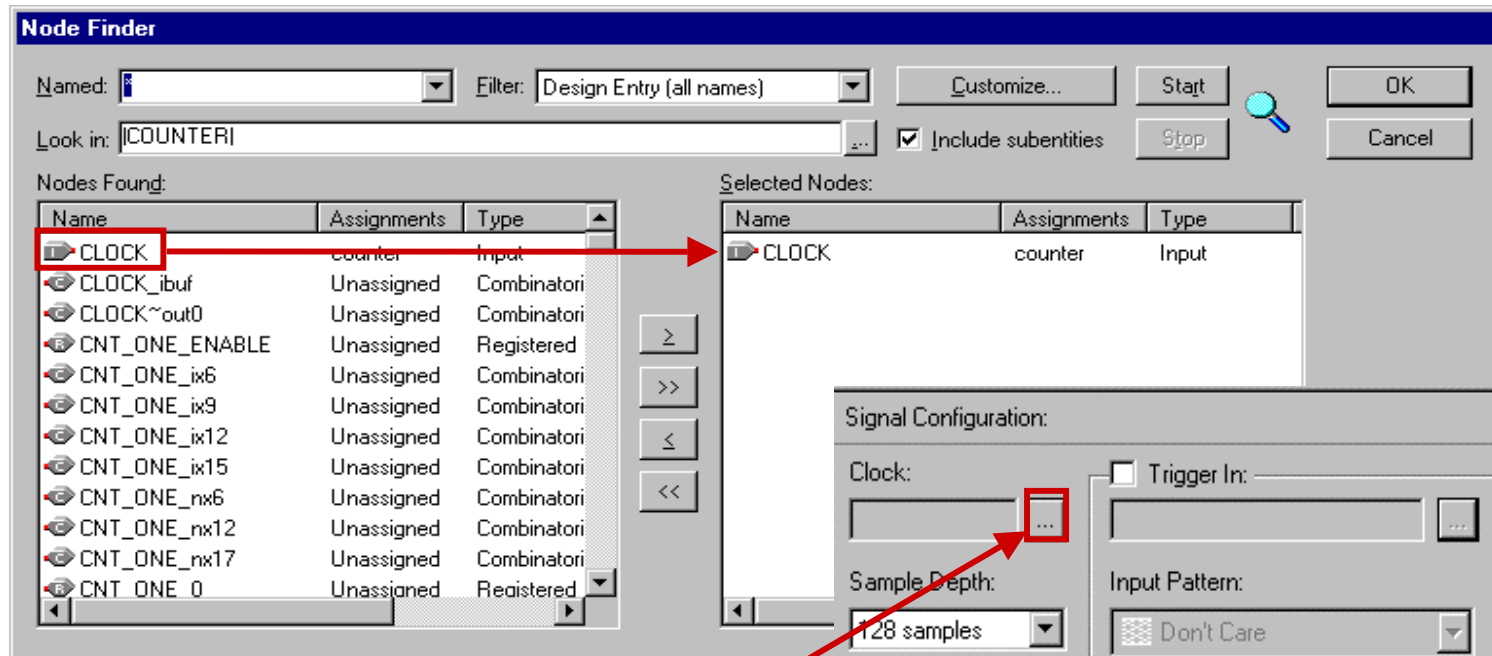
- Choose Save (File menu)

Analyzer 2: Select Nodes

- Right Click in Instance Manager & Select Create Instance
- Open the Node Finder Window
- Click Start to List All Pins & Internal Nodes
- Add Following Signals
 - CNT_ONE_0, CNT_ONE_1, CNT_ONE_2, CNT_ONE_3
 - CNT_ONE_ENABLE
 - CNT_TEN_0, CNT_TEN_1, CNT_TEN_2, CNT_TEN_3
- Click OK

Analyzer 2: Acquisition Clock

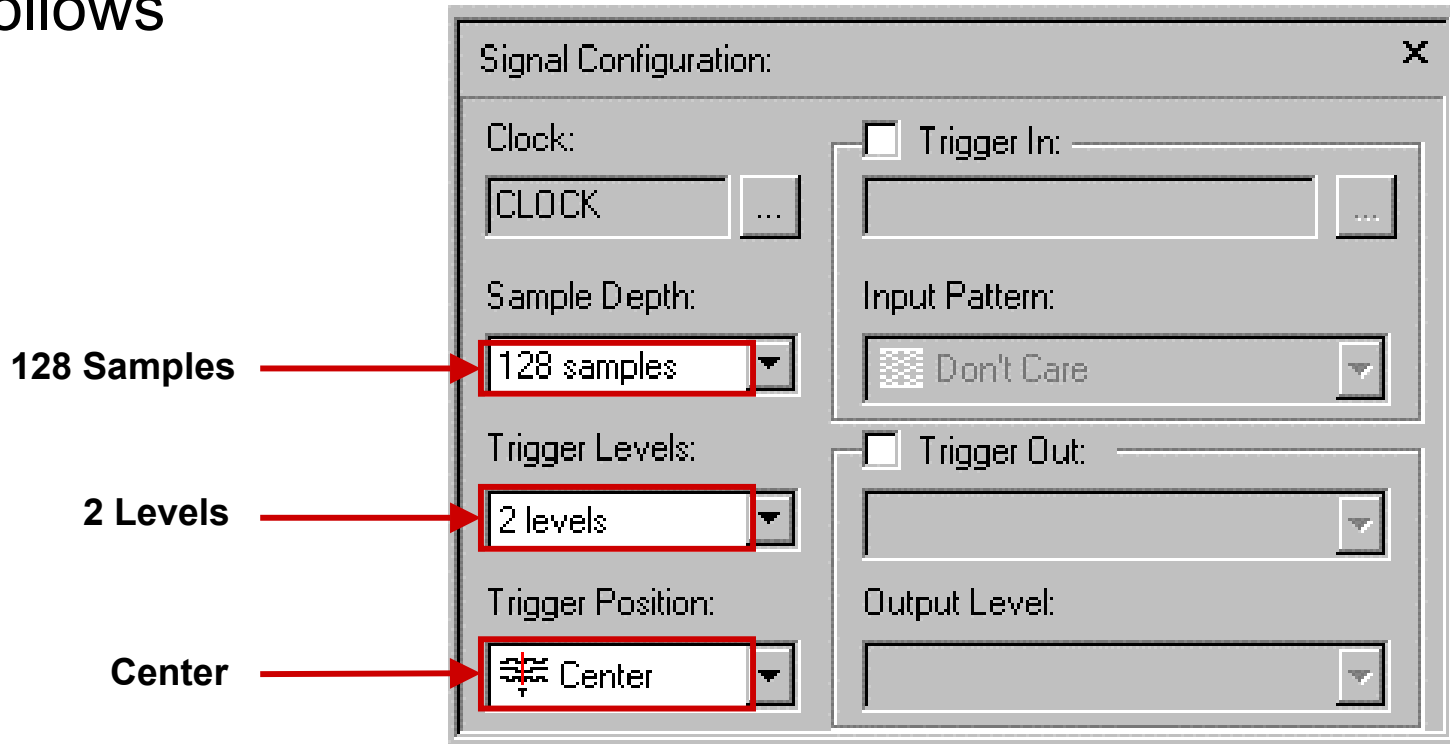
■ Add Clock Input Pin from Node Finder



Open Node Finder for Clock

Configure Analyzer 2

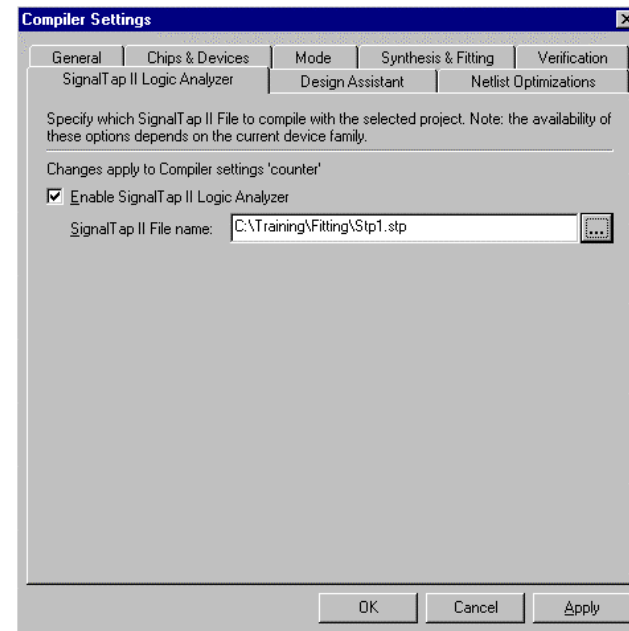
- Configure Logic Analyzer 2 (auto_signaltap_1) as Follows



- Choose Save (File menu)

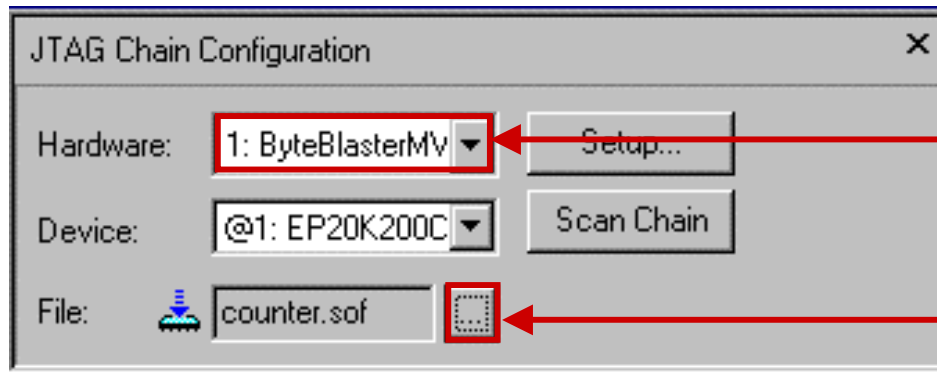
Compile Design with SignalTap II Logic Analyzer

- Choose Compiler Settings (Processing Menu)
- Click the SignalTap II Logic Analyzer Tab
- Turn on Enable SignalTap II Logic Analyzer
- Browse to analysis.stp File
- Click OK
- Choose Start Compilation
- (Processing menu)



Running Analysis

■ Select Hardware & Configuration File



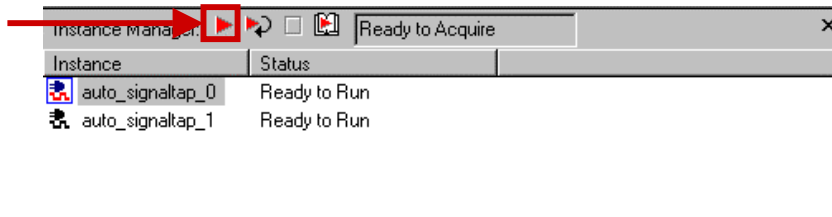
Selects ByteBlasterMV Cable

Opens File Browser
Select counter.sof

Running Analyzer 1

- Set Trigger Pattern When CNT_ONE Reaches 9
- Configure Device
- Run SignalTap II Analyzer

3. Run SignalTap II Analyzer

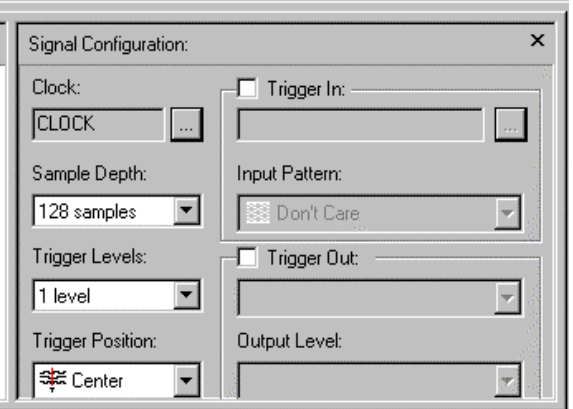
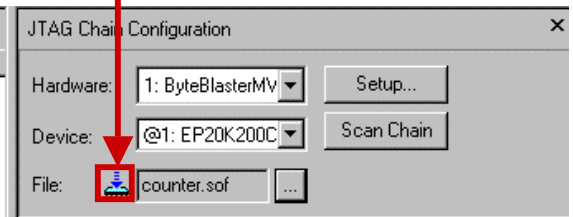


1. CNT_ONE Value 9

trigger: 2002/05/23 13:26:04 #1

Type	Alias	Name	Out	Data	Trigger	✓ L1
		CNT_ONE0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
		CNT_ONE1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
		CNT_ONE2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
		CNT_ONE3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
		CNT_ONE_ENABLE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

2. Configure Device



View Results

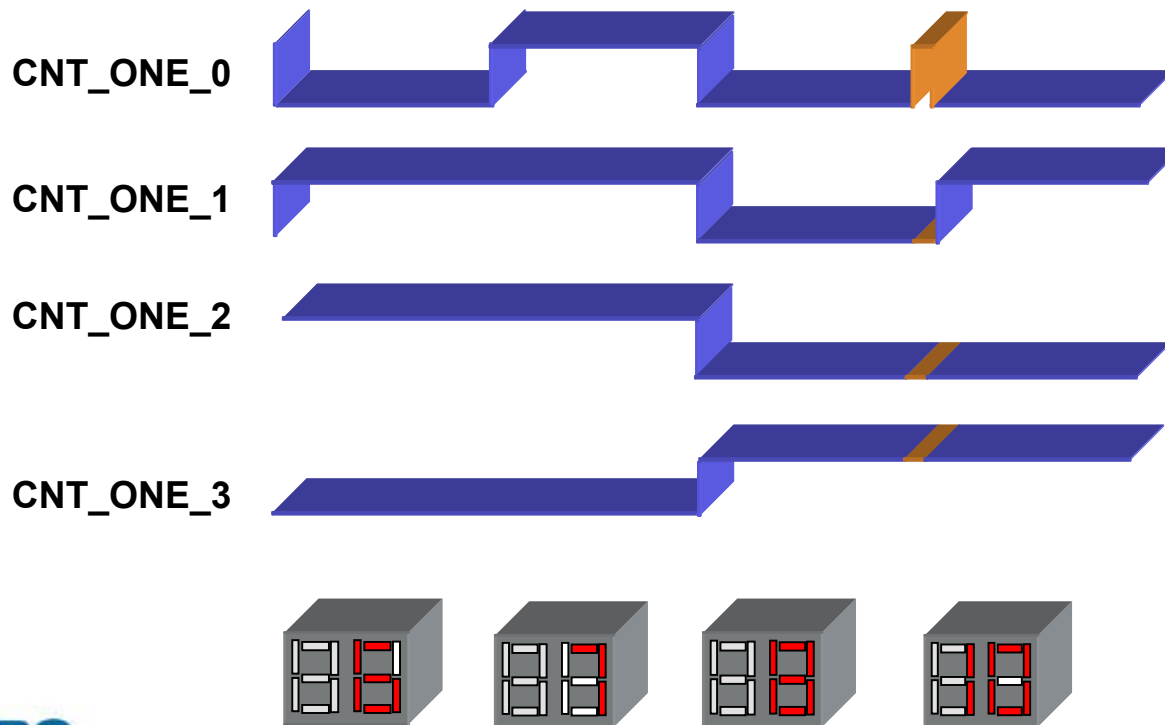
The screenshot displays the 'analysis.stp' application window. At the top, the 'Instance Manager' tab shows two instances, 'auto_sgnaltap_0' and 'auto_sgnaltap_1', both in a 'Ready to Run' state. To the right, the 'JTAG Chain Configuration' panel is visible, showing hardware settings like '1: ByteBlasterMV' and a device '@1: EP20K200C'. The main area features a signal viewer with a table of signals and their waveforms. The table has columns for 'Type', 'Alias', and 'Name'. The signals listed are CNT_ONE0, CNT_ONE1, CNT_ONE2, CNT_ONE3, and CNT_ONE_ENABLE. Below the table, there are tabs for 'Data' and 'Setup'. The 'Data' tab is selected, and a red arrow points to it from the text 'Results Displayed in Data Tab of Signal Viewer'. The 'Data' tab shows a 'Hierarchy Display' with a tree view containing 'counter'. To the right of the 'Data' tab is a 'Data Log' panel showing 'auto_sgnaltap_0'. At the bottom, there are buttons for 'auto_sgnaltap_0' and 'auto_sgnaltap_1'.

Type	Alias	Name
		CNT_ONE0
		CNT_ONE1
		CNT_ONE2
		CNT_ONE3
		CNT_ONE_ENABLE

Results Displayed
in Data Tab
of Signal Viewer

Isolating the Problem

- The CNT_ONE Counts to 9 but its Synchronous Reset Signal Occurs on the Next Clock Immediately Resetting the Counter to Zero



Analyzer 2 - Further Analysis

- Select Analyzer 2 - auto_signaltap_1

Double Click



The screenshot shows the 'analysis.stp' application window. The 'Instance Manager' panel at the top left contains a table with the following data:

Instance	Status
auto_signaltap_0	Ready to Run
auto_signaltap_1	Ready to Run

The 'JTAG Chain Configuration' panel on the top right shows: Hardware: 1: ByteBlasterMV, Device: @1: EP20K200C, File: counter.sof.

The 'Signal Configuration' panel on the right shows: Clock: CLOCK, Sample Depth: 128 samples, Trigger Levels: 2 levels, Trigger Position: Center.

The 'Hierarchy Display' panel at the bottom left shows a tree structure with 'counter' selected.

The 'Data Log' panel at the bottom right shows 'auto_signaltap_1' selected.

At the bottom of the window, there is a tab bar with two tabs: 'auto_signaltap_0' and 'auto_signaltap_1'. The 'auto_signaltap_1' tab is highlighted with a red box.

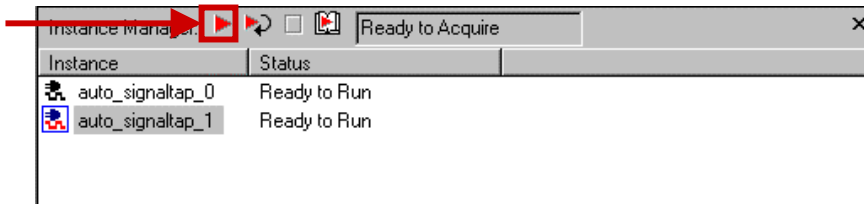
Select Tab



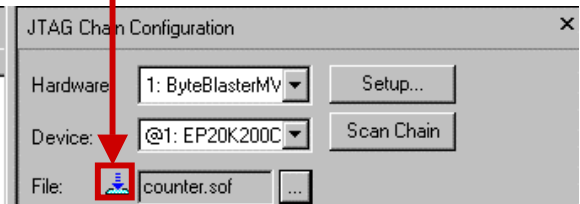
Running Analyzer 2

- Set Trigger Pattern As Shown
- Configure Device
- Run SignalTap II Analyzer

Run
SignalTap II
Analyzer



Configure
Device



trigger: 2002/05/24 09:34:45 #0

Type	Alias	Name	Out	Data	Trigger	✓L1	✓L2
		CNT_ONE0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—	—
		CNT_ONE1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—	
		CNT_ONE2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—	
		CNT_ONE3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—	—
		CNT_ONE_ENABLE		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
		CNT_TEN0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—	—
		CNT_TEN1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—	—
		CNT_TEN2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—	—
		CNT_TEN3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	—	—

Signal Configuration:

Clock: ...

Sample Depth:

Trigger Levels:

Trigger Position:

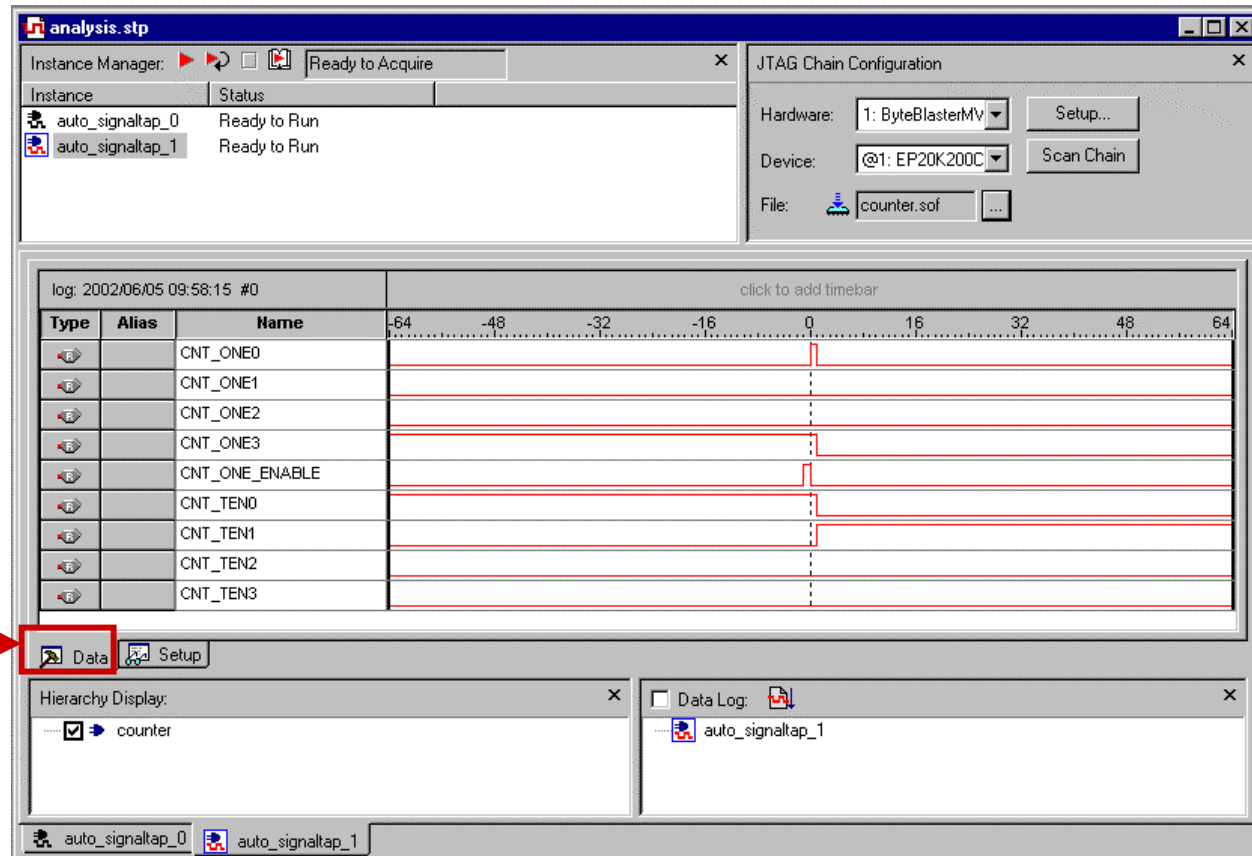
☐ Trigger In:

☐ Trigger Out:

Trigger on 19 after Encountering 20 –
Notice that Analyzer Does Trigger on First Instance of 19

View Results

Results Displayed
in Data Tab
of Signal Viewer





SOPC
WORLD
2 0 0 2

Conclusions

SignalTap II Benefits

- Access Internal Signals within Design
- Easy Configuration through Quartus II Interface
- Using SignalTap II Analyzer Does Not Require Making Modifications to Design Files
- Available with Quartus II Software