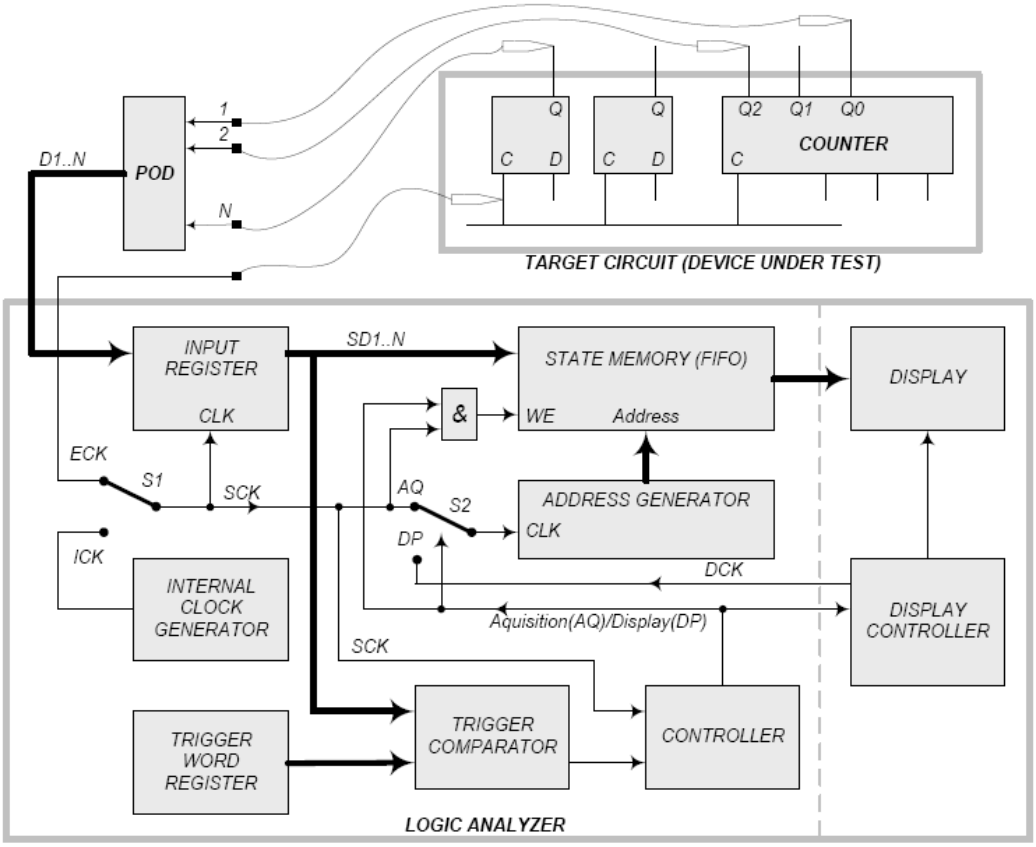
Logic analyzers and Xilinx ChipScope

1. About logic analyzers

For the examination of logic circuits and digital devices, one needs to register simultaneously plenty of digital (0-1) signals. The logic waveforms should be stored for evaluation. This can be done with a logic analyzer. The schematic of a logic analyzer is shown below.

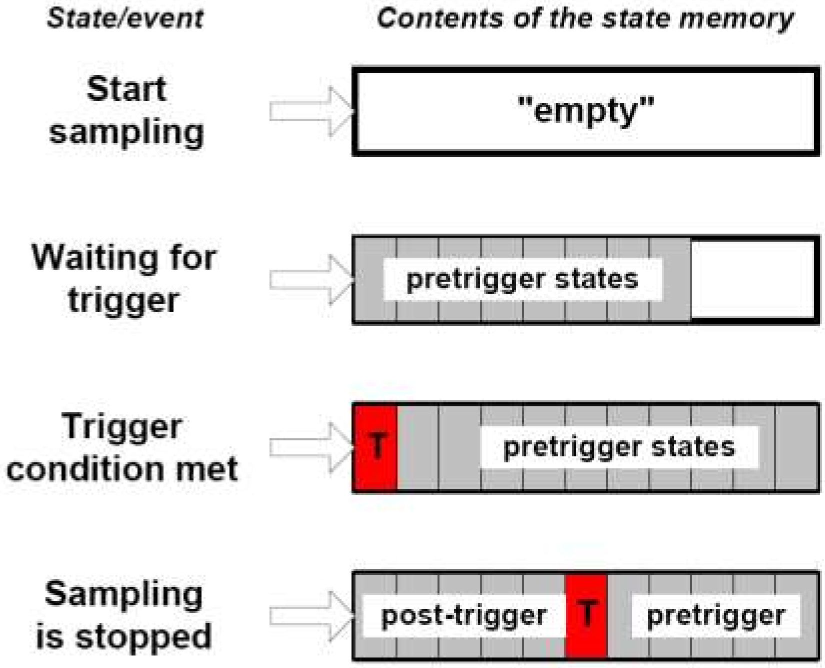


The signals of the device under test (DUT) are connected to D1..N analyzer inputs. The POD performs signal conditioning towards the stable detection of the logic levels. The signals will be sampled and stored in the state memory. The sampling is scheduled by the SCK Clock. The analyzer has two different modes:

In **state analysis mode** the sampling clock is the same as the DUT clock (connected to the ECK input of the analyzer), the sampling itself is synchronous. The DUT states will be stored in the state memory.

In **timing analysis mode** the analyzer uses its own internal clock source (ICK) for sampling, the timing of the DUT can be evaluated, if the ICK frequency is high enough. This frequency must be selected carefully, determined by the desired resolution in time and the relevant input signal length. (A high frequency ICK results high accuracy but the state memory will be filled rapidly.)

The sampling is controlled through the **trigger condition**. After the condition is fulfilled, the sampling is stopped and the state memory can be read (displayed). The exact trigger position can be set up, pre- and post-trigger events can be evaluated.



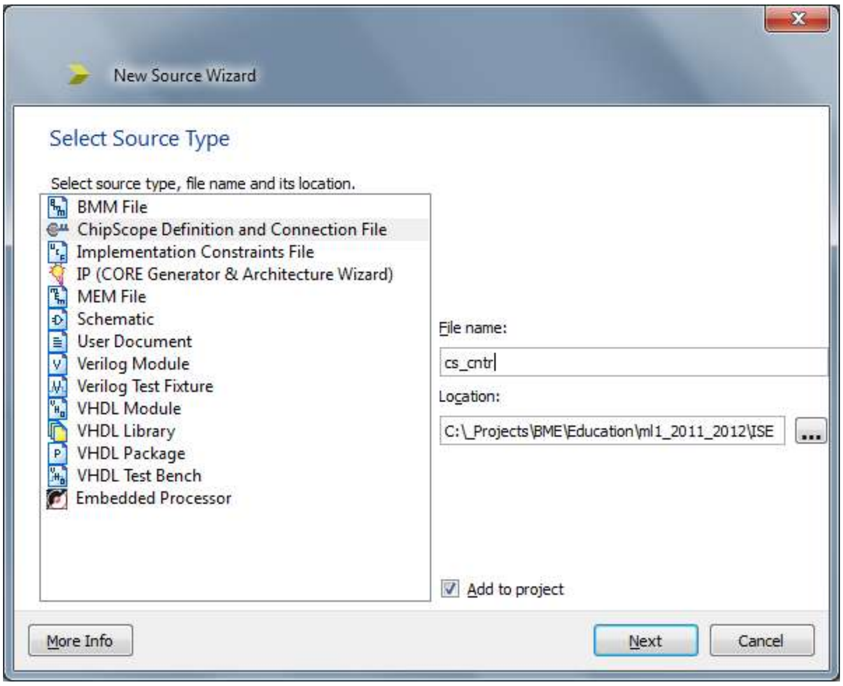
2. The Xilinx ChipScope

The ChipScope is a logic analyzer implemented in the FPGA together with the designed hardware to test (DUT). Both DUT and ChipScope use the System Clock, thus ChipScope is used generally for state analysis.

The ChipScope can be added to a design and configured with ChipScope Core Inserter. ChipScope Analyzer is the GUI for the measurement

2.1. Using ChipScope Core Inserter

Add a **ChipScope Definition and Connection File** as a new source to your project!

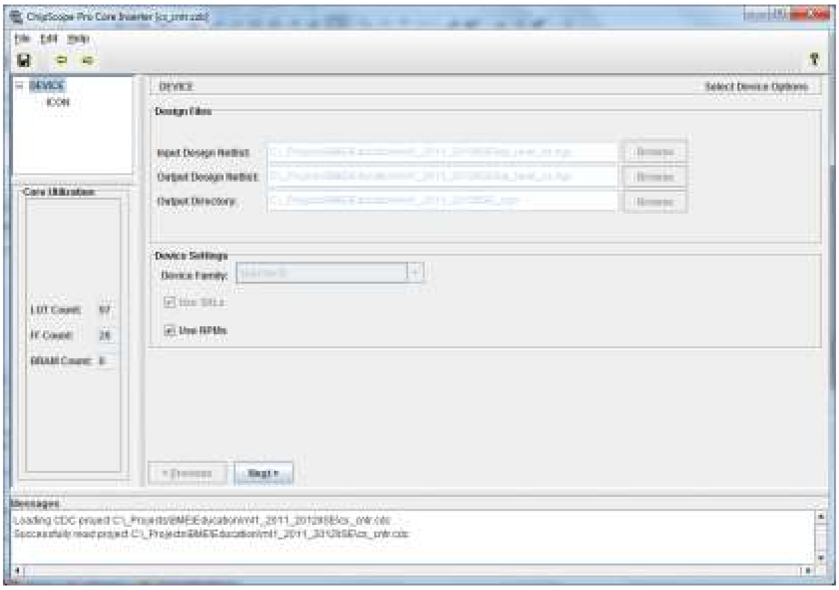


File name: cs_cntr

Location: C:_Projects\BME\Education\ml1_2011_2012\ISE

☑ Add to project

Double clicking the cdc file, you can open the core inserter. In the first window just click next!



LUT Count: 97

FF Count: 26

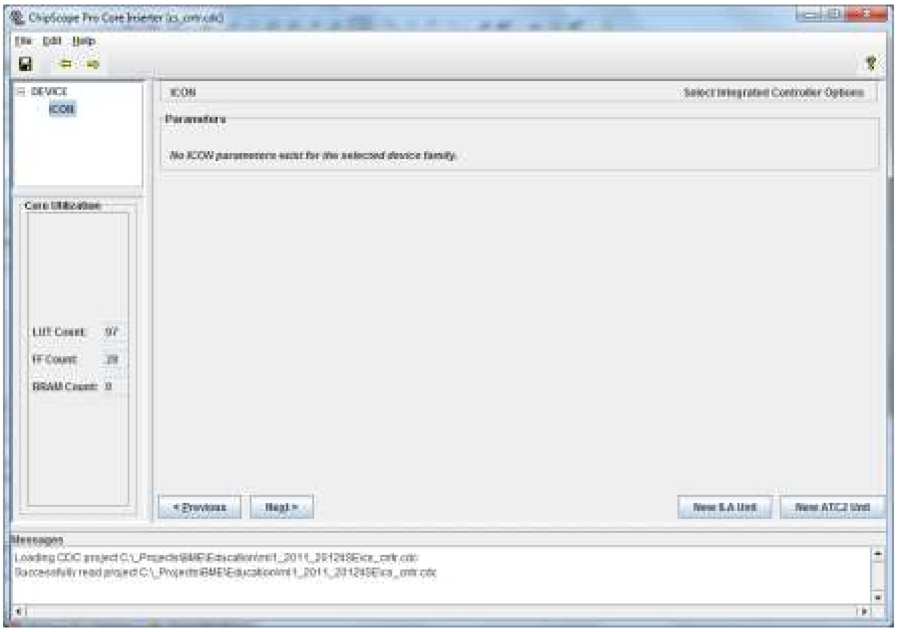
BRAM Count: 0

Messages

Loading CDC project C:_Projects\BME\Education\ml1_2011_2012\ISE\cs_cntr.cdc

Successfully read project C:_Projects\BME\Education\ml1_2011_2012\ISE\cs_cntr.cdc

The Integrated controller (ICON) is the main modul interfacing the PC through the JTAG cable and the Integrated Logic Analyzer (ILA) moduls. Using a Spartan-3 FPGA, ICON has no configuration parameters and only one ICON module can be used, so click next!

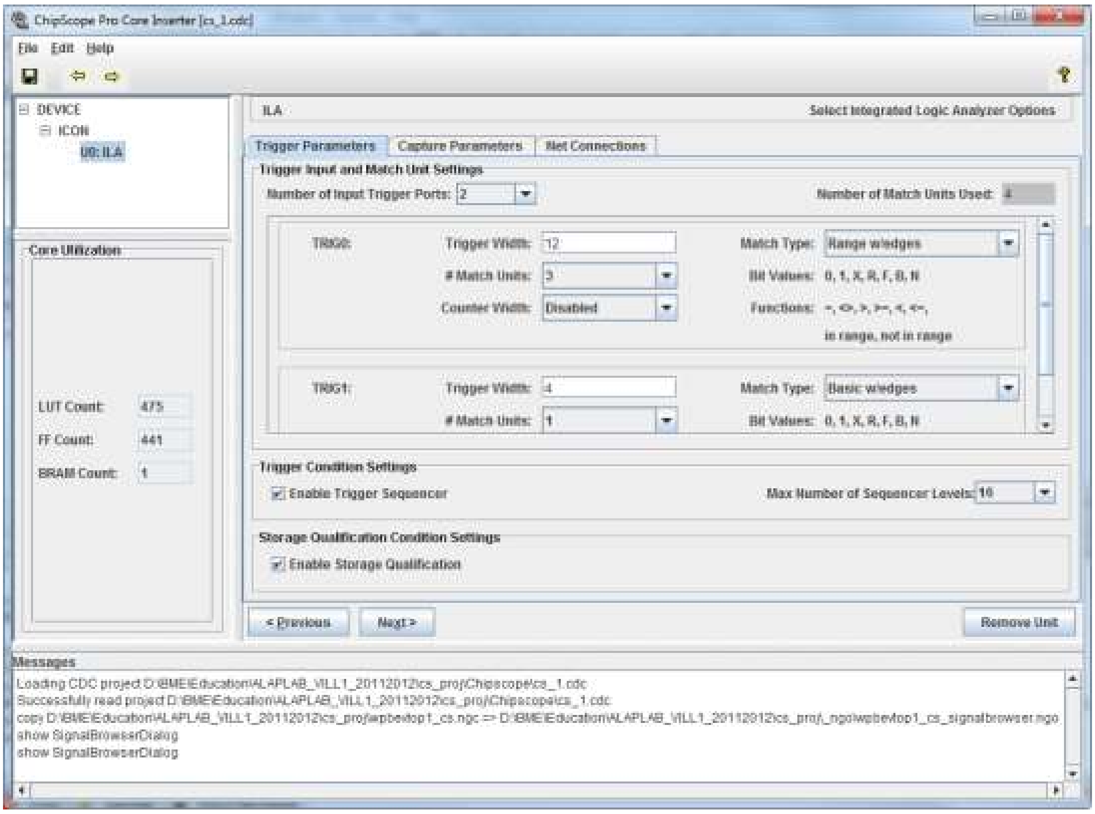


No ICON parameters exist for the selected device family.

LUT Count: 97
FF Count: 28
BRAM Count: 0

```
Loading CDC project C:\_Projects\BME\Education\mit1_2011_2012\ISE\cs_cntr.cdc
Successfully read project C:\_Projects\BME\Education\mit1_2011_2012\ISE\cs_cntr.cdc
```

16 ILAs can be connected to one ICON, but during the lab exercises you will need only one. The ILA configuration defines the parameters of the logic analyzer. In the main window, you can see the currently used resorces (LUT,FF,BRAM count) and the configuration options on the right side.


ChipScope Pro Core Inserter [cs_1.cdc]
File Edit Help
DEVICE
ICON
U0: ILA
ILA
Select Integrated Logic Analyzer Options
Trigger Parameters
Capture Parameters
Net Connections
Trigger Input and Match Unit Settings
Trigger Condition Settings
Storage Qualification Condition Settings
Core Utilization
< Previous
Next >
Remove Unit
Messages

LUT Count: 475
FF Count: 441
BRAM Count: 1

Number of Input Trigger Ports: 2
Number of Match Units Used: 4

TRIG0: Trigger Width: 12
# Match Units: 3
Counter Width: Disabled
Match Type: Range w/edges
Bit Values: 0, 1, X, R, F, B, N
Functions: =, <>, >, >=, <, <=, in range, not in range

TRIG1: Trigger Width: 4
# Match Units: 1
Match Type: Basic w/edges
Bit Values: 0, 1, X, R, F, B, N

☑ Enable Trigger Sequencer
Max Number of Sequencer Levels: 16

☑ Enable Storage Qualification

```
Loading CDC project D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\Chipscope\cs_1.cdc
Successfully read project D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\Chipscope\cs_1.cdc
copy D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\wpbevtop1_cs.ngc => D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\_ngo\wpbevtop1_cs_signalbrowser.ngo
show SignalBrowserDialog
show SignalBrowserDialog
```

Trigger parameters

You can define maximum 16 trigger ports. Each of them can use different trigger signals. The trigger conditions can be defined by means of Match units. The match units can count the fulfilled trigger conditions, so it is possible to trigger for example on the 42th rising edge of a signal. First and last, the **number of trigger ports** must be given, and you have to define **the trigger bitwidth**, **number of match units** and **counter width**. The **match type** must be one of the following:

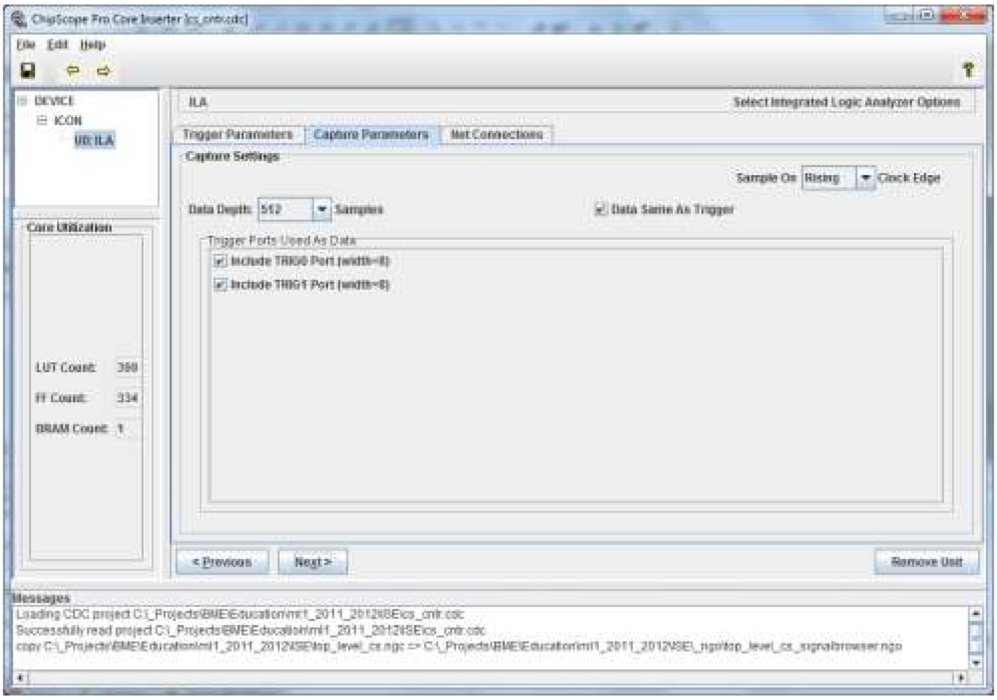
- Basic: the trigger condition can be defined with 0,1 or X (don't) care bit values.
- Basic w/edges: besides the signal levels, edges can be specified in the trigger: Rising edge, Falling edge, Both edges or No transition.
- Extended: bit values (0,1,X) and trigger world relations (equal, less/greater than…) can be evaluated.
- Extended w/edges: same as previous and edges (R,F,B,N)
- Range: extended and range check (Check if trigger word is between two defined values)
- Range w/edges: same as previous and edges (R,F,B,N)

Trigger sequencer makes it possible to trigger only if various match unit conditions are met in the defined order. **Sequencer level** must also be defined. E.g. M0 → M1 → M0 is a 3 level sequence (using 2 match units.)

If **enable storage qualification** is checked, the sampling (based on system clock edge) can be enabled/disabled by means of a match unit.

Capture parameters

You can select, whether the data bits to sample are the same as the trigger port bits (**Data same as trigger**). The sampling occurs on rising or falling edge of the system clock. **Data depth** gives the size of the state memory in words. (The word length is the bit-width of the data port.)



Sample On: Rising Clock Edge

Data Depth: 512 Samples

☑ Data Same As Trigger

Trigger Ports Used As Data

☑ Include TRIG0 Port (width=8)

☑ Include TRIG1 Port (width=8)

LUT Count: 399

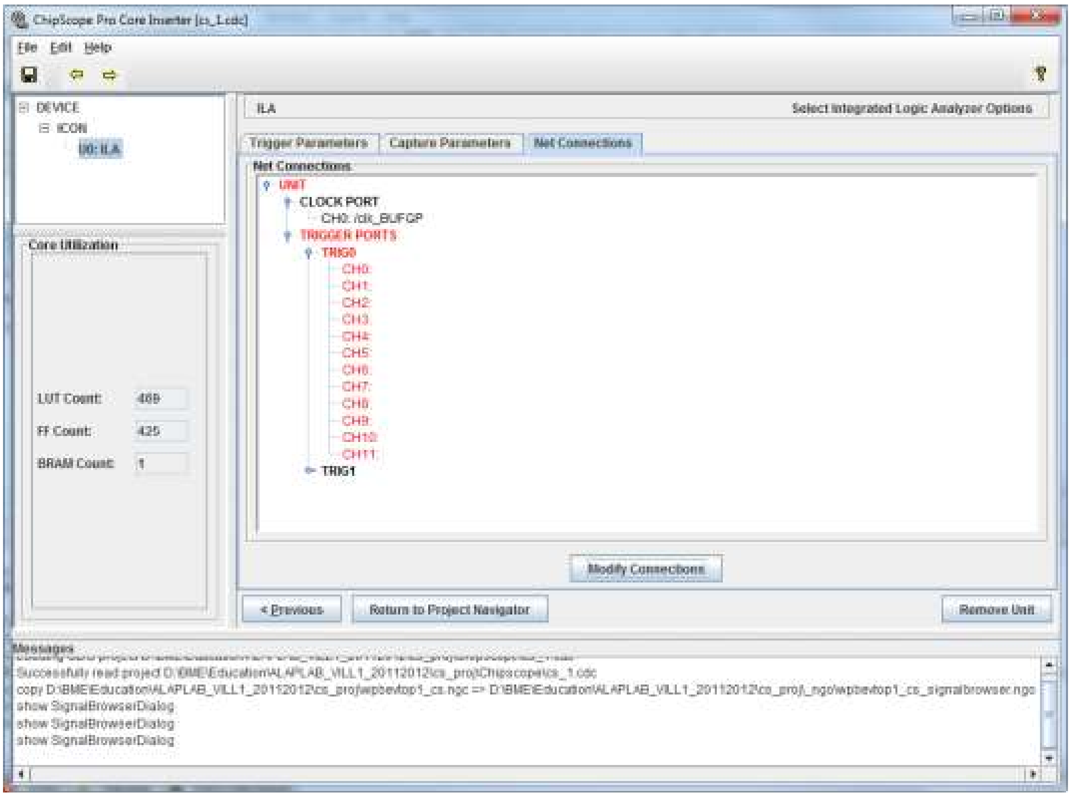
FF Count: 334

BRAM Count: 1

```
Loading CDC project C:\_Projects\BME\Education\ml1_2011_2012\ISE\cs_cntr.cdc
Successfully read project C:\_Projects\BME\Education\ml1_2011_2012\ISE\cs_cntr.cdc
copy C:\_Projects\BME\Education\ml1_2011_2012\ISE\top_level_cs.ngc => C:\_Projects\BME\Education\ml1_2011_2012\ISE\_ngo\top_level_cs_signalbrowser.ngo
```

Net connections

The connections between the DUT and ChipScope can be defined after clicking on 'Modify connections'. The *post-synthesis signals* of the DUT can be assigned to ChipScope channels.

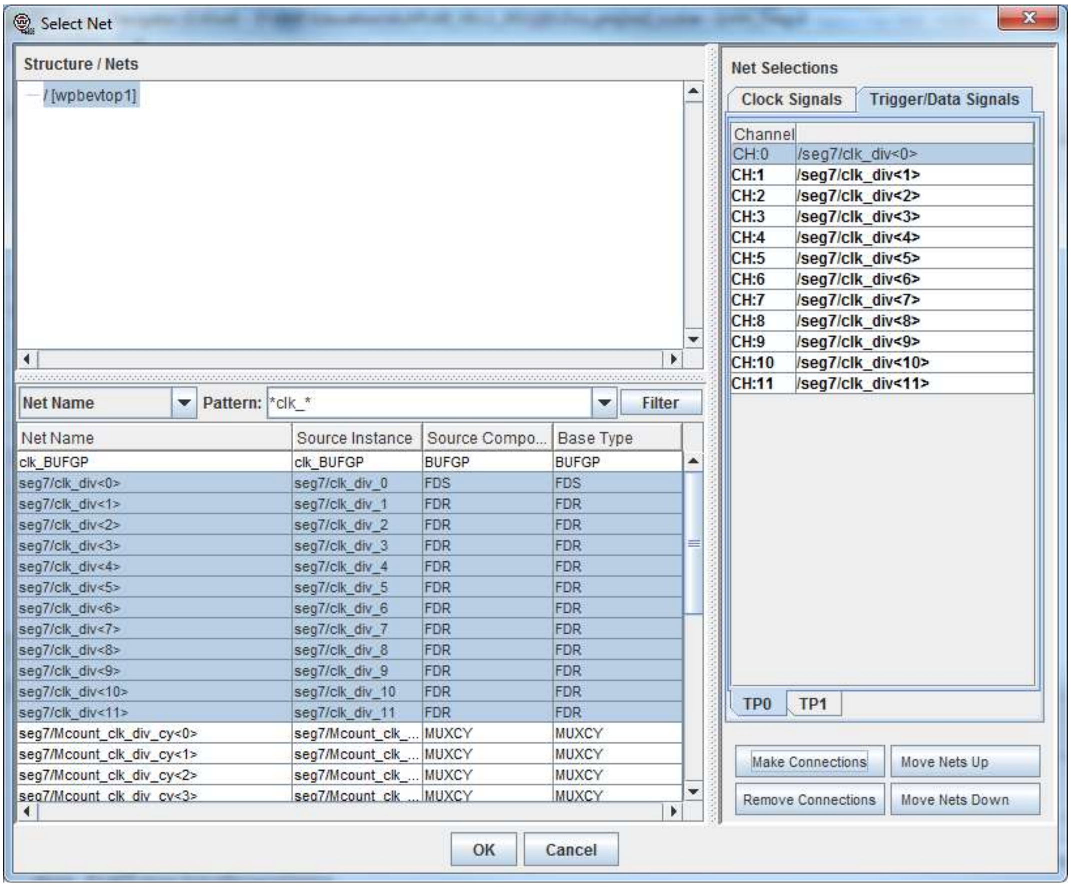


Net Connections

- UNIT
  - CLOCK PORT
    - CH0: /clk_BUFGP
  - TRIGGER PORTS
    - TRIG0
      - CH0:
      - CH1:
      - CH2:
      - CH3:
      - CH4:
      - CH5:
      - CH6:
      - CH7:
      - CH8:
      - CH9:
      - CH10:
      - CH11:
    - TRIG1

LUT Count: 469
FF Count: 425
BRAM Count: 1

```
Successfully read project D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\Chipscope\cs_1.cdc
copy D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\wpbevtop1_cs.ngc => D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\_ngo\wpbevtop1_cs_signalbrowser.ngc
show SignalBrowserDialog
show SignalBrowserDialog
show SignalBrowserDialog
```



Pattern: *clk_*

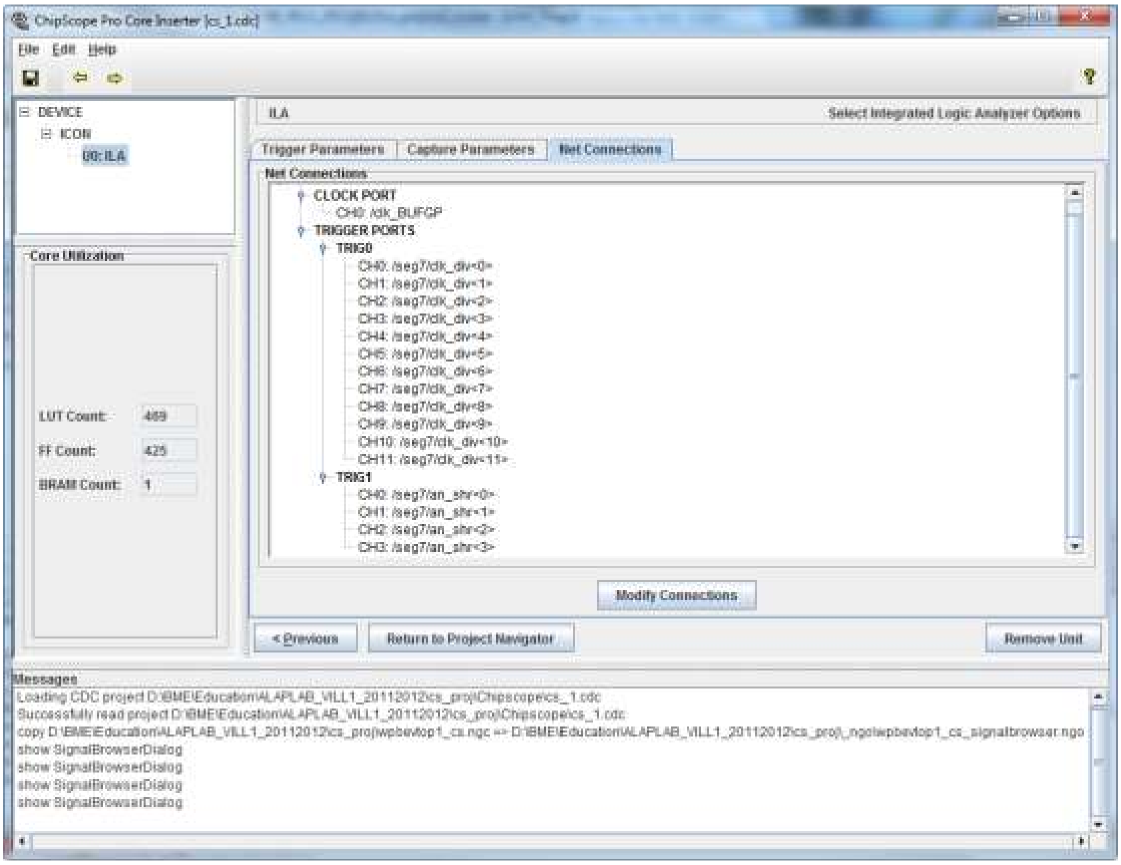
Net Name	Source Instance	Source Compo...	Base Type
clk_BUFGP	clk_BUFGP	BUFGP	BUFGP
seg7/clk_div<0>	seg7/clk_div_0	FDS	FDS
seg7/clk_div<1>	seg7/clk_div_1	FDR	FDR
seg7/clk_div<2>	seg7/clk_div_2	FDR	FDR
seg7/clk_div<3>	seg7/clk_div_3	FDR	FDR
seg7/clk_div<4>	seg7/clk_div_4	FDR	FDR
seg7/clk_div<5>	seg7/clk_div_5	FDR	FDR
seg7/clk_div<6>	seg7/clk_div_6	FDR	FDR
seg7/clk_div<7>	seg7/clk_div_7	FDR	FDR
seg7/clk_div<8>	seg7/clk_div_8	FDR	FDR
seg7/clk_div<9>	seg7/clk_div_9	FDR	FDR
seg7/clk_div<10>	seg7/clk_div_10	FDR	FDR
seg7/clk_div<11>	seg7/clk_div_11	FDR	FDR
seg7/Mcount_clk_div_cy<0>	seg7/Mcount_clk_...	MUXCY	MUXCY
seg7/Mcount_clk_div_cy<1>	seg7/Mcount_clk_...	MUXCY	MUXCY
seg7/Mcount_clk_div_cy<2>	seg7/Mcount_clk_...	MUXCY	MUXCY
seg7/Mcount_clk_div_cy<3>	seg7/Mcount_clk_...	MUXCY	MUXCY

Channel	
CH:0	/seg7/clk_div<0>
CH:1	/seg7/clk_div<1>
CH:2	/seg7/clk_div<2>
CH:3	/seg7/clk_div<3>
CH:4	/seg7/clk_div<4>
CH:5	/seg7/clk_div<5>
CH:6	/seg7/clk_div<6>
CH:7	/seg7/clk_div<7>
CH:8	/seg7/clk_div<8>
CH:9	/seg7/clk_div<9>
CH:10	/seg7/clk_div<10>
CH:11	/seg7/clk_div<11>

Generally, in a proper design register names will be kept after synthesis, but combinational logics can be simplified. The base type gives the module type of the signal source:

- BUFGP: clock buffer. Use this as the ChipScope clock signal
- IBUF: input buffer, connected to an FPGA pin, used as input.
- OBUF: output buffer, connected to an FPGA pin, used as output.
- IOBUF: bi-directional buffer. It cannot be connected to ChipScope.
- GND/VCC: constant logical 0/1.
- FD*: some version of D flip-flop (eg.: FDRSE is a D FF with reset, set, enable).
- LUTx stands for Look Up Table with x used inputs.

The patter field can help to search for the desired signals. You can use wildcards. After assigning all ChipScope channels you can return to project navigator.



Net Connections

```
CLOCK PORT
   CH0: /clk_BUFGP
TRIGGER PORTS
   TRIG0
      CH0: /seg7/clk_div<0>
      CH1: /seg7/clk_div<1>
      CH2: /seg7/clk_div<2>
      CH3: /seg7/clk_div<3>
      CH4: /seg7/clk_div<4>
      CH5: /seg7/clk_div<5>
      CH6: /seg7/clk_div<6>
      CH7: /seg7/clk_div<7>
      CH8: /seg7/clk_div<8>
      CH9: /seg7/clk_div<9>
      CH10: /seg7/clk_div<10>
      CH11: /seg7/clk_div<11>
   TRIG1
      CH0: /seg7/an_shr<0>
      CH1: /seg7/an_shr<1>
      CH2: /seg7/an_shr<2>
      CH3: /seg7/an_shr<3>
```

LUT Count: 469

FF Count: 425

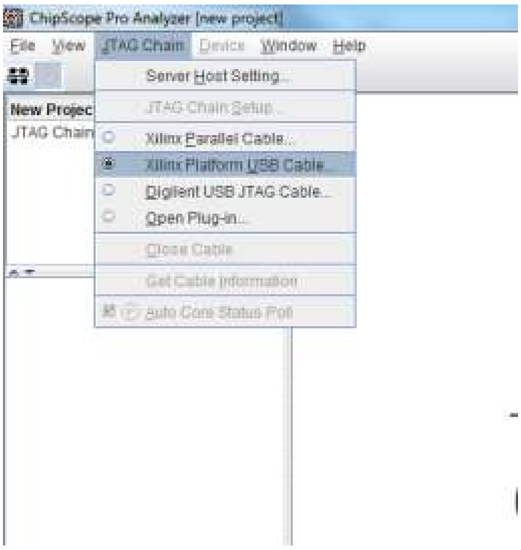
BRAM Count: 1

```
Loading CDC project D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\Chipscope\cs_1.cdc
Successfully read project D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\Chipscope\cs_1.cdc
copy D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\wpbevtop1_cs.ngc => D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\_ngo\wpbevtop1_cs_signalbrowser.ngo
show SignalBrowserDialog
show SignalBrowserDialog
show SignalBrowserDialog
show SignalBrowserDialog
```

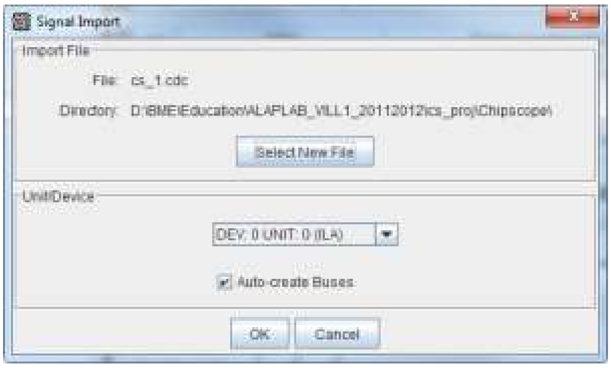
The design with the CDC file must be re-synthesized. The synthesis of the analyzer takes quite long time, and it must be repeated every time when you change the chip scope configuration. So double-check the settings if you want to analyze a design!

2.2. Analyzis with ChipScope

Download the new bit-file and start the ChipScope Analyzer GUI! Select JTAG Chain/Xilinx Platform USB Cable in the upper menu, click Ok on the pop-up windows to connect to ChipScope module!



Select File/Import to import the signal names and define busses based on the cdc file you have made before!


Signal Import

Import File

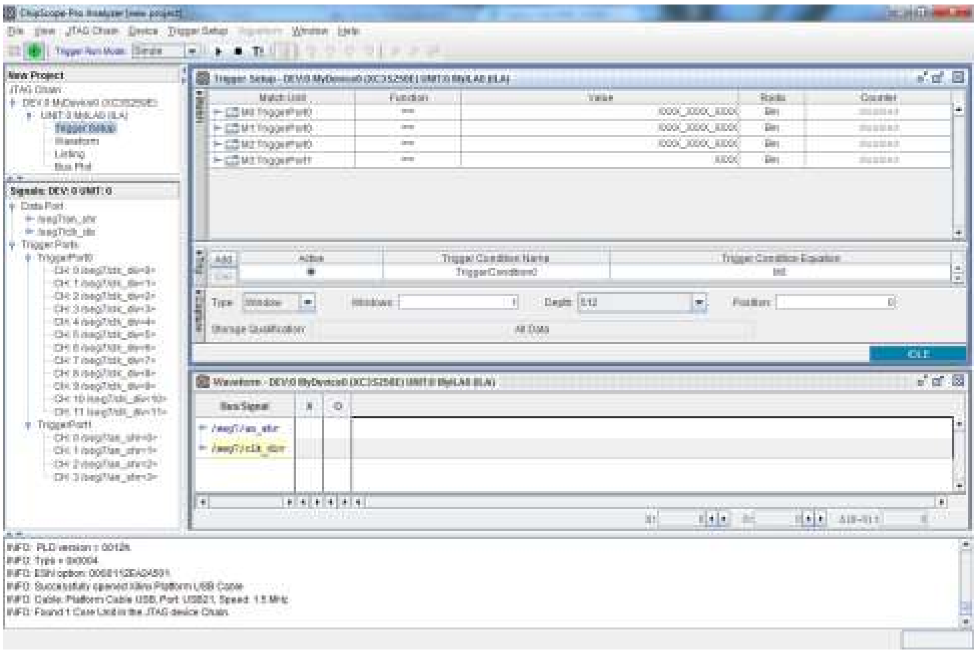
File: cs_1.cdc

Directory: D:\BME\Education\ALAPLAB_VILL1_20112012\cs_proj\Chipscope\

Unit/Device: DEV: 0 UNIT: 0 (ILA)

☑ Auto-create Buses

In our example, the signals are the 12-bit clock divider and the 4-bit anode signal for a 4-digit 7-segment display controller.


Trigger Setup - DEV:0 MyDevice0 (XC3S250E) UNIT:0 MyILA0 (ILA)

Match Unit	Function	Value	Radix	Counter
M0:TriggerPort0	==	XXXX_XXXX_XXXX	Bin	disabled
M1:TriggerPort0	==	XXXX_XXXX_XXXX	Bin	disabled
M2:TriggerPort0	==	XXXX_XXXX_XXXX	Bin	disabled
M3:TriggerPort1	==	XXXX	Bin	disabled

Active	Trigger Condition Name	Trigger Condition Equation
●	TriggerCondition0	M0

Type: Window   Windows: 1   Depth: 512   Position: 0

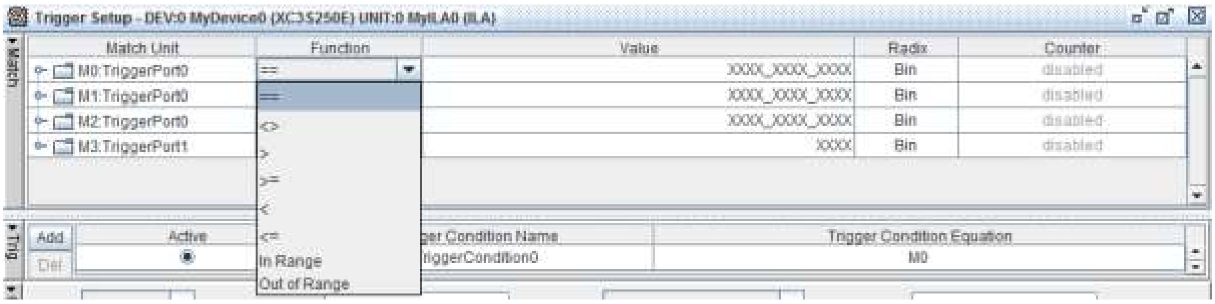
Storage Qualification: All Data

```
INFO: PLD version = 0012h
INFO: Type = 0X0004
INFO: ESN option: 0068152EA24591
INFO: Successfully opened Xilinx Platform USB Cable
INFO: Cable: Platform Cable USB, Port: USB21, Speed: 1.5 MHz
INFO: Found 1 Core Unit in the JTAG device Chain.
```

All the match units are initially set to don’t care.

Trigger setup

For the **match units**, you can set up a trigger conditions. The possible settings depend on the match unit type. *Don't care bits are only allowed, if the function is equal (==) or not equal (<>),* in all other cases, all bits of a match unit must be defined!

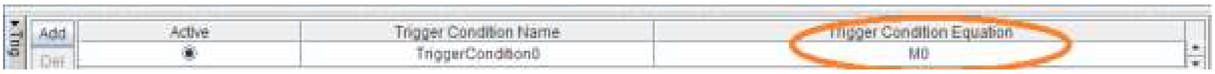


Trigger Setup - DEV:0 MyDevice0 (XC3S250E) UNIT:0 MyILA0 (ILA)

Match Unit	Function	Value	Radix	Counter
M0:TriggerPort0	==	XXXX_XXXX_XXXX	Bin	disabled
M1:TriggerPort0		XXXX_XXXX_XXXX	Bin	disabled
M2:TriggerPort0		XXXX_XXXX_XXXX	Bin	disabled
M3:TriggerPort1		XXXX	Bin	disabled

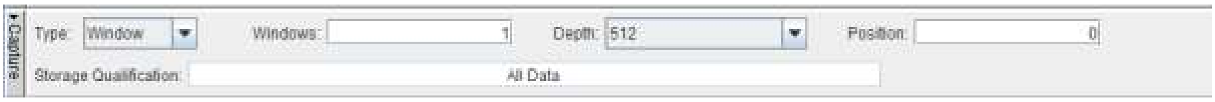
Active	Trigger Condition Name	Trigger Condition Equation
●	TriggerCondition0	M0

The **trigger condition** can be defined as a logical function or a sequence of match unit conditions. Click on 'Trigger condition equation' to set it!



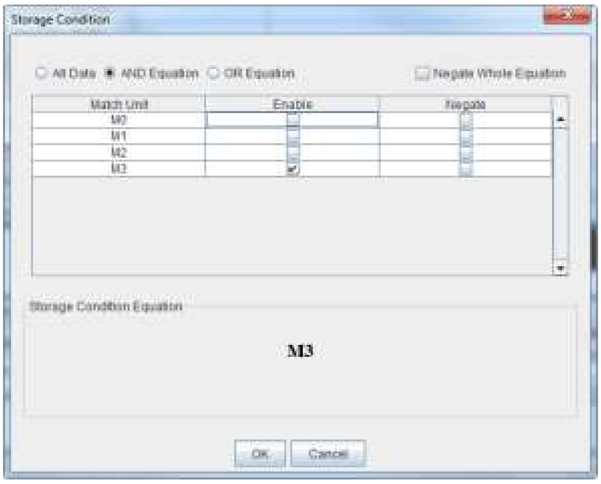
Active	Trigger Condition Name	Trigger Condition Equation
●	TriggerCondition0	M0

The sampling can be set up using the **capture** panel. In window mode you define, how many trigger conditions should be stored in the state memory during the session. In N samples mode, the number of words to stored after one trigger is defined. Setting the (trigger) position to grater than 0, the samples before the trigger event can be examined.



Type: Window
Windows: 1
Depth: 512
Position: 0
Storage Qualification: All Data

You can assign the logical combination of match units not only to a trigger condition, but also to a storage qualification enable signal. E.g. you can configure the ChipScope to sample the data only if M3 match unit condition is fulfilled.



Storage Condition

○ All Data ● AND Equation ○ OR Equation □ Negate Whole Equation

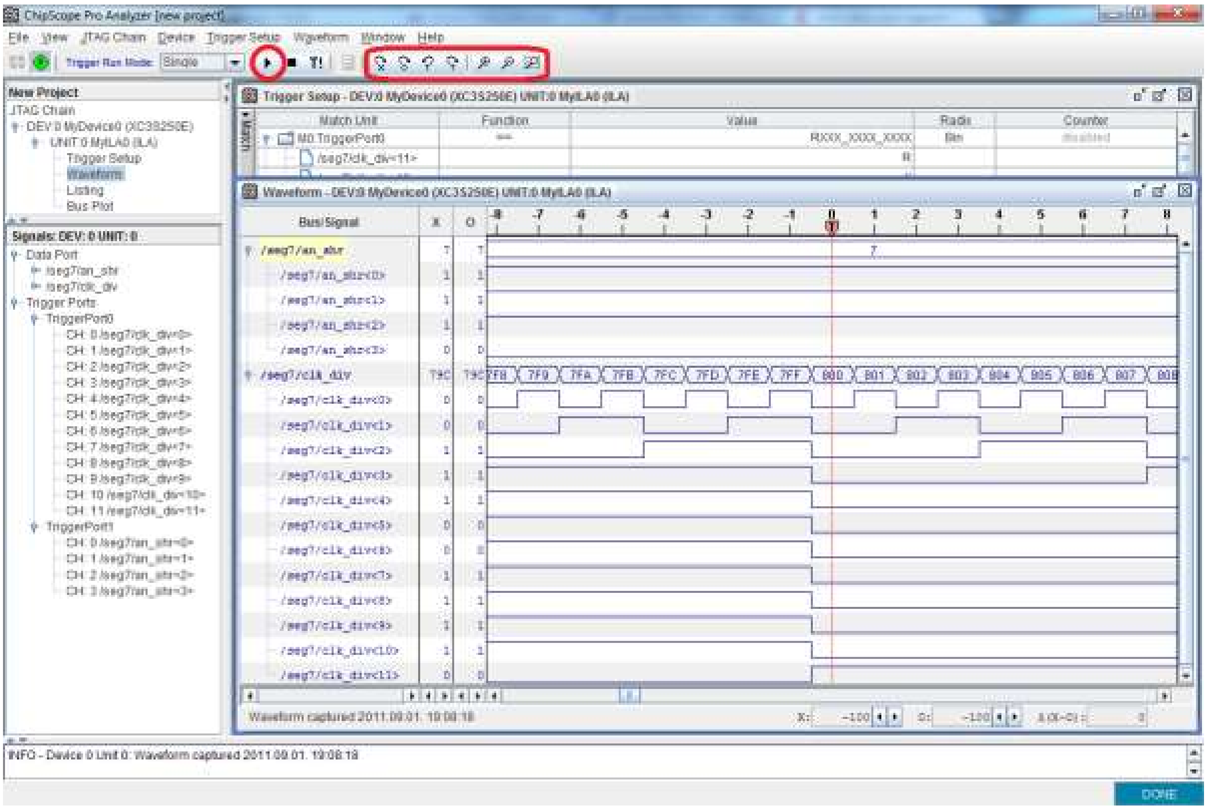
Match Unit	Enable	Negate
M0	□	□
M1	□	□
M2	□	□
M3	☑	□

Storage Condition Equation

M3

Analysis

After the apropriate setup, the analyisis itself can be started using the **toolbar of the ChipScope**. In 'Single' trigger run mode after pressing the 'Play' button, the analyzer waits for trigger, after the trigger it fills the state memory and if it is full, the result can be examined in the **waveform window**. You can stop the analyzer any time, but the waveform is refreshed only if the the state memory is totally filled. (Thus, for „sorter” waveforms set depth in capture panel!)



INFO - Device 0 Unit 0: Waveform captured 2011.09.01. 19:08:18

Using the „T!” button, sampling can be started immediately (the trigger condition is not considered.). Selecting the ’Repetitive’ trigger run mode, the analyzer waits automatically for the next trigger event after the state memory is fulfilled.

Reference

The detailed documentation of the ChipScope can be downloaded from the Xilinx webpage.
http://www.xilinx.com/support/documentation/dt_chipscopepro_chipscope13-2.htm