# ChipScope

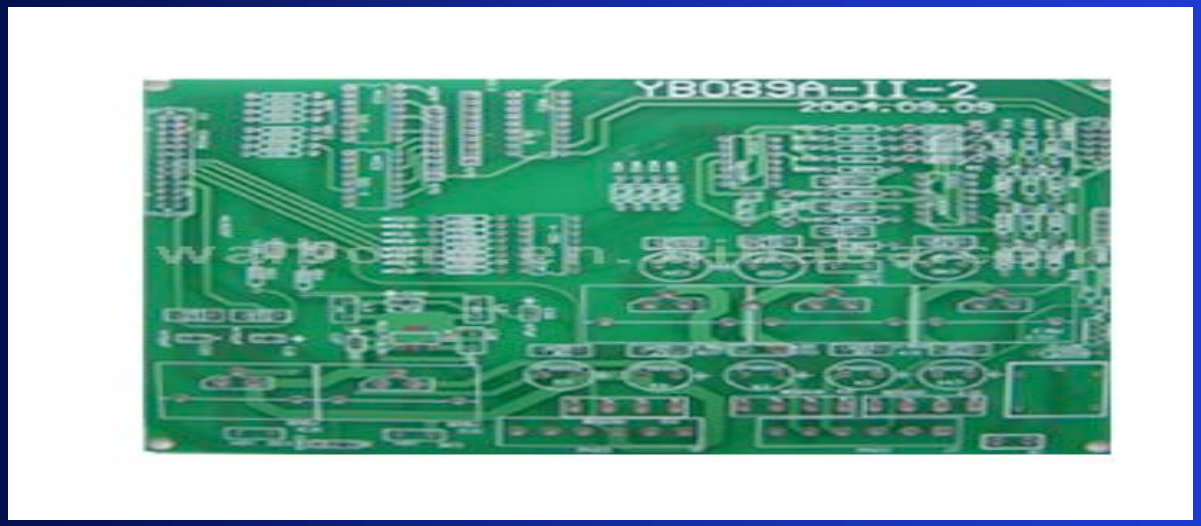
"Looking" At Signals
Inside of the FPGA

By Jason Agron

# Debugging

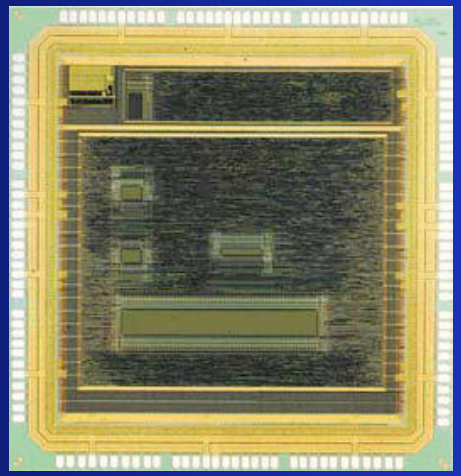
- For SW, this is a familiar process:
  - Add in “print” statements:
    - Monitor flow through code.
    - Monitor values of important variables.
  - Use a debugger:
    - Monitor low-level CPU behavior.
    - Monitor memory access.
- How does one debug hardware?
  - Additionally, how does one debug SW and HW at the same time?

# Monitoring Signals




- Theoretically, all signals (wires) on a traditional circuit board can be monitored.
  - Use a probe, an oscilloscope, or a logic analyzer.
- But what about the internal state of devices?
  - Registers, busses, etc.

# Systems-On-Chip (SoC)



- Completely implemented in a single piece of silicon.
  - Therefore not all important signals are “exposed”.
- FPGAs:
  - Same problem - lots of internal logic, but not a lot of I/O ports.
- This is where ChipScope comes in handy.

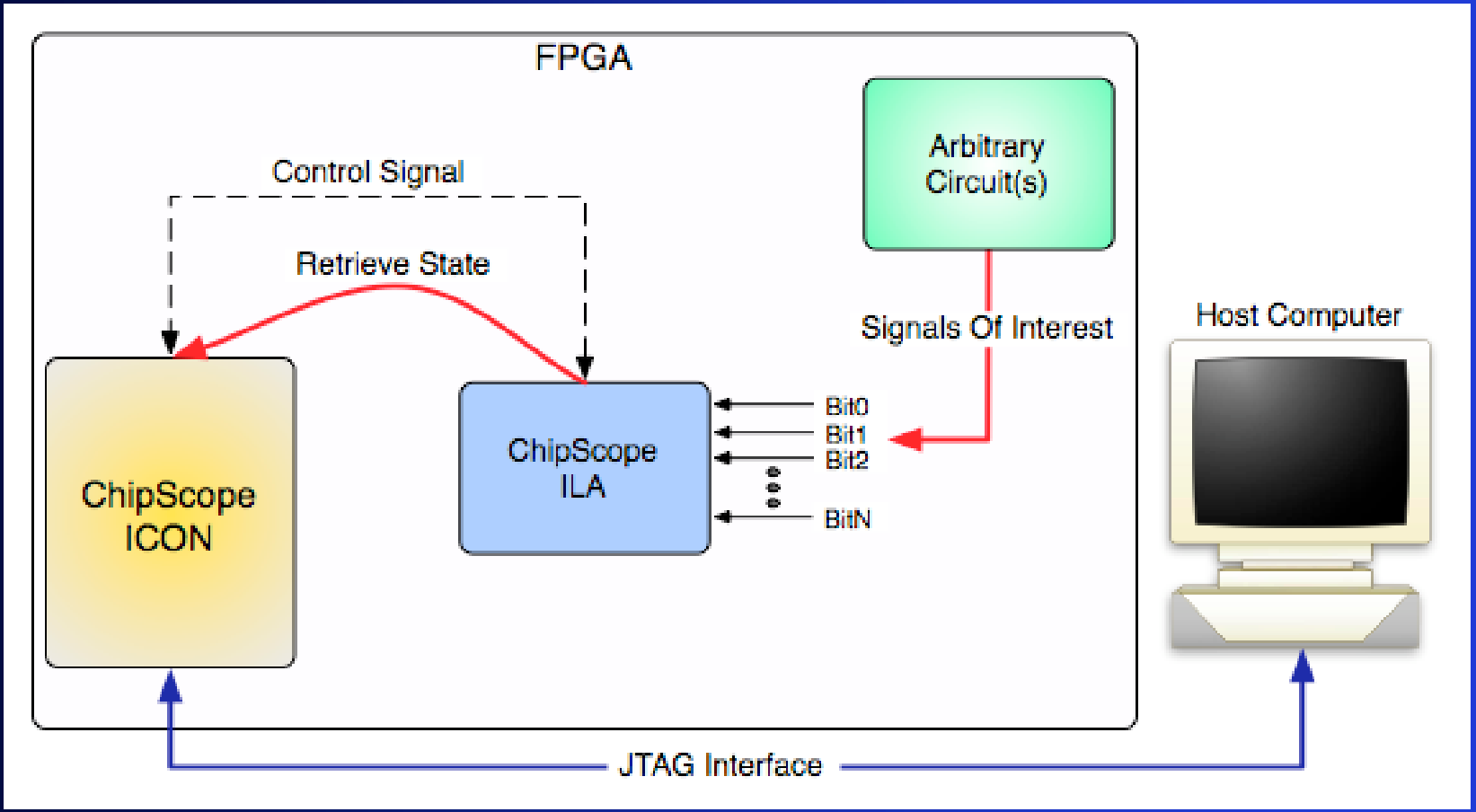
# ChipScope Overview

- A useful tool for monitoring arbitrary on-chip signals in real-time.
- Has two main parts:
  - Embeddable IP cores that capture and store values of signals within the FPGA.
    - Like an “embedded” logic analyzer.
  - Software tool that allows one to “read” the captured data and visualize it on a host computer.
    - Like the “screen” and “control panel” of a logic analyzer.

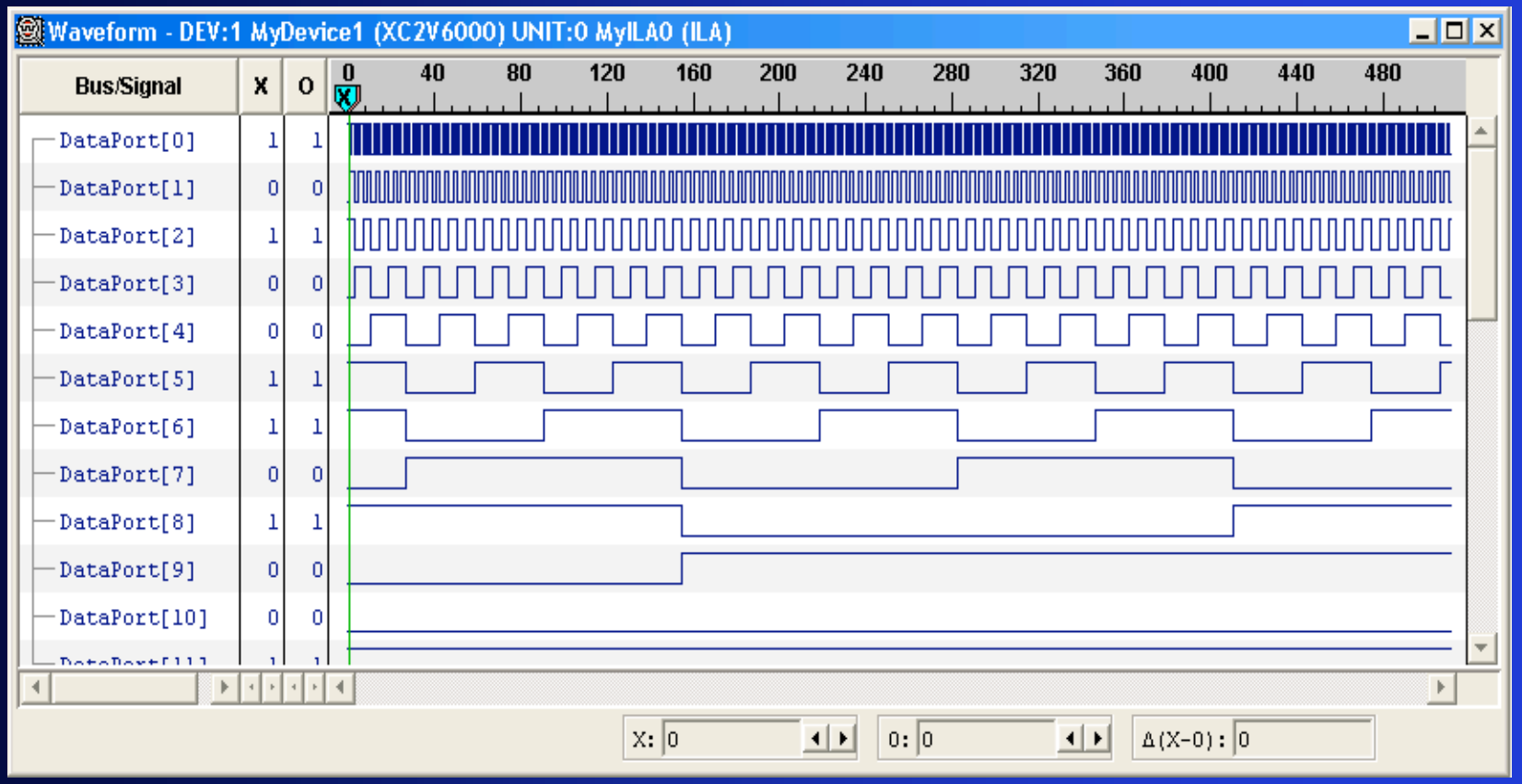
# How It Works

- The designer connects signals of interest to a ChipScope ILA core.
  - ILA = Integrated Logic Analyzer.
- The designer connects the ILA to a ChipScope ICON core.
  - Allows a host computer to access the data stored in the ILA.
  - An ICON core is needed in ALL ChipScope designs!!!
- Test the system and use the ChipScope Analyzer tool to monitor the hardware signals connected to the ILA.
  - Setup trigger conditions.
    - i.e. “Begin to capture data when input3 goes high”.
  - Plot captured data.

# ChipScope System Setup



# ChipScope Analyzer



# Example: ChipScope IBA

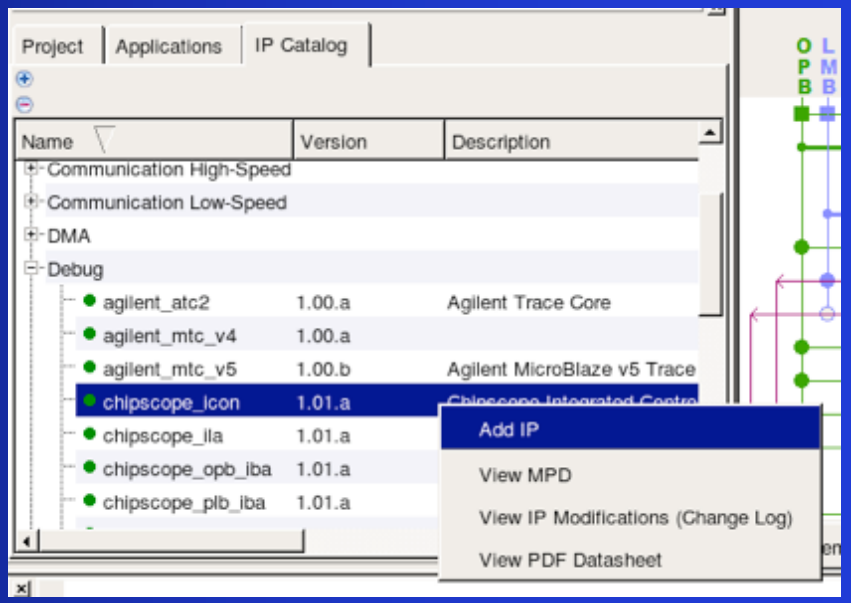
- IBA = Integrated Bus Analyzer.
- A “special” version of an ILA tailored specifically for PLB and OPB CoreConnect busses.
- Allows one to passively monitor all bus traffic.
  - Address and data lines.
  - Bus control lines.
- Easy to use…

# IBA: General Instructions

- Instantiate ChipScope cores:
  - Instantiate an ICON core.
  - Instantiate an OPB IBA core.
  - NOTE - all ChipScope cores can be found in the IP catalog under “Debug”.
- Connect ChipScope cores to the “system”:
  - Connect the IBA core to the bus of interest.
  - Connect a control line from the ICON core to the IBA core.
- Configure the settings of the IBA/ILA.
  - Which signals to monitor, buffer-depth, etc.

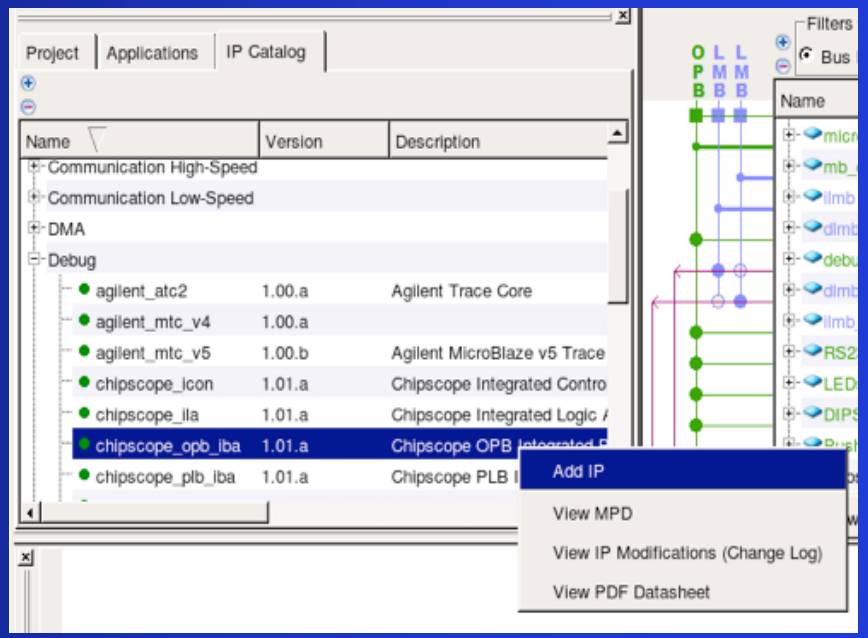
# Adding the ICON Core

- Open up a system to add ChipScope capabilities to…
- Go to the IP Catalog.
- Expand options for “Debug”.
- Right-click on ICON.
- Click on “Add IP”.

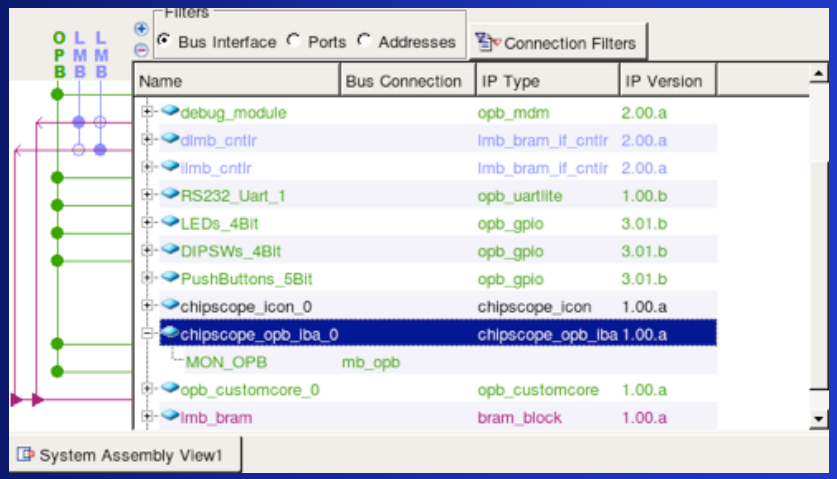


# Adding the IBA Core

- Go to the IP Catalog.
- Expand options for “Debug”.
- Right-click on OPB-IBA.
- Click on “Add IP”.

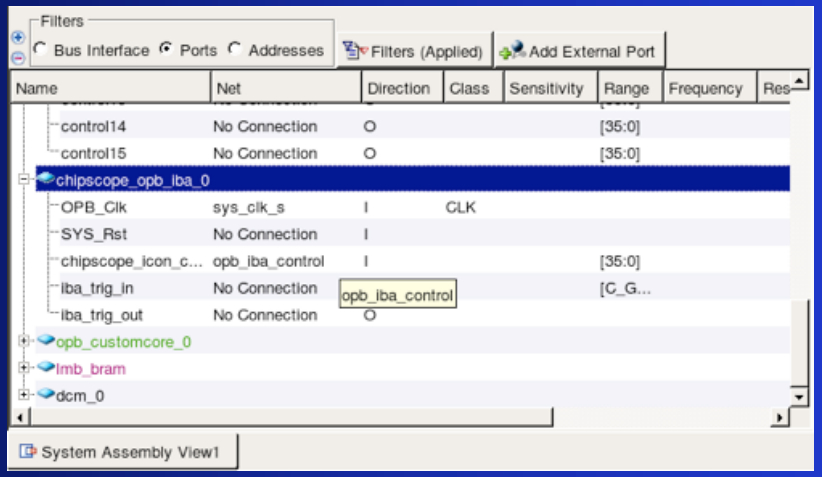


# Connect OPB-IBA to the OPB



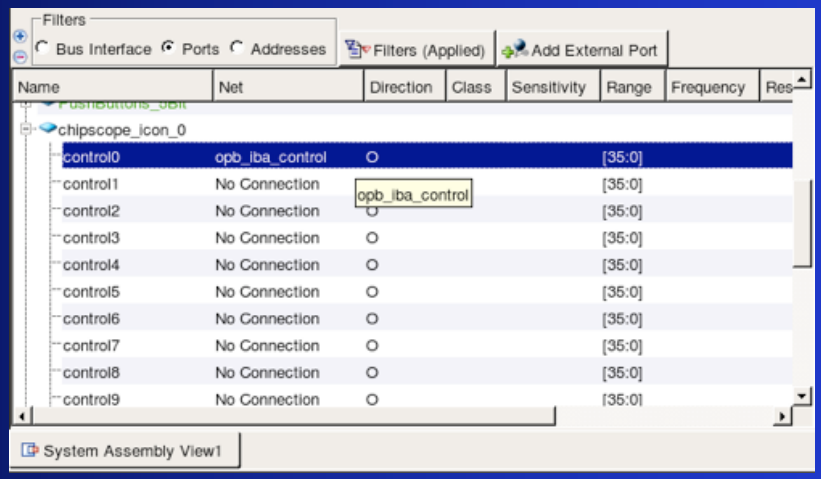

- Go to the System Assembly View.
  - Then go to the “Bus Interface” view.
- Expand options for the OPB-IBA.
- Now click on it’s bus interface circle to connect it to the OPB bus.
  - The circle should go from hollow to shaded.

# Connect a Control Line to IBA



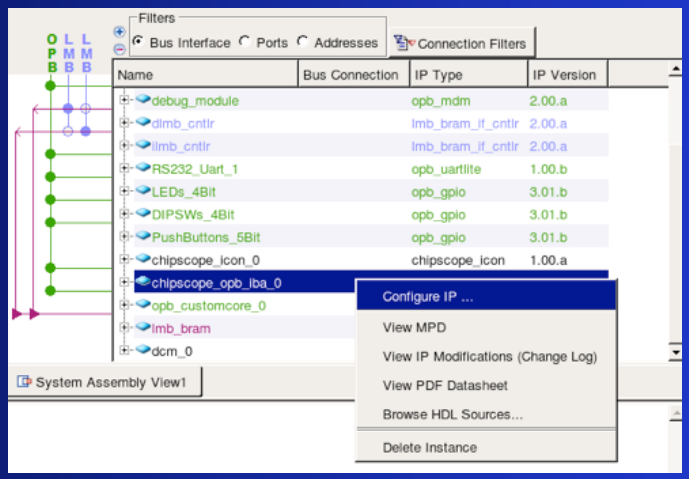

- Go to the System Assembly View.
  - Then go to the "Ports" view.
- Expand options for the OPB-IBA.
- Now connect a signal to it's icon control line…
  - I called my signal"opb_iba_control".
- Now we must connect the other "end" to the ICON core.

# Connect Control Line to ICON



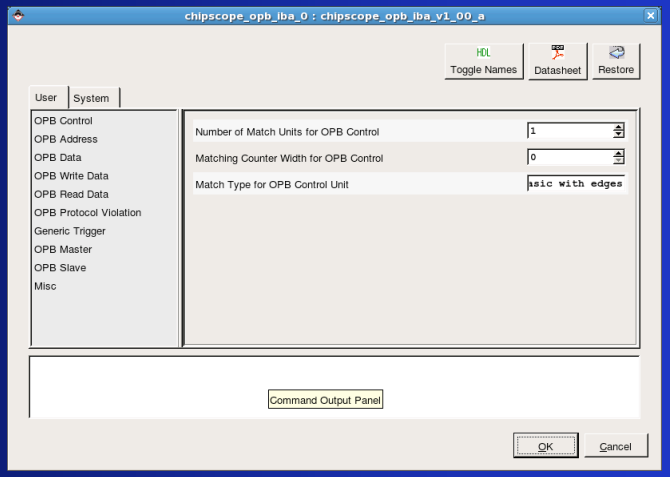

- Go to the System Assembly View.
  - Then go to the "Ports" view.
- Expand options for the OPB-ICON.
- Now connect the previously created control signal to it's control0 line…
- Now a control line is connecting the IBA and the ICON cores.

# Configuring the IBA Core




- Go to the System Assembly View.
  - Then go to the "Bus Interface" view.
- Right-click on the OPB-IBA core.
- Click on "Configure IP" to open up this cores configuration GUI.

# Configuring the IBA Core




- This GUI allows one to graphically configure the core's "generics".
  - Generics are compile-time parameters used to adjust the functionality of IP cores written in VHDL/Verilog.
- You can adjust what types of signals are monitored, buffer depth, etc. from here.

# The Results

- EDK provides a GUI interface to build systems for platform FPGAs.
- The .mhs file is a textual representation of the components and connections that compose an EDK system.
- The ChipScope core instantiations should look like this in the .mhs…

```
BEGIN chipscope_icon
 PARAMETER INSTANCE = chipscope_icon_0
 PARAMETER HW_VER = 1.00.a
# Needed if opb_mdm is also present
 PARAMETER C_SYSTEM_CONTAINS_MDM = 1
 PARAMETER C_NUM_CONTROL_PORTS = 1
 PORT control0 = opb_iba_control
END

BEGIN chipscope_opb_iba
 PARAMETER INSTANCE = chipscope_opb_iba_0
 PARAMETER HW_VER = 1.00.a
 PARAMETER C_DATA_UNITS = 1
 PARAMETER C_CONTROL_UNITS = 1
 PARAMETER C_WRDATA_UNITS = 1
 PARAMETER C_RDDATA_UNITS = 1
 PARAMETER C_ADDR_UNITS = 1
 BUS_INTERFACE MON_OPB = mb_opb
 PORT OPB_Clk = sys_clk_s
 PORT chipscope_icon_control = opb_iba_control
END
```

# Using ChipScope

- The ChipScope Analyzer is the tool that you will actually use.
- It provides an interface…
  - To configure the ILA core.
    - Triggering setup.
  - To view captured data.
- This tools uses the JTAG interface to “talk” to the ICON core.
  - *Where is the XUP’s JTAG interface?*

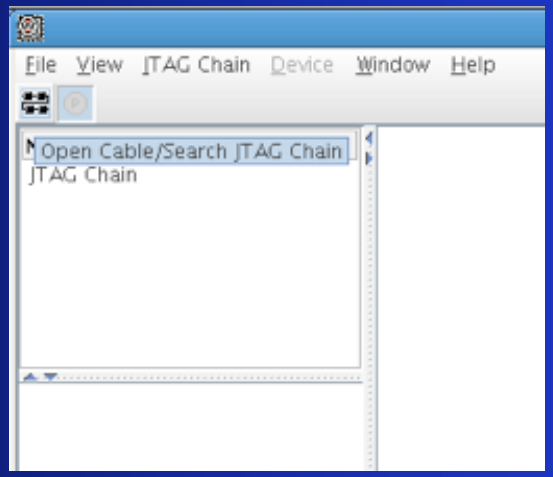
# Using ChipScope Analyzer

- First, download the ChipScope-enabled bitstream to the FPGA.
- Make sure to be running a program that generates bus traffic on the OPB.
- Now launch the analyzer application…
  - We will now use the analyzer to look at the bus traffic as well as the OPB protocol.

# Setting Up Analyzer For IBA

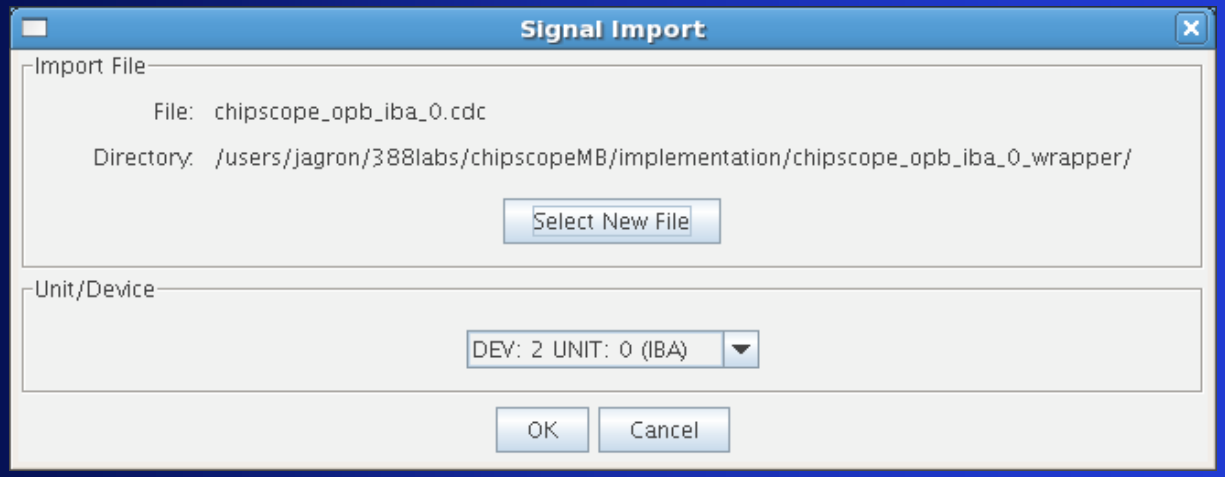
- First, click on the “Open Cable…” icon.
  - This opens up a JTAG connection to the ChipScope cores.
  - Click OK on any pop up dialogs.
- Now click on “File”, “Import”.
  - Select the IBA configuration (*.cdc) file from…
    - /implementation/<iba_core>/<iba_core>.cdc
  - This groups and names all of the bus signals.
- Now we can begin analyzing signals by using the trigger buttons…

# Establish JTAG Connection




- First, establish a JTAG connection between host and FPGA.
  - Click on icon with black boxes in it.
- Click OK on subsequent dialog boxes.
- Now, Analyzer is connected to the ChipScope cores.

# Import Configuration File

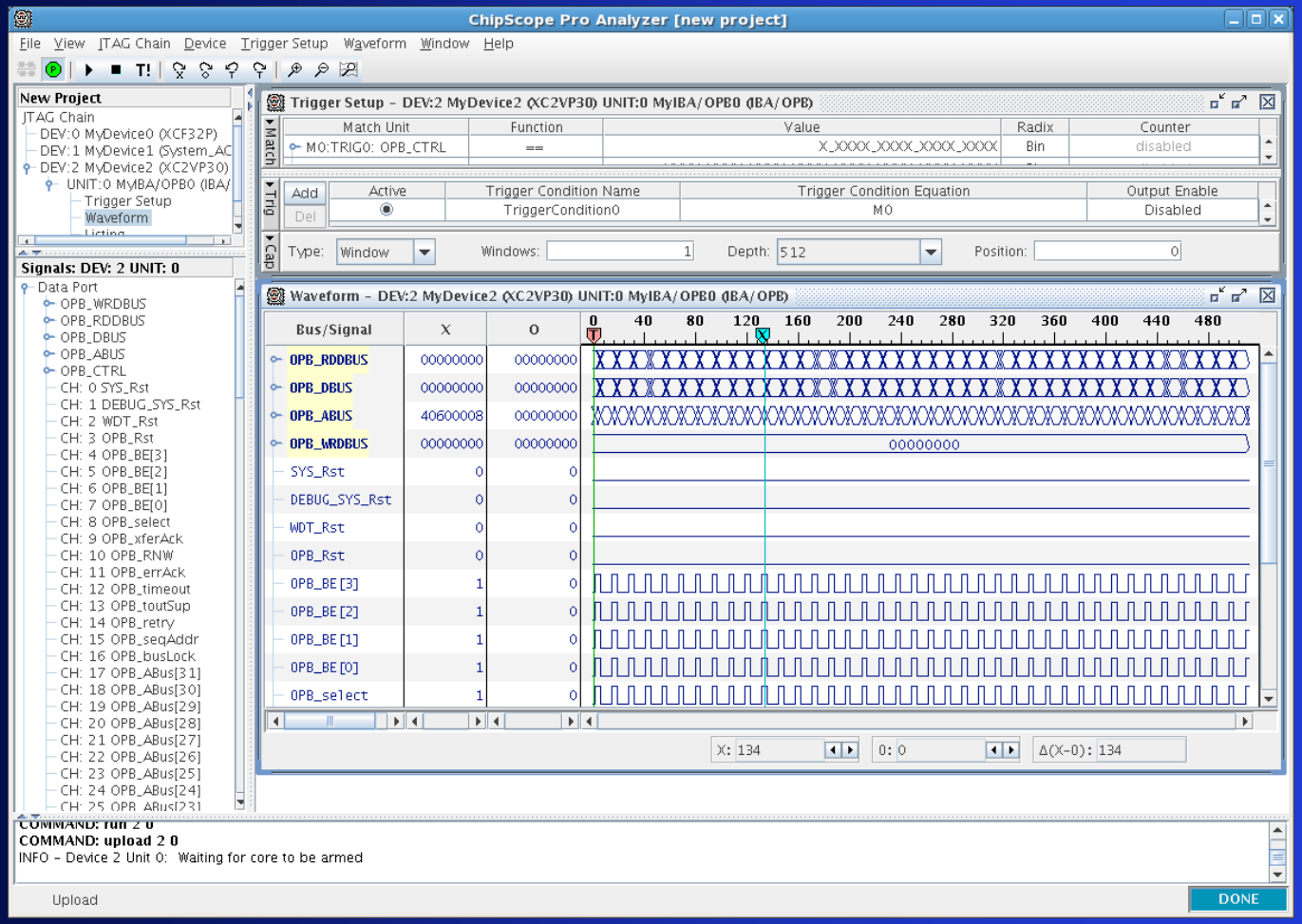



- The Xilinx toolset automatically produces a .cdc configuration file for IBA cores.
  - This file contains all of the grouping and naming conventions for the IBA signals.
- Go to “File”, “Import”, and select the .cdc file from…
  - /implementation/<iba_core>/<iba_core>.cdc

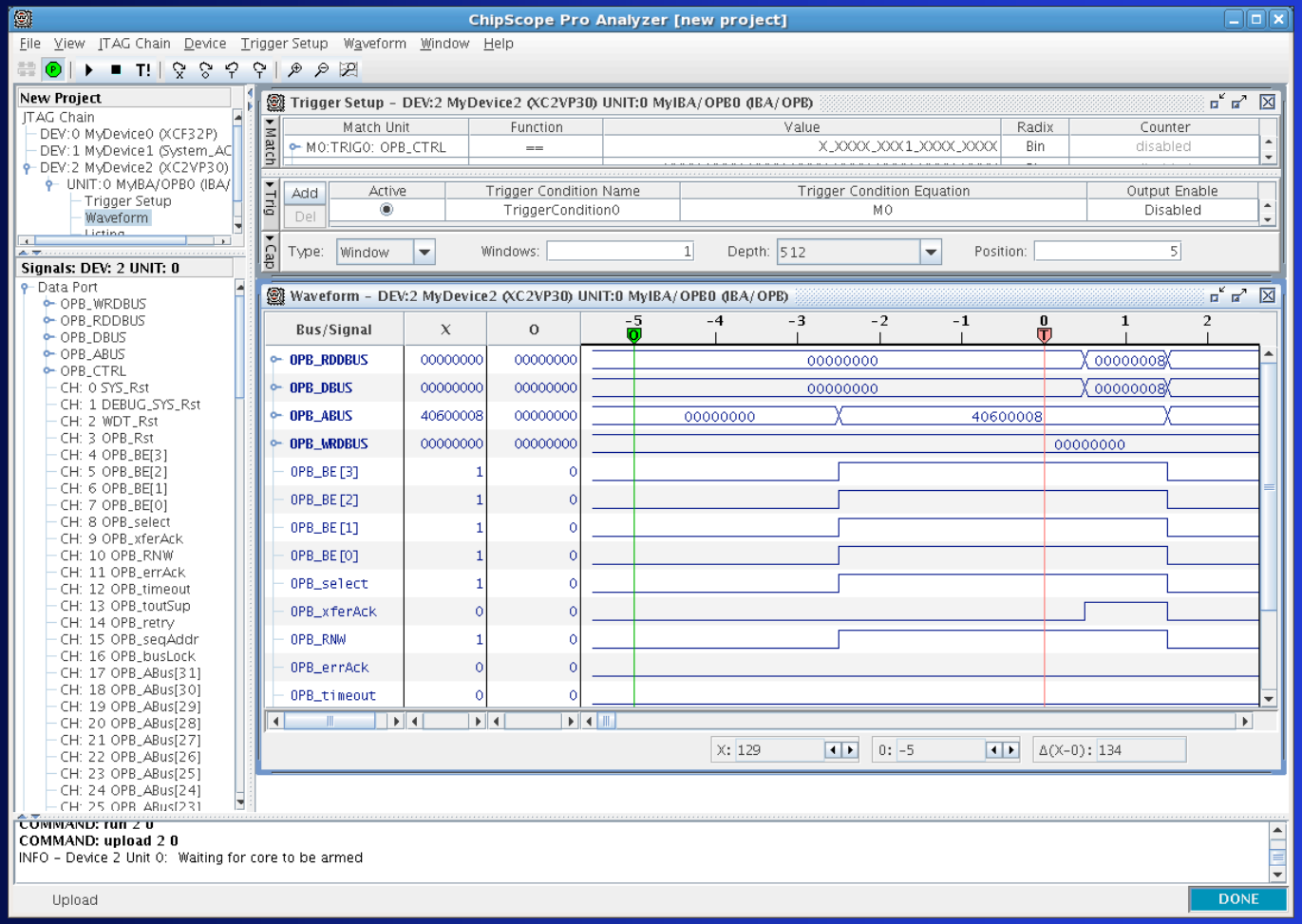
# Triggering Setup

- Now, ChipScope is fully setup for use with the IBA core.
- To capture data immediately, click on the “T!” icon.
- To capture data based on custom trigger conditions…
  - First setup the proper Boolean triggering condition.
  - Then click on the sideways triangle icon to arm the trigger.
  - Now, wait for condition to occur.

# Example: Immediate Trigger



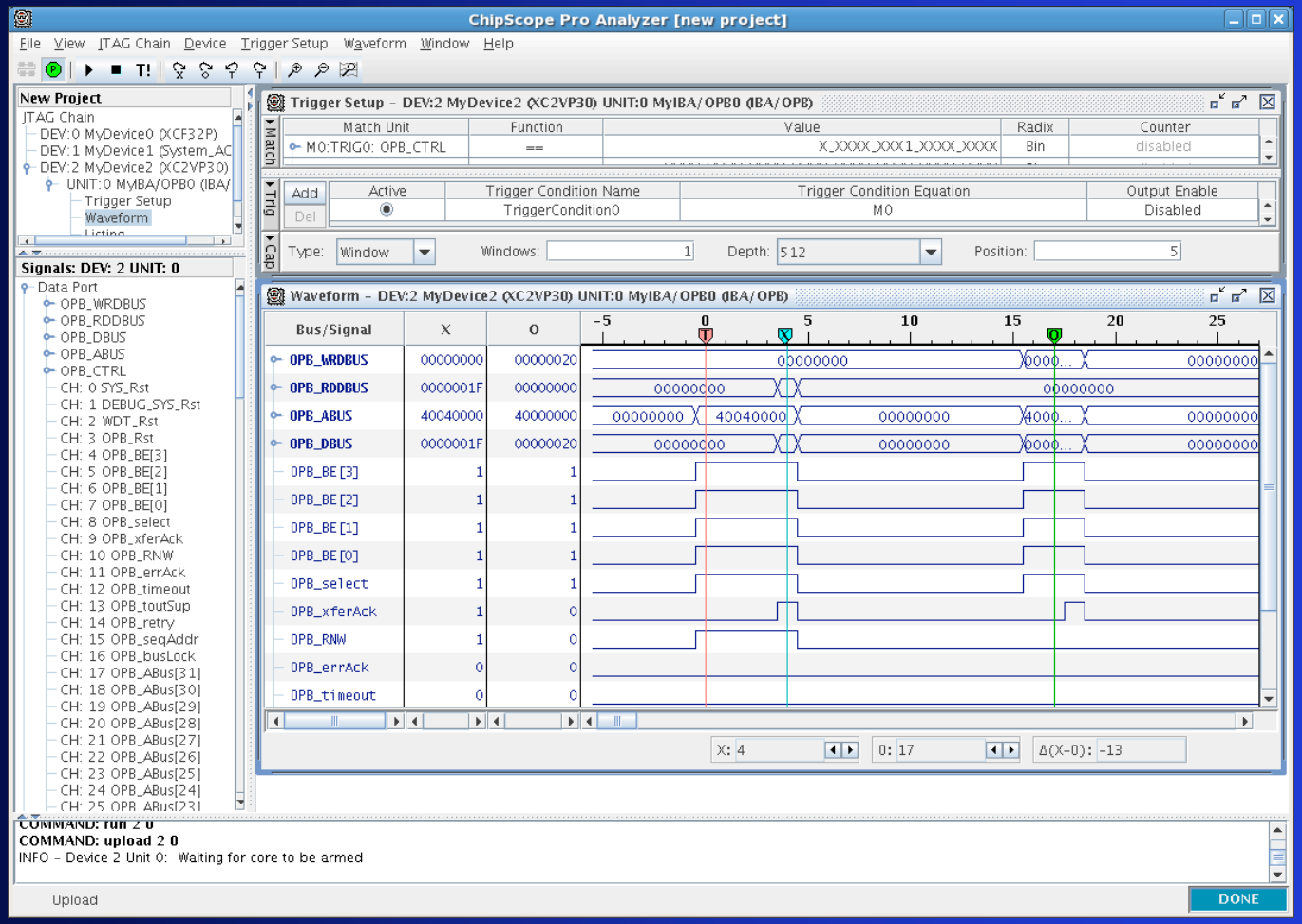
# Example: Custom Trigger (trigger when opb_select = '1')



# Project

- Create a ChipScope-enabled hardware system with IBA.
- Test the setup using a software application.
- Use ChipScope Analyzer to capture the following situation.
  - Read a value from an address.
  - Write the (value + 1) to (address + offset).
- *NOTE - address offsets of are meant to be word-aligned!!!!*

# Example Results



# ChipScope ILA Setup

- ILA setup is the same as for IBA except…
  - ILA can monitor arbitrary signals.
  - One must connect up each signal that needs to be monitored to the ILA unit.
- For further help…
  - ChipScope Online Documentation.
  - EECS 388 Wiki Documentation.
  - TA Assistance.