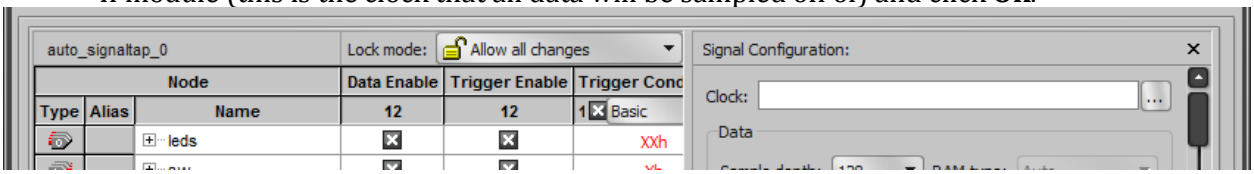
# SIGNALTAP II TUTORIAL

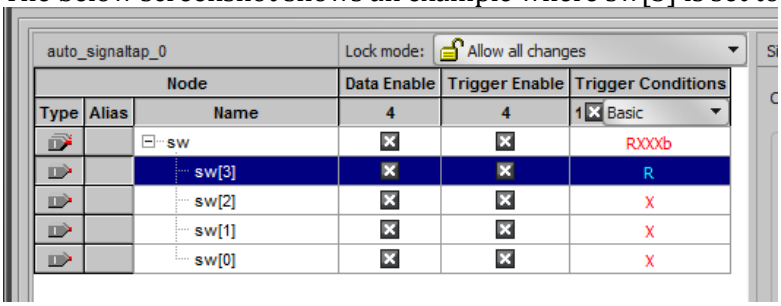
E155: Microprocessors
Matthew Watkins

## Setting up SignalTap II

1. Open or create your project file in Quartus II. If not already done, create the base design, build, and assign pins.
2. If you are using the Web Edition of Quartus II, you will need to enable the Talk Back feature. Select **Tools→Options…**, go to the *Internet Connectivity* page. Select **TalkBack Options…** and check **Enable sending TalkBack data to Altera**. Close Quartus II and reopen so that this change will be recongnized.
3. Add SignalTap II file. **File→New** and select *SignalTap II Logic Analyzer File*.
4. When the SignalTap II window opens up, save the SignalTap II file (**File→Save As…**)
5. Click **Ok** regarding Input "Data and Trigger" is empty.
6. Click **Yes** to add file to current project.
7. Click **Yes** to enable SignalTap II File for current project
    a. To disable SignalTap at a later time, go to **Assignments→Settings**, select *SignalTap II Logic Analyzer* and uncheck enable.
8. Add nodes of interest to project. In the *Setup* tab double-click where it says *Double-click to add nodes*. Under filter select *SignalTap II: pre-synthesis* (if the signal is not shown, you might also look under *SignalTap II: post-fitting*) and click **List**. Once you have added the desired nodes, click **Ok**.
9. Back in the main SignalTap II window, click the **…** button next to *Clock* in the *Signal Configuration:* section. Select the signal that you want to serve as the clock for the SignalTap II module (this is the clock that all data will be sampled off of) and click **Ok**.




10. For simple triggering (more complex triggering will be discussed later), in the *Setup* tab make sure that *Basic* is showing under the *Trigger Conditions* column. Right-click on a particular signal and select the condition specifying when you'd like to trigger the sampling. The below screenshot shows an example where sw[3] is set to trigger on a rising edge.




11. At this point make sure that your board is powered on and connected to the USB Blaster via the JTAG port. Under the *Hardware* section, click **Setup…** and select **USB-Blaster** under *Currently selected hardware:*.
12. Back in the main Quartus II window compile and program your design. (**Processing→Start Compilation** followed by **Tools→Programmer** and program your board.)

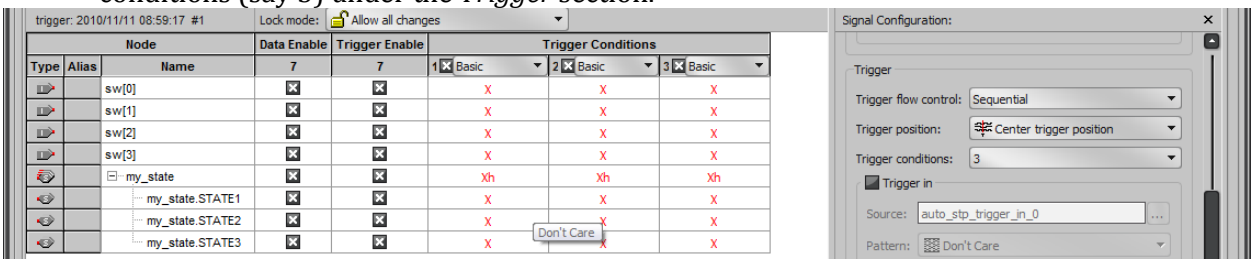
## Using SignalTap II

1. After following the above steps and programming the board, switch back to the SignalTap II Window.
2. Select **Processing→Run Analysis** (F5).
3. Select the *Data* tab to see the available signals. The *Status* column of the Instance pane should say "Waiting for trigger."
4. Perform the triggering event (such as toggling sw[3] from low to high). The data window should now display the data levels for all listed signals.
5. To repeat this, simply select **Run Analysis** again. Alternatively, you can select **Tools→Autorun Analysis** (F6) to have the analysis immediately resume after a triggering event. To stop the analysis click on the stop icon (**Stop Analysis** or <Esc>).

## Multiple Trigger Levels

You can setup SignalTap to trigger on a multi-event sequence. For example, sw[0] going high followed by sw[1] going low.

1. Click the *Setup* tab.
2. In the *Signal Configuration* section, set *Trigger Conditions* to the desired number of conditions (say 3) under the *Trigger* section.

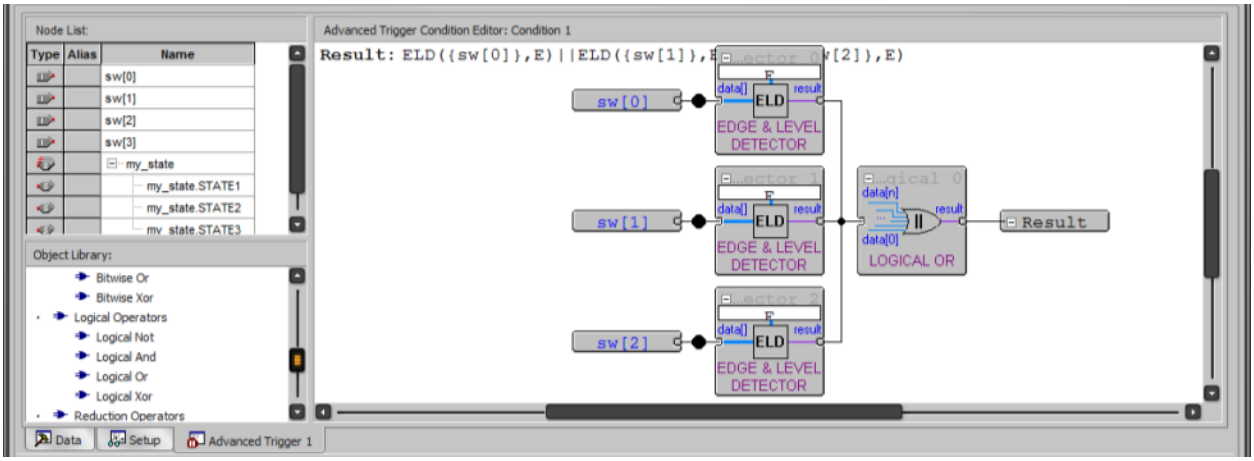



3. The *Setup* tab should now show multiple columns under Trigger Conditions. Setup the triggers as desired.
4. Recompile the design and load.
5. Back in the *SignalTap II* window, select the *Data* tab and then **Run Analysis**. It will now wait for the prescribed sequence of events before displaying results.

## Advanced Triggering

You can create even more complex triggering options (such as some logical combination of signals) using the advanced triggering option.

1. Click the *Setup* tab.
2. If not already done, set *Trigger Conditions* under *Signal Configuration* to 1.
3. In the *Setup* tab, change *Trigger Conditions* to ***Advanced***.
4. This will bring up the graphical Advanced Trigger tab. This allows you to graphically create the triggering condition that you desire. You can add signals from the design by dragging signals from the *Node List:* into the main editing window. Drag the desired signals into the editing window.
5. The *Object Library* pane includes a variety of operations that can be used to condition the signals, such as comparisons, logical operations, and reductions. Select the desired components and drag into the window.
6. Connect the inputs and result to the operations. The figure below shows an example of creating a triggering event that occurs when any of three signals sees an edge.

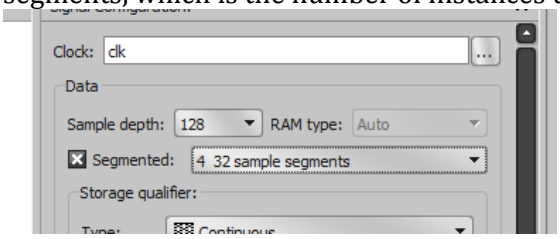



7. Recompile the design and program. Run as you did in the previous examples.

## Segmented Buffer Mode

The segmented buffer mode allows you to subdivide the single buffer into multiple separate buffers to capture multiple instances of the same triggering event.

1. In the *Signal Configuration* pane check the **Segmented** box. Select the desired number of segments, which is the number of instances to capture.




2. Recompile and program.
3. When you **Run Analysis** this time, the results will appear after the signaling event occurs the specified number of times. The results will be shown one after the other in the *Data* tab.

## Other notes

- You have three options on how much data is shown before and after the triggering event. The options are to 1) mostly show data before an event, 2) mostly show the data after an event, or 3) show an even split of data before and after an event. This can be set under *Trigger position* under the *Signal Configuration:* pane.
- You can change the number of collected samples by changing *Sample depth:* within the *Signal Configuration* pane. Be aware that the more samples collected the more resources that will be required on the FPGA. The size of the FPGA and base design may limit the buffer size you can use.
- SignalTap II can automatically recognize mnemonics associated with FSMs. Right click and select **Add State Machine Nodes…**.
- If signals are being optimized out of your design, you can place keywords to prevent optimization of certain signals. Realize that this may increase resource utilization or decrease speed.
    - `keep` – for combinational signals (ex. `(*keep*) wire foo;`)
    - `preserve` – for registers (ex. `(*preserve*) reg bar;`)