**Dimas Dimitrios Report**

**(IAPP 1st secondment to Pisa¸ months of June and July 2013)**



Contents

[Scope of the Report 3](#_Toc363580298)

[Technical Description 3](#_Toc363580299)

[Problems Arisen 5](#_Toc363580300)

[Solution Suggestions 7](#_Toc363580301)

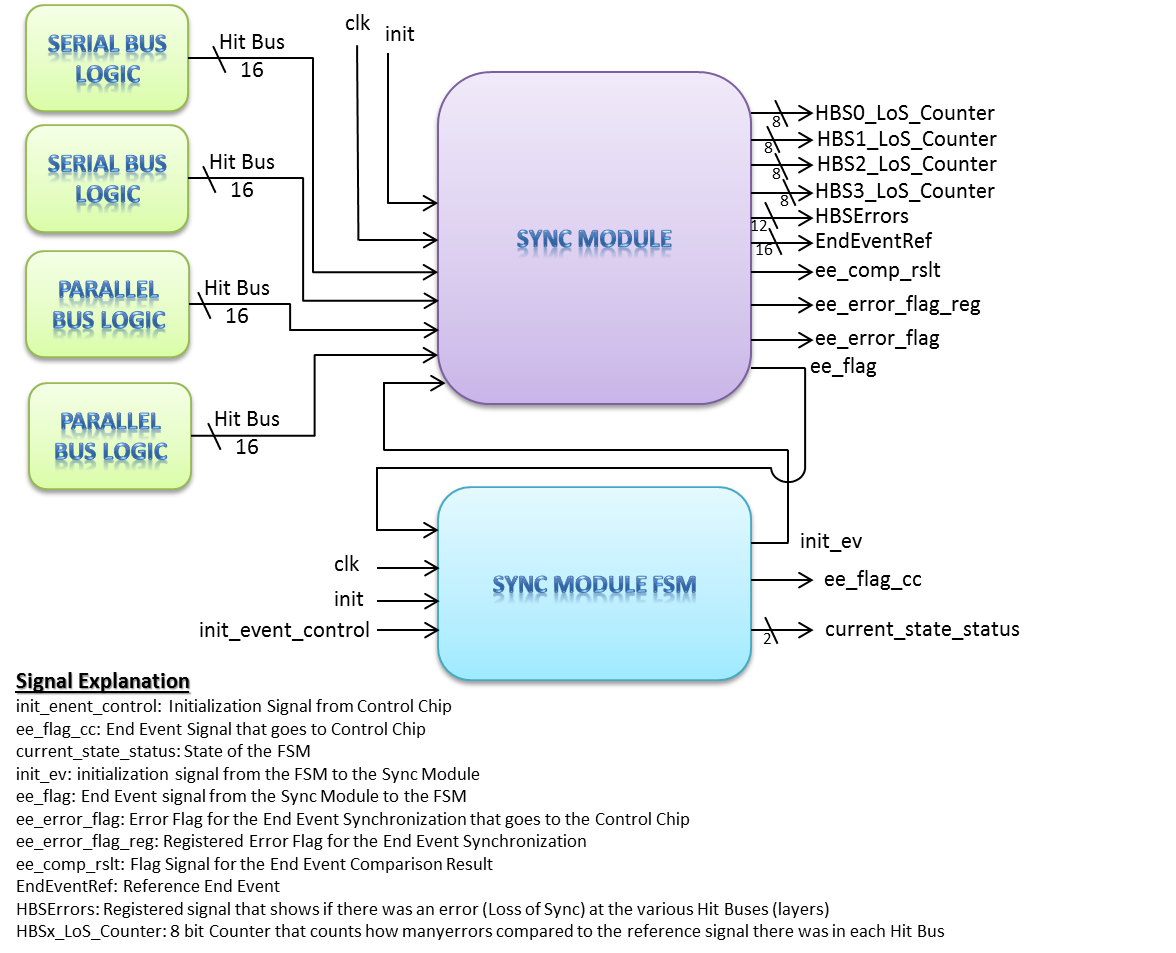
[Files Provided 7](#_Toc363580302)

# Scope of the Report

Scope of this report is to report the work that was done by Dimas Dimitrios, during his stay at Pisa, Italy and specifically at the premises of University of Pisa for the months of June and July 2013.

The work regards a module for detection of the Loss of Synchronization at the SCT\_Input FPGA. The module takes the 4 inputs of the SCT\_Input FPGA and proceeds to a comparison of these signals, specifically of the End Event Words and detects, if existing, the Loss of Syncronization.

# Technical Description



On the scheme above, you may see the two modules that were designed, the sync module and the FSM that controls it. The module is designed to accept 12 input 16-bit signals, not just 4, for the future case of integration of the SCT\_INPUT FPGA and PIXEL\_HIT FPGA to one FPGA[[1]](#footnote-1). The various signals are explained below the scheme, but are also explained thoroughly, as well as all of the logic designed, at the respective vhd files.

The sync module produces a 12-bit registered signal named HBSErrors. Each bit represents one layer (one Hit Bus). When a bit is '1' means that during a test there was a Loss of Sync at the Hit Bus. Moreover this signal concatenated with the Reference End Event, which for now is Hit Bus 0, represent a Read Only Register named LoSError, so it can be easily read with the help of VME commands.

The sync module also produces twelve 8-bit registers, one for each Hit Bus, that count how many Losses of Sync the bus had during the test compared to the Reference End Event Word. This registers concatenated with the Reference End Event represent twelve Read Only Registers named HitBus'x'SynchroErrorReg, so they can be easily read with the help of VME commands.

The other signals are various information signals that are described to the respective vhd files.

Moreover, all the design that has been deleted in order to adapt the new logic, has remained inside the vhd file as comments.

The following vhd files are changed:

1. FSM vhd file. The 4 states have become 5 and new logic is declared. This is done in order to have one more clock cycle, during send\_init\_ev2 state, for the input signals that come from the Hit Bus MUXs (and respectively from the outputs of the FIFO\_INPUTs) to pop out the End Event Word from the FIFOs. The outputs of the FIFO\_INPUTs must not have the End Event Word, and respectively the input of the sync module, when the FSM that controls the sync module goes to state "wait\_ee", in order to avoid counting twice the Loss of Sync Error that occurred.
2. sync\_module vhd file. This file has replaced the old one in total.
3. ParallelBusLogic vhd file. This file has been slightly changed in order to take from this module, not only the End Event Words, but all the words from the Hit Buses.
4. SerialBusLogic vhd file. This file has been slightly changed in order to take from this module, not only the End Event Words, but all the words from the Hit Bus.
5. SctHitRegisters vhd file. In this file, some new VME registers have been declared in order to store Loss of Sync information, as well as constant addresses have been declared in order to read the respective VME registers.
6. vme\_control vhd file. In this file, the read process of the new VME registers is declared.
7. sct\_input vhd file. This file has changed in order to adopt the new FSM and sync module, the changed ParallelBusLogic and SerialBusLogic modules. Moreover, because the
8. SCT\_INPUT ucf file. This ucf file has changed in order to send two new signals to the Control Chip. Specifically SPARE\_A (2 downto 0) signal pins have become SPARE\_A, EE\_COMP\_RSLT and EE\_ERROR\_FLAG. This change has also been done at the sct\_input file.

All the changes (additions, deletes etc) inside the vhd files abovementioned can be easily seen, except the FSM and sync module vhd files that are changed in total, because they always start with explession: "dimas modification start" and finish with the expression "dimas modification stop". Moreover, as abovementioned, all the signals for the rest 8 inputs are commented and just need to be uncommented when needed.

Lastly, the FSM state signal has become from the old 2-bit to a new 3-bit signal in order to have the new fifth state. However, because the pins are limited at the sct\_input FPGA, only the two LSBs are sent to the Control Chip, so the MSB is not sent to the Control Chip.

## Problems Arisen

The two modules were simulated in a behavioral way and worked properly. Unfortunately in the post place and route simulation sometimes the error counters counted an error twice, so at the end of the test appeared more errors than it should be. This is probably because the time that the ee\_pulse signal, that acts like count enable, is at logic '1' is more than it should be, see signal ee\_pulse\_out at figure 1 for the behavioral simulation and figure 2 for the post PAR simulation, so the counters have enough time to count once again if an error has occurred. This does not happen every time. From the post PAR simulation, I have noticed that in many occasions, see figure 3, the ee\_pulse signal is short enough not to produce a second count. Please watch the changes in the ee\_pulse\_out of sync\_module at the test bench written for these two modules. As you can see, sometimes the signal is at logic '1' more than is needed so the counter counts the error twice, and other times, as much as needed, so the counter counts correctly. Unfortunately, I did not have enough time to find out why this happens.

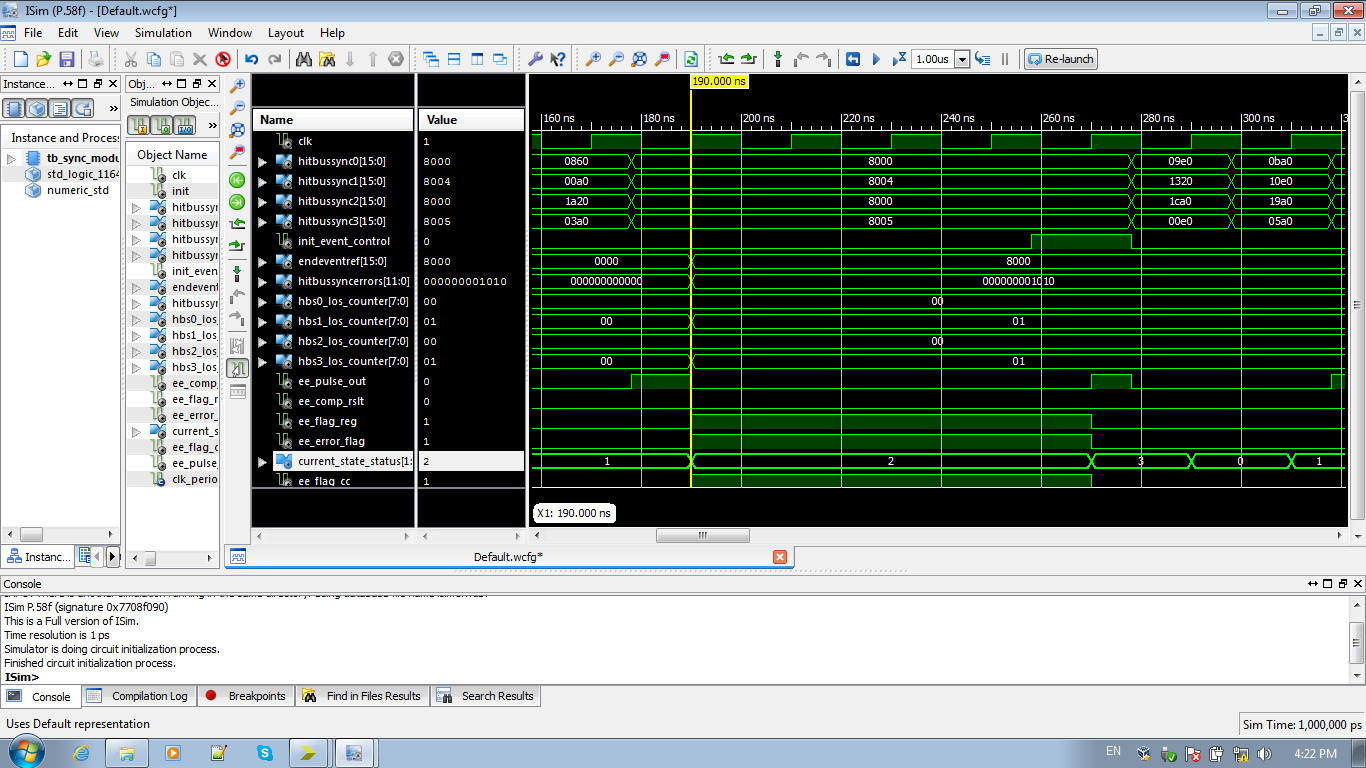
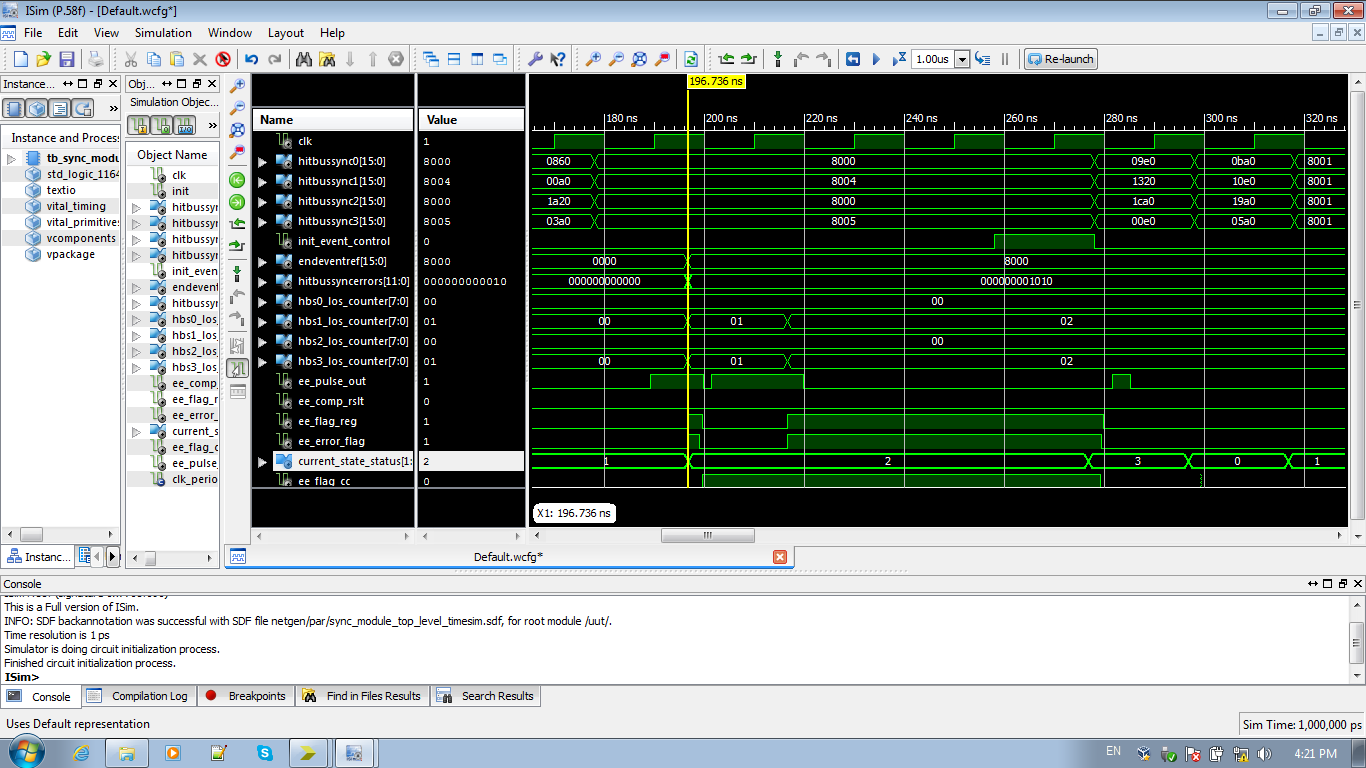
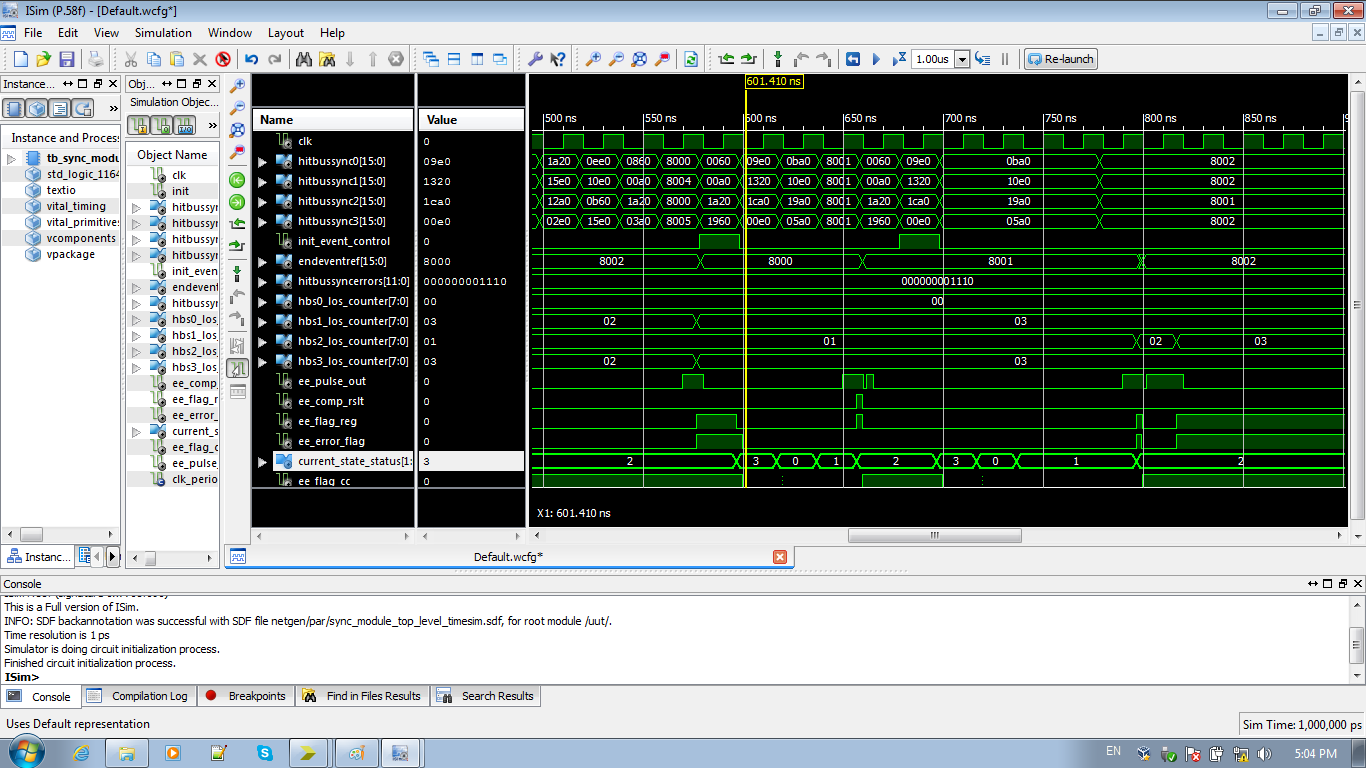


Figure 1: Behavioral Simulation of the two modules



Double count

Figure 2: Post PAR Simulation of the two modules



Right

Double count

Figure 3: Post PAR Simulation of the two modules

## Solution Suggestions

* In order to correct the double count mistake, probably the count enable" ee\_pulse" signal must be changed, either with a new one or by trying to solve the problem of the time that the signal is at logic '1'. By correcting this mistake, the counter will probably count correctly.
* Moreover, maybe it is proper to try a simulation with the original 4 FSM states and see if the data inputs of the sync module (respectively the outputs of the FIFO\_INPUTs) have piped out the End Event Word, so the 5th sate is not needed.

## Files Provided

Two projects are given. The first one, synchronization test, has just the two modules designed, FSM and sync module, as well as the top level and the test bench created for the simulation. The second one, sct\_input, is the SCT\_INPUT project with the new modules and the appropriate changes described above.

1. All the signals for the rest 8 inputs are commented and just need to be uncommented when needed. [↑](#footnote-ref-1)