Reconfigurable module design in Xilinx ISE

1: Introduction

This tutorial explains in steps how to design reconfigurable modules in Xilinx ISE. The purpose is to be able to design reconfigurable modules separate from a base system which can radically decrease development time. Xilinx ISE and Xilinx Planahead are required; versions from the design suite 10.1 are used throughout this tutorial.

2: Design

Launch Xilinx ISE and create a new project through *file->new Project*. Add sources to the project in the *source* window. Synthesize the project by double clicking the synthesis link in the process window. This requires the top level entity to be selected.

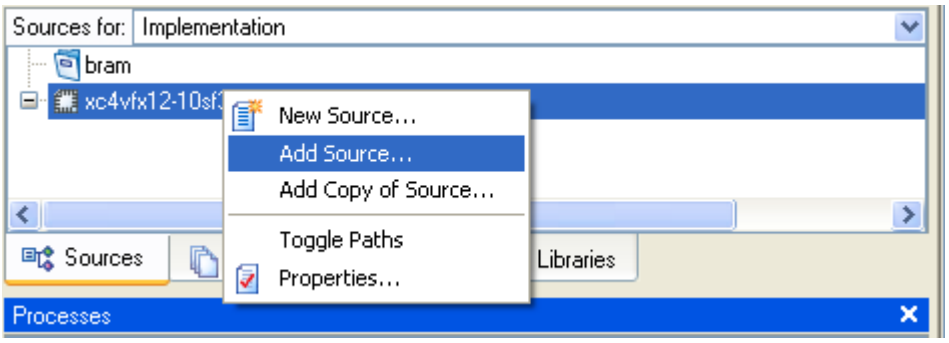

Figure 2.1: Adding sources

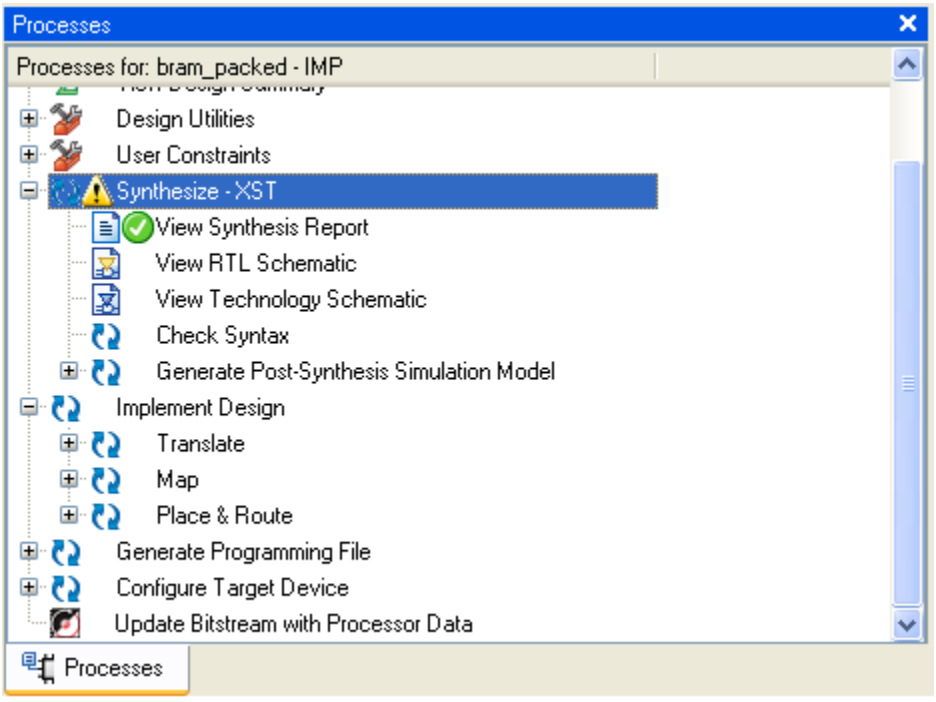

Figure 2.2: Synthesis

Signals in and out of the reconfigurable module are assumed to be piped through bus macros. In order to complete synthesis the bus macro NCD files must be copied into the root directory of the ISE project.

After completing synthesis open Xilinx Planahead. Create a new project and during select import of NGC and EDIF files. Import the NGC file from the current ISE project; it should have the name of the top level entity with a .ngc extension. FPGA target should be specified to a 4vfx12sf363-10 for the Suzaku sz410 platform. Don't mind importing any UCF files during project creation. After creating the project a Planahead workspace should appear showing a list design components and a floorplan.

The design must now be placed in Pblocks to create placement constraints. It important that all parts of the reconfigurable module is isolated on one side of the bus macros whereas the rest of the system must be placed on the other side. This should prevent external signals to cross the bus macro section or the bus macros. Clock signals will not be routed through bus macros so they are exceptions. As bus macro placement is done manually as a final step in placement constraints, place the reconfigurable module and the remaining parts so that virtual bus macros will have good conditions. After completing Pblock assignments export the floorplan to a UCF file, don't mind using EDIF output. After exporting the floorplan to UCF bus macros must be placed with hard UCF code as shown in source 4.1. Use of Planahead and physical constraints are further elaborated in [1].

```
INST "bm_r2l_ctrl" LOC = "SLICE_X30Y126";
INST "bm_r2l_data" LOC = "SLICE_X30Y124";
INST "bm_l2r_ctrl" LOC = "SLICE_X30Y122";
INST "bm_l2r_data" LOC = "SLICE_X30Y120";
```

Source 4.1

Another notice for physical constraints is to prevent use of I/O banks as shown in source 4.2. This might be necessary to avoid external wires crossing the reconfigurable module.

```
CONFIG PROHIBIT= BANK6;
```

Source 4.2

With all placement constrains prepared the UCF file is to be added in the ISE project. After doing this press implement design as showed in figure X. After synthesis completion open FPGA editor. Open the NCD file in the ISE project root directory which should have the top entity name. If manual rerouting is required it could be wise to create a backup of this file.

3: Verification

The NCD file contains the routed design mapped to FPGA. Before considering the implementation a success it must be verified that no external wires cross into the reconfigurable region. Internal wires of the reconfigurable module must not interest with the bus macros other than in the end points. If these requirements are fulfilled then a correct design has been achieved and one can skip to section 5.

4: Manual rerouting

Manual rerouting is likely to be necessary after place and route. [2] demonstrates forced rerouting by adding CLB and pins to a net. This only seems to work in one direction though, and another technique for manual rerouting will shortly be presented.

Start rerouting by unrouting the net in question. To do manual rerouting, select a pin of the unrouted net and a routing resource and press route. This will force the selected pin to use the selected routing resource. Manual rerouting can be performed to a certain extent followed by an autoroute command, thus making this a forced autoroute process. Nets can be completely manually routed but this might require both skills and efforts. A progression in pictures is shown bellow.

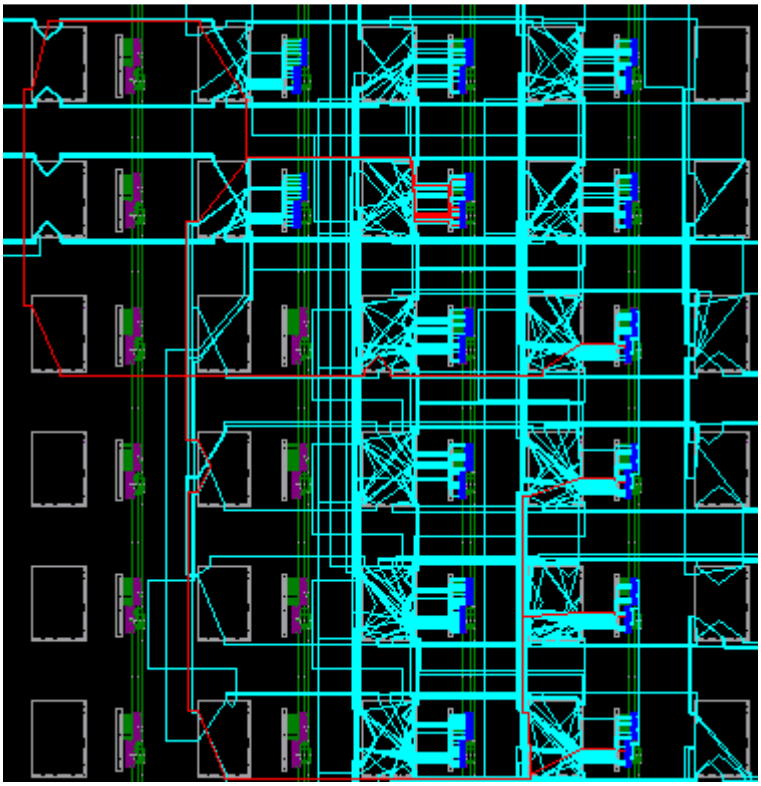

Figure 4.1: Illegal net

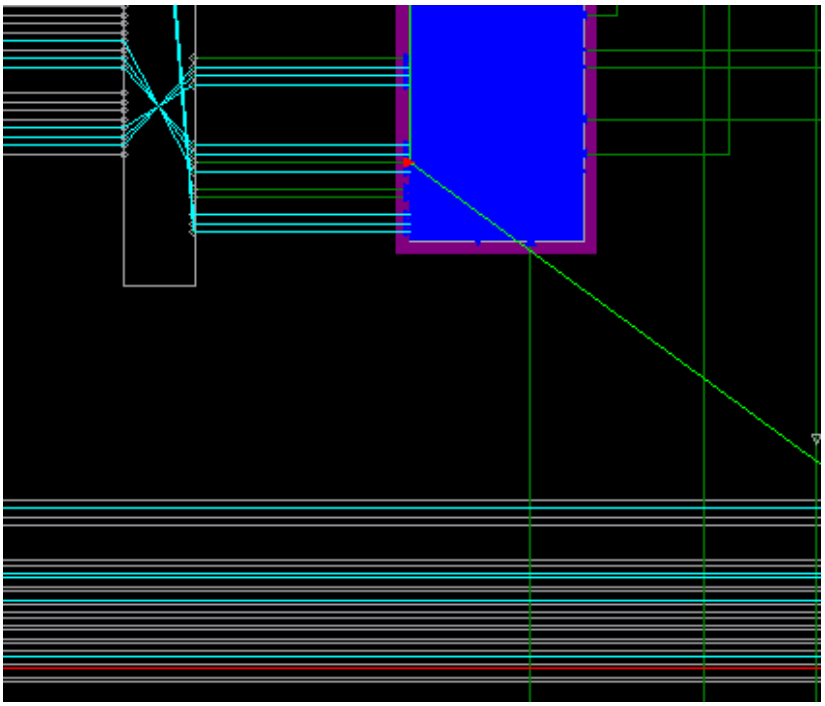

Figure 4.2: Selection of routing resource and pin after unrouting the net

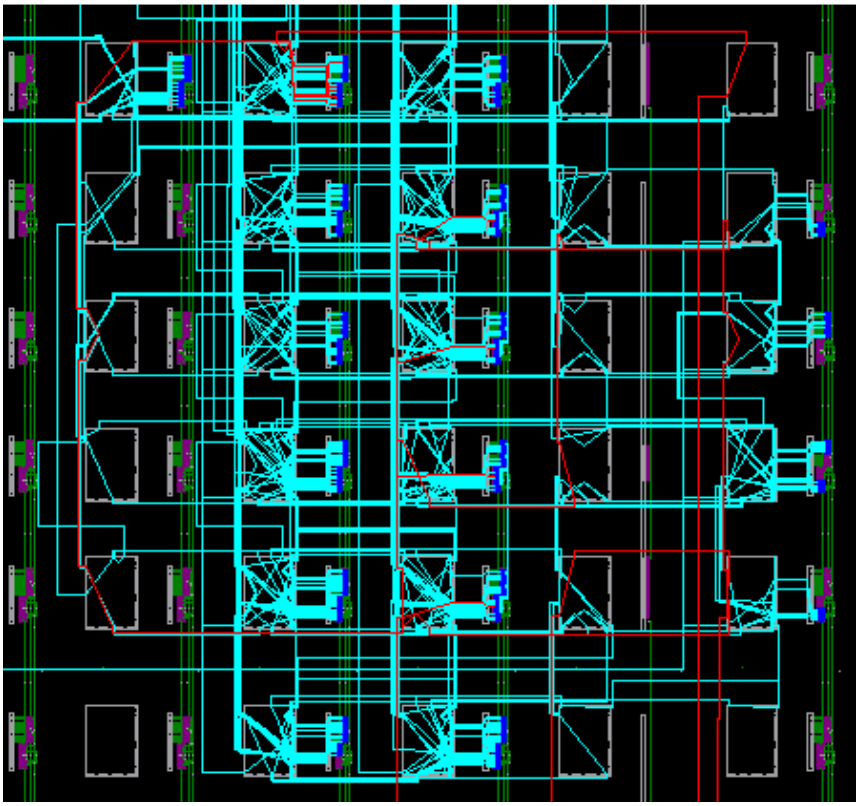

Figure 4.3: Net after rerouting

5: Bitfile export

The final step in reconfigurable module design is to export a design to bitfile. This can be done based on a NCD file through bitgen as shown in source 5.1. As this design methodology creates standalone reconfigurable modules, no processing through bitinit is required.

```
bitgen –w example.ncd
```

Source 5.1

6: References

[1] V. Endresen, Using Xilinx Planahead to create physical constraints, 2009
[2] V. Endresen, Creating a reconfigurable FPGA system, 2009

7: Final words

Comments or questions to this tutorial can be sent to vegarend@gmail.com