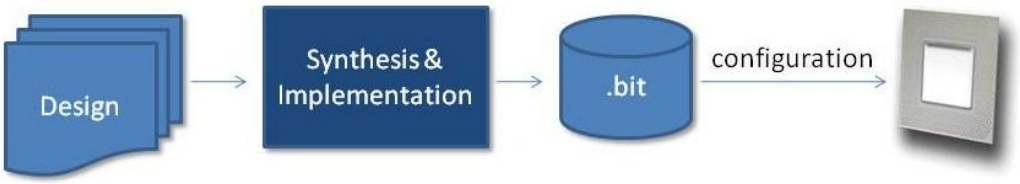
Partial Reconfiguration of Xilinx FPGAs

Programming an FPGA

An FPGA is, by definition, a programmable device. An FPGA designer defines the function that the FPGA is to implement using one or more design entry methods. These include schematics, hardware description languages such as VHDL or Verilog, and tools to implement DSP algorithms and embedded processors. A combination of synthesis and implementation tools converts the design into a bitstream, which is used to program the FPGA.

Configuration

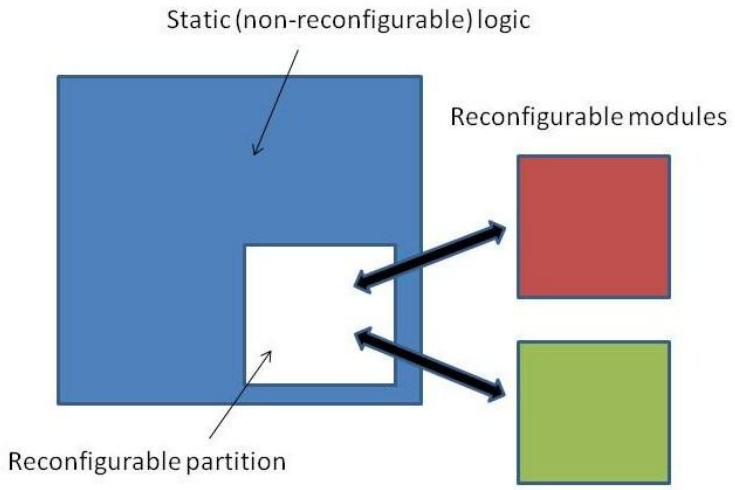
Many FPGAs are SRAM-based: the bitstream is loaded into the FPGA's configuration SRAM, which in turn programs the logic in the FPGA. The process of loading the FPGA with a bitstream is called *configuration*. Configuration usually takes place during system bootup. During development and debugging, configuration may be performed interactively, for example by downloading the bitstream from a PC using a USB cable.



Once an FPGA has been configured in this way, it starts to perform the required function. To change the FPGA's function normally requires that a new bitstream be downloaded, completely overwriting the original configuration; a system reset or power cycle may be required.

What is Partial Reconfiguration?

Partial Reconfiguration is where part of the FPGA is reprogrammed, using a partial bitstream, while the rest of the FPGA continues to operate without interruption.



Why bother?

There are a number of reasons why it might be useful to perform partial reconfiguration.

- to 'time multiplex' the function of an FPGA
  - reduces chip count: saves space and reduces cost
  - saves power by 'swapping out' unused functionality or I/O standards
  - creates a more flexible system: choose appropriate algorithms and protocols
- to provide a faster boot-up time
  - for example to meet PCI Express enumeration time requirement of 100ms
- to enhance system security
  - meet security requirements for classified bitstreams
  - enable more secure storage of private keys for decryption

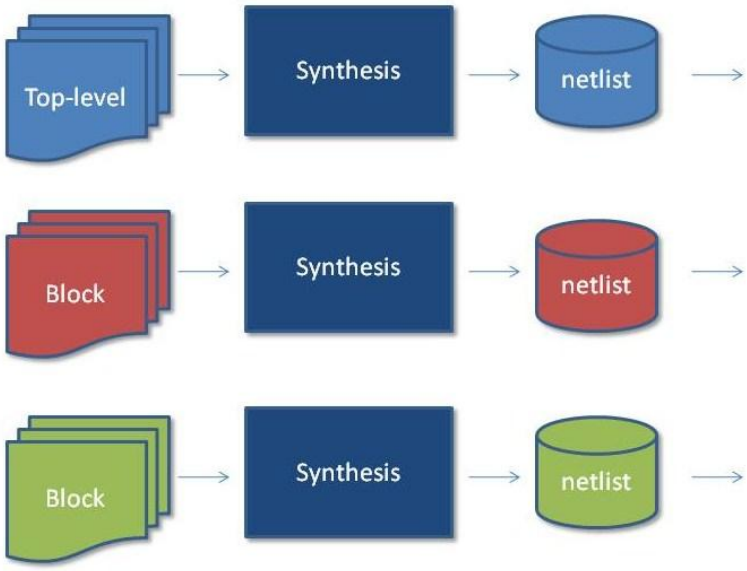
How does it work?

The remainder of this article describes partial reconfiguration as implemented in Xilinx® Virtex® FPGAs using ISE 12.1 and later. Altera® has announced a similar capability in their upcoming 28nm Stratix® V FPGAs; you can find details from Altera's web site.

HDL and Synthesis

1. Break the design into a hierarchy, where each reconfigurable partition is in its own hierarchical block. Ideally, both the inputs and outputs of each block should be registered.

2. Synthesize each such block as a separate netlist. There will be two or more alternative netlists for each reconfigurable partition. Each of these is termed a reconfigurable module.



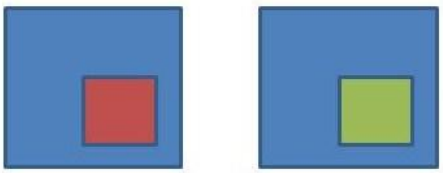
Synthesis is supported using Xilinx's XST and third-party synthesis tools such as Synopsys Synplify Pro and Mentor Graphics Precision Synthesis tools.

Implementation

3. Create partitions and area constraints (a floorplan) for these hierarchical blocks and define them to be reconfigurable partitions.

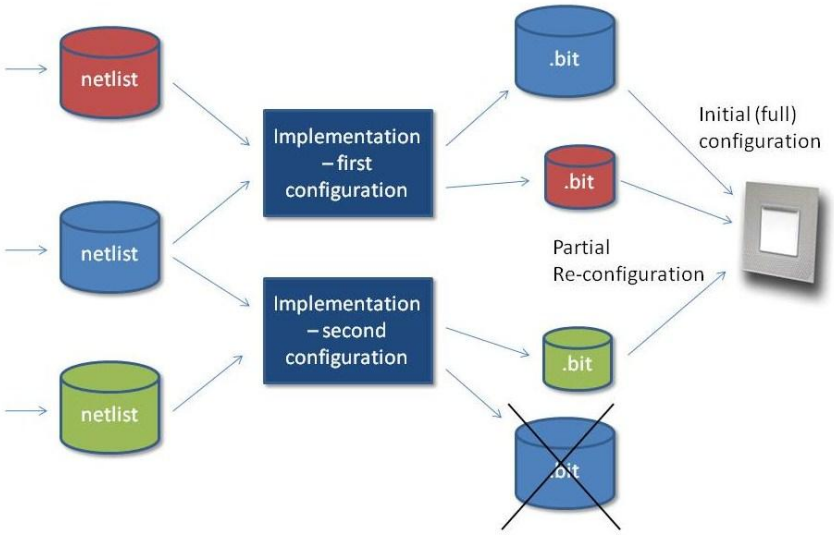
4. Allocate reconfigurable modules (netlists) to these reconfigurable partitions.

5. Define one or more configurations. Be careful about the terminology – a 'configuration' in this context comprises a number of netlists, where each reconfigurable partition is associated with one netlist, or none: the partition could be a 'black box'.



Alternative configurations

6. Implement a default configuration and any additional configurations. Each configuration will generate one full bitstream and one partial bitstream for each reconfigurable partition/module.



For Design Implementation using the Xilinx ISE tools, Partial Reconfiguration is supported with the PlanAhead tool and also from the command-line. It is not supported in the Project Navigator.

Configuration

8. Configure the FPGA using the full bitstream.

9. Perform partial reconfiguration using one of the partial bitstreams.

Various configuration methods are supported.

Special Considerations

As well as following usual good practice for hierarchical design, there are some special considerations.

Clocks – you have to be very careful with global clocks. Because of the way the tools reserve clock resources, the static (non-reconfigurable) logic may be starved of global clock resources.

Resets – Xilinx FPGAs include a global set/reset (GSR) signal, which is asserted during configuration. This ensures that after configuration all flip-flops are in a known state. However, the GSR is NOT asserted after a partial reconfiguration. You may therefore need to provide a separate, local reset for your reconfigurable modules.

Simulation – you can simulate any given design configuration, but at present you can't simulate the actual process of partial reconfiguration.

Decoupling – the design should be created so that it works correctly for every combination of reconfigurable modules, and during partial reconfiguration itself. To achieve this, special decoupling logic may need to be added to the design. For example, a reconfigurable module could be held in a reset state during reconfiguration or the flip-flops at the module's inputs and outputs could have their clock enables deasserted.

Further Information

This article is intended to provide a technical overview of Partial Reconfiguration as implemented in Xilinx FPGAs. To use this feature, you should refer to Xilinx's documentation, in particular the Partial Reconfiguration User Guide (UG702).

You can find this and further information about partial reconfiguration at Xilinx's web site: http://www.xilinx.com/tools/partial-reconfiguration.htm.

Doulos is Xilinx's Authorized Training Partner (ATP) in UK & Ireland and Northern California. Please contact Doulos or see http://www.doulos.com/xilinx for information about Xilinx training from Doulos.

About the Author

Michael Smith is a freelance consultant and trainer, with many years experience in the field of FPGA design. He has special expertise with Xilinx FPGAs and design software along with third-party software for FPGA design, as well as with tools and FPGAs from other vendors. Michael can be contacted at michael.smith@didache.co.uk.