

FPGA-based SOC Verification

Mike Dini
President
The DINI Group
<http://www.dinigroup.com/>

The paper will briefly discuss the issues related to implementing system-on-a-chip (SOC) intellectual property (IP) using FPGA's.

FPGA selection

Xilinx and Altera are the leading FPGA vendors. In order to emulate or prototype SOC-type IP, large FPGA's are needed. Furthermore, SRAM-based technologies, as opposed to one-time-programmable (OTP), FPGA's are desired so that the functionality of the resulting circuit board can be modified quickly, easily, and on-the-fly. SRAM-based technologies also gain the benefit of the SRAM fabrication technology curves, which means that SRAM-based FPGA's are typically one to two process technologies ahead of flash and antifuse.

Until recently, Xilinx has had a commanding lead over the competition with its VirtexII family of FPGA's, but recently Altera, with the Stratix family, appears to have caught up. The VirtexII family from Xilinx and the Stratix family from Altera represent the best available technologies to use for emulating or prototyping SOC IP.

Xilinx's VirtexII

This paper is being written at the end of October 2002. At of this date the largest FPGA available from any vendor is the 2v8000 from Xilinx. This device is very good, but an aggressive price probably limits it application in SOC emulation. This means that in practical application, the 2v6000 is probably the largest member of the VirtexII family that is applicable to real world SOC emulation.

The 2v8000 has a variety of features that are applicable to SOC emulation. The part has a large number of flip-flops, 93,000. In front of each flip-flop is a lookup table capable of performing any logic function of 4 variables, although this is an oversimplification. 168 separate, 18kbit dual-port memory blocks are included. Each of these memory blocks can be configured to a data width of 1,2,4,9,18 or 36. Several memory blocks can be combined to create larger memories. The structure of the embedded memory blocks allows for just about any type of storage. FIFO's, dual-port memories can be easily created. More complex memories such a 3-port must be created using flip-flops.

Embedded multipliers are also included – a function that doesn't map well to any known FPGA architecture. In the 2v8000, 168 -- 18x18 multipliers are available. DSP's and other applications that require filtering benefit greatly from these embedded multipliers, significantly increasing the amount of functionality possible in the device.

The 2v8000 can have up to 1108 I/O's.

Altera Stratix

As of press time, the largest part available from Altera's Stratix family is the 1S80. The 1S80 is slightly smaller than the 2v8000 with 79,000 total flip-flops. The pricing appears to be less aggressive, so, while still expensive, this part is applicable to real world applications.

The amount of logic per flip-flop is slightly less than VirtexII, but contains more than two times the amount of embedded memory. The 1S80 has 767, 32x18-bit memories, 364, 128x36-bit memories, and 9, 4096x144-bit memories. This amount of memory, at first blush, seems to be overkill, but functionality always tends to increase to utilize the available resources. The 1S80 can interface to 1238 I/O's.

Estimating FPGA Size

Xilinx calls the 2v8000 an ‘8 million system gate device’, but the datasheets never define what a ‘system gate’ is. For the purpose of determining the size of FPGA needed for a particular application, the number of ‘system gates’ is useless or worse.

Generally the number of flip-flops in a target FPGA is the constraining resource since LUT-based architectures such as VirtexII and Stratix have a good amount of logic (in the form of LUT’s) in front of each flip-flop. Designs that have a high logic to flip-flop ratio will violate this rule and the most common function that is in this category is an ALU. Large multiplexers implemented for the purpose of reading many memory-mapped registers also increase the logic to flip-flop ratio.

What to look out for when doing an FPGA conversion

SOC IP that has been written for an ASIC rarely, if ever, can be synthesized into an FPGA without modifications. Clock distribution is generally the worst of the problems, particularly if the original IP implements any form of power management/conservation. In comparison to ASIC’s, FPGA’s have very limited number of clock networks – typically on the order of 4-16. It is not uncommon for a carefully managed SOC IP component design intended for portable applications to have hundreds of gated clocks. There isn’t anything wrong with this, provided the clock gating is done correctly, but since FPGA’s cannot route clocks, more clock networks cannot be accommodated.

If the clock gating is relegated to a single RTL module, at least that module could be rewritten to better match the clock network architecture of the targeted FPGA.

Note that since SRAM-based FPGA’s lose configuration when power is removed, power management functions that require the tracking of the power state (D0, D3cold, et al.) cannot be implemented at all.

Generally all the memories must be adapted, and FPGA’s usually contain more embedded memory than an equivalent ASIC. Synthesis tools are well able to take functional descriptions of memories and map them to the embedded memory of an FPGA. Always include functional RTL code for a memory so that synthesis can do this work for you.

DesignWare libraries from Synopsys are very troublesome since no equivalent library exists for FPGA’s. When using Verilog, ‘ifdefs’ that switch in RTL describing the function can be used.

Partitioning

The best way to partition designs across multiple FPGA’s is to write the RTL with the partitioning constraint as a design constraint. This is rare since most ASIC designers generally think that they have other, more important problems to solve. If large designs must be partitioned across multiple FPGA’s, tools such as Certify from Synplicity <http://www.synplicity.com/products/certify/index.html>, and SpeedGate from Mentor <http://www.mentor.com/speedgatedsv/> can help.

Which tools to use

The design and debug of SOC IP can be divided into the following tasks: text editing, simulation, synthesis, place/route, and in-system debugging.

Text editing is a deeply personal thing, and no recommendation will be made. For simulation, several good simulators are competitively priced. We use ModelTech for both VHDL/Verilog, and Silos for Verilog. The Silos simulator from Simucad has a better user’s interface and is easier to use. The compile and execution time, however, is slower. Also, Silos apparently suffers from memory leaks and this problem forces frequent rebooting. The ease of locating errors in the Silos simulator, however, outweighs the bugs.

For synthesis, we prefer the products from Synplicity, although Leonardo Spectrum from Mentor works well also. Synopsys, unfortunately, is not competitive in the FPGA synthesis market. This is troublesome since most ASIC RTL is littered with Synopsys-related directives. It is easier to convert the code to something that Synplicity can handle than to deal with the problems associated with FPGA products from Synopsys.

The FPGA vendors provide Place and route tools. Different than a decade ago, SRAM-based FPGA’s route to completion with fixed I/O pins and high internal utilization. High utilization will hurt the performance of a design. Don’t expect to get high utilization (>90%) and fast speeds. Depending on the constraints, place and route can take anywhere from dozens of minutes to several days.

Existing Platforms

Several good off-the-shelf solutions are available in addition to user-designed custom boards. Insight and Avnet have PCI-based boards that contain the smaller FPGA's. The DINI Group has boards that contain the largest parts from either Xilinx or Altera. For DSP-related applications based on the Xilinx technology, Nallatech is a leading vendor.