Design of reusable VHDL models

Vassiliy Tchoumatchenko, Tania Vassileva, Borislav Korchev*, Alain Guyot**
*Technical University of Sofia, Department of Electronics
**TIMA/INPG, Grenoble, France

Abstract

The paper includes a brief overview to acquaint the reader with importance of design reuse, as well as a detailed discussion on how generics have been used in adders description, promoting models reusability. A parameterised structural VHDL models of fast prefix adders have been written using generics.

Proposed models are technology and tools independent. Synthesis has been carried out by Synopsys and Galileo tools. Different input operand size adders have been implemented targeting several FPGAs and standard cell technologies

1. Introduction

Today's deep submicron process technology affords designers the unprecedented opportunity to integrate complete system functionality on a single IC. Success with system on chip requires a completely different approach to design. Managing the complexities of the design environment and improving engineering productivity became critically important. Incremental and hierarchical design techniques have become increasingly important. Breaking a design into manageable segments allows to put more engineering resources into the design and to exploit concurrent software and design techniques. Further, to increase the productivity of the total design process, the system level design may be altered to accommodate available intellectual property (IP) blocks.

Design re-use is the only way to dramatically reduce system-chip cycle time, and to amortise the development costs on each unique block over multiple designs. Just like physical components of the PCB world, systems on silicon use on-chip virtual components. These design elements consist largely of "hard" and "soft" cores.

Hard cores offer the greatest potential for functionality, density and productivity. They provided a pre-designed, highly optimised, and completely characterised, fixed physical layout and are incorporated into ASIC design in a manner similar to standard-cell library element. The fixed form and function, however, make optimisation of hard cores for performance or density difficult.

Soft cores provide implementation flexibility. They are in HDL form and require synthesis into the target ASIC technology. For features that are commonly customised, VHDL generic parameters can be used that enable designers to select during synthesis data bus width, numbers of bits in a data path (e.g. bit-width of an adder, multiplier or ALU) and width and/or depth of a RAM or ROM. Such parameterisation eliminates the need to modify the actual design source and any subsequent re-verification, while at the same time allowing needed customisation.

A parameterised structural VHDL models of fast parallel prefix adders have been written using generics. The paper includes a brief overview to acquaint the reader with importance of design reuse, as well as a detailed discussion on how *generics* and *generate* statements have been used in adders description, promoting models reusability.

2. The Importance of Design Reuse

Design reuse is quickly becoming the primary means for reducing design time.

The essence of reuse is the capture and codification of knowledge. There is no way that engineers can reuse an entity without knowing its precise function and the parameters under which it operates. Reuse did work its way into chip level design in a standard cell design methodology. However, there is still more to gain through wider use of megacells and functional blocks. Furthermore, there is still vast potential for reuse at different levels of design abstraction.

The ability to reuse individual or multiple blocks from old designs increased reliability and productivity. Using VHDL virtual component allows to reduce amount of time and bugs introduced by data re-entry. Parameterised VHDL models minimise rework, maximise reuse and optimise the choice of architecture for the specific application. Incorporating variable word length they ensure more design flexibility.

The functionality during synthesis will accommodate increased design flexibility by providing a high-level technology-independent code that designers may use to parameterise, or modify.

3. Keys to productivity

Recently the importance of megafunctions in the automatic synthesis environment has been clearly stated. The appropriate use of megafunctions reduces design time, increases revenues by getting to market sooner, and gains time to focus on differentiating features. Megafunctions are available from silicon providers, who specialise in functions optimised for their own architectures and independent developers, who have expertise in functions for specific applications.

Megafunctions are called through instantiation or inference in VHDL synthesis tool.

When using synthesis tools, designers must keep in mind several considerations to achieve the expected results. The synthesis results strongly depends on the specification of correct optimisation constraints and the description style of the synthesis input. However, many digital systems contain frequently occurring design parts, mostly data path components like adders, counters etc. The synthesis of such components often leads to unacceptable performance. Schematic based design may provide better solutions than the synthesis from formal hardware descriptions. A way to overcome these problems is the use of automatic module generation. The generation algorithms base on proved designs, their quality is guaranteed by the applied expert knowledge of the algorithm developer. Because of the simple specification needed and the short generation time, the design effort is reduced significantly and the generation result is independent of the designer's knowledge.

In case of technology-specific module generation the algorithms imply some information to get the best results on the desired target architecture, so one module generator system is required for every technology or group of devices with similar properties. Especially for FPGA and CPLD architectures, this leads to a great development effort. To support a wider range of device families, a new approach is needed.

Our goal was to develop a design methodology and capability that permits and promote maximum reusability of megafunctions by using structural VHDL models. The models are parameterised in order to incorporate variable word length for application flexibility. Proposed models contain no technology dependant information thus facilitating design migration.

4. Design of reusable VHDL models

The fast prefix adders were chosen because they have a significant number of application. The addition is the most frequently used arithmetic operation, involved not only in simple addition, but also in more complex operations like multiplication and division. Fast adders are used in the ALU, the floating-point unit as well as for address generation in case of memory access.

4.1 Using *generics*

Capturing a complex function in a VHDL description provides technology and vendor independence for its implementation. Models can also be made independent of an instantiating architecture, i.e., they can be parameterisable. In VHDL, this is accomplished through the use of generic parameters (*generics*). Generics are a mechanism used to pass information such as delay times, load capacitance, word lengths, number of inputs, etc., to an instance of an entity, thus allowing much more application independence and, therefore, code reusability.

Generic constants, such as number of inputs, are declared in entity section of the models. Figure 1 illustrate how generics and variable length vectors are used to describe the inputs and outputs of parameterised adder model.

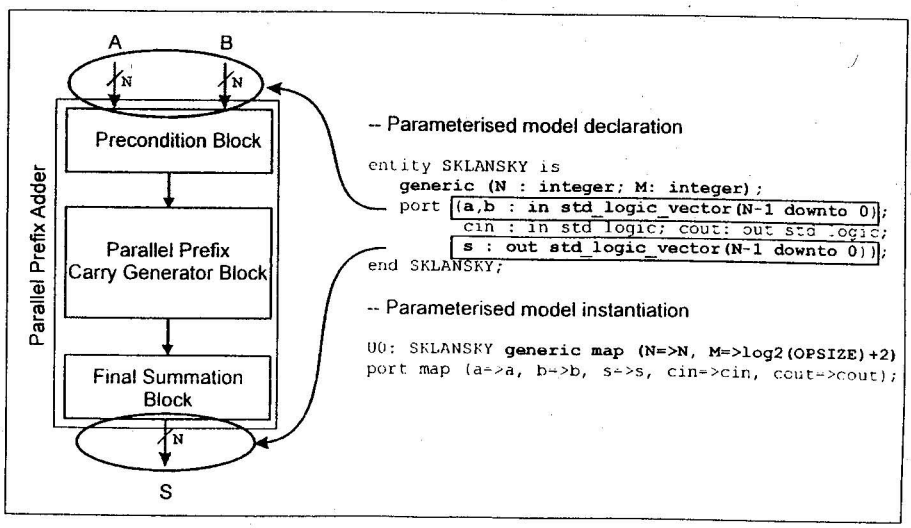

Fig.1. Adder's operands and result size parameterisation using generics

4.2 Using *generate* statements

Like the input-output port declarations, the internal adder architecture also depend on operands width. In structural VHDL description stile this dependence is accounted for by using *for-generate* and *if-generate* statements. Figure 2 shows the schematics of 8-bit Sklansky adder and part of the VHDL code used to describe the Carry Generator Block. The bulk of the carry generation array is created by generate loops (GEN4, GEN5) which instantiate GP and G components (U3, U4). The formation of the array (GEN6) in controlled by the function *bit_is_one(i,j)*, which returns Boolean value "true" when the *j*-th bit of *i* is "1". This follows from the observation that in the *i*-th column of the array "o" cells should be placed in nodes which correspond to "1" in the binary representation of *i*. For example, in the 5-th column of the array the cells are placed on the first and third rows since the binary representation of 5 is 0101.

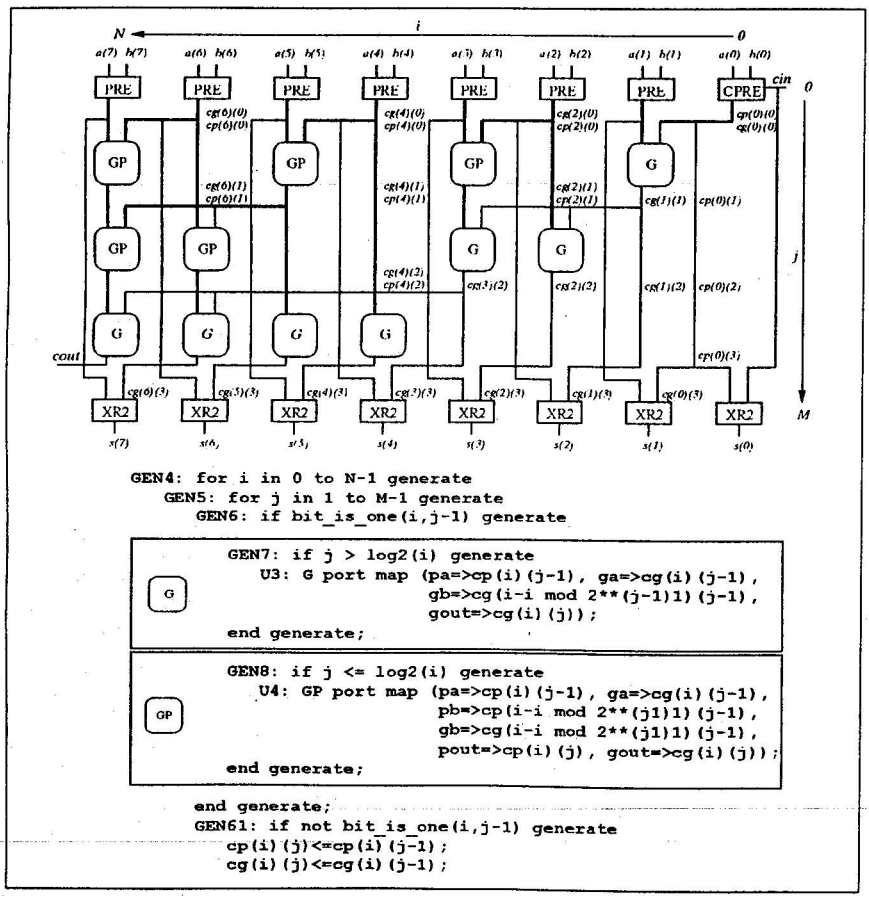

Fig. 2. Schematics and parameterised description of 8-bit Sklansky adder

5. Examples

We have done synthesis to gate-level on several cases. Several fast adders architecture with 8, 16, 32, and 64 bits were synthesised using Synopsys Design Compiler and FPGA Compiler targeting Xilinx 4000, Xilinx 5200 and Altera Flex8000 FPGA families as well as Atmel/ES2 ECPD10 standard cell library. A gate-level circuit of 8-bit adder synthesised from the Sklansky parameterised model is shown on fig.3.

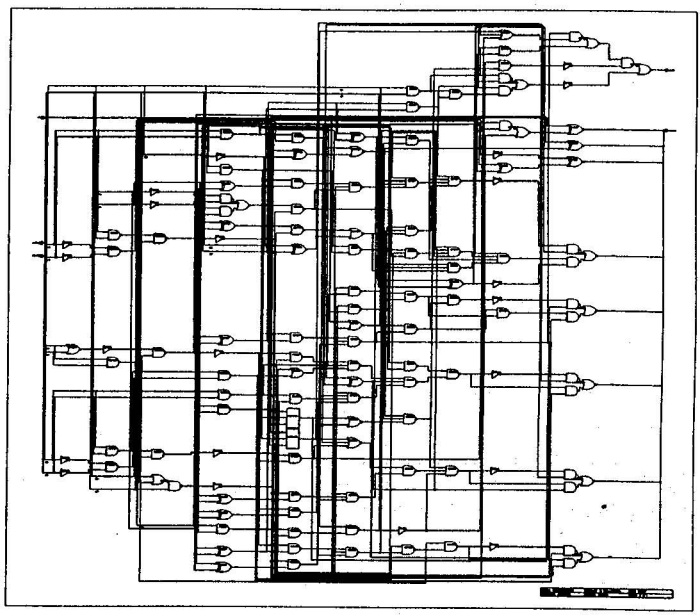

Fig.3. Gate-level circuit of 8-bit Sklansky adder (ECPD10 target technology)

Synthesised adders were compared in terms of delay and circuit area (fig.4). Evaluations have been made according the synthesis tools estimations. The Synopsys Design Ware adder generator *DW01_add* is used as a reference implementation. The performance of the circuits, produced by the proposed VHDL models, is comparative and in most cases better that the performance of structures synthesised by DesignWare module generator.

Summary

Reducing the cost of designing very large integration (VLSI) microcircuits is a continuing objective of the designers. Design re-use is the only way to dramatically reduce system-chip cycle time, and to amortise the development costs on each unique block over multiple designs. Parameterised models promote re-usability. To use them for different applications the operands size should be specified by giving a value to only one parameter.

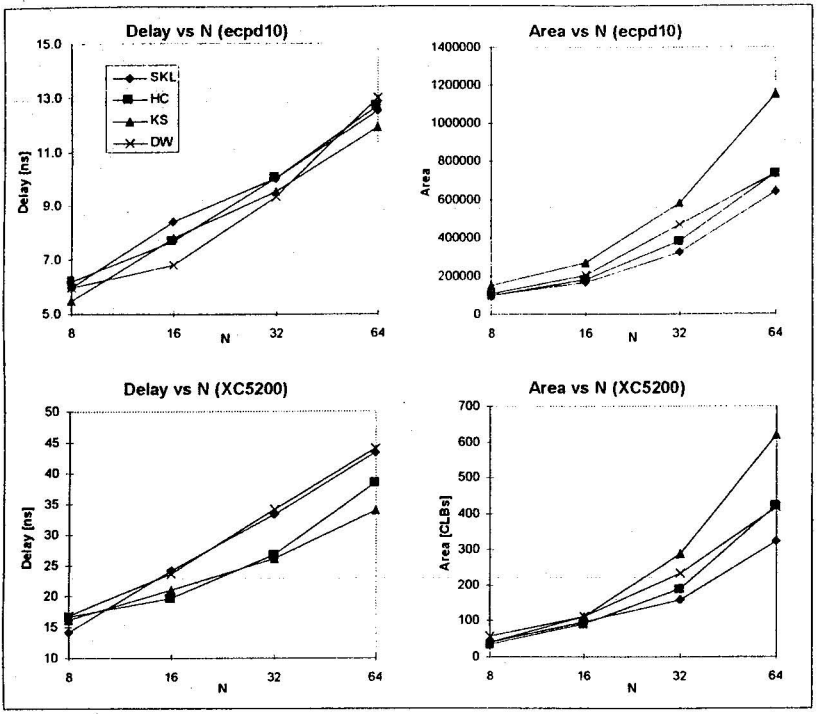

Fig.4. Comparition of Sklansky (SKL), Han-Carlson (HC), Kogge-Stone (KS) and DW01_add (DW) adders for two technologies – standard cell (ECPD10) and FPGA (Xilinx XC5200)

Parameterised VHDL models for fast parallel prefix adders was developed. The models are synthesis-ready and can be included in larger designs. They permit synthesis of Sklansky, Han-Carlson and Kogge-Stone adders architectures for different target technologies and any input word lengths.

Proposed models are technology and tools independent. Synthesis has been carried out by Synopsys Design Compiler, FPGA Compiler and FPGA Express and Galileo Exemplar. Different input operand size adders have been implemented targeting several FPGAs and standard cell technologies (Xilinx 4000, Xilinx5200, Altera Flex8000 and ES2 ECPD10 standard cell).

References

[1] J. Buurma, Virtual Components and the Well-Connected Engineer, ED, pp. 53-56, January 6, 1997

[2] M. Aberbour, A. Houelle, H. Mehrez, N. Vaucher, A. Guyot, A Time Driven Adder Generator Architecture, in Proc. *IX IFIP International Conference on VLSI, VLSI'97*, Gramado, Brasil, August 1997

[3] T.Vassileva, V. Tchoumatchenko, V. Shishkov, A. Guyot, High-performance adders synthesis using efficient macro generator, Proc. *European Conference on Circuit Theory and Design*, ECCTD'97, Budapest, Hungary August 1997