COMMUNICATION PROTOCOLS IN INTEGRATED CIRCUITS: EXPERIMENTS AND RESULTS

Ricardo N. B. Lima, Emerson Carli, Aloysio C. P. Pedroza,
{belem, emerson, aloysio}@gta.ufrj.br
Luci Pirmez and Antônio C. de Mesquita
{luci}@nce.ufrj.br and {mesquita}@lpc.ufrj.br

Grupo de Teleinformática e Automação - GTA
Universidade Federal do Rio de Janeiro
COPPE/EE - Programa de Engenharia Elétrica
P.O. Box 68504 - 21945-970 - Rio de Janeiro - RJ - Brazil

Abstract

This work presents the results obtained using a methodology that allows the hardware implementation and the rapid prototyping of communication protocols. The implementation results of different description styles of protocols and recommendations to the use of this methodology are discussed. A comparison among the protocol implementation using the High Level Synthesis technique with standard cells library and the implementation using PLD is presented. The results include area analysis and clock frequency evaluation of synthesized hardware.

1 Introduction

This work presents the results obtained using a methodology that allows the hardware implementation and the rapid prototyping of communication protocols. This methodology is based on the use of Estelle [1] and VHDL (*Very High-speed Integrated Circuits Hardware Description Language*) [2] languages, High Level Synthesis (HLS) techniques for digital systems design automation [3] involving two design environments, one for protocols design and another for the integrated circuit implementation.

The design process of integrated circuits generally begins with a high level system specification, where this term refers to a system level behavioral description, using a set of design tools integrated into a design environment. The design process of communication protocols, in many cases, involves system specifications written using Formal Description Techniques (FDT) such as the ISO (*International Standard Organization*) Estelle language. These languages allow to generate specifications in a level of abstraction even higher than that commonly attained in the description of digital systems. As a result, the high speed protocol designers will face the problem of interacting with two design environments: one for protocol specification, verification and testing and the other for synthesis, simulation and physical implementation of a circuit. In general these two environments are based on different description languages requiring an interface to translate one language into the other.

In the methodology [4, 5] used in this work, the design process assumes as input description a correctly compiled and simulated Estelle specification, obtained in a suitable protocol design environment, called CAD-Estelle. This description is mapped into a VHDL behavioral description, similar to the protocol functional specification which is translated into a RTL (*Register Transfer Logic*) equivalent level through the Synopsys Behavioral Compiler tool (SBC) [6]. The resulting lower-level design specification is then used as input of the logical synthesis tool Synopsys Design Compiler (SDC) [7] and finally the layout editor tool Opus [8] generates the circuit layout.

Some constraints in the use of this methodology are discussed. Comparisons using different VHDL behavioral

descriptions are shown in order to evaluate the complexity of the resulting hardware. Besides, a comparison among the protocol implementation using the HLS technique with standard cells library and the implementation using Programmable Logic Devices (PLD) is presented.

The paper is organized as follows. Section 2 presents the main characteristics of the Estelle and VHDL languages and the HLS concepts. Section 3 contains the main steps involved in the application of the methodology for the hardware implementation of protocols. Different description styles for protocols in VHDL efficient for HLS are presented in section 4. In section 5, the obtained results are examined for the effectiveness of the hardware implementation proposed in terms of the synthesized circuits area and clock frequency. Final remarks are drawn on section 6.

2 Specification Languages and High Level Synthesis

Software implementation of communication protocols involves a specific environment for protocol verification and testing. Hardware synthesis of systems involves an environment aiming the simulation and physical implementation of a circuit. In general, these two environments are based on different description languages requiring an interface to translate one language into the other. This section presents the main specification languages characteristics used in this work and the HLS concepts.

2.1 Specification Languages

This work uses for the software implementation of communication protocols the Estelle language. In Estelle, a *specification* is composed of a set of *modules*, where each module is composed of a *header* and a *body*. The interaction points, the exported variables and the class attribute are defined in the header. The module behavior is defined in the body through an extended state machine.

The hardware description language used in this work, VHDL, is a standard language to describe digital eletronic systems. The basic component of a VHDL description is the *entity/architecture* pair. A VHDL "entity" describes, through a set of input/output ports, the external interface of a given system. The "architecture" describes the entity's function either in the behavioral or the structural domains. For an efficient information exchange between VHDL entities describing protocols it is necessary some kind of synchronization among them, by control signals included in both entities.

2.2 High Level Synthesis

The adoption of multi-domain and multi-level languages able to describe digital systems at several representation domains and levels stimulated the HLS usage. Among these languages, VHDL is most used language by the synthesis tools, this allowed the development of hardware design environments, where the synthesis internal steps are automatically implemented.

A complete description of a VLSI (*Very Large Scale Integrated*) circuit involves three domains of representation: the *behavioral*, the *structural* and the *physical* domain. At the behavioral domain only the functionality of the system is described while in the structural domain a specific architecture implementing this functionality. VHDL do not include the physical domain, where the layout of actual components are specified to the fabrication process.

Each domain can be hierarchically divided in several abstraction levels, causing the design of integrated circuits to be classified by their domain and by their level. HLS corresponds to a behavioral-to-structural mapping in a level higher than RTL. The behavioral synthesis tool used through out this work is the SBC. It maps a high-level circuit design specification into a RTL equivalent level and then synthesizes this lower-level specification by the logical synthesis tool SDC.

Communication protocols are control flow dominated machines. This simplifies the synthesis procedures for this kind of system. The starting point of the HLS procedure is a behavioral VHDL specification in the system level. The target architecture of HLS is a general computing model that contains data path, memory, and control elements. This architecture is synthesized automatically by the SBC and SDC tools through scheduling, allocation, register sharing, memory and control inferencing. Designers do not need to specify the exact architecture of a design, and they can explore many implementations to find the optimal one.

3 The Adopted Design Methodology

The steps involved in the methodology of design for hardware implementation of protocols are summarized in the diagram of Figure 1. The procedure uses initially a suitable design environment of protocols in Estelle, the CAD Estelle. An Estelle specification correctly simulated (steps 1,2) is converted into a VHDL description using the Estelle-to-VHDL Translation Strategy (step 3) developed by Pirmez in [9].

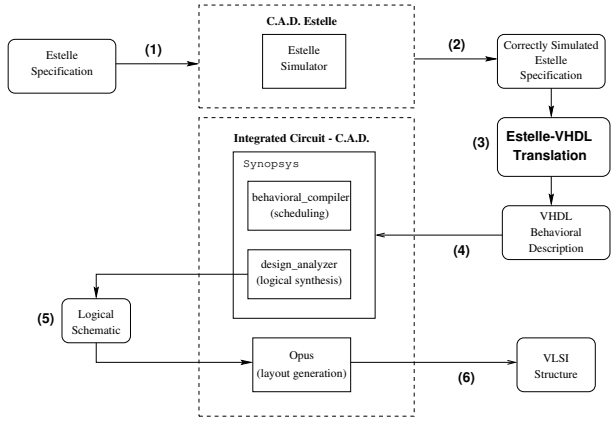

Figure 1: The design methodology.

This VHDL description will be used as input (step 4) of a HLS tool, that includes the SBC and SDC tools and Opus layout editor. The SBC maps a high-level specification into a RTL or an equivalent level description which is syhtesized by the logical synthesis tool SDC (step 5). The Synopsys tool perform also VHDL behavioral and structural simulations. Opus is applied to the layout generation (step 6) of the VLSI structures.

4 Description Styles for Protocols in VHDL

A protocol can be described by many different VHDL behavioral descriptions. These descriptions are semantically equivalent and produce the same simulation results. However, they can produce results in the HLS procedure widely different. The quality of synthesis process is measured by the number of control states generated by the scheduling process performed by the SBC tool, which is responsible by the total area of the resulting integrated circuit. It will be shown that the quality of the HLS output description will depend on the type and order of statements used in the HLS input description, which characterizes a "description style" [10, 11].

Style	Main Characteristics
1st	1) "*case state*" clause 2) "*wait until...*" statement to each branch of "*case*" clause 3) output signals are reseted inside the "*case*" clause
2nd	equal to First style, but "*wait until...*" is placed outside the "*case*" clause
3rd	1) "*if..then..elsif..end if*" construct 2) test variable is *state* variable 3) "*wait until...*" is placed outside the "*if*" construct 4) output signals are reseted inside the "*if*" construct
4th	equal to Third style, but test variable is *input signal*
5th	1) "*case state*" clause 2) "*wait until...*" is placed outside the 2) "*case*" clause 3) output signals are reseted immediatly after the "*case*"
6th	equal to Fifth style, but "*if..then..elsif..end if*" instead "*wait until...*" statement

Table 1: Main characteristics of VHDL description styles.

To illustrate this fact, consider the ISO's ABRACADABRA reference protocol modelled by state machines that interact through the exchange of messages. These message exchanges are performed by an opening/closing connection process. This protocol has two phases: the initialization phase and the processing phase. The last one consists of a loop that waits for an input signal, process the input data and outputs the corresponding result. This model will be used to illustrate the influence of the several description styles possible in VHDL on the HLS results.

The Table 1 shows the main characteristics of six different VHDL description styles used for implementing the state machine associate to the ABRACADABRA protocol. These different description styles are introduced

in SBC and SDC tools. The circuits synthesized in each case and the complexity of the resulting hardware will be analyzed in the section 5.

5 Results

In this section, the effect of the different VHDL behavioral description styles in the synthesized hardware is evaluated, a comparison among the protocol implementation using the HLS technique with the standard cells library and the implementation using the PLD technique is shown and some design recommendations are discussed.

5.1 Description Styles

As discussed, VHDL allows the description of a given behavior in many different ways. In order to improve the performance of the synthesized hardware it is necessary to adopt a VHDL description style that accounts for the structure of the scheduling algorithm used by the synthesis tool.

Styles	Control States	Area (mm^2)	Clock Frequency
First	14	1,33	125 MHz
Second	9	1,10	149 MHz
Third	9	1,13	130 MHz
Fourth	9	1,37	130 MHz
Fifth	9	1,27	154 MHz
Sixth	4	0,66	154 MHz

Table 2: Effect of differences in VHDL descriptions on the synthesis results.

To illustrate this fact, a comparison among the synthesis results generated by the SBC tool for six different description styles is shown on Table 2. This comparison is done in terms of the number of states corresponding to the state machines generated by the synthesis algorithm for each description style, synthesized circuit area and clock frequency. The clock frequency evaluation is made from the critical path analysis found in the synthesized circuit. From this table some important conclusions are drawn:

1. use the "*wait until...control signals*" statement outside the "*case state*" clause. The circuit area decreases and clock frequency increases;

2. use the "*case*" statement instead "*if..then..elsif..end if*" structure. The circuit area decreases and clock frequency increases;

3. reseting the different output control signals inside the "*case*" clause the circuit area decreases. While reseting the different output control signals immediatly after the "*case*" the clock frequency increases;

4. employ the "*if..then..elsif..end if*" with the *state* variable as the outer signal instead the *input* variable. The circuit area decreases and the output controls are smaller;

5. use the "*if..control signals*" structure instead "*wait until...control signals*" constructs. The circuit area decreases.

In terms of circuit area and clock frequency the sixth description style presents the best results. In Figure 2 the logical schematic of the opening/closing connection process of the ABRACADABRA protocol generated by the HLS tool for the sixth description style is shown. The synthesized circuit has a 0,66 mm^2 estimated area, obtained using the AMS, 0,8μm technology standard cells library.

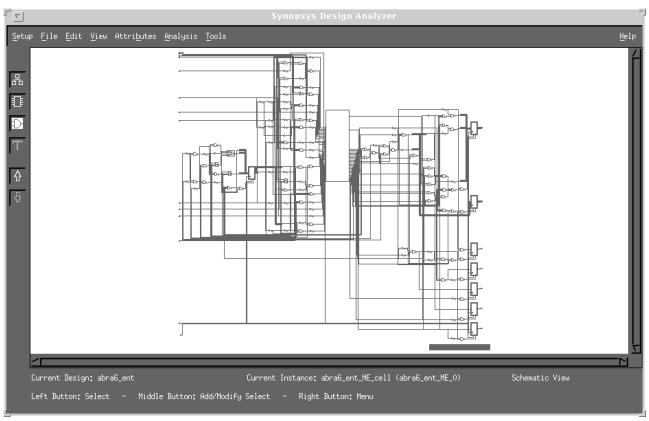

Figure 2: Logical Schematic of part of ABRACADABRA protocol.

5.2 Comparison of Implementation Techniques

Programmable Logic Devices (PLDs) are digital integrated circuits that can be programmed to perform complex functions, i. e, PLDs can implement any boolean expression defined through programming.

However, implementing the structure generated by the HLS process by means of a standard cells library has some definite advantages by report to PLD implementation. The standard cells design offer greater degree of freedom in allocation and routing of cells in the layout of the chip, implying in a best tradeoff between area and speed.

A comparison was performed using the opening/closing connection phase of the ABRACADABRA protocol. This specification was implemented using AMS, 0,8μm technology standard cells library and the ALTERA Max+Plus II (*Multiple Array Matrix Programmable Logic User System*) design compiler [12]. During the compilation, the ALTERA was mantained at the auto mode to choose the component that fits better the project. In this case the component EPF10K10LC84-3 was chosen.

This example shows a clear advantage of the standard cells implementation by report to PLD implementation. The clock frequency obtained in each case was 154 MHz for the standard cells implementation and 33 MHz for the PLD implementation. On the other hand, it must be stressed that PLD implementations are immediate, that is the system can be re-programmed without a new fabrication cycle, as is the case of standard cells, being an advantageous tool for rapid prototyping. Moreover the area of the standard cells implementation was considerably smaller than the PLD implementation. Note that once chosen, the ALTERA component has a fixed area independent of the system being implemented.

5.3 Design Recommendations

5.3.1 Use of Integer Variables in VHDL

When a software implementation of a protocol is though it is common practice to use integer variables and signals in the formal protocol specification. At the Estelle description level, this should cause no major problems. Nevertheless, when hardware description languages, like VHDL, are used it is necessary to limit the range of each integer variable or signal in order to reduce the number of bits of the registers associated to each variable by the synthesis tool.

For example, if no range is attributed to an integer variable the SBC tool will allocate a default number of 32 bits to each register. To illustrate this fact, two VHDL descriptions of the opening/closing connection process of the ABRACADABRA protocol containing one integer variable were synthesized. In the first description the range of the integer variable was not limited. In the second description the range was set from 0 to 7. The table 3 shows a comparison of the synthesized circuit area and the number of bits allocated to the registers corresponding to each description.

Description	**Synthesized Registers**	**Area (mm^2)**
Range	1 of 7 bits	1,27
No range	3 of 32 bits	2,37

Table 3: Comparison of synthesized circuits.

Observe that in the description with no range limitation, three 32 bits register were associated to the integer variable, while in the other description only a 7 bits register was used. This represents a reduction in area of the synthesized circuit of approximately 54%.

5.3.2 Data Storage

The first step in circuit synthesis consists in partitioning the specification in two blocks: the logic structure and the the data storage structure. The data storage structure is generally implemented through an external memory. Thus, before performing the Estelle-VHDL translation all data storage structures corresponding to internal variables, as buffers, should be implement as a separate VHDL entity.

	Synthesized Registers	**Standarcell Memory**	**Area (mm^2)**
SAR1	3 of 704 bits 1 of 384 bits	NO	77,41
SARmem	NO	1K x 8bits	9,79

Table 4: Comparison of synthesized circuits.

This fact is illustrated in table 4 by the implementation of the SAR module of the AAL5 protocol [4]. This module is responsible for the segmentation of the received data. In this implementation, two buffers are allocated to the storage of characters, the first buffer stores 88 characters and the second one 48 characters. Both were specified by internal variables in the VHDL description, called SAR1.

It should be noted that the SBC tool synthesizes these buffers as a group of registers internal to the circuit. A better solution would be to synthesize the control logic and connect it to an external memory. This solution is illustrated in the VHDL description, called SARmem.

6 Final Remarks

The results obtained from a methodology that allows the hardware implementation and rapid prototyping of communication protocols were presented. It was shown that the circuit resulting from the HLS procedure depends on the VHDL description style used to specify the system to the HLS tool. This was evaluated from the number of control states generated by the schedule step in the HLS process, that are directly coupled with the area and clock frequency of synthesized circuit. An example using the opening/closing connection phase of the ABRACADABRA protocol was discussed. Depending on adopted description style, a 49% reduction of the synthesized circuit area and a 23% increase of the clock frequency were noticed. The logical schematic of this protocol process, considering the best description style, was presented.

It was also shown that the standard cells implementation is 4.5 times faster than a PLD implementation. Nevertheless, PLDs are important tools for rapid prototyping of communication systems.

Some design recommendations refering to the limitation of the VHDL integer variables range may result in a 54% reduction of the synthesized circuit area. In the example discussed the data storage structures involved in protocol operation were assumed to be implemented an external memory connected to the control logic. This results in a 87% reduction of the synthesized circuit area.

References

[1] ISO/TC97/SC21/WG16-1, "Estelle: a Formal Description Technique based on a Extended State Transiction Model", *DP9074*, 1987.

[2] IEEE Std 1076-1987, *IEEE Standard VHDL Language Reference Manual*, Mar. 1988.

[3] Gajski, Daniel D. et al., *High-Level Synthesis - Introduction to Chip and System Design.* University of California, Irvine: Kluver Academic Publishers, 1992.

[4] Lima, R.N.B., "Projeto de Protocolos visando a Implementação em Hardware: Aplicação ao Caso do Protocolo AAL5 para Redes ATM", Master's thesis, Universidade Federal do Rio de Janeiro, PEE/COPPE/UFRJ, Rio de Janeiro, Brasil, 1998.

[5] Carli, E., "Projeto de Protocolos visando a Implementação em Hardware: aplicação ao Protocolo IPv6 no Ambito da Abordagem IP over ATM", Master's thesis, Universidade Federal do Rio de Janeiro, PEE/COPPE/UFRJ, Rio de Janeiro, Brasil, 1998.

[6] Synopsys, Inc., *Behavioral Compiler User Guide Online Documentation, v1998.02*, 1998.

[7] Synopsys, Inc., *Synopsys Online Documentation, v1998.02*, 1998.

[8] Cadence Design Systems, Inc., *Cadence Online Library*, 1994.

[9] Pirmez, L., *Uma Metodologia para a Implementação de Sistemas Distribuídos em Hardware a partir de uma Descrição Formal.* PhD thesis, Universidade Federal do Rio de Janeiro, PEE/COPPE/UFRJ, Rio de Janeiro, Brasil, 1996.

[10] Pirmez, L. and Rahmouni, M. and Kission, P. and Pedroza, A. and Mesquita, A. C. and Jerraya, A. A., "Analysis of Different Protocol Description Styles in VHDL to High-Level Synthesis", in *EURODAC'96*, Geneve, 1996.

[11] Pirmez, L. and Pedroza, A. and Mesquita, A. C., "High Level Synthesis of Protocols Described by a Formal Description Technique", in *IX IFIP International Conference on VLSI'97*, Gramado, RS, Brazil, 1997.

[12] Altera, Corp., *Data Book and Max + PlusII Getting Started*, 1997.