Int. J. Engng Ed. Vol. 15, No. 1, pp. 72–76, 1999
Printed in Great Britain.

0949-149X/91 $3.00+0.00
© 1999 TEMPUS Publications.

Successful CAD Tools Application to FPGA/ASIC Design*

D. NICULA
Transilvania University, Department of Electronics and Computers, 29, B-dul Eroilor, 2200, Brasov, Romania

M. CIRSTEA
De Montfort University, Electronics Department, The Gateway, Leicester LE1 9BH, UK.
E-mail: marci@dmu.ac.uk

The paper presents the successful experience developed in the area of FPGA/ASIC design at Transilvania University, Brasov, Romania. A state-of-the art computer-aided design (CAD) studio has been set up as a result of a successful TEMPUS grant. The present project is concerned with the design of an adaptive differential pulse code modulation (ADPCM) hardware decoder, aiming to develop and verify a top-down design methodology using advanced CAD tools. The digital part of the system was modelled and designed using very high speed integrated circuit hardware description language (VHDL), then it was synthesised and downloaded into a XILINX FPGA for rapid prototyping. Successful tests carried out have allowed the final implementation into a standard cell ASIC. The pioneering of a complete integrated circuits design flow using modern CAD tools at Transilvania University in Romania had a major positive impact for this important higher education establishment.

INTRODUCTION

THE CONTINUOUS progress of application-specific integrated circuits (ASICs) fabrication technology is leading to an exponential increase in the single-chip system complexity. Modern design methodologies use a top-down approach which involves the use of advanced computer-aided design (CAD) tools in order to shorten the design cycle. A key point of this approach is a hardware description language (HDL) capable of offering support at all levels of description (behavioural, structural, physical). The verification of textual description correctness requires a test-bench that will ideally be able to exercise the modelled device under test at all levels of description.

This article presents the pioneering of VHDL/FPGA/ASIC design at Transilvania University in Romania, using modern software tools like V-System, CADENCE, XACT (XILINX). The supportive successful collaboration with De Montfort University in Leicester, UK, has previously been established. The goal of this initial experience was to prove the existence of software, hardware and knowledge resources at Transilvania University in order to design a complex digital system in order to modernise the curriculum provision for undergraduate students in electronics. The resources have been created as a direct result of a very successful three-year TEMPUS programme, JEP-08180-94. The programme had participating institutions from Ireland, UK, Spain, Belgium and Romania. The Department of Electronics and Computers at Transilvania University became a member of Europractice and has purchased within this TEMPUS project the hardware and software resources required for setting up a state-of-the art EDA studio consisting of 1 HP workstation and 20 PCs running the following software packages: Cadence, Synopsys (ws), Xilinx, Synopsys FPGA Express, V-System, Max-Plus II Altera (PC). The studio has allowed the training of the lecturers in the most modern EDA packages, providing an up-to-date curriculum to students. A course in Modern EDA Techniques was set up. The first complete design flow was experimental, which was finalised by FPGA and Standard Cell implementation of the design, as described. This has opened the way for complex final year projects to be carried out in the Romanian establishment. Figure 1 shows the block diagram of the methodology containing:

- behavioural VHDL description and simulation;
- VHDL code validation using a prototype board and programmable devices;
- VHDL synthesis on a standard cell technology.

A speech synthesiser acting as a 'talking toy' has been designed, based on a hardware Adaptive Differential Pulse Code Modulation (ADPCM) decoder. The ASIC includes the ADPCM decoder itself, a ROM and a DAC. Figure 2 illustrates an algorithmic view of the decoder and transfer functions, where:

* Accepted 20 November 1998

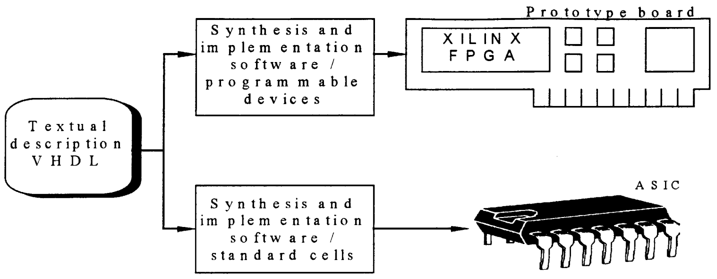
Textual description VHDL
Synthesis and implementation software / programmable devices
Synthesis and implementation software / standard cells
Prototype board
XILINX FPGA
ASIC

Fig. 1. VHDL to standard cell top-down approach.

$L(k)$ = code word at time k;
$M(|L(k)|)$ = multiplication step at time k (dependent on $L(k)$);
$P(k)$ = prediction at time k;
$P(k) = 0.875Y(k-1)$;
$\Delta(k)$ = quantisation step at time k;
$U(k)$ = product of code word and quantisation step at time k;
$Y(k)$ = output at time k.

THE TOP-DOWN APPROACH

A top-down methodology was used to model the system. The first decomposition identifies three main components: a ROM, a multiplier and a controller. On the second decomposition step, the controller is divided into a control path (*Ctrl*) and a data path (*Data*):

- **Control path** is implemented like a Moore Finite State Machine (FSM). The state register receives the general clock and reset signals and provides enable and selection signals for data path.
- **Data path** basically consists of data registers, multiplexers and some combinatorial logic. The data path makes use of two additional external components: a ROM and a multiplier unit. During the first iteration these two units will be behavioural modelled, for simulation purpose only. Then, a synthesisable model for the multiplier will be developed. Depending on the technology used, the ROM will be implemented as an external chip (if the whole system will be implemented using FPGAs) or as a macro cell (if the whole system will be implemented using standard cell). An architectural view of the data path is illustrated in Fig. 3.

The ROM

The behavioural model is capable to initialise the content of the ROM using data provided in an ASCII file. The contents file format is the following: ⟨Address⟩⟨Data⟩⟨Comments⟩ where:

- *Address* = integer number in range of 0 to $2^{(\text{number of bits of address})} - 1$
- *Data* = string of ASCII characters '0' and '1' (same number of characters as ROM data dimension).
- Comments = ASCII string containing any character

The multiplier

The external multiplier is devoted to perform two multiplications per sample $M(|L(k)|)\cdot\Delta(k-1)$ and $L(k)\cdot\Delta(k)$. It has to multiply two numbers represented in C2 format. These reasons have determined the choice of a multiplier using the Booth algorithm.

MODELLING STRATEGY

Modelling strategy follows the rules in [1]. Data type *std_logic* declared in *IEEE package std_logic_1164* and arithmetic operators declared in the *IEEE package std_logic_arith* were used.

The package

Definitions for some constants are included in a package for parametrisation purposes. There are two kinds of parameters: data path dimensions and clock parameters. The first category includes the number of bits for representing variables (Δ, M, U, L, Y). The second category includes sample period, the number of clock periods per sample, and other three values that must be correlated with the precedent ones.

After the computation related to one sample is performed, the controller must measure the time to another sample. Nothing happens during this time, but it has to be exactly determined, otherwise the sampling rate is modified. The VHDL model implements a counter for the number of clock pulses between ending computation for one sample and starting computation for the next sample. To parameterise this counter and to implement it using a minimum D-type flip-flop count,

$$\Delta(k) = M(|L(k)|)\cdot\Delta(k-1)$$
$$U(k) = L(k)\cdot\Delta(k)$$
$$Y(k) = P(k) + U(k)$$

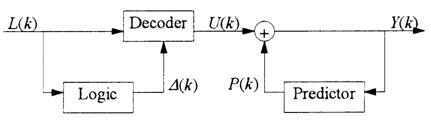

Fig. 2. ADPCM decoder XEDX algorithmic view.

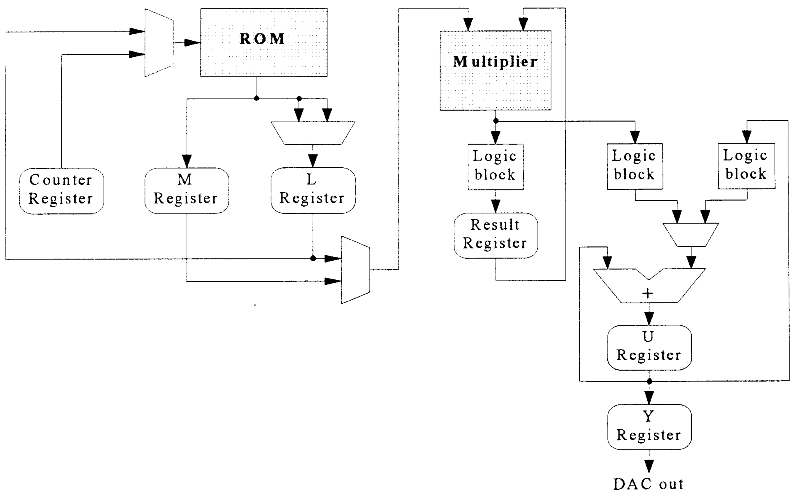

Fig. 3. ADPCM decoder XEDX architectural view of data path.

after synthesis additional constants must be declared, such as number of bits and stop value. Due to the fact that package declaration contains only constants, package body is empty.

The control path

This is modelled like an implicit Moore FSM as stated in [1]. There is no explicit state register declaration. To ease the understanding all output signals were assigned values in all states. The '*after*' clause is ignored by the synthesiser. The placement of an '*after*' clause to all signal assignments was determined by the need to avoid 'set-up time violation' error messages generated during mixed simulations of a structural model of data path (after synthesis) and a behavioural model of control path (before synthesis and without *after* clause).

The error messages were due to the virtual configuration simulated: the control path has no delay between the clock signal and the output signals which act as ENABLE signals for the register in the data path. This means that the registers in the data path would receive exactly at the same time both clock and data signals, a condition which is preferred to be avoided. The value of the delay is not essential as long as it is greater than D-type flip-flop set-up time. The controller has three states where the two inputs are tested. On two states the controller is waiting for multiplier result. On the third state, the controller is waiting for starting the next sample computations. These three states are modelled using '*while*' loops.

The data path

This is modelled using many processes. Each process relates to a one-state FSM, modelling a register and the upper combinatorial logic as illustrated in Fig. 3. All processes have only one output that is actually the register output. All registers receive the global clock and reset signal. The reset signal is an asynchronous one and it is modelled in accordance with [1].

The ROM model

This makes use of *textio standard package* in order to read the memory content from an ASCII file; this model is used for simulation purpose only. An empty architecture of the ROM called 'dummy' is also developed and it is used during the synthesis of the system (including the ROM). The ROM model generates 'X' values on the output if a reading is made on an un-initialised or invalid address.

The multiplier

The multiplier implements a Booth algorithm. It is able to multiply two numbers represented in C2 format. On its turn, the multiplier was divided in two parts: *control path* and *data path*. Again, the control path is modelled as a Moore FSM. The data path contains two registers (n bits and $2n$ bits) and an ALU used only for addition and subtraction.

Design for test

Many internal signals from the control path and the data path were modelled for testability and debugging purposes. These signals could be easily hidden during the final synthesis process. The simulation has been carried out partially on a PC using V-System/Windows and partially on a workstation, using Leapfrog from Cadence. The test bench has a particular process devoted to generating an ASCII file including all modifications of DAC outputs. The format of the file is:

⟨sample number⟩⟨time⟩⟨binary value⟩⟨integer value⟩

For example:

33 4252151 NS 1111111100 −4

VHDL MODEL VALIDATION VIA FPGA IMPLEMENTATION

Simulation results are very valuable [2] in order to check the functional behaviour of the VHDL

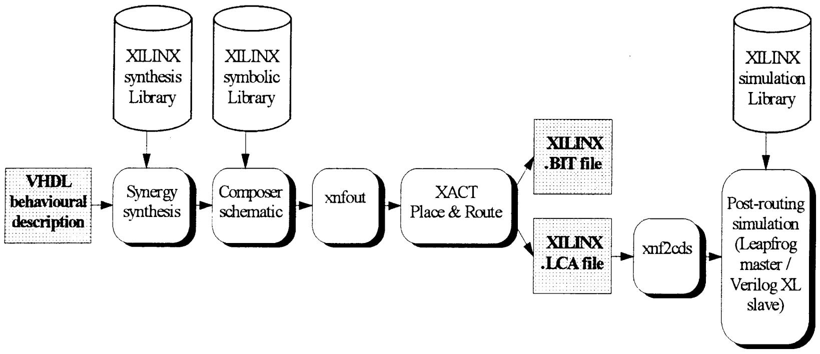

Fig. 4. Design flow—FPGA implementation.

model, but on many occasions it is the hardware validation which provides significant information before the decision is made to invest in an ASIC. The cheapest and fastest way to validate the VHDL code is to develop a prototype board containing re-programmable devices.

There are several advantages of using this method before converting the design into an ASIC [3]:

- It can ensure an error-free design before the silicon implementation.
- Using a re-programmable device allows the shortening of the time required to correct a mistake.
- It allows the use of the prototype board for the hardware testing of other components of the system.

Considering the ADPCM decoder architecture, the choice was made to implement the design into an XC4004 Xilinx FPGA device. The design flow is illustrated in Fig. 4 involving the use of CADENCE (workstation software) and XILINX XACT (PC version). The VHDL code has been synthesised using Synergy from Cadence. A VHDL structural model, a Verilog structural model and a schematic representation were obtained. After synthesis the structural VHDL code has been simulated again. A mixed VHDL-Verilog simulation was performed using Leapfrog and Verilog XL from Cadence. This was the only choice in order to preserve the test-bench (written in VHDL) when only Verilog library of components was available.

A top-level schematic was drawn using Composer from Cadence and a symbol was mapped to the structural schematic obtained from synthesis. A

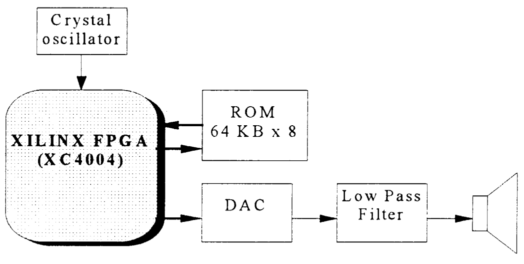

Fig. 5. Demonstration board.

netlist file in XNF format was obtained from the schematic. Following the design flow proposed by XACT a .BIT file was then generated on a PC and downloaded using an Xchecker cable into an FPGA device for implementation. After partition, place-and-route, a post-routing mixed simulation, was performed on the workstation: the device under test has been converted from .XNF to Verilog and the test-bench remained in VHDL.

The demonstration board illustrated in Fig. 5 contains a XC4004 device, a crystal clock oscillator, an EPROM, a DAC, a low pass filter and an amplifier. An EXO-3 (2 MHz) was chosen as clock oscillator. The digital-to-analogue conversion is performed by a 10-bit DAC AD7533. The 16kB EPROM allows a 4 second message to be stored. The sound is of good quality for intended applications.

STANDARD CELL IMPLEMENTATION

Standard cell could be viewed as a trade-off between a relatively short design time of gate arrays and the high performance of a full-custom design. This technology gained a high popularity especially due to the development of synthesis, place-and-route software tools. Standard cell offers some benefits:

- short design time due to optimised logic cells;
- support for top-down design methodology;
- good trade-off between chip area and speed.

The implementation of the ADPCM decoder in *ES2* technology [4], process *ecpd07*, was chosen. This technology has only got digital cells. Macrocells for ROM and RAMs could be generated automatically. The design flow starts with the same VHDL description, validated through the FPGA implementation. Cadence offers support for the entire design flow presented in Fig. 6.

VHDL behavioural description has been synthesised using ES2 *ecpd07*, with constraints regarding minimum silicon area and particular signals (clock and reset) distribution. For the clock signal (fan-out of about 100 D flip-flops) a special buffer was required in order to avoid clock-skew. Structural description was simulated using

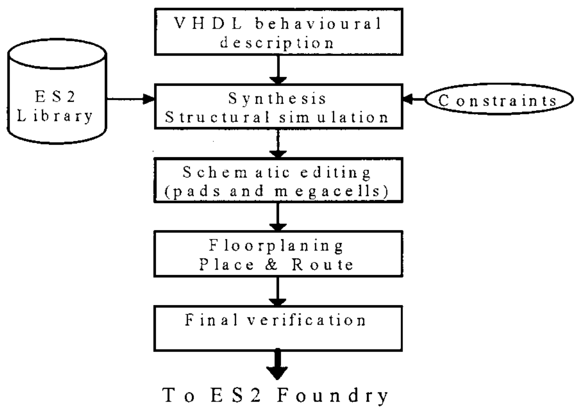
VHDL behavioural description
ES2 Library
Synthesis Structural simulation
Constraints
Schematic editing (pads and megacells)
Floorplaning Place & Route
Final verification
To ES2 Foundry

Fig. 6. Design flow—standard cell implementation.

the same test-bench and the results have been compared with those obtained from the behavioural simulation. Using Composer from Cadence a top-level schematic was drawn, including pads and a ROM megacell symbol. Then using Preview the floorplan was designed and the components were placed and routed. The critical paths (clock and supply) were individually routed. The final layout has been checked using DIVA and FAST. All files have been prepared in order to send the design to foundry.

CONCLUSIONS

The aim of pioneering the FPGA/ASIC design flow at Transilvania University was fulfilled with an important support from a TEMPUS contract. Using VHDL as a system textual description, the methodology assumes an intermediate phase of code validation using a prototype board and programmable devices. The final target was ES2 standard cell technology. The ADPCM decoder has been successfully implemented.

This first project development experience described in the paper has enabled a design methodology, using advanced CAD tools, to be established, with major benefits for the curriculum provision in this key Romanian higher education establishment.

REFERENCES

1. *Synergy VHDL Synthesizer and Optimizer Modeling Style Guide 1.2.1,* Cadence On-line (1998).
2. J. Ashenden, *The Designer's Guide to VHDL,* Morgan Kaufmann Publishers, San Francisco (1996).
3. S. Sjoholm and L. Lindh, *VHDL for Designers,* Prentice-Hall, London (1997).
4. *ES2 ASIC Design Guidelines,* European Silicon Structures, (1998).

D. Nicula is a senior lecturer at Transilvania University in Brasov, Romania, Department of Electronics and Computers. He graduated from the Polytechnic Institute of Bucharest, Romania, in 1990 with a M.Sc. in electronics. In 1997 he was awarded the Ph.D. in computer sciences by Transilvania University. His research interests are mainly in the area of VLSI design and VHDL modelling. Teaching duties include VHDL, microprocessors and digital electronics. Dr Nicula wrote the first Romanian book regarding VHDL and he is the leader of a team involved in VHDL modelling for third-parties.

M. Cirstea was appointed senior lecturer at De Montfort University in Leicester, UK, after completing his Ph.D. at Nottingham Trent University in October 1996. He graduated in 1990 from Transilvania University of Brasov in Romania with an M.Sc. in electrical and electronic Engineering. His research interests are mainly in the area of intelligent control systems for power electronics and drives, ASIC/FPGA design and implementation. He has published more than 30 scientific papers and is an active researcher, currently supervising five research students. Dr Cirstea is an Associate Member of IEE, member of the UK Engineering Council, Member of IEEE.