IP/SOC 2006

Session : Design Forum

SILICON VALIDATED IP CORES DESIGNED BY THE BRASIL-IP NETWORK

Ana Karina Rocha[4], Patricia Lira[4], Yang Yun Ju[4], Edna Barros[2], Elmar Melcher[1] and Guido Araujo[3]

[1]Federal University of Campina Grande, [2]Federal University of Pernambuco, [3]University of Campinas

[4]LINCS-CETENE, Recife, Brazil

Abstract :

This paper presents the complete VLSI design flow of the IP cores: MPEG4 decoder, MP3 decoder and 8051 microcontroller. Their implementation uses the ipPROCESS to guide the project flow and VeriSC methodology for functional verification.

1. INTRODUCTION

With the rapid evolution of the VLSI design technology, time-to-market is one of the crucial factors in the ultimate success of an integrated circuit project. Designers have, therefore, increasingly adhered to rigid, rapid and consistent design methodologies that are more amenable to design automation.

The traditional abstraction levels used by the digital design world are composed by 5 entities: the system, the module, the gate, the circuit and the device. Due to this design philosophy, the emergence of elaborate computer aided design frameworks for digital integrated circuit design is made possible .

By the effect, VLSI designers were freed from great part of back-end design flow and just concerned about the front-end functionality verification jobs. The substantial decrease in project cycle time and number of designers in VLSI projects could be seen as the direct consequences of this approach.

Unfortunately, by the increased design complexity, the CAD approach can not fully substitute the front-end verification task, which is fundamental to guarantee the equivalence between project specification and final circuit functionality.

Due to this fact, the ipPROCESS design methodology and VeriSC verification approach emerge as an attempt to guide the VLSI design flow and verification task for all abstraction level design.

2. DEVELOPMENT METHODOLOGY

The proposed methodology is based on a rigorous software design process and co-verification methodology. It has been used by over 70 designers of the BrazilIP Network[1], to design a real-world platform called Fenix containing a number of IP-cores: MPEG4, MP3, USB, Bluetooth, Keyboard/LCD, 8051 and Ethernet controller.

Figure 1 shows the ipPROCESS [2] development flow used. It provides a disciplined approach to assigning tasks and responsibilities within a development organization. Its goal is to ensure the production of high-quality IP-cores that meet the needs of system integrators.

The first step of the development flow is to identify and describe all project requirements in a document, which will be used throughout the entire project development. Based on this document a SystemC RTL specification of the project is constructed, that implements all the functionalities described before.

At this point VeriSC [1] is used for functional verification. Testbench templates in SystemC are automatically generated around each SystemC RTL block. These transaction level testbenches provide support for directed and randomly generated stimuli and use reference models in C or C++ to be self checking.

[1] BrazilIP Network is a consortium formed by eight large Brazilian universities (UNICAMP, USP, UFPE, UFCG, UFMG, UNB, UFRGS and PUCRS)

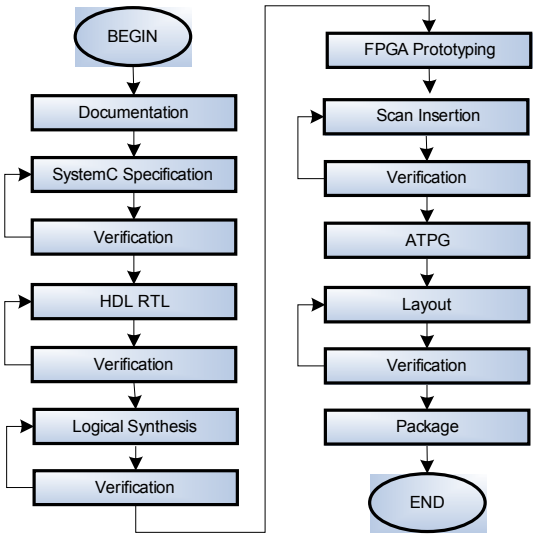

Figure 1. Development Flow.

The verification engineer implements the reference and the interface protocols (transaction level to signal level) using templates provided by the VeriSC Tool. Once the SystemC RTL specification is verified at block level and top level and all functionalities are working correctly, it is translated into Verilog RTL. Once again the project is verified inside its testbench using Verilog/SystemC cosimulation.

From the Verilog RTL, logical synthesis is executed and the generated Verilog netlist is verified again by cosimulation. Besides the functional verification, equivalence checking, timing analysis, and power analysis is used.

The design is then validated inside a real application on an FPGA-based development board.

The next steps, according to the flow, are to insert the test structures, verify again and then generate the test patterns.

With the final netlist it is possible to elaborate the IP-core layout that must be back annotated and verified again inside the top level testbench. The verification also includes DRC, LVS and antenna checking.

3. DEVELOPED IP-CORES

Three IP cores described in the following subsections were designed using this process.

They were developed using Forte and Synopsys tools. Cynthesizer was used for SystemC to Verilog translation, VCS was used for SystemC/Verilog cosimulation, PrimeTime for timing analysis, PrimePower for power analysis, Formality for equivalence checking, Design Compiler for logic synthesis, DFT Compiler for scan insertion, Tetramax for test patterns generation and Astro for the layout generation.

FPGA validation was done directly from Verilog-RTL using Quartus software and an Altera Stratix II development board [3].

Austriamicrosystem's C35B4 technology was used for layout. The LEF cells provided by the Hit Kit v.3.7 design kit had to be imported into a Synopsys Milkyway database.

The test methodology adopted for the project was full scan, and the scan style was multiplexed flip-flop.

3.1 8051 MICROCONTROLLER

An 8051 Microcontroller was the first module developed using the methodology presented in this paper. The 8051 developed is the standard version, with an instruction set of 255 opcodes. Also, this version includes a USART (Universal Synchronous Asynchronous Receiver/Transmitter) with 4 operational modes, 2 timers with 4 operational modes, an 8-bit parallel port, an interrupt handler with 5 interrupts and 4 priority levels and an internal RAM memory with 256 bytes plus an additional 128 bytes used by instructions with indirect addressing only.

The block diagram for the 8051 IP-core can be seen in Figure 2.

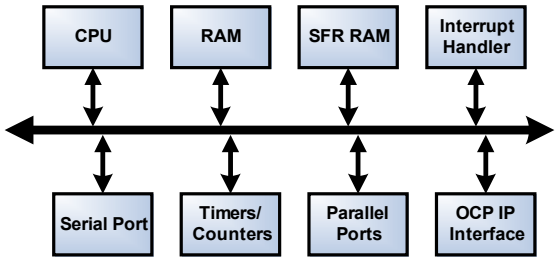

Figure 2. Block Diagram.

The 8051 microcontroller reference model used in VeriSC verification methodology was described using ArchC language [4]. ArchC [5] is an architecture description language, initially conceived for architecture description, which aims to facilitate and accelerate processor description.

The final layout of the 8051 Microcontroller can be seen in Figure 3.

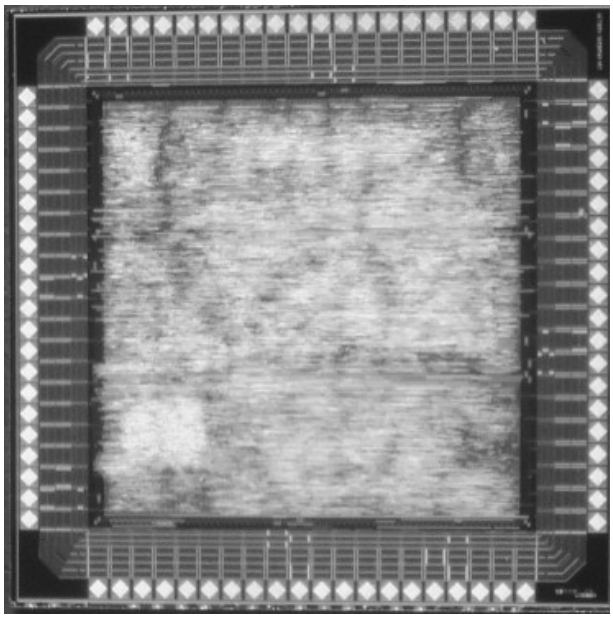

Figure 3. 8051 Microcontroller die.

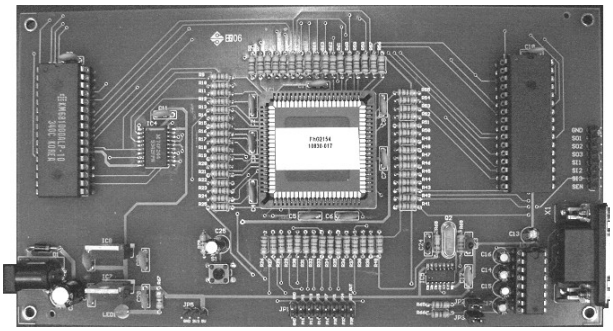

Figure 4. Printed Circuit Board.

The printed circuit board shown in Figure 4 was used to test the microcontroller, which is located at the center of the board.

A set of programs to validate the project was prepared and also benchmarks available at the Dalton project, from University of California [6] were adopted and the free version of Keil´s i8051 compiler was used. Additionally, a set of programs to test the 8051's peripherals was developed.

The Table 1 shows the programs used to test the 8051 microcontroller.

Table 1. Programs used to test the 8051

Programs
Testall
Negcnt
Gcd
Int2bin
Cast
Divmul
Fib
Sort
Xram
Program 1 – Port 1
Program 2 - Port 1, Timer 0, Interrupt
Program 3 – Serial Transmitter, Timer 1, Interrupt
Program 4 – Serial Echo, Timer 1, Interrupt

The testall program tests all instructions except ACALL, LCALL, RET, RETI, and MOVX. Each instruction is tested and, if applicable, the carry flag is also tested.

Additionally, for testing the integrated circuit in a real application, it was used to control a robot, called Jubinha. This robot, in Figure 5, uses a radio frequency transmitter, which sends the commands for the USART; a proximity sensor connected to the external interruption pin and two DC motors connected to the parallel port.

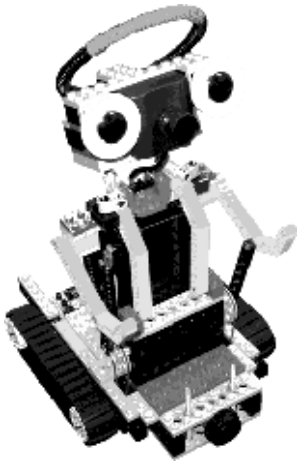

Figure 5. Jubinha robot.

3.2 MP3 DECODER

According the ISO/IEC 11172-3, the MPEG-1 layer-3 audio decoder is composed by several signal processing stages: Frame Depackage, Huffman Decoder, Signal Dequatization, Stereo Signal Decoder, Alias Reduction, Inverse Modify Discrete Cosine Transform && Signal Filter (IMDCT-WINDOW) and finally Discrete Cosine Transform && Signal Filter (DCT- WINDOW), see the Figure 6.

Throughout the MP3 decoder pipeline, all the processing jobs between IMDCT-WINDOW and DCT-WINDOW are called signal recuperation stage.

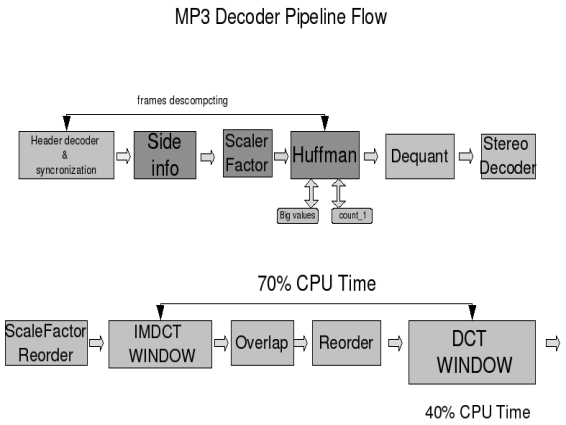

Figure 6. MP3 Decoder Functional Diagram.

The module DCT-WINDOW, specially, is responsible for 40% of hardware processing time, although it just represents 10% of estimated area. Due to this fact, it was chosen to be the prototyped as an ASIC design, whereas the other stages are designed to be in FPGA or software approach.

By the VeriSC verification methodology rules[1], a GNU licensed audio library, the LibMAD–0.15b, is chosen as the design reference model. The output PCM audio quality is guaranteed by comparing the results of DUT from that is of LibMad-0.15b.

According to the ISO reference suggesting, all the comparisons are done in a RMS (Root Mean Square) measurement.

The maximum absolute value of difference must be below 2^-14 relative to scale. This comparison criterion is employed in all development levels of the VeriSC design rule.

A raw die (Figure 7) and an encapsulated die (Figure 8) of DCT-Window is shown. The two blank areas in the die are 2Kx16 bits SRAMs.

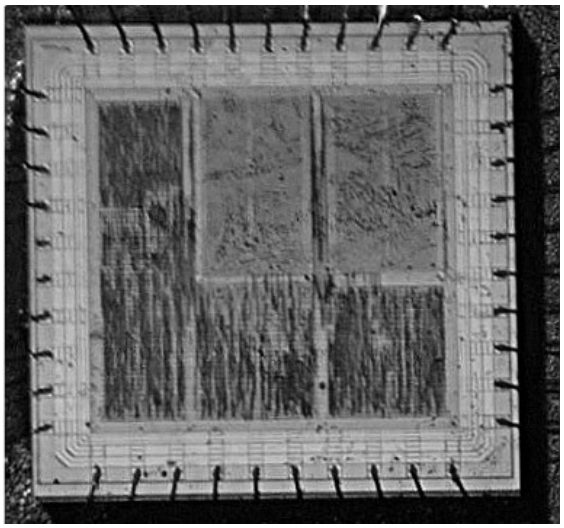

Figure 7. DCT-WINDOW Raw Die.

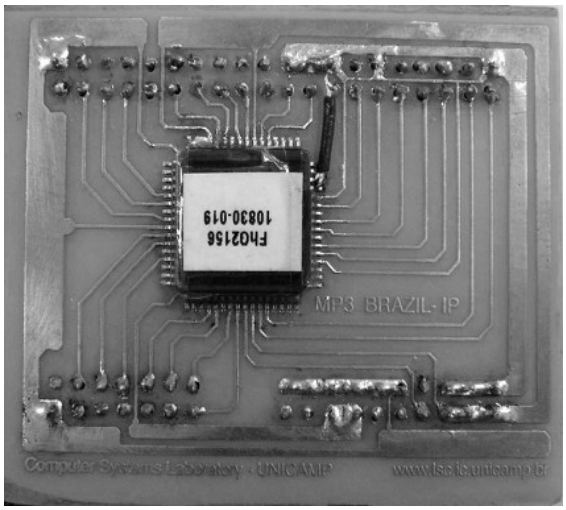

Figure 8. DCT-WINDOW Chip.

3.3 MPEG-4 DECODER

The third module developed using the presented methodology was an MPEG4-IP decoder. It is a Simple Profile / Level 0 video decoder. The MPEG4-IP architecture comprises custom hardware designs for bitstream parsing and variable length code (VLC) decoding, texture decoding, motion compensation and color space conversion. The MPEG4-IP decoder contains two OCP-IP interfaces, one slave interface for bitstream input and one master interface for bitstream output. A frame rate of 30 frames per second was achieved.

The block schematic of the MPEG4-IP decoder is reported in Figure 9. For bitstream parsing (BS) a processor with a application specific instruction set for VLC decoding was developed. The firmware is contained in a 16 KB ROM. All the other functional blocks in Figure 9 are dedicated FSMs.

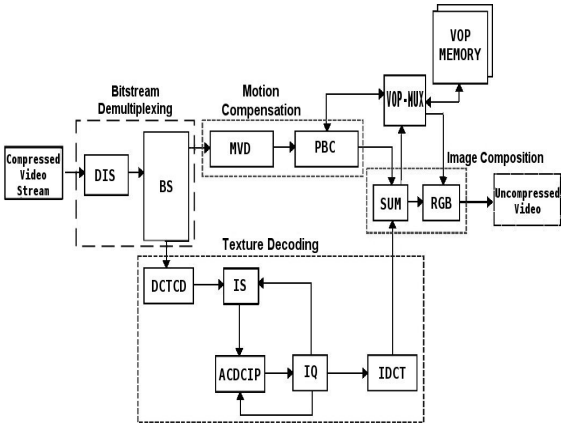

Figure 9. MPEG4-IP decoder schematic.

The design has been validated in FPGA-with encoded bitstreams read from FlashCard memory

and a LCD panel for visual demonstration of real-time decoded streaming video sequences.

The Figure 10 shows the final layout from MPEG-4 Decoder. A 2Kx16 bit RAM was used for the ACDCIP block. The two VOP memories of 38 KB each were left external to the chip layout to reduce prototyping cost.

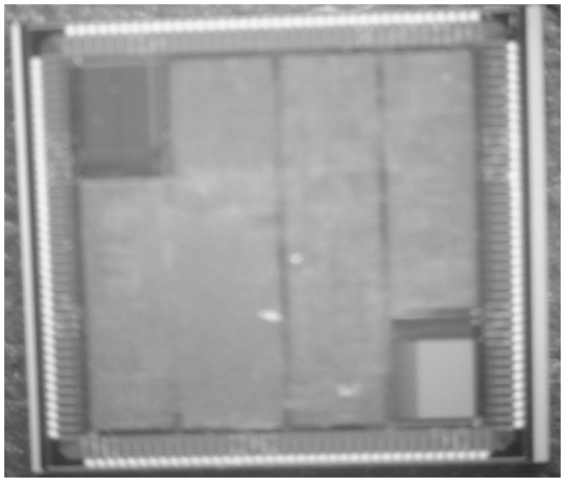

Figure 10. MPEG-4 die.

3.4 RESULTS

Table 2 shows the most important design parameters of the three developed layouts.

Table 2. IP-cores results

Data	8051	DCT Window	MPEG-4
Testbench code line number	23,762	898	51,382
RTL code line number	28,548	5,898	21,473
Logic gate count	20,700	17,950	48,095
Transistors	~186,531	307,691	~430,000
Silicon area	8.04mm²	9.86mm²	22.7 mm²

The three layouts have been developed in six months including the learning process for test and layout synthesis methodologies and tools.
All tests of the 8051 IP core chip and the DCT-WINDOW chip passed with success. Due to budgetary limitations, the test board of the MPEG-4 decoder chip is still being finished for thests to start.

4. CONCLUSION

The complexity of the IP cores designed demonstrates the effectiveness of ipPROCESS and VeriSC design methodologies to handle complex designs and enable the first time silicon success.. The MPEG-4 decoder is one of the most complex digital chip ever designed in Brazil.

5. ACKNOWLEDGEMENTS

The validation tests of the layouts would not have been possible without the assistance of Jean-François Naviner of Télécom Paris - ENST (École Nationale Supérieure des Télécommunications).
The work was suported by the Brazilian Research Council CNPq 55278/02-6 and the Synopsys University Program.

REFERENCES

[1] K. R. G. da Silva, E. U. K. Melcher and G. Araújo, An Automatic Testbench Generation Tool for a SystemC Functional Verification Methodology, SBCCI 2004 – 17th Symposium on Integrated Circuits and System Design, 2004, 66-70

[2] ipPROCESS*, development process for Soft-IP,* http://www.lins.org.br/ipprocess (accessed September 21, 2006).

[3] Nios II Development Kit*, Stratix II Edition,* http://www.altera.com/products/devkits/altera/kit-niosii-2S30.html (accessed September 21, 2006).

[4] P. Lira, V. Schwambach and E. Barros, A Strategy for Specifying SystemC Micro-controllers Models Using the ADL ArchC, IP/SoC, 2005

[5] Rigo, S., Azevedo, R., Araújo, G., Araújo, C. and Barros, E.: The ArchC Architecture Description Language and Tools. In International Journal of Parallel Programming. Kluwer Academic Publishers, 2005, 453 – 484

[6] Dalton Project, http://www.cs.ucr.edu/~dalton/i8051/i8051syn/ (accessed November 13, 2006)