DESIGNING AN IP-CORE FOR EDGE DETECTION IN MONOCHROME IMAGES USING THE SOBEL OPERATOR

Jody Maick Araujo de Matos, Fladmy Alves de Souza, Anfranserai Morais Dias

State University of Feira de Santana

ABSTRACT

The integrated circuit designs are reaching high levels of complexity. Due to the great importance of these devices nowadays they are performing increasingly complex functions. In this case the use of methodologies and tools in development process such devices are essential, as well as projecting Systems-on-Chip (SoC) with reusable IP-Cores. This paper describes the designing of a Soft IP-Core for edge detection in monochrome images using the Sobel Operator, applying the ipPROCESS methodology, a Brazilian initiative in order to create a standard and enhance the development of integrated circuit design in the country.

INTRODUCTION

Nowadays, electronic devices are more complex, adding more information, communications and entertainment, reaching more consumers. This market growth is due to decreased production costs [1].

However, the increasing complexity of Integrated Circuits designs and the speed at which products should hit the market, that resulted in the creation of tools and methodologies to be applied in development projects. One of the methods for IC's development are based on pre-designed and reusable components, called IP-cores (Intellectual Property - IP) [2, 3, 4].

In last few years, the federal government is encouraging the development of IC's, among the various initiatives highlight the Brazil-IP project. One of the many results produced was the development process for Soft IP-core designs, called ipPROCESS. Your goal is to ensure the production of high quality IP-cores [5].

This paper presents an IP-core development which performs edge detection in monochrome images using the Sobel operator [6]. Tools and methodologies specified by ipPROCESS were applied in the project. Although, the IP-core was embedded in an FPGA connected to an analog camera.

The approach taken in this paper surrounds a brief explanation of the ipPROCESS, describing their main characteristics and development methodology. The section 4 presents a description of the conception, architecture and functional verification stages for this project. Section 5 shows the prototype used in the tests and the results achieved by the project. Finally, there is a general review of the project and future works.

IPPROCESS

The design of an IP-core requires a high-added knowledge and involves different working groups, which perform the specification, implementation using a hardware description language (HDL), simulation, functional verification, synthesis, prototyping and protection of intellectual property. All these aspects of the project requires many skills in different areas, as well as mechanisms to support the teamwork [4,7].

The development of an IP-core should allow its interaction with other components to form a System-on-Chip (SoC). Therefore, it's essential that each component has the smallest faults as possible [4, 5, 7].

The ipPROCESS is a process of IP-cores development with FPGA prototyping which provides a disciplined approach, assigning tasks and responsibilities within the context of an organized development. The goal is to ensure the production of high quality modules that meet the needs of its end users, while meeting the demands of a competitive design market [5].

The ipPROCESS was based on the Rational Unified Process (RUP), a software engineering process that provides an approach to assign tasks and responsibilities, where a "discipline" is a related collection works. It also surrounds a set of practices, important in the context of reuse and detection errors, such as requirements management and changes in visual modeling with UML, use of component architectures and constant quality check [1, 4, 7].

Some techniques used in eXtreme Programing (XP) were placed in ipPROCESS. His emphasis on teamwork, with managers, developers and customers as part of a dedicated team to product’s quality and its focus on automation, validation and use of a regression test suite, allow the ipPROCESS application a relatively simple set of rules and practices related to planning, design, coding and testing [8].

The phases of ipPROCESS are shown in Figure 1. The first phase is the Conception. At this point, the goal is to utilize the customer understanding needs to recognize the functional and nonfunctional requirements. At this step of the process, the scope must be defined.

In the Architecture phase is used the survey requirements made in the previous phase to design the structure of the IP-core. Each requirement is mapped to one or more components then are created communication protocols between them. Still in the Architecture, plans for verifying and integration between the components must be documented.

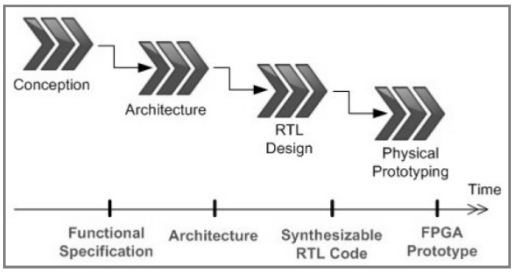


Figure 1 – Phases of ipPROCESS [5].

The third phase is called RTL Design. In this step the conceptions of design, defined previously, are used for deployment with hardware description languages (HDL). The coding is done based on the strategies for verification and integration of modules documented.

Finally, the last phase is the Physical Prototyping. At this stage, the IP-core physical design should be done. All components must be synthesized and tested, information about the FPGA used should be documented, and the steps needed for the prototype have to be reproduced.

METHODOLOGY

In this paper we applied the methodology of ipPROCESS and some artifacts were produced according to the discipline process. The making of these products is dynamic, with each IP-Core update the documents are reviewed and refined.

The ipPROCESS workflow must meet the stages characteristics of development. Figure 2 illustrates how this flow should occur. The phases of the process can easily be identified in the sequence in which courses should be conducted.

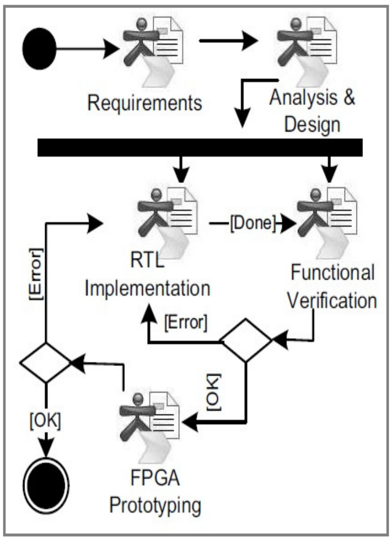


Figure 2 – ipPROCESS Workflow [3].

DEVELOPMENT

In this section, the development process will be briefly described using the ipPROCESS's phases for structure it. Because of the reduced space in this paper, this description phases haven't the due attention, but detailed explanations can be founded in the referenced works [3, 4, 10].

Conception

Early in the project, three documents of the Requirements discipline were produced. The first was the glossary, it's used to identify terms and expressions and to establish a unique understanding of the project among its members.

Another artifact was the Vision Document, which provides an initial view of the IP-Core scope, defining its characteristics and constraints, besides identifying which users should be involved in each process stage.

Architecture

This step is the conversion of the ideas into an architectural model. During this phase are identified the architecture components, once considered the requirements of IP-core. The elements are modeled in class diagrams and use cases. In some cases it is also possible to describe the processes through finite state machines (state diagrams) [2]. The Figure 3 presents a Class Diagram, produced at this step.

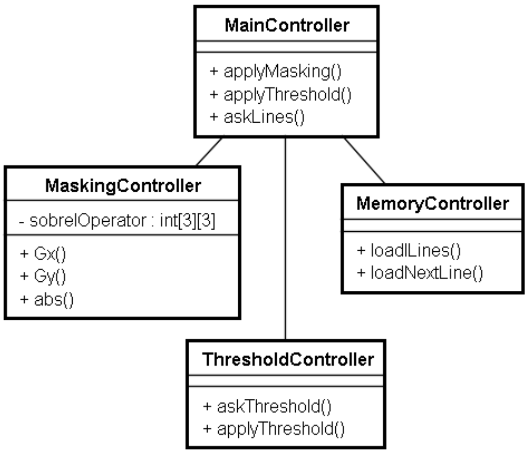


Figure 3 – Class Diagram.

The Use Case model defines the actors involved, the flow of primary and secondary events, special requirements, preconditions and postconditions, nonfunctional requirements and associated extension points. The Figure 4 presents a version of Use Case produced on the development process.

RTL Design and Physical Prototyping

In this step, the produced artifacts were the RTL simulation model and the tests description for each component implemented.

The RTL design was described in Verilog, which is a hardware description language. The project was embedded in a kit DE2, which offers several hardware resources such as SDRAM, a FPGA IC Altera Cyclone II

2C35, analog digital conversion module for the camera input signal and VGA output module.

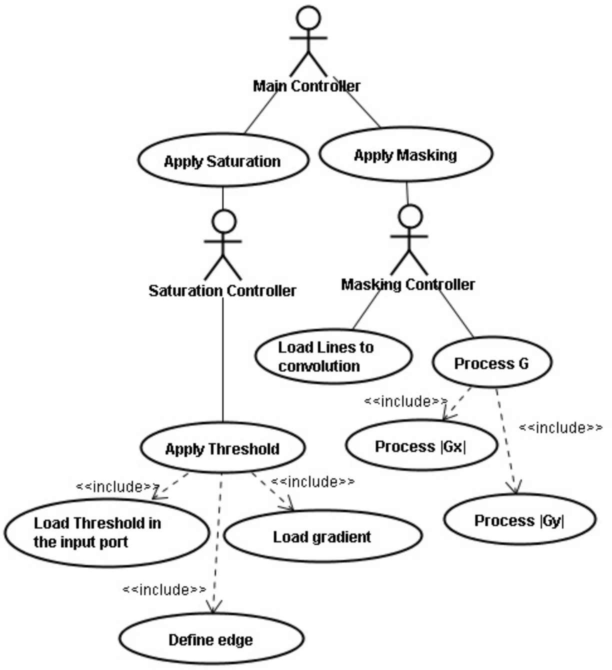


Figure 4 – Use Case Diagram.

The development was focused on the implementation of the Sobel's operator [6]. In Figure 5 you can see the overall system architecture, highlighting the region that includes the CONTROL, CONVOLUTION and BUFFER LINE modules, that are effectively developed, the other blocks are provided by Terasic, that made the kit and authorizes the use and modification for non-commercial applications.

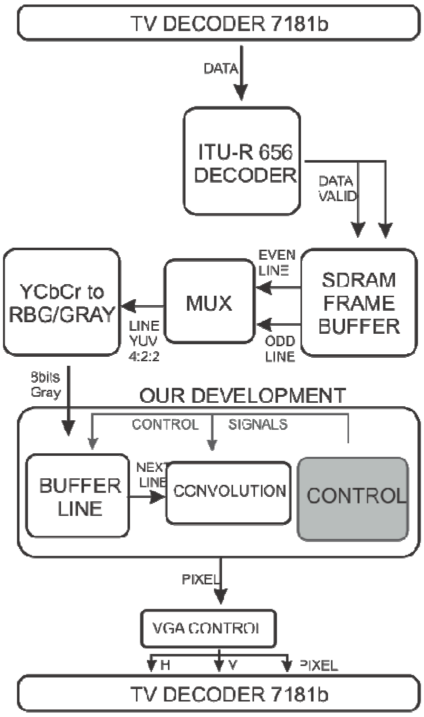
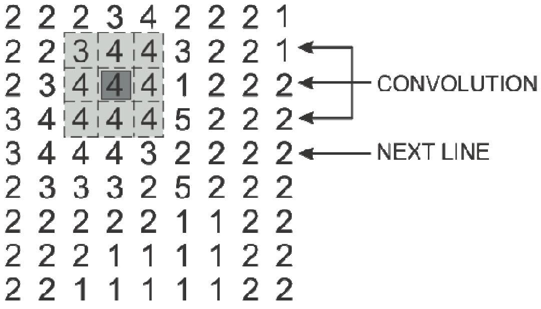


Figure 5 – Overall System Architecture.

The approach aims to decrease SDRAM memory access. The Sobel operator uses a 3x3 mask to perform a convolution of the image. This calculation is done iteratively. Therefore, the CONVOLUTION block processes only three lines of the image at every step. The line immediately below, which will be the next line to be calculated, it is loaded to the LINE BUFFER block in parallel, Figure 6 illustrates this process.



2 2 2 3 4 2 2 2 1
2 2 3 4 4 3 2 2 1
2 3 4 4 4 1 2 2 2
3 4 4 4 4 5 2 2 2
3 4 4 4 3 2 2 2 2
2 3 3 3 2 5 2 2 2
2 2 2 2 2 1 1 2 2
2 2 2 1 1 1 1 2 2
2 2 1 1 1 1 1 2 2

Figure 6 – Design Strategy Illustration.

The CONTROL module controls the entire process of defining what are the lines involved in the convolution and what is the line being loaded into BUFFER LINE. This entire process is done in parallel and synchronized with the clock generated by the ITU-R 656 DECODER, i.e., while the block is performing the convolution calculation, the next line is stored in BUFFER LINE.

Finally, after the processed pixel is sent to the module CONTROL VGA, which generate signals of horizontal and vertical sync to be shown on a VGA monitor.

Functional Verification

In the fourth stage, the RTL design is embedded in an FPGA and quality IP-Core is evaluated and estimated to validate the implementation. This validation process is called Functional Verification.

There are three types of functional verification that can be applied: the static, dynamic, and hybrid. The static (or formal) verification focuses on the code structure, like case statements incomplete, assignments, if/else inconsistent or missing signals on sensitivity list. The formal verification can prove the absence of errors through mathematical equations and verification of model properties.

The dynamic verification is coverage-driven, checking how of the design functionalities was tested, which can give a security of the completeness degree of the verification process. The constrained-random is a very used strategy in this type of verification, in which the input signals are generated based on a probability distribution function directed to the Design Under Verification (DUV).

The hybrid verification combines the static to the dynamic. The tactics used in this project is the hybrid verification using the Grey Box approach, which is a mix between the Black Box, where the verification process sees the DUV as a function, deriving the outputs from the inputs, and White Box, where the DUV variables and internal signals are visible and can be accessed by the verification environment.

The reference models were programmed in System Verilog language, using the same documentation as applied to the RTL design, thus producing similar modules. The DUV could be tested in an iterative form, where corrections were made in the project and tested again until they reached satisfactory levels.

RESULTS

The designed IP-core was embedded in a kit of Terasic DE2, connected to a mini-CCD camera-SH ¼ 0.1Lux, to test its operation. Figure 7 shows the screen of a VGA monitor with the result of Sobel filter. In the Figure 8 can be viewed all the equipment up and running. The processing is done in real time, because processing of each pixel is accomplished in one clock cycle. In tests the frame rate was around 60 frames per second, this number is limited by the capacity of the camera and TV decoder DECODER 7181B.

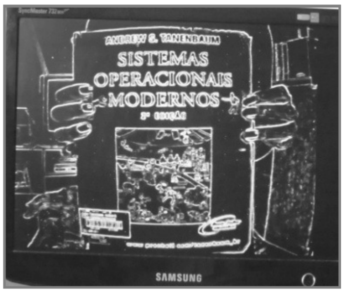


Figure 7 – Frames processed.

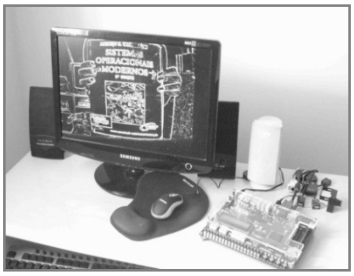


Figure 8 – Working System.

CONCLUSION AND FUTURE WORK

The use of ipPROCESS methodology provides an important documentation, and a series of well defined activities streamlines the design and decreases errors. The implementation is facilitated, since the requirements are well defined and the elaborate diagrams. The project can be divided into a great team without causing problems at the time of integration. The documentation produced decreases the learning time of students who will continue the project development. In addition, the results achieved by RTL simulation model and the functional verification, are in agreement with expectations.

This phase of the project is not finished yet. The analog camera generates image noise, which will be reduced by a pre-filter. It will also be made a qualitative analysis of the resulting image, as well as a new functional verification for IP-core developed.

The objective of this project aims to build a camera to be used in a robot. In future work, the analog camera will be replaced by a digital, the IP-core developed will be used as an initial stage of the embedded vision system that will result in the coordinates of objects in the captured image.

REFERENCES

[1] H. Chang et al, *Surviving the SoC Revolutions: a guide to platform-based design*, Kluwer Academic Press, 1999.

[2] M. Lima and others, “ipPROCESS: A Development Process for Soft IP-core with Prototyping in FPGA”, *Forum on Specification & Design Language*, Lausanne, Switzerland, pp. 1-13, 2005.

[3] M. Lima and others, “ipPROCESS: Using a Process to Teach IP-core Development”, *International Conference on Microelectronic Systems*, Anaheim, California, EUA, pp. 1-2, 2005.

[4] M. Keating and P. Bricaud, “Reuse Methodology Manual for System-on-a-Chip Designs”, *Kluwer Academic Publishers*, 2002.

[5] Brazil-IP, Rede Brasileira de Centros de Concepção de Sistemas Digitais e IP´s, URL: http://www.brazilip.org.br/. Acesso em 26 de junho de 2011.

[6] R.D. Gonzales and R.E. Woods, *Digital Image Processing*, Edgar Blücher, São Paulo, 2000.

[7] K.R.G. da Silva, “Uma Metodologia de Verificação Funcional para Circuitos Digitais”, *Master’s thesis*, Campina Grande: UFCG, 2007.

[8] D. Wells, “eXtreme Programming: A gentle introduction”, URL: http://www.extremeprogramming.org/, Acesso em 26 de junho de 2011.

[9] Terasic URL: http://www.terasic.com.tw/en/. Acesso em 26 de junho de 2011

[10] F. C. Branco and others, “Uso de Ferramentas e Metodologias no Projeto de um Sistema Digital!, unpublished.