Model Based Development of the Digital Part of a RFID Transponder with Xilinx System Generator for a FPGA Platform

Venkata Rajesh Pamula
ESAT, Katholiek Universetiet Leuven, Belgium
pamula@imec.be

Wolfram Strauß
Fraunhofer IIS Erlangen, Germany
Wolfram.Strauss@iis.fraunhofer.de

Josef Bernhard
Fraunhofer IIS Erlangen, Germany
Josef.Bernhard@iis.fraunhofer.de

Abstract – **For the development of an RFID system a flexible counterpart on the transponder side is necessary. On the system it must be possible to implement different standards and allow access to major interfaces of the processing chain of the tag. In the development of new RFID approaches a flexible platform close to the IC is necessary, thus an implementation on an FPGA is beneficial. A model-based approach allows the development of complex systems, starting with system simulation in Simulink/MATLAB in conjunction with additional Xilinx tools. The target bitfile is created from the System Generator toolbox for Simulink/MATLAB. A software defined radio platform based on a Xilinx Virtex-4 FPGA was used as a target platform. A specific and well-tested framework is used to facilitate the integration process. The mandatory EPC specification parts of the digital portion of an UHF transponder are implemented and the FPGA performance is tested towards the initial Simulink/MATLAB simulation.**

I. Introduction

Extensive usage of the Radio Frequency Identification Technology (RFID) has become a routine aspect of everyday life. Today, RFID is widely deployed in logistics, animal and human tracking, contactless readout in scientific experiments and even in national security in the form of biometric passports [1]. Figure 1 shows a typical RFID system, which essentially consist of a reader and a transponder, also known as tag, which communicate with each other through electromagnetic radiation using a specific protocol for data exchange. These systems are usually classified according to the electromagnetic spectrum they use for the communication. Many of the RFID systems have requirements that are defined on ad hoc basis. For example, the requirements posed by an application like contactless readout might be entirely different from that of a logistic tracking. There have been several attempts to standardize the RFID hardware and protocol for the data exchange to increase the interoperability. As a result several competing standards exist today such as ISO/IEC JTC1 [2], ANSI [3] and EPC [4]. However majority of the RFID systems are deployed in closed loop applications using either proprietary protocols or non-intersecting standards with non-reusable readers and transponders [5]. As a result, in most applications, RFID transponder and reader hardware and software must be specifically designed for each particular application, and must be physically modified or re-designed every time the specification for the current application is adjusted, as new applications are introduced, and/or the standards are modified or new standards are developed. This keeps the overall design time long and the system costs high [5]. This creates the necessity for having a flexible system for the transponder hardware, which can be customized depending upon the application. The system should have the characteristics of being adaptable to different standards and close to integrated circuit implementation there by making the option of implementation on FPGA attractive. This also enables extending and improving existing standards to meet the customer needs and reduce the time to market.

Traditional FPGA implementations rely on describing the hardware in one of the hardware description languages (HDL) like Verilog or VHDL and use appropriate tools to translate the design on to FPGA. This approach is still the most widely used in current day industry. However, with the advancements in semiconductor technology, with ever-increasing complexity of the systems, a system level design approach is highly desirable. MATLAB/Simulink provides the flexibility of describing the system by using a combination of graphical and language based methods which make MATLAB easy-to-use environment for the development and debugging of sophisticated algorithms [6].

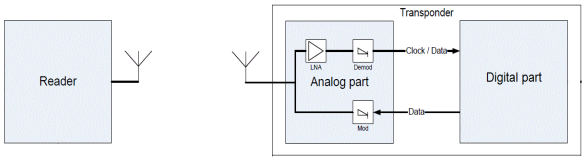

FIGURE 1 – TYPICAL RFID SYSTEM CONSISTING OF A READER AND A TRANSPONDER

Logic synthesis from high-level languages is one of the widely studied areas in recent times. Tools that are capable of converting C/C++ and MATLAB descriptions to Verilog and VHDL have allowed the designers to describe the systems at a much abstract level, thereby reducing the design time. Examples for such tools include Catapult C, SPARK, ROCCC and DWARV (C to VHDL) and System Generator and AccelDSP from Xilinx for MATLAB [7]. This approach is very much preferred by the systems engineers as they do not have to get into the details of the actual hardware description and can build the systems starting with a high level specification and this is the key motivation for the current work. [8-10] have studied the application of such an approach to the design of digital controllers and filters. In the current work, we demonstrate the suitability of the approach to design the systems, which are more control path intensive than data path.

In this paper, we discuss the architecture of a ultra high frequency (UHF) RFID transponder adhering to EPC standard that has been developed using Xilinx system generator tool box in conjunction with SIMULINK/MATLAB. The transponder thus developed has been synthesized for a Xilinx Virtex-4 FPGA and is tested for the standard compliance and the results are presented.

II. Architecture

In the framework of EPC standard, a transponder can be viewed as a customized processing unit, which receives a set of well-defined commands from the reader and responds to them as specified by the standard. For a transponder to be conformant to the standard, it needs to respond appropriately to the mandatory clauses specified in the standard. The transponder is described as a finite state machine (FSM) as shown in Figure 2 [4], whose response depends on the current state and the command received. To achieve error resilience, the standard specifies that the commands issued by the reader and in some cases, the response of the transponder, such as the reply to **ACK** command are protected by incorporating a cyclic redundancy

978-0-7695-4813-5/12 $26.00 © 2012 IEEE
DOI 10.1109/RFID.2012.29

check (CRC). EPC specification uses CRC-16 and CRC-5 for different commands. This necessitates the requirements of a CRC computation engine on transponder. The transponder stores the information in its memory, which can be accessed and modified by the reader through certain commands like **READ** and **WRITE** respectively. The size of the memory is very much application dependent. In the current work, the memory of the transponder is restricted to 64 bytes, which is adequate to identify the transponders uniquely in a population of transponders.

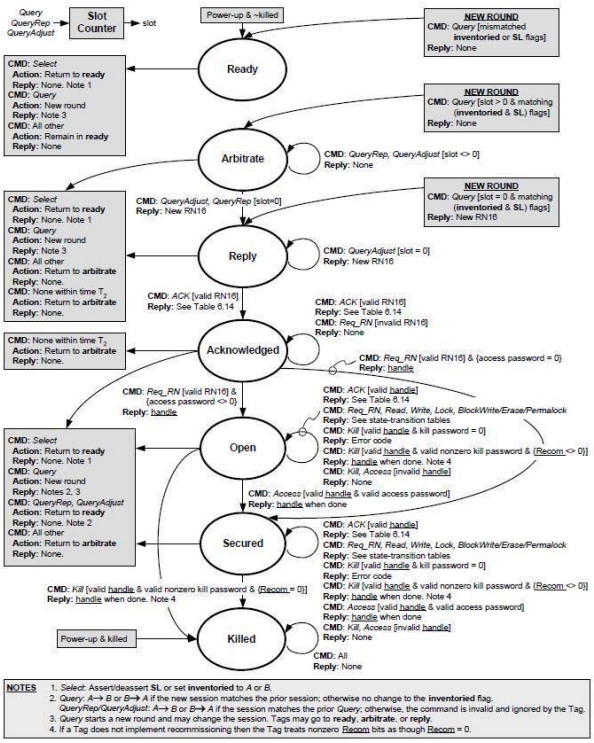

FIGURE 2 – EPC RFID TRANSPONDER STATE MACHINE

In the context of the current work, since the digital part of the transponder is of the interest, the interface between the reader and the transponder has been modified from that as shown in Figure 1 to the one in Figure 3. The underlying significance is that the transponder running on a FPGA can be directly interfaced to the reader running on a of the shelf digital signal processor.

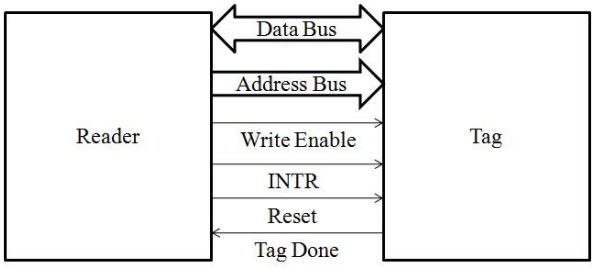

FIGURE 3 – READER-TRANSPONDER INTERFACE

In the presence of an analog front end, the interfacing is between the digital part of the analog front end and the FPGA. In the current work, the interface between the reader and the transponder is realized using a shared memory. The reader writes the command and data into the shared memory and initiates the processing on the transponder by raising an interrupt signal (**INTR**). In response the transponder does the necessary processing based on the command and replies back to the reader by writing the response in the shared memory and interrupting the reader through **Tag Done** signal. An asynchronous reset is provided to the reader, which can be used to reset the transponder.

Based on the foregoing discussion, the architecture of the transponder can be conceptualized as a combination of shared memory interface for reader-transponder interactions, a data processing unit (DPU) to interpret and process the commands received, an internal memory to store the data and a CRC engine, which is shown in Figure 4. Note that the numbers associated with each signal indicate its bit width.

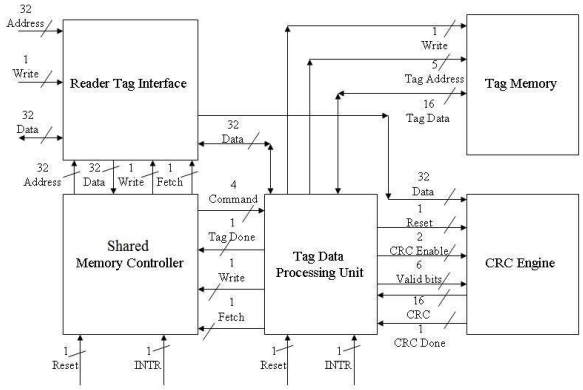

FIGURE 4 – HIGH LEVEL ARCHITECTURE OF THE DIGITAL PART OF EPC RFID TRANSPONDER

III. IMPLEMENTATION USING XILINX SYSTEM GENERATOR

MATLAB/Simulink provides a graphical environment for creating and modeling dynamic systems. System Generator provides a Simulink library called the Xilinx blockset and software to translate a Simulink model into a hardware realization of the model [11]. The blockset provides the typical functional blocks ranging from simple arithmetic operations to more complex DSP functions. Additional blocks are provided to realize control structures such as FSMs and to embed user defined intellectual property (IP) blocks.

The bottom-up approach for the design of digital systems has been employed during the development of the transponder. The individual blocks of the transponder system such as the DPU, CRC engine etc. are developed independently and tested, before interfacing with each other. This increases the ease of reusability as well as reduces the debugging and verification effort. Figure 5 shows the CRC engine implemented using the primitive blocks of the Xilinx system generator and its associated timing diagram.

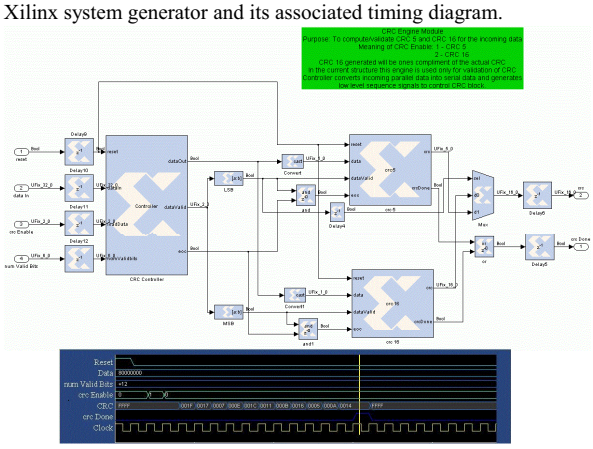

FIGURE 5 – CRC ENGINE AND TIMING DIAGRAM FOR A CRC 5 COMPUTATION

The individual blocks are then put together to form the transponder system. Register retiming technique is used to redistribute the slack in the critical path, which is realized by introducing the delay blocks in the critical path of the DPU. Figure 6

shows the complete transponder system with the timing diagram for a specific transaction from the reader. As it can be seen from the timing diagram, there is a set of four commands issued by the reader each followed by the **INTR** high, to which the transponder responds by replying with the data followed by a **Tag Done** high signal. This demonstrates the accuracy of the model developed. In the next section we compare the results obtained from the simulation with the ones obtained experimentally, by deploying the synthesized model on to a Virtex-4 FPGA.

It is appropriate to discuss the benefits of the model based approach for the development of digital systems. As previously mentioned, model based approach can reduce the development time and help the designer to develop complex digital systems with limited HDL knowledge. However, the optimum translation of the high level description into a gate-level description is not always guaranteed as it is very much dependent on the implementation of the synthesis software (in this case, System generator for MATLAB).

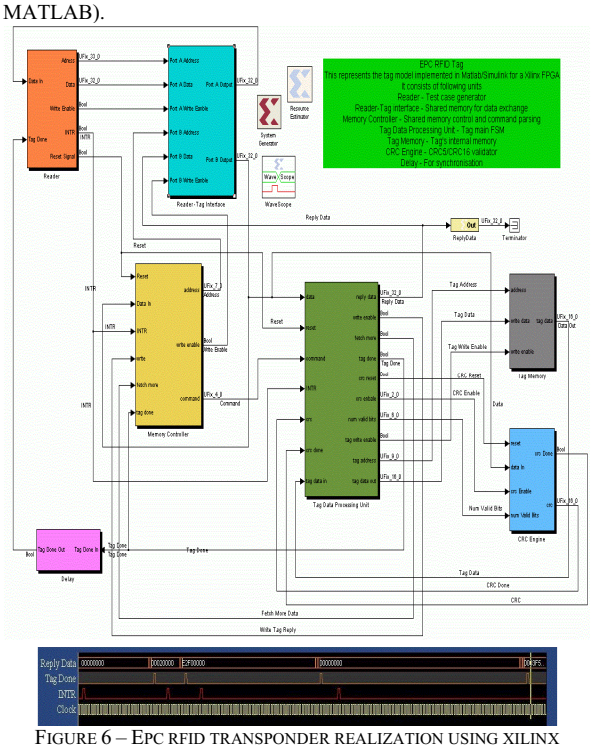

FIGURE 6 – EPC RFID TRANSPONDER REALIZATION USING XILINX SYSTEM GENERATOR AND ITS TIMING DIAGRAM FOR A TRANSACTION

IV. FPGA SIMULATION

Xilinx system generator accompanies with an interface to synthesize the SIMULINK/MATLAB model to a target FPGA architecture using Xilinx XST synthesis tool. However, in the current work, we choose to integrate the transponder model into the software defined radio (SDR) platform at the Fraunhofer Institute [12]. In order to achieve this objective, the model is compiled to generate the corresponding VHDL description and the netlist. The detailed procedure for the generation of the netlist is outlined in [13]. The SDR is based on a Xilinx Virtex-4 xc4vfx60-10ff1152 core as shown in Figure 7. A framework specific to this SDR is developed, which is quite flexible and allows the designer to embed into the framework, without knowing the board specific details. Also, the flexibility of SDR platform allows the possibility to optimize the platform for an application or family of applications, making the SDR platform a multi-standard terminal [14]. The transponder is integrated into the framework and synthesized to generate the bitfile, which is deployed onto the FPGA for verification.

The synthesis report shows that the transponder can run at a maximum clock frequency of 74.72 MHz while consuming the resources summarized in Table 1. The FPGA is interfaced with an external signal generator to derive the clock signal of 50 MHz and the other signals such as the command data and **INTR** are derived from the reader. The output of the transponder is observed on an Agilent logic analyzer and the results shown in Figure 8 are in accordance with the simulation, which verifies the conformance of the FPGA realization.

TABLE 1 – FPGA RESOURCE UTILIZATION

Resource	Number utilized
Slices	1607
Flipflops	548
Block RAMs	2
Lookup Tables	2441
Input/Output Buffers	32
Multipliers	0
Tristate Buffers	0

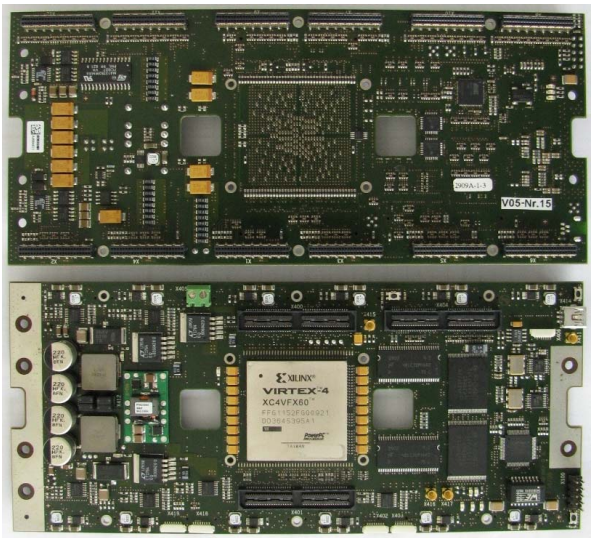

FIGURE 7 – FPGA BASED SDR AT FRAUNHOFER IIS

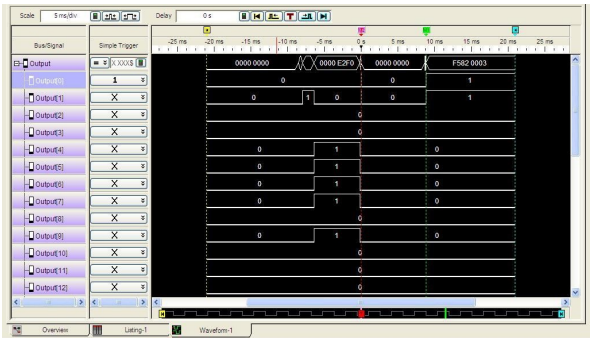

FIGURE 8 – TRANSPONDER'S RESPONSE TO A TRANSACTION FROM READER OBSERVED ON LOGIC ANALYZER

V. CONCLUSION

A model based approach for the RFID transponder for FPGA platform is presented. The transponder model developed has been simulated using SIMULINK/MATLAB in conjunction with the Xilinx system generator toolbox. The model is synthesized for a SDR platform based on Xilinx Virtex-4 FPGA and is validated against the simulated results. The approach thus reported proves the efficacy of usage of system-level abstraction of hardware-oriented programming, as an alternative to gate-level hardware descriptive language, to satisfy the conformant RFID product development at a minimal development-to-market time. However, it has to be noted

that the final hardware realization may not be really optimum in terms of gate count and other hardware resources. Further research could be done on the development of a design methodology which would result in an optimum hardware realization when described using model based approach.

ACKNOWLEDGEMENTS

The authors would like to thank Robért Glein, Alexander Hoffman and Stefan Riess of the HF department of Fraunhofer IIS, for their invaluable technical support and discussions during the development of the current work. Also the authors would like to thanks the anonymous reviewers for their suggestions and comments.

REFERENCES

[1] V Rajesh Pamula, "Model based development of the digital part of a RFID transponder with SIMULINK/MATLAB for a FPGA platform", MSc Thesis submitted to Imperial College London 2010.

[2] ISO/IEC 18000-1:2008, "Information technology – Radio frequency identification for item management – Reference architecture and definition of parameters to be standardized", 2008.

[3] ANSI/INCITS 256, "Radio Frequency Identification (RFID)", NCITS 256.

[4] EPC Radio-Frequency Identify Protocols Class-1 Generation-2 UHF RFID protocol for Communications at 860-960 MHz.

[5] Alex K. Jones, Raymond Hoare, Swapna Dontharaju, Shenchih Tung, Ralph Sprang, Joshua Fazekas, James T. Cain, and Marlin Mickle. "An automated, FPGA-based reconfigurable, low-power RFID tag", in Elsevier B.V. Microprocessors and Microsystems 31, also available on Science Direct, pages 116-134, 2006.

[6] Xilinx. "System Generator for DSP", 10.1 edition, March 2008.

[7] Arcilio J. Virginia, Yana D. Yankova, and Koen L.M. Bertels, "An empirical comparison of ANSI-C to VHDL compilers: SPARK, ROCCC and DWARV", Annual Workshop on Circuits, Systems and Signal Processing (ProRISC), Veldhoven, Netherlands, 2007.

[8] Vikas Gupta, Kavita Khare and R. P. Singh, "Efficient Design and FPGA Implementation of Digital Controller Using Xilinx SysGen", International Journal of Electronics Engineering, 2(1), 2010.

[9] Behzad Behnam and Masoud Mansouryar, "Modeling and simulation of a DC motor control system with digital PID controller and encoder in FPGA using Xilinx system generator", 2nd International Conference on Instrumentation, Control and Automation, 2011.

[10] Hasan S, Boussakta A and Yakovlev A, "Improved parameterized efficient FPGA implementations of parallel 1-D filtering algorithms using Xilinx system generator", IEEE International Symposium on Signal Processing and Information Technology, 2010.

[11] Xilinx, "Xilinx System Generator v2.1 for Simulink", March 2008.

[12] Software Defined Radio at Fraunhofer Institute for Integrated Circuits, available online at http://www.iis.fraunhofer.de/en/bf/ec/dk/sdr/index.jsp.

[13] Xilinx. "System Generator for DSP Reference Guide", release 10.1 edition, March 2008.

[14] Design Article on Software defined radio: defining the challenges, available online at http://www.eetimes.com/design/microwave-rf-design/4212244/Software-defined-radio--defining-the-challenges?pageNumber=0