Design of Digital Circuit for a Passive RFID Tag

Panagiotis Bouklis
Dept. of Electrical & Computer Engineering
Aristotle University of Thessaloniki
Thessaloniki, Greece
pmpoukli@auth.gr

Leonidas Katselas
Dept. of Electrical & Computer Engineering
Aristotle University of Thessaloniki
Thessaloniki, Greece
katselas@auth.gr

Alkis A. Hatzopoulos
Dept. of Electrical & Computer Engineering
Aristotle University of Thessaloniki
Thessaloniki, Greece
alkis@eng.auth.gr

***Abstract*—The low power design of the digital block for a passive RFID tag, conforming to the ISO/IEC 15693 standard, is presented in this work. The fundamental digital design flow and typical low-power techniques are discussed. These techniques were efficiently exploited in this design to reduce the power consumption of the block about 95% with respect to the initial design, down to 10 μW.**

***Index Terms*—application specific integrated circuits, design methodology, high level synthesis, RFID tags**

I. Introduction

In an RFID application there are two entities: the RFID reader, placed at the physical place that requires identification, and the RFID tag, the integrated circuit with unique stored data that help identify the object. Various types of RFID tags and standards have been introduced. Among them, is the ISO/IEC 15693 standard, which sets the requirements for RFID systems with passive RFID tags that operate at 13.56 MHz, harvesting energy from the electromagnetic field of the RFID reader, and can be identified from a distance of up to 1.5 m. The integrated circuit of the tag is placed on a ID-1 formatted card [1], conforming to ISO/IEC 7810 standard, and therefore, is ideal for the use of services requiring person identification.

An RFID tag consists of the antenna, an analog and a digital part. The analog part demodulates the incoming signal, generates the main clock, harvests energy from the electromagnetic fields, and supplies this energy to the digital part. Then, when the digital part is ready to send data, the analog part modulates and finally transmits the response signal. The digital part reads and decodes the incoming data, executes the request commands, decides when the tag has to respond and what the response data should be, and then encodes them according the coding scheme selected by the reader. A design for the digital part of an RFID tag is presented.

II. Digital Design Flow

For every integrated circuit design there are some main steps that have always to be followed. These steps, illustrated in Fig. 1, may be repeated until the design meets the requirements set. Starting from the system specification, where the functionality of the system is described, the design is processed until the important step of finally manufacturing the chip is reached. [3]

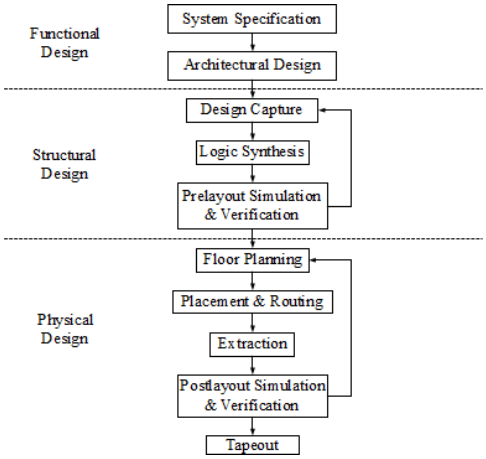

Fig. 1. Digital design flow steps

III. Functional Design

The first step of digital design is to capture the system requirements and design the main system architecture. In the case of this RFID tag, the general functional requirements can be extracted by the ISO/IEC 15693 standard.

A. System Specification

Some of the key requirements for the design of this RFID tag are [1][2]:

- Incoming data coding uses 1 out of 4 or 1 out of 256 pulse position modulation.
- CRC16 (Cyclic Redundancy Check) is used for every signal sent or received.
- Every request made to the tag starts with a SOF (Start Of Frame) and ends with an EOF (End Of Frame). Also flags, command code, CRC and command-specific data are set according to a specific format. Response data format is also specified for every command.
- Commands for inventory with slot-based and mask anticollision algorithm, select, quiet, fetching system

information, read, write and lock data are specified. The commands can be addressed to a specific tag according to its UID (Unique ID), or un-addressed.

- Response data can be transmitted with two different methods, with ASK modulation on a 423.75 kHz subcarrier, or with FSK modulation using a Manchester code and two subcarriers, 423.75 kHz and 484.25 kHz. Also two different data-rates, low and high, are specified for each method.
- One important constraint that was set in the system design, is that the total system power should not exceed 11 μW.

B. *Architectural Design*

Before starting the actual design of the circuit the system architecture is presented, as in Fig.2, and certain functions are assigned to different subsystems.

Due to low power requirements, a globally asynchronous-locally synchronous scheme has been used. Each subsystem has its own clock domain, while using asynchronous enable and ready signals for the communication with other subsystems.

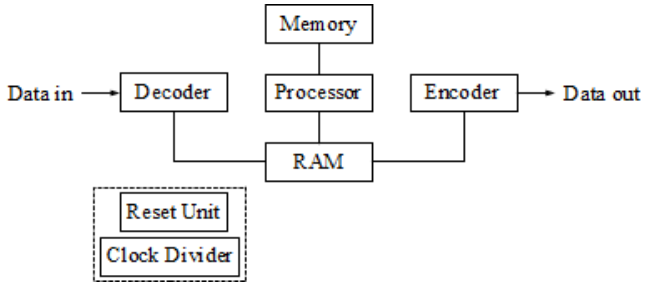

Fig. 2. System architecture

This system consists of:

1) Decoder

Detects the SOF and EOF, the coding method and decodes the incoming signal to bytes, saving these data to the RAM.

2) Processor

Reads and checks the integrity of the data saved by the decoder. Decides if the tag should execute the command that has been received and, if so, executes it and prepares the data to be transmitted, saving them to the RAM.

3) Encoder

Reads the response data and the encoding scheme that the reader requested. Then it encodes the data and sends them to the external modulator for transmission.

4) Reset unit

Controls the reset signals of the subsystems.

5) Clock divider

Generates the different clocks needed by the subsystems, by dividing the 13.56 MHz clock that is extracted from the received signal.

6) RAM

Volatile memory that stores received and to-be-transmitted data.

7) Memory

Non-volatile memory of the tag that stores data like UID, AFI (Application Family Identifier) and DSFID (Data Structure Format Identifier). This module usually is an EEPROM or Flash Memory with write functionality. However the write commands of the ISO/IEC 15693 standard are optional, so a ROM can be also used. In this design, although write commands can be executed by the system, a custom ROM has been used, as for the time of the design, EEPROM IP was not available for the technology manufacturing process. The write-command outputs have been connected to output pins of the design for verification purposes.

IV. HDL Design

Each subsystem's functionality is analyzed and then the subsystem is decomposed into components. Each component is described using a HDL, like VHDL in this design. Each component is then tested and verified to meet the requirements, the components are put together, being tested and verified again, and at last the subsystems are put together to form the final system to be synthesized, tested and verified. The steps of this process are continuously repeated until the system meets all the specified requirements.

Before proceeding to synthesis, the code is analyzed and optimized using a CAD tool and then simulated & verified. Tools used for this process are Cadence NCLaunch and SimVision for the simulation and functional verification, and Cadence Incisive HAL Analysis, for the code analysis and optimization.

A. *Decoder*

The Decoder consists of the following components:

1) decoder: Decodes data saving them to the RAM and detects the EOF.

2) sof_detect: Detects the SOF and the coding scheme.

B. *Processor*

The Processor consists of the following components:

1) command_control: Detects the request flags and the command, controls the CRC check procedure, and decides if the request should be executed, taking into account the flags, the data integrity and the UID when the request is addressed.

2) command_exec: Executes the command and prepares the response data, saving them to the RAM.

3) compare: Upon an inventory request, this unit compares the request parameters, slot and mask, with the tag's stored data.

4) CRC: CRC16 decoder and encoder.

5) tag_state: Stores the tag's current state.

C. *Encoder*

The Encoder consists of the following components:

1) bit_encoder: Encodes each data bit that is going to be transmitted.

2) data_out_mux: Controls the output data according to the signals received by the rest of the subsystem's components.

3) encoder_ctrl: Central control unit of the subsystem.

4) eof_encoder: Encodes the EOF.

5) pulses: Generates the pulses that need to be transmitted.

6) sof_encoder: Encodes the SOF.

V. Low Power Techniques

Up to this point the design has been tested and the code meets the functional requirements. Low power constraints though make the use of low power techniques critical. The total chip power consumption depends on many variables, however an estimation can be expressed as in (1) where [4]:

P_{total} = Chip's total power consumption

P_{block} = Full block's total power consumption

P_{core} = Core cell's total power consumption

P_{clock} = Clock tree's total power consumption

P_{io} = I/O cell's total power consumption

P_{dc} = DC current's total related power consumption

$P_{leakage}$ = Leakage power consumption

$$P_{total} = P_{block} + P_{core} + P_{clock} + P_{io} + P_{dc} + P_{leakage} \quad (1)$$

A. Low power coding

The core cell's total power consumption can be estimated as in (2) where [4]:

P_{core} = Core cell power consumption

$1/2 x C_{net(i)} x V_{dd}^2$ = Switching power associated with the related net (i)

C_{net} = Total net capacitance

V_{dd} = Supply voltage

$E_{i(i)}$ = Internal power associated with the related net (i)

$F_{t(i)}$ = Toggle count associated with related net (i), during normal operation.

$$P_{core} = \sum_{\forall net(i)} \left(1/2 \times C_{net(i)} \times V_{dd}^2 + E_{i(i)}\right) \times F_{t(i)} \quad (2)$$

During the code description of the design, the last operand of (2) can be affected. Special care should be taken so that signal transitions, which have no functional purpose or effect on a component's output, are eliminated.

Based on the same rule, gray-type coding should be used for FSM (Finite State Machines) and counters.

Clock frequency will also affect the core power. The ideal, as for power consumption, clocks for each subsystem are specified, according to the local requirements:

- 847.5 kHz clock for the Decoder
- 105.9375 kHz clock for the Processor and the Memory
- 6.78 MHz clock for the Encoder
- RAM is in general asynchronous, but encompasses a synchronizer component to be synchronized with the specific subsystem that requests access.

B. Resource sharing

In some cases it is possible for the subsystems of a design to share resources. This way parts of the circuit which have no functionality at a specific moment but consume power are minimized. In this design resource sharing was possible, like counters in the Decoder and Encoder subsystems.

C. Operand and input isolation

Operand isolation is based on the principle that an operand should be stopped from feeding the input of an arithmetic unit, unless the output is required. In Fig. 3 inputs a, b and c are isolated using an enable signal, which determines when the output is required.

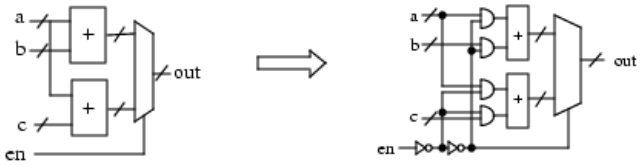

Fig. 3. Operand isolation [5]

On the same principle, the component inputs can be isolated, unless their output is required. This adds extra logic, but net and component signals' toggle rates are minimized, reducing the total power. In this design, input isolation has been applied to the Decoder, for the feeding of the input to the subsystem's components, and to the Processor and Encoder, mainly for the feeding of RAM-data signals to the local components.

D. Clock gating

One of the most effective ways to reduce power is clock gating, which adds extra logic to prune the clock tree and disable parts of the circuit, so that flip-flops do not switch states, as in Fig. 4. With clock gating, only leakage currents are incurred, and the total power dissipation can be significantly reduced. Clock gating uses enable conditions which have to be present in the code. Clock gating was used in all parts of this design.

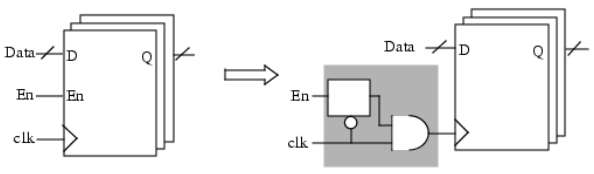

Fig. 4. Clock gating [5]

E. Other low power techniques

CAD tools may automatically optimize the dynamic and leakage power dissipation. Other low power techniques can be also applied, like Multiple Supply Voltage, Power Shutoff Methodology, DVFS (Dynamic Voltage Frequency Scaling) etc. In this design though, the power requirements have been already met with the techniques described above.

VI. Synthesis

Adding the extra logic required for the low power techniques used, the final HDL code is simulated and verified. As long as the functional requirements of the system are met, the next step is the synthesis of the design using a CAD tool. For this design, Encounter RTL Compiler has been used, using the UMC 0.18μm process, and the final synthesized design is illustrated in Fig. 5.

CAD tools can set default toggle probabilities to estimate power dissipation. For more accurate estimation, an activity file can be generated, TCF or SAIF formatted, containing the signals and nets toggle probabilities, based on a specific simulation. In this design, NClaunch was used to extract this information.

After synthesis, the synthesized HDL code is exported by the CAD tool, and once again simulated and verified to meet the system requirements. NClaunch and SimVision were used to verify the functionality of the synthesized design. Then a new activity file is generated and imported to the synthesis tool, and various reports (area, timing, power etc) are generated.

VII. Conclusion

In this design a total of 3388 cells have been synthesized with a cell area of 68451.87. The power is at the level of 10 μW, and meets the power requirements. Of this power, about 50% is consumed by the Encoder subsystem and 30% by the clock divider. The circuit percentage that has been clock gated is 80%. Without applying any low power technique, the power of the initial design was 200 μW, which means the percentage of power reduction achieved is 95%.

References

[1] Identification cards – Contactless integrated circuit(s) cards – Vicinity cards – Part 2: Radio frequency power and signal interface, ISO/IEC 15693-2, 1999

[2] Identification cards – Contactless integrated circuit(s) cards – Vicinity cards – Part 3: Anti-collision and transmission protocol, ISO/IEC 15693-3, 2000

[3] Jan M. Rabaey, Anantha Chandrkasan, Borivoje Nikolić, "Designing complex digital integrated circuits" in *Digital Integrated Circuits*, 2nd ed., USA: Prentice Hall, 2003, ch. 8, sec. 4

[4] *Faraday Standard Cell Library – FSA0L_A – 0.18μm Low Leakage Process,* Faraday Technology, Taiwan, 2005, pp. 17-18

[5] *Low Power in Encounter RTL Compiler, Product Version 10.1,* Cadence Design Systems, San Jose, CA, 2011

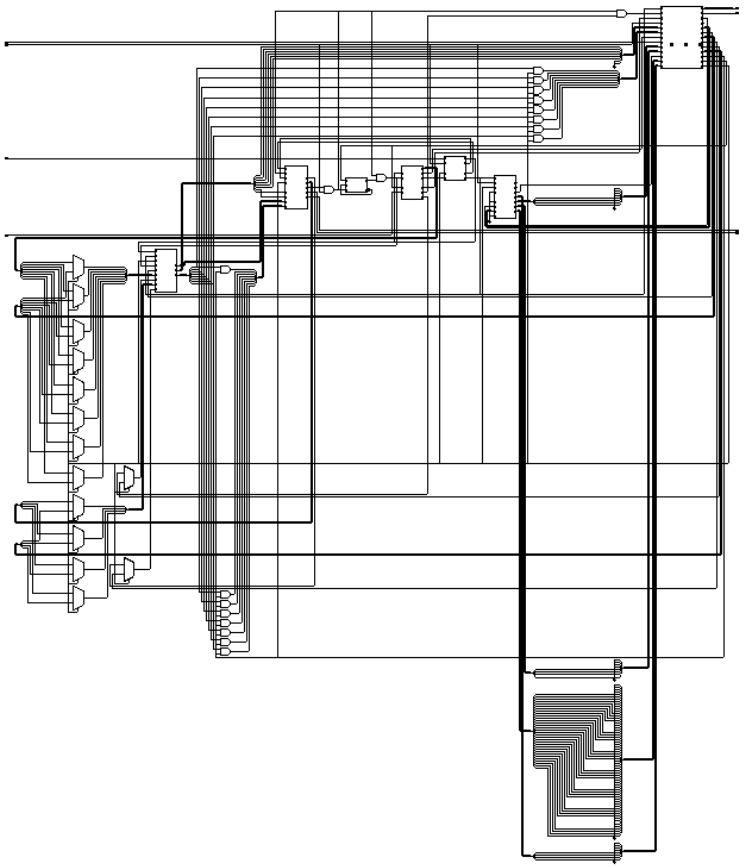

Fig. 5. Synthesized design, top module