A 915 MHz UHF Low Power RFID Tag

César Marcon[1], José Palma[2], Fabiano Hessel[1], Eduardo Bezerra[1], Guilherme Rohde[1], Carlos Reif[1], Luciano Azevedo[1], Carolina Metzler[1]

[1]PPGCC / FACIN / PUCRS
Av. Ipiranga, 6681
Porto Alegre, RS – Brazil
+55 51 3320 3611

[2]PPGC / INFORMÁTICA / UFRGS
Av. Bento Gonçalves, 9500
Porto Alegre, RS – Brazil
+55 51 3308 6166

[cesar.marcon, fabiano.hessel, eduardo.bezerra, grohde]@pucrs.br, jcspalma@inf.ufrgs.br

ABSTRACT

This paper describes the implementation of a passive low power tag, which works on 915 MHz UHF frequency. The tag implements an innovative and efficient synchronization algorithm, which deals with significant reference clock variations. A flexible design flow is proposed for the customization, verification and synthesis of the digital block, targeting low power requirements.

Categories and Subject Descriptors

B.7.1 [**Integrated Circuits**]: Types and Design Styles – *advanced architectures, algorithms implemented in hardware, VLSI (very large scale integration).*

General Terms

Performance, Design, Economics, Verification.

Keywords

RFID Systems, Synchronization Algorithms, low power consumption

1. INTRODUCTION

RFID (Radio Frequency IDentification) technology has been around since the Second World War, but only today it is becoming attractive [1]. This is motivated by technology advances and lower costs of components.

RFID systems can be used in applications such as supply chain management, tracking, and security. Several large companies have shown interest in this technology since Wall Mart's announce of start employing RFID.

RFID systems are composed of tags, readers and antennas. Tags exchange data with the reader (or interrogator) through radio frequency (RF) communication [2]. The reader has one or more antennas that send/receive radio waves to/from tags. The number of antennas depends on the application, as more antennas increase the communication zone.

Tags can be active or passive. Active tags have their own power supply, operating in high frequencies and usually are larger than passive tags. Passive tags need to be smaller and low power, since they do not have power supply. They transform the communication RF signals from reader in energy to operate. As a result, passive tags are cheaper than the active ones [3].

This paper describes a passive 915 MHz UHF low-power RFID tag and its flexible design flow. Its design flow allows command set customization according to application requirements. Tag area and power can be optimized, once the tag is implemented with a reduced command set. Moreover, this paper proposes a synchronization algorithm, which allows data decoding even with significant reference clock variation during the communication between tag and reader.

This work focuses the implementation and validation of the tag's digital block. The tag, as a whole, is class 1 generation 2, according to ISO 18000-6B and EPC standards.

This paper is organized as follows. Section 2 presents related works in RFID systems. Section 3 presents the structure of the proposed tag. Section 4 describes an innovative synchronization algorithm used in the communication between reader and tag. Section 5 discusses the flexible design flow. Section 6 describes the system validation. Section 7 shows synthesis results and Section 8 presents conclusions.

2. RELATED WORK

RFID technology is back to the research agenda as a result of new application domains and costs reduction (e.g. supply chain). Hassan and Chatterjee [4] propose a new taxonomy to classify RFID systems. Choi et al. [5] present a 13.56 MHz RFID reader for home security applications, which allows multi-tag recognition. Leong et al. [6] propose an approach to synchronize multiples RFID readers in order to enable successful dense RFID reader deployment, and minimize the reader collisions.

An important issue on RFID systems is the reading distance, typically limited to less than 1.2 m. Karthaus and Fisher [7] propose a passive UHF transponder with reading distance increased to 4.5 m by using a different voltage generator, a PSK in backward link, and a careful layout and antenna matching.

The low power consumption is the major concern in RFID systems design. Leenaerts [8] discusses the low power techniques associated to radio frequency IC design. Vacherand [9] presents a new approach to improve the low power performances of RFID chips. Porret [10] discuss low power and low voltage trade-offs for CMOS RF design. Jones [11] presents an ultra low power active RFID tag system and an automated flow for tag controller design.

Guo [12] introduces a new device called OCA (on-chip-antenna), and claims that is the best approach compared to an off-chip antenna. However, it faces many challenges in realizing OCA for a passive RFID tag because the efficiency of the antenna is limited by its allowable area.

Similarly to [8], [9], [10] and [11], this work presents the design of a flexible low power RFID tag. The main contribution is the flexible tag design aiming to reduce the power consumption. In addition, an efficient synchronize algorithm is proposed.

3. PROPOSED ARCHITECTURE

Figure 1 shows the basic diagram of RFID tag architecture, which has three main blocks: External Antenna, Analog Block and Digital Block.

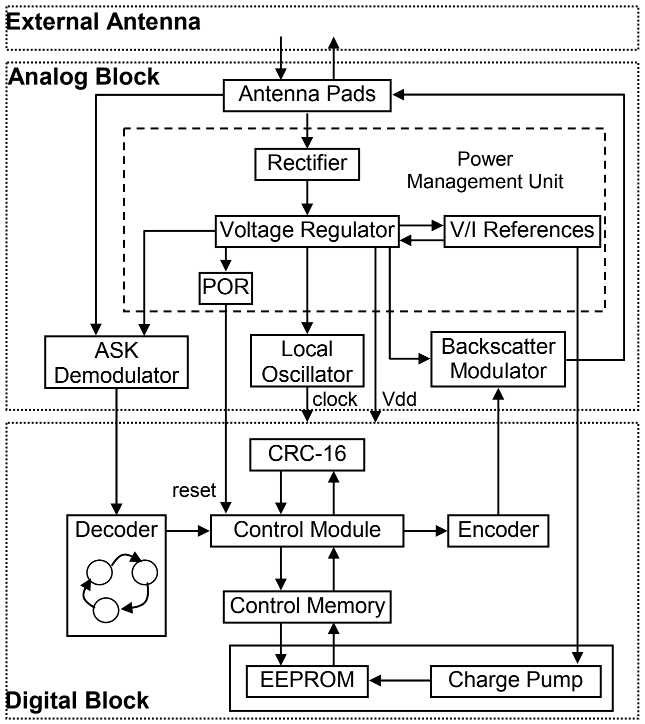

Figure 1 – RFID tag architecture diagram.

The External Antenna interfaces the IC tag to the environment, and it is responsible for capturing data as electromagnetic waves. The tag power supply is implemented by the Analog Block that converts the electromagnetic waves into electric energy, which is stored in a bank of capacitors. In addition, this block is responsible for analog and digital signals demodulation/modulation. Finally, the Digital Block performs all tag operations, such as data decoding, commands interpreting, collision arbitration, sent data encoding and memory access. The Decoder module, in the Digital block, implements the synchronization algorithm, which is an important contribution of this work.

3.1 Analog Block

The Analog Block has five modules: (*i*) Antenna Pads, which interfaces the tag IC to the tag External Antenna; (*ii*) Power Management Unit, that manages the IC start up and supplies a regulated voltage and current to the Digital Block; (*iii*) Amplitude Shift Keying (ASK) Demodulator that receives incoming messages from this Antenna Pads module and outputs digital messages, including preambles and cyclic redundancy check (CRC) bits, to the Digital Block; (*iv*) Backscatter Modulator, that varies IC impedance, in order to achieve backscatter response (backward link); and (*v*) Local Oscillator, that generates a 640 kHz reference clock for Digital Block operation.

The Power Management Unit also supplies voltage and current references, including EEPROM band gap voltage. It is made of four modules: Power-on Reset (POR), Rectifier, Voltage/Current (VI) Reference and Voltage Regulator.

The Backscatter Modulator receives control signals from the Digital Block to control the antenna's impedance, which is achieved by switching a capacitive load. The reader detects impedance variation enabling the backscatter communication.

The Local Oscillator module has a complex design since the reference clock has to be generated through analog circuits, which are high dependent on the tag power consumption variation. This dependence implies up to 10% of clock variation during a single period of clock. To mitigate this effect, oscillation frequency is generated as low as possible, and some modules are enabled only when needed.

3.2 Digital Block

The Digital Block performs data decoding and encoding, collision arbitration, and non-volatile memory control. Six modules described next perform these functionalities.

3.2.1 Decoder

The Decoder receives from the ASK Demodulator a 40 Kbps or 160 Kbps Manchester encoded data, which is decoded to a binary format that is sent to the Control Module. The data decoding is done with pairs of bits, according to the Manchester method. When a transition from ‘1’ to ‘0’ is detected, a bit in logic ‘1’ is sent to the Control Module. Similarly, if a transition is from ‘0’ to ‘1’, a bit in logic ‘0’ is sent to the Control Module. In the case of transition absence, a Manchester error is detected and an error signal is set.

Before starting data decoding, it is necessary to synchronize the input signal with a 640 KHz clock provided by the Local Oscillator module. This task is accomplished through the synchronization algorithm, described in section 4.

3.2.2 Control Module

The Reader sends commands to the tag that are identified and handled by the Control Module. As a result, control signals to other digital modules. For instance, Control Module controls EEPROM module, performing read, write and lock actions.

The Control Module functionality is based on two state machines. The first one is responsible for collision arbitration and command detection. The second one is responsible for command treatment, controlling all digital modules during the command execution.

3.2.3 CRC-16 Module

The CRC-16 module creates a 16-bit cyclic redundancy check of a new packet and checks the integrity of a packet received from the reader. When a CRC error is detected, all data are discarded and the frame is suspended. The polynomial used to calculate the CRC-16 is $X^{16} + X^{12} + X^5 + 1$, which is the CRC-CCITT International Standard.

3.2.4 Control Memory

The Control Memory module is the interface between the Control Module and the EEPROM Memory. It is responsible for enabling the memory access and for returning the instruction results.

The Memory module is 2-byte addressed. However, commands that perform write and read operations are byte addressed. This architecture could result in waste of memory space, and in an increase of the power consumption and access time. In order to solve this problem, an extra input control signal was added to this module. Since every memory position has 2 bytes, this signal identifies in what byte (left or right) the write/read command must be accomplished.

3.2.5 EEPROM Memory

The tag Digital Block has an internal 512 bits EEPROM, arranged in 8 rows of 4 words.

The memory operation requires first to erase a word before writing it, implying extra power consumption. A strategy to improve energy efficiency allows the writing of a new word, without erasing the stored data. This strategy can be employed only when there are no bit changes from 0 to 1 between the stored data and the next data to be written.

3.2.6 Encoder

The Encoder module codifies the output frame received from the Control Module, and that must be sent to the Backscatter Modulator module. The coding is done according to FM0 encoding method. In FM0 encoding, also known as biphase space, a transition is present in the beginning of every clock period, and an additional transition may be present in the middle of the clock period. In this method, a '1' bit is transmitted as no transition in the middle of the clock period, while a '0' bit is transmitted as a transition in the middle of the clock period.

4. SYNCHRONIZATION ALGORITHM

The input frame is not synchronized to the local reference clock and, to decode data without losses, it is necessary synchronizing it to the local clock (640 KHz), which may vary up to 10% from successive clock periods. This huge variation implies a complex design of the synchronization circuit, whose proposed algorithm is described next.

Figure 2 shows the input frame format, which has five fields: (*i*) the preamble detect field is a sequence of bits in logic '1' during at least 400 µs. This sequence is used by Demodulator to start the execution of other modules; (*ii*) the preamble field is a sequence of 9 bits encoded in Manchester ("010101010101010101"). It is used as reference to identify the external clock variation; (*iii*) the start delimiter field may have four possible values: "1100111010", "0101110011", "0011100101" or "11011100101". It enables the transmission rate identification, which must be used in the communication from tag to reader and allows the command identification; (*iv*) the command field has the command to be performed by the tag; (*v*) the CRC-16 field has the packet CRC.

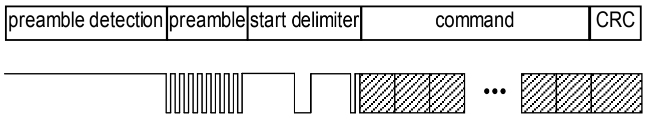

Figure 2 – Input frame format.

The frame decoding process has three stages: synchronization, start delimiter detection and data decoding from Manchester to NRZ. In the first stage, the preamble field is used to estimate the transmission clock frequency. Therefore, the system can calculate the number of clock cycles needed to receive each bit without losses. The reading is done in the raising edge of clk_sync_s signal, as far as possible from the input data signal edge, and the read data is stored in the last_read_s signal.

Figure 3 shows some important points to help understanding the frame detection algorithm:

1. The Detection of a falling edge in the lastread_s signal, indicates the beginning of a preamble.
2. The count_preamble_s signal counts the number of clock cycles needed to receive the first 8 bits of the preamble.
3. The count_p_bits_s signal counts the number of received preamble bits.
4. At every received preamble bit the value of count_preamble_s is stored in the count_last_p_s signal. This signal is used to identify 2 or more consecutive bits with the same logic value, indicating the reception of an invalid preamble.
5. Count_sync_s is reset at every transition edge of the last_read_s signal. While last_read_s maintains its logical value, Count_sync_s is incremented each clock transition. If last_read_s receives 4 consecutive bits with the same logic value, which is not allowed at any field of the frame, the Count_sync_s increment is stopped.
6. Sync_mode_s determines the input data-reading mode, which reflects the input frequency.
7. The raising edge of clk_sync signal indicates the moment for data reading, which depends on the reading mode (sync_mode_s) and on the number of clock cycles (count_sync_s) without transition edge in the last_read_s signal.
8. Every received input bit is stored in a shift buffer (buffer_s), which is used to detect the start_delimiter field.

9. After receiving the 8th preamble bit, which is indicated by count_p_bit_s, the reading mode (sync_mode_s) is adjusted, according to the value of count_preamble_s. At this moment, the decoder starts trying to detect the start_delimiter field. This is done by comparing the stored value in buffer_s with predefined values, established by ISO 18000:6-B.

10. After receiving the last bit of start_delimiter, the value inside buffer_s is valid, setting the delimiter_ok_s signal.

11. In this moment, the state machine starts decoding input commands data.

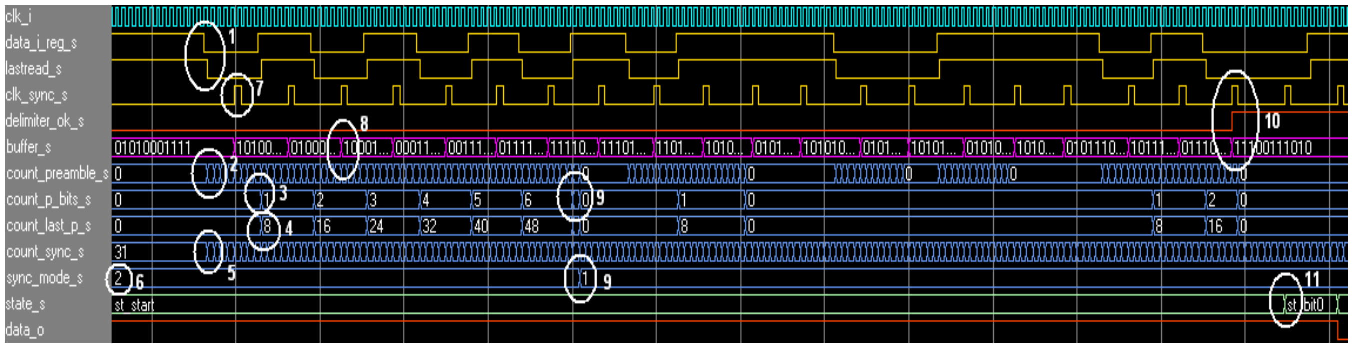

Figure 3 – Important points of frame detection algorithm.

During the frame detection process, the synchronization algorithm is able to avoid data losses resulting from the reference clock variation. Figure 4 shows an example of an error on start delimiter detection. This error may occur after have received three consecutive bits with the same logic value, which was caused because of a variation in the reference clock and the synchronization algorithm was disabled and there was.

The following simulation points in Figure 4 are important to mention:

1. The first bit of start_delimiter is received.
2. The second bit of start_delimiter is also received correctly. Note that this is the third consecutive bit in the lastread_s signal with the same logic value ('1').
3. Here, the third consecutive bit received with the same logic value is lost, due to a synchronism error, caused by a variation in the reference clock signal (clk_i).
4. At this point, the signal delimiter_ok_s should be set to '1', after receiving the last bit of start delimiter. However, the bit lost in the simulation point 3 makes the start delimiter recognition impossible.

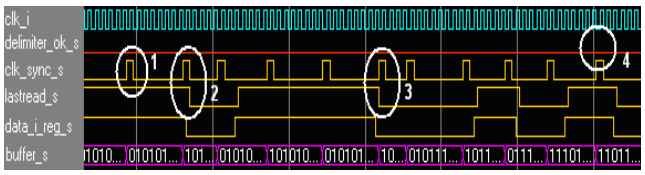

Figure 4 – Example of error on start delimiter detection.

5. FLEXIBLE DESIGN FLOW

The proposed architecture implements a facility to allow the selection of different command sets, according to the target application needs. This feature makes the RFID tag more flexible, when considering power consumption and read/write range specifications.

Customization is performed over Digital Control module. Since it is the most area and power consuming module, as shows Table 1, the facility for performing different configurations in the command set will result in a significant impact on the overall tag costs.

Figure 5 shows the commands select flow. It starts from the library of commands and an application specification file. Based on these inputs, the customer can select the set of commands to be implemented in the Digital Control module.

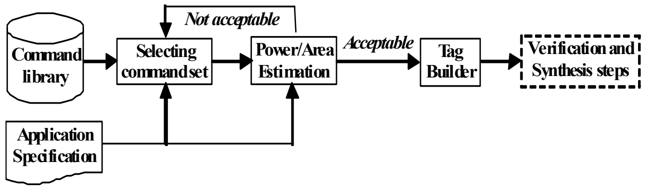

Figure 5 – Command select flow.

The next step is the area and power estimation, which allows iterations, selecting a new instruction set, in order to meet the expected area and power design requirements. When the requirements are met, the next steps concern the new tag production (VDHL design flow).

6. SYSTEM VALIDATION AND SYNTHESIS

This Section describes the system validation through functional and gate-level simulation. Before the validation stage, a tag was configured according to the command select flow, described in the last section. The configuration of a generic tag used in supply chain applications has been chosen as a case study. Such tag supports all the commands determined by ISO 18000-6B standard, and some proprietary commands planed to accomplish supply chain purposes.

Figure 6 shows the system validation flow, which is performed by functional and gate-level simulation.

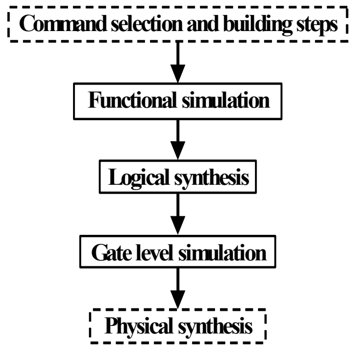

Figure 6 – System validation flow.

The validation of the tag VHDL design using Modelsim is split in four stages. In the first three stages it is performed VHDL functional simulation and, the last one is a gate-level simulation.

In the first stage, the digital modules were individually submitted to white box testing during the development, in order to improve their correct functioning. After this stage, the digital modules were interconnected, creating a new entity named Top Digital Block.

Second stage consists of a white box testing of Top Digital Block, which is performed using waveform analyses. The Top Digital Block has been simulated and its basic functions have been tested. The basic functions include: receiving packets, decoding commands, checking CRC, building and sending responses. A C program randomly generates frames containing commands, which are read from a text file.

In the third stage, the tests were automated, in order to perform massive verification of the Top Digital Block. First, a C program, named C-Tag, generates all Digital Block outputs according to its inputs. The C-Tag works executing the same stages of a real tag: data receiving, data decoding, command executing and response sending. In other words, it reproduces the Digital Block behavior under many conditions, following the ISO 18000-6B standard. This standard defines how this tag should work. Massive verification uses around of ten thousand commands per session, and each command have the same size and data of commands received by a real tag.

The next step simulates both C-Tag and Top Digital Block. During C-Tag execution, all outputs and some internal debug signals are stored in a text file. In a similar way, signals from Digital Block are also monitored and stored in another text file. After that, another external program analyses the data in the two text files and verifies if the behavior of Digital Block matches C-Tag behavior.

If no errors are detected, a new test round begins. Otherwise, detected errors can be checked several times, simulating the same error state until the errors remain. Simulation status is stored in a third text file. In case of error detection, all debug signals are also stored in this file. The simulation stop criteria are: complete command coverage for each possible digital module state; and 100% of code coverage. Once simulation stops, the system is ready for logical synthesis. This process creates a Verilog netlist and delay files.

The last stage consists of an automated gate-level simulation with netlist and delay files. The simulation process of the third functional validation stage is performed again, but now with gates and interconnections details. The Verilog netlist file is used to estimate power consumption, while delay files have all signals delays according to connection wires characteristics.

When the simulation reached the stop criteria with no errors found, the system was sent to the foundry. In a first moment, the foundry will provide a small set of tags for exhaustive testing activities. Afterwards, the final tag design will be used to produce a commercial set of tags.

7. SYNTHESIS RESULTS

After system validation, it is possible to extract the VCD files, which have information about the behavior of each internal module. Synthesis tool uses VCD files in power consumption analysis. After this, all verification and synthesis steps are performed again, but now for the Top Digital Block, which integrates all internal modules. Physical synthesis determines the integrated circuit layout using optimizations of area usage and power consumption, which is shown in Table 1. The first three columns present, respectively, the total cell area occupied by each module, the total number of cells and the number of sequential cells inside each module. The difference in the number of cells between the Top Digital Block and the sum of all internal modules is due to the insertion of buffers, used for power optimizations in the Top Digital Block.

The fourth and fifth columns show the typical and worst case of power consumption of each module in a 3.0 V and 25^{o}C scenario. Similarly, the sixth and seventh columns show the power consumption in a 3.3V and 70^{o}C scenario.

Figure 7 shows the RFID tag circuit final layout using 0.35μm standard CMOS technology.

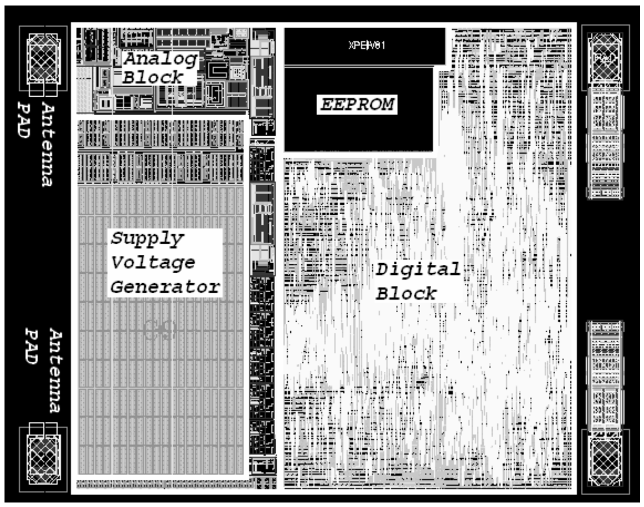

Figure 7 – Circuit layout.

Table 1: Area occupation and power consumptions results for Digital Block and its internal modules.

Digital Block	Area occupation			Power consumption (w)			
	Total Cell Area (μm^2)	Total Cell	Sequential Cells (FFs)	3.0V and 25°C		3.3V and 70°C	
				Typical	Worst	Typical	Worst
Decoder	34143.2	270	48	7.0454e-03	7.9445e-03	9.1280e-03	1.0269e-02
Control	440203.4	4471	340	3.6410e-02	4.0963e-02	4.7220e-02	5.3013e-02
CRC	7425.6	38	16	1.7745e-03	1.9919e-03	2.3047e-03	2.5814e-03
Memory Control	108945.2	972	125	1.3791e-02	1.5470e-02	1.7896e-02	2.0030e-02
Encoder	9864.4	96	14	1.2958e-03	1.4648e-03	1.6803e-03	1.8925e-03
Test	13722.8	123	2	3.0224e-03	3.4164e-03	3.9190e-03	4.4184e-03
Top Digital	616652.4	5998	545	8.8119e-02	9.9060e-02	1.1264e-01	1.2637e-01

8. CONCLUSIONS

This wok describes the implementation of a passive 915 MHz UHF low power RFID tag. The main contributions are: (i) a new tag architecture, which has been written in VHDL, simulated and synthesized; (ii) a synchronization algorithm, which allows the correct data decoding even in the presence of reference clock variations during execution; and (iii) a flexible design flow, allowing the selection of selected commands aiming power consumption improvement.

Different configurations for the tag's command set have been tried, and a chip design for a typical application was chosen as a case study.

The tag has been fully tested and validated at several simulation levels, and the synchronizing algorithm performed according to the expected.

The resulting design has been sent to a foundry. Field tests are going to be performed on the actual chip, and a new and re-validated design version will be sent to the foundry for commercial production in large scale.

9. REFERENCES

[1] Want, R. The Magic of RFID Just How does Those Little Things Work Anyway? ACM Queue, v. 2, n. 7, Oct. 2004.

[2] Mamei, M. et al. Making Tuple Spaces Physical with RFID Tags. Symposium on Applied Computing, ACM Press, pp.434-439, 2006.

[3] Weinstein, Ron. RFID: A Technical Overview and Its Application to the Enterprise. IEEE Computer Society, v. 7, n. 3, pp. 27-33, 2005.

[4] Hassan, T. and Chatterjee, S. A Taxonomy for RFID. Hawaii International Conference on Systems Science. Jan. 2006.

[5] Choi, N-G. et al. A 13.5 MHz RFID System. IEEE International Solid-State Circuits Conference. Dec. 2005.

[6] Leong, K. et al. Synchronization of RFID Readers for Dense RFID Reader Environments. International Symposium on Applications and the Internet Workshops, Jan. 2006.

[7] Karthaus, U. and Fisher, M. Fully Integrated Passive UHF RFID Transponder IC with 16.7 µW Minimum RF Input Power. IEEE Journal on Solid-State Circuits, v. 38, n. 10, pp. 1602-1608, Oct. 2003.

[8] Leenaerts, D. Low Power RF IC Design for Wireless Communication. International Symposium on Low Power Electronics and Design. pp. 428-433, Aug. 2003.

[9] Vancherand, F. New Technologies for Contactless Microsystems. ACM International Conference Proceeding Series, v. 121, pp. 13-17, 2005.

[10] Porret, A-S. et al. Tradeoffs and Design of an Ultra Low Power UHF Transceiver Integrated in a Standard Digital CMOS Process. IEEE Journal of Solid-State Circuits, v. 35, n. 3, Mar. 2000.

[11] Jones, A. et al. An automated, FPGA-based reconfigurable, low-power RFID tag. Proceedings of the 43rd Annual ACM IEEE Design Automation Conference, pp. 131-136, 2006.

[12] Guo, L. H. A Small OCA on a 1x0,5-mm2 2.45-GHz RFID Tag-Design and Integration Based on a CMOS - Compatible Manufacturing Technology. IEEE Electron Device Letters, v. 27, n. 2, Feb. 2006.