Implementation of a Receiver for an EPC Tag Emulator for Performance Evaluation

Michael Winkler [#1], Thomas Faseth [#], Juergen Steininger [#], Holger Arthaber [#]

[#] *Institute of Electrodynamics, Microwave and Circuit Engineering, Vienna University of Technology*
Gusshausstrasse 25/354, 1040 Vienna, Austria
[1] michael.winkler@tuwien.ac.at

***Abstract*— In this paper the receiver path of an EPC Class-1 Gen-2 conform RF identification (RFID) Tag Emulator at 868 MHz is presented. The underlying hardware avoids the disadvantages of common tag emulators which emulate the logical behavior only. The implemented EPC Tag Emulator allows the precise emulation of the physical behavior of an UHF RFID tag. Therefore, it can handle worst case test scenarios for reader performance evaluation. In particular, the backlink frequency (BLF) of each backscattered symbol as well as the tag to reader link timing can be varied beyond the EPC specifications. The continuously changeable complex input reflection coefficient allows the investigation of the influence of a changing phase and/or a Doppler shift on the reader and the evaluation of the reader's receiver sensitivity.**

The bit decoder which is implemented in FPGA hardware is discussed in detail in this paper. The data symbols are decoded autonomously after a correct *Preamble*/*Framesync* is detected. Therefore, the bit decoder unit cross-checks the timing values and an undesired command is discarded. A valid *Preamble*/*Framesync* starts the command parser which accesses the bit decoder unit. The command parser and the protocol stack are handled by the microcomputer unit in the higher layers in software. The protocol stack includes all mandatory commands. The receiver path is tested under real conditions and the sensitivity of $P_{T_{in}} = -29.3\,\mathrm{dBm}$ is much better than the sensitivity of common tag ICs. Furthermore, it is shown that the tag emulator replies faster than specified by the tag to reader (T⇒R) link timing. The reader's behavior on a faster tag reply can be investigated.

I. Introduction

In passive UHF RFID the tag is exclusively powered by the reader's electromagnetic field. It is important to say that the tag's power consumption limits the distance between reader and tag [1][2]. Therefore, a lot of work has be done to increase the energy efficiency of the tag [3][4]. The tag communicates with the reader by backscattering the incident electromagnetic wave by changing its antenna impedance. The higher the reflection coefficient the more signal is backscattered. This simplifies the correct decoding for the reader. But there is an important trade off because the more signal is reflected the less power can be used for supplying the tag's chip [1][2].

In order to determine such system margins, RFID systems have to be verified under worst case conditions. A very useful tool for evaluating the performance of a reader is a tag emulator. Because of the limited size, power supply, and/or computing power most tag emulators are not able to map the physical behavior of the tag [5]. This limits the range of test scenarios substantially and different investigations on readers cannot be done under realistic conditions. These disadvantages are eliminated by the presented EPC Tag Emulator [6][7]. Essentially, the presented EPC Tag Emulator can precisely emulate the physical behavior of a common tag and furthermore worst case test scenarios can be implemented. The basic concept of the underlying hardware is explained in Sec. II. Detailed information about the implementation of the receiver path (especially the bit decoder block, the command parser, and the protocol stack) and significant measurement results are presented in Sec. III.

II. The Underlying Hardware

What is the typical physical backscatter behavior of a tag and what is required for precise emulation?

The backlink frequency (BLF) of a backscattered tag reply varies enormously. The maximum frequency tolerance is 22 % as specified in [8]. The frequency changes even during backscattering and furthermore the tag to reader T⇒R link timing varies. For a reader receiver it is difficult to decode such a backscattered signal. In order to emulate such a behavior, a lot of processing power is required.

Furthermore, the tag's reflection coefficient depends on the received signal strength because the tag's chip impedance changes due to the internal rectifier. Furthermore, the reader's receiver sensitivity should be evaluated by decreasing the reflection coefficient. The influence of a changing phase or a Doppler shift on the receiver should be investigated. In order to handle this, a backscatter modulator with a continuously changeable complex reflection coefficient and an adequate baseband processing is required.

Conventional tag emulators consist of a simple microcontroller and changes the reflection coefficient only between two states [5]. They measures the received power only at a single point in time altough it would be very useful to monitor the complete downlink (reader to tag) signal.

All these aspects and their consequences were considered in the implementation of the EPC Tag Emulator. Apart from the power supply, the required circuitry is very difficult to implement on a PCB with the size of a typical tag antenna. The main idea of the underlying hardware is to split the EPC tag emulator into the RF Transceiver Board and the Baseband Processing Board. The miniaturized and low power RF Transceiver Board is placed into the reader's electromagnetic field like a common tag and is powered by

a photonic power converter driven by a laser source. The Baseband Processing Board includes a powerful Virtex5 FPGA where all the baseband processing of the receiver path and of the transmitter path is flexibly implemented. In order not to influence the electromagnetic field near the tag antenna [9], both boards communicate by a bidirectional 120 Mbit/s optical link. Detailed information about the RF Transceiver Board, the Baseband Processing Board, the optical links and its robust synchronization algorithm is given in [6].

The I/Q Backscatter Modulator, one of the main parts of the RF Transceiver Board, in combination with the appropriate transmitter baseband processing offers the ability to emulate worst case test scenarios. The BLF of each backscattered symbol as well as the T⇒R link timing can be varied far beyond the EPC Class-1 Gen-2 specifications [8]. The minimum and maximum link timing values are specified for different BLFs (compare Fig. 1).

It is shown that the underlying hardware is furthermore able to change the amplitude and the phase of the input reflection coefficient of the I/Q Backscatter Modulator continuously [6][7].

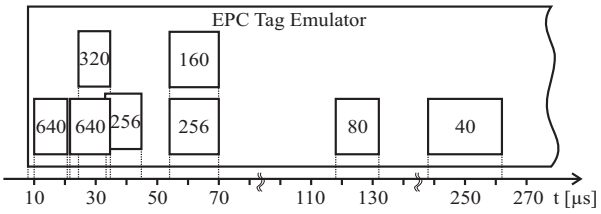

Fig. 1. Tag to reader (T⇒R) link timing specified for different BLFs.

III. The Concept of the RX Path

A. Principles of Operation

A simplified block diagram of the receiver path of the EPC tag emulator is depicted in Fig. 2. The RF Transceiver Board's antenna is designed for 868 MHz. The power detector (PD) measures the antenna's received signal strength. The output signal of the power detector with a dynamic range of -50 dBm. . .-5 dBm is sampled by an ADC (3 MSps/10 Bit) and is sent to the Baseband Processing Board by the optical link where the frame synchronization is done in a first step. The sensitivity of the receiver path is better than the typical UHF RFID tag IC sensitivity [10][11] in order to achieve the same reading distances for the EPC Tag Emulator.

The data of an EPC conform downlink signal (reader to tag

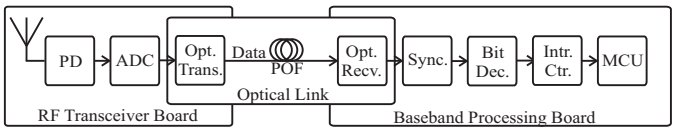

Fig. 2. Simplified block diagram of the complete EPC tag emulator receiver path.

link) is Pulse Interval Encoded (PIE) [8] and has to be decoded regarding two important aspects. First, a signal with a not EPC conform *Framesync*/*Preamble* (maybe from an interferer) has to be ignored. The decoding and incorrect interpretation of the received signal should be avoided as far as possible. The second aspect is a timing aspect. The moment of the tag's reply, the so called T⇒R link timing, is well defined. The minimum delay and, therefore, the critical one for the tag response is $T_{1min} \approx 10.2\,\mu$s [8]. Within this period the EPC protocol stack has to be handled and the transmitter path has to be configured if needed.

In order to handle these requirements, the bit decoding is done in FPGA hardware in the autonomous Bitdecoder unit. A level higher, the microcomputer unit (MCU) processes the decoded symbols of the Bitdecoder in the command parser. Furthermore, the MCU handles the protocol stack in the higher levels. Detailed information about the implementation is given in the following section.

B. Technical Implementation and Measurement Results

In the Bitdecoder block the digitized signal from the power detector is split in a first step (Fig. 3). One signal is averaged by a lowpass (FIR) filter. Then, both signals are compared by the comparator and the binary output signal indicates the notches of the PIE downlink signal. The coefficients of the FIR filter are weighted to avoid a noisy output signal of the comparator because similar signals are compared if there is no modulation from the reader.

The following logic analyses the timing between the notches. Therefore, the rising edges of the comparator's binary output signal are detected. If a command from a reader is received, the first rising edge after the delimiter [8] starts the state machine (FSM) in the Bitdecoder. The state machine furthermore analyses and validates the timing values of the preceding *Preamble*/*Framesync* [8]. Therefore, the correlation between *Tari*, *RTcal*, and *TRcal* is checked. Remember that *Tari* defines the length of *Data 0* symbol, with the help of *RTcal* a common tag decides between a *Data 0* and a *Data 1* symbol, and *TRcal* specifies the BLF. The state machine is reset if the timing values are out of the EPC specifications. If a valid *Preamble*/*Framesync* is detected by the decoder, the interrupt signal "Start Detected" interrupts the MCU.

After the preceding *Preamble*/*Framesync* the Bitdecoder unit

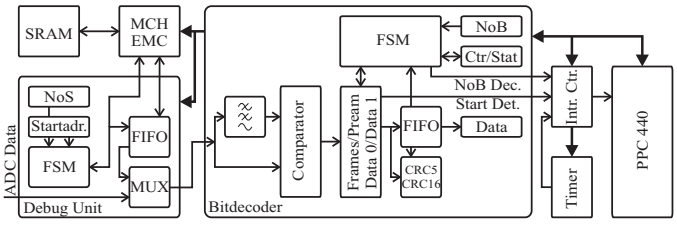

Fig. 3. Block diagram of the receiver path implemented on the Baseband Processing Board.

starts decoding the data symbols. Therefore, the Bitdecoder unit decides if the symbol is Data 0/Data 1 based on the *RTcal* value from the *Preamble*/*Framesync*. Again, the Bitdecoder's

TABLE I
TYPICAL SENSITIVITY VALUES OF EPC CLASS-1 GEN-2 TAG ICS

EPC Class-1 Gen-2 IC	Condition	typ. Sensitivity
RI-UHF-IC116-00 [10]	Write Read	-9 dBm -13 dBm
IPJ-W1510 [11]	Write Read	-17.1 dBm -19.9 dBm
EPC Tag Emulator	Write Read	$\leq$-29.3 dBm $\leq$-29.3 dBm

state machine is reset if the length of the data symbol is out of range. In the other case the decoded symbol is written into the FIFO buffer.

As mentioned before, the command parser and the protocol stack are implemented in higher levels in the software by the MCU. By writing the Number of Bits (NoB) register, the MCU requests an adequate amount of decoded symbols. The interrupt signal "NoB decoded" is set after the requested number of bits are decoded and the command parser is interrupted. The decoded symbols which are stored in the FIFO buffer are read out and processed. The third interrupt source in Fig. 3 is a timer unit which is configured for indicating a time out.

For debug purposes and for regression tests, the Debug unit (Fig. 3) was implemented in the FPGA hardware. This unit allows the emulation of a received downlink signal. Therefore, a test sequence is written over the multi channel external memory controller (MCH EMC) into the SRAM by the PPC440 in a first step. After that, the stored test sequence can be read out by the Debug unit as often as wanted.

In order to show the functionality under real conditions and not only by test sequences of the Debug unit, the EPC Tag Emulator was connected to a commercial reader by a cable. The cable connection ensures an adequate reproducibility of the measurement results. The measured signal strength on the input of the Tag Emulator is $P_{T_{in}} = -29.3\,\mathrm{dBm}$ (in Table I typical sensitivity values of tag ICs are presented). Assuming that the reader's maximum transmission power is $P_{R_{out}} = 33\,\mathrm{dBm}$, this corresponds to a free space loss of $FSL = 62.3\,\mathrm{dB}$ which is equivalent to a distance of $d = 35.8\,\mathrm{m}$!

In Fig. 4 the measurement result of a received and decoded

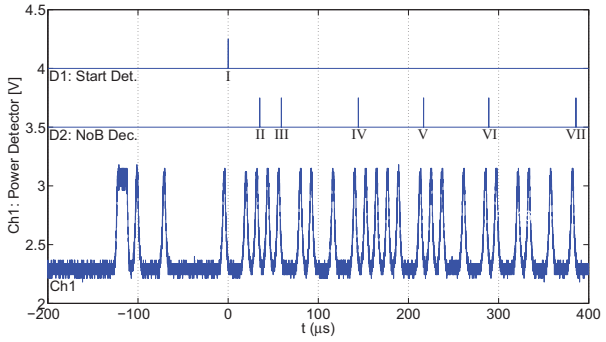

Fig. 4. Measurement result for a received and decoded Query command; Ch1: output signal of the power detector (inverted characteristic), D1: interrupt signal Start Detected, D2: interrupt signal NoB Decoded

Query command is depicted. "Channel 1" shows the output signal of the power detector (remember the inverted characteristic). The interrupt sources marked by "D1" and "D2" in Fig. 4 activate the MCU which parses the command and handles the protocol stack. The simplified state diagram for parsing the Query command is presented in Fig. 5. The roman numbers in Fig. 4 indicates the transitions in the state diagram. After detecting a valid Preamble (compare digital signal "D1: Start Detected" in Fig. 4) the command parser changes from the IDLE State into the Start Detect State (transition I in Fig. 5) where the first two bits of the command code [8] are requested. The event when the requested number of bits is decoded is indicated by the digital signal "D2: NoB Decoded" in Fig. 4. After event III, the four bit command code of the

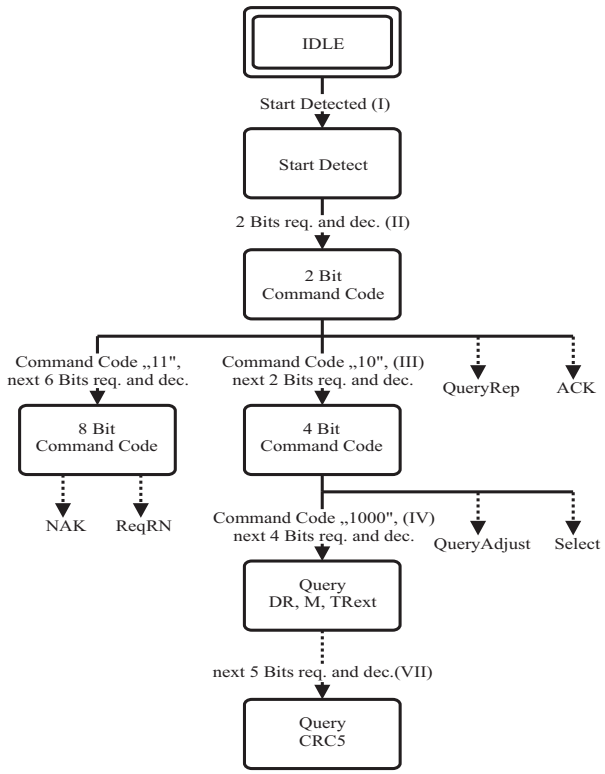

Fig. 5. Simplified state diagram of the command parser.

Query command is decoded. In order to meet the requested tag to reader link timing, it is very important that the command code and different command parameters are already decoded during receiving the command. The higher levels in the protocol stack can be processed and a tag reply in time is possible if necessary.

For example, after event IV, the Query command parameter *DR*, *M*, and *TRext* are known. *TRext* chooses whether the tag's reply preamble is extended with a pilot tone and *M* (cycles per symbol) sets the data rate and the modulation format. Therefore, these are important configuration parameter for the Backscatter baseband processing unit [7]. *DR* (in combination with the *TRcal* value from the *Preamble*) defines the BLF. The BLF can independently be set for each backscattered symbol

in order to enable powerful statistical test scenarios [7]. The calculation of the BLF needs an adequate computing time and has to be completed within the limits of the T⇒R link timing. Event VII indicates the end of the *Query* command which is protected by a CRC5 and is calculated in the FPGA hardware. An appropriate bit which indicates the validity is set in the Status register.

In EPC Class-1 Gen-2 the so called random-slotted collision arbitration is used. Based on the decoded *Q* command parameter, the internal slot counter is loaded with a random number (details about selecting, inventorying, and accessing tags can be found in [8]). The EPC Tag Emulator replies a 16 Bit Random Number (*RN16*) if the slot counter is zero. The slot counter is decremented by single *QueryRep* commands until zero.

A related measurement result of a *QueryRep* command and a *RN16* tag reply is depicted in Fig. 6. The preceding *Framesync* is detected after the first data symbol (a *Framesync* is without *TRcal*). The *RN16* (BLF=320 kHz, *M*=1, *TRext*=1) is measured at the DAC output for the I channel which is the input signal of the I/Q Backscatter Modulator on the RF transceiver Board [6]. The delay of the tag response is $\approx 8.9\,\mu$s and therefore closer than the required minimum delay of $T_{1min} \approx 10.2\,\mu$s [8]. This is important because the reader's behavior on reply delays which are closer than defined in the specifications can be investigated.

The protocol stack is implemented for all mandatory commands. Commands which are unsupported, with a wrong CRC or wrong command parameter are ignored. Furthermore the validity of a command depends on the current tag state [8] which is also implemented.

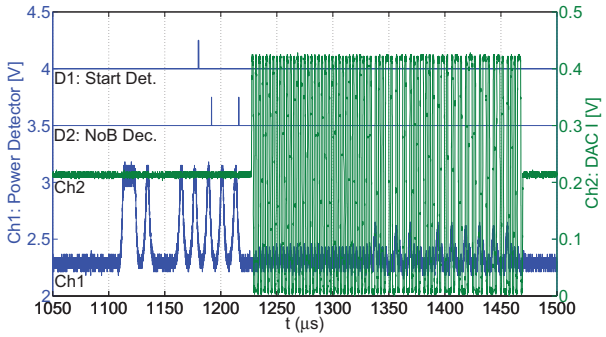

Fig. 6. Measurement result for a received and decoded QueryRep command; Ch1: output signal of the power detector, Ch2: output signal of DAC I-channel (input signal for modulator), D1: interrupt signal Start Detected, D2: interrupt signal NoB Decoded

IV. Conclusions

In this paper the implementation of the receiver path for an UHF RFID tag emulator for 868 MHz is presented. The underlying hardware is designed in order to precisely emulate the physical behavior of an UHF RFID tag [6]. The transmitter path of the EPC Tag Emulator, including the I/Q Backscatter Modulator and the Backscatter Baseband Processing, is able to evaluate the reader performance under worst case conditions [7].

The implementation of a decoder for EPC Class-1 Gen-2 conform downlink signals is presented. The Bitdecoder is implemented in FPGA hardware and works autonomously to save processing resources. In order to avoid an unintended begin of decoding, the Bitdecoder unit starts only after a valid *Preamble*/*Framesync*. Therefore, the relation between *Tari*, *RTcal* and *TRcal* is verified. The Decoder unit is tested under real conditions by a commercial reader with a measured input power of $P_{T_{in}} = -29.3\,\mathrm{dBm}$ and different measurement results are depicted. It is shown that the sensitivity of commercial tag chips is much lower than the sensitivity of the EPC Tag Emulator.

Furthermore, the command parser and the protocol stack, both handled in higher layers in software by the MCU, are presented exemplarily. All mandatory commands are implemented in the protocol stack. It is shown that the EPC Tag Emulator is able to reply faster than specified by the tag to reader link timing. This is important because the influence of a non specified link timing on the reader can be investigated.

Acknowledgment

This work was supported by Vienna's technology agency "Zentrum für Innovation und Technologie" (ZIT).

References

[1] L. Sydanheimol, J. Nummela, L. Ukkonen, J. McVay, A. Hoorfar, and M. Kivikoski, "Characterization of Passive UHF RFID Tag Performance," *IEEE Antennas and Propagation Magazine*, vol. 50, no. 3, pp. 207 –212, 2008.

[2] F. Fuschini, C. Piersanti, F. Paolazzi, and G. Falciasecca, "Analytical Approach to the Backscattering from UHF RFID Transponder," *IEEE Antennas and Wireless Propagation Letters*, 2008.

[3] J. Rodriguez-Rodriguez, M. Delgado-Restituto, and A. Rodriguez-Va? andzquez, "Baseband-processor for a passive UHF RFID transponder," in *2010 International Conference on Green Circuits and Systems (ICGCS)*, 2010, pp. 344 –348.

[4] S.-J. Kim, M.-C. Cho, J. Park, K. Song, Y. Kim, and S. Cho, "An Ultra Low Power UHF RFID Tag Front-End for EPCglobal Gen2 with Novel Clock-Free Decoder," in *IEEE International Symposium on Circuits and Systems, 2008. ISCAS 2008.*, May 2008, pp. 660 –663.

[5] R. Redemske and R. Fletcher, "Design of UHF RFID emulators with application to RFID testing and data transport," in *Proc. IEEE International Workshop on Automatic Identification Advanced Technologies*, Buffalo, New York, Oct. 2005, pp. 193–198.

[6] M. Winkler, T. Faseth, H. Arthaber, and G. Magerl, "An UHF RFID Tag Emulator for Precise Emulation of the Physical Layer," in *2010 European Microwave Conference (EuMC)*, Paris, France, 2010, pp. 1750 –1753.

[7] M. Winkler, T. Faseth, and H. Arthaber, "Implementation of an EPC Tag Emulator for Reproduction of Worst Case Test Scenario," in *Proc. IEEE Third International EURASIP Workshop on RFID Technology*, La Manga del Mar Menor, Spain, 2010.

[8] *EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz - 960 MHz Version 1.2.0*, EPCglobal Specification for RFID Air Interface, Rev. 1.2.0, 2008.

[9] T. Fukasawa, K. Shimomura, and M. Ohutsuka, "Accurate measurement method using fiber-optics for an antenna on a portable telephone," in *Proc. IEEE Topical Conference on Wireless Communication Technology*, Honululu, Hawaii, Oct. 2003, pp. 138–139.

[10] "RI-UHF-IC116-00 SMT EPC Gen2 IC Datasheet," Texas Instruments, Dallas, USA.

[11] "IPJ-W1510, IPJ-W1512, IPJ-W1513, Monza 4 Tag Chip Datasheet," Impinj, Seattle, USA.