We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



15

Design and Implementation of a Multi-protocol UHF RFID Tag Simulation Platform

Bo Zhang, Dongkai Yang and Qishan Zhang
School of Electronics and Information Engineering, Beihang University
China

1. Introduction

To raise the RFID UHF reader's performance, a test environment must be established. Using a lot of real tags to establish a test bed may induce some demerits, such as high resources costs and so on. So some institutions which engaged in RFID research developed some simulation systems for algorithm designing to provide reference information and technical support.

1. TI Gen2 standard tag simulator

TI (Texas Instrument Company) which is the world's largest RFID reader and transponder producer, developes a Gen2 standard tag simulator and provides to 5 main RFID reader and printer producers. This research is going on with the Gen2 standard product development in parallel, aiming to keep the compatibility between the transponder with the RFID reader and printer based on the UHF Gen2 standard. The tag simulator can generate 96bit codes and has other abilities defined by the EPC Gen2 protocol, is very helpful to the EPC Gen2 standard products development.

2. RFID tag simulator by CISC

The main part of the CISC RFID tag simulator is a high power and multifilament FPGA chip. The sensor modules are connected to the simulation tools. These sensor modules play a very important role in many simulations and tests.

The tag simulator has an unprecedented creativity virtue that the user can determine the tag's parameters based on different applications. Tags simulated can be completely controlled by setting the parameters to determine the worst and best cases. The simulator can not only be used for general tests, but can also be used for the reader's test at the case of boundary defined. The real tags can't be used for this test because the boundary parameters can't be adjusted. So, CISC sensor modules and tag simulator can be used together to emulate the UHF RFID tag to verify and analysis the UHF RFID system.

CISC tag simulator can emulate at most 16 tags, each tag have a separate RF module, to be equivalent to 16 real tags.

3. MATLAB used for RFID anti-collision simulation

Besides the tag simulators produced by the TI and CISC, there are several ways of RFID simulation reported, such as the way of using MATLAB to emulate the anti-collision algorithm, the way of using DSP and FPGA to establish a RFID system, and so on, and provide some reference values to the tag simulation design.

In this chapter, a tag simulator which can emulate the ISO/IEC 18000-6 type B and type C tags is introduced, and the parameters and numbers of the tags can be set also. It has the virtues such as:

1. Better for research and improvement of the collision algorithm

ISO 18000-6 Type B and Type C protocols are all involved in the anti-collision algorithm, Type B protocol is based on the Binary tree principle, and the Type C protocol is based on the ALOHA principle. With the tag simulator, the two protocols can all be tested and analyzed in convenient.

2. Emulates the actual situation

Signal's parameters (such as t_{ari} tolerance, RTcal tolerance and TRcal tolerance the timing parameters; Reverse scattering signal intensity and the signal's accuracy) can be directly modified and set for some specific scenes to emulate the real situation. Compared with the software simulator, the simulator introduced in this chapter is more reliable.

3. Compatible with various standards

The DSP and FPGA system both have the strong programmable and portable abilities, so the improvement and redevelopment based on it become more convenient. The tag simulator is compatible with various standards, has a higher reference value to consummate the standards.

2. RFID tag simulator hardware design

The general structure of the tag simulator is shown in Fig. 1, the whole system is divided into two major parts, the RF board PCB2 and the baseband board PCB1. The PCB2 board is mainly composed of antenna and RF module. And the PCB1 board is composed of DSP, FPGA, ADC, DAC and some other peripheral modules.

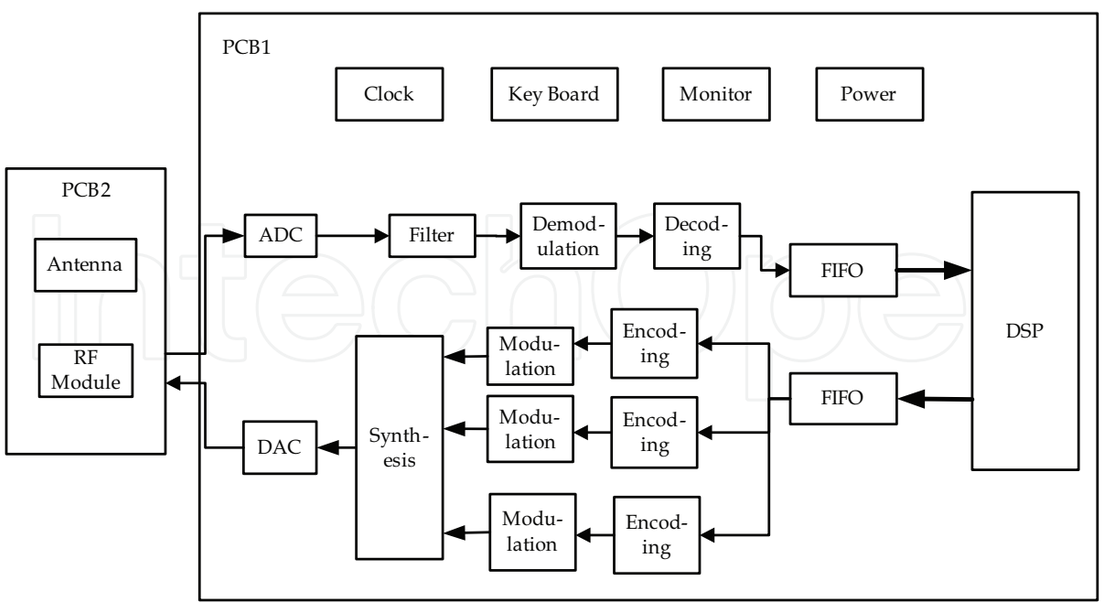


Fig. 1. General structure of the tag simulator

The RF module is mainly used for the RF and IF signals' frequency conversion. The baseband module is mainly used to realize the IF signals' demodulation, decoding and

radiation. The RF module and the IF module are connected via the ADC and DAC. In some process, it can be realized in digital way and is so called software radio technology. The virtue of digital frequency mixer is that it can avoid the discord between the I Q branches, but needs a high quality AD converter.

The DSP and FPGA chips are the cores of the baseband board, and this kind of system scheme is more and more prevalent and efficient now. The DSP chip can be used to realize the tag's protocol switching, state transferring, and data response. And the FPGA is mainly used in the signal's encoding, decoding, modulation, demodulation, data inspection, synthesis of multi channel and so on, and can also provide a stable clock, for its advantages in data processing speed.

2.1 Main hardware module

2.1.1 DSP and FPGA Interface

Connection between the DSP and FPGA includes the 32 bit data line evm_D[31:0], 20 bit address line evm_A[22:3], the DSP's general interface general_port[10:0], Serial Port 0 and Serial Port 2, control signal AWE, ARE, AOE and the chip select signal ACE2_n, ACE3_n, external interrupt signal EXT_INT[7:4], DSP's external input clock signal DSP_ECLKIN, DSP's external output clock signal TAECLKOUT2, and the DSP's reset signal evm_RESET.

The corresponding BEA mounts' level can be set as shown in Table 1.

mounts	functional description
BEA[19:18]	Bootmode [1:0] 00 – No boot 01 − HPI boot 10 − EMIFB 8-bit ROM boot with default timings (default mode) 11 − Reserved
BEA[17:16]	EMIFA input clock select Clock mode select for EMIFA (AECLKIN_SEL[1:0]) 00 – AECLKIN (default mode) 01 − CPU/4 Clock Rate 10 − CPU/6 Clock Rate 11 − Reserved
BEA[15:14]	EMIFB input clock select Clock mode select for EMIFB (BECLKIN_SEL[1:0]) 00 – BECLKIN (default mode) 01 − CPU/4 Clock Rate 10 − CPU/6 Clock Rate 11 − Reserved

Table 1. Settings of the DSP

2.1.2 ADC module

AD9433 is an Anolog to Digital converter chip produced by ADI company, it is a 12 bit sampling AD converter, and it also has the tracking/maintaining circuit on the chip. The conversion speed is as high as 125MSPS, and some optimize designs are made to adapt to the broadband and the dynamic performance of the high IF carrier system.

AD9433 needs a 5V analog power and a differential encoding clock to fit the chip's whole performance. In many applications, it does not require external benchmark and drive units. The digital output of the converter is compatible with the TTL/CMOS level. The proprietary circuit on chips can optimize the relationship between the spurious free dynamic range (SFDR) and the signal to noise and distortion ratio (SINAD) performance when input different frequency signals, the SFDR is up to 83dBc in the bandwidth from DC to 70MHz.
The chip has a 16 bit data line, each power mount of the chip is equipped with a filtering capacitance, the analog signal passing through the SMA interface is converted into digits by the AD converter after a 2 level transformer, a LVPECL difference clock and a matching network are adopted also.

2.1.3 DAC module

AD9777 is a 16bit DAC produced by ADI company, the input maximum data rate is 160MSPS (no interpolation) and 400MSPS (8 times interpolation). It has a optional interpolation ratio(2x/4x/8x) and a complex modulation. The direct IF pattern allows the synthetic intermediate frequency (IF) up to 70MHz.
The Designing of the DAC module is similar with the ADC module, but have several more serial peripheral interface configuration lines, and are connected to the FPGA, to configure the DAC's controlling memory.

2.1.4 User's interface module

The user's interface module is composed of keys and LED, the keys are input tools for users to choose the protocol (ISO 18000-6 Type B or Type C) and the tag numbers, LED is used to show the corresponding content, such as the status of collision, empty, successfully read, and so on.

2.2 FPGA circuit design

2.2.1 FPGA transmit link design

Parameters comparison of the two protocols in the transmit link are shown in table 2.

	Type B	**Class 1 Gen2（Type C）**
Modulation Mode	Bi-state Amplitude scattering modulation	ASK or PSK
Encoding Mode	FM0	FM0 or Miller
Data Rate	40 or 160kbit/s	FM0: 40 to 640kbit/s Miller: 5 to 320kbit/s
Preamble	16 bit Scattering modulation sequence	depends on command
Debugging Mode	16 bit CRC	CRC-16

Table 2. Parameters Comparison of the Tag -> Reader Link

The transmit link module of the FPGA includes FIFO, CRC check, FM0 and Miller encoding module, Digital Direct Frequency Synthesis (DDS), and Multi-channel synthesis module.

1. Transmit link FIFO design

FIFO is a first in first out data buffer, the difference with the general register is that it has no external address lines, so it can be used in a very simple way. The data address can be established by the internal reading and writing pointer automatically, and it can not read or write into an appointed address settled by the address line as the general register.

The FIFO can be used as the data buffer in different clock domain, and it can be used in the different width data interface too. In this chapter, data transferred between the DSP and FIFO are all in 32bit pattern, as shown in Fig. 2, but in the FPGA, most data are in serial pattern, the FIFO is used here to match the data transferred between the FPGA and DSP.

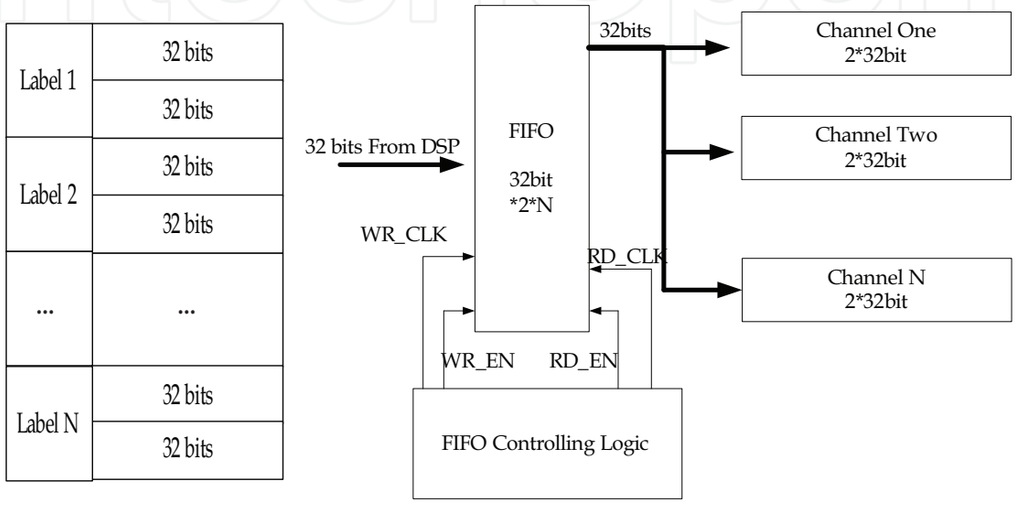


Fig. 2. Transmitting FIFO and the channel assignment principle

In the DSP commands, each tag needs to write the replied data into the FIFO, and the width of the data is 32bit, the width of the replied data is 64bit. But in Type C protocol, most data is not as wide as 32bit, only apart of it is used. For Type B protocol, the replied data's width is limited into 64bit, the CRC-16 data is also calculated and added in the FPGA. Each tag would write data into the FIFO in 2 times when replying, in the second time, if there is no additional data, 0x0000_0000 can be used to fulfill it.

When the assignments of writing the FIFO in all tags are finished, a signal is given to the FGPA, and when the FPGA gets the signal, it begins to assign the data for the FIFO. Each tag channel has a 64bit buffer. A synchronous clock (40kHz or 80kHz) is generated after the assignment, then it begins to process the data right shift, encoding and CRC generation and checking.

2. CRC-5 and CRC-16 checking

The main purpose of the Cyclic Redundancy Check (CRC) is to use the linear encoding theory to generate an n bit CRC checking codes in a certain standard based on the k bit transmitting binary message sequence end, it is attached behind the message, and is made up to a (k+n) bit binary sequence. In the receiving end, the message sequence in a standard way with the CRC checking codes is checked to make sure the message is right or not.

3. FM0 encoding module

FM0 is a kind of bi-phase space encoding method. The command from tag to reader begins with 2 preamble codes, which preamble code is chosen depends on the value of the TRext specified by the Query command.

The simulated waveform of the FM0 is shown in Fig. 3, including preamble code 1010v1 (v means the phase should be reversed but not), data_out is the output of serial data converted from the parallel data fm0_out[1..0].

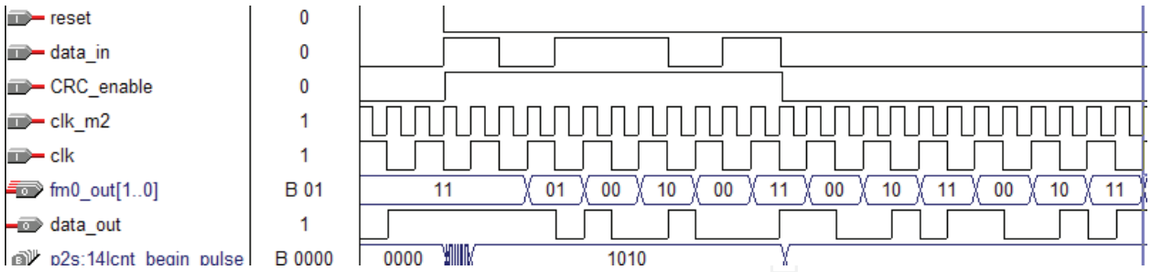


Fig. 3. Waveform of the FM0 encoding module

4. Miller Encoding module

In the ISO 18000-6 Type C protocol, the Miller Encoding performance is promoted by setting different sub_carrier wave frequency. By setting the 1X, 2X, 4X and 8X sub_carrier wave frequency, the reading scope, speed, and signal's bandwidth can be optimized, simulation in Fig. 4 shows the preamble code of the Miller Encoding, data_out is the output of serial data converted from the parallel data miller_out[1..0].

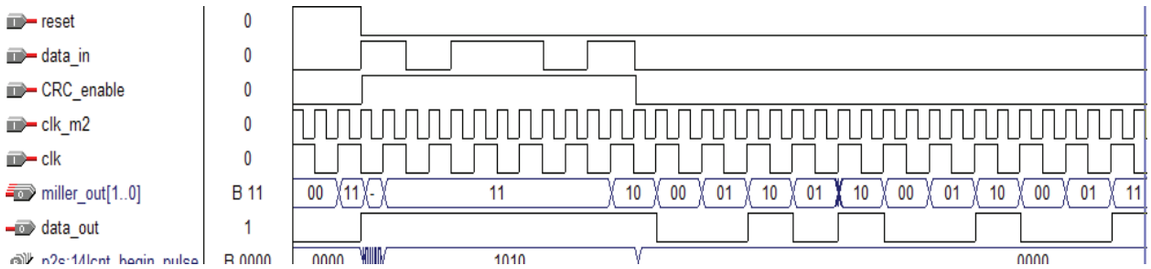


Fig. 4. Waveform of the Miller encoding module

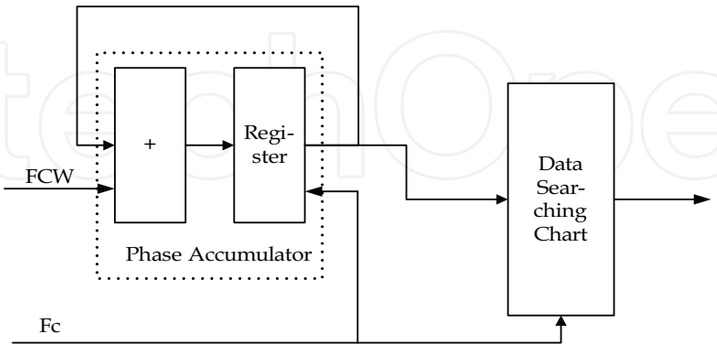


Fig. 5. Block Diagram of the DDS

5. DDS module

DDS is a new kind of frequency synthesizer invented in recent years, with applications of all digital large scale integration technology, it has some prominent characteristics such as low cost, high frequency resolution, fast switching, easy to be controlled and so on. And the

output signal's phase can be kept in continuous when frequency is switched into another one, and has low phase noise too. So it can be used to improve the reference frequency source's performance, and to be used to generate random waveform. The DDS module is composed of freqency controlling words (FCW), phase accumulator and data searching chart, as shown in Fig. 5.

6. Multi-channel synthesizer module

Multi-channel synthesizer is used to combine each channel which is simulated as a tag to one output.

2.2.2 FPGA receive link design

The receive link module of the FPGA including Low Pass Filter (LPF), Demodulation Module, Receive link FIFO and PIE decoding module.

1. Low Pass Filter

The design of Low Pass Filter (LPF) has many kinds of methods including Intellectual Property (IP) core from ALTERA with automatic COE parameters generation, or COE file data introduced from other sources. MATLAB Filter Design Toolbox was used in this LPF design as there are many functions to simplify the design and give quantitative effect analysis. Such parameters as filter type, order, sampling frequency and cutting frequency, amplitude attenuation could be configured during the design phase. And the various properties are shown with the graph like Fig. 6.

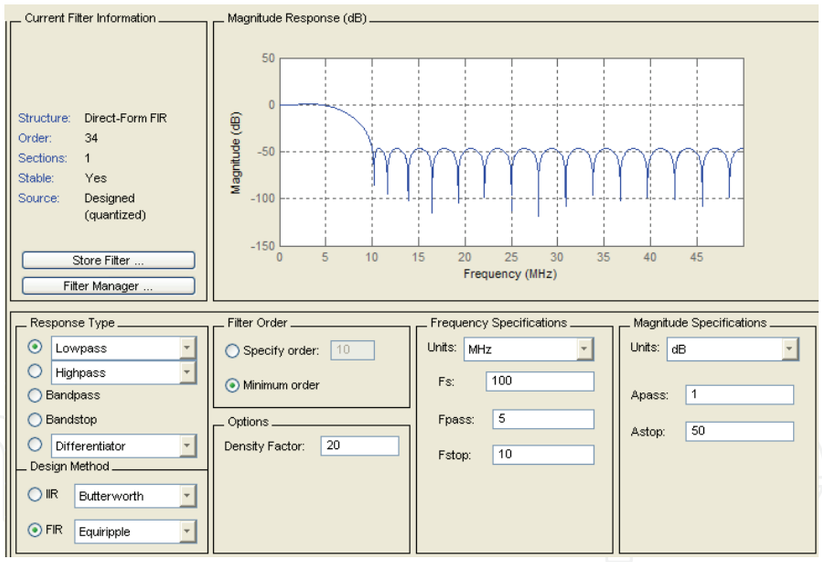


Fig. 6. Filter design interface

The proposed filter with valid specifications could be transferred from MATLAB to VHDL through assembling in FDA toolbox.

2. Demodulation module

Tag received the radio signals from the reader with different carrier phase changing over time delay. Orthogonal demodulation algorithm was used for ASK signal in the tag simulator [1]. Given the signal is expressed as equation (1),

$$S(t) = A\cos(\omega t) \tag{1}$$

The local in-phase and quadrature carrier are $\cos(\omega t+\varphi)$ and $\sin(\omega t+\varphi)$. out_I and out_Q are as follows after multiplier operation,

$$out_I = S(t) * \cos(\omega t+\varphi) = \frac{A}{2}\left[\cos(2\omega t)+\cos(\varphi)\right] \tag{2}$$

$$out_Q = S(t) * \sin(\omega t+\varphi) = \frac{A}{2}\left[\sin(2\omega t)+\sin(\varphi)\right] \tag{3}$$

The two signals will be remained phase related through this designed LPF and shown as in equation (4) (5),

$$filter_I = \frac{A}{2}\cos(\varphi) \tag{4}$$

$$filter_Q = \frac{A}{2}\sin(\varphi) \tag{5}$$

After squared and summed, we will get the constant $\frac{A^2}{4}$, i.e. correct signal could be got by judging this signal. ASK demodulation algorithm model is shown in Fig. 7.

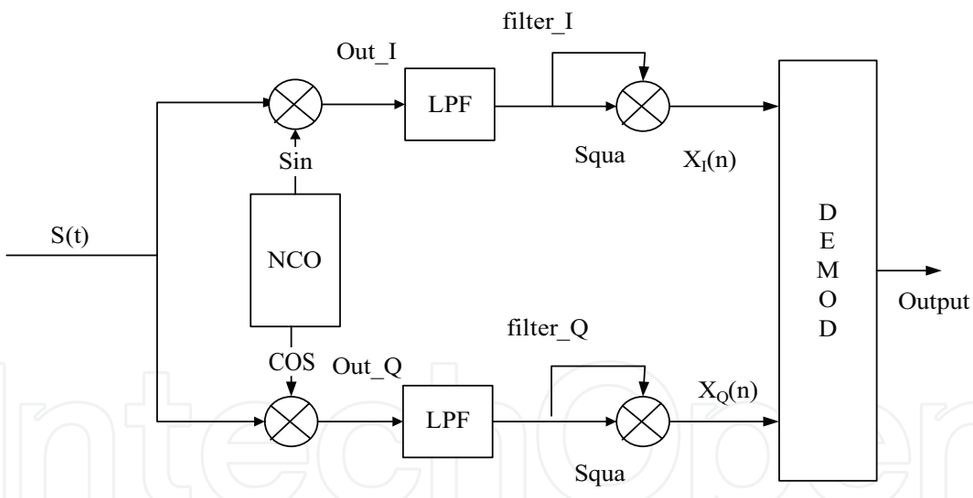


Fig. 7. ASK demodulation algorithm model

3. Receive link FIFO

In receive link, the FIFO has no multiple channels. All the 32bit data of one word would be written into FIFO in order. As the command length is no more than $32\times 5 = 160 bits$, FIFO depth is set to be 6 and the data are expressed by $data[0]-data[5]$. FIFO architecture is described in Fig. 8.

FIFO control module select PIE decode or Manchester decode to write into FIFO, where $data[0]$ stores the current protocol (i.e. Type B or C), tag number and some operation parameters. Several unoccupied bit in $data[0]$ are reserved for extension, whereas $data[1]-data[5]$ are used for command data.

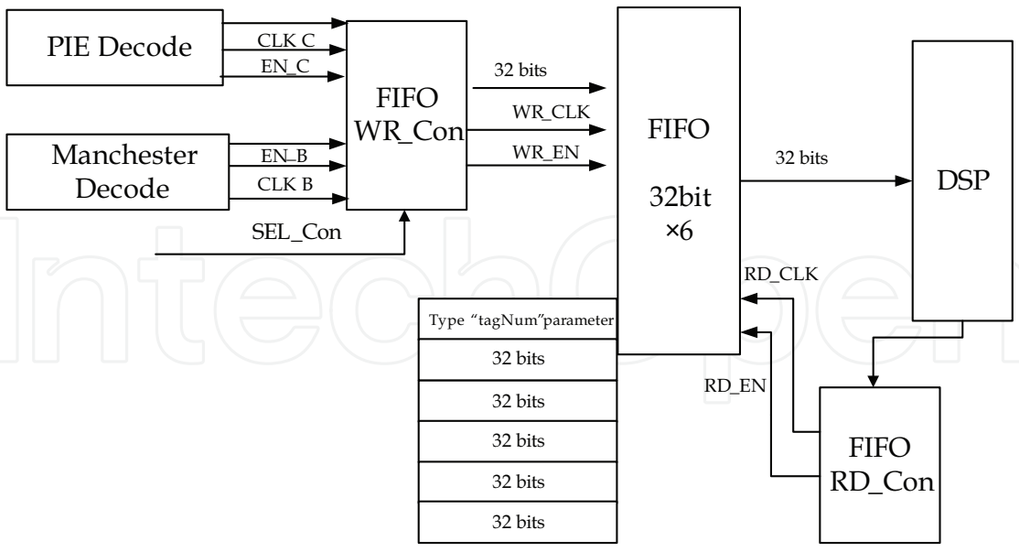


Fig. 8. Receive link FIFO design

After FIFO was written, external interrupt signal would be sent to DSP for specific command operation and state transit.

4. PIE decoding module

Pulse Interval Encode (PIE) distinguish 0 and 1 by different pulse interval, with one phase transit in the middle of any symbol. It has clock information to maintain better data synchronization and robust transmission under wireless environment.

Given $Tari = 12.5us$, length of 0 is $1Tari$ and length of 1 is $2Tari$. Detect CLOCK could be selected as follows,

$$f = \frac{1}{12.5} \times 1000 \times 2KHz = 160KHz \tag{6}$$

Pre-amble from reader to tag is shown as in Fig. 9.

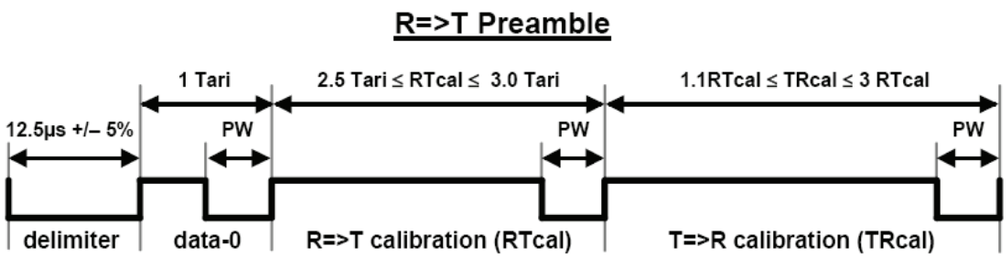


Fig. 9. Preamble for PIE

When one data, i.e. pulse arrives at the tag, its width was compared with the reference (half of the sum of "1" and "0" width). If the pulse width is larger than reference width, when the received data is "1", otherwise it is "0".

To make the two protocols, i.e. Type B and C share the same architecture FIFO, the data frame is designed as follows,

1st 32bit are protocol control and tags number;
2nd and 3rd 32 bit are command data;
And the last 3*32bit are reserved for backup.

3. Software design

3.1 Architecture

In the proposed tag simulator, the DSP exchange data only with FPGA, where FPGA chipset completes the protocol detection, tag number detection, however, quick command response, protocol switch, state transition, data feedback are finished by the DSP chipset. The whole simulator software workflow is shown as in Fig. 10.

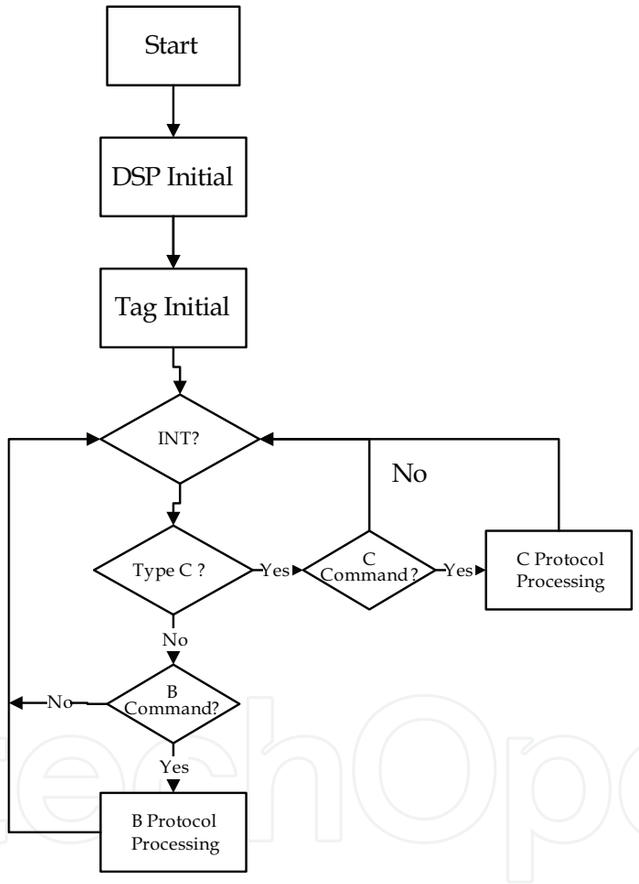


Fig. 10. Protocol processing workflow

3.2 Initialization of DSP

The DSP chipset has many peripherals and various registers, whose manually configuration will be complicated. Chip Support Library (CSL) function could be used to conveniently access DSP register and hardware resources to improve the development efficiently, where CSL_init () function is for loading and initializing these libraries. And PER_config () is mainly for configuration of given parameters,
PER_config([handle], *config Structure)

1. General purpose Input/Output

There are sixteen pins GPIO[15:0] to be set Input or Output. As Output, it could be used to control its driven state by writing inner register, while as input it could be used to detect input state by reading inner register.

Here, GPIO[11:9] are output for DSP to notice FPGA that all these tags response, and finish writing FIFO. FPGA could allocate FIFO data to each tag channel for FM0, Miller encoding, modulation and synthesizer. GPIO[7:4] is multiple used as EXT_INT [7:4], i.e. external interrupt, besides general purpose I/O.

2. External memory interface A/B

This simulator makes use of TMS320C6416 from TEXAS INSTRUMENT, which has EMIFA and EMIFB. EMIFA has 64pin memory bus with four spaces ACE0-ACE3 separately configurable and could be connected with SRAM, ROM or SDRAM. EMIFB has 16pin memory bus with BCE0-BCE3 four spaces. Both A and B have one external clock and two internal clock, CLK/4 and CLK/6.

3. Interrupt register

Interrupt is C6416's main work mode to control the peripheral device by executing the interrupt service routine. All the interrupt have priority level used by central processor to select. The main interrupt type is reset, NMI (non-mask interrupt) and mask interrupt, i.e. INT4-INT15.

- Reset interrupt

  It is the highest priority in the DSP to halt the processor's work and return to one known state. There are ten clock cycles before this interrupt signal change from valid low level into high level to guarantee successful configuration. The reset operation stops all the executed command and all these registers returns to the default state. In addition, it is not affected by transition command.

- Non –mask interrupt (NMI)

  It has one enable bit set to 1 for NMI work, which warns CPU the serious hardware interrupt.

- INT4-INT15

  These 12 interrupts are mask connecting to the peripheral, and they could also be controlled through software or made to be unused. The processor replies INT4-INT15 only when the following conditions are well met:

  a) Interrupt Flag=1
  b) No higher interrupts occur
  c) Enabler bit =1
  d) Global interrupt enabler bit=1

3.3 Tag initialization

Each tag in the working environment should be configured as known state including its register and memory through a loop process. For Type B, state register, random number register and 64bit ID are to be set, while for Type C, counter, Handle are also to be set besides what is similar as Type B.

3.4 Interrupt response

In the tag simulator, FPGA chipset get the command data and write the protocol type, tag number into FIFO, INT4 is set to 1 so that DSP call interrupt void c_extint4() to response.

All the command data in Type B are 8 bit, and if-else sentence is used to inquiry one by one. Flag_B is set to 1 if the command is coming.
However, in Type C there are many commands with less than 8 bit. They are stored in one byte from the most significant bit (MSB). And the Flag_c is set to 1 if the command is coming whatever it has 2 bit, 4 bit or 8 bit.

3.5 State transition for different protocol

Both Type B and Type C have the same process architecture for the state transition, where current status and the coming new command with parameters are response to feedback new data and transit into a new state. For example, ID state in Type B is processed as the following routine.

```
case state_b_ID:// unselect ->state_b_ready
   switch(the_com_typeb.COMMAND)
   {
   case bcmd_FAIL:   //for FAIL Command
      if (mytag_b[i].count != 0)
      {
         mytag_b[i].count++;  // Not zero, plus 1
      }
      else //random Number 0 or 1
      {
         mytag_b[i].count=random_counter_gen();//give random number
         if (mytag_b[i].count==0)
         {
            Write_FIFO(mytag_b[i].TID[0]);//reply the TID
            Write_FIFO(mytag_b[i].TID[1]);
         }
      }
      mytag_b[i].state=state_b_ID;  //Next state
      break;
   case bcmd_SUCCESS:  // SUCCESS command
      mytag_b[i].count--;  //minus 1
      if (mytag_b[i].count==0)
      {
         Write_FIFO(mytag_b[i].TID[0]); // reply the TID
         Write_FIFO(mytag_b[i].TID[1]);
      }
      mytag_b[i].state=state_b_ID;   //Next state
      break;
```

4. Experiment result

The main circuit board for tag simulator is shown as in Fig. 11.
SignalTap II in Quartus II is a practical tool to analyze the signal state by collecting internal node or I/O pins signal. CCS is used for DSP to compile, load and debug the program with step by step, register result tracking and set breakpoint etc.

In the FPGA chipset, FIFO get the DSP written data and allocate to each transmit link. At the same time, FIFO_buffer could know how many tags response based on the times reading FIFO and record this value. Signal synthesizer module gives the combined signal of multiple tags shown as in Fig. 12 in SignalTap II, i.e. two tags.

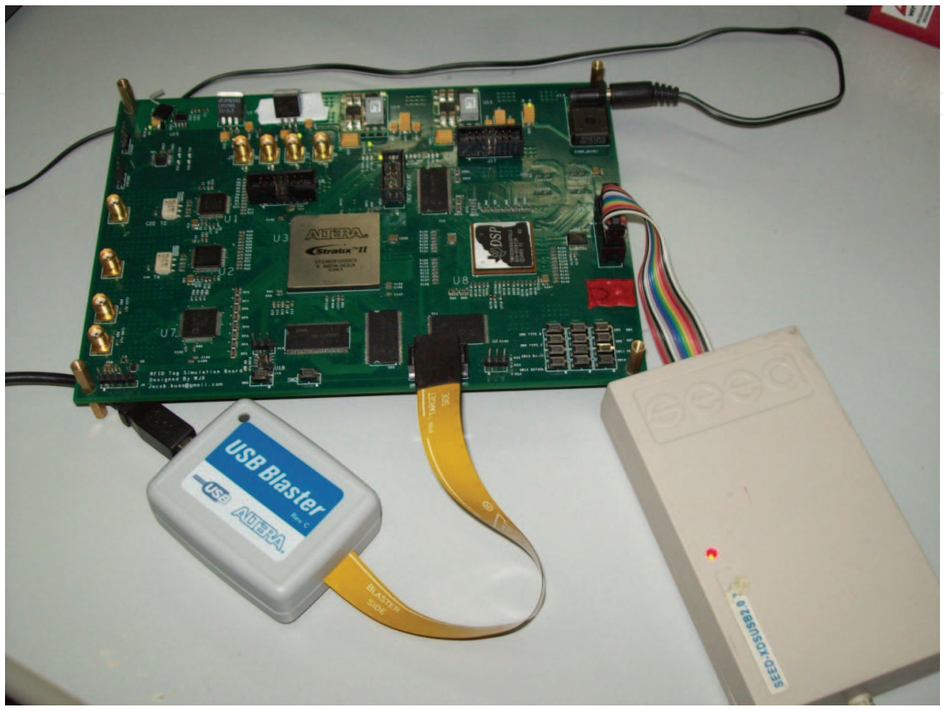


Fig. 11. Tag simulator main circuit board

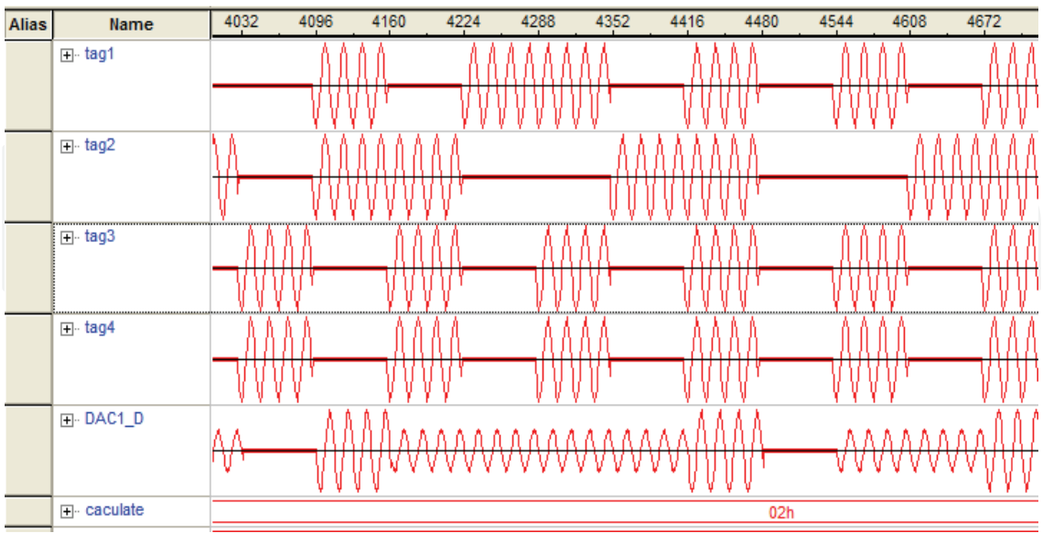


Fig. 12. Two tags synthesizer signal

If single tag returns data, the synthesizer signal will be tag1. When there are two tags response simultaneously, both tag3 and tag4 have no data and enencoding a series of “01”.

If only one of the two tags has modulation data, then the synthesizer signal is half of the original magnitude; If both the two tags are 0, final signal is also 0 as the combined output to the reader.

5. Summary

In this chapter, the UHF RFID tag simulator is designed for the anti-collision algorithm study and simulation, and even the application system architecture design. The key technologies including the protocol analysis, the hardware design, software design, debug and simulation, practical test result are given as a whole.

The traditional software radio method is adopted, that is, the FPGA and DSP chipset are selected as the main components to work as tag simulator. Both Type B and Type C protocol are implemented, and the signal synthesize is given for more than one tag. The circuit design in FPGA chipset and the work flow in DSP chipset are well introduced, such as the transmission link, receive link, the low pass filter module, digital orthogonal demodulation etc. The work mode based on the interrupt is also given in this Chapter. And finally the test result is shown as verification of the proposed tag simulator.

6. Reference

Zhang Yang, Wang Hui(2008). Design and Simulation of FIR Digital Filter Based on MATLAB and Quartus II. *Electronic Engineering*, Vol.34, No.8, pp.25-27 ,ISSN 1006-7787.

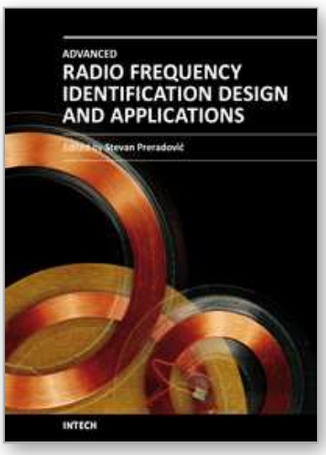
YUAN Jie,ZHAO Zhi-jin,ZHANG Fu-hong (2008). Design of Modem of RFID Based on FPGA. *Chinese Journal of Electron Devices*, Vol.31, No.5, pp. 1635-1638, ISSN 1005-9490

Shi Jingzhuo, Xu Meiyu, Xu Dianguo (2006). Manchester Encoder and Decoder Based on CPLD. *Electrotechnical Application*, Vol.25, No.5, pp.62-64, ISSN 1672-9560

TI Company(2004). TMS320C6000 DSP General-Purpose Input/output (GPIO) Reference Guide[J]. Literature Number: SPRU584A

TI Company(2003). TMS320C6000 Chip Support Library API Reference Guide. Literature Number: SPRU401

TI Company(2001). TMS320C6000 Peripherals Reference Guide. Literature Number: SPRU190



Advanced Radio Frequency Identification Design and Applications
Edited by Dr Stevan Preradovic

ISBN 978-953-307-168-8
Hard cover, 282 pages
Publisher InTech
Published online 22, March, 2011
Published in print edition March, 2011

Radio Frequency Identification (RFID) is a modern wireless data transmission and reception technique for applications including automatic identification, asset tracking and security surveillance. This book focuses on the advances in RFID tag antenna and ASIC design, novel chipless RFID tag design, security protocol enhancements along with some novel applications of RFID.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Bo Zhang, Dongkai Yang and Qishan Zhang (2011). Design and Implementation of a Multi-protocol UHF RFID Tag Simulation Platform, Advanced Radio Frequency Identification Design and Applications, Dr Stevan Preradovic (Ed.), ISBN: 978-953-307-168-8, InTech, Available from: http://www.intechopen.com/books/advanced-radio-frequency-identification-design-and-applications/design-and-implementation-of-a-multi-protocol-uhf-rfid-tag-simulation-platform

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2011 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen