Design and Implementation of a Multichannel Pulse Compression System Based on FPGA

Xiaojing Zhang
Department of Software Engineering, School of Software, Harbin University of Science and Technology, Harbin, China
Email: xjz81@163.com

Yajie Yue
School of Software, Harbin University of Science and Technology, Harbin, China
Email:jielu1981@yahoo.com.cn

Chenming Sha
School of Software, Harbin University of Science and Technology, Harbin, China
Email: shachm@163.com

***Abstract*—The implementation of digital signal process system based on FPGA is an important method for embedded system. Pulse compression can be implemented in digital method. The realization of a multichannel pulse compression system by using all digital method has the characteristics of high reliability, strong anti-disturbance, good flexibility and convenient for application. In essence, pulse compress is a method of frequency spectrum expanding, and is used in matched filtering. It incarnates the matching level of filter and the expectant phase of received signal. A multichannel pulse compression system is designed using FFT IP core which can be reused in different periods of digital pulse compression, respectively performing FFT and IFFT calculation, so that the hardware consumption is saved significantly. This paper presents the logic programming of Multichannel Pulse Compression System based on Xilinx FPGA, and introduces the high-speed transmission module, the digital signal process module and the data buffer module in detail.**

***Index Terms*—FPGA, Pulse Compression, Xilinx, System Generator**

I. INTRODUCTION

With the development of the high-speed large scale integrated circuit, generating linear frequency modulated signal and pulse compression can be implemented in digital method. Besides the matched- filtering processing the digital pulse compression system can also apply sidelobe suppression[1]. It can decrease the size of the system and it also has high stability and maintainability, and this can improve the programmable ability of the system. So the digital processing method is of catholic concern and has been widely applied[2].

Digital pulse compression processing of the linear FM usually adopts the double channels of perpendicular processing scheme to avoid the effect of Echo Signal's random-phase, which may reduce the loss of system processing by about 3dB, meantime, this method can reduce the demand for AD acquisition devices[3].

The current FPGA chip has been becoming the core component of digital signal processing system, no longer playing the role of glue logic[4]. In the chip, it includes not only the logical resources as well as multiplexers, memory, hard-core multiply-add units and embedded processors and other equipments, but also can be provided with the ability of highly parallel computing, which has made FPGA become the ideal device of high-performance digital signal processing, particularly suitable for completing digital filtering, fast Fourier transform etc.

This design uses Xilinx's Virtex5 FPGA family, and achieves the pulse compression processing on AD collection down-converted signal by the method of frequency domain pulse pressure, meantime, it uses System Generator to perform the program's development in digital signal processing, which is the Xilinx's latest integrated development tools for digital processing system[5][6].

The structure of the paper is as follows. Section II introduces the chosen process platform. Section III introduces the system function and structure. Section IV introduces the design of the key modules in FPGA program. And Section V gives the simulation and implementation results.

II THE CHOSEN PLATFORM

The FPGA device in this system use is XC5VSX95T(Virtex-5). The Virtex-5 family provides the newest most powerful features in the FPGA market. Using the second generation ASMBL™ (Advanced Silicon Modular Block) column-based architecture, the Virtex-5 family contains five distinct platforms (sub-families), the most choice offered by any FPGA family. Each platform contains a different ratio of features to address the needs of a wide variety of advanced logic designs. In addition to the most advanced, high-performance logic fabric, Virtex-5 FPGAs contain many hard-IP system level blocks, including powerful 36-Kbit

doi:10.4304/jcp.8.1.121-128

block RAM/FIFOs, second generation 25 x 18 DSP slices, SelectIO™ technology with built-in digitally controlled impedance, ChipSync™ source-synchronous interface blocks, system monitor functionality, enhanced clock management tiles with integrated DCM (Digital Clock Managers) and phase-locked-loop (PLL) clock generators, and advanced configuration options. Additional platform dependant features include power-optimized high-speed serial transceiver blocks for enhanced serial connectivity, PCI Express® compliant integrated Endpoint blocks, tri-mode Ethernet MACs (Media Access Controllers), and high-performance PowerPC® 440 microprocessor embedded blocks. These features allow advanced logic designers to build the highest levels of performance and functionality into their FPGA-based systems. Built on a 65-nm state-of-the-art copper process technology, Virtex-5 FPGAs are a programmable alternative to custom ASIC technology. Most advanced system designs require the programmable strength of FPGAs. Virtex-5 FPGAs offer the best solution for addressing the needs of high-performance logic designers, high-performance DSP designers, and high-performance embedded systems designers with unprecedented logic, DSP, hard/soft microprocessor, and connectivity capabilities. The Virtex-5 LXT, SXT, TXT, and FXT platforms include advanced high-speed serial connectivity and link/transaction layer capability.

III. System Function And Structure

Pulse compression system receives the six-channel PRT wave gate acquired data transmitted by AD acquisition card, which is processed through the digital-down-conversion and its data format is IQ two-way 16 bit fixed-point. After the six-channel pulse fix-point compression processing performed by the pulse compression system, its data format is converted to single-precision floating-point, then is transmitted to the DSP board for further processing.

The main parameters on pulse compression processing are as follows: Pulse compression points: 1024, 2048 and 4096. Input format of digital down conversion: I/Q two-way, 16bit signed integer. Data format after pulse compression: I/Q two-way, standard single-precision floating point. Frequency of AD down-convert signal acquisition: 70MHz IF, 30MHz bandwidth. Channels after pulse compression parallel processing:6. Time requirements on single pulse compression processing: 70us. Method of pulse compression processing: frequency domain approach. Speed of data transmission: 800MB/s.

The main modules of pulse compression system includes the Rapid IO transmission interface, the Control-FPGA module, the RocketIO trans-mission interface, the digital signal processing FPGA module and PCI bus interface.

The Rapid IO transmission interface is used for high-speed data transmission between boards. As for this pulse compression system, it's mainly used to receive the data after AD sampling and digital down converter and transmit the data after pulse compression to the DSP board which makes further data processing.

The Control-FPGA is used for data transmission and the system working state control. The data transmission includes PCI transmission, serial Rapid IO transmission and serial RocketIO transmission.

PCI bus interface is used for the transmission of control and status information between the upper machine and the Control-FPGA.

RocketIO transmission interface is used for the high-speed transmission between the Control-FPGA and the signal-processing FPGA.

The main tasks of the signal-processing FPGA include receiving the data after AD sampling and digital down converter, multichannel pulse compression process and transmitting the data after pulse compression to the DSP board. The signal-processing FPGA is the core of the pulse compression system. The following section mainly introduce the software structure of the signal-processing FPGA and the design of the key modules. As shown in Fig1.

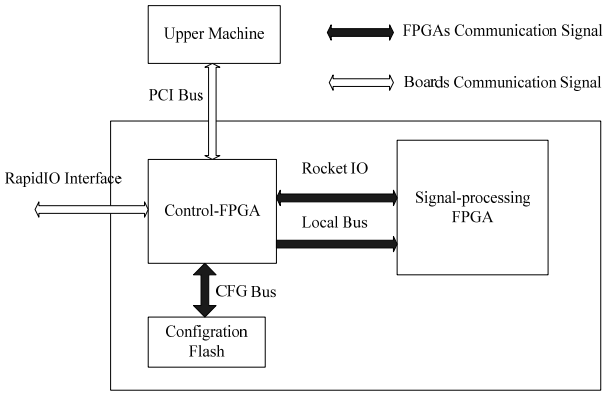


Figure 1. The structure block diagram of the hardware system

The software structure block diagram of the signal-processing FPGA is shown in Fig2. The signal-processing FPGA uses Rocket IO for the high-speed data transmission between the Control-FPGAs. The reference clock of the Rocket IO module is 125MHz. Six-channel pulse compression module uses parallel processing, and it employs FFT and multipliers, adders and other logic IP cores to achieve specific development, which is provided by Xilinx.

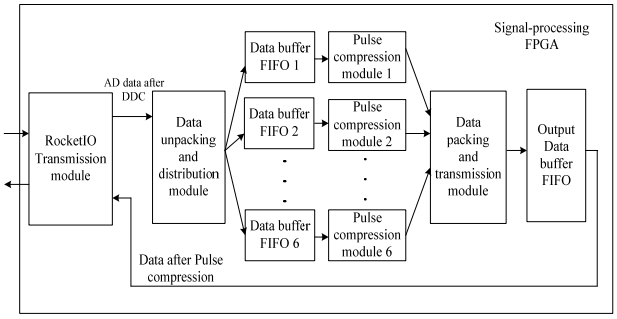


Figure 2. The software structure block diagram of the signal processing FPGA

IV. DESIGN OF THE KEY MODULES

The Key technology modules of FPGA in the signal-processing are mainly the high-speed serial transceiver module, pulse compression module and the data buffer module. The following will focus on the introduction of program design of the three modules.

A. The Design of High-Speed Serial Transceiver Module

Rocket IO, which is a programmable high-speed serial transceiver integrated in Xilinx FPGA chip, uses two pairs of differential pair to transmit and receive data, thus can achieve two simplex work or a pair of two full-duplex's data transmission. Rocket IO also supports the full-duplex transmission rate in the range of 622Mbit/s to 3.75Gbit/s, also has channel bonding, 8B/10B encoding and decoding (balanced code), clock generation and recovery technology and other functions, which can be ideally applied to the high-speed serial data transceiver mode of chips or backplane[7]. The Rocket IO structure as shown in Fig3.

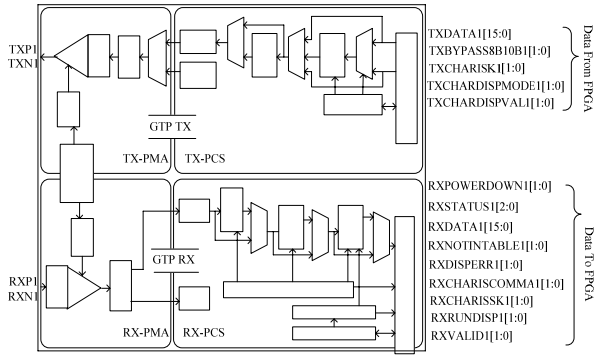


Figure 3. Virtex5 FPGA Rocket IO structure

Rocket IO consists of PMA (physical medium adaptation layer) and PCS (Physical Coding Sub-layer) two parts. Among them, PMA sub-layer is mainly used for serialization and de-string, PCS includes circuit encoding and CRC encoding.

Rocket IO in the signal processing FPGA is used for the transmission and control of high-speed data between FPGAs. Its bit rate is 2.5Gbit/s.

Rocket IO uses 4X mode(four serial transceivers are combined into one group to transmit and receive), and channel bonding modes whose channel bonding code is 5C and channel bonding length is 8 bytes, and comma code is BC. Channel bonding is used for the data alignment between four channels, while comma code is used for the bytes separation between channels. Channel bonding cancels the skew between GTP lanes by using the RX elastic buffer as a variable latency block. The transmitter sends a pattern simultaneously on all lanes, which the channel bonding circuit uses to set the latency for each lane so that data is presented without skew at the FPGA RX interface[8]. Channel Bonding Conceptual View As shown in Fig4.

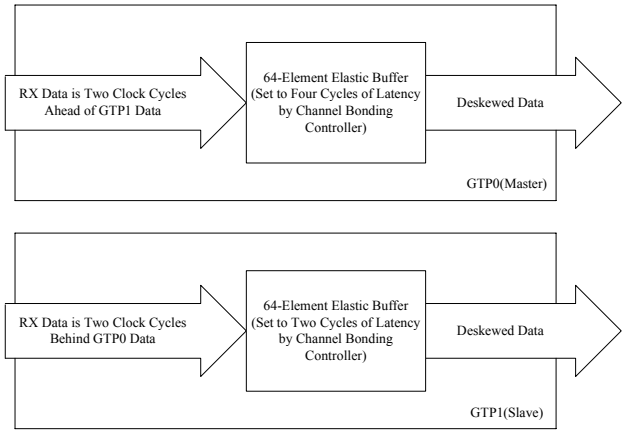


Figure 4. Channel Bonding Conceptual View

For other modules inner FPGA, they provide the receiving and transmitting data ports, which have a finite state machine to control the logic timing respectively. Structure diagram as shown in Fig5.

Through the test, Rocket IO's transmission rate can be more than 800MB / s in the system.

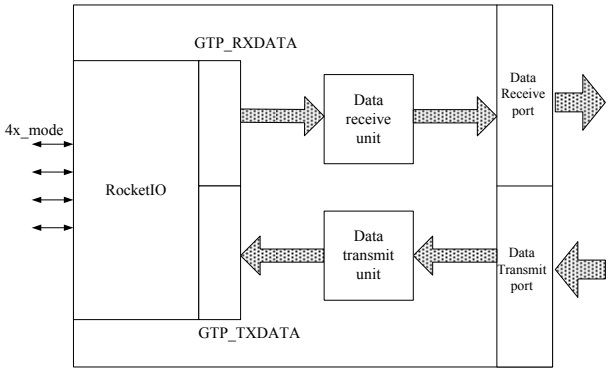


Figure 5. The program structure of the signal-processing FPGA Rocket IO module

B. The Design of Pulse Compression Module

The linear frequency modulation pulse compression can be implemented in time domain or frequency domain[9]. In time domain, the process of the pulse compression can be expressed as:

$$y(n) = x(n) * h(n) = \sum_{k=0}^{N-1} x(k)h(n-k) \qquad (1)$$

In frequency domain, the process of the pulse compression can be expressed as:

$$y(n) = IFFT[X(\omega)] \cdot H(\omega)] \qquad (2)$$

$$= IFFT\{FFT\{[x(n)] \cdot FFT[h(n)]\}$$

The time-domain digital pulse compression system usually uses the FIR filter, which is implemented by convoluting two finite-length sequences. The real part and imaginary part of the output data needs correlation operation. The operational volume of FIR complex-number correlation process gradually increases

as the signal time width increases. And the magnitude of the processing device also increases. So, the time-domain pulse compression has advantages when the time width of the signal is not wide [10]. But when the width increases, the amount of the processer will increase.

For the frequency-domain pulse compression, when the time width of the signal is wide, the magnitude of the processing device doesn't increase much, because the process system is based on the high-speed and effective FFT device[11]. The operational volume of the time domain pulse compression is N2, and N is the number of the input data sequence. If Radix-2 FFT Algorithms is applied, the operational volume of the frequency-domain pulse compression can be reduced to $(1/2)N[\log_2 N]$ [12]. So compared with the time-domain process, the frequency-domain process has large benefits. Besides, because the FFT and digital signal process technology develops, the operational volume becomes smaller and the process speed becomes faster. So we choose the frequency domain method in the pulse compression system.

The main parts of the frequency-domain pulse compression processs is the input data buffer unit, FFT unit, complex multiplication unit, IFFT unit, data-format converting unit and the output data buffer unit. The frequency-domain pulse compression processs structure as shown in Fig6.

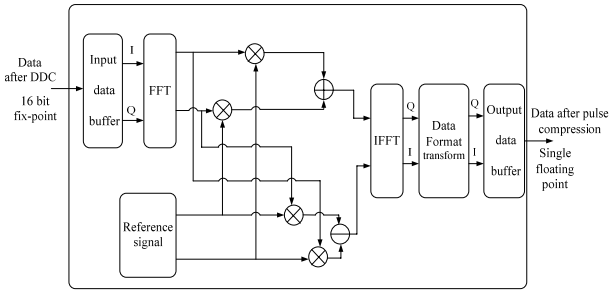


Figure 6. The frequency-domain pulse compression processs structure

The pulse compression module is the core of the signal processing FPGA. We use System Generator to develop the FPGA program of the pulse compression module. System Generator is a DSP design tool from Xilinx that enables the use of the Mathworks model-based Simulink design environment for FPGA design. Previous experience in Xilinx FPGAs or RTL design methodologies is not required when using System Generator. Designs are captured in the DSP friendly Simulink modeling environment using a Xilinx specific blockset. All of the downstream FPGA implementation steps including synthesis and place and route are automatically performed to generate an FPGA programming file.

System Generator have many Key Features. Build and debug high-performance DSP systems in Simulink using the Xilinx Blockset that contains functions for signal processing (e.g., FIR filters, FFTs), error correction (e.g., Viterbi decoder, Reed-Solomon encoder/decoder), arithmetic, memories (e.g., FIFO, RAM, ROM), and digital logic.

Automatic code generation of VHDL or Verilog from Simulink. Implement behavioral (RTL) generation and target specific Xilinx IP cores from the Xilinx Blockset. System Generator also supports custom HDL through its HDL import flow.

Hardware co-simulation. A code generation option that allows you to validate working hardware and accelerate simulations in Simulink and MATLAB. System Generator supports Ethernet (10/100/Gigabit) and JTAG communication between a hardware platform and Simulink.

Xilinx Power Analyzer (XPA) Integration. Integration with XPA enables designers to analyze power requirements and explore design modifications to meet their power targets.

Hardware/software co-design of embedded systems. Build and debug DSP co-processors for the Xilinx MicroBlaze™ soft processor core. System Generator provides a shared memory abstraction of the HW/SW interface, automatically generating the DSP the bus interface logic, software drivers, and software documentation .

The FPGA program structure of the pulse compression in the System Generator as shown in Fig7. It realizes the digital signal process by FFT, adder, multiplication, ram and other logic IP core.

The data from the AD acquisition board has been processed by DDC (digital down converter), and the data format is 16-bit signed integer. The data after pulse compression will be transmitted to the DSP board for the further process. Because the data format in the DSP is floating-point, in the last stage of pulse compression FPGA program is the format converting.

This system use the FFT core of the Xilinx to implement FFT and IFFT operation. The Xilinx LogiCore™ IP Fast Fourier Transform (FFT) implements the Cooley-Tukey FFT algorithm, a computationally efficient method for calculating the Discrete Fourier Transform (DFT). The FFT core can work at very high clock frequency (395MHz), and has good reliability.

The FFT core computes an N-point forward DFT or inverse DFT (IDFT) where N can be 2m. For fixed-point inputs, the input data is a vector of N complex values represented as dual bx-bit two's complement numbers, that is, bx bits for each of the real and imaginary components of the data sample, where bx is in the range 8 to 34 bits inclusive. Similarly, the phase factors bw can be 8 to 34 bits wide. For single-precision floating-point inputs, the input data is a vector of N complex values represented as dual 32-bit floating-point numbers with the phase factors represented as 24-bit or 25-bit fixed-point numbers.

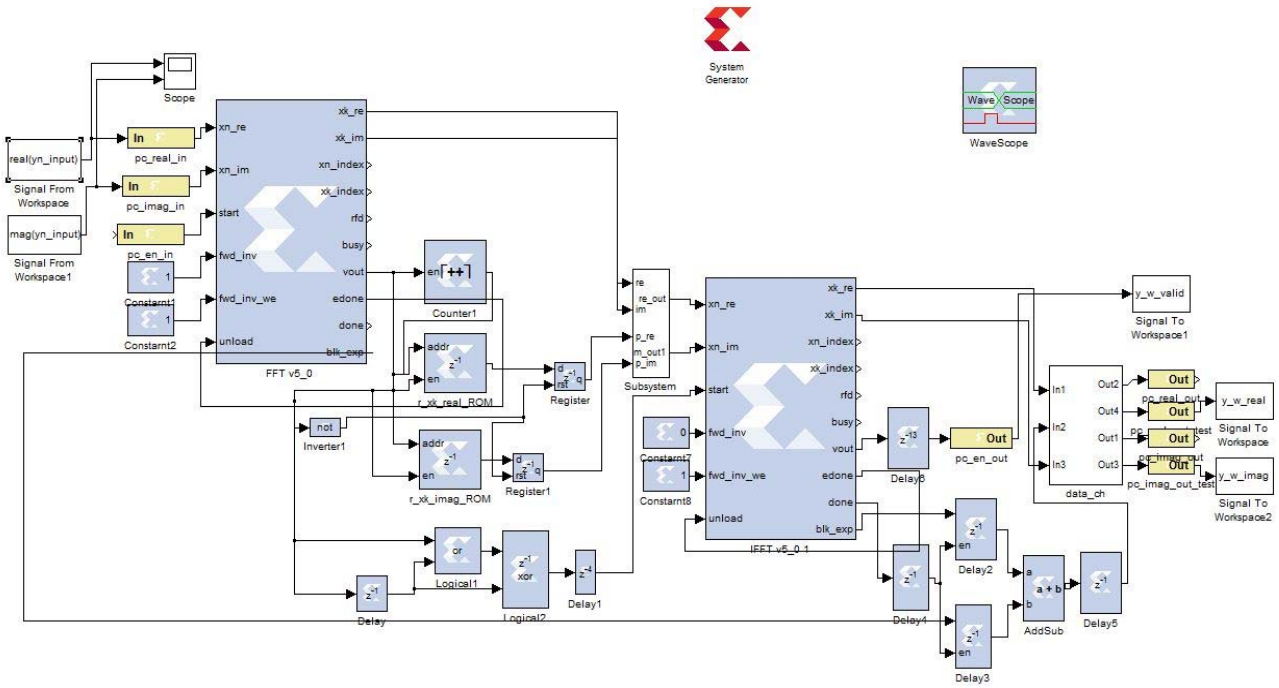


Figure 7. Pulse compression process structure in System Generator\

Though the FFT core supports floating-point operation, we weight the operation volume and output data precision, and we choose the fixed-point way.

Three arithmetic fixed-point options are available for computing the FFT:

Full-precision unscaled arithmetic.

Scaled fixed-point, where you provide the scaling schedule.

Block floating-point (run-time adjusted scaling).

With block floating-point, each stage applies sufficient scaling to keep numbers in range, and the scaling is tracked by a block exponent. The block floating-point mode may use significantly more resources than the scaled mode, as it must maintain extra bits of precision to allow dynamic scaling without impacting performance. Therefore, if the input data is well understood and is unlikely to exhibit large amplitude fluctuation, using scaled arithmetic (with a suitable scaling schedule to avoid overflow in the known worst case) is sufficient, and resources may be saved.

The FFT core provides four architecture options to offer a trade-off between core size and transform time. Four architecture options are available:

Pipelined Streaming I/O – Allows continuous data processing.

Radix-4 Burst I/O – Loads and processes data separately, using an iterative approach. It is smaller in size than the pipelined solution, but has a longer transform time.

Radix-2 Burst I/O – Uses the same iterative approach as Radix-4, but the butterfly is smaller. This means it is smaller in size than the Radix-4 solution, but the transform time is longer.

Radix-2 Lite Burst I/O – Based on the Radix-2 architecture, this variant uses a time-multiplexed.

To achieve maximum operation speed, we choose the pipelined streaming I/O operation.

The data format of the pulse compression FPGA program is fixed-point, and the data format in DSP (Ts201) is single floating point. So the last operation of the pulse compression is data transformation. We use Xilinx Floating-Point core and other logic to implement the data format transformation. The Xilinx Floating-Point core provides designers with the means to perform floating-point arithmetic on an FPGA device[13]. The core can be customized for operation, word length, latency, and interface.

Block Diagram of Generic Floating-Point Binary Operator Core as shown in Fig 8.

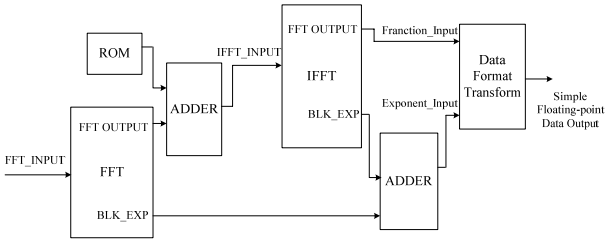


Figure 8. Block Diagram of Generic Floating-Point Binary Operator Core

When use this core to achieve fixed-point to floating-point data transform, the two inputs A and B are the fraction input and exponent input, and the output result is the floating-point data output. Signal floating-point number uses 32 bits, with a 24-bit fraction and 8-bit exponent[14]. We add the two block-floating outputs of the FFT and IFFT, and make it as the exponent input of the core. And we use the output of the IFFT core as the fraction input of the core. This method not only transforms the fixed-point format to the single floating-

point format, but also uses the block-floating output to weight the IFFT output, which is final pulse compression output.

C. The Design of Data Buffer Module

In the pulse compression system, data buffer module is used mainly for receiving the data acquisition after down converter transmitted by ADC board, after meeting the data requirements on a pulse compression, the data waiting to receive is transmitted to the pulse compression module for processing. In addition, a buffer is required for the data after pulse compression processing, After meeting the size of Rocket IO transmission packet only once, the data after processing is transmitted to Rocket IO module for external sending[15]. The asynchronous FIFO is used to achieve the data buffer design in this design.

Virtex-5 FPGA provides two memory structures: Distributed memory architecture and block memory structure. Distributed Memory (Distributed Select RAM) is achieved by the CLB's lookup table (LUT). Block memory (Block RAM) is a special memory module in the FPGA, each 18Kb, number varies as the device size, which can be configured for single or dual port Block RAM. Compared with the distributed storage structure, block storage architecture can achieve higher clock speed, therefore, we used block memory to achieve the asynchronous FIFO in the design. Block RAM schematic symbol as shown in Fig9.

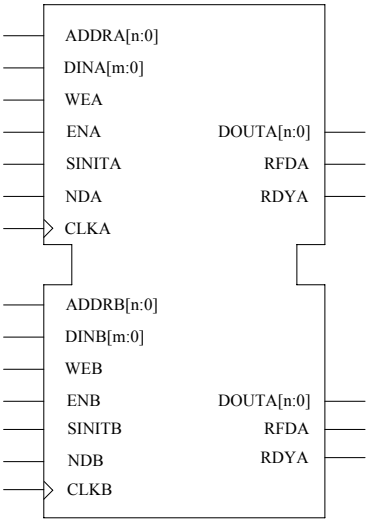


Figure 9. Block RAM schematic symbol

The block RAM in Virtex-5 FPGAs stores up to 36K bits of data and can be configured as either two independent 18 Kb RAMs, or one 36 Kb RAM. Each 36 Kb block RAM can be configured as a 64K x 1 (when cascaded with an adjacent 36 Kb block RAM), 32K x 1, 16K x 2, 8K x 4, 4K x 9, 2K x 18, or 1K x 36 memory. Each 18 Kb block RAM can be configured as a 16K x 1, 8K x2 , 4K x 4, 2K x 9, or 1K x 18 memory. Similar to the Virtex-4 FPGA block RAMs, Write and Read are synchronous operations, the two ports are symmetrical and totally independent, sharing only the stored data. Each port can be configured in one of the available widths, independent of the other port. In addition, the read port width can be different from the write port width for each port. The memory content can be initialized or cleared by the configuration bitstream. During a write operation the memory can be set to have the data output either remain unchanged, reflect the new data being written or the previous data now being overwritten.

In the Virtex-5 architecture, the special logic in the Block RAM enable users to easily achieve synchronize or multiple rate (asynchronous) FIFO. This eliminates using other CLB logic since for the counter, comparator, or markers' generating, Instead, each FIFO can use only a Block RAM. Standards and the first words fall through (FWFT) can support both.

The Xilinx LogiCORE™ IP FIFO Generator is a fully verified first-in first-out (FIFO) memory queue for applications requiring in-order storage and retrieval. The core provides an optimized solution for all FIFO configurations and delivers maximum performance (up to 500 MHz) while utilizing minimum resources. Delivered through the Xilinx CORE Generator™ software, the structure can be customized by the user including the width, depth, status flags, memory type, and the write/read port aspect ratios. The FIFO Generator core supports Native interface FIFOs and AXI4 interface FIFOs. The Native interface FIFO cores include the original standard FIFO functions delivered by the previous versions of the FIFO Generator (up to v6.2). Native interface FIFO cores are optimized for buffering, data width conversion and clock domain decoupling applications, providing in-order storage and retrieval.

Top-Level View of FIFO in Block RAM as shown in Fig 10.

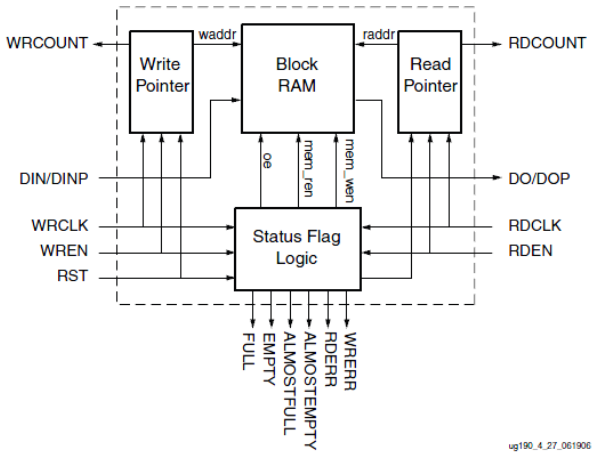


Figure 10. Top-Level View of FIFO in Block RAM

V. Simulation Results

Digital block was partitioned into hierarchy modules firstly, the RTL coding of sub-modules have been simulated in ModelSim. The simulation result as shown in Fig11. The last waveform line is the input LFM signal. The last two line waveforms are the real part and imaginary part of the data after pulse compression. So,

the functional simulation results in the ModelSim verify the design and the code before downloading the FPGA program ,which is the necessary step in the FPGA design to avoid the hardware damage.

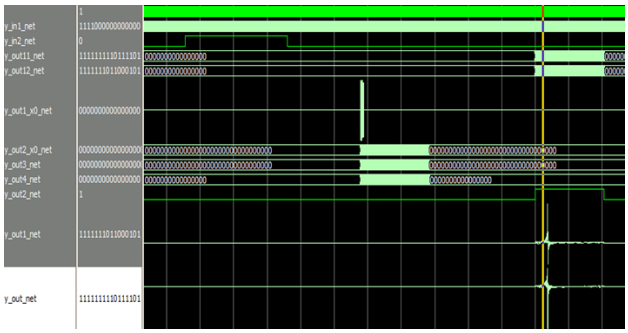
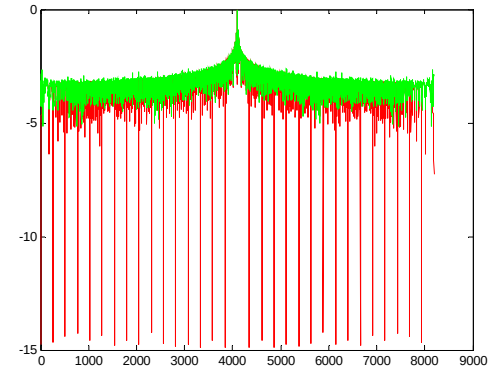
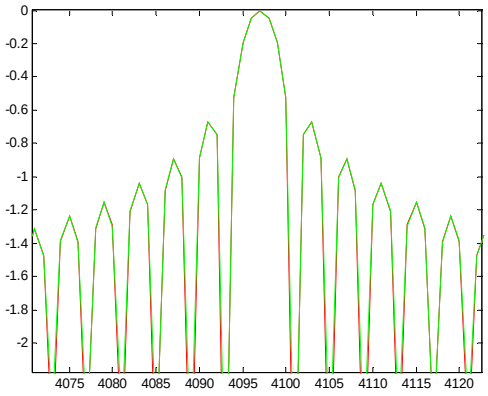


Figure 11. Modelsim simulation results

The FPGA program fixed-point calculation result and Matlab floating-point result have been shown in Fig12.The green one is the FPGA program result, and the red one is the Matlab result. The floating-point result has a wider dynamic range, but The PSLR and ISLR of the two are similar.The FPGA calculation result meets the requirement of the DSP process.



(a)



(b)

Figure 12. The comparison of FPGA fixed-point calculation result and Matlab floating-point result

VI. CONCLUSION

This paper introduces the software design of the signal-processing FPGA in the multichannel pulse compression system. We develop the FPGA program by ISE and System Generator, and use ModelSim and Chipscope for the function simulation and on-line debugging.

In FPGA program, the high speed control and transmit modules are suitable for design with HDL in the ISE software, and the digital signal process modules are suitable for design in System Generator software. Designing the digital signal process modules with System Generator can greatly increases the efficiency of software development without losing its accuracy. So System Generator will be the preferred tool in the FPGA digital signal process development.

REFERENCES

[1] Jincheng Xiang and Mingyou Zhang, “Radar System”, Electronics industry Press, Beijing, China, 2001, pp. 21-26.
[2] A. Giorgetti, M. Chiani, D. Dardsari, R. Piesiewicz and G. H. Bruck, “The cognitive radio paradigm for ultra-wideband systems: the european project EUWB,” Proceedings of the 2008 IEEE International Conference on Ultra-wideband (ICUWB2008), Vol. 2, pp. 169-172, September 2008.
[3] Li Haining, Zhao Jian, HongWen, Yu Bianzhang and XiWei “SAR Imaging Algorithm Based on Fractional Fourier Transform,” Journal, Journal of Telemetry, Tracking, and Command Vol. 28, No.1 January 2007, pp. 20~24
[4] Xilinx, Inc. “Virtex5 FPGA RocketIO GTP Transceiver User Guide,” 2008, pp. 12-14
[5] Xiaoniu Yang, Caiyi Lou and Jianliang Xu. “Principle and Application of Software Radio”, Secend Edition, Electronics industry Press, Beijing, China, 2002.
[6] Xilinx, Inc. “LogiCORE IP Virtex-5 FPGA RocketIO GTP Transceiver Wizard Getting Started Guide,” 2010.
[7] Zhiquan Ding, “LFM pulse compression system design and FPGA implementation,” dissertation, University of Electronic Science and Technology, China, 2006.
[8] Yucun Zeng, Baojun Zhang, Tingzhi Shen and Xiaoying Tang, “Signal and System”, Second Edition, The Press of Beijing University of Technology, 2006.
[9] Xilinx,Inc. “System Generator for DSP Getting Started Guide,” 2010
[10] Xilinx.Inc. “Fast Fourier Transform v6.0 datasheet,” 2008.
[11] ANSI/IEEE, IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Standard 754-1985. IEEE-754.
[12] ADSP-TS201TigerSHARC Embedded Processor Data Sheet, Analog Device, 2005
[13] Yiping Wang, Tianzhu Wen and Yongshuang Shang, “Research on Development and Test of Signal-oriented Driver,” Joural of Software, Vol. 7, No. 8, pp. 1807-1815, 2012.
[14] Yuehua Zhang, Zhibin Chen and Xinhe Zhang, “The Design and Research of FFT Processor Based on FPGA,” Journal,Chinese Instrument, 2006, pp. 50-53.
[15] Yuping Hu, Zhijian Wang, Huiliu and Guangjun Guo, “A Geometric Distortion Resilient Image Watermark Algorithm Based on DWT-DFT,” Journal of Software, Vol. 6, No. 9, pp. 1805-1812, 2011.

Xiaojing Zhang received the B.S. degree in Communication Engineering from Northeast Dianli University in 2002. And she received the M.S. degree in Software Engineering from Yunnan University in 2005. Her primary research area focused on Embeded System. Currently she is a Lecturer of the School of Software, Harbin University of Science and Technology, Harbin, China.

Yajie Yue received the B.S. degree in automation from Harbin University of Science and Technology in 2004. And she received the M.S. degree in Control Theory and Control Engineeringfrom Harbin University of Science and Technology in 2007. She has been working in Harbin University of Science and Technology from 2007. Currently her primary research area focused on IC design and Computer Architecture.

Chenming Sha was born in Heilongjiang Province, China, in 1981. She received the B.S. degree in Computer Science and Technology from Beijing Normal University, China in 2004, and M.S. degree in Computer application technology from Beijing University of Technology, China in 2007. Now she is working at Harbin University of Science and Technology, Harbin, China as a Lecturer.