Several Implementation Methods of Signal Processing Algorithm Based on FPGA

Several Implementation Methods of Signal Processing Algorithm Based on FPGA

Wang Xin, Hu Zhi-Qiang, Jiang Da-Qing, Sun Zhi-Xiao, Zhang Yu and Liu Yong[*]

College of Electronic Engineering, Heilongjiang University, Harbin 150080, China

[*]E-mail: liuyong@hlju.edu.cn

Abstract: Field Programmable Gate Array (FPGA) devices are widely used in the field of Digital Signal Processing (DSP), and the algorithm of DSP has been used in many fields such as speech signal processing, audio signal processing, image processing, information system, control system and so on. In the development and design of modern digital systems, the use of programmable logic devices is becoming more and more common. There are many ways to implement signal processing algorithm based on FPGA: the first one is to design with VHDL or Verilog HDL language; the second one is to use mature IP core encapsulated by FPGA company; the third one is to design with EDA tools such as MATLAB fdatool/Simulink/DSP Builder/Modelsim/Quartus II. Taking the digital filter as an example, this paper uses these three methods to realize the filtering algorithm, and introduces the detailed design steps and the final simulation verification.

Keywords: FPGA; VerilogHDL; IP core; Fdatool; DSP Builder; Digital Signal Processing

1. Introduction

The rapid development of digital signal processing and related chips is closely related to digital filtering. Digital filtering has long been a research hotspot [1]. Commonly used digital filters can be divided into finite impulse response (FIR) filters and infinite impulse response (IIR) filters. Two FIR filters (Finite Impulse Response) are filters of finite-length unit impulse response. It is a non-recursive filter [2] if it is classified by recursive type.

Usually digital signal processing implementations can be roughly divided into two categories. The first type is implemented on a PC by software, and is implemented on a general-purpose computer using a computer language such as C/C++, MATLAB, and the like. This method is mostly used for teaching or algorithm simulation, and can not achieve real-time [3]. The other is board-level implementation on hardware: one is using traditional DSP processing chip programming; the other is using Application Specific Integrated Circuits (ASIC); the third is using programmable logic devices. (CPLD/FPGA). FPGA has a full parallel processing architecture, which has advantages in real-time signal processing, portable code, etc. [4], has been accepted by the majority of users, and more examples are used to implement algorithms.

FIR filters play a very important role in the application of digital signal processing because they can obtain strict linear phase, do not produce large amplitude oscillation offset in the calculation of limited precision, and their operation speed is in the digital filter. At the forefront, the advantages are favored by the majority of users.

2. FIR filter FPGA design and implementation

2.1 Theoretical basis of FIR filter

Through the filtering operation, a set of input data sequences is transformed into another set of output data sequences, thereby realizing the change of signal properties in the time domain or frequency domain [5]. A digital filter is a linear time-invariant (LTI) discrete-time system implemented by a finite-precision algorithm for implementing digital signal filtering.

The expression of the FIR filter shown in equation (1) can be expressed by a K-order convolution sum:

$$\mathrm{y(n)} = \sum_{k=0}^{k-1} \hbar(k) \bullet x(n-k) \tag{1}$$

Where: k: the number of taps of the FIR filter, ie the order of the filter;
$h(k)$: the k-th tap coefficient, ie the unit impulse response;
$x(n-k)$: the input signal delayed by K taps.

Convolution is one of the most frequently used basic operations for arithmetic signal processing. It describes the interaction between the system input and the system to produce an output. Typically, the convolved system output will be the input delay, attenuation, or amplification.

If the unit pulse coefficient of the filter satisfies the condition of (2) (symmetric or antisymmetric), the filter has a linear phase characteristic, ie:

$$\mathrm{h(n) = h(-n) \text{ or } h(n) = -h(n)} \tag{2}$$

FIR filters have a variety of basic structures that are the basis for FPGA implementation. Although the FPGA is used to implement a certain FIR filter, it is necessary to adopt the corresponding implementation form of the FPGA [6]. However, no matter which FPGA implementation form is adopted, the basic structure of the FIR filter to be implemented is first determined [7], according to the basic structure of the FIR, and then proceed to choose the implementation structure of the FPGA. This article is based on different EDA tools to achieve different filter structure. In general, the basic structure of FIR can be divided into four types: direct type, cascade type, frequency sampling type and fast convolution type.

2.2 FIR filter design method

There are three methods for FIR filter design: window function method, frequency sampling method and optimization method based on firls function and remez function. The most basic design method is to require the frequency response of the designed FIR filter to approximate the frequency response required by the performance index. This paper uses the commonly used design method and window function design method in the design of FIR digital filter.

Commonly used window functions are: triangular window, rectangular window, hanning window, hamming window, Blackman window, Kaiser window. The basic parameters are shown in Table 1.

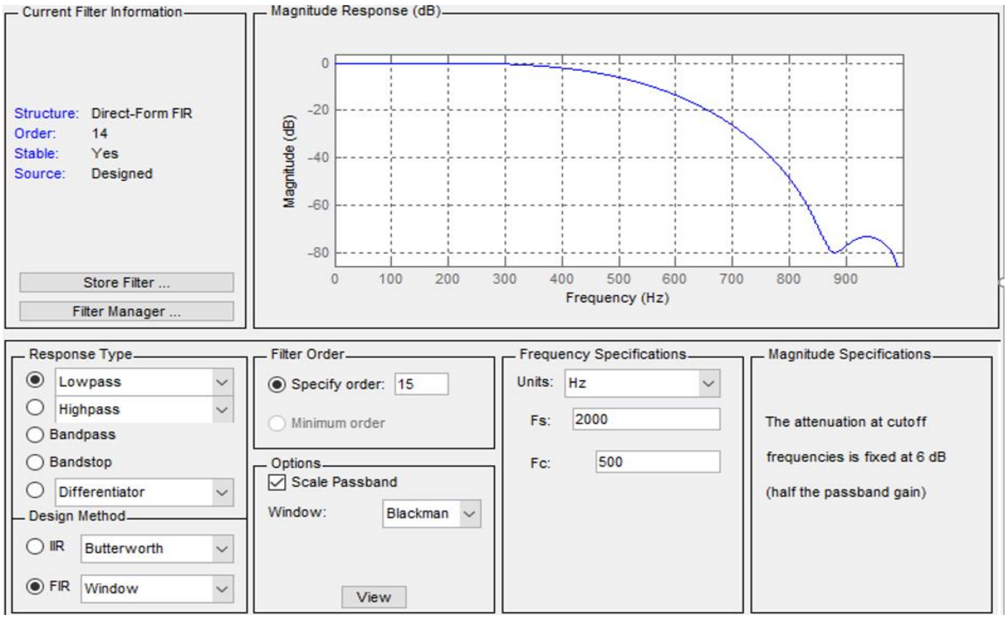
Table 1 Common window function basic parameters

Window function	Sidelobe peak/dB	main lobe width	Transition band width	stopband minimum attenuation/dB
triangularwindow	-13	4π/N	0.9	-21
rectangularwindow	-25	8π/N	2.1	-25
hanning window	-31	8π/N	3.1	-44
hamming window	-41	8π/N	3.3	-53
Blackman window	-57	12π/N	5.5	-74
Kaiser window	-57		5	-80

3 FIR filter FPGA implementation

3.1 VerilogHDL language design
There are two ideas for this design method: the first is to use the shift instead of the multiplier to achieve the multiply-add operation of the filter; the second is to use Verilog to instantiate the IP core of the associated multiplier and accumulator. The first implementation language is complex, but saves chip resources; the second is more common in engineering design, greatly simplifying the language and reducing the error rate. In this paper, a 15th-order (length 16) low-pass linear phase FIR filter is designed as an example: Blackman window function design, cutoff frequency is 500Hz, sampling frequency is 2000Hz; coefficient quantization bit is 12bit, input data bit width is 12bit The output data bit width is 26 bit system clock 2KHz.

Firstly, the filter is designed by the FDATool in the APPS (application) provided by MATLAB, and the parameters are set according to the design requirements, and the filter coefficients of the corresponding requirements can be obtained. FDATool (Filter Design & Analysis Tool) is a special filter design analysis tool in MATLAB signal processing toolbox. MATLAB6.0 and above also specifically added Filter Design Toolbox. FDATool can design conventional digital filters that meet almost all needs, including various design methods for FIR and IIR. Its operation is very simple and flexible. As shown in Figure 1, the number of filter taps, sampling frequency, passband frequency, and selection window function type are set according to design requirements, after the setting is completed, Design Filter is saved, and the coefficient output is saved to the workspace, and then the corresponding gain is added to round the output. Quantization coefficient is 0, -1, 6, 21, -52, -116, 261, 905, 905, 261, -116, -52, 21, 6, -1, 0.


Current Filter Information

Structure: Direct-Form FIR
Order: 14
Stable: Yes
Source: Designed

Response Type: [✓ Lowpass] Highpass Bandpass Bandstop Differentiator

Design Method: IIR Butterworth [✓ FIR Window]

Filter Order: [✓ Specify order: 15] Minimum order

Options: [✓ Scale Passband]
Window: Blackman

Frequency Specifications
Units: Hz
Fs: 2000
Fc: 500

Magnitude Specifications
The attenuation at cutoff frequencies is fixed at 6 dB (half the passband gain)

Figure 1. FDA.parameter setting

Then create a new project FIR_16 in Quartus II, select the matching hardware parameter EP4CE30F23C7; select New Design Verilog HDL file; instantiate 8 multipliers in the code (you need to compile LPM_MULT core first); manually input the above generated quantized coefficients into the multiplier in. Finally fully compiled. Check the syntax for errors and make sure the compilation is passed.

3.2 Using the FIR IP core design
According to the general PPGA design rules, for the universal functional modules implemented by the manual code, if the target device provides the corresponding IP core, the IP core is generally selected for design. The Quartus II also provides a common FIR filter IP core for most FPGA chips [8].

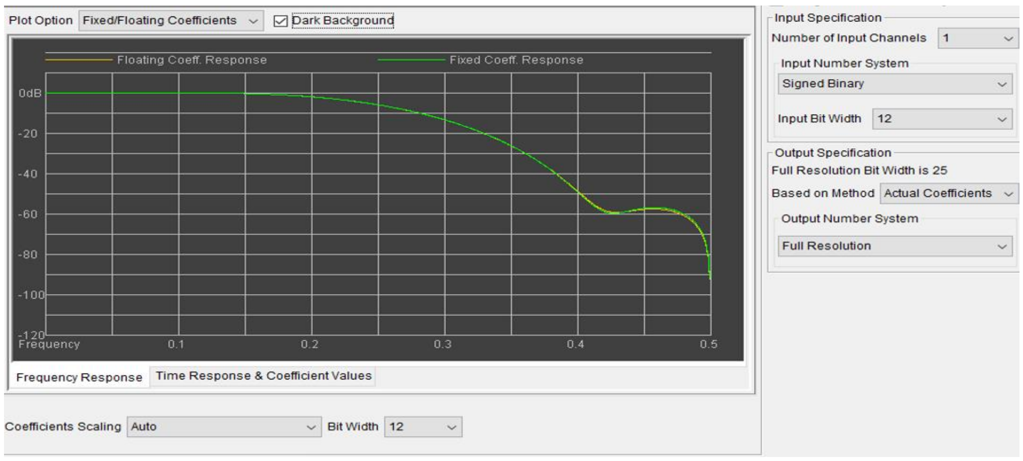
Therefore, in actual engineering practice, most of us will directly use the IP core provided by Altera to design the FIR filter.

Quartus Il 13.0 offers two very powerful FIR cores: FIR Compiler v13.0 and FIR Compiler II v13.0. The FIR Compiler v13.0 core data sheet details the IP core functionality and technical description. The following example of the same section above illustrates the FIR Compiler v13.0 core implementation of the distributed structure filtering algorithm.

As with Verilog's implementation of the filter, the first step still needs to generate quantized coefficients and save them as TXT format files. After opening the Quartus II software and selecting the target device EP4CE30F23C7, start the “Mega Wizard Plug-In Manager”.

Altera offers extremely rich library functions in the Mega Wizard Plug-In Manager tool, which are optimized for Altera's devices and have a simple circuit structure that reduces the designer's workload.

Select “DSP - Filters - FIRCompiler v13.0”, set the target device cluster, the language model of the output file (VerilogHDL/VHDL), determine the IP storage path and name, click “Next” to enter the IP core tool. In the interface, select “Step: Parameterize” to enter the parameter setting interface shown in Figure 2. Set the FIR core parameters. It should be noted that the bit width of the filter coefficient is set to 12 bits, the device family is CycloneIVE, and the pipeline level is (Pipline Level). The storage resources of the input data and filter coefficients are kept at default. Logic Cells. Click on the filter structure list box (Structure) to see that the IP core provides four different implementation structures: Distributed Arithmetic: Fully SerialFilter, Distributed Arithmetic: Multi-Bit Filter, Distributed Arithmetic: Fully Parall Fiter, Variable/Fixed Coefficient: Multi-Cycle. The chip resources required for different structures are different, and the operation speed is also different. For the Distributed Arithmetic: Fully Serial Filter structure option, since the filter coefficient is set to 12 bits wide (the wider the bit width, the higher the filter accuracy, the more resources are occupied); click “Edit Coeffcients Set” to configure the filter. The coefficient of the device, the IP core gives us a number of choices, we choose to automatically import the coefficients import the coefficient file of the TXT format saved in front of us.


Plot Option: Fixed/Floating Coefficients
☑ Dark Background

Input Specification
Number of Input Channels: 1
Input Number System: Signed Binary
Input Bit Width: 12

Output Specification
Full Resolution Bit Width is 25
Based on Method: Actual Coefficients
Output Number System: Full Resolution

Coefficients Scaling: Auto
Bit Width: 12

Figure 2. Filter parameter setting

3.3 Designing with DSP Builder

DSP Builder is a system-level design tool developed by Altera Corporation of the United States for DSP development. It integrates MATLAB and Simulink DSP development software in the Quartus II design environment. When co-simulation experiments on a computer, it is necessary to pay attention to the version matching problem. The matching of different versions of Quartus, DSP Builder and MATLAB can be queried on Altera's official website.

First, the filter coefficients are generated in the same way as before, and will not be described. Then create a new file, select Simulink Library, and create a new model, save it to the specified folder and name it *.mdl file.

Add Altera DSP Builder Blockset; there are two options in this step: the first option is very simple select the packaged IP core in the MegaCore Functions. As in the previous section, you can design the filter by directly configuring the IP core. The other is to build some algorithms using the integrated modules given in Arithmetic in the Altera DSP Builder Standard Blockset. The 15th-order filter uses four Multiply Add modules to divide the 16 taps (coefficients) into four groups and write them to the "Constant Values" of the four-input multiply-add module. There are also parameter settings such as the number of input bits. As shown in Figure 3.

After the core of the algorithm is packaged as Subsystem, then the peripheral circuit is built, installed as Subsystem, add counter(Increment/Decrement)and LUT (the lookup table is similar to ROM), and a Signal Compiler (linked with Quartus); double-click Signal Compiler to configure, Signaltap II Check Enable; The deeper the depth is, the more obvious the graphics will appear and the more resources will be occupied at the board level. Therefore, the depth(simulation depth) in this paper is selected as 256-2k.Finally click Export to output VHDL project.

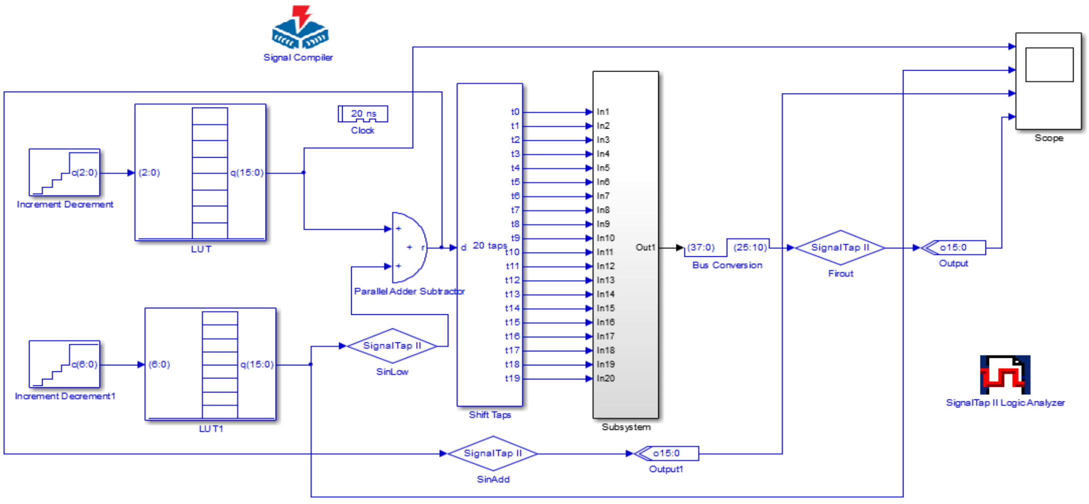


Figure 3. Overall block diagram of DSP Builder

4 Results Simulation & Conclusion

4.1 Results Simulation

Put the above three designs (take one) into the Quartus II, select Start Test Bench Template Writer in Start, generate the TestBench file, and write the testbench file. Add the testbench file to the simulation; finally click Tools, select Run Simulation Tool, select RTL Simulation, as shown in Figure 4, enter the simulation phase, and finally "dout_s" is displayed as the filtered waveform.

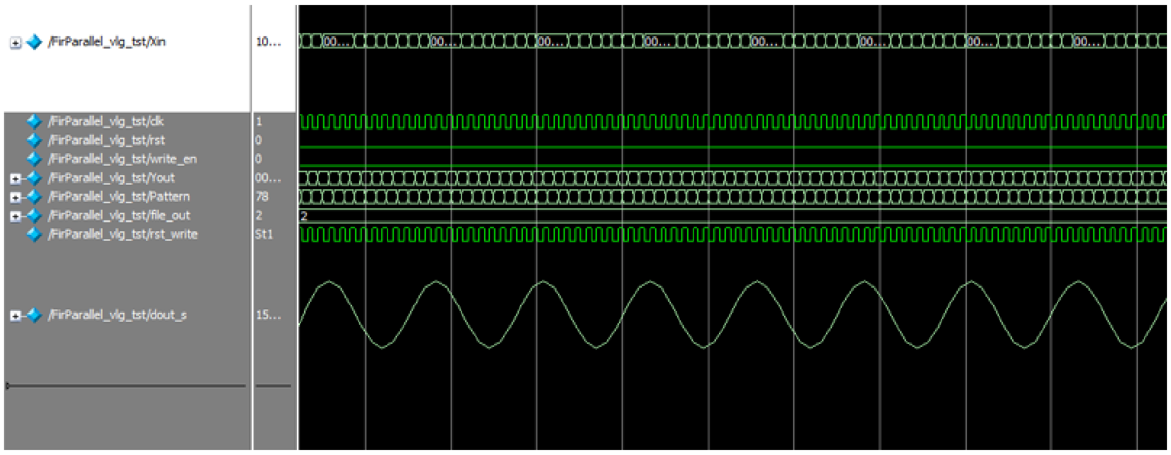

/FirParallel_vlg_tst/Xin 10...
/FirParallel_vlg_tst/clk 1
/FirParallel_vlg_tst/rst 0
/FirParallel_vlg_tst/write_en 0
/FirParallel_vlg_tst/Yout 00...
/FirParallel_vlg_tst/Pattern 78
/FirParallel_vlg_tst/file_out 2
/FirParallel_vlg_tst/rst_write St1
/FirParallel_vlg_tst/dout_s 15...

Figure 4. Simulation results of ModelSim

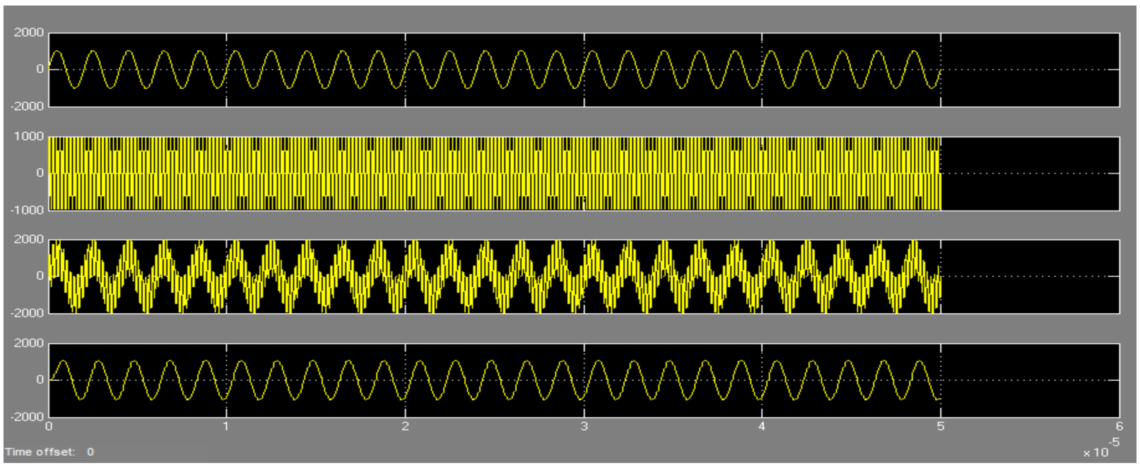


Figure 5. Simulation results of Scope

About the filtering algorithm built in DSP Builder, the simulation results can also be displayed in MATLAB's own Scope (virtual filter). As shown in Figure 5, the first window is the input sinusoidal signal; the second *window* is the superimposed high frequency signal; the third window is the accumulated mixed signal; the last window is the filtered signal.

4.2 Conclusion

Compared with DSP processing digital signals, FPGA has the advantage of saving time in parallel processing, which is not available in the latter. In the same price range, FPGA resources are much richer than DSP; in practical applications, there are many ways to implement signal processing algorithms using FPGA. Of course, in the case that the accuracy requirements are not so high, we still prefer FPGA companies to build algorithms for our encapsulated IP core and DSP Builder. The design process is simple and error-free, and further optimization of the design is needed in large projects.

References

[1] Wang Yingwei, Wang Zhenyu, Yan Wei, Shi Guangwei. FPGA Implementation and Optimization of Full Parallel FIR Filter[J]. Electronic Design Engineering, 2015, 23(22): 94-97.
[2] Yang Luxi. Modern Digital Signal Processing [M]. Beijing: Science Press, 2007.
[3] Xie Liying, Fang Limin.Design of FIR Digital Filter Based on MATLAB[J].Journal of Guangdong Second Normal University,2018,38(05):61-66.
[3] Chao Cheng, Parhi K K. Low-cost parallel FIR filterstructures with 2-stage parallelism [J]. Circuits and Systems I, 2007, 280:290.
[4] Liu Pengquan. Design and implementation of FIR digital filter based on FPGA [D]. Northwestern Polytechnical University, 2006.
[5] Chen Jiazhen, Zheng Zihua, Ye Feng, et al. Design of Digital Filter System Based on FPGA[J]. Computer Simulation, 2009, 26(12): 329-332.
[6] Dong Xianglei. Research and implementation of anti-jamming technology for navigation system based on multi-beam [D]. University of Electronic Science and Technology of China, 2014.
[7] Shan Wenjun, Zhou Xuechun, Li Wenhua.Design and Implementation of FIR Digital Filter Based on FPGA[J].Modern Electronic Technique,2013,36(14):123-126.
[8] Chen Yuanyuan, Liu Youyao. FPGA design and implementation of FIR filter[J]. Electronic Design Engineering, 2017, 25(24): 65-69+73.