The FPGA Implementation of Wavelet-Neural Network Based On DSP Builder

Bing Qian Liu[1, a], Mao Fa Gong[1, a*], Wen Shuo Wang[1, b], Yu Qing Lin[1, a]
Yi Sheng Xu[1, a], Fan Jiao Yin[1, a], Mei Rong Li[1, a]

[1]College of Electrical Engineering and Automation, Shandong University of Science and Technology, Qingdao 266590, China

[a]email: 237788152@qq.com, [b]email: 729398564@qq.com

Keywords:wavelet-neural network; DSP Builder; differential protection; FPGA

Abstract. According to the wavelet-neural network algorithm of relay protection device has the characteristics of requirement of high sampling frequency and large amount of data calculation, we use FPGA to realize the algorithm in hardware. A system model is built and converted into VHDL code through the system-level digital signal processing development tool named *DSP Builder* from Altera Company. We verified the correctness and feasibility of the method by the timing simulation diagram get from *Modelsim SE* software. This paper implemented protection method of the wavelet transform and neural network with hardware, and completed a series of research work in the field of transformer differential protection.

Introduction

The differential protection can effectively distinguish between transformer internal fault and external fault [1,2], But in identifying excitation inrush current and internal fault current is still flawed. Wavelet transform have the characteristics of good localization and multi-resolution. It can decompose signals on different scales [3,4], and accurately extract energy eigenvalues of fault current and excitation inrush current signal. Trained neural network can correctly identify fault current and excitation inrush current of transformers. The proposed algorithm can correctly distinguish fault current and excitation inrush current verified by PSCAD and *Matlab* simulation.

Neural network is a kind of parallel and distributed information processing system [5], so the hardware realization of neural network can give full play to the parallel ability of the network. FPGA has been widely used in hardware implementation of neural network because it has the parallel characteristic, and rich hardcore unit and storage resources [6].

The FPGA Implementation of Wavelet Transform

Since the 1990 s, the wavelet analysis algorithm is highly valued gradually by domestic and foreign scholars. They think that this method is the continuation and development of Fourier transform and can be considered to be a breakthrough of Fourier transformation. In dealing with practical problems of the signal, we usually use the high-pass filter and low-pass filter to realize the multiresolution analysis.

Mallat algorithm of signal decomposition are divided into three parts: initialization, iterations, and terminated. Wavelet transform hardware mainly include data processing module, the zero crossing detection module, filter module, down-sampling by two module, etc. The hardware block diagram of the design drawn by *DSP Builder* is shown in Figure 1.

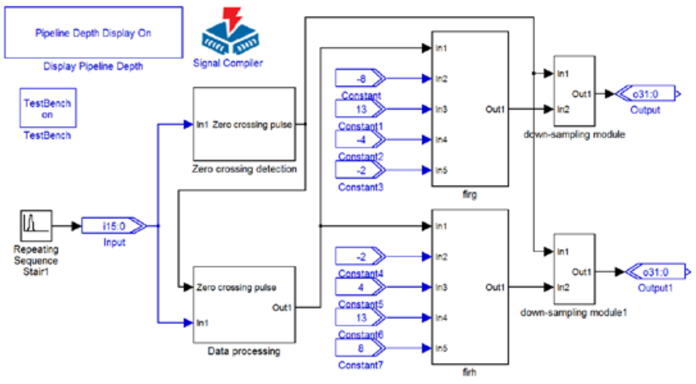

Fig.1 Wavelet transform hardware model

In this paper, FIR filter is used to realize the Mallat algorithm of wavelet transform. In order to facilitate processing, we amplify filter coefficient to 16 times and round which get from the DB2 wavelet. The high-pass filter coefficient is G = (2, 4, 13, 8), and low pass filter coefficient is H = (8, 13, 4, 2) in the FPGA design. The input data sampling from current of transformer is IN= （2, 13, 36, 1464, 6682, 9294, 8562, 4685, 95, 23, 7, 0）., Amplify the filter coefficients to 16 times, the wavelet coefficients will also increase to 16 times. By the *dwt(,)* command in *Matlab*, we get the result is cH= (-108, -10882, 6586, 23286, -33436, -185), cL= (107, -2713, 24593, 197033, 129160, 1067). According to the *Modelsim SE* software simulation, The FPGA hardware calculation results are shown in Figure 2, cH= (-112, -11300, 6586, 21864, -35533, -191), cL= (112, -2599, 27460, 198796, 132075, 1087). Hardware simulation result and *Matlab* calculation are basically identical. The generation of error is mainly due to the rounding of filter coefficient, and the final result is cumulative magnified after calculating.

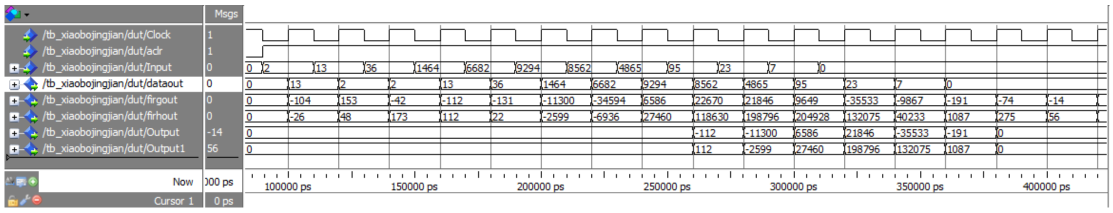
/tb_xiaobojingjian/dut/Clock
/tb_xiaobojingjian/dut/adr
/tb_xiaobojingjian/dut/Input
/tb_xiaobojingjian/dut/dataout
/tb_xiaobojingjian/dut/firgout
/tb_xiaobojingjian/dut/firhout
/tb_xiaobojingjian/dut/Output
/tb_xiaobojingjian/dut/Output1
Now
Cursor 1

Fig.2 Wavelet calculation timing simulation waveform

The FPGA Implementation of Neural Network

The Implementation of Incentive Function. The incentive function of neural network is hyperbolic tangent function and its Analytical expressions is shown in Eq. 1.

$$f(x)=\tanh(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{1}$$

Because the hyperbolic tangent function is symmetric about the origin center, only need to consider the approximate calculation of the part greater than zero. For the part of the input less than zero, taking negative of the corresponding number greater than zero is its output. The part is greater than zero is divided into $[0,L]$ and $[L,+\infty]$ two parts. In $[L,+\infty]$, $f(x)\approx 1$. In $[0,L]$, the number are divided into M equal parts. Then, in every segmented area get the best square approximation with a polynomial approximation, $f(x)\approx y=a_0+a_1x$. The block coefficient a_0 and a_1 are stored in the storage space in the FPGA chip.

By calling the components in the library of *Altera DSP Builder Blockset* to build the hyperbolic tangent function model is shown in Figure 3. This model mainly includes absolute value computation module, address translation module, ROM storage table module, Multiplication and addition operation module, and output module.

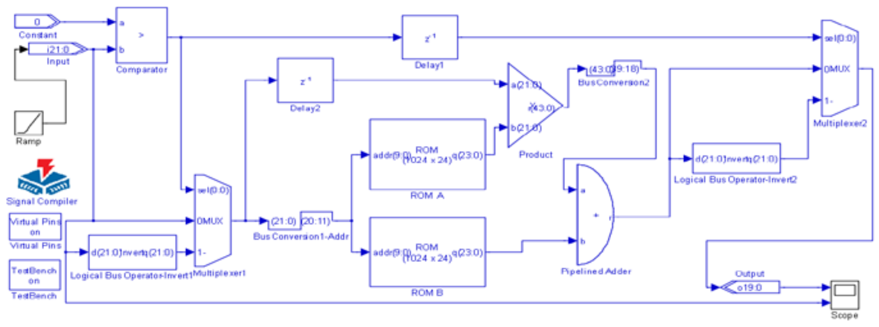
Fig.3 The hardware model of excitation function

Calling *Modelsim* software get the timing simulation waveform of the design, and the Waveform is shown in Figure 4. Waveform in the figure shows the output of input from -8 to 8. In order to facilitate the follow-up design of the neural network, the design is packaged into sub-module *Driving Function* by '*Create Subsystem*' command.

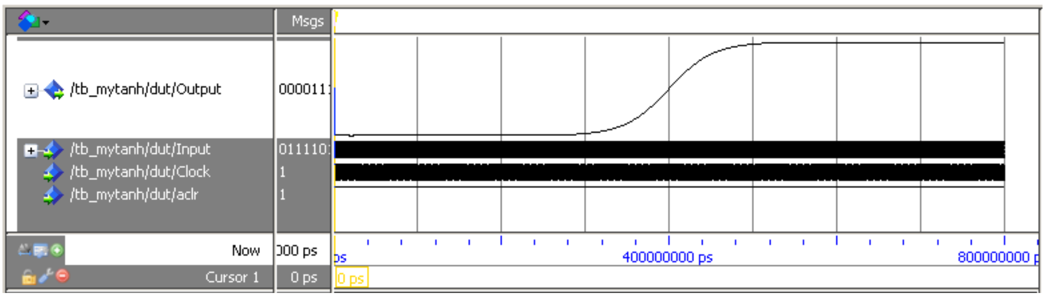
Fig.4 Excitation function timing simulation waveform

Design of Single Neuron. Operation of single neuron mainly include weighted multiplication, addition and nonlinear mapping. The weight values are stored in the single port RAM, and the RAM can read the corresponding weights according to the given address. Because the weight reading is a serial operation, the serial input data of neuron should transfer into parallel data. Considering RAM working frequency can reach hundreds MHz and the multiplication unit can invoke inline hardcore unit, the neuron can still work in high frequency after transforming from serial to parallel. The simplified neuron structure is shown in Figure 5.

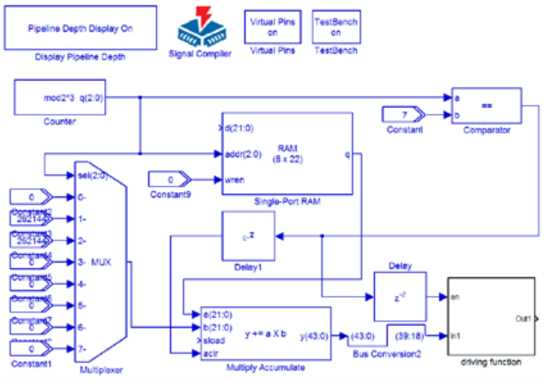
Fig.5 Neuron hardware model

The weight values are stored in RAM memory, and the control unit generate the corresponding address according to the input signal, then output the corresponding weights. The input is multiplied by weight then send the results into the accumulator. When all the inputs and the weights of neurons are multiplied, accumulator will output accumulated value, and the control unit generate reset signal to reset accumulator. Accumulated value through the output buffer, into the excitation function unit, then produce the corresponding output of neuron.

Because the neuron input datas and weights are floating point data, we amplify both to 2^18 times and round, in order to simplify the calculation of the FPGA. Assuming the 8 roads input datas of neuron are IN=（0, 1, 1, 0, 0, 0, 0, 0）, and the corresponding weights are Weight=（1, 0, 1, 2, 0, 0, 0, 1）, so the data before into excitation function is OUT= 0*1+1*0+1*1+0*2+ 0*0+0*0+0*0+ 0*1=1. The output of incentive function is tanh(1)=0.7616, in other words, the output value of the neuron should be 0.7616. *Modelsim* timing simulation result of the design is 199648, and the result divided by 2^18 is 0.7616, which is consistent with the theoretical calculation. Neuron timing simulation result is shown in Figure 6.

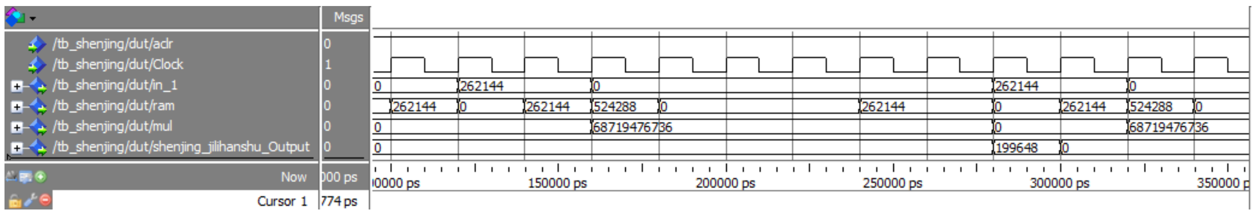

Fig.6 Neuron timing simulation waveform

Offline Learning of Neural Network. Neural network learning use offline learning method. After offline studying, the structure of the neural network and weights are fixed which can used to deal with specific problem. We designed a certain scale neural networks based on single neuron basic module to identify fault current and excitation inrush current of differential protection for transformer.

Summary

FPGA has a powerful hardware resources, which can realize both speed and accuracy requirement of the protection. We design and complete the wavelet transform and neural network on the FPGA hardware based on the wavelet-neural network algorithm. Comparing with the *Matlab* calculation results verify the design and result are correct and feasible. We completed a series of research work in the fields of transformer differential protection. It has theoretical significance and application value.

Acknowledgements

This study is funded by Natural Science Foundation of Shandong Province, China (ZR2012EEM021) and Graduate Scientific & Innovation Foundation of Shandong University of Science and Technology (YC140338).

References

[1] Dejun Shao. Research on Transient Mechanism and New Protection Principle of Large-scale Transformer [D]. Wuhan: Huazhong University of Science and Technology,2009.

[2] Baohui Zhang, Xianggen Yin. Power System Relaying Protection [M]. Beijing: China power press, 2009, 32-45.

[3] Lizhi Cheng, Hongxia Wang etc. Wavelet theory and its application [M]. Science Press, 2004.

[4] Changfa Xu, Guokuan Li. Practical wavelet analysis [M]. Wuhan: Huazhong University of Science and Technology Press, 2005, 77-92.

[5] Xiaofan Yang, Tinghuai Chen etc. The advantages and disadvantages of artificial neural network inherent [J]. Computer Science, 1994, 21 (2) : 23-26.

[6] Ming Yan. Neural Network Hardware Implementation Based on FPGA [D]. Qingdao: China ocean university, 2008.