2009 5th International Colloquium on Signal Processing & Its Applications (CSPA)

Implementing Gabor Filter for Fingerprint Recognition Using Verilog HDL

A. H. A. Razak and R. H. Taharim
Center for Electronics Engineering Studies (CEES), Faculty of Electrical Engineering, Universiti Teknologi MARA, 40450 Shah Alam, Selangor.

***Abstract* - This paper present the implementations of Gabor filter for fingerprint recognition using Verilog HDL. This work demonstrates the application of Gabor Filter technique to enhance the fingerprint image. The incoming signal in form of image pixel will be filter out or convolute by the Gabor filter to define the ridge and valley regions of fingerprint. This is done with the application of a real time convolve based on Field Programmable Gate Array (FPGA) to perform the convolution operation. The main characteristic of the proposed approach are the usage of memory to store the incoming image pixel and the coefficient of the Gabor filter before the convolution matrix take place. The result was the signal convoluted with the Gabor coefficient.**

Keywords – Gabor filter, digital filter, fingerprint, image processing, biometric, digital design, verilog.

I. INTRODUCTION

Fingerprint enhancement is a necessary process for fingerprint verification process. The most important measurement element in fingerprint recognition process is the texture of the fingerprint. The main reason is because the imperfect live-scan fingerprint-sensors with the current technology[1].

Fingerprint enhancement using Gabor filter is one of highly computational complexity in fingerprint verification process[1]. Gabor filter have a complex valued convolution kernel and a data format with complex values is used. So implementing Gabor filter is very significant in fingerprint verification process[2]. Designing Gabor filter will help enhancing the quality of fingerprint image. In fingerprint recognition, Gabor filter optimally capture both local orientation and frequency information from a fingerprint image. By tuning a Gabor filter to specific frequency and direction, the local frequency and orientation information can be obtained. Thus, they are suited for extracting texture information from images.

The main disadvantage of fingerprint recognition is the bad quality of its images when matched. An important task in image processing is the task of segmenting region of different texture in an image. Enhancing fingerprint images and texture segmentation has a great deal of importance and it is the focus of this project[3].

In fingerprint recognition process, there was few step or sequence that must be follow before the fingerprint image can be verified. But this project only focuses on developing the filter for fingerprint texture segmentation. In the digital signal processing the digital filter is the most important part to filter out the signal noise thus eliminating the unwanted image in fingerprint image. This project used Gabor type filter to segment the fingerprint texture. Texture segmentation is the most important in fingerprint recognition process. That was why this project only focus on designing the filter compares to other part. This filter enhances the image quality as its make ridges clearly differentiated from each other.

The objective of implementing Gabor Filter in fingerprint recognition was to segment the texture of fingerprint. Texture segmentation was the process of partitioning an image into regions based on their texture. The reason using Gabor filter was due to it characteristic. Gabor filters have the properties of spatial localization orientation selectivity and spatial-frequency selectivity. Depending on the parameter used, this filter made possible better characterize information about structure of directional angles and width of ridges and valleys in fingerprint images.

1.1 Gabor filter

A Gabor filter is linear filter whose impulse response is defined by a harmonic function multiplied by Gaussian function. The Fourier transform of a Gabor filter's impulse response is the convolution of Fourier transform of harmonic function and the Fourier function of Gaussian function. This is the formula of the complex Gabor function:

$$g(x, y) = s(x, y)\, w_r(x, y)$$

where s(x, y) is a complex sinusoidal, known as the carrier, and $w_r(x, y)$ is a 2-D Gaussian-shaped function, known as the envelop. So the general function of Gabor filter can be represent as below[4]:

$$G(x, y, \theta, f_0) = \exp\left\{-\frac{1}{2}\left(\frac{x_\theta^{\,2}}{\sigma_x^2} + \frac{y_\theta^{\,2}}{\sigma_y^2}\right)\right\} \cdot \cos(2\pi f_0 x_\theta), \quad (1)$$

$$\begin{bmatrix} x_\theta \\ y_\theta \end{bmatrix} = \begin{bmatrix} \sin\theta & \cos\theta \\ -\cos\theta & \sin\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \quad (2)$$

where θ is the ridge orientation respected to vertical axis, f_0 is the selected ridge frequency in x_θ – direction, σ_x and σ_y are the standard deviation of Gaussian function along the x_θ and y_θ axes respectively and the $[x_\theta, y_\theta]$ are the coordination of [x,y] after a clockwise rotation of the Cartesian axes by an angle of (90-θ).

Referring to the function in (1), this function can be decomposing into two orthogonal parts, one parallel and the other perpendicular to the orientation θ.

$$G(x,y,f_0)\big|_{\theta=90^\circ} = G_{BP}(x,f_0)G_{LP}(y)$$

$$G_{BP}(x,f_0) = \exp\left\{-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}\right)\right\}\cos(2\pi f_0 x)$$

$$G_{LP}(y) = \exp\left\{-\frac{1}{2}\left(\frac{y^2}{\sigma_y^2}\right)\right\} \quad (3)$$

where G_{BP} is only a band-pass Gaussian function of x and f_θ parameters while G_{LP} is only a low-pass Gaussian filter of y parameter[5].

Parameter f denotes the frequency of the wave in the sinusoidal plan. Since most local ridge structure of fingerprint come with well-defines local frequency and orientation, f can be set by the reciprocal of the average inter-ridge distance K, thus frequency is set as f=1/K. If using a too big f, it can create noise in the filtered image. If f too small, it can interlace ridges. The standard deviation of a two-dimensional normal(or Gaussian) distribution(or envelope) is represented by σ. It s related to the width of the Gaussian that modulates the Gabor filter. If σ is too big, the filter is more robust to noise, but it does not capture the ridge details. If σ is too small, the filter does not remove noise, but it capture the ridge details[6].

1.2 Gabor filter/Digital filter

The image convolution or real-time convolution using digital filter was already implemented but some used difference methodology. In technical paper title 'A memory based architecture for real-time convolution with variable kernels' by Vasily G. Moshnyaga, Kazuhiro Suzuli and Keikichi Tamoru, the main idea of architecture came from transforming the convolution flow in a way that keeps the pixel location stationary while shifting the intermediate results. The convolution kernel is swept over the image and at each step a pixel multiplies its value by all the kernels values and accumulates the product in its appropriate neighbors. Thus to perform the convolution within memory, it integrate the memory with processing elements which at each pixel point multiply the kernel value with the pixel, add product to the intermediate result and shift the computed sum to the neighboring element[7].

II. METHODOLOGY

The focus of this work was not to study on the characteristic of the Gabor filter. But it was to implement the characteristic of the Gabor filter into the digital filter. So since the studied has been done on the characteristic of Gabor filter, this work used the suggesting coefficient and implemented it to the filter design. The 5 variable of the filter was set up based on the previous work. Below were the table shows the value setup for the variable. The variable then was implemented into the Gabor equation as mention in the previous chapter. By applying the variable into the equation, the coefficient or the kernel of the Gabor filter can be determined. The kernel is shown below. For this work, 3x3 kernel was used.

TABEL 2.1
VARIABLE VALUE

Variable	Frequency ,f	Angle ,θ	Stand. Deviation ,σ_x	Stand. Deviation, σ_y
Value	1/3	45	0.5	0.5

TABLE 2.2
KERNEL VALUE

G(X,Y)	1	2	3
1	0.006737943	1.29E-05	-4E-08
2	2.35672E-07	4.14E-08	1.45E-12
3	-1.35859E-11	2.65E-14	8.53E-17

2.1 Gabor filter

In digital signal processing, the output of the signal was the convolution between the input signals with the filter coefficient. So mainly, the digital filter circuit was the circuit to convolute the input signal with the filter coefficient. In digital image processing, the image was presented in matrix form or in pixel. So basically the convolution involves the matrix convolution-convolution between image pixels with coefficient kernel.

Difference filter have difference method of filtering or sampling input signal. For this filter, it implemented a memory base architecture for real-time convolution with variable kernels[7]. Firstly the input data which was in pixel format will enter the filter and store it in the memory. The size of the memory was depending on the pixel size. If the pixel was 16x16 then the memory size would be 16x16 too. It means that every memory location will store for value for 1 image pixel.

After the image had been stored in the memory then it would start the convolution process. The control unit would call the data from determine memory location and sent it to the multiplication-accumulator (MAC). In MAC there was also a ROM which would permanently store the coefficient kernel value. The value of kernel would also be called by the control into the convolution circuit. When both data had entered the convolution circuit the process of multiplication and accumulation would take place. The result of MAC was the result of filtered image. This would also be the result of the filter. Figure 2.1 is the flow chart of how the filter works.

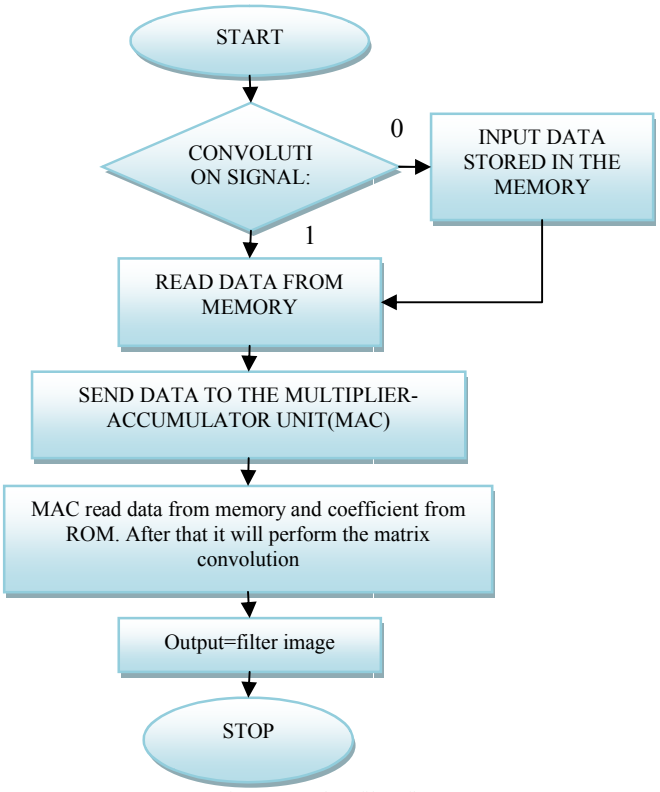

Fig. 2.1. Gabor filter flow

2.2 Filter design

After the coefficient and the method of filter design has been determine then the designing process started. This filter used field programmable gate array (FPGA). To design the logic gate of the filter, it used Verilog high device language (HDL). In designing process, this project used top-down method of design.

From the method of design defined above, the designing process started with defining the block diagram of the top level. Then the design went down to the lower level to get the full architecture of the filter. The design then was transform into the verilog code using Xilinx design tool. Figure below explain on the flow when the design has been transform into Verilog code.

2.3 Convolution between image pixel and coefficient kernel

In this chapter it discusses on the method of convolution between coefficient kernels with the image data in the memory. It discusses on the flow of which image data will be convolute with the kernels and the next data to be convolute. This methodology was very important as it will determine on how the control logic unit need to control the flow of the data and the arithmetic process. Figure 2.3 may better explain on the flow of the convolution between image pixel and coefficient kernels[8].

From the figure 2.3, it well explains on how the matrix convolution took place. Firstly the pixel D11 is convolute with the kernel. The value of D11 after convolution is D11=(D00xW11)+(D01xW12)+(D02xW13)+(D10xW21)+(D11xW)+(D12xW23)+(D20xW31)+(D21xW32)+(D22xW33).

The next pixel is D12=(D01xW11)+(D02xW12)+ (D03xW13)+(D11xW21)+(D12xW22)+(D13xW23)+(D21xW31)+(D22xW32)+(D23xW33). The sequence of next pixel to be convolute is show in figure 2.4[7].

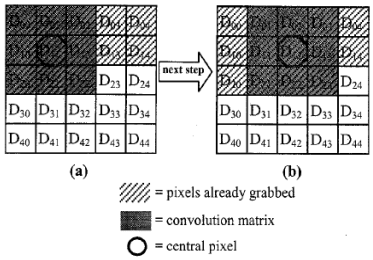

Fig. 2.3. method of convolution

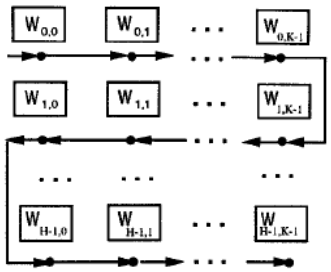

Fig. 2.4. sequence of memory reading

2.4 Design implementation/specification

The design started from the block diagram of the top level. The top level consists of 3 components: control logic unit(CLU), arithmetic logic unit(ALU) and memory unit(MEM). The top level should have signalled to indicate the operation of the filter whether to do the convolution or to receive image data. The data input was in 32-bit binary number. The reason 32-bit was used because the coefficient of the Gabor filter was in the floating-point. The floating-point number was represent based on IEEE-754 single precision standard.

The CLU was used to control the sequence of the convolution process. The CLU should have counters to generate the address location that would be supplied to the memory. Since the memory used was 16X16 so the counter should be generating 16 numbers. The third counter was to generate location for the coefficient in the ROM. Since the kernel used was 3X3 so there should be 9 locations. The counter should be able to counting until 9 times.

As mention the memory was 16X16. So there should have 256 locations. Each location should be able to store 32-bit binary number. The memory was RAM type. So it should have a enable signal to select the write or read operation.

The ALU was the main part of the filter where the convolution took place. In the ALU there was also a ROM to store the coefficient value. Besides the ALU also have a multiplier and an adder. The multiplier was to multiply the image data and coefficient. The adder was to add all the multiplied value. The filtered data was the total value of multiplication and additional.

III. Result and Discussion

After transforming the design into verilog language using Xilinx software and synthesize the code, the simulation will generate the schematic according to the code. After that the verification process took place. Below are the result of synthesize and verification of the design.

3.1 Filter (Top Level)

Basically there were 3 major parts in the filter: CLU, ALU and MEMORY. The other blocks were the signal and data selection. The 'convolution' signal was to indicate the operation of the filter. If the signal was high then the convolution process took place. If it went low then the filter receive image input and stored it to the memory based on the input location. The data entered the filter pixel by pixel. The 'PIXEL_X' and 'PIXEL_Y' signal gave the address of the memory location.

3.1.1 Control Logic Unit (CLU)

The control logic unit functioned to control the data flow in the filter. It gave instruction to the other block to do their job. Basically, it gave the memory address to read data to the MEMORY and give address of coefficient to the ALU. In the CLU there were 3 counters and a decoder as shown in figure 3.2.1.

This CLU would only generate the address location when the 'conv' signal was high. This signal indicated the convolution process was taken place but if the signal was low, it indicated that the writing image data into the memory took place.

This CLU have 3 difference clock signal. The first clock which connected to the Q_counter used normal clock,connected directly from the filter clock. The second filter,N-counter clock, have a long clock period compare to the normal clock. The high time of second clock was about 9 cycle of normal clock. It was made so that the second clok would only triger after the Q_counter finish it counting cyle. The third clock was also the same as the second clock. The third clock would only trigger after N_counter finish it counting cycle which was 16 cycle.

So if refered to the normal clock, N_counter triggered after 9 cylce clock and the M_counter triggered after 144 cycle clock. It took 2304 cycle clock to finish read all the memory data. Figure 3.2.2 showed the timing diagram for 1 counting cycle only.

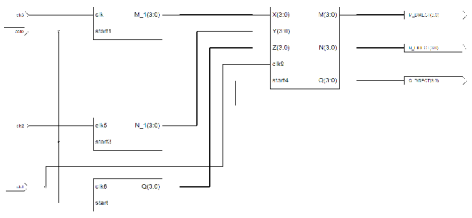

Fig. 3.2.1. block diagram of CLU

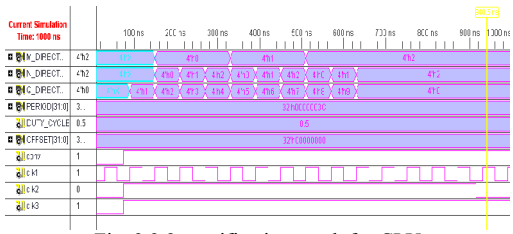

Fig. 3.2.2. verification result for CLU

3.2 Memory

The memory block was used to store the image pixel. The decoder was only to decode address for y direction only. The adress for x direction was supply directly from the CLU or from the filter input. The image input was also connected directly from the filter input. The writenable signal indicated whether the operation was write data or read data.

From the figure 3.3.2, first the 'writenable' signal was high to indicate the writing process was taking place. Then the signal went low to read the data in the memory. The moment the signal went low, it took about 1 clock cycle or 1 clock period for memory to read the address location and give the data to the output. So the memory would only give the output 1 clock period after the address location was entered.

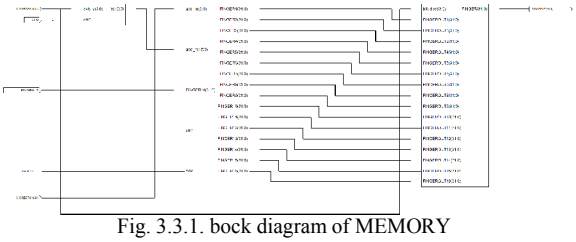

Fig. 3.3.1. bock diagram of MEMORY

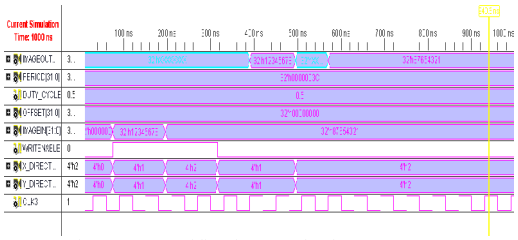

Fig. 3.3.2. verification results for MEMORY

3.3 Arithmetic Logic Unit(ALU)

This was the main part of the filter. This was the part that did the convolution process. This was where the Gabor coefficient was stored. It consists of 3 parts: ROM, DECODER and MAC. The ROM has 16 address locations but only 9 of it were used to store the coefficient. The decoder was used to determine which coefficient to be convolute with the image input. The MAC divided into 2 parts: multiplier and adder. The multiplier has 9 parallel multipliers. So the multiplication was done in the same time. This was to speed up the convolution process. The adder consists of 8 adder connected sequent. The adder was to sum up all the 9 multiplier output. Both multiplier and adder use Xilinx IP

cogen floating_point V3.0. This IP Cogen was generated from the Xilinx library.

Figure 3.4.2 shows the verification result for ALU. The total process of multiplication-accumulation was 127 clock cycle. Since this was a digital filter, there must be error. This was because in floating-point arithmetic operation, the system tends to round off the number. From the figure the output result was 0.006764772 but the expected result was 0.006764705. The difference was 0.00000068. It was so small. But in designing digital filter, this was the figure that the designer tried to reduce as small as possible. This was the measurement of the accuracy of the digital filter. In fingerprint recognition system, this figure will enhance the quality of the image of ridge and valley.

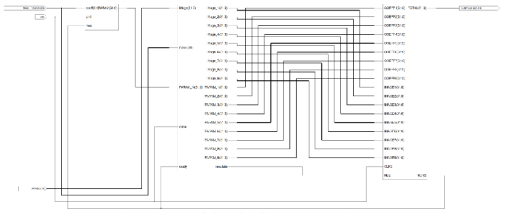

Fig. 3.4.1. block diagram of ALU

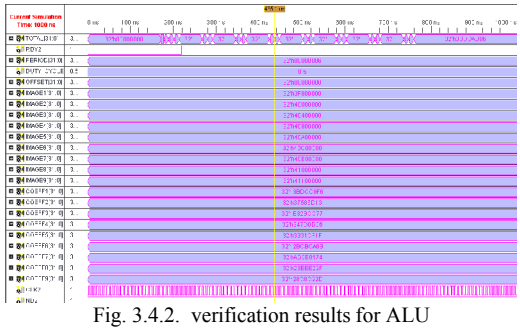

Fig. 3.4.2. verification results for ALU

IV. Conclusion

In conclusion, the filter was successfully design and verified. The filter functioned as expected. Even though the verification for the top level cannot be done because of the software limitation, but the filter can produce expected result based on the every part verification. The most important part was the ALU. Based on the verification on ALU, it produced expected result. Of course the multiplication and accumulation of the signal could not exactly the same as normal calculation because this was a digital filter. There must be some error. This was the error that the designer has been trying to reduce in digital design. For this filter, the error was so small. So the image of the filter should not much deter. Because of the software limitation, the timing constraint could not be set and the operation running time could not be determined.

In future, maybe this filter can be run using full version software. Other aspect is to change the coefficient. This filter is reconfigurable filter. The coefficient can be change to suit to the implementation. This filter uses the previous work on Gabor algorithm. If later someone come up with improve Gabor coefficient than the coefficient can simply stored into the ROM of the filter.

References

[1] U. W. Vutipon Areekul, Kittiwat Suppasriwasuseth,Saward Tantaratana, "Seperable Gabor Filter Realization for Fast Fingerprint Enhancement," 2005.

[2] P. H. W. L. Ocean Y. H. Cheung, Eric K.C. Tsang,Bertam E.SHi, "Implementing Of Gabor-type Filters on Field Programmable Gate Arrays," 2005.

[3] P. E. J. Khaled Hammouda, "Texture Segmentation Using Gabor Filters."

[4] L. L. Jianwei Yang, Tianzi Jiang,Yong Fan, "A modified Gabor filter design method for fingerprint image enhancement," 2003.

[5] Y. W. Lin Hong, Anil Jain, "Fingerprint Image Enhancement:Algorithm and Performance Evaluation," vol. 2, 1998.

[6] F. V. Sanderson L. Gonzaga de Oliveira, Aura Conci, "Enhancing Fingerprint image by an adaptive Gabor filter," 2006.

[7] K. S. Vasily G. Moshnyaga, Keikichi Tamaru, "A Memory based architecture for real-time convolution with variable kernels," 1998.

[8] A. P. Arrigo Benedetti, Nello Scarabottolo, "Image Convolution o FPGAs:the implementation of a multi-FPGA structure," 1998.

[9] Abdul Hadi Abdul Razak, Muhamad Iqbal Abu Zaharin and Nor Zaidi Haron, "Implementing Digital Finite Impulse Response Filter Using FPGA", Asia-Pacific Conference on Applied Electromagnetics (APACE2007), 2007.

[10] Abdul Hadi Abdul Razak, Nor Zaidi Haron and Mohd Faizul Md Idros, " Development of Logic Analyzer using Spartan 3E FPGA", 6th Student Conference on Research Development (SCOReD 2008), 2008.