

ADAPTABLE FINGERPRINT MINUTIAE EXTRACTION ALGORITHM BASED-ON CROSSING NUMBER METHOD FOR HARDWARE IMPLEMENTATION USING FPGA DEVICE

Sunny Arief Sudiro⁽¹⁾ and Rudi Trisno Yuwono⁽²⁾

⁽¹⁾STMIK Jakarta STI&K, Jl. BRI 17, Radio Dalam, Jakarta, Indonesia
sunny@jak-stik.ac.id

⁽²⁾Gunadarma University, Jl. Margonda Raya 100, Depok, Indonesia
rudity@staff.gunadarma.ac.id

ABSTRACT

In this article, a main perspective of developing and implementing fingerprint extraction and matching algorithms as a part of fingerprint recognition system is focused. First, developing a simple algorithm to extract fingerprint features and test this algorithm on PC. The second thing is implementing this algorithm into FPGA devices. The major research topics on which the proposed approach is developing and modifying fingerprint extraction feature algorithm. This development and modification are using crossing number method on pixel representation value '0'. In this new proposed algorithm, it is no need a process concerning ROI segmentation and no trigonometry calculation. And specially in obtaining their parameters using Angle Calculation Block avoiding floating points calculation. As this method is local feature that usually involve with 60-100 minutiae points, makes the template is small in size. Providing FAR, FRR and EER, performs the performance evaluation of proposed algorithm. The result is an adaptable fingerprint minutiae extraction algorithm into hardware implementation with 14.05 % of EER, better than reference algorithm, which is 20.39 %. The computational time is 18 seconds less than a similar method, which takes 60-90 seconds just for pre-processing step. The first step of algorithm implementation in hardware environment (embedded) using FPGA Device by developing IP Core without using any soft processor is presented.

KEYWORDS

Angle calculation block, extraction algorithm, fingerprint identification, minutiae points, matching algorithm, fingerprint verification, FPGA.

1. INTRODUCTION

Many systems require trusted mechanism for identification purpose in order to confirm or to identify person who requests for a specific service. This mechanism is used to ensure that provided a right person accesses service. One of the mechanisms is the biometric recognition system that used human biometric features to provide personal identification. In a publication by Wayman et.al, it was described that biometric technologies are automated methods of verifying or recognizing the identity of a living person based on a physiological or behavioral characteristics [1].

The demand of using biometric recognition system in many applications grows rapidly. This swift growing demand has triggered many researches in this area. A number of research has been published that offered novel methods in biometric recognition systems which lead to some improvements to the methods. A growing interest in fingerprint recognition algorithm was indicated by the increasing number of participants in fingerprint verification contest held in year 2000 (11 algorithms), 2002 (31 algorithms), 2004 (67 algorithms) and 2006 (70 algorithms) as described in [2], [3], [4], [5] and [6].

Typical biometric systems use pattern recognition that takes biometric data from specific individual and then extracts the features of these biometric data, which so called template, and comparing it with other features from the database as reference.

Fingerprint matching can be separated into two categories, which are verification and identification. It is the comparison of a claimant fingerprint against an enrolled fingerprint, where the intention is that the claimant fingerprint matches the enrolled fingerprint. In order to prepare for verification, a person initially enrolls his or her fingerprint into the verification system. A representation of that fingerprint is stored along with the person's name or other identity. The person identifying him or her, and then applying the fingerprint to the system such that the identity can be verified authenticates each access. Verification is also termed, one-to-one matching. On the other hand, identification is a fingerprint matching where fingerprint of unknown ownership is matched against a database of known fingerprints to associate with an identity. Identification is also termed, one-to-many matching. In other ways, the objective of identification is to search that owns the current biometric data, while the verification is to make sure that biometric data belongs to specific person. Jain et al. explain that using biometric feature is possible to confirm the existence of individual based on "who she/he is" and not on "what she/he possesses" or such as ID card, either on "what she/he remembers", such as password/PIN [7].

Fingerprint recognition systems use fingerprint structure of human being that differ one to another. This system involves the similarity level between two fingerprint images by comparing ridge or valley structure and spatial distribution of minutiae points or templates. For obtaining the templates, several processes are needed to capture fingerprint image using sensors, including pre processing such as, image enhancement, filtering, smoothing, binarization, thinning, and extracting features. The processes are then followed by false minutiae removal. For identification and verification purposes, process called comparison or matching is applied to both templates. Many researches of fingerprint recognition systems using personal computer have been conducted and proposed. Kasaei et al. proposed a method called Structural Feature Extraction Algorithm. This method uses crossing number for pixel representation value '1'. However, there was no explanation in how to obtain minutiae parameters, described in [8].

Salil Prabhakar uses filtering techniques to classify fingerprints. This method works on global features that usually have a bigger template size, as explained in [9].

Raimond Thai also uses crossing number method for pixel value representation '1' as presented in [10]. In his research, a post processing technique to remove false minutiae points was introduced. However, there were no clear explanations of how to obtain minutiae point parameters. Sharat Cikkerur uses chain code contour processing to detect minutiae points, which is translation invariant. It can be made rotation invariant if relative direction is used. This method also uses trigonometry calculation and works with floating point calculation as it was described in [11].

Tsai-Yang Jea utilizes crossing number to determine and to detect minutiae points. However, pixel pattern is used instead of mathematical calculation. In this method, a window of pixel pattern is scanned into fingerprint image and if the block of image is match with this pattern, a minutiae point is detected. It is still unclear of how to detect minutiae points parameter, as it was described in [12].

Josef Ström Bartunek introduces neural networks method to detect and to determine minutiae points from fingerprint image as published in [13]. The problem with this method is the need of training process every time the data is changed or grown. In this technique, 3 windows in different size are used for the training data. There is not enough explanation of how to obtain the direction of the minutiae points.

In order to reduce time calculations, many researches in hardware implementation of fingerprint recognition systems have been conducted and proposed. N.K. Ratha et al [14] and [15] proposed a sequential and parallel fingerprint-matching algorithm based on FPGA especially for Splash 2 architecture. The system consists of an array of Xilinx 4010 FPGAs connected to the host through an interface board. The Sun host realizes the query of database fingerprint and the matching process is executed on FPGA. Focusing on point pattern matching algorithm they obtained matching speed of the order of 105 matches per second for rolled fingerprint. In this system, matching process is performed without fingerprint feature extraction and overall performance of this design depends on communication between FPGA and Sun host through interface board.

Gwo-Cheng Chao, et al [16] proposed fingerprint verification on System on Chips (SoC). This system uses a Nios CPU, memory, sensor controller, gradient fields, Gab or Redundancy Circuit (GRC), thinning hardware, Avalon bus and Infineon Finger TIP sensor. Image acquisition, extracting feature and matching are implemented on this hardware system and this gives an acceleration of 10 times in comparison with a full software solution. This article explains that developing system in SOC environments is difficult for customizations and modification in the future.

In [17] Shenglin Yang et al described a hardware implementation of secure and memory-efficient fingerprint verification system with complete operation on XC2V1000 and 32Mbytes of RAM with LEON-2 soft processor. It consists in fingerprint capture, feature extraction, and matching can be executed in real time in less than 4 seconds on 50 MHz embedded platform. This paper proposed the use of a large size of template that is around 0.5 Kbytes. This approach can be optimized using a proper minutiae extraction algorithm. This design is also known as a ThumbPod (<http://www.thumbpod.com>), an embedded system for fingerprint recognition using FPGA device with several programming language in development: JAVA - KVM, C and VHDL.

In [18] the authors describe about implementation of MINDTCT fingerprint extraction feature algorithm and BOZORTH3 matching algorithm from NIST in the NFIS2 (NIST Fingerprint Image Software) using Spartran 3 FPGA device and Leon2 32-bit Sparc Processor (soft processor in FPGA device). The initial design is without FPU and H/W Co-Processor, with 75 % computation time is used by MINDTCT algorithm. By adding the FPU (the Leon2 soft processor is fixed-point) 94.14 % reduction time is obtained, and again by adding the H/W Co-Processor 97.89 % execution time reduction is estimated (at 40MHz/4KB cache).

Several research described above using fingerprint image from the host computer, so it is dependent system, to use this system the board must be connected to host. There is possibility using fingerprint sensor for capturing fingerprint image and send directly to embedded device, interfacing this sensor for fingerprint recognition embedded system is greatly described in [19] and [20].

In working with hardware implementation of fingerprint recognition system, a number of requirements must be fulfilled. These requirements are (1) real-time, (2) embedded, (3) small size, and (4) low power consumption. The algorithm must be optimized when it is implemented in FPGA device due to some limitations or constraints, such as memory capacity, components, power consumptions, cost, module dependencies and development difficulties.

In order to reduce difficulties during the development phase, a number of researches have been done using soft-core (soft processor). However, by using soft-core, more FPGA resources are needed as well as module dependency. Additionally, the use of translation algorithm in the soft-core increases the computational time. Another approach is the development of logical component/module for each process directly in the FPGA without using soft-core. It is more difficult, but the utilization of FPGA resources can be reduced. Moreover, since there are no dependencies with other module, the process becomes faster.

This research is focused on developing a simple and fast fingerprint feature extraction algorithm with small size of templates; this algorithm can be implemented on embedded environment using Field Programmable Gate Array (FPGA) devices.

This algorithm will be based on local structure called minutiae and crossing number method on pixel representation value '0' or valley structure. For testing and performance evaluation, three data sets, including database of fingerprint from Fingerprint Verification Competition 2004 as described in [5] and [4], will be used. The extraction algorithm will be developed and simulated using Matlab and C++ programming, as well as VHDL for the deployment into FPGA devices.

This research attempts to answer the following questions,

1. How to develop an algorithm to extract fingerprint feature in fingerprint recognition system that is adaptable to hardware implementation using FPGA device?
2. How to get minutiae points parameters in feature extraction process with simple calculations?

Therefore, the objectives of this research are, (1) to obtain simple (hardware-adaptable, low complexity) fingerprint feature extraction algorithm and (2) to implement the proposed algorithm into embedded environment using FPGA devices.

This article is decomposed as follows. In the second section present the description of biometric recognition specially based on fingerprint feature and minutiae extraction. In the section 3, the analysis and development of corresponding algorithm is presented. In section 4, result and discussion of performance evaluation and finally conclusion and suggestion are presented in the last section.

2. AN OVERVIEW OF BIOMETRIC RECOGNITION

In this section, main research domains on fingerprint as a biometric characteristic for recognition purposes are reviewed. According to Maltoni et al [21] biometric recognition refers to the use of distinctive physiological (such as: face, face thermo-gram, retina, fingerprint, DNA, ear, iris, palm thermo-gram, palm vein, palm geometry, palm print, voice) and behavioral characteristics (such as: gait and signature), called biometric identifiers or biometric characteristic (or simply biometrics) for automatically recognizing individuals. Sometimes all biometric identifiers are a combination of physiological and behavioral characteristics and they should not be exclusively classified into either physiological or behavioural characteristics.

Identification and verification are different problems with their own complexity. A template is usually taken in enrolment process and it is depending on the application. Sometimes, biometric recognition terminology is used for not concerning identification or verification process. Every physiological or behavioral characteristic of human being can be used as biometric feature as long as:

- Universality: each person has those characteristics.
- Distinctiveness: there is enough differences of the characteristics for two people.
- Permanence: these characteristics are not changing with time.
- Collectability: these characteristics can be easily collected.

Several considerations and conditions must be taking care when using biometric system, as follows:

- Performance: it is about accuracy and the speed of process and also a kind of device that must be used to obtain this level of accuracy and speed, also the environment of the system that influence the process.
- Acceptability: it is about society acceptance for biometric application in usual life.
- Circumvention: it is about the robustness of the system in presence of fake biometric characteristic.

2.1 Introduction to Fingerprint Recognition

Fingerprint recognition is the oldest method in biometric identification, at least since 2200 BC. People from Assyria, Babylonia, China and Japan used this method. Since 1897, dactyloscopy (manually fingerprint identification) is used in criminal investigation. Fingerprint pattern consists of ridges (lines across fingertip) and valleys (distance between ridges). This ridges-valley structure is unique for every human being (there is similarity with very small percentages). This system involves the similarity level between two fingerprint images by comparing ridge or valley structure and spatial distribution of minutiae points (templates), as presented in [22]. Figure 1 presents this fingerprint image structure.

In fingerprint recognition, matching process of fingerprint pattern does a comparison. In [23] there are two principal methods in fingerprint pattern matching:

- Minutiae matching (local structure): this method examines the existence of ridges bifurcations and ridges ending. There are about 100 minutiae points in fingertip. Depending on size of measurement there will be 30-60 minutiae points (dimension of fingerprint and sensor). Figure 2 shows the structure of minutiae points. This collection of minutiae points is used as templates and saved as a code of fingerprint pattern.
- Global pattern matching: this approach concerns the flow of fingerprint lines and can be formed as: arches, loops and whorls, also detection of singular points existence (core or delta points).

2.2 Fingerprint Templates Standardization

According to ANSI ISO/IEC FCD 19794-2 standard, a fingerprint template in local structure is a collection of minutiae points. Minutiae points are various anomalies in terms of ridge bifurcations, ridge endings, ridge crossovers, and small ridges. These points, for automatic feature extraction and matching are usually restricted to two types of minutiae: ridge terminations/endings (EP, End Points) and ridge bifurcations (BP, Bifurcations Points). Also, the

valley structure can be used for minutiae points, valley ending or valley bifurcations. A minutiae is identified by its position (x-y co ordinate) and the angle of dominant ridge with the x-axis at the point of interest. Type of minutiae (ridge/valley bifurcation or ridge/valley terminations) is also very important, and this information increases the accuracy of the fingerprint identification. Each minutiae point has a "type" associated with it. There are two major types of minutiae: a "ridge skeleton end point" and a "ridge skeleton bifurcation point" or split point. There are other types of "points of interest" in the friction ridges that occur much less frequent and are more difficult to define precisely. More complex types of minutiae are usually a combination of the basic types defined above. Some points are neither a ridge ending nor a bifurcation. Therefore, the following types are distinguished [24]:

- Ridge ending (also identifiable as a valley skeleton bifurcation point);
- Ridge bifurcation (also identifiable as valley skeleton end point);

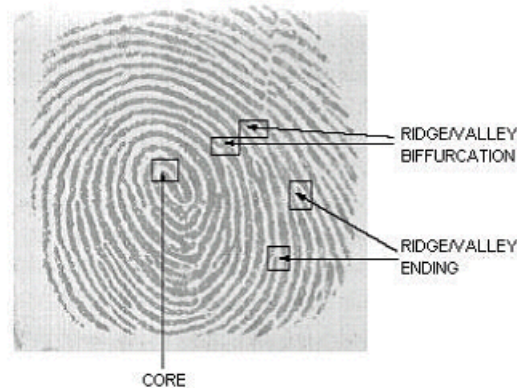


Figure1. Fingerprint Image Structure[22]

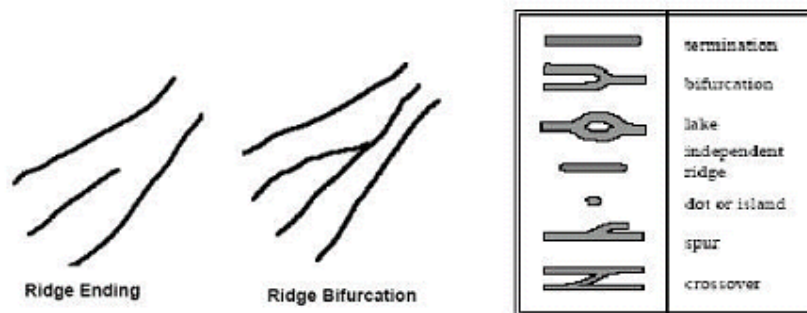


Figure2. Example of Minutiae Points.[23]

The minutiae angle is measured by increasing counter-clockwise starting from the horizontal axis to the right. There are three possibilities, such as [24]:

1. The minutiae point for a ridge ending is defined as the point of forking of the medial skeleton of the valley area immediately in front of the ridge ending. If the valley area were thinned

down to a single-pixel-wide skeleton, the point where the three legs intersect is the location of the minutia. In simpler terms, the point where the valley "Y"s, or (equivalently) where the three legs of the thinned valley area intersect. A ridge ending (encoded as valley skeleton bifurcation point) has three arms of valleys meeting in one point. Two valleys encompass an acute angle. The tangent to the third valley lying opposite of the enclosed ridge defines the direction of a valley bifurcation. The direction is again measured as the angle of the fingerprint line tangent with the horizontal axis in the right direction (see figure 3.a.).

2. A ridge bifurcation (encoded as valley skeleton end point) has three arms of ridges meeting in one point. Two ridges encompass an acute angle. The tangent to the third ridge lying opposite of the enclosed valley defines the direction of a ridge bifurcation. The direction is again measured as the angle of the fingerprint line tangent with the horizontal axis in the right direction (see figure 3.b.).
3. The direction of a ridge skeleton end point is defined as the angle of the ending ridge tangent with the horizontal axis in the right direction (see figure 3.c.). Ridge skeleton end points are only used in one type of the card formats, whereas in the other type ridge ending and ridge bifurcation is used as in the record format.

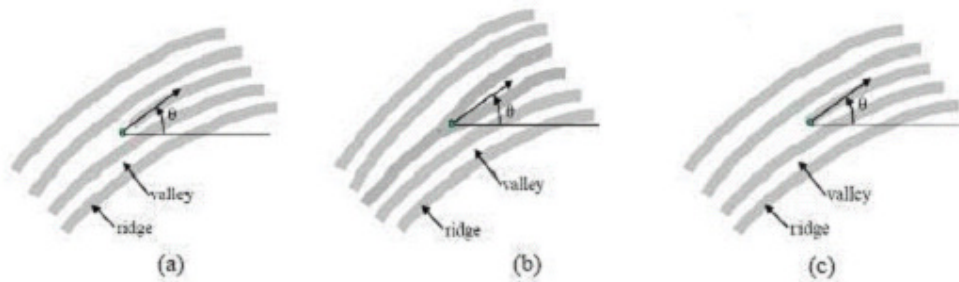


Figure3. Minutiae Points Parameters..[24]

2.3 Biometric Performance Evaluation

As described in [21], a typical biometric verification system commits two types of errors, such as: two different fingers considered as the same fingers (called false match) and two same fingers considered as two different fingers (called false non-match). These two types of errors are also often denoted as false acceptance and false rejection. A distinction has to be made between positive and negative recognition: in positive recognition systems (e.g., an access control system), a false match determines the false acceptance of an impostor, whereas a false nonmatch causes the false rejection of a genuine user. On the other hand, in a negative recognition application (e.g., preventing users from obtaining welfare benefits under false identities), a false match results in rejecting a genuine request, whereas a false non-match results in falsely accepting an impostor attempt. The notation "false match/false non-match" is not application dependent and therefore, in principle, is preferable to "false acceptance/false rejection." However, the use of false acceptance rate (FAR) and false rejection rate (FRR) is more popular and largely used in the commercial environment.

There are some error rates in fingerprint identification system that usually evaluate to see the performance of the system, such as [21]:

1. FAR, false acceptance rate or there is a mistake when accepting fingerprint template that should not belong to the same fingerprint.
2. FRR, false rejection rate or there is a mistake when rejecting fingerprint template that should belong to the same fingerprint.
3. EER, equal error rate or when FAR and FRR are at the same value.

Some others are: ZeroFMR, ZeroNFMR, Failure to Capture (FTC), Failure to Enrol (FTE), and Failure to Match (FTM). Of course quality of the image and threshold value setting are affecting these performances. A system designer may not know in advance the particular application for which the system may be used (or a single system may be designed for a wide variety of applications). Usually, they report system performances at all operating points (threshold, t). For this, they plot a Receiver Operating Characteristic (ROC) curve. A ROC curve is a plot of FMR against (1 - FNMR) for various decision thresholds (often FNMR is reported along the vertical axis instead of (1 - FNMR)). Some others are: ZeroFMR, ZeroNFMR, Failure to Capture (FTC), Failure to Enroll (FTE), and Failure to Match (FTM). Figure 4 show graphically illustrates the computation of FMR (FAR) and FNMR (FRR) over genuine and impostor distributions, it is evident that FMR is the percentage of impostor pairs whose matching score is greater than or equal to t , and FNMR is the percentage of genuine pairs whose matching score is less than t .

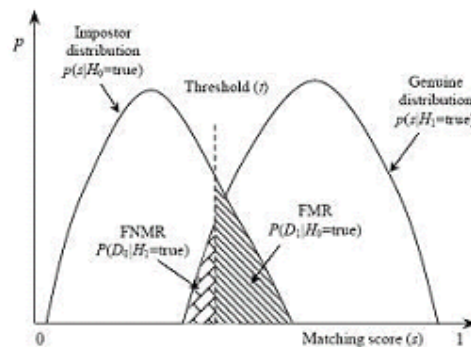


Figure4. FMR and FNMR for a given threshold t are displayed over the genuine and impostor score distributions t . [21]

Evaluation of fingerprint recognition algorithm is difficult because it needs homogeneous material for different users. As described in [1], many academic and industrial research groups have created new measurement techniques and new acquisition sensors for automatic fingerprint recognition in the past few years. Fingerprint-based biometric systems represent a significantly growing commercial segment for pattern recognition applications. Nevertheless, given the lack of standards, in the past most developers have generally performed only internal tests over self-collected databases. Few standardized benchmarks have been available for comparing developments in fingerprint verification. This deficiency has unavoidably led to the dissemination of confusing, incomparable and irreproducibility results, sometimes embedded in research papers and sometimes enriching the commercial claims of marketing brochures.

2.4 Structural Feature Extraction Algorithm as Fingerprint Feature Extraction on Local Structures

Kassei et al proposed this algorithm, described in [8]. This algorithm consists 5 steps, the first step is image enhancement using DRD filter (dominant ridge direction) on every 16x16 block.

The second step is segmentation process for separating foreground and background of the image to obtain ridge structure. The pre processing called thinning is applied to obtain one pixel representation for ridge structure. After this step, minutiae extraction process can be applied. The final process is the post-processing in order to remove false minutiae points.

Enhancement processes will produce the image as shown in figure 5. Summing the variation and orientation on the region does foreground separation; if the different value is higher than threshold (for example 0.35) then that region can be assumed as foreground.

The first step of ridge extraction consists in image binarization. Minutiae extraction is done after thinning process. Figure 6 shows ridge extracted and thinned image. Minutiae points is obtained using crossing number method (CN) on point P, by this formula 1:

$$CN = 0.5 \sum_{i=1}^8 |P_i - P_{i+1}| \text{ with } P_9 = P_1 \quad (1)$$

P ₄	P ₃	P ₂
P ₅	P	P ₁
P ₆	P ₇	P ₈

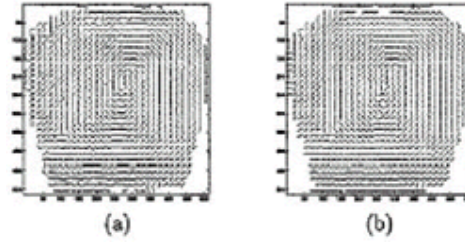


Figure5. (a) Block-direction image (each line representing dominant ridge direction at each block), (b) Smoothed block direction image.[8]

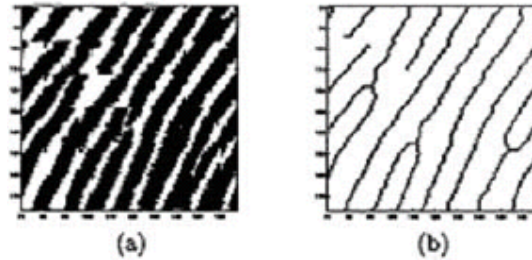


Figure6. (a) Original Image and (b) thinned image. [8]

P_i is a pixel element of the binary belonging to 3x3 P windows. If CN = 1, end point (EP) is obtained and if CN = 3, bifurcation point (BP) is obtained. Other values of CN are not applicable. This algorithm is fast enough because it works only for a specific foreground block.

Other methods or algorithms for extracting fingerprint feature based on local structures (minutiae points) are as follows: Adaptive Tracing the Gray Level Ridge of Fingerprint Image in [25], as described in [26] Fuzzy Logic and Neural Networks is proposed, as presented in [27] a method called Relationship of ridge and furrows examinations is proposed, Using Logical Templates [28], and in [29] described about Using Chaincoded Contour Representation of Fingerprint Images.

2.5 Hardware Implementation Overview (Embedded System)

Hardware implementation of the fingerprint recognition system is a method to obtain a system that responds to the following requirements: (1) real time, (2) portability, (3) embedded, (4) small size and (5) low power consumption. In order to reach these previous constraints, there are several choices in hardware implementation of the system, such as:

- Micro controller (MCU), is a computer-on-a-chip. It is a type of microprocessor emphasizing self-sufficiency and cost-effectiveness, in contrast to a general-purpose microprocessor (the kind used in a PC). A typical micro controller contains all the memory and interfaces needed for a simple application, whereas a general-purpose microprocessor requires additional chips to provide these functions. A micro controller is a single integrated circuit, commonly with the following features: central processing unit - ranging from small and simple 4-bit processors to sophisticated 32- or 64-bit processors, input/output interfaces such as serial ports, peripherals such as timers and watchdog circuits and signal conversion circuits, RAM for data storage, ROM, EPROM, EEPROM or Flash memory for program storage.
- DSP Processor is a specialized microprocessor designed specifically for digital signal processing, generally in real-time with the following main characteristics: designed for real-time processing, optimum performance with streaming data, separated program and data memories (Harvard architecture), special Instructions for SIMD (Single Instruction, Multiple Data) operations, the ability to act as a direct memory access device if in a host environment. Analog signals inputs can be converted in digital (using an internal Analog-to digital converter) and then processed. Output is then converted back to analog form using an internal Digital-to-analog converter (DAC)
- ASIC, An application-specific integrated circuit (ASIC) is an integrated circuit (IC) customized for a particular use, rather than intended for general-purpose use. For example, a chip designed solely to run a cell phone is an ASIC. Intermediate between ASICs and standard products are application specific standard products (ASSPs). The general term application specific integrated circuit includes FPGAs, but most designers use ASIC only for non field programmable devices (e.g. standard cell or sea of gates) and differ between ASIC and FPGAs
- FPGA, A field programmable gate array (FPGA) is a semiconductor device containing programmable logic components and programmable interconnects. The programmable logic components can be programmed to duplicate the functionality of basic logic gates such as AND, OR, XOR, NOT or more complex combinational functions such as decoders or simple math functions. In most FPGAs, these programmable logic components (or logic blocks, in FPGA parlance) also include memory elements, which may be simple flip-flops or more complete blocks of memories. A hierarchy of programmable interconnects allows the logic blocks of an FPGA to be interconnected as needed by the system designer, somewhat like a one-chip programmable breadboard. These logic blocks and interconnects can be programmed after the manufacturing process by the customer/designer (hence the term "field programmable", i.e. programmable in the field) so that the FPGA can perform whatever logical function is needed. FPGAs are generally slower than their application-specific

integrated circuit (ASIC) counterparts, can't handle as complex a design, and draw more power. However, they have several advantages such as a shorter time to market, ability to re-program in the field to fix bugs, and lower non-recurring engineering costs.

Some reasons using FPGAs as a good choice for implementing digital systems, particularly for fingerprint recognition application are:

- Availability of large logic capacity, exceeding several million equivalent logic gates, and includes dedicated memory resources.
- Include special hardware circuitry that is often needed in digital systems, such as digital signal processing (DSP) blocks (with multiply and accumulate functionality) and phase-locked loops (PLLs) (or delay-locked loops (DLLs)) that support complex clocking schemes.
- Support a wide range of interconnection standards, such as double data rate (DDR) SRAM memory, PCI and high-speed serial protocols.
- Soft and hard processors capability.
- Users can, easily program user-programmability.
- Design Partitioning, the system can be divided into several partitions into FPGA devices.

2.5.1 ZestSC1 FPGA board

One of known FPGA boards on the market is ZestSC1 FPGA board; block diagram of this board is shown in figure 7. As described in [30] this board uses a FPGA Xilinx Spartan-3 with up to 1 million gates. There are also up to 8MBytes of ZBT SRAM (Zero Bus Turnaround) memories with no turnaround cycles required when switching between read and write cycles. There are 49 header pins for user I/O connections (in order to connect other devices such as fingerprint sensor and LCD for displaying the message of the application). This board can be interfaced to a host PC through High Speed USB (480Mbits/s).

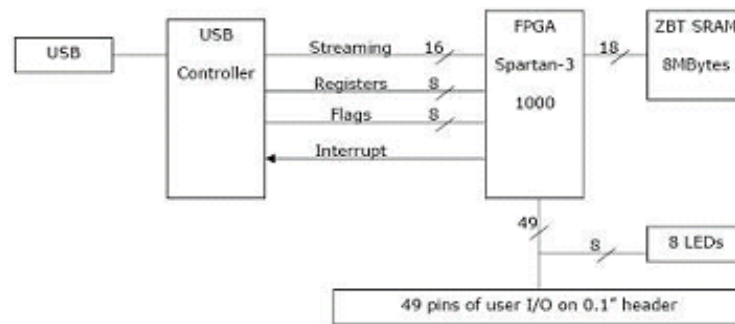


Figure7. Diagram Block of ZestSC1 FPGA Board.[30]

2.5.2 Fingerprint Sensor

There are many kinds of technologies used in the acquisition of live-scan digital fingerprint image using an electronic fingerprint scanner, without ink, as required in off-line sensing. As described in [31], these kinds of sensors are: 1. Optical Technology that consists of FTIR or known as Frustrated Total Internal Reflection, FTIR sweep, Sheet Prism, Fiber Optic and Electro-optical. 2. Solid-State which consists of Capacitive, Capacitive Sweep, Thermal Sweep, Electric Field and Piezo electric. For each of the above technologies, there are several interfacing methods in connection with the computer, such as USB, Serial and MCU connections. For example, USB

fingerprint reader called TouchChip using PerfectPrint software described in [32], takes at least 250 mS to obtain the fingerprint image.

As describe in [33], fingerprint sensor in this system (MBF200) consists of sensor arrays with 256 columns and 300 rows of sensor plates, resulting 76800 pixels in grey level value (cells). The block diagram of this sensor is illustrated in Figure 8.

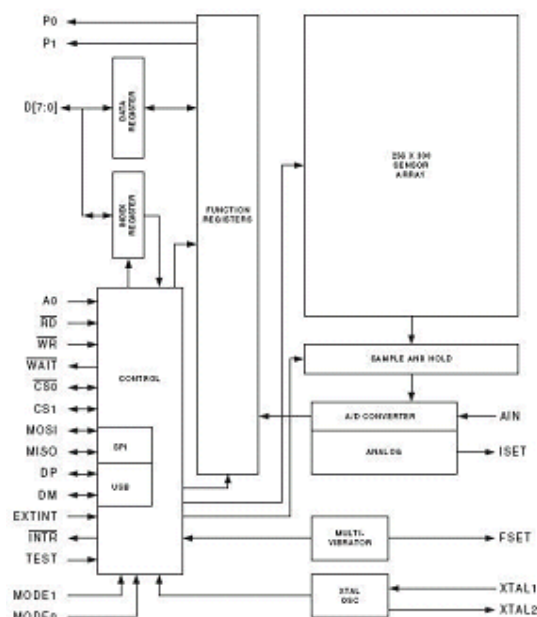


Figure8. Block Diagram of Fingerprint Sensor Reader MBF200.[33]

Associated with each column are two sample-and-hold circuits. A fingerprint image is sensed or captured one row at a time. This "row capture" occurs in two phases. In the first phase, the sensor plates of the selected row are pre-charged to the VDD voltage. During this pre-charge period, an internal signal enables the first set of sample-and-hold circuits to store the pre-charged plate voltages of the row. In the second phase, the row of sensor plates is discharged with a current source. The rate at which a cell is discharged is proportional to the "discharge current." After a period of time (referred to as the "discharge time"), an internal signal enables the second set of sample-and-hold circuits to store the final plate voltages.

3. RESEARCH METHOD

This research started with study literature to find out the current status in fingerprint extraction method, many experiments are carried out using Matlab programming in personal computer. Some testing is done based on FVC 2004 data set, SfinGe data set [21], and private data set collected using USB fingerprint sensor reader. In performance evaluation, other fingerprint minutiae points' extraction called mindtct and matching algorithm called b oorth3 from NFIS is used as reference. Finally, optimization, finalization and conversion for the proposed algorithm are performed in hardware implementation using FPGA devices.

In developing algorithm their some considerations must be taken, such as: properties and constraints. In software base, these properties are: performance (accuracy and speed), complexity,

size of code, size of templates, difficulty of development, dependency, and in hardware base these properties are: performance, size of block/modules, size of templates. And in implementing software algorithm into hardware base some constraints must be taking care, such as: memory, component/block device, module dependency, difficulty of development, interfacing and handshaking, licensing, etc.

Image acquisition and analysis from the fingerprint sensor determine the pixel width of the structure and the representation pixel value of the structure of fingerprint image as well as the image background. This characteristic must be considered when applying specific algorithm in fingerprint recognition system to have a good performance. There is a fact that ridge structure is thicker than valley structure. Average ridge width (typically 6 pixels) is thicker than average valley width (typically 4 pixels) [34]. Minutiae extraction is obtained from binary images of fingerprint. This thinner binary image will improve the performance, because it is easier for skeleton computation and it increases the speed of the process.

Image acquisition and analysis from the fingerprint sensor determine the pixel width of the structure and the representation pixel value of the structure of fingerprint image as well as the image background. This characteristic must be considered when applying specific algorithm in fingerprint recognition system to have a good performance. Fingerprint image representation depends on fingerprint sensor and program driver to construct the image. Ridgelines are represented by pixel value '1' and valley can be constructed with pixel value '0'. The appearance of the image on the screen depends on the software used to load this image. Some software shows pixel value '1' as black and others as white. Figure 9 shows an image of the fingerprint structure used in this paper. Black line represents pixel value '0' which corresponds to valley structure and white lines represent pixel value '1' which corresponds to ridge structure.

3.1 Fingerprint Minutiae Extraction Algorithm

This algorithm is developed based on crossing number method and consist several processes, such as: image enhancement process, binarization, thinning process, fingerprint feature extraction and inside this algorithm there is specific process to obtain minutiae points parameters called angle calculation block. More detail, the fingerprint minutiae extraction algorithm schema can be seen on figure 10.

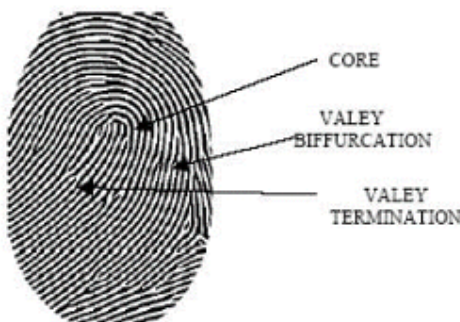


Figure9. Structure of Fingerprint Image

Image Enhancement: this process is to enhance fingerprint image and to obtain good quality of fingerprint image from poor fingerprint image. By processing good quality of fingerprint image,

will produce a good result in detecting minutiae points as templates. **Binarization** : this process is to convert gray-scale fingerprint image representation in to binary image.

Thinning: this process is to obtain one pixel width representation of fingerprint structure. Proposed algorithm is applied on pixel value representation '0' adapted to minutiae points detection algorithm and avoiding the ROI process.

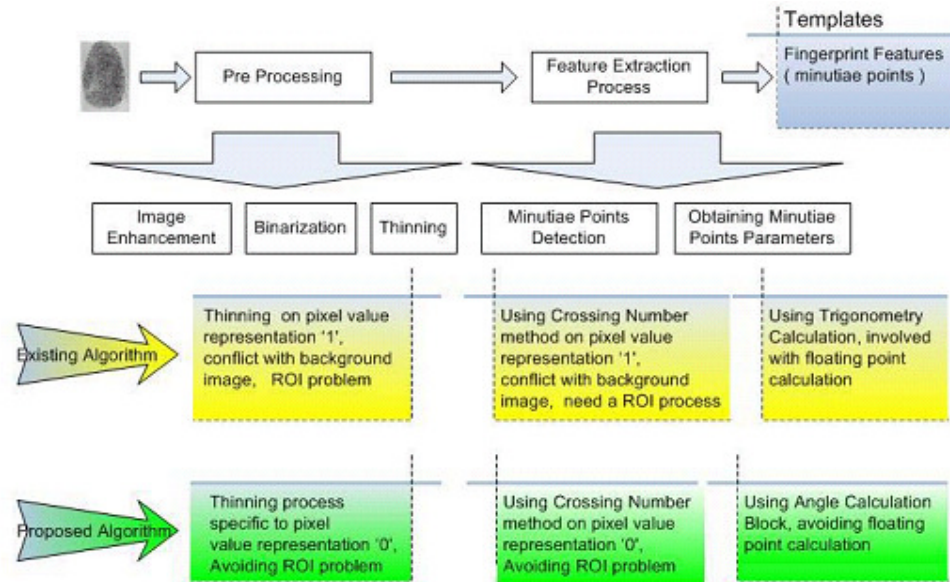


Figure10. Schema of fingerprint minutiae extraction algorithm.

Minutiae Points Detection: this process will detect the minutiae points existence and define type of minutiae based on Crossing Number method described in 2.4. This algorithm is concern on pixel value representation '0', in consequences, ROI process is not necessary.

Obtaining Minutiae Points Parameters: this process calculate the direction of minutiae points defined in using Angle Calculation Block. This method use simple calculation and avoiding floating point calculation.

3.1.1 Fingerprint Image Enhancement

The method described in [35] and [36] has been modified with some adjustments and modification to be well adapted to minutiae extraction algorithm (described in [37] and [38]) and to fingerprint image representation. This approach use Gab or filters as bandpass filters to remove the noise and preserve true ridge/valley structures. The adjustment and modification focus on binarization and thinning process following the method in [39] so this enhancement process can be adapted to extraction process based on crossing number on valley structure. Figure 11 shows the enhancement process results. This enhancement process will improve quality of fingerprint image with pixel representation value -80 to 80 instead of 0-255 (gray-scale of image). This image representation is not suitable for minutiae point detection algorithm using crossing number on valley structure. Adjusting the threshold value of binarization process used in minutiae point detection algorithm solves this problem.

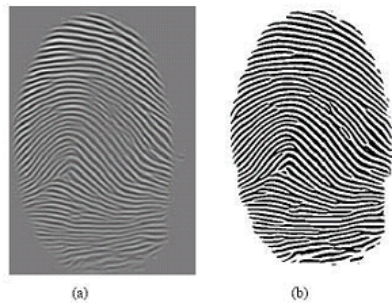


Figure 11. Binarization process result in image enhancement algorithm, (a) enhanced image and (b) binarized image.

Because of the pixel representation median value is 0, this value is used as threshold value in binarization process. Finally the simple algorithm of this process is:

- Get the size of wide and height of the image
- Check for all pixel value representation of the image If the value is greater than or equal to '0', change the value to '1', otherwise change to '0'.

3.1.2 Thinning Process

There is also binarization process in enhancement, so the next step is thinning. Algorithm Zhang and Suen's described in [40] and [41] is modified, this modification is presented in [39] so it is suitable for the image with pixel value representation of '0' as a concern. Based on evaluating pixel value and the 8-neighborhood pixel, with pixel reference P1 as shown in figure 12. And variable A is the number of transition from '0' to '1' clockwise from P2 ... P9 with representation of image is '0' for dark (black) and '1' for light (white) or region point is for pixel value '1' and background point is '0' [40] see figure 12.

The original algorithm is modified and the process will be based on representation of image with '1' for light (white) and '0' for dark (black) or region point is for pixel value '0' and background point is '1'. Actually this process is the same with inverting the image and use the first algorithm to do thinning, of course inverting the image means more computational time.

3.1.3 Minutiae Extraction Algorithm using Crossing Number Method

After image pre-processing step, minutiae extraction process is applied. Proposed algorithm can be seen on figure 13. Minutiae point detection depends on pixel value ('0' or '1'). Two methods are possible: first method processes only pixel with '1' value and second method is dedicated for pixel with '0' value. Method 1 count Crossing Number value on pixel value '1' or $P=1$, and method 2 do this process on pixel value '0' or $P=0$. Preprocessing: binarization and thinning algorithm on pixel representation of '0' will precede the minutiae point detection process (on '1' pixel value or on '0' pixel value). The Crossing Number calculation is based on formula 1 described in 2.4. This proposed method in more detail is presented in [37].

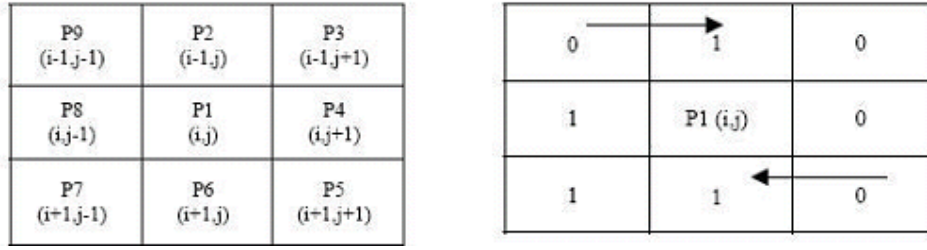


Figure12. 8 pixel neighborhood and Pixel Transition

3.1.4 Obtaining Minutia Points Parameters using Angle Calculation Block

Obtaining parameter of position and type of minutiae is not difficult; it is depend on position of pixel P and the CN value. If CN=1 it corresponds to the End Point (EP) and if CN=3, it corresponds to Bifurcation Point (BP) of minutiae. Proposed algorithm can be seen on figure 13. To reduce false minutiae detected at the edge of the fingerprint image, process to check candidate minutiae point whether at the edge of image or not are applied. Checking the existence of pixel value '0' at the right, left, top or bottom of candidate minutiae points for specific distance does this process. This proposed method in more detail is presented in [38].

To obtain angle of minutiae point, Angle Calculation Block for each minutiae type is used. This block is 7x7 so the scale of angle is $360/24 = 15^\circ$, the evaluation of tree line that exist at the edge of block and calculate the smallest distance between two line. The middle of this two line is the direction or angle of BP minutiae points, more detail can be seen of figure 14.a. For example in figure 14.a., line1=1, line 2= 7 and line 3 =16 are obtained, the smallest distance between two line is line 1 and line 2 that is 1 and 7 or the distance is 6. By this example, the middle between smallest distances of two lines is line position=4, and using equation (2), angle of minutiae point is 45° . For EP the process is finding position of line existence and calculates the angle using equation (2), for the example in figure 14.b. is 225° . Steps to find line position for angle of BP minutiae are:

1. Find each three line of bifurcation leg.
2. Find distance between the lines.
3. For smallest distance, line position is position of the first line added by half of the smallest distance.
4. Count the angle using equation (2).

$$\text{Angle} = (\text{lineposition} - 1) * 15^\circ \quad (2)$$

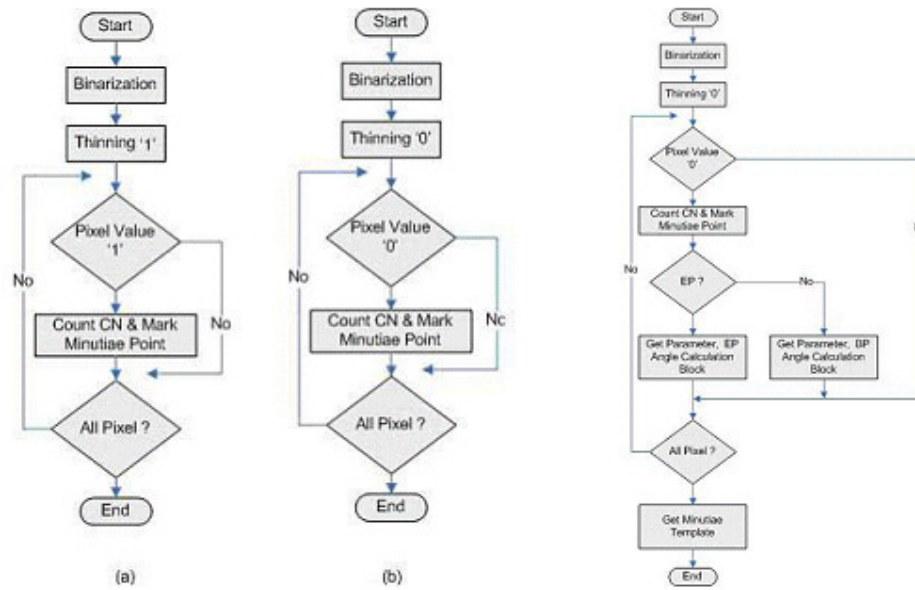


Figure13. Proposed algorithm, (a) method 1 and (b) method 2 and proposed algorithm in obtaining minutiae points parameters

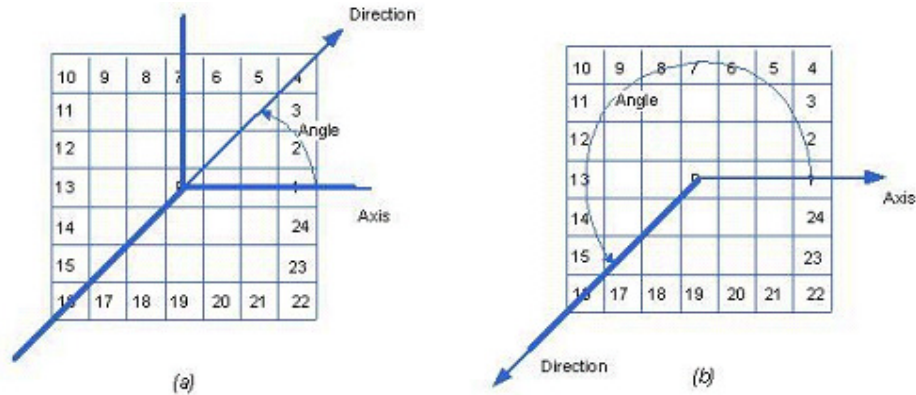


Figure14. Angle calculation block for calculating the angle of minutiae points direction.

3.2 Hardware Implementation

3.2.1 FPGA Board Interfacing

The system consists of a fingerprint sensor MBF200 from Fujitsu. This sensor is connected to the FPGA board, which is connected to PC via USB. The general system architecture can be seen on figure 15. The fingerprint sensor connected to I/O pins of the FPGA board is used to collect fingerprint image from the user. The algorithm of feature extraction and matching is implemented onto the FPGA. The template of fingerprint, after extraction, can be sent to the PC via USB connection in learning process. For verification process, this template is recalled from the PC and compared with new template from fingerprint image.

3.2.2 Fingerprint Sensor Interfacing Module

Fingerprint sensor reading operation needs signal that must be supplied sequentially to the sensor. So, a module is needed in the FPGA that able to generate those signals, or sequential system preferably synchronous sequential system that is controlled by clock is needed. There are two common models of synchronous sequential systems: Moore machine and Mealy machine, like described in [42]. A single clock controls both systems. The next state is determined by some (combinational) function of the inputs and present state. The difference between the two models is that in Moore machine the outputs are solely a function of present state, while in the Mealy machine the outputs are a function of the present state and of the inputs. Both systems are commonly referred to as state machines, which have an internal state that changes in time. As presented in figure 16, an entity is developed, which consists of state machines to generate signal controller for the fingerprint sensor. During the data reading from the sensor, signal WAIT to see if the next data is ready (normally it takes 28 clock cycles for first data and 6 clock cycles for every next data) must be checked, so the Mealy state machine is developed.

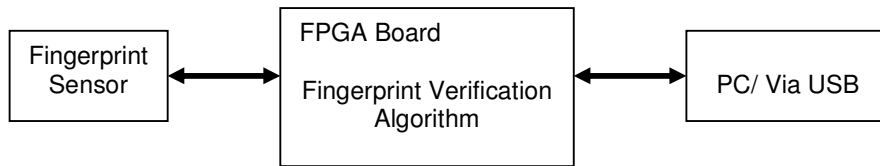


Figure15. Fingerprint Recognition System on FPGA.

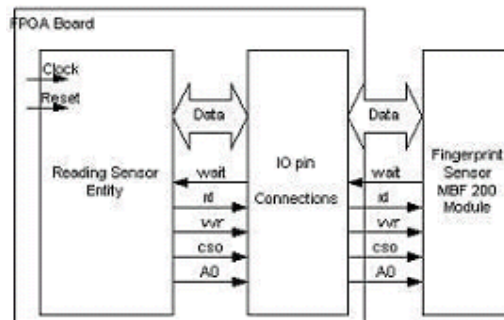


Figure16. Fingerprint Sensor Interface.

As presented in [43], state machine consisting 12 states is developed, which are :

1. State 0 is the initial state.
2. State 1, send signal to start and enable the sensor.
3. State 2 and 3, select index register CTRL B by sending '00001001' data.
4. State 4 and 5, set register CTRL B with '00000101' and then waits for 30 uS.
5. State 6 and 7, select index register CTRL A by sending '00001000' data.
6. State 8 and 9, set register CTRL A with '00000010' and waits for 28 clocks cycles.
7. State 10 and 11, reading pixel elements data if it is ready, this state is repeated until all data have been read with 6 clocks cycles delay for every next data.
8. State 12 is the closing state.

3.2.3 Fingerprint Pre Processing Module

For pre-processing process corresponding to binarization simple 8 bits comparator gates is used. With these operators, pixel value from grayscale level is coded to binary level 0 or 1 depending on specific used threshold value. VHDL version of thinning algorithm is based on thinning algorithm adapted to pixel value as described in [41] [39]. Figure 17 shows the binarization hardware implementation.

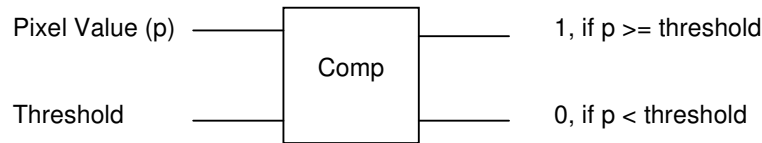


Figure17. Hardware implementation of binarization process.

In high level language accessing image pixel value stored in matrix is easy by using index of the matrix for example P(row,col), see figure 18. But in hardware, this job is more difficult, usually data is stored in memory with sequential/serial access on it, see figure 19. For accessing several pixel value in the same time like 3x3 window, an approach present in figure 18 is used.

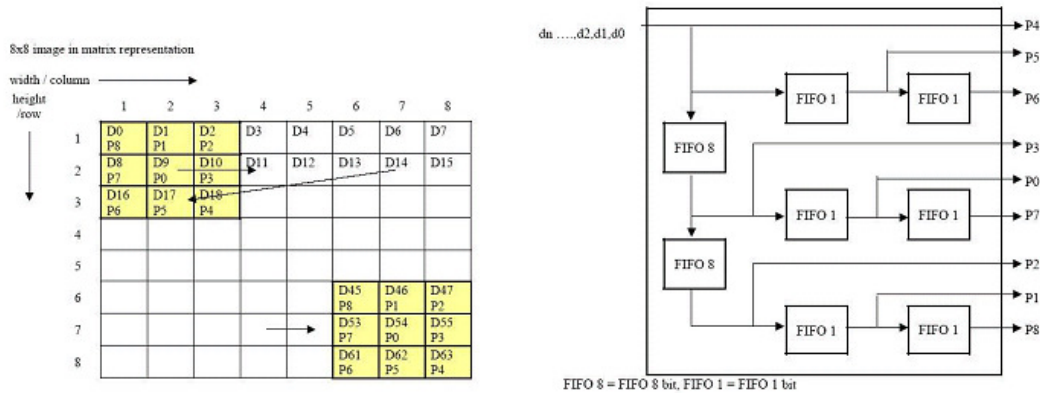


Figure18. Accessing image in matrix and Module block in hardware for accessing 3x3 window from the image.

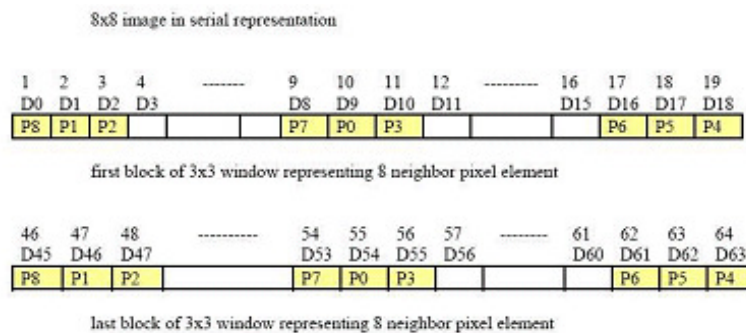


Figure19. Accessing image in serial/sequential.

A FIFO register will delay incoming data by several bit according to the position as needed. For 8x8 image the FIFO8 with 8 bit delayed is used, and for width x height a FIFO with width x height bit delayed is used. In consequence, there is b order area, because the first 3x3 window is achieved after $(2 \times \text{width} + 2)$ data is sending to this block/module. For example, the first 3x3 window in 8x8 image is achieved after D18 is sending to this module. It is happened also in the last 3x3 window, when all data has been send D63 in the cell 64, the center of the window is on D54.

This approach is implemented using VHDL programming and some modules (FIFO and Memory Module) are developed based on IP Core Generator.

3.2.4 Fingerprint Feature Extraction Module

Fingerprint extraction algorithm based on crossing number approach is implemented in FPGA device using several gates such as: Substraction, Addition, Inverter, Multiplexer and Divider (Shift Right Register). For hardware absolute value implementation, multiplexer device as shown on figure 20.a is used. The multiplexer gate operation is described in equation (3):

$$Y = \begin{cases} p, & \text{if } s = 1 \quad (\text{borrow}=1) \\ q, & \text{if } s = 0 \end{cases} \quad (3)$$

This multiplexer will chose either p or q as an output value (Y) depending on borrow value from subtraction operation $A - B$. The component p correspond to $B - A$ and is obtained using a two's complement operation of the direct output subtractor (q). Depending of the borrow value, the output of the multiplexer will correspond to p ($B - A$) or q ($A - B$).

The 8 subtractions gates with 2 inputs, 8 absolute values blocks, one adder with 8 inputs, and one shift right register gate dedicated to division by 2 operations for crossing number operation are needed. Figure 20.b. shows the logical schematic for Crossing Number formula.

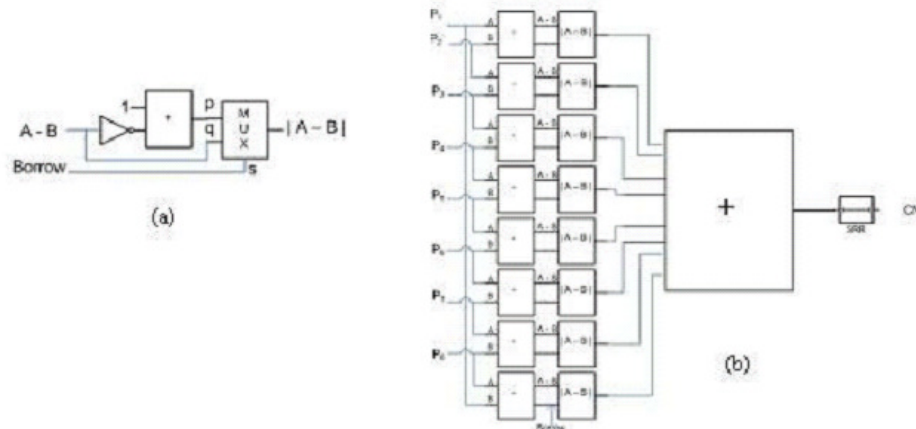


Figure20. Hardware implementation of minutiae points extraction process.

4. RESULT AND DISCUSSION

This performance evaluation consists of individual sub module result and overall performance using 3 data set and the diagram of this performance evaluation is presented in figure 21. Every fingerprint images from all data set will be extracted using proposed algorithm and mindtct program to obtain fingerprint template consist of minutiae points. And using bozorth3 matching program, the matching score of all the fingerprint images comparing to the others are calculated. And finally some rule to obtain the performance consisting of FAR, FRR and EER are applied. From them the ROC curve to present the performance of two-extraction algorithm for each data set are presented.

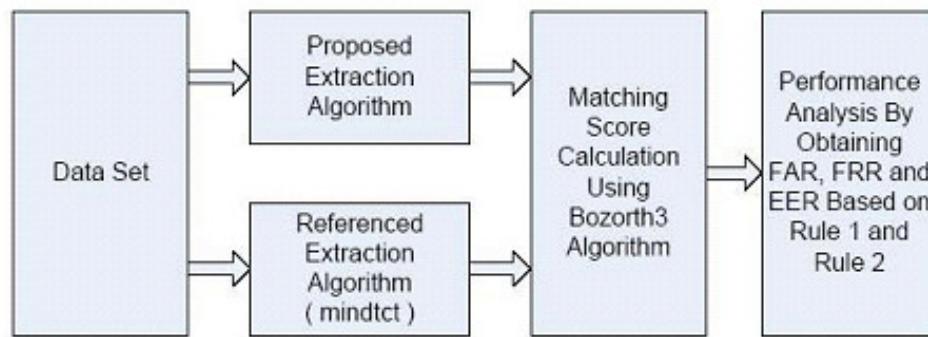


Figure21. Performance evaluation diagram

4.1 Data Set and Evaluation Method

To perform the algorithm evaluation, 3 data set consisting fingerprint images are used, such as:

1. Data set 1:

This fingerprint images data are from FVC2004 and consist of 80 fingerprints image from 10 users with 8 impression every user. According to [5] these fingerprint images are obtained from students (24 years old on the average) enrolled in the Computer Science degree program at the University of Bologna kindly agreed to act as volunteers for providing fingerprints.

At the end of the data collection, for each database a total of 120 fingers and 12 impressions per finger (1440 impressions) were gathered. As in previous editions, the size of each database to be used in the test was established as 110 fingers wide (w) and 8 impressions per finger deep (d) (880 fingerprints in all); collecting some additional data gave a margin in case of collection/labeling errors.

2. Data set 2:

This fingerprint images data are is captured using USB Fingerprint reader from all fingerprint of 10 people, it means that 100 users are obtained. 8 impression every fingerprint with some deformation are taken, individuals were asked to put the finger at a slightly different vertical position and to alternate low and high pressure against the sensor surface also make some rotations of 45 degree (left and right direction) then 180 degree. At the end of the data collection, totally 800 fingerprint images are collected.

3. Data set 3:

This fingerprint images data are captured using USB Fingerprint reader from all fingerprint of 10 people, it means that 100 users are obtained. 10 impression every fingerprint with some small deformation are taken, individuals were asked to put the finger at a slightly different vertical position and to alternate low and high pressure against the sensor surface but without any rotation. At the end of the data collection, totally 1000 fingerprint images are collected.

For calculating the FAR, FRR and EER we use this rule:

Each sample in the subset A is matched against the remaining samples of the same finger to compute the False Non Match Rate FNMR (also referred to as False Rejection Rate - FRR). If the matching g against h is performed, the symmetric one (i.e., h against g) is not executed to avoid correlation. The total number of genuine tests (in case no enrollment rejections occur) is:

$$(\text{Impression} * (\text{Impression} - 1)/2) * \text{Number of Data}$$

The first sample of each finger in the subset A is matched against the first sample of the remaining fingers in A to compute the False Match Rate FMR (also referred to as False Acceptance Rate - FAR). If the matching g against h is performed, the symmetric one (i.e., h against g) is not executed to avoid correlation. The total number of false acceptance tests (in case no enrollment rejections occur) is:

$$(\text{Number of Data} * (\text{Number of Data} - 1))/2$$

Or each sample of each finger in the subset A is matched against each sample of the remaining fingers in A according to the order of sample (impression).

$$((\text{Number of Data} * (\text{Number of Data} - 1))/2) * \text{impression}$$

4.2 Sub Program Evaluation and Experimental Result

4.2.1 Thinning Algorithm Experimental Result

The thinning algorithm is modified to match with fingerprint image representation value of pixel element in the image. Figure 22 show this thinning result for the same fingerprint image, for original algorithm the result is not good and the computation time 14.67 second (figure 22.a) longer than modified algorithm (0.42 second), and this modified algorithm present better result as can be seen in (figure 22.b).

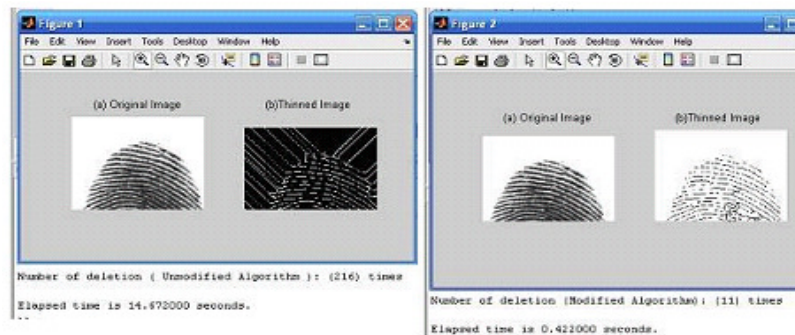


Figure22. Thinning result, (a) original thinning algorithm (b) modified thinning algorithm .

4.2.2 Minutiae Points Extraction Algorithm Experimental Result

For minutiae points extraction algorithm including pre processing, some results are shown in Figure 23 using fingerprint image file 101_2.tif from FVC2004. Figure 23.c shows that minutiae extraction process result from fingerprint image without enhancement was not well performed with so many false minutiae and unreasonable numbers of minutiae (more than 520 minutiae points). Better results are shown in figure 23.d.

With enhancement process, the result is clearly better with the number of minutiae points is 52. According to [21], it is more reasonable because the number of minutiae for good quality fingerprint image is around 40-100 points. Eliminating minutiae points at the edge of the fingerprint image produces a better result.



Figure23. Image enhancement result, (a) original image 101_2.tif, (b) thinned image with out enhancement (c) minutiae point detected with out enhancement and (d) minutia point detected with image fingerprint enhancement.

As presented in [44], table 1 is for minutiae points detected from both algorithm and table 2 show computation time result, with and without enhancement process. The enhancement process requires 12 second of computation time so the total processing time for 101_2.tif image is 18 seconds with enhancement process. The number of minutiae point detected is around 52 points. Without enhancement, the computation time takes 7 seconds and around 520 points of minutiae are detected including false minutiae points.

Table1. Minutiae Points Detected Using Algorithm With and Without Fingerprint Enhancement Process

Type	File	Computational Time (second)		
		BP	EP	Total
A	101_2.TIF	25	27	52
	102_2.TIF	41	15	56
	103_3.TIF	35	24	59
	104_4.TIF	50	17	67
	105_7.TIF	41	24	65
Average		38	21	59
B	101_2.TIF	376	144	520
	102_2.TIF	1095	75	1170
	103_3.TIF	1025	86	1109
	104_4.TIF	789	81	870
	105_7.TIF	420	181	601
Average		741	113	854

Note : Type of algorithm : A = With Fingerprint Enhancement Process,
 B = Without Fingerprint Enhancement Process

From these two tables, enhancement process is time consuming, but this process is needed to obtain good minutiae points and reduce false minutiae points detected by proposed algorithm. This result, 18 seconds for all fingerprint minutiae extraction process is better than other similar method proposed by Lorenzo et al which is 60-90 seconds just for thinning process part of all extraction process, as described in [45].

Table 2. Computation Time for Algorithm With and Without Fingerprint Enhancement Process.

Type	File	Computational Time (second)				
		(1)	(2)	(3)	(4)	(5)
A	101_2.TIF	11.25	0.015	0.29	6.07	17.63
	102_2.TIF	12.15	0.015	0.41	6.01	18.59
	103_3.TIF	12.25	0.015	0.25	6.29	18.81
	104_4.TIF	12.56	0.032	0.39	5.98	18.96
	105_7.TIF	12.56	0.032	0.36	6.57	19.52
Average		12.15	0.022	0.34	6.18	18.70
B	101_2.TIF	0	0.031	0.61	6.23	6.87
	102_2.TIF	0	0.031	0.63	7.11	7.77
	103_3.TIF	0	0.047	0.68	7.28	8.01
	104_4.TIF	0	0.031	0.64	6.84	7.51
	105_7.TIF	0	0.047	0.61	6.90	7.56
Average		0	0.037	0.63	6.87	7.55

Note : Type of algorithm : A = With Fingerprint Enhancement Process,
 B = Without Fingerprint Enhancement Process.

Computation Time : (1) fingerprint enhancement process, (2) binarization process, (3) thinning process, (4) minutiae extraction process, (5) total

Questionnaire consisting of 30 extracted fingerprint images are analyzed by 100 respondents visually to see how many minutiae points are not detected by this algorithm. These respondents are students of university at 19-22 years old. In [46], there is a result 12.3 % of minutiae points in the fingerprint cannot be detected by proposed algorithm. In detail, this result can be observed in table 3.

Table 3. Minutiae Points Not Detected by Proposed Algorithm

Group Respondent	Average of Minutiae Points Detected		
	BP	EP	All
Group 1	1.71	8.69	10.4
Group 2	1.72	8.03	9.75
Group 3	1.71	6.25	7.96
Group 4	1.79	5.84	7.63
Group 5	1.86	5.86	7.72
Group 6	1.99	5.94	7.93
Group 7	1.63	10.23	11.86
Group 8	1.63	6.55	8.18
Group 9	1.74	4.46	6.2
Group 10	1.65	4.92	6.57
Total Average	1.74	6.67	8.42
Average of Minutiae Points Detected	33.433	27.066	60.499
Percentage of Average Minutiae Points Not Detected (%)	4.955	19.787	12.37

4.3 Extraction Algorithm Performance Evaluation Result

The performance evaluation results using all data set are presented in figure 24. Overall performance of proposed algorithm is slightly better than mindtct algorithm. These values are 14.05 % comparing with 20.39 %. This result shows that there is improvement in proposed algorithm comparing to mindtct algorithm in extracting fingerprint minutiae point even there is no false minutiae removal process in proposed algorithm like in mindtct algorithm. If proposed algorithm has this post processing for removing false minutiae, the EER value will be better.

Figure 24 show the ROC curve for all data set in average, from this figure that is clear that proposed algorithm (blue line) has better performance comparing to referenced algorithm (green line). It can be seen in this curve, almost all curve of proposed algorithm is below the referenced algorithm. In Biometric Recognition, FAR and FRR is always in trade off condition. Because of that, the EER value (the error rate value when FAR=FRR) is concern, the smallest EER value is better, and the resulting value is 14.05 % comparing to 20.39 %.

4.4 Evaluation and Experimental Result on FPGA Board

For accessing fingerprint sensor MBF200 directly from the FPGA board, a module that generates control signals to activate and to access this sensor have been developed, simulated and tested using logic analyzer. Figure 25 presents the simulation result using Modelsim, and figure 26 is the result from logic analyzer when reading the control signal in I/O pin of FPGA board. These figures, show that there is a good sequence of generation signal from state 0,1,2 ...,B, and there is one clock for every state. There are 8 clocks for initialization (at least) and 2 clocks for every pixel read from the sensor. Figure 25 and 26 show 3 times reading of pixels, and without waiting time, there will be $76800 \text{ pixel elements} * (2 + 6 \text{ clocks waiting time}) = 614400 \text{ clocks}$.

There are 2 clocks for starting and closing state. There are 30 μ S waiting time before register CTRL A can be set and 28 clocks cycles before the first data is ready. Therefore,

$$\begin{aligned}
 \text{TotalTimes} &= \text{InitializationTimes} + \text{ReadingTimes} \\
 \text{InitializationTimes} &= 10\text{clocks} + 30\mu\text{S} + 28\text{clocks} \\
 \text{ReadingTimes} &= 614400\text{clocks}
 \end{aligned} \tag{4}$$

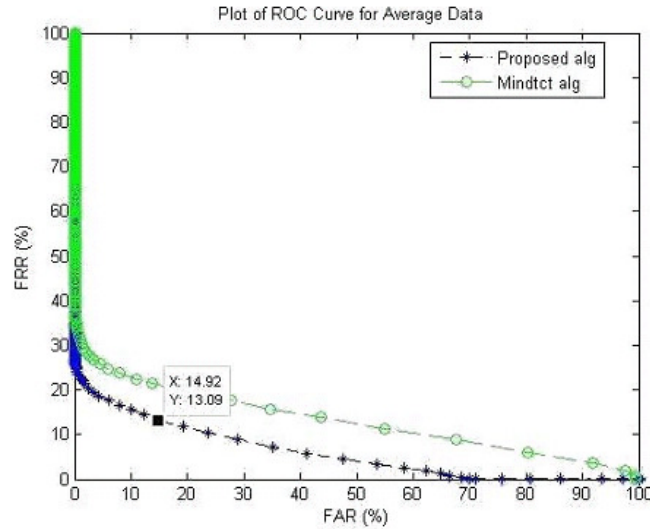


Figure24. Roc Curve for all dataset in average.

The clock cycle is running at 12 MHz, which means 0.083 μ S for every clock cycle. The initialization time is $38 * 0.083 \mu\text{S} + 30 \mu\text{S} = 33.154 \mu\text{S}$ and reading time is $614400 * 0.083 \mu\text{S} = 50.99 \text{ mS}$. The total times is 51.22 mS.

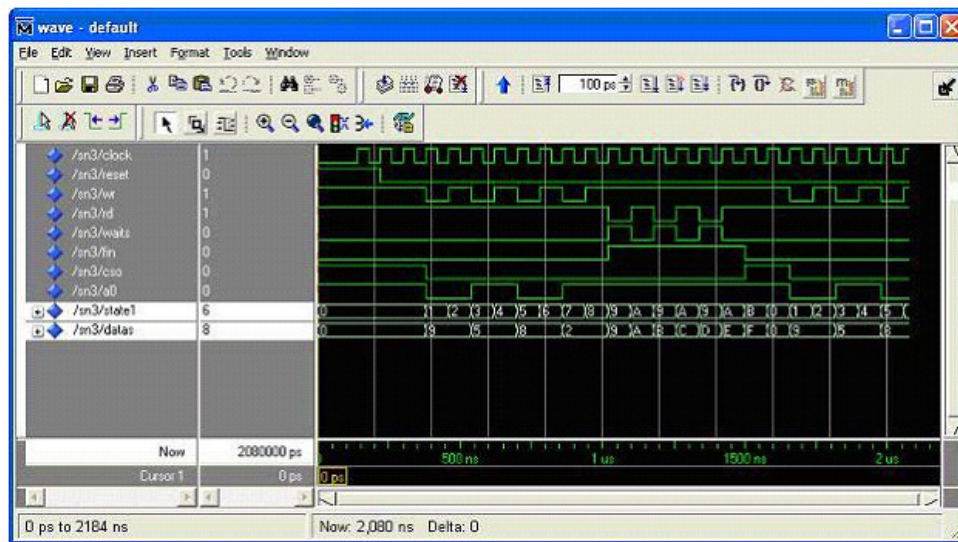


Figure25. Simulation result of control module for fingerprint sensor

For this evaluation, an application has been developed based on proposed algorithm using Visual C++ and VHDL programming for the hardware implementation. For this application, a fingerprint image is captured using USB fingerprint sensor and minutiae points are extracted in PC and also in FPGA board. The hardware implementation of the algorithm has been proposed in previous section, but the experimental result is for thinning process in hardware. An image from the sensor that has been enhanced is sent to the FPGA board to be thinned. In comparison with a PC (Pentium 4-2.9GHz with 512Mbytes), computational time with FPGA is 3 times faster. Implementation of other process in hardware will be continued.

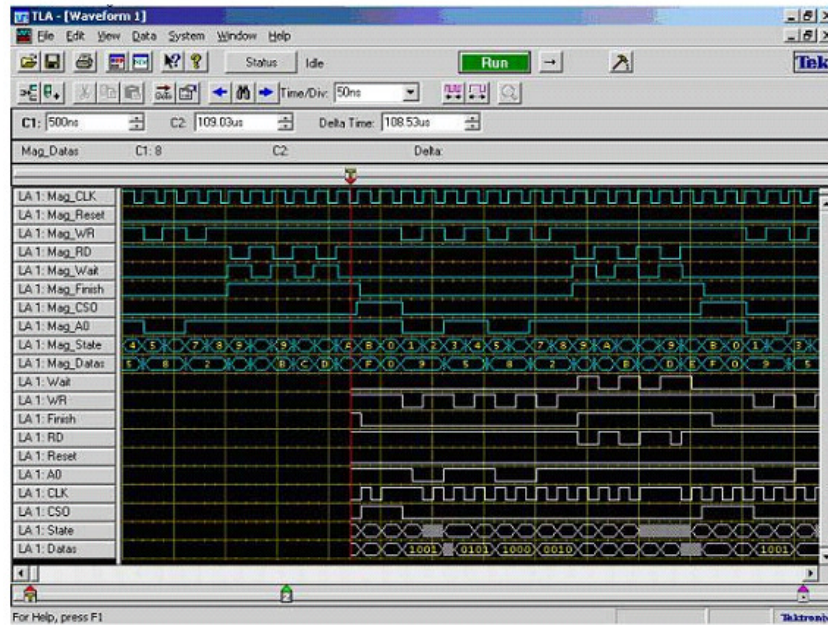


Figure26. Control signal for fingerprint sensor read by logic analyzer

5. CONCLUSIONS

In this section, the main research objectives and the key ideas of the fingerprint recognition system are recalled. A summary of the major achievements and indication of further research issues that need to be addressed are presented. The objectives of this research are, (1) to obtain simple (low complexity, hardware-adaptable) fingerprint feature extraction algorithm (2) to implement the proposed algorithm into embedded environment using FPGA devices. In obtaining fingerprint feature extraction algorithm, a minutiae point based algorithm has been developed, with:

- Angle Calculation Block technique for obtaining parameters which do not contain floating point operations which is well suited for hardware implementation,
- On average, computational time using standard PC (Pentium 4-2.9GHz with 512Mbytes) for fingerprint minutiae extraction process is 6.18 seconds, and all processes to extract fingerprint minutiae is 18.70 seconds. There is other similar method as reference, which is 60-90 seconds (using PC Pentium 4 - 3 GHz with memory 2GBytes) just for thinning process in preprocessing for extracting minutiae points.
- An average of 12.37 percents of minutiae points are missed with the proposed algorithm,

- Based on rule explained in 4.1, average EER of 14.05 percents shows better performance than mindtct algorithm as stated in the literature, which are 20.39 percents.

We implemented the proposed algorithm into the hardware environment. The modules for accessing fingerprint sensor have been developed, simulated and tested. For thinning process, there is a reduction of computational time, which is 3 times faster with FPGA. There are some difficulties in implementing image enhancement process in the hardware platform, because floating point operation still exists in this method. However, based on error measurements, the proposed algorithm out performs mindtct algorithm in terms of the percentage of the EER. From this result the advantages from proposed algorithm are: (1) simple operations, (2) low complexity, (3) avoiding floating points calculation, and (4) adaptable to hardware implementation.

ACKNOWLEDGMENTS

This work is financially supported by STMIK Jakarta STI&K in cooperation with Gunadarma University, Jakarta- Indonesia.

REFERENCES

- [1] James Wayman, Anil K. Jain, Davide Maltoni, and Dario Maio, (2005), "Biometric Systems : Technology, Design and Performance Evaluation", Springer-Verlag Limited, London.
- [2] Unibo, 2000. Fingerprint Verification Competition (2000), <http://bias.csr.unibo.it/fvc2000/>.
- [3] Unibo, 2002. Fingerprint Verification Competition (2002), <http://bias.csr.unibo.it/fvc2002/>.
- [4] David Maiao, Davide Maltoni, R. Cappelli, J.L. Wayman, and A.K. Jain, (2004), "FVC 2004 : Third Fingerprint Verification Competition", In Proceedings of International Conference on Biometric Authentication (ICBA). Hong Kong.
- [5] Unibo, 2004. Fingerprint Verification Competition (2004), <http://bias.csr.unibo.it/fvc2004/default.asp>.
- [6] Unibo, 2006. Fingerprint Verification Competition (2006), <http://bias.csr.unibo.it/fvc2006/>.
- [7] Anil K. Jain, A. Ross, and S. Prabhakar, (2004), "An Introduction to Biometric Recognition", IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Image- and Video-Based Bimetrics, 14(1):4– 20.
- [8] S. Kasaei, M. Deriche, and B. Baashash, (1997), "Fingerprint Feature Extraction Using Block-Direction on Reconstructed Images", IEEE Tencon (Speech and Image Technologies for Computing and Telecommunications), pages 303–306.
- [9] Salil Prabhakar, (2001), "Fingerprint Classification and Matching Using A Filterbank", Ph.D. thesis, Department of Computer Science and Engineering, Michigan State University.
- [10] Raimond Thai, (2003), "Fingerprint Image Enhancement and Minutiae Extraction", Master's thesis, University of Western Australia.
- [11] Sharat Chikkerur, 2005. "Online Fingerprint Verification System", Master's thesis, State University of New York at Buffalo.
- [12] Tsai-Yang Jea, (2005), "Minutiae-Based Partial Fingerprint Recognition", Ph.D. thesis, State University of New York at Buffalo.
- [13] Josef Strom Bartunek, (2005), "Minutiae Extraction from Fingerprint with Neural Network and Minutiae based Fingerprint Verification", Master's thesis, Blekinge Tekniska Hogskola.
- [14] N. Ratha, D. Rover, and Anil.K. Jain, (1995), "An FPGA-Based Point Pattern Matching Processor with Application to Fingerprint Matching", CAMP '95 Italy, pages 394–401.
- [15] N. Ratha, D. Rover, and Anil.K. Jain, (1996), "Fingerprint Matching on Splash 2", In J. Arnold D. Buell and W. Kleinfolder (eds.), editors, Splash 2: FPGAS in a Custom Computing Machine, IEEE Computer Society Press, pages 117–140.
- [16] Gwo-Cheng Chao, Shung-Shing Lee, Hung-Chuan Lai, and Shi-Jinn Horng, (2005), "Embedded Fingerprint Verification System", In Proceedings of the 2005 11th IEEE International Conference on Parallel and Distributed Systems (ICPADS'05).

- [17] Shenglin Yang, Kazuo Sakiyama, and Ingrid Verbauwhede, (2006), "Efficient and Secure Fingerprint Verification for Embedded Devices", URASIP Journal on Applied Signal Processing, 2006:1-11.
- [18] Maitane Barrenechea, Jon Altuna, and Miguel San Miguel, (2007), "A Low Cost FPGA-Based Embedded Fingerprint Verification Embedded Fingerprint Verification and Matching System", Fifth Workshop on Intelligent Solutions in Embedded Systems (WISES 07).
- [19] Christopher Baker, (2002), "Integrating the MCF5249 to Biometric Fingerprint Sensor", Freescale Semiconductor, Inc., application note edition.
- [20] Peter Bishop, (2002), "Atmel's FingerChip™ Technology for Biometric Security", Atmel Corporation.
- [21] Davide Maltoni, David Maio, and Anil K. Jain, (2003) "Handbook of Fingerprint Recognition", Springer Verlag.
- [22] A. Ross, S. Dass, and Anil K. Jain, (2004) "Estimating Fingerprint Deformation", In Proc. of International Conference on Biometric Authentication (ICBA). Hongkong.
- [23] Gerik Alexander von Graevenitz, (2003), "Introduction to Fingerprint Technology", A and S International, Taipei, 53:84-86.
- [24] ANSI, (2004), "Biometrics - Biometric Data Interchange Formats - Part 2: Finger Minutiae Data". ISO/IEC FCD 19794-2. ANSI.
- [25] Xudong Jiang, Wei Yun Yau, and Wee Ser, (1999), "Minutiae Extraction by Adaptive Tracing the Gray Level Ridge of the Fingerprint Image", IEEE, pages 852-856.
- [26] V.K. Sagar, Koh Jit Beng, and Alex, (1999), "Fingerprint Feature Extraction by Fuzzy Logic and Neural Networks", IEEE, pages 1138-1142.
- [27] Jinxiang Liu, Zhongyang Huang, and Kap Luk Chan, (2000), "Direct Minutiae Extraction from Gray-Level Fingerprint by Relationship Examination", IEEE, pages 427-430.
- [28] Bir Bhanu, (2000), "Logical Templates for Feature Extraction in Fingerprint Images", IEEE, pages 846-850.
- [29] Venu Govindaraju, Zhixin Shi, and John Schneider, (2003), "Feature Extraction Using a Chain-coded Contour Representation of Fingerprint Images In Audio-and Video-Based Biometric Person Authentication", 4th International Conference, AVBPA 2003, pages 268-275. Cedar (Center of Excellence for Document Analysis and Recognition), State University of New York, Guilford, UK.
- [30] Charles Sweeney and Matt Bowen, (2006), "ZestSC1 User Guide", Orange Tree Technologies, version 1.5 edition.
- [31] Sunny Arief Sudiro, Michel Paindavoine, and Tb. Maulana Kusuma, (2007), "Real Time Fingerprint Recognition Using FPGA Device", 4th Summer School for Advanced Study on Biometric for Authentication: New Technologies and Embedded Systems, Alghero-Italy.
- [32] Zhaoliang Wang, (2003), "PerfectPrint for PC API 8.4 Release . 1.0", STMicroelectronics Group of Companies.
- [33] Anonim, 2001. MBF 200 Solid State Fingerprint Sensor. Datasheet.
- [34] Feng Zhao and Xiaohou Tang, (2002), "Preprocessing for Skeleton-Based Fingerprint Minutiae Extraction", In Proceeding of the International CISST - Conference on Imaging Science, System and Technology.
- [35] L. Hong, Y. Wan, and Anil K. Jain, (1998), "Fingerprint image enhancement: Algorithm and performance evaluation", IEEE Transactions on Pattern Analysis and Machine Intelligence, 20:777-789.
- [36] Peter D Kovesi, (2006), "MATLAB and Octave Functions for Computer Vision and Image Processing.", School of Computer Science and Software Engineering, The University of Western Australia. Available at : <http://www.cs.uwa.edu.au/pk/Research/MatlabFns/index.html>.
- [37] Sunny Arief Sudiro, Michel Paindavoine, and Tb. Maulana Kusuma, (2007), "Simple Fingerprint Minutiae Extraction Algorithm Using Crossing Number on Valley Structure", 5th IEEE Workshop on Automatic Identification Advanced Technologies AutoID2007, Alghero Italy, pages 41-44.
- [38] Sunny Arief Sudiro, Michel Paindavoine, Trini Saptariani, and Rudi Trisno Yuwono, (2007), "Obtaining Parameter of Minutiae Points Detected by Crossing Number Algorithm on Valley Structure", International Conference SI IT 2007, Bali-Indonesia, pages 104-109.
- [39] Sunny Arief Sudiro, (2005), "Thinning Algorithm for Image Converted in Fingerprint Recognition System", National Seminar Soft-Computing Intelligent Systems and Information Technology 2005, Universitas Kristen Petra, Surabaya.

- [40] Rafael C. Gonzales and Paul Wintz, (1987), "Digital Image Processing", Addison-Wesley Publishing Company, Inc., Canada, 2nd edition.
- [41] Erin Hastings, (2004), "A Survey of Thinning Methodologies", College of Engineering and Computer Science, University of Central Florida. <http://www.cs.ucf.edu/hastings/papers/Thinning-MethodologiesHastings.zip>.
- [42] Mark Zwolinski, (2000), "Digital System Design with VHDL", Pearson Education Ltd, Edinburgh Gate, Harlow, Essex CM20 2JE, England.
- [43] Sunny Arief Sudiro, Michel Paindavoine, and Tb. Maulana Kusuma, (2008), "Improvement of Fingerprint Sensor Reading Using FPGA Devices", International Conference On Computer and Electrical Engineering (2008), Phuket-Thailand, ISBN : 978-0-7695-3504-3, pages 829–833.
- [44] Sunny Arief Sudiro, Michel Paindavoine, and Tb. Maulana Kusuma, (2007), "Image Enhancement in Simple Fingerprint Minutiae Extraction Algorithm Using Crossing Number on Valley Structure", International Conference IAS 2007, ISBN: 1-4244-1355-9, Kuala Lumpur- Malaysia.
- [45] V. L'opez Lorenzo, P. Huerta Pellitero, J. Castillo Villar, and J.I. Martnez, (2005), "Fingerprint Minutiae Extraction Based On FPGA and MatLab", XXth Conference on Design of Circuits and Integrated Systems (DCIS 2005), ISBN 972-99387-2-5, Lisboa-Portugal.
- [46] Sunny Arfief Sudiro, Michel Paindavoine, Rudi Trisno Yuwono, and Lussiana, (2008) "Performance Evaluation of Simple Fingerprint Minutiae Extraction Algorithm Using Crossing Number on Valley Structure", 5th International Conference on Innovations in Information Technology (Innovations 2008), Al Ain City-UAE, ISBN :978-1-4244-9-3397-1, pages 623–627.

Authors

Sunny Arief Sudiro received the Bachelor degree from Gunadarma University in Jakarta Indonesia, in 1994, the Master Magister degree from Gunadarma University in Jakarta Indonesia, in 1998, and the Doctoral degree in Information Technology from Gunadarma University in Jakarta Indonesia, in April 2009. He also received Doctoral degree in Instrumentation and Image Informatics from Université de Bourgogne in October 2009. From 2010 up to now, he is a senior member of International Association of Computer Science and Information Technology (IACSIT) and also chief of Computer Science Study Program in STMIK Jakarta STI&K.



Rudi Trisno Yuwono received the Bachelor degree from Gunadarma University in Jakarta Indonesia, in 1993, the Master Magister degree from Gunadarma University in Jakarta Indonesia, in 1997, and the Doctoral degree in Information Technology from Gunadarma University in Jakarta Indonesia, in 2010. From 1994 up to now, he is a Senior Staff of IT Department and senior lecturer in Gunadarma University.

