Binary Operation Procedure for Thinning Process and Its VLSI Implementation in a Pixel Array Circuit

Kuo-Ting Wu

A Thesis
in
The Department
of
Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science at
Concordia University
Montreal, Quebec, Canada

March 2005

Your file Votre référence
ISBN: 0-494-13247-7
Our file Notre référence
ISBN: 0-494-13247-7

Abstract

Binary Operation Procedure for Thinning Process and Its VLSI Implementation in a Pixel Array Circuit

Kuo-Ting Wu

Thinning process is an important operation in image processing systems. It is often used to extract features of an image to facilitate further processes in many applications, such as the minutiae detection in fingerprint identification systems. Thinning process is usually implemented in a system of data processors that may not meet the requirements of size and power dissipation in applications of small and portable systems.

In this thesis, the design of a pixel array circuit for thinning process is presented. The research efforts of this work are on the two aspects: the processing algorithm and the circuit design. In the aspect of algorithm, a new thinning process is designed aiming at an easy implementation in a VLSI pixel array circuit. The computation is made in parallel and with a small number of direct data transfers among the neighboring pixels, which leads to a fast processing and fewer metal connections in the circuit. The effectiveness of the algorithm has been proved in the simulation.

The other aspect of this work is the design of a pixel array circuit implementing the image acquisition and the proposed thinning process. In this circuit, each pixel needs only four connections to its neighbors, and requires only simple logic gates and memory cells for the thinning process. The operations in each pixel are synchronized with the inter-pixel signal transfer. Partial results are shared by the pixels in the neighborhood to reduce the data traffic, to shorten the processing time and to minimize the number of the gates required for the computations. Furthermore, the power dissipation is made very low by means of minimizing the activity rate of the circuit. The pixel can be implemented in a silicon space of $50\times50\ \mu m^2$, if a $0.18\ \mu m$ CMOS technology is used. The power efficiency has been verified by electrical simulation. The circuit can be used in image processing systems for different applications such as security check, personal and object identifications.

Table of Contents

List of Figures

Acronyms and Abbreviations

ASIC	Application Specific Integrated Circuit
CMOS	Complementary Metal Oxide Semiconductor
CPU	Central Processing Unit
DSP	Digital Signal Processing
IC	Integrated Circuit
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
PDN	Pull-Down Network
PTL	Pass-Transistor Logic
SOC	System-On-A-Chip
VLSI	Very Large Scale Integration

1

INTRODUCTION

Thinning process is widely used in image processing applications to extract the features of an image, reduce the size of an image, and simplify the complexity of an image for further processes. In particular, it is an important operation in pattern recognition systems for fingerprint identification [1], medical image analysis [2], printed circuit inspection [3], and optical character recognition [4]. In many applications, a high-speed and real-time thinning process is needed. Especially, the demand for the applications of fingerprint recognition to small portable systems for security and user authentication is increasing [5]. An implementation suitable for small-volume and

low-power applications is becoming an important issue. However, conventional camera-memory-processors implementations may not meet the requirements of such applications in terms of size and power dissipation. Hence, studies of investigation of existing algorithms and their implementations need to be conducted, and new algorithms and systems of small volume, low power, and relatively high speed, need to be developed.

1.1 Motivation and Objective

In a conventional camera-memory-processors system, a large amount of data needs to be transmitted among the modules of the system, which restricts the processing speed. Also, separated modules for signal acquisition, processing and storage, do not yield the optimal results in terms of power dissipation. For image signal acquisition and processing in applications requiring high speed, small volume and low-power dissipation, pixel array circuit implementations, instead of camera-memory-processors ones, can be more suitable and efficient.

In a pixel array circuit, each of the pixels comprises an acquisition unit and a processing one, as shown in Fig.1-1. Each acquisition unit transduces the input signal, e.g. the incident light, into a current signal, and then the signal could be processed by means of various operations implemented in the processing unit. The signals can be transferred among the adjacent pixels via the interconnections according to the algorithm implemented. Each pixel provides the signal that can be the results of processing, instead of the acquired one. The pixel array circuit can thus perform a combined function of camera and processor in a conventional system. Hence, the size and the power

dissipation of the image processing system can be reduced, and the speed of the processing improved.

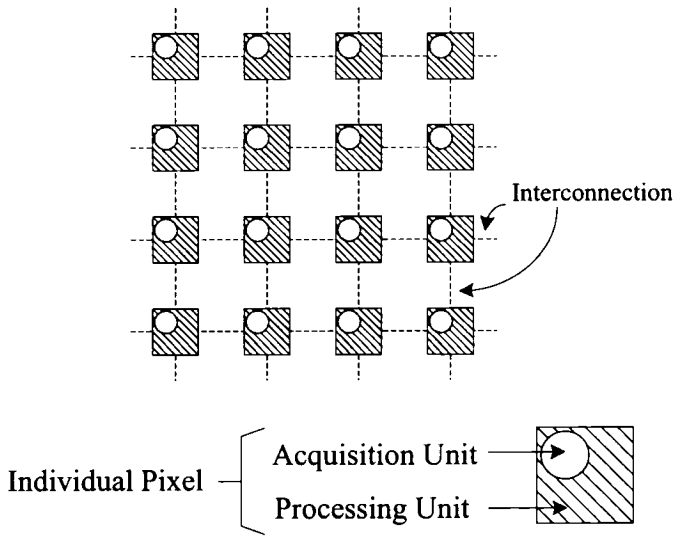


Fig.1-1 Pixel array structure. The processing unit at the pixel level processes image signals acquired by the acquisition unit.

The objective of the work of this thesis is to develop a pixel array circuit for image acquisition and thinning process. It should include two stages. The research on defining an algorithm suitable for the implementation in pixel array circuits needs to be conducted in the first stage of the work. In the second stage, a pixel array circuit for image acquisition and the proposed thinning process will be designed. The focus of the design is to have a great simplicity of the pixel circuit in order to obtain acceptable resolution of the processed image. It includes designing a simple structure with a minimum number of devices used in each pixel and that of interconnections.

1.2 Scope and Organization of the Thesis

Like most of the operations in image processing, those in a thinning process are usually repetitive computations, which are suitable for parallel processing in a pixel array circuit. However, in such a circuit, to achieve an acceptable resolution, the area for each pixel is limited, so the operations performed in each pixel should be simple. Also, the density of metal inter-pixel connections needs to be minimized. Otherwise, many problems, such as cross-talk and requirement for large area, would arise. Thus, to achieve the objective, the following research areas are included in the work of this thesis.

For the research on algorithms of thinning process, comprehensive studies of the existing algorithms will be carried out. These algorithms, although some of them look simple, are not easy to be implemented in pixel array circuits, as they may lead to complex interconnection structure and/or pixel circuit unit. The research effort in this part of the work is to find an efficient algorithm of thinning process that can be easily implemented in a silicon pixel circuit with a minimum number of interconnections and devices per pixel.

In the circuit design, the constraints of limited area per pixel may favor analog processing more than the digital one. However, the operation accuracy of analog processing is usually poor. Thus, the processing unit to be integrated in a pixel of the circuit is preferably digital. Nevertheless, the area of each pixel should still be minimized, which requires an optimized algorithm, in terms of suitability for circuit implementation, and a design methodology adapting to the features of the computations in the algorithm.

This thesis is organized as follows. In Chapter 2, a brief description of existing parallel thinning algorithms is given, and some designs of pixel array circuits are presented. In Chapter 3, a thinning process suitable for pixel array circuit implementation is proposed and its effectiveness is demonstrated by simulation results. Chapter 4 is dedicated to the description of the design of the pixel array circuit, in which the proposed algorithm is implemented. Finally, in Chapter 5, the work of the research is summarized, and some directions for future study are suggested.

2

BACKGROUND AND RELATED WORK

The work of this thesis is related to two aspects: thinning algorithms and pixel array circuits. In this chapter, a brief description of the background and related work with respect to existing parallel thinning algorithms is given, and some existing designs of pixel array circuits are described. Furthermore, the issues concerning the design of pixel array circuits for thinning process are addressed.

2.1 Parallel Thinning Algorithms

The objective of a thinning process is to thin one kind of regions of an image as much as possible while preserving the original connectedness. In a thinning process, every pixel is examined with respect to the value of its neighboring pixels, and if it satisfies certain conditions not to be a pixel constituting the skeleton of its region, its value will be deleted, i.e. reset to 0.

A thinning process contains a sequence of calculations with the signals from the neighboring pixels. According to the way of sequencing the signals required for the calculations, the algorithms for the thinning process can be classified as sequential or parallel ones. In parallel thinning algorithms, the signal in a pixel in a given step, e.g. N_{th}, is generated with the signals of its neighboring pixels resulting from the previous step, i.e. $(N\text{-}1)_{th}$, whereas in sequential algorithms, the signals of the neighbors may result from different steps of computation. As the signals of all the pixels in a parallel algorithm are processed simultaneously, parallel thinning algorithms are more suitable for the implementation in pixel array circuits.

Several algorithms for parallel thinning have been developed in the past four decades [6]-[18]. The term of "thinning" may be expressed in different ways, e.g. skeletonization and medial axis transform. An early work on thinning process is described in §2.1.1, and some later developments relevant to the thesis work in §2.1.2. In most of the thinning algorithms, the calculations are applied to binary pixel signals based on a 3x3 window, as shown in Fig.2-1. Conventionally, the region pixel having a logic value of 1, and logic-0 for those in the background otherwise.

P_8	P_1	P_2
P_7	P	P_3
P_6	P_5	P_4

Fig.2-1 Pixels in a 3x3 operation window

2.1.1 Early Work on Thinning Process

The work presented by Rutovitz in 1966 [6] is usually considered as an early one in the development of thinning algorithm. Its basic operation involves:

1) Examining if pixels are at the edges of regions.
2) Identifying the pixels that are at edge positions but not part of the skeleton.
3) Deleting the pixels identified in the above processes.
4) Repeating the processes for the thinning until a region is reduced to its skeleton.

The deletion conditions proposed in this thinning algorithm are presented by the following logic expressions:

(a_1) $X(P) = 2$, where $X(P) = \sum_{i=1}^{8} |P_{i+1} - P_i|$ and $P_9 = P_1$

(a_2) $b(P) \geq 2$, where $b(P) = \sum_{i=1}^{8} P_i$

(a_3) $P_1 \times P_3 \times P_7 = 0$ or $X(P_1) \neq 2$

(a_4) $P_1 \times P_3 \times P_5 = 0$ or $X(P_3) \neq 2$

where P_1, P_2, ... P_8 are the pixel values in the 3×3 window shown in Fig.2-1.

Condition (a_1) is used to examine if a pixel is at the edge of a region by counting the number of 0-1 (or 1-0) transitions between the neighboring pixels and comparing this

number with a threshold value of two. In the calculation related to (a_2), eight surrounding pixels are summed up and compared with a threshold of two to identify if the pixel is located at the end of a skeleton. As all the pixels are processed in parallel, simultaneous deletions can be performed in both side of a region. There is thus a risk of completely deleting pixels of a region with two-pixel width, as shown in Fig.2-2. Additional conditions (a_3) and (a_4) are used to identify such a case, so that this kind of over-deletion will be avoided. With this algorithm, pixels will be deleted, i.e. reset to 0, if they satisfy conditions (a_1), (a_2), (a_3) and (a_4).

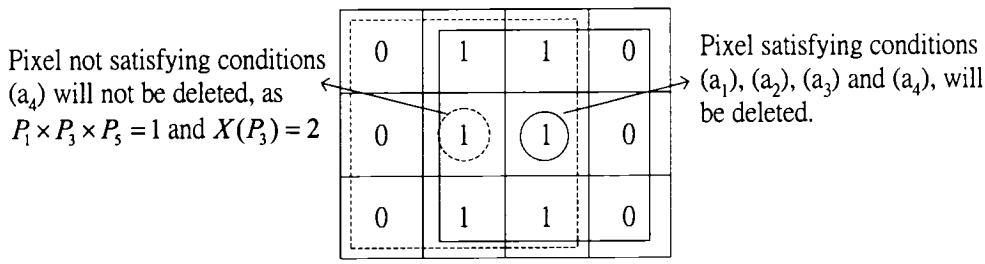


Fig.2-2 Non-deletion case.

Despite the precautions taken, the examination of the deletion conditions in this process does not cover all the critical cases. It may result in over-deletion or under-deletion in certain patterns, such as diagonal patterns with two-pixel width. Also, due to the asymmetric nature of logic expressions (a_3) and (a_4), some of the skeletons may not lie centrally in the original regions, such as the pattern shown in Fig.2-3. Later developments aiming at solving some of these problems are described in the following sub-chapter.

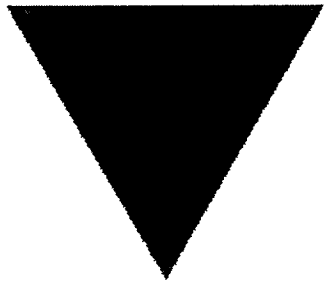


Fig.2-3 Skeletons that do not lie centrally in the original region, due to the asymmetric nature of the logic expressions (a_3) and (a_4) in [6]. The area in gray is the original region. The block lines are the skeletons after the thinning process.

2.1.2 Later Developments

Based on the early work mentioned in the previous sub-chapter, several algorithms have been developed. One of them is presented in [7]. In this algorithm, two additional conditions are added to fix the problem of the under-deletion of diagonal patterns. In the thinning algorithm presented in [8], the deletion conditions are divided into two sub-sets, corresponding to two sub-iterations. The two sub-sets of conditions are designed to detect the edge pixels at two opposite directions of a region. In this way, the problem of the asymmetric nature of the logic expressions (a_3) and (a_4) in [6] and [7] can be solved.

As mentioned above, more conditions are added to the algorithms proposed after [6]. The procedures of these algorithms become more complex. A thinning algorithm, initially reported in [9] and later included in [10], is proposed based on [6], with a simpler computation procedure, but it provides a better result. However, there are still some cases of over-deletion or under-deletion to be fixed. For example, diagonal patterns with two-pixel width could be over-deleted with [9]. To prevent this problem, a suggestion for changing the minimum threshold value, which is used to identify the pixels at the end of skeletons, to three is proposed in [11]. As expected, this modification

prevents the over-deletion but, in consequence, the diagonal patterns cannot be thinned to one-pixel width. The thinning algorithm presented in [12] suggests adding more conditions in each sub-iteration to thin the diagonal patterns into one-pixel width without the over-deletion of two-pixel-wide diagonal patterns. Similar approach is also used in [13], in which the more condition checking is added. Eventually, more detection of conditions requires more computations in the procedure.

Besides the thinning algorithms mentioned above, some other ones are also reported. These algorithms do not aim at detecting edge pixels or 0-1(or 1-0) spatial transitions in neighborhood. In the algorithm presented in [14], the pixels in a given window are compared with a set of predetermined patterns, and based on the comparison results, deletion decision may be made. In such an algorithm, as a large number of patterns are used for comparison, longer procedure is required. An algorithm reported recently [15] is also based on comparison of patterns.

2.1.3 Summary of the Algorithm Development

The objective of studying algorithms is to identify the detection methods suitable for parallel processing implementation. To be implemented in pixel array circuits, simple computations and communications among pixels in a thinning process are required.

Most of the existing thinning algorithms involve the detection of the pixels at edges and those at skeletons. Those detections can be done by counting the 0-1 (or 1-0) transitions, summing pixel values, and comparisons. Different algorithms are proposed to solve various problems of over or under deletions. Additional conditions are

introduced, attempting to make a "perfect" preservation of the connectedness, which usually results in more complex procedure.

It should be mentioned that the existing thinning algorithms are not developed for implementations in pixel array circuits, and thus the way of computation and communication among the pixels may not be easily implemented in pixel array circuits. Hence, for the work of this thesis, a "full-custom" thinning process needs to be developed.

2.2 Pixel Array Circuits for Image Processing

Many pixel array circuits have been reported in the past two decades. Most of them are developed for varieties of detections based on pre-image processing. Only a few of them are for thinning process. In the following sub-chapters, some of the pixel array circuits are described.

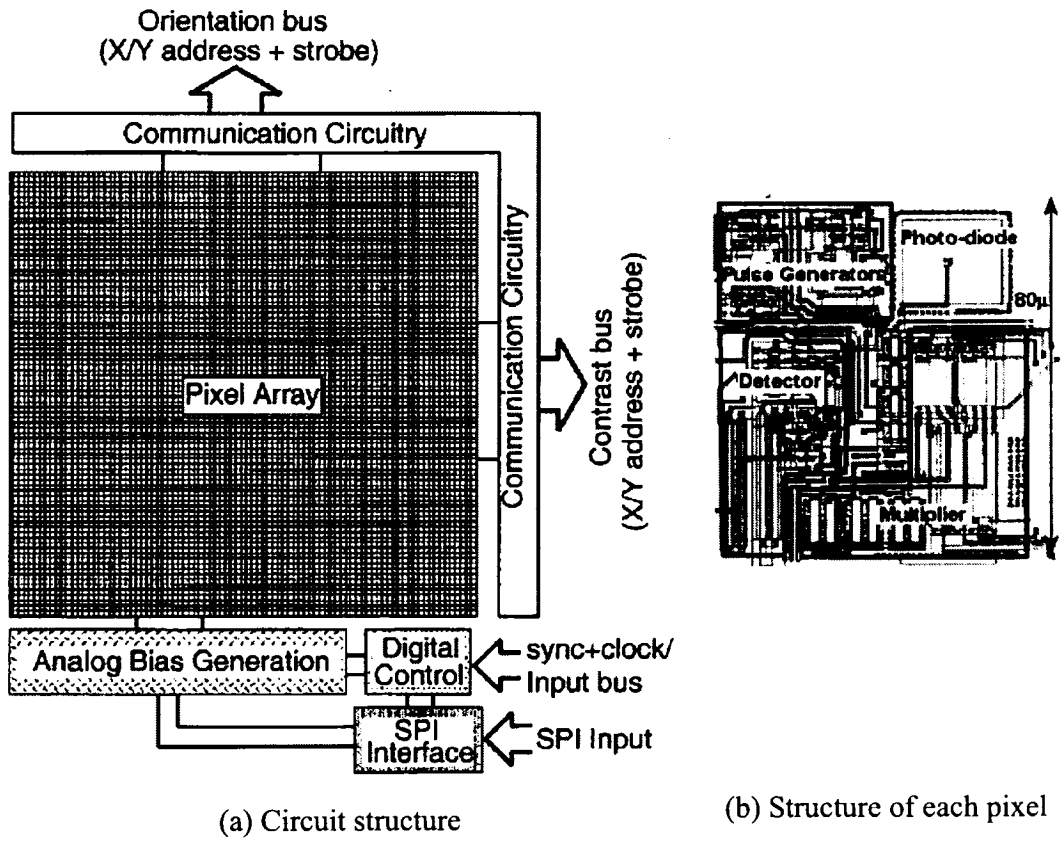
2.2.1 Early Pixel Array Circuit

Motion detection chip in the optical mouse presented by Lyon in 1981 [19] is one of the earliest pixel array circuits designed and implemented in silicon. It consists of an array of pixels, in each of which analog and digital circuitry is integrated. These pixels are connected in certain way so that the neighboring pixels can have the communications as defined in the implemented detection algorithm. The incident light is transduced into a current in each pixel, and it then is converted into a binary voltage signal. The pixels of the array operate in parallel and produce a 2-D signal reflecting the motion of the mouse. It shows some of the very fundamental aspects of the design of pixel array circuits:

- Implementing data acquisition and simple processing in each pixel to have parallel operations in the array.
- Using simple devices for simple operations so that the area required for each pixel can be minimized.
- Determining the structures of the inter-pixel connections to implement designed data communications among pixels.

2.2.2 Pixel Array Circuits for Image Processing

Thanks to the development of both fabrication technologies and circuit design, the pixel array circuits presented lately are made to have small area per pixel and to integrate more functions. One of them is an array circuit presented in [20] for the extraction of magnitude and direction of image's spatial gradients. The structure of the circuit is shown in Fig.2-4, and the structure of each pixel as well. In this circuit, each pixel contains a photodiode for the light-to-current transduction along with the analog circuitry for signal processing. As shown in Fig.2-5, the light acquired by the photodiode is transduced into a current, and then the current is input to the multiplier in the pixel. Using the current signal as the input one, four signals are generated by the multiplier, and are sent to the four adjacent pixels respectively. Therefore, the detector in each pixel receives four signals from its four adjacent pixels. Based on these four signals, two signals encoding the magnitude and direction of the spatial gradients are generated in each pixel, which are collected by the communication circuitry, as shown in Fig.2-4(a), and are output to the user as pulses distributed signals.



(a) Circuit structure

(b) Structure of each pixel

Fig.2-4 Array circuit reported in [20].

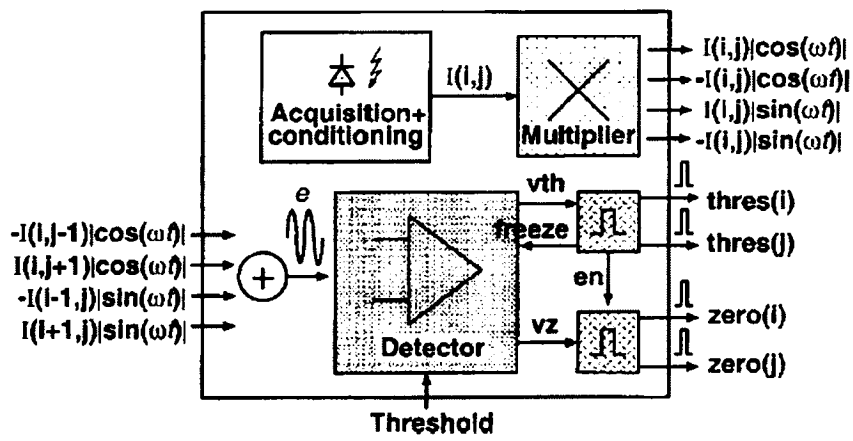


Fig.2-5 Block diagram of a pixel.

As the extensive computations are required in most of thinning processes, only a few array circuits for thinning process have been reported and fewer implemented. The one with silicon implementation is presented in [21]. As shown in Fig.2-6, a hexagonal arrangement of the pixels is used in this design, and each pixel consists of a sensor unit and a processing unit. The sensor unit acquires the fingerprint image by the capacitive sensing scheme, as shown in Fig.2-7, and output a binary voltage signal. The binary signal is stored in the memory of the processing unit for the following image process.

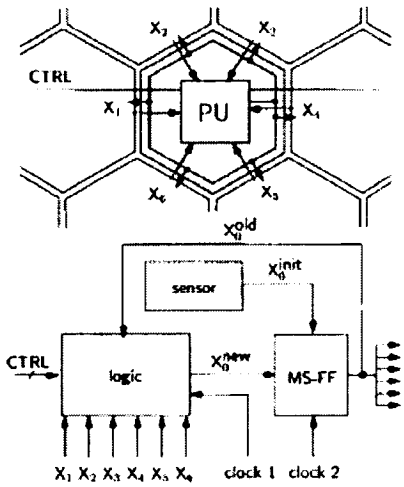


Fig.2-6 Basic structure and input-output configuration of a pixel with its six neighboring pixels in [21].

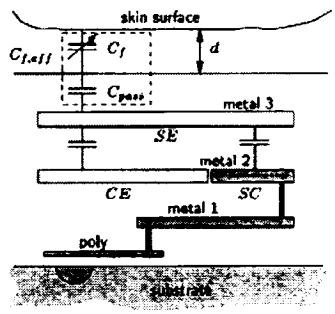


Fig.2-7 Schematic cross section of the acquisition unit in [21].

The thinning algorithm implemented in the processing unit is based on the comparisons between the sum of the signals of the neighboring pixels and the given threshold values. The sum operations are performed in a quasi-analog manner by using float-gate transistors with several coupling gates, as shown in Fig.2-8. The binary signal stored in the memory is converted into current signals first, added together, and then compared with the threshold signal that is also converted into current one for this comparison.

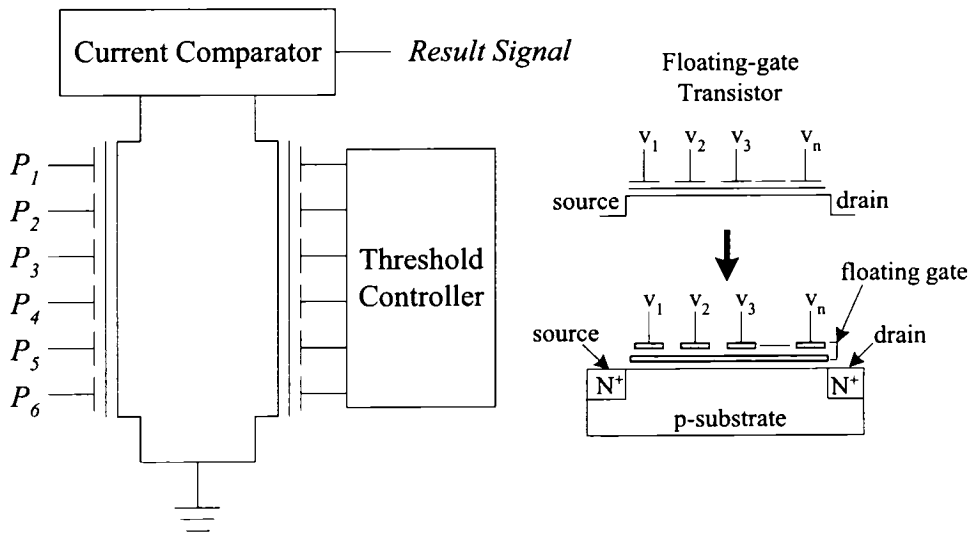


Fig.2-8 Structure of the implemented logic operations in [21]. The current controlled by the floating-gate transistors corresponds to the size of the coupling gates. As the size of each gate here is same, the current corresponds to the number of control signals, i.e. V_i, in this structure.

In this circuit, each pixel contains 123 transistors, and four of them are of the floating-gate type. The power dissipation per pixel is about 6.25×10^{-9} watts, when the clock rate is set to perform a complete thinning process for a fingerprint image in 20 ms. As the floating-gate transistors are used in this design, this circuit needs be implemented

in a double-poly fabrication process.

Except the implementation of operations in analog mode, an array processor for parallel thinning process but implemented with logic gates is reported in [22]. In this design, pixels are placed orthogonally. The thinning algorithm implemented in this design is based on the function

$$X(P) = \sum_{i=1}^{8} |P_{i+1} - P_i|, \text{ where } P_9 = P_1.$$

Without any specific arrangement for the data transfer, each pixel has a minimum of eight connections with their eight adjacent neighboring pixels. As the thinning algorithm is implemented with logic gates instead of analog circuitry, the pixels in this design have homogeneous characteristics, but the number of transistors in each pixel is large.

Another pixel array circuit is proposed in [23], which integrates a fingerprint image acquisition and identification in a single chip. But in that structure, the image thinning process is done in an external processor, and the results of the thinning process are then fed back to the pixel array for the computation of identification. The connections between the array circuit and the external processor are required in this design. Some other pixel array circuits are presented for capacitive fingerprint sensor recently, e.g. automatic calibration circuit scheme [24], fingerprint-image adjustment scheme [25], pixel-parallel image enhancement and rotation scheme [26], and automatic local contrast adjustment scheme [27]. However, these circuits are not designed for the implementation of thinning algorithm.

2.3 Conclusion

In this chapter, the principles used in different computation procedures for thinning process have been studied, and the analysis on some of the well-known algorithms has been presented. The focus of the studies on the algorithms is on the analysis of the data structure and the complexity of the computations of these algorithms to assess the feasibility of their implementations in VLSI pixel array circuits. Also, some of existing VLSI pixel array circuits have been described. Two of them are reported to be used for thinning process. One of them has analog operations implemented, and the circuit has to be fabricated in double poly process, which may not facilitate its integration in digital systems. The other pixel array circuit for thinning process has digital process implemented, but the number of transistors used in each pixel may not help to yield a small area for a high density of the pixel array.

Based on the studies of the related work, in the design of a thinning process and its implementation in a pixel array circuit, the following issues should be addressed:

1) To place a large number of pixels in a limited semiconductor space for an acceptable resolution, the area for each pixel should be minimized.
2) It would be desirable not to have arithmetic operations in the thinning process, as they usually need space-consuming complex digital blocks. One may use analog modules for such operations, but it is difficult to achieve homogeneity of characteristics of each pixel due to technological process variations. As the input for thinning process is a binary image, it is thus reasonable to develop a parallel thinning process involving only simple logic operations and inter-pixel data transfer.

3) The inter-pixel data transfer for computation is a particularly critical issue in the design of the thinning process, as it is related to the number of inter-pixel connections to be implemented in the pixel array.
4) To facilitate the integration of a circuit in digital signal processing systems, the circuit should be designed to be fabricated in a single-poly process.

3

THINNING PROCESS

In this chapter, the development of a thinning process suitable for the implementation in pixel array circuits is presented. In the thinning algorithms described in Chapter 2, the results of each pixel can be obtained by logic and arithmetic computation on the originally acquired signals. These algorithms can be implemented for parallel processing, but they are not developed for pixel array implementation. The proposed thinning process should be designed to have the computations in parallel and

to include only simple logic operations so that the structure of each pixel will be simple enough to be implemented in a small space and the array circuit will have an acceptable resolution. The basic points of the proposed thinning process are described in §3.1. The detection of edge pixels of a region is described in §3.2. The logic operations for examining the conditions of deletion are presented in §3.3. The procedure of the process is described in §3.4. Also a reformulated scheme is proposed in §3.5 to simplify the data transfer of the procedure in order to further facilitate the implementation of pixel array circuits. In §3.6, simulation results with images of different patterns, including fingerprints are presented to demonstrate the effectiveness of the algorithm.

3.1 Basic Points of the Proposed Thinning Process

The objective of a thinning process is to thin one kind of regions, such as ridge regions in a fingerprint image, as much as possible while preserving the original connectedness of the regions. The region pixels are assumed to have a value of logic-1 and background pixels of logic-0. In a thinning process, every pixel is examined with respect to its neighbors, and if it satisfies certain conditions not to be a pixel constituting the skeleton of its region, its value will be deleted to 0.

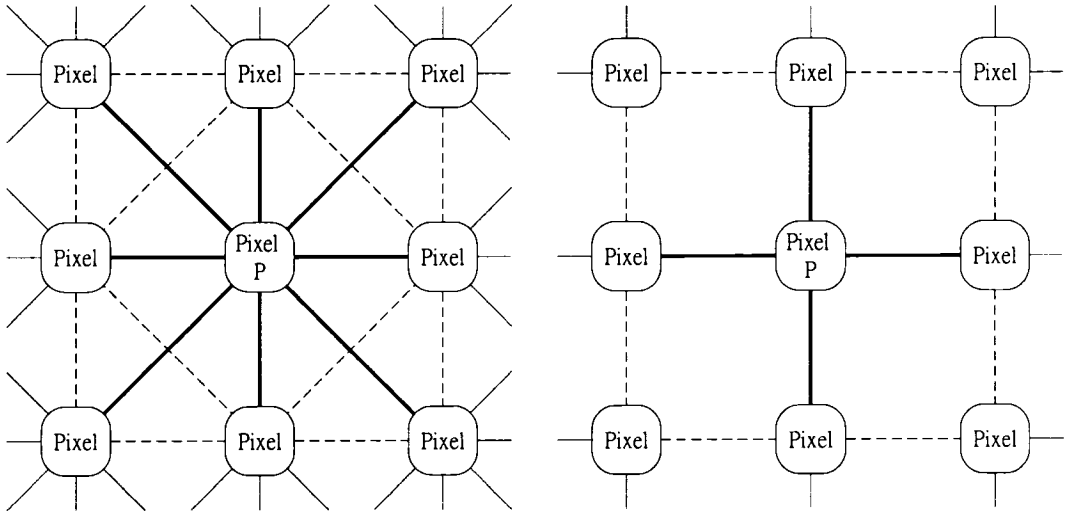
As mentioned previously in Chapter 2, a thinning process is to detect the edges of the regions where two adjacent pixels have different values, and usually the number of 0-1 (or 1-0) changes in a window of 3x3 pixels and that of logic-0 (or logic-1) adjacent pixels are counted. These numbers are then compared with given threshold values to determine if the deletion should be done. Some additional conditions may be added to improve the results of the thinning process. Such a process requires both arithmetic and

logic operations, which may result in certain difficulties in the implementation in silicon, such as poor accuracy of computation and need of large area.

The principle of the proposed thinning process is based on that reported in [6] and [9]. It involves the examinations of the pixel values and the conditional deletions to grow the background regions, i.e. shrinking the ridge regions. In the process of developing this new algorithm, only simple logic operations are considered. A detection of the difference between two adjacent pixels is performed with a 2-variable XOR operation. The results of such XOR operations are used to examine the conditions of deletion. The procedure of the examination contains only simple logic operations without counting the two numbers mentioned in the preceding paragraph. Such a logic operation should be expressed in a simple form of sum of product, or product of sums, and each term in the expression should contain only two variables. In this case, the logic operation can be implemented in such a way:

1) Each term can be performed in a simple 2-input gate.
2) If the signals X_1, X_2, ... and X_n are inputted in series, all the terms can be calculated using the same gate.

It should be mentioned that the way of providing signal variable for computations in the algorithm is related to the structure of the interconnections of the pixel array circuit to be implemented. Minimizing the number of interconnections per pixel can minimize the area per pixel. Thus, four connections per pixel, instead of eight, as shown in Fig.3-1, are desirable. With an appropriate clock timing and serial inputs, the logic operation can be made implementable with a minimum number of two-input gates.



(a) Eight connections per pixel (b) Four connections per pixel

Fig.3-1 Structure of the interconnections of a pixel array circuit.

3.2 XOR Operations — Preparation for the Detection

In the proposed process, like most of the existing ones, a 3×3 operation window, as shown in Fig.3-2, is used. The examination of a pixel at the edge of ridge regions starts with XOR operations performed with the signals of its adjacent neighbors, naming $P_1, P_2, \ldots P_7$ and P_8. These XOR operations are illustrated in Fig.3-3, where $X_1, X_2, \ldots X_{11}$ and X_{12} are the signals resulting from these operations. It should be noted that, in this procedure, the calculation for examining the conditions of deletion is based on signals X_1, $X_2, \ldots X_{11}$ and X_{12}, each of them results from the difference between two adjacent pixels. A center pixel P, as shown in Fig.3-1 and Fig.3-2, has no direct connections to its Northeast (NE), Northwest (NW), Southwest (SW) and Southeast (SE) neighboring pixels, but it can get the signals of difference, such as $P_8 \oplus P_1$ and $P_4 \oplus P_5$, from its North (N), South (S), East (E) and West (W) neighboring ones. Thus the pixel has the access to the information of all its eight adjacent pixels. Hence, the exclusion of the

direct diagonal inter-pixel connections does not affect the quality of the processing. Also as all the XOR operations are performed between two adjacent pixels regularly, it implies that by sharing the XOR results among neighboring pixels, only a few XOR operations are needed to be performed in each pixel in the implementation of array circuits.

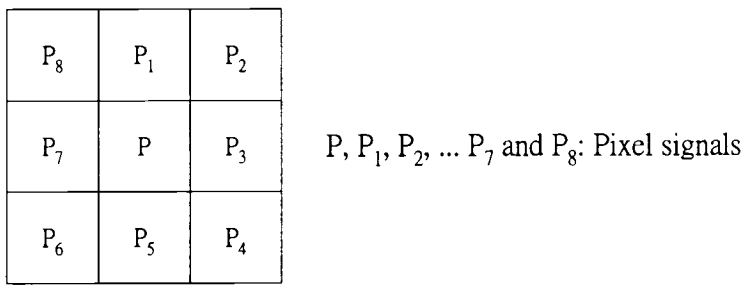


Fig.3-2 Operation window of 3×3 pixels.

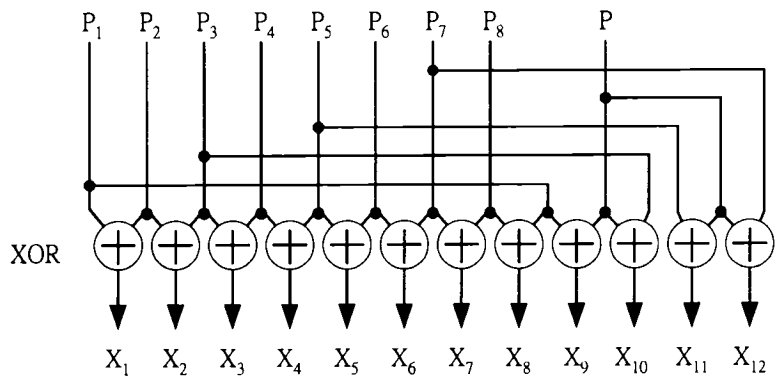


Fig.3-3 XOR operations for comparisons. Each of the signals X_1, X_2, ... X_{11} and X_{12}, results form the comparison between two adjacent pixels.

In the circuit implementation, the two-input XOR gate can be made easily using PTL. Moreover, if the original signal at each pixel is acquired simultaneously, the generation of X_1, X_2, ... X_{11} and X_{12} can be done simultaneously. It should be mentioned

that each pixel needs 12 X_i (i =1, 2, ... 12) to determine the final results in case of parallel calculation of X_i in pixels, but each of them can just perform two XOR operations by means of sharing the other ten X_i from neighboring pixels.

3.3 Conditions of Deletion

As described previously, the comparisons of adjacent pixels are done by means of XOR operations. The signals X_1, X_2, ... X_{11} and X_{12} are the results of the comparisons, as illustrated in Fig.3-4.

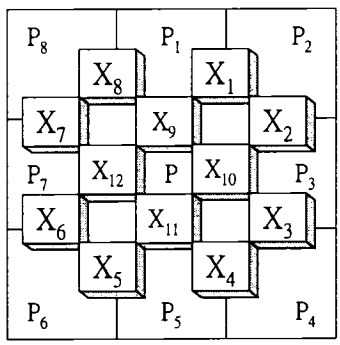


Fig.3-4 Pixel signals P, P_1, P_2, ... P_7 and P_8, and the comparison results X_1, X_2, ... X_{11} and X_{12} in a 3×3 window. There are $X_1 = P_1 \oplus P_2$, $X_2 = P_2 \oplus P_3$, ... $X_{11} = P \oplus P_5$ and $X_{12} = P \oplus P_7$.

It is desirable to have the conditions of deletion expressed in logic functions involving 2-variable terms, as mentioned in §3.1. This will lead to an easy implementation of pixel circuits using 2-input gates.

In the following parts, the logic functions for the conditions of deletion are identified.

3.3.1 Consecutive Discontinuities — Condition to Detect Pixels in a Skeleton

The prime task in this thinning process is to delete the pixels located at the edges of the "ridge" regions while preserving the pixels at the skeletons. With the comparison results generated by the XOR operations, it is important to examine the patterns of the discontinuities of the values of the eight pixels surrounding the center pixel in a 3×3 window.

If a pixel is located at the edge of a region, such as the center pixel shown in Fig.3-5, the pixel value is likely to be deleted. In such a case, $X_3 = P_3 \oplus P_4 = 1$ and $X_6 = P_6 \oplus P_7 = 1$ are considered two discontinuities in the eight surrounding pixels of the 3x3 window. However, these two discontinuities do not appear in a consecutive way.

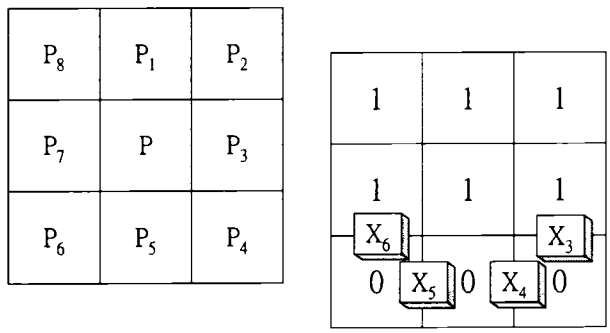
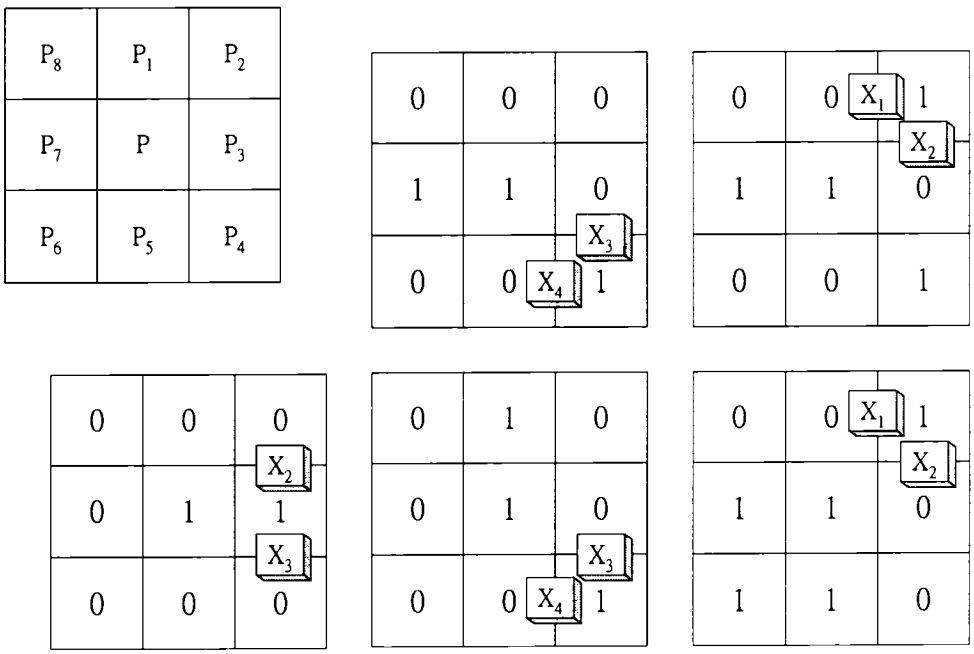


Fig.3-5 Pattern of edge in a 3×3 window. There are $X_3 = 1$, $X_4 = 0$, $X_5 = 0$ and $X_6 = 1$. Two discontinuities are found with $X_3 = X_6 = 1$, but $X_4 = 0$ and $X_5 = 0$.

In Fig.3-6, several patterns are shown in 3x3 windows. In any of these patterns, the center pixel is located at the middle or the end of a skeleton. Observing these patterns of skeletons, it can be found that each of them satisfies the condition

$X_i \cdot X_{i+1} = 1$. For example, in the window shown in Fig.3-6(a), $X_2 \cdot X_3 = (P_2 \oplus P_3) \cdot (P_3 \oplus P_4) = 1$. As $X_i = 1$ indicates a discontinuity of pixel value, $X_i \cdot X_{i+1} = 1$ indicates two discontinuities in a consecutive manner, and they are called, in this thesis, ***consecutive discontinuities***. Thus, the detection of the consecutive discontinuities is an important stage in the thinning process.



(a) The center pixel is at the end of a skeleton.

(b) The center pixel is at the middle of a skeleton.

Fig.3-6 Patterns of skeletons. The center pixel in each of the windows should not be deleted. In each of the cases, there is $X_i \cdot X_{i+1} = 1$.

Based on the analysis described above, the logic expression

$$F_a = (X_1 \cdot X_2) + (X_2 \cdot X_3) + ...(X_7 \cdot X_8) + (X_8 \cdot X_1)$$

is designed to detect the consecutive discontinuities, which is used to identify pixels of

skeleton. That is, F_a = 1 corresponds to the case of skeleton pixels, while F_a = 0 is the prime condition of an eventual deletion. It must be mentioned that the condition F_a = 1 is sufficient, not necessary, to detect a pixel at a skeleton. Thus, additional conditions are needed to make the decision of deletion.

3.3.2 Additional Conditions

As the condition of the consecutive discontinuities to detect the pixels of a skeleton is sufficient, not necessary, it is still possible that the center pixel is part of a skeleton, but the condition F_a = 0 is satisfied. Some of such pixel patterns are shown in Fig.3-7(a), where the center pixel is likely to be part of the skeleton. Also, in another case shown in Fig.3-7(b), the condition F_a = 0 is satisfied. However, all the pixels, P_1, P_2... P_8 and P, being 1, indicates that the center pixel P is not at the edge of its region. As this thinning process is, in fact, based on background region growing, the pixel value, in such a case, should remain unchanged.

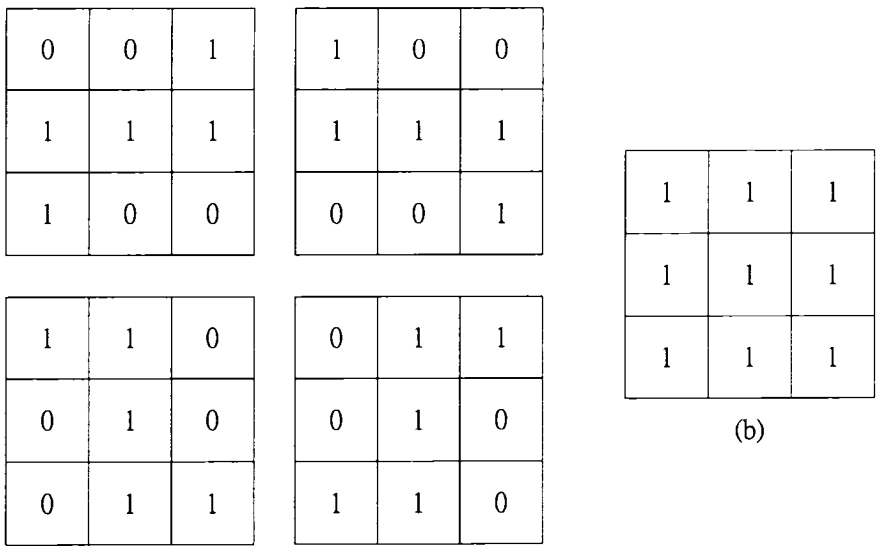


Fig.3-7 Non-deletion cases that satisfy $F_a = 0$. In these cases, the center pixel is either part of the skeleton or located inside of its region. No deletion should be done.

The pixel patterns shown in Fig.3-7 satisfy the condition $F_a = 0$, but should not be deleted. Logic function $F_b = (X_9 \oplus X_{11}) + (X_{10} \oplus X_{12})$ is used to detect the cases mentioned above. The combination of conditions $F_a = 0$ and $F_b = 1$ is sufficient and necessary to detect the pixels of the edges that should be deleted. Fig.3-8 illustrates the result produced by the processing with conditions $F_a = 0$ and $F_b = 1$, where '@' represents a pixel having a value of logic-1, and '.' represents logic-0.

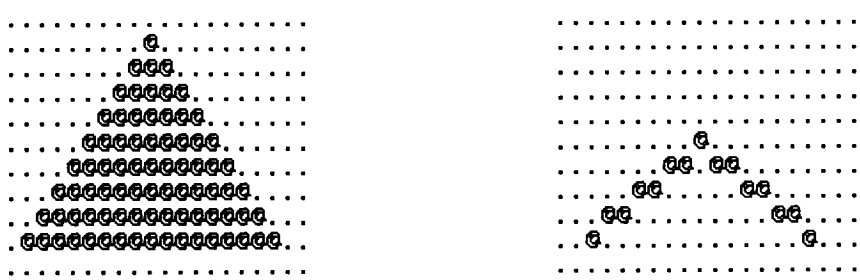


Fig.3-8 Result produced by the processing with conditions $F_a = 0$ and $F_b = 1$.

It should be mentioned that some over-erosion problems caused by simultaneous deletion also need to be solved in the study of the algorithm. The conditions of deletion are checked in each pixel of the array at the same time and simultaneous deletions can take place. As the conditions of deletion are performed in parallel in each pixel, all the pixels satisfying conditions F_a = 0 and F_b = 1 will be deleted simultaneously, and the skeleton, the result after the thinning process, is anticipated to lie on the medial axis of the original region, as shown in Fig.3-8(b). In case of a region with an even-number-pixel width, such as that shown in Fig.3-9, each of the circled pixels satisfies the conditions F_a = 0 and F_b = 1 and will be deleted simultaneously in its window. In such a case, all the pixels in this region could be deleted to zero, which lead to an over-erosion of the region. Thus, the parallel processing in the pixel array needs to be controlled so that the deletion in pixels, i.e. the shrinking of regions, can be effective alternatively on the two sides of each region.[9]

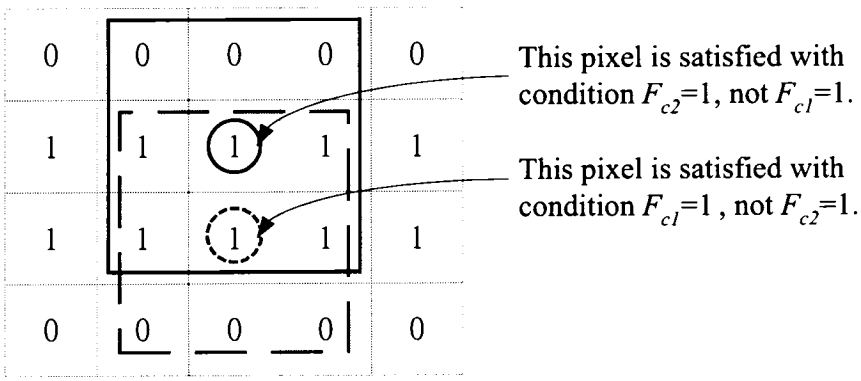


Fig.3-9 Non-deletion case for conditions F_{c1} and F_{c2}.

To implement the conditional deletion in a manner described above, two logic functions $F_{c1} = X_{10} + X_{11} + (X_9 \cdot X_{12})$ and $F_{c2} = X_9 + X_{12} + (X_{10} \cdot X_{11})$ are defined for

the additional conditions on top of those with F_a and F_b. Pixels located in lower-right or upper-left sides of a region satisfy the conditions $F_{c1}=1$ or $F_{c2}=1$ respectively. In cycles of process, the two conditions should be checked alternatively so that the deletion will not effect in both side of a region. In this case, only one of the circled pixels in Fig.3-9 will be deleted in an operation cycle, and the other will be treated as skeleton pixel in the following cycle and thus be preserved.

It should be mentioned again that prime condition F_a preserves most of the pixels at the skeletons by the detection of consecutive discontinuities. However, it is still possible that the condition F_a is satisfied but the pixels are part of the skeleton and should not be deleted. Condition F_b is added to detect these cases. Conditions F_{c1} and F_{c2} are added to prevent over-deletion of the skeleton pixels of rectangular patterns in parallel processing.

3.4 Procedure for the Thinning

Summarizing the discussions presented in §3.3, we can conclude that this thinning process starts with XOR operations to determine the difference of two adjacent pixels. The results of XORs, X_1, X_2, ... X_{11} and X_{12}, as indicated in Fig.3-3, are the variables used for the examination of the conditions of deletion. These conditions can be interpreted as the following logic expressions.

$$F_a = (X_1 \cdot X_2) + (X_2 \cdot X_3) + ...(X_7 \cdot X_8) + (X_8 \cdot X_1) = 0 \quad (1)$$

$$F_b = (X_9 \oplus X_{11}) + (X_{10} \oplus X_{12}) = 1 \quad (2)$$

$$F_{c1} = X_{10} + X_{11} + (X_9 \cdot X_{12}) = 1 \quad (3)$$

$$F_{c2} = X_9 + X_{12} + (X_{10} \cdot X_{11}) = 1 \quad (4)$$

where $X_1 = P_1 \oplus P_2$, $X_2 = P_2 \oplus P_3$, ... $X_8 = P_8 \oplus P_1$, $X_9 = P \oplus P_1$, $X_{10} = P \oplus P_3$, $X_{11} = P \oplus P_5$ and $X_{12} = P \oplus P_7$.

The proposed process is composed of iterations of simple computations. One iteration, i.e. one cycle of operation, consists of two sub-iterations. In the first sub-iteration, conditions $F_a = 0$, $F_b = 1$ and $F_{c1} = 1$ are examined. If $\overline{F_a} \cdot F_b \cdot F_{c1} = 1$, the center pixel will be deleted to 0. In the second sub-iteration, the comparisons, i.e. XOR operations, will be redone. The condition for the center pixel P to be deleted to 0 is $\overline{F_a} \cdot F_b \cdot F_{c2} = 1$. The flow chart of the proposed thinning process is shown in Fig.3-10. To complete the thinning process of an image, several iterations may be needed to obtain the thinnest skeleton in the image.

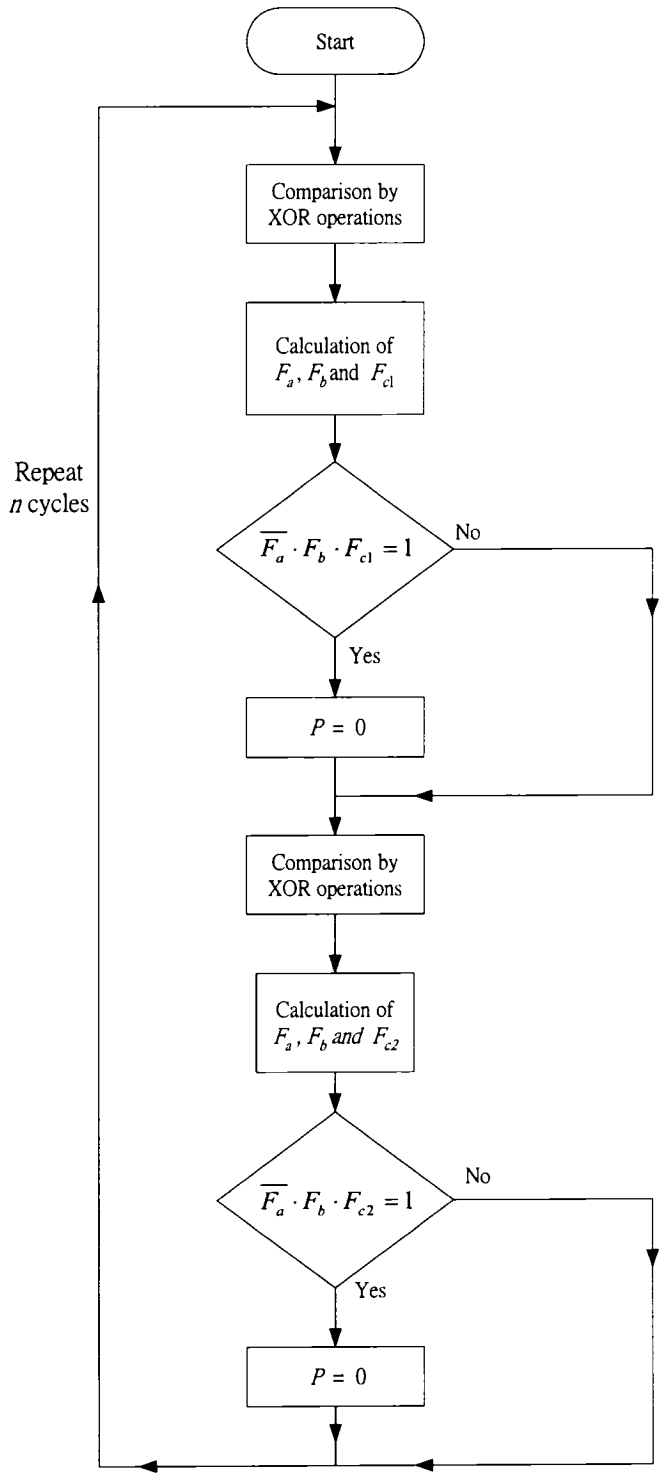


Fig.3-10 Procedure of the proposed thinning process. The number of cycles, n, is related to the thickness of the thickest region of the original binary image.

3.5 Reformulated Scheme

In order to simplify the data transfer in the implementation of pixel array circuits, the signal inputs for the logic calculations need to be carefully examined. It should be noted that eight pairs of variables (X_1, X_2), (X_2, X_3), (X_3, X_4), (X_4, X_5), (X_5, X_6), (X_6, X_7), (X_7, X_8) and (X_8, X_1), are needed to perform the operations specified by logic expression (1), i.e. verifying the condition

$$F_a = (X_1 \cdot X_2) + (X_2 \cdot X_3) + ...(X_7 \cdot X_8) + (X_8 \cdot X_1) = 0 .$$

The input of those eight pairs of signals may require complex structure that cannot be allocated in a small pixel circuit. Therefore, the logic functions having fewer pairs of input combinations are preferable for a simpler implementation. To the end, as shown in Fig.3-11, the condition defined by the logic function F_a is replaced by the two conditions

$$F_d = (X_2 \cdot X_3) + (X_4 \cdot X_5) + (X_6 \cdot X_7) + (X_8 \cdot X_1) = 0 \quad \text{and}$$

$$F_e = (X_2 + X_3) \cdot (X_4 + X_5) \cdot (X_6 + X_7) \cdot (X_8 + X_1) = 0 ,$$

which required only the half number of input-pairs of F_a. This replacement means that expression $\overline{F_d} \cdot \overline{F_e} \cdot F_b \cdot F_{c1} = 1$ in the first sub-iteration, or $\overline{F_d} \cdot \overline{F_e} \cdot F_b \cdot F_{c2} = 1$ in the second sub-iteration, is used to examine the conditions of deletion without involving F_a.

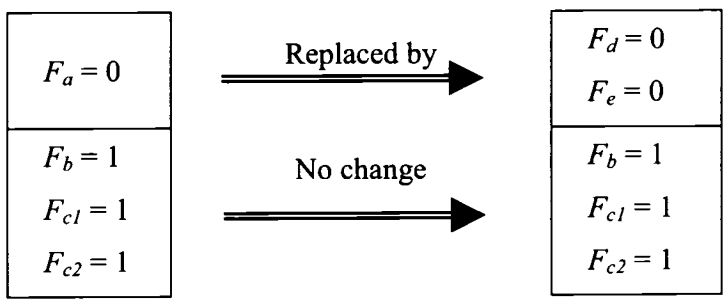


Fig.3-11 Replacement of $F_a = 0$ by $F_d + F_e = 0$ to minimize the number of time unit for the execution of the deletion conditions in the circuit implementation.

As mentioned in §3.3.1, condition $F_a = 1$ is sufficient, not necessary, to detect a pixel at a skeleton. Function F_d that consists only partial terms of function F_a, so condition $F_d = 1$ will be more sufficient to detect a skeleton pixel. Thus, additional conditions are needed to eliminate over-erosions.

Compared to the logic function F_a, the logic function F_d has four terms, i.e. $(X_2 \cdot X_3)$, $(X_4 \cdot X_5)$, $(X_6 \cdot X_7)$ and $(X_8 \cdot X_1)$, in common. These terms are used to detect the consecutive discontinuities among the pixels P_1, P_2, ... P_7 and P_8, in the 3×3 window. As the terms $(X_1 \cdot X_2)$, $(X_3 \cdot X_4)$, $(X_5 \cdot X_6)$ and $(X_7 \cdot X_8)$ are missing in F_d, this detection of the consecutive discontinuities is only partial. For instance, in the cases illustrated in Fig.3-12, at least one pixel located in the northeast corner of the 3×3 window has different logic value from its two neighbors, which can be identified by F_a, but not F_d, as the term $(X_1 \cdot X_2)$ is missing in function F_d. Consecutive discontinuities are presented in these cases but not detected by using F_d, which could lead to a deletion, when the center pixel is likely to be part of a skeleton.

Logic expression F_e is used to identified the non-deletion patterns that can be detected by F_a but not F_d. Most of these non-deletion patterns can be detected by $F_e = 1$. However, there are still some of them that cannot be detected by using only the condition $F_d + F_e = 1$, such as the cases shown in Fig.3-12(b). Additional conditions to detect these cases are needed.

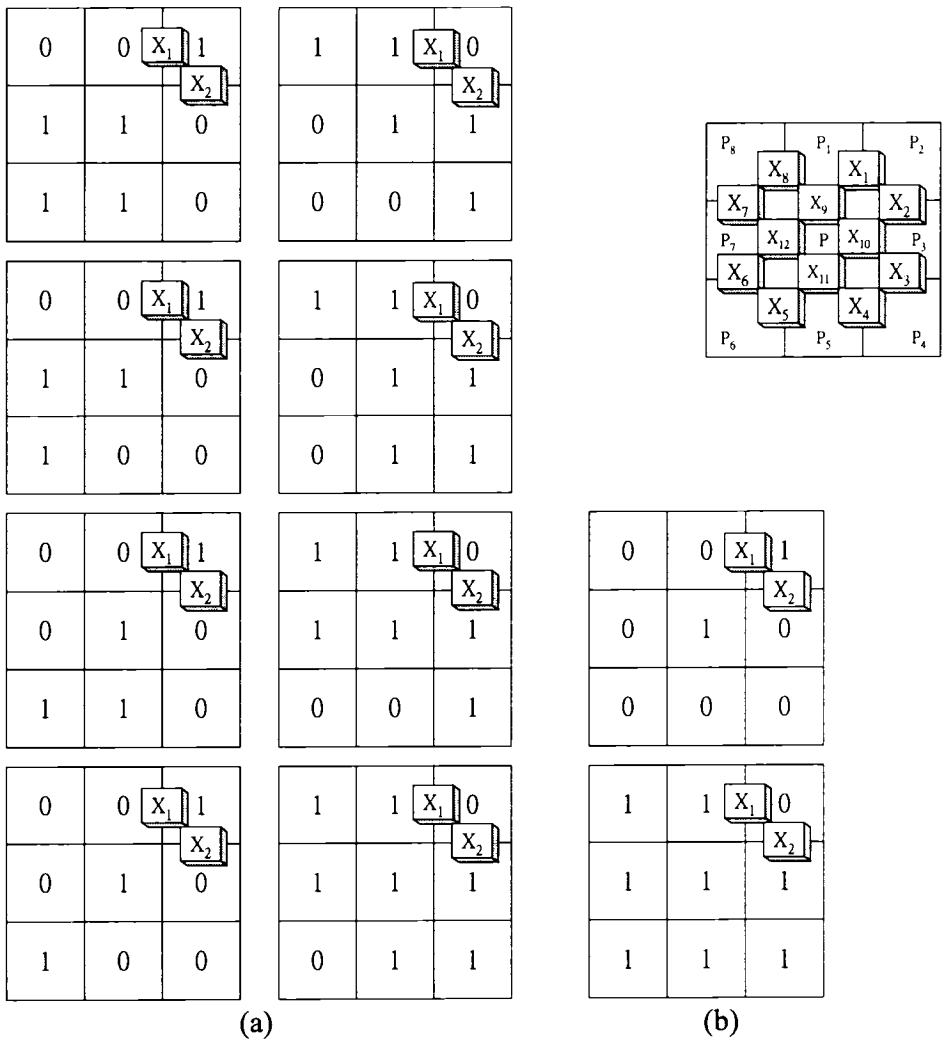


Fig.3-12 Non-deletion cases that can be identified with F_a but not F_d. Each of the cases satisfies F_d = 0, which does not give the indication of consecutive discontinuities in the window. However, the consecutive discontinuities in these cases are found in the northeast corner of the 3x3 window, which can be detected by function F_a with the term $(X_1 \cdot X_2)$. The non-deletion cases with the consecutive discontinuities in the northwest, southwest and southeast corners of the 3×3 window are not shown in this figure. Function $F_e = (X_2 + X_3) \cdot (X_4 + X_5) \cdot (X_6 + X_7) \cdot (X_8 + X_1)$ is designed to detect the cases in (a). The cases in (b) are not detected by function *Fe*, but are detected by function $F_b = (X_9 \oplus X_{11}) + (X_{10} \oplus X_{12})$.

Logic expression $F_b = (X_9 \oplus X_{11}) + (X_{10} \oplus X_{12})$ is defined in §3.3.2 to make the deletion conditions sufficient and necessary. It also can be used to identify the patterns satisfying condition $F_a = 1$, but not condition $F_d + F_e = 1$, such as the non-deletion cases shown in Fig.3-12(b). Thus, the condition for deletion is $\overline{F_d} \cdot \overline{F_e} \cdot F_b = 1$.

The equivalence of the condition $\overline{F_d} \cdot \overline{F_e} \cdot F_b = 1$ to $\overline{F_a} \cdot F_b = 1$ has been proved by comparison of the patterns that satisfy these two conditions. The results of the comparison show that all the patterns satisfying condition $\overline{F_a} \cdot F_b = 1$ also satisfy condition $\overline{F_d} \cdot \overline{F_e} \cdot F_b = 1$, and all the patterns satisfying condition $\overline{F_d} \cdot \overline{F_e} \cdot F_b = 1$ also satisfy condition $\overline{F_a} \cdot F_b = 1$. Hence, condition $\overline{F_d} \cdot \overline{F_e} \cdot F_b = 1$ can be used to replace condition $\overline{F_a} \cdot F_b = 1$, and the effectiveness of the thinning process is not affected.

By replacing F_a with F_d and F_e, only four different input-pairs, i.e. (X_2, X_3), (X_4, X_5), (X_6, X_7) and (X_8, X_1), instead of eight, are needed for the logic operations in the pixel. This replacement will simplify significantly the data transfer, and reduces the number of time units required to execute the procedure of operations in the circuit implementation.

3.6 Simulation Results

In order to verify the efficiency of the proposed thinning process, we simulated it, using C programming, with different image patterns in two steps. In the first step, the simulation is conducted iteration by iteration to show the results of each cycle of computation. One such example is illustrated in Fig.3-13. In this example, the image contains a region with horizontal and vertical edges. Some segment has a width of even-number pixels. Each iteration in the process makes the region (or segment) 2-pixel

thinner. After a sufficient number of iterations, the region is thinned to its skeleton.

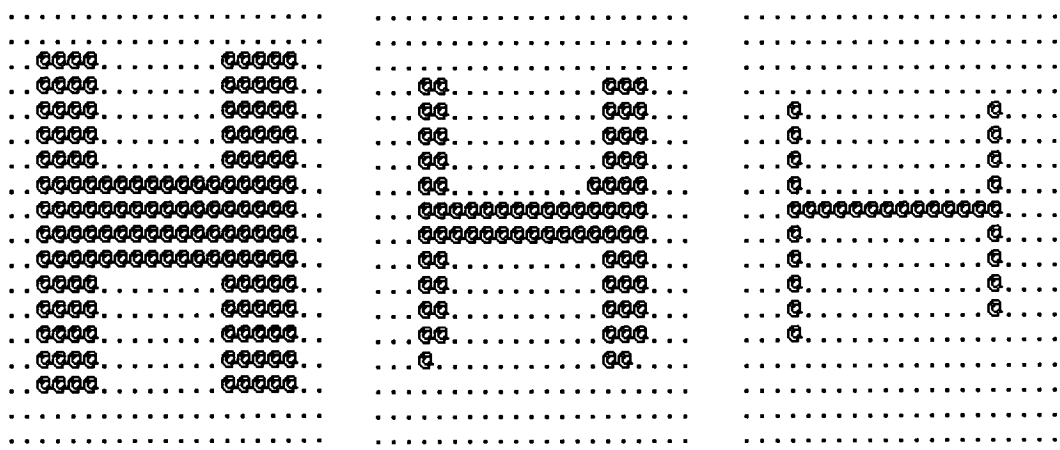


Fig.3-13 Illustration of the proposed thinning process on an iteration-by-iteration basis: (a) Original pattern, (b) Result of the first iteration, and (c) Result of the last iteration. The region is initially four pixels wide. After one iteration, the width of the region is reduced by two pixels. The last iteration results in the thinnest skeleton of the region.

In the second step, different patterns are used to test the effectiveness of the process with different kinds of regions. Fig.3-14(a) is a binary fingerprint image, and (b) is the processed image of the skeletons. Some details of the thinned ridge regions are shown in (c). Examining these images, we can find that the processed image (b) preserves the connectedness of its original image (a). Moreover, all the ending pixels in (c) have only one ridge neighbor, which meets the requirement of the minutia detection for personal identification.

The proposed thinning process can also be applied to images which have thick regions combined with fine lines. Fig.3-15(a) is such an image, and the thinned image is

shown in Fig.3-15(b). The original images of text characters and their skeleton images after the process are shown in Fig.3-16.

The simulation results provide that the proposed thinning process can be applied effectively to images of varieties of patterns, and it yields good-quality thinned ridge images, i.e. with thinnest skeletons, from different kind of regions. It should also notes that the connectedness of the regions can be well preserved in the process.

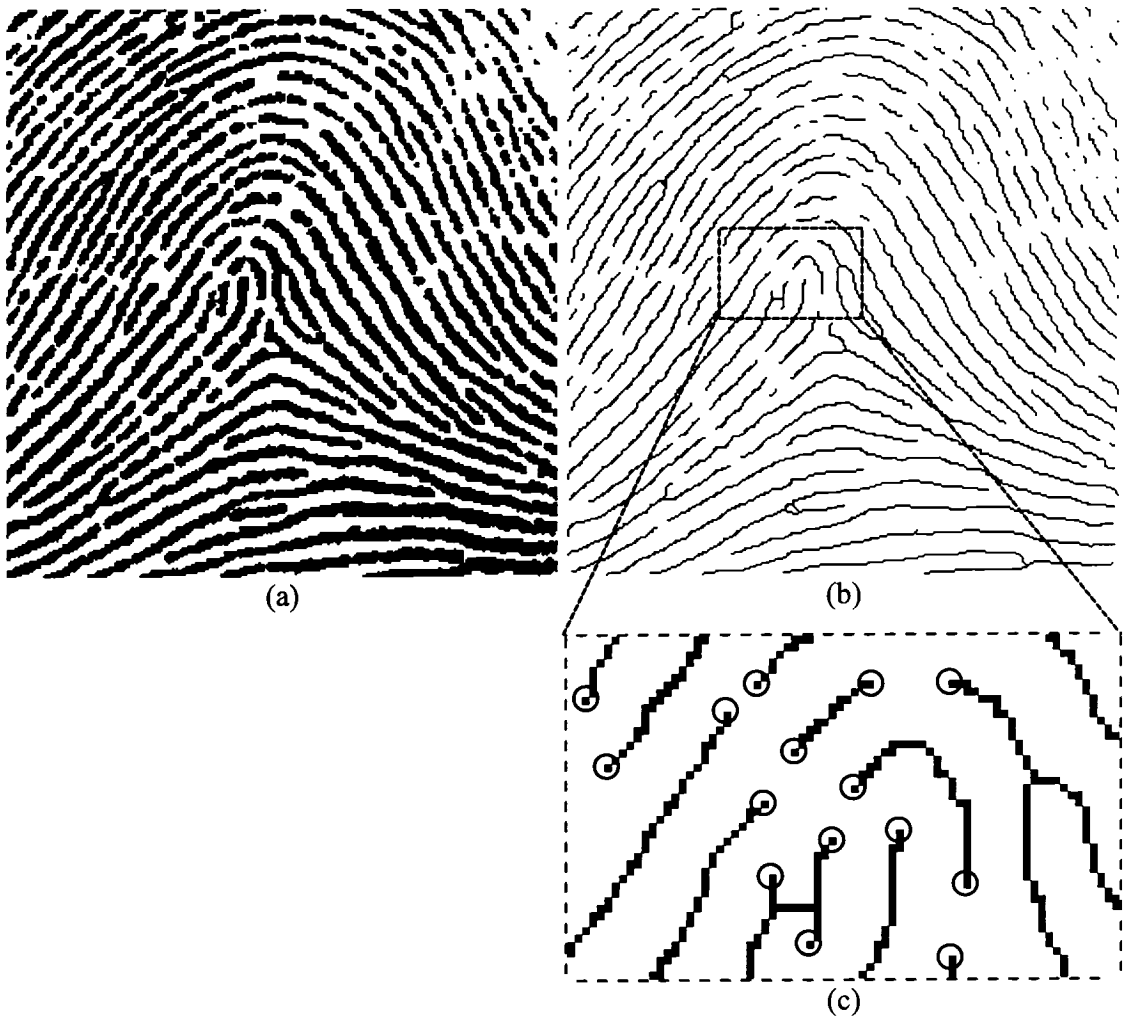
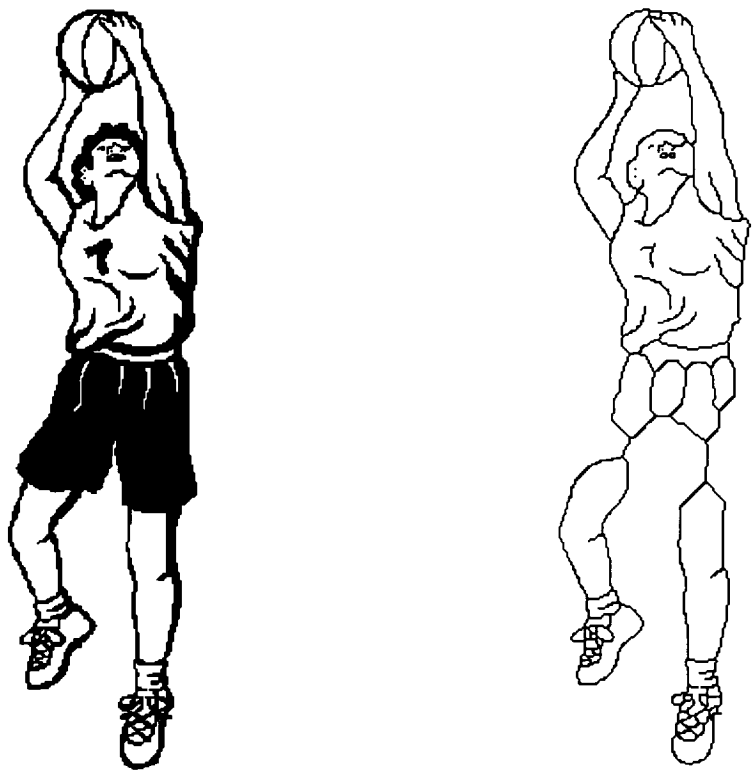


Fig.3-14 Simulation result of a fingerprint image.

(a) Binary fingerprint image.

(b) Image processed by the proposed thinning process.

(c) Enlarged image of the region framed in (b). It shows all skeletons with one-pixel-width at their ends, marked with small circles. All the ending pixels meet the requirement of the ridge ending as they have only one ridge neighbor in their 3×3 windows, which is required to facilitate the minutia detection for identification.



(a) (b)

Fig.3-15 Simulations with images having thick regions combined with fine lines.

(a) Original binary images.

(b) Skeleton image after the proposed thinning process.

(a) (b)

(c) (d)

Fig.3-16 Simulations with images of characters in text.

(a), (c) Original binary images.

(b), (d) Corresponding results after the proposed thinning process.

3.7 Conclusion

In this chapter, a thinning process suitable for pixel array implementation for parallel processing has been proposed. The data structure of the proposed thinning process is orthogonal, and the access of the data is arranged in the way that only vertical and horizontal directions are needed. Therefore, the array circuit for the process can be designed to have only four interconnections per pixel.

In the proposed process, 2-variables XOR operations are used to detect the difference between two adjacent pixels in vertical and horizontal directions. The results of XOR operations provide the information of discontinuities of pixel values. In a 3×3 window, with information provided by the eight pixels surrounding the center one, term of "consecutive discontinuities" has been defined. The "consecutive discontinuities" is used to identify pixels of a skeleton, which is an important step in the detection of the deletion conditions. Thus, the proposed method of thinning process differs from the existing ones by this detection of the consecutive discontinuities with the data obtained from the XOR operation.

Aiming at a simplification of the circuit implementation in the aspect of data transfer among the pixels, the detection of the consecutive discontinuities is made of simple logic operations. For a further simplification of the circuit to be implemented, the number of input data combinations for these logic operations is also minimized by a replacement of deletion conditions using logic functions of simpler terms. The two detections, the initial and the replacing one, yield the same results in the process. In this way, the data transfer in the circuit implementation will be simplified, as the replacement helps to reduce the number of input combinations from eight to four. As the process is to

be implemented in each pixel, such a reduction will simplify the circuit structure significantly.

The proposed thinning process has been simulated for an evaluation of its efficiency. The results demonstrated that the proposed process is efficient for images with different patterns, and the quality of the thinned regions meets the requirement of a further processing such as minutiae detection for personal-identification. A pixel array circuit implementing this proposed thinning process will be presented in Chapter 4.

4

CIRCUIT DESIGN

In this chapter, the design of a pixel array circuit implementing the thinning process proposed in Chapter 3 is presented. As mentioned in previous chapters, the complexity both in interconnections and circuits in each pixel can be a critical issue in the design and the implementation. Particular approaches for data transfer and design of computation units are needed.

The circuit structure of the pixel array is presented in §4.1. The interconnections designed for data transfer, and particular timing logic operations, are described in great detail in §4.2. The composition of the circuit in each pixel is presented in §4.3. The logic

units for the operations required to perform the thinning process are designed in such that the space needed is minimized and the process is performed in a simple manner. The procedure of the circuit operation implementing the thinning process is also elaborated in this chapter.

4.1 Structure of the Pixel Array and Composition of Each Pixel

It is desirable to have orthogonal data structure, so that the circuit is compatible with other processing algorithms. The pixels in the array circuit for the proposed thinning process are designed to be placed orthogonally. The simplified diagram of the structure of the proposed circuit is shown in Fig.4-1.

An XOR operation is used to detect the discontinuity between two adjacent pixels, as mentioned previously, and the results of such XOR operations in a 3×3 window are used for the detection of the consecutive discontinuities in order to make the deletion decision. In the pixel array, each pixel has connections to four XOR gates. The results of twelve XOR operations, i.e. X_1, X_2, ...X_{11} and X_{12}, can be produced simultaneously with such direct connections.

In each pixel, there are an optical acquisition unit, a switch unit, two 2-input XOR gates and other logic units, as shown in Fig.4-2. The optical acquisition unit is composed of a photoreceptor and a current comparator for signal sensing and binarization [28]. The switch unit is to make the connections controllable and bi-directional. The details of the design of the pixel unit are presented in the following sub-chapters.

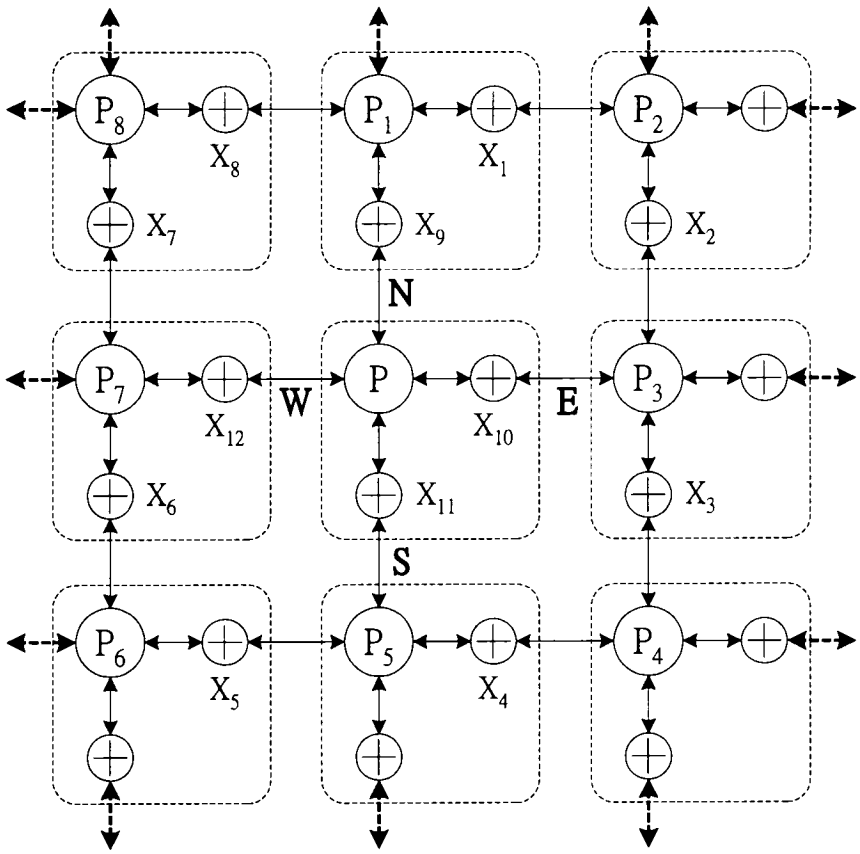


Fig.4-1 Simplified diagram of the structure of the pixel array for image acquisition and thinning process. Each pixel is indicated by a frame. It involves two XOR gates and other modules. More details of the pixel are shown in Fig.4-2 and Fig.4-4. Each of the inter-pixel connections, such as the one indicated with *N*, is bi-directional and used for different data transfers between the center pixel and one of its adjacent pixels, such as that located in its north.

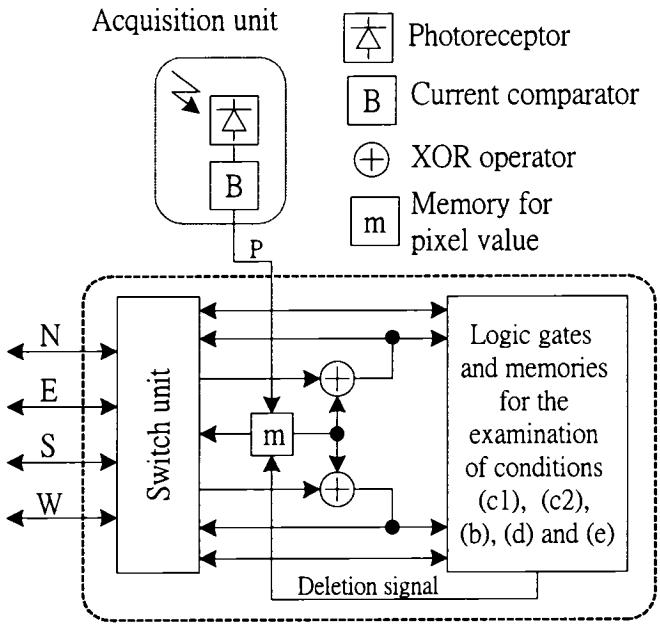


Fig.4-2 Pixel structure. Each pixel comprises a photoreceptor, a current comparator to convert a current into a binary signal, a switch unit, logic gates and memories. The switch unit activates selectively the signal inputs and outputs of the pixel. More details of the pixel circuit are given in Fig.4-4.

4.2 Data Transfer

To perform the logic operations for the examination of deletion conditions with the restraint of only four interconnections per pixel, the way of the data transfer of the array circuit is very important. The four connections *N*, *E*, *S* and *W* from the center pixel, to the four neighboring pixels are bi-directional, and each connection, such as the one indicated *N* in Fig.4-1, is used to transfer three kinds of signals.

1) The acquired signal to an XOR gate located in an adjacent pixel. For example, the signal acquired at the center pixel is sent to its northern adjacent pixel as one of the two input signals to perform the XOR operation and to produce the

signal X_9.

2) The result of the XOR operation from an adjacent pixel. For instance, connection N is also used to send X_9 back to the center pixel.
3) The signals resulting from other logic operations from an adjacent pixel. For example, the center pixel can get the result of $(X_1 \cdot X_8)$ from its northern pixel.

The data transfer can be arranged by means of a control sequence applied to the switch unit. Therefore, the data transfer sequence can be programmed to optimize the operation process. It should be noted that the interconnections are used not only for the initially acquired signals, but also the intermediate results. For this thinning process, the switch unit of a pixel selects two connections for the input signals from connected adjacent pixels, and the other two for the output signals simultaneously during every operation time unit.

In the beginning of the process, the XOR operations are performed simultaneously in all pixels. The signal acquired at the pixel is applied to each of the two local 2-input XOR gates, as shown in Fig.4-1, and it is also applied, via the connections, to two of the four adjacent pixels. At the same time, these two XOR gates receive two signals acquired at the other two adjacent pixels respectively. Therefore, in each pixel, two XOR operations are performed simultaneously, and all the pixels do so at the same time. The comparison results X_1, X_2, ...X_{11} and X_{12} of the XOR operations are produced during this operation time unit.

Because the XOR operations described above are performed in parallel in all pixels in the array, each pixel will have four comparison results, two produced locally

and two obtained from neighbors via the connections. For example, the center pixel in Fig.4-1 has X_{10} and X_{11} inside it and X_9 and X_{12} from adjacent pixels. Since the four comparisons results, i.e. X_9, X_{10}, X_{11} and X_{12}, the result of the examination of conditions

$$F_b = (X_9 \oplus X_{11}) + (X_{10} \oplus X_{12}) = 1,$$

$$F_{c1} = X_{10} + X_{11} + (X_9 \cdot X_{12}) = 1, \text{ and}$$

$$F_{c2} = X_9 + X_{12} + (X_{10} \cdot X_{11}) = 1,$$

can be obtained. At the same time, the four terms $(X_9 \cdot X_{11})$, $(X_{10} \cdot X_{12})$, $(X_9 + X_{11})$ and $(X_{10} + X_{12})$ are produced and ready in the center pixel P. Two of these terms will be sent to the two vertically adjacent pixels and the other two to the two horizontally adjacent pixels. In other words, each of the four connected adjacent pixels will get two terms from the center pixel. For example, $(X_{10} \cdot X_{12})$ and $(X_{10} + X_{12})$ are to be sent from the center pixel to both its northern neighbor P_1 and southern one P_5, and $(X_9 \cdot X_{11})$ and $(X_9 + X_{11})$ to P_3 and P_7. During the same period, each pixel in the array gets the results of the same examinations of $F_b = 1$, $F_{c1} = 1$ and $F_{c2} = 1$, and has the four terms at its pixel ready to be sent to its connecting neighbors.

For the following operations, it is the data of the terms, instead of the acquired signals or the results of the XOR operations, that are transferred through the inter-pixel connections. The center pixel, like all the others, will get eight terms from its connected neighbors, i.e. $(X_8 \cdot X_1)$ and $(X_8 + X_1)$ from P_1, $(X_2 \cdot X_3)$ and $(X_2 + X_3)$ from P_3, $(X_4 \cdot X_5)$ and $(X_4 + X_5)$ from P_5, and $(X_6 \cdot X_7)$ and $(X_6 + X_7)$ from P_7. Therefore, the terms for the conditions

$$F_d = (X_2 \cdot X_3) + (X_4 \cdot X_5) + (X_6 \cdot X_7) + (X_8 \cdot X_1) = 0 \text{ and}$$

$$F_e = (X_2 + X_3)\cdot(X_4 + X_5)\cdot(X_6 + X_7)\cdot(X_8 + X_1) = 0$$

in the center pixels P are ready for the examinations. Also, each pixel of the array receives its eight terms for the examinations of $F_d = 0$ and $F_e = 0$. In this way, the intermediate results obtained in each pixel are shared by the others in the neighborhood, which reduces significantly the amount of computation in each pixel. Furthermore, it also makes the data transfer among the pixels more efficient.

The operations in the thinning process are performed with the signals acquired and processed in the pixels of a determined neighborhood. The data transfer among the pixels is a very important issue, and it is related to the timing of the system and the operations in each pixel. In the proposed circuit, the nature of identical operations needed for determining the result in each pixel in the array is fully exploited, and the interconnections among the pixels are designed to transfer not only the acquired signals but also the intermediate results, so that they are shared by their neighbors. In this way, the operation units in each pixel can be simplified, the number of cycles required to complete the procedure minimized, and the traffic of data transfer reduced. Also the pixel circuit is designed to suit the way of data transfer, which is presented in the following sub-chapter.

4.3 Logic Operations in each Pixel

As mentioned in Chapter 3, the principle of the proposed thinning process is based on the "growth" of the background regions, of which the pixels have a value of logic-0. If a pixel with a logic-1 at a ridge region satisfies the deletion conditions, its value will be deleted, i.e. a reset of the pixel output in the circuit. The thinning process in each

pixel is, in fact, a process of a reset, or a non-reset, of the pixel signal according to the result of the logic operations. If the pixel signal is stored in a 1-bit memory, the logic operations are performed to enable a conditional reset of the memory.

The basic point of the design of the proposed pixel array is to construct a memory-based circuit, of which a simplified scheme is shown in Fig.4-3. A latch can be used as the storage device, and a Pull-Down Network (PDN) is connected as its reset circuitry. The PDN is controlled by the signals that are the results of the examinations of the deletion conditions. When the current path of the PDN is turned on by the signals, the deletion, i.e. a reset, is made effective.

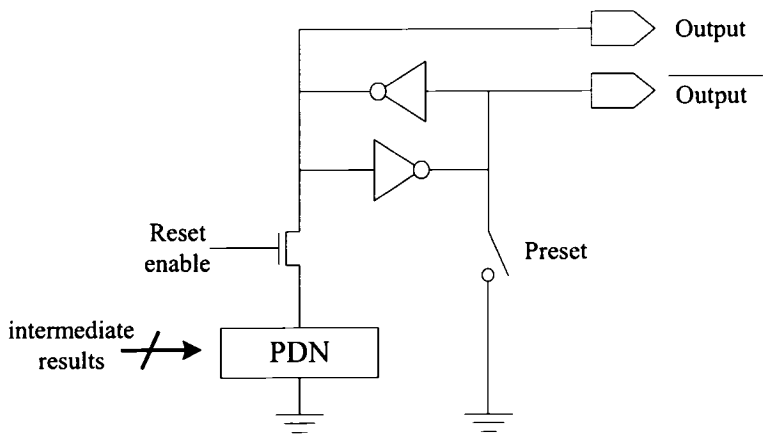


Fig.4-3 Structure of a memory cell with a conditional reset. The conditional reset is controlled by the PDN of which state depends on the intermediate results of the examination. A *Reset_enable* signal is used for the reset to be synchronized with the timing of the switch unit.

Because each pixel performs only two XOR operations each time, and the examination of the deletion conditions needs the results of twelve XOR operations, a

number of 1-bit memories are needed. In this design, seven latches, as shown in Fig.4-4, are used in each pixel for three kinds of functions.

1) The latch m is the main memory cell and is used to store the initial binary signal at the beginning of the process, and then the result of a sub-iteration or the complete process.
2) The latches m_1, m_2, m_3 and m_4, are used to store X_9, X_{10}, X_{11} and X_{12}, the signals resulting from four out of the twelve XOR operations. These signals are used repeatedly for logic functions F_b, F_{c1} and F_{c2}, at this pixel, and F_d and F_e in its neighboring pixels.
3) The signals of the third kind are those transferred from the adjacent neighbors, such as $(X_2 \cdot X_3)$, $(X_4 \cdot X_5)$, $(X_6 \cdot X_7)$ and $(X_8 \cdot X_1)$. They are used to examine the conditions of $F_d = 1$ and $F_e = 1$.

As mentioned, the latch m is loaded with the initial pixel signal, whereas all the other latches are preset to logic-1 at the beginning of each sub-iteration. Each pixel receives, at the beginning of the detection process, two signals from two of its four connected neighbors respectively, and sends its own acquired signal to the other two neighbors. Then the results of two XOR operations are produced in each pixel. In the following step, two of the four connections of each pixel are used to output these two XOR results to two of the four neighbors, and the other two connections to input two XOR results from the other two neighbors. With the results of the four XOR operations, two produced locally and two received, logic functions such as F_b, F_{c1} and F_{c2}, can be performed, and partial results for other functions can be produced. Then, the partial results are transferred and effectively used in the pixel and the neighboring pixels.

Each of the seven latches has a reset circuitry of PDN. By which, certain logic functions are integrated in the latches for an eventual "pull-down" of the output voltage. The PDNs are controlled by the signals that are results of early-stage logic operations performed in NMOS PTL gates. To drive NMOS transistors in the PDNs of the latches, as one shown in Fig.4-3, the high voltage level of the signals does not have to be perfect. Thus, the signals from the NMOS PTL gates can be applied directly to the PDNs without needs for level compensation. The use of NMOS PDN and PTL units helps to simplify the circuit and improve the performance. In the following sub-chapters, the structure of the pixel circuit and its operation procedure will be described.

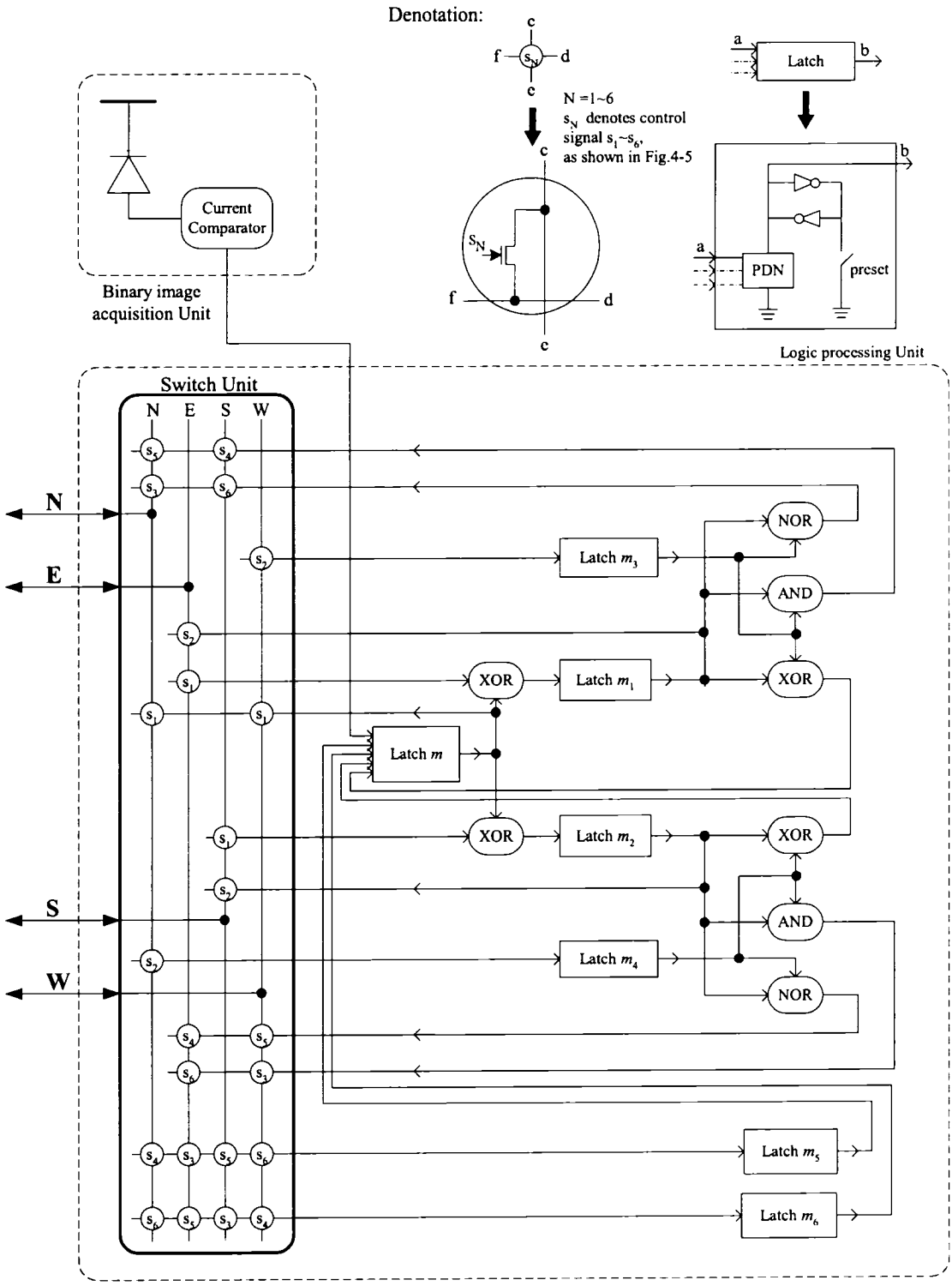


Fig.4-4 Structure of the pixel circuit.

4.4 Structure of a Pixel and Operation Procedure

The diagram of a complete structure of the pixel circuit is shown in Fig.4-4. This circuit comprises an acquisition unit, a switch unit, eight PTL gates and seven latches. The acquisition unit receives incident signal and converts it into a binary voltage signal that is then provided to the latch m. The output of this latch will be updated every sub-iteration according to the result of the examination of the deletion conditions.

In the circuit shown in Fig.4-4, there are four connections, namely N, E, S and W, to the neighboring pixels. The input and output of the signals are controlled by a signal sequence of 6 bits, applied to the switch unit. The 6-bit signal consists of non-overlap pulses comprising s_1, s_2, ...s_5 and s_6. During each time unit, one in six connections located in each of the four vertical lines is activated, and thus each of the four connections, N, E, S and W, is used for signal input or output. The timing of the operation procedure is determined by the control signal sequence and the switch unit.

As shown in Fig.4-4, the logic gates employed in the circuit are 2-input units. The inputs of these gates are provided by the latches in the pixel or from the connected neighbors. The destinations of the output signals of the logic operations in the PTL gates are the PDNs of the latches inside the pixel, or those in the neighbors, so that partial results are distributed in the neighborhood.

The signal transfer among the adjacent pixels and the logic operations for the calculation of the deletion conditions are performed with the constraints of four connections to each pixel. The timing of the operation procedure is controlled by the signals s_1 to s_6 and additional signals. One iteration in the thinning process consists of two sub-iterations, and each sub-iteration in this design consists of eight time units. As

shown in Fig.4-5, the first time unit (t_0, t_1) is used to load the acquired signal in the latch *m* and to preset all the latches in the pixel. During the second time unit, every pixel produces two XOR operation results, and the results of all the XOR operations are obtained in the neighboring pixels. During the five following time units (t_2, t_7), the logic operations are performed keeping pace with the signal transfer among the connected pixels. During this period, the signals being transferred are intermediate results. During the last time unit of the sub-iteration, namely (t_7, t_8), the signal enabling the pull-down of the output voltage of the latch *m* is made active. If the signals controlling the PDN activate the connection in the PDN, the output voltage will be reset.

During one time unit, only one logic operation is executed in each pixel. The duration of such a time unit can be very short. In the following sub-chapter, the simulation results related to the minimum duration of the time unit are presented. Estimations of the required area and the power dissipation are described as well.

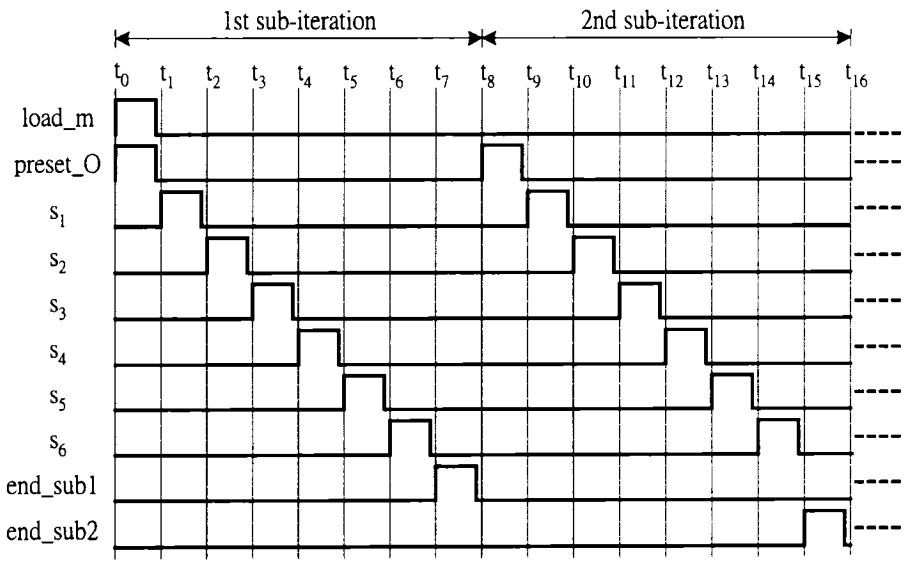


Fig.4-5 Timing of the control signals of the switches and memories in a complete iteration. Signal *load_m* sets the memory *m* at "1" state at the very beginning of the process according to the acquired signal, and *preset_O* set all the other memories at the beginning of each sub-iteration. Signal s_1~ s_6 control the switch unit. The signal *end_sub1* enables the reset of the memory *m* if the condition $\overline{F_d} \cdot \overline{F_e} \cdot F_b \cdot F_{c1} = 1$ is satisfied, whereas *end_sub2* enables the reset by the condition $\overline{F_d} \cdot \overline{F_e} \cdot F_b \cdot F_{c2} = 1$.

4.5 Performance Estimation

The performance of the proposed array circuit for the thinning process is evaluated in terms of required silicon area, operation speed and power dissipation. The structure of the pixel is the same as shown in Fig.4-4. The control sequence shown in Fig.4-5 is applied. The transistor models of a 0.18 μm CMOS technology is used for the performance estimation of the circuit.

In this design, the number of transistors per pixel, including those for the interconnection controls, is 115. As all of these transistors serve as switches, they are all

very small in size, and 100 of them are minimum-sized NMOS ones. Therefore, each pixel can be integrated in a tiny space of 50x50 μm^2 with a fill factor of at least 50%. Thus, the proposed thinning process can be implemented in a chip of pixel array with a resolution of 500 dpi, which satisfies the requirements of [29] and [30].

As mentioned previously, the time required for the operation in each pixel can be very short, and the results of the simulation show that the operations to be performed within a time unit, specified in §4.4, need only 2 ns. Taking the parasitic elements in interconnection into consideration, if the number of pixels in the array is 256x256, the duration of the time unit can be as short as 10 ns. As one sub-iteration needs eight time units and in case of 10 ns per unit, every sub-iteration takes 80 ns. In a typical case of a fingerprint image with a resolution of 500 dpi, twenty sub-iterations are needed to complete the proposed thinning process. Thus, only 80ns×20=1.6 μs are needed to perform all the required logic operations of the process in each pixel.

The circuit is very power efficient because of its low activity rate. It is composed of PTL gates and preset latches. The objective of the operations is to reset eventually the latches in each sub-iteration. In other words, during one sub-iteration consisting of eight time units, the outputs of the latches and most of the circuit nodes can have one transition of voltage level, or no transition at all. Moreover, the latch m is reset only once, at most, during the entire process of a large number, e.g. 20, of sub-iterations. Because of the low activity rate resulting form this particular mechanism of the circuit operation, the power dissipation is minimized.

In order to estimate the power dissipation of the proposed array circuit and to make an appropriate comparison with the circuit reported for the same task, a simulation

is done with the duration of the time unit of 125 μs. In this way, a complete thinning process for a typical fingerprint image with a resolution of 500 dpi needs 20 ms. Under this condition, the total power dissipation of the circuit of 256×256 pixels, including the wires of control signals, is about 2.5767×10^{-4} Watts, and it means that the power dissipation per pixel is about 3.93×10^{-9} Watts, which is about 12.5% less than that of the array circuit with analog pixels reported in [21].

4.6 Conclusion

In this chapter, a pixel array circuit for image acquisition and the thinning process described in Chapter 3 has been proposed. In this circuit, the thinning process is implemented in such a way that only four interconnections per pixel are needed, and only simple logic gates and memory cells are required in each pixel. As the nature of repetitive calculations in the thinning process, the computation results in mid-stages of the examination of deletion are shared among the pixels. In order works, the pixels do partial operations and then share the results with their neighboring pixels via the interconnections.

The interconnection between two adjacent pixels is bi-directional, and is used to transfer not only the acquired signals but also the intermediate results. By sharing the intermediate results in the neighborhood, the logic operations for each pixel is simplified, and the number of time units for the process is minimized.

As the implemented thinning process is based on a conditional reset, the circuit in each pixel is based on a scheme of memories preset to logic-1. The memories, with Pull-Down Networks (PDNs) integrated, can be reset according to the intermediate

signals produced in the pixel or from its neighbors. By implementing the logic operations in Pass-Transistor Logic (PTL) gates and the PDNs of the memories, the circuit structure is simplified. Furthermore, as the memories are preset at the beginning of every sub-iteration, and may be reset once, or none, during the whole sub-iteration, the number of the voltage transitions in this circuit is very small, which leads to an extremely low activity rate of the circuit. Thus, the proposed circuit is very power efficient.

The performance of the proposed circuit, in terms of required silicon space, operation speed and power dissipation, has also been evaluated. Each pixel consists of 115 transistors, and most of them are minimum-sized NMOS ones. The estimated area per pixel is $50\times50\ \mu m^2$ with a fill-factor of more than 50%, if $0.18\ \mu m$ CMOS technology is used. In case an array circuit of 256x256 pixels is evaluated, the duration of each time unit can be set as short as 10 ns, and the duration of a complete thinning process in each pixel for a typical fingerprint image is only $1.6\ \mu s$.

The power dissipation of the circuit is also evaluated, and compared to an existing one [21]. Based on the same processing time, the proposed circuit, consisting of digital units, dissipates less power than the one of low-power analog pixels.

With the features of small space, low power, relatively high speed, and facility of integration, the proposed circuit can have promising potential applications in different systems.

5

CONCLUSION

In this thesis, the design of a thinning process and its implementation in a pixel array circuit have been presented. The research efforts of this work are in the two aspects: the processing algorithm and the circuit design for implementing the algorithm. In the first aspect, an algorithm of an efficient thinning process, in terms of suitability for VLSI implementation, is developed. For this implementation, the design of a pixel array circuit is proposed.

5.1 Summary of the Work

As mentioned above, the work of this thesis has been conducted in two aspects. In the aspect of the algorithm, a new thinning process that suits the implementation of pixel array is developed. In the thinning process, the concept of "consecutive discontinuities" in a 3×3 window is introduced to identify pixels located at the edges of regions. With this concept, a simple procedure for the operation is proposed. Aiming at an easy implementation of the procedure in a pixel circuit, two issues have been taken into consideration in the design of this operation procedure. The first issue is that, in each pixel, the operations in each step use the signals from only four neighboring pixels instead of eight in the 3×3 window. This will result in a simple structure of inter-pixel connections in the circuit. The other issue is concerning the type of operations in the procedure. Two-variable XOR operations are included to detect the discontinuities in vertical and horizontal directions, and other logic operations are also made to be two-variable ones, which facilitate the decomposition of the computations and further reuse of the partial results.

The effectiveness of the proposed thinning process has been verified by simulation. The simulation results demonstrated that the proposed thinning process is efficient with a good preservation of connectedness of images with different patterns, and the quality of the processed images meets the requirement of a further processing such as minutia detection for personal identification.

The other aspect of this work is the design of a pixel array circuit implementing the image acquisition and the proposed thinning process. In this circuit, each pixel in the array needs only four connections to its neighbors, and requires only simple logic gates

and memory cells. The examination of the deletion conditions lasts only a few clock cycles. It is performed step by step with the pace of the signal transfer. In other words, the signal of each pixel is updated in each clock cycle while the updated signals are transferred among the pixels. Thus, from the second clock cycle on, the interconnections are used to transfer the updated results. In this way, partial results of the computation in a pixel can be shared with the connected neighboring pixels, so that each of these pixels needs to do only part of the computations for obtaining the final result. Also, it should be noted that, the signal in each pixel is updated with those of its four neighbors, and the signal is propagated in each clock cycle. Hence, in a pixel, the deletion decision can be made based on the data from the pixels directly connected to the pixel and the data of those that have no direct connections.

As the thinning process in each pixel is to generate eventually a signal of deletion, the operation of the pixel circuit is based on a conditional reset of a memory. The conditions of the reset are interpreted as logic operations implemented in NMOS Pass-Transistor Logic (PTL) gates and Pull-Down Networks (PDNs) integrated in the reset circuitry of the memory. The use of NMOS PTL gates and PDNs helps to simplify the circuit and improves the performance. Also, in this structure, there are only "pull-down" operations, so the activity rate of the circuit is very low during one cycle of the detection. Thus, the proposed circuit is very power efficient.

The performance of the circuit has been evaluated in terms of required silicon area, operation speed and power dissipation. The number of transistors per pixel used for the thinning process and the interconnection controls is 115, and most of them are simple minimum-sized NMOS ones. Using a 0.18 μm CMOS technology, it is estimated that

the size of each pixel, including the space for interconnections, is about $50\times50\ \mu m^2$ with a fill factor of at least 50%. Hspice simulations with the transistor models of a $0.18\ \mu m$ technology have been conducted to evaluate the circuit performance. The simulation results show that the circuit can be made to operate with a clock cycle of 10ns. A complete thinning process for a typical fingerprint image with a resolution of 500dpi needs about $1.6\ \mu s$. The simulation results also show that the circuit dissipates less power than that of the low-power analog pixel array circuit reported for the same task of thinning process.

The proposed array circuit can be used in image processing systems for applications in communications, security and industrial detections. In particular, because of its simple structure and easy integration with other circuitry, the proposed circuit can be used in portable systems for signal detection.

5.2 Suggestions for Future Work

Based on the work of this thesis, future work will involve a silicon implementation of the pixel array circuit for thinning process. Also, the applications of the design methods proposed in this work to other edge detection based image processing will be investigated.

In the work of the silicon implementation of the pixel array circuit, a binarization circuit in each pixel needs to be designed. Moreover, pixel-read-out circuitry needs to be improved. It may be more preferable to read out only the signals of the skeleton pixels instead of the signals of all pixels. A scheme for such a read-out needs to be proposed.

Comprehensive studies about algorithms of image feature extraction will be

conducted, and the suitability of their implementations in digital pixels will be explored. New schemes of parallel detection in pixel array circuits will be proposed.

Reference

[1] A. K. Jain, L. Hong, S. Pankanti and R. Bolle, "An Identity-Authentication System Using Fingerprints," *Proc. IEEE*, vol. 85, pp. 1365-1388, Sept. 1997.

[2] A. P. Kiraly, J. P. Helferty, E. A. Hoffman, G. McLennan and W. E. Higgins, "Three-dimensional path planning for virtual bronchoscopy," *IEEE Trans. Med. Imag.*, vol. 23, no. 11, pp. 1365-1379, Nov. 2004.

[3] M. Ito, I. Fujita, Y. Takeuchi and T. Uchida, "Pattern defect analysis and evaluation of printed circuit boards using CAD data," in *Proc. IEEE Symp. Electronic Manufacturing Technology*, 1993, pp. 7-10.

[4] S. Zhao, Z. Chi, P. Shi and Q. Wang, "Handwritten Chinese character segmentation using a two-stage approach," in *Proc. IEEE Conf. Document Analysis and Recognition*, 2001, pp. 179-183.

[5] D. D. Hwang and I. Verbauwhede, "Design of portable biometric authenticators - energy, performance, and security tradeoffs," *IEEE Trans. Consumer Electronics*, vol. 50, no. 4, pp. 1222-1231, Nov. 2004.

[6] D. Rutovitz, "Pattern recognition," *Journal of the Royal Statistical Society Series A*, vol. 129, no. 4, pp. 504-530, 1966.

[7] E. S. Deutsch, "Comments on a line thinning scheme," *British Computer Journal*, vol. 12, no.11, pp. 142, Nov. 1969.

[8] E. S. Deutsch, "Thinning algorithms on rectangular, hexagonal, and triangular arrays," *Communications of the ACM*, vol. 15, no. 9, pp. 827-837, Sept. 1972.

[9] T. Y. Zhang and C. Y. Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns," *Communications of the ACM*, vol. 27, no. 3, pp. 236-239, Mar. 1984.

[10] R. C. Gonzalez and R. E. Woods, "Skeletons," in *Digital Image Processing 2nd ed.*, Prentice Hall, pp. 650-653, Jan. 2002.

[11] H. E. Lu and P. S. P. Wang, "A comment on "A Fast Parallel Algorithm for Thinning Digital Patterns," *Communications of the ACM*, vol. 19, no. 3, pp. 239-242, Mar. 1986.

[12] P. S. P. Wang and Y. Y. Zhang, "A fast and flexible thinning algorithm," *IEEE Trans. Computers*, vol. 38, pp. 741-745, May 1989.

[13] J. S. Kwon, J. W. Gi, and E. K. Kang, "An Enhanced Thinning Algorithm Using Parallel Processing," in *Proc. IEEE Conf. Image Processing*, 2001, vol. 3, pp. 752 –755.

[14] R. Stefanelli and A. Rosenfeld, "Some parallel thinning algorithms for digital pictures," *Journal of the ACM*, vol. 18, no. 2, pp. 255-164, 1971.

[15] V. Espinosa-Duro, "Fingerprints thinning algorithm," *IEEE Aerospace and Electronic Systems Magazine*, vol. 18, pp. 28-30, Sept. 2003.

[16] C. M. Holt, A. Stewart, M. Clint, and R. H. Perrott, "An improved parallel thinning algorithm," *Communications of the ACM*, vol. 30, no. 2, pp. 156-160, 1987.

[17] Y. Y. Zhang and P. S. P. Wang, "A modified parallel thinning algorithm," in *Proc. IEEE Conf. Pattern Recognition*, 1988, vol. 2, pp. 1023-1025.

[18] R. W. Hall, "Fast parallel thinning algorithms: Parallel speed and connectivity preservation," *Communications of the ACM*, vol. 32, no. 1, pp. 124-131, 1989.

[19] R. F. Lyon, "The optical mouse, and an architectural methodology for smart digital sensors," in *Proceedings of the CMU Conference on VLSI Systems and computations*, 1981, pp. 1-19.

[20] M. Barbaro, P. Y. Burgi, A. Mortara, P. Nussbaum, F. Heitger, "A 100×100 pixel silicon retina for gradient extraction with steering filter capabilities and temporal output coding," *IEEE J. Solid-State Circuits*, vol. 37, no. 2, pp. 160-172, Feb. 2002.

[21] S. Jung, R. Thewes, T. Scheiter, K. F. Goser, and W. Weber, "A low-power and high-performance CMOS fingerprint sensing and encoding architecture," *IEEE J. Solid-State Circuits*, vol. 34, no. 7, pp. 978-984, Jul. 1999.

[22] N. Bourbakis, N. Steffensen, and B. Saha, "Design of an array processor for parallel skeletonization of images," *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol. 44, pp. 284-298, Apr. 1997.

[23] S. Shigematsu, H. Morimura, Y. Tanabe, T. Adachi and K. Machida, "A single-chip fingerprint sensor and identifier," *IEEE J. Solid-State Circuit*, vol. 34, no. 12, pp. 1852-1859, Dec. 1999.

[24] H. Morimura, S. Shigematsu, T. Shimamura, K. Fujii, C. Yamaguchi, H. Suto, Y. Okazaki, K. Machida and H. Kyuragi, "An advanced fingerprint sensor LSI and its application to a fingerprint identification system," in *Proc. IEEE Symp. VLSI Circuits Digest of Technical Papers*, 2002, pp. 272-275.

[25] H. Morimura, S. Shigematsu, T. Shimamura, K. Machida and H. Kyuragi, "A pixel-level automatic calibration circuit scheme for capacitive fingerprint sensor LSIs," *IEEE J. Solid-State Circuit*, vol. 37, no. 10, pp. 1300-1306, Oct. 2002.

[26] S. Shigematsu, K. Fujii, H. Morimura, T. Hatano, M. Nakanishi, T. Adachi, N. Ikeda, T. Shimamura, K. Machida, Y. Okazaki and H. Kyuragi, "A 500 dpi 224×256-pixel single-chip fingerprint identification LSI with pixel-parallel image enhancement and rotation schemes," in *Proc. IEEE Conf. Solid-State Circuits Digest of Technical Papers*, 2002, vol. 1, pp. 354-473.

[27] S. Jung, R. Thewes, T. Scheiter, K. F. Goser, and W. Weber, "CMOS fingerprint sensor with automatic local contrast adjustment and pixel-parallel encoding logic," in *Proc. IEEE Symp. VLSI Circuits Digest of Technical Papers*, 1999, pp. 161-164.

[28] D.A. Freitas and K.W. Current, "CMOS current comparator circuit," *IEE Electronics Letters*, vol. 19, no. 7, pp. 694-697, Aug. 1983.

[29] C. M. Brislawn, "The FBI Fingerprint Image Compression Specification," in *Wavelet Image and Video Compression*, P. N. Topiwala, Ed. Boston, MA: Kluwer, 1998, ch. 16, pp. 271 - 288, invited book chapter.

[30] "American national standard for information systems—Data Format for the Interchange of Fingerprint Information," American National Standards Institute, New York, NY, Doc. No. ANSI/NIST-CSL 1-1993.