Second Prize

Intelligent Card Technology-Based Biometrics Identification System

Institution: **Institute of Information Science, School of Computer, Beijing JiaoTong University**

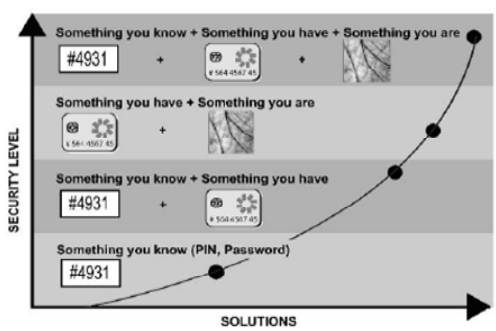
Participants: **Tang Hui, Liu Lulu, and Qin Lunming**

Instructor: **Ding Xiaoming**

Design Introduction

The design of a secure-identity authentication system involves a complicated decision-making process that verifies security certificates on-the-fly. When creating this design, we had to make a choice between design complexity and security considerations to arrive at an optimized solution. Combining biometrics recognition with smart card technology helped us design a good security/authentication system. Figure 1, an excerpt from a thesis of the Smart Card Alliance, 2004, illustrates the advantages of this approach.

Figure 1. Relationship between the Smart Card with Biologic Features & Security Level (Palm Print Recognition)



Advances in biometrics recognition and smart card technologies have enabled designers to use them in many applications. We carefully reviewed the status of these two technologies to use the best principles in our design, and then we combined our design knowledge with the Altera® system-on-a-programmable-chip (SOPC) concept. We used the cutting-edge features of the Nios® II soft core processor to develop a Smart Card Biometrics System. For the biometric feature, we used palm print identification because of its usage worldwide and because we have a strong theoretical foundation on the biometrics of palm printing. (We are also familiar with voice print biometrics.) Given this backdrop, we can design richly featured, highly secure identification systems based on either technology. Alternatively, we can combine both palm print and voice print to arrive at a multimode biometric feature smart card.

Application Scope

Combining biometrics recognition with smart card technologies allows high-performance deployment in many areas. For example, biometrics recognition technology can enable an enhanced security level feature for smart card applications. Additionally, using smart card technology saves time in storing and retrieving data from a local database, thus accelerating the matching process. The application scope and solutions that are available today using these two technologies are described as follows:

- *School e-Card*—Today, most schools are still using magnetic cards, which have severe limitations such as short lifespan, limited data volume, and poor security performance. Instead, using a combination of biometrics and smart card technologies, schools can enhance security, data volume, card lifetime, and application scope.

- *Talent Card*—At present, the Beijing Talent Service Center is developing relevant standards for the talent industry, and is going to distribute cards, using smart card technology, for all talents to facilitate planning and management. Security is a big factor because of the importance of the talent cards. Incorporating biometrics into the smart card will undoubtedly ensure high security for the whole system.

- *High-Class Club Member Card*—Member cards of a high-class club are very valuable. Losing a card results in a loss to both the club and the card owner. Combining biometrics recognition with smart card technologies can make member cards more secure.

- *Door Lock Management System*—The biometrics recognition system will be a great advantage in the performance of a door lock secure management system. These systems are traditional security applications in which one can deploy biometrics and smart card technologies to a great advantage.

The users for these applications is decided by the application scope. Our smart card biometrics recognition system has a variety of users ranging from high-end users, such as senior club members, to ordinary users such as school students and residents in a housing colony.

Advantages of Using the Nios II Processor

There are two main advantages of using the Nios II processor: it can be broken into modules and configured. Each subsystem is composed of function modules, which coordinate with each other to form a complete system. The SOPC design approach minimizes the system's reliance on submodules during the design process and increases the cohesion between modules. Therefore, our system design tasks are divided into small modules. Taking this design approach we can rapidly build a complete system with greater precision that meets the design requirements. This design method minimizes the influence of a single module on the whole system.

The Nios II processor features configurable, built-in hardware and software system modules. We can configure the system dynamically for different system requirements during the development stage to optimize system performance under different conditions.

Function Description

In applications involving biometric recognition, the recognition function needs to be executed with precision at high speed for large-scale deployment of the system. The essence of biometric system design is to speed up the processing in each segment while ensuring precise recognition. Based on this concept, our system has distinct features in many aspects. Our system implements the features described in the following sections.

Collecting Biometric Data & Transmission in DMA Mode

A simulated collector collects the biometric data, which saved development time. We also reduced the time required for image data transmission after collection. Because the data is stored in blocks, it can be transmitted in DMA mode. The Nios II processor provides good support for data transmission in DMA mode, which makes it possible for designers to select appropriate transmission mode to meet the design requirements.

Biometric Data Extraction & Compression, DSP Builder & Customized User Instructions

The extraction and compression of biometric data is complicated because it involves digital signal processing (DSP). We had some problems determining which algorithm was more effective when deployed under an embedded system environment. To solve the problem, we tested existing algorithms in a PC environment, and chose the most appropriate algorithm that met the practical requirements for use in an embedded system.

It was quite difficult to achieve real-time processing using software mode while executing the algorithm. Therefore, we used a hardware mode to accelerate processing in key segments of the algorithm to satisfy the system design requirements. Specifically, we used the following approaches:

- The MATLAB software is easy to use and allows for quick computation of arithmetic operations, so we used it to perform algorithm-level emulation. Next, we generated hardware modules using Altera's DSP Builder tool, and integrated it into our system as the system's DSP module. This approach enhanced system performance.

- We implemented frequently called basic functions and statements used during processing with the Nios II processor's customizable user instructions. These custom instructions improved the processing speed.

Card Reader/Writer Integration & User-Defined Peripherals

To integrate the smart card reader/writer module, we removed the redundant serial port components and assigned the Rx/Tx pins of the serial port to the extended I/O using SOPC Builder. This scheme made it possible to operate the card reader/writer module and collection module in one daughter board. Then, we integrated the control unit of the card reader/writer to the system bus using customized peripheral feature of the system. In this way, we simplified the control design and ensured faster data processing. Because the biometric feature data in the smart card is stored in blocks, we used DMA transmission mode to reduce processing time during transmission.

Control of a Complex System Using RTOS vs Multi-Core Technology

We tried to avoid a complex control mode scheme using the modularization concept during the control design of the whole system. However, we anticipated that the system complexity might exceed our expectations at the early stages of the design. Fortunately, the Nios II processor solved the problem by making it easy to load the real-time operating system (RTOS) to the system. The management of real time tasks, memory, and peripheral devices using the real time operating system allowed us to take advantage of the capabilities of Nios II processor fully. By doing so, we enhanced the utilization of system resources and processing capability, reduced processing time, and speeded up the system response. In our opinion, you can also consider the multi-core operation in your design.

Performance Parameters

Effective recognition and speed are the most important performance parameters for the biometric identification system. Our goal was to enhance the recognition rate and processing speed. For faster processing, we took advantage of Altera's design platform and reduced recognition speed to an acceptable figure. This figure included the time for a single recognition process (including complex pre-processing, feature extraction, compression, and smart card reading) to within 4 seconds, without any design modification.

To effectively verify the recognition rate, we exercised the system 200 times, each time verifying the palm print samples against a database of palm print samples library. The experimental data indicated that through the selection of an appropriate threshold value, the system can resist all of the 200 attacks.

The excellent performance of the Nios II processor made it easy for us to realize our design goals. We chose the Nios II/f core because we had a strict requirement with regard to the processing speed. Combining the Nios II processor and FPGA design, we developed an ideal mix of processor, peripherals, memory, and I/O interface. This design approach enabled us to meet the design requirements with a high resource utilization: 92% of logic element (LE) usage and 84% of on-chip memory usage.

Using the Nios II processor helped us enhance system performance by modifying function blocks, even amidst design stages. For example, we used a 50-MHz MCLK and the Nios II/s core, but this design required 12 seconds to extract voice prints, which was unacceptable. Next, we used a 75-MHz MCLK and the NiosII/f processor, and added ICACHE and DCACHE functions, which took about 3 seconds to extract the biometrics data and met our design requirements.

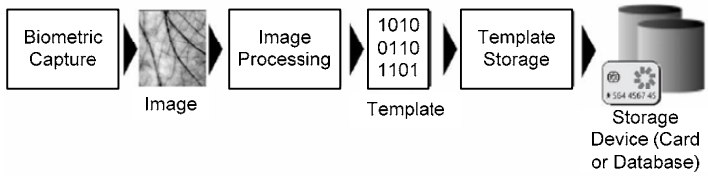
We were able to enhance system modules' performance further by using RTOS, DMA, and user-defined instructions from the Nios II processor.

Design Architecture

Taking palm-print recognition as an example, this section describes the design flow and system modules. For other biometric systems, the difference lies in the collection and feature extraction modules. The rest of the modules remain the same, which is a major advantage of using SOPC in a modular design.

The system has two major functions: registration and authentication. In the registration function, the system has to complete at least the extraction and comparison of biometric data and card reader/writer control. During registration, the palm print feature data is extracted through a palm print collection terminal and is stored in the smart card using the card reader/writer module. Figure 2 outlines the registration and authentication system tasks.

Figure 2. Registration Process Diagram of Palm Feature



During authentication, both the smart card and related palm print data are displayed. The collected palm print data is compared with the palm print data stored in the smart card in real time through processing of data in collection terminal. The authentication is a success when the two data values are consistent with each other, or else the authentication fails. See Figures 3 through 7.

Figure 3. Authentication Process Diagram of Palm Print Data

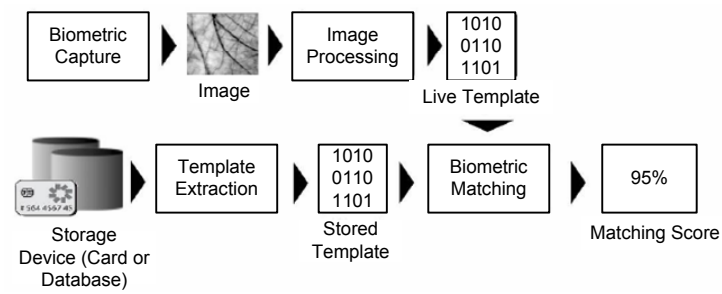


Figure 4. Simple Block Diagram of Biometric System

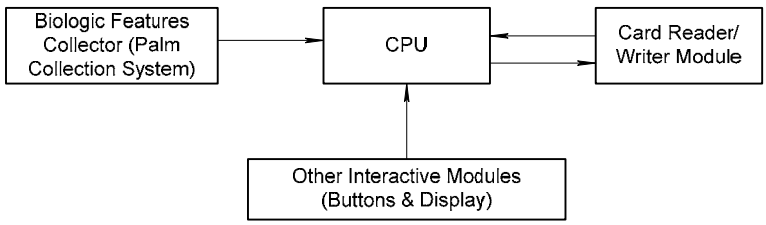


Figure 5. Design Diagram of System Hardware

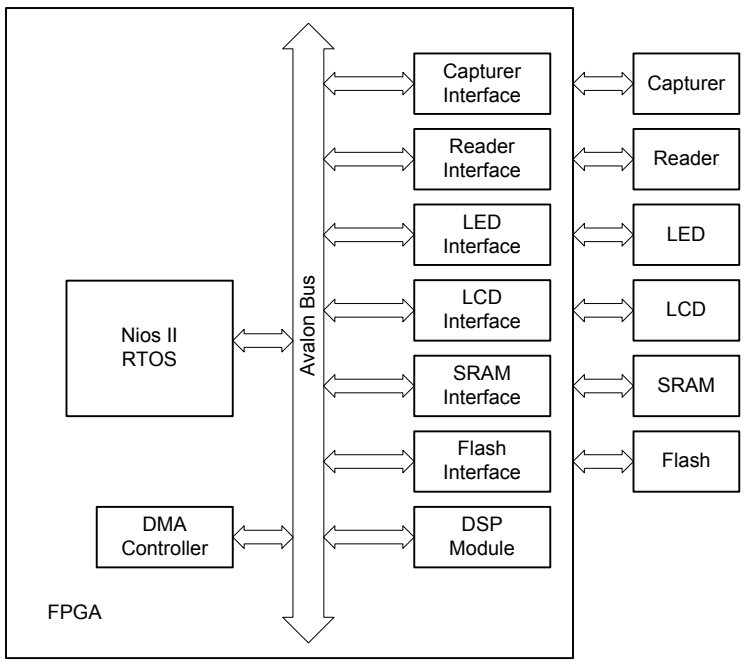


Figure 6. Operating System & Task Distribution Diagram

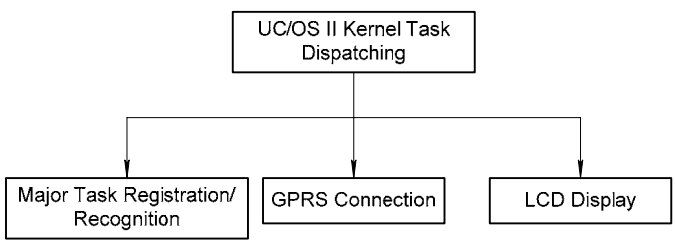
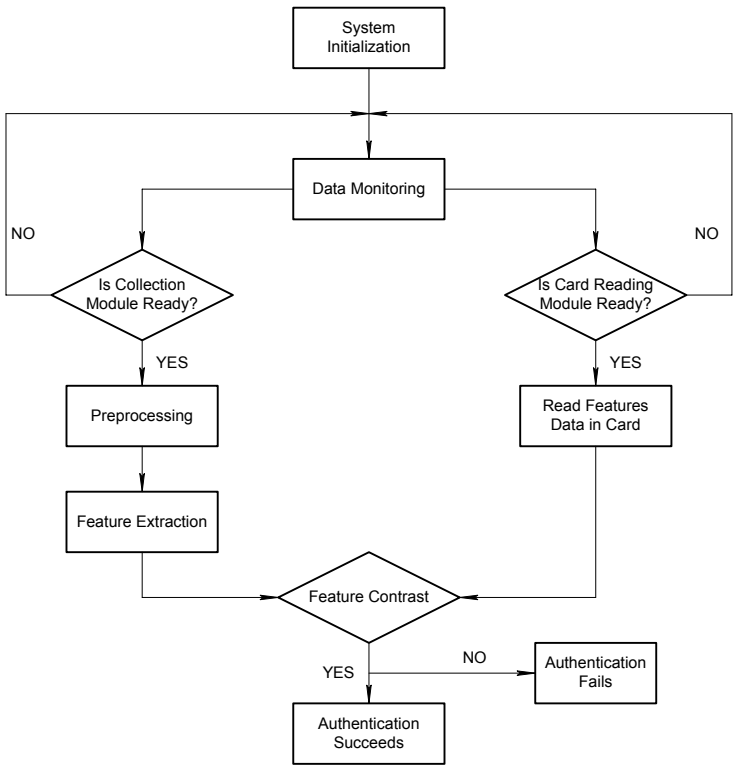


Figure 7. Main Task Workflow Diagram of the System



Design Methodology

This section describes the design methodology for the system.

Design Environment

For the purposes of this contest, we wanted to use a common design environment based on UP3 education suites. Based on the UP3 board, we developed and realized many basic modules, such as palm print (voiceprint) collection, preprocessing, feature extraction, and the read/write tasks of the smart card. Due to a lack of sufficient resources on the UP3 board, it was hard to meet the design requirements for system speed, design integration, and hardware processing speed. Therefore, we decided to switch to Altera's development board during the middle phase of the design.

Design of System Hardware

Figure 8 shows the SOPC Builder tool settings.

Figure 8. System Design Configuration

Use	Module Name	Description	Clock	Base	End
☑	⊞ cpu	Nios II Processor - Altera Corporation	clk	0x00800000	0x0
☑	⊞ jtag_uart	JTAG UART	clk	0x008008E0	0x0
☑	⊞ ext_flash	Flash Memory (Common Flash Interface)		0x00000000	0x0
☑	⊞ ext_ram	IDT71V416 SRAM		0x02000000	0x0
☑	⊞ ext_ram_bus	Avalon Tri-State Bridge	clk		
☑	⊞ sys_clk_timer	Interval timer	clk	0x00800800	0x0
☑	⊞ high_res_timer	Interval timer	clk	0x00800820	0x0
☑	⊞ sdram	SDRAM Controller	clk	0x01000000	0x0
☑	⊞ pio_remain	PIO (Parallel I/O)	clk	0x00800860	0x0
☑	⊞ reset_getimage	PIO (Parallel I/O)	clk	0x00800870	0x0
☑	⊞ seven_seg_pio	PIO (Parallel I/O)	clk	0x00800880	0x0
☑	⊞ rdy	PIO (Parallel I/O)	clk	0x00800890	0x0
☑	⊞ i2c	OpenCores I2C Master	clk	0x00800940	0x0
☑	⊞ button_pio	PIO (Parallel I/O)	clk	0x008008A0	0x0
☑	⊞ reconfig_request_pio	PIO (Parallel I/O)	clk	0x008008D0	0x0
☑	⊞ uart1	UART (RS-232 serial port)	clk	0x00800840	0x0
☑	⊞ uart2	UART (RS-232 serial port)	clk	0x00800900	0x0
☑	⊞ cam_buf_bus	Avalon Tri-State Bridge	clk		
☑	⊞ cam_buf	IDT71V424 SRAM		0x00880000	0x0
☑	⊞ dma	DMA	clk	0x00800920	0x0
☑	⊞ onchip_ram_64k	On-Ch (avalon_tristate) or ROM)	clk	0x02100000	0x0

Design Implementation

We broke down the system tasks into five parts:

- Biometric Feature (Data) Collection Module
- Preprocessing Module
- Voice Information Preprocessing
- Feature Data Extraction & Compression
- Smart Card Read/Write Module

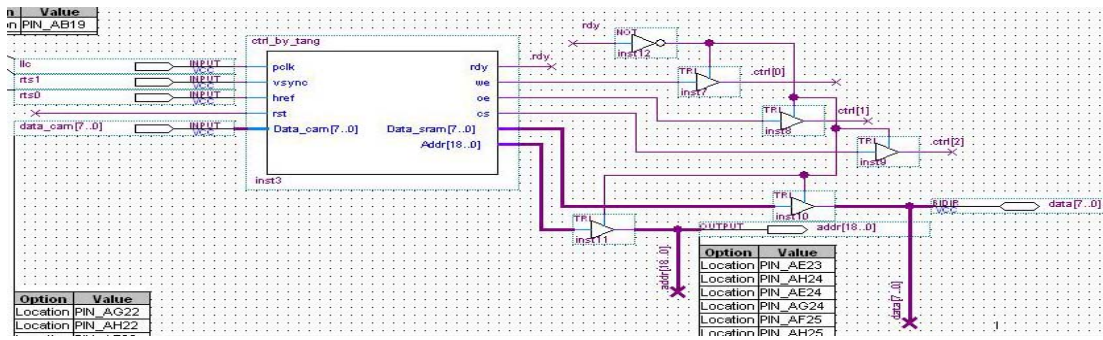
Biometric Feature (Data) Collection Module

The biometric feature collection module is the indispensable front-end, whose quality of data collection directly impacts the recognition effectiveness of the whole system. This module also affects the collection speed, which can undermine system performance if not addressed properly.

In the voiceprint data collection system, data is captured through PC microphone and audio card. For collecting palm print data, we simulated camera collection action and processed this data using the SAA7113 chip. The SAA7113 chip in turn is controlled by I^2C bus, whose control module is realized using the customized peripheral function feature available with the Altera SOPC Builder design tool. Also, thanks to the availability of many intellectual property (IP) cores and customized peripherals, you can easily integrate them into your system using SOPC Builder. This is one of the major advantages of SOPC design.

We designed the IP core that controls camera and external storage devices and integrated it into the system; this IP core can complete most of the control operations of palm print image collection. We wrote this IP core in VHDL and its function module is shown in Figure 9.

Figure 9. Front-End Palm Print Collection Control Module



Preprocessing Module

The collected palm print image data contains many kinds of noise and features different palm print sizes, whose image locations may vary based on the angle of exposure. So it is necessary to preprocess the collected image that forms the basis of the palm print feature extraction so it can be compared with stored image data. Therefore, the preprocessing module plays a key role in the system's palm print recognition effectiveness. The preprocessing module we used has a three-step approach:

- *Area of Interest Positioning*—The positioning algorithm first makes a binary processing of the source image, and then obtains data of the interested area using a geometry morphology algorithm. The existing positioning algorithm is quite complicated, so it is realized mainly with software. Due to the large size of the image, we used hardware to perform binary processing, and subsequently used DMA to transmit the image data to the main storage.

- *Image Balance*—Because the captured image data is easily affected by light, we performed light-balance processing to set the image data on a grayscale standard. To complete the image balance, we analyzed the image grey histogram, and then processed this data by hardware.

- *Median Filter*—Median filtering is a common method to remove noise effects present in the signal. We developed a parameterized median filter hardware module that can handle a 2-D median filter on an N[x]N window. In our system, we used a 3[x]3 window to perform median filtering on the interested area to remove noise.

Voice Information Preprocessing

Voice preprocessing includes quantifying, split frame, pre-weighting, breakpoint checking, and removing noise from voice data.

- We digitized the input simulation voice signal by quantifying and sampling the audio signal. The digitized signal is then downloaded onto the development board flash memory.

- The voice signal remains stable from 10 ms up to 30 m, which means all algorithms have to process data within this short time period. We chose a 30-ms voice sample as a data frame in our design.

- Because the voice signal is affected by the glottis pulse shape and lip radiation, the voice-signal spectrum brings down the high-frequency component, which equals to a 6-db drop per octave. To remove this effect, we enhanced the high-frequency component using a pre-weighted, simple, first-order FIR filter, in the form of H(z)=1-a*z-1. The pre-weight filter helps to compress the dynamic range of the input voice effectively, and makes the linearity forecast analysis more stable. Moreover, this high-pass filter can also filter the DC component in the input signal.

It is important to determine the voice signal start and end points correctly. Some popular voice parameter data that help to judge voice signal start and end points include short-time energy, short-time average power, and short-time zero-cross rate.

Feature Data Extraction & Compression

We used the I2DPCA algorithm to perform feature extraction on the preprocessed palm print image. The I2DPCA is a proven subspace analysis algorithm whose validity has been verified in laboratory analysis. A major advantage of the I2DPCA algorithm lies in the fact that it helps you to reduce feature dimensions while ensuring a high-recognition rate. Because there is limited storage space on a smart card, the I2DPCA algorithm was a great help in storing the palm and voice feature data. But deploying the I2DPCA algorithm was not easy; it took complex arithmetic operations that included many iterative floating-point operations. To ensure real-time operation of the system, we added custom instructions for addition, subtraction, multiplication, and division operations on floating-point data.

The extraction of voice print data features involves keeping the language content while preserving individual voice characteristics. There are two kinds of voice characteristics: the difference of inborn vocal organs, such as the acoustic duct length and vocal cords and acquired speech characteristics, such as the dialect and tones. Because it is hard to extract these characteristics separately, we stored both.

Presently, the Mel-Frequency Cepstral Coefficients (MFCC) method is extensively used to differentiate speakers' voice feature parameters. The MFCC method involves computing real voice signals, applying a FFT on them, and then convoluting the resulting logarithm energy spectrum with Mel-Scale Triangular Filter. Finally, we carried out a discrete cosine transform (DCT) on the vector composed by filter outputs.

Smart Card Read/Write Module

This module uses ZLG500B from ZLG Corporation. The following table describes the system interface definition.

Pin	Symbol	Type	Description
J2-1	CTRL	Output	Control wire output
J2-2	BZ	Output	Buzzer signal output, high usually, output square-wave or low-level enabled
J2-3	CON485	Output	RS485 control, low usually, high during TXD sending
J2-4	VCC	PWR	Power plus end
J2-5	RST	Reset	MCU reset, high-level enabled
J2-6	GND	PWR	Power minus end
J2-7	RXD	Input	UART receiving end
J2-8	TXD	Output	UART sending end

Adding two serial ports and three PIOs into SOPC Builder completed the hardware design of this module. We had to assign the pins of the serial port to receive/send and set up the baud rate accordingly.

We defined the following control characters in our software design:

```
#define STX           0x20   //Start of Text
#define ACK           0x06   //Affirmative Acknowledgment
#define NAK           0x15   //Not Affirmative Acknowledgment
#define ETX           0x03   //End of Text
```

For communication purposes, the host sends data to the ZLG500B device, and after it executes the command, it returns the state of command execution and relative data to the Nios II processor. The receiving/sending part must be initialized before communication starts. First, the Nios II processor

sends out an STX, and waits for the ACK response from the ZLG500B device. If the Nios II processor does not receive a response within 10 ms or receive a NAK, it sends another STX. It repeats this action three times. Then, the Nios II processor quits the transmission mode and returns an error code to the main program that processes the error. On the other hand, if the Nios II processor receives the ACK response from the ZLG500B device, it sends out a block of data, and transmits an ETX to signal the end of transmission. The following table shows the format of Nios II transmission:

Nios II Processor	Data Transmission Direction	ZLG500B	Description
STX	——>		If the Nios II processor is unable to receive an ACK or NAK within 10 ms, it resends STX at least once; after receiving an ACK, the Nios II processor must send data within 50 ms, and the time interval between 2 bytes sent must be less than 10 ms.
	<——	ACK	
DATA+ETX	——>		

After this, the Nios II processor waits for status data and response from the ZLG500B device. If it does not receive a response within 300 ms, the Nios II processor quits transmission mode and reports an error code to main program. The format of ZLG500B transmission is shown in the following table:

ZLG500B	Data Transmission Direction	Nios II Processor	Description
STX	——>		ZLG500B won't resend STX if it hasn't received ACK within 50ms.
	<——	ACK	
DATA+ETX	——>		

The biometric feature information read from a user's smart card is matched with collected data using the above processes. During matching of biometric features, the algorithm adopts Euclidian distance to represent the distance between two palm print features. First, the palm print feature data in the user's smart card are read into system and matched with palm print features obtained using the described processing tasks. By converting the obtained biometric feature data from a floating-point to fixed-point format, the smart card can process data faster and can conserve storage space.

The SOPC concepts utilized in our design are described as follows:

- *Modular System Design*—In the early stages of design, we spent extra time partitioning the system design. Design partitioning is useful because it helps to define every task of the system. It also helps to simplify system design, thereby improving the designers' confidence. Based on the partitioned modules, the applications' scope and expandability can be correctly evaluated at the beginning of the design cycle. Therefore, it is possible to initiate marketing tasks during product design while simultaneously working on R&D, which is a great benefit to enterprises because it reduces the product's launch period.

- *System Integration*—Embedded systems need to have a balanced design that encompasses aspects of product volume, power, and design integration. Therefore, it is important that designers carefully think over integration and balance between integration and cost during the realization of the system. In our design, besides the front-end collection module, all other system functions were completed using a development board, which made it possible to achieve a highly integrated design under the same design goals. It would be very difficult to realize the system without using a soft core processor based on an FPGA design approach.

- *Diverse Implementation of Different Modules*—The implementation methods are diverse for different modules. For example, the front-end collection module used an IP core for its realization while the preprocessing module relied on software and hardware (processing arithmetic functions based on customized instructions). The palm feature extraction module adopted a combination of software and hardware and the smart card module used built-in peripherals. Thanks to the SOPC methodology, it was quite easy to implement these modules using Altera's development tools.

- *Diversification of Module Combination*—After attaining the basic design goals of the design, we could easily redesign using the existing modules, if needed. For example, using the μC/OS-II operating system, we added a general packet radio transmission (GPRS) module, and at the same time managed to run the registration recognition program. Thus, we were able to display two subtasks with GPRS on LCD, greatly enhancing the application scope of the system.

  Further, the choice of biometric feature can be selected using palm print data collection module or voiceprint module. Or, we can also choose a multi-module biometric feature module, which can handle both voice and palm print data with enhanced security features.

- *System Upgrade*—Using the SOPC approach, it is fully possible to reconfigure the system both at the beginning and middle of design cycle. If need be, you can also reconfigure the system after completing the design.

Design Features

The following table shows a comparison of system demands versus SOPC design platform features:

	System Demand	Platform Features
Hardware	High-speed computing capacity is required in the extraction and compression of biometric features and feature matching. When integrating a smart card reader module with the system, the traditional method of serial port communication will not help in achieving a highly integrated system.	When designing system hardware, the designer can use SOPC Builder development platform to choose between a hardware acceleration module or a DSP module according to system demands so as to meet the special requirements of hardware acceleration. SOPC design is convenient for management and operation of peripherals. For our system, the serial port can be located at the expanded I/O interface easily to realize interconnection between boards. By doing so, you can avoid the serial port's unreliability of long wire connection. Furthermore, the smart card reader module can be integrated into the system as a user-defined peripheral to improve system integration.
Software	Currently, the most efficient extraction and compression algorithms as well as matching of biometric features algorithms are all available on PC and are simulated using simulation tools like MATLAB. The embedded implementation will not be as efficient because of above reasons. The effective and coordinated operation of modules requires the best use of RTOS.	The Nios II processor can be adapted to enhance and reduce system performance according to specifications. For example, we can take advantage of the Nios II processor's flexibility and clipping of data values when comparing the advantages of arithmetic with embedded systems to choose the best option for implementation. Additionally, the FPGA provides advantages in time sequence and logical processing. The custom instructions are crucial for high-speed processing in some applications. In a Nios II system, the RTOS is very easy to deploy, which makes it easy for a designer to handle a complex system. In addition, the multi-processor kernel technology and DMA functions ensure excellent system performance.
Feature upgrades and cost	System developers expect quick launch of products and a longer life-cycle to sharpen the competitive edge and beat rivals. At the same time, users want the latest features in their products. Cost is a crucial element in the development of embedded systems. Therefore, lowering the system cost through design is a key problem for designers at the beginning of the project.	Because the FPGA is a programmable device, the time to market is relatively short. The Nios II processor's flexibility coupled with Altera's integrated development kit, abundant reference designs, powerful hardware development tools (SOPC Builder), and software development tools (Nios II IDE) make it easy to achieve all the expected design goals. The Nios II processor allows easy upgrade of both hardware and software in real time. An FPGA-based system design integrates processor, peripheral, memory, and I/O interface to lower cost, complexity, and power consumption.

Conclusion

Referring to the design requirement documents provided in the contest, we summarized our system design in the following table:

Design Phase	Category	Score	Examples
Design Concept	Complexity	5	The design adopts two tasks of RTOS: high PRI of major task is the processing of biometric features the subtask is the LCD display.
		4	Design uses a DSP algorithm. The palm print recognition uses AOI-orientated algorithm of the lab and I2DPCA feature-based recognition algorithm. Voice print recognition adopts FFT transforms.
		3	Design uses greater than 70% LE/memory utilization (refer to compilation report LE/memory utilization which are 92% and 84%, respectively).
		2	Design uses more than two masters on Avalon® bus (DMA utilization).
	Functionality	5	Design realized customized peripherals and customized instruction on the same chip. These included customized peripherals idt71lv424, I^2C control and customized peripherals floating point instructions.
		2	Design connects more than eight peripherals on SOPC bus.
Design Implementation	Completeness	5	Design fulfills complete software implementation and hardware demonstration.
Documentation	Completeness	5	We submitted an integrated design document with diagrams and complete description of the final design.

Throughout this paper, we have described the convenience of using the Nios II processor and an SOPC design approach for embedded designs. Using tools like the Quartus II software and SOPC Builder, as well as embedded debugging software, gave us great confidence while accomplishing our design.

Altera provided us with an excellent design approach—SOPC design based on the Nios II processor and an FPGA as well as popular design tools—Quartus II software and SOPC Builder. At the same time, Altera and many third-party providers supplied plenty of debugging software. We made good use of these tools in our design, which shows our design strength.

Early on in the design cycle we needed to be careful in formulating our design so that we could solve problems with a clear mind. A strong careful beginning also helped avoid problems and is a good design technique. It is natural that we would still encounter some problems, but a solution to these problems enables a 'great leap' in reaching the design goals. Of course, this maturity towards problem solving comes from being exposed to many design approaches and experience.

There are many ways of implementing the design and a specific implementation depends on the design target and ease of implementation. Do abstain from trying to realize all functions in software owing to your familiarity with the C programming language. Also, your familiarity with VHDL may make you overlook publicly available IP resources. Make sure that the SOPC design based on Altera FPGAs calls for a simplified design and emphasizes the idea of system analysis design in an embedded system. This approach is not to be confused with module design, which is a necessary requirement. This is the trend of embedded design today.

The quality of a design team is very important. Good coordination between team members is important to accomplish increasingly complex embedded design systems.