

三等奖

基于Nios的指纹识别系统

大学院校： 华中科技大学

参赛队员： 李临川 张尧 葛成东

指导教师： 肖看

一. 设计概述

1. 设计背景

近年来，随着数字化、信息化、网络化等技术深入发展，人们的生活变得越来越方便快捷，同时对于各种电子系统的安全性要求也越来越高。电子商务、ATM、门禁控制及各种智能卡等都需要一种安全而且易用的个人身份识别技术。传统的“用户ID+密码”的认证方式存在密码遗忘、黑客攻击和被别人窃取等问题，已经渐渐无法适应社会的需要。基于人体生物特征的身份识别技术为我们提供了一种很好的解决方案。

生物特征识别技术根据生理特征或行为特征对个人身份进行鉴别，因此它具有更高的安全性和可靠性，同时使用方式也更富于人性化。常见的生物特征包括指纹、掌纹、虹膜、脸像、声音、笔迹、DNA等。综合考虑准确性、永久性、易用性和成本代价，指纹识别技术是一种费效比和安全性都很高的方案，同时技术上也比较成熟，因而在社会上有很广泛的应用。据统计，国内市场上基于生物特征识别的身份认证系统中，指纹识别产品占到90%以上。

随着微电子技术的发展，可编程逻辑器件规模越来越大，越来越快，功能越来越强。目前已经有若干种FPGA器件支持嵌入式处理器软核，为基于FPGA的硬件开发提供了更多的选

择。Nios® II嵌入式处理器是Altera公司开发的一种采用流水线技术、单指令流的RISC嵌入式处理器软核，可以将它嵌入FPGA内部，与用户自定义逻辑结合构成一个基于FPGA的片上系统。与嵌入式硬核相比较，嵌入式软核具有更大的灵活性。而FPGA的高速度，恰恰满足了指纹识别系统对速度的要求。

本设计考虑了工作在验证模式下的分散认证的身份识别系统，用户在终端经声明身份（这里简化为输入ID）、输入指纹后进行认证（或注册），由一个主机通过网络管理多个终端，管理员拥有系统管理权。

2. 针对用户群

本系统可以在整合其它服务功能模块后作为一个公共服务系统，其中指纹识别作为关键的身份认证手段；系统也可稍加修改后作为人员管理、安全保护等产品。具体的应用可能有：

- (1) 电子商务：如信用卡消费、E购网络。
- (2) 银行：ATM等。
- (3) 企事业单位人事管理。
- (4) 安全管理：如门禁系统。
- (5) 资格认定：如考试。

本系统的主从机网络模式可以有效的实现分散认证、集中管理的工作，因而比较适用与局部范围的认证系统，如果在通信效率和安全性上加以改进，可以扩展为大系统。

3. 应用 Nios 的优势

传统的指纹识别技术大多依靠PC或DSP实现，PC实现图像处理成本高、体积大，速度无优势，DSP虽然处理速度快，但有功能和相关参数固定，灵活性不足的缺点。FPGA应用于指纹识别后，体现出越来越大的优势，它兼顾了处理速度和灵活性，同时有嵌入式系统固有的低成本、便携的优点，因而获得了广泛的应用。Nios作为一种高性能、可配置的软核，又有它独特的优势。它基本用C语言编程，开发周期短，代码可移植，结合用户自定义硬件逻辑，可以并行的完成复杂的图像处理任务，不失FPGA的优势。对本系统来说，相关组件可以很容易整合，图像处理过程中的相关参数可以很方便的调节，这对于实现不同的性能指标，满足不同条件下的用户需求是很有利的。

二. 功能描述

1. 实现了可扩展的认证网络：

系统为主机与终端模式，采用总线型局域网进行连接，便于集中化管理，并且具有良好的可扩展性。主机管理员只需在“终端管理”选项卡中新增终端，就可以方便地加入新的终端。

2. 良好的人机交互：

采用液晶显示与键盘录入的方式，便于客户的使用。

3. 指纹采集：

通过自制的指纹采集器采集用户的指纹，驱动程序访问 SPI 接口获取数据。程序设有

手指自动检测功能，采集指纹时针对手指不同干湿湿度设置了三套参数，选出效果最好的一幅图像，另外，指纹采集器平时处于休眠状态，采集指纹时才激活，降低了功耗。

4. 身份验证:

终端采集器采集指纹信号，进行图像处理，并提取出指纹特征信息。在注册情况下，主机从终端获得指纹信息，及对应的 ID，并存入指纹库中；在登陆情况下，主机返回 ID 对应的指纹信息，由从机进行比对，并且显示相应的登陆信息。

5. 信息管理:

主机由功能较强的 PC 担任，运行一套管理程序，其功能包括：

用户账户管理，查看或修改注册用户；

终端管理，新增终端或修改终端权限；

日志查看，浏览系统访问日志；

密码修改，修改管理员登录密码。

三. 性能参数

性能参数包括两个方面，指纹图像处理速度和指纹识别正确率。

1. 指纹图像处理速度:

下表列出了采用硬件加速前后指纹图像处理的主要过程耗时及总耗时（硬件加速的具体配置和原理见第五部分“设计方法”）。处理对象为 256×300 的8位灰度图像。

	未采用硬件加速耗时	采用硬件加速后耗时
图像滤波	36.40s	4.77s
纹线细化	13.54s	2.67s
总计	54.93s	11.57s

可以看到，在采用了针对图像处理的硬件加速后，图像处理速度有了很大的提高。

2. 指纹识别正确率

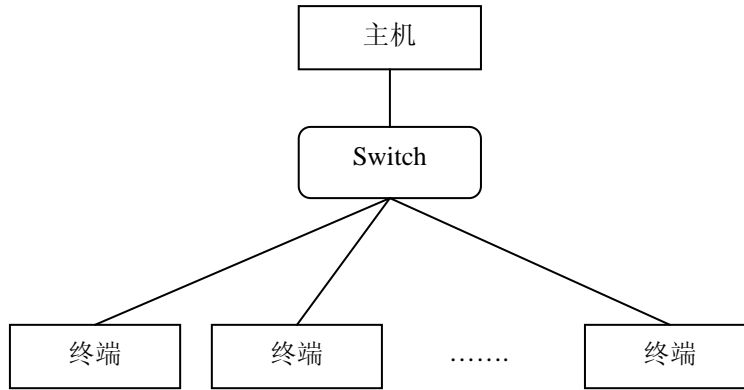
由于本系统的指纹图像处理和比对的算法是针对特定的指纹采集器设计的，故没有在通用指纹库中进行测试，而是随机选取了10个测试者共计约40枚指纹测试系统的指纹识别正确率，具有一定的代表性。

经过统计，系统误识率FAR（False Accept Rate，即非同一指纹误识为同一指纹概率）在5%以下，拒识率FRR（False Reject Rate，即同一指纹误识为不同指纹概率）在20%以下。

由于识别正确率很大程度上受手指清洁度和干湿度的影响，故以上的结果为系统指纹识别综合性能，没有单独进行算法性能测试。

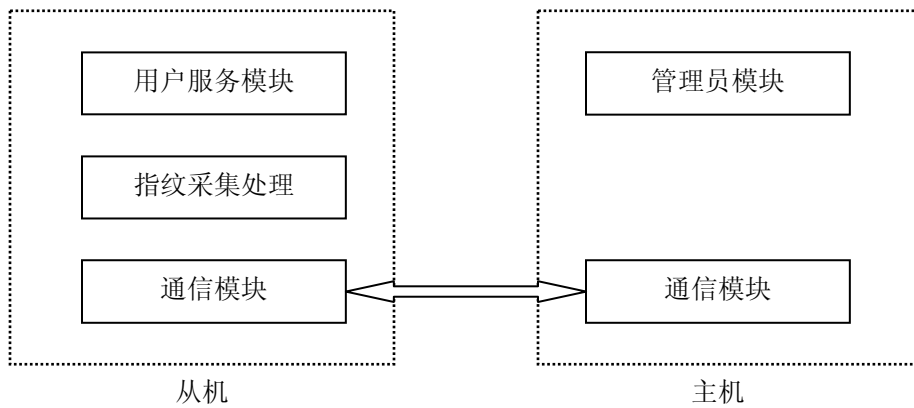
四. 设计结构

1. 网络拓扑结构



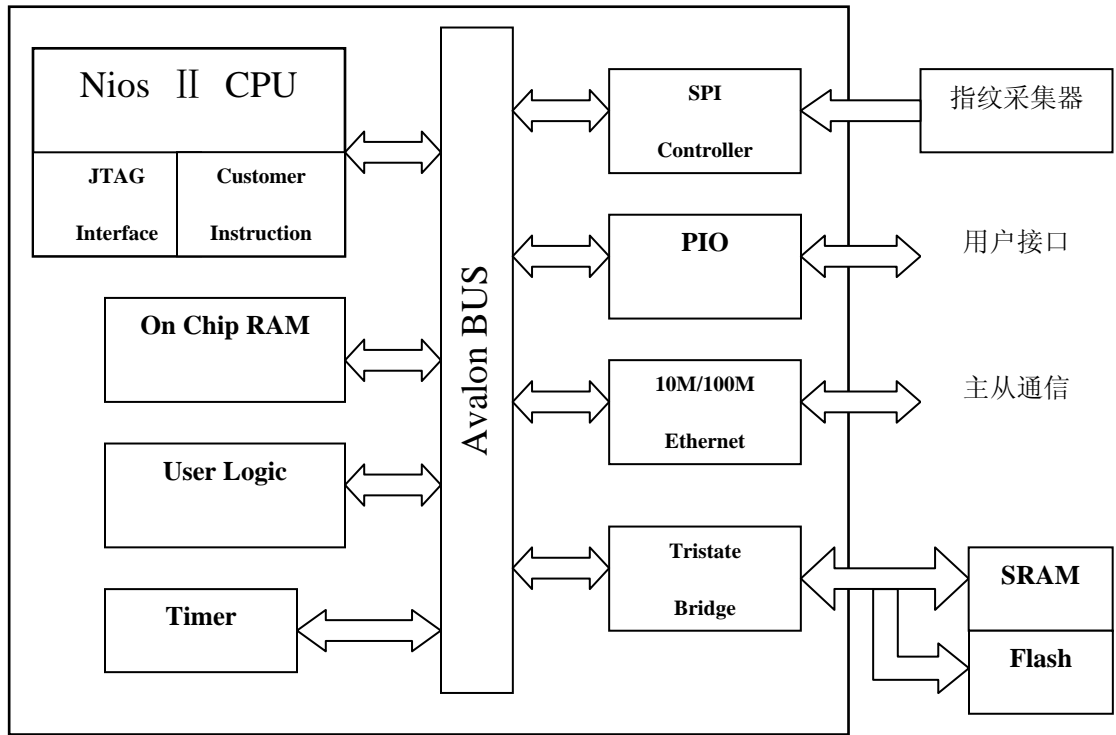
系统以交换机为中心节点进行组网，建立各个终端与主机的连接。

2. 模块划分:



3. 硬件设计:

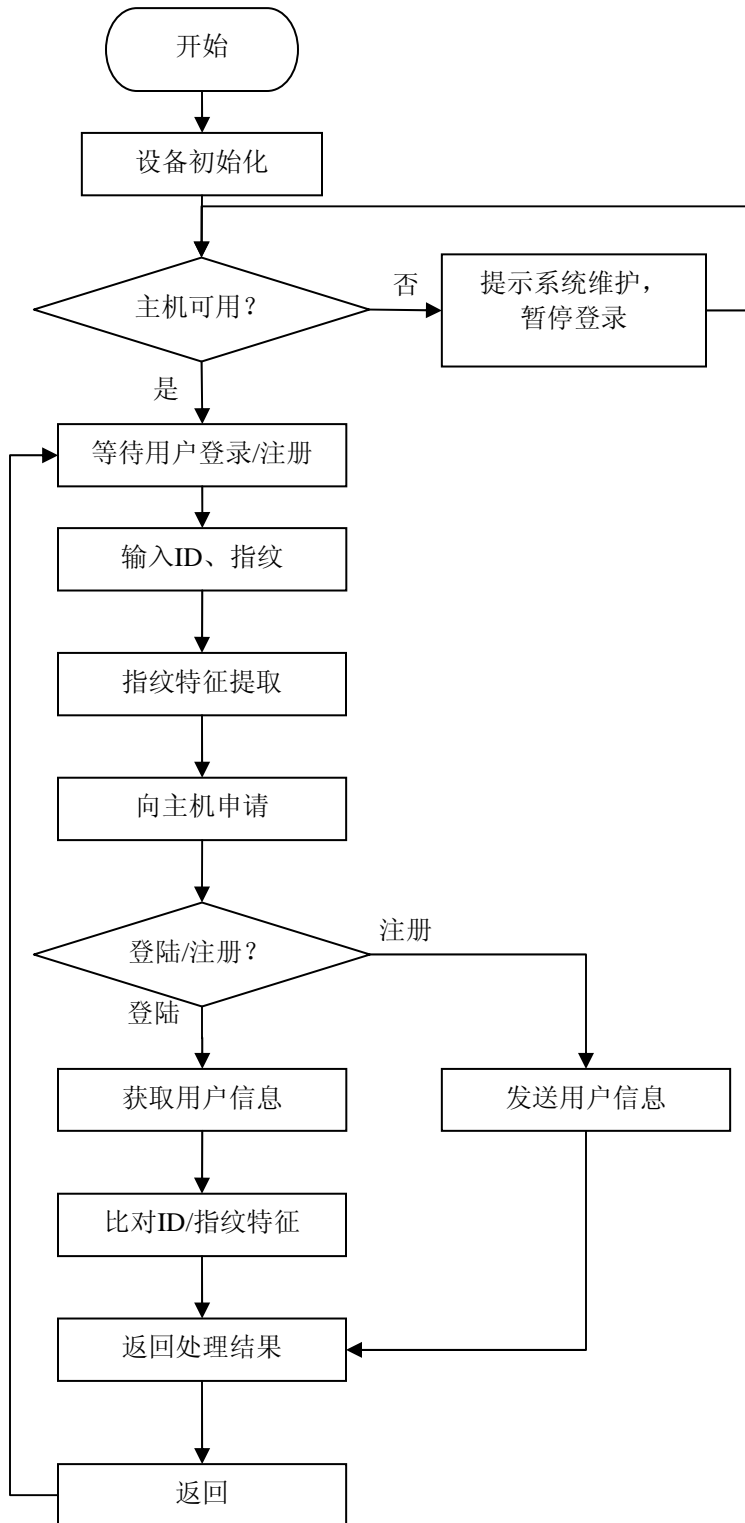
从机:



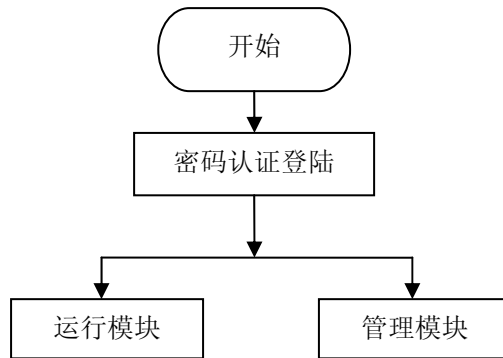
主机: 由PC担任。

4. 软件设计:

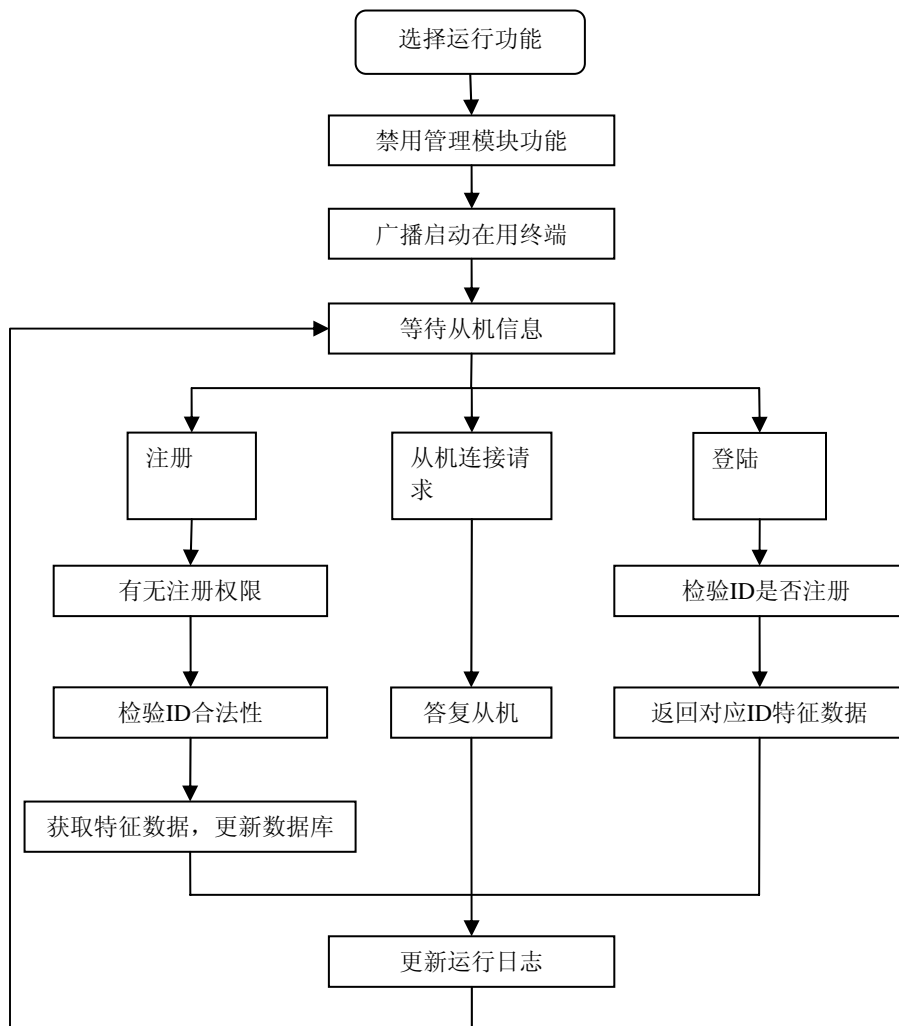
从机:



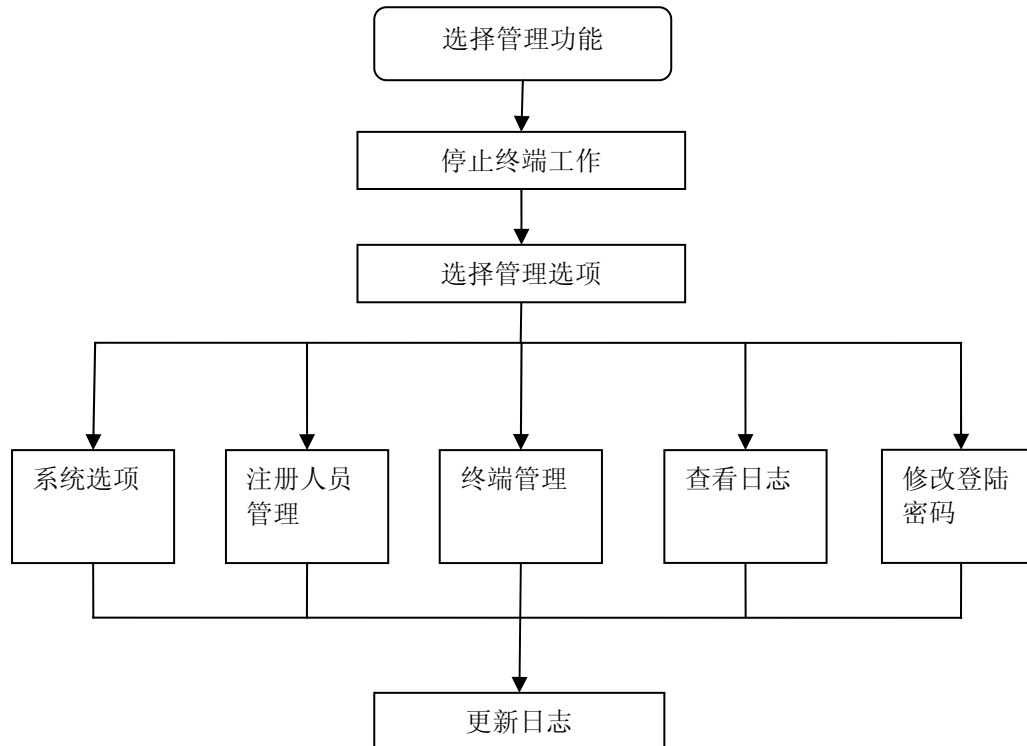
主机:



运行模块:



管理模块：



五. 设计方法

1. 系统开发流程

本系统大部分设计功能是采用Altera的DE2开发板、外围指纹采集电路的电路实现。我们先实现系统各功能模块，最后总装调试。

1. 参照 DE2 开发板的部分实例和试验文档，在 DE2 开发板上实现了 Nios 软核构建，IDE 环境下的 C 语言编程以及程序的在线调试和烧录。
2. 在 DE2 开发板上利用 SOPC Builder 实现了对外围存储器，RS232 串口，DM9000A 网络接口，PIO 接口的访问。
3. 在 DE2 开发板上实现了通过扩展 PIO 口对 4×4 键盘的访问。
4. 在 DE2 开发板上实现了通过 SPI 核对指纹采集数据的读取。
5. 完成指纹处理算法，在 DE2 开发板上实现了自定义外设和自定义指令硬件加速。
6. 利用购买的指纹采集芯片自行设计并实现了指纹采集电路，实现自动检测手指功能。

与此同时开发PC平台的管理员程序，最后联机调试。

2. 硬件部分

Nios软核的配置:

- 1) 确定采用 DE2 开发板上的 Cyclone® II EP2C35 FPGA 作为主控芯片，所有的功能都是围绕着这块芯片来进行设计的，从硬件上体现了 SOPC 高度集成的设计思想和设计原则。
 - 2) 确定 50MHz Nios II CPU Fast 版，支持 JTAG level 3。
 - 3) 12.5MHz SPI 核，完成指纹数据传输。
 - 4) 加入 DM9000A 以太网控制芯片，实现以太网物理层功能。
 - 5) 加入 jtag 调试模块，以利于系统的在线调试。
 - 6) 串口通信模块采用 38400bps。利用 uart 我们可以在网络通信还未实现的情况下完成 Nios 系统和 PC 机的数据传输，进而在 PC 机上监测指纹图像处理的过程。
 - 7) SRAM, SDRAM 和 FLASH 分别被用来实现程序运行时的内存，和数据的分配空间和程序烧写的空间。
 - 8) 存储器之间三态桥的连接。
 - 9) 两个定时器的，可以实现系统中对延时的要求，同时可以测试 Nios 系统处理指纹数据所需要的时间。
 - 10) 按键、数码管、LED 和 16×2 字符液晶接口可以很好的实现人机交互。
- 采用 SOPC Builder，我们能够方便的组建一个可裁减、可配置的系统，实现需要的功能。

指纹采集器的制作:

- 11) 功能设计
使用 fps200 指纹采集芯片采集指纹原始图像数据。该电路必须具有自动手指检测功能，在需要采集指纹且检测到手指时才会通知 CPU 取数据，然后图像数据通过 SPI 接口传送到 Nios II 处理器中。
- 12) 原理图设计。
在认真研究指纹采集芯片的 datasheet 之后，我们设计出原理图，实现了 SPI 接口，并预留 MCU 和 USB 接口备用。
- 13) PCB 图的制作。
最终设计的电路板采用双面 pcb 板方案。
- 14) 电路板的焊接调试。
实验验证了指纹采集器的功能，数据采集和传输正确，最后调整指纹采集相关的参数。

指纹图像处理硬件加速:

在Nios CPU配置为Fast，并且加入fpoint浮点运算指令的情况下，使用C语言实现的图像处理算法仍然耗时50s以上（参见的第三部分“性能参数”），对于一个实时处理的系统是不能承受的。

提高图像处理效率的方法有两种：一是在FPGA上实现一个DSP处理模块，二是采用针对图像处理的硬件加速。比较这两种方案，DSP实现图像处理效率更高，且不依赖CPU的运算速度，但实现难度很大；图像处理硬件加速相对容易实现，而下面我们将会看到采用硬件加速大幅提高算法执行效率是可行的。

指纹图像处理算法有其自身的特点：存在大量重复性的运算，且对每个像素的处理过程是相同的。这样我们只要改进这些最基本运算的执行效率，就能提高整体执行效率。我们将整个指纹处理过程划分为方向图求取、图像滤波、二值化、纹线细化、特征点定位等子过程（具体原理参考下面“软件部分”），分别统计每个子过程的耗时，确定了两个必须加速的子过程，即“图像滤波”（36.4s）和“纹线细化”（13.5s）。

图像滤波原理简述：取目标像素周围52个像素数据，分别乘以对应的滤波系数最后累加作为目标点新值，滤波系数从一个方向相关的系数阵列中取出。纯软件方法需要作52次乘法累加，针对图像滤波运算，我们设计了一条自定义指令CI_multi_accumilate，三个时钟周期完成一次乘法累加。加速后图像滤波子过程耗时缩减到4.77s。

纹线细化原理简述：取目标像素连同周围15个像素数据，与16个模板比较，确定目标像素是否清除。经过若干次全图扫描，指纹纹线被细化为单像素宽度。纯软件方法需要作16次比较操作，针对纹线细化运算，我们设计了两个自定义外设prematch和user_delete，共计六个时钟周期完成16次比较。加速后纹线细化子过程缩减到2.67s。

以太网实现方法：

系统以交换机为中心节点进行组网，建立各个终端与主机的连接。由于终端与主机的通信数据量并不是很大，没有复杂的路由，系统没有也不允许与公网有物理连接，但是如果要实现上层的IP协议，就需要使用嵌入式实时操作系统，而我们的从机系统没有必要使用操作系统来进行调度，所以没有采用TCP/IP协议进行组网，而直接采用了MAC---MAC的方式，采用物理地址寻址，即直接发送加入源和目标MAC地址为报头的原始数据包。

3. 软件部分

整体设计：

软件设计包括Nios平台程序和PC平台程序两部分。

Nios平台程序可以划分为初始化、指纹采集、图像处理、主从通信、人机交互等模块，通过这些模块的有机组合，我们完成了一个简单的指纹认证客户端程序。

1) 系统初始化

系统开机之后，必须对系统的各个模块进行初始化，其中比较重要的是对网卡的初始化，让 Nios 能够跟 pc 机建立连接；对指纹采集器的初始化让指纹采集器处于低功耗模式，需要采集指纹时才唤醒。

2) 指纹采集

在驱动程序的基础上，通过 SPI 接口控制指纹采集器采集图像。需要完成手指自动检测和手指湿度自适应。

3) 指纹处理

这一模块主要完成指纹数据处理和比对。包括两个子模块：指纹图像处理，指纹比对。

4) 主从通信

这一部分实现 Nios 从机和 PC 的通信。发送主要的主要内容是：发送各种请求命令，如：注册、登陆请求，发送指纹的特征信息，发送 ID 的信息等。接收的主要内容是：各种请求 PC 机所给出的回复信息，PC 机发送的对 Nios 的控制信息，PC 机返回给 Nios 的指纹模板信息等。

5) 人机交互模块

这部分主要由 4×4 键盘、16×2 液晶、LED、数码管组成。其中 LED 可以用作指示现在 Nios 系统的工作状态，还可以提供成功、失败、超时等信息，数码管主要用于显示输入的 ID 信息，液晶用于显示系统实时的信息，包括系统状态和操作提示。

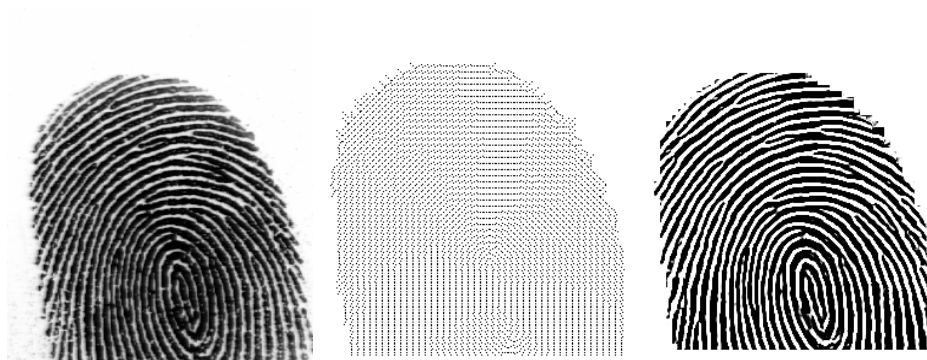
PC平台程序是指纹认证管理员程序，完成登录/注册请求的响应和系统管理等功能。

由于局域网中传输的是加入源和目标MAC地址为报头的原始数据包，我们应用了winpcap协议。winpcap(windows packet capture)是windows平台下一个免费、公共的网络访问系统。开发winpcap这个项目的目的在于为win32应用程序提供访问网络底层的能力。它提供了以下的各项功能：1> 捕获原始数据报，包括在共享网络上各主机发送/接收的以及相互之间交换的数据报；2> 在数据报发往应用程序之前，按照自定义的规则将某些特殊的数据报过滤掉；3> 在网络上发送原始的数据报；4> 收集网络通信过程中的统计信息。通过试验与最后作品的证明，它能很好的实现我们的功能需求，很方便的实现了PC与Nios的收发数据包的功能。

指纹图像处理模块：

整个指纹处理过程划分为方向图求取、图像滤波、二值化、纹线细化、特征点定位等子过程：

(更详细的原理参见参考文献【1】【2】)



原始图像

方向图

图像滤波



二值化

纹线细化

特征点定位

(1) 方向图求取

分两个步骤：一是根据目标点周围点的灰度值计算出单点方向，为简化计算分 180° 为8个方向；二是用统计法得到 5×5 方块的块方向图，对于没有明显方向的块标记为背景。

方向图求取为下面基于方向的图像滤波打下基础。

(2) 图像滤波

预先设计好针对8个方向的滤波系数模板，取目标像素周围52个像素数据，分别乘以对应的滤波系数最后累加作为目标点新值。

滤波的作用是加强图像沿纹线方向的连续性，同时提高垂直纹线方向的对对比度以分割相邻纹线。

(3) 二值化

经过滤波后的图像纹线清晰，所以简单的采用固定阈值二值化法，即以某一固定灰度值作为标准将图像分割为黑白两色图像。

只有经过二值化的图像才能进行完全的细化。

(4) 纹线细化

采用改进的OPTA法，逐步腐蚀纹线，直到纹线成为单像素宽度。

细化是为了方便下一步特征点定位。

(5) 特征点定位

首先扫描方向图，定位指纹中心点。然后由细化后的纹线图定位末梢点和分叉点，判断方法：末梢点为黑点，且周围只有一个黑点，分叉点周围有三个黑点。

指纹比对：

我们经过多次实验，改进了基于中心点的指纹比对方法，分粗匹配、坐标变换、精匹配三步：

(1) 粗匹配

两幅指纹图像分别以各自的中心点为原点， $\pm 30^\circ$ 的旋转角范围进行粗匹配，确定匹配程度最高的旋转角，同时获得部分特征点对。

(2) 坐标变换

根据已经获得的特征点对，计算两幅图像精确的坐标平移和旋转参数。

(3) 精匹配

由得到的坐标平移和旋转参数重新精确匹配特征点，并评价匹配程度。类型、位置都匹配的点对加3分，只有位置匹配的点对加2分。总评分超过阈值认为匹配成功。而以上三步任何一步失败，则认为匹配失败。

基于中心点的指纹比对方法可以有效的抵抗图像平移的干扰，而经过我们改进的方法有新的特点：最后的精匹配结合了基于点对点的比对方法，一定程度的抵抗了中心点定位误差带来的干扰；考虑了特征点类型（末梢或分叉）常常不准确的事实，认为类型不匹配而位置匹配也属于匹配的情况，提高了比对的准确率。

六. 设计特点

1. 主机和从机之间总线型的拓扑结构

考虑到本系统应用的单位和机构会需要在多个地点来进行指纹身份的识别，而如果在每个地点都配置一个系统是不切合实际的也是不允许的，因此我们想到运用主机和从机的模式来完成这一任务，从机用来完成指纹采集处理，将处理结果通过网络传输给主机，主机来完成管理和存储。这样主从机分工明确，发挥各自的优势。

2. 用户自定义指令实现关键算法的硬件加速

由于指纹特征的提取是一个很复杂的数字信号处理，如果仍用软件的方式去实现的话必然会大大降低系统的速度。恰恰好的是Altera公司提供的SOPC Builder不仅可以创建和配置用户的Nios,还可以添加自定义用户指令。正是基于这一点，我们在系统中采用硬件描述语言来实现指纹特征提取的算法，运用自定义指令方式将该算法IP定义成特殊指令，供系统直接的调用，实现硬件加速，进而达到系统速度大大提升的目的。

3. Ethernet 网络传输：

采用Ethernet网络传输每一个从机的请求，交给主机进行处理。同时我们采用终端竞争访问的机制，这种方法易于实现并且在系统负荷不是很高的情况下有比较高的效率。Ethernet网络为系统的实际应用提供了保证，同时系统以后继续添加终端也很方便。

4. 系统的软硬件升级便捷:

虽然一个实用系统所面对的用户群是庞大的,但是它仍然不可能完全符合每一个具体的用户。而采用Altera的SOPC系统设计方案却可以解决这一问题,针对不同的用户可以轻松的在原有的基础上实现软硬件的更改,加上FPGA的可编程性,可以在很短的时间内就可以开发出新的产品,提高了产品的竞争力。在本系统中我们完全可以根据用户终端数量的多少来决定是将指纹特征提取模块放在终端还是主机,这样可以保证通信量和主机负担的权衡,因此系统的适应性大大加强。

5. 系统的成本, 功耗, 便携性, 集成性良好:

基于FPGA的系统设计,将处理器、外设、存储器,和I/O接口集成到一个单一的FPGA中,从而降低了系统的成本、复杂性和功耗。同时Nios软核在成本等方面都优于硬核,这也给系统的集成性和成本性方面的性能大大的提高。

七. 总结

两个月的竞赛,是一个对 SOPC 开发、学习和应用的过程,是对 Nios 嵌入式设计逐渐熟悉的过程。我们作为此开发设计技术的使用者,对这项技术也有了更加深刻的认识。

1. 清晰良好的设计环境。在两个月的时间完成开发软、硬件基础的学习,进行系统设计、实现及验证,这对我们是一场挑战。包含在 Quartus II 开发工具中的 SOPC Builder 提供了一种可视化的快速架构方法,以及 Nios IDE 集成开发环境,方便了我们短期目标的实现,大大减化了我们学习的进程。
2. 软件、硬件协同设计和验证。软、硬件并行设计是嵌入式系统设计的一项关键任务。在设计过程中的主要问题是软、硬件设计的同步与集成。系统的周密设计,软、硬件的合理划分, SOPC Builder 使我们能相对独立地进行软、硬件的实现,各自独立的功能验证也缩短了开发周期。
3. Nios 系统硬件加速方案是一个创举,相对于传统的 DSP 实现方案,自定义指令或自定义外设能更好的融入整个系统,使 NIOS 在保持控制灵活的优势下处理速度不占下风,而相对 ASIC 实现方案, Nios 的成本优势体现出来,无需外加芯片,只用 FPGA 内部的资源也可实现同样的功能。
4. Nios 系统的稳定性有所欠缺,在设计小系统时或许体现不出来,但对于一个复杂的系统,问题暴露出来。表现是同样的硬件配置,软件运行随机性出错,重新编译布线后问题可能消除,而出错部分往往是底层 I/O 驱动。推测原因 avalon 总线不是固定的连线,而是 Quartus 编译后在 FPGA 内生成的一系列连线的集合,布线的随机性使总线不稳定。推测没有验证,不过这个问题曾经让我们相当的头疼。
5. 许多问题现在回头看来其实是很简单的,但是当第一次遇到它,尤其在资料很少没有任何头绪时,似乎成为不可逾越的障碍。需要我们沉下心来努力思索,广泛地收集任何可能有帮助的信息,并且要敢于大胆的尝试,更重要的是要有持久的耐心,往往是山穷水尽疑无路,柳暗花明又一村。虽然最后发现解决问题绕了很大的圈子,但是通过这个过程,我们不仅是学到了知识,更重要的是学到了解决问题的方法,这对我们以后的学习生活无疑是最大的一笔财富。

参考文献:

- 【1】** 刘灵丽,《指纹图像预处理与特征提取》湖南大学硕士学位论文, 2005年12月
- 【2】** 李春雷,《指纹识别算法的研究及基于FPGA的硬件实现》山东大学硕士学文论文, 2005年4月