

原创性声明

郑重声明：此篇题为《基于 SOPC 的嵌入式文字识别系统设计》的论文是作者在导师的指导下，于武汉大学攻读硕士学位期间，进行研究工作所取得的成果。根据作者所知，论文中除了参考文献列举的地方外，不包含他人已经发表或撰写过的研究成果。

作者签名：李梦竹 潘超

导师签名：黄启俊 常胜

撰写日期：二零一一年七月七日

基于 SOPC 的嵌入式文字识别系统设计

作者：李梦竹 潘超

导师：黄启俊，常胜

(武汉大学物理科学与技术学院电子科学与技术系，武汉 430072)

摘要：本文设计了一种基于SOPC的嵌入式文字识别系统。在FPGA平台下，基于SOPC的框架搭建软硬件协同系统，设计硬件电路完成文字图像的采集和预处理，嵌入Linux系统，使用其下的识别引擎完成文字图像的识别。系统采用Altera公司的SOPC builder构建系统框架，Quartus II完成硬件电路的设计，在宿主机Linux环境下完成了软件部分的交叉编译并嵌入到FPGA平台。整体设计在DE2-70开发板上完成了系统验证。通过合理的软硬件划分，充分利用了硬件处理速度快和软件识别算法成熟的优势，使得本设计具有便携性好、实时性强的特点。此外，本文所使用的在SOPC平台上嵌入linux系统的方法对于扩展FPGA器件的应用方法和应用领域也具有一定参考价值。

关键词：文字识别；可编程逻辑器件（FPGA）；可编程片上系统（SOPC）；Linux

An SOPC Design of Character Recognition System

Authors: Li Mengzhu, Pan Chao

Tutors: Huang Qijun, Chang Sheng

(Department of Electronic Science and Technology, School of Physics and Technology, Wuhan University, Wuhan, 430072)

Abstract: This paper introduces a design of an embedded character recognition system based on SOPC, which is using hardware circuit to complete image collection and processing while using character recognition engine under Linux to implement character recognition on FPGA platform, and setting up the hardware and software co-working system based on SOPC. The whole design uses SOPC builder which is provided by Altera to complete SOPC design, and uses Quartus II to finish hardware development, and finishes software part by using cross-compile in host Linux environment which is embedded into FPGA. At last, software and hardware is downloaded in Altera DE2-70 development board to finish the function validation of the whole system. Through the rational division of hardware and software, taking full advantage of hardware processing speed and software to recognize the advantages of sophisticated algorithms, makes the design portable, good, strong real-time. In addition, the method of SOPC platform embedded with Linux has a certain value for application of extension for FPGA.

Keywords: Character recognition; FPGA; SOPC; Linux

一、引言

计算机文字识别也被称为光学文字识别^[1], 英文名称为 Optical Character Recognition(OCR), 在智能计算机和办公自动化领域有着极其重要的应用。文字识别的基本原理是通过诸如照相机、扫描仪等图像输入设备获取文字图片, 经过图像处理使用光学模式判别等算法分析文字图片, 最后将判断出的文字编码储存起来从而完成文字识别。

文字识别设备对识别速度的要求较高, 因此图像采集和预处理的速度十分关键。同时识别率要求高, 识别字体种类要多, 因此选用带学习功能的软件引擎非常重要。另外系统需要交互界面和大量外围设备的驱动, 因此需要引入操作系统方便设计开发。目前现有的文字识别解决方案主要分为以下几种:

- 1、通过软件对文字图片进行文字识别。目前无论在 Windows 下还是在 Linux 下都有不少文字识别软件, 比较著名的有清华紫光 OCR、Mini OCR 和汉王 OCR, 微软的 office 软件里面也有内置 OCR 的插件, 可以将图像转化成文字。通过软件进行文字识别可选的软件资源丰富, 操作方便。缺点是缺少文字图片的处理, 在图像质量较差的情况下, 文字识别率无法得到保证, 同时使用软件进行文字识别缺乏便携性。

- 2、通过便携式文字识别产品进行文字识别。此类文字识别产品具有图像捕

捉模块，通过摄像头或者扫描仪作为文字图片输入媒介，然后进行文字识别，最后将识别的文字存入 SD 卡等存储设备。国内便携式文字识别产品有汉王扫描仪，便携式文字识别产品的文字识别率高，缺点是产品功能较为单一，操作性较差。

考虑到以上方案存在的问题，本文设计了一种基于 SOPC 的软硬件相结合的便携式文字识别系统。针对图像采集和预处理中存在的大量数据流，用硬件模块来实现图像的采集和预处理，以保证系统的处理速度。使用 Nios 嵌入式处理器作为系统的主控，以满足对数据流较多操控的需求。使用嵌入式 Linux 下的文字识别引擎以确保文字识别的质量。从以上系统框架可以看出，本文在 SOPC 的概念下，通过合理的软硬件划分（硬件完成图像算法处理，软件进行文字识别），发挥了软硬件处理各自的优点。

二、SOPC 系统设计

2.1 系统功能描述

目前人们对文字识别的需求是操作方便智能、便携性好、文字识别率高、识别速度快。因此，本设计主要实现的功能如下：

- 1、图像采集功能，通过外接的摄像头模块完成图像的采集；
- 2、由硬件模块完成文字图像预处理功能，能够对采集的文字图像进行倾斜纠正、图像分割、二值化等处理，保证系统的性能和速度，为后续的文字识别提高识别率；
- 3、文字图像识别功能，能够将文字图像中文字识别出来并保存在 TXT 中；
- 4、识别结果的存储和发送功能，能够将识别结果文件存入 SD 卡或 U 盘中，或者通过网络发送给上位机；
- 5、交互界面功能，能够通过 LCD 模块或者外接的 VGA 设备作为输出设备，通过键盘、触摸屏或者鼠标作为输入设备和使用者交互。

由以上功能需求提出本设计的整体方案如图 1 所示，整个系统由一块 FPGA 和摄像采集、输入输出等外围设备组成。FPGA 硬件模块完成摄像采集控制和图像预处理，CPU 作为系统主控控制外设和硬件模块的数据流。文字识别模块和外围设备驱动在 Linux 平台上进行开发或移植。

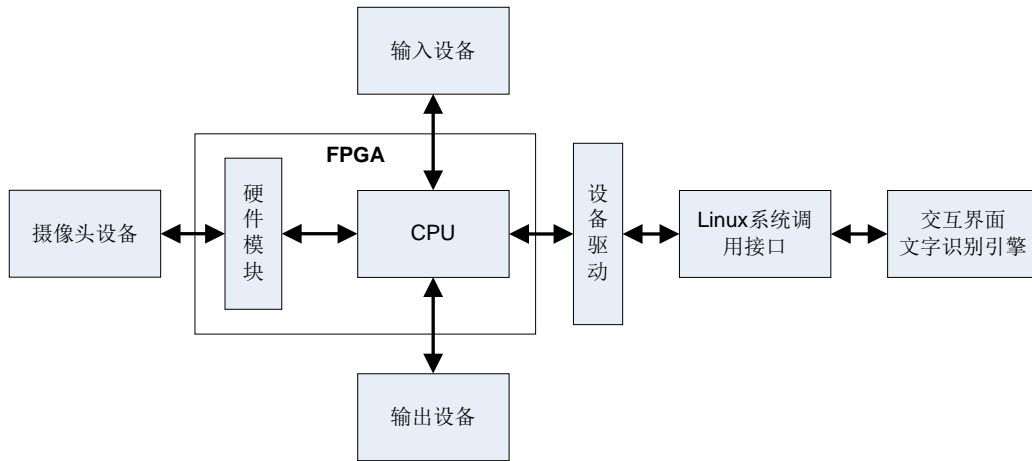


图 1 文字识别系统整体结构框图

2.2 主要技术指标

主控参数：Nios II 处理器，工作频率 100MHZ。

系统内存：32MByte SDRAM，8MByte Flash，2MByte SRAM。

Linux 内核：Linux2.6

显示参数：分辨率 640*80，16 位色（RGB565）。

摄像头参数：最大支持 1280*1024 分辨率，RGB888。

图片参数：JPG 和 BMP 格式等常用图片格式。

文本参数：TXT 格式。

网络参数：10M-100M 的自适应。

文字识别引擎：Tesseract 开源引擎，支持引擎训练。

2.3 系统整体结构设计

本文采用 Altera Cyclone2 系列的 FPGA，通过 Altera 提供的 SOPC 技术，可以将 Nios II 软核处理器和外围设备接口 IP 通过 Avalon 总线连接起来，并集成在一块 FPGA 上。图 2 是整个系统的 SOPC 设计：

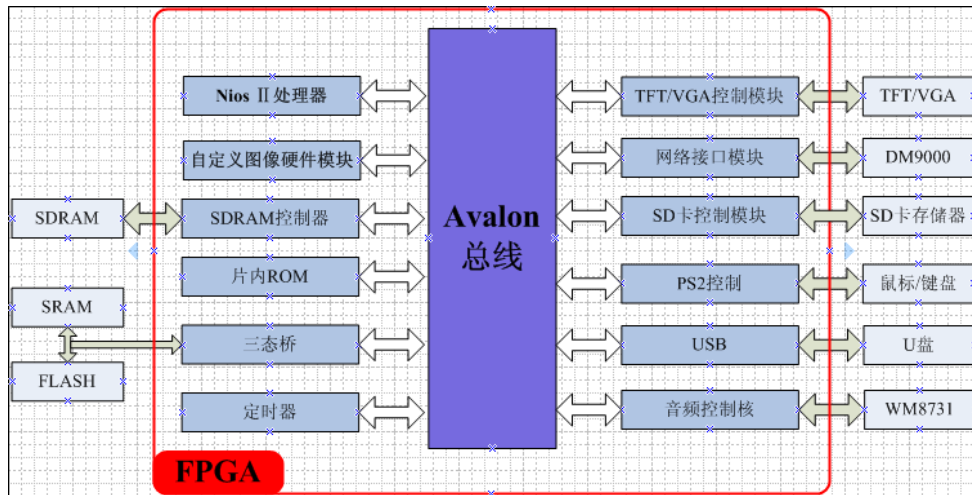


图 2 文字识别系统 SoPC 的设计

从图中可以看出，本 SoPC 系统的设计中除了加入了 Nios II 处理器外，还加入了 SRAM、SDRAM、FLASH 等存储设备接口。其中选用 SDRAM 作为系统的程序存储器，将两片 SDRAM 的地址线、读写控制信号线、片选线接在一块，数据线并联，可以将外围的两块 16 位 SDRAM 用 1 个 32 位的 SDRAM IP 核控制，从而提高了数据吞吐量，提高了系统性能。

定时器的加入可以给系统提供计时功能，是后续 Linux 系统移植中必要的组成部分。

外围设备接口方面，本设计加入了 VGA、SPI、DM9000 等 IP 核作为系统的输出设备接口，PS2、USB 等 IP 核作为系统的输入设备接口。其中 VGA 接口是系统的显示输出接口，采用 Altera 的 VGA IP 核作为 SoPC 的系统组件，VGA IP 核采用 DMA 技术，在需要显示的时候 CPU 放开总线，由 VGA IP 核直接读写 SDRAM 里面开辟的 framebuffer，从而提高了显示效率。另外本设计添加了 USB 芯片 IP 核、网卡 IP 核、PS2 IP 核。这三个 IP 核都是将芯片输入输出转接成符合 Avalon 总线协议的信号，从而集成进 SoPC 系统中。外围设备的驱动程序在 Linux 中实现，通过 CPU 控制外围设备的工作。图 3 是整个 SoPC 系统图：

Use	Connections	Module Name	Description	Clock	Base	End	IRQ
☑		cpu_0	Nios II Processor	clk			
		instruction_master	Avalon Memory Mapped Master				IRQ 0
		data_master	Avalon Memory Mapped Master				IRQ 31
		jtag_debug_module	Avalon Memory Mapped Slave		0x05001000	0x050017ff	
☑		cfi_flash_0	Flash Memory (CFI)	clk	0x04800000	0x04ffffff	
		sdram_0	SDRAM Controller	clk	0x02000000	0x03ffffff	
		s1	Avalon Memory Mapped Slave	clk	0x02000000	0x03ffffff	
☑		vga_controller_0	VGA Controller	clk	0x05002080	0x0500208f	
		s1	Avalon Memory Mapped Slave	clk	0x05002080	0x0500208f	
		m1	Avalon Memory Mapped Master	clk	0x05002080	0x0500208f	
☑		Audio_0	AUDIO_DAC_FIFO	clk	0x050020d0	0x050020d3	
		avalon_slave_0	Avalon Memory Mapped Slave	clk	0x050020d0	0x050020d3	
☑		ps2_0	PS2 Serial Port	clk	0x050020b8	0x050020bf	3
		avalon_PS2_slave	Avalon Memory Mapped Slave	clk	0x050020b8	0x050020bf	
☑		cmos0	CMOS Slave Controller	clk_50	0x050020a0	0x050020af	
		s1	Avalon Memory Mapped Slave	clk_50	0x050020a0	0x050020af	
☑		ISP1362	ISP1362	clk	0x05002090	0x0500209f	2
		avalon_slave_0	Avalon Memory Mapped Slave	clk	0x05002090	0x0500209f	
		avalon_slave_1	Avalon Memory Mapped Slave	clk	0x050020d4	0x050020d7	
☑		mmc_spi	SPI (3 Wire Serial)	clk	0x05002060	0x0500207f	5
		s1	Avalon Memory Mapped Slave	clk	0x05002060	0x0500207f	
☑		dms9000	DMS9000A	clk	0x050020c0	0x050020c7	6
		avalon_slave	Avalon Memory Mapped Slave	clk	0x050020c0	0x050020c7	
☑		display_adjust_0	display_adjust				7

图 3 SOPC 系统图

三、系统硬件设计

文字识别硬件设计主要包括文字图像的采集控制、文字图像的预处理。文字图像的预处理又分为边缘检测、倾斜纠正、文字区域提取、二值化。图像预处理需要大量的数据流的控制和中间结果的存储，而 FPGA 具有密度高、容量大、体积小，易于用在便携式设备中，可满足高速系统设计等特点，因此用硬件来实现图像的预处理，完全符合设计的需求。本设计引入 CPU 作为主控控制图像流的传输。因此在模块外部设计了 Avalon 接口以便接入 SOPC 系统中，同时利用片内和片外的 RAM 资源存储处理中的中间信息。硬件模块总体设计图如图 4：

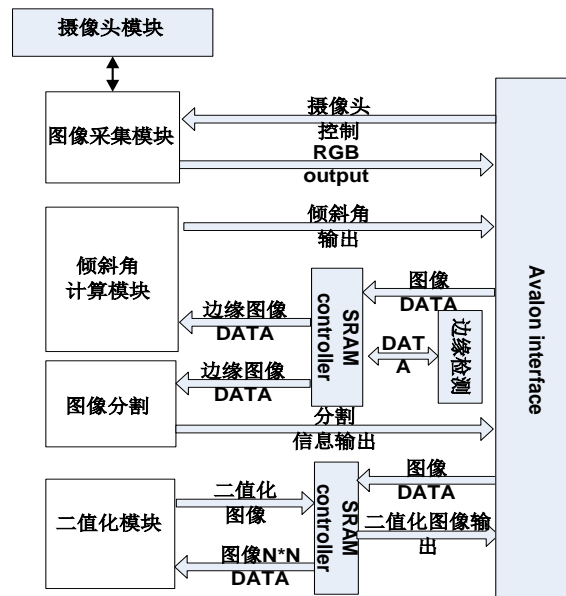


图 4 文字识别系统硬件部分总体框图

3.1 图像采集的硬件设计

图像采集模块是文字识别系统重要的组成环节，图像采集质量的好坏不仅影响后端文字识别的准确率，而且也决定文字预处理难度。

本设计图像采集设备采用 Terasic 公司提供的 TRDB-D5M 摄像头套装。TRDB-D5M 摄像头具有 500W 像素，支持最大 1280x1024 RGB888 图像输出。TRDM-D5M 可通过 GPIO 与 FPGA 相连，传输数据并控制摄像头的采集、曝光。TRDM-D5M 支持手动调焦，可以避免在不同距离下图像质量出现较大的差别。图像采集模块的框图如图 5：

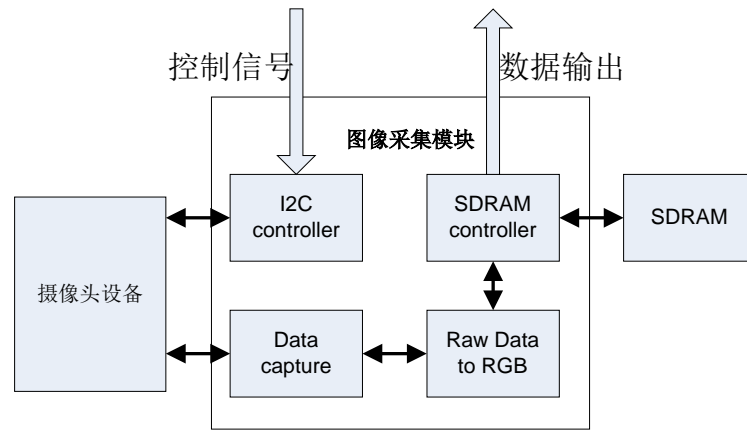


图 5 图像采集模块的硬件框图

TRDB-D5M 摄像头具有 I2C 接口，可以通过 I2C 协议对摄像头进行控制。本设计的图像采集模块内置 I2C 控制器，通过 I2C 协议对摄像头进行控制，可以控制摄像头的曝光、图像采集的开始和停止。摄像头传回的数据进行简单的转换可以变成 RGB888 的数据，通过 SDRAM 控制器存入 SDRAM 进行暂存。本设计中 SDRAM 存满一帧数据后数据才向外输出，保证了图像的完整性。

3.2 文字识别的图像预处理设计

对于待识别的文字图像，影响文字识别准确率的不仅仅是后端引擎的性能，更重要的是文字图像的质量。图 6(a)是理想的文字输入图像。整副图片只有黑白两色、无倾斜。图 6(b)是实际使用中获取的文字图像，除了光照不均匀且有倾斜外，图片中还有不少干扰信息。因此，本设计选取倾斜纠正、文字图像分割、二值化等预处理方法来减少干扰因素，保证后续文字识别的质量。

This is a lot of 12 point text to test ocr code and see if it works on a different file format.
The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox. The quick brown dog jumped over the lazy fox.

图 6(a) 理想的文字输入图像

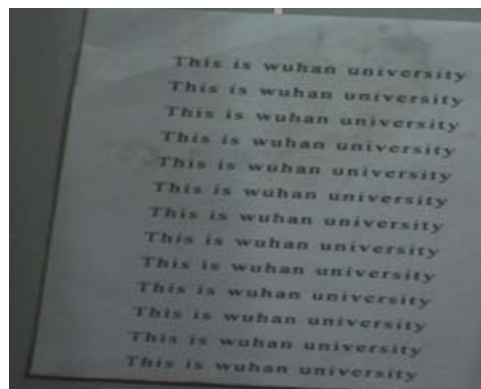


图 6(b) 实际拍摄的文字图像

3.2.1 文字图像的边缘检测

对图像进行边缘检测可以大致区分文字信息区域和背景区域。本设计后续的倾斜角计算和图像分割都是基于边缘检测后的图像进行的。

图像边缘就是图像中灰度发生急剧变化的地方，目前边缘检测中多采用边缘检测算子分析图像边缘的突变。本设计不需要较为复杂的边缘检测算子，采用一阶导数的 Sobel 算子的边缘检测算法已经可以满足设计的需要。

本设计中将最后得到的梯度值简化为水平方向梯度绝对值和垂直方向梯度绝对值之和。求出梯度后采用基本全局门限：当某像素点 (x, y) 的梯度值大于或等于设定的门限 T 时，规定该点的像素值为 1，反之则为 0。

3.2.2 文字图像的倾斜纠正

文字图像倾斜纠正包括倾斜角计算和图像旋转。就是将拍摄中获取的倾斜图像根据倾斜角度进行旋转，最后得到校正后的图像。目前主要采用四种方法来估算文档图像的倾斜角：Hough 变换、投影特性法、傅立叶变换和行间距跨度相关法^[2]。

考虑到基于 Hough 变换的倾斜角计算方法对图像的噪声不敏感，同时算法结构较简单，适合硬件实现，本设计采用基于 Hough 变换来计算图像的倾斜角。Hough 变换计算倾斜角的基本原理是通过找出图像中通过最多点的直线，即为图像的倾斜角。Hough 变换中，将二维图像中的点通过极坐标系表示，而平面坐标系中一条直线上的所有点对应的极坐标系曲线交于极坐标系中唯一一点 (ρ, θ) ，因此，只需要寻找极坐标系中最大值即可。

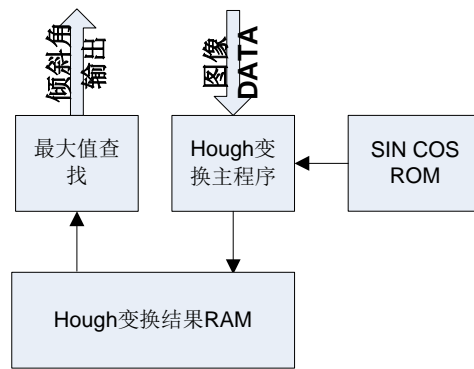


图 7 倾斜角计算模块的硬件设计

图 7 是倾斜角计算模块的硬件设计框图。Hough 变换主程序接收经过边缘处理的图像数据，根据图像的像素位置进行 Hough 变换，算出 (ρ, θ) ，将 RAM (ρ, θ) 地址加 1。为了加快 Hough 变换的速度，本设计中将 SIN 和 COS 数据放大一定的倍数取整数放入 ROM 中，计算是可以直接调用这些数据。最后，最大值查找模块读取 RAM 中的结果，将最大值找到，所对应的地址 (ρ, θ) 中 θ 值就是倾斜角度。本设计采用流水线技术，数据的输入和中间的计算可以并行操作，提高了计算的速度。

3.2.3 文字图像的分割

拍摄的文字图像不仅包含着文字信息，也会或多或少的掺入一些背景或其他不相关的干扰信息。因此，文字图像的分割可以提取文字信息区域以减少背景等干扰，同时，由于图像区域的减少，对于光照不均匀也有一定的改善。目前图像分割算法主要有：Otsu 大津阈值分割法、边缘检测分割法、基于神经网络的分割法^[3]。

本设计根据文字图像的具体特点设计分割方法，首先将经过倾斜纠正的文字图像通过边缘检测算法得到边缘检测图片。然后通过逐行和逐列扫描确定行和列的阈值。接着对所有符合阈值的区域进行分析，相隔较近的区域进行联通，从而确定最终的分割区域。最后根据分割区域对原图进行图像分割，分割的总体流程如图 8。

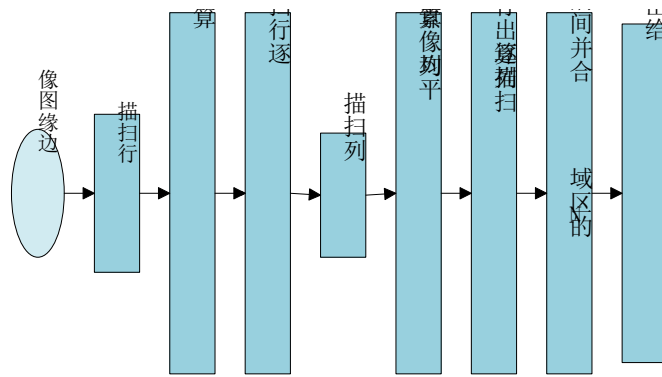


图 8 图像分割的硬件处理流程

3.2.4 文字图像二值化

经过倾斜纠正和图像分割后得到的文字图像仍有光照不均匀和噪声的影响，因此需要进行二值化处理。对图像进行二值化不仅能提高文字识别的精度，对后端引擎的识别速度也有不少的帮助。目前常用的图像二值化方法有：基于全局阈值的二值化、基于局部阈值的二值化、全局和局部阈值混合二值化^[4]。

考虑到实际拍摄的图片光照不均匀的程度较高，使用全局或者混合二值化的效果都不理想，且考虑到硬件实现等因素，本设计采取局部阈值二值化。具体的模块设计图如图 9：

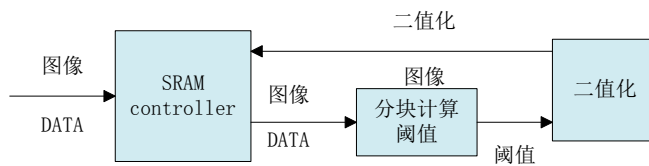


图 9 图像二值化硬件模块

四、系统软件设计

本设计软件部分基于嵌入式 Linux。首先在 Nios II 处理器中植入嵌入式 Linux；然后在 Linux 内核里配置外围设备驱动并加入一些自定义的设备驱动；最后基于 Linux 移植或者开发交互界面、文字识别引擎等。另外，还需要移植 bootloader，系统起电后，bootloader 将 Linux 内核搬到 SDRAM 中运行，从而实现系统开机自启动的功能。

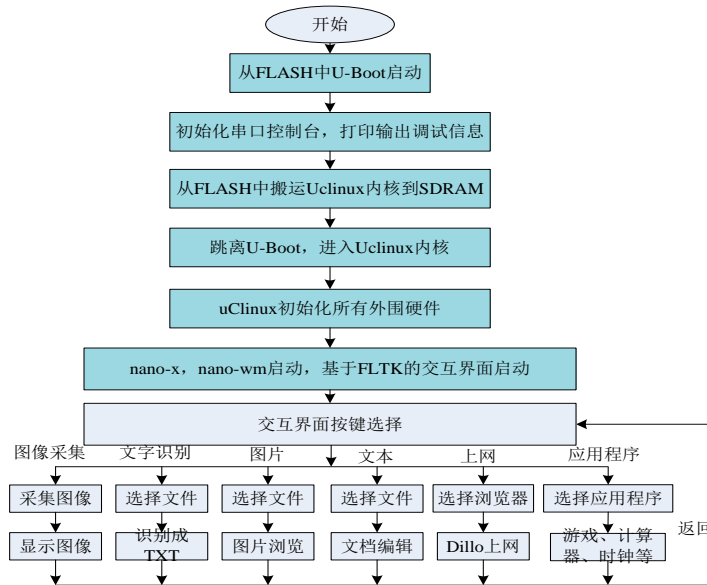


图 10 软件流程图

如图 10 所示为软件整体流程图，将 bootloader 和 Linux 内核下载到 Flash，同时设定复位地址为 Flash 中内核 bootloader 的入口地址。系统启动时 bootloader 先启动，初始化外围设备，并通过串口打印输出信息，同时将 Linux 内核搬运到 SDRAM 进行解压。Linux 内核解压完成后内核启动。只是 Linux 初始化所有的外围硬件设备，启动 nano-X 图形接口和 nano-wm 窗口管理器和基于 FLTK 的交互界面。交互界面上是各种的按钮选择，包括图片浏览、图像采集、文本浏览编辑、网络连接及文字识别等应用程序的按钮。当按键选择各种功能后会进入相应的功能模块。当程序运行完成后通过选择返回按钮再次进入到交互界面的按键选择部分。

由于 Linux 强大的任务调度功能，本设计支持多任务浏览和切换。同时交互界面的引入使得操作十分方便。本设计的交互式界面如图 11：

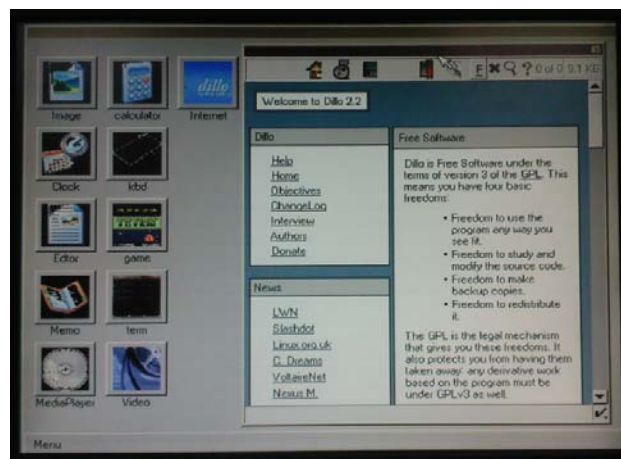


图 11 交互式界面的设计

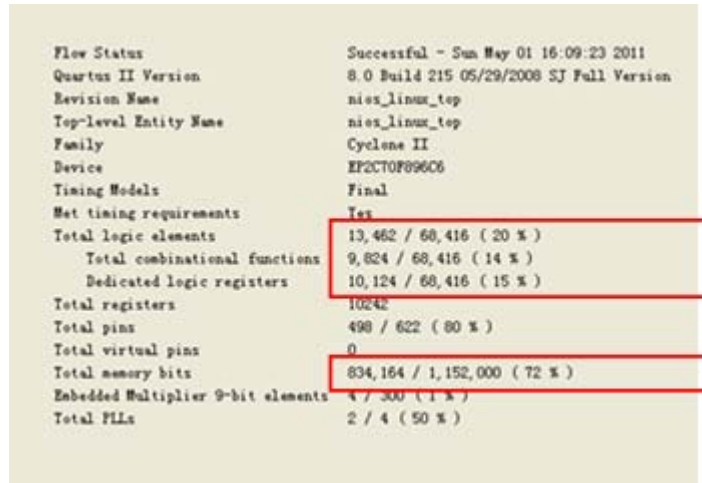
五、 验证与讨论

本设计使用基于 EP2C70F896C6 的 DE2-70 开发板为验证平台。DE2-70 拥有 2 块 32Mbyte 的 SDRAM 作为程序存储器，外围还有 8Mbyte Flash 芯片以及 1Mbyte 的 SRAM 作为存储芯片且拥有诸如 USB、PS2、VGA、SD 卡口、网卡等丰富的外围设备和接口，非常适合本设计的验证。

5.1 系统实现

5.1.1 硬件综合结果

本系统硬件代码通过 Quartus 编译综合最终下载进入 FPGA 实现，其中硬件使用情况如下图：



Flow Status	Successful - Sun May 01 16:09:23 2011
Quartus II Version	8.0 Build 215 05/29/2008 SJ Full Version
Revision Name	nies_linux_top
Top-level Entity Name	nies_linux_top
Family	Cyclone II
Device	EP2C70F896C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	13,462 / 68,416 (20 %)
Total combinational functions	9,824 / 68,416 (14 %)
Dedicated logic registers	10,124 / 68,416 (15 %)
Total registers	10242
Total pins	498 / 622 (80 %)
Total virtual pins	0
Total memory bits	834,164 / 1,152,000 (72 %)
Embedded Multiplier 9-bit elements	4 / 300 (1 %)
Total PLLs	2 / 4 (50 %)

图 12 系统的硬件综合结果

图 12 所示为构建系统的 FPGA 资源使用情况，占用的逻辑资源和片上存储器分别为 20%和 72%，逻辑资源的占用率较低而存储器占用率较大，因此本设计所选用的平台是合理的。

5.1.2 软件编译结果

本设计先将 U-boot 烧入 Flash，U-boot 初始化 uart 串口，通过串口将交叉编译后并进行了压缩的内核下载进 Flash。经过压缩的 Linux 内核需要 6MByte 空间。系统开机运行后可以通过 PS 命令查看各程序的内存占用情况，如图 13：

程序名		内存	CPU占用率
窗口管理器	nanowm	262	4.2
软键盘	nxkbd	262	0.1
时钟	nxclock	262	0.1
uClinux终端	nxterm	275	0.2
主界面程序	main	1497	8.9
视频解码	mpeg2dec	1715	39.9
Nano-X图形接口	nano-X	2245	45.7
浏览器	dillo	4434	4.7

图 13 系统使用时内存的占用情况

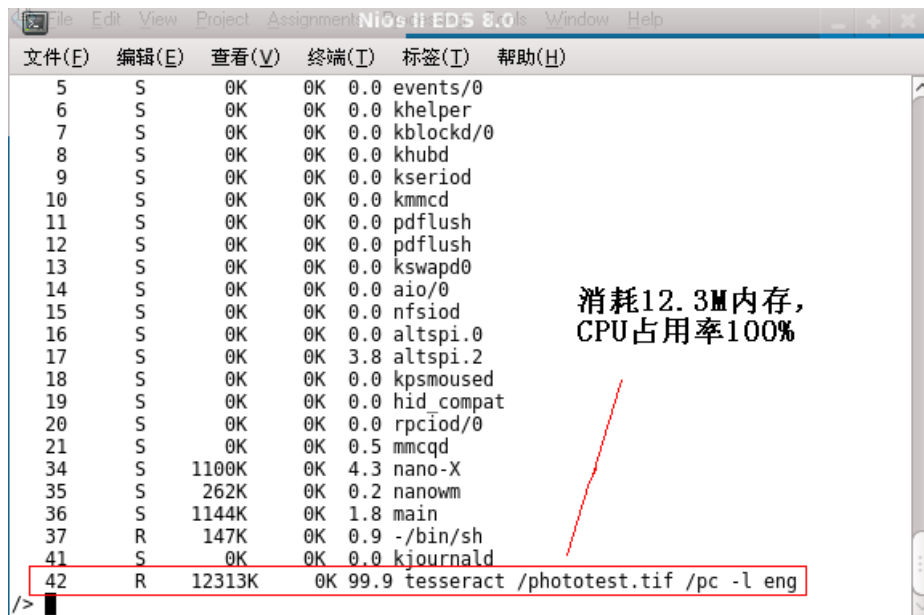


图 14 文字识别引擎的资源占用情况

图 13 所示为本设计除文字识别引擎外均打开时的 CPU 占用率，共使用了 12700K 的内存。图 14 是文字识别引擎工作中的内存占用情况。文字识别引擎需要 12M 的内存空间，因此在文字识别引擎打开时，需要减少其他程序的使用。

5.2 功能验证

将硬件信息配置进入 FPGA，同时，通过 Flashprogrammer 将 bootloader 和 Linux 下载进入 Flash。Linux 内核启动完成后，调用交互界面程序，系统可以通过鼠标键盘进行输入操作，通过 VGA 输出交互界面。此外，可通过串口对系统进行调试并看到打印出来的系统信息。

- 1、将拍摄到的文字图像存入 SD 卡中插入开发板。
- 2、编写程序，通过 Nios II 将图像从 SD 卡中读入，并写入硬件模块。
- 3、查询硬件模块发出的计算完成信号。
- 4、读取硬件模块送回的倾斜角度。
- 5、进行图像旋转并将新图像存入 SD 卡中。
- 6、取出旋转后的图像在电脑上打开以验证功能的正确性。

硬件模块返回的倾斜角度和经过变换的图像效果如图 17 所示：

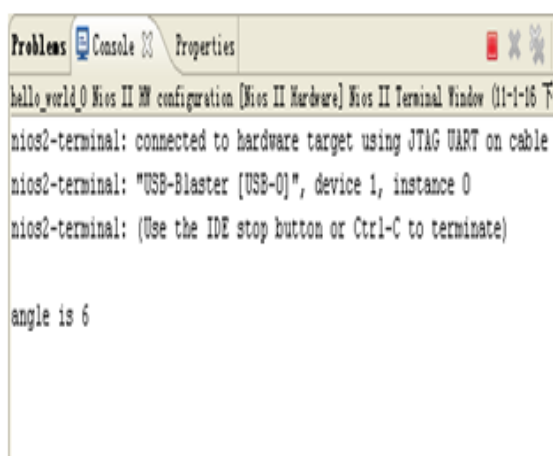


图 17(a) 倾斜角度计算结果



图 17(b) 图像旋转效果图

图 17(a) 是本系统对偏斜图像进行倾斜角计算的结果。本系统对前端采集回的图像（如图 16）进行了倾斜角计算，得到了图像的倾斜角。图 17(b) 是本系统图像倾斜纠正的效果图。从效果图中可以看出，本设计成功的将图像进行了倾斜纠正。

5.2.3 图像分割模块的验证

本设计通过 Nios II 将图像写入图像分割模块，得到图像分割模块给出的边界信息后，CPU 将图像进行分割，分割后的图像存入 SD 卡，从电脑中打开此来验证图像分割模块的功能。最终得到的图像分割效果如图 18(b)：

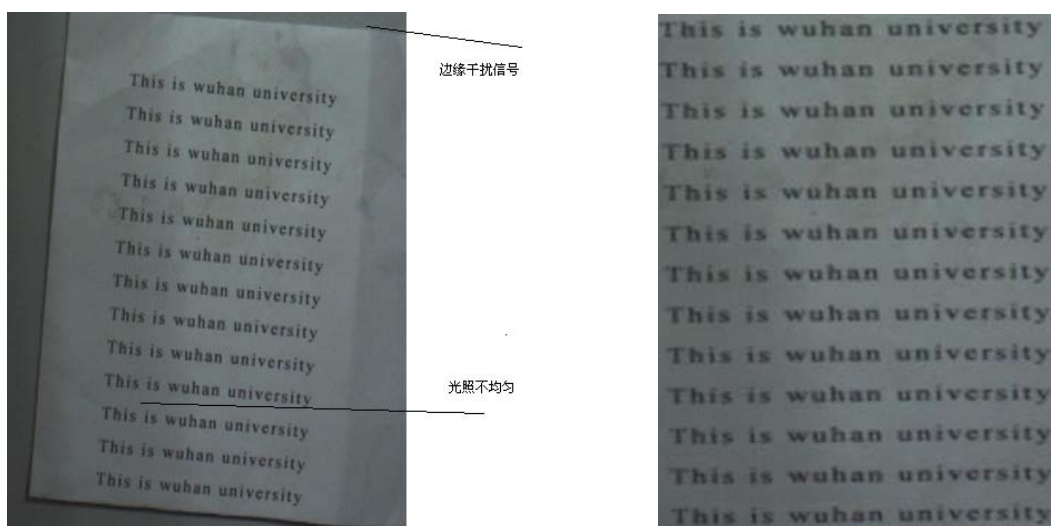


图 18(a) 原文字图像存在的干扰

图 18(b) 文字图像分割效果图

图 18(b) 是本系统对倾斜纠正后的图像进行分割，提取文字区域的效果图。与图 18(a) 原文字图像的效果图对比，可以得出图像分割模块很好的完成了文字区域的提取，提取后的区域光照不均匀情况得到了改善，同时也避免了边缘噪声对文字识别的影响。

5.2.4 图像二值化模块的验证

本设计将经过分割后的图像写入二值化模块，经过处理后读出存入 SD 卡中。通过这个流程可以验证图像二值化模块的功能，具体效果图如图 19：

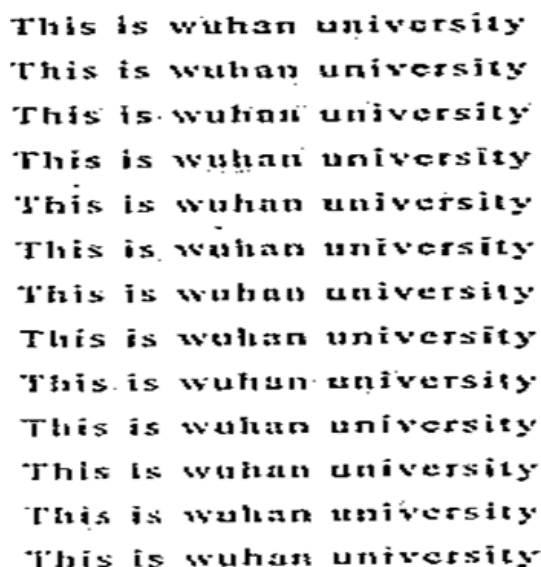


图 19 文字图像二值化效果图

图 19 是本系统对分割后的图像进行二值化后的效果图。二值化后的图像较好的保留了文字信息，完全去除了光照不均匀带来的影响。

5.2.5 系统软件验证结果

由于 DE2-70 开发板上 SDRAM 大小的限制，本设计 Tesseract 引擎的数据文件不能太大，因此先使用默认引擎进行英文的识别，然后用拍摄的几副文字图像对引擎进行了训练，再使用经过训练的引擎对图像进行识别。最终的图像识别结果存入 SD 卡中。文字识别的效果图如图 20:

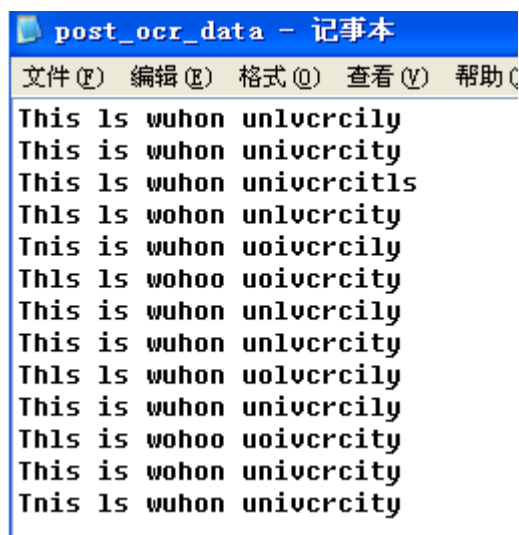
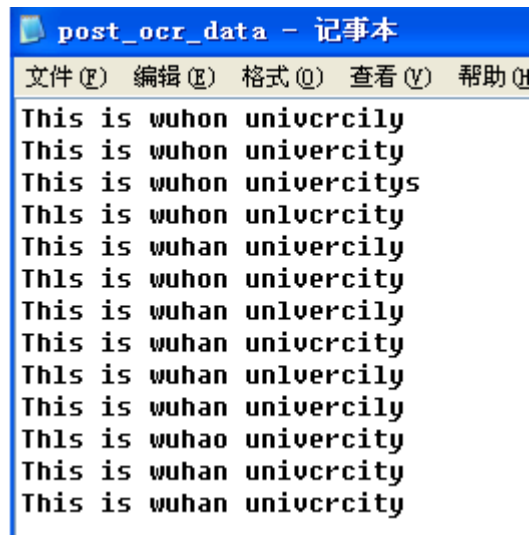


图 20(a) 默认引擎识别效果图



图(b) 经过训练的引擎识别效果图

图 20(a) 是默认引擎的识别效果图，图 20(b) 是经过训练的引擎识别效果图。通过对比可以看出，未经训练的识别效果错误率高达 18.59%，而经过训练的引擎识别效果明显好于默认引擎识别的结果，错误率仅为 7.7%。但是在一些图像伪影和不连续处，两个引擎都出现了不同程度的错误。其原因在于目前的文字识别理论还尚未成熟，难以达到百分之百的文字识别精确度。

5.3 比较与讨论

基于 SOPC 和嵌入式 Linux 的便携式文字识别系统方向的同类型研究较少。本文选取了已有的嵌入式电子系统和已有的文字识别系统两个方向分别进行了对比。

5.3.1 与已有嵌入式电子系统研究的比较

表 1 与已有嵌入式电子系统研究的比较

参数	本文	文献【5】	文献【6】
----	----	-------	-------

FPGA	EP2C70	EP2C35	EP1C12
操作系统	uClinux	无	uc/os-II
图形接口	X-windows	LCD 控制器	framebuffer
GUI	fltk	自己编写	mini GUI
网络协议	Linux 下完整的 TCP/IP 协议	无	LWIP (简单的 TCP/IP 协议实现)
图片/文档	支持	仅支持 TXT 文档	不支持
存储扩展	SD 卡/U 盘	SD 卡	SD 卡/U 盘
软件扩展	可以方便加入交叉编译后的 linux 执行文件	需要重新开发、编译、下载	需要重新开发、编译、下载
其它功能	时钟、计算器、软键盘输入、游戏、linux 终端控制台	无	时钟、闹钟、月历、计算器、加密、游戏

由以上对比可知，本文设计的便携式文字识别系统功能丰富，可扩展性强。与文献[5]相比具有多任务处理能力、较好的软件扩展性和更多的软件资源。与文献[6]相比也具有较好的软件扩展性和更多的软件资源。

5.3.2 与已有的文字识别系统研究的比较

表 2 与已有的文字识别系统研究的比较

参数	本文	文献【7】	文献【8】
实现方式	FPGA EP2C70	FPGA	软件
前端采集	摄像头套装	扫描仪	CCD 摄像头
识别方法	文字预处理+文字识别引擎	文字识别	文字分割+文字识别

和已有的文字识别系统的对比中可以看出，本设计将软硬设计协同起来，硬件实现了文字预处理的诸多算法，软件完成了文字识别引擎的移植。和以前的研究相比，本设计在功能多样性、系统集成性方面有一定的优势。

六、 总结

本文通过硬件电路完成文字图像采集和文字图像预处理,嵌入式软件完成文字识别,实现了一种基于 SOPC 的嵌入式文字识别系统。此设计中构建了完整的 SOPC 系统,嵌入了 linux 操作系统,通过软硬件协同充分发挥各自的优点,达到了较好的文字识别效果。本文对文字识别系统的实现以及基于 FPGA 的嵌入式的系统设计应用具有参考价值。

参考文献

- [1] 盛蹇, 刘伟. 计算机文字识别的发展及应用[J]. 科技信息. 2008, (13): 65
- [2] 王希常, 刘江. 一种扫描文档图像的快速纠偏算法设计[A]. 2010 Third International Conference on Education Technology and Training. 2010
- [3] 瞿燕慧. 图像分割常用算法优缺点探析[J]. 科教新报(教育科研). 2010, (3):57
- [4] Maya R.Gupata, Nathaniel P.Jacobson, Eric K.Garcia, 章晟(译). 旧式文本的 OCR 二值化和图像预处理研究[J]. 图象识别与自动化. 2007, (1):14
- [5] 谢辉程, 郭莉. 基于 SOPC 的 PDA 设计与实现[J]. 信息科技. 2008, (20):102.
- [6] 周鼎. 基于 nios 处理器的个人多媒体助手[D]. 中国地质大学(武汉). 2008:22
- [7] 龙钧宇, 张向群, 薛江清, 吴永. 一种基于 FPGA 的文字识别系统[J]. 现代电子技术. 2006, (11):103-105
- [8] 乔海晔, 肖南峰. 基于视觉的文字识别系统的设计与实现[J]. 交通与计算机. 2005, 23(5):95-98