

基于 FPGA 的哈夫曼图像压缩系统的设计与实现

解佩淼¹ 张霞¹
XIE Peimiao ZHANG Xia

摘要

随着信息数字化的飞速发展,数据压缩技术在节省存储空间和提高传输效率等方面发挥着关键作用。BMP 作为一种广泛应用的图像格式,因文件体积较大而限制了其传输和存储。Huffman 编码作为一种基于字符频率的前缀编码方法,在数据压缩领域具有显著效果。然而,传统 Huffman 编码软件实现方法在处理大数据量时效率低下。利用 FPGA 的并行处理能力,通过 Verilog HDL 设计了基于 Huffman 算法的 BMP 图像压缩与解压缩硬件系统。系统包括读取模块、压缩模块(含频次统计、排序、Huffman 树构建、码表创建、编码等子模块)、整合模块和解码模块。仿真结果显示,对同一张图像进行压缩时,硬件系统消耗的时间为软件平台的 41.06%,且压缩比最低可达 0.181 2。在 AMD Zynq 7000 SoC ZC706 FPGA 开发板上进行验证,验证结果表明系统能有效实现 BMP 图像的 Huffman 压缩与解压缩。

关键词

数据压缩; BMP 文件; Huffman 编码; FPGA

doi: 10.3969/j.issn.1672-9528.2024.11.019

0 引言

BMP 文件是 Windows 系统的标准图片格式,几乎被所有 Windows 软件所支持。它以未经压缩的方式直接存储图像像素数据。这种格式保留了图像的原始数据,因此具有较高的图像质量,但也导致文件体积通常较大,这使得它不太适合在网络环境中进行传输和存储^[1]。因此,可以采用无损压缩算法对 BMP 文件进行压缩,在降低其文件体积的同时保留高图像质量的特点。

无损压缩算法通过减少数据中的冗余部分来实现压缩,常见的有 Huffman 算法^[2]和 LZ 系列算法^[3]。LZ 系列算法需要构建索引字符串字典,在能够快速对数据进行压缩和解压缩的前提下,还能获得较好的压缩率,因此使用较为广泛。而 Huffman 编码是通过构造 Huffman 树来进行编码的, Huffman 算法在对基于字符的数据进行压缩时,具有很大的优越性。Huffman 编码能够根据数据中字符出现的频率来构建最优的编码表,这种自适应性使得 Huffman 算法能够针对不同数据集进行自动调整,实现最佳的压缩效果^[4]。Huffman 算法也常与其他压缩方法相结合,并可以取得更好的压缩效果^[5-7]。Huffman 编码的原理相对简单,可以通过软件较为容易地实现。然而,由于 CPU 是按指令顺序执行的,当需要处理大量数据的压缩和解压任务时,可能会消耗大量 CPU 资源并且花费较长时间。现场可编程门阵列(field programmable gate array, FPGA)是一种可编程逻辑器件,具有强大的并

行处理能力,以及灵活性高、处理速度快等优点,因此在数据压缩领域也得到了广泛应用^[8-13]。本文拟基于 FPGA 设计与实现 Huffman 编码,可以有效地提高编码速率。

1 Huffman 编码原理

Huffman 编码是基于字符出现频率的前缀不重复编码。通过构建 Huffman 树实现对频率较高的字符赋予较短的编码,从而实现数据压缩。主要步骤为:(1)统计数据源中每个字符出现的频率;(2)根据字符频率构建 Huffman 树;(3)遍历树,生成字符对应的码表;(4)通过码表对数据源进行编码。

以序列“AABBBCCCCDDDDDEEEEE”为例,构建的 Huffman 树如图 1 所示。

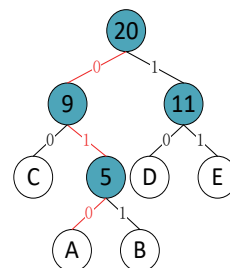


图 1 Huffman 树

从最顶层的根节点往下读,左节点为 0,右节点为 1,读到最后层即可生成对应的编码。例如,字母 A 的编码为 010。根据 Huffman 树对“AABBBCCCCDDDDDEEEEE”重新进行编码,最终输出的编码为“010010011011011000000”。

1. 西安邮电大学电子工程学院 陕西西安 710121

0010101010101111111111”。编码长度为 45 比特，与初始长度 160 比特相比，数据长度减少了 115 位。

2 系统设计方案

本系统按功能划分为读取、压缩、解压缩、输出四个部分。其中压缩部分最复杂，由频次统计、排序、Huffman 树构建、码表创建、编码模块共同组成。图像数据由读取模块调用 ROM IP 核输入，经过压缩和解压缩操作，最终通过输出模块调用 UART 串口进行输出。系统框图如图 2 所示。

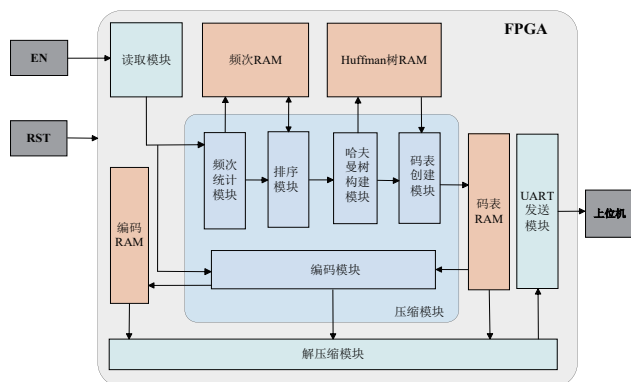


图 2 系统框图

2.1 读取模块

读取模块需要对图像数据进行预处理后再进行输出。Huffman 编码是基于数据源中数据的出现频率进行压缩，当数据中每种符号都趋近于均匀分布时，会导致最终的压缩效果不理想。利用差分编码对文件中的数据进行预处理，有可能将原本大而稀疏的数据转变为小而密集的数据^[14]。这对于 Huffman 编码是极为有利的。本设计使用 32 位 BMP 图像，32 位 BMP 图像按照 BGRA 的顺序来存储每个像素的各颜色通道的值，每个像素占用一个字节。一个像素的所有颜色分量值都存完后才存下一个像素。因此，只需要间隔 4 位对数据进行差分处理，就可以实现对每个通道的数据单独差分处理。在读取模块中设置 4 个寄存器来暂存前 4 个数据的值，在输出时将当前值与第 4 个寄存器中值进行差分操作即可实现对图像的差分处理。

按照 BMP 文件数据头的定义，图像的大小信息按照小端对齐的方式保存在第 3、4、5、6 个字节中，如表 1 所示。

表 1 BMP 文件头描述（部分）

地址	位宽	描述
0	2	头文件字段
2	4	bmp 文件大小
6	2	预留字段
8	2	预留字段

在读取时将图像的大小信息储存在寄存器中，通过比对寄存器中的数据，可以在读取完一张图片的完整数据后，发出停止信号。

2.2 压缩模块

压缩模块由频次统计、排序、Huffman 树构建、码表创建以及编码模块共同组成。

频次统计模块较为简单，根据读取模块的输出对相应字符的频次数据进行加一操作。

排序模块作为 Huffman 编码中的一个关键组成部分，在算法效率方面有着重要影响，其设计和实现直接影响了整个算法的性能表现。

传统的排序方法，主要采用串行方式实现，大多采用循环比较，较为费时。目前学者们基于 FPGA 的硬件特性，已设计出不同的并行排序方案^[15-16]。本模块基于并行全比较算法进行修改。并行全比较算法利用 FPGA 并行处理的能力，同时对多个数据进行比较和排序，从而可以大幅提高数据处理的实时性。它的原理为所有数据同时进行两两比较，每两个数进行比较都会产生一个比较结果，可以根据两个数的大小定义输出结果为 0 或 1。将这些比较结果进行累加计算，即可得到每个数在序列中的排序值。

在本设计中，差分后的数据量最多为 512 个，每个数据占用 16 位宽。按照并行全比较算法进行设计，资源需求量巨大。在第二步的两两比较阶段，将原本的并行全比较改为串行全比较，每个时钟周期只选取一个数据与其他所有数据进行比较。如图 3 所示。

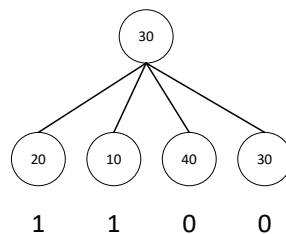


图 3 全比较示意图

对于 m 个数，串行全比较在比较阶段和累加所花费的时间分别由 1 个时钟周期增加到 m 个时钟周期。但由于模块本身的读取和输入就各自需要耗费 m 个时钟周期，因此，串行全比较所需时钟周期增加了约一倍，但大大降低了资源需求。以 512 个数为例，在全比较阶段，若采用并行全比较，则至少需要 $511 \times 511 = 261\,121$ 个比较器，而串行全比较仅需 511 个比较器。这一优化策略使得算法在资源有限的情况下仍能高效运行。

Huffman 树构建模块需要根据每个字符以及其出现的频次构建对应的 Huffman 表。按照 Huffman 树的结构，用 3 个 RAM 来储存整颗 Huffman 树，分别为 F_RAM，L_RAM，R_RAM。F_RAM 来储存生成的父节点。L_RAM 和 R_RAM 分别储存组成父节点的左右节点。通过不断合并频次中的最小值与次小值来完成 Huffman 树的构建。合并时，最小值存入 L_RAM，次小值存入 R_RAM，合并值存入 F_RAM。合并完成后，舍弃当前的最小值与次小值，将合并值加入频次

数据中,进行下一次合并操作直至 Huffman 树构建结束。状态转移图如图 4 所示,其中 data1 和 data2 分别为当前合并过程中最低位和次低位数据。

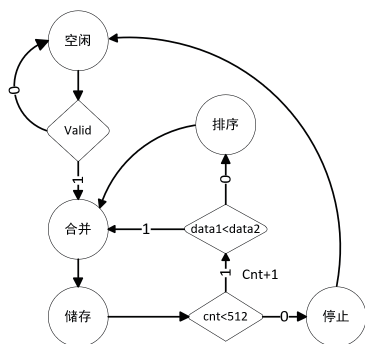


图 4 Huffman 树构建模块状态转移图

码表创建模块需要对整棵树进行遍历并生成码表,在 Huffman 树中,左枝干代表 0,右枝干代表 1,从最顶层节点读取到字符所在节点即可得到该字符对应编码。通过对 F_RAM、L_RAM、R_RAM 进行遍历即可得到完整码表。该模块的状态转移图如图 5 所示,其中,Jud_reg 代表当前读取到的节点类型,1 代表原始数据所在节点,0 代表合并数据所在节点,我们只需要构建出原始数据的码表。

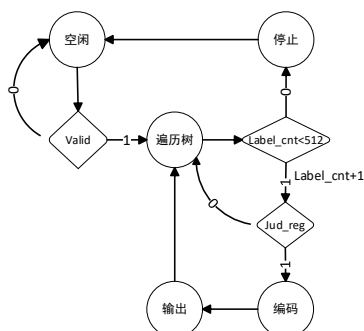


图 5 码表创建模块状态转移图

当码表生成结束后,编码模块调用读取模块,再次读取图像数据。在读取过程中根据生成的码表对数据进行编码,即可完成图像数据的压缩。压缩后的数据保存在编码 RAM 中。

2.3 解压缩模块

解压缩模块通过生成的码表对编码后的数据进行解码操作。在本设计中,解压缩模块直接调用编码 RAM 和码表 RAM,每读取编码 RAM 中的一位数据就进行一次查表操作,如果没有查找到对应的标签值,就读取下一位数据,并将其与上一位数据合并继续进行查表操作,直到查找到对应的标签值。当查表成功时,对该数据进行反差分操作并输出。不断重复即可完成对图像数据的解压操作。

2.4 输出模块

输出模块采用 UART 来传输解压缩后的数据。UART 是一种串行、异步、全双工的通信协议,数据传送速率用波特

率来表示。本设计采用无校验位的 UART 协议。解压模块每解码出一位数据便调用输出模块进行一次输出。

3 系统功能验证

3.1 编解码正确性验证

系统最终通过 UART 发送解压缩后的数据,因此添加一个 UART 接收模块,将发送的数据进行接收。仿真时将接收到的数据以十六进制的形式保存在 TXT 文件中。用 MATLAB 脚本读取 TXT 文件中的十六进制数据,将其重构为图像,以图 6 为例,对其进行处理后与原图数据进行对比,对比结果表明数据完全一致,如图 7、图 8 所示。证明系统的压缩和解压缩功能是正确的。



图 6 印章 256×256 图

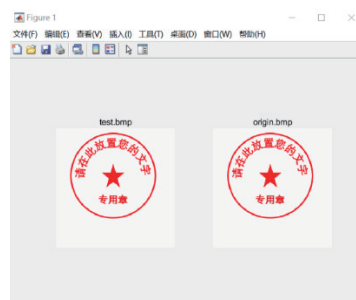


图 7 解压缩图（左）与原图（右）对比



图 8 解压缩数据与原图数据对比结果

3.2 压缩比仿真测试

添加一个整合模块,将码表和编码后的数据整合为八位宽的数据一起进行输出。在输出时通过统计输出数据的总量即可得到数据压缩后的大小。

以 256×256 的印章图标为例(如图 6 所示)。通过仿真并查看寄存器的值,可知最终整合后的数据总量为 47 506 位,如图 9 所示。其中包括了码表和编码后的数据。初始数据大小为 262 200,经过计算,压缩比为 0.181 2。

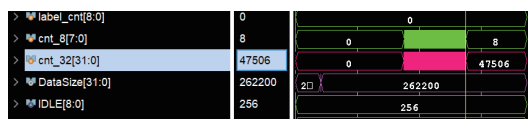


图 9 压缩比仿真图

之后对其余 4 张图像（图 10~ 图 13）进行了测试，并将数据进行汇总，如表 2 所示。

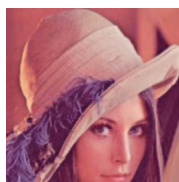


图 10 Lena256×256

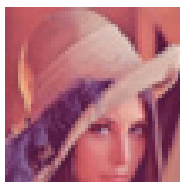


图 11 Lena53×53



图 12 线条画

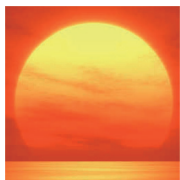


图 13 夕阳

表 2 压缩比仿真数据

BMP 图像	初始大小 (字节)	压缩后大小 (字节)	压缩比
Lena256×256	262 200	146 463	0.558 6
Lena53×53	11 292	8481	0.751 1
线条画 256×256	262 200	91 750	0.349 9
夕阳 256×256	262 200	175 593	0.669 7
印章 256×256	262 200	47 506	0.181 2

从表 2 可以看出，对于自然图像，本设计压缩效果较差，而对于线条画、印章这种简单的人工图像，本设计压缩效果较好。根据 Lena 256×256 和 Lena 53×53 的测试结果来看，当数据总量减少时，压缩效果也随之降低。这是由于码表的储存要占用一部分空间。因此当数据总量较少时，不适合用 Huffman 算法进行压缩。

3.3 板级验证

本次验证用到的工具为：AMD Zynq 7000 SoC ZC706 FPGA 开发板，CH340 串口模块，PC 机，串口助手软件，MATLAB。测试图像为 256×256 印章图像。

用 Vivado 对项目进行综合、布局布线等过程，将最终生成的比特流文件下载到开发板。将开发板与 CH340 串口模块连接，再将 CH340 串口模块连接到电脑。在开发板上运行电路，在电脑上通过串口助手接收电路发送的数据。接收到的数据会保存在一个 TXT 文件中，用 MATLAB 脚本对文件中的数据进行验证。验证结果表明数据与原图完全一致，如图 14 和图 15 所示。



图 14 FPGA 解压缩图像与原始图像的对比

命令行窗口

```
>> zhuantuxiang
两个文件中的十六进制数据完全相同。
fx>>
```

图 15 FPGA 处理后的数据与原图数据对比

4 性能分析

本设计最大工作频率约为 73.65 MHz。在该频率下，对 256×256 lena 图像压缩需要的时间为 81.3 ms。在使用最大频率为 5 GHz 的锐龙 57 500 F 处理器的情况下，在 MATLAB 中，使用 Huffman 编码对该图像压缩 1000 次，平均压缩时间为 198 ms。由此证明本设计有效提升了 Huffman 编码的速度。

5 结论

本文基于 FPGA 进行了针对 BMP 图像文件的 Huffman 编码压缩算法的设计与实现。利用 FPGA 平台的高并行处理性能，针对 BMP 图像文件体积大、不利于传输和储存的问题，通过像素值差分处理与 Huffman 编码相结合成功对 BMP 图像文件进行了压缩，并验证了其正确性。实验结果表明，利用 FPGA 平台实现的 Huffman 编码系统能有效提升压缩速率。

参考文献：

- [1] 王盼盼. 计算机图形图像处理的关键技术 [J]. 信息记录材料, 2024, 25 (3): 68-70.
- [2] 吕姣霖, 徐艳. 哈夫曼编码在图像压缩中的应用与分析 [J]. 数字通信世界, 2021 (1): 189-190.
- [3] 成小勇. LZW 数据压缩的改进与加密研究 [D]. 南京: 东南大学, 2022.
- [4] 颜丽君. 基于自适应 Huffman 的海洋多道地震拖缆数据压缩方法的研究与实现 [D]. 荆州: 长江大学, 2021.
- [5] MORE S, SANIVARAPU V, SHARMA Y, et al. Enhancing data compression: a dynamic programming approach with huffman C-oding and burrows-wheeler transform[C]// 2023 2nd International Conference on Aut-omation, Computing and Renewable Systems (ICACRS). Piscataway: IEEE, 2023: 82-88.
- [6] 李勋卓. 高性能 JPEG2000 图像压缩算法硬件实现研究 [D]. 武汉: 华中师范大学, 2021.
- [7] 张婉莹. 美元纸币冠字号码图像压缩算法研究 [D]. 鞍山: 辽宁科技大学, 2022.
- [8] SIVANANDAM L, PERIYASAMY S, OORKA-VALAN U M. Power transition X filling based selective Huffmanen-coding technique for test-data compression and scan power R-eduction for SOCs[J]. Microprocessors and microsystems, 2020,72:102937.

基于改进型 Faster R-CNN 的变电站设备缺陷检测

康 曦^{1,2} 邓琛韬² 伍 毅¹ 葛 林² 朱东松² 王 军¹

KANG Xi DENG Chentao WU Yi GE Lin ZHU Dongsong WANG Jun

摘 要

为了保障变电站设备的稳定运行,准确检测变电站设备缺陷,提出一种基于改进型 Faster R-CNN 的设备缺陷检测方法。通过引入哈希辅助分支,抑制背景区域干扰,提高了检测精度,同时避免了非目标区域的无效计算,可以有效提高检测效率。实验表明,引入深度哈希辅助分支后,与原 Faster R-CNN 模型对比,在非目标区域上的计算时间从 0.082 2 s/张提高到 0.044 3 s/张,检测结果的 mAP 从 76.43% 提高到 81.24%,在保持较高检测速度的同时可以有效地提高检测精度,验证了模型的可靠性与高效性。

关键词

Faster R-CNN; 缺陷检测; 哈希辅助分支

doi: 10.3969/j.issn.1672-9528.2024.11.020

0 引言

变电站是电网不可或缺的一环,在运行过程中,其金属设备可能会出现放电痕迹、坏点损伤、漆泡损伤等问题。为保障电力系统的稳定运行,对无人值守变电站设备进行实时检测^[1]已然成为一项重要的任务。传统基于机器学习的设备表面缺陷损伤检测方法,如图像分割方法^[2]、特征提取

方法^[3]、光谱算法^[4]采集图像受环境因素影响较大,且变电站设备的缺陷具有连续扩展性,所以需要泛化性能更强的深度学习模型用于设备表面缺陷损伤检测。为了有效提高检测准确率和检测速度,本文提出一种基于改进型 Faster R-CNN 的金属缺陷检测方法。样本采用 NEU-DET 数据集,将无目标背景图像块与有目标图像块分开,统计加入深度哈希辅助分支前后的平均测试时间与检测准确率。实验结果表明,加入深度哈希辅助网络模型不仅能大幅度缩短模型检测时间,还可以提高检测精度。

1. 江西理工大学机电工程学院 江西赣州 341000
2. 国网江西省电力有限公司赣州供电分公司 江西赣州 341000

- [9] 胡楠. 基于 FPGA 的随钻数据压缩系统设计 [D]. 荆 州: 长江大学, 2021.
- [10] 刘谱光, 魏子令, 黄成龙, 等. 一种基于 FPGA 加速的高性能数据解压方法 [J]. 计算机学报, 2023, 46 (12): 2687-2704.
- [11] 陈晓杰, 李斌, 周清雷. RTL 级可扩展高性能数据压缩方法实现 [J]. 电子学报, 2022, 50 (7): 1548-1557.
- [12] 王飞, 李钊, 尹晓华, 等. 高速数据压缩及加密硬件加速电路研究 [J]. 计算机与数字工程, 2020, 48 (1): 212-216+246.
- [13] 张成哲, 张迎春, 曹中清, 等. 基于 FPGA 的波形数据压缩算法设计与实现 [J]. 自动化与仪器仪表, 2022(12): 1-5.
- [14] 梁捷, 蒋雯倩, 李金瑾. 基于动态字典和差分编码的计量数据压缩研究 [J]. 信息技术, 2020, 44 (10): 116-120.
- [15] ANTONOV A, BESEDIN D, F-ILIPPOV A. Research of the efficien-cy of high-level synthesis tool for FP-GA based

hardware implementation of some basic algorithms for the big d-ata analysis and management tasks[C]//2020 26th Conference of Open Inno-vations Association (FRUCT). Piscataway: IEEE, 2020:1-7.

- [16] YAN D, WANG W X, ZUO L, et al. A novel scheme for real-time max/Min-set-selection sorters on FPGA[J]. IEEE transactions on circuits and systems,ii.express briefs, 2021, 68(7): 2665-2669.

【作者简介】

解佩森(2002—), 男, 陕西汉中, 本科, 研究方向: 集成电路设计。

张霞(1983—), 女, 内蒙古牙克石人, 博士, 副教授, 研究方向: 集成电路设计。

(收稿日期: 2024-07-30)