高技术通讯 2009 年 第 19 卷 第 3 期:247 ~ 252

doi:10.3772/j.issn.1002-0470.2009.03.006

分步式并行 MQ 编码及其 VLSI 设计①

王 前② 吕东强*

(北京航空航天大学计算机学院数字媒体实验室 北京 100083)

(*第二炮兵装备研究院四所 北京 100085)

摘 要 针对 MQ 编码的环路反馈结构的高复杂度对实现快速图像压缩硬件的限制,研究分析了 MQ 编码的基本算法,提出了"区间编码"和"位填充"之间有一定的独立性,可用先进先出(FIFO)管道连接后并行处理的思想,并设计了一种适合 MQ 编码算法特点的异步流水线与有限状态机(FSM)相结合的分步式并行结构。该结构简单合理,FIFO 管道的引入可支持异步流水电路,FSM 的动态优化策略有效地防止了流水的阻塞,复杂环路的逐层分解显著降低了编码的反馈效应,根据程序运行过程中的数据操作动态特征,利用概率统计规律和状态机分割减小了系统的关键路径长度。该结构的资源利用率高,现场可编程门阵列(FPGA)原型系统最高时钟工作频率为 233MHz,吞吐率与其它同类结构相比有明显提高,达到 116.5Mbps。

关键词:算术编码,分步式,图像压缩,关键路径

0 引 言

成像侦察是航天侦察的重要组成部分。高分辨率传感器技术的突破可提高成像侦察的精度,但产生的大量数据受到数传带宽的限制,必须经过压缩才能解决数据传输问题。以 JPEG2000 为代表的压缩编码标准具有卓越的图像压缩性能和良好的抗误码能力,但它会带来实现复杂度的提高。实验表明,JPEG 2000 的核心编码结构——嵌入式优化截断块编码(embedded block coding with optimized truncation, EBCOT)部分占整个编码时间的 60%以上,位平面部分可以实现码块、编码通道、位平面、样本等多种并行方式。因此,EBCOT 中的算术编码器——MQ 编码器的设计与实现就成为关键,因为 MQ 编码器中包含典型的环路反馈结构,其高复杂度限制了硬件快速实现。

针对 MQ 的硬件实现,不少学者提出了自己的算法和设计结构。如文献[1]提出将寄存器 C 分解成 16 位和 12 位两部分以减少关键路径长度;文献[2]提出基于反向多分支选择的每周期能编多位符号的编码器结构,文献[3]运用提前计算和数据关联分析提高编码吞吐率,文献[4]利用优化组合方法复制部分使用频度较高的模块以增加系统的并行度,能在一定条件下处理多位数据,文献[5]采用基于位平面编码层次上的通道并行策略,其实质是完整 MQ 编码器的同等复制。这些思想大都基于同步流水线方法,对编码局部或整体进行复制或改进,在一定程度上提高了编码速度,但没有充分利用 MQ 算法内部的并行特性,因而资源消耗较多,编码速度仍不很高。本文在分析 MQ 编码器算法特点的基础上,结合状态机优化分割方法,提出了新型的分步式全并行结构,其逻辑控制简单,避免了不必要的时钟浪费,编码的速度和资源效率都得到明显的改善。

1 MQ 编码分析

MQ 编码器是在原有 IBM 的 Q 编码器基础上的优化编码算法,由于压缩效率高,逐渐得到广泛应用。该算法有效去除了编码过程中的乘除法运算,仅使用加减和移位运算来进行区间的计算和调整,提高了算术编码器的处理速度。图 1 为 MQ 串并方案对比图,图中上半部分表示 MQ 编码的串行基本结构。MQ 编码器接收位平面预测模块产生的编码数据(D)和对应的上下文(CX)归属,CX 内容是位平面预测中根据系数相关性归纳而来的,共有 19 项,

① 863 计划(2006AA701121)、教育部博士点基金和新世纪优秀人才支持计划资助项目。
② 男,1978 年生,博士生;研究方向:图像压缩技术及其硬件实现,联系人,E-mail: wqaloha@163.com
(收稿日期:2008-04-16)

每项对应概率估计表中某个确定的地址索引位(index)。概率估计表如表 1 所示,共有 47 个单元。

表中 index 表示存储单元的地址,NLPS 表示下一个 LPS(小概率符号)的 index,NMPS 表示下一个 MPS(大概率符号)的 index,switch 是概率交换标志。设定好 index 的初值后,可通过输入数据和归一化后的更新数据确定 index 的下一状态。

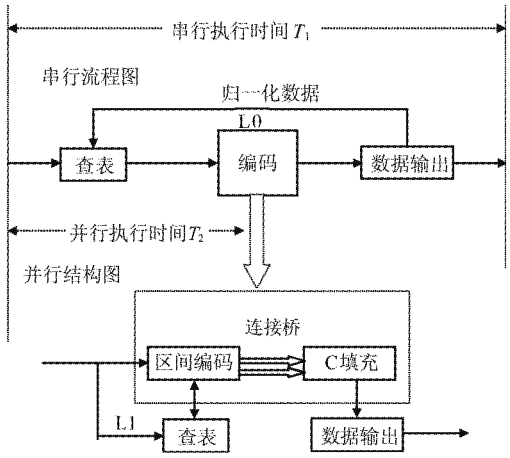


图 1 MQ 串并方案对比图

表 1 概率估计表

index	Qe 值	NMPS	NLPS	switch
0	0x5601	1	1	1
1	0x3401	2	6	0
·				
·				
45	0x0001	45	43	0
46	0x5601	46	46	0

MQ 编码的基础是递归概率区间分割。编码过程就是根据输入的编码数据 D 进行区间分割,设 A 为编码区间,C 指向当前区间底端。由于乘法在具体实现时花费代价太大,因此 MQ 编码通过将 A 保持在一定的区间内来近似 A * Qe 为 Qe,从而免除乘法运算。当 D 被判断为 MPS 时,A = A - Qe,C = C + Qe;当 D 被判断为 LPS 时,A = Qe,C = C。需要注意的是,上述操作可能会出现 LPS 对应区间大于 MPS 对应区间的情况,此时需要进行区间对调。随着编码的进行,当编码区间 A 小于一个预定的最小值(0x8000),就需要进行归一化。此时 A 和 C 同时左移,当 C 移出一定位数时,产生溢出位,并最终形成压缩码流。后续编码需得到先前编码完成后的更新值,才能进入系统。因此,1bit 数据的完整时耗为

$$T_1 = T_{\text{table}} + T_{\text{encode}} + T_{\text{out}} \tag{1}$$

式中 T_{table} 为查表时间,T_{encode} 为编码时间,T_{out} 为数据输出时间。

2 分步式并行 MQ 设计

由图 1 上部分 MQ 编码过程可知,MQ 软件结构串行特征明显,且存在反馈结构,不适合应用在现场可编程门阵列(FPGA)的硬件平台中。算术编码属于典型的环路结构模型,其吞吐率由环路执行的最短时间 T_∞(即“环路边界”)决定,可表示为

$$T_\infty = \max_{l \in L}\left\{\frac{t_l}{w_l}\right\} \tag{2}$$

其中 t_l 为环路计算时间,w_l 为环路的时延数目。为此,本文提出了一种新型的 MQ 硬件结构方案,即通过分步式并行结构分解串行结构中的大环路 L_0,尽可能缩短 T_∞ 的大小,有效降低编码过程中的时间冗余。

MQ 算法中的归一化是由编码区间 A 的缩小引起的,概率表的状态跳转即 index 的确定也是由 A 和 Qe 决定的,只有 A 的变化才会引起 C 的相应变化。编码过程中 A 和 C 同时操作可简化为 A 操作时保存相关信息后,不必等待 C 填充操作及后续的码流输出等一系列的复杂操作,直接进行下一个编码过程。这样在等价编码的情况下,大大节省编码时间。查表地址序列 index 的确定与编码区间 A 是否归一化更新无关,可通过组合逻辑以及固化在系统内部的 ROM 查找表求得下一个查表地址,这样查表与区间编码同时操作,可进一步降低编码时间。C 填充器与数据输出也存在并行的可能,由此设计出如图 1 下半部分所示的分步式 MQ 编码器硬件结构。从图 1 可知,1bit 数据的并行编码时耗为

$$T_2 = \max\{T_{\text{table}}, T_{\text{dist\_encode}}\} = T_{\text{dist\_encode}} \tag{3}$$

式(1)中的 T_{encode} 为区间编码和 C 寄存器填充这两部分的调整时间和,式(3)中的 $T_{\text{dist\_encode}}$ 仅为区间编码的时间。很显然,$T_{\text{encode}} > T_{\text{dist\_encode}}$,由此可进一步得到 $T_2 << T_1$,这就为并行编码时间小于串行编码时间奠定了理论基础。此外,可用环路优化的方法来理解图 1 中的对比关系。上下两种结构虽然都存在反馈,但环路 L1 的长度很明显小于 L0 的长度,这样便更有利于分级流水操作。接下来的重点工作是如何设计合理完善的硬件结构,尽可能地减小 T_2 即 $T_{\text{dist\_encode}}$ 的数值。下面给出分步式并行 MQ 的编码结构和实现细节。

2.1 分步式并行 MQ 的整体结构

整体架构分为如图 1 下半部分所示的四级流

水,流水内部模块采用优化的状态机电路,彼此间并行工作,采用两个先进先出(FIFO)管道连接区间编码模块和 C 填充器模块,C 填充器模块通过标志信号来控制输出模块的启动。完整的 MQ 编码结束后,应有 flush 模块来清除 C 填充器中的残留数据。由于这部分的实现较为简单,对编码器的性能影响甚微,因此本文省略了这部分的描述。

2.2　区间编码与查表的并行

区间编码与查表的并行主要指编码区间 A 在归一化和输入数据的同时进行查表操作。在具体实现中,用阵列数组存储上下文编号以及对应的表索引(index)和 MPS 值,用 Xilinx 自带的 IP 核生成 ROM,用来存放表 1 的内容。A 的归一化采用并行一次归一化的方法。此方法运用串并转换原理,能够在一个时钟周期内计算出 A 的移位次数,无需通过移位次数的高位字段情况判断低位字段的数值状态。区间编码分解成区间调整和归一化分两阶段操作,既减少了环路的关键路径长度,也无需如文献[6]所述的扩展概率表以保证适时更新,这样可节省硬件资源,确保查表和编码两模块的同步进行,提高并行的效率。具体的设计框图,即图 1 中 L1 环路的分解图如图 2 所示。在查找概率表时需要用到区间调整模块的输出信息,由此形成反馈回路 L2,小于优化前整个区间编码形成的回路 L1。

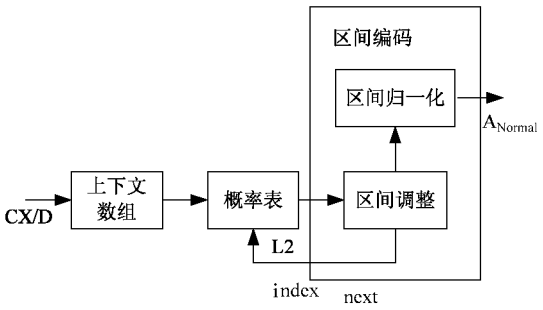


图 2　L1 环路分解图

2.3　连接桥的设计

连接桥是整个系统的核心纽带,可分解为同步并行的的 bridge0 和 bridge1。Bridge0 用来传送 C 的变化量,一般为零或概率值;bridge1 用来传送归一化时的移位量。之所以引入连接桥,是为了防止区间编码模块和 C 填充器模块速度不匹配而造成的数据丢失,保证各模块高速稳定的运转。连接桥采用先入先出(FIFO)机制,区间编码模块在入口端写入数据源,填充器模块在出口端读出数据。由于整个电路需要形成完整的流水线,各类时序关系(主要有 bridge0、bridge1、区间编码及 C 填充器这四部分时序)较复杂。两个 bridge 中的数据不是同步进行读写的,更容易造成时序的混乱。因此需要合理的利用好管道中读写控制信号以及空/满状态信号,保证时序的有序运行就显得尤为重要。

在 FIFO 通道没有数据输入时,可强制输入零代表空数据,以此来匹配 FIFO 的读写时序。当 FIFO 的所有存储空间被占满时,会出现“写停滞”状态。由此可见,FIFO 的深度大小对系统的运行效果会产生一定影响。当然,FIFO 空间太大的话不仅造成资源的浪费,而且也不能显著提高系统的性能。最合适的 FIFO 深度就是要匹配两连接模块的吞吐量之差。图 3 表明了 FIFO 深度与系统性能的对比图,横轴代表 FIFO 的深度,纵轴为处理率的倒数。由图中上曲线可知,随着 FIFO 深度的增加,系统的处理率逐渐增加,但幅度不明显,且所需资源却急剧增多。因此需要优化模块中的电路结构。

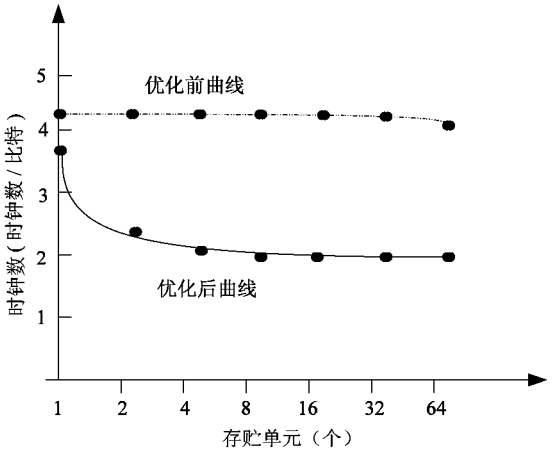
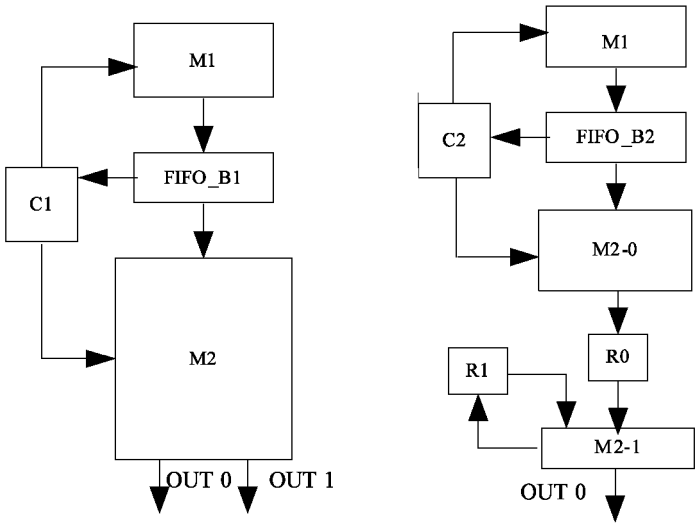


图 3　FIFO 深度与处理性能趋势图

抽象出的结构关系如图 4(a)所示,桥接电路 FIFO 连接模块 1(M1)和模块 2(M2),M1 代表区间编码,M2 代表 C 填充和数据输出,控制器 C1 根据 FIFO 的状态控制 M1 和 M2 的运行。M1 的输出速率为 v_0,由于输入码流的多样性,v_0并不是恒定的,可统计表示为 $\bar{v}_0$;M2 涉及到数据输出,在数据没有输出、或输出 1 字节或 2 字节等多种情况下,有 3 种处理速度,发生的概率分别为 $\{P_1, P_2, P_3\}$,而系统的关键路径由最长的一段输入输出路径决定,因此处理速度为 $v_1 = \min\{v_{1-0}, v_{1-1}, v_{1-2}\} = v_{1-2}$。设处理图像的尺寸为 256×256,则需要编码的数目为 $n = \frac{256 \times 256 \times 8}{\eta}$,$\eta$ 与压缩倍率以及编码器的压缩性

能有关,根据统计表 2,大致估算出在 *FIFO* 中的最多"滞留"数目为 $\bar{m} = n\left[\frac{1}{v_0} - \frac{1}{v_1}\right] = 37575$。由此可见, m 太大会造成硬件资源的严重浪费,必须增加 M2 的处理速度。



(a) 优化前的模块关系　　(b) 优化后的模块关系

图 4　模块关系

表 2　MQ 编码操作类别统计表

	City	Couple	Factory	Lena	Cman
编码数目	89343	112254	93174	131618	137235
移出 0 字节数目	85268	108198	89097	127559	133175
移出 1 字节数目	4059	4021	4063	4027	4029
移出 2 字节数目	16	35	14	32	31

一种直观的办法是利用 FIFO 的异步性,提高 M2 的工作频率,但受到关键路径的制约,且会增加功耗;第二方法是根据编码过程中表现出的统计规律,用状态机电路优化 M2 的结构。分解移出 2 字节和 1 字节这两个复杂操作,用寄存器缓存操作的中间结果。具体操作如图 4(b)所示。M2-0 表示没有数据输出的模块,寄存器 R0 缓存发生 1 字节输出时 M2-0 的输出结果,减少了关键路径的长度。寄存器 R1 缓存发生 2 字节输出时 M2-1 的输出情况,然后再环回到 M2-1 作进一步输出处理。这在一定程度上会造成 M2-1 的工作"停滞",但由于这个操作的发生概率较小,由表 2 的统计结果可知在 0.1% 以下,因此 M2 的总体处理能力增强, $v_1 = P_3 v_{1-2} + (1 - P_3) v_{1-0} \approx v_{1-0}$, 优化后的性能曲线如图 3 下曲线所示。当 FIFO 深度小于 4 时,处理率变化很剧烈,而深度大于 4 以后,处理率的变化趋于平缓。模块 2 的关键路径长度由优化前的 9.1ns 下降到优化后的 4.3ns,既节省了硬件资源,又大幅提高了系统的处理能力。

2.4　输出单元的设计

输出单元的主要工作是移出 C 寄存器中的高位字节,形成压缩码流,移位后的 C 寄存器需环回覆盖。标准算法中规定 C 寄存器的位长为 28bit,包含 1bit 进位字段。在输出缓存为 0xFF 时,要考虑到进位传播的影响,只能移出 C 寄存器中的前 7 位加进位 bit,否则,移出 C 寄存器中的 8 位数据。因此,可根据 C 寄存器的不同移位方式提前计算其更新值,缩短环路更新周期。

3　性能分析

本文采用 Verilog 语言对分步式 MQ 编码器进行描述,选用的器件是 Xilinx 公司的 Virtex4 系列中的 xc4vlx25-12sf363,布局布线在 ise7.1 环境下完成,最后的仿真在 modelsim 5.8b 中进行,得到的结果与软件版本完全一致。另外,本文还对经典的串行状态机控制结构进行了对比实验。

为全面测试两类编码器的性能,本文选择 5 幅

标准 RAW 格式的图片,每幅图片各抽取一条属于同一上下文的重要位(Sig)和精细位(Mag)通道[7]数据进行实验。由于这两个通道的 0、1 分布比例有一定的差异,所以会对压缩性能产生一定的影响,重要位通道的加速比略高于精细位通道。但从表 3 中的数据来看,影响不会太大,在相同的系统工作频率下,两个通道的加速比一般都在 3.3 左右。

表 3　图像压缩速度比较表

图像	通道名称	数据量(bit)	串行时钟数	本文时钟数	加速比
beijng	Sig	11712	82145	24353	3.37
	Mag	3928	24666	7715	3.20
barbara	Sig	15118	106556	31568	3.38
	Mag	8275	55039	16965	3.24
lena	Sig	7057	48293	14452	3.34
	Mag	3343	20952	6521	3.21
boat	Sig	10444	72914	21655	3.37
	Mag	4551	29567	9073	3.26
goldhill	Sig	8255	57019	17004	3.35
	Mag	3524	22092	6889	3.21

本文从不同角度分析编码器的综合性能,具体对比结果如表 4 所示。由表 4 可知,本文结构在 Virtex 2 芯片上的性能和资源利用率等方面均高于文献[8]的流水线结构。该流水线结构可在一个时钟周期内处理 1bit 数据,但关键路径较长,使频率大幅降低。文献[5]采用四级流水的 MQ 编码结构,最高工作频率有所提高,但对复杂环路的分解力度不够。文献[3]采用状态机控制方式,同等工作频率下的吞吐率相同(0.5 比特/时钟),但本文采用了分步式结构,把大的串行环路分解成若干个并行的小环路,复杂度大为降低,因此得到较高的工作频率。串行结构的最高工作频率略高于本文结构,但处理周期过长,需要 6 到 7 个系统时钟。本文算法在资源耗费方面高于串行算法,这主要是由于算法的并行结构造成的。在实际运用中,一片 Virtex4 系列的芯片一般能提供几万门的 slice,因此这样的消耗就显得微不足道。当然最高工作频率并不是衡量系统性能的唯一指标,单位系统时钟周期内的数据处理率也是重要检验因素。这两者相互制约,在同等硬件资源条件下,数据处理率降低,系统的复杂性也随之降低,由此导致关键路径的降低,因而能大大提高主时钟频率。一个好的系统应综合权衡这两者的权重,力争达到最大的吞吐率。

表 4　系统性能比较表

结构名称	器件	slice 数量	最高频率(MHz)	控制方式	吞吐率(Mbps)
串行	Virtex 4	374	242	状态机	37
本文	Virtex 4	682	233	状态机 + 流水线	116.5
	Virtex II	693	150	状态机 + 流水线	75
文献[8]	Virtex 2	725	33	流水线	33
文献[3]	0.35μm ASIC	NA	120	状态机	60
文献[5]	Virtex 2	NA	55.44	流水线	55.44

4　结　论

算术编码是图像压缩的重要组成部分,长期以来是困扰压缩速度的瓶颈之一。本文提出一种新型的分步式并行架构,此结构简单合理,FIFO 管道的引入可支持异步流水电路,状态机的动态优化策略有效地防止了流水的阻塞。复杂环路逐层分解的方法显著降低了编码的反馈效应,整个系统已经通过 FPGA 验证,最高工作频率可达 233MHz,同等频率下,加速比是串行结构的 3.3 倍。系统的吞吐率与其它同类结构相比也有大幅提升,从速度、面积和实现复杂性等多方面因素综合考虑,本文所提出的 MQ 编码 VLSI 结构具有很好的应用前景。

目前本文设计的 MQ 编码器从形式上来说是基于单输入体系的,为满足更大数据量的处理要求,设计更为高效的多输入编码器是必由之路。MQ 编码的多重输入势必会引起更大规模的反馈效应,从而降低编码器的吞吐率,因此,找到编码更新规律以简化电路的复杂性,是提高多输入 MQ 编码器性能的关键所在。

参考文献

[1] Lian C J, Chen K F, Chen H H, et al. Analysis and architecture design of block-coding engine for EBCOT in JPEG2000. *IEEE Transactions on Circuits and Systems for Video Technology*, 2003,13(3): 219-230

[2] Pastuszak G. A high-performance architecture of arithmetic coder in JPEG 2000. In: Proceedings of the IEEE International Conference on Multimedia and Expo, Taipei, Taiwan, China, 2004. 1431-1434

[3] Yu C, Hu H T. Design and implementation of an ASIC architecture for the context-based binary arithmetic encoder. In: Proceedings of the Ninth International Symposium on Consumer Electronics, Macau, 2005. 83-86

[4] Zhang Y Z, Xu C, Wang W T, et al. Performance analysis and architecture design for parallel EBCOT encoder of JPEG2000. *IEEE Transactions on Circuits and Systems for Video Technology*, 2007,17(10): 1336-1347

[5] 李萱, 郭炜. 并行通道编码 EBCOT 中 MQ 编码器的硬件设计. 信息技术,2007,31(5):51-53

[6] Tarui M, Oshita M, Onoye T, et al. High-speed implementation of JBIG arithmetic coder. In: Proceedings of the IEEE Region 10 Conference, Korea, 1999. 1291-1294

[7] 焦润海. 图像压缩中的高效预测编码及其优化实现技术:[博士学位论文]. 北京:北京航空航天大学计算机学院, 2007

[8] 朱珂. 基于 JPEG2000 的静态图像压缩算法及 VLSI 实现的结构研究:[博士学位论文]. 上海:复旦大学微电子系, 2004

A separate parallel MQ coder and its VLSI architecture

Wang Qian, Lv Dongqiang*

(Digital Media Processing Laboratory, School of Computer Science and Engineering, Beihang University, Beijing 100083)

(* The Fourth Institute of The Second Artillery Equipment Academe, Beijing 100085)

Abstract

In view of the fact that the MQ encoder's high complex loop feedback structure restricts its fast image compression hardware implementation, the paper proposes the concept that the encoding module and the bit stuffing module can be connected by FIFO channel and operated simultaneously for their independency based on the analysis of the original MQ coding algorithm, and gives the design of a separate parallel architecture combining asynchronous pipelining with the finite state machine (FSM), which is suitable for software algorithm characteristics. Based on the dynamic feature of data operation in the processing of program, the length of the critical path is reduced by the probability statistical law and the division technology of the state machine. The experimental results show that the architecture is with a high resource utilization ratio, and its throughput rate is significantly increased to 116.5Mbps at the highest working frequency of 233MHz on a field-programmable gate array (FPGA) prototype chip compared with the up-to-date design.

Key words: arithmetic coding, separate, image compression, critical path