Area Efficient Fast Huffman Decoder for Multimedia Applications

Heonchul Park, Jae-Chul Son and Seong-Rae Cho

DSP Team, Semiconductor Division

Samsung Electronics Co., Ltd.

Suwon P. O. Box 105

Kyoungki-Do, Korea

hpark@jupiter.info.samsung.co.kr

ABSTRACT

In this paper, we propose an area-efficient VLSI architecture for fast Huffman decoder which can support HDTV rates. Huffman coding, which is known as optimal variable length coding, has been widely used to reduce storage and communication channel bandwidth, and several emerging image compression standards such as JPEG, MPEG, CCITT H.261 require to perform Huffman coding in real-time with high throughput. However, most known designs are not suitable for real-time operation or for implementation, especially in HDTV, since these require large amount of VLSI area or large number of processing elements (PEs) for high performance. The proposed VLSI architecture for Huffman decoder requires less numbers of comparators and smaller size of data rotator to simulate Barrel shifter. It can decode up to 17 bits per cycle and employ 40MHz clock which can support HDTV rates. Thus, it can decode Huffman coded sequence up to 680Mbits/s at peak. Compared with parallel implementation in [3]which requires upto 1460 PEs and has 10 Mbps of throughput, the proposed architecture is a single PE design with competitive processing power. It requires 25% of the area of the known single PE design in [2].

1. INTRODUCTION

Low bit rate coding is essential for image applications such as TV transmission, video conferencing, remote sensing via satellite, computer communication, facsimile transmission, etc. Data compression techniques reduce the storage and communication channel bandwidth needed to process such signals while maintaining acceptable fidelity. Most conventional digital image compression techniques require lossless coding method at the end of the encoder to obtain additional compression. One of these lossless coding schemes is Huffman coding which provides optimal variable length coding for a fixed length input. It is recommended in international standards in MPEG, JPEG, H. 261, etc [1].

Huffman coding assigns a 0/1 sequence for each input symbol such that the total average length of the output code is minimal based on a predetermined weight of each input symbol. The Huffman code is represented as a binary tree where each leaf of the tree represents a symbol. A symbol is represented by the path from the root to the leaf. Thus, a symbol with larger weight has shorter path length.

The HDTV applications, which follows MPEG2 standard, require to process over 100Mbits/s of input rates for Huffman decoder. In Huffman decoder design, we have to consider idle time of Huffman decoder for global synchronization to ochestrate with other functional units in HDTV decoder. Also, video syntax parsing, usually performed in RISC processors, prevents decoder from its operations, since MPEG 2 video bitstream for HDTV requires to process Huffman decoding and syntax parsing sequentially. Notice that the bitstream is fed into decoder and stored into buffer during idle time. Based on these requirements, Huffman decoder for HDTV have to process over 200Mbits/s of input rates at peak performance. In addition, overall VLSI area should be smaller for one chip solution with other functional units for HDTV decoder.

Several Huffman decoder architectures have been proposed [2, 3, 4, 5]. An architecture for Huffman codec [3] has been reported using parallel implementation based on reversed Huffman tree. It results in multi-chip solution, since it requires over a thousand of PEs. In [2], it provides single chip solution which can sustain up to 100Mbits/s of input rates using hardwired table lookup approach. It leads to large VLSI area due to its PLA approach for input sequence comparison and barrel shifter for sequence manipulation. Although a compact design for Huffman codec has been shown in [5], it is difficult to support HDTV applications, since it employs bit-by-bit decoding approach and its performance is limited to 40Mbits/s of input. Recently, the mixed approach of [5] and [2] has been shown in [4] for multimedia applications which can support up to 165Mbits/s of input rates. It is targeted for MPEG2 applications for digital NTSC quality TV. Considering HDTV applications, these known solutions should be improved in performance or be smaller in VLSI area.

The proposed VLSI architecture for Huffman decoder can decode up to 17 bits per cycle and employ 40MHz clock which can support HDTV rates. Thus, it can decode Huffman coded sequence up to 680Mbits/s at peak. Compared with parallel implementation in [3]which requires upto 1460 PEs and has 10 Mbps of throughput, the proposed architecture is a single PE design with competitive processing power.It requires 25% of the area of the known single PE design in [2].

This paper is organized as follows: in the next section, Huffman coding is briefly introduced. In Section 3, VLSI

0-7803-2431-5/95 $4.00

architecture for the proposed Huffman decoder is shown and concluding remarks follow.

2. HUFFMAN CODING

A major obstacle in many image applications is the vast amount of storage required to directly represent digital images. A digitized color picture at TV resolution contains on the order of 1 Mbyte of information and 35mm film resolution has ten times that amount. Use of digital images often is not viable due to high storage, associated transmission costs, and computational costs, even though image capture and display devices for digitized images are currently available.

A common feature of JPEG, MPEG, and p x 64 kbits/s standards is that they recommend entropy coding at the end of the encoder. It provides additional lossless compression by encoding the lossy compressed output. Huffman coding and arithmetic coding schemes are well known entropy coding methods. In many cases, arithmetic coding results in 5-10% less output size than Huffman coding. However, the computational requirements for the arithmetic coding is very high, compared with Huffman coding. It can save from 20 % to 90 % of amount of storage or communication channel bandwidth needed,depending on the characteristics of the input being compressed. No information loss occurs after decoding. Table 1 shows the throughput requirements when the MPEG standard is applied to higher image quality motion pictures than TV quality. Even though it may not show exact throughput requirement, it clearly shows that it is necessary to design Huffman codecs with higher throughput than the current MPEG rates.

3. VLSI ARCHITECTURE

Assume we have a tree corresponding to Huffman codes, such that all the leaves are to the left side of all the internal nodes in any level of the tree. It is known that any arbitrary Huffman tree can be transformed into such a tree without increasing the average code length [5]. Each edge has a label '1' (left edge) or '0' (right edge) such that decoding corresponds to traversal from the root corresponding to the symbol to the leaf based on input value. Leaves are grouped based on its number of consecutive '0's within code from the left. The number of group is less than or equal to the number of levels in the Huffman tree and each group becomes a sub-tree of the Huffman tree. Then, the decoding sequence is as follows: For a given bitstream, count number of consecutive '0's from the left and find the group. Within the group, use the remained bitstream as an address of ROM table which stores corresponding output symbol. Notice that Huffman decoding table is already fixed before decoding. One symbol is found, then shift left the bitstream corresponding to the length of the code. Repeat again.

Overall block diagram is shown in Fig. 1. The width of D-latch is max, where max is the length of the longest code word. Also, input width of codeword block in Fig. 1 is max. Group search is performed by hardwired comparators and the rest of bitstream is fed into address decoder such that a location of ROM, which has symbol, is selected. The number of comparators is same as the number of groups in the Huffman tree. The width of comparator corresponds to the number of consecutive '0's in each group. The length of bitstream fed into address decoder depends on the number of symbols in each group. The length of current code word is fed into adder such that the next bitstream is identified using the address decoder output. The adder and latches in Fig. 1 compute modulo max operation to control update of D-latches and compute the number of bits to shift left. If the adder value is greater than or equal to max, then carry signal is generated to request D-latch update and the remainder of modulo max is used for shift left operation. Otherwise, adder value is used for shift left operation without D-latch update.

The data selection of length max from the bitstream in the two D-latches is performed by data rotator and 2:1 MUX. The two D-latches hold bitstream of length 2x max, a0, a1,..., a2 x max. Assume we have to shift left i positions, where $0 < i < max + 1$, to obtain ai, ..., ai + max - 1 for the next table lookup. Using 2:1 MUX and input i which generates control signal for 2:1 MUX via unary decoder, obtain amax,..., amax + i - 1, ai,..., amax - 1. There are max 2:1 multiplexers in MUX and input of the k-th multiplexer is ak and amax + k, $0 \leq k \leq$ max. The next step is to rotate these bitstream reduced by half in length using data rotator. The rotator consists of $\lceil \log max \rceil$ stages. Also, the input value i is used to control data rotator and the j-th bit of the binary number i is connected to the j-th stage. In the data rotator, the j-th stage shifts left data 2^j bits, $0 \leq j \leq \lceil \log max \rceil - 1$, if control is 1. Otherwise, it does not shift data. Fig. 2 shows a blockdiagram of the rotator for max = 4. The search and data shift can be performed in one cycle, since there is no sequentiality.

Consider a discrete cosine transform coefficient retrieval which is most frequently used in MPEG Huffman decoding. In MPEG2 standard, there are 114 codewords which longest codeword is 17-bit except escape code. Compared to the design in [2] which requires 114 comparators and the average length of each comparator is 12.8 bits, the proposed Huffman decoder employs 13 comparators and its average length is 8.2 bits. Also, the data rotator has 25% area of the barrel shifter in [2]. In addition, ROM table takes 3/4 area of [2], since the proposed scheme does not store the code length information in ROM.

4. CONCLUSION

In this paper, we employ fast table lookup approach. For a given input bitstream which is stored in data latch, data matching is performed using hardwired comparators and codeword corresponding to input code output and the next

data is selected using data manipulators. The proposed architecture for Huffman decoder based on the above algorithm requires a simple control logic. Most of the area is consumed by the hardwired comparators and data manipulator. The parallel implementation in [2] requires 14 60 PEs to obtain 10 Mbps throughput. Compared to the design in [3], the proposed design requires approximately 25% less area if the same CMOS technology is assumed, since the number of comparators is 1/12 of [2] and the data manipulator part is 1/4 of [3]. In addition, the proposed design is implemented in VLSI using 0.65 micron CMOS technology and has an area of 3.5 mm2. Since it employs 40 MHz clock and can decode 17-bit codeword in a cycle, it can support up to 680Mbits/s of input rates, which is suitable for multimedia applications including HDTV.

References

[1] E. A. Fox, "Advances in interactive digital multimedia systems," IEEE Computer, pp. 9-21, Oct. 1991.

[2] S.-M. Lei and M.-T. Sun, "An entrophy coding system for digital HDTV applications," IEEE Trans. on Circuits and Systems for Video Technology, pp. 147-155, 1991.

[3] A. Mukherjee, N. Ranganathan, and M. Bassiouni, " Efficient VLSI designs for data transformation of tree-based codes," IEEE Trans. on Circuits and Systems, pp. 306-314, 1991.

[4]Y. Ooi, A. Taniguchi and S. Demura, "A 162Mbi/s variable length decoding circuit using an adaptive tree search technique," IEEE Custom Integrated Circuits Conference, 1994.

[5] H. Park and V. K. Prasanna, "Area efficient VLSI architectures for Huffman coding," IEEE Trans. on Circuits and Systems-II, pp. 568-575, 1993.

Image class	Size	Display frequency	Compressed bit rate
SIF	352 X240	30Hz	1.2-3 Mbps
CCIR	720 x 480	30Hz	4-15 Mbps
HDTV	1920 x 1080	30Hz	20-115Mbps

Table 1. Compressed bitrates for various images

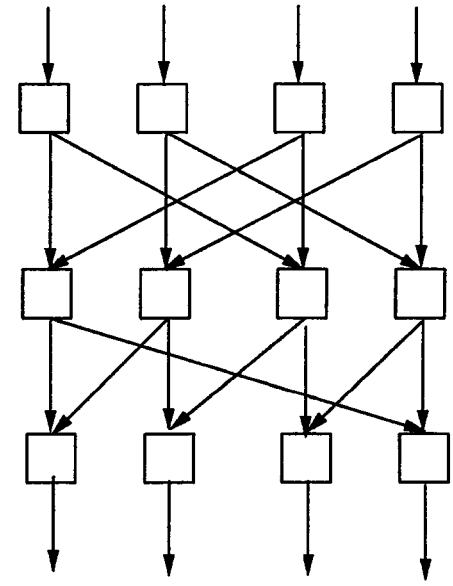

Figure 2. Data rotator for max = 4 (each box denotes 2:1 MUX)

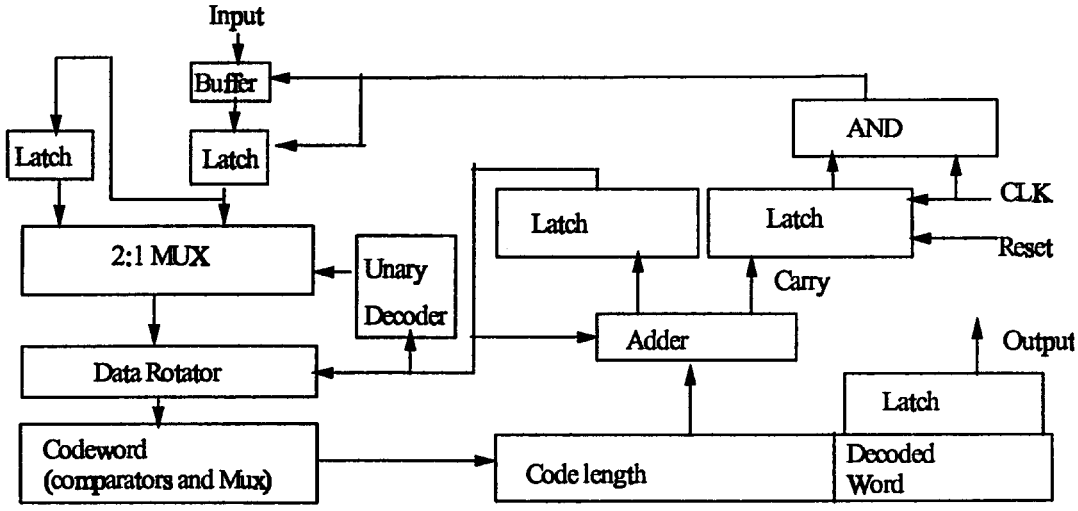

Figure 1. Block diagram of the proposed Huffman decoder