International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Development of Huffman Encoder For MPEG-2 Video Applications

Dr.Tumu Srinivas Reddy[1], Dr N.Subbu Lakshmi[2]
Associate Professor[1], Professor[2]
Department of Electronics and Communication Engineering
Malla Reddy Engineering College

Abstract- The project deals with a Huffman encoder and a method for Huffman encoding in which a data compression rate can be easily changed and a recording medium having a program for a Huffman encoding process recorded there on. A quantizer quantizes DCT coefficients output by a DCT device to output quantized DCT coefficients. A comparator judges a quantized DCT coefficient to be an invalid coefficient when the absolute value of the DCT coefficient is equal to or smaller than a threshold and judges the quantized DCT coefficient to be a valid coefficient when the absolute value of the DCT coefficient is greater than the threshold. A run length counter counts the number of consecutive invalid coefficients to output run lengths and outputs valid coefficients. An encoder performs encoding based on the valid coefficients and the run lengths to output encoded data. Image data includes a huge amount of information. It is therefore impractical to process image data as it is from the viewpoint of the memory capacity required there for and communication speed. Techniques for compressing image data are therefore important. One of international standards for image data compression is the JPEG (Joint Photographic Expert Group) standard. JPEG adopts the DCT (discrete cosine transformation) method which involves irreversible encoding and the reversible encoding method which involves DPCM (differential PCM) in a two-dimensional space. The compression of image data according to the DCT method will now be described. In the Huffman encoder according to the invention, the determination means determines whether each item of a series of data is within a predetermined range. The encoding means treats data among the series of data which are within the predetermined range as invalid coefficients, treats data out of the predetermined range as valid coefficients and performs encoding using combinations of the number of consecutive invalid coefficients and valid coefficients. The Huffman encoder may further comprises quantization means for quantizing DCT coefficients using predetermined quantization coefficients and for supplying quantized DCT coefficients to the determination means as a series of data.
Index Terms- DCT, Joint Photographic Expert Group, DPCM, Huffman encoder

I. INTRODUCTION

Huffman coding is an entropy encoding algorithm used for lossless data compression. The term refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file) where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. It was developed by David A. Huffman while he was a Ph.D. student at MIT, and published in the 1952 paper "A Method for the Construction of Minimum-Redundancy Codes". Huffman coding uses a specific method for choosing the representation for each symbol, resulting in a prefix code (sometimes called "prefix-free codes") (that is, the bit string representing some particular symbol is never a prefix of the bit string representing any other symbol) that expresses the most common characters using shorter strings of bits than are used for less common source symbols. Huffman was able to design the most efficient compression method of this type: no other mapping of individual source symbols to unique strings of bits will produce a smaller average output size when the actual symbol frequencies agree with those used to create the code. A method was later found to do this in linear time if input probabilities (also known as weights) are sorted. For a set of symbols with a uniform probability distribution and a number of members which is a power of two, Huffman coding is equivalent to simple binary block encoding, e.g., ASCII coding. Huffman coding is such a widespread method for creating prefix codes that the term "Huffman code" is widely used as a synonym for "prefix code" even when such a code is not produced by Huffman's algorithm. Although Huffman coding is optimal for a symbol-by-symbol coding (i.e. a stream of unrelated symbols) with a known input probability distribution, its optimality can sometimes accidentally be over-stated. For example, arithmetic coding and LZW coding often have better compression capability. Both these methods can combine an arbitrary number of symbols for more efficient coding, and generally adapt to the actual input statistics.

II. EXISTING WORK OR LITERATURE SURVEY

VHDL (VHSIC (Very High Speed Integrated Circuits) hardware description language) is commonly used as a design-entry language for field-programmable gate arrays and application-specific integrated circuits in electronic design automation of digital circuits. VHDL was originally developed at the behest of the US Department of Defense in order to document the behavior of the ASICs that supplier companies were including in equipment. That is to say, VHDL was developed as an alternative to huge, complex manuals which were subject to implementation-specific details. The idea of being able to simulate this documentation was so obviously attractive that logic

simulator was developed that could read the VHDL files. The next step was the development of logic synthesis tools that read the VHDL, and output a definition of the physical implementation of the circuit. Modern synthesis tools can extract RAM, counter, and arithmetic blocks out of the code, and implement them according to what the user specifies. Thus, the same VHDL code could be synthesized differently for lowest area, lowest power consumption, highest clock speed, or other requirements. VHDL borrows heavily from the Ada programming language in both concepts (for example, the slice notation for indexing part of a one-dimensional array) and syntax. VHDL has constructs to handle the parallelism inherent in hardware designs, but these constructs (processes) differ in syntax from the parallel constructs in Ada (tasks). Like Ada, VHDL is strongly-typed and is not case sensitive. There are many features of VHDL which are not found in Ada, such as an extended set of Boolean operators including nand and nor, in order to directly represent operations which are common in hardware. VHDL also allows arrays to be indexed in either direction (ascending or descending) because both conventions are used in hardware, whereas Ada (like most programming languages) provides ascending indexing only. The reason for the similarity between the two languages is that the Department of Defense required as much of the syntax as possible to be based on Ada, in order to avoid re-inventing concepts that had already been thoroughly tested in the development of Ada.VHDL is a fairly general-purpose language, and it doesn't require a simulator on which to run the code. There are a lot of VHDL compilers, which build executable binaries. It can read and write files on the host computer, so a VHDL program can be written that generates another VHDL program to be incorporated in the design being developed. Because of this general-purpose nature, it is possible to use VHDL to write a testbench that verifies the functionality of the design using files on the host computer to define stimuli, interacts with the user, and compares results with those expected. VHDL is a strongly typed language. It is relatively easy for an inexperienced developer to produce code that simulates successfully but that cannot be synthesized into a real device, or is too large to be practicable. One particular pitfall is the accidental production of transparent latches rather than D-type flip-flops as storage elements. VHDL is not a case sensitive language. One can design hardware in a VHDL IDE (such as Xilinx or Quartus) to produce the RTL schematic of the desired circuit. After that, the generated schematic can be verified using simulation software (such as ModelSim) which shows the waveforms of inputs and outputs of the circuit after generating the appropriate testbench. To generate an appropriate testbench for a particular circuit or VHDL code, the inputs have to be defined correctly. For example, for clock input, a loop process or an iterative statement is required. The key advantage of VHDL when used for systems design is that it allows the behavior of the required system to be described (modeled) and verified (simulated) before synthesis tools translate the design into real hardware (gates and wires). Another benefit is that VHDL allows the description of a concurrent system (many parts, each with its own sub-behavior, working together at the same time). VHDL is a Dataflow language, unlike procedural computing languages such as BASIC, C, and assembly code, which all run sequentially, one instruction at a time.

FPGA: A field-programmable gate array (FPGA) is a semiconductor device that can be configured by the customer or designer after manufacturing—hence the name "field-programmable". FPGAs are programmed using a logic circuit diagram or a source code in a hardware description language (HDL) to specify how the chip will work. They can be used to implement any logical function that an application-specific integrated circuit (ASIC) could perform, but the ability to update the functionality after shipping offers advantages for many applications. FPGAs contain programmable logic components called "logic blocks", and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together"—somewhat like a one-chip programmable breadboard. Logic blocks can be configured to perform complex combinational functions, or merely simple logic gates like AND and XOR. In most FPGAs, the logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory. Historically, FPGAs have been slower, less energy efficient and generally achieved less functionality than their fixed ASIC counterparts. A combination of volume, fabrication improvements, research and development, and the I/O capabilities of new supercomputers have largely closed the performance gap between ASICs and FPGAs.[14] Advantages include a shorter time to market, ability to re-program in the field to fix bugs, and lower non-recurring engineering costs. Vendors can also take a middle road by developing their hardware on ordinary FPGAs, but manufacture their final version so it can no longer be modified after the design has been committed. Xilinx claims that several market and technology dynamics are changing the ASIC/FPGA paradigm [15].

DATA COMPRESSION: Data compression or source coding is the process of encoding information using fewer bits (or other information-bearing units) than an unencoded representation would use through use of specific encoding schemes. As with any communication, compressed data communication only works when both the sender and receiver of the information understand the encoding scheme. For example, this text makes sense only if the receiver understands that it is intended to be interpreted as characters representing the English language. Similarly, compressed data can only be understood if the decoding method is known by the receiver. Compression is useful because it helps reduce the consumption of expensive resources, such as hard disk space or transmission bandwidth. On the downside, compressed data must be decompressed to be used, and this extra processing may be detrimental to some applications. For instance, a compression scheme for video may require expensive hardware for the video to be decompressed fast enough to be viewed as it's being decompressed (the option of decompressing the video in full before watching it may be inconvenient, and requires storage space for the decompressed video). The design of data compression schemes therefore involves trade-offs among various factors, including the degree of compression, the amount of distortion introduced (if using a lossy compression scheme), and the computational resources required to compress and uncompress the data. Lossless compression algorithms usually exploit statistical

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

redundancy in such a way as to represent the sender's data more concisely without error. Lossless compression is possible because most real-world data has statistical redundancy. For example, in English text, the letter 'e' is much more common than the letter 'z', and the probability that the letter 'q' will be followed by the letter 'z' is very small. Another kind of compression, called lossy data compression or perceptual coding, is possible if some loss of fidelity is acceptable. Generally, a lossy data compression will be guided by research on how people perceive the data in question. For example, the human eye is more.

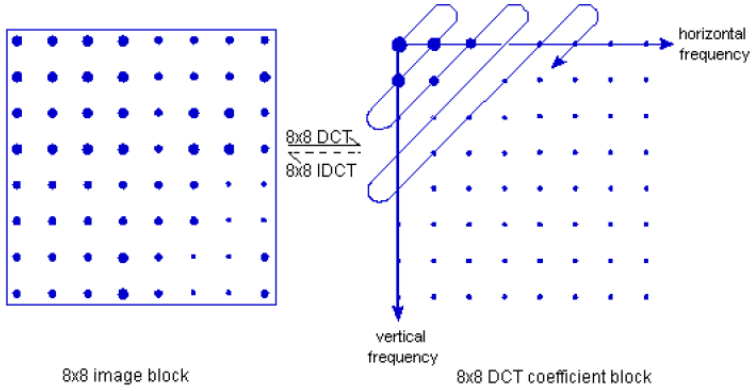
MPEG-2 is a standard for "the generic coding of moving pictures and associated audio information". It describes a combination of lossy video compression and lossy audio data compression methods which permit storage and transmission of movies using currently available storage media and transmission bandwidth. MPEG-2 is widely used as the format of digital television signals that are broadcast by terrestrial (over-the-air), cable, and direct broadcast satellite TV systems. It also specifies the format of movies and other programs that are distributed on DVD and similar discs. As such, TV stations, TV receivers, DVD players, and other equipment are often designed to this standard. MPEG-2 was the second of several standards developed by the Moving Pictures Expert Group (MPEG) and is an international standard (ISO/IEC 13818). Parts 1 and 2 of MPEG-2 were developed in a joint collaborative team with ITU-T, and they have a respective catalog number in the ITU-T Recommendation Series. While MPEG-2 is the core of most digital television and DVD formats, it does not completely specify them. Regional institutions can adapt it to their needs by restricting and augmenting aspects of the standard. See Profiles and Levels. MPEG-2 includes a Systems section, part 1, that defines two distinct, but related, container formats. One is the Transport Stream, designed to carry digital video and audio over possibly lossy media, such as broadcasting, examples of which include ATSC, DVB and SBTVD. MPEG-2 Systems also defines Program Stream, a container format designed for reasonably reliable media such as optical discs, DVDs and SVCDs. MPEG-2/System is formally known as ISO/IEC 13818-1 and as ITU-T Rec. H.222.0.[2] The Video section, part 2 of MPEG-2, is similar to the previous MPEG-1 standard, but also provides support for interlaced video; the format used by analog broadcast TV systems. MPEG-2 video is not optimized for low bit-rates, especially less than 1 Mbit/s at standard definition resolutions. However, it outperforms MPEG-1 at 3 Mbit/s and above. All standards-compliant MPEG-2 Video decoders are fully capable of playing back MPEG-1.

III. WRITE DOWN YOUR STUDIES AND FINDINGS

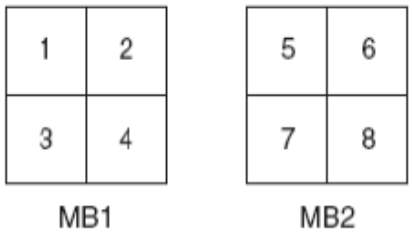
MPEG is a non-adaptive coding system; the code table does not change with the input video sequence. Many image sequences were coded and the statistics used to define the entries in the MPEG-2 Huffman table. JPEG still image compression, using adaptive entropy coding, where the table entries can be modified to better suit a particular picture. JPEG also uses adaptive arithmetic coding where the code table is changeable depending on the change in the statistics of the JPEG image. MPEG defines a set of variable-length code (VLC) for each of the probable run/level combinations. The run/level combinations not found in the table are represented by an escape code followed by a six-bit code for the run and an eight or 16-bit code for the level. The end of block (EOB) code is used when all the remaining coefficients in the 8 x 8 block are zeroes. Coding of the 8 x 8 block starts from the DC coefficient in the zigzag order. When there are no more nonzero coefficients remaining in the zigzag order, the EOB code is used to terminate coding. Since the probability of occurrence of the EOB symbol is high, it is assigned a two-bit code "10". The VLC tables used in MPEG-2 are not true Huffman codes. They are optimized for a range of bit rates to sample rate ratios. Most of the code words in the MPEG-2 tables were carried over from the H.261 standard. The DCT coefficient tables in MPEG-2 assume equal probability for both positive and negative coefficients. MPEG-2 VLC uses a new table. It is better suited for the statistics of intra-coded blocks. The EOB code for has two bits but for the EOB has four bits. This implies that an intra-coded block can have on average of 24 or 16 non-zero AC coefficients. For non intra-coded blocks, the statistics point to an average of 22 or four non-zero AC coefficients. Both have 113 entries. The VLC table consists of Huffman codes for different run/level combinations. The last bit "s" of the code denotes the sign of the level with s = 0 for positive and s = 1 for negative. The VLC table also includes the EOB code to indicate the status of the rest of the coefficients aszero. The EOB cannot occur as the only code in a block since no coding was done in the block. There are run/level combinations not defined in the VLC tables. When the variable length coder sees an undefined run/level combination, it codes the run into a 6-bit binary value and the level into a

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

12-bit signed level value. Before coding an undefined run/level pair, a 6-bit "escape code" is used to denote that the next six to12 bits are not from the VLC table.



DC COEFFICIENTS: Due to high redundancy between adjacent quantized DC coefficients of 8 x 8 blocks, the difference in DC values is encoded using VLC. The difference signals range from –255 to 255 in MPEG-1 and from –2047 to 2047 in MPEG-2. The size of the differential DC value (dct_dc_size) is found in Table 1. The size denotes the number of bits used to represent a particular value (e.g., if the differential DC value is –78, the table shows a size seven. The seven bits will be used to represent the value –78. The dct_dc_size value is variable length coded using table Table 5 or Table 6. After coding the size bits, –78 is represented as 111110, 0110010 where the first six bits define the dct_dc_size and the next seven bits are used to code the value –78. Two different VLC codes are used for coding the dct_dc_size for luma and chroma. The difference path for 8 x 8 blocks within a macro block are calculated in the order shown in the following figure.



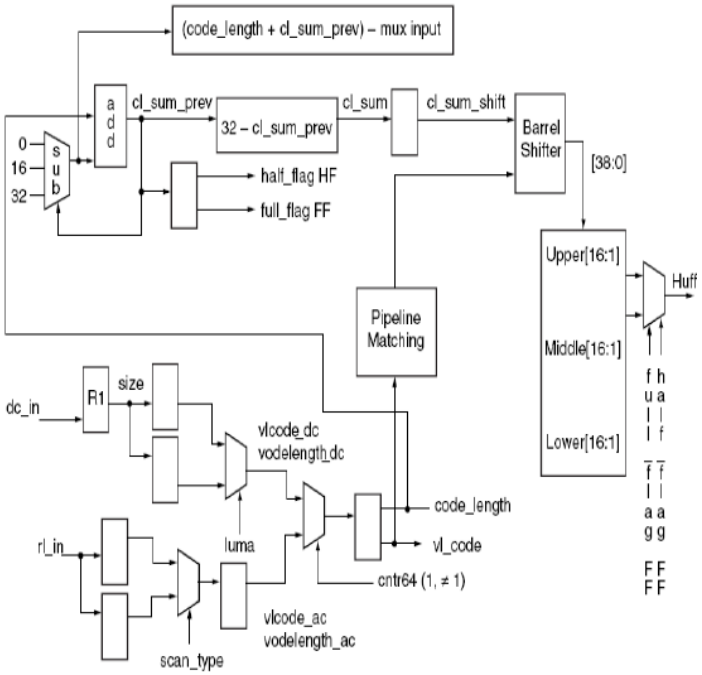
IV. RESULTS AND DISCUSSION

The Huffman tables used for coding AC coefficients are selected based on the macroblock type and intra_vlc_format value according to Table 2. There are two huffman codes for the run/level combination of 0/1. The code "1s" is used if the 0/1 pair represents the first coefficient or the DC coefficient in the block. For subsequent 0/1 run/level pairs, "11s" is used as the Huffman code. The "s" in the code denotes the sign of the coefficient, "0" for positive and "1" for negative. The run/level pair for DC coefficient of "+1" and the EOB code has the same Huffman code. The differentiation is apparent since the EOB code will not be the first code in the block. In intra coding, since the DC value is coded separately, the first coded symbol is the first AC value. In this case, this first coded value can have a run/level of 0/1. A Huffman code of "11s" will not conflict with the EOB symbol. For a run/level pair that is not defined in the Table, an escape code is used, followed by a 6-bit run symbol and 12-bit levelDC COEFFICIENT: Three predictor values are maintained for each color component. The predictor values are set at the start of the slice, or when a non-intra macroblock is decoded, or when a macroblock is skipped. The predictor values for different intra_dc_precisions are shown in Table.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

intra_dc_precision	Bits of Precision	Reset Value
0	8	128
1	9	256
2	10	512
3	11	1024

The DC coefficient is decoded by first getting the differential value from the coded stream. This differential_dc value is then added to the predictor to get the actual DC value. The new predictor value then becomes the actual DC value just decoded. AC COEFFICIENT: The run/level pair is decoded from the Huffman code using a look-up table (LUT). There are three possible options for the Huffman code. If the code represents an EOB, all the remaining coefficients are set to "0". If the code represents an escape code, the next six bits represent the run and the following 12 bits represent the level. If the VLC denotes a normal coefficient, then the run coefficients are set to zero and the following level coefficient is set to the level value depending on the value of s. When s == 0, the signed level is the same as level. When s == 1, the signed level is equal to (–level).



Block diagram

V. CONCLUSION

This Huffman coding application note describes the Huffman coding algorithms used in an MPEG-2 encoder. The reference design files show the efficient implementation of the algorithms on Xilinx devices. The code can be used to target any Xilinx device. The code can be optimized by instantiating the adder/subtractor and multiplier units when targeting Virtex devices. Adding more pipeline stages can increase the performance of both blocks. The code takes in variable length data and sends out a fixed length (16-bit) Huffman coded data.

REFERENCES

1. MPEG-2 Video IS document, ISO/IEC 13818-2: 1995(E)
2. MPEG Video Compression Standard, by Mitchell, Pennebaker, Fogg, and LeGall, ISBN 0-412-08771-5
3. www.xilinx.com 4. www.wikipedia.org