BIT-PLANE AND PASS DUAL PARALLEL ARCHITECTURE FOR COEFFICIENT BIT MODELING IN JPEG2000

Chao Xu, Yanju Han, and Yizhen Zhang

Peking University, Beijing, 100871, CHINA

ABSTRACT

In this paper, the bit-plane and pass dual parallel architecture for coefficient bit modeling in JPEG2000 is proposed. It is a very high speed and efficient structure that is capable of encoding all bits of the wavelet coefficient in only one scan, and largely decreases the memory requirement. Additionally in order to decrease the logic circuit requirement we propose a partial primitive-parallel technique to replace the whole channel-parallelism. Experimental results show that the architecture can encode about 15 times more than the pass-parallel encoding for the coefficient with 16 bits. And it only requires 3K bits memory instead of 16K bits for the pass-parallel encoding.

1. INTRODUCTION

JPEG2000 is the latest image compression standard which has been developed to meet the demand for efficient, flexible, and interactive image representation [1]. The central concept in JPEG2000 is scalability based on EBCOT with the fractional bit-plane (FBP) coding technique. The FBP coding divides each bit-plane coding into 3 coding passes that makes the code stream more scalable but the coding computation more complex. For example, if the wavelet coefficient is of 16 bits, one sign bit, 15 magnitude bits, the FBP coding has to suffer from 14×3+1=43 scans that usually take over 50% of the whole coding time. The 5-level wavelet decomposition takes less than 25% of the coding time.

To reduce the FBP coding time, many methods have been presented, such as the subband-parallel coding [2], the optimal serial coding [3] and the pass-parallel coding [4][5]. The methods increase the coding speed 2 to 3 times.

Here we present a novel method of the bit-plane and pass dual parallel (BPDP) coding that can increase the coding speed 43 times or more and can complete the FBP coding within one scan.

The rest of the paper is organized as follows. In Section 2, a brief description about the FBP coding, called coefficient bit modeling in JPEG2000, is given. In Section 3, the issues of the BPDP coding are discussed. And the corresponding architecture is presented in Section 4. Experimental results are illustrated in Section 5, and the paper is concluded in Section 6.

2. THE FBP CODING ALGORITHM

The JPEG2000 encoder mainly contains color transformation, wavelet transformation, quantization, coefficient bit modeling (the FBP coding), arithmetic coding and data ordering. The FBP coding is performed on wavelet coefficients to produce pairs of context (CX) and decision (D) that are provided to arithmetic coding.

Before the FBP coding, each wavelet subband is partitioned into rectangular blocks called code-blocks whose typical dimensions are 64×64 or 32×32 coefficients. In this paper, the 64×64 code blocks are used as usual. Each code-block is encoded independently, bit-plane by bit-plane, and pass by pass in each bit-plane. The bit-planes are ordered from most significant magnitude bit-plane to least significant magnitude bit-plane, and the most significant bit-planes with all zero bits are skipped.

The FBP coding consists of 3 coding passes in each bit-plane coding: significance propagation pass (SP), magnitude refinement pass (MP) and cleanup pass (CP). And 3 associated binary state variables are required.

1) Coefficient significance state σ, changes from insignificant σ=0 to significant σ=1 whenever the most significant bit ν_{pM} equal to one is encoded.
2) Coefficient refinement state σ', changes from 0 to 1 whenever the refinement is performed, in fact, when the bit ν_{pM-1} is encoded.
3) Magnitude bit ν_p coded state η, changes from 0 to 1 once the bit ν_p is encoded.

In a bit-plane coding via the 3 variables the SP is applied to the coefficient X that is insignificant and has at least one significant neighbor coefficient among the 8 neighbor coefficients shown in the left of Figure 1. The MP is used to the coefficient that has become significant. The CP is utilized to all the remaining coefficients. The 3 coding passes produce the pair of CX and D with 4 coding primitives below.

1) Zero coding (ZC), is used in the SP and the CP for the magnitude bit ν_p ($p \geq pM$) to create 9 pairs of CX

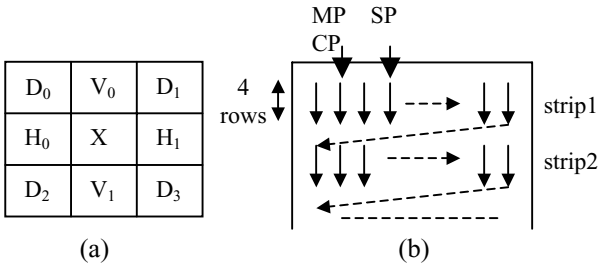

Fig. 1: (a) X and its 8 neighbors.
(b) Scan pattern of a code block.

and D. The D is equal to the bit ν_p.

2) Sign coding (SC), is used in the SP and CP and only follows the bit ν_{pM} to create 5 pairs of CX and D. This D is generated by exclusive OR operation of the sign bit and the XOR bit relative to the context.
3) Magnitude refinement coding (MRC), is used only in the MP for the magnitude bit ν_p ($p < pM$) to create 3 pairs of CX and D. This D is equal to the ν_p.
4) Run length coding (RLC), is used in the CP for a column of ν_p ($p \geq pM$) to create 2 pairs of CX and D. For one CX, the D is zero if the 4 ν_p bits of the column are all equal to zero. For the other CX, the D is of 2 bits representing the position of the first ν_{pM}.

The 4 primitives are employed at individual situation.

The 3 passes work in a special scan pattern, as shown in the right of Figure 1, which starts at the top left of a code-block, scans the first 4 coefficients of the first column, then the first 4 coefficients of the second column and so on. Every 4 rows form a strip. The scanning is done strip by strip till the bottom right of the code-block.

In the first most significant bit-plane with non-zero elements, only the CP is utilized in terms of the primitive rules. In the other lower bit-planes, the SP is applied first; and then the MP is used; finally the CP is utilized. So if the wavelet coefficient has 15 magnitude bits, 14×3+1=43 scans are required for the FBP coding. Obviously, it is hard work especially for the real-time applications.

3. DUAL PARALLEL CODING

A large number of scans consume not only much time but also a lot of power for accesses to memory. To solve the problem, parallel processing is usually a good solution. In JPEG2000, each code-block is independent. So the code block parallel coding is easy to implement, but it requires much more circuit resource. As in [2], 3 code-blocks parallel coding requires 3 independent coding channels. If to encode one coefficient per cycle, 43 independent channels have to be provided, that will cost a lot of circuit resource. For memory only, 20×43=860K bits are needed.

Here we present the bit-plane and pass dual parallel (BPDP) coding. It can encode one coefficient per cycle, and it does not need 43 independent channels. As the BPDP coding encodes only one coefficient rather than 43 coefficients, it does not need 43 independent coding channels. And it requires much less memory. Here we presented a partial primitive-parallel technique that designs the number of primitives instead of channels in terms of requirement.

In the following, the preprocessing part is introduced first, and then the BPDP coding is discussed.

3.1. Preprocessing

The preprocessing that makes each bit-plane coding independent is the basis of the BPDP coding. It tries to produce the two associated variables σ_p, σ'_p for each magnitude bit ν_p. The other variable η_p needs only to initialize to zero before encoding.

As introduced in Section 2, the σ_p changes from zero to one when the first significant bit ν_{pM} is encoded. Thus the σ_p ($p \geq pM$) must be zero; otherwise it equal to one. The σ_p can be derived from the higher magnitude bits of the coefficient. Assume a coefficient with P-1 magnitude bits, and use the symbol "#" as "OR" logic operator.

$$\sigma_p = \nu_{p+1} \# \nu_{p+2} \# \ldots \# \nu_{P-2} \quad (1)$$

Similarly, it can be derived that the σ'_p ($p \geq pM-1$) must be zero; otherwise equal to one, $\sigma'_p = \sigma_{p+1}$. So in the BPDP coding, the σ'_p can be neglected.

For pass parallel encoding, the key is to provide the right variables σ_p for each pass, even if it is changed in the SP or CP. In terms of the order of the encoding, for the MP and CP, the σ_p changed in the SP is required. While for the SP and MP, the σ_p must not be changed in the CP. A general solution is to have the SP start earlier than the MP, and to have the CP start later than the MP. However the method can consume much memory.

For simplicity, we add a variable α_p to represent the change of the σ_p in the CP, and let the σ_p change only in the SP. Thus, the MP and CP can start simultaneously and much memory can be saved. In the SP and MP, use the σ_p for encoding, while in the CP, the $\sigma_p \# \alpha_p$ is used.

3.2. The BPDP Coding

With the σ_p, α_p and σ_{p+1}, each magnitude bit ν_p and the sign bit χ of the coefficient can be encoded independently. As the sign bit χ only follows the first significant bit ν_{pM} to be encoded, it needs only one SC primitive. Similarly, only one MRC primitive is required for the first refinement bit ν_{pM-1}. The ZC and RLC primitives can be needed for all magnitude bits of a coefficient, so P-1 sets of ZC and RLC primitives are provided.

As mentioned above, the SP should start earlier than the MP and CP. So two coefficients are encoded at a time, correspondingly two sets of ZC and SC primitives are required respectively for the SP and CP for the two

coefficients.

In the CP, the RLC primitive is used to encode one column of bits. Before it is used, it needs to evaluate each bit of the column whether the condition holds. The evaluation can delay the encoding process. To avoid the delay, we add a column state variable δ, which is decided by the preceding SP or CP. When the column has all CX=0 and all σ=0 in the SP, set the δ to 1. When the column has all σ=0 in the CP, also set δ=1. So if the previous, current and following columns all have δ=1, the RLC primitive can be used. If only the following column has δ=0, then evaluate whether all the σ equal to zero. When it is true the RLC primitive is used.

Table 1 lists the numbers of the primitives and the memory required in some typical methods. The serial coding needs 4 primitives, 20K bits memory and 43 scans. Its coding speed is assumed to be 1 as a benchmark. The subband parallel coding [2] requires 12 primitives and 60K bits memory, and increases the coding speed to 3. The optimal serial coding [3] needs the primitives and memory as much as the serial, and enhances the speed to 2.5 around. The pass parallel coding [4] needs 6 primitives, and increases the speed to 3. Our BPDP coding requires 47 primitives, but reduces the memory to 3K bits (discussed below), and increases the speed to 43.

Table 1. Resource and performance comparison with other architectures

	ZC	RLC	SC	MRC	Total	Memory	Scans	Speed
serial	1	1	1	1	4	20Kbits	43	1
subband parallel	3	3	3	3	12	60Kbits	15	3
Optimal serial	1	1	1	1	4	20Kbits	43	2.5
Pass parallel	2	1	2	1	6	16Kbits	15	3
Dual parallel	29	15	2	1	47	3Kbits	1	43

3.3. Memory Requirement

Why the memory requirement is decreased in the BPDP coding? In the serial method, the coefficient is encoded pass by pass and bit-plane by bit-plane. The associated variables have to be stored through all the pass and bit-plane encoding. And the coefficient bits that can be coded in any pass need be stored for all the 3 passes of one bit-plane. So for the 64×64 code block, 64×64×5=20K bits memory is required. However, in our BPDP method, all passes and all bit-planes are completed in one scan. The time of keeping the coefficient bits and the associated variables is shortened greatly, even much less than one pass. So the memory is reduced correspondingly. The number of the scan is also reduced to one that saves the power considerably.

In the BPDP coding, the coefficient and associated variables are kept till the encoding of the coefficient and its neighbor coefficients finishes. And the time must include the interval between the SP and the MP, CP, which is determined by the primitives. The MRC, ZC, SC primitives encode merely using information of the 8 neighbors. But the RLC primitive encodes with information of the adjacent whole columns. So for the RLC primitive, the interval should be two columns of 8 coefficients. However, if the neighbor coefficients locate in other strips, more time is needed.

In JPEG2000, a simple "vertical causal" mode is provided that treats the neighbor coefficients in the following strip as insignificant coefficients. However the neighbor coefficients in the previous strip must be regarded as those in the current strip. Since the neighbor coefficients are limited among the 8 neighbors, only the last row of the previous strip is needed. And merely the χ and σ, α are needed for encoding. So the memory for the previous strip is 64×(15×2+1)≈2K bits.

In the current strip, the interval between the SP and the MP, CP is two columns. And the MP and CP (the RLC primitive) need the information of the previous column. The SP requires only the information of the first two coefficients of the following column. So the information of 4 columns plus two (4×4+2=18) coefficients is required, the corresponding memory is 18×(15×4+1)≈1K bits. So the entire memory for the strip and the previous strip is 3K bits around.

4. ARCHITECTURE

Based on the BPDP coding, we design the architecture with pipeline mode. First load the coefficients from the external memory, and preprocessed. Then send them and the associated variables σ, α and η into a group of shifters. After shifted, they are transferred to the corresponding primitives in the bit-plane order except for the SC and MRC primitive. The SC primitive can use information from any magnitude bit-plane. The MRC primitive can use information from any magnitude bit-plane except the highest magnitude bit-plane. The primitives produce the pairs of CX and D and organize for each pass.

The preprocessor is a simple OR logic circuit as given in Equation 1. The group of shifters appears as a rectangular grid as shown in the left of Figure 2. Its each column stores a coefficient and each row stores all the related bits in one bit-plane. The bits in each column are right shifted with clock cycle. Some specific bit positions are connected to the corresponding primitives as input or output.

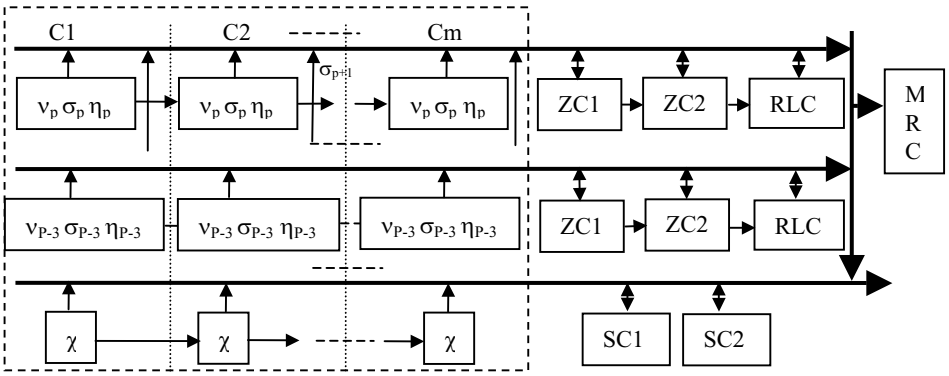

Fig. 2 Shifters and primitives in the dual parallel architecture

For the sign bit χ, two SC primitives are designed for the SP and CP separately. The input of them can be from any magnitude bit-plane. Two ZC primitives are employed for each magnitude bit-plane along with one RLC primitive except for the highest magnitude bit-plane. Since the highest magnitude bit-plane has only the CP, one ZC primitive along with one RLC primitive is enough. Finally, one MRC primitive is arranged for the MP, and its input can be from any magnitude bit-plane except the highest magnitude bit-plane.

5. RESULTS

We test the architecture on the FPGA platform of Altera Company. It requires about 4K Logic Cells in which the memory resource is contained. The memory is implemented with D Flip-Flops rather than the specific memory block. The four primitives ZC, SC, MRC and RLC need about 60 Logic Cells, while all primitives in the BPDP coding require about 900 Logic Cells. That is, the BPDP coding uses about 15 times primitives of the serial coding, but it obtains about 43 times speed-up.

The architecture of the BPDP coding is capable of encoding one coefficient per clock cycle. However the coding delays still can occur because the sign bit is encoded and inserted code stream of the magnitude bit after encoding the magnitude bit ν_M. If the following arithmetic coding is supposed to be able to encode one pairs of context and data per cycle, the delay will occur. The delay can be smoothed with the buffers or be removed completely with the large buffers. In each pass, just partial coefficient bits meet the condition and are encoded. So with the buffers, the delay can be compensated in the following time without encoding.

Table 2. Delayed cycles with 3 buffers

	8 pairs	12 pairs	16 pairs
Lena	318 cycles	36 cycles	5 cycles
Goldhill	167 cycles	12 cycles	0 cycles
Peppers	308 cycles	35 cycles	5 cycles

Table 2 lists the delay cycles when encoding test images, Lena, Goldhill and Peppers separately. The delay cycles all happen in the SP. The delay also may occur in the CP, but the probability is little. In Table 2, when using the buffers with length of 8 pairs of CX and D, there are hundreds of the delay cycles. However, when using the buffers with 16 pairs, the delay cycles are reduced to only a few.

6. CONCLUSION

The BPDP coding is very effective technique for hardware implementation of the fractional bit-plane coding in JPEG2000. It can accelerate the encoding by a factor of 15 over the pass parallel coding and the subband parallel coding, and by a factor of 43 over the serial coding. And it doesn't need to increase the memory and logic circuit in proportional to the speed-up factor like the subband parallel coding. On the contrary, it reduces the memory to 3K bits around. It requires about 15 times logic circuit more than the serial coding for primitives by proposed partial primitive-parallel technique. Additionally, all the passes and bit-planes are encoded within one scan, so that a lot of power is saved.

ACKNOWLEDGEMENT

The work was supported by China 863 (No. 2001AA114141) and 973 (No. G1998030606) plans.

REFERENCES

[1] JPEG 2000 Part 1 020719 (Final Publication Draft), ISO/IEC JTC1/SC29/WG1 N2678, July, 2002

[2] K. Andra, C. Chakrabarti, and T. Acharya, "A High-Performance JPEG2000 Architecture," IEEE Trans. Circuits Syst. Video Technol, vol. 13, pp. 209-218, March 2003

[3] C. J. Lian, K. F. Chen, H. H. Chen, and L. G. Chen, "Analysis and Architecture Design of Block-Coding Engine for EBCOT in JPEG2000," IEEE Trans. Circuits Syst. Video Technol., vol. 13, pp. 219-230, March 2003

[4] J. S. Chiang, Y. S. Lin, and C. Y. Hsieh, "Efficient Pass-Parallel Architecture for EBCOT in JPEG2000," IEEE ISCAS-2002, May 2002

[5] Y. Li, R. E. Aly, M. A. Bayoumi, and S. A. Mashali, "Parallel High-Speed Architecture for EBCOT in JPEG2000," IEEE ICASSP-2003, May 2003