

FIFO Controller Design Technique in JPEG 2000 Encoder Using Weibull Distribution

Sumit Srivastava, and Nameeta Sharma

Abstract—In this paper we have introduced the concept and design architecture for *FIFO Controller*, which is a module of JPEG2000 Encoder. JPEG2000 stands for Joint Photographic Experts Group 2000 which is an international standard for still image compression application in lossy or lossless environment. *FIFO Controller* is an inevitable interfacing unit in JPEG2000 architecture for its successful compression efficiency. This FIFO Controller takes the account of the average rate at which the compressed image data is outputted by the Embedded Block Coding with Optimized Truncation unit (a data compression unit) or the rate at which the FIFO is filling and the maximum number of bytes, which the particular EBCOT unit can output. The architecture has been implemented in Verilog using Actel Libero IDE. The results in terms of frequency of operation, slack time, gate count etc. obtained after synthesis are quite good. The distribution used can be the Exponential Distribution or the Weibull Distribution for the optimization. The Weibull is preferred instead of Exponential Distribution as the number of Parameter studied in Weibull is more as compared to Exponential.

Keywords—JPEG2000, EBCOT, FIFO Controller.

I. INTRODUCTION

FIFO Controller is the user defined module and having no resemblance with the available FIFO Controllers. The name FIFO Controller is derived from the data controlling mechanism for FIFOs. FIFO Controller is the important interface unit in JPEG2000[1]. JPEG2000 is a new compression standard for still images intended to overcome the shortcomings of the existing JPEG standard. The standardization process is coordinated by the Joint Technical Committee on Information technology of the International Organization for Standardization (ISO)/ International Electro technical Commission (IEC). JPEG2000 makes use of the wavelet and sub band technologies.

The JPEG2000 compression standard is composed of the stages shown in the flow graph in Fig.1. JPEG2000 is a wavelet-based image compression standard using the EBCOT (Embedded Block Coding with Optimized Truncation) image compression algorithm [2,3]. The JPEG2000 encoding process decomposes a still image into a hierarchical organization involving multiple resolution levels, sub bands, precincts, and code blocks. The encoder first translates the separate RGB

color components of the image into the corresponding YCbCr color space. Subsequently, the wavelet transform decomposes each component into several resolution levels, each containing a series of sub bands (3 sub bands at each level, except for the lowest resolution level, which has 4). The coefficients in each wavelet sub band are then quantized and divided into regular arrays of precincts and code blocks for entropy coding.

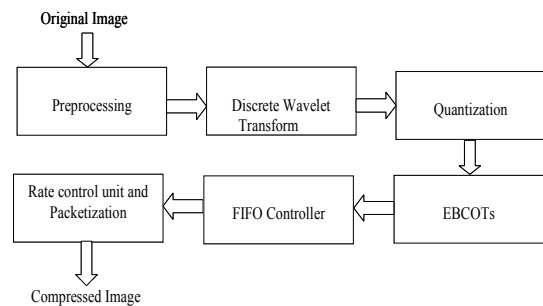


Fig. 1 JPEG 2000 flow

Each entropy-coded code block is composed of several coding passes (usually called embedded bit-streams) and each coding pass provides a variable quality contribution to the reconstructed image. Following entropy-coding, a post compression rate allocation algorithm selects coding passes from each entropy-coded code block levels, sub bands, precincts and code blocks so that an optimal allocation, minimizing image distortion according to the desired bit rate, is achieved.

II. JPEG2000 AND FIFO CONTROLLER

This shows the interfacing position of FIFO Controller in JPEG2000 encoder with its adjacent units. Mainly this unit lies in between EBCOT unit and Rate control unit.

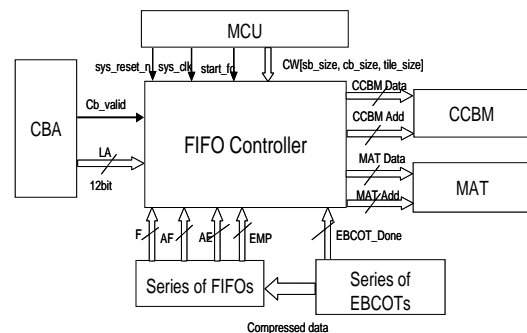


Fig. 2 FIFO Controller interfacing

Sumit Srivastava is with Department of Information Technology, Poonmima College of Engineering, Sitapura, Jaipur, Rajasthan, 302017, India.

Nameeta Sharma, is with Department of Electronics and Communicatin, Poonmima College of Engineering, Sitapura, Jaipur, Rajasthan, 302017, India.

The FIFO Controller block (Fig. 2) has interface to the code block allocator (CBA), FIFOs and EBCOTs input side, and compressed code block memory (CCBM) and memory allocation table (MAT) output memories at output side. F, AF, AE and EMP are the flags for indicating the current position of particular FIFO full, almost full, almost empty and empty respectively. Along with FIFO flags compression unit i.e. EBCOT also gives indication that it has finished its working. CBA gives logical address (LA) which contains resolution number, Sub band number and code block number for making the entry in a MAT of each code block. Here three main SRAM memories are used known as CCBM, MAT and FIFOs. The moment FIFO Controller gets start signal from master controller it displaces data from FIFOs and relocates in CCBM.

III. ARCHITECTURE OF FIFO CONTROLLER

The architecture of FIFO Controller is shown in the Fig. 3 given below. Complete architecture is comprises of four main blocks known as FIFO arbitration unit, CCBM address generator, MAT address generator and RAM memory for storing logical address from CBA.

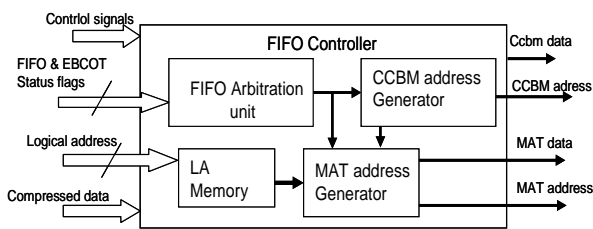


Fig. 3 Architecture of FIFO Controller

The inputs to the FIFO Controller are control signals from master controller unit (clock, reset, start); status flags from FIFO and EBCOT which develops the criteria for the arbitration, logical address from CBA and the compressed code block data from FIFOs.

A. FIFO Arbitration Unit

FIFO arbitration unit is responsible for arbitrating FIFO among six or more FIFOs. The arbitration is done on every arbitration clock cycle which is having double period of system clock. When system reset becomes inactive and FIFO controller gets start pulse then the arbitration would be done on the basis of FIFO flags as well as on EBCOT status flag which indicates whether EBCOT is working or not. Under reset state the priority of priority registers is zero and the output (i.e. arbitrated FIFO) is under high impedance state. There are mainly two prior conditions on which arbitration is done. Priority wise first it will check whether one or more than one EBCOT is working i.e. there EBCOT status flag is low. Now the flag position of only those FIFOs will be checked on priority basis from full, almost full and almost empty whosoever satisfy above condition. After prioritizing from above three it will again priorities from FIFO five to FIFO one. On the second priority it will check whether one or more than one EBCOT has finished its working but the FIFO

corresponding to that EBCOT is not empty, i.e. there EBCOT status flag is high and empty flag is low. Now the flag position of only those FIFOs will be checked on priority basis from almost empty, almost full and full whosoever satisfy above condition. In this case the order of priority is reverse as that in previous condition. After prioritizing from above three it will again priorities from FIFO five to FIFO one. The arbitration is done combinatorially and sequential circuit is for updating with current value. Its output i.e. arbitrated FIFO is input for CCBM address.

B. CCBM Address Generator

CCBM address generator is mainly responsible for generating address of the locations in CCBM where compressed code block data from the FIFO is written. The complete functionality of the CCBM address generator is governed by partitioning scheme of CCBM in block and pages, which is shown in Fig. 4 given below.

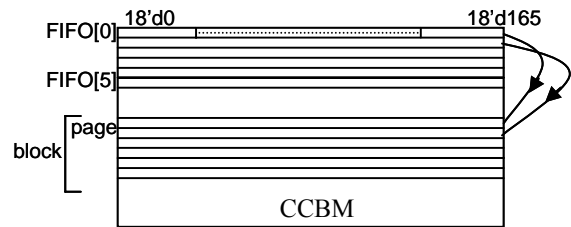


Fig. 4 CCBM partitioning scheme

If any page is full then it will take the jump on every sixth page. So every nth multiple of six page from a particular page are dedicated for the data from particular FIFO. When reset is active then the memory named as Add_mem and count_mem are having above data respectively. Count_mem counts the locations of CCBM and if it reaches to e.g. one hundred sixty fifth location then it will be again reset to zero. Every time the value of counter count_mem is added in the previous value of address for current address. After completion of one page the address will be started from the sixth page by keep on adding the counter value. The starting address and the ending address are taken from memory address and are registered. The memory address is designated as a starting or ending address depending on the current position of Empty flag of FIFO and EBCOT status flag. If the EBCOT status flag and Empty flags of a arbitrated FIFO are low then the memory address will be stored in a starting address register keeping the care of the fact that after getting one starting address of a particular code block other should not come even if the above condition is satisfied. If the almost empty, almost full, full and Empty flags of a arbitrated FIFO are low and EBCOT status flag is high then the memory address will be stored in a ending address register.

C. MAT Address Generator

MAT address generator is mainly responsible for generating address of the locations in MAT where identity information of each code block can be kept as a record. This identity information is mainly consists of starting address of

CCBM from where the first data of compressed code block is kept and the ending address where the last data of compressed code block is kept, along with the logical address given by the code block allocator. For the generation of addresses it first calculates the index number of currently acting code block in the respective sub band. The index number will act as a address in the MAT for storing identity information of each code block. The output of arbitration unit will be a input to this unit so that there should be one to one correspondence between code block data to be entered and its identity to be recorded in MAT.

D. The Exponential Distribution

Let us consider a situation that optimization follows exponential distribution with probability density function.

$$f(t) = \alpha \exp(-\alpha t) \text{ where } t \geq 0, \alpha \geq 0 \tag{1}$$

$$I_{t_1} = \int_0^{t_1} \mu.R(t_1 - t)dt \tag{2}$$

Then $R(t)$ becomes $\exp(-\alpha t)$. Substituting this result(1) into equation (2) and evaluating the integration, we get:

$$I_{t_1} = \frac{\mu}{\alpha} (1 - \exp(-\alpha t_1)) \text{ } 0 \leq t_1 \leq T_1 \tag{3}$$

In a similar way we can get:

$$(p - \mu)\Delta t_2 = -\mu.\exp(-\alpha(t_2 - t)) \tag{4}$$

and

$$I_{t_2} = \frac{\mu}{\lambda} \exp(-\alpha t_2) (\exp(\alpha t) - 1) \tag{5}$$

Solving equation (4) with the boundary condition of $t = T_1$ at $t_2 = T_1$ gives the following solution

$$\mu \exp(\alpha t) = p \exp(\alpha T_1) - (p - \mu) \exp(\alpha t_2). \tag{6}$$

With this solution, equation (5) is converted into:

$$I_{t_2} = \frac{1}{\alpha} [p \exp(\alpha(T_1 - t_2)) - (p - \mu) - \mu \exp(-\alpha t_2)] \tag{7}$$

E. Weibull Distribution

The Optimie equation for the Weibull Distribution

$$f(t) = \begin{cases} \alpha \beta t^{\beta-1} \exp(-\alpha t^\beta) & \text{where } t \geq 0, \alpha > 0, \beta > 0 \\ 0 & \text{otherwise,} \end{cases}$$

$R(t)$ becomes $\exp(-\alpha t^\beta)$ where α and β are some constants determined by the optimization process. Substituting $R(t)$ into equations (2), (3) and (4) yield

$$I_{t_1} = \int_0^{t_1} \mu \exp(-\alpha(t_1 - t)^\beta) dt, \tag{8}$$

$$I_{t_2} = \int_0^{t(t_2)} \mu.\exp\{-\alpha(t_2 - t)^\beta\} dt \tag{9}$$

And

$$(p - \mu)\Delta t_2 = \mu.R(t_2 - t).(-\Delta t) \tag{10}$$

Notice that it is essential to find $t(t_2)$ in evaluating I_{t_2} . However, solving equation (10) with respect to $t(t_2)$ is extremely difficult because finding $t(t_2)$ in a closed form with given t_2 seems impossible. Therefore, we examine two approaches in order to estimate $t(t_2)$. One is a numerical method. A computer program is developed where the Runge-Kutta method is adopted to solve the differential equation (10). The other is an approximation under the assumption that $\alpha \ll 1$. Ghare and Schrader(1973) elaborated the validity and physical meaning of the assumption.

Let $\mu = t_2 - t$ and $\lambda = p - \mu$. Then the equation (10) becomes

$$\lambda \left(\frac{du}{u} + 1 \right) = -\mu.\exp(-\alpha.u^\beta) \tag{11}$$

With the boundary condition that $t = T_1$ at $u = 0$, equation (11) is transformed to

$$\int_n^u \frac{-\lambda}{\lambda + \mu.\exp(-\alpha x^\beta)} dx = \int_r^t dy = t - T_1 \tag{12}$$

Using the Taylor series generated by exponential function ignoring terms with higher than third order powers, equation (12) becomes

$$\frac{\lambda}{\mu} \left[1 + \frac{\lambda}{\lambda + \mu} \frac{\alpha u^\beta}{\beta + 1} + \frac{\mu(\mu - \lambda)\alpha^2 u^{2\beta}}{2(\lambda + \mu)^2 (2\beta + 1)} + O(\alpha^3) \right] = t - T_1 \tag{13}$$

Letting $t = g_0 + g_1(t_2) \alpha + g_2(t_2) \alpha^2 + O(\alpha^3)$, then we get

$$u = t_2 - g_0 + g_1(t_2) \alpha + g_2(t_2) \alpha^2 + O(\alpha^3) \tag{14}$$

applying approximation formula, $(1 - x)^\beta \approx 1 - \beta x$ after substituting equation (14) into Equation (13), we obtain:

$$\frac{-\lambda\mu}{\lambda+\mu}(t_2 - g_0 - g_1\alpha - g_2\alpha^2).$$

$$\left[\begin{aligned} &1 + \frac{\lambda}{\lambda+\mu} \frac{\alpha}{\beta+1} (t_2 - g_0)^\beta \left(1 - \frac{\alpha\beta g_1}{t_2 - g_0}\right) \\ &+ \frac{\mu(\mu-\lambda)\alpha^2}{2(\lambda+\mu)^2(2\beta+1)} (t_2 - g_0)^{2\beta} + O(\alpha^3) \end{aligned} \right] \quad (15)$$

$$= g_0 + g_1\alpha + g_2\alpha^2 - T_1 + O(\alpha^3)$$

Using the above procedure with higher powers of α , it is possible to find an approximate value of t to any desired accuracy.

IV. FUNCTIONALITY OF FIFO CONTROLLER

The name FIFO Controller is derived from the data controlling mechanism for FIFOs. This FIFO Controller takes the account of the average rate at which the compressed image data is outputted by the EBCOT, or the rate at which the FIFO is filling and the maximum number of bytes, which the particular EBCOT unit can output. It regularly empties the FIFOs, and organizes the data into CCBM and correspondingly updates the addresses of CCBM in the MAT.

This work also includes the estimation of optimized size of FIFOs and to generate the best possible technique to empty the FIFOs. FIFO memory (SRAM) used are asynchronous in nature. The module FIFO Controller lies in between EBCOT unit and the Rate Control unit. There are number of EBCOT units in parallel used to achieve a high frame rate and corresponding to each EBCOT unit there will be one FIFO for storing the compressed code block byte-stream i.e. the result of EBCOT. All the FIFOs will be having some handshaking signals (Flags) which give indication to the FIFO Controller about the current position of FIFO depth i.e. whether it is full, almost full, almost empty or empty. FIFO Controller is interfaced with the Code Block Allocator, FIFOs, MAT and CCBM. It regularly empties the FIFOs, and organizes the data into CCBM and simultaneously updates the MAT. The complete process is performed on the basis of the information of code block data provided by Code Block Allocator. If the EBCOT corresponding to arbitrated FIFO has completed its compression and at the same time the empty flag of that FIFO is high then the address generated is the ending address and its vice versa process will generate starting address. The resulting starting address and the ending address will be further entered as data into the MAT along with the logical address for further identification of code block in CCBM. To facilitate other units the data in the MAT is entered by generating the address on the basis of information provided by Code Block Allocator and Master Controller.

V. RESULTS

FIFO Controller logic has been implemented in Verilog Hardware Description Language using 'Actel Libero IDE v 7.1' and verified using simulation tool 'ModelSim.' Verilog

codes are also synthesized using synthesis tool 'Synplicity', to generate gate level architecture along with the Exponential and the Weibull Distribution. Design is implemented on RTAX 1000S-S, target device from 'Actel Axcelerator'. The resource usage report with frequency of operation is as follows for each of the distribution are shown:

Resource Usage Report
Target Part: rtax1000s-s
Combinational Cells : 897 of 12096 (7.4%)
Sequential Cells : 425 of 6048 (7.02%)
Total Cells : 1188 of 18144 (6%)
Performance Summary
Frequency of operation : 92.5MHz

Fig. 5 The output with Exponential

Resource Usage Report
Target Part: rtax1000s-s
Combinational Cells : 803 of 12096 (6.67%)
Sequential Cells : 385 of 6048 (6.63%)
Total Cells : 1188 of 18144 (6%)
Performance Summary
Frequency of operation : 92.5MHz

Fig. 6 The output with Weibull

VI. CONCLUSION

After designing the FIFO Controller in JPEG 2000 encoder, it is concluded that for the efficient working of JPEG 2000 it is a inevitable interfacing unit. It ensures correct management of the compressed data which is the output of EBCOT. The functionality of successive units is also achieved due to its well organised working. It stamps out the problem of overflowing of FIFO. Its arbitration sub unit plays key role for its unbeatable performance. The results presented in this section show that, from a functionality point of view, it has achieved appreciable frequency of operation and having endurable slack in design. Further the output obtained using Weibull are more precise as compared to Exponential.

REFERENCES

- [1] D. S. Taubman and M. W. Marcellin, "JPEG2000: Fundamentals, Standards and Practice" Kluwer Academic Publishers, Boston, 2002.
- [2] M. D. Adams, "The JPEG-2000 Still Image Compression Standard," Tech. Rep. N2412, ISO/IEC JTC 1/SC 29/WG 1, September 2001.
- [3] M. Rabbani and D. Santa Cruz, "The JPEG2000 Still-Image Compression Standard", Course given at the 2001 International Conference in Image Processing (ICIP), October 2001.
- [4] S. Saha, "Image Compression - from DCT to Wavelets: A Review," ACM Cross-roads Student Magazine, vol. 6.3, Spring 2000.
- [5] R. Clark, "An Introduction to JPEG2000 and Watermarking", Elysium Ltd. <http://www.jpeg.org/public/jpgintro.pdf> (as of 01.10.02).
- [6] W. B. Pennebaker and J. L. Mitchell, "JPEG: Still Image Compression Standard", Kluwer Academic Publishers, 1993.
- [7] Ghare, P. M. and Schrader, G. F. 1973. A Model for Exponentially Decaying Inventories. Journal of Industrial Engineering, 5 (4).
- [8] Hwang, H. 1982 An EPQ Model for Deteriorating Items under LIFO Policy. Journal of the Operations Research Society of Japan, 25 (1).

- [9] Hwang, H. and Hwang, H.S. 1982. Optimal Issuing Policy in Production Lot Size System for Items with Weibull Deterioration. *International Journal of Production Research*, 20 (1).
- [10] Koh, S.G., Hwang, H., Sohn, K.I., and Ko, C.S. 2002. An Optimal Ordering and Recovery Policy for Reusable Items. *Computers & Industrial Engineering*, 43.
- [11] Nahmias, S. and Rivera, H. 1979. A Deterministic Model for a Repairable Item Inventory System with a Finite Repair Rate. *International Journal of Production Research*, 17(3).
- [12] Schrady, D.A. 1967. A Deterministic Inventory Model for Repairable Items. *Naval Research Logistics Quarterly*, 14.
- [13] Teunter, R.H. and Vlachos D. 2002. On the necessity of a disposal option for returned items that can be remanufactured. *International Journal of Production Economics*, 75.
- [14] Wu, K.S. 2001. An EOQ inventory model for items with Weibull distribution deterioration, ramp type demand rate and partial backlogging. *Production Planning and Control*, 12 (8).