Journal of Information & Communication Technology
Vol. 1, No. 1, (Spring 2007) 42-50

Large Data handling Technique for Compression Pre-coder using Scalable Algorithm

Muhammad Kamran[*]
Beijing Institute of Technology, Beijing, China
University of Engineering and Technology, Lahore, Pakistan
Shi Feng[*]
Beijing Institute of Technology, Beijing, China
Ji Weixing[*]
Beijing Institute of Technology, Beijing, China

ABSTRACT

This paper presents the procedure of handling and testing digital design module used for the image data separation using averaging scalable method. The design will incorporate all constraints and after satisfactory mathematical interpretation, source code of algorithm is developed in VHDL. Afterwards, pre and post simulations will be run using MODEL SIM to verify proper design operation. The proposed work is the part of scalable image compression pre coder used for the compression of image. This paper explains the effective solution to deal with large input data which can not be handled properly by conventional design procedure. In this research area or DSP applications, data information is very large in number and for hardware verification required to be transmitted at reasonably high speed without any considerable loss. The problem of large data input vectors is encountered in different applications and needs special attention for rectification while designing the chip. For this purpose, simulated large data is required to be generated representing the actual input video data. Actual video data may be of any image standards like JPEG, MPEG or H.26X. This paper presents the architecture of pre coder chip and the results of the work carried out till RAM module interfaced between buffer control and pre coder modules with all limitations and their corresponding solutions.

Inspec Classification: C6130M

Keywords : Layered scheme, large data; data file call; vhdl code; ROM data file

[*] The material presented by the authors does not necessarily portray the viewpoint of the editors and the management of the Institute of Business and Technology (BIZTEK).

[*] Muhammad Kamran: kamarn.uet@gmail.com
[*] Shi Feng: shifengyoujian@tom.com
[*] Ji Weixing: pass@bit.edu.cn

© JICT is published by the Institute of Business and Technology (BIZTEK).
Main Ibrahim Hydri Road, Korangi Creek, Karachi-75190, Pakistan.

1) INTRODUCTION

Simulation is the first step for any kind of chip design prior to its synthesis and realization. Many simulation and synthesis tools are available for this purpose like Active HDL, VERILOG for design and MODEL_SIM, SYNPLIFY etc., for simulation and synthesis. For the designing of chip, efficient algorithms play vital role for the design confirmation. Algorithmic design is pre and post simulation and information from different tools verify the optimization of design with respect to operating time and area occupied. This means chip will perform improved operation and proved optimized if some innovative scheme is introduced in algorithm as compare to prevailing techniques. After realizing the importance of algorithm, behavioral or structural model of chip is developed. Professional engineers use the structural modeling as this is easy to simulate and synthesize. But many engineers and scientists use behavioral model because source code used may be half in length as compare to structural model code. For our research work of ASIC design methodologies, we are using Scalable Image Compression Algorithm proposed by Wang and Shi Wang Yi-zhuo and Shi Feng (2003). This algorithm has few limitations which will be rectified while completing the design of our ASIC. Adaptability and error bearing tolerance for network bandwidth is effectively controlled using scalable compression algorithm and multicast transmitting mode Mccanne S (1996) and Li Xue (1998).

Scalable compression divides the original image into multilayer stream blocks belonging to their respective groups. Receivers are specified with respect to their own level of subscription by joining a subset of the groups. They obtain video of different quality stream number. Due to the network characteristics of heterogeneity, encoder parameters are set up or one can utilize different coders to compress the image. The operating mode is called simulcast in which each stream is relatively independent to each other, and can decode independently to get the video signal of different quality, so it doesn’t utilize statistical relativity among streams, the amount of the redundant information is too large, and the compression efficiency will be low, causing enormous bandwidth waste in transmitting terminal Wang Yi-zhuo and Shi Feng (2003). In section 2 of the paper, there is introduction of DCT and our layered compression algorithm based on DCT is described showing data flow in different steps of our proposed ASIC. Section 3 depicts the architecture of the chip with the explanation of how to append our large input data file developed using any high level language like C++ or Java. Section 4 is comprised of all test results showing substantiation of our concept to transmit data efficiently for chip simulation in VHDL and in MODEL SIM.

2) LAYERING PROCEDURE

Prior to make idea of layering procedure, it is important to clarify few aspects regarding compressing video or still data.

2.1) COMPRESSION OF VIDEO DATA

Compression of image data is the first step in image processing area. There are different standards of video coding like JPEG, JPEG2000, MPEG series etc. There are two major categories defined for video and still image compression;

i) Discrete Cosine Transform
ii) Wavelet Transform

Authors in Siva Somasundaram and K.P. Subba Lakhshmi (2003) gave the picture of scalable algorithm using wavelet transform. In the area of image compression 2DWT and 3DWT is utilized now for best video performance on the receiver end. Wang and Shi have proposed an algorithm which divides the video image in multi-layer system by using DCT. DCT algorithm is very effective due to its symmetry and simplicity. It is good replacement of FFT due to consideration of real component of the image data. In DCT we leave the unwanted frequency components while opting only required frequency components of the image. Image is divided into blocks and each block is compressed using quantization. Moreover, many simulation tools like MATLAB are available to estimate the results prior to realization of design in real time. Equation (1) and (2) gives the one dimension DCT standard equations for the data out put and coefficient calculation respectively during compression, Equation (2) is used for the calculation of DCT coefficients which are fixed for different block of input image Zheng Baoyu (1992).

$$y(k) = \sqrt{\frac{2}{N}}c(k)\sum_{n=0}^{N-1} x(n)\cos\frac{(2n+1)k\pi}{2N} \qquad (1)$$

$$c = \sqrt{\frac{2}{N}}\left[c(k)\cos\frac{(2n+1)k\pi}{2N}\right]_{N\times N} \qquad (2)$$

2.2) PROPOSED ARCHITECTURE AND PROCEDURAL STEPS

DCT algorithm is utilized for the behavioral architecture design verification. The full image is divided into blocks or layers carrying pixels. Pixels of different layers travel from input image source to the output coders through number of processes and steps. The pixels are always kept in their own block or layer by introducing most important signal like “Current layer” as in our design. First process is the compression of data using encoding process. This is achieved using “Down sample” module in our ASIC. Couple of RAM circuits are appended in the design to calculate data writing address in RAM1 and carry rebuild data of B1 and B2 in RAM2 as shown in Fig. 5.

The explanation of our Algorithm is as follows;

First step is the division of block of data into 4 layers, Base layer B1 and the extended layers E1, E2 and E3. From first extended layer E1 or any other extended layer E2 or E3, another base layer is extracted named as B2. B2 generation causes the number of layers equal to five. Size of B2 with respect to pixels is ¼ of all other image layers in the system. In up sampling model this B2 re-gain its size equals to any other layer in the system.

After rebuilding of pixel data, extended layers E1, E2 and E3 are changed to E1’, E2’ and E3’ in second last pre coder module and stored in output buffer-1 and B1 and B2 after rebuild process is stored in buffer-2, from where signals are transferred to coders. B1 after rebuild will remain same, but 5th layer B2 which is extracted from E1 will be changed in size. The process of missing pixels is explained in equation (3), while Fig. 1, Fig. 2, Fig. 3 and Fig. 4 are the matrices containing the pixels explain the steps of data from selected input image block of size 8X8.

Pix(X) = (Pix (i) + Pix (j))/2 (3)
Where i and j = 1, 2, 3, 4……n,
Pix (n) = Pix (n-1) for Row and for Column as well.

Pix(X) = Pixel created by formula and nth pixel is same as that of n-1st pixel on row as well as on column of our pixel data matrix. For our understanding, we selected an 8X8

pixel Block from input image **“PEPPER.PNG”.** We use our Scalable Algorithm for the calculation of pixel layers as shown in Fig-1 with pixel number in their corresponding block.

Pixels belonging to E1 are separated as shown in Fig. 1 and B2 extraction from E1 is obtained using same idea of leaving alternate row and column.

Lastly, before getting the complete data for the coders connected on the out put of our pre-coder ASIC, up sampling takes place and 4 pixel B2 will be converted into 16 pixel B2 by using equation (3) to make the size of B2 same as that of any block of pixel in the design. The up sampled B1 and B2 used to calculate new extended layers E1', E2' and E3' in pre-coder module (Not in the scope of paper). The calculations for the rebuilding of pixels process is represented in Fig. 2.

The matrix of Fig. 3 describes X1, X2, X3 & X4 are the generated pixels using average formula, while last row and the column of the matrix is just the repetition of 2nd last row and column.

Fig 1
Total Frame of 64 pixels with E1 in Bold Letters

154 E1	123 B1	123 E1	123 B1	123 E1	123 B1	123 E1	136 B1
192 E2	180 E3	136 E2	154 E3	154 E2	154 E3	136 E2	110 E3
254 E1	198 B1	154 E1	154 B1	180 E1	154 B1	123 E1	123 B1
239 E2	136 E3	180 E2	180 E3	166 E2	29 E3	123 E2	123 E3
180 E1	154 B1	136 E1	167 B1	166 E1	149 B1	136 E1	136 B1
128 E2	136 E3	123 E2	136 E3	154 E2	180 E3	198 E2	154 E3
123 E1	105 B1	110 E1	149 B1	136 E1	136 B1	180 E1	166 B1
110 E2	136 E3	123 E2	123 E3	123 E2	136 E3	154 E2	136 E3

Fig 2
B2 in bold letters-extraction from E1

154	123	**123**	123
254	154	180	123
180	136	**166**	136
123	110	136	180

$\longrightarrow$

154	123
180	166

Fig. 3
Rebuilding of B2

154	X1=138	123	123
X2=167	X3=155	X4=144	144
180	X3=173	166	166
180	173	166	166

Fig. 4 provides final B2 layer with 16 pixels after rebuilding process used to calculate new enhanced layers in pre coder module of our ASIC.

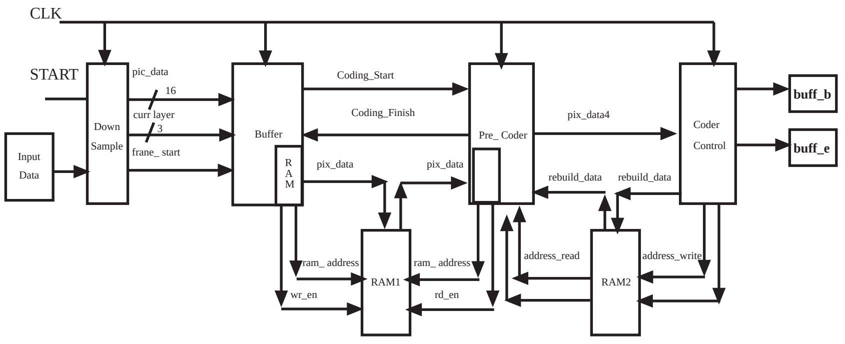
Fig 4
Complete 16 pixels B2 layer after rebuild

154	138	123	123
167	155	144	144
180	173	166	166
180	173	166	166

This averaging formula is not only applied to get the pixels in the rows, but also for the columns. Moreover, last row and last column is same as that of second last row and column. For the design of pre-coder ASIC for our research, each layer is comprised of 64 X 128 pixels of 16 bits each. This means there are 64 rows and 128 columns of the pixels for five layers in which image data is to be divided. The scope of our research is to prove the proposed algorithm by selecting the best architecture for the chip operation.

Input data is large enough not to be simulated by manual selection of data pixels. To overcome the problem of data generation, we developed a ROM which keeps data into itself at different addresses. ROM program can be developed in any high level language like C++, JAVA etc. This data from ROM is linked with VHDL source code by taking privilege of IEEE_1164 standard library.

Fig 5
Image Compression Pre-Coder, Behavioral Architecture



2.3) DATA GENERATION AND TRANSFER TECHNIQUE

The proposed architecture for pre coder chip of the research work is shown in Fig.5. As described earlier, RAM1 arranges the pixel layers and places the data into their corresponding blocks while second RAM reads the rebuild data of B1 and B2 from the last module coder control and transmits it to pre coder module for calculating new enhanced layers E1',E2',E3'. Block diagram for behavioral design is developed followed by VHDL code for initializing the data to verify the operation of each module. The chip will be ready for real time operation after interconnection of all modules, simulation, synthesis and post simulation process. Successful post simulation ensures the ASIC download on FPGA (Field Programmable Gate Array).

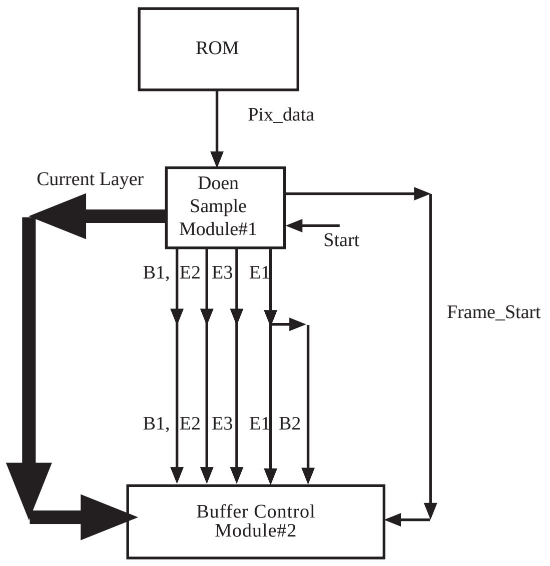
2.3.1) SOURCE CODE INCLUSION IN VHDL

Block diagram of Fig. 6 is extracted from our main pre coder ASIC in Fig-5, showing flow of data generated by ROM automatically in a random way, which is further added in VHDL code to call for random pixel data for the simulation of ASIC input data coming from video device. Data is read from ROM using “READ” command executed i n a loop. Most important signals current layer, start and frame_start are also prominent in Fig. 6, which divides data stream into their corresponding blocks. As discussed earlier, for the verification and realization of ASIC, first step is the simulation of chip Fig. 5 shows the behavioral model of our complete pre coder ASIC.

The RAM used in the design is of quite large size to store pixel data and is not able to be synthesized by XILINX ISE easily Siva Somasundaram et al. (2003). For the sake testing and verification of architecture we will utilize comparatively small pixel data or XILINX IP core will be used for handling such a big RAM.

The process of simulation using ROM of large data is successfully and efficiently carried out by using VHDL code and IEEE_1164 standard library, but synthesis of such designs can not be done using XILINX ISE, Mentor Graphics or SYNPLIFY synthesis tools.

Fig 6
Input vector generation ROM and Down sample module process



The computer program in C++ is developed to generate ROM carrying data input vectors representing the pixel data of real time video. This program file is stored as *.dat file and this data is later on fetched by VHDL program for the down sampling. VHDL code will be described for the problem as given below;

```
Library IEEE;
Use IEEE.std_logic_1164.all;
Use STD.TEXTIO.all;
Entity ROM is
Port(start,clk:in std_logic;
Rom_adr_i:in std_logic_vector(6 downto 0);
Pix_data:out std_logic_vector(15 downto 0); Current_layer:in std_logic_vector(2 downto 0));
End ROM;
--Architecture will contain
Image_loop:=0;
while ((not endfile(External file)) and (Image_loop<127)) loop
readline (External file, v_line);
read(v_line,v_rom_data(v_loop));
inc image_loop
end loop;
```

2.3.2) SOURCE CODE EXPLANATION

Clock is defined with time period of 20 ns. Program will keep on fetching the data till while loop is terminated. We used 128 pixel data for testing and even low for the representation of simulation results. But it may have any value depending upon the system or image loop defined in program. C++ code has to carry a pointer for pointing out the data defined in a function; two “FOR” loops are utilized. First works for given number of pixels defined, (128 in our case) and second “FOR” loop works for the 16 bits for each pixel with the condition of random data pick up represented in binary form. For our ASIC simulation, 128 pixels video data of 16 bits each is generated from the source. By appending library function USE TEXTIO.ALL in our VHDL code, we can read data from some externally generated file developed in any higher level language. From our code, input vector file “ROM.dat” is added into the project or SRC directory (Holding VHDL Source Code and Test Bench) and acts as a ROM storing data with maximum 128 pixels. This technique is very helpful in generating the data input vectors for testing of any other type of ASIC for simulation. Data from Read Only Memory is read from specified addresses automatically generated and acquired by VHDL source code with out manual insertion in test bench. It is for sure, this data is used for testing and simulation of the chip only and not for synthesis.

This problem will be encountered with such file during realization of ASIC on experimental FPGA board of ALTERA or XILINX. For the implementation of design, these FPGA boards are used to extract data from VHDL or VERILOG code after successful synthesis.

1) SIMULATION RESULTS

Our VHDL code reads the data stored in the file in random order according to the automatically generated address. The test results of our work corresponds to the verification of the idea innovated for the testing of ASIC. Fig-7 gives the successful simulation wave results using MODEL_SIM simulator. Signals apparent from Fig. 7 are Rom_adr_i, Start, Clock, frame _start, Pixel data, and Current layer. Number of pixels generated depends on number of rom_adr-I signal. Frame_start and start should be equal to ‘1’ prior to data transfer process.

Fig 7
Result of Data Transmission in MODEL_SIM

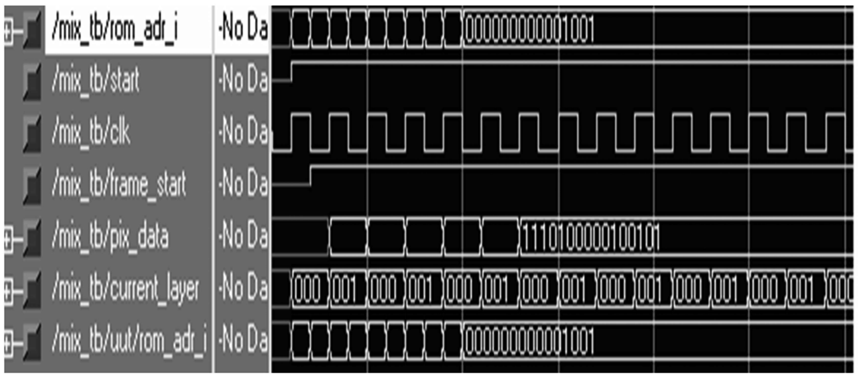


Fig. 8 gives the simulation result of the code developed in VHDL to prove our work. We have to apply rom_adr_i (Input address) in test bench causing its corresponding data to be available at the input of first module "Down Sample" of chip. This input data matrix will be further divided into 5 layers. Fig-7 and 8 verifies the results of operations on pixel data layer division as indicated in Fig-6. Table-1 is displaying the experimental results in list form till 210 ns time with current layer representation in Hexadecimal representation. Current layer is assigned decimal "0" for B1,"1" for E2, "2" for E3,"3" for E1 and "4" for B2 pixel data for distinction between different layers. If the time is increased for simulation beyond 210 ns, we can be able to see more information in timing wave form as well as in VHDL listing table. ROM_Address is the input address of ROM containing input data simulating the actual input video data of image pixels. The bold arrows in table-1 on 30-40 ns time and 50-60 ns gives clear picture that address of ROM does not change as well as input pixel data remains same. The input pixel value change takes place when we get positive edge of clock i.e., 0 to 1 after every 20 ns time.(Fixed Frequency in test Bench) For every odd value time in "ns" we get positive edge of clock and hence the data.

Fig 8
VHDL Response with ROM Address and Corresponding signal

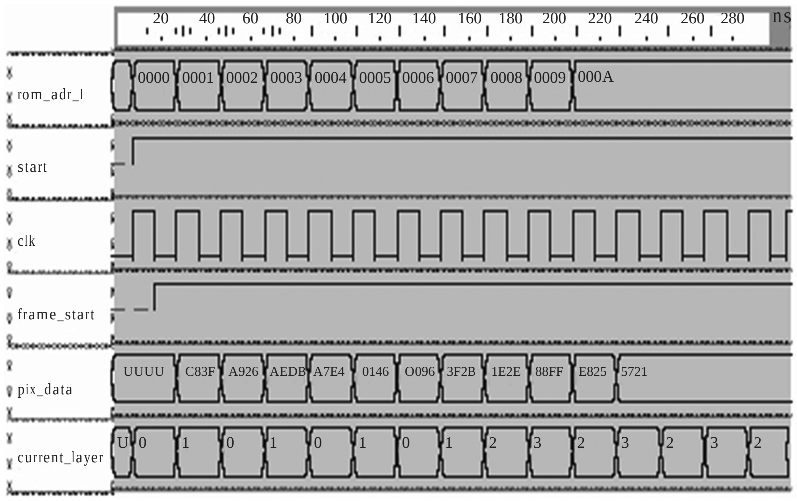


Table 1
Listing of Module1 with Automatic generated data

Time(ns)	Delta	ROM_Address	Start	Input Data	Current_Layer
0.0	1	uuuu	uu	uuuu	UUU
010	0	0000	1	uuuu	UUU
020	1	0001	1	uuuu	000
030	2	0001	1	c83f	000
040	1	0001	1	c83f	001
050	1	0002	1	aefd	000
060	0	0002	1	aefd	001
070	2	0003	1	12cb	000
090	2	0004	1	7dfa	001
110	2	0005	1	3344	010
130	2	0006	1	ccds	011
150	2	0007	1	54bc	010
170	2	0008	1	112f	011
190	2	0009	1	23df	100
210	2	000A	1	9a6b	100

4) CONCLUSION AND FUTURE WORK

This paper presents the starting portion of our research project indicating the problems of ASIC design verification facing the problem of large data transformation. In future complete pre coder design will be appended to the coders and a complete image compression system or scheme will be realized after mathematical verification of algorithm. The intention is to use XILINX Spartan 2E FPGA with 30000 gates for implementation of ASIC Design. Moreover, this design will be tested in real time using TEKTRONIX Logic analyzer and digital scope.

5) ACKNOWLEDGEMENT

This work is the part of the project assigned during PhD work in Embedded System Laboratory, Department of Computer Science and Engineering, Beijing Institute of Technology, Beijing, China. This design is planned to be commercialized afterwards, which will provide good contribution in the area of image compression with respect to stability and band width variations. The PhD program is totally funded by Higher Education Commission, Government of Pakistan.

REFERENCES

WANG YI-ZHUO AND SHI FENG, "Scalable Compression Algorithm for Video Monitoring System" Journal of Beijing Institute of Technology, 2003, Vol. 12 suppl.

MCCANNE S, "Scalable compression and transmission of internet multicast video" PhD dissertation, Berkeley: Computer Science Division (EECS), University of California Berkeley, 1996.

LI XUE," Scalable and adaptive video multicast over the internet" Atlanta: PhD Thesis, Georgia Institute of Technology, 1998.

Mysupport.XILINX.com

SIVA SOMASUNDARAM AND K.P. SUBBA LAKHSHMI, "A Novel 3D scalable algorithm" Proceedings of SPIE, Volume 5022, May 2003, pp 966-972

ZHENG BAOYU,"A New Algorithm for the 2-D Discrete Cosine Transform IEEE Transactions on Image processing, Vol. 40, pp 2166-2173, September 1992