International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 7– No. 1, April 2014 – www.ijais.org

FPGA based Hardware Implementation of Variant DPCM Image Compression Technique (Multiple LUT-DPCM)

Vandana Chahar
Department of Electronics Science
Kurukshetra University, Kurukshetra, India

Seema Narwal
Department of Electronics Science
Kurukshetra University, Kurukshetra, India

ABSTRACT:

This paper proposed FPGA implementation of variant DPCM based image compression technique (Multiple LUT-DPCM). Image compression addresses the problem of reducing the amount of data required to represent a digital image. Such data compression can be achieved by reducing the redundancies of the image data in order to be able to store or transmit data in an efficient form. There are various onboard data compression techniques used in image processing. Out of which DPCM based compression technique is the most commonly used one. For the hardware implementation, an algorithm for Multiple LUT DPCM is designed. VHDL code is developed based on the algorithm designed. After simulation and verification of developed code with test data generated, it implemented in to hardware. The developed hardware's functionality, then verified with the already simulated data and also with real image data.

Keywords
Image Compression, Redundancies, LUT, Differential Pulse Code Modulation (DPCM), FPGA

1. INTRODUCTION

Information, in its many forms, is a valuable commodity in today's society, and the amount of information is increasing at a phenomenal rate. As a result, the ability to store, access, and transmit information in an efficient manner has become crucial. This is particularly true in the case of digital images. A large number of bits are typically required to represent even a single digital image, and with the rapid advances in sensor technology and digital electronics, this number grows larger with each new generation of products. In order to utilize digital images effectively, specific techniques are needed to reduce the number of bits required for their representation. The branch of digital image processing [1] that deals with this problem is called image compression. A wide range of techniques has been developed over the years, and novel approaches continue to emerge. The goal of this project is to understand different available image compression techniques and to implement a different variant of DPCM technique[2] for onboard data compression.
The need for image compression becomes apparent when one computes the number of bits per image resulting from typical sampling rates and quantization schemes.

2. IMAGE COMPRESSION

The process of image compression can be explained via the block diagram given in Figure 1. In general, image compression can be separated into two processes: the encoder and the decoder. The encoder takes an input image and compresses it, while the decoder un compresses a compressed [3] image. The goal of the encoder is to eliminate any coding, spatial, or visual redundancies that might be present within the original gray level image. The encoder can be separated into three functional processes, as shown in Figure 1. The purpose of the mapping function is to transform the original image into a new domain that provides an easy method of eliminating spatial redundancies that may be present within the image. The quantizer process reduces the number of values that are generated from the mapping function by digitizing these values to a smaller subset. It is in this stage that the visual redundancies of the original image are reduced. The final stage, data coding, reduces data redundancies by assigning variable length codes to the output values from the quantizer. Of these three stages, both the mapping function and the data encoder are completely reversible. Hence, these two stages do not remove any information from the image. The quantizer stage, on the other hand, does remove information from the image, resulting in the uncompressed image being a degraded version of the original gray level image. Error-free compression requires that the quantizer be removed from the encoder and that only the mapping function and the data encoder be used.

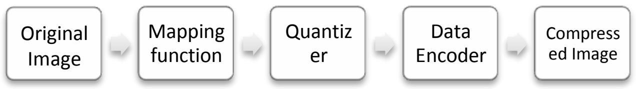


Figure 1: Image Compression (Encoder)

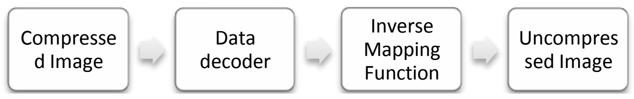


Figure 2: Image Compression (Decoder)

The decoder stage shown in Figure 2 takes a compressed image and uncompressed it to produce a new gray level image that represents the original image. For error-free compression, this output image is an exact replica of the original image. For an image compressed using lossy compression, this output image is a degraded version of the original image. The purpose of the data decoder is to perform the reverse operation of the data encoder, producing values that are equal to the output of the quantizer or to the output of the mapping function, depending on whether error-free or lossy compression was used. The final stage of the decoder applies the inverse to the mapping function given in the encoder to obtain the uncompressed image. There are many approaches to image compression [4], but they can be categorized into two fundamental groups: lossless and lossy image compression.

3. DIFFERENTIAL PULSE CODE MODULATION

Differential pulse code modulation (DPCM) is nothing but a predictive coding method which can be lossy or lossless. This paper aims the onboard implementation of a variant of lossy DPCM technique [2] [5].

As we know predictive coding scheme, the correlation between the neighboring pixel values is used to form a prediction for each pixel. By far, the most common approach to predictive coding is differential pulse code modulation (DPCM). In DPCM, the prediction is subtracted from the actual pixel value to form a differential image that is much less correlated than the original image data. The differential image is then quantized and encoded [6]. The quantization process determines the resulting bit rate and image quality.

4. DPCM IMPLEMENTATION

The process flow of the project is shown in the Figure 4. The entire project can be divided into software part and hardware part. This flow chart explains the software part of the implementation. As shown in the Figure 4 the software part contains project design, VHDL code development, and its simulation and synthesis and hardware part contains hardware implementation and verification.

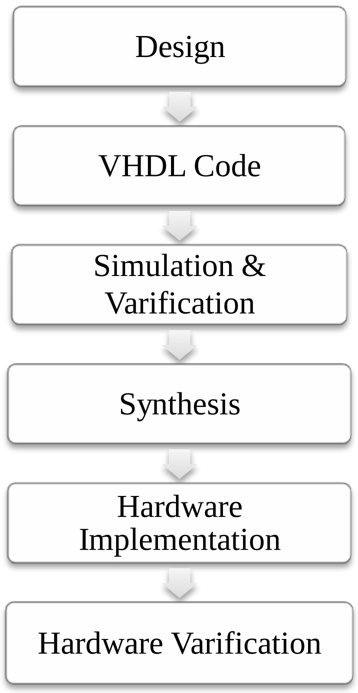


Figure 4: Project Flow Diagram

4.1 Multiple LUT DPCM Programming

The programming of multiple LUT DPCM was quite complex than single LUT DPCM because of the extra stages required for different operations. The timing diagram of Multiple LUT DPCM is shown in Figure 5.

Unlike single LUT DPCM here it needs seven cycles to start the transmission of first pixel. Eight stage processes required to four pixels group to complete the compression process. An eight state master counter designed for the entire process. The input data latching happens at the falling edges of each clock. The four pixels assigned to four internal signals and it can hold the values up to eight clock cycle. Selecting reference pixels is different in this case from normal DPCM method. For avoiding sign bit overhead here the lowest value pixel is considered as the reference pixel. The reference pixel had to find out after the latching of fourth pixel of the group, and that delay is visible from the timing diagram. The minimum value pixel and its position are stored in different variables for next eight clock cycles. The subtraction operations of the other three pixels from reference pixel can be done at one clock cycle after finding out the reference pixel and its position. At next clock edge this input difference values arranged in a single signal for simplicity of LUT mapping and it is shown in Figure 5 as Q1, Q2 and Q3. The next process is the LUT mapping and before that LUT selection logic should be programmed. For that the following logic can be used at next clock edge.

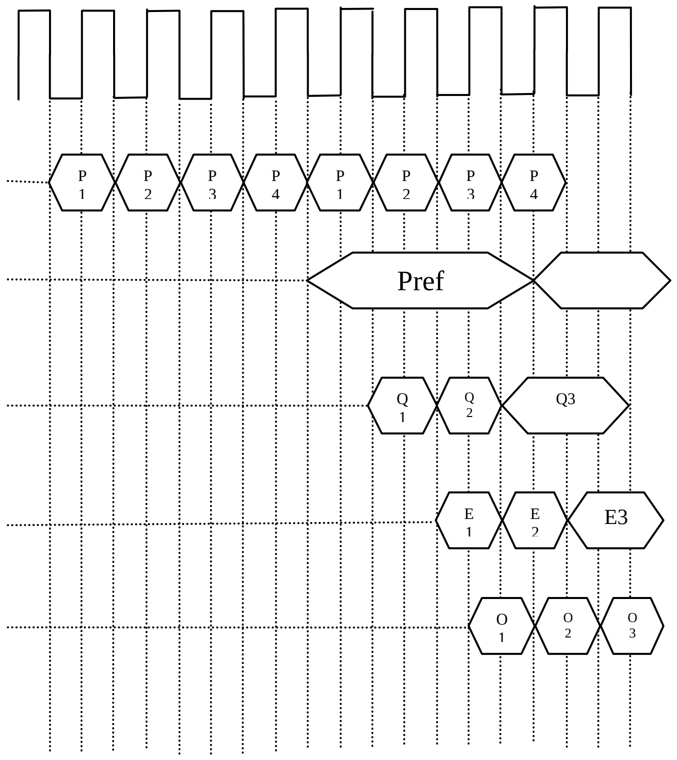


Figure 5: Timing diagram of Multiple LUT DPCM

```
If (maximum difference count > LUT threshold)
        Apply LUT-coarse and s=1
Else
Apply LUT-fine and s=0
End.
```

The LUT's are stored in a package file for better user interface. Repacking is the final process in which the three LUT mapped data's of 5 bits, reference pixel of 10 bit, the reference pixel position of 2 bit, LUT selection bit(S) of 1 bit and thus a total of 28 bit are packed to a group of four pixels of 7 bit.

4.2 Simulation And Verification

Test benches are developed for simulating the developed VHDL code based on the test cases generated.

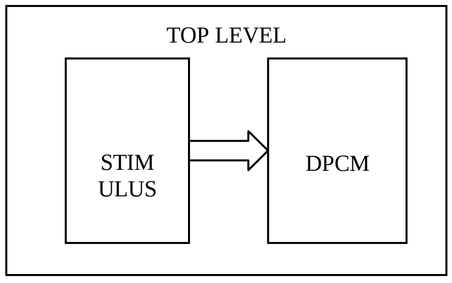


Figure 6: Top Level Module

There are different approaches for feeding test cases to the main DPCM block. First method is that design a top level module and to instantiate both DPCM block and stimulus block insides it. The stimulus block, which generates test data and feed to DPCM block. Its block diagram representation given in Figure 6.

4.3 Synthesis and Power Usages of Multiple LUT-DPCM

In Synthesis process a behavioral level design description is translating to a structural level description. By synthesis the VHDL code is compiled and mapped into the hardware of FPGA. Simplify Pro from synopsis is used for synthesis and it will provide the information of net list, port list, clock tree and total hardware usage from the FPGA board. RTL view of top multiple LUT DPCM is shown in Figure 7 below.

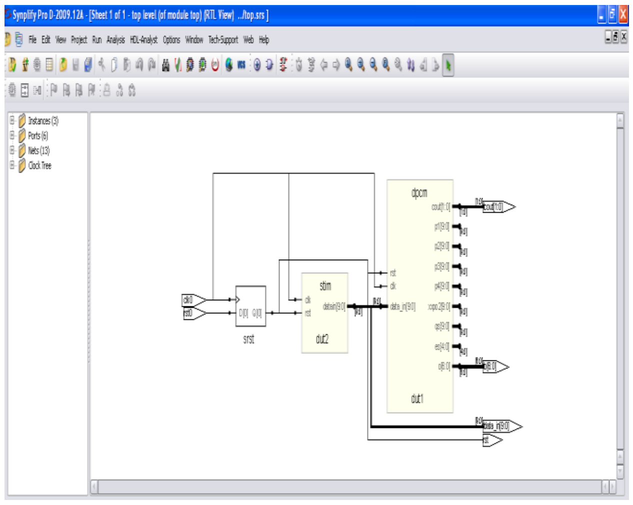
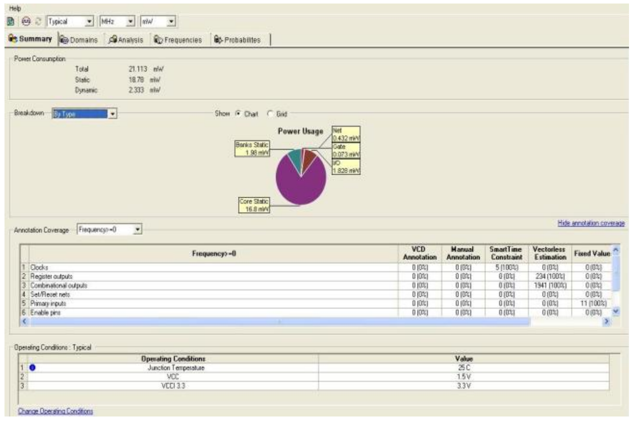

Synplify Pro D-2009.12A - [Sheet 1 of 1 - top level (of module top) (RTL View) ../top.srs]
File Edit View Project Run Analysis HDL-Analyst Options Window Tech-Support Web Help
Instances (3)
Ports (6)
Nets (13)
Clock Tree
dpcm
stim
srst
dut2
dut1
clk
rst

Figure 7: RTL view of top Multiple LUT DPCM

The total power usage of entire design can be obtained by Smart Power Tool and the estimated result is shown Figure 8. The total power usage in typical case is 21.1mW for multiple LUT DPCM out of which static power dissipation is about 18.78mW and dynamic power is about 2.33mW.



Typical | MHz | mW

Power Consumption

Total	21.113 mW
Static	18.78 mW
Dynamic	2.333 mW

Breakdown: By Type — Show: [✓ Chart] Grid

Annotation Coverage: Frequency>=0

	Frequency>=0	VCD Annotation	Manual Annotation	SmartTime Constraint	Vectorless Estimation	Fixed Value
1	Clocks	0 (0%)	0 (0%)	5 (100%)	0 (0%)	0 (0%)
2	Register outputs	0 (0%)	0 (0%)	0 (0%)	234 (100%)	0 (0%)
3	Combinational outputs	0 (0%)	0 (0%)	0 (0%)	1941 (100%)	0 (0%)
4	Set/Reset nets	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
5	Primary inputs	0 (0%)	0 (0%)	0 (0%)	0 (0%)	11 (100%)
6	Enable pins	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)

Operating Conditions : Typical

	Operating Conditions	Value
1	Junction Temperatue	25 C
2	VCC	1.5 V
3	VCCI 3.3	3.3 V

Figure 8: Power usage of Multiple LUT DPCM

5. HARDWARE IMPLEMENTATION AND RESULTS

After generating the post layout file and PDB file we can be fuse the program to FPGA through Flash Pro. The input and output data's designed in the hardware are parallel in nature. There are twelve input pins for ten bit image pixel data in addition to clock and reset signals and seven output pins for seven bit compressed data. The onboard clock provided in FPGA is of 40 MHz, but by considering practical applications the data is expected to arrive at 2 MHz speed. There are two methods to set the input clock frequency as 2 MHz. First method is to use a frequency divider circuit in front of the DPCM block. Second method is to use external clock, since the board having the option to use external clock. It should be ensured that the clock must be synchronous to other inputs, when we are using external clock. In this project we preferred to use external clock and reset signals to avoid the extra hardware design of frequency divider circuit. Figure 9 shows the developed hardware for Multi LUT DPCM.

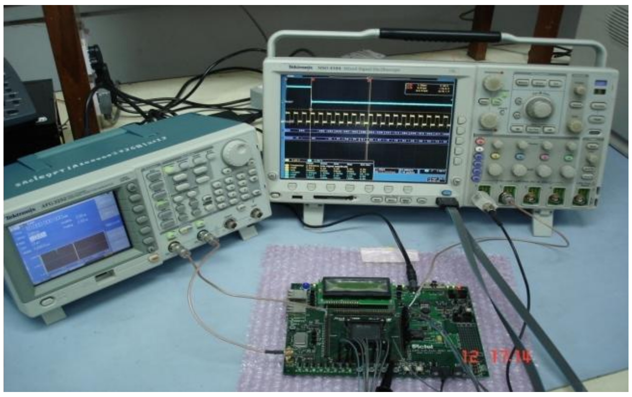


Figure 9: DPCM Hardware

5.1 Hardware Verification

The hardware verification is done by test data .This method is verifying the outputs by feeding hardware with the test data already generated and simulated.

5.1.1 Verification By Test Data

As we already knows the generated test data and its corresponding compressed values we can be test the hardware by feeding the same data and obtain the output in an oscilloscope and compare it with simulated results. The experimental setup for this verification is shown in Figure 10.

.

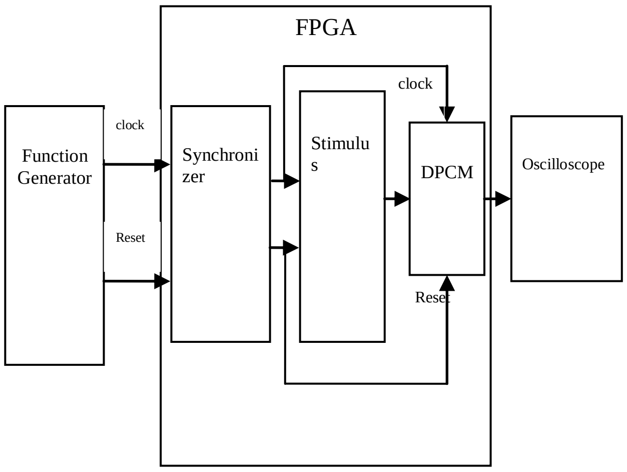
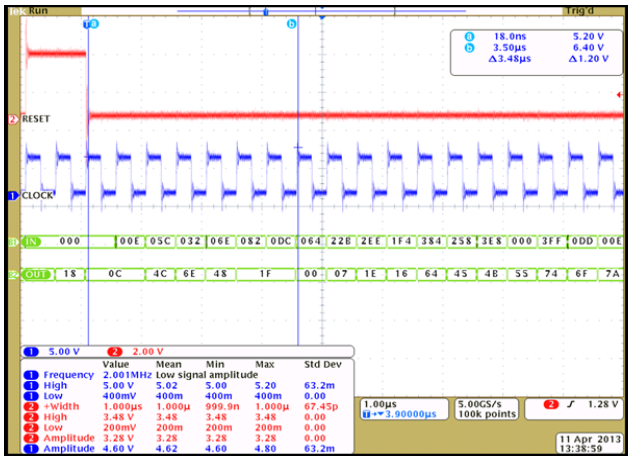


Figure 10: Block Diagram of Hardware Verification by Test data

The clock and reset signal are generated externally by a function generator and fed to FPGA. A synchronizer is required to synchronize both signals. The clock signal used is of 2 MHz and reset signal width is of 1μs. Both signals are connected to stimulus block and DPCM block. The stimulus block generate test data at each clock cycle and fed to DPCM block. The 7 bit parallel output obtained from FPGA connected to oscilloscope through digital channels. The obtained result in oscilloscope is shown Figure 11 and the result is matching exactly with the simulation results. Figure shows the reset, clock, input and compressed output signals. Cursor 'a' shows the starting point of input signal at 'IN' channel and from 'b' onwards shows the compressed values at 'OUT' channel.



	Value	Mean	Min	Max	Std Dev
1 Frequency	2.001MHz	Low signal amplitude			
1 High	5.00 V	5.02	5.00	5.20	63.2m
1 Low	400mV	400m	400m	400m	0.00
2 +Width	1.000µs	1.000µ	999.9n	1.000µ	67.45p
2 High	3.48 V	3.48	3.48	3.48	0.00
2 Low	200mV	200m	200m	200m	0.00
2 Amplitude	3.28 V	3.28	3.28	3.28	0.00
1 Amplitude	4.60 V	4.62	4.60	4.80	63.2m

Figure 11: Results of Hardware verification by test Data

6. CONCLUSION AND FUTURE SCOPE

Lossy image compression techniques enable the storage of digital images using reasonable space and their transmission with acceptable speed. In this project study of different image compression technique has done out of which the single LUT DPCM and multiple LUT DPCM compression techniques are simulated. The implementation of multiple LUT DPCM is successfully done in ProAsic 3E1500 FPGA device. The hardware verification have been done to check the functionality of hardware

The work can extend in future to achieve more compression by avoiding coding redundancy. There are number coding technologies available to use. One method is to transform the image data to bit planes after doing input pixel subtraction. By this method for one image, we will be having a number of bit planes with better correlation. After that variable length coding can be used for achieving high compression ratio.

7. REFERENCES

[1] Arthur R. Weeks, 2004 "Fundamentals of Electronic Image Processing", SPIE Optical Engineering Press, pp 471-522.

[2] Ashok Kumar, Rajiv Kumaran, Sandip Paul, 13-08-2012 "study on DPCM techniques for better radiometric performance" SFED-SEG-SAC.

[3] Michael W. Marcellin, Michael J. Gormish, Ali Bilgin, Martin P. Boliek, 2000 "An Overview of JPEG-2000", IEEE Data Compression Conference, pp. 523-541.

[4] MajidRabbani, Paul W. Jones, 1991 "Digital Image Compression Techniques", Volume TT 7, Donald C. O'Shea, Series Editor Georgia Institute of Technology, pp 49-92.

[5] Young SooSuh, Young Shick Ro and Hee Jun Kang dec2010 "Inertial Sensor Data Compression using Modified ADPCM", 11th international conf. control, automation, robotics and vision Singapore.

[6] S.R Subramanya, Abdou Youssef, 1998 "performance evaluation of Lossy DPCM coding of image using different predictors and quantizers."

[7] Katherine Bouman, VikasRamachandra, KalinAtanassov, Mickey Aleksic and Sergio R. Goma, 2011 "RAW camera DPCM compression performance analysis", SPIE-IS&T Electronic Imaging, SPIE Vol. 7867, 78670N.