Volume: 5 Issue: VI Month of publication: June 2017

DOI:

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Image Compression using Variable Size 2D DCT with FPGA platform

Monika Rani Jha[1], Mr. Neeraj Gupta[2], *Ms. Shruti Karkra[2]
[1]Student, Amity University Gurgaon, Haryana, [2]Asst. Prof., Amity University Gurgaon, Haryana
** Corresponding Author*

Abstract- The discrete cosine transform (DCT) is basically known as compressing mpeg and jpeg images. The basic reason why it is widely used all over is because of its simplicity and easily computed for faster execution. A lot of research is in progress for faster execution of DCT mean while many of algorithms are also developed for compression of video and images. The ancient technique which is being used for limited sizes of images whether it is Chen's technique or Row column decomposition technique. Hardware implementations are difficult and complicated using these codes, have to go through a lengthy and time taking coding scheme. In this work a new and efficient method is proposed using Simulink system generator due to which codes are automatically generated and hardware implementation has become easy. This architecture is also useful for different size of images. It supports variable size image. The performance analysis of hardware utilization is effectively carried out. The generated codes are generated in Verilog and hardware utilization is carried out by FPGA Virtex 5 xc5vlx110t.

Keywords- DCT, Simulink System Generator, FPGA, Verilog.

I. INTRODUCTION

The discrete cosine transform (DCT) introduced in 1974 by Ahmed *et al.* It has become an intensively useful concept for image, audio filters and video signal processing due to its uses and being adopted in standards like Moving Picture Experts Group (MPEG), Joint Photographic Experts Group (JPEG) [21]. DCT is also playing an important role in digital watermarking [3]. In addition to transforms, the working of the DCT is exactly same to the optimal Karhunen–Loeve Transform (KLT) [11].For the real-time audio, image and video processing requirement, DCT and inverse DCT (IDCT) are needed [6]. Different algorithms are developed for computation of Discrete cosine Transforms [8]-[9]. The 2D DCT is used for achieving high speed, high throughput and very less latency computing architectures [18].An effective number of compression methods are present, that are being took to a many applications, as: videophones, compact disc and videoconference, multimedia systems. Such applications calculate the bandwidth of transmission line through the compression applied on such application for use [10].DCT and IDCT combines a particular sort of attribute that allows essential image compression. Compression for video audio, pictures are applied in both hardware and software utilizations [2].

In any case, the equipment executions are particularly significant for the finish of enormously calculations and may gain a dreadful parcel preferable throughput over programming. Digitalizing the world, a lot of research is going to advance as well save the memory. Image compression has played an important role in commercial photography, industrial imaging & video processing.

Image can contain a large data which can further cause complexity in various applications in image processing. Image compression allows mapping of data from a high space to low space. The objective of image compression is to represent an image with few numbers of bits without losing of any bits containing useful information .Image compression transforms the image from time domain into frequency domain which further helps in saving the memory. The most common technique used for transforming is DCT (discrete cosine transforms). It is a kind of lossy transform where transform is done losing of some extravagant information which has no use & occupying large area of memory. As DCT only contains real part where imaginary part is ignored. This dissertation unfolds an efficient way of 2D-DCT with FPGA implementation.

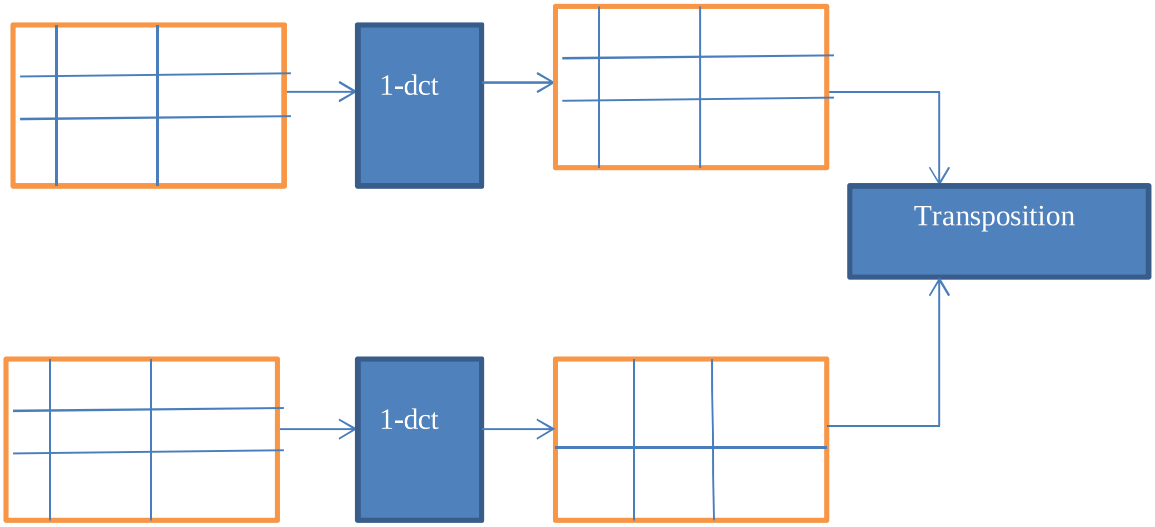


Fig-1 2D DCT

$$A(u)=\sqrt{2}/nc(u)\sum_{x=0}^{n-1}A(x)\cos\frac{(2x+1)u\pi}{2n} \qquad u=0,\ldots\ldots\ldots,n$$

Where $c(u)=2^{-\frac{1}{2}}$ for u=0

=1 otherwise

Equation-1(1D DCT)[3]

For calculating 2-DCT

$$A(u,v)=\frac{\sqrt{2}}{n}*\sqrt{2}/m\sum_{i=0}^{n-1}a(i)\sum_{i=0}^{m-1}a(j).\,cos\frac{(2x+1)u\pi}{2n}.\,cos\frac{(2x+1)v\pi}{2m}\ \text{for}0<n<N\text{-}1$$

=0, otherwise

Equation-2(2D DCT)[4]

A. *Hardware design*

1) *Xilinx system generator:* Xilinx System Generator (XSG) is an Integrated Design Environment (IDE) for FPGAs within the ISE design suite, which enables using SIMULINK, for a model based design. In real time implementation 2D DCT image processing algorithm can be carried with FPGA boards. Here implementation of 2D DCT is done with Virtex 5 xc5vlx110t device and least resources are used. Once the design is modeled in XSG, Verilog or VHDL code can be generated automatically.
2) *Core Generator:* For hardware simulation, the architecture has to be converted to synthesizable code. The entire model is converted to HDL code and then bit file is generated from it. .For this model had to be converted to JTAG hardware and software co- simulation. The block shown below in Fig shows the parameters for target device and generate (*.bit) file is generated.

II. DESIGN FLOW FOR 2D DCT-

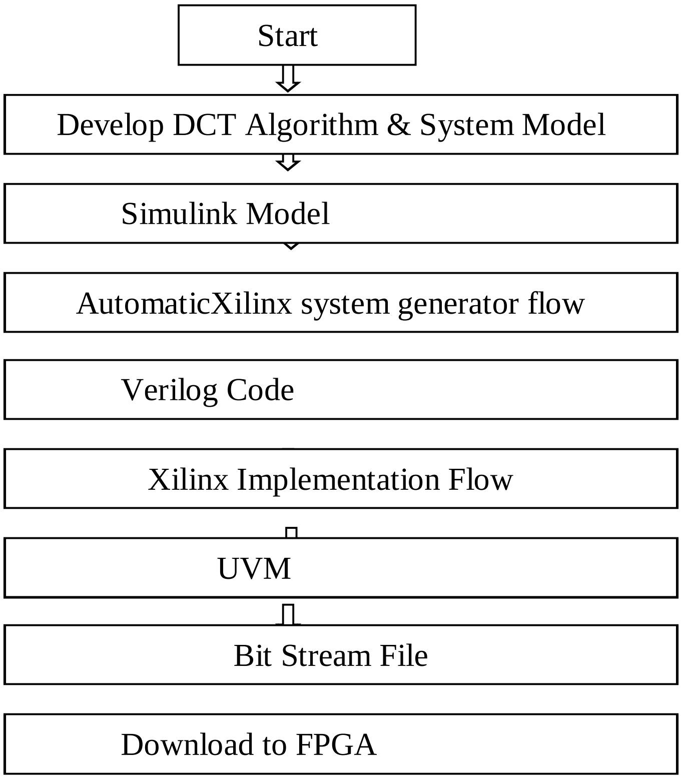

Start
Develop DCT Algorithm & System Model
Simulink Model
AutomaticXilinx system generator flow
Verilog Code
Xilinx Implementation Flow
UVM
Bit Stream File
Download to FPGA

Fig-4 Flow chart for co-simulation

A. *Image processing algorithm for compression*

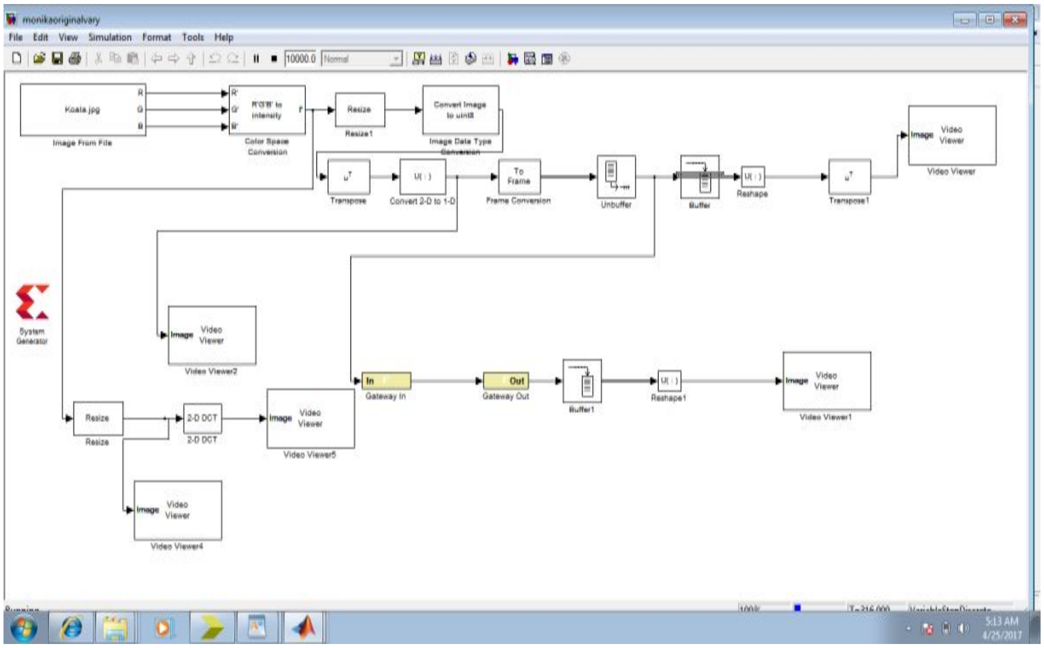


Fig-5 Simulink model for image compression

III. RESULT

Image-1

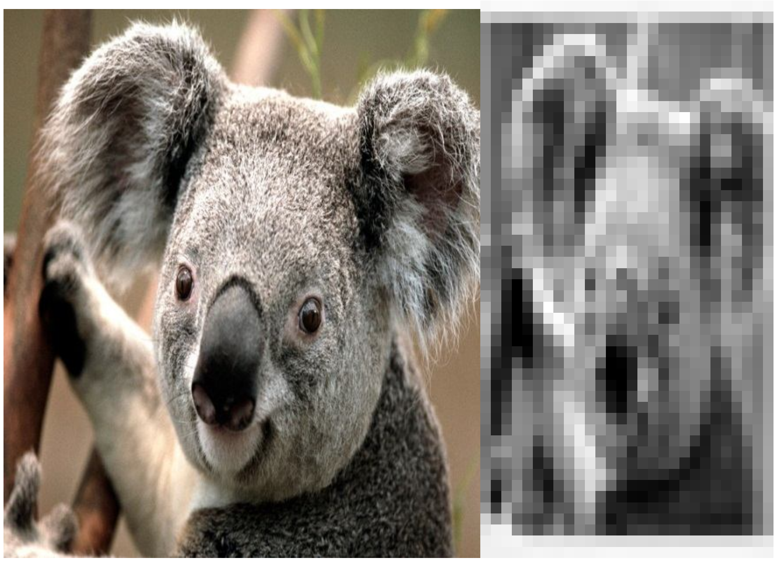


Image-2

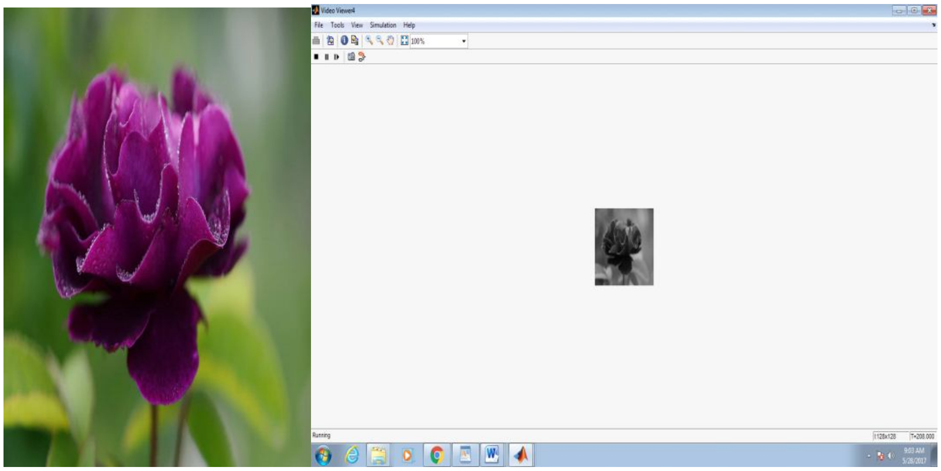
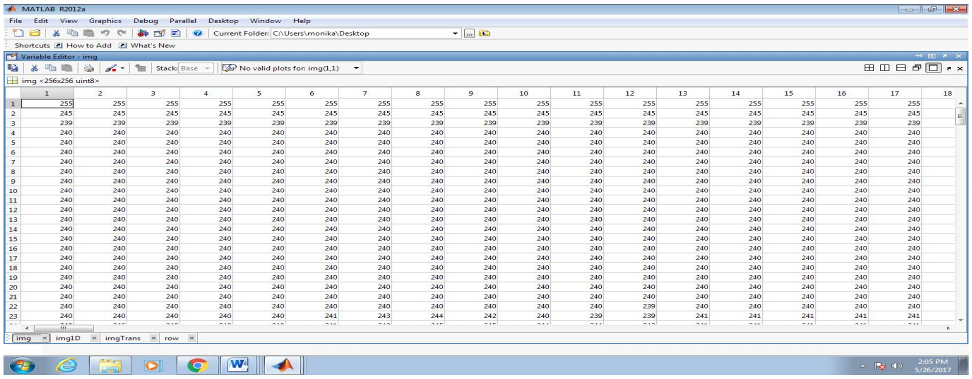


Image-3

IV. DISPLAYING THE IMAGE

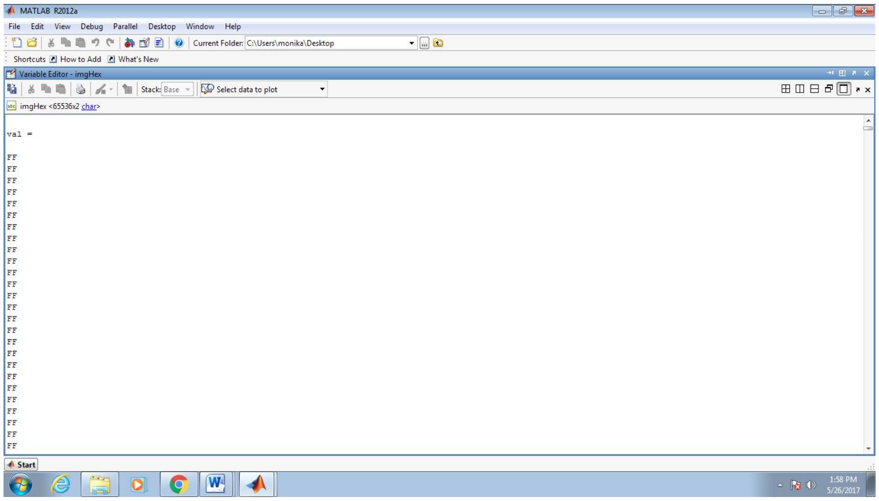

MATLAB R2012a
File Edit View Graphics Debug Parallel Desktop Window Help
Current Folder: C:\Users\monika\Desktop
Shortcuts How to Add What's New
Variable Editor - img
Stack: Base
No valid plots for: img(1,1)
img
img1D
imgTrans
row
2:05 PM
5/26/2017

img <256x256 uint8>

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	
2	245	245	245	245	245	245	245	245	245	245	245	245	245	245	245	245	245	
3	239	239	239	239	239	239	239	239	239	239	239	239	239	239	239	239	239	
4	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
5	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
6	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
7	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
8	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
9	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
10	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
11	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
12	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
13	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
14	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
15	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
16	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
17	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
18	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
19	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
20	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
21	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	
22	240	240	240	240	240	240	240	240	240	240	240	239	240	240	240	240	240	
23	240	240	240	240	240	241	243	244	242	240	239	239	241	241	241	241	241	

Original image pixel

Convert the pixel into hex file.


MATLAB R2012a
File Edit View Debug Parallel Desktop Window Help
Current Folder: C:\Users\monika\Desktop
Shortcuts How to Add What's New
Variable Editor - imgHex
Stack: Base
Select data to plot
Start
1:58 PM
5/26/2017

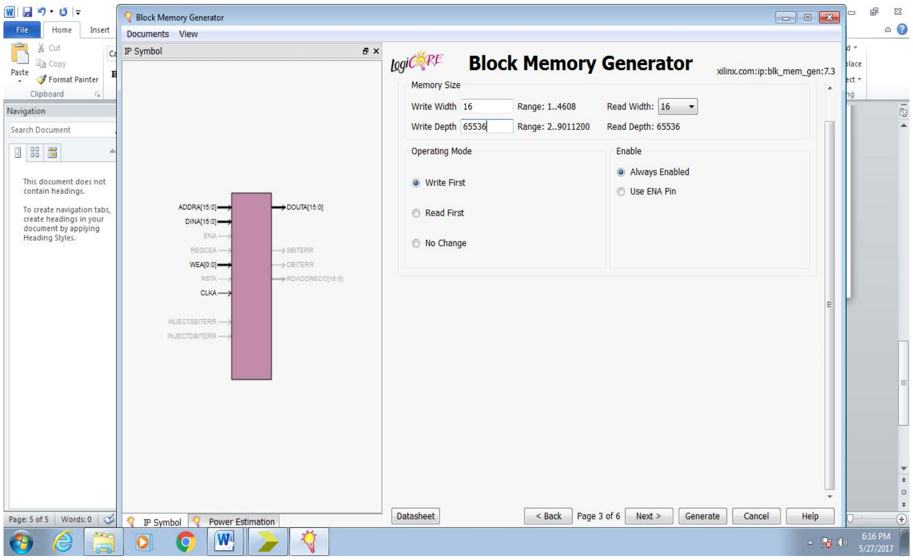
imgHex <65536x2 char>

```
val =

FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
FF
```

Hex file with extension coe

Read the coe file by setting the write depth.



Block Memory Generator

xilinx.com:ip:blk_mem_gen:7.3

Memory Size

Write Width 16 Range: 1..4608 Read Width: 16

Write Depth 65536 Range: 2..9011200 Read Depth: 65536

Operating Mode

● Write First

○ Read First

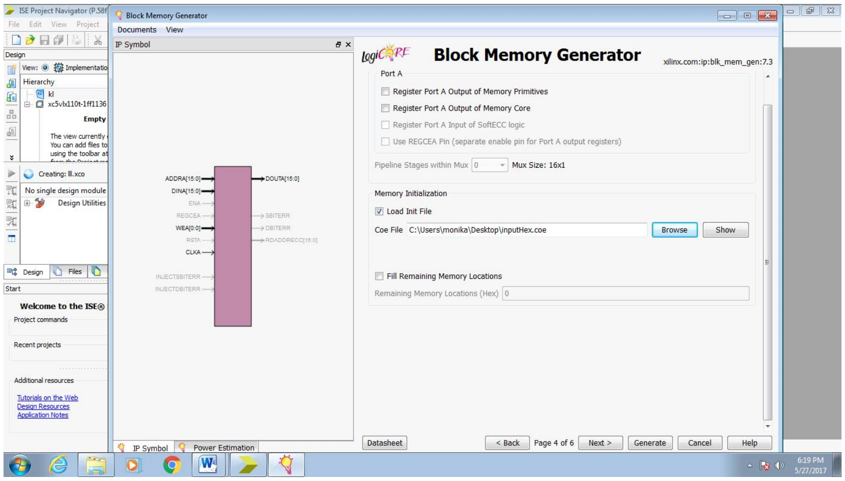
○ No Change

Enable

● Always Enabled

○ Use ENA Pin

Width and Depth specification



Block Memory Generator xilinx.com:ip:blk_mem_gen:7.3

Port A

□ Register Port A Output of Memory Primitives
□ Register Port A Output of Memory Core
□ Register Port A Input of SoftECC logic
□ Use REGCEA Pin (separate enable pin for Port A output registers)

Pipeline Stages within Mux: 0 Mux Size: 16x1

Memory Initialization

☑ Load Init File

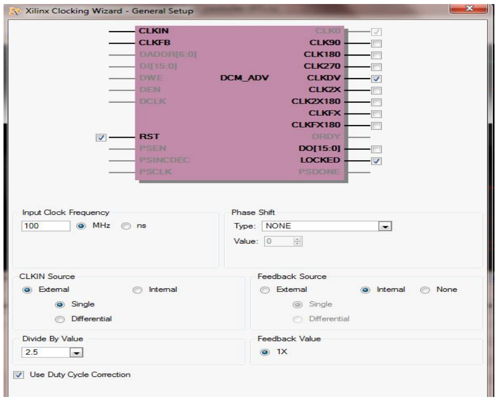
Coe File: C:\Users\monika\Desktop\inputHex.coe

□ Fill Remaining Memory Locations

Remaining Memory Locations (Hex): 0

Page 4 of 6

Then click on generate. Xcw and .v file are created, synthesizing .v file we've ngc file. Copy these file to the main module. Generate DCM clock.



Xilinx Clocking Wizard - General Setup

Input Clock Frequency: 100 ● MHz ○ ns

Phase Shift Type: NONE Value: 0

CLKIN Source: ● External ○ Internal; ● Single ○ Differential

Feedback Source: ○ External ● Internal ○ None

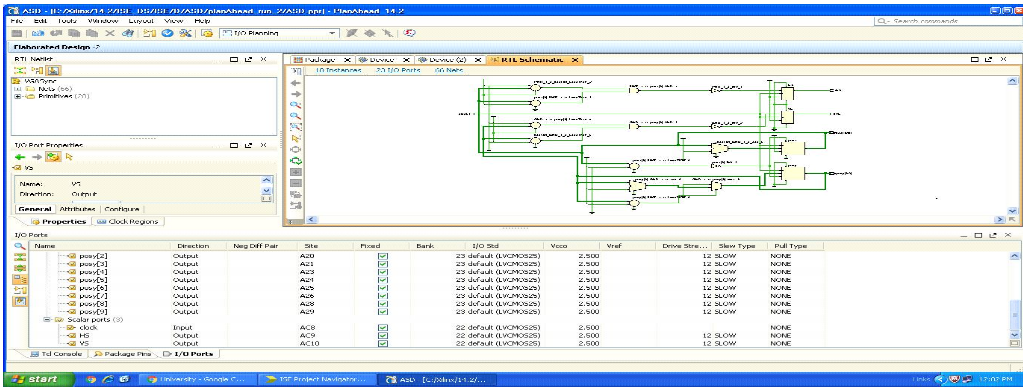
Divide By Value: 2.5

Feedback Value: ● 1X

☑ Use Duty Cycle Correction

A. DCM clock

The DCM block (Digital clock manager) can be made using IP core and the clocking frequency can also be setup. The other ports can be easily be enabled by just checking it. As shown above the DCM block is designed at 100/2.5 MHz. The phase shift value is given none and clkdv is selected in the block.



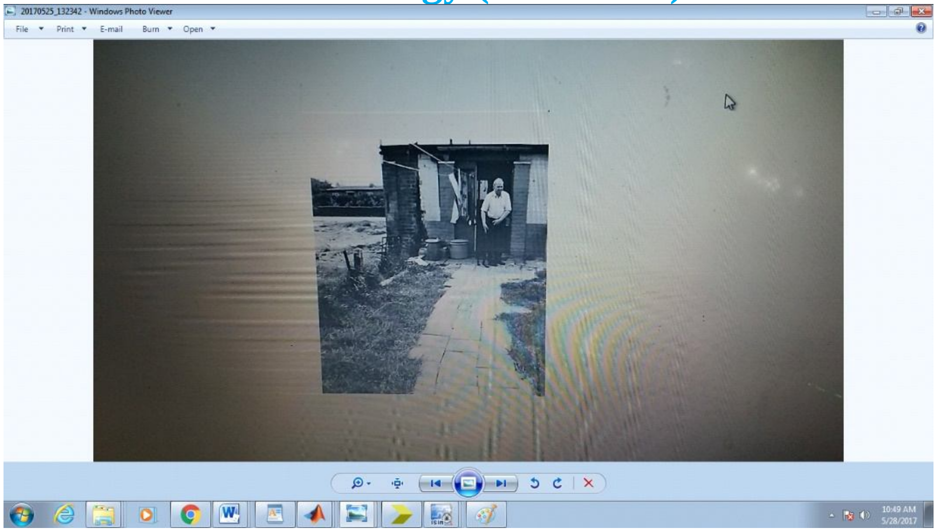
ASD - [C:/Xilinx/14.2/ISE_DS/ISE/D/ASD/planAhead_run_2/ASD.ppr] - PlanAhead 14.2

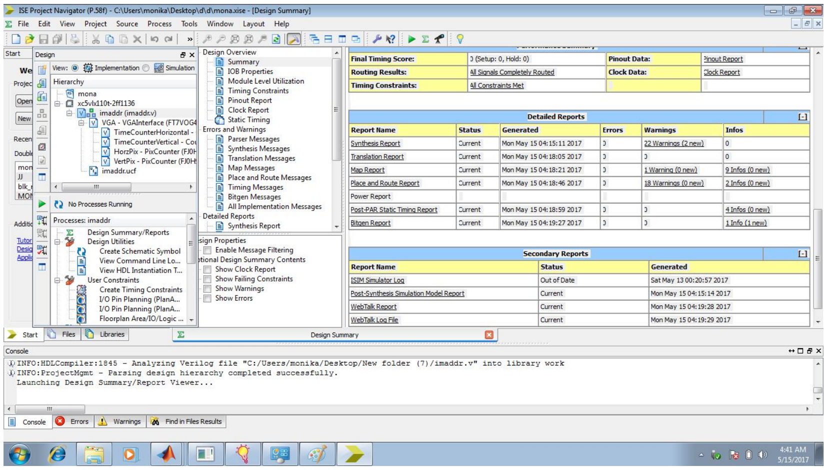
I/O Port Properties: Name: VS Direction: Output

I/O Ports

Name	Direction	Neg Diff Pair	Site	Fixed	Bank	I/O Std	Vcco	Vref	Drive Stre...	Slew Type	Pull Type
posy[2]	Output		A20	☑	23	default (LVCMOS25)	2.500		12	SLOW	NONE
posy[3]	Output		A21	☑	23	default (LVCMOS25)	2.500		12	SLOW	NONE
posy[4]	Output		A23	☑	23	default (LVCMOS25)	2.500		12	SLOW	NONE
posy[5]	Output		A24	☑	23	default (LVCMOS25)	2.500		12	SLOW	NONE
posy[6]	Output		A25	☑	23	default (LVCMOS25)	2.500		12	SLOW	NONE
posy[7]	Output		A26	☑	23	default (LVCMOS25)	2.500		12	SLOW	NONE
posy[8]	Output		A28	☑	23	default (LVCMOS25)	2.500		12	SLOW	NONE
posy[9]	Output		A29	☑	23	default (LVCMOS25)	2.500		12	SLOW	NONE
Scalar ports (3)											
clock	Input		AC8	☑	22	default (LVCMOS25)	2.500				NONE
HS	Output		AC9	☑	22	default (LVCMOS25)	2.500		12	SLOW	NONE
VS	Output		AC10	☑	22	default (LVCMOS25)	2.500		12	SLOW	NONE

Fig-6. RTL Schematic





Final Timing Score:	0 (Setup: 0, Hold: 0)	**Pinout Data:**	Pinout Report
Routing Results:	All Signals Completely Routed	**Clock Data:**	Clock Report
Timing Constraints:	All Constraints Met		

Detailed Reports

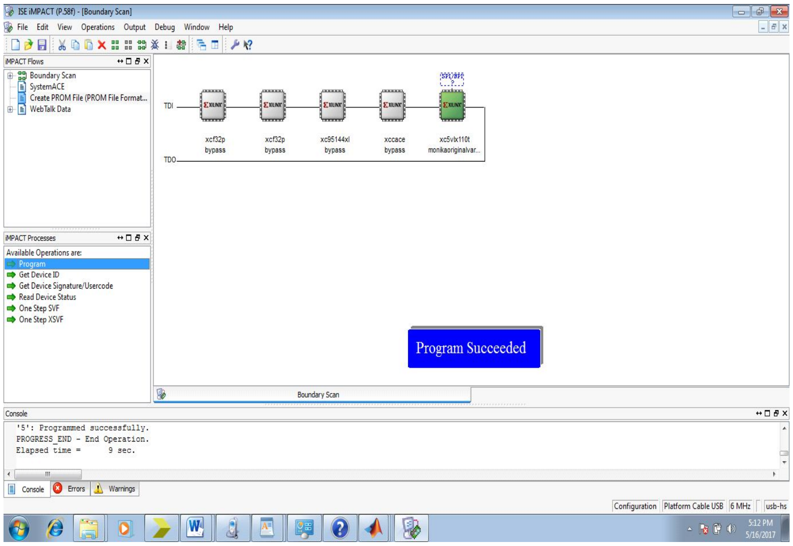
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Mon May 15 04:15:11 2017	0	22 Warnings (2 new)	0
Translation Report	Current	Mon May 15 04:18:05 2017	0	0	0
Map Report	Current	Mon May 15 04:18:21 2017	0	1 Warning (0 new)	9 Infos (0 new)
Place and Route Report	Current	Mon May 15 04:18:46 2017	0	18 Warnings (0 new)	2 Infos (0 new)
Power Report					
Post-PAR Static Timing Report	Current	Mon May 15 04:18:59 2017	0	0	4 Infos (0 new)
Bitgen Report	Current	Mon May 15 04:19:27 2017	0	0	1 Info (1 new)

Secondary Reports

Report Name	Status	Generated
ISIM Simulator Log	Out of Date	Sat May 13 00:20:57 2017
Post-Synthesis Simulation Model Report	Current	Mon May 15 04:15:14 2017
WebTalk Report	Current	Mon May 15 04:19:28 2017
WebTalk Log File	Current	Mon May 15 04:19:29 2017

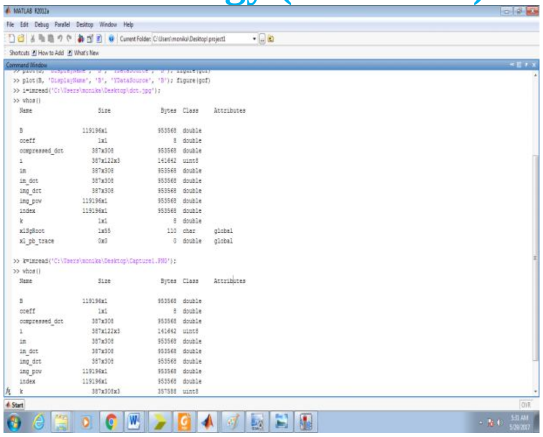
```
INFO:HDLCompiler:1845 - Analyzing Verilog file "C:/Users/monika/Desktop/New folder (7)/imaddr.v" into library work
INFO:ProjectMgmt - Parsing design hierarchy completed successfully.
Launching Design Summary/Report Viewer...
```

Fig-8 synthesis report



```
'5': Programmed successfully.
PROGRESS_END - End Operation.
Elapsed time =      9 sec.
```

Fig-9 FPGA simulation


MATLAB R2012a
File Edit Debug Parallel Desktop Window Help
Current Folder: C:\Users\monika\Desktop\project1
Shortcuts How to Add What's New
Command Window
Start

```
>> plot(B, 'DisplayName', 'B', 'YDataSource', 'B'); figure(gcf)
>> i=imread('C:\Users\monika\Desktop\dct.jpg');
>> whos()
  Name                 Size              Bytes  Class     Attributes

  B                 119196x1            953568  double
  coeff                  1x1                 8  double
  compressed_dct       387x308          953568  double
  i                    387x122x3        141642  uint8
  im                   387x308          953568  double
  im_dct               387x308          953568  double
  img_dct              387x308          953568  double
  img_pow           119196x1            953568  double
  index             119196x1            953568  double
  k                      1x1                 8  double
  xlSgRoot               1x55              110  char      global
  xl_pb_trace            0x0                 0  double    global

>> k=imread('C:\Users\monika\Desktop\Capture1.PNG');
>> whos()
  Name                 Size              Bytes  Class     Attributes

  B                 119196x1            953568  double
  coeff                  1x1                 8  double
  compressed_dct       387x308          953568  double
  i                    387x122x3        141642  uint8
  im                   387x308          953568  double
  im_dct               387x308          953568  double
  img_dct              387x308          953568  double
  img_pow           119196x1            953568  double
  index             119196x1            953568  double
  k                    387x308x3        357588  uint8
```

Compressed image detail

V. CONCLUSION

The DCT is widely used for compression system. In comparison with different transform which uses lots of multipliers and adder, It needs less multiplier and adders. The total power used in this project is 1.042 w. no. of slices used 1out of 69120. Only 1% is utilized here. This is comparatively low to the previous technique of implementing 2D DCT. 8 luts are used out of 69,120. The utilization is 1%. Which is very less than popular algorithm of chen's of implementing 2D DCT and other algorithm. Total fan out is 1.39 and 34.5% compression is done using this technique.

REFERENCES

[1] Kathoriya, S. Patel and M. Goyani, "Comparative analysis of DCT and DWT Techniques of image compression," Journal of Information Engineering and Applications, P.P-1-5, VOL. 1, NO. 2 ISSN: 2030-4100, 2011.

[2] H. L. P. A. Madanayake, R. J. Cintra, D. Onen, V. S. Dimitrov and T. Bruton, "Algebraic integer based 88 2-d DCT architecture for digital video processing," in IEEE International Symposium on Circuits and System, ISSN:20041-20043, 2012.

[3] M. El Aakif, A. Belkouch, N. Chabini, and M. Hassani, "Low power and fast DCT architecture using Multiplier-less method," in Faible Tension FaibleConsommation (FTFC), 2013, P.P- 63–66.

[4] T. Pradeepthi and P. Ramesh, "Pipelined architecture of 2d-dct, quantization and zigzag process for JPEG image compression using vhdl," International Journal of VLSI design & communication systems (vlsics), VOL. 2, NO. 3, P.P- 99–110, September 2014.

[5] H.V., Jeyakumar, A. Kasat, P.S., Warty, C., "VLSI design of fast DCTQ-IQIDCT processor for real time image compression," Tenth International Conference on Wireless and Optical Communications Networks (WOCN), VOL.1, NO. 5, P.P-26-28 July 2015.G. Ravi kumar, G. Sateesh Kumar" Implementation of 2-D DCT Architecture for Optimized Area And Power Utilization IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) Volume 5, Issue 4, Ver. I, P.P-11-16, ISSN: 2319 – 4200, Jul - Aug. 2015

[6] R. Uma, "FPGA implementation of 2D-DCT for JPEG image compression," International Journal of Advanced Engineering Sciences and Technologies (IJAEST), P.P-21-26, VOL. 7, NO.1, 2015.

[7] R.R.A.S. Narasimha Reddy, T.Madhu "Image Compression Using 1-D, 2-D DCT And 3-D Discrete Cosine Transform" IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) ISSN: 2278-8735.Volume 9, Issue 2, Ver. III (. 2015), P.P-07-1, Mar – Apr,2015

[8] P. Kasat, D. Bilaye, H. V. Dixit, R. Balwaik and A. Jeyakumar, "Multiplication Algorithms for VLSI-a review," International Journal on Computer Science and Engineering (IJCSE), VOL. 4, NO. 11, P.P-1761– 1765, Nov., 2015.

[9] M. Nisha Monnappa& Sonia Kuwelkar" Implementation of Image Compression Using CL-DCT on FPGA" International Journal of Innovative Research in Science, Engineering and Technology Vol. 5, Special Issue 9, May 2016.

[10] Reem T. Haweel "Fast Approximate DCT with GPU Implementation for Image Compression" Journal Visual Communication, P.P-1414-1415, ISSN- 2100-2125, 7 July 2016.

[11] Min. Chen" Efficient architecture of variable size HEVC 2D-DCT for FPGA platform" International Journal Electronics and Communication, P.P-1-4, ISSN-2121-2123, 2 Jan 2017

[12] Text book of "Advance image processing" by Manoj K. Arora.