Qiao Gao

ECE 584

DCT Implementation based on Algorithmic Strength Reduction and binDCT Algorithm

05/04/2014

Abstraction

This paper introduces FPGA implementations of Discrete Cosine Transform (DCT) depending on Algorithm Strength Reduction and binDCT algorithm. DCT is a frequent use calculation for telecommunication and image processing. Therefore, a fast and low resource usage hardware implementation of DCT is highly demanded. In this report, specifically, an 8-input-8-output DCT module will be sketched using both Algorithm Strength Reduction and binDCT, which are all algorithms to gain low cost and high efficiency design of DCT. The function correctness of both designs will be test and a comparison between them will be made in terms of both scale and timing.

Keywords: FPGA, DCT, Algorithm Strength Reduction, binDCT

1 Introduction

The main purpose of this project is to compare different DCT algorithm implementations on function accuracy, resource use and throughput. Firstly, Verilog codes of these designs depending on all the structures are done in Modelsim firstly. After that, all the simulation data will be used to compare with the Matlab build in DCT to check correctness. Then, synthesis will be done in ISE and both scaling and timing report will be got at the same time. Depending on these reports, we can tell the differences between DCT structures and choose a better one from them.

The challenge of this project focuses on:

1. All of the design should be founded on gate level with structural Verilog in order to make perform the exact structure of specific structure.

2. All the operands are in 2' s complement with sign bits. As a result, it is important to scale all the values for shifters and adders in this design to avoid overflow. Also, there should be an overflow network to check any possible overflow in this design.

3. Make high efficiency multipliers for constant values

To solve these problems, methods below are used correspondingly:

1. All of the designs are made from the most basic level. I design a half adder and a full adder first at gate level description. Then, all the structures can be implemented depending on half adders, full adders and some basic logic gate.

2. Expand all the operands to the highest possible bit of the result so that all the values inside the circuit can be contained in the expended bits with no ignorance. Build all overflow possible blocks with overflow check signals and combine all of them together in top module as an overflow output.

3. Scale the constant operands for multiplier correctly, reduce the numerical strength case-by-case and build specified multiplier for each operand.

All design details will be served in part 3.

2 Background

2.1 Discrete Cosine Transform

The Discrete Cosine Transform is useful function in communication device and image/video

processing. In Matlab, DCT is provided as a build in function: y=dct(x), which complete the function below:

$$y(k) = w(k)\sum_{n=1}^{N} x(n)\cos\left(\frac{\pi}{2N}(2n-1)(k-1)\right), \quad k = 1,2,\ldots,N,$$

where

$$w(k) = \begin{cases} \frac{1}{\sqrt{N}}, & k = 1, \\ \sqrt{\frac{2}{N}}, & 2 \le k \le N, \end{cases}$$

In our case, for 8-input-8-output design, N=8. Moreover, as we count from zero instead of one, we can rewrite the formula as below:

$$y(k) = \omega(k)\sum_{n=0}^{7} x(n)\cos\left[\frac{(2n+1)k\pi}{16}\right], \qquad k = 0,1\ldots,7$$

$$\omega(k) = \begin{cases} \frac{1}{2\sqrt{2}}, & k = 0 \\ \frac{1}{2}, & 1 \le k \le 7 \end{cases}$$

The number of multiplication for DCT is N(N-1) which is a big value and may slow down the whole system. To reduce the number of multiplication in this design for hardware optimization, we should use algorithm strength reduction method or another algorithm called binDCT.

2.2 Algorithm Strength Reduction

The DCT module of 8 inputs with algorithm strength reduction is shown below:

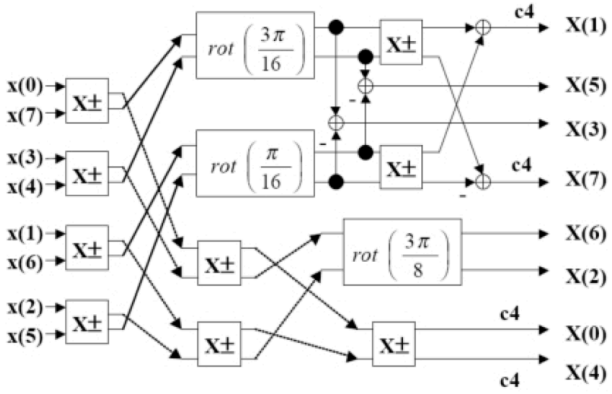


Figure 1

Where

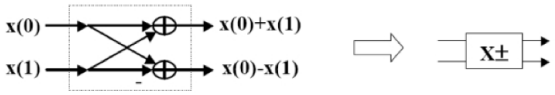
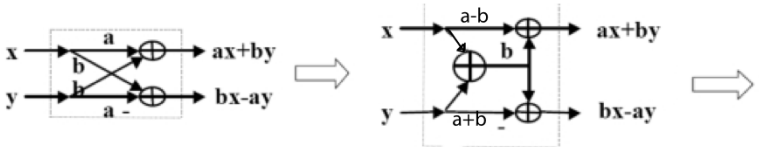


Figure 2

and for the rotation modules:



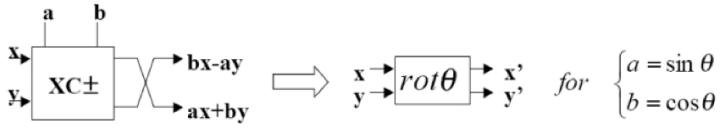


Figure 3

It is easy to find that the number of multipliers is highly reduced from 8*(8-1)=56 to in this structure to 13. One thing that should be noticed is the scale value $\omega(k)$ is not the same with the Matlab build in DCT function, which is:

$$\omega(k) = \begin{cases} \frac{1}{\sqrt{2}}, & k = 0 \\ 1, & 1 \le k \le 7 \end{cases}$$

So we have $c4 = \frac{1}{\sqrt{2}}$. The final result of DCT Algorithm Strength Reduction should be one time larger than the result from both Matlab and binDCT.

2.3 binDCT Algorithm

In binDCT the number of multiplications is reduced to zero, which means we can finish DCT by using just shift and add function. Any rotation function in the original design could be transformed by using a method called lifting. For example, lifting steps of a rotation could be performed as below:

$$\begin{bmatrix} cos(\alpha) & -sin(\alpha) \\ sin(\alpha) & cos(\alpha) \end{bmatrix} = \begin{bmatrix} 1 & p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ u & 1 \end{bmatrix} \begin{bmatrix} 1 & p \\ 0 & 1 \end{bmatrix}$$

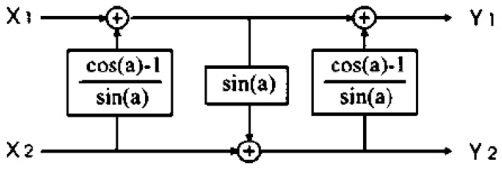


Figure 4

Since there will always be a scale value for every end of the outputs, we can optimize the lifting steps to a scaled lifting structure:

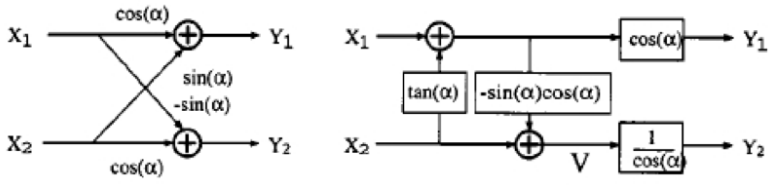


Figure 5

By doing this to the rotations before outputs, we can merge some of the coefficients to the scaling value and reduce the complexity of calculation. Eventually, we can get the structure below for 8-input-8-output design.

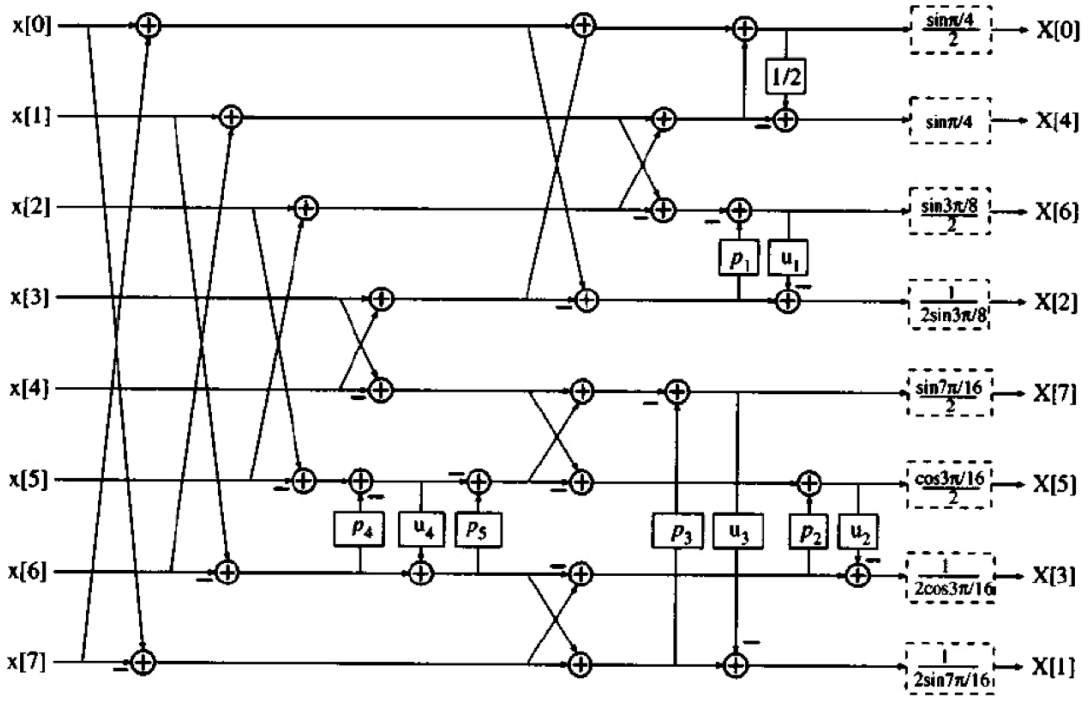


Figure 6

In which all the parameters are shown below:

	Floating-point	C5
p_1	0.4142135623	3/8
u_1	0.3535533905	3/8
p_2	0.6681786379	7/8
u_2	0.4619397662	1/2
p_3	0.1989123673	3/16
u_3	0.1913417161	3/16
p_4	0.4142135623	7/16
u_4	0.7071067811	11/16
p_5	0.4142135623	3/8
Shifts	-	17
Adds	-	36
MSE	-	$4.2e-4$
$C_g(8)$ (dB)	-	8.8159
$C_g(4)$ (dB)	-	7.5566

Figure 7

3 Implementation Details

To implement both designs, firstly, we should build the basic element modules first. Here is the list for what we need.

1. A bit extension module: all inputs should be extended to more bits to avoid overflow.
2. A multi-bit two operands adder which can perform both add and subtract operation.
3. A direct transformation block to achieve butterfly function with a=1 and b=1.
4. Scaling modules for all the scale function in this design. For example, c4 in the first design, p1 in the second design.
5. Rotation blocks with adders and multipliers.

3.1 Word length extension

Firstly, we have to expand the length of operands. More specifically, in this design, we have eight 8-bit 2' s complement inputs. So the largest absolute value should be $2^7 - 1 = 127$. When k=0, we get the maximum value of the output, which is the add up of all the inputs: X(0)=x(0)+x(1)+x(2)+x(3)+x(4)+x(5)+x(6)+x(7). Then $X(0)_{max}$ should be 8*127=1016. As the outputs

are also 2's complement numbers, the output bit number should be $\lceil \log_2 1026 \rceil + 1 = 11$.

So we need three more bits to avoid overflow in our design. I specified the function with Verilog as below:

```
module ex8_11(out, in);

 input [7:0] in;
 output [10:0] out;

assign out[10:8] =  {in[7],in[7],in[7]};
assign out[7:0]  =  in[7:0];

endmodule
```

Figure 8

3.2 Adder design

To make all the basic functions all in gate level description, I design a half adder a full adder as basic elements of all the structures. In detail, the structures of full adder and half adder are shown in Figure 9. Referring to this, I made the Verilog descriptions for them as below: (Figure 10)

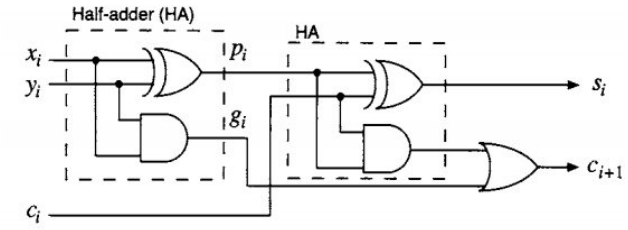


Figure 9

```
module halfadder(x,y,s,co);
    input x,y;
    output s,co;

    xor(s,x,y);
    and(co,x,y);
endmodule
```

(a) Verilog code of half adder

```
module fulladder(x,y,ci,s,co);
    input x,y,ci;
    output s,co;

    wire p,g,o;

    halfadder HA1(x,y,p,g);
    halfadder HA2(p,ci,s,o);
    or(co,o,g);

endmodule
```

(a) Verilog code of full adder

Figure 10

Then, for 11-bit operands, connect 11 full-adders in carry ripple structure:

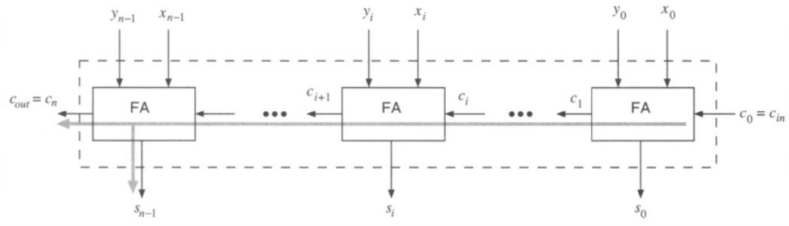


Figure 11

Notice that we do not really need signal co in our design, since there is no propagation structure between two 2-operand adders. Instead, we use co to determine the overflow signal of with the code below:

```
fulladder a0(s[0], c1, a[0], b[0], ci);
fulladder a1(s[1], c2, a[1], b[1], c1);
fulladder a2(s[2], c3, a[2], b[2], c2);
fulladder a3(s[3], c4, a[3], b[3], c3);
fulladder a4(s[4], c5, a[4], b[4], c4);
fulladder a5(s[5], c6, a[5], b[5], c5);
fulladder a6(s[6], c7, a[6], b[6], c6);
fulladder a7(s[7], c8, a[7], b[7], c7);
fulladder a8(s[8], c9, a[8], b[8], c8);
fulladder a9(s[9], c10, a[9], b[9], c9);
fulladder a10(s[10], co, a[10], b[10], c10);

xor(of, co, c10);
```

Figure 12

```
wire [10:0]  xb;

xor x0(xb[0], b[0], sub);
xor x1(xb[1], b[1], sub);
xor x2(xb[2], b[2], sub);
xor x3(xb[3], b[3], sub);
xor x4(xb[4], b[4], sub);
xor x5(xb[5], b[5], sub);
xor x6(xb[6], b[6], sub);
xor x7(xb[7], b[7], sub);
xor x8(xb[8], b[8], sub);
xor x9(xb[9], b[9], sub);
xor x10(xb[10], b[10], sub);

adder11 adder(s, of, a, xb, sub);
```

Figure 13

Then we need add subtract function to the adder. Build a block “addsub” with a sub signal, use XOR function with every bit of the second input signal to get the 1’s complement value. Then make “sub” as the “ci” of the adder to get the equivalent 2’s complement as the operand of the adder. By doing this we finish the subtract function. The Verilog code should be what is shown in Figure 13.

For the multipliers, since one of the operand I choose to leave a 6 bits decimal part, an 11+6=17 bits adder is needed. So I build a 17 bits adder with the same method above. The detail of the multiplier will be found in the rotation part.

3.3 Direct Transformation

As it is shown in Figure 2, a direct transformation module can be built by simply applying follow code:

```
wire of1,of2;

addsub11 trans_adder1(out1, of1, in1, in2, 1'b0);
addsub11 trans_adder2(out2, of2, in1, in2, 1'b1);
or(of,of1,of2);
```

Figure 14

Moreover, this code also shows how overflow is collected when there is more than one overflow signal in a module. I use “or” function and this method is implemented to all the modules with multi overflow signal.

3.4 Scaling module

As I have mentioned, the scaling module should be achieved by only shift and add operations. Use p1 as an example, as we see from Figure 7, p1=3/8=1/2-1/8. So we can get two shift value firstly, then do the subtraction. The code could be:

```
wire [10:0] in_ex_1,in_ex_2;

assign in_ex_1={in[10],in[10:1]};
assign in_ex_2={in[10],in[10],in[10],in[10:3]};

addsub11 p1_adder1(out, of, in_ex_1, in_ex_2, 1'b1);
```

Figure 15

All the scale modules in binDCT are implemented by this method.

3.5 Rotation module

To achieve less resource usage, for all the multipliers in rotation part, I choose to design each one case by case to make optimization since all the scale values are constant. Specifically, in order to achieve $\mathrm{rot}(\frac{\pi}{16})$, we should get the binary value of each coefficient first:

$$a = \sin\left(\frac{\pi}{16}\right) = 0.195090322 = 0.001100$$

$$b = \cos\left(\frac{\pi}{16}\right) = 0.98078528 = 0.111111 = 1.00000\bar{1}$$

$$a - b = \sin\left(\frac{\pi}{16}\right) - \cos\left(\frac{\pi}{16}\right) = -0.785694958 = -0.110010$$

$$a + b = \sin\left(\frac{\pi}{16}\right) + \cos\left(\frac{\pi}{16}\right) = 1.175875602 = 1.001011$$

In which we only use b, a-b and a+b according to Figure 3. For b, I make a numerical strength reduction. For a-b, though the value is negative, we can still use its absolute value and make a subtract operation in the following adder. In terms of multiplier, use b as an example, I build the function with following code:

```
wire [16:0] in_ex_1,in_ex_2,out_ex;

assign in_ex_1={in[10:0],6'b0};
assign in_ex_2={in[10],in[10],in[10],in[10],in[10],in[10],in[10:0]};

addsub17 b2_adder1(out_ex, of, in_ex_1, in_ex_2, 1'b1);

assign out=out_ex[16:6];
```

Figure 16

The principle could be shown as below:

Input[10:0]						x	x	x	x	x	x	x	x	x	x	x
	*									1	0	0	0	0	0	$\bar{1}$
	x	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0
	-						x	x	x	x	x	x	x	x	x	x	x
	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
	|	|	|	|	|	|	|	|	|	|	|						
Output[10:0]	x	x	x	x	x	x	x	x	x	x	x						

As a multiplier we cannot abandon any part of the operands before calculation, so we cannot shift operands right as what we have done in the scaling part. Instead, I scale up the decimal part to integer and down scale after calculation. Moreover, a 17-bit adder is needed, that is why I make a 17-bit adder additionally.

Additionally, for a+b, use the code below:

```
wire [16:0] in_ex_1,in_ex_2,in_ex_3,in_ex_4,out_ex_1,out_ex_2,out_ex;
wire of1,of2,of3;

assign in_ex_1={in[10:0],6'b0};
assign in_ex_2={in[10],in[10],in[10],in[10:0],3'b0};
assign in_ex_3={in[10],in[10],in[10],in[10],in[10],in[10:0],1'b0};
assign in_ex_4={in[10],in[10],in[10],in[10],in[10],in[10],in[10:0]};

addsub17 aplusb1_adder1(out_ex_1, of1, in_ex_1, in_ex_2, 1'b0);
addsub17 aplusb1_adder2(out_ex_2, of2, in_ex_3, in_ex_4, 1'b0);
addsub17 aplusb1_adder3(out_ex, of3, out_ex_1, out_ex_2, 1'b0);

or(of,of1,of2,of3);

assign out=out_ex[16:6];
```

Figure 17

Though it has four operands inside and it is not easy to do numerical strength reduction, I build a tree structure for the three adders to reduce propagation.

All the multipliers are made one by one through this method. Then we can connect them and 11-bit adders as Figure 3. The code for $\mathrm{rot}(\frac{\pi}{16})$ is shown below:

```
wire [10:0] add1,mult1,mult2,mult3;
wire of1,of2,of3,of4,of5,of6;

aminusb2 rot2_aminusb(mult1,of1,in1);
addsub11 rot2_adder1(add1, of2, in1, in2, 1'b0);
b2 rot2_b(mult2,of3,add1);
aplusb2 rot2_aplusb(mult3,of4,in2);
addsub11 rot2_adder2(out2, of5, mult2, mult1, 1'b1);
addsub11 rot2_adder3(out1, of6, mult2, mult3, 1'b1);

or(of,of1,of2,of3,of4,of5,of6);
```

Figure 18

Finally, connect the extension module, adders, transformation module, scaling module and rotation module all together as Figure 1 and Figure 6 to implement both Algorithm Strength

Reduction DCT and binDCT correctly. (All source code can be find in Appendix Ⅰ)

4 Functional Validation

4.1 RTL simulation

RTL simulation is the process to ensure that all our designs have the correct functions we want, which is essential before we start logic synthesis. In order to do RTL simulation, we can add a series of stimulating signals, which is known as a test bench, to the top level of our design. The code of all the test benches can be found in Appendix Ⅰ.

Set the input in the test bench with three set of signals with step of 5:

x0 = 8'd0; x1 = 8'd5; x2 = 8'd10; x3 = 8'd15; x4 = 8'd20; x5 = 8'd25; x6 = 8'd30; x7 = 8'd35;

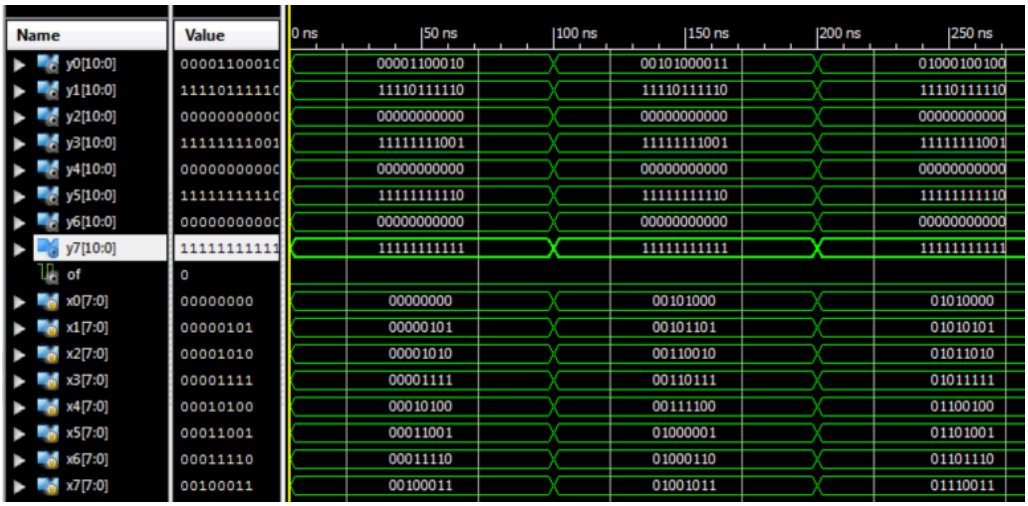
#100;

x0 = 8'd40; x1 = 8'd45; x2 = 8'd50; x3 = 8'd55; x4 = 8'd60; x5 = 8'd65; x6 = 8'd70; x7 = 8'd75;

#100;

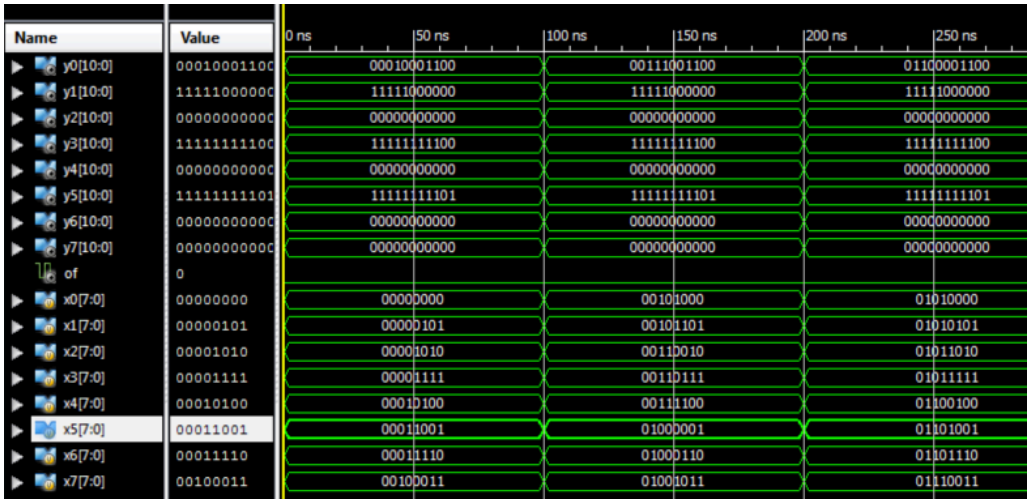
x0 = 8'd80; x1 = 8'd85; x2 = 8'd90; x3 = 8'd95; x4 = 8'd100; x5 = 8'd105; x6 = 8'd110; x7 = 8'd115;

Running the test bench for both Algorithm Strength Reduction (DCT_ASR) and binDCT in ISE for simulation. Then we get the waveform below:



Algorithm Strength Reduction DCT

Figure 19



binDCT

Figure 20

As all the outputs are in 2's complement mode, rewrite the result in signed integer, we get:

DCT_ASR	X(0)	X(1)	X(2)	X(3)	X(4)	X(5)	X(6)	X(7)
0-35	98	-66	0	-7	0	-2	0	-1
4-75	323	-66	0	-7	0	-2	0	-1
80-115	548	-66	0	-7	0	-2	0	-1

binDCT	X(0)	X(1)	X(2)	X(3)	X(4)	X(5)	X(6)	X(7)
0-35	140	-64	0	-4	0	-3	0	0
4-75	460	-64	0	-4	0	-3	0	0
80-115	780	-64	0	-4	0	-3	0	0

Table 1

Next step is to compare the result to the result of Matlab DCT function to check correctness.

Before the comparison, as I have mentioned in Part2.2, all the values should be divided by two in the result of DCT_ASR. For binDCT, as all the scaling modules with dash box are not included in our design, we should multiply all the parameters in Matlab. Use the Matlab code below:

```
bindct_y00=[140, -64, 0, -4, 0, -3, 0, 0];
sc=[sin(pi/4)/2,1/2/sin(7*pi/16),1/2/sin(3*pi/8),1/2/cos(3*pi/16),sin(pi/4),cos(3*pi/16)/2,sin(3*pi/8)/2,sin(7*pi/16)/2];
bindct_y0=sc.*bindct_y00;
```

Make all the function above in Matlab, and use y=dct(x) to get the accurate values in Matlab. Then the result is:

0-35	X(0)	X(1)	X(2)	X(3)	X(4)	X(5)	X(6)	X(7)
Matlab	49.49	-32.21	0	-3.36	0	-1.00	0	-0.25
DCT_ASR	49	-33	0	-3.50	0	-1	0	-0.50
binDCT	49.49	-32.62	0	-2.40	0	-1.24	0	0

40-75	X(0)	X(1)	X(2)	X(3)	X(4)	X(5)	X(6)	X(7)
Matlab	162.63	-32.21	0	-3.36	0	-1.00	0	-0.25
DCT_ASR	161.50	-33	0	-3.50	0	-1	0	-0.50
binDCT	162.63	-32.62	0	-2.40	0	-1.24	0	0

80-115	X(0)	X(1)	X(2)	X(3)	X(4)	X(5)	X(6)	X(7)
Matlab	275.77	-32.21	0	-3.36	0	-1.00	0	-0.25
DCT_ASR	274	-33	0	-3.50	0	-1	0	-0.50
binDCT	275.77	-32.62	0	-2.40	0	-1.24	0	0

Table 2

The result can also be plotted as a bar graph:

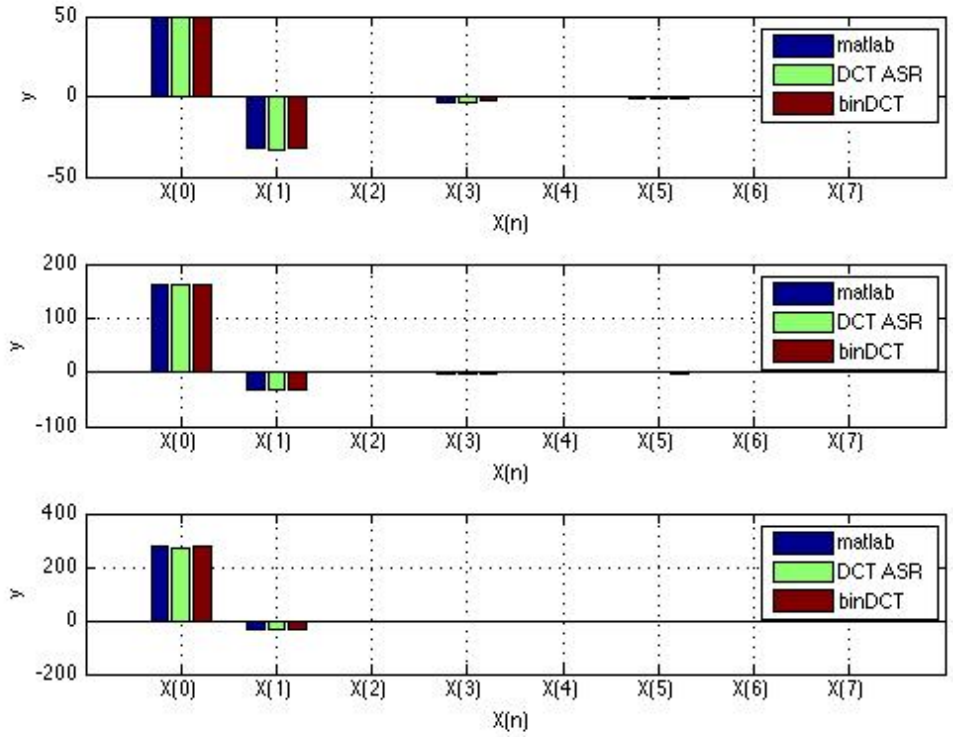


Figure 21

It can be seen that the results of both designs are very close to the value provided by Matlab. Therefore, the correctness of the implementation of DCT_ASR and binDCT is verified.

5 Synthesis Result

Before synthesis, we should specify the FPGA board we are going to use. Only when we use same device for both designs, the result can be comparable with each other. In this case, target the device as below:

Property Name	Value
Evaluation Development Board	Virtex 6 ML605 Evaluation Platform
Product Category	All
Family	Virtex6
Device	XC6VLX75T
Package	FF484
Speed	-1
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store all values
Manual Compile Order	☐
VHDL Source Analysis Standard	VHDL-93

Figure 22

Then, by doing the synthesis and place and route, we can get corresponding reports for

each design. All these reports are provided in Appendix Ⅱ.

After that, according to the full reports of delay and resource usage, I summarize the results below:

	Synthesis		**Place and route**	
	Delay (ns)	Resource cost (# of LUTs)	Delay (ns)	Resource cost (# of LUTs)
DCT_ASR	14.828	823	27.880	555
binDCT	18.361	532	28.939	349

Table 3

Because both DCT_ASR and binDCT are all pure logic design, we cannot get the clock information from the report. However, delays from pad to pad are provided as a static timing reports. We can select the largest one as critical path, as it is shown below:

	Source Pad	Destination Pad	De
1	x5<0>	of	27.880
2	x1<1>	of	27.776
3	x1<0>	of	27.725
4	x7<0>	of	27.528
5	x0<0>	of	27.348
6	x6<0>	of	26.981

DCT_ASR

	Source Pad	Destination Pad	Delay
1	x1<0>	of	28.939
2	x1<1>	of	28.933
3	x6<1>	of	28.377
4	x1<0>	y1<10>	28.275
5	x1<1>	y1<10>	28.269
6	x6<0>	of	28.177

binDCT

Figure 23

As I use FPGA as the implementation device and I did not use any on chip registers, the number of LUT units involved in our design will determine the area cost.

From Table3, we can see that for both Synthesis result and Place and Route result, the binDCT reduced the resource usage to a large extent. Meanwhile, the DCT with algorithm strength reduction performs a little bit faster. Eventually, the conclusion can be get that the lifting transformation reduces area cost by avoiding the use of multipliers but adds propagations

Appendix Ⅰ Source code

DCT_test.m:

```
x0=[0,5,10,15,20,25,30,35];
y0=dct(x0);
DCT_ASR_y00=[98, -66, 0, -7, 0,-2, 0, -1];
DCT_ASR_y0=DCT_ASR_y00/2;
bindct_y00=[140, -64, 0, -4, 0, -3, 0, 0];
sc=[sin(pi/4)/2,1/2/sin(7*pi/16),1/2/sin(3*pi/8),1/2/cos(3*pi/16),sin
(pi/4),cos(3*pi/16)/2,sin(3*pi/8)/2,sin(7*pi/16)/2];
bindct_y0=sc.*bindct_y00;
plot_y0=[y0',DCT_ASR_y0',bindct_y0'];
subplot(3,1,1);
b0=bar(plot_y0);
grid on;
ch0 = get(b0,'children');
set(gca,'XTickLabel',{'X(0)','X(1)','X(2)','X(3)','X(4)','X(5)','X(6)
','X(7)'})
legend('matlab','DCT ASR','binDCT');
xlabel('X(n)');
ylabel('y');
x1=[40,45,50,55,60,65,70,75];
y1=dct(x1);
DCT_ASR_y01=[323, -66, 0, -7, 0,-2, 0, -1];
DCT_ASR_y1=DCT_ASR_y01/2;
bindct_y01=[460, -64, 0, -4, 0, -3, 0, 0];
bindct_y1=sc.*bindct_y01;
plot_y1=[y1',DCT_ASR_y1',bindct_y1'];
subplot(3,1,2);
b1=bar(plot_y1);
grid on;
ch1 = get(b1,'children');
set(gca,'XTickLabel',{'X(0)','X(1)','X(2)','X(3)','X(4)','X(5)','X(6)
','X(7)'})
legend('matlab','DCT ASR','binDCT');
xlabel('X(n)');
ylabel('y');
x2=[80,85,90,95,100,105,110,115];
y2=dct(x2);
DCT_ASR_y02=[548, -66, 0, -7, 0,-2, 0, -1];
DCT_ASR_y2=DCT_ASR_y02/2;
bindct_y02=[780, -64, 0, -4, 0, -3, 0, 0];
bindct_y2=sc.*bindct_y02;
plot_y2=[y2',DCT_ASR_y2',bindct_y2'];
subplot(3,1,3);
```

```
b2=bar(plot_y2);
grid on;
ch2 = get(b2,'children');
set(gca,'XTickLabel',{'X(0)','X(1)','X(2)','X(3)','X(4)','X(5)','X(6)
','X(7)'})
legend('matlab','DCT ASR','binDCT');
xlabel('X(n)');
ylabel('y');
```

Verilog codes:

```
`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:      10:04:12 05/05/2014
// Design Name:      DCT_ASR
// Module Name:      C:/Users/gaoqiao/Desktop/ISE_prj/DCT_ASR/test_DCT_ASR.v
// Project Name:     DCT_ASR
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: DCT_ASR
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////

module test_DCT_ASR;

	// Inputs
	reg [7:0] x0;
	reg [7:0] x1;
	reg [7:0] x2;
	reg [7:0] x3;
	reg [7:0] x4;
	reg [7:0] x5;
	reg [7:0] x6;
	reg [7:0] x7;
```

```
// Outputs
wire [10:0] y0;
wire [10:0] y1;
wire [10:0] y2;
wire [10:0] y3;
wire [10:0] y4;
wire [10:0] y5;
wire [10:0] y6;
wire [10:0] y7;
wire of;

// Instantiate the Unit Under Test (UUT)
DCT_ASR uut (
	.y0(y0),
	.y1(y1),
	.y2(y2),
	.y3(y3),
	.y4(y4),
	.y5(y5),
	.y6(y6),
	.y7(y7),
	.of(of),
	.x0(x0),
	.x1(x1),
	.x2(x2),
	.x3(x3),
	.x4(x4),
	.x5(x5),
	.x6(x6),
	.x7(x7)
);

initial begin
	// Initialize Inputs
	x0 = 8'd0;
	x1 = 8'd5;
	x2 = 8'd10;
	x3 = 8'd15;
	x4 = 8'd20;
	x5 = 8'd25;
	x6 = 8'd30;
	x7 = 8'd35;

	// Wait 100 ns for global reset to finish
	#100;
```

```
        // Add stimulus here
        x0 = 8'd40;
        x1 = 8'd45;
        x2 = 8'd50;
        x3 = 8'd55;
        x4 = 8'd60;
        x5 = 8'd65;
        x6 = 8'd70;
        x7 = 8'd75;
      #100;
        x0 = 8'd80;
        x1 = 8'd85;
        x2 = 8'd90;
        x3 = 8'd95;
        x4 = 8'd100;
        x5 = 8'd105;
        x6 = 8'd110;
        x7 = 8'd115;

    end

endmodule

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:      10:10:37 05/05/2014
// Design Name:      binDCT
// Module Name:      C:/Users/gaoqiao/Desktop/ISE_prj/binDCT/test_binDCT.v
// Project Name:     binDCT
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: binDCT
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
```

```
////////////////////////////////////////////////////////////////////////////////

module test_binDCT;

	// Inputs
	reg [7:0] x0;
	reg [7:0] x1;
	reg [7:0] x2;
	reg [7:0] x3;
	reg [7:0] x4;
	reg [7:0] x5;
	reg [7:0] x6;
	reg [7:0] x7;

	// Outputs
	wire [10:0] y0;
	wire [10:0] y1;
	wire [10:0] y2;
	wire [10:0] y3;
	wire [10:0] y4;
	wire [10:0] y5;
	wire [10:0] y6;
	wire [10:0] y7;
	wire of;

	// Instantiate the Unit Under Test (UUT)
	binDCT uut (
		.y0(y0),
		.y1(y1),
		.y2(y2),
		.y3(y3),
		.y4(y4),
		.y5(y5),
		.y6(y6),
		.y7(y7),
		.of(of),
		.x0(x0),
		.x1(x1),
		.x2(x2),
		.x3(x3),
		.x4(x4),
		.x5(x5),
		.x6(x6),
		.x7(x7)
	);
```

```
        initial begin
            // Initialize Inputs
            x0 = 8'd0;
            x1 = 8'd5;
            x2 = 8'd10;
            x3 = 8'd15;
            x4 = 8'd20;
            x5 = 8'd25;
            x6 = 8'd30;
            x7 = 8'd35;

            // Wait 100 ns for global reset to finish
            #100;

            // Add stimulus here
            x0 = 8'd40;
            x1 = 8'd45;
            x2 = 8'd50;
            x3 = 8'd55;
            x4 = 8'd60;
            x5 = 8'd65;
            x6 = 8'd70;
            x7 = 8'd75;
          #100;
            x0 = 8'd80;
            x1 = 8'd85;
            x2 = 8'd90;
            x3 = 8'd95;
            x4 = 8'd100;
            x5 = 8'd105;
            x6 = 8'd110;
            x7 = 8'd115;
        end

endmodule

module ex8_11(out, in);

  input [7:0] in;
  output [10:0] out;

assign out[10:8] =    {in[7],in[7],in[7]};
assign out[7:0]    =    in[7:0];

endmodule
```

```
module halfadder(x,y,s,co);
	input x,y;
	output s,co;

	xor(s,x,y);
	and(co,x,y);
endmodule

module fulladder(s,co,x,y,ci);
	input x,y,ci;
	output s,co;

	wire p,g,o;

	halfadder HA1(x,y,p,g);
	halfadder HA2(p,ci,s,o);
	or(co,o,g);

endmodule

module adder11(s, of, a, b, ci);

    output [10:0] s;
    output       of;
    input [10:0]       a, b;
    input ci;

    wire   c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, co;

    fulladder a0(s[0], c1, a[0], b[0], ci);
    fulladder a1(s[1], c2, a[1], b[1], c1);
    fulladder a2(s[2], c3, a[2], b[2], c2);
    fulladder a3(s[3], c4, a[3], b[3], c3);
    fulladder a4(s[4], c5, a[4], b[4], c4);
    fulladder a5(s[5], c6, a[5], b[5], c5);
    fulladder a6(s[6], c7, a[6], b[6], c6);
    fulladder a7(s[7], c8, a[7], b[7], c7);
    fulladder a8(s[8], c9, a[8], b[8], c8);
    fulladder a9(s[9], c10, a[9], b[9], c9);
    fulladder a10(s[10], co, a[10], b[10], c10);

    xor(of, co, c10);

endmodule

module addsub11(s, of, a, b, sub);
```

```
    output [10:0] s;
    output       of;
    input [10:0]       a, b;
    input         sub;

    wire [10:0]        xb;

    xor x0(xb[0], b[0], sub);
    xor x1(xb[1], b[1], sub);
    xor x2(xb[2], b[2], sub);
    xor x3(xb[3], b[3], sub);
    xor x4(xb[4], b[4], sub);
    xor x5(xb[5], b[5], sub);
    xor x6(xb[6], b[6], sub);
    xor x7(xb[7], b[7], sub);
    xor x8(xb[8], b[8], sub);
    xor x9(xb[9], b[9], sub);
    xor x10(xb[10], b[10], sub);

    adder11 adder(s, of, a, xb, sub);

endmodule

module adder17(s, of, a, b, ci);

    output [16:0] s;
    output       of;
    input [16:0]       a, b;
    input ci;

    wire   c1, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12, c13, c14, c15, c16, co;

    fulladder a0(s[0], c1, a[0], b[0], ci);
    fulladder a1(s[1], c2, a[1], b[1], c1);
    fulladder a2(s[2], c3, a[2], b[2], c2);
    fulladder a3(s[3], c4, a[3], b[3], c3);
    fulladder a4(s[4], c5, a[4], b[4], c4);
    fulladder a5(s[5], c6, a[5], b[5], c5);
    fulladder a6(s[6], c7, a[6], b[6], c6);
    fulladder a7(s[7], c8, a[7], b[7], c7);
    fulladder a8(s[8], c9, a[8], b[8], c8);
    fulladder a9(s[9], c10, a[9], b[9], c9);
    fulladder a10(s[10], c11, a[10], b[10], c10);
    fulladder a11(s[11], c12, a[11], b[11], c11);
    fulladder a12(s[12], c13, a[12], b[12], c12);
```

```
    fulladder a13(s[13], c14, a[13], b[13], c13);
    fulladder a14(s[14], c15, a[14], b[14], c14);
    fulladder a15(s[15], c16, a[15], b[15], c15);
    fulladder a16(s[16], co, a[16], b[16], c16);

    xor(of, co, c16);

endmodule

module addsub17(s, of, a, b, sub);

    output [16:0] s;
    output       of;
    input [16:0]       a, b;
    input         sub;

    wire [16:0]        xb;

    xor x0(xb[0], b[0], sub);
    xor x1(xb[1], b[1], sub);
    xor x2(xb[2], b[2], sub);
    xor x3(xb[3], b[3], sub);
    xor x4(xb[4], b[4], sub);
    xor x5(xb[5], b[5], sub);
    xor x6(xb[6], b[6], sub);
    xor x7(xb[7], b[7], sub);
    xor x8(xb[8], b[8], sub);
    xor x9(xb[9], b[9], sub);
    xor x10(xb[10], b[10], sub);
    xor x11(xb[11], b[11], sub);
    xor x12(xb[12], b[12], sub);
    xor x13(xb[13], b[13], sub);
    xor x14(xb[14], b[14], sub);
    xor x15(xb[15], b[15], sub);
    xor x16(xb[16], b[16], sub);

    adder17 adder(s, of, a, xb, sub);

endmodule

module aminusb1(out,of,in);

input [10:0] in;
output [10:0] out;
output of;
```

```
wire [16:0] in_ex_1,in_ex_2,out_ex;

assign in_ex_1={in[10],in[10],in[10:0],4'b0};
assign in_ex_2={in[10],in[10],in[10],in[10],in[10],in[10:0],1'b0};

addsub17 aminusb1_adder(out_ex, of, in_ex_1, in_ex_2, 1'b0);

assign out=out_ex[16:6];

endmodule

module b1(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,in_ex_3,in_ex_4,out_ex_1,out_ex_2,out_ex;
wire of1,of2,of3;

assign in_ex_1={in[10],in[10:0],5'b0};
assign in_ex_2={in[10],in[10],in[10:0],4'b0};
assign in_ex_3={in[10],in[10],in[10],in[10],in[10:0],2'b0};
assign in_ex_4={in[10],in[10],in[10],in[10],in[10],in[10],in[10:0]};

addsub17 b1_adder1(out_ex_1, of1, in_ex_1, in_ex_2, 1'b0);
addsub17 b1_adder2(out_ex_2, of2, in_ex_3, in_ex_4, 1'b0);
addsub17 b1_adder3(out_ex, of3, out_ex_1, out_ex_2, 1'b0);

or(of,of1,of2,of3);

assign out=out_ex[16:6];

endmodule

module aplusb1(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,in_ex_3,in_ex_4,out_ex_1,out_ex_2,out_ex;
wire of1,of2,of3;

assign in_ex_1={in[10:0],6'b0};
assign in_ex_2={in[10],in[10],in[10:0],4'b0};
```

```
assign in_ex_3={in[10],in[10],in[10],in[10:0],3'b0};
assign in_ex_4={in[10],in[10],in[10],in[10],in[10],in[10],in[10:0]};

addsub17 aplusb1_adder1(out_ex_1, of1, in_ex_1, in_ex_2, 1'b0);
addsub17 aplusb1_adder2(out_ex_2, of2, in_ex_3, in_ex_4, 1'b0);
addsub17 aplusb1_adder3(out_ex, of3, out_ex_1, out_ex_2, 1'b0);

or(of,of1,of2,of3);

assign out=out_ex[16:6];

endmodule

module rot1(out1,out2,of,in1,in2);

input [10:0] in1, in2;
output [10:0] out1, out2;
output of;

wire [10:0] add1,mult1,mult2,mult3;
wire of1,of2,of3,of4,of5,of6;

aminusb1 rot1_aminusb(mult1,of1,in1);
addsub11 rot1_adder1(add1, of2, in1, in2, 1'b0);
b1 rot1_b(mult2,of3,add1);
aplusb1 rot1_aplusb(mult3,of4,in2);
addsub11 rot1_adder2(out2, of5, mult2, mult1, 1'b1);
addsub11 rot1_adder3(out1, of6, mult2, mult3, 1'b1);
or(of,of1,of2,of3,of4,of5,of6);

endmodule

module aminusb2(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,in_ex_3,out_ex_1,out_ex;
wire of1,of2;

assign in_ex_1={in[10],in[10:0],5'b0};
assign in_ex_2={in[10],in[10],in[10:0],4'b0};
assign in_ex_3={in[10],in[10],in[10],in[10],in[10],in[10:0],1'b0};

addsub17 aminusb2_adder1(out_ex_1, of1, in_ex_1, in_ex_2, 1'b0);
```

```
addsub17 aminusb2_adder2(out_ex, of2, in_ex_3, out_ex_1, 1'b0);

or(of,of1,of2);

assign out=out_ex[16:6];

endmodule

module b2(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,out_ex;

assign in_ex_1={in[10:0],6'b0};
assign in_ex_2={in[10],in[10],in[10],in[10],in[10],in[10],in[10:0]};

addsub17 b2_adder1(out_ex, of, in_ex_1, in_ex_2, 1'b1);

assign out=out_ex[16:6];

endmodule

module aplusb2(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,in_ex_3,in_ex_4,out_ex_1,out_ex_2,out_ex;
wire of1,of2,of3;

assign in_ex_1={in[10:0],6'b0};
assign in_ex_2={in[10],in[10],in[10],in[10:0],3'b0};
assign in_ex_3={in[10],in[10],in[10],in[10],in[10],in[10:0],1'b0};
assign in_ex_4={in[10],in[10],in[10],in[10],in[10],in[10],in[10:0]};

addsub17 aplusb1_adder1(out_ex_1, of1, in_ex_1, in_ex_2, 1'b0);
addsub17 aplusb1_adder2(out_ex_2, of2, in_ex_3, in_ex_4, 1'b0);
addsub17 aplusb1_adder3(out_ex, of3, out_ex_1, out_ex_2, 1'b0);

or(of,of1,of2,of3);

assign out=out_ex[16:6];
```

```
endmodule

module rot2(out1,out2,of,in1,in2);

input [10:0] in1, in2;
output [10:0] out1, out2;
output of;

wire [10:0] add1,mult1,mult2,mult3;
wire of1,of2,of3,of4,of5,of6;

aminusb2 rot2_aminusb(mult1,of1,in1);
addsub11 rot2_adder1(add1, of2, in1, in2, 1'b0);
b2 rot2_b(mult2,of3,add1);
aplusb2 rot2_aplusb(mult3,of4,in2);
addsub11 rot2_adder2(out2, of5, mult2, mult1, 1'b1);
addsub11 rot2_adder3(out1, of6, mult2, mult3, 1'b1);
or(of,of1,of2,of3,of4,of5,of6);

endmodule

module aminusb3(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,in_ex_3,out_ex_1,out_ex;
wire of1,of2;

assign in_ex_1={in[10],in[10:0],5'b0};
assign in_ex_2={in[10],in[10],in[10],in[10],in[10],in[10:0],1'b0};
assign in_ex_3={in[10],in[10],in[10],in[10],in[10],in[10],in[10:0]};

addsub17 aminusb3_adder1(out_ex_1, of1, in_ex_1, in_ex_2, 1'b0);
addsub17 aminusb3_adder2(out_ex, of2, in_ex_3, out_ex_1, 1'b0);

or(of,of1,of2);

assign out=out_ex[16:6];

endmodule

module b3(out,of,in);
```

```
input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,out_ex;

assign in_ex_1={in[10],in[10],in[10:0],4'b0};
assign in_ex_2={in[10],in[10],in[10],in[10:0],3'b0};

addsub17 b3_adder1(out_ex, of, in_ex_1, in_ex_2, 1'b0);

assign out=out_ex[16:6];

endmodule

module aplusb3(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,in_ex_3,out_ex_1,out_ex;
wire of1,of2;

assign in_ex_1={in[10:0],6'b0};
assign in_ex_2={in[10],in[10],in[10:0],4'b0};
assign in_ex_3={in[10],in[10],in[10],in[10],in[10:0],2'b0};

addsub17 aplusb3_adder1(out_ex_1, of1, in_ex_1, in_ex_2, 1'b0);
addsub17 aplusb3_adder2(out_ex, of2, in_ex_3, out_ex_1, 1'b0);

or(of,of1,of2);

assign out=out_ex[16:6];

endmodule

module trans(out1,out2,of,in1,in2);

input [10:0] in1, in2;
output [10:0] out1, out2;
output of;

wire of1,of2;

addsub11 trans_adder1(out1, of1, in1, in2, 1'b0);
```

```
addsub11 trans_adder2(out2, of2, in1, in2, 1'b1);
or(of,of1,of2);

endmodule

module mult_c4(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [16:0] in_ex_1,in_ex_2,in_ex_3,in_ex_4,out_ex_1,out_ex_2,out_ex;
wire of1,of2;

assign in_ex_1={in[10],in[10:0],5'b0};
assign in_ex_2={in[10],in[10],in[10],in[10:0],3'b0};
assign out_ex_2={out_ex_1[16],out_ex_1[16],out_ex_1[16],out_ex_1[16:3]};

addsub17 c4_adder1(out_ex_1, of1, in_ex_1, in_ex_2, 1'b0);
addsub17 c4_adder3(out_ex, of2, out_ex_1, out_ex_2, 1'b0);

or(of,of1,of2);

assign out=out_ex[16:6];

endmodule

module DCT_ASR(y0,y1,y2,y3,y4,y5,y6,y7,of,x0,x1,x2,x3,x4,x5,x6,x7);

input [7:0] x0,x1,x2,x3,x4,x5,x6,x7;
output [10:0] y0,y1,y2,y3,y4,y5,y6,y7;
output of;

wire [10:0] x_ex0,x_ex1,x_ex2,x_ex3,x_ex4,x_ex5,x_ex6,x_ex7;
wire [10:0] line1,line2,line3,line4,line5,line6,line7,line8,line9,line10,line11,line12,line13
                ,line14,line15,line16,line17,line18,line19,line20,line21,line22,line23,line24;
wire of1,of2,of3,of4,of5,of6,of7,of8,of9,of10,of11,of12,of13,of14,of15,of16,of17,of18,of19,of20;

ex8_11 ex1(x_ex0, x0);
ex8_11 ex2(x_ex1, x1);
ex8_11 ex3(x_ex2, x2);
ex8_11 ex4(x_ex3, x3);
ex8_11 ex5(x_ex4, x4);
ex8_11 ex6(x_ex5, x5);
ex8_11 ex7(x_ex6, x6);
ex8_11 ex8(x_ex7, x7);
```

```
rot1 r1(line9,line10,of1,line2,line4);
rot2 r2(line11,line12,of2,line6,line8);
rot3 r3(y6,y2,of3,line14,line16);

trans tr1(line1,line2,of4,x_ex0,x_ex7);
trans tr2(line3,line4,of5,x_ex3,x_ex4);
trans tr3(line5,line6,of6,x_ex1,x_ex6);
trans tr4(line7,line8,of7,x_ex2,x_ex5);
trans tr5(line13,line14,of8,line1,line3);
trans tr6(line15,line16,of9,line5,line7);
trans tr7(line17,line18,of10,line9,line10);
trans tr8(line19,line20,of11,line11,line12);
trans tr9(line21,line22,of12,line13,line15);

addsub11 ad1(y3, of13, line9, line12, 1'b1);
addsub11 ad2(y5, of14, line10, line11, 1'b1);
addsub11 ad3(line23, of15, line17, line19, 1'b0);
addsub11 ad4(line24, of16, line18, line20, 1'b1);

mult_c4 c4_1(y1,of17,line23);
mult_c4 c4_2(y7,of18,line24);
mult_c4 c4_3(y0,of19,line21);
mult_c4 c4_4(y4,of20,line22);

or(of,of1,of2,of3,of4,of5,of6,of7,of8,of9,of10,of11,of12,of13,of14,of15,of16,of17,of18,of19,of20
);

endmodule

module p1(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [10:0] in_ex_1,in_ex_2;

assign in_ex_1={in[10],in[10:1]};
assign in_ex_2={in[10],in[10],in[10],in[10:3]};

addsub11 p1_adder1(out, of, in_ex_1, in_ex_2, 1'b1);

endmodule

module p2(out,of,in);
```

```
input [10:0] in;
output [10:0] out;
output of;

wire [10:0] in_ex_1,in_ex_2;

assign in_ex_1=in[10:0];
assign in_ex_2={in[10],in[10],in[10],in[10:3]};

addsub11 p2_adder1(out, of, in_ex_1, in_ex_2, 1'b1);

endmodule

module u2(out,in);

input [10:0] in;
output [10:0] out;

assign out={in[10],in[10:1]};

endmodule

module p3(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [10:0] in_ex_1,in_ex_2;

assign in_ex_1={in[10],in[10],in[10:2]};
assign in_ex_2={in[10],in[10],in[10],in[10],in[10:4]};

addsub11 p3_adder1(out, of, in_ex_1, in_ex_2, 1'b1);

endmodule

module p4(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [10:0] in_ex_1,in_ex_2;
```

```
assign in_ex_1={in[10],in[10:1]};
assign in_ex_2={in[10],in[10],in[10],in[10],in[10:4]};

addsub11 p4_adder1(out, of, in_ex_1, in_ex_2, 1'b1);

endmodule

module u4(out,of,in);

input [10:0] in;
output [10:0] out;
output of;

wire [10:0] in_ex_1,in_ex_2,in_ex_3,out_ex;
wire of1,of2;

assign in_ex_1={in[10],in[10:1]};
assign in_ex_2={in[10],in[10],in[10],in[10:3]};
assign in_ex_3={in[10],in[10],in[10],in[10],in[10:4]};

addsub11 u4_adder1(out_ex, of1, in_ex_1, in_ex_2, 1'b0);
addsub11 u4_adder2(out, of2, in_ex_3, out_ex, 1'b0);

or(of,of1,of2);

endmodule
```

Appendix Ⅱ Reports

DCT_ASR:

Synthesis Report:

```
Release 14.7 - xst P.20131013 (nt64)
Copyright (c) 1995-2013 Xilinx, Inc.  All rights reserved.
--> Parameter TMPDIR set to xst/projnav.tmp


Total REAL time to Xst completion: 0.00 secs
Total CPU time to Xst completion: 0.11 secs

--> Parameter xsthdpdir set to xst


Total REAL time to Xst completion: 0.00 secs
```

Total CPU time to Xst completion: 0.11 secs

--> Reading design: DCT_ASR.prj

TABLE OF CONTENTS


```
=========================================================================
*                     Synthesis Options Summary                         *
=========================================================================
---- Source Parameters
Input File Name                    : "DCT_ASR.prj"
Ignore Synthesis Constraint File   : NO

---- Target Parameters
Output File Name                   : "DCT_ASR"
Output Format                      : NGC
Target Device                      : xc6vlx75t-1-ff484

---- Source Options
Top Module Name                    : DCT_ASR
Automatic FSM Extraction           : YES
FSM Encoding Algorithm             : Auto
Safe Implementation                : No
FSM Style                          : LUT
RAM Extraction                     : Yes
RAM Style                          : Auto
```

ROM Extraction : Yes
Shift Register Extraction : YES
ROM Style : Auto
Resource Sharing : YES
Asynchronous To Synchronous : NO
Shift Register Minimum Size : 2
Use DSP Block : Auto
Automatic Register Balancing : No

---- Target Options
LUT Combining : Auto
Reduce Control Sets : Auto
Add IO Buffers : YES
Global Maximum Fanout : 100000
Add Generic Clock Buffer(BUFG) : 32
Register Duplication : YES
Optimize Instantiated Primitives : NO
Use Clock Enable : Auto
Use Synchronous Set : Auto
Use Synchronous Reset : Auto
Pack IO Registers into IOBs : Auto
Equivalent register Removal : YES

---- General Options
Optimization Goal : Speed
Optimization Effort : 1
Power Reduction : NO
Keep Hierarchy : No
Netlist Hierarchy : As_Optimized
RTL Output : Yes
Global Optimization : AllClockNets
Read Cores : YES
Write Timing Constraints : NO
Cross Clock Analysis : NO
Hierarchy Separator : /
Bus Delimiter : <>
Case Specifier : Maintain
Slice Utilization Ratio : 100
BRAM Utilization Ratio : 100
DSP48 Utilization Ratio : 100
Auto BRAM Packing : NO
Slice Utilization Ratio Delta : 5

==========

```
=========================================================================
*                          HDL Parsing                                  *
=========================================================================
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\halfadder.v" into library
work
Parsing module <halfadder>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\fulladder.v" into library work
Parsing module <fulladder>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\adder17.v" into library work
Parsing module <adder17>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\addsub17.v" into library
work
Parsing module <addsub17>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\adder11.v" into library work
Parsing module <adder11>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\b3.v" into library work
Parsing module <b3>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\b2.v" into library work
Parsing module <b2>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\b1.v" into library work
Parsing module <b1>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aplusb3.v" into library work
Parsing module <aplusb3>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aplusb2.v" into library work
Parsing module <aplusb2>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aplusb1.v" into library work
Parsing module <aplusb1>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aminusb3.v" into library
work
Parsing module <aminusb3>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aminusb2.v" into library
work
Parsing module <aminusb2>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aminusb1.v" into library
work
Parsing module <aminusb1>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\addsub11.v" into library
work
Parsing module <addsub11>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\trans.v" into library work
Parsing module <trans>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\rot3.v" into library work
Parsing module <rot3>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\rot2.v" into library work
Parsing module <rot2>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\rot1.v" into library work
```

Parsing module <rot1>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\mult_c4.v" into library work
Parsing module <mult_c4>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\ex8_11.v" into library work
Parsing module <ex8_11>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\DCT_ASR.v" into library work
Parsing module <DCT_ASR>.

===
*   HDL Elaboration   *
===

Elaborating module <DCT_ASR>.

Elaborating module <ex8_11>.

Elaborating module <rot1>.

Elaborating module <aminusb1>.

Elaborating module <addsub17>.

Elaborating module <adder17>.

Elaborating module <fulladder>.

Elaborating module <halfadder>.

Elaborating module <addsub11>.

Elaborating module <adder11>.

Elaborating module <b1>.

Elaborating module <aplusb1>.

Elaborating module <rot2>.

Elaborating module <aminusb2>.

Elaborating module <b2>.

Elaborating module <aplusb2>.

Elaborating module <rot3>.

```
Elaborating module <aminusb3>.

Elaborating module <b3>.

Elaborating module <aplusb3>.

Elaborating module <trans>.

Elaborating module <mult_c4>.

=========================================================================
*                            HDL Synthesis                              *
=========================================================================

Synthesizing Unit <DCT_ASR>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\DCT_ASR.v".
    Summary:
    no macro.
Unit <DCT_ASR> synthesized.

Synthesizing Unit <ex8_11>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\ex8_11.v".
    Summary:
    no macro.
Unit <ex8_11> synthesized.

Synthesizing Unit <rot1>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\rot1.v".
    Summary:
    no macro.
Unit <rot1> synthesized.

Synthesizing Unit <aminusb1>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aminusb1.v".
    Summary:
    no macro.
Unit <aminusb1> synthesized.

Synthesizing Unit <addsub17>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\addsub17.v".
    Summary:
Unit <addsub17> synthesized.

Synthesizing Unit <adder17>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\adder17.v".
    Summary:
```

Unit <adder17> synthesized.

Synthesizing Unit <fulladder>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\fulladder.v".
    Summary:
    no macro.
Unit <fulladder> synthesized.

Synthesizing Unit <halfadder>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\halfadder.v".
    Summary:
Unit <halfadder> synthesized.

Synthesizing Unit <addsub11>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\addsub11.v".
    Summary:
Unit <addsub11> synthesized.

Synthesizing Unit <adder11>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\adder11.v".
    Summary:
Unit <adder11> synthesized.

Synthesizing Unit <b1>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\b1.v".
    Summary:
    no macro.
Unit <b1> synthesized.

Synthesizing Unit <aplusb1>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aplusb1.v".
    Summary:
    no macro.
Unit <aplusb1> synthesized.

Synthesizing Unit <rot2>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\rot2.v".
    Summary:
    no macro.
Unit <rot2> synthesized.

Synthesizing Unit <aminusb2>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aminusb2.v".
    Summary:
    no macro.
Unit <aminusb2> synthesized.

Synthesizing Unit <b2>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\b2.v".
    Summary:
    no macro.
Unit <b2> synthesized.

Synthesizing Unit <aplusb2>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aplusb2.v".
    Summary:
    no macro.
Unit <aplusb2> synthesized.

Synthesizing Unit <rot3>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\rot3.v".
    Summary:
    no macro.
Unit <rot3> synthesized.

Synthesizing Unit <aminusb3>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aminusb3.v".
    Summary:
    no macro.
Unit <aminusb3> synthesized.

Synthesizing Unit <b3>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\b3.v".
    Summary:
    no macro.
Unit <b3> synthesized.

Synthesizing Unit <aplusb3>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\aplusb3.v".
    Summary:
    no macro.
Unit <aplusb3> synthesized.

Synthesizing Unit <trans>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\trans.v".
    Summary:
    no macro.
Unit <trans> synthesized.

Synthesizing Unit <mult_c4>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\DCT_ASR\mult_c4.v".
    Summary:

no macro.
Unit <mult_c4> synthesized.

===
HDL Synthesis Report

Macro Statistics
Xors : 2406
1-bit xor2 : 2406

===

===
* Advanced HDL Synthesis *
===

WARNING:Xst:1290 - Hierarchical block <a0> is unconnected in block <adder>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a1> is unconnected in block <adder>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a2> is unconnected in block <adder>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a3> is unconnected in block <adder>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a4>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA1> is unconnected in block <a0>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a0> is unconnected in block <adder>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a1>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a0> is unconnected in block <adder>.
It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA1> is unconnected in block <a1>.
It will be removed from the design.

WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a1>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a0> is unconnected in block <adder>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a1> is unconnected in block <adder>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a2>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a0> is unconnected in block <adder>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a1> is unconnected in block <adder>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a2> is unconnected in block <adder>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <a3> is unconnected in block <adder>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a4>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
    It will be removed from the design.
WARNING:Xst:1290 - Hierarchical block <HA2> is unconnected in block <a0>.
    It will be removed from the design.

===
Advanced HDL Synthesis Report

Macro Statistics
Xors   : 2406
  1-bit xor2   : 2406

===

===
*   Low Level Synthesis   *
===

Optimizing unit <DCT_ASR> ...

Optimizing unit <addsub11> ...

Optimizing unit <adder11> ...

Optimizing unit <adder17> ...

Mapping all equations...
Building and optimizing final netlist ...
Found area constraint ratio of 100 (+ 5) on block DCT_ASR, actual ratio is 2.

Final Macro Processing ...

===
Final Register Report

Found no macro
===

===
*   Partition Report   *
===

Partition Implementation Status

   No Partitions were found in this design.

===
*   Design Summary   *
===

Top Level Output File Name   : DCT_ASR.ngc

Primitive and Black Box Usage:

BELS   : 841
GND   : 1
LUT2   : 4
LUT3   : 83
LUT4   : 40
LUT5   : 544
LUT6   : 152
MUXCY   : 15
MUXF7   : 1
VCC   : 1
IO Buffers   : 153
IBUF   : 64
OBUF   : 89

Device utilization summary:

Selected Device : 6vlx75tff484-1

Slice Logic Utilization:
  Number of Slice LUTs: 823 out of 46560 1%
    Number used as Logic: 823 out of 46560 1%

Slice Logic Distribution:
  Number of LUT Flip Flop pairs used: 823
    Number with an unused Flip Flop: 823 out of 823 100%
    Number with an unused LUT: 0 out of 823 0%
    Number of fully used LUT-FF pairs: 0 out of 823 0%
    Number of unique control sets: 0

IO Utilization:
  Number of IOs: 153
  Number of bonded IOBs: 153 out of 240 63%

Specific Feature Utilization:

Partition Resource Summary:

  No Partitions were found in this design.

==========
Timing Report

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:

No clock signals found in this design

Asynchronous Control Signals Information:

No asynchronous control signals found in this design

Timing Summary:

Speed Grade: -1

Minimum period: No path found
Minimum input arrival time before clock: No path found
Maximum output required time after clock: No path found
Maximum combinational path delay: 14.828ns

Timing Details:

All values displayed in nanoseconds (ns)

===
Timing constraint: Default path analysis
Total number of paths / destination ports: 1609105530 / 89

Delay: 14.828ns (Levels of Logic = 23)
Source: x0<1> (PAD)
Destination: of (PAD)

Data Path: x0<1> to of

Cell:in->out	fanout	Gate Delay	Net Delay	Logical Name (Net Name)
IBUF:I->O	8	0.003	0.684	x0_1_IBUF (x0_1_IBUF)
LUT4:I0->O	2	0.068	0.497	tr1/trans_adder2/adder/a1/co1 (tr1/trans_adder2/adder/c2)
LUT5:I3->O	5	0.068	0.802	tr1/trans_adder2/adder/a3/HA2/Mxor_s_xo<0>1 (line2<3>)
LUT5:I0->O	7	0.068	0.815	r1/rot1_adder1/adder/a3/HA2/Mxor_s_xo<0>1 (r1/add1<3>)
LUT6:I1->O	3	0.068	0.505	r1/rot1_b/b1_adder2/adder/a3/co1 (r1/rot1_b/b1_adder2/adder/c4)
LUT5:I3->O	1	0.068	0.638	r1/rot1_b/b1_adder2/adder/a5/HA2/Mxor_s_xo<0>1 (r1/rot1_b/b1_adder3/xb<5>)
LUT6:I2->O	6	0.068	0.524	r1/rot1_b/b1_adder3/adder/a5/co1 (r1/rot1_b/b1_adder3/adder/c6)
LUT5:I3->O	3	0.068	0.505	r1/rot1_b/b1_adder3/adder/a7/co1 (r1/rot1_b/b1_adder3/adder/c8)
LUT5:I3->O	3	0.068	0.505	r1/rot1_b/b1_adder3/adder/a9/co1 (r1/rot1_b/b1_adder3/adder/c10)
LUT5:I3->O	5	0.068	0.518	r1/rot1_b/b1_adder3/adder/a11/co1 (r1/rot1_b/b1_adder3/adder/c12)

LUT5:I3->O 5 0.068 0.608 r1/rot1_b/b1_adder3/adder/a13/co1 (r1/rot1_b/b1_adder3/adder/c14)
LUT5:I2->O 4 0.068 0.795 r1/rot1_b/b1_adder3/adder/a15/HA2/Mxor_s_xo<0>1 (r1/mult2<9>)
LUT5:I0->O 6 0.068 0.808 r1/rot1_adder3/adder/a9/HA2/Mxor_s_xo<0>1 (line9<9>)
LUT5:I0->O 2 0.068 0.781 tr7/trans_adder2/adder/a9/HA2/Mxor_s_xo<0>1 (line18<9>)
LUT5:I0->O 2 0.068 0.423 ad4/adder/a9/co1 (ad4/adder/c10)
LUT5:I4->O 10 0.068 0.697 ad4/adder/a10/HA2/Mxor_s_xo<0>1 (c4_2/c4_adder3/xb<13>)
LUT5:I1->O 2 0.068 0.423 c4_2/c4_adder1/adder/a13/co1 (c4_2/c4_adder1/adder/c14)
LUT3:I2->O 4 0.068 0.658 c4_2/c4_adder1/adder/a14/HA2/Mxor_s_xo<0>1 (c4_2/c4_adder3/xb<11>)
LUT5:I1->O 5 0.068 0.608 c4_2/c4_adder3/adder/a11/co1 (c4_2/c4_adder3/adder/c12)
LUT6:I3->O 2 0.068 0.784 c4_2/c4_adder3/adder/a15/co1 (y7_10_OBUF)
LUT6:I0->O 1 0.068 0.417 out51_SW0 (N2)
LUT6:I5->O 1 0.068 0.399 out51 (of_OBUF)
OBUF:I->O 0.003 of_OBUF (of)

Total 14.828ns (1.434ns logic, 13.394ns route)
(9.7% logic, 90.3% route)

==========

Cross Clock Domains Report:

==========

Total REAL time to Xst completion: 13.00 secs
Total CPU time to Xst completion: 12.79 secs

-->

Total memory usage is 283980 kilobytes

Number of errors : 0 (0 filtered)
Number of warnings : 28 (0 filtered)
Number of infos : 0 (0 filtered)

Place and route report

Release 14.7 par P.20131013 (nt64)

375A:: Mon May 05 10:07:11 2014

par -w -intstyle ise -ol high -mt off DCT_ASR_map.ncd DCT_ASR.ncd DCT_ASR.pcf

Constraints file: DCT_ASR.pcf.
Loading device for application Rf_Device from file '6vlx75t.nph' in environment C:\Xilinx\14.7\ISE_DS\ISE\.
  "DCT_ASR" is an NCD, version 3.2, device xc6vlx75t, package ff484, speed -1

Initializing temperature to 85.000 Celsius. (default - Range: 0.000 to 85.000 Celsius)
Initializing voltage to 0.950 Volts. (default - Range: 0.950 to 1.050 Volts)

INFO:Par:282 - No user timing constraints were detected or you have set the option to ignore timing constraints ("par
  -x"). Place and Route will run in "Performance Evaluation Mode" to automatically improve the performance of all
  internal clocks in this design. Because there are not defined timing requirements, a timing score will not be
  reported in the PAR report in this mode. The PAR timing summary will list the performance achieved for each clock.
  Note: For the fastest runtime, set the effort level to "std". For best performance, set the effort level to "high".

Device speed data version: "PRODUCTION 1.17 2013-10-13".

Device Utilization Summary:

Slice Logic Utilization:

Number of Slice Registers:	0 out of	93,120	0%
Number of Slice LUTs:	555 out of	46,560	1%
Number used as logic:	555 out of	46,560	1%
Number using O6 output only:	280		
Number using O5 output only:	0		
Number using O5 and O6:	275		
Number used as ROM:	0		
Number used as Memory:	0 out of	16,720	0%
Number used exclusively as route-thrus:	0		

Slice Logic Distribution:

Number of occupied Slices:	295 out of	11,640	2%

Number of LUT Flip Flop pairs used:	555			
Number with an unused Flip Flop:	555	out of	555	100%
Number with an unused LUT:	0	out of	555	0%
Number of fully used LUT-FF pairs:	0	out of	555	0%
Number of slice register sites lost to control set restrictions:	0	out of	93,120	0%

A LUT Flip Flop pair for this architecture represents one LUT paired with one Flip Flop within a slice. A control set is a unique combination of clock, reset, set, and enable signals for a registered element.
The Slice Logic Distribution report is not meaningful if the design is over-mapped for a non-slice resource or if Placement fails.
OVERMAPPING of BRAM resources should be ignored if the design is over-mapped for a non-BRAM resource or if placement fails.

IO Utilization:

Number of bonded IOBs:	153	out of	240	63%

Specific Feature Utilization:

Number of RAMB36E1/FIFO36E1s:	0	out of	156	0%
Number of RAMB18E1/FIFO18E1s:	0	out of	312	0%
Number of BUFG/BUFGCTRLs:	0	out of	32	0%
Number of ILOGICE1/ISERDESE1s:	0	out of	360	0%
Number of OLOGICE1/OSERDESE1s:	0	out of	360	0%
Number of BSCANs:	0	out of	4	0%
Number of BUFHCEs:	0	out of	72	0%
Number of BUFIODQSs:	0	out of	36	0%
Number of BUFRs:	0	out of	18	0%
Number of CAPTUREs:	0	out of	1	0%
Number of DSP48E1s:	0	out of	288	0%
Number of EFUSE_USRs:	0	out of	1	0%
Number of FRAME_ECCs:	0	out of	1	0%
Number of GTXE1s:	0	out of	8	0%
Number of IBUFDS_GTXE1s:	0	out of	6	0%
Number of ICAPs:	0	out of	2	0%
Number of IDELAYCTRLs:	0	out of	9	0%
Number of IODELAYE1s:	0	out of	360	0%
Number of MMCM_ADVs:	0	out of	6	0%
Number of PCIE_2_0s:	0	out of	1	0%
Number of STARTUPs:	1	out of	1	100%
Number of SYSMONs:	0	out of	1	0%
Number of TEMAC_SINGLEs:	0	out of	4	0%

Overall effort level (-ol): High
Router effort level (-rl): High

Starting initial Timing Analysis. REAL time: 9 secs
Finished initial Timing Analysis. REAL time: 9 secs

Starting Router

Phase 1 : 5366 unrouted; REAL time: 9 secs

Phase 2 : 4948 unrouted; REAL time: 10 secs

Phase 3 : 1404 unrouted; REAL time: 26 secs

Phase 4 : 1404 unrouted; (Par is working to improve performance) REAL time: 29 secs

Updating file: DCT_ASR.ncd with current fully routed design.

Phase 5 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 6 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 7 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 8 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 9 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 10 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs
Total REAL time to Router completion: 30 secs
Total CPU time to Router completion: 29 secs

Partition Implementation Status

No Partitions were found in this design.

Generating "PAR" statistics.
INFO:Par:459 - The Clock Report is not displayed in the non timing-driven mode.
Timing Score: 0 (Setup: 0, Hold: 0)

Generating Pad Report.

All signals are completely routed.

Total REAL time to PAR completion: 30 secs
Total CPU time to PAR completion: 29 secs

Peak Memory Usage: 486 MB

Placer: Placement generated during map.
Routing: Completed - No errors found.

Number of error messages: 0
Number of warning messages: 0
Number of info messages: 2

Writing design to file DCT_ASR.ncd

PAR done!

Static timing report

binDCT:

Synthesis report
Release 14.7 - xst P.20131013 (nt64)

--> Parameter TMPDIR set to xst/projnav.tmp

Total REAL time to Xst completion: 0.00 secs
Total CPU time to Xst completion: 0.09 secs

--> Parameter xsthdpdir set to xst

Total REAL time to Xst completion: 0.00 secs
Total CPU time to Xst completion: 0.09 secs

--> Reading design: binDCT.prj

TABLE OF CONTENTS

```
=========================================================================
*                      Synthesis Options Summary                        *
=========================================================================
---- Source Parameters
Input File Name                    : "binDCT.prj"
Ignore Synthesis Constraint File   : NO

---- Target Parameters
Output File Name                   : "binDCT"
Output Format                      : NGC
Target Device                      : xc6vlx75t-1-ff484

---- Source Options
Top Module Name                    : binDCT
Automatic FSM Extraction           : YES
FSM Encoding Algorithm             : Auto
Safe Implementation                : No
FSM Style                          : LUT
RAM Extraction                     : Yes
RAM Style                          : Auto
ROM Extraction                     : Yes
Shift Register Extraction          : YES
ROM Style                          : Auto
Resource Sharing                   : YES
Asynchronous To Synchronous        : NO
Shift Register Minimum Size        : 2
Use DSP Block                      : Auto
Automatic Register Balancing       : No
```

---- Target Options
LUT Combining : Auto
Reduce Control Sets : Auto
Add IO Buffers : YES
Global Maximum Fanout : 100000
Add Generic Clock Buffer(BUFG) : 32
Register Duplication : YES
Optimize Instantiated Primitives : NO
Use Clock Enable : Auto
Use Synchronous Set : Auto
Use Synchronous Reset : Auto
Pack IO Registers into IOBs : Auto
Equivalent register Removal : YES

---- General Options
Optimization Goal : Speed
Optimization Effort : 1
Power Reduction : NO
Keep Hierarchy : No
Netlist Hierarchy : As_Optimized
RTL Output : Yes
Global Optimization : AllClockNets
Read Cores : YES
Write Timing Constraints : NO
Cross Clock Analysis : NO
Hierarchy Separator : /
Bus Delimiter : <>
Case Specifier : Maintain
Slice Utilization Ratio : 100
BRAM Utilization Ratio : 100
DSP48 Utilization Ratio : 100
Auto BRAM Packing : NO
Slice Utilization Ratio Delta : 5

===

===
* HDL Parsing *
===
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\halfadder.v" into library work
Parsing module <halfadder>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\fulladder.v" into library work
Parsing module <fulladder>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\adder11.v" into library work
Parsing module <adder11>.

Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\addsub11.v" into library work
Parsing module <addsub11>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\u4.v" into library work
Parsing module <u4>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\u2.v" into library work
Parsing module <u2>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\trans.v" into library work
Parsing module <trans>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\p4.v" into library work
Parsing module <p4>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\p3.v" into library work
Parsing module <p3>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\p2.v" into library work
Parsing module <p2>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\p1.v" into library work
Parsing module <p1>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\ex8_11.v" into library work
Parsing module <ex8_11>.
Analyzing Verilog file "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\binDCT.v" into library work
Parsing module <binDCT>.

===
*   HDL Elaboration   *
===

Elaborating module <binDCT>.

Elaborating module <ex8_11>.

Elaborating module <trans>.

Elaborating module <addsub11>.

Elaborating module <adder11>.

Elaborating module <fulladder>.

Elaborating module <halfadder>.

Elaborating module <p1>.

Elaborating module <p2>.

Elaborating module <u2>.

Elaborating module <p3>.

```
Elaborating module <p4>.

Elaborating module <u4>.

=========================================================================
*                            HDL Synthesis                              *
=========================================================================

Synthesizing Unit <binDCT>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\binDCT.v".
    Summary:
    no macro.
Unit <binDCT> synthesized.

Synthesizing Unit <ex8_11>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\ex8_11.v".
    Summary:
    no macro.
Unit <ex8_11> synthesized.

Synthesizing Unit <trans>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\trans.v".
    Summary:
    no macro.
Unit <trans> synthesized.

Synthesizing Unit <addsub11>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\addsub11.v".
    Summary:
Unit <addsub11> synthesized.

Synthesizing Unit <adder11>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\adder11.v".
    Summary:
Unit <adder11> synthesized.

Synthesizing Unit <fulladder>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\fulladder.v".
    Summary:
    no macro.
Unit <fulladder> synthesized.

Synthesizing Unit <halfadder>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\halfadder.v".
    Summary:
```

Unit <halfadder> synthesized.

Synthesizing Unit <p1>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\p1.v".
WARNING:Xst:647 - Input <in<0:0>> is never used. This port will be preserved and left unconnected if it belongs to a top-level block or it belongs to a sub-block and the hierarchy of this sub-block is preserved.
    Summary:
    no macro.
Unit <p1> synthesized.

Synthesizing Unit <p2>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\p2.v".
    Summary:
    no macro.
Unit <p2> synthesized.

Synthesizing Unit <u2>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\u2.v".
WARNING:Xst:647 - Input <in<0:0>> is never used. This port will be preserved and left unconnected if it belongs to a top-level block or it belongs to a sub-block and the hierarchy of this sub-block is preserved.
    Summary:
    no macro.
Unit <u2> synthesized.

Synthesizing Unit <p3>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\p3.v".
WARNING:Xst:647 - Input <in<1:0>> is never used. This port will be preserved and left unconnected if it belongs to a top-level block or it belongs to a sub-block and the hierarchy of this sub-block is preserved.
    Summary:
    no macro.
Unit <p3> synthesized.

Synthesizing Unit <p4>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\p4.v".
WARNING:Xst:647 - Input <in<0:0>> is never used. This port will be preserved and left unconnected if it belongs to a top-level block or it belongs to a sub-block and the hierarchy of this sub-block is preserved.
    Summary:
    no macro.
Unit <p4> synthesized.

Synthesizing Unit <u4>.
    Related source file is "C:\Users\gaoqiao\Desktop\ISE_prj\binDCT\u4.v".

WARNING:Xst:647 - Input <in<0:0>> is never used. This port will be preserved and left unconnected if it belongs to a top-level block or it belongs to a sub-block and the hierarchy of this sub-block is preserved.

Summary:

no macro.

Unit <u4> synthesized.

===

HDL Synthesis Report

Macro Statistics

Xors : 1224

1-bit xor2 : 1224

===

===

* Advanced HDL Synthesis *

===

===

Advanced HDL Synthesis Report

Macro Statistics

Xors : 1224

1-bit xor2 : 1224

===

===

* Low Level Synthesis *

===

Optimizing unit <binDCT> ...

Optimizing unit <addsub11> ...

Optimizing unit <adder11> ...

Mapping all equations...

Building and optimizing final netlist ...

Found area constraint ratio of 100 (+ 5) on block binDCT, actual ratio is 1.

Final Macro Processing ...

```
=========================================================================
Final Register Report

Found no macro
=========================================================================

=========================================================================
*                            Partition Report                           *
=========================================================================

Partition Implementation Status
-------------------------------

   No Partitions were found in this design.

-------------------------------

=========================================================================
*                            Design Summary                             *
=========================================================================

Top Level Output File Name         : binDCT.ngc

Primitive and Black Box Usage:
------------------------------
# BELS                             : 544
#      GND                         : 1
#      LUT2                        : 4
#      LUT3                        : 55
#      LUT4                        : 35
#      LUT5                        : 364
#      LUT6                        : 74
#      MUXCY                       : 10
#      VCC                         : 1
# IO Buffers                       : 153
#      IBUF                        : 64
#      OBUF                        : 89

Device utilization summary:
---------------------------

Selected Device : 6vlx75tff484-1

Slice Logic Utilization:
 Number of Slice LUTs:                  532  out of  46560     1%
    Number used as Logic:               532  out of  46560     1%
```

Slice Logic Distribution:
  Number of LUT Flip Flop pairs used: 532
    Number with an unused Flip Flop: 532 out of 532 100%
    Number with an unused LUT: 0 out of 532 0%
    Number of fully used LUT-FF pairs: 0 out of 532 0%
    Number of unique control sets: 0

IO Utilization:
  Number of IOs: 153
  Number of bonded IOBs: 153 out of 240 63%

Specific Feature Utilization:

Partition Resource Summary:

   No Partitions were found in this design.

==========
Timing Report

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:

No clock signals found in this design

Asynchronous Control Signals Information:

No asynchronous control signals found in this design

Timing Summary:

Speed Grade: -1

    Minimum period: No path found
    Minimum input arrival time before clock: No path found
    Maximum output required time after clock: No path found
    Maximum combinational path delay: 18.361ns

Timing Details:

All values displayed in nanoseconds (ns)

==========

Timing constraint: Default path analysis

Total number of paths / destination ports: 11897703864 / 89

Delay: 18.361ns (Levels of Logic = 28)

Source: x1<1> (PAD)

Destination: of (PAD)

Data Path: x1<1> to of

Cell:in->out	fanout	Gate Delay	Net Delay	Logical Name (Net Name)
IBUF:I->O	10	0.003	0.697	x1_1_IBUF (x1_1_IBUF)
LUT4:I0->O	5	0.068	0.518	tr3/trans_adder2/adder/a1/co1 (tr3/trans_adder2/adder/c2)
LUT5:I3->O	4	0.068	0.511	tr3/trans_adder2/adder/a3/co1 (tr3/trans_adder2/adder/c4)
LUT5:I3->O	7	0.068	0.621	tr3/trans_adder2/adder/a5/HA2/Mxor_s_xo<0>1 (line7<5>)
LUT6:I3->O	2	0.068	0.781	p_4/p4_adder1/adder/a1/HA2/Mxor_s_xo<0>1 (line9<1>)
LUT6:I1->O	4	0.068	0.511	ad1/adder/a1/co1 (ad1/adder/c2)
LUT5:I3->O	4	0.068	0.511	ad1/adder/a3/co1 (ad1/adder/c4)
LUT5:I3->O	3	0.068	0.505	ad1/adder/a5/co1 (ad1/adder/c6)
LUT5:I3->O	3	0.068	0.431	ad1/adder/a7/co1 (ad1/adder/c8)
LUT5:I4->O	11	0.068	0.704	ad1/adder/a8/HA2/Mxor_s_xo<0>1 (line10<8>)
LUT5:I1->O	4	0.068	0.511	u_4/u4_adder1/adder/a5/co1 (u_4/u4_adder1/adder/c6)
LUT5:I3->O	2	0.068	0.784	u_4/u4_adder1/adder/a7/HA2/Mxor_s_xo<0>1 (u_4/out_ex<7>)
LUT6:I0->O	2	0.068	0.644	u_4/u4_adder2/adder/a7/HA2/Mxor_s_xo<0>1 (ad2/xb<7>)
LUT5:I1->O	3	0.068	0.505	ad2/adder/a7/co1 (ad2/adder/c8)
LUT5:I3->O	4	0.068	0.795	ad2/adder/a9/co1 (ad2/adder/c10)
LUT5:I0->O	8	0.068	0.684	ad2/adder/a10/HA2/Mxor_s_xo<0>1 (line12<10>)
LUT5:I1->O	2	0.068	0.781	p_5/p1_adder1/adder/a7/HA2/Mxor_s_xo<0>1 (line13<7>)
LUT5:I0->O	4	0.068	0.658	ad3/adder/a7/HA2/Mxor_s_xo<0>1 (line14<7>)
LUT5:I1->O	5	0.068	0.518	tr7/trans_adder1/adder/a7/co1

(tr7/trans_adder1/adder/c8)
LUT5:I3->O 2 0.068 0.781 tr7/trans_adder1/adder/a9/co1
(tr7/trans_adder1/adder/c10)
LUT5:I0->O 2 0.068 0.781
tr7/trans_adder1/adder/a10/HA2/Mxor_s_xo<0>1 (line19<10>)
LUT5:I0->O 4 0.068 0.601 ad8/adder/a10/HA2/Mxor_s_xo<0>1
(y7_10_OBUF)
LUT4:I1->O 2 0.068 0.644
u_3/p3_adder1/adder/a7/HA2/Mxor_s_xo<0>1 (line36<7>)
LUT5:I1->O 4 0.068 0.511 ad9/adder/a7/co1 (ad9/adder/c8)
LUT4:I2->O 1 0.068 0.778 ad9/adder/a9/co11 (ad9/adder/c10)
LUT6:I0->O 1 0.068 0.000 out_wg_lut<13> (out_wg_lut<13>)
MUXCY:S->O 1 0.490 0.399 out_wg_cy<13> (of_OBUF)
OBUF:I->O 0.003 of_OBUF (of)

Total 18.361ns (2.196ns logic, 16.165ns route)
(12.0% logic, 88.0% route)
==========

Cross Clock Domains Report:

==========

Total REAL time to Xst completion: 9.00 secs
Total CPU time to Xst completion: 9.22 secs

-->

Total memory usage is 282956 kilobytes

Number of errors : 0 (0 filtered)
Number of warnings : 5 (0 filtered)
Number of infos : 0 (0 filtered)

Place and Route report
Release 14.7 par P.20131013 (nt64)

375A:: Mon May 05 10:11:57 2014

par -w -intstyle ise -ol high -mt off binDCT_map.ncd binDCT.ncd binDCT.pcf

Constraints file: binDCT.pcf.
Loading device for application Rf_Device from file '6vlx75t.nph' in environment
C:\Xilinx\14.7\ISE_DS\ISE\.

"binDCT" is an NCD, version 3.2, device xc6vlx75t, package ff484, speed -1

Initializing temperature to 85.000 Celsius. (default - Range: 0.000 to 85.000 Celsius)
Initializing voltage to 0.950 Volts. (default - Range: 0.950 to 1.050 Volts)

INFO:Par:282 - No user timing constraints were detected or you have set the option to ignore timing constraints ("par
-x"). Place and Route will run in "Performance Evaluation Mode" to automatically improve the performance of all
internal clocks in this design. Because there are not defined timing requirements, a timing score will not be
reported in the PAR report in this mode. The PAR timing summary will list the performance achieved for each clock.
Note: For the fastest runtime, set the effort level to "std". For best performance, set the effort level to "high".

Device speed data version: "PRODUCTION 1.17 2013-10-13".

Device Utilization Summary:

Slice Logic Utilization:

Number of Slice Registers:	0 out of	93,120	0%
Number of Slice LUTs:	349 out of	46,560	1%
Number used as logic:	349 out of	46,560	1%
Number using O6 output only:	169		
Number using O5 output only:	0		
Number using O5 and O6:	180		
Number used as ROM:	0		
Number used as Memory:	0 out of	16,720	0%
Number used exclusively as route-thrus:	0		

Slice Logic Distribution:

Number of occupied Slices:	155 out of	11,640	1%
Number of LUT Flip Flop pairs used:	349		
Number with an unused Flip Flop:	349 out of	349	100%
Number with an unused LUT:	0 out of	349	0%
Number of fully used LUT-FF pairs:	0 out of	349	0%
Number of slice register sites lost to control set restrictions:	0 out of	93,120	0%

A LUT Flip Flop pair for this architecture represents one LUT paired with
one Flip Flop within a slice. A control set is a unique combination of
clock, reset, set, and enable signals for a registered element.
The Slice Logic Distribution report is not meaningful if the design is
over-mapped for a non-slice resource or if Placement fails.
OVERMAPPING of BRAM resources should be ignored if the design is

over-mapped for a non-BRAM resource or if placement fails.

IO Utilization:

Number of bonded IOBs:	153 out of	240	63%

Specific Feature Utilization:

Number of RAMB36E1/FIFO36E1s:	0 out of	156	0%
Number of RAMB18E1/FIFO18E1s:	0 out of	312	0%
Number of BUFG/BUFGCTRLs:	0 out of	32	0%
Number of ILOGICE1/ISERDESE1s:	0 out of	360	0%
Number of OLOGICE1/OSERDESE1s:	0 out of	360	0%
Number of BSCANs:	0 out of	4	0%
Number of BUFHCEs:	0 out of	72	0%
Number of BUFIODQSs:	0 out of	36	0%
Number of BUFRs:	0 out of	18	0%
Number of CAPTUREs:	0 out of	1	0%
Number of DSP48E1s:	0 out of	288	0%
Number of EFUSE_USRs:	0 out of	1	0%
Number of FRAME_ECCs:	0 out of	1	0%
Number of GTXE1s:	0 out of	8	0%
Number of IBUFDS_GTXE1s:	0 out of	6	0%
Number of ICAPs:	0 out of	2	0%
Number of IDELAYCTRLs:	0 out of	9	0%
Number of IODELAYE1s:	0 out of	360	0%
Number of MMCM_ADVs:	0 out of	6	0%
Number of PCIE_2_0s:	0 out of	1	0%
Number of STARTUPs:	1 out of	1	100%
Number of SYSMONs:	0 out of	1	0%
Number of TEMAC_SINGLEs:	0 out of	4	0%

Overall effort level (-ol): High
Router effort level (-rl): High

Starting initial Timing Analysis. REAL time: 8 secs
Finished initial Timing Analysis. REAL time: 8 secs

Starting Router

Phase 1 : 4196 unrouted; REAL time: 9 secs

Phase 2 : 3874 unrouted; REAL time: 9 secs

Phase 3 : 894 unrouted; REAL time: 27 secs

Phase 4 : 894 unrouted; (Par is working to improve performance) REAL time: 30 secs

Updating file: binDCT.ncd with current fully routed design.

Phase 5 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 6 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 7 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 8 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 9 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs

Phase 10 : 0 unrouted; (Par is working to improve performance) REAL time: 30 secs
Total REAL time to Router completion: 30 secs
Total CPU time to Router completion: 30 secs

Partition Implementation Status

No Partitions were found in this design.

Generating "PAR" statistics.
INFO:Par:459 - The Clock Report is not displayed in the non timing-driven mode.
Timing Score: 0 (Setup: 0, Hold: 0)

Generating Pad Report.

All signals are completely routed.

Total REAL time to PAR completion: 31 secs
Total CPU time to PAR completion: 31 secs

Peak Memory Usage: 484 MB

Placer: Placement generated during map.
Routing: Completed - No errors found.

Number of error messages: 0
Number of warning messages: 0
Number of info messages: 2

Writing design to file binDCT.ncd

PAR done!