Politecnico di Torino

Collegio di Elettronica, Telecomunicazioni e Fisica

Lab1 Design and Implementation of a FIR Filter Integrated Systems Architecture

Master degree in Electrical Engineering

Group: 22

Authors: Chen Xiao, Li Shancheng, Yang Haifeng, Yang Zeyu

Contents

Table 1: coefficient of the filter

b_0	-13
b_1	-28
b_2	104
b_3	544
b_4	830
b_5	544
b_6	104
b_7	-28
b_8	-13

1 Introduction

The goal of this laboratory is to design an FIR filter. First, use Matlab to make a model which is a relatively ideal expectation, and take it as the reference for the later design of the C model and the Very Large Scale Integration Circuit design. The design of the C model is to reduce the bitwidth (and so the area later on) of the filter compared to the Matlab model, the THD is required lower than -30db. Of course, this is not as good as the performance of Matlab's filter but is used to achieve an acceptable filtered output. Furthermore, the main content of the laboratory is the VLSI design, which is to design the hardware description script that matches the ability as the C model does. Finally, the synthesis is performed, comparing the tool between Design Compiler and Cadence, respectively in the area, of power estimation and testing the netlist whose switching activity is generated by these two synthesis tools.

2 Reference model development

Our group number is 22, thus we design the FIR filter, p=1. We have 4 members, reverse-order (alphabetically) the surnames are Yang, Yang, Chen and Li, so x=4 and y=4, obtain the order of the filter (N) and the number of bits (n_b) as follows:

$$N = 2^p \cdot [(x \mod 2) + 1] + 6 \cdot p = 8$$
$$n_b\text{=(y mod 7)+8=12}$$

2.1 Matlab model

The coefficients of the filter have one bit for the integer part and 11 bits for the fractional part. They correspond to the integer values shown in Table 1 . The FIR filter output is the Figure 1, y is the output of Matlab, x1 is 500hz signal, x2 is 3.5khz signal, and x is the mixture of x1 and x2.

2.2 C model and THD

The convolution calculation process of the C model is obtained by truncating the least significant 11-bit in the multiplication, which is the result of the sampled data and coefficient. The Total Harmonics Distortion of C program output shown in Figure 2, the Total Harmonics Distortion is -33 dB, still satisfy the requirement below -30 dB.

2.3 Explanations, comparisons, and comments

As it can be observed in Figure 3, the curve of the C model is different from Matlab, because in the C model we simply shift the result of multiplication right 16 bits and then shift left 5 bits, to ignore

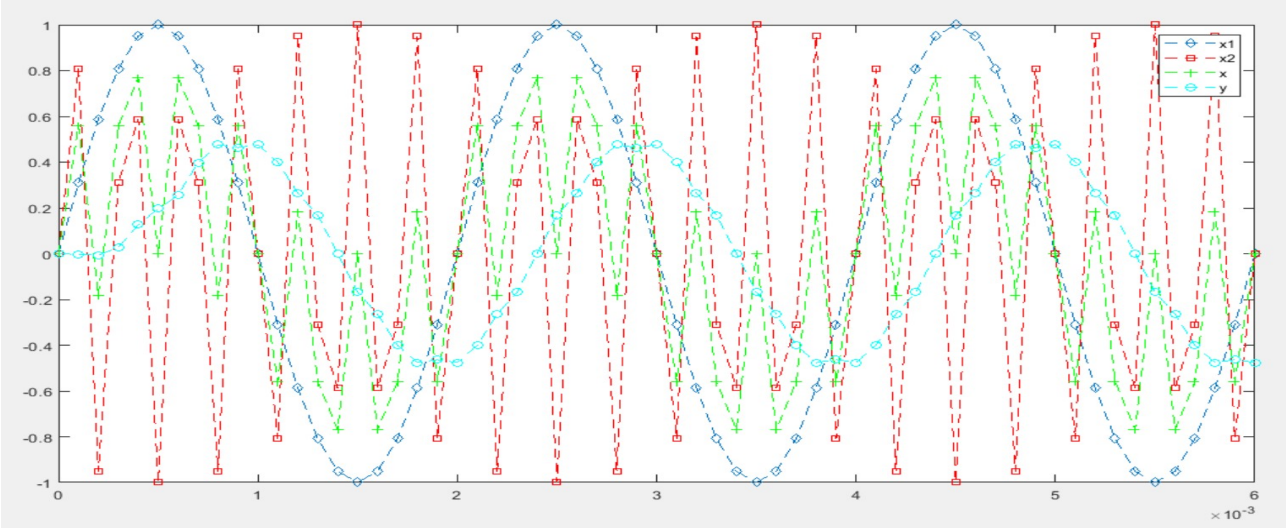

Figure 1: Matlab output

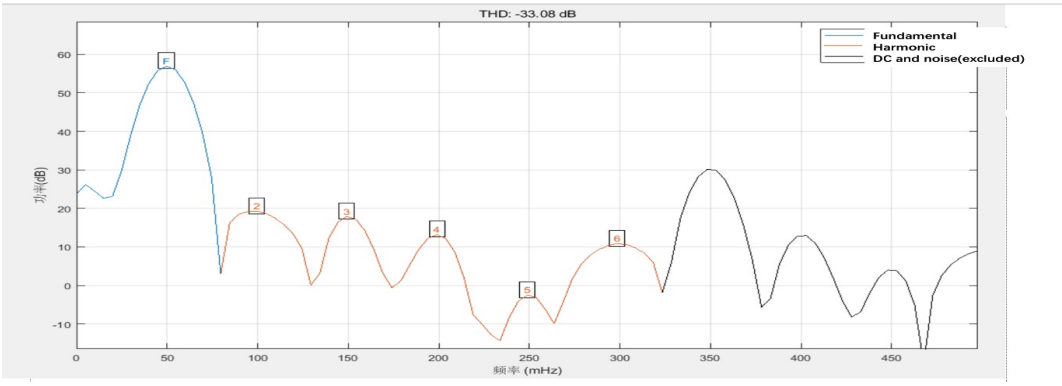

Figure 2: Total Harmonics Distortion of C program output

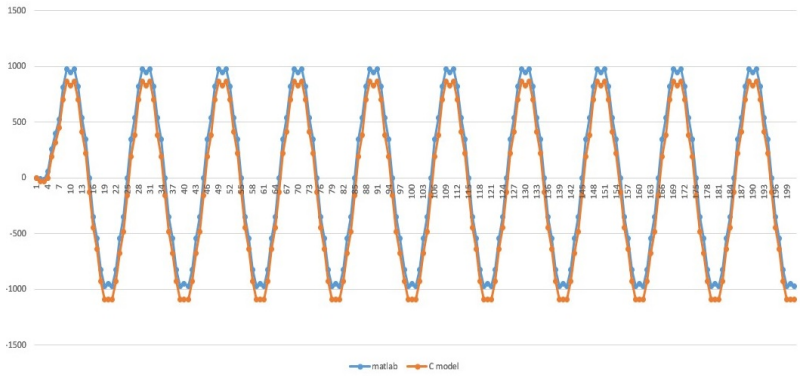

Figure 3: Comparison between Matlab and C model

the least significant bit, thus obtain the area saving for the VLSI design, the THD thus is not good as the Matlab model.

3 VLSI implementation

We have to make the output of the hardware description exactly match every point with respect to the output of the C program, what we did in the design is based on the idea of following the description. The final output of the filter is 12-bit, which is the same as the input data and coefficient, a 12-bit binary number divided by the number of orders plus 1. According to the previous formula, the result needs to be 12 bits wide, but in order to meet the THD, after shifting, the lower 4 bits are all 0. Inconvenience, and in order to calculate the convolution in one clock cycle, at the same time the multiplication has to mask out the bits except for the 8 MSB, we provide the approach with "and" operation, the code as the listing '1' show in the line from 10 to 18. Thus the result of the convolution is a 23-bit binary number, so the last step is to take the 11 MSB of convolution as the final output of the filter, the code as the listing '1' shown in line 19.

It is worth pointing out that, the variable "ss" in the code is a "variable" data type, and the convolution assignment manner is ":=", which means blocking assign, this data is transferred and updated at the current clock cycle, thus the one following code at listing '1' line 19 is able to update the data at the same clock cycle.

下划线说的是12位2进制数除以阶数加一的和，也就是12位数除以9，结果是455，455大于8位二进制数（255），小于9位二进制数（511），应该取8位二进制数，这样9个8位二进制数相加的和是12位（1000，1111，0111）刚好和最终输出一样，如果是9个9位二进制数相加得数是13位（1，0001，1111，0111）就比最终输出的结果位数多了一位。验证过没有and高8位的综合面积比有and高8位的大

Listing 1: State transition

```
1
2  if (CLK'event and CLK = '1') then
3          if VIN='1' then
4              INPUTDATA : for i in 0 to N-1 loop
5              din_buff(0)<=signed(DIN);
6              din_buff(i+1)<=din_buff(i);
7          end loop INPUTDATA;
8          if cnt=1 then
9              VOUT<='1';
10         ss:=std_logic_vector(
11         signed((din_buff(0)*signed(B0)) and X"ff0000")+
12         signed((din_buff(1)*signed(B1)) and X"ff0000")+
13         signed((din_buff(2)*signed(B2)) and X"ff0000")+
14         signed((din_buff(3)*signed(B3)) and X"ff0000")+
15         signed((din_buff(4)*signed(B4)) and X"ff0000")+
16         signed((din_buff(5)*signed(B5)) and X"ff0000")+
17         signed((din_buff(6)*signed(B6)) and X"ff0000")+
18         signed((din_buff(7)*signed(B7)) and X"ff0000")+
19         signed((din_buff(8)*signed(B8)) and X"ff0000"));
20         DOUT(11 downto 0)<=ss(22 downto 11);
21     else
22         cnt<=cnt+1;
23 end if;
```

我画下划线那里应该是说每个被加数的位数是 8 位的意思

As the code 'State transition' in Listing'1' shows, the variable of "cnt" records the clock accessing the process, when the variable is 0 the output of the filter is not ready when the variable is 1 then the filter is ready, means the filter needs 2 cycles to prepare the output in the beginning. The code above also describes the multiplexes behavior, when VIN is 1 then do the shift in process, otherwise do nothing.

3.1 Architecture

The basic filter architecture as Figure 4 shows, with the enable multiplexer at the input, and the the enable multiplexer at the output. When the input signal "VIN" is high the filter can shift the data in the buffer and after the initial first clock proceeds with the convolution, otherwise, the filter is doing nothing, but the data are still saved in the buffer, the buffer is an array of 9 elements with 12 bit. This multiplexer thus supports the function of whenever the "VIN" changes during the process of the filter, the data is saved before the valid signal vanishes, and again when the "VIN" is valid, the shifting process is resumed as before, looks like the "VIN" never changed.

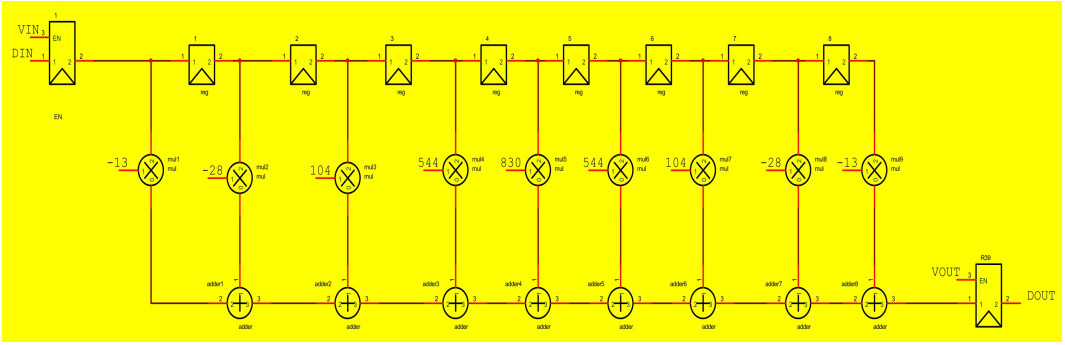

Figure 4: basic architecture

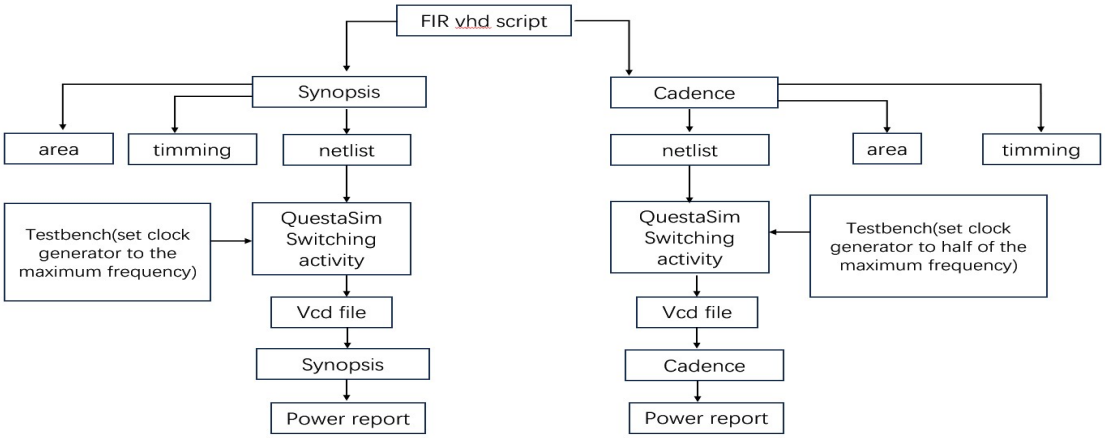

Figure 5: Logic synthesis work flow

3.2 Logic synthesis

Logic synthesis is a process to optimize vhd file code base on the algorithm which the tools have, we need to synthesis our code on Synopsis Design Comipler and Cadence. The work flow of synthesis is show as the Figure 5. The flow diagram is not only for basic architecture but also for advanced FIR architecture later on this report.

Synthesis is first performed on the environment of Synopsys Design Compiler with Nangate45 Cell library.

Without clock gating after trying many times to set the period by sending the command into the shell prompt of Design Compiler, from 10 ns to 3.4 ns until the slack does not match, then stop. Finally got the maximum frequency of 294 Mhz, noted as (F_{max}) (in period is 3.4 ns) we obtained the area is 5934.2 µm2.

Base on the synthesis of Design Compiler without clock gating at the maximum frequency, changed the data maker vhd file, let the tco time to be 0.1 ns, then proceed the switching activity, the simulation does not get error out. At frequency 294 Mhz an estimation of power consumption is described of Table 2.

Without clock gating synthesis, the power estimation is show as Table 3.

With clock gating synthesis, the power estimation is show as Table 4. It is worthy to point out that, the basic version FIR synthesis in Design Compiler with clock gating, simulated output is not

Table 2: Design Compiler basic architecture 294 Mhz estimation of power consumption

$CellInternalPower$	1.6684 mW (54%)
$NetSwitchingPower$	1.3679 mW (46%)
$TotalDynamicPower$	2.9962 mW (100%)
$CellLeakagePower$	130.239 uW

Table 3: Synopsis basic architecture Power frequency=(F_{max}) without clock gating

	Internal	Switching	Leakage	Total
PowerGroup	Power	Power	Power	Power (%)
io_pad	0.0000	0.0000	0.0000	0.0000 (0.00%)
memory	0.0000	0.0000	0.0000	0.0000 (0.00%)
black_box	0.0000	0.0000	0.0000	0.0000 (0.00%)
clock_network	0.0000	0.0000	0.0000	0.0000 (0.00%)
register	510.3693	226.1225	1.3463e+04	749.9546 (23.99%)
sequential	0.0000	0.0000	0.0000	0.0000 (0.00%)
combinational	1.1180e+03	1.1417e+03	1.1678e+05	2.3765e+03 (76.01%)
Total	1.6284e+03 uW	1.3679e+03 uW	1.3024e+05 nW	3.1265e+03 uW

matching every points, the reason still need to find out, but the pipeline version is able to match every points, the pipeline version is discussed later.

Without clock gating set the frequency as $(F_{max})/2 = 147$ Mhz, in period is 6.8 ns we obtained the area is 5877.0 µm2. The area synthesis of the lower frequency is a little smaller than the higher one, may be the reason that the higher frequency needs more space for accommodation.

At frequency 147 Mhz an estimation of power consumption as the description of Table 5, more detail about $(F_{max})/2$ estimation of power show as Table 6.

3.3 Place and Route

Set the clock frequency to maximum frequency (294 Mhz) divided by 2, denoted as $(F_{max})/2$, which means the clock period is 6.8 ns when the valid signal "VIN" is high means the filter is allowed to proceed the calculation.

Changed the file clk_gen.vhd to adjust the frequency of that is half of maximum frequency, the power estimation generated by Cadence about the basic filter architecture shown as the Table 7, and more detail shown as the Table 8, the area data show as the Table 9 .

Table 4: Synopsis basic architecture Power frequency=(F_{max}) with clock gating

	Internal	Switching	Leakage	Total
PowerGroup	Power	Power	Power	Power (%)
io_pad	0.0000	0.0000	0.0000	0.0000 (0.00%)
memory	0.0000	0.0000	0.0000	0.0000 (0.00%)
black_box	0.0000	0.0000	0.0000	0.0000 (0.00%)
clock_network	0.0000	0.0000	0.0000	0.0000 (0.00%)
register	508.5216	242.5085	1.3296e+04	764.3264 (24.42%)
sequential	0.0000	0.0000	0.0000	0.0000 (0.00%)
combinational	1.1189e+03	1.1285e+03	1.1813e+05	2.3656e+03 (75.58%)
Total	1.6275e+03 uW	1.3710e+03 uW	1.3143e+05 nW	3.1299e+03 uW

Table 5: Design Compiler basic architecture 147 Mhz estimation of power consumption

$CellInternalPower$	447.5643 uW (51%)
$NetSwitchingPower$	425.3140 uW (49%)
$TotalDynamicPower$	872.8784 uW (100%)
$CellLeakagePower$	120.1777 uW

Table 6: Synopsis basic architecture total Power with the frequency of $(F_{max})/2$

	Internal	Switching	Leakage	Total
PowerGroup	Power	Power	Power	Power (%) Attrs
io_pad	0.0000	0.0000	0.0000	0.0000 (0.00%)
memory	0.0000	0.0000	0.0000	0.0000 (0.00%)
black_box	0.0000	0.0000	0.0000	0.0000 (0.00%)
clock_network	4.8428	21.1097	178.2704	26.1308 (2.63%)
register	92.8125	72.5026	1.2852e+04	178.1674 (17.94%)
sequential	0.0000	0.0000	0.0000	0.0000 (0.00%)
combinational	349.9091	331.7007	1.0715e+05	788.7576 (79.43%)
Total	447.5644 uW	425.3131 uW	1.2018e+05 nW	993.0558 uW

Table 7: Cadence basic architecture total Power with the frequency of $(F_{max})/2$ (Power Units = 1mW)

$TotalInternalPower$:	0.57381256	49.0314%
$TotalSwitchingPower$:	0.46911506	40.0852%
$TotalLeakagePower$:	0.12736805	10.8834%
$TotalPower$:	1.17029567	

Table 8: Cadence basic architecture total Power with the frequency of $(F_{max})/2$ detail (Power Units = 1mW)

Group	Internal	Switching	Leakage	Total	Percentage
	Power	Power	Power	Power	(%)
Sequential	0.1964	0.07873	0.0133	0.2884	24.65
Macro	0	0	0	0	0
IO	0	0	0	0	0
Combinational	0.3721	0.349	0.114	0.835	71.35
Clock(*Combinational*)	0.005325	0.04144	9.167e-05	0.04685	4.003
Clock(*Sequential*)	0	0	0	0	0
Total	0.5738	0.4691	0.1274	1.17	100

Table 9: Cadence basic architecture area with the frequency of $(F_{max})/2$

Gate area $0.7980um^2$			
Level 0 *Modulemyfir*	Gates= 7698	Cells= 3089	Area= 6143.5 um^2

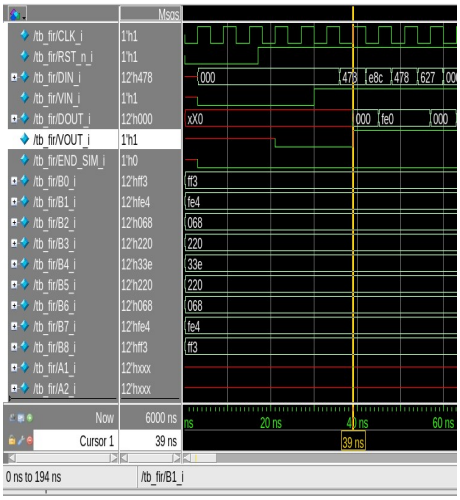

Figure 6: Basic architecture Cadence post route and place simulation start

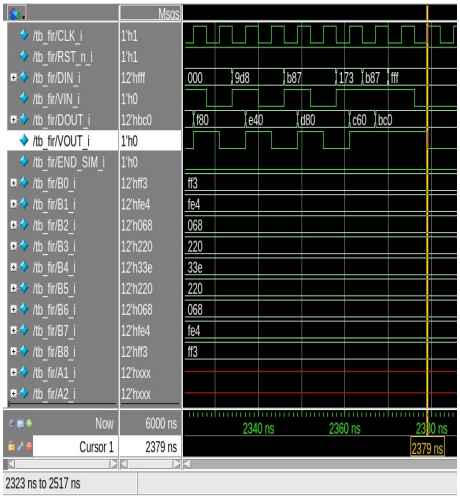

Figure 7: Basic architecture Cadence post route and place simulation end

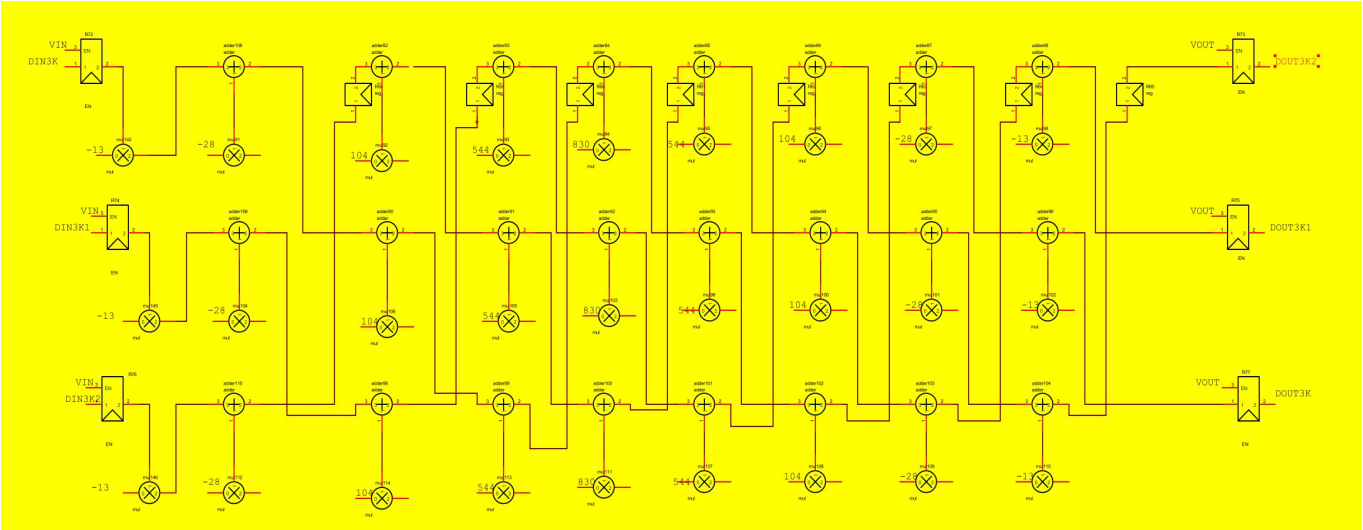

Figure 8: Formal unfolded architecture

4 Advanced architecture development

Advanced architecture design follows the rule of converting the single input and single output processing to multiple input and multiple outputs, also called parallel processing, but this parallel processing has the same critical path as before only throughput is increased, and then based on this parallel processing architecture to inert the register in order to reduce the critical path. According to these two formulas:

$$\text{j} = \text{mod}\ (\frac{i+w}{P})$$
$$\text{wi}=[\ \frac{i+w}{P}\ ]$$

Where j is the destination node, "i" is the source node (in our case is 0, 1, 2), "P" is the order of unfolding, which is 3, "w" is the number of registers in the original DFG arch, "wi" is the integer part of the expression, we can calculate the destination nodes of the arch which the source node should connect to and the number of registers along the arch.

Based on the rule I stated before, the architecture is designed like the Figure 8 shows.

4.1 Architecture

Modified a little from the formal unfolded architecture, make it more clear, as Figure 9 shows. The advanced architecture is to unfold the basic architecture to increase the throughput by 3 times, the process is split into 3 streams, and after the calculation and schematic design, eventually insert 8 registers, 2 of them located in the first line, 3 registers located in the second line and the remaining 3 registers locates in the third line And the input-output valid multiplexers are also triplicated. The personal summary may not be so universal, but it is easier for me personally to understand this 3-degree unfolded parallel processing architecture, is each data stream from left to right there are 2 streams that come from its two neighbors, and then one data stream from itself, and so on feeding the 8 multipliers.

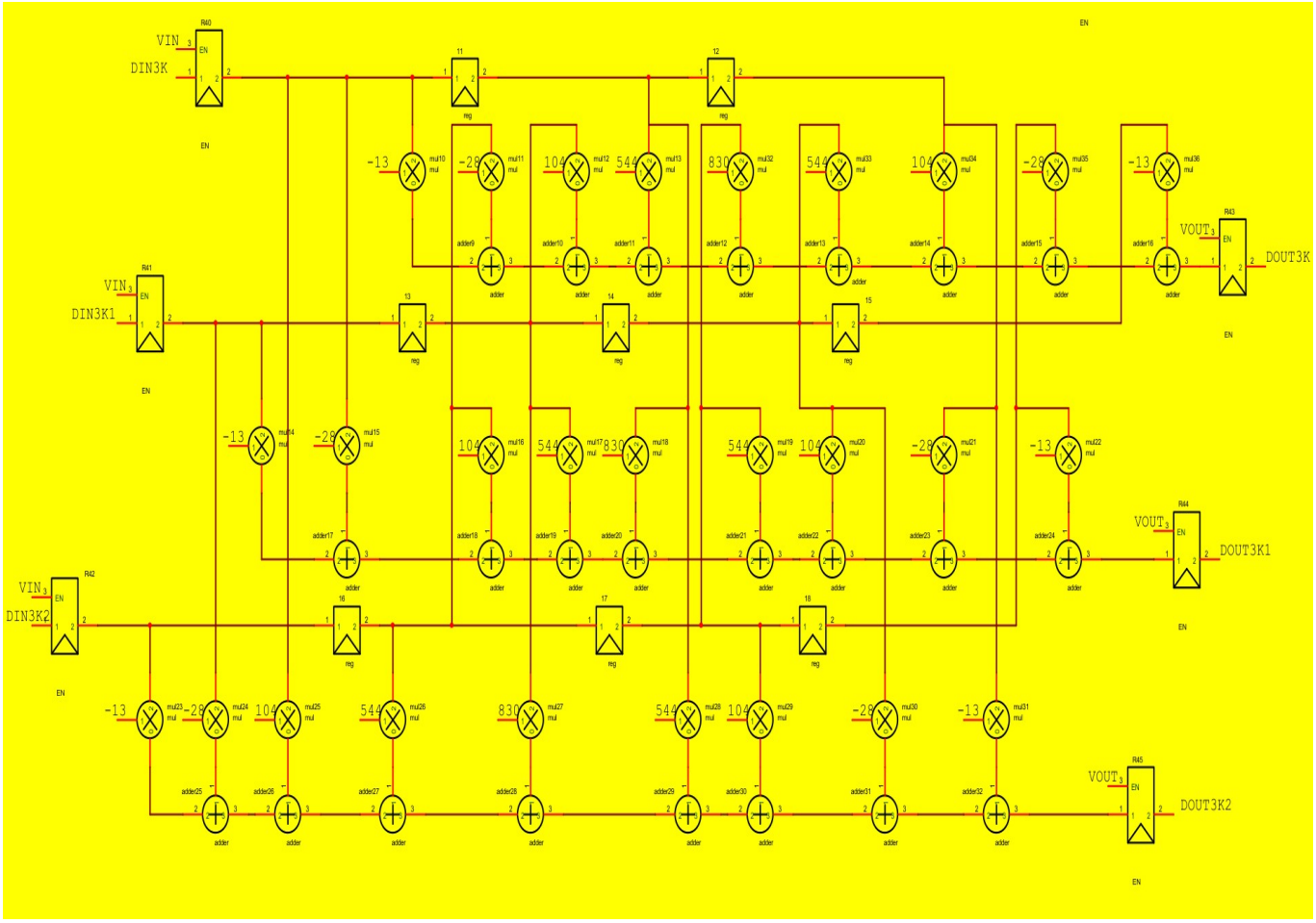

Figure 9: unfolded architecture

4.2 Logic synthesis

Without pipelined the unfolded architecture is able to handle the period of 2.9 ns, i.e. maximum frequency is 333 Mhz, each cycle has three times output, the slack time as Figure 12 shows. The area obtained from the Synopsis is 18144.4 um^2, larger than basic architecture, but smaller than pipeline architecture which we will see in the following description of this report.

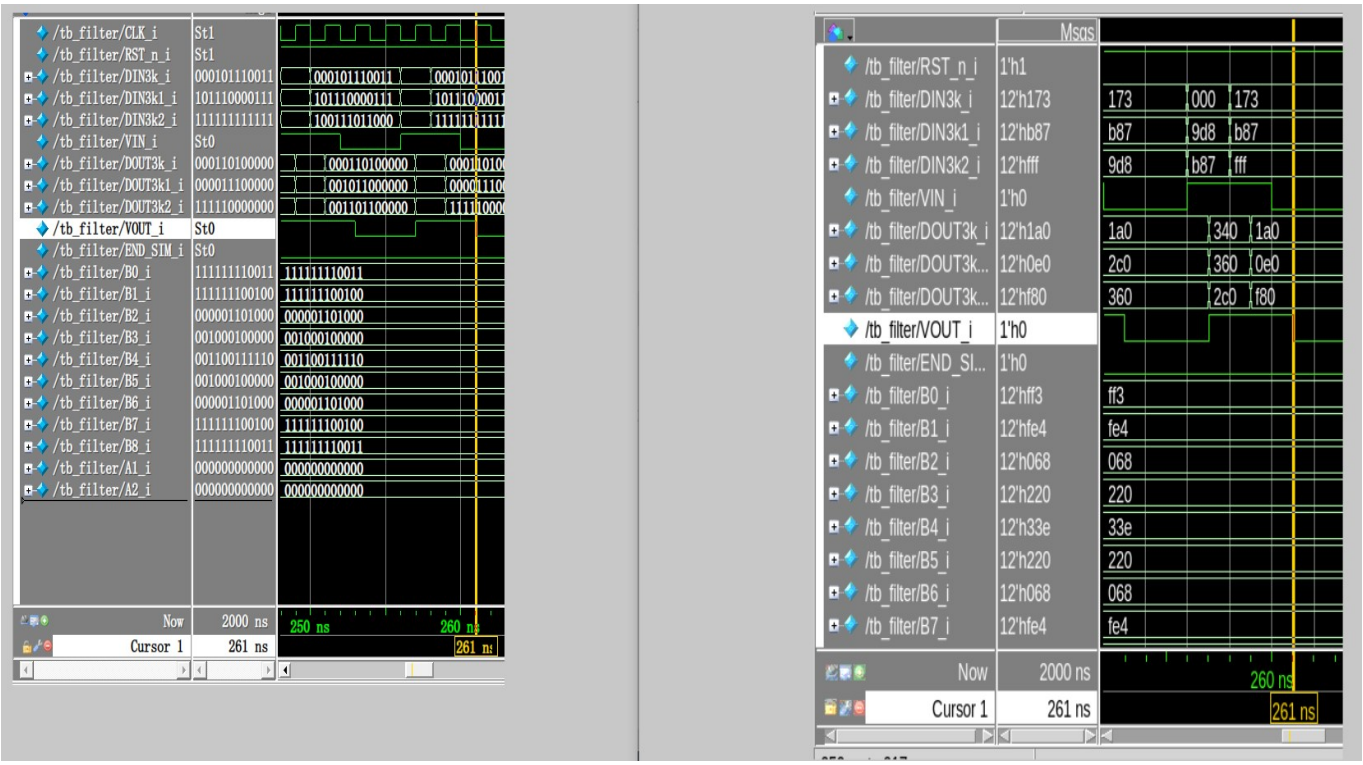

Figure 10: Pipeline maximum frequency simulation

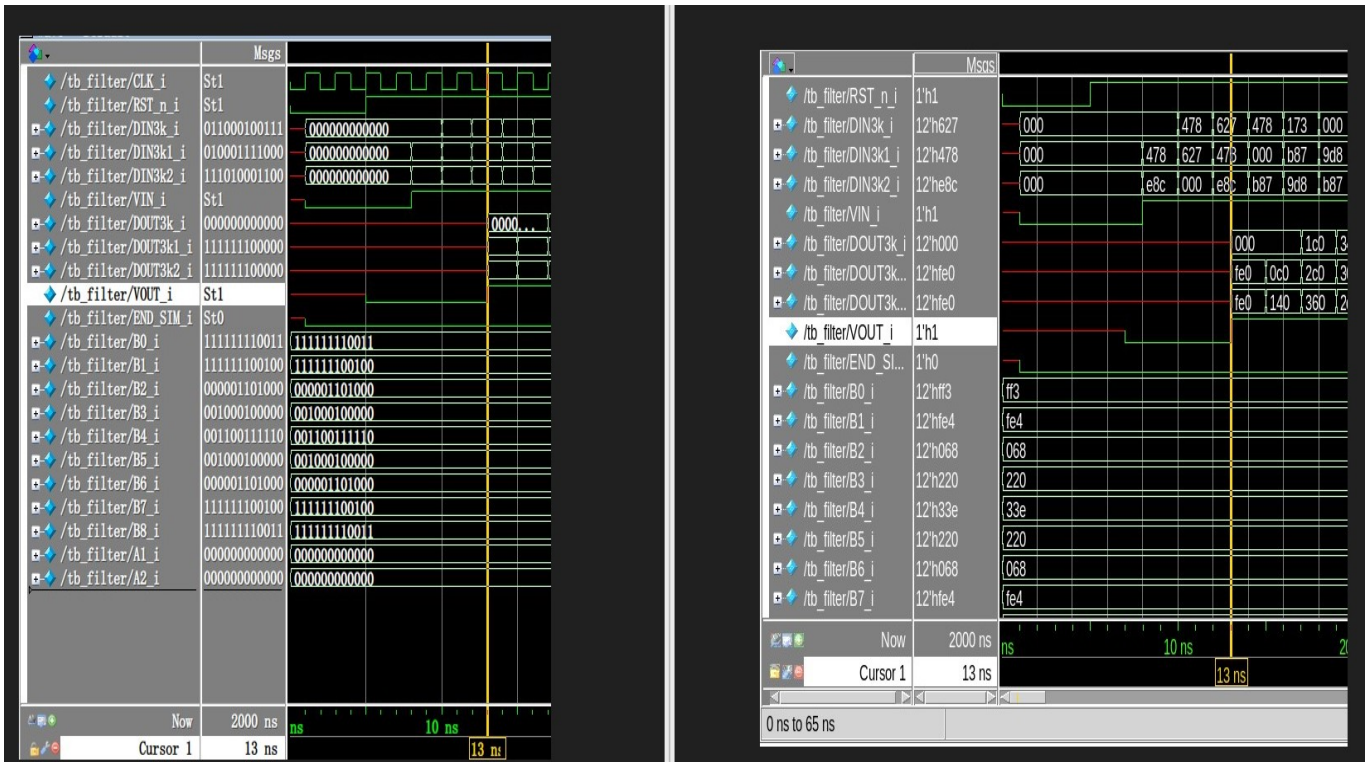

Figure 11: Pipeline maximum frequency simulation begin

```
clock clk (rise edge)                                  3.00      3.00
clock network delay (ideal)                            0.00      3.00
clock uncertainty                                     -0.07      2.93
sum_arr_reg[0][22]/CK (DFFR_X1)                        0.00      2.93 r
library setup time                                    -0.03      2.90
data required time                                               2.90
--------------------------------------------------------------------------
data required time                                               2.90
data arrival time                                               -2.90
--------------------------------------------------------------------------
slack (MET)                                                      0.00
```

Figure 12: Unfolded slack time

The pipeline base on the unfolded architecture is able to handle the period of 2.4 ns, i.e. maximum frequency is 417 Mhz, while synthesis with clock gating the minimum period is 2.41 ns, in frequency is 415mHz, obtained the power estimation as Table 10.

Table 12: Cadence advanced architecture total Power with the frequency of $(F_{max})/2$ detail (Power Units = 1mW)

Group	Internal Power	Switching Power	Leakage Power	Total Power	Percentage (%)
Sequential	0.4734	0.09511	0.03569	0.6042	16.92
Macro	0	0	0	0	0
IO	0	0	0	0	0
Combinational	1.177	1.307	0.3267	2.811	78.71
Clock(Combinational)	0.01381	0.1212	0.0001641	0.1352	3.786
Clock(Sequential)	0.01131	0.0094	0.000276	0.02099	0.5877
Total	1.676	1.532	0.3628	3.571	100

Table 13: Cadence advanced architecture area with the frequency of $F_{max}/2$

Gate area=0.7980 um^2			
Level 0 Module myfir	Gates=22321	Cells=9420	Area=17812.2 um^2

Table 10: Synopsis pipeline architecture Power frequency $=(F_{max})$ (Power Units = 1mW) Clock Gating

PowerGroup	Internal Power	Switching Power	Leakage Power	Total Power (%)
io_pad	0.0000	0.0000	0.0000	0.0000 (0.00%)
memory	0.0000	0.0000	0.0000	0.0000 (0.00%)
black_box	0.0000	0.0000	0.0000	0.0000 (0.00%)
clock_network	14.5652	185.6115	177.7137	200.3544 (2.63%)
register	676.1982	135.6332	3.1534e+04	843.3655 (11.07%)
sequential	1.1574	0.0000	4.2332e+03	5.3906 (0.07%)
combinational	2.9173e+03	3.3005e+03	3.4991e+05	6.5677e+03 (86.23%)
Total	3.6092e+03 uW	3.6217e+03 uW	3.8585e+05 nW	7.6168e+03 uW

While pipeline base on the unfolded architecture synthesis on Design Compiler without clock gating at maximum frequency shows the power estimation as Table 11.

Table 11: Synopsis pipeline architecture Power frequency $=(F_{max})$ (Power Units = 1mW) Without Clock Gating

PowerGroup	Internal Power	Switching Power	Leakage Power	Total Power (%)
io_pad	0.0000	0.0000	0.0000	0.0000 (0.00%)
memory	0.0000	0.0000	0.0000	0.0000 (0.00%)
black_box	0.0000	0.0000	0.0000	0.0000 (0.00%)
clock_network	0.0000	0.0000	0.0000	0.0000 (0.00%)
register	1.1453e+03	199.0721	3.2685e+04	1.3771e+03 (16.12%)
sequential	1.2236	0.0000	4.2332e+03	5.4568 (0.06%)
combinational	3.1348e+03	3.6544e+03	3.7224e+05	7.1615e+03 (83.82%)
Total	4.2813e+03 uW	3.8535e+03 uW	4.0916e+05 nW	8.5440e+03 uW

The simulation of original pipeline source script and the netlist simulation end time show as the Figure 10, the beginning is Figure 11 (the left one is original script).

4.3 Place and route

Changed the file clk_gen.vhd to adjust the frequency of that is half of the maximum frequency, then at the environment of QuestaSIM simulated, generated the data in the vcd file, turn to the tool of Cadence to read the vcd file, generate the data of power estimation as Table 12, area as Table 13.

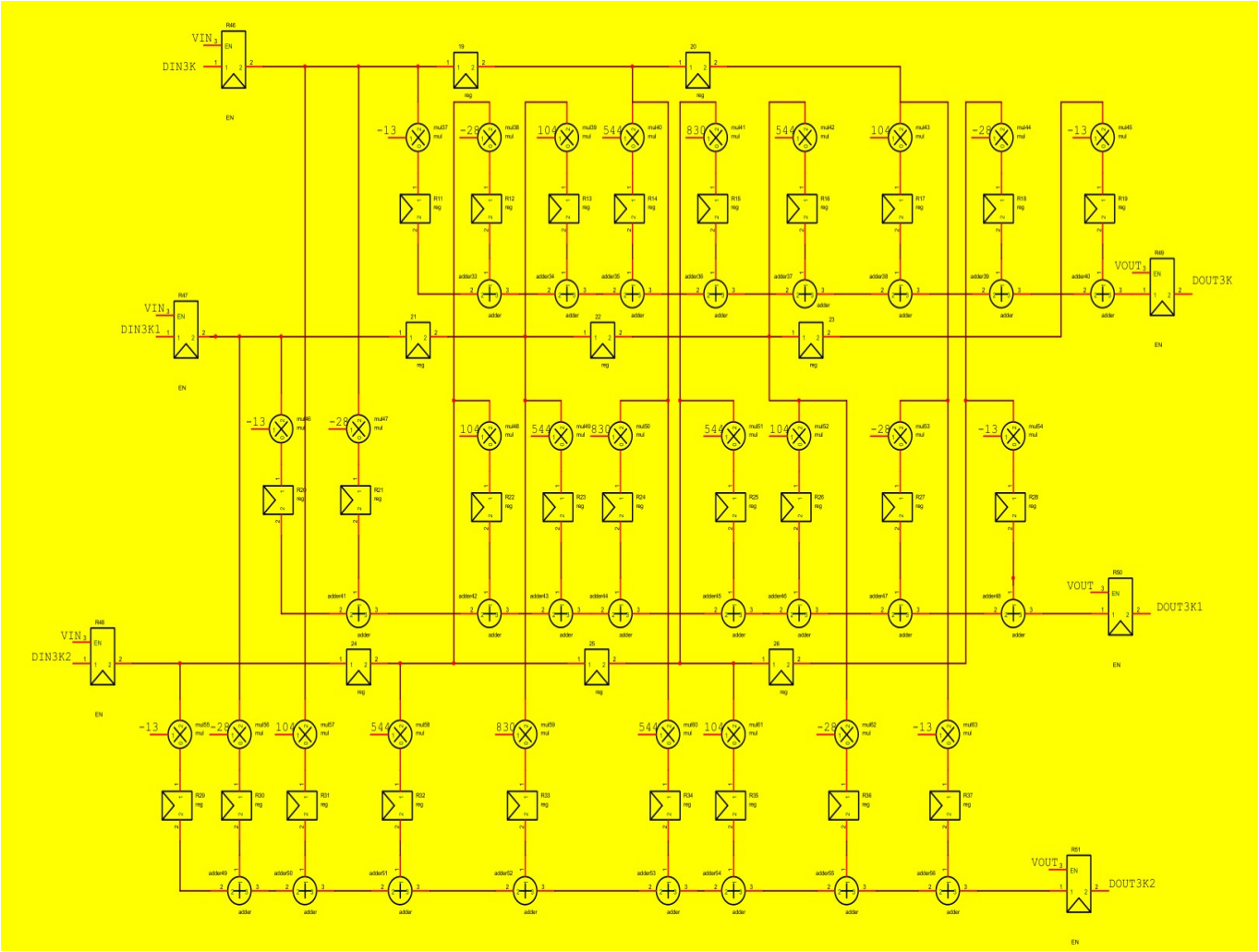

Figure 13: Pipeline architecture

4.4 Pipeline architecture

As Figure 13 shows, insert registers between the multiplier and adder, store the result of multiplication every clock cycle, and then reduce the length of the critical path. The possibility of pipeline architecture handling the shortest period is 2.4 ns i.e. maximum frequency is 417 MHz, and the slack time coming from Synopsis Design Compiler is 2.4ns. The difference between Synopsis and Cadence post route and place initial time, is shown in Figure 14 and end time Figure 15, both are simulated at the half of the maximum frequency.

The ability of pipeline architecture handling the shortest period is 2.4 ns i.e. maximum frequency is 417 MHz

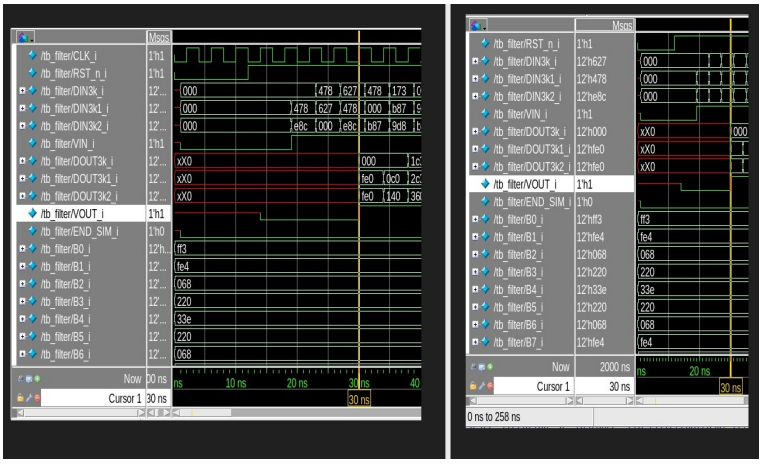

Figure 14: Pipeline Cadence VS Synopsis initial

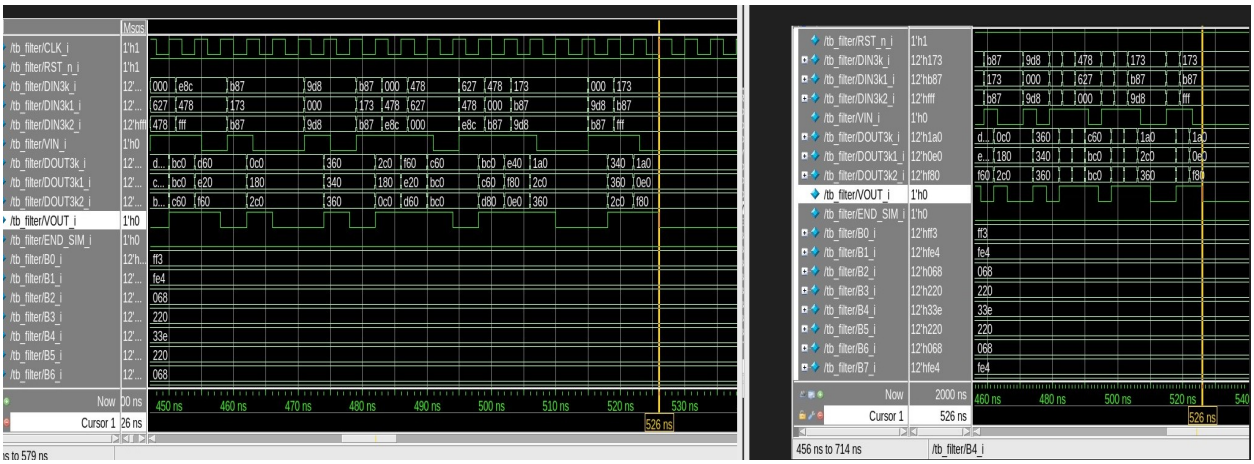

Figure 15: Pipeline Cadence VS Synopsis end

原始代码的PIPELINE TOTAL TIME 1303 NS

原始代码的basic famx/2 Total time 5535

4.5 Explanations, comparisons and comments

To make the conclusion among the different architectures, show in Table 14 is the total time difference, Table 15 is the time needed to prepare the first data output, these time data is obtain from the wave form that is described also in the picture, the total simulation time is recorded in the "saif" too, which is 567 ns longer than the time screen shot shows and the number in the table.

Total time basic Fmax/2 的DC NETLIST仿真时间是 5 5 3 7 ns

basic Fmax/2 的Cadence NETLIST仿真时间也是 5 5 3 7 ns

Table 14: Total time (ns)

	Synopsis	Post route and place	clock period
basic	2373	2379	6.8
pipeline	526	526	4.8

Table 15: Ready time (ns)

	Synopsis	Post route and place	clock period
basic	39	39	6.8
pipeline	30	30	4.8

Total time Cadence pipe line fmax/2 1305 ns

都验证过二分之 1 FMAX的结果和C一样

Comparison between basic filter architecture and advanced architecture is shown in Table 16 and Table 17 , advanced architecture which unfolds the basic architecture by the level of three and then inserts registers in the middle of multipliers and adders, so the area of the pipelined architecture is larger, the maximum frequency of the pipelined architecture is higher, but decrease the critical path delay. The pipelined architecture cannot reach the maximum frequency two times of basic, it is because the multiplier has a longer critical path than the adder.

Table 16: Area and power Comparison, all is at its half of maximum frequency

	Basic architecture	Pipeline base on unfolded architecture
Design Compiler		
Area	6173.9 µm2	19110.8 µm2
InternalPower	447.5644 uW	408.3982uW
SwitchingPower	425.3131 uW	236.7000uW
TotalDynamicPower	872.8784uW	645.980uW
LeakagePower	1.2018e+05W	3.6770e+05nW
TotalPower	9930.558uW	1012.8uW
Cadence		
Area	6143.5 µm2	17812.2 µm2
InternalPower	0.57381256mW	0.7278mW
SwitchingPower	0.4691156mW	0.6453mW
LeakagePower	0.1274mW	0.370mW
TotalPower	1.1703mW	1.7433mW

Table 17: Period Comparison

T_{min}	3.4ns	2.4ns
$2T_{min}$	6.8ns	4.8ns
f_{max}	294.1MHz	417MHz
$f_{max}/2$	147MHz	208MHz