國 立 交 通 大 學

電子工程學系 電子研究所碩士班

碩 士 論 文

可重組態之低功率
快速傅立葉轉換處理器

A Low Power Reconfigurable FFT Processor with Minimum Switching Activity

研 究 生：黃謙若　　chien-jo huang

指導教授：溫瓌岸　博士　　Dr. Kuei-Ann Wen

中 華 民 國 九 十 七 年 七 月

可重組態之低功率
快速傅立葉轉換處理器

A Low Power Reconfigurable FFT Processor with Minimum Switching Activity

研 究 生：黃謙若　　Student ： chien-jo huang

指導教授：溫瓌岸 博士　　Advisor ： Dr. Kuei-Ann Wen

國 立 交 通 大 學

電子工程學系 電子研究所碩士班

碩 士 論 文

A Thesis

Submitted to Department of Electronics Engineering & Institute of Electronics

College of Electrical & Computer Engineering

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of Master

in

Electronic Engineering

June 2007

中 華 民 國 九 十 七 年 七 月

可重組態之低功率
快速傅立葉轉換處理器

學生：黃謙若　　指導教授：溫瓌岸 博士

國立交通大學

電子工程學系　電子研究所碩士班

摘要

本論文提出了一個可重組態之低功率快速傅立葉轉換處理器，此記憶體式架構之處理器能夠處理 64 到 8192 點的傅立葉轉換。此外，本篇論文提出了一個藉由改變傅立葉係數之順序而達到最小化 switching activity 的方法。藉由 Synopsys Design Complier 的合成，在 UMC 0.18 um COMS 的製程環境下，所提出之設計在不包含記憶體的情況下僅需 53306 個邏輯閘，並且在 1.8 伏特之電壓源供應下，僅有 75.82 毫瓦的低功率消耗。

A Low Power Reconfigurable FFT Processor with Minimum Switching Activity

Student：chien-jo huang Advisor : Dr. Kuei-Ann Wen

Department of Electronics Engineering

Institute of Electronics

National Chiao-Tung University

Abstract

In this thesis, a low power reconfigurable FFT processor is proposed. The memory based FFT can be configured as from 64-point to 8192-point. Besides, a modified coefficient ordering method with minimum switching activity is proposed for low power design. The switching activity of twiddle factor computation can be reduced from 633 to 480 at the first stage of 64-point and 139 to 0 at the first stage of 16-point. The proposed design synthesized to UMC 0.18um CMOS standard cell technology library with Synopsys Design Compiler. The gate count of the proposed architecture without ram is 53306 at 111 MHz clock rate and power consumption is 75.82 mW at power supply 1.8 V.

誌　　謝

首先，第一個要感謝的是指導教授，溫瓌岸教授。感謝老師在兩年研究生涯中，不斷的給予謙若指導與督促。溫老師的循循教誨，讓學生在學習訓練的路途上，能夠快速而正確的修正自己的研究方向，並且保持不鬆懈的心態進行研究。也感謝 TWT_LAB 在這兩年中提供的豐富研究資源，讓我在研究上無後顧之憂。

感謝實驗室的學長們的指導與照顧:彭嘉笙，溫文燊，林立協，陳哲生，鄒文安，趙晧名，蔡彥凱，張懷仁，莊翔琮，吳家岱，梁書旗，李漢健，侯閎仁。感謝兩年來一起打拚的同學:葉柏麟，黃俊彥，楊士賢，王磊中，黃國爵，林佳欣。大家在生活上的互相扶持與鼓勵，讓原本辛苦煩悶的研究工作，也變的輕鬆愉快許多。

同時也要感謝實驗室的助理:淑怡，恩齊，慶宏，苑佳、智伶、宛君、嘉誠，有妳們幫忙處理實驗室的雜務，才能讓我們能夠專心致力於研究。

最後，感謝默默支持我的家人，爸爸，媽媽，及哥哥。你們不斷的支持與鼓勵，讓我覺得更需要努力來回報你們。

黃謙若

2008 年 7 月

Contents

List of Figures

List of Tables

Chapter 1. Introduction.

1.1. Motivation

Fast Fourier Transform (FFT) plays an important role in digital signal processing and wireless communication systems. Each operation standard has different Size of FFT. In other words, a FFT processor should be able to support diverse required of applications. This is the reason why reconfigurable FFT processor becomes a notable issue. Another notable issue is low power issue especially for mobile and wireless applications. A low power device is able to be used for long time. In conclusion, a FFT processor with low area and low power consumption is needed by portable feature of applications. Due to the recursive computation architecture, the memory based FFT has less hardware than the pipeline based architecture for long-length point FFT. This thesis proposed a reconfigurable memory based FFT processor with low power consumption.

1.2. Discrete Fourier Transform

In the field of digital signal processing, the Discrete Fourier Transform (DFT) plays an important role. It can be used to calculate a signal's frequency response. The N-point DFT of a sequence x(n) is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \qquad k = 0,1 \mathbf{L}\, N-1 \tag{1.1}$$

Where x(n) and X(k) are the sequence of complex numbers.

The twiddle factor is

$$W_N^{nk} = e^{-j(2p\,nk/N)} = \cos(\frac{2p\,nk}{N}) - j\sin(\frac{2p\,nk}{N}) \tag{1.2}$$

Accord to eq. (1.1), the computational complexity is $O(N^2)$. It needs N^2 complex multiplications and N(N-1) complex additions [1]. The FFT is an efficient algorithms to compute the DFT. The computational complexity can be reduced to $O(N\log_r N)$, where r means the radix-r FFT. The radix-r FFT can be derived from DFT by decomposing the N-point DFT into a set of recursively related r-point transform. There are two basic types of FFT algorithm, decimation in time (DIT) and decimation in frequency (DIT). The DIT algorithm is to decompose x(n) into radix-r modules sequence, and the DIF algorithm is to decompose X(k) in the same way.

1.3. Introduction to FFT algorithm

Cooley-Tukey FFT algorithms [2] are efficient realization techniques of DFT operations. The radix-2 FFT algorithms are the simplest FFT algorithms. The decimation-in-time (DIT) radix-2 FFT recursively partitions a DFT into two half-length DFTs of the even-indexed and odd-indexed time samples. Similarly, the decimation-in-frequency (DIF) radix-2 FFT partitions the DFT computation into even-indexed and odd-indexed outputs, which can each be computed by shorter-length DFTs of different combinations of input samples. Recursive application of this decomposition to the shorter-length DFTs results in the full radix-2 decimation-in-frequency FFT. The DIF radix-2 FFT is derived as follows:

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \qquad k = 0,1\,\mathbf{L}\,N-1 \tag{1.3}$$

Where $W_N^{nk} = e^{-j(2p\,nk/N)}$

Then x(n) is decomposed into two parts as shown in eq. 1.4.

$$\begin{aligned} X[k] &= \sum_{n=0}^{\frac{N}{2}-1} x[n]*W_N^{nk} + \sum_{n=\frac{N}{2}}^{N-1} x[n]*W_N^{nk} \\ &= \sum_{n=0}^{\frac{N}{2}-1} x[n]*W_N^{nk} + (-1)^k \sum_{n=0}^{\frac{N}{2}-1} x[n+\frac{N}{2}]*W_N^{nk} \end{aligned} \tag{1.4}$$

Let even-indexed and odd-indexed of X[k] are 2r and 2r+1, respectively.

$$\begin{aligned} X[2r] &= \sum_{n=0}^{\frac{N}{2}-1} x[n] * W_N^{2nr} + (-1)^{2r} x[n+\frac{N}{2}] * W_N^{2nr} \\ &= \sum_{n=0}^{\frac{N}{2}-1} (x[n] + x[n+\frac{N}{2}]) * W_{N/2}^{nr} \end{aligned} \tag{1.5}$$

$$\begin{aligned} X[2r+1] &= \sum_{n=0}^{\frac{N}{2}-1} x[n] * W_N^n * W_N^{2nr} + (-1)^{2r+1} x[n+\frac{N}{2}] * W_N^n * W_N^{2nr} \\ &= \sum_{n=0}^{\frac{N}{2}-1} (x[n] - x[n+\frac{N}{2}]) W_N^n * W_{N/2}^{nr} \end{aligned} \tag{1.6}$$

According to eq. (1.5) and (1.6), the even-index of X[k] is equal to the result of N/2-point DFT operation of x[n] +x[n+N/2], and the odd-index of X[k] is equal to the result of N/2-point DFT operation of (x[n]-x[n+N/2])* W_N^n , as shown in Fig. 1.1. A radix-2 butterfly diagram is shown in Fig. 1.2.

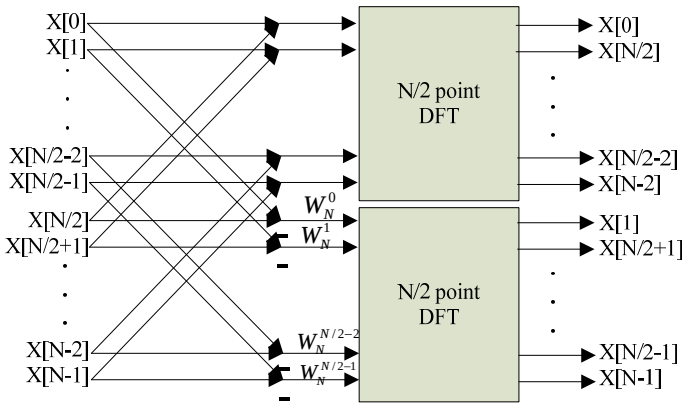


Figure 1.1 Decomposition graph of DIF FFT

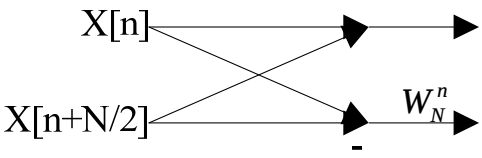


Figure 1.2 Radix-2 butterfly diagram

DIF radix-4 algorithm is similar to DIF radix-2 algorithm. In the DIF radix-4 algorithm, x[n] is decomposed into four parts instead of two parts.

$$X[4r]=\sum_{n=0}^{\frac{N}{4}-1}\{(x[n]+x[n+\frac{N}{2}])+(x[n+\frac{N}{4}]+x[n+\frac{3N}{4}])\}*W_{N/4}^{nr} \quad (1.7)$$

$$X[4r+1]=\sum_{n=0}^{\frac{N}{4}-1}\{(x[n]-x[n+\frac{N}{2}])-j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])\}W_N^n*W_{N/4}^{nr} \quad (1.8)$$

$$X[4r+2]=\sum_{n=0}^{\frac{N}{4}-1}\{(x[n]+x[n+\frac{N}{2}])-(x[n+\frac{N}{4}]+x[n+\frac{3N}{4}])\}W_N^{2n}*W_{N/4}^{nr} \quad (1.9)$$

$$X[4r+3]=\sum_{n=0}^{\frac{N}{4}-1}\{(x[n]-x[n+\frac{N}{2}])+j(x[n+\frac{N}{4}]-x[n+\frac{3N}{4}])\}W_N^{3n}*W_{N/4}^{nr} \quad (1.10)$$

The results of each part are shown in eq. 1.7 to eq. 1.10, respectively. Fig 1.3 shows a radix-4 butterfly diagram. Radix-4 algorithm has lower computational complexity than radix-2 algorithm for N-point FFT which N is equal to 4^n. In conclusion, the high radix algorithm has low computational complexity but it is complex to implement.

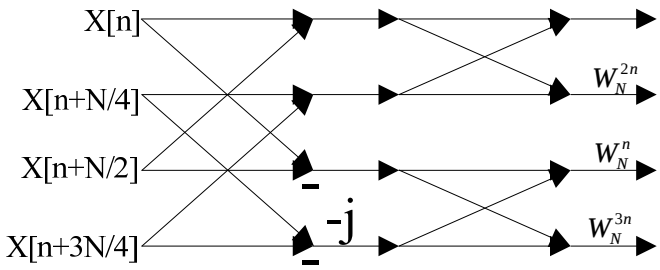


Figure 1.3 Radix-4 butterfly diagram

1.4. Introduction to FFT architecture

In general, there are two kinds of FFT architecture. One is pipeline based architecture. The other is memory based architecture. Pipeline based architecture has higher throughput than memory based architecture. On the other hand, memory based has the advantage of low area for long-length point FFT due to a fixed processing element. We discuss several kinds of architecture in the following sections.

1.4.1. Memory based FFT architecture

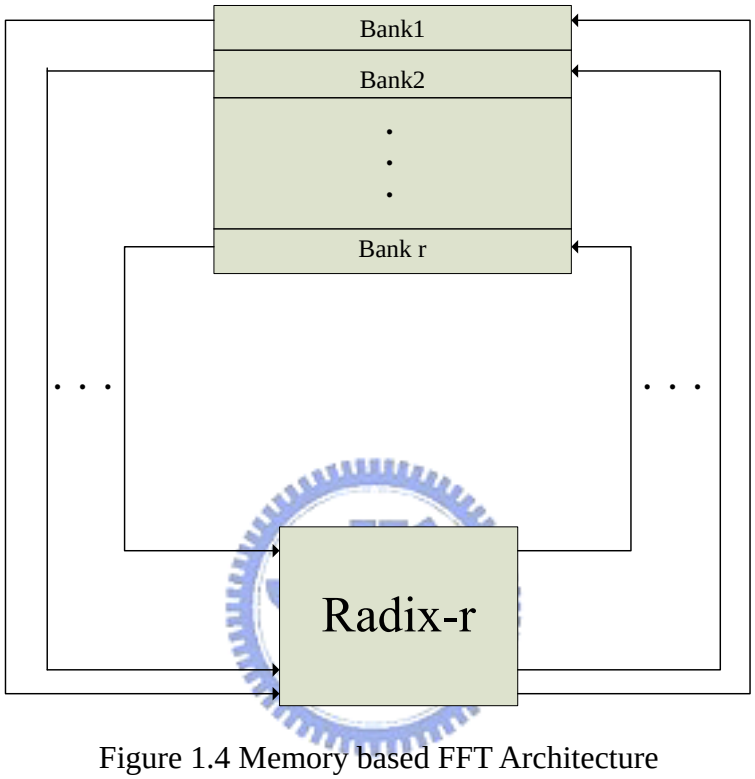


Figure 1.4 Memory based FFT Architecture

Fig. 1.4 shows a memory based FFT architecture. It consists of a processing element with radix-r and a dual port memory. The recursive computation is a problem due to only one processing element. Therefore, the memory based architecture usually has lower throughput than pipeline base architecture. In general, a complete computation of N-point FFT needs $N/r*\log_r N$ butterfly operations.

1.4.2. Pipeline based FFT architecture

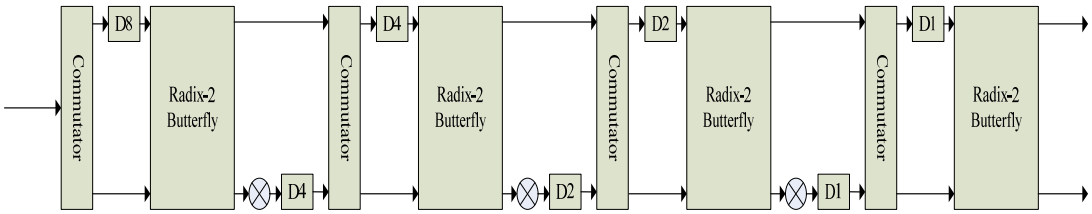


Figure 1.5 Radix-2 Multipath Delay Commutator (R2MDC)

Fig. 1.5 shows R2MDC [3] architecture of 16-point FFT. R2MDC is the simplest pipeline based architecture to implement. An input sequence is divided to two parallel data streams, and then enters the butterfly operator at the correct time. The utilization of each complex multiplier and butterfly operator is about 50%. It needs $\log_2 N - 1$ complex multipliers, $2 * \log_2 N$ complex adder and 1.5N-2 registers to complete a 16-point FFT.

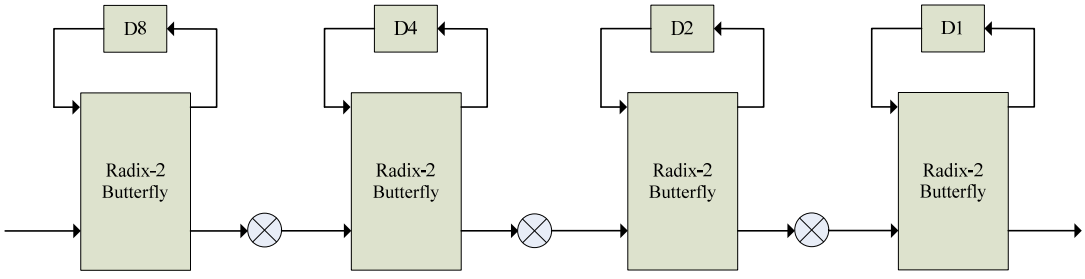


Figure 1.6 Radix-2 Single-path Delay Feedback (R2SDF)

Fig. 1.6 shows R2SDF [4] architecture of 16-point FFT. Since the utilization of registers in R2MDC is only 50%, the R2SDF architecture reduces its registers by half. With feedback delay registers, the utilization can achieve 100%. The modified butterfly architecture can not only do butterfly operation, but also pass to the next stage.

1.4.3. Summary

According to the above section, we can obtain the table 1.1. From table 1.1, we can know the number of processing element of pipeline based architecture increased for long-length point FFT. From hardware consideration, memory based architecture is a good choice for long-length point FFT due to a fixed processing element.

Table 1.1 Comparisons of FFT architectures

	R2MDC	R2SDF	Memory-based Radix-r
# of complex multiplier	$\log_2 N-1$	$\log_2 N-1$	r-1
# of complex adder	$2*\log_2 N$	$2*\log_2 N$	$r*\log_2 r$
Memory Size	1.5N-2	N-1	N

Chapter 2. Low Power Design

2.1. Introduction to Switching Activity

Low power is an important issue in hardware design because of mobile and wireless systems. In this thesis, we discuss low power architecture by reducing the dynamic power. Dynamic power dissipation occurs when the transistor is charging or discharging. As shown in Fig. 2.1, A CMOS inverter with output load capacitance charged.

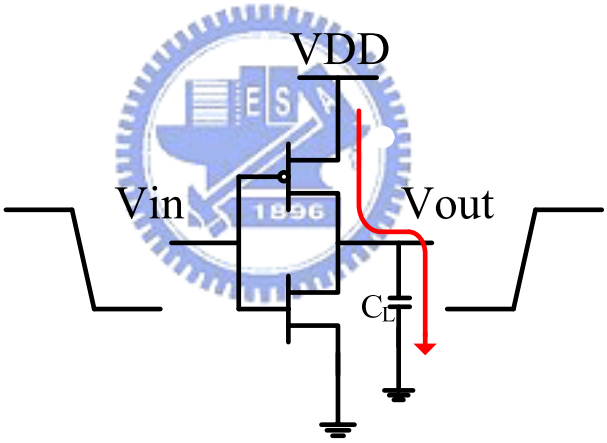


Figure 2.1 Charging of inverter output load capacitance C_L

The dynamic power dissipation is given by:

$$P_{dynamic} = C_L V^2 f \alpha \quad (2.1)$$

Where C_L is the total output capacitance, V is the supply voltage, f is the operating frequency and α is the switching activity factor which is defined as the average number of times the gate makes an active transition in a single clock cycle. Therefore,

in order to achieve low power design in CMOS circuits, minimum switching activity is an efficient method.

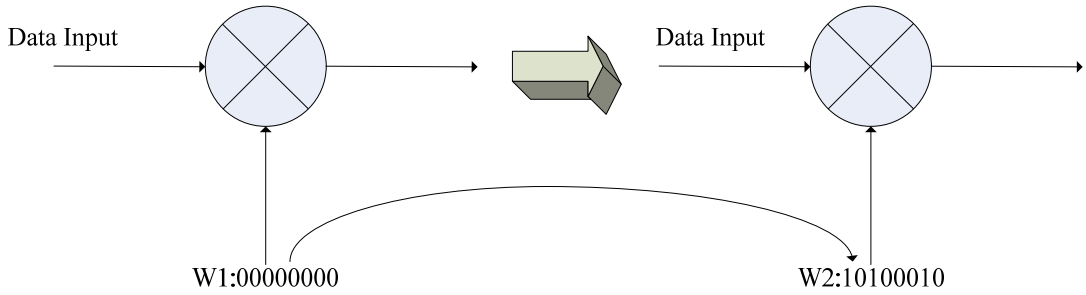


Figure 2.2 Switching activity of a multiplier

From a multiplier consideration, as shown in Fig. 2.2, W1 is one input of the multiplier. When W2 enter the multiplier in the next clock cycle, dynamic power consumes because of switching activity. We can obtain the switching activity by hamming distance [5].

Hamming distance is defined as the number of 1's of XOR operation between two binary coefficients. Fig 2.3 shows that three bit is different from W1 to W2. It means that the hamming distance between W1 and W2 is 3.

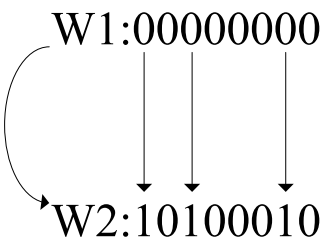


Figure 2.3 Hamming distance between W1 and W2

We discuss the switching activity from a system consideration. Fig. 2.4 shows a general architecture of memory based FFT with a radix-4 processing element. Dynamic power dissipation occurs during the operation of processing element. Further, the complex multiplier within the processing element is one of the most power-consuming units of the FFT processor. We investigate to reduce dynamic power consumption by minimizing the switching activity of the complex multipliers in the processing element. The following section discusses minimum switching activity by coefficient ordering technique.

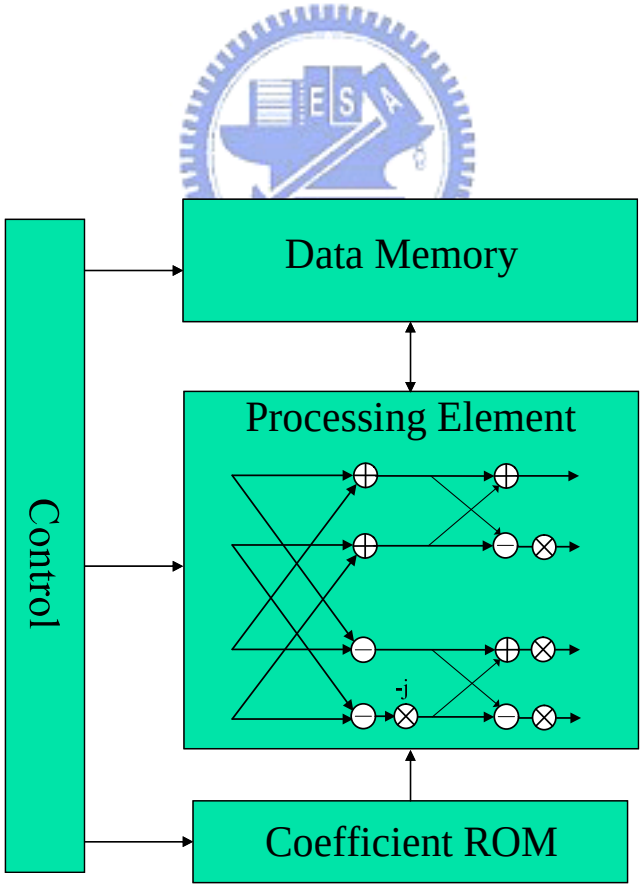


Figure 2.4 Memory based FFT with radix-4 processing element

2.2. Minimum Switching Activity

2.2.1 Minimum switching activity of 16-point FFT

In this section, we investigate to minimize switching activity by reordering coefficient sequence. Fig. 2.5 shows the data flow graph of a 16-point FFT. The number in the open circle represents the data sequence at the corresponding stage. The signal flow of the first, the second, the third and the fourth four data with the corresponding coefficients at stage 010 are shown in Fig. 2.6(a), 2.6(b), 2.6(c) and 2.6(d) respectively.

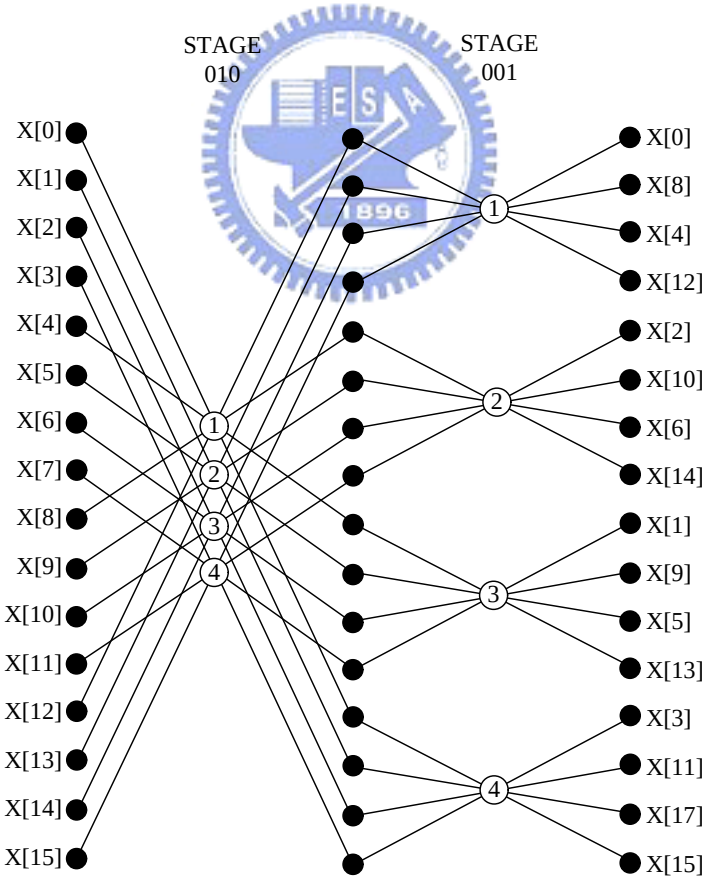


Figure 2.5 The data flow graph of a 16-point FFT

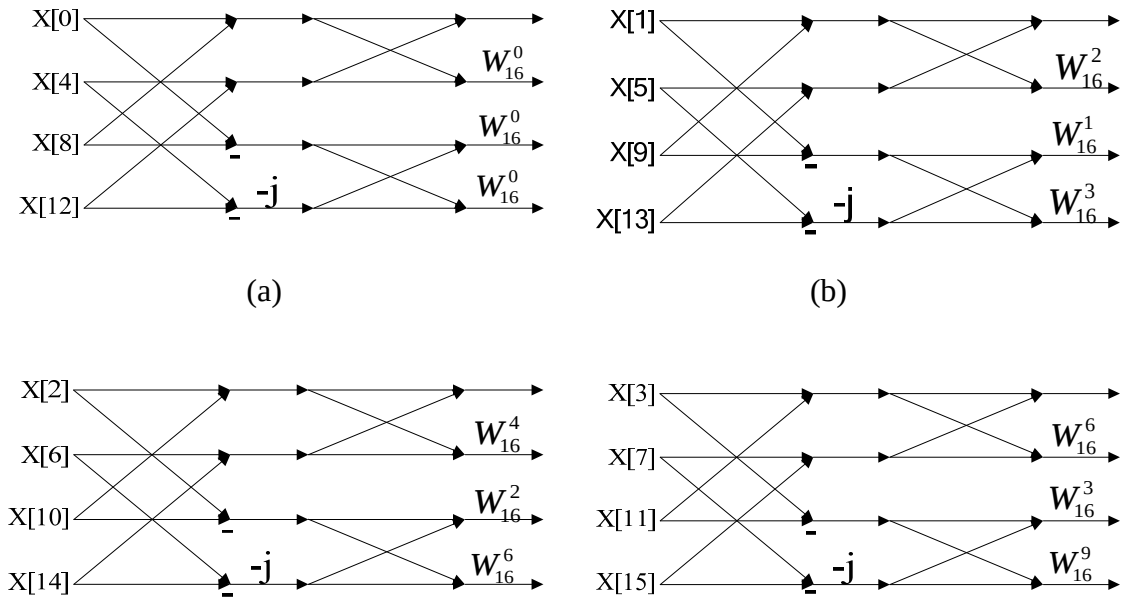


Figure 2.6 Signal flow of 16-point FFT

According to Fig. 2.6, we arrange data sequence and the corresponding coefficients at stage 010 of 16-point FFT in table 2.1. From this table, we can obtain the switching activity of the successive coefficients by hamming distance when the processing element finished computation of stage 010 of 16-point FFT.

Table 2.1 Data sequence and the corresponding coefficients at stage 010 of 16-point

Data Seq.	1	2	3	4
Corresponding Coefficients	W_{16}^{0} W_{16}^{0} W_{16}^{0} W_{16}^{0}	W_{16}^{0} W_{16}^{2} W_{16}^{1} W_{16}^{3}	W_{16}^{0} W_{16}^{4} W_{16}^{2} W_{16}^{6}	W_{16}^{0} W_{16}^{6} W_{16}^{3} W_{16}^{9}

Table 2.2 is obtained from table 2.1 by computing the hamming distance between each coefficient set (Let four coefficients of the same data sequence are a set). We define that TM (i, j) is the switching activity between coefficient set of data seq. i and coefficient set of data seq. j. For example, TM (1, 2) represents the switching activity of the two coefficient sets of data seq.1 and data seq.2. The value 51 means the hamming distance between this two coefficient sets, as shown in Fig. 2.7. Form this transition matrix, we can compute the switching activity of original coefficient sequence at stage 010 of 16-point FFT by the eq. 2.2.

$$total\_switching\_activity = TM(1,2) + TM(2,3) + TM(3,4) = 139 \quad (2.2)$$

Table 2.2 Transition matrix of

the switching activity with 16 bit word length at stage 010 of 16-point FFT

Data Seq.	1	2	3	4
1	0	51	39	55
2	51	0	42	48
3	39	42	0	46
4	55	48	46	0

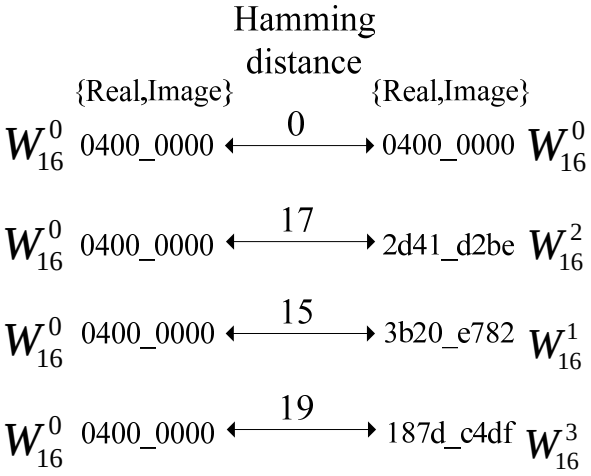


Figure 2.7 Hamming distance between two coefficient sets

In order to minimize switching activity, we can reduce the diverseness of any two successive coefficient sets by twiddle factor symmetry property. For example, table 2.1 lists a common coefficient sets at stage 010 of 16-point FFT. By means of twiddle factor, we can modify the coefficient sets so that the diverseness of any two successive coefficient sets is reduced, as shown in Table 2.3.

Table 2.3 The modified coefficient sets from Table 2.1

Data seq.	1	2	3	4
Corresponding Coefficients	W_{16}^0 W_{16}^0 W_{16}^0 W_{16}^0	W_{16}^0 W_{16}^2 W_{16}^1 W_{16}^3	W_{16}^0 $W_{16}^0*(-j)$ W_{16}^2 $W_{16}^2*(-j)$	W_{16}^0 $W_{16}^2*(-j)$ W_{16}^3 $W_{16}^1*(-1)$

It means that the coefficient, W_N^C, can be decomposed as eq. 2.3.

$$\begin{aligned} W_N^C &= \{W_N^{C'}, W_N^{C'+N/4}, W_N^{C'+N/2}, W_N^{C'+3N/4}\} \\ &= W_N^{C'} * \{1, -j, -1, j\} \end{aligned} \tag{2.3}$$

This technique is usually used to reduce the coefficient memory size [6] because the coefficient memory size just requires N/4 word. A novel application of this technique is proposed in this thesis. By means of associativity of multiplication, we can let the data and the modified coefficient do multiplication first, and then the phase of the output of multipliers is recovered. In another word, dynamic power consumption is reduced in the multipliers because the diverseness of coefficients is obviously reduced.

In Fig. 2.1, we can see a radix-4 processing element which has three multipliers in order to deal with four data at a time. Table 2.3 shows that only three kinds of coefficients are needed at stage 010 of 16-point FFT (W_{16}^0 is ignored because it doesn't need do multiplication). A novel multiplier module is proposed. It doesn't change any coefficient in multipliers at stage 010 of 16-point FFT to minimize switching activity. We discuss the detail in chapter 3.3.

2.2.2 Minimum switching activity of 64-point FFT

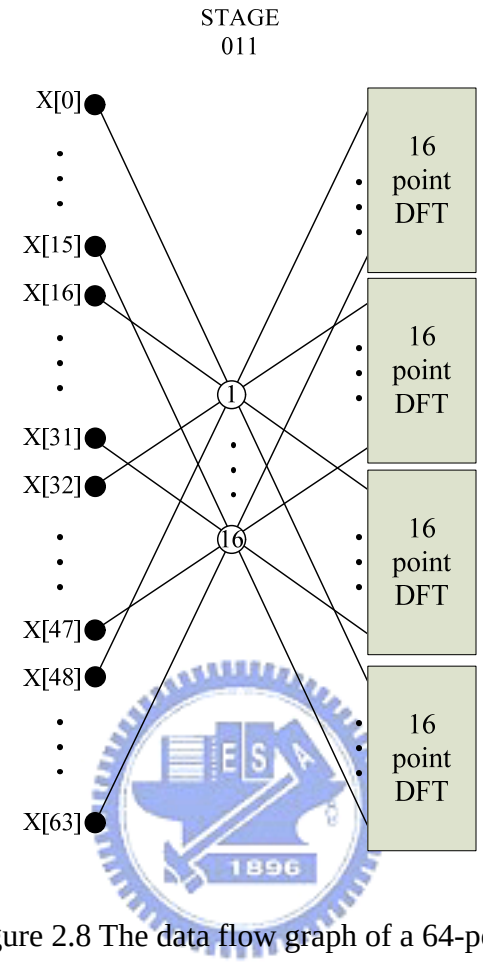


Figure 2.8 The data flow graph of a 64-point FFT

Fig. 2.8 shows the data flow of a 64-point FFT. The number in the open circle represents the data sequence at stage 011. Table 2.4 lists the corresponding coefficient sets which are similar to 16-point FFT in section 2.2.1. We also need to modify the coefficient sets so that the diverseness of any two coefficient sets is reduced, as shown in Table 2.5.

Table 2.4 Data sequence and

the corresponding coefficients at stage 011 of 64-point

Data seq.	1	2	3	4	5	6	7	8
Corresponding Coefficients	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}
	W_{64}^{0}	W_{64}^{2}	W_{64}^{4}	W_{64}^{6}	W_{64}^{8}	W_{64}^{10}	W_{64}^{12}	W_{64}^{14}
	W_{64}^{0}	W_{64}^{1}	W_{64}^{2}	W_{64}^{3}	W_{64}^{4}	W_{64}^{5}	W_{64}^{6}	W_{64}^{7}
	W_{64}^{0}	W_{64}^{3}	W_{64}^{6}	W_{64}^{9}	W_{64}^{12}	W_{64}^{15}	W_{64}^{18}	W_{64}^{21}

Data seq.	9	10	11	12	13	14	15	16
Corresponding Coefficients	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}	W_{64}^{0}
	W_{64}^{16}	W_{64}^{18}	W_{64}^{20}	W_{64}^{22}	W_{64}^{24}	W_{64}^{26}	W_{64}^{28}	W_{64}^{30}
	W_{64}^{8}	W_{64}^{9}	W_{64}^{10}	W_{64}^{11}	W_{64}^{12}	W_{64}^{13}	W_{64}^{14}	W_{64}^{15}
	W_{64}^{24}	W_{64}^{27}	W_{64}^{30}	W_{64}^{33}	W_{64}^{36}	W_{64}^{39}	W_{64}^{42}	W_{64}^{45}

Table 2.5 The modified coefficient sets from Table 2.4

Data seq.	1	2	3	4	5	6	7	8
Corresponding Coefficients	W_{64}^{0} W_{64}^{0} W_{64}^{0} W_{64}^{0}	W_{64}^{0} W_{64}^{2} W_{64}^{1} W_{64}^{3}	W_{64}^{0} W_{64}^{4} W_{64}^{2} W_{64}^{6}	W_{64}^{0} W_{64}^{6} W_{64}^{3} W_{64}^{9}	W_{64}^{0} W_{64}^{8} W_{64}^{4} W_{64}^{12}	W_{64}^{0} W_{64}^{10} W_{64}^{5} W_{64}^{15}	W_{64}^{0} W_{64}^{12} W_{64}^{6} $W_{64}^{2}*(-j)$	W_{64}^{0} W_{64}^{14} W_{64}^{7} $W_{64}^{5}*(-j)$

Data seq.	9	10	11	12
Corresponding Coefficients	W_{64}^{0} $W_{64}^{0}*(-j)$ W_{64}^{8} $W_{64}^{8}*(-j)$	W_{64}^{0} $W_{64}^{2}*(-j)$ W_{64}^{9} $W_{64}^{11}*(-j)$	W_{64}^{0} $W_{64}^{4}*(-j)$ W_{64}^{10} $W_{64}^{14}*(-j)$	W_{64}^{0} $W_{64}^{6}*(-j)$ W_{64}^{11} $W_{64}^{1}*(-1)$

Data seq.	13	14	15	16
Corresponding Coefficients	W_{64}^{0} $W_{64}^{8}*(-j)$ W_{64}^{12} $W_{64}^{4}*(-1)$	W_{64}^{0} $W_{64}^{10}*(-j)$ W_{64}^{13} $W_{64}^{7}*(-1)$	W_{64}^{0} $W_{64}^{12}*(-j)$ W_{64}^{14} $W_{64}^{10}*(-1)$	W_{64}^{0} $W_{64}^{14}*(-j)$ W_{64}^{15} $W_{64}^{13}*(-1)$

Table 2.6 Transition matrix of

the switching activity with 16 bit word length

Date seq.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	63	52	55	51	44	56	50	34	52	46	58	51	47	50	39
2	63	0	29	48	36	45	47	47	49	29	39	45	42	46	43	52
3	52	29	0	53	39	46	54	48	44	44	30	44	47	39	46	43
4	55	48	53	0	42	45	37	41	49	47	47	29	42	44	39	46
5	51	36	39	42	0	39	47	33	45	35	39	43	32	42	47	42
6	44	45	46	45	39	0	46	44	40	40	46	46	43	29	44	45
7	56	47	54	37	47	46	0	36	48	48	46	46	39	47	30	39
8	50	47	48	41	33	44	36	0	40	42	48	40	35	47	44	29
9	34	49	44	49	45	40	48	40	0	44	44	42	45	47	48	45
10	52	29	44	47	35	40	48	42	44	0	36	44	33	41	48	47
11	46	39	30	47	39	46	46	48	44	36	0	46	47	37	54	47
12	58	45	44	29	43	46	46	40	42	44	46	0	39	45	46	45
13	51	42	47	42	32	43	39	35	45	33	47	39	0	42	39	36
14	47	46	39	44	42	29	47	47	47	41	37	45	42	0	53	48
15	50	43	46	39	47	44	30	44	48	48	54	46	39	53	0	29
16	39	52	43	46	42	45	39	29	45	47	47	45	36	48	29	0

Table 2.6 is obtained from table 2.5 by computing the hamming distance between any two coefficient sets. From this transition matrix, we can arrange the sequence of coefficient sets in order to minimize switching activity. However, finding an optimal sequence with minimum switching activity is difficult by manpower. An algorithm for finding the sequence with minimum switching activity is shown in Fig.2.9. Step 1 is set maximum value to itself and set initial state. Where cs represents the current sequence, index represents the corresponding index, and t represents the remaining

times needs to execute. Step 2 is find the next sequence with minimum switching activity. Step 3 is update cs, index, and t to next state. Step 4 is return to step 2 until t is equal to zero.

```
For i=1 to N/4
   TM(i,i)=99 (set maximum)
cs=1
index=1
t=N/4-1
minimum switching activity=0
 while t
      minimum switching activity = minimum switching activity + min(TM(cs,-))
       TM(-,cs)=99
       cs=find(min(TM(cs,-)))
       min_seq(index)=cs;
       t=t-1
       l=l+1
```

Figure 2.9 Algorithm of minimum switching activity

According to this algorithm, we can find an optimal data sequence with minimum switching activity. The switching activity of twiddle computation is thus reduced from 633 to 480. The reordered data sequence is 1, 9, 6, 14, 11, 3, 2, 10, 13, 5, 8, 16, 15, 7, 4, 12.

2.3. Summary

Table 2.7 Comparison of original and proposed switching activity for different stage

Stage	010	011	100	101	110
Original Switching Activity	139	633	2237	7707	24721
Proposed Switching Activity	0	480	1900	6844	22839
Reduction (%)	100	24.2	15.1	11.2	7.6
Trade off	Proposed multiplier module	6*16 ROM	8*64 ROM	10*256 ROM	12*1024ROM

Table 2.7 shows comparison of original and proposed switching activity for different stage. Stage 100 is the first stage of 256-point FFT. Stage 101 is the first stage of 1024-point FFT. Stage 110 is the first stage of 256-point FFT. In general, the coefficient ordering technique can be used at the complex multiplier of any stage. However, reference [7] shows that the coefficient ordering technique better used in the short-length FFT processor like 16-point FFT or later stage of long-length FFT processor. This is the reason that we choose the stage 010 and 011 to implement this technique.

Table 2.8 Comparison of original and proposed switching activity for different Size of FFT

FFT Size	64	128	256	512	1024	2048	4096	8192
Original Switching activity	1189	3134	6993	16618	35679	79714	167437	360124
Proposed Switching activity	480	1716	4157	10946	24335	57026	122061	269372
Reduction (%)	59.6	45.2	40.6	34.1	31.8	28.5	27.1	25.2

Table 2.8 shows the switching activity of each Size of FFT. From table 2.8, we know that the reduction of power consumption decreases when the size of FFT increases. The report of power consumption is shown in chapter 4.

Because of coefficient ordering technique, we need to reorder the address sequence so that we can obtain the correct data at the correct time. Table 2.9 and Table 2.10 show the switching activity of original and proposed address sequence respectively. We can see that the proposed address sequence almost don’t cause extra power consumption.

Table 2.9 The switching activity of original address sequence

DATA seq.	ADDR1[3:0]	ADDR2[3:0]	ADDR3[3:0]	ADDR4[3:0]
1	0000	0100	1000	1100
2	1100	0000	0100	1000
3	1000	1100	0000	0100
4	0100	1000	1100	0000
5	1101	0001	0101	1001
6	1001	1101	0001	0101
7	0101	1001	1101	0001
8	0001	0101	1001	1101
9	1010	1110	0010	0110
10	0110	1010	1110	0010
11	0010	0110	1010	1110
12	1110	0010	0110	1010
13	0111	1011	1111	0011
14	0011	0111	1011	1111
15	1111	0011	0111	1011
16	1011	1111	0011	0111
Total switching activity	25	25	25	25

Table 2.10 The switching activity of proposed address sequence

DATA seq.	ADDR1[3:0]	ADDR2[3:0]	ADDR3[3:0]	ADDR4[3:0]
1	0000	0100	1000	1100
9	1010	1110	0010	0110
6	1001	1101	0001	0101
14	0011	0111	1011	1111
11	0010	0110	1010	1110
3	1000	1100	0000	0100
2	1100	0000	0100	1000
10	0110	1010	1110	0010
13	0111	1011	1111	0011
5	1101	0001	0101	1001
8	0001	0101	1001	1101
16	1011	1111	0011	0111
15	1111	0011	0111	1011
7	0101	1001	1101	0001
4	0100	1000	1100	0000
12	1110	0010	0110	1010
Total switching activity	25	26	25	26

Chapter 3. Proposed architecture

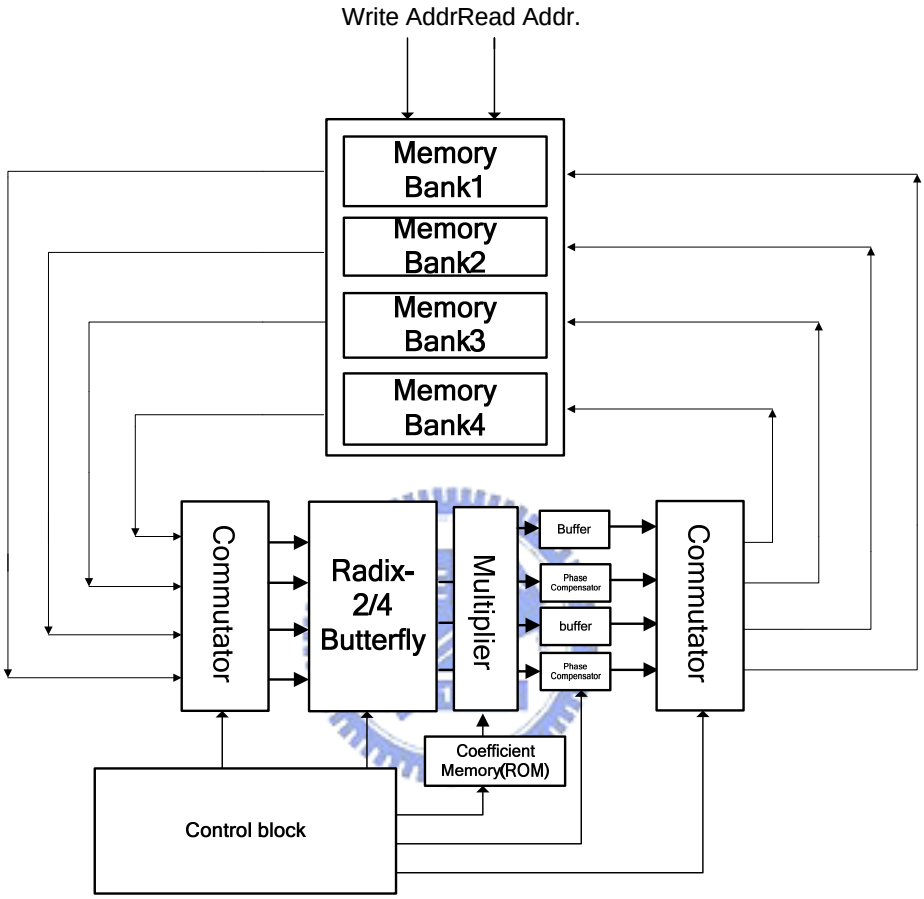


Figure 3.1 The proposed Memory based FFT architecture

Fig. 3.1 shows the proposed Memory based FFT processor. This architecture adopts a dual port memory with four banks because of radix-4 algorithm. The size of Each bank is 32*2048 bit. The first commutator receives the outputs from memory and arranges them to the right inputs of butterfly operator. The butterfly operator and

multiplier module carry out two parallel radix-2 or one radix-4 butterfly computation. Coefficients are provided by the coefficient ROM. The outputs of multiplier module are fed into the phase compensators or buffers. The phase compensators recover the right phase of outputs of multiplier module because of modified coefficients which discuss in chapter 2. The buffers just delay one clock cycle because the four data need to be synchronized. The second commutator receives the outputs from phase compensators and buffers and arranges them to the right inputs of memory.

3.1. Design issue

We discuss some idea of hardware design in this section. Table 3.1 shows the control code for different Size of FFT. In order to achieve simple control circuit, control code adopts four bit instead of three bit. The most significant bit (MSB) of control code is the Radix-2_Flag. When the Radix-2_Flag is high, the radix-2/4 butterfly operator executes radix-2 butterfly computation. The last three bit of control code is related to initial stage, as shown in eq. 3.1. According to Fig. 3.2, when N is equal to 128, 512, 2048 and 8192, butterfly operator need an extra step of radix-2 computation first.

Table 3.1 Different size of FFT and the corresponding control code

FFT Size	64	128	256	512	1024	2048	4096	8192
Control code	0100	1100	0101	1101	0110	1110	0111	1111

$$\text{Initial stage} = \begin{cases} \text{control code}[2{:}0] & \text{,Radix-2\_Flag=1} \\ \text{control code}[2{:}0]\text{-}1 & \text{,Radix-2\_Flag=0} \end{cases} \quad (3.1)$$

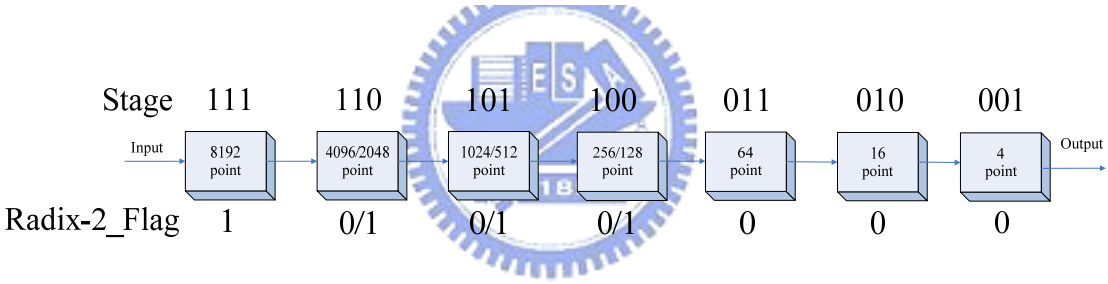


Figure 3.2 Control of signal flow

3.2. Radix-2/4 Butterfly

The proposed radix-2/4 butterfly operator is shown in Fig. 3.3. Radix-4 computation is the common mode. Radix-2 mode is enabled when N is equal to 128, 512, 2048 and 8192. The proposed architecture can execute two radix-2 computations at a time. So the throughput is the double of conventional architecture.

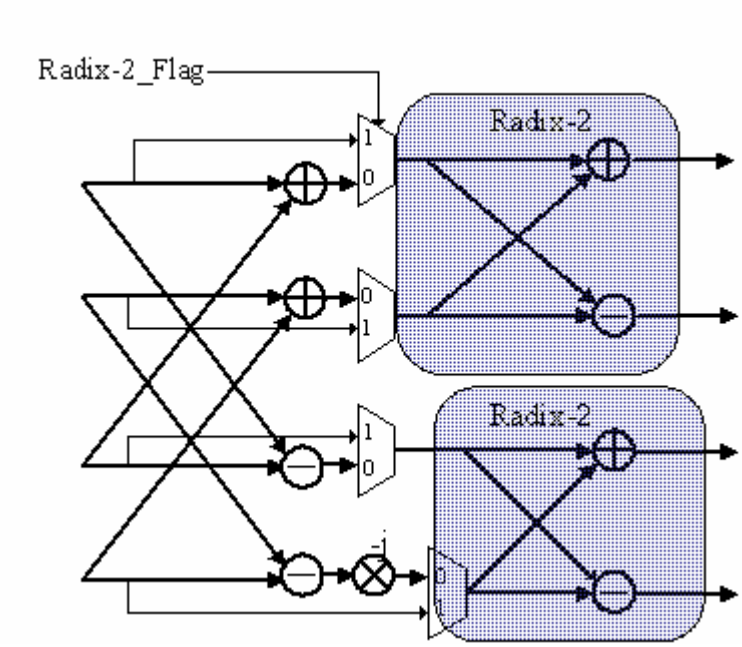


Figure 3.3 Radix-2/4 butterfly operator

3.3. Multiplier module

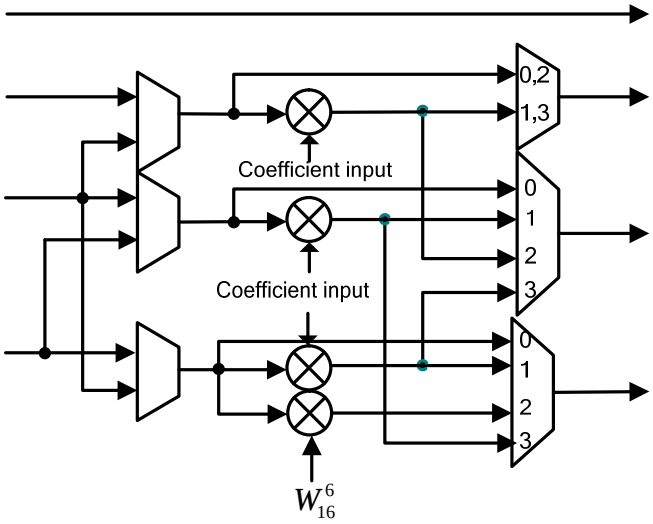


Figure 3.4 The proposed multiplier module

Fig. 3.4 shows the proposed multiplier module. It consists of three complex multiplier and several multiplexers. Exactly as we discussed in chapter 2.2.1,it has no switching activity at stage 010, as shown in Fig. 3.5. Fig. 3.5(a), (b), (c) and (d) represent the state of execution of data sequence 1, 2, 3, and 4 at stage 010, respectively.

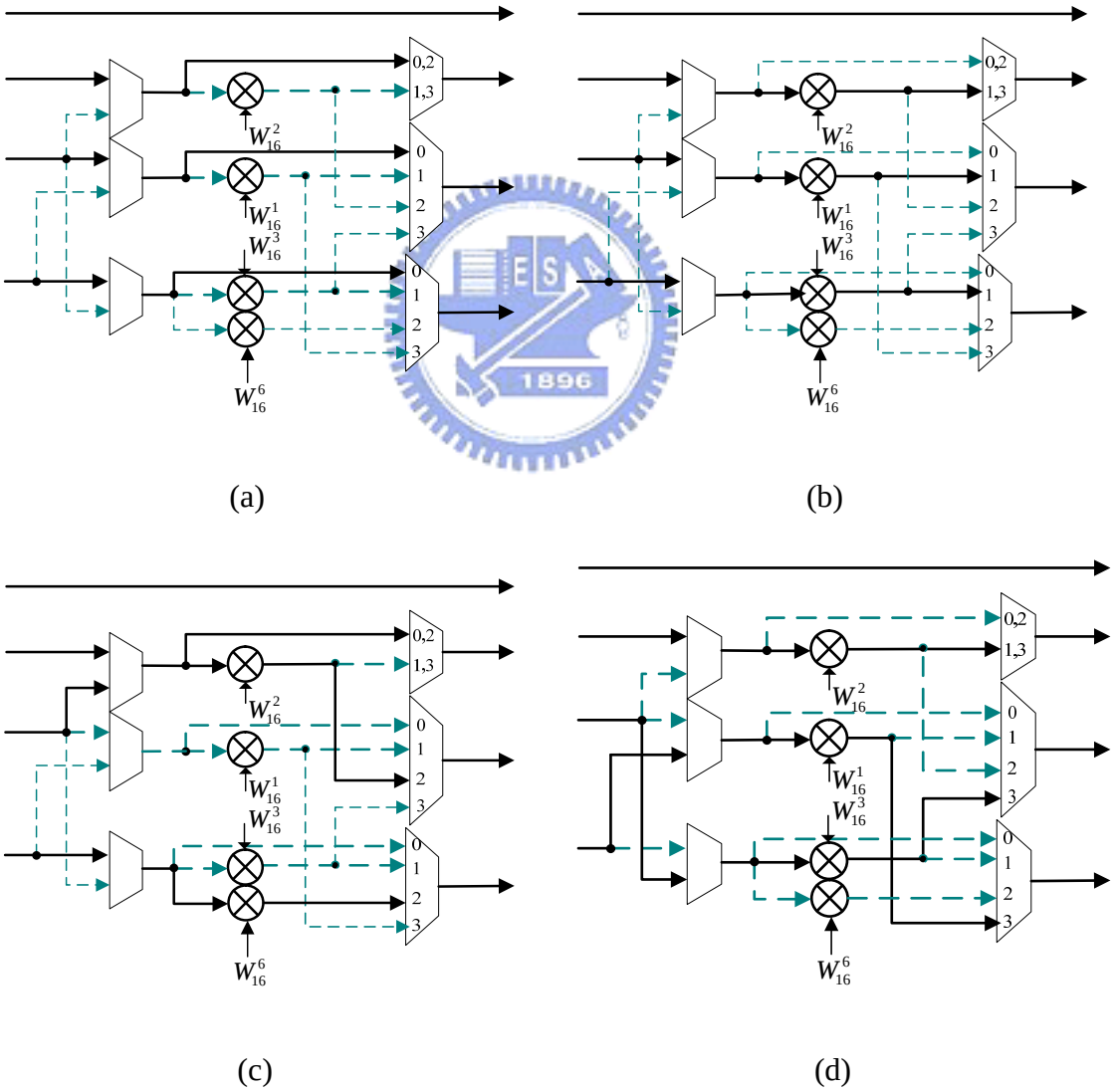


Figure 3.5 The behavior of multiplier module at stage 010

3.4. Phase Compensator

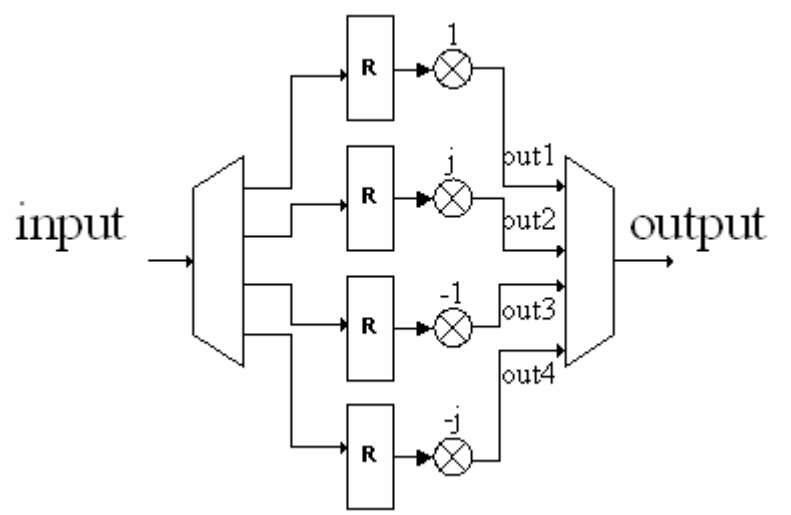


Figure 3.6 Phase compensator

Fig 3.6 shows the architecture of phase compensator. The purpose is to recover the phase of outputs from multiplier module since the modified coefficients are fed into the multiplier module. From eq. 2.3, we assume the modified coefficient, $W_N^{C'}$, is fed into the multiplier module. We derived the output of phase compensator as follows:

Assume $input = X * W_N^{C'}$

$$out1 = input * 1 = X * W_N^{C'}$$

$$out2 = input * j = X * W_N^{C'+3N/4}$$

$$out3 = input * (-1) = X * W_N^{C'+N/2}$$

$$out4 = input * (-j) = X * W_N^{C'+N/4}$$

$$output = X * \{W_N^{C'}, W_N^{C'+N/4}, W_N^{C'+N/2}, W_N^{C'+3N/4}\}$$
$$= X * W_N^{C}$$

3.5. Memory Address Assignment

We adopt the in-place memory addressing scheme for the radix-4 FFT algorithm [8]. For the concurrent read and write operations, the memory is partitioned into four banks. Table 3.2 shows the address assignment for a 16-point FFT.

Table 3.2 Address assignment for a 16-point FFT

SEL					
00	Bank1	0	7	10	13
01	Bank2	1	4	11	14
10	Bank3	2	5	8	15
11	Bank4	3	6	9	12

According to this table, four inputs can be read from different banks and four outputs can be written to different banks for all butterfly computation of 16-poin FFT. SEL is the selection of memory banks. In our design, we use several adders to implement, as shown in eq. 3.2.

```
SEL=addr_counter[1:0]+addr_counter[3:2]
      +addr_counter[5:4]+…
```

(3.2)

This memory assignment strategy can be extended for long-length point FFT. Table 3.3 shows the address assignment for a 64-point FFT. We can see table 3.2 is a sub-block in table 3.3. It means that the memory assignment strategy is adaptable for reconfigurable design.

Table 3.3 Address assignment for a 64-point FFT

Bank1	0	7	10	13	19	22	25	28	34	37	40	47	49	52	59	62
Bank2	1	4	11	14	16	23	26	29	35	38	41	44	50	53	56	63
Bank3	2	5	8	15	17	20	27	30	32	39	42	45	51	54	57	60
Bank4	3	6	9	12	18	21	24	31	33	36	43	46	48	55	58	61

Chapter 4. Simulation and Performance Analysis

In this chapter we discuss simulation and verification with ideal model which is built by MATLAB. The ideal model can provide a complete mathematical and simulation environment. The design flow is illustrated in Fig 4.1, and this is a kind of waterfall models which is worked well up to 100k gate count design.

After RTL code is developed, we verify the ideal model and RTL model to check if they have the same function. There are two ways for implement design after function verification, one is synthesis for ASIC, the other is FPGA prototyping. FPGA prototyping is usually used to verify design, because FPGA can verify the behavior of real hardware. We synthesize the RTL design to Gate-level netlist by reasonable design constrain after the FPGA prototyping. The synthesis report shows the timing, area. We also run Gate-level simulation to double check the function and then run PrimePower by waveform from Gate-level simulation to analyze power consumption.

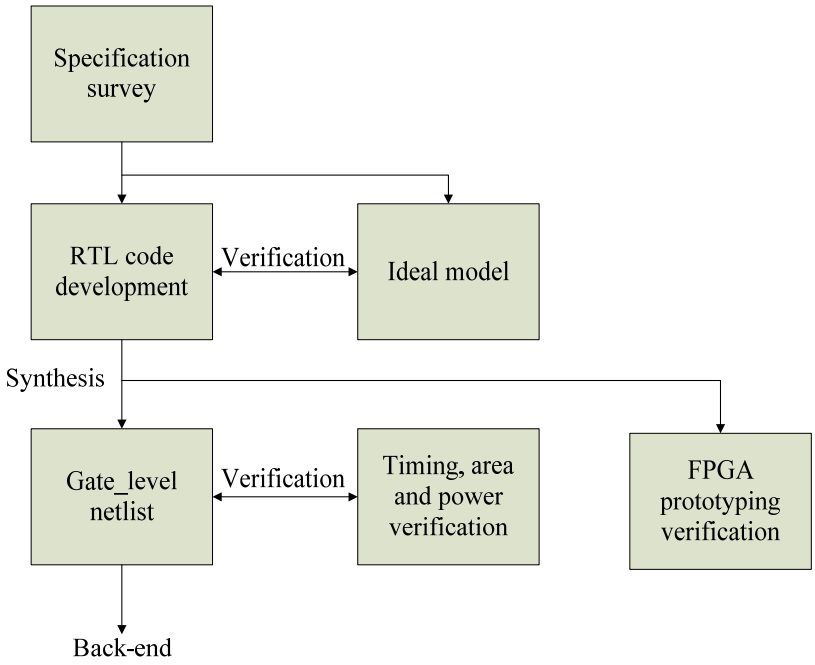


Figure 4.1 Design and verification flow

4.1. Simulation

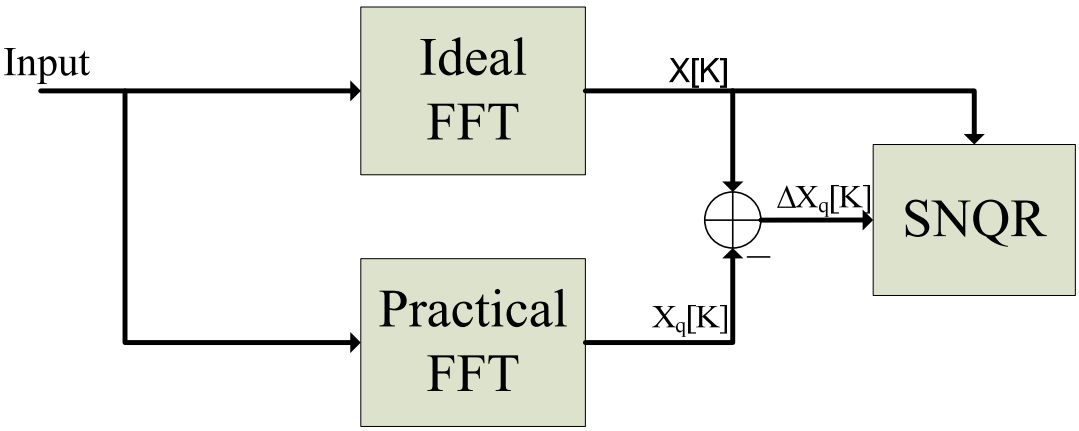


Figure 4.2 Simulation environment

Fig. 4.2 shows the RTL simulation environment that determines the signal to quantization noise ratio (SNQR) between the ideal FFT and a Fixed-point FFT model. Ideal FFT is built by MATLAB and practical FFT is our RTL design. The input data are 100 random patterns with 16 bit word length for each Size of FFT. The definition of SNQR is

$$SNQR = 10 * \log\left(\frac{\sum_{n=0}^{N-1} X(n)^2}{\sum_{n=0}^{N-1} \Delta X_q(n)^2}\right) \quad (4.1)$$

Table .4.1 shows the mean square error for each Size of FFT. Fig. 4.3 is plotted according to table 4.1. We can see that mean square error increases when the size of FFT increases. The SNQR has the same conclusion, as shown in Fig 4.4.

Table 4.1 Mean square error for each size of FFT

FFT size	64	128	256	512	1024	2048	4096	8192
mean square error (10^-4)	1.5	2.29	2.4	3.96	4.3	7.13	8.59	14.1

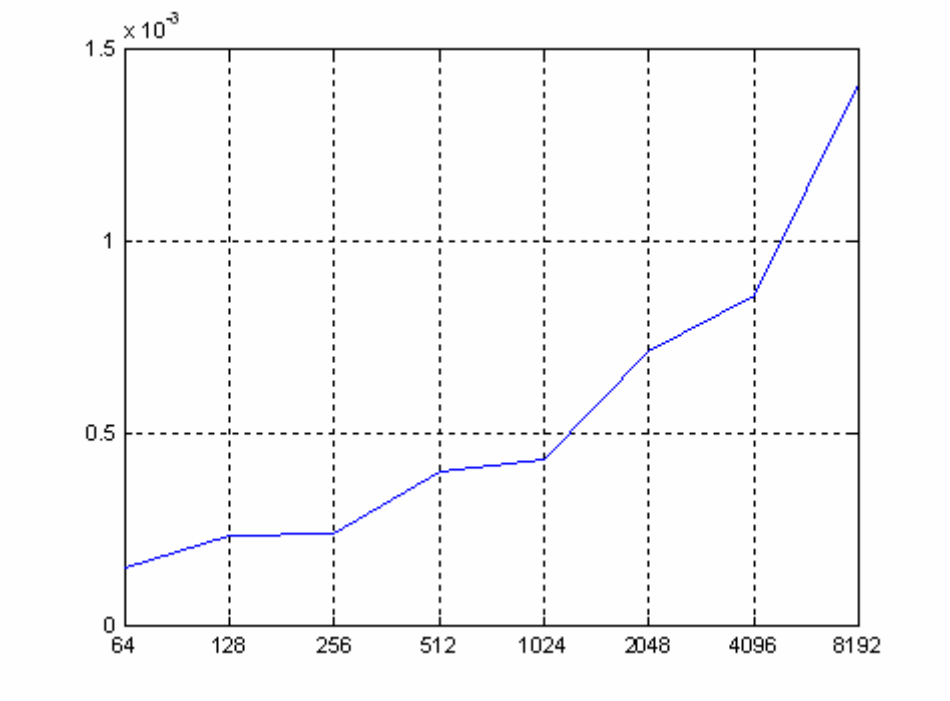


Figure 4.3 Mean square error versus different size of FFT

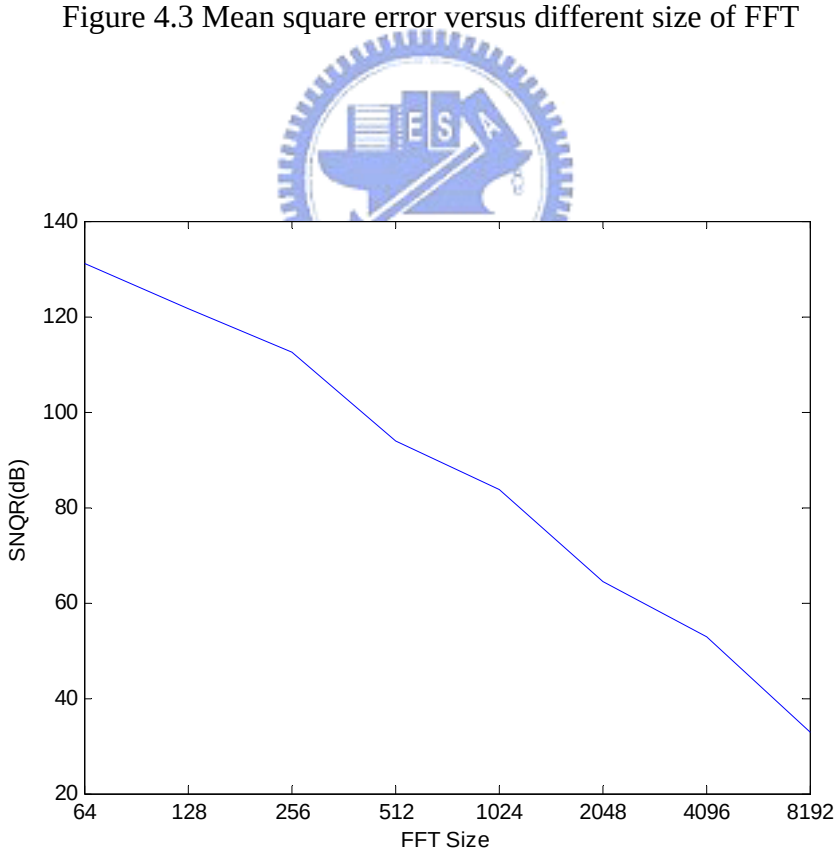


Figure 4.4 SNQR of each Size of FFT

4.2. FPGA prototyping

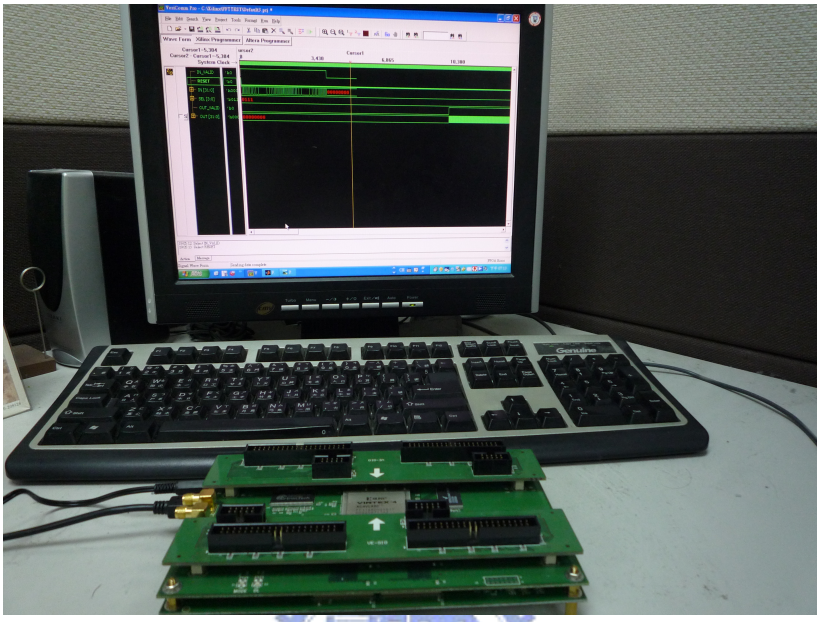


Figure 4.5 XILINX VIRTEX-4 FPGA

Fig. 4.5 shows the used FPGA. It is convenient to verify the design because it can connect to computer through USB. Patterns are fed to FPGA by computer and the results of computation return back to monitor. Fig 4.6 shows the waveform of the design in FPGA. The first four signals are inputs which are made by MATLAB, and the rest are outputs which are delivered to computer by FPGA.

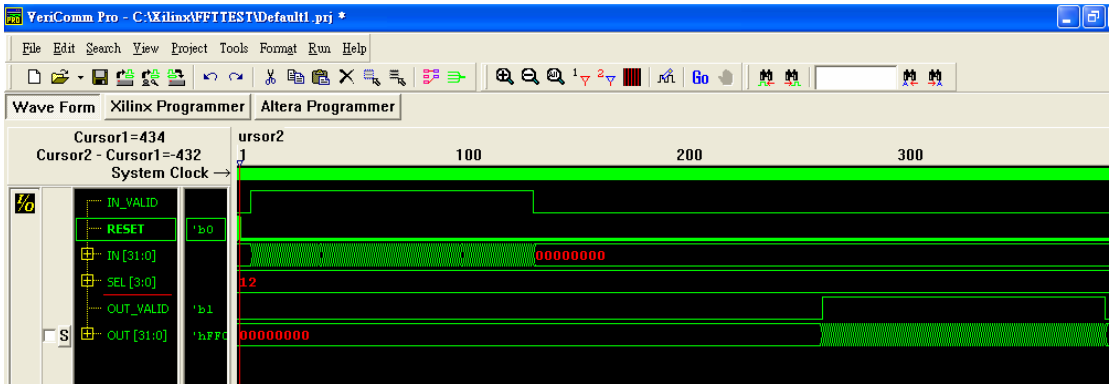


Figure 4.6 Waveform of the proposed design in FPGA

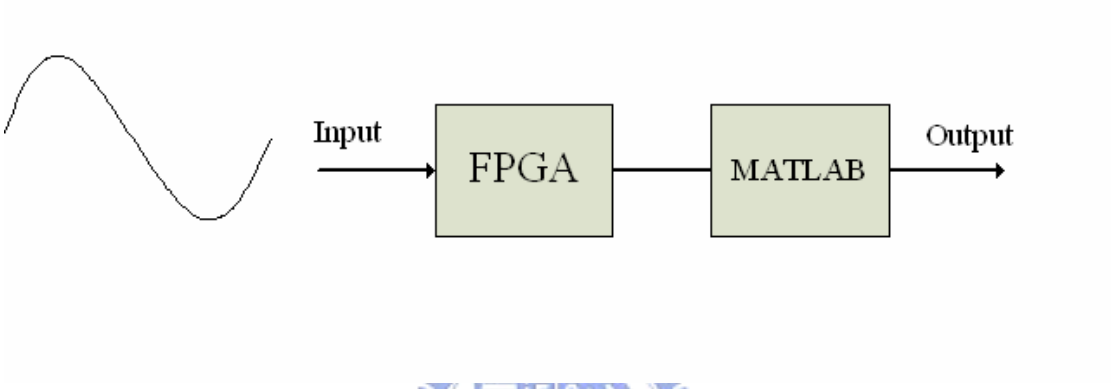
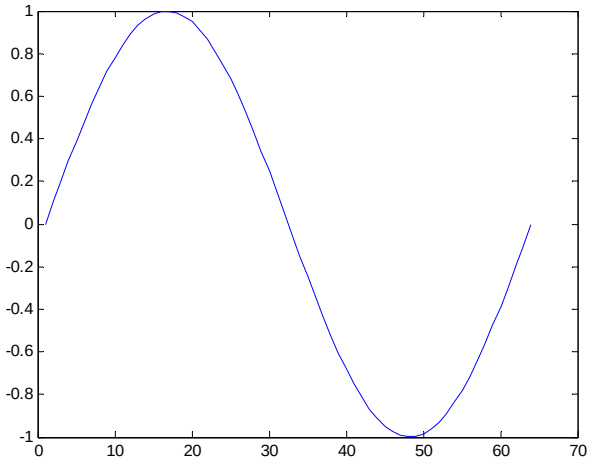


Figure 4.7 Verification flow of FPGA

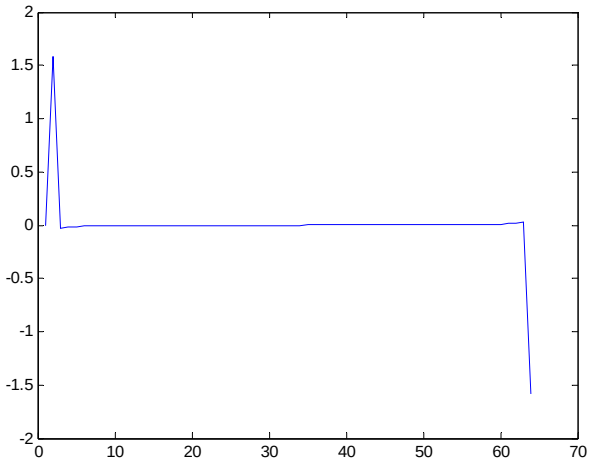
Fig 4.7 shows the verification flow of FPGA. The sine wave which is sampled by N-point is fed to FPGA, and then we export the output file from FPGA to MATLAB. The output file is converted to waveform by MATLAB in order to verification easily. The following figures show the verifications for each Size of FFT according to Fig 4.7.

We feed a sine wave in image part and observe the output from FPGA. We decreased the magnitude of sine wave for long-length point FFT like 4096 and 8192 to avoid overflow. So the figures have a little distortion. Fig 4.16 shows that the synthesis report of proposed architecture from Xiline ISE.

N=64:



Input signal



Real part of output

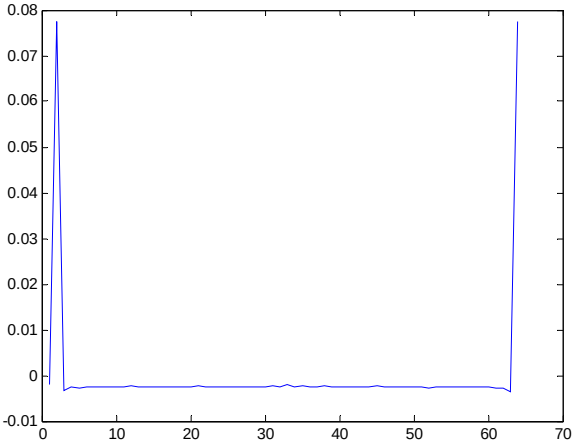
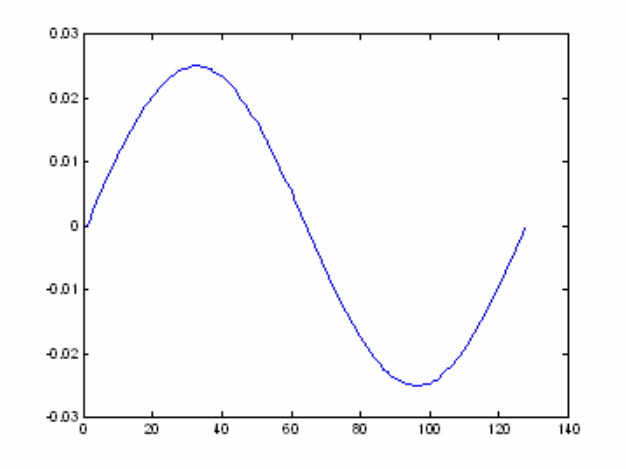


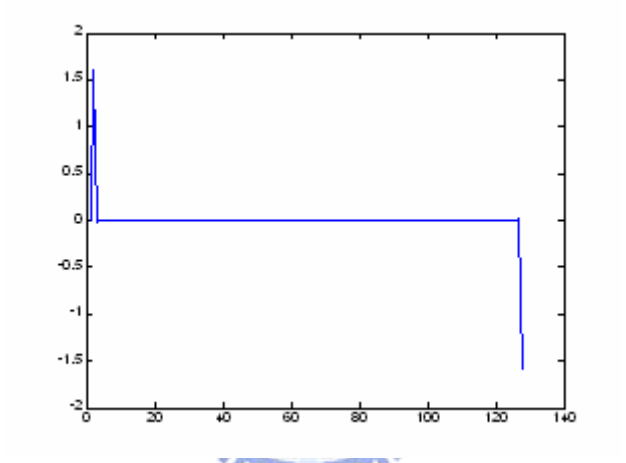
Image part of output

Figure 4.8 Verification of 64-point FFT

N=128



Input signal



Real part of output

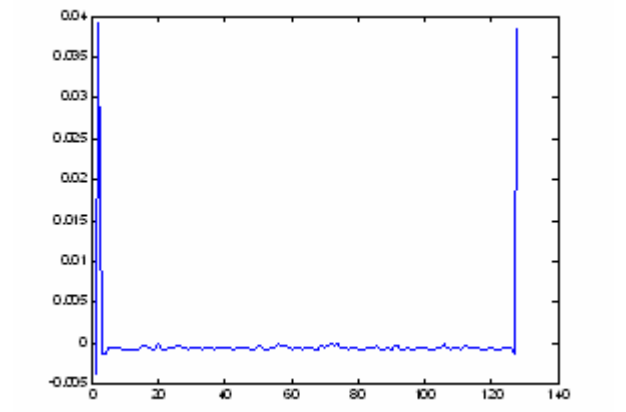
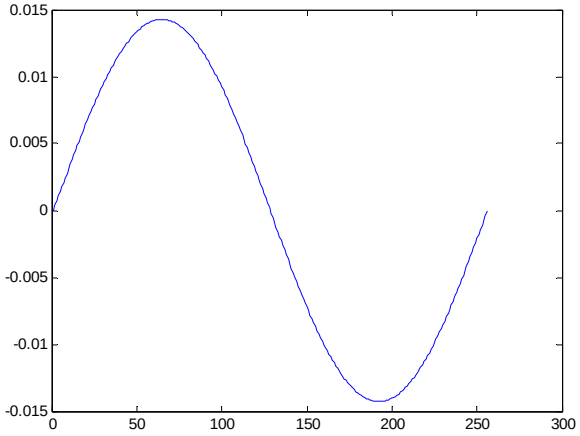


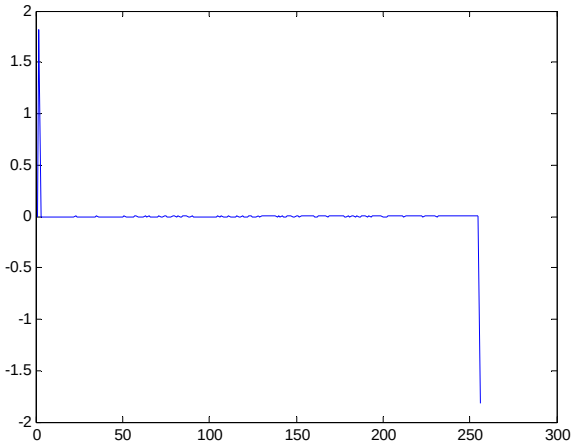
Image part of output

Figure 4.9 Verification of 128-point FFT

N=256



Input signal



Real part of output

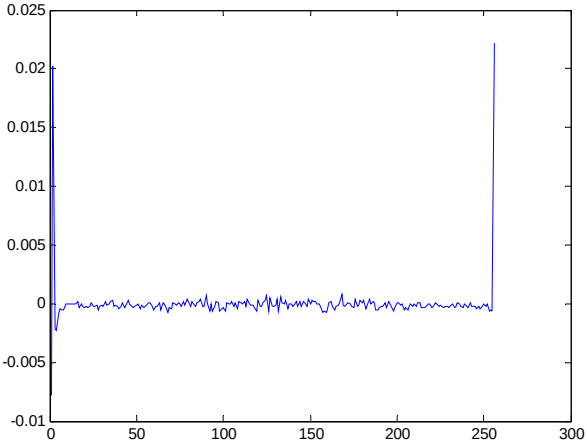
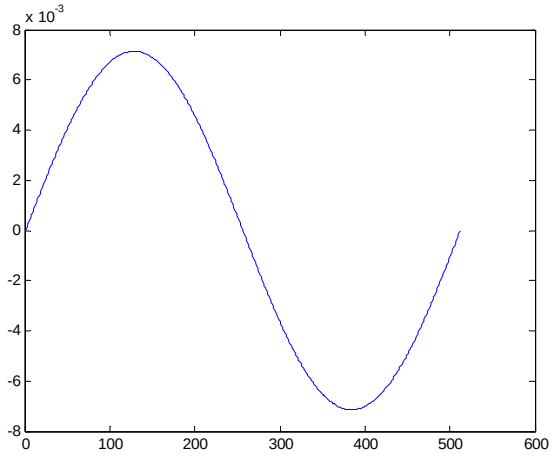


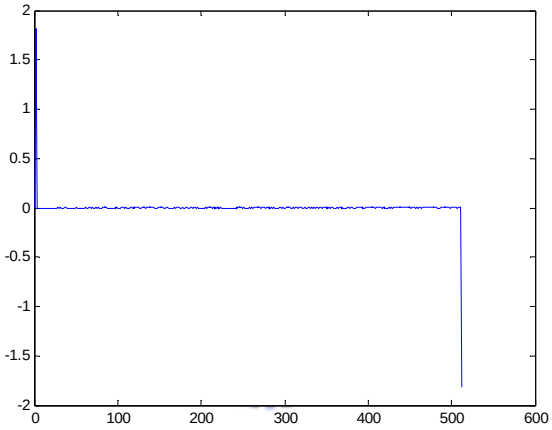
Image part of output

Figure 4.10 Verification of 256-point FFT

N=512



Input signal



Real part of output

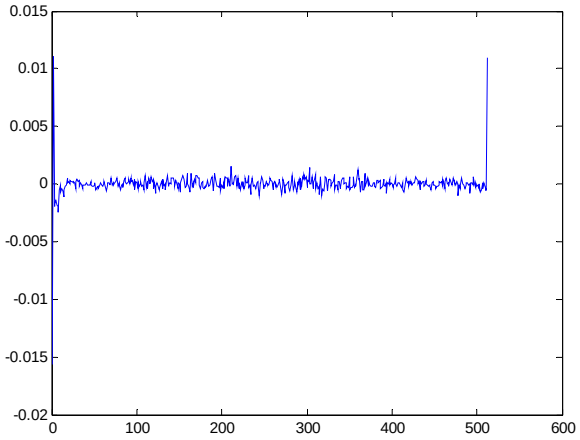
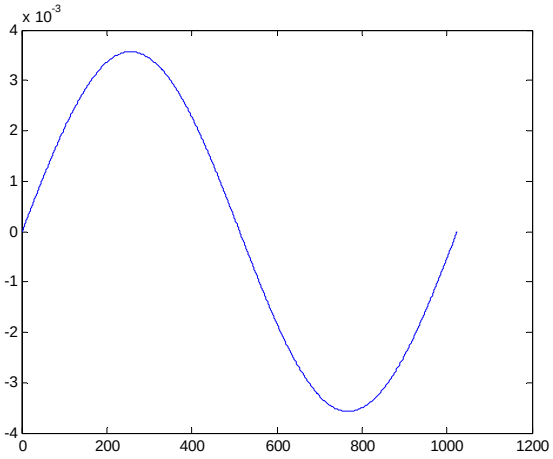


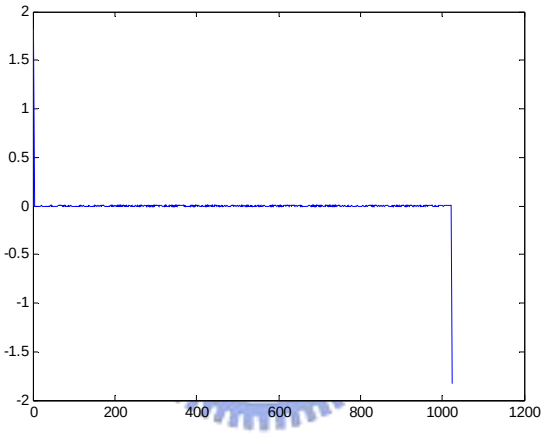
Image part of output

Figure 4.11 Verification of 512-point FFT

N=1024



Input signal



Real part of output

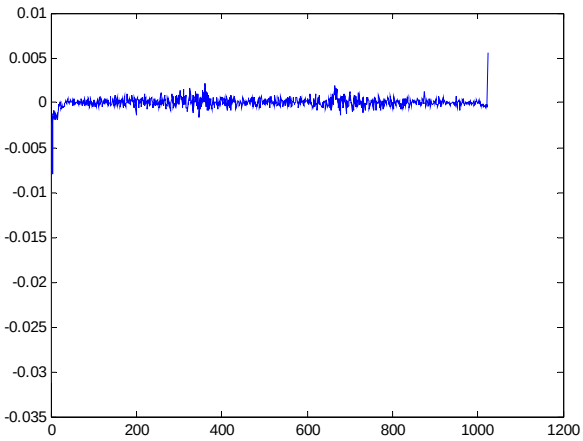
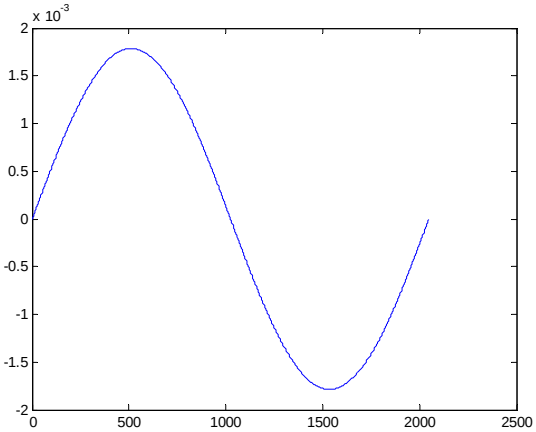


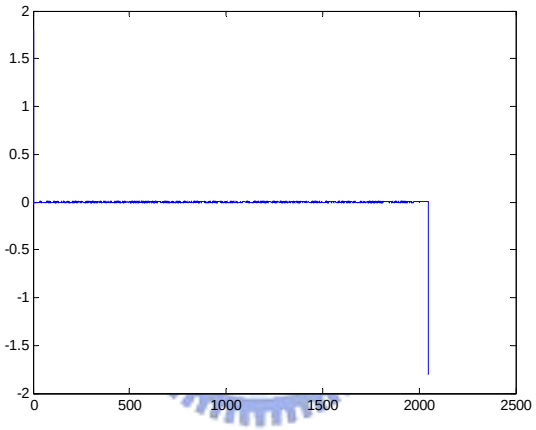
Image part of output

Figure 4.12 Verification of 1024-point FFT

N=2048



Input signal



Real part of output

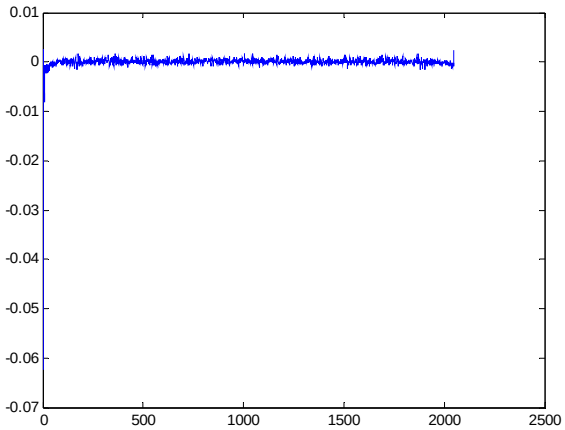
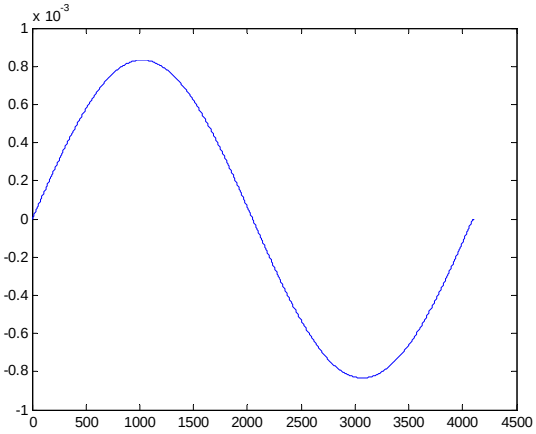


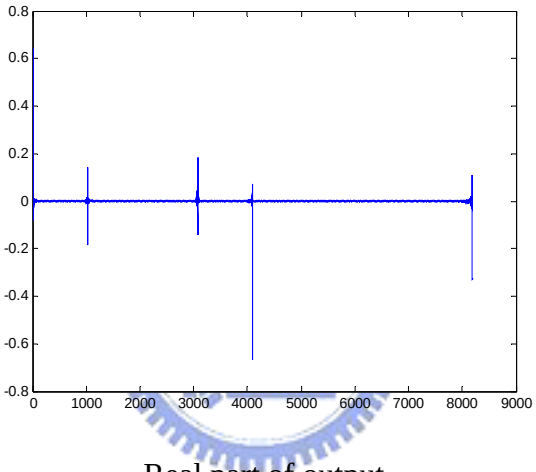
Image part of output

Figure 4.13 Verification of 2048-point FFT

N=4096



Input signal



Real part of output

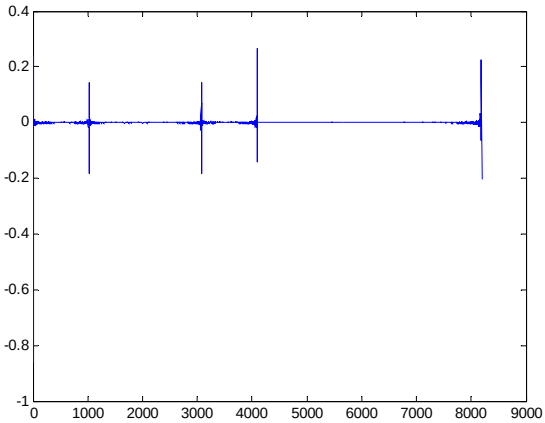
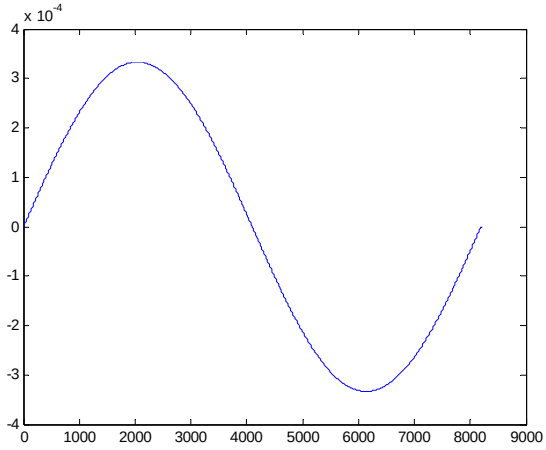


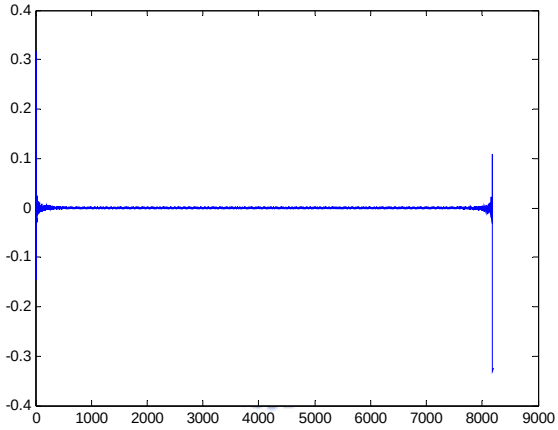
Image part of output

Figure 4.14 Verification of 4096-point FFT

N=8192



Input signal



Real part of output

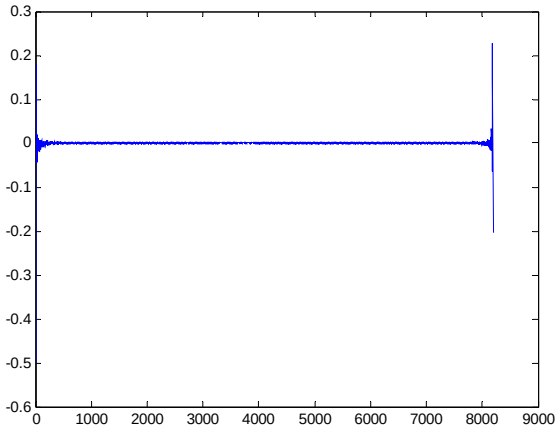


Image part of output

Figure 4.15 Verification of 8192-point FFT

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	1,079	53,248	2%	
Number of 4 input LUTs	14,669	53,248	27%	
Logic Distribution				
Number of occupied Slices	8,603	26,624	32%	
Number of Slices containing only related logic	8,603	8,603	100%	
Number of Slices containing unrelated logic	0	8,603	0%	
Total Number 4 input LUTs	14,714	53,248	27%	
Number used as logic	14,669			
Number used as a route-thru	45			
Number of bonded IOBs	72	640	11%	
Number of BUFG/BUFGCTRLs	1	32	3%	
Number used as BUFGs	1			
Number used as BUFGCTRLs	0			
Number of FIFO16/RAMB16s	16	160	10%	
Number used as FIFO16s	0			
Number used as RAMB16s	16			
Total equivalent gate count for design	115,416			
Additional JTAG gate count for IOBs	3,456			

Fig 4.16 The synthesis report of proposed architecture from Xilinx ISE

4.3. Synthesis Reports and Power Analysis

In this section, we discuss the implementation of the proposed FFT design. The proposed design synthesized to UMC 0.18um CMOS standard cell technology library with Synopsys Design Compiler. The gate count of the proposed architecture is shown in Table 4.2.

Table 4.2 Synthesis report of proposed architecture

	Gate count	Size
Proposed FFT (not include memory)	53306	x
RAM	4*483248	4*32*2048
Coefficient Rom1	6201	32*1024
Coefficient Rom2	10417	32*2048
Coefficient Rom3	6502	32*1024

Table 4.3 shows the power consumptions for each Size of FFT. The report of power consumptions is obtained by the waveform of gate level simulation which is fed to PrimePower. The whole time of power consumption measurement includes the time which data write into memory and the time of computation and the time which data read from memory to output. According to table 2.8, the reduction of switching activity decreases as the increasing size of the FFT. Table 4.3 proves the conclusion of chapter 2.3. Fig. 4.17 is plotted according to table 4.3.

Table 4.3 Power consumption for each Size of FFT

FFT size	64	128	256	512	1024	2048	4096	8192
Power Consumption (mW)	70.11	70.76	73.06	73.71	74.58	74.94	75.82	75.56

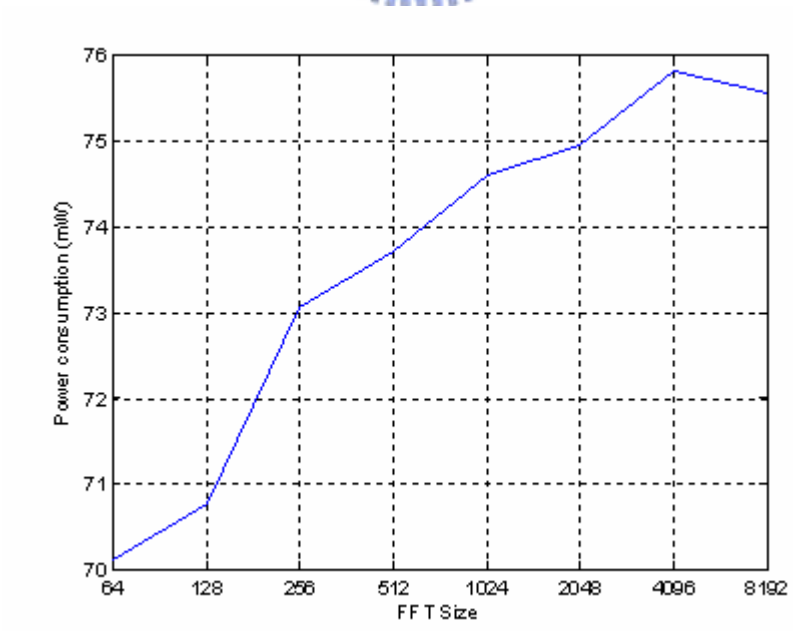


Figure 4.17 Power consumption versus each size of FFT

4.4. Comparisons

The following tables shows the comparisons of gate count, power consumption and latency.

Table 4.4 Comparisons of gate count and power consumption

	Proposed (not include Memory)	Reference[9] (not include Memory)	Reference[10]	Reference[11]
Gate Count	53306 (16 bit)	91000 (8 bit)	245400 (12 bit)	N.A
Power consumption (mW)	75.82 (83Mhz @1.8 V)	190 (12Mhz @2.5 V)	N.A	307.7 (71.4Mhz @1.8 V)

Table 4.5 Comparisons of latency

FFT size	64	128	256	512	1024	2048	4096	8192
Proposed	51	131	259	643	1283	3075	6147	14339
Reference[9]	63	148	276	665	1305	3102	6174	14371
Reference[10]	82	178	386	834	1794	3842	8194	17410

Chapter 5.
Conclusions and Future works

In this thesis, we propose a low power reconfigurable FFT/IFFT processor. The proposed memory based FFT processor can be configured as from 64-point to 8192-point. A low power design with minimum switching activity is proposed. Chapter 4.3 shows that it is efficient for short-length point FFT. The maximum power consumption is 75.82 at power supply 1.8 V. The gate count of the proposed architecture without memory is 53306 under Synopsys Design Complier with UMC 0.18um library.

A low power reconfigurable FFT is presented in this thesis. The minimum switching activity is efficient to reduce the dynamic power consumption. However, this technique restricts the flexible of the FFT processor. In the future, we can try to use the digital signal processor to implement this technique.

Bibliography

[1] A. V. Oppenheim and R. W. Schafer, DISCRETE-TIME SIGNAL PROCESSING, New Jersey, 2nd Edition, Prentice-Hall, 1999.

[2] J.W. CooIey and J.W. Tukey, "An algorithm for the machine calculation of complex Fourier series. *Math. Comp.*, 19:297-301. April 1965.

[3] L. R. Rabiner and B. Gold, Theory and Application of Digital Signal Processing, Prentice-Hall. 1975.

[4] E. H. Wold, A. M. Despain, "Pipeline and Parallel-Pipeline FFT Processors for VLSI Implementation", IEEE Trans. Comput., Vol. 33, no. 5, pp. 414—426, May 1984.

[5] Jen-Ming Wu, and Yang-Chun Fan,""Coefficient Ordering Based Pipelined FFT/IFFT with Minimum Switching Activity for Low Power OFDM Communication", IEEE Int't Symposium on Consumer Electronics, St. Petersburg, Russia, 2006.

[6] "A low-power VLSI architecture for a shared-memory FFT processor with a mixed-radix algorithm and a simple memory control scheme", Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium

[7] M. Hasan, T. Arslan and J.S. Thompson "*A Novel Coefficient Ordering based Low Power Pipelined Radix-4 FFT Processor for Wireless LAN Applications*" IEEE transactions on consumer electronics, Vol. 49, No. 1, Feb. 2003.

[8] L. G. Johnson ‘Conf1ict free memory addressing for dedicated FFT hardware.” IEEE Trans. Circuits Syst. II, vol. 39. pp. 312-316. May 1992.

[9] Xiaojin Li, Zongsheng Lai, Jianmin Cui “A Low Power and Small Area FFT Processor for OFDM Demodulator” IEEE Transactions on Consumer Electronics, Vol. 53, No. 2, MAY 2007

[10] Chin-Long Wey, Wei-Chien Tang, and Shin-Yo Lin “Efficient Memory-Based FFT Architectures for Digital Video Broadcasting (DVB-T/H)” VLSI Design, Automation and Test, 2007. VLSI-DAT 2007.

[11] Guihua Liu,Quanyuan Feng” ASIC Design of Low-power Reconfigurable FFT Processor” ASIC, 2007. ASICON '07.

Vita

姓名：黃謙若

性別：男

出生地：台北市

生日：民國七十三年八月二十日

地址：台北市北投區復興四路 99 號 3 樓

學歷：國立交通大學電子工程研究所碩士班　　2006/09~2008/06

國立中興大學電機工程學系　　2002/09~2006/06

國立師範大學附屬高級中學　　1999/09~2002/06

論文題目：A Low Power Reconfigurable FFT Processor with Minimum Switching Activity

可重組態之低功率快速傅立葉轉換處理器