

A COEFFICIENT MEMORY ADDRESSING SCHEME FOR VLSI IMPLEMENTATION OF FFT PROCESSORS

*M. Hasan and T. Arslan**

Department of Electronics and Electrical Engineering,
The University of Edinburgh,
The King's Buildings, Mayfield Road,
Edinburgh EH9 3JL, Scotland, UK

ABSTRACT

A novel scheme is presented for coefficient address generation in VLSI implementation of FFT processors. The scheme involves manipulation of address lines taking into consideration coefficient addresses required at various FFT stages. We show with the aid of examples that the scheme can lead to more efficient hardware realisations, with significant reduction in hardware for all FFT lengths. This leads to faster, more power and area efficient realisation of FFT processors than approaches published to date. The paper describes the scheme, its implementation in hardware, and presents results showing more than 80 % reduction in area and power for almost all FFT lengths.

1. INTRODUCTION

High performance fast Fourier transform (FFT) processors are widely used in different application areas such as communications, radars, imaging etc. Enhancing the performance of the FFT is a major concern for researchers. Previously, research has concentrated on speed enhancement, however, with the advent of portable computation, area and specially power consumption have become of special interest [1].

Memory addressing is a key research concern for enhancing the performance of FFT processors. To date, the most popular and computationally efficient method of coefficient address generation was proposed by Cohen [2]. According to this

method, address generation occurs through the application of variable shifts to address lines through a cascade of Barrel shifters. The variable shifts are obtained by manipulating the bits of the butterfly counter by using a subtractor. The work in [3] modified the data generation scheme proposed by Cohen while retaining the coefficient address generation method.

This paper presents a novel scheme for coefficient address generation in VLSI implementation of FFT processors. The scheme involves manipulation of address lines taking into consideration coefficient addresses required at various FFT stages. We describe with the aid of examples that the scheme can lead to more efficient hardware realisation, with significant reduction in hardware for all FFT lengths. This leads to faster, more power and area efficient realisation of FFT processors than approaches published to date. The paper describes the scheme, its implementation in hardware, and presents results showing more than 80 % reduction in area and power for almost all FFT lengths.

2. IMPLEMENTATION

The N-point discrete Fourier Transform (DFT) is defined by

$$X_k = \sum_{m=0}^{N-1} x_m \cdot W_N^{mk}$$

where $W_N = \exp(-j2\pi/N)$ and $k=0,1,\dots,N-1$. The radix-2 FFT, shown in Figure 1 for $N=16$, is an efficient way to compute an N-point Discrete Fourier Transform (DFT). It has been assumed that

* Dr. T. Arslan is also associated with the Institute of System level Integration in Livingston, Scotland

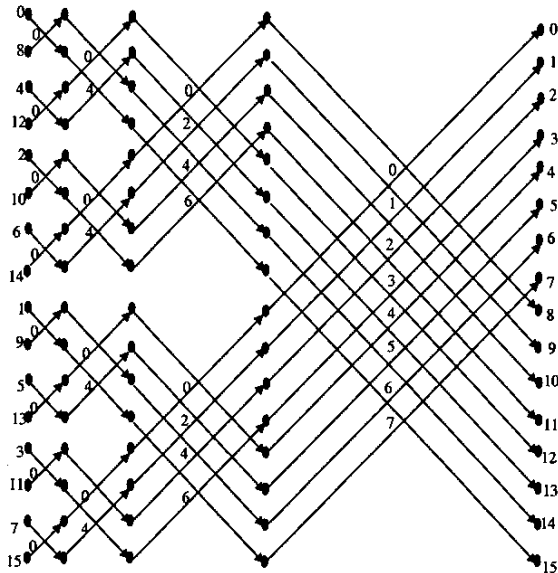


Figure 1: Signal flowgraph of a 16-point FFT as an example

the data inputs are arranged in bit reverse order and the outputs are produced in normal order. The basic operations in FFT are multiplication of the complex data inputs by the FFT coefficients ($\exp(-j2\pi k/N)$) at each stage in the signal flowgraph. This is followed by their summation or subtraction and associated data and coefficient address generation. Coefficient and data address generation logic are required to be fast, area and power efficient in order to realise fast, miniaturised and low power FFT processors which could be integrated into complex VLSI systems. In this paper, the coefficient address generation in an FFT is accomplished by partitioning a k -bit counter into two sections as shown in Figure 2. The more significant counter section comprises of (C_{k-1}, \dots, C_b) bits whereas the lower section has (C_{b-1}, \dots, C_0) bits. Let the more and less significant counter section bits be represented by a $(k-b)$ bit array x and a b -bit array y respectively. The coefficient address (CA) is then expressed as follows.

$$CA = F(x, y) \quad (1)$$

Let us also assume that 'i' is the decimal equivalent of x , which is always greater than or equal to 'b' depending upon the FFT size.

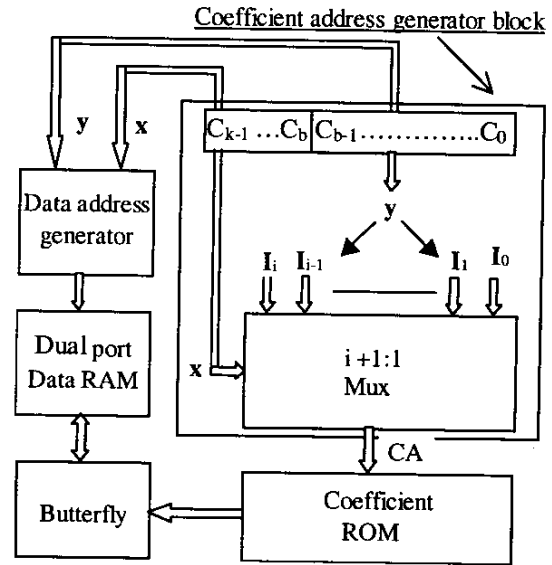


Figure 2: Block Diagram of an FFT processor based on our scheme

Hence, (1) can be written as follows,

$$CA = I_i,$$

where I_i is a b -bit array, which is expressed in terms of an array y according to the following set of equations.

$$\begin{aligned} I_0 &= (0 \dots 0) \\ I_1 &= (y[b-1]0 \dots 0) \\ I_2 &= (y[b-1] y[b-2] 0 \dots 0) \\ &\vdots \\ I_{b-1} &= (y[b-1] y[b-2] y[b-3] \dots y[1] 0) \\ I_b &= y \\ I_{b+1} &= (X \dots X) \\ &\vdots \\ I_i &= (X \dots X) \end{aligned} \left. \vphantom{\begin{aligned} I_0 \\ I_1 \\ I_2 \\ \vdots \\ I_{b-1} \\ I_b \\ I_{b+1} \\ \vdots \\ I_i \end{aligned}} \right\} \text{Not required}$$

Where N = Size of the FFT,
 $b = \log_2(N/2)$,
 X = Don't care bit
 $k = b + \{\log_2(\log_2(N))\}$ (rounded up to the nearest integer)

The above set of operations could be realised in hardware by using a single multiplexer based coefficient address generation logic as shown in

Figure 2. The input channels of the multiplexer are set according to I_i and selected by x .

In order to illustrate this scheme, consider a 16-point radix-2 FFT example shown in Figure 1. In the first stage, only the first coefficient with an address value $0(W^0)$, is required for all the eight butterflies. In the second stage, W^0 and W^4 coefficients are required. In the third stage, W^0 , W^2 , W^4 and W^6 coefficients are needed whereas in the last stage all the coefficients from W^0 to W^7 are required for the butterfly operations.

In this example, the coefficient memory has eight locations for storing coefficients (W^0 to W^7) and hence the lower counter section should comprise of only three bits C_2 , C_1 and C_0 . The more significant counter section, which tracks the four stages of the 16-point FFT, will be having only two bits namely C_4 and C_3 .

It is clear from Figure 1 that in the first stage, the coefficient address remains equal to "000"(only W^0) irrespective of the lower section counter value. The first stage is represented by the "00" combination of the higher section bits and hence the input channel of the multiplexer corresponding to this bit combination must always remains at "000". In the second stage, the address "000"(W^0) is required for the first four computed butterflies and address "100"(W^4) is required for the next four butterflies. The second stage "01" of the FFT is dependent on the status of the C_2 bit of the counter. This means that the second channel input should be set to " C_200 ". The third stage "10" of the FFT requires the generation of four different addresses "000"(W^0), "010"(W^2), "100"(W^4) and "110"(W^6). It is clear that these addresses differ only in the C_2C_1 combination and C_0 remains equal to zero always. It means that the third channel of the multiplexer should be connected to " C_2C_10 " to accomplish this task. The last stage "11" needs all the coefficients and hence the last channel must be connected directly to " $C_2C_1C_0$ ".

The same technique can be very easily extended to any FFT size by following the formulations described earlier. It is evident that the only change in hardware for longer FFT's is either the introduction of some new input channels or at most a larger multiplexer if the FFT length is much

longer than the existing length.

3. RESULTS

The scheme has been implemented in register transfer level Verilog hardware description language for different FFT lengths and then synthesized using Cadence *BuildGates* with 0.35u Alcatel MTC 45000 CMOS technology library. Power evaluation was carried out using Synopsys *DesignPower* for the circuit netlist.

Simulations were carried out for one million clock cycles using a supply voltage of 3.3V and a clock frequency of 100 MHz. The same procedure was followed for Cohen's approach and the comparative results, in terms of area and power, for the different FFT lengths are given in Figure 3 and Figure 4 respectively. The power consumption, given in Figure 3, is the average power consumed by the respective circuit per clock cycle. It is clear from Figure 5 that our approach results in power and area savings of 80% to 90% when compared to Cohen's approach for the whole spectrum of FFT lengths. In Figure 3, the area is slightly smaller for the coefficient address generation logic for a 2K-point FFT as compared to 1K-point in a Cohen's approach on account of the optimisation of the subtractor with one fixed input by the synthesis tool.

It is evident from Figure 4 that the power consumption associated with the coefficient address logic depends on the switching activity of the inputs and outputs and not on the FFT size. The delay through our logic is only 1.13ns whereas Cohen's approach has a delay of 9.88ns under identical input/output load conditions. This implies that our logic can operate at frequencies up to 885 MHz whereas Cohen's logic is limited to frequencies below 101 MHz.

4. CONCLUSION

This paper has presented an effective coefficient addressing scheme for VLSI implementation of FFT processors. We have shown that the scheme can result in VLSI hardware that are faster, more power and area efficient than the existing approaches. The scheme maintains an improvement

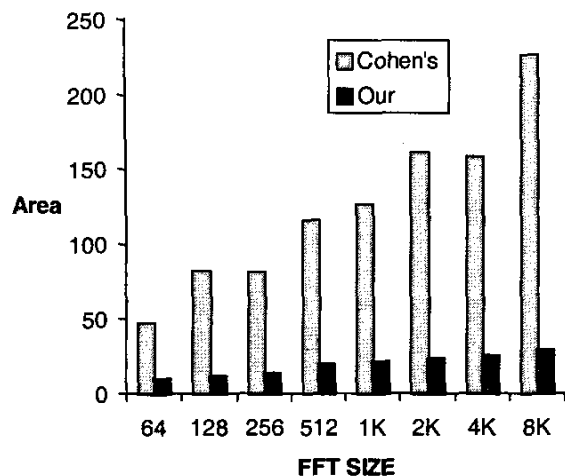


Figure 3: Comparison of the two approaches in terms of area expressed in terms of the area of a NAND gate.

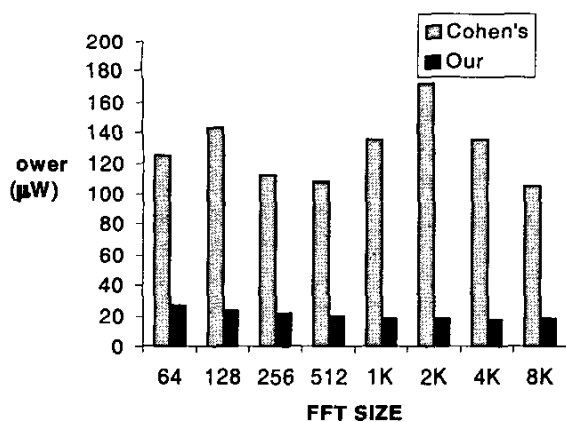


Figure 4: Comparison of the two approaches in terms of power.

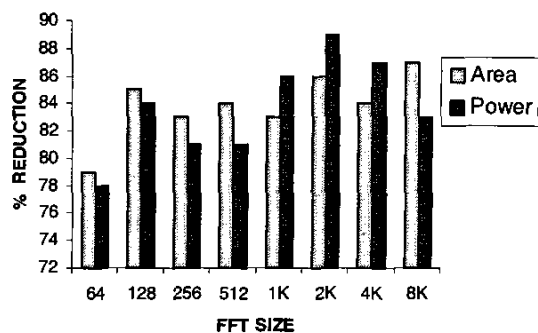


Figure 5: Percentage reduction in area and power of our approach over Cohen's approach.

of more than 80% over existing addressing schemes with almost all FFT lengths. The scheme will lead to the design of faster and more power and area efficient systems with embedded FFT processors.

5. REFERENCES

- [1] A.P. Chandrakasan and R.W. Brodersen, "Low Power Digital CMOS Design" Kluwer Academic Publishers, 1995.
- [2] D.Cohen, "Simplified Control of FFT hardware", IEEE transactions on acoustics, speech and signal processing, vol. ASSP-24, pp. 577-579, Dec. 1976.
- [3] Y.Ma and L. Wanhammar, "A Hardware efficient control of memory addressing for high performance FFT processors", IEEE transactions on signal processing, vol. 48, no. 3, pp. 917-921, March, 2000.