A Novel Approach in Pipeline Architecture for 64-Point FFT Processor without ROM

A. Manimaran, Dr. S. K. Sudheer, Manu. K. Harshan
Associate Professor, Department of ECE, Karpaga Vinayaga College of Engineering and Technology
Chennai, India

M. E. Embedded System Technologies, Department of ECE, Karpaga Vinayaga College of Engineering and Technology, Chennai, India

Abstract – FFT processor is an important unit in modern wireless communication system. So more research and developments take place in this field. The paper reports low power efficient implementation of FFT processor. Proposed architecture in the design is single-path delay feedback (SDF) pipeline architecture. The requirement of memory and utilization of multipliers is comparatively less in this architecture so that this architecture is very efficient for low power and smaller area FFT designs that is mainly using in portable DSP devices. Proposed architecture completely eliminates the use of ROM by using a reconfigurable complex multiplier and bit-parallel multipliers. Symmetric property of twiddle factor is also used in the proposed multiplier to get low power.

INTRODUCTION

In digital signal processing (DSP) discrete fourier transform (DFT) is a very important technique. For telecommunication, particularly for orthogonal division multiplexing (OFDM)[2] systems fast fourier transform is a critical block. Time complexity (O(N^2)) and computational difficulty of DFT increase the FFT. FFT is an efficient method to reduce the time complexity to O(N log_2 N) which was proposed by Cooley and Turkey[5]. Here N denotes FFT size.

Implementation of hardware in various papers[6]-[10] is mainly classified into memory based and pipeline architecture style. Mainly memory based and pipeline architecture is adopted to design FFT processor. Design method composed of a main single processing element (PE) and several memory units. The hardware cost and power consumption of this kind of architecture style is lower. But its disadvantage is long latency, long throughput and it cannot be parallelized. Pipeline architecture can get rid off the disadvantage of the above architecture.

In pipeline architecture each stage of FFT using separate arithmetic unit. This approach increase the throughput by a factor of log_2 N when different units are pipelined. This architecture is also known as cascade FFT architecture and will be used in our proposed design.

Pipeline FFT processors have two popular design types. One uses asingle-path delay feedback (SDF) pipeline architecture[6]-[7], and the other uses a multiple-path delay commutator (MDC) pipeline architecture. The single-path delay feedback (SDF) pipeline FFT is good in its requiring less memory space (about N-1 delay elements) and its multiplication computation utilization being less than 50%, as well as its control unit being easy to design. Such implementations are advantageous to low-power design, especially for applications in portable DSP devices. Because of these reasons SDF pipeline architecture is adopted in our design.

FFT computation need to multiply input signals with different twiddle factors, which result in more hardware cost because large size ROM is required and it also increase the area. Designing a FFT processor without ROM we can increase the performance and also can reduce the area. Commonly using word length complex multipliers increase the cost so we are using complex multiplier realized with shift and add operation. The architecture design also use of the symmetric property of twiddle factor[1][3].

The rest of this paper is organized as follows. Abrief review of the fast Fourier transform is described in Section II. In Section III presents our proposed FFT
architecture. In section IV simulation and result. In section V concluding remarks has given.

ILFFT ALGORITHM

The DFT of an N-point discrete-time signal is defined by:

\[ X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \quad 0 \leq k \leq N-1 \quad (1) \]

Where the coefficient \( W_N^{kn} \) (also called the twiddle factor) is a complex number given by:

\[ W_N^{kn} = e^{-j\frac{2\pi kn}{N}} \quad (2) \]

Straight implementation of this algorithm is impractical because large number of multipliers is required for its implementation. So FFT algorithm is required for its efficient implementation and to reduce the hardware cost. Generally, FFT analyses an input signal sequence by using a decimation-in-frequency (DIF) or decimation-in-time (DIT) decomposition to construct an efficiently computational signal-flow graph (SFG).

Decimation in Frequency algorithm This algorithm decomposes even and odd-indexed frequency samples as shown mathematically in equation sets (3) & (4).

\[
X(2k) = \sum_{n=0}^{N/2-1} x(n) W_N^{kn} W_N^{k+N/2n} \\
= \sum_{n=0}^{N/2-1} x(n) W_N^{kn} W_N^{k+nN/2} \\
= \text{DFT}_{N/2} [x(0:2:N-1)] \\
X(2k+1) = \sum_{n=0}^{N/2-1} x(n) W_N^{kn} W_N^{k+N/2n+1} \\
= \text{DFT}_{N/2} [x(0:2:N-1)] \\
\]

(3)

(4)

An example of radix-2 DIF FFT SFG for \( N = 8 \) is shown in Fig. 1.

From figure we can analyse that some complex multiplication can be simplified to reduce the chip area and to avoid ROM. Input signal multiplied by \( W_N^k \) in figure can be expressed as:

\[ (a + jb) W_N^k = [(a + b) + j(b - a)]/\sqrt{2} \quad (5) \]

Where \( (a + jb) \) denote discrete-time signal in complex form. In similar manner complex multiplication of \( W_N^k \) is given by:

\[ (a + jb) W_N^k = [(b - a) - j(a + b)]/\sqrt{2} \quad (6) \]

Both these equations are required for hardware implementation. Multiplication by \( 1/\sqrt{2} \) can be obtained easily by using the bit-parallel multiplier explained in the later section.

III. PROPOSED ARCHITECTURE

By considering symmetry of twiddle factors we can reduce the complexity of complex multiplication. Complex multiplication in an FFT must be one of the type given below:

\[ W_N^k (a + jb) = W_N^{k+N/4} (b - ja), \quad \frac{N}{4} < k < N \quad (7) \]

\[ W_N^k (a + jb) = -W_N^{k-N/4} (a + ja), \quad \frac{N}{4} < k < 3N/4 \quad (8) \]
Twiddle factors are generating using cosine and sine functions. Therefore using all the values of sine and cosine function coming between 0-\pi/4 the complex multiplication with twiddle factors can be done.

The proposed architecture is composed of three different types of processing elements (PEs), a complex constant multiplier, delay-line (DL) buffers (as shown by a rectangle with a number inside). The proposed architecture uses single path delay feedback. A reconfigurable complex constant multiplier is used to eliminate the twiddle factor ROM. Thus the new multiplication structure becomes the important component in reducing the area and hardware cost. The proposed architecture is shown in Fig. 2.

PE3 stage is used to implement a simple radix-2 butterfly structure only, and serves as the submodules of the PE2 and PE1 stages. In the figure, \(I_{in}\) and \(I_{out}\) are the real parts of the input and output data, respectively. \(Q_{in}\) and \(Q_{out}\) denote the image parts of the input and output data, respectively. Similarly, \(DL_{I_{in}}\) and \(DL_{I_{out}}\) stand for the real parts of input and output of the DL buffers, and \(DL_{Q_{in}}\) and \(DL_{Q_{out}}\) for the image parts.

respectively. The working of processing element 3 (PE3) is as follows and illustrated in Fig. 3. When \(S_0 = 0\), \(DL_{I_{in}} = I_{in}, DL_{I_{out}} = I_{out}\). When \(S_0 = 1\), \(DL_{I_{in}} = DL_{I_{out}} + (I_{in}), DL_{I_{out}} = I_{out} + \langle -DL_{I_{out}}\rangle\). In PE2 stage, we need to perform the multiplication by \(-1\). In PE2 stage, it is required to compute the multiplication by \(-j\) or \(1\). Note that the multiplication by \(-1\) in Fig. 4 is practically to take the 2’s complement of its input value.

In the PE1 stage, the calculation is more complex than the PE2 stage, which is responsible for computing the multiplications by \(-jW_N^{N/2}\) and \(W_N^{N/4}\). But we have seen \(W_N^{N/2} = -jW_N^{N/2}\) can be given by either the multiplication by \(-j\) or the reverse of the previous calculation. Hence, the designed hardware utilizes this kind of cascaded calculation and multiplexers to realize all the necessary calculations of the PE1 stage. This manner can also save a bit-parallel multiplier for computing, which further forms a low-cost hardware.
In section-II multiplication by $\frac{1}{\sqrt{2}}$ can be employ by bit-parallel multiplier. The bit-parallel operation in terms of power of 2 is given by

$$\text{output} = I_n \times \frac{1}{\sqrt{2}} = I_n (2^{-2} + 2^{-3} + 2^{-4} + 2^{-5})$$

(12)

If a straightforward implementation for the above equation is adopted, it will introduce a poor precision due to the truncation error, and will spend more hardware cost. Therefore, to improve the precision and hardware cost, Eq. (12) can be rewrite as:

$$\text{output} = I_n \times \frac{1}{\sqrt{2}} = [1 + [1 + 2^{-2}] [2^{-4} + 2^{-5} - 2^{-7}]]$$

(13)

the circuit diagram of bit-parallel multiplication is illustrated in the fig. 6. The complex multiplication by $w_N^{\pi/8}$ is realised as shown in fig. 7 respectively.

Multiplication by $w_N^{\pi/8}$ is done by a reconfigurable complex constant multiplier. Structure of this complex multiplier also adopts a cascaded scheme to achieve low-cost hardware. Structure is as illustrated in the figure. Circuit in fig. 8 is responsible for the multiplication of $w_N^{\pi/8}$ in the proposed architecture that is shown in the fig. 2.

Fig. 4 PE2 circuit diagram

Fig. 5 PE1 circuit diagram

Fig. 6 Circuit diagram of the bit-parallel multiplication by $\frac{1}{\sqrt{2}}$

Fig. 7 Circuit diagram of the multiplication by $w_N^{\pi/8}$

Fig. 8 Proposed reconfigurable complex constant multiplier.
The multiplier in fig.9 is responsible for the twiddle factor complex multiplication in the reconfigurable complex multiplier shown in fig.8. The coefficient values $f_1$ and $f_2$ are listed in table I. The twiddle factors in our proposed design is generating using the values in the table.

**TABLE I**

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>value</th>
<th>Coefficient</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>0.7071</td>
<td>$f_2$</td>
<td>0.7071</td>
</tr>
<tr>
<td>$f_2$</td>
<td>0.7730</td>
<td>$f_3$</td>
<td>0.6343</td>
</tr>
<tr>
<td>$f_4$</td>
<td>0.8314</td>
<td>$f_5$</td>
<td>0.5555</td>
</tr>
<tr>
<td>$f_5$</td>
<td>0.8819</td>
<td>$f_6$</td>
<td>0.4713</td>
</tr>
<tr>
<td>$f_6$</td>
<td>0.9238</td>
<td>$f_7$</td>
<td>0.3826</td>
</tr>
<tr>
<td>$f_7$</td>
<td>0.9569</td>
<td>$f_8$</td>
<td>0.2902</td>
</tr>
<tr>
<td>$f_8$</td>
<td>0.9807</td>
<td>$f_9$</td>
<td>0.1950</td>
</tr>
<tr>
<td>$f_9$</td>
<td>0.9951</td>
<td></td>
<td>0.0980</td>
</tr>
</tbody>
</table>

**IV.SIMULATION RESULTS**

Simulation of 64-point FFT was described in VHDL and Simulation was done in modelsim and the code was functionally verified to be correct.
From the synthesised power report the following result is obtained. The total dynamic power only 2.1875mW and cell leakage power only 43.6134uW.

V. CONCLUSION

This approach using without ROM and low-power pipeline FFT for OFDM applications have been described in this paper. Considering the symmetric property of twiddle factors in FFT, we have designed a reconfigurable complex constant multiplier such that the size of twiddle factor ROM is significantly shrunk, especially no ROM is needed in our work. By using proposed structure there should be significant reduction in area and hence power. So the proposed architecture can be used in portable DSP devises.

REFERENCES