Abstract—With the aim of minimizing memory and latency, this letter presents a novel bit-reversal architecture for continuous-flow parallel pipelined FFT processors. It harnesses the theory that any permutation can be decomposed to a series of elementary bit-exchanges. The main contribution of this letter are twofold. First, it achieves continuous-flow bit reversal in parallel with the minimum memory and minimum latency. Second, the architecture, composed of memory and 2-to-1 multiplexers, are simple and regular for general power-of-2 parallelism. Furthermore, it supports different common radices, including radix-2, radix-4, and radix-8.

Index Terms—Bit-reversal architecture, continuous-flow parallel architecture, fast fourier transform (FFT).

I. INTRODUCTION

F

The Bit Reversal function block is to move the data with index position to efficient reordering based on indexed bit-exchanges. This means that bit-reversal can be implemented by a cascade of basic circuits for bit-exchanges.

Let us consider \( N = 2^n \) indexed data and denote its index as \( I \equiv b_{n-1}, \ldots, b_1, b_0 \), wherein \( n \) is a non-negative integer and the symbol \((\equiv)\) is the binary representation of index. The bit-reversing function block is to move the data with index \( I = b_{n-1}, \ldots, b_1, b_0 \) to the place \( HR(I) \equiv b_0, b_1, \ldots, b_{n-1} \). This phase can be decomposed to a series of bit-exchanges between \( b_j \) and \( b_k \), where \( j \) and \( k \) are non-negative integers. Therefore, it is a key point to efficiently reorder data based on indexed bit-exchanges between \( b_j \) and \( b_k \).

\[
I_A = b_{n-1}, \ldots, b_j, 0, b_{j-1}, \ldots, b_{k+1}, 1, b_{k-1}, \ldots, b_0
\]

\[
I_B = b_{n-1}, \ldots, b_j, 1, b_{j-1}, \ldots, b_{k+1}, 0, b_{k-1}, \ldots, b_0.
\]

Note that only samples for which \( b_j \neq b_k \) should be exchanged their positions, namely that sample with index \( I_A \) should be swapped position with one with index \( I_B \). To maintain the continuity of data, memory is necessary for swapping data and the depth of memory \( I \) for bit-exchange is \( 2^j - 2^k \).
According to this,[15] proposed the basic single-path circuit for bit-exchange, as shown in Fig. 1. The control signal $S$ can be generated by only AND operations as follows

$$S = b_j \& \overline{b_k}.$$  

(3)

By cascading the basic circuit for bit-exchange, bit-reversing circuits can be derived and the total memory $M_{total}$ used in circuits, equivalent to latency $L_{total}$, can also be calculated as follows.

$$M_{total} - L_{total} = \begin{cases} \frac{(\sqrt{N} - 1)^2}{2} & \text{for even } n, \\ \frac{(2\sqrt{N} - 1)(\sqrt{\frac{N}{2}} - 1)}{2} & \text{for odd } n. \end{cases}$$

(4)

In summation,[15] has proposed a novel circuit to efficiently implement bit-reversal permutation whereas it is dedicated for single-path design. For high throughput optimum circuits for parallel bit reversal should be designed to simultaneously permute the multiple outputs of parallel pipelined FFT.

III. OPTIMUM APPROACH FOR PARALLEL BIT-EXCHANGE PERMUTATION

For parallel pipelined bit-reversing, it is challenging to permute among multi-path data flows and generate multiple outputs continuously [17]. Based on the algorithms in [18], once the basic circuits for parallel pipelined bit-exchanges are proposed, the whole architecture can be obtained by simply cascading these basic circuits. In this section, the circuits for parallel continuous-flow bit-exchanges are designed.

Consider the size of indexed data as $N = 2^n$ and the parallelism of data flows as $P = 2^q$. The range of $P$ is limited to no more than $2\sqrt{N}$, where $r$ is the radix, i.e. $r = 2$ for radix-2.

If $P > \frac{2\sqrt{N}}{r}$, the parallelism is so high that most bit-exchanges are performed by simply swapping data between different paths. This situation rarely occurs in most applications and thereby is not discussed in this letter. The index of input data and output data is respectively represented as (5) and (6). And the index of data path is represented by the least low $q$ bits.

$$I \equiv b_{n-1}, \ldots, b_{n-q}, b_{n-q-1}, \ldots, b_q, b_{q-1}, \ldots, b_0$$

Path index

$$BR(I) \equiv b_0, \ldots, b_{q-1}, b_q, \ldots, b_{n-q-1}, b_{n-q}, \ldots, b_{n-1}.$$  

Path index

(5)

(6)

Let $\sigma_i$ represents the bit-exchange between $b_i$ and $b_{n-i-1}$ as follows

$$\sigma_i(b_{n-1}, \ldots, b_{n-i}, b_{n-i-1}, \ldots, b_{n-1}, b_i, \ldots, b_j) = b_{n-1}, \ldots, b_{n-i}, b_{n-i-1}, b_{n-1}, b_i, \ldots, b_0$$

(7)

which can be also represented as follows

$$\sigma_i : b_i \leftrightarrow b_{n-1-i}.$$  

(8)

Observes that bit reversal from (5) to (6) is composed of $[n/2]$ bit-exchanges $\sigma_i$, where $i \in [0, [n/2] - 1]$. Let us define $k = i$ and $j = n - 1 - i$. The bit-exchanges can be discussed on two cases: bit-exchange $\sigma_i$ within each path where $i \in [q, [n/2] - 1]$ and bit-exchange $\sigma_i$ cross the paths where $i \in [0, q - 1]$.

A. Bit-Exchange within Each Path

For one parallel bit-exchange $\sigma_i$ when $i \in [q, [n/2] - 1]$, it can be performed by just swapping position between the data with index $I_A^i$ and the data with index $I_B^i$. It is obvious that $P_A^i = P_B^i$ and thereby each path just permutes data inner the path.

$$I_A^i = b_{n-1}, \ldots, b_{j+1}, 0, b_{j-1}, \ldots, b_{k+1}, 1,$$

$$b_{k-1}, \ldots, b_q, b_{q-1}, \ldots, b_0.$$  

$$P_{A}^i$$

$$I_B^i = b_{n-1}, \ldots, b_{j+1}, 1, b_{j-1}, \ldots, b_{k+1}, 0,$$

$$b_{k-1}, \ldots, b_q, b_{q-1}, \ldots, b_0.$$  

$$P_{B}^i$$

Consequently, $P$ number of single-path circuits in Fig. 1 are respectively adopted for each path. But the depth of memory for each circuit is $2^{n-1-i} - 2^i$ instead of $2^{n-1-i} - 2^i$. As a result, the summation of memory for $P$ paths stays $2^{n-1-i} - 2^i$. Correspondingly, the control signals $S$ of $P$ paths can be generated by a single counter $C$, where $C = c_{n-1-q}, \ldots, c_1, c_0$ and $S = c_{n-1-i-q} \& c_{n-i-q}$. When $S = 0$, the positions of data are not changed. On the other hand, when $S = 1$, it swaps data as the rule mentioned above.

B. Bit-exchange Cross the Paths

As for one bit-exchange $\sigma_i$ where $i \in [0, q - 1]$, it swaps position between the data with index $I_A^i$ and the data with index $I_B^i$. Note that $P_A^i \neq P_B^i$ and data in path $P_A^i$ are only swapped with data in path $P_B^i$ if necessary.

$$I_A^i = b_{n-1}, \ldots, b_{j+1}, 0, b_{j-1}, \ldots, b_q,$$

$$b_{k-1}, \ldots, b_{k+1}, 1, b_{k-1}, \ldots, b_0.$$  

$$P_{A}^i$$

$$I_B^i = b_{n-1}, \ldots, b_{j+1}, 1, b_{j-1}, \ldots, b_q,$$

$$b_{k-1}, \ldots, b_{k+1}, 0, b_{k-1}, \ldots, b_0.$$  

$$P_{B}^i$$

This means that $P$ paths can be divided into $P/2$ pairs in which data interchange their positions and thereby the basic two-parallel circuit for bit-exchange when $i \in [0, q - 1]$ is the key point of the design.

We propose a simple but efficient circuit to perform one bit-exchange $\sigma_i$ between $P_A^i$ path and $P_B^i$ path as shown in Fig. 2.

The depth of memory $L$ in Fig. 2 is $2^{n-1-i} - 2^i$ and thereby the total memory for this bit-exchange permutation is $2^{n-1-i}$, which is $2^i$ more than that for single-path implementation. In addition, the control signals $S$ of all $P/2$ pairs are also controlled by a single counter $C$, where $S = c_{n-1-i-q}$.
IV. THE OPTIMUM ARCHITECTURE FOR PARALLEL BIT REVERSAL

This section cascades basic bit-exchange circuits to achieve bit reversal and extends this architecture to common radices.

A. Bit-Reversal Architecture

The bit reversal consists of \(|n/2|\) bit-exchanges and the order of bit-exchanges has no influence on the results. In terms of the regularity and simplicity, we do bit-exchanges \(\sigma_i\) as the sequence \(i = |n/2| - 1, \ldots, 0\). Without loss of generality, the architecture of 8-parallel pipelined bit reversal is illustrated as an example in Fig. 3.

Fig. 3 shows that the architecture is divided into two parts, corresponding to the algorithm in Section III. In part I, one single-path element (SPE) stands for the single-path circuit in Fig. 1. Here one bit-exchange \(\sigma_i\) is accomplished by \(P\) number of SPEs that are distributed in \(P\) paths and the control signals of them are generated by one and the same \(\log_2(N/P)\)-bits counter. In part II, the symbol \(\bigcirc\) in Fig. 3 stands for the two-path circuit in Fig. 2. For one bit-exchange, it contains \(P/2\) number of two-path circuits and all control signals of them are controlled by one bit of a \(\log_2(N/P)\)-bits counter as Section III discussed.

From the Fig. 3, we can find that the switch scheme of data-flows for bit-exchanges in part II is very similar to that of FFT architecture. It is not surprising at all for the reason that the bit-reversing order is generated just by FFT architecture. In addition, the architecture of \(P = 2\) and \(P = 4\) for part II is also illustrated in Fig. 3. Through this regularity, higher parallelism can also be obtained. As for the case of single-path [15], i.e. \(P = 1\), no bit-exchanges in part II occur and the architecture only contains one path of SPEs, which is also generalized in the proposed architecture.

To clarify the parallel processing of bit reversal, Table I shows the timing diagram of each bit-exchange output for \(P = 4\) and \(N = 64\). The input data arrives in bit-reversed order and are listed below the column of input data. Firstly, the bit-exchange \(\sigma_2\) permutes input data within each path and outputs them as the sequence below the column of \(\sigma_2\). Subsequently, the bit-exchange \(\sigma_1\) permutes the output of \(\sigma_2\). It divides the four paths into two pairs, i.e. \(\{p_0, p_2\}\) and \(\{p_1, p_3\}\), and each pair interchanges its data by the circuit in Fig. 2. The output of \(\sigma_1\) is listed below \(\sigma_1\) column. Finally, the bit-exchange \(\sigma_0\) permutes data like \(\sigma_1\) but the difference is that the paths of data are divided to \(\{p_0, p_1\}\) and \(\{p_2, p_3\}\). After three pairs of bit-exchanges, namely \(\sigma_2, \sigma_1,\) and \(\sigma_0\), the output in nature order is generated. In Table I, the memory depth \(I\) of each path and the control signal \(S\) for each bit-exchange are shown as well.

B. Memory and Latency

In this subsection, the total memory and latency are calculated for the proposed architecture. Compared with single-path bit reversal, additional \(2^l\) memory is consumed for each bit-exchange \(\sigma_i\), where \(i \in \{0, q - 1\}\). Consequently, the summation of additional memory is \(\sum_{i=0}^{q-1} 2^i = P - 1\).

Based on the analysis for single-path bit reversing in (4), the final memory of parallel bit reversal \(M_{total}\) is

\[
\left\{ \begin{array}{ll}
(N - 2\sqrt{N} + P)/P & \text{for even } n, \\
(N - \sqrt{2N} - \sqrt{N}/2 + P)/P & \text{for odd } n.
\end{array} \right.
\]

And the latency \(L_{total} = M_{total}/P\) is

\[
\left\{ \begin{array}{ll}
((N - 2\sqrt{N} + P))/P & \text{for even } n, \\
((N - \sqrt{2N} - \sqrt{N}/2 + P))/P & \text{for odd } n.
\end{array} \right.
\]

The results in (9) and (10) are just the theoretical minimum memory and minimum latency of parallel continuous-flow bit reversal. They can be easily derived by the method in [11], namely calculating the max time interval between the moment of input sample and that of output sample.

C. Extension to Other Radices

The bit reversal mentioned above are all about radix-2 FFT architecture. In many applications, radix-4 and radix-8 FFT architectures are adopted for the saving of multiplications. The order of the output data has some differences between different radices.

In general, the bit reversal of any radix-\(r\) \((BR_r)\) groups \(\log r\) bits of the index and performs a reversal of these groups. All of them are essentially composed of \(|n/2|\) bit-exchanges and the only difference is the mapping of bit-exchanges \(\sigma_i\).

For radix-\(r\), it can be generally represented as follows

\[
\sigma_i : b_i \leftrightarrow b_{n - 1 - i \log r + \text{mod}(1, \log r)}
\]

Analogously, these bit-exchanges are discussed on two cases: bit-exchange \(\sigma_i\) within each path where \(i \in [q, |n/2| - 1]\) and bit-exchange \(\sigma_i\) cross the paths where \(i \in [0, q - 1]\). The circuits
TABLE I
TIMING DIAGRAM OF EACH BIT-EXCHANGE OUTPUT FOR PARALLEL BIT REVERSAL

<table>
<thead>
<tr>
<th>Time (cycles)</th>
<th>Input Data</th>
<th>Part I</th>
<th>Part II</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>σ₁₂ : b₃ ↔ b₂ (L = 1)</td>
<td>σ₁₂ : b₄ ↔ b₁ (L = 4)</td>
<td>σ₀ : b₅ ↔ b₀ (L = 8)</td>
</tr>
<tr>
<td>p₀</td>
<td>p₁</td>
<td>p₂</td>
<td>p₃</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>32</td>
<td>16</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>40</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>36</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>44</td>
<td>28</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>34</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>42</td>
<td>26</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>38</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>14</td>
<td>46</td>
<td>30</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>33</td>
<td>17</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>41</td>
<td>25</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>37</td>
<td>21</td>
</tr>
<tr>
<td>11</td>
<td>13</td>
<td>49</td>
<td>29</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>35</td>
<td>19</td>
</tr>
<tr>
<td>13</td>
<td>11</td>
<td>43</td>
<td>27</td>
</tr>
<tr>
<td>14</td>
<td>7</td>
<td>39</td>
<td>23</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>47</td>
<td>31</td>
</tr>
<tr>
<td>16</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>17</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>18</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>19</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>20</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>21</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>22</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>23</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>24</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>25</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>26</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>27</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>28</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

TABLE II
COMPARISONS OF SEVERAL BIT-REVERSAL CIRCUITS

<table>
<thead>
<tr>
<th>Supported</th>
<th>Memory Size (words)</th>
<th>Throughput</th>
<th>Latency (cycles)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallelism</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[4]</td>
<td>8</td>
<td>PN</td>
<td>P</td>
</tr>
<tr>
<td>[17]</td>
<td>2/4/8</td>
<td>N</td>
<td>P</td>
</tr>
<tr>
<td>[15]</td>
<td>1</td>
<td>M_{[15]}^{(2)}</td>
<td>1</td>
</tr>
<tr>
<td>This letter</td>
<td>2^{α} (3)</td>
<td>M_{[19]}^{(4)}</td>
<td>P</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. L_{[17]}^{(1)} = \frac{\alpha}{P} - \frac{\sqrt{2\alpha \frac{P}{N}}}{\sqrt{2P}} + 1 for even n and \frac{\alpha}{P} - \frac{\sqrt{2\alpha \frac{P}{N}}}{\sqrt{2P}} + 1 for odd \alpha, where \alpha = \log_2 \frac{3P}{N}.
2. M_{[15]}^{(2)} = N - 2\sqrt{N} + 1 for even n and N - \sqrt{N} - \sqrt{2N} + 1 for odd n. And L_{[15]}^{(2)} = M_{[15]}^{(2)}.
3. n \in [0, \log_2 \frac{2\sqrt{N}}{P}]
4. M_{[19]}^{(4)} = N - 2\sqrt{N} + P for even n and N - \sqrt{N} - \sqrt{2N} + P for odd n. L_{[19]}^{(4)} = M_{[19]}^{(4)}, that is \frac{\alpha}{P} - \frac{\sqrt{\alpha \frac{P}{N}}}{\sqrt{2P}} + 1 for even n and \frac{\alpha}{P} - \frac{\sqrt{\alpha \frac{P}{N}}}{\sqrt{2P}} - \frac{\sqrt{\alpha \frac{P}{N}}}{\sqrt{2P}} + 1 for odd n.

VI. CONCLUSION

In this letter, a new parallel bit-reversing architecture is proposed for parallel pipelined FFT with the aim of minimizing memory and latency. The proposed architecture is efficient because it requires lowest memory and achieves lowest latency in theory. Furthermore, this architecture is simple, regular and scalable for higher parallelism and common radices.

This paper previously published in IEEE Signal Processing Letters.
REFERENCES


