FPGA Implementation of Low-Power Split-Radix FFT Processors

Zhuo Qian, Nasibeh Nasiri, Oren Segal
University of Massachusetts Lowell
Lowell, MA, USA
{Zhuo_Qian, Nasibeh_Nasiri, Oren_Segal}@student.uml.edu

Martin Margala
University of Massachusetts Lowell
Lowell, MA, USA
Martin_Margala@uml.edu

Abstract—Fast Fourier Transform (FFT) is one of the fundamental operations in digital signal processing area. Split-radix Fast Fourier Transform (SRFFT) approximates the minimum number of multiplications by theory among all the FFT algorithms, therefore SRFFT is a good candidate for the implementation of a low power FFT processor. In this PhD work, we aim to implement a novel low power Split-Radix FFT processor using shared-memory architecture and extend this work to a parallel structure based on FPGA. We started by designing a new radix-2 butterfly unit using clock gating approach to block unnecessary switching activity in the multiplier. Compared to existing SRFFT processors which are based on the “L” shaped butterfly, our implementation simplifies the address generation process for FFT data. Furthermore, because the number of multiplications required by SRFFT algorithm significantly decreases as the FFT size increases, it is reasonable to assume the proposed architecture will save more power when it comes to larger points of FFT.

Keywords—Split-Radix FFT; Low Power; Parallel Architecture

I. MOTIVATION

Split-Radix FFT algorithm was developed by Duhamel and Hollman [1] in 1984. The idea behind their algorithm is the application of a radix-2 index mapping to the even-index terms and a radix-4 mapping to the odd-index terms, which results in an “L” shaped butterfly. Their algorithm requires least number of multiplications and additions among all the FFT algorithms with input length N equals to $2^m$ (m is any natural number). Since arithmetic operations significantly contribute to overall system power consumption, SRFFT is a good candidate for the implementation of a low power FFT processor. Previous research has focused on implementing SRFFT with “L” shaped butterfly. However, comparing to standard radix-2 FFT processor, the mixed radix property of “L” shaped butterfly either increases the complexity of control logic or it requires extra hardware to buffer interim data.

We observed that flow graph of split-radix algorithm is the same as radix-2 FFT except for the location and value of twiddle factors. Therefore address generation scheme for conventional radix-2 FFT algorithm could also be applied to SRFFT. In our recent work [3], we have followed ASIC flow and developed a low power SRFFT processor using shared-memory architecture. We adopted address generation scheme in [2]. In this paper we will discuss the method of using FPGA to implement our proposed design. Section II presents the structure of our design and preliminary simulation results on Xilinx Spartan 6 FPGA. Conclusion and future research plan is provided in section III.

II. DESIGN METHOD AND PRELIMINARY RESULTS

A. Unit Design

The structure of our SRFFT processor is shown in figure 1. It consists of two RAM banks, two ROM banks, one butterfly unit and address generation circuits. FFT data are stored in RAM banks and Xiao’s [2] algorithm could be used to generate their corresponding addresses. However, for address generation of twiddle factors, which are stored in the ROMs, we cannot simply adopts Xiao’s algorithm, since conventional radix-2 FFT only has twiddle factor at the lower leg of butterfly while SRFFT has twiddle factors at both upper leg and lower leg. In our implementation, we use a decoder (look-up table) which has the combination of pass counter P and butterfly counters B as input and generates the address of twiddle factors as well as control signal for the clock gating registers.

Fig.1. FFT block diagram

In our previous publication [3], we have proposed a low power radix-2 butterfly unit using tristate buffer. Since there are no internal tristate buffers in modern FPGA, we modify our butterfly unit and use clock gating approach to block unnecessary switching activity in multiplier. The structure of our butterfly unit is shown in figure 2.
When we need to multiply twiddle factors, data go through the multiplier path. Otherwise, data just go to output.

Table I shows the preliminary results of implementing proposed design and reference design [2] on Xilinx Spartan 6 XC6SLX75T. For 32-point FFT the proposed design could achieve a power saving ratio up to 11.2% with a slightly increased critical path delay and static power. Because the number of multiplications required by SRFFT algorithm significantly decreases as the FFT size increases, it is reasonable to assume the proposed architecture will save more power when it comes to larger points of FFT.

We also observe that on target device the resource utilization ratio is about 2% for a 32-point FFT processor, therefore it is possible to implement multiple copy of the proposed design on the same device and throughput will significantly increase. This part of work is still in progress and we will discuss current status in next section.

![Fig. 2. Low power butterfly structure](image)

**Table I**

<table>
<thead>
<tr>
<th>Power and Delay Comparison on Spartan 6 FPGA</th>
<th>Proposed Design</th>
<th>Xiao’s Design [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>32-point FFT with 32-bit data width</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delay (ns)</td>
<td>15.581</td>
<td>14.965</td>
</tr>
<tr>
<td>Static power (mW)</td>
<td>650</td>
<td>640</td>
</tr>
<tr>
<td>Dynamic power (mW)</td>
<td>220</td>
<td>340</td>
</tr>
<tr>
<td>Total power saving ratio</td>
<td>11.2%</td>
<td></td>
</tr>
</tbody>
</table>

**B. Parallel Architecture on FPGA**

Recently there is increasing popularity of parallel architecture and better performance could be obtained by only scaling to platforms with larger number of cores. In implementing parallel multi-core FFT processor in FPGA, we must optimize the architectures to save resources since there are usually limited resources on FPGA. We proposed a scalable architecture which employs small-size reconfigurable SRFFT processor as basic building block to form large array of processor networks. A key observation we made is that the entire address generation logic and ROM banks could be decoupled from each unit so that a single address generation engine can be shared between multiple units. Figure 3 shows column-sharing topology of processor networks but it is also possible to use other types of topology such as row-sharing topology.

![Fig. 3. Column-sharing topology](image)

**III CONCLUSION AND FUTURE WORK**

In this paper we described a low power SRFFT processor structure that is suitable for implementation on FPGA. A trade-off has been made between hardware resources and dynamic power. Preliminary simulation results demonstrate the benefits of our approach. So far we only compare simulation results with one reference design. In future we will compare our approach with other existing works. Another aspect of ongoing work would involve developing efficient twiddle factor addressing scheme. In our implementation, we have used the look-up table approach to generate addresses of twiddle factors. If a well defined algorithm could be found, our design could be configured to any FFT size without the need to reprogram the address decoding part.

**REFERENCES**


[3] Z. Qian and M. Margala, “A Novel Low-Power and In-Place Split-Radix FFT Processor,” in *ACM Proc. of the 24th Great Lakes Symp. on VLSI (GLVLSI’14)*, Houston, TX, USA, 2014, pp.81-82