Architectures for Dynamic Data Scaling in 2/4/8K Pipeline FFT Cores

Lenart, Thomas; Öwall, Viktor

Published in: IEEE Transactions on Very Large Scale Integration (VLSI) Systems

DOI: 10.1109/TVLSI.2006.886407

2006

Link to publication


Total number of authors: 2

General rights
Unless other specific re-use rights are stated the following general rights apply:
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.
• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Read more about Creative commons licenses: https://creativecommons.org/licenses/

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
Architectures for Dynamic Data Scaling in 2/4/8K Pipeline FFT Cores
Thomas Lenart and Viktor Öwall

Abstract—This paper presents architectures for supporting dynamic data scaling in pipeline fast Fourier transforms (FFTs), suitable when implementing large size FFTs in applications such as digital video broadcasting and digital holographic imaging. In a pipeline FFT, data is continuously streaming and must, hence, be scaled without stalling the dataflow. We propose a hybrid floating-point scheme with tailored exponent datapath, and a co-optimized architecture between hybrid floating point and block floating point (BFP) to reduce memory requirements for 2-D FFT processing. The presented co-optimization generates a higher performance and memory efficiency compared to (a).

Index Terms—Block floating point (BFP), convergent BFP (CBFP), digital holography, digital video broadcasting (DVB), dynamic data scaling, fast Fourier transform (FFT), hybrid floating point, orthogonal frequency-division multiplexing (OFDM).

I. INTRODUCTION

The fast Fourier transform (FFT), is one of the most commonly used operations in digital signal processing and, currently, the demands increase towards larger and multidimensional transforms. Larger transforms require more processing on each data sample, which increases the total quantization noise. This can be avoided by gradually increasing the wordlength inside the pipeline, but affects memory requirements as well as the critical path in arithmetic components. For large size FFTs, dynamic scaling is, therefore, a suitable tradeoff between arithmetic complexity and memory requirements. The following architectures have been evaluated and compared with related work.

a) A hybrid floating-point pipeline with fixed-point input and tailored exponent datapath for 1-D FFT computation.

b) A hybrid floating-point pipeline for 2-D FFT computation, which also requires the input format to be hybrid floating point. Hence, the hardware cost is slightly higher than in (a).

c) A co-optimized design based on a hybrid floating-point pipeline combined with block floating point (BFP) for 2-D FFT computation. This architecture has the processing abilities of (b) with hardware requirements comparable to (a).

The primary target application for the implemented FFT core is a microscope based on digital holography [2] where visible images are to be digitally reconstructed from an interference pattern. The pattern is recorded on a large digital image sensor with a resolution of 2048 × 2048 pixels and processed by a reconstruction algorithm based on a 2-D Fourier transformation. Hence, the architectures outlined in (b) and (c) are suitable for this application. Another area of interest is in wireless communication systems based on orthogonal frequency division multiplexing (OFDM). The OFDM scheme is used in, for example, digital video broadcasting (DVB) [3], including DVB-T with 2/8-K FFT modes and DVB-H with an additional 4-K FFT mode. The architecture described in (a) is suitable for this field of application.

Section II gives a brief introduction to the FFT and Section III presents different dynamic data scaling alternatives for pipeline FFTs, with additional architectural features described in Section IV. Section V shows software simulation results in terms of precision and memory requirements. Finally, Section VI presents the VLSI implementation and measurements on the fabricated application-specific integrated circuit (ASIC) prototype, and a conclusion is given in Section VII.

II. FFT ARCHITECTURE

The fast Fourier transform is a decomposition of an N-point discrete Fourier transform (DFT) into successively smaller DFT transforms [4]. This paper describes pipeline FFT architectures, constructed from a number of cascaded Radix-2 butterfly blocks and complex multipliers each dividing the sequence into r smaller FFTs. Another common approach is parallel FFT architectures, placing the computational blocks in parallel instead of cascaded. Simulations and implementations presented in this paper are all based on the Radix-2^r single path delay feedback (R2^r SDF) algorithm [5]. Input data is supplied in linear sample order, hence, requiring the largest delay feedback buffer size NFFT/2 in the initial butterfly unit. The NFFT-point Radix-2^r pipeline has low memory requirements (NFFT − 1 words), simple butterfly architecture and requires only log2(NFFT − 1) complex multipliers. The Radix-2^r butterfly is constructed from two Radix-2 butterflies divided by a trivial multiplication, as shown in Fig. 1.

III. DYNAMIC DATA SCALING

Fixed-point is a widely used format in realtime and low-power applications due to the simple implementation of arithmetic units. In fixed-point arithmetic, a result from a multiplication is usually rounded or truncated to avoid a significantly increased wordlength, hence, generating a quantization error. The quantization energy caused by rounding is relatively constant due to the fixed location of the binary
In practical applications the flow of data between each stage requires a large amount of memory, and in particular the size and characteristics of the input signal [6]. Finding a common data buffer scheme for pipeline architectures [7] has been proposed for pipeline architectures [7]. By placing buffers between intermediate stages, data can be rescaled using BFP, as shown in Fig. 3. The block size will decrease as data propagates through the pipeline until each value has its own exponent. Intermediate buffering of data between each stage requires a large amount of memory, and in practical applications the first intermediate buffer is often omitted to save storage space. However, this leads to a reduced SQNR as will be shown in Section V and referred to as CBFP$_{\text{low}}$ due to the lower memory requirements.

### A. Hybrid Floating-Point

Floating-point arithmetic increases the dynamic range by expressing numbers with a mantissa $m$ and an exponent $e$, represented with $M$ and $E$ bits, respectively. A hybrid and simplified scheme for floating point representation of complex numbers is to use a single exponent for the real and imaginary part. Besides reduced complexity in the arithmetic units the total wordlength for a complex number is reduced from the real and imaginary part. Besides reduced complexity in the arithmetic building blocks, and the number of exponent bits for the same stage is denoted $N_r$. The length of the delay feedback memory, or first-in, first-out (FIFO), at stage $i$ is

$$N_r = 2^i, \quad 0 \leq i \leq i_{\text{max}} = \log_2 N_{\text{FFT}} - 1$$

and the number of exponent bits for the same stage is denoted $E_i$. The block size $D$ spans from single elements to $2N_r$, which can be expressed as

$$D(\alpha) = 2^{\alpha+1}, \quad 0 \leq \alpha \leq i + 1.$$ 

The total bits required for supporting dynamic scaling is the sum of exponent bits in the delay feedback unit and the total size of the intermediate buffer. This can be expressed as

$$\text{Mem}_{\alpha} = E_i \left[ \frac{\gamma N_r}{D(\alpha)} \right] + L \left( D(\alpha) - 1 \right)$$

where

$$\gamma = \begin{cases} 1, & \text{Radix-2} \\ 3/2, & \text{Radix - } 2^2 \end{cases}$$

and

$$L = \begin{cases} 2M + E_i, & i = i_{\text{min}} \\ 2(M + T), & 0 \leq i < i_{\text{max}} \end{cases}$$

Fig. 1. Radix-2$^2$ ($R = 2^2$) butterfly is constructed from two Radix-2 ($R = 2$) butterflies, separated with a trivial multiplication.

Fig. 2. Building blocks for a hybrid floating-point implementation. (a) Symbol of a modified butterfly containing an align unit on the input. (b) Symbol for a modified complex multiplier containing a normalization unit.

Fig. 3. Example of CBFP. The buffer after each complex multiplier selects a common exponent for a group of values, allowing fixed-point butterfly units. The first buffer is often omitted to save storage space, but will have a negative impact on signal quality.

Fig. 4. (a) Hybrid floating point. (b) CBFP with $D = 2N - 1$ using a large buffer and fixed-point butterfly. (c) Small buffer reduces the exponent storage space in the delay feedback memory.

### B. BFP

BFP combines the advantages of simple fixed-point arithmetic with floating point dynamic range. A single exponent is assigned to a group of values to reduce memory requirements and arithmetic complexity. However, output signal quality depends on the block size and characteristics of the input signal [6]. Finding a common exponent requires processing of the complete block. This information is directly available in a parallel FFT architecture, but for pipeline FFT architectures scaling becomes more complicated since data is continuously streaming. A scheme known as convergent BFP (CBFP) has been proposed for pipeline architectures [7]. By placing buffers between intermediate stages, data can be rescaled using BFP, as shown in Fig. 3. The block size will decrease as data propagates through the pipeline until each value has its own exponent. Intermediate buffering of data between each stage requires a large amount of memory, and in practical applications the first intermediate buffer is often omitted to save storage space. However, this leads to a reduced SQNR as will
For Radix-$2^2$ butterflies, (1) is only defined for odd values of $i$. This is compensated by a scale factor $\gamma = 3/2$ to include both delay feedback units in the Radix-$2^2$ butterfly, as shown in Fig. 1. The buffer input wordlength $L$ differs between initial and internal butterflies. For every butterfly stage, $\alpha_i$ is chosen to minimize (1). For example, an 8192-point FFT using a hybrid floating-point format of $2 \times 10 + 4$ bits require 16 Kb of memory in the initial butterfly for storing exponents, as shown in Fig. 5. The number of memory elements for supporting dynamic scaling can be reduced to only 1256 bits by selecting a block size of 32, hence, removing over 90% of the storage space for exponents. The hardware overhead is a counter to keep track of when to update the block exponent in the delay feedback, similar to the exponent control logic required in CBFP implementations. Thus, the proposed co-optimized architecture supports hybrid floating point on the input port at very low hardware cost. Since the input and output format is the same, this architecture then becomes suitable for 2-D FFT computation.

### IV. ARCHITECTURAL EXTENSIONS

The architectures described in this paper have been extended with support for bidirectional processing, which is important for the intended application and also in many general applications. A pipeline FFT can support a bidirectional dataflow if all internal butterfly stages have the same wordlength. The advantage with a bidirectional pipeline is that input data can be supplied either in linear or bit-reversed sample order by changing the dataflow direction. One application for the bidirectional pipeline supports hybrid floating point on the input port at very low hardware cost. Since the input and output format is the same, this architecture then becomes suitable for 2-D FFT computation.

#### TABLE I

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Delay feedback</th>
<th>Intermediate buffers</th>
<th>Total memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>1D Co-optimization</td>
<td>45.8</td>
<td>1.6</td>
<td>47.4</td>
</tr>
<tr>
<td>1D Hybrid FP (A)</td>
<td>49.0</td>
<td>-</td>
<td>49.0</td>
</tr>
<tr>
<td>1D CBFP$_{low}$</td>
<td>45.7</td>
<td>14.7</td>
<td>60.4</td>
</tr>
<tr>
<td>1D CBFP</td>
<td>45.7</td>
<td>60.0</td>
<td>105.7</td>
</tr>
<tr>
<td>2D Co-optimization(C)</td>
<td>50.0</td>
<td>0.4</td>
<td>50.4</td>
</tr>
<tr>
<td>2D Hybrid FP (B)</td>
<td>53.9</td>
<td>-</td>
<td>53.9</td>
</tr>
</tbody>
</table>

#### Fig. 7. Decreasing the energy in a random value input signal affects only the architecture when scaling is not applied in the initial stage. Signal level = 1 means utilizing the full dynamic range.

#### Fig. 8. Decreasing the energy in a random value input signal with peak values utilizing the full dynamic range. This affects all block scaling architectures, and the SQNR depends on the block size. The co-optimized architecture performs better than CBFP since it has a smaller block size through the pipeline.

#### V. SIMULATIONS

A simulation tool has been designed to evaluate different FFT architectures in terms of precision, dynamic range, memory requirements, and estimated chip size based on architectural descriptions. The user
can specify the number of bits for representing mantissa $M$, exponents $E$, twiddle factors $T$, FFT size $(N_{FFT})$, rounding type, and simulation stimuli. To make a fair comparison with related work, all architectures have been described and simulated in the developed tool.

First, we compare the proposed architectures with CBFP in terms of memory requirements and signal quality. In addition to the lower memory requirements, we will show how the co-optimized architecture produces a higher SQNR than CBFP. Second, we will compare the fabricated design with related work in terms of chip size and data throughput.

Table I shows a comparison of memory distribution between delay feedback units and intermediate buffers. 1-D architectures have fixed-point input, whereas 2-D architecture supports hybrid floating-point input. The table shows that the intermediate buffers used in CBFP consume a large amount of memory, which puts the co-optimized architecture in favor for 1-D processing. For 2-D processing, the co-optimized architecture also has lower memory requirements than hybrid floating point due to the buffer optimization. Figs. 7 and 8 present simulation results for the 1-D architectures in Table I. Fig. 7 is a simulation to compare SQNR when changing energy level in the input signal. In this case, the variations only affect CBFP. Second, we will compare the fabricated design with related work in terms of chip size and data throughput.

Table II shows an extended comparison between the proposed architectures and related work. The table includes two pipeline architectures and related work. The table shows that the memory requirements and signal quality are important, the wordlength in arithmetic components and, therefore, also the chip area. The proposed architectures have low hardware requirements and produce high SQNR using dynamic data scaling. They can easily be adopted to 2-D signal processing, in contrast to architectures without data scaling or using CBFP. The pipeline implementation results in a high throughput by continuous data streaming, which is shown as peak 1-D transforms in Table II.

### Table I

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
<td>Pipeline (1D)</td>
<td>Pipeline (2D)</td>
<td>Co-optimization</td>
<td>BFP (D=64)</td>
<td>Pipeline</td>
</tr>
<tr>
<td>Dynamic scaling</td>
<td>Hybrid FP</td>
<td>Hybrid FP</td>
<td></td>
<td>CBFP</td>
<td></td>
</tr>
<tr>
<td>Technology (µm)</td>
<td>0.35</td>
<td>0.35</td>
<td></td>
<td>0.18</td>
<td>0.35</td>
</tr>
<tr>
<td>Max Frequency (MHz)</td>
<td>76</td>
<td>50</td>
<td></td>
<td>56</td>
<td>16</td>
</tr>
<tr>
<td>Input wordlength</td>
<td>$2 \times 10$</td>
<td>$2 \times 10 + 4$</td>
<td>$2 \times 10 + 4$</td>
<td>$2 \times 10$</td>
<td>$2 \times 8$</td>
</tr>
<tr>
<td>Internal wordlength</td>
<td>$2 \times 11 + (0 \ldots 4)$</td>
<td>$2 \times 11 + 4$</td>
<td>$2 \times 11 + 4$</td>
<td>$2 \times 11 + 4$</td>
<td>$2 \times 12 + 4$</td>
</tr>
<tr>
<td>Transform size</td>
<td>2K</td>
<td>1/2/4/8K</td>
<td>2K</td>
<td>2K</td>
<td>8K</td>
</tr>
<tr>
<td>SQNR (dB)</td>
<td>45.3</td>
<td>44.0</td>
<td>45.3</td>
<td>44.3</td>
<td>41.2</td>
</tr>
<tr>
<td>Memory (bits)</td>
<td>40K</td>
<td>196K</td>
<td>53.9K</td>
<td>50.4K</td>
<td>185K</td>
</tr>
<tr>
<td>Norm. Area (mm²)</td>
<td>7.58</td>
<td>16</td>
<td>5.92</td>
<td>18.3</td>
<td>49</td>
</tr>
<tr>
<td>Peak 1D Transform/s</td>
<td>37109</td>
<td>9277</td>
<td>24414</td>
<td>24414</td>
<td>3905</td>
</tr>
</tbody>
</table>

1 Area normalized to 0.35 µm technology.
input/output (I/O) pads and 26 power pads. The implementation requires 11 delay feedback buffers, one for each butterfly unit. Seven on-chip RAMs are used as delay buffers (approximately 49 Kb), while the four smallest buffers are implemented using flip-flops. Twiddle factors are stored in three ROMs containing approximately 47 Kb. The memories can be seen along the sides of the chip. The number of equivalent gates (two-input NAND) is 45900 for combinatorial area and 78300 for noncombinatorial area (including memories). The power consumption of the core was measured to 526 mW when running at 50 MHz and using a supply voltage of 2.7 V. The pipeline architecture produce one output value each clock cycle, or 37-K transforms per second running at maximum clock frequency. The 2-D FFT architecture (b) has been implemented on FPGA in [12].

VII. CONCLUSION

New dynamic data scaling architectures for pipeline FFTs have been proposed for both 1-D and 2-D applications. Based on hybrid floating point, a high-precision pipeline with low memory and arithmetic requirements has been constructed. A co-optimization between hybrid floating point and BFP has been proposed, reducing the memory requirement further by adding small intermediate buffers. A 2048 complex point pipeline FFT core has been implemented and fabricated in a 0.35-μm, 5-ML CMOS process, based on the presented scaling architecture and a throughput of 1 complex point/cc. The bidirectional pipeline FFT core, intended for image reconstruction in digital holography, has also been integrated on a custom designed FPGA platform to create a complete hardware accelerator for digital holographic imaging.

REFERENCES