A Systolic FFT Architecture for Real Time FPGA Systems

Preston A. Jackson, Cy P. Chan, Jonathan E. Scalera, Charles M. Rader, and M. Michael Vai
MIT Lincoln Laboratory
244 Wood ST, Lexington, MA 02420
{preston,cychan,jscalera,cmr,mvai}@ll.mit.edu

Abstract

MIT Lincoln Laboratory has recently developed a new systolic FFT architecture for FPGAs. This architecture utilizes a parallel design to provide high throughput and excellent numerical accuracy. Using this design, an 8192-point real-time FFT, operating at 1.2 billion samples per second and performing 78 Gops with 70 dB of accuracy, fits on a single Xilinx Virtex II 8000. Keywords: FPGA, DFT, FFT, high performance, parallel, systolic, correlation.

1. Introduction

The Fast Fourier Transform (FFT) has become almost ubiquitous in high speed signal processing. Using this transform, signals can be moved to the frequency domain where filtering and correlation can be performed with fewer operations. This paper presents a new high performance systolic FFT architecture for use on field programmable gate arrays (FPGAs). The details of this architecture will be presented in the following sections. Section 2 begins with an overview of the architecture, next, Section 3 discusses performance, and finally, Section 4 closes with a brief conclusion with a look to future work.

2. Architecture

A fully parallel hardware implementation of the FFT can be quickly derived from the the data flow graph of the algorithm. Figure 1 shows such an architecture for an 8 point FFT. This architecture consists of $\log_2(N)$ pipeline stages, where N is the number of samples, to concurrently produce an entire output sequence. However, this benefit is realized only if an input sequence is available simultaneously every cycle. It is often the case that data are introduced serially into the device, for example by an analog-to-digital converter (ADC). Using serial data with this architecture would leave much of the circuitry idle until the input data sequence has streamed in.

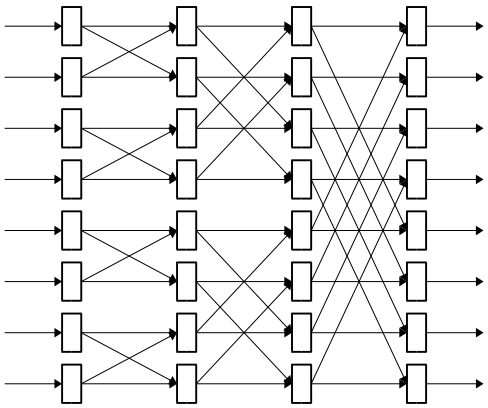

Figure 1. Traditional FFT Data Flow

If only one data sample is ready at a time only one butterfly per stage can be active at a time. Taking this input pattern into account, the entire column of butterflies is collapsed into a single butterfly [1] as shown in Figure 2. Using this hardware efficient implementation, the circuitry achieves full efficiency, but the entire design must operate at the same frequency that the input data arrive.

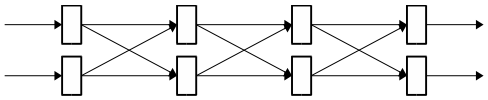

Figure 2. Serial FFT Data Flow

The structure of an individual collapsed butterfly stage can be seen in Figure 3. The stage adds two FIFOs and two muxes to the typical butterfly architecture. This added circuitry allows for the scheduling of intermediate results so that the correct butterfly operations are preserved. The twiddle factors are precomputed and are

This work was sponsored by DARPA ATO under Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

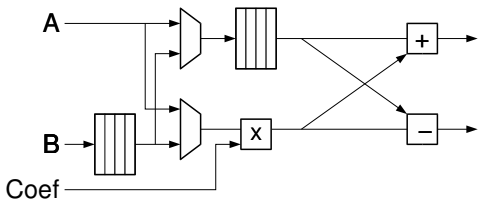

Figure 3. Serial Butterfly Stage

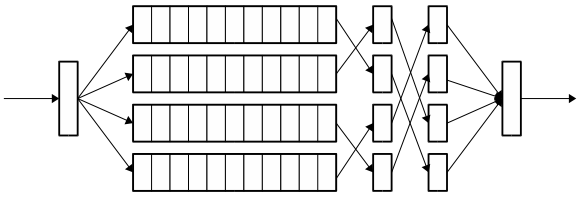

Figure 4. Parallel Serial System Block Diagram

stored in local memory within the FPGA at configure time.

This paper presents a hardware architecture which parallelizes the serial architecture of Figure 2 into M parallel pipelines. Working in parallel reduces the input data to a manageable rate. An example of the end-to-end structure for $N = 8192$, $\log_2(N) = 13$, and $M = 4$ can be seen in Figure 4. Each of the small boxes represents a butterfly stage. A high speed input distributer turns the serial stream of complex data at X MHz into M parallel streams of complex data, allowing the internal design to operate at $\frac{X}{M}$ MHz. The splitter is followed by $N - \log_2(M)$ columns of butterflies grouped together into M pipelines. These pipelines are synchronized but do not share any data. They are followed by $\log_2(M)$ final columns of butterflies. The final module on the right converts the slow parallel output data stream into a high speed serial output.

One important feature of this architecture is that the pipelines can be performed on multiple FPGAs, as long as there is a high speed bus between them with a consistent and synchronous delay. However, the final $\log_2(M)$ stages have highly dependent data flow which must be computed on the same FPGA if high performance is to be achieved.

3. Performance

This paper discusses the specific implementation of an 8192-point FFT developed at MIT Lincoln Laboratory. The input is 1.2 GSPS of real data produced by a Max 108 ADC. The real data sequence is converted to complex data and down sampled to 600 MSPS.

This parallel systolic architecture can be used to implement FFTs of different sizes and on FPGAs or application specific integrated circuits (ASICs). An FPGA was chosen in the current implementation because it met the performance requirements and did not have the high cost and rigid architecture of an ASIC. The performance of this system makes it ideal for real time applications. Once the pipeline is full, the architecture will produce a valid data output every cycle.

All computations were fixed-bit operations. As mentioned above, the inputs to the FFT are eight bits wide, four bits of fraction and four bits of integer. Immediately after receiving the inputs, they are extended to 18 bits, four bits of integer and 14 bits of fraction. As the data flows through the butterfly stages, the boundary between the fractional portion and the integer portion is changed to provide appropriate dynamic ranges. After each operation, overflow and underflow are tested and if detected, the result of the operation is replaced by the maximum or minimum representable number, respectively. Using these techniques, the design achieves greater than 70 dB of accuracy.

The entire design can fit on a single Xilinx Virtex II 8000 FPGA, requiring 71% of the slices, 33% of the BlockRAMs and 85% of the multipliers. It can consume data and produce results at 1.2 billion samples per second (600 million complex samples per second). The design is well suited for real-time applications because of its pipelined design. Each sample has an identical latency of 1024 cycles. The entire chip runs 23.4 billion multiplies and 54.6 billion adds per second, for a total of 78 billion operations per second.

4. Conclusion

This paper presents an efficient and high performance systolic architecture for computing the FFT. The specific implementation presented in this paper consumes and computes 1.2 billion 8-bit samples per second, performs 78 billion 18-bit operations per second while achieving over 70 dB of accuracy on a single Xilinx Virtex II 8000.

Further optimizations may be possible for this design. The logic requirements of this FFT do not exceed those provided on a Xilinx Virtex II 6000 FPGA, however the automatic place and route tools exhaust all of the routing resources. Due to its highly regular structure and mostly local communication pattern, a manually guided place and route effort could allow the design to fit entirely on this smaller part.

References

[1] L. R. Rabiner and B. Gold. *Theory and Application of Digital Signal Processing*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1975.