A 9.5mW 330μsec 1024-point FFT Processor

Bevan M. Baas

Department of Electrical Engineering, Stanford University, Stanford, CA 94305-9515
bbaas@nova.stanford.edu • http://nova.stanford.edu/~bbaas/

Abstract

This paper presents an energy-efficient, single-chip, 1024-point FFT processor. The full-custom, 460,000-transistor design has been fabricated in a standard 0.7μm ($L_{poly} = 0.6\mu$m) CMOS process and is fully functional on first-pass silicon. At a supply voltage of 1.1V, it calculates a 1024-point complex FFT in 330μsec at a clock speed of 16 MHz while consuming 9.5 mW, resulting in an adjusted energy efficiency more than 16 times greater than the previously most-efficient known FFT processor. At 3.3V, it operates at 173 MHz.

Introduction

The Fast Fourier Transform (FFT) is one of the most widely used digital signal processing algorithms. While advances in semiconductor processing technology have enabled the performance and integration of FFT processors to increase steadily, these advances have also, unfortunately, lead to an increase in power consumption as well. This has resulted in a situation where the number of potential FFT applications that are limited by power—not performance (e.g., portable applications)—is significant and growing.

For many CMOS circuits, energy consumption is proportional to the supply voltage squared [1]. Consequently, tremendous efficiency can be gained by aggressively reducing the supply voltage. Unfortunately, circuit performance is reduced with lower supply voltages. The processor presented here is designed to operate with a low supply voltage, V_{dd}, which approaches the value of the transistor thresholds, V_t, to dramatically increase the overall energy-efficiency. To regain some lost performance, the processor utilizes a high-performance algorithm and architecture that is shown to be better performing than previous designs.

Processor Architecture

As with most DSP algorithms, FFTs are very memory-intensive. FFTs are calculated in $\mathcal{O}(\log N)$ *stages*, where N is the length of the transform, and each stage requires the reading and writing of all N data words. To maintain good performance, many previous "longer-length" ($N \geq 1024$) FFT processor designs used multiple datapaths and large crossbar, bus, or network structures connected to a partitioned memory. To avoid this interconnection bottleneck, the chip presented here implements a data-caching algorithm which provides increased energy-efficiency (by reducing communication energy) and increased performance (through deep pipelining). Figure 1 is a high-level block diagram of the system showing the tightly-coupled processor-cache pair and the N-word main memory.

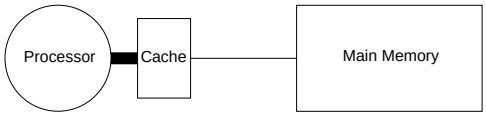

Figure 1: System block diagram

It is well known that data caches increase the effective bandwidth to a memory—but only if the memory access pattern exhibits a fair amount of locality. Although nearly all FFT algorithms have very poor locality, in [2], an algorithm is described which offers good locality over large portions of the computation. The global communication inherent in the FFT is concentrated into a few (typically 1 or 2) intermediate steps and is easily accomplished through appropriate addressing when filling and flushing the cache. Because the FFT algorithm is deterministic, cache tags are unnecessary and correct cache operation is achieved through predetermined cache addressing and pre-fetching of data from main memory.

The Spiffee Processor

A 1024-point single-chip cached-architecture FFT processor named *Spiffee* was designed and fabricated. It operates on complex 36-bit (18-bit Real + 18-bit Imaginary) fixed-point data and has internal datapath widths varying between 20–24 bits. The processor was designed with two *epochs*, E, so each word in main memory is read and written twice per transform. With $N = 1024$, the cache size C equals $\sqrt[E]{N} = \sqrt[2]{1024} = 32$ words. Although the architecture easily supports multiple processors, the chip presented here contains a single processor/cache pair and a single set of main memory. The data cache reduces traffic to main memory by a factor of $\log_r(N)/E$, which in this implementation is $\log_2(1024)/2 = 5$. This allows more processors to be added and/or a slower lower-power main memory to be used. Power used to access data decreases since data words are stored in a smaller memory and nearer to the datapath.

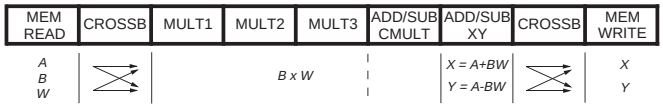

Figure 2: Pipeline diagram

While higher-radix, prime-factor, and other FFT algorithms have been shown to require fewer operations than Radix-2 [3], Spiffee was designed with a Radix-2 decomposition. This is due to the fact that for a VLSI implementation, the regularity and simplicity of an algorithm are very important factors in determining the clock speed, design time, and other key parameters. The processor's datapath calculates one complex radix-2 decimation-in-time butterfly [4] per cycle. This requires the calculation of two butterfly outputs, X and Y, from two butterfly inputs, A and B, and a complex coefficient W, using the equations: $X = A + BW$ and $Y = A - BW$. In general, all variables are complex.

The datapath and cache are aggressively pipelined to retain high performance, as evidenced by the 9-stage pipeline diagram shown in Figure 2. In the first pipeline stage, A and B are read from the caches and W is read from a ROM. In stage two, the operands are routed through two 2×2 crossbars to the correct functional units. Four $B_{\{real,imag\}} \times W_{\{real,imag\}}$ multiplications of the real and imaginary components of B and W are calculated in stages three through five. Stage six completes the complex multiplication, stage seven performs the remaining additions or subtractions to calculate X and Y, and pipeline stages eight and nine complete the routing and write-back of the results. The deep pipeline causes a read-after-write data hazard to occur once every 80 cycles and is handled by stalling the pipeline for one cycle.

Figure 3 is a block diagram of the chip and Figure 4 is the corresponding die microphotograph. Four signed, pipelined, array multipliers produce 24-bit products from 20-bit operands. They use booth-2 encoding and use (4,2) and (3,2) adders to reduce partial products. Six single-cycle 24-bit adders and subtractors propagate carries using a hybrid of carry-lookahead and ripple techniques. The ROM is organized as two 256-word $\times$ 40-bit arrays and stores complex W coefficients used in the FFT kernel calculations. The two sets of caches are designed so that one set can perform calculations while the other is being flushed and filled from memory. Each set is organized as two banks of 16-word $\times$ 40-bit dual-ported SRAM arrays using 10-transistor cells. The main memory is made up of eight 128-word $\times$ 36-bit SRAM arrays using 6-transistor cells.

The full-custom design contains 460,000 transistors and was fabricated in a standard single-poly, triple-

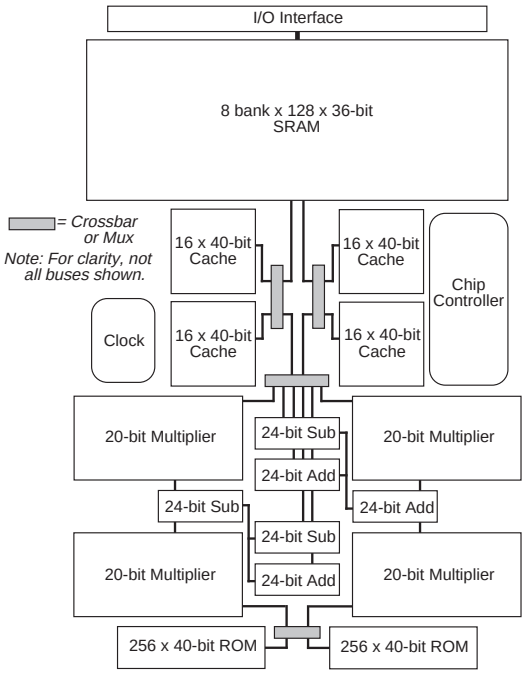

Figure 3: Chip block diagram

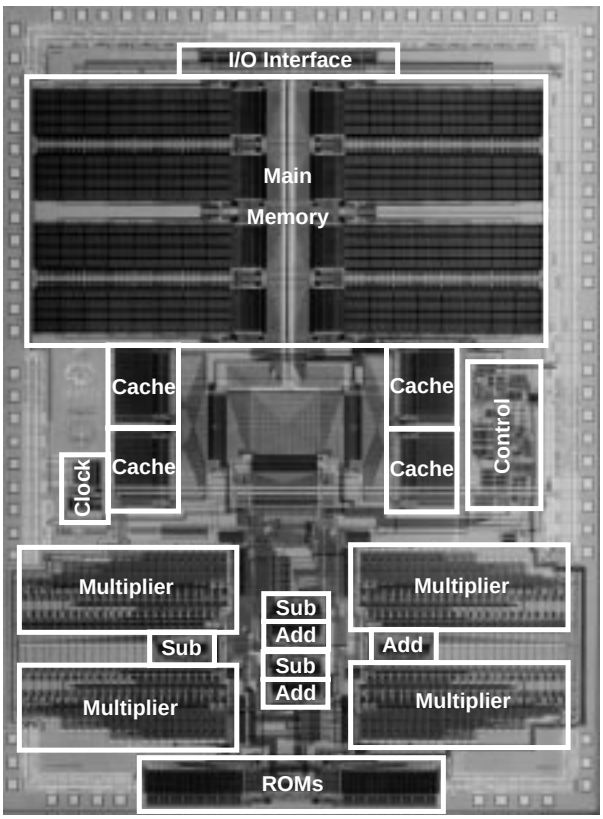

Figure 4: Die microphotograph

Well/substrate V_{bs} bias (Volts)	NMOS V_t (Volts)	PMOS V_t (Volts)
$-2.0V$	0.96	-1.14
$0.0V$	0.68	-0.93
$+0.5V$	0.48	-0.82

Table 1: Measured V_t values

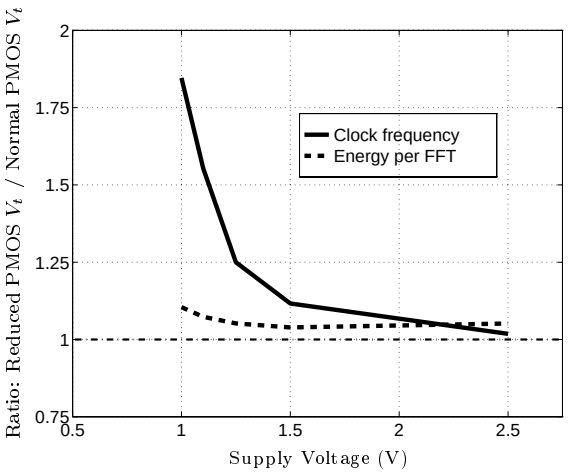

Figure 5: Measured change in performance and energy-consumption with an n-well bias applied.

metal CMOS process using 0.7μm design rules with $L_{poly} = 0.6\mu$m. It occupies 5.985mm × 8.204mm and is fully functional on first-pass silicon. Static circuits are used almost exclusively, and were optimized for low power and robust low-voltage operation.

Low-power Operation

To enhance performance at very low voltages, the processor was designed with n-well and p-substrate nodes not connected to V_{dd} and Gnd, but instead routed to pads to allow the biasing of transistor bodies and thereby adjust threshold voltages [5]. Table 1 details the 480mV and 320mV V_t shift range that was measured for the NMOS and PMOS respectively. Because the PMOS thresholds were so much larger than the NMOS thresholds, they were the primary performance limiter at low V_{dd}. The chip was measured functional down to $V_{dd} = 1.0$V. At $V_{dd} = 1.1$V, it ran at 16MHz and 9.5mW with the nwells forward biased +0.5V—which was a 60% improvement over the 10MHz operation without bias. With that bias, 11μA of current flowed from the n-wells while the chip was active. Latchup was never observed. Figure 5 shows the dramatic improvement in operating frequency and the slight increase in energy consumption per FFT caused by adjusting PMOS V_ts—for various values of V_{dd}.

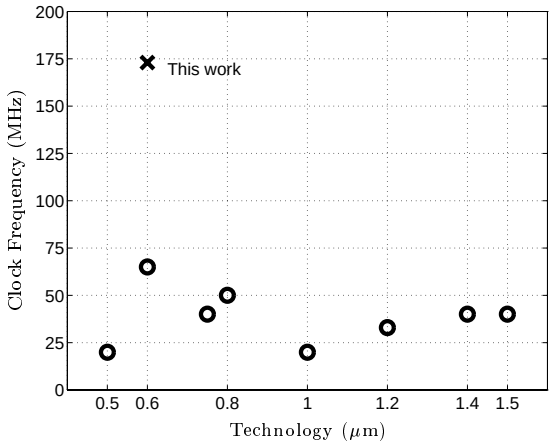

Figure 6: CMOS technology vs. clock frequency for various FFT processors.

Results and Analysis

In addition to excellent low-V_{dd} performance, Spiffee also demonstrated good high-V_{dd} performance. At $V_{dd} = 3.3$V, it ran at 173MHz calculating a 1024-point complex FFT in 30μsec while consuming 845mW.

Table 2 contains a summary of relevant characteristics of ten commercial and academic FFT processors calculating 1024-point complex FFTs. *CMOS Technology* is the minimum feature size of the CMOS process the chip was fabricated in. *Datapath width* is the width, in number of bits, of the multipliers for the scalar datapaths. *Number of chips* values with +'s indicate additional memory chips for data and/or coefficients are required. *Normalized Area* is the silicon area normalized to a 0.5μm technology with the following relationship:

$$Normalized\ Area = \frac{Area}{(Technology/0.5\mu\text{m})^2} \qquad (1)$$

The final column, *FFTs per Energy*, compares the number of 1024-point complex FFTs calculated per energy and attempts to factor out technology and the datapath word width, *DPath*. It makes use of the observation that roughly 1/3 of the energy consumption of the 20-bit Spiffee processor scales as $DPath^2$ (e.g., multipliers) and approximately 2/3 scales linearly with *DPath*.

$$FFTs/Energy = \frac{Tech \times \left(\frac{2}{3}\frac{DPath}{20} + \frac{1}{3}\left(\frac{DPath}{20}\right)^2\right)}{Power \times Exec\ Time \times 10^{-6}} \qquad (2)$$

The processor presented here is 16× more energy-efficient than the previously most efficient known processor.

While clock speed is not the only factor, it is certainly an important factor in determining the performance and area-efficiency of a design. Figure 6 compares clock speeds of this cached FFT design running at $V_{dd} = 3.3$V

Processor	CMOS Tech (μm)	Datapath width (bits)	1024-pt Exec Time (μsec)	Power (mW)	Clock Freq (MHz)	Num of chips	Norm Area (mm^2)	FFTs per Energy
LSI, L64280 [6]	1.5	20	26	20,000	40	20	233	2.9
Plessey, 16510A [7]	1.4	16	98	3,000	40	1	22	3.6
Honeywell, DASP [8]	1.2	16	102	$\sim$ 5,250	–	2+	–	1.7
Y. Zhu, U of Calgary	1.2	16	155	–	33	–	–	–
Dassault Electronique	1.0	12	10.2	15,000	25	6	240	3.4
Tex Mem Sys, TM-66	0.8	32	65	7,000	50	2+	–	3.4
Cobra, Col. State [9]	0.75	23	9.5	7,700	40	16+	1104+	12.4
Sicom, SNC960A	0.6	16	20	2,500	65	1	–	9.0
CNET, E. Bidet [10]	0.5	10	51	300	20	1	100	13.6
Spiffee, $V_{dd} = 1.1$V	0.7/0.6	20	330	9.5	16	1	25	**223**
Spiffee, $V_{dd} = 3.3$V	0.7/0.6	20	30	845	173	1	25	27.6

Table 2: Comparison of processors calculating 1024-point complex FFTs

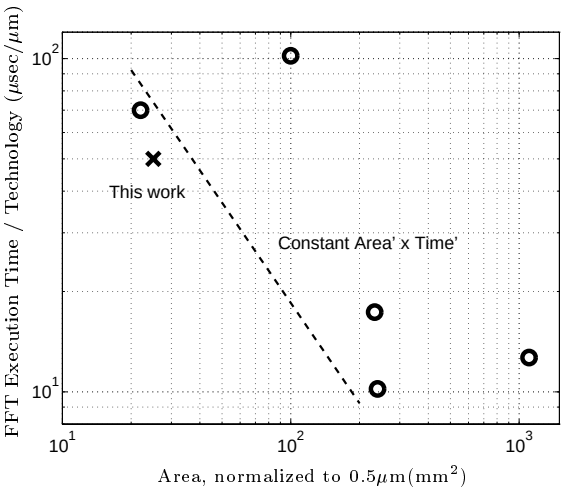

Figure 7: Silicon area vs. FFT execution time for several FFT processors.

with other FFT processors, versus their CMOS technologies. Spiffee operates with a clock frequency that is 2.6× greater than the next fastest processor.

The cost of a device is a strong function of its silicon area. So processors with high performance and a small area will be the most cost efficient. Figure 7 shows the first-order-normalized FFT calculation time (*Exec Time/Technology*) versus silicon area normalized to 0.5μm, for several FFT processors. The dashed line shows a constant Area$'$ × Time$'$ contour. The processor presented here compares favorably with the other processors despite its lightly-compacted layout, and its less area-efficient circuits which were designed for low-voltage operation and V_t tunability.

Acknowledgments

The author gratefully acknowledges valuable guidance from Jim Burr, Masataka Matsui, and Len Tyler; support from NASA through GSRP fellowship NGT-70340; and chip fabrication by MOSIS.

References

[1] Weste, N. and K. Eshraghian, *Principles of CMOS VLSI Design*, Addison-Wesley, Reading, MA, 1985.

[2] Baas, B. M., "An Energy-Efficient Single-Chip FFT Processor," *Symposium on VLSI Circuits*, June 1996.

[3] Burrus, C. S. and T. W. Parks. *DFT/FFT and Convolution Algorithms*, John Wiley & Sons, NY, NY, 1985.

[4] Oppenheim, A. V. and R. W. Schafer. *Discrete-Time Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[5] Burr, J. B., "Ultra low power CMOS technology," *NASA VLSI Design Symposium*, Oct 1991.

[6] Ruetz, P. A. and M. M. Cai, "A Real Time FFT Chip Set: Architectural Issues." *ICPR*, June 1990.

[7] O'Brien, J. *et al.*, "A 200 MIPS Single-Chip 1K FFT Processor." *ISSCC*, 1989.

[8] Magar, S., *et al.*, "An Application Specific DSP Chip Set for 100 MHz Data Rates." *ICASSP*, April 1988.

[9] Sunada, G. *et al.*, "COBRA: An 1.2 Million Transistor Expandable Column FFT Chip," *ICCD*, 1994.

[10] Bidet, E. *et al.*, "A Fast Single-Chip Implementation of 8192 Complex Point FFT." *JSSC*, March 1995.

Information for un-cited processors was gathered from company literature, WWW pages, and/or private communication with the designers.