EMPIRICAL PERFORMANCE PREDICTION FOR IFFT/FFT CORES FOR OFDM SYSTEMS-ON-A-CHIP

Kostas Pagiamtzis and P. Glenn Gulak

Dept. of Electrical and Computer Engineering
University of Toronto
Toronto, ON M5S 3G4, Canada
{pagiamt,gulak}@eecg.toronto.edu

ABSTRACT

Quick and accurate prediction of area, speed, and power of IP cores for SoC implementations reduces the design time and thus the overall cost. Accurate performance estimation early in the design cycle also is valuable for identifying trade-offs in the different blocks that make up an SoC. This work provides an empirical estimation method for IFFT/FFT blocks in multicarrier (OFDM, DMT) systems. The relative accuracy of the predictions allows the comparison of implementation alternatives, and the absolute accuracy enables the prediction of the final performance of fabricated cores. The methodology was verified through the fabrication of a 0.18 μm CMOS test chip. The methodology inefficiency factor (MIF) is introduced as metric for evaluating the area efficiency of a digital design flow.

1. INTRODUCTION

A large number of existing and emerging communications applications use multicarrier modulation or orthogonal frequency domain modulation (OFDM) for data modulation. These applications include high-speed data communication over copper [1], video broadcasting [2], in-building or fixed wireless LAN networking [3], and power line communication. Many of these applications are implemented using a system-on-a-chip (SoC) solution.

The fast Fourier transform (FFT) is the central arithmetic kernel of OFDM systems. Quick and accurate performance prediction of a variety of specific FFT configurations is valuable in the multicarrier SoC design process. The FFT is briefly described below followed by the discussion of an empirical estimation methodology and its evaluation with a 0.18 μm CMOS test chip.

2. BACKGROUND

In multicarrier modulation, data signals are modulated on a number of carriers rather than on a single carrier as in traditional AM or FM radio systems. Typically, the carriers are orthogonal and the scheme is referred to as OFDM. The FFT can be used to efficiently perform the modulation of data onto orthogonal carriers.
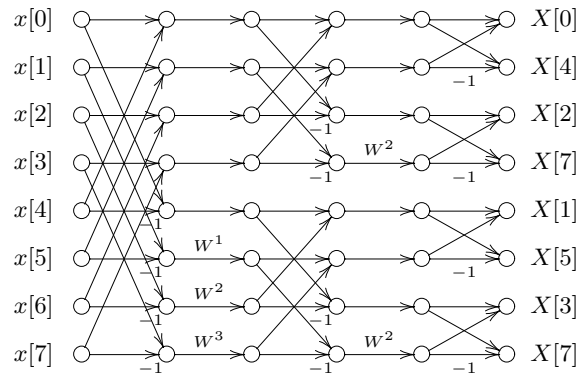
Figure 1: Signal flow graph of an 8-point, decimation-in-frequency, radix-2 FFT.

2.1. Fast Fourier Transform

The discrete Fourier transform (DFT) is a mapping of a discrete set of samples (or points) from the time domain to the frequency domain. A well-defined inverse mapping, the inverse FFT (IFFT) also exists. The N-point DFT and the inverse DFT (IDFT) are defined as

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \quad \text{and} \qquad (1)$$

$$x[n] = \sum_{n=0}^{N-1} X[k] W_N^{-kn} . \qquad (2)$$

The symbol n denotes the discrete time index, k is the discrete frequency index, $x[n]$ is the time-domain sequence, $X[k]$ is the frequency-domain sequence, and $W_N = e^{-j(2\pi/N)}$ is the complex-valued twiddle factor where $j = \sqrt{-1}$. The DFT and IDFT implemented naively as indicated by (1) and (2) have $\mathcal{O}(N^2)$ time complexity. The name FFT refers to a class of algorithms that efficiently implement the DFT in $\mathcal{O}(N \log N)$ time. The efficiency of FFT algorithms comes from the fact that the DFT can be recursively decomposed into smaller DFTs until trivial one-point DFTs are reached. The signal flow graph of an 8-point FFT is displayed in Figure 1. The open circles represent complex-valued addition, and a number beside a directed edge indicates complex-valued multiplication. Reference [4] provides a comprehensive introduction to the FFT.

I-583
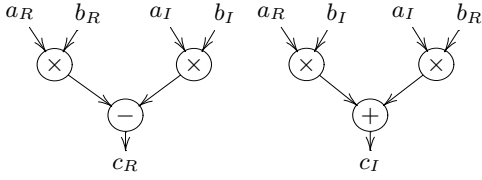
(a) 2-adder, 4-multiplier implementation.
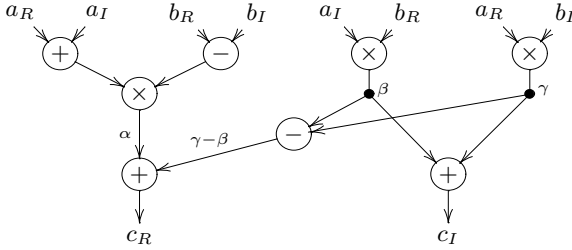
(b) 5-adder, 3-multiplier implementation.

Figure 2: Data flow in two complex multiplier implementations from real-valued components.

2.2. Multicarrier Modulation

At the transmitter of an OFDM system, data are apportioned in the frequency domain and an IFFT is used to modulate the data into the time domain. The FFT output data are guaranteed to be real-valued if conjugate symmetry is imposed on the input data. In the receiver, an FFT is used to recover the original data. The FFT allows an efficient implementation of modulation of data onto multiple carriers [5]. Due to the similarity between the forward and inverse transform apparent in (1) and (2), the same circuitry, with trivial modifications, can be used for both modulation and demodulation in a transceiver.

2.3. A Plethora of Configurations

The are a variety of algorithmic, architectural, resource allocation and scheduling alternatives in the design of an FFT. Algorithmic choices in FFT implementations including whether the recursive decomposition (or decimation) is performed in time or in frequency, the radix (the number of sub-DFTs per decomposition stage). Radix-2, radix-4, radix-8, split-radix (a combination of radix-2 and radix-4 approaches) and even mixed-radix implementations are possible [4]. After algorithmic choices have been made, there are a variety of architectural choices. An example of the choices at the architectural level is the implementation of the multiplication operation. One possible decomposition of the complex multiplication $c_R + jc_I = (a_R + ja_I) \times (b_R + jb_I)$ is the straightforward decomposition displayed in Figure 2(a) that requires 4 real-valued multipliers and 2 real-valued adders. Alternatively, as depicted in Figure 2(b), the intermediate products $\alpha = (a_R + a_I)(b_R - b_I)$, $\beta = a_I b_R$ and $\gamma = a_R b_I$ are computed, then the result is $c_R + jc_I = (\alpha - \beta + \gamma) + j(\beta + \gamma)$. The latter implementation potentially reduces overall area depending on the relative area of the multipliers and adders, whereas the former im-

plementation is faster due to the shorter critical path. A CORDIC implementation of the multiplier/rotator may be a good fit because the twiddle factor multiplier, $W_N = e^{-j(2\pi/N)}$, is a rotation in the complex plane.

There are three main classes of FFT resource allocation and schedulings: i) pipeline (row-oriented) FFTs, ii) column FFTs, and iii) fully-parallel FFTs [6]. In categories i) and ii) a portion of the FFT flow graph is implemented and the computation is scheduled on those resources. Extra resources are usually required to enable to the proper scheduling. In pipeline FFTs, for example, FIFO queues of varying sizes are needed to properly synchronize the incoming data.

The vast array of implementation options for FFTs at the algorithmic, architectural, and resource allocation and scheduling levels is a strength due to the flexibility and a drawback because comparing alternatives has traditionally been done at a very coarse-grained level. However, the basic building block of the FFT, the butterfly, and the regular interconnection of butterflies make the FFT highly amenable to the empirical prediction technique presented in the paper and in turn enable the comparison of a diverse set of implementations.

2.4. Analytical Evaluation

Traditionally, the discovery of a new FFT algorithm is characterized in terms of the number of adders, multipliers and the total memory footprint. A qualitative analysis of the routing sometimes given. Unfortunately, there is no clear mapping from the analytical description to VLSI performance estimates, making the evaluation of tradeoffs between various algorithms and architectures difficult.

Analytical investigation has shown that FFT VLSI implementations have an optimal asymptotic lower bound area·time^2 complexity of $\Omega(N^2 \log^2 N)$ [7]. This analytical technique is valuable for a coarse-grained comparison of various algorithms and architectures but the asymptotic nature of the bound makes it difficult to use as an evaluation or estimation tool for detailed (VLSI) implementations.
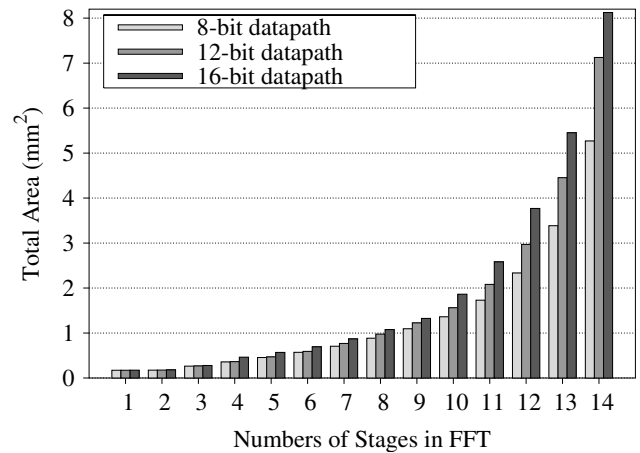

Figure 3: Area comparison for 8-, 12- and 16-bit radix-2 FFT implementations.
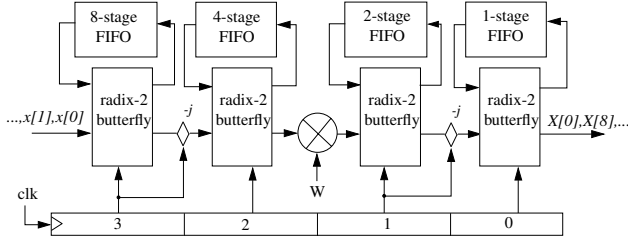
Figure 4: Block diagram of 16-point radix-2^2 pipeline FFT [8]. The control is implemented by a 4-bit counter.

Parameter	Units	Estimated	Measured
Min. Period	ns	6	8.8
Power	mW/MHz	3.3	2.3
Area	mm^2	2.07	4.38

Table 1: 16-bit, 1024-point radix-2^2 estimates and test results.

3. EMPIRICAL BOTTOM-UP ESTIMATION

The shortcomings of the analytical evaluation and estimation methods can be overcome by using empirical prediction. The regular structure of FFT implementations makes them particularly amenable to these methods. FFT blocks are composed of real-valued adders, real-valued multipliers, commutators, and FIFO queues (implemented by an array of registers or SRAM with simple collar circuitry). A library of adders, multipliers, commutators, and register arrays were synthesized, placed, and routed for a variety of bit resolutions in a 0.18 μm 1P6M CMOS process. The design flow was automated using scripts; therefore, the effort required to re-generate these components for a different technology is minimal. In addition, a memory compiler was used to generate estimates for all possible single- and dual-port, power-of-two address sizes with several bit resolutions. A total of 155 memory estimates were generated.

The speed, area, and power consumption data for the library components were compiled in a spreadsheet to generate estimates for FFT blocks. The timing estimates were generated by assuming a course grained pipelining between blocks (in addition, some of the multiplier blocks were internally pipelined). The minimum clock period is therefore equal to the longest flop-to-flop path including latching overhead. The power consumption and area are calculated by adding up the values for each individual component. Once the subcomponent data have been garnered, estimation of a large number of specific architectures and a variety of word sizes and FFT lengths can be generated in a matter of minutes. Figure 3 provides an example of the type of data that can be generated. The figure depicts the area required for radix-2 FFT implementations with datapath bit widths of 8-, 12-, and 16-bits for a range of FFT lengths (the length of the FFT is $N = 2^q$ where q is the number of stages).

Relative accuracy of the performance predictions for comparison of FFT implementations is inherent in the method of generation of the library components. A test chip was fabricated to verify the *absolute accuracy* of the estimation methodology.

4. RADIX-2^2 PIPELINE FFT TEST CHIP

For OFDM applications with a large number of carriers and real-time requirements, a pipeline FFT is a popular architecture. A 16-bit, 1024-point implementation of the radix-2^2 pipeline FFT presented in [8] was implemented in a 0.18 μm test chip to verify the estimation technique. A block diagram of a shorter, 16-point, radix-2^2 pipeline FFT is depicted in Figure 4 excluding twiddle factor circuitry. The same architecture can be extended to any number of stages. This architecture features fewer multipliers than a traditional radix-2 pipeline FFT. The basic observation that motivated this pipeline architecture is that the complex-valued multiplications in the FFT can be reorganized in the signal-flow graph. In the radix-2^2 FFT the multiplications are organized so that only every other stage contains nontrivial complex multiplications (i.e. terms of the form W^α). Complex multiplication by $-j$ is trivial since it can be implemented in digital hardware by simple inversion and cross wiring.

The empirical bottom-up estimates and measured performance parameters for the test implementation are listed in Table 1. The timing estimates do not take into account the clock tree insertion delay, which, in the final chip, was rather large due to the chip's global clock pad driver. The estimated clock insertion delay of 1.2 ns was subtracted from the measured minimum period.

The timing and power estimates are reasonably accurate given that they are available before HDL code has been written. The discrepancy in the minimum period estimate occurs because the subcomponents are each separately synthesized whereas the chip was synthesized in a single top-down compile (which is appropriate for the size of the block). An empirical cumulative distribution plot of the (flop-to-flop) slack is displayed in Figure 5. The slack in a flop-to-flop timing path is the difference between the period and the amount of time actually used. A positive amount of slack in a path serves as a guard interval, whereas a negative slack indicates a timing violation. A circuit with one ore more paths with negative slack cannot operate at the target frequency. It is evident from the slack plot that, after layout, the paths are pushed to the left, reducing slack with respect to the pre-layout paths. This timing degradation does not occur for the relatively small library components that are easily placed and routed under their respective constraints. The timing degradation in the FFT block is due to the lack of placement information in the synthesis tool that was used to generate the pre-layout netlist. However, the magnitude of the pre- versus the post-layout mismatch is not unusual a 0.18 μm CMOS technology.

The power consumption of each subcomponent was separately simulated assuming a 50% input transition probability which is somewhat higher than the actual transition probabilities. This accounts for the slightly pessimistic power consumption estimate.

Area	Predicted (mm^2)	Actual (mm^2)	MIF
Memory blocks	1.64	3.03	1.9
Standard cells	0.43	1.35	3.1
Empty Space	—	2.17	—

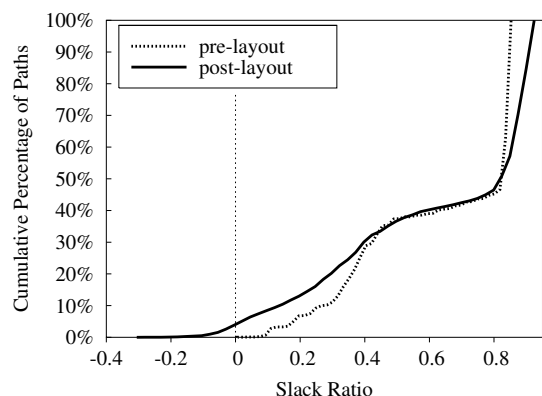Table 2: Area breakdown and MIF for the FFT chip.

Figure 5: Slack ratio empirical cumulative distribution plot for FFT chip. The slack ratio is the path slack time divided by the target clock period (6.7 ns in this case).
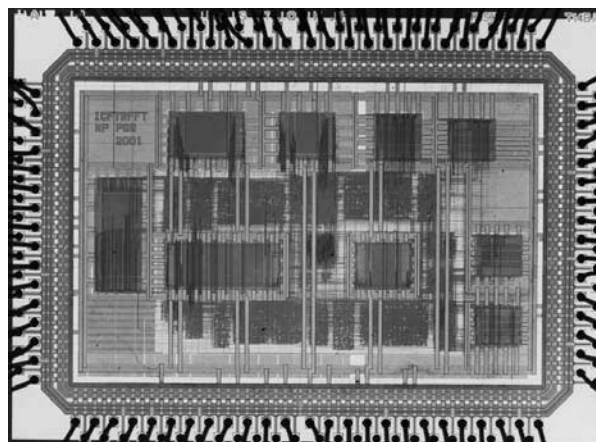

Figure 6: Bonded die photomicrograph of 16-bit, 1024-point radix-2^2 FFT chip. The chip die area is 4.3 mm $\times$ 3.1 mm.

4.1. Methodology Inefficiency Factor

The final area of a digital chip is highly dependent on the amount of design time spent performing place and route and the quality of the digital design flow methodology. Since each subcomponent was separately placed and routed, the subcomponent area estimates are lower bounds and thus the final area estimate can be viewed as a baseline area. The ratio of the final layout area to the estimated baseline area for a given FFT block is used to evaluate and characterize the design methodology used to produce the block. We call this ratio the methodology inefficiency factor (MIF), with MIF $= 1.0$ being the ideal. The MIF for the standard cells and memory blocks are displayed separately in Table 2 (the chip die photo is displayed in Figure 6). The large memory MIF is due to the power rings surrounding the memories (which are included in the memory area). A more refined power grid layout could significantly reduce the memory MIF. Nevertheless, the memory MIF is a good characterization of the current memory power ring methodology. The standard cell MIF is large because space was used liberally due to design time constraints.

Once a chip design has been completed in a given technology, and an MIF has been calculated, the area of future designs within the same class of applications is predictable. Since the memory MIF and the standard cell MIF are separately calculated, implementations with a range of RAM usage can be have their area more precisely estimated.

5. CONCLUSION

An empirical, bottom-up estimation methodology was presented. First, a library of FFT subcomponents in a 0.18 μm technology was created. Then, post-layout area, speed, and power data of FFT library components were used to generate the FFT performance estimates. The power and timing estimates for a specific test im-

Estimated	Pre-layout	Post-layout	Measured
6.0 ns	6.7 ns	7.9 ns	8.8 ns

Table 3: Clock period at various design stages.

plementation were shown to be relatively accurate. In the case of area estimation, the predicted value serves as a baseline area, and the MIF metric can be used to characterize and evaluate a digital design flow. Finally, the estimation method is also applicable to locally connected structures related to the FFT, including bitonic sort, convolution, polynomial multiplication, and the Viterbi algorithm [9]. A similar methodology could also be applied to Galois field arithmetic components for error control codes and cryptography systems.

6. REFERENCES

[1] International Telecommunication Union (ITU). *Asymmetrical Digital Subscriber Line (ADSL) Transceivers*, ITU-T, Rec. G.992.1 edition, February 1999.

[2] European Telecommun. Standards Inst. *Digital Video Broadcasting (DVB): Framing structure, channel coding and modulation for digital terrestrial television*, draft EN 300 744 V1.2.1 (1999-01) edition, January 1999.

[3] LAN/MAN Standards Committee of the IEEE Computer Society. *High-speed Physical Layer in the 5 GHz Band*, IEEE Std 802.11a-1999 edition, 1999.

[4] P. Duhamel and M. Vetterli. Fast Fourier transforms: a tutorial review and a state of the art. *Signal Processing*, 19:259–299, April 1990.

[5] John A. C. Bingham. Multicarrier modulation for data transmission: An idea whose time has come. *IEEE Communications Magazine*, 28(5):5–14, May 1990.

[6] Weidong Li and Lars Wanhammar. A pipeline FFT processor. In *IEEE Workshop on Signal Processing Systems*, pages 654–662, 1999.

[7] Clark D. Thompson. Fourier transforms in VLSI. *IEEE Transactions on Computers*, C-32(11):1047–1057, November 1983.

[8] Shousheng He and Mats Torkelson. A new approach to pipeline FFT processor. In *10th Parallel Processing Symposium*, pages 766–770, 1996.

[9] P. Glenn Gulak and Thomas Kailath. Locally connected VLSI architectures for the Viterbi algorithm. *IEEE Journal on Selected Areas in Communications*, 6(3):527–537, April 1988.