A Design Environment for High Throughput, Low Power Dedicated Signal Processing Systems

W. Rhett Davis, Ning Zhang, Kevin Camera, Fred Chen,
Dejan Markovic, Nathan Chan, Borivoje Nikolic, Robert W. Brodersen

University of California at Berkeley
Berkeley, California , USA

Abstract

A hierarchical automated design flow for low-energy direct-mapped signal processing integrated circuits is presented. A modular framework based on a combined Simulink and floorplan description drives automatic layout generation. Automatic characterization of layout improves system-level estimates. The flow is demonstrated on the subsystems of CDMA and OFDM receivers and a 300k transistor test-chip.

Introduction

The signal processing architectures that are the most efficient in terms of power and area can be obtained by directly mapping the algorithms into hardware. In this way, the maximum parallelism can be obtained, allowing the minimum clock rate and supply voltage to be used, resulting in reduced energy per operation. Direct-mapped design requires an initial description which allows specification of algorithmic parallelism as well as control structures. Typical metrics that can be achieved with this approach are computational efficiencies of 100-1000 MOPS/mW with computation densities of 100-1000 MOPS/mm^2 . These efficiencies are 2 to 3 orders of magnitude higher than can be achieved by software processors (1). This efficiency makes these architectures especially attractive for wireless systems.

In spite of the enormous advantage of this type of architecture, it is not commonly used unless the application cannot be accomplished by any other means. Direct-mapped IP cores for some communication system components (such as Viterbi decoders) are readily available, but most of today's wireless devices are based on programmable solutions. Sequential execution destroys the parallelism inherent in algorithms and wastes the opportunity to save power by lowering the supply voltage. The complexity of algorithms for future wireless receivers is so great that they are not presently feasible with programmable solutions (2).

Direct mapping is typically avoided because of the indeterminism of design time for such highly dedicated chip solutions. This is due in part to the unpredictability of timing closure (3) but is exacerbated by the lack of a consistent flow from an initial system level description through to the physical layout. A standard design flow has three phases which are typically handled by three different engineering teams. System designers conceive the chip and deliver a specification to ASIC designers, often in the form of a Matlab or C simulation. This system level description can be used in a simulation to generate system characterizations such as a bit-error-rate vs. signal-to-noise ratio (BER vs. SNR) curve for a communications chip. The ASIC designers map the simulation to VHDL or Verilog code that they deliver to the physical design team. Physical designers synthesize the code to a standard-cell library and use place route tools to generate layout mask patterns for fabrication. This procedure requires 3 translations of the design with requirements for re-verification at each stage. Opportunities for algorithmic modifications to reduce power and area are lost, since the system designer does not have information about physical performance bottlenecks.

Recent studies of improved design methodologies stress ASIC designer control over physical structure (4) and early floorplanning (5). To achieve the full benefit of direct-mapping, however, control of the physical structure and early floorplanning is needed by system designers as well. To accomplish this, a new kind of design environment is needed which offers fast, automatic generation of physical information from a system-level description. This contrasts the majority of recent work, such as (3), which focuses on automatic generation of physical information from register-transfer level (RTL) descriptions.

The goal of the design environment described here is to take an initial description that is appropriate for a signal-processing algorithm (including data-flow and control) and to automatically generate a direct mapped architecture. The MathWorks' tool Simulink was chosen as the basis of this description because it captures algorithmic parallelism and is tightly integrated with the popular system design tool Matlab. A framework for module generators aids the creation of raw physical design information, and a design flow for fast creation of mask-layout is supported. This paper presents this framework and flow with some examples of their usage.

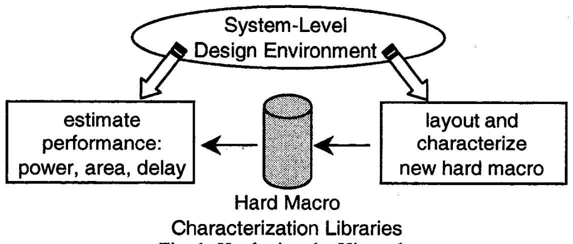
Chip-in-a-Day Design Flow

This environment accelerates the design process by automating the design flow to mask layout from a single description. This description encapsulates the important decisions typically made by system, ASIC and physical designers separately. These design decisions are divided into the following types:

- *Function Decisions* – basic block diagram and input-output behavior of each block
- *Signal Decisions* – physical signals specifications
- *Circuit Decisions* – specification of the transistors to implement each block
- *Floorplan Decisions* – physical location specifications

By pulling all of these decisions into a single description, system, ASIC and physical designers can more easily come together to make the trade-offs necessary to optimize the design. In order for the designers to understand these trade-offs, an accurate method of estimating power, area and delay from the unified description is needed. To facilitate these estimates, our design flow assumes that each functional block corresponds to a unit of layout (sometimes called a "hard macro") which has been extracted and characterized for power, area, and delay. These functional blocks are the leaf-cells of the design hierarchy and will be simply called "macros" for the remainder of this paper.

As the design progresses, the system becomes larger, and interconnect capacitance makes these estimates less accurate. We therefore use the automated flow to produce layout, extract and characterize portions of the design hierarchy. This process is called "hardening the hierarchy". As illustrated in Fig. 1, hardening creates a new hard macro which can be instantiated in the design to improve estimation accuracy. The design converges manually as the hierarchy is hardened rather than relying on a timing-driven flow to manage constraints. Once the entire design hierarchy is hardened in this way, the chip is ready to be fabricated.


System-Level Design Environment
estimate performance: power, area, delay
layout and characterize new hard macro
Hard Macro Characterization Libraries
Fig. 1: Hardening the Hierarchy

Signal processing architectures are typically data-path heavy. Control logic accounts for less than one-tenth the power and area of the data-path logic for baseband signal processing in wireless sytems. Simulink offers a number of primitives such as adders, multipliers, and switches which allow easy description of a data-path. Fig. 2 shows shows a simple Simulink implementation of an Add/Compare/Select (ACS) block from a Viterbi decoder. In addition to data-path primitives, Simulink also offers a control primitive called Stateflow. Stateflow is a graphical language for describing control flow with state machines. Fig. 3 shows an example of a simple Stateflow chart. Using the data-path and control primitives provided with Simulink, it is possible to create a discrete-time simulation which completely specifies the behavior of a system.

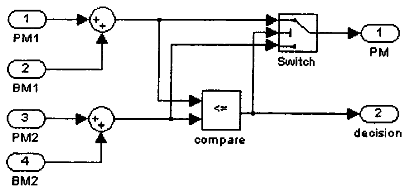


Fig. 2: Simulink implementation of an ACS block from a Viterbi decoder

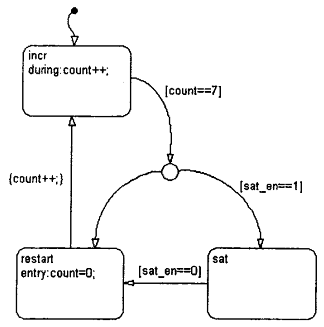


Fig. 3: An example of a Stateflow chart that counts from 0 to 7 and restarts from zero if the input signal *sat_en* is low or stays at 7 while *sat_en* is high

Signal decisions are made through the use of Simulink's Fixed-Point Blockset. This blockset allows the replacement of normal Simulink primitives with versions that truncate values according to fixed-point types of arbitrary word-lengths. These signal decisions are captured by a tool developed in-house which creates a hierarchical EDIF file from a Simulink system. Another type of signal-level optimizations is use of multiple clock domains. The design flow recognizes the Simulink Enable block as a gated clock specification and builds a clock tree automatically.

Circuit decisions are encapsulated through the specification of a parameterized generator for each macro. Simulink parameters can be passed to the macro generators, thus allowing exploration of design trade-offs from Simulink. Each generator produces a netlist of cells for which layout and schematic views exists. There are three main types of macros supported by our flow:

- *Data-path Macros* – Ranging in complexity from arithmetic blocks (such as adders, registers, and multpliers) to complex pipelines, these macros are typically implemented with Synopsys' Design Compiler or Module Compiler or semi-custom tiled layout generators.
- *Stateflow Macros* – Stateflow charts are translated to VHDL by a tool developed in-house. This VHDL code is then synthesized with Design Compiler.

- *Composite Macros* – These are portions of a design hierarchy which have been hardened.

Specification of physical placements and other floorplan decisons in Simulink is inconvenient, so one additional input description, the floorplan, is required. The Simulink system and floorplans are developed side by side. When a block is added in Simulink, the macro generators execute automatically, and an unplaced block will appear in the floorplanning tool. The main design decisions encapsulated by the floorplan are the boundary for the cell and the standard-cell rows for automatic placement.

Timing Issues

One of the greatest difficulties of digital design is timing closure. Timing closure is the process of making sure that the layout meets the timing constraints assumed in the initial description. The difficulty arises from the fact that interconnect capacitance is unknown at design time. The most common problems that arise during hardening are failure to meet cycle-time goals and generation of clock-trees with excessive skew

Cycle time goals can be met by improving the accuracy of pre-layout delay estimates. Several aspects of our flow allow us to improve this accuracy. First, early specification of the floorplan provides detailed placement information to estimation tools. This placement information can be used to estimate interconnect capacitance more accurately than statistical wire-load models. When combined with macro timing models, these capacitance estimates improve delay estimates. Secondly, our focus on reduced supply voltages and low clock rates reduces the significance of distributed RC effects. Fig. 4 shows simulated results for an inverter with a distributed RC load. For lower supply voltages, the delay at the driver (logic delay) grows while the additional RC delay of the interconnect remains relatively constant.

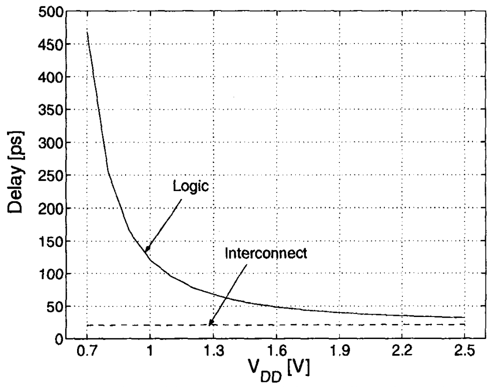


Fig. 4: Logic and interconnect delay scaling for an inverter with a distributed RC load in a 0.25 μm technology.

A tool is being developed in-house for the purpose of automatic, hierarchical insertion of the clock tree during hardening. This tool uses a tree topology because the lower wire capacitance leads to less power consumption. This approach does require optimization and balancing. The clock is inserted into the floorplan and balanced for a target non-zero skew. The amount of allowed clock skew is determined by the inherent race immunity of the flip-flops, which is given by the difference between the clock-to-output delay and the hold time. For lower supply voltages, flip-flops are more race immune, allowing more clock skew. The cycle-time penalty of large skew is small, given the low clock rates of interest.

Examples

To demonstrate the power of this approach, we present here five complex macros and a test-chip designed with simple macros. All examples were designed in Simulink out of macros for which the generator had already been established as functionally equivalent to its corresponding Simulink model. Test-vectors from the Simulink simulation were saved and used for power estimation.

At this point, the automated flow takes over and invokes each macro generator. The resulting hierarchy is then hardened by merging the placement information in the floorplan and routing with Cadence's IC Craftsman. The resulting layout is verified with Calibre design rule and layout vs. schematic checks (DRC & LVS), and parasitics are extracted with Arcadia. EPIC PowerMill simulations of the extracted netlist then characterize the power consumption of the layout using the Simulink test-vectors, and EPIC PathMill finds the critical-path delay.

The macros presented here were designed with Synopsys' Module Compiler to be used in one of two receiver systems under investigation for indoor wireless networks. The first is a CDMA system described in (6). The synchronization subsystem shown in Fig. 5. is intended to provide coherent timing recovery and code acquisition for a stream of soft symbols. The white blocks in Fig. 5 were hardened with results given in Table 1. The results show large discrepancies of critical-path delays. These discrepancies represent flexibility to the system designer when making performance trade-offs.

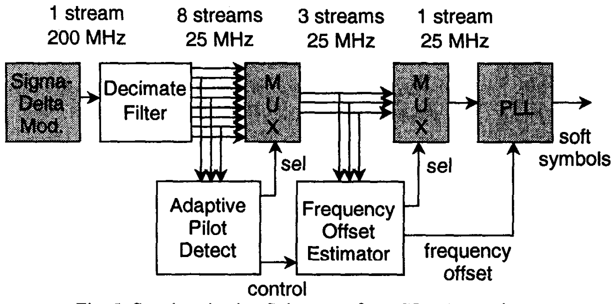


Fig. 5: Synchronization Subsystem for a CDMA Receiver

Table 1: Results of hardening the CDMA system macros

		decimation filter	adaptive pilot detect	frequency offset estimator
area in 0.25 μm		1.4 mm^2	1.2 mm^2	0.22 mm^2
power @ 25 MHz	2.5 V	120 mW	88 mW	5.2 mW
	1.0 V	13 mW	7.6 mW	0.47 mW
critical-path delay	2.5 V	5.6 ns	10 ns	5.2 ns
	1.0 V	18 ns	32 ns	17 ns
cells		21 k	3500	15 k
transistors		240 k	37 k	220 k

The second system under investigation is an orthogonal frequency-division multiplexing (OFDM) system with 128 sub-carriers with a target aggregated sample rate of 25 Msamples/sec. Two dominant computational blocks in an OFDM receiver are fast Fourer transform (FFT) and Viterbi decoder (7). A pipelined FFT architecture with 16-bit precision and single-path delay feedback was used for the macro in Table 2. The 64-state, 8-level-soft-input Viterbi decoder in Table 2 has a parallel state metric update unit, which uses modulo arithmetic based normalization scheme. The register-exchange method is used for survivor path decoding.

Table 2: Results of hardening the OFDM system macros

		FFT	viterbi decoder
area in 0.25 μm		1.4 mm^2	0.71 mm^2
power @ 25 MHz	2.5 V	150 mW	69 mW
	1.0 V	16 mW	7.0 mW
critical-path delay	2.5 V	20 ns	4.8 ns
	1.0 V	63 ns	15 ns
cells		19 k	10 k
transistors		270 k	130 k

All results listed here were generated in less than 24 hours for each macro. Smaller macros ran through the flow in 3 hours. Most of the time is spent routing and extracting.

To verify the approach of this flow, a test-chip was developed based on the decimation filter block described earlier. Unlike the complex macro above, the test chip was designed with several hundred hard macros placed manually in 3 layers of physical hierarchy. The design contained 307 k transistors and was fabricated in a 0.25 μm technology. Table 3 shows the performance evaluation of the chip at 1.0 V and 25 MHz. The table shows very close agreement between Simulink and layout estimates and measured values of power and delay. The area discrepancy arises from the fact that the hard macros could not be abutted, leading to a wasted space. The performance of this chip relative to the Module Compiler macro indicates that this level of detailed floorplanning is not necessary for this particular structure.

Table 3: Comparison of test-chip performance estimates

	Simulink	layout	actual
area	2.0 mm^2	6.8 mm^2	6.8 mm^2
power	16 mW	17 mW	14 mW
criti.-path delay	19 ns	21 ns	in progress

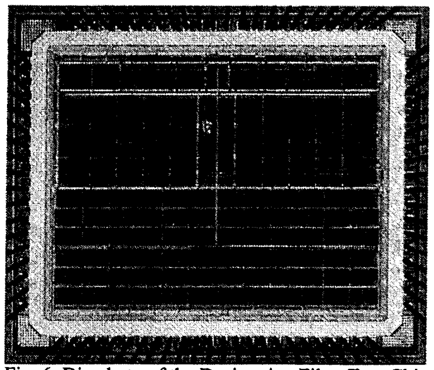


Fig. 6: Die photo of the Decimation Filter Test-Chip

Conclusions and Future Work

The success of the test-chip and ease of the macro hardening flow are encouraging. The next step is to apply the flow to the design of systems in the 1M to 10M transistor range. The most difficult aspect of this flow is the verification of functional equivalency of macro generators and their Simulink models. As macros become more complex, more opportunities for discrepancy arise, leading to potential problems when macros are combined. Future work will focus on comparisons of the estimates gained from this approach to estimates made with other system-level design methods. Also, much more investigation is needed into the level of detail needed during floorplanning and into which macro granularities scale best to future process generations.

Acknowledgements

Many thanks to Cadence Design Systems, Ericsson and Lucent for their support, and to ST Microelectronics for fabricating our test chip. Thanks to Hayden So, Ben Coates, and Dave Wang for their help developing the flow, and to Paul Hustead, Josie Ammer, and Mike Sheets for using the flow and providing valuable feedback.

References

(1) R. Brodersen, "The network computer and its future," *Proc. of the 1997 International Solid State Circuits Conference.*

(2) N. Zhang, *et al.*, "Trade-offs of performance and single chip implementation of indoor wireless multi-access receivers," *Proc. of the 1999 Wireless Communications and Networking Conference.*

(3) D. E. Lackey, "Applying placement-based synthesis for on-time system-on-a-chip design," *Proc. of the 2000 Custom Integrated Circuits Conference.*

(4) W. J. Dally, A. Chang, "The role of custom design in ASIC Chips," *Proc. of the 2000 Design Automation Conference* (DAC).

(5) J. S. Yim, *et al.*, "A floorplan-based planning methodology for power and clock distribution in ASIC's," *Proc. of the Design Automation Conference*, New Orleans, LA, 1999.

(6) C. Teuscher, *et al.*, "Design of a Wideband Spread Spectrum Radio Using Adaptive Multiuser Detection," *Proc. of the 1998 International Symposium on Circuits and Systems.*

(7) N. Weste, D. J. Skellern, "VLSI for OFDM," *IEEE Communications Magazine*, p.127-31, vol.36, (no.10), Oct. 1998.