An Open-source Framework for Autonomous SoC Design with Analog Block Generation

Tutu Ajayi*, Sumanth Kamineni†, Yaswanth K Cherivirala*, Morteza Fayazi*, Kyumin Kwon*, Mehdi Saligane*, Shourya Gupta†, Chien-Hen Chen†, Dennis Sylvester*, David Blaauw*, Ronald Dreslinski Jr*, Benton Calhoun†, David D. Wentzloff*

* University of Michigan, Ann Arbor, MI
† University of Virginia, Charlottesville, VA

***Abstract*—We present the world's first autonomous mixed-signal SoC framework, driven entirely by user constraints, along with a suite of automated generators for analog blocks. The process-agnostic framework takes high-level user intent as inputs to generate optimized and fully verified analog blocks using a cell-based design methodology.**

Our approach is highly scalable and silicon-proven by an SoC prototype which includes 2 PLLs, 3 LDOs, 1 SRAM, and 2 temperature sensors fully integrated with a processor in a 65nm CMOS process. The physical design of all blocks, including analog, is achieved using optimized synthesis and APR flows in commercially available tools. The framework is portable across different processes and requires no-human-in-the-loop, dramatically accelerating design time.

***Index Terms*—analog synthesis, analog generator, SoC generator**

I. Introduction

There is an ever-growing need for automation in analog circuit design, validation, and integration to meet modern-day SoC requirements. Time-to-market constraints have become tighter, design complexity has increased and more functional blocks (in number and variety) are being integrated into SoCs. These challenges often translate to increased manual engineering efforts and non-recurring engineering (NRE) costs. We present FASoC, an open-source[1] framework for Fully-Autonomous SoC design. Coupled with a suite of analog generators, FASoC can generate complete mixed-signal system-on-chip (SoC) designs from user specifications. The framework leverages differentiating techniques to automatically synthesize correct-by-construction RTL descriptions for both analog and digital circuits, enabling a technology-agnostic, no-human-in-the-loop implementation flow.

Analog blocks like PLLs, LDOs, ADCs, and sensor interfaces are recasted as structures composed largely of digital components while maintaining analog performance. They are then expressed as synthesizable Verilog blocks composed of digital standard cells and auxiliary cells (aux-cells). We employ novel techniques to automatically characterize aux-cells and develop models required for generating bespoke analog blocks. The framework is portable across processes, EDA tools and scalable in terms of analog performance, layout, and other figures of merit.

The SoC generation tool translates user intent to low-level specifications required by the analog generators. To achieve full SoC integration, we leverage the IP-XACT [1] standard and added vendor extensions to capture additional meta-data from the generated blocks. This enables the composition of vast numbers of digital and analog components into a single correct-by-construction design. The fully composed SoC design is finally realized by running the Verilog through synthesis and automatic place-and-route (APR) tools to realize full design automation.

II. Framework Architecture

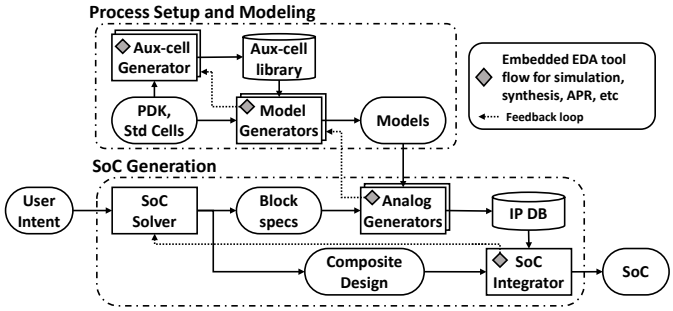

Fig. 1. FASoC Framework Overview

A high-level representation of the framework is shown in Fig. 1. The *Process setup and modeling* phase is performed once for the process design kit (PDK), and it involves the generation of the aux-cells and models for the generator. The *SoC generation* phase begins by translating high-level user-intent into analog specifications that satisfy the user constraints. The block generators are invoked as needed and the SoC integrator stitches the composed design and walks it through a synthesis and APR flow to create the final SoC layout. The FASoC framework is tightly integrated with analog generators for PLL, LDO, temperature sensor, and SRAM blocks. Section III describes the circuit architecture adopted by the different generators.

[1]Source code for the framework and all generators developed as part of this work can be downloaded from https://fasoc.engin.umich.edu

978-1-7281-5409-1/20/$31.00 ©2020 IEEE

A. Process Setup and Modeling

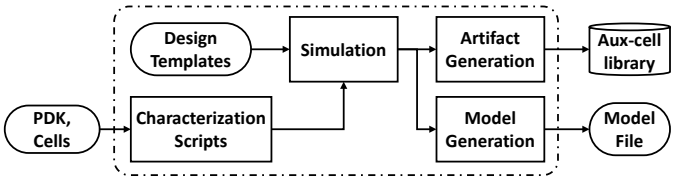

Fig. 2. Aux-cell and model file generation flow

FASoC employs a synthesizable cell-based approach for generating analog blocks, significantly cutting back on manual layout and verification efforts. Aux-cells are small analog circuits that buttress the standard cell library and provide specific analog functionality required by the generators. Each cell is no larger than a D flip-flop and can be placed on the standard cell rows. We simplify the creation of aux-cells by using a suite of design templates in tandem with PDK characterization scripts. The templates capture the aux-cell's precise circuit behavior without including any PDK-specific information. The characterization scripts operate on the PDK to derive technology-specific parameters required to set knobs within the templates. Example parameters extracted from the PDK include threshold voltage, metal parasitics, MOSFET behavior, and Fan-out of 4. The knobs set within the template include device type, transistor sizing, and other circuit design options. The results from aux-cell generation include the netlist, layout, timing library, and other files required to proceed with conventional synthesis and APR. At present, the layouts for the aux-cells are manually created, however, we are currently evaluating several layout automation tools [2]–[4] that are showing promising results. We find our template-based methodology for creating aux-cells enhances process-portability and significantly cuts down on design time. All of the generators presented in this work leverage 8 aux-cells that are depicted in Fig. 3.

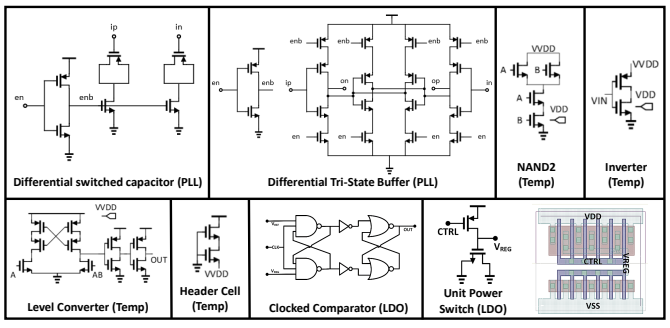

Fig. 3. Schematic for aux-cells used across PLL, LDO and temperature sensor generators

The analog generators use models to predict performance and select design parameters to create optimized block designs that satisfy the input specifications. The models are derived from the parameterized templates that incorporate the aux-cells. The models for each generator vary and are developed from a combination of mathematical equations, machine learning, and design space exploration. The modeling exercise is also performed once per PDK and the results are saved into a model file. Sections III briefly describes the modeling approach adopted by each generator integrated into the framework.

B. SoC Generator

This stage begins with an iterative *SoC solver* to determine the optimal *composite design* description which is a combination of blocks, analog specifications, and connections. The strategy is guided by high-level user intent (i.e. target application and power/area budgets), available analog block generators, and a database of IPs. Analog generators are invoked as necessary to generate bespoke blocks required to satisfy the specifications within the composite design. The generator outputs include all artifacts required to push the block through standard synthesis and APR tools. The outputs are also cached in an *IP database*, allowing for block generation to be skipped if a matching entry already exists. Entries in the database can also be populated with 3rd party IPs such as processors and other peripherals.

We adopt the IP-XACT format to describe the composite design as well as the block designs stored in the database. We use an extended format [5] to capture additional analog data, simulation, and verification information.

The *SoC integrator* begins by stitching the composite design together and translating it to its structural Verilog equivalent that can be run through digital simulation tools. The structural Verilog, along with all required artifacts from the database, is then passed through the embedded tool flow to generate the final verified GDS. This same flow is pervasive across the framework and is also used by all generators (aux-cell, model, and analog). Tools within the flow cover all aspects of chip design including SPICE simulations, digital simulations, synthesis, APR, DRC, LVS, and extraction.

III. Analog Generator Architecture

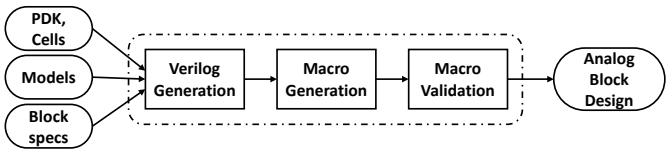

Fig. 4. Analog generator flow

Synthesizable analog blocks were introduced a few decades ago and have continued to evolve, closely matching the performance obtainable by full custom designs. Prior works have described techniques for synthesizing analog blocks for UWB transmitters [6], PLLs [7], DACs [8], and other types of analog blocks [9]–[11]. This approach lowers engineering design costs, increases robustness, eases portability across PDKs, and continues to show promise even at advanced process nodes [12]–[14]. The analog generators developed as part of this work can be likened to ASIC memory compilers

that take in a specification file and produce results in industry-standard file formats, which can then be used in standard synthesis and APR tools. Unlike typical memory compilers, our generators are open-source, process agnostic, and share a scalable framework amenable to different types of blocks. The framework is modular and share a similar process as depicted in Fig. 4. The full generation process is broken down into three steps:

Verilog Generation: This step leverages models to produce a synthesizable Verilog description of the block that conforms to the input specifications. It also generates guidance information in a vendor-agnostic format. The guidance includes synthesis constraints, placement instructions, and other information that may be required by the synthesis and/or APR tool to generate blocks that achieve the desired performance. In addition, this step also reports early estimates on performance and the characteristics of the block to be created.

Macro Generation: The Verilog and guidance information is passed to a digital flow to create macros that can be embedded into larger SoC designs. The digital flow in this step performs synthesis, APR, DRC, and LVS verification. The digital flow includes an adapter to translate the guidance into vendor-specific commands used in synthesis and APR. The adapter abstraction allows us to (1) express additional design intent without exposing protected vendor-specific commands and (2) easily support multiple cad tools including open-source alternatives [15]–[17]

Macro Validation: The last step is a comprehensive verification and reporting of the generated block. The full circuit goes through parasitic extraction, SPICE simulations, requirement checks and other verification to culminate in a detailed datasheet report.

The generators can be invoked standalone, outside of the full SoC generator flow. To simplify the system integration, we adopt the AMBA™ APB protocol as the register interface to all blocks. The following sub-sections briefly describe the analog generators currently integrated into the FASoC framework.

A. PLL

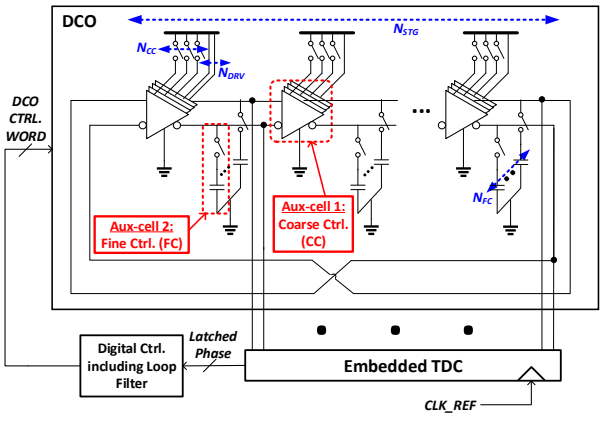

Fig. 5. DCO architecture indicating the aux-cells and designs parameters

The generated PLLs (Fig. 5) share the same base architecture as ADPLL [18]. The phase difference of the reference and output clocks are captured by the embedded time-to-digital converter (TDC), while the digital filter calculates the frequency control word for the digitally controlled oscillator (DCO). The input specification to the generator defines the nominal frequency range and in-band phase noise (PN). The PLL generator uses a physics-based mathematical model [19] for characterization. We first build a mathematical relationship between DCO design parameters (number of aux-cells and stages) and the required DCO specifications. Using simulation results from a parametric sweep, we then find the effective ratio of drive strength and capacitance for each aux-cell. This ratio enables us to predict frequency and power results (frequency range, frequency resolution, frequency gain factor, and power consumption) given a set of input design parameters.

B. LDO

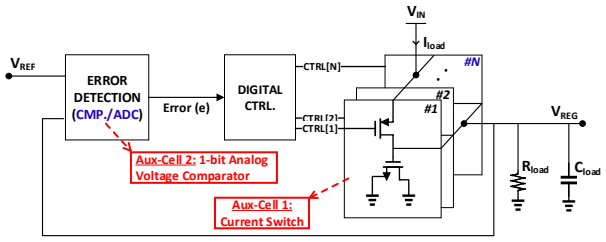

Fig. 6. LDO architecture indicating the aux-cells and design parameters derived from input specifications of V_{IN} , I_{load} and desired transients

The generated LDOs (Fig. 6) share the same base architecture as DLDO [20]. The LDO leverages an array of small power transistors that operate as switches for power management. Based on design requirements, the generator can swap the clocked comparator with a synthesizable stochastic flash ADC [21] to improve transient response. The input specifications to the LDO generator are the V_{IN} range, $I_{load,max}$ range, and the dropout voltage. The generator uses a poly-fit model of the load current ($I_{load,max}$) performance with respect to various combinations of aux-cell connections (connected in parallel and for different VDD inputs) in both ON and OFF states. We create the model by simulating various test circuits after parasitic extraction.

C. Temperature Sensor

The generated sensors (Fig. 7) share the same base architecture as [22]. The sensor relies on a temperature-sensitive ring oscillator and stacked zero-VT devices for better line sensitivity. The input specifications include the temperature range and optimization strategy, for either error or power. For a given temperature range, the models attempt to select the optimal circuit topology that minimizes error and/or performance. The generator relies on a predictive Bayesian neural network model to select design parameters that satisfy the input specifications.

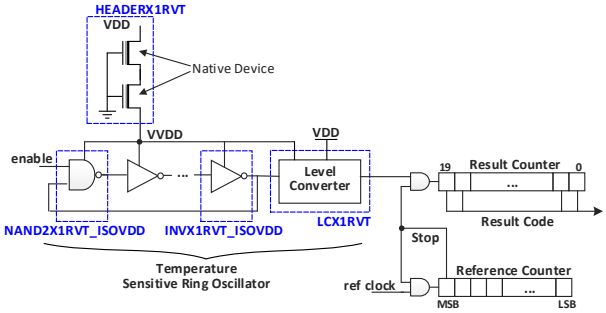

Fig. 7. Temperature sensor architecture indicating the aux-cells

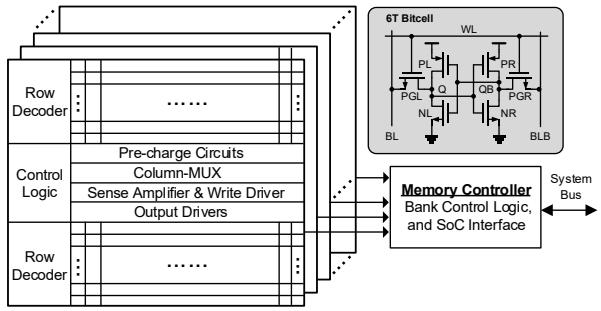

Fig. 8. SRAM architecture showing macros and bank strategy

D. SRAM

The compiled SRAMs (Fig. 8) follow a standard multi-bank memory architecture. Unlike other generators in the framework, the memory generator uses a combination of macros instead of aux-cells. The macros used include a 6T bitcell, a row decoder, column mux, wordline driver, sense amplifier, write driver, and a pre-charge circuit. The macros are stitched together, bottom-up, to form a bank. The user input specifications are capacity, word size, operating voltage, and operating frequency. The generator adopts a hierarchical meta compiler (HMC) [23] for technology characterization and a hierarchical memory model to determine the optimal row and column periphery. The model helps to select the SRAM architecture and the leaf-level components that best satisfy the user specifications while minimizing energy consumption and delay.

IV. Evaluation

The framework has been fully validated in a 65nm process. Our evaluation begins with a focus on the individual generators. We present results that explore the design-space possible with each generator and demonstrate full adherence to the user input specification. We then present results from a prototype SoC created using this framework.

A. Analog Generation Results

Fig. 9 presents the results of several PLLs generated using different input specifications. It compares the input requirements against the simulated results after parasitic extraction. The results show that the generated frequency ranges cover that of the input requirements and with better phase noise

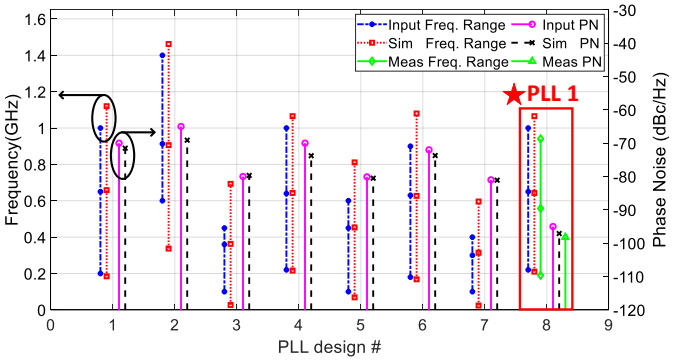

Fig. 9. Generated PLL designs for eight different input specifications. PLL1 is taped-out in the SoC prototype

levels. The highlighted PLL 8, corresponds to one of the PLLs integrated into the SoC prototype and also shows measured results that satisfy the given specifications

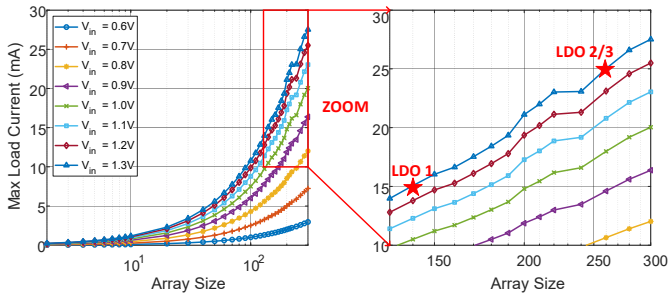

Fig. 10. $I_{load,max}$ vs. array size, for multiple LDO designs generated

Fig. 10 shows the spice simulation results of multiple LDO designs after parasitic extraction. The graph shows the maximum load current at different input voltages corresponding to the input parameter array size for a dropout voltage of 50mV. The highlighted measurements correspond to the input specification for blocks integrated into the SoC prototype with $V_{IN} = 1.3V$ and $V_{REG} = 1.2V$.

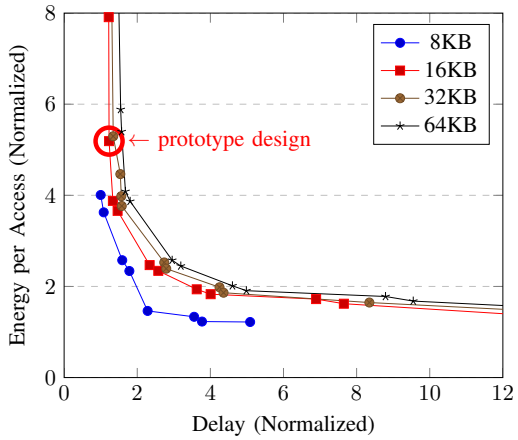

Fig. 11. Normalized energy and delay plots for various memory sizes while sweeping VDD.The results are normalized with respect to the 8KB memory.

Fig. 11 presents the simulation results of various memory capacities across a broad range of architectural options and operating voltages (VDD). Each point on the curve corresponds to an energy-delay pair specific to an architecture (rows, columns, and banks) and VDD combination. The generator selects the Pareto-optimal design that satisfies the user requirements. The highlighted point on the 16KB curve corresponds to the memory block integrated into the SoC prototype.

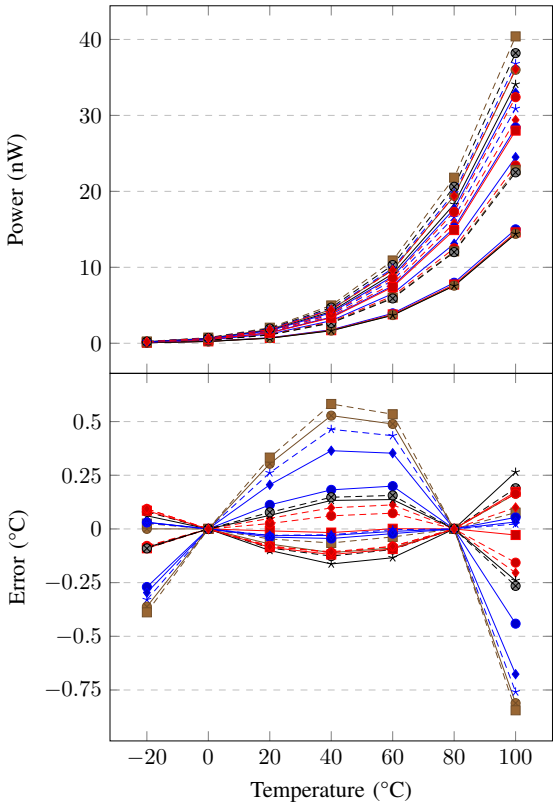

Fig. 12. Power and Error results against temperature for various temperature sensor designs (each fitted plot represents a unique design)

Fig. 12 shows the spice simulation results of multiple temperature sensor designs after parasitic extraction.

B. Prototype Chip Results

The prototype SoC design (Fig. 13) includes 2 PLLs, 3 LDOs, 1 16KB SRAM, and 2 temperature sensors fully integrated with an Arm® Cortex™-M0 in a 65nm CMOS process. Using off-chip connections, we were successfully able to power the entire SoC using one of the LDOs and clock it using the PLLs while monitoring the temperature of the chip.

Fig. 9 presents results for 8 PLL designs generated from different input specifications, including one from the prototype, and the results show output performances in-line with the input specifications. The measured frequency is 10% slower while the phase noise matches the simulation and specification requirement. Table I summarizes the results for all PLLs in the prototype.

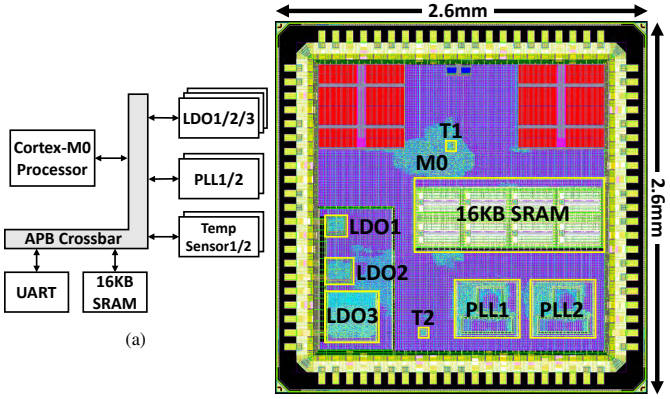

Fig. 13. Simplified block diagram (a) and annotated die photo (b) for the 65nm prototype SoC

TABLE I
PLL SIMULATION VS MEASUREMENT RESULTS

Output Specifications	PLL1		PLL2	
	Sim	Meas	Sim	Meas
Min Freq (MHz)	200	190	170	150
Max Freq (MHz)	1,060	920	1,080	930
F_{nom} (Mhz)	643	558	627	548
Power@F_{nom} (mW)	7.20	6.90	8.06	7.70
Area (μm^2)	167,639.04		167,639.04	

Table II shows the LDO $I_{load,max}$ measurements closely matching the input specification requirements. Compared to the comparator-based architecture (LDO1/2), the ADC based controller architecture (LDO3) achieves better transient performance with a 10x and 7x improvement in settling time and undershoot voltage respectively. The line and load regulation values are measured at V_{IN}=1.3V, V_{REF}=1.2V, and I_{load}=10mA. LDO3 load regulation is comparatively worse due to the high gain of the ADC based controller. As we operate at lower V_{REF} and I_{load} conditions, the line/load regulation degrades for all the LDOs because of the increase in relative switch strength.

The temperature sensor has an area of 2,620μm^2. A 2-pt calibration is performed at 0°C and 80°C. Measured results show a sensing range between -20°C and 100°C with an accuracy of ±4°C.

Fig. 14 summarizes the SRAM measured and simulated performance across the input operating voltage range of 0.8V to 1.2V. The SRAM peak performance is at 65MHz with the power consumption of 2.09mW at 1.2V, which exceeds the targeted frequency of 50MHz. The measured power for

TABLE II
LDO SIMULATION VS MEASUREMENT RESULTS @ 200MHZ CONTROL CLOCK

Output Specifications	LDO1		LDO2		LDO3	
	Sim	Meas	Sim	Meas	Sim	Meas
Dropout Voltage (mV)	50	70	50	80	50	80
$I_{load,max}$ (mA)	15.00	15.38	25.00	24.84	25.00	23.72
Settling Time - Ts (µs)	1.1	1.8	2.1	2.9	0.12	0.19
Max Undershoot (V)	0.35	0.98	0.57	0.98	0.38	0.14
Max Current Eff. (%)	94.2	96.4	95.7	94.5	81.9	74.0
Load Regulation (mV/mA)	-	-1.00	-	-0.35	-	-3.6
Line Regulation (V/V)	-	0.180	-	0.004	-	0.950
Area (μm^2)	17,318.56		31,187.56		127,163.56	

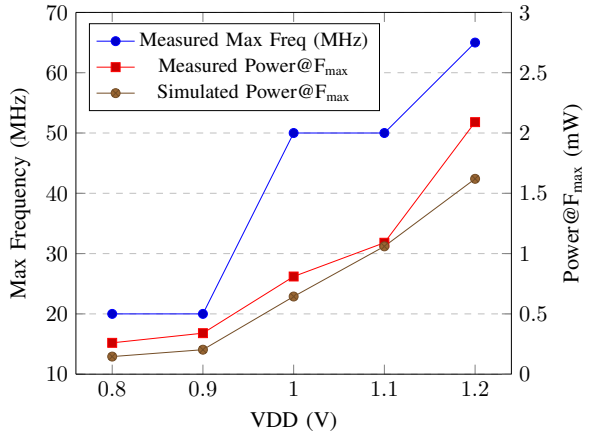

Fig. 14. Measured and simulated performance and power results of SRAM across VDD

the SRAM also include the leakage power of the processor and peripheral interface. The generated SRAM has an area of 0.68mm^2 with the custom bitcell area occupying 0.4mm^2.

V. Conclusion

We presented an autonomous framework that generates a completely integrated SoC design based on user input specifications. This framework is PDK agnostic and allows for faster turn-around times when building custom analog blocks and integrating them into larger SoC designs. The framework includes generators for PLL, LDO, temperature sensor and SRAM blocks. The framework can easily be extended to support more generators and different PDKs. We fabricated an SoC prototype in a 65nm process and presented silicon measurements to validate the framework's accuracy. Our work establishes a new milestone in creating a silicon compiler [24] that further reduces the complexity of realizing modern SoCs and cuts down on design time.

Acknowledgment

This material is based on research sponsored by Air Force Research Laboratory (AFRL) and Defense Advanced Research Projects Agency (DARPA) under agreement number FA8650-18-2-7844. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

[1] Accellera, "IP-XACT - Accellera." https://www.accellera.org/downloads/standards/ip-xact. Last accessed 2020-05-03.

[2] C.-Y. Wu, H. Graeb, and J. Hu, "A pre-search assisted ilp approach to analog integrated circuit routing," in *2015 33rd IEEE International Conference on Computer Design (ICCD)*, pp. 244–250, IEEE, 2015.

[3] K. Kunal, M. Madhusudan, A. K. Sharma, W. Xu, S. M. Burns, R. Harjani, J. Hu, D. A. Kirkpatrick, and S. S. Sapatnekar, "ALIGN: Open-source analog layout automation from the ground up," in *Proceedings of the 56th Annual Design Automation Conference 2019*, pp. 1–4, 2019.

[4] B. Xu, K. Zhu, M. Liu, Y. Lin, S. Li, X. Tang, N. Sun, and D. Z. Pan, "MAGICAL: Toward fully automated analog ic layout leveraging human and machine intelligence," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–8, IEEE, 2019.

[5] R. Dreslinski, D. Wentzloff, M. Fayazi, K. Kwon, D. Blaauw, D. Sylvester, B. Calhoun, M. Coltella, and D. Urquhart, "Fully-autonomous soc synthesis using customizable cell-based synthesizable analog circuits," tech. rep., University of Michigan Ann Arbor United States, 2019.

[6] Y. Park and D. D. Wentzloff, "An all-digital 12pj/pulse 3.1–6.0 ghz ir-uwb transmitter in 65nm cmos," in *2010 IEEE International Conference on Ultra-Wideband*, vol. 1, pp. 1–4, IEEE, 2010.

[7] Y. Park and D. D. Wentzloff, "An all-digital pll synthesized from a digital standard cell library in 65nm cmos," in *2011 IEEE Custom Integrated Circuits Conference (CICC)*, pp. 1–4, IEEE, 2011.

[8] E. Ansari and D. D. Wentzloff, "A 5mw 250ms/s 12-bit synthesized digital to analog converter," in *Proceedings of the IEEE 2014 Custom Integrated Circuits Conference*, pp. 1–4, IEEE, 2014.

[9] S. Bang, A. Wang, B. Giridhar, D. Blaauw, and D. Sylvester, "A fully integrated successive-approximation switched-capacitor dc-dc converter with 31mv output voltage resolution," in *2013 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pp. 370–371, IEEE, 2013.

[10] W. Jung, S. Jeong, S. Oh, D. Sylvester, and D. Blaauw, "A 0.7 pf-to-10nf fully digital capacitance-to-digital converter using iterative delay-chain discharge," in *2015 IEEE International Solid-State Circuits Conference-(ISSCC) Digest of Technical Papers*, pp. 1–3, IEEE, 2015.

[11] M. Shim, S. Jeong, P. Myers, S. Bang, C. Kim, D. Sylvester, D. Blaauw, and W. Jung, "An oscillator collapse-based comparator with application in a 74.1 db sndr, 20ks/s 15b sar adc," in *2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits)*, pp. 1–2, IEEE, 2016.

[12] S. Bang, W. Lim, C. Augustine, A. Malavasi, M. Khellah, J. Tschanz, and V. De, "A fully synthesizable distributed and scalable all-digital ldo in 10nm cmos," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, pp. 380–382, IEEE, 2020.

[13] S. Kundu, L. Chai, K. Chandrashekar, S. Pellerano, and B. Carlton, "A self-calibrated 1.2-to-3.8 ghz 0.0052mm^2 synthesized fractional-n mdll using a 2b time-period comparator in 22nm finfet cmos," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, pp. 276–278, IEEE, 2020.

[14] A. Rovinski, C. Zhao, K. Al-Hawaj, P. Gao, S. Xie, C. Torng, S. Davidson, A. Amarnath, L. Vega, B. Veluri, *et al.*, "A 1.4 ghz 695 giga risc-v inst/s 496-core manycore processor with mesh on-chip network and an all-digital synthesized pll in 16nm cmos," in *2019 Symposium on VLSI Circuits*, pp. C30–C31, IEEE, 2019.

[15] C. Wolf, "Yosys open synthesis suite." http://www.clifford.at/yosys/. Last accessed 2020-05-08.

[16] "Ngspice, the open source spice circuit simulator." http://ngspice.sourceforge.net/. Last accessed 2020-05-08.

[17] S. N. Laboratories, "Xyce parallel electronic simulator (xyce)." https://xyce.sandia.gov/. Last accessed 2020-05-08.

[18] D. M. Moore, T. Xanthopoulos, S. Meninger, and D. D. Wentzloff, "A 0.009 mm 2 wide-tuning range automatically placed-and-routed adpll in 14-nm finfet cmos," *IEEE Solid-State Circuits Letters*, vol. 1, no. 3, pp. 74–77, 2018.

[19] M. H. Perrott, M. D. Trott, and C. G. Sodini, "A modeling approach for Σ-Δ fractional-N frequency synthesizers allowing straightforward noise analysis," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 8, pp. 1028–1038, 2002.

[20] Y. Okuma, K. Ishida, Y. Ryu, X. Zhang, P.-H. Chen, K. Watanabe, M. Takamiya, and T. Sakurai, "0.5-v input digital ldo with 98.7% current efficiency and 2.7-μa quiescent current in 65nm cmos," in *IEEE Custom Integrated Circuits Conference 2010*, pp. 1–4, IEEE, 2010.

[21] S. Weaver, B. Hershberg, P. Kurahashi, D. Knierim, and U.-K. Moon, "Stochastic flash analog-to-digital conversion," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 11, pp. 2825–2833, 2010.

[22] M. Saligane, M. Khayatzadeh, Y. Zhang, S. Jeong, D. Blaauw, and D. Sylvester, "All-digital soc thermal sensor using on-chip high order temperature curvature correction," in *2015 IEEE Custom Integrated Circuits Conference (CICC)*, pp. 1–4, IEEE, 2015.

[23] S. Nalam, M. Bhargava, K. Mai, and B. H. Calhoun, "Virtual prototyper (vipro): An early design space exploration and optimization tool for sram designers," in *Design Automation Conference*, pp. 138–143, IEEE, 2010.

[24] D. Johannsen, "Bristle blocks: A silicon compiler," in *16th Design Automation Conference*, pp. 310–313, IEEE, 1979.