Adaptive Clocking Techniques for SoC Supply Droop Response in Predictive 7nm CMOS

Dan Fritchman, *Member, IEEE* and Wahid Rahman, *Member, IEEE*

***Abstract*—Adaptive clock generation techniques have emerged in recent generations of high-performance SoCs to mitigate timing failure due to supply voltage droops. Resilient techniques vary from analog voltage mixing to digital sensing and clock actuation. This work identifies a taxonomy for adaptive clock generation systems: adaptive clock distribution (ACD) and adaptive PLL-based schemes. Reported realizations in state-of-the-art processor SoCs are reviewed and key performance metrics for comparisons are identified. An evaluation plan is also proposed to compare such systems in a predictive 7nm CMOS technology.**

***Index Terms*—Adaptive clocking, adaptive frequency, power efficiency, supply-droop mitigation, supply-voltage droop.**

I. Introduction

Power management techniques in modern system-on-chips (SoCs) are critical for energy-efficient processors ranging from data servers to mobile devices. SoC thermal dissipation constraints and energy-saving modes necessitate system-level power management to decrease the power supply or reduce the number of active processing cores. Such techniques exhibit a decrease (i.e. droop) in the SoC supply voltage (V_{DD}) due to: the controlled decrease of V_{DD} from a supply regulator or DC-DC converter; and the transient $L\frac{di}{dt}$ supply voltage ripple due to sudden current changes through package inductances when dynamically enabling or disabling on-chip processing cores. To maximize processing throughput, SoCs are designed to operate close to the maximum possible clock frequency (f_{MAX}) for the target (V_{DD}) with minimal guardbanding. Decreasing V_{DD} reduces the drain currents of CMOS transistors, thereby increasing propagation delays in the critical paths of digital logic. If sufficient timing margin is not available in the design, these critical paths can fail to meet timing and cause unrecoverable errors.

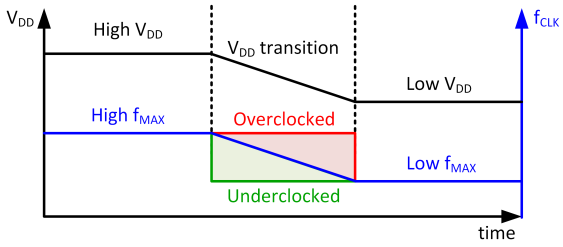

Fig. 1. Reduction of V_{DD} and corresponding reduction in f_{MAX} (adapted from [1]).

To mitigate such failures, adaptive clock generation techniques have emerged in recent generations of high-performance SoCs [1]–[5]. Adaptive-clock systems detect transient supply events and dynamically adjust clock frequencies to not exceed a changing f_{MAX} as V_{DD} decreases. As Fig. 1 illustrates, a decrease from high to low

Date of publication March 25, 2020.

D. Fritchman and W. Rahman are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA 94720 USA (e-mail: dan_fritchman@berkeley.edu; wahid.rahman@berkeley.edu).

This work was supported by Professor Borivoje Nikolić.

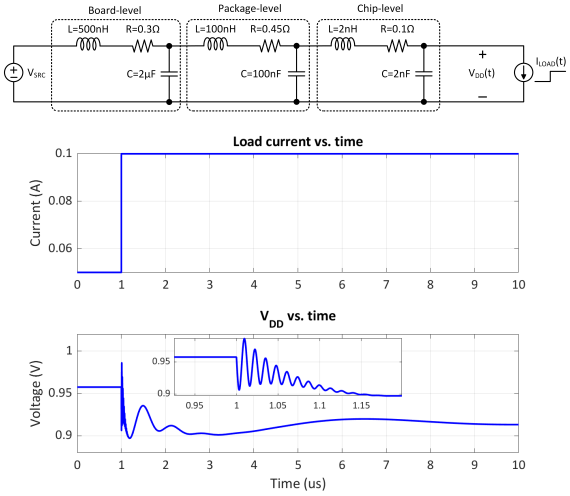

Fig. 2. Supply droop transients due to package inductances and load current (adapted from [2]).

V_{DD} to save power corresponds to a decrease in f_{MAX}, and the SoC clock frequency (f_{CLK}) must track this change without exceeding the correct f_{MAX} (overclocking) and with minimal guardbanding (underclocking) to minimize processing throughput loss during this change [1].

Transient response of adaptive clock techniques is also vital. The external package routes supplying power to the SoC impact supply voltage settling during changes to on-chip core utilization [2], as illustrated in Fig. 2. The range of inductances & capacitances in this network induces a medley of fast and slow time constants. Resilient adaptive clock generation circuits react quickly to fast ripples in V_{DD} ranging from 50-100MHz [2], [3] while also tracking slower transient settling at 1MHz and below [2], [3], [5].

Several schemes exist to realize supply-induced clock adaptation ranging from directly controlling the clock-synthesizing phase-locked loop (PLL) [1], [2] to modulating a tunable delay line or phase rotator downstream in an adaptive clock distribution (ACD) network [3]–[5]. In this work, we survey and contrast the adaptive PLL- and ACD-based mechanisms presented in [2] and [3], respectively. This brief is organized as follows: Section II provides an overview of adaptive clocking schemes and defines the scope of this work; Section III discusses the operating principles of mechanisms presented in [2] and [3]; Section IV provides a comparison of [2] and [3]; and finally Section V discusses planned work and concludes this brief.

II. Adaptive Clocking Schemes

Adaptive clocking systems include two fundamental components:

(a) a *power-supply sensor*, which measures and reports transient droops in supply voltage; and

(b) a *clock period actuator*, which modulates the system clock period in response to reports from the power-supply sensor.

During a supply drop, low-latency sensing and actuation is crucial to detect and adjust the clock frequency quickly to ensure error-free

operation continues throughout the event. If and when V_{DD} recovers from this event to its nominal value, additional circuitry can assist with managing the corresponding f_{CLK} recovery.

ACD- and adaptive PLL-based systems differ in their implementation of the clock period actuator. ACD-based systems decrease the clock frequency by extending clock periods using an ACD circuit placed between the clock-synthesizing PLL and the global clock distribution network as illustrated in Fig. 3. A digital code $CTRL_{FCLK}$ from the power supply sensor controls the ACD circuit such that it extends the period of the synthesized clock, CLK_{PLL}, to the desired frequency given the sensed change in V_{DD}. This lower-frequency clock, CLK_{SOC}, is then distributed through the global clock network to the SoC processing cores. Examples of this ACD circuit include tunable-length delay [5] and phase rotators [3], of which the latter is discussed in Section III-B.

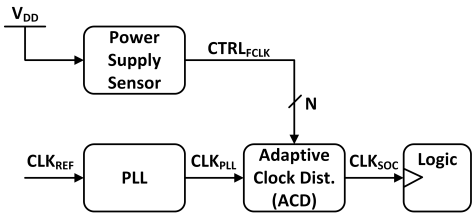

Fig. 3. Adaptive clock distribution (ACD) based system.

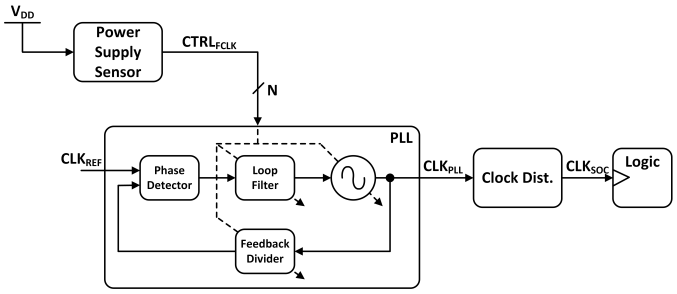

Fig. 4. Adaptive PLL-based system with possible intra-PLL control variants.

Adaptive PLL-based systems, in contrast, incorporate information from the power-supply sensor to dynamically change the behaviour of PLL sub-components. Typically, these schemes affect oscillator control on top of the traditional PLL loop. In [1] for example, the oscillator is controlled indirectly through a loop filter modified for droop response: when a droop is detected, the traditional proportional- and integral-loop filter switches to proportional-only to form a type-I PLL that tunes the oscillator to frequency lock without hazardous over- or underclocking. In [2], this idea is further extended by interrupting the PLL feedback loop to modify the oscillator directly, reacting to a supply droop event with relatively low latency. The feedback divider is then slowly modified to restore PLL lock. This latter work is further discussed in Section III-A.

The focus of this brief is to compare the design and effectiveness of clock period actuators in ACD and PLL-based schemes. While high-resolution power-supply sensors and their low-latency integration are critical to the overall adaptation time of such schemes, it remains beyond the scope of this work.

III. Operating Principles

The two main works for comparison are the PLL- and ACD-based adaptive clocking schemes in [2] and [3] respectively. The operating principles of each are described in this section.

A. PLL-based adaptive clocking

The scheme reported in [2] presents a PLL-based adaptive clocking control for three SPARC processor cores in a 20nm CMOS testchip. A sense point placed near one of the SPARC cores transmits the core V_{DD} through low-impedance on-package (off-chip) routing to the on-chip power-supply sensor. This sensor converts V_{DD} droops into a thermometer-coded quantized digital signal, $q[n]$, by use two calibrated delay lines and a 8-bit time-to-digital converter (TDC).

A major contribution of [2] is to use the instantaneous value of q[n] to immediately reduce the PLL's oscillator frequency to correct for high-frequency droops and use a filtered version of $q[n]$ to correct for low frequency droops and ultimately re-lock the PLL feedback loop. As shown in Fig. 5 and Fig. 6, $q[n]$ is processed by the frequency control logic to generate two control signals: ΔF_{code} that can instantaneously respond to changes in $q[n]$ to directly change the digitally-controlled oscillator (DCO) frequency, and N_{code}^{sync} that relies only on a filtered value of $q[n]$ to track low-frequency droops and re-lock the PLL loop at the new frequency by changing the feedback division value. The minimum function and the non-linear filter together effectively construct an all-pass filter when V_{DD} decreases and a low-pass filter when V_{DD} increases. They allow the loop to actuate the clock with minimal latency during a V_{DD} droop event. During droop recovery, they low-pass filter V_{DD} transients and overshoots as the voltage ultimately settles to its nominal value.

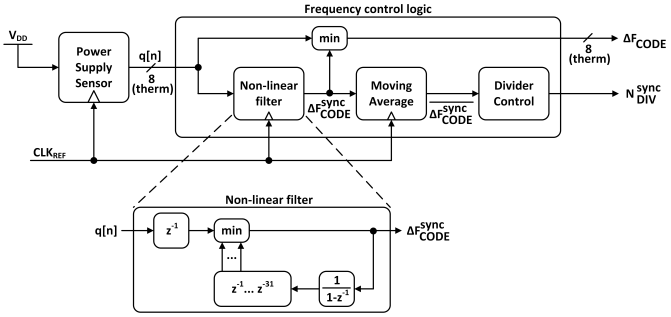

Fig. 5. Frequency control logic for PLL-based adaptive clocking in [2].

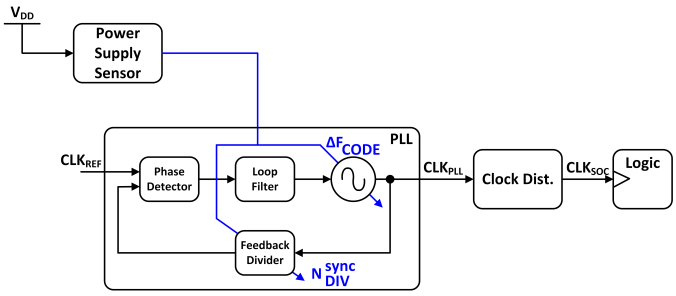

Fig. 6. Intra-PLL control for PLL-based adaptive clocking in [2].

A fast response to high-frequency supply droops is an important criterion in [2], significant design efforts were made to ensure low-latency clock period actuation. The reported design target is 8× faster than the first-droop frequency of 50MHz. This permits a time window of 2.5ns from the time V_{DD} crosses the first supply sensor threshold to the time the SoC clock frequency is corrected. This narrow timing constraint is met by generating and propagating the critical oscillator control signal ΔF_{code} asynchronously. Crucially, the 8-bit thermometer-coded ΔF_{code} controls a DCO that can tolerate asynchronous changes in its code without glitching its output clock.

The DCO reported in [2] is a bank of nine ring inverters with eight of the rings independently enabled by each thermometer bit of ΔF_{code}. Such a structure can tolerate asynchronous arrival of ΔF_{code} when suddenly decreasing frequency, as the case of a droop. Note that frequency increases are restricted: the combination of non-linear filter memory with the masking of minimum function ensures ΔF_{code} only increases one bit at a time after droop recovery [2]. This obviates synchronization of ΔF_{code}, reducing adaptation latency.

The reduction in adaptation latency directly correlates with reported experimental results. Fast response to droops allows for tighter guardbanding and therefore higher f_{MAX}. While [2] does not report the exact clock adaptation latency, the work improves f_{MAX} by 7.5%, from 4.65GHz without adaptive clocking to 5.00GHz with the proposed scheme. This is the largest frequency gain reported in compared works.

B. ACD-based adaptive clocking

The scheme reported in [3] presents a ACD-based adaptive clocking control for AMD's Steamroller CPU in 28nm CMOS. The power supply sensor consists of a DLL-based droop detector acting as a delay line connected to the SoC core V_{DD}: as V_{DD} changes, the DLL phases change with respect to a regulated PLL's output clock. A programmable threshold selects one of four DLL phases to compare against the reference PLL phase to signal a 1-bit droop activity.

Fig. 7 illustrates the reported scheme. A DLL-based phase rotator realizes the ACD circuit. When a droop activity is detected and $DroopDetected$ is asserted, phase rotation between the 40 DLL phases occurs to effectively "stretch" the clock period and reduce the clock frequency. To ensure glitchless phase rotation, small positive pre-programmed phase increments occur at each step, and the phase rotator rotates through all 40 phases before continuing the cycle. Rotation can be bypassed entirely by de-asserting $DroopEnable$.

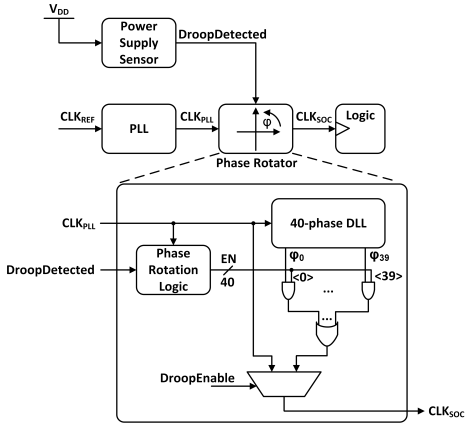

Fig. 7. Frequency control logic for ACD-based adaptive clocking in [3].

This scheme reports to target fast initial droops of 100MHz and slower 1-5MHz secondary droop transients by means of the programmed droop detector threshold [3]. As the detector signal is binary, the ACD-based reduces the SoC clock frequency by a fixed amount (phase rotator steps through a fixed phase step per cycle) or does not change the frequency at all. Voltage recovery is triggered when V_{DD} crosses back beyond the detector threshold. The work reports a total adaptation latency of 3 cycles at 3.4GHz from droop detection to clock frequency reduction. While experimental results do not report frequency gain as in [2], silicon results demonstrated a reduction of 3-6% in V_{MIN}, the minimum supply voltage tolerable at a given frequency for the SoC. No comparison was provided with prior works.

IV. Comparison of ACD and PLL Actuators

A. Comparison Scope

As discussed in Section II, clock adaptation consist of various subcomponents and their latencies. This is illustrated in Fig. 8. Design of a complete adaptation system requires a V_{DD} sense point, routing delay, detector and actuation latencies, and finally clock propagation delay. This comparison focuses on the detection logic, clock actuation, and their combined actuation latency. There are indeed important specifications beyond this when designing adaptation circuits. For example, sensor resolution, bandwidth, and ease of design integration with digital SoCs are important considerations: highly digital sensors [2], [3] to custom analog voltage mixing [6] have their bandwidth and integration tradeoffs. However, the scope of this comparison focuses on PLL- and ACD-based adaptive clocking presented in [2] and [3] respectively, agnostic to extraneous design considerations.

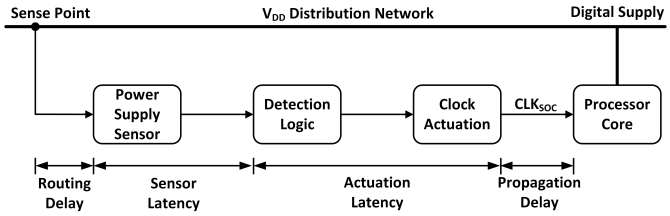

Fig. 8. Various latencies in adaptive clock system (adapted from [2]).

In comparing the PLL- and ACD-based adaptive clocking schemes, key metrics are identified: actuation latency, power/area, calibration, and frequency gain/V_{MIN} reduction. Forecasts for these metrics in the predictive 7nm technology ASAP7 [7] are also discussed.

B. Actuation Latency

Minimizing actuation latency helps reduce overall system response time. In [2], the PLL-based actuation latency varies depending on the direction of V_{DD} change due to the non-linear adaptation filter. As previously discussed, for decreases in V_{DD}, the latency is less than one clock cycle and limited by asynchronous propagation delay through the minimum function in Fig. 4 and the control bandwidth of the oscillator. The control port time constant is approximately $\frac{T_{VCO}}{2N}$ for the reported oscillator [2]. Assuming the routing parasitic from the minimum logic to the oscillator control port is negligible, this allows a latency below one clock cycle at the reported f_{MAX} = 5GHz, i.e. $t_{act} < 200$ps.

Conversely, the ACD-based actuation latency is reported as 2 clock cycles at 3.4 GHz [3], implying $t_{act} = 588$ps. This is due to synchronizing $DroopDetected$ to the PLL clock domain and generating the corresponding phase rotation enable signal as was shown in Fig. 7.

We predict these results will be comparable in ASAP7. The actuation latency in [3] will remain 2 clock cycles by design. That in [2] will remain less than one cycle again by design, with the potential for t_{act} to be reduced due to increased f_T. One drawback may be higher resistance routing which could decrease oscillator control bandwidth if not carefully designed.

C. Power/Area

While neither works report measured power and area overhead of the actuation or overall adaptation circuits, the use of a DLL in [3] is estimated to cost more. Consider phase mismatch and jitter amplification in the DLL [3]. The 40-phase DLL consists of 20

pseudo-differential delay cells, each designed as a pair of cross-coupled inverters with tunable MOS capacitor (MOSCAP) loads. In theory, the k^{th} and $(k+20)^{th}$ phases have a standard deviation of $\sqrt{k}\sigma_u$ where σ_u is one stage's delay variation. The DLL must, at a minimum, ensure phase monotonicity to guarantee glitchless phase rotations. σ_u is directly correlates to V_t and thereby area.

The DLL can also have power implications on the upstream PLL through system jitter specifications. Assume the DLL nears a $2\times$ tuning range, similar of ring-based PLLs. Each DLL stage must achieve a min/max delay spread of $2\times$. Fig. 9 shows schematic simulations in ASAP7 of an example delay stage operating at maximum $f_{CLK} =$ 3GHz. To achieve higher delay spread per stage, higher MOSCAP sizing is needed. However, this also decreases the RC bandwidth at each stage, increasing deterministic jitter amplification such as duty-cycle distortion (DCD) as also shown in Fig. 9. To ensure jitter is not amplified through the DLL, the overall delay chain should target jitter amplification < 1. Otherwise, power must be expended in the upstream PLL to reduce input jitter and DCD.

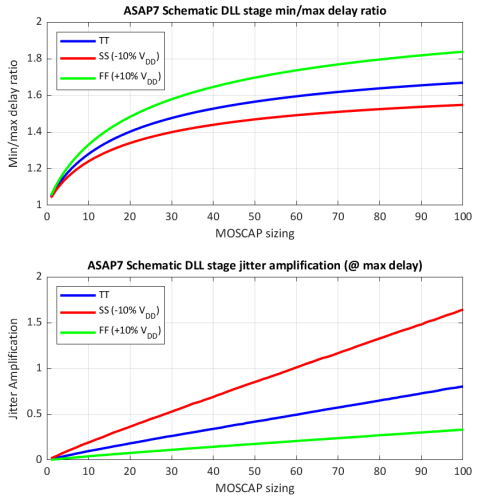

Fig. 9. Schematic simulation of example DLL stage in ASAP7.

D. Design & Calibration

Design-time complexity of the PLL-based scheme in [2] involves an unconventional PLL loop, a detection filter overriding the VCO control and the restoring of PLL lock during frequency recovery by ramping the feedback divider value. Conversely, the ACD-based scheme in [3] is a simpler design that does not require modifications to a PLL, and instead inserts a phase rotator in the clock network between it and the SoC. Both works report manually calibrate supply sensor thresholds to optimize measured f_{MAX} or V_{MIN} margin [2], [3]. The DLL in [3] incurs a further calibration overhead: the DLL can take up to 180 PLL clock cycles to frequency lock after start-up or during a dynamic voltage and frequency scaling (DVFS) event. To mitigate impact of supply noise, the CPU activity is limited during this time. In contrast, the PLL-based approach in [2] does not incur calibration overhead beyond normal PLL locking, the dynamics of which can be particularly fine-tuned in a digital PLL. However, to restore clock frequency after a droop event, the non-linear filter in [2] slowly adjusts the feedback divider (once every 32 reference clock cycles) to mitigate frequency overshoot and current surges during recovery.

E. Silicon Performance

The key performance metric reported in both adaptation schemes is reducing voltage/frequency margin. As [2] claims, this is directly proportional to adaptation latency: the shorter the adaptation response time, the closer the SoC can operate to safety limits. Both systems achieve similar results. In terms of V_{MIN}, the PLL-based scheme achieves 5% reduction at 4.65GHz [2] while the ACD-based scheme reports up to 6% reduction [3] at 4GHz. The PLL-based scheme also reports a 7.5% gain in f_{MAX} to 5GHz if V_{DD} is unchanged [2]. These results require integration of the adaptation system with the complete entire SoC, which may not be feasible in ASAP7 for this project scope: adaptation latency will be used as a proxy.

F. Proposed Project Objectives

While reported in planar technologies [2], [3], the reduced actuation latency and the lack of immediate power, area, and calibration overhead for similar reported reduction in V_{MIN} suggest the PLL-based adaptation circuit [2] will be the favourable candidate in FinFET technologies. We propose to evaluate this in ASAP7 with the following project objectives:

(a) Initial design and power/area/latency estimates of the ACD-based actuator in [3], in particular the DLL-based phase rotator. Estimated mismatch models may be required if not immediately available in ASAP7 [7].

(b) Initial design and power/area/latency estimates of the PLL-based actuator in [2], in particular the PLL loop and non-linear filter interaction. Estimated models of the oscillator may be required.

(c) (*Time-permitting*) Investigations into possible improvements in [2], such as derivative loop filter control or modifications to non-linear filter to reduce frequency recovery time.

V. Conclusion

This work compares the costs and effectiveness of ACD- and PLL-based adaptive clocking systems [2], [3]. ACD-based systems exchange design complexity for higher power and area, but both systems achieve less than 2 cycle actuation latency and V_{MIN} reductions by up to 6%. The PLL-based system is predicted to achieve lower actuation latency and power/area in predictive 7nm CMOS, and initial designs of these systems are planned to evaluate this claim.

References

[1] F. Ahmad *et al.*, "A 0.5-9.5-GHz, 1.2-μs Lock-Time Fractional-N DPLL with $\pm$1.25%UI Period Jitter in 16-nm CMOS for Dynamic Frequency and Core-Count Scaling," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 21–32, Jan. 2017.

[2] T. Hashimoto *et al.*, "An Adaptive-Clocking-Control Circuit with 7.5% Frequency Gain for SPARC Processors," *IEEE J. Solid-State Circuits*, vol. 53, no. 4, pp. 1028–1037, Aug. 2018.

[3] K. Wilcox *et al.*, "Steamroller Module and Adaptive Clocking System in 28 nm CMOS," *IEEE J. Solid-State Circuits*, vol. 50, no. 1, pp. 24–34, Jan. 2015.

[4] M. S. Floyd *et al.*, "Adaptive clocking in the POWER9™ processor for voltage droop protection," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, San Francisco, USA, Feb. 2017, pp. 444–445.

[5] K. A. Bowman *et al.*, "A 16nm All-Digital Auto-Calibrating Adaptive Clock Distribution for Supply Voltage Droop Tolerance Across a Wide Operating Range," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 8–17, Jan. 2016.

[6] N. Kurd *et al.*, "Next Generation Intel Core™ Micro-Architecture (Nehalem) Clocking," *IEEE J. Solid-State Circuits*, vol. 44, no. 4, pp. 1121–1129, Mar. 2009.

[7] V. Vashishtha *et al.*, "ASAP7 predictive design kit development and cell design technology co-optimization," in *IEEE/ACM Int. Conf. on CAD Dig. Tech. Papers*, Irvine, USA, Nov. 2017, pp. 992–998.