Near-Threshold Computing Revisited

Chris Gonzales
Mario Lok
Judson Porter

***Abstract*—Near-threshold computing (NTC) has been suggested as a way to continue Moore's law scaling by offering a 10x improvement in energy efficiency [1]. With this energy improvement however comes a range of problems including a 10x increase in delay (reduction in performance), 5x increase in global process variation, and 5 orders of magnitude increase in failure rate. In this paper, we reexamine several techniques proposed for mitigating these issues, including device optimizations, soft-edge clocking, body biasing, and cluster-based architectures, and address why they may be less effective than stated at dealing with the issues of NTC. We also examine the additional issue of DC-DC converter efficiency in NTC, and further show how the lack of application-level parallelism will provide limited opportunities for regaining performance.**

I. Introduction

Power consumption and heat removal have become two of the top concerns in the datacenter space [2]. According to the International Technology Roadmap for Semiconductors, power consumption for each generation of chips is growing, making it one of the most significant roadblocks to future scaling [3].

One recently proposed solution to this increased power is to operate the microprocessor at a significantly lowered voltage, approaching the threshold voltage of its transistors [4]. This near-threshold computing (NTC) will yield approximately a 10x energy savings at the cost of 10x performance loss and a 20x total performance uncertainty. The NTC proposal claimed that many of these downsides can be mitigated by techniques such as device optimizations, variation tolerant circuit design, and body biasing as well as architectural changes such as increased parallelism and a move to a clustering-based architecture.

Although at first glance these improvements may seem to be able to regain much of the performance loss associated with near-threshold computing, the techniques do not hold up to closer scrutiny. In section 1, we will introduce an expression for the dependency of delay on operating voltage in near-threshold. Section III will expand upon this to show that the potential device optimizations cited by Dreslinski's NTC paper as methods to increase transistor speed will be ineffective in the near-threshold regime. Moving on to concerns over variability, section IV derives a conservative model for how the number of potential critical paths increases due to the additional delay variation inherent in near-threshold computing and section refsec:softedge raises concerns about the ability of soft-edge flip flops to overcome delay variations in near-threshold designs. **ADD MARIO'S PART** Concerns about the methodology of Dreslinski's proposed clustering architecture are addressed in section VIII, which is generalized in section IX to show that any parallel architecture will have problems regaining performance lost to NTC.

We have shown that many of the ideas presented in Dreslinski's NTC paper [4] are ineffective at recovering the performance lost due to near-threshold computing, especially in the context of a datacenter environment. Although datacenters have a strong motivation to reduce their power consumption, the performance sacrifices inherent to NTC makes it an impractical choice for future servers.

II. Determining the effect of ΔV_{th}

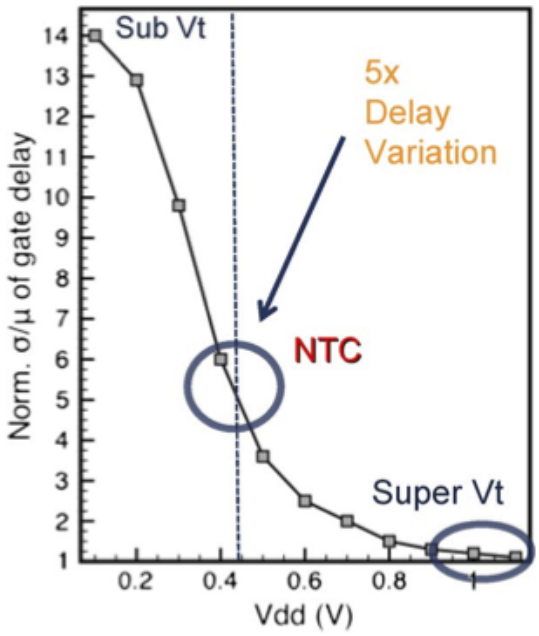

Fig. 1. Dependence of delay on voltage in near-threshold. [4]

Because the delay of a gate has a strong dependence on gate voltage in near-threshold operation (see Figure 1), determining the effect of small changes in ΔV_{th} is essential to understanding the operation of near-threshold circuits. This delay is determined by the drain current of the transistor. The drain current has fundamentally different relationships to voltage in sub- and super-threshold operation; in the sub-threshold mode, the current has an exponential relationship to voltage [5]

$$I_D \propto e^{\frac{V_P - V_S}{U_T}} \tag{1}$$

while super-threshold operation yields a quadratic relationship

$$I_D \propto (V_P - V_S)^2 \tag{2}$$

where $V_P = V_G - V_{TH} - \gamma$ with γ being a body effect factor. For near-threshold operation, however, an interpolation has to be used to bridge these two expressions [5], giving us the forward drain current as

$$I_D \propto \left[\ln\left(1 + e^{\frac{V_G - V_{TH} - \gamma}{2}}\right)\right]^2 \tag{3}$$

The relationship between delay and current, as derived in [6], is

$$t_p = \frac{k_d \cdot C_L \cdot V_D}{I_{on}} \tag{4}$$

When equation 3 is plugged into this equation, this delay becomes

$$t_p \propto \frac{V_D}{\left[\ln\left(1 + e^{\frac{V_G - V_{TH} - \gamma}{2}}\right)\right]^2} \tag{5}$$

showing that small variations in V_G or γ can cause large differences in transistor delay.

III. Device Optimizations

Dreslinski et al. [4] suggest modifications of the transistor structure to reduce delay by reducing inverse sub-threshold slope (S_S). This can take the form of either modifying the channel doping profile [7] or increasing oxide length [6].

Hanson shows that the main delay benefit from scaling S_S comes from the assumption that the system voltage set to the minimum energy point of the CMOS logic, a point far into sub-threshold that is proportional to S_S [6].

$$t_p = \frac{k_d \cdot C_L \cdot K_{V_{min}} \cdot S_S}{I_{off} \cdot 10^{\frac{K_{V_{min}} S_S}{S_S}}} \tag{6}$$

$$\propto \frac{C_L \cdot S_S}{I_{off}} \tag{7}$$

For near-threshold operation, the voltage is instead set by the transistor threshold voltage. With the voltage dependence on S_S removed, this equation instead becomes

$$t_p \propto \frac{1}{e^{\frac{V_D - V_{TH}}{S_S}}} \tag{8}$$

which is a much weaker effect considering that S_S in the range of 80-90mV/decade.

The actual correlation of delay to S_S is even weaker because Hanson is using an equation that assumes the circuit is operating far sub-threshold. As V_{DD} passes V_{TH}, the current and delay begin to instead scale with an interpolation between super-threshold and sub-threshold models rather than just the sub-threshold model. Because the super-threshold model doesn't have a reliance on S_S, this means that the effect of modifications to S_S on system delay get weaker as voltage moves from sub-threshold into the near-threshold regime.

Raychowdhury [8] shows this weakening of S_S's effect on delay in Figure 2. As $V_D D$ increases, the delays of the standard and optimized CMOS devices get closer.

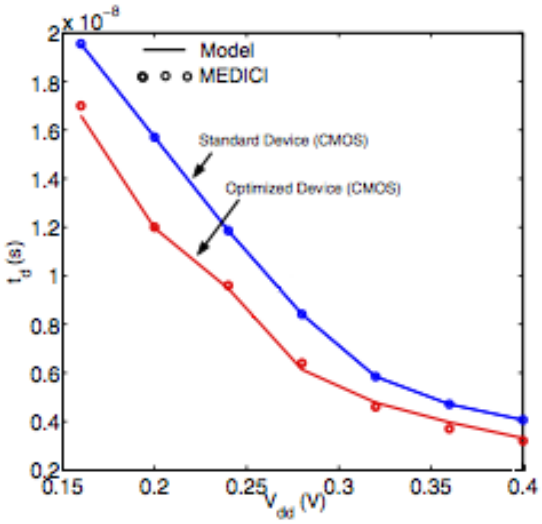

Fig. 2. Effect of sub threshold slope optimization on delay at different voltages. [8]

Although Dreslinski is correct that several research groups have shown optimizations that allow sub-threshold devices to operate at a reduced delay [4], citing these as evidence that near-threshold delay can also be improved is misleading. Near-threshold devices, by their very definition, operate in a different region of the transistor's current-voltage curve and optimizations for sub-threshold operation will have a greatly diminished effect on near-threshold delays.

IV. Increase in the number of critical paths

The increased variance in delay caused by near-threshold operation is directly responsible for an increase in the number of critical paths. A critical path can be defined as any path that has a high probability of exceeding a given clock [9]. For our case of trying to find the maximum frequency a given device can run, we can instead consider a critical path as a path that has a high probability of setting FMAX; that is, of being the slowest path in the system.

Consider a distribution the of nominal delays of paths within a chip (Figure 3). In the case of no variations, clock speed is set by the path with the highest nominal delay. Once delay variation is added in, however, points that are close in delay to the path with the highest nominal delay could act instead as the critical path in a portion of chips. This means that it is any path that falls within $n\frac{\sigma_v}{\mu}$ of the maximum nominal delay for a given n that is set by the design where σ_v is the standard deviation and μ is the mean of the timing path variation distribution. The number of critical paths then becomes the area under this delay-count curve where the delay is greater than $n\frac{\sigma_v}{\mu}$

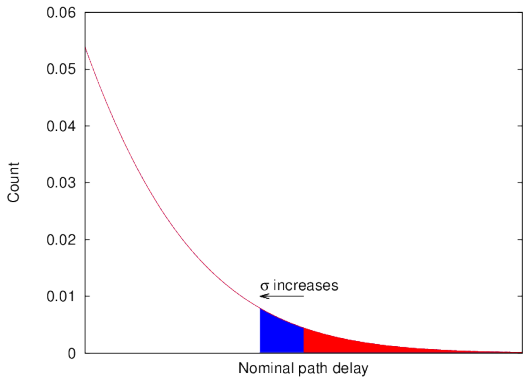

Fig. 3. As σ_v increases, the number of critical paths (shaded area) increases

Although the shape of the nominal delay distribution will vary depending on the design of the chip, if it is assumed this shape is strictly convex, it can be shown that the number of critical paths N_{cp} increases at least linearly with σ_v. Considering that $\frac{\sigma_v}{\mu}$ increases approximately 7x as the voltage drops into the near-threshold regime, there is at least a 7x increase in the number of critical paths. This will translate into a reduction in the maximum frequency of the design [10] and added design complexity, as more critical paths will have to be optimized.

V. Soft-edge clocking

Another proposal has been the use of soft-edge clocking, such as the soft-edge flip-flop (SFF) to increase speed in near-threshold devices by reducing the maximum clock frequency's dependence on critical path delay [11]. In this design, a traditional D-flip flop (DFF) is modified to be driven by two offset clocks, generating a short transparency window. At the limit where the offset between the clocks is 0, the SFF acts identically to a standard DFF. As the offset between the two clocks increases, the SFF's transparency window increases, allowing more time borrowing and allowing the chip to run faster, as shown in Figure 4. If the offset becomes too large, functional failures can occur in short paths as signals race through the transparent flip-flops.

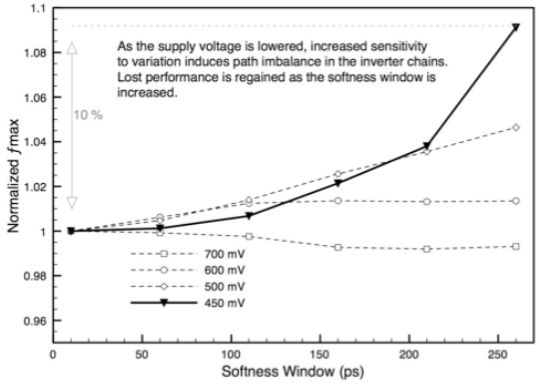

Fig. 4. Frequency improvement as SFF transparency increases. [11]

Generating the offset between the two clocks becomes an issue as variability increases. The simple method used in the paper of having a chain of inverters would be heavily susceptible to process variations. As σ_v/μ increases, the variance of the delay through this chain of inverters increases. Because this delay sets the offset between the two clocks, the increased variance results in a wider range of performance for the SFFs in the design.

VI. Voltage Regulators

We already know logic gates computes more efficiently at a supply voltage level that is near the device threshold voltage. When we attempt to translate the energy efficiency of these gates to the energy efficiency of a whole system, we need to carefully evaluate the associated overhead. Both [12] and [13] raised the concern that the efficiency of the voltage regulator would be lower for delivering power to a near-threshold computing design. [12] presents evidence that typical voltage regulator efficiency is lower at light load and argues that if a near threshold design consumes less power the efficiency will be lower. [13] considers the case where the chip supply voltage is generated with an on-chip switch-capacitor converter. They shows that the efficiency of the regulator is at 90% for an output voltage of 1.2V and is at 70% for an output voltage of 0.5V.

In the context of server computing, the power is delivered to the microprocessors typically through a buck converter which down converts the 12V mother board voltage [14]. Although it is difficult to derive a quantitative comparison between the efficiency of a 1.2V output for an above threshold design and a 0.5 output for a near threshold design, it is possible to consider the difference in requirement for these two regulators and understand where the difference in efficiency may come in.

The performance of digital circuits is highly susceptible to power supply noise. This susceptible is worse when the design is operating in the near threshold region, as shown earlier in figure 1). On the other hand, the power supply noise is a function of load stepping and decoupling capacitor. Since load stepping is a function of the processor architecture, we can assume simply scale down the supply voltage would not change the magnitude of the power supply noise. This power supply noise is a larger percentage of the supply voltage when the design is operating in near threshold region. As a result, it cause more delay variation in the design. As performance variation has already been one of the key limiting factors for near threshold computing, it become necessary to suppress this voltage noise at the price of efficiency, board area or chip area by increasing the feedback loop bandwidth and adding more decoupling capacitors.

In addition, one of the goals of employing near threshold computing for servers is to reduce power consumption while maintaining throughput. However, unless near threshold computing can achieve processor power reduction by a factor that is larger than the factor supply voltage is reduced by, the current drawn by a near threshold design is going to be

higher. This higher output current is particular problematic for the power loss due to the output rectification. The efficiency loss in the output rectification is the ratio between the output voltage and the voltage drop across the output transistor. Since the output voltage is small, the regulator is more sensitive to voltage drop across the output transistor. On top of that, the higher output current further increases the voltage drop across the output stage.

VII. Body Biasing

Bodying biasing has been proven to be effective in compensating for the performance loss of a design due to within-die systematic variation and intra-die variation. However, it has been increasingly challenging to employ this technique because the effect of body biasing is decreasing as transistor technology scales to smaller dimension. In every technology generation, the gate oxide capacitance C_{ox} is scaled up for a factor 0.7 and the doping concentration of the channel is scaled up by a factor of 0.7. By equation X1, the combined effect of scaling these two parameters reduces the body effect parameter gamma, which determines the effectiveness in using body biasing to modify the device threshold voltage as shown in equation X2.

$$V_T = V_{TO} + \gamma(\sqrt{|V_{SB} + 2\phi_F|} - \sqrt{|2\phi_F|} \tag{9}$$

$$\gamma = (1/C_{ox})\sqrt{2q\epsilon_{si}N_A} \tag{10}$$

On the other hand, systematic variation such as gate length is increasing with every successive technology. [] shows that sigma of the critical dimension variation caused by line edge roughness has been around 5nm and remained roughly constant across different technology node. As a result, the variability in delay continues to rise with technology scaling.

Under these two trends, the need for body biasing compensation is increasing while the effectiveness of body biasing is decreasing. Nevertheless, when the body biasing technique is applied to a near threshold design, the delay tuning range is much larger for the same amount of threshold modulation. Yet at the same time, since a near-threshold design is more sensitive to process variation, it is not obvious whether body biasing is more effective for a near threshold design.

To make a quantitative comparison between the effectiveness of body biasing for both an above threshold design and a near threshold design across different technology nodes, we simulate the propagation delay of a 6 FO4 inverter delay chain in 32nm, 22nm and 16nm using ASU Technology Predictive Model []. To simplify the analysis, the only variation considered is within-die systematic and die-to-die gate length variation. Table X and table y shows the delay of the inverter chain without applying body biasing and with maximum forward body biasing when no variation is presented. The results show that body biasing indeed influence the delay of the inverter chain by a large percentage in the near threshold region, but across technology node, the performance improvement with maximum body biasing is decreasing.

	32nm	22nm	16nm
nominal delay(ps)	45.0	26.0	17.2
maxiumum V_{BBfwd} (ps)	35.5	21.1	14.2
percentage change	21%	18.9%	17.1%

	32nm	22nm	16nm
nominal delay(ps)	1709	464	207
maxiumum V_{BBfwd} (ps)	541	195	105
percentage change	69%	58%	49.3

Table Z shows the result When we include variation in the model. The effectiveness of process compensation by body biasing is measured in terms of magnitude of gate length variation body biasing can restore. As suggested by the model, although the compensation is very effective in 35nm for near threshold computing, being able to compensate for 53% of gate length variation, this number will quickly be quickly be reduced to 11.8% in two technology node. At the same time, as argued earlier the gate length variation is increasing. In future technology generation, the body biasing technique would not be sufficient to offset the delay variation for near threshold computing.

	32nm	22nm	16nm
above Vt(nm)	3.8	1.5	0.8
% compnesation in L	10.8%	6.8%	5%
near Vt(nm)	17	4.3	1.9
% compnesation in L	53%	19.5%	11.8%

VIII. Near-Threshold Parallel Architectures

Dreslinski et al. [1] claim "In applications where there is an abundance of thread-level parallelism the intention is to use 10 s to 100 s of NTC processor cores that will regain 10-50X of the performance, while remaining energy efficient." In order to regain the performance lost from using near-threshold techniques, Zhai et al. [15] and Dreslinski et al. [16] present a technique for leveraging parallelism in the NTC regime. The proposed architecture groups multiple slower-cores into clusters which share a faster L1 cache. The cache operates at n times higher frequency than the cores, where n is the number of cores in a cluster. This is motivated by the observation that SRAM has a higher energy optimal V_{dd} and V_{th} than logic due to its lower activity factor and higher relative leakage. As a result of SRAMs higher energy optimal V_{dd}, the energy optimal frequency of memory is higher than that of logic. Based on this observation, the proposed technique shares the first-level cache with multiple, slower cores allowing individual tuning of V_{dd} and V_{th} between the cores and memory. Using this architecture, the cores still maintain single-cycle memory accesses while the core and memory can each operate at their energy optimal V_{dd} and V_{th}. Running the memory at a higher V_{dd} also helps mitigate many of the reliability issues affecting SRAM in the near-threshold regime.

Using this technique, a 71% energy savings over a baseline single core machine on the highly parallel SPLASH2 benchmark is demonstrated. However, investigating these claims reveals some shortcomings with this approach. Of primary

concern is the large area overhead required to achieve the same benchmark performance. In order to achieve the same performance as a single core baseline system on the *Cholesky* benchmark, 6 cores and 3 times the baseline amount of cache were required. In the worst case, some benchmarks require as many as 16 cores and 32 times as much L1 cache (2 MB vs 64 kB) to achieve the same performance as the baseline. It is important to remember that these results are being presented in comparison to a single core reference on a highly parallel workload. By Amdahl's law, the speedup from parallelism will have diminishing returns as more cores are added. The corollary is that the biggest benefit from parallelism will be realized by going from a single core to several cores. Emphasizing this point, their results show only a an approximate 15% energy efficiency improvement when moving from a traditional V_{dd} scaled CMP to a clustered architecture without V_{th} tuning.

The clustering technique also uses separate V_{th} tuning for the core and cache to find the energy optimal voltage for the same performance. However, in examining these voltages actually are, it is revealed that neither V_{dd} of the cache nor the logic is actually in the near-threshold regime as defined by the paper, and both are in fact operating at a V_{dd} 2x–3x higher than the selected V_{th}. In modern process technologies, the standard V_{dd} is already approaching 2x V_{th}. As described in Section VII, the techniques for tuning V_{th} are also becoming less effective.

One final concern is that while energy optimal cluster configurations are presented for each SPLASH2 benchmark, no co-optimized configuration is given. This could potentially be an issue as the energy optimal range of cores, clusters, cache sizes, V_{dd} and V_{th} is large. It is unclear what the energy savings across benchmarks for different configurations will be.

IX. Shortcomings of Application Parallelism

Dreslinski et al. [1] state "More gates can now fit on a die, but a growing fraction cannot actually be used due to strict power limits." This issue is commonly referred to as dark silicon. While the number of transistors on a given die has been doubling every generation, the number of transistors that can be powered for a fixed power budget has not been increasing due to the slowing of transistor energy scaling. Since power budgets have not been increasing past the limits defined by air cooling, a situation has arisen where future generations of chips will potentially have more transistors than can be powered at any given time. Several recent works [17], [18] have looked at the impact of dark silicon on computing in the near future.

Dark silicon seemingly presents an opportunity for near-threshold computing, and as a way of addressing the large area overheads of the clustering architectures discussed in Section VIII. By reducing the power consumed per-core, more cores can be simultaneously powered. Since these cores could not be powered in a super-threshold chip, this represents an opportunity for near-threshold chips to regain performance compared to super-threshold operation.

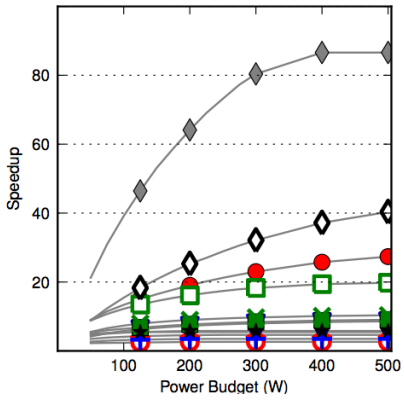

Fig. 5. Projected speedups on PARSEC benchmarks while the chip power limit (and therefore the number of cores) is increased. [17]

While dark silicon is an opportunity for devices to reach higher levels of integration, recent studies analyzing dark silicon have not born out the promise of higher levels of performance with increasing core counts. Work by Esmaeilzadeh et. al [17] developed an analytical model analyzing the impact of dark silicon for a range of CMP configurations in future process technologies. They show projected speedups for these configurations using the PARSEC benchmark suite, which represents workloads similar to the SPLASH2 benchmark suite used in the previously discussed clustering architecture papers [1], [15], indicating that both works are targeting similar applications. This analytical model does project that dark silicon will become a significant portion of CPU area in the near future, dominating as soon as 2016 with a conservative scaling model. However, the authors also analyze the case where the power constraint is lifted, which allows more cores to be powered and reduces the amount of dark silicon. Figure 5 shows the projected speedups on different PARSEC benchmarks as the chip power is increased. The paper projects that if the amount of parallelism in applications were increased to 99%, then the best case speedup for power limited cores in 8 nm is 15x relative to a quad-core Nehalem processor at 45 nm. However, as shown in Figure 5, with an unconstrained power budget, and the same current amount of application level parallelism, 8 out of 12 PARSEC benchmarks achieve no more than 10x speedup. What this indicates is that the amount of parallelism in general-purpose parallel workloads is currently not exploitable even by future power constrained multi-core chips. While near-threshold computing would allow more chips to be powered, it would not realize a significant speedup on most general-purpose parallel workloads due to the lack of exploitable parallelism.

Further work by Hardavellas et al. [18] confirms the observation that future speedups are limited by application level parallelism, not power constraints. In this study the authors built an analytical chip performance model for different types of cores, including low-power embedded cores (EMBs) similar to an ARM11 MPCore, and modeled how the performance of these chips scale in future process technologies given

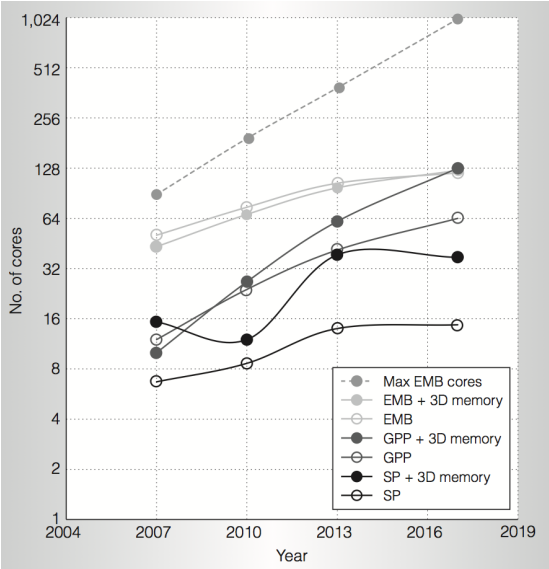

Fig. 6. Projections for core count scaling. [18]

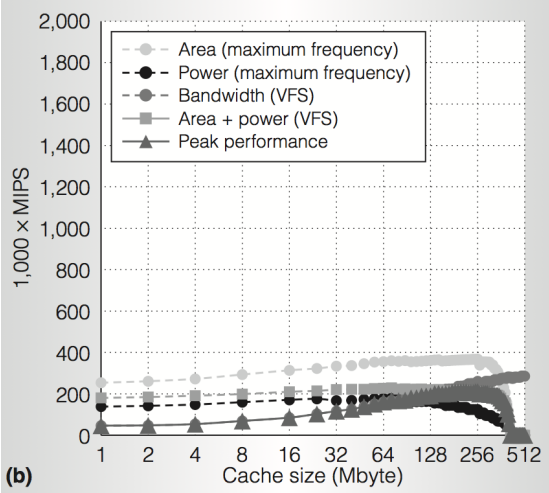

Fig. 7. Embedded core scaling. [18]

area, power, and memory bandwidth constraints. Figure 6 shows projected core count scaling trends in future process technologies. The difference between the "Max EMB cores" line and the "EMB" line represents the difference in how many cores can fit on a die versus how many can be powered given current power constraints. Their model projects that in 2017 over 1024 cores will be able to fit on a die, but only 12% of them will be able to be powered at any given time. Again, this looks like an opportunity for near-threshold computing to increase core counts over traditional super-threshold CMPs.

Figure 7 shows the maximum performance of an EMB-based CMP given different constraints for a 99% parallel workload. The "Area (maximum frequency)" line represents the best performance with only a die area constraint, while the "Peak Performance" line represents the best performance factoring in area, power, and memory bandwidth constraints. While the constrained case only has 12% of the cores of the area constrained case, it only has approximately half the performance of the unconstrained case. Again, the performance is limited by the application parallelism, not the number of cores integrated onto the chip.

While NTC processors could support increased core counts over super-threshold chips, they would not be able to gain back a significant amount of the performance loss as super-threshold core counts are approaching the limits of exploitable parallelism in general-purpose workloads. Without a focus on increasing the amount of parallelism in these applications, opportunities for NTC processors to regain performance loss through parallelism will be limited.

X. Conclusion

In the general-purpose server computing space, near-threshold computing is unfortunately not a viable solution to achieving more energy efficiency computing while maintaining system throughput. The inherent problems of slower device, larger variability and higher system failure rate in the near-threshold computing approach are still major barriers for a near-threshold server processor to achieve the same throughput level as an above-threshold server processor. The state-of-the-art device optimization for low voltage operation provides little performance boost to transistors operating in the near-threshold region. Soft-edge clocking, an effective technique in combating variation for an above threshold design, incurs a much larger overhead when it is scaled to near-threshold. At the same time, body-biasing which has been a reliable method for variation compensation, is fading away as technology scales. Adding to the difficulties of applying near-threshold computing, the inefficiency of delivering power to a low voltage chip and the stricter requirement on the power supply diminishes the power reduction benefits from operating the server processor at low voltage. Worst of all, the marginal parallelism that can be extracted from the already highly parallel server workload is far from enough to bridge the 10X performance gap between near-threshold computing and traditional above threshold computing.

References

[1] R. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge, "Near-Threshold Computing: Reclaiming Moore's Law Through Energy Efficient Integrated Circuits," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 253–266, 2010.

[2] "Report to congress on server and data center energy efficiency: Public law 109- 431," 2007. [Online]. Available: http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_R

[3] "International technology roadmap for semiconductors," 2009. [Online]. Available: http://www.itrs.net/

[4] R. Dreslinski, M. Wieckowski, D. Blaauw, D. Sylvester, and T. Mudge, "Near-Threshold Computing: Reclaiming Moore's Law Through Energy Efficient Integrated Circuits," in *Proceedings of the IEEE*, 2010, pp. 253–266.

[5] C. Enz and F. Krummenacher, "An analytical MOS transistor model valid in all regions of operation and dedicated to low-voltage and low-current applications," *Analog integrated circuits and ...*, 1995.

[6] S. Hanson, M. Seok, D. Sylvester, and D. Blaauw, "Nanometer Device Scaling in Subthreshold Circuits," in *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE*, 2007, pp. 700–705.

[7] B. Paul, A. Raychowdhury, and K. Roy, "Device Optimization for Ultra-Low Power Digital Sub-Threshold Operation," in *Low Power Electronics and Design, 2004. ISLPED '04. Proceedings of the 2004 International Symposium on*, 2004, pp. 96–101.

[8] Raychowdhury, "Ultralow Power Computing with Sub-threshold Leakage: A Comparative Study of Bulk and SOI Technologies," in *Design, Automation and Test in Europe, 2006. DATE '06. Proceedings*, 2006, pp. 1–6.

[9] L. C. Wang, J. J. Liou, and K. T. Cheng, "Critical Path Selection for Delay Fault Testing Based Upon a Statistical Timing Model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 11, pp. 1550–1565, Nov. 2004.

[10] K. Bowman, S. Duvall, and J. Meindl, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 2, pp. 183–190, 2002.

[11] M. Wieckowski, Y. M. Park, C. Tokunaga, D. W. Kim, Z. Foo, D. Sylvester, and D. Blaauw, "Timing yield enhancement through soft edge flip-flop based design," in *Custom Integrated Circuits Conference, 2008. CICC 2008. IEEE*, 2008, pp. 543–546.

[12] R. Abdallah, P. Shenoy, N. Shanbhag, and P. Krein, "System energy minimization via joint optimization of the dc-dc converter and the core," in *Low Power Electronics and Design (ISLPED) 2011 International Symposium on*, aug. 2011, pp. 97 –102.

[13] Y. Pu, X. Zhang, J. Huang, A. Muramatsu, M. Nomura, K. Hirairi, H. Takata, T. Sakurabayashi, S. Miyano, M. Takamiya, and T. Sakurai, "Misleading energy and performance claims in sub/near threshold digital systems," in *Computer-Aided Design (ICCAD), 2010 IEEE/ACM International Conference on*, nov. 2010, pp. 625 –631.

[14] Y. Ma, X. Xie, and Z. Qian, "An active-clamped buck converter for 12v vrm application," in *Power Electronics Specialists Conference, 2006. PESC '06. 37th IEEE*, june 2006, pp. 1 – 5.

[15] B. Zhai, R. Dreslinski, D. Blaauw, T. Mudge, and D. Sylvester, "Energy efficient near-threshold chip multi-processing," *Low Power Electronics and Design (ISLPED), 2007 ACM/IEEE International Symposium on*, pp. 32–37, 2007.

[16] R. Dreslinski, B. Zhai, T. Mudge, D. Blaauw, and D. Sylvester, "An Energy Efficient Parallel Architecture Using Near Threshold Operation," in *Parallel Architecture and Compilation Techniques, 2007. PACT 2007. 16th International Conference on*, 2007, pp. 175–188.

[17] H. Esmaeilzadeh, E. Blem, R. St Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *ISCA*. New York, NY, USA: ACM, 2011, pp. 365–376.

[18] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki, "Toward Dark Silicon in Servers," *IEEE Micro*, vol. 31, no. 4, pp. 6–15, 2011.