A 90nm Data Flow Processor Demonstrating Fine Grained DVS for Energy Efficient Operation from 0.25V to 1.2V

Y. Shakhsheer, S. Khanna, K. Craig, S. Arrabi, J. Lach, and B. H. Calhoun
Dept. of Electrical and Computer Engineering,
University of Virginia,
Charlottesville, VA, USA, E-mail: yousefshak@virginia.edu

Abstract – We present a 90nm data flow processor that executes DSP algorithms using fine grained DVS at the component level with rapid V_{DD} switching and V_{DD} dithering for near-ideal quadratic dynamic energy scaling from 0.25V-1.2V. Measurements show energy savings up to 50% and 46% compared to single-V_{DD} and multi-V_{DD} alternatives.

I. Introduction and Motivation

Energy efficiency is the most critical metric in modern integrated circuits, with applications converging towards increased portability. The motivations to lower energy range from enabling battery-less operation in energy harvesting body sensor networks, to maximizing battery life in mobile platforms, to managing thermals in processor cores. Many systems occasionally require high performance, but their varying workload requirements remain below this upper limit for the majority of their lifetimes. Designing the system in a static fashion to support this peak performance can substantially increase the total system power. Dynamic voltage scaling (DVS) provides the ability to trade-off the energy and delay of such variable workload systems.

A method called Panoptic ("all-inclusive") Dynamic Voltage Scaling (PDVS) extends DVS to finer granularity in space and time, allowing for much more flexible and energy efficient designs [1]. PDVS uses multiple PMOS header switches at the component level (Fig.1) to select the local block V_{DD} from among a discrete set of shared V_{DD} rails depending on the workload requirement and timing slack available for a given program, thus allowing for fast switching times and independent block voltages. Recent DVS processors, shown in Table I, limit the spatial granularity of varying V_{DD} to the core level or above and rely on external DC-DC converters to adjust V_{DD}, which takes tens to hundreds of µsecs [2].

This paper presents the first processor using PDVS – a synchronous data flow processor in 90nm bulk CMOS (Fig. 12). This processor is to our knowledge the first to demonstrate single clock cycle V_{DD}-switching at the component level, integrated V_{DD} dithering [6] for near optimal energy scalability, and the capability to switch efficiently between high performance DVS and subthreshold (sub-V_T) modes.

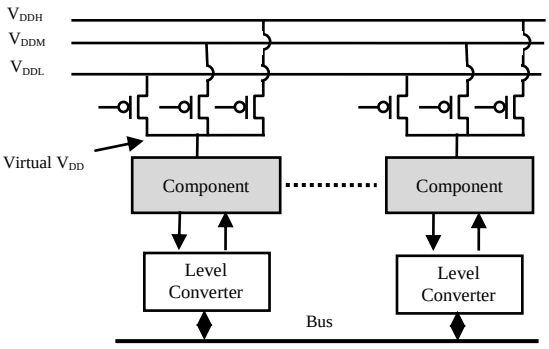

Fig.1. Illustration of PDVS architecture, which enables local fine-grained DVS using header switches and a small set of shared V_{DD}s.

TABLE I.
DVS State of the Art Implementation Comparisons

Feature	*[3]*	*[4]*	*[5]*	*This work*
V_{DD} Granularity	6 cores	1 core	1 core	Add, Mult
Speed of V_{DD} change	>10µs (e.g.[2])	2-5ns	>10µs (e.g. [2])	1ns
V_{DD} dithering	No	No	No	Yes
Sub-V_T operation	No	No	No	Yes

The rest of the paper is organized as follows: Section II introduces the data flow processor's architecture. Section III discusses the area and energy overheads of the PDVS scheme on the chip. Section IV details the data path modifications to enable switching efficiently between an ultra low power subthreshold mode and high performance modes that use higher V_{DD}s. Section V discusses the design and the features of the data and instruction memories. Section VI describes the measurement results from the 90nm test-chip, and Section VII concludes our paper.

II. PDVS Architecture

A. Data flow processor architecture

Fig. 2 shows a block diagram of the 32b data flow processor, designed to execute arbitrary data flow graphs (DFGs) at 1 GHz at 1.2V. The PDVS data path includes 4 multipliers and 4 adders, each of which uses three header switches to connect to one of three shared V_{DD} rails (V_{DDH}, V_{DDM}, & V_{DDL}). Using more rails provides rapidly diminishing returns. The inputs to the arithmetic components are fed via a programmable crossbar from other computed results, registers, or the 32kb data memory. Level converters (LCs) at the output of each PDVS component prevent short circuit currents. Each 160b word in the 40kb instruction memory gives control signals (including header gate control) to the data path for one clock step of a DFG. The control structure supports nested looping. This architecture provides flexibility for exploring the benefits of PDVS.

The chip contains three additional data paths for direct comparison with PDVS: single-V_{DD} (SV_{DD}: all components share one V_{DD}), multi-V_{DD} (MV_{DD}: each component is permanently tied to one of the three V_{DD}s), and a PDVS data path optimized to support sub-V_T operation. We choose lower V_{DD}s that stretch delay by integer numbers of clock cycles. PDVS uses a fixed frequency clock to serve blocks at the highest rate.

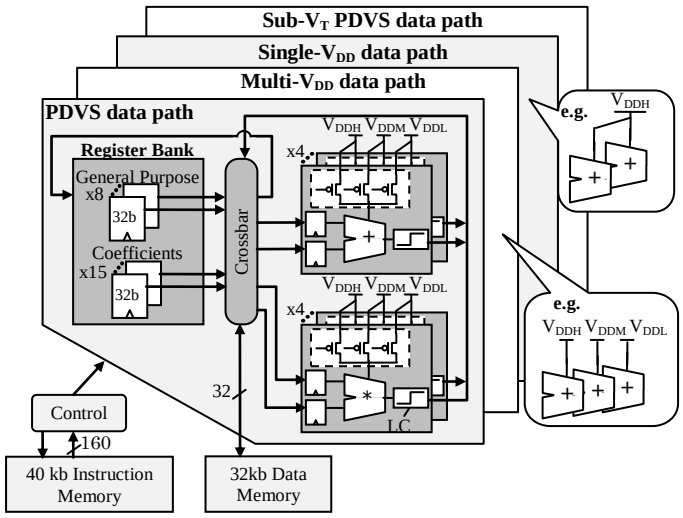

Fig. 2. Block diagram of the PDVS data flow processor. SRAMs and control serve four data paths for direct comparison of PDVS with SV_{DD} & MV_{DD}.

B. V_{DD} switching

Using headers and shared V_{DD} rails instead of DC-DC converters speeds up V_{DD} switching, allowing PDVS to save energy even for brief changes in workload. The simulated overheads of V_{DD} switching correspond to a break even time of less than one multiply operation at V_{DDL} (1.2V down to 0.6V). The speed of the switch depends on the header size; for this processor, the block level V_{DD} can switch in <1ns (equivalent to one clock cycle at the highest frequency). V_{DD} switching creates noise on the shared supplies, but a chip with core level V_{DD} selection using headers [3] shows that this noise is manageable and that slowing the transition by tapering the header turn-on can reduce the noise substantially. Even when using slower transitions with the header to reduce noise, PDVS is >1000x faster than a DC-DC converter tracking time of ~100µs/V [2], so our scheme provides rapid transitions relative to alternatives.

III. PDVS Overheads

There are area, energy and delay overheads for the additional LCs and the headers associated with PDVS compared to SV_{DD} and MV_{DD}. The 32b Kogge-Stone adder and Baugh-Wooley multiplier have 2.4% and 1.7% header switch area overhead, and 11.4% and 2.1% level converter (LC) area overhead, respectively. The LCs have a 32.0% and 2.0% LC simulated delay overhead, and 8.0% and 0.3% LC energy overhead for converting from 0.8V to 1.2V (Fig. 3a) relative to a single add or multiply operation in SV_{DD}. The LC overhead is minor in the overall timing and energy budget, since the multipliers dominate DFG delay and energy. Additionally, there is an energy and delay for having to recover the rail when switching from a lower voltage operation to a higher voltage operation. The adder and multiplier have a delay overhead for charging the virtual rail of 10.4% and 12.0% of the normal operation time (Fig. 3b). There is a 215.3% and 35.0% V_{DD} switching energy overhead, leading to breakeven times of <4 and <1 operation for adds and multiplies. The energy benefits of PDVS overwhelm these overheads in the benchmark DFGs.

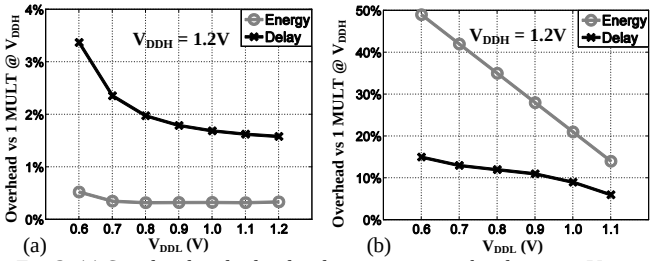

Fig. 3. (a) Simulated multiplier level conversion overhead varying V_{DDL}. (b) Simulated multiplier virtual-V_{DD} switching overhead varying V_{DDL}.

IV. Subthreshold Optimizations

Sub-V_T operation maximizes energy efficiency, and Ultra-DVS [7] supports transitions from full V_{DD} to sub-V_T. Our processor is the first to support rapid and efficient transitions from high performance dithering (V_{DDH}/V_{DDM}) to sub-V_T. Since sub-V_T operation is much slower, dithering rapidly in and out of sub-V_T rarely makes sense. Instead, we hop down to sub-V_T and slow the clock frequency as a mode change. Design changes to the data path help optimize sub-V_T operation. First, we add headers to the register bank and crossbar to allow the entire data path to operate in sub-V_T when performance requirements are low (Fig. 4a). Level converting from sub-V_T to 1.2V requires a special level converter [8]. Since the register file and crossbar are also operating at sub-V_T, during sub-V_T operation the super-threshold level converter in the data path is bypassed to avoid any unnecessary delay, and sub-V_T level converters are enabled to allow the data path to communicate with the data memory (Fig. 4b). Finally, V_{DDL} header and component bulks are tied to the virtual rail [9] to avoid reverse body biasing the PMOS headers thus decreasing the energy per operation by 20% vs. tying it to V_{DDH} body connection (Fig. 5).

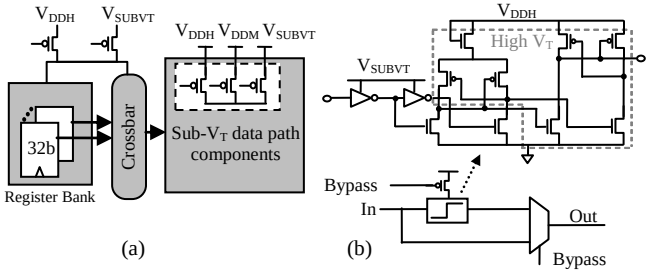

Fig.4. (a) Register bank and cross bar with sub-V_T mode header. (b) Special LC with bypass and power gating capability.

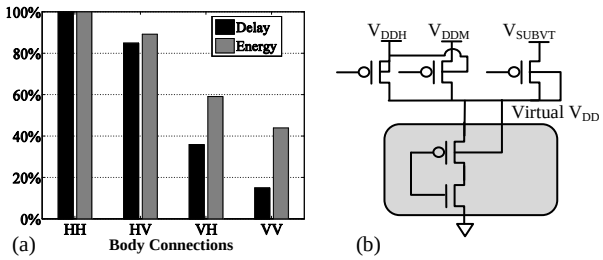

Fig. 5. (a) Simulated delay & energy of adder at 0.3V. Circuit & header bulk (Adder, Header) are tied to V_{DDH} (H) or to virtual V_{DD} (V), eg VV = adder and header bulks tied to virtual V_{DD} (b) Body connections of the sub-V_t data path.

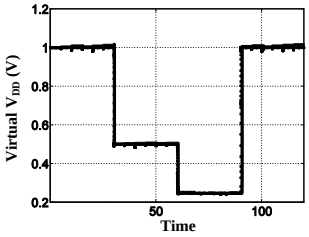

Fig. 6. Sub-V_T data path: Virtual rail during V_{DD} switching. $V_{DDH/M/L}$= 1V, 0.5V, 0.25V with transition times of 5ns, 200ns, & 2ns. Verified with

Subthreshold operation was simulated (Fig. 6) and was functionally verified with hardware measurements for $V_{DDH/M/L}$ values of 1V, 0.5V and 0.25V, respectively, to demonstrate virtual V_{DD} switching capability across a broad voltage range.

V. Instruction and Data Memory Design

The PDVS processor contains a 40kb and 32kb instruction and data memory, respectively. Both memories operate at a nominal voltage of 1.2V and use 6T SRAM. These SRAMs are designed to run at the maximum processor frequency of 1 GHz. The instruction memory is in the critical path of the processor; it must be read each clock cycle. The data memory, however, is only read and written to the start or the end of a DFG iteration, and thus it is not in the single cycle critical path.

Fig. 7 shows a conventional SRAM's read timing. The read operation can be divided into three phases: row decode and word-line (WL) pulse generation (T_{DEC}), bit-line (BL) droop development (T_{BC}), and sense amplifier (SA) resolution and latching (T_{SA}). The cycle time is the sum of these delays.

To reduce the cycle time of the instruction memory, we pipelined the SRAM read operation into two cycles. Fig. 8 shows the modified timing scheme with pipelined sensing. In the first cycle, row decode and BL droop development is completed. The final BL voltages get stored in the inputs of the SA and the column multiplexor signal decouples the BL from the SA inputs. At the beginning of the next cycle, SA enable signal occurs. The key is that during the first phase of a conventional SRAM read access when row decode occurs, the SA lies idle. We use pipelining to efficiently use this duration

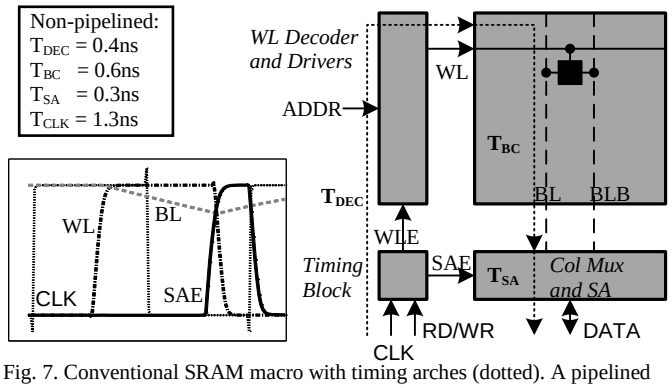

Fig. 7. Conventional SRAM macro with timing arches (dotted). A pipelined SRAM with decoder and BL droop dev in the first stage and SA in the second stage would structurally look the same, but will have a different timing than that shown here. The capacitance of the SA inputs along with the column multiplexor acts as a dynamic pipeline latch.

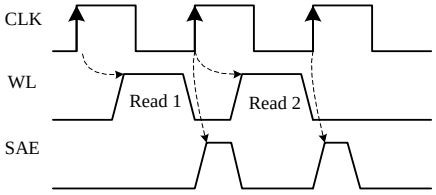

Fig. 8. Frequency-scalable SRAM timing with pipeline sensing scheme

to sense for one read access and decode for the next. This helps lower the cycle time by T_{SA}. Thus, instead of operating at a clock period of 1.3ns, we can operate at 1ns, the specified processor cycle time. Borrowing from the next cycle increases SRAM F_{max} while maintaining 1 cycle throughput. The total read access time, however, remains the same at 1.3ns. The timing pulse generation also scales with frequency. The WL pulse is the inverted clock with a modified duty cycle, and sense amp enable (SAE) is set from the next clock period's rising edge.

This pipelined timing scheme requires the instruction address to be presented to the memory one cycle in advance as compared to a conventional SRAM. Most DFGs have a sequential instruction access pattern, making this constraint simple. However, every time there is a jump, a 1-cycle pipeline penalty is incurred. For our array of sample DFGs, jumps are very infrequent, and the 23% reduction of the cycle time (from 1.3ns to 1ns) was much more beneficial than the loss of performance caused by a stall for jumps.

VI. Measured Results

Fig. 9a shows the measured energy per operation of an add and multiply vs. V_{DD}. Fig. 9b and 9c show the measured delays vs. simulated delay of an add and multiply. We were unable to measure faster delays for higher voltages due to limitations of the on-chip voltage controlled oscillator. Simulations and measured delays match closely for lower voltages at which

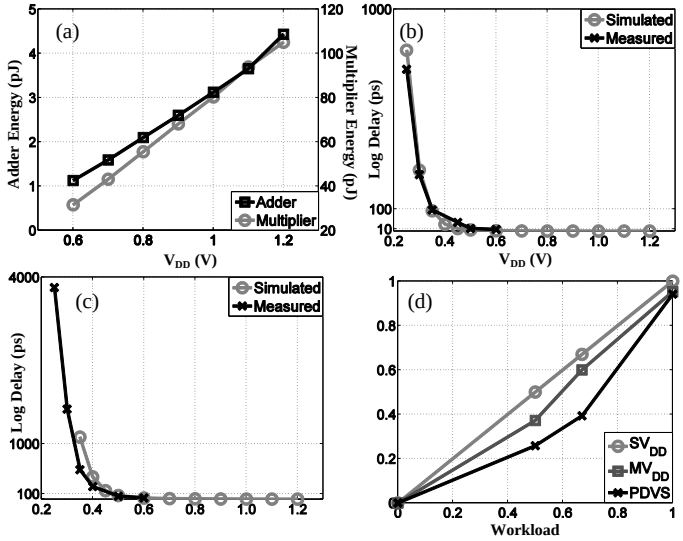

Fig. 9. (a) Measured energy of adder and multiplier vs V_{DD} (b) Measured vs simulated delay of adder vs V_{DD} (c) Measured vs simulated delay of multiplier vs V_{DD} (d) Average measured energy (w/ overheads) vs. workload across 4 different DFGs.

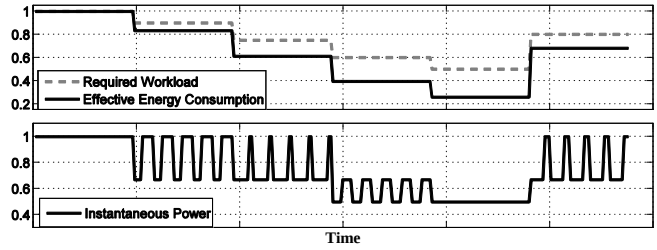

Fig. 10. Change in average power & instantaneous power as the workload changes over time; power waveform shows dithering between two rates to achieve an intermediate rate, resulting in near optimal average energy savings

the VCO was able to generate a clean clock. Measured DFG energies, shown in Fig. 9d, demonstrate PDVS savings across various workloads. For the same area, PDVS gives lower energy for varying workloads than MV_{DD} by switching components to lower V_{DD}s when possible. PDVS headers enable V_{DD} dithering (rapid switching between two V_{DD} rate pairs) to approximate ideal DVS. Dithering is shown by the line between these points. At a rate of 1, the PDVS and MV_{DD} curves are slightly lower energy than SV_{DD} since they remove timing slack in the DFG. This energy profile in Fig. 9d matches the anticipated savings for PDVS and shows how PDVS with dithering closely approximates the ideal savings achievable if we could provide the perfect voltage for each rate of operation. Fig. 10 demonstrates energy savings from dithering for a varying workload, verified by hardware measurements.

Fig. 11a-c shows results for the benchmark DFGs we ran on all the data paths to demonstrate PDVS's benefits over multiple rates. Energy benefits increase as the timing constraint of the DFGs is relaxed. The PDVS data path shows up to 50% and 46% measured energy savings over SV_{DD} and MV_{DD} for the same area, respectively. MV_{DD} can provide the same energy as PDVS for a given DFG by replicating blocks and tying them to different V_{DD}s. PDVS saves up to 65% area (Fig. 11d) for the optimal energy schedule by reusing components over MV_{DD}, since individual components are not statically assigned to a voltage. Table 2 shows the chip summary.

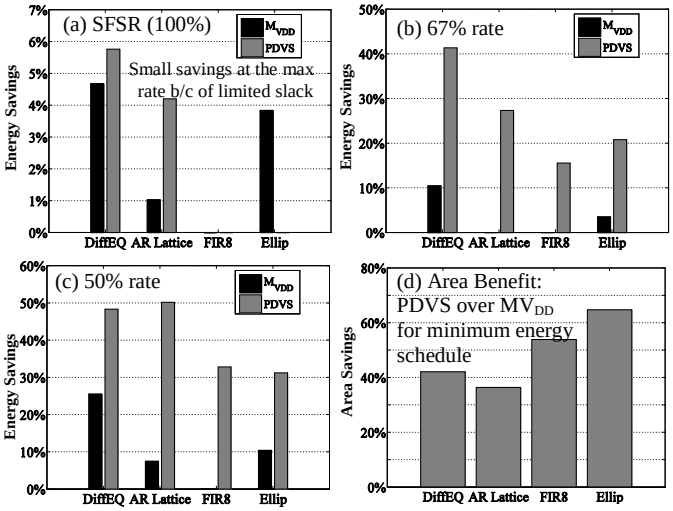

Fig. 11. (a-c) Measured energy benefit (including overhead) of PDVS & MV_{DD} vs SV_{DD} for single function single rate (SFSR) & single function multi rate (SFMR) at 67% and 50% at constant area (d) Area benefit of PDVS over MV_{DD}.

VII. Conclusion

A 90nm PDVS data flow processor demonstrates single clock cycle V_{DD}-switching at the component level, integrated V_{DD} dithering for near optimal energy scalability, and the capability to switch efficiently between high performance DVS and subthreshold modes. We also showed the benefits of energy savings over MV_{DD} through the use of PDVS headers. Compared to DVS implementations that must change the output voltage of a DC-DC converter, our fine grained voltage scaling allows the chip to save energy for rapid variations in workload down to the single operation level. We demonstrated measured energy savings in benchmark DFGs of up to 50% and 46% over SV_{DD} and MV_{DD} for a minimal area overhead. The ability to operate in subthreshold and to power gate idle blocks lets the processor provide a minimum energy mode when performance requirements are low. This processor demonstrates the potential for larger energy efficient systems and SoCs using PDVS.

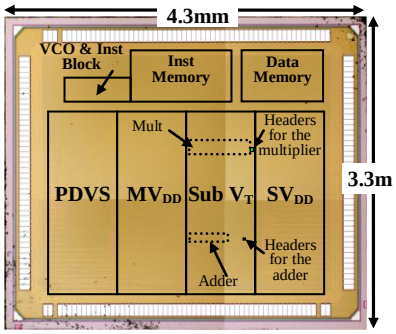

Fig. 12. Annotated die photo showing PDVS, SV_{DD}, MV_{DD} and sub-V_T PDVS data paths.

Table II. Chip Summary

Feature	*This Chip*
Process	90nm CMOS Bulk w/ Dual V_T
Area	4.3mm x 3.3mm
Transistor Count	~2 million
V_{DD}	250mV – 1.2V
SRAMs	40kb & 32kb

Acknowledgment

This work was supported by DARPA, AMD, and NSF CCF-0903471.

References

[1] M. Putic et al., "Panoptic DVS: A Fine-Grained Dynamic Voltage Scaling Framework for Energy Scalable CMOS Design," *ICCD*, pp.491-497, 2009.

[2] C. Zheng and D. Ma, "A 10MHz 92.1%-Efficiency Green-Mode Automatic Reconfigurable Switching Converter with Adaptively Compensated Single-Bound Hysteresis Control," *ISSCC*, pp.204-205, 2010.

[3] J. Howard et al., "A 48-Core IA-32 Message-Passing Processor with DVFS in 45nm CMOS," *ISSCC* , pp. 22-33, 2010.

[4] D. Truong et al, "A 167-processor 65 nm Computational Platform with Per-Processor Dynamic Supply Voltage and Dynamic Clock Frequency Scaling," *Symposium on VLSI Circuits*, pp.22-23, 2008.

[5] B. Nam et al., "A 52.4mW 3D Graphics Processor with 141Mvertices/s Vertex Shader and 3 Power Domains of Dynamic Voltage and Frequency Scaling," *ISSCC*, pp.278-603, 2007.

[6] V. Gutnik and A.P. Chandrakasan. "Embedded Power Supply for Low-Power DSP," *TVLSI*, vol.5, no.4, pp.425-435, Dec 1997.

[7] B.H. Calhoun and A.P. Chandrakasan, "Ultra-Dynamic Voltage Scaling (UDVS) using Sub-Threshold Operation and Local Voltage Dithering," *JSSCC*, vol.41, no.1, pp. 238- 245, Jan. 2006.

[8] S.N. Wooters et al., "An Energy-Efficient Subthreshold Level Converter in 130-nm CMOS," *IEEE Trans. Circuits Syst. II* , vol.57, no.4, pp.290-294, April 2010

[9] B.H. Calhoun et al., "Flexible Circuits and Architectures for Ultralow Power," *Proceedings of the IEEE* , vol.98, no.2, pp.267-282, Feb. 2010.