A 32 b 90 nm Processor Implementing Panoptic DVS Achieving Energy Efficient Operation From Sub-Threshold to High Performance

Kyle Craig, Yousef Shakhsheer, Saad Arrabi, Sudhanshu Khanna, John Lach, *Senior Member, IEEE*, and Benton H. Calhoun, *Senior Member, IEEE*

***Abstract*—This paper presents a 32 b, 90 nm data flow processor capable of executing arbitrary DSP algorithms using fine grained Dynamic Voltage Scaling (DVS) at the component level with rapid V_{DD} switching and V_{DD} dithering for near-ideal quadratic dynamic energy scaling from 0.25 V–1.2 V. This is the first full processor with Panoptic (all-inclusive) DVS, single clock cycle V_{DD} switching, V_{DD} dithering, and the ability to switch between high performance DVS operation and a sub-threshold mode of operation. This paper also explores V_{DD} header switching and voltage selection considerations for additional savings. Measurements show up to 80% and 43% energy savings of using PDVS over single V_{DD} (SV_{DD}) and multi-V_{DD} (MV_{DD}), respectively. Additionally, PDVS shows area savings of up to 65% over MV_{DD} given the same energy consumption.**

***Index Terms*—Dynamic voltage scaling, energy efficiency, sub-threshold CMOS circuits, ultra low voltage design.**

I. Introduction

ENERGY efficiency has emerged as a critical metric for modern integrated circuits across essentially the entire application design space. For designs in the low power, low performance design space, energy efficient operation is needed to extend battery lifetimes or even to allow for battery-less operation running solely on harvested energy. For designs in the high performance design space, energy efficient operation reduces hot spots, lowers cooling costs, and avoids dark silicon issues. Many systems across this broad design space have applications that require high performance. However, due the varying nature of their applications, the workload requirements remain below this upper limit for the majority of their lifetime. Since different applications have varying workload requirements an energy efficient solution, such as dynamic voltage scaling (DVS), is needed.

Dynamic voltage scaling is the conventional solution for adjusting energy consumption based on varying workload requirements. When timing slack exists in a given application, DVS adjusts the supply voltage (V_{DD}) and frequency to match a circuit's workload, providing quadratic energy savings at these lower workload requirements. Traditionally, DVS implementations suffer from coarse spatial and temporal granularities. Spatial granularity is the ability to assign different components in a design to different voltages. Most recent DVS implementations are limited to a spatial granularity at the microprocessor core level to entire chip [1]–[3]. Temporal granularity refers to the speed at which the V_{DD} to a component can change. DVS techniques generally rely on DC-DC converters to adjust V_{DD}. These off-chip DC-DC converters traditionally limited temporal granularity, since they take tens to hundreds of μsecs to adjust the V_{DD} [4]. The coarse spatial and temporal granularity of traditional DVS limits the energy efficiency of these systems.

Further, these coarse grained DVS blocks are typically supplied a voltage that is generated directly by a DC-DC converter. Practical cost considerations of DC-DC converters, such as on-chip area and off chip passives, can limit the number of blocks that can be supplied with separate supply voltages. More recent work on integrated DC-DC converters show significant speedups in switching time, for example $>$ a 1 V transition is achieved in roughly 20 ns in [5], but including dedicated DC-DC converters for each block is still impractical for designs with fine grained spatial granularity that have many distinct power regions. To maximize energy efficiency for varying workload requirements, DVS would ideally support voltage control across a broad range for multiple blocks with fine grained spatial and temporal granularity.

To meet these needs for improving energy efficiency, we propose to use a method called Panoptic ("all-inclusive") Dynamic Voltage scaling (PDVS). To improve spatial and temporal granularity, PDVS uses multiple PMOS header switches at the component level (Fig. 1) to provide a local V_{DD} (virtual-V_{DD}) from a discrete set of chip-wide shared V_{DD}s (e.g., V_{DDH}, V_{DDM}, V_{DDL}). This allows for an individual components' virtual-V_{DD} to be set independently from any other component as well as allowing for fast local V_{DD} switching. A more extensive background on the benefits of PDVS can be found in [6]. The use of voltage dithering [7], or using a division of operations across two voltage/frequency points to approximate an effective intermediate operating point, further enables the approach to closely approximate an ideal energy/performance tradeoff across a broad range.

This paper describes the first processor that was implemented using the PDVS architecture. To our knowledge, this is the first

Manuscript received June 03, 2013; accepted September 04, 2013. Date of publication November 14, 2013; date of current version January 24, 2014. This paper was approved by Associate Editor Dejan Markovic.

K. Craig, Y. Shakhsheer, S. Arrabi, J. Lach, and B. H. Calhoun are with the Department of Electrical Engineering, University of Virginia, Charlottesville, VA 22904 USA (e-mail: KyleCraig@virginia.edu).

S. Khanna was with the University of Virginia, Charlottesville, VA 22904 USA, and is now with Texas Instruments, Dallas, TX 75081 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSSC.2013.2285384

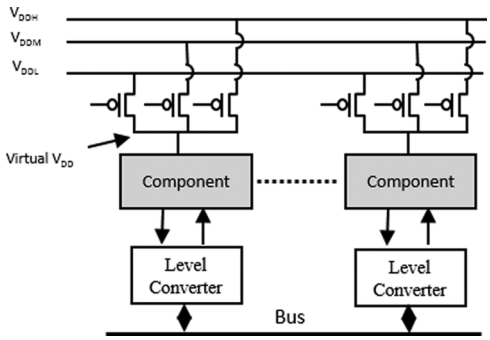


Fig. 1. PDVS architecture, which enables fine spatial and temporal granularity DVS granularity [8].

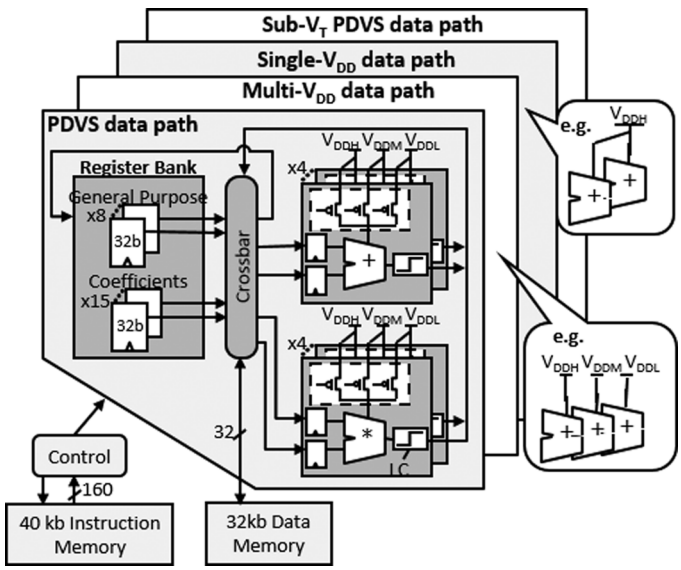


Fig. 2. Block diagram of the PDVS data flow processor. SRAMs and control serve four data paths for direct comparison of PDVS with $\mathrm{SV_{DD}}$ & $\mathrm{MV_{DD}}$ [8].

full processor with PDVS, single clock cycle $\mathrm{V_{DD}}$ switching, $\mathrm{V_{DD}}$ dithering, and the ability to switch between high performance DVS operation and a sub-threshold (sub-$\mathrm{V_T}$) mode of operation. The paper is organized as follows. In Section II, we introduce the PDVS processor, focusing on the components used to create the data flow processor, memory design, $\mathrm{V_{DD}}$ switching methodology, voltage selection methodology, and the overheads associated with PDVS. In Section III, we focus on the architecture and circuit enhancements that enable sub-threshold operation. In Section IV, we describe our test setup and testing methodology, show results from a demonstration, and present measured results. Section V concludes the paper.

II. Panoptic Dynamic Voltage Scaling (PDVS)

A. 32 Bit Processor

To explore the full benefits of PDVS, we designed a 32 bit data flow processor, Fig. 2, capable of executing arbitrary data flow graphs (DFGs) at 1 GHz at 1.2 V. We used the PDVS architecture to implement the data path of the processor. The data path consists of four Baugh-Wooley multipliers and four Kogge-Stone adders. Each of these components uses three PMOS header switches tied to the three $\mathrm{V_{DD}s}$ ($\mathrm{V_{DDH}}$, $\mathrm{V_{DDM}}$, $\mathrm{V_{DDL}}$) that are common throughout the processor. The processor includes a programmable crossbar that feeds input registers of the data path components either directly from the datapath, the register bank, or the memory. To prevent short circuit current from blocks operating below the nominal $\mathrm{V_{DD}}$, level converters (LCs) are used at the output of each multiplier and adder to up-convert their outputs to the $\mathrm{V_{DDH}}$ level that is used at the register file.

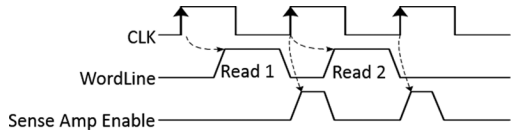


Fig. 3. Frequency-scalable SRAM timing with pipeline sensing scheme [8].

In order to provide a fair hardware comparison to PDVS, we include three additional data paths on the chip that are functionally identical but that use different power management options: single-$\mathrm{V_{DD}}$ ($\mathrm{SV_{DD}}$), multi-$\mathrm{V_{DD}}$ ($\mathrm{MV_{DD}}$), and a sub-$\mathrm{V_T}$ optimized PDVS data path. In the $\mathrm{SV_{DD}}$ data path, the four multipliers and adders all share the same $\mathrm{V_{DD}}$. In the $\mathrm{MV_{DD}}$ data path, the four multipliers and adders are permanently tied to either $\mathrm{V_{DDH}}$, $\mathrm{V_{DDM}}$, or $\mathrm{V_{DDL}}$, and operations can be scheduled for execution on any of these components based on the timing requirements. The processor has a 32 kb data memory and a 40 kb instruction memory that are shared for all of the data paths. The control word for controlling the data flow (and header control where applicable) of the various data paths is 160 b for this test chip.

B. Memory Design

The PDVS processor contains a 40 kb data memory and a 32 kb instruction memory. Since the focus of the processor was to evaluate the benefits of the PDVS data path compared to the other data paths, the memories operate at a nominal voltage of 1.2 V and use 6 T 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; the 160 b control word must be read every clock cycle. However, the data memory is not on the critical path since it is only read and written at the start or the end of a DFG iteration. To reduce the cycle time of the instruction memory, instead of using a conventional timing scheme, we pipeline the SRAM read operation into two cycles. Fig. 3 shows the modified timing scheme with pipelined sensing. In the first cycle, we row decode and allow the bit line (BL) droop to develop. Since the column multiplexor signal decouples the BL voltage from the sense amplifier (SA) input, the BL voltages are effectively stored at the inputs of the SA. In the second cycle, the SA enable signal triggers. We are able to do this pipelining since during the first phase of a conventional SRAM read access when row decode occurs the SA is idle. Using this scheme we were able to reduce the clock period from 1.3 ns to 1 ns, which was our specified processor cycle time. 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 relatively infrequent.

C. V_{DD} Switching

To improve energy efficiency, PDVS uses headers and three global V_{DD} rails instead of dedicated block level DC-DC converters. This architecture speeds up virtual-V_{DD} switching, allowing PDVS to lower energy even for brief changes in workload that cannot be realized in conventional DVS implementations. The delay of switching the virtual-V_{DD} depends on the header size, but for our processor, this time is less than our target clock period. V_{DD} switching creates noise on the shared supplies however [2] shows how gradually turning on headers can reduce noise, while [9] shows that switching to an intermediate voltage can also reduce noise.

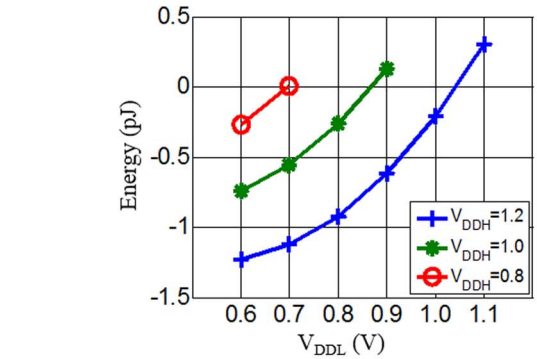
PDVS's structure and savings bring about an interesting exploration of the optimal switching control scheme. We separate switching into two cases: switching from a higher voltage operation to a lower voltage operation and switching from a lower voltage operation to a higher voltage operation. When switching from a higher voltage to a lower voltage, if the voltage difference is large enough, the switching will result in a "free" operation at V_{DDL}. Fig. 4(a) shows the energy per operation consumed across two operations, where the first occurs at a higher voltage and the second occurs at a lower voltage. The negative energy shown in the figure occurs when current enters the source of the supply. In our design, this energy would be reused through the power distribution network or would charge the capacitor at the driving DC-DC converter. In other designs, this negative energy could be lost, resulting in the operation costing leakage energy.

We investigated two strategies for switching from a high voltage operation to low operation to investigate how to make use of the extra charge that is being dissipated from the virtual rail as shown in Fig. 4(a). The two methodologies are turning off all headers (powergating) and keeping the lower voltage header (WithOp) on during the transition (Fig. 4(c)). Fig. 4(b) shows the comparison of the two methods over three additions (Powergating: V_{DDH}, off, V_{DDL}; WithOp: V_{DDH}, V_{DDL}, V_{DDL}). For both header connection strategies, we execute operations during all three cycles. WithOp shows an energy savings over the powergated approach by allowing current to flow back into the V_{DDL} rail instead of severing the connection.

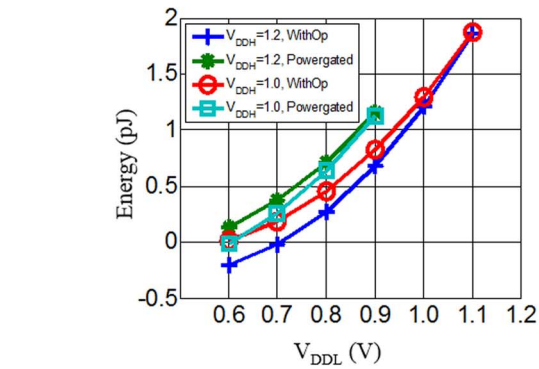
During a switching operation from a low voltage operation to a high voltage operation, an operation needs to be completed, and the virtual-V_{DD} rail must be charged to the higher voltage. We investigated two approaches to doing this: allowing the operation to run and rail to recover in the same operation or running a NOP for a single clock cycle to allow time for the virtual rail to settle to the high voltage and then running the operation. Both methods consume roughly the same amount of energy consumed over the three additions. The NOP approach carries a time penalty for recovery but can reliably execute the second operation in the time constraint.

D. Voltage Selection

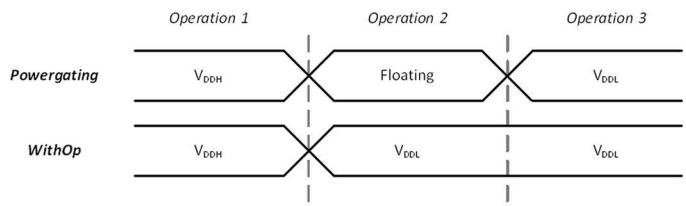
With PDVS, we only have a limited number of voltage rails (e.g., three in our examples). It is very important to choose these voltage rails appropriately, since these values have a direct impact on the efficiency of the DFG schedule at the available rates. For example, having high voltages values will increase



(a)



(b)



(c)

Fig. 4. (a) Simulated energy per operation while switching from V_{DDH} to V_{DDL} (b) Simulated energy of running three operations with and without a NoOp across various V_{DD}s (c) Timing diagram of the switching methodologies.

the schedule efficiency when operating at the higher rates due to the ability to exploit the small timing slack. However, it will limit the efficiency of the schedule at the lower rates. If the rail voltages are spaced too far apart, the efficiency of the schedules of all rates is reduced, but it enables a broad range of potential operating rates. Ideally, for optimal efficiency, we would need to know the exact applications and operating rates to choose the best voltage values. However, since this is impractical, we assume a uniform distribution of the rates between 1 and 0.2 (normalized to the fastest rate possible), which encompasses a wide range of applications. In our test setup, the system operates at any voltage between 1.2 V and 0.6 V with 0.1 V increments. To show the energy differences between various voltage selection methods, we compare four different voltage selection techniques (Fig. 5). The different voltage selection techniques are described as follows:

High Rate: $V_{DDH/M/L}$ of 1.2 V, 1.1 V, and 1.0 V, respectively.

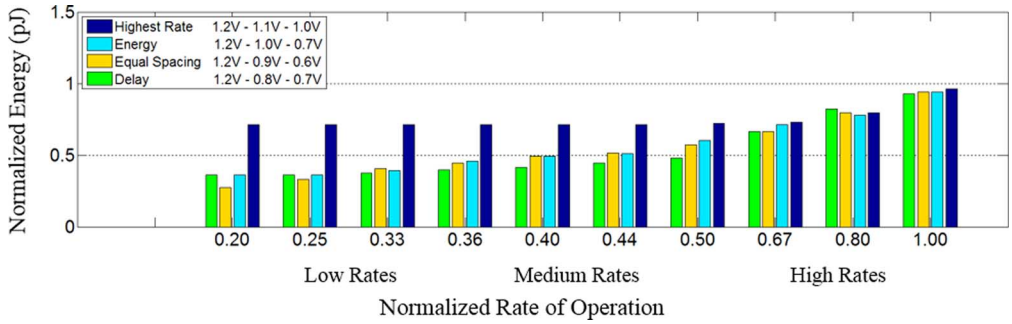


Fig. 5. Estimated DFG energy for various V_{DDH}, V_{DDM}, V_{DDL} configurations and rates.

Energy: $V_{DDH/M/L}$ of 1.2 V, 1.0 V, and 0.7 V, respectively. Choosing the voltages to achieve 1 ×, 0.5 ×, and 0.33 × of the energy of the adder

Equal Spacing: $V_{DDH/M/L}$ of 1.2 V, 0.9 V, and 0.6 V, respectively. Choosing the voltages with equal spacing

Delay: $V_{DDH/M/L}$ of 1.2 V, 0.8 V, and 0.7 V, respectively. Choosing the voltages to achieve 1 ×, 2 ×, and 3 × of the delay of the adder.

Fig. 5 shows the average energy results of our comparison for simulations of across all seven benchmarks. The highest rate technique is able to achieve the lowest power consumption at .95 but cannot take advantage of slack at lower rates. The energy, equal spacing, and delay techniques provide good savings opportunities at all rates. The differences in energies are caused by "sweet spots" in the rates. This happens when one technique's voltage is more optimal than others near a given rate target, requiring those techniques to have to run operations at a higher voltage. Though it is possible to have a voltage selection of "Low Rate" with the V_{DD}s at 0.8 V, 0.7 V, and 0.6 V, this system would never be able to achieve the highest system rate and therefore is not considered. We choose to use the "Delay" voltage selection for our system. It is able to achieve desirable energy results since energy consumption is our main metric. Additionally, having components at integer delay simplifies re-timing by allowing us to schedule for the register file to latch results from lower voltage components after an integer number of clock cycles and increases processor utilization since the components will have little slack in this retiming scheme.

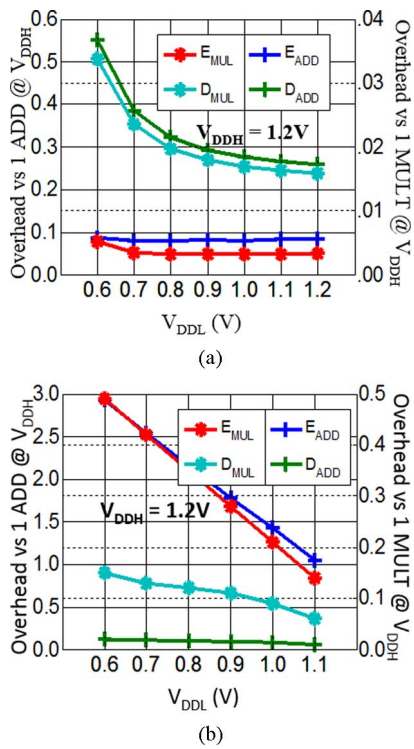


Fig. 6. (a) Simulated level conversion overhead varying V_{DDL} for both the adder and multiplier. (b) Simulated virtual-V_{DD} switching overhead varying V_{DDL} for both the adder and multiplier.

E. PDVS Overheads

There are overheads associated with the PDVS architecture compared to SV_{DD} and MV_{DD}. The primary overheads are the area, energy, and delay overheads associated with the inclusion of LCs and the headers associated with PDVS compared to SV_{DD} and MV_{DD}. The adder and multiplier have 2.4% and 1.7% header area overhead, and 11.4% and 2.1% level converter (LC) area overhead, respectively. From simulations, we see the LCs have a 32.0% and 2.0% LC delay overhead, and 8.0% and 0.3% LC energy overhead for converting from 0.8 V to 1.2 V (Fig. 6(a)) relative to a single addition or multiplication operation in SV_{DD}. From simulations, we also see the headers have a 35% and 12% delay overhead, and 215% and 10% energy overhead switching the virtual-V_{DD} from 0.8 V to 1.2 (Fig. 6(b)) relative to a single add or multiply operation in SV_{DD}. Although the overheads may appear large at first, they actually are minor in the overall timing and energy budget, since the multiplier dominates DFG delay and energy. Using the header switches to switch the components' virtual-V_{DD} from a lower V_{DD} to a higher V_{DD} (i.e., V_{DDL} to V_{DDH}) incurs both energy and delay overheads. For the adder and multiplier, this energy overhead leads to breakeven times of < 4 and < 1 operations for the adder and multipliers, respectively. This means that the multiplier can switch to the low voltage, execute just one instruction, and then switch back to the high voltage and save energy relative to executing that one instruction at the high voltage. The adder however, must execute four consecutive instructions at the low voltage in order to overcome the energy overhead of V_{DD} switching. As mentioned previously, since the multiplier dominates the DFG energy, the energy benefit of PDVS overwhelms the overheads.

III. Sub-Threshold Operation

An emerging trend in energy efficient operation, specifically for applications with low throughput requirements such as sensor nodes, is sub-threshold (sub-V_T) operation [10], [11]. One common application for sub-V_T operation is in medical sensors, such as ECG shown in [12]. In [13] a sub-V_T SoC was shown to be capable of running battery-less, solely off harvested energy. Designing for sub-V_T and super-threshold operation is a challenge, the authors in [14] explore many of the design considerations for operating across a wide voltage range. The authors in [15] however use multiple cores: one designed for nominal operation, and two designed for sub-V_T. Memory design in sub-V_T has many design challenges [16], [17], since sub-V_T memory design was not our focus, we kept our memories in super-threshold. We will show how a PDVS architecture can be used to enable sub-V_T operation. With sub-V_T, the component's V_{DD} (or virtual-V_{DD}) is lowered below the device threshold voltage, drastically reducing energy. In our sub-V_T enabled data path, the V_{DDH} and V_{DDM} supply rails are the same as before, and the V_{DDL} rail becomes the sub-threshold supply, V_{SUBVT}. Our processor is the first to support rapid and efficient transitions from high performance dithering between V_{DDH} & V_{DDM} to sub-V_T operating mode with V_{SUBVT}. In sub-V_T, drastically reduced energy comes at a cost of much slower operation. With this cost in mind, rapidly dithering between V_{SUBVT} and V_{DDH}/V_{DDM} rarely makes sense. Instead, we infrequently transition into sub-V_T, only using it as a data path mode change when workload requirements are severely relaxed.

A. Architecture Enhancements

In order to achieve sub-V_T operation and maintain super-threshold operation as well, design changes need to be made to optimize the data path. In our traditional PDVS implementation, headers were placed on the arithmetic components, but the crossbar and register bank were hard-tied to V_{DDH}. For every component, short circuit current was avoided by level converting from the virtual-V_{DD} up to the nominal V_{DD} after every operation. However, in the sub-V_T data path, leaving the crossbar and register bank at the nominal V_{DD} would be inefficient during sub-threshold operation. Instead we add two headers (V_{DDH}, V_{SUBVT}) to these blocks to allow them to operate in sub-V_T (Fig. 7(a)). During sub-V_T operation since the crossbar, register bank, and arithmetic components are all operating at V_{SUBVT}; we bypass the LCs that are normally used for super-threshold operation at the output of the arithmetic components to avoid energy and delay penalties. For communication with the super-threshold memories, a special design LC capable of converter from sub-V_T up to 1.2 V was used [18]. We use a similar bypass scheme as described for the super-threshold LCs, for the sub-V_T LCs (Fig. 7(b)). We only utilize the sub-V_T LCs while operating in sub-V_T, avoiding any unnecessary delay and energy overheads while operating in super-threshold. Special consideration needs to be taken when deciding how to tie the bulk connections of the V_{SUBVT} header and the component. For ease, in the PDVS data path, all the header bulk connections and component bulk connections were tied to V_{DDH}. In the sub-V_T data path, leaving the bulk connection at V_{DDH} would lead to reverse biasing of the header and component [19]. Instead, by tying the bulk to the virtual-V_{DD}, we are able to decrease the energy per operation by 20% when compared to V_{DDH} (Fig. 7(c),(d)). Sub-threshold operation was simulated and was functionally verified in hardware for $V_{DDH/M/L}$ values of 1 V, 0.5 V and 0.25 V, respectively, to demonstrate virtual V_{DD} switching capability across a broad voltage range.

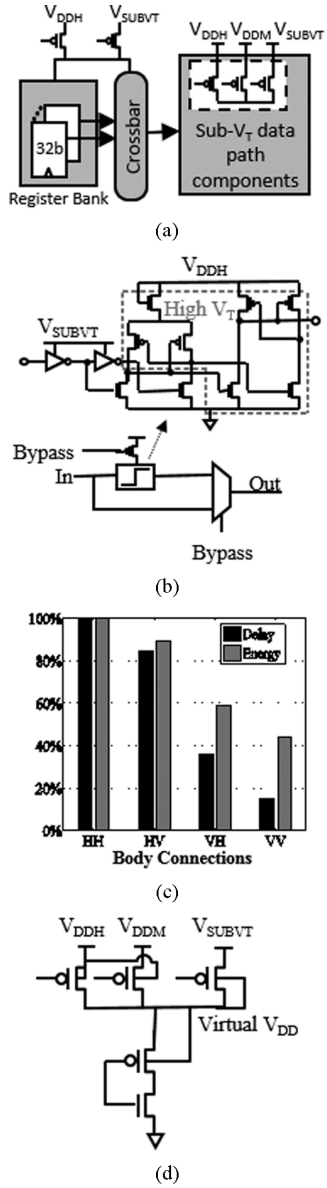


Fig. 7. (a) Register bank and cross bar with sub-V_T mode header. (b) Special LC with bypass and power gating capability. (c) Simulated delay & energy of adder at 0.3 V. Circuit & header bulk (Adder, Header) are tied to V_{DDH} (H) or to virtual V_{DD} (V), eg $\mathrm{VV} = \mathrm{adder\ and\ header\ bulks\ tied\ to\ virtual}\ V_{DD}$ (d) Body connections of the sub-V_t data path [8].

IV. Measured Results

A. Test Setup

In order to compare post fabricated results with our design, we set up a test platform that generates inputs and compares outputs and allows us to run the same benchmarks on a VHDL model, Spectre netlist, and physical hardware. The benchmarks are developed in VHDL, and custom scripts translate them

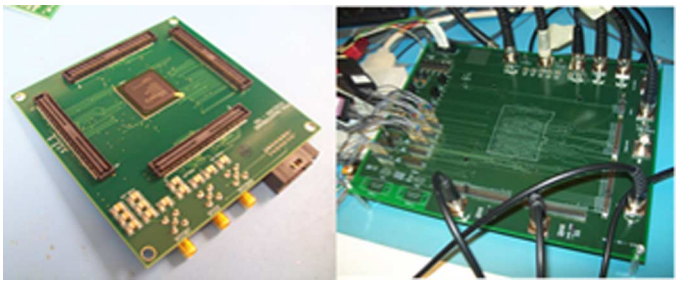


Fig. 8. (left) FPGA daughter board; (right) main testing board.

into a Spectre stimulus file and a VHDL state machine for the Spectre simulation and hardware testing, respectively. Only the Spectre simulation and test chip are used to measure energy. We use a custom synthesis script to map the benchmark DFGs to the architecture and use Matlab to create the 160 b instruction words. We designed two custom printed circuit boards (PCBs) to test and measure the four different data paths on our test chip. To provide testing flexibility, one of the PCBs contains a Xilinx Spartan-6 XC6SLX150 FPGA (Fig. 8). As previously mentioned, our comparison is between three energy efficient topologies: $\mathrm{SV_{DD}}$, $\mathrm{M_{VDD}}$ and PDVS. To achieve a fair comparison, all of the measured data in this section comes from a single data path (PDVS) implementing various techniques to emulate the other two energy efficient topologies. To emulate $\mathrm{SV_{DD}}$, we power the three rails using the same voltage source and enable all the headers to minimize resistance across the header. To emulate $\mathrm{MV_{DD}}$, we assign a different voltage to each rail and make sure each component is powered by only one of these rails at all times (i.e., no header switching is allowed).

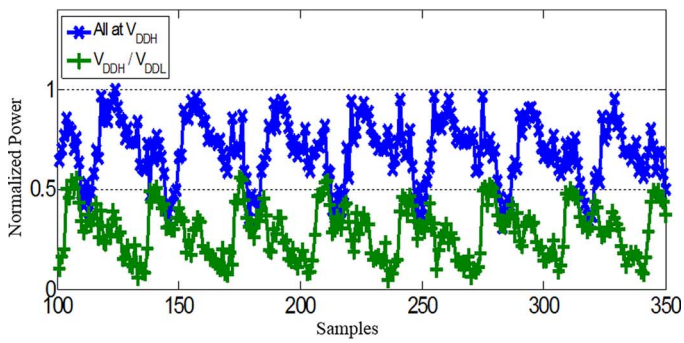
B. Brightness Demonstration

As a demonstration of the benefits of PDVS, we implemented a video processing application that brightens dimly lit areas of a frame. The workload of this application varies according to the number of dark pixels of each frame. The number of dark pixels that need to be brightened can easily be calculated from each image. With this information, we can compute the workload needed to achieve the required application rate, e.g., 24 frames per second (FPS) or 30 FPS. For the sake of simplicity, we created a program that brightens pixels by multiplying the pixels below a threshold value by a specific constant. The multiplication can be done either at $\mathrm{V_{DDH}}$ or $\mathrm{V_{DDL}}$ (i.e., fast or slow). With the knowledge of the total number of pixels that need brightening, we calculate the number of multiplications that will be scheduled at $\mathrm{V_{DDH}}$ and the multiplications scheduled at $\mathrm{V_{DDL}}$.

Since our test chip was a custom processor designed to demonstrate the benefits of PDVS, it does not contain any operating system that enables video data input, so we feed each frame as an image to the input data. We can see in the Fig. 9(a)/(b) the demo images before and after the processing that was executed on our test chip. The graph in Fig. 9(c) shows the measured instantaneous power consumption during the demo for all $\mathrm{V_{DD}}$s set to the $\mathrm{V_{DDH}}$ value (1.2 V) and with $\mathrm{V_{DDH}}$ at 1.2 V and $\mathrm{V_{DDL}}$ at 0.7 V. To obtain this data, we used a lower demo frequency to accommodate data transfer to/from

(a)

(b)



(c)

Fig. 9. (a) Original picture fed into chip (b) post processed image from the chip (c) normalized measured instantaneous power from our demonstration. All at $\mathrm{V_{DDH}}$ has all $\mathrm{V_{DD}}$s at the nominal 1.2 V, $\mathrm{V_{DDH}}/\mathrm{V_{DDL}}$ are set at 1.2 V and 0.7 V, respectively.

the chip and removed leakage. With this example, we are able to see a 40% average power reduction by using PDVS.

C. Results

Our test chip was fabricated in a commercial 90 nm bulk CMOS process. The average of seven measured DFG energies, shown in Fig. 10, demonstrates PDVS savings across various workloads. Given the same area constraint, PDVS gives lower energy for varying workloads than $\mathrm{MV_{DD}}$ by operating components at lower $\mathrm{V_{DD}}$s when possible while $\mathrm{MV_{DD}}$ components are hard-tied to higher voltages. PDVS headers enable $\mathrm{V_{DD}}$ dithering (rapid switching between two $\mathrm{V_{DD}}$ rate pairs) to approximate ideal DVS, providing the energy vs. rate profile that lies on the line between these points. With a nominal workload of 1, the PDVS and $\mathrm{MV_{DD}}$ curves have slightly lower energy than $\mathrm{SV_{DD}}$ since timing slack was removed by running some components at lower $\mathrm{V_{DD}}$s.

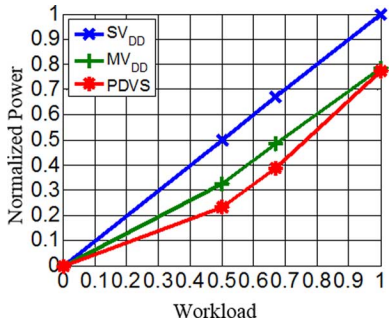


Fig. 10. Average measured power (w/ overheads) vs. workload across seven different DFGs.

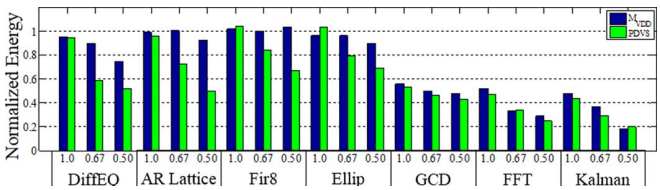


Fig. 11. Measured energy benefit (including overhead) of PDVS & $\mathrm{MV_{DD}}$ normalized to $\mathrm{SV_{DD}}$.

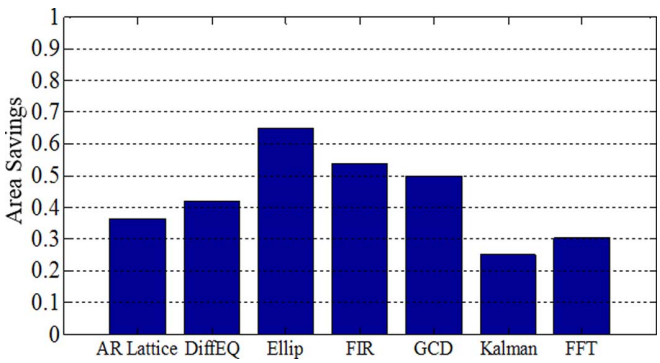


Fig. 12. Area savings of PDVS over $\mathrm{M_{VDD}}$ for the same energy constraint.

Fig. 11 shows results for the seven benchmark DFGs we ran on all the data paths to demonstrate PDVS's benefits for various rates. As the workload rate decreases, the energy benefits increase due to the timing constraint being relaxed. For the given DFGs, the PDVS data path shows up to 80% and 43% energy savings over $\mathrm{SV_{DD}}$ and $\mathrm{MV_{DD}}$, respectively. The largest energy savings were in the FFT DFG. This is because the FFT heavily uses the multiplier, which dominates the energy budget. Given unlimited area, $\mathrm{MV_{DD}}$ can theoretically provide the same energy as PDVS for a given DFG by having the exact number of components needed at a given $\mathrm{V_{DD}}$ to implement a given DFG across workloads. To illustrate this point, assume a trivial DFG requires three multiplies at $\mathrm{V_{DDH}}$ for a workload of 1, two multiplies at $\mathrm{V_{DDM}}$ and one at $\mathrm{V_{DDL}}$ for a workload of 0.66, and three multiplies at $\mathrm{V_{DDL}}$ for a workload of 0.5. PDVS could achieve all workloads with only three multipliers, while $\mathrm{M_{VDD}}$ would require eight. For our non-trivial DFGs given the same energy constraint, PDVS saves up to 65% area (Fig. 12) over $\mathrm{MV_{DD}}$, since it allows individual components to be reused at different voltages while $\mathrm{MV_{DD}}$ would require multiple copies of each component at different voltages.

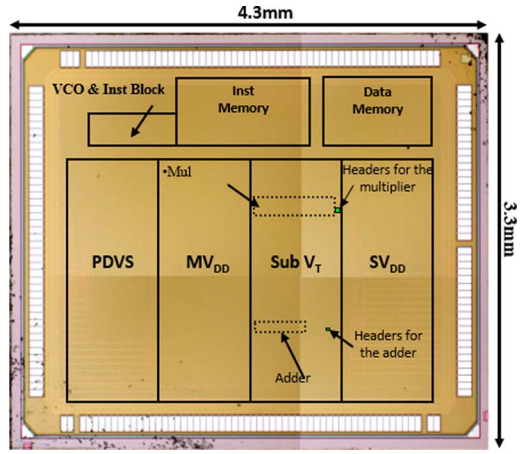


Fig. 13. Annotated die photo [8].

TABLE I
CHIP SUMMARY [8]

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

V. CONCLUSION

This paper presents the first processor implementing using the PDVS architecture. Our 32 bit data flow processor was designed and fabricated in a conventional 90 nm CMOS process. This processor demonstrates single clock cycle $\mathrm{V_{DD}}$-switching at the component level, implements integrated $\mathrm{V_{DD}}$ dithering for near optimal energy scalability, and can switch efficiently between high performance DVS and a sub-threshold mode of operation. Unlike traditional DVS implementations, PDVS does not use dedicated DC-DC converters, and our fine grained voltage scaling allows the chip to save energy for rapid variations in workload even at the component level. Through spatial and temporal granularity, PDVS is able to improve energy efficiency over $\mathrm{SV_{DD}}$ and $\mathrm{MV_{DD}}$. We show measured energy savings in seven benchmark DFGs of up to 50% and 46% over $\mathrm{SV_{DD}}$ and $\mathrm{MV_{DD}}$.

REFERENCES

[1] J. Howard *et al.*, "A 48-core IA-32 message-passing processor with DVFS in 45 nm CMOS," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2010, pp. 22–33.

[2] D. Truong, W. Cheng, T. Mohsenin, Z. Yu, T. Jacobson, G. Landge, M. Meeuwsen, C. Watnik, P. Mejia, A. Tran, J. Webb, E. Work, Z. Xiao, and B. Baas, "A 167-processor 65 nm computational platform with per-processor dynamic supply voltage and dynamic clock frequency scaling," in *Symp. VLSI Circuits Dig.*, 2008, pp. 22–23.

[3] B. Nam, J. Lee, K. Kim, S. Lee, and H. Yoo, "A 52.4 mW 3D graphics processor with 141 Mvertices/s vertex shader and 3 power domains of dynamic voltage and frequency scaling," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2007, pp. 278–603.

[4] C. Zheng and D. Ma, "A 10 MHz 92.1%-efficiency green-mode automatic reconfigurable switching converter with adaptively compensated single-bound hysteresis control," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2010, pp. 204–205.

[5] W. Kim, D. M. Brooks, and W. Gu-Yeon, "A fully-integrated 3-level DC/DC converter for nanosecond-scale DVS with fast shunt regulation," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2011, pp. 268–270.
[6] M. Putic, L. Di, B. H. Calhoun, and J. Lach, "Panoptic DVS: A fine-grained dynamic voltage scaling framework for energy scalable CMOS design," in *Proc. Int. Conf. Comput. Design (ICCD)*, 2009, pp. 491–497.
[7] V. Gutnik and A. P. Chandrakasan, "Embedded power supply for low-power DSP," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 5, no. 4, pp. 425–435, 1997.
[8] Y. Shakhsheer, S. Khanna, K. Craig, S. Arrabi, J. Lach, and B. H. Calhoun, "A 90 nm data flow processor demonstrating fine grained DVS for energy efficient operation from 0.25 V to 1.2 V," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 2011.
[9] S. Khanna *et al.*, "Stepped Supply Voltage Switching for energy constrained systems," in *Proc. Int. Symp. Quality Electron. Design*, 2011, pp. 1–6.
[10] H. Soeleman and K. Roy, "Ultra-low power digital subthreshold logic circuits," *Low Power Electronics and Design*, pp. 94–96, 1999.
[11] A. Wang, A. P. Chandrakasan, and S. Kosonocky, "Optimal supply and threshold scaling for subthreshold CMOS circuits," in *Proc. Comput. Soc. Annu. Symp. VLSI*, 2002, pp. 5–9.
[12] J. M. Ashouei *et al.*, "A voltage-scalable biomedical signal processor running ECG using 13 pJ/cycle at 1 MHz and 0.4 V," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2011, pp. 332–334.
[13] Y. Zhang *et al.*, "A batteryless 19 μW MICS/ISM-band energy harvesting body sensor node SoC for ExG applications," *IEEE J. Solid-State Circuits*, pp. 199–213, 2013.
[14] G. Ruhl, S. Dighe, S. Jain, S. Khare, and S. R. Vangal, "IA-32 processor with a wide-voltage-operating range in 32-nm CMOS," *IEEE Micro*, pp. 28–36, 2013.
[15] S. Lutkemeier, T. Jungeblut, H. K. O. Berge, S. Aunet, M. Porrmann, and U. Ruckert, "A 65 nm 32 b subthreshold processor with 9 T multi-Vt SRAM and adaptive supply voltage control," *IEEE J. Solid-State Circuits*, pp. 8–19, 2013.
[16] B. H. Calhoun and A. P. Chandrakasan, "A 256-kb 65-nm sub-threshold SRAM design for ultra-low-voltage operation," *IEEE J. Solid-State Circuits*, pp. 680–688, 2007.
[17] M. Qazi, M. E. Sinangil, and A. P. Chandrakasan, "Challenges and directions for low-voltage SRAM," *Design & Test of Computers*, pp. 32–43, 2011.
[18] S. N. Wooters, B. H. Calhoun, and T. N. Blalock, "An energy-efficient subthreshold level converter in 130-nm CMOS," *IEEE Trans. Circuits Syst. II: Expr. Briefs*, vol. 57, no. 4, pp. 290–294, 2010.
[19] B. H. Calhoun, J. Ryan, S. Khanna, M. Putic, and J. Lach, "Flexible circuits and architectures for ultralow power," *Proc. IEEE*, vol. 98, no. 2, pp. 267–282, Feb. 2010.

Kyle Craig received the B.S. degree in computer and electrical engineering from the University of Virginia, Charlottesville, VA, USA, in 2008. He is in his final year working towards his Ph.D. in electrical engineering and is co-advised by Ben Calhoun and John Lach.

He spent the summer of 2010 and spring of 2011 as an intern at AMD Research. His research interests are in low power circuit techniques and implementation, circuit design, and power grid distribution and integrity.

Yousef Shakhsheer received the B.S. degree in computer and electrical engineering from the University of Virginia, Charlottesville, in 2008. He is currently pursuing the Ph.D. degree at the University of Virginia.

Saad Arrabi received the B.S. degree in computer engineering from the University of Hartford, West Hartford, CT, in 2008, He is currently a Ph.D. candidate for a computer engineering degree at the University of Virginia, Charlottesville, VA. He is expected to graduate in 2014.

His research interests include low power computer architecture, configurable processor design, multi-core processor design, application-specific processor design, and low level digital design.

Sudhanshu Khanna received the B.Tech. degree in electronics and communication engineering from Delhi College of Engineering, India, in 2005, and the M.S. degree from University of Virginia, Charlottesville, in 2011. His thesis focused on low power SRAM design.

He is currently a Senior Design Engineer with Texas Instruments, Dallas, TX, USA. From 2005 to 2008, he was a Design Engineer with Texas Instruments, India, where he worked on IO circuit design.

John Lach (SM'05) received the B.S. degree in science, technology, and society from Stanford University in 1996 and the M.S. and Ph.D. degrees in electrical engineering from the University of California, Los Angeles in 1998 and 2000, respectively.

Since 2000, he has been a faculty member in the Charles L. Brown Department of Electrical and Computer Engineering at the University of Virginia, where he currently serves as Professor and Chair. His primary research interests include digital system design methodologies, body sensor networks (BSNs), and wireless and mobile health.

Dr. Lach is a former Associate Editor for the IEEE TRANSACTIONS ON COMPUTERS and the IEEE TRANSACTIONS ON COMPUTER AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS. He is a Co-Founder and Steering Committee member for the Wireless Health conference series and is a Co-Founder and Co-Director of the UVA Center for Wireless Health.

Benton H. Calhoun (S'02–M'05–SM'12) received the B.S. degree in electrical engineering from the University of Virginia, Charlottesville, in 2000. He received the M.S. degree and Ph.D. degree in electrical engineering from the Massachusetts Institute of Technology, Cambridge, MA, in 2002 and 2006, respectively.

In January 2006, he joined the faculty at the University of Virginia in the Electrical and Computer Engineering Department, where he is now an Associate Professor. He is a Campus Director and Technical Thrust Leader in the NSF Nanosystems Engineering Research Center for Advanced Self-Powered Systems of Integrated Sensors and Technologies (ASSIST). His research interests include body area sensor nodes (BSN), wireless sensing, low power digital circuit design, sub-threshold digital circuits, SRAM design for end-of-the-roadmap silicon, variation tolerant circuit design methodologies, and low energy electronics for medical applications. He is a co-author of "Sub-threshold Design for Ultra Low-Power Systems" (Springer, 2006), author of "Design Principles for Digital CMOS Integrated Circuit Design" (NTS Press, 2012), and has over 100 peer reviewed publications.