Intel's Near-Threshold Voltage Computing and Applications

September 18, 2012 by David Kanter

At ISSCC 2012, Intel showed off Near-Threshold Voltage (NTV) techniques that promise to significantly increase energy efficiency by decreasing the supply voltage. The most impressive demonstration was a full x86 microprocessor (Claremont) that consumed as little as 2mW and could run off a solar panel. Intel's research on NTV is particularly interesting, because it promises a path to continue scaling voltage and decreasing dynamic power.

Moore's Law has always been about scaling transistor density and count with advances in semiconductor process technology. As discussed in an earlier article from IEDM 2005, this is often conflated with higher performance and lower power consumption. Historically, the main driver for power consumption was reducing the operating voltage. The dynamic switching power of a chip is proportional to the square of voltage, so a small 10% voltage reduction translates into 19% lower active power.

For nearly 30 years the supply voltage scaled down, reaching 1V around 2005. At that point, voltage scaling slowed to nearly a stand still. The bandgap for silicon is about 1eV, meaning that it takes around 1V for silicon to switch between acting as an insulator and a conductor (where electrons move). Modern transistors use a variety of materials beyond silicon, but if the supply voltage is too far below 1V, transistors perform poorly and the vulnerability to errors increases.

Voltage has modestly decreased since 2005 and stands at around 0.8-0.9V, with some mobile chips operating as low as 0.65V (e.g. Sandy Bridge and Ivy Bridge). This was accomplished using careful design techniques that address the many problems of low voltage operation. One key challenge that was highlighted at ISSCC for 22nm and beyond is variability. Modern chips incorporate billions of transistors, and purely statistical effects dictate that some are slower or faster than others. Yet semiconductor manufacturers must produce large volumes of chips that function correctly over many years and a wide range of conditions. Dynamic operating conditions are a problem as well and become worse at low voltage. Large spikes or drops in current (known as dI/dt) can easily cause errors, similar in nature to brown-outs that hit overloaded power grids. Similarly, bit flips caused by alpha particles can corrupt data stored in circuits and large memory arrays.

The point where a transistor begins to turn 'on' and start conducting a little current is described as the threshold voltage, which is around 0.2-0.3V in modern process technology (although transistors do not reach the full saturating current till around 1V). The essence of Intel's NTV research is observing that the most energy efficient supply voltage is just slightly higher than the threshold. A supply voltage that is substantially higher than the threshold is a convenience, but one that increases frequency at the cost of power. The big challenge for NTV is variability and ensuring correct functionality and high yields.

The goal of NTV techniques is to enable extremely low supply voltages, by using circuits that are extremely robust; tolerating variability and resilient against errors. While this sounds quite challenging, it is eminently feasible. Consider the problem of dI/dt, where a sudden demand for current (e.g. a floating point unit that wakes up and starts executing) causes voltage to drop across the chip. Chips using NTV tend to use less power, meaning that the current consumed is much smaller. Moreover, NTV circuits at low voltage will run at lower frequencies, meaning that the supply has more time to stabilize before errors start to occur.

Modern chips are largely composed of digital logic and memory, along with portions of analog and I/O circuitry. At a high level, CPUs and GPUs resemble large islands of memory for storing data (caches) and logic for computation (cores). Even logic such as a floating point unit contains memory elements though, such as register files and latches to hold intermediate data values. But logic is largely dominated by combinatorial logic and wiring, rather than data storage elements.

Generally, memory is much more difficult to scale than logic. At lower operating voltages, structures like SRAM cells are less stable and become very vulnerable to errors reading or writing data. In contrast, the impact of low voltage on logic tends to be an increase in delay, which reduces frequency. While lower performance is undesirable, it is also far more tolerable than data corruption. To illustrate, mobile and server SoCs often have separate voltages for the CPU cores and the large caches. For example, Intel's Medfield operates the L2 cache at 1.05V, while the CPU core can dip down to 0.7V.

Intel's research on NTV relies on novel circuit techniques that enhance the robustness of both logic and memory. The results presented at ISSCC come from several papers. The first describes an entire 32nm Pentium core implemented using NTV. The second paper focuses on NTV techniques in a 22nm SIMD permute unit. A third paper on a 32nm variable precision floating point unit does not use NTV, but helps to understand potential applications. Based on our analysis of these papers, Near-Threshold Voltage computing techniques are most applicable to highly parallel workloads. Generally, NTV is an ideal fit for HPC workloads and works very well for graphics, but not general purpose CPUs.

Pages: 1 2 3 4 Next »

Discuss (89 comments)

RELATED ARTICLES

- ISSCC 2012 Preview
- Computational Efficiency for CPUs a 2012
- Silvermont, Intel's Low Power Archit
- Medfield, Intel's x86 Phone Chip
- AMD's Analyst Update

Intel's Near-Threshold Voltage Computing and Applications

September 18, 2012 by David Kanter

32nm CPU Logic

Generally speaking, the combinatorial logic that performs calculations inside of modern chips scales relatively well with voltage. Variability is a significant concern, but typically manifests as slower operation, rather than incorrect results. To address variability and preserve high performance, several changes in logics design techniques were necessary.

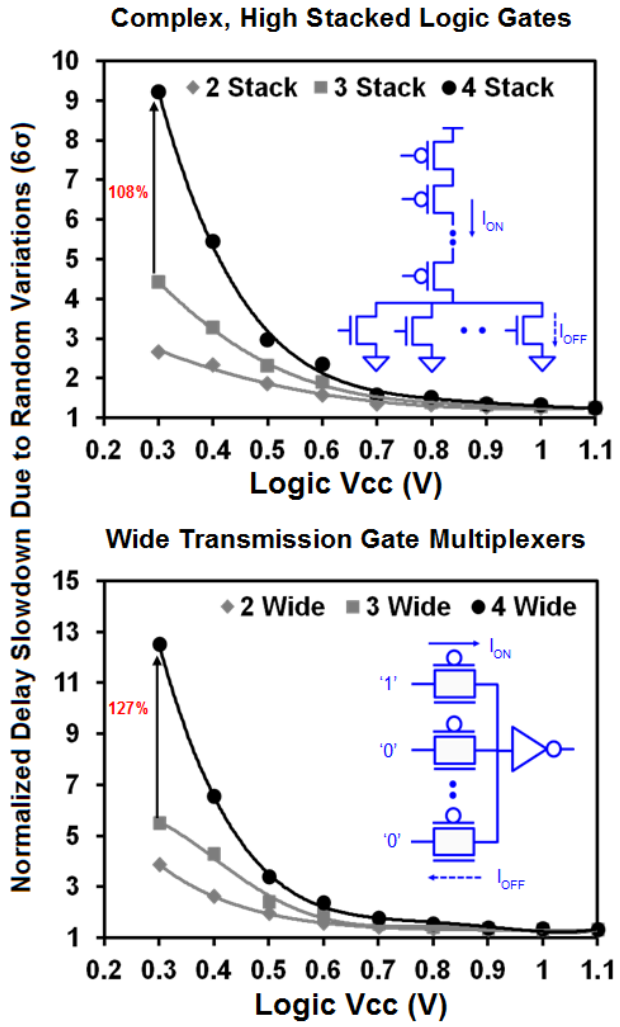

Figure 1. Delay for Highly Loaded Near-Threshold Circuits, courtesy Intel

The first set of changes focused on logical design and eliminating highly loaded logic circuits that are more vulnerable to variability. Intel's simulation results shown in Figure 1 indicate that large fan-in and fan-out configurations significantly worsen variability. Restricting the logic libraries to 3 output stacks and 3 inputs for multiplexers improved low voltage performance by cutting the delay in half. Of course, these design rules tend to increase the transistor count and area for functions that were previously implemented with more heavily loaded circuits; the area impact was estimated as 10%.

RELATED ARTICLES

- ISSCC 2012 Preview
- Computational Efficiency for CPUs a 2012
- Silvermont, Intel's Low Power Archit
- Medfield, Intel's x86 Phone Chip
- AMD's Analyst Update

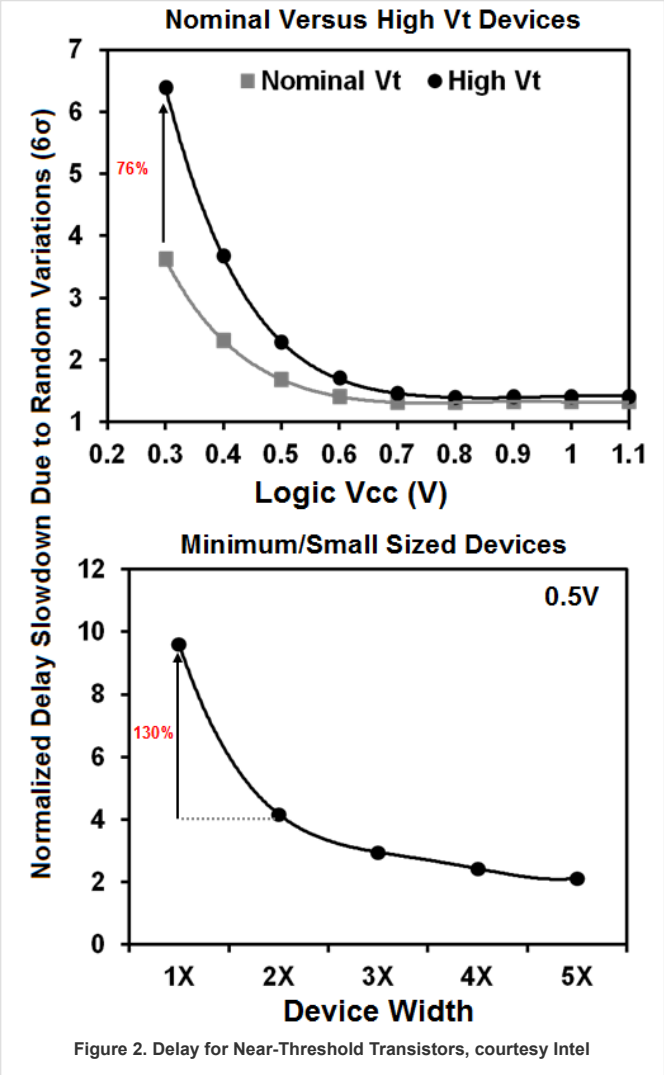

Figure 2. Delay for Near-Threshold Transistors, courtesy Intel

A second set of changes emphasized the importance of physical design, and dealt with variability at the transistor, rather than gate, level. The first set of simulation results in Figure 2 show the impact of device threshold on variation, with substantially worse problems for high V_t devices. Moving to nominal V_t substantially reduces delay and improves performance at low voltage, but will also increase leakage power. The second set of simulations show variability problems for minimum sized transistors when operating at 0.5V. Minimum sized devices increase delay by about a factor of two; so requiring larger devices significantly reduces variability issues. The downside is that larger transistors end up using more area, although this only impacts logic and the cost is around 5%.

32nm CPU Memory

In contrast to logic, memory elements like latches, registers, and SRAM are far more difficult at low voltages and are commonly the limiting factor in modern designs. Whereas logic tends to slow down and rarely produces incorrect results, memories are incredibly sensitive to voltage and are very likely to produce incorrect data. Conveniently, larger caches can easily be placed on a separate voltage plane from the logic in a CPU or GPU core. However, the logic circuits in a core, such as floating point units, still incorporate a variety of state elements such as latches and register files.

As many are familiar, Intel already uses 8T cells for low voltage SRAMs. This practice started with the L1 data cache on the 45nm Silverthorne and Nehalem. To operate at NTV, further efforts are necessary to reduce the read and write voltages. Intel's engineers use a 10T cell, which can write data at 0.25V lower than an 8T cell. The read circuits are programmable, and can shift the voltage to compensate for variation, ensuring robust reads at low voltage. The area impact is around 20%, compared to 8T cells.

The other significant type of memory are the flip-flops that hold data in logic circuits. Even a simple adding circuit requires a little storage to hold the input and output operands, before they can be safely stored in the registers or memory. Intel's team also described several techniques to improve the stability of flip-flops at low voltage. All together, the NTV flip-flops are about 35% larger than normal.

NTV Pentium Results

The 32nm NTV Pentium core is 2mm^2 and uses 6 million transistors. The processor is split into two voltage domains, one for logic, and another for caches. The caches can safely operate down to 0.55V, while the logic can reach 0.28V; the maximum voltage is 1.2V. The entire chip is synthesized for 0.5V operation, to ensure good clock frequencies at low voltage. Targeting a high voltage would substantially reduce near-threshold performance, while only saving a little leakage. Additionally, the floating point unit is power gated based on instruction level activity, with a single cycle wake-up.

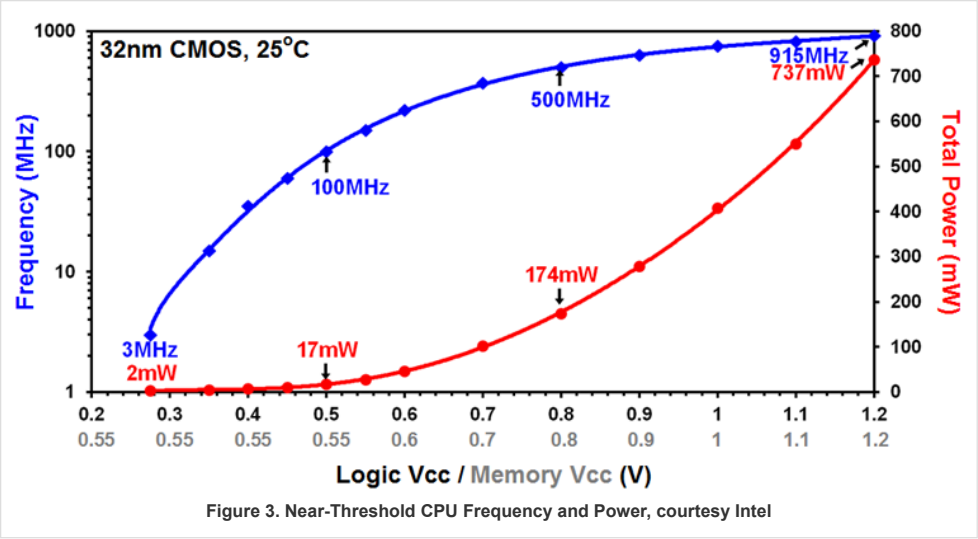

Figure 3. Near-Threshold CPU Frequency and Power, courtesy Intel

Figure 3 shows measured frequency and power as a function of logic and cache voltage, scaling from 2mW at 3MHz up to 737mW at 915MHz. At the lowest voltage levels and 3MHz, the power consumption is almost entirely leakage from the caches (62%). At the near-threshold voltage levels and 80MHz, the logic activity is 53%, while leakage is 42%. At the maximum voltage, power is almost entirely due to logic activity (81%). The most optimal operating point is 0.45V for logic and 0.55V for memory, achieving 4.7× energy savings.

Pages: « Prev 1 2 3 4 Next »

Discuss (89 comments)

About Us Contact Us Reprints Advertise at RWT Write for Us Privacy Policy Forum Guidelines Membership TOS

Intel's Near-Threshold Voltage Computing and Applications

September 18, 2012 by David Kanter

22nm NTV SIMD Permute

A second paper from Intel Labs described a 22nm 256-bit SIMD permute unit designed for NTV. The vector unit operates at 0.28-1.1V and runs at 17MHz to 2.5GHz. The key components are a large 32-entry, 256-bit wide register file and a crossbar for moving data. The register file has 3 read and 1 write ports and can perform vertical shuffles across multiple entries, bypassing the crossbar entirely. The crossbar is capable of byte-wise any-to-any permutes, which matches and exceeds the requirements for Intel's SSE and AVX instructions. As to be expected, the most significant adaptations were needed for memory, rather than the logic. The performance is primarily limited by the register file, rather than the crossbar logic.

The team from Intel highlighted three techniques which collectively reduced the voltage for logic by 0.15V. The most significant was enhancing multiple flip-flops with shared circuitry to tolerate variation. Special level shifters enable the crossbar logic to efficiently operate at a different voltage from the flip-flops. The last technique added shared redundant transistors to several circuits, which counteracts variation.

The register file in the permute unit was heavily modified to run at NTV, with attention paid to both reading and writing to the register file cells. The register file read path was converted from dynamic to static circuits, which netted a substantial 0.2V saving. This is a huge change in design; nearly all register files and SRAMs use dynamic circuits. However, dynamic circuits do not function well at low frequencies. In a way, this mirrors Intel's choice to convert all the datapath logic in Nehalem to static circuits and a total reversal from the dynamic circuit techniques used in the Pentium 4.

To reduce the voltage for writes as well, two techniques were used in conjunction. The first is changing the write circuits to a topology with redundant transistors, which is substantially more robust. The write circuits were further enhanced by stabilizing the voltage into the register file cells. Overall the write voltage for the register file decreased by 0.25V.

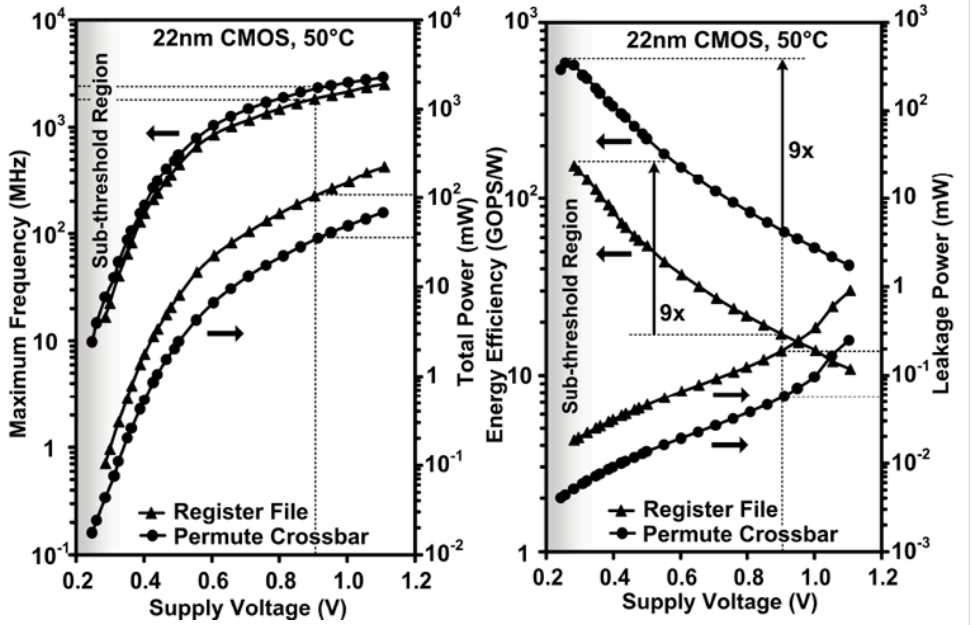

Figure 4. Near-Threshold SIMD Permute Frequency, Power, and Energy, courtesy Intel

The measured results show significant gains in energy efficiency at a nominal 0.9V and 1.8Ghz due to a highly efficient architecture. The register file performs basic blend operations in a single cycle, reducing energy by 66%. A more complex 64-bit 4×4 matrix transpose drops from 12 to 7 cycles and reduces energy by 53%. Moving data from an array-of-structs to a struct-of-arrays showed more modest benefits, taking 6 cycles instead of 8 and uses 40% less power. Additionally, the energy gains from near-threshold operation are nearly an order of magnitude compared to nominal voltages, as shown in Figure 4.

32nm Variable Precision FPU

Intel presented another paper that was not directly related to NTV, but is indirectly quite interesting because it hints at the most likely applications. The paper describes a 32nm variable precision floating point unit that executes fused multiply-adds every cycle, with three cycles of latency.

RELATED ARTICLES

- ISSCC 2012 Preview
- Computational Efficiency for CPUs a 2012
- Silvermont, Intel's Low Power Archit
- Medfield, Intel's x86 Phone Chip
- AMD's Analyst Update

The FPU can operate in three different modes for varying performance, energy efficiency, and accuracy. All modes use a normal 8-bit exponent, but vary the size of the mantissa based on the necessary precision. The first is a standard IEEE-754 single precision mode that executes a single FMA per clock with a 24-bit mantissa. The second executes two FMAs with two 12-bit mantissas and the most efficient mode has four 6-bit mantissas. The floating point unit has new certainty tracking circuits, that determines when greater accuracy is necessary for a correct calculation. The certainty tracking and extra exponent calculations increase power by roughly 21% compared to single precision.

As shown in Figure 5, the energy efficiency increases by about 1.5× for two-wide execution and over 3× for four-wide operations. The efficiency also depends on the operating conditions, with about 7× greater savings at low voltage (although not quite near-threshold levels).

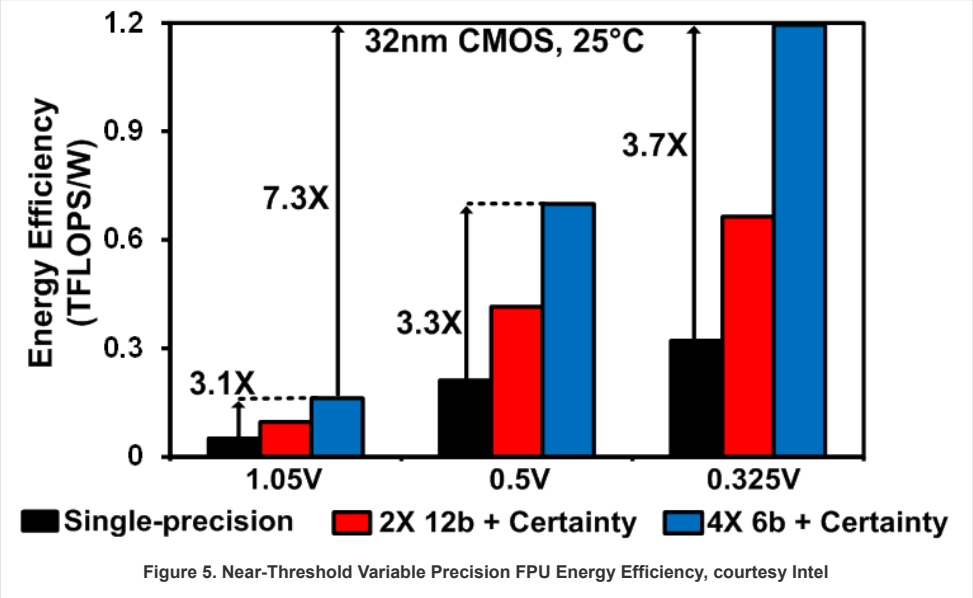

Figure 5. Near-Threshold Variable Precision FPU Energy Efficiency, courtesy Intel

To improve performance and energy efficiency, the FPU speculatively computes results using reduced precision. The initial attempt will calculate four operations with 6-bit mantissas. If more accuracy is required, then the second attempt will use two 12-bit mantissas, before finally using standard single precision.

The total energy savings strongly depend on the data involved. Successfully using 6-bit mantissas will save considerable power. Calculations that must be recomputed with 12-bit mantissas still save about 7% energy compared to single precision. However, operations that only succeed in single precision will use roughly twice as much energy due to the retries. In all likelihood, a real implementation would need more intelligent speculation, to avoid wasting power. Additionally, the variable precision FPU did not address double precision (i.e. 53-bit mantissa), but it is easy to imagine extending the technique.

Intel's Near-Threshold Voltage Computing and Applications

September 18, 2012 by David Kanter

Near-Threshold Voltage Analysis

Individually, each of the three papers show intriguing results for improving energy efficiency. Beyond academic curiosity though, the real question is: Where will near-threshold voltage will be adopted in real products? Looking at all three papers collectively, there are hints and suggestions where near-threshold voltage is a good fit.

The NTV results present a consistent set of trade-offs. Compared to conventional designs, NTV techniques enable dramatically lower voltages and substantially better energy efficiency. At near-threshold voltages, the efficiency improves by around 4-7×. In the modern power constrained world, using the available power much more effectively translated into substantially increasing performance.

However, there are significant costs. NTV circuits use substantially more area and transistors than conventional designs. The 0.6µm Pentium used 3.3 million transistors. In contrast, the 32nm NTV Pentium was implemented with 6 million transistors. The 80% increase in transistor count was largely driven by NTV circuit techniques and it is safe to assume that the area would increase as well (relative to a 'normal' implementation of the Pentium core in 32nm).

Moreover, the optimal operating point for NTV designs tends to be quite low. The 32nm Pentium core increased efficiency by about 5×, by running at slightly under 100MHz. The maximum frequency was 915MHz, so the absolute performance decreased by about an order of magnitude. That is a tremendous sacrifice to achieve energy efficiency; one that may not be feasible for many applications.

As an additional observation, the 22nm NTV vector permute unit and the 32nm variable precision FPU papers were partially funded by a grant from the US government. This suggests that the two techniques are aimed at a single larger goal. Presumably, this grant is related to research programs on energy efficient computing, or perhaps the Exascale program.

The variable precision FPU also has a particular set of trade-offs. Speculatively reducing precision can save substantial energy by improving throughput, while keeping power roughly constant. However, it is unlikely to be quite as useful in the context of software that relies on scalar computations. In that case, reducing precision should modestly improve power. So the ideal workload would be amenable to vectorization.

Moreover, retrying uncertain calculations introduces an element of unpredictability. In the best case, the FPU can perform 4 FMAs/cycle. However, calculations requiring single precision would proceed at a single FMA/cycle, while wasting at least 1 cycle on the initial attempt. So the latency for 4 FMAs could vary from 3 cycles to 13, and perhaps more depending on the implementation and uncertainty detection. On an in-order processor core, this could potentially stall the pipeline while recalculating at higher precision. It would also significantly complicate code scheduling for compilers and programmers. However, an out-of-order processor core can easily inject the retry operations into the pipeline without stalling other instructions, minimizing the impact.

Future Near-Threshold Voltage Applications

The trade-offs associated with near-threshold voltage techniques strongly suggest certain applications. General purpose CPUs for client systems are unlikely to benefit from NTV. While energy efficiency is important, sacrificing frequency is inconsistent with the overall design targets. Client workloads are very bursty, with a small number of programs executing at any one time. The responsiveness of the system is largely determined by latency, and performance does not always improve with throughput. Modern CPUs run at high frequencies (2-4GHz) because software is inherently unpredictable. Increasing frequency improves performance for nearly all workloads, including those with many branches and unpredictable memory references. Sacrificing frequency for higher throughput would simply decrease performance in many cases. Moreover, client systems are very cost sensitive and the extra area is simply not worth the minimal benefits.

General server processors are in a very similar situation, because many applications need high single threaded performance. However, massively parallel workloads, like those targeted by low power 'cloud servers' (e.g. SeaMicro, Calxeda), would be a reasonable fit. Massive server farms are often power constrained, and if single threaded performance is not critical, then the gain in energy efficiency is likely to be worthwhile. Additionally, server CPUs are expensive and high margin, so it is reasonable to spend die area to increase energy efficiency.

The strongest matches for NTV are graphics-like workloads and high performance computing (HPC). Graphics processors (as well as image processors and DSPs commonly found in tablets and smartphones) are inherently throughput focused, with minimal emphasis on each individual thread. Modern GPUs run at fairly low frequencies, from roughly 200MHz for mobile graphics to 1.5GHz for high-end solutions. Nearly all GPUs are power limited; so improving efficiency would directly translate into performance. Graphics workloads are inherently floating point heavy and easy to vectorize, with a minimal emphasis on accuracy. The variable precision FPU would be a good fit here as well; errors caused by reduced precision may not even be perceptible to consumers. Die area is still constrained though, as GPUs are primarily useful for client systems and cost is a major factor. However, smartphones and tablets are so energy sensitive that the costs could be outweighed by greater battery life.

The best fit is the world of HPC, which combines the favorable qualities of the graphics and server market. As the Exascale project has highlighted, power is the most significant constraint for HPC systems. Like GPUs, workloads are very parallel and performance is largely measured in terms of throughput. Reducing frequency to improve efficiency is an excellent choice. Accuracy is crucial for HPC, and a variable precision FPU fits well. However, a more advanced design would be needed, since HPC has significant double precision requirements. Unlike

RELATED ARTICLES

- ISSCC 2012 Preview
- Computational Efficiency for CPUs a 2012
- Silvermont, Intel's Low Power Archit
- Medfield, Intel's x86 Phone Chip
- AMD's Analyst Update

client GPUs, HPC accelerators are fairly expensive. Each accelerator can typically perform the work of 2-4 server CPUs and costs several thousand dollars, while using 400mm^2 or more. There is no question that a higher performance design would be able to sustain higher prices and pay for the additional die area. Perhaps most telling, US government grants typically focus on areas of national interest. Graphics simply is not vital to the country, whereas HPC is a critical tool for the Departments of Defense, Energy, and any number of intelligence agencies.

In summary, Intel's work on near-threshold voltage is a novel approach that demonstrates substantially greater throughput and energy efficiency, at the cost of frequency and die area. The nature of NTV is uniquely suited to HPC workloads, which will significantly benefit from the energy advantages and can afford the associated costs. It is likely that the first commercial implementations will appear in Intel's HPC-oriented products, such as Knights Landing or successor projects. Other potential targets include Intel's integrated GPUs for PCs, tablets and smartphones as well as SoC components such as DSPs and image processors.