SAND2016-3166C

Millivolt Switches Will Support Better Energy-Reliability Tradeoffs

Erik DeBenedictis[1], Hans Zima[2]
[1]Center for Computing Research, Sandia National Laboratories, Albuquerque, NM, USA, epdeben@sandia.gov
[2]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA, hpzima@gmail.com

Millivolt switches will not only improve energy efficiency, but will enable a new capability to manage the energy-reliability tradeoff. By effectively utilizing this system-level capability, it may be possible to obtain one or two additional generations of scaling beyond current projections. Millivolt switches will enable further energy scaling, a process that is expected to continue until the technology encounters thermal noise errors [Theis 10]. If thermal noise errors can be accommodated at higher levels through a new form of error correction, it may be possible to scale about 3× lower in system energy than is currently projected. A general solution to errors would also address long standing problems with Cosmic Ray strikes, weak and aging parts, some cyber security vulnerabilities, etc.

As illustrated in Fig 1, integrated circuits have been experiencing decreasing dynamic power (green) but the device in use today (MOSFET) has a rapidly rising leakage current as supply voltage decreases and resulting rise in static power (orange). As a result, continued dimensional scaling accompanied by traditional voltage reduction would actually result in a rise in power (red). This makes further dimensional scaling unusable for logic. However, there are various approaches to addressing leakage current [Yablonovitch 11] that are a focus of E3S, with success of at least one of the options being considered inevitable. Should such a device enter production, scaling along the green curve could continue. This scaling will continue until some other effect is encountered.

The expectation is that the next effect will be thermal noise [Theis 10]. Signaling within a chip is governed by a slight variant of the Shannon-Hartley theorem [Shannon 48], which was originally formulated for communications links. The Shannon-Hartley theorem was intended to account for only the energy in a transmitted signal whereas an application to computers should consider the total energy drawn from the power source. We therefore adopt the formula $p_{error} = \exp(-e_{signal} / (F\mathrm{kT}))$, where p_{error} is the probability of error, e_{signal} is signal energy, and F is in the role of the noise factor of an analog circuit (radio). However, we expand the scope of F to include any other factors that scale the energy. Typically, this is (a) the fraction of energy actually delivered to the transistor(s) on the next gate as opposed to being CV^2 energy in the wire, (b) the fanout factor, and (c) average logical masking of signals.

This leads to an enhanced computational model for millivolt switch-based circuits. The idea is that a circuit will have a power supply terminal to which a changeable voltage V is applied. By varying V either statically or under programmatic control, $e_{signal} = CV^2$ can be controlled and consequently p_{error}. The presence of a specific functional relationship between energy and error rate allows quantitative reasoning about the effectiveness of error detection and correction. For example, Fig 2 shows two copies of an embedded circuit f (using color as an additional subscript) comprising N gates with no error detection. The probability of error is shown as p_{error}, given that voltage has been adjusted so signals have energy e_s. Fig 2 also shows the composite circuit g containing two copies of f with a comparator to detect errors if the two circuits give different answers. However, faults in the same gates in both circuits would present the comparator with two identical but incorrect results and the errors would be undetected with probability p_u. However, $p_u \geq p_{error}^2 / N$ (noting that the inequality is due to other error combinations that do not effect the reasoning in this brief abstract). While the probability of an undetected error declines, the circuit with error detection has twice the gates and consumes twice the energy. If the voltage is adjusted so overall energy is the same ($e_s = e_s/2$), the equations in Fig. 2 show that the probability of undetected error is the same as well.

We view the above result as a logic analogy of Shannon's channel capacity theorem [Shannon 48]. According to the channel capacity theorem in communications, it is impossible to use error correction to get more data through a channel than its capacity. Likewise, we show (by the single example in Fig 2) that error correction by duplication and comparing output cannot reduce the energy required to get a result to a given probability of correctness. From the equations in Fig 2, this result depends on the specific functional relationship between reliability and energy.

However, there are benefits to duplication when the energy-reliability tradeoff is different. While the detection by duplication is at an absolutely flat spot in terms of the energy-reliability tradeoff, circuit *g* in Fig 2 will catch non-energy errors such as Cosmic Ray strikes and the other errors listed in the first paragraph.

It is possible to use error correction to compute at lower energy, with Fig 2 merely showing an example of an error correction circuit exactly at the boundary between helping and hurting energy consumption. To reduce energy, checking a calculation needs to be more efficient than performing the calculation in the first place.

Redundant Residue Number Systems (RRNS) form an existence proof that circuits exist that have this property. RRNS has been proposed as the basis of general purpose computing in the past [Watson 66]. A tutorial on RRNS is not possible given space limitations and the reader is referred to [Watson 66] – but we will include a few sentences here. The idea is to represent a ranged whole number not by its binary representation but by remainders when divided by a series of relatively prime moduli (such as 199, 233, 194, 239). When represented this way, addition, subtraction, and multiplication can be performed independently on the remainders – and the original integer can be reconstructed. By including additional moduli (such as 251 and 509), one or more errors in remainders can be detected and corrected. Significantly, this also detects two errors and corrects one error in an arithmetic function!

The example in Fig 3 shows two types of advantage from RRNS. A 31-bit integer multiply requires a 31×31 array of gated full adders, whose circuit complexity and power consumption would be proportional to the large yellow rectangle on Fig 3A. However, the RRNS in Fig 3B represents a slightly larger number range by four remainders of 8 bits each. The aggregate area of the yellow multipliers is less by nearly a factor of four. This well-known complexity advantage is independent of a new energy advantage. The new advantage is due to the check of four remainders (in yellow) requiring two additional ones (in green – although one is 9 bits instead of 8). This meets the condition that the check be less resource intensive than the original calculation. Fig. 3B is an example where adding the green check remainders to the baseline yellow remainders would cut energy at constant reliability.

Software support will be required to realize an energy efficiency gain of this type. The authors have developed an "assertion language" that can be added to C, C++, Fortran, etc. and augments the standard language with constructs that would allow a compiler, run time system, or programmer to institute error checking and correction in such as way as to meet overall reliability goals.

For example, the line of code below defines the number system in Fig 3, but includes assertion language annotations in bold face that check the consistency of the remainders (function *ED*) and correct them (function *EC*) in case of error. The idea is that these assertions could be added to existing code without rewrite. The compiler or run time system would choose when to validate the assertions, balancing the resources required to validate an assertion even when an error is unlikely against the risk that multiple errors would accumulate and become uncorrectable.

struct RRN { int r1:8, r2:8, r3:8, r4:8, r5:8, r6:9; } **assert(*ED*(...)) error(*EC*(...))**;

Furthermore, the example below of multiplication using the residue system in Fig 3 shows how user-generated or system generated annotations would inform the system of the probability of undetected error (**p_u**), detected error (**p_d**), and energy consumption (**E**) for the function. Each of these annotations is an expression in *V*, or voltage. The run time system or compiler would additionally perform an optimization to find the value of *V* that meets a user-specified reliability requirement at the top level (i. e. **p_u** for the main program) for minimum energy. The computer would then activate system calls to control the voltage applied to the hardware as the program runs.

struct RRN mul (RRN a, RRN b) **voltage_options{ V, p_u($f_1(V)$), p_d($f_2(V)$), E($f_3(V)$)}** {
    return RRN (a.r1*b.r1%199, a.r2*b.r2%233, a.r3*b.r3%194, a.r4*b.r4%239, a.r5*b.r5%251, a.r6*b.r6%509); }

In conclusion, millivolt switches may be able to convert computer reliability from an annoying problem that can be "swept under the rug" into a key enabler for additional generations of scaling. Since the introduction of the integrated circuit, reliability of parts has been high enough that it could be dealt with through *ad hoc* techniques (like error correction for memory) and inefficient recovery methods (like checkpoint-restart). As the largest computers continue to grow to Exascale, the scalability of the existing methods is being tested and coming up short. While millivolt switches are not currently available, it can be projected theoretically that realizing their maximum potential will require a more systematic solution to reliability issues to address thermal noise. However, the argument just presented has transformed reliability from a problem into an enabler for increased power efficiency. This transformation is of more than scientific interest, as funding tends to be more accessible to projects that realize upside potential than projects that fix problems.

The class of solution proposed here is to make the entire technology stack aware of the energy-reliability tradeoff, including devices, circuits, architecture, and software. This would appear to be a capability that has not been important to consider before, as devices less "advanced" than the millivolt switch do not exhibit the necessary energy-reliability tradeoff because they essentially waste energy.

We can make a rough projection of the potential gain due to the error correction described in this paper. Without any error detection or correction, an Exascale supercomputer (10^{18} ops/sec) would execute about e^{71} gate operations over a three-year lifetime. A simple spreadsheet probability calculation (not included here) suggests that the RRNS in [Watson 66] could operate at about $E_{signal} = 25$ kT, albeit with 50% overhead. This suggest about a 2× net increase in energy efficiency.

Acknowledgment: Thanks to Thomas Theis for discussions, notably leading to Fig. 2. Unclassified, approved for public release, tracking number 321060. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References:

[Theis 10] Theis, Thomas N., and Paul M. Solomon. "In Quest of the" Next Switch": Prospects for Greatly Reduced Power Dissipation in a Successor to the Silicon Field-Effect Transistor." Proceedings of the IEEE 98.12 (2010): 2005-2014..

[Yablonovitch 11] Yablonovitch, Eli. Replacing the transistor: searching for the milli-volt switch. Hong Kong University of Science and Technology, 2011.

[Shannon 48] "A mathematical Theory of Communication", BSTJ. Vol. 27, 1948, pp379-423.

[Watson 66] Watson, Richard W., and Charles W. Hastings. "Self-checked computation using residue arithmetic." Proceedings of the IEEE 54.12 (1966): 1920-1931.

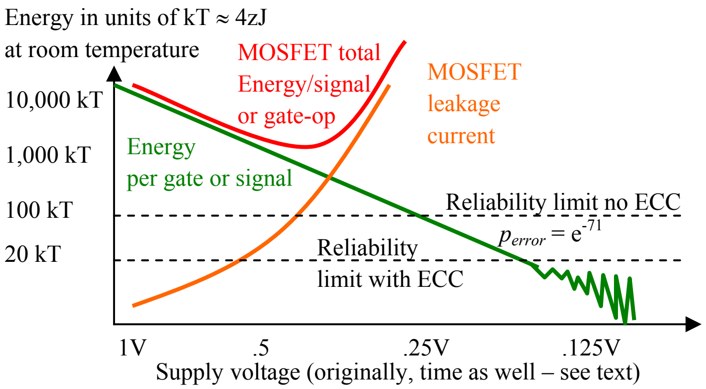

Fig. 1. Energy scaling to reliability limits

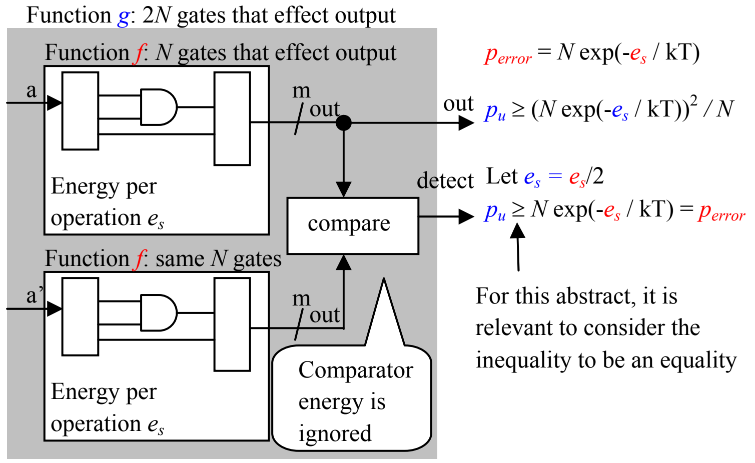
Function g: 2N gates that effect output
Function f: N gates that effect output
a
m
out
out
Energy per
operation e_s
compare
detect
Function f: same N gates
a'
m
out
Energy per
operation e_s
Comparator
energy is
ignored
$p_{error} = N \exp(-e_s / \text{kT})$
$p_u \geq (N \exp(-e_s / \text{kT}))^2 / N$
Let $e_s = e_s/2$
$p_u \geq N \exp(-e_s / \text{kT}) = p_{error}$
For this abstract, it is
relevant to consider the
inequality to be an equality

Fig. 2. Logic error correction enabled by millivolt switches

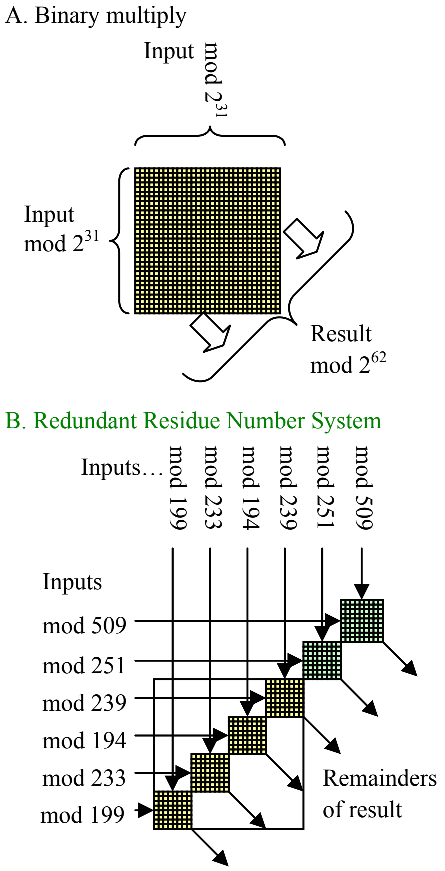

Fig. 3. Error correction to reduce energy