Stochastic ADC using Standard Cells

Design, Implementation and Eventual Fabrication of a 4.7-bit ADC

Author: Zachary Baltzer

***Abstract*—As process nodes shrink, analog design increasingly becomes difficult due to space, signal, and noise concerns. With highly synthesized digital design, analog design innovation lags as these specific considerations are to be accounted for. The analog to digital converter, proposed by Weaver et al., is a completely digital design relying on comparator offsets to produce a digital counter that tracks the difference between the input voltage and a reference voltage. To soon be fabricated on GlobalFoundry's 130 nm CMOS process, the proposed 5-bit ADC uses approximately 90,000 transistors with 1,500 comparators and a full-adder tree consisting of 1,500 adders to produce a digital output with a reference voltage of 500 mV and a range of ± 50 mV.**

I. Introduction

The ADCs of today fall into two groups of schematic design, single-clock conversion and multi-clock conversion. The most recognizable single-clock conversion type is the flash ADC which uses a resistor ladder to divide a reference voltage into 2^n divisions, where n is the number of bits. A bank of comparators each has its own reference voltage, depending on how far down the resistor tree it sits, and compares that to the input voltage. Temperature error correction and bubble error logic is needed after the comparator digitizing to ensure an accurate output. While the design is extremely fast, which is where the name flash comes from, the ability to produce thousands of resistors and comparators within tolerance is difficult. A popular multi-clock conversion ADC is the successive approximation register (SAR) ADC, which iterates a digital-to-analog converter (DAC) to match its digital code to the input voltage to the best of its ability. [1]The DAC inside typically includes an extensive R-2R resistor bank, or a large capacitor bank to track and hold the analog output.

Saxena et al. [3] demonstrated that as devices shrink, variation in their characteristics increase and gives explanation for these variations. These include OPC errors giving poly gates variation in width and shape, implant scattering giving junctions differing properties, and varying gate counter-doping giving channels different current characteristics. This causes the error correction logic on analog designs to become more complex, or simply impossible if variation is too large.

As features on an integrated circuit become smaller, design rules become more stringent due to lithographic and etch tolerances becoming ever tighter. Modern nodes may have over 1,000 design rules governing a design. This causes manual designs to become increasingly difficult for non-synthesized work. For many IC generations, digitally synthesized design has been the fastest and most cost-effective way to produce ICs due to these complex rules. Analog design, however, fails to benefit due to largely being a custom process and being unable to synthesize easily coupled with design variation discussed earlier.

The lack of synthesizable analog design along with increasing variation in ever-shrinking nodes proves to be an issue as digital design continues to move forward in these fields, leaving analog design behind. The Stochastic ADC proposed works on the principle of non-ideal devices in fabrication, creating a device that can scale to the latest digital design nodes and all of the benefits that come with it, trading for die area in the process.

II. Theory

Device mismatches will cause transistors to have differing current drive at a given gate voltage, which will be observed as differing threshold voltages. As more devices are fabricated, these threshold voltages will have a mean, μ, and a standard deviation, σ, and is assumed to follow a gaussian distribution. The stochastic ADC operates best when σ is as large as possible, while not causing errors in digital operation. The digital clocked comparator, at the heart of the stochastic ADC, operates on two competing transistor branches, whichever can drain charge faster is locked at ground, and a positive feedback loop causes the opposing branch to lock at logic high. Due to these mismatches, it is possible that a branch with a lower input voltage than its competing branch to have a higher drive current, setting the comparator to a digital code that would have otherwise been the result of the branch having a higher input voltage than the competing branch, giving an incorrect result. Put another way, most comparators will fire at the appropriate voltage, some will fire at a voltage lower than expected, and some will fire at a voltage higher than expected.

For example, if the reference voltage is 1 V, and measured σ for comparator offset is 100 mV across 1000 comparators, following basic gaussian distribution theory, 2% of the comparators will trip with an input voltage of 800 mV, 15.9% will trip with an input voltage of 900 mV, and 97.7% of comparators will trip with an input voltage of 1200 mV. When these comparators are counted in a Wallace adder tree, the digital count output directly relates to the input voltage. Figure 1a shows the probability density of comparator offsets, Figure 1b shows a generalized schematic design of the stochastic ADC, and Figure 1c shows the cumulative output of the summation function for the 1024 comparators shown in Figure 1b. The design is very similar to the flash ADC, however does not need error correction logic to operate correctly. Due to the random nature of operation, however, a large number of comparators are needed to create a reliable output. A flash ADC requires 2^n comparators to operate, while a stochastic ADC requires $2*4^n$ comparators, where n is number of bits.

Zachary Baltzer is a student at Rochester Institute of Technology in the Department of Microelectronic Engineering.

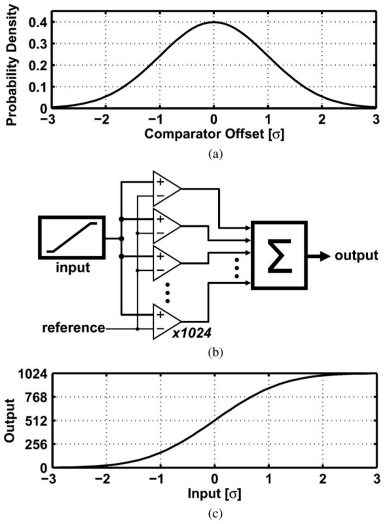


Figure 1. Stochastic ADC theory of operation. (a) Comparator offsets will follow a probability density function, (b) A reference voltage is shared amongst multiple comparators, as well as the input voltage, (c) the output of comparators in (b) as a function of input voltage. [1]

The stochastic ADC output has the issue of having a non-linear response. To linearize the output, computation logic is needed after the output to be useable by a control device using its output as a measurement tool. Figure 2 shows, graphically, how best to operate such logic on a gaussian function.

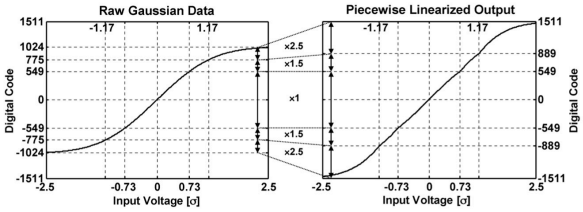


Figure 2. Gaussian linearization graphical representation, typically the function is best fit linearly between ± 0.73 σ. With added logic, this range can be extended to ± 1.17 σ, a 60.3 % increase in useable range. [2]

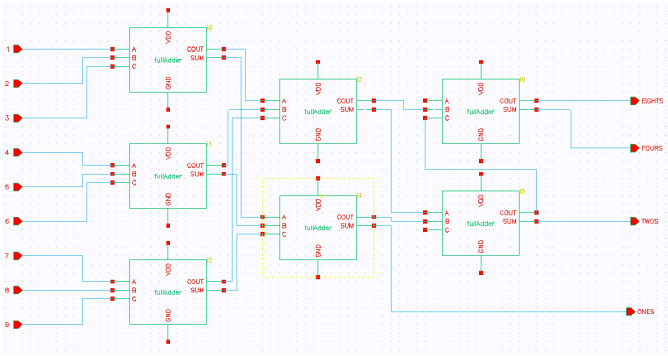
The function described in Figure 2 does not need to be implemented on die if properly described in its related data sheet and implemented in a microcontroller. Figure 2 implements a design using 2,048 comparators, for this specific design, Equation 1 describes the transfer function necessary to linearize the function described in Figure 2.

$$f(v) = \begin{cases} 2.5v - 0.512 & : \quad 1.166\sigma < \quad v \\ 1.5v - 0.134 & : \quad 0.732\sigma < \quad v \quad \leq 1.166\sigma \\ v & : -0.732\sigma \leq \quad v \quad \leq 0.732\sigma \\ 1.5v + 0.134 & : -1.166\sigma \leq \quad v \quad < -0.732\sigma \\ 2.5v + 0.512 & : \quad v \quad < -1.166\sigma \end{cases}$$

Equation 1. Transfer function *f(v)* relating the σ of the input voltage to the output for a stochastic ADC with 2,048 comparators. [2]

Where *v* is the input voltage, in relation to the offset standard deviation. and *f(v)* is the piecewise linearized output.

A Wallace tree adder is used as the summation stage for the comparators to compile a digitized output. Figure 3 shows a 9-input Wallace tree adder, if any one of the inputs is a logic



high, only the ONES output will be a logic high, if 7 inputs are a logic high, the FOURS, TWOS, and ONES output will be a logic high. At large bit counts, this method of counting creates significant propagation delay and can become the limiting factor for timing within the circuit.

Figure 3. Wallace tree adder for 9 inputs, this system digitizes the amount of logic highs on the left to a binary encoding system on the right side.

Another novel aspect is that of the synthesizable nature of this device. Typically, digital clocked comparators use a non-standard design as shown in Figure 4. When observing in the light of standard logic gates, the clocked comparator is 2 2-input NAND gates with a shared PMOS and shared NMOS transistor. If these are split into each side, to no longer be shared, 2 3-input NAND gates are realized. The clocked-comparator using standard digital logic gates is shown in Figure 5.

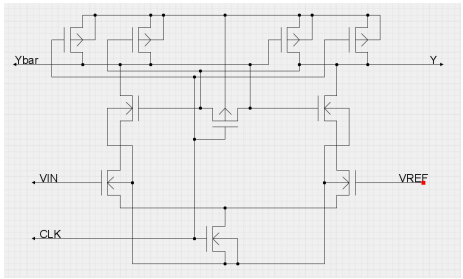


Figure 4. Clocked comparator comprising of 2 2-input NAND gates and a shared NMOS and PMOS transistor.

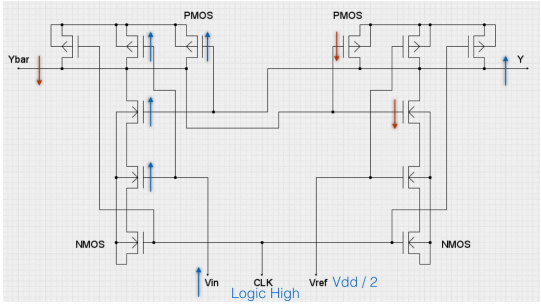


Figure 5. Clocked comparator comprising of 2 3-input NAND gates, this design is superior over Figure 4 due to its completely synthesizable design.

As described earlier, when the branch of NMOS gates connected to the input voltage has a higher current drive then the reference voltage branch, a positive feedback loop creates the comparator output. When the clock goes to logic low, both branches are set to VDD. To sync the circuit to the clock, a nor-latch is used. The full logic-gate schematic is shown in Figure 6.

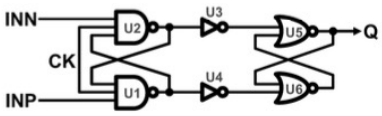
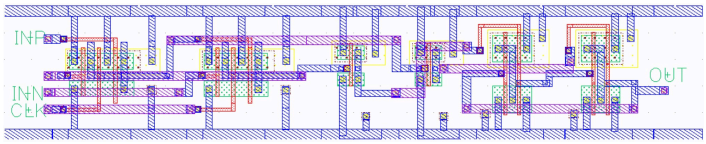


Figure 6. Logic level schematic of the synthesizable clocked comparator using standard logic gates.

III. DESIGN

With the intent of fabricating the device in the RIT SMFL, design was first done using TSMC 0.35 µm CMOS design using the Mentor Graphics design suite. The TSMC rule set was used because it had a minimum drawn feature size of 0.5 µm, similar to the SMFL, and being freely available inside the Mentor Graphics suite. At this size, the resulting area for a 5-bit ADC would exceed the specifications for a mask to be used in the ASML housed in the SMFL. Even if the allowable design area was large enough for the design time for an IC along with a full CMOS run inside the SMFL would not be possible.

Design then transitioned to contracting out the design work, and the GlobalFoundries' 130 nm CMOS PDK was acquired through MOSIS, a service done by the University of Southern California funded through the National Science Foundation to provide custom ICs for university coursework and research. The minimum drawn feature is 65 nm, providing a 9x reduction in feature size, or a 81x reduction in die area for a



given design. The PDK is designed for the Cadence design suite. Figure 7 shows the clocked comparator with latch, drawn in the GlobalFoundries PDF.

Figure 7. Clocked comparator with latch circuitry designed with λ = 65 nm

Where λ is the minimum drawn feature size. The total drawn size of Figure 7 is 26 µm by 5 µm. Figure 8 shows the same schematic, but drawn in Mentor Graphics. Due to rule differences, and the standard cell library differing, Figure 8 has a total drawn size of 200 µm by 100 µm, almost 80 times larger in area.

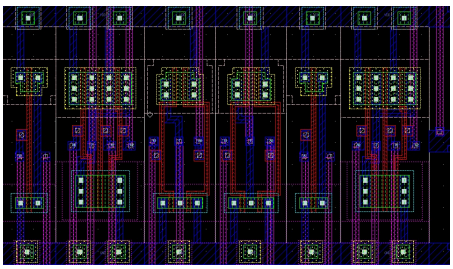


Figure 8. Clocked comparator design with latch circuitry design with λ = 500 nm

As shown in Figure 3, a Wallace tree adder is used in conjunction with comparators to produce a readable output. A compute cell is then produced using 9 comparators and 7 full adders for 16 devices. The full adder and comparator was designed specifically to have the same dimension, to make layout more efficient. The compute cell is shown in Figure 9 an has a total size of 104 µm by 26 µm.

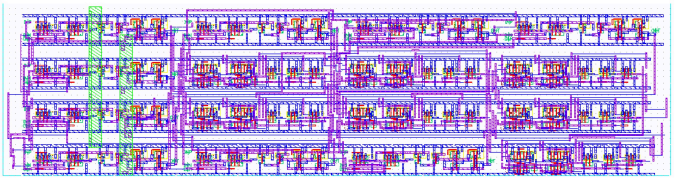


Figure 9. Compute cell consisting of 9 comparators and 7 full adders.

27 compute cells are then bundled into a 'compute row', which then has an additional 54 full adders to link the compute cells together. Figure 10 shows the design, with a total layout size of 1,200 µm by 150 µm.

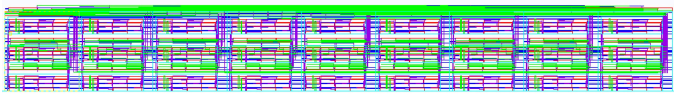


Figure 10. Compute row consisting of 27 compute cells and 54 additional full adders.

Figure 11 shows the final design, consisting of 6 compute cells to a design that is 1.4 mm by 1.3 mm in size, it consists of a total of 1,458 comparators (162 compute cells) and total of 1,510 full adders. At full efficiency, the Wallace tree adder is capable of 2/3 of the 1,510 full adders to produce the bits needed, due to time constraints, design optimization was not performed. For the Cadence designs, the metal layers were routed automatically, while the p wells, n wells, poly and device placement were placed manually.

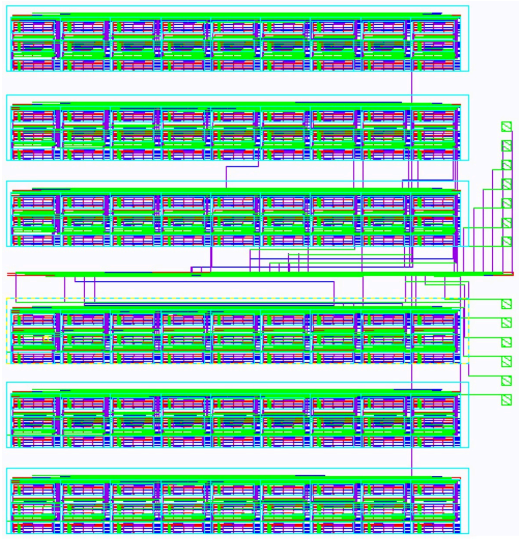


Figure 11. Final layout design of the 5-bit stochastic ADC, consisting of ~70,000 transistors in a 1.82 mm^2 die area.

IV. Simulation

Simulation files for the design were given in HSPICE, written and maintained by Synopsis. For simulation, instead of using proprietary industry tools, the world of free and open source software (FOSS) was used. This was due to speed and availability of server space. A server consisting of 54 Intel Haswell server cores with 40 GB of RAM was rented from AWS for simulation. The GPL licensed electronic design and automation suite (gEDA) was used for simulation and schematic capture. gSchem was used for schematic capture, ngSPICE was used for simulation and gNetlist was used to convert the schematic to a SPICE netlist. The most difficult part of this investigation was to fit HSPICE to be compatible with gEDA. The engine behind HSPICE is the level 54 Berkeley simulator (BSIM v4.4), is compatible with gEDA, the syntax, however, is wildly different.

From GlobalFoundries, the simulations are given in 4 different files, one for the NMOS device, one for the PMOS device, and 2 global device parameter files detailing tolerance and variations in the process. First, these 4 files had to be compiled into 1, as gSchem did not work properly when SPICE variables were stored in separate files from the device parameters. HSPICE has all of its equation variables inside of quotations and braces, for example:

```
acwv_fet_dist    = {0.01652e-6/(3*'total_acwv_fet')}
```

This equation, along with the rest had to be modified to remove these separators as shown in the following:

```
acwv_fet_dist    = 0.01652e-6/(3*total_acwv_fet)
```

After variable referencing issues are solved within, gSchem was used to produce the schematic of the simulation. gSchem has an issue when copying and pasting within a schematic, and due to the repetitive design of the stochastic ADC, this become an issue when doing so for 60 different devices and the need to rename each transistor with a unique name. A program was written, included in the appendix, that scanned a gSchem file and gave each transistor its own identifier.

After the schematic was written, gNetlist was used to convert the schematic into a SPICE netlist, however, as gSchem had repetitive attribute issues, so did gNetlist. A program was written to give each repeated net a unique name, also included in the appendix. The commands used to run the simulation are as follows:

(1) gSchem
(2) ./convertSchematic
(3) gnetlist -v -g spice-sdb -o ADC.net converted.sch
(4) ./convertNet
(5) ngspice
(6) set num_threads=54
(7) source converted.net
(8) run

In non-technical terms, line 3 states to use the converted schematic file, using a SPICE netlister, to create ADC.net, the ngSPICE compatible netlist with a verbose output. After the run command is finished, plots are produced following the schematic and the HSPICE specification files. Figure 12 shows that running 2 identical simulations can produce 2 different results, stemming from the statistical variation that is built into the SPICE model.

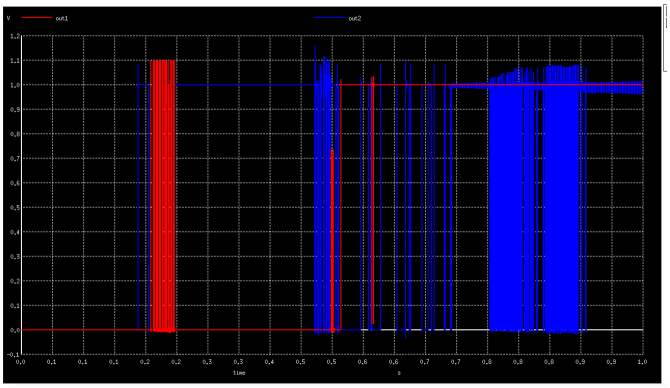


Figure 12. Sample output of 2 different simulation runs of identical schematics.

When 60 of these comparators are wired to the same input, clock, and reference voltage, one would expect the output to be similar to what is shown in Figure 1. Figure 13 shows the output of 60 comparators simulated using this method in ngSPICE.

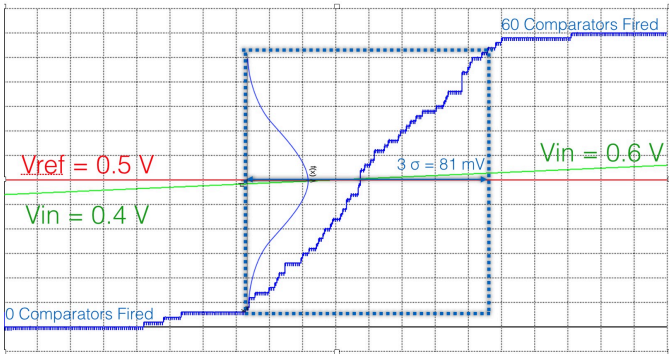


Figure 13. Output of 60 comparators simulated in ngSPICE. VDD = 1 V, Vref = 0.5 V, and a ramp of the input voltage from 0.4 V to 0.6 V. Simulated time of the ramp is 2 µs, with a clock of 10 ns, and a simulation step size of 10 ps.

The simulation proved that the stochastic ADC is functional, with a comparator offset standard deviation of 27 mV. This gives the ADC an effective range of 81 mV with a 500 mV reference voltage. Due to computational difficulty, more devices were not simulated. With 54 compute cores, this simulation ran for approximately 24 hours.

With significant rework, the simulation files successfully worked on gEDA, potentially allowing for open source device designs in the future.

V. Fabrication

The design shown in Figure 11 will be submitted to fabrication to GlobalFoundries in Summer of 2017, and be delivered to RIT for characterization and testing.

ACKNOWLEDGMENT

The author would like to thank Dr. Robert Pearson, Dr. Dale Ewbank, and Dr. Lynn Fuller for showing patience and assistance in the largely untested area of large scale simulation and design of ICs for the microelectronics engineering program at RIT.

REFERENCES

[1]S. Weaver, B. Hershberg and U. Moon, "Digitally Synthesized Stochastic Flash ADC Using Only Standard Digital Cells", IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 61, no. 1, pp. 84-91, 2014.

[2]S. Weaver, B. Hershberg, P. Kurahashi, D. Knierim and U. Moon "Stochastic Flash Analog-to-Digital Conversion", IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 57, no. 11, pp. 2825-2833, 2010.

[3] Saxena et al. "Variation in Transistor Performance and Leakage in Nanometer-Scale Technologies", IEEE Transactions on Electron Devices, vol. 55, no. 1, pp. 131-144, 2008

Zachary Baltzer received a A.S. in engineering science from Monroe Community College, Rochester, NY in 2014 and went on to receive a B.S. in microelectronic engineering from Rochester Institute of Technology, Rochester, NY in 2017.

He co-founded Hz Innovations, a Deaf technology startup company in Rochester, NY designing and programming a sound recognition device. He is currently with Nanya Technology, Burlington, VT designing DRAM.