A Gray-Code MOS Current-Mode Analog-to-Digital Converter Design

Philippe O. Pouliquen†, Kwabena A. Boahen‡, and Andreas G. Andreou†
†Dept. of Electrical and Computer Engineering
The Johns Hopkins University, Baltimore, MD 21218, U.S.A.
‡Dept. of Computation and Neural Systems
California Institute of Technology, Pasadena, CA 91125, U.S.A.

Abstract—**The purpose of this report is to expose a novel ADC, which was designed to operate without any clocking circuitry such as a sample-and-hold. Performance degrades gracefully as the input exceeds the ADC's maximum sampling rate, enabling one to use a single ADC for a large variety of applications. In addition, the converter presented here has other useful features such as low power dissipation, and high area efficiency.**

1 Introduction

Analog to digital converters (ADCs) are used in many applications to convert signals from the real world (which are continuous in both time and magnitude), to data streams that can be managed by digital computers (streams of numbers which are discrete in time and magnitude, and of finite length). In the process, a significant portion of the information is thrown out—a tradeoff that makes the information manageable. However, it is always the goal of the hardware designer to preserve as much as is possible or useful. In concrete terms, this means that the signal is sampled at a rate that is high enough to satisfy the Nyquist criterion (the sampling rate is twice the frequency of the the highest frequency component of the analog signal) and that the sampling resolution is fine enough that all relevant information is measurable.

Unfortunately, there is a tradeoff between sampling resolution and sampling rate for ADCs, and in some applications, one cannot get the desired speed and resolution [1].

2 The algorithmic ADC

When designing ADCs in VLSI, one of the most popular design is the algorithmic converter. The algorithmic converter is a variant of the Successive Approximation ADC (SA-ADC [1]), in which the digital controller is left out, and the ADC is constructed from a chain of n cells, each producing one bit of the digital output (figure 1). Each cell amplifies the input by a factor of two, and compares the result with a global reference level: if the result is less than the reference level, then it outputs a digital 0 and passes the result to the next cell. Otherwise, it outputs

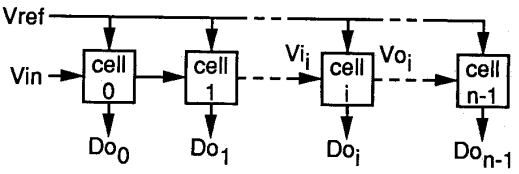

Figure 1: Block diagram of an algorithmic converter. Bit 1 is the MSB.

a digital 1 and passes the result less the reference level to the next cell. The analog transfer function is shown in figure 2. Note that in some applications, the reference

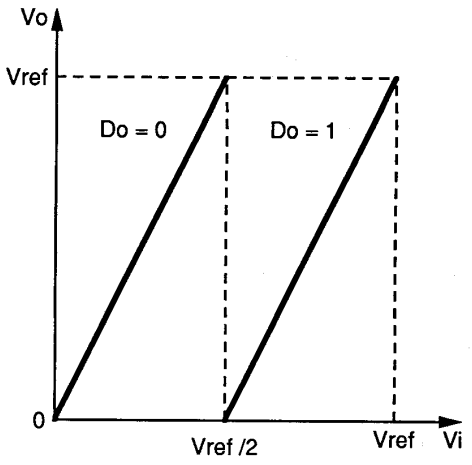

Figure 2: Transfer function for a cell of the binary code algorithmic converter.

itself is scaled, rather than amplifying the input signal at each stage. In either case, it should be clear that an arbitrary number of cells can be used (although the linearity is dependent on the accuracy of the cells), and that the conversion time is dependent only on the time it takes the signal to ripple through the n stages.

CH 3006-4/91/0000 – 1924 $1.00 © IEEE

In our applications [2], we want to place our ADC on the same silicon chip as the devices and circuits that we are testing. Although the algorithmic converter might require more area than a SA-ADC for some exceptional implementations, it is well suited for our application because the absence of the digital controller and the n bit digital-to-analog converter (DAC) simplifies the design considerably.

3 Gray-code output

To further reduce the size of the design, and eliminate any clocked circuitry (which introduces noise in the analog circuits, and increases power dissipation [3]), we would also like to eliminate the bulky sample-and-hold (S&H) circuitry. Unfortunately, when used without a S&H circuit, the algorithmic ADC is prone to disastrous errors: suppose an analog signal has been correctly digitized as 0110. The analog quantity increases slightly so that each cell passes the increase on to the next cell, and the last cell flips its state from 0 to 1. The digital output now correctly reads 0111. The analog quantity continues to rise, so that it approaches 1000. In this case, the first cell senses that its threshold has been exceeded, flips its state to a 1, and significantly reduces its output. The next cells each successively detect the change and flip their bits to 0. The output now reads 1000, but it has transitioned through 1111, 1011, and 1001 first (figure 3). If the digital output were latched at the wrong time, the software might conclude that a transient occurred in the data: 0111, 1111, 1000. Without a S&H circuit to synchronize the data latching, it is impossible to tell whether 1111 was a spurious or real transient. In most cases, these transients cannot be digitally filtered: the transient is composed of a step and an exponential decay (it contains both high and low frequency components).

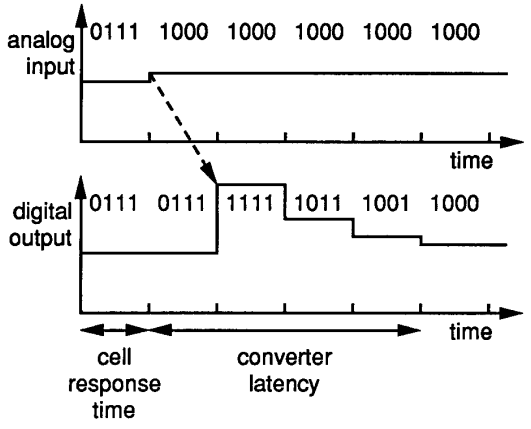

Figure 3: Intermediate states of the algorithmic converter.

Analysis of the algorithmic converter indicates that a step change in the input to the SAADC of less than 1 least-significant-bit (LSB) or $1/2^n$ of the range can cause an error of at most 1 most-significant-bit (MSB) or 1/2 the range under asynchronous latching of the digital output. This problem can be traced to the discontinuity in the analog transfer function of the converter's cells: small changes in the input to a cell can cause arbitrarily larger changes in the output (when the input is near the reference).

The gray-code algorithmic converter (GA-ADC) is an algorithmic converter with a continuous analog transfer function as shown in figure 4(a). In this case, the output of a cell changes at most twice as fast as the input. Although it may not be immediately obvious, the continuous transfer function results in a digital output that is Gray-coded.

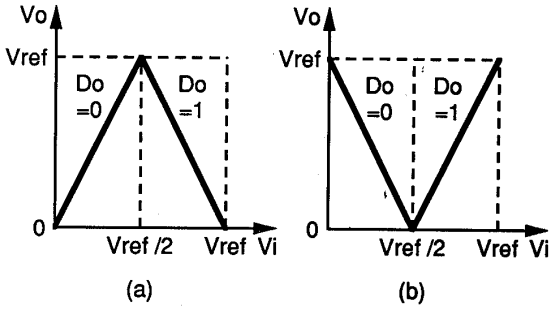

Figure 4: Two possible transfer functions of the Gray code algorithmic converter.

The tables in figure 5 compares the worst case errors in the digital output of the binary-coded algorithmic con-

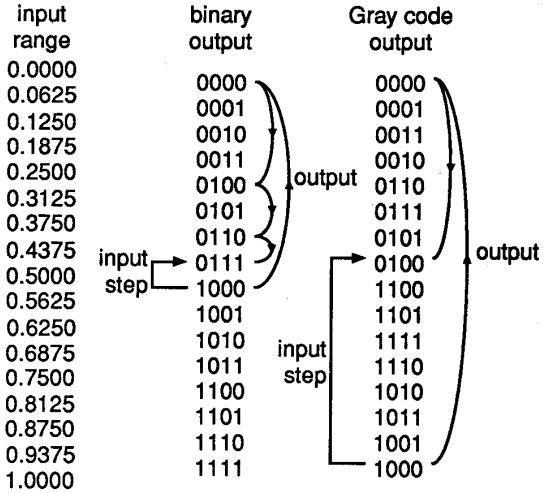

Figure 5: Worst case errors for algorithmic converters.

verter and the GA-ADC due to asynchronous data latching. To obtain an error of 1/2 the range from the GA-ADC, the input step must be at least 1/2 the range also. However, because of the basic property of Gray code that only one bit changes value between each successive Gray code value, a step change in the input of 1 LSB cannot cause a glitch in the digital output. For both converters, the worst case error is therefore the same (1/2 the range). However, for the binary output, the ratio of the error to the input step size tends to infinity as the number of bits goes to infinity. For the GA-ADC, this same ratio goes to 1.

4 Performance degradation

From the above discussion, we can conclude that the digital output of the GA-ADC will be correct under asynchronous latching if the input changes by less than 1 LSB in the time it take the signal to propagate through the GA-ADC. More explicitly, if for an n bit GA-ADC:

$$\text{input slew rate} < \frac{A}{dn2^n}$$

where A is the input range of the GA-ADC and d is propagation delay through a single cell, then the digital output will be correct under asynchronous latching. However, if this limit is exceeded, the errors in the digital output occur at the least-significant-bits first: if i is the position of a bit in the output ($i = 0$ being the most-significant-bit), the digital output may be in error only for bits i where:

$$\text{input slew rate} \geq \frac{A}{di2^i}$$

However, the GA-ADC will introduce a distortion due to the variable latency. This will not present a significant problem if:

$$\text{input slew rate} \ll \frac{A}{dn2^n}$$

5 Implementation and simulation

In our implementation, we made no effort to obtain pure Gray code digital output: as long as the analog transfer function is continuous, the analog to digital converter will have the properties described above. Our actual transfer function (shown in figure 4(b)) results in a reverse Gray code digital output.

To implement the GA-ADC we used the circuit shown in figure 7. Transistors M_0, M_1, M_4, and M_5 buffer the reference current I_{ref} which allows V_c to vary about the current comparator's inversion voltage. Transistors M_{13}, M_{14}, M_{15}, M_{17}, M_{18}, and M_{19} buffer the input current I_{in} and provide the gain (of 2) required for scaling the weight of each bit in the digital output. The current comparator, formed by transistors M_8 and M_{11}, is simply a digital inverter: if I_{ref} is larger than $2I_{in}$, V_c tends to increase towards the positive supply, the comparator's output goes low, and the difference current $I_{ref} - I_{in}$ is steered through M_9 to the next cell. Otherwise, V_c is low, the comparator's output is high, and the difference current is steered through M_{12} and the output mirror (composed from transistors M_{12}, M_3, M_6, and M_7) to the next cell. Since the source terminals for both M_9 and M_{12} are at V_c, and their gates are tied together, only one of the two transistors may be on at any given time, preventing the output mirror's input from being tied to its output. Note also that the reference current is used only once (as opposed to Nairn and Salama's design [4]), preventing the digital output (from each cell) from being skewed with respect to the analog output.

We designed a VLSI CMOS layout for the circuit described above, using $10 \times 10\mu m$ transistors for the mirrors. We then extracted the $10 \times 10\mu m$ layout and simulated it using Spice (version 3d1) with a $5\mu A$ reference current (figure 6).

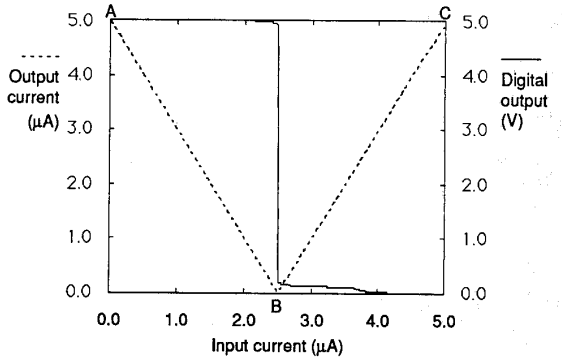

Figure 6: Simulated transfer function.

The intercepts of the transfer function at the edges of the range (points A and C) do not occur at exactly $5\mu A$, but rather at $4.986\mu A$ and $4.939\mu A$ respectively. Furthermore, there is a small leakage current on the order of $1pA$ at the midpoint of the range (point B). Finally, the digital output is slightly off centered: the digital output is at the inverter inversion voltage when I_{in} is $2.5031\mu A$. We therefore expect range-compression because the output range of a cell is smaller than the input range.

We also investigated the transient response of a cell. The delay d is strongly dependent on the input current: it is largest when the input current step is small and centered around $2.5\mu A$, since the difference current (which can be arbitrarily small) must charge or discharge the parasitic capacitance at node V_c.

6 Fabrication and testing

We designed a test chip containing four complete 18-bit converters and other test structures. The test chip was

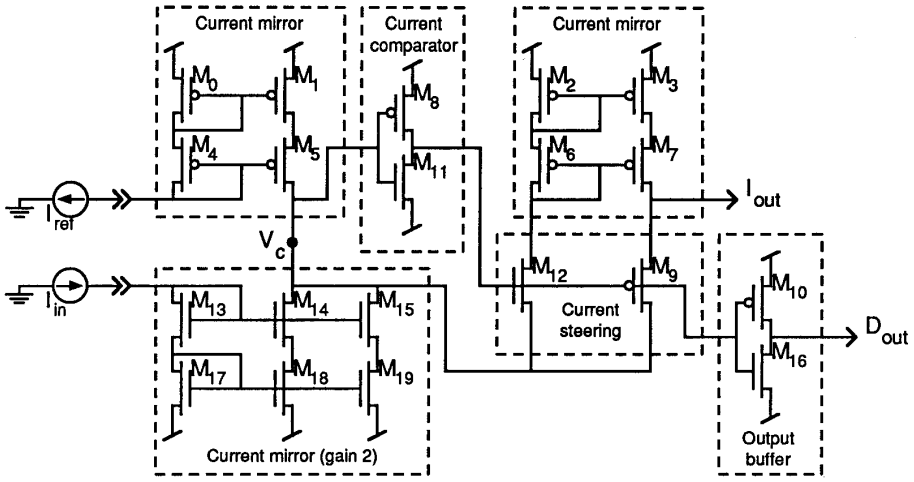

Figure 7: Schematic diagram of the GA-ADC.

fabricated through MOSIS in a $2\mu m$ N-well CMOS process.

Results from the test structure were consistent with the Spice simulations. Figure 8 shows the transfer curve for the first 8 bits of a GA-ADC. The transistors used

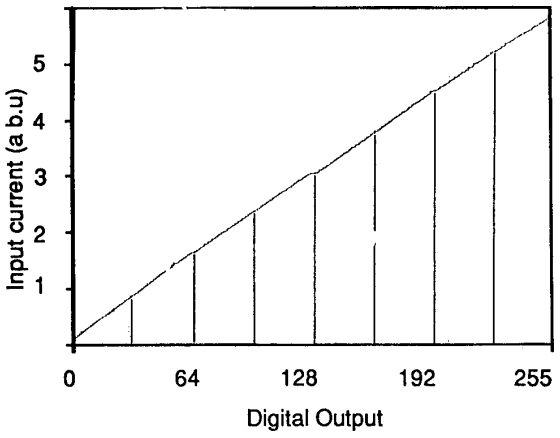

Figure 8: GA-ADC transfer curve for the 8 MSBs.

in the current mirrors of this implementation were $10 \times 10\mu m$. Note the 'flat' zone in the middle of the range (which is replicated at 1/4 and 3/4) due to the range-compression.

Only the first 10 bits were judged to be useful, since the other bits oscillated uncontrollably for a constant input. The GA-ADC was connected to a computer which satisfactorily displayed signals up to $10KHz$.

7 Conclusion

The design presented here is our first generation GA-ADC. Tests have indicated that although it performs satisfactorily, there is much room for improvements. We are currently developing new GA-ADC cells to reduce the noise in the LSBs, to reduce the delay d to a predictable value, and improve linearity.

Acknowledgments

This research was partially supported by the Independent Research and Development program of the Johns Hopkins University Applied Physics Laboratory and by an NSF Research Initiation Award, MIP-9010364.

References

[1] P. Horowitz and W. Hill, *The Art of Electronics.* Second edition (1989), Cambridge University Press.

[2] A. G. Andreou, K. A. Boahen, P. O. Pouliquen, A. Pavasović, R. E. Jenkins, and K. Strobehn, "Current-Mode Subthreshold MOS Circuits for Analog VLSI Neural Systems," *IEEE Trans. Neural Networks,* vol. 2, pp. 205–213, March 1991.

[3] V. Kantabutra and A. G. Andreou, "On a General Design Principle for Low-Power Digital VLSI Circuits," *Proc. Twenty Fifth Annual Conference on Information Sciences and Systems,* The Johns Hopkins University, 1991.

[4] D. G. Nairn and C. A. T. Salama, "Algorithmic analogue/digital convertor based on current mirrors," *Electronics Letters,* vol. 24, pp. 471–472, April 1988.