SELF-TIMED ARCHITECTURE FOR MASKED SUCCESSIVE APPROXIMATION ANALOG-TO-DIGITAL CONVERSION

TASKIN KOCAK, GEORGE R. HARRIS and RONALD F. DEMARA

School of Electrical Engineering and Computer Science
University of Central Florida, Orlando, Florida 32816-2450

tkocak@cpe.ucf.edu

Received (13 December 2004)
Revised (13 October 2005)
Accepted (15 June 2006)

In this paper, a novel architecture for self-timed analog-to-digital conversion is presented, designed using the NULL Convention Logic (NCL) paradigm. This analog-to-digital converter (ADC) employs successive approximation and a one-hot encoded masking technique to digitize analog signals. The architecture scales readily to any given resolution by utilizing the one-hot encoded scheme to permit identical logical components for each bit of resolution. The 4-bit configuration of the proposed design has been implemented and assessed via simulation in 0.18-μm CMOS technology. Furthermore, the ADC may be interfaced with either synchronous or four-phase asynchronous digital systems.

Keywords: Asynchronous digital systems, self-timed designs, analog-to-digital conversion, null convention logic, successive approximation

1. Introduction

The need for high performance, low power, and low electromagnetic interference (EMI) analog-to-digital converters (ADCs) has led researchers to consider asynchronous approaches as alternatives to conventional clocked designs. A motivating factor has been that as clock frequencies increase, so do complications regarding the clocks effects on EMI, power dissipation, and average-case performance [18] [21] [22]. In addition, clock transitions facilitate the simultaneous occurrence of multiple switching events. This results in maximum taxation of the supply rails at nearly identical time intervals creating a power-rail grouping effect. Unfortunately, this may corrupt sensitive analog input signals as they are being sampled, and consequently lead to inaccurate conversions. While the benefits of asynchronous design have been demonstrated in digital logic circuits [15], we investigate here novel means by which these advantages can be carried over into the mixed-signal domain.

Asynchronous circuits fall into two main categories: self-timed and bounded-delay models. NCL assumes delays in both logic circuits and interconnects to be unbounded. On the other hand, well-known methods such as Huffman circuits [10], burst-mode circuits [11] and micropipelines [12] assume that delays in both gates and wires are bounded. Delays are added based on worst-case scenarios to avoid

hazard conditions. This leads to extensive timing analysis of worst-case behavior to ensure correct circuit operation [9]. Self-timed designs have demonstrated higher application-level throughput since average case delay is less than that of synchronous circuits which are designed to accomodate the worst-case propagation delays [18] [19]. Due to elimination of clock trees they have demonstrated reduce power consumption [20] [22]. Also important for ADC applications, NCL and other self-timed circuits have shown reduced Electromagnetic Interference effects and noise due to their inherently non-synchronized switching characteristics [21] [22]. A design consideration is the increased area required when using NCL, which is approximately 1.5 to 2 times as much as the equivalent synchronous design. However, for large designs, such as SoCs, the processor core(s) normally require(s) less than half of the chip's total area. Therefore, the increased area for the NCL implementation of the processor core(s) is less significant, especially considering the increased robustness and numerous other advantages. For example, the initial version of the Motorola HCS08 processor implemented in the NCL paradigm (the NCL08) and fabricated using static CMOS gates with a 0.25-μm process, shows a 40% reduction in power and a 10 dB reduction in peak noise over its clocked Boolean counterpart, while operating at a comparable throughput [22].

The state of asynchronous ADC design is still in its infancy, with relatively few designs being formally presented [1] [2] [3] [4]. To date, existing designs have demonstrated comparable or faster average conversion times when evaluated against synchronous converters. They have also demonstrated various means of achieving metastability-free conversion under low power and low noise constraints.

In spite of the potential advantages of asynchronous conversion approaches, a fundamental question arises. That is whether asynchronous converters that show temporally indeterministic nature can work in real-time applications, which require conversions within a fixed time interval. However, it is possible to guarantee an asynchronous converter can complete operations within a time bound [2], but these circuits reintroduce the need for stringent timing analysis similar to that found in clocked systems. Unbounded delay converters such as the ones presented in this paper can deliver predictable maximum and average conversion rates, but do not guarantee a minimum rate. Nonetheless, the minimum achieved rates for synchronous converters remain influenced by physical operating conditions in a similar manner.

In the following sections, an overview of NULL Convention Logic (NCL) is first presented, as it is used to realize the digital logic functions in the ADC. Next, the proposed architecture of the self- timed successive approximation (SA) ADC is described, independent of resolution, with discussions on both the digital and analog functions. The 4-bit configuration of the ADC architecture is then simulated in SPICE using Cadence design tools and a 0.18-μm CMOS technology library. The simulation results and their implications are subsequently discussed.

2. NULL Convention Logic (NCL)

NULL Convention Logic (NCL) is a self-timed logic paradigm developed by Theseus Logic Inc., whereby control is inherent in each datum [5]. NCL is unlike conventional Boolean logic, where the control variable, time, is external to the logic expression and must be carefully exercised in order to maintain optimal and yet safe operating circuitry. NCL conforms to an unbounded delay model, assumes wires that fork are isochronic, and is based on: a 1-of-m encoding scheme, the use of state-holding gates, and completion detection on the output of processing stages, allowing for a handshaking protocol to control input wavefronts. The typical 1-of-m encoding chosen in most designs is a dual-rail realization; whereby two wires encode three logic states (NULL, DATA1, and DATA0) to represent the value of a single bit. Figure 1 depicts the dual-rail state assignments in NCL. DATA0 corresponds to a Boolean logic 0 value, DATA1 corresponds to a Boolean logic 1 value, and the NULL state denotes an indeterminate value that acts as a spacer between successive DATA wavefronts. In effect, the propagation of a NULL wavefront clears the state-holding capability of intermediary gates, while simultaneously indicating that the output is not yet available. The wires of a NCL 1-of-m encoding scheme are mutually exclusive, so only one rail is ever asserted at any time.

	DATA0	DATA1	NULL	Undefined
Rail0	1	0	0	1
Rail1	0	1	0	1

Fig. 1. Dual-rail NCL state encoding

NCL state-holding gates, termed threshold gates, can be viewed as an extension of the Muller C-element [14]. The primary type of NCL threshold gate is the THmn gate, where $1 = m = n$ as illustrated in Fig. 2a. THmn gates have n inputs, and at least m of the n inputs must be asserted before the output signal will assert. The gates are designed with hysteresis, so all asserted input signals must be de- asserted before the output signal is de-asserted. Hysteresis ensures a transition back to the NULL state before the next DATA state. An example static CMOS implementation of a TH32 gate is given in Fig. 2b.

3. ADC Architecture

The basic principles of the asynchronous successive approximation ADC architecture that we propose in this paper are similar to those used in conventional synchronous designs [16], and can be organized into four functional components: a sample and hold mechanism to capture and maintain various values of the analog signal, a

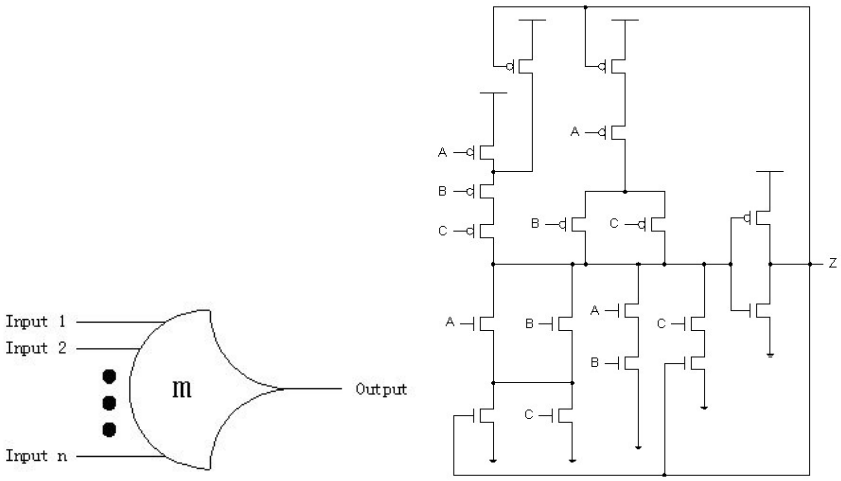

Fig. 2. a) THmn threshold gate; b) Static CMOS implementation of TH32 gate

digital logic section that generates the approximations, a digital-analog converter (DAC) to generate an analog signal based on a digital word, and finally an analog comparator used to compare the output of the DAC against the captured analog input. The successive approximation algorithm initiates conversion by asserting only the most significant bit (MSB) of the data path. This initial value represents the midpoint of the allowable analog range. Comparison between the initial value and the outside analog signal determines if the MSB should remain asserted. If the analog signal is less than the initial guess, the MSB is de-asserted. Conversion continues by asserting the next MSB and performing again another comparison. The process iterates until the states of all bits have been determined. The number of conversion cycles is directly proportional to the number of bits of resolution. More details about the successive approximation ADCs can be found in [17].

We call this new ADC architecture, masked asynchronous successive approximation (MASA) ADC due to the inclusion of a novel one-hot masking function used in the combinational logic component. The architecture block diagram is shown Figure 3 and is independent of resolution. The four basic functional components are evident, with the sample-and-hold, DAC, and comparator circuits on the left-hand side, whereas the digital logic section is located on the right.

The digital circuitry is comprised of three NCL registers, combinational logic, and a NCL modulator. The three NCL registers accommodate circulation of the DATA and NULL wavefronts while prohibiting improper overwriting or deadlock scenarios [5]. Initially, the register stages NCL Register 1, NCL Register 2, and NCL Register 3 are initialized to DATA, NULL, and NULL, respectively. NCL Register 1 will request for NULL, NCL Register 2 will request for DATA, and NCL Register 3 will also request for DATA. Each register provides acknowledgements to

the next upstream register in terms of request for DATA (rfD) or request for NULL (rfN). The lines labeled *ki* and *ko* provide handshaking signals to a register input or output, respectively. Finally, *Completion Detection* logic on the output of each register senses whether a complete set of DATA or NULL values for all bits of the word are currently available. The handshaking signal is inserted into each of the threshold gates of the upstream register, allowing only complete DATA or NULL wavefronts for all bits to pass each stage of registration.

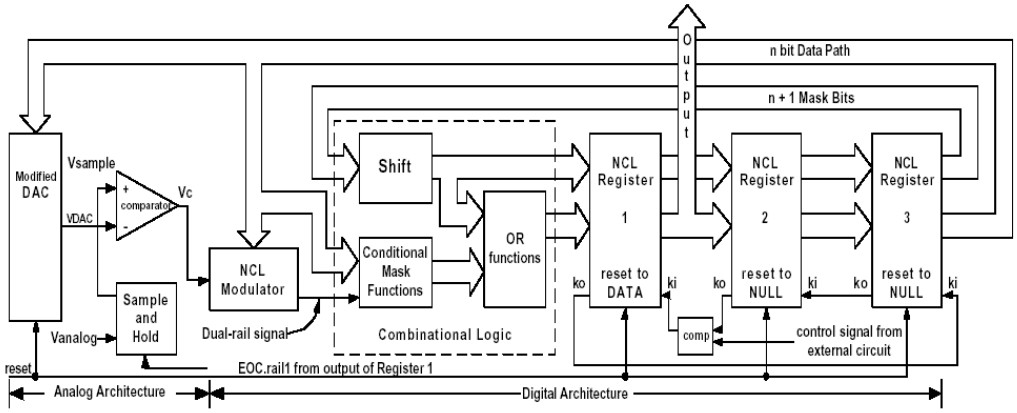

Fig. 3. Masked asynchronous ADC architecture

3.1. *Combinational Logic*

Combinational logic circuitry is responsible for executing the successive approximation algorithm. The function of the logic for any given iteration n, excluding the last iteration, is to determine if the asserted bit n should remain asserted or be set to logic '0', i.e. DATA0.

The status of bit n is given by the output voltage of the comparator, Vc, and sent to the combinational logic circuitry via the NCL Modulator. Concurrently, the logic circuitry must also set the next most significant bit, $n + 1$, to logic '1', i.e. DATA1, so further iterations are possible. When the ADC has performed its last iteration, an End-Of-Conversion (EOC) bit is asserted, signifying a conversion has been completed. The logic then resets the ADC by asserting only the MSB to DATA1 and all other bits to DATA0. The next conversion may then occur. The combinational logic circuitry is derived using the Threshold Combinational Reduction (TCR) methodology [6] that employs truth tables, karnaugh maps or a NCL logic minimization program to generate the expressions. Once the expressions have been derived, they must be checked for completeness of input to ensure delay insensitivity. Expressions are made complete with respect to the inputs if not already

so. The expressions are then realized into two-level logic, providing the maximum number of inputs does not exceed the capacity of the threshold gates in the NCL libraries used. The result is a NCL equivalent form of the Boolean sum-of-products expression. To reduce logic complexity, an extra set of dual-rail wires is used in conjunction with the data path. Termed mask bits, this NCL signal set is one-hot encoded, with each mask bit associated exclusively with a particular bit in the data path. Therefore the asserted bit in the mask corresponds to the data bit under refinement, allowing the combinational logic to render a decision. There are $n+1$ mask bits for every n bits of resolution. In particular, an additional mask bit is needed to indicate a conversion is complete on the output (EOC signal) and it is used to reset the converter for the subsequent computation. In essence, the additional mask bit is identical in purpose to the EOC signal seen in the previous design. Thus, after accounting for n data bits plus $n+1$ mask bits, a total of $2(2n+1)$ wires are required to achieve a resolution of n bits using a dual-rail encoding. In addition, provision of independent mask lines facilitates combinational logic circuit by using only three levels of threshold gates regardless of the number of bits. This implies that conversion to arbitrarily fine resolution can be obtained without increasing logic levels in the data path assuming $2(2n+1)$ wires are available. Design modularity is obtained as the same functions are performed on each bit of the data path. To accomplish this task, combinational logic consists of three sub-components: a shift operation, an OR function, and a conditional mask.

3.1.1. *Shift operation*

The shift operation moves the one-hot NCL mask bit from MSB to MSB-1 to MSB-2 etc., by simply rerouting wires accordingly. Figure 4 depicts the shift operation. Two rails or wires are required for each bit since a dual-rail encoding is used. In this figure, $m0$ is the MSB, with $m0^0$ representing rail0 of the MSB and $m0^1$ representing rail1 of the MSB. Intuitively, subsequent bits follow this logical pattern. When the one-hot signal reaches the LSB, end-of-conversion has occurred, since there are $n+1$ mask bits for n data bits, and the system will reset itself to its initial approximation during the next iteration. The LSB shifts the one-hot signal to the MSB, ensuring further approximations.

3.1.2. *OR function*

The OR function shown in Fig. 5 is the Boolean equivalent of an OR gate, although the OR function must incorporate the dual-rail nature of NCL and thus requires two threshold gates. The OR function is said to be complete with respect to its inputs, implying that the output will never assert until both input bits have arrived, assuring a more delay insensitive design. For every bit in the data path there is a corresponding OR function. The OR function uses the one-hot mask to set the next bit under refinement in the data path. Each OR function accepts one mask bit as

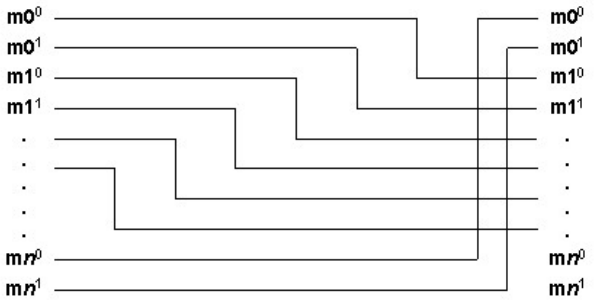

Fig. 4. Shifting the mask bits

its input along with its corresponding data bit, and provides a new data signal as output. All previously determined bits maintain their values as they pass through their respective OR functions.

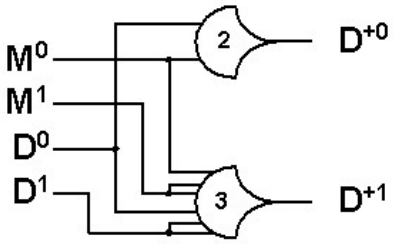

Fig. 5. The OR function

3.1.3. *Conditional mask functions*

The conditional mask is the core of the combinational logic circuitry, as it determines whether the bit pointed-to by the mask should remain set or be de-asserted. Like the OR function, conditional mask logic is exclusive to one data path bit, and is replicated for all bits in the data path. Figure 6 illustrates the functionality of the conditional mask logic by means of a Karnaugh map, and equations 1 and 2 provide the output expressions. Since the expressions are derived in NCL, covering all ones in the Karnaugh map results in an expression for rail 1 of the output variable. Likewise, covering all zeros in the Karnaugh map leads to an expression for rail 0 of the output variable. Variable D represents the data path bit, M represents the corresponding mask bit, Vc specifies the voltage comparison from the comparator sent via the NCL modulator, and R designates the LSB of the mask set (EOC),

which reinitializes the logic so the next approximation can occur.

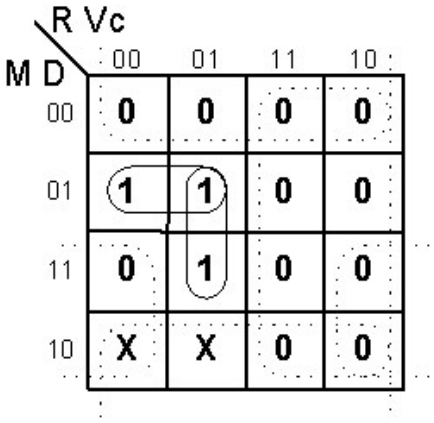

Fig. 6. Karnaugh map for the conditional mask

$$D^{+1} = M^0 D^1 R^0 + D^1 R^0 Vc^1 \tag{1}$$

$$D^{+0} = D^0 + R^1 + M^1 Vc^0 \tag{2}$$

NCL allows simplification of expression through the mapping to weighted NCL gates [6]. Thus only one threshold gate is required per equation, resulting in only two threshold gates per bit. Figure 7 depicts the two weighted gates utilized in the conditional mask design.

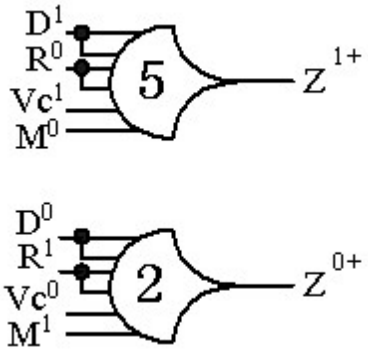

Fig. 7. Gate logic for 1-bit conditional mask

3.1.4. *NCL modulator and analog modules*

The NCL modulator performs three primary functions: it converts the comparator output from a Boolean to a dual-rail signal, inserts NULLs into the data stream in the appropriate order - as the analog circuitry is not able to generate such a signal on its own, and it also adds a delay during transitions from the NULL state to the DATA state. Delay is needed to allow the DAC time to settle to an accurate value and to allow the comparator time to produce the correct output. Delay ensures voltage differences between the outputs of the DAC and sampled analog signal $\geq \frac{1}{2}$ LSB will resolve. Therefore, the amount of delay inserted into the system corresponds to the resolution of the ADC. The greater the resolution, the greater the delay required. However, delay is small, and is on the order of a few nanoseconds for 0.18-μm CMOS technology. No delay is required on transitions from DATA to NULL, as the DAC maintains its previous value during a NULL cycle. This is the only delay insertion necessary and maintains glitch-free operation. Since the digital circuitry was realized using NCL threshold gates, modifications were required to the analog portion in order to properly handle NULL values. The DAC is a conventional resistor ladder, however, data registers are placed on the inputs to the DAC in order maintain the current data while the NULL transpires. The previous data set is maintained on the DAC during a NULL state. The comparator is a standard Boolean design. The sample and hold circuit is also standard, yet it uses rail1 of the EOC bit to either sample or to hold. The control signal is used to interface the ADC to either asynchronous or synchronous digital inputs.

4. Simulation Results

The masked asynchronous successive approximation ADC described above has been implemented in Cadence modeling software using 0.18-μm CMOS technology. The overall performance summary is given in Table 1. The converter has a resolution of 4-bits. The converter evaluates analog signals ranging from 0 to 1V peak-to-peak (pp). The $2^4 = 16$ unique states are contained in the 4-bits of resolution, thus providing step sizes of 1V/16 = 62.5 mV between quantization levels. The digital circuitry of the masked asynchronous successive approximation ADC contains 141 threshold gates, resulting in a transistor count of 1764. The number of transistors used to realize the analog architecture is 184. Therefore, the total amount of transistors in the ADC is 1930.

A simulation conversion cycle is shown in Fig. 8. The analog signal to be converted is Vanalog, a 9MHz sine wave ranging from 0-1V pp. After an initial reset stage of 5ns, mask4.rail1 (EOC signal) deasserts, indicating to the sample and hold device to operate in hold mode. This is verified by Vsample, capturing a Vanalog value of 674.603mV, and retaining the signal throughout the conversion process. The ADC is expected to convert the 674.603mV analog value into a NCL dual-rail digital word analogous to a Boolean output of 1010. Let bit A denote the MSB and bit D represent the LSB. Accordingly, when the first conversion is complete,

Table 1. Overall Performance Summary

Technology	0.18-um CMOS
Resolution	4 bits
SNR	24.6 dB
Dynamic range	21.1 dB
Sampling rate	33.5 Msamples/s
Number of transistors	1930
Supply voltage	2.5 V
Power consumption	2.77 mW

A is anticipated to be in a DATA1 state, implying the de-assertion of A.rail0 and the assertion of A.rail1. Furthermore, bit B is expected to be set to DATA0, bit C to DATA1, and finally bit D to a DATA0. Multiplying our predicted answer of 1010 (decimal 10) by the step size of 62.5mV, we obtain 625mV, the closest discrete representation of 674.603mV by means of this conversion process.

In the first conversion cycle, the converter is set to a NCL value analogous to a Boolean representation of 1000, the midpoint value, physically realized as 500mV. This can be verified as the only rail1 asserted in the data path lies in the MSB, indicating DATA1 present on A. All other bits are set to DATA0. Since 674.603mV is greater than the initial approximation of 500mV, A retains the DATA1 value. While the result of A is being determined, B.rail1 is set by the OR function, and a DATA1 is present on MSB-1 before the next conversion cycle commences. Conversion cycle two determines Vsample is less than the NCL value analogous to Boolean 1100, which converts to 750mV. Thus, MSB-1 is set to DATA0, as depicted by the assertion of B.rail0. Again the next bit is set to DATA1, this time bit C, before starting the third conversion cycle. The remainder of the timing diagram proceeds similarly until the approximation process ends around 30ns, as indicated by bit mask4 (EOC) being set to DATA1. At this time the ADC has approximated Vanalog to a NCL value analogous to Boolean 1010, as predicted, and the data-set may be transferred to the external digital system for processing. During the iteration proceeding mask4 being set to DATA1, the ADC resets the logic back to the midpoint value, readying itself for the subsequent approximation. A second conversion is shown in Fig. 8, between the time intervals of approximately 36-66ns. Vsample captures and holds a value of 982.68mV. This analog value is above the largest equivalent quantization level for the device and therefore is converted to a NCL equivalent form of the Boolean value 1111.

The average conversion time is approximately 29.8ns, yielding a sampling rate of 33.56 Mega Samples Per Second (MSPS). Consequently, this ADC can accept input analog signals with frequencies up to approximately 16.5MHz as governed by the Nyquist frequency criterion $f_{ADC} \geq 2 * f_{analoginput}$.

Simulations regarding current consumption were also conducted. The masked asynchronous successive approximation ADC draws on average, approximately

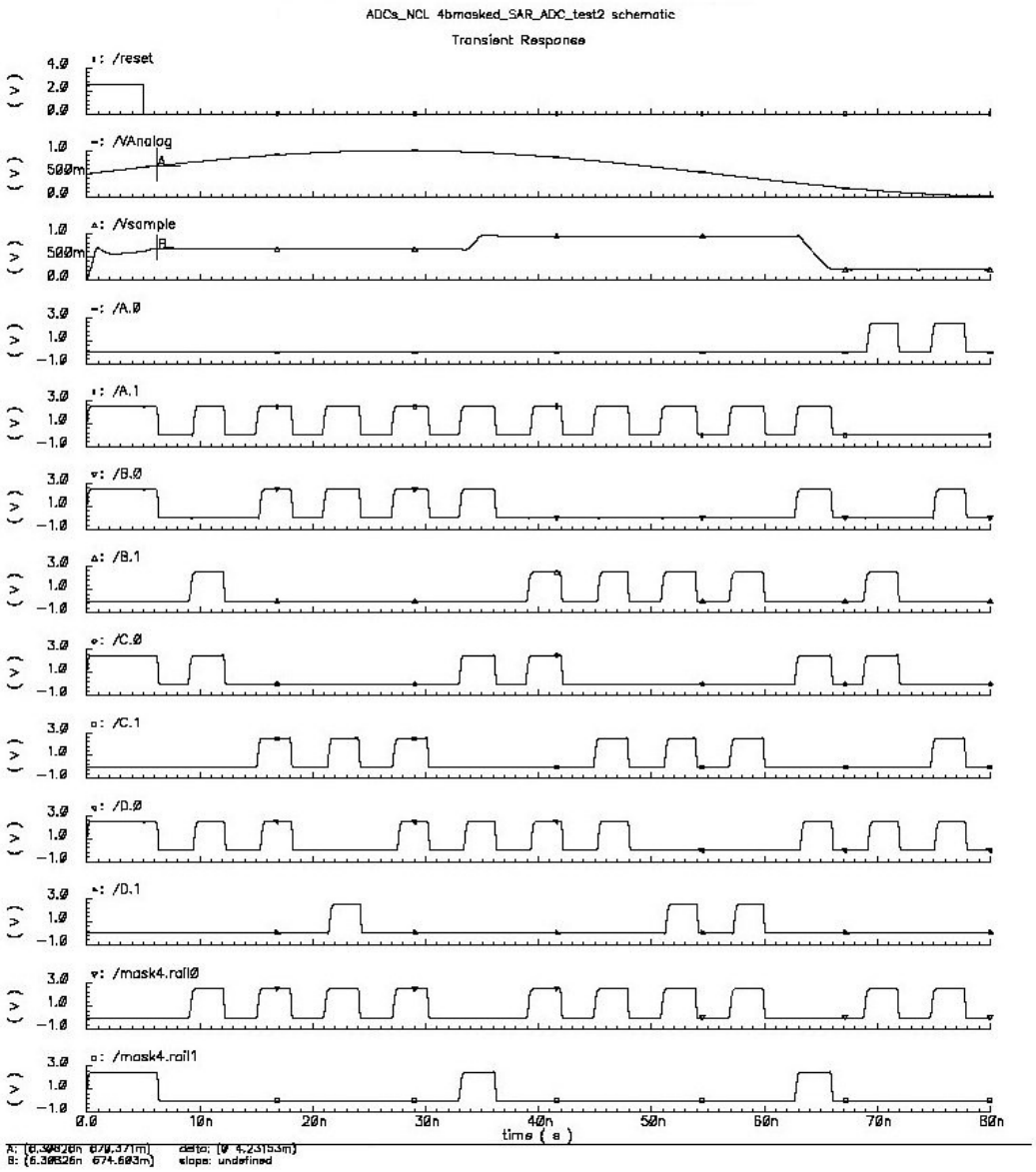

Fig. 8. Simulation results

1.108mA of current, using a 2.5V power source. Providing that device operation is relatively predictable, the ADC dissipates 2.770mW of power. 2.5V supply voltage was used because previous NCL libraries used larger line widths, however using a lower voltage source such as 1.8V will further reduce the power consumption.

4.1. *Comparison of synchronous and asynchronous ADCs*

Table 2 compares the recently reported both synchronous and asynchronous ADCs with our proposed design. In terms of conversion time, the asycnhronous designs seem to do better due to average case analysis. Our proposed NCL based ADC has a slightly lower conversion time compared to the similar asynchronous successive approximation ADC [1] [2]. The power consumption of the proposed ADC is also lower than the other reported designs. Chip layout area as well as the transistor count information are not explicitly available for the other asynchronous designs. Hence, we compared the number of transistors in our design with that of the synchronous ADC reported in [13]. Due to the NCL pipelines, our design employs more transistors than the synchronous version.

Table 2. Comparison with other designs

Design	Conversion time	Power	Transistor Count
Sync. 4-bit SA [2]	48ns	-	-
Asynch. 4-bit SA [1] [2]	32.5ns	-	-
Sync. 4-bit pipeline [7]	-	4.8 mW	-
Sync. 4-bit ripple-flash [8]	40ns	15 mW	-
Async. 7-bit micropipeline flash [3]	-	27.7 mW	-
Sync. 4-bit flash [13]	-	3.9 mW	1470
Our design	29.8ns	2.8 mW	1930

5. Conclusions

In this paper, a novel self-timed ADC architecture is proposed based on the NULL Convention Logic (NCL) paradigm. The ADC employs an innovative masked architecture, whereby an additional set of bits is used in conjunction with other circuitry to effectively and efficiently convert the analog signal. The masked architecture scales readily to multiple bit resolutions due to identical logic for each bit. A 4-bit version of the architecture is implemented using 0.18-μm CMOS technology indicates correct functionality of the design and provides a measure of performance. Comparison with other designs show that the proposed design performs better with respect to conversion time and power dissipation with some increase in the overall transistor count.

References

1. D.J. Kinniment, A.V. Yakovlev, and B. Gao, "Synchronous and Asynchronous A-D Conversion", *IEEE Transactions on Very Large Integration Systems*, Vol. 8, No. 2, pp. 217-220, April 2000.

2. D.J. Kinniment, B. Gao, A.V. Yakovlev, and F. Xia, "Towards asynchronous A-D conversion", *Fourth International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pp. 206 -215, 1998.
3. D.J. Kinniment, A.V. Yakovlev, "Low power, low noise, micropipelined flash A-D converter", *IEE Proc.-Circuits, Devices and Systems*, Vol. 146, No. 5, pp. 263-267, October 1999.
4. M. Conti, S. Orcioni, C. Turchetti, and G. Biagetti, "A Current Mode Multistable Memory using Asynchronous Successive Approximation A/D Converters", pp. 513-516, 1999.
5. K. M. Fant and S. A. Brandt, "NULL Convention Logic: A Complete and Consistent Logic for Asynchronous Digital Circuit Synthesis",*International Conference on Application Specific Systems, Architectures, and Processors*, pp. 261-273, 1996.
6. S.C. Smith, "Gate and Throughput Optimizations for NULL Convention Self-Timed Digital Circuits", *Ph.D. Dissertation*, School of Electrical Engineering and Computer Science, University of Central Florida, 2001.
7. A. Stojcevski, J. Singh and A. Zayegh, "A reconfigurable analog-to-digital converter for UTRA-TDD mobile terminal receiver", *Proc. of the 45th IEEE Midwest Symp. on Circuits and Systems*, Tulsa, OK, Aug. 2002.
8. R. Baumgartnet and Y. Leblebici, "Realization of compact low-power ripple-flash A/D converter architectures using conventional digital CMOS technology", *Proc. of the IEEE ASIC/SOC Conf.*, Rochester, NY, Sept. 2002.
9. S.C. Smith, R.F. DeMara, J.S. Yuan, D. Ferguson, D. Lamb, "Optimization of Null convention self-times circuits", *Integration, the VLSI Journal*, vol. 37 pp. 135-165, 2004.
10. S.H. Unger, *Asynchronous Sequential Switching Circuits*, Wiley, New York, NY, 1969.
11. S.M. Nowick, D.L. Dill, "Synthesis of asynchronous state machines using a local clock", *Proc. of the ICCAD*, pp. 192-197, 1991.
12. I.E. Sutherland, "Micropipelines", *Comm. of the ACM*, vol. 32, no. 6, pp. 720-738, 1989.
13. L. Lamarre, M. Louerat, A. Kaiser, "A simple 3.8 mW, 300 MHz, 4-bit flash analog-to-digital converter", *Proc. of the SPIE*, vol. 5837, pp. 825-832, 2005.
14. D. E. Muller and W.S. Bartky, "A theory of asynchronous circuits", *Proc. of Int. Symp. on Theory and Switching*, vol. 29, pp. 204-243, 1959.
15. J. Sparso and S. Furber, *Principles of Asynchronous Circuit Design*, Kluwer Academic Publishers, 2001.
16. S. Mortezapour and E. Lee, "A 1-V, *-bit successive approximation ADC in standard CMOS process", *IEEE Journal of Solid-State Circuits*, vol. 35, no. 4, pp. 642-646, 2000.
17. W. Kaster, *The Data Conversion Handbook*, Analog Devices, Inc., published by Elsevier, 2005.
18. I. Sutherland and J. Lexau, "Designing fast asynchronous circuits," *Proc. of Int'l Symp. on Advanced Research in Asynchronous Circuits and Systems*, March 2001, pp. 184-193.
19. S. Rotem, K. Stevens, R. Ginosar, P. Beerel, C. Myers, K. Yun, R. Kol, C. Dike, M. Roncken, and B. Agapiev. "RAPPID: An asynchronous instruction length decoder," *Proc. of Int'l Symp. on Advanced Research in Asynchronous Circuits and Systems*, April 1999, pp. 60-70.
20. S. B. Furber, D. A. Edwards, and J. D. Garside. "AMULET3: A 100 MIPS asynchronous embedded processor," *Proc. of Int'l Conf. on Computer Design*, Septem-

ber 2000.

21. K. van Berkel, R. Burgess, J. Kessels, A. Peeters, M. Roncken, and F. Schalij, "A fully asynchronous low power error corrector for the DCC player," *Proc. of Int'l Solid State Circuits Conference*, pp. 88-89, February 1994.
22. J. McCardle and D. Chester, "Measuring an asynchronous processor's power and noise," *Proc. of Synopsys User Group Conference*, Mountain View, Cailf., 2001, pp. 66-70.