

1.8V 0.18 μ m CMOS Novel Successive Approximation ADC With Variable Sampling Rate

Abstract

With the increased sophistication of System-on-Chip (SOC) architectures comes an increased need for low power Analog to Digital converters. These converters have many uses including Built-in-Self-Test (BIST) applications. In this paper, we present a novel Successive Approximation ADC. We show how one can utilize an existing DAC structure on an SOC to realize an effective Analog-to-Digital converter for self-testing applications. The achieved ADC operates at 125 MSamples/s consuming only 3.8 mW of power. The design is implemented in 0.18 μ m 1.8-V CMOS technology.

1. Introduction

With the increased density of deep submicron technologies come further integration techniques, combining many mixed signal applications with digital circuitry. With process variation it is necessary to ensure proper functionality of analog circuitry. In order to reduce the cost of testing, BIST methodologies have been proposed [1]. In addition with the highly integrated nature of modern devices, it is desirable to implement these BIST devices with the least amount of area and power overhead possible. To reduce cost, the reuse of existing circuitry is highly desirable.

In this paper, we present a methodology of constructing an analog to digital converter appropriate for testing low frequency signals. In the particular application considered, high resolution is not required as we are only trying to ensure basic functional characteristics of the signal generators. However, the criteria requires that the full operating range of the signal generator is valid and that the signal shape is correct and is free of large signal noise often associated with digital circuits in close proximity.

The design considered will take advantage of preexisting structures common on SOC applications. DACs are often used in Mixed Signal SOC designs, and with a minimal amount of additional circuitry, we are able to build a Successive Approximation ADC. While these ADC's are inherently slow, we demonstrate a novel algorithm that enables us to get changing output values every other clock cycle under

“ideal” conditions (see Section 4). Other converter designs that enable output values to be achieved every cycle are Flash, Folding, and Pipeline designs. However, Flash and Folding in particular tend to have high power consumption [7]. Pipeline converters are difficult to implement in low voltage operating conditions due to there extensive analog design requirements [8].

2. Architecture

The Successive Approximation ADC is composed of 3 major parts. These are: a DAC, a Comparator and a Successive Approximation register [3, 4]. As shown in Figure 1, we see that this ADC functions by iterating through values, via the Successive Approximation Register, until the DAC and input are equal [5].

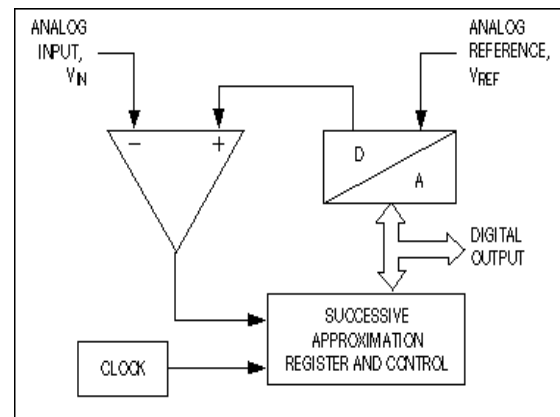


Figure 1 – Block Diagram of basic Successive Approximation ADC.

This type of converter is algorithmic in nature and as such, the implementation of the register dictates the performance limitation of the converter. Typically, the register implements a divide and conquer style algorithm to find the value of the input voltage [2]. This is done by iteratively dividing the operating range in two until the voltage in question is

found. In our design, we implemented a new algorithm that has many advantages over the conventional algorithm. This new algorithm is optimized for near continuous signals. This is achieved by assuming that the next value will be close to the previous value, rather than searching the extreme values first. To realize this, we take a modified Delta converter methodology, where by rather than doing incremental changes in the direction of change, we do exponential incremental changes. A base-two power increment in rate of change is done until the voltage value is matched or passed. If the voltage is passed, the power of the exponent is reset to 0 and the voltages are reduced in the same iterative process (See Figure 2 for an example). This insures convergence provided that boundary conditions are appropriately handled. To deal with the boundary conditions, we simply do not allow the value of the register to overflow/underflow, but rather overflow/underflow conditions set the register to the Max/Min values of the register respectively. The convergence of this algorithm can be better understood, if one considers the following:

The device increments by

$2^0 + 2^1 + 2^2 \dots + 2^n$ where by the exponential term n increments by 1 for each iteration until

$$\sum_{k=0}^j 2^k > X \text{ or } \sum_{k=0}^j 2^k = X \quad (1)$$

where X is the desired value and " j " is the iteration limit (determined by the maximum exponent used, or when the conditions are met)

$$\text{if } \sum_{k=0}^j 2^k > X \quad (2)$$

then " k " is reset to 0 and the result is decremented by

$$\sum_{k=0}^j 2^k \text{ until } \sum_{k=0}^j 2^k < X \text{ or } \sum_{k=0}^j 2^k = X \quad (3)$$

$$\text{if } \sum_{k=0}^j 2^k < X \quad (4)$$

then the result is incremented as shown in formula 1.

Thus the entire behavior can be described as

$$\sum_{k=0}^j 2^k - \sum_{k=0}^l 2^k + \sum_{k=0}^m 2^k \dots \text{ where } j > l > m \dots \quad (5)$$

This sequence converges since each summations bounds must be smaller then the previous, and can never be negative thus eventually this upper bounds will be 0 and hence a finite end to the sequence. The worst case can be found if we consider formula 5 and note that the maximum number of iterations would be when " j ", " l ", " m ", etc would be maximum numbers.

Thus we would have

$$\sum_{k=1}^p n \text{ cycles where } p \text{ is the resolution in bits.}$$

An additional cycle is required in our design for the sample and hold to fetch the data. Hence the real time is

$$1 + \sum_{k=1}^p n \text{ cycles} \quad (6)$$

For 6 bits this is 22 cycles.

Thus, we see that this algorithm leads to solutions to the voltage in a best case of one cycle (when the voltage hasn't changed), and a worst case of 22 cycles for a 6-bit ADC. However, this worst case can be easily avoided, as it only occurs when the data changes rapidly and goes to extremes of the device. This can be seen if we consider the Pseudo code below describing the behavior of the converter.

Pseudo-code:

```
//initialize converter at midpoint
x=31

//y is some input voltage value (between 0-63)
//that is a function of clock cycle t
:Loop
t=t+1 //new clock cycle every iteration
y=Input Value[t] //Sample and hold
//upward cycle
while x<y do begin //until the value is found do...
if x<y then begin
n=1 //initialize increment to 1
while x<y do begin
x=x+n //increment comparison value by n
n=n*2 //increase size of increment by factor
t=t+1 //increment clock as
//every iteration takes a clock cycle
end //while statement

end //if statement
//downward cycle
if x>y then begin
n=1 //initialize increment to 1
while x>y do begin
x=x-n //decrement comparison value by n
n=n*2 //increase size of increment by factor
t=t+1 //increment clock as
//every iteration takes a clock cycle
```

```

end //while statement
end //if statement
end // main while statement
Goto Loop //repeat for all time

```

We see in the pseudo code that if the input “y” is equal to the search value “x” then we fall through all the code and iterate to the next fetch of data. If the data is only out one position either up or down, it only takes one cycle, as we only operate in one loop for an additional clock cycle.

Though this large variation leads to a worst-case overall performance, however, it has ideal best-case performance. In addition, the worst-case performance is still nearly 3 times better than a pure delta converter, which has a worst case of 63 cycles. While the data can be high frequency, it is the large rapid voltage changes that result in worst-case performance of this converter. To limit the impact of the worst-case scenario, we implemented the device asynchronously. This leads to a variable sampling rate. While a variable sampling rate is a non-typical feature of an analog to digital converter, it has several advantages over a typical fixed rate converter. This type of interface returns more data points when run under conditions where the data doesn’t change dramatically. If the data isn’t returned every clock cycle (in the case of suddenly changing voltages for example), the number of cycles can be used to reconstruct data points using any number of interpolation techniques.

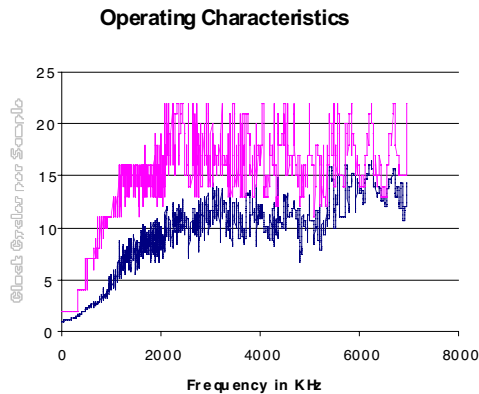


Figure 2. The upper graph reflects worst-case sampling time in clock cycles, the bottom graph indicates the average sampling time in clock cycles, and both graphs are for a full amplitude sine wave at the frequencies indicated.

In order to better understand the behavior of this converter, the algorithm was implemented in an object-oriented language. This program was then used to simulate the performance for a given input signal type and frequency. In Figure 2 is the average number of cycles required to find a data point for a

given frequency of a sine wave that operated over the full range of the converter (lower plot). Also plotted is the worst case time (upper plot). This occurs at the most rapidly changing voltage point (See Figure 2). Under these conditions we see that the worst case performance is better than the conventional algorithm up to a frequency of approximately 1.8 MHz (assuming a 125 MHz clock speed) and then the sampling rate is lower than the standard 8 cycles per sample of a conventional Successive Approximation ADC of this resolution. However, if we examine Chart 2 we observe that for signals that are half the amplitude, but same frequency, that the converter maintains the average sampling time to less than 8 cycles per sample up to approximately 3.6 MHz, and never exceeds a worst case of 16 cycles, which can be calculated from formula 6 assuming an initial value of 5 for “p”. From this we can determine that if the data points never shift more than 32 points in a sample period that the worst sampling time will be 16 cycles, rather than 22 cycles (see Figure 3).

Operating Characteristics

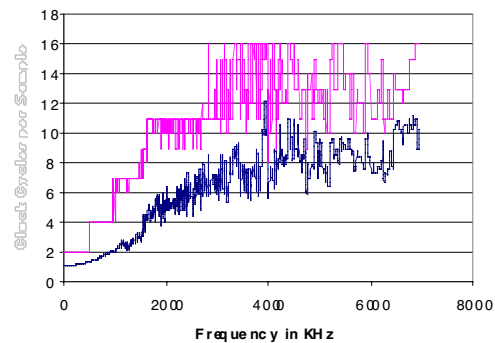


Figure 3. The upper graph reflects worst-case sampling time in clock cycles, the bottom graph indicates the average sampling time in clock cycles, and both graphs are for a half of max amplitude sine wave at the frequencies indicated.

3. Application to BIST

The intended purpose of this device is to test on chip signal generators. With this architecture, we can utilize the asynchronous interface as a means to determine, to a functional degree, if the data is valid. A long duration of the asynchronous data line being low indicates that the converter is having difficulty converging on the signal. This would be the case of a rapidly changing signal, such as a very noisy source. If the converter sets the signal line high on every cycle this indicates that the data isn’t changing at all, and is characteristic of a dc source. In this way, we can obtain a basic amount of information about a particular signal. Thus, we can use this line with a minimal amount of digital circuitry to determine an

immediate fail situation for our testing application. Meanwhile if we get positive results over regular intervals, we can determine the finer characteristics of our signal generator, via the digital output of the converter. Various waveforms also have a characteristic asynchronous ready rate, which is dependent upon the frequency and shape of the signal. Thus, to create a basic testing device, one only has to monitor the asynchronous line to determine if the signal is valid with this architecture. While this technique alone may not provide sufficient information, it can be utilized to reduce the complexity of circuitry required for further analysis of the signal. As amplitude and frequency information is reflected in the modulation rate of the Asynchronous data ready line.

4. Hardware Implementation

To implement our algorithm in hardware, we used a 3-bit counter, 3-bit decoder, and a 6-bit up-down accumulator. The 3 bit counter increments the power of the next step increment, while the decoder implements the exponential function (power) that is fed into the accumulator. The accumulator stores the previous value and increments the stored value by the amount fed into it. When an overflow/underflow occurs, the accumulator value is set to 111111 or 000000 respectively.

This Register controls the values that are compared to the input via a DAC. The feedback from the 2 comparators dictates whether the counter increments, or holds, and whether the accumulator adds or subtracts the value that is received from the

decoder. It should be noted that when the counter goes into a hold mode, all the data lines on the decoder drop to 0 and the accumulator is set so that no data change occurs. The hold mode line is activated when the data point is between the two DAC output lines. The operation of the accumulator is determined by the state of the upper DAC line in relation to the input. If the input is below the current DAC value, a subtraction operation occurs. This is achieved by connecting the top comparator to the accumulator's subtraction line. By using this methodology, a minimal amount of digital logic is required and enables this device to operate as a low power converter.

During the operation of the proposed circuit, the location of a data value forces the asynchronous data ready line, this in turn causes the sample and hold circuit to fetch a new value. Within one cycle, the converter determines if the value is the same, and within 2 cycles, it determines if the value has changed either up or down one bit level. This leads to the sampling rate of one half the clock rate of the circuit (what we refer to as the ideal operation rate for this converter). If the data point moved more than one bit level either up or down, one has to use the algorithm to determine the number of cycles required. Basic operation of this circuit can be seen in Figure 4. The two DAC output lines are successively adjusted until the Sample signal is between them (see Figure 4 between 600 and 650ns). When the signal is between them the Data Ready Line is set active (upper waveform of Figure 4). This intern causes a new sample to be fetched (Figure 4 at 670ns).

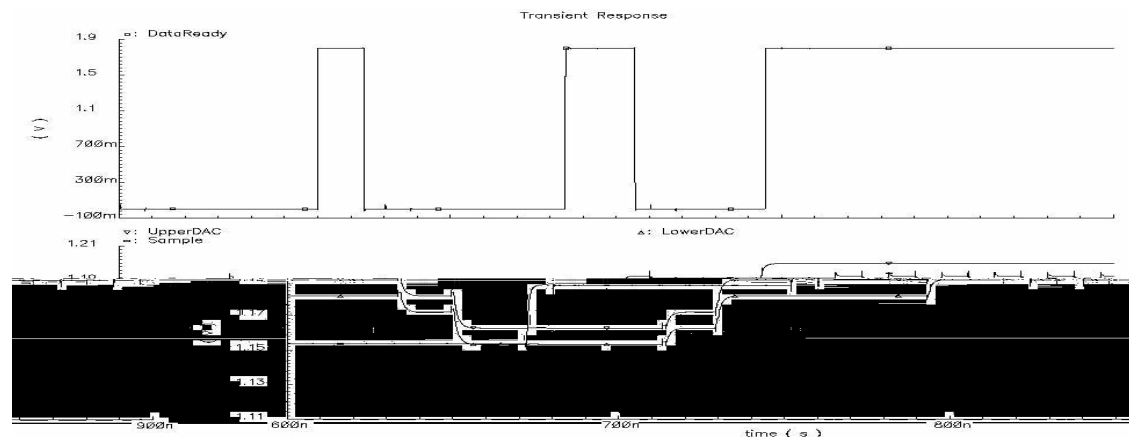


Figure 4 – This figure demonstrates the Successive Approximation ADC synchronizing to the input sample and then fetching a new sample. Valid data is indicated by the top waveform, which is the Asynchronous data ready line. On the bottom you can see the 2 DAC outputs syncing up with the input sample.

4. Results

The Successive Approximation ADC was able to achieve 125MSamples/s and its performance is limited by the speed of the feedback loop. Due to gate propagation delay, the counter doesn't respond fast enough to the hold signal and overshoots if the speed is increased. This device performed well with a low power consumption of 3.8mW and with a maximum sampling rate of 125 MSamples/s. It should be noted that due to the asynchronous nature of the device, the sampling rate could slow down to as low as 5.7 MSamples per second under worst-case conditions.

While this converter is slow, its efficiency is higher than other converters. When operating on low frequency signals, the converter consumes approximately 3.8mW, versus 30mW for the device built by C. Lin et. al; in addition this device has greater accuracy, with better INL and DNL performance [6]. While we recognize that the processes are different as well as the supply voltages, and operating speed the power consumption is still considerably lower even accounting for these differences. Table I below summarizes the performance of this device.

Table I. SAR ADC performance summary

	C. Lin et. al	Current
Technology	0.35 μ CMOS	0.18 μ CMOS
Max. Sampling Rate	250 Msample	5.7-125 MSamples
Resolution	6 Bits	6 Bits
INL	0.65 LSB	0.3 LSB
DNL	1 LSB	0.3 LSB
Power Consumption	30 mW	3.8 mW
Power Supply	3.3V	1.8 V

5. Conclusion

In summary, we presented a Successive Approximation Analog to Digital converter utilizing a new algorithm that tracks the input signal, and can

produce output values every cycle. The performance of 125 MSamples/s, and low power consumption, of just 3.8mW, of this converter makes it suitable for the analysis of low frequency signals. In addition this device was developed with a minimal amount of additional circuitry as we were able to reuse a preexisting DAC structure.

6. References

- [1] F. Corsi, C. Marzocca, G. Matarrese "Defining A BIST-Oriented signature for Mixed-Signal Devices" IEEE Proceedings of Southwest Symposium of Mixed Signal Design, pp.202-207, 2003.
- [2] A. Rossi and G. Fucili, "Non-redundant successive approximation register for A/D converters" Electron. Lett., Vol. 32, NO. 12, pp.1055-1057, June 1996.
- [3] Paul. G.A. Jespers. *Integrated Converters D to A and A to D Architectures, Analysis, and Simulation*. Oxford: New York, 2001.
- [4] Alfi Moscovici. *High Speed A/D Converters Understanding Data Converters Through Spice*. Kluwer Academic Publishers: Massachusetts, 2001.
- [5] R. Jacob Baker, Harry W. Li, and David E. Boyce. *CMOS Circuit Design, Layout, and Simulation*. IEEE Press: New York, 1998.
- [6] C. Lin and B. Liu. "A New Successive Approximation Architecture for Low-Power Low-Cost CMOS A/D Converter," IEEE Journal of Solid-State Circuits, Vol. 38, NO. 1, pp.54-62, January 2003.
- [7] Robert M. Senger, Paul M. Walsh, and Jerome Le Ny. "A 150 Msamples/s Folding and Current Mode Interpolating ADC in 0.35 μ m CMOS." EECS 598-02 Analog to Digital Integrated Circuits, 1-7, 2002.
- [8] B. Vaz, N. Paulino, J. Goes, R. Costa, R. Tavares, and a. Steiger-Garcia, "Design of Low-Voltage CMOS Pipelined ADC's using 1 pico-Joule of Energy per Conversion" IEEE Journal of Solid-State Circuits, No.1, pp.921-924, 2002.