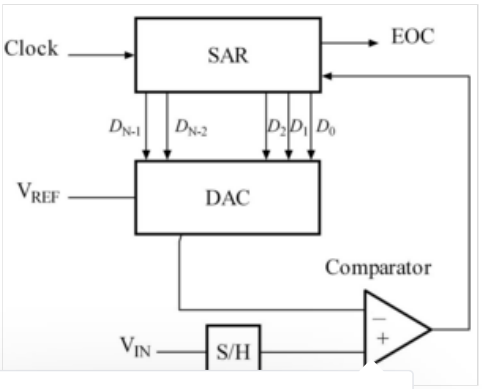
Successive-approximation ADC

(Redirected from Successive approximation ADC)

A **successive-approximation ADC** is a type of analog-to-digital converter (ADC) that converts a continuous analog waveform into a discrete digital representation using a binary search through all possible quantization levels before finally converging upon a digital output for each conversion.



Algorithm

The successive-approximation analog-to-digital converter circuit typically consists of four chief subcircuits:

1. A sample-and-hold circuit to acquire the input voltage V_{in}.
2. An analog voltage comparator that compares V_{in} to the output of the intern DAC and outputs the result of the comparison to the successive-approxima register (SAR).
3. A successive-approximation register subcircuit designed to supply an approximate digital code of V_{in} to the internal DAC.
4. An internal reference DAC that, for comparison with V_{ref}, supplies the comparator with an analog voltage equal to the digital code output of the SAR_{in}.

The successive approximation register is initialized so that the most significa equal to a digital 1. This code is fed into the DAC, which then supplies the anal this digital code ($V_{\text{ref}}/2$) into the comparator circuit for comparison with th voltage. If this analog voltage exceeds V_{in}, then the comparator causes the S bit; otherwise, the bit is left as 1. Then the next bit is set to 1 and the sa continuing this binary search until every bit in the SAR has been tested. The the digital approximation of the sampled input voltage and is finally output b end of the conversion (EOC).

Mathematically, let $V_{\text{in}} = xV_{\text{ref}}$, so x in $[-1,\ 1]$ is the normalized input voltage. to approximately digitize x to an accuracy of $1/2^n$. The algorithm proceeds as foll

1. Initial approximation $x_0 = 0$.
2. i^{th} approximation $x_i = x_{i-1} - s(x_{i-1} - x)/2^i$, where, $s(x)$ is the signum func using mathematical induction that $|x_n - x| \leq 1/2^n$.

As shown in the above algorithm, a SAR ADC requires:

1. An input voltage source V_{in}.
2. A reference voltage source V_{ref} to normalize the input.
3. A DAC to convert the i^{th} approximation x_i to a voltage.
4. A comparator to perform the function $s(x_i - x)$ by comparing the DAC's voltage with the input voltage.
5. A register to store the output of the comparator and apply $x_{i-1} - s(x_{i-1} - x)/2^i$.

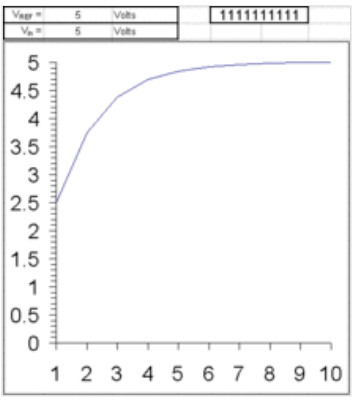
Examples

Example 1: The ten steps to converting an analog input to 10 bit digital, using successive approximation, are shown here for all voltages from 5 V to 0 V in 0.1 V iterations. Since the reference voltage is 5 V, when the input voltage is also 5 V, all bits are set. As the voltage is decreased to 4.9 V, only some of the least significant bits are cleared. The MSB will remain set until the input is one half the reference voltage, 2.5 V.

In electronics, an **analog-to-digital converter** is a system that converts an analog signal, such as a sound picked up by a microphone or light entering a digital camera, into a digital signal. An ADC may also provide an isolated measurement such as an electronic device that converts an analog input voltage or current to a digital number representing the magnitude of the voltage or current. Typically the digital output is a two's complement binary number that is proportional to the input, but there are other possibilities.

The binary weights assigned to each bit, starting with the MSB, are 2.5, 1.25, 0.625, 0.3125, 0.15625, 0.078125, 0.0390625, 0.01953125, 0.009765625, 0.0048828125. All of these add up to 4.9951171875, meaning binary 1111111111, or one LSB less than 5.

When the analog input is being compared to the internal DAC output, it effectively is being compared to each of these binary weights, starting with the 2.5 V and either keeping it or clearing it as a result. Then by adding the next weight to the previous result, comparing again, and repeating until all the bits and their weights have been compared to the input, the result, a binary number representing the analog input, is found.

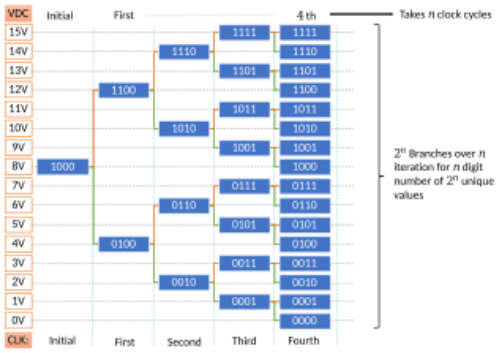


Operation of successive-approximation ADC as input voltage falls from 5 to 0 V. Iterations on the *x* axis. Approximation value on the *y* axis.

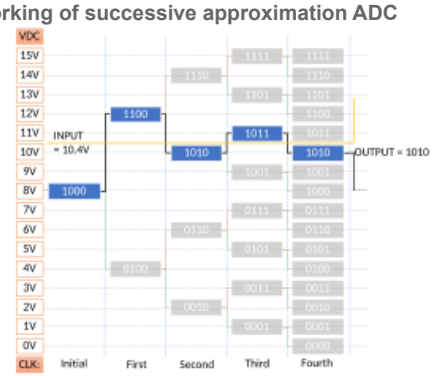
Example 2: The working of a 4-bit successive approximation ADC is illustrated below. The MSB is initially set to 1 whereas the remaining digits are set to zero. If the input voltage is lower than the value stored in the register, on the next clock cycle, the register changes its value to that illustrated in the figure by following the green line. If the input voltage is higher, then on the next clock cycle, the register changes its value to that illustrated in the figure by following the red line. The simplified structure of this type of ADC that acts on 2^n volts range can be expressed as an algorithm:

1. Initialize register with MSB set to 1 and all other values set to zero.
2. In n-th clock cycle, if voltage is higher than digital equivalent voltage of the number in register, the (n+1)-th digit from the left is set to 1. If the voltage were lower than digital equivalent voltage, then n-th digit from left is set to zero and the next digit is set to 1. To perform a conversion, an N-bit ADC requires N such clock cycles excluding the initial state.

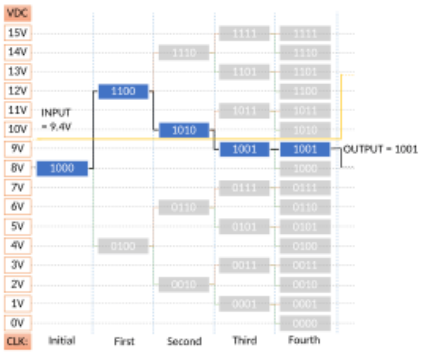
Working of successive approximation ADC



Setup where output values of the ADC are arranged in a grid, vertical axis corresponding to voltage. It is a 4-bit ADC that measures input voltages from 0V to 15V.



Previously established setup where an input voltage of 10.4V is provided.



Previously established setup where an input voltage of 9.4V is provided.

The successive approximation ADC can be alternatively explained by first uniformly assigning each digital output to corresponding ranges as shown. It can be seen that the algorithm essentially divides the voltage range into two regions and checks which of the two regions the input voltage belongs to. Successive steps involve taking the identified region from before and further dividing the region into two and continuing identification. This occurs until all possible choices of digital representations are exhausted, leaving behind an identified region that corresponds to only one of the digital representations.

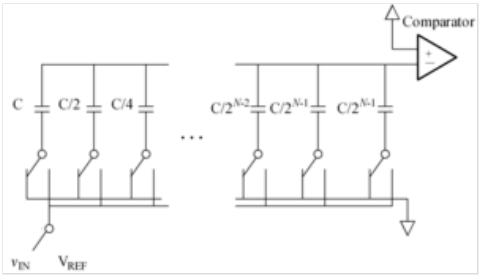
Variants

- *Counter type ADC*: The D to A converter can be easily turned around to provide the inverse function A to D conversion. The principle is to adjust the DAC's input code until the DAC's output comes within $\pm 1/2$ LSB to the analog input which is to be converted to binary digital form.
- *Servo tracking ADC*: It is an improved version of a counting ADC. The circuit consists of an up-down counter with the comparator controlling the direction of the count. The analog output of the DAC is compared with the analog input. If the input is greater than the DAC output signal, the output of the comparator goes high and the counter is caused to count up. The tracking ADC has the advantage of being simple. The disadvantage, however, is the time needed to stabilize as a new conversion value is directly proportional to the rate at which the analog signal changes.

Charge-redistribution successive-approximation ADC

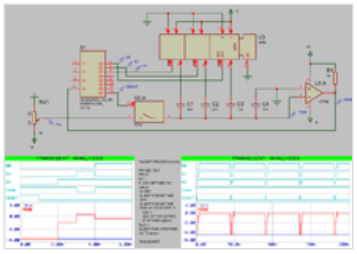
One of the most common implementations of the successive-approximation ADC, the *charge-redistribution* successive-approximation ADC, uses a charge-scaling DAC. The charge-scaling DAC simply consists of an array of individually switched binary-weighted capacitors. The amount of charge upon each capacitor in the array is used to perform the aforementioned binary

search in conjunction with a comparator internal to the DAC and the successive-approximation register.



Charge-scaling DAC

1. The capacitor array is completely discharged to the offset voltage of the comparator, V_{OS}. This step provides automatic offset cancellation (i.e. the offset voltage represents nothing but dead charge, which can't be juggled by the capacitors).
2. All of the capacitors within the array are switched to the input signal V_{in}. The capacitors now have a charge equal to their respective capacitance times the input voltage minus the offset voltage upon each of them.
3. The capacitors are then switched so that this charge is applied across the comparator input, creating a comparator input voltage equal to $-V_{\text{in}}$.
4. The actual conversion process proceeds. First, the MSB capacitor is switched to V_{ref}, which corresponds to the full-scale range of the ADC. Due to the binary-weighting of the array, the MSB capacitor forms a 1:1 charge divider with the rest of the array. Thus, the input voltage to the comparator is now $-V_{\text{in}} + V_{\text{ref}}/2$. Subsequently, if V_{in} is greater than $V_{\text{ref}}/2$, then the comparator outputs a digital 1 as the MSB, otherwise it outputs a digital 0 as the MSB. Each capacitor is tested in the same manner until the comparator input voltage converges to the offset voltage, or at least as close as possible given the resolution of the DAC.


3 bits simulation of a capacitive ADC

Use with non-ideal analog circuits

When implemented as an analog circuit – where the value of each successive bit is not perfectly 2^N (e.g. 1.1, 2.12, 4.05, 8.01, etc.) – a successive-approximation approach might not output the ideal value because the binary search algorithm incorrectly removes what it believes to be half of the values the unknown input cannot be. Depending on the difference between actual and ideal performance, the maximal error can easily exceed several LSBs, especially as the error between the actual and ideal 2^N becomes large for one or more bits. Since the actual input is unknown, it is therefore very important that accuracy of the analog circuit used to implement a SAR ADC be very close to the ideal 2^N values; otherwise, it cannot guarantee a best match search.

See also

- Quantization noise
- Digital-to-analog converter

References

Further reading

- *CMOS Circuit Design, Layout, and Simulation, 3rd Edition*; R. J. Baker; Wiley-IEEE; 1208 pages; 2010; ISBN 978-0-470-88132-3
- *Data Conversion Handbook*; Analog Devices; Newnes; 976 pages; 2004; ISBN 978-0750678414

External links

- Understanding SAR ADCs: Their Architecture and Comparison with Other ADCs (https://pdfserv.maximintegrated.com/en/an/AN1080.pdf) - Maxim
- Choose the right A/D converter for your application (https://www.ti.com/europe/downloads/Choose%20the%20right%20data%20converter%20for%20your%20application.pdf) - TI