e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

SIGMA DELTA DAC USING VHDL-AMS

Mr. S.A. Utage*1

*1Assistant Professor, Department Of Electronics Engineering, Walchand Institute Of Technology, Solapur, Maharashtra, India.

DOI : https://www.doi.org/10.56726/IRJMETS59810

ABSTRACT

Sigma Delta Digital to analog converters (DACs) convert a binary number into a voltage directly proportional to the value of the binary number. A variety of applications use DACs including waveform generators and programmable voltage sources. This paper describes a Delta-Sigma DAC implemented in a FPGA. The only external circuitry required is a low pass filter comprised of just one resistor and one capacitor. Internal resource requirements are also minimal. The speed and flexible output structure of the FPGAs make them ideal for this application.

Keywords: FPGA, DAC, Sigma Delta, VHDL, VLSI.

I. INTRODUCTION

In past years Sigma-Delta modulators have been a very popular means for A/D and D/A conversion, mainly because of the low impact of circuit imperfections on the behavior of the converter, due to the noise shaping. Research in this area has mainly concentrated on architectural exploration and the impact of circuit imperfections. Due to the noise-shaping loop is entirely in the digital domain for a Sigma-Delta D/A converter. But mainly in the analog domain for the A/D converter, a greater part of the research was devoted to the latter.

II. METHODOLOGY

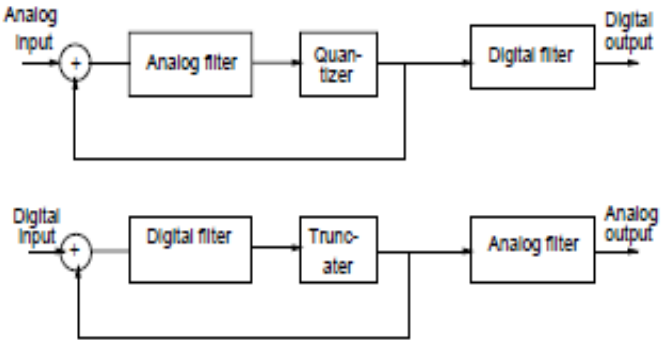

Figure 1: A/D and D/A Sigma-Delta converters

Above fig.1 represents the digital domain for Analog to Digital converter. The Sigma-Delta D/A converter can be split up into different components, this is shown in Figure2. For each of the different components a behavioral model is constructed using VHDL_AMS.

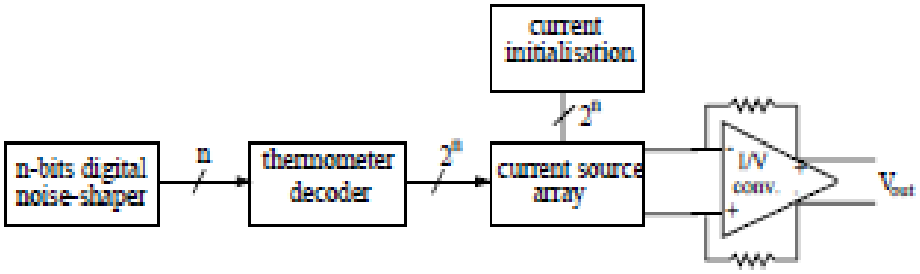

Figure 2: Block scheme of the Sigma-Delta D/A converter.

1) The digital noise-shaper

This component is the heart of the Sigma-Delta modulator. It is a feedback loop consisting of a truncator and a noise-shaping filter Figure 3. The truncator just strips the least significant bits of the k-bit input word, thus

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

performing a coarse quantization of the signal. In the model the digital words are replaced by their integer equivalents to facilitate the modeling of the filter and the adders. The truncator now performs an integer division and the adder is the normal addition.. The order of the filter can then be selected by means of a *case...when* statement. Alternatively, the adders and truncator can be put in separate asynchronous processes. These are not sensitive to the clock, but to the output of the preceding block. On the event of a changing input, the process becomes active and will use the new input value to calculate the new output. For higher-order filters overload can occur, which can be prevented by the use of a limiter. Modeling of such a limiter can be done using an *if...then* statement.

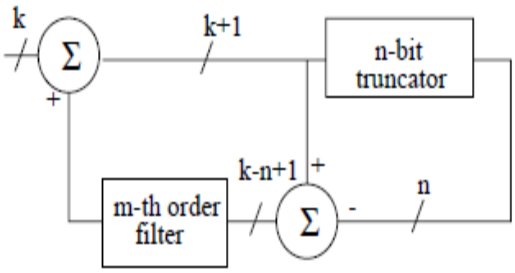

Figure 3: n-bit digital noise-shaper

2) Thermometer decoder

The decoder is the block which converts the bit stream from the noise shaper into a thermometer code. This can be done in the standard way, but also dynamic element matching (DEM) techniques can be used. In VHDL-AMS a counter comes down to storing an index in memory, which can easily be done using a variable. Since the noise-shaper output is an integer, the indices are too, which means normal addition and modulo statements can be used to model the counter. The output of the decoder is a *bit-vector* of length *nl*.

3) Current source D/A converter

The D/A converter in a multi-bit Sigma-Delta, needs to be fast, but has relatively few output bits. This makes it suitable to be realized using a current source array. By using DEM techniques, clever layout and scrambling of the order of current sources, this can be achieved. In general it will cost more area and power to do so. A system like this can be modeled using the scheme depicted in Figure 4.

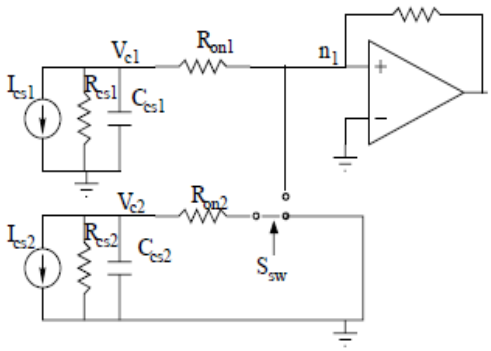

Figure 4: Lumped model of D/A converter current sources.

The current *Ics*2*; Rcs*2*; Ron*2 and *Ccs*2 are the lumped current and impedances of the part of the array which is switched on at that moment. The current *Ics*1*; Rcs*2*; Ron*2 and *Ccs*2 are the lumped current and impedances of the part of the circuit that was already on. system is by an array of (nearly) identical current cells. This can be done by using the generate statement. This approach is of course much more straightforward, but is also slower, because the number of quantities used is larger.

4) The current to voltage converter

If the output of the current-source array would feed into a normal resistor, its voltage would be sweeped from the highest output to the lowest output, which would increase the settling time. To prevent this from happening, a virtual ground is created using an opamp The opamp model used is depicted in Figure 5.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

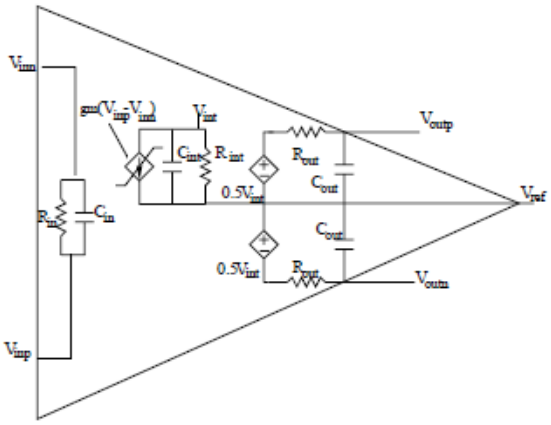

Figure 5: Model for the opamp.

5) Top Level Entity

Below is a top-level schematic diagram Figure 6 of a typical DAC implementation. As shown in this diagram, the inputs include reset and clock signals, in addition to the binary number bus. DACoutDrvr (output pin) drives an external low-pass filter. V_{OUT} can be set from 0V to V_{CCO}, where V_{CCO} is the supply voltage applied to the FPGA I/O bank driving the resistor-capacitor filter.

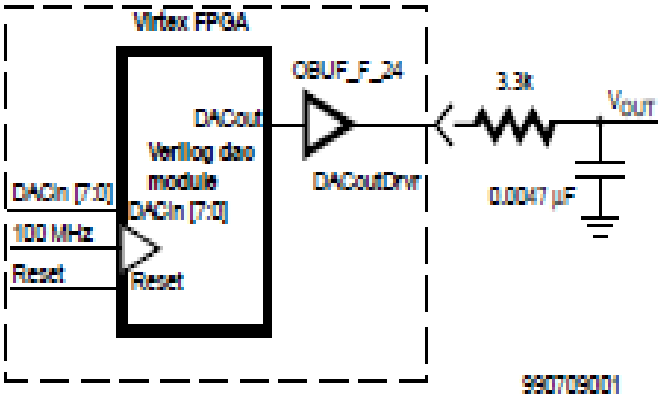

Figure 6: Top level DAC Implementation.

The classic current summing digital to analog converter uses matched resistors to convert a binary number to a corresponding voltage level. This technique works well for high-speed DACs when the binary number is up to ten bits wide. However, it is difficult to maintain accuracy over a range of temperatures as the number of bits increases.

6) Delta-Sigma Architecture

A Delta-Sigma DAC uses digital techniques. Consequently, it is impervious to temperature change, and may be implemented in programmable logic. Delta-Sigma DACs are actually high-speed single-bit DACs. Using digital feedback, a string of pulses is generated. The average duty cycle of the pulse string is proportional to the value of the binary input. The analog signal is created by passing the pulse string through an analog low-pass filter. The basic architecture, implementation, and trade-offs are covered. Delta-Sigma DACs are used extensively in audio applications. They are suited for low frequency applications that require relatively high accuracy.As is standard practice, the DAC binary input in this implementation is an unsigned number with zero representing the lowest voltage level. The analog voltage output is also positive only. A zero on the input produces zero volts at the output. All ones on the input cause the output to nearly reach V_{CCO}. For AC signals, the positive bias on the analog signal can be removed with capacitive coupling to the load.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

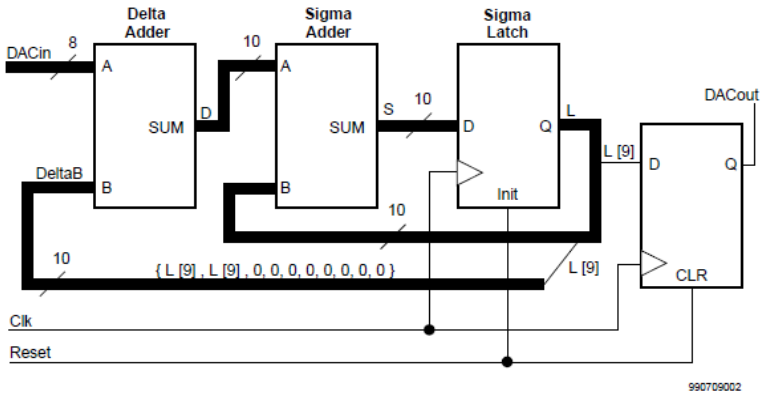

Figure 7: Delta-Sigma DAC Internal Block Diagram

Above Figure 7 is a block diagram of a Delta-Sigma DAC. The width of the binary input in the implementation described below is configurable. For simplicity, the block diagram depicts a DAC with an 8-bit binary input. The term "Delta-Sigma" refers to the arithmetic difference and sum, respectively. In this implementation, binary adders are used to create both the difference and the sum. Although the inputs to the Delta adder are unsigned, the outputs of both adders are considered signed numbers. The Delta Adder calculates the difference between the DAC input and the current DAC output, represented as a binary number. Since the DAC output is a single bit, it is "all or nothing"; i.e., either all zeroes or all ones. As shown in Figure 8, the difference will result when adding the input to a value created by concatenating two copies of the most significant bit of the Sigma Latch with all zeros. This also compensates for the fact that DAC in is unsigned. The Sigma Adder sums its previous output, held in Sigma Latch, with the current output of the Delta Adder. In most cases, the Delta adder is optimized out when the high level design is synthesized. This is because all bits on either the A or B inputs are zero, so A and B are simply merged, rather than added. As noted below, the DAC input can be widened by one bit to allow the full analog range of 0V to V CCO . In this case, the Delta adder is needed. The interface to HDL module DAC in Figure 7 includes one output and three input signals as defined in Table 1. All signals are active high.

Table 1: DAC Interface Signals

Signal	Direction	Description
DACout	Output	Pulse string that drives the external low pass filter
DACin	Input	Digital input bus. Value must be setup to the positive edge of CLK. For high-speed operation, DACin should be sourced from a pipeline register that is clocked with CLK. For full resolution, each DACin value must be averaged over 2 (MSBI+1) clocks, so DACin should change only on intervals of 2 (MSBI+1) clock cycles.
CLK	Input	Positive edge clock for the SigmaLatch and the output D flip-flop.
Reset	Input	Reset initializes the SigmaLatch and the output D flip-flop. In this implementation, SigmaLatch is initialized to a value that corresponds to 0V in and 0V out. If DACin starts at zero, there is no discontinuity.

For the implementation in Figure 7, the output voltage (V_{OUT}) as a function of the DAC input may be expressed as follows:

$$V_{OUT} = (DACin/(2\ (MSBI+1)\ )) \times V_{CCO} \text{ Volts}$$

For example, for an 8-bit DAC (MSBI = 7) the lowest V_{OUT} is 0V when DACin is 0. The highest V_{OUT} is 255/256 V_{CCO} volts when DACin is FF_{16} .

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

For some applications, it may be important for V_{OUT} to swing through the entire voltage range: 0V to V_{CCO} (rail-to-rail). This is accomplished by increasing the DACin bus width by one bit and leaving all other bus widths the same. For an 8-bit DAC with an input value of 256,

$$V_{OUT} = V_{CCO}.$$

Note that all DACin values greater than 256 are illegal and should not be used. It is often advantageous to use a high-frequency clock. The desired clock may be faster than that which

can be practically sourced externally.

III. PROJECT DEVELOPMENT STAGES

The DAC can be implemented in a single VHDL- AMS module.

A) DESIGN METHODOLOGY:

The process starts from the Continuous-Time Delta-Sigma modulator system specification and simulation at the top-level abstraction. Then, a top-down approach is applied, replacing certain ideal blocks with non-ideal models or even their SPICE representation to obtain greater accuracy. Each building-block of the Continuous-Time Delta-Sigma modulator is modeled using a VHDL-AMS language. The VHDL-AMS codes are compiled using the ADVance MS compilator. The resulting behavioral models are converted in functional VHDL-AMS blocks to be used in the CADENCE environment. This design methodology allows the mixing of behavioral models and transistor level models in the same simulation environment (CADENCE), improving the simulation speed. The modulator output is obtained both in the time domain and in the frequency domain (through FFT). These simulations run with ADVance MS (ADMS) under CADENCE environment. Figure 8 shows the design methodology using both ADVance MS and CADENCE.

B) THE DESIGN PROCESS

A typical digital design flow is as follows:

Specification
Architecture
RTL-Coding
RTL-Verification
Synthesis
Backend

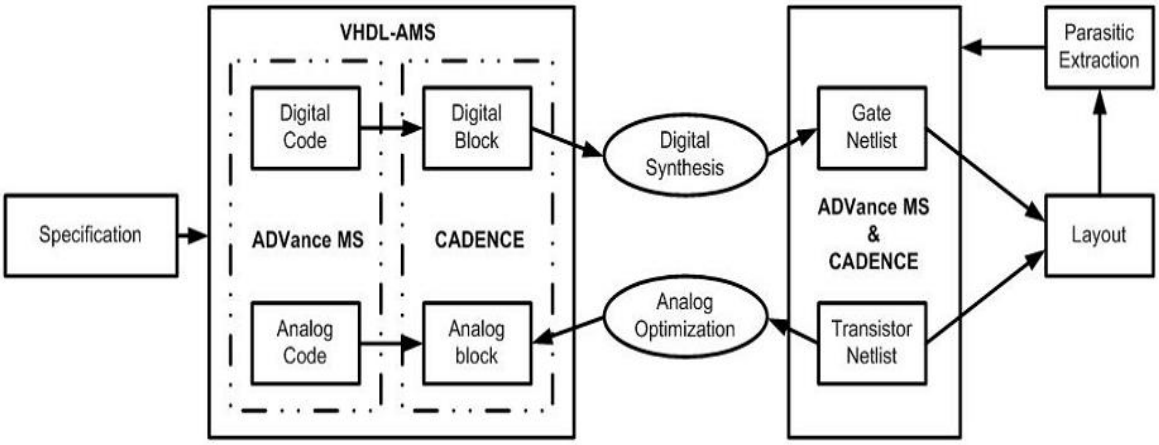

Figure 8: Design Methodology

Tape Out to Foundry to get end product….a wafer with repeated number of identical ICs. All modern digital designs start with a designer writing a hardware description of the IC (using HDL or Hardware Description Language) in Verilog/VHDL. A Verilog or VHDL program essentially describes the hardware (logic gates, Flip-Flops, counters etc) and the interconnect of the circuit blocks and the functionality. Various CAD tools are available to synthesize a circuit based on the HDL. The most widely used synthesis tools come from two CAD companies.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

Without going into details, we can say that the VHDL, can be called as the "C" of the VLSI industry. VHDL stands for "VHSIC Hardware Definition Language", where VHSIC stands for "Very High Speed Integrated Circuit". This languages is used to design the circuits at a high-level, in two ways. It can either be a behavioural description, which describes what the circuit is supposed to do, or a structural description, which describes what the circuit is made of. There are other languages for describing circuits, such as Verilog, which work in a similar fashion. Both forms of description are then used to generate a very low-level description that actually spells out how all this is to be fabricated on the silicon chips. This will result in the manufacture of the intended IC. A typical analog design flow is as follows:

In case of analog design, the flow changes somewhat.

Specifications
Architecture
Circuit-Design
SPICE-Simulation
Layout
Parametric Extraction / Back Annotation

Final-Design
Tape-Out-to-foundry.

While digital design is highly automated now, very small portion of analog design can be automated. There is a hardware description language called AHDL but is not widely used as it does not accurately give us the behavioral model of the circuit because of the complexity of the effects of parasitic on the analog behavior of the circuit. Many analog chips are what are termed as “flat” or non-hierarchical designs. This is true for small transistor count chips such as an operational amplifier, or a filter or a power management chip. For more complex analog chips such as data converters, the design is done at a transistor level, building up to a cell level, then a block level and then integrated at a chip level. Not many CAD tools are available for analog design even today and thus analog design remains the most useful simulation tool for analog as well as digital design.

IV. CONCLUSION

The Delta-Sigma DAC is an example of how high speed FPGAs may be used in mixed-signal systems to minimize the number of components. The speed and density of the FPGAs makes them ideal for a wide range of analog signal generating and processing applications. VHDL-AMS model of complete Sigma-Delta D/A converter can be developed. This model can be used to perform high-level analysis of the impact of system non-idealities, such as opamp slewing and current source mismatch. The results of the analysis can be used to find anoptimum set of building block specification yielding optimum system performance

V. REFERENCES

[1] J. Candy and G. C. Temes, “Oversampling methods for A/D and D/A conversion,” in Oversampling 16 converters, pp. 1–25, IEEE Press, 1992.

[2] “URL:http://www.eda.org/vhdl-ams/.”

[3] V. Peluso, A. Marques, M. Steyaert, and W. Sansen, “Optimal Parameters for Single Loop 16 Modulators,” in Proc. of the Int.Symp. Circuits Syst., (Hong Kong), pp. 57–60, 1997.

[4] G. C. Temes, S. Shu, and R. Schreier, “Architectures for 16 DAC’s,” in Oversampling 16 converters, pp. 309–322, IEEE Press, 1992.

[5] “URL: http://www.Xilinx.com36 th Design Automation Conference New Orleans, June 21- 25, 1999