www.czasopisma.pan.pl PAN www.journals.pan.pl
INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2023, VOL. 69, NO. 3, PP. 425-430
Manuscript received May 25, 2023; revised July, 2023. DOI: 10.24425/ijet.2023.144379

# ADC Emulation on FPGA

Huma Tabassum, Krishna Prathik BV, and Sujatha S Hiremath

***Abstract*—Analog-to-Digital Converters (ADCs) are devices that transform analog signals into digital signals and are used in various applications such as audio recording, data acquisition, and measurement systems [1]. Prior to the development of actual chip, there is a need for prototyping, testing and verifying the performance of ADCs in different scenarios. Analog macros cannot be tested on an FPGA. In order to ensure the macros function properly, the emulation of the ADC is done first. This is a digital module and can be designed in System Verilog. This paper demonstrates the design of the module on FPGA for Analog to Digital Converter (ADC) emulation. The emulation is done specific to the ADC macro which has programmable resolutions of 12/10/8/6 bit.**

***Keywords*—ADC; resolution; FPGA; sampling; conversion**

## I. Introduction

THE emulation of the ADC on FPGA is done to mimic the behaviour of the ADC on FPGA. This is useful when testing the entire microcontroller on FPGA. The module developed in this paper can be used in place of the actual ADC, for testing the functionality, since the ADC is an analog macro and cannot be implemented on FPGA. The ADC which is emulated operates in different phases which include sampling of data and conversion of analog to digital data. The data can be output either in serial or parallel fashion. The ADC can also be configured to operate in different resolution modes. The resolution is set by the controller.

## II. Theory

The module that performs the ADC emulation is designed according to Fig. 1. The signals required by the ADC are obtained first. The ADC emulation module controller is then designed taking into consideration the different modes of operation of the ADC and the number of clock cycles required for each mode. There are different types of ADCs such as Continuous Time Pipelined ADC, which is an architecture that combine pipelining with continuous time operation [2]. The ADC considered in our design is a typical Successive Approximation Register ADC.

### *A. ADC Controller Interface*

An ADC controller interfaces between the analog input sources and the ADC, providing configuration, control, and synchronization functionalities [3]. The ADC and the controller are interfaced through several signals. See Fig. 2

- clock signal: The clock signal is required by the ADC and is output by the controller.

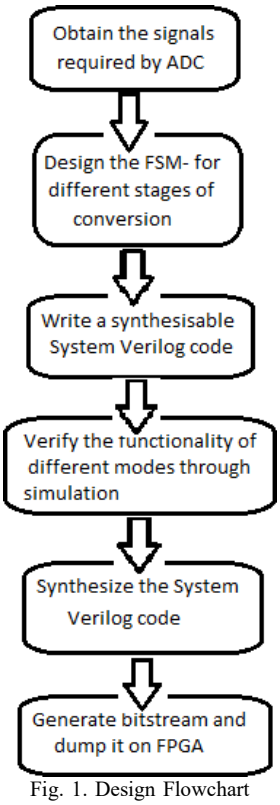


Fig. 1. Design Flowchart

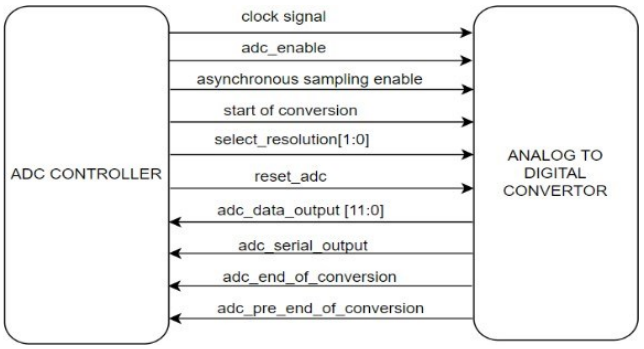


Fig. 2. ADC-Controller Interface

Authors are with RV College of Engineering, India (e-mail: humatabassum812@gmail.com, bvkrishnap2001@gmail.com, sujathah@rvce.edu.in).

- adc_enable: When the enable signal goes high, the ADC is enabled and switched on.
- asynchronous_sampling_enable: The ADC operates in two modes. This signal decides which mode the ADC operates in. This signal has to be held constant during sampling.
- start_of_conversion: This signal indicates the start of conversion. The controller checks the start of conversion signal on the positive edge of the clock. It enters the conversion phase on the immediate posedge after the start of conversion is detected.
- select_resolution: The 2 bit signal selects either 6,8,10 or 12 bit resolution.
- reset_adc: This active high signal is used to reset the adc.
- adc_data_output: This is the parallel digital data output of the ADC.
- adc_serial_output: This signal is the serial output of the digital data.
- adc_end_of_conversion: This signal depicts the end of conversion.
- adc_pre_end_of_conversion: This signal is driven high one clock cycle before the end of conversion signal.

### *B. Finite State Machine Design*

The ADC Emulation controller module is designed based on a finite state machine. A finite state machine (FSM) is a mathematical model used to represent systems that exhibit distinct states and transition between them based on inputs. It consists of a finite number of states, transitions, and events [4]. The different states depict the phases through which the ADC moves when it is required to convert data. See Fig. 3.

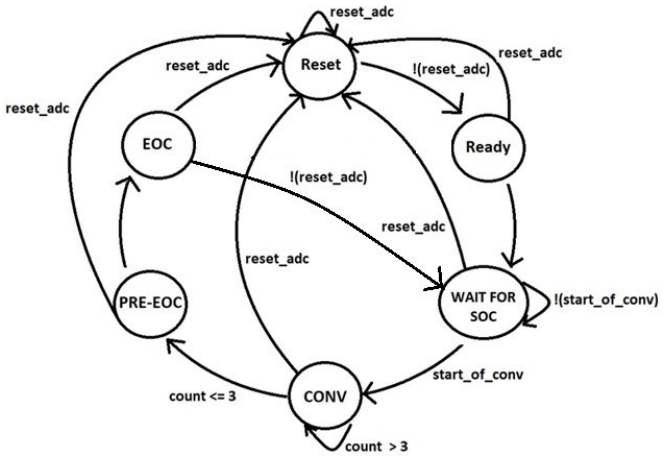


Fig. 3. FSM Design

Transition from one state to another occurs based on signals recieved.

- RESET STATE: Reset phase can be entered with logic 'H' on reset when module is enabled. In the reset state,all activities are aborted, and the serial and the parallel outputs are driven low.
- READY STATE: The next state that the ADC enters is the ready state. In this state, the adc pre_end_of _conversion and the adc end_of_conversion signals are set high simultaneously on the first posedge of clock after the deassertion of reset signal. The ADC enters the sampling mode and starts to respond to control signals.
- WAIT FOR SOC: This phase is also referred toas the sampling phase. During this phase, the ADC samples the data. The end of sampling is indicated by the start of conversion signal going high. The adc pre_end_of _conversion and the adc end_of_conversion signals remain low.
- CONVERT STATE: ADC enters the conversion phase on first posedge of clock after it receives high on thestart of conversion signal. The conversion phase lasts for N-cycles, where N is the resolution of the ADC.
- PRE END OF CONVERSION STATE: During this state, the ADC pre_end_of_conversion signal goes high. This signal has to go high 1 clock cycle before the adc end _of_conversion is set high. ADC remains in this state only for one clock cycle.
- END OF CONVERSION STATE: In this state, the adc end_of_conversion is set high. This indicates that the ADC has completed the conversion of analog data. Onthe same posedge of clock when this signal goes high,the parallel data outputs are available on the data output bus.

### *C. Data Output modes of the ADC*

- Parallel mode: In this mode, the data is output in parallel form i.e. all the bits of data are output at the same time [2]. The data is output only after the entire conversion cycle completes. The data is output on the negative edge of the clock. The size of adc data_output depends on the resolution set. If the resolution is set to 12 bits, the adc data_output is of 12 bits.
- Serial mode: In this mode, the data is output in serial form i.e. the bits are output one after the other [2]. As soon as a bit is obtained after conversion, it is output.The bits are made available on the adc serial_output and data is output from the most significant bit first to the least significant bit in the end.

### *D. Sampling modes of the ADC*

- Instantaneous Sampling Mode: This mode is enabled when the asynchronous sampling enable is high. In this mode the end of sampling defined by deassertion of asynchronous trigger asynchronous sample. In this mode there is a one clock cycle gap between the start of conversion and when the start of conversion signal goes high. The number of clock cycles taken to convert the data depends on the resolution set. If the resolution set is 'N', the number of clock cycles taken is 'N+1' [5]. For serial output mode, the first two clock cycles after startof conversion goes high, no output is produced. The next 'N' cycles will produce the output.

- Typical SAR Mode of Operation This mode is enabled when the asynchronous sampling enable is low. Here, the sampling ends on the negative edge of the clock after the positive edge on which the start of conversion was detected high. This is the typical mode of operation. In this mode the start of conversion and end of samplingare defined by the same start of conversion signal unlike the instantaneous sampling mode where a seperate signal was present to signify the end of sampling.

### *E. Resolution of the ADC*

Analog-to-Digital Converter resolution is used to describe or measure the performance of an ADC [5]. The resolution of an A/D converter (ADC) is specified in bits. The resolution of the converter indicates the number of discrete values it can produce over the range of analog values. A resolution is the smallest voltage increment corresponding to a 1 LSB change [6]. It is an important specification for ADC because it determines the smallest analog input signal that an ADC can resolve. It determines the magnitude of the quantization error and therefore gives the maximum possible average signal-to-noise ratio for an ideal ADC without oversampling [6].

TABLE I
Select Resolution of the ADC

| select resolution | number of bits of resolution |
|---|---|
| 2'b11 | 12 |
| 2'b10 | 10 |
| 2'b01 | 8 |
| 2'b00 | 6 |

Table I highlights the value of the select resolution signal which will produce the required resolution. The ADC used in this project can provide 4 resolution values. The resolution can be set using the select resolution signal.

## III. Simulation and Results

### *A. Outputs for different resolutions*

The simulation of the System Verilog code was carriedout to obtain the waveforms. Different modes of operation, resolution and data output formats were simulated.

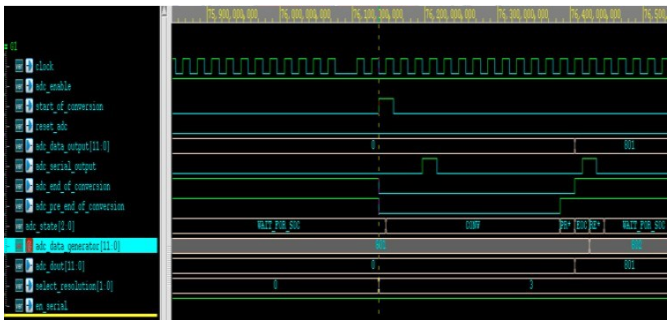

Fig. 4. Output waveform for 12-bit resolution of ADC

The different resolutions of ADC include: 12 bit, 10bit, 8 bit and 6 bit. The above waveform in the figure 4 shows the output for 12 bit resolution of ADC. After the ADC is enabled, enough time is given for the warmup process. The start of conversion is the input signal depicting the end of sampling and start of conversion of data. On the first posedge of clock after the start of conversion signal goes high, the pre end of conversion and the end of conversion signals go low. Since ADC is of 12 bit resolution, it takes 12 clock cyclesfor data conversion as seen in the waveform. The conversion starts on the immediate next posedge of clock after start of conversion is detected. The pre end of conversion and the end of conversion signals go high after 11 and 12 clock cycles respectively. The parallel data output appears as soon as the conversion ends, i.e. on the same posedge of clock when end of conversion signal goes high.

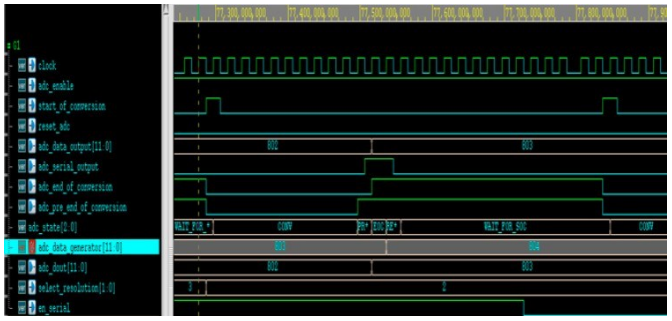

Fig. 5. Output waveform for 10 bit resolution of ADC

Figure 5 shows the output for 10 bit resolution of ADC. Here, the ADC takes 10 clock cycles for conversion of data. The parallel data output appears after 10 clock cycles and the data is continuously driven on the bus until the next converted data is available. To set ADC resolution as 10, the selectresolution bits are set to value 2. The serial data output also appears because the enable serial bit is set to 1.

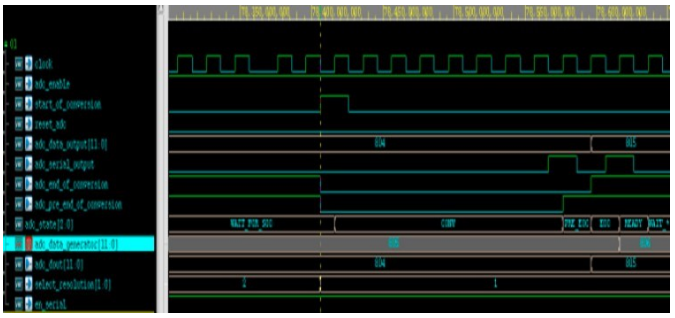

Fig. 6. Output waveform for 8 bit resolution of ADC

The waveform in Figure 6 shows the output for 8 bit resolution of ADC. Here, the ADC takes 8 clock cycles for conversion of data. The parallel data output appears after 8 clock cycles and the data is continuously driven on the bus until the next converted data is available. In order to set the resolution of ADC as 8 bits, the select resolution bits are set to value 1. The serial data output also appears because the enable serial bit has been set to 1. Serial output starts appearing 1 clock cycle after the start of conversion. The last converted bit appears one clock cycle after the end of conversion bit is set high.

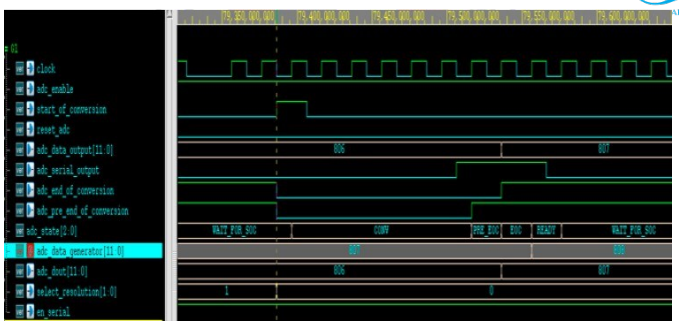

Fig. 7. Output waveform for 6 bit resolution of ADC

The waveform in Figure 7 shows the output for 6 bit resolution of ADC. Here, the ADC takes 6 clock cycles for conversion of data. The number of start of conversion bits shown in the waveform is one, hence one conversion occurs. The parallel data output appears after 6 clock cycles and the data is continuously driven on the bus until the next converted data is available. In order to set the resolution of ADC as 6 bits, the select resolution bits are set to value 0.

### *B. Normal and Burst mode of operation*

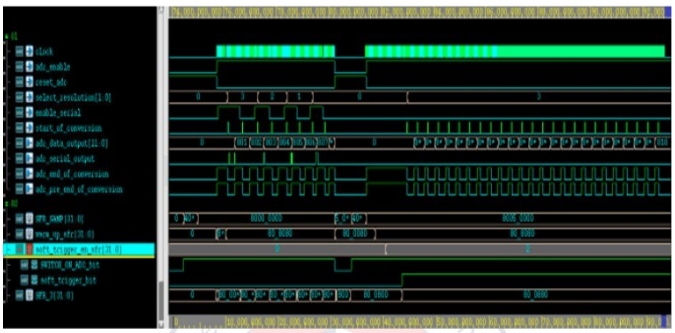

Fig. 8. Normal mode of operation of ADC

In normal mode of operation, the start of conversion bit has to be set manually for each conversion. Whereas in burst mode, few software triggers have to be enabled which cause a series of start of conversion bits to appear one after the other. The waveform in figure 8 shows the ADC working in normal mode for the initial half period and in burst mode for the next half period. ADC is disabled for a while between the two transitions.

The figure 9 shows the burst mode of data conversion. In burst mode of operation, a series of start of conversion signals appear one after the other. The minimum time difference between the two start of conversion signals is 12 clock cycles. In this mode, the resolution is always set to 12 by assigning the select resolution bits a value 3. Serial data is disabled and only parallel data appears on the bus. A series of converted data outputs can be observed in the waveform along with the end of conversion and the pre end of conversion signals.

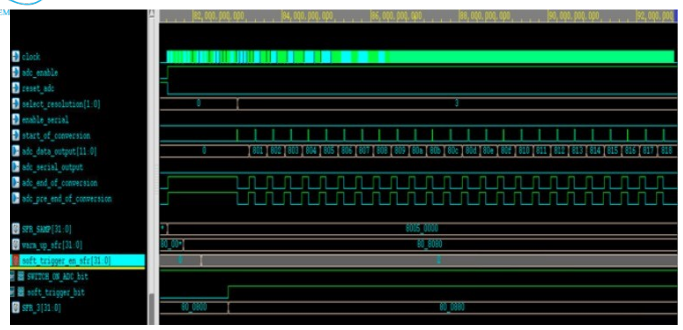

Fig. 9. Burst mode of operation of ADC

### *C. RTL Schematic of the design*

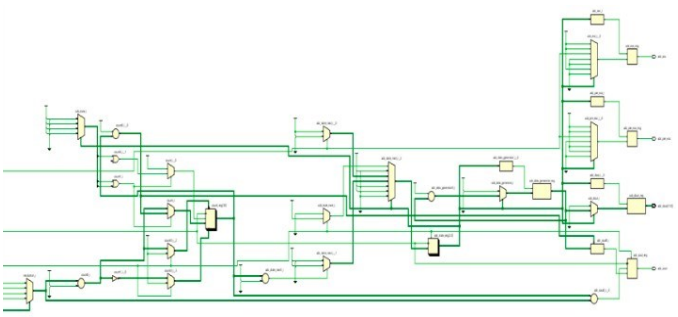

Fig. 10. RTL Schematic of the Design

The RTL schematic provides a higher-level abstraction of the design, illustrating the behavior and interconnections of the registers and combinational logic [6]. It serves as a valuable tool for comprehending and visualizing the design's structure and functionality prior to synthesis and implementation on the target device. As seen in the above figure 10, every block within the design represents a module or component, such as multiplexers, gates, and flops. The interconnections between these blocks represent the signals and buses responsible for carrying data and control information [6].

### *D. Synthesis of the design*

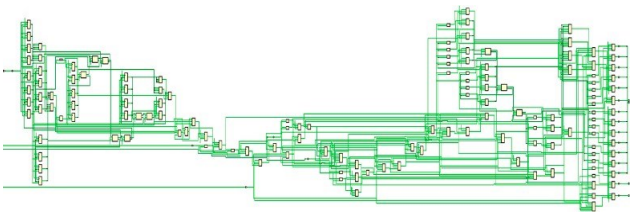

Fig. 11. Synthesized Design

The synthesized design is the outcome of applying the synthesis process to the RTL representation of a digital design. The synthesized design reflects the design at a reduced level of abstraction in comparison to the RTL description [7]. As depicted in Figure 11, the synthesized design encompasses gates, flip-flops, and various digital components, constituting

the logical representation of the design. It incorporates details regarding the interconnections among these elements and their respective logical functions. Synthesizing the design enables the evaluation of its estimated performance characteristics and facilitates the analysis of logic utilization [8].

### *E. Logic Utilization of the design*

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| LUT | 38 | 41000 | 0.09 |
| FF | 41 | 82000 | 0.05 |
| IO | 20 | 300 | 6.67 |

Fig. 12. Logic Utilization of the Design

After running synthesis of the design, a project summary is obtained. The resource utilization of the design is shown in the figure 12. The Look Up Table (LUT) serves as the fundamental component of an FPGA, possessing the capability to implement any logic function involving N Boolean variables [9]. The LUT utilization of the design is 38, the number of flip flops needed is 41 and the input output pin utilisation is 20. The LUTs and the flip flops used in teh design form 1 percent of the total LUT count and the input output pin utilisation is 7 percent of the total availability

### *F. Power Utilization of the design*

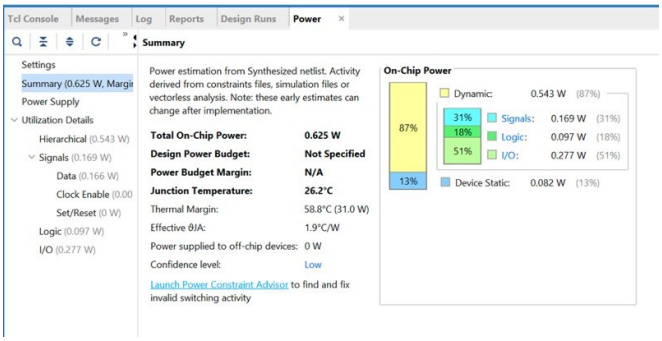


Fig. 13. Power Utilization of the Design

The figure 13 shows the power utilization of the design. Dynamic power refers to the power consumed by the de-sign while undergoing switching activities [10]. The dynamic power utilization of the design is 87 percent. Static power is the power consumed by the design in the absence of any switching activity. The static power utilization of the design is 13 percent. The dynamic power utilized is 0.543W and the static power utilised is 0.082W.

## IV. Conclusions

The aim of our work was to design a controller on FPGA for ADC emulation. This involved designing a Finite State Machine in System Verilog whose states represent the various stages through which the ADC passes during conversion of analog input to digital. The FSM has 6 states- Reset, Ready, Wait for Start of conversion (Sampling state), Conversion state, Pre-end of conversion state and the End of conversion state.

The functioning of the design is verified from the sim-ulated output waveforms. The waveforms show the outputsfor different configurations, different resolutions, normal and burst mode of operation. The RTL schematic of the design is obtained which is followed by synthesis using Vivado tool. Artix 7 was chosen as the board. The logic utilization and power consumption information is obtained. Fininally, the bitstream is generated and the code is dumped on FPGA to verify its functionality. The LUT utilization is 38, the number of flip flops needed is 41 and the input output pin utilisation is 20. The dynamic power utilization is 0.543W which is 87% of the total power utilized and the static power utilization is 0.082W which contributes to 13% of the total power.

As a future scope of the work, the controller design can be extended to include different types of ADCs. The code can be enhanced to reduce the amount of logic cells used. The power consumption of the design can also be reduced by using low power techniques.

## Acknowledgment

We would like to express our gratitude to each and everyone, who provided valuable input, insights, and assistance at every stage of the project. Their contributions were crucial to the success of this work, and we are deeply grateful for their hard work and dedication.

## References

[1] S. Bashir, S. Ali, S. Ahmed and V. Kakkar, ”Analog-to-digital converters: A comparative study and performance analysis,” 2016 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, 2016, pp. 999-1001, https://doi.org/10.1109/CCAA.2016.7813861

[2] S. Pavan and H. Shibata, ”Continuous-Time Pipelined Analog-to-Digital Converters: A Mini-Tutorial,” in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 68, no. 3, pp. 810-815, March 2021, https://doi.org/10.1109/TCSII.2020.3048850

[3] Y. Hu, K. Yan and W. Jing, ”Design of ADC Control Module in a MCU,” 2008 Fourth International Conference on Natural Computation, Jinan, China, 2008, pp. 133-137, https://doi.org/10.1109/ICNC.2008.204

[4] C. Sapsanis, M. Villemur and A. G. Andreou, ”Real Number Modeling of a SAR ADC behavior using SystemVerilog,” 2022 18th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD), Villasimius, Italy, 2022, pp. 1-4, https://doi.org/10.1109/SMACD55068.2022.9816309

[5] G. G. E. Gielen, L. Hernandez and P. Rombouts, ”Time-Encoding Analog-to-Digital Converters: Bridging the Analog Gap to Advanced Digital CMOS-Part 1: Basic Principles,” in IEEE Solid-State Cir- cuits Magazine, vol. 12, no. 2, pp. 47-55, Spring 2020, https://doi.org/10.1109/MSSC.2020.2987536

[6] Jinyuan Wu, Sten Hansen and Zonghan Shi, ”ADC and TDC implemented using FPGA,” 2007 IEEE Nuclear Science Symposium Conference Record, Honolulu, HI, USA, 2007, pp. 281-286, https://doi.org/10.1109/NSS-MIC.2007.4436331

[7] P. H. W. Leong, ”Recent Trends in FPGA Architectures and Applications,” 4th IEEE International Symposium on Electronic Design, Test and Applications (delta 2008), Hong Kong, China, 2008, pp. 137-141, https://doi.org/10.1109/DELTA.2008.14

[8] J. E. Istiyanto, ”A VHDL-based ADC on FPGA,” International Conference on Instrumentation, Communication, Information Technology, and Biomedical Engineering 2009, Bandung, Indonesia, 2009, pp. 1-3, https://doi.org/10.1109/ICICI-BME.2009.5417248

[9] H. Homulle, S. Visser and E. Charbon, "A Cryogenic 1 GSa/s, Soft- Core FPGA ADC for Quantum Computing Applications," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 63, no. 11, pp. 1854-1865, Nov. 2016, https://doi.org/10.1109/TCSI.2016.2599927

[10] Bin Le, T. W. Rondeau, J. H. Reed and C. W. Bostian, "Analog-to- digital converters," in IEEE Signal Processing Magazine, vol. 22, no. 6, pp. 69-77, Nov. 2005, https://doi.org/10.1109/MSP.2005.1550190

[11] J. -H. Tsai, Y. -J. Chen, M. -H. Shen and P. -C. Huang, "A 1-V, 8b, 40MS/s, 113µW charge-recycling SAR ADC with a 14µW asynchronous controller," 2011 Symposium on VLSI Circuits - Digest of Technical Papers, Kyoto, Japan, 2011, pp. 264-265.

[12] E. Monmasson, L. Idkhajine and M. W. Naouar, "FPGA-based Controllers," in IEEE Industrial Electronics Magazine, vol. 5, no. 1, pp. 14-26, March 2011, https://doi.org/10.1109/MIE.2011.940250

[13] V. Giannini, P. Nuzzo, V. Chironi, A. Baschirotto, G. Van der Plas and J. Craninckx, "An 820µW 9b 40MS/s Noise-Tolerant Dynamic-SAR ADC in 90nm Digital CMOS," 2008 IEEE International Solid-State Circuits Conference - Digest of Technical Papers, San Francisco, CA, USA, 2008, pp. 238-610, https://doi.org/10.1109/ISSCC.2008.4523145

[14] J. Bergeron, "Writing testbenches using SystemVerilog," Springer Science Business Media, 2013

[15] S. Sutherland, S. Davidmann, and P. Flake, "SystemVerilog for Design Second Edition: A Guide to Using SystemVerilog for Hardware Design and Modeling," Springer Science Business Media, 2006

[16] J. Estarán, S. Almonacil et.al "Sub-Baudrate Sampling at DAC and ADC: Toward 200G per Lane IM/DD Systems," J. Lightwave Technol. 37, 1536-1542 (2019).

[17] H. -Y. Tai, Y. -S. Hu, H. -W. Chen and H. -S. Chen, "11.2 A 0.85fJ/conversion-step 10b 200kS/s subranging SAR ADC in 40nm CMOS," 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), San Francisco, CA, USA, 2014, pp. 196-197, https://doi.org/10.1109/ISSCC.2014.6757397

[18] Keaveney, Martin and McMahon, Anthony et.al, "The development of advanced verification environments using system verilog," Institution of Engineering and Technology,2014.