**CINEFORUM**
ISSN: 0009-7039
Vol. 66. No. S6, 2026 (Special Issue 6)

# Design and Implementation of a Complete-Folding-Based Optimal-Combination Calibrated Segmented DAC for Static-Linearity Enhancement

**Dikonda Manasa[1*], Mr A. Venugopal[2*], Dr.V. Sapthagiri[3*]**

[1]Student of MTech in VLSI System Design, Department of Electronic Communication and Engineering, Sree Dattha Institute of Engineering and Sciences, Hyderabad, Telangana, India.
[2]Assistant Professor, Department of Electronic Communication and Engineering, Sree Dattha Institute of Engineering and Sciences, Hyderabad, Telangana, India.
[3]Assistant Professor, Department of Electronic Communication and Engineering, Sree Dattha Group of Institutions, Hyderabad, Telangana, India.
**Corresponding Author**
Email Id: manasadikonda31@gmail.com

**Abstract-**This work proposes a digitally assisted segmented digital-to-analog converter (DAC) architecture that improves static linearity through a Complete Folding (CF)-based Optimal Combination Algorithm (OCA).The proposed converter operates in two phases. During startup calibration, mismatch-affected unit elements in the most-significant-bit (MSB) section are compared, sorted, recursively folded, and reorganized into compensated binary-weighted groups. The resulting element-to-group assignment is stored in a mapping memory.The digital input word is split into MSB and least-significant-bit (LSB) portions during normal operation; the calibrated MSB mapping is read from memory, a switch-control encoder activates the selected physical elements, and the LSB sub-DAC provides fine conversion. By moving the computationally intensive sorting and grouping process to startup, the run-time conversion path remains simple and deterministic. The architecture is suitable for synthesizable RTL development and can be evaluated using transfer-curve, differential-nonlinearity, integral-nonlinearity, monotonicity, area, power, and timing metrics. The proposed method offers a practical way to obtain improved static accuracy from imperfect unit elements without relying solely on device upsizing or continuous dynamic element scrambling.

**Keywords—**segmented DAC, Complete Folding, optimal combination algorithm, digital calibration, INL, DNL, element mismatch, Verilog RTL.

## 1. Introduction


**Figure 1. Basic digital-to-analog conversion from a binary input to an analog output waveform.**

Digital-to-analog converters are essential mixed-signal building blocks translate digitally processed information into continuous voltage or current levels. They are used in communication transmitters, waveform generators, instrumentation, sensor interfaces, programmable references, motor-control systems, biomedical electronics, and embedded control platforms. As converter resolution increases, the output step corresponding to one least significant bit becomes small, and therefore the permissible errorcaused by device mismatch, parasitic loading, and switching nonidealities becomes increasingly restrictive.[1]-[3].

Among the available converter structures, segmented DACs provide a useful compromise between a fully binary-weighted architecture and a fully unary architecture. The MSB region is commonly implemented with unit elements or thermometer-like control to preserve monotonicity and reduce large transition errors, while the LSB region is realized with a compact binary or ladder-based structure. However, random mismatch among the MSB unit elements still creates code-dependent output deviations. These deviations directly affect Differential nonlinearity (DNL), Integral nonlinearity (INL), monotonicity, and accuracy of the complete transfer characteristic.[5],[6],[7].

Conventional remedies include larger devices, careful common-centroid layout, trimming, and dynamic element matching. Device enlargement improves matching statistically but increases silicon area and parasitic

capacitance.[4] Dynamic element matching can average mismatch over time, yet it does not necessarily optimize every static output code. For precision and low-to-medium-speed applications, a foreground calibration method that explicitly rearranges the available unit elements can provide a more direct improvement in static linearity.

The work presented here adopts a Complete Folding-based Optimal Combination Algorithm. The method first establishes the relative ordering of mismatch-affected MSB elements and then repeatedly combines complementary low- and high-valued elements. The folding operation is performed hierarchically so that pairs, groups of four, groups of eight, and larger binary-weighted groups are progressively formed. The final assignment is preserved in mapping memory and is used during normal conversion. In this way, the analog array is not assumed to be ideal; instead, digital calibration reorganizes imperfect elements into more accurate effective weights.[8]-[10].

## 2. Background and Related Work

### *2.1 DAC Architectures and Segmentation*

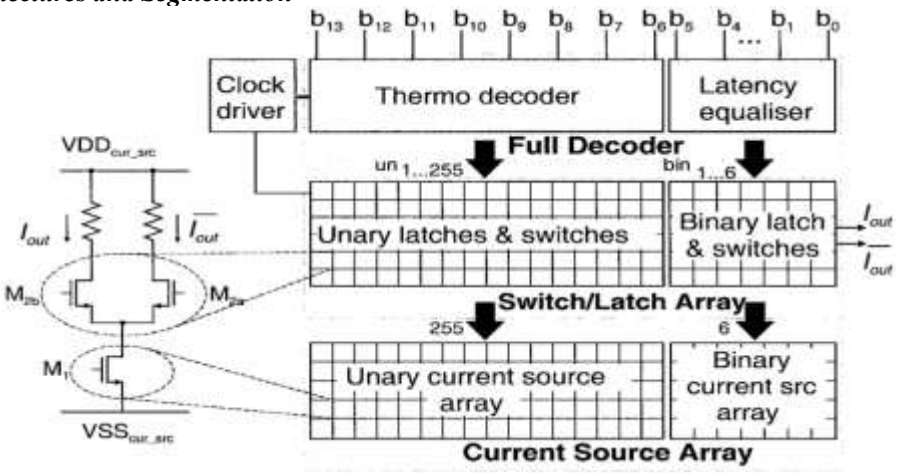


**Figure 2. Conventional segmented current-steering DAC with unary MSB and binary LSB sections.**

Binary-weighted DACs offer compact decoding but are highly sensitive to the accuracy of large-weight elements. Resistor-string DACs provide monotonic output levels but require an exponentially increasing number of taps. R–2R ladders simplify resistor ratios, whereas current-steering DACs are attractive for high-speed operation. Segmentation combines these benefits by assigning coarse conversion to a mismatch-sensitive MSB array and fine conversion to a compact LSB sub-DAC.

### *2.2 Static Linearity*

DNL measures a deviation of each actual code step from the ideal one-LSB step. INL measures the deviation of full transfer characteristic from an ideal straight line after offset and gain normalization. Large negative DNL can produce missing codes or nonmonotonic behavior, while large INL creates cumulative transfer error and waveform distortion.

$$\mathrm{DNL(k)} = \frac{\mathrm{V(k+1) - V(k)}}{\mathrm{V_{LSB,ideal}}} - 1$$

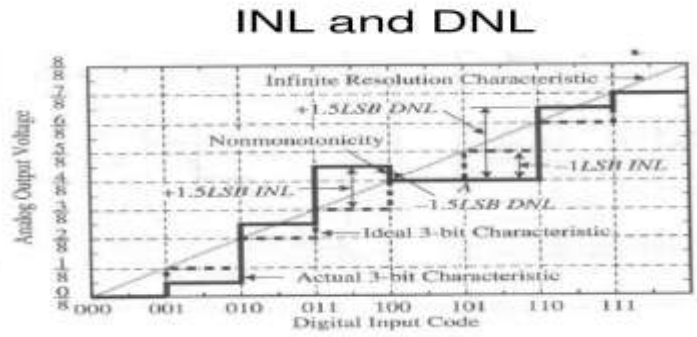


**Figure 3. Illustration of integral nonlinearity, differential nonlinearity, and nonmonotonic behavior in a DAC transfer characteristic.**

$$\mathbf{INL(k)} = \frac{\mathbf{V(k) - V_{ideal}DNL(k)}}{\mathbf{V_{LSB,ideal}}}$$

### 2.3 Mismatch Mitigation and OCA Methods

Mismatch mitigation can be performed through layout optimization, trimming, averaging, or calibration. Optimal Combination Algorithms differ from dynamic element matching because they use measured ordering information to generate a fixed, improved element arrangement. Representative approaches include switching-sequence post-

adjustment, bisection, sorting-and-grouping, and Complete Folding. CF is particularly effective because it applies compensation recursively at multiple grouping levels rather than only once.

### 2.4 Research Gap

Many published OCA studies emphasize statistical or behavioral analysis, while comparatively fewer describe a modular RTL architecture that covers comparator handshaking, sorting, exact CF construction, mapping storage, and calibrated run-time decoding. The present work addresses this gap by defining a complete two-phase digital architecture suitable for verification and synthesis.

## 3. Proposed CF-Based Calibrated Segmented DAC

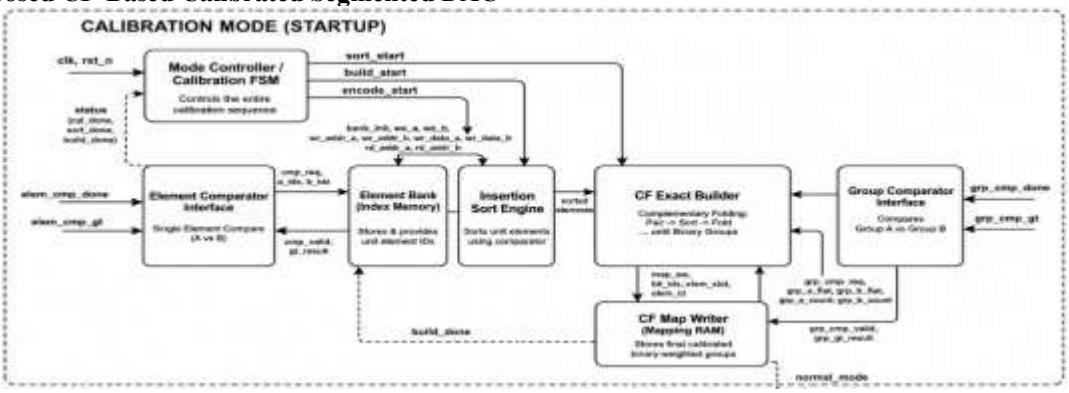


**Figure 4. Startup calibration architecture for element comparison, insertion sorting, Complete Folding, and calibrated mapping storage**

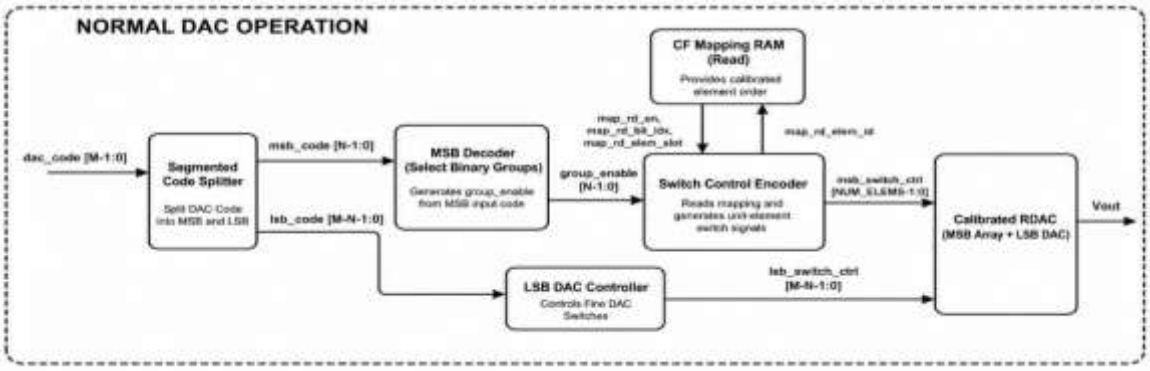


**Figure 5. Normal operating architecture using segmented code splitting, calibrated MSB mapping, switch encoding, and the LSB sub-DAC.**

### 3.1 System Overview

The proposed system is divided into startup calibration and normal DAC operation. During startup, the MSB unit elements are initialized, compared, sorted, folded, and mapped. After the calibration-done signal is asserted, the system enters normal mode, where incoming digital codes are converted using the stored optimized mapping. The two-phase organization concentrates complexity in the foreground calibration interval and maintains a lightweight conversion path afterward.

| Module | Phase | Primary Function |
|---|---|---|
| Mode Controller / Calibration FSM | Startup | Sequences initialization, sorting, folding, map writing, and transition to normal mode. |
| Element Comparator Interface | Startup | Requests and captures pairwise element comparisons. |
| Element Bank | Startup | Stores element indices and the evolving sorted order. |
| Insertion Sort Engine | Startup | Generates an ascending element order using compare-and-shift operations. |
| Group Comparator Interface | Startup | Compares folded groups during higher CF stages. |

| | | |
|---|---|---|
| CF Exact Builder | Startup | Forms complementary pairs and recursively produces calibrated binary-weighted groups. |
| CF Mapping RAM | Both | Stores physical element indices associated with each effective calibrated group. |
| Segmented Code Splitter | Normal | Separates the input word into MSB and LSB fields. |
| MSB Decoder / Switch Encoder | Normal | Reads the calibrated map and activates the required MSB elements. |
| LSB DAC Controller | Normal | Controls the fine DAC section. |

### *3.2 Segmented Input and Output Model*

For an N-bit DAC with M calibrated MSB bits and L = N-M LSB bits, the input code D is represented as the concatenation of DMSB and DLSB. The normalized ideal output can be expressed as the weighted sum of coarse and fine contributions.

$$D = 2^{L}D_{MSB} + D_{LSB}$$

$$V_{out,ideal}(D) = V_{ref}\left(\frac{D}{2^{N}-1}\right)$$

In the proposed implementation, the MSB contribution is generated from calibrated physical groups selected by the mapping RAM, while the LSB contribution is generated by a conventional fine sub-DAC.

## 4. Startup Calibration Method

### 4.1 Element Characterization

Each nominally identical MSB unit element has an actual value ui = u0 + ei, where u_0 is the ideal unit value and ei is its mismatch error. The calibration hardware does not require a full numerical measurement of ei. It reauires only reliable greater
-than or less-than decisions between two elements or two groups.

$$\boldsymbol{u_i = u_0 + e_i, e_i \sim \mathcal{N}(0, \sigma_u^2)}$$

### *4.2 Insertion Sorting*

The insertion sort engine builds an ascending index sequence. At each iteration, a candidate element is compared with preceding sorted elements. If the candidate is smaller, stored indices are shifted and the candidate is inserted at the correct location. The resulting order x(1) ≤ x(2) ≤ ... ≤ x_(Q) is retained in the element bank.

### *4.3 Complete Folding*

Complete Folding begins from the sorted sequence. The smallest and largest elements are paired, followed by the second-smallest and second-largest, and so on. The same complementary principle is applied recursively to the sums of existing groups. At each stage, the formed groups are ranked and folded again until all effective binary-weighted groups required by the calibrated MSB DAC are obtained.

$$g_j^{(1)} = x_j + x_{Q+1-j} \qquad (2)$$

$$g_j^{(s+1)} = g_j^{(s)} + g_{R_s+1-j}^{(s)} \qquad (3)$$

This recursive compensation reduces the probability that an effective group is composed only of undersized or only of oversized elements. The final physical indices belonging to each group are written to the CF mapping RAM.

### 4.4 Calibration Complexity

The sorting and folding process requires multiple comparison cycles, but it is executed only during startup or on explicit recalibration. For Q elements, insertion sorting has a worst-case comparison count proportional to Q(Q−1)/2. The complete folding stage adds group comparisons across logarithmic grouping levels. The resulting startup overhead is traded for a simple mapping-based run-time datapath.

## 5. Normal DAC Operation

After calibration, the mode controller enables the normal conversion path. The incoming code is split into MSB and LSB components. The MSB decoder determines which calibrated groups are required, the mapping RAM returns the physical element identifiers, and the switch-control encoder generates the unit-selection vector. The LSB controller simultaneously generates the fine-section controls. The analog output is sum of both sub-DAC contributions.

$$V_{out}(D) = V_{MSB,cal}(D_{MSB}) + V_{LSB}(D_{LSB})$$

Because the CF computation has already been completed, each conversion requires only decoding, memory lookup, and switch activation. This preserves deterministic output behavior and avoids the continuous switching activity associated with dynamic element matching.

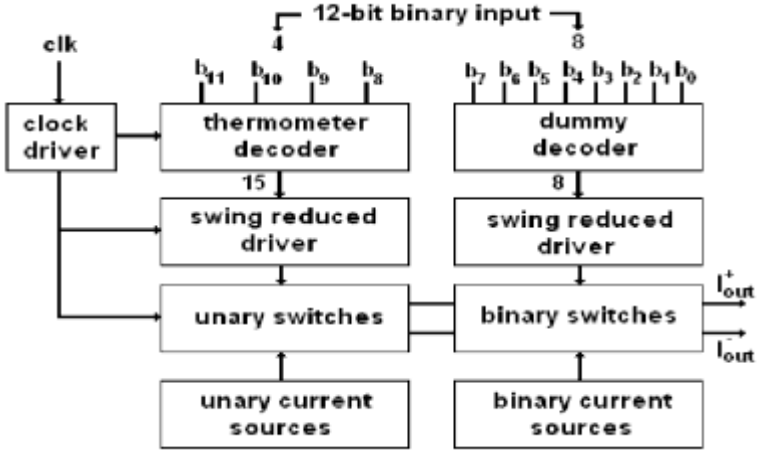


**Figure 6. Representative reduced-switch segmented DAC organization with thermometer-decoded coarse control and binary.**

## 6. RTL Implementation Strategy

### *6.1 Finite-State Control*

A synthesizable finite-state machine can use states such as IDLE, INIT_BANK, SORT_REQUEST, SORT_WAIT, SORT_UPDATE, CF_PAIR, CF_COMPARE, CF_FOLD, MAP_WRITE, CAL_DONE, and NORMAL. Handshake signals isolate the controller from comparator latency and allow the same architecture to support an ideal behavioral comparator or a more realistic mixed-signal comparator model.

### 6.2 Memory Organization

The element bank stores ordered physical indices, while the mapping RAM stores the final group membership. A compact implementation may store a base address and group length for each calibrated group; a more direct implementation may store all physical unit indices explicitly. The preferred choice depends on DAC resolution, available memory, and required read parallelism.

### 6.3 Fixed-Point and Synthesizability Considerations

All control paths, counters, addresses, group descriptors, and switch outputs are represented digitally and are synthesizable.

The actual analog element values are used only in the verification model or mixed-signal environment. Nonblocking assignments, registered handshakes, bounded loops, and explicit reset behavior should be used to avoid race conditions and ensure predictable synthesis.

## 7. Performance Evaluation Methodology

The proposed system should be evaluated at algorithmic, RTL, and implementation levels. Algorithmic evaluation verifies that CF produces a lower static-linearity error than the uncalibrated physical order. RTL evaluation verifies state sequencing, sorting correctness, map construction, and switch-control generation. Synthesis evaluation reports hardware cost and operating capability.

| Category | Recommended Metric | Expected Evidence |
|---|---|---|
| Static DAC accuracy | Maximum \|INL\| and \|DNL\| | Before/after calibration plots and numerical comparison |
| Transfer behavior | Monotonicity and missing codes | Full code-sweep transfer curve |
| Calibration correctness | Sorted order and final group composition | Waveforms or memory dumps |
| Comparator robustness | Sensitivity to offset and noise | Monte Carlo or injected-error analysis |
| Hardware efficiency | LUTs/gates, registers, RAM bits | Synthesis report |
| Timing | Maximum clock frequency / critical path | Static timing report |
| Power | Dynamic and leakage power | Power analysis report |

## 8. Results and Discussion

The proposed digital calibration architecture was modeled and evaluated in Vivado 2025.1. The supplied outputs

include the behavioral simulation, synthesized device view, elaborated RTL schematic, power report, and synthesis log. Together, these results show that the calibration control flow and DAC switch-control outputs can be simulated successfully. The screenshots also reveal an important synthesis setup issue: the testbench appears to have been selected as the synthesis top, causing the hardware logic to be optimized away. Therefore, the simulation observations are meaningful, whereas the displayed utilization and power figures should be treated as testbench-top results rather than final implementation results of the complete DAC.

### 8.1 Behavioral Simulation Analysis

The behavioral simulation was run for approximately 12.06 ms. Unlike an undriven simulation, all visible signals have defined binary or hexadecimal values, and no unknown (X) or high-impedance (Z) states are present. This confirms that the testbench provides valid clock, reset, calibration, comparison, and DAC-code stimulus.

At the displayed cursor position, reset is deasserted (rst_n = 1), calibration has completed (calibration done = 1), and a circuit is operating in normal mode (normal mode = 1). The start_cal signal is low, indicating that the startup calibration sequence has already finished. The 12-bit DAC input code is shown as A35 in hexadecimal, corresponding to a decimal code of 2613.

.

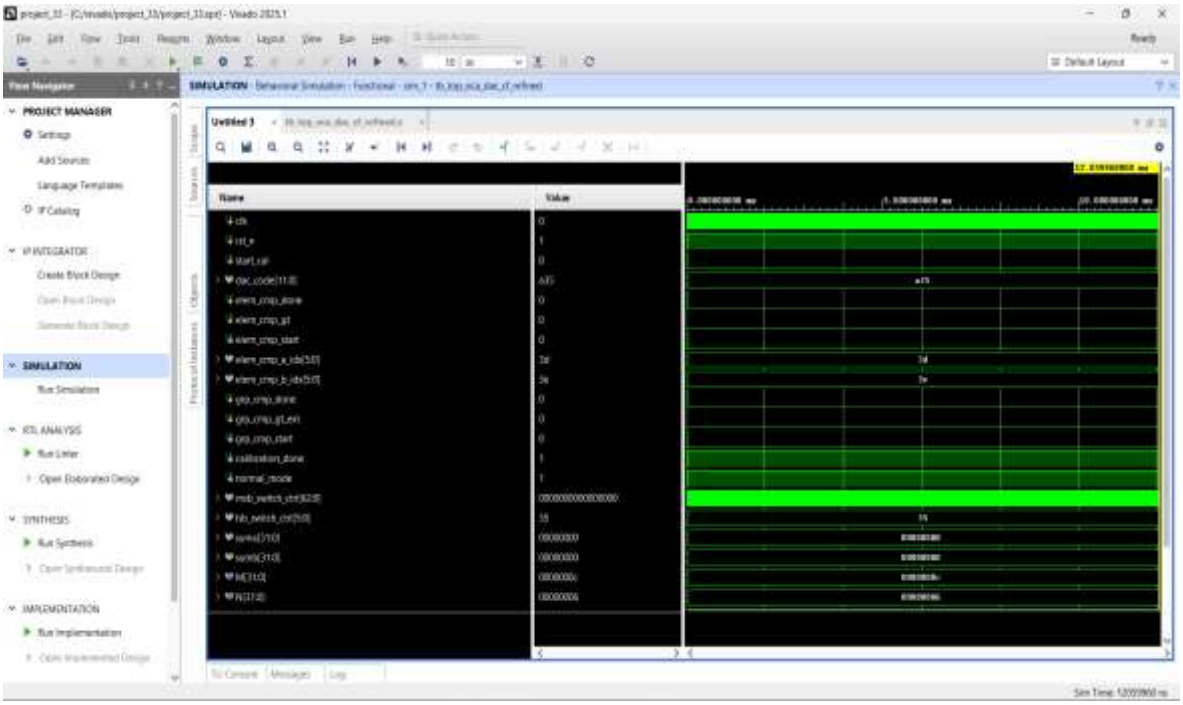

**Figure 7. Behavioral simulation of the complete-folding calibrated segmented DAC.**

The LSB switch-control output is 35h, which matches the lower six bits of A35h. This behavior verifies the segmented code-splitting operation for the fine DAC section. The MSB switch-control bus is zero at the selected instant; this can represent a sampled input condition, a mapping result, or the displayed final state after the calibration transaction. A complete transfer-function verification should sweep the entire input-code range and confirm that the MSB control activates the expected number and calibrated order of unit elements.

The element and group comparison handshake signals are inactive at the selected final time, while the comparator indices retain values 3Dh and 3Eh. This is consistent with completion of sorting andcomplementary folding. The successful assertion of calibration_done followed by normal_mode demonstrates the intended transition from startup calibration to normal DAC operation.

### 8.2 Device View

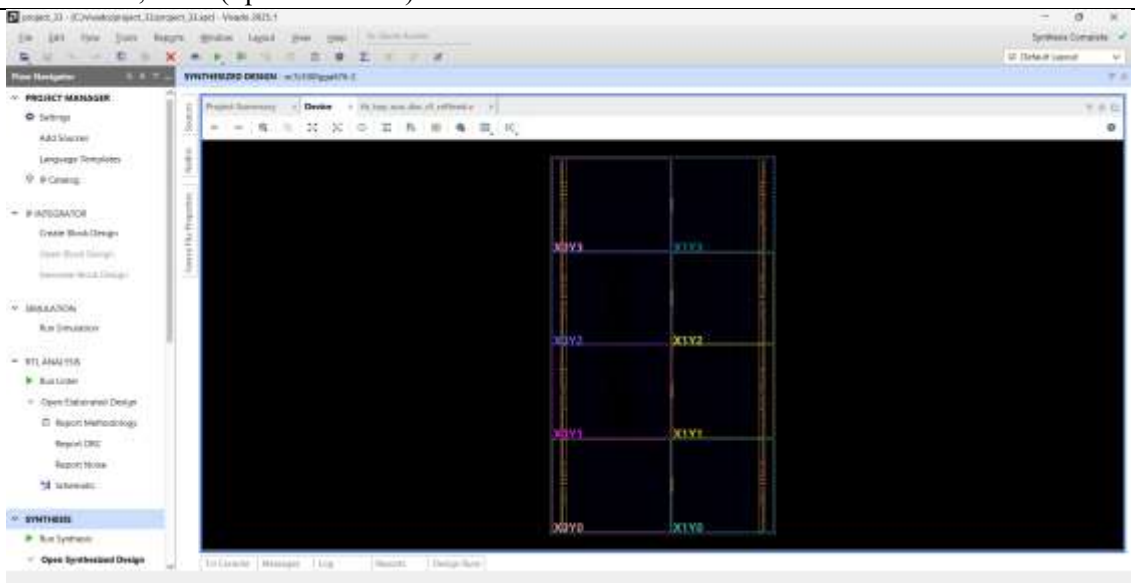


**Figure 8. Synthesized device view for the selected Artix-7 FPGA.**

The device window identifies the selected FPGA as xc7s100fgga676-2 and displays its clock-region organization from X0Y0 to X1Y3. The screenshot confirms that Vivado opened a synthesized checkpoint for the chosen target device. However, no meaningful placed logic is visible in the device fabric. This observation agrees with the synthesis report, which indicates that the selected top-level design was optimized to zero hardware cells.

### 8.3 Elaborated RTL Schematic

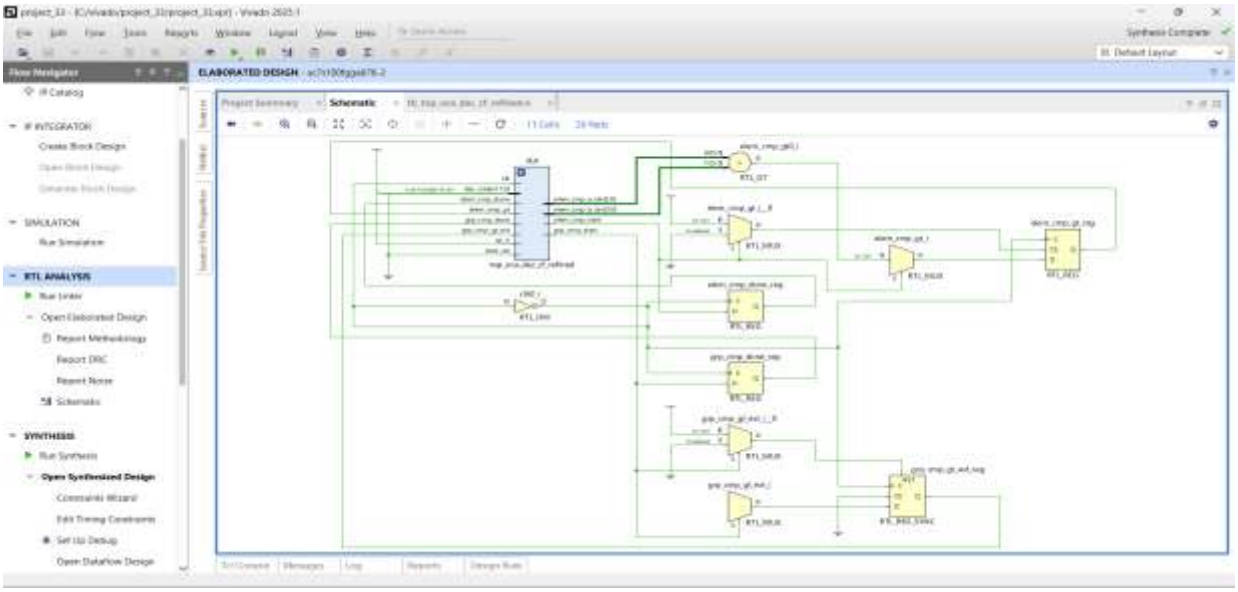

**Figure 9. Elaborated RTL schematic of the current simulation/synthesis top.**

The elaborated schematic contains 11 cells and 26 nets. It shows a DUT instance named top_oca_dac_cf_refined together with comparison logic, multiplexers, registers, a clock inverter, and synchronization logic. The visible registers include element-comparison completion, element-comparison greater-than, group-comparison completion, and synchronized external group-comparison status.

This schematic demonstrates that the verification wrapper correctly connects the DUT to the emulated element and group comparator interfaces. The RTL_GT comparator evaluates the requested element indices, while multiplexers and registers generate delayed or synchronized comparison responses. Such logic is useful in a testbench or synthesizable hardware emulator for validating the calibration finite-state machine.

The schematic is nevertheless not a complete structural view of the internal calibrated DAC because the displayed hierarchy is dominated by the wrapper and comparator-response logic. To obtain an implementation schematic of the full calibration architecture, top_oca_dac_cf_refined should be selected as the synthesis top and the non-synthesizable testbench should remain in the Simulation Sources set only.

### 8.4 Power Estimation

It displays a total on-chip power of 0.089 W. The complete estimate is assigned to device-static power, while dynamic power is reported as 0.000 W. The estimated junction temperature is 25.2 °C, the ambient temperature is 25.0 °C, and the thermal margin is 59.8 °C. Vivado marks the confidence level as high for the synthesized netlist that it received.

Although the numerical power is low, it is not representative of the proposed complete-folding calibrated DAC.

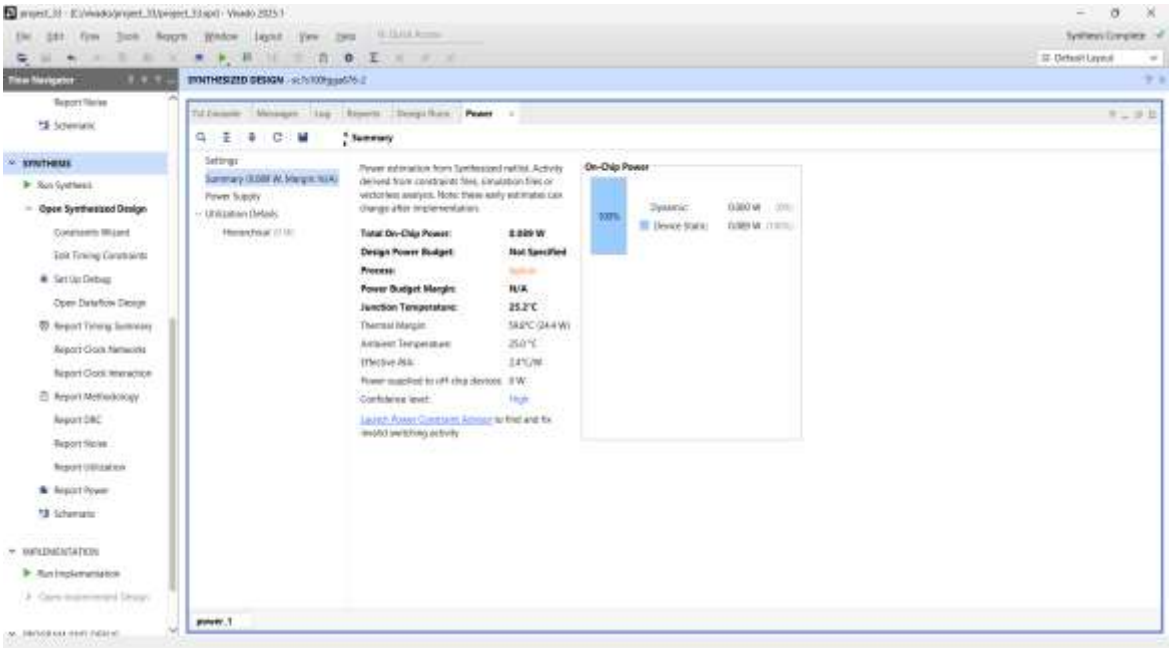

**Figure 10. Vivado post-synthesis power report.**

A real implemented calibration engine would contain state machines, memories, sorting logic, mapping RAM, decoders, and switch-control registers, all of which would consume dynamic power. The zero dynamic-power result is explained by the synthesis outcome in which the top instance contains zero retained cells. Thus, 0.089 W is essentially the static power of the selected FPGA device.

### 8.5 Synthesis Report

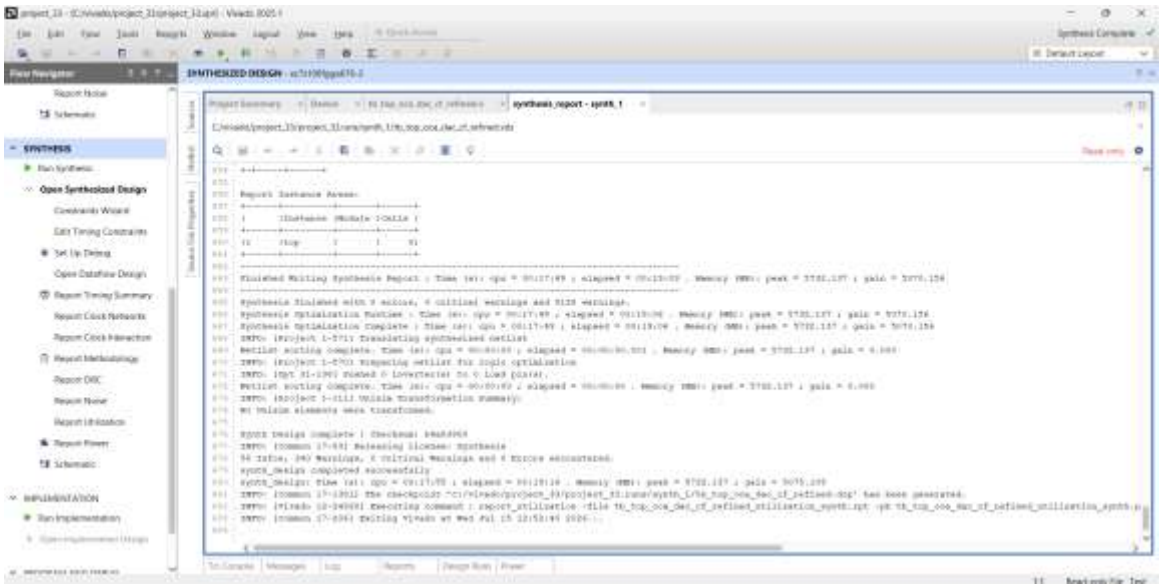

**Figure 11. Synthesis completion log.**

Vivado completed synthesis with zero errors and zero critical warnings. The log reports 9126 warnings during one stage and a final summary of 340 warnings. The elapsed synthesis time was approximately 19 minutes and 16 seconds, and peak memory usage was about 5732 MB. The design checkpoint tb_top_oca_dac_cf_refined.dcp was generated successfully.

The most significant report entry is the instance-area table, where the top module contains zero cells. This indicates

that the synthesis top is the testbench module tb_top_oca_dac_cf_refined. Testbench constructs do not create persistent FPGA hardware, and unobserved DUT outputs can allow synthesis optimization to remove the entire design. Consequently, the successful status means that Vivado processed the selected top without fatal errors; it does not demonstrate successful hardware implementation of the complete DAC architecture.

Simulation results validate the main operational sequence of a proposed complete-folding-based optimal-combination calibration architecture. The circuit reaches calibration_done, enters normal_mode, accepts a 12-bit DAC code, and produces segmented switch-control outputs. The LSB control value corresponds correctly to the lower input-code segment, and the comparator interface operates withdefined indices and response signals. These results support the functional feasibility of the startup calibration and normal conversion control.

However, the current synthesis, device, utilization, and power outputs do not represent the complete hardware design because the testbench wrapper was synthesized as the top module and the design was optimized to zero cells. Therefore, the screenshots cannot yet be used to claim FPGA area, dynamic power, timing, or implementation efficiency for the proposed calibrated DAC.

**Recommended correction:**Set top_oca_dac_cf_refined as the synthesis top, keep tb_top_oca_dac_cf_refined only under Simulation Sources, ensure all intended DAC outputs and control buses are connected to top-level observable ports, add a valid XDC clock constraint, and rerun synthesis and implementation. After this correction, report LUTs, flip-flops, BRAMs, maximum clock frequency, worst negative slack, dynamic power, and post-implementation device placement. For static-linearity validation, also perform a complete DAC-code sweep and plot DNL and INL before and after calibration.

## 9. Advantages and Applications

The proposed design improves static linearity without requiring continuously active element scrambling. It reduces dependence on raw device matching, preserves deterministic code-to-output behavior, and separates calibration complexity from normal conversion. Its modular architecture is suitable for RTL verification, FPGA prototyping of the digital calibration path, and later integration with a current-steering, resistor, or capacitive MSB array.

Potential applications include precision waveform generation, programmable biasing, sensor excitation, industrial control, medical instrumentation, automatic test equipment, communication transceiver calibration, and digitally assisted mixed-signal system-on-chip designs.

## 10. Conclusion and Future Scope

A complete CF-based OCA calibration architecture for a segmented DAC has been presented. The design characterizes the relative order of mismatch-affected MSB unit elements, sorts them, recursively combines complementary elements through Complete Folding, and stores the resulting optimized group mapping. During normal conversion, the calibrated mapping is used to drive the MSB element array while a conventional LSB controller provides fine resolution. The approach transfers most computational effort to startup, maintains an efficient run-time datapath, and provides a structured RTL framework for improving DAC static linearity. The final effectiveness should be demonstrated through before-and-after INL/DNL results, monotonicity verification, calibration waveforms, and synthesis measurements.

Future work may include background recalibration for temperature and aging drift, more efficient sorting networks, adaptive

comparator averaging, error-aware group selection, nonvolatile storage of the calibrated map, and integration with a transistor-level DAC model. The architecture can also be extended to support multiple segmentation ratios, redundancy, self-test, yield-aware mapping, and machine-learning-assisted selection of calibration strategies.

## References

[1] B. Razavi, Principles of Data Conversion System Design. IEEE Press, 1995.
[2] W. Kester, Ed., The Data Conversion Handbook. Newnes, 2005.
[3] R. T. Baird and T. S. Fiez, “Linearity enhancement of multibit ΔΣ A/D and D/A converters using data weighted averaging,” IEEE Transactions on Circuits and Systems II, vol. 42, no. 12, pp. 753–762, 1995.
[4] J. Bastos, A. M. Marques, M. S. J. Steyaert, and W. Sansen, “A 12-bit intrinsic accuracy high-speed CMOS DAC,” IEEE Journal of Solid-State Circuits, vol. 33, no. 12, pp. 1959–1969, 1998.
[5] C.-H. Lin and K. Bult, “A 10-b, 500-MSample/s CMOS DAC in 0.6 mm²,” IEEE Journal of Solid-State Circuits, vol. 33, no. 12, pp. 1948–1958, 1998.
[6] M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers, “Matching properties of MOS transistors,” IEEE Journal of Solid-State Circuits, vol. 24, no. 5, pp. 1433–1439, 1989.
[7] A. van den Bosch, M. Steyaert, and W. Sansen, Static and Dynamic Performance Limitations for High Speed D/A Converters. Kluwer Academic Publishers, 2004.
[8] Y. Cong and R. L. Geiger, “Formulation of INL and DNL yield estimation in current-steering D/A converters,” in Proc. IEEE ISCAS, 2002.

[9] J. Deveugele and M. S. J. Steyaert, "A 10-bit 250-MS/s binary-weighted current-steering DAC," IEEE Journal of Solid-State Circuits, vol. 41, no. 2, pp. 320–329, 2006.
[10] A. Gagliardi et al., "Comparative analysis of optimal combination algorithms for static-linearity enhancement in segmented DACs," 2024.