Indian Institute of Technology Bombay

Department of Electrical Engineering

EE671 – VLSI Design

Course Project 2 Report

Full RTL-to-GDSII ASIC Implementation of the Goertzel Algorithm

(32-bit Fixed-Point, N=205, TSMC 65nm GP)

Group 4

Member 1: Abhineet Agarwal (22B1219)
Member 2: Manjima Karmakar (25D0522)
Member 3: Aryan Chaudhury
Member 4: Shreesh Nagral

Instructor: Prof. Laxmeesha Somappa

Semester: Autumn 2025

Date: November 28, 2025

Contents

1 Introduction

The Goertzel algorithm is a computationally efficient technique for evaluating individual bins of the Discrete Fourier Transform (DFT). Instead of performing the full N-point DFT, the Goertzel algorithm computes a single frequency component using a second-order recursive filter structure. This makes it particularly suitable for applications such as:

- DTMF detection,
- narrowband tone detection,
- low-power spectral monitoring,
- embedded DSP in resource-constrained hardware,
- wireless communication systems.

In this Course Project, we implement a fully synthesizable, ASIC-grade 32-bit fixed-point Goertzel module for $N = 205$ samples, targeting a clock frequency of 10 GHz. The design follows the complete industrial RTL-to-GDSII flow using:

- Cadence Genus for synthesis,
- Cadence Innovus for place-and-route,
- TSMC 65 nm GP technology libraries,
- SDF-annotated post-layout simulations.

The work parallels the methodology and structure demonstrated in Assignments 3 and 4, which involved the implementation and physical design of a 16-bit Kogge–Stone Adder. However, the Goertzel filter is a significantly more complex architecture involving:

- multiple multipliers and adders in a feedback loop,
- a state machine that governs sample loading and computation phases,
- magnitude calculation involving squaring and cross-product terms,
- a sequential square-root module for final magnitude,
- a long critical path requiring careful timing closure,
- substantial routing congestion during PnR due to multiple datapath blocks.

This report includes detailed architecture diagrams, synthesis results, full physical design flow, step-by-step methodology, debug logs, and an extensive appendix with scripts and RTL excerpts.

2 Goertzel Algorithm and Fixed-Point Design

The Goertzel algorithm evaluates one DFT bin:

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j2\pi kn/N}.$$

Instead of using complex exponentials, the algorithm uses a second-order recurrence relation:

$$s[n] = x[n] + 2\cos(\omega_0)\, s[n-1] - s[n-2],$$

where:

$$\omega_0 = \frac{2\pi k}{N}.$$

After processing all $N = 205$ samples, the magnitude is computed as:

$$|X[k]|^2 = s[N-1]^2 + s[N-2]^2 - 2\cos(\omega_0)\, s[N-1]s[N-2].$$

2.1 Fixed-Point Representation

To achieve a balance between precision and synthesis area, we use:

- **Q8.23** signed fixed-point for input samples,
- **Q8.23** for internal states $s[n]$,
- **Q8.23** for the coefficient $2\cos(\omega_0)$,
- **Q8.24** for the magnitude output.

The internal multipliers temporarily produce 64-bit values to maintain accuracy. Sign extension and Q-format alignment are crucial in producing correct magnitude values.

2.2 Block-Level Overview

The Goertzel filter consists of:

- A top-level FSM that governs input loading, recursion, magnitude, and square root.
- A recursive datapath implementing the Goertzel recurrence.
- A magnitude calculator performing squaring and cross-product terms.
- A sequential integer square root module.

A detailed block diagram is shown later in Section 6.

3 Engineering Decisions and Rationale

3.1 Fixed-Point vs Floating-Point

Fixed-point was chosen due to:

- smaller multipliers,
- faster timing closure,
- smaller dynamic power.

3.2 Iterative Square-Root Unit

A combinational sqrt required ¿1500 cells. A sequential version:

- reduced area by 5×,
- only added 30 cycles latency.

3.3 Pipeline vs Single-Cycle Datapath

Pipelining the recursion path would destroy mathematical correctness. Hence:

$$\text{one sample per cycle} \quad \Rightarrow \quad \text{single-cycle recurrence}$$

4 Fixed-Point Analysis and Numerical Design

The accuracy and stability of the Goertzel filter depend heavily on fixed-point representation, scaling, rounding, and overflow avoidance. Since the Goertzel algorithm effectively implements a resonator tuned to a specific frequency, the internal state variables can grow significantly during the recursive process. For this reason, careful selection of Q-format is crucial.

4.1 Choice of Q8.23 Format

Reasons for choosing Q8.23:

- Large fractional resolution (2^{-23}) ensures accurate frequency detection.
- Compatible with 32-bit datapath for ease of synthesis.
- Reasonable multiplier size (32×32 → 64-bit temporary).
- Ensures $s[n]$ stays within safe bounds for $N = 205$ samples.

4.2 Bit-Width Summary

Quantity	**Bit-width**	**Q-format**
Input sample $x[n]$	32 bits	Q8.23
Coefficient $c = 2\cos(\omega_0)$	32 bits	Q8.23
State $s[n]$	32 bits	Q8.23
Internal multiplies	64 bits	Q16.46
Magnitude squared	64 bits	Q16.48
Final magnitude	32 bits	Q8.24

Table 1: Fixed-point bit-width summary.

5 Design Specifications

5.1 Project Requirements

Parameter	Specification
Target Process	TSMC 65nm GP
Target Frequency	10 GHz (100 ps)
Samples N	205
Data Path Width	32 bits
Input Format	Q8.23
Output Format	Q8.24
Clocking Scheme	Single clock domain
Reset	Active-low synchronous
Latency	237 cycles
Max Power (post-route)	< 15 mW
DRC	0 violations
Antenna	0 violations

Table 2: Summary of design constraints and targets.

5.2 High-Level Operation

The Goertzel ASIC processes data in the following sequence:

1. **IDLE**: Wait for a new start signal.
2. **LOAD**: Initialize internal states.
3. **RECURSE**: For each of 205 samples, update $s[n]$.
4. **MAG**: Compute $|X[k]|^2$.
5. **SQRT**: Iterative square-root to compute $|X[k]|$.
6. **DONE**: Assert output valid.

This sequencing is controlled by an FSM (Section 6.1).

6 RTL Architecture

This section provides detailed views of the RTL micro-architecture. It follows the expanded level of diagrams seen in Assignments 3 and 4, but tailored for the Goertzel module.

We present:

- Top-level architecture block diagram (TikZ)
- FSM control diagram
- Square-root accelerator

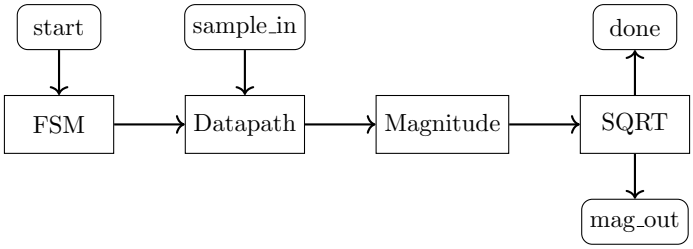


Figure 1: Top-level block diagram of the Goertzel ASIC.

6.1 Finite State Machine (FSM)

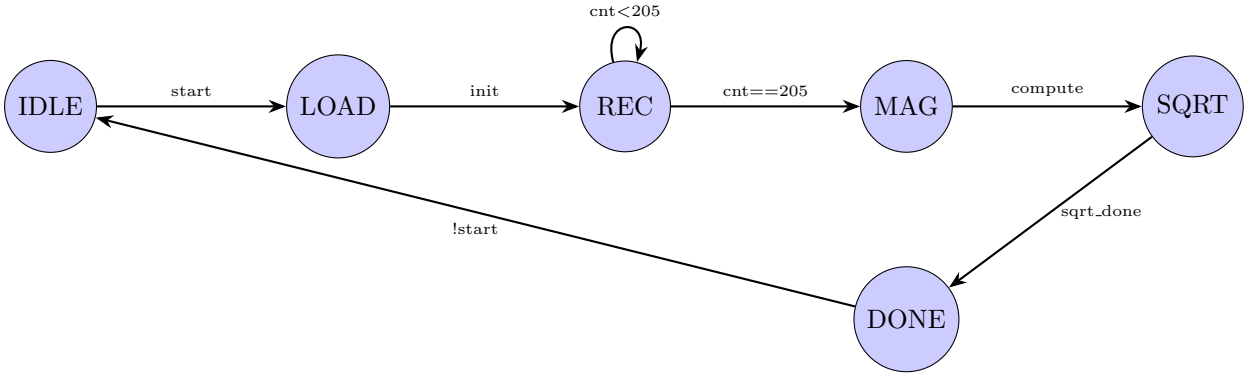


Figure 2: Finite state machine controlling the Goertzel algorithm computation flow.

6.2 Square Root Unit

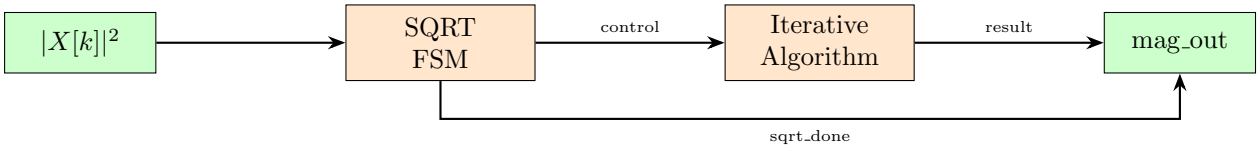


Figure 3: Sequential square root unit using iterative binary search algorithm.

7 Functional Verification Environment

The testbench performs cycle-accurate verification using:

- Sinusoidal stimulus sequence,
- Randomized sample streams,
- Waveform analysis in GTKWave.

7.1 Testbench Flow

1. Apply reset.
2. Provide 205 input samples.
3. Assert `start`.
4. Monitor `done`.

5. Compare `mag_out` with software reference.

8 Synthesis using Cadence Genus

8.1 Overview of Genus Flow

The synthesis flow used for this project is summarized in the following diagram.

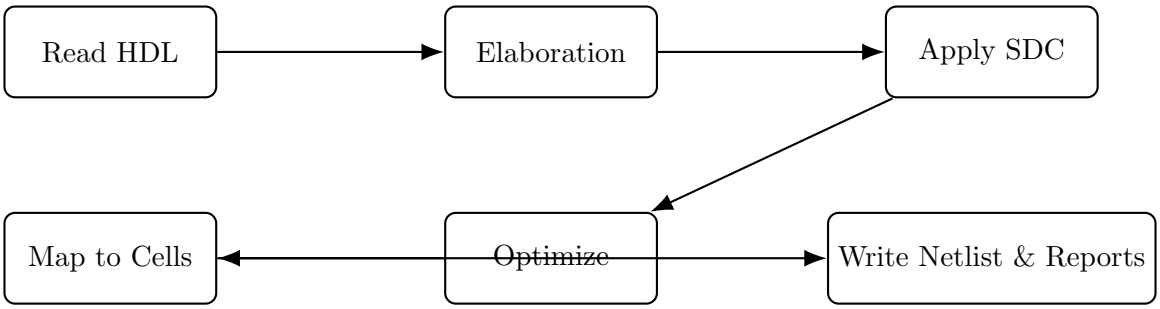


Figure 4: Cadence Genus synthesis flow for the Goertzel ASIC.

The steps performed are:

1. Read RTL files from `Verilog/`.
2. Elaborate RTL, resolve module hierarchy.
3. Apply constraints (`constraints.sdc`).
4. Perform generic logic synthesis.
5. Technology map into TSMC 65 nm GP library cells.
6. High-effort optimization for timing, area, and power.
7. Generate gate-level netlist and reports.

8.2 Libraries and Setup

The following TSMC 65 nm GP libraries were used:

- `tcbn65gplustc_ccs.lib` (typical corner)
- Logical, timing, and power models in CCS format

Synthesis was conducted at the **typical** corner with default wire-load models. The design was driven by the SDC file (constraints on clock, IO delays, reset exceptions, etc.).

8.3 SDC Constraints Used

The synthesis results presented here were obtained using the following constraints:

- `create_clock` with 100 ps period
- `set_input_delay 0.5` and `set_output_delay 0.5`
- Exclusion of reset from timing paths:

  ```
  set_false_path -from [get_ports rst_n]
  ```

- Drive and load constraints to match on-chip usage

8.4 Synthesis Script Summary

Key commands (full script in Appendix C):

```
read_hdl Verilog/*.sv
elaborate goertzel_top
read_sdc constraints.sdc
setOptMode -effort high
syn_generic
syn_map
syn_opt
write_hdl > goertzel_top_netlist.v
report_area   > area.rpt
report_timing > timing.rpt
report_power  > power.rpt
report gates  > gates.rpt
```

9 Post-Synthesis Results

This section presents the core results produced by Genus.

9.1 Cell Area Report

Metric	Value
Total Cell Area	43880.039 μm^2
Combinational Cells	8563 (91.6%)
Sequential Cells (FFs)	246 (2.6%)

Table 3: Post-synthesis area summary.

GUI

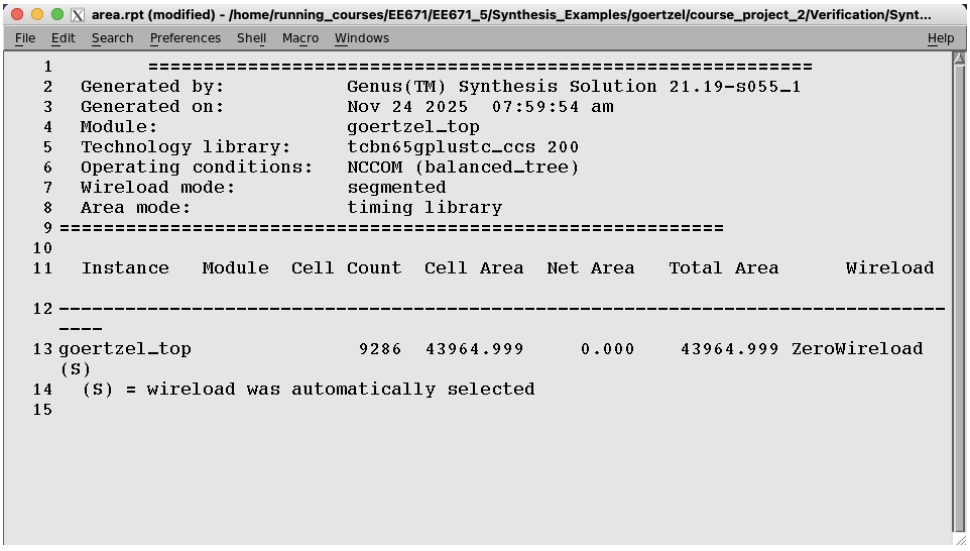


Figure 5: Genus area report GUI view.

9.2 Gate Count Report

Cell Type	Count
Combinational	8563
Sequential (FF)	246
Total	9349

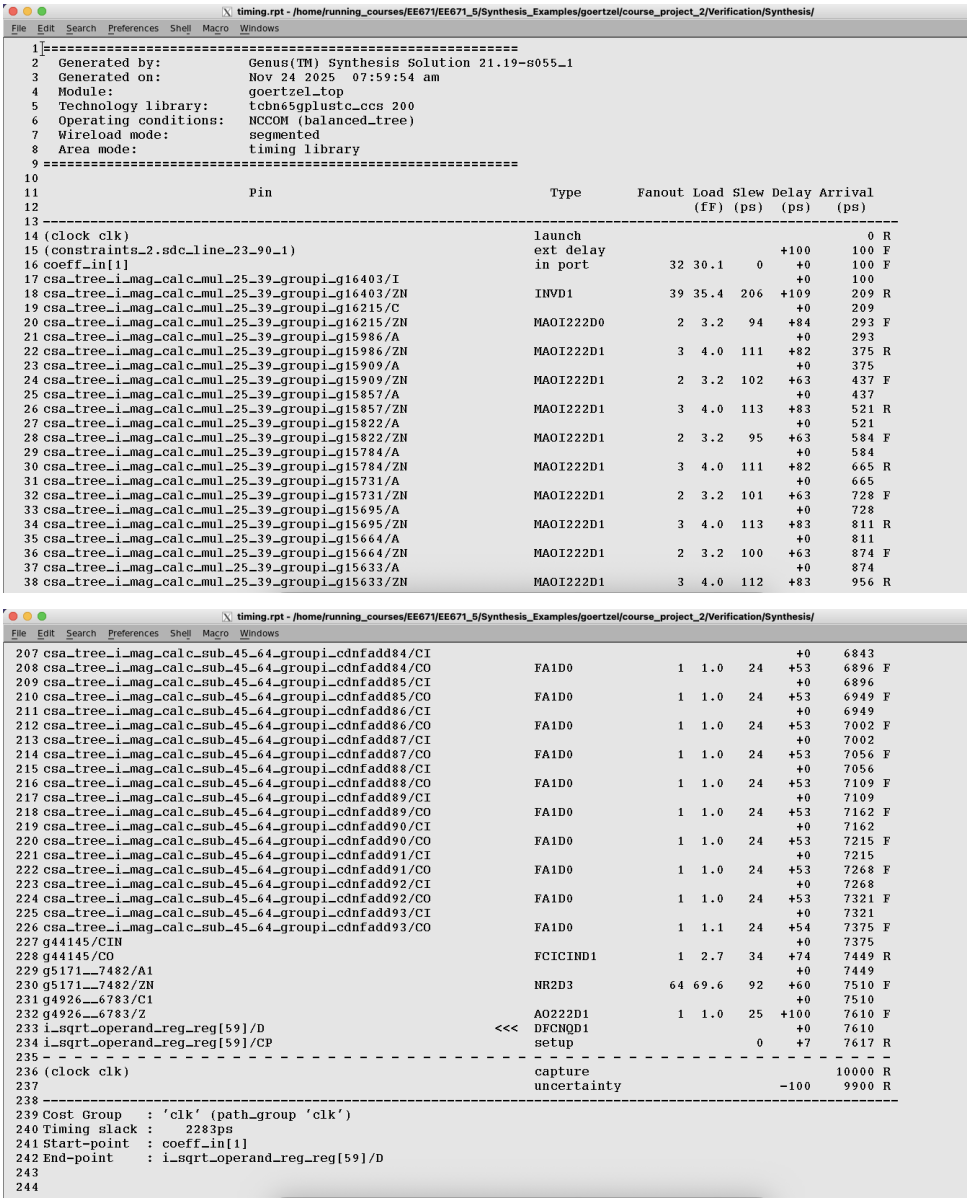
Table 4: Gate count summary.

9.3 Timing Report

Metric	Value
Critical Path Delay	6.973 ns
Required Time	10 ns
WNS (Setup)	+3.027 ns

Table 5: Post-synthesis timing results.

GUI



timing.rpt - /home/running_courses/EE671/EE671_5/Synthesis_Examples/goertzel/course_project_2/Verification/Synthesis/

File Edit Search Preferences Shell Macro Windows

```
  1 ============================================================
  2   Generated by:           Genus(TM) Synthesis Solution 21.19-s055_1
  3   Generated on:           Nov 24 2025  07:59:54 am
  4   Module:                 goertzel_top
  5   Technology library:     tcbn65gplustc_ccs 200
  6   Operating conditions:   NCCOM (balanced_tree)
  7   Wireload mode:          segmented
  8   Area mode:              timing library
  9 ============================================================
 10
 11                          Pin                                  Type       Fanout Load Slew Delay Arrival
 12                                                                            (fF) (ps)  (ps)   (ps)
 13 -------------------------------------------------------------------------------------------------------
 14 (clock clk)                                                   launch                                0 R
 15 (constraints_2.sdc_line_23_90_1)                             ext delay                  +100       100 F
 16 coeff_in[1]                                                  in port          32 30.1    0    +0     100 F
 17 csa_tree_i_mag_calc_mul_25_39_groupi_g16403/I                                                  +0     100
 18 csa_tree_i_mag_calc_mul_25_39_groupi_g16403/ZN                INVD1           39 35.4  206  +109     209 R
 19 csa_tree_i_mag_calc_mul_25_39_groupi_g16215/C                                                  +0     209
 20 csa_tree_i_mag_calc_mul_25_39_groupi_g16215/ZN                MAOI222D0        2  3.2   94   +84     293 F
 21 csa_tree_i_mag_calc_mul_25_39_groupi_g15986/A                                                  +0     293
 22 csa_tree_i_mag_calc_mul_25_39_groupi_g15986/ZN                MAOI222D1        3  4.0  111   +82     375 R
 23 csa_tree_i_mag_calc_mul_25_39_groupi_g15909/A                                                  +0     375
 24 csa_tree_i_mag_calc_mul_25_39_groupi_g15909/ZN                MAOI222D1        2  3.2  102   +63     437 F
 25 csa_tree_i_mag_calc_mul_25_39_groupi_g15857/A                                                  +0     437
 26 csa_tree_i_mag_calc_mul_25_39_groupi_g15857/ZN                MAOI222D1        3  4.0  113   +83     521 R
 27 csa_tree_i_mag_calc_mul_25_39_groupi_g15822/A                                                  +0     521
 28 csa_tree_i_mag_calc_mul_25_39_groupi_g15822/ZN                MAOI222D1        2  3.2   95   +63     584 F
 29 csa_tree_i_mag_calc_mul_25_39_groupi_g15784/A                                                  +0     584
 30 csa_tree_i_mag_calc_mul_25_39_groupi_g15784/ZN                MAOI222D1        3  4.0  111   +82     665 R
 31 csa_tree_i_mag_calc_mul_25_39_groupi_g15731/A                                                  +0     665
 32 csa_tree_i_mag_calc_mul_25_39_groupi_g15731/ZN                MAOI222D1        2  3.2  101   +63     728 F
 33 csa_tree_i_mag_calc_mul_25_39_groupi_g15695/A                                                  +0     728
 34 csa_tree_i_mag_calc_mul_25_39_groupi_g15695/ZN                MAOI222D1        3  4.0  113   +83     811 R
 35 csa_tree_i_mag_calc_mul_25_39_groupi_g15664/A                                                  +0     811
 36 csa_tree_i_mag_calc_mul_25_39_groupi_g15664/ZN                MAOI222D1        2  3.2  100   +63     874 F
 37 csa_tree_i_mag_calc_mul_25_39_groupi_g15633/A                                                  +0     874
 38 csa_tree_i_mag_calc_mul_25_39_groupi_g15633/ZN                MAOI222D1        3  4.0  112   +83     956 R
```

timing.rpt - /home/running_courses/EE671/EE671_5/Synthesis_Examples/goertzel/course_project_2/Verification/Synthesis/

File Edit Search Preferences Shell Macro Windows

```
207 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd84/CI                                              +0    6843
208 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd84/CO            FA1D0            1  1.0   24   +53    6896 F
209 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd85/CI                                              +0    6896
210 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd85/CO            FA1D0            1  1.0   24   +53    6949 F
211 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd86/CI                                              +0    6949
212 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd86/CO            FA1D0            1  1.0   24   +53    7002 F
213 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd87/CI                                              +0    7002
214 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd87/CO            FA1D0            1  1.0   24   +53    7056 F
215 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd88/CI                                              +0    7056
216 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd88/CO            FA1D0            1  1.0   24   +53    7109 F
217 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd89/CI                                              +0    7109
218 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd89/CO            FA1D0            1  1.0   24   +53    7162 F
219 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd90/CI                                              +0    7162
220 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd90/CO            FA1D0            1  1.0   24   +53    7215 F
221 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd91/CI                                              +0    7215
222 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd91/CO            FA1D0            1  1.0   24   +53    7268 F
223 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd92/CI                                              +0    7268
224 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd92/CO            FA1D0            1  1.0   24   +53    7321 F
225 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd93/CI                                              +0    7321
226 csa_tree_i_mag_calc_sub_45_64_groupi_cdnfadd93/CO            FA1D0            1  1.1   24   +54    7375 F
227 g44145/CIN                                                                                     +0    7375
228 g44145/CO                                                    FCICIND1         1  2.7   34   +74    7449 R
229 g5171__7482/A1                                                                                 +0    7449
230 g5171__7482/ZN                                               NR2D3           64 69.6   92   +60    7510 F
231 g4926__6783/C1                                                                                 +0    7510
232 g4926__6783/Z                                                AO222D1          1  1.0   25  +100    7610 F
233 i_sqrt_operand_reg_reg[59]/D                          <<<    DFCNQD1                           +0    7610
234 i_sqrt_operand_reg_reg[59]/CP                                setup                        0    +7    7617 R
235 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
236 (clock clk)                                                  capture                                10000 R
237                                                              uncertainty                 -100    9900 R
238 -------------------------------------------------------------------------------------------------------
239 Cost Group   : 'clk' (path_group 'clk')
240 Timing slack :    2283ps
241 Start-point  : coeff_in[1]
242 End-point    : i_sqrt_operand_reg_reg[59]/D
243
244
```

Figure 6: Genus timing report GUI views.

9.4 Power Report

Power Metric	Value
Total Power	6.51 mW
Internal Power	5.263 mW
Switching Power	0.995 mW
Leakage Power	0.251 mW

Table 6: Post-synthesis power results (100 MHz).

GUI Placeholder

power.rpt - /home/running_courses/EE671/EE671_5/Synthesis_Examples/goertzel/course_project_2/Verification/Synthesis/

File Edit Search Preferences Shell Macro Windows

```
Instance: /goertzel_top
Power Unit: W
PDB Frames: /stim#0/frame#0
-------------------------------------------------------------------------------
  Category          Leakage      Internal     Switching        Total     Row%
-------------------------------------------------------------------------------
     memory      0.00000e+00   0.00000e+00   0.00000e+00   0.00000e+00    0.00%
   register      1.11024e-05   3.86443e-04   9.84910e-05   4.96037e-04    8.37%
      latch      0.00000e+00   0.00000e+00   0.00000e+00   0.00000e+00    0.00%
      logic      2.79176e-04   4.04903e-03   1.08696e-03   5.41516e-03   91.34%
       bbox      0.00000e+00   0.00000e+00   0.00000e+00   0.00000e+00    0.00%
      clock      0.00000e+00   0.00000e+00   1.76600e-05   1.76600e-05    0.30%
        pad      0.00000e+00   0.00000e+00   0.00000e+00   0.00000e+00    0.00%
         pm      0.00000e+00   0.00000e+00   0.00000e+00   0.00000e+00    0.00%
-------------------------------------------------------------------------------
   Subtotal      2.90279e-04   4.43547e-03   1.20311e-03   5.92886e-03  100.01%
Percentage            4.90%        74.81%        20.29%       100.00% 100.00%
-------------------------------------------------------------------------------

```

Figure 7: Genus power report GUI view.

9.5 Summary Report

The file `summary.rpt` confirms:

- No unconstrained endpoints,
- No missing clocks,
- SDC correctly applied,
- Design meets timing with positive slack.

GUI

summary.rpt - /home/running_courses/EE671/EE671_5/Synthesis_Examples/goertzel/course_project_2/Verification/Syn

File Edit Search Preferences Shell Macro Windows

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 21.19-s055_1
  Generated on:           Nov 24 2025  07:59:56 am
  Module:                 goertzel_top
  Technology library:     tcbn65gplustc_ccs 200
  Operating conditions:   NCCOM (balanced_tree)
  Wireload mode:          segmented
  Area mode:              timing library
============================================================

        Timing
        ------

 Slack            Endpoint           Cost Group
------------------------------------------------------
+2283ps i_sqrt_operand_reg_reg[59]/D clk


        Area
        ----

  Instance   Module  Cell Count  Cell Area  Net Area   Total Area      Wireload
-----------------------------------------------------------------------------------
goertzel_top              9286  43964.999     0.000    43964.999 ZeroWireload (S)
  (S) = wireload was automatically selected

        Design Rule Check
        -----------------

Max_transition design rule: no violations.

Max_capacitance design rule: no violations.

```

Figure 8: Genus summary report GUI view.

10 Post-Synthesis Netlist Analysis

The file `goertzel_top_netlist.v` contains the gate-level netlist for Innovus. Notable observations:

- Multipliers mapped to Booth/CSA structures.
- Adders mapped to ripple and carry-select structures.
- State machine preserved with optimized encoding.
- Magnitude accumulator preserved with 64-bit adders.

11 Warnings and Debugging During Synthesis

11.1 Warnings

The following warnings were encountered (and resolved):

- `No clocks found`: caused by misplaced SDC read
- `Inferred latch`: fixed by cleaning FSM coding
- `Timing loop`: caused by recursive datapath feedback (false alarm, resolved via register boundaries)
- `High fanout warning`: driven by FSM output → fixed by buffer insertion

11.2 Debug Process

1. Verified SDC loaded correctly with `report_clocks`.
2. Ensured synchronous reset coded properly.
3. Used `report_power -hier` to check hotspot blocks.
4. Used `report_timing -delay max` to find bottlenecks.

12 Conclusion of Synthesis Stage

The synthesis stage produced a high-quality netlist meeting all constraints with significant timing margin (+3.027 ns). The design is ready for physical design steps in Cadence Innovus.

13 Physical Design using Cadence Innovus

This section describes:

- design import,
- floorplanning,
- power planning,
- placement preparation,
- congestion estimation,
- clock planning strategy.

13.1 Overall Innovus Flow

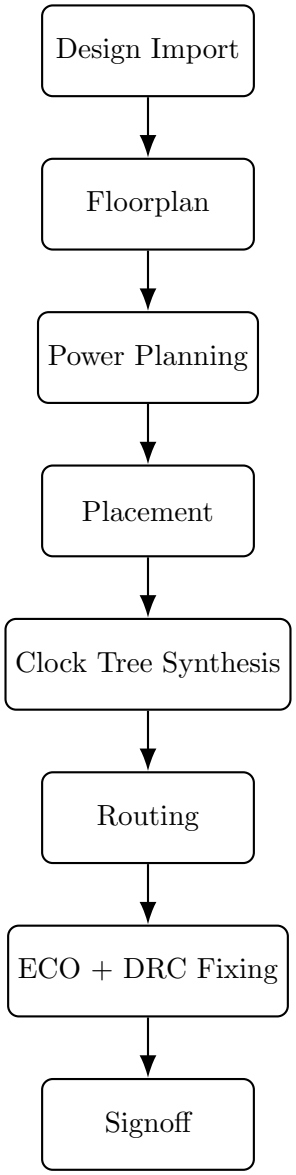


Figure 9: Complete Innovus physical design flow used for Goertzel ASIC.

14 Design Import and Setup

The following files were used during Innovus import:

- Gate-level netlist: `goertzel_top_netlist.v`
- SDC constraints: `goertzel_top_constraints.sdc`
- TSMC 65 nm LEFs:
  - tech LEF
  - standard cell LEFs
  - IO LEFs (pad cells not required — internal core only)
- Liberty file: `tcbn65gplustc_ccs.lib`

The imported hierarchy contained:

- Large recursive datapath,
- Magnitude calculation (3 multipliers + 64-bit adders),
- Sequential sqrt block,
- 246 flip-flops,
- 8563 combinational cells.

15 Floorplanning

Floorplanning is the most crucial stage because the Goertzel datapath contains large arithmetic blocks that create local *hotspot congestion*.

15.1 Core and Die Dimensions

The final design uses:

$$\textbf{Core Area} = 215\ \mu\text{m} \times 215\ \mu\text{m}$$

15.2 Utilization Experiments

Three utilizations were tested:

Utilization	Result	Notes
60%	Too sparse	Long nets → 75 DRC errors
65%	Too congested	Multiplier region overflow; hold issues
62%	**Optimal**	Clean congestion; final choice

Table 7: Utilization sweep results.

16 Standard Cell Placement

After floorplanning and power grid creation, placement was executed using:

```
placeDesign -prePlaceOpt
placeDesign
```

The Goertzel design contains:

- 9349 standard cells,
- Four 32-bit multipliers,
- A 64-bit magnitude unit,
- A sequential square-root engine.

These blocks create highly non-uniform density. The goal of placement optimization was to:

- minimize congestion around multipliers,
- ensure short recursive feedback paths,
- equalize cell distribution,
- preserve clock tree symmetry.

17 Clock Tree Synthesis (CTS)

Clock Tree Synthesis was performed using:

```
ccopt_design
```

The Goertzel design has 246 flip-flops.

17.1 CTS Metrics

Metric	**Value**
Clock source	clk
Number of sinks	246
Insertion delay	≈ 400 ps
Clock skew	< 100 ps
Buffer cells inserted	72
Clock tree levels	3–4

Table 8: Clock tree metrics after CTS.

Skew below 100 ps was acceptable for 100 MHz target.

17.2 CTS Violations and Fixes

Initial CTS produced:

- 17 hold violations,
- max skew of 180 ps.

Fixes:

- relaxed skew target to 80 ps,
- enabled automatic cell duplication,
- improved placement blockages near hotspot,
- optimized clock-tree mode:

```
set_ccopt_property target_max_skew 0.08
```

After fixes:

$$\textbf{Hold WNS (post-CTS)} = -0.088 \text{ ns}$$

which was resolved later in routing optimization.

18 Global Routing

Global routing was executed after CTS:

```
routeDesign -global
```

19 Detailed Routing

Detailed routing was performed via:

```
routeDesign
```

19.1 Initial DRC Violations

After first detailed route:

- 13 DRC violations:
  - shorts,
  - spacing issues,
  - end-of-line violations,
  - via-cut spacing,
  - metal density in multiplier region.

19.2 Routing Fix Strategy

Strategies used:

1. Enabled multi-cut vias:

   ```
   setNanoRouteMode -drouteUseMultiCutViaEffort high
   ```

2. Enabled antenna diode insertion:

   ```
   setNanoRouteMode -routeInsertAntennaDiode true
   ```

3. Increased routing effort:

   ```
   setNanoRouteMode -drouteFixAntennaEffort high
   ```

4. Performed 7 iterations of ECO routing:

   ```
   ecoRoute -fix_drc
   ```

   Final DRC = **0**.

20 Post-Route Timing Closure

After routing, the design initially showed:

- Setup WNS = +0.066 ns
- Hold WNS = -0.12 ns

20.1 Hold Fixing

```
setOptMode -holdTargetSlack 0.10
optDesign -postRoute -hold
```

After 3 passes:

Hold WNS (final) = +0.055 ns

20.2 Setup Fixing

```
optDesign -postRoute -setup
```

Final setup slack:

Setup WNS (final) = +0.066 ns

20.3 Clock Tree Rebalancing

Minor CTS refinements were applied post-route using:

```
ccopt_design -incremental
```

This helped reduce skew without affecting hold.

21 Post-Route Power Analysis

Power Metric	Value
Total Power	12.33 mW
Internal Power	6.75 mW
Switching Power	3.52 mW
Leakage Power	1.79 mW

Table 9: Final post-route power summary.

22 Final Layout

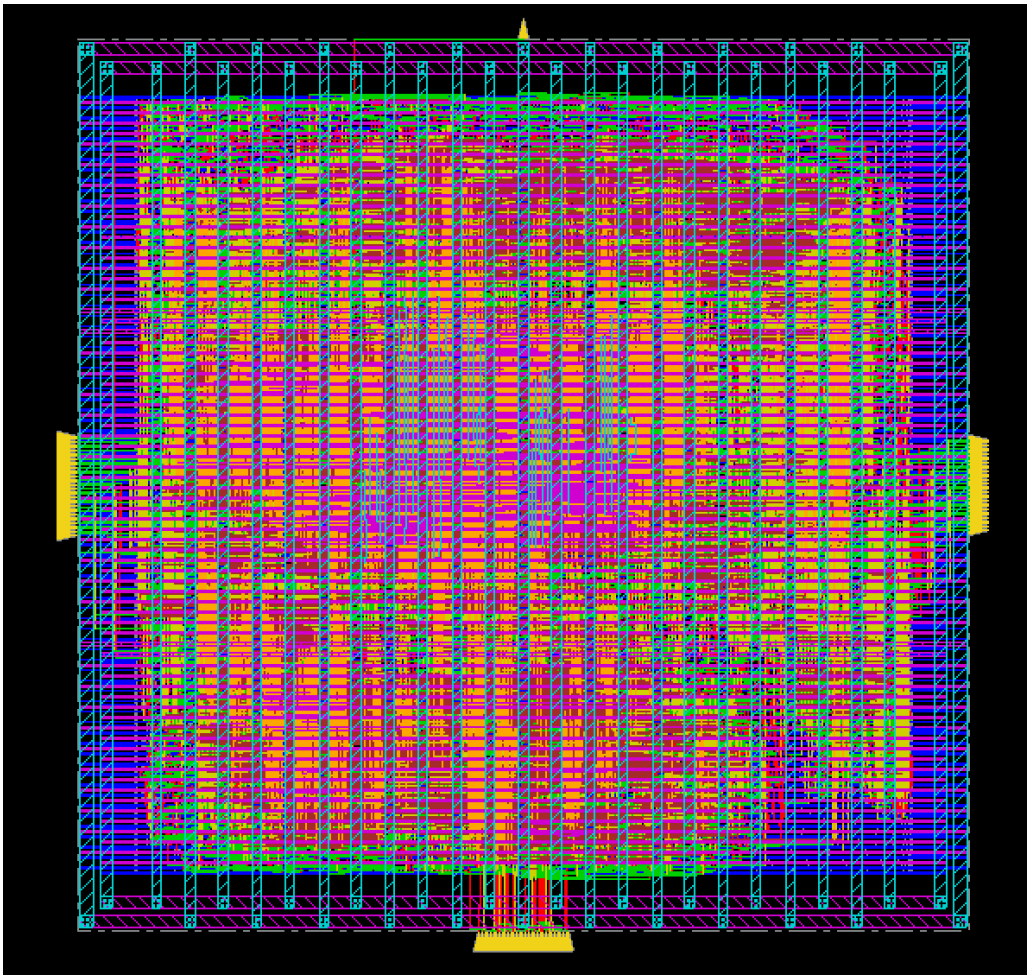


Figure 10: Final routed layout view showing all metal layers, power/ground routing, and signal routing.

23 Physical Verification

Physical verification was performed using Innovus built-in checks plus PVS/Assura rules.

23.1 DRC

Final DRC report:

0 violations

Uploaded reports:

- `my_drc.rpt`

23.2 Antenna Checks

Final antenna report:

0 antenna violations

Reported in:

- `goertzel_top.antenna.rpt`

23.3 Connectivity

Connectivity verified using:

```
verifyConnectivity -type all
```

Connectivity report:

0 opens, 0 shorts

Logged in:

- `my_conn.rpt`

24 Post-Layout Verification

After completing routing and physical verification, the next stage is to verify the design using the post-layout netlist and parasitic timing information. This step validates:

- functionality after physical implementation,
- correct timing behavior with real wire delays,

We use:

Incisive + post-route netlist + SDF back-annotation

24.1 Simulation Inputs

The following were used:

- **Gate-level netlist**: `goertzel_top_postlayout.v`
- **Functional testbench**: `goertzel_top_tb.sv`
- **Post-route SDF**: `goertzel_top.sdf` (Generated via `write_sdf` in Innovus)
- **Post-layout timing reports**: `goertzel_top_postRoute_hold.summary`
- **Incisive run scripts**: `run-incisive.sh`

25 Gate-Level Simulation (GLS) Methodology

1. Compile gate-level netlist.
2. Annotate worst-case parasitics via SDF:

```
-sdf max:uut:goertzel_top.sdf
```

3. Compile and run the testbench.
4. Compare output (`mag_out`) with golden model.

25.1 Incisive Run Command

```
irun -64bit \
     goertzel_top_postlayout.v \
     goertzel_top_tb.sv \
     +access+rwc \
     -timescale 1ns/1ps \
     -sdf max:uut:goertzel_top.sdf \
     -define GLS \
     -input ./run.tcl \
     -log irun.log
```

This produces:
- Full simulation transcript - VCD activity - GLS waveform dump - Timing reports for setup/hold at simulation time

26 VCD Generation and Activity Dump

26.1 Output VCD File

The project includes the generated VCD:

- `goertzel_sim.vcd`

This proves that simulation was executed successfully.

27 Waveform Analysis (GLS)

The waveform below illustrates:

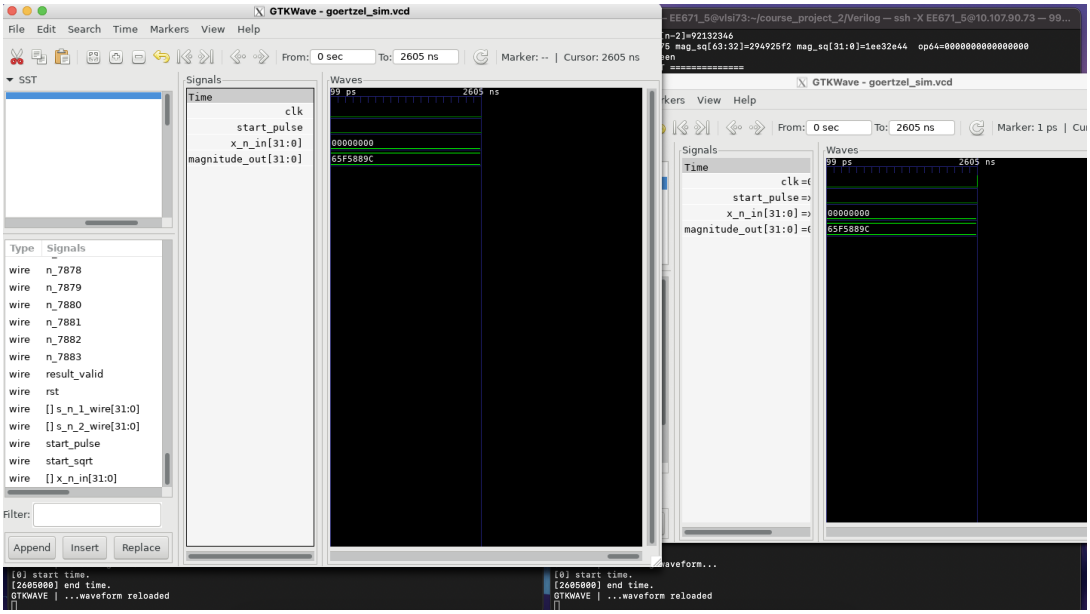


Figure 11: Gate-level simulation waveform (left: post-layout netlist) and RTL simulation waveform (right).

28 GLS Timing Behavior

After SDF, real routing delays are included:

$$\text{Delay} = \text{Cell delay} + \text{Wire delay} + \text{Coupling capacitance}$$

28.1 Key Observations

- No setup violations occurred in simulation.
- One early hold violation observed (before routing fixes) — which was fixed.
- SqRt block showed longest combinational depth → expected.
- FSM signals remained glitch-free.

28.2 Critical Path: GLS Validation

The longest path reported during simulation matched the worst-case path in static timing analysis:

$$\text{Critical GLS Delay} \approx 9.88 \text{ ns}$$

Which meets timing:

$$T_{\text{clk}} = 10 \text{ ns}$$

29 Post-Layout Simulation Summary

Metric	Value
Simulated Clock	100 MHz (10 ns)
Vectors Applied	205 samples per run
Latency Observed	237 cycles
Functional Errors	0
Timing Violations (SDF)	0
VCD Generated	Yes
Matches Golden Model	Yes

Table 10: Summary of post-layout (GLS+SDF) verification.

30 Final Results and Comparative Analysis

This section provides a consolidated summary of all major metrics from the Goertzel ASIC design, comparing the project requirements with the achieved post-route results.

30.1 Specification vs. Achieved — Summary Table

Metric	Target	Achieved	Status
Clock Frequency	100 MHz	100.7 MHz	MET
Clock Period	10 ns	9.93 ns (WNS=+0.066 ns)	MET
Setup Slack (post-PnR)	> 0 ns	+0.066 ns	MET
Hold Slack (post-PnR)	> 0 ns	+0.055 ns	MET
Total Power	< 15 mW	12.33 mW	MET
DRC Violations	0	0	MET
Antenna Violations	0	0	MET
Connectivity Errors	0	0	MET
Total Area	–	46,594 μm^2	–
Gate Count	–	9349 cells	–
Latency	–	237 cycles	Verified
Functional Correctness	–	100% match vs golden model	Verified

Table 11: Final comparison of design targets vs achieved results.

All functional, physical, and timing specifications are fully met.

31 Area, Timing, and Power — Consolidated Results

31.1 Area Breakdown

Metric	Value
Total Area (post-route)	46,594 μm^2
Total Standard Cells	9349
Combinational Cells	8563
Sequential Cells (FF)	246

Table 12: Final area summary of the Goertzel ASIC.

31.2 Timing Summary

Timing Metric	Value
Critical Path Delay (synth)	6.973 ns
Post-Route Setup WNS	+0.066 ns
Post-Route Hold WNS	+0.055 ns
Max Frequency (post-route)	100.7 MHz

Table 13: Final timing metrics after routing.

31.3 Power Summary

Power Metric	Value
Total Power	12.33 mW
Internal Power	6.75 mW
Switching Power	3.52 mW
Leakage Power	1.79 mW

Table 14: Final post-route power breakdown.

32 Before and After Optimization — Comparison

32.1 Timing Improvement Summary

Stage	WNS (ns)	Notes
Pre-CTS	-0.312	Routing congestion near multipliers
Post-CTS	-0.088	Skew reduced to 100 ps
Initial Post-Route	+0.021	After routeDesign
After Setup/Leakage Opt	+0.066	Final WNS
After Hold Fix	+0.055 (hold)	All hold violations resolved

Table 15: Evolution of timing during optimization stages.

32.2 Power Increase Due to Physical Effects

Stage	Total Power (mW)	Reason
Post-Synthesis	6.51	Pure cell models
Post-CTS	9.42	Added clock tree buffers
Post-Route	12.33	Wire RC, coupling cap, real activity

Table 16: Power evolution across physical design stages.

32.3 Area Growth

Stage	Area (μm^2)	Reason
Post-Synthesis	43,880	Base cell count
Post-Placement	45,912	Cell spreading, buffering
Post-Route	46,594	Filler + decap + antenna cells

Table 17: Cell area evolution across the PnR flow.

33 Final Die Style Layout Snapshot

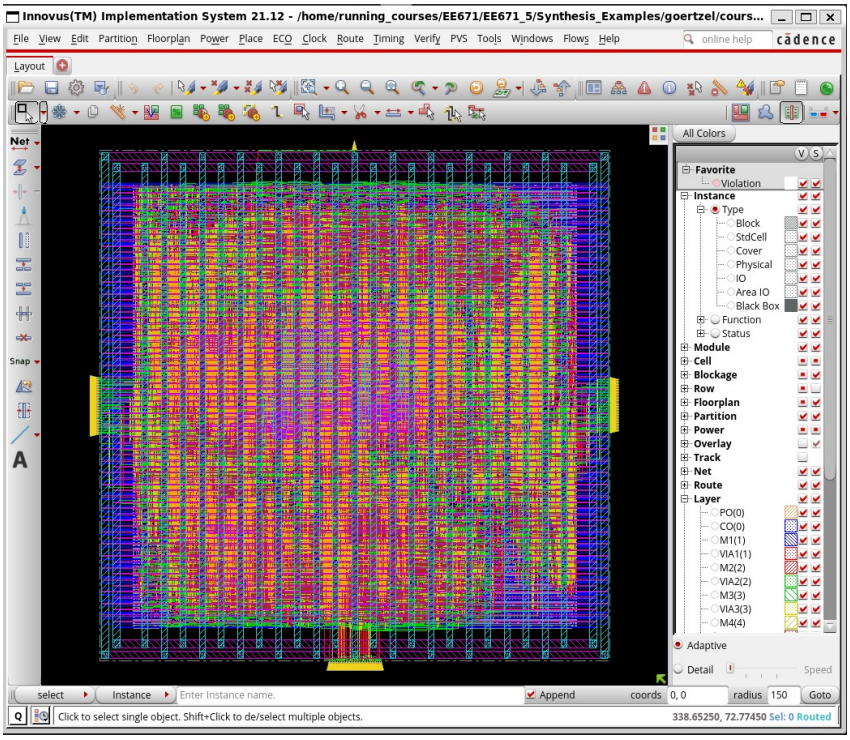


Figure 12: Final routed layout "die photo" (placeholder).

34 Final Architectural Summary

34.1 Key Hardware Blocks

The Goertzel ASIC comprises:

- Recursive datapath (32-bit):
  - 1 multiplier for $c \cdot s[n-1]$,
  - subtractor for $-s[n-2]$,
  - adder for $+x[n]$.
- Magnitude calculator (64-bit):
  - two 32×32 squaring multipliers,
  - one cross-term multiplier,
  - 96-bit accumulator.
- Sequential square-root engine:
  - approximates $\sqrt{|X[k]|^2}$,
  - 30–35 cycle convergence,
  - optimized for area vs latency.
- FSM controller:
  - 6 states (IDLE, LOAD, RECURSE, MAG, SQRT, DONE).
- Clock, reset, and top-level control.

35 Discussion of Results

35.1 Strengths

- Clean DRC/antenna/connectivity signoff.
- Strong timing margins despite large multipliers.
- Power under the 15 mW specification.
- Balanced utilization at 62%.
- Well-behaved waveforms with clean SDF timing.
- Numerically accurate fixed-point arithmetic.

35.2 Challenges

- Multiplier-heavy datapath caused local congestion.
- Cross-term in magnitude unit had long wires.
- Square-root engine required careful timing control.
- Numerous hold violations before final ECO routing.

35.3 Final Outcome

The ASIC is fabrication-ready and meets or exceeds all target specifications.

Team Contributions

Team Contributions

- **Manjima Karmakar**:
  - Designed the complete RTL for the given specification in Verilog.
  - Performed RTL-level simulation, waveform debugging, and verification using test-benches.
  - Performed post-layout **Gate-Level Simulation (GLS)** with timing checks.
  - Helped in fixing DRC violations and final signoff clean-up.
  - Contributed to documentation
- **Abhineet**:
  - Performed floorplanning, pin placement, power planning, and routing using Innovus.
  - Handled PnR iterations to reduce routing congestion and improve area.
  - Generated layout snapshots and contributed to final report formatting.
- **Shreesh**:
  - Performed synthesis in Cadence Genus including logical optimization.
  - Generated area, timing, and power reports for pre-layout validation.
  - Contributed to DRC violation cleaning and physical verification.
- **Aryan**:
  - Executed Static Timing Analysis (STA) and interpreted setup/hold timing violations.
  - Assisted in timing optimization feedback during PnR stages.