ANALOG DEVICES

AHEAD OF WHAT'S POSSIBLE™

FFT Implementation

Muhammed Adel Ahmed El-Sayed

Agenda

- Introduction
- FFT Architecture
- System Modeling
- RTL Design
- Design Verification
- FPGA Implementation
- ASIC Implementation

Introduction

FFT (Fast Fourier Transform) is an efficient algorithm that computes the Discrete Fourier Transform (DFT) much faster than direct calculation.

Why FFT rather than DFT ?

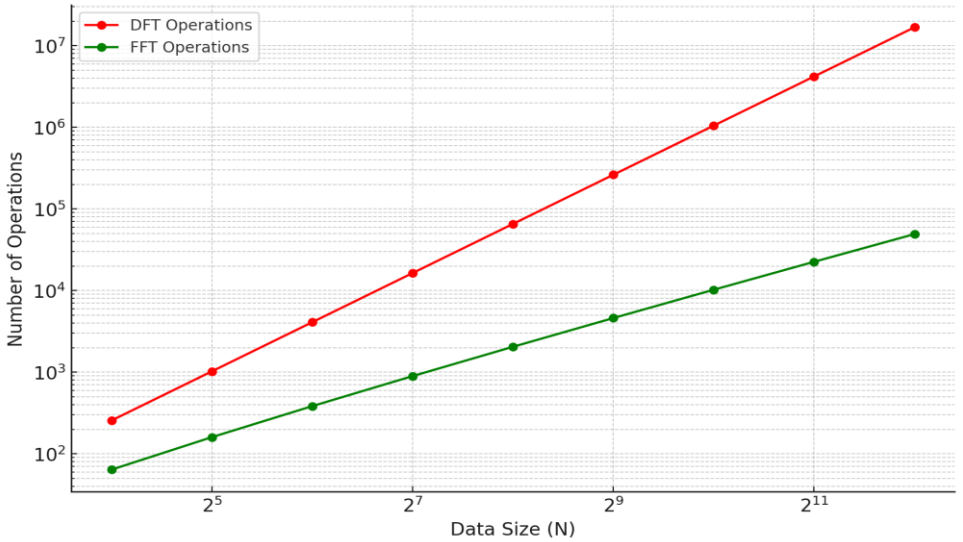
- Speed
- Efficiency
- Scalability

Key Statistics

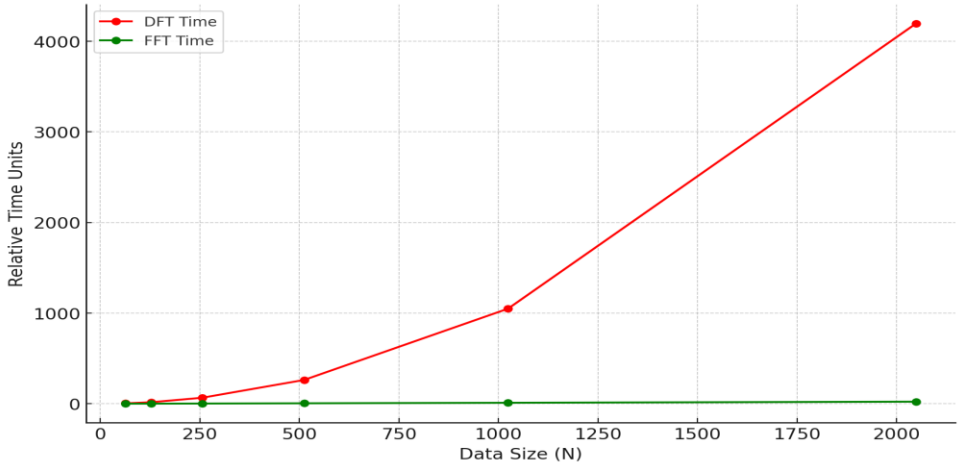
For N = 1024 ⟶ FFT is ~85x faster

For N = 4096 ⟶ FFT is ~341x faster

Complexity Improvement: $O(N^2) \longrightarrow O(N \log N)$



Computational Operations Required



Relative Execution Time

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

FFT Architecture

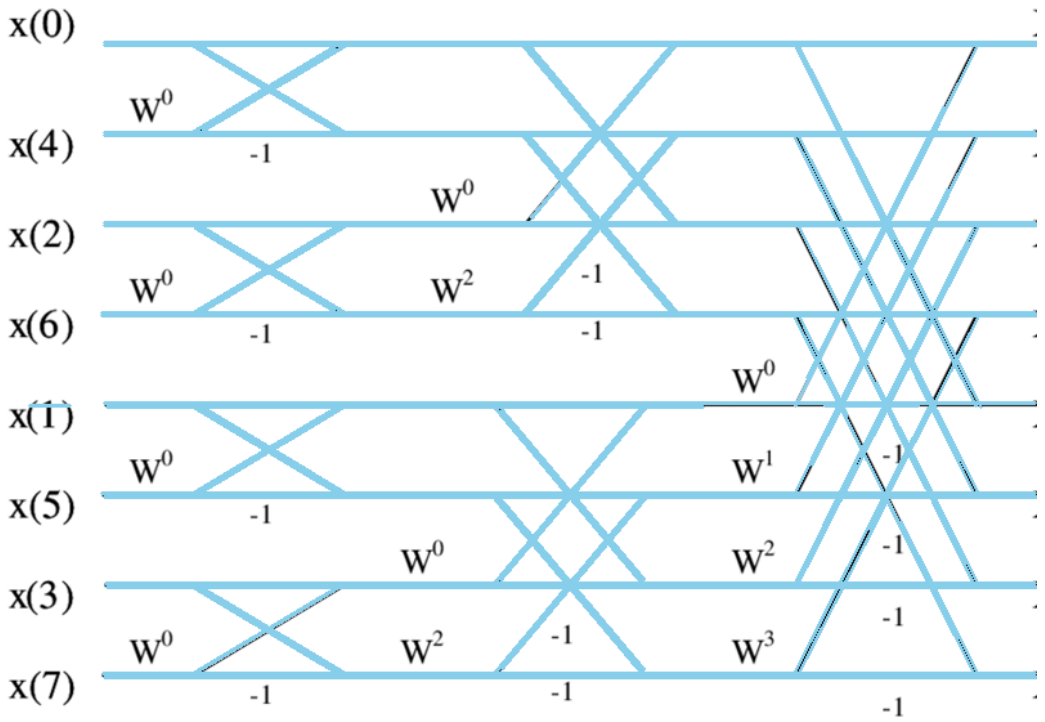
FFT Architecture

Algorithm

- Radix-2 Decimation in Time (DIT) based on the Cooley–Tukey algorithm.

Features

- Fully Pipelined: 3-stage pipeline for high throughput.
- FFT with 8 points each 16-bit width
- Fixed-Point Arithmetic
- Complexity: $O(8 \log_2(8))$



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

System Modeling

System Modeling

- FFT Core isolated

```
function y = fft_trans(x_in,T)
    % Input: x_in - 1x8 complex vector
    % Output: y - 1x8 complex vector (FFT result)
    x_in = cast(x_in, 'like', T.x);
    % STAGE 1: Bit-reversal and first butterfly operations
    % Bit-reverse input ordering: [0,4,2,6,1,5,3,7]
    x_stage1_in = cast([x_in(1), x_in(5), x_in(3), x_in(7), ...
                        x_in(2), x_in(6), x_in(4), x_in(8)], 'like', T.x_stage1_in);
    % First stage butterflies (twiddle factor W^0 = 1)
    x_stage1_out = cast(complex(zeros(1, 8)), 'like', T.x_stage1_out);
    x_stage1_out(1) = cast(x_stage1_in(1) + x_stage1_in(2), 'like', T.x_stage1_out); % x0 + x4
    x_stage1_out(2) = cast(x_stage1_in(1) - x_stage1_in(2), 'like', T.x_stage1_out); % x0 - x4
    x_stage1_out(3) = cast(x_stage1_in(3) + x_stage1_in(4), 'like', T.x_stage1_out); % x2 + x6
    x_stage1_out(4) = cast(x_stage1_in(3) - x_stage1_in(4), 'like', T.x_stage1_out); % x2 - x6
    x_stage1_out(5) = cast(x_stage1_in(5) + x_stage1_in(6), 'like', T.x_stage1_out); % x1 + x5
    x_stage1_out(6) = cast(x_stage1_in(5) - x_stage1_in(6), 'like', T.x_stage1_out); % x1 - x5
    x_stage1_out(7) = cast(x_stage1_in(7) + x_stage1_in(8), 'like', T.x_stage1_out); % x3 + x7
    x_stage1_out(8) = cast(x_stage1_in(7) - x_stage1_in(8), 'like', T.x_stage1_out); % x3 - x7
    % STAGE 2: Apply twiddle factors W^0, W^2 = -j
    x_stage2_out = cast(complex(zeros(1, 8)), 'like', T.x_stage2_out);
    x_stage2_out(1) = cast(x_stage1_out(1) + x_stage1_out(3), 'like', T.x_stage2_out); % W^0 = 1
    x_stage2_out(2) = cast(x_stage1_out(2) + cast(x_stage1_out(4) * cast(-1j, 'like', T.x_stage2_out), 'like', T.x_stage2_out), 'like', T.x_stage2_out); % W^2 = -j
    x_stage2_out(3) = cast(x_stage1_out(1) - x_stage1_out(3), 'like', T.x_stage2_out); % W^0 = 1
    x_stage2_out(4) = cast(x_stage1_out(2) - cast(x_stage1_out(4) * cast(-1j, 'like', T.x_stage2_out), 'like', T.x_stage2_out), 'like', T.x_stage2_out); % W^2 = -j
    x_stage2_out(5) = cast(x_stage1_out(5) + x_stage1_out(7), 'like', T.x_stage2_out); % W^0 = 1
    x_stage2_out(6) = cast(x_stage1_out(6) + cast(x_stage1_out(8) * cast(-1j, 'like', T.x_stage2_out), 'like', T.x_stage2_out), 'like', T.x_stage2_out); % W^2 = -j
    x_stage2_out(7) = cast(x_stage1_out(5) - x_stage1_out(7), 'like', T.x_stage2_out); % W^0 = 1
    x_stage2_out(8) = cast(x_stage1_out(6) - cast(x_stage1_out(8) * cast(-1j, 'like', T.x_stage2_out), 'like', T.x_stage2_out), 'like', T.x_stage2_out); % W^2 = -j
    % Compute twiddle factors as doubles
    W1d = exp(-1j * 2 * pi * 1 / 8);
    W2d = exp(-1j * 2 * pi * 2 / 8);
    W3d = exp(-1j * 2 * pi * 3 / 8);
    % Then cast to fixed-point
    W1 = cast(W1d, 'like', T.W1);
    W2 = cast(W2d, 'like', T.W2);
    W3 = cast(W3d, 'like', T.W3);
    y = cast(complex(zeros(1, 8)), 'like', T.y);
    y(1) = cast(x_stage2_out(1) + x_stage2_out(5), 'like', T.y); % W^0 = 1
    y(2) = cast(x_stage2_out(2) + cast(x_stage2_out(6) * W1, 'like', T.y), 'like', T.y); % W^1
    y(3) = cast(x_stage2_out(3) + cast(x_stage2_out(7) * W2, 'like', T.y), 'like', T.y); % W^2 = -j
    y(4) = cast(x_stage2_out(4) + cast(x_stage2_out(8) * W3, 'like', T.y), 'like', T.y); % W^3
    y(5) = cast(x_stage2_out(1) - x_stage2_out(5), 'like', T.y); % W^0 = 1
    y(6) = cast(x_stage2_out(2) - cast(x_stage2_out(6) * W1, 'like', T.y), 'like', T.y); % W^1
    y(7) = cast(x_stage2_out(3) - cast(x_stage2_out(7) * W2, 'like', T.y), 'like', T.y); % W^2 = -j
    y(8) = cast(x_stage2_out(4) - cast(x_stage2_out(8) * W3, 'like', T.y), 'like', T.y); % W^3
end
```

Outcomes

- Verified FFT algorithm
- Fixed-Point appropriate sizing

First Stage

Second Stage

Third Stage

System Modeling

```
% DESIGN PARAMETERS
L = 50; % Number of test cases
N = 8; % FFT size
nSeeds = 50; % Number of random seeds
```

1

```
% TEST INPUTS - Generate random complex input
 x_real = cast(randn(L, N),'like',T.x_real);
 x_imag = cast(randn(L, N),'like',T.x_imag);
 x = cast(x_real + cast(1j * x_imag, 'like', T.x), 'like', T.x);
```

2

```
if seed == 1
 buildInstrumentedMex fft_trans -args {x(seed, :) , T}
```

3

```
% Write test cases from first seed to file
fprintf('Writing test cases from seed 1 to file...\n');
for test_case = 1:L
    % Write each test case as a line with real and imaginary parts
    line_str = '';
    for n = 1:N
        if n == 1
            line_str = sprintf('%.6f+%.6fj', real(x(test_case, n)), imag(x(test_case, n)));
        else
            line_str = sprintf('%s, %.6f+%.6fj', line_str, real(x(test_case, n)), imag(x(test_case, n)));
        end
    end
    fprintf(test_cases_file, '%s\n', line_str);
end
end
```

4

```
% 8-POINT RADIX-2 FFT ALGORITHM (3 stages)
% Initialize output arrays
 y = cast(zeros(L, N),'like',T.y);
for test_case = 1:L
% Apply our custom 8-point FFT function
 y(test_case, :) = fft_trans_mex(x(test_case, :),T);
end
```

5

```
% VERIFY RESULTS against MATLAB's built-in FFT
 test_errors = zeros(L, 1);
 test_passed = true;
 signal_power_total = 0;
 noise_power_total = 0;
for test_case = 1:L
 y_expected = fft(double(x(test_case, :)));
 error_vector = y(test_case, :) - y_expected;
 error_magnitude = abs(mean(error_vector));
 test_errors(test_case) = error_magnitude;
```

System Modeling

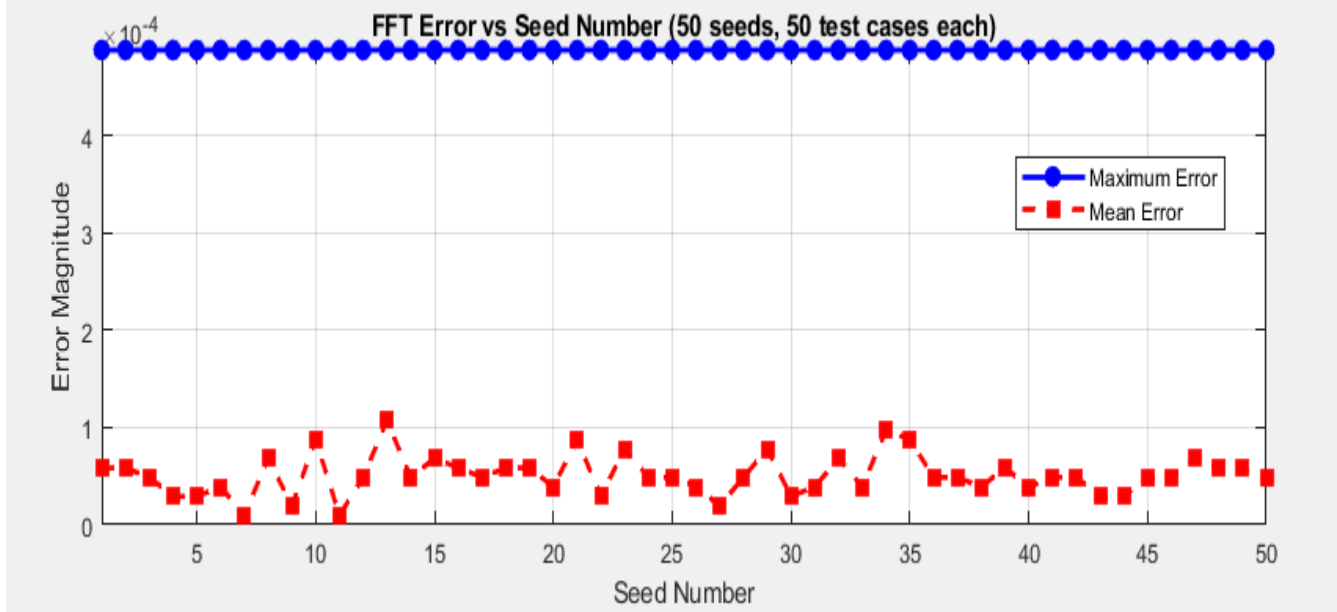
Name	Type	Size	Class	DT Mode	Signednes	WL	FL	Proposed Signednes	Proposed WL	Proposed FL
y	Output	1 × 8	complex embedded.fi	-	Signed	16	11	-	-	-
▸ T	Input	1 × 1	struct	-	-	-	-	-	-	-
x_in	Input	1 × 8	complex embedded.fi	-	Signed	16	12	-	-	-
W1	Local	1 × 1	complex embedded.fi	-	Signed	16	15	-	-	-
W1d	Local	1 × 1	complex double	-	-	-	-	Signed	32	31
W2	Local	1 × 1	complex embedded.fi	-	Signed	16	15	-	-	-
W2d	Local	1 × 1	complex double	-	-	-	-	Signed	32	31
W3	Local	1 × 1	complex embedded.fi	-	Signed	16	15	-	-	-
W3d	Local	1 × 1	complex double	-	-	-	-	Signed	32	31
x_stage1_in	Local	1 × 8	complex embedded.fi	-	Signed	16	12	-	-	-
x_stage1_out	Local	1 × 8	complex embedded.fi	-	Signed	16	12	-	-	-
x_stage2_out	Local	1 × 8	complex embedded.fi	-	Signed	16	11	-	-	-

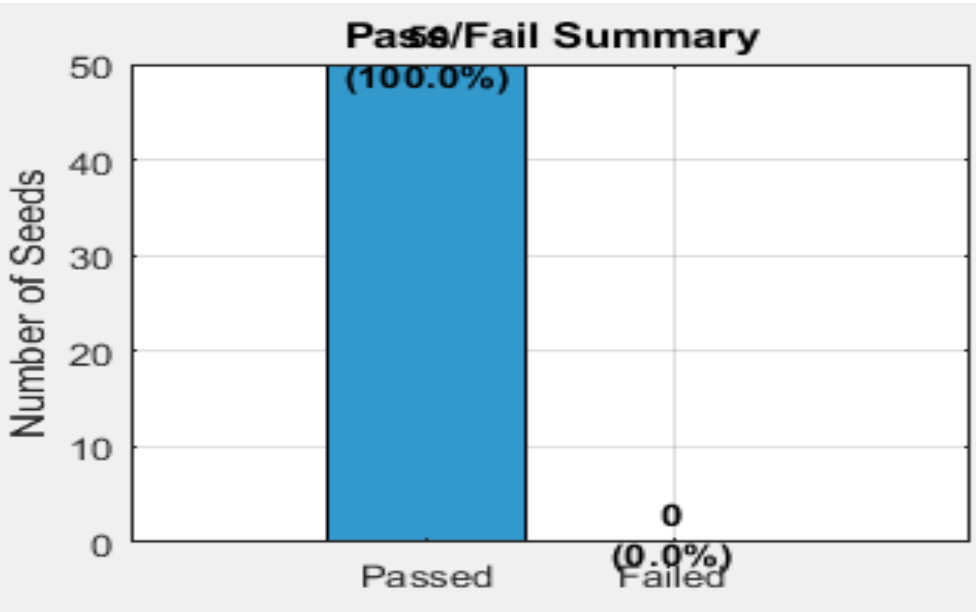
```
case 'FxPt'
    T.x_real = fi([], 1, 4 + 12, 12);
    T.x_imag = fi([], 1, 4 + 12, 12);
    T.x = fi([], 1, 4 + 12, 12);
    T.x_stage1_in = fi([], 1, 4 + 12, 12);
    T.x_stage1_out = fi([], 1, 4 + 12, 12);
    T.x_stage2_out = fi([], 1, 5 + 11, 11);
    T.W1 = fi([], 1, 1 + 15, 15);
    T.W2 = fi([], 1, 1 + 15, 15);
    T.W3 = fi([], 1, 1 + 15, 15);
    T.y = fi([], 1, 5 + 11, 11);
```

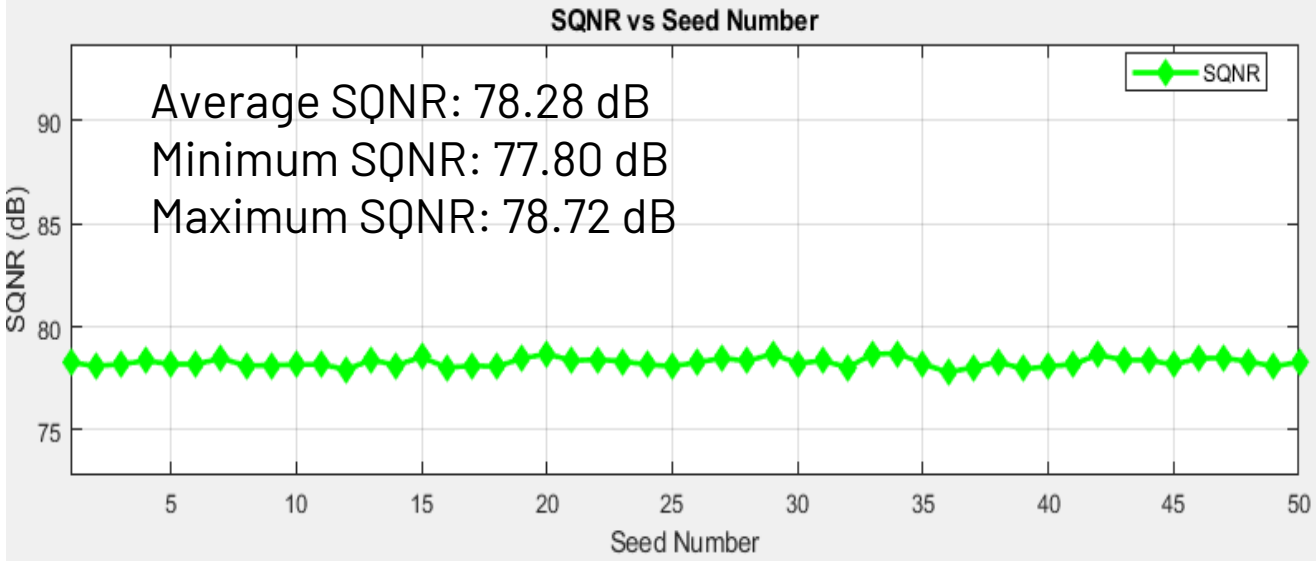
- Seeds passed: 50/50 (100.0%)
- Seeds failed: 0/50 (0.0%)
- Overall maximum error: 4.88e-04
- Overall mean error: 5.12e-05

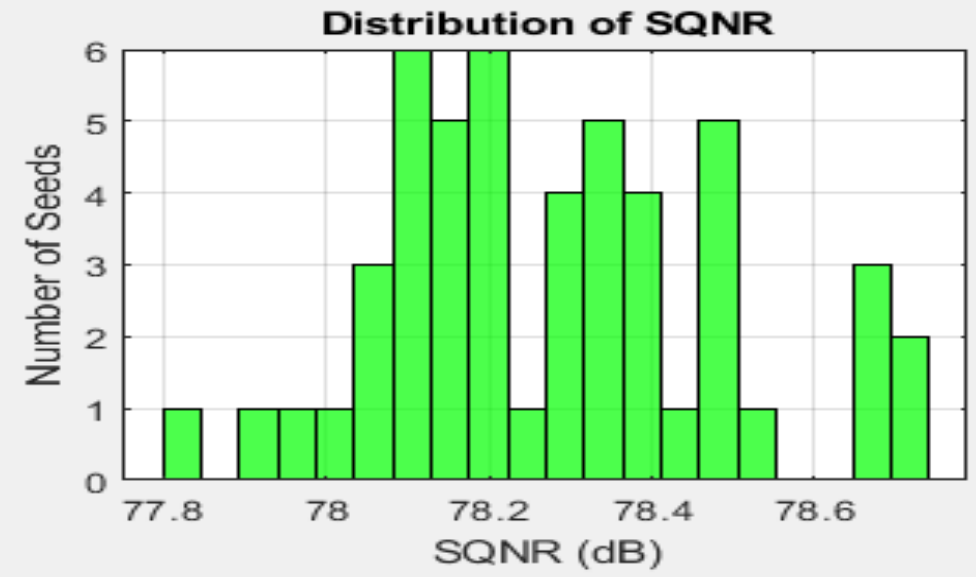
```
Running FFT error analysis for 50 seeds...

Writing test cases from seed 1 to file...
Writing test outputs from seed 1 to file...
Seed 1: All 50 test cases passed! Max error: 4.88e-04, Mean error: 5.86e-05, SQNR: 78.3 dB
Seed 2: All 50 test cases passed! Max error: 4.88e-04, Mean error: 5.86e-05, SQNR: 78.1 dB
Seed 3: All 50 test cases passed! Max error: 4.88e-04, Mean error: 4.88e-05, SQNR: 78.2 dB
Seed 4: All 50 test cases passed! Max error: 4.88e-04, Mean error: 2.93e-05, SQNR: 78.4 dB
Seed 5: All 50 test cases passed! Max error: 4.88e-04, Mean error: 2.93e-05, SQNR: 78.2 dB
Seed 6: All 50 test cases passed! Max error: 4.88e-04, Mean error: 3.91e-05, SQNR: 78.2 dB
Seed 7: All 50 test cases passed! Max error: 4.88e-04, Mean error: 9.77e-06, SQNR: 78.5 dB
Seed 8: All 50 test cases passed! Max error: 4.88e-04, Mean error: 6.84e-05, SQNR: 78.1 dB
```

System Modeling





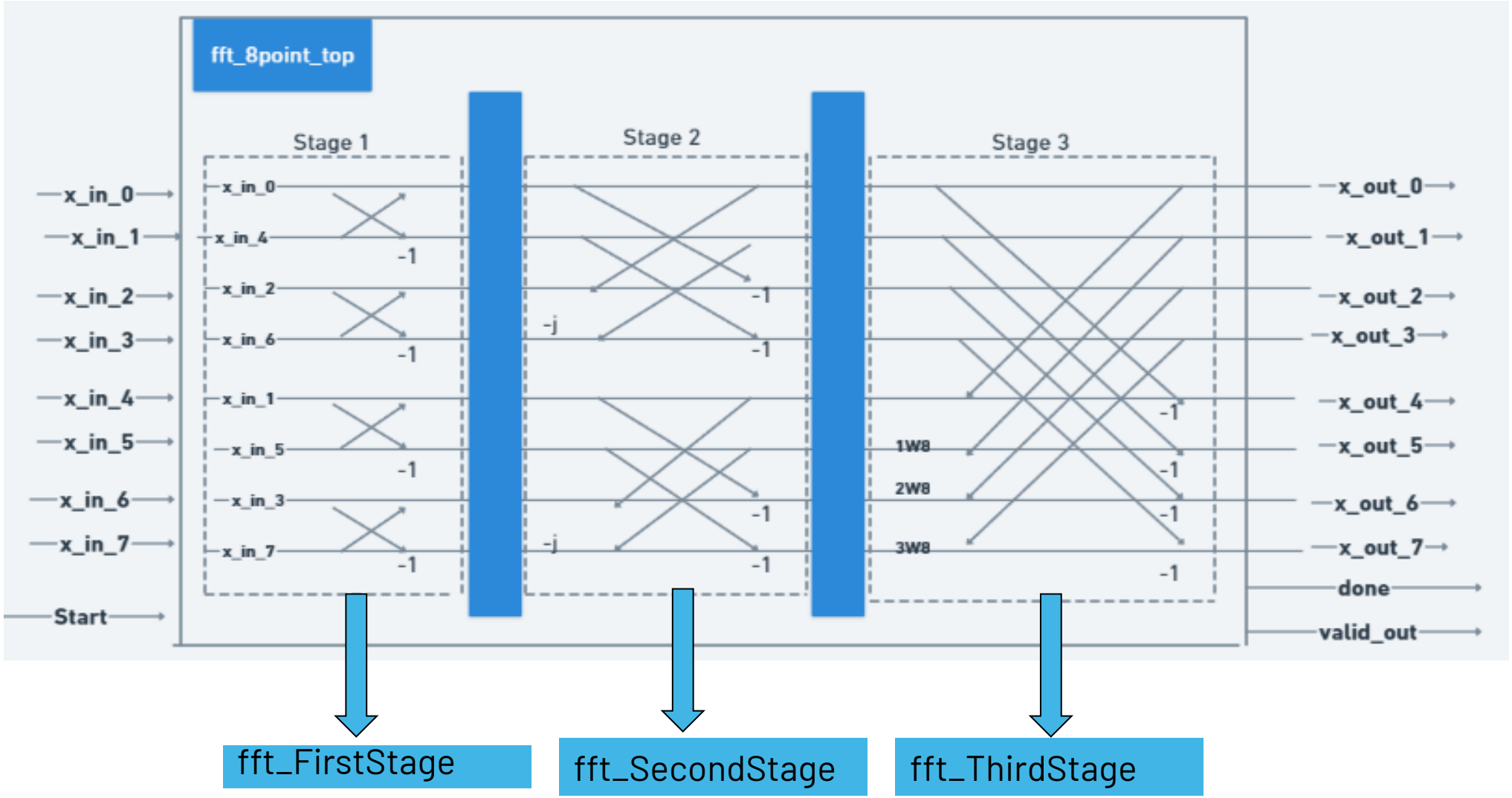




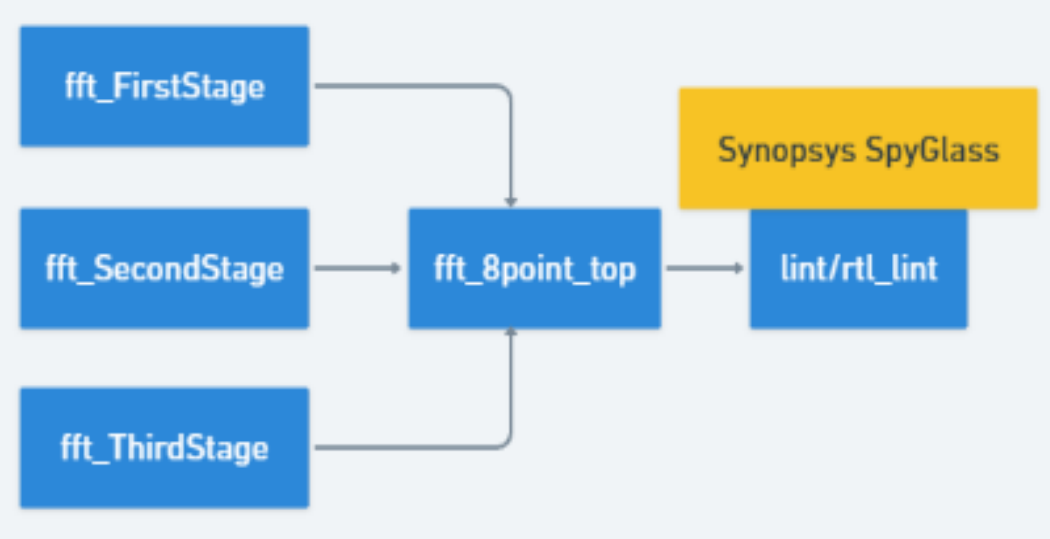
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

RTL Design

RTL Design



RTL Design




```
Goal Name         : lint/lint_rtl
Scenario Name     : default_scenario
Top               : fft_8point_top

Message Summary
Severity    Non-Waived  Waived
FATAL            0          0
ERROR            0          0
WARNING          4          0
INFO             2          0
```

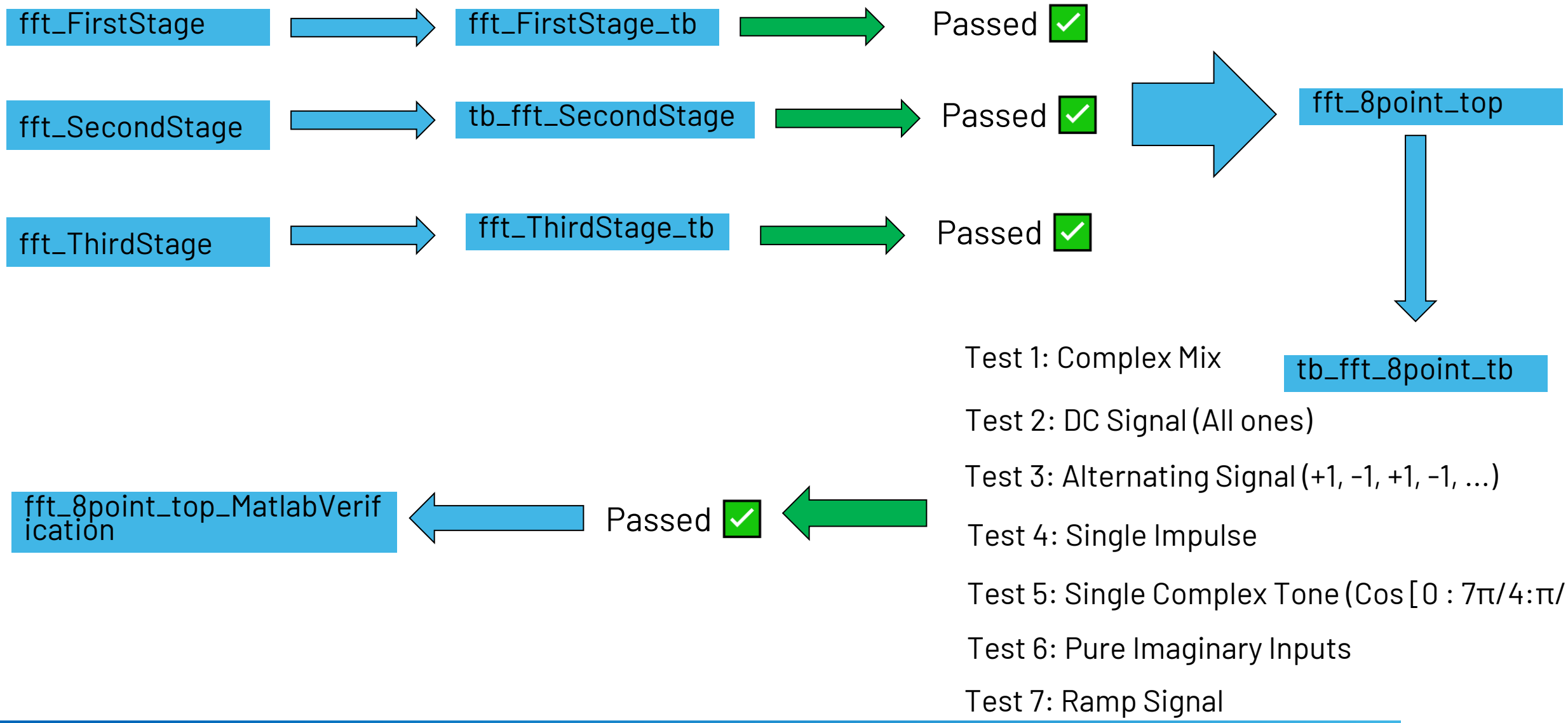
Lint-Clean RTL ✅

```
Project Name      : spyglass-1
Goal Name         : lint/lint_abstract
Scenario Name     : default_scenario
Top               : fft_8point_top

Message Summary
Severity    Non-Waived  Waived
FATAL            0          0
ERROR            0          0
WARNING          0          0
INFO             3          0
```

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

Design Verification

Design Verification



Design Verification

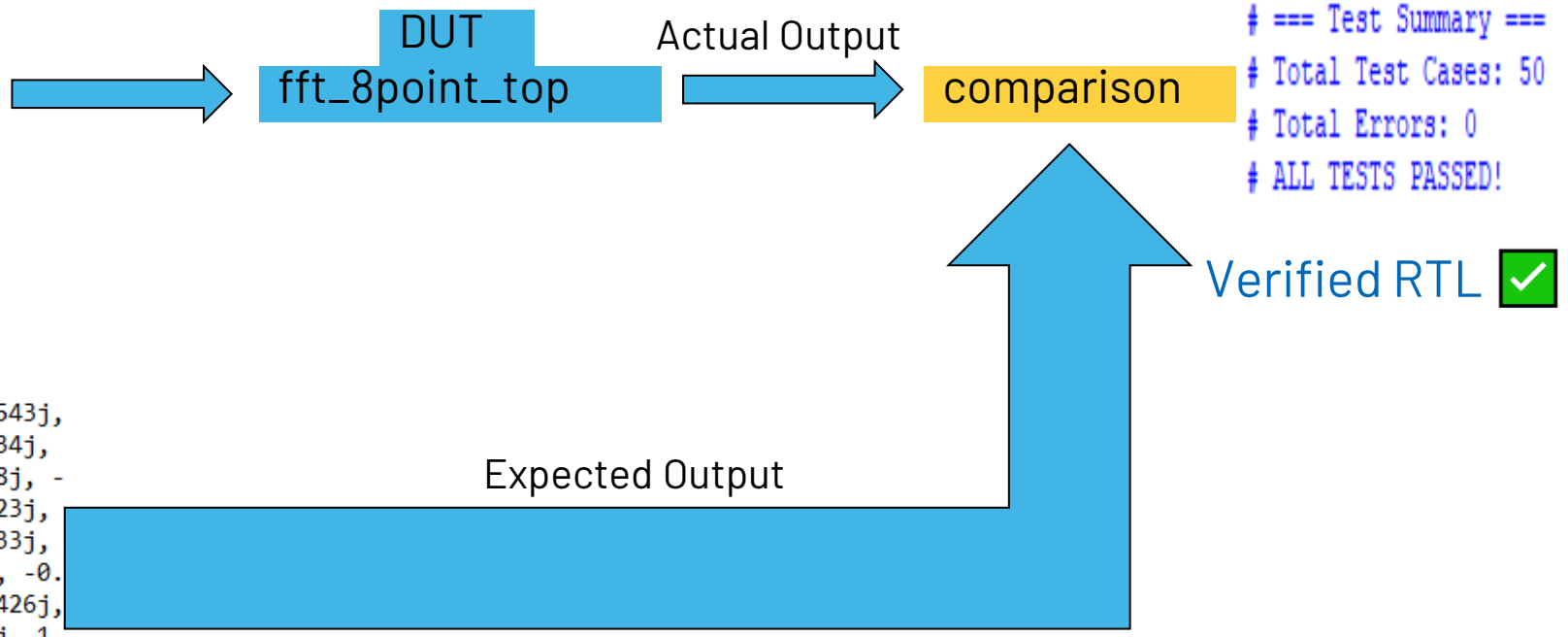
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

Test inputs from MATLAB

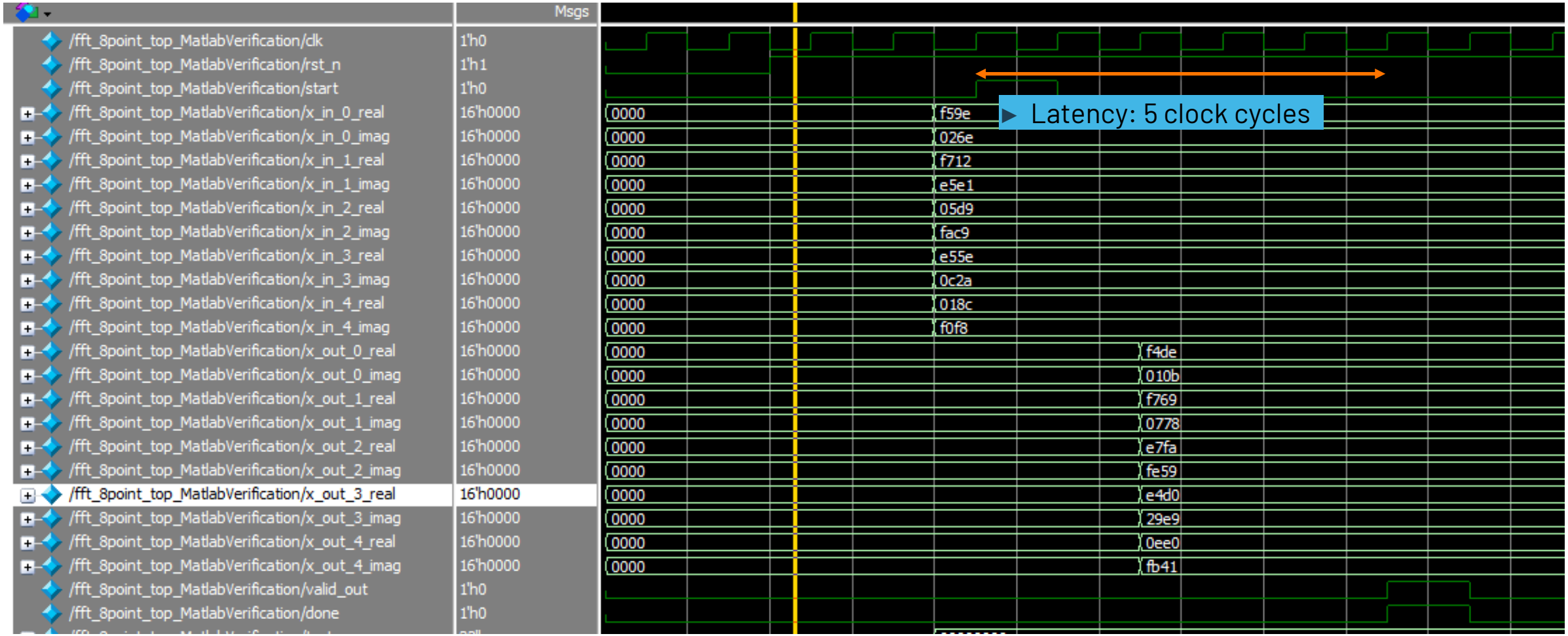
```
-0.648926+0.151855j, -0.558105+-1.632568j, 0.365479+-0.325928j,
1.181152+0.816406j, -0.028564+-0.851807j, -1.097168+0.283447j,
-0.758545+0.577881j, -1.476318+1.077393j, 1.930176+-0.011475j,
-1.109619+1.063965j, 0.258789+-0.440186j, 0.623047+0.036377j, 1
-0.845459+-0.803223j, -2.018799+3.327881j, 0.657227+-1.098389j,
-0.572754+0.798584j, 0.199707+0.103760j, -1.463379+0.375732j, -
-0.558594+-0.315186j, 0.425781+-0.773193j, 0.854004+0.170166j,
0.178467+1.186035j, -1.270020+0.677490j, 0.580566+1.139648j, 1.
-0.196777+0.084961j, -0.485107+-0.696777j, -0.918701+1.657471j,
0.586426+-1.425049j, 0.594238+-0.075439j, 0.794922+0.338135j, -
-0.851807+-1.454102j, -0.276367+0.026123j, 0.517578+-1.819092j,
0.800293+-1.809326j, -1.857666+2.005615j, 0.494629+-0.802246j,
-1.509521+-1.462891j, 0.040771+-1.129639j, 0.663818+-0.574463j,
0.875977+-0.138184j, 0.282959+0.654297j, -0.710205+-0.234375j,
-0.242676+-1.005615j, 0.063477+1.419189j, -1.306885+-0.305176j,
0.166748+-2.733398j, 0.433350+-0.825684j, -0.741699+1.639404j,
-1.965332+1.692383j, 0.422852+0.053223j, -1.467773+1.897217j,
```

Test outputs from MATLAB

```
-1.391602+0.130371j, -1.073730+0.933594j, -3.002930+-0.206543j,
-0.768066+-2.215332j, 0.014160+2.951660j, 0.687988+-1.740234j,
2.857910+4.188965j, -5.115723+-1.055176j, 0.695312+1.236328j, -
5.167969+-0.250977j, -1.511230+3.386719j, -6.001953+2.061523j,
-2.488281+0.628418j, -2.189453+3.493164j, 4.189941+-0.661133j,
1.329590+4.616699j, 0.064941+1.900391j, 2.053711+0.855957j, -0.
0.216309+-2.314453j, 2.073730+-4.678223j, -1.747070+-0.461426j,
0.310059+3.907227j, 0.912109+-0.804199j, 0.315430+3.882324j, 1.
-4.809570+3.811523j, 1.628906+-1.684570j, -0.211426+-3.821289j,
-1.508789+-1.223145j, 1.875000+-4.049316j, 3.814941+-4.488281j,
-5.193848+-3.022949j, -2.836426+-2.666016j, -2.504395+-0.967285
1.225098+3.851562j, -5.668945+-2.342285j, 5.317871+-1.139160j,
1.258301+0.124023j, -4.170898+-3.243652j, -3.205078+1.403320j,
-1.843262+0.449219j, 1.953613+0.048340j, 0.339844+0.205078j, 2.
-3.346191+1.426758j, -0.407715+0.640625j, 2.898926+-3.896973j,
-0.403809+-1.485840j, 2.315430+-4.730957j, 2.539062+-0.273926j,
```



Design Verification



Design Verification

Code Coverage

```
================================================================================
=== File: fft_8point_top.v
================================================================================
Statement Coverage:
    Enabled Coverage          Active      Hits    Misses % Covered
    ----------------          ------      ----    ------ ---------
    Stmts                         16        16         0     100.0
```

```
Branch Coverage:
    Enabled Coverage          Active      Hits    Misses % Covered
    ----------------          ------      ----    ------ ---------
    Branches                       9         8         1      88.8
```

```
Condition Coverage:
    Enabled Coverage          Active   Covered    Misses % Covered
    ----------------          ------      ----    ------ ---------
    FEC Condition Terms            2         1         1      50.0
```

```
Toggle Coverage:
    Enabled Coverage          Active      Hits    Misses % Covered
    ----------------          ------      ----    ------ ---------
    Toggle Bins                 2066      2065         1      99.9
```

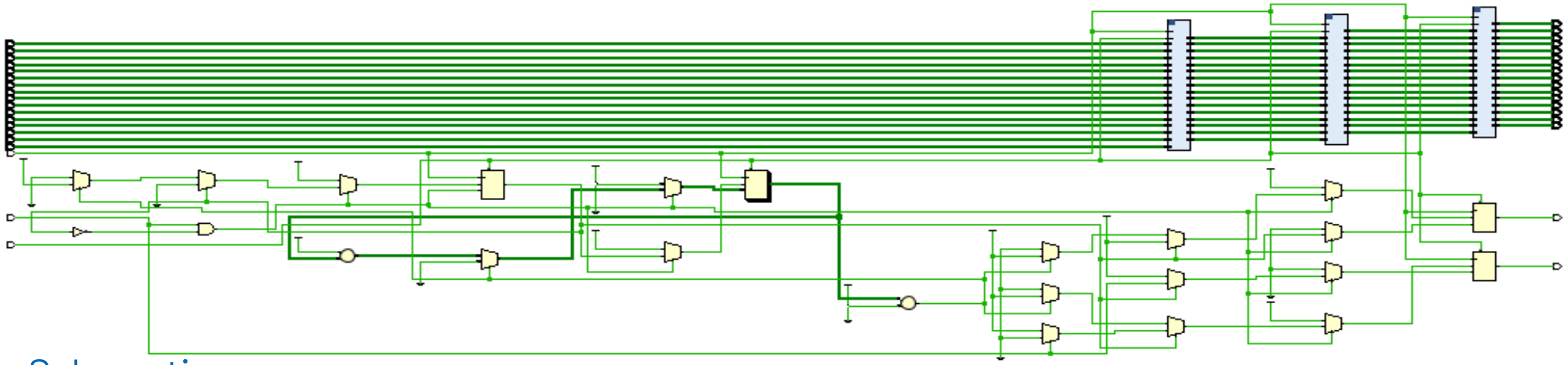
ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

FPGA Implementation

FPGA Implementation

Design Elaboration

Artix-7 AC701 Evaluation Platform (xc7a200tfbg676-2)

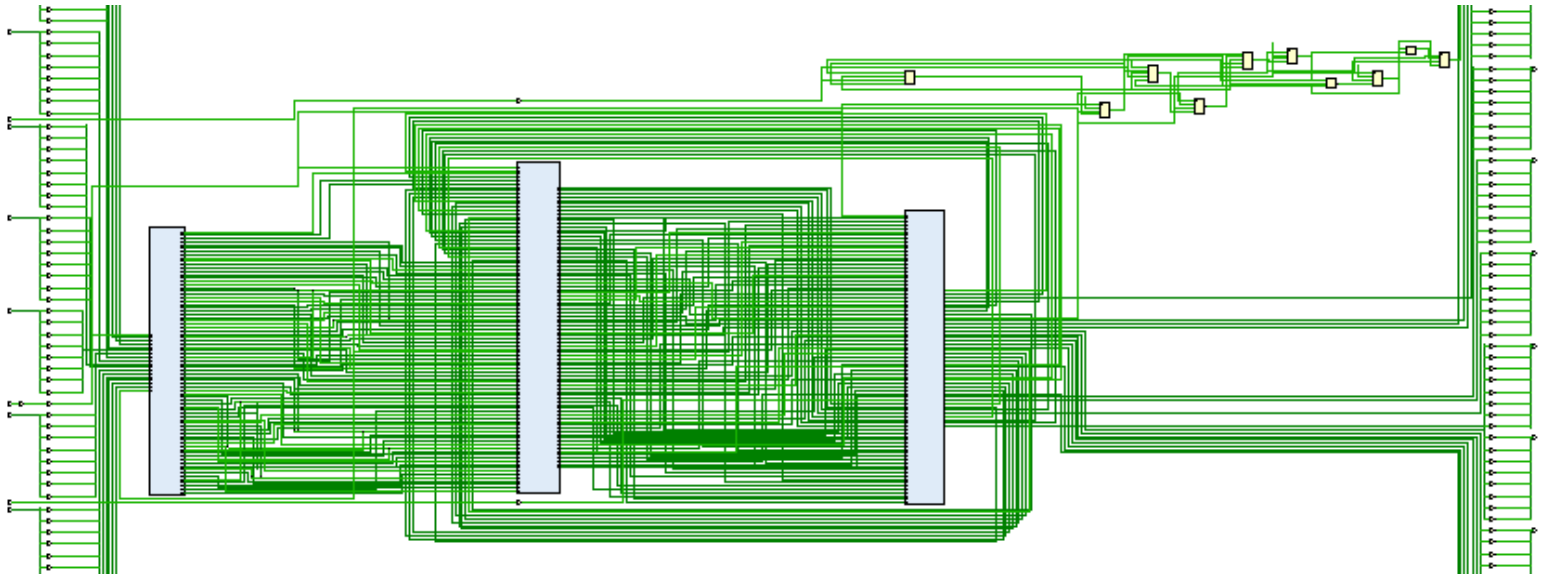


Design Schematic

275 cells

261 I/O ports

1189 Net



FPGA Implementation

Utilization Design Information

Ref Name	Used	Functional Category
LUT2	489	LUT
FDCE	389	Flop & Latch
CARRY4	170	CarryLogic
IBUF	131	IO
OBUF	130	IO
LUT1	63	LUT
LUT4	55	LUT
LUT3	33	LUT
LUT5	22	LUT
LUT6	18	LUT
BUFG	1	Clock

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	606	0	134600	0.45
LUT as Logic	606	0	134600	0.45
LUT as Memory	0	0	46200	0.00
Slice Registers	389	0	269200	0.14
Register as Flip Flop	389	0	269200	0.14
Register as Latch	0	0	269200	0.00
F7 Muxes	0	0	67300	0.00
F8 Muxes	0	0	33650	0.00

Timing summary

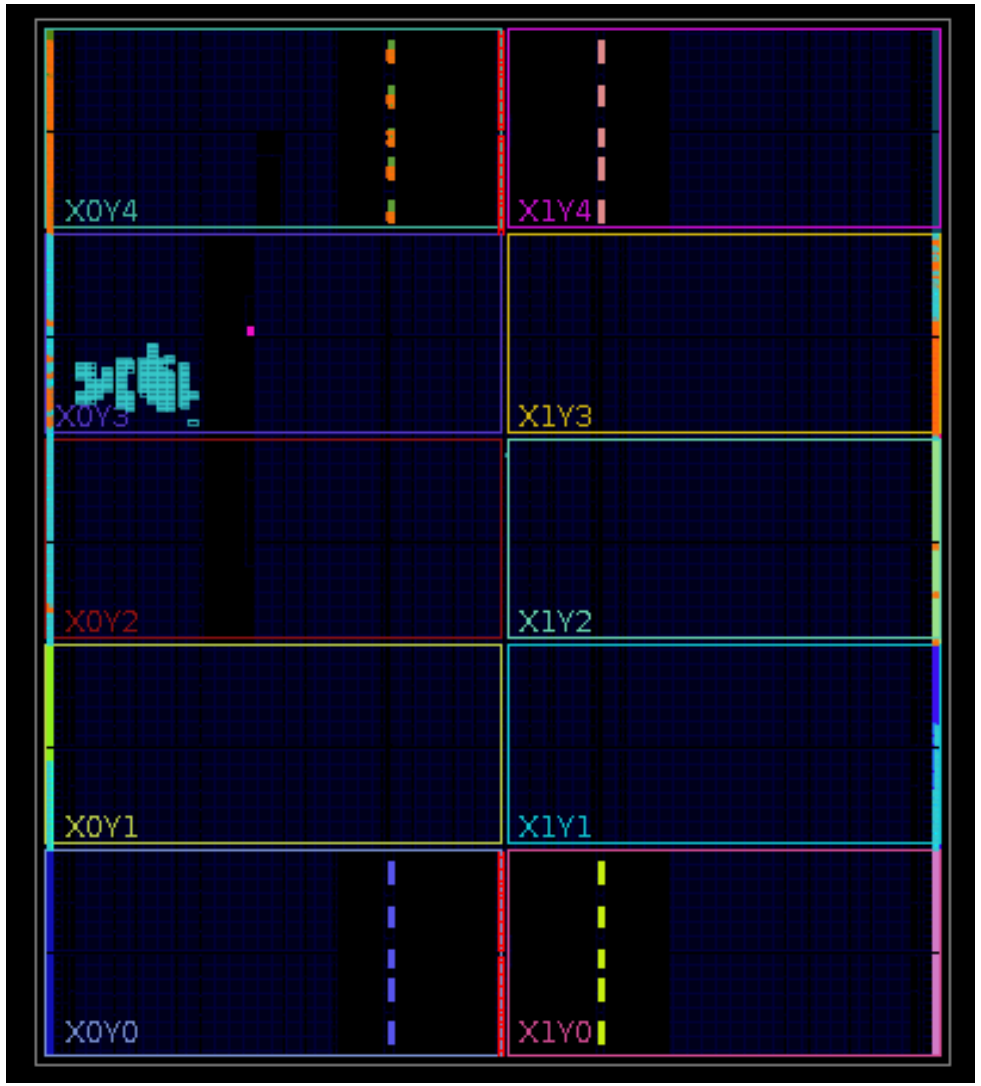
Max frequency: 143 MHz

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	0.008 ns	Worst Hold Slack (WHS):	0.127 ns	Worst Pulse Width Slack (WPWS):	3.000 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	262	Total Number of Endpoints:	262	Total Number of Endpoints:	390

All user specified timing constraints are met.

FPGA Implementation

Implemented Design



ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

ASIC Implementation

ASIC Implementation – Openlane

Timing Results

Max frequency : 118 MHz

Parameter	Post-Routing
Setup Slack (ns)	0.45
Hold Slack (ns)	0.11
TNS (Total Negative Slack)	0.00

Design Area

Parameter	Post-Routing
Design Area (μm^2)	197,894
Utilization (%)	5%

Power Results

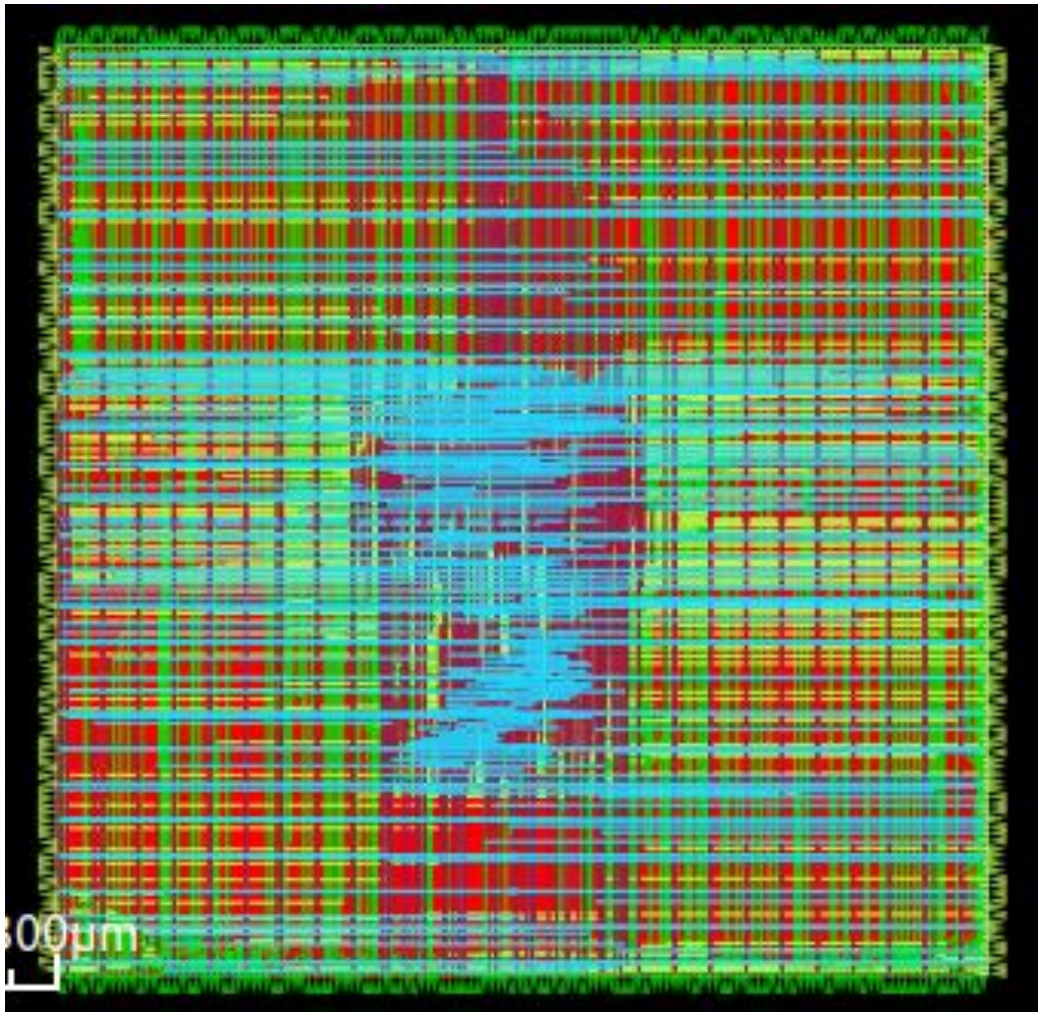
Parameter	Post-Routing
Total Power (Watts)	0.295
Internal Power (%)	38.1%
Switching Power (%)	61.9%
Sequential Logic Power (%)	8.9%
Combinational Logic Power (%)	91.1%

ASIC Implementation – Openlane

Max frequency : 118 MHz

PDKs: sky130A 130nm

Std cell library: sky130_fd_sc_hd



ASIC Implementation – ICC

Timing Results

Max frequency : 1 GHz

Parameter	Post-Routing
Setup Slack (ns)	0.00
Hold Slack (ns)	0.01
TNS (Total Negative Slack)	0.00

Design Area

Parameter	Post-Routing
Design Area (μm^2)	20,360
Utilization (%)	30%

Power Results

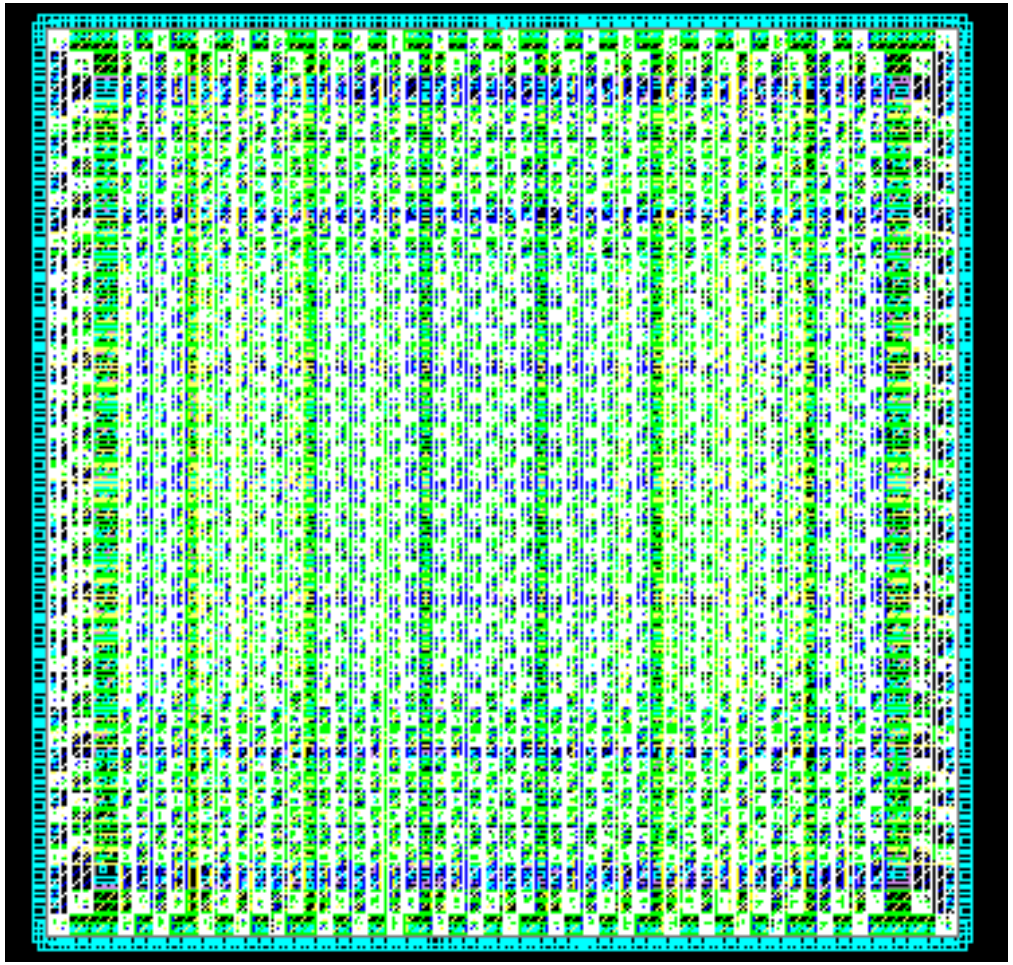
Parameter	Post-Routing
Total Power (Watts)	13.45 mW
Internal Power (%)	8.5285 mW
Switching Power (%)	14.5331 mW
Sequential Logic Power (%)	4.8 mW
Combinational Logic Power (%)	8.6519 mW

ASIC Implementation – ICC

Max frequency : 1 GHz

PDKs: Nangate 45nm

Std cell library: NangateOpenCellLibrary



ASIC Implementation

Timing Results

Parameter	Openlane	ICC
Max Frequency	118 MHz	1 GHz
Setup Slack	0.45	0.00
Hold Slack	0.11	0.01
TNS (Total Negative Slack)	0.00	0.00

Design Area

Parameter	Openlane	ICC
Design Area	197,894	20,360
Utilization	5%	30%

Power Results

Parameter	Openlane	ICC
Total Power	295	13.45
Internal Power	112.5	8.5285
Switching Power	182.6	14.5331
Sequential Logic Power	26.3	4.8
Combinational Logic Power	268.7	8.6519

References

DERIVATION OF THE RADIX-2 FFT ALGORITHM | Chapter Four. The Fast Fourier Transform

Can anyone explain to me the concept of DFT and FFT please | Forum for Electronics

ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

Thank You!