Empowering DTCO Innovation with AI and Machine Learning

Future of DTCO.ML™ and DTCO.GenAI™

(Draft)

Hock Chen
崛智科技股份有限公司
DIGWISE TECHNOLOGY CORPORATION, LTD.

Table of Contents

AI-Driven Semiconductor Chip Design Efficiency and Productivity Revolution

Preface

In the era of AI-driven innovation, the semiconductor industry is shifting from traditional yield optimization to comprehensive productivity enhancement. To stay competitive, chip design now focuses on enhancing performance, production capacity, and market competitiveness, beyond just yield and cost. This book explores how AI can optimize design margins, timing signoff, testing strategies, data analysis, process adjustments, binning strategies, and system-level compensation, driving a revolution in semiconductor design and productivity.

Book Positioning

This book approaches semiconductor design from a physical implementation perspective, focusing on the application of machine learning (ML) in Design-Technology Co-Optimization (DTCO). It covers traditional ML and cutting-edge generative AI (GenAI), exploring strategies to enhance chip performance and productivity.

Target Audience

Semiconductor Professionals: Chip design engineers, EDA developers, and researchers, offering practical examples and insights.

Cross-Disciplinary Researchers: Helping them understand DTCO and explore AI applications.

Book Structure

We introduce DTCO principles, discuss machine learning applications in optimization, and analyze how generative AI shapes the future of semiconductor design. The book also explores innovative EDA development, showing how new technologies improve design efficiency and performance.

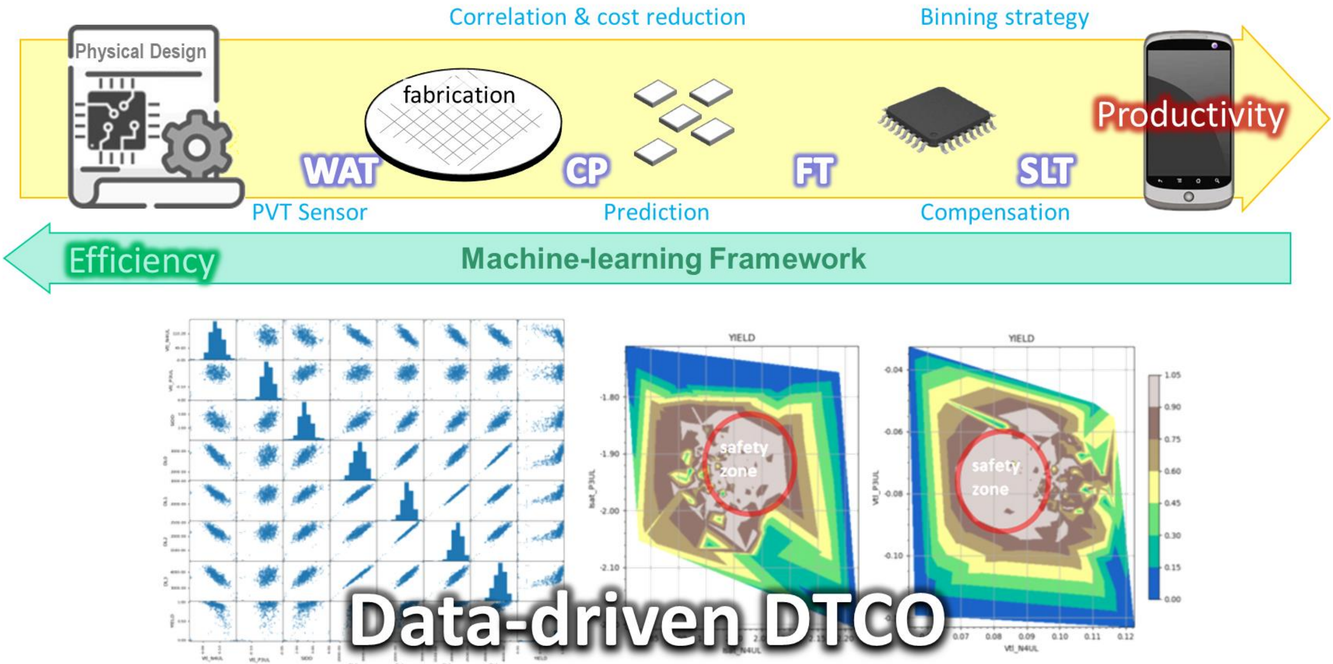
Part I: Design Technology Co-Optimization, DTCO

Chapter 1. Overview and Evolution of DTCO

Design-Technology Co-Optimization (DTCO) is widely used in semiconductor physical design to improve chip manufacturing efficiency and competitiveness. As illustrated in Fig. 1-1, the DTCO process can be compared to optimizing a large-scale neural network.

During the forward inference phase, the primary objective is to maximize productivity. This includes chip monitoring, Wafer Acceptance Test (WAT), Chip Probe (CP)/Final Test (FT), System-Level Test (SLT), feature correlation analysis, and machine learning, ultimately optimizing yield and performance.

In the back-propagation optimization phase, the focus shifts to improving chip efficiency. This involves calibrating chip models with real measurement data, adjusting process parameters, refining timing extraction (Re-K) based on WAT results, customizing and optimizing standard cell libraries, performing On-Chip Variation (OCV) regression analysis, and enhancing design margins and timing signoff strategies.



US 11144695, 11880643, 11506714, TW I700598, I769829

Fig. 1-1 Unleashing Productivity and Efficiency with DTCO

1.1. Principles of Design for Productivity

Common defects in the wafer manufacturing process include symmetrical resonance patterns (such as concentric ripples or donut-shaped waves), scratches from the wafer polishing process (Polish Pattern), oxide layer unevenness and tilt, as well as system-level defects (Test Site Pattern) caused by mask exposure interference and test environment mismatches (e.g., Load Board and Probe Card).

These defects, combined with physical random variations, significantly impact device behavior. As shown in Fig. 1-2, key parameters such as SIDD (leakage current) and RO (Ring Oscillator) can be affected, with analog monitoring circuits (e.g., V/T Sensors) being particularly sensitive to test environment mismatches. While the industry often focuses on fine-tuning microscopic parameters (such as metal layer thickness and parasitic capacitance), systemic challenges that erode design margins and their corresponding mitigation strategies are frequently overlooked.

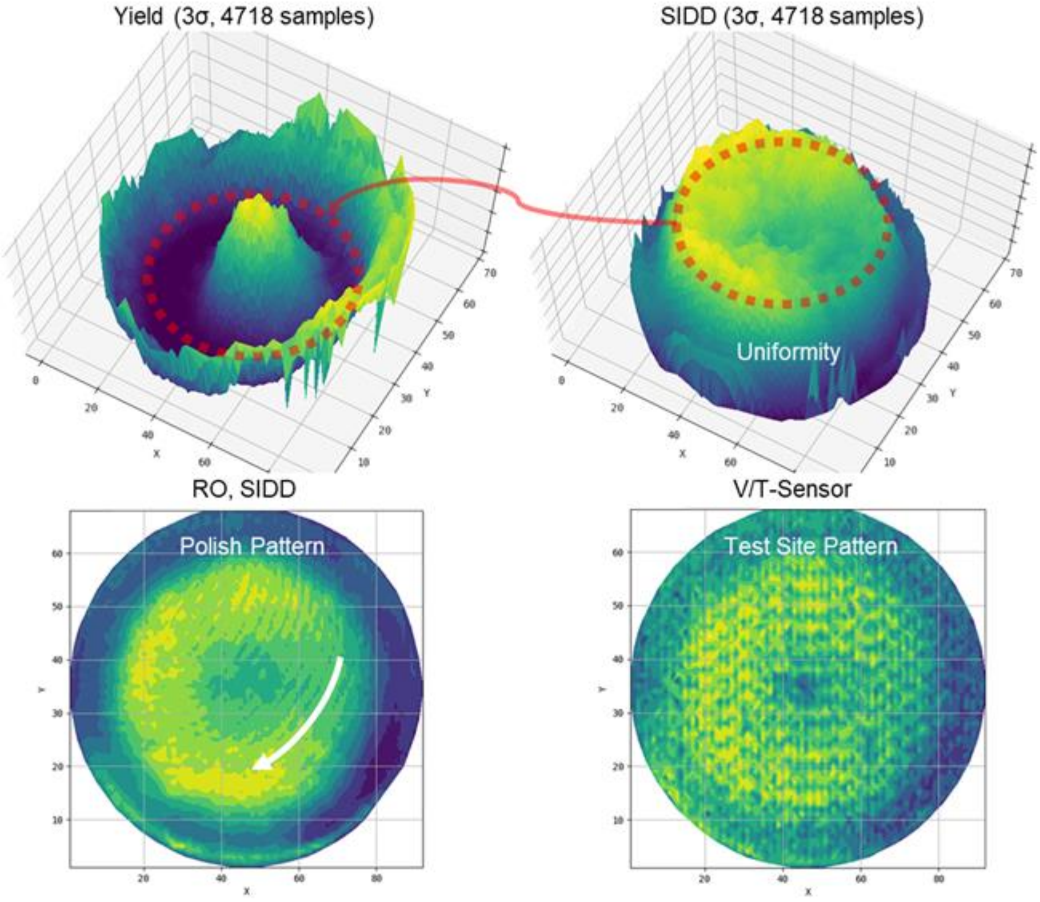


Fig. 1-2 Impact of Wafer Defects on System Performance Metrics

To effectively tackle these systemic challenges, WAT/CP analysis combined with Process Control Management (PCM) is used to continuously monitor process parameters and chip test data during mass production. The focus is on the distribution, probability density, and SPICE-to-Silicon (S2S) analysis of each parameter, facilitating systematic compensation and post-silicon adjustments. By mapping and correlating WAT and CP data, process optimization windows are identified, enabling the development of chip performance grading and binning strategies, and applying compensation techniques to enhance production efficiency, yield, and chip competitiveness.

1.2. Design Methodology for Ultimate Efficiency

During the design phase, process and cell library analysis optimize the drive strength, area, and power consumption of key components in the chip architecture, improving overall reliability. Customized design and microarchitecture optimization are applied to critical components, while on-chip sensors implement defense strategies to address process variations and dynamic voltage drop (IR-drop), ensuring design robustness. In the production phase, machine learning techniques analyze actual data distributions, enabling the development of timing Re-K and signoff strategies (Timing Signoff) to better understand process uniformity and OCV, achieving optimal design margins.

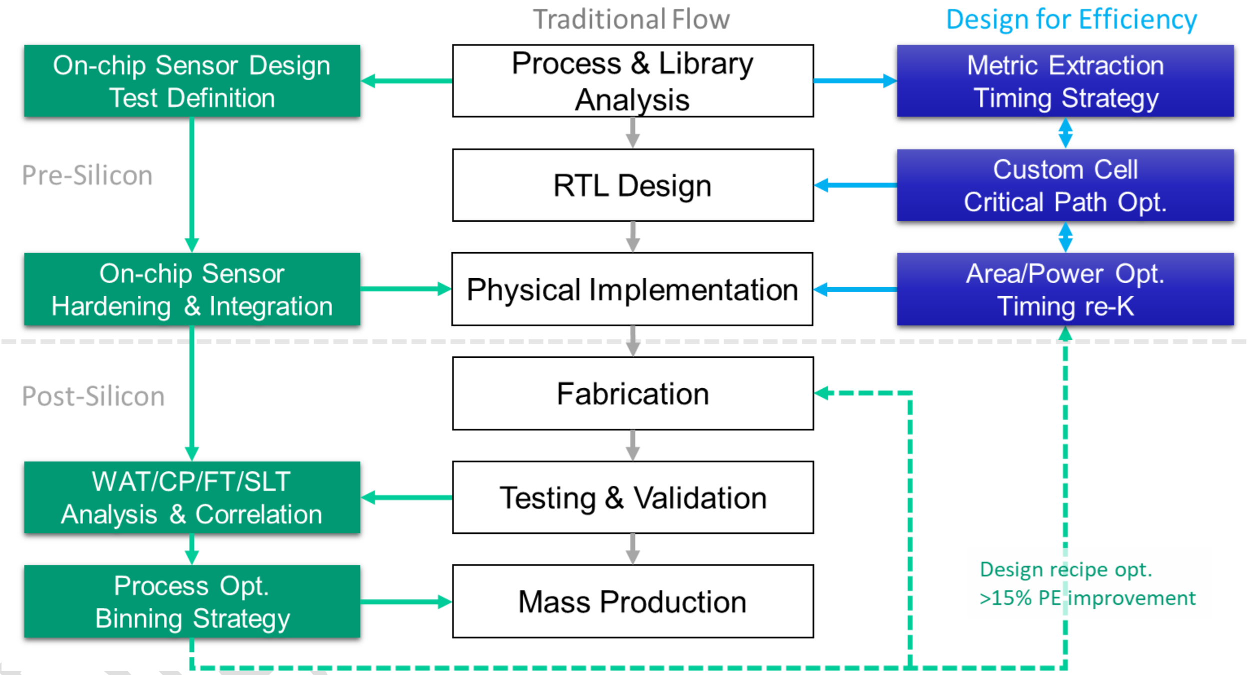

Traditional Flow
Design for Efficiency
On-chip Sensor Design Test Definition
Process & Library Analysis
Metric Extraction Timing Strategy
Pre-Silicon
RTL Design
Custom Cell Critical Path Opt.
On-chip Sensor Hardening & Integration
Physical Implementation
Area/Power Opt. Timing re-K
Post-Silicon
Fabrication
WAT/CP/FT/SLT Analysis & Correlation
Testing & Validation
Process Opt. Binning Strategy
Mass Production
Design recipe opt.
>15% PE improvement

Fig. 1-3 DTCO Physical Design Flow

Fig. 1-3 illustrates the DTCO implementation flow, encompassing both pre-silicon preparation and post-silicon analysis and feedback, driven by data and machine learning optimization. Pre-silicon preparation includes process analysis, SPICE simulation, cell library analysis, feature extraction and modeling, critical path performance improvement, and area and power optimization. Post-silicon analysis involves designing and integrating on-chip monitoring circuits, testing, data correlation analysis, machine learning platform application, automation of binning strategies, and optimization feedback.

1.3. Future Directions of DTCO

The introduction of machine learning (ML) and generative AI (GenAI) is unlocking the full potential of DTCO. From data-driven design analysis to generating realistic virtual chip data (Virtual Silicon), DTCO not only tackles current challenges but is also set to drive future innovations in semiconductor design. The following chapters will delve into how the DTCO.ML™ machine learning platform and the DTCO.GenAI™ generative model platform can revolutionize design efficiency and productivity, paving the way for new possibilities in the semiconductor industry.

Chapter 2. Key Challenges and Strategies in Driving DTCO

2.1. Key Challenges in Implementing DTCO

While DTCO holds immense potential, its implementation faces several challenges, as shown in Fig. 2-1:

- **Data-Driven Bottlenecks**: Successful DTCO implementation depends on accurate process and design data, but data collection and integration are often limited by commercial secrets and technical barriers.

- **Cross-Disciplinary Expertise**: DTCO requires close collaboration between design engineers and process experts, both needing a deep understanding of each other’s technical domains, which challenges traditional division-of-labor models.

- **Support for Innovative Methods**: Existing EDA tools and algorithms focus on isolated aspects, lacking comprehensive integration of design and process interactions, which limits the ability to address DTCO's complexity and optimization potential.

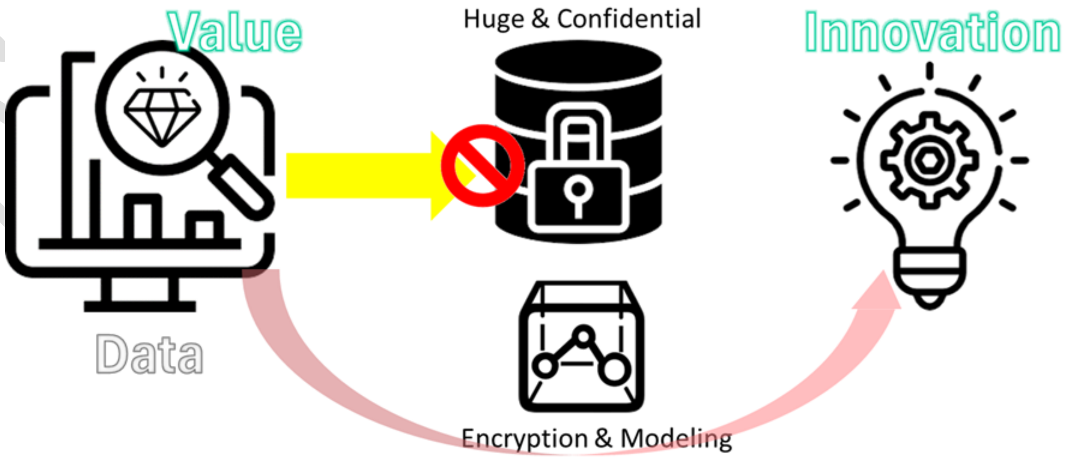


Fig. 2-1 Data Acquisition Barriers

2.2. Demand for Innovative Design Methods

High-reliability chip design and precise monitoring are unlocking new opportunities in markets like smart devices, automotive systems, low Earth orbit (LEO) satellites, co-packaged optics (CPO) and and medical applications. This includes high-precision power management systems for extended battery life and integrating ultra-low energy (ULE) and AI technologies for inference in microcontrollers (MCUs).

- **Reliability and Value Enhancement**: Using machine learning and AI to optimize performance, enhance reliability, and improve chip monitoring. Integrating process and chip optimization accelerates market responsiveness and boosts the value of internal IP and EDA development.

- **Precise Design Margins and Strategies**: Through chip monitoring and data analysis, detailed insights into process and component characteristics enable precise management of design margins and compensation strategies, while fostering innovative design concepts to ensure stable performance.

- **High-Performance Design**: Combining AI and generative models to transform design processes and EDA tools, delivering significant gains in efficiency and productivity, and driving chip design toward greater intelligence and performance.

2.3. Productivity Optimization Platform Development

To address the challenges in DTCO implementation, developing a productivity optimization platform is crucial, enhancing chip competitiveness and productivity through the integration of systems, talent, tools, and methodologies:

- **Data Center Integration**: Establish a centralized system architecture to consolidate, extract, and track design metrics, process parameters, and production data. This resolves data fragmentation and provides efficient data support for design-process interaction optimization.

- **Cross-Disciplinary Expertise Development**: Strengthen the training of data science talent to improve programming and data analysis skills. Foster deep collaboration between design and process experts through visualization tools and correlation analysis, breaking down technical barriers.

- **Tool Innovation and Standardization**: Develop new EDA tools to support comprehensive DTCO optimization, and standardize monitoring IPs and data formats. Optimize design and testing methods to overcome the limitations of current tools in handling design-process interactions.

- **AI and Machine Learning Applications**: Leverage machine learning for feature modeling, regression, and trend forecasting to optimize design-process interaction, including variation analysis, performance grading, and dynamic compensation, driving DTCO efficiency improvements.

Chapter 3. Optimizing Chip Energy Efficiency and Productivity

Fig. 3-1 illustrates the main phases of project execution: design, manufacturing (packaging and testing), and mass production. Each phase provides key support solutions to ensure the optimization of productivity and efficiency throughout the entire process.

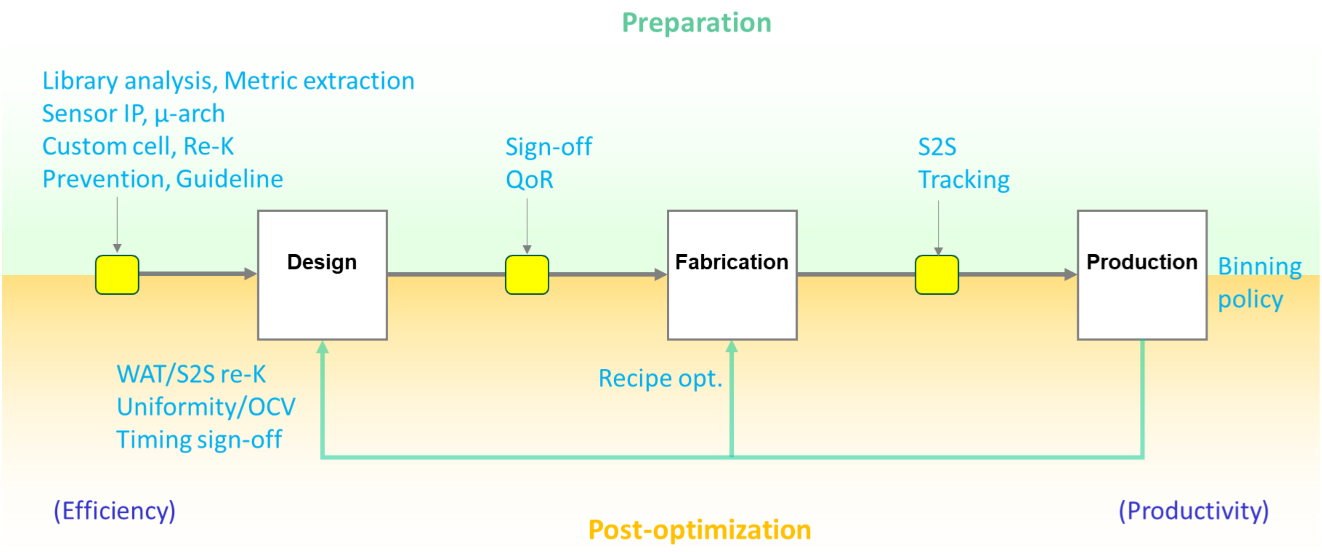


Fig. 3-1 Chip Productivity and Efficiency Optimization Practices

3.1. Preparatory Work Before Project Initiation

- **Design and Process Characterization**: Develop integrated tools to consolidate dispersed component library data (e.g., area, timing, power) into a unified database. Use numerical analysis and feature quantification to efficiently extract the physical parameters of core components.

- **Process Analysis**: Conduct WAT characteristic distribution analysis (e.g., Isat N/P, Vsat N/P), correlate SPICE models with post-silicon data, identify key mean shifts, and establish design margins to ensure process-design compatibility.

- **Cell Characterization**: Perform in-depth analysis of core library units, examining their behavior under different switching speeds and load scenarios. Study the impact of process parameters, channel length, threshold voltage, and temperature on cell properties.

- **Preventive Design Guidance**: Integrate decoupling capacitors early in design, such as incorporating them into clock network components or using stacked Multi-bit DFFs with built-in capacitors to control current density. Apply strategies like test clock separation and scan reordering to mitigate IR-drop risks in later stages.

- **Supporting Measures**: Integrate monitoring IP for post-silicon data analysis and establish a comprehensive measurement strategy, including SPICE simulations, monitor integration, testing methods, data collection, and validation processes to ensure system stability and reliability.

3.2. Custom Cell and Timing Signoff Strategy

- **Optimizing Chip Efficiency**: Analyze voltage, temperature, and design constraints to perform timing corrections, addressing key drift from WAT data and identifying optimal operating points.

- **Design Analysis**: Evaluate the impact of key components on area and power consumption, focusing on critical paths such as multi-bit registers (MBDFF), inverters, and adders across voltage scenarios, minimizing signal distortion at lower voltages.

- **Standard Cell Library Optimization**: Optimize components affecting routing (e.g., XOR/XNR, MUX) and clock cells (CKINV/BUF), balancing drive strength to improve reliability, power efficiency, and frequency while reducing uncertainty.

- **Custom Cell Optimization**: Customize cells based on library and critical path evaluations (e.g., multi-bit registers, pulse-latches), ensuring signal balance across voltages, preventing hold-time issues, and using SPICE Monte Carlo for voltage and margin analysis.

- **Timing Re-K**: Adjust timing Re-K and signoff strategies based on WAT distributions to reduce margins and boost competitiveness, estimating voltage gradients and precise margins through physical parameter distributions and regression analysis.

- **Preventive Physical Design**: Enhance performance, area utilization, power efficiency, and reliability with custom circuits, incorporating features like clock slew balance (CKINV/CKBUF) and dynamic IR-drop prevention with built-in decoupling capacitors.

To reduce the risk of excessive local current density, as shown in Fig. 3-2, components with high toggle rates and high current density are vertically arranged, with decoupling capacitors integrated around them. This design effectively mitigates hotspot issues, as such hotspots, if not addressed during the design phase, are difficult to resolve later, even if detected by tools.

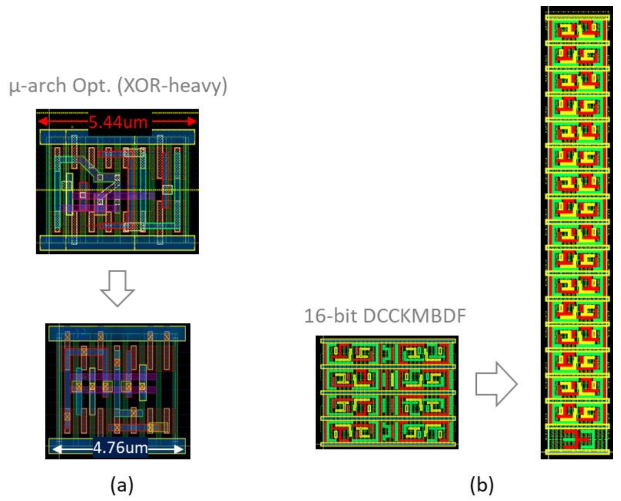


Fig. 3-2 μ-arch Optimization

To address routing issues caused by algorithms, a comprehensive optimization strategy is employed, combining top-down (considering routing and microarchitecture adjustments) and bottom-up (optimizing individual or combined MEGA Cells [1]) approaches. For instance, optimization is performed on the XOR-heavy portion of the SHA3 algorithm and the MUX-heavy section of the switch mechanism to improve layout design efficiency, as shown in Fig. 3-3.

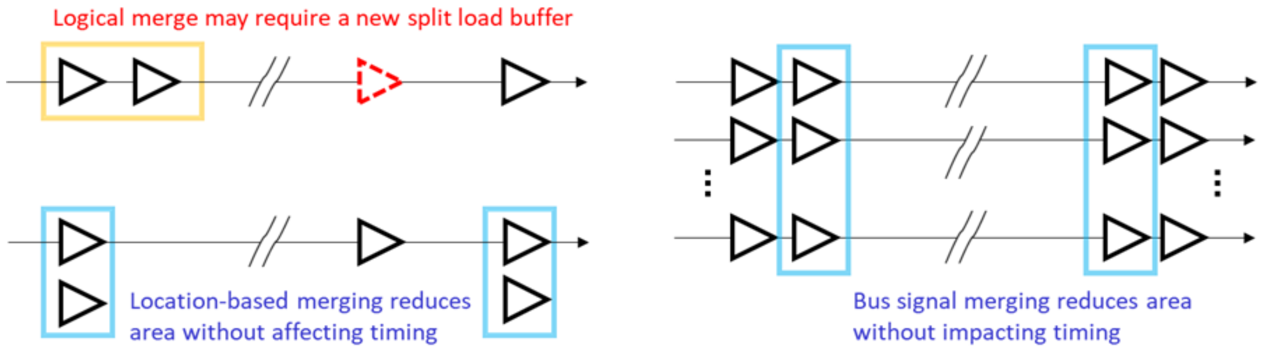

Logical merge may require a new split load buffer
Location-based merging reduces
area without affecting timing
Bus signal merging reduces area
without impacting timing

Fig. 3-3 MEGA Cell Exploration

3.3. Process Optimization and Analysis Techniques

- **WAT and CP/FT Mapping and Correlation**: Analyze WAT parameter distributions (e.g., Isat N/P, Vtl N/P, Ioff N/P) and perform multi-dimensional correlation with CP/FT data, using data visualization to identify defects and guide optimization.

- **S2S Correlation and Library Data Integration**: Correlate process and library data, analyzing SPICE models and RO design measurements to ensure consistency and accuracy.

- **Uniformity/OCV Analysis**: Conduct wafer uniformity regression and in-chip variation (OCV) analysis, refining timing margin distribution to enhance performance and reliability.

- **Process Recipe Optimization and Machine Learning**: Use WAT and CP data correlation and visualization to identify optimal process windows and design recipes, as shown in Fig. 3-4. Apply machine learning and neural networks for feature modeling and regression, driving intelligent production and design optimization.

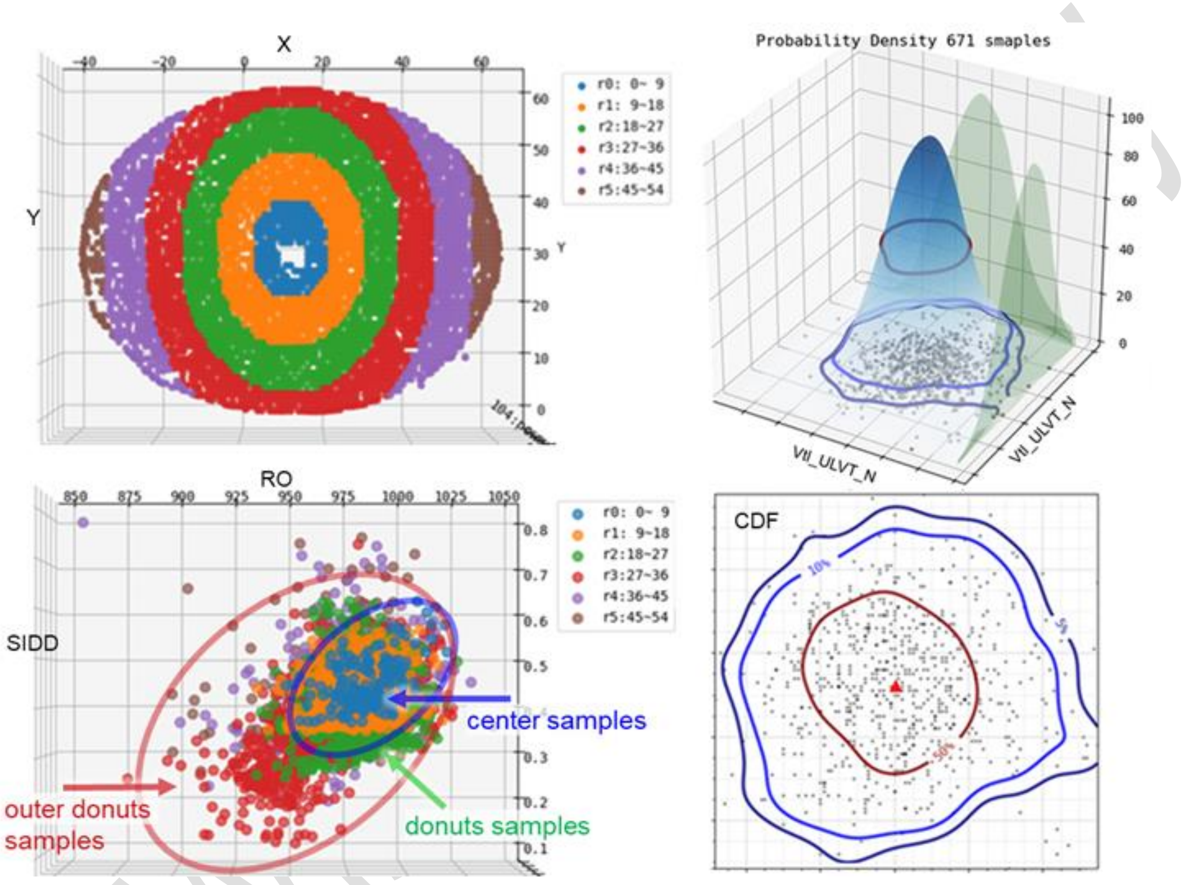


Fig. 3-4 Data Visualization and Feature Correlations

3.4. Compensation Mechanism Design and Implementation

- **Chip Performance Rating and Binning Strategy**: Leverage machine learning or neural networks to automate chip performance rating and binning, optimizing chip, wafer, and batch rankings to enhance yield and production efficiency.

- **Aging and Voltage Compensation**: Design strategies to compensate for chip aging and voltage fluctuations, improving lifespan and reliability.

- **Dynamic Margin Alert and Adjustment**: Implement a dynamic slack alert system to adjust voltage and clock strategies (e.g., clock frequency, gating, stretching) in real-time, optimizing performance and fault prediction based on margin changes.
- **IR/Variation Tolerance**: Develop solutions for managing IR drop and process variation, including dynamic voltage drop management, local gain, transient response, and fault tolerance, to ensure stability and adaptability to PVT variations.

Fig. 3-5 shows the wafer-level RO frequency distribution, where the unevenness of the RO gradient (Local Variation) cannot be eliminated through voltage compensation. This prompts a reassessment of AOCV and LVF adequacy in STA methods and the integration of performance gradients. To address these challenges, industry has proposed solutions such as LDO Array regional compensation and multi-chiplet packaging to resolve process-related issues.

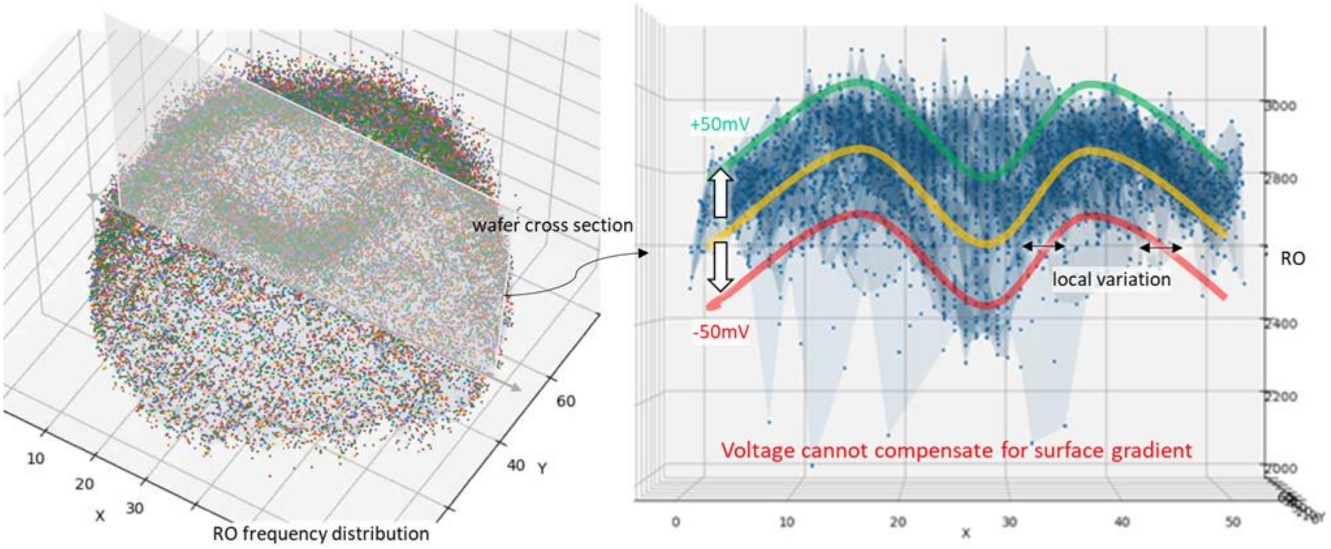


Fig. 3-5 RO Gradient Uniformity

As shown in Fig. 3-6, process control variations can cause significant performance gradients between the northern and southern hemispheres and the inner and outer rings of the wafer. During chip mass production and packaging testing, the critical path often involves a mix of components, leading to poor correlation between performance metrics like Small-delay Fault (DFT) or MBIST and Fmax. Furthermore, process variations can cause tools and algorithms to identify sensitive critical paths from different groups, complicating the issue.

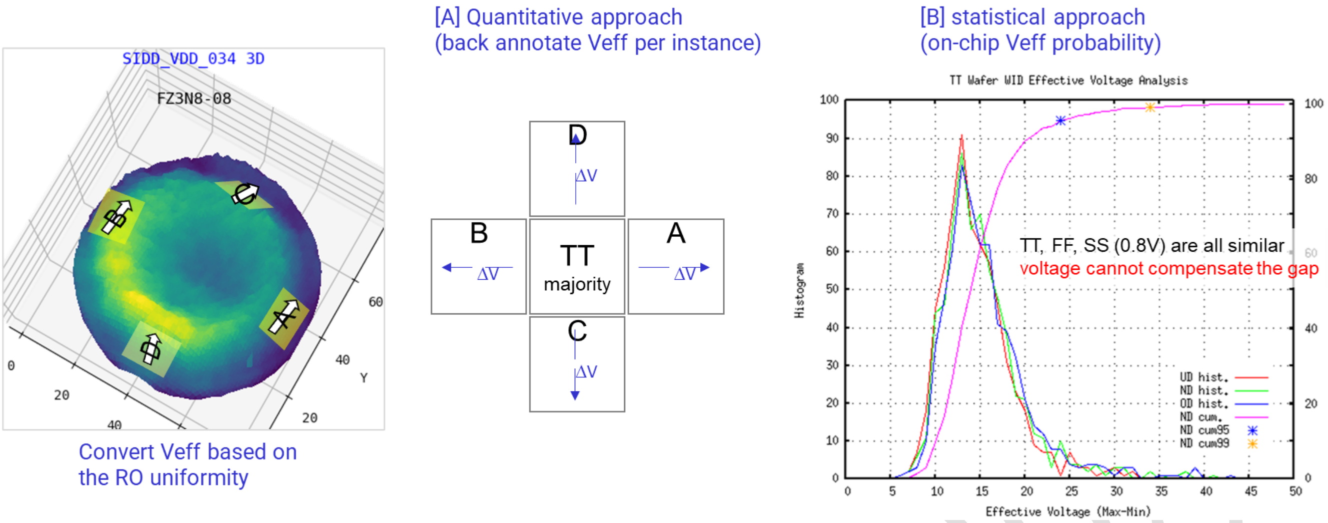


Fig. 3-6 Uniformity and Effective IR-drop

3.5. Challenges and Demands of Near-Threshold Voltage Technology

At near-threshold voltage (<300mV), the 'reverse temperature anomaly' is common, as shown in Fig. 3-7. Traditional design processes rely on SPICE models and Liberty libraries from the foundry to set timing signoff boundaries. However, critical parameters like WAT Isat and Vtl are typically only available after mass production acceptance, revealing insufficient design margins and highlighting the limitations of current design processes in both scientific and practical terms.

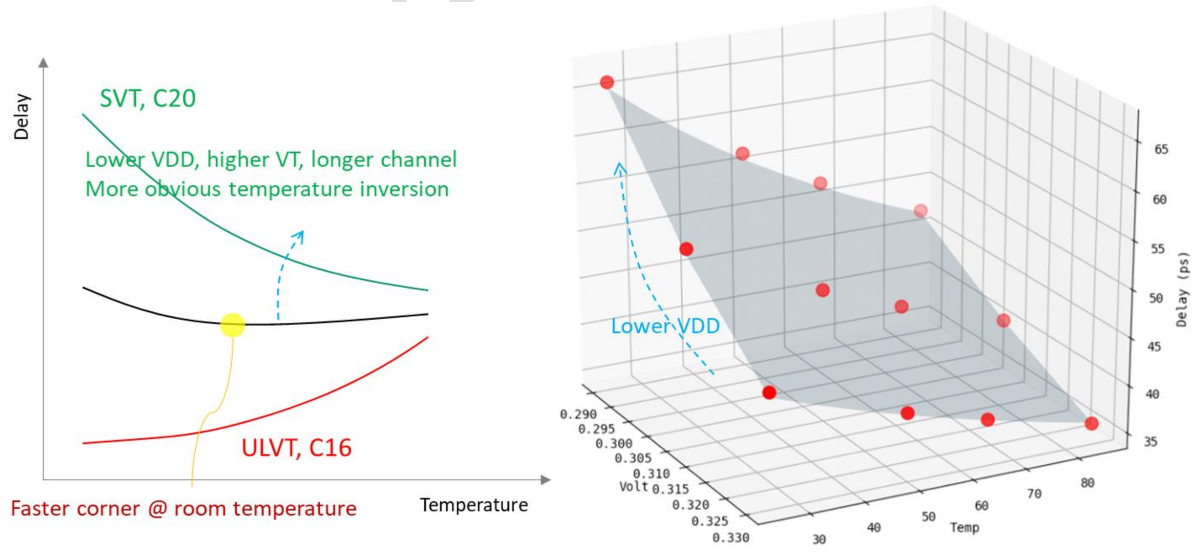
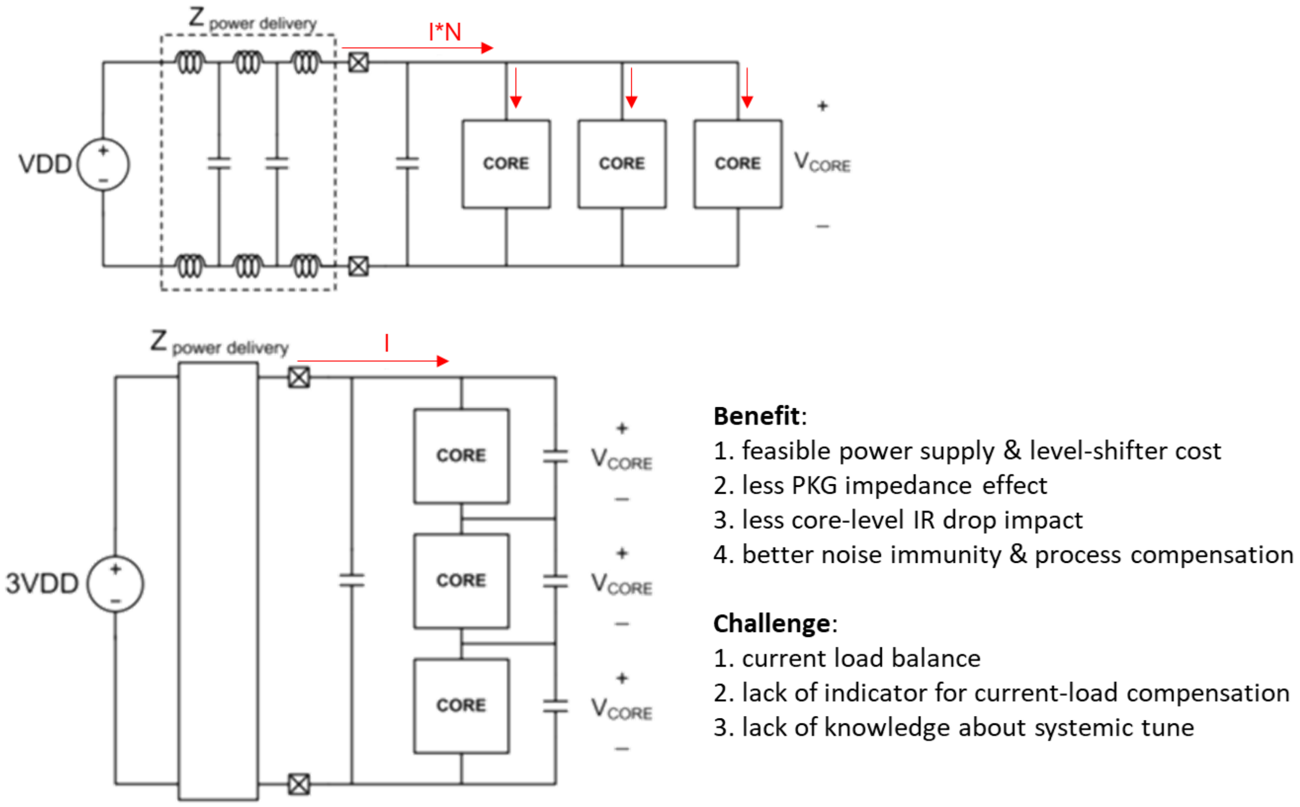


Fig. 3-7 Temperature Inversion

In early multi-core chip designs, Package RLC was crucial for power layout and analysis. The increased total current during parallel processing amplified voltage drop noise (Ldi/dt). To mitigate this, the 'Voltage-Stacking' design was introduced.



Benefit:
1. feasible power supply & level-shifter cost
2. less PKG impedance effect
3. less core-level IR drop impact
4. better noise immunity & process compensation

Challenge:
1. current load balance
2. lack of indicator for current-load compensation
3. lack of knowledge about systemic tune

Fig. 3-8 Voltage Stacking

As shown in Fig. 3-8, stacking chips to reduce Vmin (e.g., below 300mV) helps distribute noise across more chips, creating a self-compensating 'current dynamic balancing' system. However, leakage currents from thermal effects or process variations may lower voltage thresholds for underperforming chips, while currents from inactive chips can negatively affect the system. This challenges traditional low-voltage regulation techniques (e.g., CG/DVFS/AVS), requiring system-wide adjustments rather than individual chip tuning, making it a promising area for further research.

Part II: DTCO.ML™ - Machine Learning-Driven Semiconductor Process Optimization

Chapter 4. The Integration of Machine Learning and DTCO (DTCO.ML™)

Chip design and manufacturing depend on industry collaboration. Traditional methods struggle to fully capture device and system behavior, resulting in overly conservative timing signoff and challenges in managing design margins. With data science, the design process has become more scientific and efficient. As shown in Fig. 4-1, integrating machine learning, IP, EDA tools, and design processes creates a high-efficiency solution that enhances design productivity.

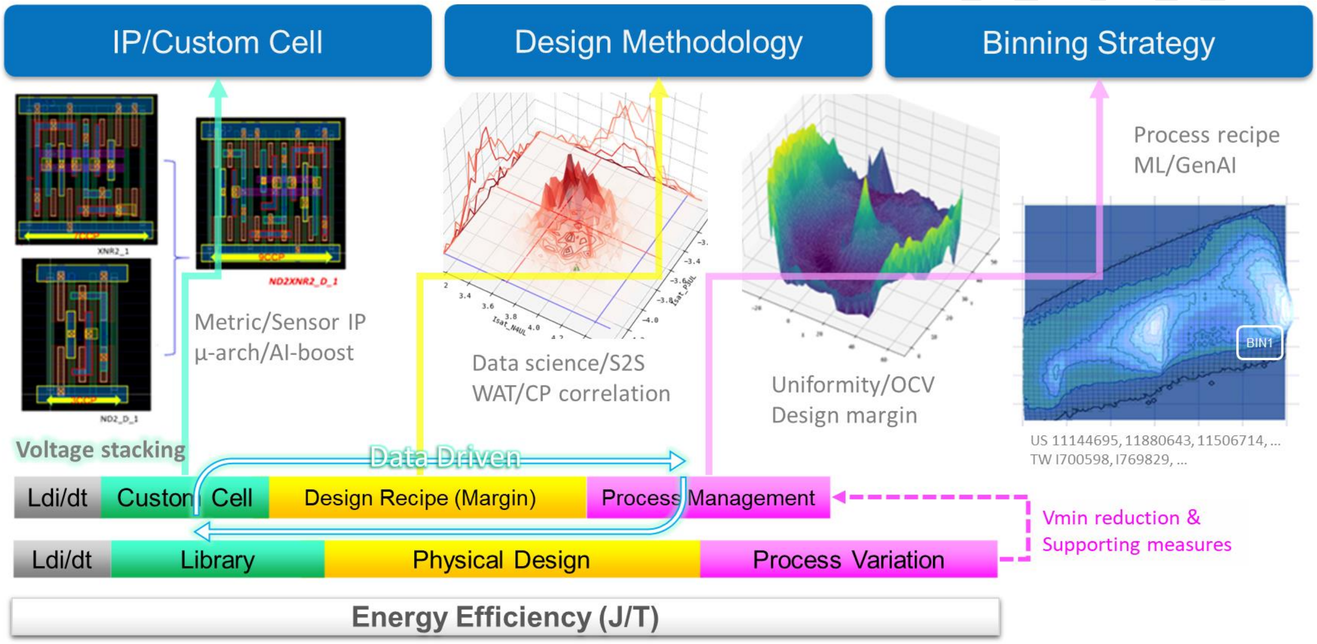


Fig. 4-1 Revolutionizing Efficiency & Productivity through DTCO.ML

- **IP Solutions and Custom Cells**: Enhance energy efficiency, area, and reliability with on-chip monitoring, dynamic voltage/temperature sensing, slack alerts, and computational units. Support diverse clock schemes (asynchronous, self-clocked, PL/PG) and custom cells for clock networks and critical paths, optimizing performance, power, and area. Include dynamic IR prevention, microarchitecture (µ-arch), and MEGA cell optimization to improve routing and multi-logic efficiency.

- **EDA and Design Process Development**: Employ machine learning and AI to develop innovative EDA tools and design processes, overcoming traditional cross-domain and data interaction limitations for more accurate design iterations and optimizations.

- **Binning Compensation Strategy**: Leverage advanced tools for feature extraction and library analysis, capturing high-dimensional trends. Use the DTCO machine learning platform to optimize design and process, implementing binning and compensation strategies to boost chip efficiency and productivity.

4.1. Virtual Wafer Data Modeling (Virtual Silicon)

We manually construct virtual wafer data by observing, transforming, and standardizing the data, combined with stochastic process modeling. First, we create virtual wafer coordinates, retaining only the data within the radius of the wafer center. Example 4-1 demonstrates the generation of virtual wafer coordinates, with a wafer size of 65×65 samples, as shown in Fig. 4-2.

Example 4-1 Virtual Silicon: Gaussian-Volcan Wafer Data

```python
# Virtual Silicon : Gaussian-Volcano Wafer Data
import matplotlib.pyplot as plt
import numpy as np

def genWaferXY(w=65, h=65):
    ix, iy = np.linspace(1, w, w), np.linspace(1, h, h)
    gx, gy = np.meshgrid(ix, iy)
    cx, cy = np.ceil(w/2).astype(int), np.ceil(h/2).astype(int)
    cr = min(cx, cy)
    r = np.sqrt((gx-cx)**2 + (gy-cy)**2)
    mask = (r <= cr)
    x, y, r = gx[mask], gy[mask], r[mask]
    if False:
        plt.figure(figsize=(6,6))
        plt.scatter(x, y, s=15, alpha=0.4, marker='s')
        plt.scatter(cx, cy, s=15, c='r', marker='s')
        plt.xlabel('X')
        plt.ylabel('Y')
        plt.tight_layout()
    return x, y, r

def volcanoSin(r):
    θ = np.interp(r, [1,65], [-np.pi, np.pi])
    z = np.sin(θ - np.pi/2) - 1.5 * np.sin(θ/2 - np.pi/2)
    return np.interp(z, [z.min(), z.max()], [0, 1])

def volcanoGaussian(r, h1=1, s1=15, h2=1, s2=5):
    g1 = h1 * np.exp(-r**2/(2*s1**2))
    g2 = h2 * np.exp(-r**2/(2*s2**2))
    z = g1-0.5*g2
    return np.interp(z, [z.min(), z.max()], [0, 1])

#%% gendataset
x, y, r = genWaferXY()
z1 = volcanoSin(r)
z2 = volcanoGaussian(r)
```

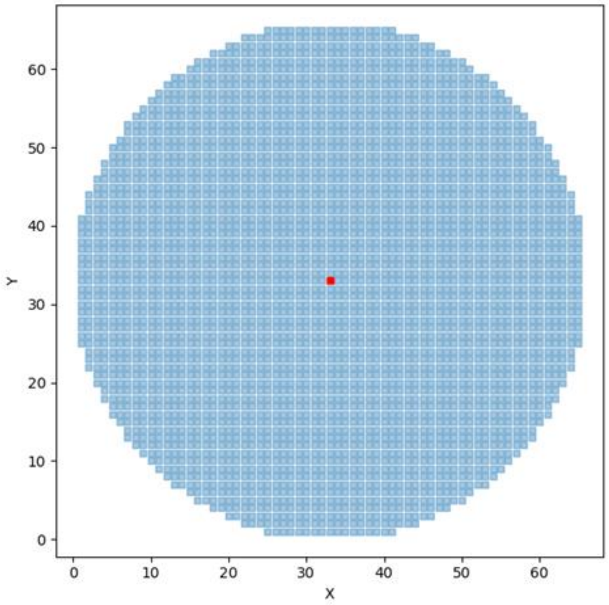


Fig. 4-2 Virtual Wafer Data Mapping

Different fabs exhibit distinct uniformity patterns. For example, some processes show systematic defects, such as volcano-shaped structures in RO or SIDD data along the XY coordinates. We simulate this using Sin or Gaussian functions with varying amplitudes. By applying these functions to the wafer coordinates, we generate volcano-like structures ($z0$), with probability density distributions typically U-shaped, resembling a hyperbolic cosine (Cosh) distribution. Gaussian functions produce smoother distributions compared to Sin, as shown in Fig. 4-3.

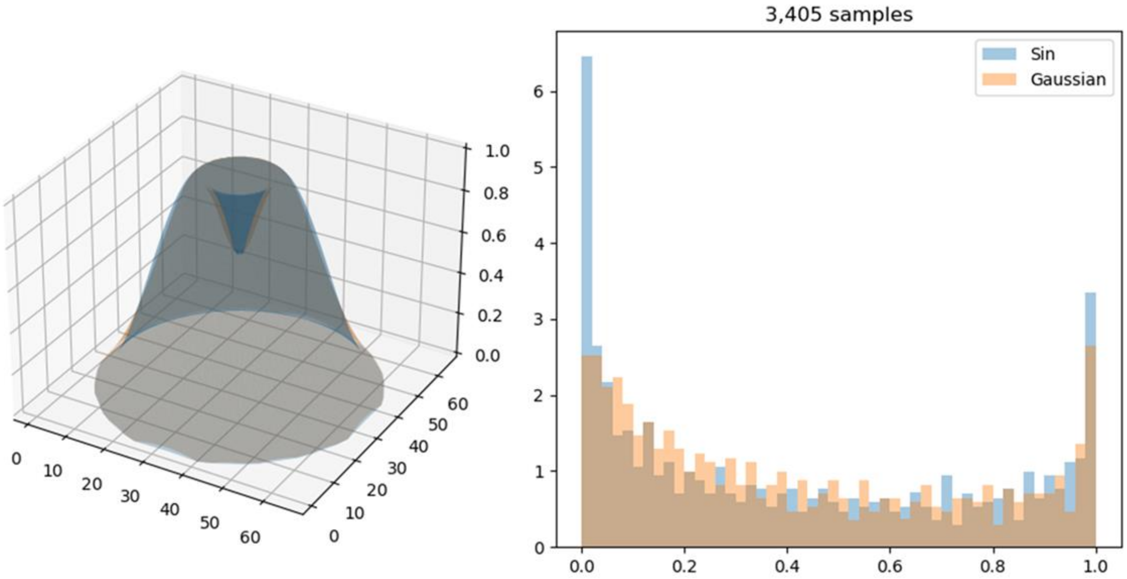


Fig. 4-3 Volcano Surface Distribution and Probability Density

Example 4-2 generates the first set of virtual wafer data $z1$ (e.g., RO) based on the Gaussian-Volcano surface. First, the smooth surface $z0$ is transformed into a standard normal distribution ($\epsilon 0 \sim N(0,1)$), then Gaussian noise ($\epsilon 1 \sim N(0,1)$) is added before scaling back to the original $z0$ range. For visualization and comparison, we use np.interp(x, [*original range*], [*target range*]) to normalize $z1$ to [0,1]. After adding Gaussian noise, $z1$ follows a skewed Gaussian distribution, with mean and variance adjustable to match real-world observations (e.g., RO), as shown in Fig. 4-4.

Example 4-2 RO Silicon Data Modeling and Standardization

```
#%% generate an RO surface with Gaussian noise
x, y, r = genWaferXY()
z0 = volcanoGaussian(r)
ϵ0 = 1/z0.std()*(z0-z0.mean()) # z0 to ~N(0,1)

# combine with noise
ϵ1 = np.random.normal(0, 1, len(z0)) # noise ~N(0,1)
z1 = (ϵ0 + 0.2*ϵ1) # ~N(0,1)
z1 = z0.std()*z1 + z0.mean() # back to z0 scale
z1 = np.interp(z1, [z1.min(), z1.max()], [0, 1]) # to [0, 1] for comparison

#%% conditional Gaussian standarization
# generate SIDD with a correlation of 0.9 with z1
ρ = 0.9
ϵ1 = 1/z1.std()*(z1-z1.mean()) # z1 to ~N(0,1)
ϵ2 = np.random.normal(0, 1, len(z1)) # noise ~N(0,1)
z2 = ρ*ϵ1 + np.sqrt(1-ρ**2)*ϵ2 # ~N(0,1)
```

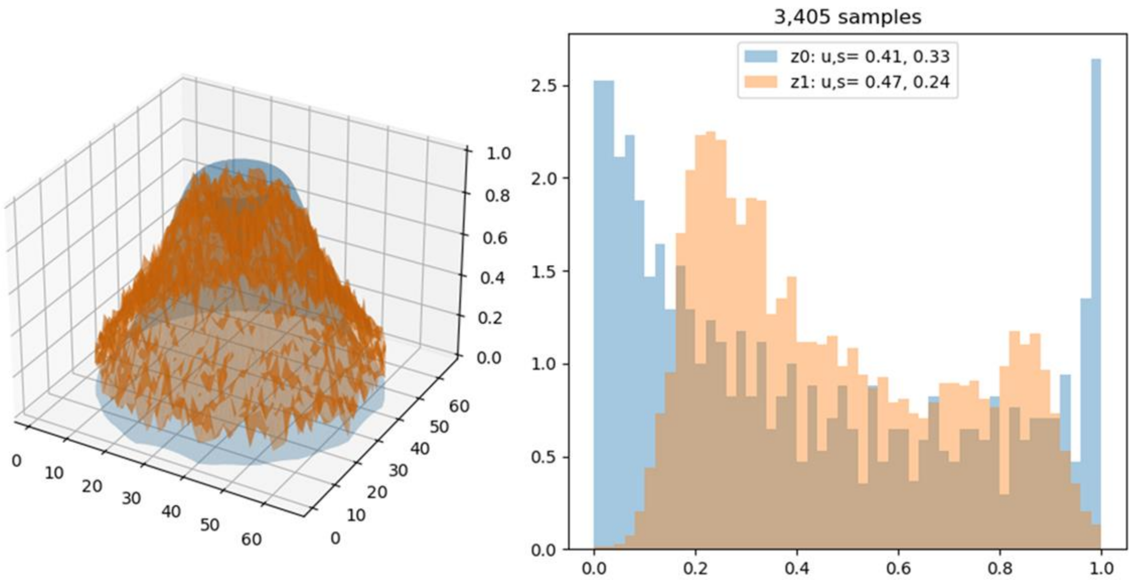


Fig. 4-4 Volcano Surface with Gaussian Noise

Example 4-3 generates the second set of data ($z2$, e.g., SIDD) using a Conditional Gaussian distribution. Observations indicate a strong positive correlation (ρ=0.9) between RO and SIDD, so we aim to maintain the same correlation between $z2$ and $z1$. First, $z1$ is standardized to a normal distribution ($\epsilon1 \sim N$(0,1)), then the second dataset is generated using the formula $\rho \cdot \epsilon1$+np.sqrt(1−ρ**2)$\cdot \epsilon2$, where $\epsilon2$ is is random noise following N(0,1).

Example 4-3 SIDD Modeling and Standardization

```
def logNormScale(u, s): # normal to log-normal
    log_u = np.log(u**2 / np.sqrt(s**2 + u**2))
    log_s = np.sqrt(np.log(1 + s**2 / u**2))
    return log_u, log_s

mu, sigma = 0.5, 0.1 # target mu and sigma of SIDD
log_u, log_s = logNormScale(mu, sigma) # retarget to log-normal scale
z2 = np.exp(log_s*z2 + log_u) # to z2 scale
z2 = np.interp(z2, [z2.min(), z2.max()], [0, 1]) # to [0, 1] scale

plt.figure(figsize=(10,9))
ax1 = plt.subplot(221, projection='3d', title='z1')
ax1.plot_trisurf(x, y, z0, alpha=0.3)
ax1.plot_trisurf(x, y, z1, alpha=0.4)
ax2 = plt.subplot(222, projection='3d', title='z2')
ax2.plot_trisurf(x, y, z0, alpha=0.3)
ax2.plot_trisurf(x, y, z2, alpha=0.4)
ax3 = plt.subplot(223, title=f'{len(z1):,} samples')
ax3.hist(z0, bins=50, density=True, alpha=0.4, label=f'z0: u,s= {z0.mean():.2f}, {z0.std():.2f}')
ax3.hist(z1, bins=50, density=True, alpha=0.4, label=f'z1: u,s= {z1.mean():.2f}, {z1.std():.2f}')
ax3.hist(z2, bins=50, density=True, alpha=0.4, label=f'z2: u,s= {z2.mean():.2f}, {z2.std():.2f}')
ax3.legend()
ax4 = plt.subplot(224, title=f'{len(z1):,} samples')
ax4.scatter(z1, z2, alpha=0.4)
ax4.set_xlabel('z1: RO')
ax4.set_ylabel('z2: SIDD')
plt.tight_layout()
```

While RO and SIDD are highly correlated, SIDD follows a log-normal distribution. To reflect this, we adjust the mean and standard deviation accordingly and apply an exponential transformation. This method handles missing data in one dimension, such as sparse WAT data alongside complete x, y, and RO data. By leveraging reliable boundaries, statistical properties, and high-dimensional correlations, we generate accurate predictions.

As shown in Fig. 4-5, we obtain two-dimensional virtual wafer data: $z1$ (RO) and $z2$ (SIDD), maintaining a correlation of ρ=0.9. Their feature vectors preserve the overall structural trends in the wafer coordinate system. RO follows a skew-normal distribution, while SIDD exhibits a log-normal distribution.

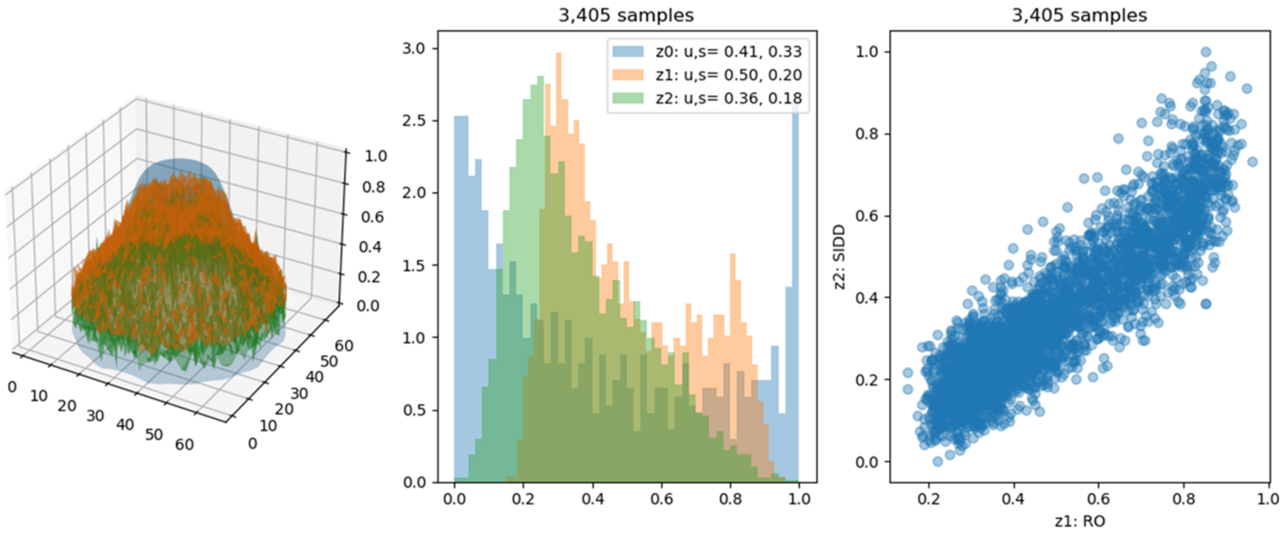


Fig. 4-5 Log-Normal Conditional Gaussian with Proper Covariance

Real silicon data often exhibits complex nonlinear relationships and large-scale uniformity structures beyond simple skew-normal or log-normal distributions. The next chapters explore how generative AI enhances precise modeling and analysis.

4.2. Building and Inferring Regression Models

Estimating timing margins and power consumption in physical design is time-consuming. For leakage current evaluation, traditional methods require LIB CAD engineers to rebuild libraries for numerous PVT corners (Re-K) and use EDA tools for analysis. However, for undefined corners, reliability decreases, and conclusions may be impossible.

To address this, we perform SPICE simulations and characterization at key voltage and frequency points, as shown in Fig. 4-6 (a). The sparse discrete data points are then fitted with a quadratic regression surface (orange curve), modeling leakage as leak=$f(V,T)$. This enables machine learning to reliably predict values and trends beyond the library-defined conditions, such as 0.32V at 70°C. ◦

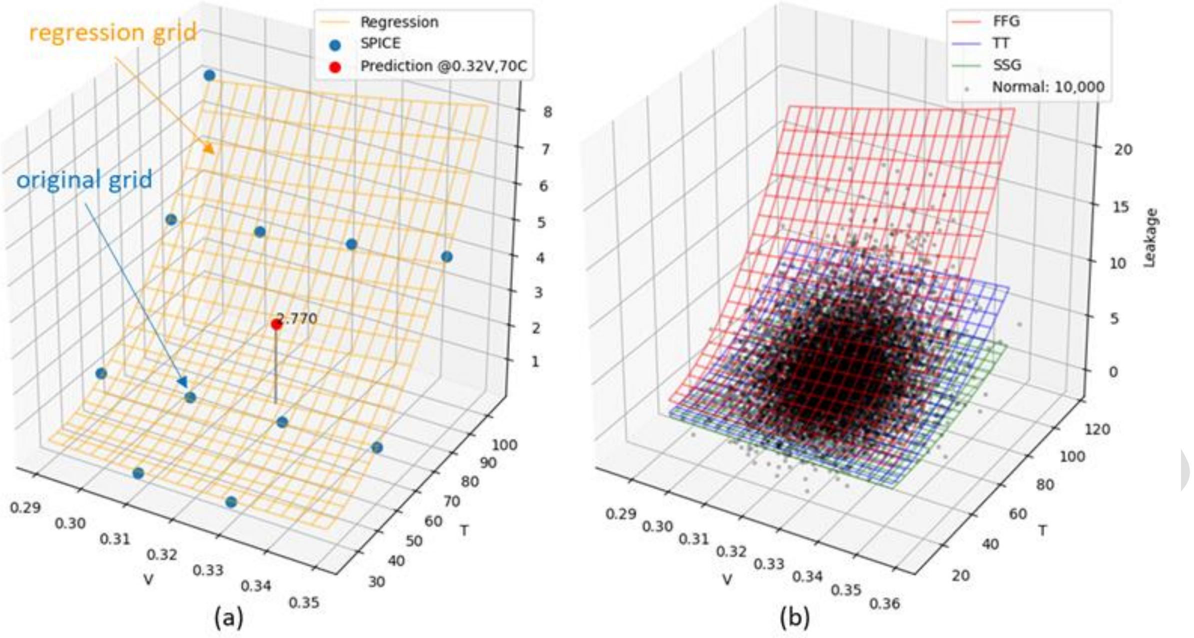


Fig. 4-6 Precise Leakage Prediction

Example 4-4 simulates the SPICE data for CKINV under the TT process, covering 11 discrete voltage and temperature conditions. Using Least Squares Regression (LSR), we can accurately predict values not included in the original PVT grid, such as leakage at 0.32V and 70°C.

Example 4-4 Least Squares Regression

```
# Leakage model regression and prediction.
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import scipy.linalg

# CLKINV1_C10UL @TT
tt = pd.DataFrame(
   [[ 0.29,  50, 1.1656 ], [ 0.29,  85, 3.8962 ], [ 0.29, 105, 7.0997 ],
    [ 0.31,  25, 0.4655 ], [ 0.31,  50, 1.2776 ], [ 0.31,  85, 4.2603 ],
    [ 0.33,  25, 0.5086 ], [ 0.33,  50, 1.3934 ], [ 0.33,  85, 4.6352 ],
    [ 0.35,  50, 1.5129 ], [ 0.35,  85, 5.0208 ]], columns=['V','T','leak'])

#%% LS Regression
x, y, z = tt.values.T
X = np.array([np.ones_like(x), x, x**2, y, y**2, x*y]).T # training set
C,_,_,_ = scipy.linalg.lstsq(X, z) # LS regression coefficient

# regression grid
ix, iy = np.linspace(x.min(), x.max(), 20), np.linspace(y.min(), y.max(), 20)
gx, gy = np.meshgrid(ix, iy)
tx, ty = map(np.ravel, (gx, gy))
T = np.array([np.ones_like(tx), tx, tx**2, ty, ty**2, tx*ty]).T
tz = np.dot(T, C)
```

```
# predict condition 0.32V, 70°C not present in the SPICE metric
v, t = 0.32, 70
p = np.dot(np.array([1, v, v**2, t, t**2, v*t]), C)

plt.figure(figsize=(6,6))
ax = plt.subplot(projection='3d')
ax.plot_wireframe(gx, gy, tz.reshape(gx.shape), alpha=0.4, colors='orange', label='Regression')
ax.scatter(x, y, z, s=50, alpha=1, label='SPICE')
ax.scatter(v, t, p, s=50, c='r', alpha=1, label=f'Prediction @{v}V,{t}C')
ax.plot([v,v], [t,t], [0,p], c='gray')
ax.text(v, t, p, f'{p:.3f}')
ax.set_xlabel('V')
ax.set_ylabel('T')
ax.set_zlabel('Leakage')
ax.set_box_aspect((1,1,1), zoom=1.0)
plt.legend()
plt.tight_layout(rect=(-0.1,0.05,1,1))
```

Additionally, we can generate a dense grid of data based on the process corner models and elevate it to a 3D model, constructing leak=$f(V,T,P)$. Using a Gaussian distribution probability cloud, we can quickly identify trends for faster process corners (such as FF +1σ or +1.5σ), enabling comprehensive analysis and optimization of design parameters, as shown in Fig. 4-6 (b).

Continuing with the previous example, Example 4-5 constructs LS surfaces from discrete points of FFG and SSG and generates a dense grid for the training set. Assuming the range from FFG to SSG covers 6σ, the model's input parameters are expanded to three dimensions, $f(V,T,P)$, to construct the leakage model. Then, based on a Gaussian normal distribution, 10K random samples are generated around the mean (0.32V, 65°C, TT) of (V,T,P), predicting the complete leakage distribution under this model.

Example 4-5 Leakage Modeling and Prediction

```
# model fitting: leak = f(V,T,P)
ss = pd.DataFrame(
    [[ 0.31,  25, 0.1249 ], [ 0.31,  50, 0.3709 ], [ 0.31,  85, 1.3588 ],
     [ 0.31, 105, 2.5932 ], [ 0.33,  25, 0.1361 ], [ 0.33,  50, 0.4033 ],
     [ 0.33,  85, 1.4742 ], [ 0.35,  50, 0.4365 ], [ 0.35,  85, 1.5924 ],
     [ 0.35, 105, 3.0324 ]], columns=['V','T','leak'])

ff = pd.DataFrame(
    [[ 0.29,  50,  3.889 ], [ 0.29,  85, 11.802 ],
     [ 0.31,  25,  1.687 ], [ 0.31,  50,  4.276 ], [ 0.31,  85, 12.939 ],
     [ 0.33,  25,  1.849 ], [ 0.33,  85, 14.115 ]], columns=['V','T','leak'])

# model fitting for each process corner
gnum = 20
px,py = np.linspace(0.29,0.35,gnum),np.linspace(25,105,gnum)
gx,gy = np.meshgrid(px,py)
tx,ty = gx.ravel(),gy.ravel()
tzL = []
T = np.array([np.ones_like(tx), tx, tx**2, ty, ty**2, tx*ty]).T
```

```
for d in [tt, ff, ss]:
    x,y,z = d.values.T
    X = np.array([np.ones_like(x), x, x**2, y, y**2, x*y]).T
    C,_,_,_ = scipy.linalg.lstsq(X,z) # LS regression
    tzL += [np.dot(T,C)]

ptt,pff,pss = tzL

# model fitting for all corners
size = gnum**2
a,b,c = np.array(list(tx)*3),np.array(list(ty)*3),np.array([3]*size+[0]*size+[-3]*size)
z = np.array(list(pff)+list(ptt)+list(pss))
X = np.array([np.ones_like(z), a, a**2, b, b**2, c, c**2, a*b, a*c, b*c]).T
C,_,_,_ = scipy.linalg.lstsq(X,z) # LS regression, as f(V,T,P)

# Generate 10K random samples from a Gaussian normal distribution
size = 10000
V = np.random.normal(0.32, (0.35-0.29)/6, size)
T = np.random.normal(65, (105-25)/6, size)
P = np.random.normal(0, 1, size)
a,b,c = V,T,P

X = np.array([np.ones_like(a), a, a**2, b, b**2, c, c**2, a*b, a*c, b*c]).T
p = np.dot(X,C) # batch prediction

# predict condition 0.32V, 70°C, TT+1.5σ not present in the SPICE metric
#pa,pb,pc = 0.32, 70, 1.5
#pp = np.dot(np.array([1, pa, pa**2, pb, pb**2, pc, pc**2, pa*pb, pa*pc, pb*pc]).T, C)

plt.figure(figsize=(6,6))
ax = plt.subplot(projection='3d')
ax.plot_wireframe(gx, gy, pff.reshape(gx.shape), colors='r', alpha=0.4, label='FFG')
ax.plot_wireframe(gx, gy, ptt.reshape(gx.shape), colors='b', alpha=0.4, label='TT')
ax.plot_wireframe(gx, gy, pss.reshape(gx.shape), colors='g', alpha=0.4, label='SSG')
ax.scatter(a, b, p, s=5, alpha=0.2, c='k', label=f'Normal: {size:,}')
ax.set_xlabel('V')
ax.set_ylabel('T')
ax.set_zlabel('Leakage')
ax.set_box_aspect((1,1,1), zoom=1.1)
plt.legend()
plt.tight_layout(rect=(-0.1,0.05,1,1))
```

4.3. Application of Data Tracking and Production Optimization

As shown in Fig. 4-7, feature regression analysis predicts chip quality, enabling the classification of chips, wafers, and batches. By leveraging cross-dimensional modeling and extensive data analysis, precise classification strategies and yield assessments are developed, optimizing design recipes to enhance production efficiency and process reliability. This data-driven approach overcomes traditional design limitations, making chip design more scientific and addressing challenges from overly conservative timing signoff margins.

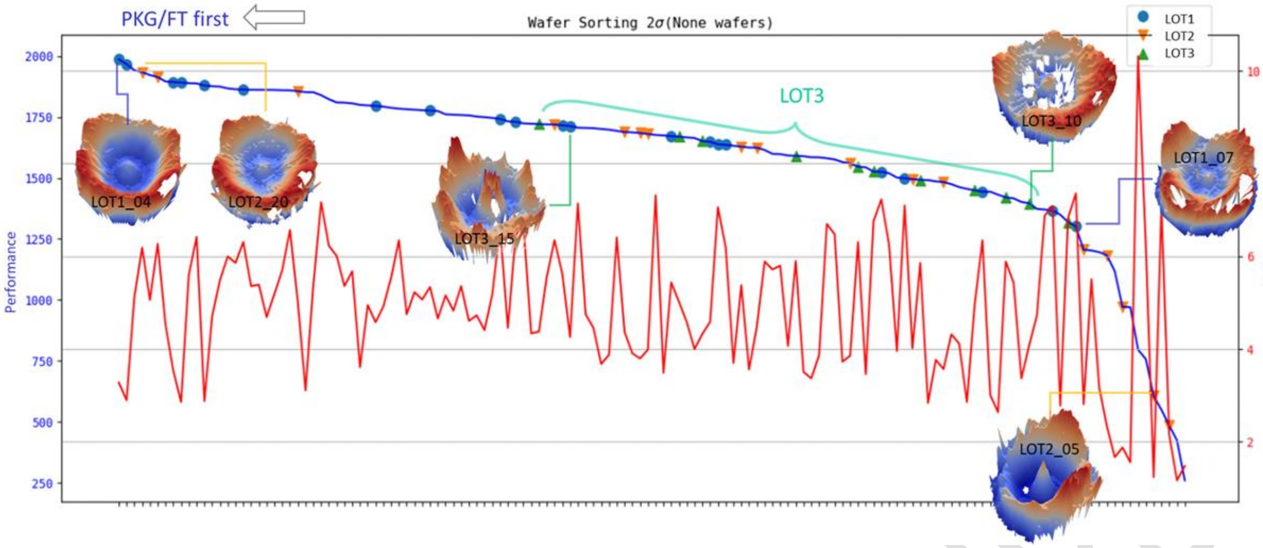


Fig. 4-7 Product Shipment Optimization

By integrating hardware detection circuits with machine learning and regression models, optimal recommendations for design margins and timing signoff are derived. Using chip production test data (WAT, CP, FT, SLT), wafer fab terminology (e.g., WAT Isat, Vtl) and chip design terminology (e.g., CP Performance, Leakage) are mapped to high-dimensional features (e.g., yield or energy efficiency), ensuring alignment between design and manufacturing, as shown in Fig. 4-8.

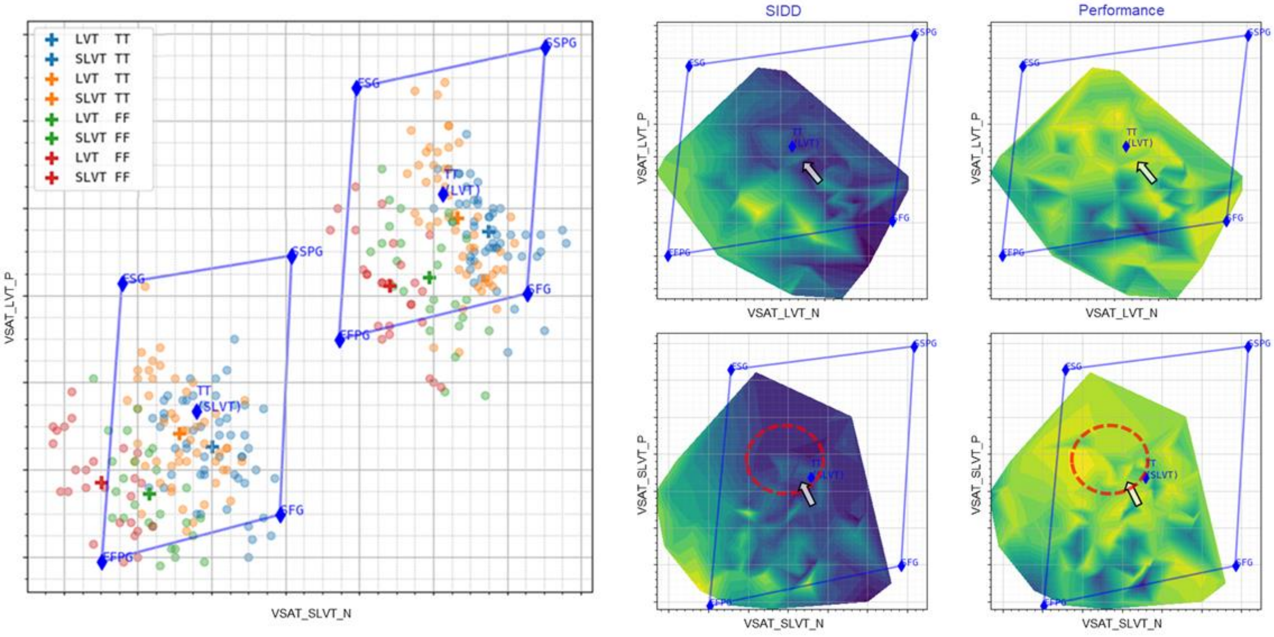


Fig. 4-8 Process Window Aligned with SPICE Target

Chapter 5. Library Metric Extraction and Analysis System (libMetric™)

Library cell benchmarking effectively quantifies performance by evaluating key metrics such as drive strength, delay, and power consumption, providing a basis for design optimization. However, a great design isn't solely reliant on high-driving components, just as a high-powered sports car isn't suited for rugged, winding mountain roads, as shown in Fig. 5-1. The design process requires balancing factors such as drive strength (cell driving), threshold voltage (Vth), architecture, and layout.

Library feature extraction (Metric Extraction) and trend analysis are crucial for identifying mismatches between cell parameters and design tools or algorithms. This process is essential for efficient cell library development. Performance gaps can arise between cells with varying Vth or channel lengths. When traditional EDA tools can't meet timing requirements, compromises like increasing area (up-sizing) or reducing power efficiency (adjusting Vth or channel length) may be necessary, potentially impacting overall design performance.

Fig. 5-1 The Art of Design Performance: Navigating Mountain Roads Like a Sports Car

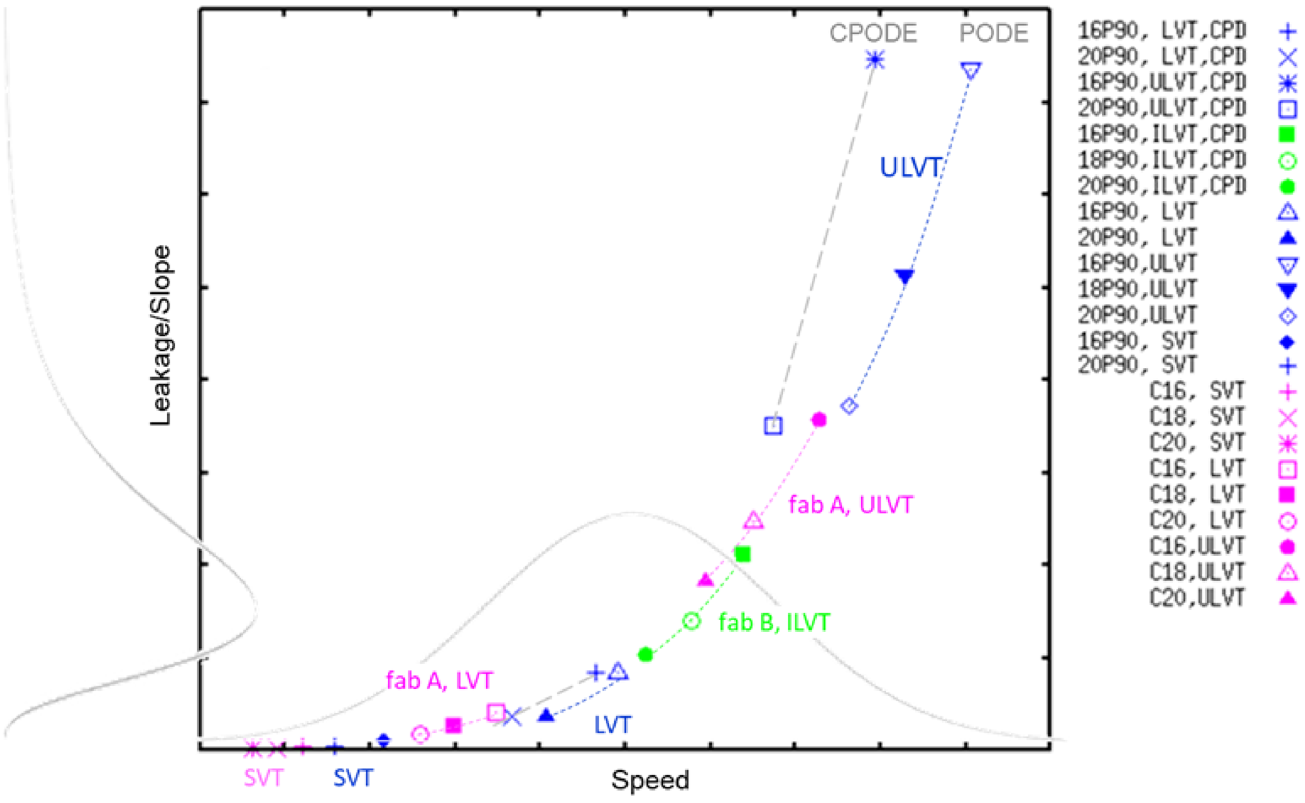


Fig. 5-2 Cell Speed and Leakage Evafuation

As shown in Fig. 5-2, data extraction highlights the gaps between standard cell libraries and EDA tool optimizations. Targeted improvements can boost chip competitiveness, such as balancing drive strength in critical path components, optimizing clock tree components for voltage regulation (Dynamic IR prevention), reducing area and power in layout design, and solving APR routing issues through pin placement optimization. Additionally, cell integration and customization, along with re-architecting and redesigning, can enhance performance.

In practice, optimization strategies are based on critical path and bottleneck analysis. By combining system tasks with market demands, statistical methods help redevelop key components. Prioritizing and optimizing performance bottlenecks leads to the best balance of performance, power, and area, resulting in significant design improvements.

5.1. Cell Timing and Power Modeling

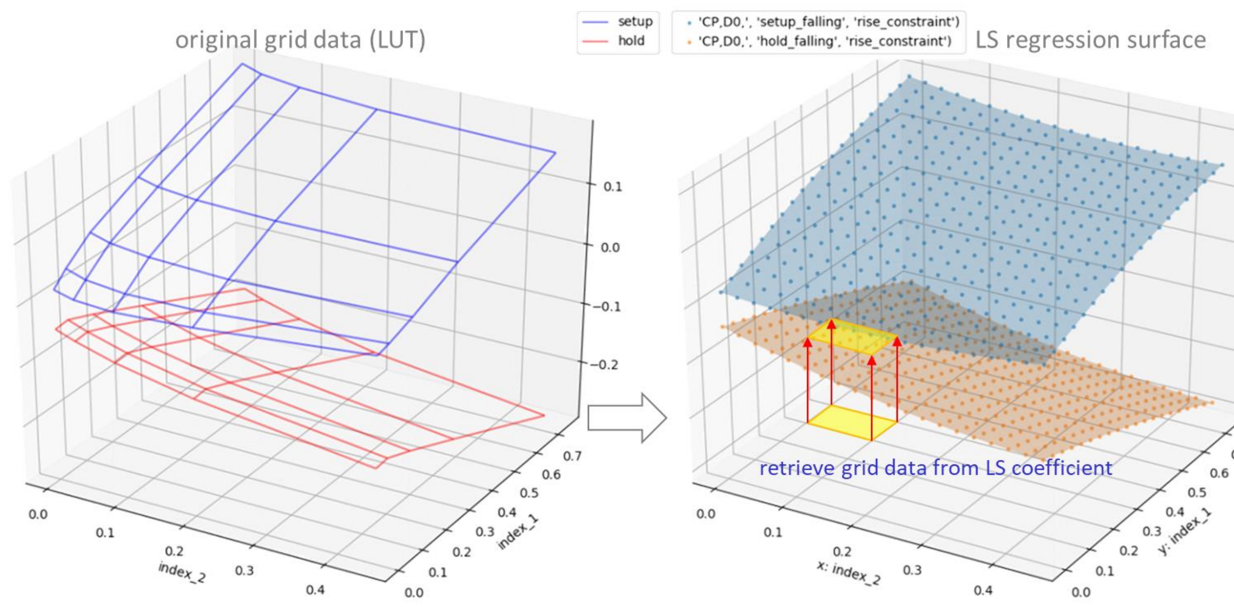


Fig. 5-3 LS Regression

As shown in Fig. 5 3, the current timing and power data for the cell library are stored in discrete grid (lookup table) format. This method consumes considerable memory and presents challenges, such as difficulty in efficiently assessing cell performance sensitivity to PVT variations, accurately comparing cells, and potential misunderstandings. For instance, different fabs lack standardized units, resolutions, and indexing methods, with index scales often increasing in powers of 2 to compress data. Misaligned index ranges can lead to incorrect conclusions. Furthermore, engineers must spend significant time performing interpolation across different cells, with varying index ranges for different PVT corners, adding to the workload.

Through regression analysis, discrete grid data can be fitted into a quadratic equation with six coefficients (bias, x, x^2, y, y^2, $x*y$), enabling unified indexing for comparison and efficient surface reconstruction via inner product computation. This approach also extends to broader ranges, facilitating trend observation and deeper data analysis.

For example, for INVD1, its delay (Delay) is determined by the regression coefficients Sr (cell rise) and Sf (cell fall), which are used to calculate the original delay times zr and zf. The calculation formula is as follows:

$Delay = C0 + C1 * x + C2 * x^2 + C3 * y + C4 * y^2 + C5 * x * y,$

Here, x represents load (in units of pF), and y represents transition time (in units of ns).

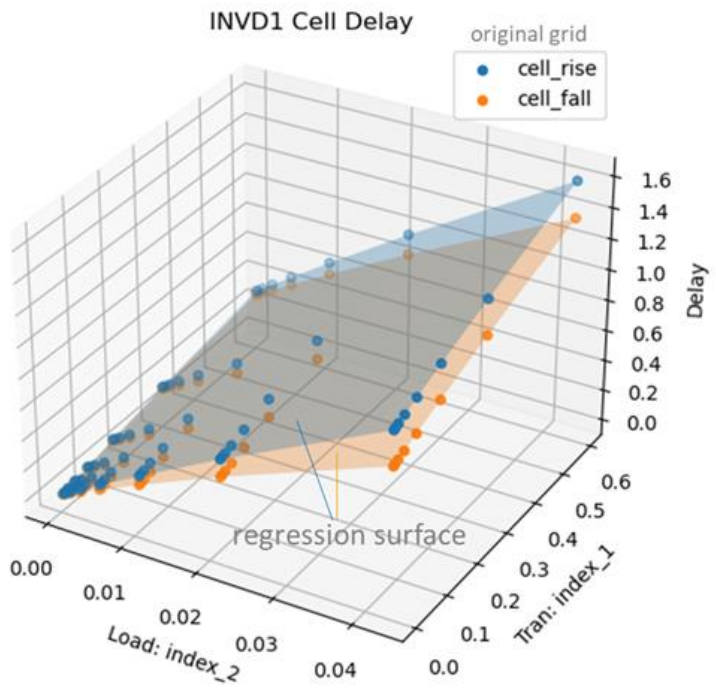


Fig. 5-4 INVD1 Cell Delay

Code Example 5-1 uses Least Squares Regression (LS Regression) to fit the delay time (cell rise and cell fall) of the digital circuit unit INVD1 and visualizes the results, as shown in Fig. 5-4.

Example 5-1 Cell Timing Modeling

```
# INVD1 Cell Timing Modeling.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.linalg

x = np.array([0.0002, 0.0005, 0.0012 , 0.0026, 0.0053, 0.0107, 0.0215, 0.0432]) # index_2: load (pF)
y = np.array([0.0032, 0.0079, 0.0173, 0.036 , 0.0735, 0.1485, 0.2984, 0.5983]) # index_1: tran (ns)
Sr = np.array([0.01, 27.422, -57.761, 0.603, -0.211, 8.85]) # cell rise regression coefficient
Sf = np.array([0.007, 22.323, -56.998, 0.577, -0.223, 9.034]) # cell fall regression coefficient

gx,gy = np.meshgrid(x,y) # mesh grid
tx,ty = map(np.ravel, (gx,gy)) # flattened mesh grid
T = np.array([np.ones_like(tx), tx, tx**2, ty, ty**2, tx*ty]).T # regression grid

zr = np.dot(T, Sr) # original cell rise (ns)
zf = np.dot(T, Sf) # original cell fall (ns)

# LS regression
Cr,_,_,_ = scipy.linalg.lstsq(T, zr) # LS regression coefficient
Cf,_,_,_ = scipy.linalg.lstsq(T, zf) # LS regression coefficient
```

```
# visualization
plt.figure(figsize=(7,6))
ax = plt.subplot(111, projection='3d')
ax.scatter(gx,gy,zr.reshape(gx.shape), label='cell_rise')
ax.scatter(gx,gy,zf.reshape(gx.shape), label='cell_fall')
ax.plot_surface(gx,gy,np.dot(T, Cr).reshape(gx.shape), alpha=0.3)
ax.plot_surface(gx,gy,np.dot(T, Cf).reshape(gx.shape), alpha=0.3)
ax.set_xlabel('Load: index_2')
ax.set_ylabel('Tran: index_1')
ax.set_zlabel('Delay')
plt.legend()
plt.tight_layout()
```

5.2. Cell Feature Extraction

To perform batch cell feature extraction, we need to ensure that each cell operates and is evaluated under the same environmental conditions. As shown in Fig. 5-5 (a), we define the NAND gate of D1 as the basic unit and use the chip's primary operational conditions (e.g., TT, 0.75V, 25°C) as the sampling conditions. The load is defined as the input capacitance driving four identical basic units, and this structure is referred to as BU. As shown in Fig. 5-5 (b), after connecting six stages of BU, the stable waveform (converged transition) will serve as the input for each cell to be featured for extraction (DUE), as shown in Fig. 5-5 (c).

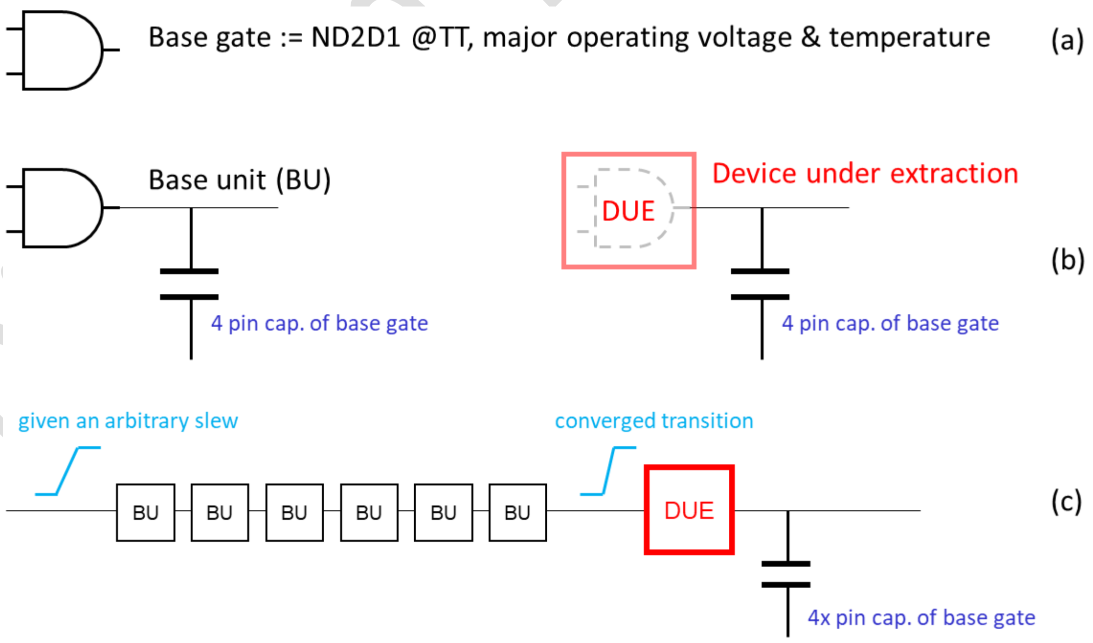


Fig. 5-5 Cell Metric Extraction

Currently, standard cell libraries primarily use the Liberty format (such as NLDM, CCS, LVF, etc.) for characterization. However, this format includes large data volumes, legacy burdens, and redundant data blocks defined by various EDA vendors for their tools. To accelerate data extraction, analysis, and debugging, we convert key timing and power information into a lightweight JSON format (see the open-source project https://github.com/dipsci/DTCO/tree/main/libMetric).

We then apply Least Squares Regression (LS Regression) to each cell's timing path, retaining only the regression coefficients (LSC) and generating a concise snapshot with core parameters, including area, input capacitance, driving strength, timing, and power, enhancing efficiency and accuracy. Finally, Liberty files from various PVT corners are consolidated into a unified database, enabling seamless querying and batch comparisons, optimizing data management and workflows, as shown in Fig. 5-6.

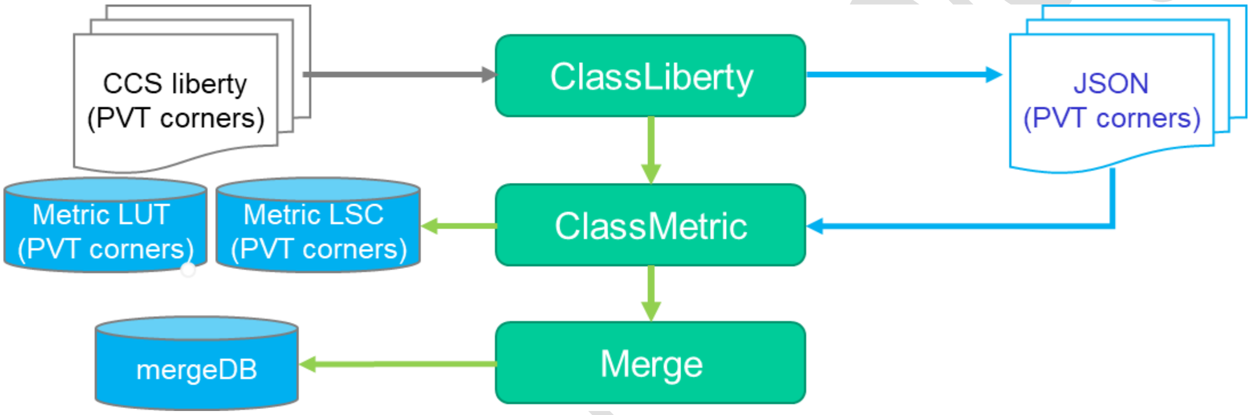
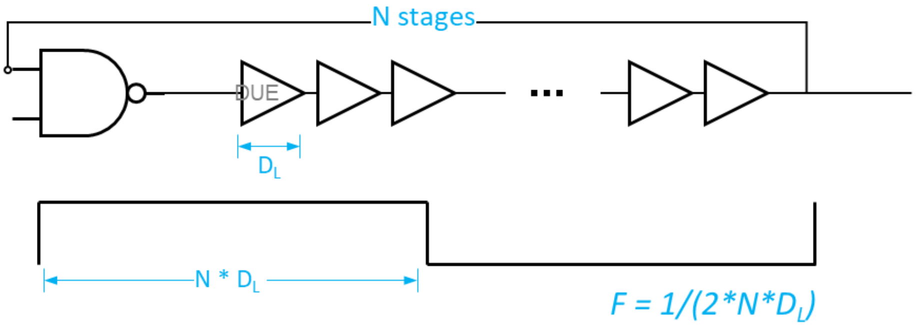


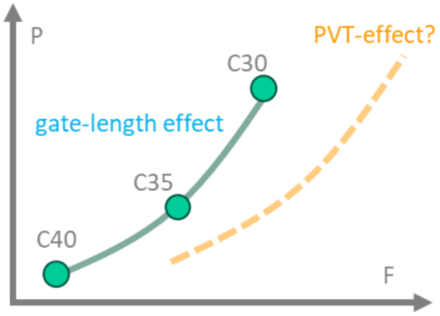
Fig. 5-6 Metric Extraction Flow

By extracting and analyzing physical features, we can efficiently study each cell's behavior under varying input transitions, load conditions, and different process parameters, such as channel length, threshold voltage (Vth), operating voltage, and temperature.

5.3. RO Simulation

A common method for process and device performance evaluation involves designing a ring oscillator (RO) and running SPICE simulations to generate a 2D scatter plot of frequency and power consumption. However, using the extracted feature data, we can bypass the need to construct an RO or run SPICE simulations, avoiding complex steps like logic synthesis, gate-level simulation, LVS verification, and SPICE circuit extraction. The approximate calculation method shown in Fig. 5-7 enables quick estimation of results.





$$P(int.) \coloneqq \frac{uW}{MHz} = \frac{10^{-6}W*sec}{10^{6}\ op} = \frac{10^{-12}W*sec}{op} \quad (pJ/op)$$

$$P' \coloneqq F * P(int.) = \frac{\cancel{op}}{\cancel{sec}} * \frac{10^{-12}W*\cancel{sec}}{\cancel{op}} \quad (pW)$$

$$P \quad \coloneqq N * P' * 10^{-3} + N * Leak \quad (nW)$$

Fig. 5-7 RO Simulation

*P*int. represents the internal power consumption of the component, measured in $\mu W/MHz$(or pJ/op, the energy consumed per circuit transition). Given the target frequency, the power consumption is the product of these values (unit: *pW*). The required number of stages N can be calculated, neglecting the delay of the first-stage NAND gate. The RO frequency is the reciprocal of the delay period $2N{\cdot}D_L$, where D_L is the delay snapshot extracted from the component (DUE) under specific conditions.

5.4. Standard Cell Library Batch PPA Benchmarking

Using the methods described, we can efficiently compare and evaluate cell libraries across multiple PVT corners. For example, in cell library A, SVT cells from C20 to C16 show a negative gain in total power consumption (including dynamic and leakage power), which further decreases with higher operating frequency and switching rate, as shown in Fig. 5-8.

When selecting cell library A for practical applications, low-speed designs typically require leakage optimization through adjustments to threshold voltage (Vth) and channel length, while excluding SVT and C20 cells to avoid negative effects. For high-speed designs (e.g., CPUs or GPUs), it may be necessary to temporarily disable leakage optimization or replace C20 cells to reduce power consumption and meet performance requirements.

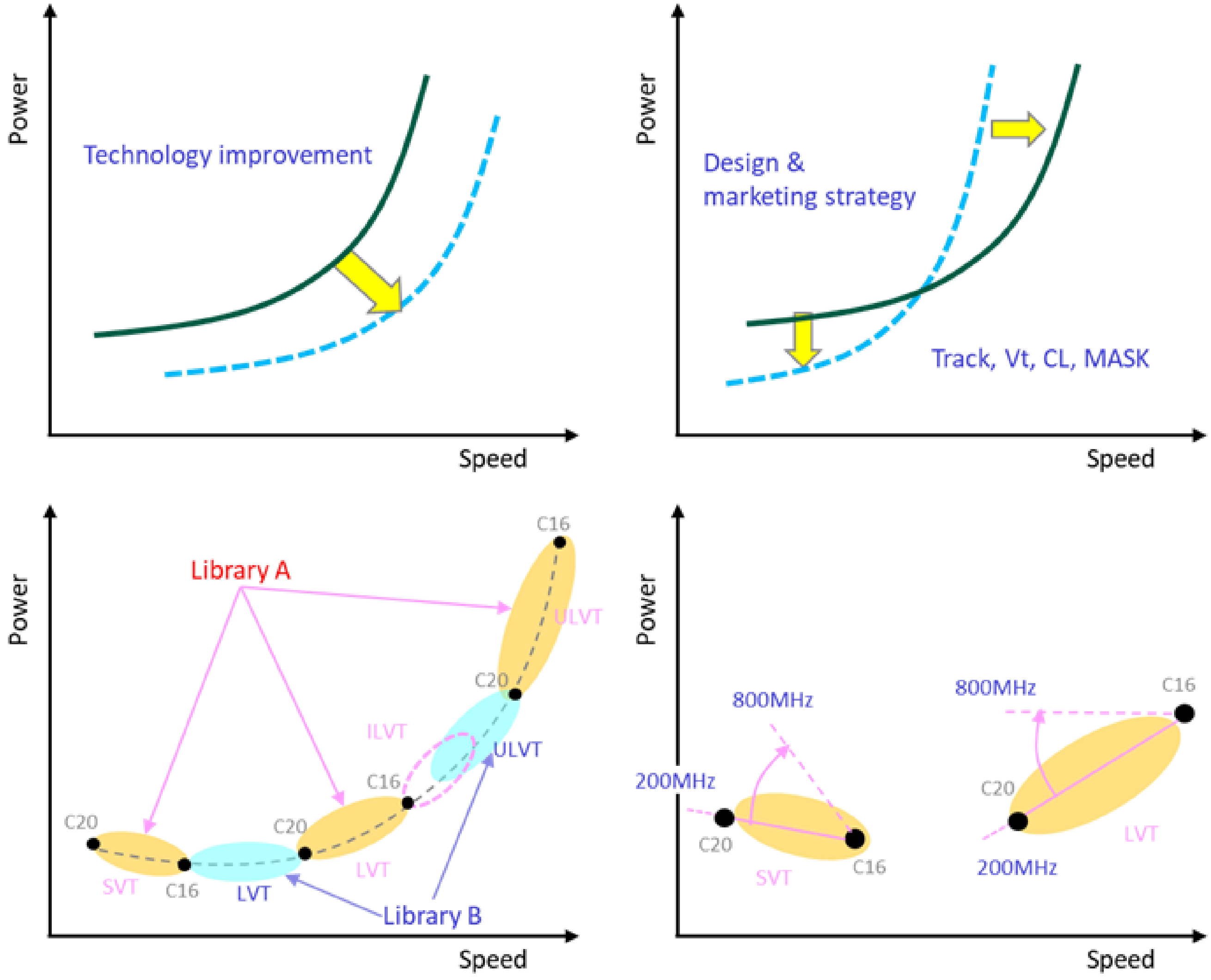


Fig. 5-8 Library Evaluation

Library Metrics helps engineers assess potential risks early in design, such as voltage-dependent requirements, temperature sensitivity, and timing Re-K cell skew or timing window shifts. As shown in Fig. 5-9, designers can evaluate setup and hold window requirements for DFFs. If setup skew issues arise (e.g., good front-end timing but inadequate back-end hold time), components can be marked as 'don't use' to improve efficiency and reliability.

With LS Coefficients recording timing characteristics, engineers can adjust conditions for specific timing arcs or expand the working range (via vector inner product) to explore trends, as shown in the right side of Fig. 5-9.

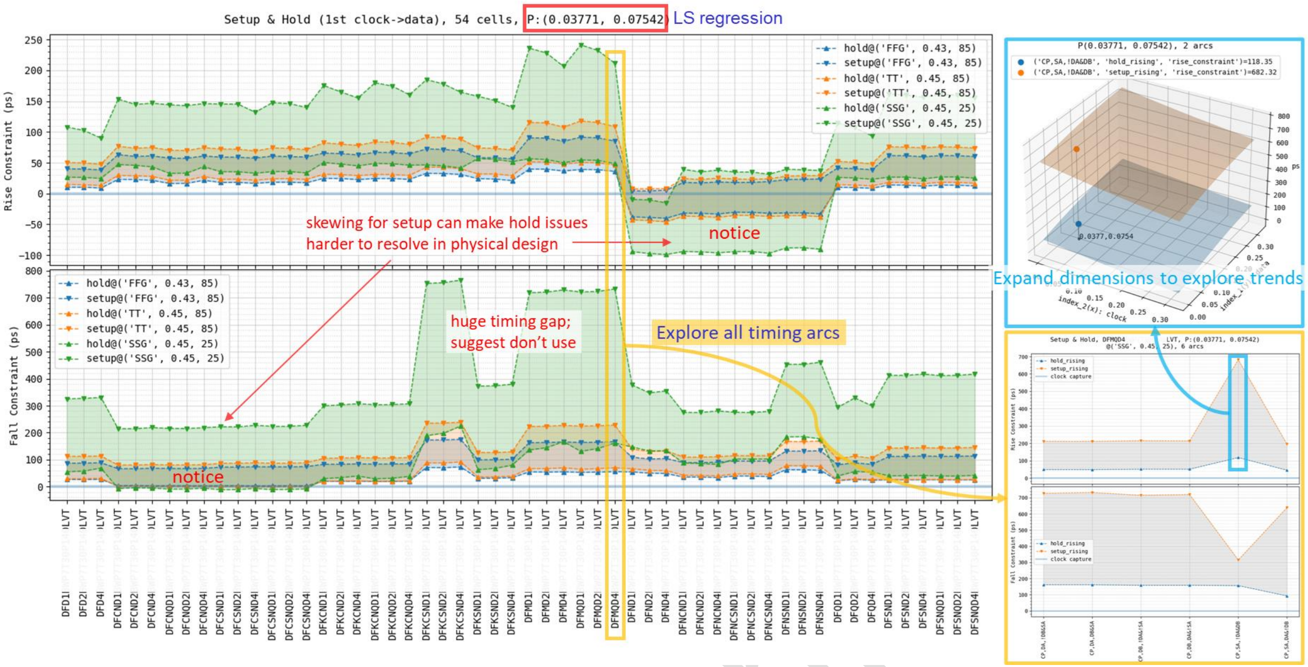


Fig. 5-9 Timing Constraint Batch Comparation

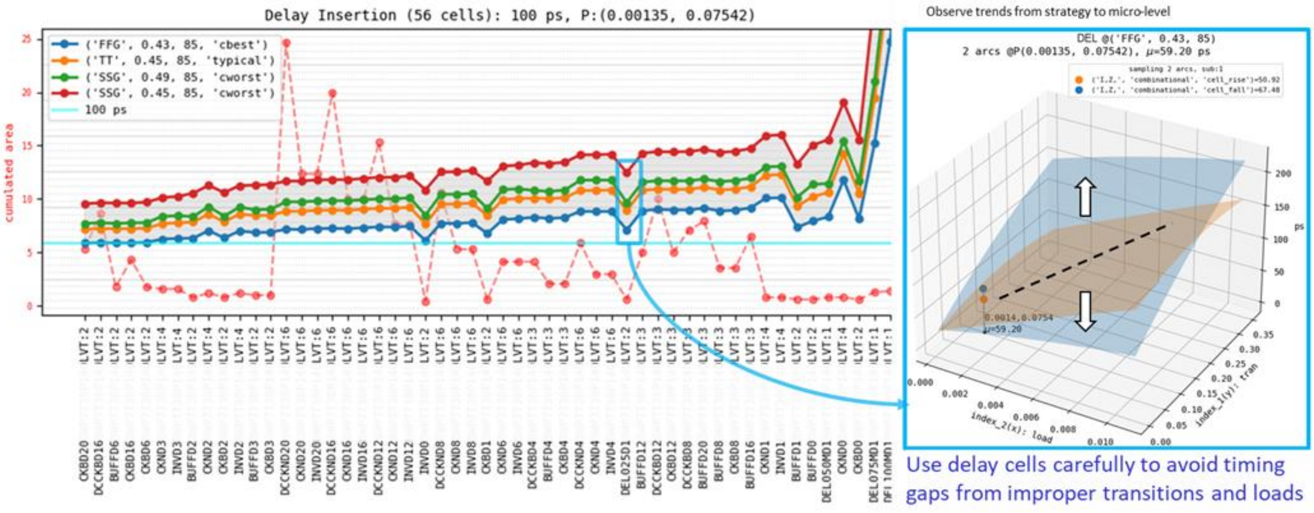


Fig. 5-10 Slew Balance Batch Comparison

As shown in Fig. 5-10, batch feature extraction helps engineers identify risks in the selected cell library and offers optimization suggestions. Based on requirements, engineers can choose appropriate cells for clock-tree optimization or hold buffer correction. This method also detects transition rise/fall imbalances early and predicts cell performance under various PVT conditions. Additionally, before physical design, design specifications and strategies can be formulated, such as:

- **Initial Design Phase Deactivation**: For Cell Library A, in the 200MHz to 1.2GHz range (10% switching rate), it is recommended to temporarily disable SKP, SKN, and OPT cells.

- **Driving Strength Evaluation**: Prioritize the placement of PLLs and clock inverters to meet drive strength requirements for critical paths, with feedback for custom cell adjustments if necessary.

- **Routing-friendly Evaluation**: For complex cells with a high pin count (e.g., INR, IND, IINR, IIND, IAO), increase size to at least D2 to avoid routing bottlenecks and layout issues, or provide feedback for optimization, as shown in Fig. 5-11.

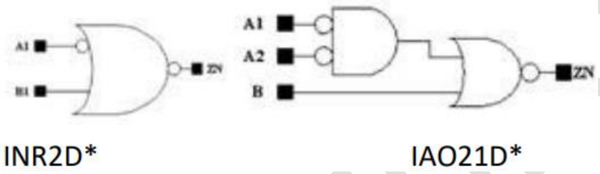


Fig. 5-11 Cell Candidates for Routability Optimization

Chapter 6. On-Chip Sensor Design and Integration (GRO Compiler)

As machine learning evolves, on-chip monitors like Ring Oscillators (RO) are shifting from optional to essential components. However, challenges remain in design optimization, architectural simplification, and maximizing benefits. To tackle these, we can focus on the following strategies:

- **Objective Clarification**: What are our goals? For example, understanding the deviation between SPICE models and actual silicon performance, optimizing process parameters and signoff recipes, or developing chip compensation strategies.

- **Indirect Benefits**: What additional insights can machine learning provide? For example, analyzing wafer uniformity and quantifying on-chip variation (OCV).

- **Advanced Applications**: How can we refine data analysis and enable real-time monitoring? For instance, dynamic IR sensors, cycle-based slack alerts, and dynamic adjustments with compensation strategies.

By addressing these questions, we can define clear objectives and systematically plan RO integration into the chip. Through traditional design and testing processes, we gather targeted data to optimize the process and enhance chip performance. Figure Fig. 6-1 shows the architecture of a Grid Ring Oscillator (GRO), where various components are integrated with selectable delay lines, managed by an SPI controller. This design allows multiple ROs to be strategically placed across the chip, particularly in areas affected by IR drop, for more efficient monitoring.

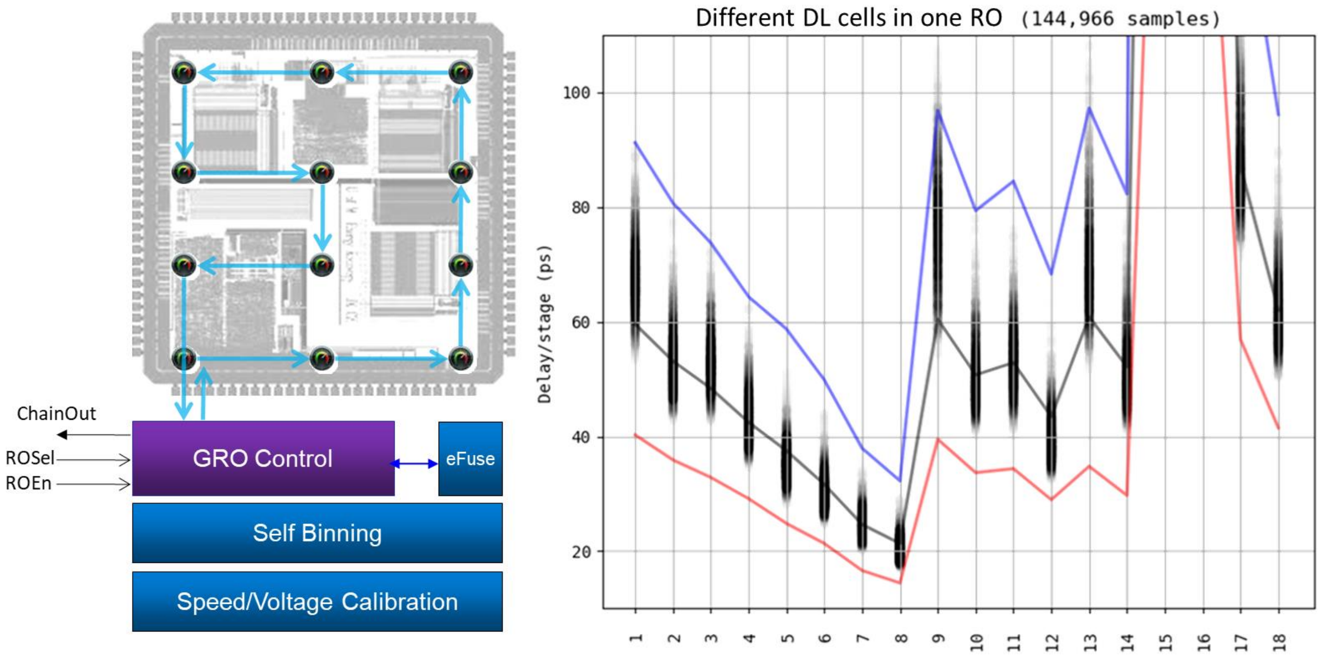
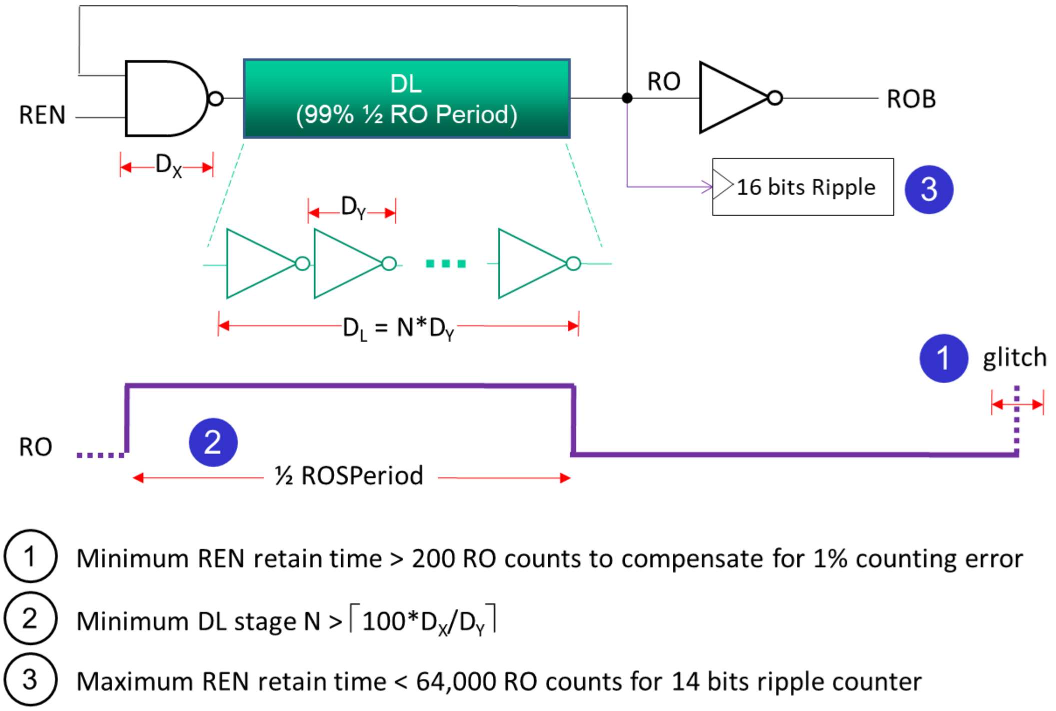


Fig. 6-1 GRO Integration and SPICE-Silicon Correlation

As shown in Fig. 6-2, a typical RO (Ring Oscillator) circuit is controlled by a NAND gate, with the REN pin acting as a switch and multiple delay elements (Delay-Line, DL) connected in series. A multi-bit counter records the output frequency. For comparing cell delays with a SPICE model, the delay elements should occupy over 99% of the RO cycle to minimize control gate variability. In the layout, it's recommended to arrange RO groups vertically or staggered and ensure sufficient power distribution to reduce the impact of localized dynamic voltage drops on data accuracy.



1. Minimum REN retain time > 200 RO counts to compensate for 1% counting error
2. Minimum DL stage $N > \lceil 100*D_X/D_Y \rceil$
3. Maximum REN retain time < 64,000 RO counts for 14 bits ripple counter

Fig. 6-2 RO Design Guideline

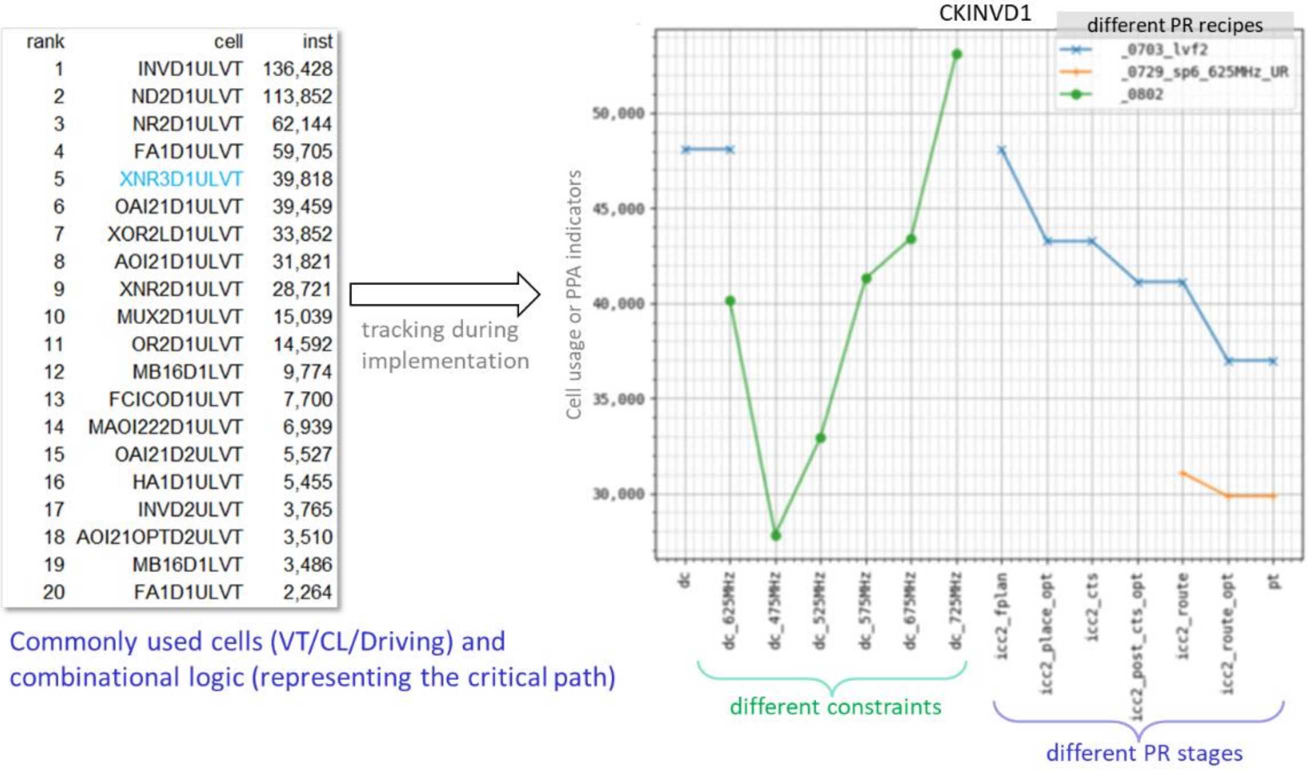
We can use the unique chip identifier (UID) to adjust performance ranking strategies and explore new RO structures, such as connecting two distant functional ROs to form a larger-scale gross-effect RO. Additionally, RC variation effects, though related to component drive capability, can be indirectly quantified through data analysis to derive a regression model.

6.1. Goal-Oriented RO Design

The challenge is maximizing design efficiency while minimizing area overhead. RO delay elements (DL) can be built from various standard cells, but selecting the optimal combination is key.

As shown in Fig. 6-3, the physical design process spans multiple stages, from design constraints and F/V Shmoo evaluation to pre-/post-CTS and post-route. Analysis of cell usage reports reveals that 80% of area and power consumption stems from 20% of core cells and clock-tree components, making them prime targets for S2S (SPICE-to-Silicon) optimization. For example, SHA-3 relies heavily on XNR/XOR cells, while Switch modules often use multi-bit MUX elements.

To evaluate N/P process balance, P-MOS stacked NOR and N-MOS stacked NAND can serve as auxiliary indicators in post-silicon measurement, providing key insights for process optimization while meeting performance demands.



rank	cell	inst
1	INVD1ULVT	136,428
2	ND2D1ULVT	113,852
3	NR2D1ULVT	62,144
4	FA1D1ULVT	59,705
5	XNR3D1ULVT	39,818
6	OAI21D1ULVT	39,459
7	XOR2LD1ULVT	33,852
8	AOI21D1ULVT	31,821
9	XNR2D1ULVT	28,721
10	MUX2D1ULVT	15,039
11	OR2D1ULVT	14,592
12	MB16D1LVT	9,774
13	FCICOD1ULVT	7,700
14	MAOI222D1ULVT	6,939
15	OAI21D2ULVT	5,527
16	HA1D1ULVT	5,455
17	INVD2ULVT	3,765
18	AOI21OPTD2ULVT	3,510
19	MB16D1LVT	3,486
20	FA1D1ULVT	2,264

Fig. 6-3 Cell Usage and PPA Evaluation

6.2. SPICE-to-Silicon Correlation

As shown in Fig. 6-4, post-silicon RO measurements align precisely with SPICE simulations under different process conditions (SS, TT, FF), enabling accurate identification of distribution and deviation from SPICE predictions with pico-second accuracy. Incorporating Lot-Wafer ID or time-axis parameters allows for tracking process adjustments and assessing control stability.

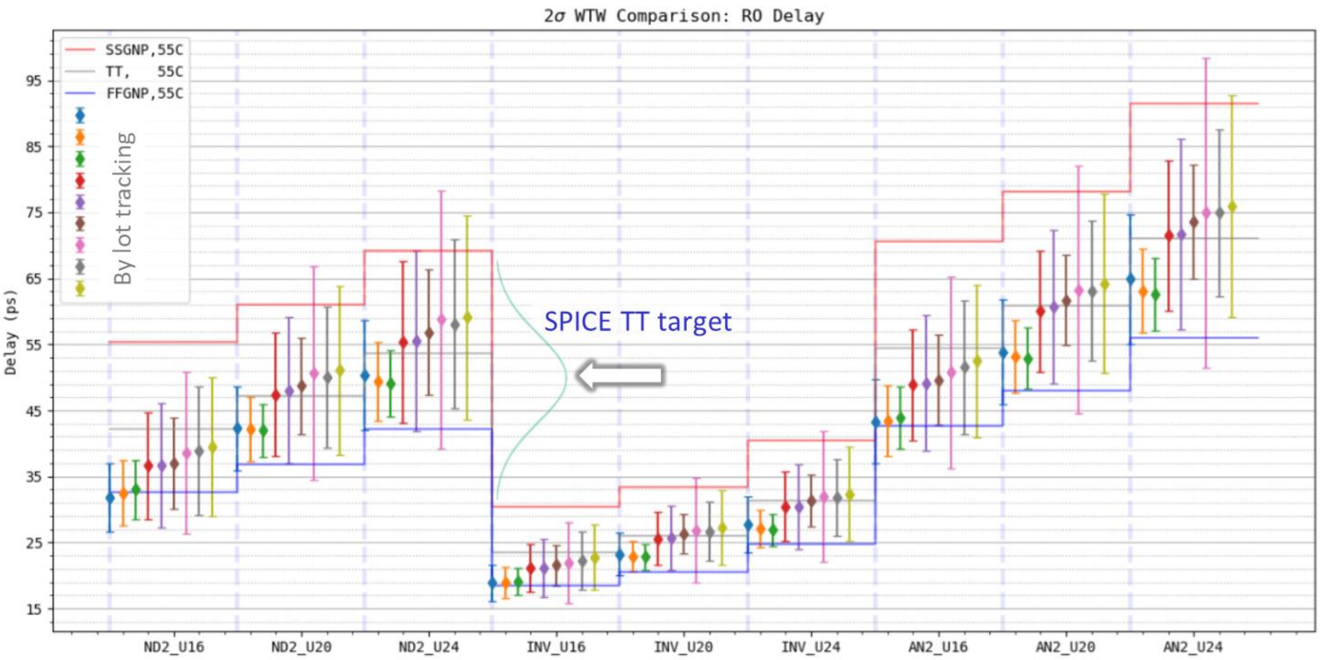


Fig. 6-4 RO SPICE-Silicon Correlation

With limited measurement flexibility in post-silicon testing and uncontrollable factors like IR drop and temperature variations, SPICE simulations cannot cover all scenarios. A regression model built on sparse data can predict unmodeled conditions, enhancing calibration and analysis efficiency, as shown in Fig. 6-5.

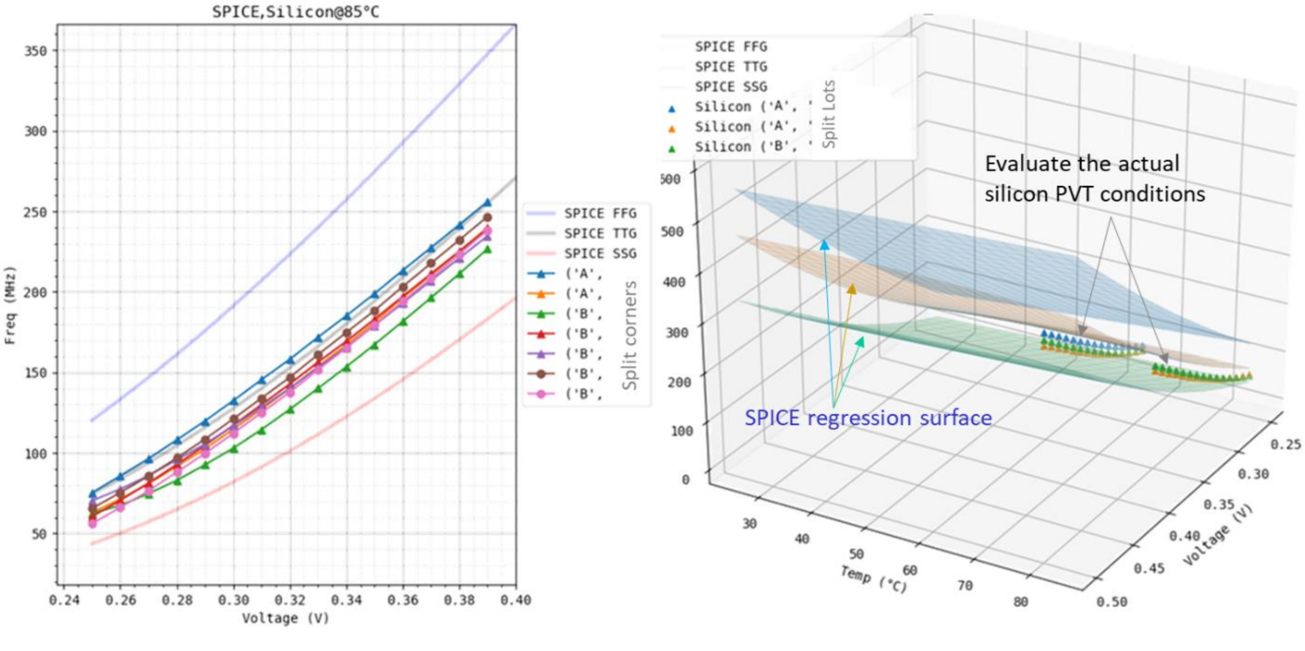


Fig. 6-5 V-F Shmoo and SPICE Correlation

6.3. Process Monitoring and Optimization

By integrating foundry WAT/CP data with on-chip RO measurements, engineers can track process variations (e.g., Vth shift, SIDD changes) and assess their impact on circuit performance, as shown in Fig. 6-6. RO's high-sensitivity timing data further aids in evaluating process uniformity and local variability. Key optimization strategies include:

- **Process Calibration**: Adjust critical process parameters, refine timing extraction (S2S/Re-K), and align timing signoff targets to mitigate process drift.
- **Functional Compensation**: Leverage ML-driven binning and dynamically adjust chip parameters (e.g., DVFS) to counter process inconsistencies.
- **Trend Analysis**: Aggregate RO data across batches to identify long-term process trends for continuous optimization.
-

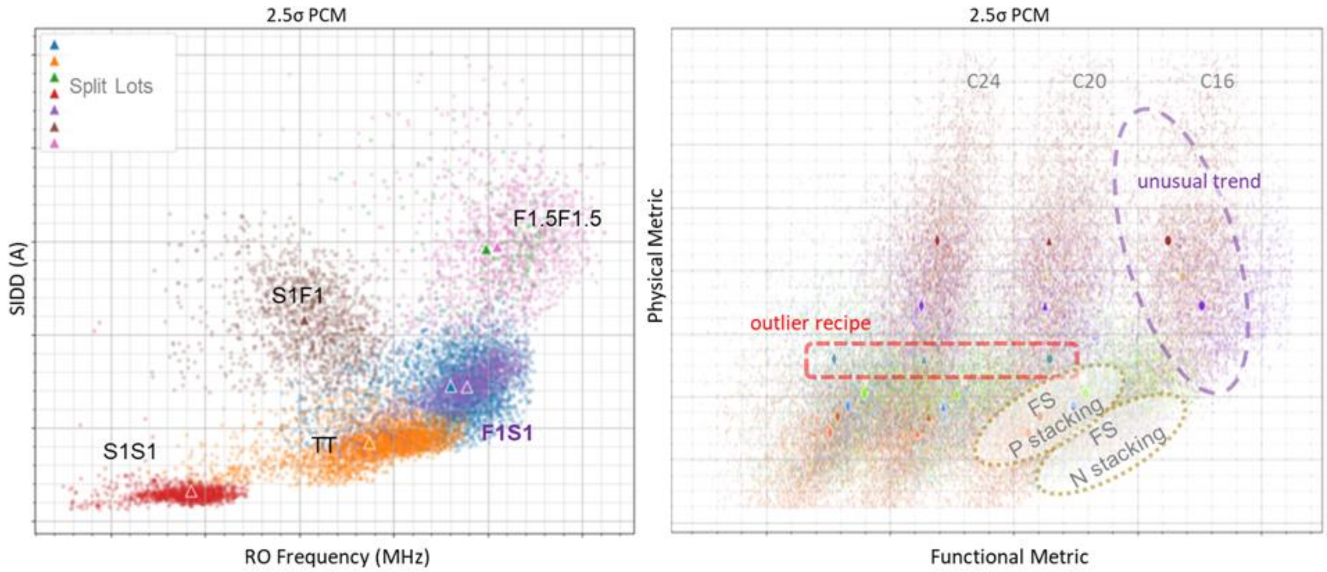


Fig. 6-6 Process Tuning

6.4. On-Chip Effective Voltage Analysis

6.4.1. Local Voltage Distribution Monitoring

If space permits, high-density ROs can be evenly distributed across the chip, with V-F sweep measurements used to capture RO frequency, as shown in Fig. 6-7. The frequency is mostly linearly related to voltage, with process variations affecting the intercept, while the slope remains relatively stable. Each color in the figure represents the frequency distribution of 24 ROs at different locations within a single chip.

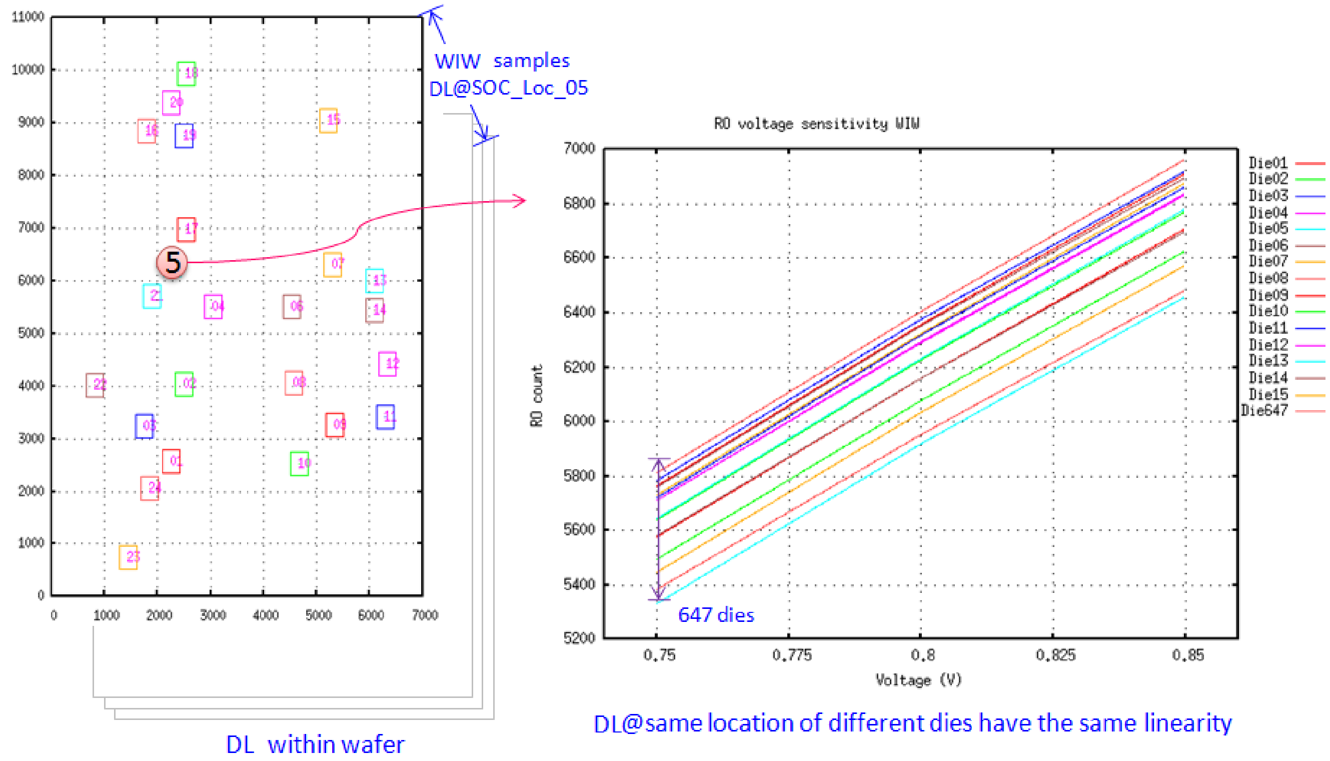


Fig. 6-7 On-chip Ring Oscillators (ROs) and V-F Curve

Using this characteristic, the RO frequency difference can be converted into relative voltage variation to reflect the voltage distribution. However, this method only applies to relative voltage differences and cannot estimate absolute voltage values. As shown in Fig. 6-8, different RO data can infer the equivalent voltage at each detection point, with the voltage distribution across a single chip appearing random and varying between chips.

However, as shown in Fig. 6-9, analyzing data from around 700 chips on the same wafer and averaging the frequencies at corresponding RO locations reveals a strong correlation between frequency and location (bold black line), which exceeds the impact of process variations. This trend remains consistent across SS, TT, and FF process conditions.

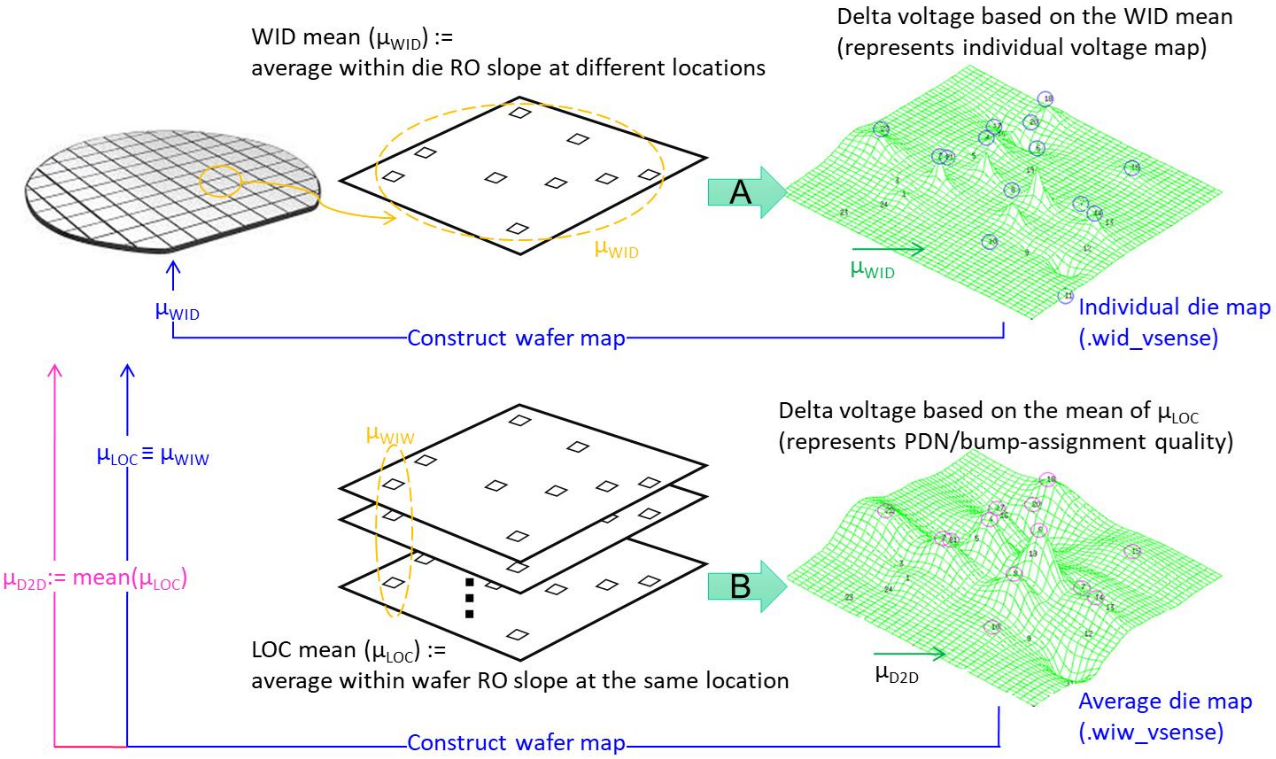

WID mean (μ_{WID}) :=
average within die RO slope at different locations
Delta voltage based on the WID mean
(represents individual voltage map)
μ_{WID}
A
μ_{WID}
μ_{WID}
Individual die map
(.wid_vsense)
Construct wafer map
μ_{WIW}
$\mu_{LOC} \equiv \mu_{WIW}$
Delta voltage based on the mean of μ_{LOC}
(represents PDN/bump-assignment quality)
μ_{D2D}:= mean(μ_{LOC})
B
LOC mean (μ_{LOC}) :=
average within wafer RO slope at the same location
μ_{D2D}
Average die map
(.wiw_vsense)
Construct wafer map

Fig. 6-8 On-chip Effective Voltage Regression

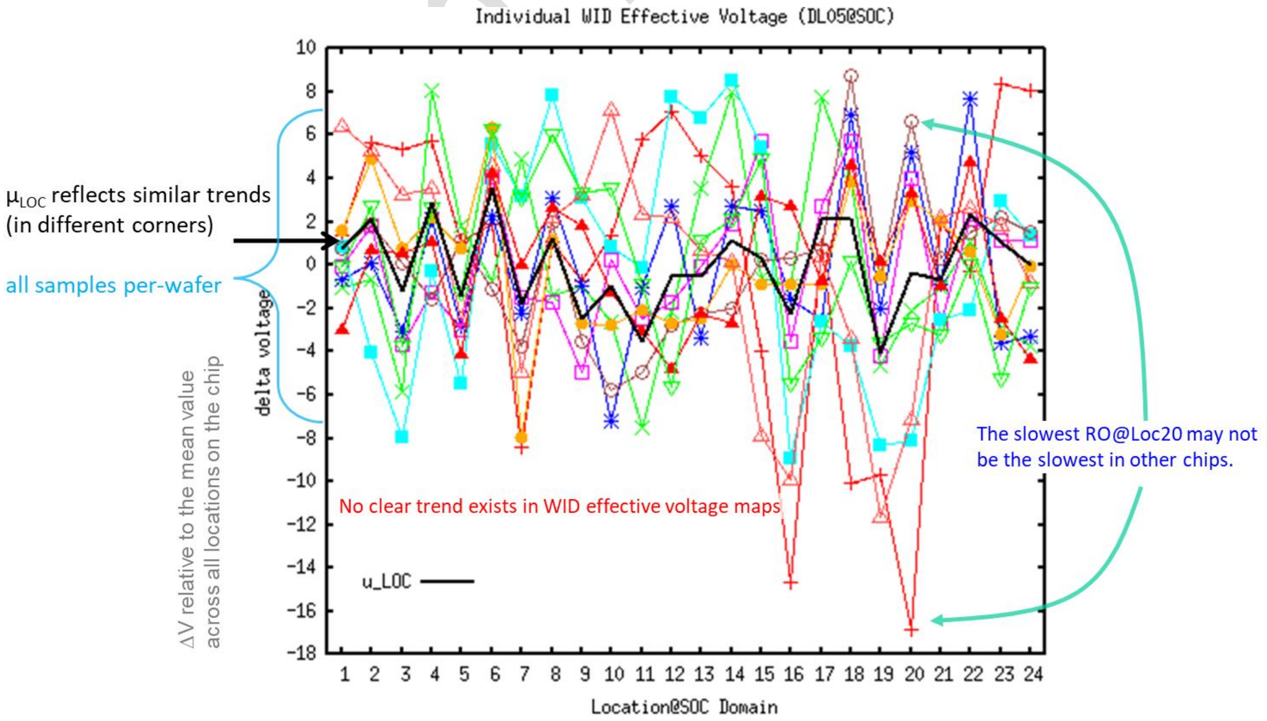
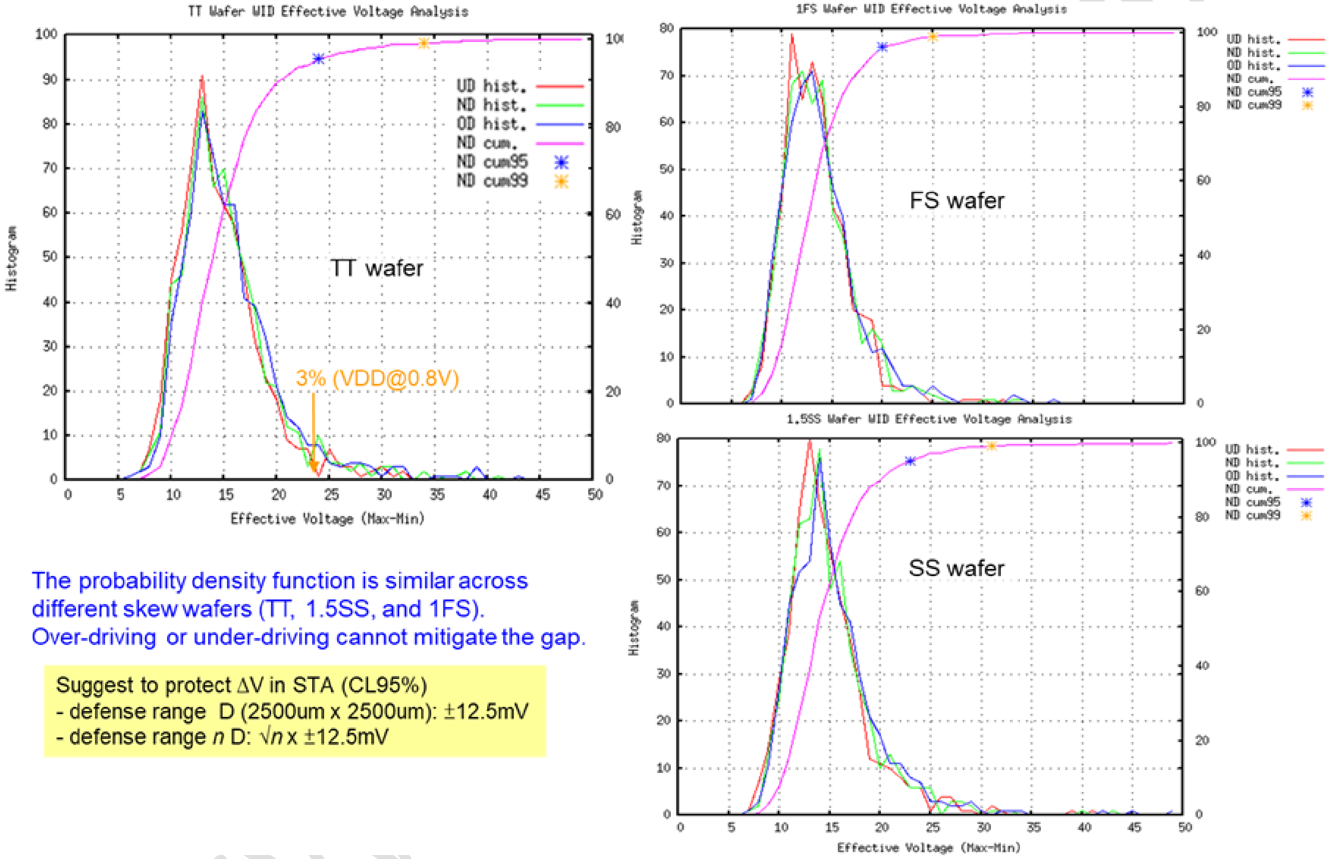


Fig. 6-9 On-chip RO Effective Voltage Analysis

This indicates a systematic issue or structural defect, possibly linked to factors like metal density, layout effects, or PDN/Bump distribution, causing uneven voltage distribution. Such factors must be addressed in the physical implementation flow. Additionally, when analyzing the WID (Within-Die) equivalent voltage difference distribution across all chips on the same wafer, the trend remains consistent across various process conditions (SS, TT, FF, or skewed processes like SF/FS). Notably, this equivalent voltage difference trend persists regardless of overall voltage adjustments, as shown in Fig. 6-10.



The probability density function is similar across different skew wafers (TT, 1.5SS, and 1FS). Over-driving or under-driving cannot mitigate the gap.

Suggest to protect ΔV in STA (CL95%)
- defense range D (2500um x 2500um): ±12.5mV
- defense range *n* D: √*n* x ±12.5mV

Fig. 6-10 Voltage-drop Probability Density

After performing cubic polynomial surface regression on the WID RO mean discrete data points, a heatmap of the effective voltage contours is generated, as shown in Fig. 6-11. The analysis shows a strong positive correlation between the voltage difference and power bump distribution density, with an inherent voltage difference range of -12mV to +6mV (chips powered but not running). Using a third-party ERA (Early Rail Analysis) tool, tap-current can predict potential issues and identify hotspots, which may develop into low-voltage regions on the chip.

This static effect is process-independent and non-random, and its impact on yield requires closer monitoring. Addressing voltage distribution issues early in the design phase, rather than relying on late-stage dynamic IR analysis, can reduce design risks and improve chip reliability and yield.

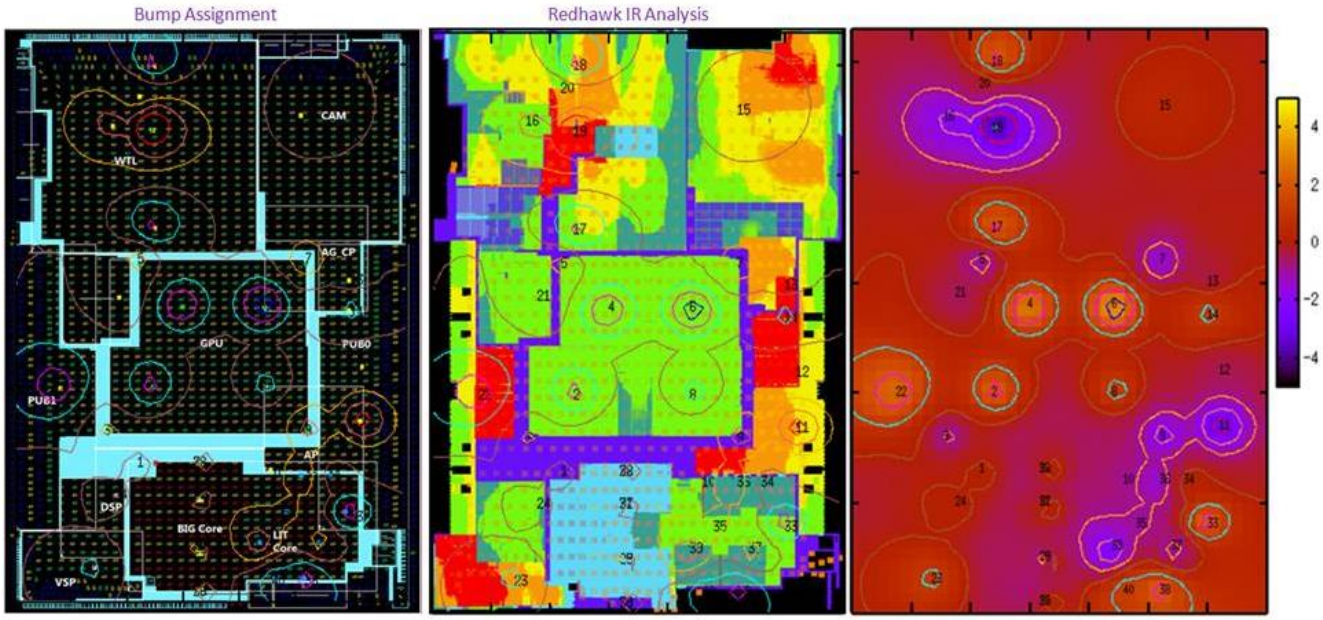


Fig. 6-11 Effective Voltage and ERA Correlation

6.4.2. Compensation Strategy

In the chip mass production process, unexpected results often arise due to various factors, such as mismatches between SPICE models and silicon, imperfections in the physical design process, overly aggressive or conservative timing signoff strategies, and improper process control (including production process parameter drift and wafer non-uniformity). As shown in Fig. 6-12, if significant adjustments are made to the process parameters for compensation—such as shifting from TT to FF process parameters—it may meet speed requirements but could lead to exponential increases in leakage current and power consumption, making the chip performance harder to control with temperature changes. Therefore, voltage compensation, whether static or dynamic, provides a more predictable and cost-effective method to balance performance and power consumption.

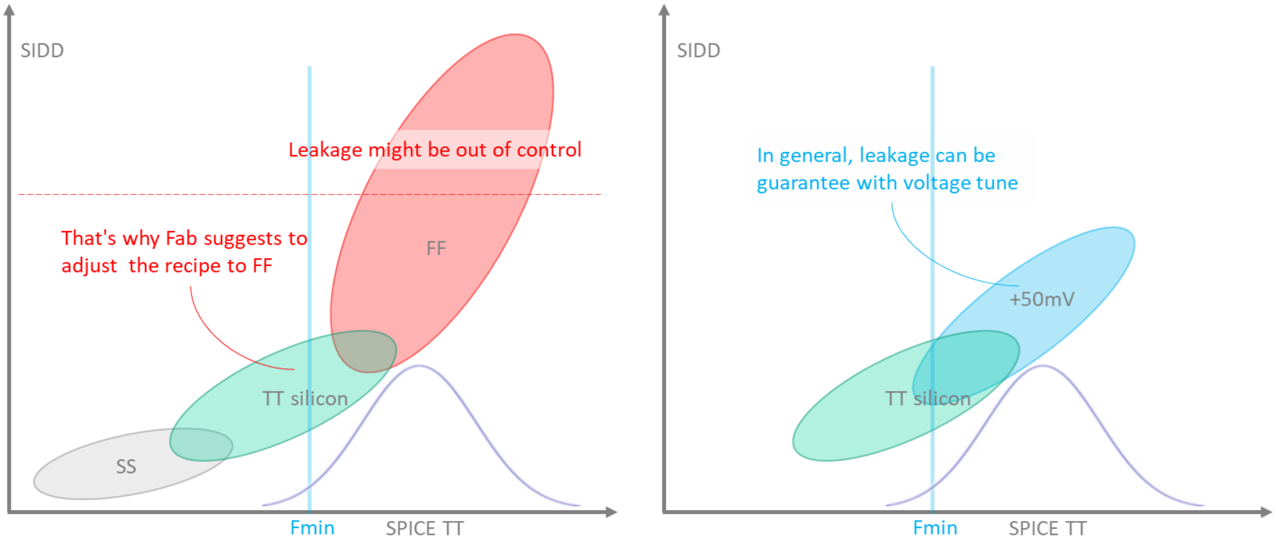

SIDD
Leakage might be out of control
That's why Fab suggests to adjust the recipe to FF
FF
TT silicon
SS
Fmin
SPICE TT
SIDD
In general, leakage can be guarantee with voltage tune
+50mV
TT silicon
Fmin
SPICE TT

Fig. 6-12 Process and Voltage Tuning

Process parameter shifts are not random or Gaussian, but often exhibit mechanical harmonic spins, which hinder design margin accuracy. These non-random effects, compounded by process and test environment deviations, introduce randomness, undermining what was thought to be a reliable strategy. Fig. 6-13 shows that wafer characteristics like RO frequency or SIDD change with voltage and temperature, revealing that local unevenness cannot be corrected by voltage compensation alone, and performance gradients cannot be eliminated through this method.

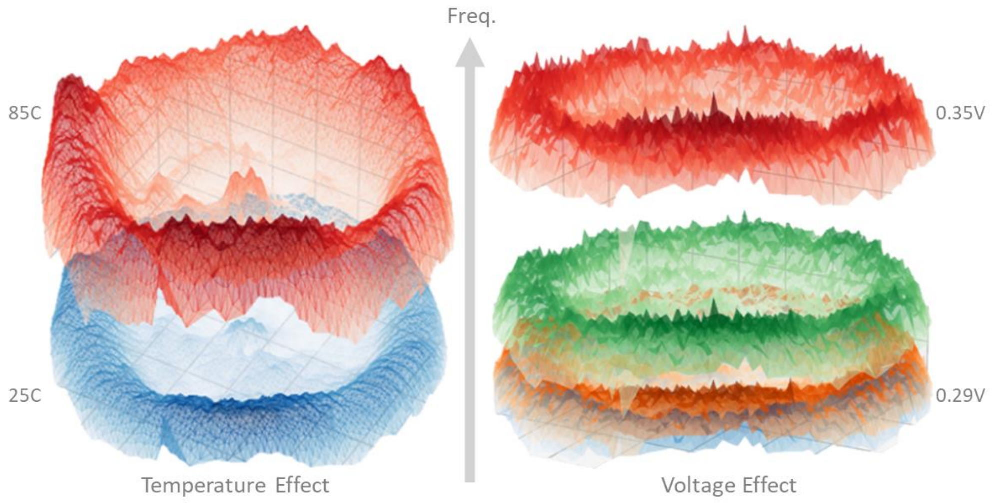


Fig. 6-13 RO Frequency vs. Temperature and Voltage Effects

In advanced chip design, integrating on-chip monitors (OCMs) with machine learning-based compensation mechanisms is crucial for enhancing performance and reliability. OCMs collect real-time parameters like voltage, frequency, and temperature, providing precise data for both static and dynamic compensation. Fig. 6-14 illustrates a multi-layered performance compensation strategy, including process optimization, chip binning, voltage-frequency configuration, and real-time adjustments, to improve chip performance and stability.

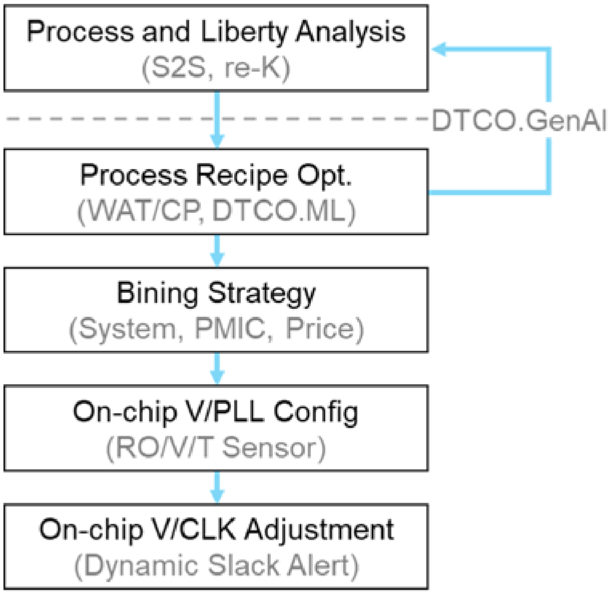


Fig. 6-14 Multi-level Compensation Strategy

By combining static RO with machine learning, designers can optimize process parameters, implement chip binning, and achieve system-level voltage compensation, balancing pricing and production capacity, as shown in Fig. 6-15. Real-time feedback from dynamic slack-alert circuits enables voltage and clock adjustments, balancing performance, power, and reliability. This method not only benefits high-performance computing chips but also boosts design efficiency and energy utilization in IoT and embedded systems.

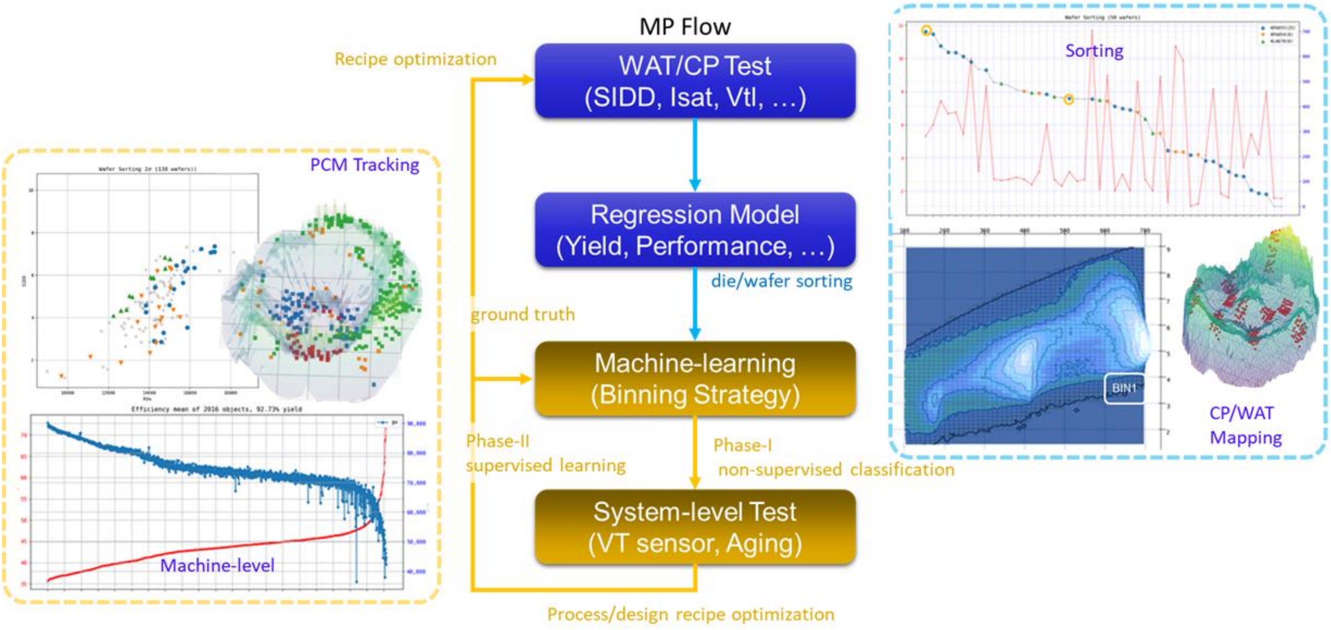


Fig. 6-15 Binning Strategy

6.4.3. Dynamic Timing Slack Alerts and Layout

RO-based sensing solutions typically have slower response times and cannot directly reflect timing slack in data paths. To address this, a dynamic timing slack alert system based on the RAZOR architecture, an advanced feature of chip monitoring IP, optimizes setup alert (SA) and hold alert (HA) configurations. This system measures timing slack at different locations in real-time based on internal voltage and temperature variations. The results are compiled into high-resolution bits, reflecting setup and hold time uncertainties in data paths. The system's serial output supports dynamic voltage and frequency scaling (AVS), effectively tracking timing signoff, avoiding overdesign, and identifying minimum frequency errors, supporting timing verification in physical design.

In practice, it is recommended to perform preliminary STA and IR analysis after P&R in the physical design phase. For areas with higher IR risks, select sub-critical timing paths with larger margins and replace the last stage DFF with SA/HA to ensure that penalties from additional capacitance or routing do not affect the performance of the critical path, as shown in Fig. 6-16.

In advanced technology nodes, timing signoff becomes more challenging, requiring careful consideration of process variations, temperature changes, and static/dynamic IR hotspots to avoid overdesign. Traditional methods typically rely on adding uncertainty to the clock tree and derating data path timing, which can lead to overdesign. To address this, embedding ring oscillators (RO) and simulating key data paths on-chip, correlating internal chip features such as leakage/dynamic currents (SIDD/DIDD) and maximum operating frequency (Fmax) during CP and WAT testing, simplifies timing signoff and introduces Design Technology Co-Optimization (DTCO) for advanced node circuit design.

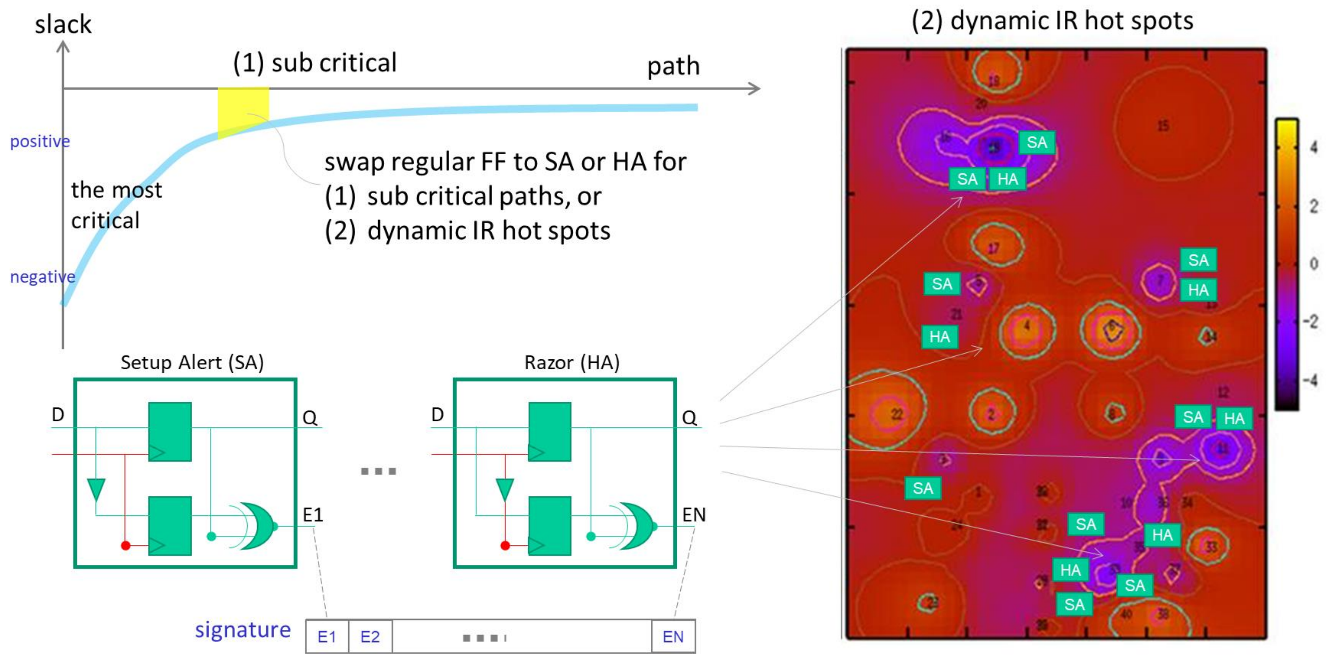


Fig. 6-16 Dynamic Slack Alert Integration Guideline

6.5. GRO Automation Tool and Verification Process

In the future, embedding monitoring circuits within chips will become widespread. However, due to the lack of standard specifications, the industry needs a reliable, convenient solution to automatically generate monitoring circuit IPs and test program samples. By combining standardized data from tests like eFuse, WAT/CP/Aging, and machine learning, scalable EDA applications such as chip performance grading and voltage/frequency compensation will emerge, requiring further planning and development.

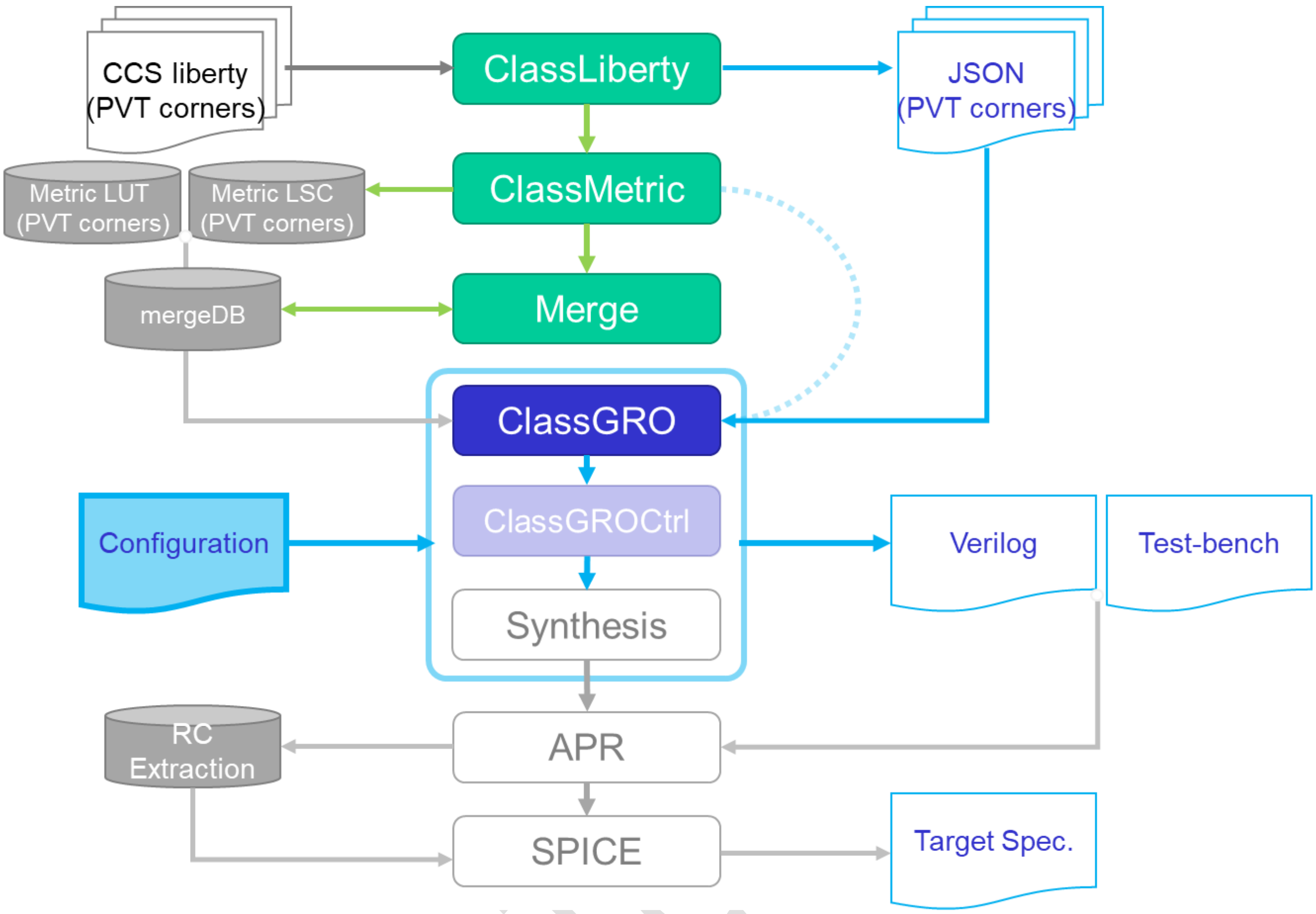


Fig. 6-17 GRO Automation Flow

Fig. 6-17 shows the proposed open-source EDA tool (https://github.com/dipsci/DTCO/tree/main/GRO) and its automated workflow. The tool reads a configuration file that defines target Delay-Line, frequency, and maximum count values. It automatically extracts Liberty data, retrieves timing and I/O information for key components, and generates the RO monitor RTL based on the target settings.

Additionally, the tool automates synthesis and test environment generation for circuit functionality and integrity verification, as shown in Fig. 6-18. The generated RO monitor includes multiple Delay-Lines, and after hardening and LVS verification, RC parameters are extracted for SPICE simulation. The simulation results serve as a calibration reference for SPICE-silicon comparison in post-silicon measurements.

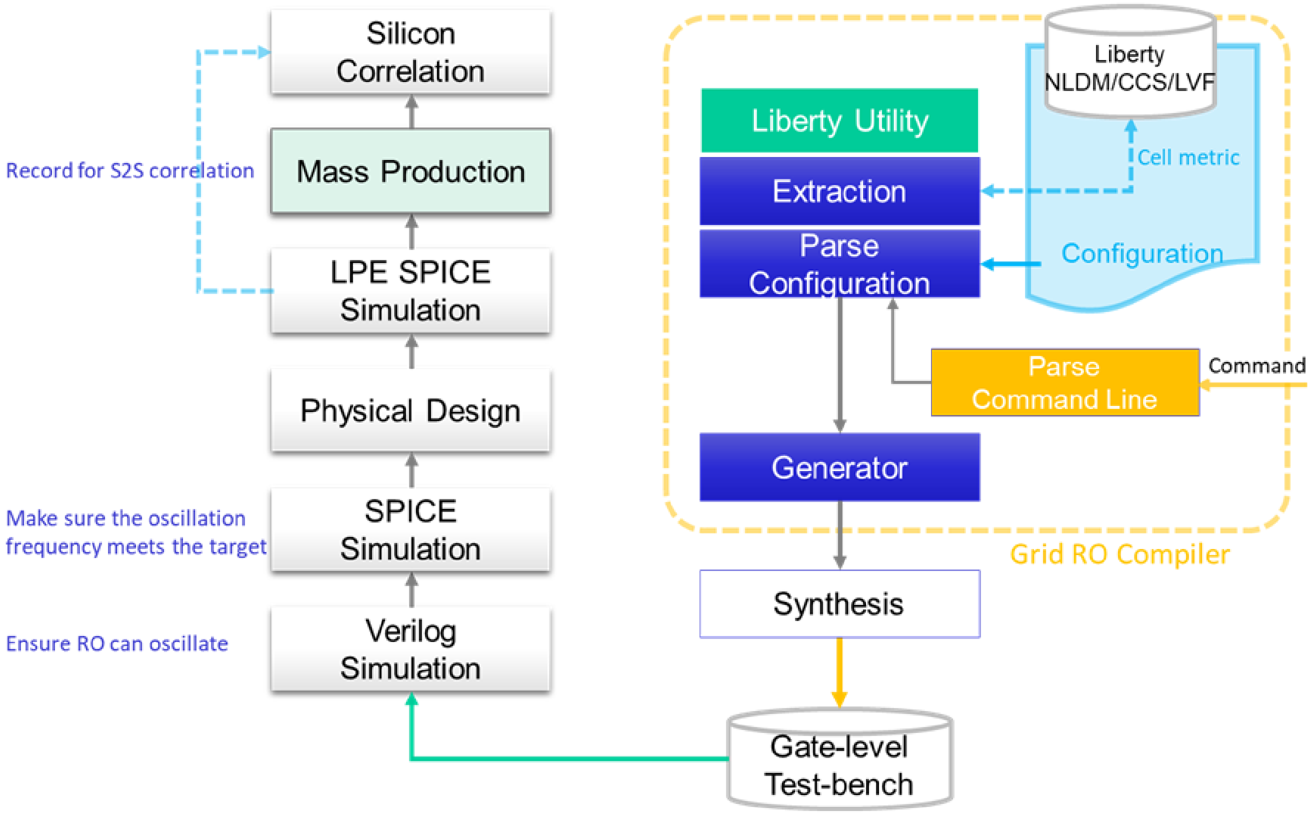


Fig. 6-18 GRO Verification and S2S Flow

Chapter 7. Data Analysis and Machine Learning Platform (Copernic™)

7.1. Data Standardization and Visualization

DTCO.ML is an innovative open EDA ecosystem (https://github.com/dipsci/DTCO/tree/main/copernic) aimed at enhancing data correlation and visualization in chip production and testing. With the current lack of unified data formats and standards in the industry, we must define a scalable format and develop conversion tools for seamless cross-domain data mapping. This improves data processing efficiency, ensures smooth integration of test data from different sources, and supports advanced methods like machine learning and neural networks. Ultimately, it optimizes chip energy efficiency and production capacity, leading to more accurate process and design decisions, as shown in Fig. 7-1.

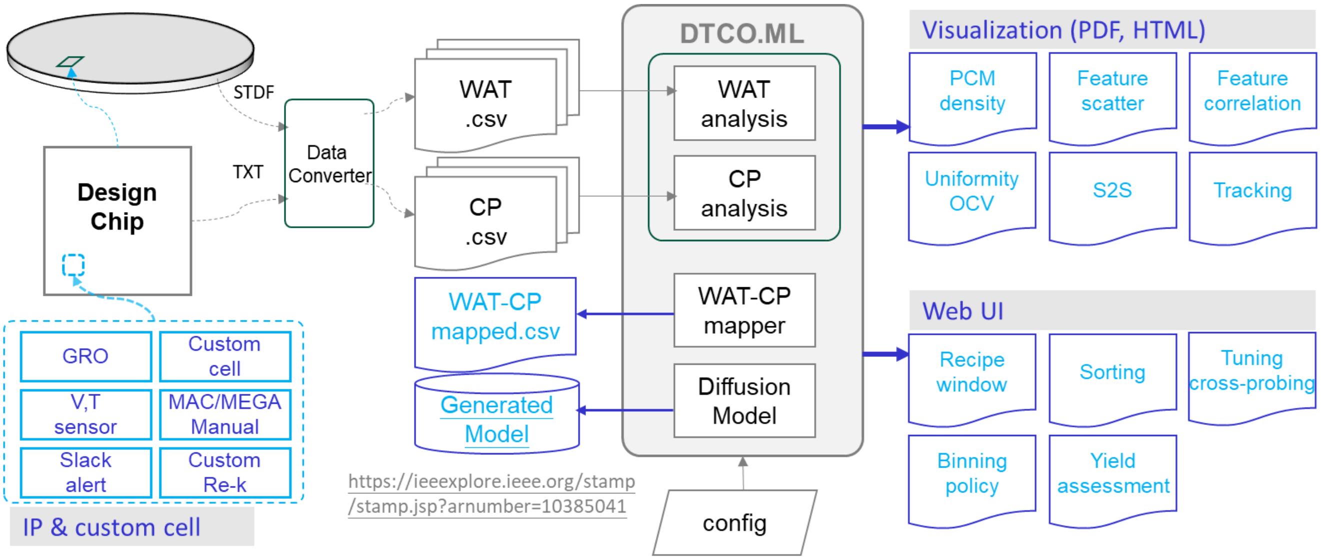


Fig. 7-1 DTCO.ML™ amd Copernic™ Platform

To ensure high-quality test data, pre-planning during design is crucial, such as integrating PVT sensors or embedding RO (Ring Oscillator) with custom circuit IP into the design process. This enables continuous collection of process and circuit data during mass production, forming a strong foundation for process optimization, timing analysis, and yield improvement.

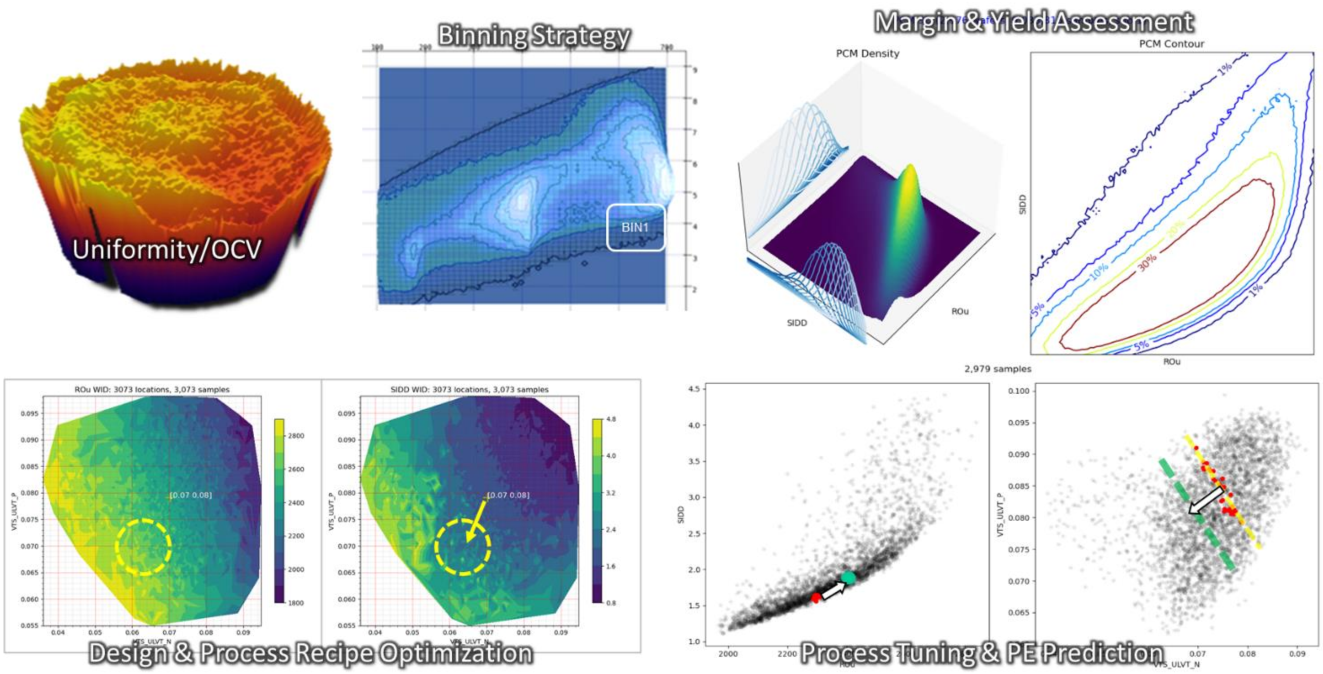


Fig. 7-2 Empowering DTCO.ML Applications

The DTCO.ML/Copernic platform offers intuitive visualization tools to track and analyze test data distribution and uniformity in wafer production. Using statistical and regression analysis, it estimates local variation (OCV) within the chip and optimizes design margins to reduce over-design. Cross-dimensional data correlation identifies optimal design parameters, offering chip performance binning and voltage compensation metrics to enhance yield and reliability. Data visualization showcases physical characteristic distributions and trends, helping engineers quickly identify the optimal process parameter ranges, as shown in Fig. 7-2.

7.2. Cross-Domain Mapping of Multi-Dimensional Data

Using Lot-Wafer-ID for mapping and standardizing WAT (Wafer Acceptance Test) and CP (Chip Performance) data reveals key correlations between wafer tests and chip performance, as shown in Fig. 7-3. This method identifies performance variations across process conditions and pinpoints key process variables that impact chip performance, enabling precise adjustments to design and process parameters, optimizing chip quality, yield, and supporting process improvements.

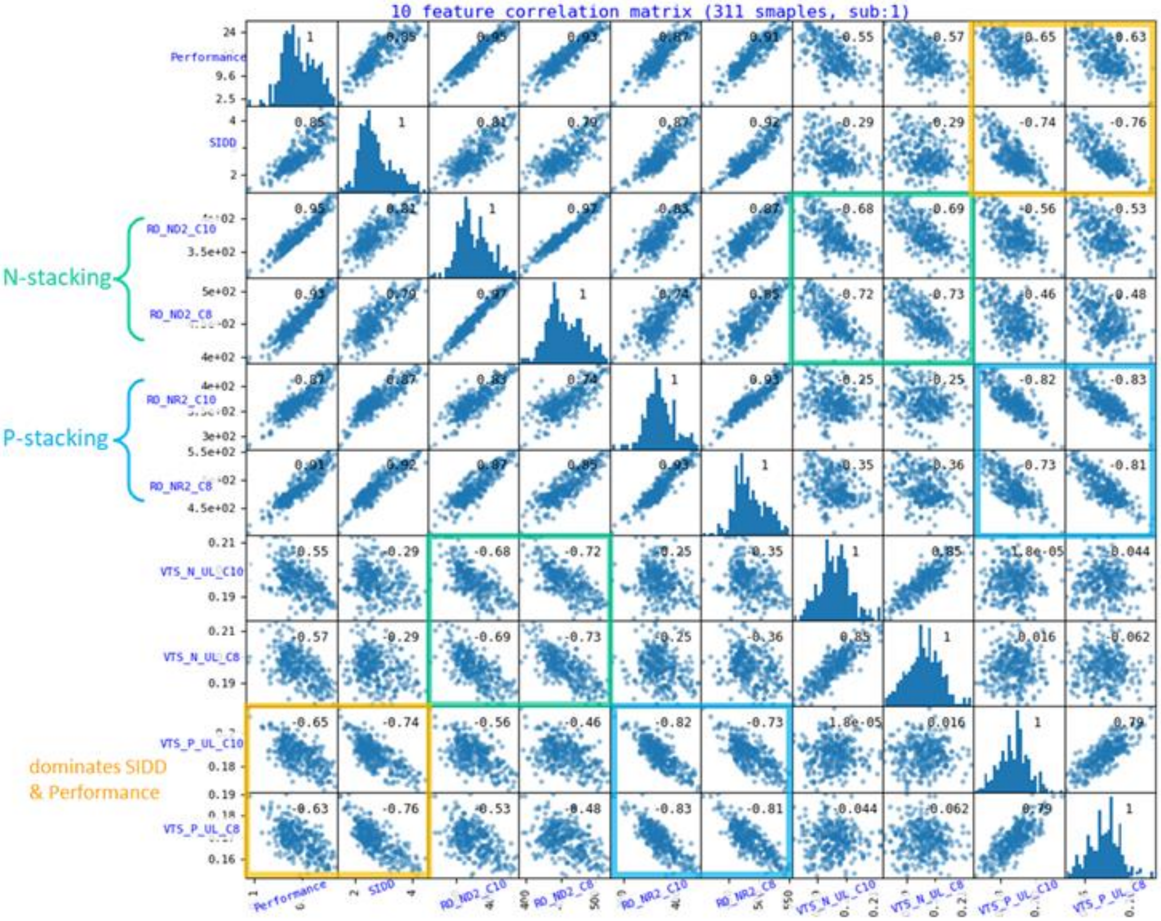


Fig. 7-3 WAT-CP Mapping and Correlation

7.3. Design Flow Integration Strategy

The DTCO.ML design flow incorporates SPICE-Silicon wafer correlation, process variation analysis, library cell optimization, and timing Re-K based on actual WAT parameter distribution. This enables effective defense strategies to enhance chip yield and reliability. Through WAT and CP/FT mapping, correlation analysis, and modeling, we quickly identify optimal process recipes and refine the binning strategy.

7.3.1. WAT-aware Timing Re-K

In current process technologies, real process parameters often deviate from SPICE models, as shown in Fig. 7-4. Especially under non-standard operating voltages, critical clock components may experience significant distortion. Relying solely on timing Re-K under varied voltage conditions can lead to functional failure. Analyzing WAT-SPICE model deviations is essential for accurate timing extraction from the standard cell library. By correlating the two, we adjust simulation calibration, timing margins, and defense strategies, dynamically adapting to the chip’s actual conditions. Understanding mass production trends and implementing defense strategies enhances yield and competitiveness.

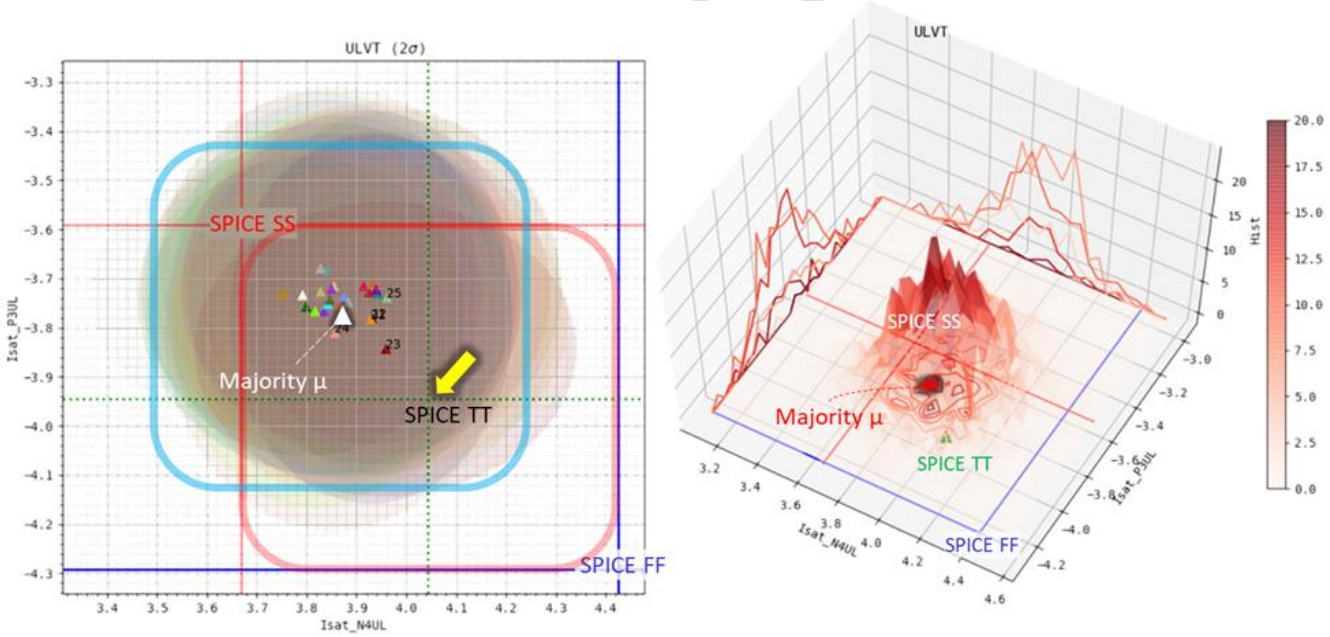
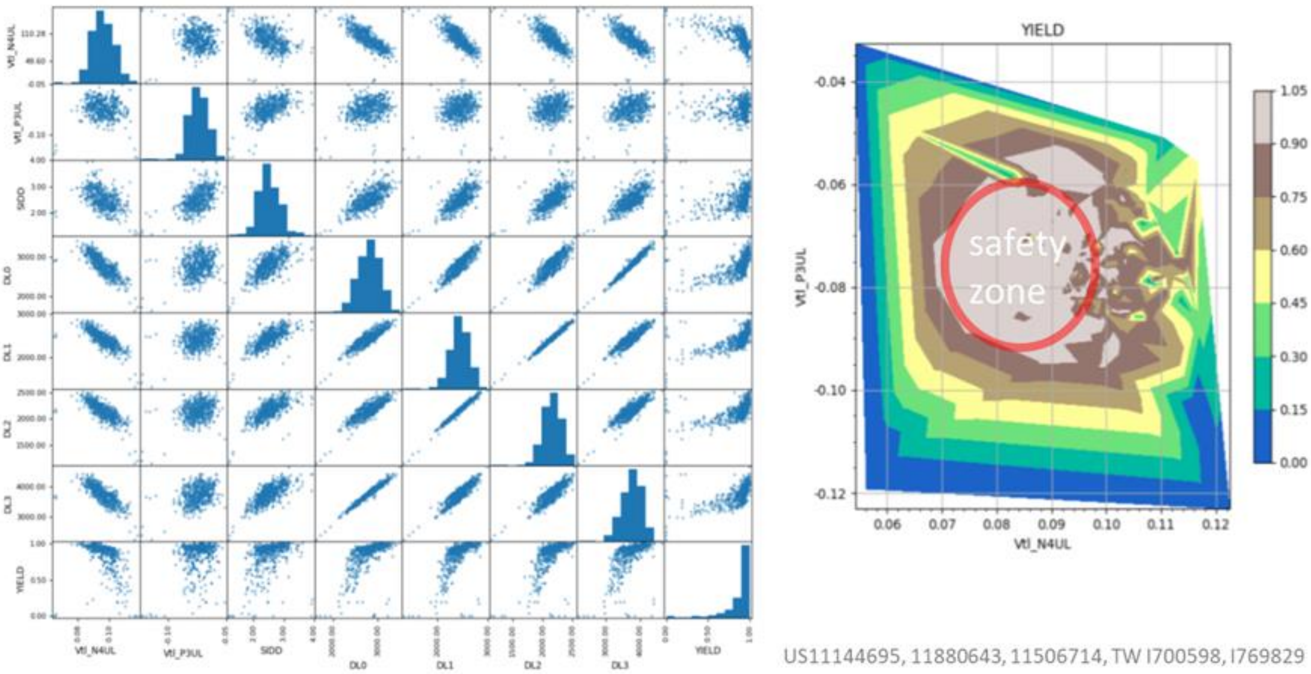


Fig. 7-4 WAT-aware SPICE-to-Silicon Correlation

7.3.2. WAT-CP Mapping and Correlation Analysis

The main challenge with WAT is insufficient data samples, leading to biased conclusions. To address this, we expand the data set using feature surface regression and correlate WAT with CP/FT/SLT test data. By plotting WAT's N/P process parameters (e.g., Vsat, Isat) on the XY axes and CP/FT/SLT test features (e.g., RO, leakage current, yield, Fmax) on the Z axis, we create a contour map to visualize the impact of WAT parameters on chip performance. Through cross-dimensional mapping and dimensionality reduction, wafer fabs and chip design engineers can align process and design strategies, ensuring consistent optimization, as shown in Fig. 7-5.



US11144695, 11880643, 11506714, TW I700598, I769829

Fig. 7-5 WAT/CP Correlation and Process Window Optimization

7.3.3. OCV Analysis

A single RO per chip is enough to estimate wafer uniformity, voltage differences, and OCV through machine learning. During CP testing, we calculate RO differences between die-to-die. As shown in Fig. 7-6, by selecting RO at the same coordinates on two wafers and gradually increasing the distance, we can use regression analysis to extrapolate to zero distance (D0) and estimate OCV, thus assessing on-chip variations.

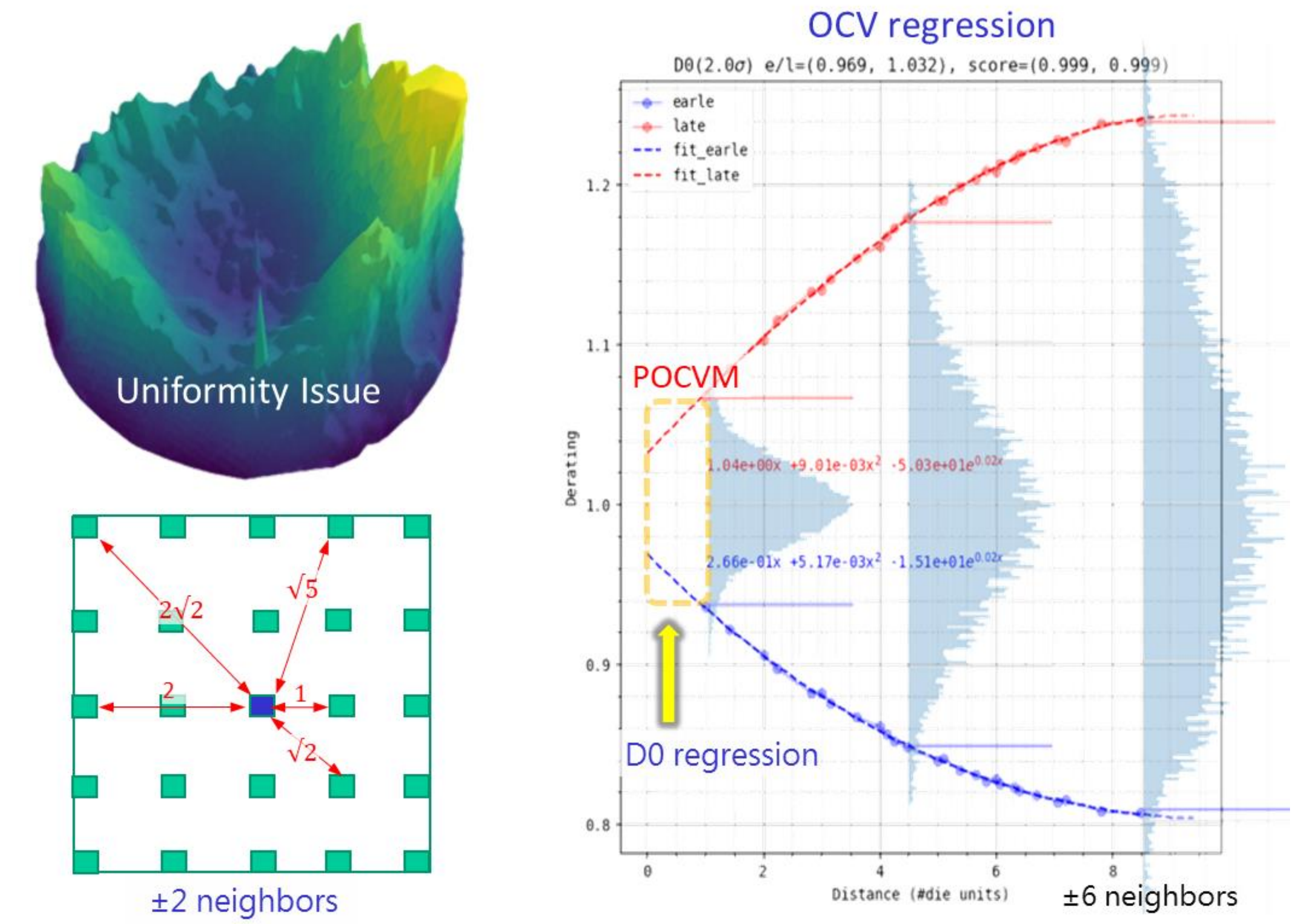


Fig. 7-6 On-chip Sensor and OCV Regression

7.4. OCV Analysis and Design Margin Optimization

System-level static chip performance gradients may result from surface irregularities caused by optical, chemical, and mechanical effects during wafer fabrication. As shown in Fig. 7-7, each chip contains six Ring Oscillators (RO). Using RO data from all chips on the wafer, we construct a feature surface with the Z-axis representing the RO frequency. A magnified view shows the RO frequency distribution across four adjacent chips, with the light pink surface representing the average frequency of the six ROs (in different colors).

This average surface reveals the performance gradient and on-chip variability (OCV), which affects chip performance uniformity. Non-random variations like these cannot cancel out. Additionally, random local fluctuations from ROs at different locations further exacerbate local performance variation.

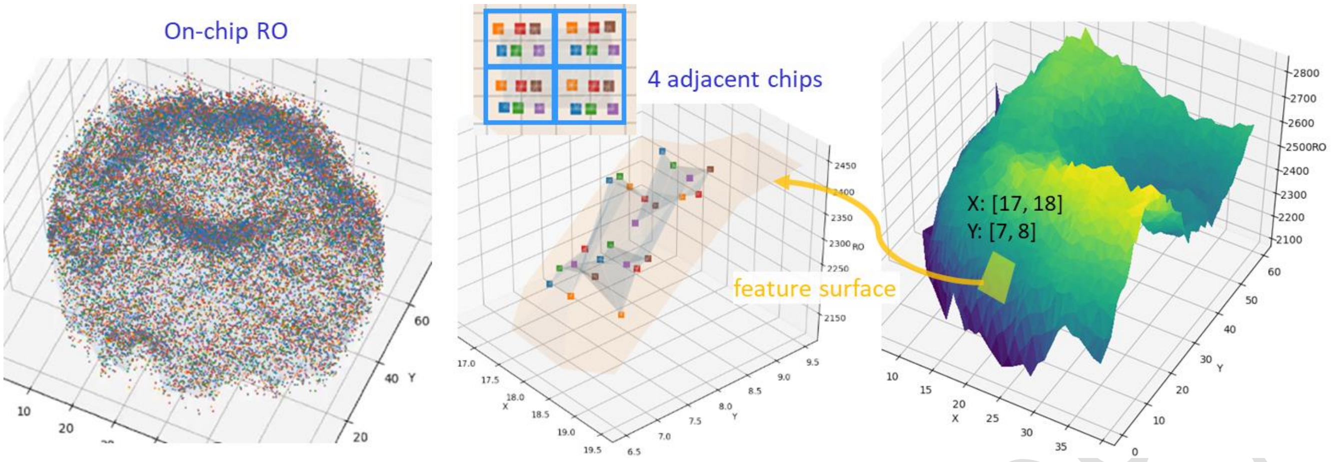


Fig. 7-7 RO Uniformity and and OCV Analysis

The non-randomness mentioned above can distort traditional timing signoff tools, as shown in Fig. 7-8. Local voltage gradient effects must be controlled to prevent excessive resource usage when handling traditional global PVT corners. Unless wafer foundries improve process uniformity or chip designers provide costly local voltage sources (LDO arrays), local voltage gradients cannot be fully eliminated. Combining system-level binning with voltage compensation, it's recommended to relax defense boundaries (e.g., 1.65 sigma) to avoid overly conservative margins that could impact energy efficiency and competitiveness.

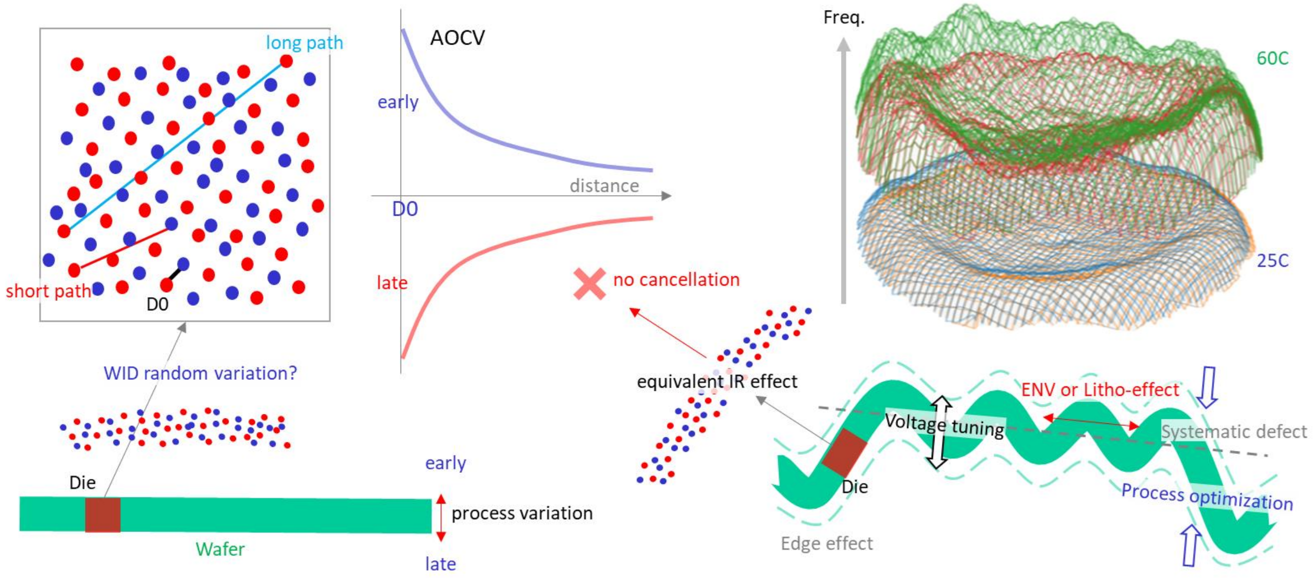


Fig. 7-8 OCV Derating and Cancellation

7.5. Post-Silicon Analysis and Optimization

By reconstructing wafer-level physical parameter surfaces from CP's XY coordinates, as shown in Fig. 7-9, three key offsets and non-random variations are typically observed:

Coarse-grained Harmonics: Wafer data (e.g., SIDD) exhibit donut-shaped patterns of low-frequency sine waves, with uneven deposition tilt. Certain polishing steps create Polish Patterns, amplifying non-uniformity and revealing system-level defects from specific process steps, as shown in Fig. 7-10.

Test Environment: Analog components like V/T Sensors, designed to mitigate PVT variations, highlight deviations from test environment factors, such as mismatches in Load Board or Probe Card resistance, creating Test Site Patterns. However, PVT variations often remain due to significant Leakage Uniformity gradients. RO measurements capture system-level unevenness, compounded with low-frequency oscillations, revealing non-randomness common in production testing, as shown in Fig. 7-11.

Litho-effects: Analyzing physical location and data reveals that lithographic effects or test board resistance mismatches significantly affect electrical characteristics and delays. Mask exposure boundaries often show regular peaks in SIDD, corresponding to regular valleys in cell delays.

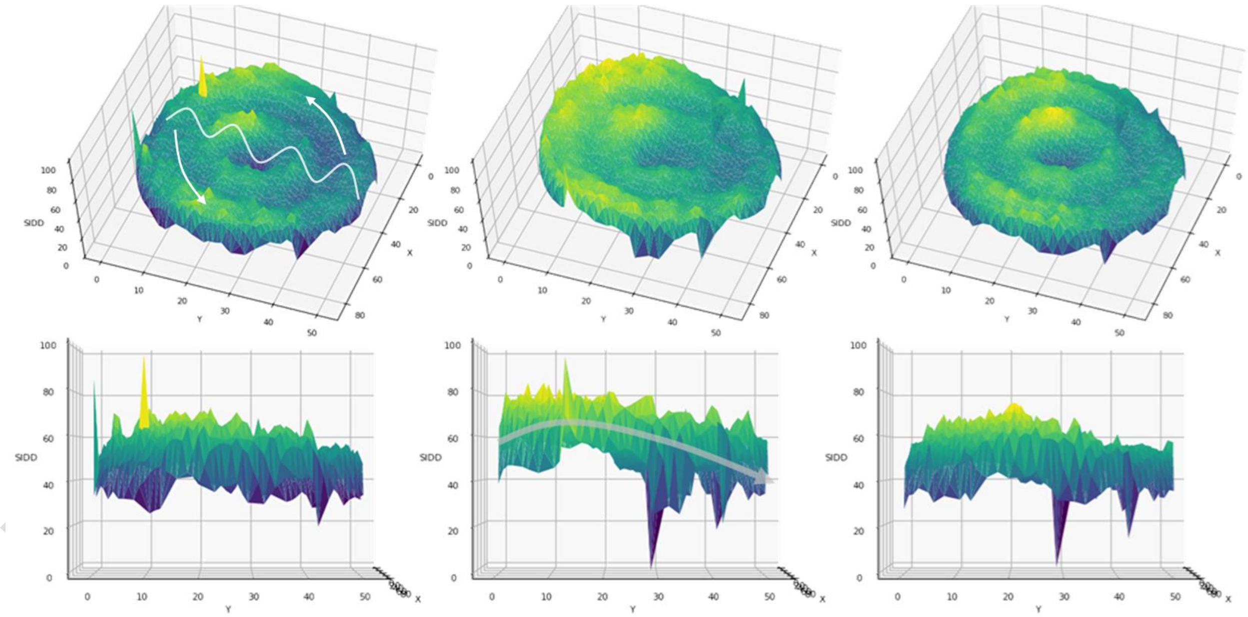


Fig. 7-9 Wafer-level Feature Surface

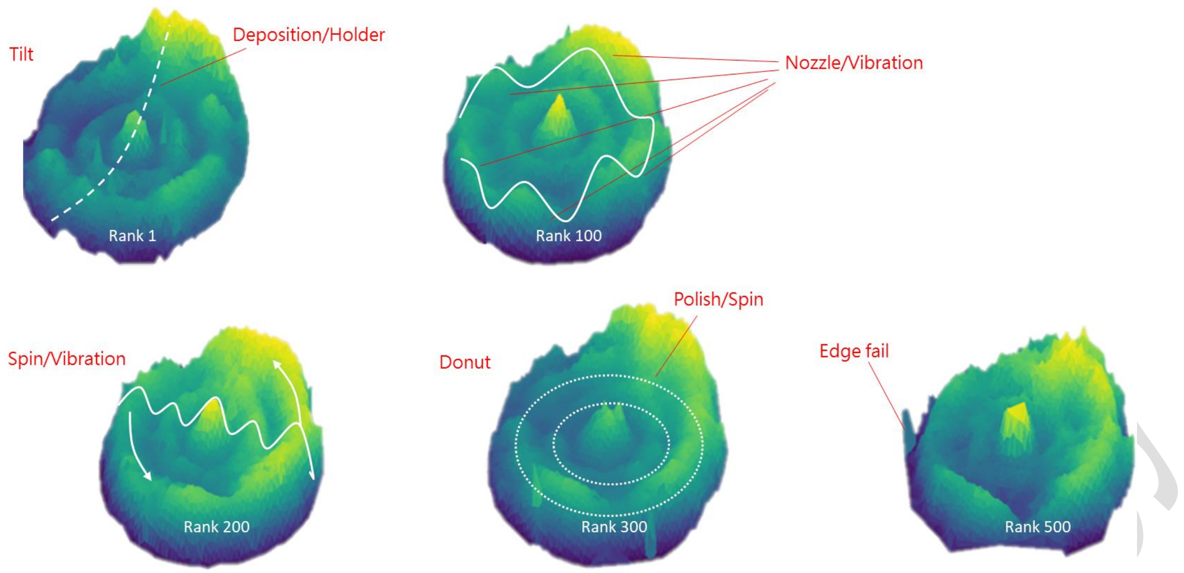


Fig. 7-10 Concentric Ripple Patterns

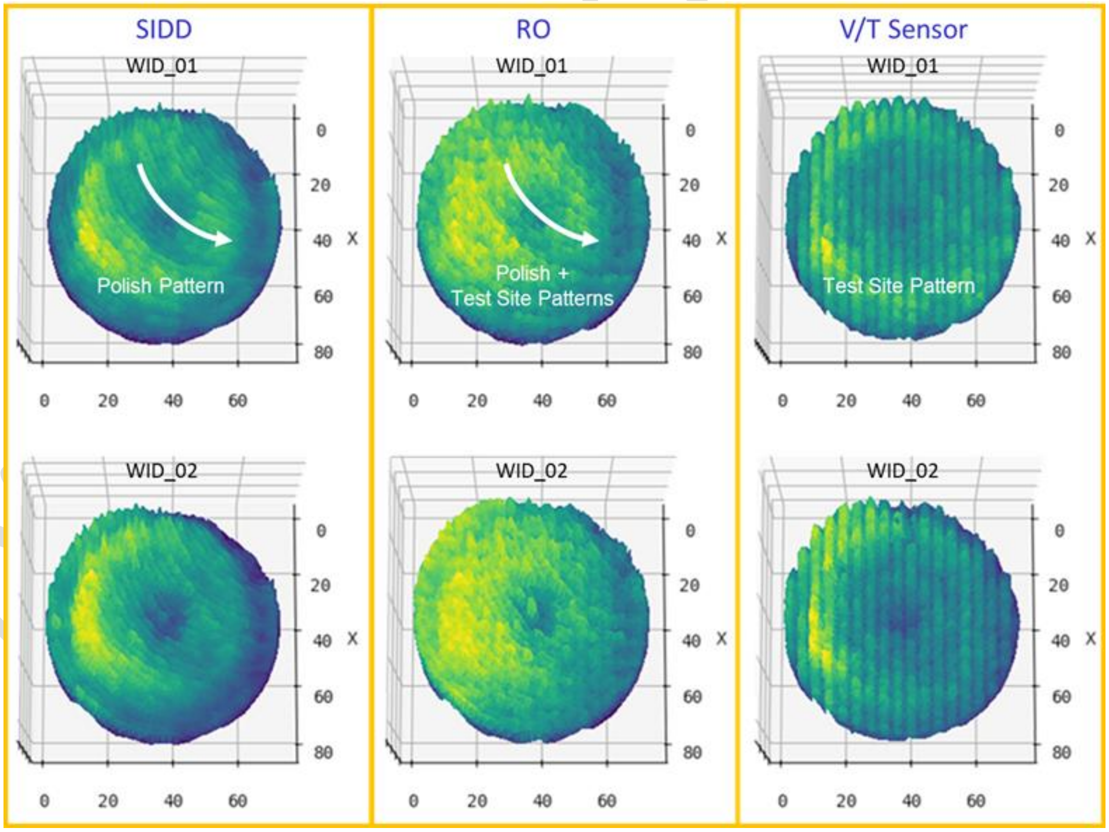


Fig. 7-11 Polish and Test Site Patterns

Chapter 8. Chip Performance Rating Strategy and Optimization (Binning-PG™)

8.1. Impact of Binning Strategy on Productivity

Binning strategy in semiconductors classifies chips based on performance characteristics to ensure stability and reliability across varying conditions. Its impact on productivity includes:

Improved Yield and Test Efficiency: Accurate binning enhances yield, reduces defective products, and minimizes test time and costs.

Optimized Production Efficiency and Stability: Binning helps tailor the production process to chip performance needs, improving efficiency and reducing disruptions.

Enhanced Product Quality and Competitiveness: Proper classification ensures chips meet performance standards, boosting quality, consistency, and competitiveness. It also supports pricing strategies by setting performance-based price bins, maximizing revenue.

In the case of multi-core machines (such as BTC mining machines [2]), binning categorizes machines based on computing power. After classifying the machines, their total computing power (Tera-hash rate, TH) is tested. As shipment volume grows, the cumulative computing power (cumulate TH) increases. This metric helps assess binning strategies and identify the most effective method for maximizing computing power.

As shown in Fig. 8-1, comparing cumulative computing power (cumulate TH) from different binning methods helps identify the optimal strategy for balanced shipments, minimizing performance fluctuations, and maximizing cumulative power to meet demand and boost shipment efficiency. These strategies enhance overall product performance, energy efficiency, and productivity.

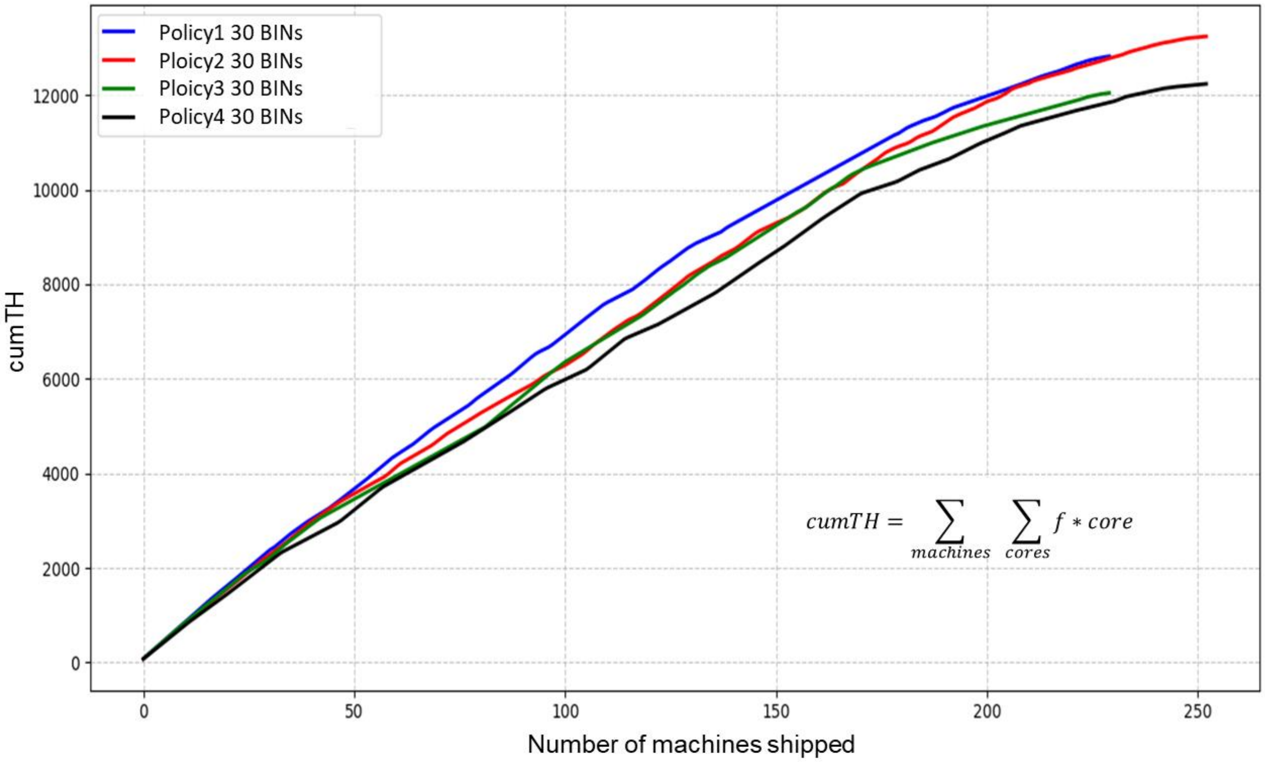


Fig. 8-1 Multi-core Machine Shipment and Cumulative Computing Power Evaluation

8.2. Chip Characteristics Analysis and Challenges

Traditional binning methods typically use a two-dimensional pipeline strategy based on chip performance (e.g., Fmax) and SIDD leakage current, as shown in Fig. 8-2. For instance, SIDD is used for coarse segmentation, followed by fine segmentation based on functional patterns' performance. This approach heavily relies on engineers' time for coding and trial-and-error testing to define segmentation conditions and boundaries. However, it cannot ensure consistency in voltage or temperature sensitivity across chips. As shown on the right, even within the same group, their performance in other dimensional indicators may vary significantly.

In reality, factors influencing chip performance often extend beyond two or three dimensions, as shown in Fig. 8-3. The performance surface of the same chip under different voltage and temperature conditions is typically non-linear. For example, under a fixed voltage, chips at certain frequencies may show similar performance (as seen in the pink cutline of the V-F curve). However, temperature changes may cause dramatic, cliff-like performance shifts.

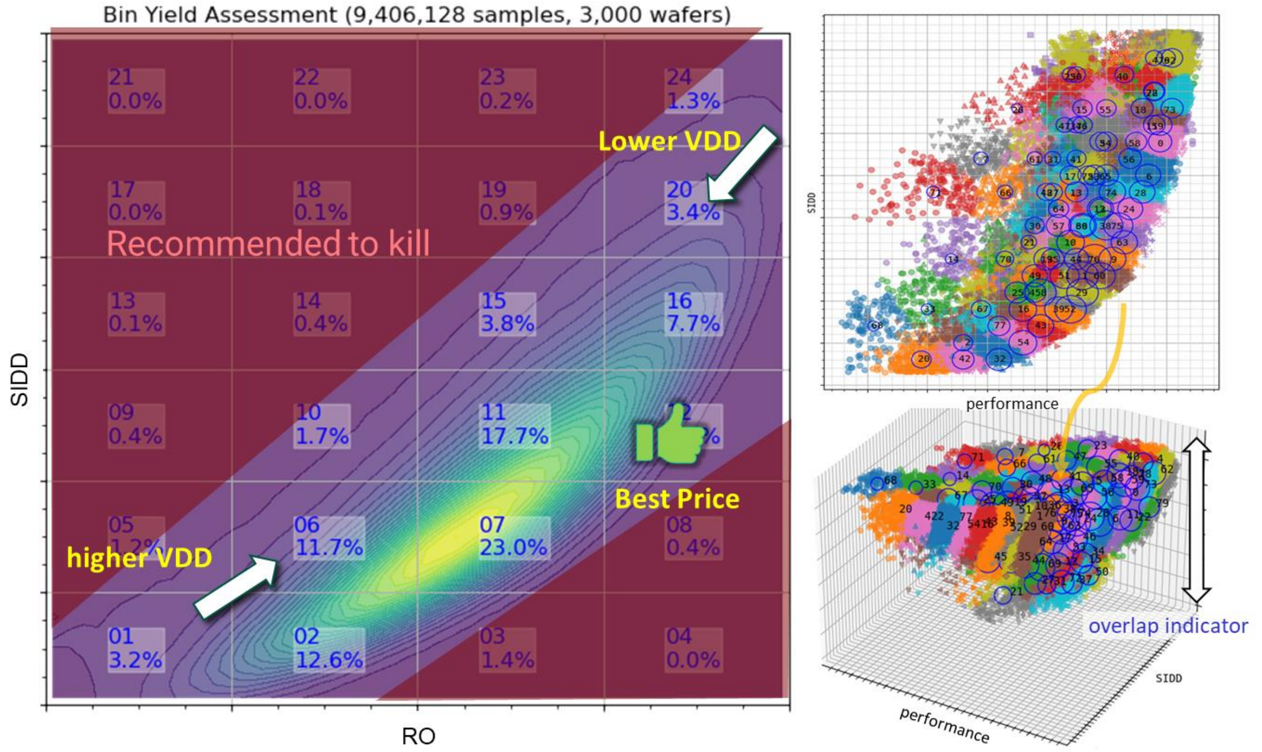


Fig. 8-2 Traditional 2D Pipeline Binning and Potential Issues

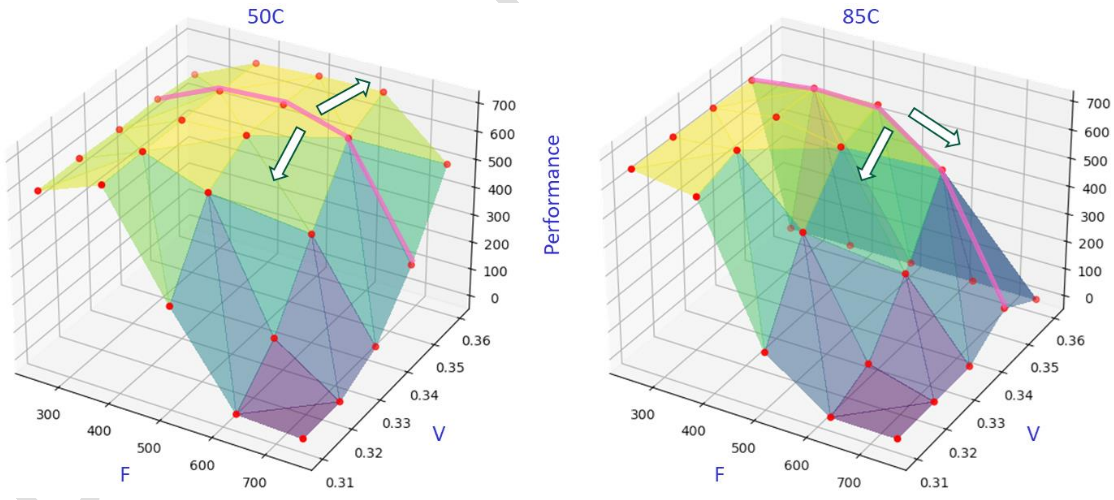


Fig. 8-3 V-T Sweep and Performance Analysis

To address the challenge of high-dimensional data in chip performance clustering, K-means clustering is often used to group data by minimizing variance within each group. However, this method has some shortcomings, particularly at the machine level:

Ignoring Feature Importance: Traditional algorithms like K-means treat all dimensions equally, lacking the ability to identify differences in feature importance. This requires manual intervention, such as feature augmentation or weight adjustments, which can lead to increased workload and reduced accuracy. As shown in Fig. 8-4, adjusting weights for six-dimensional data allows K-means to better account for the impact of SIDD weight changes in chip features.

Discreteness of Multi-dimensional Features: Multi-dimensional features in chip data are often discrete and may form distinct high-dimensional manifolds, making it difficult to differentiate between them. This is especially true when dealing with subtle variations or anomalies in chip characteristics, which K-means may fail to detect, particularly when handling PVT sensitivity differences.

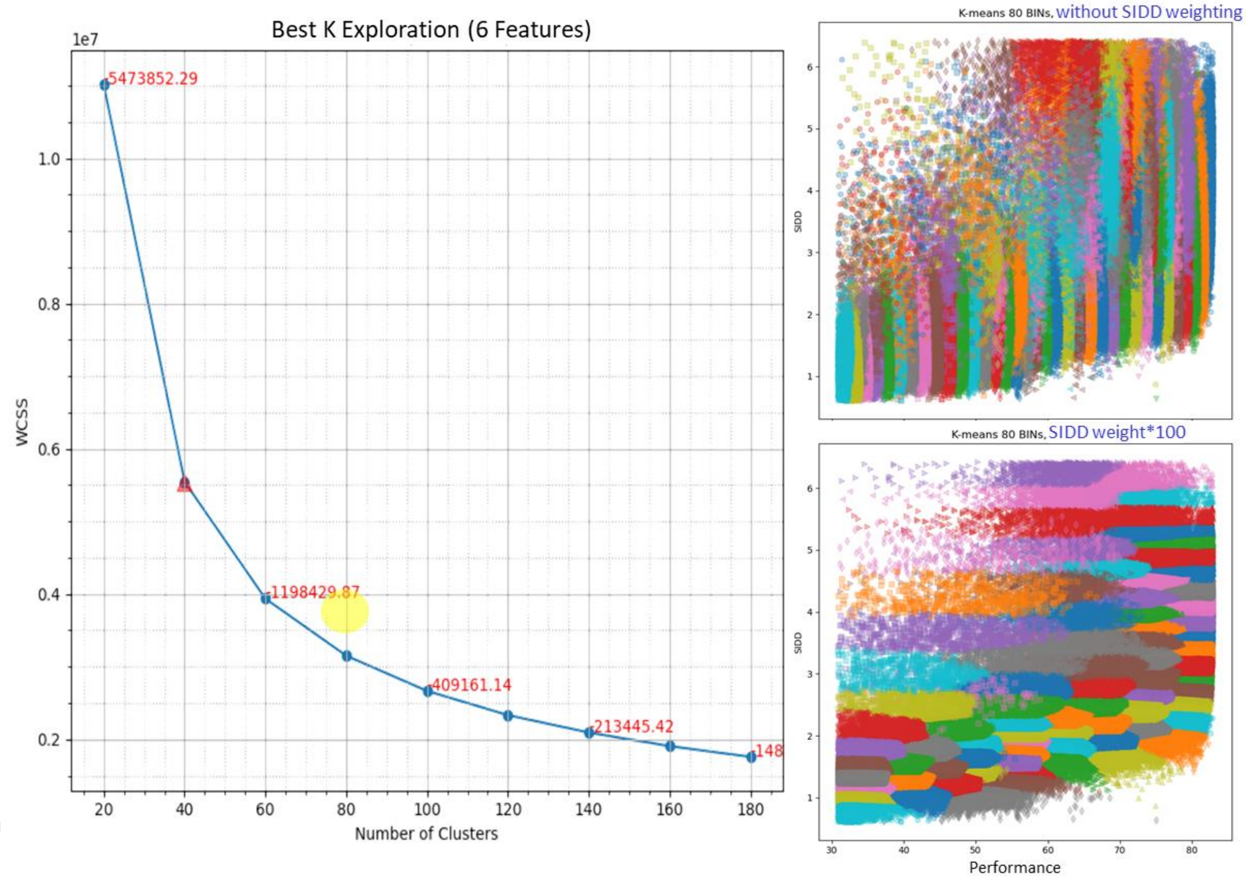


Fig. 8-4 Best-K Exploration

Traditional methods typically use a fixed voltage configuration of frequency and performance as features for K-means clustering (e.g., the pink dashed line in Fig. 8-3). However, this approach struggles to differentiate at the machine level, especially with high-dimensional discrete features. To improve this, we can use discrete grid points to build interpolation surfaces, as shown in Fig. 8-5. This enhances the resolution of grid points, improving the continuity and representation of the feature space. Additionally, adjusting the weight of the SIDD feature increases its impact during clustering, effectively addressing the challenge of distinguishing high-dimensional discrete features.

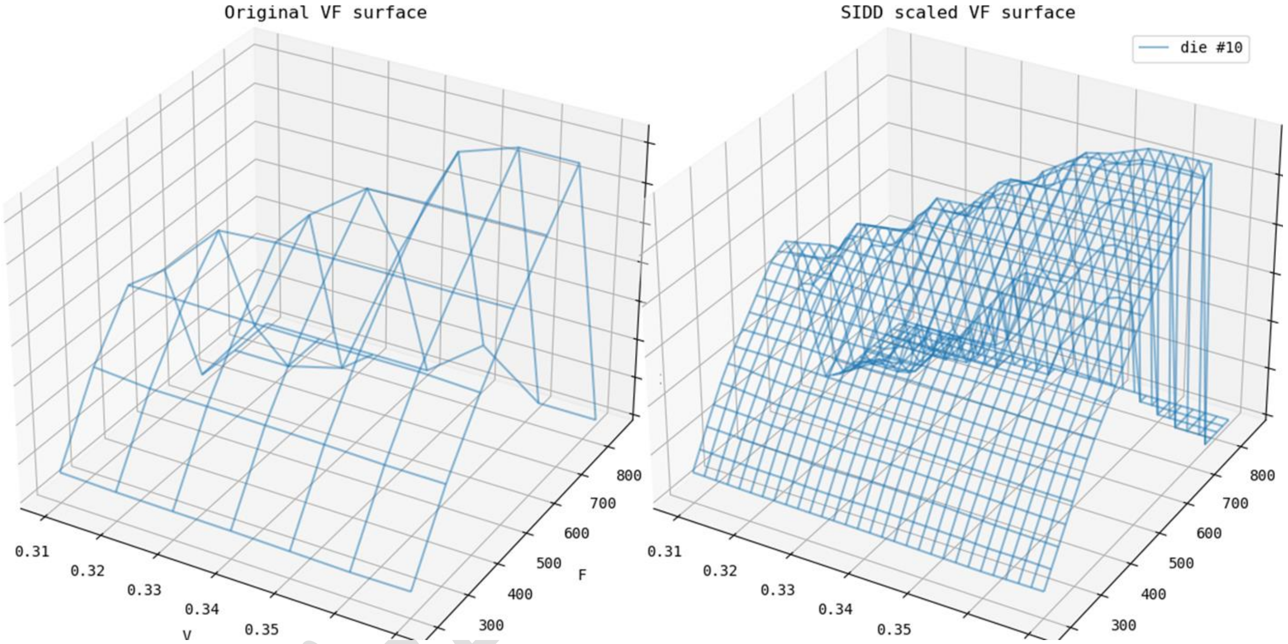


Fig. 8-5 V-F Grid Interpolation

In practical applications, feature augmentation techniques (e.g., interpolation) can expand the original 6-dimensional CP test feature space (including RO, SIDD, and 4 performance indicators) into dozens of dimensions for clustering, ensuring accurate chip performance evaluation. This enhancement significantly improves clustering and provides a more reliable basis for machine-level performance differentiation.

8.3. Binning Policy Generation (Binning-PG™)

Binning policy generation is a systematic process for categorizing chips based on their performance characteristics to ensure consistent performance under various operating conditions. The key steps include:

Data Collection and Preprocessing: Collect multi-dimensional performance data (e.g., frequency, power, SIDD leakage) from chips, clean and standardize the data to ensure consistency.

Feature Extraction and Clustering: Analyze key features and use clustering algorithms (e.g., K-means, DBSCAN) to group chips by performance, identifying performance patterns and binning boundaries.

Strategy Adjustment and Optimization: Optimize the initial clustering results based on production needs, adjusting bin ranges to meet performance targets.

Strategy Validation and Implementation: Conduct small-scale tests or simulations, adjust based on results, and implement the strategy in large-scale production to ensure effectiveness and stability.

In the early stages, unsupervised learning using initial characteristics like WAT and CP/FT is employed to group chips with similar traits. This may involve human-in-the-loop (HITL) active learning to refine clustering. As production data accumulates, initial clustering results evolve into ground truth, supporting supervised learning to further optimize the binning strategy.

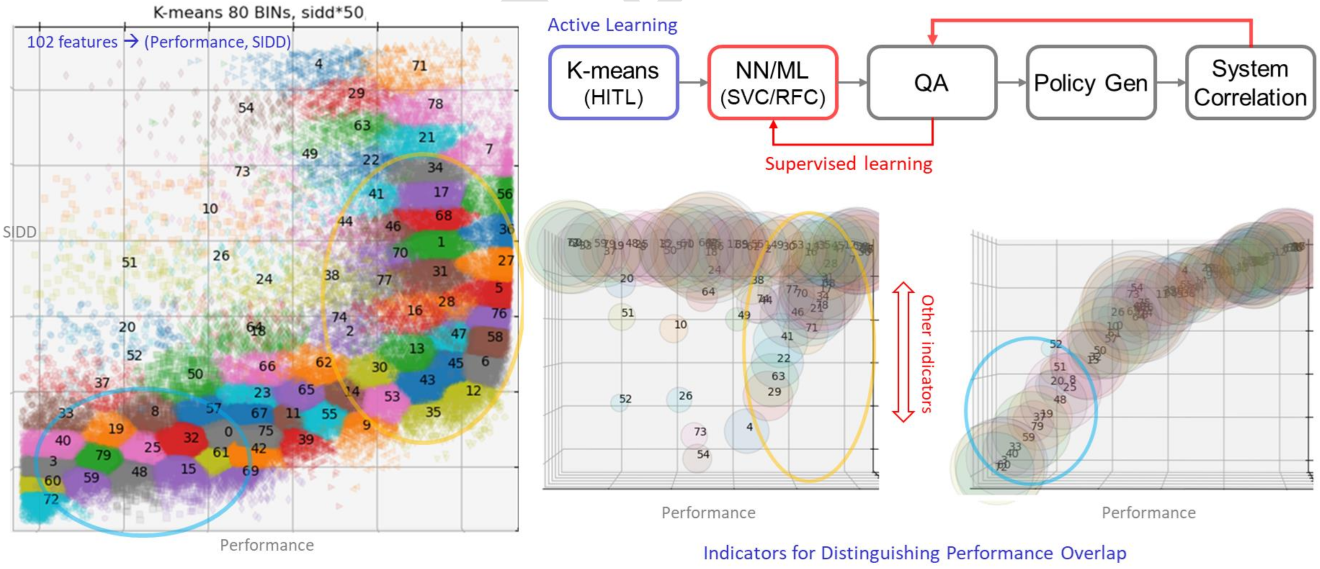


Fig. 8-6 Binning Policy Generation

8.4. Automated Policy Generation and Optimization

The goal of binning strategy automation is to improve chip classification efficiency through precise data analysis and algorithm application, enhancing productivity and product quality. Key objectives include:

Performance Evaluation and Qualification Criteria: Quantify chip performance with composite metrics (e.g., SIDD, ∑Fmax, cumHash:=∑Fmax * pass_core, SIDD-std, cumHash-std) to ensure binning reflects chip behavior under various conditions and establish clear qualification standards, as shown in Fig. 8-7.

Clustering Algorithm and Optimization: Use hybrid K-means clustering to automate grouping, classifying chips by performance similarity, and improving intra-group homogeneity. This enhances binning accuracy, reduces manual intervention, and improves process automation and stability.

Effective Workflow Methodology: Design CP/FT testing processes based on successful projects, providing consistent binning guidance for future projects, enabling rapid adaptation of optimal strategies and long-term production efficiency improvements.

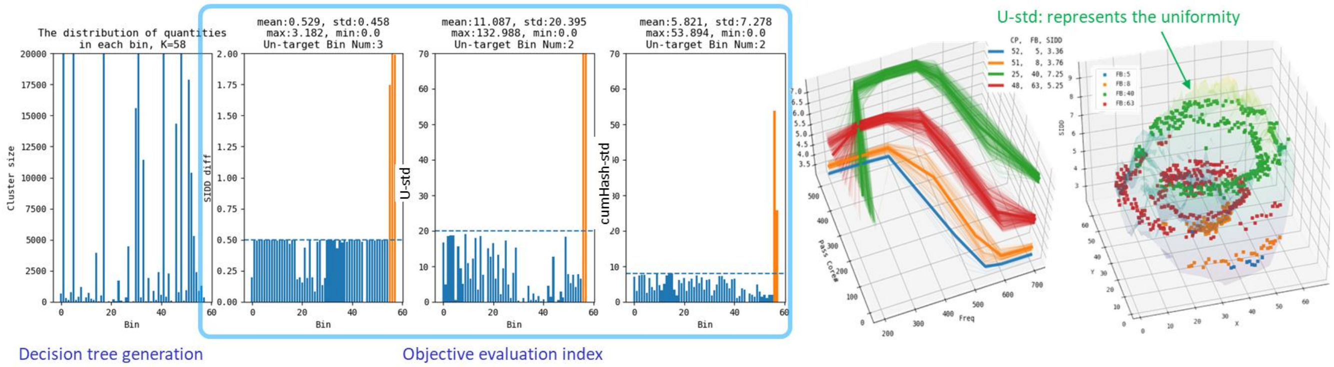


Fig. 8-7 Key indicators for Chip Classification

Fig. 8-8 illustrates a hybrid strategy combining K-means clustering with Human-in-the-Loop (HITL) to enhance clustering accuracy and efficiency. HITL techniques, including feature augmentation (e.g., weight adjustment and interpolation surface construction), generate fine-grained clustering results (Soft Bin) that serve as the benchmark (Ground Truth) for active learning, allowing flexible adjustment of clustering standards in the early stages and setting the foundation for model training.

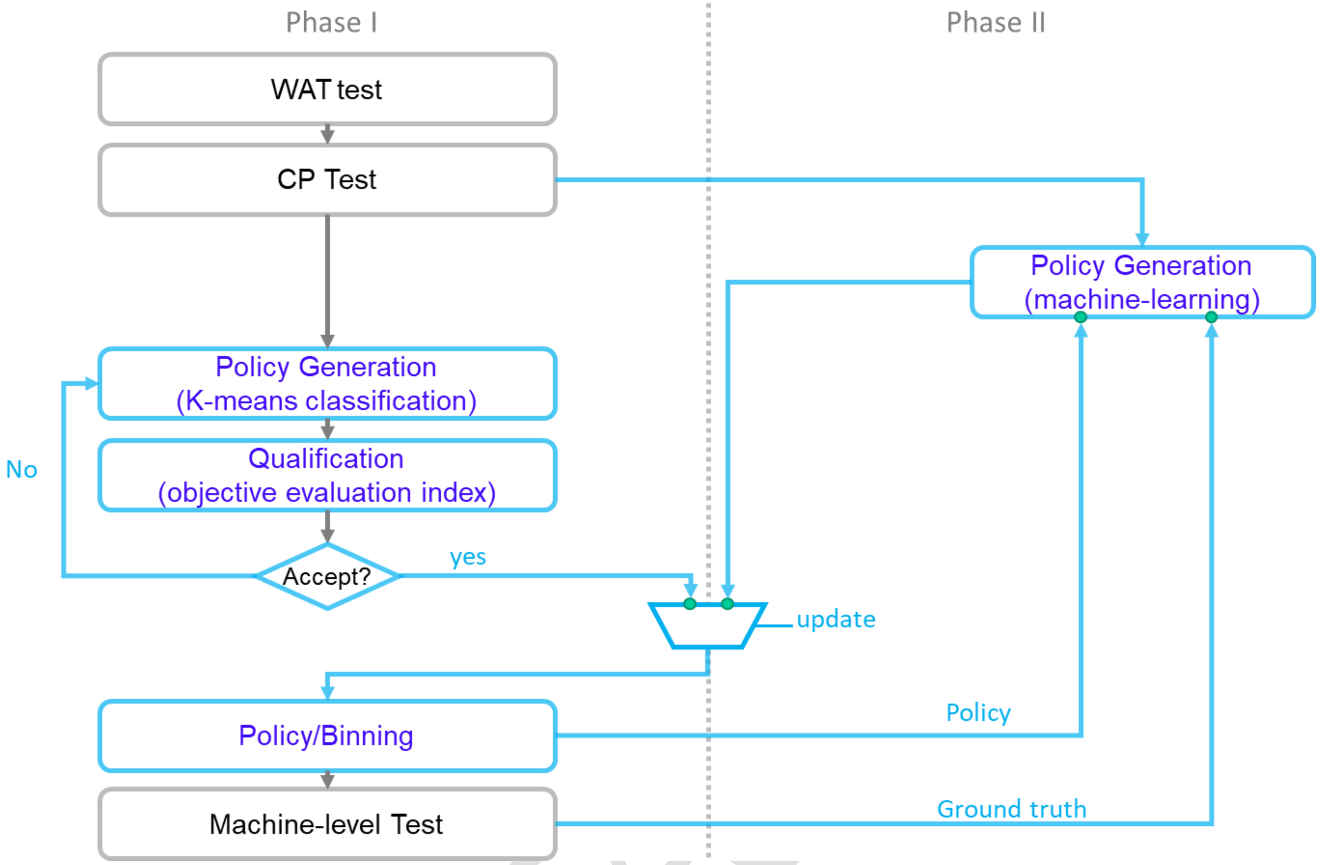


Fig. 8-8 Binning Policy Optimization Flow

Next, Ground Truth data is used with supervised learning techniques (such as SVM, random forests, and neural networks) to train classifiers and create precise binning strategies. This optimizes clustering, reduces bin count, improves efficiency and market value, and balances accuracy with production costs. The approach automates and refines strategy design, accelerating iteration cycles and boosting chip manufacturing flexibility and efficiency.

8.5. On-chip Self-binning Application

On-chip Self-binning (OCSB) is an innovative technology that integrates self-testing and performance evaluation directly within the chip, enabling automatic classification during production. It enhances binning efficiency, reduces testing costs, and minimizes reliance on external equipment. Key features include:

Built-in Self-binning: The chip integrates a self-test module to automatically measure performance parameters (e.g., RO frequency, effective voltage, dynamic slack) under various conditions, providing rapid and accurate data.

Reduced Testing Costs: By utilizing self-test data, the chip analyzes performance and slack in real-time and automatically determines binning, reducing reliance on external tests and human intervention, thus improving efficiency and automation.

Real-time Performance Evaluation: Using real-time monitoring circuits and AI inference, the chip adjusts binning dynamically based on the operating environment, enhancing reliability and market competitiveness.

OCSB uses neural networks to infer performance within the chip's memory and MAC units, enabling automated binning. The neural network is trained pre-production to understand the relationship between performance and binning, storing results internally. This technology reduces testing time, minimizes external equipment dependency, and allows flexible adjustments, boosting efficiency, product quality, and cost-effectiveness, as shown in Fig. 8 9.

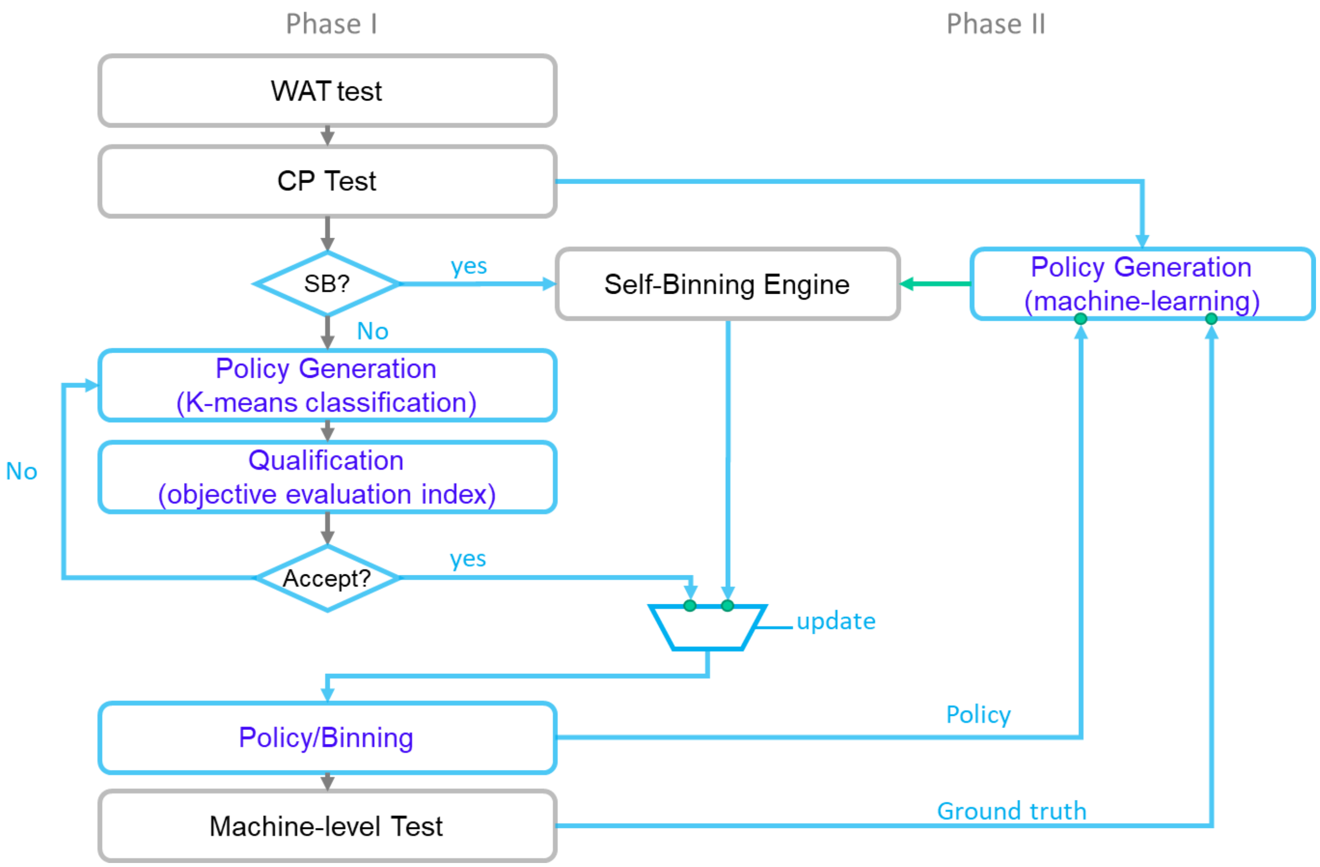


Fig. 8-9 On-chip Self-binning (OCSB)

Part III: DTCO.GenAI™ - Generative AI-Driven Chip Design Innovation

This section explores the applications of Generative AI in semiconductor and chip development, covering GAN, Diffusion Models, high-σ Monte Carlo simulation acceleration, and WAT super-resolution technology [3][4][5].

Chapter 9. Generative AI and DTCO Integration (DTCO.GenAI™)

9.1. Limitations of Traditional Modeling Methods

Traditional models often assume Gaussian distributions, ignoring that real chip data typically follows skewed-normal or log-normal distributions. They also overlook interrelationships between vectors in high-dimensional space. As shown in Fig. 9-1, while individual features may follow population distributions, their combinations in high-dimensional space may lose correlation.

Real-world data concentrate on low-dimensional manifolds within high-dimensional space. Assuming independence between features, high-dimensional distributions may become unrealistic as dimensionality increases. The key to addressing this is considering feature dependencies and using more suitable techniques for high-dimensional modeling.

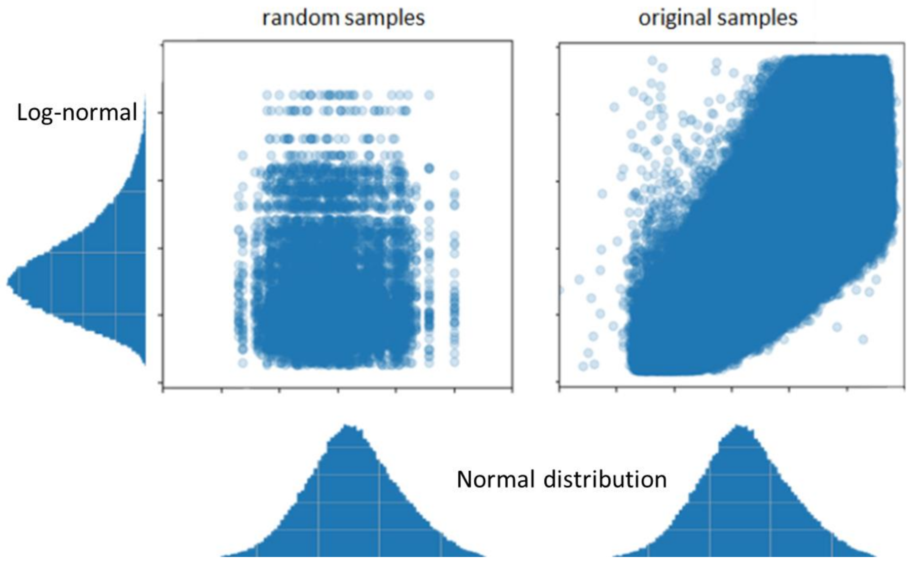


Fig. 9-1 Interdependencies of Physical Quantities

The assumption of independent feature distributions often fails to accurately reflect the complexities and interrelationships in the chip manufacturing process. To more effectively simulate chip behavior, it’s crucial to consider non-Gaussian, skewed, or log-normal distributions, while fully accounting for interdependencies in high-dimensional space.

The numerous process parameters in wafer fabrication create complex relationships with wafer or chip-level test data, making traditional modeling methods insufficient. As shown in Fig. 9-2, even with a complete understanding of the feature distributions and relationships at the chip level, the lack of coordinate information results in the loss of true process uniformity at the wafer level. Consequently, the distribution of high-dimensional feature vectors lacks realism, leading to significant discrepancies between production and simulation data.

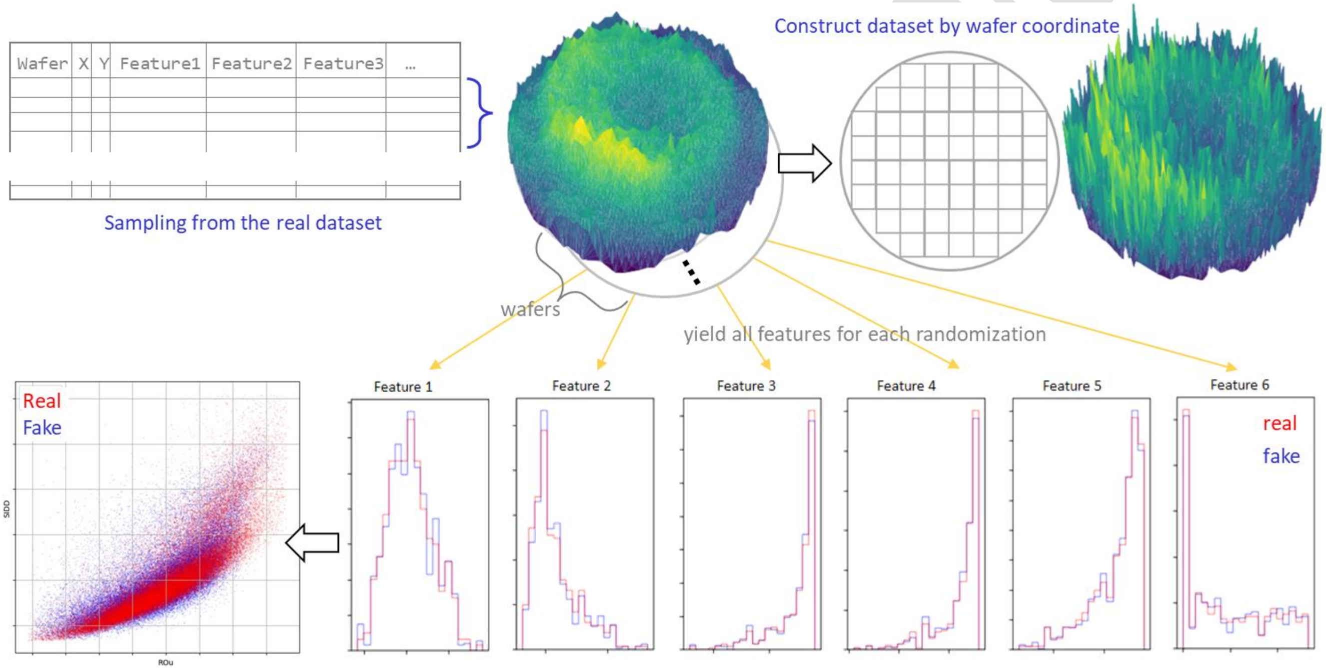


Fig. 9-2 Dissipation of Feature Spatial Continuity

This discrepancy hinders our understanding and simulation accuracy of the chip manufacturing process. To address this, advanced modeling methods are needed to account for the complexity of process parameters and wafer coordinates, enabling more accurate simulations and reliable production data.

9.2. Following the Trail: Multivariate Normal Distribution

Incorporating feature correlations (e.g., the covariance matrix in GMM) effectively captures dependencies between high-dimensional features, leading to a more realistic data distribution. As dimensions increase and more feature correlations are modeled, the generated distribution aligns more closely with real data.

For example, a simple correlation analysis report between the process and chip performance may be based on limited early-stage data, such as 420 chip test results from six wafers, including TT and skew wafer data.

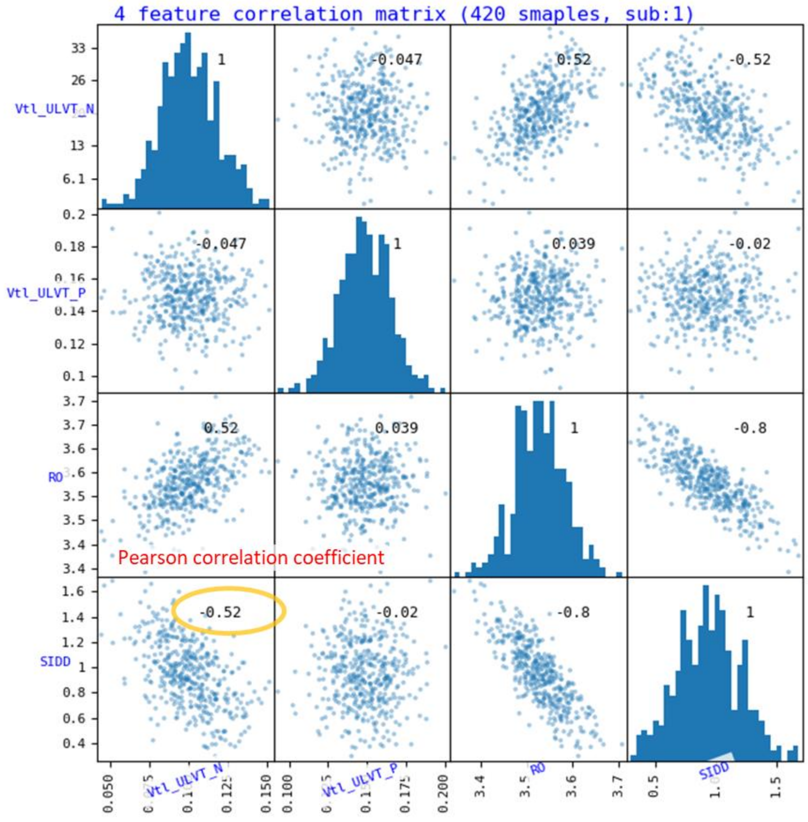


Fig. 9-3 Multivariate Feature Correlation

Fig. 9-3 shows a typical correlation matrix mapping WAT site ID to CP data, with numbers representing Pearson correlation coefficients. Using this matrix, we can apply Multivariate Normal Distribution (MVN) to quickly create an approximate high-dimensional model, generating dense data to enhance trend analysis, boundary profiling, and decision confidence. Fig. 9-4 demonstrates 10,000 data points randomly generated with MVN based on this correlation matrix.

Example 9-1 Multivariate Normal Distribution

```
import numpy as np
import pandas as pd

μ = [0.1, 0.15, 3.53, 0.95] # features mean
σ = np.array([0.11, 0.1, 0.37, 1.4])/6 # min-max 6 sigma (Gaussian)
σ1,σ2,σ3,σ4 = σ

# Pearson coefficient based on the scatter metrix
ρ = [np.array([-0.047, 0.52,  -0.52]),
     np.array([ 0.039, -0.02]),
     np.array([-0.8])]

(ρ12,ρ13,ρ14), (ρ23,ρ24), (ρ34,) = ρ

# covariance matrix
Σ = [[σ1**2,     ρ12*σ1*σ2, ρ13*σ1*σ3, ρ14*σ1*σ4],
     [ρ12*σ1*σ2, σ2**2,     ρ23*σ2*σ3, ρ24*σ2*σ4],
     [ρ13*σ1*σ3, ρ23*σ2*σ3, σ3**2,     ρ34*σ3*σ4],
     [ρ14*σ1*σ4, ρ24*σ2*σ4, ρ34*σ3*σ4, σ4**2]]

data = np.random.multivariate_normal(μ, Σ, 10000) # generate 10K samples

# convert to Dataframe
dt = pd.DataFrame(data, columns=['Vtl_ULVT_N', 'Vtl_ULVT_P', 'RO', 'SIDD'])
```

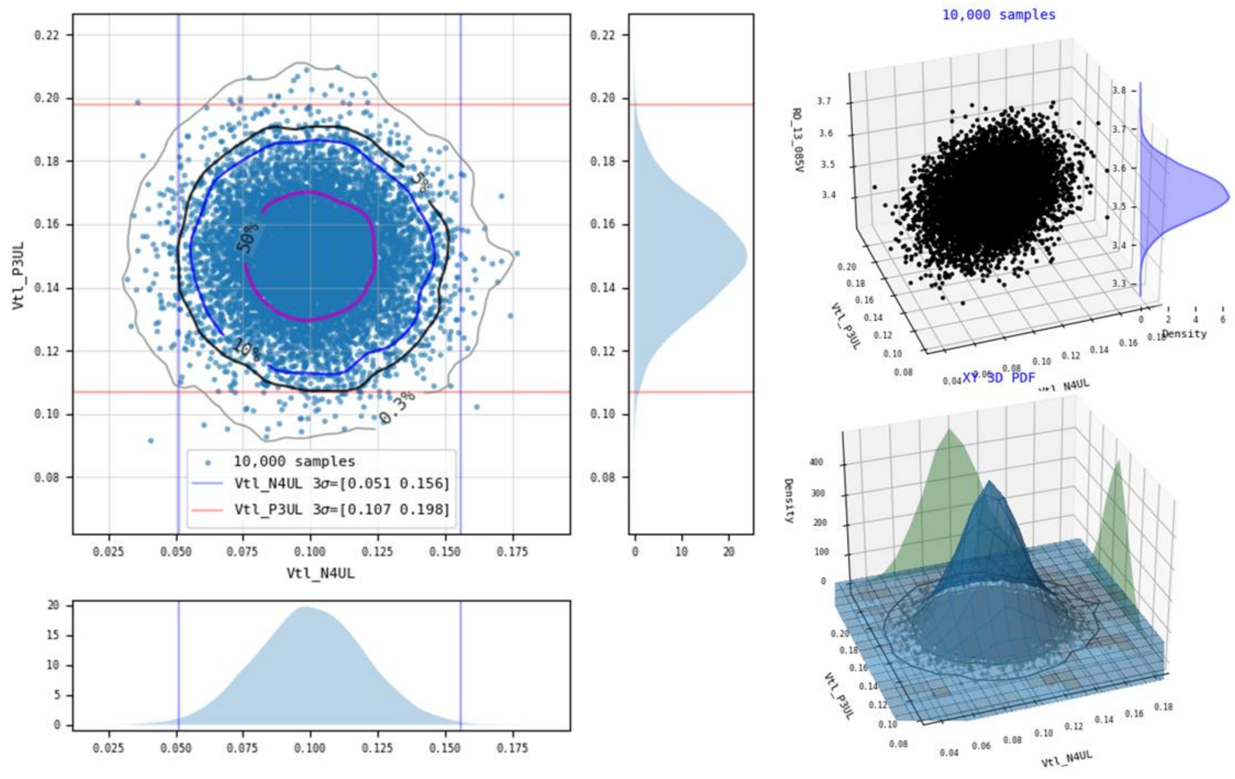


Fig. 9-4 Feature Distribution and CDF Contours

9.3. Virtual Silicon Data in DTCO (DTCO.VS)

Design Technology Co-Optimization (DTCO) is vital at advanced nodes, but designers and EDA developers often lack reliable process and test data. In such cases, virtual silicon data provides high-quality, realistic data while preserving confidentiality. By analyzing chip characteristics like leakage current and performance across frequencies, designers can pinpoint defects and optimization opportunities. The generation of large volumes of silicon data enhances chip performance evaluation, boosts product competitiveness, and optimizes wafer performance.

Fig. 9-5 shows the use of virtual chip data, which helps increase trust levels and identify the trade-off between yield and design margins, guiding future design and capacity optimization.

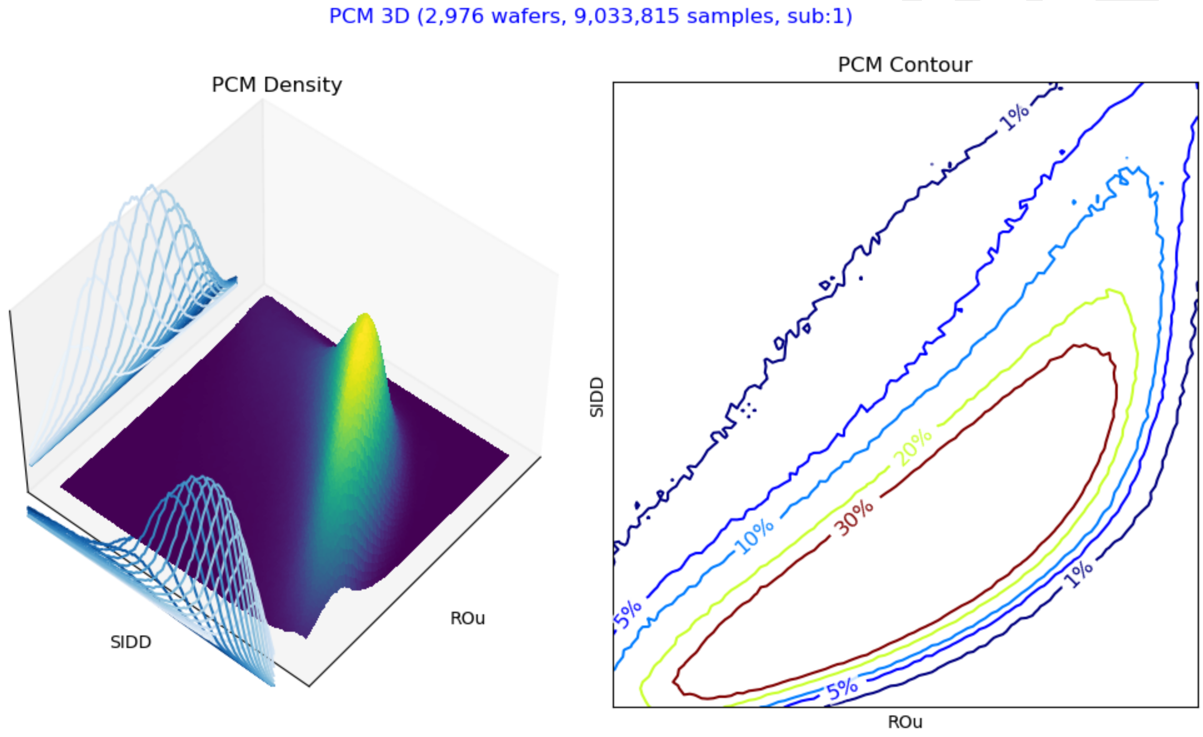


Fig. 9-5 CDF Contour

Fig. 9-6 illustrates a typical two-dimensional binning strategy for mass production. By leveraging virtual chip data that closely mirrors real-world scenarios, we can accurately predict performance grading and compensation, boosting optimization efficiency, increasing capacity, and reducing costs, while ensuring product quality stability, reliability, and competitiveness.

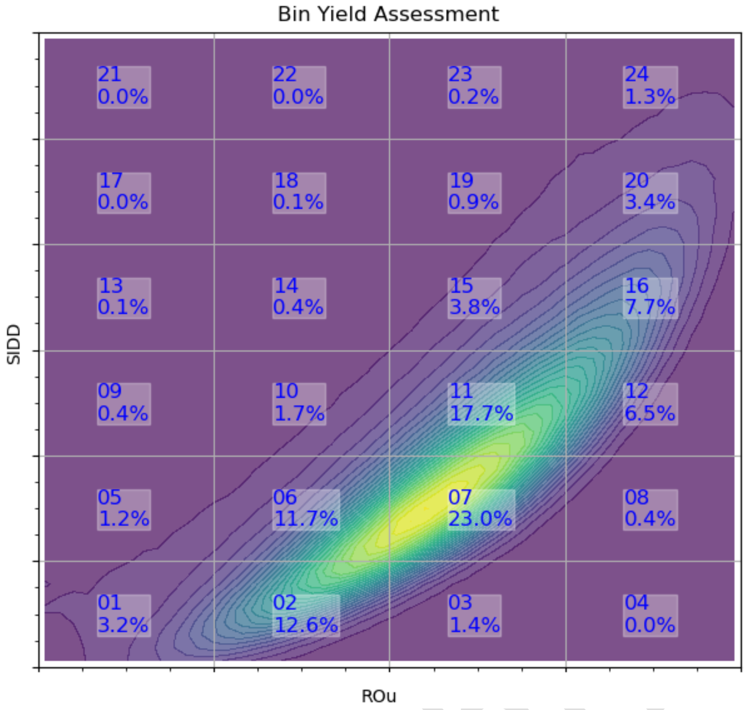


Fig. 9-6 Speed and SIDD Binning Strategy

Fig. 9-7 shows the application of cross-dimensional feature contours, where WAT N/P process parameters form the XY axis base, and CP performance and power data serve as the Z axis. This method acts like a literary translator, bridging terminology gaps between process and chip design. With virtual chip data, engineers can balance design goals and optimize processes and performance. By analyzing cross-dimensional data and correlations, such as using wafer-level parameters, we can offer process adjustments to improve chip performance while maintaining physical characteristics (e.g., SIDD and RO frequency).

Additionally, the model's cross-dimensional correlations provide valuable insights for DTCO. As shown in Fig. 9-8, correlations between wafer-level process parameters (e.g., N-MOS and P-MOS threshold voltage Vth) and chip-level performance (Fmax, SIDD) highlight significant relationships. Variability in N/P Vtl shows negative slope coefficients and intercepts. This approach functions like a dynamic oral translator, facilitating cross-dimensional correlation analysis and data interaction (cross-probing), allowing us to effectively balance energy efficiency targets with process optimization.

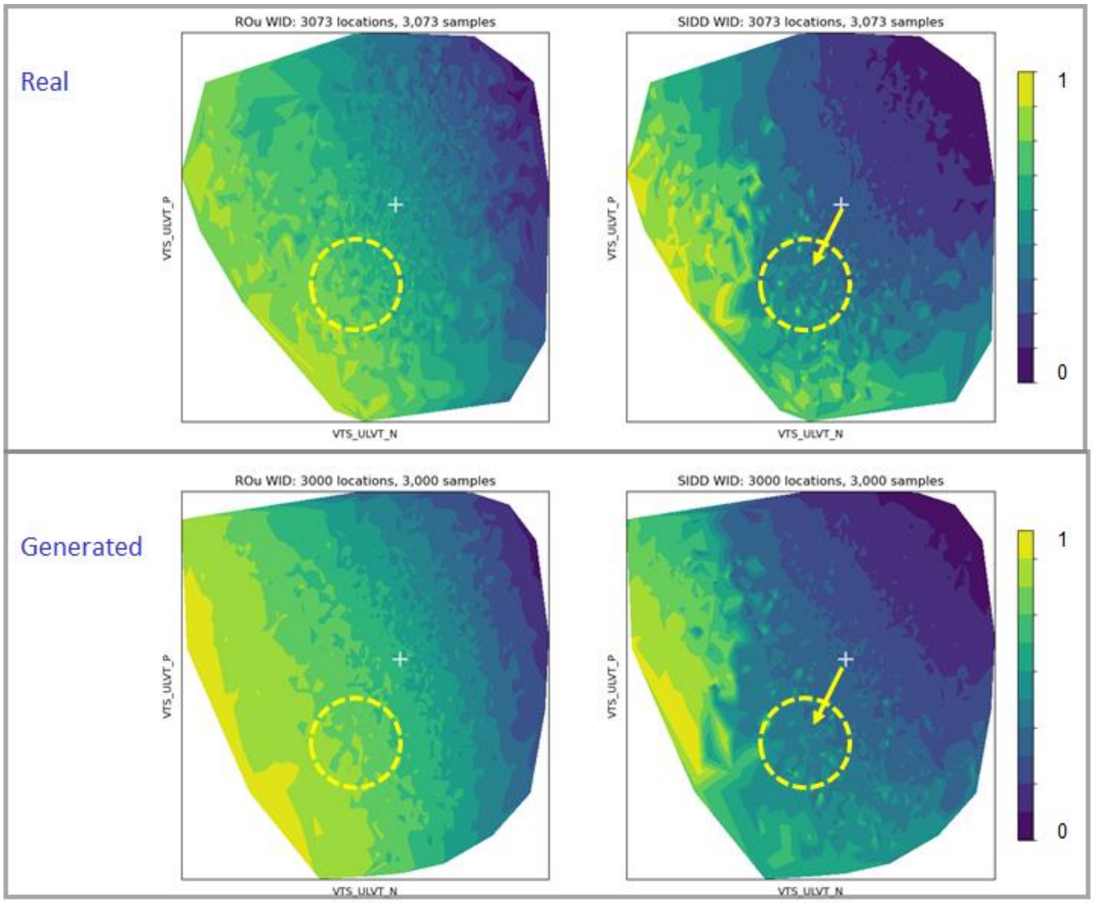


Fig. 9-7 Process Recipe Window Optimization

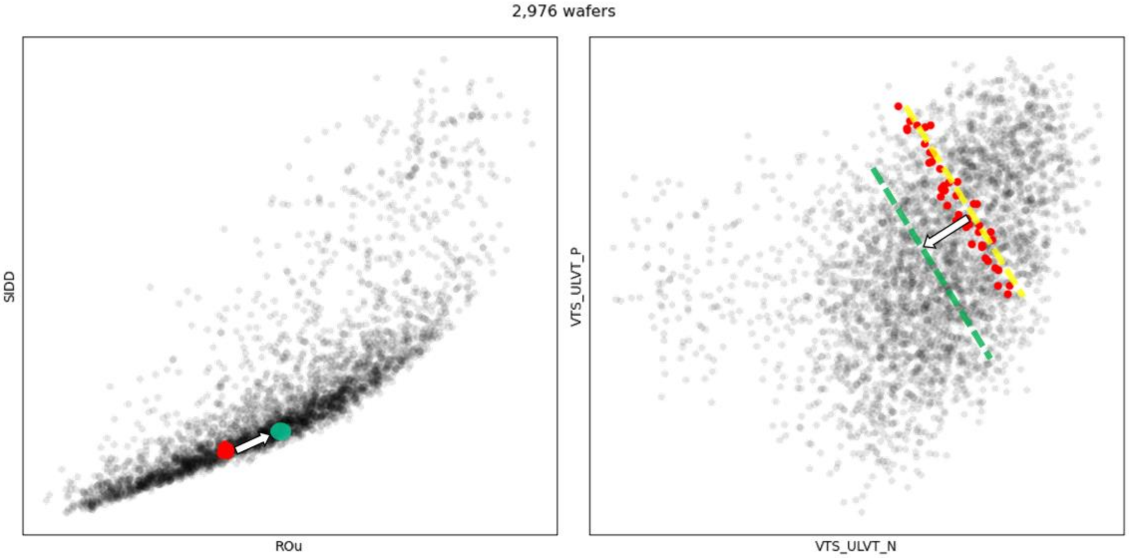


Fig. 9-8 Multi-Feature Cross-Probing

Chapter 10. Virtual Silicon Data Generation Technology (DTCO.VS)

With the rapid growth of silicon data, machine learning plays a vital role in enhancing chip performance. Analyzing the distribution of key wafer characteristics helps predict process parameters and chip behavior accurately. However, real silicon data acquisition and sharing face challenges due to commercial confidentiality. This chapter presents a high-quality silicon data generation method, combining Chip Performance (CP) and Wafer Acceptance Test (WAT) data, validated through various metrics to create a comprehensive dataset that supports Design Technology Co-Optimization (DTCO).

Generative Artificial Intelligence (Generative AI) is used to learn chip and wafer test features, addressing the challenge of real silicon data collection. By leveraging generative models to capture physical characteristics, designers can simulate mass production testing, develop binning strategies, and optimize designs while overcoming data sharing challenges.

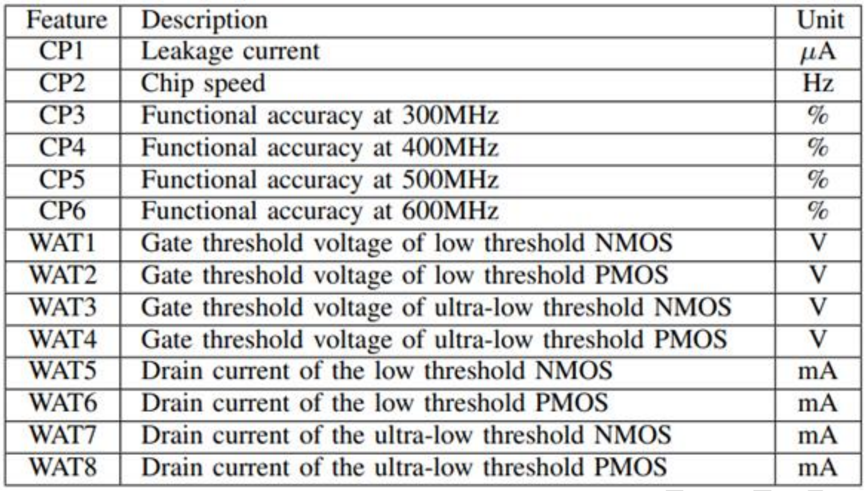
This chapter introduces chip and wafer test data modeling using Generative Adversarial Networks (GAN) and Diffusion Models, and explores the application of chip virtualization in secure data packaging, compression, and cross-domain delivery. These methods enable designers to generate realistic design examples, perform DTCO, enhance production capacity, and achieve energy-efficient designs.

10.1. Dataset Preparation

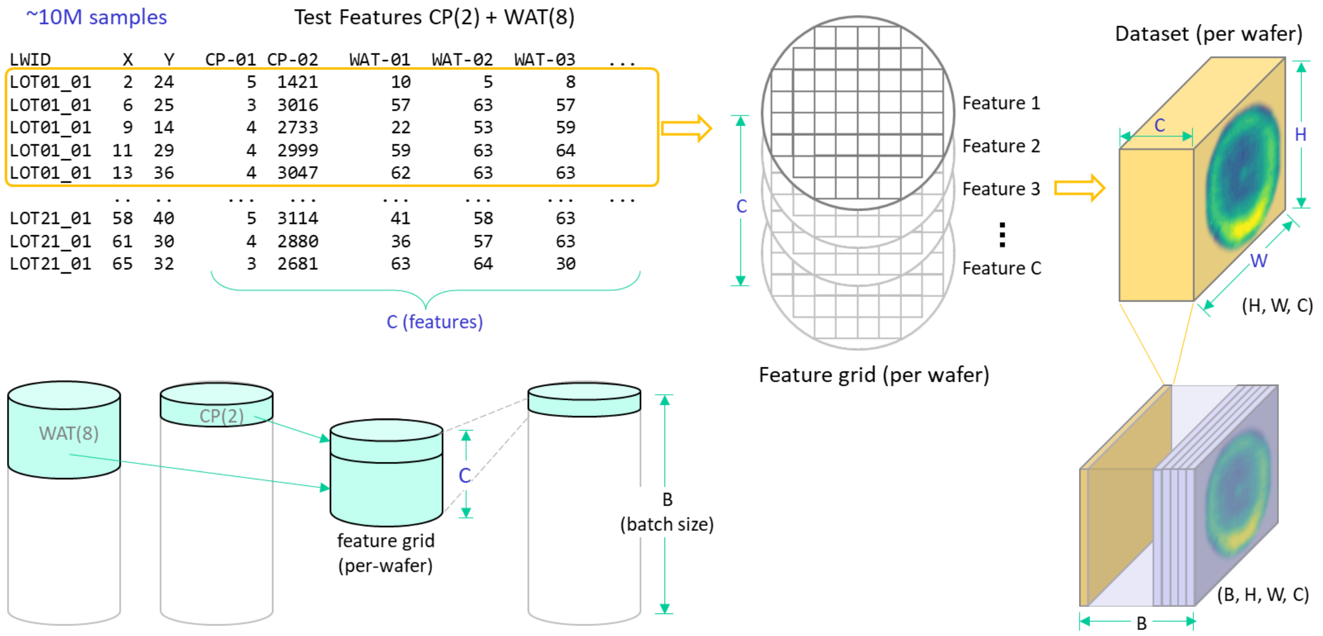
The dataset used in this study includes data from 3,984 wafers, totaling around 12 million chip data points. Outliers beyond 3σ and missing data are removed, while chips with process uniformity defects are retained for model training. To maintain confidentiality, the data is normalized to a range of -1 to 1. The dataset, containing chip coordinates and 14 features, is converted into a multi-channel image format, resulting in a size of 65×66×14 (height × width × features).

TABLE I outlines the 14 features, six from chip tests (CP) and eight from wafer acceptance tests (WAT). Each feature represents a specific physical property, such as CP1 for leakage current and CP2 for chip speed. Four other frequency-related features (CP3 to CP6) describe functional accuracy at 300MHz, 400MHz, 500MHz, and 600MHz. Functional accuracy refers to the number of processor cores that produce correct outputs at each frequency. These features offer valuable insights into chip behavior, supporting design margin analysis and optimization..

TABLE I: Features of the Dataset



Feature	Description	Unit
CP1	Leakage current	μA
CP2	Chip speed	Hz
CP3	Functional accuracy at 300MHz	%
CP4	Functional accuracy at 400MHz	%
CP5	Functional accuracy at 500MHz	%
CP6	Functional accuracy at 600MHz	%
WAT1	Gate threshold voltage of low threshold NMOS	V
WAT2	Gate threshold voltage of low threshold PMOS	V
WAT3	Gate threshold voltage of ultra-low threshold NMOS	V
WAT4	Gate threshold voltage of ultra-low threshold PMOS	V
WAT5	Drain current of the low threshold NMOS	mA
WAT6	Drain current of the low threshold PMOS	mA
WAT7	Drain current of the ultra-low threshold NMOS	mA
WAT8	Drain current of the ultra-low threshold PMOS	mA



~10M samples

Test Features CP(2) + WAT(8)

LWID	X	Y	CP-01	CP-02	WAT-01	WAT-02	WAT-03	...
LOT01_01	2	24	5	1421	10	5	8	
LOT01_01	6	25	3	3016	57	63	57	
LOT01_01	9	14	4	2733	22	53	59	
LOT01_01	11	29	4	2999	59	63	64	
LOT01_01	13	36	4	3047	62	63	63	
	..	..	...	...	...	...	...	...
LOT21_01	58	40	5	3114	41	58	63	
LOT21_01	61	30	4	2880	36	57	63	
LOT21_01	65	32	3	2681	63	64	30	

Fig. 10-1 Dataset Conversion

As shown in Fig. 10-1, the figure illustrates converting wafer data into a multi-channel image format. The original CP and WAT test data are transformed into 2D images with multiple feature dimensions (parameter C). The size of C is directly related to model size, and training time increases non-linearly with C. The dataset in the legend includes 8 WAT measurements and 2 CP measurements, totaling 10 dimensions (C=10). Notably, the wafer is circular, but the data is represented in a rectangular format, with data outside the wafer boundary filtered using a mask to maintain analysis accuracy.

To further augment the training set, slight angle rotations are applied, especially useful when data is limited during early mass production. This method increases data diversity and enhances model stability. By simulating potential rotational defects and process parameter distributions, as shown in Fig. 10-2, we can capture key features of the chip manufacturing process, improving the model's training effectiveness.

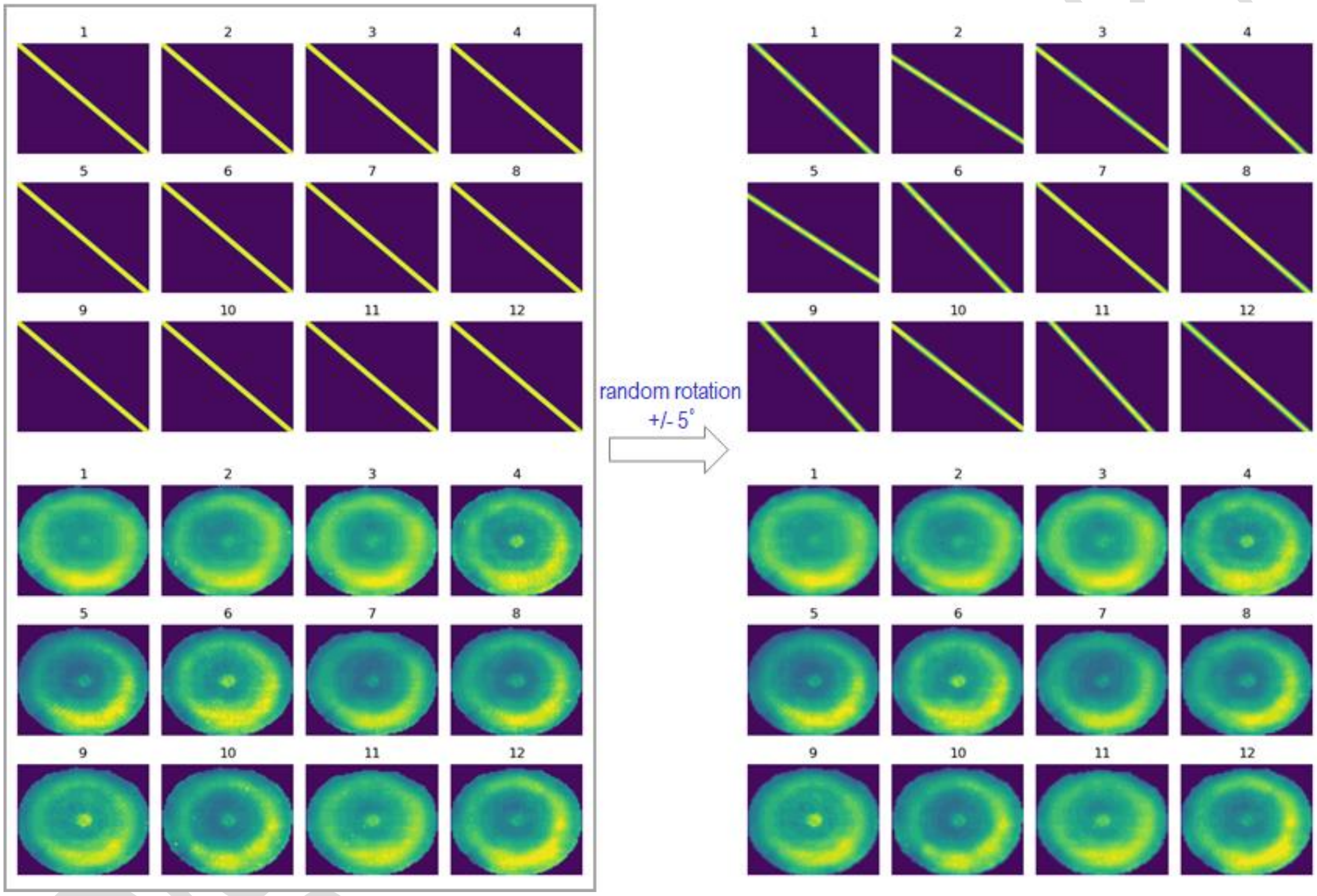


Fig. 10-2 Data Augmentation

10.2. GAN-based Virtual Silicon (GAN-VS)

This section explains how the GAN model is used to model multi-dimensional test data from the chip manufacturing process, covering chip performance, wafer process characteristics, and potential defects, to accurately capture process defects and parameter uniformity.

10.2.1. GAN Model

As shown in Fig. 10-3, the generator consists of convolutional layers and outputs multi-dimensional images through a Tanh layer to simulate chip images. The discriminator uses convolutional layers and a Sigmoid layer to assess the authenticity of the chip data. The two components work together to generate high-quality chip data.

During training, we use BCE Loss and minimize the difference between generated and real chips through gradient descent. To enhance training stability, Batch Normalization and LeakyReLU activation are applied. The Adam optimizer is used with an initial learning rate of 0.001, decaying by 0.9 every 100 epochs. Training runs for 10,000 epochs with a batch size of 20. Ultimately, the GAN model generates realistic chip data, including chip location, wafer flatness, and process defects, forming a reliable foundation for simulating and analyzing the manufacturing process.

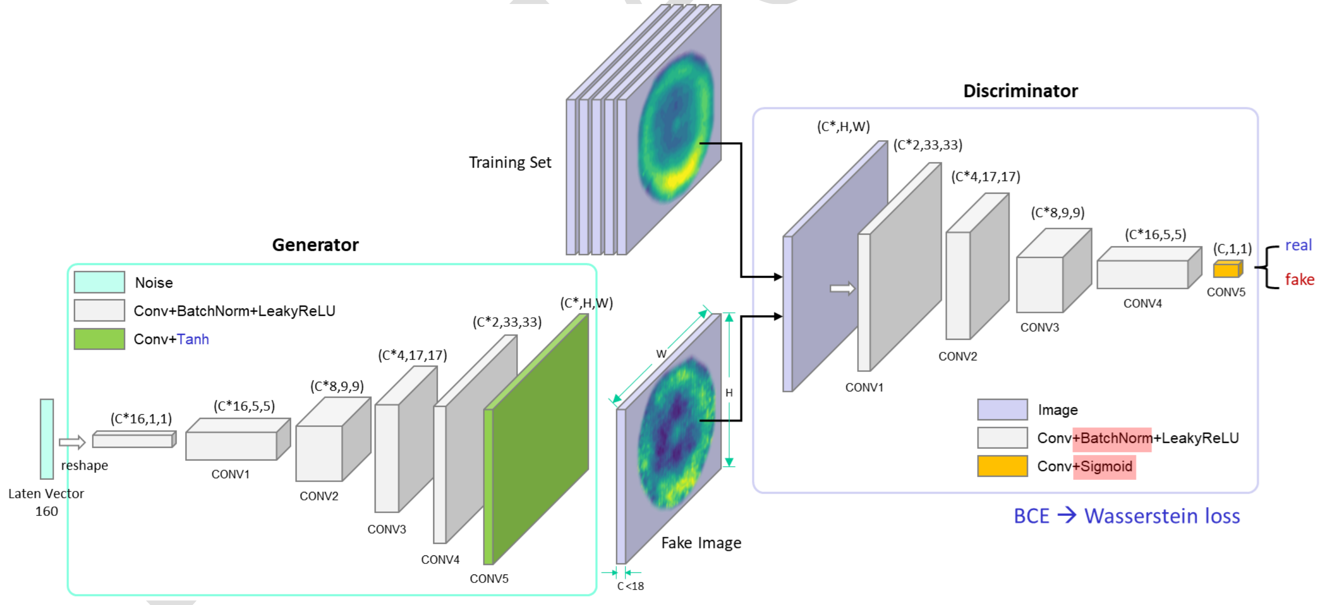


Fig. 10-3 GAN Modeling

10.2.2. GAN Model Performance Evaluation

We evaluate the GAN model by plotting 2D scatter plots of generated and real samples, with feature combinations as axes. Ideally, the scatter plots should overlap, ensuring that the generated data aligns with the real data in both individual and joint feature distributions, as shown in Fig. 10-4.

Additionally, we use quantitative metrics, such as Jensen-Shannon divergence (JS Divergence) to compare the probability distributions of the generated and real data, and the KDE metric to assess the differences between feature distributions, ensuring the reliability and accuracy of the generated chip data.

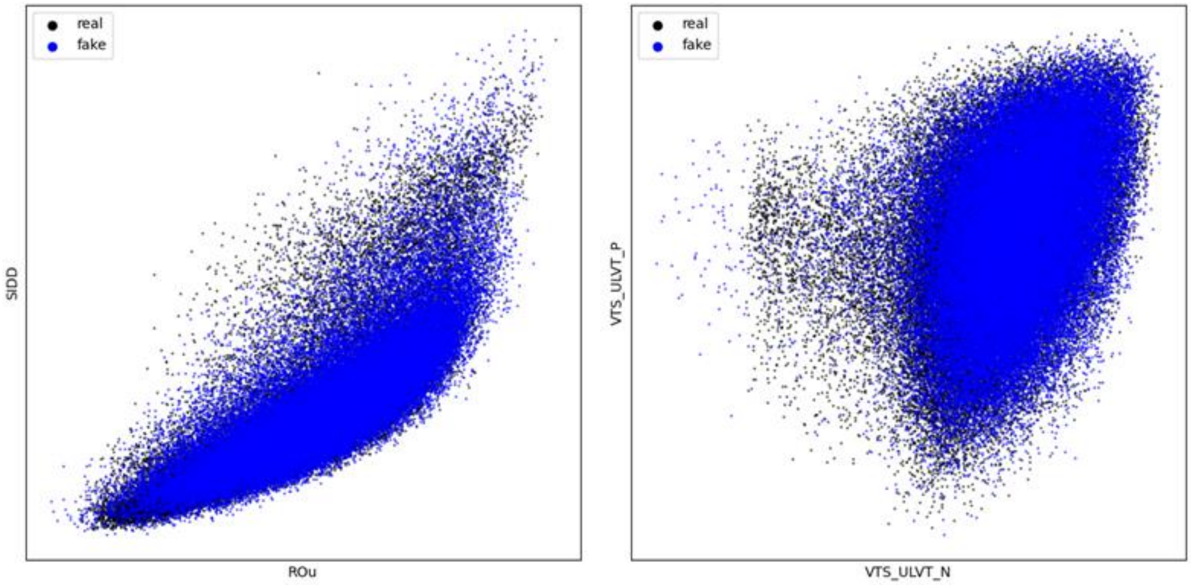


Fig. 10-4 Feature Scatter Plot for GAN Model Similarity

We evaluate the similarity between real data and GAN-generated samples by comparing their feature distributions. Using Jensen-Shannon (JS) divergence, lower values indicate greater similarity between the generated and real samples. The JS divergence between the real distribution P and generated distribution Q is defined as follows:

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M) \quad (1)$$

The Kullback-Leibler divergence DKL(P||M) can be calculated using the following formula:

$$D_{KL}(P||M) = \sum_{x \in X} P(x) \log \frac{P(x)}{M(x)} \qquad (2)$$

where M represents the mixed distribution of P and Q, defined as:

$$M = \frac{1}{2}(P + Q) \qquad (3)$$

Since the JS divergence between the two distributions ranges from 0 to 1, the similarity of the JS divergence between P and Q is defined as:

$$Similarity = 1 - D_{JS}(P||Q) \qquad (4)$$

JS divergence analysis (Similarity=1-JS) shows that on a 10-dimensional dataset excluding CP3-CP6, the model-generated virtual chip data closely matches real data, with similarity ranging from 0.98 to 1.0. Additionally, the model exhibits strong robustness and generalization performance for anomalous datasets. See Fig. 10-5 for details.

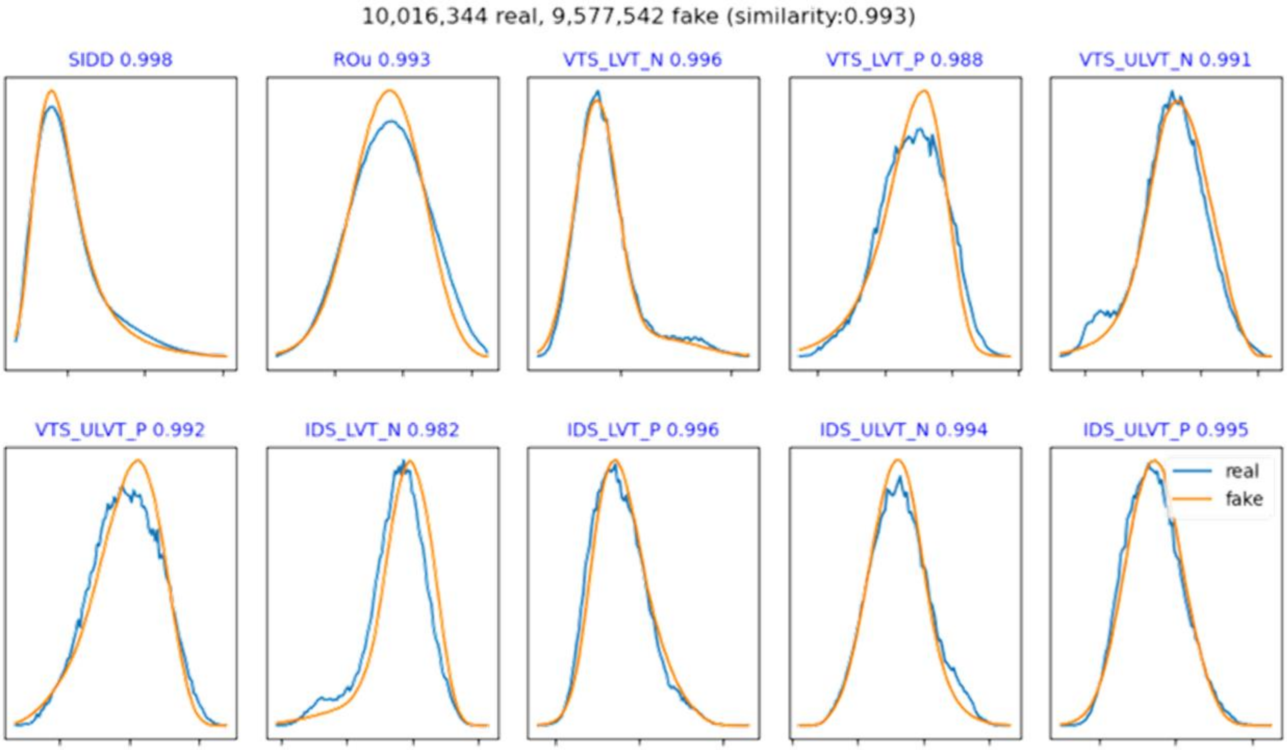


Fig. 10-5 Feature Distribution Similarity (C=10)

As shown in Fig. 10-6, experimental results reveal that the GAN model's high-dimensional chip data closely matches real chip data in scatter plots between any two features. The model successfully captures process variability during process adjustments, marked by yellow, green, and blue circles. In high-dimensional space, the relationships between features and their joint distributions remain consistent with the original data.

To enhance the model, we can exclude data irrelevant to mass production, such as intentionally skewed wafers, to avoid learning anomalies from early process adjustments. Unlike traditional methods, the GAN model effectively learns nonlinear relationships in the chip manufacturing process, capturing subtle details and fitting the real data distribution more precisely. Further analysis confirms that the GAN model accurately captures parameter distributions, wafer-level uniformity, and manufacturing process details, reflecting more realistic wafer defects, as seen in Fig. 10-7.

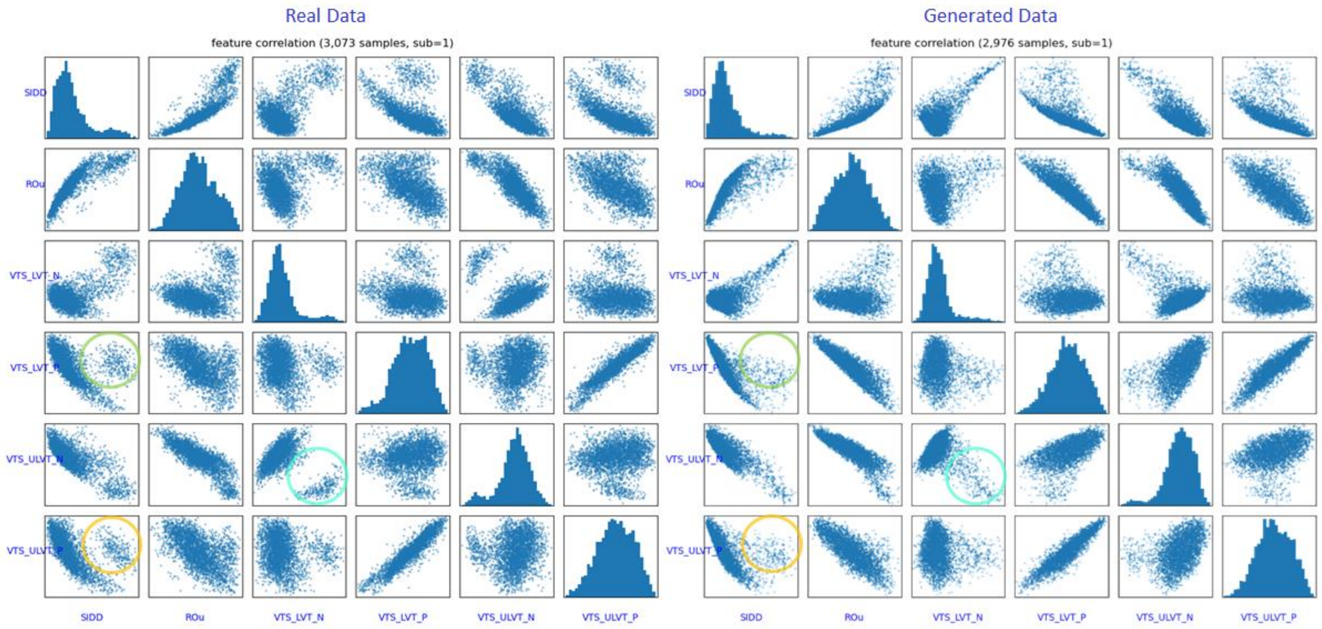


Fig. 10-6 Feature Correlation Matrix between Generated and Real Silicon

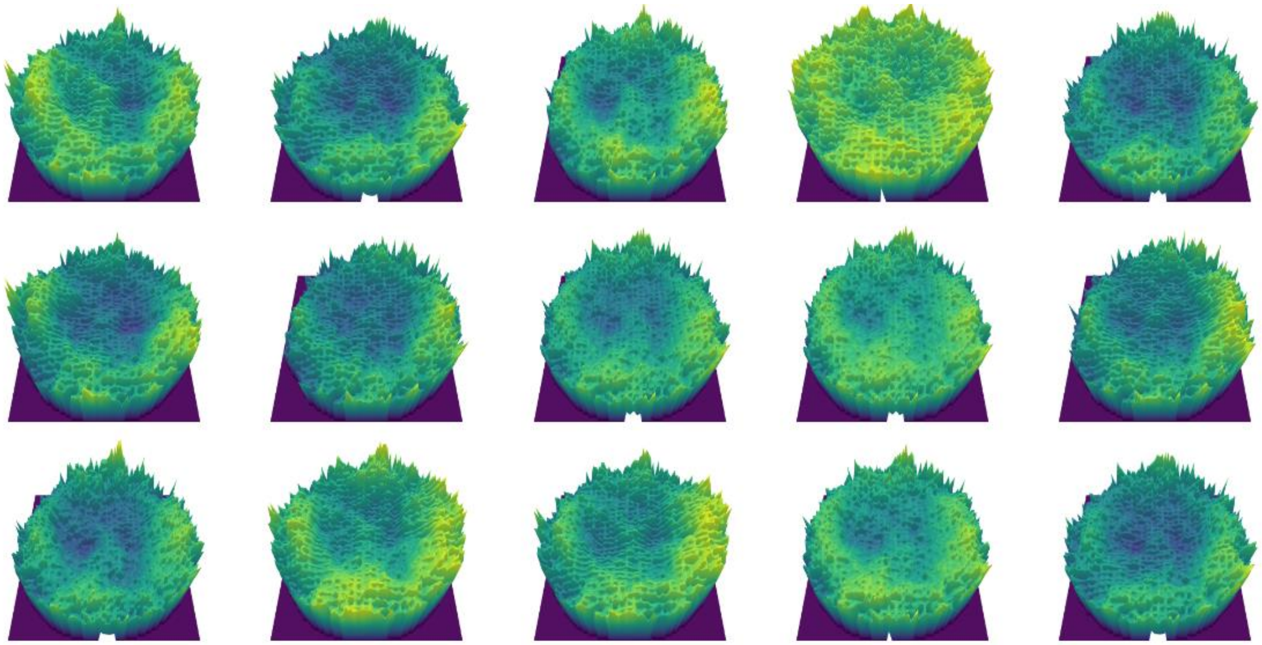


Fig. 10-7 Wafer-level Feature Uniformity of Generated Silicon

GAN-based methods often face challenges in convergence and instability. While the improved WGAN (using Wasserstein Loss and removing BatchNorm and Sigmoid layers, as shown in Fig. 10-3) enhances performance, it still struggles with data distributions that have multiple peaks. Additionally, chip functional accuracy (e.g., frequency-dependent features like CP3-CP6) typically follows non-Gaussian distributions, such as bimodal, skew-normal, log, or cosh forms. As multi-core chip performance is highly influenced by operating frequency, GANs face difficulties in handling such tasks. The next section will explore how diffusion models can address these challenges.

10.3. Diffusion Model-based Virtual Silicon (DM-VS)

This section introduces the Denoising Diffusion Probabilistic Models (DDPM), which generate higher-quality silicon wafer data and overcome GAN limitations in multi-core chip performance features. Evaluating data distribution and quality with JS divergence and Fréchet Inception Distance (FID), the results show that the diffusion model accurately extracts feature distributions from silicon wafer data, generating numerous samples to support deeper analysis and accelerate the DTCO process. Compared to GANs, the diffusion model generates virtual wafers with a data distribution closer to real data on a 14-dimensional dataset, achieving a JS divergence similarity of 0.987 and an FID of 6.28.

10.3.1. Denoising Diffusion Probabilistic Model (DDPM))

The Denoising Diffusion Probabilistic Model (DDPM) is a generative model that refines noisy samples into high-quality data by gradually removing noise. Widely successful in text and image generation, DDPM produces highly detailed and realistic results. Its core concept involves reversing the diffusion process, starting from noise samples and gradually correcting them to match real data distributions, yielding high-quality outputs. Fig. 10-8 illustrates the DDPM process, which includes both the forward and reverse processes, each involving T steps.

In the forward process, DDPM gradually adds Gaussian noise to the data, transforming it into a simpler distribution (usually Gaussian). This is done step by step, with the initial wafer sample $w0$ having noise added at each step, eventually turning into pure Gaussian noise wT. The network learns how to add noise, enabling accurate predictions in the reverse process.

In the reverse process, the model gradually reverses the forward process, recovering the original data distribution from the noisy samples. Specifically, the generation of the wafer begins with the noisy sample wT, which is then input into the neural network along with the time step t=T−1 to predict the noise added at the current step. Subsequently, the predicted noise is subtracted from wT to obtain wT−1. This process is repeated for T steps, eventually transforming the noisy sample wT into a high-quality wafer sample $w0$.

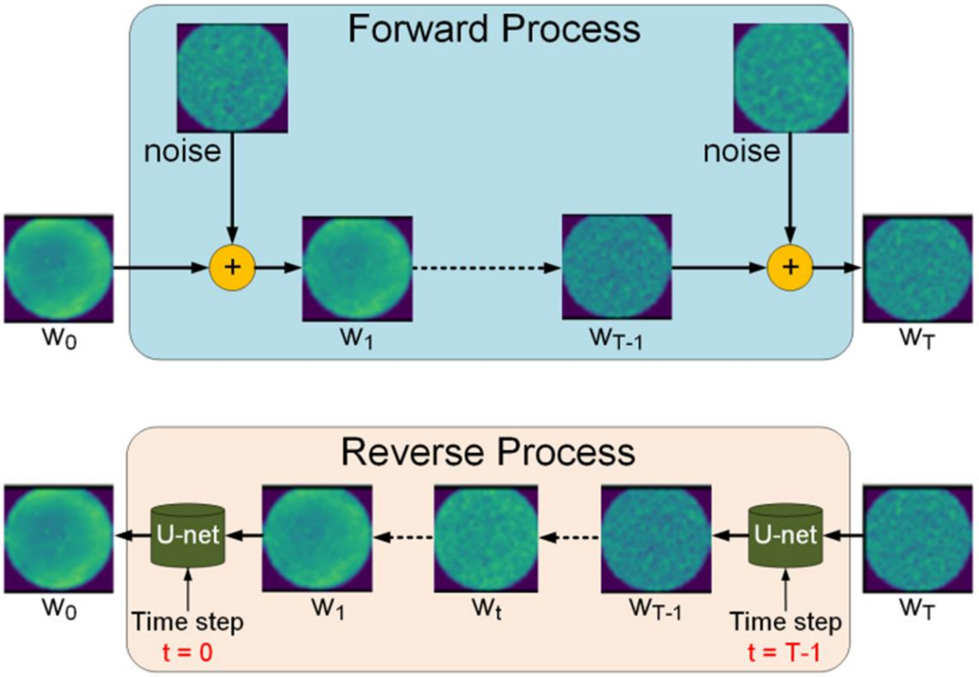


Fig. 10-8 Forward and Reverse Processes of the Diffusion Model

U-Net, commonly used for image segmentation and generation, has the structure shown in Fig. 10-9. The input passes through a convolutional layer to expand the channels to 16. The model contains two downsampling blocks that increase the channels to 64, followed by two upsampling blocks that restore the resolution while reducing the channels back to 16. A final convolutional layer generates the output with the desired dimensions. Fig. 10-10 provides further details of the ResNet block.

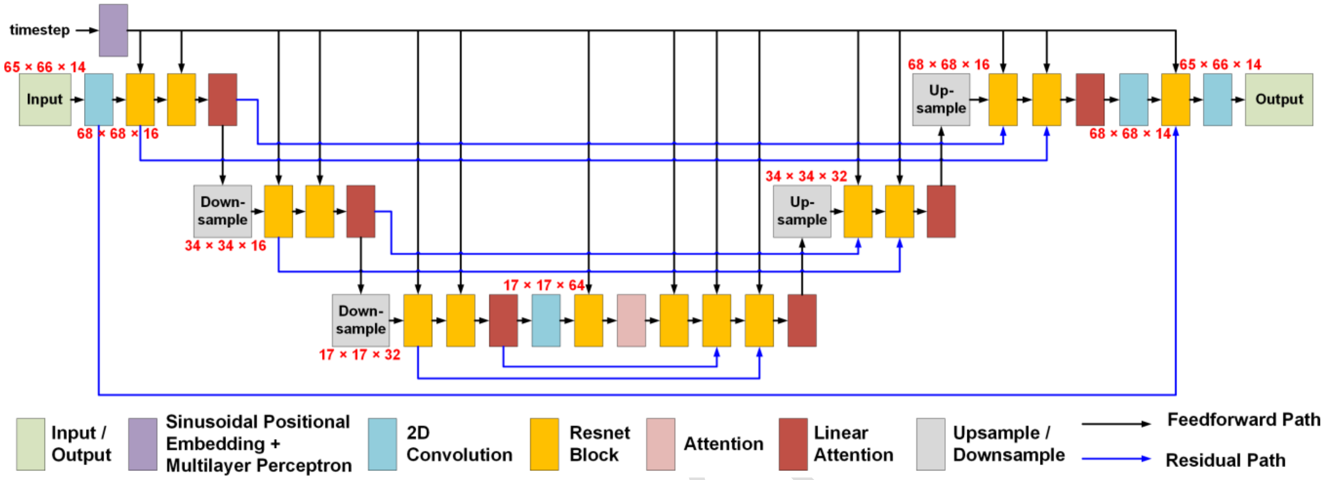


Fig. 10-9 U-Net of Diffusion Model

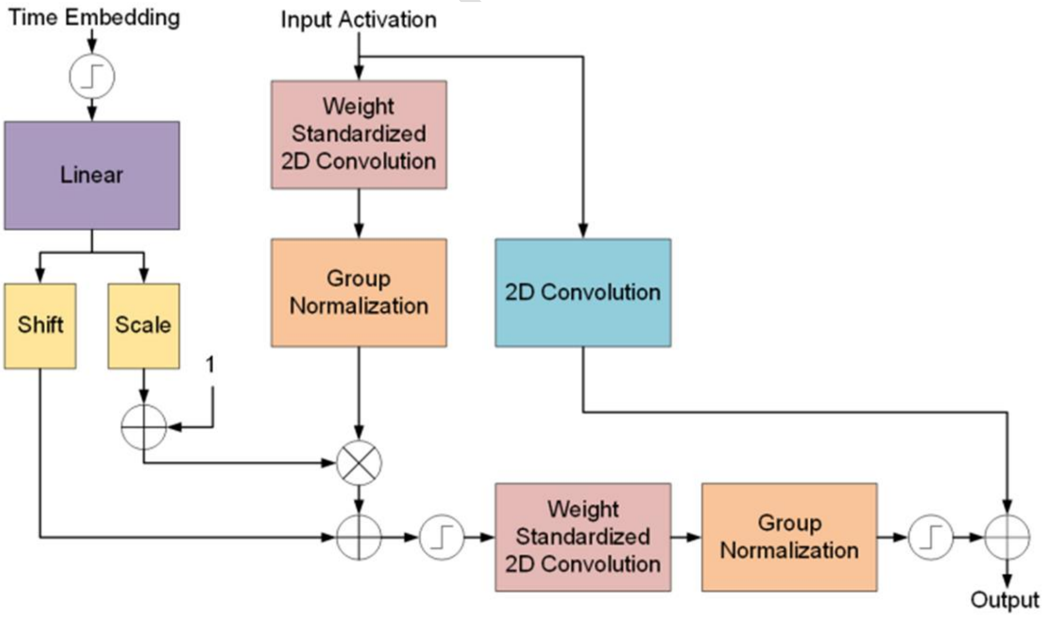


Fig. 10-10 Illustration of a ResNet Block

10.3.2. Diffusion Model Performance Evaluation

This section introduces wafer-level evaluation metrics in addition to chip-level analysis. Unlike chip-level analysis, wafer-level focuses on chips from the same wafer, allowing for a more effective evaluation of whether the diffusion model successfully captures the complexity of the training data. This provides a more objective way to compare the data generated by the diffusion model with that generated by GAN.

In image generation tasks, FID is commonly used to evaluate the quality and diversity of generated data relative to real data. A lower FID indicates that the generated data more closely resembles the real data, reflecting higher generation quality. The FID is calculated by passing both real and generated data through a pre-trained Inception-v3 model and comparing their feature distributions. Fig. 10-11 illustrates the process of transforming wafer data into the dimensions required for FID calculation.

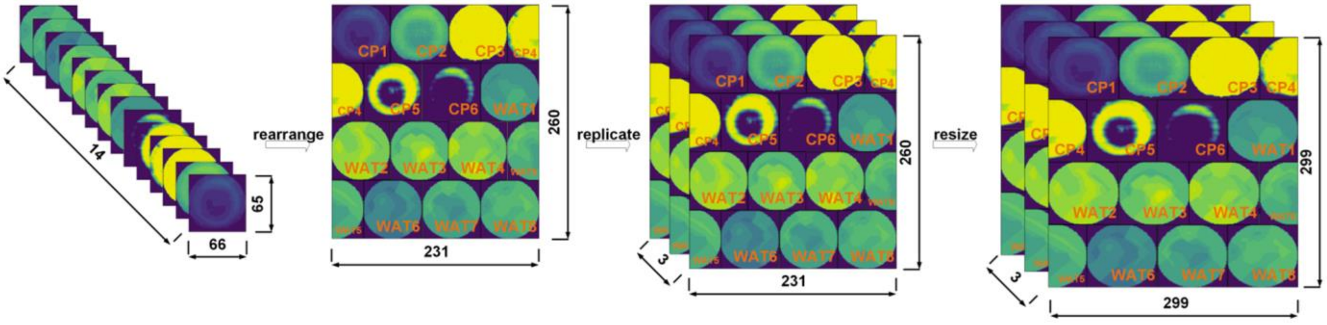


Fig. 10-11 Data Shape Transformation for FID Evaluation

First, the wafer sample of size 65×66×14 is rearranged to 231×260×1, duplicated across 3 channels, and resized to 299×299×3 using bilinear interpolation. The data is then passed through the Inception-v3 network, and the activation vector from the final pooling layer is extracted to obtain the wafer's vector representation. After converting both real and generated data into vector representations, their means (μ1, μ2) and covariance matrices (C1, C2) are computed. The FID is then calculated using the following formula:

$$FID = \|\mu_1 - \mu_2\|_2^2 + Tr(C_1 + C_2 - 2(C_1 C_2)^{\frac{1}{2}} \qquad (5)$$

Here, the symbol ‖·‖₂ represents the L2 norm or Euclidean norm, and Tr(·) denotes the trace of a matrix, which is the sum of the elements on its main diagonal.

- **Die-Level Analysis**

At the chip level, scatter plots compare the joint distributions of multiple features between real and generated data. Fig. 10-12 shows the scatter plots for four feature pairs generated by the diffusion model, where the generated data points closely match the real data points, demonstrating high similarity in their joint distributions.

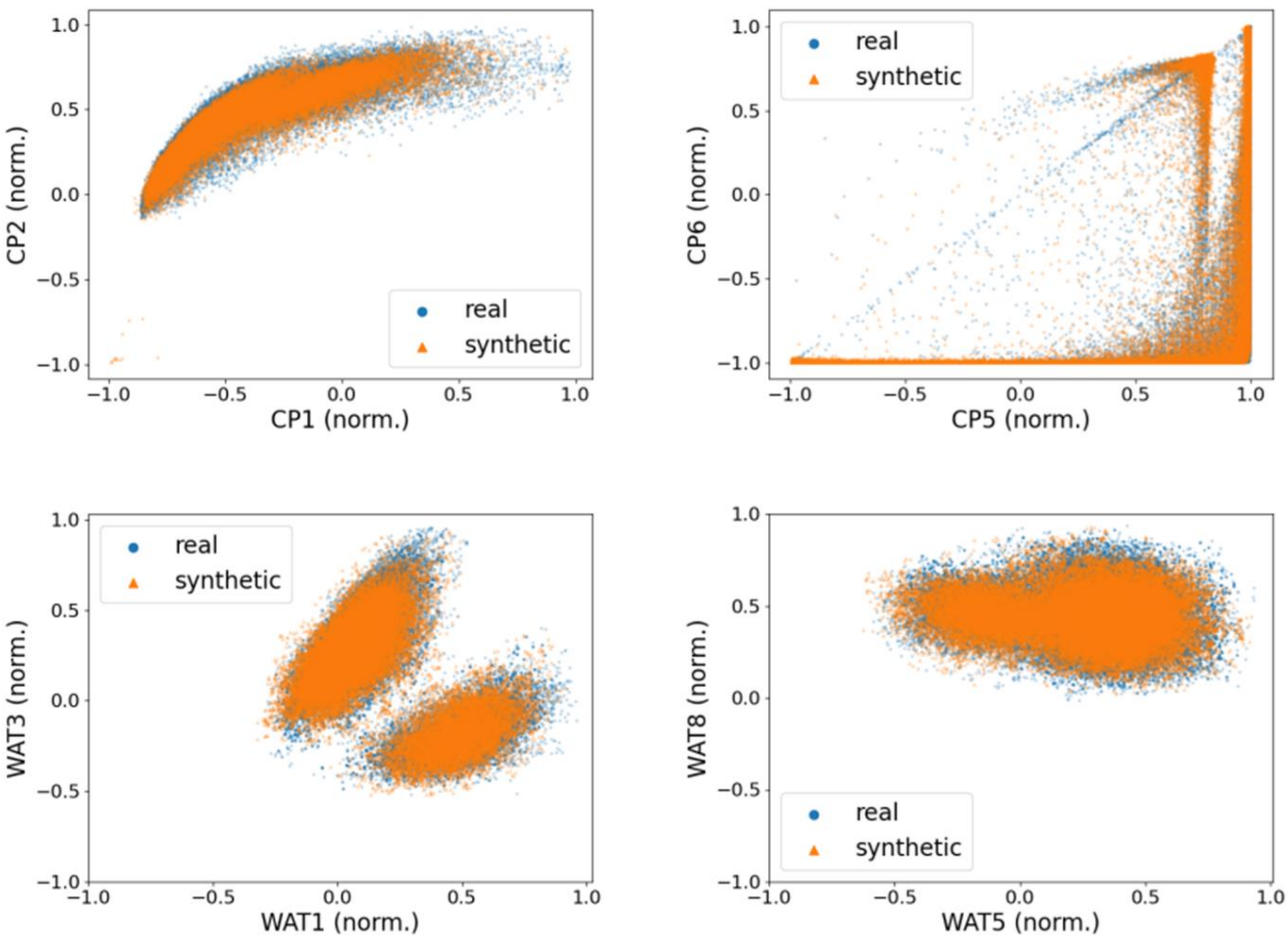
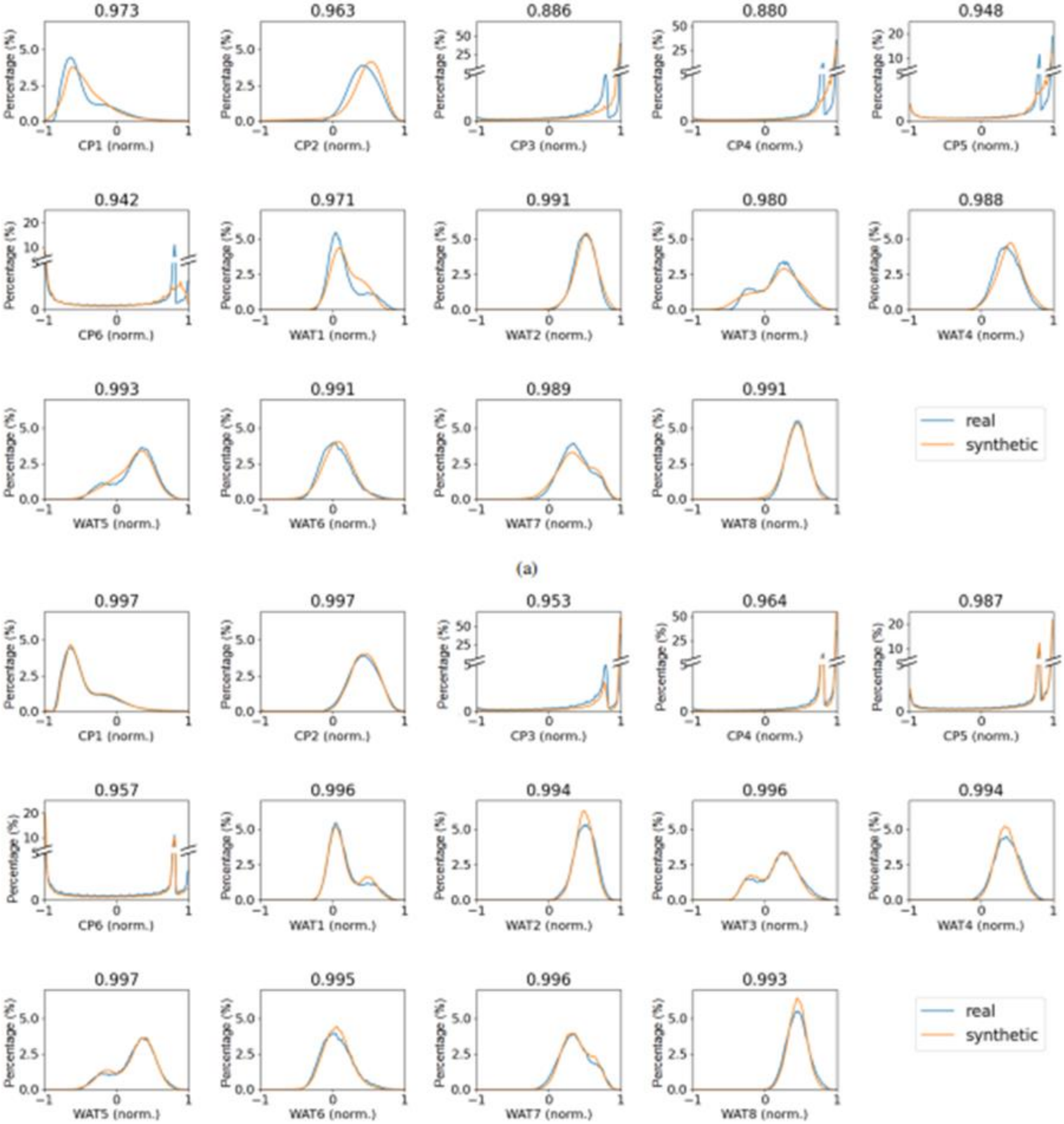


Fig. 10-12 Feature Scatter Plot for Diffusion Model Similarity

Fig. 10-13 shows the distribution of all feature values, with the JS divergence similarity indicated above each chart. In Fig. 10-13 (a), the GAN-generated distribution underperforms on features CP3, CP4, CP5, and CP6 due to the multiple peaks in their distributions. The mismatch in peak height and position leads to lower JS divergence similarity, and the GAN fails to capture the prominent peaks accurately.

Fig. 10-13 (b) shows the diffusion model's distribution, demonstrating its superiority in capturing the real distribution. For features CP3, CP4, CP5, and CP6, the diffusion model accurately reproduces the peak positions and heights. Specifically, for multi-core frequency performance features (CP3–CP6) with complex distributions like log or cosh, the GAN achieves a JS divergence similarity of 0.963, while the diffusion model improves this to 0.987, highlighting its superior performance on intricate distributions.



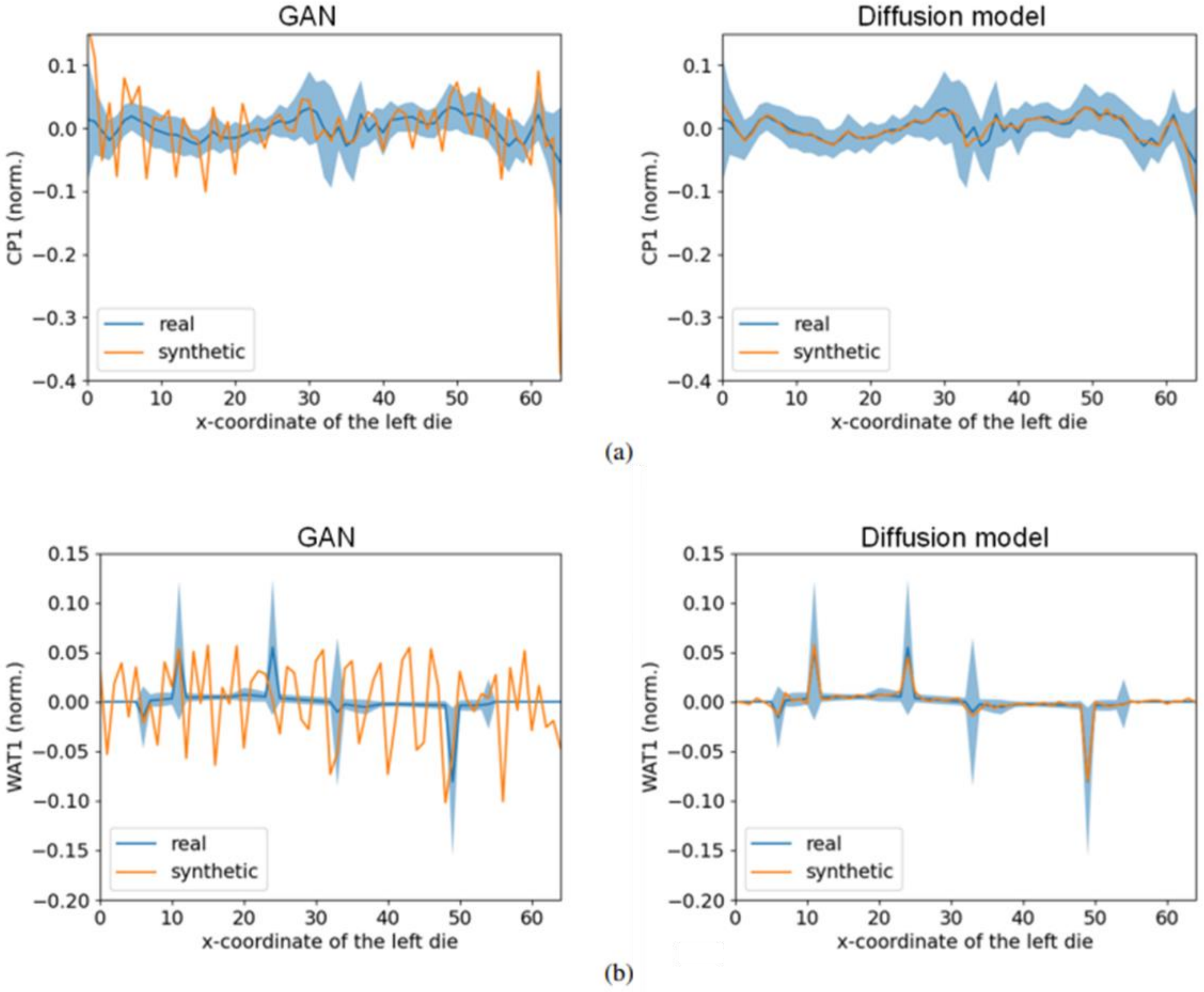
(a)

Fig. 10-13 Feature PDFs of GAN (a) and Diffusion Model (b)

- **Wafer-Level Analysis**

At the wafer level, we assess the differences between adjacent chips along a specific direction to evaluate how well the spatial variations in the generated data match those in the real data. By calculating the average differences across wafers, we can quantify the similarity between real and generated data.

For simplicity, this analysis focuses on the secant along the horizontal direction of the wafer, examining the average differences between adjacent chips in that direction. Fig. 10-14 shows the average difference analysis of CP1 and WAT1 generated by the GAN and diffusion model. The x-axis represents the horizontal coordinates, and the y-axis shows the average difference. The light blue area indicates one standard deviation range of the real data's average difference, providing an intuitive comparison between generated and real data.



(a)

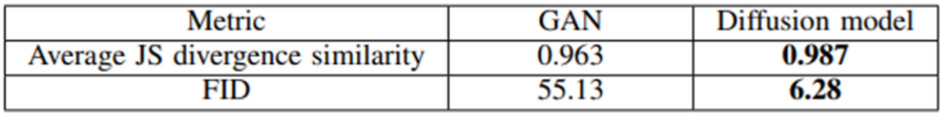
(b)

Fig. 10-14 Average Difference Analysis for (a) CP1 and (b) WAT1

Fig. 10-14 (a) shows the average difference of CP1. For GAN-based methods, the generated data's average difference often falls outside the standard deviation range, with significant fluctuations indicating inaccurate patterns. In contrast, the diffusion model keeps the average difference consistently within the standard deviation range. Fig. 10-14 (b) shows similar results for WAT1, where GAN-generated data exhibits notable fluctuations, while the diffusion model captures the peak differences of real WAT1 data. The average difference from the diffusion model stays within one standard deviation of real data, showing similar spatial variations in both horizontal and vertical directions.

Table II compares the FID (Frechet Inception Distance) of virtual wafers generated by GAN and the diffusion model to real data. The real data's FID is 1.39, calculated by splitting the wafer data into two equal parts. GAN's FID is 55.13, indicating difficulty in capturing multi-modal distributions, while the diffusion model's FID is 6.28, closely matching real data and showing a significant improvement in generation quality. Overall, while GAN performs well on some features, it struggles with multi-modal distributions. The diffusion model accurately simulates real data's multi-feature distribution, with high consistency in chip distribution and horizontal secant differences.

TABLE II: Quality Comparison of Generated Data



Metric	GAN	Diffusion model
Average JS divergence similarity	0.963	**0.987**
FID	55.13	**6.28**

Chapter 11. Generative AI-Driven Chip Efficiency Optimization and Modeling

11.1. WAT Super Resolution (WAT-SR)

Traditional WAT sampling, due to sparse data points, fails to fully capture the overall process characteristics, impacting decision reliability and accuracy. Insufficient samples can lead to biased decisions and reduced parameter adjustment precision. High-resolution CP data reveals significant chip characteristic differences even within the same exposure range (e.g., 5×7 shot), as shown in Fig. 11-1.

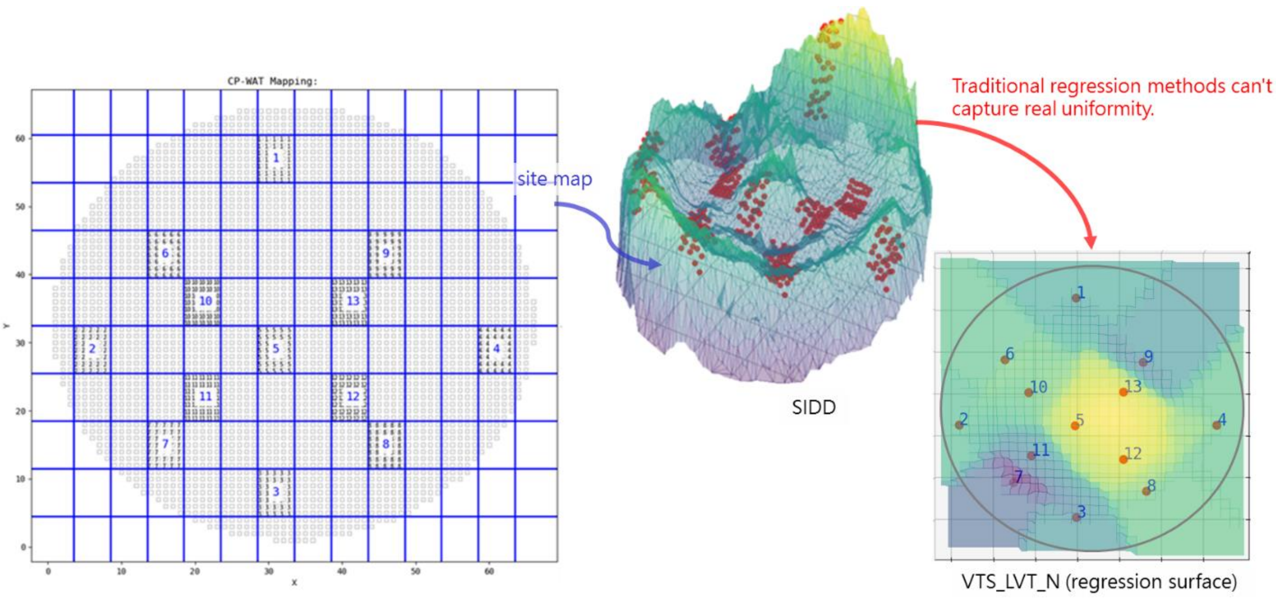


Fig. 11-1 WAT Sampling Issue

Using the average data from a single lot (25 wafers) as an example, as the number of WAT sampling points increases (e.g., 80-point full-map), the uniformity of WAT process parameters and CP high-resolution features becomes clearer and more consistent, as shown in Fig. 11-2. Relying solely on sparse WAT sampling during mass production (e.g., 13 points or fewer) or linear interpolation would make it difficult to capture the true wafer surface uniformity. Therefore, careful evaluation of sampling strategies is needed to ensure sufficient data for more accurate process control and decision-making.

Traditional polynomial regression methods struggle to capture wafer-level process parameter distributions due to their limited ability to handle high-dimensional, spatially correlated nonlinear variations, as shown in Fig. 11-3. To improve data quality and decision-making, innovative approaches such as machine learning or generative models should be adopted.

Fig. 11-4 illustrates how most spatial domain signals can be approximated by summing a few key harmonics in the frequency domain. Using 3 to 4 different harmonic periods, radii, and phase shifts can effectively approximate wafer process uniformity and system-level defects like edge effects, volcanic cones, and concentric circular nonuniformities. The WAT-SR (WAT Super-Resolution) technique enhances WAT resolution from sparse sampling points using neural networks, capturing true wafer-level uniformity.

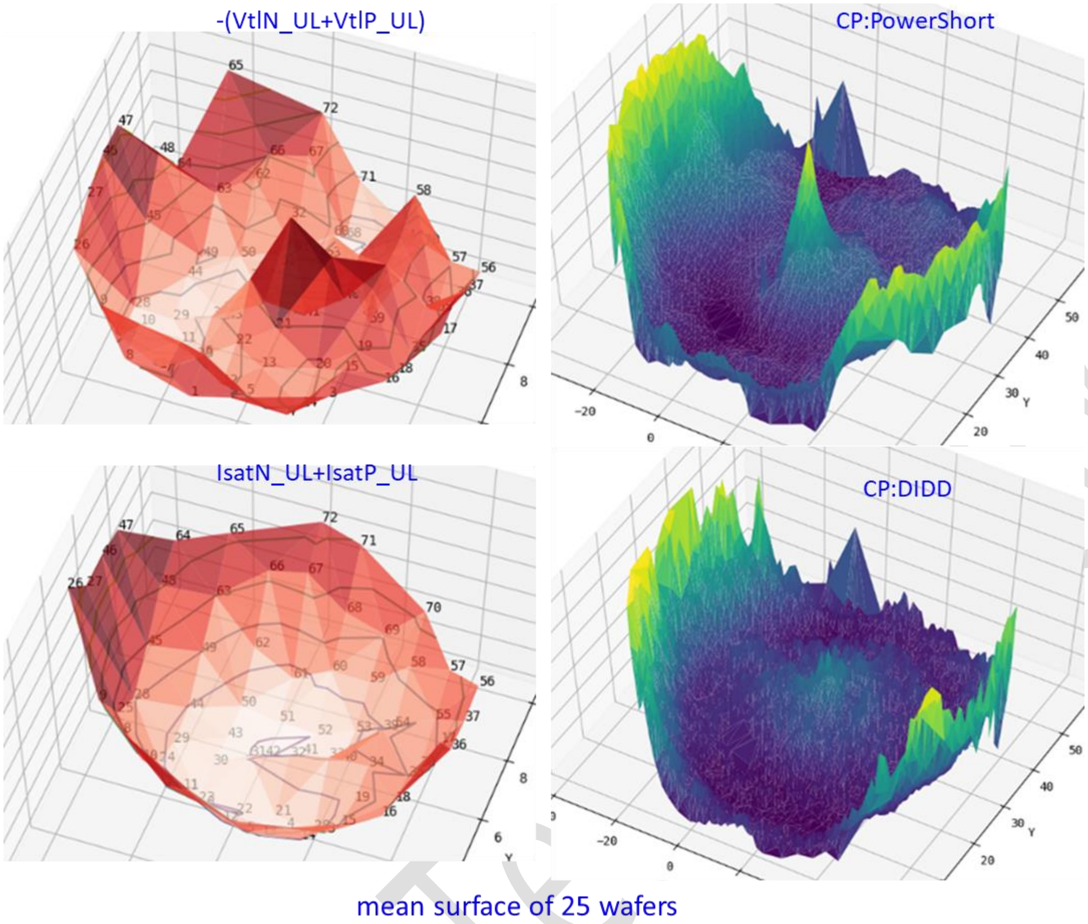


Fig. 11-2 Full Map Uniformity Comparison of WAT and CP

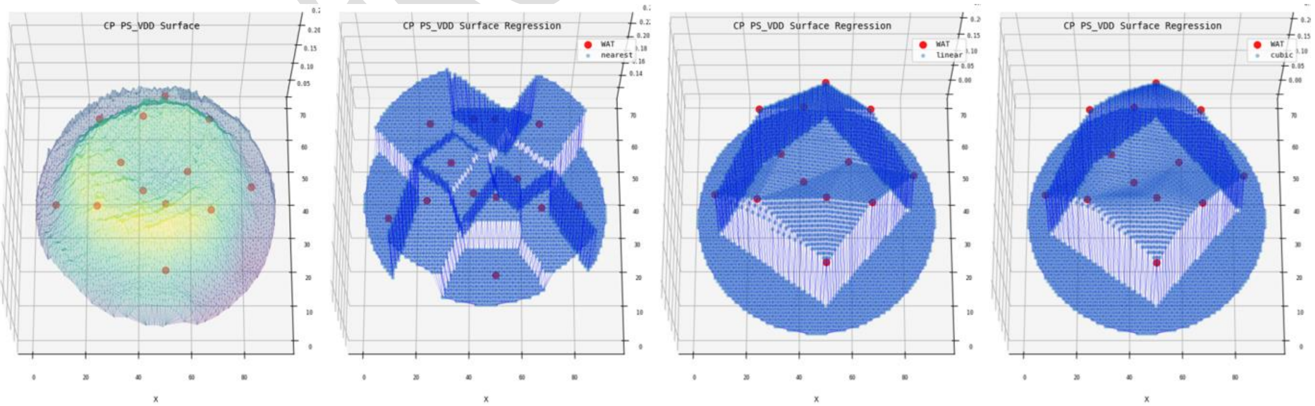


Fig. 11-3 Traditional Regression Models Miss Structural Information

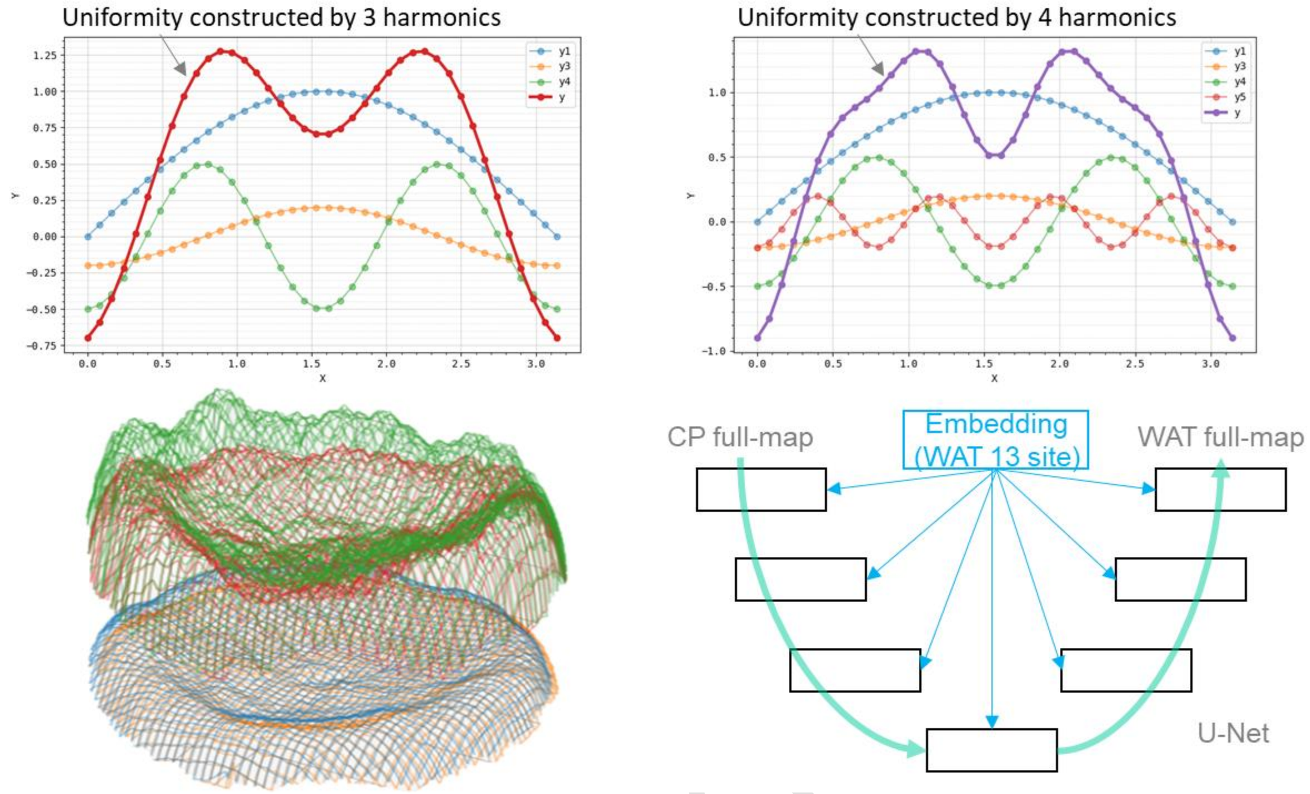


Fig. 11-4 Harmonic in Frequency Domain

The core idea is to use U-Net to convert sparse WAT data (e.g., 13 SITE points) into embedding vectors via MLP, as shown in Fig. 11-5. These embeddings, combined with high-resolution CP data (such as RO or SIDD, 65x66 chip count), serve as training samples to generate high-resolution WAT data that closely matches the original. This approach not only accurately reconstructs wafer uniformity but also captures defect features in the wafer process.

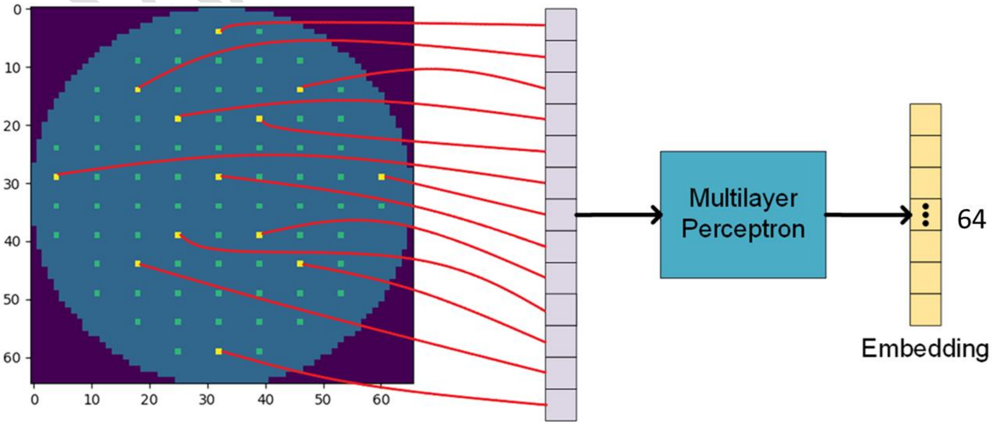


Fig. 11-5 Embedding 13 Sparse WAT Points Using an MLP

U-Net effectively balances global structure inference and local detail preservation, as shown in Fig. 11-6. The latent space captures global process patterns, while skip connections provide local details, enabling the model to capture both global features and low-level spatial information. For example, the latent space captures global CP process features but may lose precise location details. Skip connections address this by passing low-level encoder features directly to the decoder, enhancing the model's ability to preserve local structure and spatial details.

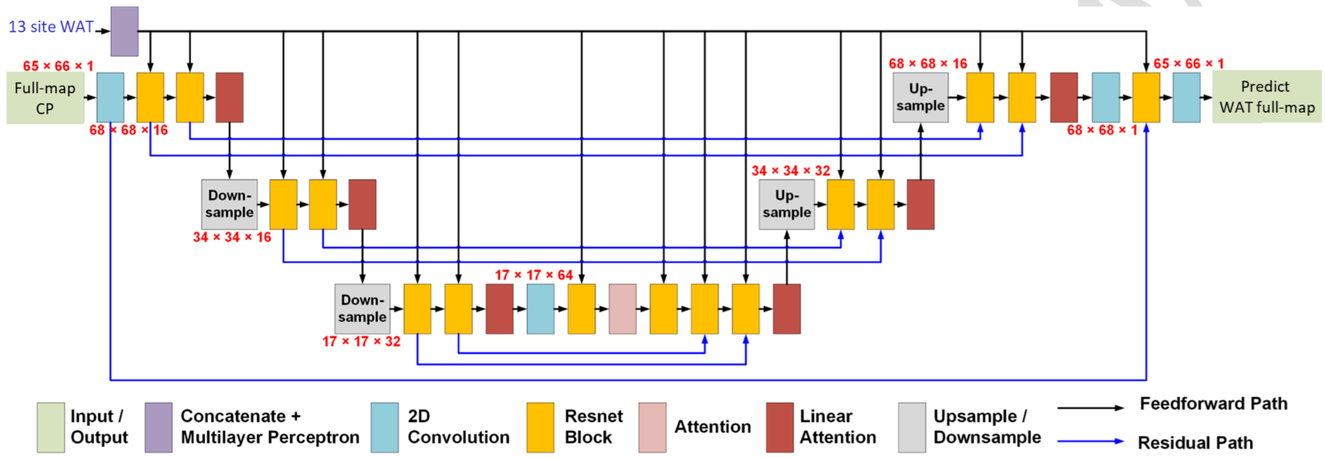


Fig. 11-6 U-Net of WAT-SR

Convolutional neural networks, through layer-by-layer compression and feature extraction, facilitate multi-frequency decomposition and synthesis in the spatial domain while effectively capturing multi-variance in high-dimensional spaces. By using sparse WAT as embedding vectors and combining them with high-resolution CP data, the model extracts structural features and details, improving performance.

The model learns the inherent distribution and structure of data through high-resolution CP, accurately representing wafer uniformity and elevating sparse WAT data to high resolution. This enhances data resolution, captures true wafer features, and improves process control accuracy and decision reliability. The challenge of sparse data is effectively addressed, leading to more precise predictions of wafer uniformity and greater decision confidence. The complete process is shown in Fig. 11-7.

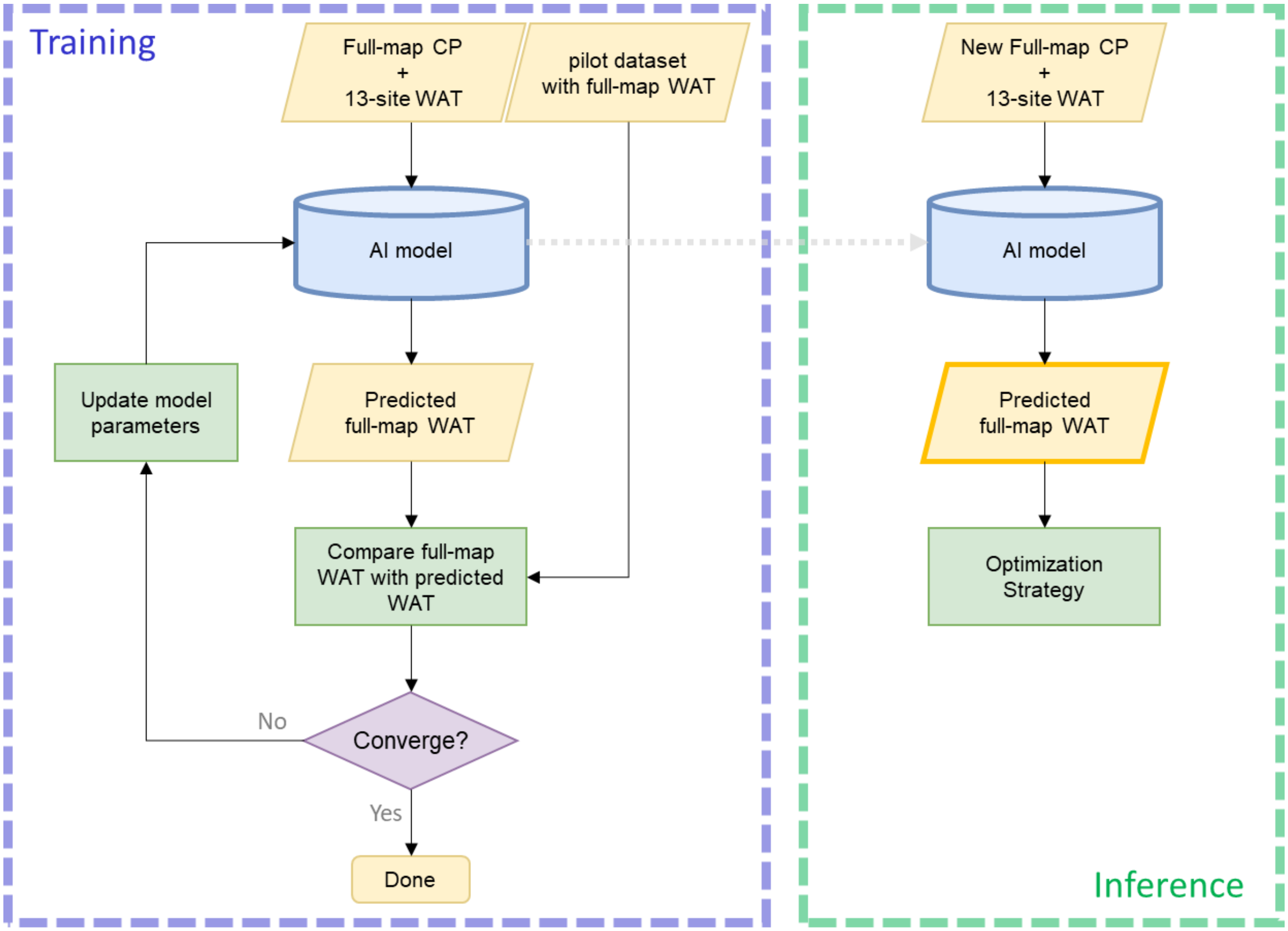


Fig. 11-7 WAT Super Resolution (WAT-SR) Flow

11.2. High-Efficiency SPICE-Silicon Bias Modeling (He-SSBM)

11.2.1. Design Principle of One-shot SPICE-Silicon N/P Correlation

The RO, located within the chip, includes multiple Delay-Lines (DL) composed of various components. By analyzing production data, we can assess the differences from the SPICE model, as shown in Fig. 11-8. For example, in a process biased towards FS (NMOS fast, PMOS slow), the DL of P-stacking components results in a lower post-silicon frequency than SPICE predicts, while N-stacking components result in a higher frequency. By evaluating the post-silicon measurement or target values from multiple DLs, we can adjust the SPICE N/P parameters, recalculating the Re-K values to better match actual silicon behavior.

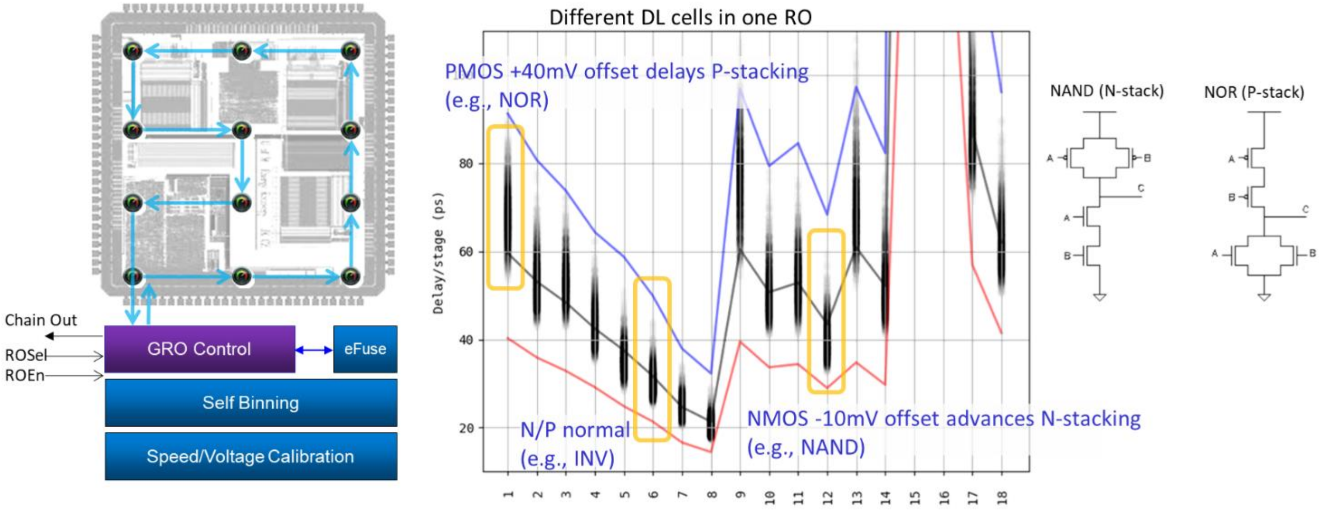


Fig. 11-8 RO Integration and S2S Correlation

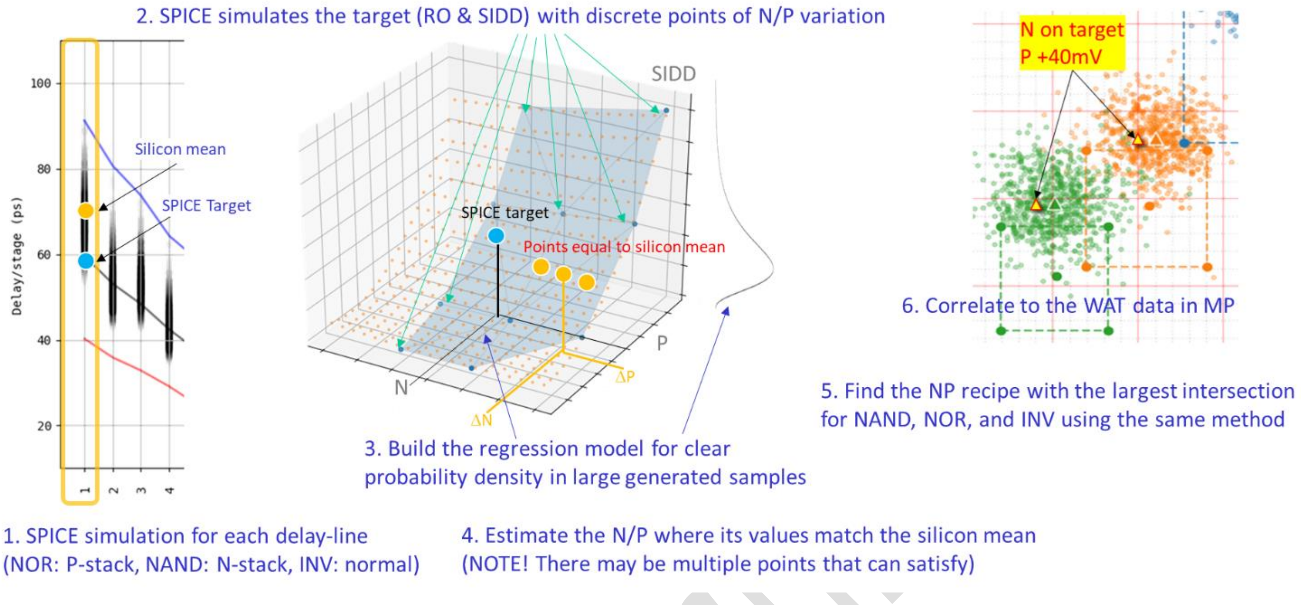
11.2.2. Design and Signoff Strategy Optimization

During chip physical design, SPICE simulations are conducted on each RO's Delay-Line (DL), including NOR (P-stack), NAND (N-stack), and INV (N/P balanced). In the mass production phase, RO measurements reveal performance deviations from the SPICE model, as shown in TABLE III. The challenge is to map these deviations to N/P offsets in SPICE to calibrate timing, Re-K, and implement a signoff strategy that aligns with actual chip data, enhancing performance and competitiveness.

TABLE III Measured Target Silicon Data

Silicon1	**Type**	**DL1**	**DL2**	**DL3**	**DL4**	**DL5**	**DL6**	**DL7**	**DL8**	**DL9**	**DL10**
	Delay	4.068	3.138	3.862	1.924	2.497	1.337	2.947	1.430	1.922	1.026
	Power	145.000	548.300	163.100	81.720	577.900	301.500	146.300	107.200	513.700	393.200
Silicon2	**Type**	**DL1**	**DL2**	**DL3**	**DL4**	**DL5**	**DL6**	**DL7**	**DL8**	**DL9**	**DL10**
	Delay	6.437	4.971	5.772	3.218	3.790	2.251	4.310	2.395	2.850	1.720
	Power	230.400	872.300	244.900	137.100	884.200	511.800	216.200	181.700	770.100	670.600

Fig. 11-9 illustrates the process of adjusting N/P offsets via RO. First, SPICE simulations generate target data (e.g., RO and SIDD) and perform discrete grid point analysis based on N/P variations. A regression model is then built to characterize the probability density distribution of large-scale samples, estimating the N/P offsets that match the chip's average values (note that there may be multiple solutions that satisfy the conditions). The same method is applied to search for common solutions across all DLs (e.g., NAND, NOR, INV), determining the N/P recipe with the maximum overlap. Finally, the adjustment results are validated through correlation with WAT data from mass production.



2. SPICE simulates the target (RO & SIDD) with discrete points of N/P variation

3. Build the regression model for clear probability density in large generated samples

6. Correlate to the WAT data in MP

5. Find the NP recipe with the largest intersection for NAND, NOR, and INV using the same method

1. SPICE simulation for each delay-line (NOR: P-stack, NAND: N-stack, INV: normal)

4. Estimate the N/P where its values match the silicon mean (NOTE! There may be multiple points that can satisfy)

Fig. 11-9 S2S and N/P Bias Prediction

To enhance efficiency and accuracy in N/P offset estimation, we can train an MLP model to optimize the regression model's performance and improve offset correction precision, as shown in Fig. 11-10.

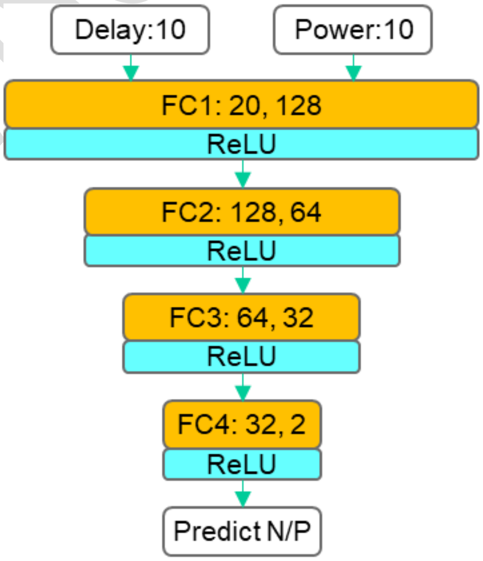
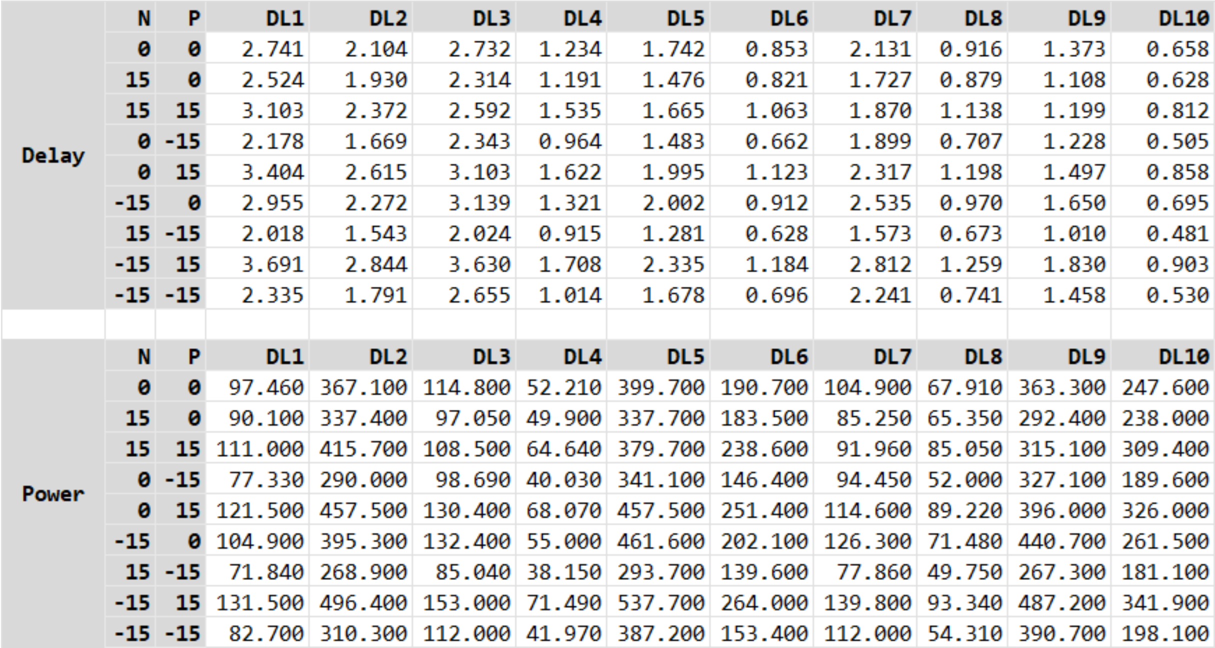


Fig. 11-10 MLP for N/P Bias Prediction

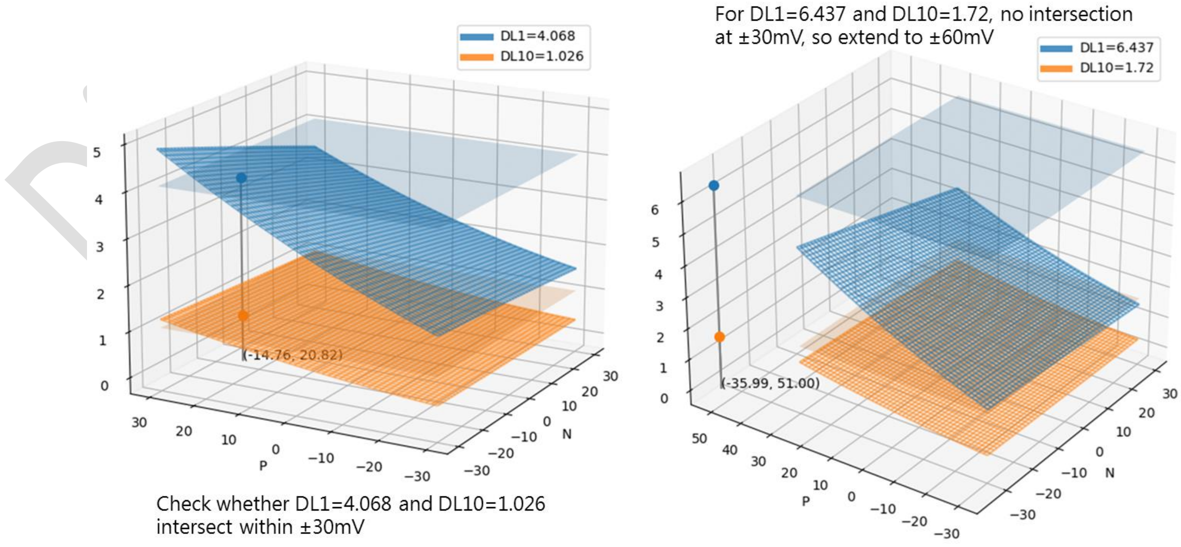
We start by conducting surface regression on a small set of discrete data for each DL. For instance, Table IV shows the SPICE simulation results of 9 N/P offset adjustment parameters, corresponding to different delay values in a 3×3 N/P grid. We then construct a regression model from these discrete data and interpolate to a higher resolution (e.g., 100×100 points) to generate a large sample set for training.

TABLE IV SPICE Simulation Grid with N/P Bias



	N	P	DL1	DL2	DL3	DL4	DL5	DL6	DL7	DL8	DL9	DL10
Delay	0	0	2.741	2.104	2.732	1.234	1.742	0.853	2.131	0.916	1.373	0.658
	15	0	2.524	1.930	2.314	1.191	1.476	0.821	1.727	0.879	1.108	0.628
	15	15	3.103	2.372	2.592	1.535	1.665	1.063	1.870	1.138	1.199	0.812
	0	-15	2.178	1.669	2.343	0.964	1.483	0.662	1.899	0.707	1.228	0.505
	0	15	3.404	2.615	3.103	1.622	1.995	1.123	2.317	1.198	1.497	0.858
	-15	0	2.955	2.272	3.139	1.321	2.002	0.912	2.535	0.970	1.650	0.695
	15	-15	2.018	1.543	2.024	0.915	1.281	0.628	1.573	0.673	1.010	0.481
	-15	15	3.691	2.844	3.630	1.708	2.335	1.184	2.812	1.259	1.830	0.903
	-15	-15	2.335	1.791	2.655	1.014	1.678	0.696	2.241	0.741	1.458	0.530
	N	P	DL1	DL2	DL3	DL4	DL5	DL6	DL7	DL8	DL9	DL10
Power	0	0	97.460	367.100	114.800	52.210	399.700	190.700	104.900	67.910	363.300	247.600
	15	0	90.100	337.400	97.050	49.900	337.700	183.500	85.250	65.350	292.400	238.000
	15	15	111.000	415.700	108.500	64.640	379.700	238.600	91.960	85.050	315.100	309.400
	0	-15	77.330	290.000	98.690	40.030	341.100	146.400	94.450	52.000	327.100	189.600
	0	15	121.500	457.500	130.400	68.070	457.500	251.400	114.600	89.220	396.000	326.000
	-15	0	104.900	395.300	132.400	55.000	461.600	202.100	126.300	71.480	440.700	261.500
	15	-15	71.840	268.900	85.040	38.150	293.700	139.600	77.860	49.750	267.300	181.100
	-15	15	131.500	496.400	153.000	71.490	537.700	264.000	139.800	93.340	487.200	341.900
	-15	-15	82.700	310.300	112.000	41.970	387.200	153.400	112.000	54.310	390.700	198.100

Finally, we verify the target and inference results. If no N/P intersection exists between multiple targets, the problem has no solution, as shown in Fig. 11-11. To account for solutions beyond the N/P variation's standard deviation, we may need to expand the value range to ±50mV or more.



Check whether DL1=4.068 and DL10=1.026 intersect within ±30mV

For DL1=6.437 and DL10=1.72, no intersection at ±30mV, so extend to ±60mV

Fig. 11-11 Search Range Exploration and Data Augmentation

In an RO instance with 10 DLs (DL1 to DL10), we can compare the target values (Z-axis plane) of DL1-DL10 against a broader regression surface to identify potential optimal N/P solutions, where the regression surface intersects the target values. As shown in Fig. 11-12, no intersection was found within the ±30mV range of the target delay values, indicating the need for data augmentation. To achieve this, the data range may need to be extended to at least ±60mV.

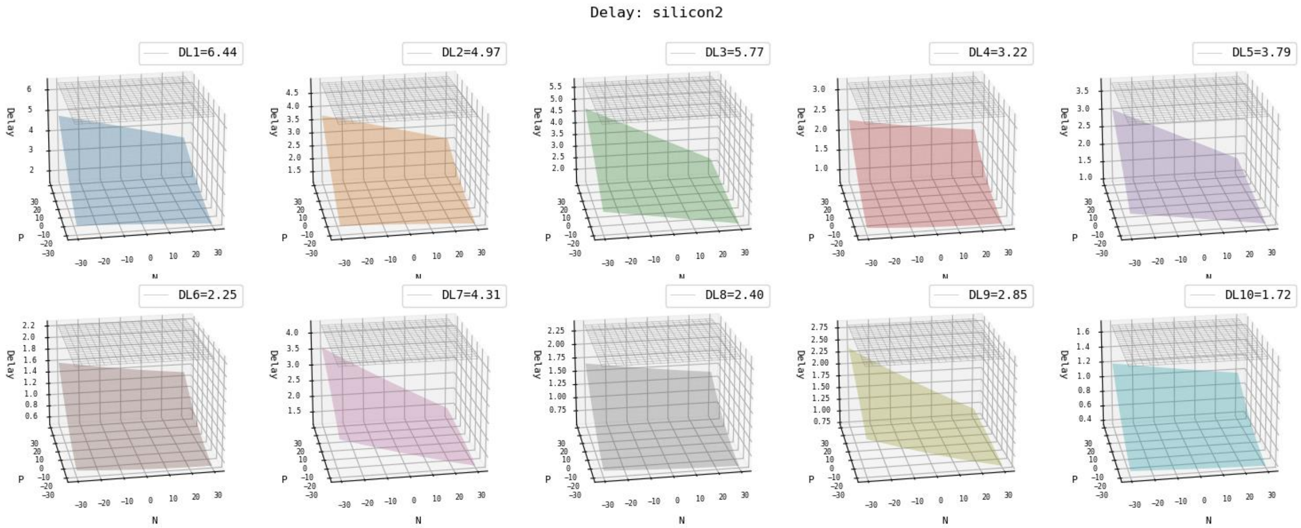


Fig. 11-12 Target N/P Feasibility Assessment

Fig. 11-13 shows the optimal N/P Bias solution that simultaneously satisfies the delay and power conditions of all 10 delay lines for the target Silicon2 in Table III.

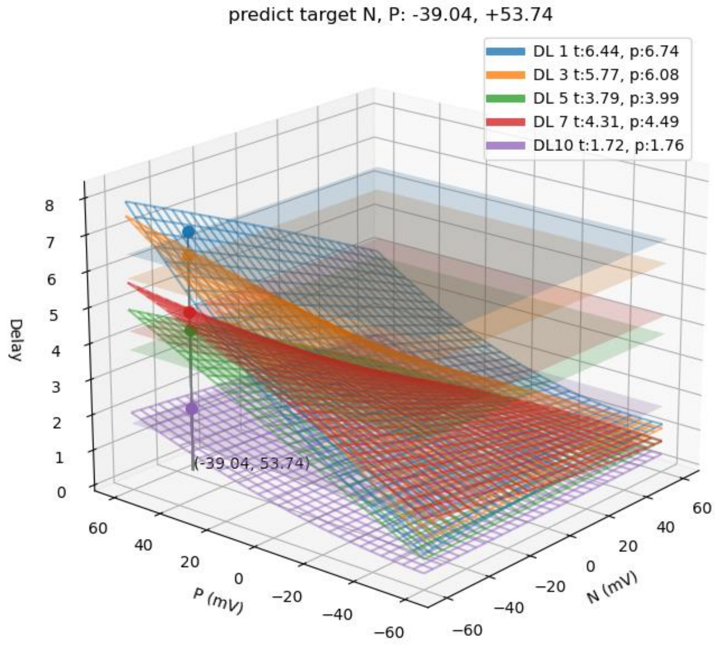


Fig. 11-13 One-shot SPICE-Silicon N/P Bias Prediction

11.3. High-Fidelity Generative Monte Approximation (HΣ-GMA)

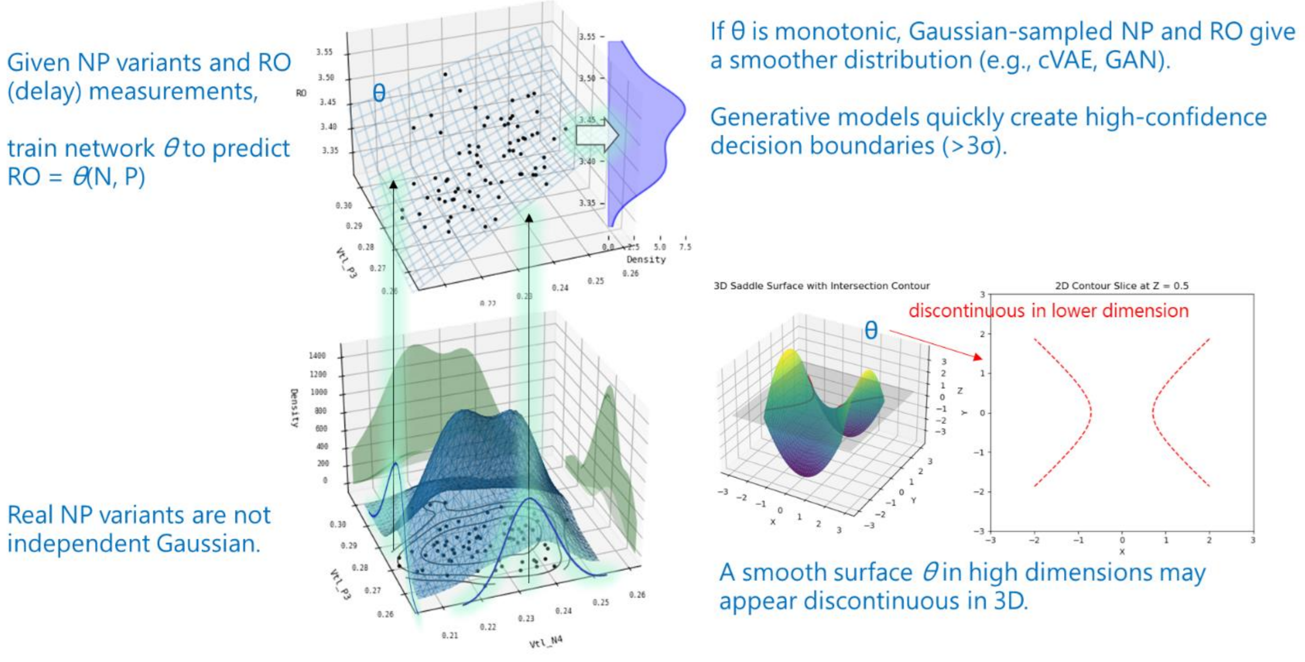
11.3.1. Limitations of Traditional Monte Carlo Methods

Traditional SPICE Monte Carlo simulations encounter challenges in identifying high-decision boundaries:

- **Sample Sparsity**: High-sigma regions, being low-probability events, are difficult to sample effectively, leading to insufficient or uneven distribution, hindering accurate estimation.
- **High Computational Demand**: Due to slow convergence, high-sigma analysis requires many samples, consuming significant computational resources and increasing simulation time, especially in large-scale circuits.
- **Parameter Correlation and Nonlinearity**: In high-sigma regions, complex parameter relationships and nonlinear behaviors increase uncertainty, making traditional methods ineffective.

11.3.2. Innovative Application of Generative Neural Networks

When a high-dimensional hypersurface (network θ) is projected to lower dimensions (e.g., 3D), topology deformation may occur, causing loss of continuity. As shown in Fig. 11-14, the conic hypersurface may lose its original structure. However, if the data's inherent relationships (e.g., monotonicity) are preserved, generative models like cVAE or GAN can learn smoother and more interpretable latent distributions.



Given NP variants and RO (delay) measurements,

train network θ to predict RO = θ(N, P)

Real NP variants are not independent Gaussian.

If θ is monotonic, Gaussian-sampled NP and RO give a smoother distribution (e.g., cVAE, GAN).

Generative models quickly create high-confidence decision boundaries (>3σ).

A smooth surface θ in high dimensions may appear discontinuous in 3D.

Fig. 11-14 Dimensionality Reduction and Smoothness Preservation

Given N/P variations and RO measurements, we train the network θ to predict RO=θ(N, P). However, actual N/P variations are not independent Gaussian distributions. If θ is monotonic, Gaussian N/P sampling can predict a smooth RO distribution that closely resembles the original distribution. This capability is embodied in generative models like cVAE and GAN. The advantage of generative models lies in their ability to quickly create high-confidence decision boundaries (>3σ), significantly enhancing prediction accuracy and stability.

Example 11-1 XOR2D1 Power Modeling and KDE Density

```
# Model the power distribution of XOR2D1 using a two-component Gaussian mixture.
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import gaussian_kde, norm

def sigma_percentage(self, sigma):
    '''return left and right % boundary based on the specified sigma value'''
    return stats.norm.cdf((-sigma, sigma))*100

def featureKDE(v, res=100):
    kde = gaussian_kde(v)
    t = np.linspace(min(v)-v.std(),max(v)+v.std(),res)
    p = kde(t)
    return t,p

low, high = 4.4e-6, 4.8e-6 # Data range
mean1, std1, weight1 = 4.57e-6, 3e-8, 0.4  # low-density Gaussian
mean2, std2, weight2 = 4.63e-6, 3e-8, 0.6  # high-density Gaussian

# Generate samples for each Gaussian
n_samples = 1000
n_samples1 = int(n_samples * weight1)
n_samples2 = n_samples - n_samples1

# Combine and clip the samples within the specified range
np.random.seed(42)
samples1 = np.random.normal(mean1, std1, n_samples1)
samples2 = np.random.normal(mean2, std2, n_samples2)
samples = np.concatenate([samples1, samples2])
```

Example 11-2 Gaussian Mixture Model (GMM)

```
#  Gaussian Mixture Model (GMM) Sample Generator
from sklearn.mixture import GaussianMixture

def GMM(v, n_samples=1000, n_components=2, scale=1e2):
    gmm = GaussianMixture(n_components=n_components, covariance_type='full', random_state=0)
    gmm.fit(v*scale)
    dv, _ = gmm.sample(n_samples=n_samples) # generated samples
    return dv/scale

fake = GMM(samples.reshape(-1,1), n_samples=len(samples)*50, scale=1e8).reshape(-1)

# statistical quantization
q1 = np.quantile(samples, q=sigma_percentage(3.0)/100)
q2 = np.quantile(fake, q=sigma_percentage(4.5)/100)
t1, p1 = featureKDE(samples)
t2, p2 = featureKDE(fake)
```

```
# visualization
plt.figure(figsize=(8,5))
plt.title('XOR2D1 Power Distribution')
plt.hist(samples, bins=50, density=True, alpha=0.4, color='skyblue', label=f'Histogram
({len(samples):,})')
plt.hist(fake, bins=50, density=True, alpha=0.4, color='orange', label=f'Fake ({len(fake):,})')
plt.plot(t1, p1, c='k', ls='--', lw=2, alpha=0.5, label='PDF (real)')
plt.plot(t2, p2, c='b', lw=3, alpha=0.5, label='PDF (fake)')
plt.axvline(q1[0], c='r', alpha=0.3)
plt.axvline(q1[1], c='r', alpha=0.3, label=f'Real 3$\sigma$: {q1[0]:.2e}, {q1[1]:.2e}')
plt.axvline(q2[0], c='b', alpha=0.3)
plt.axvline(q2[1], c='b', alpha=0.3, label=f'GMM 4.5$\sigma$: {q2[0]:.2e}, {q2[1]:.2e}')
plt.xlabel('Value')
plt.ylabel('Density')
plt.grid(which='major',linestyle='-',zorder=0, alpha=0.5)
plt.grid(which='minor',linestyle=':',zorder=0, alpha=0.5)
plt.minorticks_on()
plt.legend()
plt.tight_layout()
```

Example 11-1 simulates the power distribution of the XOR2D1 unit after 1,000 Monte Carlo runs using two Gaussian distributions. Example 11-2 applies a Gaussian Mixture Model (GMM) to the data, generating 50,000 samples. The program defines the means, standard deviations, and weights for the distributions, and generates samples accordingly. The original XOR2D1 power distribution and its 3σ boundary are compared with the 4.5σ boundary of the simulated data. A smooth probability density function (PDF) is then created using Kernel Density Estimation (KDE) for visualization and analysis.

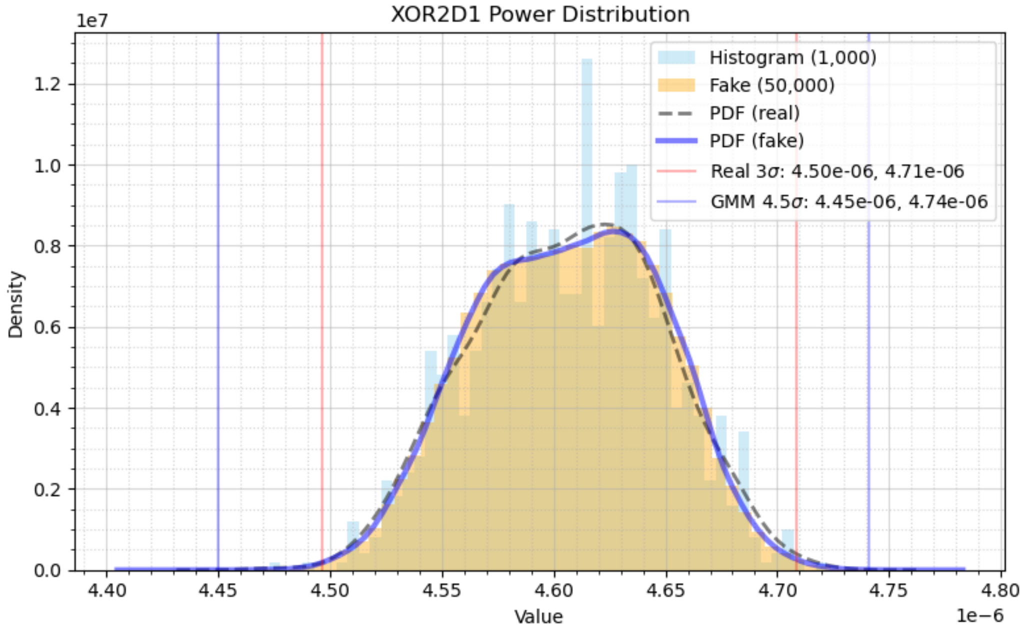


Fig. 11-15 XOR2D1 Power Probability Density Distribution

Chapter 12. Conclusion and Outlook

Fig. 12-1 illustrates the U-Net structure used to model semiconductor production and optimization. The ultimate goal is to enhance productivity and competitiveness by refining chip design through data-driven feedback to achieve optimal energy efficiency. By combining on-chip monitoring with machine learning and cross-dimensional data analysis, DTCO.ML offers an effective solution for optimizing both chip efficiency and productivity, making it a critical research focus in the field.

However, challenges persist, including the lack of standardized monitoring IPs and test data, barriers to cross-domain data interaction, and traditional modeling discrepancies. Sparse test data can introduce bias, while high-confidence simulations are time-consuming and require optimization. The integration of generative AI holds great promise, with DTCO.GenAI emerging as a key focus for future development.

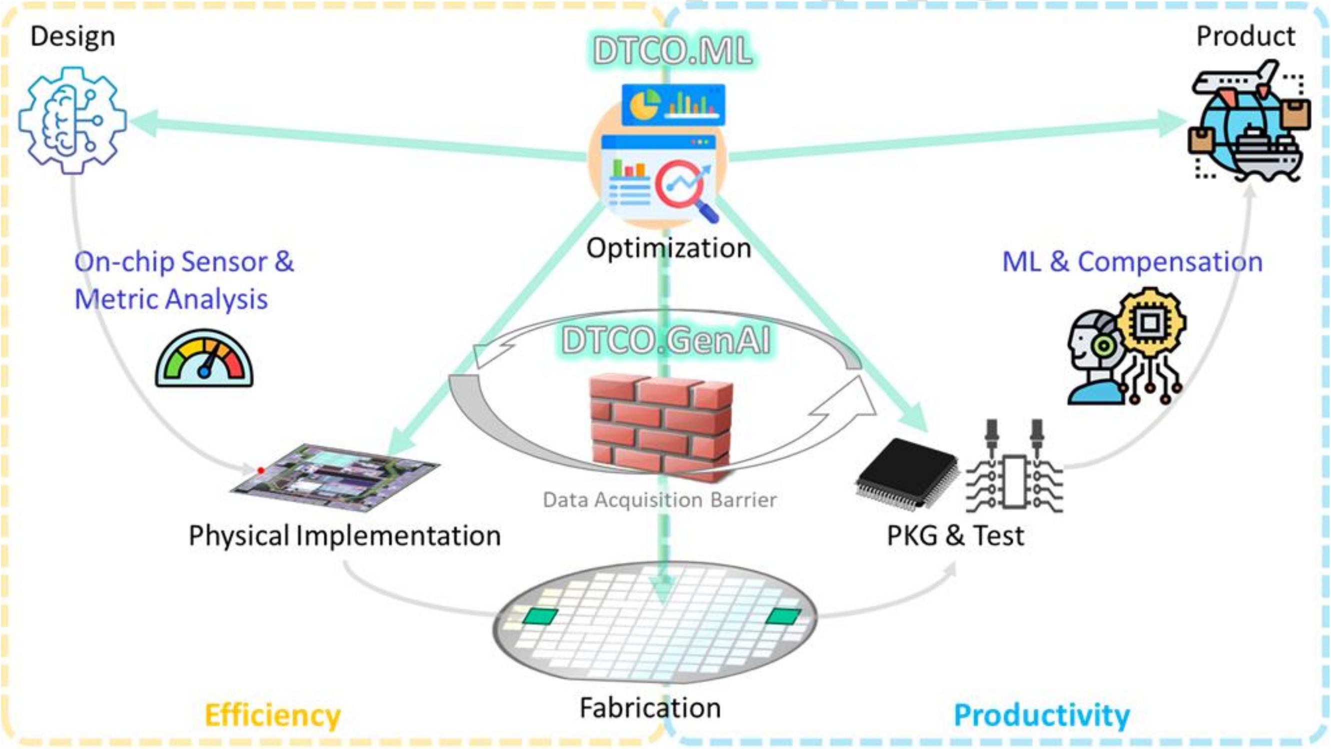


Fig. 12-1 Semiconductor Industry and Chip Design Innovation

12.1. AI-Enhanced DTCO: Revolutionizing Chip Design and Process Optimization (DTCO.ML™)

Integrating machine learning with DTCO enables precise process recipe guidance, reducing test costs and enhancing product quality. Key applications include:

- **Design and Process Recipes Optimization**: Machine learning models recommend optimal parameter combinations by analyzing their impact on chip performance.
- **Cell and Physical Design Flow Optimization**: Mathematical models facilitate dynamic adjustments between cell design and physical implementation, shortening development cycles.
- **Binning Strategy Generation and Optimization**: Predictive models based on historical data accurately forecast chip performance and optimize binning strategies. System-level compensation boosts yield, reliability, and competitiveness.

12.2. Generative AI-Driven Optimization (DTCO.GenAI™)

Machine learning models, trained on extensive chip data, analyze the relationships between process parameters, performance variability, and efficiency. Virtual Silicon Data, encompassing component performance, electrical characteristics, and process distribution, is crucial in DTCO. Key benefits include:

- **Overcoming Data Acquisition Barriers**: Virtual silicon reflects real process and production test data, compressing the data through generative models while ensuring data confidentiality, effectively addressing data acquisition issues.
- **Enabling Cross-Disciplinary Collaboration**: Providing a standardized, shareable data platform to facilitate efficient collaboration between chip design and process teams.
- **Enhancing Product Optimization**: Virtual data enables rapid iteration and optimization through performance prediction and process simulation.

12.3. EDA Innovation and Future Outlook

AI and machine learning are reshaping DTCO, enhancing efficiency and accuracy while paving the way for intelligent EDA tools. Virtual chip generation, leveraging GANs and Diffusion Models, precisely models microscopic data distributions and system-level variability, optimizing design margins and chip efficiency. Meanwhile, generative AI advances WAT data super-resolution and accelerates SPICE Monte Carlo approximations, reducing verification time and strengthening decision confidence. These breakthroughs are driving next-generation DTCO EDA tools, improving production efficiency, lowering costs, and optimizing quality, ushering in a smarter, more efficient semiconductor future.

Appendix

Open Source Resource List

DTCO Framework

https://github.com/dipsci/DTCO

https://pypi.org/project/DTCO/0.1.4

Google Colab: libMetric

https://colab.research.google.com/drive/1KIJpLU4oZM4lTJ8XgJ7e947N6QyIlp9H

https://colab.research.google.com/drive/16Y2aNTqC_v2vtJCqwqgpMn0LQk_eLxWA

https://colab.research.google.com/drive/1pj0fnW09y2Jh7XkOHn5fK60U7oN04Vus

Google Colab: Generated Model (Virtual Silicon)

https://colab.research.google.com/drive/1oags2cgVHDtQ2UECVjbFYKxL8RWvxjmq

https://colab.research.google.com/drive/1JNai0O36dDg2siIpaOwmtf6WveUa1z-k

Reference List

[1] LBR: Design Dependent Mega Cell Methodology for Area and Power Optimization, DAC, 2020

[2] Micro-Architecture Optimization for Low-Power Bitcoin Mining ASICs, VLSI-DAT, 2019

[3] Multi-Feature Data Generation for Design Technology Co-Optimization A Study on WAT and CP, FC, 2023

[4] Application of Generative Adversarial Networks for Virtual Silicon Data Generation and Design-Technology Co-Optimization: A Study on WAT and CP, IEEE-Access, 2024

[5] A Diffusion Model-Based Methodology for Multi-Feature Wafer Data Generation, IEEE-TSM, 2025

Glossary of Terms

A

AOCV : Advanced On-Chip Variation
AVS : Adaptive Voltage Scaling

B

BCE: Binary Cross Entropy
Binning : Chip classification by electrical metrics
BIST : Built-In Self-Test
BTC : Bitcoin

C

CCS : Composite Current Source (Liberty)
CDF : Cumulative Distribution Function
CG : Clock Gating
CP: Chip Probe
CPO : Co-Packaged Optics
CTS : Clock Tree Synthesis
cVAE : Conditional Variational Autoencoder

D

DDPM : Denoising Diffusion Probabilistic Model
DFF : D Flip-Flop
DFT : Design for Test
DL : Delay Line
DTCO : Design-Technology Co-Optimization
DUE : Device Under Extraction
DVFS : Dynamic Voltage and Frequency Scaling

E

EDA : Electronic Design Automation
ERA : Early Rail Analysis

F

FID : Frechet Inception Distance
FT : Final Test

G

GAN : Generative Adversarial Network
GenAI : Generative AI
GMM : Gaussian Mixture Model
GRO : Grid RO

H

HITL : Human-in-the-Loop

I

Isat : WAT Saturation Current

J

JS : Jensen-Shannon Divergence

K

KDE : Kernel Density Estimation

L

LEO : Low Earth Orbit Satellite
LDO : Low Dropout Regulator
LS : Least Squares Regression
LSC : LS Coefficient
LVF : Liberty Variation Format (Liberty)
LVS : Layout Versus Schematic

M

MBIST : Memory Built-In Self-Test
MCU : Microcontroller Unit
ML : Machine Learning
MVN : Multivariate Normal Distribution

N

NLDM : Non-Linear Device Model (Liberty)
N/P : N-type vs. P-type Semiconductor

O

OCM : On-chip Monitors
OCSB : On-chip Self-Binning
OCV : On-chip Variation

P

PDF : Probability Density Function
PL/PG : Pulse-Latch / Pulse Generator
PPA : Power, Performance, and Area
PVT : Process, Voltage, Temperature

R

Re-K : Re-characterization
RO : Ring Oscillator

S

S2S : SPICE-to-Silicon
SIDD : Static IDD, CP Leakage Current
SPICE : Simulation Program with Integrated Circuit Emphasis
SLT: System-level Test
STA : Static Timing Analysis

U

UID : Unique Identifier
ULE : Ultra-low Energy

V

Vsat : WAT Saturation Voltage
Vtl : WAT Threshold Voltage Low

W

WAT: Wafer Acceptance Test
WAT-SR : WAT Super Resolution
WID : Within Die