Team Details

Team Name: CodeX

SR. NO	ROLE	NAME	ACADEMIC YEAR
1	**Team Leader**	*Nachiket Mudhol*	*2025-26*
2	Member 1	*Durvesh Sathe*	2025-26
3	Member 2	*Parth Kulkarni*	*2025-26*
4	Member 3	*Mrunal Doifode*	*2025-26*

COLLEGE NAME

Marathwada Mitramandal’s College of Engineering, Karvenagar

TEAM LEADER CONTACT NUMBER

+91 9834827550

TEAM LEADER EMAIL ADDRESS

nachiketmudhol@gmail.com

Problem Statement Addressed:

Problem & Significance	Proposed Edge-AI Idea	Key Objectives & Impact
The Problem: 100s of precision steps generate TBs of image data (SEM/AFM) that overwhelm networks.	**The Idea:** Build an Edge-AI system for local wafer/die defect detection and classification.	**Detection:** Classify defects into predefined categories with high accuracy.
Why it's Significant: Microscopic defects directly impact **chip performance, yield, and reliability.**	**Technical Approach:** Use **lightweight AI/ML models** optimized for hardware like NXP eIQ.	**Performance:** Support real-time, high-throughout workflows with low compute needs.
The Bottleneck: Traditional centralized/manual review causes **high latency** and "wafer scrapping."	**Local Processing:** Analyze images at the source to reduce cloud/bandwidth dependency.	**Efficiency:** Faster response times and significantly reduced infrastructure costs.
Scalability Risk: Traditional centralized inspection systems struggle to meet real-time, high-volume production demands.	**Industry 4.0 Alignment:** Enables smart manufacturing with autonomous, on-device decision making.	**Outcome:** A scalable, deployable solution that improves overall **fab productivity.**

IDEA DESCRIPTION:

Key Concept & Approach

- •Automates defect detection in semiconductor manufacturing.
- •Uses deep learning to classify defects from microscopy images.
- •Reduces manual inspection time and human errors.
- •Runs on edge devices for real-time, on-line inspection.
- •Employs lightweight models for efficient and accurate processing.

Solution Overview

- •The system takes raw defect images as input and classifies them into predefined defect categories.
- •It reduces human dependency and speeds up inspection.
- •The approach improves manufacturing yield by **early and consistent defect detection**.
- •The solution is scalable and can be integrated into existing semiconductor inspection pipelines.

Class Design and Matrix:

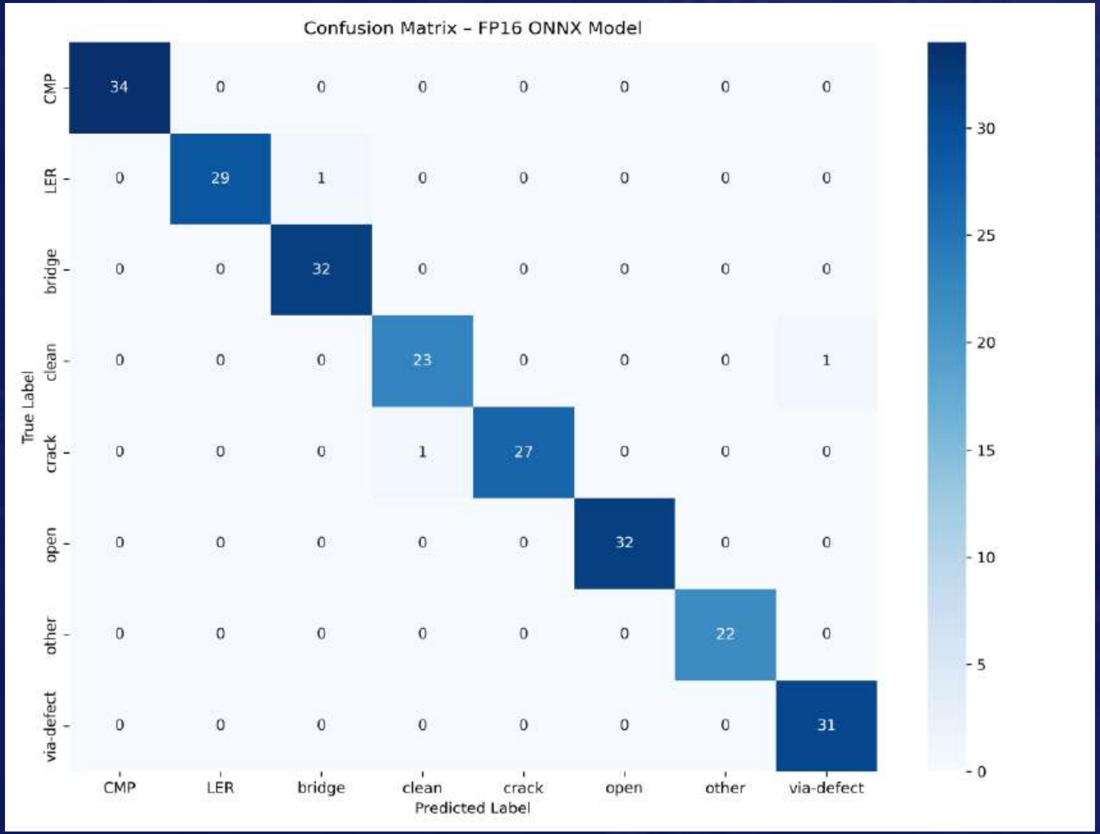
CLASS DESIGN:

- **Total images planned/current:** 1528
- **No. of classes:** 8 (6 defect + Clean + Other)
- **Class list:** Bridge +Clean +CMP +Crack +LER +Open +VIAS+ other
- **Class balance plan:** 191
- **Train/Val/Test split:** 70% /15%/ 15%
- **Image type:** Grayscale
- **Labeling method/source:** manual / public dataset / generated

Metrics on your test split

- **Accuracy:** 98-99%
- **Precision/Recall:** 99%

Confusion Matrix



True Label \ Predicted Label	CMP	LER	bridge	clean	crack	open	other	via-defect
CMP	34	0	0	0	0	0	0	0
LER	0	29	1	0	0	0	0	0
bridge	0	0	32	0	0	0	0	0
clean	0	0	0	23	0	0	0	1
crack	0	0	0	1	27	0	0	0
open	0	0	0	0	0	32	0	0
other	0	0	0	0	0	0	22	0
via-defect	0	0	0	0	0	0	0	31

Proposed Solution –

Data Processing
Image collection, preprocessing, and dataset splitting

Model Architecture
MobileNetV3-Small CNN for efficient edge inference

Training Strategy
Supervised learning with improvement and accuracy monitoring

Evaluation & Validation
Performance analysis using accuracy, F1-score, and confusion matrix.

Edge Optimization
FP16 precision conversion is applied to reduce model size.

Deployment
Real-time, low-power defect detection on edge devices

Innovation ,Uniqueness and Model details:

Key Innovation

- Edge-based AI for semiconductor defect detection (no cloud needed).
- Small, efficient MobileNetV3 model optimized for wafer/die images.
- FP16 precision conversion for fast, low-power edge deployment.
- Complete, deployment-ready pipeline from training to edge inference.

Competitive Advantage

- Ultra-low model size – 3.2MB
- Runs on low-power edge devices.
- Keeps high accuracy.
- Real-time inspection without GPU or cloud.
- Low latency and power use, suitable for fab environments.

Model Details

- **Architecture:** MobileNetV3
- **Training approach:** Transfer learning
- **Input size**: 224x224
- **Model size:** 3.2MB
- **Framework:** PyTorch

Impact and Benefits

Primary Impact

- Automated, accurate AI-based defect detection
- Less manual inspection and fewer human errors
- Real-time defect classification
- Edge-ready for factory-floor use
- Improved manufacturing yield and reliability

Quantifiable Outcomes

- ~98-99% classification accuracy on FP16 model
- Up to 2 times reduction in model size using FP16 weight-only quantization
- 30–50% faster inference on CPU-based edge devices
- Lower hardware and operational costs due to lightweight architecture
- Reduced inspection time, enabling higher throughput in Labs

Technology & Feasibility/Methodology Used

Implementation Strategy

- End-to-end **deep learning pipeline** for semiconductor defect classification.
- Uses **efficient CNN architecture (MobileNetV3-Small)** for edge feasibility.
- Model trained in **FP32**, then optimized using **ONNX and FP16 quantization.**
- Designed for **CPU-based edge deployment** with low latency and memory usage.

Software Architecture

•ONNX for portable deployment.
•FP16 precision conversion for model size reduction
•ONNX Runtime for edge inference

Hardware Components

•CPU-based edge/industrial PCs
•No GPU required for deployment
•Factory-floor compatible setup

Development Tools

•Python, PyTorch, Torchvision
•ONNX & ONNX Runtime
•OpenCV, NumPy, Scikit-learn,Matplotlib

GitHub & Video Link

GitHub Repository

https://github.com/NachiketMudhol/CodeX

Dataset ZIP

https://drive.google.com/file/d/1f-pAzTFqA0ocK5_EVW5oCbRZ-w36DOD1/view?usp=drive_link

Research and References

Research Background & Methodology

•Semiconductor defect detection requires fast and accurate classification of microscopy images.
•Deep learning–based convolutional neural networks (CNNs) provide higher accuracy than traditional inspection methods.
•A lightweight architecture (MobileNetV3) is selected for efficient edge inference.
•The model is trained using PyTorch on labeled defect images.
•The trained model is exported to ONNX format for cross-platform deployment.
•FP16 quantization is applied to reduce model size and improve inference efficiency while maintaining accuracy.

References & Citations

1 . MobileNetV3: Searching for MobileNetV3- https://arxiv.org/abs/1905.02244
2. PyTorch Official Documentation-https://pytorch.org/docs/stable/index.html
3. ONNX Runtime Model Optimization Guide-https://onnxruntime.ai/docs/performance/model-optimizations/