AFEKA | Electrical Engineering

Dar Eshel Epstein
Advisor: Dr. Binyamin Abramov

Real-Time License Plate Recognition System Using FPGA

1. Background

This project began with a simple goal: explore digit detection on FPGA. But it quickly grew into a full license plate recognition pipeline — integrating CPU-side detection using Nanodet and NCNN, and **FPGA-side OCR using a handcrafted CNN**. While tools like PyTorch and OpenCV were used for training and preprocessing, all real-time inference runs directly on the board, **without cloud, GPU, or external compute** — pushing every part of the system to its limit.

2. Objective

Develop a modular ALPR system where the CPU handles detection and preprocessing, and the FPGA performs OCR independently. This enables concurrent operation: while the FPGA processes one strip, the CPU prepares the next—maximizing throughput.
The **OCR engine is designed to be future-proof**: weights and biases are loaded from .mif files, and thresholds and filter size are adjustable without RTL changes.

3. Functional structure

The system has three components:

- CPU: Detects plates with NanoDet and builds digit strips with breakpoints.
- AHIM: Manages memory, OCR execution, and CPU-FPGA communication.
- OCR Engine: Performs sliding-window CNN inference using INT8 precision and per-class thresholds.
  All model parameters are externally configurable, making the FPGA pipeline modular and reusable for new datasets or future detection tasks.

All units communicate through burst-mode data and run in parallel.

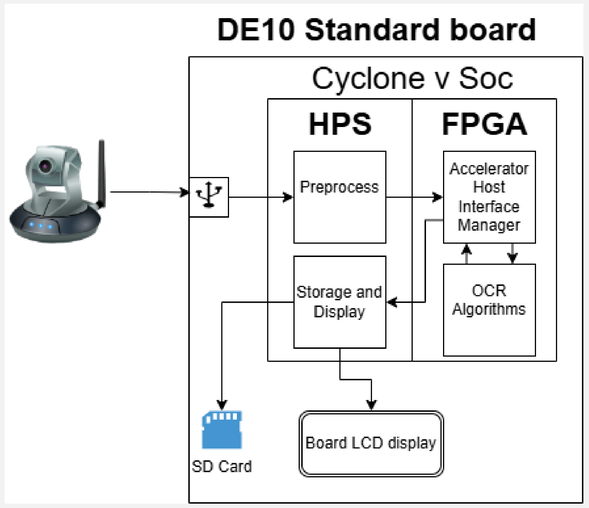

Figure 1: Full system overview
From Camera to LCD: ARM Preprocesses, FPGA Does AI – All On-Board.
Total Latency: 33ms.

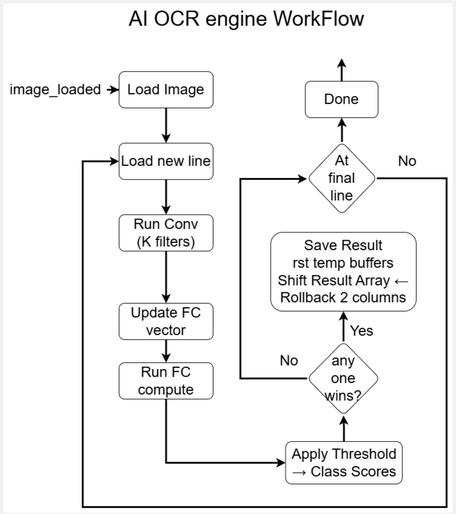

Figure 2: AI OCR Engine
877µs per Plate – A CNN Crafted in VHDL, Not TensorFlow.
(See Section 4)

4. Design and products

CPU: Dual core arm cortex A9 mpcore:
Runs custom Arch Linux. Prepares digit strips with breakpoints.

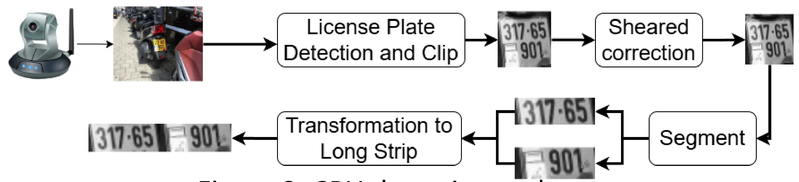

Figure 3: CPU detection and preprocess
ARM preprocessing on a CPU 5× weaker than a Raspberry Pi.

GitHub Repository

Live Demo
(103+ Hours Running)

AHIM: Accelerator hot interface manager
Stores data, triggers OCR per region.

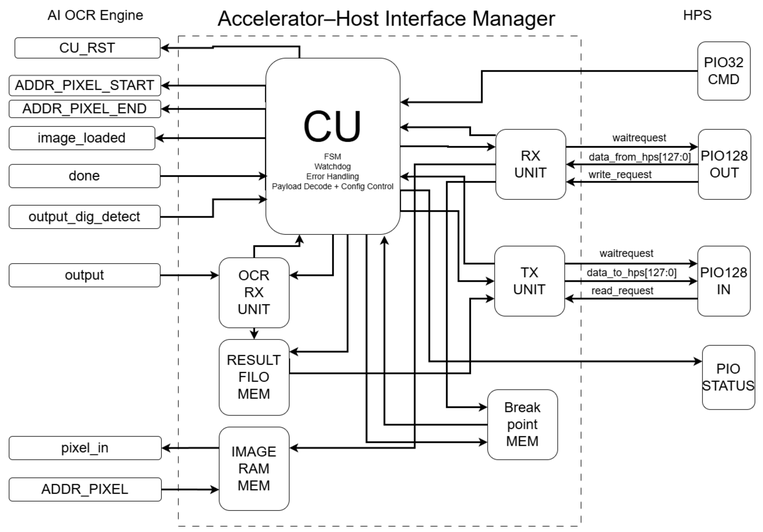

Figure 4: AHIM architecture
Not just a data bridge — a fault-tolerant, watchdog-enabled co-processor link.

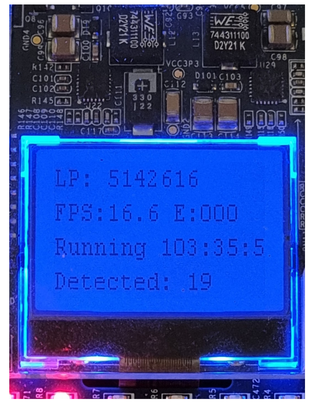

Figure 5: 100H+ Runtime

AI OCR Engine:
Performs sliding-window CNN inference using INT8 precision and per-class thresholds (see Figure 2). Model parameters are reconfigurable, enabling reuse for other tasks.

Training & Scalability:

- Real digits from 30k hand-labeled plates.
- Full plates synthetically generated (class balancing in progress).
- Modular Design: Swap models/thresholds via .mif files. Ready for real-world data.

5. Summary and conclusions

This single-student project delivers a robust working real-time ALPR prototype.

- Full plate OCR latency: ~877 µs
- FPS: 15-17
- Accuracy: 74 % digit-level and 40 % full-plate exact-match
  Digits come from real plates; full plates are synthetically combined. The pipeline is functional, modular, and ready for accuracy improvement through dataset expansion.

Development story