Hardware Acceleration Analysis for Driver Drowsiness Detection System

# Objective

The goal of this project is to implement a hardware-accelerated dense layer for a Driver Drowsiness Detection System (DDDS). This layer processes 136 input features (e.g., facial landmarks such as eye/mouth coordinates) and applies a multiply-accumulate (MAC) operation with pre-trained weights to classify driver alertness.

# RTL Module Functionality

The RTL module accepts 136 input features through `in\_data` and `in\_index`, stores them, and performs a signed multiply-accumulate operation using preloaded weights (`weights.mem`). It outputs the low 8 bits of the accumulator (`out\_data`) on each cycle during computation.

# Acceleration Comparison: Hardware vs Software

|  |  |  |
| --- | --- | --- |
| Metric | Software Model (e.g., Python) | Hardware Model (RTL Implementation) |
| Execution Medium | CPU executes general-purpose instructions (e.g., NumPy dot) | FPGA/ASIC logic gates execute specialized MACs |
| Cycle-by-Cycle Operation | Each MAC takes multiple CPU cycles | One MAC per clock cycle |
| Pipelining | Sequential looping | Synchronized accumulation per clock |
| I/O Handling | RAM read + compute | Preloaded + streamed into MAC unit |
| Parallelism | Limited by CPU cores | Parallel MAC units possible |
| Latency | High (~1–10 ms) | Low (~1.5 µs for 136 cycles at 100MHz) |
| Throughput | 1 result per function call | 1 result every 136 cycles |
| Power Efficiency | High (software stack overhead) | Low (direct hardware transitions) |

# Real-World Relevance in DDDS

In a real-time driver monitoring system, camera frames must be processed at ~30–60 frames per second. Each frame’s feature vector must be classified rapidly. This hardware MAC layer processes the feature vector in around 150 clock cycles, allowing inference in under 2 microseconds, enabling effective real-time alert systems.

# Next Steps for Full Acceleration

- Add activation functions (e.g., ReLU, Sigmoid) to the output.

- Expand MAC logic to multiple output neurons for multi-class classification.

- Integrate real-time input streaming (e.g., AXI, DMA).

- Profile performance using `$time` or synthesis timing reports.

# Conclusion

This RTL-based MAC accelerator significantly reduces inference time compared to a software-based approach, making it suitable for embedded AI applications such as Driver Drowsiness Detection in automotive safety systems.