Design and ASIC Implementation of an Edge-AI SoC Subsystem for Detecting Driver Unresponsiveness and Requesting a Minimum Risk Manoeuvre

Roobashri S, Sanjeykrishna V, Subhashini N, Varsha V R, and Renita Pearlin

Department of Electronics Engineering (VLSI Design & Technology), Sri Shakthi Institute of Engineering and Technology (Autonomous), Coimbatore – 641 062, Tamil Nadu, India

Abstract—Driver drowsiness and loss of responsiveness are major causes of road accidents, and a warning buzzer alone cannot protect a vehicle whose driver has stopped responding. This paper presents a digital hardware subsystem for Advanced Driver Assistance Systems (ADAS) that takes precomputed class scores (logits) from an upstream Convolutional Neural Network (CNN), judges their reliability, cross-checks them with an independent fatigue indicator, and escalates from a warning to a Minimum Risk Manoeuvre (MRM) request if the driver does not respond. The design contains a CNN confidence unit, a PERCLOS-inspired fatigue unit using a 32-frame sliding window, a safety arbiter and a risk-adaptive finite state machine (FSM) whose response timeout shortens as obstacle and lane-drift risk rises. The Verilog RTL was verified with a 12-scenario testbench in Synopsys VCS and Verdi. Synthesis with Synopsys Design Compiler on the SAED32 library at a 4 ns clock (250 MHz) gave 582 cells, a total cell area of 1726.40 units, 195.69 µW dynamic power and a setup slack of 0.00 ns. A Synopsys IC Compiler II analysis reported 667 cells and 79.38% core utilization. These are pre-layout and placement-stage estimates, and their limitations are stated explicitly.

Keywords—ADAS, Edge AI, driver drowsiness detection, Minimum Risk Manoeuvre, PERCLOS, finite state machine, Verilog HDL, ASIC synthesis

I. INTRODUCTION

Advanced Driver Assistance Systems (ADAS) combine sensors, artificial intelligence and control hardware to reduce the chance and severity of road accidents. Driver monitoring is different from most ADAS functions because the hazard is the driver, not the road: if the driver is incapacitated, every other assistance function depends on a human who is no longer reacting. A conventional drowsiness alert sounds a buzzer, but if the driver does not respond the vehicle simply continues.

Edge AI, in which inference and decision making are performed locally in the vehicle, avoids network dependence and reduces response latency. Dedicated digital hardware adds deterministic timing, low power and small area, and its behaviour can be analysed at design time. These properties suit a safety function that must act on its own.

This paper presents a hardware subsystem, named mrm_top, that sits after a driver-monitoring CNN. It does not implement the CNN. It receives the CNN's logits and produces warning and emergency requests. The contributions are:

- A confidence unit that accepts a CNN classification only when the margin between the two largest logits exceeds a programmable threshold.
- A PERCLOS-inspired fatigue unit and a safety arbiter that cross-check the CNN so that an unconfirmed UNRESPONSIVE classification is not accepted blindly.
- A six-state FSM that escalates from warning to an MRM request, with a response timeout that adapts to obstacle and lane-drift risk.
- RTL verification, ASIC synthesis (area, power, timing) and an ICC2 placement-stage analysis, reported with their limitations.

II. BACKGROUND AND RELATED WORK

Camera-based driver monitoring systems commonly track eye closure, blink behaviour, gaze and head pose, and raise a visual or audible alert when drowsiness is suspected. PERCLOS, the proportion of time the eyes are closed over a window, is a widely used fatigue measure. At higher levels of driving automation, the concept of a minimal risk condition requires the system to bring the vehicle to a safer state if the driver does not take over [1].

Neural-network inference for driver monitoring is increasingly executed on dedicated Edge-AI hardware near the sensor. A neural network classification should not be treated as ground truth, because lighting, camera noise and brief eye closure can all produce wrong predictions. A hardware safety layer after the network can apply simple, verifiable rules before any vehicle action is requested, which supports functional-safety arguments such as those in ISO 26262 [2]. The proposed subsystem is such a layer: it combines confidence checking, an independent fatigue indicator, response monitoring and risk-dependent escalation in one small synchronous block.

III. PROPOSED ARCHITECTURE

A. Overview

The design consists of four functional modules (conf_calc, perclos_calc, safety_arbiter and risk_mrm_controller) connected by the top module mrm_top (Fig. 1). It is fully synchronous with a single

clock and an active-low asynchronous reset. The CNN logits (logit0, logit1, logit2: AWAKE, DROWSY, UNRESPONSIVE; 8-bit signed) and driver observations (eye_closed, head_down) enter with a frame_valid flag. Obstacle and lane-drift risk are supplied as 2-bit buckets.

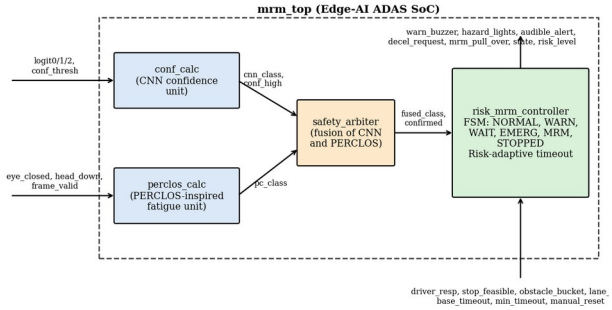


Fig. 1. Block diagram of the mrm_top subsystem.

B. CNN Confidence Unit (conf_calc)

The predicted class is the one with the largest logit. The unit also finds the second-largest logit and computes the margin between them. The margin is compared with a 9-bit threshold conf_thresh, and the class cnn_class and the flag conf_high are registered. For example, logits of −30, −20 and 70 give UNRESPONSIVE, with a margin of 90.

C. PERCLOS-Inspired Fatigue Unit (perclos_calc)

The unit keeps a sliding window of the last 32 valid frames, each recording whether the eyes were closed or the head was down. The number of set entries is compared with two 6-bit thresholds, th_drowsy and th_unresp, to produce pc_class. A single blink changes one entry and does not change the class, whereas prolonged closure or head-down posture fills the window and raises it.

D. Safety Arbiter (safety_arbiter)

The arbiter combines cnn_class and pc_class and produces fused_class, the flags confirmed and disagree, and fused_valid. Its purpose is to prevent an unconfirmed UNRESPONSIVE classification from the CNN alone from triggering the emergency sequence.

E. Risk-Adaptive MRM Controller

The controller is a six-state FSM (Table I). If the driver responds while in WARN or WAIT, it returns to NORMAL. If the response timeout expires, it enters EMERG, then MRM, and finally STOPPED once the vehicle has stopped. A manual reset returns it to NORMAL. The decel_request output is high only in MRM, audible_alert is high in EMERG, MRM and STOPPED, and mrm_pull_over captures stop_feasible during EMERG and is held until the controller returns to NORMAL. The RTL generates request signals only; steering and braking are left to a vehicle-level controller.

TABLE I. FSM states and outputs

State	Code	Buzzer	Hazard	Audible	Decel
NORMAL	000	0	0	0	0
WARN	001	1	0	0	0
WAIT	010	1	0	0	0
EMERG	011	0	1	1	0
MRM	100	0	1	1	1
STOPPED	101	0	1	1	0

F. Risk-Adaptive Timeout

The risk level is the larger of the obstacle bucket and the lane-drift value, and the effective timeout is the base timeout shifted right by the risk level, limited below by a minimum:

$$risk = \max(\text{obstacle_bucket}, \text{lane_drift})$$

$$effective_timeout = \max(\text{base_timeout} \gg risk, \text{min_timeout})$$

Table II shows the timeout halving with each risk level for a base timeout of 40 clock cycles. A higher risk therefore gives the driver less time to respond before the controller escalates.

TABLE II. Risk-adaptive timeout (base = 40)

Obstacle	Lane drift	Risk	Timeout
0	0	0	40
1	0	1	20
0	2	2	10
3	1	3	5

IV. IMPLEMENTATION AND VERIFICATION

The design is written in synthesizable Verilog (modules conf_calc, perclos_calc, safety_arbiter, risk_mrm_controller and mrm_top). It was simulated with Synopsys VCS using a testbench with 12 scenarios that drive the DUT inputs and observe the outputs, and the waveforms were dumped to an FSDB file and inspected in Verdi (Fig. 2). The FSM state signal takes the values 0 (NORMAL), 2 (WAIT), 4 (MRM) and 5 (STOPPED) over the roughly 5000 ns run, the effective timeout changes with the risk level (values 20 and 40 appear as 14 and 28 in hexadecimal), and the timeout and responded signals appear as one-cycle pulses. The waveform shows the escalation sequence and the risk-dependent timeouts; it does not by itself prove that all 12 scenarios pass.

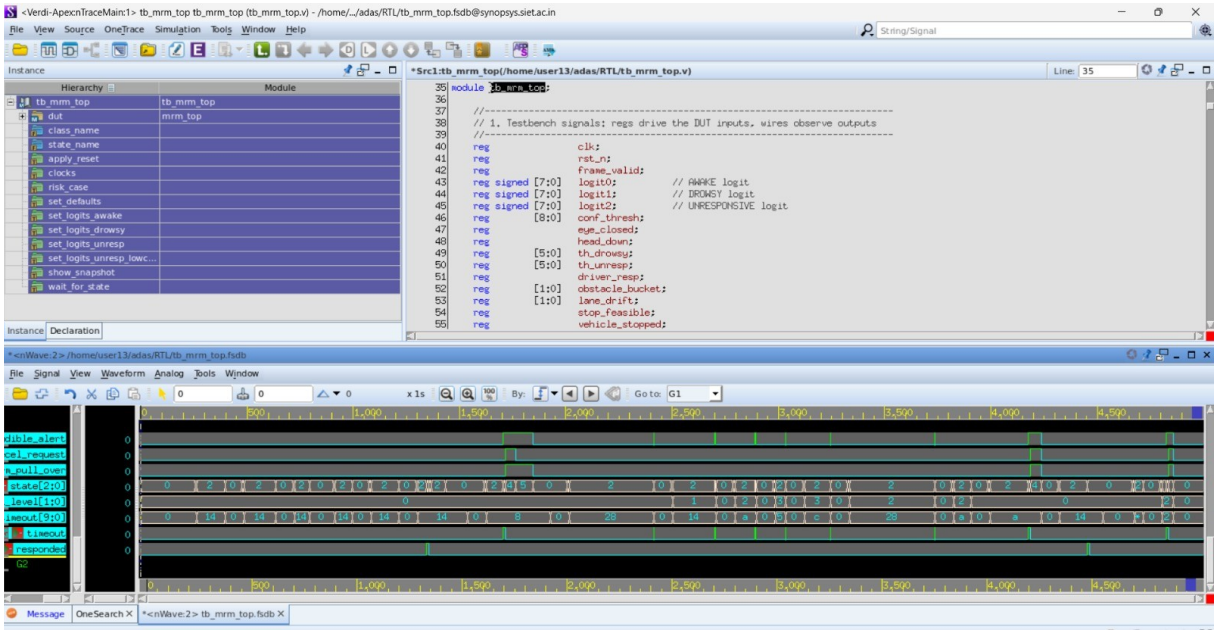


Fig. 2. Verdi view of the testbench hierarchy and the controller waveforms (state, risk level, effective timeout, timeout and responded).

V. SYNTHESIS AND PHYSICAL DESIGN RESULTS

A. Synthesis

The design was synthesized with Synopsys Design Compiler (V-2023.12-SP5) using the SAED32 HVT, LVT and RVT standard-cell libraries (ff0p85v125c corner, 0.85 V) with a 4 ns clock period (250 MHz). The mapped netlist has 112 ports, 582 cells (504 combinational, 78 sequential) and no macros (Fig. 3). Table III lists the results. Cell area is 68% combinational and 32% sequential.

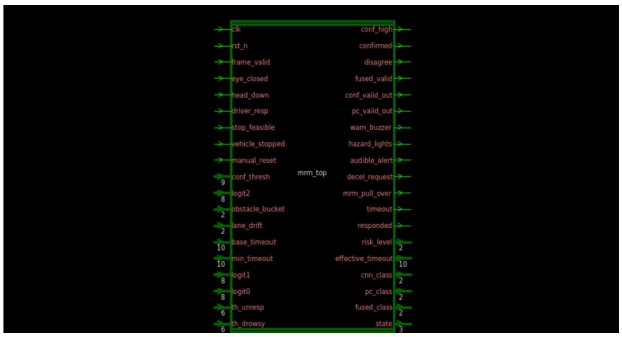


Fig. 3. Top-level symbol of the synthesized mrm_top.

TABLE III. Design Compiler results

Parameter	Value
Cells (comb. / seq.)	582 (504 / 78)
Combinational area	1170.84
Sequential area	555.56
Total cell area	1726.40
Net interconnect area	388.84
Total area	2115.24
Dynamic power	195.69 µW
Leakage power	1.3592 mW
Total power	1.5548 mW
Setup slack	0.00 ns (met)

Area values are in the library's area unit. The critical path runs from logit0[1] through the logit comparison, margin subtraction and threshold comparison to the conf_high register: the arrival time is 3.97 ns against a required time of 3.97 ns (4.00 ns minus 0.03 ns setup), so the path meets the constraint with no positive margin. Dynamic power is 96% cell internal power; by group, combinational logic accounts for 55.69% of the total, registers 30.12% and the clock network 14.19%.

B. Physical Design Analysis (ICC2)

The synthesized design was analysed in Synopsys IC Compiler II (U-2022.12-SP3). Table IV summarizes the reported results. No floating nets and no maximum-transition or maximum-capacitance violations were reported. The reports do not include clock-tree synthesis, routing, physical verification or GDSII results, so this is a placement-stage analysis only.

TABLE IV. ICC2 results

Parameter	Value
Standard cells (comb. / seq.)	667 (589 / 78)
Total cell area	1907.35
Core area	2402.68
Core utilization	79.38%
Total wire length	10440.06 µm
Input-to-register slack	−0.00 ns (1 path)
Register-to-output slack	+0.45 ns
Dynamic / leakage power	97.0 / 41.8 µW

VI. DISCUSSION AND LIMITATIONS

The design is compact, and the simulation shows the intended escalation behaviour. The following limitations apply:

- The confidence calculation is timing-critical. The synthesis slack is 0.00 ns and ICC2 reports one input-to-register path at about −0.00 ns, so the 4 ns clock has no margin. Pipelining the logit comparison and margin subtraction is the obvious remedy.
- Only the single worst path of the synthesis timing report is available; a full timing summary with all path groups and hold analysis has not been produced.
- The Design Compiler power estimate used low analysis effort without a simulation activity file. The ICC2 power analysis used only six simulation cycles, 0.75 V instead of 0.85 V, and different tool versions, so the two power figures should not be compared as a before-and-after experiment. All power results are preliminary.
- The cell area rose by about 10.5% from synthesis to ICC2, but the reports alone do not identify the cause.
- The design was verified on 12 directed scenarios using precomputed logits. It has not been evaluated with a real CNN or real driver data, and the thresholds were not tuned for detection accuracy.

VII. CONCLUSION AND FUTURE WORK

This paper presented a digital hardware subsystem that checks the confidence of a driver-state CNN, cross-checks it with a PERCLOS-inspired indicator and escalates through warning, emergency and MRM states with a risk-adaptive response timeout. Simulation showed the intended state sequence and timeouts, and synthesis gave 582 cells, 1726.40 area units and 195.69 µW dynamic power at 250 MHz, with no timing margin on the confidence path. Future work includes timing optimization of that path, completing clock-tree synthesis, routing and physical verification, power analysis with simulation activity and post-layout parasitics, integration with the CNN, and a formal ISO 26262-oriented safety analysis.

REFERENCES

[1] SAE International, “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles,” SAE J3016.

[2] ISO 26262, “Road vehicles – Functional safety.”

[3] IEEE Std 1364-2005, “IEEE Standard for Verilog Hardware Description Language.”

[4] N. H. E. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*. Pearson.

[5] J. M. Rabaey, A. Chandrakasan and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*. Pearson.

[6] Synopsys, *Design Compiler User Guide*; *IC Compiler II User Guide*; *VCS and Verdi User Guides*.