Amygdalic Decision Architecture for Embedded AI Systems

Hardware-Enforced Action-Space Determinism via Distilled Perception and Circuit-Level Arbitration

Shivam Sudhakar

Independent Researcher

March 10, 2026

Abstract

This paper describes a hardware architecture that moves AI safety constraints out of software and into circuit topology — and analyzes the cost inversion that makes this viable on sub-$15 NPU hardware. A distilled vision model, structured as N parallel narrow classifiers, replaces the vision-detectable subset of conventional discrete sensor arrays at significantly lower bill-of-materials cost. Classifier outputs are routed to a dedicated hardware node — the *amygdala circuit* — implemented as a sparse FPGA-based lookup table that maps perception state to one element of a pre-wired actuator option set $\mathcal{O}$. The resulting guarantee is precisely stated: the actuator command space is deterministically bounded by circuit topology regardless of inference-layer behavior, under the assumption that the FPGA and communication bus function correctly. This is not a claim that selected actions are always correct; it is a claim that no action outside $\mathcal{O}$ is physically reachable. We show this distinction is structurally compatible with hardware certification frameworks (IEC 61508, DO-178C, IEC 62304) in a way that software-only AI systems are not, and analyze the cost inversion that makes deployment viable at scale. Open implementation questions regarding sparse LUT construction, confidence propagation, DAG guarantee extension, and option-set governance are stated precisely.

Keywords: knowledge distillation, embedded inference, bounded action selection, hardware safety interlock, FPGA arbitration, autonomous systems, NPU

1 Introduction

The standard way to make a device sense its environment is to add hardware for every condition it needs to detect. One sensor per variable. PIR for motion, IR for proximity, separate chips for light and temperature — each requiring its own interface, calibration, and firmware logic. Edge cases compound this: a child behaving like an adult, a component occluded at an unusual angle, variable lighting. Each one is either another hardware component or another firmware rule. The cost structure scales badly in both directions.

A distilled vision model on a $15 NPU changes this for vision-detectable conditions. One camera. One inference chip. Robustness to lighting variance, occlusion, and unusual angles is handled in training data, not hardware. The sensor array problem becomes a classification problem, and classification problems are now cheap.

Putting AI in a physical actuation system creates a different problem. Software safety constraints — output filters, alignment training, reward shaping — cannot be formally verified. Any software layer is reachable from the model's execution environment. A sufficiently corrupted or adversarially perturbed model can in principle reach any software state. This makes software-only AI unsuitable for regulated physical actuation: you cannot inspect your way to a certification.

The fix is not better software. It is moving the constraint out of software entirely. If the AI selects from options that are physically pre-wired into a circuit, it cannot produce an output that does not exist. Not because of a rule. Because there is no signal path there.

We propose an architecture that implements this directly. The perception layer is a distilled classifier grid running on a low-cost NPU. The decision layer is a dedicated hardware circuit — the amygdala circuit — that maps perception outputs to one of M physically pre-wired actuator states. The AI perceives; the hardware selects which pre-wired option to activate. No signal path from the inference layer to any actuator bypasses the amygdala node. We state precisely what this guarantees and what it does not.

2 Related Work

Hardware Safety Interlocks. PLCs constraining actuators from control computers have existed in industrial settings for decades [7]. The gap this paper addresses is that nobody has applied the same principle formally to an AI inference layer — or combined it with distilled perception to make the full stack economically viable on sub-$15 silicon.

Action Masking and Constrained MDPs. Action masking in reinforcement learning restricts the selectable action set based on current state [9, 8]. The limitation of software action masking is not theoretical. No runtime filter can be formally verified against all model failure modes because the filter and the model share an execution environment. A hardware constraint does not share an execution environment with anything. That is the difference this architecture exploits.

Behavior Trees. Behavior trees [10] provide hierarchical task decomposition for robotic systems, structurally related to the decision DAG in Section 5. Behavior trees are software structures. The DAG in this work is partially implemented in hardware at the terminal action layer.

Distilled Edge Models. Knowledge distillation [1] trains compact student models on outputs of larger teacher models. MobileNet [3] and SqueezeNet [4] demonstrated competitive vision at embedded compute cost. Hailo-8 [13] and RK3588 [14] NPU platforms demonstrate competitive inference below $20 in silicon. We leverage this to replace the vision-detectable subset of conventional sensor arrays.

Multitask and Mixture-of-Experts Models. The classifier grid is related to multitask learning [11] and mixture-of-experts architectures [12]. The distinction is that rows are deliberately narrow and independently trained to minimize parameter coupling, enabling

per-row replacement without full retraining — which matters more than representational efficiency for the deployment model in Section 6.

3 System Architecture

The system comprises four layers. Signal flow:

$$\text{Camera} \rightarrow \text{Distilled Classifier Grid (NPU)} \xrightarrow{\text{SPI, 8-bit argmax}} \text{Amygdala Circuit (FPGA LUT)} \rightarrow \text{Pre-wired Actuator Options}$$

Latency budget: NPU inference ∼10 ms, SPI transmission <1 ms, FPGA LUT lookup <0.1 ms, total pipeline ∼11–15 ms.

Figure 1 shows the concrete implementation sketch.

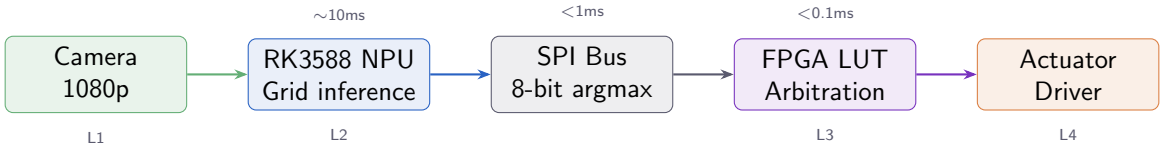

Figure 1: Concrete implementation. Camera feeds RK3588 NPU for grid inference. Argmax output transmitted over SPI. FPGA LUT performs arbitration. Total pipeline latency: ∼11–15 ms.

3.1 Layer 1 — Sensor Input

A camera module provides the primary input. This architecture addresses vision-detectable conditions — object presence, motion state, subject classification, spatial geometry. It does not claim to replace non-optical sensors (temperature, gas, pressure, vibration, LiDAR). In many product categories, vision-detectable conditions constitute the majority of sensing cost; this is the targeted scope.

3.2 Layer 2 — Distilled Classifier Grid

Definition 1 (Classifier Grid)**.** *A classifier grid G consists of N independent narrow classifiers $\{c_1, \ldots, c_N\}$, where c_i maps image input x to a probability distribution over K_i discrete classes:*

$$c_i : \mathcal{X} \rightarrow \Delta^{K_i}$$

The joint output is $\mathbf{p} = (p_1, \ldots, p_N)$. The argmax of each p_i is encoded as an 8-bit integer and transmitted over SPI. In case of equal maximum probability across classes in any row, a fixed tiebreaking rule — lowest class index wins — is applied deterministically before encoding.

Each row c_i is independently distilled from a larger teacher vision model. Row c_1 might classify object type from 500 classes; row c_2 classifies lighting condition from 5 classes; row c_3 classifies age bracket from 6 classes. A single multitask model would couple rows through shared representations, making per-task updates expensive. Independent rows trade some representational efficiency for modularity — per-row retraining with no effect

on other rows. For the deployment model in Section 6, modularity matters more than efficiency.

On a Hailo-8 NPU (26 TOPS, $10–15 at volume), 50 narrow classifiers with ∼1M parameters each run in approximately 5–15 ms depending on input resolution, within real-time requirements for most physical actuation contexts (>30 ms loop period).

3.3 Layer 3 — The Amygdala Circuit

Definition 2 (Amygdala Circuit)**.** *Let $\pi : \prod_{i=1}^{N}[K_i] \to \prod_{i=1}^{N}[K_i]$ be a fixed priority projection defined at manufacture. The priority ordering is a total order $<$ on classifier rows $\{1, \ldots, N\}$: for any two rows $i < j$ with conflicting actuation triggers, row i takes precedence. Formally, π selects the argmax tuple entry from the highest-priority row whose output has a defined LUT entry in f. The amygdala circuit implements:*

$$A = f \circ \pi$$

where $f : \prod_{i=1}^{N}[K_i] \to \mathcal{O}$ is a sparse lookup table, and $\mathcal{O} = \{o_1, \ldots, o_M\}$ is the set of physically pre-wired actuator options. Priority resolution is part of LUT construction, not a separate runtime operation.

Implementation. The amygdala circuit is an FPGA lookup table. The argmax vector, encoded as a fixed-width integer over SPI, passes through priority projection π and indexes into sparse LUT f, outputting a drive signal for one of M actuator circuits.

Sparse LUT construction. The full input space has $\prod K_i$ possible combinations — infeasible as a flat table for large N. The LUT is sparse: only combinations corresponding to meaningful actuation states have explicit entries. All unmapped combinations route deterministically to $o_{\text{safe}} \in \mathcal{O}$, typically `escalate_to_human` or `do_nothing`. Algorithm 1 formalizes construction.

Algorithm 1 Sparse LUT Construction

Require: Priority ordering $<$ on rows $\{1, \ldots, N\}$, domain constraint set $\mathcal{C}$
Ensure: Sparse LUT f, default o_{safe}
  for each actuation-relevant combination $\mathbf{y} \in \mathcal{C}$ **do**
    $\mathbf{y}' \leftarrow \pi(\mathbf{y})$ {Apply priority projection}
    Assign $f(\mathbf{y}') \leftarrow o_m$ for appropriate $o_m \in \mathcal{O}$
  end for
  for all remaining combinations $\mathbf{y} \notin \mathcal{C}$ **do**
    Assign $f(\mathbf{y}) \leftarrow o_{\text{safe}}$
  end for
  return f

3.4 Layer 4 — Pre-Wired Actuator Options

Terminal outputs are physical circuit states. Option o_m corresponds to a specific actuator drive signal, hardwired at manufacture. FPGA output pins connect directly to actuator driver circuits. There is no software-accessible bus between the amygdala circuit and actuator outputs in the primary signal path.

4 Formal Guarantees

Assumptions.

A1. FPGA logic functions correctly (no bit-flip faults, no gate failures).

A2. SPI bus transmits argmax values without corruption.

A3. Actuator driver circuits respond correctly to FPGA output signals.

Hardware fault analysis covering A1–A3 failure modes, including Worst-Case Execution Time (WCET) analysis of the SPI-to-FPGA arbitration path required for formal real-time certification, is outside the scope of this technical note and identified as future work in Section 8.

Proposition 1 (Action-Space Determinism)**.** *Under assumptions A1–A3, for any input $x \in \mathcal{X}$ and any state of the classifier grid:*

$$\text{output} \in \mathcal{O}$$

This holds regardless of model failure mode, adversarial input, weight corruption, or distribution shift in the inference layer.

Proof sketch. The only signal paths from the NPU to any actuator pass through the FPGA LUT. The composed mapping $A = f \circ \pi$ maps every possible argmax input tuple to exactly one element of $\mathcal{O}$ — either through an explicit LUT entry or through default safe-state routing (Algorithm 1). No path exists from the inference layer to an actuator bypassing this mapping. Therefore no output outside $\mathcal{O}$ is physically reachable under A1–A3. □

Remark 1 (What this does not guarantee)**.** *Proposition 1 does not guarantee that the selected action is contextually correct. If the classifier grid misidentifies a target due to sensor noise, occlusion, or distribution shift, the amygdala circuit will correctly select the option corresponding to the misclassified percept. Behavioral correctness depends on classifier grid accuracy. The architecture bounds which actions are physically possible; it does not bound which actions are triggered by a given percept. These are distinct problems requiring distinct solutions. Both are required for a safe system; this paper formalizes the second.*

Regulatory compatibility. The action-space determinism property is verifiable by circuit inspection. This architecture is structurally compatible with hardware certification frameworks (IEC 61508, DO-178C, IEC 62304) in a way that software-only AI systems are not. Full certification requires additional lifecycle processes, verification artifacts, and fault analysis beyond the scope of this note.

5 Hierarchical Decision DAGs

For complex domains, classifier grid output does not map directly to a terminal action. A sequence of amygdala nodes, each with its own option set, forms a Directed Acyclic Graph (DAG) of bounded decisions.

Definition 3 (Decision DAG)**.** *A decision DAG $\mathcal{D} = (V, E, \{\mathcal{O}_v\}_{v \in V})$ consists of decision nodes V, directed edges E encoding conditional routing, and per-node option sets $\mathcal{O}_v$. Each non-terminal node v receives classifier output, selects $o \in \mathcal{O}_v$ via its local amygdala circuit,*

and routes to the child node specified by that selection. Terminal nodes map to physical actuator commands.

Proposition 2 (DAG Action-Space Determinism)**.** *Under assumptions A1–A3 applied to each node $v \in V$:*

$$\forall v \in V, \quad \text{output}_v \in \mathcal{O}_v$$

Therefore the final action $\in \bigcup_{v \in V} \mathcal{O}_v$, fully specified at manufacture. No action outside this union is physically reachable regardless of inference-layer behavior at any node.

Example — Autonomous Drone Intercept. Table 1 shows a four-node DAG for drone engagement decisions.

Node	Input	Pre-wired options
1: Target assessment	Classifier grid output	`engage` | `hold` | `await_confirmation`
2: Threat geometry	Anti-drone operator position	`intercept_likely` | `intercept_unlikely` | `abort`
3: Approach maneuver	Threat bearing	`accel_right` | `accel_left` | `dive` | `afterburner` | `abort`
4: Strike solution	Position + maneuver state	$\{s_1, \ldots, s_K\}$ pre-computed solutions

Table 1: Four-node DAG for autonomous drone intercept. Rules of Engagement are encoded at the option-set level. A prohibited action is not a rule the system might violate — it is an option the circuit does not contain.

6 Cost Analysis

6.1 The Distillation Lag

Frontier vision models require expensive compute when first developed. Within 18–36 months, distillation compresses equivalent capability into models running on consumer NPU hardware. This has happened predictably across multiple capability generations [1, 2, 3]. The implication is direct: whatever a frontier model can do today in vision-detectable sensing, a $15 chip will handle within two years. The sensor array replacement argument does not depend on current model capability — it depends on where the distillation lag puts capability in the deployment window.

$$\text{BOM}_{\text{traditional}} \approx \sum_{i=1}^{n} \left(\text{Sensor}_i + \text{Interface}_i\right) \qquad \text{(vision-detectable components)}$$

$$\text{BOM}_{\text{ADA}} \approx \text{Camera} + \text{NPU} + \text{Logic gate array}$$

6.2 Bill of Materials Comparison

6.3 Scalability

Adding capability to the traditional system requires hardware change. Adding capability to this architecture requires retraining one classifier row — no hardware change. The amygdala circuit remains fixed and certified while the inference layer updates. OTA

Component	Traditional	This Architecture
Primary sensing	$40–120 (multi-sensor array)	$3–8 (camera module)
Inference compute	$20–60 (DSP/ASIC vision)	$8–15 (Hailo-8 / RK3588)*
Arbitration logic	$10–25 (MCU + firmware)	$3–7 (iCE40 FPGA)**
Per-feature R&D	Hardware change required	Training data addition only
BOM total	**$70–205**	**$14–30**

Table 2: Bill-of-materials comparison for a vision-detectable smart sensing application. *Hailo-8: $10–15 at volume [13]; RK3588: $8–20 [14]. **Lattice iCE40: $3–7 depending on LUT count [15].

updates to the inference layer must be treated as a re-certification event for safety-critical deployments. The certified option set $\mathcal{O}$ remains unchanged; only perception accuracy changes.

7 Domain Applications

All applications are scoped to vision-detectable sensing tasks only.

Agricultural sensing. Crop disease, pest classification, harvest readiness, irrigation indicators — offline, no connectivity. Option set: {`alert_disease`, `alert_pest`, `ready_harvest`, `irrigate`, `no_action`}.

Industrial quality control. Defect detection, orientation verification, component completeness for vision-inspectable defects. Option set: {`pass`, `reject`, `flag_human`, `halt_line`}. Nothing outside these four options is physically reachable.

Autonomous defense systems. As described in the DAG example (Table 1). Rules of Engagement encoded in option sets at manufacture. A prohibited engagement type is an absent circuit option, not a violated rule. This is the basis for legal certification compatibility under international humanitarian law frameworks for lethal autonomous systems.

Medical devices. Vision-based monitoring and environmental classification feeding bounded decisions: {`alert_physician`, `administer_A`, `administer_B`, `escalate`, `no_action`}. Structurally compatible with IEC 62304 for AI-assisted medical devices.

Accessibility hardware. Standalone object identification and scene description for visually impaired users. On-device, no data transmission. BOM compatible with low-income market deployment.

8 Open Engineering Questions

8.1 Sparse LUT complexity. Formalizing the sparse LUT for a given domain — determining which input combinations need explicit entries, which collapse to o_{safe}, and how the priority projection π is specified for a given application — is an open implementation problem. Computational complexity of this construction as a function of N rows and K classes per row requires formal characterization.

8.2 Confidence propagation. The classifier grid outputs probability distributions but the amygdala receives argmax indices, discarding uncertainty. Two approaches under consideration: (a) transmit full softmax distribution over a wider bus with a confidence threshold gate in the FPGA before LUT lookup; (b) add an explicit `uncertain` class to each row routing to o_{safe}. The second is simpler to implement and maintain.

8.3 Temporal robustness. The current architecture processes each frame independently. A state buffer requiring T consecutive consistent outputs before committing to an action is a natural extension. The latency–robustness tradeoff is domain-dependent and requires empirical characterization.

8.4 Hardware fault analysis. Assumptions A1–A3 are stated but not analyzed. Fault mode analysis covering FPGA bit-flip faults, SPI bus corruption, and driver failure — and their effect on the bounded-output guarantee — is required for safety-critical deployment. WCET analysis of the SPI-to-FPGA arbitration path, required for formal real-time certification, is also future work.

8.5 Option set governance. Who encodes which options into the circuit, and what accountability structure governs that encoding, is an ethical and legal question outside this technical note but essential to responsible deployment. In defense contexts the option set constitutes machine-readable Rules of Engagement. Its governance should be treated accordingly.

9 Conclusion

The core claim of this paper is narrow and precise: if AI output options are physically prewired into a circuit, the action space is bounded by topology, not by software rules that can be circumvented. The architecture implements this via a distilled classifier grid for perception and an FPGA-based amygdala circuit for hardware arbitration. Key contributions: (1) a distilled classifier grid replacing the vision-detectable subset of conventional sensor arrays at substantially lower BOM cost, with capability expansion via retraining rather than hardware change; (2) a sparse FPGA-based amygdala circuit, formally defined as $A = f \circ \pi$ with explicit priority projection, enforcing action-space determinism by circuit topology; (3) a precise formal statement — with explicit assumptions A1–A3 — of what the resulting guarantee provides and what it does not; (4) extension of the single-node guarantee to hierarchical decision DAGs; and (5) the observation that hardware-certifiable action-space bounding is structurally compatible with certification frameworks unavailable to software-only AI systems.

The architecture is at pre-prototype stage. Section 8 constitutes the immediate implementation work program. The author invites collaboration on the sparse LUT formalization and confidence propagation subproblems.

Acknowledgements

The author used AI-assisted tools for LaTeX formatting and prose editing. All technical ideas, architectural decisions, formal definitions, and open problem statements originated with the author. This work was conducted independently, without institutional affiliation or funding.

References

[1] Hinton, G., Vinyals, O., and Dean, J. (2015). *Distilling the Knowledge in a Neural Network*. NIPS Deep Learning Workshop. arXiv:1503.02531.

[2] Polino, A., Pascanu, R., and Alistarh, D. (2018). *Model Compression via Distillation and Quantization*. ICLR 2018.

[3] Howard, A. G., et al. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. arXiv:1704.04861.

[4] Iandola, F. N., et al. (2016). *SqueezeNet: AlexNet-level accuracy with 50x fewer parameters*. arXiv:1602.07360.

[5] Cai, H., Gan, C., and Han, S. (2020). *TinyTL: Reduce Activations, Not Trainable Parameters for Efficient On-Device Learning*. NeurIPS 2020.

[6] Amodei, D., et al. (2016). *Concrete Problems in AI Safety*. arXiv:1606.06565.

[7] Storey, N. (1996). *Safety-Critical Computer Systems*. Addison-Wesley.

[8] Altman, E. (1999). *Constrained Markov Decision Processes*. CRC Press.

[9] Huang, S. and Ontañón, S. (2020). *A Closer Look at Invalid Action Masking in Policy Gradient Algorithms*. arXiv:2006.14171.

[10] Colledanchise, M. and Ögren, P. (2018). *Behavior Trees in Robotics and AI*. CRC Press.

[11] Caruana, R. (1997). *Multitask Learning*. Machine Learning, 28(1), 41–75.

[12] Jacobs, R. A., et al. (1991). *Adaptive Mixtures of Local Experts*. Neural Computation, 3(1), 79–87.

[13] Hailo Technologies Ltd. (2022). *Hailo-8 AI Processor Datasheet*. https://hailo.ai/products/ai-accelerators/hailo-8-m2-ai-acceleration-module/

[14] Rockchip Electronics. (2022). *RK3588 Technical Reference Manual*. https://www.rock-chips.com/a/en/products/RK35_Series/2022/0926/1660.html

[15] Lattice Semiconductor. (2023). *iCE40 LP/HX Family Data Sheet*. https://www.latticesemi.com/Products/FPGAandCPLD/iCE40

[16] LeDoux, J. (2000). *Emotion circuits in the brain*. Annual Review of Neuroscience, 23(1), 155–184.