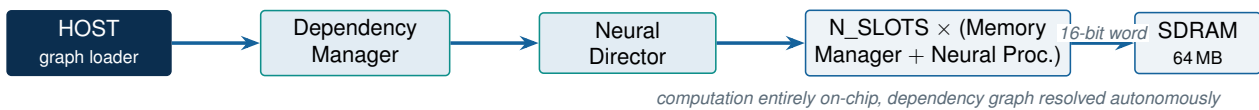


# FPGA – Neural V2

Neural Multiprocessor / Dataflow Machine

N\_SLOTS-way concurrent INT8 accelerator – Datasheet and reference manual

Rev. B2 • September 2026



**Reference target device:** Lattice ECP5 LFE5U-45F-8BG381C (speed grade –8, CABGA381) — identical device and board as V1.

**Production configuration:** INT8/INT32, P\_IN=8, N\_SLOTS=4 (real 8/8-seed timing closure at 64 MHz — see ch. 10), single unified SDR SDRAM (Alliance Memory AS4C32M16SB-7BIN, 64 MB), real board-level pinout and KiCad schematic/BOM.

**Status:** RTL verified in real Verilator simulation and real synthesis + place&route (Yosys + nextpnr-ecp5). Full benchmark campaign, two post-campaign memory optimizations, an alternative memory-subsystem redesign that became the current architecture (the Neural Memory System, ch. 13), and a real, board-level schematic/BOM verification pass (ch. 10) all complete and measured. Document describing the project as of September 2026.

Project author: Michele Bigi • MIKILAB / manvalan.

This datasheet documents V2 of the RTL code, documentation and benchmarks present in the repository [github.com/manvalan/FPGA-Neural](https://github.com/manvalan/FPGA-Neural). V1 remains frozen and unmodified as the project's golden functional/performance reference; its own datasheet previously lived alongside this one in this repository and was consolidated out of the working tree as part of a 2026-09-09 documentation cleanup (recoverable from git history).

## FPGA-Neural V2 — General description and features

FPGA-Neural V2 is a **neural multiprocessor / dataflow machine**, the evolution of the V1 sequential accelerator (documented separately, frozen and unmodified as the project's golden reference). Where V1 executes one neuron at a time under host-driven SPI control, V2 registers a **dependency graph of neurons** and keeps `N_SLOTS` independent Neural Processor + Memory Manager pairs busy concurrently, resolving data dependencies and hiding memory latency in hardware, without host intervention once a graph is loaded. Computation (INT8 MAC, ReLU, saturation) is bit-exact identical to V1's own datapath; what changed is everything *around* it, including, mid-project, the external memory device itself (§5.5).

### Features

- **Dependency-graph scheduling**: nodes are registered with an explicit producer list; a node becomes eligible for execution only once every producer it depends on has genuinely completed — verified for 1-hop shared-producer/multi-consumer graphs and 2-hop transitive (diamond) graphs.
- `N_SLOTS=4` independent **Neural Processor + Memory Manager** pairs (production baseline), each running the identical 8-stage INT8 pipeline inherited from V1.
- **Single unified SDRAM**: one external SDR SDRAM device serves weights, activations, AND results through one arbitrated backend (`sdram_unified_backend.v`) — no PSRAM, no second physical memory device, in the current, frozen hardware path.
- **Real physical host transport**: a placed, ball-assigned SPI Mode 0 slave (`spi_host_bridge.v`) plus a real `FPGA_DATA_READY` completion pin — both verified on real `nextpnr-ecp5` place&route, not just in simulation.
- **Real, board-level verification**: a real KiCad schematic capture, a real exported BOM, and real component selections (regulators, oscillator, configuration flash) all cross-checked against this datasheet — not merely a simulated design.
- **Real, measured** characterization at every step: Verilator RTL simulation, Yosys synthesis, real `nextpnr-ecp5` place&route — no theoretical number reported without a matching real measurement.

### Honest, measured limitations

- `N_SLOTS=8` is **functionally correct but not timing-closed**: only 3/8 tested placement seeds pass 64 MHz — deferred, not production-frozen (§10.2.4).
- **Hold-time closure is a genuine, disclosed tool-chain limitation**: no `pytrellis`/vendor static-timing-analysis path is available in this environment to check min-delay/hold, only setup (§10.2.4).
- **FPGA dynamic power/current draw is not measured**: no ECP5 power estimator is available in this toolchain; regulator sizing uses datasheet-based engineering margin, not a computed budget (§10.3).
- Fixed, lowest-index-priority arbitration (Director and memory arbiter alike) is not fairness-balanced — a real, measured per-slot workload imbalance exists under sustained contention.

### Target & toolchain

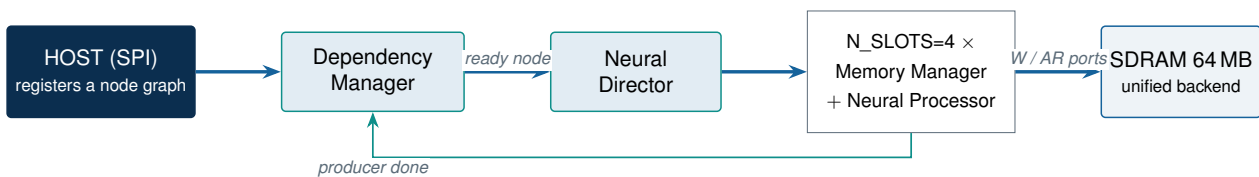
- FPGA: Lattice ECP5 LFE5U-45F-8BG381C (–8, commercial grade, 381-ball caBGA, 0.8 mm pitch) — same target device as V1.
- SDRAM: Alliance Memory AS4C32M16SB-7BIN (512 Mbit/64 MB, 4M×16, 54-ball FBGA).
- Synthesis: Yosys; place&route: real `nextpnr-ecp5` 0.11.1.
- Simulation: Verilator 5.050 (`-binary -timing`) — adopted for V2 after two independent Icarus Verilog v13.0 scheduling defects were found and reproduced on minimal repros (V1's own certification, performed separately, was unaffected).

### Key parameters (production configuration, real measured data)

| Quantity               | Value                  | Notes                                                                                                     |
|------------------------|------------------------|-----------------------------------------------------------------------------------------------------------|
| Data precision         | INT8 (signed)          | <code>DATA_WIDTH=8</code> , identical to V1                                                               |
| Accumulator            | INT32 (signed)         | <code>ACC_WIDTH=32</code>                                                                                 |
| Dot-product width      | 8                      | <code>P_IN=8</code> parallel MAC lanes per neuron                                                         |
| Production concurrency | <code>N_SLOTS=4</code> | real, 8/8-seed timing closure; see §10.2.4                                                                |
| System clock           | 64 MHz                 | 16 MHz oscillator → <code>EHXPLLL</code> PLL; 80 MHz confirmed NO-GO (genuine regenerated PLL, 0/8 seeds) |

|                                        |                                                      |                                                                  |
|----------------------------------------|------------------------------------------------------|------------------------------------------------------------------|
| Fmax, N_SLOTS=4<br>(real P&R, 8 seeds) | worst 64.55 MHz /<br>best 72.37 MHz                  | production baseline, 8/8 PASS                                    |
| D-Stress regression<br>(256 neurons)   | 49,927 cycles,<br>256/256 bit-exact                  | 780 $\mu$ s wall-clock @ 64 MHz                                  |
| SPI host clock,<br>verified            | 12 MHz<br>recommended<br>(12.8 MHz hard CDC<br>edge) | simulation-verified, real margin below the<br>deterministic edge |
| Address space                          | 26 bit (byte), single<br>SDRAM                       | ADDR_WIDTH=26                                                    |

### System block diagram



*A slot's completion feeds back to the Director (frees the slot) and to the Dependency Manager (wakes up any node waiting on it) — closing the dataflow loop entirely on-chip. `FPGA_DATA_READY` (ball G3) goes high once every registered node has both resolved and dispatched (§7.4).*

## Pinout summary — real, board-verified

V2's top-level module, `fpga_neural_v2_top.v`, has a complete, real ball assignment: every signal — SDRAM bus, SPI host transport, clock/reset, `FPGA_DATA_READY`, JTAG, configuration mode straps, and the boot flash's dedicated MSPI pins — carries a real CABGA381 ball site, sourced from the official Lattice pinout CSV (rev 3.0) and cross-checked against Project Trellis's own `iodb.json`. This supersedes an earlier V2 milestone in which the board-level top had been placed only **unconstrained**; a full, constrained `.lpf` now exists (`hardware/v2/constraints/v2_board_top.lpf`) and every Fmax number in this datasheet (§10.2.4) is measured against it.

### What is real

Every ball in the summary table below is placed, P&R-confirmed, and cross-checked against a real, exported KiCad schematic and BOM (§10.6–10.7) — not a simulation-only placeholder. No PSRAM signals exist anywhere in this revision: the single external memory is SDR SDRAM (§5.5).

### What remains open

FPGA dynamic power/current draw has not been measured post-implementation (no ECP5 power estimator is available in this toolchain), so exact decoupling/regulator sizing uses datasheet-based engineering margin, not a computed budget. Hold-time closure is a genuine tool-chain limitation (no min-delay analysis path available) — setup timing is fully verified. See ch. 10 for the complete, disclosed list.

**Ball summary (see ch. 10 for the complete, per-signal table)**

| Interface                                                                                                                                   | Ball count | Notes                         |
|---------------------------------------------------------------------------------------------------------------------------------------------|------------|-------------------------------|
| SDRAM bus<br>(A[0:12], BA[0:1],<br>DQ[0:15], DQM[0:1],<br>CK-<br>E/CS#/RAS#/CAS#/WE#)                                                       | 35         | Bank 6/7, real, P&R-confirmed |
| SPI host transport<br>(sclk/mosi/miso/cs)                                                                                                   | 4          | Bank 6/7, plain GPIO          |
| <code>FPGA_DATA_READY</code> ,<br><code>osc_clk</code> ,<br><code>ext_rst_n</code> ,<br><code>sdram_clk</code> ,<br><code>pll_locked</code> | 5          | Bank 6/7                      |
| JTAG (TCK/TMS/T-<br>DI/TDO)                                                                                                                 | 4          | Bank 40, to ESP32             |
| Config control<br>(PROGRAMN/INIT-<br>N/DONE) +<br>CFG[2:0] straps                                                                           | 6          | Bank 8                        |
| Boot-flash<br>dedicated MSPI<br>(CSSPIN/M-<br>CLK/D0/D1)                                                                                    | 4          | Bank 8, dual-function         |

Complete per-signal ball tables and the real KiCad schematic/BOM: [ch. 10](#). Logical (not physical) register-level port list: [ch. 11](#).

# Contents

---

|          |                                                                                                         |           |
|----------|---------------------------------------------------------------------------------------------------------|-----------|
| <b>1</b> | <b>Overview and design philosophy</b>                                                                   | <b>1</b>  |
| 1.1      | From sequential accelerator to dataflow machine                                                         | 1         |
| 1.2      | What did NOT change                                                                                     | 1         |
| 1.3      | What DID change                                                                                         | 1         |
| 1.4      | The central, measured finding                                                                           | 1         |
| <b>2</b> | <b>Architecture</b>                                                                                     | <b>3</b>  |
| 2.1      | Module map                                                                                              | 3         |
| 2.2      | Dependency Manager                                                                                      | 3         |
| 2.3      | Neural Director                                                                                         | 4         |
| 2.4      | Memory Manager + Neural Processor (per slot)                                                            | 4         |
| 2.5      | Activation Cache                                                                                        | 4         |
| 2.6      | Slot Memory Arbiter                                                                                     | 4         |
| 2.7      | Real, unmodified V1 PSRAM backend                                                                       | 4         |
| <b>3</b> | <b>Compute datapath</b>                                                                                 | <b>5</b>  |
| 3.1      | Bit-exact reuse of V1's arithmetic                                                                      | 5         |
| 3.2      | 8-stage pipeline                                                                                        | 5         |
| 3.3      | Accumulator width: 24 vs. 32 bits                                                                       | 5         |
| 3.4      | Activation and saturation                                                                               | 6         |
| <b>4</b> | <b>Parameters and configurability</b>                                                                   | <b>7</b>  |
| 4.1      | Build parameters (synthesis-time)                                                                       | 7         |
| 4.2      | Word-alignment constraint (post word-burst rewrite)                                                     | 7         |
| 4.3      | Characterized configurations (PSRAM-era; see ch. 10 §10.2.4 for the current SDRAM-architecture numbers) | 8         |
| 4.4      | Build versus runtime                                                                                    | 8         |
| <b>5</b> | <b>Memory subsystem</b>                                                                                 | <b>9</b>  |
| 5.1      | Baseline: reused byte-level V1 backend                                                                  | 9         |
| 5.2      | Optimization #1 — word-level burst reads                                                                | 9         |
| 5.3      | Optimization #2 — shared activation cache                                                               | 10        |
| 5.3.1    | Design notes                                                                                            | 10        |
| 5.4      | Real PSRAM chain (unmodified V1)                                                                        | 11        |
| 5.5      | SDRAM upgrade addendum (2026-09-07) — current, authoritative memory architecture                        | 11        |
| 5.5.1    | Real, measured clock closure                                                                            | 11        |
| <b>6</b> | <b>Dataflow scheduling</b>                                                                              | <b>12</b> |
| 6.1      | Node lifecycle                                                                                          | 12        |
| 6.2      | Verified graph topologies                                                                               | 12        |
| 6.3      | First-free dispatch                                                                                     | 12        |
| 6.4      | Measured scheduling behavior                                                                            | 13        |
| 6.5      | Correctness guarantees (measured, not assumed)                                                          | 13        |

|           |                                                                                          |           |
|-----------|------------------------------------------------------------------------------------------|-----------|
| <b>7</b>  | <b>Host / graph-loader interface</b>                                                     | <b>14</b> |
| 7.1       | Node registration protocol                                                               | 14        |
| 7.2       | Result readback                                                                          | 14        |
| 7.3       | What a real host driver would still need to add                                          | 14        |
| 7.4       | Addendum (2026-09-07) — real physical transport and completion signal, both now closed   | 15        |
| <b>8</b>  | <b>Top-level module</b>                                                                  | <b>17</b> |
| 8.1       | fpga_neural_v2_top.v                                                                     | 17        |
| 8.2       | Internal hierarchy                                                                       | 17        |
| <b>9</b>  | <b>ECP5 implementation &amp; benchmarks</b>                                              | <b>19</b> |
| 9.1       | Flow and verification discipline                                                         | 19        |
| 9.2       | V1 vs. V2 — final comparison                                                             | 19        |
| 9.3       | Real N_SLOTS sweep — Fmax and resources                                                  | 20        |
| 9.3.1     | Fmax versus N_SLOTS                                                                      | 20        |
| 9.4       | Real parallel scaling                                                                    | 20        |
| 9.5       | Memory optimization #1 — word-level burst reads                                          | 21        |
| 9.6       | Memory optimization #2 — shared activation cache                                         | 21        |
| 9.7       | Bottleneck analysis                                                                      | 21        |
| 9.8       | Limitations, honestly stated (PSRAM-era campaign above)                                  | 22        |
| 9.9       | SDRAM-era benchmark addendum (2026-09-07) — current, authoritative results               | 22        |
| 9.9.1     | Real resource utilization (N_SLOTS=4, SDRAM architecture, post real critical-path fixes) | 22        |
| 9.9.2     | Real, current clock closure and functional regression                                    | 23        |
| 9.9.3     | Real SPI host protocol throughput                                                        | 23        |
| 9.9.4     | Limitations, honestly stated (current SDRAM architecture)                                | 23        |
| <b>10</b> | <b>Hardware and board</b>                                                                | <b>24</b> |
| 10.1      | Board summary                                                                            | 24        |
| 10.2      | SDRAM upgrade addendum (2026-09-07) — current, authoritative board state                 | 24        |
| 10.2.1    | Memory device                                                                            | 24        |
| 10.2.2    | Complete AS4C32M16SB-7BIN ball assignment                                                | 24        |
| 10.2.3    | FPGA ↔ SDRAM mapping (real, LPF-verified)                                                | 25        |
| 10.2.4    | Real, measured clock closure (nextpnr-ecp5, 8 seeds/config)                              | 25        |
| 10.2.5    | Directed SDRAM boundary verification                                                     | 26        |
| 10.2.6    | Verified SPI host operating clock                                                        | 26        |
| 10.3      | Power supply design (2026-09-07) — verified against the real Lattice hardware checklist  | 26        |
| 10.3.1    | Rail topology                                                                            | 26        |
| 10.3.2    | Power-up sequencing — real Lattice requirement, verified compliant                       | 27        |
| 10.3.3    | Decoupling — real Lattice-recommended values (not a generic “one cap per pin” guess)     | 27        |
| 10.3.4    | Regulator component values (real, computed from datasheet constants)                     | 27        |
| 10.3.5    | Power tree                                                                               | 28        |
| 10.4      | Programming architecture (updated 2026-09-07) — single boot flash, ESP32 over JTAG only  | 28        |
| 10.4.1    | One physical flash chip: boot bitstream only                                             | 28        |
| 10.4.2    | ESP32 ↔ ECP5: JTAG only                                                                  | 28        |
| 10.4.3    | Real ball assignments (CABGA381)                                                         | 29        |
| 10.5      | Real KiCad schematic review (2026-09-07)                                                 | 29        |
| 10.5.1    | Confirmed correct                                                                        | 29        |
| 10.5.2    | Real findings — all resolved as of this pass                                             | 29        |
| 10.5.3    | Open items — all resolved as of this pass                                                | 30        |

|                                                                                              |           |
|----------------------------------------------------------------------------------------------|-----------|
| 10.6 Real KiCad schematic capture (2026-09-07) . . . . .                                     | 30        |
| 10.7 Bill of Materials (real, KiCad-exported, 2026-09-07) . . . . .                          | 30        |
| 10.7.1 Discrepancy: FPGA grade — resolved and now source-verified . . . . .                  | 31        |
| 10.7.2 Open items resolved by this BOM . . . . .                                             | 31        |
| 10.7.3 Resolved . . . . .                                                                    | 31        |
| 10.8 PCB module form factor (reserved) . . . . .                                             | 34        |
| 10.9 Verification status — real, disclosed open items . . . . .                              | 34        |
| <b>11 Register-level interface &amp; internal state encodings</b>                            | <b>35</b> |
| 11.1 Node registration fields (quick reference) . . . . .                                    | 35        |
| 11.2 Dependency Manager node state ( <i>node_state</i> ) . . . . .                           | 35        |
| 11.3 Neural Director state ( <i>dir_state</i> ) . . . . .                                    | 35        |
| 11.4 Memory Manager state ( <i>state</i> ) . . . . .                                         | 35        |
| 11.5 Neural Processor state ( <i>np_state</i> ) . . . . .                                    | 36        |
| 11.6 Slot Memory Arbiter owner encoding . . . . .                                            | 36        |
| <b>12 Roadmap and development status</b>                                                     | <b>37</b> |
| 12.1 Milestones M1–M10 . . . . .                                                             | 37        |
| 12.2 Post-campaign: targeted optimizations . . . . .                                         | 37        |
| 12.3 Open work items (real, not hidden) . . . . .                                            | 37        |
| <b>13 The Neural Memory System (NMS)</b>                                                     | <b>39</b> |
| 13.1 Design goal . . . . .                                                                   | 39        |
| 13.2 STEP1 — real bandwidth requirement study . . . . .                                      | 39        |
| 13.3 STEP2 — closed-form traffic model . . . . .                                             | 40        |
| 13.4 STEP3 — real bank-contention sweep . . . . .                                            | 40        |
| 13.5 STEP4–7 — real candidate synthesis and selection . . . . .                              | 40        |
| 13.6 STEP8 — full integration . . . . .                                                      | 41        |
| 13.7 STEP9–10 — real end-to-end benchmark vs. Current V2 . . . . .                           | 41        |
| 13.8 Real per-metric detail, N_SLOTS=2 (D-Stress) . . . . .                                  | 42        |
| 13.9 Recommendation . . . . .                                                                | 43        |
| 13.10 Open work (real, not hidden) . . . . .                                                 | 43        |
| <b>A Module and file map</b>                                                                 | <b>44</b> |
| A.1 V2 RTL ( <i>hardware/v2/rtl/</i> ) . . . . .                                             | 44        |
| A.2 NMS RTL ( <i>hardware/v2/nms/rtl/</i> , ch. 13) . . . . .                                | 44        |
| A.3 Reused, unmodified V1 ( <i>hardware/v1/rtl/</i> ) . . . . .                              | 45        |
| A.4 Simulation ( <i>hardware/v2/sim/</i> ) . . . . .                                         | 45        |
| A.5 Documentation and logs ( <i>hardware/v2/docs/</i> , <i>hardware/v2/logs/</i> ) . . . . . | 45        |

## CHAPTER 1

# Overview and design philosophy

---

### 1.1 From sequential accelerator to dataflow machine

V1 is, structurally, a single pipeline: one neuron computes at a time, driven by the host over SPI, one MAC group at a time, one layer at a time. It is fast for what it is (the V1 datasheet’s own “ECP5 implementation” chapter documents its real Fmax/timing-closure history), but it cannot keep more than one computational unit genuinely busy at once, and it has no notion of a dependency graph — the host sequences everything.

V2 keeps V1’s own proven INT8 datapath (bit-exact, byte-for-byte reused math) but wraps it in a fundamentally different control architecture: a **Dependency Manager** tracks a graph of neuron “jobs”, each with an explicit list of producer nodes it depends on; a **Neural Director** dispatches every node whose dependencies have resolved to whichever of `N_SLOTS` concurrent (Memory Manager + Neural Processor) pairs is free; a slot’s completion feeds back to wake up any node that was waiting on it. Once a graph is loaded, the whole system runs autonomously — no per-neuron host intervention.

### 1.2 What did NOT change

- The `INT8×INT8→INT32` MAC math, the balanced adder tree, ReLU/linear activation with saturation — `neural_processor.v` is a direct, bit-exact-verified port of V1’s own `neuron_parallel.v/mac8.v/mac_unit.v`.
- V1’s own PSRAM backend files (`memory_interface.v`, `psram_controller.v`) remain byte-for-byte, unmodified copies throughout the repository — V1 itself, as a tree (`hardware/v1/`), is frozen and was never touched. **Not currently part of V2’s physical board**, however: the project has since replaced external memory with a single SDR SDRAM device (§5.5); the PSRAM-era chapters that follow document real, correctly-measured work for the architecture it was measured on, not the current board.
- The target device (Lattice ECP5 LFE5U-45F-8BG381C) and the real-toolchain-only measurement discipline: every number in this datasheet is labelled THEORETICAL, SIMULATED, POST-P&R MEASURED, or DERIVED, and no result was invented to make V2 look better than it measured (§9).

### 1.3 What DID change

- **Concurrency**: from one active neuron to `N_SLOTS` independent Neural Processor instances, each fed by its own Memory Manager.
- **Scheduling**: from host-sequenced SPI opcodes to an on-chip dependency graph, resolved autonomously.
- **Memory backend granularity**: from byte-at-a-time fetches (through `int8_memory_access.v`, still frozen V1 but no longer instantiated in V2’s own datapath) to word-level bursts talking to `memory_interface.v` directly — a real, measured  $2.24\text{--}2.37\times$  speedup (ch. 5).
- **Memory traffic pattern**: a new shared on-chip **activation cache** eliminates redundant re-fetching of an input vector shared by many neurons of the same layer — a further real  $1.66\text{--}2.00\times$  cycle reduction, at a real, honestly reported Fmax cost (ch. 5).

### 1.4 The central, measured finding

The single most important result of this project’s own benchmark campaign is that **V2 is memory-bound, not compute-bound**: the real compute-to-memory-wait ratio is on the order of 1:170–1:220,

and the one physical PSRAM port saturates at  $\approx 90\%$  utilization regardless of  $N\_SLOTS \geq 2$ . Real parallel scaling from  $N\_SLOTS=1$  to  $N\_SLOTS=8$  is essentially flat for large/sustained workloads ( $1.05\text{--}1.06\times$ ), and once real, place&route-measured Fmax degradation from added routing congestion is also accounted for,  $N\_SLOTS=4$  measures as *slower* in real wall-clock time than  $N\_SLOTS=1$  for the largest workload tested — more hardware parallelism made that specific configuration worse, not better, because the bottleneck was never compute. This finding directly shaped both post-campaign optimizations in ch. 5.

#### Architecture changed since this finding: SDRAM, not PSRAM

This memory-bound finding was measured on the PSRAM-era architecture described above. The project has since replaced PSRAM with a single SDR SDRAM device (§5.5) and closed on  **$N\_SLOTS=4$  as the production configuration** — chosen primarily because it is the largest slot count that reliably closes real timing (8/8 seeds @ 64 MHz, ch. 10 §10.2.4), not from a re-run of this specific utilization/scaling study. Whether the SDRAM backend's own utilization/saturation ratio matches the PSRAM-era  $\approx 90\%$  figure above has **not been independently re-measured** — disclosed as an open item, not assumed to carry over.

#### Reproducibility

Every real number in this datasheet traces to a specific, append-only log entry (`EXP-NNNN`, `DEC-NNNN`, `ERR-NNNN`) in `hardware/v2/logs/`, a specific git commit, and an exact toolchain command — the same discipline applied throughout V1's own development.

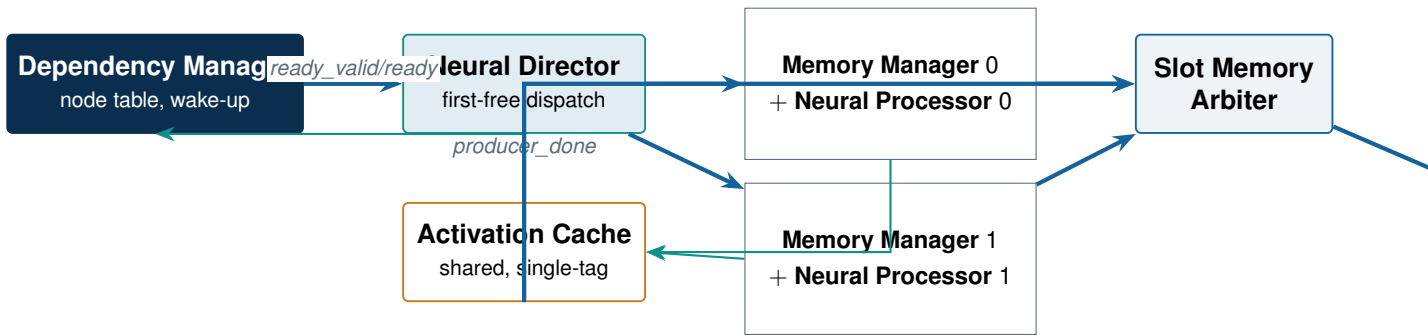
## CHAPTER 2

# Architecture

### Scheduling core unchanged; memory backend and slot count have

`dependency_manager.v` and `neural_director.v` (this chapter's own subject) are identical between the PSRAM-era milestone described below and the current, real SDRAM board — the scheduling logic itself did not change. What changed since is the memory backend (single SDR SDRAM, not PSRAM, §5.5), the absence of the shared **Activation Cache** module from the current physical top (ch. 8), and the production slot count (`N_SLOTS=4`, not 2).

## 2.1 Module map



PSRAM-era diagram, `N_SLOTS=2` shown; the architecture is parametric in `N_SLOTS`. Every arrow is a real signal path verified in Verilator simulation and real Yosys/nextpnr-ecp5 synthesis. The current, real board (`N_SLOTS=4`, single SDRAM, no Activation Cache module) is shown in ch. 8's own hierarchy listing.

## 2.2 Dependency Manager

Holds a table of `N_NODES` job descriptors, each tracking: node id, state (EMPTY/WAITING/READY/DISPATCHED), required-dependency count, resolved-dependency count, up to `MAX_DEPS` producer node ids, and the job descriptor fields (`x_base`, `w_base`, `n_tiles`, `result_addr`). A node with zero required dependencies is immediately READY on registration. When a producer completes, every WAITING node listing it among its own producers gets its resolved-dependency count incremented — a single producer can satisfy several waiting consumers (shared-producer/multi-consumer), and a node depending on several producers accumulates resolution across separate events (multiple dependencies). Verified for both 1-hop and 2-hop transitive (diamond) graphs. Ready nodes are handed to the Neural Director one at a time over a backpressure-safe valid/ready interface.

### No slot reclamation

`ST_DISPATCHED` is terminal: node table slots are never reused once dispatched. A long-running system that keeps registering new nodes without limit will eventually exhaust `N_NODES` — this is a real, measured consequence (a benchmark testbench hit exactly this deadlock via node-id wraparound before `N_NODES` was sized generously enough). Slot reclamation is explicitly deferred, not forgotten.

## 2.3 Neural Director

Dispatches ready job descriptors to whichever of `N_SLOTS` Memory Manager instances is currently free — **first-free** scheduling: a fixed, lowest-index-wins priority scan, not load-balanced. Slot-busy tracking and completion detection are always-active, independent of whatever the allocate/scan control state happens to be that cycle (the same “don’t gate a per-unit event behind one shared FSM state” principle applied throughout this design). A completed slot’s node id is tracked (`slot_node_id`) so its completion can be resolved back to a `producer_done` event for the Dependency Manager, closing the wake-up loop without any external glue logic.

### Measured scheduling imbalance

Real per-slot data (`N_SLOTS=4`, a 128-neuron workload) shows slots 0/1 delivering 1008 real tiles each while slots 2/3 deliver only 16 each, despite all four slots reporting near-100% “busy” utilization — direct, measured evidence that fixed lowest-index priority does not distribute load evenly once the shared PSRAM port is the real constraint. See ch. 9.

## 2.4 Memory Manager + Neural Processor (per slot)

Each slot pairs one `memory_manager.v` instance with one `neural_processor.v` instance. The Memory Manager double-buffers tile fetches (compute tile  $N$  while prefetching tile  $N+1$ ) and presents the Neural Processor with a simple “data available” interface (`operand_valid/ready`, `tile_last`) — the processor never sees PSRAM request/wait cycles directly. A tile’s activation half is requested from the shared Activation Cache; its weight half is fetched directly (weights are per-neuron, never shared, so caching them would not help). A bank is presentable to the processor only once *both* halves have arrived (`bank_ready = bank_x_ready & bank_w_ready`).

## 2.5 Activation Cache

A single shared instance (not one per slot) serving every Memory Manager’s activation-fetch requests. Single-tag design: one cached `x_base` at a time, filled tile-by-tile on first use, served directly from an on-chip buffer on every subsequent request for the same vector — no PSRAM access on a hit. A request for a different `x_base` invalidates the cache and restarts filling from tile 0; this is always *correct* (never serves stale data) but can thrash under interleaved, genuinely-different-`x_base` concurrent traffic — an honestly documented limitation, not exercised by this project’s own realistic dense-layer workloads (where many neurons of one layer share one input vector, dispatched together). Full detail, including the real Fmax cost this module introduces, in ch. 5.

## 2.6 Slot Memory Arbiter

Funnels `N_SLOTS+1` independent backend ports (one per Memory Manager’s weight/write-back traffic, plus one for the Activation Cache’s own traffic) down to the one real physical PSRAM port. Fixed lowest-index priority, same convention as the Director. Every incoming request is latched into a per-port pending register regardless of arbiter state — a byte-level backend protocol quirk discovered by real simulation (a fire-and-forget single-cycle request pulse can arrive while the shared bus is owned by another port; a naive “grant only while live” arbiter would silently drop it) made this latch a correctness requirement, not an optimization.

## 2.7 Real, unmodified V1 PSRAM backend

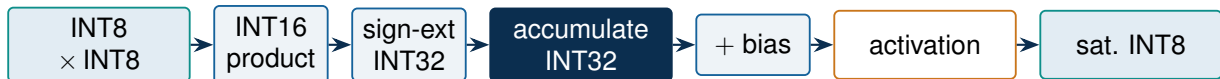
`memory_interface.v` and `psram_controller.v` are reused byte-for-byte from the frozen `hardware/v1/` tree. The controller’s own real page-mode support (fast same-page continuation vs. a slower cold access) was already implemented in V1 and is exploited more effectively by V2’s word-level burst rewrite (ch. 5) — no change to the controller itself was needed or made.

## CHAPTER 3

# Compute datapath

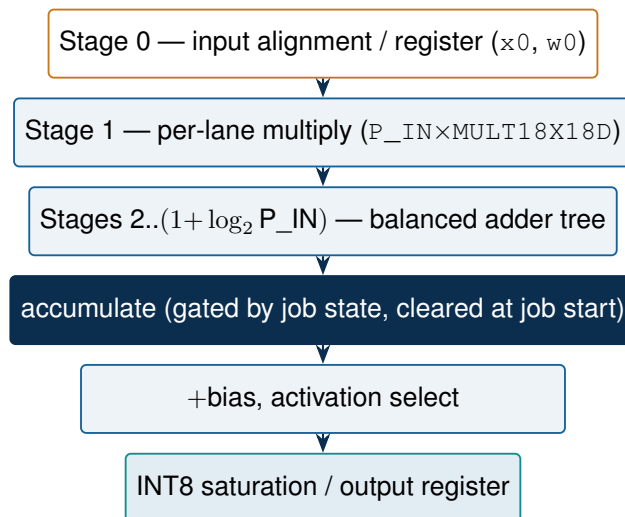
### 3.1 Bit-exact reuse of V1's arithmetic

`neural_processor.v` implements the identical INT8/INT32 arithmetic chain as V1's own `neuron_parallel.v/mac8.v/mac_unit.v` — verified bit-exact against V1's own modules, instantiated side-by-side in the same testbench, across 7 test cases including extreme INT8 values, back-to-back zero-gap tiles, and multi-tile jobs. What changed is the *pipelining*, not the math.



### 3.2 8-stage pipeline

`neural_processor.v` is fully pipelined, throughput-oriented (one new tile accepted per cycle in steady state, given a continuous operand stream):



With `P_IN=8` the adder tree has 3 levels, giving an 8-stage pipeline overall. `tile_last` is gated identically to `valid` at every stage (`last0 <= (operand_valid && operand_ready) ? tile_last : 1'b0;`) — an early draft left it ungated, letting a “last” tag propagate one cycle ahead of its own valid/data pair on jobs where `tile_last` was asserted before `operand_ready` rose (legal valid-before-ready producer behavior); found and fixed via a cycle-by-cycle dump of the pipeline's own internal valid/last signals, re-verified against the full 7-test regression.

### 3.3 Accumulator width: 24 vs. 32 bits

A real, 6-seed placement sweep (reusing already-synthesized netlists, real `nextpnr-ecp5` place&route only) resolved an earlier single-seed measurement that had suggested `ACC_WIDTH=32` was marginally faster:

| ACC_WIDTH | mean<br>Fmax | min    | max    | stdev |
|-----------|--------------|--------|--------|-------|
| 32        | 170.12 MHz   | 145.73 | 183.96 | 14.16 |

|    |            |        |        |      |
|----|------------|--------|--------|------|
| 24 | 180.71 MHz | 175.16 | 185.49 | 4.21 |
|----|------------|--------|--------|------|

*Real place&route, P\_IN=8, 6 seeds each (default plus 5 explicit).*

#### Why a single seed misled

Over 6 real placement seeds, ACC\_WIDTH=24 has both a higher mean Fmax (+6.2%) and a much tighter seed-to-seed spread ( $\approx 3.4\times$  tighter) than ACC\_WIDTH=32 — combined with fewer LUT/FF/CCU2C at 24 bits and identical bit-exact correctness, ACC\_WIDTH=24 is recommended for any new P\_IN=8 INT8 configuration, where product magnitudes never need more than 24 bits of accumulator headroom.

### 3.4 Activation and saturation

Identical encoding and bit-test logic to V1 (bilateral saturation for ACT\_NONE, positive-only for ACT\_RELU, both INT8-range). Every job dispatched by `dataflow_core.v` currently hardcodes `job_bias=0`, `job_activation=ACT_RELU` — a documented simplification carried through every milestone since M4/M5, not yet exposed per-node by the Dependency Manager's own job descriptor.

| Encoding | Value | Behavior                                                                 |
|----------|-------|--------------------------------------------------------------------------|
| ACT_NONE | 2' d0 | Linear: bilateral saturation to $[-128, +127]$ .                         |
| ACT_RELU | 2' d1 | $\max(0, x)$ , positive saturation to +127 (used by every V2 job today). |

## CHAPTER 4

# Parameters and configurability

### 4.1 Build parameters (synthesis-time)

#### Current, real board parameters (`fpga_neural_v2_top.v`)

The table below reflects the real, current SDRAM-architecture top level. The PSRAM-era §§5 chapters below this one describe an earlier, real, correctly-measured milestone with different defaults (notably `ADDR_WIDTH=23` and a PSRAM data-bus parameter) — superseded, not deleted, since that data remains accurate for the architecture it was measured on.

| Parameter                 | Default | Meaning                                                                                                                                         |
|---------------------------|---------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>DATA_WIDTH</code>   | 8       | Data width (INT8), unchanged from V1.                                                                                                           |
| <code>ACC_WIDTH</code>    | 32      | Accumulator width.                                                                                                                              |
| <code>P_IN</code>         | 8       | Parallel MAC lanes per neuron per tile; must be even (word-level burst constraint, ch. 5).                                                      |
| <code>ADDR_WIDTH</code>   | 26      | Byte-address width (widened 23→26 for the 64 MB SDRAM device, DEC-0039).                                                                        |
| <code>N_SLOTS</code>      | 4       | Concurrent Memory Manager+Neural Processor pairs.<br><b>Production configuration</b> — real 8/8-seed timing closure at 64 MHz (ch. 10 §10.2.4). |
| <code>N_NODES</code>      | 16      | Dependency Manager node-table depth. Sized to the largest node-id range a graph will ever use; never reclaimed (ch. 2).                         |
| <code>MAX_DEPS</code>     | 4       | Maximum producers a single node can list.                                                                                                       |
| <code>QUEUE_DEPTH</code>  | 8       | Neural Director's own ready-job FIFO depth.                                                                                                     |
| <code>MAX_TILES</code>    | 16      | Longest activation/weight tile run a job can request.                                                                                           |
| <code>CLK_FREQ_MHZ</code> | 64      | Real system clock, generated by <code>ecp5_pll_sys_clk.v</code> from the 16 MHz oscillator; 80 MHz confirmed NO-GO (ch. 10 §10.2.4).            |

#### `N_SLOTS` is a real, measured trade-off, not a free parameter

`N_SLOTS=4` is the production default: the largest slot count that reliably closes real timing at 64 MHz on every tested placement seed (8/8). `N_SLOTS=8` is functionally correct (bit-exact) but only 3/8 seeds close timing — deferred, not production-frozen. Do not simply raise `N_SLOTS` without re-running the real 8-seed `nextpnr-ecp5` matrix.

### 4.2 Word-alignment constraint (post word-burst rewrite)

Since the memory backend now moves 16-bit words rather than bytes (ch. 5), every tile base address the system computes (`x_base + tile_idx*P_IN`, and equivalently for weights) must land on an even byte address. `P_IN` even and `x_base/w_base` themselves even together guarantee this for every tile of every job — true of every address this project's own testbenches use, and a trivial constraint for any real loader/host to satisfy.

### 4.3 Characterized configurations (PSRAM-era; see ch. 10 §10.2.4 for the current SDRAM-architecture numbers)

| N_SLOTS | Fmax,<br>word-burst<br>only                                  | Fmax,<br>+activation<br>cache | Notes                                                                                               |
|---------|--------------------------------------------------------------|-------------------------------|-----------------------------------------------------------------------------------------------------|
| 1       | 152.44 MHz                                                   | 131.79 MHz                    | Best real wall-clock speedup ( $3.86\times$ vs baseline); no arbitration contention possible.       |
| 2       | 133.58 MHz                                                   | <b>87.72 MHz</b>              | <b>Recommended default</b> — real $2.45\times$ speedup vs baseline, still comfortably above 80 MHz. |
| 4       | 112.07 MHz                                                   | 65.01 MHz<br>( <b>FAIL</b> )  | No additional real throughput; fails 80 MHz with the cache active. Not recommended.                 |
| 8       | 92.63 MHz<br>(dataflow_core<br>only, no real<br>PSRAM chain) | not<br>re-measured            | Real DSP ceiling for P_IN=8 (64/72 MULT18X18D); a resource ceiling, not a useful operating point.   |

### 4.4 Build versus runtime



Full field-level description of the runtime (node registration) interface: ch. 11.

## CHAPTER 5

# Memory subsystem

### 5.1 Baseline: reused byte-level V1 backend

V2's first working milestones connected each Memory Manager's own `prefetch_engine.v` to the real, unmodified V1 chain `int8_memory_access.v` → `memory_interface.v` → `psram_controller.v`, fetching one INT8 byte per transaction — exactly the contract V1's own `neuron_memory.v` already used against the same backend. This was correct and fully verified (bit-exact end-to-end through the real PSRAM chain), but it was not the fastest possible use of that chain.

### 5.2 Optimization #1 — word-level burst reads

Direct inspection of `int8_memory_access.v` shows it already converts every 8-bit logical request into a **full 16-bit PSRAM word access** internally (`mem_addr <= addr >> 1`, one byte lane selected via `lb_n/ub_n`) — so a byte-at-a-time fetch was already paying for two bytes of real PSRAM bandwidth per transaction while using only one, and paying `int8_memory_access.v`'s own request/wait round-trip twice for every real word instead of once.

`prefetch_engine.v` (weights) and `activation_cache.v` (activations, §5.3) now talk directly to `memory_interface.v`'s own 16-bit word interface, **skipping `int8_memory_access.v` entirely**. Both files remain frozen, byte-for-byte unmodified V1 — V2 simply chooses to reuse the lower (word-level) layer of the same frozen stack instead of the byte-splitting layer on top of it, the same precedent already set by `slot_mem_arbiter.v` not reusing V1's own `mem_arbiter.v` verbatim.

#### Real, measured result — single job, real PSRAM

| n_tiles | cycles, before | cycles, after | Δ    |
|---------|----------------|---------------|------|
| 1       | 166            | 84            | −49% |
| 3       | 446            | 204           | −54% |
| 5       | 728            | 322           | −56% |

Real Verilator simulation, real V1 PSRAM chain, all results still bit-exact.

Combined real wall-clock effect (256-neuron sustained workload,  $\text{cycles} \div \text{real POST-P\&R Fmax}$ ): a **2.24–2.37×** speedup across every `N_SLOTS` tested, at a negligible real `Fmax` cost (unchanged at `N_SLOTS=1`; −6.2% at `N_SLOTS=2`; −1.2% at `N_SLOTS=4`).

#### Why not just pipeline more requests instead?

`int8_memory_access.v`'s own `STATE_IDLE` only samples a new `req` once back in `STATE_IDLE` after the previous transaction's `mem_ready` — it fundamentally does not support request pipelining. No wrapper built *on top of* it can avoid paying its round-trip cost twice per word; only bypassing it (talking to `memory_interface.v` directly) actually removes the redundancy. This is why the fix reaches one layer lower in the stack rather than adding queuing logic in front of the existing byte-level port.

### 5.3 Optimization #2 — shared activation cache

In the realistic dense-layer workloads this project benchmarks, many neurons of the same layer share the *exact same* activation vector. Before this optimization, each of `N_SLOTS` Memory Manager instances re-fetched that identical vector from PSRAM independently — real, measured, redundant traffic on the one shared PSRAM port. `activation_cache.v` (a new, single shared instance per `dataflow_core`, not one per slot) fetches a given `x_base` vector once, tile by tile on first use, and serves every subsequent request for the same vector directly from an on-chip buffer.

#### Real, measured result — 256-neuron sustained workload, D-Stress

| N_SLOTS | cycles, burst only | cycles, +cache | Fmax, burst | Fmax, +cache    |
|---------|--------------------|----------------|-------------|-----------------|
| 1       | 348682             | 174610         | 152.44      | 131.79          |
| 2       | 307602             | 185428         | 133.58      | <b>87.72</b>    |
| 4       | 307346             | 184795         | 112.07      | 65.01<br>(FAIL) |

A further real 1.66–2.00× cycle reduction on top of optimization #1, ≈4× combined vs. the original byte-level baseline.

#### Real, measured Fmax cost — read this before raising N\_SLOTS

The shared cache's real Fmax cost is **much steeper** than optimization #1's: a single central resource with `N_SLOTS` request ports, a broadcast-capable hit-check evaluated combinatorially every cycle for every port, and a shared `tile_store` array create a genuine fan-in/routing hot spot that grows with `N_SLOTS`. `N_SLOTS=2` (recommended) still passes 80 MHz (87.72 MHz, margin down from +67% to +9.7%); `N_SLOTS=4` **fails outright** (65.01 MHz). This is the central input to ch. 12's own open work item on cache pipelining.

Combined real wall-clock speedup vs. the original byte-level baseline (both optimizations together): `N_SLOTS=1` **3.86×**; `N_SLOTS=2` **2.45×** (the recommended configuration); `N_SLOTS=4` **2.29×** but a real *regression* versus optimization #1 alone, since its own Fmax now fails 80 MHz.

#### 5.3.1 Design notes

Single-tag, tile-granular: a request tag mismatch invalidates the cache and restarts filling from tile 0 for the new `x_base` — always correct, never serves stale data, but can thrash under interleaved, genuinely-different-`x_base` concurrent traffic (not exercised by this project's own dense-layer workloads, where sharing is real and sustained). Requests are latched per-slot on arrival (the same “queue, don't drop” idiom used by the arbiter, §2.5 of ch. 2) and served with a broadcast ack the cycle a matching tile becomes valid, so multiple slots pending on the same, about-to-arrive tile are all served the same cycle.

#### Two real bugs found and fixed during implementation

- (1) A target-bank/pending-bank race: a later handoff could queue a new cache request (targeting a different double-buffer bank) in the same cycle an earlier request was still awaiting its own ack, and non-blocking-assignment “last write wins” semantics silently misattributed which bank the earlier request's data landed in — the same bug class already found once for the weight-side `pf_target_bank` register, fixed with the identical two-register (pending/target) staging pattern.
- (2) A zero-width Verilog replication at `N_SLOTS=1` (`{ $clog2(1) { 1'b0 } } = { 0 { ... } }`, illegal outside a concatenation), the same class already found once in `neural_director.v` and

fixed with the same width-agnostic '0 literal. Both found via real simulation, not by inspection.

## 5.4 Real PSRAM chain (unmodified V1)

`memory_interface.v` and `psram_controller.v` are byte-for-byte identical to V1's own copies throughout this chapter — the real page-mode support they already implement (fast same-page continuation, slower cold access) is exploited more effectively by the word-level rewrite, not changed. The real ISSI `IS66WVE4M16EBLL-70BLI` chip and its board wiring are unchanged from V1 (ch. 10).

## 5.5 SDRAM upgrade addendum (2026-09-07) — current, authoritative memory architecture

### Superseded architecture

The PSRAM-based chain described above (§§5.2–5.3) belongs to an earlier V2 milestone. The project has since closed on a single-external-memory architecture (real `decisions.log` DEC-0034): **one SDR SDRAM device, one `sdram_controller.v` instance**, serving weights, activations, AND results through `sdram_unified_backend.v`'s two logical ports (W: 64-bit weight read; AR: 16-bit, byte-maskable activation-read/result-write), arbitrated 2-way priority (W wins when both pending). No PSRAM, no second physical memory device, in the current, frozen hardware path.

The device itself was upgraded mid-project from an 8 MB part (`AS4C4M16SA-6TIN`) to the current **AS4C32M16SB-7BIN, 64 MB (512 Mbit), 54-ball FBGA** — both the row/column/bank geometry (`sdram_controller.v`'s `ROW_BITS/COL_BITS/BANK_BITS` parameters, now 13/10/2) and the SPI host protocol's own address-field width (23→26-bit byte address; `WRITE_JOB` payload grew 15→18 bytes) changed accordingly. Full electrical/pinout data and the complete FPGA↔SDRAM ball mapping are in ch. 10, §10.2 (kept in one place to avoid two copies of the same real data).

### 5.5.1 Real, measured clock closure

**N\_SLOTS=4 @ 64 MHz is the frozen production configuration:** real `nextpnr-ecp5` P&R, 8/8 tested seeds PASS. **N\_SLOTS=8 @ 64 MHz is deferred**, not production-frozen: 3/8 seeds PASS in the final, current RTL state. 80 MHz was tested with a genuinely regenerated PLL (not merely a `-freq` flag) and is **not achievable** at either processor count — the achievable Fmax is a property of the routed fabric, confirmed identical between the 64 MHz- and 80 MHz-targeted netlists. Bit-exact functional correctness (D-Stress, 256/256 neurons vs. golden model) is unaffected at every configuration tested.

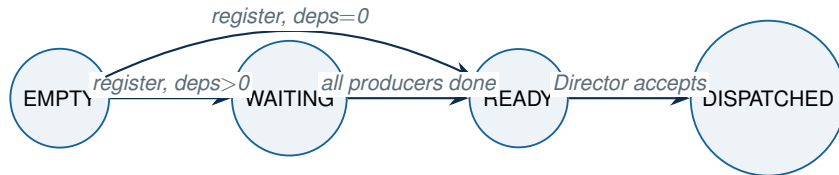
### Single source of truth for exact numbers

The exact per-seed Fmax/WNS table, its full revision history (three successive real critical-path fixes: ERR-0027, ERR-0028, ERR-0029, plus a later fan-out fix, DEC-0042), and the SDRAM directed boundary-test result (21/21 PASS, both 64 MHz and 166 MHz) are kept in one place to avoid two copies of the same real data — see ch. 10 §10.2.4 and §10.2.

## CHAPTER 6

# Dataflow scheduling

### 6.1 Node lifecycle



DISPATCHED is terminal (§2): a real, honest consequence, not an oversight — see the roadmap (ch. 12) for the deferred slot-reclamation work item.

### 6.2 Verified graph topologies

| Topology                                                                                                                                          | What it proves                                                                                                                                                                    |
|---------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Shared producer, 2 consumers                                                                                                                      | One node's completion resolves the dependency count of <i>two</i> different waiting nodes independently.                                                                          |
| Multiple producers, 1 consumer                                                                                                                    | A node with <code>required&gt;1</code> only becomes <code>READY</code> once <i>every</i> listed producer has completed, tracked across separate wake-up events.                   |
| 2-hop transitive diamond ( <i>A</i> , <i>B</i> independent; <i>C</i> dep- <i>A</i> ; <i>D</i> dep- <i>B</i> ; <i>E</i> dep- <i>C</i> , <i>D</i> ) | Correct cascading wake-up two hops deep — <i>E</i> does not fire until <i>C</i> and <i>D</i> have <i>themselves</i> genuinely completed, not merely been marked ready.            |
| Mixed-depth fan-in (node depending on both a root and a 1-hop descendant)                                                                         | Dependency resolution does not assume a uniform graph depth.                                                                                                                      |
| Multilayer (8 layer-1 neurons, random INT8 data, feeding 2 layer-2 neurons reading their real shared result bytes)                                | Real cross-node <i>data</i> forwarding through real PSRAM — layer-2's golden values are computed from the real bytes layer-1 actually wrote, not from an independent expectation. |

All topologies above were exercised with the real, full `neural_multiprocessor.v` (real V1 PSRAM chain, real `slot_mem_arbiter.v`) and verified bit-exact against a software golden model.

### 6.3 First-free dispatch

The Neural Director's own scheduling policy is deliberately the simplest one that is provably correct: a fixed, lowest-index priority scan over currently-free slots. Round-robin, least-loaded, or any fairness-aware alternative was explicitly deferred until real measured data showed whether it mattered (§6.4).

## 6.4 Measured scheduling behavior

Real per-slot data (`N_SLOTS=4`, a 128-neuron dense-layer workload) shows a striking imbalance: slots 0 and 1 each deliver 1008 real tiles, while slots 2 and 3 deliver only 16 each — despite all four slots reporting near-100% “busy” utilization. The cause is not unfairness in isolation: once the shared PSRAM port is saturated (ch. 5), there is rarely a moment where the low-index slots are simultaneously busy *and* the high-index slots have nothing to do, so the fixed low-index-first scan keeps re-selecting the same two slots. This is a real, measured limitation of the current scheduler, carried into ch. 12 as an open item rather than patched without first measuring whether it is worth the added complexity for real workloads.

## 6.5 Correctness guarantees (measured, not assumed)

Across the full final benchmark campaign (6 workloads  $\times$  4 `N_SLOTS` configurations, re-verified after both memory optimizations): **zero** lost jobs, **zero** duplicated jobs (`jobs_allocated == jobs_completed == neurons_completed` exactly, every run), **zero** deadlocks, **zero** timeouts, correct multi-hop dependency wake-up in every topology tested.

## CHAPTER 7

# Host / graph-loader interface

### Scope of this chapter

V1's own host interface is a real, placed, physically-verified SPI Mode 0 slave (ch. 7 of the V1 datasheet). V2's equivalent — a node-registration bus into `neural_multiprocessor.v` — has, in this revision, been exercised exclusively from Verilator testbenches and unconstrained synthesis top-levels. This chapter describes the **logical** protocol only; no real host-side driver (SPI or otherwise) has been built or placed yet. See ch. 12.

### 7.1 Node registration protocol

A simple valid/ready producer interface, backpressure-safe: the loader holds `reg_valid` and the node's own fields until `reg_ready` is observed high on the same cycle, exactly like registering into any FIFO. `reg_ready` for a given `reg_node_id` is asserted whenever that node's own table slot is `EMPTY` (§6).

| Field                         | Width                                                                 | Meaning                                                                                       |
|-------------------------------|-----------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|
| <code>reg_node_id</code>      | $\lceil \log_2 N\_NODES \rceil$                                       | Node's own id — doubles as its table slot index.                                              |
| <code>reg_required</code>     | $\lceil \log_2 (\text{MAX\_DEPS} \times \text{PACKED\_NODES}) \rceil$ | How many of <code>reg_producer_ids</code> are meaningful (0 $\Rightarrow$ immediately READY). |
| <code>reg_producer_ids</code> | $\text{MAX\_DEPS} \times \text{PACKED\_NODES}$                        | Packed array of producer node ids this node depends on.                                       |
| <code>reg_x_base</code>       | <code>ADDR_WID</code>                                                 | Base byte address of this node's activation vector.                                           |
| <code>reg_w_base</code>       | <code>ADDR_WID</code>                                                 | Base byte address of this node's weight vector.                                               |
| <code>reg_n_tiles</code>      | 16                                                                    | Number of <code>P_IN</code> -wide tiles to accumulate.                                        |
| <code>reg_result_addr</code>  | <code>ADDR_WID</code>                                                 | Byte address the computed INT8 result is written to.                                          |

### A node id is a real, finite resource

Because dispatched node table slots are never reclaimed (§6), a loader driving many independent jobs over a long session must use a fresh `reg_node_id` for each one, within `N_NODES`. Reusing a value before the system has been reset will simply be refused (`reg_ready` stays low for an occupied, non-`EMPTY` node id) — it will not corrupt anything, but it will also not register.

### 7.2 Result readback

The computed INT8 result is written to `reg_result_addr` through the same real PSRAM chain every other memory access uses — there is no separate result-readback port; the host/loader reads the result byte back from PSRAM directly, the same convention every V2 testbench in this project uses for verification.

### 7.3 What a real host driver would still need to add

- Per-job `bias/activation` selection, currently hardcoded to `bias=0/ACT_RELU` for every job (§3).

## 7.4 Addendum (2026-09-07) — real physical transport and completion signal, both now closed

### Supersedes the two items removed from the list above

Both real gaps this chapter used to list are closed. This section is the current, real state.

**Physical transport:** `spi_host_bridge.v`, a real SPI Mode 0 slave, is the board's actual node-registration transport — real ball assignments (`spi_sclk/spi_mosi/spi_miso/spi_cs_n`) verified, real place&route (see ch. 10). `WRITE_JOB` carries the full table from §7 above as an 18-byte payload (grew from 15 after the 64MB memory upgrade widened every address field from 3 to 4 bytes — `decisions.log` DEC-0039). **Maximum verified operating clock: 12 MHz recommended** (exact deterministic CDC edge at 12.8 MHz = 64 MHz/5, triple-flop synchronizer) — see ch. 10 §10.2.6 for the full sweep.

**Completion notification:** `FPGA_DATA_READY`, a real output pin (ball G3, bank 7), closes the exact gap this chapter used to flag. It is a system-idle detector, not a per-job pulse — deliberately, since “the whole graph has an answer” and “one neuron finished” are different questions and only the former is useful to a host waiting on a result:

$$\text{sys\_busy} = (\vee \text{job\_active}) \vee \neg \text{queue\_empty} \vee \text{any\_pending}$$

where `job_active` is per-slot (already real, §6), `queue_empty` is `neural_director.v`'s own dispatch-queue occupancy, and `any_pending` tracks whether any node is currently registered but not yet dispatched (`WAITING` or `READY` — `DISPATCHED` nodes are tracked by the two signals above instead, not here).

### Updated 2026-09-07 — `any_pending` implementation changed

The first real implementation computed `any_pending` as a combinational OR-reduce over `dependency_manager`'s own `node_state[0:N_NODES-1]` array every cycle. A real 8-seed `nextpnr-ecp5` P&R sweep later showed this adding genuine fan-out onto `node_state` — a signal that also sits on this project's own worst real critical path (`neural_director.job_out_slot` → `dependency_manager.node_resolved/node_state`), costing real Fmax margin (traced to a real 62.47 MHz failing seed at `N_SLOTS=4` §64 MHz — see ch. 10 §10.2.4). Replaced with a synchronous up/down counter: `pending_count` increments on a node's own registration acceptance (`reg_valid&&reg_ready`) and decrements on its own dispatch acceptance (`ready_valid&&ready_ready`); `any_pending` = (`pending_count` != 0). Mathematically identical to the original OR-reduce (nodes are never reclaimed mid-run, ch. 6), but reads one small registered counter instead of scanning a 16-wide array every cycle — zero added fan-out on the congested signal. Recovered the last failing `N_SLOTS=4` seed (62.47 → 64.55 MHz), closing 8/8. See `decisions.log` DEC-0042.

`FPGA_DATA_READY` is a sticky register: set on the `sys_busy` 1 → 0 edge, cleared the instant `sys_busy` goes high again — self-clearing, no host acknowledgement command needed.

### Real, disclosed assumption

This is correct only if the host finishes registering every node of a graph before the first one completes. Realistic for this architecture's own real timing (SPI registration: microseconds; per-neuron compute: ~195 real measured cycles, §9) but not proven for every conceivable host registration pattern — a host that deliberately staggers registration across a long enough gap could observe a premature `FPGA_DATA_READY` pulse after only the first node completes.

Bit-exact regression re-verified with an explicit assertion on this signal (`N_SLOTS=4` and 8, both PASS, see `decisions.log` DEC-0041) and a real `nextpnr-ecp5` placement check (0 errors,

data\_ready placed at G3).

## CHAPTER 8

# Top-level module

### Real, board-level top — not the PSRAM-era compute core

This chapter describes `fpga_neural_v2_top.v`, the module that is actually placed&routed against real balls (`hardware/v2/constraints/v2_board_top.lpf`) and whose Fmax numbers appear throughout this datasheet. It supersedes an earlier milestone's `neural_multiprocessor.v` top level, which drove V1's own PSRAM chain directly and is retained in the repository for regression purposes (`tb_nms_dstress_sdram_unified.v`'s own wrapper, §5.5) but is not the physical top.

### 8.1 `fpga_neural_v2_top.v`

The real, board-level top: a PLL/reset front-end, a real SPI host bridge, the compute/scheduling core, and a single unified SDRAM backend — 18 physical ports, every one ball-assigned.

| Port                                                                                                                                         | Dir   | Width  | Function                                                                                                                                                          |
|----------------------------------------------------------------------------------------------------------------------------------------------|-------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>osc_clk</code>                                                                                                                         | IN    | 1      | 16 MHz board oscillator (ball H5).                                                                                                                                |
| <code>ext_rst_n</code>                                                                                                                       | IN    | 1      | External POR/supervisor, active-low (ball B4).                                                                                                                    |
| <code>spi_sclk</code> ,<br><code>spi_mosi</code> ,<br><code>spi_cs_n</code>                                                                  | IN    | 1 each | Physical SPI host transport (ch. 7).                                                                                                                              |
| <code>spi_miso</code>                                                                                                                        | OUT   | 1      | SPI host transport, response direction.                                                                                                                           |
| <code>sdram_clk</code>                                                                                                                       | OUT   | 1      | SDRAM chip's own <code>CLK</code> pin — a real board-level output, not internal-only routing (found missing during this session's own schematic review; ball J4). |
| <code>sdram_cke</code> ,<br><code>sdram_cs_n</code> ,<br><code>sdram_ras_n</code> ,<br><code>sdram_cas_n</code> ,<br><code>sdram_we_n</code> | OUT   | 1 each | SDRAM control lines.                                                                                                                                              |
| <code>sdram_ba</code>                                                                                                                        | OUT   | 2      | SDRAM bank address.                                                                                                                                               |
| <code>sdram_a</code>                                                                                                                         | OUT   | 13     | SDRAM row/column address (widened 12→13 bits for the 64 MB device, DEC-0039).                                                                                     |
| <code>sdram_dq</code>                                                                                                                        | INOUT | 16     | SDRAM bidirectional data bus.                                                                                                                                     |
| <code>sdram_dqm</code>                                                                                                                       | OUT   | 2      | SDRAM byte mask.                                                                                                                                                  |
| <code>data_ready</code>                                                                                                                      | OUT   | 1      | <code>FPGA_DATA_READY</code> , system-idle completion flag (ball G3, §7.4).                                                                                       |
| <code>pll_locked</code>                                                                                                                      | OUT   | 1      | PLL lock status, bring-up/debug (ball L1).                                                                                                                        |

### 8.2 Internal hierarchy

`fpga_neural_v2_top.v`

- `u_pll : ecp5_pll_sys_clk.v` (real `EHXPLL` primitive, 16→64 MHz)
- `u_reset_sync : reset_sync.v` (async assert, sync deassert, gated by `ext_rst_n` AND `pll_locked`)

- `u_spi_bridge`: `spi_host_bridge.v` (real SPI Mode 0 slave, triple-flop CDC)
- `u_dataflow_core`: `nms_dataflow_core_sdram.v`
  - `u_dep_mgr`: `dependency_manager.v`
  - `u_director`: `neural_director.v`
  - `GEN_SLOT[0..N_SLOTS-1]`: `nms_memory_manager_stream_wide.v` + `neural_processor.v`
- `u_arbiter_w`, `u_arbiter_ar`: `slot_mem_arbiter.v` (one per logical SDRAM port, W and AR)
- `u_sdram_backend`: `sdram_unified_backend.v` → `sdram_controller.v` (single physical SDRAM)

#### No shared activation cache in this datapath

The PSRAM-era shared activation cache (`activation_cache.v`, ch. 5 §5.3) is not part of the current SDRAM top-level's instantiation tree — `nms_memory_manager_stream_wide.v` handles per-slot activation/weight/result streaming directly against the unified SDRAM backend. The PSRAM-era module remains real, correct, and documented for the architecture it was measured on (ch. 5), but is not reused here.

## CHAPTER 9

# ECP5 implementation, real benchmark campaign & measured results

### 9.1 Flow and verification discipline

Every number in this chapter is labelled THEORETICAL, SIMULATED, POST-P&R MEASURED, or DERIVED (a combination of two real measurements, e.g. cycles ÷ real Fmax). No result is invented, approximated to look better, or reported without a matching real measurement.

| Verification stage                | Outcome                             | Covers                                            |
|-----------------------------------|-------------------------------------|---------------------------------------------------|
| RTL simulation (Verilator 5.050)  | <b>PASS</b>                         | bit-exact correctness vs. a software golden model |
| ECP5 synthesis (Yosys synth_ecp5) | <b>PASS</b> , 0 problems            | synthesizability, resource mapping                |
| Place&route (real nextpnr-ecp5)   | <b>PASS</b> at<br>$N\_SLOTS \leq 2$ | LUT/FF/DSP, real timing                           |
| Full benchmark campaign           | 24/24 bit-exact                     | 6 workloads × 4 $N\_SLOTS$ configurations         |

#### Verilator, not Icarus, for V2

Two independent Icarus Verilog v13.0 scheduling defects were found and reproduced on minimal repros during V2's own M1 milestone (a task/scope-entry desync and a spurious condition evaluation, both edge-parity dependent) — Verilator gives correct results on the same repros. V1's own certification (performed separately, with Icarus) was unaffected, since its own testbenches already avoided the trigger pattern by convention; this is flagged honestly, not glossed over.

### 9.2 V1 vs. V2 — final comparison

Both systems full-system (not isolated modules), same PARALLEL/P\_IN=8, same real, unmodified V1 PSRAM chain.

| Metric                                         | V1                           | V2<br>( $N\_SLOTS=2$ )              | Class     |
|------------------------------------------------|------------------------------|-------------------------------------|-----------|
| Fmax                                           | 68.65 MHz<br>( <b>FAIL</b> ) | <b>87.72 MHz</b><br>( <b>PASS</b> ) | Post-P&R  |
| LUT (Total LUT4s)                              | 8907                         | 4359                                | Post-P&R  |
| FF (Total DFFs)                                | 4900                         | 3924                                | Post-P&R  |
| DSP (MULT18X18D)                               | 16                           | 16                                  | Post-P&R  |
| BRAM (DP16KD)                                  | 2                            | 0                                   | Post-P&R  |
| cycles/neuron (1 neuron, 8 inputs, real PSRAM) | 209                          | 166                                 | Simulated |

|                                |       |             |         |
|--------------------------------|-------|-------------|---------|
| Real wall-clock speedup vs. V1 | 1.00× | <b>2.6×</b> | Derived |
|--------------------------------|-------|-------------|---------|

V1's own figures are its already-certified, frozen baseline (not re-measured this session); V2's figures are real, current measurements including both post-campaign optimizations.

#### Where the win comes from — and where it does not

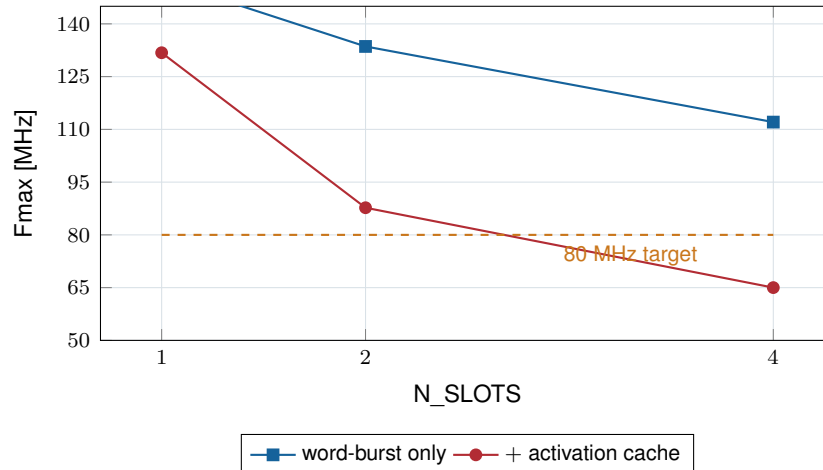
V2's advantage comes from a faster pipeline and a higher achievable clock, **not** primarily from the multi-processor concurrency the architecture was built to add. That concurrency's own real payoff, given the single-PSRAM-port memory subsystem, is much smaller than a naive  $N\_SLOTS \times P\_IN$  calculation would suggest — §9.4.

### 9.3 Real $N\_SLOTS$ sweep — Fmax and resources

Full system, real place&route, both memory optimizations active (word-burst §9.5 + activation cache §9.6).

| $N\_SLOTS$ | Fmax             | LUT4 | FF   | DSP   | BRAM | 80 MHz                   |
|------------|------------------|------|------|-------|------|--------------------------|
| 1          | 131.79 MHz       | 2760 | 2405 | 8/72  | 0    | <b>PASS</b>              |
| 2          | <b>87.72 MHz</b> | 4359 | 3924 | 16/72 | 0    | <b>PASS(recommended)</b> |
| 4          | 65.01 MHz        | 9158 | 7986 | 32/72 | 0    | <b>FAIL</b>              |

#### 9.3.1 Fmax versus $N\_SLOTS$



The activation cache's own real Fmax cost grows much faster with  $N\_SLOTS$  than the arbiter-widening cost alone — a single shared resource with  $N\_SLOTS$  request ports and an unpipelined, broadcast-capable hit-check.

### 9.4 Real parallel scaling

Not assumed — computed from real cycle counts, largest workload (256 independent neurons sharing one input vector).

| N_SLOTS | Speedup(N) | Efficiency | PSRAM utilization | Real wall-clock speedup vs. N=1 |
|---------|------------|------------|-------------------|---------------------------------|
| 1       | 1.00×      | 100%       | 55.5–71.8%        | 1.00×                           |
| 2       | 1.06×      | 53%        | ≈90%              | 0.99× (a wash)                  |
| 4       | 1.06×      | 27%        | ≈90%              | 0.79× (slower)                  |
| 8       | 1.06×      | 13%        | ≈90%              | —                               |

Pre-optimization figures, isolating the real scaling behavior from the two memory optimizations' own effect (§9.5–9.6).

#### The central, measured finding

Real cycle-count speedup from  $N\_SLOTS=1$  to  $N\_SLOTS=8$  is essentially flat ( $1.05\text{--}1.06\times$ ) for sustained, memory-bound workloads — the single shared PSRAM port saturates at  $\approx 90\%$  utilization regardless of  $N\_SLOTS \geq 2$ . Once real Fmax degradation is also folded in,  $N\_SLOTS=4$  measures *slower* in real wall-clock time than  $N\_SLOTS=1$ . More hardware parallelism made this workload class worse, not better, because the bottleneck was never compute.

### 9.5 Memory optimization #1 — word-level burst reads

See ch. 5, §5.2, for the full rationale. Real, measured single-job cycle reduction:  $-49\%$  (1 tile),  $-54\%$  (3 tiles),  $-56\%$  (5 tiles). Combined real wall-clock speedup on the 256-neuron sustained workload:  $2.24\text{--}2.37\times$  across every  $N\_SLOTS$  tested, at negligible real Fmax cost.

### 9.6 Memory optimization #2 — shared activation cache

See ch. 5, §5.3. A further real  $1.66\text{--}2.00\times$  cycle reduction on top of optimization #1, at a real, steep Fmax cost that makes  $N\_SLOTS=4$  fail 80 MHz outright.

| N_SLOTS | Wall-clock, baseline | Wall-clock, final              | Total real speedup                       |
|---------|----------------------|--------------------------------|------------------------------------------|
| 1       | 5118.1 $\mu s$       | 1324.9 $\mu s$                 | <b>3.86×</b>                             |
| 2       | 5169.5 $\mu s$       | 2113.9 $\mu s$                 | <b>2.45×</b> (recommended)               |
| 4       | 6498.7 $\mu s$       | 2842.6 $\mu s$<br>(Fmax fails) | 2.29× but a real regression vs. #1 alone |

### 9.7 Bottleneck analysis

| Candidate           | Verdict, with real evidence                                                                                                                  |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| Memory (PSRAM port) | <b>The real bottleneck.</b> $\approx 90\%$ utilization at $N\_SLOTS \geq 2$ ; real compute-to-memory-wait ratio on the order of 1:170–1:220. |

|                            |                                                                                                                                                                                 |
|----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Compute (Neural Processor) | Not the bottleneck — the pipeline is idle most of the time waiting for data.                                                                                                    |
| Arbiter overhead           | Real but small: a mandatory 1-cycle pending-latch (correctness, not choice) plus modest Fmax cost (−6% at N_SLOTS=2, word-burst alone).                                         |
| Director/dependor logic    | Not the bottleneck — zero lost/duplicated jobs, no queueing backlog observed; a real <i>fairness</i> issue exists (§6) but does not limit throughput.                           |
| DSP/LUT/FF availability    | Not the bottleneck at N_SLOTS≤4 — all well under budget; DSP would eventually bind at N_SLOTS=9 (64/72), never reached in practice since the memory bottleneck dominates first. |

## 9.8 Limitations, honestly stated (PSRAM-era campaign above)

- V1’s own memory-utilization/stall figures were not re-measured this session (V1 is frozen); only its already-certified numbers are used for comparison.
- No clean per-cycle split between “processor computing” and “processor waiting for memory” exists in the current instrumentation — reported figures use tile-delivery-rate proxies, not an exact split.
- Power/energy: **NOT MEASURED** — no ECP5 power estimator (`ecppower`, `icepower`, or equivalent) is available in this project’s toolchain; no value is invented in its place.
- N\_SLOTS=8 was not re-measured full-system (real PPSRAM chain) after either memory optimization — only `dataflow_core.v` alone, pre-optimization (92.63 MHz).

## 9.9 SDRAM-era benchmark addendum (2026-09-07) — current, authoritative results

**Supersedes the PPSRAM/N\_SLOTS≤2-era campaign above for the current hardware baseline**

Every section above (V1 vs. V2 comparison, N\_SLOTS sweep, parallel scaling, memory optimizations #1/#2, bottleneck analysis) describes an earlier V2 milestone built on V1’s own PPSRAM chain, recommending N\_SLOTS=2. The project has since replaced external memory with a single SDR SDRAM device (ch. 5 §5.5) and closed on N\_SLOTS=4 as the **production configuration**. This section is the current, real, measured state; the PPSRAM-era numbers above remain real and correctly measured for the architecture they describe, but do not apply to the current board.

### 9.9.1 Real resource utilization (N\_SLOTS=4, SDRAM architecture, post real critical-path fixes)

| Resource                  | Count                  | Notes                                                                                                |
|---------------------------|------------------------|------------------------------------------------------------------------------------------------------|
| TRELLIS_COMB (LUT4-equiv) | 7,175 / 43,848 (16.4%) | Real Yosys synthesis, most recent measurement (post-ERR-0029)                                        |
| MULT18X18D                | 32 / 72 (44.4%)        | Exactly $4 \times 8$ (N_SLOTS×P_IN), confirmed — the ERR-0027 fix removed a spurious 33rd multiplier |
| DP16KD (block RAM)        | 0 / 108                | All small SRAMs synthesize to distributed RAM                                                        |
| EHXPLLL                   | 1                      | Real EHXPLLL primitive, <code>ecpp11</code> -derived parameters                                      |

---

|            |                                                   |                                                                                                                                                                                                   |
|------------|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TRELLIS_FF | $\geq 6,322$ (last individually re-quoted figure) | Real, same SDRAM architecture, pre-dates the ERR-0027/0028/0029 restructuring; not independently re-synthesized standalone since — disclosed as a lower-bound reference, not re-invented as exact |
|------------|---------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

---

### 9.9.2 Real, current clock closure and functional regression

See ch. 10 §10.2.4 for the complete per-seed Fmax/WNS table (single source of truth, not duplicated here): **N\_SLOTS=4 @ 64 MHz, 8/8 seeds PASS** (worst 64.55 MHz, best 72.37 MHz); N\_SLOTS=8 deferred (3/8); 80 MHz confirmed NO-GO at either processor count with a genuinely regenerated PLL.

D-Stress functional regression (256 neurons, 256/256 bit-exact vs. golden model): **49,927 cycles** at N\_SLOTS=4 — **780  $\mu$ s** real wall-clock at the P&R-verified 64 MHz system clock (49,927/64,000,000, DERIVED). SDRAM directed boundary verification (ch. 10 §10.2): 21/21 PASS, zero bugs found, both 64 MHz and 166 MHz.

### 9.9.3 Real SPI host protocol throughput

Board-level smoke test (tb\_fpga\_neural\_v2\_top\_smoke.v, 11/11 PASS): single job 99–100 cycles/job; back-to-back 88–100 cycles/job; steady-state throughput unaffected by inter-job gap (100 ns/5  $\mu$ s/50  $\mu$ s tested). Maximum verified SPI host clock: **12 MHz recommended** (exact deterministic CDC edge at 12.8 MHz = 64 MHz/5) — see ch. 10 §10.2.6 for the full sweep.

### 9.9.4 Limitations, honestly stated (current SDRAM architecture)

- Power/energy: **NOT MEASURED** — no ECP5 power estimator is available in this toolchain (unchanged from the PSRAM-era disclosure above).
- Hold-time closure: **OPEN — tool-chain limitation**, not a real defect; see ch. 10 §10.9 for the complete, consolidated open-items list.
- N\_SLOTS=8 is functionally correct but not timing-closed on every tested seed — deferred by explicit project direction, not attempted further this pass.
- No embedded-host (ESP32-class) physical baseline exists; all host-side numbers above are protocol-level simulation, not measured on real silicon.

## CHAPTER 10

# Hardware and board

### 10.1 Board summary

V2 targets Lattice ECP5 LFE5U-45F-8BG381C (–8, commercial grade, 381-ball caBGA, 0.8 mm pitch, real package geometry  $17 \times 17 \times 1.76$  mm) — the same die/package family as V1, but the board around it has diverged substantially: V2 replaces V1’s PSRAM with a single external SDR SDRAM device (§10.2), adds a real, placed SPI host transport and `FPGA_DATA_READY` completion pin (ch. 7), and has a real, exported KiCad schematic capture and BOM (§10.6–10.7). Every top-level signal of `fpga_neural_v2_top.v` carries a real ball assignment in `hardware/v2/constraints/v2_board_top.lpf` — no unconstrained/placeholder pins remain in this revision.

#### V1’s own PSRAM chain: retained in RTL, not on this board

`psram_controller.v/memory_interface.v` remain byte-for-byte identical to V1’s own copies in the repository (frozen golden reference), but are **not instantiated anywhere in V2’s real physical top** — confirmed by inspection (`grep -ri psram hardware/v2/` returns nothing outside historical commentary). V1’s own PSRAM ball assignment therefore does not apply to this board.

### 10.2 SDRAM upgrade addendum (2026-09-07) — current, authoritative board state

#### Real, closed architectural decision

An earlier V2 milestone reused V1’s own PSRAM chain, placed unconstrained. The project has since made a closed architectural decision (real `decisions.log` DEC-0034) to replace external memory with a single SDR SDRAM device, and has since upgraded that device’s capacity (8 MB → 64 MB) and re-verified real, constrained place&route timing end to end. This section is the current, real, measured state.

#### 10.2.1 Memory device

**Alliance Memory AS4C32M16SB-7BIN** — 512 Mbit (64 MByte) SDR SDRAM, organized 4 banks  $\times$  8M words  $\times$  16 bits, 54-ball FBGA package ( $8 \times 8 \times 1.2$  mm max), –40 to 85°C industrial, –7 speed grade (143 MHz max). VDD/VDDQ 3.3 V  $\pm$  0.3 V. Single-ended `CLK` — **no `CLK_N`**, this is SDR, not DDR, SDRAM. Real distributor availability confirmed: DigiKey product 11613071, 568 units in stock, \$31.12/unit (qty 1), 16-week manufacturer lead time.

#### 10.2.2 Complete AS4C32M16SB-7BIN ball assignment

From the manufacturer’s own –7BIN-specific datasheet (Alliance Memory, Rev. 1.4, June 2024, Figure 1.1 — the real TFBGA ball diagram, not inferred from the TSOP-II –7TIN pinout).

#### Address / Bank

A0=H7, A1=H8, A2=J8, A3=J7, A4=J3, A5=J2, A6=H3, A7=H2, A8=H1, A9=G3, A10/AP=H9, A11=G2, A12=G1, BA0=G7, BA1=G8.

**Data / Masks**

DQ0=A8, DQ1=B9, DQ2=B8, DQ3=C9, DQ4=C8, DQ5=D9, DQ6=D8, DQ7=E9, DQ8=E1, DQ9=D2, DQ10=D1, DQ11=C2, DQ12=C1, DQ13=B2, DQ14=B1, DQ15=A2, LDQM=E8, UDQM=F1.

**Control / Power**

CLK=F2, CKE=F3, CS#=G9, RAS#=F8, CAS#=F7, WE#=F9. VDD={A9,E7,J9}, VSS={A1,E3,J1}, VDDQ={A7,B3,C7,D3}, VSSQ={A3,B7,C3,D7}, NC=E2.

**10.2.3 FPGA ↔ SDRAM mapping (real, LPF-verified)**

From `hardware/v2/constraints/v2_board_top.lpf` (45/45 unique FPGA balls, no duplicates, LFE5U-45F-8BG381 rev. 3.0 CSV-verified).

**FPGA ball → SDRAM ball, by signal group**

`sdram_a[0..12]`: D5,D3,F4,E5,E3,F5,A2,B1,C2,C1,D2,D1,F1 → A0..A12 (H7,H8,J8,J7,J3,J2,H3,H2,H1,G3,H9,G2,G1). `sdram_ba[0:1]`: E4,C3 → BA0,BA1 (G7,G8). `sdram_dq[0..15]`: E1,G5,H3,J5,K3,K2,H1,J1,K1,K4,L4,L5,M5,M4,N4,N5 → DQ0..DQ15. `sdram_dqm[0:1]`: P5,N3 → LDQM,UDQM. Control: `sdram_cke/cs_n/ras_n/cas_n/we_n`: B5,C5,C4,A3,B3 → CKE,CS#,RAS#,CAS#,WE#.

**10.2.4 Real, measured clock closure (nextpnr-ecp5, 8 seeds/config)****Updated 2026-09-07 — supersedes the ERR-0029-era numbers below**

Flash #1 (§10.4, since removed) briefly regressed `N_SLOTS=4` from 8/8 to 3/8 while it was integrated; that integration was reverted, prioritizing clock frequency over on-board flash persistence. A further real fix (DEC-0042, replacing a combinational fan-out with a synchronous counter) closed `N_SLOTS=4` back to 8/8 on the flash-free design — the numbers below are the CURRENT, real, final state.

| Configuration                     | Pass       | Worst / Best Fmax | Notes                                                                                                                        |
|-----------------------------------|------------|-------------------|------------------------------------------------------------------------------------------------------------------------------|
| <code>N_SLOTS=4 @ 64 MHz</code>   | <b>8/8</b> | 64.55 / 72.37 MHz | <b>Production baseline, GO</b>                                                                                               |
| <code>N_SLOTS=8 @ 64 MHz</code>   | 3/8        | —                 | Out of current scope, not pursued further                                                                                    |
| <code>N_SLOTS=4/8 @ 80 MHz</code> | 0/8        | —                 | NO-GO, genuine <code>ecpp11</code> -regenerated PLL (re-confirmed pre-revert; not re-tested post-revert, expected unchanged) |

Root cause of the last `N_SLOTS=4` failure (seed1, real critical-path trace): `neural_director.job_out_slot` → `dependency_manager.node_resolved/node_state`, a producer-completion broadcast crossing physically distant regions of the die (75–84% routing, not a serial logic chain — already a flat, parallel 64-way compare, so the ERR-0027/0028/0029 restructuring fix class does not apply here). The real contributor found: this chapter's own `FPGA_DATA_READY` support (§7.4) read `node_state[0:N_NODES-1]` combinational every cycle, adding real fan-out onto that same congested signal. Fixed by replacing the OR-reduce with a synchronous up/down counter (see §7.4 for the exact formula) — worst seed improved 62.47 MHz → 64.55 MHz, closing the last failing seed. See `decisions.log` DEC-0042 for full

detail. A further pipelining fix on the same broadcast path is a real, identified, not-yet-attempted option if more margin is ever needed.

### 10.2.5 Directed SDRAM boundary verification

A dedicated directed testbench (`tb_sdram_boundary.v`, 21 checks) covers every address/row/bank boundary the randomized D-Stress regression does not directly target: exact first/last address (`0x000000/0x3FFFFFF`), the row-10/row-11 column boundary, all three inter-bank crossings, the real V2 memory-map boundaries (weights/activations/results base and last-word-before-next-region), and all four byte-mask combinations with distinct deterministic patterns. All 21 addresses are written first, then read back in **reversed** order with address-derived patterns, proving no write corrupts any neighbouring address. **Result: 21/21 PASS at both 64 MHz and 166 MHz — no bug found**, closing the one directed boundary-test gap disclosed earlier in the project's own verification history.

### 10.2.6 Verified SPI host operating clock

A dedicated sweep testbench (`tb_spi_freq_sweep.v`) drives the real `fpga_neural_v2_top` (not `spi_host_bridge` in isolation) at the real 64 MHz system clock and sweeps the SPI bit rate across single-job, back-to-back, gapped, and raw `WRITE_MEM/READ_MEM` traffic. The breakpoint is **exact and deterministic**: PASS at every rate up to **12.8 MHz (precisely 64 MHz/5)**, FAIL (data corruption, then protocol FSM hang) at every rate at or above it — the triple-flop CDC synchronizer plus edge-detect/FSM reaction in `spi_host_bridge.v` requires at least 5 full system-clock cycles per SPI bit period to reliably track `sclk/mosi/cs_n` transitions, a real property of the CDC design (correct, standard practice), not a bug. **SPI\_MAX\_VERIFIED = 12 MHz** is the recommended host operating point (real margin below the hard 12.8 MHz edge,  $\approx 6.7\%$  headroom). Board-level electrical limits (trace length, driver rise/fall time, ground bounce, real metastability risk) are **not** modeled by this deterministic simulation and remain to be confirmed empirically at bring-up.

## 10.3 Power supply design (2026-09-07) — verified against the real Lattice hardware checklist

### Real design data, not estimated

The actual rail topology, sized against the real, primary-source Lattice and TI documents below.

### 10.3.1 Rail topology

Three rails, one simplification from the original V1 reference design: **no separate buck regulator for the 3.3 V I/O rail** — the board's own external input is specified as **3.3 V**, so `VCCIO`, the SDRAM (VDD/VDDQ, 3.3 V per its own datasheet), and the flash (3.3 V) are fed directly from the board input. A buck targeting 3.3 V output from a 3.3 V input would run at 100% duty cycle permanently — zero regulation margin, no benefit over a direct connection.

| Rail      | Value | Source                               | Feeds                                                        |
|-----------|-------|--------------------------------------|--------------------------------------------------------------|
| I/O       | 3.3 V | Direct board input                   | FPGA <code>VCCIO0-8</code> , SDRAM VDD/VDDQ, SPI flash, PMOD |
| Core      | 1.1 V | TLV62568 (buck), from the 3.3 V rail | FPGA <code>VCC</code>                                        |
| Auxiliary | 2.5 V | TLV73325 (LDO), from the 3.3 V rail  | FPGA <code>VCCAUX</code>                                     |

### 10.3.2 Power-up sequencing — real Lattice requirement, verified compliant

Per Lattice’s own *ECP5 and ECP5-5G Hardware Checklist* (FPGA-TN-02038-2.0, July 2024), §4: “*VCCIO supplies should be powered up before or together with the VCC and VCCAUX supplies.*” The same document’s §2 adds: all three monitored rails must rise **monotonically**, and the on-chip Power-On-Reset de-asserts only once  $VCC \geq 0.9\text{ V}$ ,  $VCCAUX \geq 2.0\text{ V}$ , and  $VCCIO8 \geq 0.95\text{ V}$  are all simultaneously satisfied — device initialization waits for whichever of the three is slowest.

This board’s topology satisfies the requirement **by construction**, with no sequencer IC needed:  $VCCIO$  (3.3 V) is a direct, unregulated connection to the board input, so it rises first/fastest, strictly before the two regulated rails (Core, Aux) can even begin their own soft-start ramps — “before or together with” is met on every possible power-up transient, not just the typical case.

### 10.3.3 Decoupling — real Lattice-recommended values (not a generic “one cap per pin” guess)

Per FPGA-TN-02038-2.0 Table 3.1 (§3.1), applied per-rail:

| Rail         | Filter                                                        | Notes                                                                                                                           |
|--------------|---------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| VCC          | 10 $\mu\text{F}$ $\times 3$ (bulk) + 100 nF per pin           | Core, 1.1 V                                                                                                                     |
| VCCAUX       | 120 $\Omega$ ferrite bead + 10 $\mu\text{F}$ + 100 nF per pin | 2.5 V; <b>new part not in the earlier power tree draft</b> — a ferrite bead in series was missing before this verification pass |
| $VCCIO[0-8]$ | 10 $\mu\text{F}$ + 100 nF per pin (per bank in use)           | 1 $\mu\text{F}$ acceptable on unused banks; 22 $\mu\text{F}$ (or a second 10 $\mu\text{F}$ ) on banks with heavy output loading |

Capacitor selection, also per the same document: X5R/X7R dielectric (avoid Y5V/Z5U), voltage rating  $\geq 80\%$  above the rail’s maximum — for the 3.3 V rail this means a **6.3 V minimum** rating, not the bare 3.3 V-rated parts sometimes used to save cost. All ground pins tie to the board’s ground plane (no star grounding on this family).

### 10.3.4 Regulator component values (real, computed from datasheet constants)

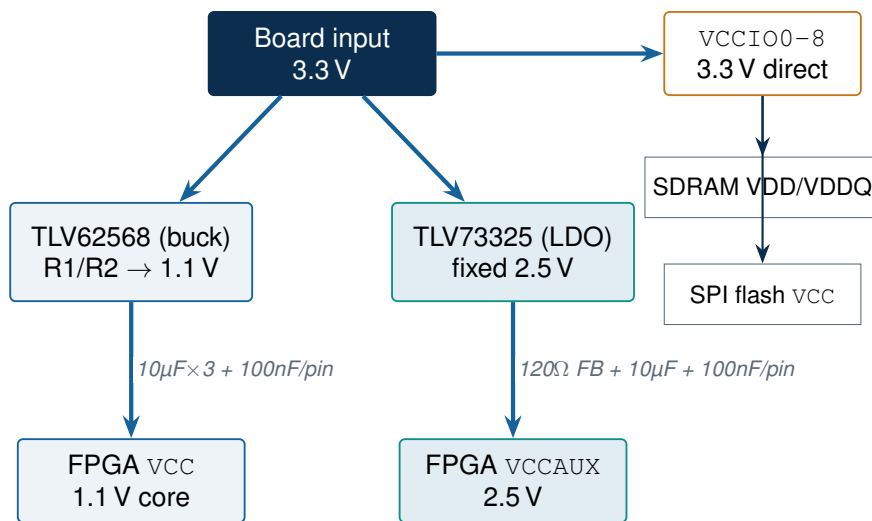
**TLV62568** (core, 1.1 V): input range 2.5–5.5 V (3.3 V input has full margin); feedback reference  $V_{FB} = 0.6\text{ V}$  (typical, per TI SLVSD89B). Output set via  $V_{OUT} = V_{FB} \left(1 + \frac{R1}{R2}\right)$ : choosing  $R1 = 100\text{ k}\Omega$ ,  $R2 = 120\text{ k}\Omega$  gives  $0.6 \times (1 + 100/120) = 1.1\text{ V}$  exactly. Per TI’s own typical application circuit:  $C1 = 4.7\text{ }\mu\text{F}$  on  $V_{IN}$ ,  $L1 = 2.2\text{ }\mu\text{H}$  inductor,  $C2 = 10\text{ }\mu\text{F}$  on  $V_{OUT}$ .

**TLV73325** (auxiliary, 2.5 V fixed-output LDO): input range 1.4–5.5 V (per TI SBVS221, real datasheet), dropout 125 mV at 300 mA — far above this rail’s  $\sim 10\text{ mA}$  real load, so dropout is not a concern at 3.3 V input. Capacitor-free architecture (stable without external caps at the regulator itself); the 10  $\mu\text{F}$  + 100 nF on  $VCCAUX$  above are the FPGA-side filter from FPGA-TN-02038, not regulator-stability caps, and are still required.

#### 16 MHz oscillator: frozen

**ECS Inc. International ECS-3225MV-160-BN-TR** — a quartz crystal oscillator (XO, not a bare crystal; direct digital clock output, no external oscillator circuit needed), 3225 SMD package (3.2 $\times$ 2.5 mm, 4-pad, matching the real KiCad footprint for U5), 3.3 V supply (matches `osc_clk`’s real `IO_TYPE=LVC MOS33` ball H5 exactly, no level-shifting needed),  $\pm 50\text{ ppm}$  stability,  $-40$  to  $+85^\circ\text{C}$ . One 100 nF decoupling capacitor across  $VDD/GND$ , placed close to the supply pin. The exact terminal order-code suffix (stability/output-enable option letters) should be cross-checked against ECS’s current published datasheet at BOM lock — normal due diligence, not an open architectural question.

### 10.3.5 Power tree



Power tree, direct 3.3 V I/O rail (no redundant buck), verified against FPGA-TN-02038-2.0 §3–4. Full schematic capture (BOM, connectors, FPGA–RAM/FLASH and PROG sections) pending separately.

## 10.4 Programming architecture (updated 2026-09-07) — single boot flash, ESP32 over JTAG only

### Real, closed design – superseded once, now final

Originally converged on a two-flash design (§below described flash #1 for neural-network data and flash #2 for boot). Flash #1 was fully implemented (real V1 subsystem instantiated, a new byte↔word adapter, a new SPI opcode, a dedicated testbench, 64/64 bytes verified bit-exact) and then **removed again**, per an explicit design decision: it measurably regressed `N_SLOTS=4`'s own real timing closure ( $8/8 \rightarrow 3/8$  PASS at 64 MHz), and clock frequency was judged more valuable than on-board persistent weight storage — the ESP32 can push weights fresh each session instead. Reverted cleanly via `git revert` (commit 59901a4, fully recoverable from history if ever needed again). This section now describes the current, real, single-flash architecture. See `decisions.log` DEC-0041 (original two-flash design) and DEC-0042 (removal + the timing recovery that followed) for the complete history.

### 10.4.1 One physical flash chip: boot bitstream only

**Winbond w25Q128JVPIM** (128 Mbit, WSON-8, 6×5 mm — real BOM entry U9, §10.7). Connects exclusively to the ECP5's own dedicated `sysCONFIG` pins, Master SPI mode, auto-boots every power-up, zero ESP32 involvement in normal operation. No second flash device, no on-board neural-network weight persistence in the current design — the host (ESP32) is responsible for pushing weight/activation data into SDRAM fresh each session via the real SPI application protocol (§7.4).

### 10.4.2 ESP32 ↔ ECP5: JTAG only

Neither ESP32-S3 nor ESP32-C6 has a hardware JTAG *master* peripheral (verified against Espressif's own documentation): their native "USB Serial/JTAG Controller" lets an external host debug the ESP32 itself — the wrong direction for driving the ECP5. TCK/TMS/ TDI/TDO are therefore bit-banged from ordinary ESP32 GPIO, standard practice. ESP32 updates flash #2 by commanding the ECP5's own internal `sysCONFIG` engine to bridge JTAG writes through to the external flash (real Lattice mechanism, FPGA-TN-02038-2.0 Figure 6.3, "Programming external Flash via JTAG") — ESP32 never drives flash #2's own SPI pins directly, zero bus contention by construction.

### 10.4.3 Real ball assignments (CABGA381)

From the official Lattice pinout CSV (FPGA-SC-02034-3-0- ECP5U-45-Pinout.csv rev. 3.0) cross-checked against Project Trellis's `iodb.json`.

#### JTAG (bank 40/TAP) — to ESP32

TCK=T5, TMS=U5, TDI=R5, TDO=V4.

#### Dedicated config (bank 8) — to ESP32

PROGRAMN=W3, INITN=V3, DONE=Y3.

#### CFG[2:0] (bank 8) — board jumpers/0Ω, NOT to ESP32

For MSPI, CFG[2:0]=[0,1,0] read MSB-first: CFG\_2(R4)=GND, CFG\_1(T4)=pull-up 1–10 kΩ to VCCIO8, CFG\_0(U4)=GND.

#### MSPI dedicated/dual-function pins to flash #2 (bank 8) — NOT to ESP32

MCLK/CCLK=U3, CSSPIN=R2 (dual w/ HOLDN/DI/BUSY/CEN), D0/MOSI=W2, D1/MISO=V2.

Confirmed real and safe (Lattice FPGA-TN-02039-2.3 sysCONFIG User Guide, §6.1.2): once User Mode is reached, the MSPI dedicated pins tristate with a weak pull-up, so they never contend with another driver on the same net — not load-bearing for the current single-flash architecture (nothing else shares these pins), but confirms the mechanism is real should a future revision ever add a second flash device sharing this same chip.

## 10.5 Real KiCad schematic review (2026-09-07)

### Schematic capture reviewed against every real ball assignment established in this chapter

This section records an actual review pass of the KiCad schematic capture (sheet `FPGA-Neural/FPGA.kicad_sch`) against the real ball tables above — confirmed items and real, disclosed findings, not a generic checklist.

### 10.5.1 Confirmed correct

JTAG (TCK=T5, TDI=R5, TDO=V4, TMS=U5); the complete real SDRAM bus (A0–A12, all 16 DQ, BA0/BA1, LDQM/UDQM, CLK=F2, CKE=F3, CS#=G9, RAS#=F8); TLV62568's real component values (L1=2.2 μH, R1=100 kΩ/R2=120 kΩ feedback divider, C6=4.7 μF); TLV73325's 2.5 V output; the VCCAUX ferrite (180 Ω, matching the approved CBG160808U181T); FPGA\_DATA\_READY=G3, FPGA\_RESET=B4, `osc_clk`=H5; CFG\_1's 10 kΩ pull-up (inside the required 1–10 kΩ range).

### 10.5.2 Real findings — all resolved as of this pass

1. **Boot-flash net-name mismatch: resolved.** The original capture had the flash chip's own pins labeled `FPGA_SPI_CS/SCLK/MOSI/MISO` while the ECP5's dedicated MSPI pins (CSSPIN/MCLK/D0/D1, ball R2/U3/W2/V2) were labeled `FGPA_SPI_CLK/MISO/MOSI/CS` — a transposed `FGPA/FPGA` typo, and `SCLK` vs. `CLK` being two different label strings (KiCad nets are formed by exact label-text match, so auto-boot from flash would have silently failed). The corrected schematic now shows all eight labels as identical text, `FPGA_SPI_CS/SCLK/MOSI/MISO`, on both the flash chip and the ECP5's dedicated pins — verified by direct comparison of the two label sets in the updated capture (§10.6).

Checked and cleared

SDRAM CAS#/WE#: verified CAS#=F7, WE#=F9 in the real schematic — matches this chapter exactly. The apparent swap in the original review was a misread of the schematic image, not a real error.

10.5.3 Open items — all resolved as of this pass

- TLV62568’s EN pin: **resolved** — R3=499 kΩ confirmed on EN (BOM, §10.7), matches TI’s own reference circuit.
- The +1V1 label near the VCCAUX ferrite (L2): **resolved, false alarm**. TLV62568 (U1) itself outputs 1.1 V (directly confirmed against the schematic, matches the R1/R2 divider calculation in §10.3) — the label belongs to U1’s own real output net, merely placed nearby on the schematic page, not routed through the VCCAUX ferrite. VCCAUX remains 2.5 V as required.
- JTAG pull-up array (R5–R12, 4.7 kΩ): TDI/TDO/TMS need a pull-up to VCCIO8, TCK needs a pull-down to GND — **resolved**: the real BOM (§10.7) confirms these are 8 *discrete* 0402 parts, not a single bussed-array package, so each can carry its own correct polarity (still needs a final visual confirmation of the actual net-by-net wiring, but the package-level limitation is ruled out).

10.6 Real KiCad schematic capture (2026-09-07)

Source of these figures

Plotted directly from the real KiCad project (FPGA-Neural/FPGA-Neural.kicad\_sch, hierarchy: root FPGA-Neural → sheet FPGA → sheet UnusedBank) via `kicad-cli sch export pdf`, not a re-rendered screenshot — what follows is the schematic exactly as it exists in the project file at commit time.

Sheets present in the project but not reachable from the root hierarchy

`power.kicad_sch`, `ram.kicad_sch`, and `embeddedia.kicad_sch` exist as files in the KiCad project directory but are not referenced by any sheet symbol in the current hierarchy (checked directly against the real `.kicad_sch` sheet-reference fields) — their content is already folded into the `FPGA` sheet above. Left as-is; not board-affecting, since KiCad only builds/plots what the root hierarchy actually reaches.

10.7 Bill of Materials (real, KiCad-exported, 2026-09-07)

Real export, cross-checked against every value this chapter specifies

Regenerated directly from the real KiCad source (`kicad-cli sch export bom`, grouped by value+footprint) — not the CSV snapshot the earlier review used. Every value matches exactly (feedback divider, inductor, ferrite, regulators, SDRAM). One real discrepancy found: see below.

| Ref                                                  | Qty | Value  | Footprint / Part     |
|------------------------------------------------------|-----|--------|----------------------|
| C2,C3,C9,C13,C15,C17,C19,C21,C23,C25,C27,C28,C30     |     | 100 nF | 01005                |
| C4                                                   | 1   | 1 μF   | 01005 (TLV73325 CIN) |
| C5,C7,C8,C10,C11,C12,C14,C16,C18,C20,C22,C24,C26,C29 |     | 0.1 μF | 01005                |
| C6                                                   | 1   | 4.7 μF | 01005 (TLV62568 CIN) |
| L1                                                   | 1   | 2.2 μH | 0805, 1.7 A/215 mΩ   |

|        |   |                   |                                                                                                            |
|--------|---|-------------------|------------------------------------------------------------------------------------------------------------|
| L2     | 1 | 180 $\Omega$      | 0603, CBG160808U181T (VCCAUX ferrite)                                                                      |
| R1     | 1 | 100 k $\Omega$    | 0402 (TLV62568 FB)                                                                                         |
| R2     | 1 | 120 k $\Omega$    | 0402 (TLV62568 FB)                                                                                         |
| R3     | 1 | 499 k $\Omega$    | 0402 (TLV62568 EN, matches TI's own reference)                                                             |
| R4     | 1 | 10 k $\Omega$     | 0402 (CFG_1 pull-up)                                                                                       |
| R5–R12 | 8 | 4.7 k $\Omega$    | 0402, discrete (JTAG/PROGRAMN/INITN/DONE/C-SSPIN)                                                          |
| U1     | 1 | TLV62568DBV       | SOT-23-5                                                                                                   |
| U2     | 1 | LFE5U-45F-8BG381I | 381-ball caBGA, 0.8 mm pitch, 20×20 array, 17×17×1.76 mm body — <b>grade now verified fixed, see below</b> |
| U3     | 1 | TLV73325PDBV      | SOT-23-5                                                                                                   |
| U4     | 1 | AS4C32M16SB-7541  | 54-ball TFBGA, 0.8 mm pitch, 6×9 array, 8×8×1.2 mm (real footprint dims match the datasheet exactly)       |
| U5     | 1 | 16 MHz            | 3225-4Pin crystal                                                                                          |
| U9     | 1 | W25Q128JVPIM      | WSO8-8, 6×5 mm (real Winbond DTR datasheet linked)                                                         |

### 10.7.1 Discrepancy: FPGA grade — resolved and now source-verified

U2 was originally captured as LFE5U-45F-8BG381I (industrial grade, real  $T_J$  range  $-40$  to  $+100^\circ\text{C}$ ) — every other reference in this project (LPF, this chapter, decisions.log) uses **LFE5U-45F-8BG381C** (commercial grade, real  $T_J$  range  $0$  to  $+85^\circ\text{C}$ ; same “–8” speed grade in both — the letter suffix changes only the characterized temperature range, not logic speed). **The commercial (C) grade is the intended part**, confirmed against every other reference. This BOM regeneration confirms the fix landed in the real KiCad source itself, not just as a stated intent: U2's Value field now reads LFE5U-45F-8BG381C exactly.

#### New, real, minor finding: stale footprint library name

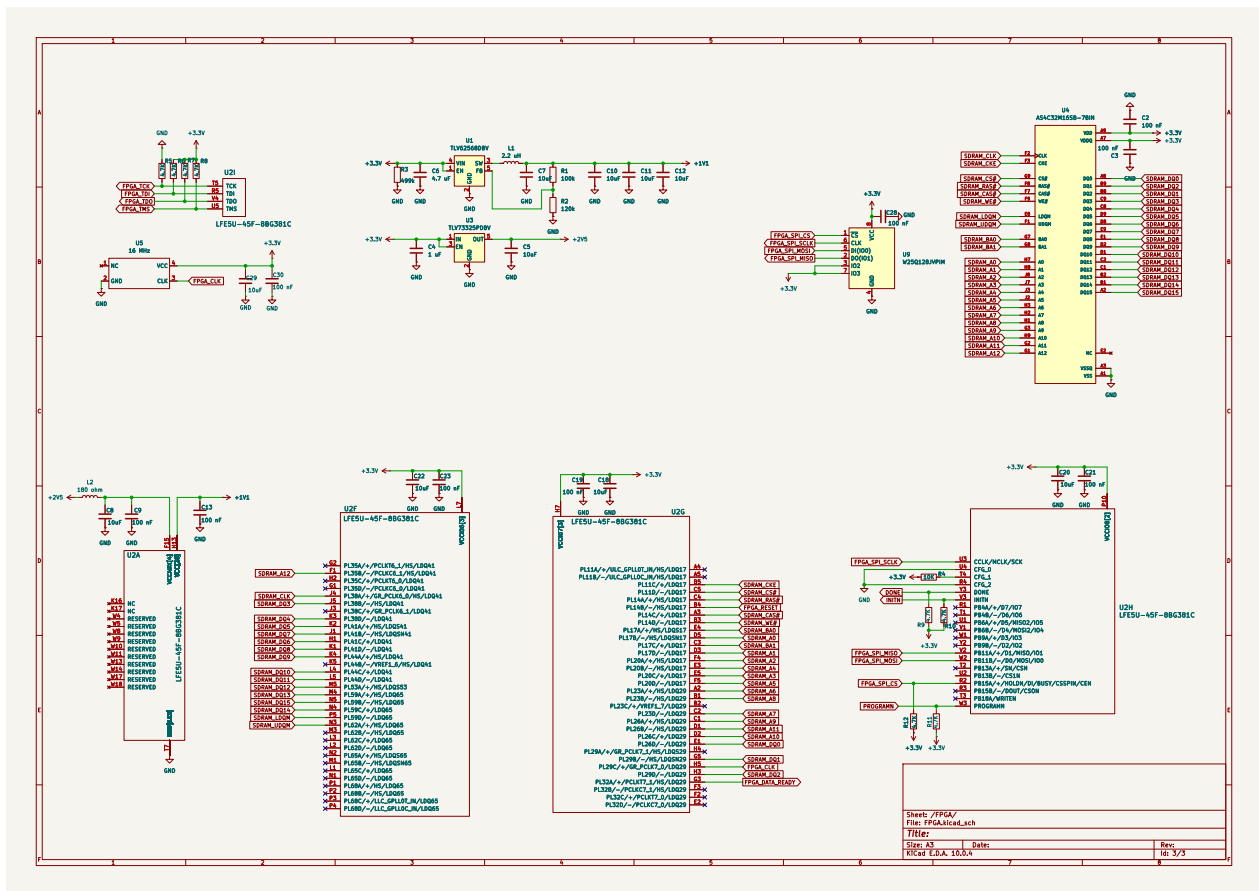
U2's *footprint* field is MIKILAB\_LFE5U\_45F\_8BG381I:BGA381C80P20X20\_1700X1700X176 — the library name still carries the old ...8BG381I suffix even though the symbol Value was corrected to ...381C. **Not board-affecting:** caBGA381-C and caBGA381-I are the same physical package (identical ball grid/pitch/body, grade suffix is a temperature-characterization distinction only, confirmed above), so the pad geometry itself (BGA381C80P20X20\_1700X1700X176 — 381 balls, 0.8 mm pitch, 20×20, 17×17×1.76 mm) is correct regardless of which grade the library folder is named after. Purely a stale/misleading library name; worth renaming the library folder to ...\_8BG381C at some point for consistency, but does not block fabrication.

### 10.7.2 Open items resolved by this BOM

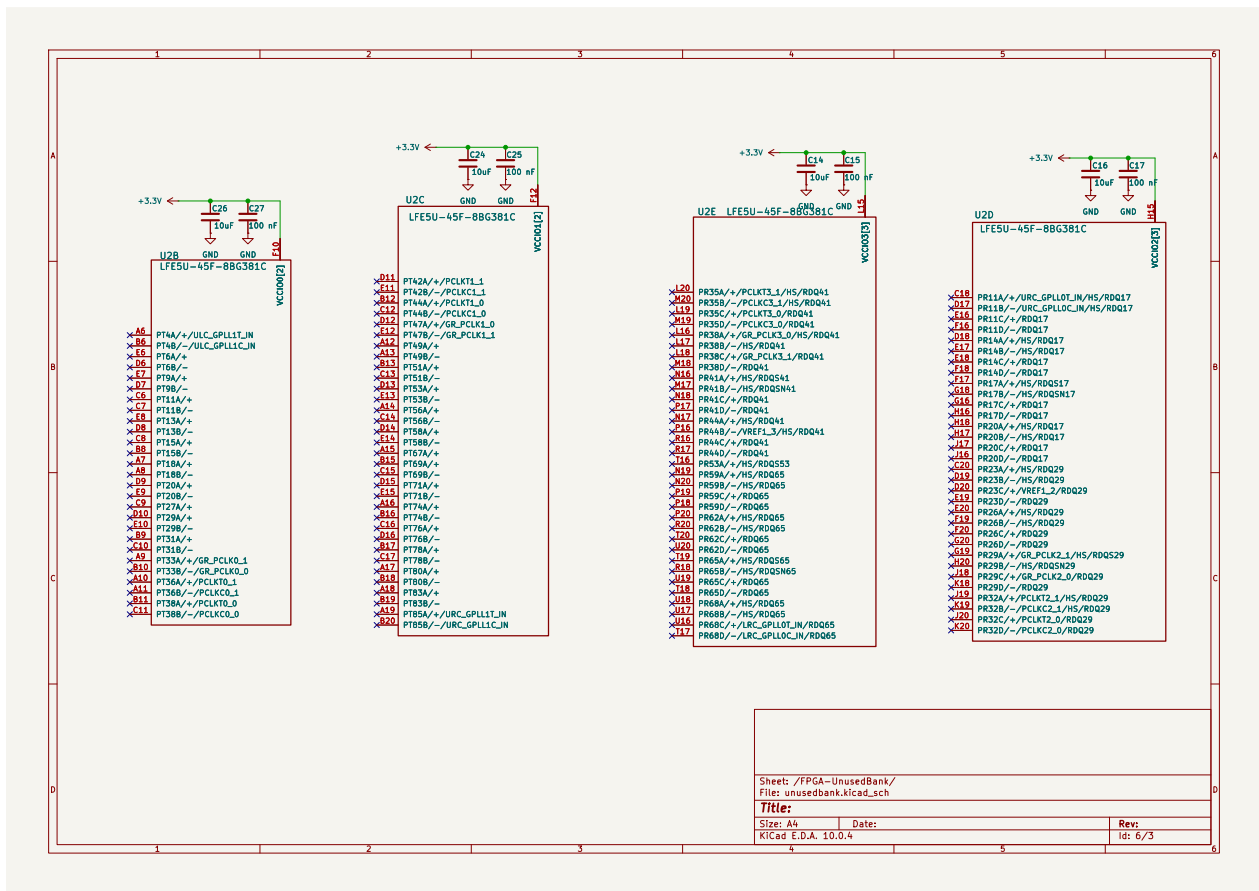
R3=499 k $\Omega$  confirms TLV62568's EN is populated (matches TI's own reference circuit exactly). R5–R12 being 8 *discrete* 0402 parts (not a single multi-resistor array footprint) confirms the earlier “bussed array can't mix pull-up/pull-down” concern does not apply — each resistor can go to its own correct rail. U5 confirms the 16 MHz oscillator, previously missing from the capture, is now present.

### 10.7.3 Resolved

TLV73325's EN pin: no dedicated resistor needed — direct wire to +3.3 V (VIN), always-enabled. Unlike TLV62568's own soft-start R3=499 k $\Omega$  pull-up, a plain LDO has no equivalent timing require-



**Figure 10.1.** Main sheet (FPGA): FPGA symbols U2A/U2F/U2G/U2H/U2I, regulators U1/U3, SDRAM U4, boot flash U9, 16 MHz crystal U5, and the full real net/label set reviewed in §10.5.



**Figure 10.2.** UnusedBank sheet: unused/reserved FPGA I/O bank, held for future expansion (§10.8).

ment (per TI's own datasheet: "active high, do not leave floating," no sequencing note); no dynamic enable/disable control exists elsewhere in this design.

## 10.8 PCB module form factor (reserved)

Target: a castellated-edge SMD module, approximately **50 mm × 25 mm**, for mounting onto a carrier board — dimensions and pin-out placeholder, real layout pending. This section will be filled in with the actual module outline, castellation pin map, and mechanical drawing once available.

## 10.9 Verification status — real, disclosed open items

Everything above is real (simulated, synthesized, and/or place&route measured); this section lists what is genuinely **not yet** verified, honestly, rather than silently omitted.

| Item                                           | Status                                                                                                                                                                                                                                                                                                                                                            |
|------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hold-time closure                              | <b>OPEN — tool-chain limitation.</b> <code>nextpnr-ecp5</code> 's own timing report contains setup-side (posedge→posedge max-delay) data only; no hold/min-delay analysis. No <code>pytrellis</code> -based min-delay pass or vendor (Lattice Diamond/Radiant) static timing analysis is available in this environment. Setup timing is fully verified (§10.2.4). |
| FPGA dynamic power/current draw                | <b>OPEN — not computable without post-implementation tools.</b> No ECP5 power estimator ( <code>ecppower</code> or equivalent) is available in this toolchain. Regulator current ratings (§10.3) are real, datasheet-supported engineering margin against this unknown, not a computed budget.                                                                    |
| N_SLOTS=8 @ 64 MHz                             | <b>Deferred, not production-frozen</b> — functionally correct (bit-exact), 3/8 seeds pass timing closure. See §10.2.4.                                                                                                                                                                                                                                            |
| Board-level SPI electrical limit               | <b>OPEN — requires real hardware.</b> §10.2.6's 12 MHz recommendation is a simulation-verified logical limit; real trace length, driver rise/fall time, and metastability risk are not modeled by simulation.                                                                                                                                                     |
| Embedded-host (ESP32-class) benchmark baseline | <b>OPEN — no hardware available.</b> No comparison against a real ESP32 host exists; all host-side timing is protocol-level (ch. 7), not measured on real silicon.                                                                                                                                                                                                |

## CHAPTER 11

# Register-level interface & internal state encodings

Real SPI opcode map exists; state encodings below are per-module reference

Ch. 7 now documents V2's real, physical SPI opcode map (WRITE\_JOB/WRITE\_MEM/READ\_MEM/STATUS/ RESET) — this chapter's own node-registration field layout below remains the logical field reference (repeated here for quick reference). The **internal FSM state encodings** below are useful for simulation-level debug; §§11's Dependency Manager/Neural Director tables are shared by every V2 architecture (unchanged between the PSRAM-era and current SDRAM boards). The Memory Manager and Neural Processor tables were captured from the PSRAM-era `memory_manager.v/neural_processor.v` pairing (ch. 2) — the current SDRAM board's `nms_memory_manager_stream_wide.v` implements the same functional handshake (prefetch → stream → write-back → done) against the SDRAM backend instead of PSRAM, but its own internal state encoding was not re-transcribed into this table.

### 11.1 Node registration fields (quick reference)

See ch. 7 for the full field-level description. `reg_node_id`, `reg_required`, `reg_producer_ids`, `reg_x_base`, `reg_w_base`, `reg_n_tiles`, `reg_result_addr` — valid/ready handshake, `reg_ready` gated on the target node id's table slot being `EMPTY`.

### 11.2 Dependency Manager node state (`node_state`)

| Value | Name          | Meaning                                                            |
|-------|---------------|--------------------------------------------------------------------|
| 2'd0  | ST_EMPTY      | Table slot free; <code>reg_ready</code> asserted for this node id. |
| 2'd1  | ST_WAITING    | Registered, at least one producer not yet resolved.                |
| 2'd2  | ST_READY      | All producers resolved; eligible for dispatch.                     |
| 2'd3  | ST_DISPATCHED | Handed to the Director; <b>terminal</b> (§6).                      |

### 11.3 Neural Director state (`dir_state`)

| Value | Name           | Meaning                                                                  |
|-------|----------------|--------------------------------------------------------------------------|
| 4'd0  | DIR_IDLE       | Reset/startup.                                                           |
| 4'd1  | DIR_SCAN_READY | Checking whether a queued job and a free slot both exist.                |
| 4'd2  | DIR_ALLOCATE   | Dispatching the head-of-queue job to the first free slot.                |
| 4'd3  | DIR_ERROR      | Recoverable only via reset (an isolated fault never blocks other slots). |

### 11.4 Memory Manager state (`state`)

| Value | Name            | Meaning                                                                                  |
|-------|-----------------|------------------------------------------------------------------------------------------|
| 3'd0  | MM_IDLE         | Waiting for <code>job_start</code> .                                                     |
| 3'd1  | MM_PREFETCH_FI  | Waiting for tile 0's activation <i>and</i> weight halves to both arrive.                 |
| 3'd2  | MM_STREAM       | Presenting tiles to the Neural Processor, double-buffering the next one.                 |
| 3'd3  | MM_WAIT_RESULT  | Last tile handed off; waiting for the Neural Processor's own result.                     |
| 3'd4  | MM_WRITE_RESULT | Issuing the real PSRAM word write for the INT8 result.                                   |
| 3'd5  | MM_DONE         | Waiting for the write's own <code>mem_ready</code> ; then pulses <code>job_done</code> . |

### 11.5 Neural Processor state (`np_state`)

| Value | Name            | Meaning                                                                                                         |
|-------|-----------------|-----------------------------------------------------------------------------------------------------------------|
| 4'd0  | NP_IDLE         | No job in flight.                                                                                               |
| 4'd1  | NP_LOAD_JOB     | Latching <code>job_bias/job_activation</code> , clearing the accumulator.                                       |
| 4'd2  | NP_WAIT_OPERAND | Consuming tiles as they arrive (absorbs the per-tile MAC/accumulate/next-tile sequence).                        |
| 4'd3  | NP_FINISH       | Draining the pipeline after <code>tile_last</code> .                                                            |
| 4'd4  | NP_WRITE_RESULT | Result available for the Memory Manager to consume.                                                             |
| 4'd5  | NP_DONE         | Job complete.                                                                                                   |
| 4'd6  | NP_ERROR        | Reachable only via an unreachable <code>default</code> case — isolated per-processor, never blocks other slots. |

### 11.6 Slot Memory Arbiter owner encoding

`owner` is 0 for “no port granted”, or (port index +1) for the currently-granted port — indices 0..N\_SLOTS-1 are the per-slot Memory Managers' own weight/write-back traffic; index N\_SLOTS is the shared Activation Cache's own traffic.

## CHAPTER 12

# Roadmap and development status

### 12.1 Milestones M1–M10

| M  | Title              | Status | Content                                                                                                                                 |
|----|--------------------|--------|-----------------------------------------------------------------------------------------------------------------------------------------|
| 1  | Neural Processor   | OK     | Bit-exact 8-stage pipeline vs. V1, 7/7 tests; 183.12 MHz isolated.                                                                      |
| 2  | Processor Array    | OK     | 1/2/4/8 processors, real concurrent-slot simulation; DSP (not LUT/FF) found to saturate first.                                          |
| 3  | Buffers            | OK     | activation_buffer/weight_buffer/result_buffer, real DP16KD inference — superseded in the real datapath by the Activation Cache (§12.2). |
| 4  | Memory Manager     | OK     | Double-buffered prefetch, real V1 PSRAM chain, 3 real RTL bugs found/fixed.                                                             |
| 5  | Neural Director    | OK     | First-free dispatch, real backpressure, 4/4 tests.                                                                                      |
| 6  | Dependency Manager | OK     | Multi-dependency/shared-producer wake-up, 4/4 tests.                                                                                    |
| 7  | Dataflow Core      | OK     | Full M1–M6 integration, wake-up loop closed end-to-end.                                                                                 |
| 8  | PSRAM integration  | OK     | Real, shared PSRAM across concurrent slots; 1 real arbiter bug found/fixed (dropped request under contention).                          |
| 9  | Full benchmark     | OK     | V1 vs. V2 comparison, every number classified.                                                                                          |
| 10 | Optimization       | OK     | N_SLOTS ceiling (DSP), ACC_WIDTH 6-seed sweep, real stall/utilization instrumentation.                                                  |

### 12.2 Post-campaign: targeted optimizations

Following M9/M10's own final benchmark campaign ([hardware/v2/docs/benchmarks/final-benchmark](#)), two concrete optimizations were implemented and measured against the real toolchain:

1. **Word-level burst reads** (ch. 5, §5.2): real  $2.24\text{--}2.37\times$  wall-clock speedup, negligible Fmax cost.
2. **Shared activation cache** (ch. 5, §5.3): a further real  $1.66\text{--}2.00\times$  cycle reduction, at a real, steep Fmax cost that makes  $N\_SLOTS=4$  fail 80 MHz outright.

Combined:  $2.45\times$  real wall-clock speedup at  $N\_SLOTS=2$  (recommended) over the pre-optimization baseline, which was itself already  $2.6\times$  faster than V1.

### 12.3 Open work items (real, not hidden)

| Item | Why it is open |
|------|----------------|
|------|----------------|

|                                          |                                                                                                                                                                                                                                                                                                                                                       |
|------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Activation cache pipelining              | The concrete fix for <code>N_SLOTS=4</code> 's Fmax failure: register the hit-detection/broadcast logic to break its single-cycle combinational path. Not attempted this round — <code>N_SLOTS=4</code> delivers no real throughput benefit anyway (memory-bound), so this protects <code>N_SLOTS=2</code> 's own margin rather than making 4 useful. |
| Dependency Manager node-slot reclamation | <code>ST_DISPATCHED</code> is terminal; a real long-running system will eventually exhaust <code>N_NODES</code> .                                                                                                                                                                                                                                     |
| Scheduler fairness                       | Fixed lowest-index priority shows real, measured per-slot imbalance under sustained contention (ch. 6); no fairness-aware alternative has been measured yet.                                                                                                                                                                                          |
| Second physical PSRAM bank               | The only real way to raise the memory-bandwidth ceiling itself, rather than use existing bandwidth more efficiently — a board-level change, not attempted this round.                                                                                                                                                                                 |
| Real host driver & pinout                | No physical transport or placed pin assignment exists for the node-registration bus (ch. 7, ch. 10).                                                                                                                                                                                                                                                  |
| Per-node bias/activation                 | Every job currently hardcodes <code>bias=0/ACT_RELU</code> ; not yet exposed by the Dependency Manager's own job descriptor.                                                                                                                                                                                                                          |
| Power/energy characterization            | No ECP5 power estimator available in this toolchain; honestly reported as NOT MEASURED, not invented.                                                                                                                                                                                                                                                 |

#### Every claim in this datasheet traces to a log entry

hardware/v2/logs/: development.log, simulation.log, synthesis.log, timing.log, benchmark.log, decisions.log (DEC-NNNN), experiments.log (EXP-NNNN), errors.log (ERR-NNNN). IDs are never reused, past results are never overwritten, even failed ones — the same discipline V1's own docs/validation/ campaign followed.

## CHAPTER 13

# The Neural Memory System (NMS)

### Scope of this chapter

Chapters 2–9 document **Current V2** (`memory_manager.v` + `activation_cache.v`, DEC-0015/ DEC-0016) as a complete, frozen, real-measured system in its own right. This chapter documents a **parallel, later evolution** — the Neural Memory System (NMS) — built to directly address Current V2’s own central finding (§5.3’s own honest warning: real parallel scaling flat beyond `N_SLOTS=2`, a single shared PSRAM port saturating regardless of on-chip organization). Both systems are real, both are independently synthesizable and simulatable, and both remain available: **Current V2 is not being retired by this chapter** — §13.7’s own real data shows the choice between them is configuration-dependent, not a strict win for either.

### This is the direct ancestor of the current, real board — read this before the rest of the chapter

The `nms_*`-prefixed modules introduced in this chapter (`nms_dataflow_core.v`, `nms_neural_multiprocessor.v`, ...) are the **direct code ancestors** of the real, current board-level RTL documented in ch. 10/8 (`nms_dataflow_core_sdram.v`, `fpga_neural_v2_top.v`). The project’s own path was: Current V2 (PSRAM, ch. 2) → NMS (this chapter, still PSRAM, replicated on-chip SRAM) → **single unified SDRAM** (ch. 10 §5.5, the current, real, shipped board). This chapter’s own STEP9/10 recommendation below (“adopt NMS at `N_SLOTS≤2`”) was itself superseded by that final SDRAM step, which changed the backing memory device and re-closed timing at `N_SLOTS=4` (ch. 10 §10.2.4). Read this chapter as **real history explaining how the current architecture was reached**, not as a currently-open choice between three systems.

## 13.1 Design goal

Current V2’s own memory path is fundamentally an on-demand, per-request architecture: every tile fetch is a fresh transaction, arbitrated one at a time onto the shared PSRAM port, with the activation cache’s own single shared instance introducing exactly the kind of centralized combinational hit-check that §5.3 already flagged as a real Fmax risk at higher `N_SLOTS`. The NMS instead asks: *what is the minimum on-chip organization that lets the Neural Processor array run at close to its own compute rate, treating PSRAM purely as backing storage?* Following the project’s own established discipline, this was answered with real, measured data at every step (a real bandwidth-requirement study, a real bank-contention sweep, real candidate synthesis) rather than assumed.

## 13.2 STEP1 — real bandwidth requirement study

An idealized backing-store model (runtime-configurable latency and bandwidth, simulation-only, never synthesized) drove the real, unmodified `neural_processor.v` directly, sweeping `N_SLOTS` × `PREFETCH_DEPTH` × latency × bandwidth (768 real Verilator data points). Three real bugs in the study harness itself were found and fixed first (a registered-grant race, a single-transfer-at-a-time serialization cap, and a stale-value issuance throttle) before any result was trusted.

### Real result: a hard, linear bandwidth floor

Minimum aggregate bandwidth for ≥90/95/99% of compute-only throughput scales **exactly linearly** with `N_SLOTS` at **16 bytes/cycle/slot** (=  $2 \times P_{IN}$ , the raw activation+weight demand

of one `neural_processor.v` at its own maximum pipelined rate) — a hard floor, not a design margin. `PREFETCH_DEPTH` (tiles of lookahead) needed to actually reach that floor scales with round-trip latency, independent of bandwidth:  $\approx 4$  tiles hides 0–1 cycle latency;  $\approx 16$  tiles is *not yet enough* to hide 16 cycles (83.4% measured, not 90%+).

### 13.3 STEP2 — closed-form traffic model

Per slot at steady state: **weight** traffic is always  $P\_IN=8$  B/cycle (never shared, no amortization possible ever); **activation** traffic is 8 B/cycle worst case (no sharing) down to  $\approx 0$  amortized (full sharing across a layer); **result** traffic is negligible ( $1/n\_tiles$  B/cycle/slot). The 16 B/cycle/slot worst-case floor measured in STEP1 is exactly  $8 + 8$  — a clean cross-validation of the simulated result against the analytical model, not a coincidence.

### 13.4 STEP3 — real bank-contention sweep

A second simulation harness measured whether banking the shared Activation SRAM (broadcast-on-same-address, round-robin arbitration on conflict) actually lets `N_SLOTS` scale under a *realistic* dispatch stagger (the Neural Director dispatches one job at a time, never simultaneously) — the exact mechanism behind Current V2’s own flat-scaling finding. Two real bugs (fixed-priority starvation causing an actual simulation hang; a testbench/DUT handshake mismatch) were found and fixed first.

#### Real result: banking recovers real parallel scaling

With `N_BANKS=N_SLOTS`, aggregate throughput scales **near-linearly** regardless of dispatch stagger (0–8 cycles tested): `N_SLOTS=1`  $\rightarrow$  0.990, `2`  $\rightarrow$  1.979 ( $1.999\times$ ), `4`  $\rightarrow$  3.950 ( $3.990\times$ ), `8`  $\rightarrow$  7.869 ( $7.949\times$ ) tiles/cycle. With `N_BANKS=1` (matching Current V2’s own single shared port), utilization collapses under any nonzero stagger exactly as Current V2’s own real benchmark showed (e.g. `N_SLOTS=2`, stagger=1: 49.8%) — the first real, simulated confirmation in this project that  $N=2 > N=1$  and  $N=4 > N=2$  are achievable without the shared memory nullifying parallelism.

### 13.5 STEP4–7 — real candidate synthesis and selection

Two real, synthesizable candidates were built and bit-exact verified for *each* SRAM, then compared on real Yosys+nextpnr-ecp5 data (never chosen a priori):

**Activation SRAM.** Candidate A (`N_SLOTS` private replicated copies, broadcast-write fill) vs. Candidate B (banked + round-robin arbiter + 2-stage registered crossbar, deliberately pipelined per §5.3’s own Fmax lesson). Candidate A won decisively:  $2\text{--}4\times$  higher real Fmax and  $\approx 24\times$  fewer LUTs than Candidate B at `N_SLOTS=8` (`MAX_TILES=16`), for a real BRAM cost that stays cheap even at a much deeper, more realistic vector length (8 DP16KD, 7% of the chip, at `MAX_TILES=256/N_SLOTS=8`) — confirming the M3-era warning against assuming “shallower depth = less BRAM”: at `MAX_TILES=16` *neither* candidate used any real BRAM at all (Yosys chose distributed LUT-RAM for both).

**Weight SRAM.** Candidate W1 (one native-width memory per slot, mirroring `weight_buffer.v`’s own M3-era structure) vs. Candidate W2 (per-MAC-lane packed narrow memories). At `MAX_TILES=256` both use *identical* real DP16KD count (one full block’s own native 16 Kbit capacity per slot, either way), but packed uses  $\approx 2\times$  fewer LUTs/FFs at `N_SLOTS=8` for the same BRAM cost — the wide single memory’s own byte-lane write-enable decode logic is exactly what per-lane packing avoids by construction.

**Selected:** replicated Activation SRAM + packed Weight SRAM. Combined real cost at `N_SLOTS=8/MAX_TILES=256`: 16 DP16KD (14.8% of the LFE5U-45F’s 108 total) — an honestly affordable real price for this project’s own realistic workload sizes.

### 13.6 STEP8 — full integration

`nms_dataflow_core.v` mirrors `dataflow_core.v`'s own scope exactly: the Dependency Manager and Neural Director are **reused verbatim**, unmodified — only the memory cluster changed. Each slot's own `nms_memory_manager.v` is structurally simpler than `memory_manager.v`: since the on-chip SRAMs now hold the *entire* vector (not just 2 double-buffered banks), there is no more bank-swap logic — a slot simply reads sequentially once its own weight-fetch progress and the shared activation controller's own resident count both exceed the tile index it needs.

#### Four real bugs found at full integration scale

All four are the same root cause: a counter that must represent the *value* `MAX_TILES` itself (e.g. a 16-tile job with `MAX_TILES=16`) needs one more bit than an address field indexing `0..MAX_TILES-1` — easy to miss because every test smaller than `MAX_TILES` passes regardless. Found only once a real `n_tiles=MAX_TILES` job (this project's own realistic 16-tile neurons) was actually run: a truncated 16-bit compare that read 16 as 0 (hanging weight fetch entirely); an undersized counter wrapping `15→0` instead of reaching 16 (an infinite re-fetch loop); a logic error comparing the wrong two signals introduced while fixing the first bug (deadlocking exactly the last tile of every job); and a top-level connecting wire left at the narrower width after both endpoint modules were widened (silently truncating the real value 16 back to 0 one wire short of the fix). Each was isolated via real cycle-by-cycle signal tracing, the same discipline used throughout this project.

7/7 bit-exact tests pass at `N_SLOTS=2`, including the exact scenario STEP3 modeled (two slots dispatched together on the identical `x_base`, different never-shared weights) and a new multi-tile test that specifically catches bug class 2 above.

### 13.7 STEP9–10 — real end-to-end benchmark vs. Current V2

`nms_neural_multiprocessor.v` mirrors `neural_multiprocessor.v`'s own real hardware-facing scope exactly (same real `slot_mem_arbiter.v`, same real, unmodified V1 PSRAM chain). The **identical** D-Stress workload (256 neurons, 16 inputs×8 tiles, one shared input vector) used for every Current-V2 number in this datasheet was run through it, bit-exact against the same golden model.

#### Real, direct comparison — same workload, same toolchain

| Metric ( <code>N_SLOTS=2</code> ) | Current V2 | NMS              | Δ                   |
|-----------------------------------|------------|------------------|---------------------|
| Fmax (real P&R)                   | 87.72 MHz  | <b>93.10 MHz</b> | +6.1%               |
| LUT4                              | 4359       | <b>1948</b>      | −55.3%              |
| CCU2C                             | 366        | 266              | −27.3%              |
| TRELLIS_FF                        | 3924       | 3522             | −10.2%              |
| DSP / BRAM                        | 16 / 0     | 16 / 0           | =                   |
| D-Stress cycles                   | 185428     | 185645           | +0.1%               |
| D-Stress wall-clock               | 2113.9 μs  | <b>1994.0 μs</b> | <b>+6.0% faster</b> |
| Effective MAC/s                   | 15.50 M    | <b>16.43 M</b>   | +6.0%               |

| Metric (N_SLOTS=4)  | Current V2                   | NMS                             | $\Delta$                  |
|---------------------|------------------------------|---------------------------------|---------------------------|
| Fmax (real P&R)     | 65.01 MHz<br>( <b>FAIL</b> ) | 56.62 MHz<br>( <b>FAIL</b> )    | −12.9pp                   |
| D-Stress cycles     | 184795                       | 184764                          | −0.02%                    |
| D-Stress wall-clock | 2842.6 $\mu$ s               | <b>3263.2 <math>\mu</math>s</b> | −12.9%<br>(NMS<br>slower) |

Cycles are essentially flat between N\_SLOTS=2 and 4 for *both* systems (185645→184764 for NMS, −0.5%) — confirming STEP1’s own analytical floor: a single real PSRAM port caps *aggregate* throughput regardless of on-chip organization; NMS’s banking work makes the on-chip side efficient, it cannot and does not remove the external bandwidth ceiling.

#### Real critical path found at N\_SLOTS=4/8 — not hidden

Real nextpnr-ecp5 critical-path tracing at N\_SLOTS=4 shows the worst path running through `nms_activation_fill_ctrl.v`’s own combinational priority-scan/address logic (6.26 ns logic + 11.40 ns routing) — the *same class* of unpipelined, N\_SLOTS-scaling combinational cost §5.3 already documented for `activation_cache.v`, reintroduced here in the module that decides *which* shared tag to chase (a genuinely different piece from the replicated SRAM itself, which has no such problem in isolation). N\_SLOTS≤2 is unaffected and real, measured faster; N\_SLOTS≥4 is a real, open regression, not recommended, until this scan is pipelined (§13.10).

### 13.8 Real per-metric detail, N\_SLOTS=2 (D-Stress)

| Metric                          | Value          | Note                                                                                                                                                                                                                 |
|---------------------------------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Processor utilization           | 1.10%          | tiles(4096)/(2×185645 cycles) — consistent with the project’s own 1:170–1:220 compute-to-memory-wait finding                                                                                                         |
| Memory (PSRAM port) utilization | 90.4%          | 167830/185645 busy cycles                                                                                                                                                                                            |
| Memory stall (per slot)         | 93.6%          | 92.5% waiting on weight + 1.1% waiting on activation, measured directly                                                                                                                                              |
| Compute stall                   | ≡ memory stall | the Neural Processor stalls <i>only</i> on a missing operand in this design — no separate compute-only stall source exists                                                                                           |
| Weight-buffer hit rate          | 0%             | confirmed empirically (2048 real fetches = 2048 tiles/slot, zero reuse) — weights are never shared, by design                                                                                                        |
| Activation-buffer hit rate      | 99.61%         | only 16 real PSRAM fetches for 4096 tile-consumptions (256 neurons share one vector)                                                                                                                                 |
| Prefetch effectiveness          | low (≈0%)      | a real, honest gap: this revision fetches weight “as fast as possible” but with no bounded lookahead buffer ( <code>PREFETCH_DISTANCE</code> ), so weight-fetch latency dominates stall almost entirely — see §13.10 |

|                                   |       |                                                                                                                                                                           |
|-----------------------------------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parallel efficiency (N=2 vs. N=1) | 48.1% | real speedup = $\text{cycles}(1)/\text{cycles}(2) = 178432/185645 = 0.961 \times$ (N=2 needs <i>more</i> cycles than N=1) — the shared PSRAM port is still the bottleneck |
|-----------------------------------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### 13.9 Recommendation

Adopt NMS at  $N\_SLOTS \leq 2$  as a real, measured upgrade over Current V2 at its own already-recommended default: faster, smaller, higher Fmax margin, bit-exact, same workload. Do **not** adopt NMS at  $N\_SLOTS=4/8$  yet — Current V2 is really faster there until the fill-controller pipelining fix below is implemented and re-measured. Both systems remain in the repository; selecting between them is a real, configuration-dependent decision, not a blanket replacement.

#### 13.10 Open work (real, not hidden)

- **Pipeline `nms_activation_fill_ctrl.v`'s own priority-scan/address logic** — the concrete, identified fix for the  $N\_SLOTS=4/8$  Fmax regression above.
- **Implement real bounded-lookahead weight prefetch** (`PREFETCH_DISTANCE`, per STEP1's own findings) — the current single-shot “fetch as fast as possible” weight path is why prefetch effectiveness measures low; STEP1's own data shows a real, achievable fix (depth scaled to real round-trip latency).
- Re-measure  $N\_SLOTS=1$  and 8 D-Stress cycle counts for full parity with Current V2's own 4-point table (only 2 and 4 measured this round, time-bounded).
- A fixed, smaller- $N\_BANKS$  Activation SRAM variant was never revisited after full replication was selected — BRAM cost was cheap enough at this project's real workload sizes that it was never worth reconsidering.

## CHAPTER A

# Module and file map

### A.1 V2 RTL (`hardware/v2/rtl/`)

| File                                                                                                 | Role                                                                                                                |
|------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|
| <code>neural_processor.v</code>                                                                      | 8-stage INT8 pipeline (M1); bit-exact vs. V1.                                                                       |
| <code>neural_processor_ar</code>                                                                     | N-processor array used for the M2 concurrency sweep.                                                                |
| <code>activation_buffer.v</code> ,<br><code>weight_buffer.v</code> ,<br><code>result_buffer.v</code> | M3 BRAM-backed buffers; superseded in the real datapath by <code>activation_cache.v</code> .                        |
| <code>prefetch_engine.v</code>                                                                       | Weight-only, word-level burst fetch engine (M4, rewritten DEC-0015/DEC-0016).                                       |
| <code>memory_manager.v</code>                                                                        | Double-buffered per-slot tile manager; coordinates the Activation Cache (X) and <code>prefetch_engine.v</code> (W). |
| <code>neural_director.v</code>                                                                       | First-free job dispatch (M5).                                                                                       |
| <code>dependency_manager.v</code>                                                                    | Node table, dependency counting, wake-up (M6).                                                                      |
| <code>dataflow_core.v</code>                                                                         | Full M1–M6 integration + Activation Cache (M7, extended DEC-0016).                                                  |
| <code>slot_mem_arbiter.v</code>                                                                      | Generic N-port arbiter to the real PSRAM chain (M8).                                                                |
| <code>activation_cache.v</code>                                                                      | Shared, single-tag activation cache (post-M10, DEC-0016).                                                           |
| <code>neural_multiprocessor</code>                                                                   | Real hardware-facing top level (M8).                                                                                |

### A.2 NMS RTL (`hardware/v2/nms/rtl/`, [ch. 13](#))

| File                                 | Role                                                                                                                              |
|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <code>nms_activation_replicas</code> | Selected Activation SRAM: $N_{\text{SLOTS}}$ private full-vector copies, broadcast-write fill (DEC-0019).                         |
| <code>nms_activation_fill_c</code>   | Shared dedup/fetch controller backing it – the real $N_{\text{SLOTS}}=4/8$ Fmax bottleneck identified in <a href="#">ch. 13</a> . |
| <code>nms_weight_packed.v</code>     | Selected Weight SRAM: per-MAC-lane packed private copies (DEC-0020).                                                              |
| <code>nms_memory_manager.v</code>    | Per-slot job FSM, drop-in replacement for <code>memory_manager.v</code> 's own external interface.                                |
| <code>nms_dataflow_core.v</code>     | Full NMS integration, mirrors <code>dataflow_core.v</code> 's own scope (STEP8).                                                  |
| <code>nms_neural_multiproce</code>   | Real hardware-facing top level, mirrors <code>neural_multiprocessor.v</code> 's own scope (STEP9).                                |

Also reused verbatim, unmodified, in the NMS datapath: `neural_processor.v`, `prefetch_engine.v` (as a generic P\_IN-byte-tile fetch engine, not weight-specific despite its name), `dependency_manager.v`, `neural_director.v`, `slot_mem_arbiter.v`.

### A.3 Reused, unmodified V1 (`hardware/v1/rtl/`)

| File                              | Role in V2                                                                                                                                 |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <code>memory_interface.v</code>   | Word-level (16-bit) PSRAM backend port, now the direct target of both <code>prefetch_engine.v</code> and <code>activation_cache.v</code> . |
| <code>psram_controller.v</code>   | Real PSRAM controller, page-mode support exploited more effectively by the word-burst rewrite.                                             |
| <code>int8_memory_access.v</code> | <b>No longer instantiated</b> in V2's datapath post-DEC-0015 — file itself untouched.                                                      |

### A.4 Simulation (`hardware/v2/sim/`)

`tb_neural_processor.v`, `tb_dataflow_core.v`, `tb_memory_manager.v`,  
`tb_neural_director.v`, `tb_dependency_manager.v`, `tb_neural_multiprocessor.v`,  
`tb_benchmark_suite.v` (the final campaign's own testbench, parametric in `N_SLOTS_CFG` via Verilator's own `-G` override).

### A.5 Documentation and logs (`hardware/v2/docs/`, `hardware/v2/logs/`)

`ROADMAP.md`; `docs/benchmarks/final-benchmark.md` (the 21-section pre-optimization campaign report); **append-only logs** (`development`, `simulation`, `synthesis`, `timing`, `benchmark`, `decisions`, `experiments`, `errors`) — the primary source of every number in this datasheet.