The Matter Stack — Illustrated Build Manual

Part A: the 4×4 build · Part B: upgrading to 8×8 · Part C: the MATD phased-array path

Step-by-step assembly with diagrams · companion to the Matter Stack design documents — July 2026

This manual builds the layered physical–acoustic–photonic display in three stages. **Part A** gets a working single-workstation display: a TinyLev-class acoustic levitator over a 4×4 ferrofluid terrain grid, with a projected photonic overlay. **Part B** upgrades the terrain to a 64-coil 8×8 grid for smoother, higher-count shapes. **Part C** replaces the levitator's fixed standing-wave trap with a steerable phased array (a Multimodal Acoustic Trap Display). Do the parts in order — each reuses the skills and most of the hardware of the one before it.

> ⚠ Read the safety notes in each part before powering anything. The three recurring hazards: flyback diodes are mandatory on every coil; ferrofluid stains permanently (keep it sealed); and 40 kHz ultrasound is intense at close range.

Figure 1 — The three-layer stack (side view)

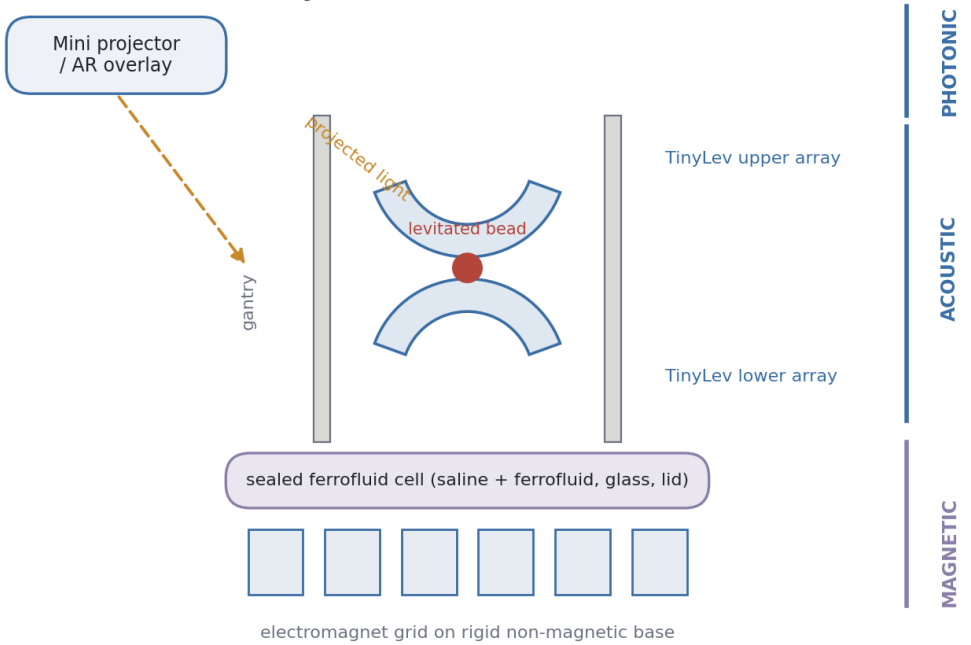


Figure 1 — How the three layers stack. Sound and magnetism share the same volume without interfering, which is what makes the display possible.

PART A Build the 4×4 Matter Stack

Budget three to five weekends and about $235–330. Build the acoustic layer first, then the magnetic layer, then integrate. Each subsystem has a milestone you should hit before moving on.

A1 · Acoustic layer — the TinyLev levitator

Follow the open-source TinyLev design (Marzo, Barnes & Drinkwater, 2017; step-by-step on Instructables as “Acoustic Levitator”). A first-time builder should consider the authorized Makerfabs kit to remove sourcing risk.

A1.1. Print the frame. Print the two-bowl TinyLev frame in PLA with a 0.4–0.6 mm nozzle, brim, no supports. Clean each transducer socket with a small file so transducers press-fit snugly.

A1.2. Test transducer polarity — the highest-leverage hour of the build. Manufacturer polarity marks are unreliable. Load the polarity-test sketch on the Arduino Nano, open the Serial Plotter, and test every transducer as in Figure 2. Mark the positive leg of each with a paint pen before any soldering.

Figure 2 — Transducer polarity test (do this for EVERY transducer)

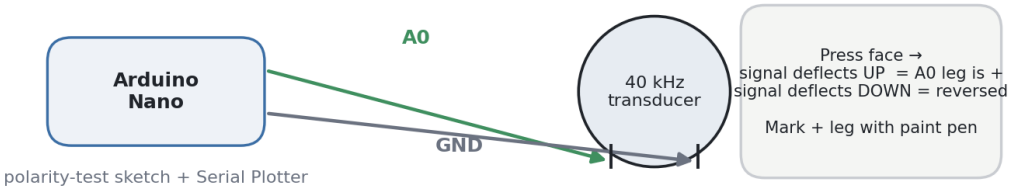


Figure 2 — Polarity test. One inverted transducer makes a dead spot; several make levitation impossible. Test all ~72.

A1.3. Populate and solder the arrays. Press transducers into both bowls with all marked (+) legs oriented the same way per bowl. Fix each with a small drop of epoxy or superglue (not hot glue). Bus all + legs together and all − legs together within each bowl; keep joints low-profile.

A1.4. Verify wiring. With a multimeter in continuity mode, confirm every + leg reaches the + bus, every − leg the − bus, and that there is no short between buses.

A1.5. Wire the drive electronics. Flash the TinyLev sketch (complementary 40 kHz square waves) to the Nano and wire it through the L298N to the two arrays, as in Figure 3. Power the L298N from the 12 V supply through the switch.

Figure 3 — Acoustic layer drive wiring (TinyLev)

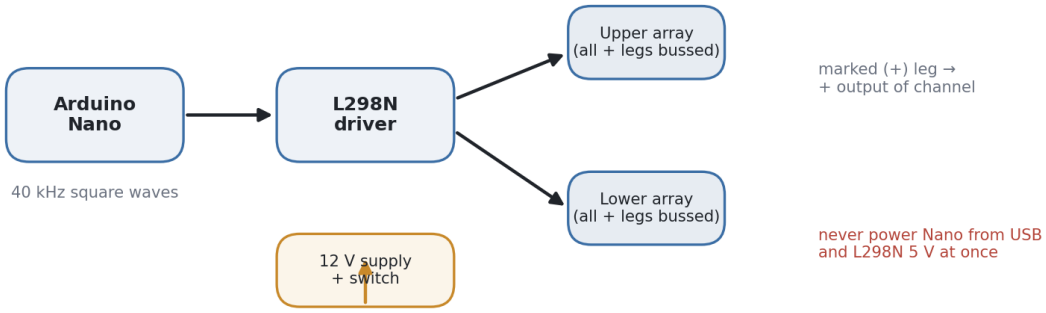


Figure 3 — Acoustic drive chain. Marked (+) leg of each array goes to the + output of its channel.

> ⚠ Power the Nano from USB **or** the L298N's 5 V regulator — never both at once.

A1.6. First levitation (milestone). Assemble the bowls facing each other at the designed separation and power on at 6 V. Drop a 1–3 mm polystyrene bead into the central region using a teabag-paper or fine-mesh scoop. It should snap into a pressure node and hover. Milestone: stable levitation for minutes, plus slow vertical movement via the driver's phase-shift feature.

A2 · Magnetic layer — ferrofluid over a 4×4 coil grid

Budget two weekends. Preparation is everything: ferrofluid stains permanently. Work gloved, over plastic sheeting.

A2.1. Build the sealed fluid cell. Make a saturated salt solution (dissolve table salt into distilled water until no more dissolves; let settle). Fill a flat glass container two-thirds with the saline, then inject 10–20 ml of ferrofluid below the surface with a syringe. Seal the lid. All interaction from here is magnetic, through the glass.

A2.2. Assemble the 4×4 grid. Mount 16 electromagnets face-up in a 4×4 grid at ~30 mm spacing on a rigid non-magnetic base (Figure 4). The cell rests 3–8 mm above the coil faces. Label coils 1–16 and route each coil's two leads to a screw-terminal strip.

Figure 4 — 4×4 electromagnet grid (top view)

16 coils · label 1–16 · leads to a screw-terminal strip

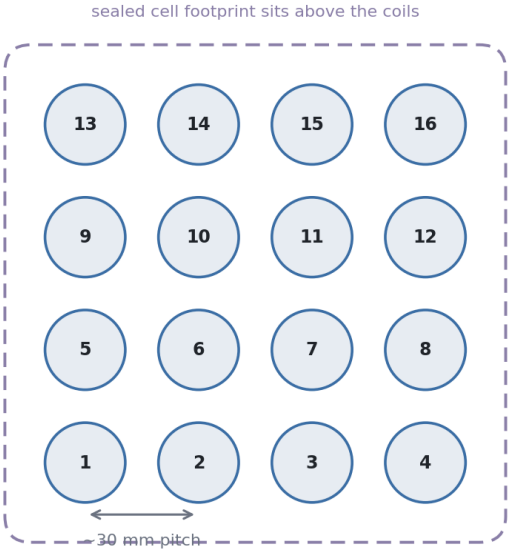
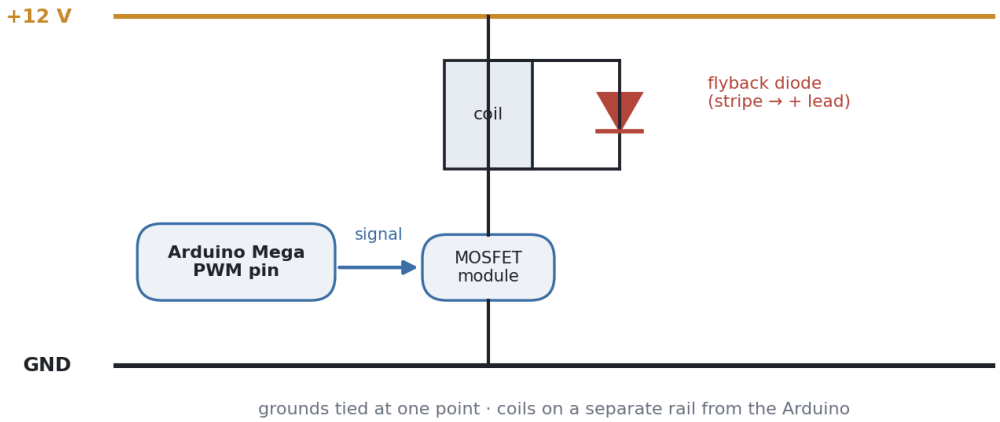


Figure 4 — 4×4 grid layout (top view). Neat labeling and lead routing saves hours of fault-finding.

A2.3. Fit a flyback diode across every coil. Solder a diode reverse-biased across each coil (stripe/cathode to the coil's + lead). This recirculates the voltage spike when a MOSFET switches off (Figure 5).

A2.4. Wire one MOSFET per coil. Coil + to +12 V; coil − to the MOSFET module's drain/load terminal; module ground common with the Arduino Mega ground. Connect each module's signal input to a PWM-capable Mega pin. Power coils from the 12 V 10 A supply on a separate rail from the Arduino, grounds tied at one point.

Figure 5 — One coil channel: MOSFET + flyback diode (repeat ×16)



Omitting even one flyback diode eventually destroys the drive electronics.

Figure 5 — One coil channel, repeated 16×. PWM cross-fading between neighbors moves a fluid bump smoothly instead of in jumps.

> ⚠ In firmware, cap duty cycle at ~60–70% and add a thermal budget (limit how long any coil exceeds 50%). Holding coils run hot — heat is this subsystem's main reliability enemy.

A2.5. First terrain (milestone). Place the sealed cell on the grid and energize one coil at 50% duty — a smooth bump should rise. Then write the traveling-bump demo: cross-fade PWM between adjacent coils so the bump orbits the grid. Measure how fast the bump can travel before it tears; that speed is the layer's spec.

A3 · Integration and the photonic layer

A3.1. Build the gantry. Suspend the TinyLev horizontally 10–15 cm above the fluid cell so the levitation zone sits over the terrain (Figure 6). Keep the Arduinos and driver boards ~15 cm from the strongest magnets.

A3.2. Two-layer milestone. From one host computer, drive both Arduinos over USB serial so a bead levitates above while a fluid bump tracks beneath it.

A3.3. Add the photonic layer. Aim a mini projector at the cell to projection-map color and labels onto the glossy fluid (or use a phone-AR app with a printed marker). Full-stack milestone: one script animates all three layers in sync — terrain rises, the bead circles the peak, the projector paints elevation color.

Figure 6 — Integration & control (one host, two Arduinos)

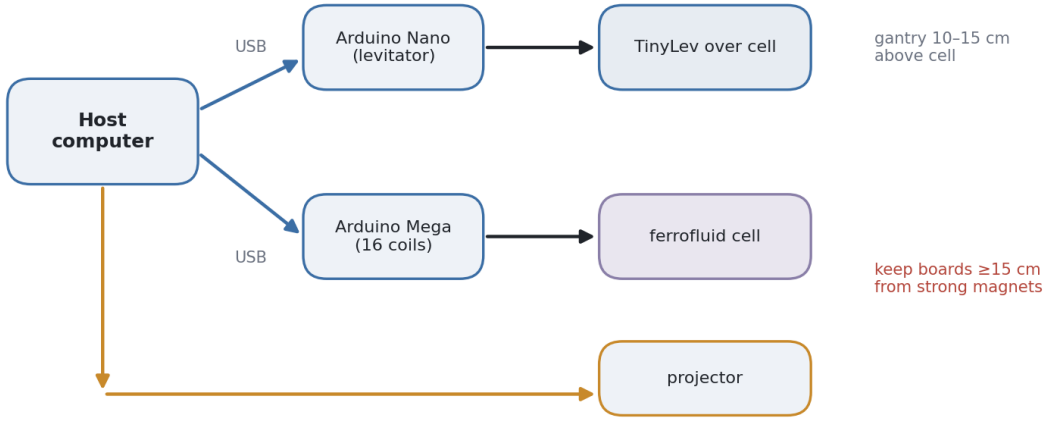


Figure 6 — Integration and control. One host drives two Arduinos; the projector maps light onto the cell.

A3.4. Optional closed loop. Add a depth camera above the cell to sense the fluid's actual shape and register the photonic overlay to measured, not commanded, geometry — the first step toward closed-loop registration.

PART B Upgrade the terrain from 4×4 to 8×8

This stage swaps the 16-coil grid for 64 coils at ~15 mm pitch, roughly doubling resolution per axis. The acoustic and photonic layers are unchanged. Three things change: the coil array, how 64 channels are addressed, and power/cooling. Budget roughly $180–320 in parts over the 4×4.

B1 · Build the 8×8 coil array

B1.1. Lay out 64 coils. Mount 64 electromagnets in an 8×8 grid at ~15 mm pitch on a new rigid non-magnetic base (Figure 7). Keep the cell standoff at 3–8 mm. The original 4×4 positions fall on every other row and column, which helps when reusing a jig.

B1.2. Fit shaped pole pieces (optional). A small shaped pole piece on each coil sharpens its field and buys a little extra separation between neighboring features. Going denser than 8×8 mostly wastes coils — below the fluid's natural spike spacing, peaks merge regardless of pitch.

Figure 7 — 8×8 grid (64 coils, ~15 mm pitch)

Figure 7 — 8×8 grid. Expect ~8–12 cleanly separable features; the ferrofluid's spike spacing, not coil pitch, sets the ceiling.

B2 · Re-address 64 channels with PCA9685 boards

An Arduino Mega has only ~15 hardware-PWM pins, so the drive path moves to dedicated PWM driver ICs.

B2.1. Chain four PCA9685 boards. Set their I²C addresses to 0x40, 0x41, 0x42, 0x43 (solder the address jumpers) and connect all four to the Mega's single I²C bus, SDA/SCL

shared (Figure 8). 4 × 16 = 64 channels.

B2.2. Keep the per-coil MOSFET + flyback. Each PCA9685 channel still drives one logic-level MOSFET, and every coil still gets its own flyback diode. The board replaces the PWM *source*, not the power stage.

Figure 8 — Addressing 64 channels with four PCA9685 boards

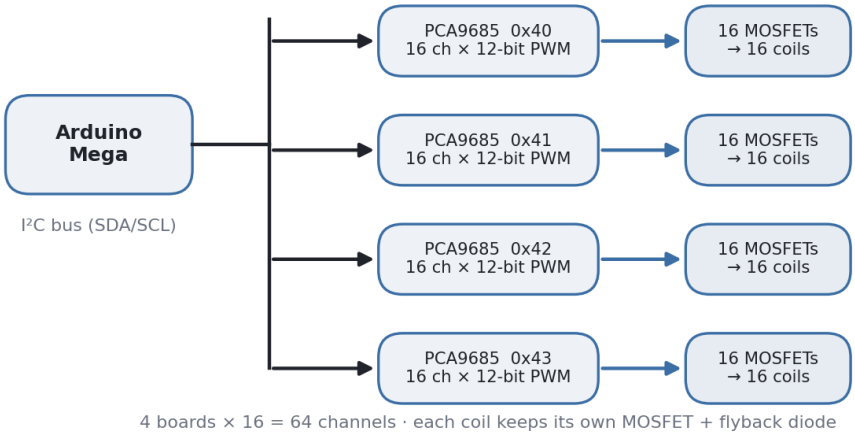


Figure 8 — Addressing architecture. Each coil keeps the MOSFET-plus-flyback channel from Figure 5.

⚠ Do not omit a single flyback diode when scaling up — with 64 coils there are 64 chances to destroy the drive electronics.

B3 · Power and cooling

B3.1. Size the supply. Sixty-four coils at 0.3–0.5 A each cannot all run at once. Fit a 12 V, 20–30 A supply, fuse the coil rail, and distribute power on 16 AWG or heavier bus bars (Figure 9).

B3.2. Add active cooling. Mount heatsinks on the coil bank and forced-air fans underneath; run the fans whenever coils are energized.

B3.3. Enforce the thermal budget in firmware. Cap duty at 60–70%, cap how many coils may run hot simultaneously, and add a per-coil dwell limit. At 4× the coil count, heat and power are the dominant constraints.

Figure 9 — Power distribution & cooling for 64 coils

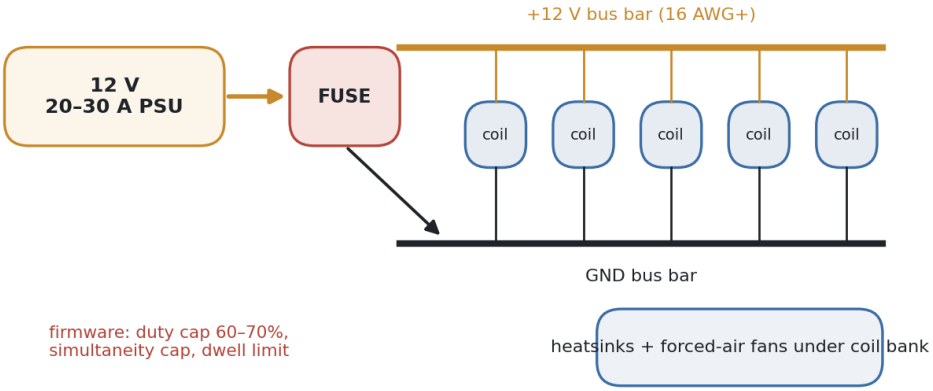


Figure 9 — Power distribution and cooling for the 64-coil bank.

B4 · Host and firmware changes

B4.1. Update the terrain output stage. The host now sends 64 duty values instead of 16; firmware maps each to its PCA9685 channel and scales to 12-bit PWM. Because authoring uses a normalized coordinate frame, existing scenes and demos run unchanged on the denser grid.

B4.2. Re-characterize. Repeat the traveling-bump test and re-measure tear-speed; the finer grid changes the achievable feature size and travel speed you can safely command.

PART C MATD upgrade — the phased-array path

The TinyLev holds a bead in a fixed standing wave and can only creep it along one axis. A Multimodal Acoustic Trap Display (MATD) gives every transducer its own phase, so the trap can be steered anywhere in the volume thousands of times per second — flying the bead at up to 8.75 m/s, coloring it with synchronized RGB LEDs, and (via amplitude modulation) adding mid-air haptics and audible sound. Expect $300–600 in parts, surface-mount or pre-fabricated driver boards, and real firmware work. Everything learned in Part A transfers directly.

> ⚠ This is an upgrade to attempt only after Parts A and B work reliably. Attempting a phased array first is the most common way the project fails for a first-time builder.

C1 · Replace the two bowls with a flat phased array

C1.1. Build or buy a flat array. Replace the facing-bowl geometry with a single flat array of 64–256 transducers on a rigid PCB or 3D-printed plate, mounted above the cell (Figure 10). The established open route is the Ultraino platform and its successor OpenMPD from the same research group.

C1.2. Reuse your polarity discipline. Every transducer still must be polarity-tested and oriented consistently — exactly the Part A procedure, at larger scale.

Figure 10 — MATD upgrade: single flat phased array replaces the two bowls

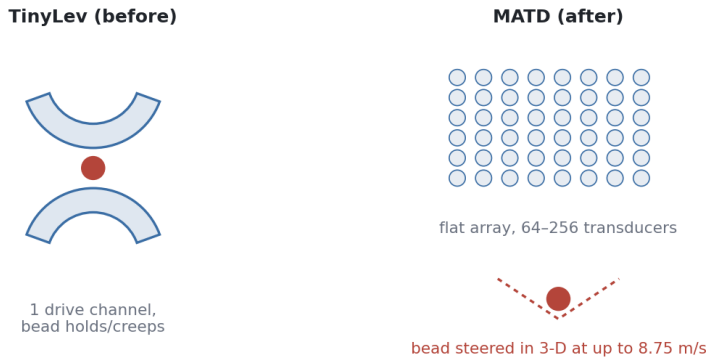


Figure 10 — From a two-bowl standing-wave trap to a flat steerable array over the cell.

C2 · Per-transducer phase control

C2.1. Add a phase controller. Drive the array from an FPGA or a microcontroller with dedicated driver boards that provide per-channel phase resolution (Figure 11). This is the core architectural change: TinyLev drove every transducer from one channel; MATD drives each independently.

C2.2. Bring up a static trap first. Before steering, reproduce a stationary trap with the array to confirm phase calibration, then command slow moves, then fast trajectories.

Figure 11 — Per-transducer phase control (Ultraino / OpenMPD class)

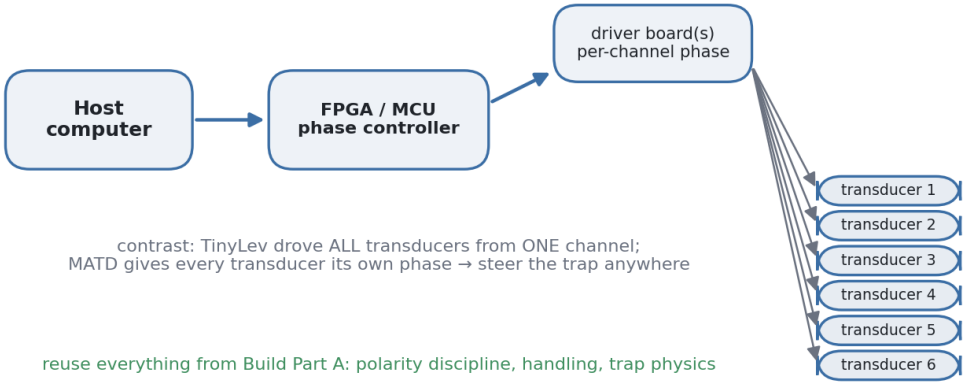


Figure 11 — Per-transducer phase control. Individual phase per transducer is what lets the trap move anywhere.

C3 · Unlock the MATD modalities

C3.1. Volumetric imagery. Fly the single bead fast enough that persistence of vision draws lines and shapes; synchronize an RGB LED to color it along the path.

C3.2. Mid-air haptics. Amplitude-modulate the same array to project focused pressure points onto a user's skin for tactile feedback.

C3.3. Audible sound. Amplitude-modulate in the audible band so the array itself emits sound — the display can voice its own soundtrack, no separate speakers.

Figure 12 — MATD modalities from one array

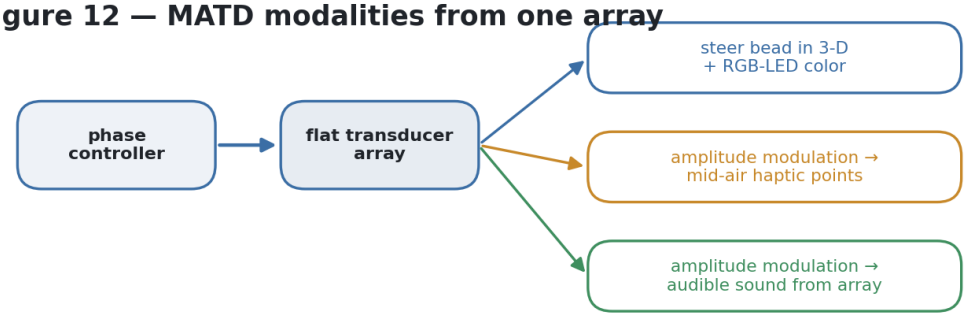


Figure 12 — One array, three added modalities: steered/colored imagery, mid-air haptics, and audible sound.

With Part C complete the stack is a full three-layer display: a persistent 8×8 ferrofluid matter layer, a steerable acoustic layer with haptics and sound, and a photonic overlay — mutually registered around the bead fiducial. That is the architecture the holodeck roadmap scales, purchasable at a bench today.