From Stone and Thread to Shared Intelligence

Fixed W0-008 release

Part IV - The Engineering Path: Develop Together, Verify in Reality

A Credit-Card-Sized Starting Point for Local Making and Global Participation

Benny (Yan) Liao - Founder, HSIMC
W0-008 · P3-D English Master · 11 November 2026

Human Authority · Local Capability · Global Participation
No One Left Behind in the GenAI Physical Era

Part IV at a glance

Parts I-III established the opportunity, the starting materials, and the collaboration model. Part IV asks the engineering question: **how can people in different places begin useful work before the final hardware is finished, while still preserving the distinction between simulation, reference validation and delivered physical evidence?**

The proposed answer is a staged engineering path built around common contracts and replaceable execution targets.

A software engineer should not need to wait for a finished custom board before testing the meaning of an operation. A hardware engineer should not need to wait for the final customer workflow before fixing power, connector and recovery constraints. A local print shop should be able to prepare a marked sample or jig while a remote vision engineer develops against a fixed image set. A reference T736 board can later replace the simulator for selected checks. Only after the intended configuration has been validated should the version be frozen for shipment. After shipment, the delivered instance should continue operating under its own identity, configuration and evidence record.

This is not an argument for pretending that simulation is hardware. It is the opposite. The purpose of parallel development is to make the boundary between each stage explicit enough that teams can work earlier without confusing what has and has not been proved.

The current HSIMC handoff provides a concrete reference. T736 is the shortest hardware route because a real module family, supplier documentation, existing design work and a source-verified simulator already exist. The evidence boundary remains strict:

- **C0-005** is a documentary hardware baseline. It contains reviewed dimensions, interface groups, pin information and conflicts that still require instance-level confirmation.
- **C0-007** is a source-verified simulator and operation ledger. It is useful for contract development and deterministic replay, not for claiming physical I/O.
- **C0-008** records a 14/14 Azure remote simulation run from 10 November 2026. It is a historical remote simulation record, not a physical T736 acceptance report.
- The real 736 instance was not re-probed in H0-001. Its present reachability, exact BOM, operating system, drivers, tool directory and Azure-to-board route remain to be bound and verified.
- A credit-card-sized complete product remains a design target. The known core module and reference board dimensions do not prove that a complete card-sized product has already been built.

The engineering principle is therefore:

> **Develop against the contract early. Bind physical capability only when the evidence supports it.**

This Part develops that principle in three steps. Chapter 10 defines the parallel development route. Chapter 11 maps the route onto the T736 reference platform. Chapter 12 follows one proposed function, `locate_marks`, from a versioned sample through simulation, local making, reference-board integration and second-location reproduction.

Chapter 10 - Build Together Before Hardware Ships

10.1 Parallel application, hardware and process development

Traditional hardware projects often force work into a sequence:

specification → board → firmware → application → test → customer

That sequence can be appropriate when the interfaces are immature or the safety case requires a tightly controlled design. It can also create long idle periods. Software waits for hardware. Hardware waits for a stable use case. Manufacturing waits for drawings. A customer sees value only after many upstream decisions have already become expensive to change.

GenAI creates an opportunity to reorganize some of this work, but only if the project has a sufficiently clear contract.

The contract does not need to describe every internal implementation detail. It needs to define what another team may rely on. For a function, that usually means:

- the name and purpose of the function;
- required inputs and accepted formats;
- output structure and units;
- errors and unsupported cases;
- expected side effects;
- timing or freshness requirements where relevant;
- the Resource or execution target to which the function applies;
- the Authority required to invoke it;
- version and compatibility information;
- evidence needed to accept the result.

Once that boundary is stable enough, several teams can begin in parallel.

Application and model work

A remote application team can build client logic, user flows, test vectors and error handling against a simulator or recorded dataset. The team does not need access to a live actuator in order to decide how a result will be represented, how duplicate requests should be handled, or how an unsupported input should be reported.

For example, a vision function can be developed against a fixed collection of images before the final camera is selected. The result is not yet a physical-camera capability. It is an algorithm and interface result under a documented input condition.

Hardware and interface work

A hardware team can work on power, recovery, connector allocation, USB host budget, serial transceivers, camera attachment, thermal constraints and mechanical packaging. Those questions can progress while the application team works on the common function contract.

The hardware team should publish the result as a Profile: what is documented, what has been measured, which interfaces conflict, what remains optional, and which evidence applies to which board revision.

Local process and fixture work

A local maker can develop the physical sample, print, jig, mount, antenna pattern, label position or fixture while the software and hardware teams work elsewhere. The local maker needs a versioned drawing or acceptance condition, not the entire software stack.

This is where local speed can become strategically important. A design revision can be turned into a visible sample in hours or a day, photographed or measured, and returned to the global team. The value is not simply low manufacturing cost. It is shorter feedback distance.

Evidence work

An evidence layer can also begin before final hardware. Test images, sample hashes, dimensions, expected outputs, known failure cases and acceptance criteria can be recorded early. When the real reference board later runs the function, the team can compare new evidence against the earlier contract instead of inventing a test after the fact.

The result is not four separate projects. It is one Work with several execution lanes tied to the same versioned boundary.

A practical parallel matrix

Lane	Can start before final hardware?	Early deliverable	Cannot claim yet
Application / GenAI	Yes	client logic, test vectors, contract handling	physical device behavior
Algorithm	Yes	result on fixed data, performance on declared dataset	live-camera performance
Hardware	Yes	interface Profile, power and connector design	final integrated product capability
Local making	Yes	sample, jig, trace, panel, fixture	repeatability beyond tested sample set
Simulator	Yes	deterministic operation semantics and replay	electrical or timing behavior of real board
Reference board	After binding	selected physical I/O and runtime evidence	delivered-unit identity or field conditions
Delivered unit	After freeze and shipment	instance-specific acceptance and local evidence	capabilities not included in that configuration

The matrix protects the central idea: **parallel work is useful only when the claim follows the evidence level.**

10.2 Remote reference validation and configuration freeze

A simulator reduces waiting. A reference board reduces uncertainty. They are not interchangeable.

The reference board stage exists to answer questions that cannot be settled from documents and host-side simulation alone.

Typical questions include:

- Does the selected operating-system image boot reliably on the actual configuration?
- Is the intended USB camera enumerated and stable?
- Does the serial path use the expected logic level and transceiver?
- Is the selected network route available under the intended security policy?

- Does the application survive restart and reconnect?
- Are operation identifiers preserved across process restarts?
- Does the device report a busy state when an exclusive Resource is already in use?
- Does cancellation produce a request, an acknowledgement and a separately verified physical stop where applicable?
- Does the evidence contain the actual board identity, software version, driver version and timestamp?

The answer should be recorded against a named physical reference instance. A successful test on an x86 Azure host is not a T736 result. A successful test on one T736 reference board is not automatically a result for every board variant. A successful physical-reference run is not automatically a result for the device later delivered to another site.

Replaceable execution targets

A useful contract can target several execution classes:

- `simulation`
- `physical_reference`
- `physical_delivered`

The same high-level operation can exist across those classes, but the capability Profile and evidence differ.

For example:

```
locate_marks(image_ref, marker_profile, calibration_ref?)
```

could be executed:

1. on Azure against a fixed image;
2. on a T736 reference board using a USB camera;
3. on a delivered T736 instance using its locally bound camera and calibration.

The application can preserve the operation meaning while the system preserves the executor identity and evidence level.

The freeze is a configuration, not a file name

A project often says “version frozen” when a software archive has been named. That is insufficient for physical delivery.

A delivery freeze should include, as applicable:

- hardware module and carrier revision;

- BOM options that affect interfaces;
- power-input assumptions and external power budget;
- connector allocation and pin usage;
- image / operating-system identifier;
- kernel, driver or device-tree changes needed by the selected peripherals;
- Runtime version;
- function-contract versions;
- application version;
- simulator/reference compatibility record;
- recovery method;
- credentials or provisioning method, without placing secrets in public artifacts;
- offline dependencies required at the destination;
- acceptance test and known limitations.

The freeze should also say what is **not** included.

This is particularly important for T736 because the documentary baseline includes optional and multiplexed functions. A connector shape or block diagram does not prove that every interface can be used simultaneously. The frozen configuration must identify the actual choices.

Freeze evidence should be readable by the next team

A freeze is successful only if another engineer can understand what was delivered.

The package should therefore contain a short START_HERE, an English collaboration Profile, source references, hashes, a supported-feature matrix, test procedure and result evidence. Large supplier source trees can remain separately indexed and retrieved only when needed.

This supports the HSIMC rule: **move the small, decision-relevant package first; retrieve large engineering sources by exact reference when the work actually depends on them.**

10.3 Shipping, local binding, recovery and continued use

Shipping hardware is not the end of the engineering path. It changes the location of the evidence problem.

Before shipment, the project can control a reference environment. After shipment, the delivered unit encounters a new network, new power conditions, new peripherals, a new operator and possibly a different maintenance model.

The delivered unit therefore needs its own binding record.

Bind the delivered instance

At minimum, local binding should record:

- delivered device Resource ID;
- hardware/BOM identity;
- software and Runtime versions;
- owner or responsible organization;
- location and local operating scope;
- connected peripherals and their Profiles;
- allowed remote operations;
- local stop/recovery responsibility;
- evidence destination;
- date and result of local acceptance.

A Card ID, QR or order number may link to this record. It should not replace the device identity.

Local operation should not depend on an always-open cloud session

A physical system should define what happens when the cloud is unavailable.

Possible local behaviors include:

- finish a safe atomic operation already accepted;
- reject new work while disconnected;
- remain in a safe idle state;
- allow a local operator to stop or recover;
- queue evidence for later upload;
- continue selected offline functions that do not require fresh cloud Authority.

The correct behavior depends on the application. A camera inspection station and a moving wheelchair do not have the same risk profile.

The July 28, 2026 MCP stateless core is helpful here because protocol continuity no longer depends on a hidden transport session. Each request can identify its context, while the application carries explicit Work, operation or Resource handles. The physical application still needs its own state machine and safe offline behavior.

Recovery is part of the product

A delivered system needs a known path back from failure.

For a T736-based node, a minimum carrier design should preserve service and recovery paths intentionally. The current documentary roadmap proposes keeping power, OTG/FEL/recovery, UART0 service, one network route, USB Host and selected serial expansion. The exact connector and pin plan remain to be frozen for the actual carrier.

Recovery evidence should answer:

- Can the unit be identified without the normal application running?
- Can a local operator return it to a known software state?
- Can a failed update be distinguished from a hardware fault?
- Can the evidence from the failed operation be retained?
- Can the next engineer reproduce the recovery method from the package?

A product that works only while the original engineer is present has not yet become a reusable capability.

Chapter 11 - The T736 Reference Path

11.1 The existing module and a minimal common interface

HSIMC is not starting the T736 path from a blank sheet. The H0-001 handoff inherits C0-005 hardware review, C0-006 remote engineering planning, the C0-007 simulator and C0-008 recorded remote simulation result.

The current shortest hardware path uses the **HelperT736C** core module as the reference. The documentary baseline records a module size of approximately **60 × 40 × 3.0 mm**, with thickness tolerance stated as ±0.2 mm, using **188 castellated peripheral pads plus 34 bottom LGA contacts**. The reviewed reference carrier is approximately **150 × 105 mm**. These are documentary dimensions, not a newly measured unit in this Part.

The credit-card plane - approximately 85.6 × 53.9 mm as an HSIMC design target - is therefore a packaging challenge, not a proven current product form.

The engineering question is not “can the reference carrier be shrunk by scaling the drawing?” The question is: **which functions must remain on the common carrier, which can move into replaceable adapter modules, and which belong in the cloud or external equipment?**

A proposed minimum common carrier

The current handoff proposes a small common carrier that preserves, at minimum:

1. defined power input and protection;
2. USB0 / OTG / FEL or equivalent recovery path;
3. reset / boot service functions required for recovery;
4. UART0 service access;
5. one deliberate network route;
6. USB Host with a declared external power budget;
7. intentionally assigned serial expansion;
8. the mechanical, thermal and connector conditions needed for the selected product form.

This is a design scope, not a frozen schematic.

It deliberately avoids trying to expose every possible T736 function on a credit-card-sized board. Optional high-speed, display, camera and expansion functions can be added through product-specific carriers or adapters when a real Work requires them.

Power must be specified by configuration

The documentary baseline contains power statements that need reconciliation. The reference carrier accepts 12 V and converts to the module's 5 V input; the reviewed documents contain conflicting interpretations of a 1.5 A statement, and some PK-bank voltage descriptions also require BOM/configuration confirmation.

Therefore a T736 Profile should not contain a single vague line such as "5 V / 1.5 A supported." It should separate:

- accepted carrier input range;
- module supply requirement;
- regulator rating and thermal condition;
- peak current assumptions;
- USB/peripheral power budget;
- start-up or transient condition;
- actual measured result on the named configuration.

This distinction matters when a camera, UWB radio, storage device or serial adapter is attached. The base module can boot successfully while the complete system still has an inadequate peripheral power budget.

Interface names are not electrical guarantees

A UART signal on the SoC side is logic-level serial. RS-232 and RS-485 are different electrical interfaces and require appropriate transceivers. A connector that resembles miniPCIe does not prove PCIe signaling. The reviewed T736 carrier includes cases where connector form and routed protocol differ. Likewise, multiplexed PHY resources mean that not every block-diagram function should be assumed to operate simultaneously.

The Profile must therefore describe:

logical interface → electrical layer → connector → driver → intended device → evidence

That chain is much more useful to a remote engineer than a connector photograph alone.

11.2 Adapters for cameras, tags, positioning and controllers

A common T736 node becomes more useful when device-specific complexity can move into small, explicit adapters.

The adapter principle is simple:

> **Keep the function contract stable; replace the physical adapter when the local equipment changes.**

Camera adapters

For the first vision path, a compatible USB camera is a practical candidate because it reduces the number of unresolved CSI-specific issues during early development. This is a route choice, not a claim that USB is always superior.

A camera adapter Profile should record:

- vendor/model and USB identity;
- resolution and supported pixel formats;
- frame rate used by the application;
- lens and focus condition;
- mount position;
- exposure / lighting assumptions;
- capture timestamp behavior;
- calibration version when metric coordinates are reported;
- driver and kernel dependencies;
- observed CPU/memory/latency on the actual T736 configuration.

Switching from a synthetic image to a real camera therefore creates a new evidence gate. The algorithm can remain the same version, but the camera, capture path and calibration become part of the result.

NFC, UHF RFID and UWB adapters

These technologies should not be collapsed into a single “wireless ID” function.

- **NFC** can provide a short-range phone or device interaction path.
- **UHF RFID** can support item identification at greater range with compatible readers and tags.
- **UWB** can provide ranging, angle or positioning observations depending on the specific hardware and geometry.

A T736 node may connect to any of them through USB, serial, network or another supported interface. The actual module, driver, antenna and firmware determine capability.

The adapter should return an observation with its own time, quality and source rather than silently converting an old reading into current truth.

Machine and motion-controller adapters

The T736 should not be required to replace deterministic motor or safety control that is better handled by an existing PLC, servo controller or dedicated motion controller.

A useful architecture is:

GenAI / client → HSIMC Work → T736 adapter / local supervisor → deterministic controller → physical machine

The T736 layer can translate a versioned operation, check local conditions, collect status and package evidence. The motion controller remains responsible for the real-time behavior for which it was designed.

For old equipment, an adapter may use RS-232, RS-485, Ethernet, discrete I/O or a vendor protocol. Each electrical conversion and timing assumption should be part of the Profile.

The adapter is also a commercial boundary

A local integrator can contribute value without redesigning the entire platform. If a factory has a legacy cutter, a hotel has an existing cart, or a workshop has a laser with a proprietary controller, the local contributor can build and validate the adapter that connects that Resource to a shared Work contract.

This turns system integration into a modular capability that can be attributed, versioned, tested and paid for.

11.3 Documented capability, simulation and physical evidence

The T736 path is useful only if the evidence ladder remains visible.

A simple ladder is:

Evidence level	What it proves	Example
Documented	supplier/reviewed documentation states a capability or constraint	pin group, module dimension, block diagram

Evidence level	What it proves	Example
Simulated	application contract behaves under declared simulated conditions	C0-007 operation replay
Remote simulated	another client reaches the simulator through a remote path	C0-008 14/14 Azure record
Physical reference	a named reference board performs the selected function	camera enumerates; UART operation succeeds
Physical delivered	the delivered unit passes local acceptance	customer site run on delivered identity
Operational	the capability performs repeatedly under real Work conditions	field evidence over time

The ladder prevents a common failure in AI-assisted engineering: a plausible explanation becomes a design assumption, the design assumption becomes a presentation, and the presentation is later remembered as a tested fact.

C0-007 remains valuable precisely because it is limited

The C0-007 source-verified simulator exposes `hsimc_get_profile`, `hsimc_sim_operation` and `hsimc_get_operation`, plus a simulated Profile resource and SQLite operation records. It supports duplicate handling and exclusive-group conflict behavior for its declared test conditions.

It also reports physical targets as unbound.

That is the correct behavior for a simulator. It allows application engineers to work on operation semantics without inventing a physical result.

The current C0-007 initialization declares protocol version `2025-11-25`. Therefore Part III's discussion of MCP 2026-07-28 is a target architecture, not evidence that the simulator has already migrated. A future migration should be implemented and tested, not achieved by editing a version string.

C0-008 is a remote-path record, not hardware acceptance

C0-008 records that an engineering client called the Azure simulator path and passed 14/14 checks on 10 November 2026. That is useful evidence that the remote simulation route worked under the recorded conditions.

It does not prove:

- current Azure availability;
- current T736 reachability;

- hardware I/O;
- camera compatibility;
- current client compatibility with the 2026-07-28 protocol;
- delivered-product behavior.

A white paper gains credibility by saying this plainly.

The next physical-reference gate

When a real T736 reference instance is reachable and authorized, the first physical-reference gate should remain intentionally narrow.

A reasonable sequence is:

1. identify the board and configuration;
2. record hardware, OS and Runtime versions;
3. verify recovery/service path;
4. verify the selected network route;
5. bind one permitted peripheral;
6. run one read-only or low-risk observation function;
7. persist operation and observation evidence;
8. test restart / reconnect / duplicate handling;
9. only then expand to a bounded physical action.

The project does not need the entire supplier BSP to be mirrored into every engineer’s workspace before this can begin. The exact source dependency can be retrieved when a driver, device tree or kernel change actually requires it.

Chapter 12 - One Function, Many Makers

12.1 The proposed `locate_marks` contract

A platform can become abstract very quickly. A small function makes the collaboration model concrete.

HSIMC therefore proposes `locate_marks` as a reference engineering function for the next phase.

The function answers a narrow question:

> **Given a versioned image and marker definition, where are the expected marks, and what evidence supports the result?**

This is useful in printing, card making, laser alignment, fixture verification, packaging, labels and other local production tasks.

It is deliberately not a machine-motion command.

Why this function is a good first reference

It connects several existing capabilities without requiring the final product to be complete:

- a designer can create the mark pattern;
- an ordinary printer can produce a sample;
- a local maker can build a jig;
- a phone or camera can capture the sample;
- a remote vision engineer can develop the detector;
- Azure can run fixed-image tests;
- T736 can later run the same function with a real camera;
- an operator can compare the returned evidence against the sample;
- a laser or other machine can consume the result only through a separate, authorized action contract.

This separation makes it possible to develop vision and physical process work in parallel without allowing a detection result to become an unintended actuator command.

Candidate input contract

A first candidate request might contain:

Field	Purpose
`work_id`	links the request to the engineering or business Work
`operation_id`	makes this specific invocation explicit and queryable
`target_resource_id`	identifies simulator, reference or delivered executor
`image_ref`	references the versioned input image or capture
`marker_profile`	defines which marks should be located
`calibration_ref`	optional calibration when metric coordinates are required
`algorithm_version`	identifies the detector implementation
`authority_context`	identifies the current permitted requester / scope

The exact wire schema remains to be implemented and tested. The important point is that the operation does not rely on hidden conversation memory to know which image, profile or executor is intended.

Candidate output contract

A useful result can include:

- `operation_id`;
- `observation_id`;
- executor identity and execution class;
- capture or source timestamp;
- algorithm version;
- detected marker IDs;
- pixel coordinates;
- confidence or quality values;
- calibration reference if metric coordinates are emitted;
- input-image hash;
- evidence references;
- warnings or unsupported conditions.

If no valid calibration is available, the function should report pixel coordinates rather than quietly inventing millimetres.

If the image is stale, damaged or unsupported, the function should say so.

Observation, not control

The result means: **this executor, using this algorithm and this input, observed these marks at these locations.**

It does not mean:

- the material is safe to process;
- the machine is aligned;
- the operator has authorized motion;
- the laser should fire;
- the result is within the customer's tolerance;
- a later image will contain the same marks.

Those are separate decisions and evidence steps.

This is exactly the kind of boundary Part III called a shared contract.

12.2 From synthetic images to a real sample

The first useful implementation can begin before a live camera is bound.

Stage A - fixed synthetic and generated inputs

Create a small versioned dataset:

- nominal sample;
- rotated sample;
- scale variation;
- one missing marker;
- partial occlusion;
- low contrast;
- unsupported format;
- wrong marker profile.

Each input has a known expected result. The purpose is to validate the function semantics, not to claim field accuracy.

A remote engineer can implement the detector and produce deterministic evidence against that set.

Stage B - locally printed sample and jig

A local print shop or workshop receives the versioned artwork and fixture dimensions. It prints a real sample, places it in the jig, and captures images under declared lighting and camera conditions.

The local deliverable should include:

- sample version;
- print method / material description;
- jig version;
- sample photograph;
- measured dimensions relevant to acceptance;
- image files and hashes;
- visible defects or deviations;
- operator / location / time evidence where appropriate.

The remote detector then runs on the real image.

Now the team can distinguish two failure classes:

1. the physical sample or capture does not match the expected condition;
2. the algorithm fails on an acceptable physical sample.

Without this separation, software and manufacturing teams can easily blame each other without evidence.

Stage C - T736 reference camera path

Once the reference board is reachable, bind a specific camera Profile.

The same function is executed from a live or locally captured frame. The evidence now adds:

- T736 Resource identity;
- hardware and Runtime version;
- camera identity and driver;
- capture settings;
- timestamps;
- processing latency;

- CPU / memory observations if relevant;
- calibration identity;
- result evidence.

The algorithm may be unchanged from Stage A. The evidence level is different because the capture and runtime are physical.

Stage D - bounded downstream use

Only after the observation is trusted should another Work consume it.

For example, a laser-alignment preparation function could transform detected marks into a proposed coordinate transform. A human or local controller could validate the proposal. A separate authorized machine action could then execute under its own operation ID.

This creates an auditable chain:

sample → image → locate_marks observation → alignment proposal → acceptance → machine action → output inspection

Each step can be owned by a different contributor while remaining part of one Work.

12.3 Independent reproduction at a second location

The strongest evidence that a capability is reusable is not that the original team can repeat it. It is that another team can reproduce the function from the package.

A second-location reproduction should therefore be an explicit engineering gate.

What the second team receives

The team should receive a compact package containing:

- START_HERE;
- function contract;
- supported / unsupported matrix;
- sample artwork;
- jig drawing or construction note;
- fixed test images;
- algorithm or client package within the permitted distribution scope;

- environment / dependency instructions;
- expected results;
- evidence template;
- source references and hashes.

Large supplier archives remain referenced separately unless the reproduction actually needs them.

What the second team must return

The return package should include:

- which instructions were sufficient;
- which instructions were ambiguous;
- environment and tool versions;
- the result for each test vector;
- real sample or image evidence if produced;
- deviations from the expected result;
- time spent setting up and running the work;
- any local changes required;
- a clear statement of what was and was not reproduced.

This produces more than a pass/fail result. It measures the quality of the collaboration interface.

Reproduction time is a product metric

One useful metric for a GenAI-native physical engineering system is:

> **time from receiving a versioned package to independently reproducing the agreed function.**

A second metric is the number of clarifications required.

A third is the amount of hidden local knowledge that had to be rediscovered.

These metrics expose a different kind of engineering debt. A design may work perfectly in the original workshop but still be difficult to transfer. That transfer difficulty is part of the product.

Many makers, one evidence chain

The `locate_marks` example shows how a simple function can connect many participants:

Contributor	Possible contribution	Evidence
Designer	marker artwork / version	design hash
Printer	physical sample	print / material record
Local maker	fixture / jig	dimensions, photos, version
Vision engineer	detector	algorithm version and test results
Cloud engineer	simulator / service	execution records
T736 engineer	camera adapter	board, driver and capture Profile
Operator	local confirmation	observation / acceptance record
Machine integrator	downstream adapter	action contract and result evidence
Second site	independent reproduction	reproduction report

None of these participants needs to own the entire product.

That is the engineering meaning of the wider HSIMC invitation:

> **Bring your existing capability. Deliver one reproducible function.**

The system becomes more powerful as the functions become easier to combine, but combination should never erase the evidence boundary of each contribution.

From one function to a capability network

After one function is reproducible, the same method can expand:

- `read_tag`
- `write_tag`
- `measure_range`
- `get_machine_state`
- `prepare_job`
- `reserve_slot`
- `verify_output`
- `request_pickup`
- `confirm_handover`

Each function can be implemented by different local Resources and different technologies.

The common requirement is not identical hardware. It is a contract that another participant can understand, an Authority model that limits action appropriately, and Evidence that shows what happened.

Part IV conclusion - The product is the reproducible path

The engineering opportunity in the GenAI physical era is not simply to make one more smart device.

It is to shorten the distance between a plan and a verified outcome while allowing more people and more existing facilities to participate.

That requires a path where:

- software can begin before custom hardware is finished;
- hardware can stabilize interfaces before every application detail is known;
- local makers can turn drawings into real samples quickly;
- a simulator can prove semantics without pretending to be hardware;
- a reference board can prove selected physical behavior;
- a frozen configuration can be shipped with enough evidence for another engineer to understand it;
- the delivered unit can be bound locally and continue working without the original project team standing beside it;
- a second team can reproduce the function and expose hidden assumptions.

T736 is the current shortest hardware reference for this route, not the destination of the entire ecosystem. The credit-card-sized form is a design challenge, not a prerequisite for useful collaboration. The deeper product is the **reproducible engineering path** connecting people, tools, resources and evidence.

This is the bridge from Part III's collaboration model to Part V's question of value: **when the path becomes reproducible, what economic, social and environmental outcomes does it create, and how should those outcomes be measured without double counting?**

Evidence and source notes for Part IV

HSIMC engineering evidence

- H0-001 Card Phone and Cloud Development handoff, 11 November 2026.
- C0-005 T736 documentary hardware standardization baseline, 9 November 2026.
- C0-006 remote engineering plan, 9 November 2026.
- C0-007 source-verified simulator and recorded simulator tests, 10 November 2026.
- C0-008 recorded Azure remote simulation result, 10 November 2026: 14/14 checks in the recorded report.

External context

- OpenAI, *Accelerating engineering for the physical world*, 3 November 2026. The article describes work across mechanical engineering, electrical engineering, software, simulation and manufacturing, and invites engineering and manufacturing collaborators. HSIMC's architecture and claims remain independent.
- Model Context Protocol, *The 2026-07-28 Specification*, 28 July 2026. The release introduces a stateless protocol core, self-describing requests, header-based routing, an extensions framework and related authorization changes. Application state may still be carried through explicit handles.

Evidence discipline

This Part distinguishes proposed architecture, documentary hardware information, simulated evidence, recorded remote-simulation evidence and future physical validation. No statement in this Part should be read as claiming a currently reachable T736, a completed credit-card-sized product, a live-camera acceptance result, or a delivered-unit test unless separately supported by dated physical evidence.