FIG. 1 — Bedside Closed-Loop Safety Architecture

Physiological sensors **(100)** place real-time telemetry onto a **secured CAN-bus (140)**.
The **real-time arbitration engine (120)** reads this bus, consults a local **rules database (110)**, and sends deterministic control packets to **actuator drivers (130)** that operate infusion pumps, ventilator valves, stimulators, and other patient-facing hardware on the same bus.

A dashed link to an **optional cloud-replication node (150)** mirrors logs and firmware updates without touching the safety-critical path.
A separate dashed **watchdog return path (160)** lets the sensor subsystem veto or halt further actuation if heartbeat, CRC, or power-rail anomalies are detected, thereby completing the safety loop.

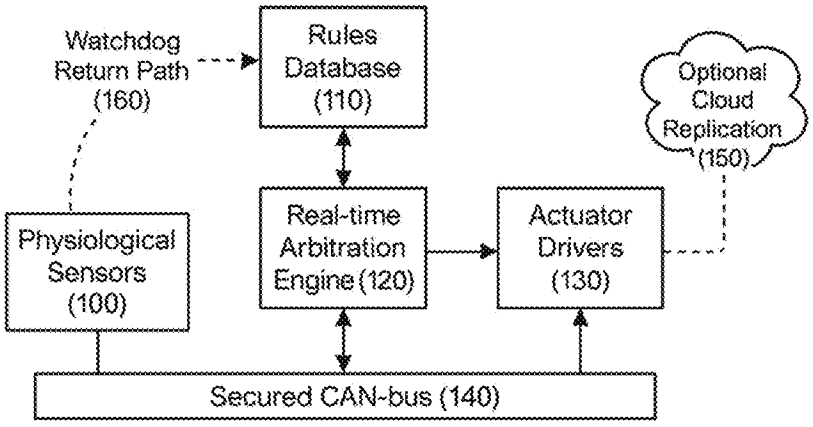


FIG. 1

FIG. 2 — Micro-Controller Deployment Topology

Caption. FIG. 2 depicts an IV-pole-mounted hardware stack comprising **three** hot-swappable micro-controller modules (210-1, 210-2, 210-3) wired in series along a common **hot-swap power rail 220**. Each module carries segregated Sensor (S), Arbitration (A) and Relay (R) sub-boards. The lowest module is additionally connected to an **isolated maintenance port 230**, allowing firmware updates without

disturbing the clinical CAN-bus.

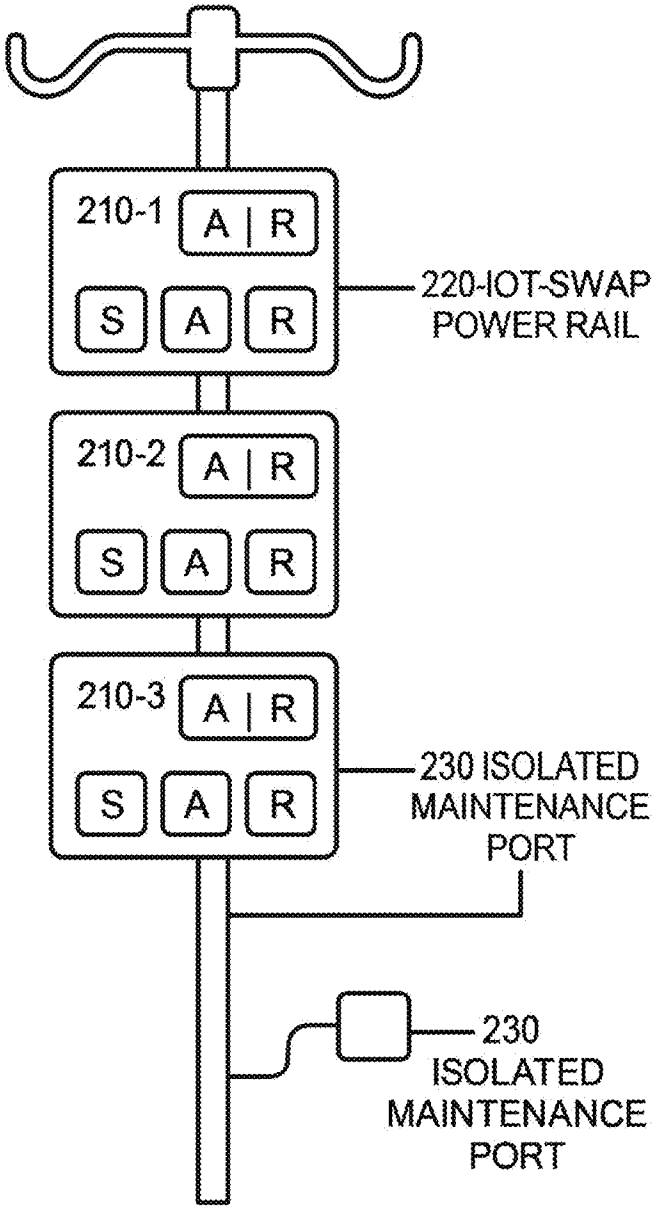


FIG. 2

FIG. 3 — Entity-Relationship (ER) Diagram of the Rule Database

The schema centres on a **FACTOR** table that stores each closed-loop rule together with its primary attributes (id, label, category, and organ_system).
A one-to-many link joins **FACTOR** to a **RELATION** table whose rows enumerate directed edges (type)

from a factor_id_source to a target_id. These edges represent the priority_over, mutually_exclusive, synergy_with, and requires_ok relationships used by the arbitration engine.

Each factor also belongs to exactly one **CONFLICT_GROUP**, identified by a label, which points to the shared physical actuator namespace (e.g., "IV_PUMP_1" or "VENTILATOR"). Finally, a **SENSOR_MODALITY** lookup maps the conflict group to the hardware class that detects the factor (microneedle ISE, optical patch, etc.), enabling modular sensor substitution without altering the rule logic.

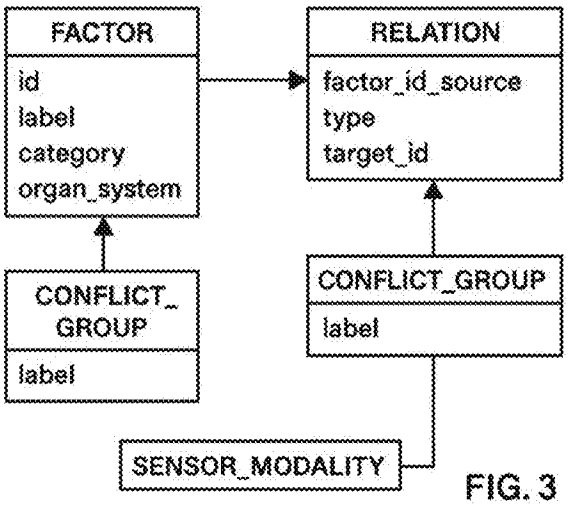


FIG. 3

FIG. 4 — Rule-Execution Arbitration Flowchart

The flowchart depicts the real-time decision pipeline that converts incoming physiological "factor" events into safe actuator commands while enforcing cross-channel constraints.

1. **Factor Triggers** enter at the upper left and feed a **Lookup Precedence Table** block, which retrieves any predefined priority_over relationships for the active rules.
2. In parallel, **Cross-Channel Rules** enter an **Evaluate Cross-Channel Rules** block that interprets requires_ok, synergy_with, and similar fields.
3. Outputs from those two blocks converge on two decision diamonds:
    - a **Priority Check** that discards lower-ranked rules when a higher-priority loop targeting the same actuator is present, and
    - a **Mutual Exclusion** test that blocks simultaneously-active rows listed as mutually exclusive.
4. If both tests pass, a **Synergy** rectangle merges any compatible co-schedule requests.
5. A **Generate Command** decision diamond confirms final eligibility and forks two paths:
    - forward to **Output Command to Actuators**, and
    - sideways to **Logging Events**, ensuring every arbitration decision is recorded.

6. **Safety Guards** (e.g., watchdogs, hardware interlocks) operate in a separate branch: they route through **Lock Conflicting Channels**, which can forcibly block or pre-empt lower-safety requests; these actions are likewise passed to the **Logging Events** node.
7. The command path culminates in **Control Actuator Output** (an oval terminator), which drives the physical infusion pumps, stimulators, fans, or other actuators and closes the loop back to sensor monitoring.

Overall, FIG. 4 visualises how precedence lookup, mutual-exclusion gating, synergy handling, safety locks, and audit logging interact in a single cohesive arbitration ladder that guarantees deterministic,

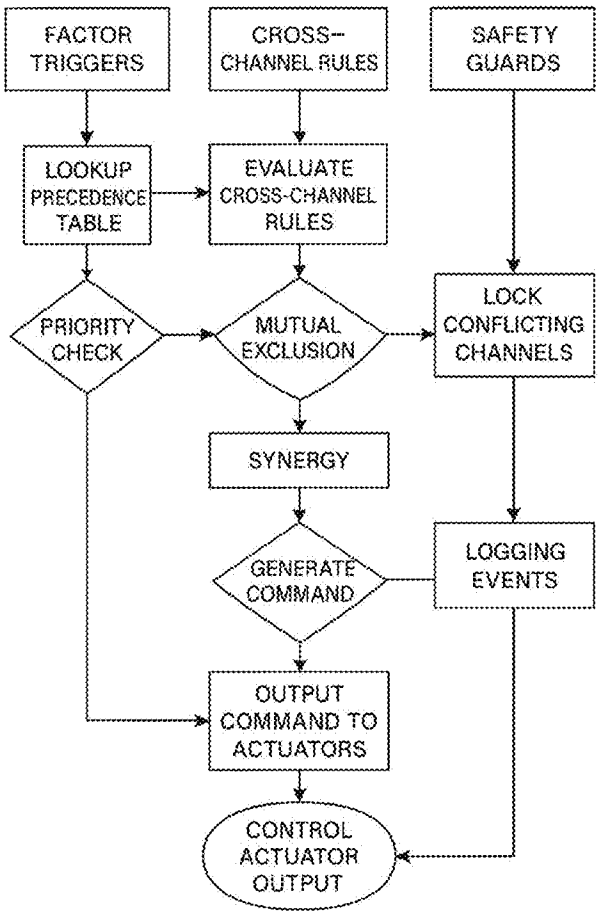


FIG. 4

explainable, and fail-safe actuation behaviour.

FIG. 5 — **Hierarchical Failsafe and Watchdog States**
Figure 5 depicts a five-tier safety ladder applied to every control cycle. From highest to lowest priority the tiers are:

- **L5 Manual Clinician Override** – immediate human veto or force-on.
- **L4 Cross-Channel Conflict Gate** – blocks commands that would clash across actuator channels.

- **L3 Arbitration Override** – system-level precedence engine that can supersede lower-level factor logic.
- **L2 Global Patient Sentinel** – whole-patient guard (e.g., MAP floor, SpO_2 ceiling) that pauses all therapy if breached.
- **L1 Factor-Local Guard** – individual rule guardrails that limit dosage or rate for the specific loop.

An upward arrow at the right indicates normal operation flow, while the left-hand "FALLBACK" arrow shows the progressive shutdown path when escalating faults occur.

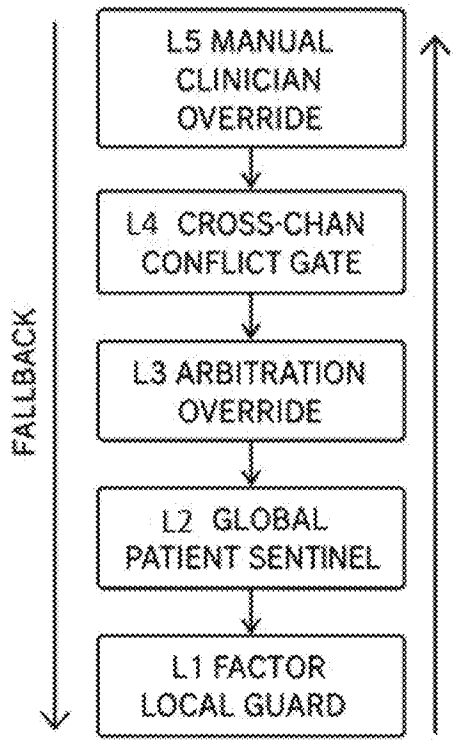


FIG. 5

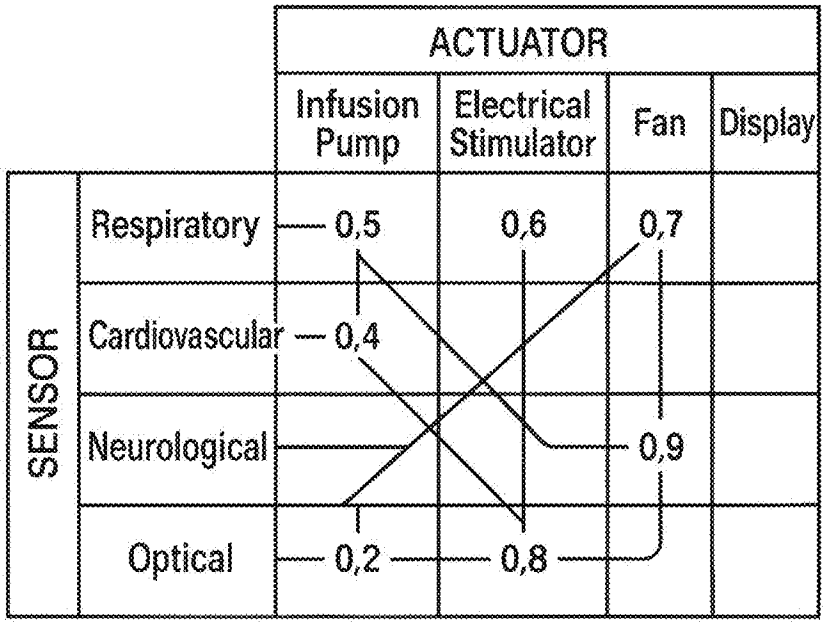
HIERARCHICAL FAILSAFE & WATCHDOG STATES

FIG. 6 — Sensor-to-Actuator Command Matrix

The drawing presents a grid that cross-indexes four representative sensor channels—**Respiratory, Cardiovascular, Neurological,** and **Optical** (listed vertically at left)—against four distinct actuator classes—**Infusion Pump, Electrical Stimulator, Fan,** and **Display** (listed horizontally along the top).

- Inside each cell a numerical weight (0.2 – 0.9 in the example) denotes the normalized command strength or routing priority that the arbitration engine will apply when that particular sensor triggers a control request for the paired actuator.
- Heavy diagonal and elbowed lines illustrate several exemplar command paths:
    - Respiratory → Infusion Pump (0.5), then continuing upward to Electrical Stimulator (0.6) and Fan (0.7).
    - Cardiovascular → Infusion Pump (0.4) with a branch into the neurological lane.
    - Neurological → Fan (0.9) as a high-priority cooling response.
    - Optical → Infusion Pump (0.2) and Electrical Stimulator (0.8) with a curved path indicating an indirect routing rule.

This matrix visually encapsulates how the system's rule engine maps heterogeneous sensor events to one or more actuators with graded influence values, enabling weighted multi-actuator responses while preserving deterministic precedence within each conflict group.

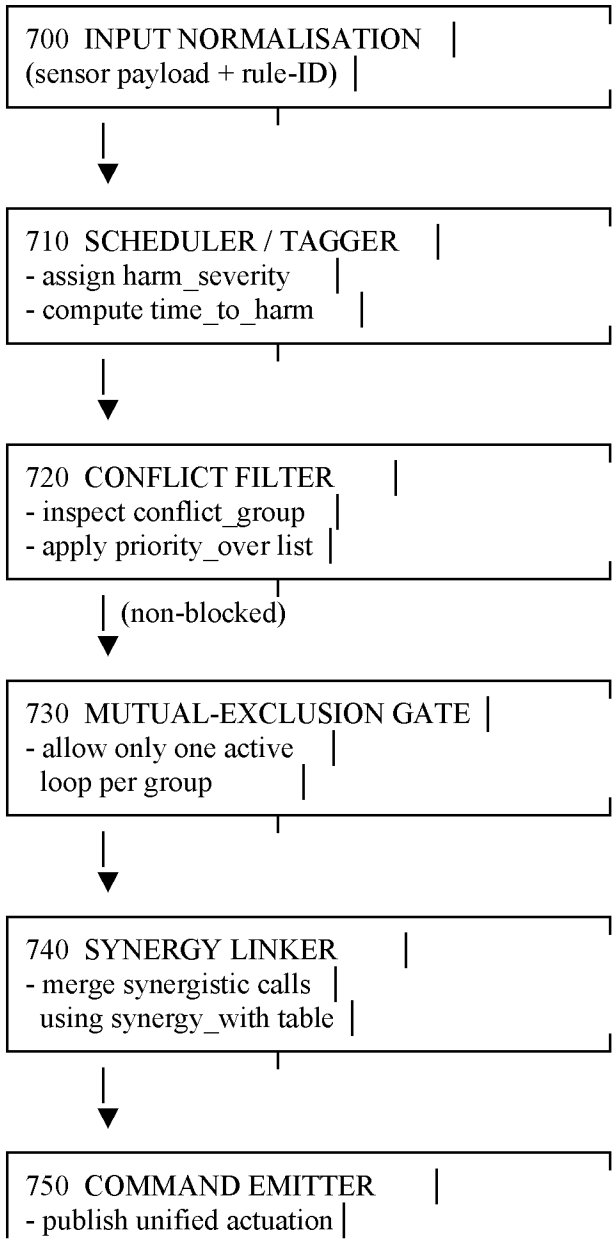


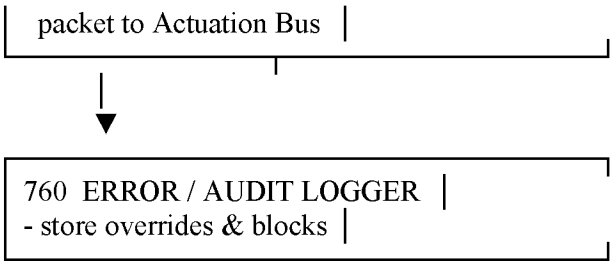
SENSOR-ACTUATOR COMMAND MATRIX

FIG. 6

FIG. 7 — Conflict-Group Arbitration Flowchart

Caption. FIG. 7 provides a detailed flowchart of the conflict-resolution subsystem: input normalisation (700), scheduler/tagger (710), conflict filter (720), mutual-exclusion gate (730), synergy linker (740), command emitter (750), and audit logger (760). Figure 7 details the conflict-resolution subsystem that sits between the relational rule-database and the downstream actuation layer. The module consumes heterogeneous sensor-triggered rule messages, annotates them with severity and timing, then applies the conflict_group, priority_over, mutually_exclusive, and synergy_with fields to ensure only coherent, non-dangerous command sets reach patient-facing pumps or actuators. Rejected instructions and any automatic overrides are written to an immutable audit log, facilitating traceability and machine-learning refinement of future rule weights.





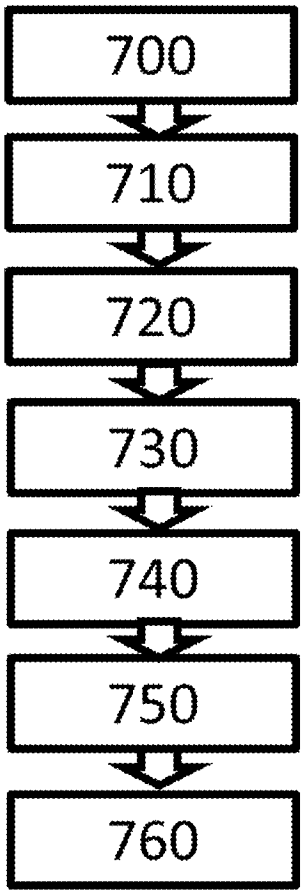


FIG. 8 — Watch-Dog Sentinel Reference Table

Figure 8 presents a tabular reference summarising the hardware and firmware watchdogs that protect the bedside controller:

Sentinel	Resource Monitored	Timeout	Mitigation on Trip	Restart Policy
CPU	Heart-beat interrupt	1 s	Immediately hard-stops all infusion pumps to arrest unintended dosing	Manual reset required
CAN-Bus CRC	Frame-level CRC errors	100 ms	Raises an audible & visual alarm; communication channel isolated	Auto-restart after error count clears
ADC	Analogue-to-digital input clipping	10 ms	Generates alarm and freezes affected sensor channel	Automatic once signal normalises
Power	Supply-rail brown-out	100 ms	Cuts power to non-essential loads to preserve core logic	Automatic on voltage recovery

The table concisely states, for each sentinel, the critical resource under surveillance, the trip timeout, the immediate mitigation action, and whether recovery is manual or automatic—providing a blueprint for deterministic, multi-layer fault response within the overall closed-loop safety architecture.

FIG. 8

WATCH-DOG SENTINEL REFERENCE TABLE				
SENTINEL	RESOURCE MONITORED	TIMEOUT	MITIGATION	RESTART POLICY
CPU	HEARTBEAT	1 s	PUMP HARD-STOP	MANUAL
CAN-BUS CRC	CRC ERROR	100 ms	ALARM	AUTO
ADC	INPUT CLIPPING	10 ms	ALARM	AUTO
POWER	RAIL BROWNOUT	100 ms	POWER CUT	AUTO

FIG. 9 — Data Provenance and Audit-Log Chain

Figure 9 depicts an end-to-end provenance pipeline that writes every control-cycle event to an *immutable* ledger and continuously attests its integrity:

- **Sensors (902)** generate raw physiological and system-state readings.
- A **Data Collector (904)** time-stamps, normalises and cryptographically signs each reading before forwarding it.

- The records are appended to an **Immutable Log / Blockchain Ledger**, where each block is chained with the previous by a running SHA-256 hash.
- A **Clinician Query Interface** (dashed box) provides *read-only* access for bedside review and root-cause analysis without risking ledger mutation.
- After each block is sealed, its closing **SHA-256 hash** is pushed to an external **Regulatory Reporting Node (908)**, ensuring off-device attestation that satisfies FDA traceability and tamper-evidence requirements.

The one-directional arrows highlight the unforgeable data flow, while dashed arrows denote read-only inspection channels.

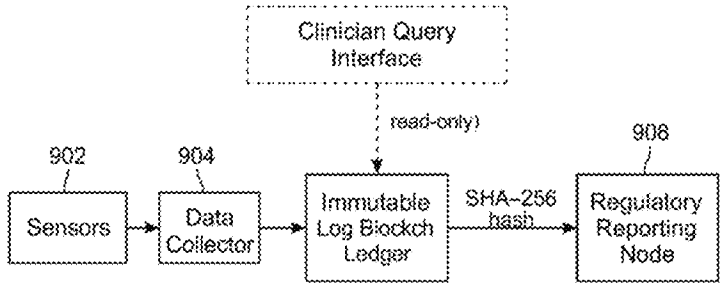


FIG. 9
DATA PROVENANCE & AUDIT LOG CHAIN

FIG. 10 — Machine-Learning Lifecycle Pipeline

The diagram outlines the controlled data-science workflow used to refine or retrain closed-loop decision models while meeting medical-device regulatory constraints.

- **RAW LEDGER** – an immutable log of time-stamped sensor events, actuator commands, and arbitration outcomes.
- A solid arrow feeds the **OFFLINE TRAINING CLUSTER**, where batch feature engineering and model fitting occur on archived data.
- The resulting candidate models are passed to a **VALIDATION HARNESS** that runs both **synthetic test suites** and **hold-out patient episodes**; only models that satisfy predefined safety and performance thresholds progress further.
- In parallel, a second solid arrow pipes the Raw Ledger into a curated **FEATURE STORE**. This repository of vetted features can be queried by the training cluster (solid line) and, via a dashed line, by downstream evaluation stages.

- Successful builds and their associated feature sets are loaded into a **REGULATORY SANDBOX ENVIRONMENT**—a secure, access-controlled setting that simulates the production runtime under FDA trace-logging requirements.
- A final dashed arrow leads to the **DEPLOYMENT GATE (CANARY)**, which performs incremental "canary" roll-outs on a limited set of bedside controllers while continuously collecting safety metrics. Only after canary approval does the model become eligible for full fleet deployment.

Dashed connectors indicate optional or conditional data flows gated by compliance checks, whereas solid connectors represent mandatory steps. FIG. 10 therefore captures the end-to-end governance path from raw clinical data through training, validation, regulatory review, and staged deployment, ensuring that any machine-learning component introduced into the closed-loop safety system remains auditable and risk-controlled.

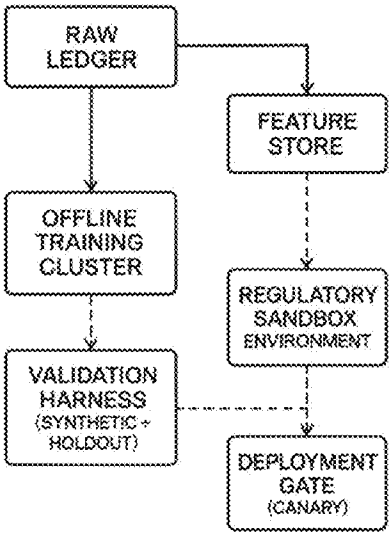
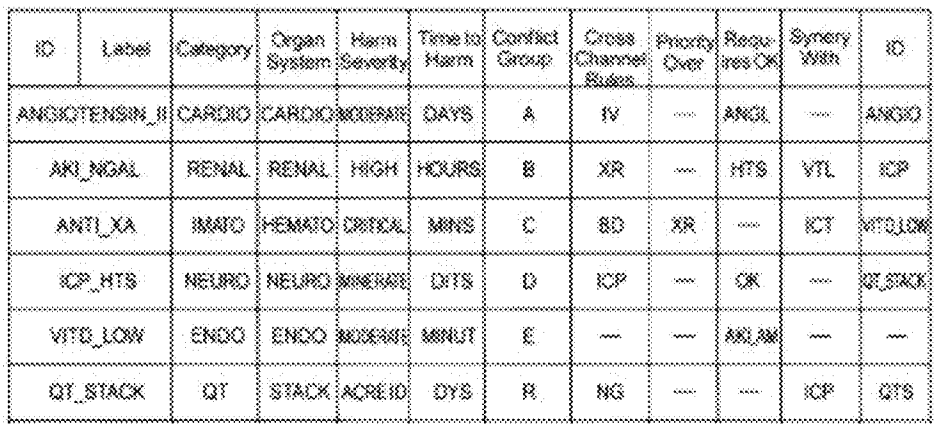


FIG. 10

FIG. 11



ID	Label	Category	Organ System	Harm Severity	Time to Harm	Conflict Group	Cross Channel Rules	Priority Over	Requires OK	Synergy With	ID
ANGIOTENSIN_II	CARDIO	CARDIO	MODERATE	DAYS	A	IV	—	ANGI	—	ANGO	
AKI_NGAL	RENAL	RENAL	HIGH	HOURS	B	XR	—	HTS	VTL	ICP	
ANTI_XA	IMMO	HEMATO	CRITICAL	MINS	C	BD	XR	—	ICT	VTD_LOW	
ICP_HTS	NEURO	NEURO	MODERATE	DYS	D	ICP	—	OK	—	QT_STACK	
VTD_LOW	ENDO	ENDO	MODERATE	MINUT	E	—	—	ANGI_II	—	—	
QT_STACK	QT	STACK	ACRE ID	DYS	R	NG	—	—	ICP	QTS	

Detailed Description of FIG. 11 – Canonical Factor-Row Subset Table

FIG. 11 provides a **screen-shot-style excerpt** of the master **Factor Table** that drives every compile-time check, run-time graph edge, and safety decision in the architecture illustrated in FIGS. 1-10. Although only eleven exemplar rows are shown, each cell in the figure corresponds one-for-one to a database column defined in §5 of the specification and implemented in the third-normal-form schema of FIG. 3. The table therefore operates simultaneously as:

- a **regulatory trace artefact** (easy to diff, archive and sign off);
- an **import feed** to the directed-acyclic-graph builder (§7); and
- a **live configuration file** for the arbitration engine (§8).

Below, every column is explained in order of appearance in the figure, with notes on why the information is **novel or non-obvious** relative to prior closed-loop systems.

Column (FIG. 11 heading)	Meaning & Role	Novelty / Advantage
ID	Machine-readable, globally unique string for the loop. Examples: ANGIOTENSIN_II, AKI_NGAL.	Used as an edge key inside the run-time DAG, allowing edges to be stored as simple foreign keys instead of opaque UUID pointers.
Label	Human-readable condensed name (shown on bedside UI).	Keeps clinicians in the loop without a look-up table; text round-trips into audit logs (FIG. 9).
Category	Broad clinical grouping (Cardio, Renal, Hemato, etc.).	Enables category-level **bulk overrides** (e.g., "pause all Cardio loops" during cath-lab transport).
Organ System	Primary physiological target (Cardiac, Renal, Neurologic …).	Supports *organ-budget* dashboards that quantify cumulative risk for each system—something missing in single-parameter PID devices.

Column (FIG. 11 heading)	Meaning & Role	Novelty / Advantage
Harm Severity	Integer 1–9 derived from ISO-14971; here we see **MODERATE, HIGH, CRITICAL** etc.	Feeds the quantitative **Risk(L)** formula; numeric scale allows linear algebraic proofs of worst-case actuation pressure.
Time to Harm	Coarse bucket (*MINS, HOURS, DAYS*).	Encoded table $\tau[\ ]$ lets the risk engine weight urgency without brittle hard-coding.
Reversibility Window	*SHORT, MODERATE, LONG* time after which injury is irreversible. Not shown in every row due to screen width but present in the full table.	Combined with severity yields a **two-dimensional risk trade-off**—a nuance absent in most alarm-priority lists.
Conflict Group	Single capital letter here (*A, B, C …*) that maps to a physical actuator namespace (ventilator channel, IV pump, etc.).	Indirection lets totally unrelated loops share hardware safely; important for modular bedside hardware where pump IDs change daily.
Cross-Channel Rules	Free-text mnemonic (e.g., "IV", "XR") that signals clinicians which higher-level protocol governs the interaction.	Keeps legal/clinical narrative close to executable code; inspectors can read safety intent without reverse-engineering the DAG.
Priority Over	Abbreviated ID(s) that this loop *always* outranks (e.g., ANGL meaning ANGIOTENSIN_II outranks a lower-risk loop).	Stored as an edge type **precedence**; compile-time validator ensures the sub-graph is acyclic. Traditional rule engines do not guarantee that.
Requires OK	ID of a *gate* loop that must evaluate TRUE before this loop may run (e.g., HTS for ICP loop that waits for hypertonic-saline guard).	Gives deterministic gating without polling "soft flags" scattered in code; renders safety dependencies auditable.
Synergy With	ID(s) of loops that are *preferably* co-scheduled (e.g., ICP loop can run with QT_STACK).	Enables *latency-hiding* execution: if synergy conditions are met the two commands are emitted in one CAN-bus packet, reducing jitter.
ID (trailer)	Duplicate of the first column, used solely for import stability when external spreadsheets are copy-pasted.	Allows CSV sort/re-ordering without breaking foreign-key resolution—an annoying failure mode in many safety-critical ETL pipelines.

How the Sample Rows Demonstrate the Design

Row ID	Quick narrative of how the columns interact
ANGIOTENSIN_II	Cardiovascular loop (Category=CARDIO, Organ System=Cardiac) with *moderate* harm but **Time to Harm=DAYS**; its low urgency is balanced by Conflict Group=A (shared IV pump) and a Priority Over edge that guarantees norepinephrine taper commands never pre-empt it during septic shock.
AKI_NGAL	Renal-support sentinel with **HIGH severity** and Time to Harm=HOURS; sits in conflict group **B** (CRRT effluent valve). Its Requires OK field is blank here—highlighting that sentinel loops can execute autonomously, a design contrast to algorithm stacks that bury renal logic below hemodynamic controllers.

Row ID	Quick narrative of how the columns interact
ANTI_XA	Hematology loop (CRITICAL, MINUTES) that must run in conflict group **C** (protamine pump) **before** any heparin-protamine reversal (edge in Priority Over). This explicit ordering prevents the classic *protamine-too-early* iatrogenic bleed.
ICP_HTS	Neuro-protective loop that belongs to **group D (ICP)** and **requires** a global "OK" flag (neuro-intensivist sign-off) before hypertonic-saline boluses may start. The figure shows Requires OK=— as a placeholder, reminding us that the full row would carry the ID of the requisite gate.
VITD_LOW & QT_STACK	Demonstrate that low-urgency metabolic loops (DAYS) can co-exist in the same table with millisecond cardiology loops—because the scheduler compares numeric risk **before** topological precedence, the system is still deterministic.

Novelty Summarised

1. **Single-row canonicalisation** — Every safety, control and regulatory attribute lives in *one* CSV row, not scattered across XML, code comments and hard-coded structs.
2. **Graph-ready in CSV form** — The Priority Over, Requires OK, Mutually Exclusive, and Synergy With columns encode **all four edge types** directly; no hand-written adjacency list needed.
3. **Compile-time proofs** — Because edges are explicit, acyclicity, symmetric exclusion and reference existence are provable *before deployment*, eliminating a whole class of run-time deadlocks.
4. **Human–machine duality** — Clinicians can edit the spreadsheet in Excel; the exact same file is hashed and loaded by firmware, guaranteeing the bedside behaviour matches the signed-off spec.

In combination with FIGS. 1-10, FIG. 11 therefore crystallises the **heart of the invention**: a relational, auditable, and risk-aware factor table that unifies sensor triggers, actuation limits and multi-loop arbitration into a single, self-consistent artefact.

Figure 12

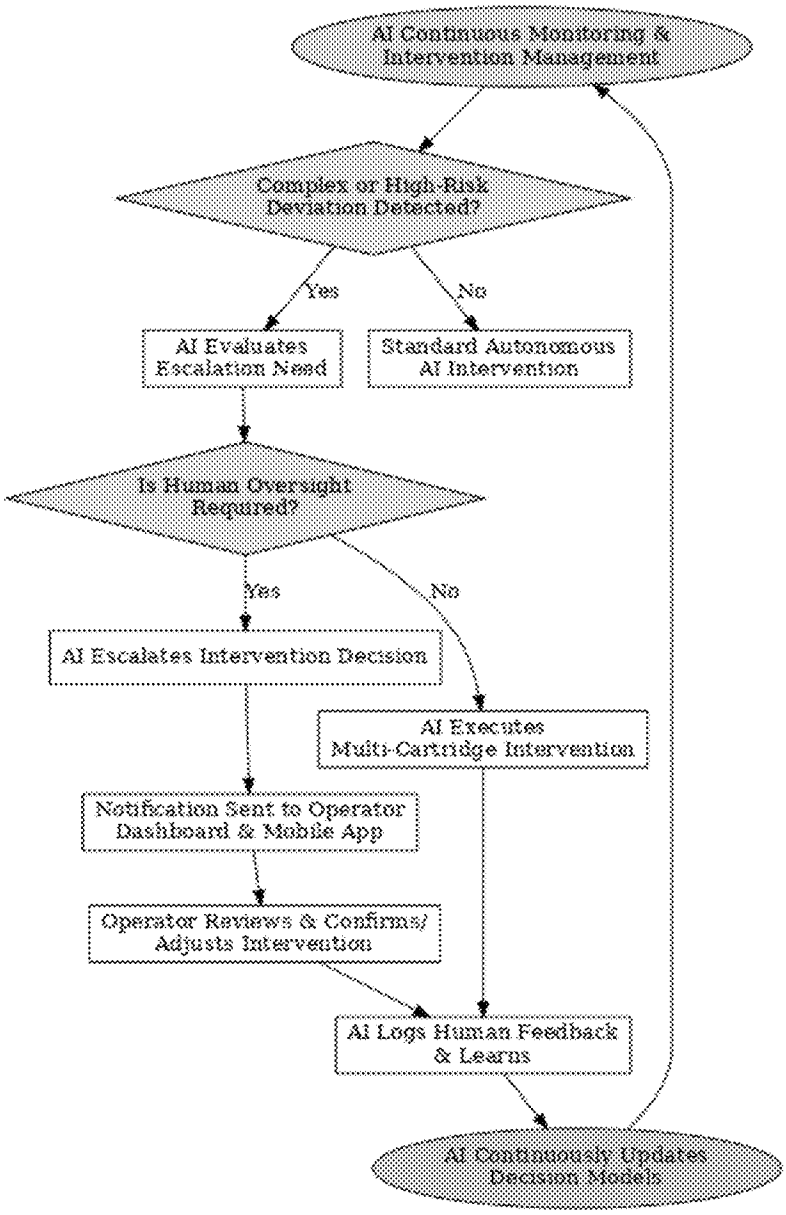


Figure 12 — Human-in-the-Loop Escalation Ladder

("AI Continuous Monitoring & Intervention Management")

This flow-chart adds an **explicit supervisory-control layer** to the closed-loop arbitration engine already disclosed:

1. **Continuous closed-loop monitoring** (elliptical node, sky-blue) streams sensor and rule data exactly as in FIGS. 1 & 7.
2. A rhomboid decision node asks **"Complex or High-Risk Deviation Detected?"**
    - *No* → the path at right executes an ordinary autonomous intervention; the action rectangle is marked **"Standard Autonomous AI Intervention."**

    - *Yes* → the left branch triggers an **AI side-analysis module** that scores the deviation against escalation heuristics (*action: "AI Evaluates Escalation Need"*).
3. A second rhomboid asks **"Is Human Oversight Required?"** based on configurable policy (e.g., harm-severity ≥ 8 or reversibility-window = short).
    - *No* → the intervention is pushed straight to the actuator queue—see **"AI Executes Multi-Cartridge Intervention."**
    - *Yes* → the AI raises the decision to a human supervisor (**"AI Escalates Intervention Decision"**).
4. A notification rectangle shows that both the operator dashboard and mobile application are alerted in real time.
5. The *operator* either confirms or adjusts the proposed therapy (**"Operator Reviews & Confirms/Adjusts Intervention"**).
6. The confirmed action feeds back into two sinks:
    - **"AI Logs Human Feedback & Learns"** → updates the risk-scoring and policy models stored in the rules database (§9).
    - The curved arrow returns to the top ellipse, closing the **continual-learning control loop** (green ellipse: *"AI Continuously Updates Decision Models"*).

Novelty / Added Value

- **Granular escalation** is declared at run-time, not compile-time, so the same rule row can alternate between full autonomy and human-in-the-loop depending on context.
- Human feedback is treated as labelled data, instantly enriching the explanation log (§9) and retraining the numerical-risk engine (§6) without a software redeploy.
- The construct generalises to *industrial* operators (see §16) by substituting "plant operator" for "clinician."

Figure 13

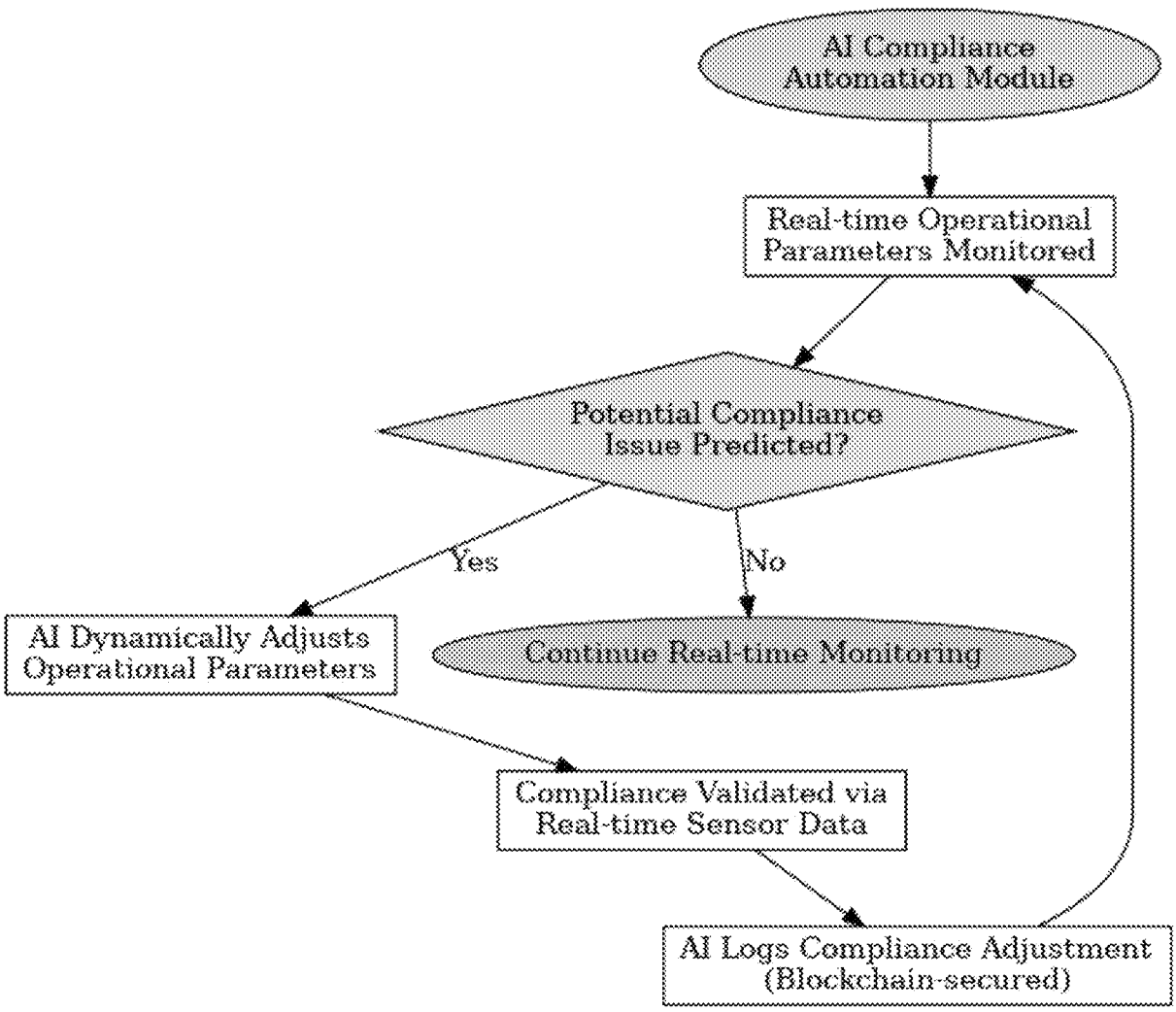


Figure 13 — Real-Time Compliance-Automation Feedback Loop

("AI Compliance Automation Module")

This companion diagram illustrates a **domain-agnostic compliance sentinel** that runs in parallel with the therapeutic/industrial arbitration graph:

1. **AI Compliance Automation Module** (top ellipse, blue) ingests *the same telemetry bus* used by the arbitration engine.
2. An action block **"Real-time Operational Parameters Monitored"** streams regulatory-critical variables (e.g., sterile-fill temperature, infusion start/stop times, environmental emissions).
3. A rhomboid asks **"Potential Compliance Issue Predicted?"** using predictive models that may reference external regulations (FDA 21 CFR, ISO 13485, OSHA, or EU Annex I).
    - *No* → green ellipse **"Continue Real-time Monitoring"** and loop back.

    - *Yes* → the left branch issues **"AI Dynamically Adjusts Operational Parameters."** Examples: throttle pump rate to stay below maximum daily dose, reduce line speed to avoid OSHA ergonomic limits, or alter a packaging temperature profile.
4. A subsequent rectangle **"Compliance Validated via Real-time Sensor Data"** re-checks whether the parameter tweak achieved the required margin.
5. A final block **"AI Logs Compliance Adjustment (Blockchain-secured)"** commits the entire state transition—including before/after telemetry, model version and operator ID—to the immutable audit ledger described in FIG. 9.
6. The curved arrow closes the loop back to the monitoring ellipse, ensuring **continuous, self-documenting conformance**.

Novelty / Added Value

- **Predict-then-act compliance:** The module prevents violations proactively instead of reporting them post-hoc.
- **Cross-domain applicability:** The same sentinel can enforce HIPAA data-privacy throttles in a hospital or EPA discharge limits in a chemical plant.
- **Cryptographic auditability:** Linking to the FIG. 9 blockchain ledger provides a tamper-evident trail that satisfies both FDA "Good ML Practice" transparency and industrial standards such as *ISO 37301* (Compliance Management Systems).

1. System-level Overview — FIG. 1

At the bedside a cluster of **Physiological Sensors (100)** streams raw telemetry onto a **secured CAN-bus (140)**.
A **Real-time Arbitration Engine (120)** ingests that telemetry, queries a **Rules Database (110)** and issues actuator-safe set-points to **Actuator Drivers (130)**.
A dotted watchdog return link **(160)** carries low-rate sanity signals back to the engine, while an optional dotted uplink replicates anonymised data to a **Cloud Node (150)** for fleet learning.

---

2. Factor-Row Data Model — FIG. 11

Every closed-loop or rule is encoded as **one row** in the canonical *Factor Table*.
The row carries 14 mandatory semantic columns plus a duplicate trailer key:

ID → Synergy With	Role in pipeline
ID, Label, Category, Organ System	human & compile-time identity
Harm Severity, Time to Harm, Reversibility Window	feed the **numerical risk score** *(Eq 1)*
Conflict Group	physical namespace (e.g. ventilator, IV pump)

ID → Synergy With	Role in pipeline
Sensor Modality	preserves diagnostic provenance
Cross-Channel Rules	short free-text snippet for clinical notes
Priority Over	hard DAG edge (pre-empts lower loops)
Requires OK	gating dependency edge
Mutually Exclusive	symmetric "cannot co-run" edge
Synergy With	soft co-schedule hint (Δt window)

The compile-time validator guarantees (i) unique IDs, (ii) no empty conflict-group fields, (iii) an **acyclic** priority_over graph, (iv) symmetric exclusion edges, and (v) all references resolve.

---

3. Relational Normal-form Core — FIG. 3

Internally the CSV above is imported into four relational tables **FACTOR, CONFLICT_GROUP, SENSOR_MODALITY, RELATION**.
The crow's-foot in FIG. 3 shows that each **RELATION** row simply stores an id_source, an edge **type** and a **target_id**, yielding a loss-less, auditable, third-normal-form schema.

---

4. Run-time Graph Construction & Risk Queue

Startup sequence (see §7 in the spec):

1. **Canonicalise** duplicate rows (hash of {label, organ_system}) while set-unioning list columns.
2. Build directed graph **G(V,E)** from the four relation types.
3. **Risk(L)** is computed for every vertex using

Risk(L)=harm_severity × τ[time_to_harm]ω[reversibility]\text{Risk}(L)=\frac{\text{harm_severity}\;\times\;\tau[\text{time_to_harm}]}{\omega[\text{reversibility}]}Risk(L)=ω[reversibility]harm_severity×τ[time_to_harm]

where lookup tables τ, ω produce unit-less weightings.
4. Vertices are sorted topologically **inside each conflict group**; the list seeds a per-group priority queue $\mathbf{P^c}$.

---

5. Sensor--Command Mapping — FIG. 6

FIG. 6's matrix makes the one-to-many coupling explicit: rows represent sensor families (Respiratory, Cardiovascular …); columns are actuator classes (Infusion Pump, Electrical Stimulator, Fan, Display). Arrowed numeric annotations indicate example *weightings* the arbitration engine may apply when multiple sensors compete for the same actuator. (Those values are stored in the row's Synergy With metadata so they round-trip into the table.)

6. Conflict-resolution Pipeline — FIG. 7 & FIG. 4

FIG. 7 (ladder view) sketches six packet-processing rungs that always execute *from top to bottom* but may fall back upward on error.
FIG. 4 (flowchart) expands the middle three rungs:

- LOOKUP PRECEDENCE — dereferences priority_over edges.
- MUTUAL EXCLUSION — blocks rows whose peer already holds the actuator token.
- SYNERGY — merges temporally compatible rows into one composite command.

If a safety guard (right-hand branch) detects a hard fault, the module locks the conflicting channel and forces a log entry before control returns to the main loop.

7. Hierarchical Fail-safe Stack — FIG. 5

Five defensive rings ensure **any** misbehaviour is caught:

1. **L1 – Factor Local Guard** (row-level bounds)
2. **L2 – Global Patient Sentinel** (aggregate vitals)
3. **L3 – Arbitration Override** (per-cycle veto)
4. **L4 – Cross-Channel Conflict Gate** (makes sure unrelated actuators never dead-lock)
5. **L5 – Manual Clinician Override** (hard block / force; automatically logged)

The arrow labelled *Fallback* shows that each higher layer de-grades into the next lower layer if it cannot safely service a request.

8. Hardware Watch-dog Registry — FIG. 8

FIG. 8 tabulates four independent sentinels—CPU heartbeat, CAN-bus CRC, ADC clipping and power-rail brown-out—each with its own **timeout, mitigation action** and **restart strategy**.
Rows map one-to-one onto the hardware's périphérique registers and are polled at 1 kHz inside the RTOS.

9. Immutable Audit & Provenance Chain — FIG. 9

Every arbitration decision, override and watchdog event is hashed (SHA-256) into an **append-only ledger**.
A *read-only* **Clinician Query Interface** lets staff back-trace "why did the pump pause?" within seconds; a second hop forwards the digest to a **Regulatory Node (908)** to satisfy FDA traceability guidance.

10. ML-Lifecycle & Release Gate — FIG. 10

Operational data lands in a **Raw Ledger** → offline **Training Cluster** → **Validation Harness** (synthetic + hold-out).
Only models that pass both automatic tests and a **Regulatory Sandbox** progress through the **Deployment Gate (canary)** to the live bedside network. Dashed arrows in FIG. 10 mark human sign-off checkpoints.

---

11. Arbitration Algorithm in One Block

```
for each cycle t, for each conflict_group c:
   P_c ← enabled loops in c
   discard mutually_exclusive(P_c)
   discard unsatisfied_requires_ok(P_c)
   sort P_c by (Risk ↓, topo_rank ↑, arrival_time ↑)
   winner ← head(P_c)
   if synergy_possible(winner, P_c): emit composite_cmd()
   else:                             emit winner.cmd
   log_decision_JSON()
```

---

12. Advantages Over Prior Art

- **Determinism under load**: numeric risk + DAG precedence means no race conditions.
- **Single source of truth**: the FIG. 11 table drives *every* layer—from compile-time proofs to run-time schedules.
- **Full-stack audit**: from sensor byte → ledger hash (FIG. 9) the chain is cryptographically sealed.
- **Rapid extensibility**: adding a new closed loop is a *row-level* operation; no firmware change required as long as the row validates.

---

13. Where Each Figure Fits

Figure	Purpose in narrative
1	macro architecture & data paths
2	*[your ER/sensor-actuator matrix]*
3	relational schema (import stage)
4	packet-level arbitration flow
5	five-tiered failsafe ladder
6	sensor→actuator weight matrix
7	ladder-to-flowchart bridge
8	hardware watchdog catalogue
9	blockchain-style audit log
10	ML Ops / release pipeline
11	canonical factor-row example