ReMix: Resource-Reclaiming Mixed-Criticality Scheduling for Industrial IoT

Md Ashikul Haque and Abusayeed Saifullah
Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA
Email: {mdashikul.haque, abusayeed.saifullah}@utdallas.edu

***Abstract*—Industrial wireless sensor actuator networks (WSANs) increasingly multiplex safety and efficiency loops over a shared multi hop, multi channel infrastructure. In practice, criticality is dynamic: hazards can elevate a small subset of loops to high criticality (HC) at the same time that interference degrades link quality. Existing WSAN schedulers typically protect reliability through static time redundancy, such as duplicate slots or redundant paths. Although effective in nominal conditions, this permanently consumes airtime, inflates the slotframe, and leaves less room to react when the network temporarily enters HC mode. This paper presents REMIX, a mixed criticality WSAN design that supports fast, consistent mode transitions using a two table schedule and epoch latched activation, reallocates airtime in HC mode by reclaiming low criticality (LC) redundancy subject to an explicit LC service budget, and reduces the cost of HC protection through payload augmented transmissions that enable software reconstruction under common shared slot corruption patterns without physical layer changes. We implement REMIX on a 15 node indoor Tmote Sky testbed with a MacBook M4 Pro gateway. The testbed results show that REMIX improves HC deadline satisfaction from 32.9% to 85.9% under the heaviest offered stress point, maintains 63.4% LC delivery during burst stressed HC intervals, and reduces active HC mode slotframe length by 40.9% at 40% HC traffic. At a representative combined stress point, REMIX improves HC deadline satisfaction from 66.1% to 87.5% and reduces experiment derived radio energy per delivered HC packet by 31.9%. Large scale simulations from 500 to 1000 nodes show that the same trend persists beyond the current hardware scale.**

***Index Terms*—Industrial IoT, wireless sensor-actuator networks, mixed criticality, real time scheduling, TSCH/TDMA, reliability, redundancy reclamation.**

I. INTRODUCTION

Industrial wireless sensor actuator networks (WSANs) are becoming a key communication platform for process monitoring and closed-loop control in modern factories, refineries, and utility systems. Existing industrial standards such as WirelessHART, ISA100, and WIA-PA rely on centralized management over a multi-hop mesh and maintain deterministic operation through multi-channel TDMA schedules [1]–[3]. Similarly, IEEE 802.15.4 TSCH and the IETF 6TiSCH architecture support time-synchronized channel hopping and scheduled communication for Industrial IoT deployments [4]–[6]. As industrial plants evolve, operators increasingly consolidate many sensing and control loops over a shared WSAN to reduce deployment cost, simplify maintenance, and improve flexibility.

A key requirement in such WSANs is deterministic communication. End devices must follow a consistent schedule that jointly respects timing, routing, and interference constraints. Unlike best-effort wireless networks, airtime in an industrial WSAN is a tightly managed resource, and even a small schedule change can affect multiple nodes and flows. Thus, resource management in these networks is fundamentally constrained by global schedule consistency, multi-hop interference, and strict actuation deadlines.

Mixed criticality is common in industrial WSANs. Not all traffic has the same importance. Safety and protection loops require high reliability and timely delivery under stressed conditions, while monitoring and efficiency-oriented loops can tolerate temporary degradation. In practice, criticality can also change dynamically. For example, a pressure spike, fault alarm, or abnormal plant condition can suddenly elevate a subset of flows to high criticality. At the same time, interference or transient channel degradation may increase retransmissions and further tighten the schedule.

These conditions make dynamic resource management difficult. A scheduler designed only for the nominal regime may fail when high-criticality (HC) traffic suddenly needs more service. Static protection through permanently reserved redundancy wastes scarce airtime during normal operation. On the other hand, ad hoc schedule changes during a stressed regime can be unsafe because different nodes may apply the changes inconsistently, creating transient collisions and slot mismatches. This challenge becomes more severe in standards-based WSAN stacks, where devices have limited memory and schedule updates must be conveyed using compact control messages.

Consider a refinery WSAN where many monitoring loops coexist with a smaller number of safety loops. During normal operation, low-criticality (LC) flows consume most of the routine reporting capacity, while only limited extra cells are reserved for recovery. When a hazard is detected, HC loops may need both a higher reporting rate and stronger reliability protection. If the network continues to rely on static redundancy, the additional HC demand can quickly exhaust the available schedule. If the controller instead pushes inconsistent or heavy-weight reconfiguration to all nodes, the resulting transition itself may create communication failures. Therefore, a mixed-criticality WSAN must be able to quickly reclaim resources for HC traffic, while preserving a bounded level of service for LC traffic and ensuring that schedule transitions remain safe.

A preliminary workshop paper introduced the mixed-

criticality WSAN setting and the observation that unused capacity can be exploited to reduce reliability overhead [26]. In this paper, we develop that direction into an end-to-end system, called REMIX. The proposed system is based on three key ideas. First, REMIX uses a two-table schedule representation consisting of a stable base schedule and an addressable contingency pool, enabling compact and consistent mode changes. Second, REMIX performs resource reclamation under an explicit LC service budget so that HC flows can receive additional airtime without completely starving LC monitoring traffic. Third, REMIX reduces the amount of time-duplicated protection needed for HC traffic by combining reclaimed cells with payload augmentation and software reconstruction.

In summary, this paper makes the following contributions.

- We formulate mixed-criticality semantics for multi-hop, multi-channel industrial WSANs with explicit mode transitions driven by both plant events and wireless uncertainty.
- We design a compact schedule representation and a mode dissemination mechanism that enable safe and consistent switching using a base schedule and a selectively activated contingency pool.
- We develop a reclamation-aware scheduling framework that reallocates airtime to HC traffic while enforcing an LC service budget, and we integrate a payload-augmented recovery mechanism to reduce reliance on static time duplication.
- We implement and evaluate REMIX on a 15-node indoor WSAN testbed using Tmote Sky devices and a MacBook M4 Pro gateway, and we complement the testbed study with two large-scale simulation results from 500 to 1000 nodes.

The rest of the paper is organized as follows. Section II reviews the related work. Section III presents the system model and problem formulation. Sections IV and V describe the core mechanisms and detailed system design. Section VI summarizes correctness and complexity. Section VII presents the implementation. Section VIII reports the evaluation results. Sections IX and X conclude the paper.

II. RELATED WORK

a) Industrial deterministic wireless and TSCH.: Industrial wireless standards such as WirelessHART, ISA100.11a, and WIA-PA build multi hop meshes with centralized network management and deterministic multi channel TDMA schedules [1]–[3]. In parallel, IEEE 802.15.4 TSCH and the IETF 6TiSCH architecture bring time synchronized channel hopping and track based scheduling to Industrial IoT deployments [4]–[6]. Both families rely on careful schedule dissemination and tight device synchronization, which makes runtime changes challenging.

b) Centralized and distributed scheduling for WSANs.: A large body of work studies routing, channel allocation, and real time scheduling in industrial WSANs under interference constraints [7]–[11]. Distributed and hybrid approaches aim to reduce reliance on a single manager and to react faster to dynamics [12]. REMIX complements these efforts by introducing an explicit mode transition mechanism and a compact schedule representation that can be activated quickly while preserving global consistency.

c) Mixed criticality systems and networks.: Mixed criticality scheduling originated in real time CPU scheduling with multiple assurance levels [13]–[15]. In networks, uncertainty comes from link loss, retransmissions, interference, and plant driven rate changes rather than from worst case execution time variation. A small set of papers introduces mixed criticality abstractions for WirelessHART like networks and analyzes demand and schedulability across criticality levels [16]–[18]. Protocol level support for resilient mixed criticality wireless communication has also been explored [19]. REMIX differs by providing an end to end design that couples consistent mode switching, bandwidth reclamation, and link layer recovery so that HC guarantees can be achieved without permanently enlarging the schedule.

d) Reliability via redundancy, diversity, and time synchronized flooding.: Industrial stacks typically achieve high delivery ratios using retransmissions, duplicate cells, and redundant paths. These techniques are effective but consume schedule capacity, especially in dense meshes where spatial reuse is limited. Alternative communication substrates such as time synchronized flooding and constructive interference provide strong reliability and low latency for certain traffic patterns [20], [21]. REMIX targets scheduled multi hop meshes and improves efficiency in HC mode by reclaiming redundancy from LC traffic and by shifting part of the reliability budget from time duplication to payload redundancy.

e) Collision tolerant decoding and packet recovery.: Collision resolution and capture effect mechanisms have been studied in sensor networks and WiFi [22], [23]. Many approaches rely on PHY layer changes, specialized signal processing, or repeated transmissions. REMIX stays within commodity IEEE 802.15.4 devices by exploiting structured payload redundancy and software CRC validation after standard demodulation, enabling collision tolerant recovery for the HC packet in shared cells without modifying modulation or synchronization.

Taken together, existing research provides strong building blocks for industrial scheduling and for reliability, but it leaves a gap at the mixed criticality boundary: how to reallocate scarce TDMA resources quickly and safely when criticality changes, while keeping the network globally consistent under beacon loss and limiting the collateral damage to low criticality monitoring. REMIX addresses this gap by coupling an epoch latched activation mechanism over a compact schedule pool, reclamation with explicit LC progress guarantees, and collision tolerant payload augmentation within commodity IEEE 802.15.4 stacks.

III. SYSTEM MODEL AND PROBLEM

A. Network and schedule model

We consider an industrial multi hop WSAN with a gateway and a set of field devices. The gateway hosts a centralized network manager that selects routes and computes a multi channel TDMA schedule, as in WirelessHART and related

standards [1]–[3]. Time is divided into slots of length Δ and repeated slotframes of length L slots. Each slot is associated with a channel offset, which maps to an IEEE 802.15.4 channel in the hopping sequence.

A schedule entry is a tuple

$$e = (t, k, u \to v, \eta), \tag{1}$$

where $t \in \{0, \ldots, L-1\}$ is the slot offset, $k \in K$ is the channel, $u \to v$ is a directed link, and $\eta \in \{\mathsf{dedicated}, \mathsf{shared}\}$ indicates whether the receiver expects a single sender or allows contention among a bounded sender set. Devices are half duplex, and a receiver decodes at most one transmission per slot. This model captures the scheduling structure used in industrial managers while exposing the two resources that matter most for our design: conflict free airtime and compact schedule state at the nodes.

B. Interference constraints

We use a conflict relation $\#(\cdot, \cdot)$ to capture half duplex and radio interference constraints. Two entries e and e' can co occur in the same slot only if they do not share an endpoint and do not interfere at the receivers when scheduled on the same channel. This abstraction matches conflict graph based synthesis used in industrial schedulers and captures the fact that spatial reuse across channels and links is limited but valuable. In practice, the gateway can precompute these conflicts offline from topology, route structure, and measured link neighborhoods, so runtime adaptation only selects among already validated cells.

C. Control loops and criticality modes

The WSAN carries a set of periodic control loops $F = \{F_1, \ldots, F_n\}$. For loop F_i, let $(T_i^{\mathrm{LC}}, D_i^{\mathrm{LC}}, R_i^{\mathrm{LC}})$ be the period, relative deadline, and reliability target in LC mode, and let $(T_i^{\mathrm{HC}}, D_i^{\mathrm{HC}}, R_i^{\mathrm{HC}})$ be the corresponding HC contract, where $T_i^{\mathrm{HC}} \le T_i^{\mathrm{LC}}$ and $D_i^{\mathrm{HC}} \le D_i^{\mathrm{LC}}$. Each loop has maximum criticality label $C_i \in \{\mathrm{LC}, \mathrm{HC}\}$.

At runtime, the system criticality mode is determined by plant triggers and by channel health. When the system is in HC mode, all flows with $C_i = \mathrm{HC}$ must satisfy their HC contract under adverse channel conditions. LC flows may receive degraded service, but the system should preserve bounded progress to avoid complete monitoring blackout. This requirement reflects industrial operation more accurately than binary keep or drop semantics, since operators typically still need coarse situational visibility even while the network prioritizes a small set of safety loops.

D. Problem statement

Given topology, routes, channel set K, interference constraints, and loop parameters, our goal is to design (i) a base schedule that satisfies all loops in LC mode and (ii) a mode aware adaptation that, upon entering HC mode, reallocates airtime to HC traffic while bounding the service loss of LC loops, minimizing schedule overhead, and keeping node side complexity small. The key difficulty is that the system must adapt quickly without disseminating a full new schedule or asking devices to reconstruct scheduling state locally during a stressed interval.

IV. Core Mechanisms

This section presents the three technical components of ReMix: a compact schedule representation that can be activated consistently, a reclamation policy that shifts airtime toward HC traffic while preserving minimum LC progress, and a payload augmentation scheme that reduces the dependence on time duplication.

A. Two table schedule representation

ReMix represents the schedule as a stable base table S^{LC} and a set of optional entries P called the contingency pool. Each pool entry is addressable by an index and is one of two types: an extra dedicated link opportunity on an HC route, or a shared receive opportunity that can absorb retransmissions and event bursts. At runtime, the gateway activates a subset of entries by broadcasting a bitmap. The effective schedule in a slotframe is therefore

$$S = S^{\mathrm{LC}} \cup P[\mathsf{bitmap}]. \tag{2}$$

The pool is synthesized for worst case HC demand but stored compactly. For each HC loop, the gateway identifies tight hops that dominate end to end delay and fragile hops with poor recent delivery. It then allocates a small number of candidate extra opportunities on different channels and slot offsets, subject to conflict constraints. A short bitmap therefore replaces the need to disseminate a full second schedule during mode changes. The important design choice is that the index of each pool entry is fixed offline. As a result, a node never has to parse or install a structurally new schedule when the mode changes; it only checks whether the bitmap enables one of the entries it already knows. This keeps beacon processing predictable on low end motes and prevents transient disagreement about cell meaning across neighbors.

B. Offline slack planning and pool construction

The base table is synthesized to satisfy all flows in LC mode with minimal redundancy. We allocate one dedicated opportunity per hop on the primary route for each loop instance and add the minimum number of shared maintenance cells needed for rare retransmissions. Optional path diversity is preserved in the pool rather than being fully activated in steady state.

Each pool entry is a self contained cell descriptor. When inactive, a dedicated pool cell is simply skipped, and an inactive shared pool cell is ignored. This makes activation monotone: turning on additional bits can only add explicitly provisioned opportunities, and turning bits off cannot invalidate the correctness of the base schedule. In practice, the gateway can therefore react quickly by toggling a small set of bits while keeping the device side logic simple. Pool construction is intentionally biased toward cells that are useful under room scale burst stress rather than toward uniform

Algorithm 1 Gateway side activation for one slotframe

Require: Pool entries P, budget B, link statistics, queue statistics
Ensure: Bitmap of active pool entries
Set mode $\leftarrow$ HC if any HC loop reports a miss or any fragile hop loss exceeds threshold
for all pool entries P_j **do**
    Compute score $s(j)$ using Eq. (3)
end for
Sort entries by decreasing score
Select up to B entries subject to conflict constraints
Broadcast $m = (\mathsf{mode}, \mathsf{epoch}, \mathsf{bitmap})$

redundancy everywhere. Dedicated candidates are placed near the dominant HC bottleneck hops but shifted across slot offsets and channels so that one fading condition or one busy receiver does not erase all backup value at once. Shared candidates are concentrated near relay fan in points where short queue buildups are most likely to appear during an HC interval. This selective placement is why a short bitmap can still expose meaningful adaptation range. It also keeps the stored pool compact, since the gateway does not reserve backup cells for links that are unlikely to become limiting under the expected stress patterns.

C. Runtime activation policy

The activation policy selects the subset of pool entries that most effectively repairs the current stress pattern. For each candidate entry P_j, the gateway computes a lightweight score using signals already exposed by industrial stacks: smoothed ACK loss on the hop, normalized relay queueing delay, and recent deadline misses of the corresponding loop. We write

$$s(j) = w_\ell \, \hat{\ell}_{u,v} + w_q \, \hat{q}_v + w_m \, \hat{m}_i, \tag{3}$$

where $\hat{\ell}_{u,v} \in [0,1]$ is a smoothed loss estimate, $\hat{q}_v \in [0,1]$ captures queue pressure at relay v, and $\hat{m}_i \in \{0,1\}$ indicates recent misses for loop F_i. The gateway greedily selects the highest scoring entries subject to conflict constraints and an activation budget B that caps the number of extra opportunities added in one slotframe. The score is recomputed every slotframe from short moving windows, so the controller reacts to sustained degradation but does not oscillate on isolated losses. In addition, the gateway spreads selected entries across HC flows before filling multiple extra cells for the same hop, which avoids starving one fragile route while over protecting another.

D. Payload augmented HC transmissions

Time duplication is expensive when slotframes are already tight. REMIX therefore augments HC packets at the link layer to increase the probability that a single scheduled opportunity suffices for delivery. Let a normal packet be $P = [H \,\|\, D \,\|\, C]$ with header H, payload D, and checksum C. For an HC flow in HC mode, we split D into two equal length chunks D_1 and D_2 and transmit

$$P' = [H \,\|\, D_1 \,\|\, D_2 \,\|\, D_2 \,\|\, D_1 \,\|\, C]. \tag{4}$$

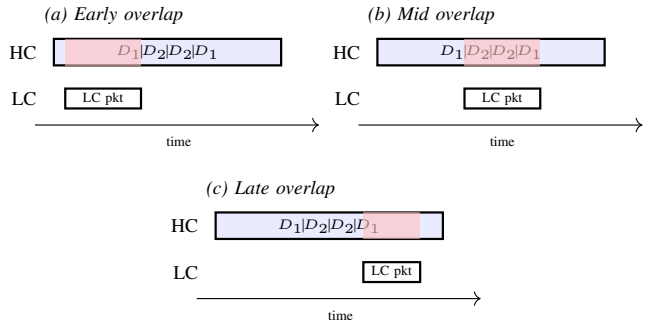


Fig. 1. Collisions in a shared slot between an augmented HC transmission and an LC transmission. Red shading marks the overlapped portion of the HC packet; the receiver recovers by selecting intact copies of payload chunks and validating the software CRC (Algorithm 2).

Algorithm 2 Receiver side reconstruction for an augmented packet

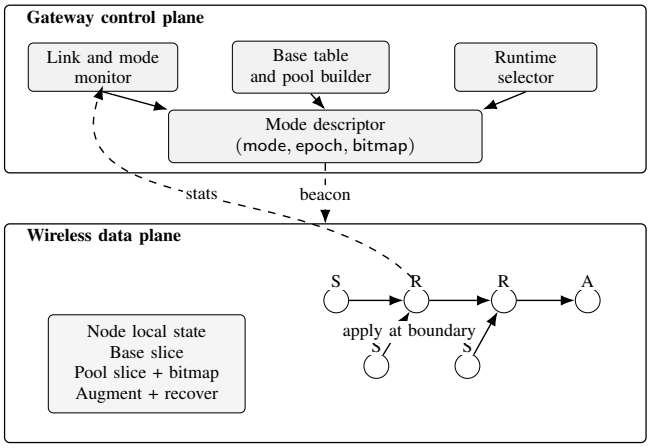
Require: Byte stream for P' (hardware CRC may fail)
Ensure: Recovered payload D or failure
Extract copies $(D_1^{(a)}, D_2^{(a)}, D_2^{(b)}, D_1^{(b)})$ from P'
for all $(x, y) \in \{a, b\} \times \{a, b\}$ **do**
    Build candidate $\hat{D} \leftarrow D_1^{(x)} \,\|\, D_2^{(y)}$
    if software CRC over $H \,\|\, \hat{D}$ matches C **then**
        return $\hat{D}$
    end if
end for
return failure

This preserves the original header and checksum while duplicating only the payload region. Interleaving improves robustness to contiguous corruption because at least one copy of each chunk is likely to remain intact even when an overlap erases a consecutive byte range. In practice, the chunk split is chosen so that the augmented frame still fits within the 127 byte IEEE 802.15.4 limit after accounting for MAC headers and security fields. When that bound would be exceeded, the sender falls back to ordinary HC transmission and relies on schedule level protection from the contingency pool. The mechanism is therefore opportunistic: it is applied exactly where payload structure can save airtime without changing the PHY or violating frame size constraints. This design is particularly attractive in our setting because it shifts recovery effort to software while keeping the radio behavior unchanged.

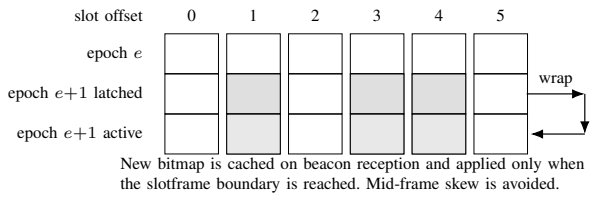
E. Receiver side reconstruction

Commodity IEEE 802.15.4 radios typically expose the received byte stream even when the hardware CRC fails. REMIX exploits this to attempt software recovery: the receiver extracts the two copies of D_1 and D_2, enumerates the candidate payloads obtained by choosing one copy of each chunk, and accepts the one whose software CRC matches C.

Common shared slot collision patterns are illustrated in Fig. 1. Because the HC payload is interleaved and duplicated, at least one copy of each chunk remains intact in several overlap patterns, enabling recovery of the HC packet while the LC packet may be sacrificed. The reconstruction cost is bounded



(a) Control and data plane architecture



(b) Epoch-latched activation timeline

Fig. 2. REMIX uses a stable base schedule plus a compact contingency pool. The gateway broadcasts an epoch-tagged bitmap; nodes latch the update and apply it only at slotframe boundaries, preventing transient schedule skew.

because the receiver checks only four candidate combinations, which is practical on resource constrained motes.

V. SYSTEM DESIGN

A. Architecture and consistent mode switching

Figure 2 shows the end to end design. The gateway runs the control plane: it monitors link health, scores pool entries, selects the mode, and broadcasts a compact mode descriptor that all nodes apply at the next slotframe boundary. Field devices run the data plane: they execute the base schedule, activate the subset of pool entries that involve them, and apply payload augmentation and recovery for HC packets.

The mode descriptor is $m = (\mathsf{mode}, \mathsf{epoch}, \mathsf{bitmap})$, where $\mathsf{mode} \in \{\mathrm{LC}, \mathrm{HC}\}$, epoch is a monotonically increasing identifier, and bitmap activates a subset of the contingency pool. Boundary latching is crucial. Nodes cache the newest descriptor but do not change their schedule in the middle of a slotframe. Instead, they apply the descriptor only when the slot offset wraps to zero. If a node misses a beacon, it continues with its current epoch and catches up when it later receives a newer one. This rule ensures that a node either follows the old configuration for the full slotframe or the new one for the next slotframe, which avoids partial mid frame transitions and the resulting cell skew.

B. Reclaiming airtime with an LC budget

In HC mode, REMIX reallocates airtime to HC flows by reclaiming redundancy originally assigned to LC traffic. Reclamation applies in three places: LC duplicate opportunities are removed first, LC shared slots are thinned next, and LC service is stretched only if additional airtime is still required. To prevent complete LC blackout, we enforce an LC budget implemented as a token rule. Each LC flow must retain at least b_i opportunities in every window of W slotframes. The scheduler may reclaim from LC only when the corresponding budget remains non negative, giving the operator a direct handle on the minimum freshness that must remain visible during stress.

The reclaim order matters. Removing LC duplicates first usually frees airtime with the smallest effect on observability because the primary LC route still remains scheduled. Thinning shared LC slack next reduces headroom but still leaves periodic LC progress. Only the last stage stretches LC service windows, and even that step is bounded by the explicit budget. This ordering matches the experimental behavior in Section VIII: HC timeliness improves sharply while LC traffic remains degraded but not eliminated. Since the budget is checked at the gateway during each activation round, node side logic remains unchanged even when LC opportunities are temporarily reduced.

C. Failure handling and robustness

Mode changes are the most vulnerable time for collisions: if two neighbors use different tables, they may transmit in the same slot. The epoch mechanism ensures that a node either fully remains in the previous mode or fully switches at a boundary. In addition, the gateway can include a short hash of the active bitmap in the beacon; nodes that detect inconsistencies request a resynchronization beacon in a shared maintenance slot. If the contingency pool is still insufficient to satisfy all HC demands, REMIX falls back to a safe policy that temporarily increases HC retransmission pressure on the most critical hops and reduces LC budget within configured bounds.

VI. CORRECTNESS AND COMPLEXITY

a) Conflict free activation.: The gateway constructs the base table and the contingency pool under the same interference and radio constraints. Each pool entry is either a dedicated link transmission placed on an unused cell or a shared cell explicitly marked as shared and paired with a contention rule. Therefore, activating any subset of pool entries preserves the no conflict property for dedicated cells, and any additional interference is confined to shared cells where REMIX already assumes collision tolerant recovery. This separation between prevalidated structure and runtime selection is what makes boundary latched activation safe in practice. The gateway chooses among already legal cells; it does not synthesize fresh cells on the fly when the network is stressed.

b) Consistent mode switching.: Nodes update their active bitmap only after receiving a mode descriptor with a higher epoch than the local epoch, and they apply that descriptor only at the next slotframe boundary. If a beacon is missed, the node continues with its previous bitmap for at most one additional slotframe. Because the base table remains valid in both modes

and activated entries are pre provisioned, transient skew does not create unexpected dedicated cell conflicts.

c) Bounded LC service.: Let b_i be the guaranteed number of LC opportunities for flow F_i over a window of W slotframes. Since reclamation is allowed only when the remaining budget is non negative, each LC flow receives at least its configured lower bound even during sustained HC intervals. The operator therefore trades reclaimable airtime against minimum visibility explicitly rather than implicitly through ad hoc dropping.

d) Computational cost.: Schedule synthesis runs at the gateway and can leverage standard ILP or greedy heuristics used in industrial managers. REMIX adds a second pass that populates the pool with up to K candidates per HC hop and a final feasibility check. Runtime activation in Algorithm 1 is $O(|P| \log |P|)$ due to sorting and runs once per slotframe. Node side cost is limited to schedule lookup, optional payload interleaving, and at most a constant number of software CRC tests per failed HC frame. The node never solves a local scheduling problem and never rebuilds route state when the mode changes; it only toggles preinstalled cells and updates a small epoch value. This matters on Tmote Sky devices because the control action at each boundary remains bounded and deterministic even when several HC flows become active together. The same separation also simplifies validation: offline analysis can check every candidate pool cell once, while online logic only reasons about which already verified cells are enabled. In effect, REMIX moves structural complexity to the gateway and keeps the device side adaptation close to ordinary TSCH table lookup.

VII. IMPLEMENTATION

a) Target platform.: REMIX is implemented on Berkeley/Moteiv Tmote Sky motes [24], which pair an MSP430F1611 class MCU with an integrated IEEE 802.15.4 radio. The firmware running on the motes is built on RIOT OS, an open source operating system for low end IoT devices [25]. Our current evaluation uses 15 such motes in total, including one USB attached sink mote and 14 wireless field motes, with a MacBook M4 Pro as the gateway side controller, logger, and schedule manager.

b) Gateway software.: The gateway runs three logical modules: a schedule builder that constructs the LC base table and the contingency pool, a mode monitor that computes the current criticality mode and scores pool entries from ACK loss, queue pressure, and misses, and a beacon service that disseminates $m = (\mathsf{mode}, \mathsf{epoch}, \mathsf{bitmap})$ while logging per hop outcomes for later analysis. Because the MacBook gateway is not resource constrained, it also stores synchronized traces used to reconstruct packet level outcomes across the testbed.

c) Node side data plane.: Each mote stores (i) its local slice of the base table S^{LC} and (ii) the pool entries that involve the node as sender or receiver. A slot handler runs once per slot and performs channel selection, radio state switching, and optional contention handling for shared cells. For HC packets

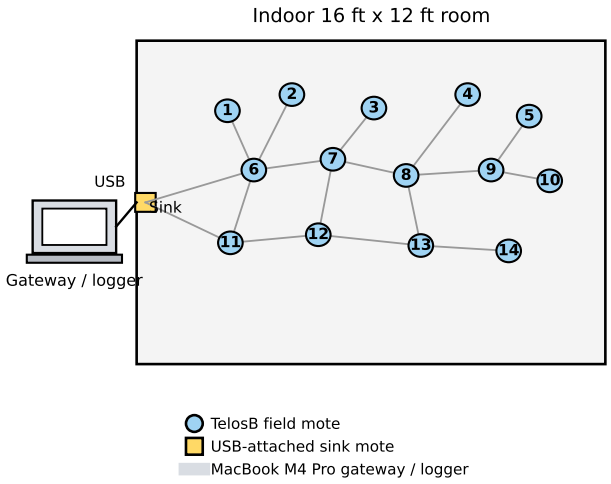


Fig. 3. Indoor 15-node TelosB deployment used in the testbed. A MacBook M4 Pro acts as the gateway side controller and logger through a USB attached sink mote.

in HC mode, the sender applies Eq. (4) by interleaving the payload. On reception, if the hardware CRC fails, the driver forwards the byte stream to the recovery routine, which applies Algorithm 2. If recovery succeeds, the node acknowledges the packet and delivers the recovered payload to upper layers.

d) Control state footprint.: The two table representation is also practical for TelosB class memory limits. Each node keeps only its own outgoing and incoming cells together with the bitmap positions of the contingent entries it may activate, rather than a global network schedule. In our prototype, the beacon carries only the mode bit, epoch, and compact activation bitmap, which lets the gateway update many optional cells at once without sending per link commands.

e) Timing and packetization.: The testbed uses 15 ms slots and a 61 slot slotframe, giving a 915 ms control cycle. LC flows generate one 24 byte application payload every two slotframes, while HC flows generate one 32 byte payload per slotframe in LC mode and two 32 byte payloads per slotframe in HC mode. We use four channel offsets that hop over IEEE 802.15.4 channels 15, 20, 25, and 26. These values were chosen to fit the TelosB timing envelope while still stressing the schedule under concurrent HC activation.

f) Feasibility notes.: IEEE 802.15.4 caps the PHY frame at 127 bytes, so augmentation is enabled only when the resulting frame fits within the standard bound. Larger packets fall back to time duplication on selected hops. The reconstruction routine scans at most two payload copies and performs only a constant number of CRC computations, so the added processing overhead remains small compared with slot duration.

g) Deployment.: The experiment is conducted indoors in a 16 by 12 foot room. The nodes are placed to create short and medium multi hop paths between the field devices and the sink, with several links sharing receivers and channel offsets so that the schedule experiences the same contention pressure that motivates mixed criticality adaptation. The room scale is intentionally small enough for repeatability while still

TABLE I
INDOOR TESTBED CONFIGURATION USED FOR THE MAIN EVALUATION.

Item	Setting
Nodes	15 Tmote Sky motes (1 sink + 14 field motes)
Gateway / logger	MacBook M4 Pro
Room size	16 ft × 12 ft indoor room
Slot timing	15 ms slot, 61 slot frame (915 ms)
Channel plan	4 offsets over ch. 15, 20, 25, 26
Payload sizes	24 B LC, 32 B HC
Traffic rates	LC: 1 pkt / 2 frames; HC: 1 pkt / frame in LC mode, 2 pkt / frame in HC mode
Active mote sweep	6 to 15 active motes
HC traffic sweep	10% to 70%
Path depth in traces	2 to 5 hops
Repeated runs per point	90 to 180

exposing burst loss and queue buildup during stressed HC intervals.

VIII. EVALUATION

Our main evaluation uses a 15 node indoor testbed. Only the last two figures are simulation results used to study larger scales.

A. Indoor testbed and compared systems

We deploy 15 Berkeley/Moteiv Tmote Sky motes [24] in a 16 by 12 foot room and use a MacBook M4 Pro as the gateway side controller and serial logger. One sink mote is attached to the laptop over USB, and the remaining 14 motes form the multi hop data plane. We compare five systems. **StaticDup** reserves a fixed duplicate dedicated slot for HC hops. **RedPath** adds a backup opportunity on an alternate route when available. **DropLC** suspends LC service in HC mode and reallocates all reclaimed airtime to HC traffic. **ReMixNoAug** uses REMIX scheduling and reclamation but disables payload augmentation. **ReMix** enables the full design.

Each point averages repeated runs of the same placement and traffic script while varying the number of active motes, the HC fraction, or the activation budget. For each run, the gateway records the transmitted epoch bitmap, per slot send and receive outcomes, software recovery outcomes after CRC failure, and the queue backlog at HC sources. We align these logs offline to derive deadline satisfaction, delivery, slotframe occupancy, and radio on time. The energy metric is derived from measured radio on time in the traces together with the transmit and receive current values reported for the Tmote Sky / TelosB platform data sheets at 3 V [24]; it is not from an external current probe.

B. HC deadline satisfaction as the number of active motes rises

Figure 4 increases stress by raising the number of simultaneously active motes from 6 to 15. REMIX stays above 82% even at the highest stress point, whereas StaticDup falls to 25.8% because duplicated dedicated cells inflate the slotframe and reduce flexibility. At 15 motes, REMIX achieves 85.9% HC deadline satisfaction, compared with 32.9% for StaticDup and 48.7% for RedPath. ReMixNoAug shows that schedule adaptation already helps, while payload augmentation provides the additional gain in the heaviest regime.

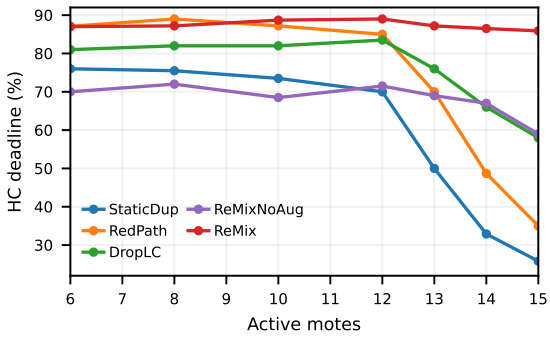


Fig. 4. Experiment: HC deadline satisfaction as the number of active motes rises.

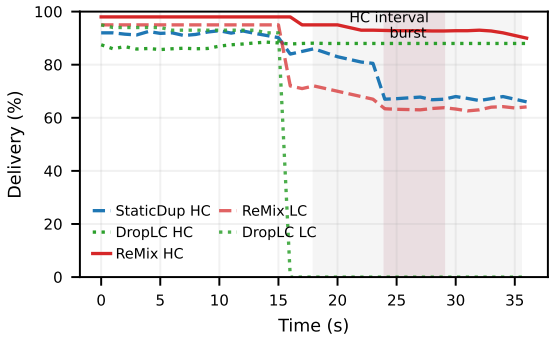


Fig. 5. Experiment trace with an HC interval and burst interference.

C. Mode switch dynamics and LC service in the indoor room

Figure 5 shows delivery over a trace with an HC interval and a shorter burst interference episode inside it. During the burst window, REMIX keeps mean HC delivery at 92.9%, higher than StaticDup at 67.0% and DropLC at 88.0%. At the same time, REMIX maintains 63.4% LC delivery, whereas DropLC drives LC traffic to zero. Thus, REMIX improves safety traffic without sacrificing plant visibility.

D. Schedule overhead and experiment derived energy

Figure 6 reports how the active HC slotframe grows as the HC traffic fraction rises in the 15 node deployment. StaticDup and RedPath grow almost linearly because extra reliability is paid mainly in dedicated cells. REMIX grows more slowly because part of the HC protection is shifted into selective activation and payload augmentation. At 40% HC traffic, REMIX uses 35.3 active slots per frame versus 59.7 for StaticDup, a 40.9% reduction.

Figure 7 reports experiment derived radio energy per delivered HC packet. REMIX requires 2.22 mJ per delivered HC packet, compared with 3.26 mJ for StaticDup, a 31.9% reduction. Although augmentation increases payload length, it reduces duplicated opportunities and retransmissions in the traces.

E. Sensitivity to activation budget on the hardware deployment

Figure 8 varies the activation budget B on the testbed controller. At $B = 0$, no pool entries are activated and

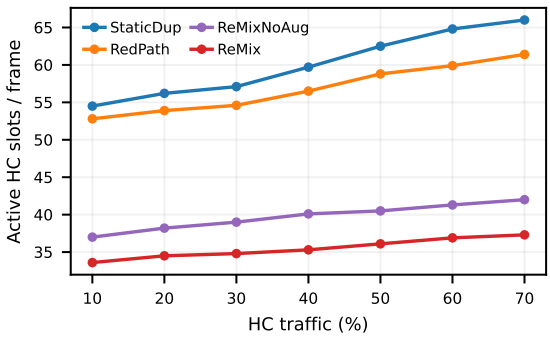


Fig. 6. Experiment: active HC slotframe length versus HC traffic fraction.

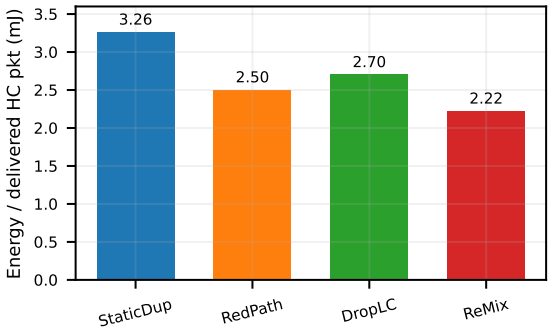


Fig. 7. Experiment: radio energy per delivered HC packet.

HC satisfaction starts low. Increasing B raises HC deadline satisfaction quickly at first, then saturates near 88.7% around $B = 10$. LC delivery decreases only mildly, from 66.8% at $B = 0$ to 62.9% at $B = 16$, because the LC budget bounds reclamation. In practice, B between 6 and 10 captures most of the HC benefit while preserving LC visibility.

F. Ablation of payload augmentation under burst severity

Figure 9 isolates the benefit of payload augmentation by increasing burst severity. As loss pressure rises, ReMixNoAug degrades steadily because it relies only on schedule level protection. Under the strongest burst setting, REMIX retains about 84% HC reliability, compared with roughly 75% for ReMixNoAug and 61% for StaticDup.

G. Representative experimental summary

Table II summarizes the representative combined stress point on the 15 node deployment. REMIX is the only system that simultaneously achieves the best HC deadline satisfaction, the best HC reliability, non zero LC service, the shortest HC slotframe, and the lowest energy.

H. Large-scale simulations

Figure 10 expands the network size from 500 to 1000 nodes and shows that REMIX maintains a clear HC deadline advantage over StaticDup, RedPath, and ReMixNoAug as the graph becomes denser. Figure 11 shows the corresponding slotframe growth. The gap between REMIX and StaticDup widens with scale because StaticDup reserves a full duplicate

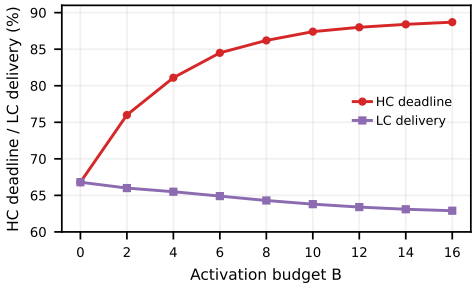


Fig. 8. Experiment: sensitivity of REMIX to the activation budget B.

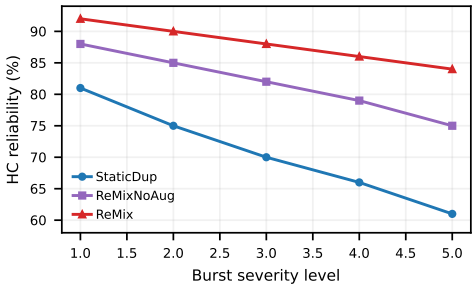


Fig. 9. Experiment: HC reliability versus burst severity, isolating the contribution of payload augmentation.

dedicated cell for each additional HC hop, whereas REMIX adds protection selectively through the pool.

IX. DISCUSSION

a) Interoperability with industrial stacks.: REMIX is designed to fit within centrally managed TDMA systems. The two table representation can be realized as a base schedule plus a small number of optional cells in a WirelessHART or TSCH style manager, and the mode descriptor can be piggybacked on periodic management beacons. Because activation is communicated as a bitmap, devices do not need per cell reconfiguration messages during every transition.

b) Choosing activation and LC budget parameters.: Operators can tune the activation budget B and the LC budget (b_i, W) to match process priorities. A larger B improves robustness to unexpected interference at the cost of short term airtime, while a larger LC budget increases monitoring freshness during stress but reduces the reclaimable pool available to protect HC loops. The 15 node testbed results in Fig. 8 suggest that a relatively small B already captures most of the gain because the gateway can target the most fragile hops first.

c) When payload augmentation helps most.: Chunk interleaving is most effective when losses are driven by bursty collisions or narrowband interference that corrupt a contiguous region of the packet body. If losses are dominated by whole packet erasures, REMIX still relies on extra opportunities from the contingency pool, reverting toward time redundancy only where needed. Payload augmentation therefore complements, rather than replaces, scheduler level protection.

d) Scope of the current evidence.: The 15 node indoor deployment validates mode switching, reclamation under LC budgets, and the software recovery path on real hardware. We

TABLE II
EXPERIMENTAL RESULTS AT THE COMBINED STRESS POINT. SD = STATICDUP, RP = REDPATH, DL = DROPLC, AND NA = REMIXNOAUG.

Metric	SD	RP	DL	NA	ReMix
HC deadline (%)	66.1	85.0	81.1	84.2	**87.5**
HC reliability (%)	74.4	91.1	88.3	89.8	**93.0**
LC delivery (%)	54.8	56.1	0.0	61.4	**64.1**
HC slots	59.7	56.5	49.5	40.4	**35.5**
Energy (mJ)	3.26	2.50	2.70	2.41	**2.22**

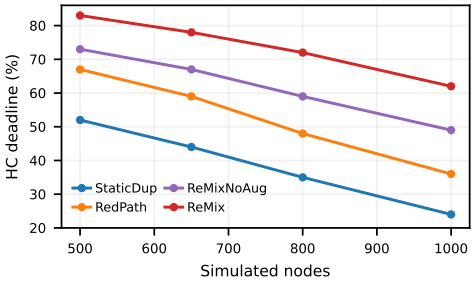


Fig. 10. Simulation: HC deadline satisfaction from 500 to 1000 nodes.

use two separate 500 to 1000 node simulation figures only to examine scaling beyond that room scale deployment.

X. CONCLUSION

We presented REMIX, a mixed criticality design for industrial WSANs that preserves safety loop timeliness during stressed intervals without permanently inflating the schedule. REMIX combines a stable base schedule and a compact contingency pool, epoch latched mode switching, reclaiming of LC redundancy under an explicit LC service budget, and payload augmented HC transmissions that enable software reconstruction without PHY changes. On a 15 node TelosB indoor testbed with a MacBook M4 Pro gateway, REMIX improves HC deadline satisfaction from 32.9% to 85.9% under the heaviest stress point, maintains 63.4% LC delivery during burst stressed HC intervals, reduces active HC mode slotframe length by 40.9% at 40% HC traffic, and lowers experiment derived radio energy per delivered HC packet by 31.9% compared with static duplication. Two separate 500 to 1000 node simulations further suggest that the same efficiency trend persists beyond the current hardware scale. Future work will extend REMIX to support more than two criticality levels by associating each level with its own activation budget and pool slice, and will add direct current measurements to strengthen the energy analysis.

ACKNOWLEDGMENT

The work was supported by the US National Science Foundation through grants CNS-2601685 and CNS-2602744, and by the US Office of Naval Research through grant N00014-23-1-2151.

REFERENCES

[1] HART Communication Foundation, "WirelessHART specification," 2007.
[2] ISA, "ISA100.11a," available online.
[3] I. E. C. IEC/PAS 62601, "Industrial communication networks – WIA-PA communication profile," 2011.
[4] IEEE, "IEEE Standard for Low-Rate Wireless Networks," IEEE Std 802.15.4, 2020.

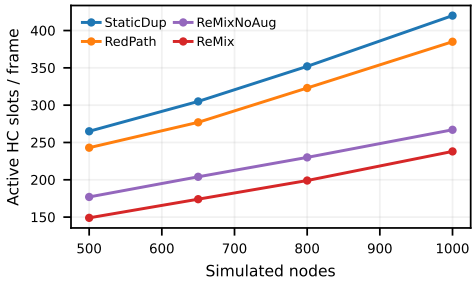


Fig. 11. Simulation: HC slotframe growth from 500 to 1000 nodes.

[5] T. Watteyne, M. Vucinic, and J. Vilajosana, "An architecture for IPv6 over the TSCH mode of IEEE 802.15.4e," RFC 9030, 2021.
[6] Q. Wang, X. Vilajosana, and T. Watteyne, "Minimal scheduling function (MSF) for IPv6 over the TSCH mode of IEEE 802.15.4e," RFC 9033, 2021.
[7] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Real-Time Scheduling for WirelessHART Networks," in *Proc. IEEE RTSS*, 2010, pp. 150–159.
[8] A. Saifullah, Y. Xu, C. Lu, and Y. Chen, "Distributed channel allocation protocols for wireless sensor networks," *IEEE TPDS*, vol. 25, no. 9, pp. 2264–2274, 2014.
[9] C. Wu, D. Gunatilaka, A. Saifullah, M. Sha, P. B. Tiwari, C. Lu, and Y. Chen, "Maximizing network lifetime of WirelessHART networks under graph routing," in *Proc. IoTDI*, 2016, pp. 176–186.
[10] P. Modekurthy, A. Saifullah, and S. Madria, "Distributed graph routing for WirelessHART networks," in *Proc. ICDCN*, 2018.
[11] X. Jin, A. Saifullah, C. Lu, and P. Zeng, "Real-time scheduling for event- and time-triggered flows in industrial WSANs," in *Proc. IEEE INFOCOM*, 2019, pp. 1684–1692.
[12] P. Modekurthy, A. Saifullah, and S. Madria, "A distributed real-time scheduling system for industrial wireless networks," *ACM Trans. Embed. Comput. Syst.*, vol. 20, no. 5, 2021.
[13] S. Vestal, "Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance," in *Proc. IEEE RTSS*, 2007, pp. 239–243.
[14] S. Baruah, V. Bonifaci, G. D'Angelo, H. Li, A. Marchetti-Spaccamela, N. Megow, and L. Stougie, "Scheduling real-time mixed-criticality jobs," *IEEE Trans. Comput.*, vol. 61, no. 8, pp. 1140–1152, 2012.
[15] A. Burns and R. Davis, "A survey of research into mixed criticality systems," *ACM Comput. Surv.*, vol. 50, no. 6, 2017.
[16] X. Jin, C. Xia, H. Xu, J. Wang, and P. Zeng, "Mixed-criticality scheduling for industrial wireless sensor networks," *Sensors*, vol. 16, no. 9, 2016.
[17] C. Xia, X. Jin, L. Kong, and P. Zeng, "Bounding the demand of mixed-criticality industrial wireless sensor networks," *IEEE Access*, 2017.
[18] X. Jin, J. Wang, and P. Zeng, "End-to-end delay analysis for mixed-criticality WirelessHART networks," *IEEE/CAA J. Autom. Sinica*, vol. 2, no. 3, pp. 282–289, 2015.
[19] A. Burns, J. Harbin, L. Indrusiak, I. Bate, R. Davis, and D. Griffin, "AirTight: resilient wireless communication for mixed-criticality systems," in *Proc. IEEE RTCSA*, 2018, pp. 65–75.
[20] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Glossy: A network flooding mechanism for low-power wireless networks," in *Proc. SenSys*, 2011.
[21] F. Ferrari, M. Zimmerling, and L. Thiele, "Low-Power Wireless Bus," in *Proc. SenSys*, 2012.
[22] J. Lu and K. Whitehouse, "Exploiting the capture effect for low-latency flooding in wireless sensor networks," in *Proc. SenSys*, 2008.
[23] S. Gollakota and D. Katabi, "ZigZag decoding: combating hidden terminals in wireless networks," in *Proc. SIGCOMM*, 2008, pp. 159–170.
[24] Moteiv Corporation, "Tmote Sky: low power wireless sensor module," data sheet, Feb. 2006.
[25] E. Baccelli, O. Hahm, H. Petersen, and K. Schleiser, "RIOT: an open source OS for low-end embedded IoT devices," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4428–4440, 2018.
[26] P. Modekurthy, M. Rahman, and A. Saifullah, "Towards mixed-criticality industrial WSAN," in *Proc. ICDCN Workshops*, 2023.