Pay-by-Wire ZIP Ledger: Architectural Design and Cryptographic Implementation of a Localized Layer 2 Clearinghouse

1. Systemic Diagnosis and the Imperative for Localized Clearing

The contemporary macroeconomic architecture governing regional micro-economies is defined by the continuous, unmonitored extraction of liquid capital. This extraction is systematically driven by centralized digital commerce platforms, automated financial intermediation, and distant corporate consolidation. Digital payment infrastructure inherently acts as a one-way monetary siphon. When discretionary spending shifts from locally rooted business ecosystems to centralized e-commerce platforms, national conglomerates, and Layer 1 card networks, the localized velocity of money drops precipitously. Capital that previously recirculated through local payrolls, municipal services, regional supply chains, and community infrastructure leaves the geographic boundary within hours of a transaction's execution.

Furthermore, global liquidity fragmentation exacerbates this inefficiency. Trillions of dollars sit idle in correspondent banking relationships and siloed clearinghouses worldwide, as intermediary banks maintain prefunded accounts for settlement in different currencies and jurisdictions. The Bank for International Settlements estimates that more than $27 trillion sits idle in these accounts globally. This duplication of liquidity requirements drives up transaction costs, turning systemic stability into economic stagnation.

To neutralize this extraction and resolve liquidity fragmentation at the community level, this report details the theoretical architectural design of the "Pay-by-Wire ZIP Ledger." Operating within the geographic boundaries of Cedar Mill, Oregon (ZIP code 97229), the ledger functions as an autonomous, privacy-preserving Layer 2 clearinghouse. It intercepts and settles community transactions prior to external banking aggregation, decoupling physical citizen identity from purchasing power. By relying on zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARKs), Rate-Limiting Nullifiers (RLN), and ephemeral session hashing, the system retains regional liquidity while generating a verifiable, Write-Once-Read-Many (WORM) compliant municipal audit trail.

2. Mathematical Modeling of Local Capital Retention

The strategic goal of the Pay-by-Wire ZIP Ledger is to establish a bounded realignment of capital by hard-bounding discretionary money velocity within defined spatial units. Rather than managing unscalable macroeconomic regions, the architecture treats individual ZIP codes as discrete operational boundaries governed by the Investor Priced Index (IPI) framework.

2.1 Spatial Quantity Theory and Velocity Decay Dynamics

Standard macroeconomic models assume uniform national money velocity. However, the localized ledger models spatial monetary velocity across discrete ZIP codes. The Spatial Quantity Theory Equation is expressed as:

$$M_{z} \cdot V_{z}(t) = P_{z} \cdot Q_{z} + E_{z}(t)$$

Where M_{z} is the total liquid monetary stock inside the ZIP code z, $V_{z}(t)$ is the internal transactional velocity, $P_{z} \cdot Q_{z}$ is the localized physical trade value, and $E_{z}(t)$ represents the net external capital extraction vector. The external capital extraction is mathematically modeled as:

$$E_{z}(t) = [s_{d}(t) \cdot (1 - \gamma) + s_{c}(t) \cdot (1 - \beta) + \eta_{banking}] \cdot R_{gross}(t)$$

Here, s_{d} is the proportion of pure digital platform extraction, s_{c} is national chain retail extraction, and $\eta_{banking}$ represents centralized interest and fee siphoning. As digital extraction approaches 50% ($s_{d}(t) \rightarrow 0.50$), the external extraction exceeds local capital inflows. This causes the internal transactional velocity to decay exponentially according to the function:

$$V_{z}(t) = V_{0} \cdot e^{-\lambda \cdot s_{d}(t) \cdot t}$$

To maintain economic equilibrium, inbound capital must balance local leakage. When digital extraction exceeds 25.0% ($s_{d} > 0.250$), the system faces systemic local insolvency unless offset by external debt.

2.2 The Keynesian Local Multiplier and Recirculation Targets

The localized recirculation rate (ρ_{z}) represents the fraction of capital retained in the local economy after a single transaction round. Empirical retention coefficients allocate 52.0% local retention for independent businesses ($\alpha = 0.520$), 13.6% for national chains ($\beta = 0.136$), and a mere 3.8% for pure digital extraction ($\gamma = 0.038$). The internal Keynesian Local Multiplier (M_L) is derived from the infinite geometric series representing successive spending rounds:

$$M_{L} = 1 + \rho_{z} + \rho_{z}^{2} + \rho_{z}^{3} + \dots = \sum_{k=0}^{\infty} \rho_{z}^{k} = \frac{1}{1 - \rho_{z}}$$

To maintain a stable local capital stock where the continuous rate of change is non-negative, the system targets a benchmark recirculation rate of $\rho_{z} \ge 0.4142$, which yields a targeted internal multiplier of $M_{L} \ge 1.7071\times$.

The IPI Valuation Dial applies these metrics dynamically across different municipal density classifications, enforcing behavioral penalties on entities that extract capital beyond the specified thresholds.

ZIP Classification	Density	Production Baseline	s_d Limit	S_l Target	M_L Target	Structural Action
Zone Alpha (Urban Core)	>8,000/\text{mi}^2	Commercial & Tech Services	52.4%	75.0%	1.71x	Enforce Marked Local Currency (MLC) boundaries on retail delivery.

ZIP Classification	Density	Production Baseline	s_d Limit	S_I Target	M_L Target	Structural Action
Zone Beta (Suburban)	2,000-8,000/\text{mi}^2	Light Assembly & Residential	41.8%	70.0%	1.63x	Consolidate commercial corridors around core services.
Zone Gamma (Rural)	<2,000/\text{mi}^2	Agriculture & Energy Production	18.2%	85.0%	1.82x	Rebalance equity into equipment electrification.

Table 1: Micro-Ecosystem Vector Matrix across discrete operational zones, guiding the deployment of Marked Local Currency (MLC).

3. Cryptographic Primitives and the Privacy Engine

To satisfy the dual mandate of preventing local double-spending while preserving absolute citizen anonymity, the network utilizes a sophisticated array of cryptographic primitives. The architecture decouples the physical identity of the citizen from their purchasing power, ensuring compliance with data minimization frameworks while establishing verifiable liquidity.

3.1 zk-SNARK Selection: Groth16 Verification Dynamics

For the foundational proof system, the architecture must balance proof generation time, proof size, and verifier computational overhead. Zero-knowledge proofs (ZKPs) allow a prover to convince a verifier that a statement is true without disclosing any information beyond the validity of the statement itself. While zk-STARKs offer post-quantum security and eliminate the need for a trusted setup, their proof sizes are substantially larger, making them less suitable for high-throughput, latency-sensitive retail environments.

Between the two dominant zk-SNARK protocols—Groth16 and Plonk—Groth16 is selected for the clearinghouse's execution environment. Groth16 requires a circuit-specific trusted setup, but it produces highly succinct proofs comprising just two \mathbb{G}_1 elements and one \mathbb{G}_2 element, totaling approximately 0.13kB in size. By contrast, Plonk proofs are considerably larger, requiring roughly 0.51kB (nine \mathbb{G}_1 elements and seven field elements). More critically, Groth16 verification logic is highly optimized. Benchmarks demonstrate that Groth16 verification consumes approximately 203,000 to 230,000 gas in Ethereum Virtual Machine (EVM) equivalent execution environments. Because municipal payment throughput requires high-velocity local processing at Point of Sale (POS) terminals, the minimal verification latency and O(1) verification cost of Groth16 are optimal for Layer 2 interception.

3.2 Semaphore V4 and LeanIMT Identity Commitments

The system leverages the Semaphore V4 protocol to enable citizens to prove membership within the ZIP Code Ledger network without revealing which specific wallet or identity they control. Semaphore V4 identities utilize the EdDSA signature scheme implemented over the

Baby Jubjub elliptic curve, chosen for its efficiency inside arithmetic circuits.
An identity serves as the unique identifier and contains a private key, from which a secret scalar is derived. To avoid exposing the public key directly, the public Semaphore identity—the identity commitment—is generated using the Poseidon hash function:
Commitment_{id} = Poseidon(PublicKey)
These commitments are appended as leaves to a LeanIMT (Optimized Binary Incremental Merkle Tree) maintained by the local ZIP node. The LeanIMT minimizes computational costs while providing strong security guarantees. When a citizen initiates a local transaction, their device locally generates a Merkle proof demonstrating they control the private key corresponding to a valid leaf in the LeanIMT, without revealing the specific leaf.

3.3 Rate-Limiting Nullifiers (RLN) and Sybil Resistance

While Semaphore guarantees anonymity by obfuscating the user within the anonymity set of the Merkle tree, it introduces the risk of double-spending or spam. In a completely anonymous environment, a user could theoretically authorize multiple transactions utilizing the same proof. To counter this, the ledger implements Rate-Limiting Nullifiers (RLN), a zero-knowledge gadget designed for spam prevention in anonymous peer-to-peer environments utilizing Shamir's Secret Sharing (SSS).
In the RLN construct, a user's private key is treated as the y-intercept (a_0) of a polynomial. During a specific time window or transaction context (an "epoch"), the user derives an external nullifier and computes a first-degree polynomial:
a_1 = Poseidon(a_0, external_nullifier, message_id) y = a_0 + a_1 \cdot x
The variable x represents the Poseidon hash of the transaction payload (the signal or message). For every transaction, the user publishes the share (x, y) and an internal nullifier.
If a user attempts to double-spend or exceed their rate limit within the same epoch, they must generate a second proof. Because the external nullifier remains constant for the epoch, the coefficient a_1 remains identical. Consequently, the user broadcasts a second point (x_2, y_2) on the exact same polynomial. Because any two points uniquely define a first-degree line, network watchtower nodes can immediately reconstruct the polynomial, calculate the y-intercept (a_0), cryptographically expose the user's private key, and automatically slash their staked localized liquidity.
A critical security consideration in the implementation of the RLN circuit (written in the Circom circuit description language) is the prevention of zero-value inputs. If the hashed message input x is allowed to equal zero, the polynomial equation simplifies to y = a_0, immediately revealing the user's private identity secret without them violating the rate limit. The architecture strictly enforces a greater_than constraint outside the circuit to ensure x > 0, maintaining mathematical confidentiality.

Circuit Parameter	Type	Description / Implementation
DEPTH	Template Param	The depth of the LeanIMT Merkle Tree (typically 20 for large populations).
identitySecret	Private Input	The randomly generated scalar (a_0) acting as the private key.
userMessageLimit	Private Input	The predefined maximum transaction count per epoch.
x	Public Input	Hash(signal), representing the

Circuit Parameter	Type	Description / Implementation
		specific transaction payload.
externalNullifier	Public Input	Hash(epoch, rln_identifier), binding the proof to the time window.
y	Output	The evaluated share of the polynomial y = a_0 + a_1 \cdot x.
nullifier	Output	Internal nullifier used to flag duplicate shares within the mempool.

Table 2: Core parameters and constraints of the Circom RLN Zero-Knowledge Circuit utilized by the clearinghouse.

3.4 Ephemeral Session Hashing for Ingress Telemetry

To completely blind IP footprints and physical network telemetry from the transaction data, the system salts and hashes physical routing data at the point of ingress. As mobile devices and POS terminals connect to the local node, the ingress gateway computes: Session_Hash = Hash(IP_Address + Salt) Ephemeral_ID = Hash(Raw_User_ID + Salt) These transient identifiers are mapped securely within the municipal event log, effectively blinding the data prior to settlement. This ensures that even if the ledger is publicly audited, it remains mathematically impossible for tech aggregators, data brokers, or advertising networks to harvest, profile, or monetize citizen purchasing habits based on device telemetry.

4. Network Architecture: Layer 2/3 Ingress and Interception

The core structural advantage of the Pay-by-Wire ZIP Ledger is its ability to operate as an intercepting proxy between the merchant's physical Point of Sale (POS) hardware and national Layer 1 payment rails (e.g., Visa, Mastercard, Stripe). By intercepting transactions at the application layer, the ledger circumvents the traditional correspondent banking system for local commerce.

4.1 ISO 8583 Point of Sale (POS) Interception

Traditional merchant terminals and ATM networks communicate via the ISO 8583 standard, transmitting electronic financial transaction messages across TCP/IP sockets. ISO 8583 is a highly structured, binary-encoded protocol that predates modern RESTful JSON architectures. An ISO 8583 message relies on three primary structural components: the Message Type Indicator (MTI), a Bitmap, and the Data Elements (DE).

- **Message Type Indicator (MTI):** The MTI is a four-digit numeric field indicating the overall function of the message (version, class, function, origin). For example, MTI 0100 designates an Authorization Request, while 0200 designates a Financial Transaction Request.
- **Bitmap Decoding:** The protocol utilizes a 64-bit primary bitmap (and an optional

secondary bitmap) to indicate exactly which of the 128 possible data elements are present in the message. Each bit set to 1 signifies the presence of a corresponding field. This allows the message payload to remain highly compressed.

- **Data Elements (DE):** The payload itself is contained in standardized fields.

Data Element (DE)	Field Name	Format	Usage in Interception Gateway
DE 2	Primary Account Number (PAN)	Variable, max 19	Truncated/Masked by proxy; bypassed for local zk-settlement.
DE 3	Processing Code	Fixed, 6 digits	Identifies transaction type (e.g., 00 for purchase).
DE 4	Transaction Amount	Fixed, 12 digits	Extracted to determine the local liquidity requirement.
DE 11	Systems Trace Audit Number (STAN)	Fixed, 6 digits	Used for localized idempotency and duplicate detection.
DE 39	Response Code	Fixed, 2 chars	Set to 00 (Approved) by the local node upon successful zk-proof.
DE 41	Card Acceptor Terminal ID (TID)	Fixed, 8 chars	Verified against the 97229 ZIP code merchant registry.
DE 42	Card Acceptor Identification Code	Fixed, 15 chars	Verified against the 97229 ZIP code merchant registry.

Table 3: Critical ISO 8583 Data Elements parsed by the localized interception gateway.

The localized clearinghouse operates a proxy gateway (similar in architecture to middleware like SocketQueue) that ingests the ISO 8583 binary payloads broadcast by the merchant's terminal. The proxy reads the MTI and the Bitmap, extracts DE 4 (Amount), DE 41 (Terminal ID), and DE 42 (Merchant ID), and translates this payload into a JSON structure digestible by the zero-knowledge verification circuit.

4.2 Geofenced Settlement and Marked Local Currency (MLC)

Once the ISO 8583 intent is parsed into JSON, the gateway queries the consumer's mobile application. The application generates the Groth16 zk-SNARK proof and RLN share, proving that the user holds sufficient liquidity on the local ledger to cover the DE 4 amount.

The clearing node verifies the cryptographic proof and checks the geographic origin. If both the sending consumer and the receiving merchant (verified via DE 41/42 mapping) reside within the 97229 ZIP boundary, the transaction is completely diverted from the Layer 1 card network. It settles instantly via Marked Local Currency (MLC) on the Layer 2 ledger.

Because the transaction is cleared locally, the proxy immediately formats a forged ISO 8583 MTI 0210 (Financial Transaction Response) with DE 39 set to 00 (Approved) and returns it to the physical POS terminal. This intra-ZIP routing circumvents traditional interchange fees (often

1.5% to 3.0%), eliminating external capital extraction and retaining the local multiplier effect.

4.3 External Routing and ISO 20022 Integration

If a transaction is destined for an entity outside the geographic boundary (e.g., an e-commerce purchase to a national conglomerate), the ZIP ledger node acts as a pass-through. The transaction is routed to traditional Automated Clearing House (ACH) rails or Layer 1 networks. For modern interbank settlements, the gateway maps the outgoing transaction to the ISO 20022 XML messaging standard, specifically utilizing the pacs.008 (FI-to-FI Customer Credit Transfer) message family. ISO 20022 utilizes a hierarchical data dictionary that supports richer structured data compared to the legacy fixed-width fields of ISO 8583, allowing the localized clearinghouse to communicate seamlessly with national Federal Reserve Fedwire or CHIPS networks for net-balance batch clearing.

4.4 Hardware Enclave Security

To ensure that the localized clearing nodes cannot be compromised to extract the underlying private keys or de-anonymize the mempool, the node software is executed within Trusted Execution Environments (TEEs) such as Intel SGX enclaves. SGX enclaves provide hardware-level memory encryption, ensuring that even if the physical node (which can be deployed on lightweight, cost-effective hardware like a Raspberry Pi) is seized or compromised by an adversary, the cryptographic state and mempool telemetry remain unreadable to the host operating system.

5. Municipal Record-Keeping and WORM Compliance

While the identities of the transacting parties are cryptographically hidden, public auditability of municipal capital flows is critical for determining the Keynesian multiplier (\rho_z) and maintaining economic transparency. The architecture employs a Write-Once-Read-Many (WORM) compliant event log to structure these records securely.

5.1 Event Log Schema

The municipal settlement log utilizes the ephemeral data extracted during the ingress phase. The schema records:

1. **Timestamp (UTC)**
2. **Ephemeral User ID** (The zk-derived pseudonym)
3. **Session IP Hash**
4. **ZIP Boundary / Destination Vector**
5. **Amount Settled (in MLC)**
6. **Network Fee / Interchange Savings**
7. **Settlement Status** (Cleared Local vs. Routed to External ACH)

Because the event log relies exclusively on transient identifiers (the zk-ID and salted IP hash), it provides a robust macroeconomic dataset for municipal planners while remaining mathematically impossible for data brokers to harvest individual purchasing habits.

5.2 Storage Immutability via AWS S3 Object Lock

To guarantee WORM compliance without incurring the severe latency and gas costs associated with committing every microtransaction to a public Layer 1 blockchain, the node relies on enterprise-grade immutable object storage, utilizing AWS S3 Object Lock.
Object Lock prevents data from being altered or deleted by any person or process, satisfying the rigorous data retention requirements of SEC Rule 17a-4, CFTC, and FINRA. When the municipal ledger writes an hourly JSON batch file of anonymized transactions to the S3 bucket, it utilizes S3 Object Lock configured in *Compliance Mode*. In Compliance Mode, the retention period is absolute; not even the root AWS account holder can overwrite or delete the object until the fixed retention period expires.
For long-term public auditing disputes or investigations, authorized municipal auditors can apply a "Legal Hold" to specific WORM objects via API. A Legal Hold provides the same immutability protection as a retention period but possesses no expiration date, remaining in effect until explicitly removed by an authorized user, thereby ensuring permanent cryptographic evidence of municipal capital flows.

6. Streamlit Interactive Dashboard Blueprint

To monitor the economic telemetry of the 97229 ZIP Code Micro-Ecosystem, the architecture includes a Streamlit-based interactive dashboard. The dashboard visualizes the core metrics of the IPI framework in real time, specifically tracking Local Capital Retained versus External Liquidity Leakage.

6.1 Dashboard UI Layout

The UI layout utilizes columns and real-time dataframes to provide an at-a-glance status panel monitoring peer-to-peer settlement throughput.

- **Telemetry Row:** Displays high-level KPIs including the total volume of Marked Local Currency (MLC) retained locally, the volume leaked externally, the aggregate network fees avoided, and the count of active zk-nodes in the local mempool.
- **WORM-Compliant Log Table:** A rendered pandas dataframe that actively filters the transient identifiers, displaying color-coded vectors for "Cleared Local" (green) versus "Routed to External ACH" (red).

6.2 Streamlit Prototype Implementation

The following Python scaffolding models the anonymized telemetry and calculates the mathematical retention metrics in real time:

```
import streamlit as st
import pandas as pd
import numpy as np
import hashlib
import time

st.set_page_config(page_title="Pay-by-Wire ZIP Ledger", layout="wide", page_icon="🛡️")
st.title("🛡️ Pay-by-Wire ZIP Ledger: Local Clearing Gateway")
```

```
def generate_zk_proof(ip_address, raw_id):
    """Simulates decoupling citizen IP/PII into an ephemeral cryptographic pseudonym."""
    salt = str(time.time()).encode()
    session_hash = hashlib.sha256(ip_address.encode() + salt).hexdigest()[:12]
    ephemeral_id = hashlib.blake2s(raw_id.encode() + salt).hexdigest()[:16]
    [span_16](start_span)[span_16](end_span)[span_31](start_span)[span_31](end_span)return
f"zk-{ephemeral_id}", f"ip-hash-{session_hash}"

@st.cache_data(ttl=5)
def fetch_local_mempool():
    np.random.seed()
    data = []
    for i in range(15):
        zk_id, ip_hash = generate_zk_proof(f"192.168.1.{np.random.randint(1,255)}", f"user_{i}")
        amount = np.round(np.random.uniform(5.0, 500.0), 2)
        is_local = np.random.choice([True, False], p=[0.8, 0.2])
        status = "Cleared Local" if is_local else "Routed to External ACH"
        fee = 0.00 if is_local else np.round(amount * 0.03, 2)

        data.append({
            "Timestamp (UTC)": pd.Timestamp.utcnow().strftime("%Y-%m-%d %H:%M:%S"),
            "Ephemeral User ID": zk_id,
            "Session IP Hash": ip_hash,
            "ZIP Boundary": "97229" if is_local else "External Export",
            "Amount (MLC)": amount,
            "Network Fee": fee,
            "Settlement Status": status
        })
    return pd.DataFrame(data)

df = fetch_local_mempool()

local_retention = df[df["Settlement Status"] == "Cleared Local"]["Amount (MLC)"].sum()
external_leakage = df[df["Settlement Status"] != "Cleared Local"]["Amount (MLC)"].sum()

col1, col2, col3, col4 = st.columns(4)
col1.metric("Local Capital Retained", f"${local_retention:,.2f}", "+ 80% Target")
col2.metric("External Liquidity Leakage", f"${external_leakage:,.2f}", "- Extracted")
col3.metric("Network Fees Avoided", f"${local_retention * 0.03:,.2f}")
col4.metric("Active zk-Nodes", len(df))

st.markdown("#### Real-Time Municipal Settlement Log (WORM Compliant)")
st.dataframe(
    df.style.apply(lambda x: ['background: #e6ffe6' if v == 'Cleared Local' else 'background:
#ffe6e6' for v in x], subset=['Settlement Status']),
    use_container_width=True,
    hide_index=True
```

)

7. Regulatory, Legal, and Security Audit

A localized Layer 2 clearinghouse that circumvents national payment rails fundamentally intersects with extensive federal banking regulations, privacy legislation, and constitutional frameworks. The architecture is engineered explicitly to remain compliant with strict legal parameters while maximizing user anonymity.

7.1 Bank Secrecy Act (BSA) and FinCEN Exemption Thresholds

Under the Bank Secrecy Act (BSA), entities facilitating the transfer of funds are typically classified as Money Services Businesses (MSBs), necessitating rigorous Anti-Money Laundering (AML) and Know Your Customer (KYC) reporting to the Financial Crimes Enforcement Network (FinCEN).

However, the Pay-by-Wire ZIP Ledger is specifically architected to leverage FinCEN's regulatory exemptions regarding "closed-loop prepaid access." Under 31 CFR 1022, FinCEN sets a specific dollar threshold of $2,000 for closed-loop prepaid access networks. Because the ledger operates geographically within a single ZIP code and intra-community transactions clear strictly via the internal Marked Local Currency (MLC) token, consumer wallets operating under the $2,000 daily threshold are legally classified as low-risk and are largely insulated from stringent federal AML surveillance requirements.

Merchants, conversely, undergo standard commercial KYC onboarding to host the local gateway terminal (verifying the DE 41 and DE 42 mapping), ensuring commercial tax liability and fraud prevention are thoroughly managed. Federal AML tracking is only triggered if a consumer attempts to bridge large MLC balances back onto centralized fiat rails, converting the transaction from an exempt closed-loop exchange into a regulated open-loop export.

7.2 Oregon Consumer Privacy Act (OCPA) vs. Data Minimization

At the state level, the Oregon Consumer Privacy Act (OCPA) grants citizens expansive rights to control, access, and delete their personal data, making it one of the broadest consumer privacy laws in the United States. The law applies to entities processing the personal data of Oregon residents, demanding strict compliance regarding data harvesting.

The Pay-by-Wire ZIP Ledger inherently satisfies and exceeds the requirements of the OCPA through radical cryptographic data minimization. By processing exclusively via zk-SNARK zero-knowledge proofs and transient Session IP Hashing, the system never ingests, stores, or processes Personally Identifiable Information (PII) at the ledger level. The WORM-compliant audit trail contains zero correlative data that could be legally classified as consumer personal data under ORS 646A.570. Consequently, the municipal database cannot be breached for PII, resolving the burden of fulfilling consumer data deletion requests, as there is mathematically no PII retained by the system to delete.

7.3 Constitutional Adherence: The Coinage Clause

The implementation of Marked Local Currency (MLC) strictly navigates Article I, Section 10 of

the United States Constitution (the Coinage Clause), which mandates that "No State shall make any Thing but gold and silver Coin a Tender in Payment of Debts".
By this constitutional mandate, a state or municipality is expressly prohibited from issuing its own sovereign fiat currency or forcing creditors to accept an alternative medium of exchange to satisfy a debt. Therefore, MLC is legally classified as a voluntary community credit facility or closed-loop exchange token, rather than state-mandated legal tender. Merchants voluntarily agree to accept MLC for the clearance of goods in exchange for the elimination of the standard 3% interchange fee. This maintains compliance with the Coinage Clause through mutual private contract; as observed in *Berry v. Hannigan*, regulations regarding payment acceptance methods do not violate the legal tender clause if the debt is discharged via third-party institutional agreements without forcing the acceptance of non-legal tender against a party's will.

7.4 The Dormant Commerce Clause and Market Participant Exception

A final potential legal vector of attack against a municipality operating a geofenced payment system is the Dormant Commerce Clause. Derived from Article I, Section 8, the Dormant Commerce Clause prohibits state and local governments from enacting legislation that discriminates against or places an undue burden on interstate commerce. Bounding local capital to a single ZIP code and actively bypassing national credit card networks could be viewed as a regulatory barrier to interstate trade.
To circumvent this constitutional hurdle, the architecture relies heavily on the "Market Participant Exception." The Supreme Court, in cases such as *Hughes v. Alexandria Scrap Corp.*, established that when a state or municipality acts as a direct participant in the market (e.g., as a buyer, seller, or service provider) rather than as a market regulator, the restrictions of the Dormant Commerce Clause do not apply. Under this exception, the municipality of Cedar Mill can preferentially disburse its own public grants, municipal payrolls, and community funding via the Pay-by-Wire ZIP Ledger, and can restrict the settlement of those specific funds to in-ZIP economic actors without violating federal commerce protections. The ledger operates as a voluntary public utility, ensuring that the municipality acts strictly as a market participant fostering local industry, rather than a regulator restricting private interstate trade.

Works cited

1. How Liquidity Fragmentation Holds Back Global Payments - Circle, https://www.circle.com/blog/why-liquidity-fragmentation-holds-back-global-payments 2. Fragmentation, Clearing, and the Future of Repo Markets - OSTTRA, https://osttra.com/articles/us-treasury-repo-clearing-fragmentation/ 3. , https://drive.google.com/open?id=14L7ur1hb9N5ySv6Dsi2vrx2-I67YtMJR0TrWJJPoyIk 4. Amazon S3 Object Lock - AWS, https://aws.amazon.com/s3/features/object-lock/ 5. A Hitchhiker's Guide to Privacy-Preserving Digital Payment Systems, https://arxiv.org/html/2505.21008v2 6. A Comparative Analysis of zk-SNARKs and zk-STARKs - arXiv, https://arxiv.org/html/2512.10020v1 7. [2512.10020] A Comparative Analysis of zk-SNARKs and zk-STARKs, https://arxiv.org/abs/2512.10020 8. PLONK Benchmarks I — 2.5x faster than Groth16 on MiMC, https://aztec.network/blog/plonk-benchmarks-i----2-5x-faster-than-groth16-on-mimc 9. B5GRoam: A Zero Trust Framework for Secure and Efficient ... - arXiv, https://arxiv.org/html/2509.16390v2 10. zk-X509: Privacy-Preserving On-Chain Identityfrom Legacy PKI via, https://arxiv.org/html/2603.25190v2 11. A ZKP-Native Blockchain Runtime with

Sub-Second Cryptographic, https://arxiv.org/pdf/2603.10242 12. zkspecs/specs/3/README.md at main - GitHub, https://github.com/zkspecs/zkspecs/blob/main/specs/3/README.md 13. Semaphore RLN, rate limiting nullifier for spam prevention in, https://ethresear.ch/t/semaphore-rln-rate-limiting-nullifier-for-spam-prevention-in-anonymous-p2p-setting/5009 14. Rate-Limiting-Nullifier/rln-circuits - GitHub, https://github.com/Rate-Limiting-Nullifier/rln-circuits 15. Rate-Limiting Nullifier (RLN) - GitHub, https://github.com/Rate-Limiting-Nullifier 16. Formal spec of circom-rln - Rate-Limiting Nullifier, https://rate-limiting-nullifier.github.io/rln-docs/formal_spec.html 17. yAudit/RLN-audit-report - GitHub, https://github.com/yAudit/RLN-audit-report 18. Payment Technology — ISO 8583 & ISO 20022 - FinPay Consultants, https://finpayconsultants.com/services/technology/ 19. ISO 8583: The Technical Foundation of Modern Payment Systems, https://isaaclanre.medium.com/iso-8583-the-technical-foundation-of-modern-payment-systems-33af3fbaee68 20. How to Connect a Modern Microservice to an ISO8583 Payment, https://neapay.com/post/how-to-connect-a-modern-microservice-to-an-iso8583-payment-network-without-writing-a-single-line-of-protocol-code_136.html 21. ISO 8583 - Wikipedia, https://en.wikipedia.org/wiki/ISO_8583 22. verygoodsecurity/SocketQueue: ISO 8583 gateway for ... - GitHub, https://github.com/verygoodsecurity/SocketQueue 23. ISO 8583: Definition, How It Works - PXP, https://www.pxp.io/payments-glossary/iso-8583 24. What is ISO 20022? Modern Payment Messaging Standard (2026), https://codegotech.com/glossary/iso-20022 25. SWIFT Cross-Border Payments Migration to ISO20022 Benefits, https://business.bofa.com/en-us/content/iso-20022-migration.html 26. Why No Consensus on Consensus? A Deep Dive Into Blockchain, https://digital-library.theiet.org/doi/full/10.1049/blc2.70050 27. A Survey of Security Strategies in Federated Learning - MDPI, https://www.mdpi.com/1999-5903/16/10/374 28. Interpol review of digital evidence for 2019–2022 - PMC, https://pmc.ncbi.nlm.nih.gov/articles/PMC10311174/ 29. Computer Architecture: A Quantitative Approach, https://acs.pub.ro/~cpop/SMPA/Computer%20Architecture,%20Sixth%20Edition_%20A%20Quantitative%20Approach%20(%20PDFDrive%20).pdf 30. Guide to Data Immutability: Concepts, Benefits, and Use Cases, https://hydrolix.io/glossary/immutability/ 31. SEC Rules 17a-4 and 18a-6 - Amazon Web Services (AWS), https://aws.amazon.com/compliance/secrule17a-4f/ 32. Locking objects with Object Lock - Amazon Simple Storage Service, https://docs.aws.amazon.com/AmazonS3/latest/userguide/object-lock.html 33. S3 Object Lock for Ransomware Protection, https://objectfirst.com/guides/immutability/s3-object-lock-for-ransomware-protection/ 34. Log Management Architecture and Retention for Compliance, https://www.the-algo.com/knowledge/log-management 35. S3 Object Lock Legal Holds vs. Retention Period | AWS re:Post, https://repost.aws/questions/QUSZMwke3uSu226EuuNnaVzg/s3-object-lock-legal-holds-vs-retention-period 36. 4.26.5 Bank Secrecy Act History and Law | Internal Revenue Service, https://www.irs.gov/irm/part4/irm_04-026-005 37. Bank Secrecy Act Regulations-Definitions and Other Regulations, https://www.federalregister.gov/documents/2011/07/29/2011-19116/bank-secrecy-act-regulations-definitions-and-other-regulations-relating-to-prepaid-access 38. Consumer Privacy - Oregon Department of Justice, https://www.doj.state.or.us/consumer-protection/id-theft-data-breaches/privacy/ 39. Oregon Consumer Privacy Act, https://privacyrights.org/resources-tools/law-overviews/oregon-consumer-privacy-act 40. Oregon Consumer Privacy Act (OCPA): What Businesses Need to,

https://www.ketch.com/regulatory-compliance/oregon-consumer-privacy-act-ocpa 41. Oregon OCPA Cookie Consent and Compliance - CookieHub, https://www.cookiehub.com/ocpa 42. ORS 646A.570 – Definitions - OregonLaws - Public.Law, https://oregon.public.law/statutes/ors_646a.570 43. Sound Money Index: Gold & Silver Laws in 2026, https://www.moneymetals.com/resources/sound-money-index 44. Article 1, Section 10, Clause 1: Joseph Story, Commentaries on the, https://press-pubs.uchicago.edu/founders/documents/a1_10_1s23.html 45. OGDEN v. SAUNDERS, 25 U.S. 213 (1827) - FindLaw Caselaw, https://caselaw.findlaw.com/us-supreme-court/25/213.html 46. BERRY v. HANNIGAN (1992) - FindLaw Caselaw, https://caselaw.findlaw.com/court/ca-court-of-appeal/1845792.html 47. Berry v. Hannigan (1992) - California Case Law, https://law.justia.com/cases/california/court-of-appeal/4th/7/587.html 48. The Market Participant Doctrine and the Clear Statement Rule, https://digitalcommons.law.umaryland.edu/cgi/viewcontent.cgi?article=1037&context=fac_pubs 49. Dormant Commerce Power: Overview | U.S. Constitution Annotated, https://www.law.cornell.edu/constitution-conan/article-1/section-8/clause-3/dormant-commerce-power-overview 50. The Dormant Commerce Clause and State-Mandated Preference, https://repository.law.umich.edu/cgi/viewcontent.cgi?article=3177&context=mlr 51. The Commerce Clause as a Restraint on State Powers - Justia Law, https://law.justia.com/constitution/us/article-1/39-restraint-on-state-powers.html 52. State Proprietary Activity (Market Participant) Exception | US Law, https://www.law.cornell.edu/constitution-conan/article-1/section-8/clause-3/state-proprietary-activity-market-participant-exception